Feature Weighted SVMs Using Receiver Operating Characteristics

Shaoyi Zhang[†] M. Maruf Hossain[*†] Md. Rafiul Hassan[†] James Bailey[†‡]
Kotagiri Ramamohanarao[†‡]

Abstract
Support Vector Machines (SVMs) are a leading tool in classification and pattern recognition and the kernel function is one of its most important components. This function is used to map the input space into a high dimensional feature space. However, it can perform rather poorly when there are too many dimensions (e.g. for gene expression data) or when there is a lot of noise. In this paper, we investigate the suitability of using a new feature weighting scheme for SVM kernel functions, based on receiver operating characteristics (ROC). This strategy is clean, simple and surprisingly effective. We experimentally demonstrate that it can significantly and substantially boost classification performance, across a range of datasets.

Key words: Receiver Operating Characteristics, Distance Function, Support Vector Machine, Classification.

1 Introduction

Support Vector Machines (SVMs) are learning systems based on statistical learning theory and the principle of structural risk minimization (SRM). The SVM is very powerful, evidenced by the fact that within a few years of its introduction, it is in wide use and commonly outperforms other classifiers for a range of classification and recognition tasks [7]. Examples include isolated handwritten digit recognition, object recognition, speech recognition, and spatial data analysis.

The SVM is a supervised learning system where the training data is mapped into a high dimensional feature space, in which an optimal separating hyperplane between the two classes of the labeled data is obtained using quadratic programming. The hyperplane is then mapped back to the input space via an inverse mapping function and thus it becomes a non-linear decision system to separate/classify the input data. The kernel trick helps avoid explicit mapping [7], reducing unnecessary computation.

Although SVMs have been applied for classification in many domains, they tend to suffer from poorer classification accuracy when i) there are very many features, or ii) the data is very noisy. These weaknesses make it difficult to use SVMs for datasets such as gene expression data [25], which are very noisy and typically have thousands of features, but only tens or at most hundreds of instances. Indeed despite their popularity, in the gene expression domain there is still substantial room for improvement when using SVMs. Developing a clean and simple strategy for improvement is the main theme of this paper. E.g., for the gene expression dataset first discussed in [30], a massive accuracy improvement from 47.62% (for standard SVMs with polynomial kernel) to 96.48% (using our techniques with polynomial kernel) can be achieved.

Our aim is to test the validity and effectiveness of using a new feature weighting scheme for SVM kernel functions. Our proposed method has a number of desirable properties:

- It is clean, simple and intuitive.
- It is easy to implement and requires minimal change to any existing SVM implementation.
- It has a negligible effect on the time required for training an SVM or using it for test instance classification.
- It can substantially boost classification accuracy.
- It outperforms other possible feature weighting schemes, such as chi squared, information gain, mutual information and Fisher's criterion.

Our weighting for each feature is computed by considering the area under the Receiver Operating Characteristics (ROC) curve [15]. Interestingly, this value is equivalent to the Mann-Whitney U statistic normalized by the number of possible pairings of positive and negative values, also known as the two sample Wilcoxon rank-sum statistic [17]. The area under the ROC curve (AUC) actually represents the probability that a randomly chosen positive example is correctly ranked with greater suspicion than a randomly chosen negative example. Moreover, this probability of correct ranking is the same quantity estimated by the non-parametric Wilcoxon statistic [2].

[*]Corresponding author {hossain@csse.unimelb.edu.au}
[†]Department of Computer Science and Software Engineering, The University of Melbourne, Australia.
[‡] NICTA Victoria Laboratory, The University of Melbourne, Australia.

Related Work: To the best of our knowledge there is only one work which used feature weighting for computing kernel functions for SVMs. Kertész-Farkas and Kocsor [22] presented a method which weights the features according to their importance instead of removing the negligible ones via kernel functions. They used several feature ranking technique (i.e., standard deviation, Fisher's criterion, entropy) to weight features and showed that weighting kernel functions obtained a significantly better classification performance than that using the usual unweighted method. Although fuzzy or "weighted" SVMs do exist [24], the weights there are not applied to features, but are applied to instances from the dataset, to reflect the degree of confidence in membership.

Feature weighting has instead played more of a role in k-nearest neighbor classifiers, where it has been used by Vivencio *et al.* [34], who proposed a feature weighting method based on the χ-squared test. In another study, Kohavi *et al.* [23] investigate the use of a weighted Euclidean metric in which the weight for each feature comes from a small set of options. They described an algorithm that directs search through a space of discrete weights using cross-validation error as its evaluation function. Recent related work by Hassan *et al.* [18] has shown the benefits of considering ROC-characteristics in the context of distance functions for k-nearest neighbor.

Furlanello *et al.* [11] ranked features using a wrapper algorithm based on recursive feature elimination (RFE) using entropy of the relevant feature. This was based on work by Guyon *et al.* [16] who developed the baseline RFE technique for SVMs. A similar wrapper approach was also developed by Shieh and Yang [29] for multiclass scenarios using SVM-RFE. As these are all wrapper approaches, their effectiveness can vary according to the choice of classifier.

Brank and Milic-Frayling [3] proposed a framework for characterizing feature weighting methods and selected features sets and exploring how these characteristics account for the performance of a given classifier. They illustrated the use of two feature set statistics: cumulative information gain of the ranked features and the sparsity of data representation that results from the selected feature set.

Weston *et al.* [35] introduced a method of feature selection for SVMs, which is based upon nding those features using gradient descent that minimize bounds on the leave-one-out error. They have shown the resulting algorithms to be superior to some standard feature selection algorithms.

Cheng *et al.* [6] developed an incremental training algorithm for SVMs where the data samples were preclustered using a k-means algorithm. Although using a clustering algorithm could help eliminate outliers in the training data, significant instances might still be overlooked due to erroneous choices of the cluster parameter k.

Contributions: Our main contributions in this paper are as follows:

- Presenting an ROC-based feature weighting metric for SVM kernel functions to improve classification. This scheme is clean and simple, does not degrade SVM running time and is very easy to incorporate into existing implementations.
- An experimental investigation which demonstrates that our new algorithm (known as *ROC-SVM*) can deliver very substantial improvements compared to standard SVMs. These improvements are strongest when applied to gene expression datasets. *ROC-SVM* also often outperforms a range of existing classifiers such as C5.0, its predecessor C4.5, Random Forest, Ferri *et al.*'s [10] AUCsplit technique for decision trees, Naïve Bayes and SVMs using three different kernels: linear, polynomial and RBF.
- The experimental investigation also demonstrates that our new algorithm outperforms other possible SVM weighting techniques, such as chi-squared, information gain, mutual information and Fisher's criterion.

2 Preliminaries

In this section, we briefly introduce SVMs and ROC.

2.1 Support Vector Machine (SVM):

Let us consider, we have n vectors $\vec{x}_0$, $\vec{x}_1$, $\vec{x}_2$, …, $\vec{x}_{(n-1)}$ from the vector space $\mathbb{R}^m$ which belong to either of the two classes $+1, -1$. The data are labeled as follows:

$$(\vec{x}_j, y_j) : j = 0, \ldots, (n-1), \qquad \text{where } y_j \in \{+1, -1\}. \tag{2.1}$$

Our goal is to find a hyperplane as in Eq. (2.2), which would classify the input vector effectively:

$$\vec{w}^T \vec{x}_j + b = 0, \tag{2.2}$$

where $\vec{w}$ = weight vector and b = bias. Thus it is assumed that the two classes can be separated by considering two hyperplanes of the margin parallel to the hyperplane described by Eq. (2.2) [31]:

$$\begin{aligned} &\vec{w}^T \vec{x}_j + b \geqslant 1, && \text{for } y_j = +1 \\ &\vec{w}^T \vec{x}_j + b \leqslant -1, && \text{for } y_j = -1 \end{aligned} \tag{2.3}$$

where $j = 0, \ldots, (n-1)$.
By combining above two constraints we get:

$$y_j(\vec{w}^T \vec{x}_j + b) \geqslant 1, \qquad \text{for } j = 0, \ldots, (n-1). \tag{2.4}$$

Constraint (2.4) is valid for a dataset which is linearly separable. For such linearly separable data, the goal for an SVM is to find the optimal weight and bias values such that the obtained hyperplane separates the two classes of training data with a maximum margin. For a dataset which is not completely separable (i.e., data instances are badly scattered), a classification violation is allowed in the SVM formulation. For this case, (2.4) is modified to (2.5), where n non-negative (slack) variables ξ_j are introduced.

$$(2.5)\quad y_j(\vec{w}^T\vec{x}_j + b) \geqslant 1 - \xi_j, \qquad \text{for } j = 0, \ldots, (n-1).$$

In (2.5), those vectors $\vec{x}_j$, for which the value of ξ_j is greater than zero, does not satisfy Eq. (2.4). To compute the width $\gamma(\vec{w}, b)$ of the margin the following equation is used:

$$(2.6)\qquad \gamma(\vec{w}, b) = \min_{\{\vec{x}|y=+1\}} \frac{\vec{w}^T\vec{x}}{\|\vec{w}\|} - \max_{\{\vec{x}|y=-1\}} \frac{\vec{w}^T\vec{x}}{\|\vec{w}\|}.$$

An optimal hyperplane is found for the SVM by optimizing the following equations:

(2.7)
$$\text{minimize } \tau(\vec{w}, \xi) = \frac{1}{2}\|\vec{w}\|^2 + C\sum_{j=1}^{n-1} \xi_j$$
$$\text{subject to } y_j(\vec{w}^T\vec{x_j} + b) \geqslant 1 - \xi, \text{for } j = 0, \ldots, (n-1),$$

where C is a constant called as the regularization parameter. By adjusting this parameter value, we can maximize the performance of the SVM. By using the saddle point of the following Lagrange function, we can solve the minimization problem defined in Eq. (2.7).

$$(2.8)\quad L_P(\vec{w}, b, \alpha) = \frac{1}{2}\vec{w}^T\vec{w} - \sum_{j=0}^{n-1} \alpha_j(y_j(\vec{w}^T\vec{x}_j + b) - 1),$$

where $\alpha_j \geqslant 0$ are Lagrange multipliers. We compute the gradient of $L_P(\vec{w}, b, \alpha)$ with respect to $\vec{w}$ and b to obtain the optimized weight $\vec{w}^*$:

$$(2.9)\qquad \frac{\delta L_P}{\delta \vec{w}} = 0, \qquad \frac{\delta L_P}{\delta b} = 0.$$

By solving Eq. (2.9), we obtain the optimized weight $\vec{w^*}$:

$$(2.10)\qquad \vec{w^*} = \sum_{j=0}^{n-1} \alpha_j y_j x_j$$

considering the following constraint

$$(2.11)\qquad \sum_{j=0}^{n-1} \alpha_j y_j = 0.$$

Using Eq. (2.10) and Eq. (2.11) into Eq. (2.8), we get

$$(2.12)\qquad L_D(\alpha) = \sum_{i=0}^{n-1} \alpha_i - \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} \alpha_i\alpha_j y_i y_j \vec{x}_i^T\vec{x}_j$$

where the constraints are:
(2.13)
$$\sum_{j=0}^{n-1} \alpha_j y_j = 0 \text{ and } \alpha_j \geqslant 0, \qquad \text{where } j = 0, \ldots, (n-1).$$

The optimized bias b^* is computed using Eq. (2.14)
(2.14)
$$b^* = -\frac{1}{2}\left(\min_{\{\vec{x}_j|y_j=+1\}} \vec{w}^{*T}\vec{x}_j + \max_{\{\vec{x}_j|y_j=-1\}} \vec{w}^{*T}\vec{x}_j\right).$$

The solution is found by minimizing L_P and maximizing L_D. The points on the margin given by L_P and L_D exhibiting non-zero values for α_j are called the support vectors. Once the support vectors and values of bias b^* are determined the two classes are separated by investigating the signs of Eq. (2.15):

$$(2.15)\qquad f(x) = sign\left(\sum_{j=0}^{n-1} \alpha_j y_j \vec{x}_j{}^T\vec{x} + b^*\right).$$

2.1.1 Kernel functions: The above mentioned solution aims to linearly separate the dataset. For a non-linear decision system, one extends the method by mapping the data points into a high dimensional space, called the feature space, as in Eq. (2.16) [7].

$$(2.16)\qquad \vec{x}_i^T\vec{x}_j \rightarrow \langle\phi(\vec{x}_i).\phi(\vec{x}_j)\rangle.$$

To explicitly map the input space to a suitable feature space is troublesome and hence a kernel function is introduced where the mapping is done implicitly. The kernel function between two vectors $\vec{x}_i$ and $\vec{x}_j$ is defined as follows:

$$(2.17)\qquad K(\vec{x}_i, \vec{x}_j) = \langle\phi(\vec{x}_i).\phi(\vec{x}_j)\rangle.$$

Using Eq. (2.16) and Eq. (2.12) we get:

$$(2.18)\quad L_D(\alpha) = \sum_{i=0}^{n-1} \alpha_i - \frac{1}{2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} \alpha_i\alpha_j y_i y_j K(\vec{x}_i, \vec{x}_j).$$

The goal is to maximize the value of $L_D(\alpha)$ to optimize the SVM for a non-linear dataset. Solutions are available via quadratic programming. However, a proper choice of Kernel function is also necessary to obtain a better SVM. In fact, the kernel function determines the degree of similarity between two data vectors/instances. There have been many kernel functions proposed. The most popular and widely used kernels are the linear, polynomial and RBF kernels.

Consider two instances $\vec{x}_i \equiv \langle x_{i1}, x_{i2}, \ldots, x_{im}, Y_i \rangle$ and $\vec{x}_j \equiv \langle x_{j1}, x_{j2}, \ldots, x_{jm}, Y_j \rangle$ comprising $(m+1)$ features each and the last feature being the class label. Some different kernel functions are described below.

Euclidean Distance: The function for Euclidean distance is given in Eq. (2.19):

$$ED(\vec{x}_i, \vec{x}_j) = \sqrt{\sum_{k=1}^{m} (x_{ik} - x_{jk})^2}, \text{ where } k = 1, \ldots, m. \tag{2.19}$$

Linear Kernel: The function for linear kernel is given in Eq. (2.20):

$$\begin{aligned} LK(\vec{x}_i, \vec{x}_j) &= \vec{x}_i \cdot \vec{x}_j \\ &= \sqrt{\sum_{k=1}^{m} (x_{ik} \cdot x_{jk})}, \text{ where } k = 1, \ldots, m. \end{aligned} \tag{2.20}$$

Polynomial Kernel: The function for polynomial kernel is given in Eq. (2.21):

$$\begin{aligned} PK(\vec{x}_i, \vec{x}_j) &= (\vec{x}_i \cdot \vec{x}_j + 1)^d \\ &= (LK(\vec{x}_i, \vec{x}_j) + 1)^d, \end{aligned} \tag{2.21}$$

where d is the degree of the polynomial equation.

Radial Basis Function (RBF) Kernel: In RBF kernel, the similarity between two vectors $\vec{x}_i$ and $\vec{x}_j$ is calculated using Eq. (2.22):

$$\begin{aligned} RK(\vec{x}_i, \vec{x}_j) &= e^{\left(-\gamma \|\vec{x}_i - \vec{x}_j\|^2\right)} \\ &= e^{\left(-\gamma ED(\vec{x}_i, \vec{x}_j)^2\right)}, \text{ for } \gamma > 0, \end{aligned} \tag{2.22}$$

where γ is the width of the Gaussian.

2.2 Receiver Operating Characteristic Curves:

The Receiver Operating Characteristic (ROC) curve was first used in signal detection theory [15]. In machine learning, the ROC curve is used to evaluate the discriminative performance of binary classifiers. This is obtained by plotting the curve of the true positive rate (*Sensitivity*) versus the false positive rate (1 – *Specificity*) for a binary classifier by varying the discrimination threshold. Figure 1 shows an example ROC curve.

Prior to plotting the ROC curve, the *sensitivity* and *specificity* need to be calculated as follows:

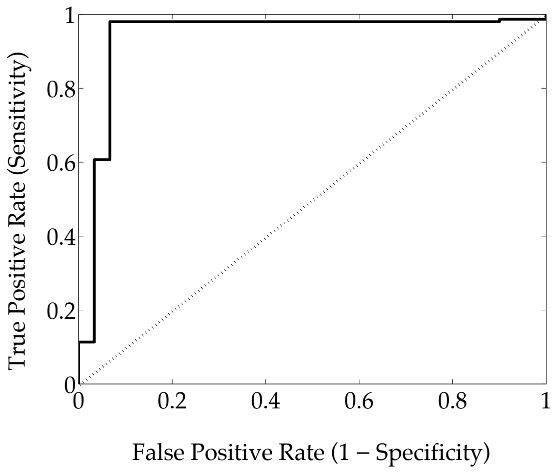


Figure 1: *A typical ROC curve*

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{2.23}$$

$$\text{Specificity} = \frac{TN}{FP + TN}, \tag{2.24}$$

where TP = True Positive, TN = True Negative, FP = False Positive and FN = False Negative.

It is to be noticed that, all the calculations above are attained when using a particular classifier threshold. By varying the threshold, a set of values for these measurements is obtained. This set of values is plotted in a two-dimensional Cartesian graph to yield the ROC curve. The ROC curve indicates the performance of the binary classifier, as it takes into account all the possible solutions by varying the discriminative threshold. The best performance would be produced, if the ROC curve meets the upper left corner of the ROC space (which yields 100% *sensitivity* and 100% *specificity*).

An ROC curve is a two dimensional illustration of classifier performance. Reducing ROC performance to a single scalar value to represent expected performance helps compare classifiers. An often used method is to calculate the area under the ROC curve (AUC) [17], [2]. There are several ways to calculate AUC. Of these, one way is to calculate using trapezoidal integration shown in Eq. (2.25) [28].

$$\begin{aligned} \text{AUC} &= \int_a^b f(\alpha) d\alpha \\ &\approx \sum_{i=1}^{n} \frac{h}{2} (f(a + (i-1)h) + f(a + ih)), \end{aligned} \tag{2.25}$$

where $\alpha = (1 - \text{Specificity})$, $a = 0$, $b = 1$,

$n =$ size to increase, and $h = \frac{b-a}{n}$.

Prior to calculating the AUC, the set of the values of *sensitivity* and *specificity* is normalized to the range [0, 1]. The AUC, being a part of the area of the unit square, has a value between 0 and 1. Since random guessing could produce the diagonal line between $(0,0)$ and $(1,1)$ with an area of 0.5, a classifier with an AUC less than 0.5 is undesirable [9]. A value of AUC equal to 1.0 represents that the performance of the binary classifier is 100%, i.e., the classifier can discriminate the dataset accurately.

3 Distance Weighting using ROC Information

We now describe the steps in our algorithm.

3.1 ROC for Feature Weighting: Previous work [20] has established the use of an ROC curve for feature ranking and selection. First, the ROC curve is plotted for each of the pairs formed by each of the features and the class label. This, in turn, means treating a single feature as a classifier and calculating the classification performance in terms of the *sensitivity* and *specificity* by varying the operating point. We shall build on this kind of idea to derive a feature weighting method to use in distance functions. For each feature, the AUC is calculated.

Let us consider a dataset $\mathcal{D}$ of N instances, where each instance comprises m features: $x_1, x_2, x_3, \ldots, x_m$. Each of the m features has a differing discriminative power reflected by its respective AUC. To calculate the discriminative power that is expressed in terms of AUC, we plot the ROC curve for each feature paired with the class label, (i.e., $\{x_i, Y_i\}$, where $1 \leqslant i \leqslant m$ and $\vec{Y}$ is the vector of class labels). I.e., we draw an ROC curve showing the performance of feature x_i as a classifier to predict the class label $\vec{Y}$. We then calculate the AUC of this ROC curve, which yields a measure of the power of feature x_i.

As described earlier, there is a strong mathematical justification for using the ROC to measure discriminative power. It is equivalent to the Mann-Whitney U Test (also known as Wilcoxon Rank sum), a non-parametric statistical test. Not employing any distributional assumptions makes it especially useful for small sample size, noisy datasets [8], such as gene expression microarrays.

3.2 Weighted Distance Metrics: For computing the distance between two instances, we modify the standard distance measure using the AUC scores as weights.

Consider two instances $\vec{x}_i \equiv \langle x_{i1}, x_{i2}, \ldots, x_{im}, Y_i \rangle$ and $\vec{x}_j \equiv \langle x_{j1}, x_{j2}, \ldots, x_{jm}, Y_j \rangle$ comprising $(m+1)$ features each and the last feature being the class label. Let, α_k be the AUC value of the k-th feature, for $k = 1, \ldots, m$. Then, the feature weighted different kernel functions are:
Weighted Euclidean distance,

$$ED_\alpha(\vec{x}_i, \vec{x}_j) = \sqrt{\sum_{k=1}^{m} \alpha_k (x_{ik} - x_{jk})^2}. \tag{3.26}$$

Weighted linear kernel,

$$LK_\alpha(\vec{x}_i, \vec{x}_j) = \sqrt{\sum_{k=1}^{m} \alpha_k (x_{ik} \cdot x_{jk})}. \tag{3.27}$$

Weighted polynomial kernel,

$$PK_\alpha(\vec{x}_i, \vec{x}_j) = (LK_\alpha(\vec{x}_i, \vec{x}_j) + 1)^d, \tag{3.28}$$

where d is the degree of the polynomial equation.
Weighted RBF kernel,

$$RK_\alpha(\vec{x}_i, \vec{x}_j) = e^{\left(-\gamma ED_\alpha(\vec{x}_i, \vec{x}_j)^2\right)}, \text{ for } \gamma > 0, \tag{3.29}$$

where γ is the width of the Gaussian.

Note that if $\alpha = 1$ for all features, then all features are equally important and no feature weighting is actually being employed.

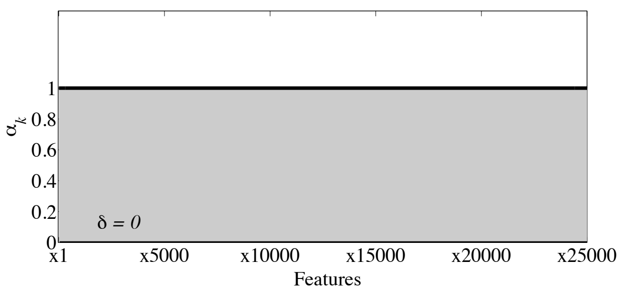
3.3 Weighted Distance with Threshold Function: To avoid using noisy features with lower predictive power and to put more importance on the features with higher discriminative power, we next define a threshold function $U_\delta(\alpha)$, to assist with feature pruning. The domain of this function is the set of feature weights and the range is a revised weight for each feature.

$$U_\delta(\alpha) = \begin{cases} 0, & \text{if } \alpha \leqslant \delta, \\ \alpha, & \text{if } \alpha > \delta; \end{cases} \tag{3.30}$$

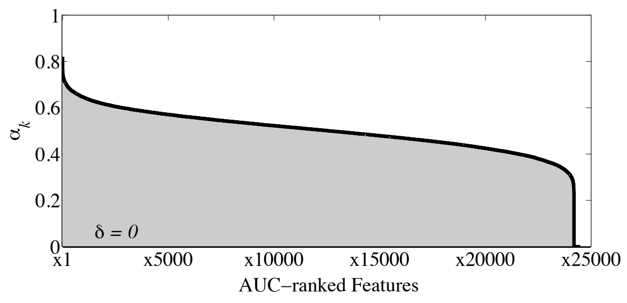
where $\alpha =$ the level of importance for a feature. This is given by either by the feature's AUC value or by using the same default level of importance for each feature (e.g., 1). δ is an AUC-based quality threshold parameter, where $0 \leqslant \delta \leqslant 1$.

Using the same weighting framework as earlier, we can now further modify the feature weights to be $U_\delta(\alpha_k)$ instead of α_k in Eq. (3.26), (3.27), (3.28) and (3.29). So the modified kernels then become:
Weighted Euclidean distance,

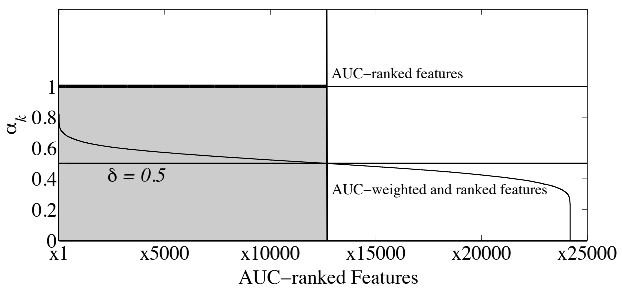
$$ED_\delta(\vec{x}_i, \vec{x}_j) = \sqrt{\sum_{k=1}^{m} U_\delta(\alpha_k)(x_{ik} - x_{jk})^2}. \tag{3.31}$$



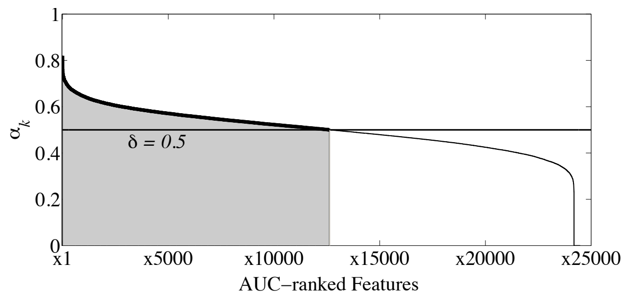
(a) Case 1: No Weighting, No Pruning



(b) Case 2: ROC Weighting



(c) Case 3: ROC-based Feature Pruning



(d) Case 4: ROC-based Feature Weighting and Pruning

Figure 2: *Variation of $U_\delta(\alpha)$ to obtain different SVM classifiers, where the x-axis represents all features ranked on their AUC value and the y-axis represents the AUC value for the features. A line parallel to the x-axis for some arbitrary value of δ is drawn to find a cut-off point from the weighted-ranked list (Case 4) and a line orthogonal to the δ-line is used to get a cut-off point from the ranked feature list (Case 3).*

Weighted linear kernel,

$$LK_\delta(\vec{x}_i, \vec{x}_j) = \sqrt{\sum_{k=1}^{m} U_\delta(\alpha_k)(x_{ik} \cdot x_{jk})}. \tag{3.32}$$

Weighted polynomial kernel,

$$PK_\delta(\vec{x}_i, \vec{x}_j) = (LK_\delta(\vec{x}_i, \vec{x}_j) + 1)^d, \tag{3.33}$$

where d is the degree of the polynomial equation.
Weighted RBF kernel,

$$RK_\delta(\vec{x}_i, \vec{x}_j) = e^{\left(-\gamma ED_\delta(\vec{x}_i, \vec{x}_j)^2\right)}, \text{ for } \gamma > 0, \tag{3.34}$$

where γ is the width of the Gaussian.

As discussed earlier, an AUC value closer to 1 indicates that the selected feature can be considered more important in discriminating the dataset, while a smaller AUC value indicates lower importance. So, the varying weights can be associated with the respective features to improve the performance of a classifier.

For the same setup, let α_k be the AUC value of k-th feature, where $k = 1, \ldots, m$. Now, this threshold function, $U_\delta(\alpha)$, can be represented by four main cases, according to the input that is passed to it.
Case 1: $U_0(1)$*:* if $\delta = 0$ and $\alpha = 1 \Rightarrow$ No Weighting, No Pruning,
Case 2: $U_0(\alpha_k)$*:* if $\delta = 0$ and $\alpha = \alpha_k \Rightarrow$ ROC Weighting,
Case 3: $U_\delta(1)$*:* if $\delta \neq 0$ and $\alpha = 1 \Rightarrow$ ROC-based Feature Pruning,
Case 4: $U_\delta(\alpha_k)$*:* if $\delta \neq 0$ and $\alpha = \alpha_k \Rightarrow$ ROC-based Feature Weighting and Pruning.

A graphical representation of these cases is shown in Fig. 2, which shows the effect of ROC-based feature weighting and how the feature pruning is done. Note that the grayed region is only for reference purposes. The actual features used by the classifier are presented by the bold line above the grayed region.

4 Experimental Setup and Datasets

We investigate the four cases for $U_\delta(\alpha_k)$ and compare the accuracy using a paired t-test (corrected) [12]. The kernels we use are linear, polynomial and RBF. We filter out all those features having an AUC score equal to or less than a quality threshold, δ. We started from 0.4 and kept increasing this threshold value by 0.1 and noted the classification performance on a separate validation set (not the test set) for both SVM and ROC weighted feature pruned SVM classifiers. For a 10-fold cross validation, we split the dataset into 10 random

Table 1: *Properties of the datasets used in this study*

Dataset	No. of Attributes	No. of Instances	Collected from	First used by
GE1	24,481	97	Integrated Tumor Transcriptome Array and Clinical data Analysis database [1]	van 't Veer *et al.* [33]
GE2	3,226	22	National Human Genome Research Institute	Hedenfalk *et al.* [19]
GE3	12,533	181	Division of Thoracic and Surgery [32], Brigham Women's Hospital, Boston	Gordon *et al.* [14]
GE4	12,600	21	Cancer Program [5], Broad Institute of MIT and Harvard	Singh *et al.* [30]
GE5	12,600	136	Cancer Program [5], Broad Institute of MIT and Harvard	Singh *et al.* [30]
GE6	7,129	72	Cancer Program [4], Broad Institute of MIT and Harvard	Golub *et al.* [13]
Hepatitis	19	155	UCI ML Repository [26]	–
Ionosphere	34	351	UCI ML Repository [26]	–
Pima	8	768	UCI ML Repository [26]	–
WBC	9	699	UCI ML Repository [26]	–
WDBC	30	569	UCI ML Repository [26]	–
WPBC	33	198	UCI ML Repository [26]	–

folds, and trained the classifier with 8 folds, kept 1 fold for internal validation (i.e., finding the best value for parameters) and 1 fold for the final testing of the classifier performance. Figure 3 shows how the accuracy of the ROC weighted SVMs was affected by varying the value of this parameter, for a number of datasets (datasets described shortly).

We also used four popular methods: χ^2 [34], information gain (IG) [21], mutual information (MI) [36] and Fisher's criterion (signal-to-interference ratio) [27], which have been used in the literature to weigh features, and compared their performance against our feature weighting technique for SVMs. In addition to comparing against traditional SVMs, we compared *ROC-SVM* against twelve other classifiers. These are: ROC-*k*NN, ROC-tree [20], *k*-NN, Ferri *et al.*'s [10] AUCsplit technique for decision trees, C5.0, its predecessor C4.5, ADTree, REPTree, Random Tree, Random Forest and Naïve Bayes. Where applicable, each of these classifiers was run multiple times on each dataset by varying its parameters. We report the best result of each such classifier on each dataset across the variation of its parameters.

Each of the classifiers was applied on 12 datasets, of which 6 were gene expression datasets and 6 are non-gene expression datasets having rather different characteristics. The properties of the datasets are illustrated in Tab. 1. For each classifier and dataset, a 10-fold cross validation scheme was used 10 times.

Using a fixed 10-fold cross validation scheme, we also conducted a win-draw-loss analysis based on a paired *t*-test (corrected) [12] with 5% significance level, for the classifiers.

5 Results and Discussion

The classification accuracies of the polynomial and RBF kernels for the four cases are presented in Tab. 2 and Tab. 3. The linear kernel, too, has similar performances (not shown) for the four cases. The classification results for all techniques on the considered datasets are presented in Tab. 4 and 5 and in those two tables, for *ROC-SVM*, the presented results were obtained using the best of three kernels, quality threshold and other SVM parameters (e.g., the margin of the model (C) for all SVMs, the degree of polynomial for the polynomial kernel, and the width of the Gaussian (γ) for RBF kernel).

Four Cases of ROC-effected SVMs: The classification performance of *ROC-SVM* (Case 4) shows remarkable improvement over the traditional SVM techniques (see Tab. 2 and 3). For most datasets and the three different kernels, *ROC-SVM* has a statistically significant improved accuracy over SVM (Case 1). For example, on GE2 and GE4 datasets (see Tab. 4), the classification accuracy increased from 68.18% and 61.90% (for SVM) to 92.73% and 97.14 (for *ROC-SVM*), respectively; which is about 24.55% and 35.24% higher accuracy.

The classification accuracies of ROC-weighted SVM (Case 2) and ROC feature pruned SVM (Case 3) are better than that of SVM (Case 1). However, they are not always statistically significant within 5% significance level. Though the classification accuracies of ROC-weighted SVM (Case 2) and ROC feature

Table 2: *Comparison of accuracy results from* 10×10*-fold cross validation on all datasets for all 4 cases for Polynomial kernel*

Datasets	SVM (Case 1)	ROC-weighted SVM (Case 2)	ROC feature pruned SVM (Case 3)	ROC-weighted and feature pruned SVM (ROC-SVM) (Case 4)
GE1	68.04 ± 2.14	$70.31 \pm 1.98^{*}$	$79.17 \pm 1.91^{*\bullet}$	$\mathbf{80.62 \pm 1.34}^{*\bullet\circ}$
GE2	59.09 ± 2.98	$65.00 \pm 5.61^{*}$	$90.48 \pm 0.0^{*\bullet}$	$\mathbf{92.73 \pm 2.49}^{*\bullet\circ}$
GE3	99.45 ± 0.11	99.39 ± 0.17	99.50 ± 0.32	$\mathbf{100 \pm 0.0}$
GE4	47.62 ± 5.63	$53.00 \pm 3.50^{*}$	$96.00 \pm 4.59^{*\bullet}$	$\mathbf{96.48 \pm 0.0}^{*\bullet}$
GE5	91.18 ± 3.12	91.63 ± 0.99	90.96 ± 0.58	$\mathbf{92.15 \pm 0.40}^{*\circ}$
GE6	98.61 ± 1.26	98.03 ± 0.73	98.59 ± 0.0	$\mathbf{98.61 \pm 0.0}$
Hepatitis	76.77 ± 4.23	$81.11 \pm 1.66^{*}$	$83.90 \pm 0.85^{*\bullet}$	$\mathbf{85.21 \pm 0.72}^{*\bullet\circ}$
Ionosphere	88.60 ± 2.43	$90.60 \pm 0.81^{*}$	$92.69 \pm 0.69^{*\bullet}$	$\mathbf{95.05 \pm 0.46}^{*\bullet\circ}$
WBC	96.85 ± 1.07	96.78 ± 0.22	96.56 ± 0.24	$\mathbf{97.44 \pm 0.34}^{*\bullet\circ}$
WDBC	97.72 ± 1.04	97.84 ± 0.17	97.96 ± 0.15	$\mathbf{98.87 \pm 0.05}^{*\bullet\circ}$
WPBC	76.26 ± 4.78	$78.33 \pm 0.79^{*\circ}$	76.14 ± 0	$\mathbf{78.86 \pm 0.38}^{*\circ}$
Pima Indian	77.34 ± 5.01	$77.09 \pm 0.82^{\circ}$	76.90 ± 0.26	$\mathbf{79.45 \pm 0.66}^{*\bullet\circ}$

* statistically significant within 5% significance level over SVM
$^{\bullet}$ statistically significant within 5% significance level over ROC-weighted SVM
$^{\circ}$ statistically significant within 5% significance level over ROC feature pruned SVM

Table 3: *Comparison of accuracy results from* 10×10*-fold cross validation on all datasets for all 4 cases for RBF kernel*

Datasets	SVM (Case 1)	ROC-weighted SVM (Case 2)	ROC feature pruned SVM (Case 3)	ROC-weighted and feature pruned SVM (ROC-SVM) (Case 4)
GE1	67.01 ± 2.36	$69.59 \pm 2.50^{*}$	$80.63 \pm 1.72^{*\bullet}$	$\mathbf{81.65 \pm 1.13}^{*\bullet}$
GE2	63.64 ± 0.94	$65.90 \pm 0.0^{*}$	$83.33 \pm 8.33^{*\bullet}$	$\mathbf{88.38 \pm 2.79}^{*\bullet\circ}$
GE3	98.34 ± 1.41	98.39 ± 0.18	$99.26 \pm 0.39^{*\bullet}$	$\mathbf{99.34 \pm 0.25}^{*\bullet}$
GE4	61.90 ± 1.39	$65.00 \pm 0.0^{*}$	$86.50 \pm 2.42^{*\bullet}$	$\mathbf{97.14 \pm 2.61}^{*\bullet\circ}$
GE5	69.12 ± 5.31	$91.26 \pm 0.84^{*\circ}$	$90.67 \pm 0.89^{*}$	$\mathbf{91.77 \pm 0.61}^{*\circ}$
GE6	80.56 ± 2.18	$95.63 \pm 1.68^{*}$	$98.45 \pm 0.45^{*\bullet}$	$\mathbf{98.61 \pm 0.0}^{*\bullet}$
Hepatitis	84.52 ± 4.02	$84.16 \pm 0.76^{\circ}$	83.57 ± 1.59	$\mathbf{85.38 \pm 1.02}^{*\circ}$
Ionosphere	91.74 ± 5.15	$94.80 \pm 0.37^{*}$	$94.77 \pm 0.49^{*}$	$\mathbf{95.05 \pm 0.40}^{*\bullet\circ}$
WBC	96.85 ± 1.29	96.85 ± 0.18	96.85 ± 0.18	$\mathbf{97.98 \pm 0.73}$
WDBC	96.07 ± 3.12	97.83 ± 0.27	97.99 ± 0.38	$\mathbf{98.87 \pm 0.35}^{*}$
WPBC	77.02 ± 2.36	$78.78 \pm 1.01^{\circ}$	76.60 ± 0.37	$\mathbf{78.78 \pm 0.33}^{*\circ}$
Pima Indian	77.47 ± 3.73	77.35 ± 0.38	$87.34 \pm 0.50^{*\bullet}$	$\mathbf{89.74 \pm 0.81}^{*\bullet\circ}$

Same legends as in Tab. 2.

pruned SVM (Case 3) are often very similar, they can vary greatly for some gene expression datasets with a very large feature space. This is due to the presence of some irrelevant features, which affects negatively to the distance calculation, even though we try to minimize their effects by weighting the features. *ROC-SVM* (Case 4) improves the accuracy significantly within 5% significance level over the ROC-weighted SVM (Case 2) and ROC feature pruned SVM (Case 3) for most datasets. Therefore, it becomes obvious that using both feature weighting and pruning is a better choice than using nothing at all when computing distances for the kernel function.

ROC-SVM versus other feature weighted SVMs: Looking at the top half of Tables 4 and 5, we can compare standard SVMs against SVMs using a varierty of different weighting techniques. In general, using some kind of weighting can increase accuracy. However, ROC-weighted SVMs are consistently the best overall. A win-draw-loss analysis of all classifiers shown in Table 6 also confirms the edge that ROC-weighting has over the other forms of feature weighting.

ROC-SVM versus non-SVM Classifiers: Surprisingly and pleasingly, the classification performance of *ROC-SVM* on all twelve datasets is the best (see Tab. 4 and 5). It outperforms all the other classifiers used in this study. This provides some evidence that *ROC-SVM* is a classifier which is able to surpass mainstream state-of-the-art techniques in these circumstances.

Statistical Significance Tests: We also carried out win-draw-loss analysis based on paired (corrected) *t*-test with 5% significance level for the considered clas-

Table 4: *Comparison of accuracy results from* 10×10*-fold cross validation on six gene expression datasets*

Method	GE1	GE2	GE3	GE4	GE5	GE6
ROC-SVM	$\mathbf{83.30 \pm 1.99}$	$\mathbf{92.73 \pm 2.49}$	$\mathbf{100 \pm 0.0}$	$\mathbf{97.14 \pm 2.61}$	$\mathbf{92.15 \pm 0.40}$	$\mathbf{98.61 \pm 0.0}$
χ^2-FW-SVM	69.11 ± 0.11	63.64 ± 0.0	98.34 ± 0.74	61.90 ± 0.0	91.21 ± 0.55	97.78 ± 0.76
IG-FW-SVM	68.04 ± 1.07	63.64 ± 0.0	99.45 ± 0.0	61.90 ± 0.0	91.18 ± 0.89	$\mathbf{98.61 \pm 0.0}$
MI-FW-SVM	76.29 ± 0.98	86.36 ± 1.28	99.45 ± 0.0	76.19 ± 2.74	91.18 ± 1.01	$\mathbf{98.61 \pm 0.0}$
Fisher's Criterion	78.44 ± 4.56	86.36 ± 2.33	99.45 ± 0.0	61.90 ± 0.59	91.18 ± 0.28	$\mathbf{98.61 \pm 0.0}$
SVM	68.04 ± 1.98	68.18 ± 1.18	$\mathbf{100 \pm 0.0}$	61.90 ± 1.39	91.18 ± 3.12	$\mathbf{98.61 \pm 0.76}$
ROC-tree	72.16 ± 4.32	77.27 ± 2.45	98.34 ± 0.89	38.10 ± 5.95	88.24 ± 2.33	94.44 ± 2.96
ROC-kNN	63.29 ± 3.13	71.07 ± 4.22	98.66 ± 0.39	61.49 ± 2.12	84.65 ± 1.22	90.33 ± 0.89
k-NN	58.45 ± 2.88	61.82 ± 6.14	94.64 ± 0.27	57.14 ± 3.89	82.35 ± 1.80	88.89 ± 0.93
AUCsplit	63.58 ± 4.59	74.39 ± 1.63	96.14 ± 1.36	34.01 ± 2.87	82.47 ± 3.96	81.61 ± 3.28
C5.0	64.95 ± 6.21	59.09 ± 4.52	92.82 ± 1.21	23.81 ± 4.65	81.62 ± 4.12	80.55 ± 3.74
C4.5	62.89 ± 3.11	72.73 ± 1.36	95.03 ± 1.05	33.33 ± 4.59	79.42 ± 5.45	79.17 ± 4.87
ADTree	61.86 ± 4.28	68.18 ± 5.68	92.82 ± 2.19	32.86 ± 3.44	86.76 ± 2.63	86.11 ± 3.77
REPTree	52.18 ± 5.45	59.09 ± 3.92	95.03 ± 1.28	32.86 ± 3.46	80.88 ± 3.33	81.94 ± 4.26
Random Tree	55.67 ± 3.54	63.64 ± 2.58	79.56 ± 2.69	32.86 ± 3.12	62.50 ± 5.23	75.00 ± 3.90
Random Forest	62.89 ± 6.43	50.00 ± 5.33	93.92 ± 1.22	38.10 ± 5.27	80.88 ± 2.56	79.17 ± 2.36
Naïve Bayes	54.64 ± 3.38	59.09 ± 4.58	98.34 ± 0.03	33.33 ± 0.78	55.88 ± 4.76	98.61 ± 1.03

Table 5: *Comparison of accuracy results from* 10×10*-fold cross validation on six non-gene expression datasets*

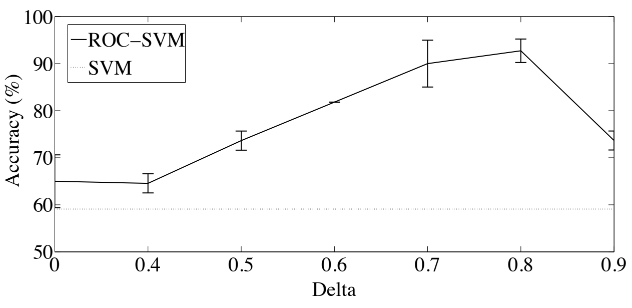
Method	Hepatitis	Ionosphere	WBC	WDBC	WPBC	Pima
ROC-SVM	$\mathbf{85.38 \pm 1.02}$	$\mathbf{95.05 \pm 0.40}$	$\mathbf{97.98 \pm 0.73}$	$\mathbf{98.87 \pm 0.05}$	$\mathbf{78.86 \pm 0.38}$	$\mathbf{89.74 \pm 0.81}$
χ^2-FW-SVM	84.00 ± 2.11	92.76 ± 0.32	96.71 ± 0.25	97.96 ± 0.16	77.02 ± 0.21	77.47 ± 0.31
IG-FW-SVM	81.29 ± 1.29	91.74 ± 2.38	97.00 ± 0.05	97.72 ± 0.06	77.78 ± 0.28	77.86 ± 1.47
MI-FW-SVM	81.29 ± 1.11	88.89 ± 2.13	97.00 ± 0.0	97.72 ± 0.04	76.26 ± 0.08	77.86 ± 1.55
Fisher's Criterion	81.93 ± 0.98	88.89 ± 1.11	97.00 ± 0.21	97.89 ± 0.33	78.79 ± 0.22	77.86 ± 1.16
SVM	84.52 ± 4.02	91.74 ± 5.15	96.85 ± 1.07	97.72 ± 1.04	77.02 ± 2.36	77.47 ± 3.73
ROC-tree	78.71 ± 7.65	84.05 ± 9.87	92.56 ± 5.43	90.69 ± 6.78	69.67 ± 8.33	63.54 ± 8.65
ROC-kNN	82.98 ± 0.87	88.60 ± 0.12	97.07 ± 0.14	97.47 ± 0.24	77.02 ± 0.68	86.60 ± 1.09
k-NN	82.58 ± 0.80	87.35 ± 0.59	96.88 ± 0.39	97.24 ± 0.29	76.21 ± 1.31	85.28 ± 1.87
AUCsplit	82.10 ± 3.43	86.00 ± 7.31	95.88 ± 1.94	93.75 ± 3.39	70.53 ± 9.67	73.82 ± 5.35
C5.0	76.13 ± 2.35	89.46 ± 1.23	93.64 ± 1.65	93.29 ± 2.23	70.70 ± 4.12	73.94 ± 2.76
C4.5	80.00 ± 4.45	91.45 ± 3.36	93.84 ± 2.63	93.15 ± 1.26	75.25 ± 3.32	73.83 ± 2.89
ADTree	76.13 ± 2.96	93.16 ± 1.65	95.14 ± 1.77	94.02 ± 1.06	77.78 ± 5.42	72.92 ± 3.23
REPTree	78.71 ± 4.23	89.46 ± 1.46	93.99 ± 2.14	92.44 ± 2.33	73.74 ± 4.85	75.39 ± 4.55
Random Tree	72.91 ± 9.21	87.75 ± 3.64	94.13 ± 2.85	89.46 ± 3.67	68.18 ± 5.45	67.97 ± 6.49
Random Forest	81.94 ± 1.26	92.59 ± 3.26	95.99 ± 1.45	95.25 ± 1.37	78.28 ± 3.47	73.70 ± 4.98
Naïve Bayes	83.87 ± 1.71	82.62 ± 3.48	95.99 ± 0.74	92.97 ± 2.58	67.68 ± 5.08	76.30 ± 3.49

sifiers (see Table 6). In this analysis, *ROC-SVM* arguably outperforms all the other techniques, as it has a significant number of wins and no losses at all, compared to the other classifiers. Against other 4 feature weighted SVMs, ROC-based weighting has 5 wins over 12 datasets. Of which, three for gene expression datasets, namely, GE1, GE2 and GE4 and two for non-gene expression datasets (Ionosphere and Pima indians). No other feature weighting scheme has any wins over the other feature weighted SVMs at all.

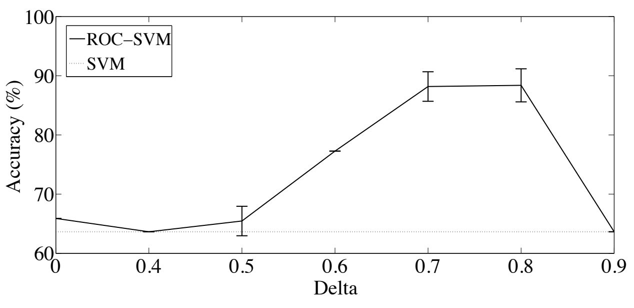
Comparison of AUC Values: We also computed the overall AUC values of all SVM-based classifiers and non-SVM classifiers (not shown), resulting from the 10×10-fold cross validation over all twelve datasets. The trends of the results for the AUC metric are quite similar to the accuracy.

Influence of the δ parameter: Figure 3 shows the effect on classification performance for varying the quality threshold (δ) for different kernels on the validation set of GE2 dataset along with the improvement over SVM, which is represented by a dotted line in the figures. We have, in fact, performed similar experiments on all considered datasets and observed similar behaviors.

Though the effect of δ is not entirely the same on every dataset, there are some clear trends. For gene expression data, $0.7 \leqslant \delta \leqslant 0.9$ is (unsurprisingly) always the best choice, due to the small number of instances in this type of data and classification accuracy

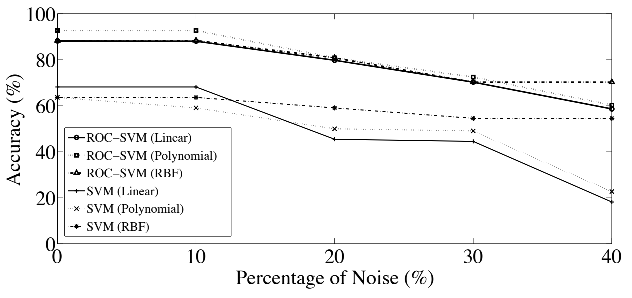


(a) Polynomial Kernel

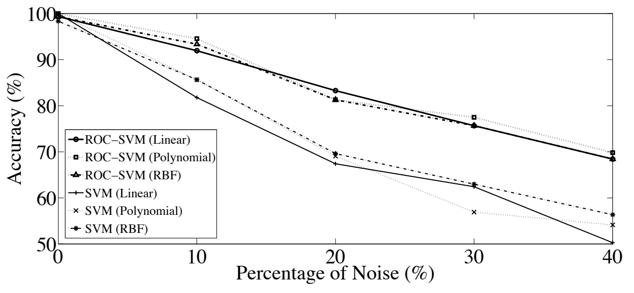


(b) RBF Kernel

Figure 3: *The effect of δ on classification accuracy for the GE2 dataset*



(a) GE2



(b) GE3

Figure 4: *Comparison of the accuracies of the classifiers: ROC-SVM and SVM with linear, polynomial and RBF kernels on gene expression datasets*

Table 6: *Win-Draw-Loss results for the seventeen classifiers using paired t-test (corrected) on 192 test combinations for each classifier*

Method	Win	Draw	Loss
ROC-SVM	**111**	**81**	**0**
χ^2-FW-SVM	57	119	16
IG-FW-SVM	56	118	18
MI-FW-SVM	75	110	7
Fisher's Criterion	79	105	8
SVM	57	118	17
ROC-tree	58	118	16
ROC-kNN	73	109	10
k-NN	49	98	45
AUCsplit	30	118	44
C5.0	16	103	73
C4.5	24	111	57
ADTree	22	106	64
REPTree	10	104	78
Random Tree	4	68	120
Random Forest	18	113	69
Naïve Bayes	23	100	69

increases monotonically with increasing δ, except when it prunes too many important features. Furthermore, by taking some even smaller interval from the specified range, it was observed that *ROC-SVM* delivers very high accuracy. Therefore, it can be expected that by picking any value from this range for δ will provide good classification accuracy.

Sensitivity Analysis: By noting the classification accuracy of classifiers trained by using noisy data of different noise levels, we can tell how robust a classifier is. To simulate the effect of noise, we replace the feature values of all training instances as follows:

$$f' = f \times (1 + r \times \lambda), \tag{5.35}$$

where f and f' are the original and new attribute values, r is a random value in the range $[-1, 1]$ and λ is the percentage of noise. We leave the testing data intact. Using these training datasets with noise, we build the standard SVM and our *ROC-SVM* classifiers and compare their performance.

The comparison of a few selected classifiers on four datasets is shown in Fig. 4. (We have performed similar experiments on all considered datasets and observed similar behaviors.) The graphs in Fig. 4 show that the classification accuracy of the *ROC-SVM* drops less than those of the standard SVMs (for all the kernels) in the presence of noise. For example, on the GE3 dataset (see Fig. 4(b)), the classification accuracy of the *ROC-SVM* with polynomial kernel decreases from 100% to only 69.82% when the noise level increases from 0% to 40%, while that of the SVM with polynomial kernel decreases from 100% to 54.14%. The standard deviation of the performance degradation measures the degree of degradation (effect of noise) on the classifiers. A smaller deviation indicates that the classifier is more tolerant to the noise. For example, on the GE3 dataset, the standard deviation of the performance degradation of *ROC-SVM* with polynomial kernel is 12.41 as opposed to SVM with polynomial kernel, which is 19.49. This indicates that our *ROC-SVM* is can be more tolerant to noise than a standard SVM classifier.

Execution Time: With an increasing δ, the execution time for the *ROC-SVM* becomes less than that of the standard SVM. E.g., for GE1 dataset, the *ROC-SVM* ($\delta = 0.5$) takes about 3.39 seconds to build a classification model using all data, while the standard SVM takes about 5.16 seconds. However, without feature pruning the feature weighted SVM takes about a fraction of second more (5.78 seconds) than that of the standard SVM to run.

6 Conclusion

This paper has investigated a new technique for improving the classification accuracy of support vector machines. It is based on applying feature weights in the kernel function, based on consideration of the receiver operating characteristics (ROC) for each feature. The technique, known as *ROC-SVM*, is simple to describe and easy to incorporate within existing SVM implementations. It has negligible effect on running time.

Most importantly, this technique can significantly improve classification accuracy. Experimental analysis, using three metrics (i.e., accuracy, AUC and t-test), confirms that *ROC-SVM* can substantially improve over standard SVMs, particularly for the challenging scenario of gene expression data. Furthermore, it is able to outperform a range of other state-of-the-art classifiers, in a range of circumstances. Comparisons against other forms of feature weighting, such as information gain, mutual information, Fisher's criterion and chi-squared, also show that the use of ROC weighting is able to deliver better accuracy.

Overall, we strongly believe that *ROC-SVM* is a highly promising enhancement to existing SVM technology.

References

[1] Integrated Tumor Transcriptome Array and Clinical Data Analysis. Online, 2006. http://bioinfo-out.curie.fr/ittaca.

[2] A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.

[3] J. Brank and N. Milic-Frayling. A Framework for Characterizing Feature Weighting and Selection Methods in Text Classification. Technical report MSR-TR-2005-09, Microsoft Research, 2005.

[4] Broad Institute Cancer Program. Online, 1999. http://www.broad.mit.edu/cgi-bin/cancer /publications/pub_paper.cgi?mode=view&paper_id=43.

[5] Broad Institute Cancer Program. Online, 2002. http://www.broad.mit.edu/cgi-bin/cancer /publications/pub_paper.cgi?mode=view&paper_id=75.

[6] S. Cheng and F. Y. Shih. An improved incremental training algorithm for support vector machines using active query. *Pattern Recognition*, 40:964–971, 2007.

[7] N. Cristianini and J. Shawe-Taylor. *Support Vector Machines and other kernel-based learning methods.* Cambridge University Press, Trumpington Street, Cambridge, United Kingdom, 2000.

[8] L. Deng, J. Pei, J. Ma, and D. L. Lee. A Rank Sum Test Method for Informative Gene Discovery. In *Proc. of SIGKDD 04*, pages 410–419, 2004.

[9] T. Fawcett. ROC Graphs: Notes and Practical Considerations for Researchers. Technical Report MS 1143 – Extended version of HPL-2003-4, HP Laboratories, 2004.

[10] C. Ferri, P. Flach, and J. Hernández-Orallo. Learning decision trees using the area under the ROC curve. In *Proc. of ICML 02*, pages 139–146, 2002.

[11] C. Furlanello, M. Serani, S. Merler, and G. Jurman. An accelerated procedure for recursive feature ranking on microarray data. *Neural Networks*, 16:641–648, 2003.

[12] S. A. Glantz. *Primer of BioStatistics.* McGraw-Hill, NY, USA, 1992. 309–310.

[13] T. R. Golub, D. K. Slonim, P. Tamayo *et al.* Molecular Classification of Cancer: Class Discovery and Class prediction by Gene Expression Monitoring. *Science*, 286(5439):531–537, 1999.

[14] G. J. Gordon, R. V. Jensen, Li-Li Hsiao *et al.* Translation of Microarray Data into Clinically Relevant Cancer Diagnostic Tests Using Gene Expression Ratios in Lung Cancer And Mesothelioma. *Cancer Research*, 62:4963–4967, 2002.

[15] D. M. Green and J. M. Swets. *Signal detection theory and psychophysics.* John Wiley & Sons Inc., New York, USA, 1966.

[16] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1–3):389–422, 2002.

[17] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143:29–36, 1982.

[18] M. R. Hassan, M. M. Hossain, J. Bailey, and K. Ramamohanarao. Improving k-Nearest Neighbour Classification with Distance Functions based on Receiver Operating Characteristics. In *Proc. of ECML/PKDD 08*, pages 489–504, 2008.

[19] I. Hedenfalk, D. Duggan, Y. Chen *et al.* Gene-expression profiles in hereditary breast cancer. *The New England Journal of Medicine*, 344(8):539–548, 2001.

[20] M. M. Hossain, M. R. Hassan, and J. Bailey. ROC-tree: A Novel Decision Tree Induction Algorithm Based on Receiver Operating Characteristics to Classify Gene Expression Data. In *Proc. of SDM 08*, pages 455–465, 2008.

[21] N. Howe and C. Cardie. Weighting unusual feature types. Technical report, Department of Computer Science, Cornell University, 1999.

[22] A. Kertész-Farkas and A. Kocsor. Kernal-based classification of tissues using feature weightings. *Applied Ecology and Environmental Research*, 4(2):63–71, 2006.

[23] R. Kohavi, P. Langley, and Y. Yun. The Utility of Feature Weighting in Nearest-Neighbor Algorithms. In *Proc. of ECML 97*, pages 85–92, 1997.

[24] C. Lin and S. Wang. Fuzzy support vector machines. *IEEE Transactions on Neural Networks*, 13(2):464–471, 2002.

[25] Z. Liu, D. Chen, Y. Xu, and J. Liu. Logistic support vector machines and their application to gene expression data. *International Journal of Bioinformatics Research and Applications*, 1(2):169–182, 2005.

[26] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI Repository of machine learning databases. Online Repository, 1998. http://www.ics.uci.edu/~mlearn/MLRepository.html.

[27] H. Ney and R. Gierloff. Speaker recognition using a feature weighting technique. In *Proc. of ICASSP '82*, pages 1645–1648, 1982.

[28] M. Nishikawa, Y. Takakura, F. Yamashita, and M. Hashida. Basic pharmacokinetics of oligonucleotides and genes. In R. I. Mahato and S. W. Kim, editors, *Pharmaceutical Perspectives of Nucleic Acid-Based Therapeutics*, chapter 19, pages 409–433. CRC Press, London, UK, 2002.

[29] M.-D. Shieh and C.-C. Yang. Multiclass SVM-RFE for product form feature selection. *Expert Systems with Applications*, 35:531–541, 2008.

[30] D. Singh, P. G. Febbo, K. Ross *et al.* Gene Expression Correlates of Clinical Prostate Cancer Behavior. *Cancer Cell*, 1:203–209, 2002.

[31] W.-H. Steeb. *The Nonlinear Workbook.* World Scientific, USA, 3rd edition, 2005.

[32] The Division of Thoracic Surgery. Online, 2002. http://www.chestsurg.org/publications/2002-microarray.aspx.

[33] L. J. van 't Veer, H. Dai, M. J. van de Vijver *et al.* Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415:530–535, 2002.

[34] D. P. Vivencio, E. R. Hruschka Jr., M. do Carmo Nicoletti, E. B. dos Santos, and S. D. C. O. Galvão. Feature-weighted k-Nearest Neighbor Classifier. In *Proc. of FOCI 07*, pages 481–486, 2007.

[35] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature Selection for SVMs. *Advances in Neural Information Processing Systems*, 13:668–674, 2001.

[36] W. Ziqiang and F. Boqin. Collaborative Filtering Algorithm Based on Mutual Information. In *Proc. of APWeb 2004, LNCS 3007*, pages 405–415, 2004.