International Business Research; Vol. 9, No. 7; 2016
ISSN 1913-9004 E-ISSN 1913-9012
Published by Canadian Center of Science and Education

Comparing the Performance of Different Data Mining Techniques in Evaluating Loan Applications

Arash Riasi[1], Deshen Wang[1]

[1]Institute for Financial Services Analytics, University of Delaware, Newark, DE, USA

Correspondence: Arash Riasi, Alfred Lerner College of Business & Economics, Amstel Ave., Newark, DE, 19716, USA. Tel: +1 (302) 898-6249. E-mail: riasi@udel.edu

Received: April 14, 2016 Accepted: May 5, 2016 Online Published: June 15, 2016

doi:10.5539/ibr.v9n7p164 URL: http://dx.doi.org/10.5539/ibr.v9n7p164

Abstract

This study compares the performance of various data mining classifiers in order to find out which classifiers should be used for predicting whether a loan application will be approved or rejected. The study also tries to find the data mining classifiers which have the best performance in predicting whether an approved loan applicant will eventually default on his/her loan or not. The study was performed using a sample of 971 loan applicants. The results indicated that the best data mining classifier for predicting whether a loan applicant will be approved or rejected is LAD Tree, followed by Rotation Forest, Logit Boost, Random Forest, and AD Tree. It was also found that the best classifier for predicting whether an approved applicant will default on his/her loan is Bagging, followed by Simple Cart, J 48, J 48 graft, END, Class Balance ND, Data Near Balance ND, ND, and Ordinal Class Classifier.

Keywords: data mining, classification, performance of classifiers, loan application, default prediction, loan approval

1. Introduction

Data mining can be classified into four different categories, namely association rule mining, classification and prediction, clustering analysis, and sequential patterns and time-series mining (Han & Kamber, 2001; Zhang & Zhou, 2004). When using data mining methods, we should consider several issues. For example, we should be aware that the data mining method which we choose needs to take ultimate applications into account, it should be dependent upon the characteristics of our data set, and it should take advantage of domain models (Zhang & Zhou, 2004). Therefore, Zhang and Zhou (2004) suggest that the three dimensions of data mining in financial application are data, applications, and finance/accounting models. Many researchers have applied data mining techniques to financial topics including: credit scoring (Barney et al., 1999; Desai et al., 1996; Glorfeld, 1996; Jagielska & Jaworski, 1996; Lee et al., 2002; Lee et al., 2006; Piramuthu, 1999; Piramuthu et al., 1994; Sinha & May, 2004; West, 2000), credit risk assessment (Doumpos et al., 2002), bankruptcy prediction (Jain & Nag, 1997; Kim & McLeod, 1999; Ryu & Yue, 2005; Shin & Lee, 2002; Sung et al., 1999; Wilson & Sharda, 1994), fraud detection (Brause et al., 1999; Chan et al., 1999; Han & Kamber, 2001; Iba & Sasaki, 1999; Kirkos et al., 2007), portfolio management (Hung et al., 1996), and financial performance prediction (Lam, 2004).

Choosing the appropriate data mining technique for analyzing financial data is an important decision. Comparing the performance of different data mining techniques can be one way to make this decision, but we should be aware that a single data mining technique may not always be the best technique for analyzing all types of financial data. In some cases, it might be a good idea to use hybrid systems that integrate various data mining techniques (Zhang & Zhou, 2004). The present study is focused on using data mining classifiers for helping banks to make better decisions about loan applications. Particularly, we are trying to find out which data mining classifier has the best performance in evaluating loan applications.

2. Literature Review

The most popular data mining algorithms used in business and finance research are artificial neural networks (Ansari & Riasi, 2016a; Fish et al., 2004; Hamid & Iqbal, 2004; Jain & Nag, 1997; Mostafa & El-Masry, 2013; Refenes et al., 1994; Saad et al., 1998; Sung et al., 1999; Walczak, 1999; Wilson & Sharda, 1994; Yoon et al., 1993; Zhang & Zhou, 2004), genetic algorithms (Ansari & Riasi, 2016b; Fish et al., 2004; Iba & Sasaki, 1999; Olaru & Purchase, 2014; Shin & Lee, 2002; Zhang & Zhou, 2004), statistical inference (Han & Kamber, 2001; Refenes et al., 1994; Tseng et al., 2001; Yoon et al., 1993), rule induction (Bose & Mahapatra, 2001; Wu et al., 1998; Zhang & Zhou, 2004), decision trees

(Brandão et al., 2005; Kim et al., 2001; Thomassey & Fiordaliso, 2006), and data visualization (Bose & Mahapatra, 2001, Chang et al., 2016; Grierson et al., 2015; Hachaj, 2014; Jahangirian et al., 2011; Pérez-Montoro & Nualart, 2015). A neural network is a multilayer perceptron with simple connections between different components (Ansari & Riasi, 2016a). In each layer, one or more processing unit(s) called artificial neurons or nodes are present which perform a simplified version of what human brain's neurons do. The behavior of the neural network depends on the relationships and connections among individual components of the network (Ansari & Riasi, 2016a; Mirghafoori et al., 2010). According to Zhang and Zhou (2004), neural networks have very high computation complexity and are highly flexible. Neural networks have been used by researchers for bankruptcy prediction (Jain & Nag, 1997; Sung et al., 1999; Wilson & Sharda, 1994), stock market prediction (Refenes et al., 1994; Saad et al., 1998; Walczak, 1999; Yoon et al., 1993), and portfolio management (Hung et al., 1996). Genetic algorithms are iterative processes based on evolutions which initiate from a population of randomly generated individuals with the ultimate goal of finding comprehensive optimized solutions (Holland, 1975). Genetic algorithms also mimic the process of natural selection and are based on the idea that the genetic pool of a specific population contains the solution to our problem (Zhang & Zhou, 2004). According to Zhang and Zhou (2004), genetic algorithms have very high computation complexity and have low accessibility. Genetic algorithms have been used by various researchers for bankruptcy prediction (Back et al., 1996; Shin & Lee, 2002), stock market prediction (Iba & Sasaki, 1999), and fraud detection (Iba & Sasaki, 1999). Statistical inference is defined as the process of drawing conclusions based on data (Bullard, 2006). According to Zhang and Zhou (2004), discriminant analysis, factor analysis, principal component analysis (PCA), and regression models have been frequently used for identifying the influential variables in financial problems or to find relationships between disparate variables and financial markets. Statistical inference has been used by various researchers for predicting the stock market (Refenes et al., 1994; Yoon et al., 1993), foreign exchange market forecasting (Tseng et al., 2001), and fraud detection (Han & Kamber, 2001). Rule induction techniques produce a set of if-then rules which are extracted from a set of observations and represent significant patterns in the data set which help to create models for prediction (Zhang & Zhou, 2004). Algorithms that produce decision trees are among the most commonly used types of rule induction (Ansari & Riasi, 2016c; Zhang & Zhou, 2004). Rule induction techniques have low flexibility, but very high interpretability (Zhang & Zhou, 2004). Finally, data visualization techniques are implemented in order to make large amount of data easily digestible by using graphics. Data visualization methods are commonly used by financial services firms and researchers in order to better present financial data.

Lee et al. (2006) studied the performance of credit scoring by using classification and regression tree (CART) and multivariate adaptive regression splines (MARS). They found that credit scoring models built by using CART and MARS have higher correct classification rates in both the testing and validation samples compared to credit scoring models produced by linear discriminant analysis (LDA), logistic regression, neural networks, and support vector machines (SVM) methods. Therefore, it can be concluded from their results that CART and MARS provide efficient alternatives for LDA, logistic regression, neural networks, and SVM in order to perform credit scoring modelling. Results of Lee et al. (2004) also revealed that CART and MARS have lower Type II errors compared to LDA, logistic regression and neural networks. Sinha and Zhao (2008) applied data mining classification methods to indirect bank lending. They studied whether the incorporation of domain knowledge improves classification performance or not. In order to do so, an expert system which captures a lending expert's knowledge of rating a borrower's credit was used. The findings from their study indicated that in the absence of credit rating knowledge, if the cost ratio is 1, decision table method has the highest mean misclassification cost and naive Bayes has the lowest mean misclassification cost. They also found that in the absence of credit rating knowledge, if the cost ratio is 5, naive Bayes method has the highest mean misclassification cost and J48 decision tree has the lowest mean misclassification cost. Overall, they found that in the absence of credit rating knowledge, decision table has the highest mean misclassification cost and SVM has the lowest mean misclassification cost. Sinha and Zhao (2008) concluded that appropriate choice of a data mining method, along with the incorporation of domain knowledge, could translate to substantial monetary benefits for a bank.

3. Research Questions

With the rapid growth of credit card industry in developed countries, large amounts of consumers' credit data are collected everyday by the credit department of the banks (Huang et al., 2007) and credit card companies. These valuable credit information can be used to determine the credit scores of customers. Credit scoring models have been broadly used in recent years in order to evaluate the customers' creditworthiness (Thomas, 2000). Unlike many countries which use credit scores as an important decision criteria for approving or rejecting loan applications, Iranian banks do not use a credit scoring system because there exists no formal credit score for individuals in Iran. Additionally most Iranian banks do not use data mining techniques and computerized systems for making loan decisions and/or to evaluate the creditworthiness of loan applicants; instead, most banks have their own loan committees which decide whether to accept or reject loan applications. The absence of credit scoring systems and computerized loan evaluations makes the decision making process very difficult for the banks and can increase the degree of error. Therefore, using data mining techniques can be a good strategy for evaluating the loan applicants and predicting whether they will eventually default on their loans or not. Perhaps, one reason that there is no credit scoring system in Iran is that there is no credit card company and none of the Iranian banks currently issue credit cards. Iranian banks only issue debit cards to their customers which are not a good resource for calculating credit scores. There are two reasons that Iranian banks do not issue credit cards. The first reason is that there are regulations that restricts them from issuing credit cards and the second reason is that Iranian banks do not have access to state of the art data centers and other facilities which are necessary for storing and analyzing credit data. Data mining techniques have contributed to the field of information science to a large extent (Chen & Liu, 2004) and they can be used for constructing efficient credit scoring models (Huang et al., 2007). This study intends to find out which data mining technique performs the best in order to predict whether a loan application will be approved or rejected. We are also trying to find the best data mining technique for predicting whether an approved loan applicant will eventually default on his/her loan or not. The study uses data from customers of an Iranian bank.

This study has 3 major contributions to the existing literature in this field. First, unlike previous studies which used only one or two performance measure for comparing the performance of various data mining techniques, we use 11 different performance measures in our study. Second, we compare the performance of 57 different data mining techniques on our data set while the majority of previous studies only compared the performance of few data mining techniques. For instance, Sinha and Zhao (2008) only used misclassification cost and Area Under the ROC Curve (AUC) as their performance measure, and compared the performance of 7 data mining techniques. Furthermore, Lee et al. (2006) used accuracy, type I error and type II error as their performance measures for comparing the performance of 5 data mining techniques. The third contribution of this study is that we compare the performance of data mining classifiers not only to predict whether a loan applicant will be approved or rejected but also to predict whether an approved loan applicant will default on his or her loan within 5 years after obtaining the loan or not. As far as we investigated, previous studies in this field only focused on performing one of these two predictions and the majority of them only tried to predict whether a loan applicant will be approved or rejected. Therefore, it is fair to claim that the current study is one of the most comprehensive studies which have ever been performed in order to evaluate the performance of data mining techinques using a bank's loan data set.

4. Methodology

4.1 Data

Our sample for this study contained 971 loan applicants of an Iranian Bank. The data set was completely anonymized in order to respect the privacy of loan applicants. The data set contained various demographic and financial information from the loan applicants. This information is used by the bank's loan committee members in order to make a decision on each loan application. Our independent variables from the data set include gender, age, annual income, marital status, education, home ownership status, years with current employer, loan amount, loan duration, loan purpose, number of other loans currently in progress, and total monthly payment on other loans currently in progress. We also have two dependent variables. Our first dependent variable indicates the decision on the loan application (i.e. approve or reject), and our second dependent variable indicates whether an approved loan applicant defaulted on his or her loan within 5 years after obtaining the loan or not. Figure 1 depicts the characteristics of loan applicants in our data set. From the 971 loan applicants in the data set, the application of 533 individuals (i.e., 54.89 % of total applicants) were approved and the loan applications of 438 loan applicants (i.e., 45.11 % of total applicants) were rejected. Among the 533 approved applicants, 436 individuals (i.e., 81.80 % of total approved applicants) did not default on their loans within 5 years after receiving their loans while 97 individuals (i.e., 18.20 % of total approved applicants) defaulted on their loans within 5 years after receiving the loan.

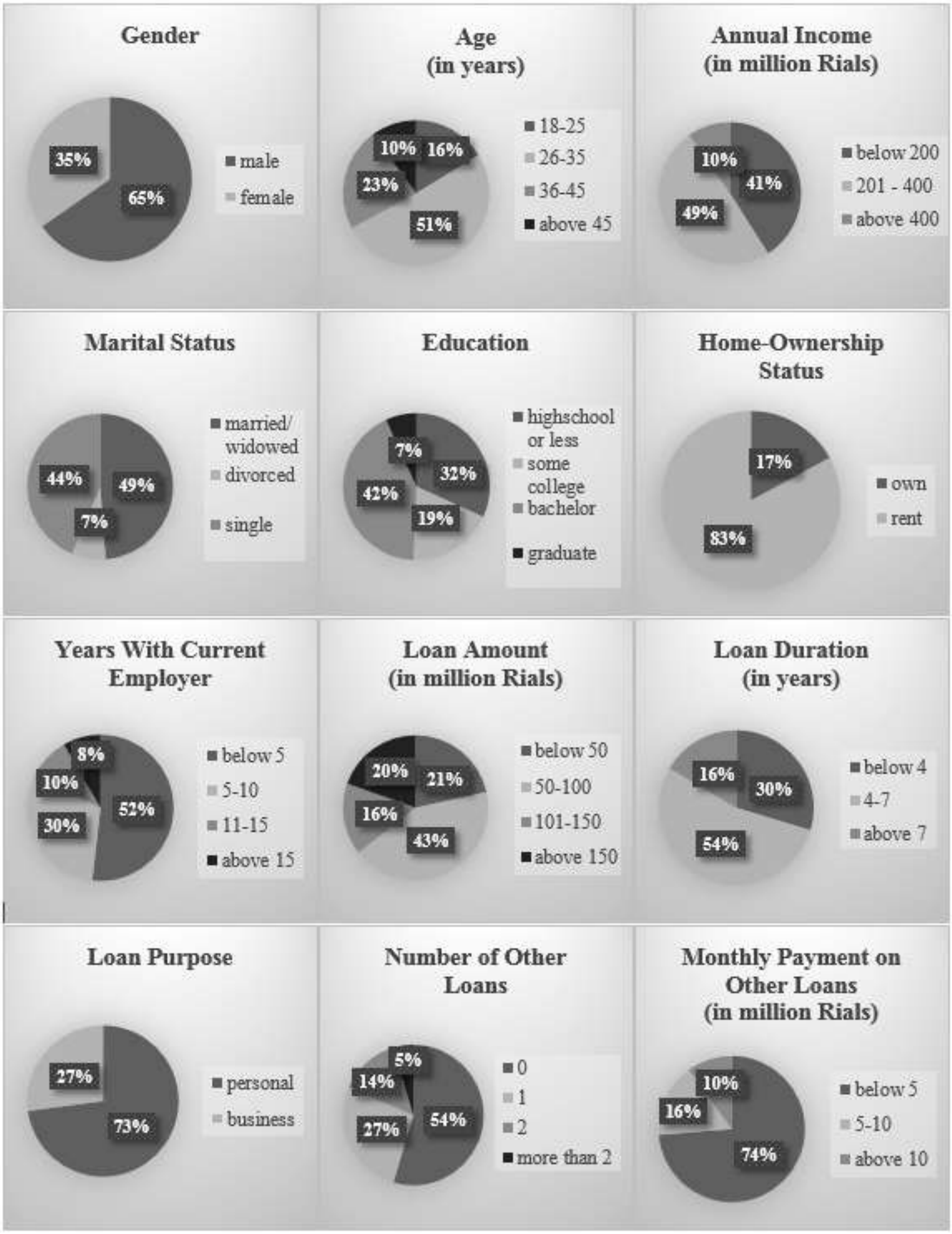

Figure 1. Characteristics of loan applicants

4.2 Classifiers

In this study we used 57 different classifiers and 11 performance measures available in Weka data mining software. Weka is a collection of machine learning algorithms for data mining tasks and contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization (Machine Learning Group at the University of Waikato, 2015). The 57 classifiers which we used in our study are categorized into 6 different groups by Weka. These six groups are: Bayes, rules, meta, functions, lazy, and trees. In the first step, we ran these classifiers on the entire dataset (i.e., 971 instances) using the “decision on the loan application” (i.e., approve or reject) as the class attribute. In the second step we removed the loan applicants whose loan applications were rejected (i.e., 438 loan applicants) and then ran all 57 classifers on the approved loan applicants (i.e., 533 loan applicant) using the “applicant default status” as the class attribute. Figure 2 depicts our research model. According to this model, loan applicants’ data are used as inputs for each of the 57 data mining classifiers. The outputs of the model are the 11 performance measures.

Figure 2. Research model

Bradley (1997) recommended that there should be no attempt to tune the classification methods to a specific problem, in order to reduce the negative effects of any bias in the empirical comparison. Therefore, we used different classifiers using the default settings of Weka in order to minimize the bias in comparing the performance of disparate dara mining classifiers. We also used 10-fold cross-validation to estimate the performance of learned classifiers. Cross-validation is a statistical method for evaluating and comparing learning algorithms which divides the data into two groups: one group is used to learn or train a model and the other group is used to validate the model (Refaeilzadeh et al., 2009). 10-fold cross-validation randomly divides the data set into 10 subsets with equal size. Then it runs 10 separate experiments and in each of these experiments one of the subsets is used for testing and the other 9 subsets are used for training. In order to calculate the success rate of the data mining method, one should then calculate the average success rate of the 10 experiments. The reason that we used 10-fold cross-validation is that this method has been widely used in previous studies and it has been proven that it is the best method to use for model selection even if computational power allows using more folds (Kohavi, 1995).

4.2.1 Meta Classifiers

According to Stolfo et al. (1997), meta-learning is defined as “a unifying and scalable solution that improves the efficiency and accuracy of inductive learning when applied to large amounts of data in wide area computing networks for a range of different applications”. Meta-learning applies learning programs to a group of independent and inherently distributed databases in parallel, in order to compute a number of independent classifiers (Stolfo et al., 1997). Each base-learner creates a base classifier and the meta-learner creates a meta-classifier (Chan & Stolfo, 1993). The goal of a meta-learner is not to choose the best base classifier; instead it tries to combine different classifiers (Chan & Stolfo, 1993). Meta-learning tries to compute a meta-classifier which integrates the separately learned classifiers to enhance the overall prediction accuracy (Stolfo et al., 1997). The meta classifiers which we used in this study are: Rotation Forest, Logit Boost, Ensemble of Nested Dichotomies (END), Nested Dichotomies (ND), Class Balance ND, Data Near Balance ND, Ordinal Class Classifier, Bagging, Classification via Regression, Decorate, Random Committee, Multi Boost AB, Ada Boost M1, Attribute Selected Classifier, Random Sub Space, Threshold Selector, Multi Class Classifier, Filtered Classifier, and Dagging.

4.2.2 Bayesian Classifiers

Poole and Mackworth (2010) define a bayesian classifier as “a classifier based on the idea that the role of a natural class is to predict the values of features for members of that class”. In a Bayesian classifier, a probabilistic model of the features is built and the learning agenet uses that model to predict the classification of a new example (Poole & Mackworth, 2010). According to Langley and Sage (1994) The Bayesian classifier is the most straightforward and widely tested method for probabilistic induction. The Bayesian classifiers which we used in our study are: Bayes Network (Bayes Net), Na ïve Bayes Simple, Na ïve Bayes, and Na ïve Bayes Updateable.

4.2.3 Rule-Based Classifiers

Rule-based classifiers produce a set of if-then rules which are extracted from a set of observations and represent significant patterns in the data set and help to create models for prediction and classification (Zhang & Zhou, 2004). The rule-based classifiers which we used in this study are: JRip, Decision Table, Decision Table/Na ïve Bayes Hybrid Classifier (DTNB), Conjunctive Rule, Ridor, OneR, PART, and Non-Nested Generalized Exemplars (NNge).

4.2.4 Function Classifiers

The function classifiers used in this study are: S Pegasos, Sequential Minimal Optimization (SMO), Logistic Regression (Logistic), Simple Logistic, Multilayer Perceptron, Radial Basis Function Network (RBF Network), and Voted Perceptron. The SMO classifier implements sequential minimal optimization algorithm (Platt, 1998) for training a support vector classifier (Machine Learning Group at the University of Waikato, 2015). Gardner and Dorling (1998) define Multilayer Perceptron as "a system of simple interconnected neurons, or nodes, which is a model representing a nonlinear mapping between an input vector and an output vector". Multilayer Perceptron uses backpropagation to classify instances, which is one of the most widely used neural network techniques for classification (Chauvin & Rumelhart, 1995; Rumelhart et al., 1986).

4.2.5 Tree Classifiers

The tree classifiers used in this study are: Alternating Decision Tree (AD Tree), J 48, J 48 graft, Logistic Model Trees (LMT), Best-First Decision Tree (BF Tree), Reduced Error Pruning Tree (REP Tree), NB Tree (decision tree with naive Bayes classifiers at the leaves), Functional Tree (FT), Decision Stump, Random Tree, Simple Classification and Regression Tree (Simple CART), LAD Tree (multi-class alternating decision tree by using the Logit Boost algorithm), and Random Forest. Breiman et al. (1984) introduced CART which is a statistical procedure that is mainlly used as a classification tool (Lee et al., 2006). Landwehr et al. (2005) define logistic model tree (LMT) as a tree which basically consists of a standard decision tree structure with logistic regression functions at the leaves.

4.2.6 Lazy Classifiers

Unlike eager classifiers in which we build a general model before receiving new samples, lazy classifiers keep all of the training samples and do not build a classifier until there is a need for classification of a new sample (Perrizo et al., 2002). The lazy classifiers which we used in this study are: Locally Weighted Learning (LWL), K Star, Nearest Neighbor Classifier (IB1), and K-Nearest Neighbors Classifiers (IB2, IB3, IB4).

4.3 Performance Measures

We used various performance measures to compare the classifiers' performance, including: Accuracy, Kappa Statistic, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Relative Absolute Error (RAE), Root Relative Squared Error (RRSE), False Positive (FP) Rate, Precision, Recall or True Positive (TP) Rate, F-Measure, and Area Under the ROC Curve (AUC).

Accuracy is defined as the percentage of instances from test set which have been correctly classified by the classifier (Stefanowski, 2010).

Kappa Statistic measures the agreement of prediction (Cohen, 1960) and it is computed as:

$$\kappa = \frac{P(A)-P(E)}{1-P\,(E)} \tag{1}$$

Where P(A) is the observed agreement among the raters, and P(E) is the expected agreement among the raters, or in other words, P(E) represents the probability that the raters agree by chance (Di Eugenio & Glass, 2004). A Kappa value of one is interpreted as perfect agreement among the raters, and a Kappa value of zero is interpreted as the agreement is similar to chance (Di Eugenio & Glass, 2004).

Mean Absolute Error (MAE) is a performance measure used to understand how close predictions are to the actual outcomes. Calculating the MAE is very simple and it only requires adding the absolute values of the errors and then dividing the sum of the errors by n (Willmott & Matsuura, 2005). The equation for calculating the MAE is:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i| \tag{2}$$

Where f_i is the predicted value and y_i is the actual value and e_i is the difference between the predicted and actual value.

Root Mean Squared Error (RMSE) is the square root of the average of the squared errors and it is calculated as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(f_i - y_i)^2}{n}} = \sqrt{\frac{\sum_{i=1}^{n} e_i{}^2}{n}} \tag{3}$$

Where f_i is the predicted value and y_i is the actual value and e_i is the difference between the predicted and actual value. By definition, RMSE is never smaller than the MAE (Chai & Draxler, 2014). Willmott & Matsuura (2005) suggest that MAE is a better measure of performance compared to RMSE.

Armstrong and Collopy (1992) introduced the concept of Relative Absoulte Error (RAE) which is calculated by dividing the absolute forecast error for a proposed model by the corresponding error for the random walk. Equation 4 is used for calculating the RAE:

$$RAE = \sqrt{\frac{\sum_{j=1}^{n} |P_{(ij)} - T_j|}{\sum_{j=1}^{n} |T_j - \bar{T}|}} \tag{4}$$

In the above equation, T_j is the target value for sample case j; $\bar{T}$ is equal to $(1/n) \sum_{j=1}^{n} T_j$; and $P_{(ij)}$ is the value predicted by the individual program i for the sample case j (Gepsoft, 2015a).

Root Relative Squared Error (RRSE) is calculated by using the equation 5:

$$RRSE = \sqrt{\frac{\sum_{j=1}^{n} (P_{(ij)} - T_j)^2}{\sum_{j=1}^{n} (T_j - \bar{T})^2}} \tag{5}$$

In this equation, T_j is the target value for sample case j; $\bar{T}$ is equal to $(1/n) \sum_{j=1}^{n} T_j$; and $P_{(ij)}$ is the value predicted by the individual program i for the sample case j (Gepsoft, 2015b).

False Positive (FP) Rate for a given class is the number of false positives (i.e., instances which are incorrectly classified by the model as being part of that specific class) divided by the total number of instances that are not in that particular class (Tan et al., 2005). We can calculate FP Rate using equation 6:

$$FP\ Rate = \frac{FP}{FP+TN} \tag{6}$$

Where FP is the number of false positives and TN is the number of true negatives (i.e., instances that are correctly classified by the model as not being in a particular class) (Tan et al., 2005).

Tan et al. (2005) define precision as “the fraction of records that actually turns out to be positive in the group the classifier has declared as a positive class”. Precision can be calculated by dividing the number of true positives (i.e., instances which are correctly classified by the model as being part of a specific class) by the sum of true positives and false positives. Equation 7 is used for calculating precision:

$$Precision, p = \frac{TP}{TP+FP} \tag{7}$$

Where TP is the number of true positives and FP is the number of false positives (Tan et al., 2005).

Recall and True Positive (TP) Rate are both calculated the same way. Tan et al. (2005) define True Positive (TP) Rate as “the fraction of positive examples predicted correctly by the model”. Recall and TP Rate can be calculated by using equation 8:

$$Recall\ or\ TP\ Rate, r = \frac{TP}{TP+FN} \tag{8}$$

Where TP is the number of true positives and FN is the number of false negatives (i.e., instances that are incorrectly classified by the model as not being in a particular class) (Tan et al., 2005).

F-Measure or F_1 is a harmonic mean between Recall and Precision (Tan et al., 2005) and it is calculated by using equation 9:

$$F_1 = \frac{2rp}{r+p} = \frac{2 \times TP}{2 \times TP+FP+FN} \tag{9}$$

Where r is Recall, p is Precision, TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives (Tan et al., 2005; Witten et al., 2011).

Receiver Operating Characteristic (ROC) curve plots the TP Rate on the vertical axis and the FP Rate on the horizontal axis (Tan et al., 2005; Witten et al., 2011). The Area Under the ROC Curve (AUC) is one of the performance measures which can be used for evaluating different models. A perfect model has an AUC equal to 1 and a model which performs random guessing has an AUC equal to 0.5 (Tan et al., 2005; Witten et al., 2011). In other words, the larger the AUC the better the model (Tan et al., 2005; Witten et al., 2011).

5. Results and Discussion

Table 1 shows the performance of different classifiers when using “decision on the loan application” (i.e., approve or reject) as the class attribute. The first column indicates the type of each classifer, the second column shows the name of the classifer, columns 3 – 13 indicate the performance of each classifier based on different performance measures, and the last column shows the overall rank of the classifier relative to other calssifiers. In order to find the overall rank of each classifier, we first determined the rank of each classifier based on each of the 11 performance measures, then we calculated the average of these 11 rankings for each of the 57 classifiers. Finally, we determined the overall rank of each classifier using its average rank across all performance measures. The results revealed that LAD Tree had the best performance on our data set for deciding whether to approve or reject a loan application. LAD Tree is a classifier for generating a multi-class alternating decision tree by using the Logit Boost algorithm (Holmes et al., 2002). Similar to AD Tree, the number of boosting iterations in LAD Tree is a parameter which can be tuned for the data at hand and it determines the size of the tree constructed (Witten et al., 2011). The LAD Tree which was generated by Weka is depicted in figure 3. The total size of the LAD Tree is 31 and it has 15 leaves. In the overall ranking of classifiers, LAD Tree was followed by Rotation Forest, Logit Boost, Random Forest, and AD Tree. The interesting result is that both Logit Boost and AD Tree classifiers which are used in the LAD Tree classifier are among the top 5 classifiers in the overall ranking. The worst classifiers in this ranking are Nearest Neighbour Classifier (IB1) and K-Nearest Neighbors Classifiers (IB2, IB3, IB4).

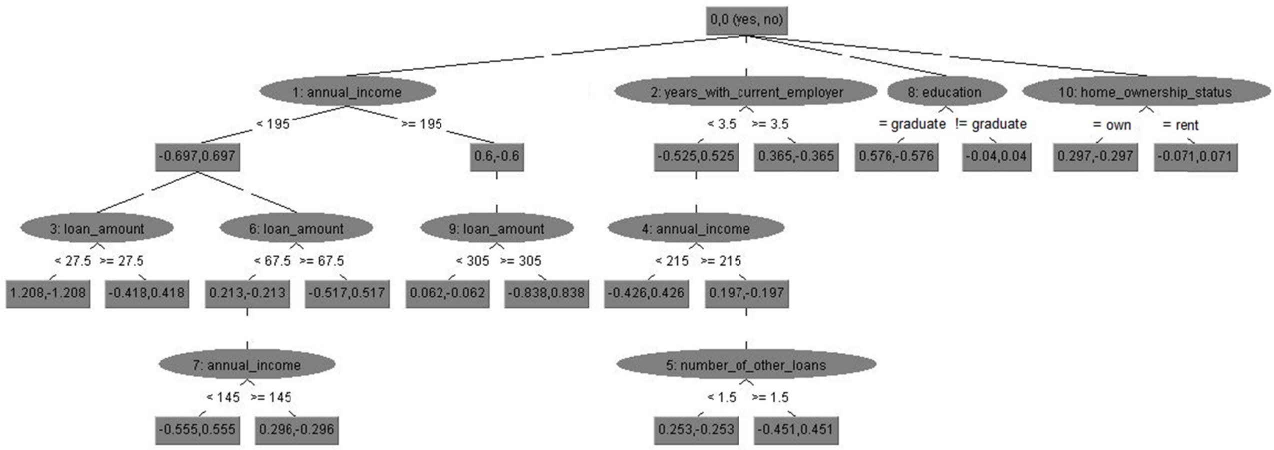

Figure 3. LAD Tree when using "decision on the loan application" as the class attribute

The highest Accuracy Rates were obtained by Rotation Forest (Accuracy = 85.9938 %), followed by LAD Tree (Accuracy = 85.5819 %) and AD Tree (Accuracy = 85.4789 %). The worst Accuracy Rate was obtained by IB3 (Accuracy = 69.5160 %). The best Kappa value was obtained by Rotation Forest (Kappa = 0.7137), followed by LAD Tree (Kappa = 0.7065) and AD Tree (Kappa = 0.7047). The lowest Kappa value was obtained by IB3 (Kappa = 0.3854). The lowest Mean Absolute Error was achieved by Ridor (MAE = 0.1710), followed by Multi Boost AB (MAE = 0.1789) and S Pegasos (MAE = 0.1792). The worst Mean Absolute Error was obtained by RBF Network (MAE = 0.3389). The best Root Mean Squared Error was achieved by Logit Boost (RMSE = 0.3246), followed by LAD Tree (RMSE = 0.3298) and Rotation Forest (RMSE = 0.3299). The highest Root Mean Squared Error was achieved by IB1 (RMSE = 0.5389). The lowest Relative Absolute Error was obtained by Ridor (RAE = 34.5210 %), followed by Multi Boost AB (RAE = 36.1282 %) and S Pegasos (RAE = 36.1846 %). The worst Relative Absolute Error was achieved by RBF Network (RAE = 68.4295 %). The best Root Relative Squared Error was achieved by Logit Boost (RRSE = 65.2239 %), followed by LAD Tree (RRSE = 66.2844 %) and Rotation Forest (RRSE = 66.3017 %). The highest Root Relative Squared Error was obtained by IB1 (RRSE = 108.3004 %). The lowest False Positive Rate was obtained by LAD Tree (FP Rate = 0.155), Rotation Forest (FP Rate = 0.155) and AD Tree (FP Rate = 0.155); followed by Logit Boost (FP Rate = 0.156) and Simple CART (FP Rate = 0.161). The worst FP Rate was achieved by Voted Perceptron (FP Rate = 0.330). The highest Precision was achieved by Rotation Forest (Precision = 0.864), followed by Conjunctive Rule (Precision = 0.859) and LAD Tree (Precision = 0.857). The worst Precision was obtained by IB3 (Precision = 0.696). The best Recall (or TP Rate) was achieved by Rotation Forest (Recall = 0.860), followed by LAD Tree (Recall = 0.856) and AD Tree (Recall = 0.855). The lowest Recall was obtained by IB3 (Recall = 0.695). The highest F-Measure was obtained by Rotation Forest (F_1 = 0.859), followed by LAD Tree (F_1 = 0.855) and AD Tree (F_1 = 0.854). The worst F-Measure was achieved by IB3 (F_1 = 0.695). The largest AUC was achieved by Logit Boost (AUC = 0.917), followed by AD Tree (AUC = 0.913) and Rotation Forest (AUC = 0.911). The smallest AUC was obtained by IB1 (AUC = 0.707). These results clearly indicate that the choice of performance measure affects the comparison of classifiers. In other words, different classifiers might perform differently according to disparate performance measures. For example

Conjunctive Rule is the second best classifiers based on Precision but its rank according to MAE is 37.

Among the function classifiers, S Pegasos (overall rank = 30) performed the best and Voted Perceptron (overall rank = 53) performed the worst. Among the lazy classfiers, LWL (overall rank = 39) had the best performance and IB3 (overall rank = 57) had the worst performance. Among the meta classifiers, the best overall performance was achieved by Rotation Forest (overall rank = 2) and the worst overall performance was obtained by Dagging (overall rank = 48). Among the Bayesian classifiers, Bayes Net (overall rank = 40) performed the best and Naïve Bayes Updateable (overall rank = 51) performed the worst. Among rule-based classifiers, JRip (overall rank = 17) had the best performance and NNge (overall rank = 46) had the worst performance. Finally, the best classifier among tree classifiers was LAD Tree (overall rank = 1) and the worst tree classifier was Random Tree (overall rank = 44). The best group of classifiers were tree classifiers. The average overall rank of the 13 tree classifiers was 18.077. The second best group of classifiers were meta classifiers. The average overall rank of the19 meta classifiers was 19.474. The worst group of classifiers were lazy classifiers. The average overall rank of the 6 lazy classifiers was 50.5.

Based on our results, the overall rank for Simple CART classifier is 13 and the overall rank for Logistic Regression classifier is 35 which is consistent with Lee et al. (2006) who found that CART classifier performs better than Logistic Regression. Our results also indicate that the accuracy of Simple CART classifier (Accuracy = 85.0669 %) is higher than the accuracy of Logistic Regression classifier (Accuracy = 81.2564 %) which is consistent with Lee et al. (2006) findings. According to our results Logistic Regression classifier performs better than Multilayer Perceptron (backpropogation neural network), J 48 (decision tree), SMO (support vector machine), Naïve Bayes, Decision Table, and IB3 (K-Nearest Neighbors classifier with k = 3) when comparing based on AUC; this finding is consistent with Sinha and Zhao (2008).

Table 1. Performance of different classifiers using the "decision on the loan application" as class attribute

Classifier Type	Classifier Name	Accuracy	Kappa Statistic	MAE	RMSE	RAE	RRSE	FP Rate	Precision	Recall	F_1	AUC	Overall Rank
Trees	LAD Tree	85.5819%	0.7065	0.2104	0.3298	42.4862%	66.2844%	0.155	0.857	0.856	0.855	0.905	1
Meta	Rotation Forest	85.9938%	0.7137	0.2299	0.3299	46.4277%	66.3017%	0.155	0.864	0.860	0.859	0.911	2
Meta	Logit Boost	85.0669%	0.6972	0.2202	0.3246	44.4722%	65.2239%	0.156	0.851	0.851	0.850	0.917	3
Trees	Random Forest	85.1699%	0.6973	0.2195	0.3428	44.3152%	68.8907%	0.162	0.854	0.852	0.850	0.909	4
Trees	AD Tree	85.4789%	0.7047	0.2665	0.3389	53.8202%	68.1067%	0.155	0.856	0.855	0.854	0.913	5
Trees	J 48	84.8610%	0.6915	0.2077	0.3566	41.9498%	71.6569%	0.163	0.850	0.849	0.848	0.849	6
Trees	J 48 graft	84.8610%	0.6915	0.2077	0.3566	41.9498%	71.6569%	0.163	0.850	0.849	0.848	0.849	6
Meta	END	84.8610%	0.6915	0.2077	0.3566	41.9498%	71.6569%	0.163	0.850	0.849	0.848	0.849	6
Meta	Class Balance ND	84.8610%	0.6915	0.2077	0.3566	41.9498%	71.6569%	0.163	0.850	0.849	0.848	0.849	6
Meta	Data Near Balance ND	84.8610%	0.6915	0.2077	0.3566	41.9498%	71.6569%	0.163	0.850	0.849	0.848	0.849	6
Meta	ND (Nested Dichotomies)	84.8610%	0.6915	0.2077	0.3566	41.9498%	71.6569%	0.163	0.850	0.849	0.848	0.849	6
Meta	Ordinal Class Classifier	84.8610%	0.6915	0.2077	0.3566	41.9498%	71.6569%	0.163	0.850	0.849	0.848	0.849	6
Trees	Simple CART	85.0669%	0.6957	0.2253	0.3472	45.4979%	69.7698%	0.161	0.852	0.851	0.850	0.863	13
Meta	Bagging	84.4490%	0.6826	0.2160	0.3328	43.6138%	66.8767%	0.169	0.847	0.844	0.843	0.908	14
Meta	Classification via Regression	84.4490%	0.6815	0.2192	0.3317	44.2627%	66.6587%	0.173	0.850	0.844	0.843	0.908	15
Trees	LMT	84.1401%	0.6771	0.2150	0.3432	43.4239%	68.9796%	0.170	0.842	0.841	0.840	0.903	16
Rules	JRip	84.7580%	0.6894	0.2200	0.3502	44.4159%	70.3777%	0.164	0.849	0.848	0.847	0.857	17
Meta	Decorate	84.4490%	0.6833	0.2586	0.3512	52.1606%	70.5796%	0.167	0.846	0.844	0.844	0.889	18
Trees	BF Tree	84.2430%	0.6791	0.2179	0.3534	44.0094%	71.0220%	0.169	0.844	0.842	0.841	0.861	19
Trees	REP Tree	84.3460%	0.6800	0.2207	0.3531	44.5744%	70.9515%	0.172	0.847	0.843	0.842	0.865	20
Meta	Random Committee	83.5221%	0.6638	0.2165	0.3576	43.7150%	71.8577%	0.178	0.837	0.835	0.834	0.884	21
Rules	DTNB	83.0072%	0.6543	0.2333	0.3528	47.1014%	70.8949%	0.181	0.831	0.830	0.829	0.891	22
Meta	Multi Boost AB	82.5953%	0.6447	0.1789	0.3726	36.1282%	74.8745%	0.188	0.828	0.826	0.824	0.884	23
Meta	Ada Boost M1	82.0803%	0.6387	0.2320	0.3427	46.8462%	68.8665%	0.181	0.821	0.821	0.821	0.901	24
Rules	Conjunctive Rule	84.3460%	0.6767	0.2537	0.3575	51.2199%	71.8476%	0.182	0.859	0.843	0.840	0.827	25
Meta	Attribute Selected Classifier	83.3162%	0.6584	0.2346	0.3546	47.3673%	71.2698%	0.184	0.838	0.833	0.831	0.864	26
Rules	Ridor	82.9042%	0.6498	0.1710	0.4135	34.5210%	83.0921%	0.189	0.834	0.829	0.827	0.820	27
Trees	NB Tree	82.3893%	0.6420	0.2190	0.3638	44.2298%	73.1109%	0.186	0.824	0.824	0.823	0.888	28
Rules	Decision Table	82.5953%	0.6440	0.2469	0.3579	49.8559%	71.9249%	0.190	0.829	0.826	0.824	0.885	29
Functions	S Pegasos	82.0803%	0.6362	0.1792	0.4233	36.1846%	85.0707%	0.188	0.821	0.821	0.820	0.816	30
Trees	Functional Tree (FT)	81.8744%	0.6305	0.1965	0.3957	39.6750%	79.5134%	0.194	0.820	0.819	0.817	0.847	31
Meta	Random Sub Space	82.9042%	0.6509	0.3204	0.3681	64.6927%	73.9814%	0.185	0.831	0.829	0.828	0.893	32
Meta	Threshold Selector	82.0803%	0.6328	0.2766	0.3692	55.8570%	74.1977%	0.197	0.826	0.821	0.818	0.885	33
Functions	SMO	81.5654%	0.6272	0.1843	0.4294	37.2244%	86.2843%	0.189	0.815	0.816	0.816	0.813	34
Meta	Multi Class Classifier	81.2564%	0.6224	0.2722	0.3677	54.9677%	73.8898%	0.189	0.813	0.813	0.813	0.887	35
Functions	Logistic	81.2564%	0.6224	0.2722	0.3677	54.9677%	73.8898%	0.189	0.813	0.813	0.813	0.887	35
Rules	OneR	81.2564%	0.6171	0.1874	0.4329	37.8483%	87.0044%	0.202	0.815	0.813	0.811	0.805	37
Rules	PART	80.7415%	0.6094	0.2073	0.4091	41.8582%	82.2050%	0.201	0.807	0.807	0.807	0.825	38
Lazy	LWL	81.8744%	0.6293	0.2924	0.3807	59.0481%	76.5053%	0.197	0.822	0.819	0.817	0.867	39
Bayes	Bayes Net	80.6385%	0.6092	0.2249	0.3791	45.4138%	76.1775%	0.197	0.806	0.806	0.806	0.879	40
Functions	Simple Logistic	80.8445%	0.6148	0.3030	0.3787	61.1924%	76.1109%	0.191	0.810	0.808	0.809	0.878	41
Trees	Decision Stump	81.8744%	0.6293	0.2961	0.3852	59.7928%	77.4205%	0.197	0.822	0.819	0.817	0.785	42
Lazy	K Star	79.7116%	0.5907	0.2277	0.3924	45.9717%	78.8496%	0.206	0.797	0.797	0.797	0.866	43
Trees	Random Tree	79.4027%	0.5846	0.2059	0.4528	41.5805%	90.9881%	0.209	0.794	0.794	0.794	0.794	44
Functions	Multilayer Perceptron	78.9907%	0.5740	0.2160	0.4257	43.6223%	85.5513%	0.218	0.790	0.790	0.789	0.857	45
Meta	Filtered Classifier	80.3296%	0.5966	0.2667	0.3851	53.8453%	77.3939%	0.216	0.808	0.803	0.801	0.839	46
Rules	NNge	78.6818%	0.5680	0.2132	0.4617	43.0472%	92.7877%	0.221	0.786	0.787	0.786	0.783	46
Meta	Dagging	75.2832%	0.4980	0.2787	0.4163	56.2734%	83.6707%	0.258	0.752	0.753	0.752	0.830	48
Functions	RBF Network	75.7981%	0.5124	0.3389	0.4165	68.4295%	83.6996%	0.244	0.759	0.758	0.758	0.812	49
Bayes	Naïve Bayes Simple	70.4428%	0.4173	0.3035	0.4706	61.2839%	94.5769%	0.277	0.724	0.704	0.704	0.814	50
Bayes	Naïve Bayes	70.1339%	0.4115	0.3050	0.4726	61.5976%	94.9810%	0.280	0.722	0.701	0.700	0.813	51
Bayes	Naïve Bayes Updateable	70.1339%	0.4115	0.3050	0.4726	61.5976%	94.9810%	0.280	0.722	0.701	0.700	0.813	51
Functions	Voted Perceptron	71.4727%	0.4006	0.2853	0.5337	57.6100%	107.2630%	0.330	0.744	0.715	0.697	0.738	53
Lazy	IB2	71.9876%	0.4170	0.3083	0.4788	62.2583%	96.2169%	0.315	0.734	0.720	0.709	0.744	54
Lazy	IB1	70.9578%	0.4135	0.2904	0.5389	58.6441%	108.3004%	0.296	0.710	0.710	0.710	0.707	54
Lazy	IB4	71.3697%	0.4111	0.3273	0.4476	66.0985%	89.9476%	0.310	0.716	0.714	0.708	0.776	56
Lazy	IB3	69.5160%	0.3854	0.3193	0.4585	64.4774%	92.1352%	0.309	0.696	0.695	0.695	0.765	57

Table 2 shows the performance of different classifiers when using "applicant default status" (i.e., whether the approved loan applicant defaulted on his/her loan within 5 years after obtaining the loan or not) as the class attribute. The first column indicates the type of each classifer, the second column shows the name of the classifer, columns 3 – 13 indicate the performance of each classifier based on different performance measures, and the last column shows the overall rank of the classifier relative to other calssifiers. In order to find the overall rank of each classifier, first we determined the rank of rach classifier based on each of the 11 performance measures, then we calculated the average of these 11 rankings for each of the 57 classifiers. Finally, we determined the overall rank of each classifier using its average rank across all performance measures. The results revealed that Bagging had the best performance on our data set for deciding whether an approved loan applicant will default on his or her loan within 5 years after obtaining the loan or not. Bagging is a classifier which generates multiple versions of a predictor and uses them to get an aggregated predictor (Breiman, 1996). These multiple versions are obtained by creating bootstrap replicates of the learning set which are then used as new learning sets (Breiman, 1996). The classifier model for Bagging is depicted in appendix 1. In the overall ranking of classifiers, Bagging was followed by Simple Cart (depicted in figure 4), J 48 (depicted in figure 5), J 48 graft, END, Class Balance ND, Data Near Balance ND, ND, and Ordinal Class Classifier. The worst classifiers in this ranking are Conjunctive Rule, 2-Nearest Neighbors (IB2), and Voted Perceptron.

```
=== Classifier model (full training set) ===

CART Decision Tree

number_of_other_loans < 2.5

|   number_of_other_loans < 1.5: no(380.0/38.0)

|   number_of_other_loans >= 1.5

|   |   annual_income < 235.0: yes(18.0/0.0)

|   |   annual_income >= 235.0: no(51.0/9.0)

number_of_other_loans >= 2.5: yes(32.0/5.0)

Number of Leaf Nodes: 4

Size of the Tree: 7

Time taken to build model: 0.1 seconds
```

Figure 4. Simple Cart model when using “applicant default status” as the class attribute

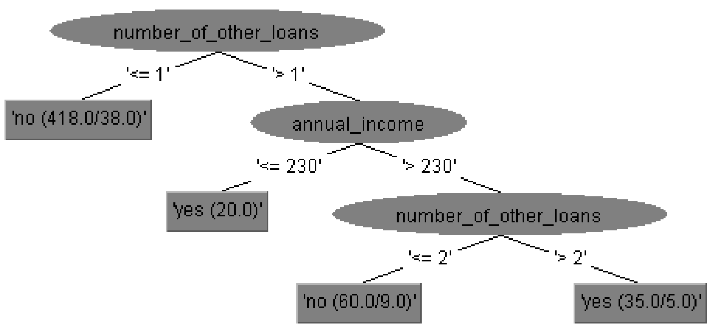

Figure 5. J 48 (decision tree) when using “applicant default status” as the class attribute

Table 2. Performance of different classifiers using the "applicant default status" as class attribute

Classifier Type	Classifier Name	Accuracy	Kappa Statistic	MAE	RMSE	RAE	RRSE	FP Rate	Precision	Recall	F_1	AUC	Overall Rank
Meta	Bagging	90.2439%	0.6098	0.1781	0.2985	59.6317%	77.3719%	0.390	0.902	0.902	0.892	0.810	1
Trees	Simple Cart	90.2439%	0.6060	0.1746	0.2962	58.4897%	76.7696%	0.398	0.903	0.902	0.891	0.735	2
Trees	J 48	89.8687%	0.5909	0.1759	0.3025	58.9065%	78.3877%	0.407	0.898	0.899	0.887	0.710	3
Trees	J 48 graft	89.8687%	0.5909	0.1759	0.3025	58.9065%	78.3877%	0.407	0.898	0.899	0.887	0.710	3
Meta	END	89.8687%	0.5909	0.1759	0.3025	58.9065%	78.3877%	0.407	0.898	0.899	0.887	0.710	3
Meta	Class Balance ND	89.8687%	0.5909	0.1759	0.3025	58.9065%	78.3877%	0.407	0.898	0.899	0.887	0.710	3
Meta	Data Near Balance ND	89.8687%	0.5909	0.1759	0.3025	58.9065%	78.3877%	0.407	0.898	0.899	0.887	0.710	3
Meta	ND (Nested Dichotomies)	89.8687%	0.5909	0.1759	0.3025	58.9065%	78.3877%	0.407	0.898	0.899	0.887	0.710	3
Meta	Ordinal Class Classifier	89.8687%	0.5909	0.1759	0.3025	58.9065%	78.3877%	0.407	0.898	0.899	0.887	0.710	3
Trees	Random Forest	89.4934%	0.5876	0.1832	0.2971	61.3493%	76.9953%	0.392	0.890	0.895	0.885	0.857	10
Meta	Random Committee	88.5553%	0.5611	0.1688	0.3023	56.5274%	78.3448%	0.394	0.878	0.886	0.877	0.858	11
Trees	NB Tree	89.4934%	0.5837	0.1708	0.3072	57.2033%	79.6146%	0.400	0.891	0.895	0.884	0.811	12
Trees	LAD Tree	89.8687%	0.5986	0.1835	0.3133	61.4388%	81.1888%	0.391	0.896	0.899	0.888	0.767	13
Meta	Classification via Regression	89.6811%	0.5853	0.1840	0.3012	61.6062%	78.0506%	0.408	0.895	0.897	0.885	0.838	14
Rules	JRip	88.9306%	0.5755	0.1819	0.3131	60.9155%	81.1317%	0.385	0.883	0.889	0.881	0.747	15
Trees	REP Tree	89.1182%	0.5689	0.1828	0.3119	61.2047%	80.8379%	0.409	0.886	0.891	0.880	0.743	16
Meta	Logit Boost	89.1182%	0.5563	0.1924	0.3121	64.4358%	80.8768%	0.433	0.889	0.891	0.878	0.796	17
Functions	Logistic	87.8049%	0.5323	0.1875	0.3109	62.7924%	80.5659%	0.412	0.869	0.878	0.869	0.832	18
Meta	Multi Class Classifier	87.8049%	0.5323	0.1875	0.3109	62.7924%	80.5659%	0.412	0.869	0.878	0.869	0.832	18
Trees	BF Tree	88.5553%	0.5356	0.1809	0.3143	60.5977%	81.4593%	0.442	0.881	0.886	0.872	0.749	20
Functions	S Pegasos	88.3677%	0.5257	0.1163	0.3411	38.9573%	88.3905%	0.451	0.879	0.884	0.869	0.716	21
Rules	Ridor	88.1801%	0.5203	0.1182	0.3438	39.5856%	89.1005%	0.451	0.876	0.882	0.868	0.715	22
Functions	SMO	88.1801%	0.5157	0.1182	0.3438	39.5856%	89.1005%	0.459	0.877	0.882	0.867	0.711	23
Rules	NNge	87.6173%	0.5273	0.1238	0.3519	41.4707%	91.1973%	0.412	0.867	0.876	0.867	0.732	24
Trees	AD Tree	89.6811%	0.5370	0.2665	0.3401	89.2577%	88.1435%	0.432	0.899	0.897	0.883	0.743	25
Rules	DTNB	87.9925%	0.5243	0.2018	0.3222	67.5840%	83.4910%	0.436	0.872	0.880	0.868	0.797	26
Functions	Multilayer Perceptron	85.9287%	0.5056	0.1539	0.3515	51.5353%	91.1002%	0.376	0.854	0.859	0.856	0.782	26
Meta	Decorate	88.3677%	0.5391	0.2085	0.3189	69.8122%	82.6433%	0.427	0.877	0.884	0.872	0.741	28
Bayes	Bayes Net	86.3039%	0.4977	0.1843	0.3225	61.7399%	83.5678%	0.407	0.854	0.863	0.856	0.825	29
Meta	Random Sub Space	88.5553%	0.5119	0.2234	0.3154	74.8147%	81.7461%	0.482	0.888	0.886	0.867	0.813	30
Meta	Rotation Forest	87.4296%	0.4899	0.1983	0.3140	66.4146%	81.3821%	0.469	0.866	0.874	0.859	0.792	31
Trees	Functional Tree (FT)	86.6792%	0.4796	0.1453	0.3484	48.6517%	90.2941%	0.454	0.855	0.867	0.855	0.740	32
Functions	Simple Logistic	86.8668%	0.4747	0.2066	0.3165	69.2037%	82.0353%	0.470	0.858	0.869	0.854	0.820	33
Trees	LMT	86.8668%	0.4747	0.2066	0.3165	69.2037%	82.0353%	0.470	0.858	0.869	0.854	0.820	33
Lazy	LWL	86.6792%	0.5115	0.2188	0.3335	73.2645%	86.4207%	0.398	0.858	0.867	0.860	0.778	35
Meta	Filtered Classifier	87.4296%	0.5089	0.2026	0.3320	67.8401%	86.0320%	0.437	0.864	0.874	0.863	0.696	36
Meta	Dagging	85.3659%	0.3091	0.1653	0.3378	55.3570%	87.5382%	0.634	0.857	0.854	0.816	0.783	37
Meta	Multi Boost AB	85.7411%	0.4351	0.1423	0.3574	47.6722%	92.6164%	0.489	0.843	0.857	0.843	0.761	38
Trees	Random Tree	84.0525%	0.4489	0.1595	0.3983	53.4092%	103.2241%	0.404	0.836	0.841	0.838	0.722	39
Rules	OneR	86.8668%	0.4192	0.1313	0.3624	43.9840%	93.9202%	0.550	0.868	0.869	0.843	0.659	40
Lazy	K Star	84.4278%	0.4187	0.1704	0.3530	54.0749%	91.4967%	0.467	0.831	0.844	0.835	0.806	41
Meta	Attribute Selected Classifier	87.0544%	0.4479	0.2108	0.3305	70.6098%	85.6563%	0.518	0.865	0.871	0.850	0.723	42
Meta	Threshold Selector	82.9268%	0.4289	0.2167	0.3309	72.5610%	85.7593%	0.399	0.830	0.829	0.830	0.821	43
Rules	Decision Table	86.4916%	0.4597	0.2231	0.3308	74.7144%	85.7270%	0.479	0.853	0.865	0.850	0.776	44
Meta	Ada Boost M1	84.8030%	0.4225	0.2082	0.3361	69.7265%	87.1076%	0.475	0.834	0.848	0.837	0.804	45
Bayes	Na ïve Bayes	84.0525%	0.4252	0.1956	0.3520	65.5116%	91.2335%	0.444	0.831	0.841	0.834	0.797	46
Bayes	Na ïve Bayes Updateable	84.0525%	0.4252	0.1956	0.3520	65.5116%	91.2335%	0.444	0.831	0.841	0.834	0.797	47
Bayes	Na ïve Bayes Simple	83.8649%	0.4259	0.1947	0.3547	65.2129%	91.9173%	0.437	0.830	0.839	0.834	0.798	48
Functions	RBF Network	84.9906%	0.4322	0.2235	0.3414	74.8517%	88.4862%	0.466	0.836	0.850	0.840	0.789	49
Rules	PART	83.3021%	0.4182	0.1904	0.3907	63.7560%	101.2580%	0.430	0.827	0.833	0.830	0.710	50
Lazy	IB1	81.6135%	0.3775	0.1839	0.4288	61.5776%	111.1279%	0.442	0.815	0.816	0.815	0.687	51
Trees	Decision Stump	82.5516%	0.3971	0.2398	0.3508	80.3227%	90.9195%	0.440	0.821	0.826	0.823	0.700	52
Lazy	IB3	83.1144%	0.2978	0.2080	0.3629	69.6725%	94.0416%	0.591	0.805	0.831	0.808	0.751	53
Lazy	IB4	83.8649%	0.2551	0.2170	0.3558	72.6724%	92.2140%	0.653	0.819	0.839	0.801	0.757	54
Rules	Conjunctive Rule	84.8030%	0.2702	0.2468	0.3530	82.6702%	91.4935%	0.659	0.850	0.848	0.806	0.675	55
Lazy	IB2	82.9268%	0.1801	0.1976	0.3736	66.1871%	96.8328%	0.703	0.801	0.829	0.782	0.721	56
Functions	Voted Perceptron	81.2383%	0.0018	0.1876	0.4331	62.8343%	112.2561%	0.811	0.706	0.812	0.737	0.524	57

6. Conclusions

Our study revealed interesting results about the performance of different data mining techniques in predicting whether a loan application is approved or rejected and also for predicting whether an approved loan applicant will default on his/her loan or not. These results can be used by banks' loan departments in to design computer programs by using data mining classifiers for facilitating the decision making. The use of data mining classifiers can help the banks to reduce the rate of default on the loans which they generate by estimating the probability of default for each new loan applicant.

Therefore banks will be able to identify which loan applicants have a higher probability of default and can reject their loan applications. Our results revealed that the best data mining classifier for predicting whether a loan applicant will be approved or rejected is LAD Tree, followed by Rotation Forest, Logit Boost, Random Forest, and AD Tree. The results also indicated that the best classifier for predicting whether an approved applicant will default on his/her loan is Bagging, followed by Simple Cart, J 48, J 48 graft, END, Class Balance ND, Data Near Balance ND, ND, and Ordinal Class Classifier. It was also found that different classifiers perform differently according to disparate performance measures, which means that the choice of performance measure can affect the comparison of classifiers.

References

Ansari, A., & Riasi, A. (2016a). Modelling and Evaluating Customer Loyalty Using Neural Networks: Evidence from Startup Insurance Companies. *Future Business Journal*, *2*(1), 15-30. http://dx.doi.org/10.1016/j.fbj.2016.04.001

Ansari, A., & Riasi, A. (2016b). Clustering the Customers of Steel Industry Using a Combination of Fuzzy C-Means and Genetic Algorithms. *Working Paper*.

Ansari, A., & Riasi, A. (2016c). Using Decision Trees to Analyze the Customers' Shopping Location Preferences. *Working Paper*.

Armstrong, J. S., & Collopy, F. (1992). Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting*, *8*(1), 69-80. http://dx.doi.org/10.1016/0169-2070(92)90008-W

Back, B., Laitinen, T., Sere, K., & Van Wezel, M. (1996). *Choosing bankruptcy predictors using discriminant analysis, logit analysis and genetic algorithms*. Turku Centre for Computer Science, Technical Report No. 40.

Barney, D. K., Graves, O. F., & Johnson, J. D. (1999). The farmers home administration and farm debt failure prediction. *Journal of Accounting and Public Policy, 18*(2), 99-139. http://dx.doi.org/10.1016/S0278-4254(98)10018-2

Bose, I., & Mahapatra, R. K. (2001). Business data mining—a machine learning perspective. *Information & Management, 39*(3), 211-225. http://dx.doi.org/10.1016/S0378-7206(01)00091-X

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, *30*(7), 1145-1159. http://dx.doi.org/10.1016/S0031-3203(96)00142-2

Brandão, L. E., Dyer, J. S., & Hahn, W. J. (2005). Using binomial decision trees to solve real-option valuation problems. *Decision Analysis*, *2*(2), 69-88. http://dx.doi.org/10.1287/deca.1050.0040

Brause, R., Langsdorf, T., & Hepp, M. (1999). Neural data mining for credit card fraud detection. In *Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on* (pp. 103-106). IEEE. http://dx.doi.org/10.1109/tai.1999.809773

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123-140. http://dx.doi.org/10.1007/BF00058655

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees*, Pacific Grove, CA: Wadsworth.

Bullard, F. (2006). *Introduction to Statistical Inference*. SAMSI/CRSC Undergraduate Workshop at NCSU.

Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. *Geoscientific Model Development*, *7*(3), 1247-1250. http://dx.doi.org/10.5194/gmd-7-1247-2014

Chan P. K., & Stolfo, S. J. (1993). Experiments on multistrategy learning by meta-learning. In *Proceedings of Proceedings of the second international conference on Information and knowledge management* (pp. 314–323). New York, NY: ACM Press. http://dx.doi.org/10.1145/170088.170160

Chan, P. K., Fan, W., Prodromidis, A. L., & Stolfo, S. J. (1999). Distributed data mining in credit card fraud detection. *Intelligent Systems and their Applications, IEEE*, *14*(6), 67-74. http://dx.doi.org/10.1109/5254.809570

Chang, V., Walters, R. J., & Wills, G. B. (2016). Organisational sustainability modelling—An emerging service and analytics model for evaluating Cloud Computing adoption with two case studies. *International Journal of Information Management*, *36*(1), 167-179. http://dx.doi.org/10.1016/j.ijinfomgt.2015.09.001

Chauvin, Y., & Rumelhart, D. E. (1995). *Back-propagation: Theory, architectures, and applications.* Hillsdale, NJ: Lawrence Erlbaum Associates Inc.

Chen, S. Y., & Liu, X. (2004). The contribution of data mining to information science. *Journal of Information Science, 30*(6), 550-558. http://dx.doi.org/10.1177/0165551504047928

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1), 37-46. http://dx.doi.org/10.1177/001316446002000104

Desai, V. S., Crook, J. N., & Overstreet, G. A. (1996). A comparison of neural networks and linear scoring models in the credit union environment. *European Journal of Operational Research*, *95*(1), 24-37. http://dx.doi.org/10.1016/0377-2217(95)00246-4

Di Eugenio, B., & Glass, M. (2004). The kappa statistic: A second look. *Computational Linguistics*, *30*(1), 95-101. http://dx.doi.org/10.1162/089120104773633402

Doumpos, M., Kosmidou, K., Baourakis, G., & Zopounidis, C. (2002). Credit risk assessment using a multicriteria hierarchical discrimination approach: A comparative analysis. *European Journal of Operational Research*, *138*(2), 392-412. http://dx.doi.org/10.1016/S0377-2217(01)00254-5

Fish, K. E., Johnson, J. D., Dorsey, R. E., & Blodgett, J. G. (2004). Using an artificial neural network trained with a genetic algorithm to model brand share. *Journal of Business Research*, *57*(1), 79-85. http://dx.doi.org/10.1016/S0148-2963(02)00287-4

Gardner, M. W., & Dorling, S. R. (1998). Artificial neural networks (the multilayer perceptron)-a review of applications in the atmospheric sciences. *Atmospheric Environment*, *32*(14), 2627-2636. http://dx.doi.org/10.1016/S1352-2310(97)00447-0

Gepsoft (2015a). *Relative Absolute Error.* Retrieved November 6, 2015, from http://www.gepsoft.com/gxpt4kb/Chapter10/Section2/SS15.htm

Gepsoft (2015b). *Root Relative Squared Error.* Retrieved November 7, 2015, from http://www.gepsoft.com/gxpt4kb/Chapter10/Section1/SS07.htm

Glorfeld, L. W. (1996). A methodology for simplification and interpretation of backpropagation-based neural network models. *Expert Systems with Applications*, *10*(1), 37-54. http://dx.doi.org/10.1016/0957-4174(95)00032-1

Grierson, H. J., Corney, J. R., & Hatcher, G. D. (2015). Using visual representations for the searching and browsing of large, complex, multimedia data sets. *International Journal of Information Management*, *35*(2), 244-252. http://dx.doi.org/10.1016/j.ijinfomgt.2014.12.003

Hachaj, T. (2014). Real time exploration and management of large medical volumetric datasets on small mobile devices—evaluation of remote volume rendering approach. *International Journal of Information Management*, *34*(3), 336-343. http://dx.doi.org/10.1016/j.ijinfomgt.2013.11.005

Hamid, S. A., & Iqbal, Z. (2004). Using neural networks for forecasting volatility of S&P 500 Index futures prices. *Journal of Business Research*, *57*(10), 1116-1125. http://dx.doi.org/10.1016/S0148-2963(03)00043-2

Han, J., & Kamber, M. (2001). *Data Mining: Concepts and Techniques*. San Francisco, CA: Morgan Kaufmann.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press.

Holmes, G., Pfahringer, B., Kirkby, R., Frank, E., & Hall, M. (2002). Multiclass Alternating Decision Trees. In T. Elomaa, H. Mannila, & H. Toivonen (Eds.), *Proceedings of the Thirteenth European Conference on Machine Learning* (pp. 161–172). Helsinki. Berlin: Springer-Verlag. http://dx.doi.org/10.1007/3-540-36755-1_14

Huang, C. L., Chen, M. C., & Wang, C. J. (2007). Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications*, *33*(4), 847-856. http://dx.doi.org/10.1016/j.eswa.2006.07.007

Hung, S. Y., Liang, T. P., & Liu, V. W. C. (1996). Integrating arbitrage pricing theory and artificial neural networks to support portfolio management. *Decision support systems*, *18*(3), 301-316. http://dx.doi.org/10.1016/S0167-9236(96)80006-6

Iba, H., & Sasaki, T. (1999). Using genetic programming to predict financial data. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on* (Vol. 1). IEEE. http://dx.doi.org/10.1109/CEC.1999.781932

Jagielska, I., & Jaworski, J. (1996). Neural network for predicting the performance of credit card accounts. *Computational Economics*, *9*(1), 77-82. http://dx.doi.org/10.1007/BF00115693

Jahangirian, M., Eldabi, T., Garg, L., Jun, G. T., Naseer, A., Patel, B., Stergioulas, L., & Young, T. (2011). A rapid review method for extremely large corpora of literature: Applications to the domains of modelling, simulation, and management. *International Journal of Information Management*, *31*(3), 234-243. http://dx.doi.org/10.1016/j.ijinfomgt.2010.07.004

Jain, B. A., & Nag, B. N. (1997). Performance evaluation of neural network decision models. *Journal of Management Information Systems*, *14*(2), 201-216. http://dx.doi.org/10.1080/07421222.1997.11518171

Kim, C. N., & McLeod Jr, R. (1999). Expert, linear models, and nonlinear models of expert decision making in bankruptcy prediction: a lens model analysis. *Journal of Management Information Systems*, 189-206. http://dx.doi.org/10.1080/07421222.1999.11518239

Kim, J. W., Lee, B. H., Shaw, M. J., Chang, H. L., & Nelson, M. (2001). Application of decision-tree induction techniques to personalized advertisements on internet storefronts. *International Journal of Electronic Commerce*, *5*(3), 45-62.

Kirkos, E., Spathis, C., & Manolopoulos, Y. (2007). Data mining techniques for the detection of fraudulent financial statements. *Expert Systems with Applications*, *32*(4), 995-1003. http://dx.doi.org/10.1016/j.eswa.2006.02.016

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* (pp. 1137–1143). San Francisco, CA: Morgan Kaufmann.

Lam, M. (2004). Neural network techniques for financial performance prediction: integrating fundamental and technical analysis. *Decision Support Systems*, *37*(4), 567-581. http://dx.doi.org/10.1016/S0167-9236(03)00088-5

Landwehr, N., Hall, M., & Frank, E. (2005). Logistic model trees. *Machine Learning*, *59*(1), 161-205. http://dx.doi.org/10.1007/s10994-005-0466-3

Langley, P., & Sage, S. (1994). Induction of selective Bayesian classifiers. In Proceedings of the Tenth international conference on Uncertainty in artificial intelligence (pp. 399-406). San Francisco, CA: Morgan Kaufmann http://dx.doi.org/10.1016/b978-1-55860-332-5.50055-9

Lee, T. S., Chiu, C. C., Chou, Y. C., & Lu, C. J. (2006). Mining the customer credit using classification and regression tree and multivariate adaptive regression splines. *Computational Statistics & Data Analysis*, *50*(4), 1113-1130. http://dx.doi.org/10.1016/j.csda.2004.11.006

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*(3), 245-254. http://dx.doi.org/10.1016/S0957-4174(02)00044-1

Machine Learning Group at the University of Waikato (2015). *Weka 3: Data Mining Software in Java*. Retrieved November 3, 2015, from http://www.cs.waikato.ac.nz/ml/weka/

Mirghafoori, S. H. A., Taheri, M., & Zareh Ahmadabadi, H. (2010). Accessing service quality measurement techniques using artificial neural networks. *Iranian Journal of Business Management Perspective*, *8*(31), 63-79.

Mostafa, M. M., & El-Masry, A. A. (2013). Citizens as consumers: Profiling e-government services' users in Egypt via data mining techniques. *International Journal of Information Management*, *33*(4), 627-641. http://dx.doi.org/10.1016/j.ijinfomgt.2013.03.007

Olaru, D., & Purchase, S. (2014). Rethinking validation: Efficient search of the space of parameters for an agent-based model. *Australasian Marketing Journal (AMJ)*, *22*(1), 60-68. http://dx.doi.org/10.1016/j.ausmj.2013.12.010

Pérez-Montoro, M., & Nualart, J. (2015). Visual articulation of navigation and search systems for digital libraries. *International Journal of Information Management*, *35*(5), 572-579. http://dx.doi.org/10.1016/j.ijinfomgt.2015.06.005

Perrizo, W., Ding, Q., & Denton, A. (2002). Lazy Classifiers Using P-trees. In *CAINE* (pp. 176-179).

Piramuthu, S. (1999). Financial credit-risk evaluation with neural and neurofuzzy systems. *European Journal of Operational Research*, *112*(2), 310-321. http://dx.doi.org/10.1016/S0377-2217(97)00398-6

Piramuthu, S., Shaw, M. J., & Gentry, J. A. (1994). A classification approach using multi-layered neural networks. *Decision Support Systems*, *11*(5), 509-525. http://dx.doi.org/10.1016/0167-9236(94)90022-1

Platt, J. C. (1998). Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods - Support Vector Learning*, B. Scholkopf, C. J. C. Burges, and A. J. Smola, Eds. (pp. 41-65). Cambridge, MA: MIT Press.

Poole D., & Mackworth, A. (2010). *Bayesian Classifiers*. Retrieved November 4, 2015, from http://artint.info/html/ArtInt_181.html#id1

Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-validation. In *Encyclopedia of database systems* (pp. 532-538). Springer US. http://dx.doi.org/10.1007/978-0-387-39940-9_565

Refenes, A. N., Zapranis, A., & Francis, G. (1994). Stock performance modeling using neural networks: a comparative Refenes, A. N., Zapranis, A., & Francis, G. (1994). Stock performance modeling using neural networks: a comparative study with regression models. *Neural networks*, *7*(2), 375-388.

http://dx.doi.org/10.1016/0893-6080(94)90030-2

Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986). Learning internal representations by error propagation, in: D.E. Rumelhart, J.L. McClelland (Eds.), *Parallel Distributed Processing* (pp. 318–362). Cambridge, MA: MIT Press.

Ryu, Y. U., & Yue, W. T. (2005). Firm bankruptcy prediction: experimental comparison of isotonic separation and other classification approaches. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *35*(5), 727-737. http://dx.doi.org/10.1109/TSMCA.2005.843393

Saad, E. W., Prokhorov, D. V., & Wunsch, D. C. (1998). Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *Neural Networks, IEEE Transactions on*, *9*(6), 1456-1470. http://dx.doi.org/10.1109/72.728395

Shin, K. S., & Lee, Y. J. (2002). A genetic algorithm application in bankruptcy prediction modeling. *Expert Systems with Applications*, *23*(3), 321-328. http://dx.doi.org/10.1016/S0957-4174(02)00051-9

Sinha, A. P., & May, J. H. (2004). Evaluating and tuning predictive data mining models using receiver operating characteristic curves. *Journal of Management Information Systems*, *21*(3), 249-280.

Sinha, A. P., & Zhao, H. (2008). Incorporating domain knowledge into data mining classifiers: An application in indirect lending. *Decision Support Systems*, *46*(1), 287-299. http://dx.doi.org/10.1016/j.dss.2008.06.013

Stefanowski, J. (2010). *Data Mining – Evaluation of Classifiers*. Retrieved November 6, 2015, from http://www.cs.put.poznan.pl/jstefanowski/sed/DM-4-evaluatingclassifiersnew.pdf

Stolfo, S. J., Prodromidis, A. L., Tselepis, S., Lee, W., Fan, D. W., & Chan, P. K. (1997). JAM: Java agents for meta-learning over distributed databases. In *Proceedings of the 3rd ACM SIGMOD International Workshop on Data Mining and Knowledge Discovery* (pp. 74–81). New York, NY: ACM Press.

Sung, T. K., Chang, N., & Lee, G. (1999). Dynamics of modeling in data mining: interpretive approach to bankruptcy prediction. *Journal of Management Information Systems*, *16*(1), 63-85. http://dx.doi.org/10.1080/07421222.1999.11518234

Tan, P., Steinbach, M., & Kumar, V. (2005). *Introduction to Data Mining.* Boston, MA: Addison-Wesley.

Thomas, L. C. (2000). A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers. *International Journal of Forecasting*, *16*(2), 149-172. http://dx.doi.org/10.1016/S0169-2070(00)00034-0

Thomassey, S., & Fiordaliso, A. (2006). A hybrid sales forecasting system based on clustering and decision trees. *Decision Support Systems*, *42*(1), 408-421. http://dx.doi.org/10.1016/j.dss.2005.01.008

Tseng, F. M., Tzeng, G. H., Yu, H. C., & Yuan, B. J. (2001). Fuzzy ARIMA model for forecasting the foreign exchange market. *Fuzzy Sets and Systems*, *118*(1), 9-19. http://dx.doi.org/10.1016/S0165-0114(98)00286-3

Walczak, S. (1999). Gaining competitive advantage for trading in emerging capital markets with neural networks. *Journal of Management Information Systems*, *16*(2), 177-192. http://dx.doi.org/10.1080/07421222.1999.11518251

West, D. (2000). Neural network credit scoring models. *Computers & Operations Research*, *27*(11), 1131-1152. http://dx.doi.org/10.1016/S0305-0548(99)00149-5

Willmott, C. J., & Matsuura, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, *30*(1), 79-82. http://dx.doi.org/10.3354/cr030079

Wilson, R. L., & Sharda, R. (1994). Bankruptcy prediction using neural networks. *Decision support systems*, *11*(5), 545-557. http://dx.doi.org/10.1016/0167-9236(94)90024-8

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques.* Burlington, MA: Morgan Kaufmann Publishers.

Wu, K. L., Yu, P. S., & Ballman, A. (1998). Speedtracer: A web usage mining and analysis tool. *IBM Systems Journal*, *37*(1), 89-105. http://dx.doi.org/10.1147/sj.371.0089

Yoon, Y., Swales Jr, G., & Margavio, T. M. (1993). A comparison of discriminant analysis versus artificial neural networks. *Journal of the Operational Research Society*, *44* (1), 51-60. http://dx.doi.org/10.2307/2584434

Zhang, D., & Zhou, L. (2004). Discovering golden nuggets: data mining in financial application. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *4*(34), 513-522. http://dx.doi.org/10.1109/TSMCC.2004.829279

Appendix 1:

Classifier model for Bagging when using the “applicant default status” as class attribute:

```
=== Classifier model (full training set) ===


All the base classifiers:


REPTree

============


number_of_other_loans < 1.5
|   loan_amount < 165
|   |   education = graduate : no (32/1) [17/0]
|   |   education = bachelor : no (124/3) [54/1]
|   |   education = some college
|   |   |   marital_status = married/widowed : no (16/0) [10/2]
|   |   |   marital_status = single : yes (8/3) [4/1]
|   |   |   marital_status = divorced : no (4/0) [1/0]
|   |   education = highschool or less : no (52/2) [35/4]
|   loan_amount >= 165
|   |   age < 26.5 : yes (2/0) [0/0]
|   |   age >= 26.5
|   |   |   loan_amount < 187.5 : no (10/0) [6/0]
|   |   |   loan_amount >= 187.5
|   |   |   |   education = graduate : no (5/2) [2/0]
|   |   |   |   education = bachelor : no (7/1) [1/0]
|   |   |   |   education = some college : yes (4/1) [2/1]
|   |   |   |   education = highschool or less
|   |   |   |   |   years_with_current_employer < 11.5 : yes (6/2) [4/1]
|   |   |   |   |   years_with_current_employer >= 11.5 : no (15/0) [3/0]
number_of_other_loans >= 1.5
|   annual_income < 285 : yes (26/0) [21/5]
|   annual_income >= 285
|   |   number_of_other_loans < 2.5 : no (29/5) [15/1]
|   |   number_of_other_loans >= 2.5 : yes (15/0) [3/0]


Size of the tree : 26




REPTree

============


number_of_other_loans < 1.5
|   loan_amount < 165
```

```
|   |   age < 25.5
|   |   |   loan_duration < 4
|   |   |   |   years_with_current_employer < 1.5 : no (3/0) [0/0]
|   |   |   |   years_with_current_employer >= 1.5 : yes (5/1) [1/0]
|   |   |   loan_duration >= 4 : no (7/0) [4/1]
|   |   age >= 25.5 : no (222/10) [112/6]
|   loan_amount >= 165
|   |   age < 25.5 : yes (2/0) [1/0]
|   |   age >= 25.5
|   |   |   years_with_current_employer < 3.5 : no (11/0) [4/0]
|   |   |   years_with_current_employer >= 3.5
|   |   |   |   loan_duration < 4.5
|   |   |   |   |   education = graduate : yes (2/0) [3/1]
|   |   |   |   |   education = bachelor : no (1/0) [1/0]
|   |   |   |   |   education = some college : yes (3/0) [1/0]
|   |   |   |   |   education = highschool or less : no (5/2) [3/0]
|   |   |   |   loan_duration >= 4.5
|   |   |   |   |   loan_amount < 227.5
|   |   |   |   |   |   education = graduate : no (1/0) [1/0]
|   |   |   |   |   |   education = bachelor : yes (1/0) [1/0]
|   |   |   |   |   |   education = some college : no (2/0) [1/0]
|   |   |   |   |   |   education = highschool or less : yes (2/0) [0/0]
|   |   |   |   |   loan_amount >= 227.5 : no (11/0) [4/0]
number_of_other_loans >= 1.5
|   years_with_current_employer < 16
|   |   education = graduate
|   |   |   age < 40.5 : no (7/0) [3/0]
|   |   |   age >= 40.5 : yes (4/0) [2/0]
|   |   education = bachelor
|   |   |   loan_amount < 67.5 : no (4/0) [3/1]
|   |   |   loan_amount >= 67.5 : yes (14/4) [12/3]
|   |   education = some college : yes (13/0) [7/1]
|   |   education = highschool or less
|   |   |   marital_status = married/widowed
|   |   |   |   loan_amount < 97.5 : no (3/1) [3/0]
|   |   |   |   loan_amount >= 97.5 : yes (6/0) [3/0]
|   |   |   marital_status = single : yes (11/0) [1/0]
|   |   |   marital_status = divorced : no (2/0) [0/0]
|   years_with_current_employer >= 16 : no (13/0) [7/1]

Size of the tree : 42
```

```
REPTree
============

number_of_other_loans < 1.5
|   education = graduate : no (38/3) [21/0]
|   education = bachelor : no (128/4) [67/6]
|   education = some college : no (51/13) [18/2]
|   education = highschool or less
|   |   marital_status = married/widowed
|   |   |   loan_duration < 4.5 : no (22/0) [7/0]
|   |   |   loan_duration >= 4.5
|   |   |   |   annual_income < 235 : yes (2/0) [2/0]
|   |   |   |   annual_income >= 235 : no (12/1) [6/2]
|   |   marital_status = single : no (17/4) [18/4]
|   |   marital_status = divorced : yes (6/1) [1/0]
number_of_other_loans >= 1.5
|   annual_income < 245 : yes (20/0) [10/0]
|   annual_income >= 245
|   |   number_of_other_loans < 2.5 : no (34/8) [15/1]
|   |   number_of_other_loans >= 2.5 : yes (25/3) [13/0]

Size of the tree : 18
```

```
REPTree
============

number_of_other_loans < 2.5
|   age < 25.5 : no (24/10) [8/1]
|   age >= 25.5
|   |   monthly_payment_other_loans < 4.5 : no (221/6) [116/7]
|   |   monthly_payment_other_loans >= 4.5
|   |   |   annual_income < 265
|   |   |   |   gender = male
|   |   |   |   |   age < 45.5 : yes (11/1) [5/1]
|   |   |   |   |   age >= 45.5 : no (2/0) [1/0]
|   |   |   |   gender = female : no (2/0) [1/0]
|   |   |   annual_income >= 265
|   |   |   |   age < 39.5 : no (39/1) [22/1]
|   |   |   |   age >= 39.5
|   |   |   |   |   age < 40.5 : yes (3/0) [3/1]
|   |   |   |   |   age >= 40.5 : no (30/3) [11/1]
number_of_other_loans >= 2.5 : yes (23/1) [11/0]
```

```
Size of the tree : 17


REPTree
============

number_of_other_loans < 1.5
|   loan_amount < 215
|   |   age < 33.5
|   |   |   education = graduate : no (12/0) [4/0]
|   |   |   education = bachelor : no (57/3) [49/0]
|   |   |   education = some college
|   |   |   |   loan_duration < 2.5 : yes (4/0) [1/0]
|   |   |   |   loan_duration >= 2.5 : no (11/1) [3/1]
|   |   |   education = highschool or less
|   |   |   |   loan_amount < 117.5 : no (21/1) [9/0]
|   |   |   |   loan_amount >= 117.5 : yes (2/0) [1/0]
|   |   age >= 33.5 : no (133/2) [65/8]
|   loan_amount >= 215
|   |   loan_amount < 225 : yes (4/0) [0/0]
|   |   loan_amount >= 225
|   |   |   marital_status = married/widowed : no (19/1) [9/0]
|   |   |   marital_status = single : no (3/0) [0/0]
|   |   |   marital_status = divorced : yes (5/1) [2/0]
number_of_other_loans >= 1.5
|   annual_income < 285 : yes (31/4) [15/2]
|   annual_income >= 285
|   |   number_of_other_loans < 2.5 : no (35/0) [15/1]
|   |   number_of_other_loans >= 2.5 : yes (18/4) [5/0]

Size of the tree : 24


REPTree
============

number_of_other_loans < 1.5
|   loan_amount < 82.5 : no (125/1) [59/3]
|   loan_amount >= 82.5
|   |   years_with_current_employer < 18.5
|   |   |   annual_income < 235
|   |   |   |   number_of_other_loans < 0.5 : no (9/0) [4/0]
```

| | | | number_of_other_loans >= 0.5 : yes (5/0) [1/0]

| | | annual_income >= 235 : no (123/6) [67/5]

| | years_with_current_employer >= 18.5

| | | annual_income < 365 : no (9/0) [6/0]

| | | annual_income >= 365 : yes (14/5) [4/1]

number_of_other_loans >= 1.5

| annual_income < 235 : yes (14/0) [4/0]

| annual_income >= 235

| | number_of_other_loans < 2.5

| | | years_with_current_employer < 4.5

| | | | education = graduate : no (1/0) [1/0]

| | | | education = bachelor : no (10/4) [8/1]

| | | | education = some college : no (2/0) [1/0]

| | | | education = highschool or less : yes (1/0) [1/0]

| | | years_with_current_employer >= 4.5 : no (24/1) [11/0]

| | number_of_other_loans >= 2.5 : yes (18/4) [11/1]

Size of the tree : 23

REPTree

============

number_of_other_loans < 1.5

| loan_amount < 47.5 : no (70/0) [29/1]

| loan_amount >= 47.5

| | education = graduate : no (20/1) [13/1]

| | education = bachelor

| | | age < 24.5 : yes (2/0) [6/2]

| | | age >= 24.5 : no (99/5) [40/2]

| | education = some college

| | | age < 27.5

| | | | loan_amount < 70 : yes (5/0) [3/1]

| | | | loan_amount >= 70 : no (3/0) [1/0]

| | | age >= 27.5 : no (18/1) [17/2]

| | education = highschool or less : no (58/16) [27/3]

number_of_other_loans >= 1.5

| education = graduate : no (13/0) [4/1]

| education = bachelor

| | monthly_payment_other_loans < 8.5 : yes (14/3) [3/1]

| | monthly_payment_other_loans >= 8.5 : no (16/4) [10/1]

| education = some college

| | years_with_current_employer < 15.5 : yes (15/0) [12/2]

```
|   |   years_with_current_employer >= 15.5 : no (2/0) [1/0]
|   education = highschool or less
|   |   monthly_payment_other_loans < 8 : yes (7/0) [1/0]
|   |   monthly_payment_other_loans >= 8
|   |   |   years_with_current_employer < 15
|   |   |   |   loan_amount < 107.5 : no (2/0) [1/0]
|   |   |   |   loan_amount >= 107.5 : yes (7/0) [7/0]
|   |   |   years_with_current_employer >= 15 : no (4/0) [3/0]

Size of the tree : 29


REPTree
============

number_of_other_loans < 2.5
|   number_of_other_loans < 1.5
|   |   loan_amount < 187.5
|   |   |   education = graduate : no (37/0) [14/0]
|   |   |   education = bachelor : no (124/4) [74/4]
|   |   |   education = some college
|   |   |   |   age < 27.5 : yes (3/0) [3/1]
|   |   |   |   age >= 27.5 : no (23/2) [14/1]
|   |   |   education = highschool or less
|   |   |   |   years_with_current_employer < 5.5
|   |   |   |   |   age < 29.5 : no (10/0) [6/0]
|   |   |   |   |   age >= 29.5
|   |   |   |   |   |   annual_income < 265 : yes (3/0) [1/0]
|   |   |   |   |   |   annual_income >= 265 : no (4/0) [2/0]
|   |   |   |   years_with_current_employer >= 5.5 : no (40/0) [9/0]
|   |   loan_amount >= 187.5
|   |   |   loan_duration < 5.5
|   |   |   |   education = graduate : yes (2/0) [1/0]
|   |   |   |   education = bachelor : no (5/0) [4/1]
|   |   |   |   education = some college : yes (5/2) [0/0]
|   |   |   |   education = highschool or less
|   |   |   |   |   annual_income < 605
|   |   |   |   |   |   loan_purpose = personal : yes (4/0) [1/0]
|   |   |   |   |   |   loan_purpose = business : no (4/2) [5/1]
|   |   |   |   |   annual_income >= 605 : no (2/0) [1/0]
|   |   |   loan_duration >= 5.5 : no (7/0) [1/0]
|   number_of_other_loans >= 1.5
|   |   annual_income < 235 : yes (13/0) [4/0]
```

```
|   |   annual_income >= 235
|   |   |   age < 37.5
|   |   |   |   monthly_payment_other_loans < 24 : no (25/4) [20/4]
|   |   |   |   monthly_payment_other_loans >= 24 : yes (2/0) [1/0]
|   |   |   age >= 37.5 : no (12/0) [5/1]
number_of_other_loans >= 2.5 : yes (30/3) [12/0]

Size of the tree : 35



REPTree
============

number_of_other_loans < 2.5
|   annual_income < 235
|   |   monthly_payment_other_loans < 3.5
|   |   |   age < 36.5
|   |   |   |   years_with_current_employer < 5.5 : no (20/1) [12/2]
|   |   |   |   years_with_current_employer >= 5.5
|   |   |   |   |   loan_amount < 12.5 : yes (3/0) [2/1]
|   |   |   |   |   loan_amount >= 12.5
|   |   |   |   |   |   loan_amount < 35 : no (12/0) [6/0]
|   |   |   |   |   |   loan_amount >= 35
|   |   |   |   |   |   |   annual_income < 155 : yes (3/0) [0/0]
|   |   |   |   |   |   |   annual_income >= 155
|   |   |   |   |   |   |   |   loan_amount < 70 : no (8/0) [6/1]
|   |   |   |   |   |   |   |   loan_amount >= 70
|   |   |   |   |   |   |   |   |   age < 34 : yes (3/0) [1/0]
|   |   |   |   |   |   |   |   |   age >= 34 : no (3/0) [0/0]
|   |   |   age >= 36.5 : no (16/0) [7/0]
|   |   monthly_payment_other_loans >= 3.5 : yes (12/0) [7/0]
|   annual_income >= 235
|   |   marital_status = married/widowed : no (154/10) [87/3]
|   |   marital_status = single
|   |   |   education = graduate : no (18/0) [4/0]
|   |   |   education = bachelor
|   |   |   |   age < 24.5 : yes (2/0) [1/0]
|   |   |   |   age >= 24.5 : no (44/2) [20/3]
|   |   |   education = some college
|   |   |   |   annual_income < 335 : no (2/0) [2/0]
|   |   |   |   annual_income >= 335 : yes (4/0) [0/0]
|   |   |   education = highschool or less : no (9/0) [3/0]
|   |   marital_status = divorced : no (21/6) [8/1]
```

```
number_of_other_loans >= 2.5 : yes (21/3) [12/0]

Size of the tree : 32


REPTree
============

number_of_other_loans < 1.5
|   education = graduate : no (31/2) [19/0]
|   education = bachelor : no (137/8) [58/2]
|   education = some college : no (39/7) [25/5]
|   education = highschool or less
|   |   loan_amount < 122.5 : no (45/5) [21/1]
|   |   loan_amount >= 122.5
|   |   |   annual_income < 410
|   |   |   |   years_with_current_employer < 9.5 : yes (11/0) [4/1]
|   |   |   |   years_with_current_employer >= 9.5 : no (8/2) [2/0]
|   |   |   annual_income >= 410 : no (8/0) [5/0]
number_of_other_loans >= 1.5
|   education = graduate : no (15/1) [3/0]
|   education = bachelor
|   |   age < 32
|   |   |   annual_income < 290 : yes (9/0) [6/2]
|   |   |   annual_income >= 290 : no (3/1) [2/0]
|   |   age >= 32
|   |   |   years_with_current_employer < 2 : yes (2/0) [1/0]
|   |   |   years_with_current_employer >= 2 : no (19/3) [13/6]
|   education = some college : yes (15/5) [11/3]
|   education = highschool or less
|   |   home_ownership_status = own : no (4/2) [2/0]
|   |   home_ownership_status = rent : yes (9/0) [6/0]

Size of the tree : 25
```