Expert Systems with Applications 40 (2013) 6910–6917

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Improving the management of microfinance institutions by using credit scoring models based on Statistical Learning techniques

María-Dolores Cubiles-De-La-Vega [a], Antonio Blanco-Oliver [b,*], Rafael Pino-Mejías [a], Juan Lara-Rubio [c]

[a] Department of Statistics and Operational Research, Faculty of Mathematics, University of Seville, Avda. Reina Mercedes, s/n, 41012 Seville, Spain
[b] Department of Financial Economics and Operations Management, Faculty of Economics and Business Studies, University of Seville, Avda. Ramon y Cajal, 1, 41018 Seville, Spain
[c] Department of Financial Economics and Accounting, Faculty of Economics and Business Studies, University of Granada, Campus Cartuja, s/n, 18071 Granada, Spain

ARTICLE INFO

Keywords:
Decision support systems
Microfinance institutions
Credit scoring
Efficiency
Statistical Learning
Data mining

ABSTRACT

A wide range of supervised classification algorithms have been successfully applied for credit scoring in non-microfinance environments according to recent literature. However, credit scoring in the microfinance industry is a relatively recent application, and current research is based, to the best of our knowledge, on classical statistical methods. This lack is surprising since the implementation of credit scoring based on supervised classification algorithms should contribute towards the efficiency of microfinance institutions, thereby improving their competitiveness in an increasingly constrained environment. This paper explores an extensive list of Statistical Learning techniques as microfinance credit scoring tools from an empirical viewpoint. A data set of microcredits belonging to a Peruvian Microfinance Institution is considered, and the following models are applied to decide between default and non-default credits: linear and quadratic discriminant analysis, logistic regression, multilayer perceptron, support vector machines, classification trees, and ensemble methods based on bagging and boosting algorithm. The obtained results suggest the use of a multilayer perceptron trained in the R statistical system with a second order algorithm. Moreover, our findings show that, with the implementation of this MLP-based model, the MFIś misclassification costs could be reduced to 13.7% with respect to the application of other classic models.

1. Introduction

Microfinance institutions (hereafter, MFIs) offer saving services and small loans (namely microcredits) to those sectors of the population with a very limited access to financial resources. For this reason, the goals and management criteria of the MFIs have less business component and higher social component than those used by new competitors (international commercial banks). The microfinance sector has rapidly grown in the last years, turning into a booming industry. As an example, the number of microfinance institutions grew by 474% in the period 1998–2008, while the number of customers grew by 1048%. This phenomenon has moved a large number of international commercial banks to operate in the microfinance sector. This reinforced interest has increased the competition between the players in this industry, but it is negatively affecting the MFIs. Therefore, the MFIs need to increase their efficiency in all their processes, minimize their costs and control their credit risk if they want to survival a long-term. In particular, credit scoring models may improve this efficiency. Their objective is to assign credit applicants to one of two groups: a 'good credit' group that is likely to repay the financial obligation or a 'bad credit' group that should be denied credit because of a high likelihood of defaulting on the financial obligation (Henley & Hand, 1996).

An appropriate automatic evaluation of the credit applicants offers several important advantages: the cost of credit analysis is reduced, cash flow is improved, faster credit decisions are enabled, the losses are reduced, a closer monitoring of existing accounts is possible, and prioritizing collections are allowed (West, 2000). In this sense, Rhyne and Christen (1999) suggest that credit scoring is one of the most important uses of technology that may affect management of MFIs, and Schreiner (2004) claims that experiments made in Bolivia and Colombia showed that the implementation of credit scoring improved the judgment of credit risk and thus cut, in more than $75,000 per year, costs of MFIs. Nevertheless, and unlike modeling in financial institutions, credit scoring algorithms in microfinance sector have been mostly based on classical statistical techniques, mainly linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), and logistic regression (LR). Some references in this sense are Vigano (1993), Sharma and Zeller (1997), Reinke (1998), Zeller (1998), Vogelgesang (2003), Kleimeier and Dinh (2007), Rayo, Lara, and Camino (2010). However, several authors, for example, Reichert, Cho, and Wagner (1983) and Karels

[*] Corresponding author. Tel.: +34 954 559 875; fax: +34 954 557 570.
E-mail address: aj_blanco@us.es (A. Blanco-Oliver).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.06.031

and Prakash (1987), point out that basic assumptions of LDA and QDA are often violated when applied to credit scoring problems. Other problems usually appearing in credit scoring data sets are the mixed nature of the data (quantitative and qualitative) and the high non-linearity in the association between the target variable and the predictors.

These problems can be faced with Statistical Learning algorithms. Statistical Learning is a framework for machine learning with a strong statistical basis. As it is remarked by Hastie, Tibshirani, and Friedman (2001), a related topic, data mining, is an important element of Statistical Learning. Both terms can be considered as parts of a wider process that was termed Knowledge Discovery from Data (KDD) by Fayad, Piatetsky-Shapiro, and Smith (1996), oriented to identify patterns in data sets.

There are many papers providing empirical evidences supporting these alternative algorithms in credit scoring. West (2000) developed several models applying various kinds of neural networks and he compared them with the classical statistical models (LDA and LR) and some non-parametric methods, such as k-nearest neighbor, kernel density and classification and regression trees (CART). Lee, Chiu, Lu, and Chen (2002) developed a two-stage hybrid credit scoring model using multilayer perceptrons (MLP) and multivariate adaptive regression splines (MARS). Multiple discriminant analysis (MDA) and MLP were compared in Malhotra and Malhotra (2003) to identify potential loans, revealing a better performance for MLP model. Kim and Sohn (2010) implemented support vector machine (SVM) models to predict the default of funded SMEs, comparing their performance with the MLP and LR models. Ince and Aktan (2009) compare the performance of several credit scoring models applied to credit card data set from a Turkish bank. These authors use four statistical methods: multiple discriminant analysis, logistic regression, artificial neural networks (ANN) and classification and regression trees, and suggest that CART obtain the best accuracy performance, following of LR, MDA and ANN.

However, similar works in the microfinance field are still expected to be developed. Following this research line, the main goal of this paper is precisely to build a wide set of credit scoring models for the microfinance institutions inside Statistical Learning framework. An empirical scheme has been adopted for this research, accessing to information of almost 5500 microcredits from a Peruvian Microfinance Institution. This data set was used to build and compare the following supervised classification rules to decide between default and non-default categories: linear and quadratic discriminant analysis, logistic regression, multilayer perceptron, support vector machines, classification trees, and three ensemble methods (bagging, random forests and boosting). According to Witten and Frank (2005), the different data mining methods correspond to different concept description spaces searched with different schemes. Thus, different description languages and search procedures serve some problems well and other problems badly, and that is the cause of the necessity to perform a careful comparison of different data mining techniques.

These classification models are freely available in the R system (R Development Core Team., 2012) which also provides the user with a powerful statistical programming language. Ihaka and Gentleman (1996) present an introduction to the main characteristics of the R system.

The remainder of the paper proceeds as follows. In Section 2, details of the analyzed data set are presented, including a detailed examination of the available variables. Classification models are presented from the point of view of the currently available R implementations in Section 3, where several practical questions associated with their use are also analyzed. In Section 4, the results for the different models are presented and a comparison of them is made. Finally, Section 5 provides the main conclusions of this study.

2. Data description

2.1. The data set

A data set of microcredits from a Peruvian Microfinance Institutions (*Edpyme Proempresa*) has been analyzed. It contains customer information during the period 2003–2008 related to: (a) personal characteristics of borrowers (marital status, sex, etc.); (b) economic and financial ratios of their microenterprise; (c) characteristics of the current financial operation (type interest, amount, etc.); (d) variables related to the macroeconomic context; and (e) any delays in the payment of a fee of microcredit. A previous cleaning of the data set was performed to improve its quality, and therefore abnormal cases, which had the top 1% and the bottom 1% of each variable, were removed. After eliminating missing and abnormal cases, 5451 cases remained. Among them, 2673 (49.03%) were default cases and 2778 (50.97%) were non-default cases. In line with other studies (for example, Schreiner, 2004), a credit is defined as default when it shows a delay in the payment of at least fifteen days.

To perform an appropriate comparison of the classification models the final data set was randomly split into two subsets; a training set of 75% and a test set of 25%. The test sample contains a total of 1363 cases (51.80% failed and 48.20% non-failed). The configuration of parameters of each model was performed through a 10-fold cross-validation procedure, as it will be described in Section 3. Our paper follows the extensive discussion in Hastie et al. (2001) regarding the mechanisms for an appropriate fitting and comparison of classification rules in the Statistical Learning framework.

2.2. Description of input variables

Tables 1–3 show the input variables used in this study. These tables also show the expected sign of the relationship between each input variable and the probability of default. Numerous qualitative variables are considered, following suggestions as Schreiner (2004), who claims that the inclusion of qualitative variables improves the prediction power of models. Moreover, since the default of borrowers has a close relationship with the general economic situation, variables linked to the macroeconomic context are also considered as input variables.

The absence of variables with information about the economic cycle has historically implied a major limitation of financial distress models. Furthermore, the macroeconomic environment is a key factor that directly affects the payment behavior of any borrower. For this reason, the following macroeconomic indicators were computed (Table 3): $\Delta VM_{i,j} = (VM_{i+j} - VM_i)/VM_i$, where $\Delta VM_{i,j}$ is the variation rate of the considered macroeconomic variable *VM*, *i* is the moment of the granting of the loan and *j* is microcredit duration.

Table 1
Description of predictor variables: financial ratios.

Variable	Description	Expected sign
R1	Assets rotation: income sales/total assets	–
R2	Productivity: gross utility/operating costs	–
R3	Liquidity: cash/total asset liquidity	–
R4	Liquidity rotations: cash/income sales × 360	+
R5	Leverage1: total liabilities/(total liabilities + shareholders' total equity)	+
R6	Leverage2: total liabilities/shareholders' equity	+
R7	ROA: Net income/total assets	–
R8	ROE: Net income/shareholders' equity	–

Table 2
Description of predictor variables: non-financial information.

Variable	Description	Expected sign
Zone	Geographical location of the agency or branch. Dummy variable: (0) central zone, (1) outskirts	+
Old	Duration as a borrower of the MFI. Numeric variable	–
Previous_Loan_Grant	Previously granted credits. Numeric variable	–
Loan_Grant	Loans granted in the last year. Numeric variable	–
Loan_Denied	Previously denied loans. Numeric variable	+
Sector	Activity sector of the micro-business. Categorical variable: (0) commerce, (1) agriculture, (2) production, (3) service	±
Purpose	Destination of microcredit. Dummy variable: (0) work capital, (1) fixed asset	+
Mfi_Class	MFI customer classification Dummy variable: (0) normal customer, (1) customer with repayment problems of any sort	+
Total_Fees	Total number of fees paid in credit history. Numeric variable	–
Arrears	Number of arrears. Numeric variable	+
Ave_Arrear	Average (days) of customer default. Numeric variable	+
Max_Arrear	Number of days of major default. Numeric variable	+
Gender	Borrower gender. Dummy variable: (0) male, (1) female	–
Age	Age at time of application. Numeric variable	±
Marital_St	Marital status. Dummy variable: (0) single, (1) family unit	-
Employm_St	Employment Status of borrower. Dummy variable: (0) owner, (1) dependent	±
Guarantee	Guarantee presented. Dummy variable: (0) sworn declaration, (1) real guarantee	+
Currency	Type of currency for loan granted. Dummy variable: (0) Nuevos Soles (ISO code PEN) (1) US Dollar ($)	+
Amount	Amount of microcredit. Numeric variable	–
Duration	Number of monthly fees for applied loan. Numeric variable.	+
Interest_R	Monthly interest rate for microcredit. Numeric variable	+
Forecast	Loan officer forecast: credit situation at expiration. Dummy variable: (0) without problems, (1) with problems	+

Table 3
Description of predictor variables: macroeconomic Indicators.

Variable	Description	Expected sign
GDP	Rate of annual change of Gross Domestic Product (GDP) during loan term	–
CPI	Rate of annual change of Consumer Price Index (CPI) during loan term	+
Empl_R	Rate of annual change of variation of employment rate (ER) during loan term	–
ER	Rate of annual change of variation of exchange rate (ER) PEN-$ during loan term	+
IR	Rate of annual change of interest rate (IR) during loan term	+
SEI	Rate of annual change of stock exchange index (SEI) during loan term	–
Water	Rate of annual change in cost of municipal water during loan term	+
Electricity	Rate of annual change in cost of electricity during loan term	+
Phone	Rate of annual change in cost of telephone consumption during loan term	+

3. Classification models and evaluation measures

3.1. Linear and quadratic discriminant analysis

LDA classification rule is based on a linear combination of the predictors, and it can also be formulated by predicting class 1 if the estimated probability for the class 1 is greater than a threshold probability p_c.

As it is pointed out by Hastie et al. (2001), the direction arising in the LDA rule does not depend on Gaussian assumptions when it is derived via least squares, but the intercept does require Gaussian data. Therefore, the intercept, or equivalently, the threshold p_c could be selected by a *K*-fold cross-validation. Thus, 99 possible values for p_c (0.01,0.02,...,0.99) were considered in our study, and the value minimizing the 10-fold cross-validation classification error set was selected, namely 0.35. This cross-validation search was also performed in the remaining techniques to be explained in this section. LDA was fitted with R function *lda* (Venables & Ripley, 2002) available in the MASS library. Previously, a variable selection process was performed with the function *greedy.wilks* of the package *kĺaR* of R (Weihs, Ligges, Luebke, & Raabe, 2005), based on the Wilḱs lambda statistic.

When the covariance matrices are not assumed to be equal, quadratic discrimination functions are computed. The R function *qda* (Venables & Ripley, 2002) in the MASS library has been used in this study. A similar search for the cut point was also followed for QDA model through the same set of 99 threshold probabilities as in LDA, obtaining 0.99.

3.2. Logistic regression

The LR model was fitted with the *glm* function in R (Venables & Ripley, 2002), which tries to compute the maximum likelihood estimators of the $p + 1$ parameters by an iterative weighted least squares (IWLS) algorithm. Like in LDA, a previous stepwise procedure in order to select the most significant variables was performed. We used the function *step.glm* of R, which applies a forward sequential procedure based on the Akaike Information Criterion. LR can be fully embedded in a formal decision framework, but in order to perform a comparison with the other models, a threshold probability must be specified, what it corresponds in fact to varying the prior classes probabilities. Thus 99 possible values for this threshold probability (0.01,0.02,...,0.99) were also considered, selecting that value minimizing the 10-fold validation error, obtaining 0.58.

3.3. Classification trees

We have used the *rpart* package of R (Venables & Ripley, 2002), which implements the CART methodology as proposed by Breiman, Friedman, Olshen, and Stone (1984). The Gini index (default impurity measure) has been considered as the splitting criterion. Given that large trees can lead to overfitting the data, with a loss in the generalization capability for new data, the user must tune a fundamental parameter: the number of terminal nodes, called the size of the tree. The one-standard-deviation rule was followed, as it can be seen in Maindonald and Braun (2003).

3.4. Multilayer perceptron

The multilayer perceptron (MLP) is the most commonly used type of neural network in business studies, Vellido et al. (1999) and Zhang, Patuwo, and Hu (1998). Several theoretic results support this particular architecture, for example the universal approximate property (Bishop, 1995). A three-layered perceptron was considered, where the output layer is formed by one node which provides the estimation of the probability of default, computed with the logistic activation function $g(u) = e^u/(e^u + 1)$, also used in the hidden layer. Denoting by H the size of the hidden layer, $\{\upsilon_{ih},\ i = 0,1,2,\ldots,p,\ h = 1,2,\ldots,H\}$ the synaptic weights for the connections between the p-sized input and the hidden layer and $\{w_h, h = 0,1,2,\ldots,H\}$ the synaptic weights for the connections between the hidden nodes and the output node, the output of the neural network from a vector of inputs $(x_1,\ldots,x_p)$ is

$$\hat{y} = g\left(w_0 + \sum_{h=1}^{H} w_h g\left(\upsilon_{0h} + \sum_{j=1}^{p} \upsilon_{ih} x_j\right)\right) \tag{4}$$

The output of this model provides an estimation of the probability of default for the corresponding input vector. A final decision can be obtained comparing with a threshold, usually 0.5, thus the decision is default if $\hat{y} > 0.5$.

The MLPs used in this paper have as inputs those variables statistically significant (p-value less than 0.05) in the previous LR model, and the range of each predictor variable was linearly mapped into the $[-1,1]$ interval. Two different programs were used to build the MLP credit scoring models.

The first choice was the freely available R system. The *nnet* R function (Venables & Ripley, 2002) fits single-hidden-layer neural networks by the BFGS procedure, a quasi-Newton method also known as a variable metric algorithm, trying to minimize an error criterion which allows a decay term λ intending to avoid overfitting problems. Let $W = (W_1,\ldots,W_M)$ be the vector of all the M coefficients of the net, and n targets $y_1,\ldots,y_n$, being $y_i = 1$ for default microcredit, and $y_i = 0$ otherwise, are known in the available sample. For classification problems, an appropriate error function is conditional maximum likelihood (or entropy) criterion (Hastie et al., 2001). Thus, the BFGS method is applied to the following problem:

$$\underset{\mathbf{W}}{Min} \sum_{i=i}^{n} (y_i \ln \hat{y}_i + (1 - y_i)\ln(1 - \hat{y}_i)) + \lambda\left(\sum_{i=i}^{M} W_i^2\right) \tag{5}$$

The R implementation of a MLP model requires the specification of two parameters: the size of the hidden layer (H) and the decay parameter (λ), and therefore a 10-fold cross validation search of the size of the hidden layer (H) and the decay parameter (λ) was carried out over a grid defined as $\{1,2,\ldots,20\} \times \{0,0.01,0.05,0.1,0.2,\ldots,1.5\}$.

Neural Network Toolbox (Demuth & Beale, 1997) with Matlab R2010b was the other tool used to fit MLP. This commercial system offers a great variety of learning rules, and the following six main learning algorithms were used: gradient descent, gradient descent with momentum, BFGS quasi-Newton (like R), Levenberg–Marquardt, scaled conjugate gradient and resilient back-propagation. The first is the traditional back-propagation method originally proposed for MLP, while the second rule is a variant based on a momentum term. These two training algorithms require a key parameter, the learning rate. According to practical suggestions in Rumelhart, Hinton, and Williams (1986), learning rate 0.010 was used for gradient descent and gradient descent with momentum. For the second rule, as is recommended by Matlab, the momentum took the value 0.90. The other four methods are suggested in the Matlab documentation for classification problems, being widely known like second-order training algorithms. These six learning rules try to minimize a sum of squared errors (SSE):

$$\underset{\mathbf{W}}{Min} \sum_{i=i}^{n} (y_i - \hat{y}_i)^2 \tag{6}$$

As in R, there remains the problem of selecting H, and it was also chosen through a 10-fold cross-validation search in $\{1,2,\ldots,20\}$ for each learning method.

Matlab allows the use of early stopping in MLP training. This well-known strategy splits the training data set into effective training and validation sets, and the error on the validation set is monitored during training. The Matlab neural nets of this work were trained both with early stopping (25% size) and without early stopping.

The basic parameters of all the fitted MLP models are presented in Table 4.

3.5. Support vector machines

We have employed the *svm* function available in the library *e1071* of the R system Dimitriadou, Hornik, Leisch, Meyer, and Weingessel (2011), which offers an interface to the award-winning C++ implementation, LIBSVM, by Chang and Lin (2011). The data set is described by n training vectors $\{(X_i, y_i)\}$, $i = 1,2,\ldots,n$, where the multi-dimensional vectors X_i contain the predictor features and the n labels $y_i \in \{-1,1\}$ identify the class of each vector. From among the several variants of SVM existing in the library e1071, following Meyer (2012), we have used C-classification with the Radial Basis Gaussian kernel function:

$$K(u, \upsilon) = exp\left(-\gamma|u - \upsilon|^2\right) \tag{7}$$

The primal quadratic programming problem to be solved is:

$$\underset{\mathbf{w},b,\xi}{Min} \quad \frac{1}{2} w^t w + C\sum_{i=1}^{n} \xi_i$$

$$y_i(w^t \phi(X_i) + b) \geqslant 1 - \xi_i \tag{8}$$

$$\xi_i \geqslant 0, i = 1,2,\ldots,n$$

$C > 0$ is a parameter controlling the trade-off between margin and error, and $\sum_{i=i}^{n} \xi_i$ is an upper bound on the sum of distances of the wrongly classified cases to their correct plane.

Two parameters must be tuned: C and γ. The suggestions of Meyer (2012) were followed to select the parameters of the SVM model, and therefore a grid search for C was first defined over the set $\{1,10,20,30,40,50,100,150,\ldots,1000\}$. Secondly, a grid search for γ was conducted in the set $\{0.10,0.15,0.20,\ldots,0.90\}$. This search of C and γ was performed by a 10-fold cross-validation mechanism with the function *tune.svm* in the library *e1071*.

Table 4
Multilayer perceptron models.

Models	Training algorithm	Statistical software	Hidden layer size	Early stopping
MLP 1	Gradient descent	Matlab	14	No
MLP 2	Gradient descent	Matlab	14	Yes
MLP 3	Gradient descent with momentum	Matlab	10	No
MLP 4	Gradient descent with momentum	Matlab	10	Yes
MLP 5	BFGS Quasi-Newton	Matlab	9	No
MLP 6	BFGS Quasi-Newton	Matlab	9	Yes
MLP 7	Levenberg–Marquardt	Matlab	2	No
MLP 8	Levenberg–Marquardt	Matlab	2	Yes
MLP 9	Scaled Conjugate Gradient	Matlab	14	No
MLP10	Scaled Conjugate Gradient	Matlab	14	Yes
MLP11	Resilient	Matlab	9	No
MLP12	Resilient	Matlab	9	Yes
MLP13	BFGS Quasi-Newton	R	10 ($\lambda = 0$)	No
MLP14	BFGS Quasi-Newton	R	3($\lambda = 0.2$)	No

3.6. Bagging

Bagging (Bootstrap Aggregating) is a method proposed by Breiman (1996) to improve the performance of prediction models. Given a classification model, bagging draws B independent samples with replacement from the available training set (bootstrap samples), fits a model to each bootstrap sample, and finally it aggregates the B models by majority voting. Bagging uses to be a very effective procedure when applied to unstable learning algorithms (i.e., "small changes in the data can cause large changes in the predicted values", Breiman, 1996) such as classification and regression trees and neural networks. The R package *ipred* (Peters & Hothorn, 2012), that computes bagged tree models (CTBag), has been used in this study, while two values for B, 50 and 100, have been considered, selecting that one minimizing the 10-fold cross-validation classification error.

3.7. Random forests

Random forests (RF) were proposed by Breiman (2001) as a way to combine many different trees. A number of trees are constructed. Each one is grown over a bootstrap sample of the training data set, and a random selection of variables is considered to choose splits in each node. As in bagging, the trees are combined by majority voting, and out-of-bag estimates can also be computed. One important feature of this ensemble method is the availability of some measures to asess the importance of each variable and to identify outlier observations. Breiman (2001) claimed that RF does not generally overfit, and he showed that Bayes consistency is achieved with a simple version of RF. The R package *randomForest* (Liaw & Wiener, 2002) has been used in our paper. It builds 500 trees by default, a suggested value used in this paper. However, the number of variables to randomly select has been chosen by a 10-fold cross-validation search around the default value (*mtry* = square root of the number of predictors), namely from *mtry* − 5 to *mtry* + 5.

3.8. Boosting

Friedman, Hastie, and Tibshirani (2000) linked AdaBoost and other boosting algorithms to the framework of statistical estimation and additive basis expansion. This point of view is followed in the library *mboost* (Hothorn, Bühlmann, Kneib, Schmid, & Hofner, 2012) in R, as it is described in Bühlman and Hothorn (2007). This library considers the problem of estimating a real-valued function

$$f^{*}(\cdot) = \arg_{f(\cdot)} \min E[\rho(Y, f(X))] \tag{11}$$

where ρ is a loss function. Supposing n training vectors $\{X_i, y_i\}$, $i = 1,2,\ldots,n$, and having selected a base procedure, the generic functional gradient descent algorithm is:

1. Initialize $\hat{f}^{[0]}(\cdot)$ with an offset value. Set $m = 0$.
2. Increase m by 1. Evaluate at $\hat{f}^{[m-1]}(X_i)$ the negative gradient of the loss function:

$$U_i = -\frac{\partial}{\partial f}\rho(Y, f)\bigg|_{f=\hat{f}^{[m-1]}(X_i)}, \quad i = 1,\ldots,n \tag{12}$$

3. Fit the base procedure to predict $\{U_i, i = 1,\ldots,n\}$ from $\{X_i, i = 1,\ldots,n\}$, obtaining $\hat{g}^{[m]}(\cdot)$.
4. Update $\hat{f}^{[m]}(\cdot) = \hat{f}^{[m-1]}(\cdot) + v\hat{g}^{[m]}(\cdot)$.
5. Iterate steps 2–4 until some stopping value M.

Bühlman and Hothorn (2007) point out that the choice of the step-length factor v is of minor importance, as long as it is small, such as $v = 0.1$, and therefore this value was used in this paper. The selection of the other elements of the algorithm drive to different boosting procedures, and the three main methods appearing in Bühlman and Hothorn (2007) have been explored in our study. They share the same base procedure: select the best variable in a simple linear model in the sense of ordinary least squares fitting. This way, the final model $\hat{f}^{[M]}(\cdot)$ is a linear combination of the input variables, and the importance of each predictor can be assessed. Table 5 shows each method, all of them were fitted with the *glmboost* function in the library *mboost*. The target variable Y is 1 for default cases and 0 otherwise, being $p(x) = P[Y = 1/X = x]$. $f^{*}(x)$ is the population minimizer of $\rho(y,f)$. The offset value of step 1 is computed replacing $p(x)$ by the proportion of defaults.

The major tuning parameter of boosting is the number of iterations M. A 10-fold cross-validation search of the value minimizing the empirical loss, in the range 1 to 3000, was performed. Table 5 contains the selected value for M in each boosting algorithm. The decision of Boosting can be expressed as $\hat{p}(x) > p_c$, with threshold $p_c = 0.5$. However, the initial results showed an important imbalance between the success rates in default and non-default cases, with values around 71% in default cases and around 96% in non-default cases. A 10-fold validation search in the range $\{0.001, 0.002, \ldots, 0.999\}$ was also carried out to avoid this problem. Last column of Table 3 displays the value of p_c.

3.9. Evaluation criteria

As is often employed in classification problems, we use the area under the receiver operating curve (AUC) like performance measure of each model. The AUC was computed with the aid of the *ROCR* library available in R (Sing, Sander, Beerenwinkel, & Lengauer, 2009). However, it is well known that the prior probabilities and the misclassification costs should also be considered (West, 2000). It is apparent that the cost associated with a Type I error (a customer with good credit is misclassified as a customer with bad credit) and a Type II error (a customer with bad credit is misclassified as a customer with good credit) are usually very different. According to West (2000), the relative ratio of misclassification costs associated with Types I and II errors must be 1:5, and hence special attention should be paid to Type II errors of all constructed models. The expected misclassification cost (EMC) is defined as follows (West, 2000):

$$\mathrm{EMC} = C_{21}P_{21}\pi_1 + C_{12}P_{12}\pi_2 \tag{13}$$

where π_1 and π_2 are the prior probabilities of good and bad credit populations, P_{21} and P_{12} measures, respectively, the probability of making Type I errors and Type II errors. P_{21} is usually estimated by the proportion of good-credit customers that are misclassified as bad-credit customers, while P_{12} is estimated by the proportion of bad-credit customers that are misclassified as good-credit customers. π_1 and π_2 have been estimated by the proportions of good and bad credits, respectively.

Table 5
Boosting algorithms.

Algorithm	$\rho(y,f)$	$f^{*}(x)$	M	p_c
AdaBoost	$\exp(-(2y-1)f)$	$\log(p(x)/(1-p(x))/2$	1628	0.040
BinomialBoosting	$\log_2(1+\exp(-2(2y-1)f))$	$\log(p(x)/(1-p(x))/2$	2303	0.001
L_2Boosting	$(y-f)^2/f$	$E[Y/X=x] = p(x)$	310	0.395

4. Results and discussion

Table 6 summarizes the results, in terms of AUC, accuracy (percent of cases that were correctly classified), Types I and II errors (expressed as percents) and EMC, on the test set. Table 6 offers a clear decision about the best model. “MLP 14” has the greatest test AUC (0.954), the greatest test accuracy (88.33%), the lowest Type II error (15.30%) and the lowest EMC (0.434). The lowest Type I error is achieved by “MLP 8” (3.70%) but it is accompanied by a higher Type II error. Other MLP model, “MLP 11”, has the same Type II error (15.30%) than “MLP 14” but the AUC is clearly lower. Fig. 1 displays the AUC and EMC for each model. The superiority of “MLP 14” can be confirmed in this figure, where “MLP 14” is in the ideal right bottom zone.

“MLP 14” is a three-layered perceptron, with 20 input nodes, 3 hidden nodes and one output node. The training has been performed with R, using a BFGS quasi-Newton learning rule, and both the hidden layer size and the decay term was selected by 10-fold cross-validation, the value of this last parameter was 0.2.

It can be observed that the highest AUC test and lowest misclassification costs for MLP are obtained when the second-order algorithms are applied. These results suggest that the gradient descendent algorithm is less efficient than the second-order algorithms considered in this study, what is confirmed by the locations of “MLP 1”, “MLP 2”, and “MLP 3”. This finding agrees with many previous works remarking the shortcomings of the traditional gradient descent. However, when the gradient descendent algorithm is implemented with momentum and early stopping the performance, both in terms of AUC test and EMC, is clearly improved (see model “MLP 4” in Table 6, whose location is the same as “MLP 10” in Fig. 1).

From Table 6 and Fig. 1, the classical statistical models, as LDA, QDA and LR are clearly overcome, both in term of AUC and misclassification costs, by other non-parametric methods. Focusing on the parametric models, the AUC of LDA and QDA models are lower than the AUC of the LR model (0.932). This fact is in line with other authors (Lee et al., 2002; West, 2000), which claim that LR outperforms both LDA and QDA. However, LDA shows the greatest test accuracy (86.43%) and QDA has the lowest EMC (0.507) of the all parametric models. Therefore, there is not a clear winner inside the parametric models, as it can also be observed from Types I and II errors.

Table 6
Test values for evaluation criteria.

MODELS	AUC	Test accuracy (%)	Type I errors (%)	Type II errors (%)	EMC
LDA	0.930	86.43	8.52	18.27	0.514
QDA	0.920	85.33	11.72	17.42	0.508
LR	0.932	86.28	5.94	20.96	0.571
MLP 1	0.902	82.80	9.40	24.40	0.677
MLP 2	0.912	84.10	8.20	22.90	0.633
MLP 3	0.901	82.60	15.30	21.50	0.630
MLP 4	0.946	87.70	7.60	16.70	0.469
MLP 5	0.908	86.60	11.00	15.70	0.460
MLP 6	0.943	87.50	7.60	17.10	0.479
MLP 7	0.939	86.30	4.40	22.40	0.601
MLP 8	0.941	86.60	3.70	22.40	0.598
MLP 9	0.915	84.40	12.60	18.30	0.535
MLP 10	0.946	87.70	7.60	16.70	0.469
MLP 11	0.939	86.90	10.70	15.30	0.448
MLP 12	0.936	86.80	8.50	17.60	0.497
MLP 13	0.924	84.96	6.68	22.81	0.623
MLP 14	0.954	88.33	7.76	15.30	0.434
SVM	0.936	86.35	8.68	18.27	0.515
CART	0.898	85.91	11.57	16.43	0.481
CTBag	0.939	86.57	11.11	15.58	0.458
RF	0.941	87.45	8.22	16.57	0.469
AdaBoost	0.930	86.35	5.93	20.82	0.568
BinomialBoosting	0.932	86.79	6.69	19.26	0.531
L_2Boosting	0.923	86.57	10.81	15.87	0.463

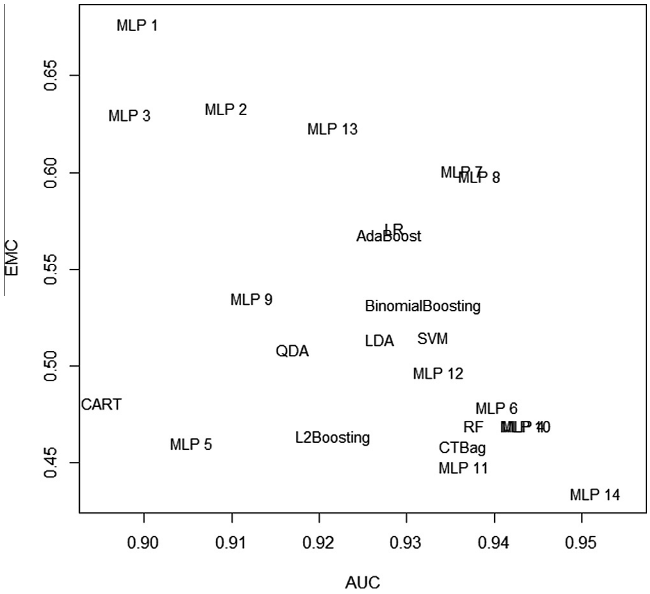

Fig. 1. AUC and EMC for each model.

Table 6 also shows that CART and some MLPs are not effective in this study, obtaining a low AUC, but the EMC and AUC of CART (in particular, the second measure) are improved when Bagging is applied (CTBag and RF, their performance measures are very similar as it can be seen in Fig. 1), a fact that agrees with the theoretical and empirical properties of this ensemble model. RF has the greatest AUC inside both bagging approaches, whereas CTBag has a lower expected misclassification cost.

Last line of Table 6 and Fig. 1 show that a boosting algorithm based on a squared loss function (L_2Boosting) outperforms the other two Boosting algorithms, apparently with more appropriate loss functions for classification problems. SVM has greater AUC value, but its EMC is worse than L_2Boosting. In fact, SVM and LDA have offered similar test measures.

Based in the previous results, we conclude, in line with other authors (for example, see Chen, Härdle, & Moro, 2011; Ince & Aktan, 2009), that, in general, not only do non-parametric models have a greater AUC but also lower misclassification costs than the classical approaches. Moreover, despite the disadvantages that the MLP method includes: (a) its black-box nature, which renders the resulting model very difficult to interpret; and, (b) its long training process in designing the topology of the optimal network, we consider MFIs should use this method instead other models since even a minor improvement in predictive accuracy of the MLP credit scoring model is of critical value. Just a mere 1% improvement in accuracy would reduce losses in a large loan portfolio and save millions of dollars (West, 2000). The differences, in terms of the misclassification costs, between the best MLP (MLP 14) with respect to the other models, vary from 2.4% of the CTBag method to 13.7% of the LR model. That is, the implementation of neural network approaches help to reduce the MFI losses significantly, and therefore, provides a way to obtain a competitive advantage over other MFIs which fail to implement this methodology. Moreover, the use of credit-scoring models provides the MFI with further relevant management advantages, such as the possibility of adopting the Basel II Internal rating-based (IRB) approach which enables MFIs to attain more risk-sensitive capital requirements and to adjust interest rates to the risk of each borrower.

5. Conclusions

In this paper, an appropriate solution is offered so that the MFIs can benefit from all the positive aspects that the implementation of the credit scoring systems involves, such as the increase in efficiency, profitability and market share, reduction of costs and losses, and professional-image management.

The results of Section 4 let to extend for microcredit framework a set of findings agreeing with previous works in credit risk analysis as West (2000), Malhotra and Malhotra (2003), Min and Lee (2005). Therefore, it can be expected for the non-parametric approaches a higher AUC and a lower EMC than those offered by the traditional LDA, QDA and LR methods for microcredits. After examining the wide set of models that were fitted in this paper, a MLP model trained inside the free statistical environment R is suggested to assess the success or default of credits in microfinance industry.

A possible reluctance to use a multilayer perceptron in credit scoring could be explained by some shortcomings as its black box nature, being very difficult to interpret the resulting model, and its long training process in designing the optimal networḱs topology. However, despite of these disadvantages, these models could clearly benefit the microfinance industry because of even a small improvement in predictive accuracy of a default prediction model is critical. According to West (2000), a 1% improvement in accuracy would reduce losses in a large loan portfolio and save millions of dollars. That is, the implementation of a classification model as in this paper supposes that the MFIs reduce their losses in terms of millions of dollars, and therefore provides a way for the MFIs to achieve a competitive advantage over their competitors (mainly commercial banks), since it constitutes a key to an increasingly constrained environment. Therefore, it is worth a careful comparison of Statistical Learning methods as in this work to

Table A.1
Statistical description of quantitative independent variables.

Variable	Failed		Non-Failed	
	Mean	Standard deviation	Mean	Standard deviation
R1	0.7637	0.8055	0.8436	0.8528
R2	3.9421	4.8284	3.8881	6.8548
R3	0.0683	0.0689	0.1448	3.2438
R4	0.1301	0.1368	0.1654	1.9812
R5	0.1421	0.1617	0.1196	0.1474
R6	0.2242	0.3227	0.1810	0.2789
R7	0.1531	0.1764	0.1771	0.2756
R8	0.1799	0.2015	0.2012	0.2911
Old	2.3468	1.5110	2.2397	1.5099
Previous_Loan_Granted	5.3900	5.0040	5.0600	4.6940
Loan_Granted	3.4600	2.3040	4.3400	2.3400
Loan_Denied	0.3200	0.5380	0.3300	0.5360
Mfi_Class	0.3500	0.4770	0.1100	0.3110
Total_Fees	36.1800	25.8510	31.7100	22.8390
Arrears	13.0400	10.7870	13.3400	11.1700
Ave_Arrear	8.0000	8.1510	6.8600	6.4340
Max_Arrears	20.2000	27.7650	16.5500	21.5030
Age	43.0175	10.6148	42.5628	10.4770
Amount	0.7338	0.6548	0.6458	0.5998
Duration	8.1100	4.7950	7.0300	3.5520
Interest_R	4.9242	0.9183	5.1255	0.8801
GDP	8.8985	29.7134	4.8139	26.3989
CPI	2.6377	2.2101	3.1247	2.1318
Empl_R	3.5702	10.6827	2.8671	9.6861
ER	−2.4123	4.4517	−5.5607	3.8899
IR	5.9631	13.9525	12.1717	11.7493
SEI	44.5991	32.4527	49.5322	33.3754
Water	2.4576	3.7483	3.1681	4.2243
Electricity	3.6054	12.2598	8.5162	10.4552
Phone	−7.1809	8.0019	−1.7179	3.8308

Table A.2
Statistical description of qualitative independent variables.

Variable	Categories	Failed (%)	Non-failed (%)
Zone	Centre	46.94	53.06
	Outskirts	55.84	44.16
Sector	Commerce	48.53	51.47
	Agriculture	60.68	39.32
	Production	53.22	46.78
	Service	54.31	45.69
Purpose	Work capital	47.07	52.93
	Fixed asset	77.51	22.49
Gender	Male	51.32	48.68
	Female	50.71	49.29
Marital_St	Single	50.73	49.27
	Family unit	51.06	48.94
Employm_St	Owner	50.81	49.19
	Dependent	70.73	29.27
Guarantee	Sworn declaration	58.50	41.50
	Real guarantee	43.47	56.53
Currency	PEN	89.30	92.10
	$	10.70	7.90
Forecast	Without problems	42.94	57.06
	With Problems	97.27	2.73

Table A.3
Significant variables using linear discriminant analysis.

Linear discriminant analysis model	
Variable[a]	Coefficient
Forecast	2.2062*
ER	0.1684*
CPI	−0.0956*
Total_Fees	0.0125*
Arrears	−0.0232*
Mfi_Class	0.7577*
Guarantee	−0.2508*
Duration	−0.0684*
IR	−0.0461*
Empl_R	−0.0290*
Electricity	−0.0125*
Purpose	0.3559*
SEI	0.0040*
GDP	−0.0052*
Zone	0.1412*
R8	−0.3811*
Max_Arrears	−0.0022*
R2	0.0077*

[a] $^{***}p$-value <0.001; $^{**}p$-value <0.01; $^{*}p$-value <0.05.

Table A.4
Significant variables using logistic regression.

Logistic regression model	
Variable[a]	Coefficient
Forecast	4.2624***
ER	0.3477***
Total_Fees	0.0221***
Arrears	−0.0449***
Mfi_Class	1.2592***
Guarantee	−0.6117***
IR	−0.1011***
Empl_R	−0.0247**
Purpose	0.6048**
GDP	−0.0235***
Zone	0.4209***
Water	0.0346*
Duration	−0.1275***
Intercept	0.2685

[a] $^{***}p$-value <0.001; $^{**}p$-value <0.01; $^{*}p$-value <0.05.

provide MFIs with all the advantages of automatic credit scoring systems, such as the increase in efficiency, profitability and market share, reduction of costs and losses, and professional-image management. Hence MFIs will have more chance to compete with commercial banks by using these advanced risk-management tools.

Appendix A

Tables A.1–A.4.

References

Bishop, C. M. (1995). *Neural networks for pattern recognition*. New York: Oxford University Press.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*, 123–140.

Breiman, L. (2001). Random forests. *Machine Learning, 45*, 5–32.

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

Bühlman, P., & Hothorn, T. (2007). Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science, 22*, 477–505.

Chang, C. C., & Lin, C. J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology, 2*, 1–27.

Chen, S., Härdle, W. K., & Moro, R. A. (2011). Modeling default risk with support vector machines. *Quantitative Finance, 11*(1), 135–154.

Demuth, H., & Beale, M. (1997). *Neural network toolbox for use with Matlab*. User's guide: The Math Works Inc.

Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., & Weingessel, D. (2011). *e1071: Misc functions of the department of statistics (e1071) TU Wien*. R package version 1.6. Available at <http://CRAN.R-project.org/package=e1071>.

Fayad, U., Piatetsky-Shapiro, G. P., & Smith, G. (1996). From data mining to knowledge discovery in databases (a survey). *AI Magazine, 3*, 37–54.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting (with discussion). *Annals of Statistics, 28*, 337–407.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The elements of statistical learning*. New York: Springer.

Henley, W. E., & Hand, D. J. (1996). A k-nearest neighbor classifier for assessing consumer credit risk. *Statistician, 44*, 77–95.

Hothorn, T., Bühlmann, P., Kneib, T., Schmid, M., & Hofner, B. (2012). *mboost: Model-Based Boosting*. R package version 2.1-2. Available at <http://CRAN.R-project.org/package=mboost>.

Ihaka, R., & Gentleman, R. (1996). A language for data analysis and graphics. *Journal of Computational and Graphical Statistics, 5*, 299–314.

Ince, H., & Aktan, B. (2009). A comparison of data mining techniques for credit scoring in banking: A managerial perspective. *Journal of Business Economics and Management, 10*(3), 233–240.

Karels, G., & Prakash, A. (1987). Multivariate normality and forecasting of business bankruptcy. *Journal of Business Finance Accounting, 14*, 573–593.

Kim, H. S., & Sohn, Y. (2010). Support vector machines for default prediction of SMEs based on technology credit. *European Journal of Operational Research, 201*, 838–846.

Kleimeier, S., & Dinh, T. A. (2007). Credit scoring model for Vietnam's retail banking market. *International Review of Financial Analysis, 16*, 471–495.

Lee, T. S., Chiu, C. C., Lu, C. J., & Chen, I. F. (2002). Credit scoring using the hybrid neural discriminant technique. *Expert Systems with Applications, 23*, 245–254.

Liaw, A., & Wiener, M. (2002). *Classification and Regression by random Forest, R News*, 2, 18-22. Available at <http://CRAN.R-project.org/doc/Rnews/>.

Maindonald, J., & Braun, J. (2003). *Data analysis and graphics using R. An example-based approach*. Cambridge: Cambridge University Press.

Malhotra, R., & Malhotra, D. K. (2003). Evaluating consumer loans using neural networks. *Omega-The International Journal of Management Science, 31*, 83–96.

Meyer, D. (2012). *Support Vector Machines. The Interface to libsvm in package e1071*. Available at <http://cran.r-project.org/web/packages/e1071/vignettes/svmdoc.pdf>.

Min, J. H., & Lee, Y. C. (2005). Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert Systems with Applications, 28*, 603–614.

Peters, A., & Hothorn, T. (2012). *ipred: Improved Predictors*. R package version 0.8-13. Available at <http://CRAN.R-project.org/package=ipred>.

R Development Core Team. (2012). *R: A language and environment for statistical computing*. Wien: R Foundation for Statistical Computing, Vienna. Available at <http://www.R-project.org>.

Rayo, S., Lara, J., & Camino, D. (2010). A credit scoring model for institutions of microfinance under the Basel II Normative. *Journal of Economics, Finance and Administrative Science, 15*, 89–124.

Reichert, A. K., Cho, C. C., & Wagner, G. M. (1983). An examination of the conceptual issues involved in developing credit-scoring models. *Journal of Business and Economic Statistics, 1*, 101–114.

Reinke, J. (1998). How to lend like mad and make a profit: A micro-credit paradigm versus the start-up fund in South Africa. *Journal of Development Studies, 34*, 44–61.

Rhyne, E., & Christen, R. (1999). *Microfinance enters the marketplace*. USAID Microenterprise Publications.

Rumelhart, D. E., Hinton, D. E., & Williams, R. J. (1986). *Learning internal representations by error propagation in parallel distributed processing*. Cambridge, MA: MIT Press.

Schreiner, M. (2004). Scoring arrears at a microlender in Bolivia. *Journal of Microfinance, 6*, 65–88.

Sharma, M., & Zeller, M. (1997). Repayment performance in group-based credit programs in Bangladesh: An empirical analysis. *World Development, 25*, 1731–1742.

Sing, T., Sander, O., Beerenwinkel, N., & Lengauer, T. (2009). *ROCR: Visualizing the performance of scoring classifiers*. R package version 1.0-4. Available at <http://CRAN.R-project.org/package=ROCR>.

Vellido, A., Lisboa, P. J. G., & Vaughan, J. (1999). Neural network in business: A survey of applications (1992–1998). *Expert Systems with Applications, 17*, 51–70.

Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with S-PLUS*. New York: Springer.

Vigano, L. A. (1993). Credit scoring model for development banks: An African case study. *Savings and Development, 17*, 441–482.

Vogelgesang, U. (2003). Microfinance in times of crisis: The effects of competition, rising indebtedness, and economic crisis on repayment behavior. *World Development, 31*, 2085–2114.

Weihs, C., Ligges, U., Luebke, K., & Raabe, N. (2005). KlaR. Analyzing German business cycles. In D. Baier, R. Decker, & L. Schmidt-Thieme (Eds.), *Data analysis and decision support* (pp. 335–343). Berlin: Springer-Verlag.

West, D. (2000). Neural network credit scoring models. *Computers and Operations Research, 27*, 1113–1152.

Witten, I. H., & Frank, E. (2005). *Data mining, practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann.

Zeller, M. (1998). Determinants of repayment performance in credit groups: The role of program design, intra-group risk pooling, and social cohesion. *Economic Development and Cultural Change, 46*, 599–620.

Zhang, G. P., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting, 14*, 35–62.