Technical Report

No: BU-CE-1003

Risk Estimation by Maximizing the Area under ROC Curve

Murat Kurtcephe, H. Altay Güvenir

July 2010

Bilkent University
Department of Computer Engineering
06800 Ankara, Turkey
http://www.cs.bilkent.edu.tr

Risk Estimation by Maximizing the Area under ROC Curve

Murat Kurtcephe

Bilkent University, Computer Engineering Department 06800 Ankara, Turkey
+90 (312) 290 1945
+90 (312) 266 4047
kurtcephe@cs.bilkent.edu.tr

H. Altay Güvenir

Bilkent University, Computer Engineering Department 06800 Ankara, Turkey
+90 (312) 290 1252
+90 (312) 266 4047
guvenir@cs.bilkent.edu.tr

Abstract: Risks exist in many different domains; medical diagnoses, financial markets, fraud detection and insurance policies are some examples. Various risk measures and risk estimation systems have hitherto been proposed and this paper suggests a new risk estimation method. Risk estimation by maximizing the area under a receiver operating characteristics (ROC) curve (REMARC) defines risk estimation as a ranking problem. Since the area under an ROC curve (AUC) is related to measuring the quality of ranking, REMARC aims to maximize the AUC value on a single feature basis to obtain the best ranking possible on each feature. For a given categorical feature, we prove a sufficient condition that any function must satisfy to achieve the maximum AUC. Continuous features are also discretized by a method that uses AUC as a metric. Then, a heuristic is used to extend this maximization to all features of a dataset. REMARC can handle missing data, binary classes and continuous and nominal feature values. The REMARC method does not only estimate a single risk value, but also analyzes each feature and provides valuable information to domain experts for decision making. REMARC's performance is evaluated with many datasets in the UCI repository by using different state-of-the-art algorithms such as Support Vector Machines, naïve Bayes, decision trees and boosting methods. Evaluations of the AUC metric show REMARC achieves predictive performance significantly better compared with other machine learning classification methods and is also faster than most of them.

Keywords: Risk Estimation, AUC Maximization, AUC, Ranking

1 Introduction

Accurate prediction of risk is essential for life. Avoiding or being aware of risks in domains such as finance or medicine can save money and lives, respectively. The main motivation behind the research on risk-prediction systems is to improve system performance to avoid unwanted events or negative consequences.

This paper proposes a new risk measure and a supervised machine learning algorithm to estimate the values of this measure. The algorithm, learning from training instances, develops a model of the domain based on receiver operating characteristics (ROC) analysis, so that the area under ROC curves (AUC) of ordering the instances (Hanley and McNeil 1982) will be maximized; hence, the algorithm is called Risk Estimation by Maximizing the Area under ROC Curve (REMARC).

Specific risk estimation methods have been developed for finance (Bradley and Taqqu 2003), medicine (Conroy *et al.* 2003; D'Agostino *et al.* 2008) and insurance (Dowd and Blake 2006), to name some examples. Some of these methods are dependent on statistical models while some are based on machine learning algorithms. The machine learning algorithms are usually classification algorithms that can associate a certainty factor with their classification. The certainty factor for a predicted unwanted case is taken as the value of risk.

The word “risk” is generally taken to mean “an unwanted situation” (Giddens 1999). Although these unwanted cases may be severe, their likelihood of occurrence is usually rare. Therefore, datasets for such domains usually are unbalanced and the costs of misclassification are not symmetric. Classification algorithms that aim to maximize accuracy are not suitable for such unbalanced datasets (Sebag *et al*. 2004; Tax *et al*. 2006; Fawcett 2006). Instead, an alternative metric called AUC, proposed by Bradley (1997), is the evaluation metric to maximize. AUC has important features such as insensitivity to class distribution and cost distributions (Bradley 1997; Huang and Ling 2005; Fawcett 2006),which make it suitable for risk domains.

In risk domains, representing the risk score as a real value between 0 and 1 may not be sufficient, and even misleading; relatively ordering instances in terms of risk values may be much more informative. For example, instances can be located on a single dimension, where the safest cases are on one side and the riskiest cases are on the other side. Since it has been shown by Hanley and McNeil (1982) that AUC is able to qualify ranking instances, maximizing AUC also leads to the best ranking. Recent research on maximizing AUC by Toh *et al.* (2008) and Rakotomamonjy (2004) also shows the importance of ranking instances.

The main contributions of the REMARC algorithm can be shown in three different ways. First, we show the conditions a risk scoring function must possess in order to achieve maximum AUC for a single feature dataset case. Second, the maximization of AUC is extended over the whole dataset by using a simple heuristic, which also depends on AUC's metric. Lastly, the human readable model formed by REMARC helps domain experts by indicating what features and how their particular values affect the risks.

In the next section detailed information about risks and risk domains are given. Section 3 covers ROC, AUC and research on AUC maximization. In Section 4, the theoretical background of the REMARC method and implementation details are given. Section 5 presents the empirical evaluation of real-life datasets. Finally, Section 6 concludes with conclusions and future work.

2 Risks

Risk has always been a normal occurrence. Risks such as a complication from surgery, a fraudulent financial transaction, a firm going into financial distress and an e-mail being spam are all part of today's world. Giddens (1999) claims that the ideas of risk and responsibility are closely linked in a risk society, and suggests that legal theorists and practitioners should also concern themselves with the idea and reality of risk. The word "risk" is commonly used in daily life, because of its popularity in the media, however, a formal definition is needed.

2.1 Definitions of Risk

Hansson (2007) gives five definitions of risk commonly used in different disciplines. Hansson's third definition is the most suitable for defining the risk used in this work: "The probability that an unwanted event may or may not occur". For example, the risk of a credit card transaction being fraudulent is 17%.

2.2 Risk Domains

Risk implies an unwanted situation. In medicine, mortality and morbidity are two unwanted situations. In finance, money loss and bankruptcy are examples. Since the consequences of these situations are crucial, in order to avoid them extensive research continues on this subject. As an example, it is possible to find books written on specific domains such as process management systems risk estimation (Cameron and Raman 2005).

According to Shishkin and Savkov (2009) some of the most popular commercial risk analysis tools for financial domains are "Risk Watch" (www.riskwatch.com, USA) and "Commercial Risk Analysis and Management Methodology-CRAMM" (www.cramm.com). Other than the commercial tools, concepts such as Value-At-Risk (VAR) and other models can be found in Bradley and Taqqu (2003) and Huang 2010. Stoyan *et al.* (2008) provide a survey on stochastic models for risk estimations. Recently, Ferrari and Paterlini (2007) proposed a new risk estimation method that claims a better performance than VAR.

In medicine, a risk scoring system based on logistic regression for cardiovascular surgery is proposed by Roques *et al.* (2003). Other scoring systems for the same domain also exist (Conroy *et al.* 2003; Hannan *et al.* 2006). A recent study by D'Agostino *et al.* (2008) shows that some of these scoring systems use Cox regression methods, which is proposed by Cox (1972).

2.3 Risk Estimation in Machine Learning

Risk estimation is not yet a major subarea of machine learning literature. Classification algorithms, which are able to output the confidence or probability of classification results, can be used to approximate risk estimation.

In a risk estimation system, a risk function that assigns higher values to risky instances than safer instances is crucial. In such a system, risk will be computed as a real value between 0 and 1, where 1 indicates the definite risk while 0 represents the safest situation. However, the absolute value of this risk score is also very important for the user. Assume *risk*() is a function that returns a real number between 0 and 1 as the estimation of the risk. Another risk function, *risk'*(), defined as $\sqrt{risk()}$, also returns a value between 0 and 1. Both of these functions will rank the instances in the same order, although their absolute risk values are different.

On any dataset gathered from a risk domain, two classes should be determined in order to distinguish a risky situation from a safe one. In this work, we will define these class labels as **p** (positive, unwanted class) and **n** (negative, safe class). For example, in a loan dataset, the class label **p** indicates a default, while label **n** indicates that the loan amount has been paid back.

Machine learning techniques have been applied to different domains in order to predict risk. In medicine, Colombet *et al.* (2000) evaluated three different machine learning algorithms in order to predict cardiovascular surgery risk. In Biagioli *et al.* (2006) Bayesian models were used to predict risks in coronary artery surgery operations and in Gamberger *et al.* (2000) machine learning results on a heart database were evaluated. Financial domains have also taken advantage of machine learning algorithms. Galindo and Tamayo (2000) evaluated machine learning and statistical methods in order predict credit risks. Kim (2003) proposed a financial time series prediction system by using a support vector machine (SVM) and Min and Lee (2005) tried to predict bankruptcy risk by using optimal kernel functions for SVM. However, to the best of our knowledge, a risk estimation system that aims to maximize the AUC metric has never been proposed. The ROC curves and AUC metric will be examined in detail before explaining the REMARC method. The next section elaborates on the features of ROC and AUC and their appropriateness for this work.

3 ROC, AUC and AUC maximization

Since their application to machine learning, ROC graphs and the AUC metric have become popular; AUC is used in evaluating machine learning algorithms and as a learning criterion. We explain the properties that make AUC a better metric than accuracy and discuss the existing research on AUC maximization.

3.1 Receiver Operating Characteristics (ROC)

The first application of ROC graphs dates back to World War II, where they were used to analyze radar signals (Krzanowski & Hand, 2009). Since then, they have been used in areas such as signal detection and medicine (Green and Swets 1966; Zweig and Campbell 1993; Pepe 2003). The first application to machine learning is done by Spackman (1989). According to Fawcett's definition, the ROC graph is a tool that can be used to visualize, organize and select classifiers based on their performance (Fawcett 2006). It has become a popular performance measure in the machine learning community after it has been realized that accuracy is often a poor metric to evaluate classifier performance (Provost and Fawcett 1997; Provost *et al.* 1998; Huang and Ling 2005).

The ROC literature is more established to deal with binary classification (two classes) problems than multi-class ones. At the end of the classification phase, some classifiers simply map each instance to a class label (discrete output). Some classifiers are able to estimate the probability of an instance belonging to a specific class such as naïve Bayes or neural networks (continuous valued output, also called score). Classifiers produce a discrete output represented by only one point in the ROC space, since only one confusion matrix is produced from their classification output. Continuous-output-producing classifiers can have more than one confusion matrix by applying different thresholds to predict class membership. All instances with a score greater than the threshold are predicted as to be **p** class and all others are predicted as to be **n** class. Therefore, for each threshold value, a separate confusion matrix is obtained. The number of confusion matrices is equal to the number of ROC points on an ROC graph. With the method proposed by Domingos (1999), it is possible to obtain ROC curves even for algorithms that are unable to produce scores.

3.1.1 ROC Space

ROC space is a two dimensional space with a range of (0.0, 1.1) on both axes. In ROC space the y-axis represents the true positive rate (*TPR*) of a classification output and the x-axis represents the false positive rate (*FPR*).

To calculate *TPR* and *FPR* values, the definitions of the elements in the confusion matrix must be given. The structure of a confusion matrix is shown in Fig. 1. True positives (*TP*) and false positives (*FP*) are the most important elements of the confusion matrix for ROC graphs. For each threshold value, *TP* is equal to the number of positive instances (those that have been classified correctly) and *FP* is equal to the number of negative instances (those that have been misclassified).

TPR and *FPR* values are calculated by using Eq. 1. In this equation *N* is the number of total negative instances and *P* is the number of total positive instances.

$$TPR = TP / P \qquad (1)$$

$$FPR = FP / F$$

3.1.2 Formation of ROC Curve

As mentioned above, the classifiers producing continuous output can form a curve since they are represented by more than one point in the ROC graph. To draw the ROC graph, different threshold values are selected and different confusion matrices are formed.

By varying the threshold between $-\infty$ and $+\infty$, an infinite number of ROC points can be produced for a given classification output. However, this operation is computationally costly and it is possible to form the ROC curve more efficiently with other approaches.

As proposed by Fawcett (2006), in order to calculate the ROC curve efficiently, classification scores are sorted in an increasing order first. Starting from $-\infty$, each distinct score element is taken as a threshold; TPR and FPR values are calculated using Eq. 1.

As an example, assume that the score values for test instances and actual class labels for a toy dataset are given in Table 1. The ROC curve for this toy dataset is shown in Fig. 2. In this figure, each ROC point is given with the threshold value used to calculate it. In a dataset with *S* distinct classifier scores, there are *S*+1thresholds including $-\infty$ and the same number of ROC points. Since there are eight distinct score values in this toy dataset, there are nine ROC points. With this simple method it is possible to calculate the ROC curve in linear time.

It is possible to divide the ROC space into three regions: the region above *y=x* line, the area below *y=x* line and the points on the *y=x* line. The points on *y=x* line represent random performance. As an example, a classifier that has a point on (0.6,0.6) guesses the positive class 60% correctly, however it also has a 60% false positive rate. The points above the *y=x* line are those belonging to the classifiers that have an acceptable trade-off between the positive and negative classes; similarly, the points below the *y=x* line correspond to an unacceptable classification performance. A classifier's ROC point below the diagonal line can be negated by simply inverting the decision criteria of the classifier, replacing all **p** class labels with **n** class labels and vice versa. According to Flach and Wu (2003), classifiers below the diagonal have valuable information, but they are not able to use it.

3.2 Area under the ROC Curve (AUC)

ROC graphs are useful to visualize the performance of a classifier but a scalar value to compare classifiers is needed. In the literature, Bradley proposes the area under the ROC curve as a performance measure (1997). According to the AUC measure, the classifier with a higher AUC value performs better in general. A classifier can be outperformed by another classifier in some regions of ROC space, for some specific threshold values, even though the classifier, which has larger AUC, is better than the other.

The ROC graph space is a one-unit square. The highest possible AUC value is 1.0, which represents the perfect classification. In ROC graphs a 0.5 AUC value

means random guessing has occurred and values below 0.5 are not realistic as they can be negated by changing the decision criteria of the classifier.

The AUC value of a classifier is equal to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. Hanley and McNeil (1982) show that this is equal to the Wilcoxon test of ranks.

3.3 Why AUC is More Proper than Accuracy

There are several reasons why we chose AUC as a learning criterion in this work. The first reason is the independence of the decision threshold of the AUC metric. Since the risk estimation methods are not actual classifiers, unless a threshold is fixed it is not possible to calculate an accuracy value. As mentioned in Section 2.3, the first task of a risk estimation method is ranking instances correctly. Since AUC has the ability to measure the quality of ranking, it is better than an accuracy metric on this basis.

Another reason regards the discrimination power of the accuracy and AUC metrics. Bradley (1997) was the first author to question the applicability of accuracy metrics in classifier algorithms and to recommend the use of AUC instead. Provost *et al.* (1998) also questioned the applicability of accuracy metrics in classification algorithms and suggested ROC analysis as a powerful alternate tool. Rosset (2004) claimed that even if the goal is to maximize accuracy, AUC may be better than empirical error for discriminating between models. The formal proof of the superiority that AUC has over accuracy is later given by Huang and Ling (2005). In their work, the authors showed that AUC is a statistically consistent and more discriminating metric than accuracy. These works clearly show the discriminatory power of the AUC metric.

Skewed (unbalanced) datasets is another reason to prefer AUC as a metric. This situation occurs when the difference between class priors is high. Risk areas such as medicine (Mac Namee *et al.* 2002; Tax *et al.* 2006) or fraud detection (Fawcett and Provost 1997) are examples of skewed datasets. For example, a classifier that predicts all instances as negative even though a few of the instances achieve very

high accuracies is misleading (Rakotomamonjy 2004). In addition, class distribution can change over time. For example, if in a financial crisis a large number of banks claim bankruptcy this can change class distribution drastically. In order to solve such problems, AUC, which is insensitive to class distributions, is preferred.

Lastly, misclassification costs cannot be determined for most risk domains. As noted above, skewed datasets are common in real life. In a domain with unbalanced class distribution, when the true misclassification cost is higher than implied by the distribution of training set examples, this situation becomes problematic (Maloof 2003). Since AUC is also insensitive to misclassification cost (Provost and Fawcett 2001) it is preferred in this work.

3.4 AUC Maximization

Most classification algorithms are designed to maximize accuracy (or error rate). Since accuracy is a classification performance criteria, algorithms that maximize it give better predictive performance. However, because of the abovementioned drawbacks to the accuracy metric for some domains, AUC has become more popular. It has been shown that maximizing accuracy does not lead to maximizing AUC (Cortes and Mohri 2003; Yan *et al.* 2003). As a result, new algorithms maximizing AUC have been proposed.

Some approximation methods to maximize the global AUC value have been proposed by researchers (Mozer *et al.* 2002; Yan *et al.* 2003; Herschtal and Raskutti 2004). Ferri *et al.* (2002) proposed a method to locally optimize AUC in decision tree learning, and Cortes and Mohri (2003) proposed boosted decision stumps. To maximize AUC in rule learning, several new algorithms have been proposed (Boström 2005; Prati and Flach 2004; Fawcett 2001). A nonparametric linear classifier based on the local maximization of AUC was proposed by Marrocco *et al.* (2008). A ROC-based genetic learning algorithm has been proposed by Sebag *et al.* (2004). Marrocco *et al.* (2006) used linear combinations of dichotomizers for the same purpose. Freund *et al.* (2003) gave a boosting algorithm combining multiple rankings. Cortes and Mohri (2003) showed that this approach also aims to maximize AUC. A method by Tax *et al.* (2006) that weighs

features linearly by optimizing AUC has been proposed and applied to the detection of interstitial lung disease. Ataman *et al.* (2006) advocate an AUC-maximizing algorithm with linear programming. Rakotomamonjy (2004) suggested rank optimizing kernels for SVMs to maximize AUC. Ling and Zhang (2002) compare AUC-based Tree-Augmented Naïve Bayes (TAN) and error-based TAN algorithms; the AUC-based algorithms are shown to produce more accurate rankings. More recently, Calders and Jaroszewicz (2007) proposed a polynomial approximation of AUC to optimize it efficiently. Linear combinations of classifiers are used to maximize AUC in biometric scores fusion in Toh *et al.* (2008). Han and Zhao (2010) propose a linear classifier based on active learning, which maximizes AUC.

4 REMARC

REMARC is a risk estimation method designed to maximize the AUC metric. The REMARC algorithm reduces the problem of finding a risk function for the whole set of features into finding a risk function for a single categorical feature, and then combines these functions to form one risk function covering all features. We will show here that it is possible to determine risk functions that achieves the maximum AUC for a single categorical feature. REMARC discretizes the numerical features by an algorithm called MAD, proposed by Kurtcephe and Guvenir (2010). The MAD method discretizes a continuous feature in a way that results in a categorical feature by maximizing the AUC.

For a given query, REMARC outputs a real value *r* in the range of [0,1] as the estimated risk of being the unwanted state. This *r* value is roughly the probability that the query instance will be in the **p** class. It is only a rough estimate of probability, since it is very likely that no other instance with exactly the same feature values has been observed in the training set. The REMARC algorithm determines this estimated probability by computing the weighted average of probabilities computed on single features. The weight of a feature is a linear function of its AUC value. calculated by the risk estimates for each instance in the training set. A higher value of AUC for a feature is an indication of its higher relevance in determining the class label.

In the following sections we will show how a risk function can be defined for a single categorical feature. Since, the REMARC algorithm requires the features to be categorical, we will propose a method to discretize a continuous feature so that the AUC of the new categorical feature will be the maximum.

4.1 Single Categorical Feature Case

A categorical feature has a finite set of choices. Let $V = \{v_1, v_2, \ldots v_n\}$ be a categorical feature and v_i a categorical value that feature V can take. The dataset D is a set of instances represented by a vector of n value and class label as $\langle v,c \rangle$, where $v \in V$ and $c \in \{\mathbf{p},\mathbf{n}\}$.

Given a dataset D with a single categorical feature whose value set is $V = \{v_0, v_1,\ldots, v_n\}$, a risk function r: $V \rightarrow [0,1]$ can be defined to rank the values in V. According to this risk function, a value v_i comes after a value v_j if and only if $r(v_i) > r(v_j)$; hence r defines a partial ordering on the set V. A pair of consecutive values v_i and v_{i+1} defines a ROC point R_i on the ROC space. The coordinates of the point R_i are (FPR_i, TPR_i).

Theorem 1: Let D be a dataset with a single categorical feature whose value set is $V = \{v_0, v_1, \ldots, v_n\}$. Let r: $V \rightarrow [0,1]$ be the risk function that orders the values of V, as v_{i+1} comes after v_i if $r(v_{i+1}) > r(v_i)$, for all values of $0 \leq i < n$. If the values of the risk function for two consecutive values v_i and v_{i+1} are swapped, then the only change in the ROC curve is that the ROC point corresponding to the v_i and v_{i+1} values moves to a new location so that the slopes of the line segments adjacent to that ROC point are swapped.

Proof: The slope of the line segment between two consecutive ROC points R_i and R_{i+1} is

$$s_i = \frac{TPR_i - TPR_{i+1}}{FPR_i - FPR_{i+1}} .$$

Since $TPR_i = \frac{TP_i}{P}$ and $FPR_i = \frac{FP_i}{N}$,

$$s_i = \frac{N}{P}\frac{TP_i - TP_{i+1}}{FP_i - FP_{i+1}} \ .$$

Further replacing $TP_i = P_i + TP_{i+1}$ and $FP_i = N_i + NP_{i+1}$,

$$s_i = \frac{N}{P}\frac{P_i}{N_i} \ .$$

Similarly, the slope of the line segment connecting the ROC points between R_{i+1} and R_{i+2} is

$$s_{i+1} = \frac{N}{P}\frac{P_{i+1}}{N_{i+1}} \ .$$

When the ranking of values v_i and v_{i+1} are changed, only the following changes take place:

$$P'_{i+1} = P_i, \quad P'_i = P_{i+1} \ ,$$
$$N'_{i+1} = N_i, \quad N'_i = N_{i+1} \ ,$$
$$\forall_{j \neq i, i+1} \quad P'_j = P_j \text{ and } N'_j = N_j \ .$$

With this change, only the ROC point R_i at (FPR_i, TPR_i) is replaced with a new ROC point R'_i at (FPR'_i, TPR'_i). The slopes of the new line segments adjoining R'_i are

$$s'_i = \frac{N}{P}\frac{P'_i}{N'_i} \text{ and } s'_{i+1} = \frac{N}{P}\frac{P'_{i+1}}{N'_{i+1}} \ .$$

Replacing the new count values with the old ones,

$s'_i = s_{i+1}$ and $s'_{i+1} = s_i$ are obtained. ■

For example, consider the dataset given below:

D={(a,n), (b,p), (b,n), (b,n), (b,n), (c,p), (c,p), (c,n), (c,n), (d,p), (d,p), (d,n)}, where *V* = {a, b, c, d}. If a risk function *r* orders the values of *V* as r(a) < r(b) < r(c) < r(d), the ROC curve shown in Fig. 3a) will be obtained. On the other hand, if the rankings of values b and c are swapped, the ROC curve shown in Fig. 3b will be obtained. A similar technique was used earlier by earlier by Flach and Wu (2003) to create better prediction models for classifies.

Theorem 1 shows how concavities in a ROC curve can be removed, resulting in a larger AUC. The next question is how to form the convex ROC curve. The

following theorem sets the necessary and sufficient condition for risk functions to satisfy so that their ROC curves are convex.

Theorem 2: Let D be a dataset with a single categorical feature that takes values from the set $V = \{v_0, v_1, \ldots, v_n\}$. Let r: $V \rightarrow [0,1]$ be the risk function that orders the values of V, as v_{1+1} comes after v_i if $r(v_{i+1}) > r(v_i)$, for all values of $0 \leq i < n$. In order for the ROC curve of the ordering by r to be convex, the following condition must be satisfied:

$$\forall_{0 \leq i < n} i \quad \frac{P_{i+1}}{N_{i+1}} \geq \frac{P_i}{N_i}, \tag{2}$$

where P_i is the number of **p**-labeled instances with value v_i, and N_i is the number of **n**-labeled instances with value v_i.

Proof: In order for the ROC curve to be convex, the slopes of all line segments connecting consecutive ROC points starting from the ROC point (1,1) must be non-decreasing.

Therefore, the condition for a convex ROC curve is

$$\forall_{0 \leq i < n} i \quad s_i \geq s_{i+1} \tag{3}$$

$$\forall_{0 \leq i < n} i \quad \frac{TPR_{i+1} - TPR_{i+2}}{FPR_{i+1} - FPR_{i+2}} \geq \frac{TPR_i - TPR_{i+1}}{FPR_i - FPR_{i+1}}$$

By definition, $TPR_i = \frac{TP_i}{P}$.

Further, due to the ordering of values, $TP_i = P_i + TP_{i+1}$.

Hence, $TPR_i - TPR_{i+1} = \frac{TPR_i}{P} - \frac{TPR_{i+1}}{P} = \frac{1}{P}\left(P_i + TP_{i+1} - TP_{i+1}\right) = \frac{P_i}{P}$

Similarly, $FPR_i - FPR_{i+1} = \frac{N_i}{N}$, $TPR_{i+1} - TPR_{i+2} = \frac{P_{i+1}}{P}$ and $FPR_{i+1} - FPR_{i+2} = \frac{N_{i+1}}{N}$.

Therefore, the inequality in Eq.3 can be rewritten as

$\forall_{0 \le i < n} i \quad \frac{P_{i+1}/P}{N_{i+1}/N} \ge \frac{P_i/P}{N_i/N}$.

Finally, $\forall_{0 \le i < n} i \quad \frac{P_{i+1}}{N_{i+1}} \ge \frac{P_i}{N_i}$. ■

Therefore, according to Theorem 2, any risk function *r* that assigns a higher value to v_{i+1} than to v_i when $\frac{P_{i+1}}{N_{i+1}} \ge \frac{P_i}{N_i}$, for all values of *V*, will result in a convex ROC curve. For example, a risk function defined as $r(v_i) = \frac{P_i}{N_i}$ will result in a convex ROC curve.

Theorem 3. Let *D* be a dataset with a single categorical feature whose value set is $V = \{v_0, v_1, \ldots, v_n\}$. Ignoring the ineffective ROC points that lie on a line, there exists exactly one convex ROC curve.

Proof: Since there exists only one possible ordering of values of *V* that satisfies the condition given in Theorem 1, there exists only one convex ROC curve. ■

The general assumptions for risk estimation problems are given in Eq. 4:

$$\forall_{0 \le i < n} i \quad P_i \ge 0 \quad , \quad \forall_{0 \le i < n} i \quad N_i \ge 0$$
$$P = \sum_{0}^{n-1} P_i > 0 \quad , \quad N = \sum_{0}^{n-1} N_i > 0 \qquad (4)$$

Although the dataset is guaranteed to have at least one instance with class label **p** and one instance with label **n**, it is possible that for some values of *i*, N_i may be 0. In such cases the risk function defined above will have undefined values. In order to avoid such problems, the risk can be defined as

$$r(v_i) = \frac{P_i}{P_i + N_i} \qquad (5)$$

Lemma 1. $\underset{0\le i<n}{\forall i} \quad \frac{P_{i+1}}{P_{i+1}+N_{i+1}} \ge \frac{P_i}{P_i+N_i}$ iff $\frac{P_{i+1}}{N_{i+1}} \ge \frac{P_i}{N_i}$

Proof : if $\underset{0\le i<n}{\forall i} \quad \frac{P_{i+1}}{P_{i+1}+N_{i+1}} \ge \frac{P_i}{P_i+N_i}$, then $\underset{0\le i<n}{\forall i} \quad P_{i+1}(P_i+N_i) \ge P_i(P_{i+1}+N_{i+1})$,

and $\underset{0\le i<n}{\forall i} \quad \frac{P_{i+1}}{N_{i+1}} \ge \frac{P_i}{N_i}$

The same arithmetic operations can be applied in the reverse direction to show that

if $\underset{0\le i<n}{\forall i} \quad \frac{P_{i+1}}{N_{i+1}} \ge \frac{P_i}{N_i}$, then $\frac{P_{i+1}}{P_{i+1}+N_{i+1}} \ge \frac{P_i}{P_i+N_i}$ ■

Since, if both P_i and N_i are 0 for some i, the corresponding value v_i can be completely removed from the dataset $\underset{0\le i<n}{\forall i} P_i + N_i > 0$, and this risk function is defined for all values of i.

The risk function $r(v_i)=P_i/(P_i+N_i)$ has another added benefit in that it is simply the probability of the **p** label among all instances of value v_i, which is easily interpretable.

Corollary: For a dataset D with a single categorical feature whose value set is $V = \{v_0, v_1, \ldots, v_n\}$, the risk function defined as $r(v_i)=P_i/(P_i+N_i)$ gives the maximum possible AUC.

Therefore, the REMARC algorithm uses $r(v_i)=P_i/(P_i+N_i)$ as the risk function for categorical features.

4.1.1 The Effect of the Class Label Choice on a Feature's AUC

In order to calculate the P and N values one of the classes should be labeled as **p** and the other class as **n**, but one can question the effect this choice has on the AUC value. It is possible to show that the AUC value of a categorical feature is independent from the choice of class labels by using the value from the Wilcoxon-Mann-Whitney statistics.

In Eq. 6, the AUC formula based on the Wilcoxon-Mann-Whitney statistics is given. P is the number of instances that have the **p** class label and N represents the number of **n**-class-labeled instances. The set D_p represents the **p**-labeled instances and D_n represents the **n**-labeled instances. An element belonging to D_p set, which is D_{pi}, is the ranking of the i^{th} instance, which is labeled **p**. Inversely, an element belonging to D_n set, such as $D_{ni,}$ is the ranking of the i^{th} instance, which is labeled **n**.

$$AUC = \frac{\sum_{i=1}^{P}\sum_{j=1}^{N} f(D_{pi}, D_{ni})}{PN} \tag{6}$$

$$f : \begin{bmatrix} D_{pi} > D_{ni} = 1 \\ D_{pi} < D_{ni} = 0 \\ D_{pi} \equiv D_{ni} = 0.5 \end{bmatrix}$$

The dividend part of the AUC formula in Eq. 6 counts the number of **p**-labeled instances for each element of the D_p set whose ranking is higher than any element of the D_n set. Then, AUC is calculated by dividing this summation by the multiplication of the **p**-labeled and **n**-labeled elements.

The effect of the class label choice on the AUC calculation should be investigated. First of all, it is straightforward that the divisor part of the AUC formula is independent of class choice. Then, assume that the risk score $r_i = \frac{P_i}{P_i + N_i}$ is used on the D dataset and D_p and D_n sets are formed. Let n_i be the number of **n**-labeled instances whose ranking is lower than the i^{th} element of the D_p set and let r_i be the score assigned to this element. When the classes are swapped, the new risk value r'_i is equal to 1- r_i. With this property all instance scores are negated. However, negating scores does not change the relative ranking but inverses it. So, the AUC formula in Eq. 6, which calculates AUC depending on the ranking of the instances, is independent of the class-label decision when the proper risk scoring is used.

4.1.2 An Example Toy Dataset

Assume that a toy training dataset with a single categorical feature is given in Table 2. In order to calculate the AUC value of this particular feature, risk values are needed. The risk values are calculated by the proposed risk function. The sorted version of the dataset according to the risk estimates is given in Table 3. The AUC value of this feature is calculated by using Eq. 2. The *P* value is 7 and the *N* value is 6. The AUC value is $\frac{34.5}{7*6} = 0.82$. In order to calculate this AUC value, for each **p**-labeled instance all **n**-labeled instances whose risk (ranking) is smaller or equal should be counted. When the class labels are swapped the risks are also swapped. The sorted version of the swapped toy dataset is given in Table 4. Since the relative ranking of the instances does not change the new AUC value is also $\frac{34.5}{6*7} = 0.82$.

4.2 Handling Continuous Features

Having found the necessary and sufficient conditions for the risk function for a categorical feature to result in the maximum possible AUC, the next problem is to determine a mechanism for handling the continuous features. An obvious and trivial risk function maps any real value seen in the training set with the class value **p** to 1 and any real value with the class value **n** to 0. This risk function will result in the maximum possible value for AUC, which is 1.0. However, such a risk function will over fit the training data, and will be undefined for unseen values of the feature, which are very likely to be seen in the query instance. So, our first requirement for a risk function for a continuous feature is that it must be defined for all possible values of that continuous feature. A straightforward solution to this requirement is to discretize the continuous feature by grouping all consecutive values with the same class value to a single categorical value; the cut off points can be set to the middle point between feature values of differing class labels. The risk function, then, can be defined using the risk function given in Eq. 5 for categorical features. Although this would result in a risk function that is defined for all values of a continuous function, it would still suffer from the over fitting problem. In order to overcome this problem, the REMARC algorithm makes the following assumption:

Assumption 1: The risk values are either non-increasing or non-decreasing for the increasing values of a continuous feature.

Although there exist some features in real-world domains that do not satisfy this assumption, in the datasets we examined this assumption is satisfied in general.

This assumption is also consistent with the interpretations of the values of continuous features in many real-world applications. For example, in a medical domain, a high value of fasting blood glucose is an indication for a high risk of diabetes. On the other hand, low fasting blood glucose is an indication of a risk for another heath problem, called hypoglycemia.

4.2.1 The MAD Method

The REMARC algorithm requires all features to be categorical. Therefore, the continuous features in a dataset need to be categorized. The aim of a discretization method is to find the proper cut-points in order to categorize a given continuous feature. After the discretization process a continuous feature is treated as a discrete feature whose number of intervals is known on the continuous.

The MAD method is designed to maximize the AUC value by checking the ranking quality of values of a continuous feature. The MAD algorithm given in Kurtcephe and Guvenir (2010) is defined for multi-class datasets. A special version of the MAD method, called MAD2C and defined for two-class problems, is used in REMARC.

In order to measure the ranking quality of a continuous feature, the instances are sorted in ascending order. Sorting is essential for all discretization methods in order to produce unambiguous intervals. After the sorting operation, feature values are used as hypothetical score values and the ROC graph of the feature is drawn. The AUC of the ROC curve shows the overall ranking quality of the continuous feature. In order to obtain the maximum AUC value, only the points on the convex hull must be selected. The minimum number of points that form the convex hull is found by eliminating the points that cause concavities on the graph. In each pass, the MAD method compares the slopes in the order of the creation of

the hypothetical lines, finds the junction points (cut-points) that cause concavities and eliminates them. This process is repeated until there is no concavity on the graph. The points left on the graph are the cut-points, which will be used to discretize the feature.

It has been proven that the MAD method finds the cut-points and the AUC value of the feature independently from the class choice. It is shown that the cut-points found by MAD never separate two consecutive instances of the same class. This is an important property, as it shows that a discretization method works properly. The implementation details, formal proofs and empirical evaluation of MAD can be found in Kurtcephe and Guvenir (2010).

4.2.2 A Toy Dataset Discretization Example

It is possible to visualize the discretization process by using the MAD method. A toy dataset for the discretization is given in Table 5. After the sorting operation, the ROC points are formed. This ROC graph is given in Fig. 5. Since the risk values are either non-increasing or non-decreasing for the increasing values of a continuous feature, two ROC graphs are formed. As can be seen in Fig. 5 one of these graphs is below the diagonal line since the risk is increasing with increasing values of the continuous feature.

The first pass of the MAD method is shown in Fig. 6. All points below or on the diagonal are ignored since they have no positive effect on the maximization of AUC. Then the points causing concavities are eliminated. MAD converged to the convex hull in one pass for this example. The points left on the graphs are the discretization cut-points.

4.3 REMARC Algorithm

The training phase of the REMARC algorithm is given in Fig. 7. In the training phase all continuous features are discretized. In order to discretize continuous features, MAD2C, which is shown on the fifth line of Fig. 7, is used. Risk values are calculated for each value of a given categorical feature (discretized continuous features are included). In this step, the risk function defined in Eq. 5 is used in order to obtain the optimal ranking for categorical features. Then, training

instances are sorted according to the risk values calculated in the previous step. Since the risk function used by REMARC always results in a convex ROC curve, the AUC is always equal to or greater than 0.5. Therefore, the REMARC algorithm learns a weight w_i for a feature f_i as

$$w = \frac{AUC_f - 0.5}{2} \quad (7)$$

The ROC curve of an irrelevant feature is simply a diagonal line from (0,0) to (1,1), with $AUC = 0.5$. The weight function in Eq. 7 assigns 0 to such irrelevant features in order to eliminate them. The risk values and weights of the features are stored for the testing phase.

The testing phase of the REMARC method is straightforward, as for each feature; the risk value corresponding to the value of the feature in the test instance is used. Then the risk of this feature is weighted by its weight, which is calculated in the training phase. The computation of the risk for a query instance q is given in Eq. 8. The maximization of AUC for whole dataset is a challenging problem. Cohen et al. (1998) showed that the problem of finding the ordering that agrees best with a learned preference function is NP-Complete. This weighting mechanism is used as a simple heuristic in order to extend this maximization over the whole feature set.

$$\text{risk}(q) = \frac{\sum_f w_f * P(p \mid q_f)}{\sum_f w_f} \quad (8)$$

$$w_f = \begin{cases} 2(AUC_f - 0.5) & q_f \text{ is known} \\ 0 & q_f \text{ is missing} \end{cases}$$

where $P(p \mid q_f)$ is the probability of q being **p**-labeled, given that the value of feature f in q is q_f, and w_f is the weight of the feature f, calculated by using Eq. 7

Finally, in order to obtain the weighted average, all risk and weight values are summed and final risk is calculated by dividing the cumulative.

The time complexity of the MAD algorithm is given as $O(n^2)$, where n is the number of training instances. After discretizing the numerical features the time complexity of the REMARC algorithm is $O(m*v\lg v+n)$, where m is the number of features and v is the average number of values per feature. As a result, REMARC is bounded by the MAD algorithm's time complexity.

4.4 Interpretation of the REMARC Predictive Model

As mentioned above, the REMARC method does not only provide risk estimation as a single real value, but the predictive model used in order to estimate risk can provide useful information to domain experts. A high weight value indicates that the corresponding feature is a highly effective risk factor in the given domain. Domain experts may choose to ignore features with low weights, potentially reducing the cost of record keeping.

Some of the categorical features are formed by discretizing continuous features. For example, age can be discretized into child, youth, adult and elderly. Assume that the impression of the feature age is investigated on a risky domain, such as medicine. The intervals should be chosen carefully since they can affect a system's predictive performance. The domain experts can provide this information. However, there can be experimental domains where this knowledge is not applicable. The MAD method used in REMARC learns the proper intervals in order to maximize AUC during the training phase. These intervals also report the risks associated with each interval. For example, consider a dataset that contains an age feature and a class label that indicates the presence of a new disease. The MAD method will find the distinct age groups in terms of this disease and the REMARC method will determine the risk for each age group.

The choice of class label during risk estimation has no effect on the feature weights. However, the risk function used by REMARC depends on this choice directly, as shown in Section 4.1, so in order to interpret risk scores correctly one must pay attention to the class label that represents the unwanted situation. Otherwise, risk scores can be misleading.

5 Empirical Evaluations

In order to maximize AUC the theoretical background of the REMARC method is given. In order to support the theoretical background with empirical results two different experiments are conducted. First, REMARC is compared with 26 different machine learning algorithms on an AUC basis. Then, since there can be domains where the predictive models have to trained often, running times of the algorithms are also measured.

The real-life datasets are provided by the UCI machine learning repository (Frank and Asuncion 2010) and are two-class problems. Ten datasets are selected from risk domains such as medicine and finance. The properties of the datasets are given in Table 6.

In order to perform the comparisons, 26 different classification algorithms are selected from the WEKA software package (Hall et al. 2009). Only the algorithms that able to produce continuous output (confidence on the class decision) are selected. As mentioned above, the ROC graphs of algorithms producing continuous output are meaningful. Since REMARC is a non-parametric method, none of the classifiers is optimized for each dataset. All classifiers are used with default settings of WEKA for the sake of fairness. The SVM is taken from the LIBSVM package provided in WEKA (Chang., C-C. & Lin, C-C. 2001).

5.1 Predictive performance

Researchers (Cortes and Mohri 2003; Han and Zhao 2010) reported that some of the algorithms that aim to maximize AUC do not obtain significantly better AUC values than the ones designed to maximize accuracy. Therefore, it is important to show that REMARC can outperform accuracy-maximizing algorithms statistically significantly.

A stratified ten-fold cross validation is employed to calculate AUC values for each datasets. As shown in Table 7, the REMARC method outperformed all algorithms on the average AUC. A paired t-test is used to decide whether the differences on averages are significant. According to the paired t-test on a 95% confidence level (the same level will be used for other t-tests) REMARC

statistically significantly outperforms 15 of the 26 machine learning algorithms on the average AUC. These algorithms include naïve Bayes, decision trees (part, C4.5) and SVM with a RBF kernel. REMARC outperformed the other 11 algorithms, as well, but the differences between the averages for these algorithms are not statistically significant.

One important point should be mentioned about the SVMs. As seen in Table 7, SVM has the worst predictive performance among all the classification algorithms because of the absence of parameter tuning. However, as mentioned before none of the algorithms is tuned for best predictive results.

The classification algorithms such as logistic (multinomial logistic regression model) and classification via regression achieve high AUC values. As mentioned above, these models are highly used in the domain of medicine, and in this work their predictive performance is validated.

The second classifier with the highest AUC was the Adaboost method. Since it is an ensembling algorithm, it uses a base classifier (default DecisionStump in WEKA). We believe that the performance of REMARC can be further improved by using an ensembling algorithm, as then, a statistically significant difference can be obtained.

5.2 Running Time

The REMARC method is designed to be simple, effective and fast. It handles categorical features close to the linear time. MAD requires more time since it uses sorting. Theoretically, REMARC seems fast, but empirical experiments must be conducted to support this claim.

The overall running times of the training phase of 25 different algorithms are calculated. The running times of all algorithms are measured using java virtual machines' CPU time and hundreds of results are averaged (to be objective). The SVM algorithm is not included in the running times section since WEKA uses an outside library for this algorithm. However, it takes seconds for SVM to complete

the training phase, so it is much slower than REMARC. The results of the overall running time for the other algorithms are shown in Table 8.

REMARC outperforms 13 different algorithms significantly according to a paired t-test on a running-time basis. These outperformed methods are shown by the ↑ symbol on Table 8. Five algorithms outperformed REMARC statistically significantly. These algorithms are shown with a ↓ symbol. The differences between the other six methods on the table and REMARC are not significant.

6 Conclusions and Future Work

In this paper, we gave a discussion of risk in real-life domains. Different risk domains are analyzed and some of the methods used specially in these domains are given. Then we showed how the risk estimation problem can be modeled as a two-class classification problem in machine learning.

We argued the effectiveness of a method that maximizes accuracy, for a risk estimation method. We proposed an AUC-based method instead of accuracy and presented important features of AUC, such as insensitivity to class distribution and error cost, as being statistically more consistent and discriminating. Then, we summarized the different methods proposed so far designed to maximize AUC.

Aiming to maximize AUC, we proposed a risk estimation method called REMARC. We have shown that for a categorical feature there is only one ordering that gives the maximum AUC. Then we showed the sufficient and necessary condition for a risk function to achieve this ordering. As a result, we proposed a risk function that finds the maximum possible AUC on one categorical feature. Aiming to maximize AUC, we handled the continuous features using the MAD method, as it can discretize a continuous variable. Then we used these AUC values as weights in computing the risk scores as weighted averages of feature value risks. With this simple heuristic we averaged all feature risk values in order to achieve maximum AUC over the whole dataset.

We present the characteristics of the REMARC risk prediction model and how it should be interpreted. REMARC's prediction model is easy to understand and interpret by domain experts.

After supporting the theoretical background, we compared REMARC with 26 different algorithms. According to empirical evaluation, REMARC significantly outperformed 15 algorithms on an AUC basis and 13 algorithms on a time basis. It also outperformed all algorithms on the average AUC and 17 of them on an average time basis.

As a future work, REMARC can be compared with other risk methods and methods designed to maximize AUC. In order to improve the performance of REMARC, ensembling methods can be employed.

To conclude, a fast and highly predictive risk estimation method is proposed in this paper. A simple yet effective predictive model, it is understandable by domain experts and will be useful for the machine learning community.

References

Ataman, K., Street, W. N., Zhang, Y. (2006). Learning to rank by maximizing auc with linear programming. In *IEEE International Joint Conference on Neural Networks* (IJCNN), 123–129.

Biagioli, B., Scolletta, S., Cevenini, G., Barbini, E., et al. (2006). A multivariate bayesian model for assessing morbidity after coronary artery surgery. *Critical Care*, doi:10.1186/cc4951.

Boström, H. (2005). Maximizing the area under the ROC curve using incremental reduced error pruning, In: *ICML 2005 Workshop on ROC Analysis in Machine Learning.*

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

Bradley, B.O., Taqqu, M.S. (2003). Handbook of Heavy-Tailed Distributions in Finance. In: Rachev, S.T. (Ed.), *Financial risk and heavy tails* (pp. 35–103). Rotterdam: Elsevier.

Calders, T., & Jaroszewicz, S. (2007). Efficient AUC Optimization for Classification. In *Knowledge Discovery in Databases*: PKDD, 42-53.

Cameron, I. T., & Raman, R. (2005). Process Systems Risk Management, Vol. 6. *Risk- Estimation, Presentation and Perception* (pp. 37-65) San Diego: Elsevier Science & Technology Books.

Chang., C.-C. & Lin, C.-C. (2001). LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

Cohen, W. W., Schapire, R. E., Singer, Y. (1998). Learning to order things. In *Advances in Neural Information Processing Systems,*10, 243-270.

Colombet, I., Ruelland, A., Chatellier, G., Gueyffier F., et al. (2000). Models to predict cardiovascular risk: comparison of cart, multilayer perceptron and logistic regression. *Proccedings of American Medical Informatics Association Symposium*, 156-160.

Conroy, R. M., Pyörälä, K., Fitzgerald, A. P. et al. (2003) Estimation of ten-year risk of fatal cardiovascular disease in Europe: the SCORE project. *European Heart Journal*, 11, 987–1003.

Cortes, C., & Mohri, M. (2003). AUC optimization vs. error rate minimization. *Neural Information Processing Systems (NIPS)*, 16, 313-320.

Cox, D.R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society*, 34(B), 187-220.

D'Agostino, R.B., Ramachandran, S.V., Pencina, J., et al. (2008). General cardiovascular risk profile for use in primary care: the Framingham Heart Study. *Circulation*, 17, 743–753.

Domingos, P. (1999). MetaCost: A general method for making classifiers cost-sensitive. In: *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 155–164.

Dowd, K., & Blake, D. (2006). After VaR: The Theory, Estimation, and Insurance Applications of Quantile-Based Risk Measures. *The Journal of Risk and Insurance*, 73(2), 193-229

Fawcett, T., & Provost, F. (1997). Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1, 291-316.

Fawcett, T. (2001). Using Rule Sets to Maximize ROC Performance. *In: Proceedings of the IEEE International Conference on Data Mining* (ICDM-2001), 131–138.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27, 861–874.

Ferrari, D., & Paterlini, S. (2007). The maximum Lq-likelihood method: An application to extreme quantile estimation in finance. *Methodology and Computing in Applied Probability*, 11, 3–19.

Ferri, C., Flach, P., Hernandez, J. (2002). Learning decision trees using the area under the ROC curve. In C. Sammut, & A. Hoffmann (Eds.), *Proceedings of the 19th International Conference on Machine Learning* (ICML-02), 139–146.

Flach, P. & Wu, S. (2003). Repairing concavities in ROC curves. In *Procedings UK Workshop on Computational Intelligence*, 38–44.

Frank, A. & Asuncion, A. (2010). UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science. http://archive.ics.uci.edu/ml

Freund, Y., Iyer, R., Schapire, R. E., and Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research* 4, 933-969.

Galindo, J. & Tamayo, P. (2000). Credit Risk Assessment Using Statistical and Machine Learning: Basic Methodology and Risk Modeling Applications. *Computational Economics*, doi:10.1023/A:1008699112516.

Gamberger, D., Krstacic, G., & Smuc, T. (2000). Medical expert evaluation of machine learning results for a coronary heart disease database. *Lecture Notes on Computer Science*, 1933, 159-168.

Giddens A. (1999). Risk and Responsibility. *Modern Law Review*, doi:10.1111/1468-2230.00188

Green, D.M., & Swets, J.A. (1966) *Signal Detection Theory and Psychophysics*. Wiley, New York.

Hall, M. et al. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, Volume 11, Issue 1.

Han, G., & Zhao, C. (2010). AUC maximization linear classifier based on active learning and its application. *Neurocomputing*, doi:10.1016/j.neucom.2010.01.001

Hand, D.J., & Till, R.J. (2001) A simple generalization of the area under the roc curve to multiple class classification problems, *Machine Learning*, 45, 171–186.

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

Hannan, E., Wu, C., Bennett, E., Carlson, R., Culliford, A., et al. (2006). Risk stratification of in-hospital mortality for coronary artery bypass graft surgery. *Journal of the American Collage of Cardiology*, doi:10.1016/j.jacc.2005.10.057.

Hansson, S. O. (2007). Risk. Stanford Encyclopedia of Philosophy. Accessed 30 May. 2010.

Herschtal, A., & Raskutti, B. (2004). Optimising the area under the ROC curve using gradient descent. In: *Proccedings of International Conference on Machine Learning*, 49–56.

Huang, A. Y. (2010). An optimization process in Value-at-Risk estimation. *Review of Financial Economics*, doi:10.1016/j.rfe.2010.03.001, In press.

Huang, J, & Ling, C. X. (2005). Using AUC and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 17(3), 299–310.

Kim, K-J (2003). Financial time series forecasting using support vector machines. *Neurocomputing*, doi:10.1016/S0925-2312(03)00372-2.

Krzanowski, W.J., & Hand, D.J., (2009). *ROC curves for continuous data*, CRC Press, Taylor & Francis Group.

Kurtcephe M., & Guvenir H. A. (2010).A Discretization Method based on Maximizing the Area Under ROC Curve. Technical Report. Bilkent University. http://www.cs.bilkent.edu.tr/tech-reports/2010/BU-CE-1001.pdf. Accessed 14 June 2010

Ling, C.L., & Zhang, H. (2002). Toward Bayesian classifiers with accurate probabilities. *Advances in Knowledge Discovery and Data Mining*, LNAI 2336, 123-134.

Mac Namee, B., Cunningham, P., Byrne, S., & Corrigan, O. (2002). The problem of bias in training data in regression problems in medical decision support. *Artificial Intelligence in Medicine*, 24, 51-70.

Maloof, M.A. (2003). Learning when data sets are imbalanced and when costs are unequal and unknown. In: *Procedings International Conference on Machine Learning. Workshop on Learning from Imbalanced Data Sets II*.

Marrocco, C., Molinara, M., Tortorella, F. (2006). Exploiting AUC for optimal linear combinations of dichotomizers. *Pattern Recognition Letters* 27(8), 900–907.

Marrocco, C., Duin, R. P.W., and Tortorella, F. (2008). Maximizing the area under the ROC curve by pairwise feature combination. *Pattern Recognition*, 41, 1961–1974.

Min, J. H., & Lee Y-C (2005). Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert Systems with Applications*, doi:10.1016/j.eswa.2004.12.008

Mozer, M. C., Dodier, R., Colagrosso, M. D., Guerra-Salcedo, C., Wolniewicz, R. (2002) Prodding the ROC curve: Constrained optimization of classifier performance. In: *Advances in Neural Information Processing Systems*, 14, 1409–1415.

Pepe, M.S. (2003). *The statistical evaluation of medical tests for classification and prediction*. Oxford, New York.

Prati, R., & Flach, P. (2004). Roccer: A roc convex hull rule learning algorithm. In *Proccedings of the ECML/PKDD Workshop on Advances in Inductive Rule Learning*, 144-153.

Provost, F. & Fawcett, T. (1997). Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *Knowledge Discovery and Data Mining*, 43–48.

Provost, F., Fawcett, T., & Kohavi, R. (1998). The Case Against Accuracy Estimation for Comparing Induction Algorithms. In: J. Shavlik (Ed.). *Proceedings of the Fifteenth International Conference on Machine Learning*, 445–453.

Provost, F., & Fawcett, T. (2001). Robust Classification for Imprecise Environments. *Machine Learning*, 42(3), 203–231.

Rakotomamonjy, A. (2004). Optimizing area under roc curve with svms. In *Workshop on ROC Analysis in Artificial Intelligence*, 71-80.

Roques, F., Michel, P., Goldstone, A., & Nashef, S. (2003). The logistic euroscore. *European Heart Journal*, doi:10.1016/S0195-668X(02)00799-6.

Rosset, S. (2004). Model selection via the AUC. In: *Procedings of International Conference on Machine Learning*, 69, 89-96.

Sebag, M., Aze, J., Lucas, N. (2004). ROC-based evolutionary learning: Application to medical data mining. *Lecture Notes in Computer Sciences*, doi:10.1007/b100704.

Shishkin, V. M., & Savkov S. V. (2009). The Method of Interval Estimation in Risk- Analysis System. *Proceedings of the 2nd international conference on Security of information and networks*, doi:10.1145/1626195.1626199.

Spackman, K.A. (1989). Signal detection theory: Valuable tools for evaluating inductive learning. In *Proceedings of the Sixth International Workshop on Machine Learning*, 160–163.

Stoyanov, Stoyan, V., Racheva-Iotova B., Rachev S. T., Fabozzi, F. J. (2008). Stochastic Models for Risk Estimation in Volatile Markets: a Survey. *Annals of Operations Research*, doi: 10.1007/s10479-008-0468-1.

Tax, D.J.M., Duin, R.P.W., Arzhaeva, Y. (2006). Linear model combining by optimizing the area under the roc curve. *Proceedings of the 18th IEEE International Conference on Pattern Recognition*, 119–122.

Toh, K.A., Kim, J., Lee, S. (2008). Maximizing area under ROC curve for biometric scores fusion, *Pattern Recognition*, 41, 3373–3392.

Yan, L., Dodier, R., Mozer, M. C., Wolniewicz R. (2003) Optimizing Classifier Performance Via the Wilcoxon-Mann-Whitney Statistics. In: *Proceedings of the Twentieth International Conference on Machine Learning*. 848–855.

Zweig, M.H., & Campbell, G. (1993). Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. *Clinical chemistry*, 39(8), 561–577.

		Actual Class	
		p	**n**
Predicted Class	**p**	TP	FP
	n	FN	TN
Column Totals:		**P**	**N**

Fig. 1. Structure of a confusion matrix.

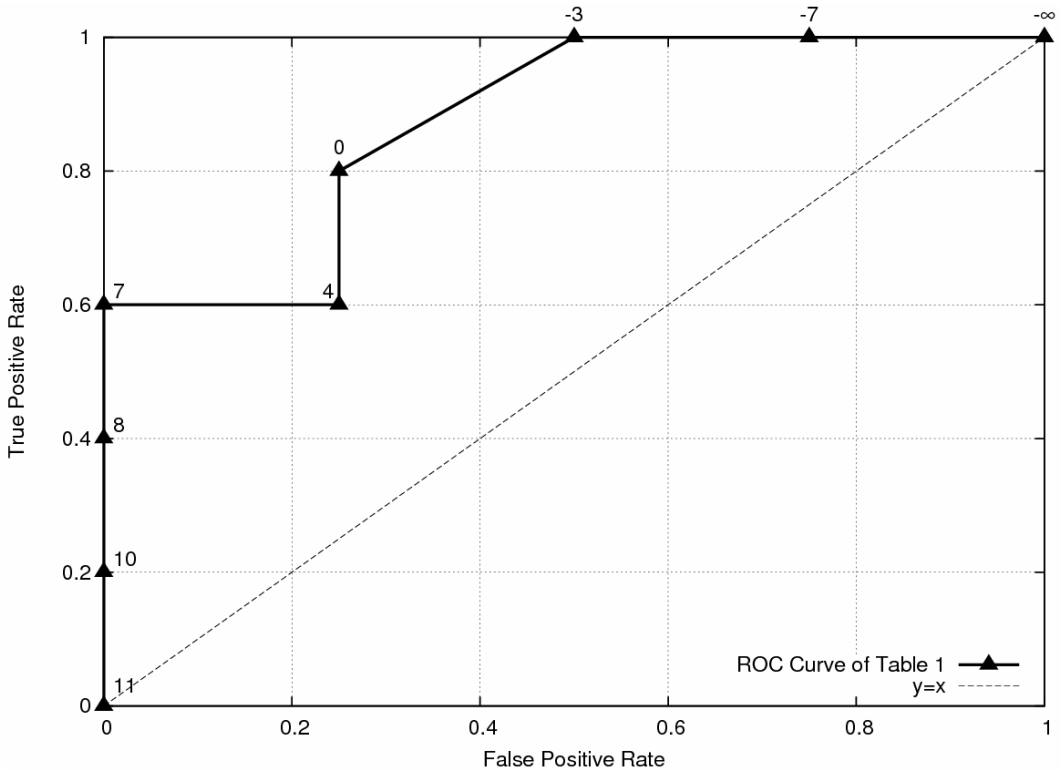

Fig. 2. ROC graph of the given toy dataset in Table 1 including the *y=x* line in order to show random performance.

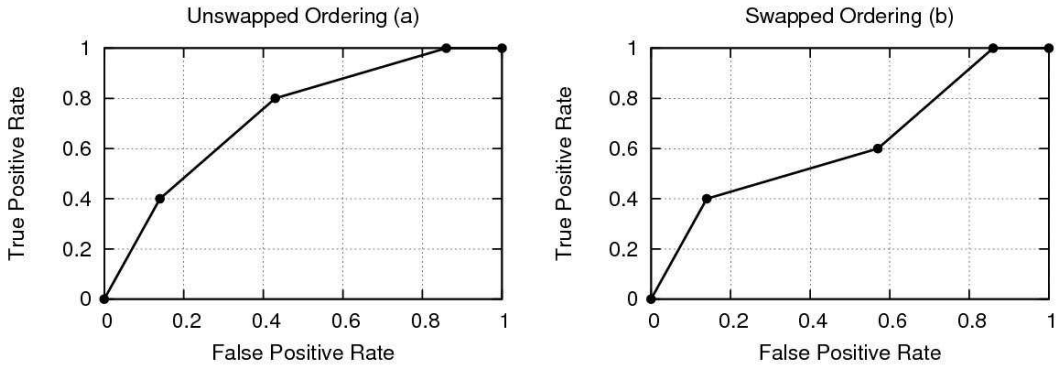

Fig. 3. Effect of swapping the risk values of two feature values

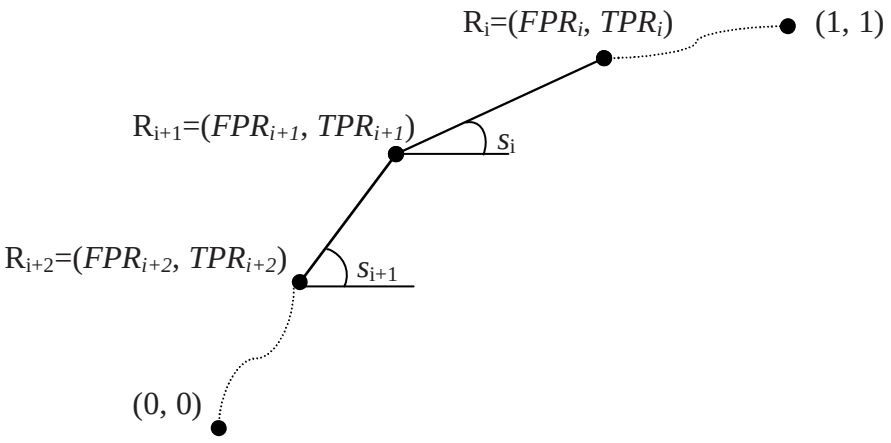

Fig. 4. Relation between the slopes of two consecutive line segments in a convex ROC curve

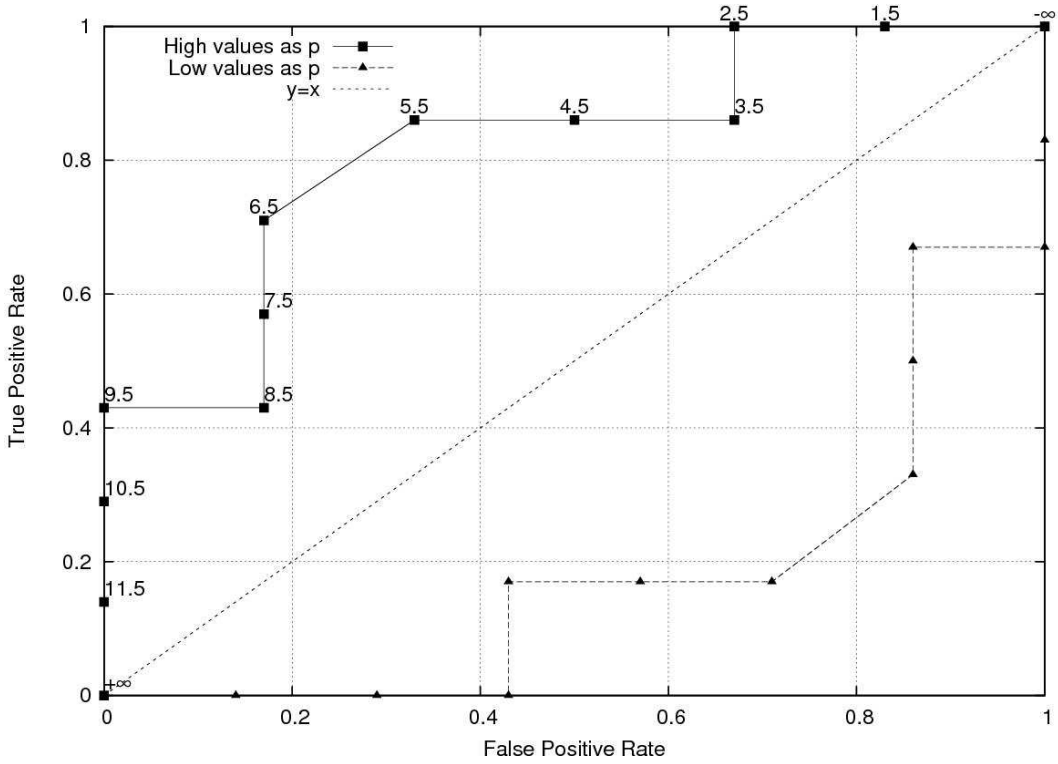

Fig. 5. Visualization of the ROC points in a two-class discretization

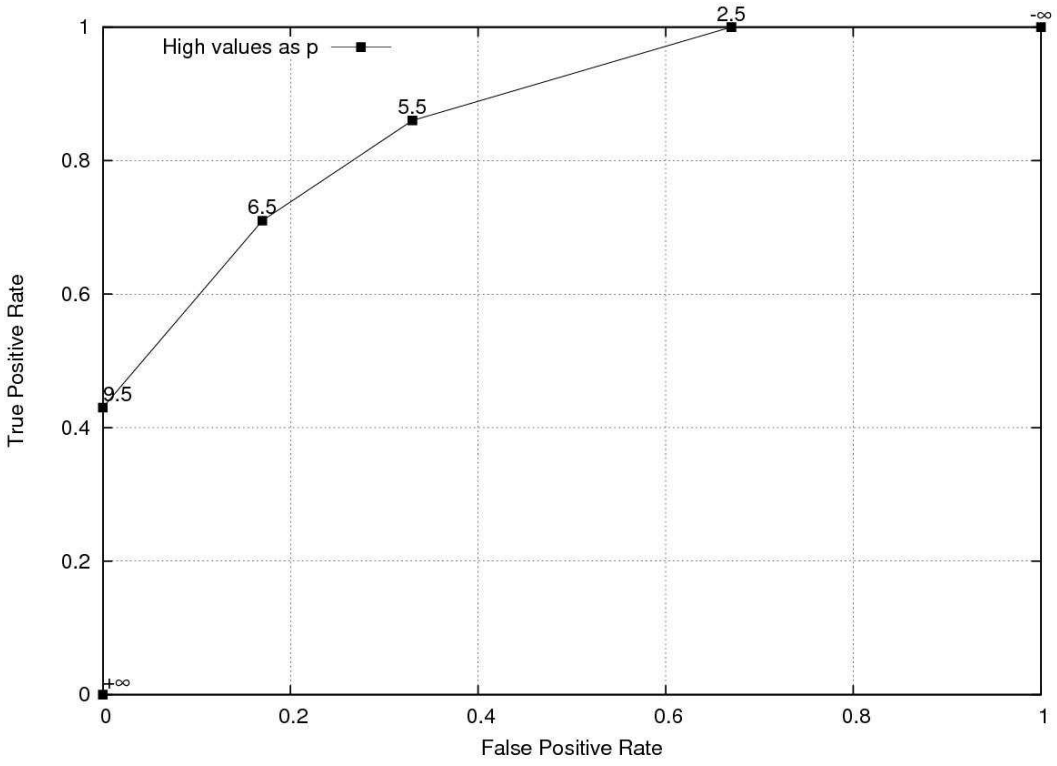

Fig. 6. Final cut-points after the first pass of convex hull algorithm.

```
1 :REMARCTrain (trainSet[M][N]) // Includes M features and N train
instances
2 :   Begin
```

```
3 :      for i=0 to M-1
4 :         if(isContinuous(trainSet[i]));
5 :            cutPoints=MAD2C(trainSet[i][0..N-1]);
6 :            numericalValuesToCatVal(cutPoints,trainSet[i]);
7 :         risks[i]<-computeCategoricalRisk(trainSet[i][0..N-1]);
8 :         sortInstancesByRisk(trainSet[i][0..N-1]);
9 :         aucValues[i]<-computeAUC(trainSet[i][0..N-1]);
10:         featureWeights([i]=aucValues[i]-0.5)*2;
11:      end
12:   end
```

Fig. 7. Algorithm of the REMARC method's training phase

```
1 :REMARCTest (testInstance[M][1])
2 :   Begin
3 :      for i=0 to M-1
4 :         oneFeatureRisk= risks[i][testInstace[i][0]];
5 :         totalRisk+= oneFeatureRisk * featureWeights[i];
6 :         totalWeight+= featureWeights[i];
7 :      end
8 :      return totalRisk/totalWeight;
9:   end
```

Fig. 8. Testing phase algorithm of the REMARC method.

Table 1. A Toy dataset given with hypothetical scores.

Class Label	n	n	n	p	p	n	p	p	p
Score	-7	-3	0	0	4	7	8	10	11

Table 2. Toy training dataset with one categorical feature.

Class Label	n	n	p	n	n	p	n	p	p	n	p	p	p
Feature Value	a	a	a	a	b	b	b	c	c	c	d	d	d

Table 3. Training datasets risk values are calculated and instances are sorted in ascending order.

Risk	0.25	0.25	0.25	0.25	0.33	0.33	0.33	0.66	0.66	0.66	1.00	1.00	1.00
Class Label	n	n	p	n	n	p	n	p	p	n	p	p	p
Feature Value	a	a	a	a	b	b	b	c	c	c	d	d	d

Table 4. Negated version of the training dataset. The risk values are calculated again and instances are sorted in ascending order.

Risk	0.0	0.0	0.0	0.33	0.33	0.33	0.66	0.66	0.66	0.75	0.75	0.75	0.75
Class Label	n	n	n	n	n	p	p	n	p	p	p	n	p
Feature Value	d	d	d	c	c	c	b	b	b	a	a	a	a

Table 5. A toy dataset for visualizing MAD in two-class problems. The name of the attribute to be discretized is F1.

Class Value	n	n	p	n	n	p	n	p	p	n	p	p	p
F1	1	2	3	4	5	6	6	7	8	9	10	11	12

Table 6. Properties of the datasets used in the empirical evalutations of the REMARC algorithm.

Dataset Name	# Instances	# Continuous Attributes	#Categorical Attributes	# Dataset Abbreviations
Australian	690	6	8	A
Bupa	345	6	0	B
Crx	653	9	6	C
Heart (Statlog)	270	7	6	H1
Hypothyroid	3164	7	18	H2
Mammographic Masses	961	1	5	M
Pima-Diabetes	768	8	0	P
Sick-Euthyroid	3163	7	18	S1
SPECTF	267	44	0	S2
Wisconsin-Breast	569	30	0	W

Table 7. The comparison of the predictive performance of REMARC algorithm with other algorithms on AUC metric. 10 datasets are used during evaluation. Algorithms marked with ++ are outperformed by REMARC method with a statistically significant difference Algorithms marked with + are outperformed by REMARC on average with no significant difference. (Higher results better)

Algorithms/Datasets	A	B	C	H1	H2	M	P	S1	S2	W	**Average**
REMARC	0,923	0,659	0,931	0,904	0,986	0,901	0,827	0,942	0,857	0,986	**0,892**
BayesNet+	0,920	0,540	0,928	0,901	0,989	0,899	0,818	0,959	0,825	0,986	0,876
NaiveBayes++	0,895	0,641	0,900	0,897	0,977	0,895	0,816	0,920	0,850	0,980	0,877
AODE+	0,928	0,540	0,930	0,904	0,989	0,900	0,823	0,963	0,820	0,988	0,879
Logistic+	0,912	0,714	0,915	0,900	0,970	0,893	0,831	0,956	0,801	0,972	0,886
RBFNetwork++	0,732	0,509	0,787	0,835	0,581	0,786	0,642	0,676	0,641	0,755	0,694
IBk++	0,801	0,634	0,798	0,743	0,766	0,799	0,648	0,752	0,592	0,947	0,748
LWL++	0,911	0,643	0,909	0,839	0,955	0,886	0,775	0,942	0,674	0,948	0,848
AdaBoostM1+	0,922	0,737	0,926	0,888	0,990	0,895	0,804	0,966	0,801	0,985	0,891
Att.Sel.Classifier++	0,869	0,584	0,875	0,801	0,952	0,867	0,786	0,914	0,624	0,938	0,821
Bagging+	0,918	0,755	0,910	0,872	0,980	0,888	0,822	0,972	0,795	0,977	0,889
Class.ViaRegr.+	0,918	0,727	0,918	0,882	0,990	0,896	0,827	0,986	0,763	0,989	0,890
END++	0,865	0,648	0,877	0,777	0,940	0,868	0,758	0,939	0,593	0,939	0,821
FilteredClassifier++	0,899	0,540	0,893	0,836	0,958	0,863	0,794	0,949	0,683	0,939	0,835
MultiBoostAB+	0,908	0,673	0,908	0,865	0,988	0,886	0,790	0,955	0,709	0,981	0,866
MultiC.Classifier+	0,912	0,714	0,915	0,900	0,970	0,893	0,831	0,956	0,801	0,972	0,886
OrdinalC.Classifier++	0,865	0,648	0,877	0,777	0,940	0,868	0,758	0,939	0,593	0,939	0,821
ThresholdSelector+	0,904	0,699	0,916	0,898	0,969	0,892	0,826	0,956	0,686	0,969	0,871
VFI++	0,913	0,562	0,910	0,871	0,782	0,836	0,550	0,755	0,853	0,946	0,798
DecisionTable+	0,917	0,574	0,910	0,883	0,989	0,876	0,801	0,971	0,678	0,972	0,857
PART++	0,867	0,645	0,853	0,785	0,966	0,882	0,778	0,954	0,652	0,937	0,832
ADTree+	0,917	0,705	0,925	0,880	0,988	0,887	0,802	0,979	0,803	0,984	0,887
DecisionStump++	0,833	0,572	0,848	0,688	0,951	0,788	0,696	0,936	0,623	0,886	0,782
FT++	0,898	0,721	0,853	0,824	0,943	0,874	0,751	0,907	0,752	0,984	0,851
J48 (C4.5) ++	0,865	0,648	0,877	0,777	0,940	0,868	0,758	0,939	0,593	0,939	0,821
SVM-RBF++	0.628	0.609	0.602	0.509	0.952	0.872	0.518	0.735	0.466	0.760	0.655
REPTree++	0,879	0,666	0,871	0,824	0,963	0,846	0,768	0,957	0,631	0,924	0,833

Table 8. The comparison of the average running time performance of REMARC algorithm with other algorithms (in ms) . 10 datasets are used during evaluation. Algorithms marked with ++ symbol are outperformed by REMARC method on running time basis with a statistically significant difference. Algorithms marked with -- symbol outperformed REMARC method on running time basis with a statistically significant difference. + marked algorithms are outperformed by REMARC on average and – marked algorithms outperform REMARC on average with no significant difference. (Lower results better)

Algorithms/Datasets	A	B	C	H1	H2	M	P	S1	S2	W	**Average**
REMARC	106	37	94	49	1131	128	164	1117	191	470	**349**
BayesNet--	49	17	52	28	425	54	60	393	81	244	140
NaiveBayes--	30	11	29	18	164	30	37	154	61	102	64
AODE--	53	16	52	27	352	40	56	350	113	280	134
Logistic++	1014	74	1215	124	3720	261	257	3667	472	935	1174
RBFNetwork++	404	136	348	148	1057	367	280	1149	712	620	522
Ibk+	172	33	169	35	7365	233	153	7465	124	222	1597
LWL+	2094	376	1940	412	52652	2360	2652	52887	1414	6919	12370
AdaBoostM1++	302	132	296	156	1584	302	394	1579	475	1135	635
Att.Sel.Classifier+	173	31	159	99	761	201	162	867	419	496	337
Bagging++	740	295	727	295	4484	636	961	6904	1070	1718	1783
Class.ViaRegr. ++	5340	1355	5573	1143	9943	3912	3593	15598	2218	2662	5134
END+	236	119	218	126	880	219	254	1563	333	460	441
FilteredClassifier-	119	16	107	48	462	86	116	858	155	265	223
MultiBoostAB++	318	133	297	164	1614	317	388	1622	477	1140	647
MultiC.Classifier++	991	79	1175	125	3698	239	226	3656	473	963	1163
OrdinalC.Classifier	162	79	150	83	681	177	197	1380	255	378	354
ThresholdSelector++	1695	137	2032	227	6442	429	387	6346	1118	2028	2084
VFI--	16	5	17	9	104	12	14	111	23	31	34
DecisionTable++	1582	156	1699	434	10824	411	635	11498	1183	2526	3095
PART++	416	102	511	193	865	230	248	2023	684	473	574
ADTree++	689	276	645	469	3149	579	973	3536	1562	2717	1459
DecisionStump--	24	9	23	11	113	17	30	116	43	105	49
FT++	4705	847	4879	927	14834	2856	2230	30959	1796	2324	6636
J48 (C4.5)-	160	75	153	81	635	147	181	1283	256	378	335
REPTree++	77	33	80	33	395	105	136	569	124	185	174