

Optimizing Area Under Roc Curve with SVMs

Alain Rakotomamonjy¹

Abstract. For many years now, there is a growing interest around ROC curve for characterizing machine learning performances. This is particularly due to the fact that in real-world problems misclassification costs are not known and thus, ROC curve and related metrics such as the Area Under ROC curve (AUC) can be a more meaningful performance measures. In this paper, we propose a quadratic programming based algorithm for AUC maximization and show that under certain conditions 2-norm soft margin Support Vector Machines can also maximize AUC. We present experiments that compare SVMs performances to those of other AUC maximization based algorithms and provide empirical analysis of SVMs behavior with regards to ROC- based metrics. Our main conclusion is that SVMs can maximize both AUC and accuracy compared to other algorithms like RankBoost that optimize only AUC².

1 Introduction

At the present time, some of the most important tasks in data-mining are classification, regression and information retrieval. Each of this domain proposes methods and algorithms that provide support for decision making and thus assessing their performances is a crucial problem. Usually, these algorithms come along with several parameters that can considerably modify their behavior and performances making the importance of appropriately selecting these parameters easily understandable.

Traditionally, evaluation of a learned model is done by minimizing an estimation of a generalization error or some other related measures [20]. However, accuracy (the rate of correct classification) of a classifier, which is the most frequently used performance measure, is not necessarily a good one. In fact, when the data are strongly unbalanced, accuracy may be misleading since the *all-positive* or *all-negative* classifier may achieve a very good classification rate. And situations for which data sets are unbalanced arise frequently in real-world problems and in these cases, model evaluation is done by means of other criteria than accuracy [23, 14]. Metrics extracted from ROC curve can be a good alternative for model evaluation, since they can make the difference between errors on positive or negative examples [12]. Besides, ROC curve can provide a set of hypothesis that are optimal according to some misclassification cost distributions and this can be interest since error costs are not necessarily known during learning. Hence, providing the final user a set of classifier is preferable, because he will be able choose the appropriate classifier according to his knowledge of costs.

However, If the goal is to achieve the best performance under a ROC based metrics, several works have shown that it is better to

use a specific induction principle [21, 2]. In fact, optimizing classification accuracy or the mean-square error of a classifier does not necessarily imply good ROC curve performance. Hence, several algorithms have recently been developed for optimizing the ROC curve [5, 7, 4, 23, 13] and they have been proven to work well with some different degrees of success, whereas some other works [10, 15] have shown that those metrics can be useful for model selection.

Support Vector Machines are now well-founded and largely used machine learning algorithms [18, 20]. They have been proved to be very effective on several real-world problems. Genuine SVMs implementation supposes that misclassification costs are equal for both classes, and thus SVMs are not suitable for problems which violated this hypothesis. Lin et al. [11] provided some simple extension of SVMs for non-standard situations in which error costs are not equal. However, these approaches assume that misclassification costs are known during training since they are based on different penalizations of positive and negative examples. Thus, if the data are strongly unbalanced and/or misclassification cost unknown, a ROC curve approach is still of interest.

Our aim in this paper is to propose an algorithm that optimizes an approximation of the Area Under ROC curve. We will show that the optimization problem that we achieve is very similar to the SVM's one and that there is connection between our so-called ROC optimizer SVM and classical SVMs. From this, one can deduce that in certain conditions, SVMs actually optimize the Area under the ROC curve.

This paper is organized as follows : Section 2 gives a brief background on ROC curve. Section 3 formally defines the problem of optimizing AUC with SVM. After having derived the algorithm, a short analysis is provided which suggests that 2-norm SVMs maximize the area under the roc curve in some feature space. In section 4, we provide some empirical analysis of these SVMs algorithms and we experimentally compare SVMs to other algorithms, such as RankBoost, with respects to the AUC measure. Finally, section 5 gives a conclusion and proposes some perspectives of this work.

2 ROC curve

The *Receiver Operating Characteristics* curve has been introduced by the signal processing community in order to evaluate the capability of an human operator to distinguish informative radar signal from noise. Then, it has mostly been used in the medical decision making community for assessing the usefulness of a diagnostic test.

ROC curve is a two-dimensional measure of classification performance. It can be understood as a plot of the probability of correctly classifying the positive examples against the rate of incorrectly classifying true negative examples. In this sense, one can interpret this curve as a comparison of the classifier performance across the entire range of class distributions and error costs. Usually, decision rule

¹ P.S.I CNRS FRE 2645, INSA de Rouen, France email: alain.rakotomamonjy@insa-rouen.fr

² Parts of this paper have already been submitted to the Modelling, Computation and Optimization Conference, Metz 2004

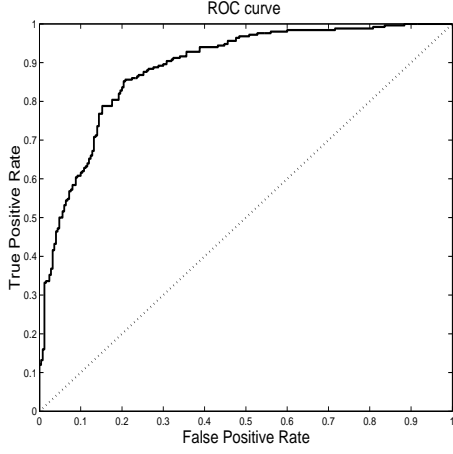


Figure 1. Example of ROC curve. The diagonal line denotes the ROC curve of a random classifier.

is performed by selecting a decision threshold which separates the positive and negative classes. Thus, when dealing with minimum error classifier, most of the time this threshold is set in order to approximate the Bayes error rate. However, class distributions or error costs can be so that the optimal threshold associated to the Bayes risk varies within a large range of values, and for each possible value of this threshold a pair of true-positive and false-positive performance rate is thus obtained. Hence, ROC curve can be completely determined by varying this threshold value. Thus, one of the most interesting point of ROC curve is that if error costs or class distributions are unknown, classifier performance can still be characterized and optimized. Figure (1) depicts an example of the ROC curve of a given classifier. The diagonal line corresponds to the ROC curve of a classifier that predicts the class at random and the performance improves the further the curve is near to the upper left corner of the plot.

The most frequently used performance measure extracted from the ROC curve is the value of the area under the curve, commonly denoted as AUC. When AUC is equal to 1, the classifier achieves perfect accuracy if the threshold is correctly chosen, and a classifier that predicts the class at random has an associated AUC of 0.5. Another interesting point of the AUC is that it depicts a general behavior of the classifier since it is independent to the threshold used for obtaining a class label. Processing the AUC would need the computation of an integral in the continuous case, however, in the discrete case, one can compute this area with step functions and the following property holds :

$$AUC = \frac{\sum_{i=1}^{n^+} \sum_{j=1}^{n^-} 1_{f(x_i^+) > f(x_j^-)}}{mn} \quad (1)$$

where $f(\cdot)$ is denoted as the scoring function (in machine learning, the decision function of a classifier is the most frequent scoring function), x^+ and x^- respectively denote the positive and negative samples and n^+ and n^- are respectively the number of positive and negative examples and 1_π is defined to be 1 if the predicate π holds and 0 otherwise. The above equation has also been pointed out as being the Wilcoxon-Mann-Whitney statistic. This equation states that if a classifier $f(x)$ is so that $f(x_i^+) > f(x_j^-), \forall i = 1, \dots, n^+, \forall j = 1, \dots, n^-$ then the AUC of this classifier is maximal. Any negative sample that happens to be ranked higher than positive sample makes the AUC decreases.

Table 1. ROC curve related metrics. tp and fp are respectively the true and false positive rate and c the skewness ratio

Metrics	Equation
Accuracy	$\frac{tp+c(1-fp)}{1+c}$
Precision	$\frac{tp}{tp+c \cdot fp}$
F-measure	$\frac{2tp}{tp+c \cdot fp+1}$
WRAcc	$\frac{4c(tp-fp)}{(1+c)^2}$

Other frequently used measures in the machine learning community are related to the ROC curve through the confusion matrix of a classifier. These metrics depend on the true positive (tp) and false positive (fp) rate and the skew parameter c which is the ratio of negative examples over positive examples. These different measures are reminded in Table (1) and an analysis of their properties can be found in Flach [6]. Precision and F-measure are for instance frequently used in the information retrieval community in order to assess algorithm's performances [8, 17, 16].

3 Optimizing AUC with SVMs

This section presents an algorithm for maximizing AUC with SVMs. After having derived the problem formulation, solutions and analysis of the method are proposed.

3.1 Optimization problem

Suppose we have some learning examples $\{x_i, y_i\}_{i=1}^\ell$, a linear decision function on the form $f(x) = \langle w, x \rangle + b$ for which the class of a new example x is given by $\text{sign}(f(x))$. the AUC equation (1) corresponding to this learning set and $f(x)$ is equivalent to :

$$AUC = \frac{\sum_{i=1}^{n^+} \sum_{j=1}^{n^-} 1_{\xi_{ij} > 0}}{n^+n^-} \quad \text{with} \quad \xi_{ij} = f(x_i^+) - f(x_j^-)$$

Hence the problem of maximizing AUC becomes :

$$\begin{aligned} \max_w \quad & \frac{1}{n^+n^-} \sum_{i=1}^{n^+} \sum_{j=1}^{n^-} 1_{\xi_{ij} > 0} \\ \text{with} \quad & \xi_{ij} = f(x_i^+) - f(x_j^-) \quad 1 \leq i \leq n^+, 1 \leq j \leq n^- \end{aligned} \quad (2)$$

This equation which can be rewritten as a minimization problem poses several issues. First of all, the objective function is not differentiable over the range of ξ_{ij} . Thus in order to make it tractable, an approximated differentiable objective is needed. Another issue is that the problem is ill-posed since solution of the problem may not be unique. In fact, if the learning examples are separable, then any linear hypothesis of the version space would be a maximizer of this cost function. Thus, in order to make it well-posed, the problem needs to be stabilized for instance by regularization. Taking into account all these points, looking for linear hypothesis that maximizes AUC is approximately equivalent to solve the problem :

$$\begin{aligned} \min_w \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n^+} \sum_{j=1}^{n^-} \xi_{ij} \\ \text{with} \quad & f(x_i^+) \geq f(x_j^-) + \rho - \xi_{ij} \quad 1 \leq i \leq n^+, 1 \leq j \leq n^- \\ & \xi_{ij} \geq 0 \quad 1 \leq i \leq n^+, 1 \leq j \leq n^- \end{aligned} \quad (3)$$

where $\rho > 0$. A square-norm regularization term has been added to the objective function. It aims at stabilizing the problem through

a Tikhonov based-regularization. The choice of this regularization component is arbitrary but the advantage of this one being its similarity with SVMs and its computational tractability since the problem is convex. Like in SVMs, C is a parameter that allows to trade-off between the regularization term and the constraint violation $\xi_{i,j}$. The step function of $\xi_{i,j}$ in equation (2) has been replaced by a linear function of $\xi_{i,j}$. This again makes the problem more tractable at the expense of roughly approximating the true objective function. Different approximation functions could have been used, for instance, sigmoid functions are frequently used as an approximation of step function. However, using these functions turn the optimization problem into a non-linear one which is hard to solve. Yan et al. [23] have used such a function for the similar problem of optimizing AUC.

The constrained optimization problem defined in equation (3) can be solved by a standard method of Lagrange multipliers. The Lagrangian function associated to the above primal problem is :

$$\begin{aligned} \mathcal{L}(w, \xi_{i,j}) &= \frac{1}{2} \|w\|^2 + C \sum_{i,j=1}^{n^+, n^-} \xi_{i,j} \\ &\quad - \sum_{i,j=1}^{n^+, n^-} \alpha_{i,j} (\langle w, x_i^+ - x_j^- \rangle - \rho + \xi_{i,j}) \\ &\quad - \sum_{i,j=1}^{n^+, n^-} \gamma_{i,j} \xi_{i,j} \end{aligned}$$

Getting Lagrangian derivatives with regards to primal variables give :

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w} &= w - \sum_{i,j} \alpha_{i,j} (x_i^+ - x_j^-) \quad \frac{\partial \mathcal{L}}{\partial \xi_{u,v}} = C - \alpha_{u,v} - \gamma_{u,v} \\ \forall u \in 1, \dots, n^+, v \in 1, \dots, n^- \end{aligned} \quad (4)$$

After making these derivatives vanished, replugging these equations back into the Lagrangian leads to the following dual optimization problem :

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2} \sum_{i,j=1}^{n^+, n^-} \sum_{u,v=1}^{n^+, n^-} \alpha_{i,j} \alpha_{u,v} \langle x_i^+ - x_j^-, x_u^+ - x_v^- \rangle \\ & + \rho \sum_{i,j=1}^{n^+, n^-} \alpha_{i,j} \\ \text{with} \quad & C \geq \alpha_{i,j} \geq 0 \end{aligned} \quad (5)$$

This is a quadratic optimization problem that can be solved using classical algorithms like interior point or active constraints methods. Similarly to classical SVMs, expression of w shows that the decision function depends only on data points. In this particular case, it depends on difference of negative and positive examples :

$$f(x) = \sum_{i,j=1}^{n^+, n^-} \alpha_{i,j}^* \langle x_i^+ - x_j^-, x \rangle + b \quad (6)$$

where $\alpha_{i,j}^*$ are the arguments that maximize the dual problem (5).

3.2 Analyzing the KKT conditions

The Lagrange multiplier values $\alpha_{i,j}$ usually denote the difficulties of satisfying the constraints of a constrained optimization problem. Thus analyzing these variables lead to a better insight of the problem. In our case, we can get the following proposition :

Proposition 3.2.1 *For all pairs of positive and negative examples (i, j) optimized with regards to AUC through the primal and dual*

problem given in equation (3) and (5), the ranking constraints are related to the Lagrange multiplier values by the equation :

$$\begin{cases} f(x_i^+) \geq f(x_j^-) + \rho & \text{if } \alpha_{i,j} = 0 \\ f(x_i^+) \leq f(x_j^-) + \rho & \text{if } \alpha_{i,j} = C \\ f(x_i^+) = f(x_j^-) + \rho & \text{if } 0 < \alpha_{i,j} < C \end{cases}$$

Proof : This proposition can be easily proved by looking at the KKT conditions of the primal optimization problem. At optimality of the problem (3), the following equations hold :

$$\alpha_{i,j} (f(x_i^+) - f(x_j^-) - \rho + \xi_{i,j}) = 0 \quad \forall (i, j) \quad (7)$$

$$\gamma_{i,j} \xi_{i,j} = 0 \quad \forall (i, j) \quad (8)$$

$$C - \alpha_{i,j} - \gamma_{i,j} = 0 \quad \forall (i, j) \quad (9)$$

$$\alpha_{i,j} \geq 0 \quad \forall (i, j) \quad (10)$$

$$\gamma_{i,j} \geq 0 \quad \forall (i, j) \quad (11)$$

then if $\alpha_{i,j} = 0$, we have $\gamma_{i,j} \neq 0$, thus $\xi_{i,j} = 0$ and finally, by plugging this into the primal problem constraints, we have $f(x_i^+) \geq f(x_j^-) + \rho$. If $\alpha_{i,j} = C$, then $\gamma_{i,j} = 0$ thus $\xi_{i,j} \geq 0$ and finally $f(x_i^+) < f(x_j^-) + \rho$. At last, if we have $0 < \alpha_{i,j} < C$, then $f(x_i^+) - f(x_j^-) - \rho + \xi_{i,j} = 0$, but since $\xi_{i,j} = 0$ also holds, we get $f(x_i^+) = f(x_j^-) + \rho$ ■

Hence, similarly to SVMs, the decision function $f(x)$ that maximizes AUC depends only on some examples of the problem. According to the above proposition, the support vector examples are those that strictly meet the ranking constraint and those that are badly ranked. This is in accordance with the intuition that since the area under ROC curve depends essentially on examples that are badly ranked, one should particularly focus on those examples for optimizing the AUC.

3.3 Speed-up tricks

The number of variables in the dual problem is equal to $n^+ \cdot n^-$, thus even for a small-scale problem, using this algorithm can rapidly be prohibitive both in time and memory complexities. For reducing the size of the quadratic programming problem, a simple heuristic based on the local neighbors of each example can be used. This idea has already been exploited in [9].

Instead of ranking all the positive examples higher than all negative examples, we only focus on a subset of a positive-negative examples. Suppose m being a user-defined constant specifying the number of interesting neighbors of a sample, our trick is the following : at first let \mathcal{N}^+ being the set of all positive examples that are in the m -nearest positive neighbors of each negative samples. In the worst-case scenario, the cardinality of \mathcal{N}^+ is $m \cdot n^-$. Then, we consider for AUC optimization the pairs of the \mathcal{N}^+ samples and their m -nearest negative neighbors. For summarizing, we maximize AUC for the following examples :

$$i \in \mathcal{N}^+, j \in B(i, m)$$

where $B(i, m)$ is the set of m -nearest negative neighbors of the positive sample i . Hence the number of variables in the QP problem is reduced to at most $m^2 \cdot n^-$ samples.

Although this is the trick that has been implemented for the experiments, it is likely that some more efficient heuristics can be used. For instance, it would be interesting to identify examples that are likely to be incorrectly ranked and then to perform AUC optimization only on these examples.

3.4 Links with SVM

Owing to the square-norm regularization of w , the primal problem in equation (3) seems to be very similar to the classical SVMs optimization problem. Our aim in this paragraph is to show that although the feasibility domain of these two problems are rather different, in some cases the resulting decision function can be identical.

Proposition 3.4.1 *Suppose that a training set $\{x_i, y_i\}_{i=1}^\ell$ is linearly separable. Then for any $\rho > 0$, the solution of the ROC-optimizer SVM w_{ROC} is related to the solution w_{SVM} of a classical SVM problem through a constant β by the equation :*

$$w_{ROC} = \beta w_{SVM}$$

Proof : First of all, since the data are separable, the convex hull of positive and negative examples (the envelope of the examples) are disjoint. Hence, the primal problem of ROC-SVM can be stated as :

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} \|w\|^2 \\ \text{with} \quad & f(x_i^+) \geq f(x_j^-) + \rho \quad 1 \leq i \leq n^+, 1 \leq j \leq n^- \end{aligned}$$

Recall that the constraints of the classical SVMs problem are, in the linearly separable case :

$$\begin{cases} f(x_i^+) \geq 1 & \forall i \in 1, \dots, n^+ \\ f(x_j^-) \leq -1 & \forall j \in 1, \dots, n^- \end{cases}$$

then for any b and w_{SVM} that satisfy these constraints, we have

$$\begin{aligned} \langle w_{SVM}, x_i^+ \rangle - \langle w_{SVM}, x_j^- \rangle & \geq 2 \\ \langle \frac{\rho}{2} w_{SVM}, x_i^+ \rangle - \langle \frac{\rho}{2} w_{SVM}, x_j^- \rangle & \geq \rho \end{aligned}$$

then up to a multiplicative constant $\beta = \frac{\rho}{2}$, any hyperplane in the feasibility domain of the classical SVMs problem is related to an unique hyperplane in the ROC optimizer SVMs one.

Now suppose that w_{ROC} is so that

$$\langle w_{ROC}, x_i^+ \rangle - \langle w_{ROC}, x_j^- \rangle \geq \rho \quad \forall i = 1, \dots, n^+ j = 1, \dots, n^-$$

Let us define the following quantities

$$\rho_+ = \min_{i=1, \dots, n^+} f(x_i^+) \quad \rho_- = \max_{j=1, \dots, n^-} f(x_j^-)$$

then by definition we have $\rho_+ - \rho_- \geq \rho$, $\langle w_{ROC}, x_i^+ \rangle \geq \rho_+$ and $\langle w_{ROC}, x_j^- \rangle \leq \rho_-$. Thus, we have :

$$\begin{cases} \frac{2}{\rho_+ - \rho_-} \langle w_{ROC}, x_i^+ \rangle - \frac{\rho_+ + \rho_-}{\rho_+ - \rho_-} \geq 1 & \forall i \in 1, \dots, n^+ \\ \frac{2}{\rho_+ - \rho_-} \langle w_{ROC}, x_j^- \rangle - \frac{\rho_+ + \rho_-}{\rho_+ - \rho_-} \leq -1 & \forall j \in 1, \dots, n^- \end{cases}$$

and then there exists a b and $w_{SVM} = \frac{2}{\rho_+ - \rho_-} w_{ROC}$ that belongs to the feasibility domain of SVMs optimisation problem.

Thus, since there exists a one-to-one relation between the two feasibility domain and since their objective function are thus equal up to a multiplicative constant, one can conclude that in this case, the solution w_{ROC} of the ROC-optimizer SVM is of the form $w_{ROC} = \beta w_{SVM}$. ■

Interestingly, this proposition tell us that SVMs maximize an approximation of AUC particularly if the data sets are linearly separable. However, in the non-separable case, the connection between ROC-SVMs and SVMs is not so straightforward. We believe that since the support vectors sets involved in both problems can be very different it is not likely that there is some relationships between solutions.

3.5 Generalization to non-linear case

Since the decision function $f(x)$ given in equation (6) and the dual problem in equation (5) that determines the optimal $\alpha_{i,j}$ depend only on the inner product between dataset, the so-called kernel trick [18] can be used for extending the ROC optimizer SVMs to non-linear problem. Hence, as for SVMs, one can use any kernel $k(x, x')$ satisfying the Mercer's conditions for maximizing AUC in some feature space \mathcal{F} associated to the $k(x, x')$. In this case, the solution of the problem is obtained from the general equations :

$$f(x) = \sum_{i=1}^{n^+} \sum_{j=1}^{n^-} \alpha_{i,j}^* (k(x_i^+, x) - k(x_j^-, x)) + b$$

where $\alpha_{i,j}^*$ are the arguments that optimize the following quadratic programming problem :

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2} \sum_{i,j=1}^{n^+, n^-} \sum_{u,v=1}^{n^+, n^-} \alpha_{i,j} \alpha_{u,v} [k(x_i^+, x_u^+) - k(x_i^+, x_v^-) \\ & - k(x_u^+, x_j^-) + k(x_j^-, x_v^-)] + \rho \sum_{i,j=1}^{n^+, n^-} \alpha_{i,j} \\ \text{with} \quad & C \geq \alpha_{i,j} \geq 0 \end{aligned} \quad (12)$$

3.6 2-norm SVMs and maximization of AUC

In a previous paragraph, we have shown that there is a strong connection between SVMs and ROC optimizer SVMs for separable data sets.

In this section, we want to show that when dealing with non-separable data sets, SVMs with quadratic penalization of errors (2-norm SVMs) produce a linear hypothesis that maximizes the AUC in some feature space.

When the data sets are not separable, the SVMs large margin optimization problem implement the following idea. At first, the data points are mapped to a feature space \mathcal{H} through a non-linear transformation $\Phi(x)$ which is implicitly defined by the kernel $k(x, x')$ of \mathcal{H} . Then the decision function obtained by SVMs is :

$$f(x) = \langle w, \Phi(x) \rangle_{\mathcal{H}} + b = \sum_{i=1}^{\ell} \alpha_i^* y_i k(x, x_i) + b \quad (13)$$

with $f(x)$ being the optimal hyperplane that optimizes the following problem :

$$\begin{aligned} \min_{w,b,\xi_i} \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{\ell} \xi_i^2 \\ \text{st} \quad & y_i f(x_i) \geq 1 - \xi_i \end{aligned} \quad (14)$$

in which the terms ξ_i are relaxing constraint variables that are quadratically penalized, and C a penalization term which makes a compromise between the margin and the amount of errors. The optimal α_i^* are then obtained from the dual of the previous problem :

$$\begin{aligned} \max_{\alpha} \quad & -\sum_{i,j}^{\ell} \alpha_i \alpha_j y_i y_j (k(x_i, x_j) + \frac{1}{C} \delta_{x_i}(x_j)) + \sum_i \alpha_i \\ \text{st} \quad & \sum_i \alpha_i y_i = 0, \quad \alpha_i \geq 0 \end{aligned} \quad (15)$$

This problem is equivalent to the dual of a large margin problem in a feature space \mathcal{H}' in which the data are linearly separable. Thus one can state the following proposition :

Proposition 3.6.1 *Given a training set $\{x_i, y_i\}_{i=1}^\ell$, SVMs with quadratic penalization maximize the area under the ROC curve for this data set in some feature space \mathcal{H}' which inner product is :*

$$k'(x, x') = k(x, x') + \frac{1}{C} \delta_x(x')$$

Proof : Since the data sets are linearly separable in \mathcal{H}' then all linear hypothesis of the version space correctly rank the positive and negative examples of the training set. Hence, AUC is maximal with value 1. ■

Usually in the 2-norm SVMs case, the optimal α obtained from the dual optimization problem are then used for building a decision function which is $f(x) = \sum_{i=1}^{\ell} \alpha_i^* y_i k(x, x_i) + b^*$. Thus, even if the optimization problem (15) can be considered as a hard-margin problem in a space \mathcal{H}' , applying the decision function (13) to the training examples can lead to misclassification since it is performed in \mathcal{H} . Hence for preserving the AUC-maximal property of this decision function on the training set, one should consider evaluating it in \mathcal{H}' and not \mathcal{H} and consequently the decision function should be :

$$f(x) = \sum_{i=1}^{\ell} \alpha_i^* y_i \left(k(x, x_i) + \frac{1}{C} \delta_{x_i}(x) \right) + b \quad (16)$$

However, assuming that the training set and the set of points on which $f(x)$ is evaluated are disjoint, the above decision function and the 2-norm SVMs one given in equation (13) are equivalent.

Let's try to get some insights on the kernel $k'(x, x')$ and its role in the AUC maximization. Suppose that $\mathcal{X} \in \mathbb{R}^d$, the positive definite kernel on \mathcal{X} defined as : $\forall x, x' \in \mathcal{X}, k_2(x, x') = \frac{1}{C} \delta_x(x') = \frac{1}{C} 1_{x=x'}$ is the kernel of a reproducing kernel Hilbert space which functions are null except on a countable set [1]. Thus \mathcal{H}' is a set of functions that have the same values of function in \mathcal{H} except that, on a countable set of points, $\frac{1}{C}$ is added. The influence of C is the following :

- When C goes to infinity, $k_2(x, x')$ vanishes in the Gram matrix of the dual problem (equation (15)). Thus the resulting decision function converges towards the usual 2-norm SVMs decision function (equation (13)) and it still maximizes the area under ROC curve.
- when C goes to zero, the influence of $k_2(x, x')$ becomes more and more important compared to $k(x, x')$, and thus the Gram matrix tends to be diagonal. In this case, each example can be seen as being orthogonal to any other, and the resulting decision function will overfit the training data [19] although it still maximizes the AUC.

Thus, any feature space \mathcal{H}' and any C can provide a linear hypothesis that is AUC maximal with regards to the training set. However, we have just noted that some feature space for maximizing AUC can lead to overfitting when C is not chosen appropriately. In fact, one conclusion that can be drawn is that C should not be set to a too small a value. A rationale for that is because the space \mathcal{H}' in which AUC has been maximized will be too different to the space \mathcal{H} in which the decision function will be tested (provided that the training set and test set are disjoint), thus leading to poor generalization capability.

4 Numerical experiments

In this section, our aim is to provide some empirical analysis of our ROC optimizer algorithm behaviour and to analyse AUC generalization capability of the 2-norm SVMs.

4.1 Toy problem

The artificial toy problem used throughout this section (4.1) is the one that has been introduced by Weston et al. [22] in their paper on

feature selection. This ten-dimensional non linear problem has been built by drawing data from four 2-dimensional normal distributions of equal covariance matrix but different means. The rest of the features are independent gaussian noise with zero mean and unit variance. And each variable is then normalized to zero mean and unit variance according to the training set scaling parameters.

In this first experiment, we have analysed the performance of the ROC- optimizer SVMs with regards to its parameters ρ and the size of the opposite class neighborhood m as specified in previous paragraph.

Training set is composed of 200 data points in which positive and negative classes are equally represented. Thus using the full data sets, the original problem given in equation (3) would have 10000 constraints, hence it is necessary to reduce the problem size by limiting the number of ranking comparisons between positive and negative examples. Our experiments follow the procedure below.

We have used a gaussian kernel $k(x, y) = \exp -\frac{\|x-y\|^2}{2\sigma^2}$ for which σ have been set to 3. The penalization term C is equal to 1000. These parameters have been chosen owing to our previous experiments and knowledge about this data set. Since we are mainly interested in how the algorithm behaves with regards to other parameters, it is natural to fix C and σ to some nearly optimal values. We have tested several pairs of ρ and m which respectively are 0.01, 0.1, 1 and 1, 5, 10, 50, 100. And for each couple, we have averaged the performance results on 20 different trials of the data sets.

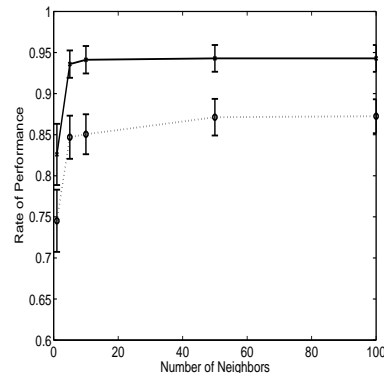


Figure 2. Results of ROC optimizer SVM on toy problem. (solid) AUC vs number of neighbors. (dotted) Accuracy vs number of neighbors.

Figure (2) depicts rate of performance on AUC and accuracy of ROC-optimizer SVMs with regards to the size of the neighborhood. It shows that AUC and accuracy both increase with m increases and then they rapidly reach a “steady” value. For AUC, this steady value is reached for $m = 10$ whereas for accuracy, a neighborhood of size 50 has to be considered before yielding to this maximum. These results can be explained by the fact that since ROC curve and AUC are somewhat related to the distributions of classes and their overlapping, optimizing all example ranking is not useful but one should rather focus on examples which correspond to distribution overlapping. This is clearly depicted by Figure (2) plots. When m is small, all examples that correspond to overlapping are not taken into account in the AUC optimization, thus leading to poor performance. As m increases, more and more critical examples are included in the optimization problem, thus leading to an increase of AUC. At last, when m is such that all critical examples are optimized with regards

to their rankings, adding other “easy” examples for which ranking are already correct do not improve AUC further.

Concerning margin effects on performances, this experiment has also shown that ρ has no effect on both AUC and accuracy. For AUC, this is trivial since the important point is to rank positive examples higher than negatives ones no matter the difference of ranking score.

To summarize, this experiment allow us to draw the conclusion that choosing a good parameter m is crucial for achieving good results in a reasonable time, and that the best value of m is problem-dependent since it is related to class distributions.

Another experiment has been driven in order to get some insights of how the kernel parameters and C parameter influence AUC performances for both ROC optimizer SVMs and 2-norm SVMs. For this latter algorithms, in other words, we are investigating on the influence of the feature space \mathcal{H}' on AUC maximization.

We have generated 100 negative examples and added different amount of positive examples which depends on a predefined skewness of the data sets. These relative frequencies of positive examples have been chosen to be 0.1, 0.3 and 0.5. Again, we have used a gaussian kernel. In this experiment, for ROC-SVMs, the margin ρ and the size of neighborhood m have been respectively fixed to 0.01 and 10. Then, for both methods, one parameter between C and σ has been kept fixed whereas the other is varying within a given range.

Results have been evaluated from 1000 samples (in which positive and negative examples are equally distributed) and are obtained from the averaging of 20 trials. For a sanity check, we have verified that for 2-norm SVMs, all training examples are correctly classified in \mathcal{H}' and that the resulting AUC is thus equal to 1.

Figures (4)and (3) depict the plots of the resulting AUC and accuracy with respect to C or σ for ROC-SVMs and 2-norms SVMs. The different curves in each figure are function of the data skewness. Figures show that regardless of algorithms, how the data are skewed, accuracy and AUC are somewhat related on average since the shapes of the curves are similar. This is in accordance with Cortes et al. [3] findings.

As expected for 2-norms SVMs, low values of C leads to overfitting. However, it seems that if C and σ parameters are not chosen properly, the resulting decision function will overfit the training data. In fact, both accuracy and AUC curves show large ranges of parameters in which performances are particularly poor, and this, independently to the considered parameter. For ROC-SVM, parameters tuning seems to be less critical since except for accuracy performance versus σ , there exists a large range of parameter values for which performances are rather equal. For instance, only very small values of C leads to poor performances.

Another interesting point is the error bars obtained for AUC curves. In fact, they corroborate the claim of Cortes et al. [3] : “the more the class distribution is skewed, the more the AUC variance will be large”. This is clear when one compares error bars of the top and bottom curve of each AUC figure independently to the algorithms.

When accuracy is considered, it seems that 2-norm SVMs are more effective than ROC-SVM. Evidences for this claim are : performance is less sensitive to parameters tuning and error bars are smaller. However, this is understandable since 2-norms SVMs have been designed for margin-maximization in some feature spaces and thus are implicitly designed to optimize accuracy whereas ROC-SVM are based on AUC maximization.

Hence, to sum up, this experiment has confirmed that i) accuracy and AUC are on average related but AUC variance can be very large, ii) model selection of these AUC maximization SVMs has to be carried out carefully since they seems to be prone to overfitting.

4.2 Benchmark datasets

This part of the numerical experiments is devoted to show that SVMs can, as well as other dedicated algorithms such as RankBoost, optimize the AUC. For this purpose, we have reproduced the experiments carried out by Cortes et al [3]. For comparison, we have used the following benchmark datasets, publicly available to the UCI Machine Learning repository : *Breast-wpbc*, *Credit Ionosphere*, *Pima* and *Spectf*. For data sets with missing values, incomplete examples have been discarded.

In order to measure accurately AUC value, a ten-fold cross-validation has been performed. For each of the resulting train/test splits, each numeric attribute has been normalized to zero mean and unit standard deviation on the training set. Test sets have also been rescaled according to the training set scaling parameters. For these experiments, a gaussian kernel has been used and again parameters C and σ are varying across a large range of values.

In this paragraph, we have compared AUC and accuracy performances of four different algorithms : ROC-SVM, 2-norm SVMs, RankBoost [7] and AUCsplit [5].

For ROC optimizer SVMs, several parameters have to be set, we have arbitrarily decided to set $\rho = 0.01$ and $m = 10$. Although we are convinced that fixing m to this value for all data sets is not optimal, this is a good compromise between performance and time complexity for solving optimization problem (3) (remind that the number of constraints in the problem can reach $m^2 \cdot n$).

Table (2) summarizes the results achieved by SVMs compared to those obtained with RankBoost and AUCsplit. (results of these latter algorithms are extracted respectively from Cortes et al. [3] and Ferri et al. [5] papers and thus we are not able to perform some statistical comparisons). Reported AUC results are the best results achieved by cross-validation for all couple of C and σ . Table (3) shows the related accuracy of the best model in the AUC sense.

Table 2. Best AUC results achieved on datasets for different performance measures with a gaussian kernel SVMs. For ROC optimizer SVMs, we have set $\rho = 0.01$ and $m = 10$.

SET	ROC-SVM	2-NORM SVM
BREAST-WPBC	73.77 ± 16.12	77.29 ± 15.7
CREDIT	88.57 ± 14.90	92.05 ± 11.1
IONOSPHERE	95.85 ± 3.58	98.7 ± 1.8
PIMA	83.01 ± 4.23	83.9 ± 4.1
SPECTF	77.45 ± 12.15	86.0 ± 7.9

SET	RANKBOOST	AUC SPLIT
BREAST-WPBC	80.4 ± 8.0	59.3 ± 16.2
CREDIT	94.5 ± 2.9	N.A
IONOSPHERE	98.0 ± 3.3	89.7 ± 6.7
PIMA	84.8 ± 6.5	76.7 ± 6.0
SPECTF	93.4	N.A

The first thing that can be observed is that, when tuned appropriately, the 2-norm SVMs can achieve AUC as high as those obtained by RankBoost at least for some of the datasets used in this experiments. The interesting point is that RankBoost is an algorithm which induction principle is based on AUC maximization, and thus, it is expected to perform well under this criterion, whereas the kernel trick and the large margin optimization principle can achieve same performances in AUC sense. Unreported experiments have also shown that, for the datasets in which gaussian kernel SVMs leads to lower AUC

Table 3. Best Accuracy results achieved on datasets for different performance measures with a gaussian kernel SVMs. For ROC optimizer SVMs, we have set $\rho = 0.01$ and $m = 10$.

NAME	ROC-SVM	2-NORM SVM
BREAST-WPBC	79.15 ± 9.89	77.8 ± 8.15
CREDIT	82.50 ± 12.21	85.4 ± 19.5
IONOSPHERE	90.31 ± 4.90	94.0 ± 3.9
PIMA	74.72 ± 4.59	77.4 ± 3.8
SPECTF	78.30 ± 4.01	79.6 ± 6.8

NAME	RANKBOOST	AUC SPLIT
BREAST-WPBC	65.5 ± 13.8	69.5 ± 10.6
CREDIT	81.0 ± 7.4	N.A
IONOSPHERE	83.6 ± 10.9	89.6 ± 5.0
PIMA	69.7 ± 7.6	72.5 ± 5.1
SPECTF	67.3	N.A

values than RankBoost, other kernels (such as polynomial kernel) can further increase the AUC. Hence these results allow to conclude that when model selection is performed accurately, SVMs can indeed maximize the area under the ROC curve. Besides, the accuracy table also proves that the large margin principle can achieve AUC maximization without harming accuracy. In fact, one can see that SVMs accuracy is always higher than those obtained with RankBoost and AUCSplit.

Tables (2) and (3) also show that ROC-SVMs perform worse than 2-norm SVMs both from AUC and accuracy point of view. One may argue that this is due to inappropriate choice of parameter m . Then, in order to analyze again how m influences performance, we have run the same experiments for different neighborhood size but only for two data sets. Figure (5) plots AUC and accuracy rate for these two data sets. It seems clear that $m = 10$ is not an optimal value although it maximizes AUC for the *wdbc* data sets. However setting m to 50 does not always lead to performance improvement although the algorithm time complexity becomes drastic. Hence for these cases, we may conclude that m does influence slightly the results but performance improvement seem not to worth the time spent for learning.

Hence, since 2-norm SVMs perform better and have a lower complexity than ROC-Optimizer SVMs (the number of constraints in the former is $n^+ \cdot n^-$ whereas for the latter it can reach $m^2 \cdot n^-$), our conclusion is that 2-norm SVMs are better for AUC maximization.

5 Discussions and conclusions

In this paper, we have proposed a SVMs based algorithm for Area Under Roc curve maximization. For this, we have derived an numerically tractable approximation of AUC criterion that leads to a quadratic programming problem. Then we have also shown that 2-norm SVMs maximize AUC in some feature space.

Experiments we carried out proved that ROC-SVMs and 2-norm SVMs can actually maximize AUC and that their performances are comparable to those of RankBoost, but they outperform algorithms such as AUCSplit. Although comparable in the AUC sense, 2-norms SVMs are cheaper in time complexity and also lead to good performance in accuracy and thus makes them more appealing.

Further studies on this topic can be the development of a theoretical framework of generalization bound on AUC and a more extensive comparison of SVMs to other algorithms and on a larger amount of data sets.

REFERENCES

- [1] S. Canu, X. Mary, and A. Rakotomamonjy, *Functional learning through kernels*, volume 190, chapter 5, 89–110, IOS Press, advances in learning theory: methods, models and applications, nato science series III: computer and systems sciences edn., 2003.
- [2] R. Caruana, S. Baluja, and T. Mitchell, ‘Using the future to sort out the present : Rankprop and multitask learning for medical risk evaluation’, in *Advances in Neural Information Processing Systems*, volume 8, (1996).
- [3] C. Cortes and M. Mohri, ‘AUC optimization vs error rate minimization’, in *Advances in Neural Information Processing Systems*, volume 15, (2003).
- [4] T. Fawcett, ‘Using rule sets to maximize ROC performance’, in *Proceedings of the IEEE International Conference on Data Mining*, (2001).
- [5] C. Ferri, P. Flach, and J. Hernandez-Orallo, ‘Learning decision trees using the area under the roc curve’, in *Proceedings of the 19th International Conference on Machine Learning*, (2002).
- [6] P. Flach, ‘The geometry of roc space : understanding machine learning metrics through roc isometrics’, in *Proceedings of the 20th International Conference on Machine Learning*, pp. 194–201, (2003).
- [7] Y. Freund, R. Iyer, R. Schapire, and Y. Singer, ‘An efficient boosting algorithm for combining preferences’, *Journal of Machine Learning Research*, **4**, 933–969, (Nov 2003).
- [8] T. Joachims, ‘Transductive inference for text classification using svms’, in *Proceedings of The 16th International Conference on Machine Learning*, (1999).
- [9] J. Kwok and I. Tsang, ‘Learning with idealized kernels’, in *Proceedings of the 20th International Conference on Machine Learning*, pp. 400–407, (2003).
- [10] N. Lachiche and P. Flach, ‘Improving accuracy and cost of two-class and multi-class probabilistic classifiers using roc curves’, in *Proceedings of the 20th International Conference on Machine Learning*, pp. 416–423, (2003).
- [11] Y. Lin, Y. Lee, and G. Wahba, ‘Support vector machines for classification in non standard situation’, *Machine Learning*, **46**, 191–202, (2002).
- [12] C. Ling, J. Huang, and H. Zhang, ‘Auc: a better measure than accuracy in comparing learning algorithms’, in *Proceedings of 2003 Canadian Artificial Intelligence Conference*, (2003).
- [13] C. Ling and J. Yan, ‘Decision tree with better ranking’, in *Proceedings of the 20th International Conference on Machine Learning*, (2003).
- [14] M. Maloof, ‘Learning when data sets are imbalanced and when costs are unequal and unknown’, in *ICML Workshop on Learning from Imbalanced Data Sets II*, (2003).
- [15] M. Mozer, R. Dodier, and M. Colagrosso and. Guerra-Salcedo, ‘Prodding the ROC curve’, in *Advances in Neural Information Processing Systems*, volume 14, (2002).
- [16] D. Musicant, V. Kumar, and A. Ozgur, ‘Optimizing f-measure with support vector machines’, in *Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference*, eds., I. Russell and S. Haller AAAI Press, pp. 356–360, (2003).
- [17] V. Raghavan, P. Bollmann, and G. Jung, ‘A critical investigation of recall and precision as measures of retrieval system performance’, *ACM Transactions on Information Systems*, **7**(3), 205–229, (1989).
- [18] B. Scholkopf and A. Smola, *Learning with Kernels*, MIT Press, 2001.
- [19] B. Scholkopf, J. Weston, E. Eskin, C. Leslie, and W. Noble, *Dealing with Large Diagonals in Kernel Matrices*, volume 243 of *Lecture Notes in Computer Science*, Springer, 2002.
- [20] V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.
- [21] H. Verrelst, Y. Moreau, J. Vandewalle, and D. Timmerman, ‘Use of a multi-layer perceptron to predict malignancy in ovarian tumors’, in *Advances in Neural Information Processing Systems*, volume 10, (1998).
- [22] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik, ‘Feature selection for svms’, in *Advances in Neural Information Processing Systems*, volume 13, (2001).
- [23] L. Yan, R. Rodier, MC Mozer, and R. Wolniewicz, ‘Optimizing classifier performance via the wilcoxon-mann-withney statistics’, in *Proceedings of the 20th International Conference on Machine Learning*, (2003).

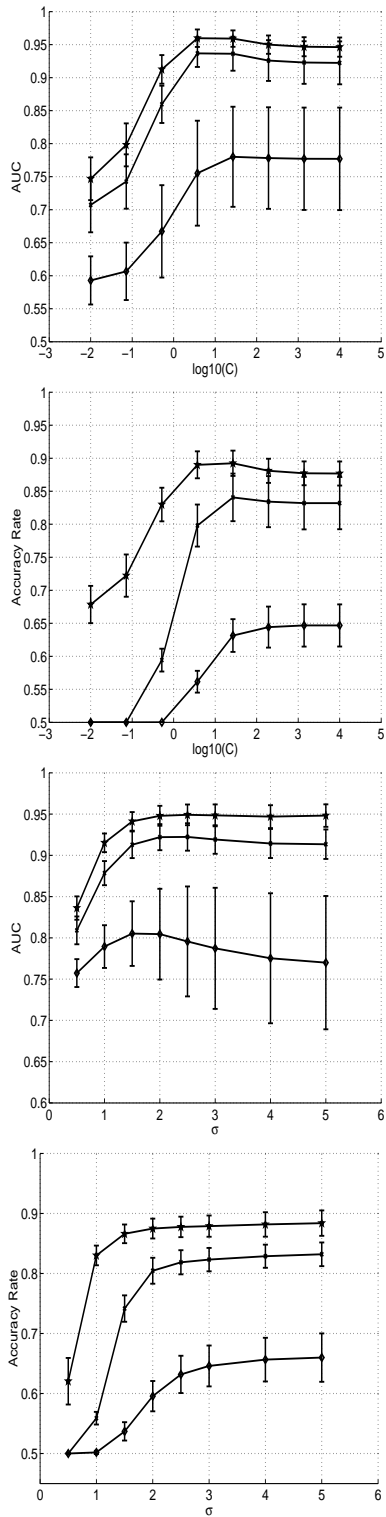


Figure 3. 2-norm SVMs : (top) AUC and accuracy performances for a fixed kernel parameter $\sigma = 3$ and a varying penalization parameter C . (bottom) AUC and accuracy performances for a fixed penalization parameter $C = 10$ and a varying kernel parameter. Curves in each plot correspond from top to bottom to different results from different data set skewness c : 0.5, 0.3, 0.1.

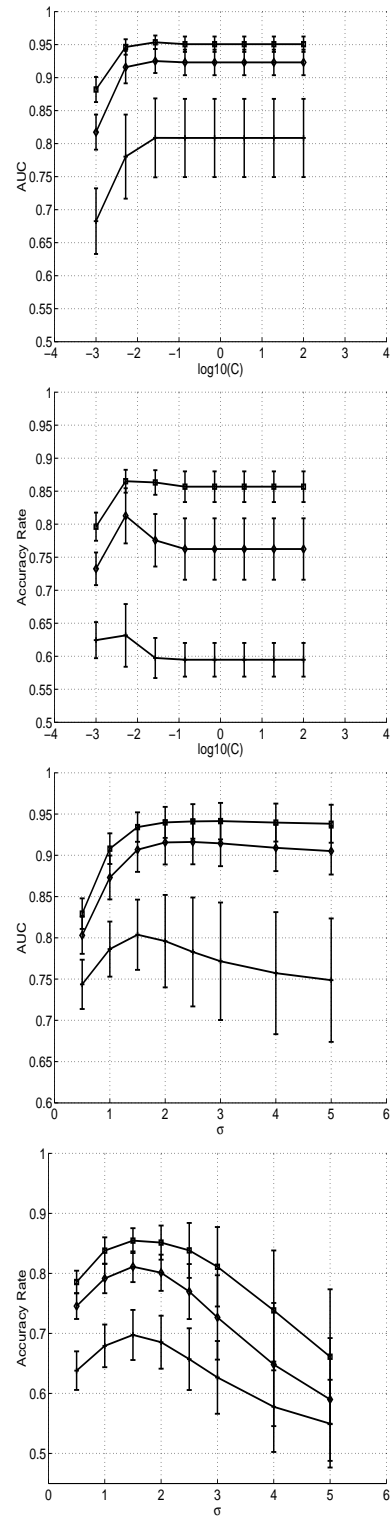


Figure 4. ROC-SVM : (top) AUC and accuracy performances for a fixed kernel parameter $\sigma = 3$ and a varying penalization parameter C . (bottom) AUC and accuracy performances for a fixed penalization parameter $C = 10$ and a varying kernel parameter. Curves in each plot correspond from top to bottom to different results from different data set skewness c : 0.5, 0.3, 0.1.

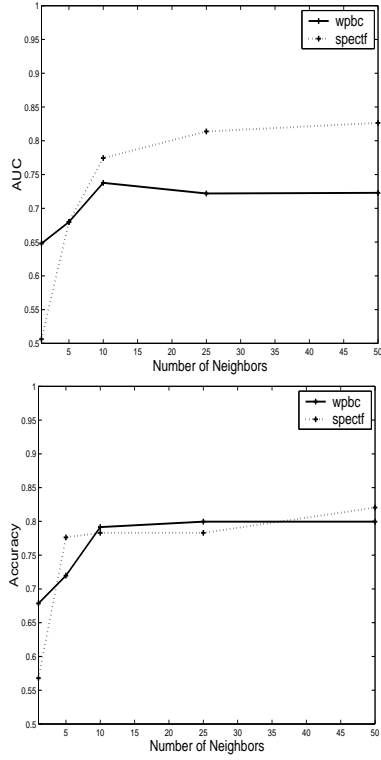


Figure 5. Results of ROC optimizer SVM on benchmark datasets. (a) Best AUC achieved over a range of kernel hyperparameters vs number of neighbors. (b) Best accuracy achieved over a range of kernel hyperparameters vs number of neighbors