# Evaluation of classifiers for an uneven class distribution problem[1]

Sophia Daskalaki[1], Ioannis Kopanas[2], Nikolaos Avouris[2],

[1]Dept. of Engineering Sciences,
[2]Human-Computer Interaction Group, Dept. of E&CE,
University of Patras, Rio Patras GR-26500, Greece
sdask@upatras.gr , ikopanas@ee.upatras.gr, n.avouris@ee.upatras.gr

*Abstract*
*Classification problems with uneven class distributions present several difficulties during the training as well as during the evaluation process of classifiers. A classification problem with such characteristics has resulted from a data-mining project where the objective was to predict customer insolvency. Using the dataset from the customer insolvency problem we study several alternative methodologies which have been reported to better suit the specific characteristics of this type of problems. Three different but equally important directions are examined; (a) the performance measures that should be used for problems in this domain, (b) the class distributions that should be used for the training data sets, (c) the classification algorithms to be used. The final evaluation of the resulting classifiers is based on a study of the economic impact of classification results. This study concludes to a framework that provides the "best" classifiers, identifies the performance measures that should be used as the decision criterion and suggests the "best" class distribution based on the value of the relative gain from correct classification in the positive class. This framework has been applied in the customer insolvency problem, but it is claimed that it can be applied to many similar problems with uneven class distributions that almost always require a multi-objective evaluation proces.*

Keywords: data mining, classification, imbalanced class distributions, voting algorithms, Cost-sensitive learning

## 1. Introduction

Classification problems with uneven distributions between classes are considered difficult problems, however they are not uncommon in data mining projects. Problems like the detection of oil spills in satellite radar images (Kubat et al., 1998), fraud detection in mobile communications (Fawcett and Provost, 1997) or credit cards (Chan et al., 1999), prediction of failures in manufacturing processes (Riddle et al., 1994), and diagnosis of rare diseases (Laurikkala, 2001), are typical examples of such problems.

In most of these projects there is a vast amount of data for data mining but only a handful of the cases of interest, a reality that makes effective classification of rare events a very difficult task. The difficulties concern the induction process, where classification algorithms produce classifiers, but also the evaluation process, where the induced classifiers are compared for their performance in order to choose the "best" or "a set of best" ones for further use. In fact, when the problem is a difficult one, constructing the classifier with the best performance involves several decisions; therefore, it is worth constructing a methodology to support the generation of classifiers and a framework for the evaluation process.

First, it is important to choose performance measures, suitable for the problem and the available data and robust to changes that may occur in the future. The performance measure mostly used for classification problems is the overall accuracy or equivalently the total error. This works fine for the majority of the problems, however highly imbalanced class distribution or specific business objectives may require high accuracy for the minority as well as

[1] A draft of a manuscript accepted for publication in Applied Artificial Intelligence, 2006.

for the majority class, which is not guaranteed from the overall accuracy. Conversely, several prediction problems require high precision, achieved when the correctly classified cases in any given class is large compared to the incorrectly classified cases in the other class. Other performance measures may easily be invented through linear or non linear combinations of those mentioned. Because the problem of choosing a classifier is in fact a multi-objective one, in the problem presented we study a number of interesting performance measures and drive conclusions from their behavior in our dataset.

Second, the classification algorithms should be trained using data that reflect the characteristics of all classes, the large and the small classes. Previous studies (Weiss and Provost, 2001; Chan and Stolfo, 1998b) have shown that training with the "natural" distribution does not always give the best classifier; therefore, new datasets should be constructed to aid the training process. Especially when the minority cases are rare, a training set "enriched" with minority examples is needed in order for the induction algorithms to create classifiers with strong predictive capabilities for both classes. This can be accomplished either by eliminating cases in the majority class until the desired distribution is achieved (undersampling) or by duplicating cases in the minority class until the desired distribution is achieved (oversampling). As discussed later, in this paper we adopt undersampling to create new datasets for training with artificial class distribution.

Furthermore, in real problems the performance measures should adopt the business objectives, which are usually posed by the users. Driving the data-mining procedure with the help of business objectives is of utmost importance for the results to be potentially useful. While this is common knowledge, there are a few different ways of achieving it. One may incorporate the performance measures in the algorithmic procedure and drive the induction of the classifier by fine-tuning the parameters of the algorithms (Gur Ali and Wallace, 1997). Alternatively, one may experiment with different class distributions (Weiss and Provost, 2003), which is equivalent to altering the costs for the error rates (Chan and Stolfo, 1998b), and choose the most suitable classifier for the problem. These alternative ways are equivalent (Elkan, 2001) and in this paper we also experiment with different class distributions for the learning process. In addition, we study the effect of class distribution on the different performance measures. It turns out that this study is inconclusive because different performance measures point to different class distribution.

Third, the selection of a classification algorithm should depend on a comparison of performances for alternative algorithms according to selected measures and class distributions. Despite all efforts, if none of the chosen algorithms prevails over the others, meeting the business objectives may be achieved with the combination of different classification algorithms. While the fine-tuning of a classification algorithm may result to a reasonably good performance, combining the outcomes of different algorithms into a voting scheme may in fact lead to the desired result much faster. In this paper, we study three simple voting schemes for classifiers induced from different algorithms, only after going through an extensive experimentation with six classification algorithms, several class distributions and performance measures.

Finally, a study on the economic impact of the classification results for the interested company leads to an evaluation framework for all involved classifiers. For this study apart from the classification results, a cost/benefit matrix is introduced based on the business criteria in our problem. It is concluded that different classifiers will be indicated as "best", depending on the relative gain from correct classification in the minority class compared to the cost for incorrect classification in the majority class. The same is true for the "optimal" class distribution and for this matter the larger the relative gain, the more balanced distribution should be chosen. The study is performed with real data from a telecommunications company with a great interest in reducing customer insolvency without risking its relations with the rest of the customers.

The paper is structured as follows. Section 2 gives a brief description of the customer insolvency prediction problem, a study that motivated the methodology for the evaluation of classifiers presented here. Section 3 reviews the solutions that other researchers have proposed for problems with uneven distribution between classes. Section 4 surveys the most common performance evaluation measures and focuses on those that are most useful to problems with highly imbalanced datasets. In section 5, a number of classifiers induced from six different classification algorithms and trained with different class distributions are compared using the chosen performance measures for our case study. Additional performance measures are introduced in this section that enhances the list of the examined measures. In section 6 we attempt the combination of the produced classifiers in simple voting schemes. We study the performance of the voting schemes compared to single algorithms and draw conclusions. A perspective that takes into account the economic aspects of a classification scheme for the company is presented in section 7 to conclude with specific rules for some of the measures and a framework for the evaluation of the resulting classifiers. Lastly, in section 8 we summarize our observations and conclusions.

## 2. The customer insolvency prediction problem

Telecommunication companies, just like other service providers, regularly face the problem known as "customer insolvency" leading to "uncollectible debts" for the affected company. This problem appears because companies

offer services to their customers trusting them that they will pay their dues at the end of each billing period. In our case study, we characterize as *insolvent customers* those that refuse to pay their bill even six months after the expiration of the deadline for payment, regardless of the receipt of several reminders. It is only then that the company characterizes the amount owed as "uncollectible" and the contract with the customer "void". The telephone connection of the customer is usually nullified one or two weeks after the expiration of the payment period, however until that time the insolvent customer has plenty of time to take advantage of the telephone services for which he/she has no intention to pay. Given the harsh competition in the telecommunications world, companies cannot afford the cost of insolvency, thus timely prediction of such incidents attains high priority. Previous work on the same problem but with a different setting, characteristics and assumptions has been reported in (Ezawa et al. 1996a) where an approach based on Bayesian Networks was adopted.

In an effort to develop a decision support system for managing insolvency, we set up an experiment, which involved application of data mining techniques in data provided by a major telecommunication company. The goal was to confirm the hypothesis that prediction of customer insolvency is possible by studying patterns in the usage of the telephone service, as well as in the customer's attitude towards previous bills. While details for this project and the data analysis that was performed may be found in (Daskalaki et al., 2003), we briefly outline those points that are important for the work presented in this paper.

The data studied for this project concerned approximately 100,000 customers from three geographic areas of Greece. For all customers in the dataset, information from the Call Detail Records (CDRs) was collected for a period of approximately 17 months. For the same period, the customers' previous financial transactions with the company were also studied using data from the billing information system. The collected raw data were processed, during the early stages of the knowledge discovery procedure, and finally produced a dataset that was used for experimentation with machine-learning algorithms. In this dataset, for each customer the data included: (i) two attributes for the customer's profile (static information), (ii) sixty-six attributes for the usage of the phone over fifteen consecutive two-week periods, and (iii) four attributes for the financial transactions of the customer (payment and agreements for payment with installments). Therefore, seventy-seven attributes were collected in total for each customer in the dataset.

A study with the collected data concluded that the problem of customer insolvency prediction is a difficult one, however when the business objectives are used as the basic criterion for choosing classifiers, prediction is still possible at some acceptable level of accuracy. The characteristics that make this problem difficult are the following:

- Very uneven distributions for the two classes (solvent, insolvent) of customers
- Small number of cases of insolvent customers (minority class) in the dataset
- Different and often unknown misclassification costs for the two classes.

More specifically, the dataset that resulted after processing the raw data carried 196 cases of insolvent and 28,024 of solvent customers (i.e. a proportion of 1:142), while this ratio can vary with time and geographic area. As for the misclassification costs, it was known that false alarms (falsely predicting a solvent customer as insolvent) were highly undesirable, because the company did not want to put at risk its relations with good customers. In competitive business sectors, companies are continuously making efforts to improve customer relationships, so hassling good customers due to false alarms is not considered good business practice. Therefore, "*while it is important to predict as many insolvent-to-be customers as possible, it is preferable to miss a portion of "bad" customers than to hassle a large number of "good" customers and possibly loose some of them to the competition*". This information, which originated from the decision makers in the company, defined the *business objectives* for our study. Even though it would had been of some help, the objectives were not quantified further, so our effort was to match the business objectives with the performance measures used for the evaluation process.

### 3. Solutions for problems with uneven class distributions

In order to tackle problems with imbalanced datasets along with some other difficult problems, a number of alternative approaches have been suggested, as discussed in this section. These involve the use of performance measures other than the traditional ones, altering the class distribution in the training set or the cost of misclassification and evaluating different algorithms or combining different classifiers. This whole process can be performed as part of a study that aims to prove or disprove feasibility of predicting the rare events. On the other hand, it is important to find the conditions under which such a prediction is useful for the interested organization.

In Knowledge Discovery projects where the induction of a classifier is required for prediction purposes, classification algorithms are routinely trained with a dataset and then tested with a different one in order to evaluate its performance. The performance measure, most widely used in this evaluation procedure, is the *average accuracy rate* (percentage of cases correctly classified) or equivalently the overall error rate (percent of cases incorrectly

classified). For problems with high imbalance in class distributions, however, the average accuracy rate is inappropriate, as well as for cases with unknown class distribution (Provost and Fawcett, 1997). In datasets with two classes, out of which one is very rare, even accuracy rates close to 100% may not be satisfactory, because the error that stems from the minority cases is always disproportionally large compared to the error that stems from the majority cases (Weiss and Provost, 2003). *ROC analysis* suggested in (Provost et al., 1998) provides a solution that counts the accuracy rates separately for the majority and the minority classes and compares algorithms in a statistical and a visual framework. Along with the ROC curves, the Area Under the Curve (AUC) becomes an additional performance measure, which is used quite often for imbalanced datasets (Chawla, 2003). Alternatively, the *geometric mean* of the accuracy rates for the majority and the minority class has been suggested as a performance measure for the comparison between different algorithms (Kubat and Matwin, 1997). All these measures study the accuracy rates for the two classes separately and attempt to choose the classifier that achieves high accuracy for the minority without loosing too much accuracy for the majority class.

In addition to the imbalance of the classes in the datasets, several of the problems we are facing often carry different misclassification costs. For example, in medical diagnosis problems it is probably more costly to miss the diagnosis of a rare fatal disease than classifying a non-sick person as sick. Conversely, in business classification problems like the customer insolvency problem, while it is important to predict as many insolvent customers as possible, it still may be very costly to misclassify a solvent customer as insolvent than the opposite. This is true considering that such actions may very well result to loss of good customers (Ezawa and Norton, 1996b). Changing the cost of each type of misclassification error alters the performance of the classification algorithms and results to better classifiers for difficult problems. Information about misclassification costs, if known, may be integrated into the learning process, or influence the choice for the splitting criteria and the pruning method such as in (Drummond and Holte, 2000a). In case the costs are not known, it is still possible to evaluate different classifiers using a minimum expected cost criterion (Chan and Stolfo, 1998a).

Altering the class distribution in the training set is another way of dealing with the problem of imbalanced class distributions. Weiss and Provost (2003) have studied a number of different datasets with varying natural distributions, but mostly imbalanced. They concluded that if the training of the classification algorithms is performed in datasets of fixed size, then the optimal class distribution for learning depends on the performance measure used. If accuracy is the measure then the "natural" distribution is in the optimal range, while if AUC is used then the balanced distribution is a better choice. In a similar study with a series of experiments and several datasets in (Chan and Stolfo, 1998b), the relationship between class distributions and classifiers' performance was investigated. Almost like in the previous article, it was demonstrated that the average error rate (or equivalently the total accuracy) is minimized (maximized) when the "natural" distribution of the classes is used for training. However, this is not true when other performance measures are utilized. Moreover, when more than one objective is used these are often conflicting and there is no single optimal solution. For example, increasing the number of the minority instances in the dataset results to greater accuracy for the minority instances and to less accuracy for the majority instances. One may thus conclude that for a given problem optimality cannot be reached through some algorithmic procedure but only by trial and error and for different problems the optimal class distribution may be different, depending on the performance measure of interest and the dataset. Moreover, under these conditions the role of the business objectives becomes more important because they may indicate which performance measures are more suitable for each problem.

Other studies have focused on alternative methods of changing the class distribution in the dataset. Oversampling and undersampling is one way of categorizing these methods, in the first case the minority class is oversampled in order for the dataset to reach the desired distribution, while in the second the majority class is undersampled until the target distribution is reached. In (Japkowicz and Stephen, 2002) both methods are presented as solutions to the problem of imbalanced dataset and their effectiveness is tested in artificial domains with varying concept complexity, size of the training set and class distribution. In the undersampling domain a few more studies have focused on the data reduction method adopted for changing the class distribution. It is argued that since the examples of the minority class are very precious, they are all kept intact, while the examples of the majority class are reduced to achieve the desired proportions. The techniques for this one-sided selection may vary from simple *random sampling*, to *instance-based data reduction techniques* (Kubat and Matwin, 1997; Laurikkala, 2001), which utilize nearest neighbor rules to reduce the larger class. Lastly, the combination of Synthetic Minority Over-sampling technique (SMOTE), a particular oversampling technique for the minority class, along with random under-sampling for the majority class has been proposed in (Chawla et al., 2002). In this paper it is argued that regular oversampling by simple replication of minority cases affects the decision regions in feature space and may lead to overfitting, thus it is necessary to use sophisticated techniques in order to increase the number of minority cases.

Among the different classification inducers, no algorithm has been reported to exhibit superior performance in cases like the one of our interest. For the Decision Tree building algorithms such as C4.5 and C5.0, it is known that they

are influenced by imbalanced datasets and thus introduce certain bias against the minority class (Weiss and Provost, 2003). Similarly, MLP (Multi-Layer Perceptrons) from the Neural Network family and Support Vector Machines (SVM) have been studied in (Japkowicz and Stephen, 2002). It is demonstrated that the performance of these algorithms is influenced by changes in class distribution, while changing the class distribution in the training dataset may improve their performance. In a more recent study (Akbani et al., 2004) SVMs have been criticized as not performing well when undersampling is the method for altering class distribution in the training set. A variant of SMOTE algorithm (Chawla et al., 2002) is proposed as a remedy for this malfunction of the algorithm.

Combining different classifiers is another suggested procedure to improve performance in classification problems. Well-known model combining methods are the *bootstrap aggregation* or *bagging* techniques (Breiman, 1996). They have been used mostly for combining the outputs of several models induced from the same decision tree algorithm but from different samples of the dataset, even though they can be used for combining classifiers induced from different algorithms as well. For the neural networks, *stacking* (Wolpert, 1992) is the equivalent counterpart model combining method. Just like with bagging techniques, stacking can be used to combine classifiers from different algorithms as well. While in most cases simple voting is the combining technique, weighing differently the outputs of the different models has also been used as an alternative. Other studies with combining methods can be found in (Dietterich, 2000; Bauer and Kohavi, 1999; Woods et al., 1997). These techniques have shown that classification performance may improve, even in the case that the parameters for the algorithms are not fine-tuned. Several issues that appear in this meta-learning strategy of combining classifiers concern either accuracy of the final classification outcome, or efficacy in producing classifications in a reasonable amount of time (Chan and Stolfo, 1997). Combining classifiers induced from different classification algorithms has also been suggested in (Seewald, 2003). An effort for combining two to six different data mining algorithms in (Abbott, 1999) exhibited significant improvement for the accuracy rates. Given that models are induced from different algorithms, experiments indicate that combining their uncorrelated outputs can be very promising.

Almost all articles reviewed in this section focus on one or two approaches of handling the problem of uneven class distribution, and in these cases the proposed approach is studied exhaustively with datasets from several problems. In this paper, we study several alternative methodologies for dealing with the problem of imbalance and focus more on combining these methodologies in one general framework that enables evaluation of the classifiers. More specifically, it is argued that in the case of difficult real world problems, while it is important to select an appropriate performance measure, it is also important to experiment with different class distributions for training the classification algorithms. In case that none of the algorithms prevails, it is almost imperative to combine different classifiers in voting schemes that adhere best to the objectives posed by the decision makers. Moreover, as a last step, relating the performance of the classifiers to economic factors it may facilitate the decision makers to understand better the impact of the classification results in terms of business and market characteristics. In order to further demonstrate this approach, the dataset of customer insolvency is solely used, because it is in this dataset that we better understand the business objectives of the problem. In addition, in this dataset the proportion between classes is 1:142 (i.e. less that 1% of the cases belong to the minority class), a situation that is much more extreme than the datasets studied in all aforementioned articles.

## 4. Performance measures for classification problems

In this section alternative performance measures that can be used for the evaluation of classifiers are surveyed and discussed. In order to clarify the relevant issues, let us assume a dataset involving two classes, the *minority* (also called *positive*) and the *majority* (also called *negative*). Classification algorithms customarily tabulate the results in confusion matrices that look like the one in Table 1.

| | | **Predicted** | |
|---|---|---|---|
| | | **Positive (P)** | **Negative (N)** |
| **Actual** | **Positive (P)** | True Positive Cases (a) | False Negative Cases (b) |
| | **Negative (N)** | False Positive Cases (c) | True Negative Cases (d) |

**Table 1.** A generic confusion matrix

In connection with this confusion matrix, the accuracy rates, averaged either over both classes or calculated for each class separately, are as follows:

*Average accuracy rate*: $$AA = \Pr\{\text{correct classification}\} = \frac{a+d}{a+b+c+d} \quad (1)$$

*True positive rate*: $$TP = \Pr\{\text{predicted P} \mid \text{actually P}\} = \frac{a}{a+b} \quad (2)$$

*True negative rate*: $$TN = \Pr\{\text{predicted N} \mid \text{actually N}\} = \frac{d}{c+d} \quad (3)$$

Instead of the accuracy rates, equivalently, one may observe their counterpart error rates. As discussed earlier, *AA* is an inappropriate performance evaluator for problems with highly imbalanced class distributions, because when $c + d >> a + b$ and $d >> a$ the successfully predicted cases in the minority class (a) play insignificant role in the calculation of measure (1). This was definitely true for the insolvency prediction dataset we used in our case study, where only 0.7% of the cases were insolvent and 99.3% solvent. Under these conditions, the minority cases may easily be treated as noise by the algorithms. Therefore, if an algorithm classifies all cases as solvent, then it achieves zero accuracy for the minority cases while it maintains high overall performance, using *AA* as performance measure. To overcome this problem measures (2) and (3) ought to be monitored separately. These rates measure the performance of a given classifier distinctively for each class and the objective is usually to keep both of them as high as possible. In order to transform this multi-objective problem to a single-objective equivalent, linear combinations of them (Chan and Stolfo, 1998b) or alternatively their geometric mean (Kubat et al., 1998) have been used. Similarly, with ROC analysis the *TP* and *FP*=1-*TN* rates of a classifier are plotted against each other to form a ROC curve. In this plot, the objective is to be as close as possible to the upper left corner, which represents the perfect classifier. In case of comparing the performance of classification algorithms, usually the objective is to choose the classifier that maximizes the area under the curve (Provost et al., 1998).

The accuracy rates (2) and (3) measure the performance of a classifier as the percentage of the actual cases predicted correctly for either class. However, in many prediction problems, it may be important to measure performance as the percentage of the correctly predicted cases out of the total number of cases named to belong to a given class (predictive value of a classifier). Especially for highly imbalanced datasets, where the interest is focused on the minority cases, by using only the accuracy rates (2) and (3) a classifier may outperform the others naming as "minority" a very large number of examples, thus resulting to an increased *TP* rate (desirable) and simultaneously to an increased *FP* rate (undesirable). In order to control such performances, a fourth performance measure that exhibits directly the ability of a classifier to be more precise with its predictions is what information retrieval community has called *precision rate* (Kubat and Matwin, 1997). According to the data in Table 1 this new measure is calculated as:

*Precision rate*: $$PR = \Pr\{\text{actually P} \mid \text{predicted P}\} = \frac{a}{a+c} \quad (4)$$

Just like *TP*, the precision rate is an important measure for problems with highly imbalanced datasets, because if a classifier classifies all cases to the majority class, then both *TP* and *PR* are zero. In fact, there is a relation between the two measures which is described by applying Bayes Theorem in (4):

$$PR = TP \frac{\Pr\{\text{actually P}\}}{\Pr\{\text{predicted P}\}} \quad (5)$$

where $\Pr\{\text{actually P}\}$ is the prior probability for the minority class and $\Pr\{\text{predicted P}\}$ is the probability that any case will be classified as positive.

Conclusively, there are three important measures that may influence our decision in choosing one classifier over the others: the *TP* and *PR* rates that measure accuracy and precision for the positive instances and *TN* rate that measures accuracy for the negative instances. However, while the benefits from monitoring the *PR* rate in a classification procedure have become apparent, a problem arises when the class distribution changes. As suggested in the next section, one may overcome this problem by taking into account the relative size of this change and multiply *c* in eq. (4) with a normalizing factor. Going one step further in our study, an effort is made to combine TP and PR rates by calculating their geometric mean as well as their F-measure. The benefits of using these measures will become apparent along with their presentation and application to the dataset. All the aforementioned performance measures are used suitably in the next section in order to evaluate the classifiers that result from several classification algorithms applied to a number of different class distributions.

## 5. Class distribution and classification performance

For the solution of the insolvency prediction problem different families of algorithms were used and their performances were compared for the given dataset. These were the *Linear Logistic Regression (LLReg)*, *Decision Trees* (*DT*) using the C4.5 algorithm (Quinlan, 1992), *Multi-layer Perceptron Neural Networks* (NN), *Bayes Networks* (Bnet), using the K2 hill-climbing search algorithm, *Multinomial Logistic Regression* (MLReg) with a ridge estimator (Le Cessie, van Houwelingen, 1992), and the *Sequential Minimal Optimization* (*SMO*) algorithm (Platt, 1986) for training a support vector classifier[2]. Working with the given dataset and maintaining the natural class distribution (1:142), the Multilayer Perceptron and the Multinomial Logistic Regression were able to train classifiers for predicting correctly only 2.04% and 1.53%, respectively, of the insolvent customers with a precision rate of 44% and 37.5%, respectively (Table 2). At the same time, the accuracy for the solvent customers is very high and reaches 99.98% for both classifiers. On the contrary, the Bayes Network, trained a classifier that provided high accuracy (64.29%) for the minority and 92.59% for the majority class, however the precision rate in this case was only 5.7%. On the other hand the Decision Tree, SMO and Linear Logistic Regression classifiers treated all positive examples as noise for the ten experiments in the 10-fold cross validation procedure, so both the *TP* and *PR* were averaged to zero. It is clear that for data sets with extremely imbalanced class distribution, training of the algorithms with the "natural" dataset is not always effective.

| *Classification Algorithm* | *Performance Measures* | | |
|---|---|---|---|
| | TP | TN | PR |
| *Neural Network (Multilayer Perceptron)* | 0,0204 | 0,9998 | 0,4444 |
| *Multinomial Logistic regression* | 0,0153 | 0,9998 | 0,3750 |
| *BayesNetwork (hill climbing search)* | 0,6429 | 0,9259 | 0,0572 |
| *Decision Tree ( Pruned C4.5)* | 0,0000 | 0,9999 | 0,0000 |
| *Support Vector Classifier (SMO)* | 0,0000 | 1,0000 | N/D |
| *Linear Logistic Regression* | 0,0000 | 1,0000 | N/D |

**Table 2.** Classification results when the "natural" distribution is used for training

While the importance of altering the class distribution is clear for the cases where the minority instances are very rare, it is also very important to find the class distribution that best works for the dataset in hand. To perform such a study a range of different class distributions was tested for the training of classifiers. More specifically, the "natural" class distribution 1:142, and the artificial ones 1:100, 1:50, 1:25, 1:15, 1:10, 1:5, and 1:1 (balanced) were employed. The new datasets were built using stratified random sampling in the original dataset, which was split into a testing set (using 25% of the data) and a training set (using the remaining 75%). The resulting training set, which carried along the natural distribution 1:142, was further used to create all the other training sets by undersampling. Since for our study all minority cases are precious, in order to change the distribution, a stratified random selection from the majority class was applied. This data reduction technique was used in order to maintain certain characteristics of the population in the same proportion as they were in the original dataset. No oversampling techniques were examimed in this study. While it is known that for a few imbalanced datasets oversampling has performed satisfactorily (Japkowicz and Stephen, 2002; Estabrooks and Japkowicz, 2004), in many other cases undersampling proves to be superior to oversampling (Domingos, 1999; Drummond and Holte, 2003).

[2]The main characteristics of the classifiers used are: *Decision Trees (DT)*: C4.5 pruned decision tree generating algorithm, using the C4.5 pruning algorithm with confidence factor 0.25 and subtree raising when pruning, with at least two instances per leaf. *Linear Logistic Regression (LLReg)* using LogitBoost with simple regression functions as base learners for fitting the logistic models, *Multi-layer Perceptron Neural Networks (NN)* using the backpropagation algorithm to train, with the initial learning rate set to 0.3 and the momentum applied to the weights during updating to 0.2, with 500 epochs and not specified number of hidden layers, *Bayesian Networks (Bnet)*, using a simple estimator with a=0.5 for finding the conditional probability tables of the network and using K2 a hill-climbing search algorithm with initial network used for structure learning a Naive Bayes Network, *Multinomial Logistic Regression (MLReg)* with a ridge estimator with ridge value in the log-likelihood = 1.0E-8, and the *Sequential Minimal Optimization (SMO)* algorithm for training a support vector classifier, which replaces all missing values and transforms nominal attributes into binary ones.

Figures 1 through 5 depict the performance of the six chosen classification algorithms for the different distributions, using the measures discussed untill now. In figures 1 and 2, the accuracy rates *TP* and *TN* are plotted. As one may observe, the accuracy rate for the minority class (*TP*) improves as the proportion of positive examples in the dataset increases (Fig. 1) to the expense of the accuracy rate for the majority class (*TN*), which decreases for the same changes (Fig.2).

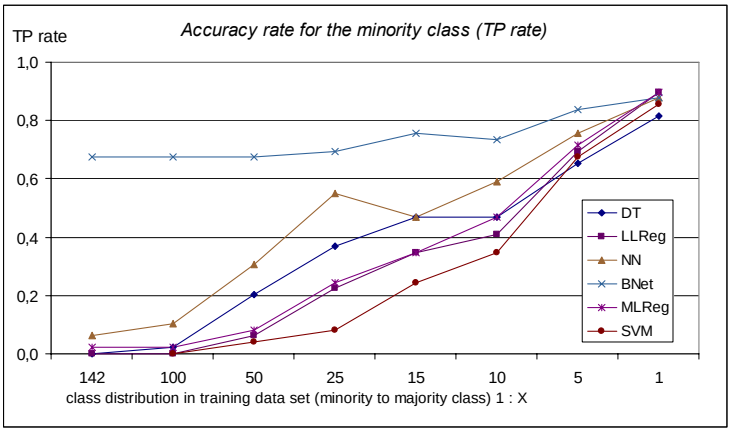



**Figure 1**. *True Positive rate* for classifiers induced by the algorithms Decision trees (DT), Linear Logistic Regression (LLReg), Neural Networks (NN), Bayes Network (Bnet), Multinomial Logistic Regression (MLReg) and Support Vector Machines (SVM) for different class distributions in the training data.

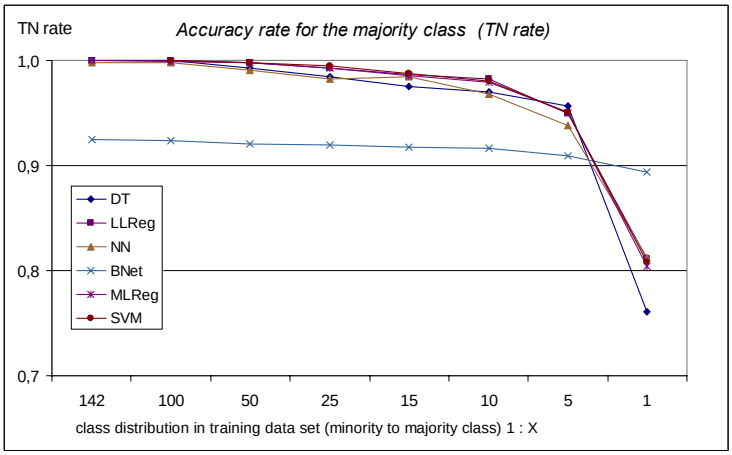



**Figure 2**. *True Negative rate* for classifiers induced by the algorithms Decision trees (DT), Linear Logistic Regression (LLReg), Neural Networks (NN), Bayes Network (Bnet), Multinomial Logistic Regression (MLReg) and Support Vector Machines (SVM) for different class distributions in the training data.

Moreover, we observe that the increase of positive examples in the dataset has roughly a similar effect to all classification algorithms but the Bayesian Network. This algorithm appears to be insensitive to changes in the class distribution, while it gives the best values for *TP* and the worst ones for *TN*. The insensitivity of the Bayesian networks to changes in class distribution has been previously observed in (Chan and Stolfo, 1998b) and rigorously discussed in (Elkan, 2001). Its extreme behavior, however, is quite unsuitable for our problem, where we are looking for reliable predictors and this is not ensured by Bayesian networks as it will be also shown using precision rate.

### *5.1. ROC curves and the AUC*

By plotting the *ROC curves* (Fig. 3) and moreover by calculating the AUC for each one of these algorithms, one may proceed with an overall comparison of the performance of the five classification algorithms, taking into account both *TP* and *TN* rates simultaneously. The ROC curves here are formed by plotting the pairs of *TP* and *FP=1-TN* rates from the classifiers that resulted after varying the class distributions in the training set. In this plot, it is desirable to be as close to the vertical axis and as high as possible. Moreover, one may compare the overall performance of different algorithms by examining whether any one of them is dominating over the others (Provost and Fawcett, 1997). In figure 3 the curves are intersecting in many points, so there is no clear dominance of any of

the algorithms. This is additionally verified from the values calculated for the AUC, which are very close for all algorithms. Therefore, it is concluded that based on a ROC analysis, their performance is indistinguishable.

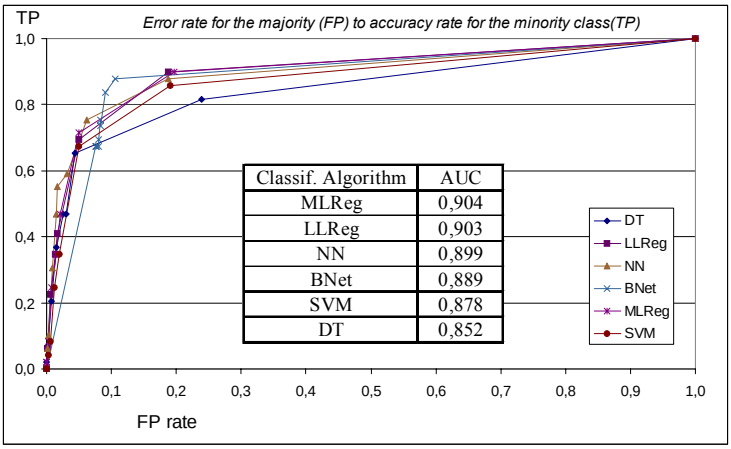



**Figure 3.** *ROC curves* for the classifiers induced by the algorithms Decision trees (DT), Linear Logistic Regression (LLReg), Neural Networks (NN), Bayes Network (Bnet), Multinomial Logistic Regression (MLReg) and Support Vector Machines (SVM) for different class distributions in the training data.

ROC curves may also be used to compare the classifiers that result from different class distributions. For this comparison we need to calculate the AUC separately for each specific class distribution. In order to achieve this, an ROC curve for any given algorithm and any given class distribution was formed from the points (TP, FP), (0, 0) and (1,1). The AUC measure for the six classification algorithms is plotted in figure 4 and indicates that the Bayesian Network gives the highest values for all different distributions. Moreover, all algorithms agree that increasing the number of minority cases in the training set improves AUC and that the optimal range includes the balanced training set (1:1), just like it was shown also in (Weiss and Provost, 2003). Comparing figures 1 and 4 it is worth noting that both measures exhibit similar behavior for all classifiers, thus the AUC calculated in this way may easily be considered as equivalent to the measure *TP*.

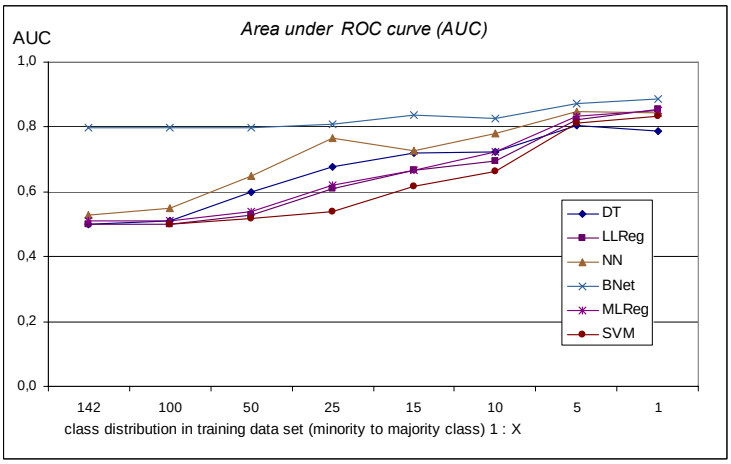



**Figure 4.** *AUC* calculated for the classifiers induced by the algorithms Decision trees (DT), Linear Logistic Regression (LLReg), Neural Networks (NN), Bayes Network (Bnet), Multinomial Logistic Regression (MLReg) and Support Vector Machines (SVM) and for different class distributions in the training data.

### *5.2 Precision rate and Normalized Precision Rate*

In order to calculate the *precision rate* (*PR*) for a class distribution other than the natural one, a modification to its definition is necessary. As noted in section 4, the *PR* rate is the percentage of the correctly classified positive cases over the total number of the cases predicted as positive. When the class distribution of a given dataset changes by

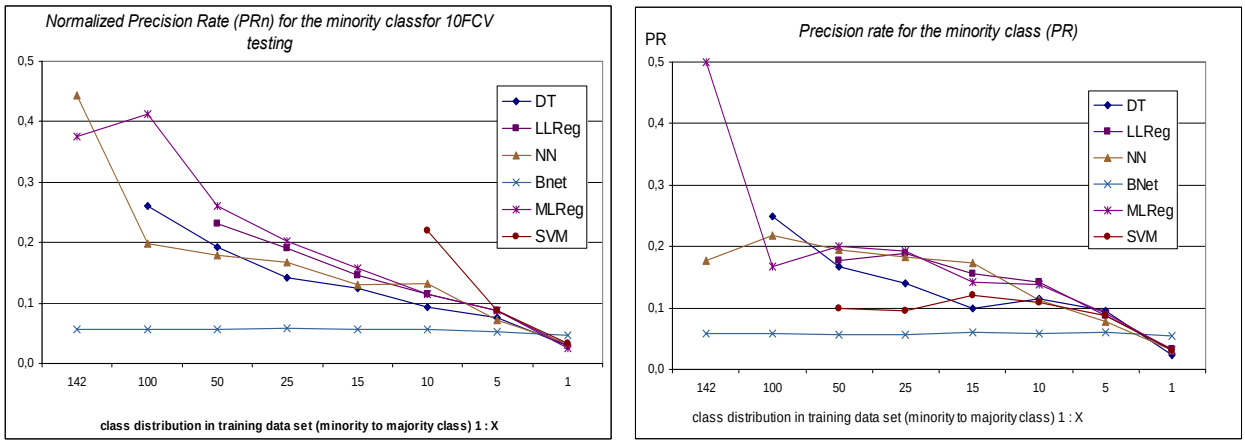
throwing away only majority cases, the calculation of the precision rate using equation (4) is no longer valid for the original population, since it does not take into account the relative size of the "uninteresting" cases (Provost and Fawcett, 1997). In order to compensate for the deformation of the results that occurs due to the one-sided downsizing of the population, a normalized counterpart of the *PR* may be used, defined as follows.

$$\textit{Normalized Precision Rate: } PR_N = \frac{a}{a + c\frac{N_o}{N_a}} \qquad (6)$$

where $N_o$ is the population for the majority class in the original dataset and $N_a$ is the population for the majority class in the dataset with the artificial distribution. Using the normalizing factor $\frac{N_o}{N_a}$, the number for the false alarms in the new confusion matrix is corrected and the *PR* rates which result from datasets with different class distributions are comparable. It is worth noting that the normalizing factor introduced here for correcting the precision rate coincides with the over-sampling ratio introduced in (Weiss and Provost, 2003) to correct the probability estimates for observing a minority instance by the decision tree inducer.

In order to check the ability of the normalized precision rate to estimate the precision rate when the test set does not carry the natural distribution, we performed the following experiment. First, from the original dataset, seven more datasets were created each with a different class distribution: (1:100, 1:50, 1:25, 1:15, 1:10, 1:5 and 1:1). Using each one of them separately, all previously mentioned algorithms were trained and tested with a ten-fold cross validation procedure. For these experiments, the testing set did not carry the "natural" distribution so the precision rate was calculated using the formula for the normalized precision (eq. 5) and produced the plots in figure 5a. Second, the original dataset was split into 25% for testing and 75% for training, using stratified random sampling. The first set was kept intact for the testing and only the second one was re-sampled to create the desired distributions. The precision rates for the classifiers that resulted from the second procedure were plotted in Fig. 5b. As one may observe the results are indeed very close, especially as we move towards the class distribution 1:1, where the testing sets have the greatest difference. To verify this observation, a pairwise t-test was performed. The test concluded that at a 5% significance level there is no evidence that for the classifiers induced from different class distributions the precision rates from testing in a dataset with the natural distribution are different from the normalized precision rates from the 10-fold procedure.

From figure 5(b) one may observe that none of the six classification algorithms prevails in performance based on this measure. On the contrary, it is certain that the Bayesian Network algorithm induces classifiers that are not reliable enough for our problem, since they achieve approximately 6% precision for all class distributions.




(a) (b)

**Figure 5**. *Precision* and *Normalized Precision Rates* for the classifiers induced by the algorithms Decision trees (DT), Linear Logistic Regression (LLReg), Neural Networks (NN), Bayes Network (Bnet), Multinomial Logistic Regression (MLReg) and Support Vector Machines (SVM) and for different class distributions in the training data.

Comparing figures 1 and 5(b), we may conclude that the measures *TP* and *PR* are conflicting each other when the class distribution changes. This can be justified as follows. It is expected that the higher the percentage of positive cases in the training dataset, the higher the probability of positive prediction from an induced classifier, thus both the

false positive (*c*) and the true positive (*a*) cases in eq. (4) are expected to increase. Moreover, if the percentage of the increase in the true positive cases is higher than the corresponding percentage for the false positive cases, then *PR* decreases. Apparently, since the majority class outnumbers severely the minority class, even a small increase for the minority is much larger percentage compared to an increase of the same order in the majority class. It is worth noting the conflict between *TP* and *PR* rates as an important feature of precision for problems with highly imbalanced datasets and the same objectives as of our problem. For any given algorithm it provides a control measure of the relative improvement in *TP* compared to the simultaneous loss in *TN*. In this sense, *PR* becomes more informative than *TN*, *ROC* curves or *AUC* and together with *TP* are the performance measures that better adopt the business objectives in our problem. For this reason in the next section two more measures which combine specifically *TP* and *PR* rates are examined.

### *5.3 The geometric mean of TP and PR*

The geometric mean of *TP* and *PR* rates, $\sqrt{TP \cdot PR}$ , combines the two preferred measures into the square root of their product, and is different from the geometric mean of *TP* and *TN* examined in (Kubat et al. 1998). It takes the value 1, when both *TP* and *PR* equal to one; also the value 0, when either *TP* or *PR* equals zero. For all other values of *TP* and *PR*, their geometric mean $\sqrt{TP \cdot PR}$ is a number in the interval $(0,1)$. Judging from figures 1 and 5, one could claim that in general the *TP* rate increases with the number of minority cases in the dataset and the *PR* rate decreases for the same changes. Thus, an increase of the geometric mean indicates that the achieved increase in *TP* is quite beneficial because it is not accompanied by a "large" decrease of *PR*. Lastly, the geometric mean attains a maximum at the class distribution where the benefit from the increase in *TP* rate is larger.
Figure 6 depicts how the geometric mean of *TP* and *PR* is influenced by the class distributions and the six classification algorithms for the customer insolvency problem. We observe that this measure exhibits an approximately concave behavior and attain maximum values that appear to be in the range of 1:25 to 1:5 for all classification algorithms. Using the geometric mean as performance measure, the classifiers induced by the Neural Network algorithm exhibit superior behavior for almost all class distributions, by achieving maximum performance at the 1:25 dataset. The rest of the classification algorithms behave in a comparable way and attain maximum performance either at the 1:10 or 1:5 dataset. The only exception we observe is due to the classifiers induced by the Bayesian network algorithm, which as already discussed, are insensitive to changes in the class distribution, therefore the geometric mean is approximately the same for all class distributions.

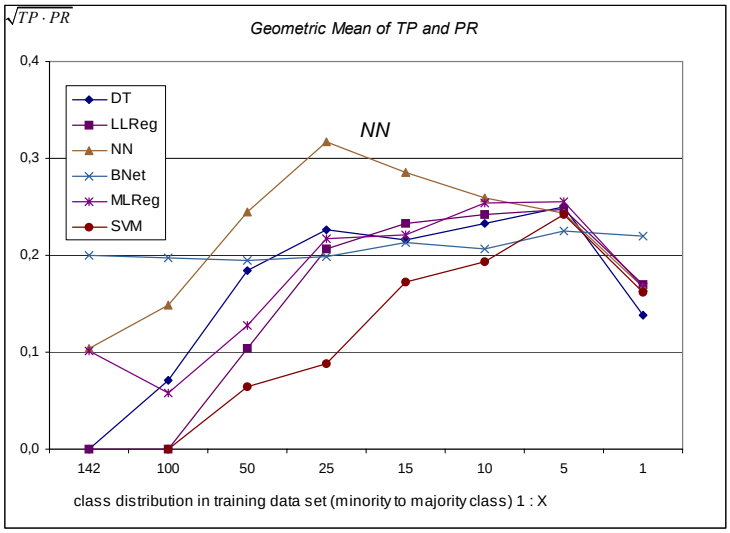



**Figure 6**. *Geometric mean of TP and PR* for classifiers induced by the algorithms Decision trees (DT), Linear Logistic Regression (LLReg), Neural Networks (NN), Bayes Network (Bnet), Multinomial Logistic Regression (MLReg) and Support Vector Machines (SVM) and for different class distributions in the training data.

### *5.4 The F-measure of TP and PR.*

The *F*-measure (Lewis and Gale, 1994) is another performance measure where *TP* and *PR* rates may be combined. It is given by the equation:

$$F = \frac{(\beta^2 + 1) * TP * PR}{\beta^2 * PR + TP} \qquad (6)$$

where the $\beta$ factor is a parameter that takes values from 0 to infinity and is used to control the influence of *TP* and *PR* separately. It can be shown that when $\beta = 0$ then *F* reduces to *PR* and conversely when $\beta \to \infty$ then *F* approaches *TP*. Moreover, if for some dataset *TP* = *PR* then all four measures *F*, *TP*, *PR* and $\sqrt{TP * PR}$ coincide. Lastly, given that *TP* and *PR* are positive numbers less than 1, it can be shown that this is true for *F* as well. In Fig. 7 the six classification algorithms are compared based on the *F*-measure, when $\beta = 1$. As one may see, the classifiers induced by the Neural Network prevail by giving the highest values for several datasets followed by the classifiers from the Decision Tree algorithm and the Linear Logistic Regression. The classifiers from Bayes Network give approximately the same value (close to 0.1) for all class distributions and the classifiers induced by Multiple Logistic Regression give the smallest values for all class distributions except only of the dataset 1:1 where it gives the highest. In addition, the datasets in the range 1:25 to 1:5 appear to train the classifiers in a way that achieve the highest *F* values for all algorithms.

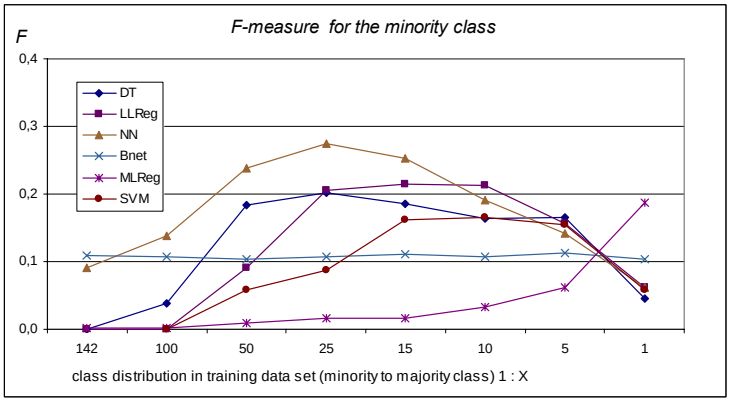



**Figure 7**. *F-measure of TP and PR* for the classifiers induced by the algorithms Decision trees (DT), Linear Logistic Regression (LLReg), Neural Networks (NN), Bayes Network (Bnet), Multinomial Logistic Regression (MLReg) and Support Vector Machines (SVM) and for different class distributions in the training data.

Concluding this part of our study, for a whole range of class distributions (from the set with the "natural" to the set with the "balanced" distribution) using six classification algorithms and several performance measures, it is clear that the measures *TP* and *PR,* which better reflect the business objectives of our problem, are conflicting to each other. The geometric mean $\sqrt{TP * PR}$ and *F*-measure of the *TP* and *PR* manage to combine the two measures in an effective way and compare the different classifiers. As suggested by these two measures the classifiers induced by the Neural Network algorithm may provide better overall predictions for the insolvent customers. They are followed by the Decision Tree algorithm and the Linear Logistic Regression. For further improving the predictive capability of our classifiers, in the next section the combination of classifiers into voting schemes is additionally studied.

## 6. Combining different classification models

In this section, the combination of those classifiers that were induced by the most successful algorithms is attempted for further improvement of the predictions for insolvent customers. Three different voting schemes, a democratic, and two veto rules were designed for this purpose. Given a dataset for training, we let the classifiers that were induced by the three most successful classification algorithms i.e. Linear Logistic Regression, Decision Trees and Neural Networks to vote separately for each case in the testing dataset. Based on the results of the voting procedure, three different decision schemes have been adopted for comparison purposes.

- *Rule* #1 (the *democratic rule*): Any given case is classified to class *i*, if two or more classifiers vote for *i*.
- *Rule* #2 (the *majority class veto rule*): Any given case is classified to the minority class, if all three classifiers vote for this class; otherwise the case is classified to the majority class.
- *Rule* #3 (the *minority class veto rule*): Any given case is classified to the majority class, if all three classifiers vote for this class; otherwise the case is classified to the minority class.

Rule #1 is the "democratic" decision scheme and therefore is quite popular in voting procedures, due to its fairness with either class. Rules #2 and #3, on the other hand, have been designed specifically for problems with highly imbalanced class distributions and different misclassification costs. Rule #2 suits better to situations where the

characterization of a negative instance as positive is more risky than the converse, and rule #3 is exactly the reverse i.e. works better when calling a positive instance as negative is more risky than the converse.
For this study, the three voting schemes were applied in conjunction with the classifiers that were induced by the algorithms of NN, DT and LLReg for all different datasets (class distributions 1:142 to 1:1). Their performance regarding the measures *TP*, *TN*, *PR*, *AUC*, $\sqrt{TP \cdot PR}$ and *F* are shown in figures 8 - 13. In each one of these graphs, besides plotting the results for the voting schemes R1 to R3, the respective plots for the three classification algorithms are also shown. A comparison of their performance exhibits that R3 gives the best *TP* rates and *AUC*, while at the same time it gives the worst *TN* and *PR* rates. Exactly the reverse is observed for rule R2, while the democratic rule R1 is always between the other two. This behavior of the rules was expected since R3 is conservative towards the negative class and R2 is conservative towards the positive class. Given that apart from high *TP* rate, in this problem we are also looking for high *PR* rate it is once more clear that combinations of *TP* and *PR* should be examined for the final decision.

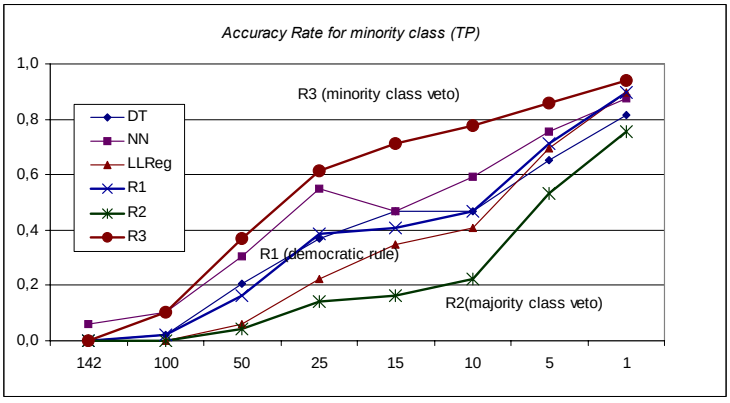



**Figure 8.** *True Positive rate* for the voting rules R1 - R3, Neural Network (NN), Decision Tree (DT) and Linear Logistic Regression (LLReg) for different class distributions in the training set.

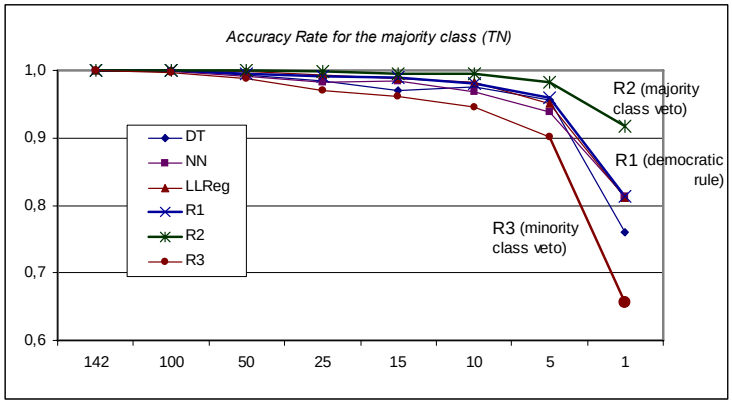



**Figure 9.** *True Negative rate* for the voting rules R1 - R3, Neural Network (NN), Decision Tree (DT) and Linear Logistic Regression (LLReg) for different class distributions in the training set.

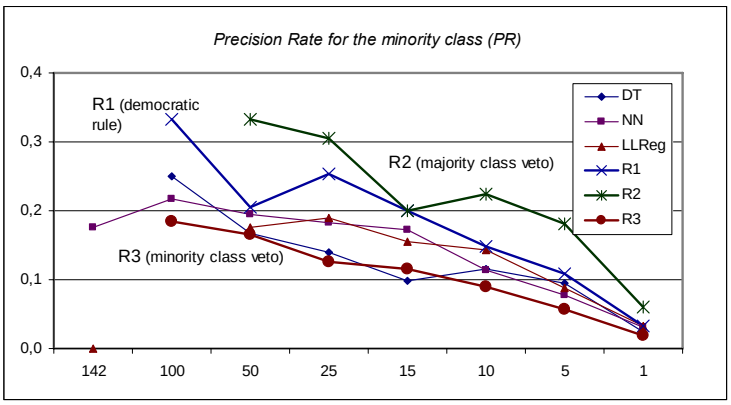



**Figure 10.** *Precision rate* for the voting rules R1 - R3, Neural Network (NN), Decision Tree (DT) and Linear Logistic Regression (LLReg) for different class distributions in the training set.

Taking into account the geometric mean (Fig. 12), it can be seen that either Rule #1 or Rule #2 from the voting rules may ensure equally good performance as long as the algorithm is trained with a suitable dataset, which is 1:25 for R1 and 1:5 for R2. Rule #3 also performs well getting its maximum at the 1:15 dataset. Comparing the voting rules with the simple classification algorithms, R1 and R3 perform in a way that is comparable to the NN classifier, while R2 outperforms all others in the dataset 1:5. Similar conclusion we can draw examining the F-measure for $\beta = 1$ (Fig. 13). Again, R1 and R2 attain their maximum value at the 1:25 and 1:5 datasets and they are followed by the NN classifier, which achieves its best performance at the 1:25 dataset.

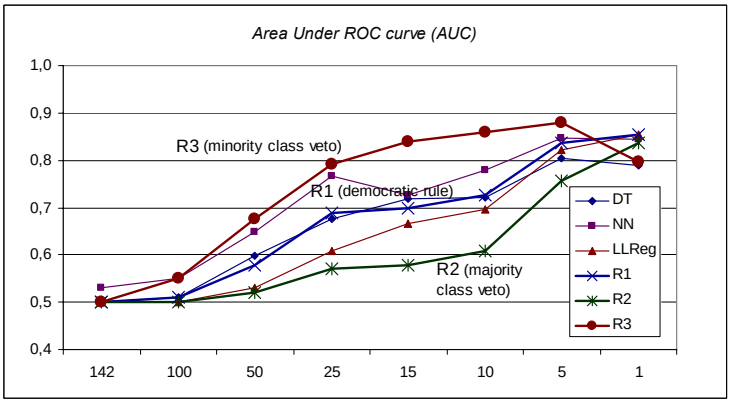



**Figure 11.** *Area under curve* for the voting rules R1 - R3, Neural Network (NN), Decision Tree (DT) and Linear Logistic Regression (LLReg) for different class distributions in the training sets.

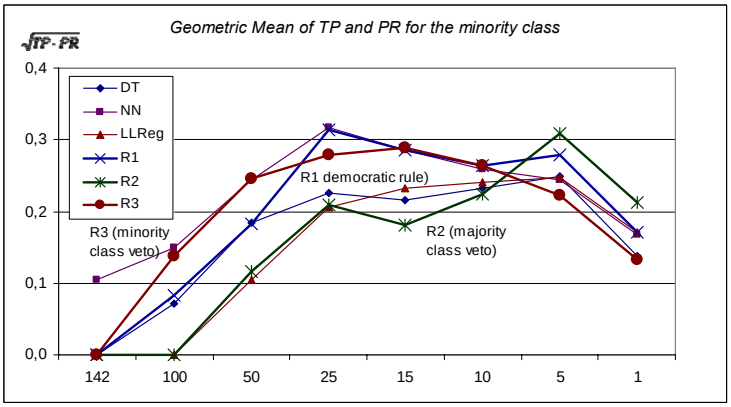



**Figure 12.** *The Geometric mean for TP and PR* for the voting rules R1 - R3, Neural Network (NN), Decision Tree (DT) and Linear Logistic Regression (LLReg) for different class distributions in the training set.s

Given that the F-measure is more than one performance measure we additionally examined more values for the $\beta$-factor. Figure 13 presents also the plots for $\beta = ¼$, $\beta = ½$ and $\beta = 5$. It is clear that for small values of $\beta$ (e.g. $\beta = ¼$), when the *PR* rate influence heavier the *F*-measure, the voting rule R2 prevails with the best values attained at the 1:25 dataset followed by the 1:10 and 1:15 dataset. The picture changes when the value for $\beta$ increases. Then the voting rules R1 and R3 respectively become more dominant, since the *TP* rate influences more the value for F. When $\beta = 5$ then R3 is the best voting scheme because it achieves higher *TP* rate.

Our conclusion for this part of our study is summarized as follows:

- The voting rules R1 to R3 always prevail with their performance compared to the simple classification algorithms.
- The geometric mean and the F-measure are performance measures suitable for selecting class distribution for the training dataset since they can combine both measures of our interest and thus exhibit an optimal value or an optimal range.
- In order to reach a decision for a given problem one more step is necessary for this analysis. It is important to understand the relative importance between the two measures, the *TP* and *PR* rates in order to select a suitable value for the $\beta$-factor.

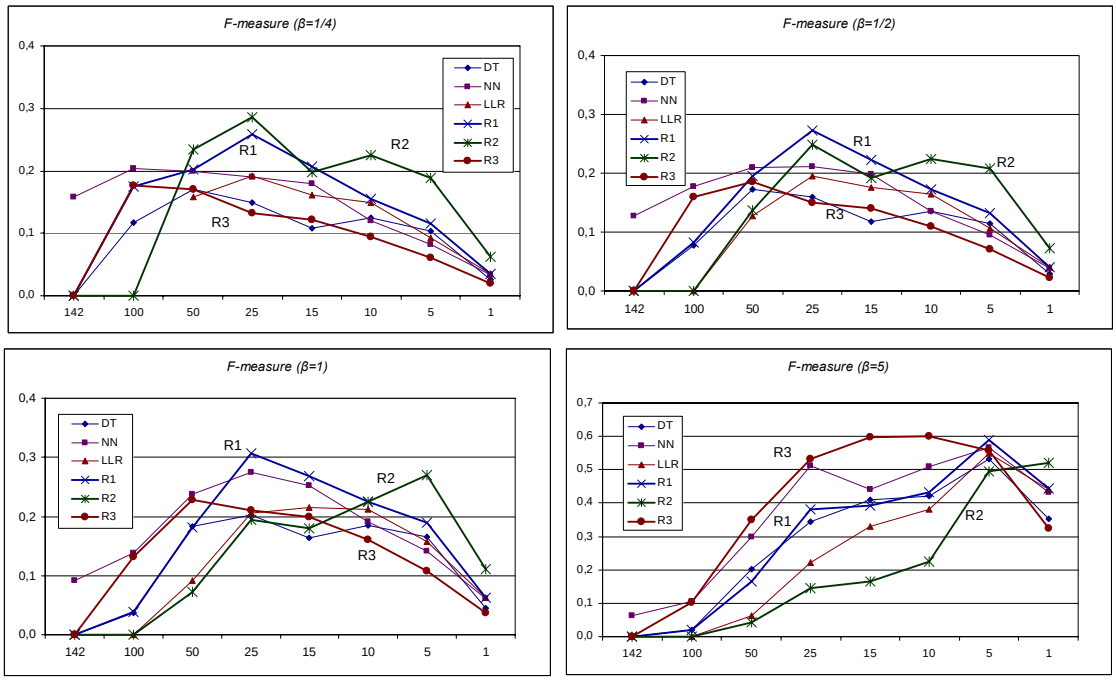



**Figure 13.** The *F-measure* for the voting rules R1 - R3, Neural Network (NN), Decision Tree (DT) and Linear Logistic Regression (LLReg) for different class distributions in the training set and for $\beta$ =1/4, 1/2, 1 and 5.

In the next section this last step is attempted, however in a different context. Since the two measures of interest are influenced only by the true positive and the false positive cases in the confusion matrix of each classifier, a method for evaluating the economic impact of the classification results using these terms is developed. Through this method the decision makers can relate more naturally measurable facts to classifier performance.

### 7. Evaluation of economic impact of classification results.

The evaluation of classifiers presented up to this point was based mainly on performance measures that do not take into account any information outside the confusion matrices that result from the classification procedures. Given that our problem emanated from an important and highly competitive business sector, a comparison based on the economic aspects of the classification results was considered also necessary. Using the business criteria originally set forth to us, a linear function that represents the *Total Gain* for the organization was defined. For a given

population of customers whose insolvency is predicted by a given classifier, this function combines *TP* and *FP=1-TN* in the following manner:

*Total Gain* (*TG*) = *Gain from (True) Positives* (*GP*)– *Loss from (False) Negatives* (*LN*)

where *GP* accounts for the gain originating from classifying correctly positive cases and similarly *LN* accounts for the loss that is possible to result from incorrect classifications on the negative cases. If we further assume that $C_P$ and $C_N$ represent the cost (gain) per customer that is correctly classified as insolvent and the cost per customer incorrectly classified as insolvent, respectively, the function *TG* is further defined as follows:

$$TG = a \cdot C_P - c \cdot \left( \frac{N_0}{N_a} \right) \cdot C_N \qquad (7)$$

where *a* and *c* are the number of true positive and false positive cases, respectively, from Table 1 and $\frac{N_0}{N_a}$ is the normalizing factor that accounts for the one-sided reduction of the data, if an "artificial" class distributions is used for the test data set. This factor equals to one if the "natural" distribution is used for testing instead. The costs $C_P$ and $C_N$ in equation (7) are assumed positive numbers. Using non zero costs only for the true positive and the false positive cases is quite a reasonable structure of the cost matrix for this type of problems (Zadrozny and Elkan, 2001). The meaning of $C_P$ and $C_N$ can be expressed through some functions $f_i(\omega_i, c_i), \quad i = 1,2$, where $\omega_i$ represent weights with values in [0,1]. These weights express the percentage of the insolvent to be customers that are expected to be turned around and eventually recover or a percentage of the solvent customers characterized as insolvent that may be distressed and cause loss to the company by stepping away. Similarly, $c_i$ represent the average amount in their monthly bills for each class separately. While the cost coefficients are not easy to calculate exactly, as will be shown in this section, it is only the relative gain $C_P / C_N$ that is necessary for indicating whether predictions provided from a classification scheme will result to profitable solutions for the company.

Since the decision makers will be interested in a classification scheme if and only if *TG > 0*, it follows that

$$\frac{c_n}{a} < \frac{C_P}{C_N} \qquad (8)$$

where $c_n = c \cdot \left( \frac{N_0}{N_a} \right)$, is the normalized value for the false positive cases in the confusion matrix. Moreover, from equations (5) and (7) a similar relation may follow for the precision rate:

$$PR_n = \frac{a}{a + c_n} \Rightarrow PR_n > \frac{1}{\frac{C_P}{C_N} + 1} \qquad (9)$$

Equations (8) and (9) provide rules of thumb for the performance measures and can be calculated from the confusion matrices. Thus the outcome of a classification algorithm can be related directly with the profitability of a system which may potentially be developed using such a classification scheme. It is suggested that in order for the *TG* to be positive, it is necessary for any selected classifier to provide a value for $\frac{c_n}{a}$ which is smaller than the relative gain per insolvent customer compared to the loss per misclassified solvent customer. Also, the precision rate must be quite high if the relative gain $C_P / C_N$ is small and conversely, if the fraction $C_P / C_N$ is larger, then the precision is allowed to be smaller and the $\frac{c_n}{a}$ to be larger. To make results (8) and (9) more intuitive, figure 14 visualizes the inequalities (8) and (9) and the shaded area shows the feasible region for *PR* and the ratio $c_n / a$. One may observe that the relative gain from correctly classified insolvent customers must always be larger than the number $\frac{-1+\sqrt{5}}{2}$ (the positive root of the equation $x^2 + x - 1 = 0$) in order for the classification to be of any value.

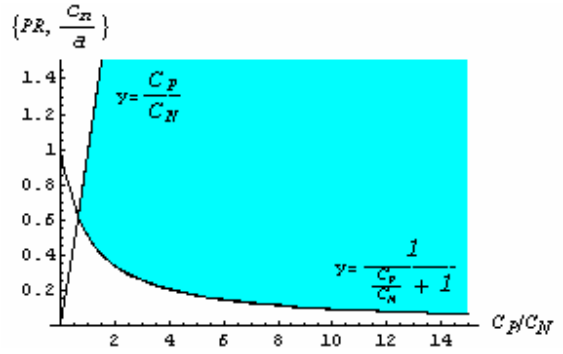



**Figure 14**. *The feasible region for PR and the ratio* $\frac{c_n}{a}$ *depends on the relative gain* $C_P / C_N$

To study further the impact of $\frac{C_P}{C_N}$ in the total gain we can write equation (7) as follows:

$$\frac{TG}{C_N} = a \cdot \frac{C_P}{C_N} - c_n \qquad (10)$$

Given any classifier it is now easy to plot the line represented by eq. 10 for different values of $C_P / C_N$, which is the relative gain from correct classification in the positive class compared to the cost from incorrect classification in the negative class. As an example in figures 15-20, these lines are plotted for each one of the voting rules studied in this section and for the three classification algorithms used for their generation. It is important to track the points where each line crosses the horizontal axis (given by $\frac{c_n}{a}$), because it is from that point and on that the total gain takes positive value (eq. 8). Also, the slope of each line is important (given by $a$) since it indicates the rate of change for the total gain. Even if the relative gain from correct classification is not known one may still find the classifiers that would maximize the total gain for different values of $C_P / C_N$. In particular, one should first choose a classifier that crosses the x-axis earlier than the others and follow this line until another one with higher slope crosses it. Repeating the procedure until there is no classifier left with higher slope, we end up with a multisegment line where each line segment represents the optimal classifier for some specific interval in the definition set.

This approach which is developed here to study the economic impact of classification is analogous to the methodology presented in (Drummond and Holte, 2000b; Drummond and Holte, 2004) where the normalized cost curves are plotted against the probability cost function. It is claimed that the cost curves are the duals to certain points in ROC space (Provost and Fawcett, 2001) and thus the optimal cost curves form the dual representation to the ROC convex hull, indicating however the range of class distributions and cost fractions where a given classifier dominates over the others. In order to demonstrate this idea with our case study, we first chose the classifier with the lowest breakeven point (figure 21). This is classifier R2_50 because it crosses the x-axis when $\frac{C_P}{C_N} = 2$ and represents the best classifier for a certain interval of values for $C_P / C_N$. It is worth noting that $2 > \frac{-1+\sqrt{5}}{2} = 0.618$, so it satisfies both rules postulated by eq. (8) and (9). The classifiers with slopes higher than the slope of the line for R2_50 will eventually cross it. The one that crosses it earlier indicates the best classifier for the next interval of values for $C_P / C_N$.

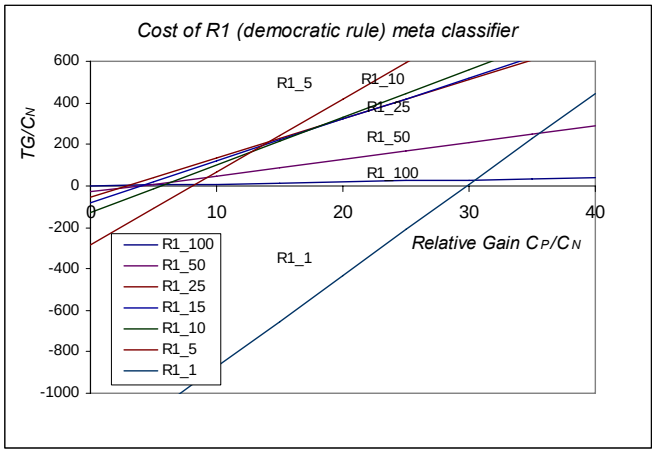



**Figure 15**. Study of total gain for the classifiers resulting from Rule #1 (democratic rule)

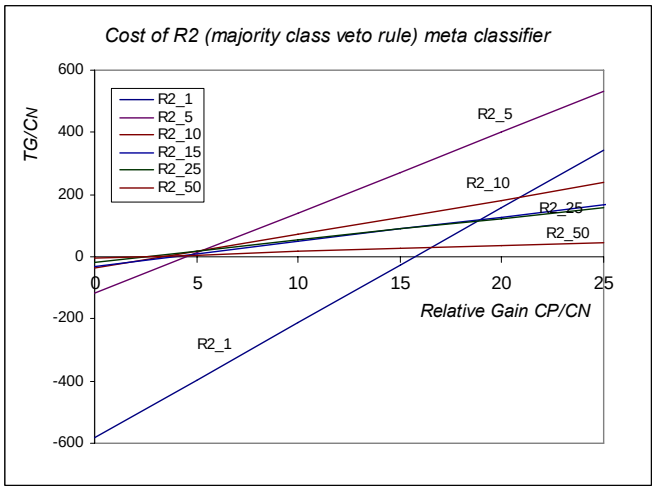



**Figure 16**. Study of total gain for the classifiers resulting from Rule #2 (majority class veto)

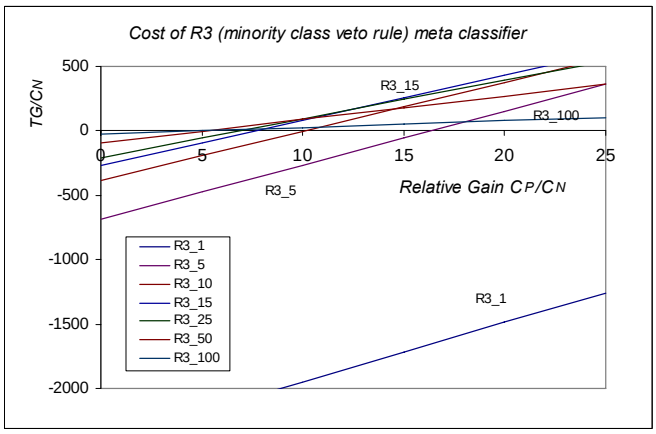



**Figure 17**. Study of total gain for the classifiers resulting from Rule #3 (minority class veto)

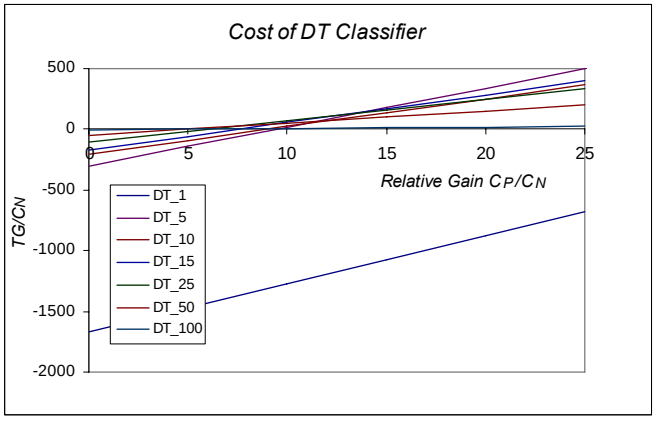



**Figure 18**. Study of total gain for the classifiers resulting from the Decision Tree algorithm

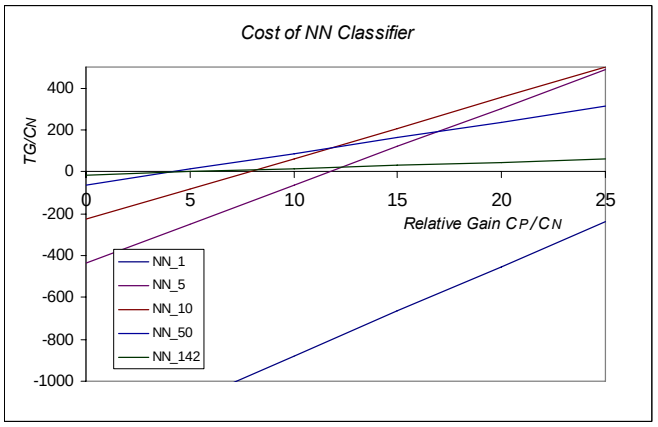



**Figure 19**. Study of total gain for the classifiers resulting from the Neural Network algorithm

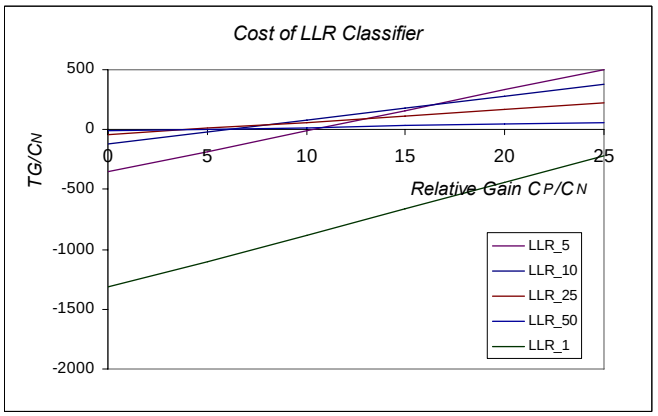



**Figure 20.** Study of total gain or the classifiers resulting from the Linear Logistic Regression algorithm

Following this procedure for all classifiers examined in this section we end up with the set exhibited in Figure 22 (with the solid line) and also in Table 3. These classifiers are the optimal ones, each for some specific interval of values of the fraction $C_P / C_N$, shown in the 1st column of Table 3. It appears that the classifiers that resulted from the voting schemes introduced in section 6 dominate the classifiers that generated them with the exception of NN_25 that exhibits superior performance in the interval (8.1 – 18.1). All three voting schemes appear to provide optimal classifiers for some specific range of values of $C_P / C_N$, and this is due to their performance examined earlier in this paper using various performance measures.

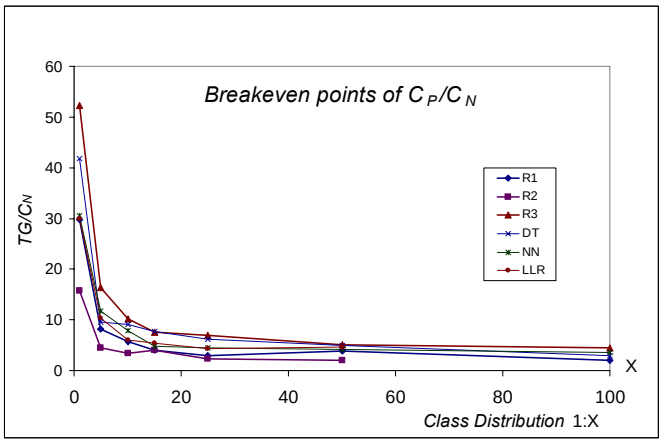



**Figure 21.** Breakeven points for all classifiers.

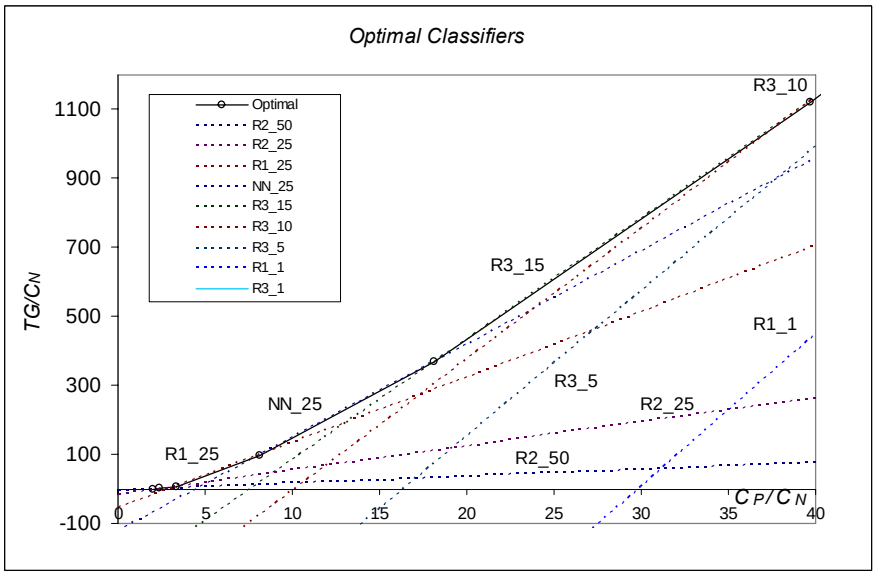



**Figure 22.** The optimal classifiers for the customer insolvency problem

In addition, in Table 3 the performance of each classifier in the optimal list is indicated with respect to the performance measures examined in section 5. We can see that for small values of $C_P / C_N$ (between 2.0 and 2.4) the classifier that resulted from the voting scheme R2 "majority class veto rule" and specifically from the dataset 1:50 is the preferred classifier. From figures 8 – 13 we can observe that the chosen classifier R2 gives the highest (H) precision rate and F-measure (with $\beta = ¼$) for the dataset with class distribution 1:50 but the lowest (L) or almost lowest (≈L) *TP*, *AUC*, *GM* and F-measure (with $\beta = ½$, $\beta = 1$, $\beta = 5$). Figure 23 brings together figures 8 – 13 and indicates (with the large ball points) the performance of the optimal classifiers with respect to the different performance measures. Similarly, for the interval (3.3, 8.1) the best classifier is R1, trained with the dataset 1:25. We note that this classifier carries the highest value for the geometric mean and highest or almost highest for the F-measure (with $\beta = ¼$, $\beta = ½$, $\beta = 1$), while for the other performance measures the classifier R1_25 take neither the lowest nor the highest value (indicated as Medium).

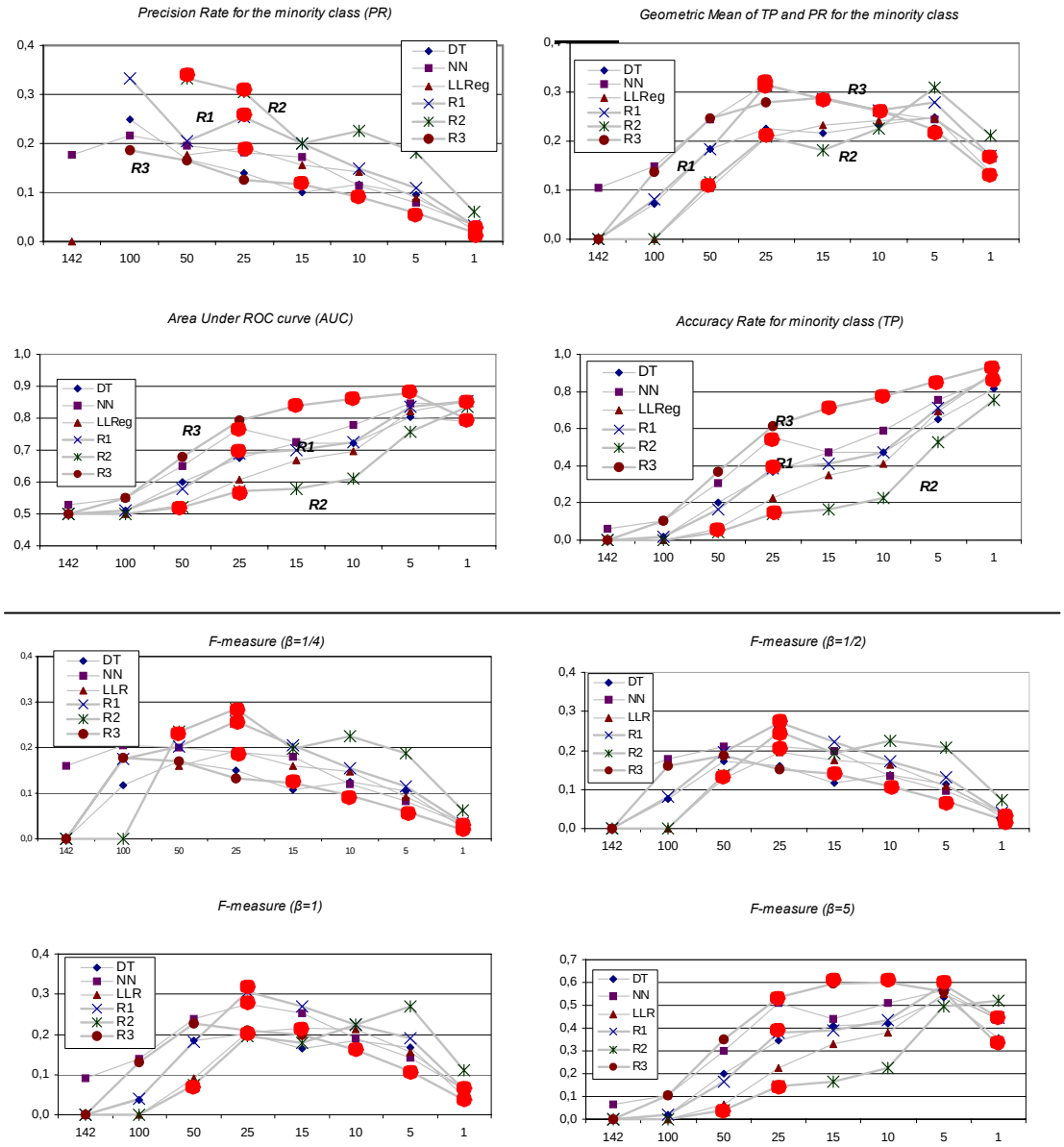



**Figure 23.** Optimal classifiers in relation to performance measures: *PR, GM, AUC, TP* and *F* (with β = 1/4, β = ½, β = 1, β = 5).

As the value of $C_P / C_N$ increases, the relative gain from correct classification in the positives is very large and the selection of the best classifier depends solely on the *TP* rate, while the other measures drop to their lowest values. This result is intuitive; however, the procedure is absolutely necessary in order to define the optimal classifiers and the interval where each one of them dominates over the others. The shadowed cells in the table indicate the measures that play the most significant role for different values of the relative gain $C_P / C_N$. It is obvious that importance is shifted gradually from *PR* to *GM* to *AUC* to *TP* or equivalently the corresponding values for the β-factor in the F-measure.

| $C_P/C_N$ | Classifier | PR | GM | AUC | TP | F – measure β=1/4 | β=1/2 | β=1 | β=5 |
|---|---|---|---|---|---|---|---|---|---|
| 2.0 - 2.4 | R2_50 | **H** | ≈L | L | L | **H** | ≈L | L | L |
| 2.4 - 3.3 | R2_25 | **H** | ≈L | L | L | **H** | **≈H** | L | L |
| 3.3 - 8.1 | R1_25 | M | **H** | M | M | **≈H** | **H** | **H** | M |
| 8.1 - 18.1 | NN_25 | M | **H** | **≈H** | **≈H** | M | M | M | **≈H** |
| 18.1- 39.7 | R3_15 | ≈L | **H** | **H** | **H** | ≈L | ≈L | M | **H** |
| 39.7- 75.5 | R3_10 | L | **H** | **H** | **H** | L | L | L | **H** |
| 75.5- 313 | R3_5 | L | L | **H** | **H** | L | L | L | **≈H** |
| 313 - 547 | R1_1 | ≈L | M | **H** | **≈H** | ≈L | ≈L | ≈L | M |
| >547 | R3_1 | L | L | L | **H** | L | L | L | L |
| **H** = Highest, L = Lowest, **≈H =** almost Highest, ≈L = almost Lowest, M = Medium | | | | | | | | | |

**Table 3.** Optimal classifiers for the customer insolvency problem

Lastly, from Table 3, it is important to understand the relationship between the class distribution in the training set and the optimal classifiers. When the gain from the correctly classified positives is a "small" multiple of the cost from the incorrectly classified negatives (e.g. low values of $C_P / C_N$) then datasets with high class distributions (e.g. 1:50 to 1:25) in connection with classifiers that ensure high precision are preferred. As $C_P$ grows to larger multiples of $C_N$, then datasets with class distributions 1:25 to 1:10 are required in connection with classifiers that ensure high value of the geometric mean of *TP* and *PR* as well as high *AUC*. Lastly, when $C_P$ is a very large multiple of $C_N$, then more balanced datasets are required for training in connection with classifiers that ensure high *AUC* and high *TP* rate. This last observation agrees also with the conclusion in (Weiss and Provost, 2003) that the balanced class distribution is in the optimal range when *AUC* is used as performance criterion.

**8. Summary and conclusions**

In this paper, we discussed issues related with certain difficult classification problems characterized by uneven class distributions. Being motivated by the problem of predicting customer insolvency in telecommunications businesses, we studied the behavior of six different classification algorithms and three voting schemes in connection to several performance measures and to varying class distributions for the training set. During this process it is suggested that the business objectives, set usually by the domain experts and users of the classification results, should be the guide for major decisions. However, for the evaluation process of the resulting classifiers it is worth setting a framework that is based on the business objectives and the economic aspects of the classification. A discussion on the role of the domain knowledge and business objectives in data mining projects, inspired by the same study, is also included in (Kopanas et al. 2002).

Summarizing our conclusions, our study involved several steps and certain key decisions that concern the classifier evaluation process. The first refers to the performance measures that should be used, which need to reflect the business objectives of the real problem. Under this assumption, the True Positive (*TP*) and Precision (*PR*) rates are suggested as the most informative measures for problems like the prediction of customer insolvency where predicting correctly as many minority cases as possible is important, as long as this is not risking classifying too many majority cases to the minority class. When used in conjunction to measure the performance of a given classifier, the *TP* and *PR* measure its predictive capability for the minority class. However, it is difficult to find classifiers that perform well in both because the objectives of maximizing *TP* and *PR* are usually conflicting. Given that the majority class is so much larger than the minority, one has to balance the success in the minority class with the error in the majority class. This can be achieved by using either the geometric mean of *TP* and *PR* or their F-measure (for various values of β) as alternative measures that combine them. As has been discussed extensively in the paper, under certain conditions these two measures can be used for selecting optimal classifiers.

The second decision concerns the distribution between the two classes in the training set. Given that classification algorithms cannot be trained satisfactorily with highly imbalanced datasets and because the "natural" distribution is not always the best distribution for training, a number of datasets with different class distributions ought to be used, so that an "optimal" one can be selected. The class distribution in the training set does play major role in the performance of most classification algorithms tested with the exception of the Bayesian Networks. Different performance measures, however, indicate different class distributions as the optimal ones, thus it is important to understand first the business environment and the risk involved with the predictions from the classification system. Then learning becomes cost-sensitive, and optimality depends on the cost ratio or relative gain. For instance, in our case study when the gain from the correctly classified positives is a "small" multiple of the cost from the incorrectly classified negatives (high risk environment) then high precision is required and predictors trained with unbalanced datasets, like the 1:50 or 1:25, perform better. On the contrary, as the relative gain becomes very large (low risk environment), then the accuracy in the minority class is highly desirable and the classifiers trained with more balanced datasets, like 1:5 or 1:5 perform better. When both precision and accuracy are required for the minority class then the Geometric Mean and F-measure (with $\beta = 1$) are better performance measures to expose the "best" class distribution.

The third decision is about the selection of classification algorithms. In general, no algorithm has been reported to prevail in highly imbalanced dataset. This has been confirmed with our problem, where six algorithms: Linear Logistic Regression, C4.5 from Decision Trees, Multi-Layer Perceptron from Neural Networks, Bayes Network, Multinomial Logistic Regression and Sequential Minimal Optimization have been applied. On the contrary, simple voting rules that combine the results from classifiers induced from different algorithms do manage to prevail with their performance and dominate according to almost all performance measures. As expected the "veto" rules exhibit extreme behavior, while the democratic rule is more conservative. In any case, however, the evaluation process indicated the voting schemes as the preferred classifiers in most cases. Unfortunately, different voting schemes prevail according to different measures, thus necessitating again a cost-sensitive decision making approach.

The proposed framework for cost-sensitive decision making is the result of a study on the economic aspects of alternative classifiers performance. It evaluates classifiers by taking into account the classification results from the confusion matrix and a cost matrix. Using a cost matrix suitable to our business objectives, a function for the total gain from classification was defined. Maximizing the total gain leads to a procedure that identifies the "optimal" classifiers from the available set, based on the relative gain from correctly classifying minority cases.

Overall, despite the fact that this work has been inspired by a specific case study, the methodology adopted here is generic enough to be applicable to many other problems that bear similar characteristics to the one discussed here.

## References


Abbott, D., 1999, "Combining Models to Improve Classifier Accuracy and Robustness", Proceedings of the International Conference on Information Fusion – Fussion99, Sunnyvale, CA, U.S.A.

Akbani, R., S. Kwek and N. Japkowicz, 2004, "Applying Support Vector Machines to Imbalanced Datasets", in J.-F Boulicaut et al. (Eds), ECML 2004, LNAI, 3204, pp. 39-50.

Brieman, L., 1996, "Bagging predictors", Machine Learning, Vol. 24, 123-140.

Bauer, E. and R. Kohavi, 1999, "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants", Machine Learning, Vol. 36, 105-139.

le Cessie, S. and van Houwelingen, J.C. (1992). "Ridge Estimators in Logistic Regression", Applied Statistics, Vol. 41, No. 1, 191-201.

Chan, P. K. and S. J. Stolfo, 1998a, "Toward Scalable Learning with Non-Uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection", Proceedings Fourth Intl. Conf. On Knowledge Discovery and Data Mining, pp. 164-168.

Chan, P. K. and S. J. Stolfo, 1998b, "Learning with Non-Uniform Class and Cost Distributions: Effects and a Multi-classifier Approach", Work. Notes KDD-98 Workshop on Distributed Data Mining, pp.1-9, August 1998.

Chan, P. K. and S. J. Stolfo, 1997, "On the Accuracy of Meta-learning for Scalable Data Mining", J. of Intelligent Information Systems, Vol. 8, pp. 5-28.

Chan, P. K., F. Wei, A. Prodromidis, and S. J. Stolfo, 1999, "Distributed Data Mining in Credit Card Fraud Detection", IEEE Intelligent Systems, 14(6), pp. 67-74.

Chawla, N. V., 2003, "C4.5 and Imbalanced Data sets: Investigating the effect of sampling method, probabilistic estimate and decision tree structure" Workshop on Learning from Imbalanced Datasets II, International Conference on Machine Learning, Washington D.C.

Chawla, N.V., K.W. Bowyer, L.O. Hall, and W. P. Kegelmeyer, 2002, "SMOTE: Syntehetic Minority Over-sampling Technique", Journal of Artificial Intelligence Research, Vol. 16, pp. 321-357.

Daskalaki, S., I. Kopanas, M. Goudara, and N. Avouris, 2003, "Data mining for decision support on customer insolvency in telecommunications business", European Journal of Operational Research, 145(2), 239-255.

Dietterich T. G., 2000, "An Experimental Comparison of Three Methods for Constructing Ensembels of Decision Trees: Bagging, Boosting, and Randomization", Machine Learning, 40, 139-157.

Domingos, P. (1999), “Metacost: A General Method for Making Classifiers Cost-sensitive”, Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 155-164.

Drummond, C. and R.C. Holte, 2004, “What ROC curves Can´t Do (and Cost Curves Can)”, Proceedings of the ROC Analysis in Artificial Intelligence, First International Workshop, Valencia, Spain, August 2004, pg. 19-26.

Drummond C. and R.C. Holte, 2003, “C4.5, Class Imbalance, and Cost Sensitivity: Why Under-Sampling beats Over-Sampling, Workshop on Learning from Imbalanced Datasets II, International Conference on Machine Learning, Washington D.C.

Drummond, C. and R.C. Holte, 2000a, “Exploiting the Cost (In)sensitivity of Decision Tree Splitting Criteria”, Proceedings of the Seventh International Conference on Machine Learning, pp. 239-246.

Drummond, C. and R.C. Holte, 2000b, “Explicitly Representing Ecpected Cost: An Alternative to ROC Representation”, Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 198-207.

Elkan, C., 2001, “The Foundations of Cost-Sensitive Learning”, Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, pp. 973-978

Estabrooks, A., T. Jo, and N. Japkowicz, 2004, “A Multiple Resampling Method for Learning from Imbalances Data Sets" , Computational Intelligence, 20(1), pp.18-36.

Ezawa, K.J., M. Singh, and S.W. Norton, 1996a, “Learning Goal Oriented Bayesian Networks for Telecommunications Management”, Proceedings of the Thirteenth International Conference on Machine Learning, 139-147.

Ezawa, K.J. and S.W. Norton, 1996b, “Constructing Bayesian Networks to Predict Uncollectible Telecommunications Accounts”, IEEE Expert/Intelligent Systems & Their Applications, 11(5), 45-51.

Fawcett T. and F. Provost, 1997, “Adaptive Fraud Detection”, Data Mining and Knowledge Discovery, 1, 291-316.

Gur Ali, F.O. and W.A. Wallace, 1997, “Bridging the gap between business objectives and parameters of data mining algorithms”, Decision Support Systems, 21, 3-15.

Japkowicz, N. and S. Stephen, 2002, “The Class Imbalance Problem: A Systematic Study”, Intelligent Data Analysis Journal, 6(5).

Kohavi, R., 1995, “A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection”, International Joint Conference on Artificial Intelligence (IJCAI), 1137-1145.

Kopanas I., Avouris N.M. and S. Daskalaki, 2002, The role of knowledge modeling in a large scale Data Mining project, in I.P Vlahavas, C.D. Spyropoulos (eds), Methods and Applications of Artificial Intelligence, LNAI no. 2308, pp. 288-299, Springer-Verlag, Berlin.

Kubat, M. and S. Matwin, 1997, “Addressing the Curse of Imbalanced Training Sets: One-Sided Selection”, Proceedings of the 14th International Conference on Machine Learning, 179-186.

Kubat, M., R. Holte and S. Matwin, 1998, “Machine Learning for the Detection of Oil Spills in Satellite Radar Images”, Machine Learning, 30, 195-215.

Laurikkala, J., 2001, “Improving Identification of Difficult Small Classes by Balancing Class Distribution”, in Artificial Intelligence in Medicine, S. Quaglini, P. Barahona, S. Andreassen (Eds.), LNAI 2101, p. 63-66.

Lewis, D.D. and W. Gale, 1994, A Sequential Algorithm for Training Text Classifiers, Proceedings of the Seventh Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, p. 3-12, Springer-Verlag.

Platt J. (1998). "Fast Training of Support Vector Machines using Sequential Minimal Optimization". Advances in Kernel Methods - Support Vector Learning, B. Scholkopf, C. Burges, and A. Smola, eds., MIT Press.

Provost, F. and T. Fawcett, 2001, “Robust Classification for Imprecise Environments”, Machine Learning, 42, p. 203-231.

Provost, F. and T. Fawcett, 1997, “Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions” Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, p. 43-48, Menlo Park, CA, AAAI Press.

Provost, F., T. Fawcett, and R. Kohavi, 1998, “The Case Against Accuracy Estimation for Comparing Induction Algorithms” Proceedings of the Fifteenth International Conference on Machine Learning (IMLC-98), pp. 43-48, Morgan Kaufmann, San Francisco, CA.

Quinlan J., 1992, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Francisco, CA.

Riddle P. R. Segal, and O. Etzioni, 1994, “Representation Design and Brute-Force Induction in a Boeing Manufacturing Domain” Applied Artificial Intelligence, Vol. 8, p. 125-147.

Seewald A.K., 2003. Towards a Theoretical Framework for Ensemble Classification. In Proceedings of the 18th Int. Joint Conference on Artificial Intelligence (IJCAI-03), Morgan Kaufmann, 2003.

Weiss, G. and F. Provost, 2001, “The effect of class distribution on classifier learning”, Technical Report ML-TR-43, Department of Computer Science, Rutgers University.

Weiss, G. and F. Provost, 2003, “Learning When Training Data are Costly: The effect of Class Distribution on Tree Induction”, Journal of Artificial Intelligence Research, 19, p. 315-354.

Wolpert, D., 1992. Stacked generalization, Neural Networks, 5, p. 241-259.

Woods, K., W. P. Kegelmeyer Jr., and K. Bowyer, “Combination of Multiple Classifiers Using Local Accuracy Estimates”, IEEE Transactions on Pattern Analysis and Machine Intelligence”, Vol. 19, Issue 4, p. 405-410.

Zadrozny B. and C. Elkan, 2001, “Learning and Making Decisions When Costs and Probabilities are Both Unknown”, Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining, p. 204-213.