International Journal of Intelligent Information Systems
2013; 2(2): 34-39
Published online May 30, 2013 (http://www.sciencepublishinggroup.com/j/ijiis)
doi: 10.11648/j.ijiis.20130202.12

An application of the logitboost ensemble algorithm in loan appraisals

Z. Kirori[1], J. Ogutu[2]

[1]School of Pure & Applied Sciences, Karatina University, Karatina, Kenya
[2]School of Computing & Informatics, University of Nairobi, Nairobi, Kenya

Email address:
zkirori@karatinauniversity.ac.ke(Z. Kirori), jogutu@uonbi.ac.ke(J. Ogutu)

To cite this article:
Z. Kirori, J. Ogutu. An Application of the Logitboost Ensemble Algorithm in Loan Appraisals, *International Journal of Intelligent Information Systems.* Vol. 2, No. 2, 2013, pp. 34-39. doi: 10.11648/j.ijiis.20130202.12

Abstract: Mitigation of credit risk is a key aspect of portfolio management in any financial institution. This is primarily due to difficulties in uncovering uncertainties in information provided by credit applicants and also due to lack of reliable automated techniques that would improve the efficiency of manual underwriting procedures. In this paper, we report on the results of a MSc. Thesis[1] in the application of an ensemble learning algorithm in development of a computer program that can greatly enhance the underwriting process. The implementation was based on the java netbeans development platform to create an interface that was used to train a model and its subsequent use in predicting credit decisions. The results obtained proved that such a mechanism can be applied to augment manual credit appraising processes, especially where large volumes of applications are to be processed within limited timeframes.

Keywords: LogitBoost, Loan Appraising, LDA, Ensemble Learning

1. Introduction

Loans constitute the cornerstone of the banking industry's financial portfolios. The performance of loan contracts in good standing guarantees profitability and stability of a bank. Therefore the screening of the customer's financial history as well as the ability to remain faithful to new financial obligations is a very significant factor before any credit decision is taken and it is a major step in reducing credit risk.

We begin by highlighting the relevant base literature upon which the experiment was setup and the subsequent experimental setup and finally the results obtained with corresponding analysis and conclusion.

1.1. Problem Statement

Despite the increase in consumer loans defaults and competition in the banking market, most of the Kenyan commercial banks are reluctant to use artificial intelligence technologies in their decision-making routines. Generally, bank loan officers rely on traditional methods to guide them in evaluating the worthiness of loan applications. A checklist of bank rules, conventional statistical methods and personal judgment are used to evaluate loan applications. Furthermore, a loan officer's credit decision or recommendation for loan worthiness is subjective.

After some experience, these officers develop their own experiential knowledge or intuition to judge the worthiness of a loan decision. Given the absence of objectivity, such judgment is biased, ambiguous and nonlinear and humans have limited capabilities to discover useful relationships or patterns from a large volume of historical data. Generally, loan application evaluations are based on a loan officers' subjective assessment. Therefore, a knowledge discovery tool is needed to assist in decision making regarding the application.

Further, the complexity of loan decision tools and variation between applications is an opportunity for the use of a machine learning tool to provide learning capability that does not exist in other technologies. Ensemble meta modeling techniques, are empirically some of the best machines learning tools applicable to financial risk analysis.

1.2. Purpose of the Study

The purpose of this study was to develop a loan decision system using the logistic regression Meta modeling algorithm - Logitboost around Java based open source software for the Kenya commercial banks. This is the first empirical research of its kind in our country that addresses in a systematic way the issue of using Meta classifiers in loan ap-

plications. Further, the study champions the use of open source software tools in business intelligence applications.

The general objectives of this study were to:

1) Implement the meta learning algorithm - LogitBoost to develop as system for evaluating credit applications to support loan decisions in Kenyan financial institutions
2) Outline some of the challenges of using the learning algorithm in the decision-making process for the banking industry in Kenya
3) Champion the applicability of Java as an open source software in business intelligence applications

1.3. Significance of the Study

From time immemorial in the banking sector, banks have relied on the personal assessment of loan risks or on the traditional statistical methods to predict the default of loans instead of using a standardized evaluation tool. These traditional methods often require a great deal of subjective input from underwriters, making them un-reliable and often lack empirical and scientific backing. The development of machine learning models and tools has been welcomed as one of the most exciting in business settings. The implementation of such models would considerably improve the quality of decision making and the efficiency of credit analysis processes.

1.4. Scope

The study was limited to the implementation of the LogitBoost meta learning algorithm for classification loan analysis. Further, the study considered a binary output from the classifier, hence dependent variable can only take on accept or reject values with an emphasis on the banking industry in Kenya; though the results can easily be generalized to institutions elsewhere.

1.5. Limitations

Loan appraisal decisions can easily extend beyond the "accept" or "reject" kind of classifications to include such other spectral values as "fairly good", "good" and so on. Although the classifier takes this into account through voting – in which those values that meet certain thresholds are promoted to either of the classification values, most of such incidences are minimal and can be handled through judgmental procedures by re-examining those peculiar cases and applying policies as laid out. Further, the classifier labels every classification instance with a level of confidence value. The study has left such analysis to oversight procedures especially where the confidence level of the classifier does not meet a certain threshold.

2. Related Work

Loan approval is normally to accept applicants with low credit risk, whereas high risk applications are rejected. This makes credit control one of the key concerns in a bank's financial management [1].

The ongoing changes in the banking industry, in the form of new credit regulations, the need for innovative marketing strategies, the ever increasing competition and the constant changes in customer borrowing patterns; call for frequent adjustments to credit management in order to remain competitive. Invariably, the amount of customer data required to effectively screen a loan application is usually huge; often not less than fifteen attributes. The traditional credit appraising techniques based on a hybrid mixture of manual and statistical techniques such as indices and reporting, credit bureau references, post screening, fact act, multiple credit accounts and initial credit line, the manual input are definitely inadequate in modern times. This calls for the use of more efficient and effective loan screening tools and procedures.

Automated techniques have progressively become popular in contemporary loan appraisal processes. However, judgmental inputs such as intuition, policy and information oversights cannot be completely eradicated. One of the earliest automated procedures uses statistical tools which have fallen short of the inherent challenge for today's commercial banks is their desire to understand large amounts of information and reveal useful knowledge to improve decision-making. This is largely because the sustainability of banks depends largely on their abilities to sift through large volumes of data, to extract useful knowledge and enforce this knowledge in their decisions.

Today, lenders are making increased use of new and innovative techniques – the key being data mining and machine learning to evaluate loan applications for business and financial prospects [2, 3]. These techniques have been found to outperform earlier approaches leading to increased competitiveness. Further, ensemble learning algorithms – those that combine a number of base algorithms, through empirical reports typically lead to better results. Credit appraisal often amounts to making a decision whether to grant or to reject an application. This is a classification problem and can easily be implemented using a classification algorithm; the output of which is Boolean or multi-valued. Boosting is one of the most important recent developments in classification methodology. Boosting works by sequentially applying a classification algorithm to reweighted versions of the training data and then taking a weighted majority vote of the sequence of classifiers thus produced [4-6]. For many classification algorithms, this innovative strategy results in dramatic improvements in performance [8]. This is a specialized case of regression analysis over discrete or ordinal values; but basic regression-based learning algorithms have inherent disadvantages. Better algorithms that overcome these pitfalls have been developed and are collectively known as Discriminant Analysis (DA) techniques or simply Meta learning algorithms [3]. One such algorithm that effectively addresses these issues is the LogitBoost Meta classifier - based on the log of the odds ratio for the dependent variable [7, 8].

In the quest to find solutions to loan approval problem [9],

the authors proposed a neural network banking model for the Jordanian banks. Although the model was reported to perform relatively better than models developed using other approaches; as part of the limitations and recommendation, they suggested that such a model is usually a black box and more insight the model parameters was required to make it more effective. Further, they suggested an improvement to the model by introducing a graphical interface for the loans officer.

There have been various other attempts to deal with the loan appraisal problem using various techniques [10] to varied degrees of success.

3. Experiment Design

The solution to the problem was an adaptation of ensemble machine learning strategies where a 'weak' classifier, commonly referred to as a base classifier was boosted through a series of adjustments through weighting and re-sampling to develop a better learner which was an additive aggregate of individual learners. The boosting method was developed around the Probably Approximately Correct (PAC) model that entails transforming 'weak learners' into 'strong learners'. The reported technique derives from the intuitive understanding that instead of putting all the effort on finding highly accurate base classifiers, it becomes sufficient or even desirable to use a set of weaker hypotheses.

3.1. Decision Stump: Base Classifier

A decision stump is a decision tree with only a single root node. It works as follows:

1. Looks at all possible thresholds for each attribute
2. Selects the one with the max information gain
3. Resulting classifier is a simple threshold on a single feature

a) Outputs a +1 if the attribute is above a certain threshold
b) Outputs a -1 if the attribute is below the threshold

3.2. Combining Classifiers

In this study, 'majority voting' was adopted for combining hypothesis from different learners. In majority voting, to predict the class of a new item, each base classifier got to vote for either the 'accept' or the 'reject' class. A accept classification for a loan decision meant pointed to a successful application while a reject classification pointed to the alternative.

It can be proven (as discussed here-under), that under the assumption that all individual classifiers have the same prediction rate and that the distribution of the data correctly classified by each base classifier is independent and random, this is the best possible strategy. Figure 1 illustrates the combination criterion.

3.3. Logistic Regression

The implementation detailed lay in the use of a logistic regression that models the posterior class probabilities Pr (G = k|X = x) for the K classes. In our study, the variable k was bi-valued and took on either 'accept' or 'reject' values and K was set at 2. Logistic regression models these probabilities using linear functions in x while at the same time ensuring they sum to one and remain in [0,1]. The model was specified in terms of K −1 log-odds that separate each class from the base class K.

3.4. Boosting Algorithm

a) With K attributes , there are K different decision stumps to choose from
b) At each stage of boosting

i. given reweighted data from previous stage
ii. Train all K decision stumps
iii. Select the single best classifier at this stage
iv. Combine it with the other previously selected
v. classifiers
vi. Reweight the data
vii. Learn all K classifiers again, select the best, combine,
viii. reweight
ix. Repeat until you have T classifiers selected

3.5. Tools and Equipment

The development platform used for this project mainly included the following open source software products:

3.5.1. Java JDK Software Kit

The Java Development Kit (JDK) which is a Sun Microsystems product released under the GNU General Public License (GPL) was one of the packages used especially for the compilation of the source files.

3.5.2. Java Netbeans IDE

The NetBeans IDE which is a Java based open-source IDE was also used in the development of the system's graphical user interface (GUI) and for coding and testing of the system.

3.5.3. Weka Class API

Weka which is open source software issued under the GNU General Public License providing a collection of machine learning algorithms for data mining tasks was integrated into the development platform.

4. Implementation and Testing

The system was implemented on a Java platform comprising of the JDK compiler, netbeans IDE developer, weka API and the exe4j executable file converter.

4.1. Model Building and Testing Strategies

The model was built using the training dataset and tested using three strategies. We report on cross validation as under.

4.1.1. Cross-Validation Strategy

i. Separate data into fixed number of partitions (or folds)
ii. Select the first fold for testing, whilst the remaining folds are used for training.
iii. Classify and obtain performance metrics.
iv. Select the next partition as testing and use the rest as training data.
v. Classify until each partition has been used as the test set.
vi. Calculate an average performance.

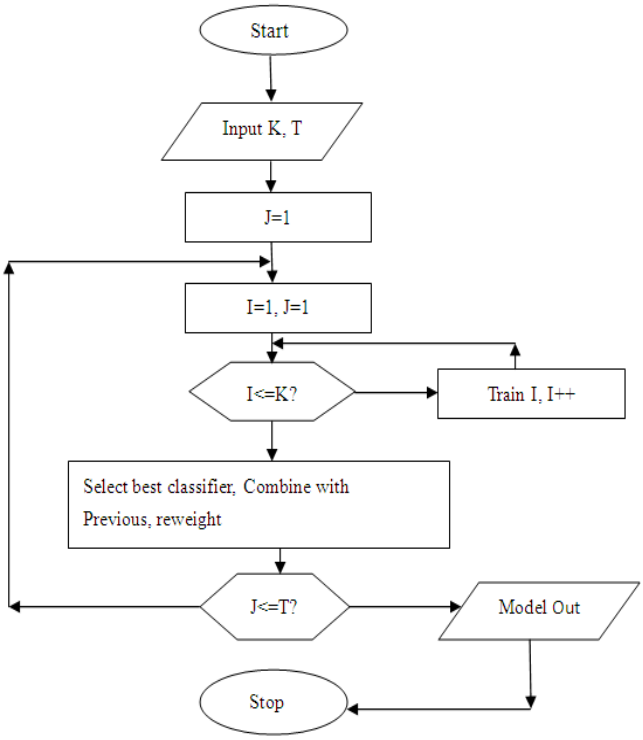

Figure 1 *Boosting Algorithm*

Empirical studies suggest that using 10 partitions (tenfold cross-validation) often yields the same error rate as if the entire data set had been used for training. This and other strategies were used and results compared.

4.1.2. Testing Dataset

This strategy relies on two separate files, one for training and the other for testing. The two files can be generated by portioning a given data set into two and saving them separately.

Table 1. *Test split error*

Correctly Classified Instances	60	68.1818 %
Incorrectly Classified Instances	28	31.8182 %
Root mean squared error	0.45	-
Coverage of cases (0.95 level)	-	97.7273 %
Total Number of Instances	-	88

Table 2. *Test split class accuracy*

Class	Precision	Recall	ROC Area
Accept	0.709	0.918	0.709
Reject	0.444	0.148	0.709
Wtd. Avg.	0.628	0.682	0.709

Table 3. *Test Split Confusion Matrix*

		Classified As	
		A=Accept	**B=Reject**
Class	**A=Accept**	56	5
	B=Reject	23	4

4.1.3. Split Dataset

This strategy is similar to the use of two files as discussed earlier but relies on the learner to automatically partition a given data set into two given a split percentage

4.1.4. Predictions Using a Test File

Options: -F -R -I 15
Number of performed iterations: 15
Time taken to build model: 0.06 seconds
Time taken to test model on training data: 0.01 seconds

5. Findings

The results were interpreted along the following parameters for all the various training and testing strategies.

5.1. Training and Testing Set

5.1.1. Testing Accuracy

The accuracy returned by the training set is 19 correctly classified instances out of 20 instances. This gives an accuracy of 19/20=95%

Table 4. *Test File Predictions*

inst#	actual	Predicted	error	distribution
1	1:Accept	1:Accept		*0.817,0.183
2	2:Reject	2:Reject		0.434,*0.566
3	1:Accept	1:Accept		*0.951,0.049
4	1:Accept	1:Accept		*0.795,0.205
5	2:Reject	2:Reject		0.38,*0.62
6	1:Accept	1:Accept		*0.563,0.437
7	1:Accept	1:Accept		*0.823,0.177
8	1:Accept	1:Accept		*0.821,0.179
9	1:Accept	1:Accept		*0.97,0.03
10	2:Reject	2:Reject		0.17,*0.83
11	2:Reject	1:Accept	+	*0.824,0.176
12	2:Reject	2:Reject		0.434,*0.566
13	1:Accept	1:Accept		*0.824,0.176
14	2:Reject	2:Reject		0.397,*0.603
15	1:Accept	1:Accept		*0.824,0.176
16	2:Reject	2:Reject		0.452,*0.548
17	1:Accept	1:Accept		*0.696,0.304
18	1:Accept	1:Accept		*0.608,0.392
19	2:Reject	2:Reject		0.103,*0.897
20	1:Accept	1:Accept		*0.922,0.078

5.1.2. Precision

Class =Accept: The number of correctly classified instances is 12 and that of instances classified as belong to the class is 13. This gives a precision value of 12/13=0.92

Class =Reject: The number of correctly classified instances is 7 and that of instances classified as belong to the class is 7. This gives a precision value of 7/7=1

5.1.3. Recall

Class =Accept: The number of correctly classified instances is 12 and the number of instances belonging to the class is 12. This gives a recall value of 12/12=1

Class =Reject: The number of correctly classified instances is 7 and the number of instances belonging to the class is 8. This gives a recall value of 7/8=0.88

Figure 2. *Test Split ROC graph*

5.1.4. Nature of ROC

ROC was developed during the World War II to statistically model false positives and false negatives of radar detections. It exhibits better statistical foundations than other performance measure techniques with diverse application in medicine and computing. The ROC graph is a plot of two measures:

Sensitivity: The probability of true classifications given true instances i.e. P(true | true) calculated as a/a+b from a standard confusion matrix

1-Specificity: The probability of true classifications given false instances i.e. P(true | false) calculated as 1- d/c+d

The ROC area has the following indicators:

. 1.0. Indicates a perfect prediction
. 0.9. Excellent prediction
. 0.8. Good prediction
. 0.7. Mediocre prediction
. 0.6. Poor prediction
. 0.5. Random prediction
. <0.5. Indicates something is wrong with the classifier

The ROC produced for the described strategy was as shown in figure 2.The ROC graph is regular with an area of 0.96. The value converted to 1 decimal place, these values indicate a perfect classification

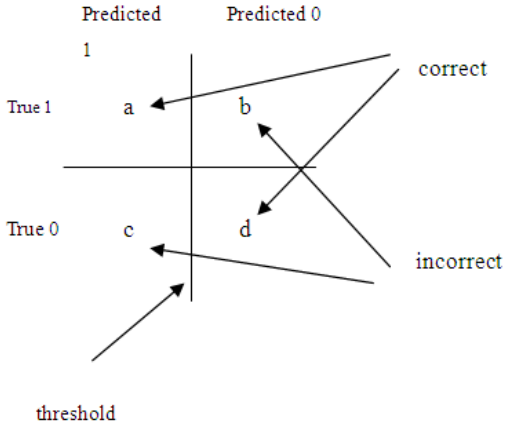

Figure 3. *Confusion Matrix*

6. Discussion

After a successful implementation of the stated system, the following were the key outcomes:

6.1. Model Accuracy

Three options were investigated for training the algorithm namely:

a) The use of single file both for training and testing the model through stratified cross validation. This is a strategy where the training file was portioned into complementary data sets called the training set and the validation set. The technique was applied repeatedly by taking different partitions every time and the results averaged on the respective bounds. The model accuracy using this procedure was 86.86% making it a fairly reliable strategy
b) The use of separate training and testing data sets

returned an accuracy of 95% making it a relatively better strategy

c) The use of a ratio to determine the size of the training and testing files from one data set returned an accuracy of 88.64%

Therefore, it implies from these findings that the use of separate files for training and testing of the model returns the best model accuracy and hence should be adopted.

6.2. Predictive Accuracy

The trained model was subjected to 20 instances of unclassified data which had been carefully selected from a portion of the training and through analysis returned 19 correctly classified instances resulting in a predictive accuracy of 95%

Table 5. *Discussion Summary*

	Training Options	Percentage
Model Accuracy	Single File	86.86
	Separate Files	95.00
	Split Ratio	88.64
Predictive Accuracy		95.00

7. Conclusion and Further Work

Three suggestions are likely improve the model and hence the predictive accuracy of the learner:

7.1. Parameter Tuning

The training and testing procedures can be done severally with different input parameters and file sizes to settle on the most effective set for different learning processes.

7.2. Cost Matrix

A cost matrix can be fined as part of the training procedure that penalizes wrong classifications especially the true negatives for this study. Further, the system can be improved by creating a web-based interface or porting it to a distributed architecture platform.

7.3. Confidence Levels

Finally, as stated earlier in the introduction, it is not prudent to completely rely on an automated credit appraising as some cases might require subjective interpretation and personal judgment. The best aspect of the classification's output is that, the classifier generates levels of confidence on each classification instance whether negative or positive. This is a good basis for manually investigating such cases whose levels of confidence go below a certain threshold. As a conclusion, the reported work indeed confirmed that:

1) Machine learning procedures can be applied in financial modeling applications to augment manual underwriting techniques
2) These procedures can greatly improve the efficiency of such techniques because of their ability to handle large items of data generating very useful statistics
3) This work can be improved through the use of enhanced data set pre-processing procedures, the use of a cost matrix as well as parameter tuning to settle on the most effective set for various data mining requirements.

References

[1] Qiwei G., Binjie L. (2008). Identifying Potential Default Loan Applicants - A Case Study of Consumer Credit Decision for Chinese Commercial Bank. Southwestern University of Finance and Economics, Chengdu, Sichuan, China.

[2] Witten, I. H., and Frank, E. (2008). Data Mining Practical Machine Learning Tools and Techniques. ACM SIGKDD Explorations Newsletter Volume 11 Issue 1, June 2009 Pages 10-18.

[3] Veronica S. M. (2003). Towards the use of C4.5 algorithm for classifying banking dataset. Integral, Vol. 8 No. 2. Pages 105-116

[4] Martin, S. (2008). Ensemble Learning. UCL Department of Computer Science. Accessed from: http://machine-learning.martinsewell.com/ensembles/ensemble-learning.pdf.

[5] Holmes, G., Pfahringer, B., Kirkby, R., Eibe, F., and Hall, M. (2003). Multiclass Alternating Decision Trees. Accessed from: www.cs.waikato.ac.nz/~mhall/pubs.html.

[6] Bauer, E., Kohavi, R. (2006). An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. Machine Learning, vv, 1-38

[7] Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting. Accessed from: http://www.stanford.edu/~hastie/Papers/AdditiveLogisticRegression/alr.pdf. Date:

[8] Agresti A. (2007). Building and applying logistic regression models. An Introduction to Categorical Data Analysis. Hoboken, New Jersey: Wiley. Accessed from: http://onlinelibrary.wiley.com/doi/10.1002/9780470114759.ch5/summary

[9] Shorouq, F. E., Saad, Ghaleb, Y., Ghaleb, A. E. (2010.). Applying Neural Networks for Loan Decisions in the JordanianCommercial Banking System. IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.1

[10] Liang-Hsuan C., and Tai-Wei C. (1999). A fuzzy credit-rating approach for commercial loans: a Taiwan case. Omega, International Journal of Management. Science. Vol 27, 407-419