A Linear Combination of Classifiers via Rank Margin Maximization

Claudio Marrocco, Paolo Simeone, and Francesco Tortorella

DAEIMI - Università degli Studi di Cassino
Via G. Di Biasio 43, 03043 Cassino (FR), Italia
{c.marrocco,paolo.simeone,tortorella}@unicas.it

Abstract. The method we present aims at building a weighted linear combination of already trained dichotomizers, where the weights are determined to maximize the minimum rank margin of the resulting ranking system. This is particularly suited for real applications where it is difficult to exactly determine key parameters such as costs and priors. In such cases ranking is needed rather than classification. A ranker can be seen as a more basic system than a classifier since it ranks the samples according to the value assigned by the classifier to each of them. Experiments on popular benchmarks along with a comparison with other typical rankers are proposed to show how effective can be the approach.

Keywords: Margin, Ranking, Combination of Classifiers.

1 Introduction

Many effective classification systems adopted in a variety of real applications make a proficient use of combining techniques to solve two class problems. As a matter of fact the combination of classifiers is a reliable technique to improve the overall performance, since it exploits the strength of the classifiers to be combined while reduces the effects of their weaknesses. Moreover the fusion of already available classifiers gives the user the opportunity to obtain simply and quickly an optimized system using them as building blocks, thus avoiding to restart from the beginning the design of a new classification system.

Several methods have been proposed to combine classifiers [11] and, among them, one of the most common technique is certainly the linear combination of the outputs of the classifiers. Extended studies have been conducted on this issue [8], and in particular have considered the weighted averaging strategies which are the basis of some popular algorithms like Bagging [2] or Boosting [7]. Boosting techniques build a classifier as a convex combination of several *weak* classifiers; each of them is in turn generated by dynamically reweighing training samples on the basis of previous classification results provided by the weak classifiers already constructed.

Such approach revealed to be really effective in obtaining classifiers with good generalization characteristics. To this regard, the work of Schapire et al. [13] has analyzed the boosting approach in terms of *margin maximization*, where the *margin* is a measure for the accuracy confidence of a classifier which can be considered as an important indicator of its generalization capacity. They calculated an upper bound on the generalization

E.R. Hancock et al. (Eds.): SSPR & SPR 2010, LNCS 6218, pp. 650–659, 2010.

error of resulting classifier and showed how the increase of the margin corresponded to an improvement of such bound. However, it is worth noting that this framework is applicable only in the cases where the accuracy is the most suitable index to evaluate the performance of the classification system, i.e. when the values of the classification costs and of the priors are known and fixed. For applications for which these parameters are not precisely known or are changing over time (*imprecise environments*), the accuracy becomes useless and other indices should be preferred such as the *Area under the ROC curve* (*AUC*). To understand the reason for this preference, we have to recall that, when the accuracy is used, we assume that a threshold is fixed on the classifier output on the basis of given costs and priors; accordingly, the accuracy measures the probability that the samples to be classified are correctly ordered with respect to the threshold. On the other side, the AUC measures the probability that a classifier correctly ranks two samples belonging to opposite classes and does not take into account any threshold; in other words, AUC provides an evaluation of the classifier quality independent of a particular setting of costs/priors.

In this framework, the concept of margin cannot be used and the *rank margin* should be employed instead, which gives a measure of the ranking confidence of the classifier. On this basis, Rudin et al. [12] have studied the generalization capability of RankBoost [6], a learning algorithm expressly designed to build systems for ranking preferences, and defined some bounds related to the rank margin value reached during the training phase. However these papers focus exclusively on how to build a new classifier from the scratch.

The aim of this paper is different from [12] and [6] since it presents a method to build a linear combination of already trained dichotomizers. The weights are determined in such a way to maximize the rank margin of the resulting system and thus to optimize its performance in terms of AUC. Several experiments performed on publicly available data sets have shown that this method is particularly effective.

The paper has been organized as follows: in section 2 the concepts of margin and rank margin are briefly explained together with their characteristics, while section 3 presents the method for calculating the weights of the linear combination based on the rank margin maximization. In section 4 experiments on some popular benchmark data are illustrated. Finally, in section 5 we draw some conclusions and propose some future developments.

2 Margins and Ranking

Let us consider a two class problem defined on a training set $S = (X, Y)$ containing N samples $X = \{\mathbf{x}_i\}$ associated to N labels $Y = \{\mathbf{y}_i\}$ with $y_i \in \{-1, +1\}$ where $i = 1, \cdots, N$. A classifier f can be described as a mapping from X to the interval $[-1, +1]$ such that a sample $\mathbf{x} \in X$ is assigned to one of the classes according to $\text{sgn}\,(f\,(\mathbf{x}))$. If we assume that y_i is the correct label of $\mathbf{x}_i$, the *sample margin* (or *hard margin*) associated to $\mathbf{x}_i$ is given by $y_i f(\mathbf{x}_i)$. As a consequence, f provides a wrong prediction for $\mathbf{x}_i$ if the sample margin is negative.

Generally *the margin of a classifier* (or *minimum margin*) f can be defined as the minimum margin value over the training set: $\mu(f) = \min_i(y_i f(\mathbf{x}_i))$. The classifier margin has a straightforward interpretation [4]: it is the distance that the classifier can travel in the feature space without changing the way it labels any of the sample points and thus, it represents one of the most relevant factor for improving generalization.

However, the concept of margin can not be used when we are in an imprecise environment where priors and costs are not known. In such a case a ranker becomes more useful than a classifier. The notion of ranking is germane to that of classification. In particular, ranking can be seen as an action on data more basic than classification: if no threshold is imposed on the output of the classifier (i.e. we are evaluating its performance independently of class priors and costs), the only possible operation is to rank the samples according to the value assigned by the classifier to each of them. Thus, the margin of a classifier should be replaced by the margin of the ranking function. To illustrate this point, let us define *crucial pair* and indicate with the concise notation (i, k) a pair of samples $\mathbf{x}_i \in X$ and $\mathbf{x}_k \in X$ associated respectively to a positive and a negative label $y_i = +1$ and $y_k = -1$. The term *crucial* is due to the fact that, for this kind of pairs, the classifier should guarantee that $f(\mathbf{x}_i) > f(\mathbf{x}_k)$, while this is not required for two samples belonging to the same class. On this basis, the *crucial pair margin* can be defined as the difference $f(\mathbf{x}_i) - f(\mathbf{x}_k)$; it is evident that a negative value for the margin indicates that the corresponding pair is erroneously ranked. Analogously to the sample margin, it is possible to define the *margin of the ranking function* or *rank margin* as the minimum value of the margin over all the existing crucial pairs:

$$\rho(f) = \min_{(i,k):\substack{i\ =\ 1,\ldots,N^+ \\ k\ =\ 1,\ldots,N^-}} \Big(f(\mathbf{x}_i) - f(\mathbf{x}_k) \Big). \tag{1}$$

As for classification, the rank margin theory has been used as a tool to analyze the generalization ability of learning algorithm for rankers based on boosting techniques. An algorithm belonging to this category is RankBoost [6] where the redistribution of the weights on the crucial pairs is done after the weak learners have been employed for ranking the pairs. As for AdaBoost [13], it has been proved that there is a strict relation between the generalization capability of RankBoost and its rank margin maximization. It is worth noting, however, that this method does not rely on a global optimization of the rank margin, but works locally. In fact, at each iteration of Rankboost, the crucial pairs with the minimum rank margin receive the highest weights and thus affect the construction of the whole ranker. Notwithstanding, this process converges towards the maximization of the rank margin [12].

Another issue to be pointed out is that this algorithm only constructs from the scratch an ensemble of classifiers as different instances of a same base learning algorithm. Instead, as far as we know, the potential effectiveness of such a combination has not yet been examined when the classifiers of the ensemble are built independently and not according to a boosting approach.

3 Rank Margin Maximization via Linear Programming

In this section we extend the concept of rank margin to the combination of K already trained classifiers $f_j\left(\mathbf{x}\right) \rightarrow [-1, +1]$ with $j = 1, \ldots, K$. Let us consider the N^+ and

N^- samples of the training set *X*. The rank margin provided by the *j*-th classifier over the crucial pair (i,k) is defined as:

$$\rho_{(i,k)}(f_j) = f_j\left(\mathbf{x}_i\right) - f_j\left(\mathbf{x}_k\right), \quad i = 1,2,\ldots,N^+, k = 1,2,\ldots,N^- \tag{2}$$

i.e., f_j correctly ranks $\mathbf{x}_i$ iff $\rho_{(i,k)}(f_j) > 0$. Let us now consider the linear combination of the K classifiers:

$$f_c\left(\mathbf{x}\right) = \sum_{j=1}^{K} w_j f_j(\mathbf{x}) \tag{3}$$

with $w_j \geq 0$ and $\sum_{j=1}^{K} w_j = 1$. The rank margin provided by f_c over the crucial pair (i,k) is thus

$$\rho_{(i,k)}(f_c) = \sum_{j=1}^{K} w_j f_j(\mathbf{x}_i) - \sum_{j=1}^{K} w_j f_j(\mathbf{x}_k) = \sum_{j=1}^{K} w_j \rho_{(i,k)}(f_j) \tag{4}$$

while the margin of f_c is $\rho = \min_{(i,k)} \rho_{(i,k)}(f_c)$. Actually the margin ρ depends on the weights $\mathbf{w} = \{w_1, w_2, \cdots, w_K\}$ and thus such weights can be chosen to make the margin as large as possible. In this way we have a max-min problem which can be written as:

$$\begin{aligned} &\text{maximize} && \left(\min_{i} \sum_{j=1}^{K} w_j \rho_{(i,k)}(f_j)\right) \\ &\text{subject to} && \sum_{j=1}^{K} w_j = 1 \\ &&& w_j \geq 0 \qquad j = 1,2,\ldots,K \end{aligned}$$

The problem can be recast as a linear problem [15] if we introduce the margin ρ as a new variable:

$$\begin{aligned} &\text{maximize} && \rho \\ &\text{subject to} && \\ &&& \sum_{j=1}^{K} w_j \rho_{(i,k)}(f_j) \geq \rho \quad i = 1,2,\ldots,N^+, k = 1,2,\ldots,N^- \\ &&& \sum_{j=1}^{K} w_j = 1 \\ &&& w_j \geq 0 \qquad j = 1,2,\ldots,K \end{aligned}$$

If we collect the margins in a $N^+N^- \times K$ matrix $\mathbf{R} = \{\rho_{(i,k)}(f_j)\}$, the weights in a vector $\mathbf{w}$ and define $\mathbf{e}_t$ the column vector consisting of t ones and $\mathbf{z}_t$ the column vector consisting of t zeros, the problem can be written in block-matrix form:

$$\begin{aligned} &\text{maximize} \qquad \begin{bmatrix} \mathbf{z}_K^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \mu \end{bmatrix} \\ &\text{subject to} \\ &\qquad \begin{bmatrix} -\mathbf{R} & \mathbf{e}_N \\ \mathbf{e}_N^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \rho \end{bmatrix} \begin{matrix} \leq \\ = \end{matrix} \begin{bmatrix} \mathbf{z}_N \\ 1 \end{bmatrix} \\ &\qquad\qquad \mathbf{w} \geq \mathbf{z}_K \end{aligned}$$

As a final remark, it is worth noting that to solve this problem we could use any one of the numerous linear programming methods available. However, it should be taken into account that the number of constraints could be very large since it equals the number of crucial pairs in the training set.

4 Experiments

Ten publicly available two class data sets were chosen from the UCI machine learning repository [1] to evaluate the performance of our approach. A summary of the employed data sets is reported in table 1. The features were previously scaled in order to have zero mean and unitary standard deviation. To avoid any bias in the comparison, 10 runs of a multiple hold out procedure have been performed on all the data sets. Each data set has been divided in three parts: a training set for the dicothomizers, a tuning set to train the combiner in order to have the optimal weights and a test set to evaluate the performance.

Modest AdaBoost [16] has been chosen as base classifier. Its algorithm adopts a CART decision tree with a maximum depth equal to 3 and decision stumps as nodes functions and a number of boosting steps equal to 10. To have a lower correlation between the built classifiers a random, but uniformly distributed, weight initialization has been done.

In order to compare the combining rules we considered the AUC as a performance measure. AUC values are unitary when all the instances are correctly interpreted by the learner, i.e. what is called a separable case. In terms of ranking it means that the algorithm is consistent with all the crucial pairs: all the positive instances are ranked

Table 1. Summary of the used data sets

Name	Samples	Features	% N^+	% N^-
Australian	690	14	44.49	55.51
Balance	625	4	54.01	45.99
Breast	699	16	65.01	34.99
Cleveland	303	13	54.13	45.87
Contraceptive	1473	9	42.70	57.30
Hayes	132	4	50.39	49.61
Housing	506	12	49.21	50.79
Ionosphere	351	34	64.10	35.90
Liver	345	6	57.97	42.03
Sonar	260	60	53.37	46.63

Table 2. AUCs obtained using 5 classifiers

Data Sets	**RankMargin**	**RankBoost**	**SVM**
Australian	**0.935(0.008**)	0.920(0.008)	0.929(0.009)
Balance	0.984(0.001)	0.959(0.016)	**0.986(0.004**)
Breast	**0.991(0.001**)	0.979(0.003)	0.979(0.010)
Cleveland	**0.885(0.010**)	0.840(0.026)	0.858(0.025)
Contraceptive	0.751(0.024)	0.752(0.013)	**0.762(0.012**)
Hayes	0.885(0.014)	0.865(0.030)	**0.893(0.039**)
Housing	**0.942(0.007**)	0.924(0.012)	**0.940(0.012**)
Ionosphere	**0.962(0.003**)	0.927(0.011)	0.944(0.019)
Liver	**0.737(0.033**)	0.707(0.035)	0.721(0.032)
Sonar	**0.892(0.016**)	0.837(0.033)	0.875(0.036)

Table 3. AUCs obtained using 7 classifiers

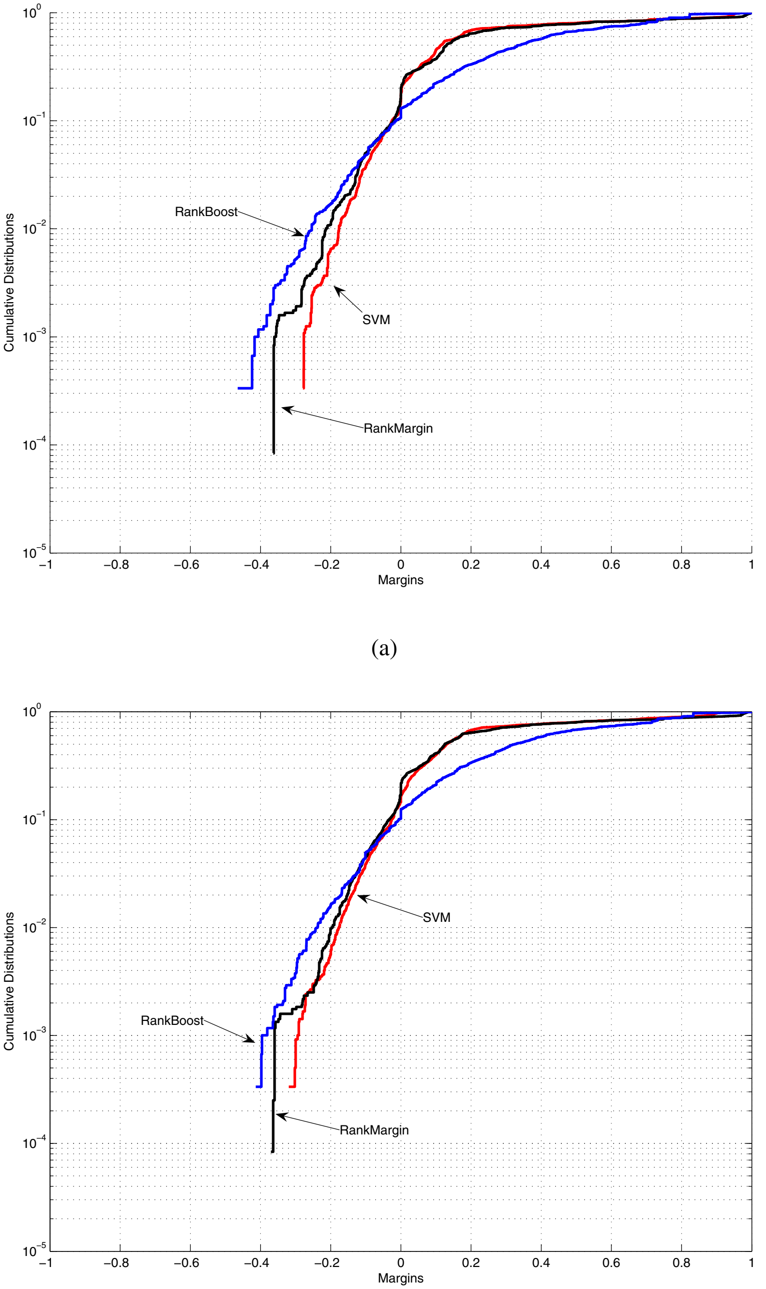
Data Sets	**RankMargin**	**RankBoost**	**SVM**
Australian	**0.932(0.007**)	0.920(0.008)	0.921(0.010)
Balance	0.984(0.001)	0.959(0.016)	**0.985(0.004**)
Breast	**0.991(0.001**)	0.979(0.003)	0.972(0.010)
Cleveland	**0.884(0.007**)	0.840(0.026)	0.847(0.022)
Contraceptive	0.753(0.015)	0.751(0.012)	**0.758(0.011**)
Hayes	**0.888(0.010**)	0.864(0.030)	0.878(0.025)
Housing	**0.942(0.005**)	0.924(0.012)	0.932(0.014)
Ionosphere	**0.962(0.002**)	0.927(0.011)	0.931(0.020)
Liver	**0.737(0.021**)	0.707(0.035)	0.702(0.034)
Sonar	**0.891(0.012**)	0.837(0.033)	0.863(0.037)

above the negative. Indeed an higher measure of the AUC is a quality factor for our combining rules.

Two classifiers notable for their ranking capacity have been adopted for a comparison with our RankMargin technique: RankBoost and Support Vector Machines (SVMs). The first one has been implemented by setting $T = 100$ iterations using a Matlab toolbox publicly available [3], the other one has been implemented by using SVM^{light} [10] with a linear kernel and default parameters.

To assess the performance of our method in comparison with the other considered combination rules, we have employed the *Friedman Two-Way Analysis of Variance by Ranks* test [14,5], a statistical non-parametric test which evaluates if in a set of L samples, at least two of the samples represent populations with different median value[1]. In this case, the null hypothesis corresponds to a not statistically significant difference in performance among the combination rules. When the null hypothesis is rejected, the *Holm's step-down procedure* [9,5] is applied as a $post-hoc$ test to identify which rule

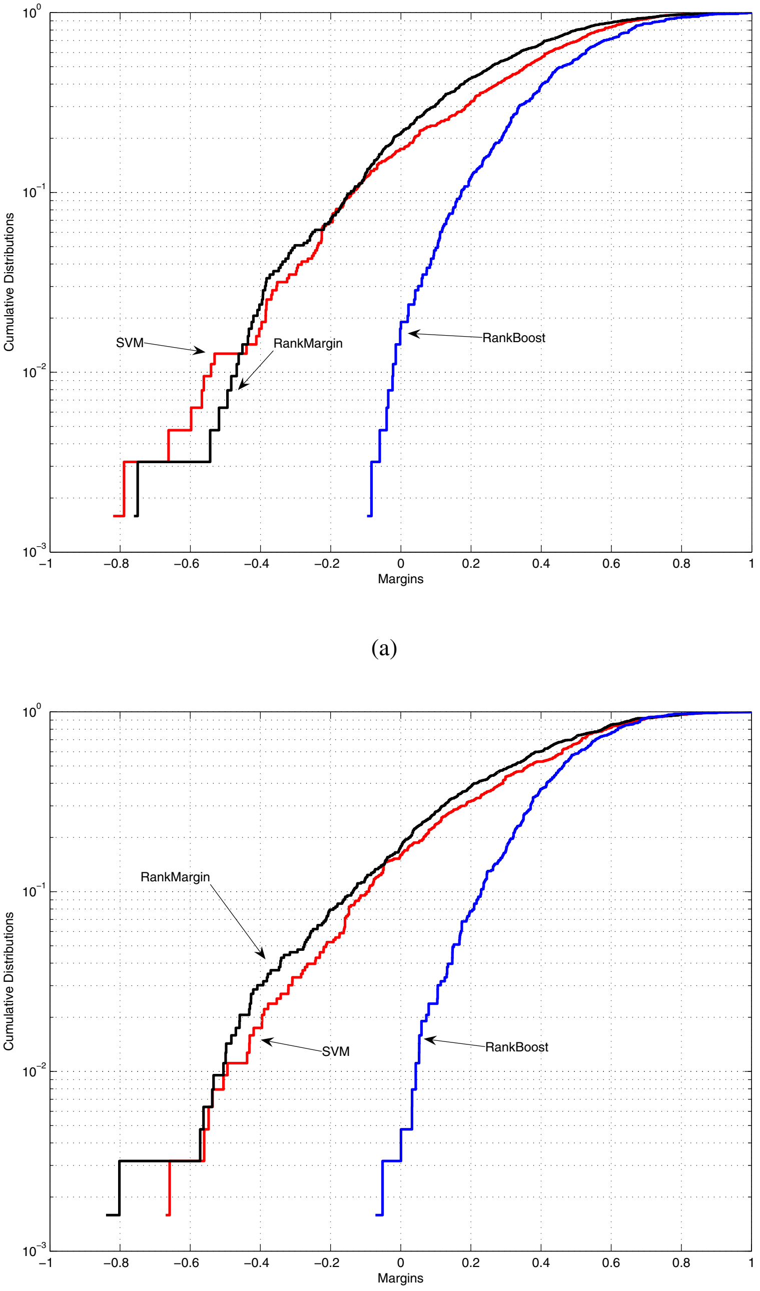
[1] We chose this test since its parametric counterpart, i.e. ANOVA, requires that the samples are drawn from normal distributions and the distributions have equal variance [14] and this is not assured in our test bed.



(a)

(b)

Fig. 1. Rank margin distributions graphs for the employed combination rules on the Contraceptive data set when combining 5 (a) and 7 (b) classifiers. The scale on y-axis is logarithmic.



(a)

(b)

Fig. 2. Rank margin distributions graphs for the employed combination rules on the Liver data set when combining 5 (a) and 7 (b) classifiers. The scale on y-axis is logarithmic.

performs significantly better or worse than the proposed method. Both the tests have been performed with $\alpha = 0.01$.

Results in terms of mean AUC (and standard deviation) are shown in tables 2 and 3 which differs for the number of combined classifiers (respectively 5 and 7). A bolded value means that the corresponding ranker has a statistically better performance on such data set.

Performance of our algorithm proved to be better for the majority of examined data sets. In particular in only 3 cases SVMs gave better performance when combining 5 classifiers, while there was a tie for the Housing data set. When combining 7 classifiers the results are even better: 8 out of 10 data sets. It is worth noting that RankBoost never outperforms our method. Some final considerations could be made about the comparison with RankBoost that never outperforms our method. Since RankBoost algorithm is not conceived to maximize the margin of the rank function at each iteration, such result is an empirical proof of how RankMargin gives an improvement of the overall performance of a ranker.

A second experiment has been done to show the behavior of the rank margin based combination rule on the training set. Accordingly we plotted the cumulative distributions of rank margins on the training set provided by RankMargin and the other employed fusion rules. In fig. 1 and 2 we report the margin cdfs for the proposed approach in comparison with the other rules respectively for the Contraceptive and Liver data sets when using 5 and 7 Modest AdaBoost as base classifiers.

The first graphs, both (a) and (b), show that the SVM gives better results on Contraceptive data set. This is perfectly coherent with the test results shown in tables 2 and 3. It can be observed how SVM maintains the same trend observed for training set when predicting test results, thus SVM keeps performing better of RankMargin in this case. RankBoost instead performs worse of both approaches even if the minimum rank margin on the training set is comparable with the other two techniques.

On the other hand in the second graphs it is possible to note that RankBoost exhibits clearly higher performance than the other approaches in terms of minimum rank margin. This is probably due to the fact that the boosting approach focuses on the most difficult samples of the training set to be classified giving almost perfect results on them. Another possible reason is given by the non linear nature of the combination built by RankBoost which could increase the minimum rank margin much more than SVM and RankMargin. Nevertheless, the higher complexity of the RankBoost combination reveals on the test set a worse generalization capability with respect to both SVM and RankMargin. These latter methods construct both a linear combination and thus the distribution of the margins are quite similar. However, SVM provide an optimal separating hyperplane with equal margins from the two classes, while RankMargin has not such a constraint of symmetrical margins and this reflects in a better generalization capability.

5 Conclusions and Future Works

In this paper we have studied a new algorithm to combine scores of base classifiers. Such algorithm aims at the maximization of the margin for the ranking function in order to accomplish a better performance in terms of AUC for the linear combination of already trained dichotomizers. Results on the UCI data sets proved that our approach is reasonable and could be extended to plenty of applications.

Future developments will focus on the application of such technique to highly unbalanced data sets where AUC, which is independent from prior probabilities and costs, is a good performance measure, e.g. biometrics data. Another development can be in the relaxation of the constraint in the rank margin maximization by introducing slack variables that could be useful to face with noisy data.

References

1. Asuncion, A., Newman, D.J.: UCI machine learning repository (2007)
2. Breiman, L.: Bagging predictors. Machine Learning 26(2), 123–140 (1996)
3. Canu, S., Grandvalet, Y., Guigue, V., Rakotomamonjy, A.: Svm and kernel methods matlab toolbox. Perception Systèmes et Information, INSA de Rouen, Rouen, France (2005)
4. Crammer, K., Gilad-Bachrach, R., Navot, A., Tishby, N.: Margin analysis of the lvq algorithm. In: NIPS, pp. 462–469 (2002)
5. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research (7), 1–30 (2006)
6. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research 4, 933 (2003)
7. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of online learning and an application to boosting. Journal of Computer and System Sciences 55(1), 119 (1997)
8. Fumera, G., Roli, F.: A theoretical and experimental analysis of linear combiners for multiple classifier systems. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(6), 942 (2005)
9. Holm, S.: A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics 6, 65–70 (1979)
10. Joachims, T.: SVM light (2002), `http://svmlight.joachims.org`
11. Kuncheva, L.I.: Combining Pattern Classifiers. Methods and Algorithms. John Wiley & Sons, Chichester (2004)
12. Rudin, C., Cortes, C., Mohri, M., Schapire, R.: Margin-based ranking meets boosting in the middle. In: Proceedings of 18th Annual Conference on Computational Learning Theory (2005)
13. Schapire, R.E., Freund, Y., Barlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods. In: ICML, pp. 322–330 (1997)
14. Sheskin, D.J.: Handbook of parametric and nonparametric statistical procedures. Chapman & Hall, CRC (2000)
15. Vanderbei, R.J.: Linear Programming: Foundations and Extensions, 2nd edn. Springer, Heidelberg (2001)
16. Vezhnevets, A., Vezhnevets, V.: Modest adaboost - teaching adaboost to generalize better. In: Graphicon 2005 (2005)