UNIVERSITÉ DE GENÈVE FACULTÉ DES SCIENCES

Département d' informatique Professeur Christian Pellegrini
Docteur Melanie Hilario

Algorithm Selection via Meta-Learning

THÈSE

présentée à la Faculté des sciences de l'Université de Genève
pour obtenir le grade de Docteur ès sciences, mention informatique

par

Alexandros KALOUSIS

de

Kallipefki, Grèce

These N^o 3337

GENÈVE
2002

To my parents Gianni and Stamati
and my sister Pagona

...
Hope your road is a long one.
May there be many summer mornings when,
with what pleasure, what joy,
you enter harbours you're seeing for the first time;
may you stop at Phoenician trading stations
to buy fine things,
mother of pearl and coral, amber and ebony,
sensual perfume of every kind -
as many sensual perfumes as you can;
and may you visit many Egyptian cities
to learn and go on learning from their scholars.

Keep Ithaca always in your mind.

Arriving there is what you're destined for.
But don't hurry the journey at all.
Better if it lasts for years,
so you're old by the time you reach the island,
wealthy with all you've gained on the way,
not expecting Ithaca to make you rich.
...

Ithaca - K.Kavafis

Acknowledgments

Now almost everything has been put in place, figures, tables, contents, so I will take a bit more than a page and try to squeeze within it all these people that in one or the other way were involved in a process that started five years ago in Greece and ends today in Geneva.

First of all I would like to thank Professor Christian Pellegrini and Assistant Professor Melanie Hilario for accepting me as PHD student at the Artificial Intelligence Group of the University of Geneva. Coming to Geneva facilitated my work and brought me in contact with many people that I couldn't even dream before, for this I am twice grateful to them. Moreover I wish to thank Melanie Hilario for all her support during these years, the fact that she had always let me free to pursue my research interests and the more than lengthy scientific discussions we had.

I thank the members of my thesis committee Dr Joao Gama and Dr Michele Sebag for their insightful comments and suggestions that not only helped this work become more complete, but also provided new ideas. I really enjoyed the exam up to the point that I was disappointed when we had to leave the exam room. I would like to thank Joao Gama for one more reason, all this work started after reading one of his papers, having him in my committee was a source of great joy.

I am grateful to my friend Aris who had a great role in my decision to come in Geneva, if it was not him on a September afternoon I might have never taken that decision.

I would like to thank Assistant Professor Theoharis Theoharis of the University of Athens who offered me the possibility to first work on the Machine Learning area. Also back at the University of Athens my officemate Vassilis Drakopoulos now Dr. Drakopoulos at the time senior PHD student who, although shouting almost all of the time, was the source of great support and I have to say inspiration. I also thank Katerina who stood by me in the first years of the PHD and I am sorry for all the stress that I was passing over to her.

I thank all the members of the METAL project for their useful comments and all the discussions we had during meetings or conferences. I hope that I will get the chance to see them and talk with them often in the future.

I would like to thank all the members of the Artificial Intelligence Group and especially the former members Dr Ahmed Rida, my first officemate in Geneva, and Dr Abdel Labbi for their warm welcome, that made the transitory period disappear, but also for their answers to my questions. I want to thank Wassim for his unquestionably sexy moral support, our lengthy meetings at Ethno were the best tension killers and definitely he did a very good job with the French summary. I also thank Dimitris for being my daily smoking companion, although we should start going on the fourth floor where smoking is allowed. Last but for sure not least I want to thank Georgia who always had a word, for a long period of time over the phone, fortunately in the last months of the PHD face to face, that could take me out even from the most doubtful situations.

Contents

List of Figures

List of Tables

Abstract

The goal of this thesis is to provide support to the analyst in selecting the appropriate classification algorithm for a specific problem, taking into consideration the nature of the problem. We make no distinction between an algorithm and the representational model of the algorithm, we consider the learning algorithm as the entity to be selected. We tackle the problem of inducer selection as a typical classification problem, although at a meta-level. In a classification problem a learner is given a dataset of training instances and is required to construct an inductive model in order to predict the classes of new unseen instances. In our meta-learning framework training instances are the descriptions of datasets, and the meta-classification or else meta-learning task is the prediction of the learning algorithm that is more appropriate for a specific dataset.

We provide a novel formulation of the meta-learning space in terms of pairwise comparisons of learning algorithms. The meta-learning space simulates closely the typical process that an analyst adopts, when he has to select among different inducers. The increased flexibility that the meta-learning space offers. The incorporation of the pairwise meta-learning problems allows for a closer study of the factors affecting the relative performance of different inducers.

We proceeded to a systematic search for features that can adequately describe a dataset, with the main emphasis placed on the description of attribute interelationships. These features will constitute the predictive features of the pairwise meta-learning problems. An effort to find an appropriate way to bridge the gap between features that describe properties of continuous attributes and features that describe properties of discrete attributes. In that context we also introduced the notion of *non-appl* values, for these features whose computation does not make sense for discrete attributes and vice versa. A new representation of the features used to describe the properties of each attribute of a dataset or combinations of attributes, via the use of histograms, in an effort to depict as close as possible their distribution, without loss of discriminating information.

We undertook extensive comparisons of different ways of characterizing datasets as well as performing inducer ranking, these included: an empirical comparison of different ways of characterizing datasets in order to perform inducer selection, under two different meta-learning frameworks; an empirical comparison of different ways of characterizing datasets under a regression approach to meta-learning; the exploitation of the regression approach to predict rankings of inducers, and a comparison of different ranking methods.

The emprical evaluation of the system has shown that it can provide successful suggestions as to which learning algorithm is more appropriate for a specific dataset. The meta-learning models constructed by the inducers applied on the meta-learning problems allow us to have a better understanding of the dataset chatacteristics that affect the performance of the learning algorithms.

Sélection d'Algorithme via Meta-apprentissage

I Introduction

Le domaine de l'apprentissage automatique est en constante évolution et produit une multitude de modèles et d'algorithmes pour effectuer des tâches de classification, tels que les arbres de décision, les réseaux de neurones, les inducteurs de règles, le plus proche voisin, etc.

L'analyste doit sélectionner, parmi ces modèles et algorithmes, ceux qui correspondent le mieux à la morphologie et aux caractéristiques spéciales d'un problème donné.

Cette sélection est un problème extrêmement fastidieux étant donné qu'il n'existe pas de modèle ou d'algorithme qui ait une meilleure performance que d'autres indépendamment des caractéristiques spécifiques du problème, comme cela a été observé dans différentes comparaisons empiriques, (Aha, 1992; Salzberg, 1991; Shavlik et al., 1991; Weis & Kapouleas, 1989).

Par la suite, les résultats empiriques ont été confirmés par différents théorèmes du type "no free lunch", (Schaffer, 1994; Wolpert, 1996b; Wolpert, 1996a). Entre autres, ils établissent le fait que pour deux algorithmes d' apprentissage quelconques, leur performance moyenne sur tous les problèmes d' apprentissage définis sur un ensemble d'entraînement spécifique, sera exactement la même. Pour un groupe quelconque de problèmes où un algorithme d' apprentissage surpasse l'autre, il existe un domaine où l'inverse est vrai.

Chaque algorithme a une "supériorité sélective", (Brodley, 1995), càd qu'il est meilleur que les autres pour un type de problèmes particulier. Ceci est dû au fait que chaque algorithme a ce que l'on appelle un "biais inductif" engendré par les hypothèses faites afin de généraliser d'une donnée d'entraînement à des exemples jamais vus auparavant.

Selon Mitchell (1997), "le biais inductif d'un algorithme d' apprentissage est l'ensemble de toutes les hypothèses requises pour justifier ses inférences inductives comme étant des inférences déductives". Donc, l'analyste doit posséder beaucoup d'expérience pour pouvoir identifier l'algorithme le plus approprié à la morphologie du problème posé. Le processus de sélection de modèles et d'algorithmes adéquats est décrit en détail par Brodley and Smyth (1997).

Le modèle d'un algorithme définit en fait l' "espace de recherche" ou l' "espace d'hypothèses", qui définit aussi le "biais représentationnel", de l'algorithme, comme par exemple k-FND (forme normale disjonctive), ou k-FNC (forme normale conjonctive), les fonctions linéaires discriminantes, les règles, etc. L' algorithme "fouille" cet espace pour rechercher la bonne hypothèse, càd l'hypothèse qui correspond le mieux aux données. L'algorithme détermine ainsi l'ordre de visite des états de cet espace, cet ordre est aussi appelé le "biais de recherche" de l'algorithme. Par exemple, entre deux algorithmes qui recherchent dans un espace FND, l'un pourrait commencer la recherche par les formes FND qui contiennent un ensemble complet de variables, tandis que l'autre pourrait commencer par des ensembles ne contenant qu'une seule variable. Par conséquent, le mauvais choix d'algorithme pourrait entraîner une convergence lente vers la bonne hypothèse, ou pourrait même ne pas aboutir à la solution optimale à cause d'un minimum local. Un mauvais choix de modèle pourrait avoir un impact plus grave encore: une hypothèse appropriée à notre problème risquerait d'être ignorée si elle n'est pas contenue dans l'espace de recherche du modèle.

Une formalisation des espaces de biais est donnée par Gordon and desJardin (1995), où sont décrits divers niveaux de biais. Au plus bas niveau nous trouvons un espace d'hypothèses spécifique et une méthode pour le fouiller. Au dessus, nous avons l'espace représentationnel qui a comme états les différentes représentations et divers espaces de recherche définis pour chacun des états de l'espace représentationnel. Le problème du biais approprié devient alors un problème de recherche dans ces espaces.

La tâche de classification est une tâche itérative. L'analyste doit tout d'abord sélectionner un model / une classe d'algorithmes, par exemple sélectionner entre la classe d'algorithme d'arbres de décision ou la classe d'algorithme des réseaux de neurones. A l'étape suivante on sélectionne un algorithme particulier implémentant une méthode spécifique pour chercher à travers l'espace représentationnel associé au modèle choisi. L'algorithme est ensuite appliqué et la qualité de ses prédictions est évaluée. Si les résultats d'évaluation sont médiocres, le processus est répété à partir du stade antérieur avec de nouvelles sélections. La procédure d'évaluation est ainsi assez coûteuse en temps et devient problématique lorsque le volume de données est important.

L'impact du travail et de l'expérience de l'expert dans cette procédure de "trial-and- error" est évident. Une pléiade de systèmes contenant une variété de modèles et d'algorithmes existent et sont à la disposition de l'analyste. Cependant, la sélection parmi ceux-ci reste de la responsabilité de l'analyste et à ce jour il n'existe aucun système qui pourrait fournir des suggestions ou un support pour déterminer laquelle des sélections serait la plus appropriée pour un problème donné.

II Contributions

Le but de cette thèse est de fournir un support à l'analyste pour sélectionner l'algorithme de classification approprié pour un problème spécifique en prenant en considération la nature du problème. Nous ne ferons pas ici une distinction entre un algorithme et le modèle représentationnel de l'algorithme. En revanche, nous considérerons que l'algorithme d' apprentissage est l'entité à sélectionner.

Nous traiterons le problème de la sélection d' inducteur comme un problème typique de classification, mais à un meta-niveau. Dans un problème de classification on donne à un algorithme d' apprentissage un ensemble d'instances d'entraînement et il doit construire un modèle inductif afin de prédire les classes de nouvelles instances inconnues. Dans notre base de travail de meta- apprentissage les instances d'entraînement seront les descriptions des ensembles de données individuels, et la meta-classification ou bien la tâche de meta-apprentissage sera de prédire quel algorithme d' apprentissage est le plus approprié pour un ensemble de données particulier.

Nous énumérons ci-dessous les principales contributions de ce travail.

1. Sur l'espace du meta-apprentissage
   - Une formulation originale de l'espace de meta-apprentissage en terme de comparaisons par paire d'algorithmes d' apprentissage. L'espace de meta- apprentissage simule d'une manière réaliste le processus qu'un analyste adopte lorsqu'il doit sélectionner parmi différents inducteurs.
   - La grande flexibilité que l'espace du meta-apprentissage offre. L' incorporation des problèmes par paire de meta-apprentissage permet une étude plus approfondie des facteurs qui affectent les performances relatives de différents inducteurs.
2. Sur la définition des caractéristiques
   - Une recherche systématique pour des caractéristiques qui peuvent décrire un ensemble de données d'une manière adéquate, avec l'accent principalement mis sur la description des relations qu'il y a entre les attributs. Ces caractéristiques deviendront les caractéristiques prédictives des problèmes par paire de meta- apprentissage.
   - Un effort pour trouver une manière appropriée pour faire la jonction entre des caractéristiques des propriétés d'attributs continus et des caractéristiques qui décrivent des propriétés d'attributs discrets. Dans ce contexte nous avons aussi introduit la notion de valeur non-applicable, pour ces caractéristiques dont le calcul n'a pas de sens pour des attributs discrets et vice-versa.
   - Une nouvelle représentation des caractéristiques utilisées pour décrire les propriétés de chaque attribut d'un ensemble de données ou combinaisons d'attributs, à travers l'utilisation d'histogrammes, dans l'optique de décrire le plus fidèlement possible leur distribution, sans perte d'informations discriminantes.
3. Des comparaisons étendues des différentes manières de caractériser les ensembles de données ainsi que l'établissement d'une classification d' inducteurs.

- Une comparaison empirique des différentes façons de caractériser les ensembles de données afin d'effectuer une sélection d' inducteurs dans deux bases de travail de meta-apprentissage différentes.
- Une comparaison empirique des différentes façons de caractériser les sets de données avec une approche de régression au meta- apprentissage.
- L'exploitation de l'approche de régression pour prédire l'ordre des inducteurs et la comparaison de différentes méthodes d'ordonnancement.

L'évaluation empirique du système a montré qu'elle pouvait fournir des suggestions exactes quant à savoir quel algorithme d' apprentissage est le plus approprié pour un ensemble de données spécifique. De plus, les modèles de meta- apprentissage, construits par les inducteurs, appliqués sur les problèmes de meta- apprentissage nous permettront d'avoir une meilleure compréhension des caractéristiques des ensembles de données qui affectent la performance des algorithmes d' apprentissage.

III Sommaire des chapitres

Ci-après nous donnons une brève description de chacun des chapitres de ce mémoire de thèse.

Chapitre 2. Méthodes de sélection d'algorithmes. Dans ce chapitre nous donnons une vue d'ensemble des différentes approches existantes pour la sélection d'algorithme. Nous présentons les méthodes utilisées pour évaluer les algorithmes d' apprentissage, estimons leurs erreurs et comment estimer la pertinence des résultats à l'aide de tests de nature statistique. Nous continuerons avec une présentation du travail effectué sur la sélection automatique d'algorithmes et nous le décrirons selon trois dimensions. Premièrement la façon dont les ensembles de données peuvent être caractérisés, deuxièmement les dispositions d'évaluation utilisées par les méthodes automatiques de sélections d'algorithmes et finalement la manière dont ces systèmes fournissent des suggestions à l'analyste. A la fin de ce chapitre nous présentons quelques travaux qui ne sont pas directement associés avec la sélection automatique d'algorithmes.

Chapitre 3. La base de travail du meta-apprentissage. Dans ce chapitre nous présentons l'architecture du système. Nous donnons une description de l'espace du meta-apprentissage et des problèmes par paire de meta-apprentissage formant cet espace. Les problèmes par paire de meta-apprentissage correspondent à toutes les paires possibles de n algorithmes, qui donne n(n-1)/2 paires. Les instances de chacune des paires sont des descriptions des ensembles de données accompagnées d'une étiquette de classe indiquant quel algorithme de la paire montre la meilleure performance pour l'ensemble de données correspondant. Nous montrons comment en combinant les meta-modèles construits à partir des problèmes par paire, nous arrivons à avoir une prédiction quant à savoir quel(s) inducteur(s) est (sont) les plus adapté(s) pour un nouvel ensemble de données qui ne faisait pas partie de l'ensemble d'entraînement. De plus, nous présentons l'ensemble des ensembles de données que nous utilisons pour entraîner et évaluer le système, ainsi que la série d' inducteurs à partir desquels nous allons effectuer la sélection d'algorithmes. Pour finir ce chapitre, nous présentons comment l'évaluation du système sera effectue sous deux angles. Le premier, que nous appellerons la précision stricte, est le pourcentage de fois que le système donne la bonne prédiction, càd. le nombre de fois que le système prédit correctement les algorithmes qui constituent le groupe des meilleurs algorithmes. Le deuxième, que l'on appelle la précision libre, correspond au pourcentage de fois que le système a donné comme résultat un ensemble d'algorithmes d' apprentissage qui était un sous-ensemble des meilleurs algorithmes.

Chapitre 4. La description des ensembles de données. La bonne sélection des caractéristiques qui seront utilisées pour décrire les ensembles de données est cruciale pour la performance du système. Ceux-ci devraient décrire des caract éristiques morphologiques des ensembles de données qui affectent la performance des algorithmes de classification. Différents inducteurs montrent différentes sensibilit és aux morphologies spécifiques des ensembles de données. Ce que nous voulons faire c'est de montrer comment ces morphologies affectent la performance relative des diff érents inducteurs. Par exemple, des inducteurs présentent différents degrés de sensibilité à la présence d'attributs incongrus. Les approches du plus proche voisin y sont très sensibles, tandis que les algorithmes d'arbres de décisions et de r éseaux de neurones sont assez robustes, car ils possèdent des mécanismes internes qui effectuent des sélections d'attributs. Un autre exemple est la distinction entre les approches numériques, comme les réseaux de neurones, ou les discriminants linéaires, et ceux basés sur une représentation symbolique tels que les arbres de décision ou les inducteurs de règles. Le premier est plus adapté aux ensembles de données où les attributs sont en majorité numériques et le second plus adapté pour les ensembles de données où les attributs sont majoritairement symboliques. Nous nous efforcerons de trouver un ensemble de caractéristiques qui décrive le mieux possible ces facteurs.

Dans ce chapitre nous établissons l'ensemble de caractéristiques qui sera utilisé pour décrire les ensembles de données et construire les problèmes par paire de meta-apprentissage après avoir passé en revue les méthodes existantes de caractérisation d'ensembles de données. Une des principales limitations de ces méthodes est la façon dont ils traitent les caractéristiques qui peuvent être définies sur la base d'un attribut ou d'une paire d'attributs. Les méthodes existantes comptent sur l'utilisation de moyennes des caractéristiques qui sont définies pour un ensemble d'attributs. Par exemple dans le cas de concentration de coefficient pour k attributs il y aura k(k-1) coefficients distincts dont chacun est associé à une paire spécifique. Les méthodes existantes remplacent ces ensembles de valeurs par leur moyenne, ce qui résulte en une perte d'information précieuse sur la distribution des coefficients. Nous proposons l'utilisation d'histogrammes pour décrire d'une manière plus fine cette distribution. Nous subdivisons les caractéristiques des ensembles de données en cinq catégories différentes selon quelle propriété de l'ensemble de données elles décrivent et nous donnons les caractéristiques qui appartiennent à chaque catégorie. La première de ces catégories contient les caractéristiques qui donnent une information sur le type des attributs qui apparaissent dans l'ensemble de données, par exemple le nombre d'attributs continus ou discrets. La seconde décrit les attributs individuels, un exemple en est l'entropie des attributs. La troisième donne des moyens d'association d'attributs comme les coefficients de corrélation et de concentration. Dans la quatrième catégorie nous plaçons les moyens d'association des attributs avec la classe, et dans la dernière catégorie nous mettons les caractéristiques qui ne peuvent pas être classées dans aucune des catégories précédentes.

A la fin nous évaluons le coût en terme de calcul de ces caractéristiques et nous comparerons avec le coût d'une validation croisée pour sélectionner le meilleur algorithme d' apprentissage.

Chapitre 5. Meta-apprentissage à partir d'instances. La sélection de l' inducteur adéquat qui sera utilisé au niveau du meta-apprentissage affectera d'une manière critique la performance du système. Pour commencer, nous utilisons un simple algorithme d' apprentissage à partir d'instances. Les algorithmes d' apprentissage à partir d'instances n' induisent pas de modèles à partir des données d'entraînement, ils se basent uniquement sur des mesures de distance des instances qui doivent être classifiées à partir des données utilisées pour l'entraînement. Il y a deux raisons principales pour la sélection d'un algorithme d' apprentissage à partir d'instances au meta-niveau. Premièrement et le plus important, est que nous nous attendons à ce que les algorithmes d' apprentissage montrent des performances similaires sur des ensembles de données avec des caractéristiques similaires, pour que l'on puisse exploiter la performance antérieure d'algorithmes pour prédire la performance sur des ensembles de données inconnus. Deuxièmement, il est facile d'adapter la mesure de distance utilisée par un algorithme à partir des instances afin qu'il intègre la valeur "non-applicable" dans ses calculs. Nous définissons la similarité entre les ensembles de données en terme de proximité géométrique dans l'espace de morphologie, dont les dimensions sont définies par les caractéristiques de l'ensemble de données, et nous observons l'espace de morphologie comme un espace euclidien conventionnel, étendu par "non-applicable". Nous avons modifié la définition de distance de l'algorithme du plus proche voisin pour prendre en compte des attributs dont le domaine est R U non-applicable. Les caractéristiques morphologiques sont toutes normalisées à l'intervalle [0,1] avant l'application de l'algorithme.

Nous présentons les résultats de l'évaluation du système lorsque l' inducteur à partir d'instances est utilisé sur le meta-niveau. Deux variantes de l'ensemble des ensembles de données de caractéristiques sont comparées par rapport à leur performance prédictive, *histo* et *+histo*. La différence entre la seconde et la première est qu'elle a été étendue pour inclure les distributions d'une mesure basée

ANOVA dans l'optique de décrire les associations entre des attributs discrets et continus. Les résultats ont montré, en ce qui concerne l'algorithme d' apprentissage à partir d'instances, que l'incorporation de ces caractéristiques réduit le pouvoir de discrimination de la caractérisation.

Une question qui a recueilli peu d'attention, pour ne pas dire aucune, dans le domaine du meta-apprentissage, est l'explication et la compréhension des facteurs qui affectent la performance des inducteurs. Tous les efforts précédents avaient pour but de maximiser les capacités prédictives du meta-apprenti sans pour autant éclaircir les facteurs (càd les propriétés des ensembles de données) qui affectent la performance des algorithmes. En appliquant la sélection de variables au meta-niveau on pourrait combler cette lacune et par la-même améliorer la performance du meta- apprentissage. En utilisant la sélection de variables, nous pouvons avoir une meilleure idée sur les facteurs qui affectent la performance des algorithmes d' apprentissage. Ceci est encore plus vrai lorsque l'algorithme de meta-apprentissage utilisé est un algorithme d' apprentissage à partir d'instances, qui ne donne aucune suggestion dans la pertinence des attributs utilisés. C'est pourquoi dans le même chapitre nous avons aussi examiné l'utilisation d'un algorithme d' apprentissage à partir d'instances en association avec un mécanisme de sélection de variables. Pour chacun des problèmes par paire de meta-apprentissage nous examinons les attributs sélectionnés par le mécanisme de sélection d'attributs et à la fin nous caractérisons chaque attribut à l'aide de son pouvoir de discrimination total, càd à quelle fréquence apparaît cette variable sur la totalité des problèmes de meta- apprentissage. Les résultats montrent que l'utilisation de sélection de variables peut en effet améliorer la performance du système en terme de son pouvoir prédictif.

Chapitre 6. Comparaison des algorithmes d' apprentissage au meta-niveau. Dans ce chapitre nous explorons l'utilisation d'algorithmes d' apprentissage plus élaborés sur le plan du meta-apprentissage : les algorithmes à base d'arbre de décisions. Le but principal est d'améliorer encore la performance du système. Nous analysons aussi les modèles produits, afin d'évaluer le pouvoir prédictif des caractéristiques des ensembles de données, ceci de la même manière que nous avions analysé les caractéristiques qui étaient sélectionnées par le mécanisme de sélection de variables du chapitre precèdent. Ce chapitre est organisé comme suit. Premièrement, nous examinons la performance des nouveaux meta-apprentis avec les deux ensembles distincts des ensembles de données de caractéristiques, càd *+histo* et histo, afin de voir avec lequel nous obtenons la meilleure performance. Les résultats sont néanmoins différents des comparaisons analogues du chapitre précédent. Avec les inducteurs basés sur les arbres de décision la différence de performance entre les deux caractérisations n'est pas statistiquement significative. Ensuite, nous comparons les performances des meta-apprentis, y compris celui de fsIBL sur l'ensemble de données *histo*. C50boost se trouve être le meilleur meta-apprenti, celui dont les prédictions sont les plus précises.

Et finalement les modèles inductifs construits par deux des meta-apprentis sont analysés afin d'examiner le pouvoir discriminatoire des caractéristiques et les caractériser à nouveau en terme de leur pouvoir discriminatoire total.

Chapitre 7. Comparaison de datasets de caractérisations. En commençant par le projet STATLOG en 1994, et jusqu'à aujourd'hui avec le projet METAL, une grande variété de mesures est utilisée pour décrire et caractériser les ensembles de données dans le but de prédire les performances d'algorithmes d' apprentissage. A notre connaissance, il n'existe pas de comparaisons systématiques des différentes approches de caractérisation d'ensemble de données. A la seule exception près, celle du travail de Bensusan et Giraud-Carrier (2000), où ils comparent la performance de "landmarking" avec celle d'une caractérisation des ensembles de données basée sur l'information dans le même esprit que STATLOG. Les ensembles de données utilisés dans cette étude sont artificiels. La description basée sur l'information consistait en une entropie de classe, nombre équivalent à d'attributs, entropie moyenne d'attributs, moyenne d'information réciproque, moyenne entropie jointe et rapport signal sur bruit ; ce qui représente un ensemble limité de caractéristiques qui est en fait un sous-ensemble de celles utilisées dans STATLOG. Les découvertes expérimentales ont montré que le "landmarking" devance la description basée sur l'information, cependant les résultats doivent être pris avec précaution, vu que l'étude a été menée seulement sur des ensembles de données artificiels, l'ensemble des caractéristiques basé sur l'information était plutôt limité et de plus il n'y avait pas de contrôle sur la pertinence statistique des résultats.

Le but de ce chapitre est d'effectuer une comparaison contrôlée et systématique des différents ensembles de données de caractérisations, sur des ensembles de données réels. Nous examinons cinq ensembles différents de caractéristiques. Quatre d'entre eux suivent l'approche basée statistique/information présentée dans STATLOG et la cinquième est l'approche "landmarking" pour caractériser un ensemble de données. Plus précisément nous examinons les ensembles suivants :

- *statlog*, l'ensemble de caractéristiques utilisé dans le projet STATLOG.
- *dct*, un ensemble des ensembles de données de caractéristiques plus riche extrait de l'outil DCT qui a été développé à la suite du projet METAL.
- *histo*, l'ensemble de caractéristiques que nous avons établi.
- *histo-limited*, une version réduite de histo.
- *land*, une caractérisation des ensembles de données en terme de performance prédictive de simples algorithmes d' apprentissage.

La comparaison implique 65 ensembles de données réels, principalement du dépôt de l'UCI et du projet METAL. Le nombre des ensembles de données aurait dû être plus élevé si l'outil de "landmarking" n'avait pas échoué en caractérisant un nombre trop important des ensembles de données. Nous utilisons deux structures de meta-apprentissage différentes pour effectuer les comparaisons. La première est la structure par paire que nous avons introduit au chapitre 3, la seconde est une approche plus simple du meta-apprentissage et son but principal étant la prédiction de l'algorithme d' apprentissage qui atteindra la plus haute précision. Ici nous n'utilisons pas de comparaisons par paire, nous avons juste un problème de meta- apprentissage simple dont le but est de prédire l'algorithme qui accomplit la plus grande exactitude pour un ensemble de données donné. Les instances de l'ensemble de données du meta-apprentissage sont les descriptions des ensembles de données accompagnées d'une étiquette de classe donnant l'algorithme qui a la plus grande précision sur l'ensemble de données, ceci est déterminé par la méthode de validation croisée stratifiée. Aucun type de test d'une pertinence statistique n'est utilisé pour sélectionner le meilleur algorithme, seule la valeur absolue de la précision estimée par la validation croisée est prise en considération.

C50boost a été utilisé dans les deux structures comme étant le meta- apprenti. La stratégie d'évaluation pour la deuxième structure de meta- apprentissage sera la précision de c50boost estimée par validation croisée stratifiée.

En ce qui concerne la première structure de meta-apprentissage et la performance sur les problèmes par paire, les résultats n'ont pas été concluants. Deux des cinq caractérisations, statlog et land n'ont pas pu devancer la précision de base à un niveau statistiquement significatif dans aucun des 28 problèmes par paire. En fait, land a eu une précision moyenne qui était encore pire que la moyenne de précision de base. Les trois caractérisations restantes ont surpassé la précision de base pour deux ou trois problèmes par paire. En ce qui concerne la performance par rapport à la suggestion finale, seule les approches basées sur l'histogramme ont pu dépasser la précision de base en terme d'exactitude stricte. Les trois approches restantes ont été encore pires que la précision de base correspondante. Malheureusement, les différences avec les approches basées sur l'histogramme quant à la précision de base n'étaient pas statistiquement pertinentes.

Les résultats sur la structure simple du meta-apprentissage sont légèrement différents. Ici, quatre des cinq caractérisations dépassent la précision de base: *histo, histo-limited, dct* et *statlog*; la seule qui ait montré une précision inférieure à celle de base était le set land, probablement du au fait que nous avons utilisé un ensemble plus limité que les landmarkers initiaux. Le set *histo* réalise la meilleure performance avec une amélioration considérable par rapport à la base, mais encore une fois cette amélioration n'est pas statistiquement significative.

Chapitre 8. Meta-apprentissage basé sur la régression. Dans tous les chapitres précédents, nous avons considéré le problème de meta-apprentissage comme une tâche de classification. Dans ce chapitre nous explorons une autre alternative où nous approchons le problème comme étant un problème de régression. Nous abordons les tâches de meta-apprentissage comme des tâches de régression où nous cherchons des rapports entre les propriétés d'un ensemble de données et la performance du classifieur. Cette approche directe est plus flexible que le meta-apprentissage pour la sélection de modèles puisque les estimations ne sont pas relatives à un ensemble spécifique de classifieurs. Avec ce scénario de régression, nous pouvons utiliser les modèles construits pour effectuer des sélections d' inducteurs, càd sélectionner les inducteurs les plus appropriés pour un ensemble de données, ainsi qu'un ordonnancement d' inducteurs, càd ordonner les inducteurs par leur performance prévisionnelle sur un ensemble de données. Nous évaluons l'approche de régression pour les deux tâches et nous examinons la performance des modèles de régression construits à partir des cinq ensembles de caractéristiques utilisés dans l'étude comparative du chapitre 7. L'algorithme de régression utilisé est un algorithme de régression-noyau. En fin, nous incluons dans notre étude la classification basée sur le "zooming" introduite dans (Soares & Brazdil, 2000). Le land accomplit de loin la meilleure performance en terme d'estimation d'erreurs des inducteurs individuels. Mais lorsque ces estimations ont été utilisées afin d'ordonner les algorithmes par rapport à leur performance prévue, le résultat fut très médiocre, même considérablement plus mauvais que l'ordre de base. Les meilleurs résultats d'ordonnancement ont été obtenus par la combinaison de Kernel et dct, suivi par celle de Kernel et *histo-limited* et *histo*. L'ordre basé sur la régression a dépassé en général l'ordre basé sur le "zooming", néanmoins la différence entre les méthodes et entre les méthodes et la performance de l'ordre de base n'ont pas été significatifs au niveau statistique. Lorsque les prédictions de régression ont été utilisées pour effectuer une sélection d' inducteur, seules trois méthodes arrivent à dépasser la précision de base, mais pas à un niveau statistiquement significatif. En première position nous trouvons Kernel et *histo*, le même ensemble qui avait montré la meilleure performance lorsque nous avions abordé le problème de sélection d' inducteur à travers la classification. Suivi de Kernel avec *dct* et *histo-limited*. Le reste des méthodes a eu une performance plus mauvaise que celle de la précision de base. En général l'approche basée sur la régression semble avoir une performance légèrement moindre à celle de l'approche de type "classification".

Chapter 1

Introduction

The machine learning field has been evolving for a long time and has given us a variety of models and algorithms to perform the task of classification, e.g. decision trees, neural nets, rule inducers, nearest neighbor etc. The analyst must select among them the ones that better match the morphology and the special characteristics of the problem at hand. This selection is one of the most difficult problems since there is no model or algorithm that performs better than all others independently of the particular problem characteristics, as it has been observed in various empirical comparisons, (Aha, 1992; Salzberg, 1991; Shavlik et al., 1991; Weis & Kapouleas, 1989).

Later the empirical results have been confirmed by the various "no free lunch theorems", (Schaffer, 1994; Wolpert, 1996b; Wolpert, 1996a). Among others they state that for any two learners, their performance averaged over all the possible learning problems defined over a specific training set, will be exactly the same. For any class of problems where one learner outperforms the other there will be another area in which the opposite situation holds.

Each algorithm has a "selective superiority",(Brodley, 1995), i.e. it is better than the rest for specific types of problems. This happens because each algorithm has a so-called "inductive bias" caused by the assumptions it makes in order to generalize from the training data to unseen examples. According to Mitchell (1997), "the inductive bias of a learning algorithm is the set of all the assumptions which are required in order to justify its inductive inferences as deductive inferences". Hence, the analyst must posses a lot of experience to be able to identify the most appropriate algorithm for the morphology of the problem at hand. The process of selecting the appropriate models and algorithms is described thoroughly by Brodley and Smyth (1997).

The model of an algorithm actually defines the "search space" or "hypothesis space", which also determine the "representational bias" of the algorithm, such as k-DNF (disjunctive normal form), or k-CNF (conjunctive normal form), linear discriminant functions, rules etc. The algorithm searches this space for the right hypothesis, i.e. for the hypothesis that better fits the data. The algorithm determines the order of visiting the states in this space, this order is also called the "search bias" of the algorithm. For example, between two algorithms that both search in DNF space, one might start the search from DNF forms that contain the complete set of features, while the other might start from sets consisting of only one feature. Hence, the wrong choice of algorithm may result in slow convergence towards the right hypothesis, or may even end at a suboptimal solution due to a local minimum. A wrong choice of model can have a more severe impact: A hypothesis appropriate for the problem at hand might be ignored because it is not contained in the model's search space.

A third source of bias comes from the method used to evaluate the learning algorithms, this form of bias is usually identified as "validation bias". As Bailey and Elkan (1993) note, the best choice of error estimation procedure depends on which learning algorithm is been evaluated.

A formalization of the bias spaces is given by Gordon and desJardin (1995), where various levels of bias are described. At the lower level we have a specific hypothesis space and a way to search it. Above it we have the representational space having as states the different representations, and various search spaces defined for each of the states of the representational space. The problem of the appropriate bias then becomes a problem of searching those spaces.

The classification task is an iterative task. The analyst must first select a model/class of algorithms, for example select between the class of decision tree algorithms or the class of neural network algorithms. On a next step a particular algorithm implementing a specific way to search through the representational space associated with the chosen model, is selected. The algorithm is then invoked and the quality of its predictions is evaluated. If the evaluation results are poor, the process is repeated from a previous stage with new selections. The evaluation procedure is quite time consuming, and becomes problematic when the volume of data is high.

The human effort and experience in this trial-and-error procedure is apparent. A plethora of systems with a variety of models and algorithms exist at the analyst's disposal. However, the selection among them is left to the analyst and so far there is no system that can provide support or suggestions as to which selections are more appropriate for a specific problem.

1.1 Contributions

The goal of this thesis is to provide support to the analyst in selecting the appropriate classification algorithm for a specific problem, taking into consideration the nature of the problem. We will not make a distinction between an algorithm and the representational model of the algorithm. We will rather consider the learning algorithm as the entity to be selected. We will tackle the problem of inducer selection as a typical classification problem, although at a meta-level. In a typical classification problem a learner is given a dataset of training instances and is required to construct an inductive model in order to predict the classes of new unseen instances. In our meta-learning framework the training instances will be the descriptions of individual datasets, and the meta-classification or else meta-learning task will be to predict which learning algorithm is more appropriate for a specific dataset.

We will now give, a list of what we think that are the main contributions of this work.

1. On the meta-learning space
   - (a) A novel formulation of the meta-learning space in terms of pairwise comparisons of learning algorithms. The meta-learning space simulates closely the typical process that an analyst adopts, when he has to select among different inducers.
   - (b) The increased flexibility that the meta-learning space offers. The incorporation of the pairwise meta-learning problems allows for a closer study of the factors affecting the relative performance of different inducers.
2. On the definition of features
   - (a) A systematic search for features that can adequately describe a dataset, with the main emphasis placed on the description of attribute interelationships. These features will become the predictive features of the pairwise meta-learning problems.
   - (b) An effort to find an appropriate way to bridge the gap between features that describe properties of continuous attributes and features that describe properties of discrete attributes. In that context we also introduced the notion of *non-appl* values, for these features whose computation does not make sense for discrete attributes and vice versa.
   - (c) A new representation of the features used to describe the properties of each attribute of a dataset or combinations of attributes, via the use of histograms, in an effort to depict as close as possible their distribution, without loss of discriminating information.
3. Extensive comparisons of different ways of characterizing datasets as well as performing inducer ranking
   - (a) An empirical comparison of different ways of characterizing datasets in order to perform inducer selection, under two different meta-learning frameworks.
   - (b) An empirical comparison of different ways of characterizing datasets under a regression approach to meta-learning.
   - (c) The exploitation of the regression approach to predict rankings of inducers, and a comparison of different ranking methods.

The empirical evaluation of the system has shown that it can provide successful suggestions as to which learning algorithm is more appropriate for a specific dataset. Furthermore the meta-learning models constructed by the inducers applied on the meta-learning problems will allow us to have a better understanding of those dataset characteristics that affect the performance of the learning algorithms.

1.2 A guide to the chapters

The remainder of this dissertation is organized as follows. In **chapter 2** we give an overview of the existing approaches to algorithm selection. In **chapter 3** we present the architecture of the system. That is, we give a description of the meta-learning space and the pairwise meta-learning problems constituting that space. We show how from combining the meta-models constructed from the pairwise problems, we can get a prediction as to which inducer(s) is(are) more appropriate for a dataset. Furthermore, we present the set of datasets we use in order to train and evaluate the system, as well as the pool of inducers from which we are going to perform the algorithm selection. Finally, we present how the evaluation of the system will be done. In **chapter 4** we establish the set of features that will be used to describe the datasets, and to construct the pairwise meta-learning problems, after reviewing the existing work in characterizing datasets. The idea of histograms is also presented, and various issues are discussed concerning the quality of the characteristics and the problems that they set. Finally, an estimation of the computational cost of the dataset characteristics is provided. In **chapter 5** we present the results of the evaluation of the system when an instance based inducer is used on the meta-level. Two variations of the set of dataset characteristics are compared with respect to their predictive performance. In the same chapter we examine the use of feature selection in order to improve the performance of the instance based inducer. One clear result is the fact that the features finally selected are different for each pairwise problem. **Chapter 6** provides a comparison of four different learners on the meta-learning level, in order to establish which is the most suitable inducer. The meta-models of two of the inducers are analyzed in order to characterize the discriminatory power of the datasets characteristics. In **chapter 7** we perform a comparative study of different ways of characterizing datasets, using only real world datasets. The comparison is done under two different meta-learning frameworks, and the results show an advantage of the histogram based characterizations, over the other characterizations. In **chapter 8** we deviate from the approach that we followed in the previous chapters, where we handled the problem of algorithm selection as a classification problem. Here instead, we explore the use of a regression algorithm to predict the relative errors of pairs of algorithms, but also to predict the absolute error of every algorithm. The predictions are used to perform algorithm selection and to provide rankings of algorithms. We examine the performances of different strategies of characterizing datasets, under the regression scenario and we compare the regression based ranking with a well established method of ranking. In the last chapter we give an overview of the work, along with the problems that we faced, the remaining open issues and possible ways to address them.

Chapter 2

Methods of Algorithm Selection

Formulating and solving a classification problem is a time intensive task consisting of many phases, which requires a considerable amount of diverse knowledge from the analyst. Before anything else, there is the definition of the classification problem and the collection of the appropriate data for solving the problem. In this stage it is the domain expert who has to formulate the problem, determine the features that provide relevant information for solving it, and of course provide them. It can be a quite time consuming process and since usually the data do not come for free, the amount of relevant information at the end can be limited. The quality of the available data is one of the most crucial factors in achieving a high performance solution. However, we will not go into the details of data collection and of the quality of the available data. The analysts's main task will be to select the most appropriate learning algorithm, according to some performance measures and within the constraints imposed by the application.

In figure 2.1, we give an overview of the analysis process. The analyst has at his disposal a pool of classification algorithms, from which he initially selects some for evaluation on the specific problem. The initial selection can be based on knowledge of the problem, that is select those algorithms whose characteristics better match the characteristics of the dataset, or even on the analyst's preferences for specific learning algorithms. The evaluation usually requires extensive experimentation, which consists in repetitive execution of the selected algorithms. The form of evaluation depends on constraints imposed by the application and the volume of the data; sometimes it is possible to perform computational intensive evaluation, while other times this is prohibitive. It can be that the results of the evaluation are poor for all the algorithms evaluated, this can be due to a bad choice of inducers, or even worse an indication that the quality of data is poor. Whichever the cause, the result is a reiteration of the process. After completing successfully the evaluation, the next step is the comparison of the achieved performances in order to select the most appropriate

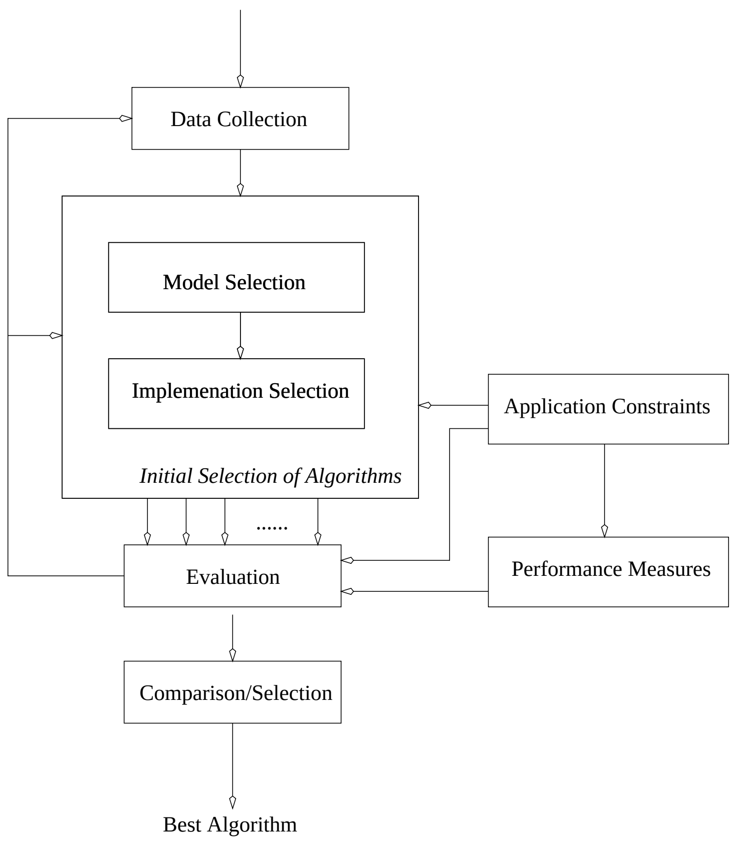

Fig. 2.1. Model and task selection for knowledge discovery

classification algorithm. The selected algorithm will be the one used to construct the final classification model from the available data.

The goal of all algorithm selection systems is to relieve the analyst of the intensive evaluation phase, using information about the problem and the dataset in order to directly suggest the more appropriate inducer.

In the forthcoming sections, we will first give an overview of the evaluation phase, the issues involved and the methods used in order to perform it, followed by an overview of the existing systems for algorithm selection.

2.1 Algorithm Evaluation and Selection

The typical process in selecting which inducer will be applied to a dataset in order to construct a classification model, depends on the goal of the analysis. For example, if the target is to acquire an understanding of the data, the analyst can choose from the beginning to eliminate from the pool of algorithms that he is considering to use, the ones that do not produce understandable and interpretable models. Examples of which are nearest neighbors methods that do not produce any kind of models, or neural nets that produce models difficult to interpret. If time is a critical concern, this can lead to a different set of possible candidates, from which the computationally intensive algorithms are excluded. In a different scenario, which will be also our working hypothesis, the analyst is interested in selecting the classification algorithm that can better fit the data, that is, he is interested in the inducer that will exhibit the lowest generalization error. To determine the appropriate inducer, he first uses some error evaluation method, in order to get an estimate of the errors of the initially selected ones. In a next step he compares the error estimates, usually via some kind of statistical hypothesis test, in order to determine which is the best for the dataset under examination.

Before proceeding to a more thorough description of the evaluation and comparison methods, we will give a description of the basic constituents of the learning process, and of the notion of generalization error.

- Consider a generator G of random vectors X, which are drawn according to an unknown, but fixed, probability distribution, $P(X)$.
- A supervisor S that assigns output values, class labels, $Y = S(X)$, to the X random vectors, according to an unknown, but also fixed, conditional probability distribution $P(Y|X)$.
- The pairs (X, Y), drawn from the probability distribution $P(X, Y) = P(Y|X)P(X)$, constitute the learning space.
- The goal of an inducer I, is to construct a hypothesis $H(X)$ that best approximates the response of the supervisor, $S(X)$.

In the real world the inducer has access to a dataset D with a limited number of examples, drawn from the distribution $P(X, Y)$. The inducer will be trained on this dataset in order to construct the approximation of $S(X)$.

The *generalization error* of the $H(X)$ hypothesis, constructed by I on the dataset D, is the probability that H will misclassify an example drawn at random from $P(X)$. That is :

$$generalization\ error = P_{x \in P(X)}(H(x) \neq S(x))$$

Since we can not draw an infinite number of new examples from $P(X)$, on which we could compute the exact generalization error, we have to rely on estimations of it using the available data, D, by some error estimation procedure. It is these error estimates which will be later used in order to select the best inducer, by means of some statistical test.

2.1.1 Error Estimation

The general idea underlying all error estimation procedures, is the division of the available set of examples in two disjoint sets. One is used for training, and the other is used for testing/evaluating the generated model. The test set should not contain examples that have been used in the training set, as this would provide optimistically biased estimates of the error. Various methods are used for obtaining the division to train/test sets and estimating the error. We will briefly present some of them.

The simplest method and the one with the lowest computational requirements is the *holdout* method, where the available set of examples is partitioned in two disjoint sets. Usually 2/3 of the initial examples are used for training the inducer and the remaining 1/3, called the holdout set, is used for testing the produced model. The method is used when the volume of the data is quite high, and the repetitive execution of training and testing phases, required by other methods, is prohibitive. It makes poor use of the examples for training, since 1/3 of them is never used. Nevertheless, when the number of examples is high it gives reliable estimates.

In *k-fold cross validation* method the available set is split into k disjoint sets. The inducer is then trained on the union of $k-1$ sets and tested on the remaining set. The whole process is repeated k times, each time a different set from the k is used as a test set. The estimation of the error is simply the average of the observed errors over the k folds. When k equals the number of examples then the method is called *leave-one-out*. A variant of cross validation is *stratified cross validation*, where the partitions are constructed in such a way, that the distribution of the classes, as it appears in the initial dataset, is preserved.

In the *bootstrap* method the initial set of examples is sampled with replacement, so that a new set of the same size is established. The instances not chosen in the sampling process will form the testing set. The whole process is repeated a number of times, k, usually between 50 and 200, each time using a different sample of the examples. The estimation of the error is given by the following formula :

$$err = \frac{1}{k}\sum_{b=1}^{k}(0.632\epsilon_{test_b} + 0.368\epsilon_{train})$$

where ϵ_{test_b} is the error of the model on the b test set, and ϵ_{train} the error of the model on the complete initial set.

Leave-one-out produces almost unbiased estimates of the true error, but with high variance. The variance is reduced when we move to k-fold cross validation, with k in the area of five to ten, and it is further reduced when we are using stratified cross validation, still being relatively high. One method to reduce the variance of cross validation is to repeat the whole procedure for a number of times. For both cross validation and stratified cross validation the estimates of the mean are almost unbiased. In bootstrap the error estimates are highly biased, but they have a very low variance. Bootstrap's bias is high especially when algorithms that fit perfectly the training data are evaluated, e.g. a nearest neighbor algorithm. In that case ϵ_{train} is zero, leading to optimistic estimation of the error. Efron and Tibshirani (1995), propose a bootstrap version which they call the 632+ rule, which is designed to provide less biased estimates of the error. A comparative study of cross validation, stratified cross validation and bootstrap can be found in (Kohavi, 1995). The author concludes that the use of ten fold stratified cross validation is appropriate for algorithm selection, even if the computational power available is sufficient for more computational intensive methods of error evaluation. In a similar study, Bailey and Elkan (1993), compared the performance of bootstrap and leave-one-out cross validation; they also concluded that the use of cross validation is preferable, since it exhibits much smaller bias than bootstrap. They noted though that the best choice of error estimation method depends on which algorithm is evaluated.

The simple comparison of the estimated errors of a number of inducers is not sufficient to determine which is the best for a specific dataset. The observed differences in the error estimates might not be significant in a statistical context. The estimates of the errors are sample estimates of the true error. It is obvious that two inducers can have the same true error, but different sample estimates. In order to establish whether the differences in the sample estimates reflect a difference in the true error or are simply the result of random fluctuations of the sample estimates around the same mean, the use of statistical significance tests is essential.

2.1.2 Algorithm Selection based on Significance Testing

Statistical significance tests are used to control whether some hypothesis holds or not. In what concerns the comparison of learning algorithms, usually the hypothesis examined is whether two inducers have the same true error. The statistical test used should depend on the way that the evaluation of the error has been done.

When the evaluation of the error is done using a single split of the dataset, there are two main options. The *McNemar* test is a test that checks for the difference of two proportions. It is based on the number of times that the two algorithms disagree in their predictions. The number of times that the two algorithms are both correct or both wrong are not considered by the test. The second option is to consider the errors of the inducers as coming from a binomial distribution, which can in turn be approximated by a normal distribution, and use a simple test of the differences of the two means based on the normal distribution assumption.

In the case of cross validation one can use the *paired* t-*test* and test whether the difference of errors of the inducers between the folds of the cross validation is zero. Another option is to use the McNemar test again, but now the number of times that the two algorithms disagree will be computed from the union of the test sets. Another possibility is the *sign test*, that tests for the sign of the error differences between the folds. It assumes a binomial distribution, if the two inducers have no significant difference, then the number of + for each one of them should be approximately equal.

Special care should be to the assumptions of the statistical tests by the error estimation procedure. For example the paired t-test requires that the test sets are independent, that is they should not overlap. The same requirement is also imposed for the training sets. In the case of k-fold cross-validation this requirement is not violated for the test sets, since they are always disjoint, but it is violated for the training sets. Obviously, if we use repeated k-fold cross-validation then even the assumption of the independence of the test sets does not hold. For a thorough description and a comparative study of some of the aforementioned statistical tests in combination with error evaluation procedures see (Dietterich, 1998).

When the number of learning algorithms compared is limited to two, the application of the statistical tests is straight forward. But if the number of algorithms compared is higher than two, there is one more factor that should be considered, which is known as the *multiplicity effect*. Every statistical test has what is called a **Type I error**, which is actually the probability of rejecting the hypothesis examined when actually it holds. This type of error is controlled by the significance level, α, set by the analyst for the test. When n comparisons take place, the probability of committing a Type I error in one of them, is no longer α, but rather $1-(1-\alpha)^n$, the quantity $(1-\alpha)^n$ is the probability that we get all the comparisons correct. In order for the complete results to be significant at a desired significance level, α', we have to adjust the significance levels, α, of each of the pairwise comparisons according to $\alpha' = 1 - (1 - \alpha)^n$. For a detailed discussion on issues concerning the comparisons of learning algorithms see (Salzberg, 1997; Feelders & Verkooijen, 1995).

It is obvious that the selection of the most appropriate inducer, the one that achieves the lowest error, is not a trivial task. Apart from extensive experimentation in order to evaluate the algorithms, a sound knowledge of the weaknesses and strengths of the evaluation strategies and a good understanding of statistics is required, in order to select the appropriate combination of statistical test and evaluation procedure.

2.2 Automatic Algorithm Selection

As it is apparent, the task of algorithm selection is quite intensive and time consuming, since most of the evaluation procedures require repetitive application of the learning algorithms. The goal of systems that provide suggestions as to which algorithm should be used is to avoid the time demanding process of evaluation. Such systems usually rely on some kind of mapping between a description of the datasets and performance measures of the algorithms. The existing approaches can be characterized along the following dimensions :

- *Dataset descriptions* : the properties used to describe the datasets.

- *Evaluation measures* : the performance measures with respect to which the suggestion is provided, e.g. error, time performance.

- *Form of suggestion* : the form in which the suggestion of the system comes, e.g. whether it proposes a single algorithm, or a list of algorithms.

- *Method used to construct the suggestion* : how the mapping from the properties of the datasets to the performance measures is done.

We will shortly review the work done along these dimensions, for some of them a more thorough description will be given in the forthcoming chapters, where this is appropriate.

2.2.1 Datasets Description

There are two main directions used so far in order to characterize a dataset for providing suggestion as to which classification algorithm(s) is(are) more appropriate for a specific dataset. In the first one, measures that describe statistical and information based properties of the datasets are used. In the second one a dataset is described using the performance of very simple learners. In a very successful metaphor the first category of measures is described as the *genotype* of the datasets, i.e. the inner structure of the dataset, and the second category as the *phenotype* of the datasets, i.e. the visible properties of the dataset produced by the interaction of its genotype with the environment in our case the simple learners.

The description of a dataset in terms of its information/statistical properties, in order to provide recommendation as to which algorithm to use, appeared for the first time within the framework of the STATLOG project (Michie et al., 1994). The authors used a set of 15 characteristics, spanning from simple ones like the number of attributes or the number of examples, to more complex ones, like the first canonical correlation between the attributes and the class attribute, or the mean mutual information between attributes and the class. The set of characteristics introduced there was later used in various studies, aimed at solving the problem of algorithm selection, (Brazdil et al., 1994; Todorovski & Dzeroski, 1999; Sohn, 1999). Lindner and Studer (1999), continue in the same way, providing an exhaustive list of information and statistical measures of a dataset computed for each attribute or pairs of attributes. They provide a tool for the automatic computation of these characteristics which they call DCT. Nevertheless, they point out that only a limited set of these measures is relevant in providing recommendation. Set that in fact, was very similar to the one defined in STATLOG. Sohn (1999) also uses the STATLOG set as a starting point, but she proceeds with a careful evaluation of their properties in a statistical framework. As a result, she discovers that some of the characteristics are highly correlated, and she omits the redundant ones from her study. Furthermore she introduces new features that are transformations or combinations of the existing ones, like ratios or seconds powers, with the goal of providing more successful predictions.

Todorovski et al. (2000) relied on the set of characteristics produced by DCT. In order to overcome the limitations of the use of the average values for the characteristics which are computed per attribute, or per attribute pair, they included in their set of characteristics the minimum and maximum values of these. They also include new characteristics which are ratios of the already existing.

In the second approach as already mentioned above a dataset is described in terms of accuracy performance that simple learners achieve on the particular dataset. This approach is called landmarking and it was introduced in (Pfahringer et al., 2000). A more thorough description of the related work will be given in section 4.1 of chapter 4, where we will introduce the set of characteristics used in the present study.

2.2.2 Evaluation measures

In order to provide suggestions as to which inducer should be applied on a particular dataset, performance measures should be used, according to which a preference order among the inducers will be established.

An obvious performance measure is that of the accuracy that the algorithms achieve. So the goal of a system providing recommendations would be to suggest the algorithm that achieves the highest accuracy. However, there can be cases where other performance dimensions are also of interest. For example the amount of training time (i.e. the amount required by the algorithm to construct a model), the amount of test time (i.e. time required to classify an example) or the memory requirements of the algorithm. There can be more, less easy to quantify, performance measures, like the simplicity or the understandability of the models that the learning algorithms produce. Each one of these could provide a basis for algorithm suggestion. When multiply criteria should be taken into account the problem of algorithm selection can be considered as a multi-objective optimization problem. Nevertheless in practice and in what concerns meta-learning endeavors, accuracy is the one most often used. Examples of studies where the selection criteria was based on accuracy, include the STATLOG project, (Michie et al., 1994), the work on landmarking, (Pfahringer et al., 2000), and the work of Sohn (1999).

The only case in which the goal of prediction was performance of the inducers in terms of time was in VBMS,(Rendell et al., 1987). VBMS was actually the first effort to predict which algorithm from a set of available algorithms will perform better for a given classification problem, by associating simple characteristics of the datasets like the number of training examples and the number of attributes, with the performance in terms of execution time. VBMS is trained in an incremental way, i.e. it acquires experience as new classification tasks are presented to it.

There are cases in which the goal is to evaluate the learning algorithms, taking into account more than one performance dimension. For example the user might be interested in a tradeoff between training time and accuracy, willing to sacrifice some level of accuracy if he can have an algorithm that can construct the learned model much faster. In such cases there is a need for a mapping of the multidimensional performance measures onto a single scalar value, which will then be used to define the preference order among the available algorithms.

The work done by Soares (2000) provides a way to combine two performance measures of classification algorithms, namely accuracy and total execution time (i.e. the sum of the time required to construct an inductive model and the time required to test that model). The user can adjust the importance of accuracy over time via a tunable parameter. The method used to define a preference order among the inducers is called the *adjusted ratio of ratios,(ARR)*. *ARR* gives a measure of the advantage of a learning algorithm A over another learning algorithm B, in terms of the accuracy they achieve and the execution time for a specific dataset. When it comes to ranking n inducers, the *ARR* of each inducer is computed, with respect to every one of the $n-1$ other inducers, resulting in a total number of $n(n-1)$ *ARRs*. The next step is to provide a summary of these ARRs for every inducer, and this is done by the *overall mean adjusted ratio of ratios,(OMARR)*, which is actually the average of all the $n-1$ *ARRs* associated with a specific inducer. The higher the value of the *OMARR*, the higher the advantage of an inducer for the specific dataset. The preference order is defined using the *OMARR* values. The method can also provide a ranking of the inducers considering their performance in m datasets. In this case the $n(n-1)$ *ARRs* are computed for each dataset. Then for every inducer its advantage over each one of the $n-1$ inducers is computed, via the *pairwise mean adjusted ratio of ratios, PMARR*, which is the average of the related ARRs over the m datasets. Finally for each inducer the *OMARR* is calculated, but now the average is taken over the *PMARR* values. The ranking is done the same way as before, based on the value of *OMARR*.

One limitation of the ranking schema proposed by Soares is that it can only accommodate accuracy and time as performance measures. A more flexible schema is proposed in (Nakhaeizadeh & Schnabl, 1997), based on Data Envelopment Analysis, (DEA). The proposed method can incorporate any number of performance criteria. Two types of performance measures are considered, those that measure positive properties of the algorithms, i.e. more is better, for example accuracy, and those that measure negative properties, i.e. less is better, like training time. In DEA the positive properties are called output components and the negative input components. They define the *efficiency* of a learning algorithm as a weighted sum of the output components over the input components. The weights of the input and output components are computed for each learning algorithm, so that its efficiency is as close as possible to one, under the constraint that there is no other inducer for which the same set of weights would give an efficiency of more than one. The method is objective, in the sense that the weights are computed in such a way that they maximize the efficiency of each inducer. Algorithms that achieve an efficiency of one, are *efficient* algorithms. The set of efficient algorithms forms the *efficiency frontier*. The efficiency of an algorithm can only provide a partial ranking, since all the efficient algorithms have the same efficiency of one. In order to provide a complete ranking they introduce the notion of the *AP-value*. The *AP-value* is the amount by which the efficient algorithms can increase their input components while still remaining efficient. For the algorithms that are not efficient the *AP-value* is equal to their efficiency value. The final ranking of the algorithms is based on the *AP-value*.

In section 2.2.4, we will see how the various performance measures, simple or complex, have been used in a number of meta-learning efforts in order to support the user in selecting the most appropriate classification algorithm, according to the performance measure under consideration.

2.2.3 Form of Suggestion

The various approaches in providing recommendation to the user, when he has to perform a classification task on a dataset, give suggestions in one of the following forms :

1. A list of applicable algorithms

2. The best algorithm

3. A ranking of the algorithms

In the first category we have the work done in STATLOG, which was also adopted in (Brazdil et al., 1994; Lindner & Studer, 1999; Todorovski & Dzeroski, 1999). In this framework the pool of available classifiers is divided in two distinct sets: the applicable inducers and the the non-applicable inducers. Applicable inducers are expected to achieve a fair performance on the dataset under examination, while non-applicable ones are expected to have a poor performance. All the mentioned methods used the accuracy of the inducers as a performance criterion.

In the second category we classify all the approaches where the suggestion consists of a single algorithm, that is, the algorithm which is expected to perform best on the dataset under examination, according to the performance criterion that it is used. In this category we find the work of Bensusan and Giraud-Carrier (2000) where landmarkers are used in order to predict the inducer that will achieve the lowest error, and the work of Koepf et al. (2000), who use a set of statistical and information based measures to predict again the algorithm with the lowest error.

In the third category the recommendation consists in providing a complete ranking of the available inducers. This order can be based on a simple performance measure like accuracy, as it was done in (Sohn, 1999) and (Bensusan & Kalousis, 2001). Or it can be based on a more complex performance measure that incorporates simpler ones. Examples are *zoomed ranking*, (Soares & Brazdil, 2000), which provides rankings of the inducers based on the *adjusted ratio of ratios* and the work by Paterson et al. (2001), where the suggested ranking is based on the *efficiency* score defined by Data Envelopment Analysis.

2.2.4 Constructing Recommendations from Dataset Characterizations

Having presented the possible ways to characterize a dataset, and the performance criteria which can be used, what remains is to give an overview of how dataset characterizations can be associated with the measures of performance. There are two main possibilities; the first one uses classification techniques to tackle the problem, while the second one uses regression.

In both approaches a collection of *Meta-Learning* problems is established, with the number of problems depending on the number of classification algorithms among which the recommendation is to be provided. We characterize these problems as meta-learning problems, since their purpose is to learn something about the performance of learning algorithms. The collection of the meta-learning problems will constitute the *Meta-Learning Space*. Usually it is the combined solution of the meta-learning problems that will provide the basis for the recommendation. By solution, we mean here a number of *Meta-Models* that will be constructed from the application of a learning algorithm on these problems. In the classification approach the meta-learning problems are formulated as classification problems, while in the regression approach they are formulated as regression problems. We will continue by first reviewing the various efforts that have adopted the classification approach, and then the ones that follow the regression approach. In what follows, unless otherwise specified, the performance criterion used is accuracy.

In STATLOG, for a pool of n inducers among which the selection will be performed, the Meta-Learning Space consists of n classification problems, each one associated with one of the n inducers. The instances of these classification problems consist of datasets descriptions and the class label. The class label can take one of the values, *appl, non-appl*, depending on whether the algorithm is considered to be applicable or not to the corresponding dataset, i.e. whether it exhibits high or low performance on the specific dataset. For each of the meta-learning problems, a meta-model is constructed, that can predict whether an algorithm is applicable or not to a specific dataset. The algorithm used to construct the meta-models was the rule system of c4.5, (Quinlan, 1992a). The approach is described in finer detail in (Brazdil et al., 1994). Exactly the same formulation of the meta-learning space was adopted in (Lindner & Studer, 1999; Todorovski & Dzeroski, 1999). Lindner and Studer advocate the use of a case based reasoning system to represent the meta-learning problems and perform the inductive process, but they did not implement that. At the end the meta-learning models that they induce are created via the c4.5 decision tree algorithm. Todorovski and Dzeroski, explore the use of first order inductive algorithms in order to make full use of the dataset characteristics which are computed on an attribute basis. Meta-learning approaches mentioned so far rely on propositional learners on the meta-level, which have limited representational powers. Normally one had to rely on the means of characteristics that were computed for each attribute so that they can be represented in a propositional framework. First order inducers overcome this limitation and are able to make full use of the information contained in the dataset characteristics.

Another formulation of the Meta-Learning Space is to create $\binom{n}{2}$ meta-learning problems that correspond to all the pairwise comparisons of learners from a given pool. The instances of the meta-learning problems will be also composed by the description of a dataset and a class label. But this time the class label will indicate which algorithm of the pair is more appropriate for the specific dataset. Here the goal of meta-learning will be to construct meta-models that will describe the conditions under which one inducer is preferable over another. The meta-models can then be combined in order to determine from the partial ordering, which inducer is the most appropriate, or even provide a ranking of them. This formulation was followed in (Pfahringer et al., 2000), but they stopped in the construction of the meta-models, without proceeding in their combination in order to select the most appropriate inducer. To construct the metal-models they used Ripper, a rule inducer (Cohen, 1995).

Finally in the simplest formulation of the Meta-Learning space, there is only one meta-learning problem. In this case the class label is simply the inducer that is most appropriate for the corresponding dataset. The meta-model, a global one that gives directly the conditions under which one inducer is preferable over all the others. This approach was adopted in (Bensusan & Giraud-Carrier, 2000). In what concerns the inducer used on the meta-level, they examined the performance of ten different learners.

In the regression approach, the goal usually is the direct prediction of the error or of the accuracy of an inducer from the characteristics of the dataset. Here the Meta-Learning Space consists of one regression problem for each inducer. The meta-models are established via the application of a regression algorithm. The predictions of the meta-models, can then be used to perform either algorithm selection, i.e. suggest the learning algorithm with the lowest predicted error, or algorithm ranking, return a ranking of the available inducers according to their predicted errors.

Regression methods were first used to predict the error of inducers by Gama and Brazdil (1995). They tested three different regression methods: simple linear regression, instance based regression and a piecewise linear method. They did not however use the produced regression models in order to perform algorithm selection or algorithm ranking. Sohn (1999) uses simple linear regression to predict the errors of inducers and then proceeds to rank the inducers according to their predicted errors with quite promising results. Finally Koepf et al. (2000), used regression in order to perform algorithm selection and they compared that with algorithm selection via classification, their results favored regression based algorithm selection over classification based. The datasets on which they worked were artificial datasets. A more detailed description of regression based approaches will be given in section 8.1.

There is a third approach in constructing suggestions which cannot be classified in classification or regression approaches, and consists of methods that produce rankings of inducers. Two methods fall in this category; they both use a combination of ranking schemas with variations of the nearest neighbor method. The use of the nearest neighbor method is essential in both systems, because in order to provide a ranking of the inducers for a specific dataset, they need the establishment of a set of similar datasets, i.e. sets with similar characteristics. In order to construct the ranking for the new dataset the performances of the inducers on the set of similar datasets will be considered.

The first method is that of zoomed ranking introduced in (Soares & Brazdil, 2000). They use a three nearest neighbor method in order to establish a similar set of datasets with the one for which a ranking should be constructed. After establishing the set of similar datasets they apply the ranking method based on the adjusted ratio of ratios, already described in section 2.2.2, in order to determine the ranking. In a similar way Paterson et al. (2001) use a combination of three nearest neighbors and the efficiency score defined by DEA (Data Envelope Analysis), in order to provide a ranking of the inducers. Since the efficiency score cannot directly accommodate the performance of inducers among different datasets, as the adjusted ratio of ratios does, they use a weighted average of the efficiency scores of every inducer among the three nearest neighbors. The weights are determined according to the distance of each of the three nearest neighbors from the dataset under examination. The ranking of the inducers is then based on the weighted average. As performance measures from which the efficiency score is computed they used, accuracy, train time, test time and hypothesis size. Hypothesis size is the model size that an inducer produces for a specific dataset.

2.2.5 Related Work

In this section, we will present work that is related to the problem of algorithm selection but could not be characterized along the dimensions that we have established, in the previous sections.

In a similar line of research that also involves dataset characterization, Aha (1992), proposes a methodology for constructing rules that describe the relative performance of inducers starting from a specific dataset for which they exhibit significant difference in performance. One of the key assumptions of the proposed method is the availability of information concerning the inner structure of the dataset. This information includes characteristics like the number of instances, number of classes, the number of prototypes per class and the number of relevant and irrelevant attributes and will be used to construct a number of, hopefully similar, artificial datasets that will populate the space around the initial dataset. The construction of the artificial datasets is done by slight perturbations of the parameters that describe the initial dataset. The characteristics used on the construction of the artificial datasets along with the performance of the inducers on them will constitute a series of meta-learning problems, from which rules will be induced by the application of a classification algorithm. The rules will describe how the relative performance of the inducers is determined by the characteristics of the datasets.

Sleeman et al. (1995), present an expert system called Consultant. The system is built to support the use of a Machine Learning Toolbox, an integrated architecture of ten machine learning tools. It relies heavily on close interaction with the user; it asks several questions trying to determine the nature of the application and the nature of the data. It does not examine the data. At the end of the interaction a list of possible algorithms is presented and the user may select one of them. The system is not expandable (i.e. it can not incorporate new algorithms) and in the end relies heavily on the user for selecting the appropriate algorithm.

Constructive induction systems face a number of problems similar to those involved in algorithm selection. These systems view learning as a dual search process. They perform a search both for an appropriate representation in the space of representational spaces and for an appropriate hypothesis in a specific representational space. The traversal of the representational space is done with the use of *constructive* and *destructive* operators. Constructive operators expand the representation space using attribute generation methods, e.g. numerical or logical combinations of the existing attributes. Destructive operators contract the representational space through attribute selection and attribute abstraction,(Bloedorn & Michalski, 1998). Again similar questions arise, i.e. which operators should one apply, in which order they should be applied, and on which attributes,(Bloedorn et al., 1994). The order of application of operators critically affects the quality of the final outcome. Bloedorn et al. (1993), built meta-rules, from meta-data characterizing datasets, to guide the selection of operators. According to the information source used to select operators and attributes, constructive induction methods are classified in three categories: data driven, hypothesis driven and knowledge driven methods. In data driven constructive induction, information from the training examples is used. For example in order to select attributes from which new ones will be derived one may use the information gain metric of the attributes,(Bloedorn & Michalski, 1998; Bloedorn et al., 1993). In hypothesis driven constructive induction, results from the analysis of the form of intermediate hypothesis are used. For example one may use patterns appearing in the intermediate hypothesis, in order to construct new attributes,(Wnek & Michalski, 1994). Finally in knowledge driven constructive induction, domain knowledge which is provided by experts is used.

2.3 Model Combination

A line of research parallel to algorithm selection is that of model combination. There, unlike algorithm selection where the goal is to select the single best performing algorithm for a particular dataset, the goal is to combine different classification models in order to improve accuracy.

There are many different ways to combine classification models, the simplest combination strategy is *voting*. There a number of classification models are constructed from the training data; when a new instance has to be classified each one of the models produces a prediction. The class most often predicted is considered as the class to which the instance belongs to. More elaborate voting strategies have been proposed, where the prediction of each model is weighted, usually by a quantity associated with the quality of prediction of the model.

A more elaborate combination schema, is *stacked generalization*, (Wolpert, 1992). In stacking each of the base inducers, referred to as level-0 inducers, produces a prediction for each instance of the training set, usually by an inner cross-validation procedure. These predictions along with the class labels will constitute a new training set which will be given as input to another inducer, called level-1 inducer. The goal is to construct a classification model that will be able to describe how the predictions of the level-0 inducers relate to each other, when they fail or succeed, and how they related to the class label. We have described a two level process, the one most often employed, but in theory there can be many levels of combinations. In order to classify a new instance, each one of the level-0 inducers produces its prediction, which are then given to the level-1 inducer in order to determine the class label. Note here that voting can be considered as a simple variant of stacking. Another variant of stacking is to use, instead of the class labels, the probability distribution of the class labels as they are calculated by the level-0 inducers. This variant has been found to produce better results than using only the predicted class label, (Ting & Witten, 1997). Another variant of stacked generalization is *cascade generalization*, proposed by Gama and Brazdil (2000). There the set of initial attributes is extended by the probability distributions of the class labels, produced by the level-0 inducers, and then passed as input to the level-1 inducer.

So far, all the model combination methods mentioned use the predictions of all the classification models to produce the final class label. There is another line of research in model combination, where at the end the prediction of only one of the base inducers is used to assign the class label to a new instance; this approach can be referred to as *dynamic model selection*. The goal in dynamic model selection, is to choose the most appropriate classification model for a region of the instance space. The assumption underlying this approach is that every classification model has a different region of competence within the instance space on which it was constructed. That is, one of the classification models might provide better predictions than the others for specific regions of the instance space, and vice versa. When a new instance has to be classified, one must identify which is the best performing model for the region in which the instance falls, and then use that model to produce the final prediction.

We must note here the similarity of this approach with the problem of algorithm selection. In algorithm selection the instance space consists of all the possible datasets, and the goal is to find for each instance, i.e. dataset, the best performing classification algorithm. In dynamic model selection the goal is, for each instance within a particular dataset, to find the classification model exhibiting the best performance, the one that has the highest probability of correctly classifying it.

The key component in dynamic model selection is the characterization of the different areas of the instance space according to the competence of each model. One simple way to achieve that is by using cross-validation within the training set and select the algorithm that exhibits the highest accuracy, to construct the final classification model, as proposed in (Schaffer, 1993). This is a rather crude approach, since it considers the whole training set to perform the selection and characterize the instance space. A more elaborate approach is to use a meta-learner to characterize the various regions of the instance space according to the performance that the classifier exhibits. Usually this characterization is done within the training set via a cross-validation procedure. Then when a new instance is to be classified the model or models constructed by the meta-learner is(are) used in order to determine in which region of the instance space it falls, and select accordingly the appropriate model. For example Woods (1997), uses a nearest neighbor approach in order to determine the region in which a new instance falls, the set of nearest neighbors is then used to get a local accuracy estimate of the base level classifiers, and the one with the highest accuracy is selected to provide the class label. Koppel and Engelson (1996), use a decision tree to characterize the accuracy of the classification models for the different regions of the instance space. Whenever a new instance has to be classified a decision tree is used for each of the available classification models, in order to determine the area of the instance space in which the instance belongs to, and the accuracy of the corresponding model for that area. The model with the highest performance is then chosen to classify the instance. Todorovski and Dzeroski (2000), also use a decision tree in order to select which model should be applied to a specific instance. Here the leaves of the decision tree, which they call meta-decision tree, use properties of the predictions of the base classifiers in order to select which base model should be selected. These properties are based on the probability distribution of the class labels that each base model produces.

In a slightly different approach which can still be characterized as dynamic model selection, the instance space is characterized based on the predictions that the base classifiers produce. Whenever a new instance is to be classified all the models are applied to it and their predictions are used in order to locate in a look-up table all the instances that exhibit the same pattern of predictions in the training set. Merz (1995) then uses an estimate of the performance of the models in that set of instances in order to select the best performing model to produce the classification, while Huang and Suen (1995) simply return the majority class that corresponds to that set of instances. This is not a majority vote, since the predictions of the models are used in order to define an area in the instance space and do not vote to classify the instance. Ting (1997) uses inducer specific measures to characterize the instances of the instance space according to their *typicality*. The more typical an instance is for a specific learning algorithm, the higher our confidence is in the prediction assigned to it by that learning algorithm. When a new instance has to be classified its typicality is computed for each inducer, then the inducer for which the instance has the highest typicality is chosen to provide the prediction for the specific instance.

Another approach to model combination are *hybrid classification algorithms*. This category of classification algorithms includes inducers that integrate different learning paradigms within a single structure without relying on individual application of base inducers. For example Tcheng et al. (1989) present the CRL/ISO, a system that uses optimization in order to search in the inductive bias space. The CRL component is a learning system that manages a set of diverse inductive biases, including multiple decomposition strategies, multiple function approximation strategies and multiple decomposition evaluation strategies, and produces hybrid concept representations. The ISO component is the optimization component that searches in the inductive bias space for an optimum bias. Ting (1994) produced a system that combines decision trees with instance based learning in order to improve the performance of decision trees for the problem of small disjuncts. However the two algorithms are not tightly coupled, since they are trained independently. The decision as to which one of the two should be used in order to classify an instance is based on whether that instance belongs to a small disjunct; if this is the case the instance based classifier is used, if not the decision tree model will be used. Domingos (1996) presents RISE, a system that tightly integrates rule and instance based induction. In this system instances are considered as rules of maximal specificity so there is no distinction between rules and instances. Brodley (1995) integrates three different learning approaches, univariate tests, linear discriminants and instance based, under a decision tree structure. At each node of the decision tree *if-then* rules are used to guide the selection of the appropriate learner based on data characteristics and the performance of the learners. Kohavi (1996) describes NBtree, a combination of decision tree and Naive Bayes inducer, where the leaves of the decision tree are replaced by a Naive Bayes classifier. Gama and Brazdil (2000) under the framework of cascade generalization describe local cascade generalization, where they combine a decision tree structure with Naive Bayes and linear discriminants. At each node of the decision tree the current set of attributes is extended by the probability distributions of the class labels that the base inducers provide, trained only on the examples that belong to that decision node. The new attributes are propagated in the tree structure and they are treated as normal attributes.

Another way to perform model combination is via the use of models constructed from the same inducer but on different versions of the training set, usually created by same kind of resampling. The two main representatives of this category of algorithms are *bagging* and *boosting*. The first one was introduced by Breiman (1996) and creates replications of the initial training set of equal size by sampling with replacement. From each replicate set a classifier is constructed and the final prediction is given by the voting of the individual classifiers. Boosting was first introduced by Schapire (1990) as a method for boosting the performance of a weak learning algorithm. AdaBoost was introduced in (Freund & Schapire, 1996). The category of boosting algorithms uses a weighted voting schema among classification models constructed by the same inducer, where the weights are determined on the basis of the performance of the classification models on the training set. The classification models are constructed sequentially starting from the initial set of training instances. The main idea of boosting is that it gives more importance to the training examples which are misclassified; this is done by the incorporation of a weighting mechanism where each example is assigned a weight. At each iteration of the training phase the weights of the training instances which are misclassified is increased by a quantity which is inversely proportional to the total error on the training set. The instance weights can then be used by a classification algorithm that is able to directly incorporate weights of instances in the learning process, or used in order to resample the training set. The process continues for a fixed number of iterations.

2.4 Summary

The process of algorithm selection is a complicated and time intensive task as it was already exhibited in section 2.1. In order to relieve the analyst from the evaluation effort, various approaches have been proposed that directly utilize information drawn from the dataset, without having to perform extensive experimentation.

Our work falls within that framework, i.e. the automatic selection of classification algorithms based on dataset characteristics, and it covers a variety of topics within it. We took special care in the construction of a modular meta-learning space and the definition of the meta-learning problems that populate it. The dataset characteristics were chosen carefully in an effort to provide a set that can best discriminate among the performance of different inducers; furthermore we proceeded to a systematic experimentation to characterize their discrimination power. We also undertook a systematic experimentation in order to determine the most appropriate inducer for meta-level learning, and compared our set of features with various different approaches to dataset characterization. Finally we also explored the use of regression in order to select the most appropriate algorithm or provide a ranking of algorithms, and compared it with a well established method of ranking. Where our work differs from and where it resembles existing approaches will be clarified in the forthcoming chapters.

Chapter 3

The Meta-Learning Framework

The goal of this work is to provide a system which will act as an assistant for algorithm selection in the context of the classification task. The task of algorithm selection is viewed as a meta-learning task. The system builds its knowledge from a number of specific training episodes, applications of classification algorithms to a set of classification problems. The system keeps a set of registered classification algorithms, among which algorithm selection will be performed in the future.

During an initialization phase, every registered inducer is applied to each one of the available classification problems and its performance is evaluated. The morphological characteristics of these classification problems/datasets are recorded and constitute a morphology space. For a new dataset the system will suggest the most appropriate inducer, deciding on the basis of morphological similarity between the new dataset and the existing collection of datasets.

The morphological characteristics along with the performance measures of the inducers will be the building elements of a meta-learning space. Any inducer can then be applied on this meta-learning space, in order to construct inductive models, on which the system will rely to suggest the most appropriate inducer for a new unseen dataset. The construction of the meta-learning space follows closely the process of evaluation and comparison of a number of classification algorithms to a specific dataset.

3.1 Conceptual Description

The overall architecture of the system is depicted in figure 3.1. The algorithm suggestions are provided by Selector, which uses for that the Knowledge Base, (KB), available to the system. The knowledge base is simply a collection of inductive models describing the areas of competence of each of the registered inducers, and it is built during the initialization phase of the system. Whenever a new dataset is presented to the system, Selector will combine the models of the knowledge base and use them, together with the morphological characteristics extracted from the new dataset, in order to provide a suggestion.

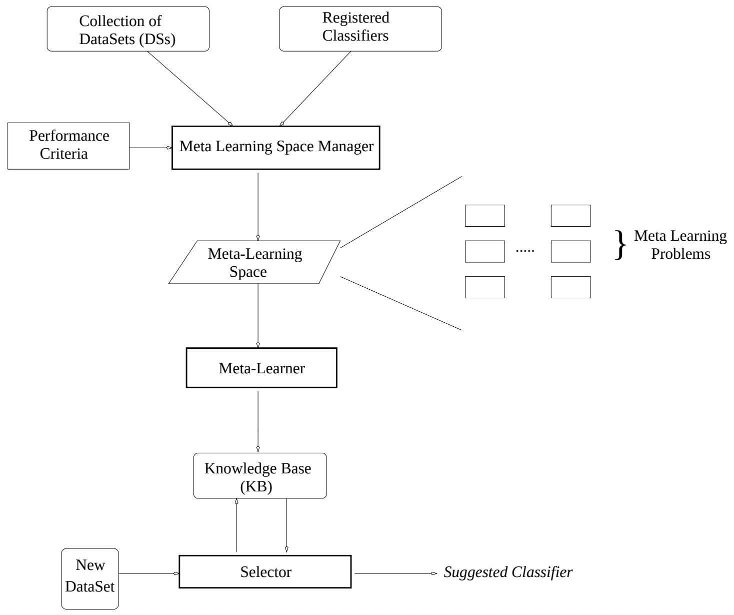

Fig. 3.1. Architecture of the system

The establishment of the knowledge base is performed by the Meta-Learner, which can be any classification algorithm. The Meta-Learner constructs Meta-Models, which are actually classification models, that associate morphological characteristics of datasets with the performance behavior of the registered inducers, performance which is measured in terms of the predictive accuracy.

Data on the performance behavior of the registered classification algorithms on the repository of datasets, are retained and managed by the Meta-Learning-Space-Manager. These performance data, along with the morphological characteristics that describe the datasets of DSs, give raise to a collection of Meta-Learning problems, a collection that constitutes the Meta-Learning-Space. It is from these meta-learning problems that the Meta-Learner, will construct the meta-models which are going to be stored in the knowledge base.

The appropriate construction of the Meta-Learning-Space is crucial, since the quality of the information contained there, will determine to a great extent the future performance of the system. This quality depends on two factors. First and most important is the set of morphological characteristics used to describe a dataset. Second the procedure via which these characteristics are mapped to the performance behavior of the inducers in order to formulated the meta-learning problems, this procedure will be described in detail in section 3.2.1.

We define *morphological characteristics* or *dataset characteristics*, as a set of structural characteristics that jointly determine the performance of a classifier on a dataset. The problem of specifying the appropriate dataset characteristics that adequately characterize the performance of a classifier can be regarded as a feature extraction problem. We need the set of features with the highest discriminatory power. It is desirable to keep this set as small as possible. However, it must be large enough to ensure that no two datasets with the same morphology have similar performance values for the same classifier. So special care must be taken in defining the right set of characteristics. The establishment of an appropriate set of data characteristics will be the subject of chapter 4. There is another alternative in establishing the most discriminatory set of features. In this we do not place any constraints in the mumber and type of the features that we use for meta-learning, but prior to using them for meta-learning we apply principal component analysis in order to get a smaller set of uncorrelated features. The main drawback of the principal component analysis is that the new features are linear combinations of the base features and the produced classification models do not directly use the initial features, being thus harder to analyze and explain.

In the remaining sections of the chapter we will give a detailed description of the various components of the system, that is how the Meta-Learning-Space is constructed and how a suggestion is provided for a new unseen dataset. We will present the process via which we populated the DSs set of datasets and the set of registered inducers among which the selection is performed. Finally we will give the evaluation measures that will be used in the forthcoming chapters to measure the quality of suggestions that the system provides.

3.2 Meta-Learning-Space

Once the morphology characteristics that will be used have been established, along with the performance measures, the construction of the Meta-Learning-Space, can take place. In this section we will give a description of the Meta-Learning-Space, after presenting the related work.

In (Michie et al., 1994; Brazdil et al., 1994; Lindner & Studer, 1999; Todorovski & Dzeroski, 1999), the methodology followed was to define for each algorithm a separate meta-learning problem. In this approach each instance of the meta-learning problem is composed of the morphological characteristics of a specific data set and one of the class labels *applicable*, *non-applicable*, describing whether the algorithm is applicable for the data set or not. In order to decide which class label should be assigned to an instance, the accuracies of all the algorithms for the specific data set are needed. Then the best accuracy is used as a basis for determining the class label of all the algorithms for that data set. If an algorithm's accuracy is found to be "much worse" than the accuracy of the best algorithm then the instance is assigned the class label non-applicable, otherwise it is assigned the label applicable.

We see two problems in this methodology, first we do not get the truly best algorithm(s), only a division of the set of algorithms to applicable and non-applicable. By truly best, we mean here that set of inducers that achieve the highest performance, and whose performance differences are not statistically significant different. Second and more important, is the way that the meta-learning problems are constructed. As mentioned above, in this framework, the accuracy is used for the assignment of class labels. The accuracy was computed by cross validation. Nevertheless, cross-validation, and in general all error estimation procedures, give only an estimate of the true accuracy, along with a confidence interval which might be quite wide if the data set on which the accuracy is measured is small. Under this framework, the assignment of classes is based on a simple comparison of the estimated accuracies, not involving any kind of test of statistical significance of the estimated differences in performance. This results in class assignments which can be erroneous, causing the meta-learning problems to be distorted by noise, thus reducing the quality of the Meta-Learning-Space and making the induction process more difficult. For a thorough description of the methodological issues involved in comparing classifiers see (Dietterich, 1998; Feelders & Verkooijen, 1995; Salzberg, 1997).

The problem of getting the best algorithm can be addressed by the various ranking systems, like ranking with zooming, (Soares & Brazdil, 2000), or DEA based ranking, (Paterson et al., 2001). The main limitation of both of the aforementioned methods is that they rely on a single global meta-model, that is based on a k-nearest neighbor approach, since in order to produce the rankings a set of similar datasets to the one under examination should be established. Moreover the single meta-model requires also the use of a unique and uniform way of describing the datasets, independently of the inducers that are involved in the selection. Nevertheless, it is quite probable that the factors that determine the relative performance of a specific pair of classifiers vary among different pairs. With a single meta-model we lose in flexibility. The same limitation of the single meta-model appears also in (Bensusan & Giraud-Carrier, 2000), where a single classification meta-model is produced.

A more flexible approach is regression based ranking as it is used in (Sohn, 1999; Bensusan & Kalousis, 2001), or regression based algorithm selection, (Koepf et al., 2000). Under the regression scenario the constraint for a single meta-model is lifted. The meta-learning-space will consist of one meta-learning problem for each inducer, thus giving rise to a distinct meta-model for each inducer. Under this scenario, it is possible to use a different set of dataset characteristics for each algorithm, i.e. that set that best describes its performance.

The way we construct the Meta-Learning-Space, closely simulates the evaluation and comparison process, followed by an analyst, when he has to select among a set of inducers the one that achieves the best accuracy. We construct meta-learning problems that correspond to pairwise comparisons of inducers. We control the class assignments on the meta-learning problems via a test of statistical significance, as it is done in a regular analysis scenario. The statistically controlled assignment of class labels results in well defined meta-learning problems, thus increasing their quality. Furthermore the formulation of the Meta-Learning-Space, based on the pairwise meta-learning problems, allows for great flexibility since different meta-models can be used to describe the relative performance of different pairs of inducers. By different meta-models we mean either models that have been constructed by a different set of dataset characteristics, or even by a different meta-learner.

3.2.1 Establishing the Meta-Learning-Space

Let us now give a description of the Meta-Learning-Space. Suppose we have a pool of n classification algorithms, from which we want to perform the selection. We create a Meta-Learning-Space containing $\binom{n}{2}$ meta-learning problems. Each one of these problems corresponds to a specific pair of classification algorithms. Each instance of a meta-learning problem corresponds to one data set from the collection of the initial datasets, DSs, and consists of the dataset's morphological characteristics and the class label. The class label is determined by a significance test on the accuracy that the two algorithms achieve and is one of (*algorithm-x, algorithm-y, tie*), depending on whether there was a statistically significant difference in favor of the first (*algorithm-x*) or second algorithm (*algorithm-y*), or no difference (*tie*).

To summarize, all the possible pairs of n inducers define a Meta-Learning-Space containing $\binom{n}{2}$ meta-learning problems. Each instance of these meta-learning problems maps the characteristics of a given dataset to a label describing a relative ranking within each inducer pair. Below we give a description of the process in pseudocode.

```
$D$ : the number of datasets, $N$ : the number of algorithms
**Establish Metalearning Space**
for $db = 1$ to $D$
    Characteristics[$db$] = get_characteristics($db$)
for $algo1 = 1$ to $N$
    for $algo2 = algo1 + 1$ to $N$
        Metalearning_Dataset[$algo1, algo2$]=Create_Metalearning_Dataset($algo1, algo2$)

**Create_Metalearning_Dataset**($algo\_x, algo\_y$)
    set significance_level = 0.05
    for $db = 1$ to $D$
        $label$=McNemarTest($algo\_x, algo\_y, db$,significance_level)
        Metalearning_DataSet[$algo\_x, algo\_y$][$db$]=(Characteristics[$db$],$label$)
```

To decide which of the three possible classes should be assigned to a specific instance of a meta-learning problem (i.e. a data set), we use stratified 10-fold cross-validation[1] and we test the significance of the difference with the binomial test. The procedure is the following:

1. Split the data set into $k = 10$ non-overlapping stratified folds

[1]The choice of 10-fold stratified cross-validation was based on the conclusions of (Kohavi, 1995), were it was stated that it is the most appropriate method for model selection

```
2. let $n = n_{xy} = n_{yx} = 0$,
   $n_{xy}$ : the number of times that *algorithm-x* is right while *algorithm-y* is wrong
   $n_{yx}$ : the number of times that *algorithm-y* is right while *algorithm-x* is wrong
   $n = n_{xy} + n_{yx}$ the number of times that the two algorithms produced different results

3. let the significance level $\alpha = 0.05$

4. for $i = 1$ to $k$ do

     • train *algorithm-x* , *algorithm-y* on the $k - 1$ folds

     • test on the remaining fold and increase $n_{xy}, n_{yx}, n$ appropriately

5. if $n_{xy} \leq n_{yx}$ then
     if $P(n_{yx}, n) \leq \alpha$ then assign label *algorithm-y*
     else assign label *tie*
   else if $n_{yx} < n_{xy}$ then
     if $P(n_{xy}, n) \leq \alpha$ then assign label *algorithm-x*
     else assign label *tie*
   else assign label *tie*
```

The fifth step of the above procedure is a binomial hypothesis test[2]. It is used to compare two algorithms in terms of their accuracy on the *same data set*, since it does not assume independence of the sets (Salzberg, 1997). Assuming that the test cases are independent then under the null hypothesis:

$$H_0 : \frac{n_{xy}}{n} = \frac{n_{yx}}{n}$$

we have:

$$n_{xy} \sim B(n, \frac{1}{2})$$

$$n_{yx} \sim B(n, \frac{1}{2})$$

$B(n, p)$ is the binomial distribution with n number of experiments and p the probability of success. $P(n_{xy}, n)$ and $P(n_{yx}, n)$ are the probabilities of having n_{xy}, n_{yx} or more successes under the null hypothesis. Success in this context

[2]Depending on the context, the binomial test can be found under the names of Sign test or McNemar test. The difference between the two is that the former is used to test for a difference between continuous variables, while the later to test for the difference between dichotomous variables. At the end both rely on the binomial distribution, or on an approximation to it

is the number of times that one algorithm predicted correctly while the other wrongly. So for example the $P(n_{xy}, n)$ is given by the formula:

$$P(n_{xy}, n) = P(s \geq n_{xy} | p(success) = \frac{1}{2}) = \sum_{s=n_{xy}}^{n} \frac{n!}{s!(n-s)!}(0.5)^n$$

where $\frac{n!}{s!(n-s)!}(0.5)^n$ is the probability of observing exactly s successes. Using directly the binomial distribution becomes problematic when the values of n, s become large, due to the presence of the factorials. The binomial distribution can be approximated, when $n > 10$, by the χ^2 distribution with 1 degree of freedom and the continuity correction term of -1 in the numerator (Yates' correction), to account for the fact that the statistic is discrete while the χ^2 is continuous, (Dietterich, 1998).

$$\chi^2 = \frac{(|n_{xy} - n_{xy}| - 1)^2}{n_{xy} + n_{yx}}$$

Strictly speaking, we are using the McNemar test of significance, since the variables that we are testing are dichotomous (correct prediction, false prediction).

The probability of falsely rejecting the null hypothesis (**TYPE I error**) is α. This means that in $\alpha 100\%$ of the cases we falsely assign the class label *TIE* resulting in $\alpha 100\%$ noise in the class labels. It is obvious that we can control that source of error by appropriately setting the value of α. Unfortunately in the case of **TYPE II error** (i.e. accept the null hypothesis when actually it does not hold), which is the type of error that we would like to control the most[3], we cannot have an estimate of the actual error since that requires knowledge of the true difference in the algorithms' performance.

The McNemar test is a non-parametric statistical test. The only assumption that it makes is the independence of the test cases (assumption which is valid when the test set is a random sample of the population). The disadvantage of the non-parametric statistical tests, when compared to tests that make use of the assumption of normality, is the fact that they are more likely to do a **TYPE II error**. However in (Dietterich, 1998), the McNemar test is compared with four other statistical tests. It is shown to have a very low **Type I error** (i.e. reject the null hypothesis when it is actually true, find a difference in accuracies when actually there is no difference). It is also shown to have a low **Type II error**, (i.e. accept the null hypothesis when it is actually false, do not spot a difference when there is one), although not the best one among the five tests examined. Taking into account the results of these study, plus the weak assumptions that the McNemar test requires, we have decided to use it in order to check the statistical significance of the differences of the algorithms.

One problem that appears due to the multiple comparisons is the multiplicity effect. We can control it by appropriately adjusting the significance level, α, as it is described in section 2.1.2.

[3]Our main goal is to spot differences when they actually exist

Meta-Learning-Space-Manager establishes the Meta-Learning-Space by applying each registered inducer on each dataset of the DSs collection, thus creating a meta-learning problem for each pair of algorithms.

Assessments on the performance of an inducer for an unknown dataset are based on the performance of the inducer for known datasets in the Meta-Learning-Space. The operability depends therefore on the density of the morphology space. Despite the number of publicly available datasets, the morphology space is very large and its density grows very slowly. In an effort to populate the space of datasets included in DSs we constructed semi-artificial datasets, using real ones as a starting point; the procedure will be given with more detail in section 3.4.

3.3 Meta-Learner and Selector: Provision of Suggestion

Meta-Learner is responsible for extracting inductive models from the Meta-Learning-Space and incorporating them into the Knowledge Base. Then, for each new dataset given to the system, the Selector module consults the Knowledge Base, triggers the inductive models and suggests the most appropriate classifier(s).

3.3.1 Establishing the Inductive Models of the KB with Meta-Learner

For the knowledge base to be created, an inductive process must be applied on every meta-learning problem of the Meta-Learning-Space. Meta-Learner applies a classification algorithm to every one of the $\binom{n}{2}$ meta-learning problem, resulting in $\binom{n}{2}$ inductive models. The produced models are stored in the KB and are later combined by the Selector to rank the algorithms, for a new data set.

```
BuildMetaModels
for $algo1 = 1$ to $N$
    for $algo2 = algo1 + 1$ to $N$
            MetaModel[$algo1, algo2$]=
            CreateMetaModel(Metalearning_Dataset[$algo1, algo2$])
```

One may easily see that we are facing again the dilemma of which classification algorithm should be used, on the problems of the Meta-Learning-Space. We could even choose to have a different classification algorithm for every meta-learning problem. As a starting point we decided to use a version of a Nearest Neighbor classification algorithm; the details of the specific implementation along with the results will be given in chapter 5. Furthermore we examined more complicated inducers, whose results will be given in chapter 6.

3.3.2 Combining the Inductive Models with Selector

Whenever a new dataset is input to the system, Selector extracts its morphological characteristics, in a preprocessing step, and feeds them to the inductive models in the KB. Each model proposes one of the two algorithms (associated with the meta-learning problem from which it was produced) or indicates a tie. The suggestions of the inductive models are then combined to impose an order on the registered algorithms. The final suggestion of the system is the classifier or the group of classifiers that get the highest rank score. A schematic description of the process is given below.

Produce a Ranking of Algorithms for a new Dataset

```
characteristics=get_characteristics($new\_db$)
rank[1..$N$]=0
for $algo1 = 1$ to $N$
    for $algo2 = algo1 + 1$ to $N$
        if $algo1$ = MetaModel[$algo1, algo2$](characteristics)
        rank[algo1]++
        else if $algo2$ = MetaModel[$algo1, algo2$](characteristics)
        rank[algo2]++
```

3.4 Populating the DSs

One of the problems that we faced was the limited number of datasets that we could use in order to populate the Meta-Learning-Space. The quality of the suggestions that the system provides, depends on the density of the Meta-Learning-Space, the more populated the space is, the higher the quality of the predictions.

In order to populate the Meta-Learning-Space we used a combination of real datasets and modifications of them. As a starting point we used 47 datasets[4], mainly from the UCI repository (Blake et al., 1998). The additional datasets were produced in the context of two big scale studies designed to explore the behavior of learning method in response to two additional dataset deficiencies, namely missing values and irrelevant attributes.

As a part of the missing values study two suites of datasets were generated. The first suite contains data that are missing completely at random (MCAR) while the second contains data that are missing at random (MAR). Data are said to be missing completely at random when their pattern of missingness is independent of the values of any features in the dataset, whether observed or incomplete. They are missing at random when the fact that they are missing depends on the values of other, completely observed, attributes. (For a more formal definition of MCAR and MAR values; the interested reader is referred to Schafer (1997).) From each original dataset, new datasets with 1%, 5%,

[4]The list of the 47 datasets can be found in section A.1 of the appendix

10%, 15%, 20%, 25%, 30%, and 40% of missing values were generated. MCAR data were produced by randomly deleting a given percentage of attribute values, whereas MAR data were produced by deleting a given percentage of values from certain features, contingent on the values of other features which are themselves completely observed. Thus the initial 47 datasets gave rise to 2*8*47=753 new ones. For a complete description of the results and the experimental design see (Kalousis & Hilario, 2000a).

To explore the impact of irrelevant attributes on the different learning algorithms, we corrupted the initial datasets by creating new attributes whose values were generated in a purely random fashion, thus ensuring their irrelevance to the class variable. This procedure was followed to produce variants with 5%, 10%, 20%, 30%, 40% and 50% of irrelevant attributes, yielding 6*47=282 new datasets. For a complete description of the results and the experimental studies see (Hilario & Kalousis, 2000).

Finally the total number of datasets used to train and test the system was $47 + 753 + 282 = 1082$, of them 7 had to be excluded from the DSs because for various reasons (either failure of some classification algorithms, or unavailability of their morphology characteristics), the full set of information required by the system was not available.

One could argue that the new datasets are morphologically close to the initial ones, so the estimation of the performance of the system is optimistic. Nevertheless it should be noted here that the performance of the inducers with respect to the new datasets differs from the performance that they exhibit in the base datasets; this makes the problem even harder since in datasets with similar features the algorithms exhibit completely different performance.

3.5 Pool of registered classifiers

The pool of registered classification algorithms, among which the selection is performed, incorporates a broad variety of learning algorithms: an orthogonal decision tree inducer from Quinlan's C5.0 (c50tree), an oblique decision tree inducer Ltree (Gama & Brazdil, 1999), two rule inducers, Ripper (Cohen, 1995) and the rule version of C5.0 (c50rules), a linear discriminant (Lindiscr), a boosting algorithm from C5.0 (c50boost), an instance-based learner (IBL), and Naive Bayes (NB) (the last two from the MLC++ library (Kohavi et al., 1996)). The choice of algorithms included in the pool was done so that they will represent a wide and diverse variety of classification algorithms with different areas of competence each.

By including eight classification algorithms, a total number of $\binom{8}{2} = 28$ meta-learning problems is created, with the application of the procedure described in section 3.2.1. Table 3.1 gives the distribution of classes algo-x, algo-y, tie for each of these problems. The last column specifies the percentage of the majority class; this will serve as the baseline or default accuracy against which to evaluate the accuracies estimated by the learned meta-models. Any reasonable model should have an accuracy that is higher than this default accuracy. If their

Table 3.1. Class Distributions for each of the meta-learning problems.

(algo–x, algo–y) pairs	algo–x	algo–y	tie	Default Accuracy
c50rules c50boost	4.47%	37.40%	58.14%	58.14%
c50tree c50boost	3.91%	38.33%	57.77%	57.77%
c50tree c50rules	13.02%	13.95%	73.02%	73.02%
Lindiscr c50boost	3.63%	64.47%	31.91%	64.47%
Lindiscr c50rules	12.09%	52.65%	35.26%	52.65%
Lindiscr c50tree	10.98%	54.70%	34.33%	54.70%
Ltree c50boost	13.21%	35.16%	51.63%	51.63%
Ltree c50rules	27.07%	14.23%	58.70%	58.70%
Ltree c50tree	25.21%	15.26%	59.53%	59.53%
Ltree Lindiscr	61.12%	6.51%	32.37%	61.12%
IBL c50boost	1.02%	64.65%	34.33%	64.65%
IBL c50rules	10.79%	49.77%	39.44%	49.77%
IBL c50tree	7.91%	52.00%	40.09%	52.00%
IBL Lindiscr	42.33%	21.21%	36.47%	42.33%
IBL Ltree	9.30%	56.65%	34.05%	56.65%
NB c50boost	2.70%	60.84%	36.47%	60.84%
NB c50rules	14.33%	51.44%	34.23%	51.44%
NB c50tree	12.09%	52.93%	34.98%	52.93%
NB Lindiscr	37.30%	21.58%	41.12%	41.12%
NB Ltree	5.58%	58.23%	36.19%	58.23%
NB IBL	29.12%	34.79%	36.09%	36.09%
ripper c50boost	1.58%	50.98%	47.44%	50.98%
ripper c50rules	8.09%	32.00%	59.91%	59.91%
ripper c50tree	2.70%	37.21%	60.09%	60.09%
ripper Lindiscr	46.14%	17.02%	36.84%	46.14%
ripper Ltree	3.44%	43.35%	53.21%	53.21%
ripper IBL	34.42%	19.44%	46.14%	46.14%
ripper NB	41.30%	16.93%	41.77%	41.77%
average				54.14%

performance is deemed acceptable, these models can then be used to provide a ranking of inducers for new datasets.

Concerning the final suggestion of the system, (i.e. the classifier or classifiers that take the first position), theoretically we can have any of the $2^8 - 1 = 255$ possible subsets of the initial set of the eight classification algorithms. Using the results of the McNemar tests we can get the true ranking of the classification algorithms for each dataset of the DSs. At the top position we observe only 80 of the 255 possible subsets of inducers. In Table 3.2 we give the distribution of classifier(s) that get the top ranking in more than 1% of the total number of the datasets. For example we can see from this table, that c50boost is the single best algorithm in 26.23% of the 1075 datasets registered in the system. Since c50boost is the classifier that most often takes the first place we will use as the default accuracy for the final suggestion of the system the percent of c50boost.

Table 3.2. Groups of inducers that were ranked at the top for more than 1% of the datasets

Group	# Datasets	Percent
c50boost	282	26.23
Ltree	140	13.02
c50rules c50tree Ltree	54	5.02
c50rules c50boost c50tree	47	4.37
c50rules c50boost c50tree Lindiscr Ltree IBL NB ripper	47	4.37
Lindiscr	41	3.81
NB	30	2.79
c50rules c50boost c50tree Ltree ripper	29	2.7
c50boost Ltree	29	2.7
c50rules c50boost c50tree Ltree IBL ripper	27	2.51
IBL	27	2.51
c50rules c50boost c50tree Ltree	25	2.33
c50rules c50boost	23	2.14
c50rules c50tree	22	2.05
c50rules	20	1.86
Ltree NB	14	1.3
c50boost IBL	14	1.3
c50boost c50tree	13	1.21
Lindiscr Ltree	12	1.12
c50tree	11	1.02
c50rules c50boost c50tree Lindiscr Ltree NB ripper	11	1.02

3.6 Evaluation Method

Since the main goal of the system is to predict which inducer(s) to use, we measure the performance of the system in terms of its predictive accuracy. The predictive accuracy we estimate is always associated with the classifier that we have chosen to apply on the meta-learning problems and the dataset characteristics that we use.

As it is described in the previous sections, the system works on two levels. The first level consists of the meta-learning models that describe the relative performance of specific pairs of learners. The second level is the combination of these models in order to provide the final suggestion (i.e. which classifier or classifiers are the best for a specific dataset). We evaluate the performance for both levels using 10-fold cross validation. In the case of the pairwise meta-learning problems the procedure is straight forward.

In the case of the final suggestion the estimation is more complicated. The meta-datasets associated with each pair of classification algorithms are split into exactly the same ten folds. That is, the i fold of all the meta-datasets will contain exactly the same datasets (i.e. the characteristics of the datasets along with their labels), with exactly the same order. In every step of the cross validation the nine folds will be used to construct the meta-learning models. The characteristics of the datasets contained in the remaining fold will be given to each of the produced meta-models and each one will output a prediction. These predictions will be combined, as described in section 3.3.2, to produce the final suggestion of the system, which will be compared with the truly best set of classification algorithms to determine whether the suggestion is successful or not.

We will use two different ways to characterize the suggestions of the system as successful or not. In the first, we consider a suggestion successful only if it matches exactly the correct class, i.e. the algorithm(s) that the system suggests is (are) exactly the one(s) that really take the first place. In this case we have a typical 0/1 loss function, and the evaluation measure that we compute is the typical accuracy, for clarity reasons we will refer to it as *strict accuracy*.

In the second the main idea is: when in the first place we have two or more algorithms whose performance is not statistically significant different, we are not that much concerned with finding all the algorithms involved in it. Instead we are satisfied if the suggested one(s) constitute a subset of the true set. With this approach we get an estimate of the percentage of cases in which the system gives an acceptable suggestion i.e. a suggestion that is a subset of the true top classifiers(s). We will call the evaluation measure that corresponds to this scenario *loose accuracy*.

3.7 Operationalization and Incrementality

If the described system is to be placed in an operational environment it should be adaptable and able to update its knowledge in an incremental way. It should be possible to incorporate new inducers whenever they become available and improve its performance incrementally as it faces new learning episodes.

The incorporation of a new inducer requires the application and evaluation of the new inducer on all the datasets of the DSs used to construct the initial KB of the system. In a next step all the pairwise meta-learning problems associated with the new inducer will be constructed and the respective meta-models will be produced. The steps involved in the incorporation phase of a new algorithm are not different from the initialization phase of the system. They have extensive computational cost which mainly comes from the extensive evaluation phase of the new inducer on all the available datasets of the DSs, but they are indispensable if we want to have an initial knowledge that will guide the use of the new algorithm.

The system should be able to benefit, and improve its knowledge, from new learning episodes. That is whenever the user asks for a suggestion on a new dataset, and finally evaluates a set of inducers on the new dataset, the system should be able to exploit the evaluation results to improve the quality of its KB. This is not straightforward since the complete results will not be available for all the inducers of the initial pool of inducers on every new dataset. The users typically evaluate only a limited number of inducers on a new dataset. Two options exist here, the first one leads to an incomplete but still operational KB, and the second one to a complete KB with the cost of extra computational requirements.

In the first option the system updates only these pairwise meta-models that correspond to the pairs of those algorithms that the user has evaluated on the new dataset. The result of this partial update is that some of the meta-models will be more complete and will better discriminate among the algorithms involved in the corresponding pairs, i.e. they will provide predictions of better quality. Apart from that there is no other implication, since the predictions of the meta-models are combined independently, and the way and the data from which each meta-model was constructed is not relevant.

In the second option the system performs a complete evaluation of all the available algorithms in the new dataset, probably in a batch mode, guarantying thus that all the meta-models have the same quality and are constructed using the full amount of available information. Both options should be available and it will be left to the analyst to decide which one he prefers every time. In any case if the system is able to meta-learn from new learning episodes it will not only improve the quality of its KB but on the same time reflect in a much more precise way the morphologies of the datasets that the user most often deals with, thus being able to provide more accurate suggestions.

Chapter 4

Description of Datasets

What is crucial for the performance of the system is the appropriate selection of the characteristics that will be used to describe the datasets. These should describe morphological characteristics of the datasets that affect the performance of classification algorithms. Different inducers exhibit different sensitivity to specific idiosyncrasies of the datasets. What we want to do is model how these idiosyncrasies affect the relative performance of the different inducers. For example inducers exhibit varying degrees of sensitivity to the presence of irrelevant attributes. Nearest Neighbor approaches are very sensitive to them, while decision tree and neural network algorithms are quite robust since they posses internal mechanisms that perform attribute selection. Another example is the distinction between numerically oriented approaches, like neural networks, or linear discriminants and symbolic based ones like decision trees or rule inducers. With the former being more appropriate to datasets where the attributes are mainly numeric and the later more appropriate for datasets where the attributes are mainly symbolic. We will strive for a set of characteristics that describe as completely as possible these factors. Before continuing to the description of the characteristics that we used let us shortly examine the related work in characterizing datasets.

4.1 Related Work

The first attempt to characterize datasets in order to predict the performance of classification algorithms was done by Rendell et al. (1987). The approach was very simple and so were the characteristics that they used, namely the number of features and the number of examples. The goal was to predict the execution time of the classification algorithms.

In STATLOG, Michie et al. (1994), studied the performance of twenty three different learning algorithms on more than twenty different datasets. As a byproduct of STATLOG an effort was done to predict the error of classification algorithms using the datasets characteristics. This approach was further examined in (Brazdil et al., 1994; Gama & Brazdil, 1995), mainly in terms of using different meta-learners but the set of dataset characteristics was always the same and no effort has been done to improve it. Here we will give a brief presentation of the dataset characteristics that were used. They distinguish three categories of dataset characteristics, namely, simple, statistical and information theory based. Statistical characteristics are mainly appropriate for continuous attributes, while information theory based are more appropriate for discrete attributes. The full list of characteristics established in the framework of STATLOG is the following:

- simple characteristics
  - Number of examples, (n)
  - Number of attributes, ($attr$)
  - Number of classes, (cl)
  - Number of binary attributes, (bin)
- Statistical Characteristics
  - Standard deviation ratio, ($SD.ratio$)
  - Mean absolute correlation of attributes, ($\widehat{|\rho|}$)
  - First canonical correlation, (ρ_{max})
  - Fraction separability due to first canonical correlation, ($frac1$)
  - Mean Skewness of Attributes, ($\hat{\gamma}$)
  - Mean Kurtosis of Attributes, ($\hat{\beta}$)
- Information theory characteristics
  - Entropy of class, ($H(C)$)
  - Mean Entropy of Attributes, ($\widehat{H(X)}$)
  - Mean Mutual Information of class and attributes, ($M\widehat{(C,X)}$)
  - Equivalent number of attributes, ($EN.attr$)
  - Noise-signal ratio, ($NS.ratio$)

Since we are also going to include all these characteristics in our characterization, their complete and detailed description can be found in section 4.2.

One of the main conceptual limitations of the STATLOG approach was the fact that most of the characteristics that they used to describe the datasets were averages over the number of attributes. For example to describe the correlation between continuous attributes they used the correlation coefficient and at the end they reported only the average of all the correlation coefficients. Clearly this results in a great loss of discriminating power, since completely different distributions of the correlation coefficients could result in the same mean. This was done for any characteristic that was computed on an attribute basis. Furthermore they did not examine associations between discrete attributes, and between continuous and discrete attributes.

Lindner and Studer (1999), keep the same meta-learning framework as the one introduced in STATLOG and extend the set of dataset characteristics. They introduce a variety of new dataset characteristics:

- new measures that are more appropriate for testing the assumption of normal distribution based on the BHEP-test

- measurements about the location and dispersion of numeric attributes like the minimum,maximum, mean, median, α trimmed mean, empirical quantiles,standard deviation, interquartile range and median absolute deviation

- measures that describe attribute-class associations: joint entropy, conditional entropy, information gain ratio, gini-index, relevance measure and the g-function

- finally as a measure of class differences they used Wilks-Lambda which examines the differences of the centers of the classes.

The main goal of the study is the identification from the set of characteristics that they have used of the ones that affect the performance of classifiers, always in the meta-learning framework of STATLOG. They finally came up with a set of characteristics that does not include any of the dispersion and location measures, and from the ones that describe the association between attributes and class, the only discriminating one was mutual information. A possible reason for the rejection of most of the measures they introduced, might be the fact that in order to use them for meta-learning they had to rely on their means, losing again valuable information about the distribution of the measures.

A completely different approach to characterizing datasets called landmarking appeared in (Pfahringer et al., 2000; Bensusan & Giraud-Carrier, 2000). They use the performance of simple learners which they call landmarkers to describe a dataset. The intuitive idea behind landmarking is to associate the performance of specific learners with the performance of landmarkers. That is, if landmarker A outperforms landmarker B on a specific task then learner X will also outperform learner Y on this task. A crucial issue in landmarking is the appropriate choice of the simple learners. It has to be ensured that the chosen landmarkers have quite distinct learning biases. In order to describe a dataset they use the following landmarkers :

- Decision node : A single decision node based on the attribute that maximizes the information gain ratio.

- Worst node : Same as above but now based on the attribute that minimizes the gain ratio.

- Randomly chosen node : Same as above, but now the attribute is randomly selected.
- Naive Bayes : The classical Naive Bayes learner.
- 1-Nearest Neighbor : The classical nearest neighbor learner.
- Elite 1-Nearest Neighbor : A nearest neighbor where the attributes that are taken into account for classification are limited to the subset of the most informative attributes, as they are determined by the information gain ratio.
- Linear Discriminant : A classical linear discriminant algorithm.

The performance of landmarkers is estimated through the use of ten fold cross-validation, resulting in an elevated computational cost, especially in the case where landmarkers are full fledged learners like Naive Bayes, Linear Discriminants and the different versions of nearest neighbors.

4.2 Dataset Characteristics

In this section we will present the characteristics that we will use to describe a dataset. We can cluster them in the following categories:

- characteristics that describe the nature of attributes
- characteristics that describe attributes
- characteristics that describe associations between attributes
- characteristics that describe associations between attributes and the target variable
- others

In general we use characteristics that are appropriate either for nominal or for continuous attributes. To describe nominal attributes and their associations we use mainly information based measures, while for the continuous we use statistical measures. This results in a non-unified treatment of the attributes of a dataset. Furthermore this causes a problem with the description of associations between nominal and continuous attributes. In an effort to describe associations between discrete and continuous attributes we will use characteristics that are used in analysis of variance. Table 4.1 gives the full set of dataset characteristics that we will use to describe a dataset. Their complete definition will be given in the forthcoming sections. Whenever a characteristic is introduced that will be a part of the dataset characteristics its entry number in Table 4.1 will be also provided.

Table 4.1. Dataset Characteristics, the superscript ^{s} is used to indicate the characteristics that where used in STATLOG.

No	*Characteristics*	Notation
1^s	number of classes	cl
2^s	number of attributes	$attr$
3^s	number of instances	n
4	dimensionality of the dataset	$dim = \frac{attr}{n}$
5	number of missing values	$mvals$
6	percentage of missing values	$\%mvals = \frac{mvals}{attr \times n}$
7	# nominal attributes	nom
8..11	max, min, mean, stdv	$max.val, min.val$
...	of nominal attributes distinct values	$mean.val, stdv.val$
12..21	concentration histogram	$[\tau_1..\tau_{10}]$
...	of discrete attributes	
22	non computable concentration	τ_{NaN}
23..32	concentration histogram	$[\tau_{C_1}..\tau_{C_{10}}]$
...	of discrete attributes and class	
33	non computable concentration	$\tau_{C_{NaN}}$
34	# continuous attributes	con
35..44	correlation histogram	$[\rho_1..\rho_{10}]$
45	non computable correlation	ρ_{NaN}
46..55	missing values histogram	$[mvals_1..mvals_{10}]$
56	percent of continuous attributes	$\%con = con/attr$
57	percent of discrete attributes	$\%nom = nom/attr$
58^s	Binary Attributes	bin
59^s	Variation from	$frac1 = \frac{\lambda_1}{\sum_i \lambda_i}$
...	first linear discriminant	
60^s	First Canonical Correlation	$\rho_{max} = \sqrt{\frac{\lambda_1}{1+\lambda_1}}$
61^s	Mean Skew	$\hat{\gamma} = \frac{\sum_{i=1}^{con} \gamma_i}{con}$
62^s	Mean Kurtosis	$\hat{\beta} = \frac{\sum_{i=1}^{con} \beta_i}{con}$
63^s	Class Entropy	$H(C)$
64^s	Mean Attribute Entropy	$\widehat{H(X)} = \frac{\sum_{i=1}^{nom} H(X_i)}{nom}$
65^s	Mean Mutual Information	$\widehat{MI(C,X)} = \frac{\sum_{i=1}^{nom} MI(C,X_i)}{nom}$
66^s	Equivalent number of attributes	$EN.attr = \frac{H(C)}{\widehat{MI(C,X)}}$
67^s	Noise to signal ratio	$NS.ratio = \frac{\widehat{H(X)} - \widehat{MI(C,X)}}{\widehat{MI(C,X)}}$
68^s	Mean Multiple Attribute Correlation	$\hat{R} = \frac{\sum_{i=1}^{con} R_i}{con}$
69^s	SD.ratio	$SD.ratio = exp(\frac{M}{con \sum_{i=1}^{cl}(n_i - 1)})$
70..79	$p - value$ histogram	$[p - val_1..p - val_{10}]$
...	of continuous attributes	
80..89	$p - value$ histogram	$[p - val_{C_1}..p - val_{C_{10}}]$
...	of continuous attributes and class	

4.2.1 Attribute Type

The balance between numerical and nominal attributes is important since some algorithms are better suited for numerical domains and others for symbolical. To describe the nature of the attributes of a dataset we used various measures. These include the number of continuous, *con*, (*No 34*), and nominal attributes, *nom*, (*No 7*), their percentages with respect to the total number of attributes, *%con*, *%nom*, (*No 56,57*). Another measure falling in this category is the number of binary attributes, *bin*, (*No 58*), that is the number of binary attributes that we get when all nominal attributes are presented using local binary encoding. This measure is crucial especially in the case of numerical based classification algorithms since they make use of local binary encoding when they deal with discrete attributes, a fact that gives them a disadvantage since it increases the dimensionality of the problem, while keeping the same number of training examples. Similarly to the *bin* characteristic, we determine for each nominal attribute the number of distinct values that it has. Then from all the nominal attributes we compute the maximum, minimum, mean and standard deviation of the number of distinct values, *max.val*, *min.val*, *mean.val*, *stdv.val*, (*No 8-11*).

4.2.2 Attribute Description

Let us introduce some notation that will be used to describe the measures that we are going to use. Consider two nominal attributes X, Y with I, J distinct values respectively. We display their joint distribution with a contingency table having I rows for variable X and J columns for variable Y. The probability distribution $\{\pi_{ij}\}$ is the joint distribution of X and Y. The marginal distributions of X and Y are given by the row and column totals obtained by summing the joint probabilities, and they are denoted with $\{\pi_{i+}\}$ (for row variable X, $\pi_{i+} = p(X = x_i) = \sum_j \pi_{ij}$) and $\{\pi_{+j}\}$ (for column variable Y, $\pi_{+j} = p(Y = y_i) = \sum_i \pi_{ij}$). The marginal distributions are single variable distributions and do not contain any information for the association between the two variables. The conditional distribution of Y given X is $\{\pi_{j|i}\}$ where $\pi_{j|i} = P(Y = y_j|X = x_i)$.

If X is a continuous variable we denote by $\mu_X, \sigma_X, \sigma_{XX}$, its mean, standard deviation, and variance. The sample mean, standard deviation and variance are denoted by $\widehat{\mu_X}, \widehat{\sigma_X}, s_{XX}$. The covariance and sample covariance of two continuous attributes X, Y are denoted by σ_{XY} and s_{XY}.

The *entropy*, $H(X)$, of a discrete attribute X is a measure of randomness or dispersion of the attribute and is given by

$$\begin{aligned} H(X) &= -\sum_i p(X = x_i) log_2(p(X = x_i)) \\ &= -\sum_i \pi_{i+} log_2(\pi_{i+}) \end{aligned}$$

The entropy takes its highest value of $-log_2(\pi_{i+}) = log_2(I)$ when all the I

distinct values of X have an equal probability of appearing. In the extreme case where one attribute takes only one value then its entropy is 0, this attribute brings no useful information. So the entropy of an attribute X lies in the interval $[0, log_2(I)]$, and the length of the interval depends on the number of distinct values of the attribute. The more uniform the distribution of an attribute is, the higher the value of its entropy; the less uniform the lower the value of the entropy.

To characterize continuous attributes two measures from descriptive statistics are used, skewness and kurtosis. Both are measures of departure of a given distribution from normality. The assumption of normality is made by various learning algorithms, e.g. linear discriminants, Naive Bayes.

Skewness is the lack of symmetry in a probability distribution. Positive skewness indicates a distribution with an asymmetric tail extending towards more positive values. Negative skewness indicates a distribution with an asymmetric tail extending towards more negative values. For example, a normal or a uniform distribution have zero skewness because they are symmetric about their mean. An exponential distribution has positive skewness equal to 2. Skewness is defined as the third moment of the distribution divided by the third power of the standard deviation:

$$\gamma = \frac{E(X - \mu_X)^3}{\sigma_X^3}$$

Kurtosis is a measure of how "fat" a probability distribution's tails are, measured relative to a normal distribution having the same standard deviation. A distribution is said to be leptokurtic if its tails are fatter than those of a corresponding normal distribution, and platykurtic if its tails are thinner than those of a normal distribution. The normal distribution has a kurtosis of 3, the uniform $\frac{9}{5}$ and the exponential 9. The kurtosis of a distribution is defined as the ratio of the fourth moment of the distribution to the fourth power of the standard deviation

$$\beta = \frac{E(X - \mu_X)^4}{\sigma_X^4}$$

Since the learning algorithms that make the assumption of normality do that on a class basis, i.e. attributes are assumed to follow a normal distribution within each class, we compute both the kurtosis and skewness of an attribute on a class basis.

4.2.3 Attribute Associations

Classification algorithms are affected by the degree of redundancy within a dataset. The redundancy comes from the fact that the attributes of a dataset are not always independent. The following metrics provide a way to measure it, to some extent, since they quantify the strength of the relationships between attributes. The list of metrics given below is in no way exhaustive, there are lot more measures that can describe attribute relations. Furthermore we restrict to metrics for linear relationships between continuous attributes, and metrics that describe associations only between pairs of discrete attributes. Higher order correlations are not examined, nor associations involving more than just two discrete attributes.

To measure the association between nominal attributes we used *Goodman and Kruskal's* τ otherwise known as the *concentration coefficient* (Agresti, 1990). Using the notation given in 4.2.2 the concentration coefficient between two nominal attributes, X, Y, with I, J, distinct values is defined as :

$$\tau_{xy} = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{\pi_{ij}^2}{\pi_{i+}} - \sum_{j=1}^{J}\pi_{+j}^2}{1-\sum_{j=1}^{J}\pi_{+j}^2} \tag{4.1}$$

Here X is considered to be the independent attribute and Y the dependent. The interpretation given to τ_{xy} in (Agresti, 1990) is that it describes "the proportional reduction in the probability of an incorrect guess predicting Y using X." τ_{xy} takes values in the interval $[0, 1]$. The higher the value of τ_{xy} the stronger the association is, in the sense that we can guess Y much better when we know X than when we don't. We have to note here that the concentration coefficient is not symmetric i.e. $\tau_{xy} \neq \tau_{yx}$. So it is not enough to compute the $\binom{nom}{2}$ coefficients, corresponding to all pairs of nominal attributes, but for each pair we have to compute both coefficients.

We measure the association between two continuous variables, X, Y, using the *correlation coefficient* ρ_{xy}. ρ_{xy} is a measure of the linear relationship between the two variables. The correlation coefficient takes values in the interval $[-1, 1]$ and it is symmetric, i.e. $\rho_{xy} = \rho_{yx}$. A value of 1 indicates a perfect positive linear relationship, a value of -1 a perfect negative linear relationship and a value of 0 the complete absence of a relationship. The correlation coefficient is given by the following formula:

$$\begin{aligned}\rho_{xy} &= \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}} \\ &= \frac{\sigma_{XY}}{\sqrt{\sigma_{XX}\sigma_{YY}}}\end{aligned}$$

For the continuous attributes we do not only examine correlations between pairs of attributes but we also examine the correlation between each attribute and the linear combination of the rest. This is done with the use of the *multiple correlation coefficient* (Engels & Theusinger, 1998). Given continuous attributes $X_1, X_2, ...X_{con}$, the multiple correlation coefficient, R_i, between attribute X_i and the multivariate variable $Z_i = (X_1, ..., X_{i-1}, X_{i+1}, ..., X_{con})$ is the maximal correlation coefficient between X_i and some linear function $Z_i\alpha$ of Z_i, (the maximum is taken over all possible nonzero vectors α). R_i is given by :

$$R_i = argmax_{\alpha\neq 0}\frac{Cov(X_i, Z_i\alpha)}{\sqrt{Var(X_i)Var(Z_i\alpha)}}$$

$$
\begin{aligned}
&= argmax_{\alpha \neq 0} \frac{\sigma_{X_i Z_i \alpha}}{\sqrt{\sigma_{X_i X_i} \sigma_{Z_i \alpha Z_i \alpha}}} \\
&= argmax_{\alpha \neq 0} \frac{\alpha' \Sigma_{X_i Z_i}}{\sqrt{\sigma_{X_i X_i} \alpha' \Sigma_{Z_i Z_i} \alpha}} \\
&= \sqrt{\frac{\Sigma'_{X_i Z_i} \Sigma^{-1}_{Z_i Z_i} \Sigma_{X_i Z_i}}{\sigma_{X_i X_i}}}
\end{aligned}
$$

Where

- a is a column vector of dimensions $(p-1) \times 1$,
- $\Sigma_{Z_i Z_i}$, is the covariance matrix of Z_i with dimensions $(p-1) \times (p-1)$,
- $\Sigma_{X_i Z_i}$ is a $(p-1) \times 1$ vector that contains the covariances of X_i, with each of the $p-1$ attributes of Z_i.

The maximum value of R_i is attained for $\alpha = \Sigma^{-1}_{Z_i Z_i} \Sigma_{X_i Z_i}$. Let E be the random vector consisting of the attributes $X_1, X_2, ..., X_{con}$. Then the sample multiple correlation coefficient, $\hat{R}_i$ between attribute X_i and Z_i is defined as

$$\hat{R}_i = \sqrt{\frac{S'_{X_i Z_i} S^{-1}_{Z_i Z_i} S_{X_i Z_i}}{s_{X_i X_i}}}$$

If S_{EE} is the unbiased sample covariance matrix[1] of the attributes of E then $S_{Z_i Z_i}$ is the submatrix that we get from S_{EE}, when we remove column and line i, which corresponds to the sample covariance matrix of the Z_i attributes. $S_{X_i Z_i}$ is the $(p-1) \times 1$ vector constructed from the i^{th} column of the S_E table, removing the element of the i^{th} line, and contains the sample covariances between attribute X_i and each of the $p-1$ attributes of Z_i. The multiple correlation coefficient takes values in the interval $[0, 1]$. A value of 1 means that X_i is a linear combination of the attributes of Z_i. A value of 0 indicates that X_i is linearly independent of Z_i.

Having described the associations between discrete attributes, and then the associations between continuous, there is still a gap. Namely we need a way to describe associations between continuous and discrete attributes. In order to do that we are going to use the F-distribution the same way as in Analysis of Variance (ANOVA), but we will not proceed to a significance test but rather report the $p-value$, as a measure of the association. With ANOVA we examine whether one independent categorical variable affects a dependent quantitative one. We can only examine one way associations, that is whether the values of a continuous variable depend on the values of a discrete variable, but not the other way around, i.e. how a continuous variable affects a discrete one. Let X be a discrete variable with I distinct values and Y a continuous one. We want to examine whether the values of X affect the values of Y. The I different values of X define I groups on Y. ANOVA examines whether the means of the

[1] $S_{EE} = \frac{1}{n-1} \sum_n (E_i - \widehat{\mu_E})(E_i - \widehat{\mu_E})'$

I groups defined on Y are different, based on a comparison of the between group variance with the within group variance. The *between group variation*, $SS(B)$[2], measures the variation of the group means, μ_{Y_i}, of the variable Y, around the global mean, μ_Y, and it is given by :

$$SS(B) = \sum_i n_i(\mu_{Y_i} - \mu_Y)^2 \tag{4.2}$$

where n_i is the number of observations of Y that belong to the i group. We could consider the between group variation, as the variation we get when all the members of a group are identical to the mean of the group. Since we have I distinct groups and we get one data value for each group (i.e. the sample mean of the group, μ_{Y_i}), the degrees of freedom of $SS(B)$ are $I-1$, so the *between group variance*, or else denoted Mean Square between groups, is $MS(B) = \frac{SS(B)}{I-1}$. The *within group variation*, $SS(W)$, is the sum of the groups variations around their corresponding means. Each group variation is given by :

$$SS_i = \sum_{y \in i}(y - \mu_{Y_i})^2 \tag{4.3}$$

so the within group variation is given by :

$$SS(W) = \sum_i SS_i \tag{4.4}$$

The degrees of freedom of $SS(W)$ is the sum of the degrees of freedom of all the SS_i. Since its one of them has $n_i - 1$ degrees of freedom, $SS(W)$ has $n - I$. And the *within group variance* is $MS(W) = \frac{SS(W)}{n-I}$. ANOVA examines the ratio of between group variance to within group variance. When the variance between groups is much larger compared to the variance within the groups, then the means of the groups probably are different. The ratio $\frac{MS(B)}{MS(W)}$ follows the F-distribution with $I-1$ and $n-I$ degrees of freedom. From the value of the ratio we retrieve the corresponding probability (*p-value*) of observing the specific value, under the assumption that the group means are equal. A *p-value* close to zero indicates that the initial assumption of means equality should be rejected, consequently the different values of the X variable define groups on Y that are different. A *p-value* close to one indicates that the assumption is true, so the values of X do not define different groups on Y. Although *p-value* is by no means a measure of association, however it gives an indication of whether the X variable affects Y and also an indication of the level of the association. Since the quantity is a probability its values are in the interval $[0, 1]$.

4.2.4 Attribute-Class Associations

A very important aspect, probably the most important one, in classification problems, is the amount of information that attributes bring about the class.

[2]SS stands for Sum of Squares

It is obvious that the higher the information content of the attributes about the class, the easier the task of classification is. The information content of attributes with respect to the class varies for different attributes. Irrelevant attributes do not contain any useful information about the class. Different learning algorithms exhibit different degrees of resilience to irrelevant attributes (Hilario & Kalousis, 2000). Decision trees or neural networks for example are considered quite robust with respect to irrelevant attributes, due to their internal mechanisms, (feature selection and weight adjustment respectively), although this behavior was not confirmed for the neural networks in the aforementioned study. On the other hand simpler algorithms like nearest neighbors are sensitive to irrelevant attributes. Datasets with attributes that individually have low information content about the class are a challenge to most classification algorithms. Usually low information content indicates the need for new higher order attributes. If this is the case feature construction algorithms have a better chance of achieving higher performance than simple ones.

One of the most common measures of the information that one attribute X conveys about another attribute Y is the *mutual information* $MI(Y, X)$. Mutual information describes the reduction in the uncertainty of Y due to the knowledge of X, and it is given by:

$$MI(Y, X) = H(Y) - H(Y|X)$$

where $H(Y|X)$ is the conditional entropy of Y given X and it is defined as :

$$\begin{aligned} H(Y|X) &= \sum_{i=1}^{I} p(X = x_i) H(Y|X = x_i) \\ &= -\sum_{i=1}^{I} p(X = x_i) \sum_{j=1}^{J} p(Y = y_j|X = x_i) log(p(Y = y_j|X = x_i)) \\ &= -\sum_{i=1}^{I} \pi_{i+} \sum_{j=1}^{J} \pi_{j|i} log(\pi_{j|i}) \end{aligned}$$

Mutual information is symmetric, that is, $MI(X, Y) = MI(Y, X)$. It is one of the most common measures used in decision trees to perform the selection of the attributes on which a test will take place. The attribute with the highest mutual information with the class, is selected for the split.In the machine learning literature it usually appears as *information gain* (Quinlan, 1992a; Cohen, 1995). The values of mutual information of two attributes X, Y fall in the interval $[0, min(H(X), H(Y))]$. A mutual information of zero means that the two attributes are independent. The maximum value of mutual information appears when one of the two attributes completely determines the other. That is when one of $H(X|Y), H(Y|X)$ is zero. We compute the mutual information of each attribute X_i with the class attribute C, that is $MI(C, X_i) = H(C) - H(C|X_i)$.

Another metric of association between nominal attributes, based on the mutual information, is the *uncertainty coefficient* (Agresti, 1990). The uncertainty coefficient is the mutual information normalized by the entropy of the dependent attribute, i.e.:

$$U(Y,X) = \frac{MI(Y,X)}{H(Y)} = \frac{H(Y) - H(Y|X)}{H(Y)} = 1 - \frac{H(Y|X)}{H(Y)}$$

This measure is also very popular in the machine learning community usually it is described as the *information gain ratio*, (Quinlan, 1992a; Gama & Brazdil, 1999), and it is used extensively in decision tree algorithms, in the same way as the mutual information. In the case of classification problems the dependent attribute is obviously the class attribute, so one would expect the normalization to be done with the class entropy. However instead of normalizing by the class entropy, they normalize with the entropy of the independent attribute, which in essence is the $U(X,C)$ concentration coefficient. So what actually the information gain ratio measures is the proportional reduction in the variation of the attribute X when C is known, and not the other way around as it would have been expected. In other words the information gain ratio selects as a splitting attribute the one for which the class attribute contains the most information! The advantage of the uncertainty coefficient over the mutual information is that since it is normalized its values fall in the interval $[0,1]$. The uncertainty coefficient is not symmetric, however there is the following extension which is symmetric with respect to the two attributes, and its values still fall in the $[0,1]$ interval :

$$U_{symmetrical}(Y,X) = \frac{2MI(Y,X)}{H(X) + H(Y)}$$

The uncertainty coefficient and the concentration coefficient presented in section 4.2.3 have the same semasiology. They both describe the proportional reduction in the variation of the dependent attribute with the knowledge of the independent attribute. Because of that we limit ourselves only to the concentration coefficient in order to describe the associations between the discrete attributes and the class. So for each discrete attribute X_i we compute the $\tau_{X_i C}$.

The above measures are not always adequate, because they are only able to capture interactions between only two attributes. Higher order relationships involving more than two attributes are not captured. For example if our concept is the parity problem the mutual information of every attribute with the class is always zero, a fact that is logical since by looking only one attribute we cannot have any information about the class, here all attributes have to be examined on the same time.

In an effort to describe the association between the continuous attributes and the class attribute we can use the *p-value* of the F-distribution as it was described previously in section 4.2.3. Note again, that what we are examining is whether the class attribute defines different groups in the continuous attributes, and not how the continuous attributes affects the class attribute. However if the values of the class attribute are associated with distinct groups of a continuous variable, then these variables can be used to discriminate between different classes. This is why we use again the same measure to describe the association between the continuous attributes and the class attribute.

Describing the associations between the continuous attributes and the class attribute requires some knowledge of discriminant and canonical correlation analysis. Canonical correlation analysis examines the correlations between two sets of variables (Anderson, 1984). Unlike traditional correlation analysis it looks for correlations in new transformed spaces where the correlations between the new variables are maximal. A new coordinate system is established for each set of variables in such a way that the new coordinates display clearly the system of correlation. This is done by finding the linear combinations of variables in each set that have the maximum correlation; these linear combinations are now the first coordinates in the new system. Then a second linear combination in each set is sought such that the correlation between these is the maximum of correlations between such linear combinations as are uncorrelated with the first linear combinations. The procedure is continued until the two new coordinate systems are completely specified. The new variables that are produced by the linear combinations are called canonical variates. Here we are interested in the maximal canonical correlation or the *first canonical correlation* which corresponds to the correlation of the first canonical variates on the new transformed system. Consider two multivariate random vectors, X, Y, and the two column vectors, w_x, w_y, then the first canonical correlation is given by:

$$\rho_{max} = argmax_{w_x,w_y} \frac{Cov(Xw_x, Yw_y)}{\sqrt{Var(Xw_x)Var(Yw_y)}} \tag{4.5}$$

When computing the first canonical correlation between the continuous attributes and the class attribute we consider as X, the random vector E consisting of the $X_1, X_2, ..., X_{con}$ continuous attributes, and as Y the class attribute represented in local binary encoding, i.e. Y is a cl dimensional vector. We are looking for the column vectors, w_x, w_y, that maximize the correlation between the attributes and the class. There are different ways to solve that problem, i.e. find the linear combinations and the maximal correlation coefficient. One of them makes use of discriminant analysis, which is closely related to canonical correlation analysis.

Let us consider again the multivariate random vector E consisting of the continuous attributes. Each observation e of the random vector belongs to class c_i; $1 \leq i \leq cl$. In discriminant analysis the goal is the computation of a set of linear transformations $Y_i = Ew_i, 1 \leq i \leq cl - 1$ of the initial set of variables that project the initial con dimensional space to a $cl - 1$ dimensional space. If we consider as W, the $con \times (cl - 1)$ matrix, that has as columns the w_i vectors, then we can write the linear transformation as a single matrix equation, $Y = EW$. The linear transformations are chosen in such a way that in the new space the projected samples are well separated. There are different criteria of class separability based on within-class, between-class and mixture scatter matrices, (Fukunaga, 1990). The within class scatter matrix gives a measure of the variation of the instances of each class around the class mean, the between-class scatter matrix a measure of the difference of the means of the different classes, and the mixture scatter matrix a measure of the variation of the whole dataset. The separability criterion that is of interest to us is the one suggested by Fisher (1936) for two class problems and extended by Rao (1948), to handle multiclass ones. The reason is that the maximization of the proposed criterion is directly associated with the canonical correlation coefficients, between the E variable and the class variable.

Let μ_{c_i}, μ be the *con* dimensional vectors giving the means of the c_i class and the total mean respectively,

$$\begin{aligned} \mu_{c_i} &= \frac{1}{n_{c_i}} \sum_{e \in c_i} e \\ \mu &= \frac{1}{n} \sum e = \frac{1}{n} \sum_{c_i} n_{c_i} \mu_{c_i} \end{aligned}$$

then the *total scatter matrix* of the dataset is given by

$$S_T = \sum_e (e - \mu)'(e - \mu) \tag{4.6}$$

the scatter matrix of the class c_i is given by

$$S_{c_i} = \sum_{e \in c_i} (e - \mu_{c_i})'(e - \mu_{c_i}) \tag{4.7}$$

the *within class scatter matrix* by

$$S_W = \sum_{c_i} S_{c_i} \tag{4.8}$$

and the *between class scatter matrix* by

$$S_B = \sum_{c_i} n_{c_i} (\mu_{c_i} - \mu)'(\mu_{c_i} - \mu) \tag{4.9}$$

Applying the W transformation on the E random vector results in the following within class and between class scatter matrices, on the new transformed space,

$$\begin{aligned} \widetilde{S_W} &= W' S_W W \\ \widetilde{S_B} &= W' S_B W \end{aligned}$$

Intuitively, to improve class separability in the new space, we can maximize the distances of the classes in the new space while minimizing the variation of the instances of each class, by appropriately selecting W. One way to do that is to maximize the ratio of the determinants of the two matrices, i.e.

$$argmax_W \frac{|\widetilde{S_B}|}{|\widetilde{S_W}|} = argmax_W \frac{|W' S_B W|}{|W' S_W W|}$$

The columns of the matrix W that maximize the ratio are the generalized eigenvectors that correspond to the largest eigenvalues of

$$S_B w_i = \lambda_i S_w w_i \tag{4.10}$$

with the eigenvectors given by

$$(S_B - \lambda_i S_W)w_i = 0 \tag{4.11}$$

If S_W is non-singular then the generalized eigenvalue problem given by equation 4.10 can be converted to a conventional eigenvalue problem of finding the eigenvalues λ_i of $S_W^{-1}S_B$. Since $S_W^{-1}S_B$ is positive semidefinite its eigenvalues are non-negative. The number of positive eigenvalues of $S_W^{-1}S_B$ is equal to its rank. And the rank of $S_W^{-1}S_B$ is bounded by

$$rank(S_W^{-1}S_B) \leq min(rank(S_W^{-1}), rank(S_B)) = min(cl-1, con)$$

S_B is the sum of cl matrices with the rank of each one being at most one. This follows from the fact that each one of them is the result of the outer product of two vectors, i.e. $(\mu_{c_i} - \mu)'(\mu_{c_i} - \mu)$. Furthermore since the cl class means satisfy $\mu = \frac{1}{n}\sum_{c_i}\mu_{c_i}$ at most cl-1 of these matrices are linear independent. As a result of these $rank(S_B) \leq min(cl-1, con)$.

It can be shown (Ripley, 1996), that the vector, w_x, that maximizes equation 4.5, is the eigenvector associated with the largest eigenvalue computed by equation 4.10. The resulting variable is called the first linear discriminant, and it is the first canonical variate. In the same way the vector that gives the second highest canonical correlation is the eigenvector associated with the second largest eigenvalue of equation 4.10, and gives the second linear discriminant or the second canonical variate, etc. Furthermore the canonical correlation coefficient of the i canonical variate with the class variable is given by $\rho_i = \sqrt{\frac{\lambda_i}{1+\lambda_i}}$. Consequently if λ_1 is the largest eigenvalue then the *first canonical correlation coefficient* is

$$\rho_{max} = \sqrt{\frac{\lambda_1}{1+\lambda_1}}$$

Because the eigenvalues are non-negative, we have $0 \leq \rho_i \leq 1$.

The w_i eigenvectors that correspond to the largest eigenvalues of equation 4.10 are the ones that have the highest discrimination power. λ_i measures the ratio of the between to within group variances on the i^{th} linear discriminant. The total variation of the linear discriminants is just the sum of the eigenvalues. The proportion of variation explained by the first linear discriminant is denoted as $frac1$ and it is computed as

$$frac1 = \frac{\lambda_1}{\sum_i \lambda_i}, \ (No\ 59)$$

this quantity can be also seen as the discriminating power of the first linear discriminant.

Note here the very close relation of the ANOVA based procedure described in section 4.2.3, used to measure the degree of association between a continuous and a discrete attribute, with discriminant analysis. The classes' scatter matrices (equation 4.7), the within class scatter matrix (equation 4.8) and the between class scatter matrix (equation 4.9) used by the discriminant analysis are the straightforward extensions of the groups' variation (equation 4.3), the within group variation (equation 4.4) and the between group variation (equation 4.2), from one dimensional variable to multivariate variable. While the ANOVA based procedure just measures the degree of association, discriminant analysis seeks for these linear transformations that maximize the associations.

The information required to specify the class of an instance is $H(C)$. The information that all the attributes provide about the class is $MI(C,E)$. It can be very easily seen that this quantity can be greater than the sum of the mutual information of the individual attributes with the class; a typical example of that is the parity problem. In a simplistic case all attributes are independent so the mutual information of the full vector of attributes with the class equals the sum of the mutual information of individual attributes with the class, i.e.

$$MI(C,E) = \sum_{i=1}^{attr} MI(C,X_i)$$

under that scenario each attribute contributes information independently of the rest. We could then consider that the mutual information of each attribute with the class attribute equals the average of the individual mutual information,

$$MI\widehat{(C},X) = \frac{\sum_{i=1}^{attr} MI(C,X_i)}{attr}, \ (No\ 65)$$

By taking the ratio

$$EN.attr = \frac{H(C)}{MI\widehat{(C},X)}, \ (No\ 66)$$

we can have a rough estimate of the number of attributes on average required to describe the class, this quantity is called the *equivalent number of attributes.*

Finally in an effort to estimate the amount of non-useful information of a dataset we use the *noise to signal ratio.* Which is given by :

$$NS.ratio = \frac{\widehat{H(X)} - MI\widehat{(C},X)}{MI\widehat{(C},X)}, \ (No\ 67)$$

$\widehat{H(X)}$ is the average information of the attributes. Then $\widehat{H(X)} - MI\widehat{(C},X)$ is the mean non-useful information of the attributes. So the ratio gives the percentage of non-useful information within the dataset.

4.2.5 Other dataset characteristics

Here we will give a description of these dataset characteristics that cannot be classified to any of the previous categories.

One factor that affects the difficulty of a classification problem is the dimensionality problem. It is known that increasing the number of features beyond a certain point is likely to be counterproductive (Duda & Hart, 1973). The number of examples required for learning grows exponentially with the number of features. But since we cannot always increase the number of examples, the learning space becomes sparsely populated and generalization becomes more difficult. As a rough measure of the dimensionality we use the ratio of the number of attributes to the number of instances, dim, (*No 4*).

The distribution of training examples among the different classes could also affect the performance of learners. One way to describe this distribution is the entropy of class, $H(C)$, (*No 63*). The lower its value is, the more skewed the distribution of instances among classes is. The higher the value the more balanced the distribution is.

Missing values also affect the performance of the learning algorithms, and different learning algorithms have different ways to handle them. For example c5.0 has a specific mechanism for them, while IBL uses a very naive strategy. So the result is that they exhibit different degrees of tolerance to presence of missing values. We used two simple measures to describe them, namely the total number of missing values, $mvals$, (*No 5*), and their proportion with respect to the product of the number of attributes with the number of instances, $\%mvals$, (*No 6*). Apart from that, we record the number of missing values for each attribute, in a later section (section 4.4) we will describe how these will be represented so that they can be handled by a propositional learner.

When one is using discriminant functions for classification the relation between the covariance matrices of the different classes determines whether linear or quadratic discriminants should be used. In the case where covariance matrices are equal then linear discriminants are used, when they are not equal then quadratic discriminant functions are used. A measure characterizing the equality or not of the covariance matrices could be predictive for the performance of linear discriminants. Box's M-statistic is a measure that can be used to test the equality of covariances. It is given by

$$M = \gamma \sum_{i=1}^{cl} (n_i - 1) log \frac{|S|}{|S_i|}$$

where

$$\gamma = 1 - \frac{2con^2 + 3con - 1}{6(con+1)(cl-1)} (\sum_{i=1}^{cl} \frac{1}{n_i - 1} - \frac{1}{n - cl})$$

S is the pooled covariance matrix

$$S = \frac{1}{n - cl} \sum_{i=1}^{cl} S_{c_i}$$

S_i is the i class covariance matrix

$$S_i = \frac{S_{c_i}}{n_i - 1}$$

and S_{c_i} is the i class scatter matrix.

The M-statistic can be reexpressed as the geometric means ratio of the pooled standard deviations to the classes' standard deviations via the *standard deviation ratio*, $SD.ratio$,

$$SD.ratio = exp(\frac{M}{con \sum_{i=1}^{cl}(n_i - 1)}), \ (No\ 69)$$

$SD.ratio$ is strictly greater than one if the covariances differ and is equal to one if and only if the M-statistic is zero, that is when all individual covariance matrices are equal to the pooled covariance matrix.

4.3 A discussion on characteristics

Here we will discuss some practical problems associated with the use of dataset characteristics.

In many of the characteristics used to describe a dataset, the range of their possible value varies depending on the specific attributes of the dataset on which they are measured. Examples of characteristics like that are the entropy of the attributes, the mutual information between an attribute and the entropy of the class attribute. This fact causes problems when it comes to the comparison of the values of a specific characteristic for different datasets, although we have the same characteristic the range of values is different. In order to be more concrete let us examine the case of the class entropy. We want to compare the class entropy of two different datasets, in order to determine how similar these two datasets are in that dimension. This is not straightforward since, as we have seen, the possible range of values of the class entropy depends on the number of different classes that each dataset has, and it is bounded by $[0, log(cl)]$. One possible solution, in order to alleviate the problem is to normalize these characteristics so that they will all fall in the same value interval. Note that here by normalizing we mean, in the case of the class entropy, the division with the maximum theoretical entropy, i.e. $log(cl)$. When extending the set of characteristics provided in STATLOG we took special care in selecting ones that had always the same range independently of the dataset attributes on which they are computed.

Another problem arises with the use of measures which can be computed only for discrete, or only for continuous attributes. There might be datasets for which it makes no sense to compute some characteristics. For example in a data set in which all the attributes are continuous, none of the characteristics for discrete attributes applies (e.g. entropy, concentration coefficient etc) and vice versa. The typical way to handle those cases so far was to consider these values as missing values but this alters the semantics of the problem, since they are not missing values but they deliver useful information about the nature of the dataset. They are rather a distinct value. To handle those cases we assign the label *non-appl* to the corresponding characteristics. This results in characteristics that have either continuous values or the value non-appl. In order to perform learning in the meta-level, we need inductive algorithms that can handle features of that nature.

A last problem with dataset characteristics is associated with specific pathological cases that may appear when computing the value of a characteristic, resulting in a non-numerical value for that characteristic. For example the concentration coefficient between two attributes can take the value of ∞ when the distinct values of an attribute are not properly defined. This situation can appear when one of the different values of one of the two attributes never appears in the dataset (although defined in the dataset schema), resulting in a π_{i+} of zero, in formula 4.1, giving the concentration coefficient. Again one solution to handle the specific problem, would be to encode these cases as missing values, but that would again alter the semantics of the problem, so we have decided to handle them as another distinct value of the characteristics.

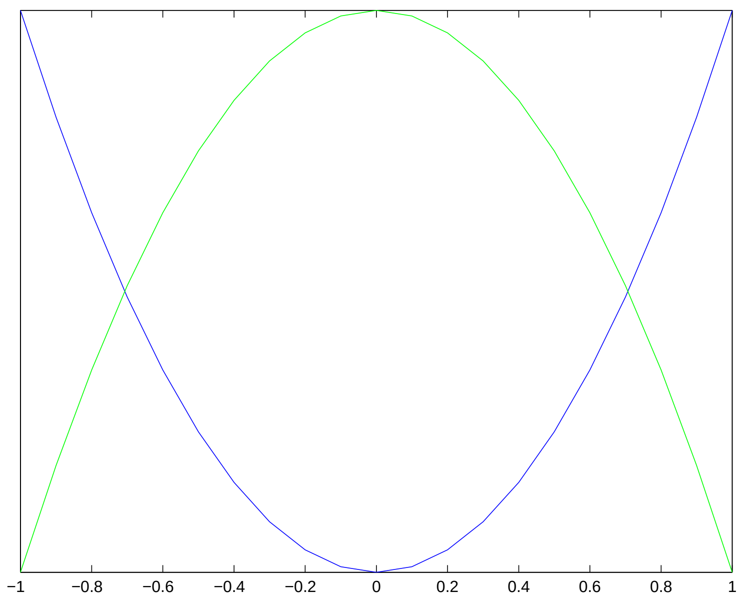

Fig. 4.1. Example of distributions of the correlation coefficients of two different datasets with the same mean, but completely different form

4.4 Staying Propositional

One of the weak points of the STATLOG approach, as well of the subsequent efforts in the spirit of STATLOG (i.e. (Brazdil et al., 1994; Lindner & Studer, 1999; Sohn, 1999; Soares & Brazdil, 2000)), was the management of the datasets characteristics which are computed for each attribute or for pairs of attributes. Let us take as example the case of the correlation coefficient. The specific characteristic is computed for all the pairs of continuous attributes, resulting in $O(\binom{con}{2})$ coefficients. The use of propositional learners on the meta-learning level requires that training instances should be described by exactly the same predictors. It is obvious that in the case of the correlation coefficients their number depends on the number of continuous attributes. So there is a need for a mapping of this varying length set of numbers to a constant representation which is independent of the number of attributes of a dataset. What was done in STATLOG and also in the subsequent similar approaches was to map every such set of characteristics to its mean value, so that finally every dataset is described by the same and fixed number of attributes. The limitations of this approach are obvious and we will exhibit them with a small example. In figure 4.1 we can see the distribution of the correlation coefficients for two different hypothetical datasets. In the distribution given by the solid line there is a strong correlation among the attributes. The values of the correlation coefficients are concentrated to the extremes of the $[-1, 1]$ interval. Consequently there is a high degree of pleonasm in the dataset. On the other side we have the dataset whose correlation coefficients distribution is given by the dotted line. Here the values of the correlation coefficients are concentrated around zero, so the dataset has a low level of pleonasm (at least when we are examining it, in terms of simple linear relationships). So what we finally have are two completely different datasets, which however when they are described by their mean, have exactly the same description, (since both distributions share the same mean of zero).

Todorovski et al. (2000) tried to attack the problem including also the minimum and maximum value for each characteristic that is computed on an attribute basis. In a more interesting approach Todorovski and Dzeroski (1999) use inductive logic programming to overcome the representational restrictions posed by the use of propositional learners on the meta-level. Inductive logic learners have richer representation power since they rely on relational representations of the instances. Like that, it is possible to maintain the full information contained in the initial characteristics without having to map them to a fixed representation. A similar approach is proposed in (Hilario & Kalousis, 2001), where they make use of a case based reasoning system. Case based systems also make use of relational representations of the instances thus overcoming the restrictions of propositional learners, however most of the available systems, including the one used in the (Hilario & Kalousis, 2001), do not induce first order rules.

Since in this work we restricted ourselves to the use of propositional learners we have to find a way to overcome their representational limitations by keeping as much information as possible from the initially available in the full set of the dataset characteristics. In order to achieve this, we use for each set of values associated with a specific characteristic, the histogram of the values in that set, instead of just its mean. More precisely for every characteristic of this type we compute its theoretical range of values, for example in the case of the concentration coefficient this is the interval $[0, 1]$. This interval is then divided in ten equal length bins. For each bin we compute the percentage of the pairs of attributes that have a concentration coefficient that belongs to that bin. Furthermore we also create one more bin which is associated with those pathological cases for which a characteristic is non-computable, as for example in the cases of concentration coefficient in which we take ∞ as a value.

The histograms where applied to all the dataset characteristics with which we have extended the STATLOG set of characteristics, plus the correlation coefficients that were used in STATLOG. More precisely we used the histogram representation for the following characteristics: the concentration coefficients between discrete attributes, $[\tau_1..\tau_{10}]$, (*No 12-21*), and between the class variable and the discrete attributes, $[\tau_{C_1}..\tau_{C_{10}}]$, (*No 23-32*); the correlation coefficient between continuous attributes, $[\rho_1..\rho_{10}]$, (*No 35-44*); the *p-value* of the F-distribution used to describe the associations between continuous and discrete attributes, $[p-val_1..p-val_{10}]$, (*No 70-79*), and also between the continuous attributes and the class variable, $[p-val_{C_1}..p-val_{C_{10}}]$, (*No 80-89*). The patterns of missing values among the attributes were also described using histograms, in the following paragraph we describe how this was done.

For a particular dataset, the percentage of the missing values alone gives almost no information on their structure. It is typical to have datasets which have exactly the same total percentage of missing values but a completely different distribution among the attributes; in that case the proportion of missing values has no discriminatory power. In (Kalousis & Hilario, 2000a) we have also shown that the way the missing values are distributed among the different attributes of a dataset critically affects the performance of the learning algorithms. To describe the way missing values are distributed among attributes, we compute the percentage of missing values for each attribute and then create a histogram of missing values, $[mvals_1..mvals_{10}]$, (*No 46-55*), i.e. the first bin contains the proportion of attributes that have between 0% and 10% missing values, the second between 10% and 20%, etc. In this way we are able to describe patterns of missing values in a finer detail.

For the rest of characteristics, whose number of values depend on the number of attributes, used in STATLOG, (i.e. attributes entropy, mutual information, skew, kurtosis) and the multiple correlation coefficient, we did not use the histogram representation but just their means, $\widehat{H(X)}, \widehat{MI(C,X)}, \hat{\gamma}, \hat{\beta}, \hat{R}$, (*No 64,65,61,62,68*), as it was done in STATLOG.

Using the histograms for the multiple correlation coefficient is straightforward since it takes values from a constant interval. For the entropy of the attributes and the mutual information the procedure is not straightforward, since their maximum theoretical value depends on the number of distinct values of the attributes on which the characteristics are computed, meaning that different attributes or different pairs of attributes will have different theoretical bounds. Consequently we can not describe their global distribution, since there is no common interval of values, on which to define the bins of the histogram. The problem could be solved with the normalization procedure described in section 4.3. Unfortunately for the kurtosis and skew values there is no way to normalize them in order to have common bounds.

Table 4.2. Computational Cost. n is the number of examples in the training set, and m the number of examples in the test set.

Algorithm	*Train Cost*	*Test Cost*	*10CV cost*
c50boost	$O(10 \times n \times con \times logn)$	$O(m)$	$O(100 \times n \times con \times logn)$
c50rules	$O(n^3)$	$O(m)$	$O(10 \times n^3)$
c50tree	$O(n \times con \times logn)$	$O(m)$	$O(10 \times n \times con \times logn)$
Ltree	$O(n \times con \times logn)$	$O(m)$	$O(10 \times n \times con \times logn)$
Lindiscr	$O(n)$	$O(m)$	$O(10 \times n)$
IBL	$O(1)$	$O(m \times n)$	$O(10 \times n^2)$
NB	$O(n)$	$O(m)$	$O(10 \times n)$
ripper	$O(n \times log^2 n)$	$O(m)$	$O(10 \times n \times log^2 n)$

4.5 Computational Complexity

One of the critical factors concerning the operational performance of the system is the computational requirements of the characteristics used to describe the datasets. If these require more computational time than the evaluation of the available learning algorithms, then clearly the system would have been useless.

Concerning the characteristics that are computed for the discrete attributes the main computational requirements stem from the construction of the contingency tables. This can be done in only one pass of the dataset, thus having a computational complexity of $O(n)$. For the characteristics that involve only one discrete attribute with I distinct values, their computation cost for all the nominal attributes is $O(nom \times I)$. The computational cost of characteristics that involve two nominal attributes with I, J distinct values, for all the pairs of nominal attributes is $O(\binom{nom}{2} \times I \times J) = O(nom^2 \times I \times J)$. So the total computational cost for the characteristics that are applied on the nominal attributes is $O(n + nom^2 \times I \times J + nom \times I)$, which is dominated by $O(n)$ when $nom \ll n$.

When it comes to the characteristics that are computed for the continuous attributes the main computational requirements stem from the construction of the covariance matrices, the computational complexity of which is $O(n)$. The computation of some of the characteristics is based on the eigenvalues of a $con \times con$ matrix, which in order to be computed require the inverse of a matrix with the same dimensions. Inversing a matrix of $con \times con$ dimensions has a cost of $O(con^3)$, the same a the computation of the eigenvalues of a matrix with the same dimensions. So the total computational cost of the characteristics used to describe continuous attributes is $O(n + con^3)$ which again is dominated by $O(n)$ when $con \ll n$.

To conclude the total computational cost of the dataset characteristics is $O(n)$ when $attr \ll n$, otherwise it is $O(n + nom^2 \times I \times J + nom \times I + con^3)$.

Let us now examine briefly the computational cost of estimating the accuracies through cross validation in order to select the best algorithm from the pool of eight learning algorithms that we are using here. In a typical application of cross validation one would use ten repetitions of train and test phases, using 90% of the available data as a training set, and the rest 10% as a test set. The computational cost of that heavily depends on the algorithm that one is currently evaluating and it will be : $10 \times (TrainCost_{algo}(90\%n) + TestCost_{algo}(10\%n))$. In order to simplify things we will consider that $attr \ll n$. Based on that assumption the computational cost of the test phase only depends on the number of instances in the test set.

Cohen (1995) computes the computational cost of ripper and they compare it with that of c4.5rules. Since *c5.0rules* is an evolution of c4.5rules, and there is no documentation of it, we will consider that it has a similar cost to that of c4.5rules. He computes the cost of ripper to be on the order of $O(nlog^2n)$ and that of c4.5rules to be $O(n^3)$, which is also confirmed by their experimental findings. In (Ruggieri, 2002) the computational complexity of c4.5tree is calculated to be on the order of $O(n \times (con \times logn + nom))$ which is roughly equal to $O(n \times con \times logn)$. For the same reasons with c5.0rules, we consider the cost of *c50tree* to be equivalent to that of c4.5tree. *Ltree* follows the same divide and conquer strategy with c50tree so they share the same computational cost. In the case of *c50boost* the computational cost is determined by the multiple executions of c50tree, ten in our experiments. So the cost of c50boost is simply $O(10 \times n \times con \times logn)$. In the case of *Lindiscr* the cost of training is $O(n + attr^3)$, since the algorithm consists of computing the covariance matrices and solving an eigenvalue problem, with respective cost of $O(n)$ and $O(attr^3)$. And under the initial assumption of $attr \ll n$, this gives a computational cost of $O(n)$. *Naive Bayes* requires just one pass of the data to build the model, and *IBL* does not even require a pass since it does not construct a model, however IBL is much more expensive in the testing phase, where its computational cost is $O(n \times m) = O(n^2)$.

The main cost of cross validation for all the algorithms except from IBL, comes from the application of the training phase for ten times. So roughly the cost of cross validation is ten times the cost of the training phase. In the case of IBL the main cost of cross validation comes from the application for ten times of the test phase. In table 4.2, we summarize the computational costs of the eight algorithms, for the train, test and the cross validation. It is obvious that the cost of using ten fold cross validation to find the best learning algorithm in a given dataset, is by far greater of that of computing the dataset characteristics.

The cost of cross-validation mentioned so far it is the worst case scenario. It is possible to develop special versions of classification algorithms that incorporate in the model construction phase the cross-validation cycle. This is straightforward for simple classification algorithms like Naive Bayes and Nearest Neighbors and results in cross validation costs similar to that of a single application of the algorithm. It is less trivial for more complex algorithms like decision trees and neural networks where the speedup of the cross validation version of the algorithm depends on the form of the concept, (Blockeel & Struyf, 2001), and it lies between the cost of a single application of the algorithm and the cost of normal cross validation.

Chapter 5

Instance Based Meta-Learning

The selection of the appropriate inducer which will be used at the meta-learning level will critically affect the performance of the system. As a starting point we use a simple instance based learning algorithm. Instance based learning algorithms induce no model from the training data, they only rely on distance measures of the instances that have to be classified from the data used for training. There were two main reasons for the selection of an instance based learner on the meta-level. First and more important is that we expect learning algorithms to exhibit similar performance on datasets with similar characteristics, so we can exploit past algorithm performance to predict performance on unseen datasets. Second, the fact that it is straightforward to adapt the distance measure of an instance based learner so that it can handle the *non-appl* values mentioned in section 4.3.

We define similarity between datasets in terms of geometrical proximity in the morphology space, whose dimensions are defined by the dataset characteristics, and observe the morphology space as a conventional Euclidean space, extended by *non-appl*. We modified the distance definition of the Nearest Neighbor algorithm to handle attributes whose domain is $\Re \cup non - appl$. The morphological characteristics are all normalized to the interval $[0, 1]$, prior to applying the algorithm.

Definition 5.0.1 *Let $M_1, \ldots, M_n$ denote the characteristics constituting a dataset morphology. The distance d between two datasets $\mathcal{D}, \mathcal{D}'$ with respective morphologies $< m_1, \ldots, m_n >$ and $< m'_1, \ldots, m'_n >$ is:*

$$d = \sum_{i=1}^{n} d_{ii}^2$$

where :

$$d_{ii} = m_i - m'_i \text{ if } m_i, m'_i \in \Re$$

$$
\begin{aligned}
&= 1 \textit{ if one of } m_i, m'_i \textit{ is non-appl} \\
&= 0 \textit{ if both of } m_i, m'_i \textit{ are non-appl}
\end{aligned}
$$

□

We experimented with two different sets of datasets characteristics. In the first one, which we will call *histo*, we excluded all the $ANOVA$ based measures, used to describe the associations between discrete and continuous attributes, that is we did not include characteristics after the 70th position of table 4.1. In the second, which we identify as $+histo$, we used all the characteristics of the given table. This was done in order to examine whether the extension of the dataset characteristics with the $ANOVA$ based measures, improves the discriminating power of the characterization.

5.1 Results with simple IBL

For the training and evaluation of the system we used the 1075 datasets described in section 3.4 and the evaluation method described in section 3.6. In effect we have two levels on which we evaluate the performance of a specific combination of dataset characterization and meta-learning algorithm. The first level is the performance in terms of accuracy for the pairwise meta-learning problems. The second level concerns the final suggestion of the system and we use both normal accuracy and the loose accuracy as it was defined in section 3.6.

5.1.1 Results on the *histo* set of characteristics

Using IBL on the *histo* set of dataset characteristics, gives quite good results both in terms of the accuracy of prediction on the pairwise meta-learning problems, as well as in the final suggestion of algorithm.

Results on the Pairwise Meta-Learning Problems In table 5.1, we give the achieved accuracies for all the pairwise meta-learning problems and the improvement over the default accuracy. For all of them the improvement is quite significant, moreover it is always significant in statistical terms. The complete class distributions for each of the pairwise meta-learning problems was already given in table 3.1.

In the most remarkable case, the pair of IBL and Naive Bayes, the default accuracy is 36.09% and the predictive accuracy is 85.02%, that is, an improvement of 48.93% over the default. It is obvious that the current set of characteristics is quite discriminating for the specific pair. This can be explained by the fact that the characteristics used give quite an emphasis on the description of interelations between the attributes, as well as between the attributes and the class, giving an implicit measure of the redundancy and the level of irrelevant information present in a dataset. In what concerns the level of irrelevant information in a dataset, we could state that Naive Bayes and simple instance based learners have a completely different region of expertise, with the former being able to cope far better on datasets with irrelevant attributes than the later. The reason is that Naive Bayes uses the class conditional probabilities of the attributes in order to classify an instance; with this mechanism the effect of the irrelevant attributes is reduced. When one attribute is irrelevant with respect to the class attribute we expect that its distribution will not change over the different classes, that is the class conditional probabilities of the irrelevant attributes do not change among the classes. When an instance is to be classified the irrelevant attributes will have exactly the same effect when computing the class posterior probabilities. On the other hand the instance based inducer used here gives equal importance to all attributes, independently of whether they are relevant or not, thus its performance is hampered by the presence of irrelevant attributes.

Overall the average improvement over the default accuracy is 27.24%, relatively high especially if one considers the difficult nature of the problem.

The achieved accuracy and the improvement over the default differ from pair to pair, implying the obvious, i.e. that the best set of discriminating characteristics is different from pair to pair.

Results on the Final Suggestion The results concerning the accuracy of the final suggestion, given in table 5.2, of the system are less satisfactory, especially if we consider the high level of accuracy that is achieved on the pairwise meta-learning problems. Here the accuracy falls down to 47.72%, nevertheless there is still a significant improvement, over the default accuracy that corresponds to the final suggestion, of 21.49%. The explanation to that is that the errors we get on the pairwise meta-learning problems do not occur on the same instances, in our case datasets. This is more easily demonstrated with an example. Let us suppose a simple scenario with three inducers on the base level. This would give rise to three different pairwise meta-learning problems. If the accuracy on all three hypothetical meta-learning problems was 90%, then in the worst case the accuracy of the final suggestion[1] would be $100\% - 3 \times 10\% = 70\%$. This scenario would happen when the produced meta-learning models make errors in disjoint sets of datasets and these errors affect the decision as to which algorithms are ranked in the top position. There could be errors in the individual pairwise meta-learning models that do not affect the correctness of the final suggestion. Thus the error of the final suggestion can be in the worst case the cumulation of the individual errors, of the pairwise meta-learning models.

Examining the performance of the system in terms of loose accuracy, i.e. the percentage of times that the system fails to find the correct set of algorithms that together occupy the top position but instead finds a subset of that set the

[1]Remember here that the final suggestion is the result of the combination of the individual suggestions of the pairwise meta-learning problems

Table 5.1. Accuracies of IBL on the *histo* feature set.

pair	Accuracy	Improvement
c50rules c50boost	82.42%	24.28%
c50tree c50boost	80.00%	22.23%
c50tree c50rules	78.14%	5.12%
Lindiscr c50boost	84.84%	20.37%
Lindiscr c50rules	85.12%	32.47%
Lindiscr c50tree	85.58%	30.88%
Ltree c50boost	78.60%	26.98%
Ltree c50rules	82.14%	23.44%
Ltree c50tree	78.79%	19.26%
Ltree Lindiscr	81.21%	20.09%
IBL c50boost	85.40%	20.74%
IBL c50rules	74.88%	25.12%
IBL c50tree	79.53%	27.53%
IBL Lindiscr	81.49%	39.16%
IBL Ltree	74.60%	17.95%
NB c50boost	85.02%	24.19%
NB c50rules	84.74%	33.30%
NB c50tree	84.47%	31.54%
NB Lindiscr	77.95%	36.84%
NB Ltree	84.37%	26.14%
NB IBL	85.02%	48.93%
ripper c50boost	85.02%	34.05%
ripper c50rules	77.86%	17.95%
ripper c50tree	84.09%	24.00%
ripper Lindiscr	78.70%	32.56%
ripper Ltree	75.81%	22.60%
ripper IBL	81.30%	35.16%
ripper NB	81.67%	39.91%
average	81.39%	27.24%

Table 5.2. Final suggestion results of IBL on *histo*

Strict Accuracy	Improvement	Loose Accuracy
47.72%	21.49%	69.67%

situation is better. The results are correct in 69.67% of the cases (table 5.2).

5.1.2 Results on the +*histo* set of characteristics

The addition of the ***p-values*** histograms derived from the $ANOVA$ based procedure was done in order to capture to some extend the interelations that exist between discrete and continuous attributes and between the continuous attributes and the class. Hopefully this would increase the discriminating power of the dataset characteristics, thus resulting in a better predictive performance. Nevertheless the results indicate that the incorporation of the ***p-values*** histograms do not improve the performance, but rather harm it.

Table 5.3. Accuracies of IBL on *+histo* feature set.

pair	Accuracy	Improvement
c50rules c50boost	80.56%	22.46%
c50tree c50boost	79.26%	21.53%
c50tree c50rules	77.96%	4.98%
Lindiscr c50boost	82.05%	17.63%
Lindiscr c50rules	81.68%	29.05%
Lindiscr c50tree	81.31%	26.70%
Ltree c50boost	76.38%	24.77%
Ltree c50rules	78.42%	19.80%
Ltree c50tree	75.73%	16.27%
Ltree Lindiscr	78.70%	17.58%
IBL c50boost	85.31%	20.75%
IBL c50rules	75.63%	25.93%
IBL c50tree	78.70%	26.70%
IBL Lindiscr	77.12%	34.87%
IBL Ltree	73.96%	17.35%
NB c50boost	81.49%	20.66%
NB c50rules	80.38%	28.96%
NB c50tree	81.87%	28.97%
NB Lindiscr	74.24%	33.18%
NB Ltree	82.24%	23.97%
NB IBL	78.52%	42.51%
ripper c50boost	82.14%	31.32%
ripper c50rules	76.38%	16.49%
ripper c50tree	82.14%	22.11%
ripper Lindiscr	75.73%	29.66%
ripper Ltree	73.49%	20.29%
ripper IBL	79.73%	33.66%
ripper NB	79.35%	37.63%
average	78.94%	24.85%

Results on the Pairwise Meta-Learning Problems The results on the *+histo* set of characteristics are given in table 5.3. The average accuracy over all the pairwise meta-learning problems is 78.94% a reduction of 2.45% compared to the average performance on the *histo* set of characteristics. In all the pairwise meta-learning problems but one, the accuracy on the *+histo* set of characteristics is lower than the corresponding on the *histo* set. In section 5.1.3 we will examine in more detail the statistical significance of the observed differences among the two sets of characteristics.

Results on the Final Suggestion Evaluating the quality of the final suggestion we can see that in terms of the accuracy of the final suggestion the *+histo* set of characteristics achieves almost the same accuracy with the *histo* with a slight decrease of 0.65%. If we move now to the loose accuracy there the reduction in performance is higher, 2.97%. The performance of the *+histo* is summarized in table 5.4.

Table 5.4. Final suggestion results of IBL on *+histo*

Strict Accuracy	Improvement	Loose Accuracy
47.07%	20.84%	66.70%

5.1.3 Comparing the discriminating power of $+histo, histo$

In this section we will proceed to the comparison of the two different ways of dataset characterization to determine whether the observed differences are statistically significant or not. In order to do that we will apply the McNemar test of statistical significance both in the pairwise metalearning problems and in the final suggestion. We set the α level of statistical significance at 0.05.

Results on the Pairwise Meta-Learning Problems In table 5.5 we give the results of the McNemar test on each of the individual pairwise meta-learning problems. The columns labeled *histo Correct*, *+histo Correct* give the number of cases where the *histo* set of characteristics gave a correct prediction while the *+histo* set gave a wrong prediction, and vice-versa. By examining the outcome of the McNemar test we can see that there is a statistical significance in favor of the *histo* set in 18 out of the 28 meta-learned models. In all the other cases there is no significant difference. There is no pairwise meta-learning problem where the *+histo* outperforms the *histo*, thus providing strong evidence for the superiority of the *histo* set over the *+histo* at least for the pairwise meta-learning problems.

Results on the Final Suggestion Examining the results of the statistical significance test on the final suggestion, table 5.6, we see that the two sets have the same performance when it comes to strict accuracy, i.e. their difference is not statistically significant. In the case of loose accuracy *histo* outperforms *+histo* and now the difference is statistical significant. The columns *histo Correct*, *+histo Correct* give the number of cases where the *histo* set of characteristics gave a correct prediction while the *+histo* set gave a wrong prediction, and vice-versa.

The overall picture from the preceding tests of statistical significance is that, at least when IBL is the meta-learner chosen, *histo* outperforms *+histo*. We see two possible explanations for this somehow counterintuitive phenomenon. One possible reason could be the fact that the extra features of *+histo* simply do not bring any useful information about the problem. As we have stated in the description of these extra features, they provide a crude way to describe the interelations between continuous and discrete attributes lacking another characteristic better adapted to that kind of need. The reason for that is their underlying assumption, i.e. the independent variable is the discrete variable and the dependent is the continuous, in that way they can only describe unidirectional dependencies. One other explanation could be the fact that the addition of new dataset characteristics increases the dimensionality of the problem thus

Table 5.5. Results of McNemar test comparing the accuracies of *histo* and *+histo*, on the pairwise meta-learning problems. = indicates no statistical difference, + indicates statistical difference.

pair	histo Correct	+histo Correct	$p - value$	
c50rules c50boost	70	50	0.083	=
c50tree c50boost	60	52	0.509	=
c50tree c50rules	49	47	0.919	=
Lindiscr c50boost	69	39	0.006	+
Lindiscr c50rules	75	38	0.001	+
Lindiscr c50tree	74	28	0.001	+
Ltree c50boost	81	57	0.051	=
Ltree c50rules	74	34	0.001	+
Ltree c50tree	76	43	0.004	+
Ltree Lindiscr	76	49	0.021	+
IBL c50boost	46	45	1.000	=
IBL c50rules	65	73	0.552	=
IBL c50tree	66	57	0.471	=
IBL Lindiscr	74	27	0.00	+
IBL Ltree	74	67	0.614	=
NB c50boost	69	31	0.001	+
NB c50rules	84	37	0.001	+
NB c50tree	74	46	0.014	+
NB Lindiscr	93	53	0.002	+
NB Ltree	60	36	0.019	+
NB IBL	100	30	0.00	+
ripper c50boost	73	43	0.008	+
ripper c50rules	58	42	0.134	=
ripper c50tree	60	39	0.045	+
ripper Lindiscr	77	45	0.006	+
ripper Ltree	73	48	0.030	+
ripper IBL	61	44	0.119	=
ripper NB	70	45	0.026	+

making learning more difficult. It still remains to be seen whether the same observation holds when more elaborate inducers are used on the meta-learning.

5.2 Meta-Feature Selection

An issue that has received little attention, if any, in the meta-learning field, is the explanation and the understanding of the factors that affect inducer performance. All previous efforts have aimed at maximizing the predictive capabilities of the meta-learner without shedding light on the factors (i.e. properties of the datasets) that affect the performance of the algorithms. Applying feature selection to the meta-level can cover this gap and at the same time improve meta-learning performance. Using feature selection we can have a better idea of the factors that affect the performance of the learners. This is especially true when the meta-learning algorithm used is an instance based learner, which gives no insight into the relevance of the attributes used for learning.

The first attempt at meta-feature selection appeared in the meta-learning

Table 5.6. Results of the McNemar test comparing the accuracies of *histo* and *+histo*, on the final suggestion, both in terms of strict and loose accuracy.

	histo Correct	+histo Correct	$p - value$	
strict Accuracy	76	69	0.6183	=
loose Accuracy	90	58	0.0119	+

framework of zooming-ranking (Todorovski et al., 2000). As it was mentioned previously the main limitation of this framework is the mandatory use of a single and global meta-learning model (an instance based model), independently of the pool of algorithms from which the selection is to be performed. However the factors that determine the relative performance of a group of algorithms, may be quite different from the factors that determine the relative performance of another group of algorithms. Moreover even within a specific group of learning algorithms the factors that determine the relative performance of a pair of algorithms from the group, can be quite different from the factors that affect the relative performance of the algorithms of an another pair. It is exactly this case that calls for use and combination of different meta-learning models. Here the diversity of the meta-models comes from the use of different sets of meta-features (i.e. dataset characteristics). This is where we can make full use of the flexibility of our meta-learning framework. By applying feature selection to the pairwise meta-learning problems we get different sets of meta-features which will give rise to different meta-learned models.

5.2.1 Feature Selection

Two are the main ways that feature selection is performed in machine learning, the *filter* and the *wrapper* approach (Liu & Motoda, 1998).

In the filter approach, only properties of the datasets are used in order to perform the selection of the features. These properties could be measures of association between features, measures of distance or dependence.

In the wrapper approach (Kohavi & John, 1997), the driving force is the accuracy of the learning algorithm that is going to be applied on the dataset. An extensive and systematic search is performed in the state space of all the possible feature subsets using heuristic search methods, like hill climbing, simulated annealing or best first. The search can begin either from the full set of features (*backward elimination*) or from the empty set (*forward selection*). The feature selection algorithm conducts the search using the estimated accuracy of the induction algorithm as the evaluation function. At the end, the feature set achieving the highest accuracy is selected.

Here we have chosen to use the wrapper approach to perform the feature selection on the meta-level. Although it requires a substantial amount of computational time, in the case of meta-learning this factor is not so important, since it will only be performed once. To perform feature selection we used MLC++ feature selection capabilities. The search strategy used is best first search. In table 5.7 we give in pseudo code the algorithm for performing the search on the state space of the feature sets. We start the search from the full set of features, thus using backward elimination. At each execution of the repeat loop the best feature set, which is not yet expanded, is chosen and all its children sets are computed by the application of the Expand function. A child set is created by the removal or addition of a feature from the parent set. The best set is defined as the one that achieves the maximum accuracy, as this is estimated by ten fold cross validation[2]. If it is better than the best set so far by more than ε it becomes the new best set. The value of ε was set to 0.001, and the value of k was set to 10. So if there is no improvement in accuracy after ten loops the search terminates, and the best set of features found so far is returned.

Table 5.7. Wrapper approach with best first search strategy

```
$NotExpanded$    $\leftarrow InitialSet$
$Expanded$       $\leftarrow \emptyset$
$BestSet$        $\leftarrow InitialSet$
repeat
      $CurSet$        $\leftarrow argmax_{set \in NotExpanded} Acc(set)$
      $NotExpanded$ $\leftarrow NotExpanded - CurSet$
      $Expanded$      $\leftarrow Expanded \cup CurSet$
      if $Acc(CurSet) - \varepsilon > Acc(BestSet)$ then
         $BestSet \leftarrow CurSet$
      $Children$      $\leftarrow Expand(CurSet)$
      foreach         $s \in Children, s \notin (Expanded \cup NotExpanded)$ do
         $NotExpanded \leftarrow NotExpanded \cup s$
until no change in $BestSet$ for $k$ loops
Return $BestSet$
```

5.2.2 Results on the Meta-Learning problems

Feature selection was applied to all of the 28 pairwise meta-learning problems resulting in a new set of features for each one of them. As a starting feature set we used the *histo*, since the wrapper feature selection process is time consuming[3], and the *histo* set exhibited superior performance compared to +*histo*. So the initial set of characteristics consisted of 69 features.

In table 5.8 we can see the accuracy results for all the meta-learning problems, along with the improvement over the simple IBL inducer, of IBL coupled with feature selection[4]. Feature selection gives a systematic improvement of accuracy, in all but two pairwise meta-learning problems (the pairs of IBL, c50tree and IBL, Lindiscr). Nevertheless the observed differences are statistically significant, under a McNemar test of significance, in only six out of the twenty eight pairwise problems. The average improvement, when compared to standard IBL,

[2]Note here that the ten fold cross validation, takes place only in the training set

[3]The whole evaluation procedure for the feature selection took twenty days, in a SUN Blade 100 workstation with 512 MB of main memory

[4]hereafter noted as fsIBL

over all the 28 meta-learning problems, is 1.21%, and the average improvement over the default accuracy is 28.45%. Apart from the improvement in performance, feature selection also reduces the number of dataset characteristics used for inducer selection from 69 to an average of 25.25 features per pairwise problem. In order to determine the final set of dataset characteristics we applied again fsIBL on each pairwise problem, but this time we used the complete data i.e. all 1075 instances of the meta-learning problems, since the goal was not the evaluation of fsIBL, but the establishment of the most discriminating set of features for every pair of inducer. These feature sets were not used for evaluating fsIBL.

Table 5.11 shows which characteristics were selected for some of the meta-learning problems. It is clear that the set of discriminating characteristics changes for different pairs of algorithms. Examining the selected characteristics for each pair of inducers we can only draw conclusions as to *which* factors impact their relative performance. The features selected by fsIBL for each meta-learning problem can be found in appendix, section A.2. However we cannot explain *how* these factors determine that relationship, i.e. what are the values of the characteristics for which it is better to use one inducer instead of another. If for example we examine the pair (IBL, Ltree) we see that all the elements of the correlation coefficients histogram have been selected, whereas the elements of the concentration coefficients histogram (both between attributes and attributes and class) have very low selectivity. Based on that we can argue that for the specific pair of inducers the correlation coefficients between pairs of continuous attributes play an important role in determining which inducer is better, whereas the concentration coefficients between the discrete attributes play only a marginal role. The presence of correlated features can affect the performance of IBL in different ways depending on whether the correlated features are relevant, in which case IBL can exhibit good performance, or irrelevant in which case the performance of IBL would deteriorate. On the other hand Ltree can exhibit a high degree of resilience to correlated attributes due to the incorporation of linear discriminant that allows the mapping of correlated features to a new uncorrelated space. An indication of the relevance of the features is given by the First Canonical Correlation, characteristic which is selected for this pair of algorithms, together with the correlation histogram they give an indication of the level of correlation among attributes and the relevance of the attributes for the classification problem. If we have a high level of correlation among attributes and a high First Canonical Correlation then we expect IBL to perform fairly. In the case that the First Canonical Correlation is low IBL should perform bad, but Ltree will be unaffected since it will reduce the effect of irrelevant redundancy through the use of linear discriminants. In another example, examining the pair (NB, c50boost) we observe that the correlation coefficients histogram has again full selectivity; but this time we get full selectivity also in the upper half of the concentration coefficients histogram, which delivers information on discrete attributes that exhibit a high degree of association, (a dataset characteristic known to affect the performance of Naive Bayes). To get a quantitative description of how the dataset characteristics determine inducers' superiority we plan to use a different meta-learning algorithm. Possible selections are inducers that produce a model of the classification, e.g decision trees or rule inducers.

Another way to examine the results is to explore the 'total' discriminating power of each dataset characteristic, that is how often it gets selected over *all* the meta-learning problems. Table 5.12 shows the relative frequency with which each feature is selected. The most often selected attribute is the noise to signal ratio, present in 25 of the 28 meta-learning problems. Although a rough approximation, since it is based on the mean attribute entropy and the mean mutual information between the attributes and the class, it is quite useful in determining the relative performance of the algorithms. The correlation histogram is another noteworthy characteristic: one of its bins shares the noise-signal ratio's extremely high selection rate, and eight of the others are above the 50% level. This seems to indicate the non negligible influence of correlated attributes on learning, due to varying degrees of sensitivity exhibited by the learners. The next characteristic in terms of 'total' discriminating power is the ratio of the number of attributes to the number of instances. It is known that increasing the number of features beyond a certain point is likely to be counterproductive (Duda & Hart, 1973). The number of classes is also selected very often as one of the features, providing an indication that inducers react differently to variations in the number of classes. Information theoretical measures such as mean mutual information and mean attribute entropy also appear to be discriminating features considering their high selection rate. The only characteristic that seems to be completely useless is the histogram of missing values, none of its elements is ever selected. This characteristic describes the distribution of percentages of missing values of the attributes. Overall the discriminating power of each characteristic is quite different, and with the notable exception of the missing values histogram, all of them are used at least in one pair of base-inducers.

5.2.3 Results on the final suggestions

In Table 5.9 we see the accuracy that the instance-based inducer, enhanced with feature selection, achieves on the final suggestion of base level inducers. For the strict accuracy, feature selection does not really affect the performance of the meta-learning, since IBL achieves a strict accuracy of 47.45%, (Table 5.2), and fsIBl a strict accuracy of 47.44%. The difference is obviously not significant with a *p-value* of 0.858, table 5.10. The loose accuracy without feature selection is 69.67% and increases to 74.98% with feature selection. The *p-value* of the test of significance for the loose accuracy is zero, which means the difference is significant at any level of significance.

5.3 Conclusions

In this chapter we tested the performance of an instance-based inducer, as the inducer to be used on the meta-learning level. The results were encouraging,

Table 5.8. Results with feature selection, fsIBL, on the *histo* feature set.

pair	Accuracy	Improvement over IBL	*p-value*	#features
c50rules c50boost	82.60%	0.18%	0.908 (=)	32
c50tree c50boost	81.12%	1.12%	0.224 (=)	30
c50tree c50rules	82.79%	4.65%	0.000 (+)	23
Lindiscr c50boost	85.67%	0.83%	0.297 (=)	19
Lindiscr c50rules	86.23%	1.11%	0.182 (=)	20
Lindiscr c50tree	86.98%	1.40%	0.091 (=)	18
Ltree c50boost	80.00%	1.40%	0.105 (=)	34
Ltree c50rules	82.70%	0.56%	0.561 (=)	31
Ltree c50tree	79.63%	0.84%	0.342 (=)	25
Ltree Lindiscr	82.33%	1.12%	0.256 (=)	25
IBL c50boost	85.86%	0.46%	0.602 (=)	22
IBL c50rules	79.07%	4.19%	0.000 (+)	16
IBL c50tree	79.07%	-0.46%	0.644 (=)	25
IBL Lindiscr	81.40%	-0.09%	1.000 (=)	24
IBL Ltree	76.84%	2.24%	0.028 (+)	27
NB c50boost	85.67%	0.65%	0.400 (=)	39
NB c50rules	86.70%	1.96%	0.038 (+)	23
NB c50tree	86.98%	2.51%	0.002 (+)	24
NB Lindiscr	79.16%	1.21%	0.227 (=)	23
NB Ltree	85.02%	0.65%	0.409 (=)	48
NB IBL	85.49%	0.47%	0.614 (=)	34
ripper c50boost	85.12%	0.10%	1.000 (=)	27
ripper c50rules	81.30%	3.44%	0.001 (+)	21
ripper c50tree	84.56%	0.47%	0.614 (=)	22
ripper Lindiscr	79.16%	0.46%	0.582 (=)	32
ripper Ltree	76.84%	1.03%	0.278 (=)	37
ripper IBL	81.95%	0.65%	0.482 (=)	22
ripper NB	82.42%	0.75%	0.388 (=)	31
Average	82.59%	1.21%		25.25
Improvement over default	28.54%			

Table 5.9. Final suggestion results with feature selection, fsIBL, on the *histo* feature set.

Strict Accuracy	Improvement	Loose Accuracy
47.44%	21.21%	74.98%

Table 5.10. Results of the McNemar test comparing the accuracies of IBL with feature selection and no feature selection, on the final suggestion, both in terms of strict and loose accuracy.

	Feature Selection Correct	No Feature Selection Correct	*p-value*	
strict Accuracy	61	64	0.858	=
loose Accuracy	84	27	0.000	+

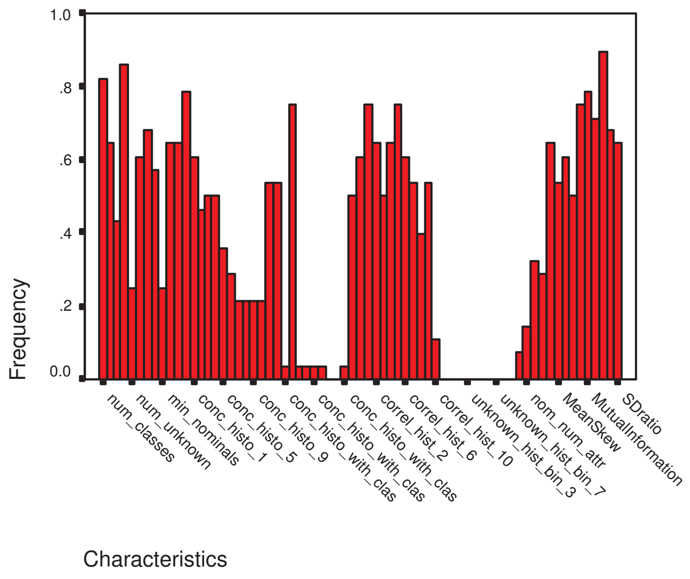

Fig. 5.1. Frequency with which every characteristic is selected among all the pairwise problems, fsIBL

the achieved performance significantly outperforms the default accuracy in all the cases. Furthermore we examined the performance of IBL with two different characterizations of the datasets. We found out that the addition of features that try to capture associations between discrete and continuous attributes, *+histo* dataset, instead of enhancing, harms the performance of the system, whether this is considered on the level of the pairwise metalearning problems or for the final suggestion. Nevertheless this should not be interpreted as a definite indication that the same will be true for more elaborate classifiers. Actually this hypothesis will be examined in the next chapter, where more elaborate inducers will be used in an effort to further improve the performance.

One of the observations made on the results of these experiments was the fact that the discriminatory power of the sets of characteristics seemed to be different for different pairs of algorithms. It was quite logical to assume that the factors that affect the relative performance of pairs of algorithms differ depending on the algorithms involved. In order to verify this hypothesis we

Table 5.11. Characteristics selected for three of the meta-learning problems, 1 indicates selection of the corresponding characteristic, 0 elimination.

Attribute	IBL NB	IBL Ltree	NB c50boost
# classes	1	1	1
# attributes	1	0	0
# instances	1	0	1
$\frac{\text{\# attributes}}{\text{\#instances}}$	1	1	0
# unknown values	1	0	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	1	1	1
# nominal attributes	1	1	1
max,min,mean,stdv of nominal attribute values	1101	1010	1010
1..10 concentration histogram	1101110000	0101000000	1010111111
non computable conc. histogram	0	0	1
1..10 concentration histogram with class	1001000000	0000000000	1101000000
non computable conc. histogram with class	0	0	0
# continuous attributes	1	0	0
1..10 correlation histogram	1111111001	1111111111	1111111111
non computable correlation histogram	0	0	1
1..10 missing values histogram	0000000000	0000000000	0000000000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0	0	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0	0	1
Binary Attributes	0	1	1
Frac1	1	0	0
First Canonical Correlation	1	1	1
Mean Skew	1	1	1
Mean Kurtosis	1	1	1
Class Entropy	0	0	1
Mean Attribute Entropy	1	1	0
Mean Mutual Information	1	1	1
Equivalent number of attributes	0	0	1
Noise to Signal Ratio	1	1	1
Mean Mult. Correl. Coef.	1	1	1
SDratio	0	1	0

Table 5.12. Frequency with which attributes are selected, by fsIBL

Attribute	frequency
# classes	82.21
# attributes	64.28
# instances	42.85
$\frac{\text{\# attributes}}{\text{\#instances}}$	85.71
# unknown values	25.00
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	60.71
# nominal attributes	67.85
max,min,mean,stdv of nominal attribute values	57.14, 25.00, 64.28, 64.28
1..5 concentration histogram	78.57, 60.71, 46.42, 50.00, 50.00
6..10 concentration histogram	35.71, 28.57, 21.42, 21.41, 21.42
non computable conc. histogram	21.42
1..5 concentration histogram with class	53.57, 53.57, 3.57, 75.00, 3.57
6..10 concentration histogram with class	3.57, 3.57, 3.57, 0, 0
non computable conc. histogram with class	3.57
# continuous attributes	50.00
1..5 correlation histogram	60.71, 75.00, 64.28, 50.00, 64.28
6..10 correlation histogram	75.00, 60.71, 53.57, 39.28, 53.57
non computable correlation histogram	10.71
1..10 missing values histogram	0, 0, 0, 0, 0, 0, 0, 0, 0, 0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	7.14
$\frac{\text{\# nominal}}{\text{\# attributes}}$	14.28
Binary Attributes	32.14
Frac1	28.57
First Canonical Correlation	64.28
Mean Skew	53.57
Mean Kurtosis	60.71
Class Entropy	50.00
Mean Attribute Entropy	75.00
Mean Mutual Information	78.57
Equivalent number of attributes	71.42
Noise to Signal Ratio	89.28
Mean Mult. Correl. Coef.	67.85
SDratio	64.28

performed feature selection on the pairwise meta-learning problems. Examining the meta-models created for each pair of inducers we saw that indeed the factors determining the relative performance of inducers vary from pair to pair. The use of feature selection not only improves the performance, but also provides a better understanding of what is relevant and what is not.

One of the main limitations of the IBL inducer used, is that it treats all features in the same way independently of the nature of the problem. For example when there is a dataset with 90% of continuous attributes and 10% of discrete attributes the same importance is given to all dataset characteristics independently of whether they describe properties of continuous or discrete attributes. One solution to that could be to use a weighted version of IBL, where the weights are not constant but depend on the proportion of continuous and discrete attributes in the dataset for which a suggestion is asked. This in effect would alter the euclidean distance putting each time the emphasis on the appropriate dimensions of the morphological space. Yet, another solution could be again the use of decision tree based algorithms. The tree based models that decision trees produce could hopefully capture this balance, for example by generating decision nodes that branch according to the percent of continuous attributes and then on the subsequent subtrees use different characteristics to perform the inducer selection.

Chapter 6

Comparing Metalearners

In this chapter we are going to explore the use of more elaborate learners on the meta-learning level, namely decision tree based learners. The main goal is to further improve the performance of the system. We also analyze the models produced, in order to characterize the dataset characteristics in terms of their predictive power, in a similar way that we analyzed the characteristics that were selected by fsIBL in the previous chapter.The rest of the chapter is organized as follows. First we examine the performance of the new meta-learners on the two distinct sets of datasets characteristics, i.e. $+histo$ and $histo$, in order to see with which one we get better performance. Then we compare the performances of the meta-learners, including that of fsIBL. Finally the inductive models constructed by two of the meta-learners are analyzed.

6.1 The Meta-Learners

We are going to examine the performance of four new learners on the meta-learning level. The c5.0 decision tree inducer (c50tree), a descendant of the c4.5 decision tree inducer that constructs decision trees where the splits on the decision nodes are orthogonal to the axes defined by attributes of the classification problem. The rule inducer of c5.0 (c50rules), also a descendant of the c4.5 rule inducer. c50rules starts with a decision tree constructed by c50tree and converts it to a set of rules. The Ltree inducer, which is also a decision tree inducer, however the splits here are not only orthogonal to the axes but they can also be oblique. Ltree constructs new attributes which are linear combinations of the initial attributes. The construction of the new attributes is done with the use of linear discriminant algorithm. We also applied c5.0 boost (c50boost), a boosting algorithm that combines multiple decision trees built by the repetitive application of c50tree on a dataset.

A problem that we have to deal with, when we apply the above inducers on the meta-learning level is the fact that they can handle only attributes which are either continuous or nominal. As mentioned already in section 4.3 the attributes that we use in order to describe the datasets can have, either a continuous value when the corresponding characteristic can be computed from a dataset, or the value *non-appl* when the computation of that feature does not make sense. Since the new learners cannot handle attributes of that type, we had to adopt a representation that would respect the semantics of the *non-appl* value. One possible solution could be their representation as missing values, however this would alter their semantics because the algorithms do not handle missing values as a distinct value, which is what we need. In order to achieve that we have chosen to recode the *non-appl* values to new numeric values which are always outside the definition domain of a given characteristic. This way a decision tree can create splits that on one side can have that special numeric value which corresponds to the non computable cases, and on the other side the normal values that correspond to the computable cases. This kind of recoding keeps the initial semantics of the *non-appl* values unchanged.

6.2 Comparing *+histo* and *histo*

We will examine the performance of the four new meta-learners on the two different sets of datasets characteristics. The results we get here are different from the ones that we obtained with IBL as the meta-learner. There we saw that the performance of the system was harmed by the incorporation of the $ANOVA$ based characteristics, both in terms of the accuracies on the pairwise meta-learning problems and on the final suggestion of algorithm(s). In what concerns the inducers examined here, we will see in the two forthcoming sections, that the incorporation of the $ANOVA$ based characteristics does not harm the performance. One reason for that could be the internal feature selection mechanism that all these inducers possess, allowing them to select the most discriminative set of characteristics on each case.

6.2.1 Results on the Pairwise Meta-learning Problems

In table 6.1 we see the mean accuracies that the four inducers achieve on the *+histo* and *histo* sets, over the 28 meta-learning problems. As it is obvious the performance on the two sets is quite similar, the differences are very small for every one of the four inducers[1]. This is more apparent when we examine the results of the McNemar test of significance to compare the performance of each meta-learner on *+histo* and *histo* for each pairwise meta-learning problem (Table 6.2)[2]. For example in the case of c50boost there is a statistically significant difference in favor of the *histo* set in only 3 out of the 28 pairwise problems, for the remaining 25 problems the differences are not statistically significant. A similar situation holds for the rest of the meta-learners where there

[1]The complete results of the four meta-learners in each one of the 28 pairwise problems can be found in the appendix, Table A.67 for the *histo* characterization and in Table A.68 for the *+histo*

[2]The complete results of the McNemar test for each pairwise problem are given in the appendix, Table A.69

Table 6.1. Mean accuracies and improvement over the mean default accuracies, that the four meta-learners achieve over the 28 meta-learning problems, for the *histo,+histo* sets of characteristics.

Meta-Learner	*histo*		*+histo*	
	Accuracy	Improvement	Accuracy	Improvement
c50boost	84.96%	30.82%	84.58%	30.44%
c50rules	82.22%	28.07%	81.75%	27.61%
c50tree	82.01%	27.86%	81.56%	27.42%
Ltree	81.18%	27.03%	81.05%	26.91%

Table 6.2. Summary of significant wins for the *histo*, *+histo* datasets for each of the four meta-learners over the 28 meta-learning problems.

Meta-Learner	*+histo* wins	Ties	*histo* wins
c50boost	0	25	3
c50rules	0	27	1
c50tree	0	26	2
Ltree	1	27	0

is no statistically significant difference for almost all the pairwise meta-learning problems.

6.2.2 Results on the Final Suggestion

Here also the performance of the four learners does not differ significantly among the two sets, both in terms of the strict and loose accuracy, (Tables 6.3,6.4). When performing the McNemar test to compare the performances on the *histo* and *+histo* we see that for all the learners there is no significant difference between the two sets with respect to the two ways of evaluating the final suggestion (Table 6.5).

Since the results show that there is no significant difference between the two ways of characterizing a dataset, in what follows and for the performance comparison of the meta-learners we will restrict ourselves only to the *histo* set of characteristics.

Table 6.3. Results on the final suggestion for all the four meta-learners on *histo*.

Meta-Learner	Strict Accuracy	Improvement	Loose Accuracy
c50boost	51.16%	24.93%	76.74%
c50rules	45.58%	19.35%	75.07%
c50tree	45.58%	19.35%	76.93%
Ltree	43.07%	16.84%	73.02%

Table 6.4. Results on the final suggestion for all the four meta-learners on *+histo*.

Meta-Learner	Strict Accuracy	Improvement	Loose Accuracy
c50boost	50.98%	24.75%	76.47%
c50rules	46.98%	20.75%	76.19%
c50tree	45.77%	19.54%	76.84%
Ltree	42.60%	16.37%	73.58%

Table 6.5. Results of McNemar's test comparing the accuracies of *histo*, *+histo*, on the final suggestion, for each of the four meta-learners.

		histo	*+histo*		
		Correct	Correct	*p-value*	
c50boost	strict Accuracy	61	59	0.927	=
	loose Accuracy	54	51	0.846	=
c50rules	strict Accuracy	73	88	0.269	=
	loose Accuracy	61	73	0.342	=
c50tree	strict Accuracy	70	72	0.933	=
	loose Accuracy	61	60	1.00	=
Ltree	strict Accuracy	91	86	0.763	=
	loose Accuracy	73	79	0.686	=

6.3 Looking for the best Meta-Learner

In this section we compare the performance of the inducers used in the meta-learning level in order to establish whether the differences observed are statistically significant, and identify the top performing one(s). The comparison will take place among five inducers, (c50boost, c50rules, c50tree, fsIBL, Ltree), on the *histo* set of characteristics. They will be compared on three levels, i.e. their performance on the individual pairwise problems, strict and loose accuracy of the final suggestion. Since we have multiply comparisons we must take into account the number of different comparisons and appropriately adjust the significance level according to the Bonferroni adjustment. The number of pairwise comparisons among five different algorithms is ten and the Bonferroni adjustment gives a significance level of 0.005 for every pair, so that results will be significant on the 0.05 level for all the comparisons.

6.3.1 Results on the Pairwise Meta-Learning Problems

Examining the mean accuracies that the five inducers achieve on the meta-learning level, table 6.6, we can see clearly that c50boost has the highest average accuracy, 84.96%, over all the other inducers. The remaining four exhibit similar performance, with an average accuracy ranging from 81.2% to 82.6%, with Ltree being the worst of the four, and fsIBL the best. The differences between the four are quite small though.

When we examine closely the performances on the individual problems, (table 6.7), in terms of the statistical significance the advantage of c50boost over the rest is further confirmed. Compared to any of the other inducers it is never significantly overtaken and more over it significantly outperforms them for the majority of the pairwise problems. The number of its significant wins ranges from 10 to 23 depending on which inducer it is compared with. One interesting observation here is that c50boost and fsIBL have a similar performance, i.e. not significantly different, on 18 out of the 28 pairwise meta-learning problems. The comparison between the remaining four inducers shows that their performance is quite similar, and there is no significant difference for the big majority of problems. One explanation for the relatively poor performance of Ltree, in terms of the average accuracy, is the way that we have chosen to encode the *non-appl* values. Since the new values are also numerical values, Ltree uses them like any other numerical value when it constructs new attributes from linear combinations of the existing ones. A fact that does not match with the semasiology of the non-appl values. These values should be handled as different-distinct values and the use of simple orthogonal splits on the space defined by the initial features satisfies this requirement. On the other hand, the clear advantage of c50boost, over the rest of the inducers, can be explained by the fact that it is a boosting algorithm, while the others are single model algorithms.

6.3.2 Results on the final suggestion

Continuing the evaluation of the performances of inducers with respect to their final suggestion we observe a similar situation in what concerns the comparison along the strict accuracy, table 6.8. c50boost achieves by far the highest strict accuracy, and Ltree is again the worst. fsIBL, c50tree and c50rules have a similar performance, with fsIBL having a small advantage. When we examine the statistical significance of the results, table 6.9, the superiority of c50boost is confirmed. Its performance with respect to the strict accuracy is significantly better than the performance of any other classifier. Between c50tree, c50rules, fsIBL and Ltree the differences are not significant, with the exception of the (fsIBL, Ltree) pair where fsIBL is significantly better than Ltree.

In terms of loose accuracy the results are slightly different, table 6.8. The worse performing inducer is still Ltree, but now c50tree very close to c50boost, is on the top position. C50rules and fsIBL have a similar performance. In terms of the statistical significance of the differences, there is no clear winner, i.e. an inducer that is significantly better than all the others. Only two pairwise comparisons revealed significant differences; the full results are given in table 6.10.

6.4 Discriminating Power of Characteristics

To perform the splits on the nodes of the tree, decision tree based algorithms possess internal feature selection mechanisms, for choosing the features that convey the highest amount of information about the class variable. All the algorithms used in this chapter use the same criterion for selecting an attribute as a split node, that of information gain.

Table 6.6. Mean accuracies and improvement over the mean default accuracies, that the five meta-learners achieve over the 28 meta-learning problems, for the *histo* sets of characteristics.

Meta-learner	Accuracy	Improvement
c50boost	84.96%	30.82%
c50rules	82.22%	28.07%
c50tree	82.01%	27.86%
fsIBL	82.59%	28.45%
Ltree	81.18%	27.03%

Table 6.7. Distribution of significant wins, based on McNemar's test, over the 28 pairwise meta-learning problems, on the *histo* set of characteristics. In a triplet AA/BB/CC, AA is the number of significant wins of the row inducer, CC is the number of significant wins of the column inducer and BB is the number of ties.

	c50rules	c50tree	fs-IBL	Ltree
c50boost	15/13/0	14/14/0	10/18/0	23/5/0
c50rules		0/28/0	0/26/2	1/27/0
c50tree			0/26/2	0/28/0
fs-IBL				3/25/0

Table 6.8. Results on the final suggestion of the five meta-learners on *histo*.

Meta-Learner	Strict Accuracy	Improvement	Loose Accuracy
c50boost	51.16%	24.93%	76.74%
c50rules	45.58%	19.35%	75.71%
c50tree	45.58%	19.35%	76.93%
fs-IBL	47.44%	21.21%	74.98%
Ltree	43.07%	16.84%	73.02%

Table 6.9. Results of the McNemar test comparing the accuracies of the five meta-learners, in terms of the strict accuracy on *histo*. + indicates a significant win for the row inducer, - a significant win for the column inducer, and = a tie.

	c50rules	c50tree	fs-IBL	Ltree
c50boost	+(0.000)	+(0.000)	+(0.004)	+(0.000)
c50rules		=(0.912)	=(0.210)	=(0.070)
c50tree			=(0.198)	=(0.078)
fs-IBL				+(0.000)

Table 6.10. Results of the McNemar test comparing the performances of the five meta-learners, in terms of the loose accuracy on *histo*. + indicates a significant win for the row inducer, - a significant win for the column inducer, and = a tie.

	c50rules	c50tree	fs-IBL	Ltree
c50boost	=(0.171)	=(0.934)	=(0.149)	+(0.005)
c50rules		=(0.030)	=(1.000)	=(0.122)
c50tree			=(0.159)	+(0.004)
fs-IBL				=(0.193)

We will analyze the models produced by c50tree and c50boost in order to characterize the discriminatory power of the different characteristics, in the same way we did with the analysis of the feature sets selected by fsIBL. The models analyzed are the results of the application of the two algorithms to the full set of training examples, that is to training sets of 1075 instances. For each model[3] we compute the number of times that a characteristic is selected as a split node. At the end we compute for each characteristic the percent of models in which it is selected at least once as a split node, (for c50tree see table 6.14, for c50boost see table 6.16). Since in decision tree models, a feature can be selected more than once, we also compute the frequency with which a characteristic is selected to become a decision node among all the decision trees. This avenue of analysis will provide a better insight in the case of c50boost, since there almost every characteristic is used at least once in every pairwise problem, resulting in a selectivity of 100%, going to a finer detail will provide a more clear picture of the discriminatory power of the characteristics, (Tables 6.15, 6.17)[4]. We make the assumption that the more often a characteristic is selected the higher its discriminatory power is.

It should be noted that the method which we used to determine the discriminatory power of the dataset characteristics based on the decision tree models is approximate and rather rough. More precisely when measuring the percentage of times that a feature appears in a decision tree we do not take into account the level of the decision tree at which the specific feature appears. As it is known features that are selected at nodes near the root of the tree have higher discriminatory power than features that are selected near the leaves. We would have a more precise picture of the discriminatory power of the features if we were weighting their appearance by the level at which they appear. A further source of imprecision, in the case of the c50boost models, comes from the aggregation of the percentages of appearances of the features among the different decision trees of a boosting model. In boosting the different decision trees are weighted by their error on the training. Consequently the importance of features that appear in different trees of a boosting model is different and depends on the weight of the corresponding tree. A more precise way to measure the discriminatory power of the features would be to weight them taking into account the weight of the decision tree in which they appear.

Examining the produced models we can see that at least for c50tree the set of discriminating characteristics differs between pairs, as it was also the case with fsIBL. When we examine the characteristics selected by c50boost, we see that it consistently uses almost all the available features. Nevertheless if we look more closely to the selection frequencies of the different characteristics, we see that these can change significantly between different pairwise problems, which is an indication that the discriminatory power of the characteristics varies among different pairs of inducers. To support these we will examine more closely the selection frequencies of the characteristics for the pairs (NB, IBL) and (NB,

[3]Remember here that a model is associated with a specific pairwise problem

[4]The frequency of selection of the characteristics for each pairwise problem are given in the appendix, section A.3 for c50boost, and section A.4 for c50tree.

Ltree). In what concerns the models produced by c50tree we will only give the bar graph that presents the selection rates of the characteristics for the two problems, figure 6.1. From the graph it is obvious that not only the selection rate of the characteristics is different, but also there are characteristics which are never selected in one pair and are selected in the other. In what concerns c50boost the situation is similar, although here we will not find often characteristics which are selected in one problem and not in the other, however the frequency of selection differs among the problems, figure 6.2. In the case of the NB-IBL pair we can see a relatively high selection rate for the higher bins of the correlation coefficient, (bins five to ten). These bins describe the percentage of attributes that exhibit medium to strong correlation, both IBL and Naive Bayes are sensitive to correlated attributes. If we examine the concentration coefficient histogram, between the attributes and the class, the selection rate is higher for the lower bins. These bins give the percentages of attributes that exhibit weak association with the class attribute. The higher their number the more difficult the classification problem is. If we now turn to the IBL-Ltree pair we observe a different pattern. In what concerns the correlation coefficient histogram all of its bins exhibit now a relatively high selection rate. For the concentration coefficient histogram, between the attributes and the class, the pattern is also slightly different, with the higher four bins never selected and the lower ones exhibiting lower selectivity with respect to the one that they exhibited on the NB-IBL pair.

c50tree and c50boost use a higher number of features in their models, compared to fsIBL. For c50tree the average number of selected features overall the pairwise problems is 38.92, while for c50boost it goes up to 61.53 here almost every characteristic is selected at least once, (Table 6.13). c50tree and c50boost select attributes that describe the patterns of missing values with a quite high frequency, (e.g number of unknown values, histogram of missing values). This is different from what we saw with fsIBL, (Table 5.12), where the selectivity of those characteristics was quite low, and in the case of the histogram of missing values it never used even one of its elements. Nevertheless the performance of fsIBL is very similar to that of c50tree, if not better. It would be interesting to examine whether c50tree and c50boost could achieve the same performance if we remove the missing values histogram for the set of characteristics. Simple features that describe the number of classes or the number of attributes have also a high selectivity rate for both c50tree and c50boost. The same also holds for STATLOG characteristics like class entropy, first canonical correlation, attribute entropy etc. They are selected frequently among the different pairwise problems and they have a high selection rate as decision nodes.

Concerning the remaining characteristics described via histograms, the correlation coefficient histogram also exhibits high selectivity rates for all of its elements. The same holds for the first five bins of the concentration coefficient histograms (both between attributes, and between attributes and the class). The upper five bins of these histograms, have quite low selectivity rate which in some cases approaches zero. In general the observations for these three types of histograms agree with the ones made with fsIBL, there also the correlation

Table 6.11. Frequency with which characteristics are selected for the NB-IBL pair by c50boost.

Attribute	frequency
# classes	3.37%
# attributes	1.80%
# instances	4.16%
$\frac{\text{\# attributes}}{\text{\#instances}}$	4.27%
# unknown values	5.51%
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	2.02%
# nominal attributes	1.01%
max,min,mean,stdv of nominal attribute values	1.12%, 0.034, 1.46%, 0.67%
1..5 concentration histogram	1.35%, 0.90%, 1.35%, 1.12%, 1.24%
6..10 concentration histogram	0.34%, 0.67%, 0.00%, 0.00%, 0.22%
non computable conc. histogram	1.01%
1..5 concentration histogram with class	1.91%, 1.24%, 1.12%, 2.02%, 0.79%
6..10 concentration histogram with class	0.56%, 0.00%, 0.11%, 0.22%, 0.11%
non computable conc. histogram with class	0.00%
# continuous attributes	1.01%
1..5 correlation histogram	2.02%, 0.90%, 1.80%, 0.90%, 1.57%
6..10 correlation histogram	1.24%, 1.12%, 1.12%, 1.46%, 0.79%
non computable correlation histogram	1.01%
1..5 missing values histogram	0.00%, 2.47%, 2.70%, 3.60%, 3.37%
6..10 missing values histogram	1.80%, 1.12%, 0.00%, 0.79%, 0.22%
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.11%
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.00%
Binary Attributes	1.01%
Frac1	1.80%
First Canonical Correlation	2.14%
Mean Skew	1.80%
Mean Kurtosis	1.69%
Class Entropy	4.16%
Mean Attribute Entropy	2.25%
Mean Mutual Information	4.05%
Equivalent number of attributes	2.14%
Noise to Signal Ratio	2.02%
Mean Mult. Correl. Coef.	2.02%
SDratio	1.69%

Table 6.12. Frequency with which characteristics are selected for the IBL-Ltree pair by c50tree.

Attribute	frequency
# classes	2.12%
# attributes	2.12%
# instances	3.39%
$\frac{\text{\# attributes}}{\text{\#instances}}$	2.75%
# unknown values	5.08%
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	1.90%
# nominal attributes	0.85%
max,min,mean,stdv of nominal attribute values	1.16%, 0.42%, 1.16%, 1.69%
1..5 concentration histogram	0.95%, 0.42%, 0.63%, 1.06%, 0.74%
6..10 concentration histogram	1.16%, 0.32%, 0.00%, 0.11%, 0.63%
non computable conc. histogram	0.74%
1..5 concentration histogram with class	0.85%, 1.06%, 0.42%, 1.69%, 0.74%
6..10 concentration histogram with class	0.42%, 0.00%, 0.00%, 0.00%, 0.00%
non computable conc. histogram with class	0.00%
# continuous attributes	1.27%
1..5 correlation histogram	1.80%, 2.12%, 2.43%, 1.27%, 1.16%
6..10 correlation histogram	1.69%, 2.01%, 1.69%, 2.43%, 0.85%
non computable correlation histogram	1.06%
1..5 missing values histogram	0.00%, 3.17%, 4.55%, 2.43%, 2.01%
6..10 missing values histogram	1.80%, 0.63%, 0.11%, 0.53%, 0.11%
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.32%
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.32%
Binary Attributes	0.63%
Frac1	2.01%
First Canonical Correlation	2.75%
Mean Skew	1.27%
Mean Kurtosis	2.43%
Class Entropy	5.93%
Mean Attribute Entropy	3.49%
Mean Mutual Information	2.65%
Equivalent number of attributes	1.48%
Noise to Signal Ratio	2.54%
Mean Mult. Correl. Coef.	2.54%
SDratio	1.90%

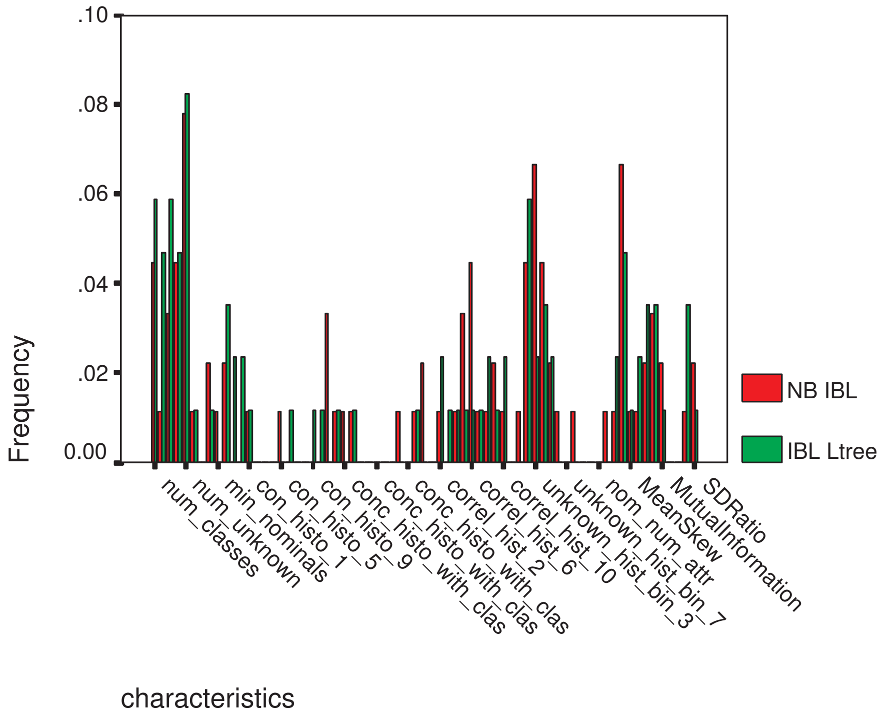

Fig. 6.1. Frequency with which characteristics are selected as decision nodes for the pairs (NB, IBL) and (IBL, Ltree) by c50tree

histogram had a high selectivity rate for all of its bins. The concentration coefficient histogram between the class and the attributes had high selectivity rate for the first few bins and quite low, even zero, for the remaining bins. In what concerns though the concentration coefficient histogram between attributes there is a slight difference from the pattern observed in c50tree and c50boost. The five first bins have high selectivity rate by fsIBL too, but in the case of the upper five, the selectivity is average and not zero, as it is the case with c50boost and c50tree.

To summarize, simple and STATLOG characteristics tend to have high selectivity rates among the three different meta-learners examined. fsIBL tends to ignore the characteristics that describe the patterns of missing values, (it never uses the histogram of missing values), while for c50tree and c50boost they are among the most often selected. However the performance of fsIBL is very similar to that of c50tree. The elements of the correlation histogram have high selectivity rates by all the three inducers. In the case of the concentration coefficient histograms, high selectivity rates are observed for their five first bins, while for the upper five the selectivity approaches zero. The only exception to

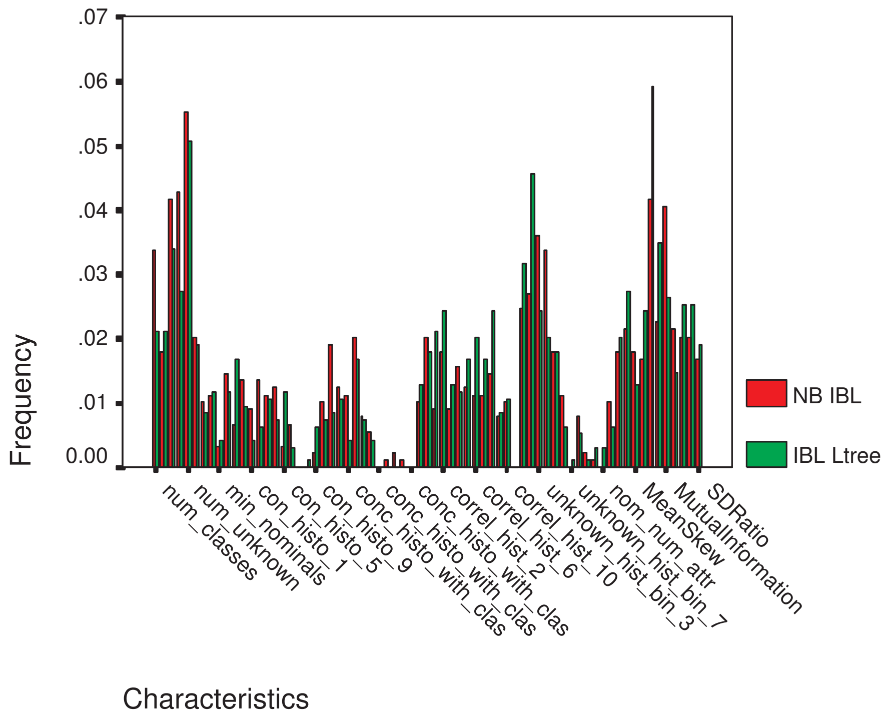

Fig. 6.2. Frequency with which characteristics are selected as decision nodes for the pairs (NB, IBL) and (IBL, Ltree) by c50boost

that is fsIBL, where for the upper five bins of the concentration coefficient between attributes the selectivity is average. We have to note here that the study and the models were produced from a set of datasets that included datasets whose characteristics were manipulated in order to increase the number of our training examples. In a following chapter we will further control the validity of our observations on the discriminatory power of the dataset characteristics, in models which are created only from real world datasets.

6.5 Summary and Conclusions

In this chapter we examined the performance of four different meta-learners on the two different ways of characterizing a dataset, *histo* and +*histo*. There was no performance variation among the two different versions. The reader may remember that the performance of the instance based inducer was deteriorating on the +*histo* set of characteristics. The performance of the four meta-learners presented here, remained unaffected due to the feature selection mechanism

Table 6.13. Number of features used for each pairwise problem.

pair	fsIBL	c50tree	c50boost
c50rules c50boost	32	40	61
c50tree c50boost	30	39	64
c50tree c50rules	23	36	61
Lindiscr c50boost	19	36	62
Lindiscr c50rules	20	39	62
Lindiscr c50tree	18	36	60
Ltree c50boost	34	44	60
Ltree c50rules	31	40	61
Ltree c50tree	25	42	62
Ltree Lindiscr	25	41	63
IBL c50boost	22	33	58
IBL c50rules	16	42	62
IBL c50tree	25	40	63
IBL Lindiscr	24	41	61
IBL Ltree	27	39	62
NB c50boost	39	35	64
NB c50rules	23	35	60
NB c50tree	24	41	59
NB Lindiscr	23	41	61
NB Ltree	48	36	62
NB IBL	34	43	62
ripper c50boost	27	34	61
ripper c50rules	21	40	63
ripper c50tree	22	39	61
ripper Lindiscr	32	43	59
ripper Ltree	37	43	65
ripper IBL	22	30	62
ripper NB	31	42	62
Average	25.25	38.92	61.53

that all of them posses. Moving to the comparison of the performances of the meta-learners, there were clear evidence for the superiority of c50boost both in terms of the quality of predictions on the individual pairwise problems and the final suggestion. In what concerns the discriminatory power of the dataset characteristics as this is determined by their selection rate this was similar among the three meta-learners. There was however one notable exception. The fact that fsIBL was ignoring the characteristics which are describing the pattern of missing values, while for c50tree and c50boost are among the ones most often selected, and still achieving similar performance to that of c50tree. A section of the following chapter will be devoted to the analysis of the models produced by c50boost on real world datasets, in order to cross check them with the ones produced here.

Table 6.14. Frequency with which characteristics are selected among the different pairwise problems, c50tree.

Attribute	frequency
# classes	100%
# attributes	96.42%
# instances	96.42%
$\frac{\text{\# attributes}}{\text{\#instances}}$	78.57%
# unknown values	100%
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	71.42%
# nominal attributes	46.42%
max,min,mean,stdv of nominal attribute values	85.71% 67.85% 53.57% 35.71%
1..5 concentration histogram	71.42% 60.71% 39.28% 32.14% 39.28%
6..10 concentration histogram	21.42% 25.00% 10.71% 0.00% 50.00%
non computable conc. histogram	64.28%
1..5 concentration histogram with class	64.28% 60.71% 32.14% 75.00% 32.14%
6..10 concentration histogram with class	10.71% 0.00% 0.00% 0.00% 32.14%
non computable conc. histogram with class	0.00%
# continuous attributes	57.14%
1..5 correlation histogram	67.85% 71.42% 78.57% 53.57% 75.00%
6..10 correlation histogram	71.42% 82.14% 71.42% 75.00% 85.71%
non computable correlation histogram	64.28%
1..5 missing values histogram	0.00% 81.14% 92.85% 92.85% 96.42%
6..10 missing values histogram	85.71% 75.00% 14.28% 67.85% 21.42%
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.00%
$\frac{\text{\# nominal}}{\text{\# attributes}}$	14.28%
Binary Attributes	53.57%
Frac1	67.85%
First Canonical Correlation	89.28%
Mean Skew	64.28%
Mean Kurtosis	50.00%
Class Entropy	96.42%
Mean Attribute Entropy	85.71%
Mean Mutual Information	75.00%
Equivalent number of attributes	64.28%
Noise to Signal Ratio	46.42%
Mean Mult. Correl. Coef.	67.85%
SDratio	82.14%

Table 6.15. Frequency with which characteristics are selected as decision nodes, c50tree.

Attribute	frequency
# classes	4.43%
# attributes	3.78%
# instances	4.29%
$\frac{\text{\# attributes}}{\text{\#instances}}$	2.07%
# unknown values	6.28%
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	1.80%
# nominal attributes	1.10%
max,min,mean,stdv of nominal attribute values	2.12% 1.15% 1.24% 0.60%
1..5 concentration histogram	1.52% 1.01% 0.69% 0.50% 0.78%
6..10 concentration histogram	0.27% 0.32% 0.13% 0.00% 0.73%
non computable conc. histogram	0.87%
1..5 concentration histogram with class	1.61% 1.24% 0.55% 1.24% 0.41%
6..10 concentration histogram with class	0.18% 0.00% 0.00% 0.00% 0.50%
non computable conc. histogram with class	0.00%
# continuous attributes	1.15%
1..5 correlation histogram	1.34% 1.61% 1.34% 1.24% 1.47%
6..10 correlation histogram	1.52% 2.03% 1.57% 1.75% 2.03%
non computable correlation histogram	1.24%
1..5 missing values histogram	0.00% 2.77% 3.23% 3.14% 3.23%
6..10 missing values histogram	1.89% 1.38% 0.18% 1.24% 0.27%
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.00%
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.23%
Binary Attributes	0.87%
Frac1	1.75%
First Canonical Correlation	3.09%
Mean Skew	1.38%
Mean Kurtosis	1.01%
Class Entropy	4.99%
Mean Attribute Entropy	2.17%
Mean Mutual Information	1.66%
Equivalent number of attributes	1.75%
Noise to Signal Ratio	0.73%
Mean Mult. Correl. Coef.	1.34%
SDratio	1.66%

Table 6.16. Frequency with which characteristics are selected among the different pairwise problems, c50boost.

Attribute	frequency
# classes	100%
# attributes	100%
# instances	100%
$\frac{\text{\# attributes}}{\text{\#instances}}$	100%
# unknown values	100%
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	100%
# nominal attributes	100%
max,min,mean,stdv of nominal attribute values	100% 100% 100% 100%
1..5 concentration histogram	100% 100% 100% 100% 100%
6..10 concentration histogram	92.85% 100% 39.28% 10.71% 100%
non computable conc. histogram	96.42%
1..5 concentration histogram with class	100% 100% 96.42% 100% 100%
6..10 concentration histogram with class	82.14% 14.28% 3.57% 32.14 64.28%
non computable conc. histogram with class	28.57%
# continuous attributes	100%
1..5 correlation histogram	100% 100% 100% 100% 100%
6..10 correlation histogram	96.42% 100% 100% 100% 100%
non computable correlation histogram	100%
1..5 missing values histogram	0.0% 100% 100% 100% 100%
6..10 missing values histogram	100% 100% 75% 100% 75%
$\frac{\text{\# continuous}}{\text{\# attributes}}$	60.71%
$\frac{\text{\# nominal}}{\text{\# attributes}}$	89.28%
Binary Attributes	96.42%
Frac1	100%
First Canonical Correlation	100%
Mean Skew	100%
Mean Kurtosis	100%
Class Entropy	100%
Mean Attribute Entropy	100%
Mean Mutual Information	100%
Equivalent number of attributes	100%
Noise to Signal Ratio	100%
Mean Mult. Correl. Coef.	100%
SDratio	100%

Table 6.17. Frequency with which characteristics are selected as decision nodes, c50boost.

Attribute	frequency
# classes	2.93%
# attributes	2.26%
# instances	3.67%
$\frac{\# \text{attributes}}{\#\text{instances}}$	2.41%
# unknown values	4.84%
$\frac{\# \text{unknown values}}{\# \text{attributes} * \# \text{instances}}$	2.24%
# nominal attributes	0.95%
max,min,mean,stdv of nominal attribute values	1.55% 0.82% 1.08% 0.95%
1..5 concentration histogram	1.25% 1.04% 1.09% 0.99% 0.92%
6..10 concentration histogram	0.52% 0.46% 0.006% 0.001% 0.80
non computable conc. histogram	0.91%
1..5 concentration histogram with class	1.31% 1.08% 0.60% 1.84% 0.73%
6..10 concentration histogram with class	0.30% 0.001% 0.000% 0.004% 0.01
non computable conc. histogram with class	0.0003%
# continuous attributes	1.31%
1..5 correlation histogram	1.96% 1.79% 1.59% 1.48% 1.58%
6..10 correlation histogram	1.54% 1.61% 1.45% 1.47% 1.45
non computable correlation histogram	1.24%
1..5 missing values histogram	0.0% 2.85% 2.92% 2.74% 2.83%
6..10 missing values histogram	1.96% 1.49% 0.22% 1.29% 0.31%
$\frac{\# \text{continuous}}{\# \text{attributes}}$	0.15%
$\frac{\# \text{nominal}}{\# \text{attributes}}$	0.25%
Binary Attributes	0.8%
Frac1	1.61%
First Canonical Correlation	2.63%
Mean Skew	1.63%
Mean Kurtosis	1.46%
Class Entropy	4.49%
Mean Attribute Entropy	2.77%
Mean Mutual Information	2.55%
Equivalent number of attributes	2.30%
Noise to Signal Ratio	2.00%
Mean Mult. Correl. Coef.	2.03%
SDratio	2.41%

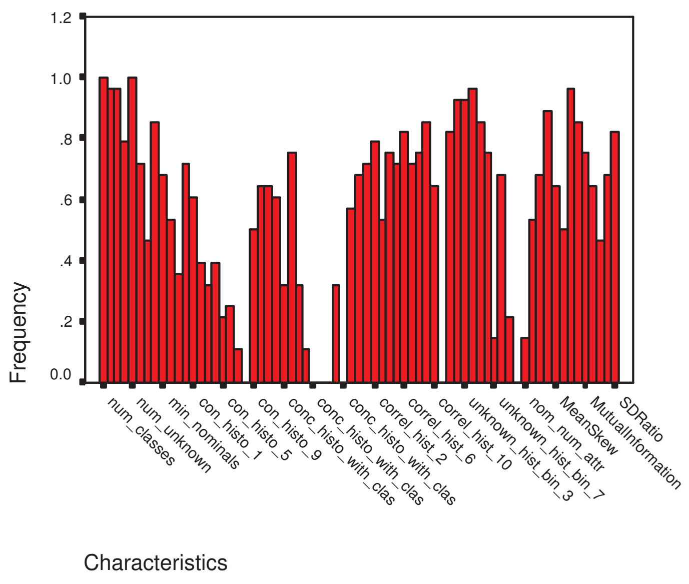

Fig. 6.3. Frequency with which characteristics are selected among the different pairwise problems, c50tree

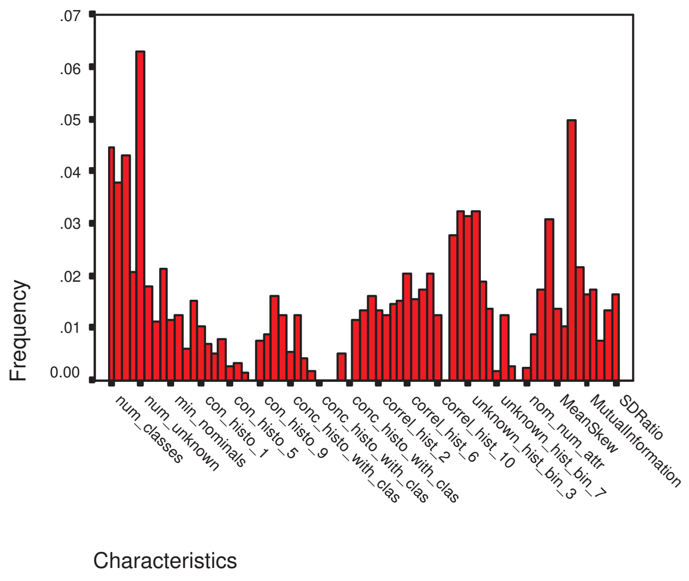

Fig. 6.4. Frequency with which characteristics are selected as decision nodes, c50tree

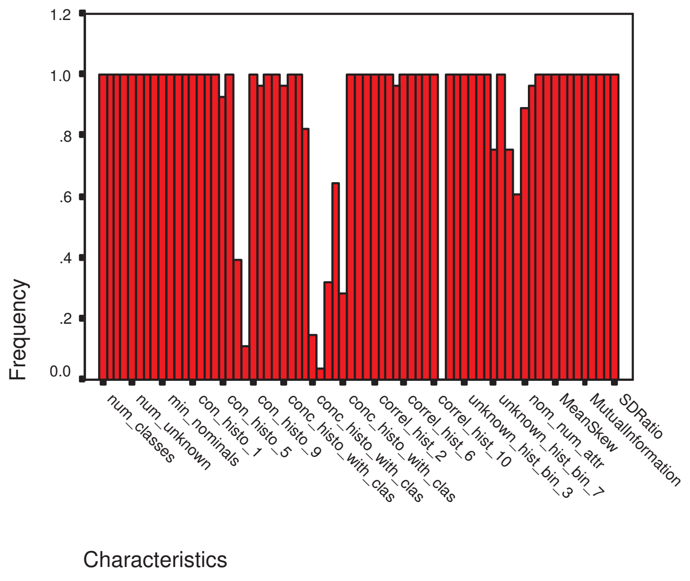

Fig. 6.5. Frequency with which characteristics are selected among the different pairwise problems, c50boost

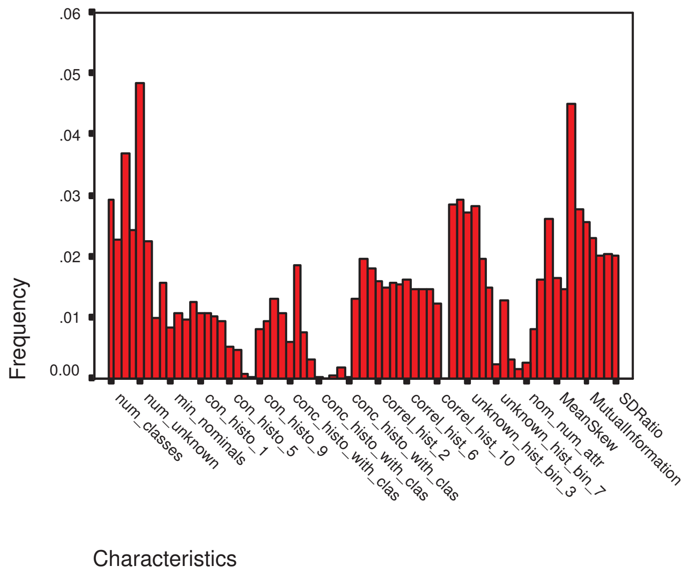

Fig. 6.6. Frequency with which characteristics are selected as decision nodes, c50boost

Chapter 7

Comparing Dataset Characterizations

Starting with the STATLOG project in 1994, and continuing until nowadays with the METAL project a variety of measures is used to describe and characterize datasets in order to predict the performance of learning algorithms. To our knowledge there is no systematic comparison of the different approaches of dataset characterization. The only exception to that is the work by Bensusan and Giraud-Carrier (2000), where they compare the performance of landmarking with that of an information based characterization of datasets in the spirit of STATLOG. The datasets used in this study were artificial datasets. The information-based description was consisting of, class entropy, equivalent number of attributes, average entropy of attributes, average mutual information, average joint entropy and signal-to-noise ratio, a rather limited set of characteristics which is actually a subset of the ones used in STATLOG. The experimental findings showed that landmarking outperforms the information-based description, however the results should be accepted with caution, since the study was done only on artificial datasets, the set of information based characteristics was rather limited and moreover there was no control on the statistical significance of the results.

The goal of this chapter is to perform a systematic and controlled comparison of different dataset characterizations, on real world datasets. We will examine five different sets of characteristics. Four of them follow the statistical/ information-based approach first presented in STATLOG and the fifth is the landmarking approach of characterizing a dataset. More specifically we will examine the following sets:

- *statlog*, the set of characteristics used in the STATLOG project, whose description was already given in section 4.1.

- *dct*, a richer set of dataset characteristics extracted from the DCT tool, which was developed as a result of the METAL project.

- *histo*, the set of characteristics that we have established.
- *histo-limited*, a smaller version of *histo*.
- *land*, the set of landmarkers given in section 4.1.

In section 7.1 we will give a more detailed description of the *dct* and of the *histo-limited* sets of characteristics. In a later section, 7.4, we will also explore the combined use of some of these sets of characteristics in order to improve predictive performance.

The comparison will involve 65 real world datasets mainly from the UCI repository and the METAL project[1]. The number of datasets should have been higher if the landmarking tool had not failed in characterizing a considerable number of datasets. We will use two different meta-learning frameworks to perform the comparison. The first one is the pairwise framework that we have developed in chapter 3. The second one is a simpler approach to meta-learning and the main goal is the prediction of the learner that will achieve the highest accuracy. Here we do not use pairwise comparisons, we have just one simple meta-learning problem where the goal is to predict the algorithm that achieves the highest accuracy for a given dataset. The instances of the meta-learning dataset are the descriptions of the datasets along with a class label which gives the algorithm with the highest accuracy on the dataset, as this is determined by 10 fold stratified cross validation. No kind of test of statistical significance is used in order to select the best algorithm, just the absolute value of the accuracy as it is estimated by the cross validation.

We will use c50boost in both frameworks as a metalearner. The evaluation strategy for the second meta-learning framework will be the accuracy of c50boost, as it is estimated by 10 fold stratified cross validation.

7.1 Description of the dataset characterizations being compared

We will describe only the *dct* and *histo-limited* sets of characteristics since the descriptions of *land, histo* and *statlog* were already given in previous chapters.

The *dct* set of characteristics is an extension of the *statlog* based set of characteristics, that includes 33 features. The additional characteristics include the following descriptions of missing values: the total number of missing values, the percentage of missing values, the number of instances with missing values and the percentage of instances with missing values. A new characteristic that gives the number of linear discriminant functions produced when a linear discriminant algorithm is applied to the dataset. The number of outliers in the dataset. A number of characteristics that describe attribute class associations; these are the gini index, (Breiman et al., 1984), attribute relevance and g-function, (Cooper & Herskovits, 1992), and finally the multiple correlation coefficient of

[1]The complete list of the 65 datasets, can be found in section B.1 of the appendix

each attribute with the other attributes. Since the last four characteristics are computed for every attribute of a dataset, apart from their mean value we also include their minimum and maximum values.

The *histo-limited* set, is just the set we obtain from *histo*, when we remove the characteristics already used in STATLOG. The set consists of the characteristics numbered 1 to 55 given in table 4.1. We decided to add this set of characteristics in the comparative study in order to get an indication of the discriminatory power of the histogram based characteristics, from which this set mainly consists.

7.2 Pairwise Framework

Before proceeding on giving the results of the comparative study we will give the details of the meta-learning problems defined by the 65 datasets used in the comparative study. In what concerns the pairwise meta-learning problems we give in table 7.1 the class distribution for each one of them, along with the default accuracy. The average default accuracy over all the 28 meta-learning problems is 54.45%. For any set of characteristics to be satisfactory, its mean accuracy over all the meta-learning problems should be better than the average default accuracy. Moreover it should outperform the default accuracy on the individual problems. It is for this reason that apart from comparing the different sets of characteristics between them, we also compare them with the default accuracy.

In what concerns the final suggestion we give in table 7.2 the frequencies of groups of inducers that are ranked in the top position. Again for any characterization strategy to be successful, it should give better results than just predicting the group that most often takes the first position. This corresponds to the default accuracy of the final suggestion and it is with this quantity that we will be comparing the strict accuracy (here the inducer taking most often the top position is Lindiscr with a frequency of 13.84%). Again we will compare all characterizations not only between them, but also with that default accuracy.

The number of comparisons involved is 15, four different ways of characterizing a dataset and the default accuracy. So we have to take into account the multiplicity effect and apply the Bonferroni adjustment. The new level of significance will be now 0.0034. For any result to be significant it has to achieve a significance level less than that.

Before continuing let us make a remark on the way that the meta-learning problems are constructed for the *land* set of characteristics. In section 4.1 we gave the list of landmarkers used in *land*, we can see that some of them, Naive Bayes, 1-nearest neighbor and linear discriminants, are also part of the pool of inducers from which the selection is performed. It is obvious that these landmarkers cannot be used as dataset descriptors in those pairwise meta-learning problems that they are involved, and consequently they are removed. For example we remove from the set of landmarkers, the Naive Bayes landmarker from all the pairwise meta-learning problems where one of the inducers is the Naive Bayes inducer.

7.2.1 Results on the Pairwise Meta-Learning Problems

All the five ways of characterizing a dataset give poor results on the pairwise meta-learning problems. Their average improvement over the average default accuracy is quite small. The highest is that of the *dct* set, (5.11%) and the lowest that of *land* where actually we cannot talk for an improvement but for a deterioration with respect to the average default accuracy of -1.7%, (Table 7.3)[2]. Examining the results with respect to the significant wins, (Table 7.4), again we see that all the five sets find it very difficult to statistically outperform the default accuracy. For *statlog* and *land* there is no pairwise problem for which their performance is statistically significantly better then default accuracy. *dct, histo* and *histo-limited*, outperform the default accuracy in a statistically significant level only for two to three pairwise problems out of the 28. One possible explanation for such a disappointing performance could be the limited number of training examples, as we have seen the training set consists of only 65 datasets. In what concerns the statistical significance of the differences between the sets we see that none of them outperforms the others in a significant level for any of the pairwise problems.

With these results it is quite hard to draw any reliable conclusions about the discriminating power of the examined characterizations; we can say that, based mainly on the results of the significant wins of the sets over the default accuracy, we can divide them to two groups. In the first group we have *dct, histo* and *histo-limited* that exhibit significant wins over the default accuracy for a limited number of pairwise problems, while on the second group we have *land* and *statlog* that never outperform the default accuracy in a statistically significant way.

7.2.2 Results on the final suggestion

When examining the performances of the different sets with respect to the quality of the final prediction that they provide the results are somehow different from the ones obtained on the pairwise problems. From the five sets, only two, (*histo,histo-limited*) achieve a strict accuracy that overpasses the default accuracy of 13.84%. The three remaining sets, *dct, statlog, land*, have a similar strict accuracy which is around 7%, considerably less than the default, (Table 7.5), in other words they are not useful in predicting the exact group of inducers that takes the top position for a specific dataset. If we compute the statistical significance of the differences, (Table 7.6), between the sets we again observe that none of them significantly outperforms the others in terms of the strict accuracy. Although the differences are quite high for some pairs, for example *histo-limited* outperforms *dct* almost by 14%, however the differences are not significant at the 0.0034 level. When comparing with the default accuracy none of *histo* and *histo-limited* outperforms it in a statistically significant level.

[2]The complete accuracy results on each pairwise problem can be found in section B.2 of the appendix

Table 7.1. Class Distributions for each of the pairwise meta-learning problems on the 65 datasets.

(algo–x, algo–y) pairs	algo–x	algo–y	tie	Default Accuracy
c50rules c50boost	6.15%	16.92%	76.92%	76.92%
c50tree c50boost	7.69%	26.15%	66.15%	66.15%
c50tree c50rules	4.61%	18.46%	76.92%	76.92%
Lindiscr c50boost	10.76%	49.23%	40.00%	49.23%
Lindiscr c50rules	10.76%	43.07%	46.15%	46.15%
Lindiscr c50tree	13.84%	41.53%	44.61%	44.61%
Ltree c50boost	9.23%	30.76%	60.00%	60.00%
Ltree c50rules	13.84%	20.00%	66.15%	66.15%
Ltree c50tree	23.07%	21.53%	55.38%	55.38%
Ltree Lindiscr	43.07%	12.30%	44.61%	44.61%
IBL c50boost	3.07%	38.46%	58.46%	58.46%
IBL c50rules	4.61%	36.92%	58.46%	58.46%
IBL c50tree	7.69%	35.38%	56.92%	56.92%
IBL Lindiscr	38.46%	24.61%	36.92%	38.46%
IBL Ltree	15.38%	43.07%	41.53%	43.07%
NB c50boost	7.69%	53.84%	38.46%	53.84%
NB c50rules	10.76%	49.23%	40.00%	49.23%
NB c50tree	15.38%	49.23%	35.38%	49.23%
NB Lindiscr	29.23%	27.69%	43.07%	43.07%
NB Ltree	7.69%	50.76%	41.53%	50.76%
NB IBL	18.46%	36.92%	44.61%	44.61%
ripper c50boost	15.38%	35.38%	63.07%	63.07%
ripper c50rules	4.61%	26.15%	69.23%	69.23%
ripper c50tree	10.76%	23.07%	66.15%	66.15%
ripper Lindiscr	38.46%	27.69%	33.84%	38.46%
ripper Ltree	9.23%	35.38%	55.38%	55.38%
ripper IBL	29.23%	16.92%	53.84%	53.84%
ripper NB	46.15%	20.00%	33.84%	46.15%
Average				54.45%

Shifting now to the performance in terms of the loose accuracy the situation is similar, (Table 7.5). The *histo* and *histo-limited* sets exhibit the highest loose accuracy, around 50%. That is for about 50% of the cases the set of inducers that they propose is a subset of the truly best inducers. *dct* and *statlog* follow with similar performance and at the last position we have the *land* set. Nevertheless when we calculate the statistical significance of the differences, (Table 7.7), we see that there is no set of characteristics that significantly outperforms any other set.

To summarize the above, in terms of the final suggestion the default accuracy is quite hard to beat and even harder to beat it at a statistically significant level. Nevertheless the histogram oriented sets, *histo* and *histo-limited* exhibit a performance that is better than the default. Between the five different ways of characterizing a dataset there is evidence to support the claim that histogram based characterization may provide increased discriminatory power over the other methods of characterization; but still the results are not conclusive.

Table 7.2. Groups of inducers that were ranked at the top for more than 3% of the 65 datasets.

Group	Frequency	Percent
Lindiscr	9	13.84%
c50boost	8	12.30%
Ltree	6	9.2%
c50rules c50tree Ltree	4	6.15%
c50rules c50boost c50tree	3	4.61%
NB	2	3.07%
IBL	2	3.07%
Lindiscr Ltree	2	3.07%
c50rules c50boost c50tree Ltree IBL ripper	2	3.07%
c50rules c50boost c50tree Lindiscr Ltree ripper	2	3.07%
c50rules c50boost c50tree Lindiscr Ltree IBL NB ripper	2	3.07%
c50boost IBL	2	3.07%
c50boost Ltree	2	3.07%
c50boost c50tree	2	3.07%

Table 7.3. Mean accuracies and improvement over the mean default accuracy, for each one of the five different dataset characterizations over the 28 meta-learning problems.

Characterization	Accuracy	Improvement
dct	59.56%	5.11%
land	52.75%	-1.70%
histo	59.18%	4.73%
histo-limited	57.80%	3.35%
statlog	57.14%	2.69%

7.2.3 Discriminatory power of the characteristics: c50boost on the 65 datasets

In section 6.4 we examined the discriminatory power of the characteristics of the *histo*, based on how often these were selected in the models that the learning algorithms constructed for each of the pairwise problems. The characterization was done based on the results that we obtained when training took place on the manipulated datasets. In order to verify the results obtained there we will repeat the same procedure here, but this time we will use only the 65 datasets used in the comparison study of this chapter. Furthermore we will only examine the models produced by c50boost. Since the number of examples is limited, the c50boost algorithm is not able to grow trees as big as when we used 1075 examples, this results in fewer attributes selected for each pairwise problem, and accordingly the selectivity of each characteristic is now reduced. In fact the average number of characteristics used in each pairwise problem drops from 61.53, (Table 6.13), to 36.46, (Table 7.8). In general the models produced from the 1075 examples were much more complex than the ones produced from the 65 examples, the availability of a big number of training instances allows decision tree algorithms to grow much more complex hypothesis, (Oates & Jensen, 1998). In order for the results to be comparable with the ones presented in section 6.4 we will limit the presentation only to the frequency with which each

Table 7.4. Distribution of significant wins, based on the McNemar's test, over the 28 pairwise meta-learning problems. In a triplet AA/BB/CC, AA is the number of significant wins of the row characterization, BB the number of significant wins of the column characterization and CC the number of ties.

Characterization	land	histo	histo-limited	statlog	default
dct	0/0/28	0/0/28	0/0/28	0/0/28	2/0/26
land		0/0/28	0/0/28	0/0/28	0/0/28
histo			0/0/28	0/0/28	3/0/25
histo-limited				0/0/28	3/0/25
statlog					0/0/28

Table 7.5. Results on the final suggestion of the five ways of characterizing a dataset.

Characterization	Strict Accuracy	Improvement	Loose Accuracy
dct	6.15%	-7.69%	43.08%
land	7.69%	-6.15%	36.92%
histo	15.38%	1.54%	52.31%
histo-limited	20.00%	6.16%	47.69%
statlog	7.69%	-6.15%	41.54%

characteristic is selected to become a decision node[3].

In table 7.9, we give the frequency with which the various characteristics are selected by the c50boost inducer; we will compare these results with the selection rates of the characteristics as they were determined on the 1075 datasets, table 6.17, in figure 7.1 we give the graph bars that correspond to the two tables. We can make the following remarks; the selectivity of the simple characteristics, like the number of attributes, number of instances or the ratio of attributes to examples, still remains quite high. The statlog based characteristics like class entropy, entropy of attributes, mutual information etc, also retain their high selectivity, although the selection rate is slightly reduced compared to the one observed on the 1075 training instances. Two notable exceptions to that, are the SDratio and Mean Multiple Correlation Coefficient characteristics, whose selection rate went down to zero. In what concerns the characteristics that describe the patterns of missing values, whereas they exhibited very high selectivity on the training set of 1075 instances, their selection rate is now considerably reduced. For example the selection rate of the number of unknown values drops from 4.84% down to 1.56%, the selection rate of the percent of unknown values drops from 2.24% down to 0.35%. A similar decrease is also observed in the histogram of missing values, where now there are more bins that are never selected, and the selection rate of the ones that are still used, is considerably reduced. An explanation could be the fact that in the 1075 datasets, there were many manipulated datasets whose difference was mainly the pattern of missing values. This could force c50boost to consider the characteristics describing the patterns of missing values highly discriminating and use them with a high frequency. Nevertheless let us note here that fsIBL was not putting a high emphasis on

[3]The selection rates of the characteristics for each pairwise problem can be found in section B.4 of the appendix.

Table 7.6. Results of the McNemar test comparing the accuracies of the five characterizations, in terms of the *strict accuracy*. + indicates a significant win for the row characterization, − a significant win for the column characterization, and = a tie.

	land	histo	histo-limited	statlog	default
dct	=(1.00)	=(0.113)	=(0.03)	=(1.000)	=(0.227)
land		=(0.267)	=(0.08)	=(1.000)	=(0.342)
histo			=(0.50)	=(0.227)	=(1.000)
histo-limited				=(0.043)	=(0.386)
statlog					=(0.422)

Table 7.7. Results of the McNemar test comparing the performances of the five characterizations, in terms of the *loose accuracy*. + indicates a significant win for the row characterization, − a significant win for the column characterization, and = a tie.

	land	histo	histo-limited	statlog
dct	=(0.454)	=(0.239)	=(0.647)	=(1.000)
land		=(0.067)	=(0.211)	=(0.628)
histo			=(0.606)	=(0.146)
histo-limited				=(0.454)

the patterns of missing values. Concerning the remaining characteristics described with the use of histograms, the correlation coefficient exhibits similar high selectivity as in the case of the 1075 instances. The bins of the histogram of the concentration coefficient between the attributes, have now a higher selection rate; even the upper five bins that in the case of the 1075 instances had a quite small selectivity are now selected with higher frequencies. This pattern was already observed in the feature sets produced by fsIBL. Finally in what concerns the histogram of the concentration coefficient between the attributes and the class attribute the selectivity is slightly increased for all of the bins, with the upper five bins still having a low selection rate. Finally some of the characteristics that had small selection rate, like the percent of continuous or discrete attributes have now a zero selection rate.

In general the central observation is that the main difference between the models produced on the 1075 datasets and the ones produced on the 65 datasets by c50boost, lies on the much smaller emphasis on the patterns that describe the missing values. We can also note a similarity between the selection rates produced by fsIBL on the 1075 instances and the selection rates produced by c50boost on the 65. We believe that the discriminatory power of the characteristics as it is given by c50boost on the 65 instances and fsIBL on the 1075, is more representative of their true status.

7.3 Simple Framework

Under the simple framework hypothesis the goal of the meta-learning problem is to predict for a specific dataset the inducer that will achieve the highest accuracy. The class distribution of the corresponding meta-learning dataset is

Table 7.8. Number of features used, by c50boost, for each pairwise problem, when trained on the 65 datasets.

pair	number of features
c50rules c50boost	27
c50tree c50boost	36
c50tree c50rules	28
Lindiscr c50boost	40
Lindiscr c50rules	39
Lindiscr c50tree	36
Ltree c50boost	41
Ltree c50rules	32
Ltree c50tree	34
Ltree Lindiscr	38
IBL c50boost	31
IBL c50rules	35
IBL c50tree	42
IBL Lindiscr	37
IBL Ltree	39
NB c50boost	35
NB c50rules	44
NB c50tree	41
NB Lindiscr	44
NB Ltree	37
NB IBL	35
ripper c50boost	37
ripper c50rules	34
ripper c50tree	34
ripper Lindiscr	31
ripper Ltree	35
ripper IBL	41
ripper NB	38
Average	36.46

given in table 7.10. In the table we can see how many times an inducer, from the pool of inducers, achieves the highest accuracy over all the 65 datasets. The inducer positioned most often at the top is c50boost, with a frequency of 32.30%. This quantity corresponds to the default accuracy of the simple meta-learning framework. For any strategy of characterization to be useful it should achieve an accuracy which is better than this default accuracy. Again the significance level is readjusted, to take into account the multiplicity effect, to 0.0034.

In what concerns the construction of the meta-learning dataset we have to note that in the case of the *land* set, for obvious reasons, we have to omit all these landmarkers that are full fledged inducers and are also a part of the pool of inducers, i.e. Lindiscr, Naive Bayes and 1-nearest neighbor. This leaves us with the four following features in the *land* set: decision node, worst node, randomly chosen node and elite 1-nearest neighbor.

Before continuing with the presentation of the results let us make more clear the relation between the two meta-learning frameworks. The class distribution of the simple meta-learning framework, table 7.10, corresponds conceptually to the distribution of the groups of inducers that take the top position for the final

Table 7.9. Frequency with which characteristics are selected as decision nodes, by c50boost when trained on the 65 datasets.

Attribute	frequency
# classes	4.07%
# attributes	2.95%
# instances	3.00%
$\frac{\text{\# attributes}}{\text{\#instances}}$	2.73%
# unknown values	1.56%
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.35%
# nominal attributes	1.03%
max,min,mean,stdv of nominal attribute values	1.74% 2.59% 0.40% 0.85%
1..5 concentration histogram	2.33% 1.07% 1.65% 1.07% 1.43%
6..10 concentration histogram	1.25% 1.56% 1.97% 0.85% 0.94%
non computable conc. histogram	0.76%
1..5 concentration histogram with class	2.73% 1.25% 1.83% 2.95% 1.21%
6..10 concentration histogram with class	1.29% 0.49% 0.00% 0.76% 0.08%
non computable conc. histogram with class	0.0008%
# continuous attributes	1.47%
1..5 correlation histogram	3.40% 1.74% 1.16% 1.16% 1.29%
6..10 correlation histogram	2.10% 1.74% 1.21% 1.65% 2.64%
non computable correlation histogram	1.03%
1..5 missing values histogram	0.00% 0.62% 1.61% 1.92% 1.52%
6..10 missing values histogram	1.74% 0.00% 1.56% 0.00% 0.00%
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.00%
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.00%
Binary Attributes	0.00%
Fract	1.03%
Cancor	2.51%
Mean Skew	2.73%
Mean Kurtosis	2.01%
Class Entropy	3.72%
Entropy Attributes	1.16%
Mutual Information	2.19%
Equivalent number of attributes	3.54%
NoiseSignal Ratio	2.42%
AttrMultiCorrel	0.00%
SDratio	0.00%

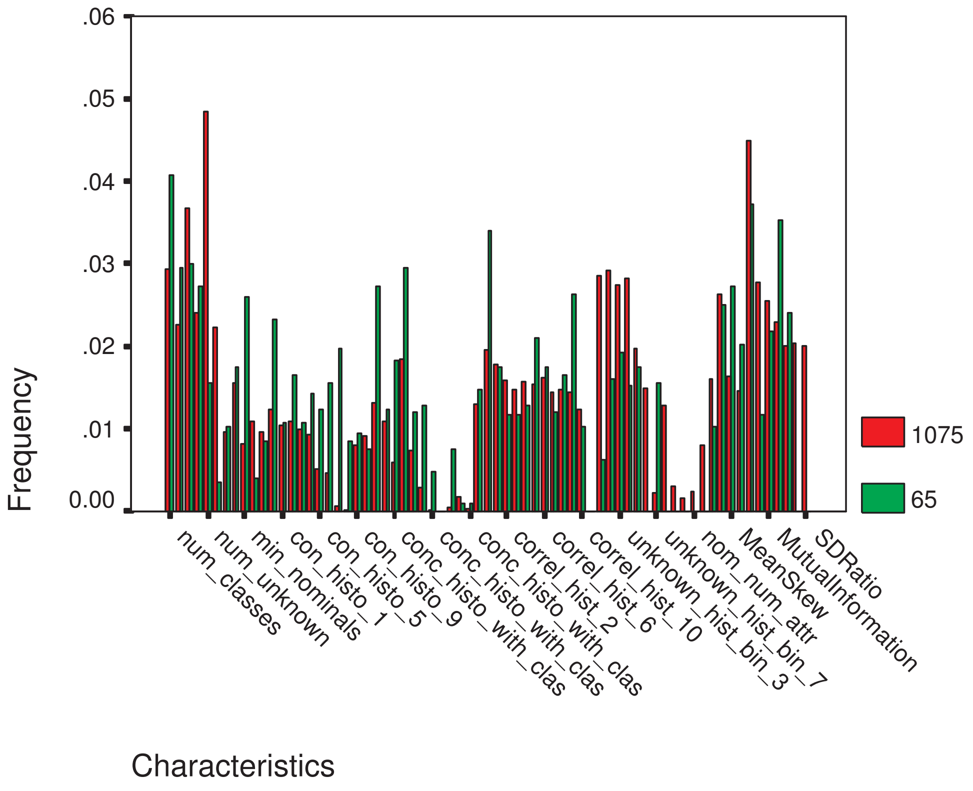

Fig. 7.1. Frequency with which characteristics are selected as decision nodes on the 1075 and 65 datasets by c50boost

suggestion of the pairwise framework, table 7.2. A source of confusion could be the fact that in the pairwise framework the group of inducers that get most often the top position consists of Lindiscr while in the simple framework the inducer that most often gets the top position is c50boost. The reason for that is that in the first framework statistical significances are taken into account whereas in the second we simply ignore them. So when we see in table 7.2 that Lindiscr is the one that gets more often the top position, (9 datasets), this means that for these datasets Lindiscr is statistically significant better than *all* the other inducers. From table 7.10 we can see that Lindiscr has the highest accuracy in 13 datasets, but only in 9 of them as it is stated in table 7.2 it is significantly better than all the other inducers. Obviously the same applies for c50boost. When in table 7.2 we see that it gets the top position in 8 datasets, this means that for these datasets it is significantly better than all the other inducers, even though as it is stated in table 7.10 it has the highest accuracy in 21 datasets.

Table 7.10. Class distribution for the simple meta-learning framework

Class	#	Frequency
c50boost	21	32.30%
Lindiscr	13	20.00%
Ltree	12	18.46%
c50rules	9	13.84%
NB	3	4.61%
IBL	3	4.61%
c50tree	3	4.61%
ripper	1	1.53%

7.3.1 Results for the simple meta-learning framework

When we examine the accuracies that the five methods of characterization achieve, table 7.11, we can distinguish three groups of sets that achieve similar performance. The top group consists of only the *histo* set, which achieves the highest accuracy with an improvement over the default of 18.47%. In the second group we have *histo-limited, statlog* and *dct* with a similar performance and an improvement over the default of around 10%. The last group consists of the *land*[4] which is the only method of characterization that has a performance worse than the default, with a deterioration of 7.68%. Checking the statistical significance of the differences between the characterizations, table 7.12, we find only one pair where the difference is significant, that of *histo, land.* With respect to the default accuracy again none of the sets manages to beat it in a statistically significant level.

Table 7.11. Accuracy and improvement over the default accuracy, of the five sets of dataset characterization, for the simple meta-learning framework.

Characterization	Accuracy	Improvement
dct	41.54%	9.24%
land	24.62%	-7.68%
histo	50.77%	18.47%
histo-limited	43.08%	10.78%
statlog	41.54%	9.24%

Table 7.12. Results of the McNemar test comparing the accuracies of the five characterizations, under the simple framework. + indicates a significant win for the row set, - a significant win for the column set, and = a tie.

	land	histo	histo-limited	statlog	default
dct	=(0.045)	=(0.263)	=(1.000)	=(1.000)	=(0.326)
land		-(0.002)	=(0.030)	=(0.037)	=(0.358)
histo			=(0.382)	=(0.211)	=(0.044)
histo-limited				=(1.000)	=(0.281)
statlog					=(0.307)

[4]Remember here that the *land* set consists of only four features

7.4 Combining Characterizations

So far the dataset characterizations that we examined were based on a unique paradigm, either a description of the datasets in terms of the statistical and information based properties, (*dct, statlog, histo, histo-limited*) or a description based on the performance of simple learners, (*land*). In an effort to further improve the description of the datasets we will combine two different paradigms in a single characterization. We will combine the *histo* characterization, the best performing among the statistical and information based, with the *land* characterization. We will evaluate the new set, which we will call *combined*, both under the pairwise and the simple metalearning frameworks.

In table 7.13 we give the performance results of the *combined* characterization for the pairwise meta-learning framework. The average accuracy overpasses the default average accuracy by 4.01% with a value of 58.46%. Compared to its constituent sets, i.e. *land* and *histo*, its average accuracy is slightly worse than that of *histo* and better than *land*, but the differences are not statistically significant for any of the 28 meta-learning problems. Compared to the performance of the default accuracy it is significantly better in three out of the 28 problems, for the remaining ones the differences are not significant, (table 7.14).

In what concerns the performance of the final suggestion in terms of the strict accuracy, *combined* is quite better than *land* but only marginally better from *histo*, in both cases the differences are not statistically significant. Measuring the performance with loose accuracy, *combined* is better than *land* but worse than *histo*. But again the differences are not statistically significant. To summarize the results on the pairwise framework, the *combined* set exhibits a better performance than *land*, but it is slightly worse than the *histo* set, (Tables 7.13, 7.15).

Table 7.13. Performance of the *combined* characterization, in terms of average, strict and loose accuracy over the 28 metalearning problems.

	Combined	Impr. over default	land	histo
Average Accuracy	58.46%	4.01%	52.75%	59.18%
Strict Accuracy	16.92%	3.08%	7.69%	15.38%
Loose Accuracy	49.23%		36.92%	52.32%

Table 7.14. Distribution of significant wins of *combined* compared with *land* and *histo* over the 28 meta-learning problems. In a triplet AA/BB/CC, AA is the number of significant wins of *Combined*, BB the number of significant losses and CC the number of ties.

	land	histo	default
combined	0/0/28	0/0/28	3/0/25

The results are similar when we examine the performance of the *combined* set under the simple framework, Table 7.16. The *combined* characterization performs better than the default accuracy and the *land* characterization but

Table 7.15. Significance levels comparing *combined* with its constituent sets and the default, final suggestion.

	land	histo	default
Strict Accuracy	=(0.181)	=(1.000)	=(0.851)
Loose Accuracy	=(0.170)	=(0.790)	

Table 7.16. Performance of the *combined* characterization on the simple meta-learning framework.

Accuracy	Impr. over default	land	histo
44.62%	12.32%	24.62%	50.77%

worse than the *histo*. Once more the differences are not statistically significant, table 7.17.

In general the performance of the *combined* in all the frameworks is lower than that of one its constituent sets, i.e. *histo*. The incorporation of the landmarking based characteristics in the *histo* set seems to harm performance, still it should be pointed that the results are not conclusive due to the absence of statistical significance. Furthermore what should be explored is the possibility to use a subset of the *combined* set that would be selected with a feature selection mechanism.

7.5 Summary and Conclusions

In this chapter we conducted a systematic study of the discriminatory power of different ways of characterizing datasets, under two meta-learning frameworks, on real world datasets. To our knowledge it is the first study of this kind.

In what concerns the first meta-learning framework and the performance on the pairwise problems the results were not conclusive. Two out of the five different characterizations, *statlog* and *land* could not outperform the default accuracy in a statistical significant level in any of the 28 pairwise problems. Actually *land* had an average accuracy which was even worse than the average default accuracy. The remaining three characterizations outperformed the default accuracy for only two or three pairwise problems. Moving to the performance with respect to the final suggestion, only the histogram based approaches managed to beat the default accuracy, in terms of strict accuracy. The remaining three approaches performed much worse than the corresponding default accuracy. Unfortunately the differences of the histogram based approaches with respect to the default accuracy were not statistically significant.

The results on the simple meta-learning framework are slightly different. Here four out of the five different characterizations beat the default accuracy, *histo, histo-limited, dct* and *statlog*; the only one that exhibited an accuracy lower than the default was the *land* set, probably due to the fact that we used a more limited set of the initial landmarkers. The *histo* set achieves the top performance with a considerable improvement over the default, however once

Table 7.17. Significance levels comparing *combined* with its constituent sets and the default accuracy for the simple meta-learning framework

land	histo	default
=(0.016)	=(0.208)	=(0.1858)

more this improvement is not statistically significant.

In an effort to further improve the performance of the predictions we examined the combined use of two different paradigms of characterizing a dataset, one based on the landmarkers and the other based on the histogram representation of the statistical and information based properties of the datasets. The results were not clear either for or against the combined use of the different paradigms since the observed differences were not statistically significant. The *combined* set exhibited a performance which was slightly worse than the *histo* set one of its constituents.

The most disappointing observation was that none of the dataset characterizations manage to beat the baseline performance, in any of the experimental frameworks that we used, in a statistically significant level. However the histogram based approaches exhibited systematically better performance than the other sets. For the first framework they were the only ones that manage to overpass the strict default accuracy, while for the second framework they were on the top two positions. Based on these observations we could argue that the use of histograms can only improve the performance in the meta-learning problems that we are dealing with, thus providing more reliable predictions. We believe that the poor performance is mainly due to the limited number of datasets used, and that with a greater number of datasets better results can be achieved.

As a byproduct of the comparative study under the pairwise framework, we analyzed the models produced by c50boost. This was done in order to determine the discriminatory power of the characteristics as it is established from real world datasets, and compare it with the one established from the manipulated datasets. The results produced showed some differences mainly with respect to the discriminatory power of the characteristics describing the patterns of missing values. This can be attributed to the presence of a considerable number of datasets, among the 1075 datasets used previously, that differed mainly in their pattern of missing values. Nevertheless the produced description by c50boost had similarities with the one produced by fsIBL on the 1075 datasets. We believe that this characterization is a more reliable one.

Chapter 8

Regression Based Meta-Learning

So far in all the previous chapters we handled the meta-learning problem as a typical classification task. In this chapter we will explore another alternative, where we cast the problem as a regression problem. We face meta-learning tasks as regression tasks whereby we look for relationships between the properties of a dataset and the performance of the classifiers. We adopt two different meta-learning frameworks. The first one is based on the pairwise meta-learning framework introduced in the previous chapters. The second one relies on the direct estimation of the expected error of inducers on a given set. The direct error estimates can be used either to perform inducer selection, i.e. select the most appropriate inducer for a dataset, or inducer ranking, i.e. rank the inducers with respect to their expected performance on a dataset.

We will evaluate the regression approach for both frameworks and examine the performance of regression models constructed from all the five sets of characteristics used in the comparative study of chapter 7. Finally we will compare the regression based ranking with zooming based ranking introduced in (Soares & Brazdil, 2000).

8.1 Related Work

Concerning the direct estimation of performance of learners, little experimental results with working systems has been reported. The idea of using regression to predict the performance of learning algorithms was first used by Gama and Brazdil (1995), while they continued on the framework adopted in STATLOG. They tried to directly predict the error of an algorithm for a specific dataset based on the characteristics of the dataset, as these were defined in the STATLOG project. For each of the learners they evaluated various regression models like linear regression, instance based regression and rule based regression. They report poor results in terms of the Normalized Mean Squared Error (NMSE).

From the 23x3x4=276 different regression models that they used only 63 were useful, i.e. had an NMSE of less than 1. They stopped there, without trying to use the regression models for model selection, or for ranking the algorithms.

Sohn in (Sohn, 1999) uses the results of STATLOG (i.e. same data characterization, same learning algorithms and same datasets) and constructs linear regression models that predict the errors of the learning algorithms on unseen datasets. As she is using the results of STATLOG, the study is limited to 19 datasets and 11 learning algorithms. To overcome the small number of datasets she used bootstrapping resampling to estimate the parameters of the regression models. The regression models used were simple linear regression models predicting the logit[1] transformation of the errors of the learners. The models were used to provide a ranking of the available learning algorithms. The results show that the statistical models produced exhibit high performance. However they must be interpreted cautiously because of the limited number of datasets used in the study.

A recent paper provided some initial results related to the use of estimated performances for model selection (Koepf et al., 2000). It shows that estimating performances leads to a better result in selecting a learner from a pool than learning through a repository of datasets classified in terms of the best performing algorithm in the pool. Using a pool composed by three classifiers, (Linear Discriminant, Quadratic Discriminant and 1-Nearest Neighbor) the paper indicates that regression (by M5, (Quinlan, 1992b)), when used to estimate the error of the three classifiers, selects the classifier with least error with better performance than using classification (with C5.0) to decide the best algorithm for a dataset. The experiments, however, were preliminary and concentrated only on one strategy of dataset characterization, on only three classifiers and were performed on artificially generated datasets.

A work with a regression like flavor is also the work on ranking with zooming (Soares & Brazdil, 2000). The goal there is to determine a preference order over a pool of classifiers, based on predictive accuracy and computational cost. The ranking for a new dataset is built by inspecting a number of k-nearest neighbors in a collection of reference datasets, that form a meta-dataset. In the meta-dataset each dataset is described by a number of features and labeled by the performance obtained by each classifier in the pool. The produced ranking is based on a preference function that weights cost and accuracy on the number of neighbors that are to be considered. This could be considered similar to kernel based regression, which fits a regression model to a neighborhood of an instance. The main difference though is that in ranking with zooming the preference function is static. The approach cannot be used as is to estimate accuracies of learners, but only to provide a relative ranking of them. One of the main limitations of the method is that it relies on a single global meta-model that is based on the k-nearest neighbor approach. Meta-models constructed by learners other than k-nearest neighbor, have been shown to give better results (Kalousis & Hilario, 2000b). Also the dataset characteristics that affect the

[1] $logit(x) = ln\frac{x}{x-1}$

performance of a learner vary from one learner to another (Kalousis & Hilario, 2001), but ranking with zooming requires the use of a single set of dataset characteristics, independently of the learner.

Our goal here is, to broaden the previous line of research, explore different strategies of dataset characterization for regression based meta-learning and compare the different approaches under a common framework. The experimental framework will be the same as the one used in chapter 7. That is, we will use the same five sets of dataset characteristics, the same 65 datasets on which the evaluation will take place, and the same pool of eight inducers.

8.2 Pairwise framework

We use the pairwise framework described in chapter 3. That is we construct $\binom{n}{2}$ pairwise metal-learning problems for n inducers, but unlike the classification based approach we learn on each of these problems using regression. The main idea is that instead of trying to predict which of the two inducers to use via classification, we try to directly estimate their relative error difference using regression and select among them the one with the lowest relative error. On the next level we combine all the pairwise predictions to take the final suggestion of the system, in the same way we did it for the classification based approach. An instance of a pairwise metalearning problem consists of a dataset description and the difference of the errors of the two inducers on the specific dataset.

Unlike the classification based approach we do not use any test of statistical significance. As a result the prediction for a pairwise problem can be only one of the two inducers involved and the notion of ties does not exist anymore. In table 8.1 we give the class distributions for each of the 28 meta-learning problems, the corresponding default accuracies, and the average default accuracy overall the 28 problems (67.36%). Since now we do not have ties the final suggestion of the system is a single inducer, the one that achieves the lowest relative error among all inducers. Consequently the final suggestion is not anymore the set of the best inducers and the notions of *strict* and *loose* accuracy merge to the notion of normal accuracy. The distribution of the inducers that take the top position has already been given in table 7.10. The default accuracy with which we will compare the performance of the final suggestion is again determined by the frequency of the top inducer, which is c50boost in 32.30% of the datasets.

8.2.1 Results on the Pairwise Meta-Learning Problems

All the five different ways of characterizing a dataset give poor results on the pairwise metalearning problems. The improvement over the average default accuracy is very small with the best characterization, *dct*, being better than the average default accuracy only by 3.02%. The *land* set is again by far the worst, having a performance which is lower than the average default accuracy

Table 8.1. Class Distributions for each of the pairwise regression based meta-learning problems on the 65 datasets.

(algo–x, algo–y) pairs	algo–x	algo–y	Default Accuracy
c50boost c50rules	67.69%	32.30%	67.69%
c50boost c50tree	75.38%	24.61%	75.38%
c50boost Lindiscr	69.23%	30.76%	69.23%
c50boost Ltree	56.92%	43.07%	56.92%
c50boost IBL	92.30%	07.69%	92.30%
c50boost NB	73.84%	26.15%	73.84%
c50boost ripper	83.07%	16.92%	83.07%
c50rules c50tree	72.30%	27.69%	72.30%
c50rules Lindiscr	63.07%	36.92%	63.07%
c50rules Ltree	63.07%	36.92%	63.07%
c50rules IBL	70.76%	29.23%	70.76%
c50rules NB	67.69%	32.30%	67.69%
c50rules ripper	76.92%	23.07%	76.92%
c50tree Lindiscr	63.07%	36.92%	63.07%
c50tree Ltree	58.46%	41.53%	58.46%
c50tree IBL	69.23%	30.76%	69.23%
c50tree NB	67.69%	32.30%	67.69%
c50tree ripper	69.23%	30.76%	69.23%
Lindiscr Ltree	27.69%	72.30%	72.30%
Lindiscr IBL	53.84%	46.15%	53.84%
Lindiscr NB	55.38%	44.61%	55.38%
Lindiscr ripper	44.61%	55.38%	55.38%
Ltree IBL	78.46%	21.53%	78.46%
Ltree NB	73.84%	26.15%	73.84%
Ltree ripper	72.30%	27.69%	72.30%
IBL NB	56.92%	43.07%	56.92%
IBL ripper	47.69%	52.30%	52.30%
NB ripper	44.61%	55.38%	55.38%
Average			67.36%

by 6.26%. In table 8.2[2] we give the average accuracy for each characterization. Examining the results with respect to the significant wins[3] (Table 8.3) we see that there is no characterization that manages to beat the default accuracy at a significant level. Moreover the differences among the characterizations for the big majority of the 28 meta-learning problems are not significant either. Note here that these results are not directly comparable with the results on the pairwise meta-learning problems with classification given in section 7.2, since the distribution of classes and the actual classes are different.

8.2.2 Results on the final suggestion

As it has already been mentioned the final suggestion of the system in the regression pairwise framework corresponds to the single inducer who is expected to

[2]The complete accuracy results on each pairwise problem can be found in section B.3 of the appendix

[3]We are taking again into account the multiplicity effect and apply the Bonferroni adjustment setting the significance level to 0.0034

Table 8.2. Mean accuracies and improvement over the mean default accuracy, for the five different dataset characterizations over the 28 meta-learning problems, for the regression based pairwise approach.

Characterization	Accuracy	Improvement
dct	70.38%	3.02%
land	61.10%	-6.26%
histo	69.29%	1.92%
histo-limited	69.45%	2.09%
statlog	68.19%	0.82%

Table 8.3. Distribution of significant wins, based on the McNemar's test, over the 28 meta-learning problems, for the regression based pairwise approach. In a triplet AA/BB/CC, AA is the number of significant wins of the row characterization, BB the number of significant wins of the column characterization and CC the number of ties.

Characterization	land	histo	histo-limited	statlog	default
dct	3/0/25	0/0/28	0/0/28	0/0/28	0/0/28
land		0/2/26	0/0/28	0/1/27	0/1/27
histo			0/0/28	0/0/28	0/0/28
histo-limited				0/0/28	0/0/28
statlog					0/0/28

achieve the highest accuracy for a given dataset. In that sense we are providing exactly the same type of predictions as in the case of the simple meta-learning framework for algorithm selection given in section 7.3. Table 8.4 gives the accuracy results for the five characterizations for the final suggestion. At the top position we have the *histo* set which is better than the default accuracy by 15.30% and again on the last position we have the *land* set of characteristics which is worse than the default accuracy by 12.30%. In what concerns the statistical significance of the results, no characterization outperforms the default accuracy in a statistical significant level (Table 8.5).

The accuracy results in table 8.4 are directly comparable with the corresponding results on inducer selection under the simple framework, table 7.11. It seems that the straight selection via classification has an advantage over the regression pairwise selection. It provides better results for all the five different ways of dataset characterization. The regression based approach exhibits the highest degradation in performance for the *statlog* and *histo-limited* characterizations, 13.85% and 6.16% respectively. For the remaining characterizations the degradation of performance is between 1% and 4%. The explanation for that difference in performance is that the pairwise selection is more prone to errors since an error in one of the pairwise selections can harm the final suggestion.

8.3 Direct accuracy prediction framework

The primary goal here is to predict the actual accuracy of an inducer on a new unseen dataset using a regression model constructed from the available training data. Then these predictions can be used either for inducer selection, or inducer

Table 8.4. Accuracy and improvement over the default accuracy, of the five sets of dataset characterization, for the final suggestion on the pairwise regression problem

Characterization	Accuracy	Improvement
dct	40.00%	7.70%
land	20.00%	-12.30%
histo	47.69%	15.30%
histo-limited	36.92%	4.62%
statlog	27.69%	-4.61%

Table 8.5. Results of the McNemar test comparing the accuracies of the five characterizations, in terms of the accuracy of the final suggestion, for the pairwise regression approach. + indicates a significant win for the row characterization, − a significant win for the column characterization, and = a tie.

	land	histo	histo-limited	statlog	default
dct	=(0.025)	=(0.301)	=(0.813)	=(0.098)	=(0.556)
land		-(0.002)	=(0.072)	=(0.423)	=(0.027)
histo			=(0.045)	=(0.012)	=(0.137)
histo-limited				=(0.211)	=(0.859)
statlog					=(0.570)

ranking.

A meta-dataset is constructed for each inducer. In order to do that, each dataset has to be characterized by a dataset characterization strategy. Every instance of the meta-dataset corresponds to a specific dataset and consists of the dataset characterization along with the accuracy of the inducer on that dataset as it is measured by 10 fold cross validation. The meta-dataset can then be treated as an ordinary regression problem.

8.3.1 Predicting accuracies

Regression was used to estimate the performance of classifiers using the different strategies of dataset characterization. Since the quality of the estimate depends on its closeness to the actual accuracy achieved by the classifier, the meta-learning performance is measured by the Mean Absolute Deviation (MAD). MAD is defined as the sum of the absolute differences between real and predicted values divided by the number of test items. It can be seen as measure of the distance between the actual values and the predicted ones.

In order to compare the estimation capabilities of the five strategies of dataset characterization we used a kernel method (Torgo, 1999) to perform regression on the meta-dataset. Kernel methods work in an instance-based principle and they fit a linear regression model to a neighborhood around the selected instance. It is straightforward to alter their distance metric in order to make better use of the semantics of the non-applicable values that occur in meta-attributes of *dct, statlog, histo* and *histo-limited.*

For each classifier meta-dataset, we run 10-fold cross-validation to assess the quality of performance estimations. The quality of the estimation is assessed by

Table 8.6. Kernel performance on estimating performance.

Classifier	dct	histo	histo-lim	land	statlog	dMAD
c50boost	0.112	0.123	0.122	0.050	0.119	0.134
c50rules	0.109	0.120	0.121	0.051	0.114	0.133
c50tree	0.109	0.125	0.123	0.054	0.116	0.137
Lindiscr	0.118	0.128	0.129	0.063	0.116	0.137
Ltree	0.105	0.115	0.113	0.041	0.108	0.132
IBL	0.120	0.140	0.137	0.081	0.133	0.153
NB	0.121	0.142	0.143	0.064	0.122	0.146
ripper	0.113	0.129	0.127	0.056	0.125	0.145

the MAD in the 10 folds, and it is compared with the default MAD (dMAD). The latter is the MAD obtained by predicting that the error of a classifier in a test dataset is the mean of the error obtained in the training datasets. dMAD is a benchmark for comparison, and one can think of it as the quantity corresponding to default accuracy in a typical classification problem. We expect regression to produce a smaller MAD than the dMAD. We have to note here that, in the case of landmarkers, whenever we build a model to predict the performance of a classifier that is a member of the set of landmarkers the corresponding landmarker is removed.

The quality of the estimation with the kernel method using different dataset characterization strategies is shown in table 8.6. The table presents the MAD in the 10 folds for every regression problem and the dMAD. Landmarking outperforms the other by far and produces estimated accuracies with a MAD smaller than 0.081 for every classifier. This means that the average error of the estimated accuracy in unseen datasets will be in the worst case (that of mlcib1) 8.1%.

The rest of the characterization strategies do not produce estimates as good as those produced by landmarking. One could suspect that this is because the meta-dataset is relatively small when compared to the large number of meta-attributes used by these two strategies of dataset characterization. To check whether reducing the dimensionality of the problem would significantly improve the estimates, we performed feature selection through wrapping on the four meta-datasets. The estimates, however, were not greatly improved. We conclude that landmarking performs best in performance estimation using kernel.

To examine whether the results presented are significant we performed paired t-tests of significance. In Table 8.7 we give the results of the paired t-test between each model and the dMAD. In this table and in the following ones, + indicates that the method is significantly better than the default, = signifies that there is no difference, and − that the method is significantly worse then the default. Since we have multiple comparisons with the default MAD we will once again adjust the significance level for the multiplicity effect, the new significance level will be 0.006. The table shows that the performance of *land* is always significantly better than the default. For the other characterizations the differences are not significantly different. However the *dct* characterization

Table 8.7. P-values of paired T-tests of significance comparing with the dMAD.

Classifier	dct	histo	histo-limited	land	statlog
c50boost	= (0.112)	= (0.430)	= (0.361)	+ (0.00)	= (0.287)
c50rules	= (0.075	= (0.346)	= (0.319)	+ (0.00)	= (0.154)
c50tree	= (0.045)	= (0.373)	= (0.242)	+ (0.00)	= (0.122)
Lindiscr	= (0.110)	= (0.485)	= (0.548)	+ (0.00)	= (0.107)
Ltree	= (0.030)	= (0.153)	= (0.115)	+ (0.00)	= (0.051)
IBL	= (0.037)	= (0.361)	= (0.269)	+ (0.00)	= (0.173)
NB	= (0.036)	= (0.758)	= (0.807)	+ (0.00)	= (0.058)
ripper	= (0.024)	= (0.243)	= (0.217)	+ (0.00)	= (0.152)

Table 8.8. Average Spearman's Correlation Coefficients with the True Ranking

	models	
Characterization	Kernel	Zooming
default	0.330	0.330
dct	0.435	0.341
histo	0.394	
histo-limited	0.405	0.371
land	0.180	
land-		0.190
statlog	0.385	

appears to have a small advantage over them, exhibiting lower significance levels.

Furthermore, *land* is always significantly better than the rest of the characterization sets for all the eight different learning algorithms. Between the four sets, the differences are not statistically significant for any of the 8 learners.

In conclusion we can say that the use of landmarkers to perform accuracy estimation is a method with very good performance and low estimation error, significantly better than the others. The reason is that landmark based characteristics are better suited for that type of task: they provide a direct estimation of the hardness of the problem since they are themselves performance estimations. On the other side, the rest of the sets give an indirect description of the hardness of the problem, through the use of characteristics like attributes correlations, which are more difficult to directly associate with accuracy.

8.3.2 Ranking inducers

An obvious way to use the accuracies predicted by regression is to build a ranking of the learners based on these predictions. In this section we give results for various ways of predicting rankings. We validate their usefulness by comparing them with the true ranking, and the performance of a default ranking.

To evaluate the different approaches, the rankings produced for a dataset are compared to the true ranking of the learners on this dataset. The true ranking is known since we know the accuracies of all the learners on the 65 datasets that we are using. As a measure of similarity of the rankings, we used Spearman's rank correlation coefficient (Neave & Worthington, 1992). We also compare our method with ranking via zooming (Soares & Brazdil, 2000).

Zooming cannot be applied to the full set of landmarkers, since that will mean using the performance of Lindiscr, IBL and NB to predict their ranking. This is why the corresponding combination, (zooming+land) is not included in the table. Also the results of ranking with zooming for the *statlog* and *histo* sets were not available. In the same table we give the average Spearman's rank correlation coefficient of the *default ranking* with the true ranking. The default ranking is a ranking that remains the same no matter what the dataset under examination is. It is computed on the basis of the mean accuracies that the learners achieve over all the datasets. The default ranking, starting from the best learner, is : c50boost, c50rules, c50tree, Ltree, ripper, IBL, NB, Lindiscr. A ranking method is interesting if it performs significantly better than this default ranking: in this case it is worth applying the meta-learning method to discover a suitable ranking for a given dataset.

One of the drawbacks of using the Spearman's correlation coefficient to evaluate rankings is the fact that it treats errors in a rank in the uniformly independently of whether they appear at the top or at the bottom of the rank. It is obvious that an error at the top of the rank is more important than an error in the bottom, since we are mainly interested in the top rated algorithms. To overcome that limitation of the Spearman's rank correlation coefficient we can focus only in the top position of the ranking and see how often the rankings predict correctly the top classification algorithm. The complete results with respect to that dimension of ranking evaluation are given in section 8.3.3. More elaborate evaluation measures of rankings exist based on modifications of the Spearman's correlation coefficient. In these modifications errors are penalized according to the position of the ranking list where they occur, (Soares et al., 2000).

The results in terms of the average Spearman rank correlation coefficient are given in table 8.8. Surprisingly enough in the top position we find the combination of Kernel with *dct* followed closely by Kernel with *histo, histo-limited* and *statlog*. What is interesting is that regression based ranking performs better than ranking with zooming, even though the latter is a method specifically designed to produce rankings. What is even more surprising is the poor performance of the landmarking based characterizations, although landmarking constructs regression models that have a very low MAD error, it fails to provide a good ranking of the classifiers. The predictions provided by Kernel and *dct*, while worse than the ones provided by landmarking based models, systematically keep the relative order of the accuracies of the classifiers. So although they do not estimate the performances accurately, they do rank the classifiers well. A reason for the poor performance of landmarking in ranking is that landmarking based regression models give the error as a function of the error of simple learners. This can lead to models where the error of an inducer is proportional to the error of another inducer resulting in a more or less fixed ranking of the available inducers, a fact that explains the poor performance of landmarkers when it comes to ranking inducers. Examining whether the differences are significant, after adjusting for the multiplicity effect on a 0.006 level, we see that none of the examined methods achieves a performance that can beat that of the

Table 8.9. P-values of paired t-tests, between the rank correlation coefficients of the models and the rank correlation coefficient of the default ranking.

	models	
Characterization	Kernel	Zooming
dct	=(0.050)	=(0.862)
histo	=(0.261)	
histo-limited	=(0.147)	=(0.482)
land	=(0.010)	
land-		=(0.018)
statlog	=(0.311)	

default ranking in a significant level, table 8.9.

To explain better the counter-intuitive bad performance of the landmarking approach with respect to ranking, we used Cubist (Quinlan, 2000), a regression algorithm that produces rule based models, and examine the structure of the constructed models. The reason for the use of Cubist is that the Kernel based regression does not produce models. The performance of *land* with Cubist is very similar to that with Kernel, both in terms of the quality of the error predictions for each inducer (very good predictions), and the ranking (the ranking correlation coefficients with the ideal ranking are much worse than the performance of the default ranking). In table 8.10 we give the regression models produced by Cubist for each one of the eight inducers. Each of the produced models gives the error of the corresponding inducer as a function of the error given by the landmarkers. Six out of the eight models are very simple linear regression equations. If we examine the equations that give the errors of c50boost and c50rules we can see that they are very similar, and they are based on the error of the IBL landmarker. The equations on the one hand give a very good prediction of the error of the two inducers, but on the same time always rank them in the same way for every dataset, with c50boost always predicted to have a smaller error than c50rules. This results in a ranking of the two inducers that is always the same irrespectively of the dataset under examination. The same situation holds also for other inducers, for example NB and ripper, whose errors are given as a function of the same landmarkers (i.e. Lindiscr and IBL). It is exactly this phenomenon, of a more or less fixed ranking between the inducers, that explains the poor performance of *land* in what concerns ranking.

Using regression to perform ranking is essentially different from ranking with zooming. Even when the regression model used is a kernel based one, whose idea is the same with that of nearest neighbors used in ranking with zooming, the similarities end there. In zooming, k-nearest neighbor is used to establish a set of similar datasets to the one under examination. In order to perform the ranking, the relative performances of the learners on the similar datasets are used. In the case of ranking through regression, we don't make use of the relative performance of the learners and there is no need to establish a set of similar datasets. Instead, the accuracies of the learners are directly predicted using the extracted characteristics of the dataset under examination. As a result of that we are not committed to any specific meta-learning model. More

Table 8.10. Models produced by Cubist on the *land* set of characteristics

Inducer	*Regression Model*
c50boost	error=-0.0127 + 0.888 IBL
c50rules	error=-0.0045 + 0.884 IBL
c50tree	error=-0.0134 + 0.595 IBL + 0.346 NB
Lindiscr	error= 0.0333 + 0.869 NB
Ltree	error=-0.025 + 0.407 IBL + 0.332 Lindiscr + 0.24 NB

IBL	IF Lindiscr <= 0.396 THEN
	error=0.0202 + 0.726 NB
	IF Lindiscr > 0.396
	error=-0.0835 + 0.82 Lindiscr + 0.34 Elite Node

NB	IF Lindiscr <= 0.237 THEN
	error=0.0211 + 0.645 IBL + 0.174 Lindiscr
	IF Lindiscr > 0.237 THEN
	error=0.0227 + 0.833 Lindiscr + 0.132 IBL

ripper	error=-0.0077 + 0.604 IBL + 0.377 Lindiscr

Table 8.11. Accuracy results on the inducer selection problem, via regression.

	Kernel		Zooming	
Characterization	Acc.	Impr.	Acc.	Impr.
dct	38.46%	6.16%	27.69%	-4.34%
histo	46.16%	13.86%		
histo-limited	36.92%	4.62%	29.24%	-3.06%
land	27.69%	-4.61%		
land-			23.08%	-9.22%
statlog	29.23%	-3.07%		

expressive models can be used that provide a better insight of how the dataset characteristics affect the performance of the learners, exactly like we do in the case of Cubist.

8.3.3 Selecting the best inducer

As already mentioned one of the drawbacks of evaluating rankings via the Spearman correlation coefficient is the uniform treatment of errors independently of their position in the ranking list. Here we will focus only in the top position of the rankings and we will examine how often they correctly predict the best classification algorithm. The meta-learning problem is exactly the same as the one described in section 7.3, the difference comes from the way we are trying to solve it. There we used classification methods, while here we are using regression methods, to solve what at the end, is a typical classification problem.

Obviously the class distribution is the one given in table 7.10 and the default accuracy is also the same. It is also the same meta-learning problem with the one defined in section 8.2. While there we were based on relative performance prediction of pairs of algorithms via regression to select the best algorithm, here we are relying on direct accuracy prediction.

We will examine the performance of Kernel combined with all the five possible ways of characterizing a dataset, plus the performance of zooming on the three sets on which the results are available. The evaluation was done using 10 fold cross validation. In table 8.11 we give the results on accuracy along with the improvement over the default accuracy. As a first fast insight into the results, we can see that only the Kernel models produce accuracy results that are better than the default for some of the sets. Models produced by zooming are all worse than the default.

Kernel gives results which are better than the default for the *dct*, *histo* and *histo−limited* sets. *histo* is again the set that achieves the highest improvement over the default, as it was also for the same problem when we tried to attack it via classification. Again the improvement over the default is not statistically significant.

Comparing the results of the regression based approach with the results obtained by the classification based approach, table 7.11, we can see that for all the characterizations regression produces worst results than direct classification. However since the differences are small we can not draw safe conclusions about the superiority of classification over regression.

8.4 Summary and Conclusions

In this chapter we examined an alternative approach to classification, for solving the meta-learning problem, which was based on regression. We explored two different meta-learning frameworks. The first one was based on the definition of pairwise problems and the second one on the direct prediction of the errors of inducers. In the later framework the regression models can be used either to rank inducers or simply to select the best inducer for a specific dataset. We examined the performance of five different dataset characterizations under both frameworks and investigated the use of regression to rank classification algorithms.

In what concerns the first framework the predictive accuracy with which the appropriate algorithm was chosen was relatively low, around 48% for the best characterization (the *histo* set). In total three characterizations overpassed the default accuracy related with the final suggestion but none at a statistically significant level.

For the direct error estimation the *land* set achieved by far the best performance. But when these estimates were used to rank the algorithms according to their expected performance it performed miserably, even significantly worse than the default ranking. The best results with ranking were obtained by the combination of Kernel and *dct*, followed by Kernel with *histo-limited* and *histo*.

The regression based ranking in general outperformed the ranking via zooming, nevertheless the differences between the methods, and between the methods and the performance of the default ranking was not statistically significant.

When we used the error estimates to select the best inducer only three of the methods managed to beat the default accuracy, but not in a statistically significant level. On the top position we found Kernel with *histo*, the same set that exhibited the highest performance when the problem of inducer selection was tackled via classification, section 7.3 or via the pairwise regression, section 8.2. Followed by Kernel with *dct* and *histo-limited*. The rest of the methods performed worse than the default accuracy. In general the regression based approach seems to perform slightly worse than the classification approach.

Chapter 9

Overview, Limitations and Future Work

The goal of this dissertation is to provide support to the analyst in selecting the most appropriate learning algorithm for a classification problem refraining from the tedious task of systematic experimentation with various learning algorithms. As a first step to that goal we relied on meta-learning, viewing inducer selection as a typical classification problem although at a meta-level. Within this approach our work spans the whole range of tasks required for the solution of a typical classification problem. That is, we searched for an appropriate formulation of the meta-learning space, and we constructed it in such a way so that it closely simulates the steps followed by the analyst when he has to select among different learners. Special care was also given to the feature extraction part of the process, in order to have a set of characteristics that can best discriminate between different datasets, a step which involved experimentation with different sets of features. We proceeded to a systematic experimentation of different learners on the meta-level and compared the set of characteristics that we established with sets of characteristics from previous similar work. In the last chapter of the dissertation we explored a different avenue that relied on regression techniques, compared different dataset characterizations under the regression scenario, and used the regression models not only to perform inducer selection but also inducer ranking.

The formulation of the meta-learning space allows for a close examination of the factors that affect the relative performance of specific pairs of inducers. The establishment though of a number of pairwise meta-learning problems poses a problem; an error of prediction in one of them could harm the final prediction. A possible solution could be a more intelligent way of combining the partial solutions to provide the final answer. This can be achieved via a meta-learning schema like stacking or global cascade generalization, where the predictions from the individual pairwise meta-learning problems will constitute a new learning problem in which the goal will be to predict the final classification algorithm based on the patterns of predictions from the pairwise problems.

The suggested formulation of the meta-learning space gave satisfactory results when applied to the set of semiartificial datasets. The performance in terms of the strict accuracy was, for the best meta-learner and the best dataset characterization, 51.16%, an improvement over the default strict accuracy of 24.93%, while for the loose accuracy it was 76.74%. When tested only on real datasets the performance was quite low, resulting in a strict accuracy of 15.38% and 20.00%, depending on which histogram based characterization was used, (corresponding improvements over the default accuracy 1.54% and 6.16%). The loose accuracy was 52.31% and 47.69%. A possible explanation for the poor performance in the real datasets is their limited number.

In defining the pairwise meta-learning problems special care was given in the appropriate definition of the datasets characterization. The idea of the histograms was introduced to describe in a finer detail the distributions of various properties of datasets whose number depends on the number of attributes of the dataset. We compared the histogram based characterizations with different dataset characterizations in varying meta-learning frameworks including one that was using regression to perform inducer selection. The histogram based approaches took the top positions whether inducer selection was performed in the pairwise meta-learning framework, the simple meta-learning framework, or via regression. We should note here that a set of dataset characteristics consisting mainly of the histogram characteristics, *histo-limited* exhibited quite good performance, with respect to the others, providing evidence that it is the use of histograms that improves the performance. Although the histogram based characterizations systematically outperformed the other characterizations, in almost all of the applied frameworks, the performance differences were not statistically significant. What was also disappointing was the fact that the difference between the histograms and the default accuracy was not statistically significant either. However the histogram based approaches were consistently better than the default accuracy, for all the different frameworks examined, even if the difference was not statistically significant. We cannot definitely support their superiority over the other methods of characterization, however since they are the only ones that systematically overpass the default accuracy and occupy the top positions we can at least argue that their presence can only improve performance in the inducer selection problem.

Before establishing the most appropriate histogram characterization we examined two different versions of feature sets with one of them trying also to capture and describe the associations between discrete and continuous attributes, using ideas from analysis of variance. Unfortunately that extended version of histograms did not bring any improvement in the performance and even more in one case it harmed the performance (when IBL was used on the meta-level). As a result we have chosen not to include these characteristics in the final set. This leaves a gap in the description of the datasets and of course obvious space for improvement. New characteristics should be devised that are able to describe in a satisfactory way the relations between continuous and discrete attributes. As a step to that we experimented with discretized versions of the datasets using the Fayyad and Irani (1995) method of discretization; this results on new versions of the datasets where all the attributes are discrete and characteristics that are appropriate for discrete attributes can be applied without a problem, unfortunately the results were not encouraging and we did not further explore that direction.

We also have to note that the concept of histograms was not applied to all the properties of the datasets that depend on the number of the attributes. For example histograms could be applied to describe the distribution of the mutual information between the attributes and the class or to the entropy of the attributes. Instead for the suite of experiments presented here we used only the means of these properties, mainly because they do not have a bounded interval of possible values, but an interval whose bounds depend on specific properties of the attributes. However with the proper normalization histograms could be also applied to describe the distributions of these characteristics too.

Another promising direction which could help improving the quality of predictions, although at the expense of loss of understandability of the produced models, is the application of principal components analysis on the set of dataset characteristics, so that the new set will contain uncorrelated features. This could prove beneficial especially for the case of the instance based learner on the meta-level.

We proceeded further in providing a characterization of the discriminating power of the individual characteristics by analyzing the models that the meta-learners produced. The characterization was based on how often a characteristic was selected to become part of a meta-learning model. The more often this was done, the higher the discriminatory power of the characteristic. The first and most important insight was the variation of the discriminatory power among different pairs of inducers. This observation provided further experimental support for the choice of the specific formulation of the meta-learning space. It is also an indication that meta-learning frameworks that rely on a single meta-learning problem will have a more difficult problem to solve. Although the pairwise meta-learning framework comes with the extra cost of the combination of the individual predictions, which is not a trivial task, the fact that allows for a focused study of the differences between inducers make us think that it is more appropriate. Returning to the discriminating power of the individual characteristics we believe that this is best reflected in the choices done by fsIBL on the set of 1075 datasets and in the similar choices done by c50boost in the 65 datasets. According to this characterization the simple characteristics, the STATLOG based ones, and the histograms describing the associations between the attributes, along with the first half of the histogram that describes the association with the class attribute are the most often selected, thus the most discriminating.

Although the final performance of the system depends heavily on the quality of the characteristics used to describe the datasets, we also tried to further improve the performance by examining the use of various inducers on the meta-learning level. The use of different inducers required some adaptation in the representation of their values due to the existence of the *non-applicable* values.

The best results were achieved by c50boost which was better than all the other inducers on the meta-learning level at a statistically significant level. The use of these inducers apart from being an effort to improve the performance, served also in the characterization of the discriminating power of the characteristics, as it was already described int the previous paragraph. It would be interesting to examine inducers that go beyond the propositional paradigm. The argument for that is the following: when we use a histogram to describe a property of a dataset we actually introduce a number of new features that describe this property; in some sense we place higher emphasis on that property since now it is described by more than one feature. To overcome this unbalanced representation a solution could be the use of learning algorithms that are able to cope with attributes whose values are lists or sets. For example consider a version of IBL where the attributes are lists or sets. What is needed in that case is a metric that can define the distance between two sets or two lists. In the case of lists, which is the one that better matches with the concept of histograms, the solution could be simply the euclidean distance between the two lists. The case of sets is not so straightforward because more elaborate metrics have to be used to define the distance between two sets, metrics that should of course respect the semantics of the problem. Whether we are dealing with attributes that are lists or sets the result of the comparison between two attributes will be a single quantity that can be incorporated naturally in the distance metric of the IBL algorithm. Similar alterations could be done to decision trees, here the splitting criterion should be altered so that it would be possible to define splits on attributes whose values are lists or sets. Again the use of a metric defined on lists or sets is essential. The use of sets is more appropriate when we would like to exploit the full set of characteristics corresponding to a dataset, without relying on the concept of histograms. For example for two datasets with k and l continuous attributes we would have two sets of $\binom{k}{2}$ and $\binom{l}{2}$ correlation coefficients respectively. If we were not to use histograms we should be able to compare the two sets in terms of their similarity. The suggestion provided above goes more into the direction of an inductive logic programming tool or a case based reasoning system, more generally a system that is able to handle multiple relations. Nevertheless the proposed system would lie between the two approaches and would be closer to the propositional framework.

A novel paradigm of representation that has risen in the statistical field can address successfully the representational issues set in the previous paragraph. Bock and Diday (2000) present the notion of *symbolic data types*. The main idea of symbolic data types is that *symbolic variables* can be defined which do not take a single value as it is done in propositional approaches. Symbolic variables may have as values: sets, intervals, and frequency or probability distributions. The two latter cases are identified as *modal variables*; a special case of modal variables are the *histogram variables*, where the distribution is typically given in the form of a histogram. The authors present various metrics that can be used in order to define distance between distributions, and extend that to vectors whose features are complete distributions. It should be noted here that although symbolic variables are typically multivalued, they are treated as a single entity, thus they do not increase overwhelmingly the dimensions of the search space, while on the same time they preserve a sufficient amount of the initial information. It is obvious that classification algorithms which are able to handle symbolic objects provide an ideal solution to the representational problem set forth by the description of datasets and they should be explored in view of meta-learning.

In a slightly different meta-learning scenario we used regression algorithms to perform inducer selection, and examined also different ways of dataset characterization. Here also the top performance was exhibited by a histogram based characterization. Compared to inducer selection via classification we noticed a slight decrease in performance. However the decrease was not significant and cannot provide clear evidence for the superiority of the classification based approach over regression for inducer selection.

The regression models produced were also used to provide rankings of algorithms. We compared the ranking performance of the different characterizations and also compared the regression approach to ranking with zooming based ranking. There were two main outcomes of this empirical comparison. The first one was that regression based ranking was found to perform better than zooming based one, even though the later is specifically designed with the problem of ranking in mind, the difference however was not statistically significant. The second one was the fact that while the landmarking set of characteristics provided the most reliable error estimates, it failed miserably with respect to ranking. The explanation to that lay in the form of the regression models that were constructed from the landmarking characterization.

Another point that is worth of further investigation is the population of the DSs set used to train the system. The approach that we have followed was based on the careful modification of some of the characteristics of an initial set of datasets to various dimensions. An alternative approach could be the use of completely artificial datasets, where the initial concepts described by the instances of a dataset are known in advance. In that way not only we will know the optimal error for every artificial dataset, but we will also be able to compute the exact error of every classification algorithm without having to rely on a resampling procedure, since we will have in our disposition as many instances as we want. This will make meta-learning datasets more dense, and improve their quality. It is even possible to use real world datasets as the starting point for the artificial datasets. In order to achieve that we can produce a classification model from a real world dataset and use that model as the starting point for the construction of artificial datasets that will populate the dataset space around the initial real world dataset.

The problem with which we tried to cope in this dissertation is the selection of the most appropriate inducer for a specific dataset in the lack of any relevant information apart from the dataset itself. Within this framework we took special care in choosing an appropriate formulation for the meta-learning problem and gave special attention in the definition of an appropriate set of characteristics. The whole approach gave satisfactory results when it was tested on a pool of semiartificial datasets. The results were not satisfactory when the tests were repeated on a limited number of real world datasets. However they provided evidence that the incorporation of histogram based descriptions of the properties of the dataset can help in solving the problem of inducer selection.

We believe that the main focus of future work should be in the refinement of the descriptions of the datasets, a quite difficult problem as it is by now obvious from the results achieved within this work. However we should keep in mind that all the information we need is there; a dataset is the ultimate–most detailed– description of itself, what we are looking for is an intelligent way to compress–describe it.

Bibliography

Agresti, A. (1990). *Categorical data analysis.* John Wiley and Sons: Series in Probability and Mathematical Statistics.

Aha, D. (1992). Generalizing from case studies: A case study. In D. Sleeman and P. Edwards (Eds.), *Proceedings of the 9th International Machine Learning Conference* (pp. 1–10). Morgan Kaufman.

Anderson, T. (1984). *An introduction to multivariate statistical analysis*, chapter Canonical Correlations and Canonical Variables, 480–520. John Wiley and Sons.

Bailey, T., & Elkan, C. (1993). Estimating the accuracy of learned concepts. In R. Bajcsy (Ed.), *Proceedings of the 13th International Joint Conference on Artificial Intelligence* (pp. 895–900). Morgan Kaufman.

Bensusan, H., & Giraud-Carrier, C. (2000). Discovering task neighbourhoods through landmark learning performances. In D. Zighed, J. Komorowski and J. Zytkow (Eds.), *Proceedings of the 4th European Conference, on Principles of Data Mining and Knowledge Discovery* (pp. 325–330). Springer.

Bensusan, H., & Kalousis, A. (2001). Estimating the predictive accuracy of a classifier. In L. Raedt and P. Flach (Eds.), *Proceedings of the 12th European Conference on Machine Learning* (pp. 25–36). Springer.

Blake, C., Keogh, E., & Merz, C. (1998). http://www.ics.uci.edu / ∼mlearn / mlrepository.html. . University of California, Irvine, Dept. of Information and Computer Sciences.

Blockeel, H., & Struyf, J. (2001). Efficient algorithms for decision tree cross-validation. In C. Broodley and A. Danyluk (Eds.), *Proceedings of the 18th International Conference on Machine Learning* (pp. 11–18). Morgan Kaufman.

Bloedorn, E., & Michalski, R. (1998). Data driven constructive induction. *IEEE Intelligent Systems*, *13*, 30–37.

Bloedorn, E., Michalski, R., & Wnek, J. (1993). Multistrategy constructive induction: Aq17-mci. In R. Michalski and G. Tecuci (Eds.), *Proceedings of the 2nd International Workshop on Multistrategy Learning* (pp. 188–202).

Bloedorn, E., Michalski, R., & Wnek, J. (1994). *Matching methods with problems: A comparative analysis of constructive induction approaches* (Technical Report MLI94-12). Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA.

Bock, H., & Diday, E. (2000). *Analysis of symbolic data.* Springer.

Brazdil, P., Gama, J., & Henery, R. (1994). Characterizing the applicability of classification algorithms using meta level learning. In F. Bergadano and L. Raedt (Eds.), *Proceedings of the European Conference on Machine Learning* (pp. 83–102). Springer Verlag.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*, 123–140.

Breiman, L., Friedman, J. H., Olshen, R., & Stone, C. (1984). *Classification and regression trees.* CRC Press.

Brodley, C. (1995). Recursive automatic bias selection for classifier construction. *Machine Learning*, *20*, 63–94.

Brodley, C., & Smyth, P. (1997). Applying classification algorithms in practice. *Statistics and Computing*, *7*, 63–94.

Cohen, W. (1995). Fast effective rule induction. In A. Prieditis and S. Russell (Eds.), *Proceedings of the 12th International Conference on Machine Learning* (pp. 115–123). Morgan Kaufman.

Cooper, G., & Herskovits, E. (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, *9*, 309–347.

Dietterich, T. (1998). Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, *10*, 1895–1024.

Domingos, P. (1996). Unifying instance-based and rule-based induction. *Machine Leanring*, *24*, 141–168.

Duda, R., & Hart, P. (1973). *Pattern classification and scene analysis*, chapter 3, 66–73. John Willey and Sons.

Efron, B., & Tibshirani, R. (1995). *Cross-validation and the boostrap: Estimating the error rate of a prediction rule* (Technical Report TR-477). Departement of Statistics, Standford University.

Engels, R., & Theusinger, C. (1998). Using a data metric for offering preprocessing advice in data mining applications. In H. Prade (Ed.), *Proceedings of the 13th European Conference on Artificial Intelligence* (pp. 430–434). IOS Press.

Fayyad, U., & Irani, K. (1995). Multi–interval discretization of continuous attributes as preprocessing for classification learning. In R. Bajcsy (Ed.), *Proceedings of the 13th International Joint Conference on Artificial Intelligence* (pp. 1022–1027). Morgan Kaufmann.

Feelders, A., & Verkooijen, W. (1995). Which method learns most from the data. In *Proceedings of the 5th International Workshop on AI and Statistics* (pp. 219–225).

Fisher, R. A. (1936). The use of multiple measurments in taxonomic problems of biological classification. *Annals of Eugenics*, *7*, 179–188.

Freund, Y., & Schapire, R. (1996). Experiments with a new boosting algorithm. In L. Saitta (Ed.), *Proceedings of the 13th International Conference on Machine Learning*. Morgan Kaufman.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition*, chapter Feature Extraction and Linear Mapping for Classification, 441–507. Academic Press.

Gama, J., & Brazdil, P. (1995). Characterization of classification algorithms. In C. Pinto-Ferreira and N. Mamede (Eds.), *Proceedings of the 7th Portugese Conference in AI, EPIA 95* (pp. 83–102). Springer-Verlag.

Gama, J., & Brazdil, P. (1999). Linear tree. *Intelligent Data Analysis*, *3*, 1–22.

Gama, J., & Brazdil, P. (2000). Cascade generalization. *Machine Learning*, *4*, 315–343.

Gordon, F., & desJardin, M. (1995). Evaluation and selection of biases. *Machine Learning*, *20*, 5–22.

Hilario, M., & Kalousis, A. (2000). Quantifying the resilience of inductive algorithms for classification. In D. Zighed, J. Komorowski and J. Zytkow (Eds.), *Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases* (pp. 106–115). Springer.

Hilario, M., & Kalousis, A. (2001). Fusion of meta-knowledge and meta-data for case-based model selection. In L. Raedt and A. Siebes (Eds.), *Proceedings of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Springer.

Huang, Y. S., & Suen, C. Y. (1995). A method for combining multiple experts for the recognition of unconstrained handwritten numeral. *IEEE Transactions On Pattern Analysis and Machine Intelligence*, *17*.

Kalousis, A., & Hilario, M. (2000a). Knowledge discovery from incomplete data. In *Proceedings of the 2nd International Conference on Data Mining*. WIT Press.

Kalousis, A., & Hilario, M. (2000b). Model selection via meta-learning: a comparative study. In *Proceedings of the 12th International IEEE Conference on Tools with AI*.

Kalousis, A., & Hilario, M. (2001). Feature selection for meta-learning. In D. Cheung, G. Williams and Q. Li (Eds.), *Proceedings of the 5th Pasific Asia Conference on Knowledge Discovery and Data Mining*. Springer.

Koepf, C., Taylor, C., & Keller, J. (2000). Meta-analysis: from data characterisation for meta-learning to meta-regression. In P. Brazdil and A. Jorge (Eds.), *Proceedings of the PKDD-2000 Workshop on Data Mining, Decision Support, Meta-Learning and ILP*.

Kohavi, R. (1995). A study of cv and bootstrap for accuracy estimation and model selection. In C. Mellish (Ed.), *Proceedings of the 14th International Joint Conference on AI*. Morgan Kaufman.

Kohavi, R. (1996). Scaling up the accuracy of naive-bayes classifiers: a decision tree hybrid. In E. Simoudis, J. Han and U. Fayyad (Eds.), *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining* (pp. 202–207). AAAI press.

Kohavi, R., & John, G. (1997). Wrappers for feature subset selection. *Artificial Intelligence Journal*, *97*, 273–324.

Kohavi, R., Sommerfield, D., & Dougherty, J. (1996). Data mining using mlc++, a machine learning library in c++. In *Tools With AI*.

Koppel, M., & Engelson, S. (1996). Integrating multiple classifiers by finding their areas of expertise. In *Proceedings of AAAI-96, Workshop on Integrating Multiple Learned Models*.

Lindner, C., & Studer, R. (1999). Ast: Support for algorithm selection with a cbr approach. In *Proceedings of the 16th International Conference on Machine Learning, Workshop on Recent Advances in Meta-Learning and Future Work*.

Liu, H., & Motoda, H. (1998). *Feature selection for knowledge discovery and data mining*. Kluwer Academic Publishers.

Merz, C. J. (1995). *Learning from data: Artificial intelligence and statistics*, chapter Dynamical Selection of Learning Algorithms. Springer Verlag.

Michie, D., Spiegelhalter, D., & Taylor, C. (1994). *Machine learning, neural and statistical classification*. Ellis Horwood Series in Artificial Intelligence.

Mitchell, T. (1997). *Machine learning*. MacGraw Hill.

Nakhaeizadeh, G., & Schnabl, A. (1997). Development of multi-criteria metrics for evaluation of data mining algorithms. In D. Heckerman, H. Mannila, D. Pregibon and R. Uthurusamy (Eds.), *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining* (pp. 37–42). AAAI Press.

Neave, H., & Worthington, P. (1992). *Distribution free tests*. Unwin Hyman, London, UK.

Oates, T., & Jensen, D. (1998). Large datasets lead to overly complex models: An explanation and a solution. In R. Agrawal, P. Stolorz and G. Piatetsky-Shapiro (Eds.), *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining,* (pp. 294–298).

Paterson, I., Berrer, H., & Keller, J. (2001). The focused multi-criteria ranking approach to machine learning algorithm selection- an incremental meta learning assistant for data mining tasks. In C. Giraud-Carrier, N. Lavrac, S. Moyle and B. Kavsek (Eds.), *Proceedings of the ECML/PKDD2001 Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning.*

Pfahringer, B., Bensusan, H., & Giraud-Carrier, C. (2000). Tell me who can learn you and i can tell you who you are: Landmarking various learning algorithms. In P. Langley (Ed.), *Proceedings of the 17th International Conference on Machine Learning* (pp. 743–750). Morgan Kaufman.

Quinlan, J. (1992a). *c4.5 : Programs for machine learning.* Morgan Kaufman Publishers.

Quinlan, J. (1992b). Learning with continuous classes. In A. Adams and L. Sterling (Eds.), *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence* (pp. 343–348). World Scientific.

Quinlan, J. (2000). *An overview of cubist.* Rulequest Research, http://www.rulequest.com/.

Rao, C. R. (1948). The utilization of multiple measurments in problems of biological classification. *Journal of the Royal Statistical Society*, 159–203.

Rendell, L., Seshu, R., & Tcheng, D. (1987). Layered concept learning and dynamically variable bias management. In J. McDermott (Ed.), *Proceedings of the 10th International Joint Conference on AI* (pp. 366–372). Morgan Kaufman.

Ripley, B. (1996). *Pattern recognition and neural networks*, chapter Linear Discriminant Analysis, 91–120. Cambridge University Press.

Ruggieri, S. (2002). Efficient c4.5. *IEEE Transactions On Knowledge and Data Engineering*, *14*, 438–444.

Salzberg, L. S. (1991). A nearest hypperectangular learning method. *Machine Learning*, *6*, 251–276.

Salzberg, L. S. (1997). On comparing classifiers: A critique of current research and methods. *Data Mining and Knowledge Discovery*, *1*, 317–327.

Schafer, J. (1997). *Analysis of incomplete multivariate data.* CRC Press.

Schaffer, C. (1993). Selecting a classification method by cross validation. *Machine Learning*, *13*, 135–143.

Schaffer, C. (1994). A conservation law for generalization performance. In W. Cohen and H. Hirsh (Eds.), *Proceedings of the 11th International Conference on Machine Learning*. Morgan Kaufman.

Schapire, R. (1990). The strength of weak learnability. *Machine Learning*, *5*, 192–227.

Shavlik, J. W., Mooney, R., & Towell, G. (1991). Symbolic and neural net learning algorithms : An experimental approach. *Machine Learning*, *6*, 111–114.

Sleeman, D., Rissakis, M., Craw, S., Graner, N., & Sharma, S. (1995). Consultant-2: Pre and post-processing of machine learning applications. *International Journal of Human Computer Studies*, *43*, 43–63.

Soares, C. (2000). Ranking classification algorithms on past performance. Master's thesis, Faculty Of Economics, University of Porto.

Soares, C., & Brazdil, P. (2000). Zoomed ranking: Selection of classification algrorithms based on relevant performance information. In D. Zighed, J. Komorowski and J. Zytkow (Eds.), *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery* (pp. 126–135). Springer.

Soares, C., Brazdil, P., & Costa, J. (2000). Measures to compare rankings of classification algorithms. In H. Kiers, J. Rasson, P. Groenen and M. Schader (Eds.), *Data Analysis, Classification and Related Methods, Proceedings of the 7th Conference of the International Federation of Classification Societies* (pp. 119–124). Springer.

Sohn, S. Y. (1999). Meta analysis of classification algorithms for pattern recognition. *IEEE Transcactions on Pattern Analysis and Machine Intelligence*, *21*, 1137–1144.

Tcheng, D., Lambert, B., Lu, S., & Rendell, L. (1989). Building robust learning systems by combining induction and optimization. In *Proceedings of the 11th International Joint Conference on AI* (pp. 806–812). Morgan Kaufman.

Ting, K. M. (1994). The problem of small disjuncts: its remedy in decision trees. In R. Elio (Ed.), *Proceedings of the 10th Canadian Conference on Artificial Intelligence* (pp. 91–97). Canadian Society for Computational Studies of Intelligence.

Ting, K. M. (1997). Decision combination based on the characterization of predictive accuracy. *Intelligent Data Analysis*, *1*.

Ting, K. M., & Witten, I. (1997). Stacked generalization: when does it work. In *Proceedings of the 15th International Joint Conference on AI* (pp. 866–871). Morgan Kaufman.

Todorovski, L., Brazdil, P., & Soares, C. (2000). Report on the experiments with feature selection in meta-level learning. In P. Brazdil and A. Jorge (Eds.), *Proceedings of the 4th European Conference on Principles on Data Mining and Knowledge Discovery, Workshop on Data Mining, Decision Support, Meta-learning and ILP* (pp. 27–39).

Todorovski, L., & Dzeroski, S. (1999). Experiments in meta-level learning with ilp. In J. Zytkow and J. Rauch (Eds.), *Proceedings of the 3rd European Conference on Principles of Data Mining and Knowledge Discovery* (pp. 98–106). Springer.

Todorovski, L., & Dzeroski, S. (2000). Combining multiple models with meta decision trees. In D. Zighed, J. Komorowski and J. Zytkow (Eds.), *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery* (pp. 55–64). Springer.

Torgo, L. (1999). *Inductive learning of tree-based regression models.* Doctoral dissertation, University Of Porto, Faculty of Sciences.

Weis, S., & Kapouleas, I. (1989). An empirical comparison of patter recognition, neural nets, and machine learning classification algorithms. In *Proceedings of the 11th International Joint Conference on AI* (pp. 781–787). Morgan Kaufman.

Wnek, J., & Michalski, R. (1994). Hypothesis driven constructive induction in aq17-hci: A method and experiments. *Machine Learning*, *14*, 139–168.

Wolpert, D. (1996a). The existense of a priori distinctions between learning algorithms. *Neural Computation*, *8*, 1391–1420.

Wolpert, D. (1996b). The lack of a priori distinctions between learning algorithms. *Neural Computation*, *8*, 1341–1390.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, *5*, 241–259.

Woods, K. (1997). Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*, 405–410.

Appendix A

Results on the 1075 datasets

A.1 The 47 initial datasets

abalone, acetylation, ann-thyroid, australian, balance-scale, bupa, byzantine, car, char, clean1, clean2, flag_language, flag_religion, flare_c, flare_c_er, flare_m, flare_m_er, flare_x, flare_x_er, german, glass, glass2, heart, ionosphere, iris, lenses, lymphography, monk1, monk2, monk3, new-thyroid, nursery, optdigits, page-blocks, parity5_5, pendigits, pima-indians-diabetes, sat, segmentation, sonar, soybean-small, titanic, vehicle, vote, waveform_21, wdbc, yeast.

A.2 Characteristics Selected by fsIBL

Table A.1. Characteristics Selected by fsIBL on the 1075 datasets for the pairs : (c50rules c50boost), (c50tree c50boost), (c50tree c50rules)

Attribute	c50rules c50boost	c50tree c50boost	c50tree c50rules
# classes	1	1	0
# attributes	0	0	0
# instances	1	1	1
$\frac{\text{\# attributes}}{\text{\#instances}}$	0	1	0
# unknown values	0	0	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0	1	0
# nominal attributes	1	0	1
max,min,mean,stdv of nominal attribute values	1011	1110	1011
1..10 concentration histogram	0011011111	1010011000	1110000000
non computable conc. histogram	1	0	0
1..10 concentration histogram with class	0101000000	1101000000	1001000000
non computable conc. histogram with class	0	0	0
# continuous attributes	1	1	0
1..10 correlation histogram	1101010011	1110111101	1100001101
non computable correlation histogram	1	0	0
1..10 missing values histogram	0	0	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	1	0	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	1	0	0
Binary Attributes	0	0	1
Frac1	0	0	0
First Canonical Correlation	0	1	0
Mean Skew	0	1	0
Mean Kurtosis	0	0	1
Class Entropy	1	0	1
Mean Attribute Entropy	0	1	1
Mean Mutual Information	1	1	1
Equivalent number of attributes	1	1	1
Noise to Signal Ratio	1	1	1
Mean Mult. Correl. Coef.	1	0	0
SDratio	1	1	1

Table A.2. (Lindiscr c50boost), (Lindiscr c50rules), (Lindiscr c50tree)

Attribute	Lindiscr c50boost	Lindiscr c50rules	Lindiscr c50tree
# classes	1	1	1
# attributes	0	1	0
# instances	1	0	0
$\frac{\text{\# attributes}}{\text{\#instances}}$	1	1	1
# unknown values	0	0	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	1	1	1
# nominal attributes	1	1	0
max,min,mean,stdv of nominal attribute values	1001	1000	0011
1..10 concentration histogram	1010000000	1011100000	0010100000
non computable conc. histogram	0	0	0
1..10 concentration histogram with class	0001000000	0001000000	1000000000
non computable conc. histogram with class	0	0	0
# continuous attributes	0	0	0
1..10 correlation histogram	1000110000	0000111000	1111010000
non computable correlation histogram	0	0	0
1..10 missing values histogram	0	0	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0	0	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0	0	0
Binary Attributes	0	1	1
Frac1	0	0	0
First Canonical Correlation	1	0	0
Mean Skew	0	0	0
Mean Kurtosis	0	0	0
Class Entropy	1	1	0
Mean Attribute Entropy	1	1	0
Mean Mutual Information	0	1	0
Equivalent number of attributes	1	1	1
Noise to Signal Ratio	1	1	1
Mean Mult. Correl. Coef.	1	0	1
SDratio	0	0	1

Table A.3. (Ltree c50boost), (Ltree c50rules), (Ltree c50tree)

Attribute	Ltree c50boost	Ltree c50rules	Ltree c50tree
# classes	1	0	1
# attributes	0	0	1
# instances	0	1	1
$\frac{\text{\# attributes}}{\text{\#instances}}$	1	1	1
# unknown values	0	1	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	1	1	1
# nominal attributes	0	1	0
max,min,mean,stdv of nominal attribute values	1101	1111	1000
1..10 concentration histogram	1100111111	1111010000	0011100000
non computable conc. histogram	1	0	0
1..10 concentration histogram with class	0110100000	1001000000	0000000000
non computable conc. histogram with class	0	0	0
# continuous attributes	0	1	1
1..10 correlation histogram	0101111111	1101110110	0111110000
non computable correlation histogram	0	0	0
1..10 missing values histogram	0	0	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0	0	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0	0	0
Binary Attributes	0	0	1
Frac1	1	1	1
First Canonical Correlation	1	0	1
Mean Skew	0	1	1
Mean Kurtosis	0	0	0
Class Entropy	0	1	1
Mean Attribute Entropy	1	1	1
Mean Mutual Information	1	1	1
Equivalent number of attributes	1	1	1
Noise to Signal Ratio	1	0	1
Mean Mult. Correl. Coef.	1	0	0
SDratio	1	1	1

Table A.4. (Ltree Lindiscr), (IBL c50boost), (IBL c50rules)

Attribute	Ltree Lindiscr	IBL c50boost	IBL c50rules
# classes	1	0	0
# attributes	1	1	0
# instances	0	0	0
$\frac{\text{\# attributes}}{\text{\#instances}}$	1	1	1
# unknown values	0	1	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0	0	0
# nominal attributes	1	1	0
max,min,mean,stdv of nominal attribute values	1001	0011	0000
1..10 concentration histogram	1101100000	1001100000	1000000000
non computable conc. histogram	0	0	0
1..10 concentration histogram with class	0001000000	0101000000	0000000000
non computable conc. histogram with class	0	0	0
# continuous attributes	0	1	1
1..10 correlation histogram	0111001111	0111111000	1010010101
non computable correlation histogram	0	0	0
1..10 missing values histogram	0	0	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0	0	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0	0	0
Binary Attributes	0	0	0
Frac1	0	1	0
First Canonical Correlation	1	0	1
Mean Skew	1	1	1
Mean Kurtosis	0	1	0
Class Entropy	0	0	1
Mean Attribute Entropy	1	0	1
Mean Mutual Information	1	0	1
Equivalent number of attributes	1	0	1
Noise to Signal Ratio	0	1	1
Mean Mult. Correl. Coef.	1	0	0
SDratio	1	0	1

Table A.5. (IBL c50tree), (IBL Lindiscr), (IBL Ltree)

Attribute	IBL c50tree	IBL Lindiscr	IBL Ltree
# classes	1	1	1
# attributes	1	1	0
# instances	0	0	0
$\frac{\text{\# attributes}}{\text{\#instances}}$	1	1	1
# unknown values	1	0	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0	1	1
# nominal attributes	1	0	1
max,min,mean,stdv of nominal attribute values	0011	0001	1011
1..10 concentration histogram	1000000000	1010100000	0000000000
non computable conc. histogram	0	0	0
1..10 concentration histogram with class	1100000000	0101000000	0101000000
non computable conc. histogram with class	0	0	0
# continuous attributes	1	0	0
1..10 correlation histogram	1110100101	0110101011	1111111101
non computable correlation histogram	0	0	0
1..10 missing values histogram	0	0	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0	0	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0	0	0
Binary Attributes	0	0	1
Frac1	0	1	0
First Canonical Correlation	1	1	1
Mean Skew	1	1	1
Mean Kurtosis	1	1	1
Class Entropy	0	1	0
Mean Attribute Entropy	1	1	1
Mean Mutual Information	0	1	1
Equivalent number of attributes	1	0	0
Noise to Signal Ratio	1	1	1
Mean Mult. Correl. Coef.	1	0	1
SDratio	1	0	1

Table A.6. (NB c50boost), (NB c50rules), (NB c50tree)

Attribute	NB c50boost	NB c50rules	NB c50tree
# classes	1	1	1
# attributes	0	1	1
# instances	1	1	0
$\frac{\# \text{ attributes}}{\#\text{instances}}$	0	1	1
# unknown values	0	0	0
$\frac{\# \text{ unknown values}}{\# \text{ attributes} * \# \text{ instances}}$	1	0	0
# nominal attributes	1	0	1
max,min,mean,stdv of nominal attribute values	1010	0011	0010
1..10 concentration histogram	1010111111	1100000000	1100000000
non computable conc. histogram	1	0	0
1..10 concentration histogram with class	1101000000	1000000000	1000000000
non computable conc. histogram with class	0	0	0
# continuous attributes	1	0	0
1..10 correlation histogram	11110111111	01000011100	01111110010
non computable correlation histogram	1	0	0
1..10 missing values histogram	0	0	0
$\frac{\# \text{ continuous}}{\# \text{ attributes}}$	0	0	0
$\frac{\# \text{ nominal}}{\# \text{ attributes}}$	1	0	0
Binary Attributes	1	1	0
Frac1	0	0	0
First Canonical Correlation	1	1	1
Mean Skew	1	1	1
Mean Kurtosis	1	0	1
Class Entropy	1	1	0
Mean Attribute Entropy	0	1	1
Mean Mutual Information	1	1	1
Equivalent number of attributes	1	1	1
Noise to Signal Ratio	1	1	1
Mean Mult. Correl. Coef.	1	1	1
SDratio	0	1	1

Table A.7. (NB Lindiscr), (NB Ltree), (NB IBL)

Attribute	NB Lindiscr	NB Ltree	NB IBL
# classes	1	1	1
# attributes	1	1	1
# instances	0	1	1
$\frac{\text{\# attributes}}{\text{\#instances}}$	1	1	1
# unknown values	0	1	1
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	1	0	1
# nominal attributes	1	1	1
max,min,mean,stdv of nominal attribute values	0100	0111	1101
1..10 concentration histogram	1111100000	0111111111	1101110000
non computable conc. histogram	0	1	0
1..10 concentration histogram with class	1101000000	1111111100	1001000000
non computable conc. histogram with class	0	0	0
# continuous attributes	0	1	1
1..10 correlation histogram	0011000000	1110011110	1111111001
non computable correlation histogram	0	1	0
1..10 missing values histogram	0	0	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0	1	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0	1	0
Binary Attributes	0	1	0
Frac1	0	0	1
First Canonical Correlation	1	1	1
Mean Skew	0	1	1
Mean Kurtosis	0	1	1
Class Entropy	1	1	0
Mean Attribute Entropy	1	1	1
Mean Mutual Information	1	0	1
Equivalent number of attributes	1	1	0
Noise to Signal Ratio	1	1	1
Mean Mult. Correl. Coef.	1	1	1
SDratio	0	1	0

Table A.8. (ripper c50boost), (ripper c50rules), (ripper c50tree)

Attribute	ripper c50boost	ripper c50rules	ripper c50tree
# classes	1	1	1
# attributes	0	1	1
# instances	0	0	0
$\frac{\text{\# attributes}}{\text{\#instances}}$	1	1	0
# unknown values	1	0	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0	0	1
# nominal attributes	0	1	1
max,min,mean,stdv of nominal attribute values	1011	0010	1010
1..10 concentration histogram	1111000000	1101100000	0100000000
non computable conc. histogram	0	0	0
1..10 concentration histogram with class	1101000000	0101000000	1101000000
non computable conc. histogram with class	0	0	0
# continuous attributes	1	0	0
1..10 correlation histogram	0101101011	1100010000	1010110100
non computable correlation histogram	0	0	0
1..10 missing values histogram	0	0	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0	0	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0	0	0
Binary Attributes	0	0	0
Frac1	1	0	0
First Canonical Correlation	1	1	0
Mean Skew	0	0	1
Mean Kurtosis	0	1	1
Class Entropy	1	0	0
Mean Attribute Entropy	1	1	0
Mean Mutual Information	0	1	1
Equivalent number of attributes	1	0	1
Noise to Signal Ratio	1	1	1
Mean Mult. Correl. Coef.	1	1	1
SDratio	0	1	1

Table A.9. (ripper Lindiscr), (ripper Ltree), (ripper IBL)

Attribute	ripper Lindiscr	ripper Ltree	ripper IBL
# classes	1	1	0
# attributes	1	1	1
# instances	1	1	0
$\frac{\text{\# attributes}}{\text{\#instances}}$	1	1	1
# unknown values	1	0	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	1	1	1
# nominal attributes	1	1	1
max,min,mean,stdv of nominal attribute values	1111	0000	0011
1..10 concentration histogram	1101111000	1100011111	1100000000
non computable conc. histogram	0	1	0
1..10 concentration histogram with class	0101000000	0101000000	1101000000
non computable conc. histogram with class	0	0	0
# continuous attributes	1	1	1
1..10 correlation histogram	00101111000	11011111111	10100110010
non computable correlation histogram	0	1	0
1..10 missing values histogram	0	0	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0	0	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0	1	0
Binary Attributes	0	1	0
Frac1	1	0	0
First Canonical Correlation	1	0	0
Mean Skew	1	1	0
Mean Kurtosis	1	1	1
Class Entropy	1	0	0
Mean Attribute Entropy	1	1	0
Mean Mutual Information	0	1	1
Equivalent number of attributes	1	1	0
Noise to Signal Ratio	0	1	1
Mean Mult. Correl. Coef.	0	1	1
SDratio	0	1	0

Table A.10. (ripper NB)

Attribute	ripper NB
# classes	1
# attributes	1
# instances	1
$\frac{\text{\# attributes}}{\text{\#instances}}$	1
# unknown values	0
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	1
# nominal attributes	1
max,min,mean,stdv of nominal attribute values	1011
1..10 concentration histogram	1100111111
non computable conc. histogram	1
1..10 concentration histogram with class	0001000000
non computable conc. histogram with class	0
# continuous attributes	1
1..10 correlation histogram	10100110010
non computable correlation histogram	0
1..10 missing values histogram	0
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0
Binary Attributes	0
Frac1	0
First Canonical Correlation	0
Mean Skew	1
Mean Kurtosis	0
Class Entropy	0
Mean Attribute Entropy	1
Mean Mutual Information	1
Equivalent number of attributes	0
Noise to Signal Ratio	1
Mean Mult. Correl. Coef.	1
SDratio	1

A.3 Selection Frequency by c50boost

Table A.11. Selection Frequency of characteristics by c50boost on the 1075 datasets, pair : c50rules c50boost

Attribute	
# classes	0.0216
# attributes	0.0266
# instances	0.0305
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0241
# unknown values	0.0355
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0355
# nominal attributes	0.0114
max, min, mean, stdv of nominal attribute values	0.0102 0.0063 0.0102 0.0089
1..5 concentration histogram	0.0102 0.0127 0.0102 0.0127 0.0076
6..10 concentration histogram	0.0038 0.0051 0.0000 0.0000 0.0076
non computable conc. histogram	0.0076
1..5 concentration histogram with class	0.0254 0.0076 0.0038 0.0178 0.0051
6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.0000 0.0013
non computable conc. histogram with class	0.0013
# continuous attributes	0.0076
1..5 correlation histogram 6..10 correlation histogram	0.0279 0.0178 0.0203 0.0152 0.0127 0.0076 0.0190 0.0152 0.0228 0.0190
non computable correlation histogram	0.0114
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0343 0.0343 0.0241 0.0266 0.0254 0.0190 0.0013 0.0114 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0025
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0013
Binary Attributes	0.0063
Frac1	0.0140
First Canonical Correlation	0.0178
Mean Skew	0.0165
Mean Kurtosis	0.0178
Class Entropy	0.0343
Mean Attribute Entropy	0.0368
Mean Mutual Information	0.0228
Equivalent number of attributes	0.0343
Noise to Signal Ratio	0.0368
Mean Mult. Correl. Coef.	0.0152
SDratio	0.0102

Table A.12. c50tree c50boost

Attribute	
# classes	0.0300
# attributes	0.0185
# instances	0.0185
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0092
# unknown values	0.0554
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0254
# nominal attributes	0.0058
max,min,mean,stdv of nominal attribute values	0.0104 0.0046 0.0150 0.0012
1..5 concentration histogram 6..10 concentration histogram	0.0115 0.0161 0.0150 0.0069 0.0092 0.0081 0.0058 0.0012 0.0000 0.0092

non computable conc. histogram
hline 1..5 concentration histogram with class
6..10 concentration histogram with class
	0.0127
	0.0208 0.0046 0.0058 0.0231 0.0058
	0.0035 0.0012 0.0000 0.0000 0.0023

non computable conc. histogram with class	0.0000
# continuous attributes	0.0161
1..5 correlation histogram 6..10 correlation histogram	0.0196 0.0185 0.0265 0.0242 0.0138 0.0161 0.0081 0.0115 0.0173 0.0104
non computable correlation histogram	0.0161
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0300 0.0438 0.0381 0.0288 0.0208 0.0208 0.0035 0.0208 0.0012
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0023
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0023
Binary Attributes	0.0058
Frac1	0.0173
First Canonical Correlation	0.0161
Mean Skew	0.0081
Mean Kurtosis	0.0081
Class Entropy	0.0450
Mean Attribute Entropy	0.0473
Mean Mutual Information	0.0208
Equivalent number of attributes	0.0219
Noise to Signal Ratio	0.0150
Mean Mult. Correl. Coef.	0.0138
SDratio	0.0138

Table A.13. c50tree c50rules

Attribute	
# classes	0.0375
# attributes	0.0250
# instances	0.0312
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0250
# unknown values	0.0413
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0325
# nominal attributes	0.0138
max, min, mean, stdv of nominal attribute values	0.0138 0.0075 0.0037 0.0088
1..5 concentration histogram 6..10 concentration histogram	0.0125 0.0138 0.0050 0.0063 0.0050 0.0037 0.0013 0.0000 0.0000 0.0063
non computable conc. histogram	0.0138
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0063 0.0112 0.0013 0.0175 0.0100 0.0013 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0163
1..5 correlation histogram 6..10 correlation histogram	0.0275 0.0163 0.0213 0.0187 0.0187 0.0325 0.0100 0.0163 0.0187 0.0163
non computable correlation histogram	0.0125
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0300 0.0275 0.0312 0.0262 0.0100 0.0163 0.0088 0.0088 0.0063
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0013
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0037
Binary Attributes	0.0075
Frac1	0.0213
First Canonical Correlation	0.0300
Mean Skew	0.0187
Mean Kurtosis	0.0163
Class Entropy	0.0362
Mean Attribute Entropy	0.0262
Mean Mutual Information	0.0150
Equivalent number of attributes	0.0125
Noise to Signal Ratio	0.0088
Mean Mult. Correl. Coef.	0.0362
SDratio	0.0213

Table A.14. Lindiscr c50boost

Attribute	
# classes	0.0274
# attributes	0.0192
# instances	0.0535
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0178
# unknown values	0.0549
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0261
# nominal attributes	0.0069
maxminmeanstdv of nominal attribute values	0.0123 0.0041 0.0123 0.0069
1..5 concentration histogram 6..10 concentration histogram	0.0192 0.0055 0.0137 0.0082 0.0069 0.0082 0.0041 0.0027 0.0000 0.0041
non computable conc. histogram	0.0055
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0137 0.0027 0.0082 0.0123 0.0123 0.0096 0.0000 0.0000 0.0000 0.0041
non computable conc. histogram with class	0.0014
# continuous attributes	0.0096
1..5 correlation histogram 6..10 correlation histogram	0.0329 0.0178 0.0137 0.0082 0.0274 0.0165 0.0192 0.0316 0.0014 0.0123
non computable correlation histogram	0.0137
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0165 0.0219 0.0302 0.0247 0.0082 0.0206 0.0000 0.0096 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0014
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0041
Binary Attributes	0.0041
Frac1	0.0343
First Canonical Correlation	0.0412
Mean Skew	0.0123
Mean Kurtosis	0.0110
Class Entropy	0.0439
Mean Attribute Entropy	0.0233
Mean Mutual Information	0.0192
Equivalent number of attributes	0.0316
Noise to Signal Ratio	0.0192
Mean Mult. Correl. Coef.	0.0192
SDratio	0.0151

Table A.15. Lindiscr c50rules

Attribute	
# classes	0.0212
# attributes	0.0279
# instances	0.0517
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0252
# unknown values	0.0464
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0199
# nominal attributes	0.0053
maxminmeanstdv of nominal attribute values	0.0292 0.0066 0.0027 0.0013
1..5 concentration histogram 6..10 concentration histogram	0.0133 0.0172 0.0146 0.0040 0.0186 0.0080 0.0053 0.0000 0.0000 0.0080
non computable conc. histogram	0.0080
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0027 0.0080 0.0093 0.0186 0.0040 0.0027 0.0000 0.0000 0.0000 0.0013
non computable conc. histogram with class	0.0013
# continuous attributes	0.0106
1..5 correlation histogram 6..10 correlation histogram	0.0172 0.0265 0.0093 0.0199 0.0186 0.0133 0.0199 0.0133 0.0199 0.0186
non computable correlation histogram	0.0080
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0225 0.0199 0.0225 0.0239 0.0186 0.0133 0.0013 0.0093 0.0080
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0027
Binary Attributes	0.0040
Frac1	0.0279
First Canonical Correlation	0.0332
Mean Skew	0.0239
Mean Kurtosis	0.0066
Class Entropy	0.0332
Mean Attribute Entropy	0.0318
Mean Mutual Information	0.0225
Equivalent number of attributes	0.0265
Noise to Signal Ratio	0.0199
Mean Mult. Correl. Coef.	0.0265
SDratio	0.0252

Table A.16. Lindiscr c50tree

Attribute	
# classes	0.0295
# attributes	0.0121
# instances	0.0456
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0295
# unknown values	0.0416
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0161
# nominal attributes	0.0040
maxminmeanstdv of nominal attribute values	0.0188 0.0067 0.0094 0.0067
1..5 concentration histogram 6..10 concentration histogram	0.0094 0.0134 0.0067 0.0027 0.0121 0.0027 0.0054 0.0027 0.0000 0.0121
non computable conc. histogram	0.0174
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0080 0.0067 0.0054 0.0134 0.0067 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0013
# continuous attributes	0.0188
1..5 correlation histogram 6..10 correlation histogram	0.0268 0.0174 0.0241 0.0094 0.0134 0.0161 0.0201 0.0228 0.0147 0.0188
non computable correlation histogram	0.0161
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0389 0.0295 0.0295 0.0255 0.0282 0.0094 0.0013 0.0027 0.0054
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0013
Binary Attributes	0.0000
Frac1	0.0188
First Canonical Correlation	0.0416
Mean Skew	0.0174
Mean Kurtosis	0.0161
Class Entropy	0.0375
Mean Attribute Entropy	0.0282
Mean Mutual Information	0.0174
Equivalent number of attributes	0.0349
Noise to Signal Ratio	0.0094
Mean Mult. Correl. Coef.	0.0241
SDratio	0.0188

Table A.17. Ltree c50boost

Attribute	
# classes	0.0355
# attributes	0.0334
# instances	0.0284
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0304
# unknown values	0.0395
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0223
# nominal attributes	0.0152
max min mean stdv of nominal attribute values	0.0142 0.0101 0.0111 0.0091
1..5 concentration histogram 6..10 concentration histogram	0.0172 0.0122 0.0203 0.0071 0.0122 0.0041 0.0061 0.0000 0.0000 0.0122
non computable conc. histogram	0.0041
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0162 0.0091 0.0091 0.0213 0.0061 0.0010 0.0000 0.0000 0.0000 0.0020
non computable conc. histogram with class	0.0000
# continuous attributes	0.0172
1..5 correlation histogram 6..10 correlation histogram	0.0152 0.0132 0.0142 0.0182 0.0142 0.0172 0.0071 0.0111 0.0111 0.0152
non computable correlation histogram	0.0051
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0193 0.0344 0.0182 0.0324 0.0253 0.0132 0.0000 0.0182 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0030
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0020
Binary Attributes	0.0030
Frac1	0.0233
First Canonical Correlation	0.0203
Mean Skew	0.0122
Mean Kurtosis	0.0263
Class Entropy	0.0537
Mean Attribute Entropy	0.0193
Mean Mutual Information	0.0284
Equivalent number of attributes	0.0233
Noise to Signal Ratio	0.0152
Mean Mult. Correl. Coef.	0.0233
SDratio	0.0172

Table A.18. Ltree c50rules

Attribute	
# classes	0.0271
# attributes	0.0188
# instances	0.0329
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0212
# unknown values	0.0588
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0212
# nominal attributes	0.0141
max, min, mean, stdv of nominal attribute values	0.0176 0.0094 0.0082 0.0035
1..5 concentration histogram 6..10 concentration histogram	0.0059 0.0106 0.0118 0.0059 0.0059 0.0012 0.0012 0.0000 0.0000 0.0059
non computable conc. histogram	0.0094
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0165 0.0118 0.0047 0.0235 0.0118 0.0000 0.0000 0.0000 0.0012 0.0047
non computable conc. histogram with class	0.0000
# continuous attributes	0.0200
1..5 correlation histogram 6..10 correlation histogram	0.0188 0.0188 0.0059 0.0176 0.0188 0.0188 0.0200 0.0212 0.0082 0.0141
non computable correlation histogram	0.0141
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0318 0.0247 0.0282 0.0282 0.0235 0.0129 0.0024 0.0129 0.0012
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0012
Binary Attributes	0.0118
Frac1	0.0224
First Canonical Correlation	0.0400
Mean Skew	0.0153
Mean Kurtosis	0.0082
Class Entropy	0.0294
Mean Attribute Entropy	0.0235
Mean Mutual Information	0.0235
Equivalent number of attributes	0.0200
Noise to Signal Ratio	0.0200
Mean Mult. Correl. Coef.	0.0282
SDratio	0.0294

Table A.19. Ltree c50tree

Attribute	
# classes	0.0206
# attributes	0.0137
# instances	0.0297
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0297
# unknown values	0.0503
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0206
# nominal attributes	0.0183
max, min, mean, stdv of nominal attribute values	0.0149 0.0069 0.0103 0.0092
1..5 concentration histogram 6..10 concentration histogram	0.0126 0.0046 0.0183 0.0183 0.0046 0.0023 0.0011 0.0000 0.0000 0.0046
non computable conc. histogram	0.0000
1..10 concentration histogram with class 1..10 concentration histogram with class	0.0172 0.0103 0.0046 0.0195 0.0103 0.0011 0.0000 0.0000 0.0000 0.0011
non computable conc. histogram with class	0.0011
# continuous attributes	0.0114
1..5 correlation histogram 6..10 correlation histogram	0.0229 0.0114 0.0149 0.0263 0.0114 0.0149 0.0137 0.0069 0.0172 0.0160
non computable correlation histogram	0.0092
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0458 0.0206 0.0240 0.0378 0.0240 0.0160 0.0011 0.0137 0.0046
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0046
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0034
Binary Attributes	0.0080
Frac1	0.0206
First Canonical Correlation	0.0423
Mean Skew	0.0217
Mean Kurtosis	0.0160
Class Entropy	0.0343
Mean Attribute Entropy	0.0206
Mean Mutual Information	0.0240
Equivalent number of attributes	0.0172
Noise to Signal Ratio	0.0240
Mean Mult. Correl. Coef.	0.0217
SDratio	0.0217

Table A.20. Ltree Lindiscr

Attribute	
# classes	0.0339
# attributes	0.0230
# instances	0.0411
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0230
# unknown values	0.0351
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0254
# nominal attributes	0.0097
max, min, mean, stdv of nominal attribute values	0.0181 0.0097 0.0109 0.0048
1..5 concentration histogram 6..10 concentration histogram	0.0169 0.0157 0.0145 0.0145 0.0024 0.0060 0.0024 0.0000 0.0000 0.0121
non computable conc. histogram	0.0048
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0060 0.0169 0.0060 0.0181 0.0060 0.0024 0.0000 0.0000 0.0012 0.0036
non computable conc. histogram with class	0.0000
# continuous attributes	0.0145
1..5 correlation histogram 6..10 correlation histogram	0.0181 0.0266 0.0133 0.0121 0.0193 0.0157 0.0266 0.0230 0.0085 0.0145
non computable correlation histogram	0.0157
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0399 0.0206 0.0206 0.0278 0.0230 0.0133 0.0012 0.0121 0.0073
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0024
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0024
Binary Attributes	0.0024
Frac1	0.0097
First Canonical Correlation	0.0339
Mean Skew	0.0242
Mean Kurtosis	0.0060
Class Entropy	0.0230
Mean Attribute Entropy	0.0411
Mean Mutual Information	0.0206
Equivalent number of attributes	0.0145
Noise to Signal Ratio	0.0181
Mean Mult. Correl. Coef.	0.0193
SDratio	0.0242

Table A.21. IBL c50boost

Attribute	
# classes	0.0233
# attributes	0.0233
# instances	0.0480
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0262
# unknown values	0.0408
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0189
# nominal attributes	0.0087
max, min, mean, stdv of nominal attribute values	0.0160 0.0102 0.0131 0.0233
1..5 concentration histogram 6..10 concentration histogram	0.0247 0.0131 0.0146 0.0058 0.0189 0.0044 0.0044 0.0000 0.0000 0.0058
non computable conc. histogram	0.0058
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0116 0.0131 0.0000 0.0160 0.0058 0.0000 0.0000 0.0000 0.0015 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0102
1..5 correlation histogram 6..10 correlation histogram	0.0175 0.0131 0.0189 0.0131 0.0102 0.0146 0.0102 0.0160 0.0204 0.0131
non computable correlation histogram	0.0175
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0277 0.0408 0.0306 0.0262 0.0175 0.0146 0.0029 0.0116 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0029
Binary Attributes	0.0087
Frac1	0.0073
First Canonical Correlation	0.0146
Mean Skew	0.0175
Mean Kurtosis	0.0204
Class Entropy	0.0408
Mean Attribute Entropy	0.0291
Mean Mutual Information	0.0335
Equivalent number of attributes	0.0116
Noise to Signal Ratio	0.0218
Mean Mult. Correl. Coef.	0.0335
SDratio	0.0146

Table A.22. IBL c50rules

Attribute	
# classes	0.0220
# attributes	0.0220
# instances	0.0356
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0251
# unknown values	0.0460
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0157
# nominal attributes	0.0052
max, min, mean, stdv of nominal attribute values	0.0136 0.0073 0.0178 0.0146
1..5 concentration histogram 6..10 concentration histogram	0.0115 0.0063 0.0073 0.0136 0.0126 0.0073 0.0010 0.0031 0.0000 0.0073
non computable conc. histogram	0.0094
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0042 0.0126 0.0073 0.0146 0.0052 0.0021 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0010
# continuous attributes	0.0167
1..5 correlation histogram 6..10 correlation histogram	0.0220 0.0209 0.0146 0.0167 0.0146 0.0178 0.0251 0.0157 0.0126 0.0115
non computable correlation histogram	0.0146
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0241 0.0272 0.0314 0.0377 0.0262 0.0188 0.0010 0.0188 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0010
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0031
Binary Attributes	0.0105
Frac1	0.0146
First Canonical Correlation	0.0282
Mean Skew	0.0126
Mean Kurtosis	0.0105
Class Entropy	0.0575
Mean Attribute Entropy	0.0241
Mean Mutual Information	0.0314
Equivalent number of attributes	0.0188
Noise to Signal Ratio	0.0126
Mean Mult. Correl. Coef.	0.0167
SDratio	0.0188

Table A.23. IBL c50tree

Attribute	
# classes	0.0282
# attributes	0.0282
# instances	0.0248
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0248
# unknown values	0.0508
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0169
# nominal attributes	0.0045
max, min, mean, stdv of nominal attribute values	0.0147 0.0147 0.0135 0.0135
1..5 concentration histogram 6..10 concentration histogram	0.0090 0.0056 0.0011 0.0169 0.0124 0.0056 0.0056 0.0011 0.0000 0.0068
non computable conc. histogram	0.0147
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0113 0.0045 0.0034 0.0169 0.0079 0.0068 0.0011 0.0000 0.0011 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0192
1..5 correlation histogram 6..10 correlation histogram	0.0181 0.0169 0.0192 0.0203 0.0192 0.0147 0.0158 0.0113 0.0113 0.0226
non computable correlation histogram	0.0113
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0271 0.0203 0.0316 0.0147 0.0169 0.0203 0.0000 0.0158 0.0011
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0034
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0034
Binary Attributes	0.0045
Frac1	0.0169
First Canonical Correlation	0.0361
Mean Skew	0.0158
Mean Kurtosis	0.0090
Class Entropy	0.0632
Mean Attribute Entropy	0.0226
Mean Mutual Information	0.0260
Equivalent number of attributes	0.0181
Noise to Signal Ratio	0.0203
Mean Mult. Correl. Coef.	0.0271
SDratio	0.0192

Table A.24. IBL Lindiscr

Attribute	Frequency
# classes	0.0409
# attributes	0.0227
# instances	0.0329
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0341
# unknown values	0.0443
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0216
# nominal attributes	0.0045
maxminmeanstdv of nominal attribute values	0.0136 0.0068 0.0136 0.0125
1..5 concentration histogram 6..10 concentration histogram	0.0125 0.0079 0.0114 0.0102 0.0125 0.0102 0.0045 0.0011 0.0000 0.0045
non computable conc. histogram	0.0114
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0102 0.0057 0.0045 0.0125 0.0057 0.0011 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0148
1..5 correlation histogram 6..10 correlation histogram	0.0295 0.0159 0.0148 0.0238 0.0170 0.0159 0.0204 0.0091 0.0125 0.0114
non computable correlation histogram	0.0148
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0261 0.0318 0.0295 0.0193 0.0238 0.0148 0.0114 0.0148 0.0045
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0011
Binary Attributes	0.0170
Frac1	0.0079
First Canonical Correlation	0.0159
Mean Skew	0.0182
Mean Kurtosis	0.0170
Class Entropy	0.0477
Mean Attribute Entropy	0.0250
Mean Mutual Information	0.0204
Equivalent number of attributes	0.0125
Noise to Signal Ratio	0.0261
Mean Mult. Correl. Coef.	0.0216
SDratio	0.0204

Table A.25. IBL Ltree

Attribute	
# classes	0.0212
# attributes	0.0212
# instances	0.0339
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0275
# unknown values	0.0508
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0190
# nominal attributes	0.0085
maxminmeanstdv of nominal attribute values	0.0116 0.0042 0.0116 0.0169
1..5 concentration histogram 6..10 concentration histogram	0.0095 0.0042 0.0063 0.0106 0.0074 0.0116 0.0032 0.0000 0.0011 0.0063
non computable conc. histogram	0.0074
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0085 0.0106 0.0042 0.0169 0.0074 0.0042 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0127
1..5 correlation histogram 6..10 correlation histogram	0.0180 0.0212 0.0243 0.0127 0.0116 0.0169 0.0201 0.0169 0.0243 0.0085
non computable correlation histogram	0.0106
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0317 0.0455 0.0243 0.0201 0.0180 0.0063 0.0011 0.0053 0.0011
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0032
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0032
Binary Attributes	0.0063
Frac1	0.0201
First Canonical Correlation	0.0275
Mean Skew	0.0127
Mean Kurtosis	0.0243
Class Entropy	0.0593
Mean Attribute Entropy	0.0349
Mean Mutual Information	0.0265
Equivalent number of attributes	0.0148
Noise to Signal Ratio	0.0254
Mean Mult. Correl. Coef.	0.0254
SDratio	0.0190

Table A.26. NB c50boost

Attribute	
# classes	0.0249
# attributes	0.0345
# instances	0.0401
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0180
# unknown values	0.0497
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0249
# nominal attributes	0.0152
maxminmeanstdv of nominal attribute values	0.0138 0.0124 0.0083 0.0152
1..5 concentration histogram 6..10 concentration histogram	0.0180 0.0069 0.0193 0.0097 0.0069 0.0041 0.0055 0.0014 0.0000 0.0083
non computable conc. histogram	0.0097
1..10 concentration histogram with class 1..10 concentration histogram with class	0.0207 0.0055 0.0097 0.0290 0.0055 0.0014 0.0000 0.0000 0.0014 0.0055
non computable conc. histogram with class	0.0014

# continuous attributes 	
1..5 correlation histogram
6..10 correlation histogram
	0.0110
	0.0166 0.0069 0.0138 0.0152 0.0124
	0.0041 0.0138 0.0193 0.0207 0.0180

non computable correlation histogram	0.0110
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0166 0.0235 0.0180 0.0221 0.0207 0.0152 0.0000 0.0097 0.0055
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0041
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0041
Binary Attributes	0.0055
Frac1	0.0055
First Canonical Correlation	0.0262
Mean Skew	0.0041
Mean Kurtosis	0.0138
Class Entropy	0.0552
Mean Attribute Entropy	0.0304
Mean Mutual Information	0.0401
Equivalent number of attributes	0.0331
Noise to Signal Ratio	0.0221
Mean Mult. Correl. Coef.	0.0138
SDratio	0.0207

Table A.27. NB c50rules

Attribute	
# classes	0.0387
# attributes	0.0193
# instances	0.0335
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0271
# unknown values	0.0593
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0258
# nominal attributes	0.0103
max, min, mean, stdv of nominal attribute values	0.0142 0.0116 0.0077 0.0077
1..5 concentration histogram 6..10 concentration histogram	0.0052 0.0155 0.0129 0.0090 0.0026 0.0000 0.0064 0.0000 0.0000 0.0077
non computable conc. histogram	0.0026
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0193 0.0168 0.0103 0.0026 0.0129 0.0026 0.0000 0.0000 0.0013 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0116
1..5 correlation histogram 6..10 correlation histogram	0.0193 0.0142 0.0155 0.0103 0.0193 0.0090 0.0155 0.0142 0.0180 0.0142
non computable correlation histogram	0.0168
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0258 0.0309 0.0374 0.0322 0.0116 0.0052 0.0026 0.0142 0.0077
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0039
Binary Attributes	0.0013
Frac1	0.0284
First Canonical Correlation	0.0245
Mean Skew	0.0219
Mean Kurtosis	0.0258
Class Entropy	0.0309
Mean Attribute Entropy	0.0193
Mean Mutual Information	0.0361
Equivalent number of attributes	0.0245
Noise to Signal Ratio	0.0155
Mean Mult. Correl. Coef.	0.0168
SDratio	0.0232

Table A.28. NB c50tree

Attribute	
# classes	0.0317
# attributes	0.0238
# instances	0.0450
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0079
# unknown values	0.0516
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0225
# nominal attributes	0.0093
max, min, mean, stdv of nominal attribute values	0.0132 0.0119 0.0132 0.0040
1..5 concentration histogram 6..10 concentration histogram	0.0093 0.0172 0.0040 0.0119 0.0066 0.0053 0.0040 0.0000 0.0000 0.0106
non computable conc. histogram	0.0079
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0198 0.0119 0.0119 0.0212 0.0106 0.0040 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0106
1..5 correlation histogram 6..10 correlation histogram	0.0251 0.0212 0.0185 0.0172 0.0212 0.0132 0.0185 0.0198 0.0238 0.0172
non computable correlation histogram	0.0079
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0119 0.0172 0.0278 0.0278 0.0146 0.0172 0.0026 0.0146 0.0093
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0026
Frac1	0.0106
First Canonical Correlation	0.0357
Mean Skew	0.0198
Mean Kurtosis	0.0212
Class Entropy	0.0370
Mean Attribute Entropy	0.0172
Mean Mutual Information	0.0278
Equivalent number of attributes	0.0198
Noise to Signal Ratio	0.0238
Mean Mult. Correl. Coef.	0.0225
SDratio	0.0146

Table A.29. NB Lindiscr

Attribute	
# classes	0.0365
# attributes	0.0166
# instances	0.0476
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0188
# unknown values	0.0465
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0310
# nominal attributes	0.0122
max, min, mean, stdv of nominal attribute values	0.0188 0.0066 0.0166 0.0066
1..5 concentration histogram 6..10 concentration histogram	0.0177 0.0122 0.0100 0.0111 0.0166 0.0077 0.0077 0.0011 0.0000 0.0122
non computable conc. histogram	0.0133
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0022 0.0166 0.0077 0.0288 0.0133 0.0077 0.0000 0.0000 0.0000 0.0011
non computable conc. histogram with class	0.0000
# continuous attributes	0.0077
1..5 correlation histogram 6..10 correlation histogram	0.0144 0.0221 0.0188 0.0100 0.0100 0.0088 0.0144 0.0122 0.0122 0.0144
non computable correlation histogram	0.0122
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0254 0.0221 0.0376 0.0288 0.0188 0.0155 0.0000 0.0122 0.0044
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0011
Binary Attributes	0.0088
Frac1	0.0122
First Canonical Correlation	0.0221
Mean Skew	0.0077
Mean Kurtosis	0.0100
Class Entropy	0.0498
Mean Attribute Entropy	0.0254
Mean Mutual Information	0.0221
Equivalent number of attributes	0.0155
Noise to Signal Ratio	0.0265
Mean Mult. Correl. Coef.	0.0077
SDratio	0.0243

Table A.30. NB Ltree

Attribute	
# classes	0.0411
# attributes	0.0279
# instances	0.0264
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0235
# unknown values	0.0558
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0250
# nominal attributes	0.0132
max, min, mean, stdv of nominal attribute values	0.0382 0.0117 0.0088 0.0132
1..5 concentration histogram 6..10 concentration histogram	0.0059 0.0059 0.0073 0.0117 0.0073 0.0059 0.0044 0.0029 0.0015 0.0059
non computable conc. histogram	0.0103
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0147 0.0191 0.0029 0.0103 0.0059 0.0029 0.0015 0.0000 0.0000 0.0059
non computable conc. histogram with class	0.0000
# continuous attributes	0.0117
1..5 correlation histogram 6..10 correlation histogram	0.0132 0.0147 0.0117 0.0088 0.0176 0.0000 0.0294 0.0015 0.0117 0.0088
non computable correlation histogram	0.0250
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0132 0.0162 0.0235 0.0338 0.0279 0.0206 0.0029 0.0132 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0059
Binary Attributes	0.0088
Frac1	0.0029
First Canonical Correlation	0.0220
Mean Skew	0.0162
Mean Kurtosis	0.0132
Class Entropy	0.0485
Mean Attribute Entropy	0.0352
Mean Mutual Information	0.0191
Equivalent number of attributes	0.0294
Noise to Signal Ratio	0.0206
Mean Mult. Correl. Coef.	0.0206
SDratio	0.0352

Table A.31. NB IBL

Attribute	
# classes	0.0337
# attributes	0.0180
# instances	0.0416
$\frac{\# \text{attributes}}{\# \text{instances}}$	0.0427
# unknown values	0.0551
$\frac{\# \text{unknown values}}{\# \text{attributes} * \# \text{instances}}$	0.0202
# nominal attributes	0.0101
max, min, mean, stdv of nominal attribute values	0.0112 0.0034 0.0146 0.0067
1..5 concentration histogram 6..10 concentration histogram	0.0135 0.0090 0.0135 0.0112 0.0124 0.0034 0.0067 0.0000 0.0000 0.0022
non computable conc. histogram	0.0101
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0191 0.0124 0.0112 0.0202 0.0079 0.0056 0.0000 0.0011 0.0022 0.0011
non computable conc. histogram with class	0.0000
# continuous attributes	0.0101
1..5 correlation histogram 6..10 correlation histogram	0.0202 0.0090 0.0180 0.0090 0.0157 0.0124 0.0112 0.0112 0.0146 0.0079
non computable correlation histogram	0.0101
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0247 0.0270 0.0360 0.0337 0.0180 0.0112 0.0000 0.0079 0.0022
$\frac{\# \text{continuous}}{\# \text{attributes}}$	0.0011
$\frac{\# \text{nominal}}{\# \text{attributes}}$	0.0000
Binary Attributes	0.0101
Frac1	0.0180
First Canonical Correlation	0.0214
Mean Skew	0.0180
Mean Kurtosis	0.0169
Class Entropy	0.0416
Mean Attribute Entropy	0.0225
Mean Mutual Information	0.0405
Equivalent number of attributes	0.0214
Noise to Signal Ratio	0.0202
Mean Mult. Correl. Coef.	0.0202
SDratio	0.0169

Table A.32. ripper c50boost

Attribute	
# classes	0.0231
# attributes	0.0272
# instances	0.0340
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0190
# unknown values	0.0367
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0286
# nominal attributes	0.0068
max, min, mean, stdv of nominal attribute values	0.0122 0.0095 0.0095 0.0150
1..5 concentration histogram 6..10 concentration histogram	0.0122 0.0109 0.0136 0.0150 0.0068 0.0041 0.0054 0.0000 0.0000 0.0041
non computable conc. histogram	0.0109
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0136 0.0136 0.0068 0.0299 0.0068 0.0041 0.0000 0.0000 0.0000 0.0014
non computable conc. histogram with class	0.0000
# continuous attributes	0.0150
1..5 correlation histogram 6..10 correlation histogram	0.0122 0.0367 0.0122 0.0150 0.0163 0.0177 0.0136 0.0204 0.0163 0.0204
non computable correlation histogram	0.0068
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0476 0.0313 0.0218 0.0354 0.0095 0.0082 0.0014 0.0122 0.0068
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0027
Binary Attributes	0.0163
Frac1	0.0136
First Canonical Correlation	0.0190
Mean Skew	0.0163
Mean Kurtosis	0.0082
Class Entropy	0.0503
Mean Attribute Entropy	0.0327
Mean Mutual Information	0.0082
Equivalent number of attributes	0.0245
Noise to Signal Ratio	0.0150
Mean Mult. Correl. Coef.	0.0136
SDratio	0.0218

Table A.33. ripper c50rules

Attribute	
# classes	0.0226
# attributes	0.0192
# instances	0.0238
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0238
# unknown values	0.0543
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0305
# nominal attributes	0.0102
max, min, mean, stdv of nominal attribute values	0.0170 0.0045 0.0057 0.0147
1..5 concentration histogram 6..10 concentration histogram	0.0045 0.0147 0.0124 0.0057 0.0102 0.0000 0.0057 0.0011 0.0011 0.0147
non computable conc. histogram	0.0102
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0204 0.0124 0.0023 0.0147 0.0102 0.0034 0.0000 0.0000 0.0000 0.0011
non computable conc. histogram with class	0.0000
# continuous attributes	0.0113
1..5 correlation histogram 6..10 correlation histogram	0.0102 0.0136 0.0147 0.0079 0.0113 0.0260 0.0192 0.0079 0.0170 0.0158
non computable correlation histogram	0.0170
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0260 0.0339 0.0192 0.0294 0.0124 0.0226 0.0057 0.0079 0.0023
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0011
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0023
Binary Attributes	0.0124
Frac1	0.0192
First Canonical Correlation	0.0192
Mean Skew	0.0204
Mean Kurtosis	0.0102
Class Entropy	0.0554
Mean Attribute Entropy	0.0204
Mean Mutual Information	0.0351
Equivalent number of attributes	0.0362
Noise to Signal Ratio	0.0283
Mean Mult. Correl. Coef.	0.0192
SDratio	0.0181

Table A.34. ripper c50tree

Attribute	
# classes	0.0221
# attributes	0.0208
# instances	0.0294
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0257
# unknown values	0.0502
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0147
# nominal attributes	0.0074
max, min, mean, stdv of nominal attribute values	0.0123 0.0086 0.0135 0.0135
1..5 concentration histogram 6..10 concentration histogram	0.0257 0.0025 0.0086 0.0061 0.0061 0.0049 0.0061 0.0000 0.0000 0.0123
non computable conc. histogram	0.0110
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0159 0.0110 0.0012 0.0221 0.0061 0.0025 0.0012 0.0000 0.0000 0.0012
non computable conc. histogram with class	0.0000
# continuous attributes	0.0172
1..5 correlation histogram 6..10 correlation histogram	0.0196 0.0110 0.0061 0.0135 0.0135 0.0270 0.0159 0.0135 0.0159 0.0172
non computable correlation histogram	0.0086
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0294 0.0319 0.0270 0.0221 0.0245 0.0159 0.0000 0.0172 0.0012
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0061
Binary Attributes	0.0110
Frac1	0.0159
First Canonical Correlation	0.0098
Mean Skew	0.0147
Mean Kurtosis	0.0147
Class Entropy	0.0613
Mean Attribute Entropy	0.0233
Mean Mutual Information	0.0306
Equivalent number of attributes	0.0429
Noise to Signal Ratio	0.0257
Mean Mult. Correl. Coef.	0.0159
SDratio	0.0172

Table A.35. ripper Lindiscr

Attribute	
# classes	0.0264
# attributes	0.0242
# instances	0.0562
$\frac{\#\text{ attributes}}{\#\text{instances}}$	0.0132
# unknown values	0.0628
$\frac{\#\text{ unknown values}}{\#\text{ attributes * }\#\text{ instances}}$	0.0242
# nominal attributes	0.0099
max, min, mean, stdv of nominal attribute values	0.0176 0.0066 0.0033 0.0055
1..5 concentration histogram 6..10 concentration histogram	0.0077 0.0110 0.0044 0.0132 0.0110 0.0077 0.0044 0.0000 0.0000 0.0055
non computable conc. histogram	0.0187
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0110 0.0143 0.0110 0.0187 0.0044 0.0000 0.0000 0.0000 0.0000 0.0044
non computable conc. histogram with class	0.0000
# continuous attributes	0.0099
1..6 correlation histogram 6..10 correlation histogram	0.0176 0.0220 0.0143 0.0132 0.0198 0.0121 0.0132 0.0165 0.0154 0.0143
non computable correlation histogram	0.0110
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0341 0.0452 0.0275 0.0441 0.0176 0.0165 0.0011 0.0143 0.0000
$\frac{\#\text{ continuous}}{\#\text{ attributes}}$	0.0011
$\frac{\#\text{ nominal}}{\#\text{ attributes}}$	0.0000
Binary Attributes	0.0066
Frac1	0.0187
First Canonical Correlation	0.0209
Mean Skew	0.0187
Mean Kurtosis	0.0099
Class Entropy	0.0220
Mean Attribute Entropy	0.0297
Mean Mutual Information	0.0308
Equivalent number of attributes	0.0154
Noise to Signal Ratio	0.0121
Mean Mult. Correl. Coef.	0.0154
SDratio	0.0209

Table A.36. ripper Ltree

Attribute	
# classes	0.0283
# attributes	0.0215
# instances	0.0351
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0261
# unknown values	0.0431
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0136
# nominal attributes	0.0091
max, min, mean, stdv of nominal attribute values	0.0170 0.0125 0.0125 0.0079
1..5 concentration histogram 6..10 concentration histogram	0.0136 0.0102 0.0079 0.0125 0.0045 0.0045 0.0079 0.0011 0.0000 0.0102
non computable conc. histogram	0.0057
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0102 0.0147 0.0034 0.0283 0.0023 0.0023 0.0000 0.0000 0.0011 0.0057
non computable conc. histogram with class	0.0011
# continuous attributes	0.0045
1..10 correlation histogram 6..10 correlation histogram	0.0125 0.0204 0.0159 0.0147 0.0125 0.0136 0.0102 0.0079 0.0113 0.0057
non computable correlation histogram	0.0113
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0408 0.0272 0.0306 0.0295 0.0261 0.0159 0.0034 0.0181 0.0023
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0011
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0011
Binary Attributes	0.0159
Frac1	0.0125
First Canonical Correlation	0.0283
Mean Skew	0.0215
Mean Kurtosis	0.0147
Class Entropy	0.0510
Mean Attribute Entropy	0.0363
Mean Mutual Information	0.0193
Equivalent number of attributes	0.0283
Noise to Signal Ratio	0.0159
Mean Mult. Correl. Coef.	0.0215
SDratio	0.0238

Table A.37. rippel IBL

Attribute	
# classes	0.0324
# attributes	0.0252
# instances	0.0372
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0288
# unknown values	0.0408
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0120
# nominal attributes	0.0120
max, min, mean, stdv of nominal attribute values	0.0096 0.0096 0.0096 0.0096
1..5 concentration histogram 6..10 concentration histogram	0.0156 0.0120 0.0084 0.0072 0.0120 0.0084 0.0024 0.0000 0.0000 0.0072
non computable conc. histogram	0.0132
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0156 0.0072 0.0048 0.0168 0.0084 0.0036 0.0000 0.0000 0.0000 0.0024
non computable conc. histogram with class	0.0000
# continuous attributes	0.0204
1..5 correlation histogram 6..10 correlation histogram	0.0216 0.0240 0.0156 0.0108 0.0180 0.0204 0.0120 0.0108 0.0096 0.0204
non computable correlation histogram	0.0132
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0300 0.0288 0.0204 0.0156 0.0168 0.0204 0.0060 0.0168 0.0036
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0036
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0048
Binary Attributes	0.0216
Frac1	0.0072
First Canonical Correlation	0.0276
Mean Skew	0.0108
Mean Kurtosis	0.0168
Class Entropy	0.0552
Mean Attribute Entropy	0.0300
Mean Mutual Information	0.0216
Equivalent number of attributes	0.0168
Noise to Signal Ratio	0.0192
Mean Mult. Correl. Coef.	0.0192
SDratio	0.0192

Table A.38. ripper NB

Attribute	
# classes	0.0363
# attributes	0.0264
# instances	0.0474
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0231
# unknown values	0.0562
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0220
# nominal attributes	0.0099
maxminmeanstdv of nominal attribute values	0.0176 0.0088 0.0143 0.0088
1..5 concentration histogram 6..10 concentration histogram	0.0077 0.0088 0.0132 0.0088 0.0066 0.0022 0.0077 0.0000 0.0000 0.0110
non computable conc. histogram	0.0022
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0110 0.0132 0.0066 0.0110 0.0033 0.0088 0.0000 0.0000 0.0011 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0088
1..5 correlation histogram 6..10 correlation histogram	0.0176 0.0143 0.0154 0.0099 0.0165 0.0143 0.0143 0.0154 0.0088 0.0143
non computable correlation histogram	0.0110
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0242 0.0341 0.0253 0.0352 0.0198 0.0055 0.0011 0.0154 0.0055
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0033
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0022
Binary Attributes	0.0033
Frac1	0.0077
First Canonical Correlation	0.0242
Mean Skew	0.0220
Mean Kurtosis	0.0209
Class Entropy	0.0540
Mean Attribute Entropy	0.0253
Mean Mutual Information	0.0275
Equivalent number of attributes	0.0308
Noise to Signal Ratio	0.0242
Mean Mult. Correl. Coef.	0.0132
SDratio	0.0198

A.4 Selection Frequency by c50tree

Table A.39. c50rules c50boost

Attribute	
# classes	0.0395
# attributes	0.0526
# instances	0.0132
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0263
# unknown values	0.0395
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0263
# nominal attributes	0.0132
max, min, mean, stdv of nominal attribute values	0.0263 0.0132 0.0263 0.0132
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0263 0.0132 0.0132 0.0132 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0263 0.0000 0.0000 0.0132 0.0132 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0395 0.0263 0.0000 0.0000 0.0132 0.0132 0.0000 0.0132 0.0263
non computable correlation histogram	0.0263
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0263 0.0395 0.0263 0.0395 0.0395 0.0263 0.0132 0.0132 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0132
Frac1	0.0263
First Canonical Correlation	0.0000
Mean Skew	0.0132
Mean Kurtosis	0.0132
Class Entropy	0.0526
Mean Attribute Entropy	0.0395
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0526
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table A.40. c50tree c50boost

Attribute	
# classes	0.0471
# attributes	0.0235
# instances	0.0471
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0353
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0235
# nominal attributes	0.0000
maxminmeanstdv of nominal attribute values	0.0235 0.0000 0.0235 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0118 0.0118 0.0000 0.0000 0.0000 0.0118 0.0118 0.0000 0.0000 0.0000
non computable conc. histogram	0.0118
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0235 0.0000 0.0000 0.0118 0.0000 0.0000 0.0000 0.0000 0.0000 0.0235
non computable conc. histogram with class	0.0000
# continuous attributes	0.0235
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0000 0.0118 0.0471 0.0118 0.0000 0.0118 0.0000 0.0353 0.0118
non computable correlation histogram	0.0235
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0588 0.0824 0.0235 0.0353 0.0235 0.0235 0.0000 0.0353 0.0118
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0118
First Canonical Correlation	0.0235
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0235
Mean Attribute Entropy	0.0235
Mean Mutual Information	0.0235
Equivalent number of attributes	0.0471
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0235
SDratio	0.0118

Table A.41. c50tree c50rules

Attribute	
# classes	0.0469
# attributes	0.0312
# instances	0.0312
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0312
# unknown values	0.0469
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0156
# nominal attributes	0.0000
maxminmeanstdv of nominal attribute values	0.0312 0.0000 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0156 0.0156 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0156
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0312 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0156
1..5 correlation histogram 6..10 correlation histogram	0.0156 0.0000 0.0156 0.0000 0.0156 0.0156 0.0156 0.0312 0.0469 0.0312
non computable correlation histogram	0.0156
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0469 0.0469 0.0625 0.0312 0.0156 0.0156 0.0156 0.0000 0.0156
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0156
Frac1	0.0469
First Canonical Correlation	0.0312
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0625
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0156
Mean Mult. Correl. Coef.	0.0312
SDratio	0.0156

Table A.42. Lindiscr c50boost

Attribute	
# classes	0.0328
# attributes	0.0328
# instances	0.0656
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0656
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0492
# nominal attributes	0.0000
maxminmeanstdv of nominal attribute values	0.0164 0.0164 0.0492 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0164
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0164 0.0000 0.0328 0.0164 0.0164 0.0000 0.0000 0.0000 0.0000 0.0164
non computable conc. histogram with class	0.0000
# continuous attributes	0.0164
1..5 correlation histogram 6..10 correlation histogram	0.0164 0.0164 0.0164 0.0000 0.0164 0.0000 0.0328 0.0164 0.0000 0.0164
non computable correlation histogram	0.0164
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0164 0.0328 0.0328 0.0164 0.0164 0.0000 0.0164 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0164
Frac1	0.0328
First Canonical Correlation	0.0492
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0492
Mean Attribute Entropy	0.0164
Mean Mutual Information	0.0164
Equivalent number of attributes	0.0820
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table A.43. Lindiscr c50rules

Attribute	
# classes	0.0159
# attributes	0.0317
# instances	0.0952
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0159
# unknown values	0.0476
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0159
max, min, mean, stdv of nominal attribute values	0.0317 0.0000 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0317 0.0159 0.0159 0.0159 0.0159 0.0159 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0159
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0159 0.0000 0.0159 0.0159 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0317 0.0317 0.0317 0.0159 0.0159 0.0159 0.0000 0.0000 0.0476 0.0317
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0159 0.0159 0.0317 0.0159 0.0159 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0159
Binary Attributes	0.0000
Frac1	0.0635
First Canonical Correlation	0.0476
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0317
Mean Attribute Entropy	0.0159
Mean Mutual Information	0.0317
Equivalent number of attributes	0.0159
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0159
SDratio	0.0159

Table A.44. Lindiscr c50tree

Attribute	
# classes	0.0484
# attributes	0.0161
# instances	0.0484
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0323
# unknown values	0.0323
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0161
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0323 0.0000 0.0161 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0484 0.0161 0.0000 0.0000 0.0000 0.0000 0.0000 0.0161 0.0000 0.0000
non computable conc. histogram	0.0323
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0323 0.0000 0.0000 0.0323 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0323 0.0000 0.0161 0.0161 0.0323 0.0161 0.0323 0.0161 0.0161 0.0323
non computable correlation histogram	0.0161
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0323 0.0000 0.0161 0.0161 0.0484 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0323
First Canonical Correlation	0.0645
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0645
Mean Attribute Entropy	0.0161
Mean Mutual Information	0.0161
Equivalent number of attributes	0.0161
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0161
SDratio	0.0161

Table A.45. Ltree c50boost

Attribute	
# classes	0.0526
# attributes	0.0316
# instances	0.0421
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0316
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0526
max, min, mean, stdv of nominal attribute values	0.0000 0.0211 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0211 0.0000 0.0211 0.0105 0.0316 0.0000 0.0105 0.0000 0.0000 0.0211
non computable conc. histogram	0.0105
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0105 0.0000 0.0211 0.0211 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0211
1..5 correlation histogram 6..10 correlation histogram	0.0105 0.0000 0.0105 0.0211 0.0211 0.0105 0.0105 0.0316 0.0000 0.0211
non computable correlation histogram	0.0211
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0211 0.0316 0.0211 0.0421 0.0316 0.0211 0.0000 0.0105 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0211
First Canonical Correlation	0.0105
Mean Skew	0.0105
Mean Kurtosis	0.0316
Class Entropy	0.0737
Mean Attribute Entropy	0.0105
Mean Mutual Information	0.0211
Equivalent number of attributes	0.0211
Noise to Signal Ratio	0.0105
Mean Mult. Correl. Coef.	0.0105
SDratio	0.0105

Table A.46. Ltree c50rules

Attribute	
# classes	0.0282
# attributes	0.0282
# instances	0.0704
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0141
# unknown values	0.0563
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0282
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0282 0.0141 0.0141 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0282 0.0423 0.0000 0.0282 0.0000 0.0000 0.0000 0.0000 0.0141
non computable conc. histogram	0.0141
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0282 0.0141 0.0000 0.0282 0.0000 0.0000 0.0000 0.0000 0.0000 0.0141
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0141 0.0141 0.0141 0.0282 0.0000 0.0423 0.0141 0.0141 0.0141 0.0000
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0423 0.0000 0.0141 0.0282 0.0282 0.0141 0.0000 0.0282 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0282
First Canonical Correlation	0.0423
Mean Skew	0.0141
Mean Kurtosis	0.0000
Class Entropy	0.0282
Mean Attribute Entropy	0.0141
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0141
Mean Mult. Correl. Coef.	0.0282
SDratio	0.0282

Table A.47. Ltree c50tree

Attribute	
# classes	0.0465
# attributes	0.0000
# instances	0.0349
$\frac{\# \text{ attributes}}{\#\text{instances}}$	0.0233
# unknown values	0.0698
$\frac{\# \text{ unknown values}}{\# \text{ attributes} * \# \text{ instances}}$	0.0349
# nominal attributes	0.0233
max, min, mean, stdv of nominal attribute values	0.0116 0.0233 0.0000 0.0116
1..5 concentration histogram 6..10 concentration histogram	0.0349 0.0000 0.0116 0.0116 0.0000 0.0000 0.0000 0.0000 0.0000 0.0116
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0233 0.0233 0.0000 0.0116 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0116 0.0116 0.0116 0.0349 0.0000 0.0000 0.0116 0.0116 0.0465 0.0233
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0581 0.0116 0.0116 0.0349 0.0233 0.0233 0.0000 0.0116 0.0116
$\frac{\# \text{ continuous}}{\# \text{ attributes}}$	0.0000
$\frac{\# \text{ nominal}}{\# \text{ attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0465
First Canonical Correlation	0.0349
Mean Skew	0.0233
Mean Kurtosis	0.0116
Class Entropy	0.0233
Mean Attribute Entropy	0.0116
Mean Mutual Information	0.0233
Equivalent number of attributes	0.0116
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0116
SDratio	0.0465

Table A.48. Ltree Lindiscr

Attribute	
# classes	0.0571
# attributes	0.0429
# instances	0.0571
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0143
# unknown values	0.0286
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0429
# nominal attributes	0.0286
max, min, mean, stdv of nominal attribute values	0.0286 0.0143 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0286 0.0143 0.0143 0.0000 0.0000 0.0143 0.0000 0.0000 0.0000 0.0143
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0286 0.0143 0.0143 0.0143 0.0000 0.0000 0.0000 0.0000 0.0143
non computable conc. histogram with class	0.0000
# continuous attributes	0.0286
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0143 0.0000 0.0000 0.0000 0.0286 0.0571 0.0429 0.0000 0.0143
non computable correlation histogram	0.0143
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0286 0.0143 0.0286 0.0286 0.0143 0.0143 0.0000 0.0143 0.0143
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0429
Mean Skew	0.0143
Mean Kurtosis	0.0000
Class Entropy	0.0143
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0143
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0143
Mean Mult. Correl. Coef.	0.0286
SDratio	0.0286

Table A.49. IBL c50boost

Attribute	
# classes	0.0484
# attributes	0.0323
# instances	0.0806
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0161
# unknown values	0.0323
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0645
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0323 0.0000 0.0161 0.0161
1..5 concentration histogram 6..10 concentration histogram	0.0323 0.0323 0.0161 0.0000 0.0323 0.0000 0.0000 0.0000 0.0000 0.0161
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0484 0.0161 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0161 0.0161 0.0000 0.0161 0.0000 0.0323 0.0161 0.0323 0.0161
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0161 0.0323 0.0484 0.0484 0.0000 0.0161 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0161
Frac1	0.0000
First Canonical Correlation	0.0323
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0645
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0323
SDratio	0.0161

Table A.50. IBL c50rules

Attribute	
# classes	0.0112
# attributes	0.0562
# instances	0.0225
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0337
# unknown values	0.0787
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0337 0.0112 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0112 0.0000 0.0000 0.0225 0.0000 0.0000 0.0000 0.0000 0.0000 0.0112
non computable conc. histogram	0.0112
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0112 0.0000 0.0112 0.0112 0.0112 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0112
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0112 0.0112 0.0225 0.0112 0.0112 0.0225 0.0112 0.0000 0.0112
non computable correlation histogram	0.0337
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0449 0.0449 0.0449 0.0225 0.0112 0.0337 0.0000 0.0112 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0112
Frac1	0.0225
First Canonical Correlation	0.0449
Mean Skew	0.0225
Mean Kurtosis	0.0112
Class Entropy	0.0899
Mean Attribute Entropy	0.0112
Mean Mutual Information	0.0337
Equivalent number of attributes	0.0225
Noise to Signal Ratio	0.0112
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table A.51. IBL c50tree

Attribute	
# classes	0.0581
# attributes	0.0581
# instances	0.0233
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0233
# unknown values	0.0930
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0116 0.0116 0.0000 0.0116
1..5 concentration histogram 6..10 concentration histogram	0.0116 0.0000 0.0000 0.0000 0.0233 0.0000 0.0116 0.0000 0.0000 0.0000
non computable conc. histogram	0.0116
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.0116 0.0000 0.0233 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0233
1..5 correlation histogram 6..10 correlation histogram	0.0116 0.0116 0.0116 0.0116 0.0116 0.0233 0.0233 0.0349 0.0116 0.0465
non computable correlation histogram	0.0116
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0116 0.0349 0.0233 0.0349 0.0000 0.0349 0.0000 0.0233 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0233
Mean Skew	0.0116
Mean Kurtosis	0.0000
Class Entropy	0.0581
Mean Attribute Entropy	0.0465
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0233
Noise to Signal Ratio	0.0233
Mean Mult. Correl. Coef.	0.0233
SDratio	0.0116

Table A.52. IBL Lindiscr

Attribute	
# classes	0.0700
# attributes	0.0700
# instances	0.0000
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0400
# unknown values	0.0500
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0100
# nominal attributes	0.0100
max, min, mean, stdv of nominal attribute values	0.0200 0.0200 0.0200 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0100 0.0000 0.0000 0.0100 0.0200 0.0000 0.0000 0.0000 0.0000 0.0100
non computable conc. histogram	0.0100
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0200 0.0000 0.0000 0.0000 0.0100 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0200
1..5 correlation histogram 6..10 correlation histogram	0.0500 0.0400 0.0000 0.0000 0.0100 0.0000 0.0500 0.0100 0.0100 0.0400
non computable correlation histogram	0.0100
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0600 0.0300 0.0400 0.0100 0.0400 0.0000 0.0100 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0100
Frac1	0.0100
First Canonical Correlation	0.0100
Mean Skew	0.0200
Mean Kurtosis	0.0200
Class Entropy	0.0600
Mean Attribute Entropy	0.0100
Mean Mutual Information	0.0100
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0100
SDratio	0.0100

Table A.53. IBL Ltree

Attribute	
# classes	0.0588
# attributes	0.0471
# instances	0.0588
$\frac{\# \text{ attributes}}{\# \text{instances}}$	0.0471
# unknown values	0.0824
$\frac{\# \text{ unknown values}}{\# \text{ attributes} * \# \text{ instances}}$	0.0118
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0118 0.0000 0.0353 0.0235
1..5 concentration histogram 6..10 concentration histogram	0.0235 0.0118 0.0000 0.0000 0.0000 0.0000 0.0118 0.0000 0.0000 0.0118
non computable conc. histogram	0.0118
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0118 0.0000 0.0118 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0118
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0000 0.0235 0.0118 0.0118 0.0118 0.0118 0.0118 0.0235 0.0118
non computable correlation histogram	0.0235
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0588 0.0235 0.0353 0.0235 0.0000 0.0000 0.0000 0.0000
$\frac{\# \text{ continuous}}{\# \text{ attributes}}$	0.0000
$\frac{\# \text{ nominal}}{\# \text{ attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0235
First Canonical Correlation	0.0471
Mean Skew	0.0118
Mean Kurtosis	0.0235
Class Entropy	0.0353
Mean Attribute Entropy	0.0353
Mean Mutual Information	0.0118
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0353
SDratio	0.0118

Table A.54. NB c50boost

Attribute	
# classes	0.0758
# attributes	0.0606
# instances	0.0455
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.1212
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0303
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0152 0.0152 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0000 0.0000 0.0000 0.0152 0.0000 0.0000 0.0152 0.0000 0.0152
non computable conc. histogram	0.0152
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0303 0.0000 0.0152 0.0303 0.0000 0.0000 0.0000 0.0000 0.0000 0.0303
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0152 0.0000 0.0000 0.0303 0.0152 0.0152 0.0152 0.0455 0.0152 0.0152
non computable correlation histogram	0.0152
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0152 0.0152 0.0000 0.0152 0.0152 0.0152 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0152
Frac1	0.0000
First Canonical Correlation	0.0606
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0455
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0152
Equivalent number of attributes	0.0455
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0303

Table A.55. NB c50rules

Attribute	
# classes	0.0513
# attributes	0.0256
# instances	0.0513
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0256
# unknown values	0.0769
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0256
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0000 0.0000 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0256 0.0128 0.0128 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0385 0.0256 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0256
1..5 correlation histogram 6..10 correlation histogram	0.0256 0.0128 0.0256 0.0128 0.0256 0.0128 0.0000 0.0128 0.0256 0.0000
non computable correlation histogram	0.0256
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0128 0.0385 0.0769 0.0641 0.0128 0.0000 0.0000 0.0256 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0256
Binary Attributes	0.0000
Frac1	0.0385
First Canonical Correlation	0.0000
Mean Skew	0.0385
Mean Kurtosis	0.0256
Class Entropy	0.0000
Mean Attribute Entropy	0.0256
Mean Mutual Information	0.0128
Equivalent number of attributes	0.0128
Noise to Signal Ratio	0.0128
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table A.56. NB c50tree

Attribute	
# classes	0.0405
# attributes	0.0405
# instances	0.0270
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0135
# unknown values	0.0946
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0135
max, min, mean, stdv of nominal attribute values	0.0270 0.0270 0.0135 0.0000
1..5 concentration histogram	0.0135 0.0000 0.0000 0.0000 0.0000
6..10 concentration histogram	0.0135 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0135
1..5 concentration histogram with class	0.0000 0.0405 0.0000 0.0135 0.0135
6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0270
1..5 correlation histogram	0.0135 0.0135 0.0135 0.0000 0.0270
6..10 correlation histogram	0.0135 0.0270 0.0000 0.0270 0.0270
non computable correlation histogram	0.0000
1..5 missing values histogram	0.0000 0.0135 0.0135 0.0405 0.0405
6..10 missing values histogram	0.0270 0.0135 0.0000 0.0270 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0541
Mean Skew	0.0270
Mean Kurtosis	0.0135
Class Entropy	0.0135
Mean Attribute Entropy	0.0135
Mean Mutual Information	0.0270
Equivalent number of attributes	0.0135
Noise to Signal Ratio	0.0270
Mean Mult. Correl. Coef.	0.0270
SDratio	0.0135

Table A.57. NB Lindiscr

Attribute	
# classes	0.0549
# attributes	0.0220
# instances	0.0330
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0879
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0220
# nominal attributes	0.0220
max, min, mean, stdv of nominal attribute values	0.0110 0.0220 0.0330 0.0110
1..5 concentration histogram 6..10 concentration histogram	0.0330 0.0110 0.0000 0.0000 0.0110 0.0110 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0110
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0220 0.0220 0.0220 0.0000 0.0110 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0220 0.0330 0.0220 0.0110 0.0220 0.0220 0.0000 0.0110 0.0330 0.0110
non computable correlation histogram	0.0110
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0110 0.0220 0.0440 0.0330 0.0000 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0220
Frac1	0.0000
First Canonical Correlation	0.0220
Mean Skew	0.0769
Mean Kurtosis	0.0220
Class Entropy	0.0000
Mean Attribute Entropy	0.0330
Mean Mutual Information	0.0220
Equivalent number of attributes	0.0110
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0110

Table A.58. NB Ltree

Attribute	
# classes	0.0541
# attributes	0.0811
# instances	0.0135
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0270
# unknown values	0.1081
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0270
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0541 0.0135 0.0135 0.0135
1..5 concentration histogram	0.0000 0.0000 0.0000 0.0000 0.0135
6..10 concentration histogram	0.0000 0.0000 0.0135 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class	0.0000 0.0405 0.0000 0.0135 0.0135
6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.0000 0.0135
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram	0.0135 0.0135 0.0270 0.0000 0.0405
6..10 correlation histogram	0.0000 0.0405 0.0000 0.0000 0.0000
non computable correlation histogram	0.0270
1..5 missing values histogram	0.0000 0.0000 0.0270 0.0135 0.0135
6..10 missing values histogram	0.0000 0.0000 0.0000 0.0135 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0135
Binary Attributes	0.0270
Frac1	0.0000
First Canonical Correlation	0.0405
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0541
Mean Attribute Entropy	0.0270
Mean Mutual Information	0.0135
Equivalent number of attributes	0.0270
Noise to Signal Ratio	0.0135
Mean Mult. Correl. Coef.	0.0135
SDratio	0.0270

Table A.59. NB IBL

Attribute	
# classes	0.0444
# attributes	0.0111
# instances	0.0333
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0444
# unknown values	0.0778
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0111
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0222 0.0111 0.0222 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0111 0.0000 0.0000 0.0000 0.0111 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0333 0.0111 0.0111 0.0111 0.0000 0.0000 0.0000 0.0000 0.0000 0.0111
non computable conc. histogram with class	0.0000
# continuous attributes	0.0111
1..5 correlation histogram 6..10 correlation histogram	0.0222 0.0000 0.0111 0.0000 0.0111 0.0333 0.0444 0.0111 0.0111 0.0222
non computable correlation histogram	0.0111
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0111 0.0444 0.0667 0.0444 0.0222 0.0111 0.0000 0.0111 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0111
Frac1	0.0111
First Canonical Correlation	0.0667
Mean Skew	0.0111
Mean Kurtosis	0.0111
Class Entropy	0.0222
Mean Attribute Entropy	0.0333
Mean Mutual Information	0.0222
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0111
SDratio	0.0222

Table A.60. ripper c50boost

Attribute	
# classes	0.0299
# attributes	0.0299
# instances	0.0597
$\frac{\#\text{ attributes}}{\#\text{instances}}$	0.0000
# unknown values	0.0746
$\frac{\#\text{ unknown values}}{\#\text{ attributes * }\#\text{ instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0149 0.0149 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0299 0.0149 0.0149 0.0149 0.0000 0.0000 0.0000 0.0000 0.0000 0.0149
non computable conc. histogram	0.0149
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.0299 0.0149 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0299
1..5 correlation histogram 6..10 correlation histogram	0.0149 0.0448 0.0000 0.0000 0.0149 0.0000 0.0448 0.0000 0.0149 0.0597
non computable correlation histogram	0.0149
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0448 0.0448 0.0299 0.0299 0.0149 0.0000 0.0000 0.0149 0.0000
$\frac{\#\text{ continuous}}{\#\text{ attributes}}$	0.0000
$\frac{\#\text{ nominal}}{\#\text{ attributes}}$	0.0000
Binary Attributes	0.0149
Frac1	0.0448
First Canonical Correlation	0.0000
Mean Skew	0.0299
Mean Kurtosis	0.0149
Class Entropy	0.0746
Mean Attribute Entropy	0.0299
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table A.61. ripper c50rules

Attribute	
# classes	0.0361
# attributes	0.0361
# instances	0.0361
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0120
# unknown values	0.0723
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0361
# nominal attributes	0.0120
max, min, mean, stdv of nominal attribute values	0.0241 0.0000 0.0120 0.0241
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0120 0.0000 0.0000 0.0000 0.0000 0.0120 0.0000 0.0000 0.0241
non computable conc. histogram	0.0120
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0482 0.0241 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0120
1..5 correlation histogram 6..10 correlation histogram	0.0120 0.0000 0.0120 0.0000 0.0000 0.0241 0.0120 0.0120 0.0241 0.0241
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0361 0.0000 0.0361 0.0120 0.0241 0.0120 0.0241 0.0120
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0120
Mean Skew	0.0361
Mean Kurtosis	0.0000
Class Entropy	0.0482
Mean Attribute Entropy	0.0602
Mean Mutual Information	0.0482
Equivalent number of attributes	0.0120
Noise to Signal Ratio	0.0120
Mean Mult. Correl. Coef.	0.0241
SDratio	0.0241

Table A.62. ripper c50tree

Attribute	
# classes	0.0141
# attributes	0.0704
# instances	0.0282
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0282
# unknown values	0.0563
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0282
# nominal attributes	0.0141
max, min, mean, stdv of nominal attribute values	0.0000 0.0141 0.0000 0.0282
1..5 concentration histogram 6..10 concentration histogram	0.0141 0.0000 0.0141 0.0000 0.0000 0.0000 0.0141 0.0000 0.0000 0.0141
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0282 0.0141 0.0000 0.0141 0.0141 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0141
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0563 0.0000 0.0000 0.0000 0.0282 0.0141 0.0282 0.0282 0.0000
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0282 0.0423 0.0141 0.0282 0.0282 0.0141 0.0000 0.0141 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0141
Binary Attributes	0.0141
Frac1	0.0141
First Canonical Correlation	0.0141
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0845
Mean Attribute Entropy	0.0423
Mean Mutual Information	0.0423
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0141
SDratio	0.0141

Table A.63. ripper Lindiscr

Attribute	
# classes	0.0230
# attributes	0.0345
# instances	0.0575
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0115
# unknown values	0.0460
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0230
max, min, mean, stdv of nominal attribute values	0.0575 0.0115 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0115 0.0115 0.0000 0.0230 0.0115 0.0000 0.0000 0.0000 0.0000 0.0115
non computable conc. histogram	0.0115
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0115 0.0115 0.0000 0.0115 0.0000 0.0000 0.0000 0.0000 0.0000 0.0115
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0115 0.0230 0.0000 0.0575 0.0000 0.0000 0.0345 0.0115 0.0230
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0805 0.0460 0.0575 0.0345 0.0115 0.0115 0.0000 0.0115 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0115
Frac1	0.0115
First Canonical Correlation	0.0230
Mean Skew	0.0345
Mean Kurtosis	0.0115
Class Entropy	0.0230
Mean Attribute Entropy	0.0115
Mean Mutual Information	0.0115
Equivalent number of attributes	0.0115
Noise to Signal Ratio	0.0115
Mean Mult. Correl. Coef.	0.0115
SDratio	0.0230

Table A.64. ripper Ltree

Attribute	
# classes	0.0370
# attributes	0.0247
# instances	0.0370
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0247
# unknown values	0.0247
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0123
# nominal attributes	0.0247
max, min, mean, stdv of nominal attribute values	0.0000 0.0000 0.0247 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0123 0.0247 0.0000 0.0123 0.0000 0.0000 0.0123 0.0000 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0123 0.0123 0.0123 0.0123 0.0000 0.0000 0.0000 0.0000 0.0000 0.0123
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0123 0.0247 0.0123 0.0370 0.0123 0.0123 0.0123 0.0000 0.0123 0.0247
non computable correlation histogram	0.0247
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0494 0.0123 0.0494 0.0123 0.0123 0.0123 0.0000 0.0247 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0247
Frac1	0.0123
First Canonical Correlation	0.0370
Mean Skew	0.0000
Mean Kurtosis	0.0247
Class Entropy	0.0617
Mean Attribute Entropy	0.0247
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0494
Noise to Signal Ratio	0.0247
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0494

Table A.65. ripper IBL

Attribute	
# classes	0.0727
# attributes	0.0545
# instances	0.0727
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0545
# unknown values	0.0364
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0182 0.0182 0.0000 0.0182
1..5 concentration histogram 6..10 concentration histogram	0.0182 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0182
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0182 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0364
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0364 0.0000 0.0364 0.0000 0.0545 0.0000 0.0000 0.0000 0.0182
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0364 0.0364 0.0182 0.0000 0.0182 0.0182 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0364
Frac1	0.0000
First Canonical Correlation	0.0364
Mean Skew	0.0182
Mean Kurtosis	0.0182
Class Entropy	0.1091
Mean Attribute Entropy	0.0182
Mean Mutual Information	0.0182
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0182
SDratio	0.0182

Table A.66. ripper NB

Attribute	
# classes	0.0435
# attributes	0.0217
# instances	0.0652
$\frac{\# \text{ attributes}}{\# \text{instances}}$	0.0217
# unknown values	0.0761
$\frac{\# \text{ unknown values}}{\# \text{ attributes} * \# \text{ instances}}$	0.0109
# nominal attributes	0.0326
max, min, mean, stdv of nominal attribute values	0.0217 0.0217 0.0217 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0217 0.0217 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0109
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0109 0.0109 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0217 0.0109 0.0109 0.0109 0.0109 0.0326 0.0217 0.0326 0.0000 0.0109
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0109 0.0435 0.0217 0.0652 0.0217 0.0109 0.0000 0.0109 0.0109
$\frac{\# \text{ continuous}}{\# \text{ attributes}}$	0.0000
$\frac{\# \text{ nominal}}{\# \text{ attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0109
First Canonical Correlation	0.0217
Mean Skew	0.0109
Mean Kurtosis	0.0326
Class Entropy	0.0652
Mean Attribute Entropy	0.0217
Mean Mutual Information	0.0217
Equivalent number of attributes	0.0217
Noise to Signal Ratio	0.0109
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0109

A.5 Accuracy Results

Table A.67. Results on the *histo* set of characteristics

	c50boost		c50rules		c50tree		Ltree	
	Acc.	Impr.	Acc.	Impr.	Acc.	Impr.	Acc.	Impr.
c50rules c50boost	84.74%	26.61%	82.23%	24.09%	82.51%	24.37%	80.84%	22.70%
c50tree c50boost	82.88%	25.12%	79.26%	21.49%	79.35%	21.58%	79.16%	21.40%
c50tree c50rules	86.23%	13.21%	82.79%	9.77%	84.00%	10.98%	82.42%	9.40%
Lindiscr c50boost	87.72%	23.26%	86.60%	22.14%	87.35%	22.88%	84.74%	20.28%
Lindiscr c50rules	87.07%	34.42%	86.14%	33.49%	85.95%	33.30%	82.70%	30.05%
Lindiscr c50tree	87.26%	32.56%	85.86%	31.16%	86.23%	31.54%	84.09%	29.40%
Ltree c50boost	81.58%	29.95%	77.67%	26.05%	76.74%	25.12%	77.49%	25.86%
Ltree c50rules	84.09%	25.40%	82.60%	23.91%	81.67%	22.98%	79.91%	21.21%
Ltree c50tree	81.77%	22.23%	79.53%	20.00%	78.51%	18.98%	77.95%	18.42%
Ltree Lindiscr	84.56%	23.44%	81.95%	20.84%	81.95%	20.84%	80.84%	19.72%
IBL c50boost	87.53%	22.88%	86.70%	22.05%	85.49%	20.84%	84.28%	19.63%
IBL c50rules	81.02%	31.26%	79.16%	29.40%	78.42%	28.65%	79.26%	29.49%
IBL c50tree	84.19%	32.19%	78.98%	26.98%	78.98%	26.98%	80.65%	28.65%
IBL Lindiscr	85.30%	42.98%	80.28%	37.95%	79.72%	37.40%	81.12%	38.79%
IBL Ltree	81.86%	25.21%	77.49%	20.84%	78.05%	21.40%	76.56%	19.91%
NB c50boost	88.56%	27.72%	86.60%	25.77%	86.70%	25.86%	84.56%	23.72%
NB c50rules	88.28%	36.84%	82.60%	31.16%	82.60%	31.16%	80.56%	29.12%
NB c50tree	87.16%	34.23%	83.26%	30.33%	83.35%	30.42%	83.63%	30.70%
NB Lindiscr	84.65%	43.54%	80.47%	39.35%	80.19%	39.07%	78.98%	37.86%
NB Ltree	87.91%	29.67%	84.37%	26.14%	84.56%	26.33%	84.00%	25.77%
NB IBL	86.79%	50.70%	84.00%	47.91%	84.37%	48.28%	81.67%	45.58%
ripper c50boost	84.65%	33.68%	85.12%	34.14%	83.72%	32.74%	84.28%	33.30%
ripper c50rules	85.21%	25.30%	81.12%	21.21%	81.02%	21.12%	81.58%	21.68%
ripper c50tree	85.95%	25.86%	83.35%	23.26%	82.23%	22.14%	81.12%	21.02%
ripper Lindiscr	82.42%	36.28%	80.00%	33.86%	79.26%	33.12%	77.12%	30.98%
ripper Ltree	78.88%	25.67%	78.98%	25.77%	78.14%	24.93%	78.70%	25.49%
ripper IBL	86.33%	40.19%	83.53%	37.40%	82.70%	36.56%	83.91%	37.77%
ripper NB	84.28%	42.51%	81.40%	39.63%	82.42%	40.65%	80.84%	39.07%
means	84.96%	30.82%	82.22%	28.07%	82.01%	27.86%	81.18%	27.03%

Table A.68. Results on the *+histo* set of characteristics

	c50boost		c50rules		c50tree		Ltree	
	Acc.	Impr.	Acc.	Impr.	Acc.	Impr.	Acc.	Impr.
c50rules c50boost	83.26%	25.12%	81.95%	23.81%	81.95%	23.81%	80.74%	22.61%
c50tree c50boost	84.56%	26.79%	80.37%	22.61%	79.35%	21.58%	78.14%	20.37%
c50tree c50rules	86.14%	13.12%	82.05%	9.02%	82.51%	9.49%	82.42%	9.40%
Lindiscr c50boost	87.53%	23.07%	84.19%	19.72%	84.47%	20.00%	83.16%	18.70%
Lindiscr c50rules	87.35%	34.70%	84.74%	32.09%	85.30%	32.65%	83.26%	30.60%
Lindiscr c50tree	87.81%	33.12%	84.28%	29.58%	84.37%	29.68%	84.56%	29.86%
Ltree c50boost	81.02%	29.40%	76.56%	24.93%	76.93%	25.30%	77.30%	25.68%
Ltree c50rules	84.74%	26.05%	81.21%	22.51%	80.84%	22.14%	80.93%	22.23%
Ltree c50tree	81.67%	22.14%	79.91%	20.37%	79.44%	19.91%	78.23%	18.70%
Ltree Lindiscr	82.42%	21.30%	81.30%	20.19%	81.21%	20.09%	80.74%	19.63%
IBL c50boost	85.67%	21.02%	85.86%	21.21%	85.40%	20.74%	83.26%	18.60%
IBL c50rules	81.58%	31.81%	79.26%	29.49%	78.88%	29.12%	77.95%	28.19%
IBL c50tree	83.16%	31.16%	79.53%	27.53%	78.79%	26.79%	80.74%	28.74%
IBL Lindiscr	84.65%	42.33%	80.56%	38.23%	79.53%	37.21%	81.86%	39.54%
IBL Ltree	80.56%	23.91%	76.74%	20.09%	77.30%	20.65%	75.53%	18.88%
NB c50boost	88.00%	27.16%	86.60%	25.77%	85.95%	25.12%	84.28%	23.44%
NB c50rules	86.51%	35.07%	81.30%	29.86%	81.77%	30.33%	81.77%	30.33%
NB c50tree	86.14%	33.21%	83.07%	30.14%	82.79%	29.86%	83.26%	30.33%
NB Lindiscr	83.81%	42.70%	80.65%	39.54%	80.37%	39.26%	80.09%	38.98%
NB Ltree	86.98%	28.74%	84.37%	26.14%	84.19%	25.95%	82.51%	24.28%
NB IBL	87.07%	50.98%	83.91%	47.81%	85.40%	49.30%	79.35%	43.26%
ripper c50boost	86.05%	35.07%	85.30%	34.33%	84.28%	33.30%	85.02%	34.05%
ripper c50rules	84.28%	24.37%	80.56%	20.65%	80.93%	21.02%	80.09%	20.19%
ripper c50tree	85.95%	25.86%	82.14%	22.05%	82.33%	22.23%	83.81%	23.72%
ripper Lindiscr	82.05%	35.91%	78.98%	32.84%	78.05%	31.91%	77.95%	31.81%
ripper Ltree	77.86%	24.65%	78.33%	25.12%	76.74%	23.54%	77.58%	24.37%
ripper IBL	86.33%	40.19%	83.81%	37.67%	83.26%	37.12%	84.28%	38.14%
ripper NB	85.02%	43.26%	81.49%	39.72%	81.49%	39.72%	80.65%	38.88%
means	84.58%	30.44%	81.75%	27.61%	81.56%	27.42%	81.05%	26.91%

Table A.69. Results of the McNemar test comparing the accuracies of *histo* and +*histo*, on the pairwise meta-learning problems, for each of the four decision treee based metalearners. (= no difference, + difference in favor of *histo*, - difference in favor of +*histo*.

	c50boost				c50rules				c50tree				Ltree			
	$-an$	$+an$	$p-val$	sig	$-an$	$+an$	$p-val$	sig	$-an$	$+an$	$p-val$	sig	$-an$	$+an$	$p-val$	sig
c50rules c50boost	46	30	0.085	=	43	40	0.826	=	46	40	0.589	=	48	47	1.000	=
c50tree c50boost	35	53	0.069	=	53	65	0.311	=	65	65	0.930	=	65	54	0.359	=
c50tree c50rules	29	28	1.000	=	45	37	0.439	=	46	30	0.085	=	46	46	0.916	=
Lindiscr c50boost	29	27	0.893	=	67	41	0.016	+	65	34	0.002	+	82	65	0.186	=
Lindiscr c50rules	32	35	0.806	=	55	40	0.150	=	50	43	0.533	=	64	70	0.665	=
Lindiscr c50tree	34	40	0.561	=	53	36	0.089	=	50	30	0.033	+	55	60	0.709	=
Ltree c50boost	42	36	0.571	=	62	50	0.298	=	56	58	0.995	=	54	52	0.922	=
Ltree c50rules	33	40	0.482	=	60	45	0.171	=	59	50	0.443	=	47	58	0.329	=
Ltree c50tree	48	47	1.000	=	66	70	0.796	=	57	67	0.418	=	78	81	0.873	=
Ltree Lindiscr	66	43	0.035	+	63	56	0.582	=	58	50	0.500	=	66	65	1.000	=
IBL c50boost	48	28	0.029	+	45	36	0.374	=	39	38	1.000	=	56	45	0.319	=
IBL c50rules	38	44	0.580	=	51	52	1.000	=	48	53	0.690	=	71	57	0.250	=
IBL c50tree	45	34	0.260	=	49	55	0.623	=	42	40	0.912	=	58	59	1.000	=
IBL Lindiscr	41	34	0.488	=	52	55	0.846	=	50	48	0.919	=	48	56	0.492	=
IBL Ltree	57	43	0.193	=	60	52	0.508	=	57	49	0.496	=	79	68	0.409	=
NB c50boost	25	19	0.450	=	49	49	0.919	=	53	45	0.479	=	66	63	0.860	=
NB c50rules	43	24	0.027	+	46	32	0.141	=	41	32	0.349	=	59	72	0.294	=
NB c50tree	38	27	0.214	=	40	38	0.909	=	38	32	0.550	=	55	51	0.770	=
NB Lindiscr	49	40	0.396	=	37	39	0.908	=	37	39	0.908	=	85	97	0.414	=
NB Ltree	30	20	0.203	=	41	41	0.912	=	44	40	0.743	=	58	42	0.133	=
NB IBL	37	40	0.819	=	62	61	1.000	=	51	62	0.346	=	94	69	0.060	=
ripper c50boost	32	47	0.115	=	26	28	0.891	=	26	32	0.511	=	34	42	0.422	=
ripper c50rules	43	33	0.301	=	35	29	0.531	=	30	29	1.000	=	61	45	0.145	=
ripper c50tree	29	29	0.895	=	37	24	0.124	=	28	29	1.000	=	21	50	0.001	−
ripper Lindiscr	54	50	0.768	=	68	57	0.371	=	76	63	0.308	=	68	77	0.506	=
ripper Ltree	54	43	0.309	=	56	49	0.558	=	51	36	0.133	=	63	51	0.302	=
ripper IBL	26	26	0.889	=	34	37	0.812	=	36	42	0.571	=	48	52	0.764	=
ripper NB	34	42	0.422	=	59	60	1.000	=	57	47	0.377	=	71	69	0.932	=

Appendix B

Results on the 65 datasets

B.1 The 65 datasets

abalone, acetylation, agaricus-lepiota, allbp, allhyper, allhypo, allrep, australian, balance-scale, bands, breast-cancer-wisc,breast-cancer-wisc_nominal, bupa, car, contraceptive, crx, dermatology, dis, ecoli, flag_language, flag_religion, flare_c, flare_c_er, flare_m, flare_m_er, flare_x, flare_x_er, fluid, german, glass, glass2, heart, hepatitis, hypothyroid, ionosphere, iris, kp, led24, led7, lymphography, monk1, monk2, monk3-full, mushrooms, new-thyroid, parity5_5, pima-indians-diabetes, proc-cleveland-2, proc-cleveland-4, proc-hungarian-2, proc -hungarian-4, proc-switzerland-2, proc-switzerland-4, quisclas, sick-euthyroid, soybean-large, tic-tac-toe, titanic, tumor-LOI, vote, vowel, waveform40, wdbc, wpbc, yeast.

B.2 C50boost on the different characterizations

Table B.1. Results of c50boost on the *dct* characterization on the 65 datasets

Pair	Accuracy	Improvement
c50rules c50boost	72.31%	-4.62%
c50tree c50boost	60.00%	-6.15%
c50tree c50rules	73.85%	-3.08%
Lindiscr c50boost	64.62%	15.39%
Lindiscr c50rules	55.38%	9.23%
Lindiscr c50tree	49.23%	4.62%
Ltree c50boost	61.54%	1.54%
Ltree c50rules	72.31%	6.15%
Ltree c50tree	63.08%	7.69%
Ltree Lindiscr	64.62%	20.00%
IBL c50boost	70.77%	12.31%
IBL c50rules	64.62%	6.15%
IBL c50tree	67.69%	10.77%
IBL Lindiscr	41.54%	3.08%
IBL Ltree	49.23%	6.15%
NB c50boost	63.08%	9.23%
NB c50rules	66.15%	16.92%
NB c50tree	56.92%	7.69%
NB Lindiscr	47.69%	4.62%
NB Ltree	69.23%	18.46%
NB IBL	43.08%	-1.54%
ripper c50boost	49.23%	-13.85%
ripper c50rules	66.15%	-3.08%
ripper c50tree	60.00%	-6.15%
ripper Lindiscr	60.00%	21.54%
ripper Ltree	58.46%	3.08%
ripper IBL	43.08%	-10.77%
ripper NB	53.85%	7.69%
Average	59.56%	5.11%
Strict Accuracy	6.15%	-7.69%
Loose Accuracy	43.08%	%

Table B.2. *land*

Pair	Accuracy	Improvement
c50rules c50boost	75.38%	-1.54%
c50tree c50boost	69.23%	3.08%
c50tree c50rules	72.31%	-4.62%
Lindiscr c50boost	44.62%	-4.61%
Lindiscr c50rules	47.69%	1.54%
Lindiscr c50tree	43.08%	-1.54%
Ltree c50boost	55.38%	-4.62%
Ltree c50rules	52.31%	-13.85%
Ltree c50tree	61.54%	6.15%
Ltree Lindiscr	50.77%	6.15%
IBL c50boost	52.31%	-6.15%
IBL c50rules	61.54%	3.08%
IBL c50tree	44.62%	-12.31%
IBL Lindiscr	41.54%	3.08%
IBL Ltree	47.69%	4.62%
NB c50boost	55.38%	1.54%
NB c50rules	53.85%	4.62%
NB c50tree	46.15%	-3.08%
NB Lindiscr	47.69%	4.62%
NB Ltree	60.00%	9.23%
NB IBL	44.62%	0.00%
ripper c50boost	56.92%	-6.15%
ripper c50rules	55.38%	-13.85%
ripper c50tree	52.31%	-13.85%
ripper Lindiscr	46.15%	7.69%
ripper Ltree	52.31%	-3.08%
ripper IBL	43.08%	-10.77%
ripper NB	43.08%	-3.08%
Average	52.75%	-1.70%
Strict Accuracy	7.69%	-6.15%
Loose Accuracy	36.92%	%

Table B.3. *histo-limited*

Pair	Accuracy	Improvement
c50rules c50boost	72.31%	-4.62%
c50tree c50boost	52.31%	-13.85%
c50tree c50rules	72.31%	-4.62%
Lindiscr c50boost	55.38%	6.15%
Lindiscr c50rules	55.38%	9.23%
Lindiscr c50tree	64.62%	20.00%
Ltree c50boost	53.85%	-6.15%
Ltree c50rules	66.15%	0.00%
Ltree c50tree	67.69%	12.31%
Ltree Lindiscr	64.62%	20.00%
IBL c50boost	73.85%	15.39%
IBL c50rules	61.54%	3.08%
IBL c50tree	55.38%	-1.54%
IBL Lindiscr	58.46%	20.00%
IBL Ltree	40.00%	-3.08%
NB c50boost	72.31%	18.46%
NB c50rules	49.23%	0.00%
NB c50tree	49.23%	0.00%
NB Lindiscr	43.08%	0.00%
NB Ltree	67.69%	16.92%
NB IBL	49.23%	4.62%
ripper c50boost	46.15%	-16.92%
ripper c50rules	58.46%	-10.77%
ripper c50tree	56.92%	-9.23%
ripper Lindiscr	60.00%	21.54%
ripper Ltree	46.15%	-9.23%
ripper IBL	47.69%	-6.15%
ripper NB	58.46%	12.31%
Average	57.80%	3.35%
Strict Accuracy	20.00%	6.16%%
Loose Accuracy	47.69%	%

Table B.4. *histo*

Pair	Accuracy	Improvement
c50rules c50boost	76.92%	0.00%
c50treec 50boost	53.85%	-12.31%
c50treec 50rules	69.23%	-7.69%
Lindiscr c50boost	58.46%	9.23%
Lindiscr c50rules	61.54%	15.39%
Lindiscr c50tree	64.62%	20.00%
Ltree c50boost	52.31%	-7.69%
Ltree c50rules	66.15%	0.00%
Ltree c50tree	66.15%	10.77%
Ltree Lindiscr	61.54%	16.92%
IBL c50boost	69.23%	10.77%
IBL c50rules	64.62%	6.15%
IBL c50tree	60.00%	3.08%
IBL Lindiscr	64.62%	26.15%
IBL Ltree	36.92%	-6.15%
NB c50boost	69.23%	15.38%
NB c50rules	53.85%	4.62%
NB c50tree	49.23%	0.00%
NB Lindiscr	41.54%	-1.54%
NB Ltree	69.23%	18.46%
NB IBL	58.46%	13.85%
ripper c50boost	64.62%	1.54%
ripper c50rules	55.38%	-13.85%
ripper c50tree	49.23%	-16.92%
ripper Lindiscr	64.62%	26.15%
ripper Ltree	52.31%	-3.08%
ripper IBL	47.69%	-6.15%
ripper NB	55.38%	9.23%
Average	59.18%	4.73%
Strict Accuracy	15.38%	1.54%
Loose Accuracy	52.31%	%

Table B.5. *statlog*

Pair	Accuracy	Improvement
c50rules c50boost	73.85%	-3.08%
c50tree c50boost	55.38%	-10.77%
c50tree c50rules	73.85%	-3.08%
Lindiscr c50boost	53.85%	4.62%
Lindiscr c50rules	56.92%	10.77%
Lindiscr c50tree	53.85%	9.23%
Ltree c50boost	60.00%	0.00%
Ltree c50rules	64.62%	-1.54%
Ltree c50tree	61.54%	6.15%
Ltree Lindiscr	61.54%	16.92%
IBL c50boost	64.62%	6.15%
IBL c50rules	69.23%	10.77%
IBL c50tree	61.54%	4.62%
IBL Lindiscr	43.08%	4.62%
IBL Ltree	47.69%	4.62%
NB c50boost	55.38%	1.54%
NB c50rules	50.77%	1.54%
NB c50tree	56.92%	7.69%
NB Lindiscr	38.46%	-4.61%
NB Ltree	75.38%	24.62%
NB IBL	49.23%	4.62%
ripper c50boost	50.77%	-12.31%
ripper c50rules	66.15%	-3.08%
ripper c50tree	60.00%	-6.15%
ripper Lindiscr	53.85%	15.39%
ripper Ltree	50.77%	-4.61%
ripper IBL	49.23%	-4.62%
ripper NB	41.54%	-4.61%
Average	57.14%	2.69%
Strict Accuracy	7.69%	-6.15%
Loose Accuracy	41.54%	%

B.3 Kernel on the different characterizations

Table B.6. Results of kernel on the *dct* characterizations on the 65 datasets

Pair	Accuracy	Improvement
c50boost c50rules	70.77%	3.08%
c50boost c50tree	70.77%	-4.61%
c50boost Lindiscr	61.54%	-7.69%
c50boost Ltree	64.62%	7.69%
c50boost IBL	90.77%	-1.54%
c50boost NB	75.38%	1.54%
c50boost ripper	76.92%	-6.15%
c50rules c50tree	73.85%	1.54%
c50rules Lindiscr	69.23%	6.15%
c50rules Ltree	63.08%	0.00%
c50rules IBL	72.31%	1.54%
c50rules NB	75.38%	7.69%
c50rules ripper	75.38%	-1.54%
c50tree Lindiscr	66.15%	3.08%
c50tree Ltree	63.08%	4.62%
c50tree IBL	69.23%	0.00%
c50tree NB	80.00%	12.31%
c50tree ripper	55.38%	-13.85%
Lindiscr ltree	73.85%	1.54%
Lindiscr IBL	66.15%	12.31%
Lindiscr NB	73.85%	18.46%
Lindiscr ripper	64.62%	9.23%
Ltree IBL	69.23%	-9.23%
Ltree NB	67.69%	-6.15%
Ltree ripper	81.54%	9.23%
IBL NB	69.23%	12.31%
IBL ripper	60.00%	7.69%
NB ripper	70.77%	15.39%
Average	70.38%	3.02%
Final suggestion	40.00%	7.70%

Table B.7. *land*

Pair	Accuracy	Improvement
c50boost c50rules	64.62%	-3.08%
c50boost c50tree	73.85%	-1.54%
c50boost Lindiscr	63.08%	-6.15%
c50boost Ltree	52.31%	-4.62%
c50boost IBL	90.77%	-1.54%
c50boost IBL	70.77%	-3.08%
c50boost ripper	70.77%	-12.31%
c50rules c50tree	52.31%	-20.00%
c50rules Lindiscr	58.46%	-4.61%
c50rules Ltree	43.08%	-20.00%
c50rules IBL	61.54%	-9.23%
c50rules NB	60.00%	-7.69%
c50rules ripper	73.85%	-3.08%
c50tree Lindiscr	56.92%	-6.15%
c50tree Ltree	44.62%	-13.85%
c50tree IBL	61.54%	-7.69%
c50tree NB	55.38%	-12.31%
c50tree ripper	60.00%	-9.23%
Lindiscr Ltree	61.54%	-10.77%
Lindiscr IBL	44.62%	-9.23%
Lindiscr NB	64.62%	9.23%
Lindiscr ripper	55.38%	0.00%
Ltree IBL	61.54%	-16.92%
Ltree NB	76.92%	3.08%
Ltree ripper	64.62%	-7.69%
IBL NB	60.00%	3.08%
IBL ripper	49.23%	-3.08%
NB ripper	58.46%	3.08%
Average	61.10%	-6.26%
Final Suggestion	20.00%	-12.30%

Table B.8. *histo-limited*

Pair	Accuracy	Improvement
c50boost c50rules	69.23%	1.54%
c50boost c50tree	73.85%	-1.54%
c50boost Lindiscr	64.62%	-4.61%
c50boost Ltree	70.77%	13.85%
c50boost IBL	89.23%	-3.08%
c50boost NB	76.92%	3.08%
c50boost ripper	80.00%	-3.08%
c50rules c50tree	73.85%	1.54%
c50rules Lindiscr	61.54%	-1.54%
c50rules Ltree	66.15%	3.08%
c50rules IBL	73.85%	3.08%
c50rules NB	70.77%	3.08%
c50rules ripper	72.31%	-4.62%
c50tree Lindiscr	67.69%	4.62%
c50tree Ltree	64.62%	6.15%
c50tree IBL	70.77%	1.54%
c50tree NB	67.69%	0.00%
c50tree ripper	63.08%	-6.15%
Lindiscr Ltree	78.46%	6.15%
Lindiscr IBL	64.62%	10.77%
Lindiscr NB	76.92%	21.54%
Lindiscr ripper	61.54%	6.15%
Ltree IBL	60.00%	-18.46%
Ltree NB	75.38%	1.54%
Ltree ripper	69.23%	-3.08%
IBL NB	61.54%	4.62%
IBL ripper	55.38%	3.08%
NB ripper	64.62%	9.23%
Average	69.45%	2.09%
Final Suggestion	36.92%	4.62%

Table B.9. *histo*

Pair	Accuracy	Improvement
c50boost c50rules	78.46%	10.77%
c50boost c50tree	72.31%	-3.08%
c50boost Lindiscr	66.15%	-3.08%
c50boost Ltree	67.69%	10.77%
c50boost IBL	89.23%	-3.08%
c50boost NB	80.00%	6.15%
c50boost ripper	81.54%	-1.54%
c50rules c50tree	75.38%	3.08%
c50rules Lindiscr	66.15%	3.08%
c50rules Ltree	63.08%	0.00%
c50rules IBL	61.54%	-9.23%
c50rules NB	70.77%	3.08%
c50rules ripper	72.31%	-4.62%
c50tree Lindiscr	69.23%	6.15%
c50tree Ltree	60.00%	1.54%
c50tree IBL	66.15%	-3.08%
c50tree NB	67.69%	0.00%
c50tree ripper	61.54%	-7.69%
Lindiscr Ltree	80.00%	7.69%
Lindiscr IBL	67.69%	13.85%
Lindiscr NB	76.92%	21.54%
Lindiscr ripper	60.00%	4.62%
Ltree IBL	56.92%	-21.54%
Ltree NB	75.38%	1.54%
Ltree ripper	67.69%	-4.61%
IBL NB	66.15%	9.23%
IBL ripper	58.46%	6.15%
NB ripper	61.54%	6.15%
Average	69.29%	1.92%
Final Suggestion	47.69%	15.30%

Table B.10. *statlog*

Pair	Accuracy	Improvement
c50boost c50rules	67.69%	0.00%
c50boost c50tree	69.23%	-6.15%
c50boost Lindiscr	53.85%	-15.38%
c50boost Ltree	61.54%	4.62%
c50boost IBL	90.77%	-1.54%
c50boost NB	76.92%	3.08%
c50boost ripper	81.54%	-1.54%
c50rules c50tree	73.85%	1.54%
c50rules Lindiscr	58.46%	-4.61%
c50rules Ltree	61.54%	-1.54%
c50rules IBL	63.08%	-7.69%
c50rules NB	73.85%	6.15%
c50rules ripper	76.92%	0.00%
c50tree Lindiscr	64.62%	1.54%
c50tree Ltree	66.15%	7.69%
c50tree IBL	69.23%	0.00%
c50tree NB	70.77%	3.08%
c50tree ripper	63.08%	-6.15%
Lindiscr Ltree	72.31%	0.00%
Lindiscr IBL	58.46%	4.62%
Lindiscr NB	67.69%	12.31%
Lindiscr ripper	61.54%	6.15%
Ltree IBL	63.08%	-15.38%
Ltree NB	75.38%	1.54%
Ltree ripper	67.69%	-4.61%
IBL NB	67.69%	10.77%
IBL ripper	61.54%	9.23%
NB ripper	70.77%	15.39%
Average	68.19%	0.82%
Final Suggestion	27.69%	-4.61%

B.4 Selection Frequency by c50boost

Table B.11. Selection Frequency of Characteristics by c50boost on the 65 datasets, pair : c50rules c50boost

Attribute	
# classes	0.0000
# attributes	0.0517
# instances	0.0517
$\frac{\# \text{attributes}}{\#\text{instances}}$	0.0517
# unknown values	0.0000
$\frac{\# \text{unknown values}}{\# \text{attributes} * \# \text{instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0517 0.0000 0.0172 0.0000
1..5 concentration histogram	0.0000 0.0000 0.0172 0.0000 0.0172
6..10 concentration histogram	0.0690 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class	0.1379 0.0000 0.0000 0.0000 0.0000
6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0172
1..5 correlation histogram	0.0000 0.0345 0.0000 0.0000 0.0172
6..10 correlation histogram	0.0172 0.0345 0.0345 0.0345 0.0172
non computable correlation histogram	0.0000
1..5 missing values histogram	0.0000 0.0172 0.0000 0.0345 0.0345
6..10 missing values histogram	0.0000 0.0000 0.0000 0.0000 0.0000
$\frac{\# \text{continuous}}{\# \text{attributes}}$	0.0000
$\frac{\# \text{nominal}}{\# \text{attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0345
First Canonical Correlation	0.0000
Mean Skew	0.0000
Mean Kurtosis	0.0172
Class Entropy	0.0345
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0345
Equivalent number of attributes	0.0690
Noise to Signal Ratio	0.0517
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.12. c50tree c50boost

Attribute	
# classes	0.0250
# attributes	0.0875
# instances	0.0125
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0125
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0125
max, min, mean, stdv of nominal attribute values	0.0125 0.0375 0.0000 0.0125
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0000 0.0125 0.0375 0.0000 0.0500 0.0000 0.0000 0.0250 0.0125
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.1000 0.0000 0.0000 0.0375 0.0000 0.0125 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0125 0.0000 0.0000 0.0000 0.0250 0.0000 0.0375 0.0250 0.0125 0.0250
non computable correlation histogram	0.0250
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0125 0.0125 0.0000 0.0000 0.0125 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0500
First Canonical Correlation	0.0125
Mean Skew	0.0250
Mean Kurtosis	0.0250
Class Entropy	0.0125
Mean Attribute Entropy	0.0125
Mean Mutual Information	0.0125
Equivalent number of attributes	0.0750
Noise to Signal Ratio	0.0750
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.13. c50tree c50rules

Attribute	
# classes	0.0000
# attributes	0.0926
# instances	0.0185
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0185
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0185 0.0370 0.0000 0.0185
1..5 concentration histogram 6..10 concentration histogram	0.0185 0.0000 0.0000 0.0185 0.0000 0.0741 0.0000 0.0556 0.0556 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0370 0.0556 0.0000 0.0370 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0556 0.0370 0.0000 0.0000 0.0185 0.0185 0.0741 0.0000 0.0185 0.0000
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0185 0.0556 0.0000 0.0000 0.0185 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0000
Mean Skew	0.0185
Mean Kurtosis	0.0000
Class Entropy	0.1111
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.14. Lindiscr c50boost

Attribute	
# classes	0.0303
# attributes	0.0202
# instances	0.0404
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0303
# unknown values	0.0202
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0101
max, min, mean, stdv of nominal attribute values	0.0202 0.0505 0.0101 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0505 0.0000 0.0404 0.0202 0.0101 0.0000 0.0303 0.0303 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0000 0.0404 0.0000 0.0202 0.0202 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0202
1..5 correlation histogram 6..10 correlation histogram	0.0303 0.0101 0.0000 0.0101 0.0101 0.0202 0.0101 0.0000 0.0202 0.0909
non computable correlation histogram	0.0101
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0101 0.0303 0.0101 0.0000 0.0404 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0202
First Canonical Correlation	0.0101
Mean Skew	0.0404
Mean Kurtosis	0.0202
Class Entropy	0.0000
Mean Attribute Entropy	0.0101
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0606
Noise to Signal Ratio	0.0202
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.15. Lindiscr c50rules

Attribute	
# classes	0.0125
# attributes	0.0250
# instances	0.0375
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0125
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0000 0.0000 0.0000 0.0125
1..5 concentration histogram 6..10 concentration histogram	0.0375 0.0125 0.0250 0.0000 0.0125 0.0125 0.0375 0.0000 0.0000 0.0250
non computable conc. histogram	0.0375
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0250 0.0000 0.0375 0.0125 0.0250 0.0125 0.0000 0.0125 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0375
1..5 correlation histogram 6..10 correlation histogram	0.0500 0.0250 0.0000 0.0375 0.0000 0.0125 0.0250 0.0000 0.0875 0.0250
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0125 0.0125 0.0000 0.0250 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0125
First Canonical Correlation	0.0375
Mean Skew	0.0375
Mean Kurtosis	0.0625
Class Entropy	0.0375
Mean Attribute Entropy	0.0125
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0250
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.16. Lindiscr c50tree

Attribute	
# classes	0.0519
# attributes	0.0000
# instances	0.0519
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0519
# unknown values	0.0260
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0000 0.0390 0.0000 0.0130
1..5 concentration histogram 6..10 concentration histogram	0.0260 0.0000 0.0000 0.0000 0.0130 0.0000 0.0260 0.0000 0.0000 0.0000
non computable conc. histogram	0.0130
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0130 0.0130 0.0260 0.0390 0.0130 0.0130 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0260
1..5 correlation histogram 6..10 correlation histogram	0.0130 0.0000 0.0260 0.0000 0.0260 0.0909 0.0000 0.0000 0.0260 0.0260
non computable correlation histogram	0.0390
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0390 0.0260 0.0000 0.0260 0.0000 0.0130 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0130
First Canonical Correlation	0.0519
Mean Skew	0.0130
Mean Kurtosis	0.0390
Class Entropy	0.0390
Mean Attribute Entropy	0.0130
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0260
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.17. Ltree c50boost

Attribute	
# classes	0.0222
# attributes	0.0222
# instances	0.0111
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0444
# unknown values	0.0222
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0444
max, min, mean, stdv of nominal attribute values	0.0222 0.0222 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0111 0.0444 0.0000 0.0333 0.0222 0.0111 0.0000 0.0444 0.0000 0.0111
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0333 0.0222 0.0000 0.0333 0.0000 0.0000 0.0111 0.0000 0.0111 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0333
1..5 correlation histogram 6..10 correlation histogram	0.0333 0.0000 0.0000 0.0000 0.0222 0.0111 0.0111 0.0111 0.0000 0.0667
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0111 0.0111 0.0111 0.0111 0.0444 0.0000 0.0111 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0000
Mean Skew	0.0111
Mean Kurtosis	0.0222
Class Entropy	0.0000
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0778
Equivalent number of attributes	0.0556
Noise to Signal Ratio	0.0444
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.18. Ltree c50rules

Attribute	
# classes	0.095
# attributes	0.015
# instances	0.000
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0159
# unknown values	0.031
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0159
# nominal attributes	0.0317
max, min, mean, stdv of nominal attribute values	0.0000 0.0000 0.0000 0.0317
1..5 concentration histogram 6..10 concentration histogram	0.0794 0.0159 0.0000 0.0000 0.0000 0.0000 0.0159 0.0476 0.0159 0.0000
non computable conc. histogram	0.0159
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0635 0.0000 0.0159 0.0317 0.0000 0.0000 0.0476 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0159
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0317 0.0159 0.0000 0.0159 0.0000 0.0159 0.0000 0.0159 0.0159
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0476 0.0159 0.0000 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0794
Mean Skew	0.0159
Mean Kurtosis	0.0159
Class Entropy	0.0317
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0635
Equivalent number of attributes	0.0159
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.19. Ltree c50tree

Attribute	
# classes	0.1481
# attributes	0.0247
# instances	0.0000
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0617
# unknown values	0.0247
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0370
max, min, mean, stdv of nominal attribute values	0.0123 0.0494 0.0000 0.0247
1..5 concentration histogram 6..10 concentration histogram	0.0247 0.0000 0.0000 0.0247 0.0000 0.0123 0.0000 0.0247 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0370 0.0000 0.0370 0.0247 0.0000 0.0123 0.0123 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0123
1..5 correlation histogram 6..10 correlation histogram	0.0370 0.0247 0.0000 0.0247 0.0247 0.0494 0.0370 0.0000 0.0000 0.0000
non computable correlation histogram	0.0123
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0000 0.0123 0.0000 0.0000 0.0123 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0247
Mean Skew	0.0247
Mean Kurtosis	0.0000
Class Entropy	0.0494
Mean Attribute Entropy	0.0247
Mean Mutual Information	0.0247
Equivalent number of attributes	0.0123
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.20. Ltree Lindiscr

Attribute	
# classes	0.0230
# attributes	0.0115
# instances	0.0575
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0230
# unknown values	0.0345
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0230
max, min, mean, stdv of nominal attribute values	0.0000 0.0115 0.0000 0.0115
1..5 concentration histogram 6..10 concentration histogram	0.0115 0.0000 0.0230 0.0345 0.0000 0.0000 0.0115 0.0000 0.0000 0.0230
non computable conc. histogram	0.0115
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0115 0.0345 0.0460 0.0115 0.0000 0.0115 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0115
1..5 correlation histogram 6..10 correlation histogram	0.1264 0.0230 0.0000 0.0230 0.0000 0.0000 0.0115 0.0000 0.0000 0.0805
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0460 0.0230 0.0000 0.0230 0.0000 0.0115 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0230
First Canonical Correlation	0.0230
Mean Skew	0.0345
Mean Kurtosis	0.0230
Class Entropy	0.0345
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0230
Equivalent number of attributes	0.0345
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.21. IBL c50boost

Attribute	
# classes	0.0323
# attributes	0.0323
# instances	0.1129
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0161
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0161 0.0968 0.0161 0.0161
1..5 concentration histogram 6..10 concentration histogram	0.0161 0.0484 0.0000 0.0000 0.0161 0.0000 0.0323 0.0000 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.0000 0.0000 0.0161 0.0000 0.0000 0.0161 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0161
1..5 correlation histogram 6..10 correlation histogram	0.0645 0.0000 0.0000 0.0000 0.0161 0.0323 0.0000 0.0161 0.0000 0.0323
non computable correlation histogram	0.0323
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0161 0.0161 0.0000 0.0000 0.0000 0.0323 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0323
First Canonical Correlation	0.0484
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0000
Mean Attribute Entropy	0.0323
Mean Mutual Information	0.0323
Equivalent number of attributes	0.0968
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.22. IBL c50rules

Attribute	
# classes	0.0972
# attributes	0.0417
# instances	0.0417
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0417
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0139 0.0139 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0139 0.0139 0.0139 0.0139 0.0000 0.0139 0.0833 0.0139 0.0139
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0139 0.0000 0.0000 0.0278 0.0278 0.0000 0.0000 0.0000 0.0278 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0417 0.0000 0.0000 0.0278 0.0000 0.0139 0.0000 0.0139 0.0139 0.0417
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0139 0.0694 0.0139 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0139
First Canonical Correlation	0.0556
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0278
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0694
Equivalent number of attributes	0.0278
Noise to Signal Ratio	0.0278
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.23. IBL c50tree

Attribute	
# classes	0.0380
# attributes	0.0127
# instances	0.0253
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0506
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0127
# nominal attributes	0.0127
max, min, mean, stdv of nominal attribute values	0.0127 0.0253 0.0253 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0127 0.0127 0.0000 0.0000 0.0127 0.0000 0.0000 0.0000 0.0253 0.0380
non computable conc. histogram	0.0506
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0380 0.0000 0.0000 0.0127 0.0127 0.0000 0.0000 0.0000 0.0380 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0127
1..5 correlation histogram 6..10 correlation histogram	0.0759 0.0000 0.0127 0.0127 0.0000 0.0127 0.0253 0.0127 0.0127 0.0127
non computable correlation histogram	0.0127
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0127 0.0127 0.0127 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0127
First Canonical Correlation	0.0253
Mean Skew	0.0127
Mean Kurtosis	0.0380
Class Entropy	0.0127
Mean Attribute Entropy	0.0127
Mean Mutual Information	0.0127
Equivalent number of attributes	0.0633
Noise to Signal Ratio	0.1013
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.24. IBL Lindiscr

Attribute	
# classes	0.0595
# attributes	0.0595
# instances	0.0238
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0833
# unknown values	0.0357
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0119 0.0000 0.0000 0.0119
1..5 concentration histogram 6..10 concentration histogram	0.0357 0.0119 0.0476 0.0000 0.0119 0.0119 0.0000 0.0238 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0238 0.0119 0.0000 0.0238 0.0119 0.0119 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0238
1..5 correlation histogram 6..10 correlation histogram	0.0952 0.0238 0.0000 0.0238 0.0000 0.0595 0.0119 0.0119 0.0119 0.0000
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0119 0.0238 0.0238 0.0238 0.0000 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0000
Mean Skew	0.0238
Mean Kurtosis	0.0238
Class Entropy	0.0000
Mean Attribute Entropy	0.0119
Mean Mutual Information	0.0119
Equivalent number of attributes	0.0476
Noise to Signal Ratio	0.0238
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.25. IBL Ltree

Attribute	
# classes	0.0612
# attributes	0.0204
# instances	0.0306
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0102
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0000 0.0612 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0408 0.0000 0.0102 0.0102 0.0102 0.0000 0.0000 0.0510 0.0204 0.0000
non computable conc. histogram	0.0204
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0306 0.0102 0.0408 0.0816 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0102
1..5 correlation histogram 6..10 correlation histogram	0.0102 0.0102 0.0102 0.0000 0.0204 0.0000 0.0204 0.0102 0.0612 0.0510
non computable correlation histogram	0.0102
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0102 0.0102 0.0102 0.0510 0.0000 0.0204 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0000
Mean Skew	0.0000
Mean Kurtosis	0.0204
Class Entropy	0.0306
Mean Attribute Entropy	0.0204
Mean Mutual Information	0.0204
Equivalent number of attributes	0.0102
Noise to Signal Ratio	0.0714
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.26. NB c50boost

Attribute	
# classes	0.0145
# attributes	0.0145
# instances	0.0435
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0145
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0145 0.0000 0.0145 0.0145
1..5 concentration histogram 6..10 concentration histogram	0.0580 0.0290 0.0435 0.0000 0.0000 0.0000 0.0725 0.0000 0.0000 0.0580
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0580 0.0145 0.0290 0.0435 0.0000 0.0145 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0145
# continuous attributes	0.0145
1..5 correlation histogram 6..10 correlation histogram	0.0290 0.0435 0.0000 0.0435 0.0290 0.0290 0.0000 0.0000 0.0145 0.0000
non computable correlation histogram	0.0290
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0435 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0145
First Canonical Correlation	0.0145
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0145
Mean Attribute Entropy	0.0145
Mean Mutual Information	0.0290
Equivalent number of attributes	0.0435
Noise to Signal Ratio	0.0290
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.27. NB c50rules

Attribute	
# classes	0.0220
# attributes	0.0220
# instances	0.0110
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0110
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0330
# nominal attributes	0.0110
max, min, mean, stdv of nominal attribute values	0.0110 0.0110 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0330 0.0110 0.0000 0.0000 0.0220 0.0110 0.0220 0.0000 0.0220 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0110 0.0000 0.0549 0.0549 0.0110 0.0440 0.0549 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0220
1..5 correlation histogram 6..10 correlation histogram	0.0549 0.0330 0.0000 0.0110 0.0330 0.0110 0.0220 0.0000 0.0000 0.0110
non computable correlation histogram	0.0220
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0330 0.0110 0.0330 0.0110 0.0110 0.0000 0.0440 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0110
First Canonical Correlation	0.0330
Mean Skew	0.0110
Mean Kurtosis	0.0220
Class Entropy	0.0440
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0110
Equivalent number of attributes	0.0220
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.28. NB c50tree

Attribute	
# classes	0.0588
# attributes	0.0353
# instances	0.0588
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0118
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0118
# nominal attributes	0.0353
max, min, mean, stdv of nominal attribute values	0.0235 0.0235 0.0000 0.0118
1..5 concentration histogram 6..10 concentration histogram	0.0118 0.0471 0.0118 0.0000 0.0235 0.0588 0.0353 0.0000 0.0235 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0118 0.0000 0.0353 0.0235 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0118
1..5 correlation histogram 6..10 correlation histogram	0.0235 0.0235 0.0000 0.0353 0.0235 0.0235 0.0000 0.0235 0.0000 0.0118
non computable correlation histogram	0.0118
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0118 0.0000 0.0235 0.0118 0.0471 0.0000 0.0118 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0235
Mean Skew	0.0235
Mean Kurtosis	0.0118
Class Entropy	0.0706
Mean Attribute Entropy	0.0118
Mean Mutual Information	0.0118
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.29. NB Lindiscr

Attribute	
# classes	0.0404
# attributes	0.0404
# instances	0.0101
$\frac{\text{# attributes}}{\text{#instances}}$	0.0101
# unknown values	0.0101
$\frac{\text{# unknown values}}{\text{# attributes * # instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0202 0.0202 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0101 0.0000 0.0000 0.0101 0.0101 0.0101 0.0303 0.0404 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0404 0.0101 0.0000 0.0404 0.0404 0.0202 0.0000 0.0000 0.0101 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0202
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0101 0.0202 0.0101 0.0202 0.0404 0.0202 0.0202 0.0101 0.0404
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0404 0.0404 0.0101 0.0303 0.0000 0.0202 0.0000 0.0000
$\frac{\text{# continuous}}{\text{# attributes}}$	0.0000
$\frac{\text{# nominal}}{\text{# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0101
First Canonical Correlation	0.0303
Mean Skew	0.0000
Mean Kurtosis	0.0707
Class Entropy	0.0303
Mean Attribute Entropy	0.0101
Mean Mutual Information	0.0303
Equivalent number of attributes	0.0101
Noise to Signal Ratio	0.0303
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.30. NB Ltree

Attribute	
# classes	0.0156
# attributes	0.0625
# instances	0.0156
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0625
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0312
# nominal attributes	0.0312
max, min, mean, stdv of nominal attribute values	0.0156 0.0156 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0312 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0156 0.0156 0.0156
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.1094 0.0156 0.0156 0.0000 0.0000 0.0000 0.0156
non computable conc. histogram with class	0.0000
# continuous attributes	0.0156
1..5 correlation histogram 6..10 correlation histogram	0.0156 0.0469 0.0625 0.0000 0.0312 0.0000 0.0000 0.0156 0.0312 0.0000
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0469 0.0000 0.0156 0.0156 0.0000 0.0469 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0312
First Canonical Correlation	0.0000
Mean Skew	0.0156
Mean Kurtosis	0.0156
Class Entropy	0.0156
Mean Attribute Entropy	0.0156
Mean Mutual Information	0.0312
Equivalent number of attributes	0.0156
Noise to Signal Ratio	0.0312
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.31. NB IBL

Attribute	
# classes	0.0659
# attributes	0.0440
# instances	0.0000
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0220
# unknown values	0.0220
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0549 0.0000 0.0110 0.0110
1..5 concentration histogram 6..10 concentration histogram	0.0549 0.0330 0.0110 0.0000 0.0000 0.0000 0.0769 0.0000 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0000 0.0110 0.0110 0.0220 0.0110 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0220
1..5 correlation histogram 6..10 correlation histogram	0.0549 0.0330 0.0000 0.0000 0.0110 0.0220 0.0330 0.0000 0.0110 0.0220
non computable correlation histogram	0.0110
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0440 0.0440 0.0000 0.0110 0.0000 0.0220 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0110
First Canonical Correlation	0.0659
Mean Skew	0.0000
Mean Kurtosis	0.0000
Class Entropy	0.0440
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0110
Equivalent number of attributes	0.0659
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.32. ripper c50boost

Attribute	
# classes	0.0119
# attributes	0.0238
# instances	0.0357
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0119
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0119 0.0000 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0238 0.0119 0.0476 0.0357 0.0357 0.0119 0.0000 0.0595 0.0000 0.0357
non computable conc. histogram	0.0238
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0357 0.0119 0.0119 0.0238 0.0357 0.0119 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0119
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0000 0.0000 0.0000 0.0119 0.0000 0.0357 0.0119 0.0000 0.0357
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0357 0.0238 0.0000 0.0714 0.0119 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0357
Mean Skew	0.0833
Mean Kurtosis	0.0119
Class Entropy	0.0476
Mean Attribute Entropy	0.0119
Mean Mutual Information	0.0119
Equivalent number of attributes	0.0119
Noise to Signal Ratio	0.0238
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.33. ripper c50rules

Attribute	
# classes	0.0132
# attributes	0.0263
# instances	0.0000
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0789
# unknown values	0.0000
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0132 0.0395 0.0000 0.0132
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0000 0.0000 0.0000 0.0526 0.0132 0.0000 0.0132 0.0000 0.0132
non computable conc. histogram	0.0132
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0395 0.0263 0.0658 0.0000 0.0263 0.0395 0.0000 0.0000 0.0000 0.0132
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0132 0.0000 0.0395 0.0000 0.0000 0.0000 0.0132 0.0658 0.0000 0.0263
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0132 0.0000 0.0395 0.0263 0.0263 0.0000 0.0263 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0395
Mean Skew	0.0263
Mean Kurtosis	0.0000
Class Entropy	0.0789
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0132
Equivalent number of attributes	0.0263
Noise to Signal Ratio	0.0263
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.34. ripper c50tree

Attribute	
# classes	0.0563
# attributes	0.0000
# instances	0.0704
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0704
# unknown values	0.0282
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0563 0.0141 0.0141 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0282 0.0423 0.0563 0.0141 0.0141 0.0141 0.0000 0.0000 0.0423 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0141
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0141 0.0282 0.0423 0.0000 0.0000 0.0282 0.0141 0.0141 0.0423
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0282 0.0141 0.0141 0.0000 0.0423 0.0000 0.0141 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0282
Mean Skew	0.0563
Mean Kurtosis	0.0000
Class Entropy	0.0423
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0141
Equivalent number of attributes	0.0000
Noise to Signal Ratio	0.0282
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.35. ripper Lindiscr

Attribute	
# classes	0.0000
# attributes	0.0000
# instances	0.0685
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0548
# unknown values	0.0274
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0000 0.0000 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0137 0.0000 0.0685 0.0548 0.0274 0.0137 0.0000 0.0000 0.0137 0.0000
non computable conc. histogram	0.0137
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0000 0.0000 0.0000 0.0274 0.0000 0.0274 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0137 0.0548 0.0548 0.0137 0.0000 0.0411 0.0000 0.0000 0.0548 0.0274
non computable correlation histogram	0.0685
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0137 0.0137 0.0000 0.0000 0.0137 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0137
First Canonical Correlation	0.0137
Mean Skew	0.0685
Mean Kurtosis	0.0000
Class Entropy	0.0548
Mean Attribute Entropy	0.0137
Mean Mutual Information	0.0411
Equivalent number of attributes	0.0137
Noise to Signal Ratio	0.0000
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.36. ripper Ltree

Attribute	
# classes	0.0575
# attributes	0.0345
# instances	0.0000
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0115
# unknown values	0.0230
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0345
max, min, mean, stdv of nominal attribute values	0.0230 0.0345 0.0000 0.0230
1..5 concentration histogram 6..10 concentration histogram	0.0115 0.0000 0.0000 0.0000 0.0230 0.0000 0.0000 0.0460 0.0000 0.0000
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0115 0.0690 0.0000 0.0115 0.0230 0.0000 0.0000 0.0000 0.0115 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0230 0.0000 0.0000 0.0115 0.0000 0.0115 0.0115 0.0230 0.0000 0.0000
non computable correlation histogram	0.0115
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0345 0.0345 0.0115 0.0000 0.0000 0.0230 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0230
Mean Skew	0.0575
Mean Kurtosis	0.0460
Class Entropy	0.0460
Mean Attribute Entropy	0.0575
Mean Mutual Information	0.0345
Equivalent number of attributes	0.0575
Noise to Signal Ratio	0.0345
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.37. ripper IBL

Attribute	
# classes	0.0323
# attributes	0.0108
# instances	0.0215
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0108
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0323 0.0645 0.0108 0.0215
1..5 concentration histogram 6..10 concentration histogram	0.0215 0.0000 0.0430 0.0000 0.0323 0.0108 0.0108 0.0108 0.0215 0.0108
non computable conc. histogram	0.0000
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0215 0.0000 0.0108 0.0215 0.0215 0.0108 0.0000 0.0000 0.0430 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0000
1..5 correlation histogram 6..10 correlation histogram	0.0645 0.0108 0.0215 0.0000 0.0000 0.0215 0.0108 0.0323 0.0215 0.0000
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0000 0.0215 0.0215 0.0000 0.0000 0.0000 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0215
Mean Skew	0.0860
Mean Kurtosis	0.0215
Class Entropy	0.0323
Mean Attribute Entropy	0.0323
Mean Mutual Information	0.0000
Equivalent number of attributes	0.0753
Noise to Signal Ratio	0.0108
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000

Table B.38. ripper NB

Attribute	
# classes	0.0238
# attributes	0.0357
# instances	0.0238
$\frac{\text{\# attributes}}{\text{\#instances}}$	0.0000
# unknown values	0.0238
$\frac{\text{\# unknown values}}{\text{\# attributes * \# instances}}$	0.0000
# nominal attributes	0.0000
max, min, mean, stdv of nominal attribute values	0.0238 0.0476 0.0000 0.0000
1..5 concentration histogram 6..10 concentration histogram	0.0119 0.0119 0.0357 0.0000 0.0238 0.0119 0.0119 0.0000 0.0000 0.0119
non computable conc. histogram	0.0119
1..5 concentration histogram with class 6..10 concentration histogram with class	0.0119 0.0119 0.0119 0.0476 0.0119 0.0238 0.0000 0.0000 0.0000 0.0000
non computable conc. histogram with class	0.0000
# continuous attributes	0.0476
1..5 correlation histogram 6..10 correlation histogram	0.0000 0.0238 0.0476 0.0000 0.0119 0.0476 0.0119 0.0000 0.0000 0.0000
non computable correlation histogram	0.0000
1..5 missing values histogram 6..10 missing values histogram	0.0000 0.0000 0.0238 0.0000 0.0000 0.0238 0.0000 0.0476 0.0000 0.0000
$\frac{\text{\# continuous}}{\text{\# attributes}}$	0.0000
$\frac{\text{\# nominal}}{\text{\# attributes}}$	0.0000
Binary Attributes	0.0000
Frac1	0.0000
First Canonical Correlation	0.0119
Mean Skew	0.0357
Mean Kurtosis	0.0476
Class Entropy	0.1190
Mean Attribute Entropy	0.0000
Mean Mutual Information	0.0119
Equivalent number of attributes	0.0238
Noise to Signal Ratio	0.0357
Mean Mult. Correl. Coef.	0.0000
SDratio	0.0000