Feature Selection with the logRatio Kernel

Julien Prados* Alexandros Kalousis* Melanie Hilario*

Abstract

In this article we present a novel kernel function, logRatio, which was designed to address two common problems in biological applications: data preprocessing and attribute interaction modelling. An extension of the SVMRFE feature selection algorithm was built around this new kernel function and compared with the original on a number of biological data and text classification problems. Experiments showed that SVMRFE based on the logRatio kernel detects relevant information and handles attribute redundancy more effectively than SVMRFE coupled with other kernels.

1 Introduction

Machine learning is faced with an ever increasing number of application problems involving very high dimensional data. Experimental biology is a rich source of such applications, as high throughput methods produce huge quantities of data comprising tens of thousands (DNA-microarrays) or even hundreds of thousands of features (mass-spectrometry). This extremely high dimensionality poses serious computational and data-analytical problems collectively known as the curse of dimensionality. Different approaches have been investigated to alleviate such problems; one of these is feature selection, which reduces dimensionality by focusing the data mining process on a subset of relevant features.

An effective feature selection algorithm is SVM-RFE (Support Vector Machine Recursive Feature Elimination)[7]. It capitalises on the remarkable performance of—usually linear—SVMs on high dimensional problems and creates nested subsets of attributes based on their impact on the SVM model predictions. However SVMRFE is sensitive to data normalisation and attribute redundancy, two common issues in biological data.

In this paper we introduce a kernel function, logRatio, which has been specifically designed for biological problems. Its properties allow us to define an attribute selection method in the spirit of SVMRFE but without its drawbacks. In the rest of this introduction we review some basic concepts before presenting our method. In Section 2 we introduce the logRatio kernel and the adaptation of SVMRFE to this kernel; in Section 3 we investigate the behavior of our approach in a number of experiments using both controlled and real-world datasets.

1.1 Support Vector Machines (SVM) Support Vector Machines are linear binary classifiers which find the maximal margin hyperplane that separates positive from negative instances[2]. Through the use of kernel functions ($K(\mathbf{x},\mathbf{z}) = \langle\phi(\mathbf{x}).\phi(\mathbf{z})\rangle$) they can implicitly map instances onto a feature space where linear separation is performed which might induce non-linear separation in the initial space, depending on the projection function ϕ. For example, a polynomial kernel of degree 2 implicitly converts a vector $\mathbf{x}$ of n attributes into a vector $\phi(\mathbf{x})$ of n^2 attributes containing all pair-wise products of attributes of $\mathbf{x}$ (that is, $\phi(\mathbf{x}) = (x_i x_j)_{(i,j)=(1,1)}^{(n,n)}$), and the linear separation computed by SVM in that space is not linear in the original space.

The decision function, $f(\mathbf{x})$, learnt by SVM is the equation of the separating hyperplane in the feature space. It is expressed as a linear combination of kernel functions, $K(.,.)$ between training instances ($\mathcal{T} = \{(\mathbf{x}_i, y_i)|\mathbf{x}_i \in \Re^n, y_i \in \{-1,1\}\}$) and the test instance ($\mathbf{x} \in \Re^n$). The linear coefficients, α_i, and the intercept, b, of the hyperplane are determined during learning. The final form of $f(\mathbf{x})$ is:

$$f(\mathbf{x}) = \mathrm{sgn}(\sum_i y_i\alpha_i K(\mathbf{x}_i,\mathbf{x}) + b) \tag{1.1}$$

If $K(.,.)$ is the linear kernel, i.e. the standard dot product, then the feature space and the initial space coincide. In this case a simplification of $f(\mathbf{x})$ is possible where the normal vector, $\mathbf{w} = \sum_i y_i\alpha_i\mathbf{x}_i$, of the separating hyperplane is used:

$$f(\mathbf{x}) = \mathrm{sgn}(\sum_i y_i\alpha_i\langle\mathbf{x}_i.\mathbf{x}\rangle + b) \tag{1.2}$$

$$= \mathrm{sgn}(\langle\mathbf{w}.\mathbf{x}\rangle + b) \tag{1.3}$$

This simplification facilitates the interpretation of SVM models because it associates a weight, w_i, to each attribute of the initial space; these weights can be used

[*] Centre Universitaire d'Informatique, University of Geneva

to identify those attributes which contribute most to the model's prediction. The next section shows how SVMRFE makes use of this interpretation to perform attribute selection.

1.2 SVMRFE This feature selection algorithm was originally proposed in [7]. It eliminates the attributes with the lowest influence on the predictions of an SVM model. An attribute is considered "useless" if large variations of its value result in small changes in the model's predictions. In the linear kernel case, these attributes are easy to identify from Eq. (1.2): they are the ones with the smallest absolute weight value, $|w_i|$. SVMRFE establishes an attribute ranking by repeating the following three steps: 1) learn a linear SVM, 2) rank attributes according to their $|w_i|$ values, and 3) eliminate the lowest ranked attributes.

Several extensions to SVMRFE have been proposed. Some of them try to relieve the computational burden by intelligently estimating the number of attributes to remove in each RFE iteration [5, 4]. In their original article, Guyon et al. [7] proposed an extension to non-linear kernels, in which attributes were removed in such a manner that predictions before and after removal were as close as possible. This criterion was modified to improve feature selection results in [14] and [11]. Finally, other authors decoupled feature selection from the linear kernel by using a set of attribute-scaling parameters based on, but distinct from, the attribute coefficients of the learnt SVM model. Chapelle et al.[1] use these scaling factors to select the attributes to remove, while Weston et al.'s [13] iterative adjustment process approximates a 0-norm SVM which automatically discards some attributes.

One problem of SVMRFE and its extensions is their sensitivity to data preprocessing, in particular to data normalisation. If no normalisation is performed, each element w_i of the weight vector comprises both a corrective normalisation factor and a factor related to the actual importance of the i^{th} attribute. The normalisation factor distorts SVMRFE's interpretation of attribute importance. It is thus indispensable to perform attribute normalisation in order to allow for meaningful comparison of attributes. Unfortunately, there are diverse ways of normalising attributes, and it is not clear which of these methods fully cancels out the corrective factor. Moreover, the result of attribute normalisation is affected by instance normalisation, which plays an important role in biological applications. Biological data often arise from experiments which are liable to errors (e.g. non-intended variations in sample quantities) that affect the reported values. In order to produce attribute values which are comparable across instances (e.g., when the same experiment is repeated several times, or the same measurement is taken from different individuals), a multiplicative factor is typically applied to instances that sets their norm to one (e.g. total ion current in mass spectrometry[9]). Once again, this does not necessarily yield the ideal normalisation coefficients.

In the next section we describe the logRatio kernel function, whose properties allow us to define an attribute selection algorithm, based on SVMRFE, that overcomes the data normalisation problems.

2 Method

2.1 logRatio kernel The feature space of the logRatio kernel is defined by the function ϕ that projects an instance $\mathbf{x}$, with n attributes, onto the vector $\phi(x)$, with n^2 features, which contains the logarithms of all pairwise ratios of the attributes of $\mathbf{x}$:

$$\phi(\mathbf{x}) = \left(\log \frac{x_i}{x_j}\right)_{(i,j)=(1,1)}^{(n,n)} \tag{2.4}$$

Our interest in this representation comes from the fact that biologists make heavy use of a similar quantity, also based on logs of ratios, in their comparative analysis of expression data, and in particular in DNA microarray analysis[10] (although it has recently been criticised[12]). However our use of logs of ratios is fundamentally different. In comparative DNA microarray experiments, log ratios are taken over the values of a given attribute in two different specimens, one of which is the reference specimen. By contrast, we define log ratios between different attributes of a single specimen; each attribute is expressed relatively to every other attribute, thus rendering normalisation unnecessary, whether through the use of reference specimens or some other procedure. The only assumption is that all attributes represent commensurate quantities, e.g. concentrations of genes, frequencies of words, etc. The log of a ratio has the interesting property that its sign is related to the result of the comparison between the numerator and the denominator, and its value changes according to the magnitude of the difference. The logRatio representation is interesting because it combines attributes pairwise (as does the polynomial kernel of degree 2) and hence focuses learning on attribute-attribute interactions, a central issue in biology. It is also clear that the logRatio mapping is insensitive to instance normalisation because $\forall \alpha > 0 : \phi(\alpha\mathbf{x}) = \phi(\mathbf{x})$.

A similar representation was proposed in [6], where the authors defined new attributes whose values result from attribute comparisons in the form of $I(x_i > x_j)$, where $I(true) = 1$ otherwise 0. However their representation was defined explicitly, and not through a

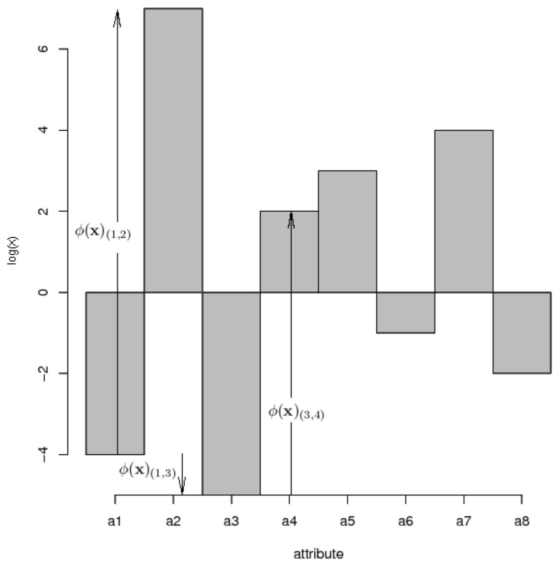


Figure 1: Example of visualisation of 64 components of $\boldsymbol{\phi}(\mathbf{x})$ vector in logRatio feature space, by plotting logarithms of the 8 components of $(\log(x))$.

kernel function, rendering learning on it intractable as the number of initial attributes increased.

The logRatio representation is easy to visualise and interpret. Because $\log \frac{x_i}{x_j} = \log x_i - \log x_j$, it is possible to visualise all the n^2 components of the vector $\boldsymbol{\phi}(\mathbf{x})$ by plotting only the n components of $\log \mathbf{x}$, and considering all differences between the components (see figure 1).

It is relatively easy to show that the logRatio kernel is actually the covariance of the log-transformed instances:

$$\begin{aligned} (2.5) \quad \mathcal{K}(\mathbf{x}, \mathbf{y}) &= \sum_{i,j} \log \frac{x_i}{x_j} \log \frac{y_i}{y_j} \\ (2.6) \quad &= 2n(n-1)\,\text{cov}(\log(\mathbf{x}), \log(\mathbf{y})) \end{aligned}$$

Covariance is simply the dot product of two vectors from which their corresponding mean values have been subtracted. Thus learning with the logRatio is equivalent to: 1) transforming the dataset values to their logarithm; 2) subtracting the mean from each instance; 3) learning with the linear kernel. At first glance, this is very similar to linear learning, but we will see that in reality the logRatio representation changes our interpretation of the results, leading to an interesting strategy of attribute selection when the logRatio is integrated into the SVM algorithm.

Properties: The following properties of the logRatio kernel are easy to prove; they are also not surprising because of the analogy with the linear kernel. The notations $\mathbf{x}^{\alpha}$ and $\mathbf{y} \otimes \mathbf{z}$, denote, respectively, component-wise exponentiation and division of vector elements.

$$(2.7) \quad \mathcal{K}(\alpha\mathbf{x}, \mathbf{y}) = \mathcal{K}(\mathbf{x}, \mathbf{y})$$

$$(2.8) \quad \mathcal{K}(\mathbf{x}, \mathbf{y}) + \mathcal{K}(\mathbf{x}, \mathbf{z}) = \mathcal{K}(\mathbf{x}, \mathbf{y} \otimes \mathbf{z})$$

$$(2.9) \quad \alpha\mathcal{K}(\mathbf{x}, \mathbf{y}) = \mathcal{K}(\mathbf{x}, \mathbf{y}^{\alpha}) = \mathcal{K}(\mathbf{x}^{\alpha}, \mathbf{y})$$

Note that attribute normalisation in the initial space is equivalent to a translation in the feature space. If attribute i is scaled by a factor β_i, then $\boldsymbol{\phi}(\mathbf{x}) = \left(\log \frac{\beta_i x_i}{\beta_j x_j}\right) = \left(\log \frac{x_i}{x_j} + \log \frac{\beta_i}{\beta_j}\right)$, that is, each feature ij is translated by $\log \frac{\beta_i}{\beta_j}$. This property of the kernel implies that the Euclidean distance of two instances in the logRatio feature space does not change, no matter how we normalise the attributes. Thus the logRatio kernel could be used to implement a nearest neighbour classifier that is insensitive to attribute and instance normalisation. In the following, we will see that the SVM classifier learnt with the logRatio is also insensitive to any normalisation.

2.2 logRatio-SVM and logRatio-SVMRFE The properties of the logRatio kernel allow a simplification of the SVM decision function similar to the linear-SVM simplification. Using properties (2.8), (2.9), Eq. (1.1), and defining $\mathbf{w} = \prod_i \mathbf{x}_i{}^{y_i \alpha_i}$, the decision function becomes:

$$\begin{aligned} (2.10) \quad f(\mathbf{x}) &= \text{sgn}(\mathcal{K}(\prod_i \mathbf{x}_i{}^{y_i\alpha_i}, \mathbf{x}) + b) \\ (2.11) \quad &= \text{sgn}(\mathcal{K}(\mathbf{w}, \mathbf{x}) + b) \end{aligned}$$

which is identical to the linear-SVM expression, Eq. (1.2), except that the scalar product is replaced by the logRatio kernel. The resulting expression is simpler because $\mathbf{w}$ contains all the model information, and a single computation of the kernel is required to make a prediction. Again, this should not come as a surprise because of the analogy with the linear kernel.

However, the interpretations of logRatio-SVM and linear-SVM models are different. In both cases, $\boldsymbol{\phi}(\mathbf{w})$ is a normal vector of the maximal margin hyperplane; however, in the linear case, $\boldsymbol{\phi}(\mathbf{w}) = \mathbf{w}$ contains n linear coefficients associated with the attributes of the initial space, while in the logRatio case, $\boldsymbol{\phi}(\mathbf{w})$ contains n^2 linear coefficients associated with the features of the logRatio space. The n^2 linear coefficients of $\boldsymbol{\phi}(\mathbf{w})$ in logRatio space can be visualised similarly to Figure 1 by plotting the values of the $\log(\mathbf{w})$ vector. With such a visualisation, we identify the most important coefficient of $\phi(\mathbf{w})$ in absolute value as the one which combines the two most extreme values (maximum and minimum). It is even possible to rank the n^2 features of the logRatio space considering all the differences among the elements of vector $\log(\mathbf{w})$. However this is of no interest to us, first because it is time consuming, and second because our goal is to rank the attributes of the initial space.

Recall that recursive feature elimination successively eliminates attributes deemed irrelevant. The question is: how can we identify the attribute of the initial space that has the smallest effect on the predictions produced by logRatio-SVM? It is clear that this is the attribute that produces the smallest linear coefficient when combined with all the others via the logRatio transform, that is, the attribute k that minimises $\sum_{i=1}^{n}(|\log(w_i) - \log(w_k)|)$. In short, it is the attribute k that has the median value of $\log(w_k)$. If we rank attributes k in increasing order of $\log(\mathbf{w})$, with the median in rank $n/2$, we can then rerank them according to relevance based on their distance to (their rank wrt) the median. The most relevant attributes are those whose $\log(w_k)$ values are farthest from the median—either the minimum (rank 1) or the maximum (rank n). By inserting this ranking strategy in an RFE loop, we define the attribute selection algorithm logRatio-SVMRFE (pseudo code given in algorithm 1).

Algorithm 1 logRatio-SVMRFE

```
Input: the dataset X
Output: r[] attribute ranking from worst to best
while X contains more than 1 attribute do
    $\mathbf{w} \leftarrow$ train logRatio-SVM over X
    Sort vector $\log \mathbf{w}$
    $F \leftarrow$ { attribute with median value in $\log(\mathbf{w})$ whose rank is $n/2$ in the sorted vector plus, eventually, attributes with ranks close to $n/2$ }
    Eliminate from X the attributes in $F$
    $r \leftarrow r \cup F$
end while
```

Insensitivity to normalisation: SVM-logRatio and SVMRFE-logRatio are insensitive to both instance and attribute scaling, thus rendering data normalisation unnecessary. They are clearly insensitive to instance normalisation because the logRatio kernel is (property (2.7)). Concerning insensitivity to attribute normalisation, we saw that attribute normalisation in the initial space is equivalent to a translation of the feature space, and distances between instances are invariant to such a transformation. As a consequence, the maximal margin hyperplane learnt by logRatio-SVM is just translated, but its direction is preserved. In other words, the hyperplane's intercept b changes, but its linear coefficients $\phi(\mathbf{w})$ remain identical. Hence attribute normalisation has no impact on our interpretation of logRatio-SVM models.

We performed the following experiment to verify that the logRatio kernel is insensitive to and in fact requires no normalisation. For a given dataset, we generated a denormalised version by multiplying each instance and each attribute by a random positive number. We then trained logRatio-SVM on both the original and the denormalised versions. The models built on the two version were identical. By contrast, we know that linear-SVM requires data normalisation[7]. We therefore normalised the dataset (using for instance L1-norm and attribute standardisation), then trained linear-SVM models on both the original and the normalised versions. The models obtained on the two datasets were different.

Attribute redundancy in logRatio-SVM models: A comment is in order concerning the effect of attribute redundancy on the models learnt by logRatio-SVM. In figure 1 we see the visual representation of a vector $\mathbf{w}$ learnt with logRatio-SVM on some dataset; consider the first three attributes a_1, a_2, a_3, the values of their weights are $\log w_1 = -4$, $\log w_2 = 7$, $\log w_3 = -5$. The smallest linear coefficient of this model in the feature space is associated with the attribute combination $a_1 a_3$ (weight difference is only 1), i.e. this feature only marginally influences the model's decisions. On the other hand, the combinations $a_2 a_3$ (difference 12) and $a_2 a_1$ (difference 11) influence the model's decisions considerably. So, a_1 and a_3 are relevant when they are combined with a_2, but not when they are combined together.

This pattern of behaviour will appear when a_1 and a_3 are redundant. In this case, the same information is combined twice in $a_1 a_3$, and so, logRatio-SVM learning will assign it a small coefficient to the benefit of other more informative attribute combinations such as $a_2 a_1$ and $a_2 a_3$. By generalising this observation, we can say that if two attributes a_i and a_j are redundant, then their values $\log w_i$ and $\log w_j$ are similar. However, the opposite is not necessarily true: if two values $\log w_i$ and $\log w_j$ are close, it is not necessarily because attributes a_i and a_j are redundant.

3 Results

Two series of experiments were conducted on the logRatio kernel. The first explored the behaviour of logRatio-SVMRFE on small artificial datasets. The second evaluated the performance of logRatio-SVM and logRatio-SVMRFE on high-dimensional datasets from the biological domain. Note also that, unlike to Guyon et al.[7], we implemented SVMRFE using ν-SVM[2] instead of C-SVM. The advantage is that ν parameter is more consistent from one RFE iteration to another whereas the C parameter should be retuned at each RFE iteration[2]. Experiments were performed using package *e1071* of *R* language(`http://www.r-project.org`) which contains a wrapper to *libsvm-2.83* (`http://www.csie.ntu.edu.tw/~cjlin/libsvm`). ν was fixed to $\nu = 0.3$ in all experiments.

3.1 Preliminary experiments The goal of these exploratory experiments is to investigate the impact of attribute redundancy and noise on the relative performance of linear-SVMRFE and logRatio-SVMRFE. In order to zoom in more precisely on the expected/observed effects, we chose to work on small samples derived from the "Iris" dataset (`http://www.ics.uci.edu/~mlearn/MLRepository.html`). This study is limited to the task of classifying the 100 instances of the virginica and versicolor classes, the most difficult to distinguish. Each flower is characterised by four attributes that represent its petal/sepal length/width expressed in centimetres (henceforth cm-attributes). To introduce redundancy, four attributes were added which simply reexpressed the original attributes in millimetres (henceforth mm-attributes). Noise was also injected by adding a random number from the normal distribution $\mathcal{N}(0, \delta_{\mathrm{mm}})$ to each mm-attribute value, and a random number from $\mathcal{N}(0, \delta_{\mathrm{cm}})$ to each cm-attribute value. This simulated two independent measurements of the same value in different units. This repetition of information is not necessarily useless because by averaging noisy attributes expressed in centimetres with those expressed in millimetres, we can expect a reduction in noise and an increase in measure precision. Thus, the ideal behaviour of attribute selection algorithms on this dataset depends on the parameters controlling the noise (δ). Three dataset variants were studied in our experiments: $\mathrm{iris}_{(\delta_{\mathrm{mm}}=0,\delta_{\mathrm{cm}}=0)}$, $\mathrm{iris}_{(\delta_{\mathrm{mm}}=0,\delta_{\mathrm{cm}}=0.5)}$, and $\mathrm{iris}_{(\delta_{\mathrm{mm}}=5,\delta_{\mathrm{cm}}=0.5)}$.

logRatio-SVM model analysis: We start by analysing the logRatio-SVM model learnt on $\mathrm{iris}_{(\delta_{\mathrm{mm}}=0,\delta_{\mathrm{cm}}=0)}$. Figure 2 visualises the **w** vector learnt by logRatio-SVM. Note the symmetry of the mm and cm attributes which have similar weights. This is in keeping with the expected behaviour and correctly reflects their redundancy. It is interesting to compare the model of Figure 2 with the correlation coefficients of the attributes (Table 1). We can see that Petal Length and Petal Width are highly correlated (correlation coefficient of 0.96); thus as expected, their weight values in the logRatio model are close. However although Petal Length and Sepal Length are also correlated, their distance in the model is high, i.e. their combined feature has a high coefficient. Thus logRatio-SVM is able to distinguish and utilize pairs of attributes that contain important discriminative information despite a high degree of redundancy. This type of analysis of the logRatio-SVM models provides end users with interesting feed-back that is especially attractive in biological applications, where attribute interactions and associations are important.

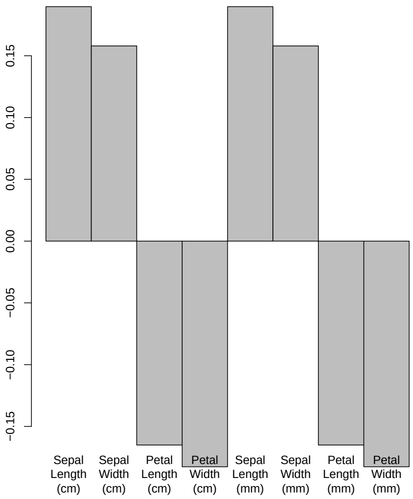


Figure 2: logRatio-SVM model trained with $\mathrm{iris}_{(\delta_{\mathrm{mm}}=0,\delta_{\mathrm{cm}}=0)}$

	Sepal Length	Sepal Width	Petal Length
Sepal Width	-0.11		
Petal Length	0.87	-0.42	
Petal Width	0.81	-0.36	0.96

Table 1: Pearson correlation coefficients of all pair-wise combinations of Iris attributes

Attribute selection results: We ran Linear-SVMRFE and logRatio-SVMRFE on the three datasets, removing a single attribute at each iteration of the RFE loop. In the linear-SVMRFE case, attributes were standardised beforehand whereas in logRatio-SVMRFE case, no preprocessing was performed. Table 2 shows the attribute rankings obtained.

$\mathrm{Iris}_{(\delta_{\mathrm{mm}}=0,\delta_{\mathrm{cm}}=0)}$ contains no noise, that is, it delivers exactly the same information twice, though expressed in different units. An ideal attribute selection algorithm should then eliminate one of the redundant attributes independently of its unit. Table 2 shows that logRatio-SVMRFE perfectly fulfils this task by top-ranking four mutually non-redundant variables, whereas linear-SVMRFE selects two variants of the same attribute. This difference in behaviour certainly comes from the fact that the linear kernel considers attributes individually whereas the logRatio kernel examines attributes by pairs and therefore readily captures attribute redundancy.

In $\mathrm{iris}_{(\delta_{\mathrm{mm}}=0,\delta_{\mathrm{cm}}=0.5)}$, only cm-attributes are noisy.

	$\mathrm{iris}_{(\delta_{\mathrm{mm}}=0,\delta_{\mathrm{cm}}=0)}$		$\mathrm{iris}_{(\delta_{\mathrm{mm}}=0,\delta_{\mathrm{cm}}=0.5)}$		$\mathrm{iris}_{(\delta_{\mathrm{mm}}=5,\delta_{\mathrm{cm}}=0.5)}$	
	linear	logRatio	linear	logRatio	linear	logRatio
1	$\mathrm{PW}_{\mathrm{mm}}$	$\mathrm{SL}_{\mathrm{mm}}$	$\mathrm{PW}_{\mathrm{mm}}$	$\mathrm{SL}_{\mathrm{mm}}$	$\mathrm{PL}_{\mathrm{cm}}$	$\mathrm{PL}_{\mathrm{cm}}$
2	$\mathrm{PL}_{\mathrm{mm}}$	$\mathrm{PL}_{\mathrm{mm}}$	$\mathrm{PL}_{\mathrm{mm}}$	$\mathrm{PL}_{\mathrm{mm}}$	$\mathrm{PW}_{\mathrm{cm}}$	$\mathrm{SL}_{\mathrm{cm}}$
3	$\mathrm{PW}_{\mathrm{cm}}$	$\mathrm{PW}_{\mathrm{cm}}$	$\mathrm{PL}_{\mathrm{cm}}$	$\mathrm{PW}_{\mathrm{mm}}$	$\mathrm{PL}_{\mathrm{mm}}$	$\mathrm{PL}_{\mathrm{mm}}$
4	$\mathrm{PL}_{\mathrm{cm}}$	$\mathrm{SW}_{\mathrm{cm}}$	$\mathrm{SW}_{\mathrm{mm}}$	$\mathrm{SW}_{\mathrm{mm}}$	$\mathrm{PW}_{\mathrm{mm}}$	$\mathrm{SW}_{\mathrm{cm}}$
5	$\mathrm{SW}_{\mathrm{cm}}$	$\mathrm{SL}_{\mathrm{cm}}$	$\mathrm{PW}_{\mathrm{cm}}$	$\mathrm{PL}_{\mathrm{cm}}$	$\mathrm{SL}_{\mathrm{mm}}$	$\mathrm{PW}_{\mathrm{cm}}$
6	$\mathrm{SW}_{\mathrm{mm}}$	$\mathrm{PL}_{\mathrm{cm}}$	$\mathrm{SL}_{\mathrm{mm}}$	$\mathrm{SL}_{\mathrm{cm}}$	$\mathrm{SL}_{\mathrm{cm}}$	$\mathrm{SL}_{\mathrm{mm}}$
7	$\mathrm{SL}_{\mathrm{mm}}$	$\mathrm{PW}_{\mathrm{mm}}$	$\mathrm{SL}_{\mathrm{cm}}$	$\mathrm{SW}_{\mathrm{cm}}$	$\mathrm{SW}_{\mathrm{cm}}$	$\mathrm{PW}_{\mathrm{mm}}$
8	$\mathrm{SL}_{\mathrm{cm}}$	$\mathrm{SW}_{\mathrm{mm}}$	$\mathrm{SW}_{\mathrm{cm}}$	$\mathrm{PW}_{\mathrm{cm}}$	$\mathrm{SW}_{\mathrm{mm}}$	$\mathrm{SW}_{\mathrm{mm}}$

Table 2: linear-SVMRFE and logRatio-SVMRFE rankings on Iris. PL:Petal Length; PW:Petal Width; SL:Sepal Length; SW:Sepal Width

	#Pos	#Neg	#Attribute	Data Type
Alt	1425	2732	2112	Text
Structure	2621	927	2368	Text
Disease	2606	631	2376	Text
Function	3089	818	2708	Text
Breast	46	51	24481	DNA-microarray
StrokeRaw	101	107	28664	Mass Spectra
OvarianRaw	162	91	15154	Mass Spectra
ProstateRaw	69	253	15154	Mass Spectra

Table 3: Real datasets summaries. Columns #Pos and #Neg indicate number of positive and negative instances. MS: mass spectrometry

Kernel	logRatio	lin	logLin	poly	logPoly
Alt	10.17	**13.80**	**12.53**	**28.57**	**16.55**
Disease	17.11	**19.74**	17.76	**19.15**	18.13
Structure	18.09	19.44	19.41	**22.35**	19.89
Function	19.17	19.98	19.91	20.78	20.01
Breast	28.86	29.89	32.98	40.20	41.23
StrokeRaw	18.26	15.38	16.34	**36.05**	**39.42**
OvarianRaw	0	1.18	0.39	**7.50**	0.79
ProstateRaw	7.45	6.52	7.76	10.24	10.24

Table 4: Classification error of SVM classifier estimated by 10-fold-cross-validation for several kernel functions. Bold characters show errors significantly higher than logRatio; no error is significantly lower (according to McNemar's test[3] with 0.05 significance level).

The ideal attribute selection algorithm should eliminate all cm-attributes and preserve mm-attributes, which contain the original information. Table 2 shows again that logRatio-SVMRFE behaves perfectly, as it places the four mm-attributes on top of its ranking. Linear-SVMRFE puts a cm-attribute in rank 3. This spurious behaviour is certainly due to the impact of noise on attribute normalisation. Since noise was added to the cm-attributes, their standard deviation is greater than that of the mm-attributes. As a consequence, different normalisation coefficients are assigned to the same information expressed differently, thus distorting the weights and the ranking produced by linear-SVM models. logRatio-SVMRFE doesn't suffer from this problem because it builds identical models whatever the normalisation performed.

Finally, in $\mathrm{iris}_{(\delta_{\mathrm{mm}}=5,\delta_{\mathrm{cm}}=0.5)}$, all attributes have identical noise levels (δ_{mm}=5mm is equivalent to δ_{cm}=0.5cm). Two variables expressed in different units are complementary because by averaging them we can reduce the impact of noise and improve precision of the real value. Thus no specific behaviour of the ideal attribute selection algorithm is expected a priori. However, we expect redundant attributes to be normalised by the same factor since they convey the same information and have identical noise levels. This is not what linear-SVMRFE does, as noise perturbs estimation of the attributes' standard deviation; e.g. $\mathrm{SL}_{\mathrm{cm}}$ has a standard deviation of 1.3 but $\mathrm{SL}_{\mathrm{mm}}$ has a standard deviation of 9.8. As previously, such normalisation differences can bias interpretation of linear-SVM but not of logRatio-SVM models.

To conclude, experiments on variants of the Iris dataset clearly show advantages of logRatio-SVMRFE and the inability of linear-SVMRFE to distinguish redundant variables. logRatio-SVMRFE displays the expected behaviour, which is not the case of linear-SVMRFE. This advantage certainly comes from the insensitivity of the method to data normalisation and from the pairwise combination of attributes which allows a more accurate detection of redundancy. However two questions remain open. The first concerns the relevance of selected attributes: we focused mainly on redundancy and still need to check the selected attributes' discriminatory power. The second issue concerns the ability of the logRatio kernel to deal with dimensionality higher than that of Iris. These questions are discussed in the next section where we study the classification error of the logRatio kernel on biological datasets of very high dimensionality.

3.2 logRatio for biological data/text mining

Our biological data/text mining experiments concern 8 high-dimensional datasets involving binary classification problems (Table 3). Four text classification datasets (Alt, Function, Disease and Structure) were constructed from biological corpora[8]. Three datasets contain raw mass spectrometry information (ovarian cancer, prostate cancer and stroke)[9]. Finally, Breast contains gene expression data from DNA microarrays (`http://sdmc.lit.org.sg/GEDatasets/Datasets.html`).

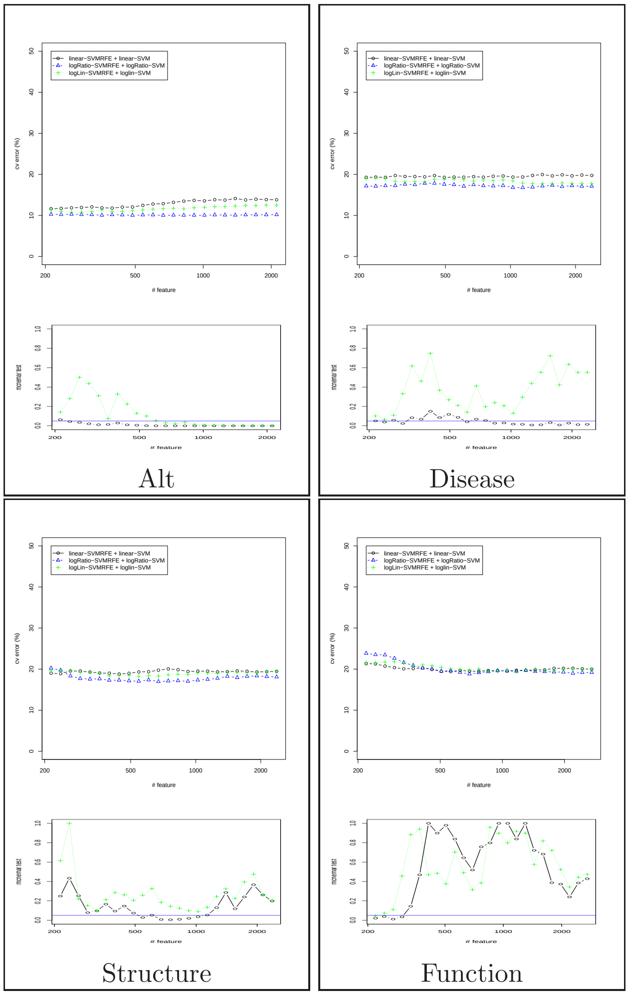
The instances of the mass spectrometry datasets are normalised by the "total ion current" in accordance with standard practice in the field. This is equivalent to setting the L1-norm of the instances to a constant value. The text datasets use the tf.idf representation, which already takes into account the size of the documents; thus we don't perform instance normalisation on this datasets. In the microarray dataset (Breast), each instance originally contained log ratios of DNA expression levels of an analysed sample to those of the reference sample. To remove negative values we applied the exponential function so that the resulting dataset simply contains ratios of expression levels.

All numerical values of these datasets are non-negative, but the presence of null values hinders the application of the logRatio kernel. In particular text datasets are very sparse: instances of Alt, Function, Structure and Disease have on average only 12 non null attributes. We eliminated null values by each of them replacing them with a positive value, ϵ, small compared to all others. More precisely, all values under the threshold ϵ were replaced by ϵ. The value $\epsilon = 0.01$ was found to be appropriate for all the datasets.

Experiments on these datasets had two goals. The first was to study the ability of logRatio to capture relevant information in high dimensional data; this was done by measuring logRatio-SVM's classification performance. At the same time we also get an indication of the quality of the representation that the logRatio uses. The second goal was to estimate the quality of our interpretation of SVM-logRatio models; this was done by measuring classification performance attained using attributes selected with logRatio-SVMRFE.

3.2.1 SVM-logRatio Classification performance achieved by logRatio-SVM is summarised in Table 4 and compared to that of lin-SVM and poly-SVM. The latter two were built respectively with a linear and a polynomial kernel of degree 2, and are similar in many ways to logRatio. Additionally, to further increase the similarity between the kernels, we effected a logarithmic transformation of the data before applying lin-SVM and poly-SVM (columns logLin and logPoly in Table 4). Attributes were standardised within the cross validation loop for all but the logRatio-based method, which has not need for this transformation. The table shows logRatio-SVM's excellent performance: its error rates are never significantly higher than those of the other kernels, but always equal or significantly lower. logRatio is particularly effective for text classification tasks, on which it consistently achieves the lowest errors (on Alt, significantly lower than the others). This comes as a surprise given the initial abundance of null values in these text datasets.

We see several reasons for these excellent results. First, logarithmic transformation of the values seems to play a beneficial role, as can be gathered by observing the significant error decrease of logLin vs lin and of log-



Alt

Disease

Structure

Function

Figure 3: Results of the evaluation of SVMRFE performance on text datasets. Top graphs show the evolution of classification errors as the number of selected attributes (n) increases; bottom graphs plot the significance level of differences in error between the logRatio kernel and the other two kernels (using McNemar test). When a point is below the horizontal line at 0.05, errors are considered significantly different.

Poly vs poly. Pairwise attribute combination also seems important, since logRatio-SVM performs better than linear kernels, which do not combine attributes. However, the way in which attributes are combined should also be taken into account: poly-SVM and logPoly-SVM, which consider pairwise products of attribute values, perform worse than logRatio-SVM. It is apparently better to use ratios of attributes, which nullify normalisation errors, rather than their products, which tend to amplify them.

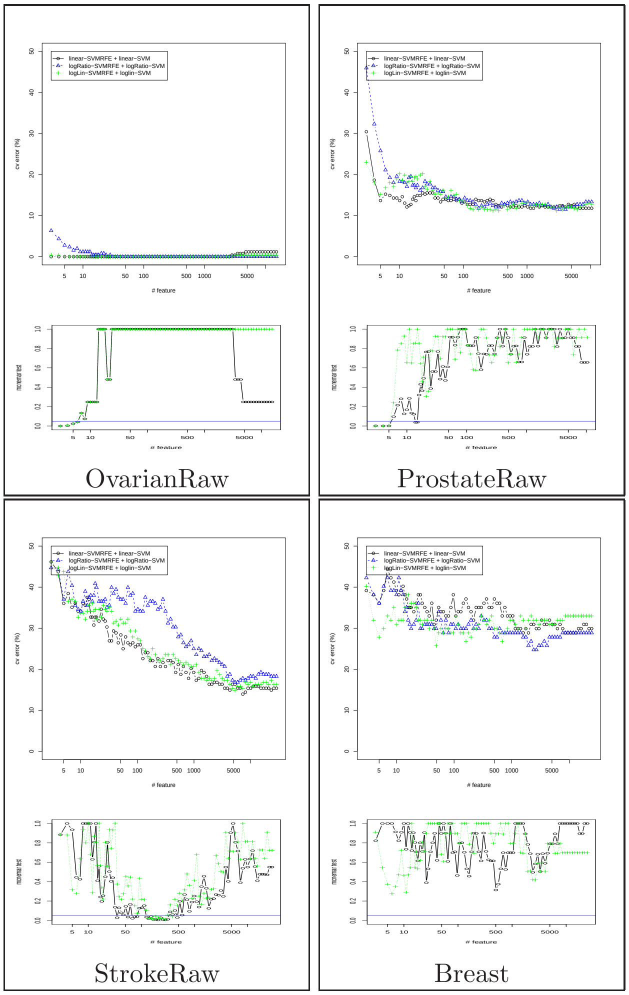
SVM-logRatio's edge over the other methods on the text datasets is most likely related to data normalisation problems in the presence of sparse data. In particular, attribute standardisation is certainly not adapted to the numerous null values (that we replace by ϵ) contained in the datasets because it biases the estimation of the standard deviation. It might be better to ignore those values in the estimation of the standard deviation. Note also that the number of instances in text datasets is much higher than in the gene/protein expression datasets; this allows us to distinguish classification algorithms with very close performance results.

The above results give us reasons to expect good behaviour from logRatio-SVMRFE: 1) logRatio-SVM shows the best classification performance; 2) it has perfectly eliminated redundancy on Iris; 3) interpretation of logRatio-SVM models should be more accurate than interpretation of lin-SVM models because logRatio is not sensitive to data normalisation. To check these expectations, the next section is devoted to a comparison of attribute selection algorithms based on the logRatio and linear kernels.

3.2.2 Attribute Selection with logRatio

We now focus on the evaluation of logRatio-SVMRFE and compare its performance to that of lin-SVMRFE and logLin-SVMRFE (obtained by applying lin-SVMRFE after logarithmic transformation of the data). In each iteration of the RFE loop, 10% of the attributes are removed to produce the rankings, then the n most relevant attributes are selected. Classification algorithms logRatio-SVM, lin-SVM and logLin-SVM are then trained on the subset of n best attributes selected by each method, and tested on a distinct validation set. The whole process is repeated ten times producing an estimation of the classification error by the resulting ten-fold cross validation procedure. Again, attributes are standardised within the cross validation loop for all but the logRatio kernel, which does not require this precaution. The classification error provides an estimation of the quality of attribute selection: the smaller the error, the more discriminatory the selected attributes.

To simplify results, we limit our study to three



OvarianRaw

ProstateRaw

StrokeRaw

Breast

Figure 4: Results of the evaluation of SVMRFE performance on expression dataset.

combinations of attribute selection and classification algorithms: logRatio-SVMRFE+logRatio-SVM, lin-SVMRFE+lin-SVM and logLin-SVMRFE+logLin-SVM. Each attribute selection algorithm is coupled with the classification algorithm for which it is expected to select appropriate attributes. We provide results for all the 8 datasets on Fig. 3 and Fig. 4. Finally, note that in the case of big datasets (Alt, Disease, Structure and Function), we were not able to compute results on subsets of fewer than 200 attributes due to excessive learning time. The algorithm implemented in *libsvm* to find the maximal margin hyperplane apparently fails to converge when fewer than 200 attributes are selected using logRatio kernel.

Figure 3 shows an interesting result on the Alt dataset: with more than 30% of attributes left (around 600 over 2112), logRatio is still significantly better than linear kernels. However, its performance remains stable as the number of attributes decreases whereas that of the linear kernels improves until the three performance levels almost coincide. On Disease, logRatio's performance also remains stable at acceptable levels whereas the error of the loglinear kernel increases abruptly with 300 selected attributes.

Despite satisfactory results on text datasets, we observed a tendency of logRatio-SVMRFE to perform worse than lin-SVMRFE when selecting attribute subsets of less than 10 attributes (e.g. OvarianRaw, ProstateRaw). This is due to a deterioration of logRatio's performance rather than an improvement of the linear kernel. Two reasons can explain this phenomenon. First, our method of ranking attributes from the logRatio-SVM model is inappropriate and leads to elimination of informative attributes. This hypothesis does not explain the good results obtained when selecting subsets of more than 10 attributes. The second, more plausible explanation is that, when fewer than 10 attributes are selected, they are all individually informative. Of all possible pairwise combinations of the 10 selected attributes, only 5 mutually disjoint pairs convey the full information, and all other pairs are of no use. Thus logRatio has to work with 5 information sources whereas the linear kernel can exploit 10.

4 Discussion and future work

There is yet another simpler alternative to the logRatio kernel; it is based on the observation that the logRatio kernel can be decomposed into a simpler kernel function, K_{cov}, applied to the log-transformed training instances. The mapping function, ϕ_{cov}, of the K_{cov} kernel, produces a vector of n^2 features containing all pair-wise differences of the n attributes of the initial log-transformed space:

$$\phi_{\text{cov}}(\mathbf{x}) = (x_i - x_j)_{(i,j)=(1,1)}^{(n,n)} \tag{4.12}$$

Then the K_{cov} kernel of two instances $\mathbf{x}, \mathbf{z}$ is given by:

$$\begin{aligned} K_{\text{cov}}(\mathbf{x}, \mathbf{z}) &= \sum_{i,j} (x_i - x_j)(z_i - z_j) \\ &= 2n(n-1)\,\text{cov}(\mathbf{x}, \mathbf{z}) \end{aligned}$$

The final kernel is simply given by:

$$K(\mathbf{x}, \mathbf{z}) = K_{\text{cov}}(\log \mathbf{x}, \log \mathbf{z})$$

where $\log \mathbf{x}$ is the component-wise application of the log function on each of the features of $\mathbf{x}$. The similarity of the K kernel with the logRatio allows us to follow the same reasoning and reach similar conclusions: 1) we can visualise the feature space in the same way that we did with logRatio, by plotting components of the $\mathbf{w}$ vector (without log transformation); 2) it is possible to perform feature selection in the same way as logRatio-SVMRFE, by removing attributes according to their distance to the median value. Furthermore, K has the advantage of not being limited to strictly positive values; but, unlike logRatio, K is sensitive to data normalisation. This drawback was the main reason we did not consider it further in our experiments.

The logRatio kernel offers several advantages in relation to the biological domain: it requires no data normalisation; it focuses learning on attribute-attribute interaction; it is simple to visualise and interpret; and finally it provides useful feed-back on redundancy of the attributes. Moreover, the logRatio-SVMRFE attribute selection algorithm is able to eliminate redundancy. The classification performance of logRatio-SVM is never lower than that of other tested kernels but systematically equal or higher. These results highlight the quality of the logRatio data representation.

On the other hand, classification performance obtained with attributes selected by logRatio-SVMRFE is low when we select attribute subsets of small cardinality. This finding calls into question the way we interpret logRatio-SVM models in order to rank attributes as well as the utility of the pairwise combination of attributes in real-world datasets. In future work, we intend to study alternative interpretations of the logRatio-SVM model in order to rank attributes, possibly in line with recent extensions to lin-SVMRFE[14].

An interesting feature of the models learnt by the logRatio-SVM is the clustering of the attributes according to their weight values in the models. Redundant attributes have similar weight values, i.e. cluster together, while non-redundant and relevant attributes have quite

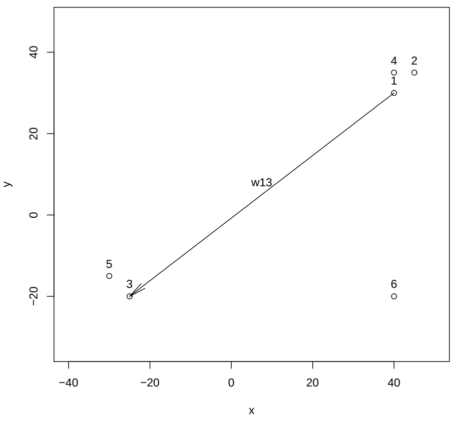


Figure 5: Example of attribute interaction map we would like to build from an SVM model. Each point should represent one of the 6 attributes of the initial space, and the distances the 6^2 linear coefficients of the SVM model for each feature of the feature space.

different weight values, i.e. do not cluster together. We would like to exploit further this model interpretation; however, the expressive power of logRatio visualisation is limited to a single dimension. Suppose we can propose an extension of the logRatio kernel in which the learnt model can be visualised in two (or more) dimensions. Each point will represent an attribute of the initial space, and the distance between two of them should be equal to the linear coefficient associated to the corresponding attribute combination. Figure 5 shows an example of the expected visualisation. Redundant attributes would be displayed close to each other, and relevant and non-redundant attributes far away. With a second degree of freedom, the visualisation would indeed convey more information.

This kind of technique would be an ideal bioinformatics tool for plotting gene/protein expression data (e.g. Breast, Stroke, Ovarian, Prostate we used in this article). Such visualisation provides simple and precious information to a biologist because it organises protein-protein (or gene-gene) interactions: attributes (e.g., genes or proteins) involved in interactions that are relevant to class prediction will be located far from each other, while those involved in interactions that are not relevant would be located close to each other.

This kind of visualisation is also specially interesting because it allows the end user to build his own classification model by selecting some of the attributes in the displayed clusters. In Figure 5, for example, we can remove two of the attributes in $\{1, 2, 4\}$ without adversely affecting predictive performance, since they are close to each and thus probably redundant. Ultimately, what is proposed to the user is not just a classification model but an attribute configuration that can be manipulated and adjusted in accordance with the user's domain knowledge.

References

[1] Olivier Chapelle and Vladimir Vapnik. Choosing multiple parameters for support vector machines. *AT&T Labs Technical Report*, 2000.

[2] N. et al. Cristianini. *An Introduction to Support Vector Machines and other kernel-based learning methods.* Cambridge University Press, 2002.

[3] T.G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923, 1998.

[4] Yuanyuan Ding and Wilkins Dawn. Improving the performance of svm-rfe to select genes in microarray data. *BMC Bioinformatics*, 7, 2006.

[5] C. Furlanello, M. Serafini, S. Merler, and G. Jurman. Entropy-based gene ranking without selection bias for the predictive classification of microarray data. *BMC Bioinformatics*, 6, 2003.

[6] Donald Geman, Christian d'Avigon, Daniel Q. Naiman, and Raimond L. Winslow. Classifying gene expression profiles from pairwise mrna comparison. *Statistical Applications in Genetics and Molecular Biology*, 3(1), 2004.

[7] Isabelle Guyon, Jason Weston, and Stephen Barnhill. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46, 2002.

[8] A. Kalousis, J. Prados, and M. Hilario. Stability of feature selection algorithms: a study on high-dimensional spaces. *Know. Inf. Sys.*, 2006.

[9] Julien Prados, Alexandros Kalousis, and Melanie Hilario. On preprocessing of SELDI-MS data and its evaluation. In *CBMS*, 2006.

[10] J. Quackenbush. Computational analysis of microarray data. *Nature Reviews — Genetics*, 2, 2001.

[11] Rakotomamonjy. Variable selection using svm-based criteria. *Journal of Machine Learning Research*, 3, 2003.

[12] A. Ultsch. Is log ratio a good value for identifying differential expressed genes in microarray experiments? *Statistical Computing 2006*, 2006.

[13] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for svms. *Proceedings of NIPS*, 13, 2000.

[14] X. Zhang, X. Lu, and Q. et al. Shi. Recursive svm feature selection and sample classificatio for mass-spectrometry and microarray data. *BMC Bioinformatics*, 7(197), 2006.