



UNIVERSITY OF NAIROBI
SCHOOL OF COMPUTING AND
INFORMATICS

A System for Credit Appraising – An application of
the LogitBoost Algorithm

BY

Kirori, Zachary Kamau
P58/9116/2005

Supervisor
Mr. J. Oguttu

August 2011


Project Submitted in partial fulfillment of the requirements of the
Master of Science in Computer Science

DECLARATION

This project, as presented in this report, is my original work and has not been presented for any other University award.

Name **Kirori, Zachary Kamau,**

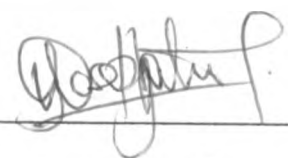
Reg. No. **p58/9116/2011**

Sign _____


Date _____
AUGUST 11, 2011

This project has been submitted as part fulfillment of requirements for the Bachelor of Science in Computer Science of the University of Nairobi with my approval as the University supervisor.

Mr. Ogutu, Joseph

Sign _____


Date _____
12/8/2011

ACKNOWLEDGEMENT

First and foremost, I wish to acknowledge the invisible hand of God for the spiritual guidance I enjoyed throughout the course of this undertaking.

Secondly I would like to appreciate my supervisor Mr. Joseph Ogutu for his counsel on procedural concepts that went into making this project a success

Finally to all of you who in one way or another offered your assistance along the various spheres that form the cornerstones for this research.

DEDICATION

To Cindy Joys, my lovely angel who has always put a smile on my face

FIGURES

Figure 3.1	Flowchart.....	19
Figure 3.2	System Diagram	21
Figure 3.3:	Login Screen	21
Figure 3.4:	Administrator's Panel.....	22
Figure 3.6:	Credit / Loan Appraising Panel.....	23
Figure 3.7:	Administrator's Output Window.....	24
Figure 3.8:	Classifications Output Window	25
Figure 5.1:	Confusion Matrix.....	38
Figure 5.2	X-validation ROC graph	41
Figure 5.3:	Test ROC graph.....	42
Figure 5.4:	Test Split ROC graph.....	45

TABLES

Table 2.1: Confusion Matrix Example	7
Table 3.1: Dataset File Attributes.....	26
Table 4.1: Base Classifier and weights	Error! Bookmark not defined.
Table 4.2: TestFile Predictions	37
Table 5.1: Error on Stratified Cross Validation.....	40
Table 5.2: Cross Validation Class Accuracy	41
Table 5.3: Cross Validation Confusion Matrix	41
Table 5.4: Error on Test Data	42
Table 5.5: Test Confusion Matrix	42
Table 5.6: Training Split Error.....	43
Table 5.7: Training Split Class Accuracy	43
Table 5.8: Training Split CM	43
Table 5.9: Test split error	44
Table 5.10: Test split class accuracy.....	44
Table 5.11: Test Split CM.....	44

Acronyms & Abbreviations

AI – Artificial Intelligence

AIS – Artificial Immune System

ARFF – Attribute Relational File Format

API – Application Programming Interface

CBK – Central Bank of Kenya

CIS – Credit Information Sharing

CRB – Credit Reference Bureau

DA - Discriminant Analysis

DFD – Data Flow Diagram

GPL – General Public License

GUI – Graphical User Interface

IDE – Integrated Development Environment

JDK – Java Development Kit

KBA – Kenya Bankers Association

K-S test – Kolmogorov Smirnov test

Logit - Log of Odds Ratio

LogitBoost – Logistic Boosting

PHP – Hypertext Pre-processor

ROC – Receiver Operating Characteristics

SDLC – Systems Development Life Cycle

SDK – Software Development Kit

SSADM – Structured Systems Analysis & Design Methodology

Weka – Waikato Environment for Knowledge Analysis

WWII – World War II

X-Validation – Cross Validation

ZeroR - 0-R classifier

Abstract

Mitigation of credit risk is a key aspect of portfolio management in any financial institution. This is chiefly 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 document, we report on an application of the logistic regression meta learning algorithm in development of a computer system that could greatly enhance the underwriting process.

The implementation is based on the java platform to create an interface that can be used to train a model and use its predictions for credit decisions. The results obtained prove that such a mechanism can be applied to augment credit appraising processes, especially where large volumes of applications are to be processed within limited timeframes.

Figure 3.2 System Diagram

3.5.2 Screen Design

LOAN APPLICATIONS EVALUATOR

LOGIN

User Type	
User Name	
Password	

Enter Clear Exit

Figure 3.3: Login Screen

3.5	System Design.....	20
3.5.2	Screen Design.....	21
3.5.3	File Design.....	26
3.7	Tools and Equipment.....	29
CHAPTER 4	IMPLEMENTATION, TESTING AND RESULTS.....	30
4.1	Introduction.....	30
4.3	Model Building & Testing Strategies.....	32
4.4	Training / Testing Results.....	33
4.4.1	Training Results: Base Classifiers and their weights.....	33
4.4.2	Testing Results: Using a Test File.....	36
CHAPTER 5	OUTPUT ANALYSIS AND INTERPRETATION.....	38
5.1	Performance Measures.....	38
5.2	Performance analysis results.....	40
5.3	Interpretation of Results.....	45
CHAPTER 6	DISCUSSION.....	50
6.1	FINDINGS.....	50
6.2	SUGGESTIONS.....	51
6.3	CONCLUSION.....	52
6.4	FURTHER WORK.....	53
APPENDICES	54
Appendix A:	References and Bibliography.....	54
Appendix B:	User Manual.....	56
Appendix C:	Samples.....	61
Appendix D:	Data Collection Questionnaire.....	64

CHAPTER 1 INTRODUCTION

1.1 Background

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. Loan approval is normally to good 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 (Oiwai G., 2008).

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 (Chan P.K, n.d.). 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 (Dongsong Z., 2004). 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 value. 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 (Martin S., 2008). For many classification algorithms, this simple strategy results in dramatic improvements in performance. 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 logistic meta learning algorithms (Holmes G., 2003). One such algorithms that effectively addresses these issues is the LogitBoost meta classifier; that is based on the log of the odds ratio for the dependent variable (Friedman J., 2000).

In the quest to find solutions to loan approval problem, Sharouq etal (2010), 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.

In a study entitled “A case based reasoning system for customer credit scoring: comparative study of similarity measures”, Yanwen Dong (n.d.) developed a case based reasoning system to assist in loan decision making. The system that relied on similarities

of past cases to grant or reject loan applications was reportedly successful but remained a study and never made it into an operational system. The author seems to have taken a stab in the dark in this study as evidenced in the conclusion where he reported a non-functional realization of efficiency of the model.

1.2 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 software 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 modeling techniques are empirically some of the best machines learning tools applicable to financial risk analysis.

1.3 Purpose of the Study

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

1.4 General Objectives

The general objectives of this study were:

- i. To implement the meta learning algorithm - LogitBoost to develop as system for evaluating credit applications to support loan decisions in Kenyan financial institutions
- ii. To outline some of the challenges of using the learning algorithm in the decision-making process for the banking industry in Kenya
- iii. To champion the applicability of Java as an open source software in business intelligence applications

1.5 Significance of the Study

For 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.

1.6 Scope of the Study

The study was limited to the implementation of the LogitBoost ensemble learning algorithm around a Java based open source software platform for classification of loan applications. Further, our study considers a binary output from the classifier; hence dependent variable can only take on Accept or Reject values. Finally, the study is limited to the banking industry in Kenya, though the results can easily be generalized to institutions elsewhere.

1.7 Limitations of the Study

Loan appraisal decisions can easily extend beyond the “accept” or “reject” kind of classifications to include such other spectral values as “fairly good”, “Accept” and so on. The study took note of this and extends the number of classes to as many as any given dataset can introduce. However, the results of multi-class predictions were not reported due to lack of appropriate datasets that could support this requirement. Further, the model accuracy could potentially have been improve if a cost matrix was introduced to reinforce the learner by penalizing wrong decisions especially where an organization stands to lose more.

2.1 Introduction

The purpose of this study was to implement the logitboost algorithm in developing a system that can be used to classify loan applications in a financial institution such as a bank. This chapter presents a review of related literature.

The management of assets, liabilities and equity capital lies at the heart of portfolio management in financial institutions. This comes with a great deal of risk mainly through defaulting and proper risk management is therefore a vital and integral part of effective institutional operation. The commonly cited risks include credit, interest, liquidity and operational, most of which emanate from the fundamental and traditional roles of lending and borrowing. Credit, which is associated with the potential variability of the stream of cash flows from an asset is one of the most crucial and is often the cause of failure for banking institutions. In order to be effective in credit risk assessment, monitoring and management, most financial institutions use a variety of methods and tools, the traditional one being credit scoring. For a very long time in the past, these institutions have adopted the system so as to evaluate certain types of loans more objectively, accurately and efficiently. Over the recent past, newer techniques such as credit rating have emerged as a mechanism to better manage credit risk and to improve overall portfolio performance.

Credit risk rating is a summary indicator of risk for an institution's individual credit exposures and is generally assigned at the time of each underwriting or credit approval process and is re-assessed during the credit review process. It functions as the barometer for these institutions to measure their credit risk exposure to each individual customer either in isolation as part of their loan portfolio. The rating allows the institutions to measure the relevant default probabilities at different rating levels more accurately.

Credit Appraisal in the Market

Is the technique by which a banker or for that matter any financial agency including financial institutions estimate the soundness of a credit proposal from the point of view of technical and financial liability or feasibility

Credit Appraisal

The decision to sanction or reject a proposal has to be based on a careful analysis of various facts and data presented by the borrower concerning them and the proposal as assessed by the relationship manager. Such an objective and in-depth scrutiny of the information and data should convince the sanctioning authority that the money lent to the borrower for the stated purpose will be safe and it will be repaid with interest over the stated period, if the assumptions and terms and conditions on which it is sanctioned are fulfilled. This is called the pre-sanction credit appraisal.

Credit appraisal focuses on:

- i. Borrower / Mgt appraisal
- ii. Technical appraisal of the project
- iii. Market appraisal determining the viability of the undertaking
- iv. Financial appraisal determining the variability of individual cash flows to meet the loan repayment requirements

2.2 Common Modeling Techniques

Financial modeling techniques can be roughly segmented into two classes: statistical and non-statistical. The most commonly cited statistical modeling technique in the former is Linear Discriminant Analysis (LDA). LDA has its origins in the discrimination methods suggested by Fisher (1936). Because of its dependence on the assumptions of multivariate normality, independence of predictor variables, and linear separability, LDA has been criticized as having restricted applicability. However, the inequality of covariance matrices, as well as the non-normal nature of the data, a common

phenomenon in credit applications, may not represent critical limitations of the technique (Reichert et al., 1983). Being one of the simpler modeling techniques, LDA continues to be widely used in practice.

A more useful statistical technique cited in most literature is logistic regression analysis - considered the most common technique of model development for initial credit decisions as reported in (Thomas, et al., 2002). For the binary classification problem (i.e., prediction of “Accept” versus “Reject”), logistic analysis takes a linear combination of the descriptor variables and transforms the result to lie between 0 and 1, to equate to a probability.

Where LDA and logistic analysis are statistical classification methods with lengthy histories and varied cited applications, neural network-based classification is a non-statistical technique, which was developed as a result of improvements in desktop computing power. Although neural networks originated in attempts to model the processing functions of the human brain, the models currently in use have increasingly sacrificed neurological rigor for mathematical expediency (Vellido, et al., 1999). Neural networks are utilized in a wide variety of fields and in a wide variety of applications, including finance and specifically, the prediction of consumer risk. In their survey of neural network applications in business, (Vellido et al., 1999), provide a comprehensive overview of empirical studies of the efficacy of neural networks in credit evaluation and decision-making. They highlight that neural networks did outperform “other” (both statistical and non-statistical) techniques, but not consistently.

2.3 Credit Appraising

Attempts to effectively deal with loan appraising have been going on for centuries through theoretical, empirical studies as well as the creating of statistical and computing models and systems in a wide range of disciplines.

When assigning a loan application to a particular grade in banking institutions, Crouhy et al. (2001) suggest that banks should analyze three different categories of variables namely quantitative, qualitative and legal. The quantitative analysis concentrates mainly on financial analysis and is often based on a firm’s financial reports. The four main quantitative factors used in the assessment model include net income, total

operating income, total equity capital and total asset values. These factors allow the banks to calculate a variety of ratios including Return on Assets (ROA), Return on Equity (ROE) and asset utilization (Au).

As for the qualitative analysis, the principal concern is the quality of a borrower's management and legal analysis refers to the capacity to borrow. This means that a bank must ensure that a customer requesting to borrow has the authority and the legal standing to sign a binding agreement.

In spite of the inclination of this study towards banking institutions, the study can clearly be generalized to encompass any such institution whose financial portfolio lies mainly in credit advance and borrowing.

Despite the advances in technology that allow the development of intelligent systems, or statistical classification models, human judgment is still an important ingredient in the credit risk assessment process. According to Treacy and Carey (2000), the rating process almost always involves the exercise of human judgment because factors to be considered in assigning a rating and the weighting given to each factor can differ significantly with borrowers. Indeed experienced lenders take credit ratings and reports as inputs for decision making process. The key reason for the models to be tempered with judgment and common sense is because, they do not, fully explain the subjective factors involved in the rating. Especially for large exposures, the benefits of such accuracy may outweigh the higher costs of a judgmental system. Because of the high cost involved, in general, banks produce credit ratings for business and institutions only.

In the quest to find solutions to loan approval problem, Sharouq et al (2010), 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.

In a study entitled “A case based reasoning system for customer credit scoring: comparative study of similarity measures”, Yanwen Dong developed a case based reasoning system to assist in loan decision making. The system that relied on similarities of past cases to grant or reject loan applications was reportedly successful but remained a study and never made it into an operational system. The author seems to have taken a stab in the dark in this study as evidenced in the conclusion where he reported a non-functional realization of efficiency of the model.

2.4 Modern Approaches to Credit Scoring

The modern approaches for credit appraisal are statistical in nature. These approaches are more objective as they are based on some statistical model. One of the commonly used approach is credit scoring

Credit Scoring

Traditionally, banks were using the method of analyzing financial statement of the applicant by which the bank was able to evaluate the applicant’s capacity to pay back the loan. Though the applicant may be financially sound to pay, it was very difficult to identify whether he or she has the willingness to pay the loan.

When the demand for consumer credits in retail markets is fast increasing, banks must have a system by which they are able to process the credit applicants professionally and at the same time to identify the potential default risk of the borrower.

Most banks presently use credit scoring model to evaluate the loan applications they receive from consumers. Banks, credit card providers, mortgage lenders and other loan providers develop their own internal credit scoring models on retail lending and use these models to evaluate their applicants. With the intro of credit scoring model in the banks, often the customer can phone in with loan request and within the shortest possible time, the bank can convey their decision. Usually the credit scoring systems are based on

discriminant models or related techniques in which variables are used jointly to establish a numerical score or ranking

Corporate Credit

This is a contractual agreement in which a corporation receives something of value now and agrees to repay the lender at some later date. It is almost identical to personal credit except it is a business entity, instead of an actual person, that receives corporate credit from lenders.

2.5 Methods of Model Evaluation

The fundamental concern of modeling techniques is an improvement in predictive accuracy. In customer risk classification, an improvement in predictive accuracy of even a fraction of a percentage can translate into significant savings. However, how can the analyst know if one model represents an improvement over a second model? The answer to this question may change based upon the selection of evaluation method. As a result, analysts who utilize prediction models for binary classification, have a need to understand the circumstances under which each evaluation method is most appropriate.

In the context of predictive binary classification models, one of four outcomes is possible: (i) a true positive – e.g., a good credit risk is classified as “Accept”; (ii) a false positive – e.g., a bad credit risk is classified as “Accept”; (iii) a true negative – e.g., Reject credit risk is classified as “Reject”; (iv) a false negative – e.g., a good credit risk is classified as “Reject”.

In principle, each of these outcomes would have some associated “loss” or “reward”. In a credit-lending context, a true positive “reward” might be a qualified person obtaining a needed mortgage with the bank reaping the economic benefit of making a correct decision. A false negative “loss” might be the same qualified person being turned down for a mortgage. In this instance, the bank not only has the opportunity cost of losing a good customer, but also the possible cost of increasing its competitor’s business.

It is often assumed that the two types of incorrect classification – false positives and false negatives – incur the exact same loss (Hand, 2001). If this is truly the case, then a simple “global” classification rate could be used for model evaluation. For instance, suppose a hypothetical classification model produced the following confusion matrix:

	True Accept	True Reject	Total
Predicted Accept	250	50	300
Predicted Reject	50	150	200
Total	300	200	500

Table 2.1: Confusion Matrix Example

This model would have a global classification accuracy of 80% ($250/500 + 150/500$). This simple metric is reasonable if the costs associated with each error are known (or assumed) to be the same. If this were the case, the selection of a “better” model would be easy – the model with the highest classification accuracy would be selected. Even if the costs were not equal, but at least understood with some degree of certainty, the total loss associated with the selection of one model over another could still be easily evaluated based upon this confusion matrix. For example, the projected loss associated with use of a particular model can be represented by the loss function:

$$L = \pi_0 f_0 c_0 + \pi_1 f_1 c_1$$

where π_i is the probability that an object comes from class i (the prior probability), f_i is the probability of misclassifying a class i object, and c_i is the cost associated with misclassifying an observation into that category and, for example, 0 indicates a “bad” credit risk and 1 indicates a “good” credit risk. Assessment of predictive accuracy would then be based upon the extent to which this function is minimized. West (2000) uses a similar cost function to evaluate the performance of several statistical and non-statistical modeling techniques, including five different neural network models. Although the author was able to select a “winning” model based upon reasonable cost assumptions, the “winning” model would differ as these assumptions changed.

A second issue when using a simple classification matrix for evaluation is the problem that can occur when evaluating models dealing with rare events. If the prior probability of an occurrence were very high, a model would achieve a strong prediction rate if all observations were simply classified into this class. However, when a particular observation has a low probability of occurrence (e.g., cancer, bankruptcy, tornadoes, etc.), it is far more difficult to assign these low probability observations into their correct class (where possible, this issue of strongly unequal prior probabilities can be addressed during model development or network training by contriving the two classes to be of equal size, but this may not always be an option).

The difficulty of accurate rare class assignment is not captured if the simple global classification is used as an evaluation method (Gim, 1995). Because of the issue of rare events and imperfect information, the simple classification rate should very rarely be used for model evaluation. However, a quick scan of papers which evaluate different modeling techniques will reveal that this is the most frequently utilized (albeit weakest due to the assumption of perfect information) method of model evaluation.

One of the most common methods of evaluating predictive binary classification models in practice is the Kolmogorov-Smirnov statistic or K-S test. The K-S test measures the distance between the distribution functions of the two classifications (e.g., good credit risks and bad credit risks). The score that generates the greatest separability between the functions is considered the threshold value for accepting or rejecting a credit application. The predictive model producing the greatest amount of separability between the two distributions would be considered the superior model.

Hand (2002) criticizes the K-S test for many of the same reasons outlined for the simple global classification rate. Specifically, the K-S test assumes that the relative costs of the misclassification errors are equal. As a result, the K-S test does not incorporate relevant information regarding the performance of classification models (i.e., the misclassification rates and their respective costs). The measure of separability then becomes somewhat hollow.

In some instances, the researcher may not have any information regarding costs of error rates, such as the relative costs of one error type versus another. In almost every circumstance, one type of misclassification will be considered more serious than another.

However, a determination of which error is the more serious is generally less well defined or may even be in the eye of the beholder. For example, in a highly competitive business environment is a worse mistake to turn away a potentially valuable customer to a competitor? Or is a worse mistake to accept a customer that does not meet financial expectations? The answers are not always straightforward. As a result, the cost function outlined above, may not be applicable.

One method of evaluation, which enables a comprehensive analysis of all possible error severities, is the Receiver Operating Characteristics (ROC) curve. It was first applied to assess how well radar equipment in WWII distinguished random interference or “noise” from the signals that were truly indicative of enemy planes (Swets, et al., 2000). ROC curves have since been used in fields ranging from electrical engineering and weather prediction to psychology and are used almost ubiquitously in the literature on medical testing to determine the effectiveness of medications. The ROC curve plots the sensitivity or “hits” (e.g., true positives) of a model on the vertical axis against 1-specificity or “false alarms” (e.g., false positives) on the horizontal axis. The result is a bowed curve rising from the 45 degree line to the upper left corner – the sharper the bend and the closer to the upper left corner, the greater the accuracy of the model.

The area under the ROC curve is a convenient way to compare different predictive classification models when the analyst or decision maker has no information regarding the costs or severity of classification errors. This measurement is equivalent to the Gini index (Thomas et al., 2002) and the Mann-Whitney-Wilcoxon test statistic for comparing two distributions (Hanley and McNeil, 1982, 1983) and is referred in the literature in many ways, including “AUC”, the c-statistic, and “ θ ”. For example, if observations were assigned to two classes at random, such that there was equal probability of assignment in either class, the ROC curve would follow a 45-degree line emanating from the origin. This would correspond to $\theta = .5$. A perfect binary classification, $\theta=1$, would be represented by an ROC “curve” that followed the y-axis from the origin to the point 0,1 and then followed the top edge of the square. The metric θ can be considered as an averaging of the misclassification rates over all possible choices of the various classification thresholds. In other words, θ is an average of the diagnostic performance of a particular model over all possible values for the relative misclassification severities (Hand, 2001). The interpretation of θ , where a “good” credit

risk is scored as a 1 and a “bad” credit risk is scored as a 0, is the answer to the question – “Using this model, what is the probability that a truly ‘good’ credit risk will be scored higher than a “bad” credit risk”? Formulaically, θ can be represented as,

$$\theta = \int F(p|0)dF(p|1)dp,$$

where $F(p|0)$ is the distribution of the probabilities of assignment in class 0 (classification of “bad” credit risk) and $F(p|1)$ is the distribution of the probabilities of assignment in class 1 (classification of “Accept” credit risk). An important limitation to note when using θ , is that in practice, rarely is nothing is known about the relative cost or severity of misclassification errors. Similarly, it is rare that are all threshold values relevant.

2.6 Banking Industry in Kenyan

The Banking industry in Kenya is governed by the Companies Act, the Banking Act, the Central Bank of Kenya Act and the various prudential guidelines issued by the Central Bank of Kenya (CBK). The banking sector was liberalized in 1995 and exchange controls lifted. The CBK, which falls under the Minister for Finance docket, is responsible for formulating and implementing monetary policy and fostering the liquidity, solvency and proper functioning of the financial system.

Sharing information on Performing Loans

Section 31 (3) (c) of the Banking Act allows the CBK and institutions licensed under the

Banking Act to exchange information on such other information as is reasonably required for the proper discharge of their functions. Regulation 14(2) of the CRB Regulations read alongside Regulation 14 (3) and Regulation (2) outlines the categories of information on performing loans which the law allows member banks to exchange.

Why credit information sharing is being introduced in Kenya

Many borrowers make a lot of effort to repay their loans, but do not get rewarded for it because this good repayment history is not available to the bank that they approach for new loans. On the other hand, whenever borrowers fail to repay their loans, banks are forced to pass on the cost of defaults to other customers through increased interest rates and other fees. Put simply - good borrowers are paying for bad. This is unfair. Credit reporting allows banks to better distinguish between good and bad borrowers. Someone who has failed to pay their loan at one bank will not simply be able to walk to another bank to get another loan without the banks knowing about it. Over time better information on potential borrowers should mean that it will be both cheaper and easier to obtain loans.

2.7 LogitBoost

Boosting is a machine learning meta-algorithm for performing supervised learning. Boosting is based on the question posed by Kearns: can a set of weak learners create a single strong learner? A weak learner is defined to be a classifier which is only slightly correlated with the true classification (it can label examples better than random guessing). In contrast, a strong learner is a classifier that is arbitrarily well-correlated with the true classification.

In statistics, logistic regression (logit) which is the inverse of sigmoidal logistic function is used for predicting the probability of occurrence of an event by fitting data to

a logit function $f(z) = \frac{e^z}{e^z + 1} = \frac{1}{1 + e^{-z}}$ logistic curve. It is a generalized linear model used for binomial regression which is a technique in which the response, often referred to as Y is the result of a series of Bernoulli trials, or a series of one of two possible disjoint outcomes; traditionally denoted "success" or 1_2 and "failure" or 0). In binomial regression, the probability of a success is related to explanatory variables or regression variables that may be either numerical or categorical. The logit of a

number p between 0 and 1 is given by the formula:

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \log(p) - \log(1-p).$$

The odds in favor of an event or a proposition are expressed as the ratio of a pair of integers, which is the ratio of the probability that an event will happen to the probability that it will not happen. Therefore, if p is a probability then $p/(1-p)$ is the corresponding odds and the logit of the probability is the logarithm of the odds. Similarly the difference between the logits – which makes the measure symmetric with respect to the ordering of groups, of two probabilities is the logarithm of the odds ratio (R). The logarithm is a measure of effect size, describing the strength of association or non-independence between two binary data values or the ratio of the odds of an event occurring in one group to the odds of it occurring in another group.

$$\log(R) = \log\left(\frac{p_1/(1-p_1)}{p_2/(1-p_2)}\right) = \log\left(\frac{p_1}{1-p_1}\right) - \log\left(\frac{p_2}{1-p_2}\right) = \text{logit}(p_1) - \text{logit}(p_2).$$

An odds ratio of 1 indicates that the condition or event under study is equally likely to occur in both groups. An odds ratio greater than 1 indicates that the condition or event is more likely to occur in the first group. And an odds ratio less than 1 indicates that the condition or event is less likely to occur in the first group

The logistic function is useful because it can take as an input any value from negative infinity to positive infinity, whereas the output is confined to values between 0 and 1. The variable z represents the exposure to some set of independent variables, while $f(z)$ represents the probability of a particular outcome, given that set of explanatory variables. The variable z is a measure of the total contribution of all the independent variables used in the model and is known as the logit. The variable z is usually defined as

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \beta_k x_k,$$

where β_0 is called the "intercept" and $\beta_1, \beta_2, \beta_3,$ and so on, are called the "regression coefficients" of x_1, x_2, x_3 respectively. The intercept is the value of z when the values of all independent variables are zeros (e.g. the value of z in someone with no risk factors). Each of the regression coefficients describes the size of the contribution of that risk

factor. A positive regression coefficient means that the explanatory variable increases the probability of the outcome, while a negative regression coefficient means that the variable decreases the probability of that outcome; a large regression coefficient means that the risk factor strongly influences the probability of that outcome, while a near-zero regression coefficient means that that risk factor has little influence on the probability of that outcome.

Logistic regression is a useful way of describing the relationship between one or more independent variables (e.g., age, sex, etc.) and a binary response variable, expressed as a probability, that has only two values, such as having cancer ("has cancer" or "doesn't have cancer"). The logits of the odds, of the unknown binomial probabilities are modeled as a linear function of the X_i .

$$\text{logit}(p_i) = \ln \left(\frac{p_i}{1 - p_i} \right) = \beta_0 + \beta_1 x_{1,i} + \dots + \beta_k x_{k,i}.$$

Decision Stump: Base Classifier

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

- i. Looks at all possible thresholds for each attribute
- ii. Selects the one with the max information gain
- iii. 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

3.1 Introduction

This chapter discusses key aspects of system design, system methodology, data collection techniques, screen and algorithm design as well as tool selection and input requirements that were used to come up with the system.

3.2 The Solution

The solution to the problem was an adaptation of ensemble machine learning strategies where a potentially ‘weak’ classifier, commonly referred to as a base classifier is boosted through a series of adjustments through weighting or re-sampling to develop a better learner which is an additive aggregate of individual learners. The boosting method is developed around the Probably Approximately Correct (*PAC*) model that entails transforming *weak learners* into *strong learners*. It 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.

3.2.1 Combining classifiers

In this study, *majority voting* was the approach 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. The final decision was formed by selecting the class with maximum number of votes as depicted by the voting equation below:

$$P(\text{vote}(x) = y(x)) = \sum_{i=1}^{k/2} \binom{k}{i} \epsilon^i (1-\epsilon)^{k-i}$$

It can be proven 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.

3.2.2 Logistic Regression

The implementation detailed shall lay 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 is bi-valued and took on either 'Accept' or 'Reject' values and K is 2. It follows from this argument that given estimates for the class probabilities as captured in the resulting model, it is possible to classify unseen instances using the given relation

$$k^* = \max (k) [\Pr (G = k|x = X)].$$

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 is specified in terms of $K - 1$ log-odds that separate each class from the base class K .

3.3 Methodology

The development methodology adopted for this study roughly corresponds to the structured systems analysis and design model (SSADM) and comprised of the following steps:

- i. Background Study of the meta modeling
- ii. Review of related literature
- iii. Design of Algorithms, DFDs, files and the interface
- iv. System implementation and testing
- v. Results and Analysis
- vi. Conclusion and Recommendations
- vii. Documentation

3.3 Data Collection Techniques

Survey

The survey method was used to collect primary data from the various sampled banks through bank representatives and customers. The required data was collected to study the issues in question. Two types of data were relied on:

Primary: First hand information collected from the various sampled players in the banking industry using structured questionnaires as well as interviews with stakeholders such as credit control personnel.

Secondary Collected mainly from literary sources such as books, journals and the internet.

Interviews

Face -to -face interviews have a distinct advantage of enabling the researcher to establish rapport with potential participants and therefore gain their cooperation. These interviews were organized and conducted to gain insight into the underwriting procedures.

Telephone interviews are less time consuming and less expensive and the researcher has ready access to anyone on the planet who has a telephone. This technique was also employed in cases where the former technique was inapplicable and also to set appointments for such meetings.

Structured questionnaires were developed and given to a sample of the target population as a way of augmenting the interviewing technique. This technique was adopted because of its inherent advantage in which the respondents are said to be more truthful while responding to the questionnaires regarding controversial issues in particular due to the fact that their responses are anonymous.

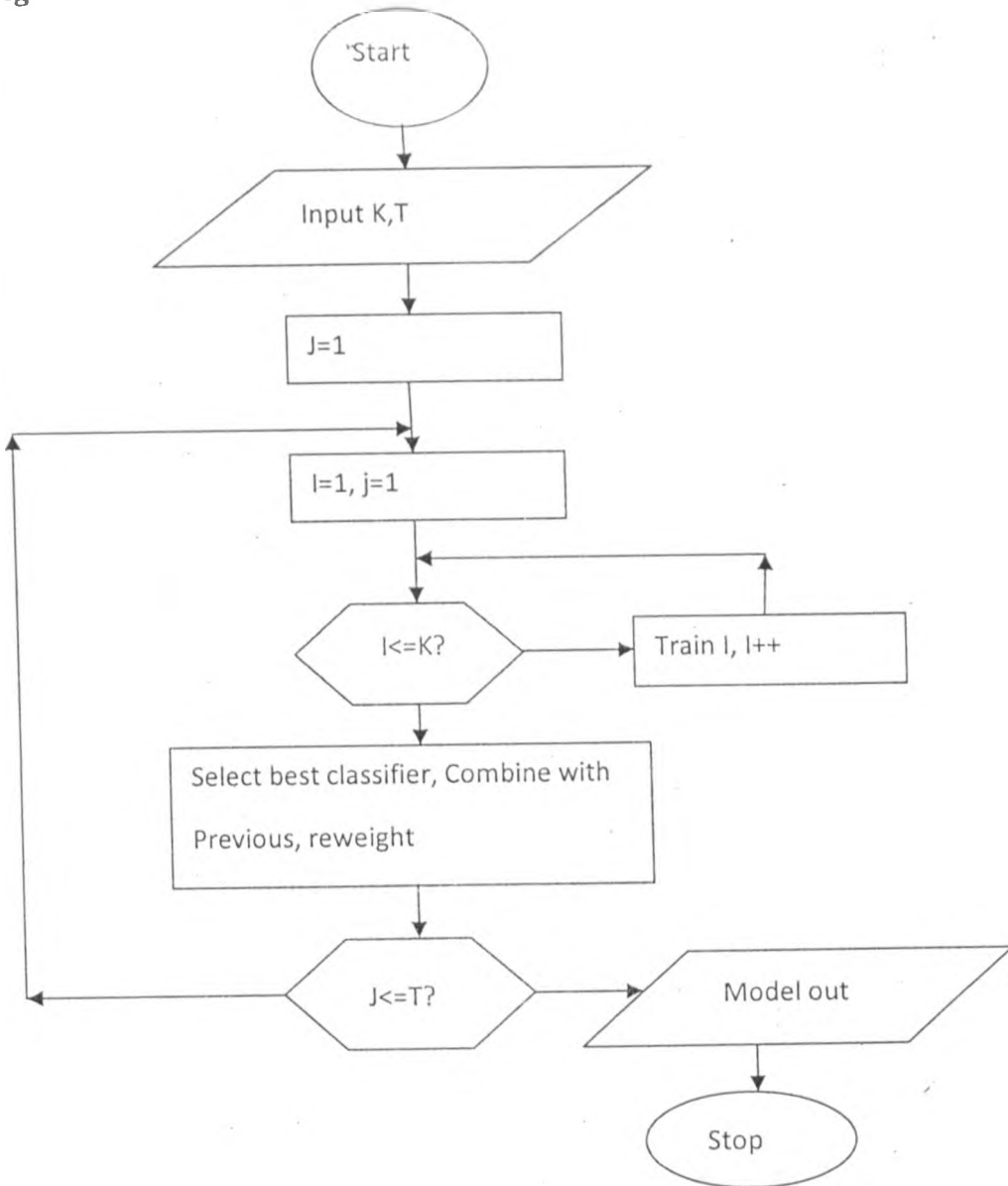
Case Study

Equity Bank - one of the key financial institutions in Kenya was selected to form the basis for the study. This included the study on the bank's 'general' lending policies, credit appraising in use and the general experience that the bank's staff could report on with respect to credit financing.

3.4 Algorithm Design: 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 classifiers
 - v. Reweight the data
 - vi. Learn all K classifiers again, select the best, combine, reweight
 - vii. Repeat until you have T classifiers selected

Figure 3.1 Flowchart



3.5 System Design

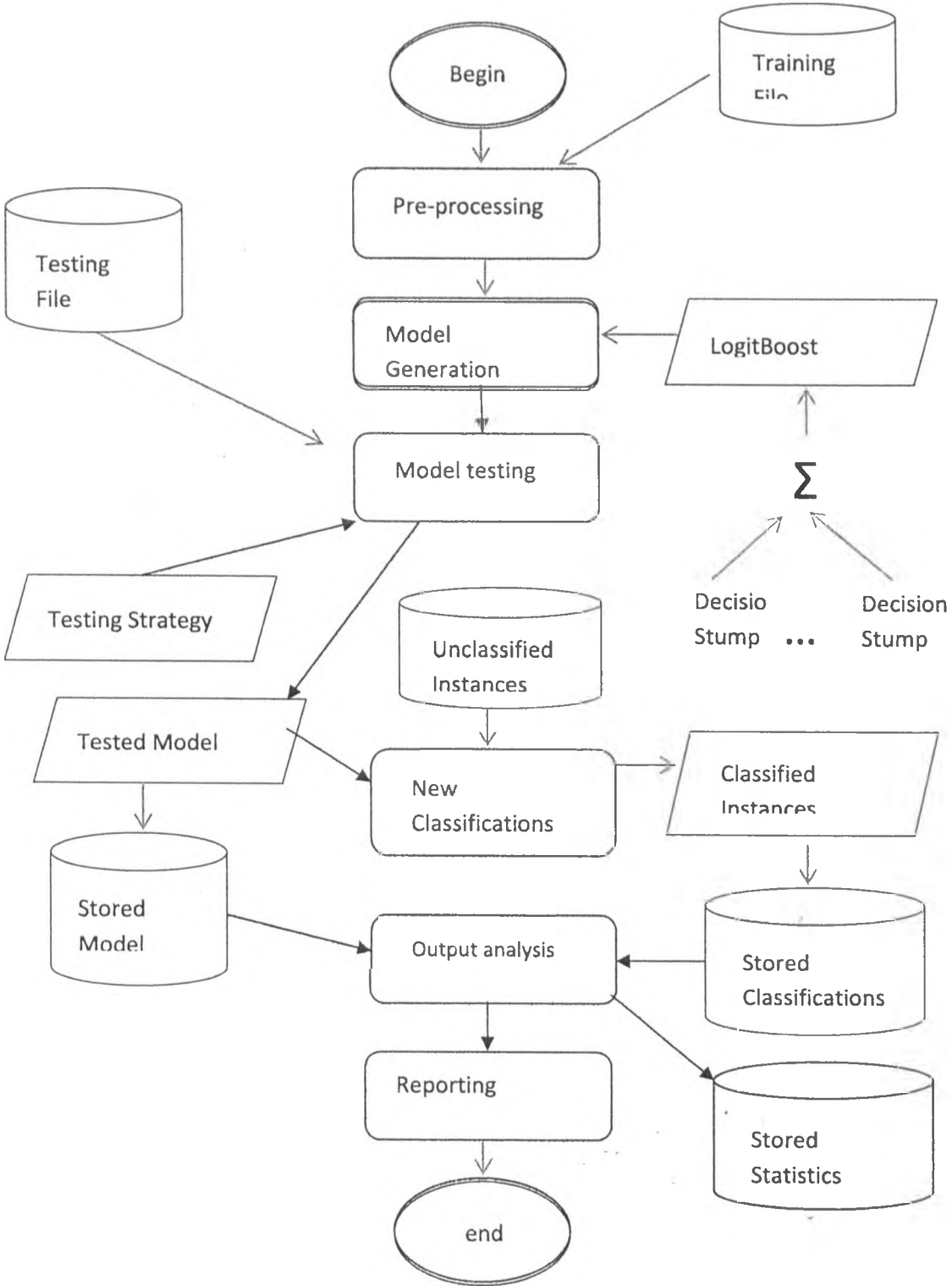


Figure 3.2 System Diagram

3.5.2 Screen Design

LOAN APPLICATIONS EVALUATOR

LOGIN

User Type

User Name

Password

Enter

Clear

Exit

Figure 3.3: Login Screen

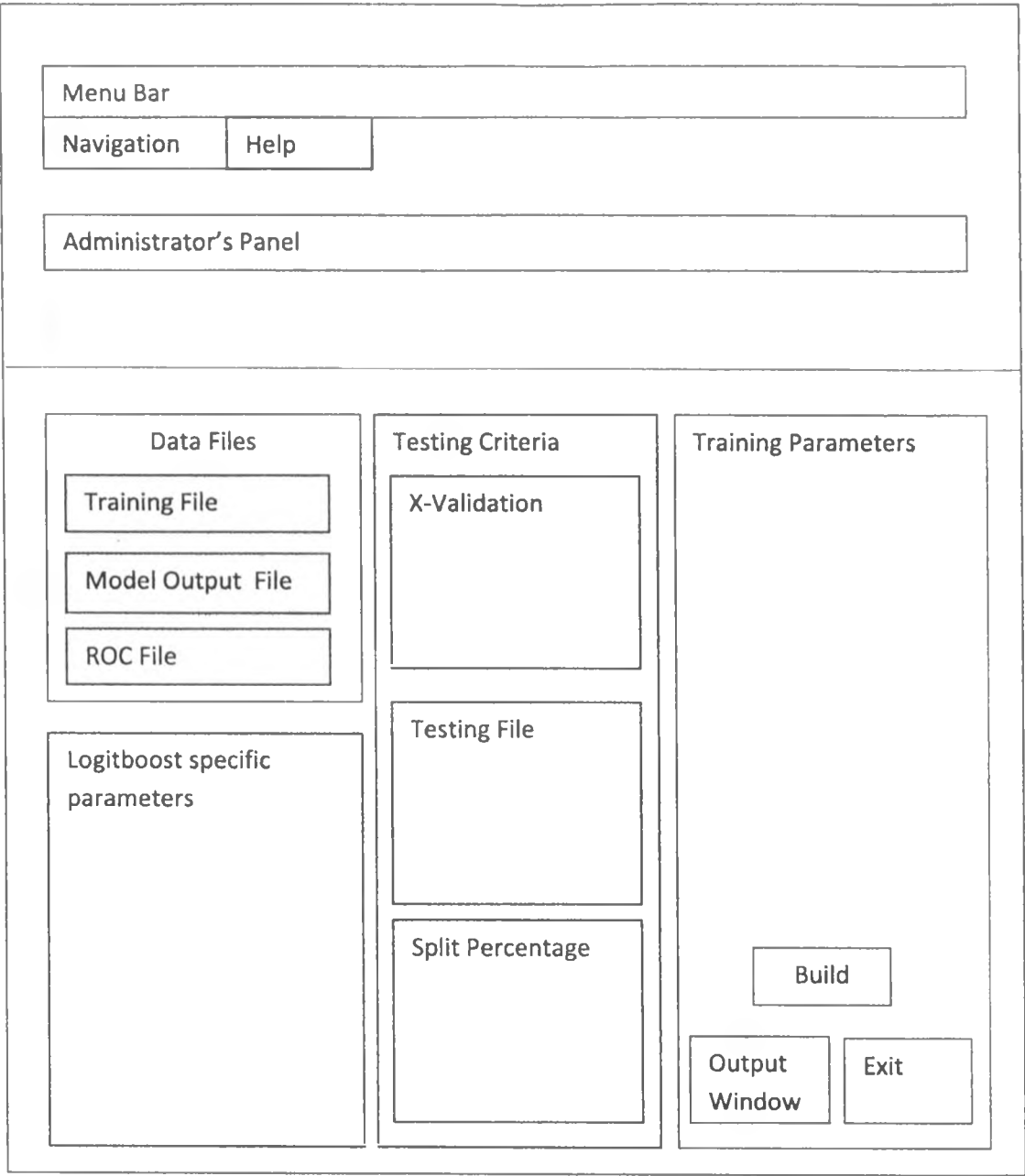


Figure 3.4: Administrator's Panel

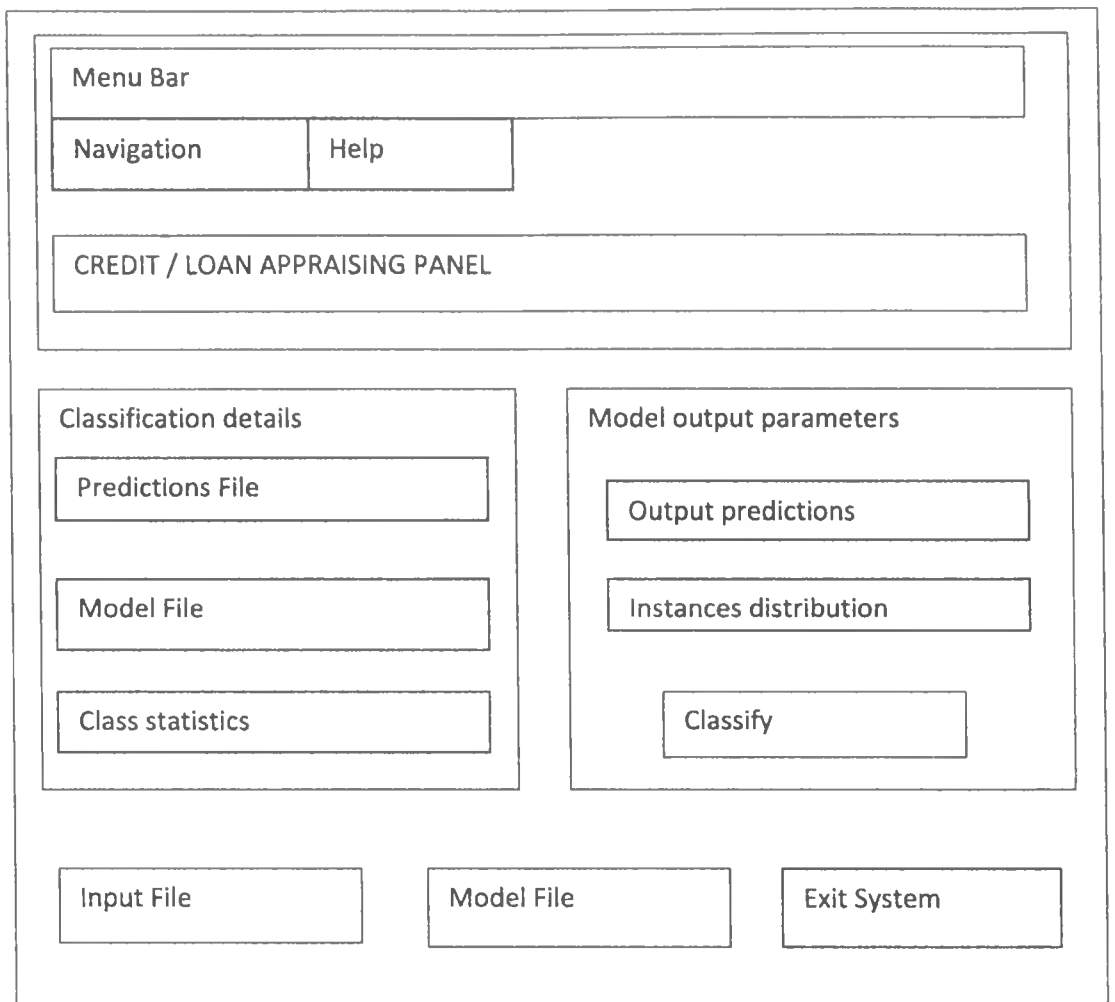


Figure 3.6: Credit / Loan Appraising Panel

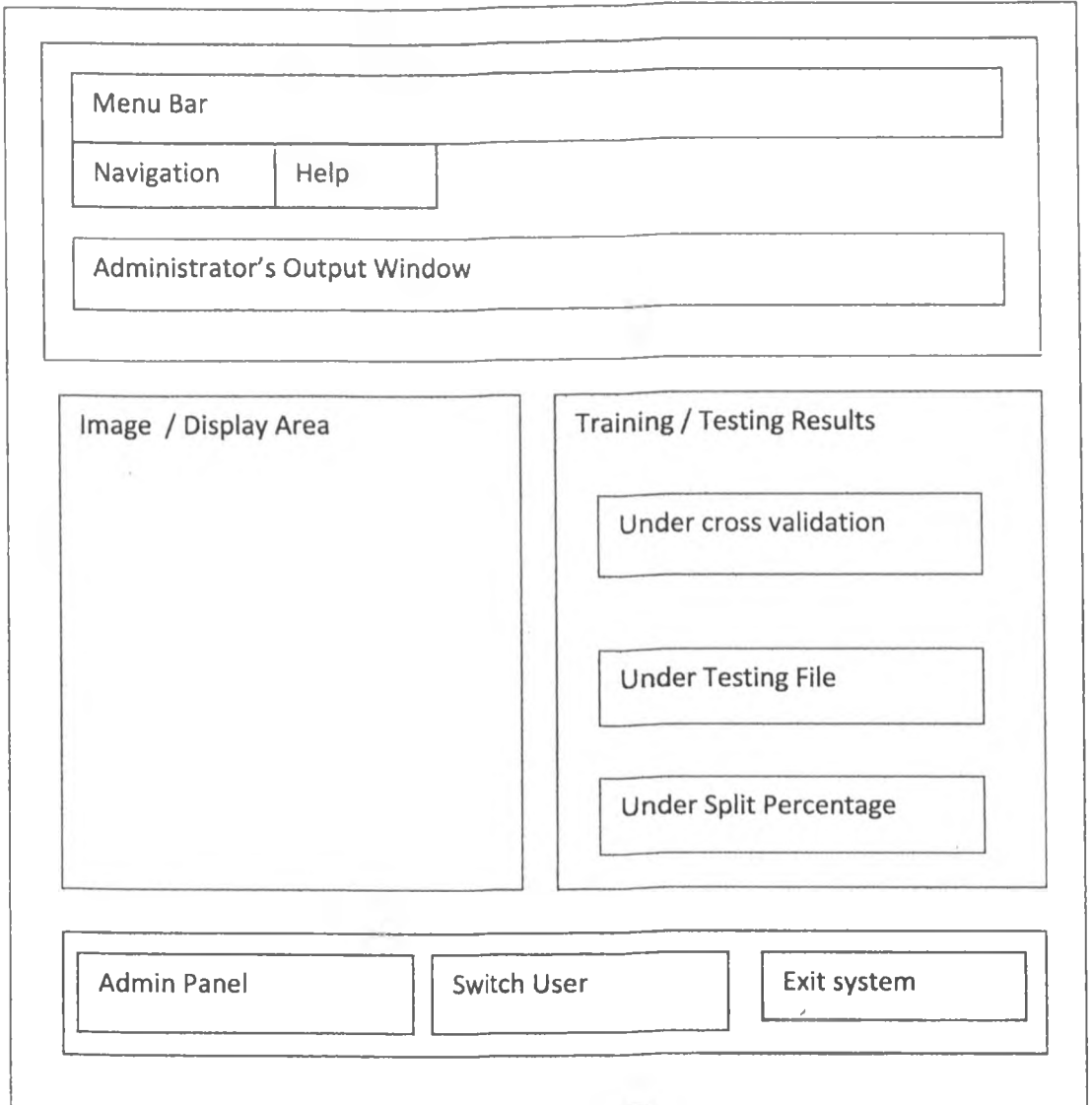


Figure 3.7: Administrator's Output Window

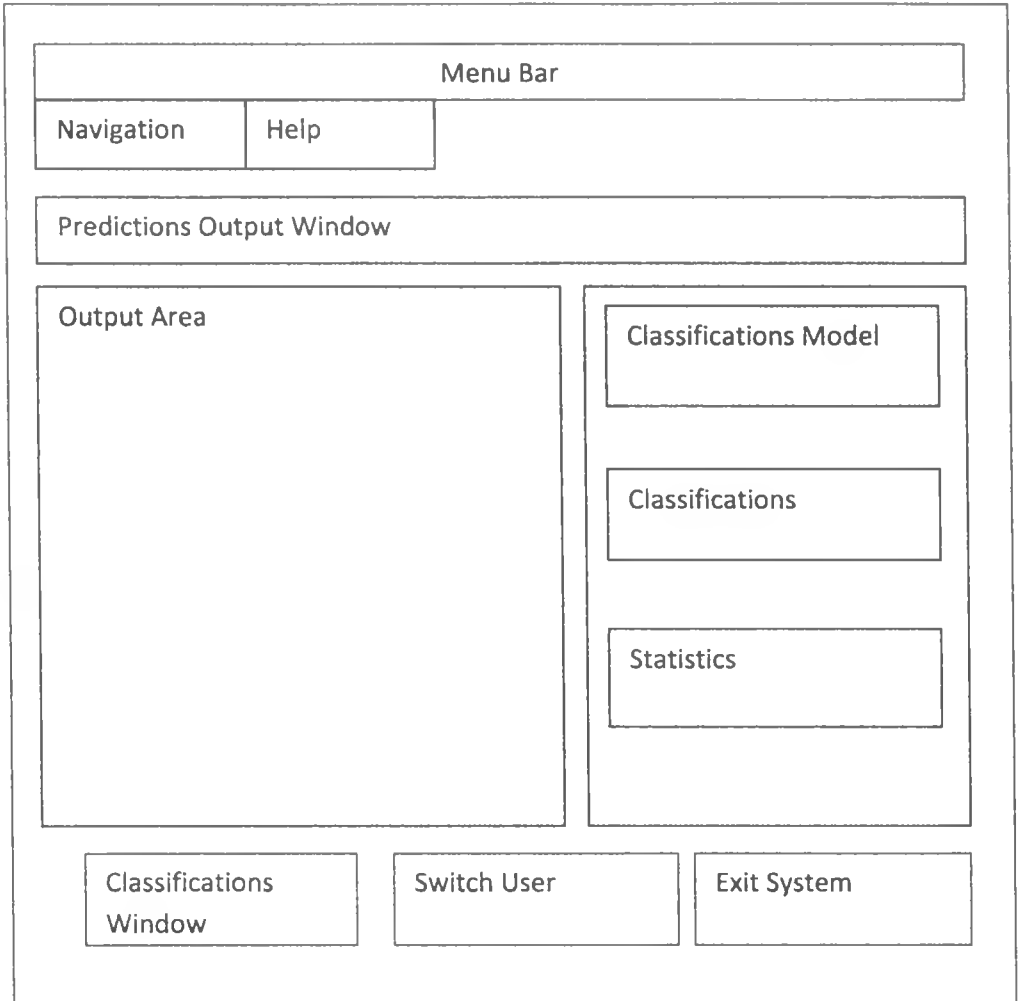


Figure 3.8: Classifications Output Window

3.5.3 File Design

Conventions

File Type *ARFF*

Comment: % a-comment

Internal Name of a dataset: @relation a-name

Nominal Attributes: @attribute attribute-Name {comma-separated list of attribute values}

Numeric attributes: @attribute attribute-Name attribute-Type

Training and Test Files

The target dataset was retrieved from the internet as given by Professor Dr. Hans Hofmann of the University of Humburg in Germany entitled “German credit dataset”

The dataset has a total of 20 attributes normalized as:

Table 3.1: Dataset File Attributes

#	Description	Type	Typical Values
1	Status of existing account	qualitative	< KES 0
			< KES 200
			>= KES 200
			No account
2	Duration (Months)	numeric	
3	Credit history	qualitative	no credits / all duly paid
			all credits at this bank paid back duly
			existing credits paid back duly

			delay in paying off in the past
			critical account/other credits existing (not at this bank)
4	Purpose	qualitative	car (new)
			car (used)
			furniture/equipment
			radio/television
			domestic appliances
			Repairs
			education
			(vacation - does not exist?)
			retraining
			business
			others
5	Credit amount	numerical	
6	Savings account/bonds	qualitative	< 100 DM
			< 500 DM
			< 1000 DM
			>= 1000 DM
			unknown/ no savings account
7	Present employment since	qualitative	unemployed
			< 1 year
			< 4 years
			< 7 years
			>= 7 years
8	Installment rate in %ge of disposable income	numerical	
9	Personal status and sex	qualitative	male : divorced/separated
			female : divorced/separated/married
			male : single
			male : married/widowed

			female : single
10	Other debtors / guarantors	qualitative	none
			co-applicant
			guarantor
11	Present residence since	numerical	
12	Property	qualitative	real estate
			building society
			car or other
			unknown / no property
13	Age in years	numerical	
14	Other installment plans	qualitative	Bank
			Stores
			none
15	Housing	qualitative	rent
			own
			for free
16	No. of existing credits at this bank	numerical	
17	Job	qualitative	unemployed/ unskilled
			unskilled – resident
			skilled employee / official
			management/ self-employed
18	dependants	numerical	
19	Telephone	qualitative	None
			yes
20	foreign worker	qualitative	Yes
			No

3.7 Tools and Equipment

Software

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

i. 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.

ii. 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.

iii. 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.

iv. ExecJ

Exe4j which is a free Java based .exe file converter for Java applications in the Windows operating environment was used to produce an executable for demonstration purposes.

CHAPTER 4 IMPLEMENTATION, TESTING AND RESULTS

4.1 Introduction

This chapter describes the implementation of the system with respect to programming, testing strategies, test data and the testing approaches employed. Further, we report on results generated through testing.

4.2 Coding

The system was implemented on a Java platform comprising of the JDK compiler, netbeans IDE developer, weka API and the exe4j executable file converter. The main components that were implemented for this system can be categorized as:

GUI components

These are form-based classes that give the look and feel of the system. They include:

1. **loginForm**

This is an interface that provides the entry point into the system for the two key users of the system namely the systems administrator and the loans officer

2. **adminPanel**

Through this interface the person responsible for performing administrative work including training and testing the system is provided with a substantial number of training and testing options

3. **processingForm**

This is the component through which the loans or credit officer uses to make predictions for fresh applications

4. **outputForm**

A class through which the vital information resulting from the classifications exercise can be viewed and printed.

5. **programmerOutput**

Through this interface, the results of training and testing can be viewed and printed.

Java Classes

- i. **Classifier:** Classifier interface.
- ii. **DecisionStump:** Class for building and using a decision stump. Usually used in conjunction with a boosting algorithm. Does regression (based on mean-squared error) or classification (based on entropy). Missing is treated as a separate value.
- iii. **Evaluation:** Class for evaluating machine learning models.
- iv. **Logistic:** Class for building and using a multinomial logistic regression model with a ridge estimator.
- v. **LogitBoost:** Class for performing additive logistic regression.
- vi. **NominalPrediction:** Encapsulates an evaluatable nominal prediction: the predicted probability distribution plus the actual class value
- vii. **Prediction:** Encapsulates a single evaluatable prediction: the predicted value plus the actual class value.
- viii. **ThresholdCurve:** Generates points illustrating prediction tradeoffs that can be obtained by varying the threshold value between classes.
- ix. **TwoClassStats:** Encapsulates performance functions for two-class problems.
- x. **VisualizeROC:** Visualizes a previously saved ROC curve.

4.3 Model Building & Testing Strategies

The model was built using the training dataset and tested using three strategies:

Cross-validation

It works as follows:

- i. Separate data in to 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 from the individual experiments.

Empirical 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.

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.

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.

4.4 Training / Testing Results

4.4.1 Training Results: Base Classifiers and their weights

Options: -F 0 -R 1 -I 15 -x 2

Iteration No.	Class	Classification Attribute	Classification Value
Iteration 1	Accept	duration <= 17.0	1.4563106796116505
		duration > 17.0	0.42735042735042733
		duration is missing	0.9090909090909091
	Reject	duration <= 17.0	-1.4563106796116505
		duration > 17.0	-.42735042735042733
		duration is missing	-0.9090909090909091
Iteration 2	Accept	checking_status = no checking	0.9500282567490651
		checking_status != no checking	-0.296298016347876
		checking_status is missing	0.12465542676453342
	Reject	checking_status = no checking	-0.9500282567490652
		checking_status != no checking	0.2962980163478756
		checking_status is missing	-.12465542676453387
Iteration 3	Accept	credit_amount <= 11280.5	0.13629029846336546
		credit_amount > 11280.5	-2.1400275297320808
		credit_amount is missing	0.06909464232014666
	Reject	credit_amount <= 11280.5	-.13629029846336568
		credit_amount > 11280.5	2.1400275297320657
		credit_amount is missing	-0.0690946423201471
Iteration 4	Accept	installment_commitment <= 3.5	0.33799918474032997
		installment_commitment > 3.5	-.37100161231012413
		installment_commitment is missing	-.00511137250068862
	Reject	installment_commitment <= 3.5	-.33799918474032986
		installment_commitment > 3.5	0.37100161231012424
		installment_commitment is missing	0.00511137250068867
Iteration 5	Accept	credit_amount <= 3914.0	0.2761626499371664
		credit_amount > 3914.0	-0.50569260444324
		credit_amount is missing	0.01896195370595613

	Reject	credit_amount <= 3914.0	-0.2761626499371664
		credit_amount > 3914.0	0.5056926044432385
		credit_amount is missing	-0.0189619537059561
Iteration 6	Accept	employment = 4<=X<7	0.7738228557656643
		employment != 4<=X<7	-0.1379745521908490
		employment is missing	0.01173882068711757
	Reject	employment = 4<=X<7	-0.7738228557656643
		employment != 4<=X<7	0.13797455219084892
		employment is missing	-0.011738820687117759
Iteration 7	Accept	employment = unemployed	-1.2299008140340528
		employment != unemployed	0.08884262006946772
		employment is missing	0.016295560681925017
	Reject	employment = unemployed	1.2299008140340528
		employment != unemployed	-0.0888426200694678
		employment is missing	-0.016295560681925284
Iteration 8	Accept	purpose = used car	0.9700418707120771
		purpose != used car	-0.11354440409096574
		purpose is missing	0.00967019943649037
	Reject	purpose = used car	-0.9700418707120773
		purpose != used car	0.11354440409096574
		purpose is missing	-0.009670199436490143
Iteration 9	Accept	duration <= 11.5	0.7755734466707114
		duration > 11.5	-0.11122330896336645
		duration is missing	0.01126023940736322
	Reject	duration <= 11.5	-0.7755734466707109
		duration > 11.5	0.11122330896336659
		duration is missing	-0.011260239407363048
Iteration 10	Accept	credit_amount <= 1144.5	-0.7358403282209007
		credit_amount > 1144.5	0.15256897720866966
		credit_amount is missing	0.026251575538684215
	Reject	credit_amount <= 1144.5	0.735840328220901
		credit_amount > 1144.5	-0.15256897720866966

		credit_amount is missing	-0.02625157553868416
Iteration 11	Accept	credit_amount <= 628.5	1.7305854652512525
		credit_amount > 628.5	-0.0749399341335183
		credit_amount is missing	0.004848401454331019
	Reject	credit_amount <= 628.5	-1.7305854652512527
		credit_amount > 628.5	0.0749399341335184
		credit_amount is missing	-0.004848401454330921
Iteration 12	Accept	credit_history = all paid	-1.4841410555617511
		credit_history != all paid	0.0870995093295415
		credit_history is missing	0.01943196797217743
	Reject	credit_history = all paid	1.484141055561751
		credit_history != all paid	-0.08709950932954144
		credit_history is missing	-0.01943196797217743
Iteration 13	Accept	purpose = education	-1.353510851609404
		purpose != education	0.08143765255723828
		purpose is missing	-1.9256805319803488E-4
	Reject	purpose = education	1.353510851609404
		purpose != education	-0.08143765255723825
		purpose is missing	1.9256805319801227E-4
Iteration 14	Accept	personal_status = female div/dep/mar	-0.48306871091333387
		personal_status != female div/dep/mar	0.2224819549318428
		personal_status is missing	3.221583707925294E-4
	Reject	personal_status = female div/dep/mar	0.48306871091333353
		personal_status != female div/dep/mar	-0.22248195493184275
		personal_status is missing	-3.2215837079251753E-4
Iteration 15	Accept	age <= 51.5	0.1464278819586059
		age > 51.5	-0.7849857127024696
		age is missing	0.004535522576763123

	Reject	age <= 51.5	-0.1464278819586059
		age > 51.5	0.784985712702469
		age is missing	-0.004535522576763188

Table 4.1: Base Classifiers and their weights

4.4.2 Testing Results: 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

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

Table 4.2: TestFile Predictions

CHAPTER 5 OUTPUT ANALYSIS AND INTERPRETATION

5.1 Performance Measures

i. Confusion matrix

Confusion matrices are very useful for evaluating classifiers, as they provide an efficient snapshot of its performance - displaying the distribution of correct and incorrect instances. A confusion matrix takes the following form

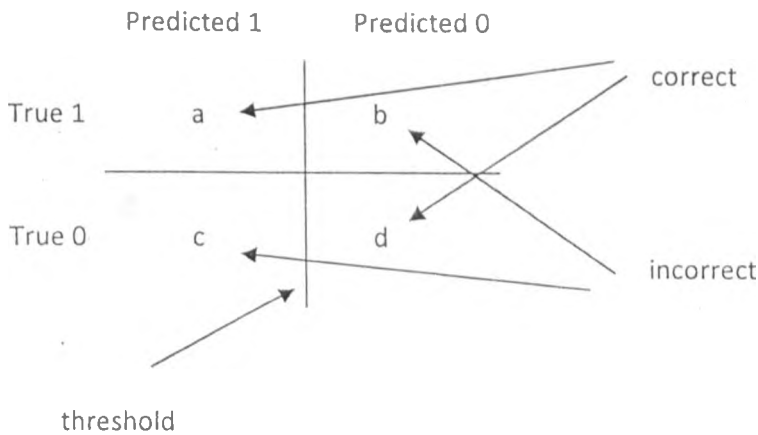


Figure 5.1: Confusion Matrix

ii. Accuracy

It is one of the most important measures that can be used to evaluate the performance of a classifier. It gives a measure for the overall predictive correctness of the classifier. Accuracy is calculated as the number of correctly classified instances divided by the number of instances. From the confusion matrix:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d}$$

Though easy, accuracy assumes equal costs for both types of errors and hence not very objective

iii. Precision and recall

Precision is a measure of the number of correctly classified instances. With respect to classifiers, this measure is computed as:

precision(X) =number of correctly classified instances of class X /number of instances classified as belonging to class X

Recall is a measure of the number of positives returned by the classifier as is computed as:

recall(X) =number of correctly classified instances of class X / number of instances in class X

iv. Lift

Is a measure that disregards the accuracy of the entire dataset, and considers the accuracy of a smaller percentage such as 5%, 10% of the dataset and disregards the the remaining 95%, 90%. Lift is based on classifications as tabulated in a confusion matrix as: It is calculated as: $\text{lift}(\text{threshold}) = \% \text{positives} > \text{threshold} / \% \text{dataset} > \text{threshold}$. From the confusion matrix:

$$\text{Lift} = \frac{a}{(a+b)} / \frac{(a+c)}{(a+b+c+d)}$$

v. ROC Curve

It 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. $P(\text{true} | \text{true})$ calculated as

$$a/a+b \text{ from a standard confusion matrix}$$

1- **Specificity:** The probability of true classifications given false instances i.e. $P(\text{true} | \text{false})$ calculated

$$\text{as } 1 - d/c+d$$

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

5.2 Performance analysis results

i. Error on Stratified cross-validation

Statistic	Value	Rate
Correctly Classified Instances	161	73.1818 %
Incorrectly Classified Instances	59	26.8182 %
Root mean squared error	0.4467	-
Coverage of cases (0.95 level)	-	95.4545 %
Total Number of Instances	220	-

Table 5.1: Error on Stratified Cross Validation

Detailed Accuracy By Class

Class	Precision	Recall	ROC Area
Accept	0.792	0.856	0.668
Reject	0.511	0.4	0.668
Wtd Avg.	0.715	0.732	0.668

Table 5.2: Cross Validation Class Accuracy

		Classified As	
		A=Accept	B=Reject
Class	A=Accept	137	23
	B=Reject	36	24

Table 5.3: Cross Validation Confusion Matrix

ROC Graph

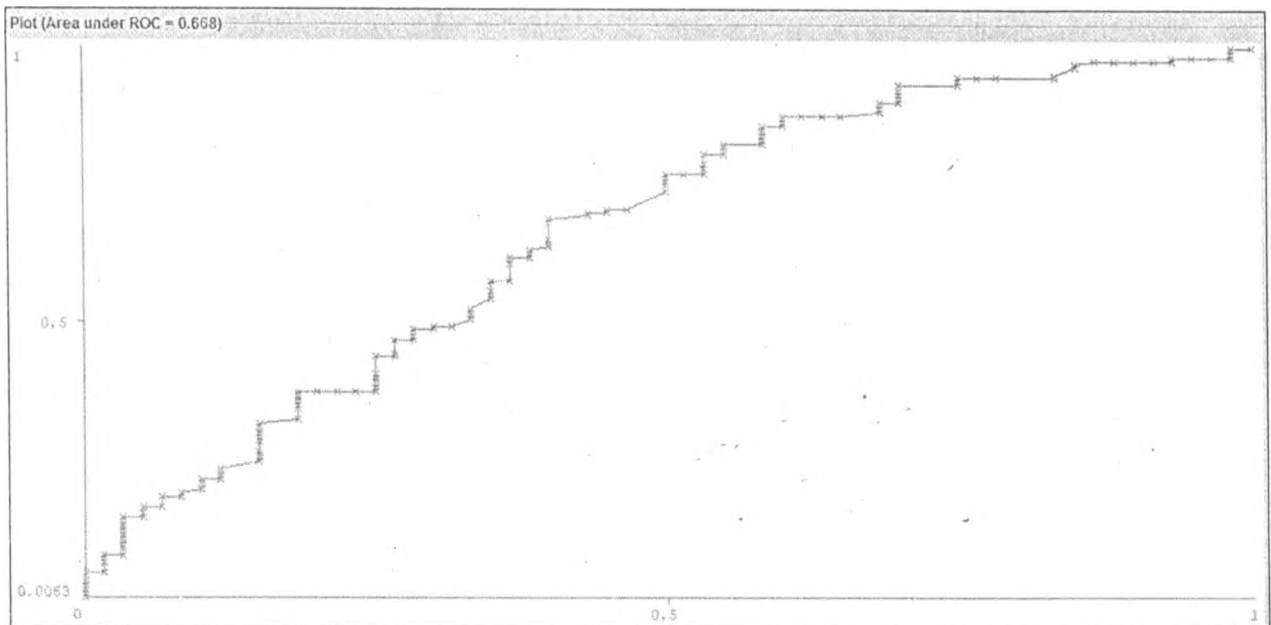


Figure 5.2 X-validation ROC graph

- ii. Error on test data

Statistic	Value	Rate
Correctly Classified Instances	19	95 %
Incorrectly Classified Instances	1	5 %
Root mean squared error	0.3353	-
Coverage of cases (0.95 level)	-	100 %
Total Number of Instances	20	-

Table 5.4: Error on Test Data

		Classified As	
		A=Accept	B=Reject
Class	A=Accept	12	0
	B=Reject	1	7

Table 5.5: Test Confusion Matrix

ROC Graph

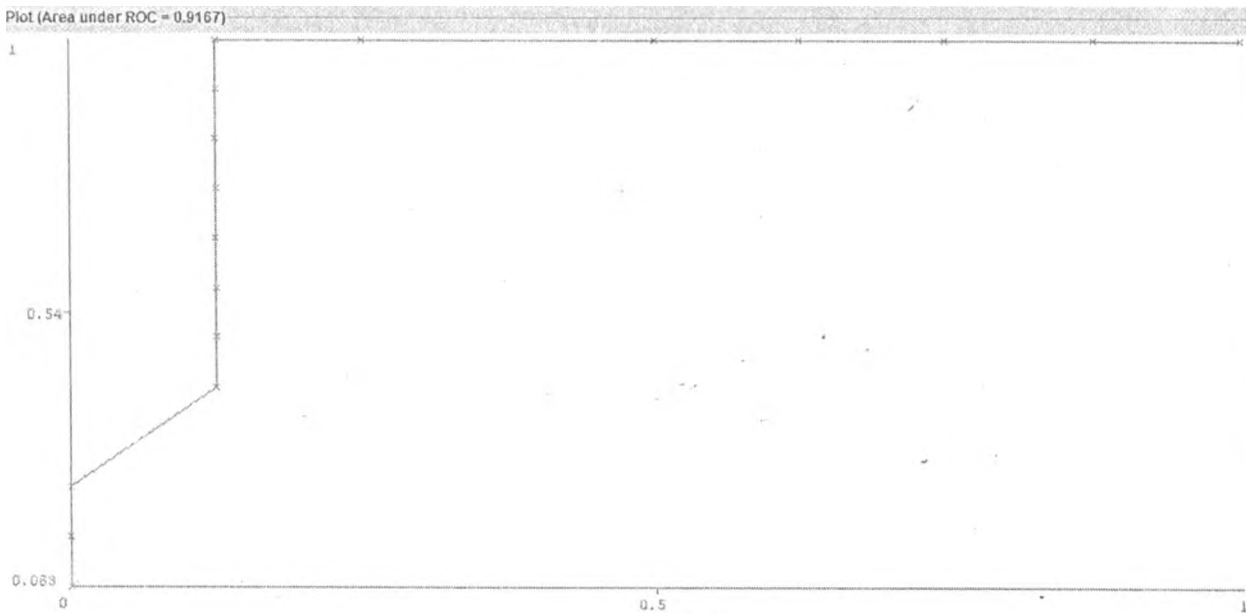


Figure 5.3: Test ROC graph

iii. Training / Test Split

Statistic	Value	Rate
Correctly Classified Instances	117	88.6364 %
Incorrectly Classified Instances	15	11.3636 %
Root mean squared error	0.2936	
Coverage of cases (0.95 level)		100 %
Total Number of Instances	220	

Table 5.6: Training Split Error

Class	Precision	Recall	ROC Area
Accept	0.882	0.98	0.951
Reject	0.909	0.606	0.951
Wtd Avg.	0.889	0.886	0.951

Table 5.7: Training Split Class Accuracy

		Classified As	
		A=Accept	B=Reject
Class	A=Accept	97	2
	B=Reject	13	20

Table 5.8: Training Split Confusion Matrix

Correctly Classified Instances	60	68.1818 %
Incorrectly Classified Instances	28	31.8182 %
Root mean squared error	0.4523	-
Coverage of cases (0.95 level)	-	97.7273 %
Total Number of Instances	-	88

Table 5.9: Test split error

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 5.10: Test split class accuracy

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

Table 5.11: Test Split Confusion Matrix

ROC Graph

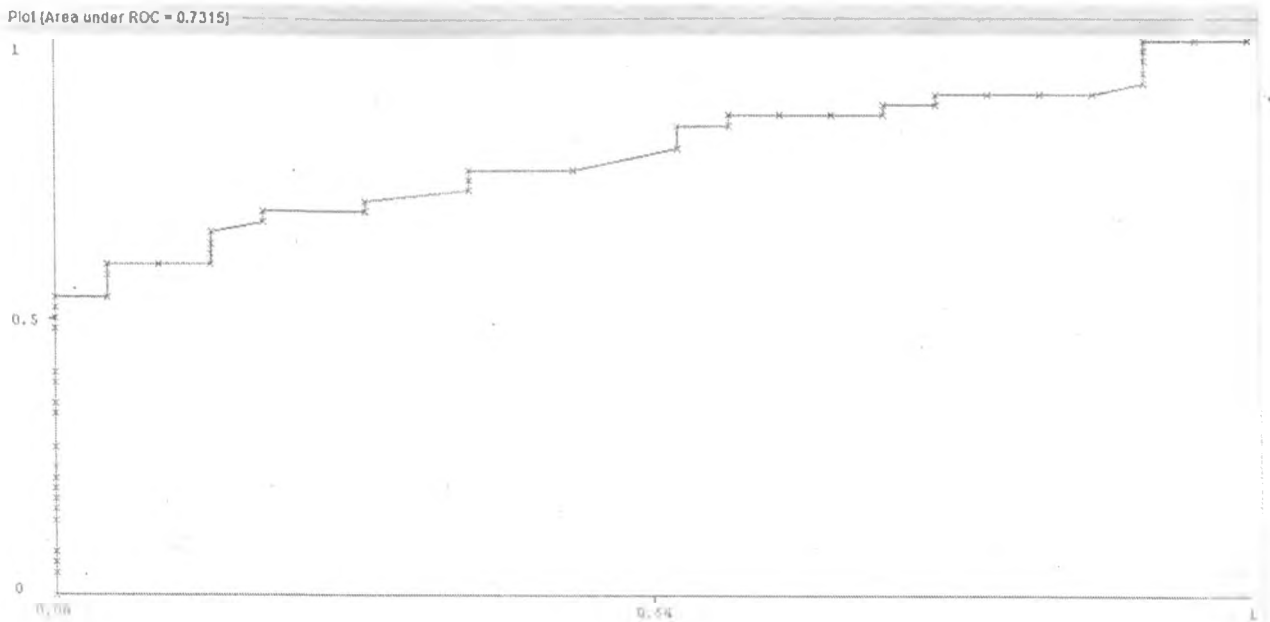


Figure 5.4: Test Split ROC graph

5.3 Interpretation of Results

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

Training Set

i. Training Accuracy

The accuracy returned by the training set is 190 correctly classified instances out of 220 instances. This gives an accuracy of

$$\frac{190}{220} \\ =86.36\%$$

ii. Precision

Class =Accept: The number of correctly classified instances is 154 and that of instances classified as belong to the class is 178. This gives a precision value of

$$\frac{154}{178} \\ =0.87$$

Class =Reject: The number of correctly classified instances is 36 and that of instances classified as belong to the class is 42. This gives a precision value of
 $36/42$
 $=0.86$

iii. *Recall*

Class =Accept: The number of correctly classified instances is 154 and the number of instances belonging to the class is 160. This gives a recall value of
 $154/160$
 $=0.96$

Class =Reject: The number of correctly classified instances is 36 and the number of instances belonging to the class is 60. This gives a recall value of
 $36/60$
 $=0.60$

iv. *Nature of ROC*

The ROC graph is regular with an area of 0.89. Converted to 1 decimal place, this values indicates an excellent classification

Training and Testing Set

i. *Testing Accuracy*

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

ii. *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

iii. 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

iv. Nature of ROC

The ROC graph is regular with an area of 0.96. Converted to 1 decimal place, this values indicates an perfect classification

Training Split

i. Training Accuracy

The accuracy returned by the training set is 117 correctly classified instances out of 132 instances. This gives an accuracy of

117/132

=88.64%

ii. *Precision*

Class =Accept: The number of correctly classified instances is 97 and that of instances classified as belong to the class is 110. This gives a precision value of

$$\frac{97}{110} \\ =0.88$$

Class =Reject: The number of correctly classified instances is 20 and that of instances classified as belong to the class is 22. This gives a precision value of

$$\frac{20}{22} \\ =0.91$$

iii. *Recall*

Class =Good: The number of correctly classified instances is 97 and the number of instances belonging to the class is 99. This gives a recall value of

$$\frac{97}{99} \\ =0.98$$

Class =Reject: The number of correctly classified instances is 20 and the number of instances belonging to the class is 33. This gives a recall value of

$$\frac{20}{33} \\ =0.61$$

iv. *Nature of ROC*

The ROC graph is regular with an area of 0.95. Converted to 1 decimal place, this values indicates an perfect classification

Test Split

i. *Testing Accuracy*

The accuracy returned by the training set is 60 cofrectly classified instances out of 88 instances. This gives an accuracy of

60/88

=68.18%

ii. *Precision*

Class =Accept: The number of correctly classified instances is 56 and that of instances classified as belong to the class is 79. This gives a precision value of

56/79

=0.88

Class =Reject: The number of correctly classified instances is 20 and that of instances classified as belong to the class is 22. This gives a precision value of

20/22

=0.71

iii. *Recall*

Class =Good: The number of correctly classified instances is 56 and the number of instances belonging to the class is 61. This gives a recall value of

56/61

=0.92

Class =Reject: The number of correctly classified instances is 4 and the number of instances belonging to the class is 27. This gives a recall value of

4/27

=0.15

iv. *Nature of ROC*

The ROC graph is regular with an area of 0.71. Converted to 1 decimal place, this values indicates an mediocre classification.

CHAPTER 6 DISCUSSION

6.1 FINDINGS

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

i. Model Accuracy

Three options were investigated for training the algorithm namely:

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

The use of separate training and testing data sets returned an accuracy of 95% making it a relatively better strategy

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.

ii. 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%

6.2 SUGGESTIONS

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

- i. **Use of Clean data:** This is the use of refined, pre-processed training data or data that is scientifically sound or data that is devoid of instances of assumptions
- ii. **Parameter Tuning:** The training and testing procedures can be done severally with different input parameters and file sizes to settle of the most effective set for different learning processes
- iii. **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.

Finally, as stated earlier on 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 good aspect of the classifications 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.

6.3 CONCLUSION

As a conclusion, the reported work indeed confirms that:

Machine learning procedures can be applied in financial modeling applications to augment manual underwriting techniques

These procedures can greatly improve the efficiency of such techniques because of their ability to handle large items of data generating very useful statistics

6.4 FURTHER WORK

This work can be improved along the parameters of data set pre-processing, the use of a cost matrix as well as parameter tuning to settle on the most effective set for various data mining requirements.

APPENDICES

Appendix A: References and Bibliography

- [1] Risk Management Examination Manual for Credit Card Activities. (2007). [Online]. Underwriting and loan approval process. Available from: www.fdic.gov/regulations/examinations/credit_card. [Accessed: 15/11/2010]
- [2] Credit Approval benefits (n.d.). [Online]. Available from: <http://www.referenceforbusiness.com/encyclopedia/Cos-Des/Credit-Approval.html>. [Accessed: 15/11/2010]
- [3] Chan, P. K. (n.d.). [Online]. Distributed Data Mining in Credit Card Fraud Detection. Available from: www.pdfhost.com/credit-card/176-7312-pdf.html. [Accessed: 13/12/2010]
- [4] Dongsong, Z., Lina, Z. (2004). [Online]. Discovering Golden Nuggets: Data Mining in Financial Applications. IEEE Transactions on Systems, Man, and Cybernetics. Available from <http://suraj.lums.edu.pk/~cs631s05/Papers/financial.pdf>. [Accessed: 17/01/2011]
- [5] Witten, I., Frank, E. (2008). [Online]. Data Mining Practical Machine Learning Tools and Techniques. Available from: [http://html-pdf-converter.com/pdf/data-mining%3A-practical-machine-learning-tools-and-techniques-\(second-edition\).html](http://html-pdf-converter.com/pdf/data-mining%3A-practical-machine-learning-tools-and-techniques-(second-edition).html). [Accessed: 11/11/2010]
- [6] Martin, S. (2008). [Online]. Ensemble Learning. Available from: <http://machine-learning.martinsewell.com/ensembles/ensemble-learning.pdf> [Accessed: 13/12/2010]
- [7] Holmes, G., Pfahringer, B., Kirkby, R., Eibe, F., Hall, M. (2003). [Online]. Multiclass Alternating Decision Trees. Available from: www.cs.waikato.ac.nz/~mhall/pubs.html. [Accessed: 17/01/2011]
- [8] Shorouq, F. E., Saad, Ghaleb, Y., Ghaleb, A. E. (n.d.). Neuro-Based Artificial Intelligence Model for Loan Decisions
- [9] Bauer, E., Kohavi, R. (2006). An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants
- [10] Witten, I. H., Eibe, F., Trigg, L., Hall, M., Holmes, G., Cunningham, S. J. (n.d.). Weka: Practical Machine Learning Tools and Techniques with Java Implementations

- [11] Friedman, J., Hastie, T., Tibshirani, R. (2000). [Online]. Additive logistic regression: a statistical view of boosting. Available from: <http://www.stanford.edu/~hastie/Papers/AdditiveLogisticRegression/alr.pdf>. [Accessed: 12/02/2011]
- [12] 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
- [13] Yanwen D. (n.d.) A case based reasoning system for customer credit scoring: Comparative study of similarity measures. Cluster of Science and Technology, Fukushima University. Kanayagawa, Fukushima City, Japan
- [14] Veronica S. M. (n.d.). Towards the use of C4.5 algorithm for classifying banking dataset .
- [15] Liang-Hsuan C., Tai-Wei C. (1998). A fuzzy credit-rating approach for commercial loans: a Taiwan case. Department of Industrial Management Science, National Cheng Kung University
- [16] Agresti A. (2007). "Building and applying logistic regression models". *An Introduction to Categorical Data Analysis*. Hoboken, New Jersey: Wiley.

Appendix B: User Manual

Starting the System

To launch the system:

- Go to start
- All Programs
- Metalib folder
- Launch the executable

Login

There are two types of users for this system

Administrator

On the login screen:

- Select Administrator as the User Type
- Type the default user name and password
- If correct, the administrator's panel is opened
- You can change the user name and password by opening the navigation menu

Loans Officer

On the login screen:

- Select Loans Officer as the User Type
- Type the default user name and password
- If correct, the classifications panel is opened
- You can change the user name and password by opening the navigation menu

Model Building

This feature is available to administrators only. To create a model, a training file is required including a testing file is this option is to be used for testing. There are three training / testing strategies:

Using Cross-Validation

- Enter / browse for the training file
- Enter / browse for the model output file
- Specify the ROC output file
- Use the default model input parameters or specify others for fine tuning
- Leave the default number of folds for cross validation or specify a new value
- Specify no values for both testing and split percentage options
- Select the training parameters as appropriate
- Click on build the model

Using a test file

- Enter / browse for the training file
- Enter / browse for the model output file
- Specify the ROC output file
- Use the default model input parameters or specify others for fine tuning
- Delete the default value specified for cross validation
- Enter / browse for the file containing the test instances
- Specify no values for split percentage option
- Select the training parameters as appropriate
- Click on build the model

Using a split percentage

- Enter / browse for the training file
- Enter / browse for the model output file
- Specify the ROC output file
- Use the default model input parameters or specify others for fine tuning
- Delete the default value specified for cross validation
- Specify no values for the training file
- Enter a split percentage for training and test instances e.g. 65 (means 65:35)
- Select the training parameters as appropriate
- Click on build the model

Classifying Instances

This option is available to the loans officer only. There are two important files required for this exercise:

- The file containing unclassified instances
- The model file to be used
- Specify the predictions output parameters as appropriate
- Click on the classify instances button

Viewing the Training output and Statistics

This option is available to the administrator only.

- From the administrator's panel click on the output window button
- This opens the administrator's output screen

On this screen, the following options are available depending on the testing strategy used:

Cross Validation

Under cross validation, three output files can be accessed

- i. Training results under stratified cross validation. This can be viewed (displayed on the space to the left) or opened as a printable file
- ii. Summary statistics for the training process. This can also be viewed (displayed on the space to the left) or opened as a printable file
- iii. The ROC curve for the training / testing process

Test File

Under Test File, three output files can be accessed

- i. Training results under testing file. This can be viewed (displayed on the space to the left) or opened as a printable file
- ii. Summary statistics for the training process. This can also be viewed (displayed on the space to the left) or opened as a printable file
- iii. The ROC curve for the training / testing process

Split Percentage

Under split percentage, three output files can be accessed

- i. Training results under a split percentage. This can be viewed (displayed on the space to the left) or opened as a printable file
- ii. Summary statistics for the training process. This can also be viewed (displayed on the space to the left) or opened as a printable file
- iii. The ROC curve for the training / testing process

Viewing the predictions output

This option is available to the loans officer.

- From the processing window, click on the output window button
- This opens the processing window

On this window, three options are available:

- i. View the model file by displaying it or opening it in a printable format
- ii. View the classified instances file by displaying it or opening it in a printable format
- iii. View the base classes used in the classifications file by displaying it or opening it in a printable format

Appendix C: Samples

Login Screen

LOANS APPLICATIONS EVALUATOR

LOGIN FORM

USER TYPE	Administrator <input type="button" value="v"/>
USER NAME	<input type="text"/>
PASSWORD	<input type="password"/>

Administrator's Panel

Navigation [Help](#)

ADMINISTRATOR'S PANEL

DATA FILES

TRAINING FILE	<input type="text" value="act_trainfile.dat"/>	<input type="button" value="BROWSE"/>
MODEL OUTPUT FILE	<input type="text" value="act_model.out"/>	<input type="button" value="BROWSE"/>
ROC CURVE FILE	<input type="text" value="act_roccurve.dat"/>	<input type="button" value="BROWSE"/>

TESTING CRITERIA

Cross Validation

Number of Folds:

Test File

Full Path to Test File:

Split Training File

Split Percentage (%):

Reverse Order?

TRAINING PARAMETERS

Generate a File of Tested Instances: No

Generate Detailed Class Statistics: No

Output Testing Distribution: No

Generate Information Theoretic Statistics: No

LOGITBOOST SPECIFIC PARAMETERS

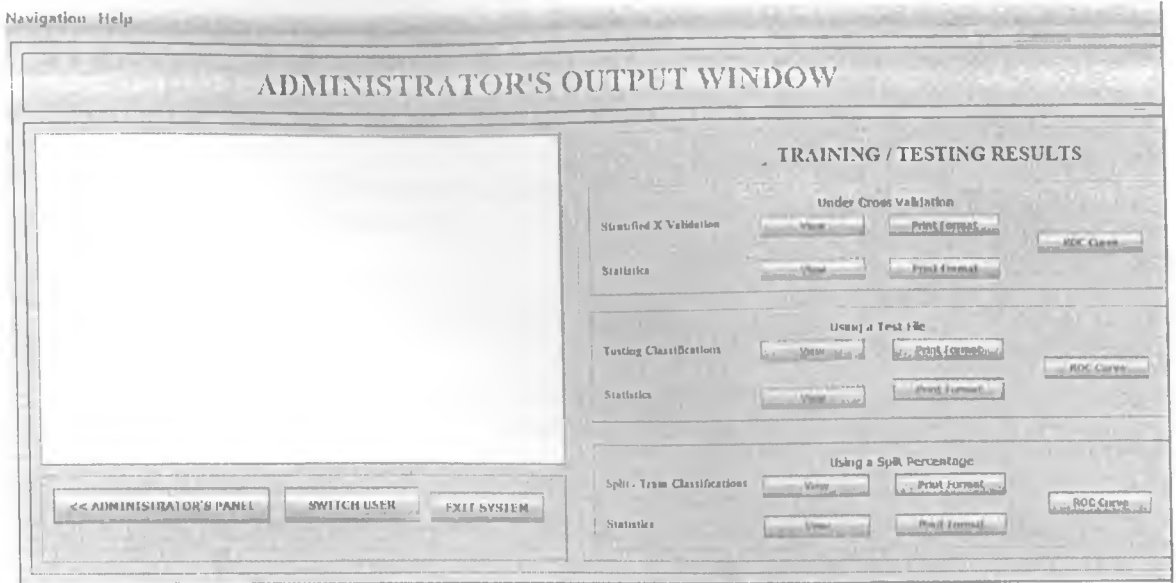
Boosting Criteria: Random

No. of Folds for Internal X Validation:

No. of Turns for Internal X Validation:

Number of Iterations:

Output Window



Sample Code

A method to build a base classifier

```
public void buildClassifier(Instances instances) throws Exception {  
  
    double bestVal = Double.MAX_VALUE, currVal;  
    double bestPoint = -Double.MAX_VALUE;  
    int bestAtt = -1, numClasses;  
  
    // can classifier handle the data?  
    getCapabilities().testWithFail(instances);  
  
    // remove instances with missing class  
    instances = new Instances(instances);  
    instances.deleteWithMissingClass();  
  
    // only class? -> build ZeroR model  
    if (instances.numAttributes() == 1) {  
        System.err.println(  
            "Cannot build model (only class attribute present in data)");  
  
        return;  
    }  
  
    double[][] bestDist = new double[3][instances.numClasses()];  
  
    m_Instances = new Instances(instances);  
  
    if (m_Instances.classAttribute().isNominal()) {
```

```

    numClasses = m_Instances.numClasses();
} else {
    numClasses = 1;
}

// For each attribute
boolean first = true;
for (int i = 0; i < m_Instances.numAttributes(); i++) {
    if (i != m_Instances.classIndex()) {

        // Reserve space for distribution.
        m_Distribution = new double[3][numClasses];

        // Compute value of criterion for best split on attribute
        if (m_Instances.attribute(i).isNominal()) {
            currVal = findSplitNominal(i);
        } else {
            currVal = findSplitNumeric(i);
        }
        if ((first) || (currVal < bestVal)) {
            bestVal = currVal;
            bestAtt = i;
            bestPoint = m_SplitPoint;
            for (int j = 0; j < 3; j++) {
                System.arraycopy(m_Distribution[j], 0, bestDist[j], 0,
                    numClasses);
            }
        }

        // First attribute has been investigated
        first = false;
    }
}

// Set attribute, split point and distribution.
m_AtIndex = bestAtt;
m_SplitPoint = bestPoint;
m_Distribution = bestDist;
if (m_Instances.classAttribute().isNominal()) {
    for (int i = 0; i < m_Distribution.length; i++) {
        double sumCounts = Utils.sum(m_Distribution[i]);
        if (sumCounts == 0) { // This means there were only missing attribute values
            System.arraycopy(m_Distribution[2], 0, m_Distribution[i], 0,
                m_Distribution[2].length);
            Utils.normalize(m_Distribution[i]);
        } else {
            Utils.normalize(m_Distribution[i], sumCounts);
        }
    }
}

// Save memory
m_Instances = new Instances(m_Instances, 0);}

```

Appendix D: Data Collection Questionnaire

SECTION 1: INTRODUCTION

SECTION 2: PRESENCE OF A CBIS (Tick as Appropriate)

a) Does your institution use computer based information system(s) for processing loan applications? Yes No

b) If Yes in (a), how was the system acquired?

In-house Devpt Outsourcing Off-the-shelf

c) Do you know the software tools, techniques or methods used to develop the system?

Yes Somehow No

d) If Yes in (c) above, what class of software tools are used?

Statistical Database Data Mining Other

SECTION 3: USAGE OF SYSTEM (Tick as Appropriate)

a) How easy is it to use the tool?

Easy Fairly Easy Difficult Complicated

b) How many loan / credit applications can the system process per hour?

Below 100 100 – 500 Over 500

c) What is the nature of system output with respect to credit / loan decisions?

Credit score Accept /Reject Other

d) Are there cases of incorrectly classified loan / credit applications? E.g. a good application classified as bad and vice versa? Yes No

e) If Yes to d) above, what is the extent of this?

Minimal Moderate High

f) How can you gauge the system in general?

Fairly Good Good Moderate