Int Adv Econ Res (2007) 13:301–312
DOI 10.1007/s11294-007-9090-2

Multiclass Corporate Failure Prediction by Adaboost.M1

Esteban Alfaro Cortés · Matías Gámez Martínez · Noelia García Rubio

Published online: 26 April 2007

Abstract Predicting corporate failure is an important management science problem. This is a typical classification question where the objective is to determine which indicators are involved in the failure or success of a corporation. Despite the complexity of the matter, a two-class problem has usually been considered to tackle this classification task. The objective of this paper is twofold. On the one hand, we apply the Adaboost.M1 algorithm to improve the accuracy of a classification tree in a multiclass corporate failure prediction problem using a set of European firms. On the other, we introduce novel discerning measures to rank independent variables in a generic classification task.

Keywords Corporate failure prediction · Ensemble classifiers · Adaboost.M1

JEL C10 · G30 · M00

Introduction

As in any classification task, initially a set of n observations is given and noted as $T_n = \{(X_1,Y_1), (X_2,Y_2), ..,(X_n,Y_n)\}$, where each X_i is a p-dimensional vector whose components are the values of the ith observation in each of the p features, that is to say $i = \{X_{i1}, X_{i2}, ..., X_{ip}\}$ and Y is the observation class label and takes values in $\{1, 2, ..., k\}$. On the basis of the training set, a classifier is constructed, in general, as a function of the p features, $C(X_i)=f(X_i)$. This function is used to predict the i class while minimizing the prediction error.

E. Alfaro Cortés (✉) · M. Gámez Martínez · N. García Rubio
Economic and Business Sciences Faculty of Albacete, Castilla-La Mancha University,
Plaza de la Universidad 1, 02071 Albacete, Spain
e-mail: Esteban.Alfaro@uclm.es

M. Gámez Martínez
e-mail: Matias.Gamez@uclm.es

N. García Rubio
e-mail: Noelia.Garcia@uclm.es

In a classification problem, a committee of classifiers can be used to increase the prediction accuracy, that is to say, it aggregates the predictions of several classifiers. Aggregation, combination and ensemble are synonymous in the literature of this field (Kuncheva 2004; Valentini and Masulli 2002). The classifier built combining some classifiers is called an ensemble of classifiers. There are several alternatives here, the first one being to build different classifiers from the data set and then combine them by simple vote or linear functions. Another possibility, perhaps more sophisticated, consists in applying the same classification method in modified versions of the learning set. Some of these techniques are quite new and have been studied quite closely in the last few years, among which, bagging and boosting methods deserve a special mention (Freund and Schapire 1996, 1997; Breiman 1998).

Although in corporate failure prediction literature only a binary classification problem is usually considered to differentiate failed from healthy firms, it makes sense to think that there are more than one type of failure. For instance, in the Spanish case only bankruptcy and temporary receivership firms are traditionally considered as failed firms, but it seems that there are other types of failure. In this study acquired and dissolved firms are also included as failed firms. We analyze whether these firms have a different financial behaviour with respect to healthy firms. With this aim, the application works with a three classes problem (healthy, failed1, and failed2), where failed1 includes acquired and dissolved firms, and failed2 includes bankruptcy and temporary receivership. The question here is if discriminating between two different states of failure is possible.

In this research, a new algorithm is proposed for predicting corporate failure. To show its utility, we apply this method over a sample of Spanish companies. In order to guarantee that our results have a general character and can be extrapolated to both European countries and the United States, we use financial ratios that have been found significant in predicting business failure in previous studies, such as Frydman et al. (1985).

Furthermore, a novel measure for the importance of variables is proposed to facilitate model interpretation. This measure takes into account how many times each variable is actually used throughout the individual trees. On the basis of this measure, the variables can be ranked in terms of importance.

Within the empirical application the following factors should be taken into account:

- Not only bankruptcy and temporary receivership firms as is usual in corporate failure prediction literature, but also acquired and dissolved firms are considered.
- Not only the usual financial ratios are included as predictors, but also qualitative variables, such as the firm size, activity, and legal structure.
- The Adaboost.M1 algorithm is applied to the corporate prediction, analysing the extent to which this methodology is suitable for the subject.

The paper is organized as follows. First, we present the boosting method included in the study. We then discuss how it works in practice, together with the Adaboost.M1 algorithm used. The following section introduces the failure prediction problem and the data used in the analysis. The classification results are then presented, with a comparison of the well-known classification tree model with the novel Adaboost.M1 classifier. The empirical analysis is followed by the main conclusions.

Boosting

As previously mentioned, given a data set, a classification method builds a model which is able to predict the class of a new observation. The accuracy of the classifier will depend on the quality of the method used and the difficulty of the specific application. If the obtained classifier achieves a better accuracy than the default rule, then the classification method has found some structure in the data enabling it to do so. Boosting (Freund and Schapire 1996) is a method that makes the most of a classifier by improving its accuracy. So, the classifier method is used as a subroutine to build a classifier with a high accuracy in the training set.

Boosting applies the classification system repeatedly on the training data but on each occasion focuses the learning attention on different examples of this set. Once the process has finished, the single classifiers obtained are combined in a final classifier with a high accuracy in the training set. The final classifier, likewise, usually achieves a high accuracy in the test set, as several authors have shown both theoretically and empirically (Bauer and Kohavi 1999; Breiman 1998; Dietterich 2000; Friedman et al. 2000; Freund and Schapire 1996, 1997).

Even though there are several boosting algorithm versions (Friedman et al. 2000), the most widely used is that by Freund and Schapire (1996) known as Adaboost. However, it can be only applied in binary classification problems, which is not our case. Among the versions of boosting algorithms for multiclass classification problems, we have chosen the most simple and natural extension of Adaboost to $k>2$ classes, which is called Adaboost.M1 (Freund and Schapire 1996).

This algorithm can be described as follows. A training set is given $T_n=\{(X_1,Y_1), (X_2,Y_2), ..,(X_n,Y_n)\}$, where Y takes values in {1, 2, ..., k}. The weight $w_b(i)$ is assigned to each observation X_i and is initially set to $1/n$. This value will be updated after each step. A basic classifier is built on this new training set, which is noted as $C_b(i)$ and applied to each training example. The error of this classifier is represented by ε_b and is calculated as

$$\varepsilon_b = \sum_{i=1}^{n} w_b(i)\xi_b(i) \quad \text{where} \quad \xi_b(i) = \begin{cases} 0 & C_b(X_i) = y_i \\ 1 & C_b(X_i) \neq y_i \end{cases} \tag{1}$$

From the error of the classifier in the bth iteration the constant α_b is calculated, this value being used for weight updating. Specifically, according to the above mentioned authors $\alpha_b=1/2\ \ln(1-\varepsilon_b/\varepsilon_b)$, and the new weight for the $b+1$th iteration will be

$$\omega_{b+1}(i) = \omega_b(i) \quad \exp(\alpha_b\xi_b(i)) \tag{2}$$

Later, the calculated weights are normalized to sum one. Consequently, the weight of the wrongly classified observations is increased, and the weight of the rightly classified is decreased, forcing the single classifier built in the next iteration to focus on the hardest examples. Moreover, differences in weight updating are greater when the error of the single classifier is low because if the classifier achieves a high accuracy the few mistakes take on more importance. Therefore, the alpha constant can be interpreted as a learning rate calculated as a function of the error made on each iteration. Moreover, this constant is also used in the final decision rule giving more importance to the individual classifiers that made a lower error.

This process is repeated in every step for b=1, 2, 3, ..,B. Finally, the ensemble classifier calculates, for each class, the weighted sum of its votes. Therefore, the class with a higher weighted vote is assigned. Specifically,

$$C(x) = \arg_{y_j}\max\sum_{b=1}^{B}\alpha_b\delta\left(C_b(x), y_j\right) = \arg_{y_j}\max \sum_{b:C_b(x)=y_j}\alpha_b \qquad (3)$$

Table 1 summarises the adaboost.M1 algorithm (Freund and Schapire 1996).

Problem Description

Predicting corporate failure is an important management science problem. The main goal of corporate failure prediction is to differentiate firms with a high probability of distress in the future from healthy firms. That is to say, a model is built to forecast the moment of distress, in order to allow economic agents related with the firm to take the appropriate decisions. To be able to predict the failure, it is absolutely necessary to have information about the situation of the company. This information is given basically by the financial ratios, but additional information should also be taken into account, such as the activity, the size, or the age of the firm.

Corporate failure prediction is not a new research field and many studies have dealt with this problem since 1966. Therefore, studying the state of the art of corporate failure prediction is an interesting task. There is no doubt that the pioneering failure prediction studies were Beaver (1966) at an univariate level and Altman (1968) applying discriminant multivariate analysis. Later, and basically because of the restrictive statistical requirements of normality for predictors variables and equality for variance–covariance matrices of the groups, logit and probit models

Table 1 AdaBoosst.M1 algorithm

1. Start with $w_i^1 = 1/n$, i=1, 2, ..., n.

2. Repeat for b=1, 2, ...,B.

   a) Fit the classifier $C_b(x)\in\{1,2,....,k\}$ using weights w_i^b on T^b.

   b) Compute $\varepsilon_b=\sum_{i=1}^{n} w_i^b\,\xi_b(i)$ and $\alpha_b = 1/2\ \ln(1 - \varepsilon_b / \varepsilon_b)$.

   c) Update the weights $w_i^{b+1}=w_i^b \exp(\alpha_b\ \xi_b(i))$ and normalize them.

3. Output the final classifier.

$$C(x)=arg\max_{y_j}\ \sum_{b=1}^{B}\alpha_b\,\delta\,(C_b(x), y_j)$$

were also applied (Ohlson 1980; Zmijewski 1984). Classification trees or recursive partitioning showed their usefulness in studies, such as Frydman et al. (1985). More recently, artificial neural networks have been introduced as a powerful approach to this task (Wilson and Sharda 1994). The comparison of the prediction ability of alternative techniques in accounting should be made carefully because of the different starting conditions in alternative studies.

Even though there is a general consensus on the importance of failure prediction, there is not the same degree of agreement on the definition of corporate failure, that is to say, when a firm is considered to have failed. From a global perspective, a firm will have failed if it does not achieve its goals, especially those related to profitability, solvency, and survival. Following this definition, a wider concept of corporate failure is used in this research. Traditionally, in Spanish failure prediction studies, only bankruptcy and temporary receivership firms have been considered, but in this study, acquired and dissolved firms are also included as failed firms. Acquired firms are considered to be failed firms because the company loses its identity sometimes as a result of poor management, but in any case this question should be studied thoroughly in future research. The main point here is to show that multiclass failure prediction problems can be managed. Consequently, we consider three classes in this study (healthy, failed1, and failed2), where failed1 includes acquired and dissolved firms, and failed2 includes bankruptcy and temporary receivership.

The companies in the sample were selected from the SABI database of Bureau Van Dijk (BVD), one of Europe's leading publishers of electronic business information databases and one of the providers of the Wharton Research Data Services. SABI covers all the companies whose accounts are placed on the Spanish Mercantile Registry. In the case of failed firms, those firms which failed during the period 2000–2003 were selected, but a further requirement was the complete data available for the moment of failure together with that of the previous 5 years.[1] Selecting failed firms from various years in order to collect a higher sample size is usual in failure prediction studies. There were firms that failed in different years of the period 2000–2003, so the information on variables should be understood in relative terms with respect to the moment of failure(t), the previous years being $t-1$, $t-2$, $t-3$, $t-4$, and $t-5$.

On the other hand, healthy firms were selected among active companies at the end of 2003 with complete data available for 2003 and the five previous years. In this case, a second requirement was added, viz., those firms with continuous negative profits for the previous 3 years were rejected. The reason is that they were active at least until December 2003, but if they keep making a loss they would soon be entering a state of failure.

Within these requirements, 512 firms were selected at random for each group (failed1/failed2/healthy). Therefore, the selection process did not pair firms of the three classes by sector and size. By contrast, these variables were used as predictors: the sector as a qualitative variable with 10 categories using the Economic Activities National Classification code (NACE-93 digit-1 level) and the size using the natural logarithm of Total Assets as a proxy variable. The legal structure is also used as a categorical predictor with three options: corporation, limited liability company, and others.

[1] In this paper, only the information from the previous year was used, but this is a part of a larger research. That is the reason for the requirement of complete data available for the five previous years

On the other hand, 14 accounting-based ratios are included in the initial data set. In failure prediction studies, financial ratios are usually selected on the basis of three criteria: they should be commonly used in failure prediction literature, the information needed to calculate these ratios should be available and, finally, the researchers' own decision based on their experience in previous studies or ad hoc on the basis of the preliminary trials must be taken into account. The same criteria have been followed in this study. Therefore, 18 predictor variables are used for each company with information for the year previous to moment of failure. These variables can be seen in the Appendix.

The total initial sample consists of 1536 Spanish companies, of which 90 percent were used as training set and the rest as a test set. Therefore the training set has 1383 observations and the test set has 153, keeping the same proportion of the classes. In the training set, a classifier was built both by the classification tree and the boosting method. Afterwards, these classifiers were tested in the remaining examples.

Empirical Results

In this paper, the same multiclass failure prediction problem is solved by two different classification methods in order to compare their performances. To estimate the real accuracy, data is divided in two sets: 90 percent is used as a training set to build the classifier, and the rest remains hidden to the classification method and is presented as new data to check the prediction ability.

In the application of the Adaboost.M1 algorithm, one difficulty arises. Unfortunately, the latter has not been implemented in any available statistical package. As a result, the researchers have had to learn to programme in R language to implement this boosting algorithm themselves.

The implementation of this paper is developed using the R program.[2] The R program has a base environment with a few statistical, mathematical, and graphical utilities. More sophisticated techniques need to be added using packages, which are available at CRAN at the website of the program (see footnotes) . These are the cases of the *rpart* library (Ripley 2004) which allows classification trees to be applied and the *adabag* library (Alfaro et al. 2006) for Adaboost.M1.

Corporate Failure Prediction Through Classification Trees

The training set is formed by 461 firms of each class (1383 firms are 90 percent of the total set). The test set is formed by 153 firms equally divided into healthy, failed1 and failed2 firms (10 percent of the total). As has been mentioned before, for each company 18 predictor variables are used with information for the year previous to the failure moment.

[2]The R program is a set of packages for data manipulation, calculus, and graphics (R Development Core Team 2004). Among other characteristics, it has a well-developed and effective-programming language (R language). The R program has much in common with the well known S-Plus program, but unlike the latter, R is a distribution-free program available on the web at http://cran.r-project.org/.

Before analysing the main results, let us give a brief review of classification tree technique. It is a non-linear and non-parametric classification method (Breiman et al. 1984). The structure looks like a real tree because it has nodes, branches, and leaves. The initial set is called "the root node." This initial set is recursively split in mutually exclusive subsets or nodes. A test is used on each node in order to split it and, thereby, increase the homogeneity of the different subsets. In binary trees only two branches can come up from each node. On each split the variable that achieves a higher separation among classes is selected. When a stop criterion is reached the majority class is assigned to the example in this node Then it is called a "terminal node" or "leaf." The information nature is not relevant, therefore, classification trees are an excellent method for qualitative variables.

A pruned tree is trained using the 1-SE rule. This rule selects the smallest tree with a cross validation error equal to or less than the minimum error plus a standard deviation. Results are shown exactly as they are given by R program.

```
  > sabi.prune<-prune(sabi.rpart,cp=0.03)
  > printcp(sabi.prune)

  Classification tree:
  rpart(formula = ESTADO ~ ., data = sabi[ind, ], method =
"class",
      cp = 0, minsplit = 1, maxdepth = 30)

  Variables actually used in tree construction:
  [1] lnAT1  PE.PT1

  Root node error: 922/1383 = 0.66667

  n= 1383

          CP nsplit rel error  xerror     xstd
  1 0.374187      0   1.00000 1.05748 0.018395
  2 0.127983      1   0.62581 0.67245 0.020059
  3 0.036876      2   0.49783 0.51302 0.019134
  4 0.030000      3   0.46095 0.50217 0.019035
```

This tree obtains an error of 30.73 percent in the training set. Its confusion matrix is shown in Table 2. The test error is 26.144 percent. Moreover, it is well worth noting that there is not any firm of failed1 neither failed2 classified as healthy, which are the worst errors in this problem.

Table 2 Confusion Matrix of the Pruned Tree in the Training and Test Sets

	Observed Class						
Predicted Class	Pruned tree	Training		30.730 percent	Test		26.144 percent
		Healthy	Failed1	Failed2	Healthy	Failed1	Failed2
	Healthy	345	0	0	42	0	0
	Failed1	83	284	132	6	33	13
	Failed2	33	177	329	3	18	38

The tree structure is as follows, and its graphic can be seen in Fig. 1.

```
> sabi.prune

n= 1383

node), split, n, loss, yval, (yprob)

      * denotes terminal node

 1) root 1383 922 Failed1 (0.33333333 0.33333333 0.33333333)

   2) PE.PT1>=-0.01480628 1038 577 Failed1 (0.44412331 0.11175337 0.44412331)

     4) PE.PT1< 0.6822301 311 126 Failed1 (0.59485531 0.18971061 0.21543408) *

     5) PE.PT1>=0.6822301 727 333 Failed2 (0.37964237 0.07840440 0.54195323)

      10) lnAT1< 6.546879 188  89 Failed1 (0.52659574 0.12765957 0.34574468) *

      11) lnAT1>=6.546879 539 210 Failed2 (0.32838590 0.06122449 0.61038961) *

   3) PE.PT1< -0.01480628 345   0 Healthy (0.00000000 1.00000000 0.00000000) *
```

In this case, the pruned tree only uses two variables. The first one is the PE.PT1 ratio, which is an indebtedness ratio. It is actually selected in two splits as the best discriminant variable. The second is the lnAT1, which, as has been mentioned, is a dummy variable for the firm size. At each node a test is set, examples which satisfy the test being assigned to the left branch, while the rest go to the right. Then a label is assigned to each leaf (healthy, failed1, or failed2) and the number of failed1, failed2, and healthy firms present on it is shown. For instance, in the leaf at the left the class is failed1, and there are 185 failed1 firms, 59 healthy firms, and 67 failed2 firms.

Corporate Failure Prediction by the Boosting Method

Even though the Adaboost.M1 algorithm can use any sort of classification system as individual classifiers, decision trees are used in this application for the task. This election is mainly based on three reasons: (1) decision trees are used in most of boosting applications, (2) they achieve good results, and, finally, (3) classification trees handle qualitative variables quite easily and satisfactory.

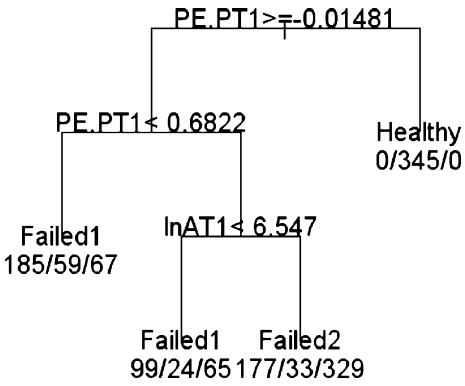

Fig. 1 Structure of the pruned tree

Three functions dealing with Adaboost.M1 are available in the *adabag* package. The first one trains the Adaboost.M1 classifier and assigns a class to the examples of the training set; the second uses a previously trained Adaboost.M1 classifier to predict the class of the cases in a new data set; and the last one allows cross validation to be applied to estimate the error of an Adaboost.M1 classifier. As in any R function, there are a few initial arguments to be set, such as the name of the data frame where the data are stored or the name of the variable that contains the observations class and the predictor variables, the number of individual trees to be used, and the size of these trees.

Using the *adabag* library, a boosting classifier is built with 1,000 trees that have been pruned using maxdepth = 3 to limit the size of the individual tree in each boosting epoch. The maxdepth parameter stops the growth of the tree when the distance between a leaf node and the root node reaches this value. The test error is reduced to 21.57 percent, so there is a reduction of 17.5 percent, compared with the individual tree test error, which is 26.14 percent. Moreover, this important reduction has been achieved keeping the number of failed firms classify as healthy at zero for both types of failure. Table 3 shows the error in the training and test sets.

Because of the nature of this problem, where there are two types of failure, we consider interesting to analyse these results from a binary classification perspective. In the binary case, healthy firms are differentiated from failed firms. This approach is more frequent in corporate failure prediction. The previous confusion matrix in the test set would be in the binary problem as it is shown in Table 4. As can be seen, the

Table 3 Matrix Confusions of the Adaboost.M1 Classifier in the Training and Test Sets

	Observed Class						
Predicted Class	Adaboost.M1 classifier	Training		24.657 percent	Test		21.569 percent
		Healthy	Failed1	Failed2	Healthy	Failed1	Failed2
	Healthy	345	3	0	42	0	0
	Failed1	67	310	94	5	37	10
	Failed2	29	148	367	4	14	41

Table 4 Confusion Matrix for the Test Set Joining Failed1 and Failed2 Classes

	Observed Class	
Predicted Class	Healthy	Failed
Healthy	42	0
Failed	9	102

error would be 9/153=0.0588 (5.88 percent). This is quite a good result because it means an accuracy of 94.12 percent. Moreover, the error in the binary analysis shows that most of the error in the multiclass problem is owed to that failed1, and failed2 classes are quite difficult to differentiate.

Going back to the adaboost.M1 function, it allows the relative importance of the predictor variables to be quantified. Understanding a small individual tree can be easy. However, it is more difficult to interpret hundreds or thousands of trees used which are in the boosting ensemble. Therefore, to be able to quantify the contribution of the predictor variables to the discrimination is a really important advantage. This measure takes into account how many times each variable is selected to realize a split. It makes sense to think that the more important variables will be used in more splits than the less important ones. Table 5 shows all variables arranged from greater to lesser relative importance. In this case, the most outstanding ratios are PE.PT, BAI.AT, and CF.PT, with values at this measure of 23.61, 14.92, and 11.07 percent, respectively. Those variables which are different from financial ratios (NACE1, lnAC, lnAT, and Juridica) have an interesting contribution of 20.98 percent as a whole.

Conclusions

In this study, an ensemble classifier method has been analysed showing, both theoretically and empirically, the improvement in accuracy that this method achieves. As has been seen, Adaboost.M1 is based on building consecutive classifiers on modified versions of the training set generated according to the error rate of the previous classifier,

Table 5 Relative Importance of Variables

Variable	Relative Importance	Variable	Relative Importance
PE.PT1	23.61	V.AC1	2.75
BAI.AT1	14.92	AC.AT1	2.51
CF.PT1	11.07	T.AT1	1.98
lnAC1	7.20	V.FP1	1.76
CNAE1	6.80	AC.PC1	1.66
V.AT1	6.44	JURIDICA	1.60
T.PC1	6.15	IN.AT1	0.83
lnAT1	5.40	FM.V1	0.48
BAI.FP1	4.50	FM.AT1	0.35

while focusing on the hardest examples of the training set. Adaboost.M1, unlike the better known adaboost algorithm, has the advantage of managing multiclass problems.

In the practice application, it has been shown that more than one type of corporate failure can be used, including bankruptcy and temporary receivership firms (failed2), as well as acquired and dissolved firms. Therefore, failed firms are grouped in these two classes. Consequently, the application has worked, unlike is usual, with three classes, where healthy companies have been contrasted versus failed1 and failed2 ones. The Adaboost.M1 method achieves a test error of 21.57 percent in this case. This result shows a reduction of 17.5 percent with respect to the test error of the classification tree (26.14 percent) and confirms that the adaboost.M1 algorithm outperforms decision trees.

The error of the adaboost.M1 in the three class problem can seem to be high, however, the binary analysis shows that failed1 and failed2 firms are properly differentiated from healthy companies. Therefore, a part of the error in the three class problem is owed to the overlap between failed1 and failed2 firms. This shows that acquired firms have a behaviour more similar to failed firms than to healthy firms. Nevertheless, this subject needs to be analyzed in the future.

Since the pioneering works of Beaver (1966) and Altman (1968), many studies have been developed to predict corporate failure using accounting-based variables. However it seems that there might be other variables that can help prediction. These variables can be quantitative or qualitative. In this research, the size of the firm, the activity sector, and the legal structure have demonstrated their usefulness. In fact, these variables, which are not accounting-based, have a joint relative importance of 20.98 percent.

The most outstanding ratios, both for the individual classification tree and for adaboost.M1, deal with indebtedness and profitability. These results are in line with previous studies of corporate failure.

Many important tasks have not been addressed in this research, such as the effect on the joint accuracy of the interdependence of combined classifiers, the behaviour of combination methods in the presence of noisy data, and the use of different basic classifiers for combinations. Consequently, these points offer future lines of research.

Appendix

Table A1 Predictor Variables

Variable	Description	Variable	Description
PE.PT1	Working capital/Sales	V.AC1	Sales/Current assets
CF.PT1	Cash flow/Total debt	T.AT1	Cash/Total assets
lnAC1	Logarithm of current assets	V.FP1	Sales/Permanent funds
CNAE1	NACE code at one digit	AC.PC1	Current assets/Current liabilities
V.AT1	Sales/Total assets	JURIDICA	Legal structure
T.PC1	Cash/Current liabilities	IN.AT1	Net incomes/Total assets
lnAT1	Logarithm of total assets	FM.V1	Working capital/Sales
AC.AT1	Current assets/Total assets	FM.AT1	Working capital/Total assets
BAI.AT1	Earnings before interest and taxes/ Total assets	BAI.FP1	Earnings before interest and taxes/ Permanent funds

References

Alfaro, E., Gámez, M., & García, N. (2006). *adabag: implements adaboost.M1 and bagging*. R package version 1.0. http://www.R-project.org

Altman, E. I. (1968). Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *Journal of Finance, 23*(4), 589–609.

Bauer, E., & Kohavi, R. (1999). An empirical comparison of voting classification algorithm: bagging, boosting and variants. *Machine Learning, 36*, 105–142.

Beaver, W. H. (1966). Financial ratios as predictors of failure. Empirical research in accounting: selected studies. *Journal of Accounting Research, 4(Supplement)*, 71–111.

Breiman, L. (1998). Arcing classifiers. *Annals of Statistics, 26*(3), 801–849.

Breiman, L., Friedman, J. H., Olshen, R., & Stone, C. J. (1984). *Classification and regression trees.* Belmont: Wadsworth International Group.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In J. Kittler, & F. Roli (Eds.). *Multiple Classifier Systems, vol. 1857 of Lecture Notes in Computer Science* (pp. 1–15). New York: Springer.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning* (pp. 148–156). Bari, Italy.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, 55*(1), 119–139.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. *Annals of Statistics, 38*(2), 337–374.

Frydman, H., Altman, E., & Kao, D. (1985). Introducing recursive partitioning for financial classification: The case of financial distress. *Journal of Finance, 40*(1), 269–291.

Kuncheva, L. I. (2004). *Combining pattern classifiers. Methods and algorithms.* New Jersey: Wiley.

Ohlson, J. A. (1980) Financial ratios and the probabilistic prediction of bankruptcy. *Journal of Accounting Research, 18*(1), 5–12.

R Development Core Team (2004). *R: A language and environment for statistical computing.* Viena: R Foundation for Statistical Computing. http://www.R-project.org

Ripley, B. D. (2004). *Rpart: Recursive Partitioning*. R package version 3.1-20. http://www.R-project.org

Valentini, G., & Masulli, F. (2002). Ensembles of learning machines. In M. Marinaro, & R. Tagliaferri (Eds). *Proceedings of the 13th Italian Workshop on Neural Nets, vol. 2486 of Lecture Notes in Computer Science* (pp. 3–19) Berlin Heidelberg New York: Springer.

Wilson, R. L., & Sharda, R. (1994). Bankruptcy prediction using neural network. *Decision Support Systems, 11*(5), 545–557.

Zmijewski, M. (1984). Methodological issues related to the estimation of financial distress prediction models. *Journal of Accounting Research, 22*, 59–86.