Credit Approval Analysis using R

Deepesh Khaneja
Carleton University

CREDIT APPROVAL ANALYSIS

Abiola Smith[1], Brendan Maher[2], and Deepesh Khaneja[3]
[1]*abiolasmith@cmail.carleton.ca,* [2]*brendanmaher@cmail.carleton.ca,* [3]*deepeshkhaneja@cmail.carleton.ca*

***Abstract* – Algorithms that are used to decide the outcome of credit applications vary from one provider to another and across sectors and geographies. There are however, high degrees of similarity in the attributes used to generate those algorithms. The differences may be in the weights applied to individual attributes. This paper analyzes credit application data in the Credit Approval dataset taken from the archives of the machine learning repository of University of California, Irvine (UCI) to (a) establish which attributes are given the highest weight and (b) determine which data analysis techniques produce the best model for predicting the outcome of an application using the same algorithm. To achieve this, we used several visualization, data reduction, clustering and classification techniques and in building the predictive model, various classification techniques were compared to find the one best suited for the dataset. These results can serve as a source of information for consumers. For analysts in the financial sector it provides a model that can be incorporated to automate the credit application approval proces**s.

***Keywords* – Credit approval, credit scoring model, credit application prediction, predictive model, classification**

I. INTRODUCTION

The decision of approving a credit card or loan is majorly dependent on the personal and financial background of the applicant. Precisely, age, gender, income, employment status, credit history and other attributes contributes to the approval decision. Credit Analysis involves the statistical – quantitative and qualitative measure to investigate the probability of a third party to pay back the loan to the bank on time and predict its default characteristic. Analysis focus on recognizing, assessing and reducing the financial/other risks involved which may otherwise results in the losses incurred by the company while lending. The risk can be business loss by not approving the good candidate or can be financial loss by approving the candidate who is at bad risk. It is very important to manage credit risk and handle challenges efficiently for credit decision as it can have adverse effects on credit management. Therefore, evaluation of credit approval is significant before jumping to any granting decision.

Considering the importance of credit evaluation, multiple studies and research have been propounded exploiting data mining tools so to enhance the existing models and prediction accuracy. Data Mining plays a crucial role in knowledge discovery, providing vast number of tools, techniques and methodologies for pattern recognition and data analysis. Importantly, correct tools and methods are need to be identified to fit the data in an appropriate manner. It's a powerful technology leverages big organizations to determine the important data in their reports, further help them gain and maintain current and potential customers from their behaviours. Various applications include fraud detection, market basket analysis, Trend/Sentiment Analysis, Market Segmentation, credit scoring. One such application which is explored in this report is Credit Approval involves classification of the applications based on classification methods of data mining. Socio-economic and demographic statistics are considered by the banks before making decision rule to minimize risks. It groups the people who can be detrimental to the credit approval and further the results estimate the important variables in the dataset for comprehending credit approval policy. Moreover, this kind of analysis can prevent credit fraud from the fake applications having fake personal information by identifying the important attributes. Artificial Neural Networks, Support vector Machines, Logistic Regression, CART are some of the commonly used techniques for classification in credit risk evaluation with promising results.

The primary objective of this analysis is to implement the data mining techniques on credit approval dataset and prepare models for prediction of approval decision using supervised learning. Therefore, the risks can be identified while lending, appropriate conclusions can be elicited about probability of repayment and recommendations can be put forward. Also, from the analysis, important predictors are determined influencing the credit approval decision and the ones with no impact are discarded. The techniques used for analysis are data visualization to get better insight of data, data and dimension reduction to locate essential attributes, supervised and unsupervised learning for preparing models. The classification algorithms used in supervised are Logistic Regression, Classification and Regression Binary Trees(CART), Neural Networks, LDA/QDA and KNN to examine the prediction accuracy. These models are evaluated and compared using confusion matrix, Area Under the Curve performance metrics. Some of the research questions are addressed while summarizing the findings the last section of this report. The whole analysis is done in an open source statistical environment of language R.

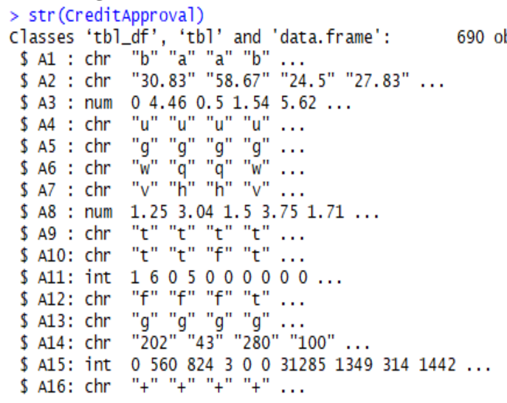
The report is structured as follows Section II describes the detail of chosen dataset and the required transformations. Followed by Section III discussing the methods and techniques used in the analysis explained. Further, Section IV summarizes the important findings of our analysis done for credit approval. Lastly, Section V concludes the report with the decisions influencing the credit approval to any applicant. Working code for the whole analysis can be found in Appendix.

•

II. EXPLORATORY ANALYSIS OF DATASET AND REQUIRED TRANSFORMATIONS

A. *Data Description*

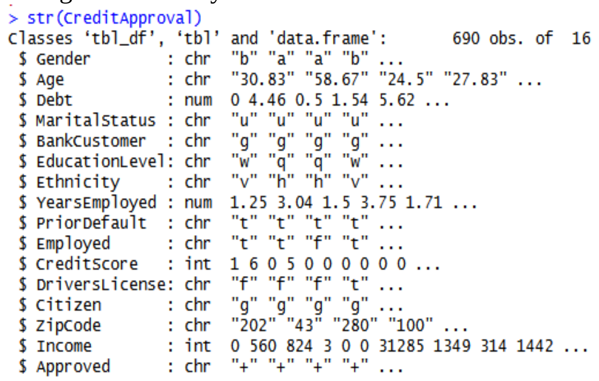
The dataset analysed in this report is the Credit Approval dataset taken from the archives of the machine learning repository of University of California, Irvine (UCI). It contains data from credit card applications. In its initial, unaltered form, the dataset made available by UCI contains 690 cases, representing 690 individuals, and 16 variables named A1 – A16. The first 15 variables represent various attributes of the individuals submitting the application and the 16th variable contains the outcome of the application, either positive (represented by "+") meaning granted or negative (represented by "-") meaning rejected. The variable with outcome generally referred as Class attribute of the dataset which in our case is 16th variable having two classes and other attributes providing useful information are called predictors. This multivariate dataset is a mix of attribute having continuous (integer, real), nominal and categorical data values along with some missing values specifically in A1, A2, A4, A5, A6, A7 and A14. These missing values constitute 5% of the entire dataset and are treated to fill them up with various methods described in further section. A first look at the structure of the dataset showed the following:



```
> str(CreditApproval)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':	690 obs. of  16 variables:
 $ A1 : chr  "b" "a" "a" "b" ...
 $ A2 : chr  "30.83" "58.67" "24.5" "27.83" ...
 $ A3 : num  0 4.46 0.5 1.54 5.62 ...
 $ A4 : chr  "u" "u" "u" "u" ...
 $ A5 : chr  "g" "g" "g" "g" ...
 $ A6 : chr  "w" "q" "q" "w" ...
 $ A7 : chr  "v" "h" "h" "v" ...
 $ A8 : num  1.25 3.04 1.5 3.75 1.71 ...
 $ A9 : chr  "t" "t" "t" "t" ...
 $ A10: chr  "t" "t" "f" "t" ...
 $ A11: int  1 6 0 5 0 0 0 0 0 0 ...
 $ A12: chr  "f" "f" "f" "t" ...
 $ A13: chr  "g" "g" "g" "g" ...
 $ A14: chr  "202" "43" "280" "100" ...
 $ A15: int  0 560 824 3 0 0 31285 1349 314 1442 ...
 $ A16: chr  "+" "+" "+" "+" ...
```

B. *Variable Names*

From the source of the dataset, it was observed that the names and values of the attributes have been changed to some generic and meaningless symbols to ensure the confidentiality and privacy of the applicants. So as to avoid confusion and for the sake of simplicity, the labels of the variables in our analysis have been assumed to be some working names [6] according to the values in the attributes like A1 changed to Gender, A16 changed to Approved. This assumption is common to find in any credit approval dataset and somewhat fit our data. Below is the short summary of the changed variable names which are used throughout the analysis.



```
> str(CreditApproval)
Classes ‘tbl_df’, ‘tbl’ and 'data.frame':	690 obs. of  16 variables:
 $ Gender        : chr  "b" "a" "a" "b" ...
 $ Age           : chr  "30.83" "58.67" "24.5" "27.83" ...
 $ Debt          : num  0 4.46 0.5 1.54 5.62 ...
 $ MaritalStatus : chr  "u" "u" "u" "u" ...
 $ BankCustomer  : chr  "g" "g" "g" "g" ...
 $ EducationLevel: chr  "w" "q" "q" "w" ...
 $ Ethnicity     : chr  "v" "h" "h" "v" ...
 $ YearsEmployed : num  1.25 3.04 1.5 3.75 1.71 ...
 $ PriorDefault  : chr  "t" "t" "t" "t" ...
 $ Employed      : chr  "t" "t" "f" "t" ...
 $ CreditScore   : int  1 6 0 5 0 0 0 0 0 0 ...
 $ DriversLicense: chr  "f" "f" "f" "t" ...
 $ Citizen       : chr  "g" "g" "g" "g" ...
 $ ZipCode       : chr  "202" "43" "280" "100" ...
 $ Income        : int  0 560 824 3 0 0 31285 1349 314 1442 ...
 $ Approved      : chr  "+" "+" "+" "+" ...
```

The description of the new variable names corresponding to old variables are shown below.

Old Variable Name	New Variable Name	Variable Description
A1	Gender	Whether the applicant as male or not
A2	Age	The age of the applicant
A3	Debt	The amount of debt the applicant has
A4	MaritalStatus	Whether the applicant is married or not
A5	BankCustomer	Which bank the applicant is a customer of
A6	EducationLevel	Education level of the applicant
A7	Ethnicity	Ethnicity of the applicant
A8	YearsEmployed	Number of years the applicant has been employed
A9	PriorDefault	Whether the applicant has previously defaulted on a credit account
A10	Employed	Whether the applicant is employed or not
A11	CreditScore	The applicant's credit score
A12	DriversLicense	Whether the applicant posses a driver's licence or not
A13	Citizen	Whether the applicant is a citizen at the time of the application
A14	ZipCode	Zip code in which the applicant resides
A15	Income	Applicant's annual income
A16	Approved	Whether the application was approved or not

***Table 1**:* Variable Names Transformation

C. *Data Type*

To have a more accurate representation of the variables, the following transformations were made:

Variable	Original Data Type	Transformed Data Type
A1 / Male	Character	Binary
A2 / Age	Character	Continuous / numeric
A4 / Married	Character	Factor
A9 / PriorDefault	Character	Binary
A10 / Employed	Character	Binary
A12 / DriversLicense	Character	Factor
A13 / Citizen	Character	Factor
A16 / Approved	Character	Binary

Table 2: Variables and transformed datatypes

D. *Data Transformations*

As mentioned, the data in this analysis contains categorical values that are transformed to binary values or factors 1s and 0s. Approved variable have values '+' and '-' in original dataset and with our assumption '+' means granted changed to 1 and '-' changed to 0. Similarly, with attributes Gender having values 'a' changed to 1 representing male and 'b' changed to 0, PriorDefault and Employed both have categorical values 't' and 'f' which are transformed to '1' and '0' respectively. They are transmuted from character to binary datatype. Here, '1' as binary value considered true/yes/pass and '0' represents false/no/fail. On investigating the dataset, the missing values are labelled as '?' which are initially replaced by NAs so that R can construe as non-characters. Further, Age is transformed from character to numeric/continuous datatype to ease with further analysis in data visualization and classification. In addition, transformation from characters to factors is done on attributes Citizen, DriversLicense, ZipCode and MaritalStatus.

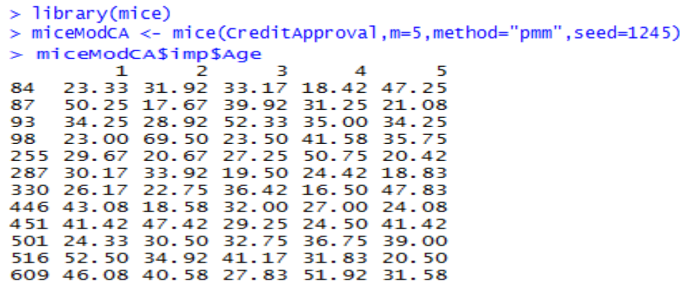
E. *Normalization*

The attributes with continuous values are each on a different scale with high variances among them. This hinders the analysis process as the results will deviate and no comparison for finding correlation can be done among variables. For better graphical representation the continuous, YearsEmployed, CreditScore, Age, Debt and Income were normalized to common scale.

F. *Missing data Treatment*

The Credit Approval dataset, in the form extracted from the UCI repository, contains missing values over 7 of the 16 variables. These missing values are found in 37 of the 690 cases representing almost 5% of the data. The missing values are found to exist in attributes Age, Gender, Marital Status, BankCustomer, EducationLevel, Ethnicity and ZipCode which was filled by NAs. Out of all, age is the continuous variable. There are various methods to address these missing values which can range from deleting the observations to fixing them with mean values. We can either delete the observations, delete the attribute if its of no importance, zero them out or plug the mean/median/mode value from all values. The simplest method would be to use the mean values to substitute NAs in Age but we imputed the values using prediction method. This method is much more accurate and efficient in meeting the equilibrium with other values. It includes different approaches like knn imputation, mice and rpart

We used the mice (Multivariate Imputation via Chained Equations) package in R to approximate missing values of Age attribute as it gave more satisfactory and accurate results than its counterparts. It creates multiple imputations of the missing values of an attribute and regress it with other attributes to predict the missing values. The methods include Logistic Regression for categorical values, PMM (predictive mean matching) for numeric data and has been used to fill. It consists of two steps using mice function where number of imputed datasets, iterations and method can be selected. In second step, complete() will merge the predicted values with the dataset choice being any imputed dataset. Below are the five different datasets of imputed values for missing observations.



```
> library(mice)
> miceModCA <- mice(CreditApproval,m=5,method="pmm",seed=1245)
> miceModCA$imp$Age
        1     2     3     4     5
84  23.33 31.92 33.17 18.42 47.25
87  50.25 17.67 39.92 31.25 21.08
93  34.25 28.92 52.33 35.00 34.25
98  23.00 69.50 23.50 41.58 35.75
255 29.67 20.67 27.25 50.75 20.42
287 30.17 33.92 19.50 24.42 18.83
330 26.17 22.75 36.42 16.50 47.83
446 43.08 18.58 32.00 27.00 24.08
451 41.42 47.42 29.25 24.50 41.42
501 24.33 30.50 32.75 36.75 39.00
516 52.50 34.92 41.17 31.83 20.50
609 46.08 40.58 27.83 51.92 31.58
```

For remaining attributes with categorical values, the missing values are approximated using the median of observations for simplicity.

III. METHODOLOGIES AND TECHNIQUES APPLIED ON DATASET

A. *Data Visualization*

A visual analysis was conducted on the dataset to have an idea of the possible relationships between the variables and observe any visible effect of each factor on whether an application is ultimately approved or not. This section details the approach taken.

Distribution of Continuous Variables

The distribution of the 5 continuous variables Age, Debt, Credit score, Income and Years employed was observed initially to have a sense of the nature of the dataset.

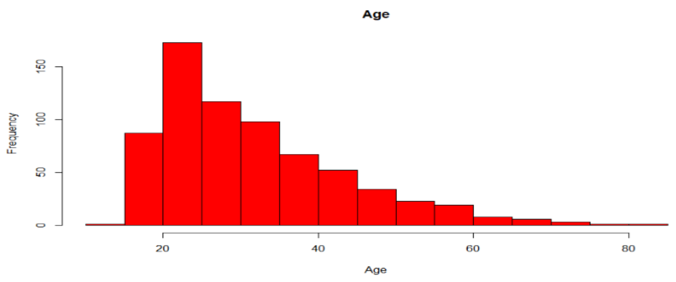


Fig. 1: Frequency distribution of Age

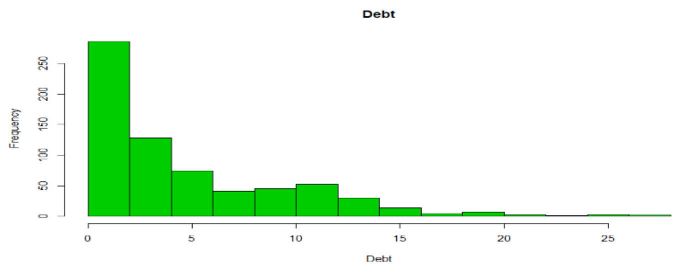


Fig. 2: Frequency distribution of Debt

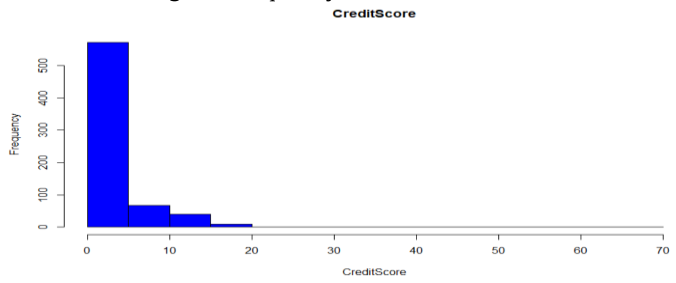


Fig. 3: Frequency distribution of Credit Score

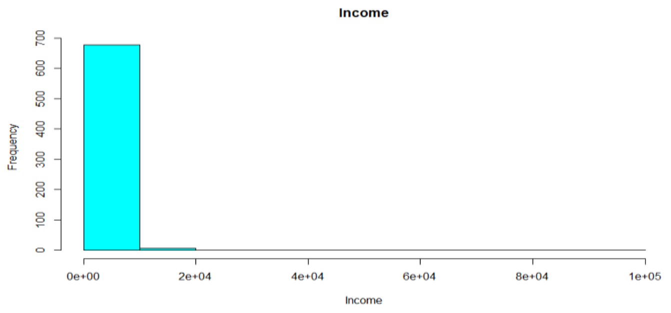


Fig. 4: Frequency distribution of Income

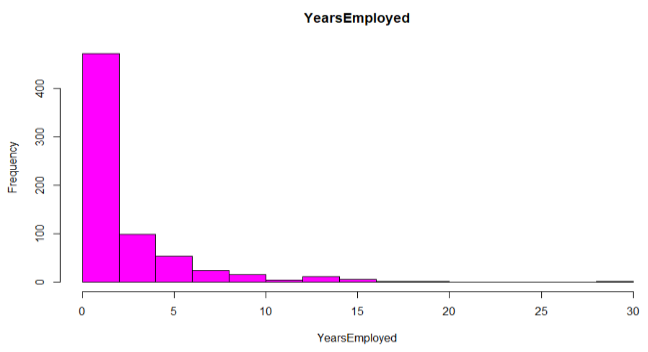


Fig. 5: Frequency distribution of Years Employed

These initial plots show that all the variables have distributions that are skewed to the right indicating that the data is not well distributed about the mean; very few persons in this sample recorded attribute values that are higher than the mean. This could be attributed to the population being from a single economic sector, for example. Log transformations were applied to these variables to reduce the skew and present better scaled plots as shown below.

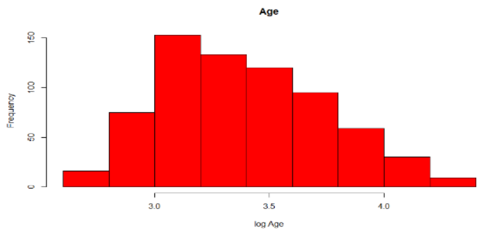


Fig. 6: Frequency distribution of log.Age

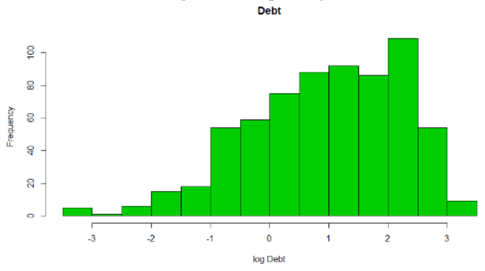


Fig. 7: Frequency distribution of log.Debt

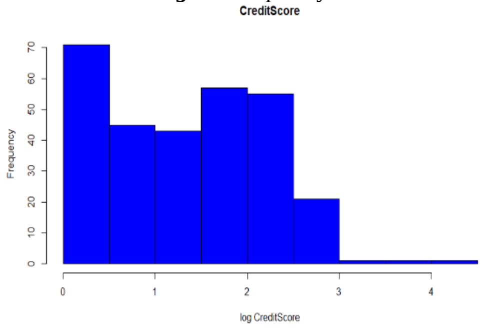


Fig. 8: Frequency distribution of log.Credit score

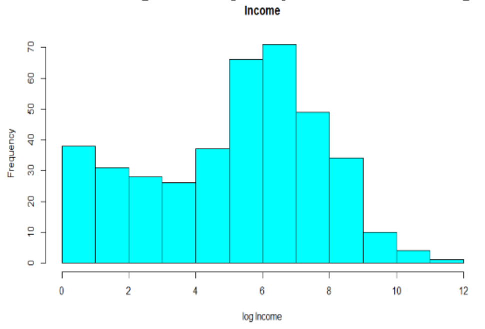


Fig. 9: Frequency distribution of log.Income

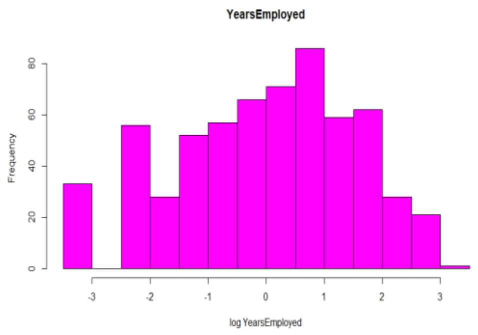


Fig. 10: Frequency distribution of log.YearsEmployed

Individual Variable's effect on Approval

Plots of individual variables were done to inspect visually whether the variable influenced application approval.

Below are plots of the discrete variables that appear to influence whether a credit application is approved.

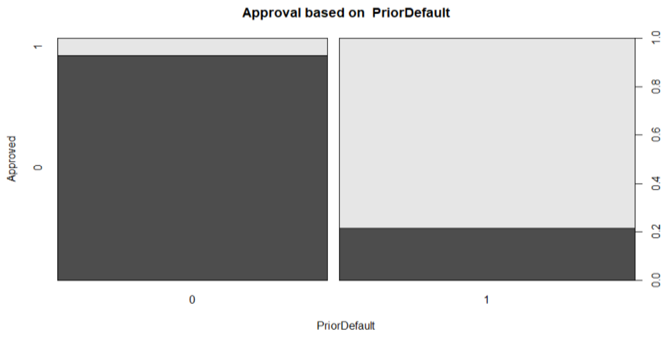


Fig. 11: Plot of Approved vs Prior default

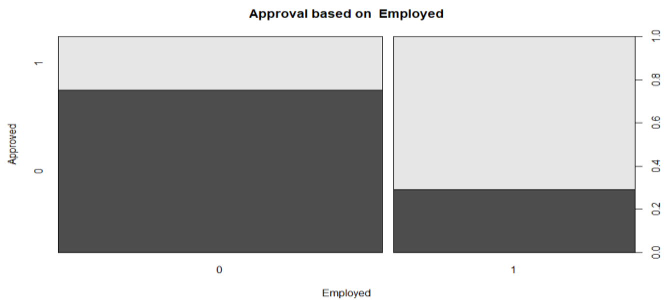


Fig. 12: Plot of Approved vs Employed

As expected, the variables Prior default and Employment status appear to have the most significant effect on credit approval. Persons with prior default are rejected more than 90% of the time versus 20% for those with no prior defaults.

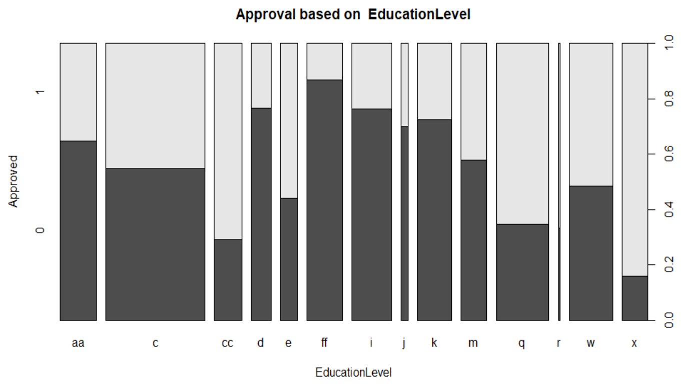


Fig. 13: Plot of Approved vs Education level

Less predictable, but with some significance in this dataset, a person's education level also seems to affect the outcome of a credit application. Persons with education level "x" have an 85% chance of being approved while those with education "ff" are rejected approximately 85% of the time.

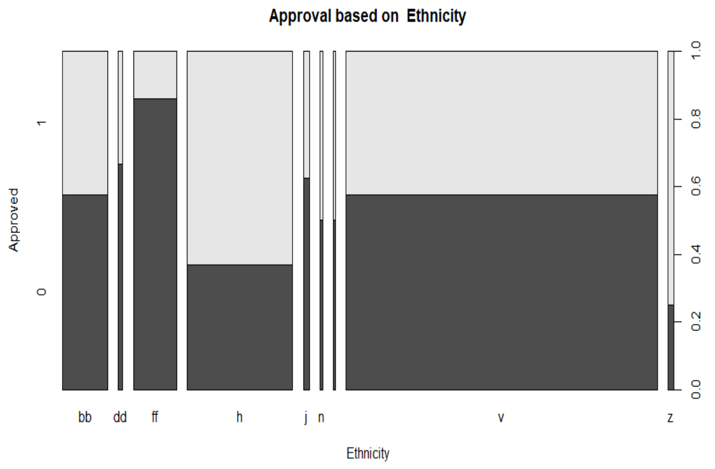


Fig. 14: Plot of Approved vs Ethnicity

The ethnicity variable, even though not used in the decision-making process for credit applications, shows an approval rate of about 75% for persons identified by group "z" and conversely, a rejection rate of nearly 90% for persons in group "ff". It would be useful to investigate the other attributes of persons in these two groups to see what results.

Among the continuous variables, Income and Credit Score seem to have significant effect on the outcome of credit applications as seen in the plots below.

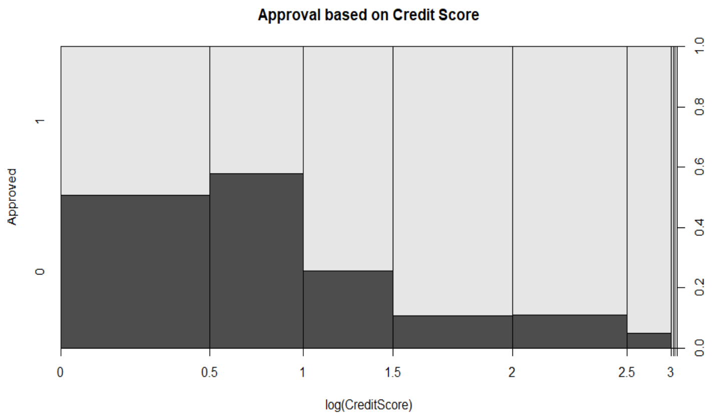


Fig. 15: Plot of Approved vs log.Credit score

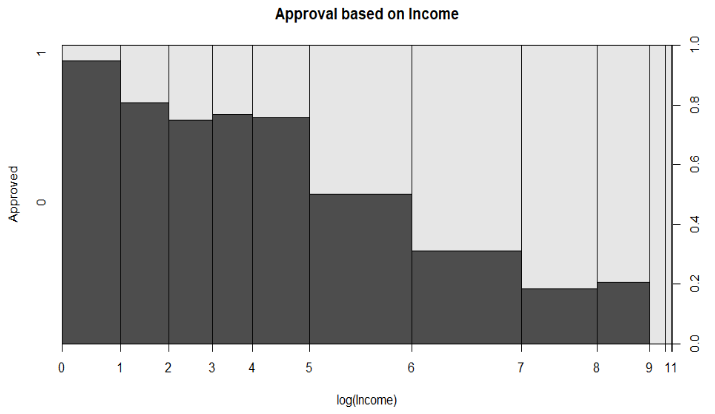


Fig. 16: Plot of Approved vs log.Income

A high credit score results in credit approval approximately 90% of the time and applicants with higher income have a higher than average approval rate.

Pairwise comparisons

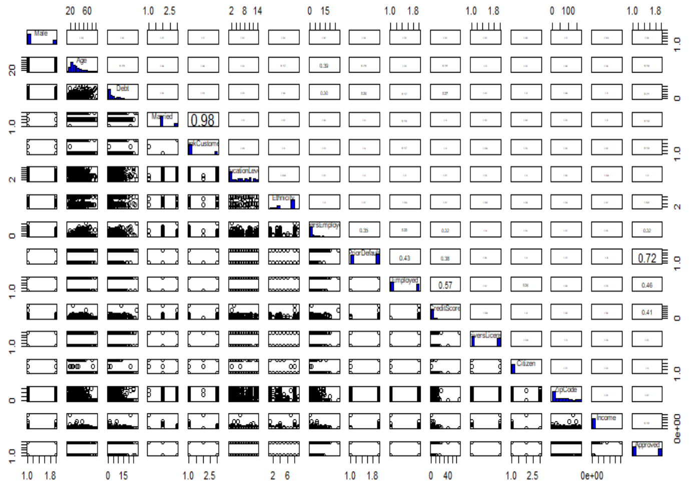


Fig. 17: Scatter plot matrix showing pairwise comparisons

From the pairwise comparisons in the scatterplot the variables Prior default, Employed, Credit score and

Years employed has the highest linear co-relation with the class attribute Approved. The relationship between Approved and the other variables is not immediately apparent from this plot.

Ggobi 2D Tour

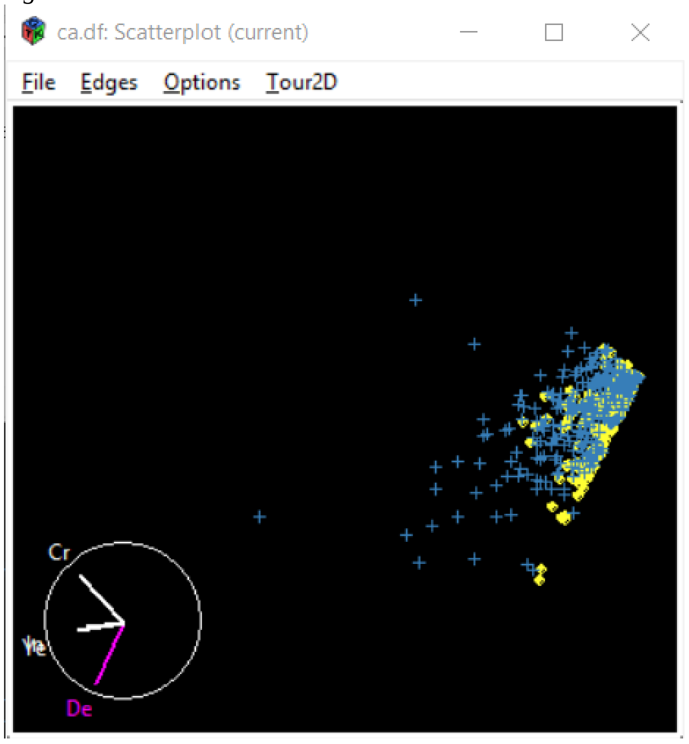


Fig. 18: Screen capture of 2D Tour

For this plot, the approved cases are represented by blue crosses and the rejected cases with yellow dots. In this display, the 5 continuous variables are projected in 2 dimensions and throughout the rotations, the rejected cases stay closely connected, while the approved cases move around much more freely. This suggests that there is much less variance in the characteristics of the rejected cases than in the approved cases.

Parallel Coordinate Plot

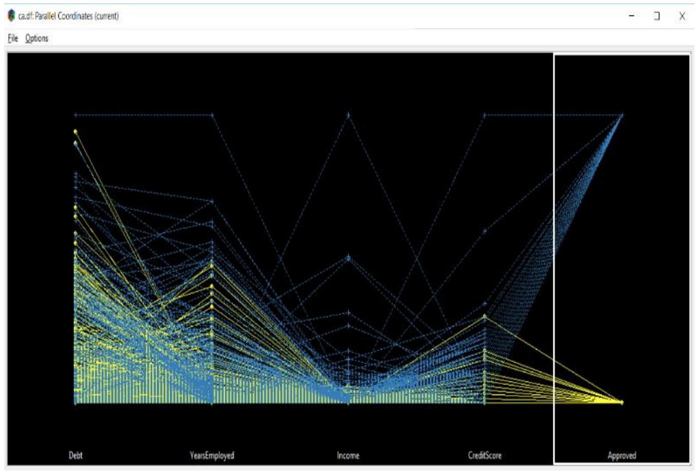


Fig. 19: Parallel coordinate plot

B. *Dimension Reduction*

The 15 input variables consist of 6 continuous and 4 binaries and 5 discreate. The binaries can be split into new binaries, for instance $Gender can be split into male and female. Discrete variables can be split into new binaries for each possible value, for instance $Citizen has possible values “g” and “p” and “s” to make 3 new binaries, with the 3rd as a combination of the 1st and 2nd. This makes a total 47 variables, with 9 binaries as combinations of other binaries, so that to start only 38 variables are needed. After filling in missing values and normalizing the continuous variables, Principal Component Analysis gives a data rotation, into 38 continuous principle components. The first 8 cover over 90% of the variance, with about 64% in the 1st and 7% in the 2nd.

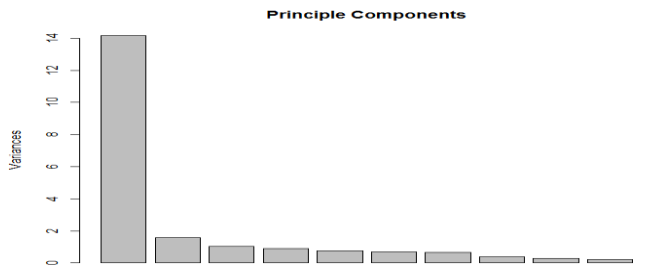


Fig. 20: Principal Component Variances

C. *Supervised Learning*

Classification: Classification is termed under supervised learning where the information/class to be predicted is already available. It helps in training the model with available results and can be tested on unknown similar dataset using past information. Prediction and estimation is a kind of classification.

Generating Analytic Models

The chosen dataset is split into training and test datasets for evaluating model and its accuracy. The training set is used to build the model and test dataset will validate the accuracy of the developed model. The dataset is divided in the ratio of 65 % training set and 35 % test dataset. Training set will constitute almost in the range of 448 observations and 242 observations for test cases. Out of 426 observations, major portion around 237 cases or 56% are rejected. The baseline models are prepared using training set and it is expected that in the predictive model using test dataset more of the cases would be denied. The models should be at least accurate 56% of the time to call it an efficient model.

```
> library(caTools)
> set.seed(1234)
> split<- sample.split(CoCA3, SplitRatio=0.65)
> Train<- subset(CoCA3,split==TRUE)
> Test <- subset(CoCA3, split==FALSE)
> table(Train$Approved)

  0   1
237 189
```

Seed() is used for the reproducibility of the results.

Logistic Regression

Regression models generally deal with continuous values but in our case the target values in Approved are binary. Liner Regression would not work well because of its tendency to produce the result outside our binary range. Logistic Regression works well with categorical values. As it deals with probabilities, the predicted values are based on likelihood of events using logit function. LR are special part of generalized linear models and there exists linear relation between link function and predictors. The general equation of linear model is

$g(E(y)) = \alpha + \beta x1 + \gamma x2$

and the probability is given by

$$p = E(Y) = \left[1 + e^{-(\beta_0 + \beta_1 X_1 + \ldots + \beta_k X_k)}\right]^{-1}$$

Creating Model

The model is prepared using *glm* function and binomial family. Choosing the potential attributes determined before, class variable is compared with them for finding the significance of each variable. The summary of the model shows the values of coefficients and the asterisk on the selected attributes states the importance. Therefore, the attributes with no asterisk signs are removed for revised model and IncomeLog, PriorDefault, and YearsEmployed are considered reflecting more influence.

```
> summary(model)

Call:
glm(formula = Approved ~ PriorDefault + CreditScoreLog + YearsEmployedLog +
    Employed + AgeLog + DebtLog + IncomeLog, family = binomial,
    data = Train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3852  -0.3908  -0.2319   0.4957   2.7634

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)       -2.4771     0.4179  -5.927 3.09e-09 ***
PriorDefault1      3.4200     0.3630   9.421  < 2e-16 ***
CreditScoreLog     0.3463     0.3338   1.038  0.29941
YearsEmployedLog   0.4811     0.1772   2.716  0.00661 **
Employed1          0.3554     0.6021   0.590  0.55503
AgeLog            -0.1379     0.1547  -0.892  0.37264
DebtLog           -0.1631     0.1553  -1.050  0.29354
IncomeLog          0.4859     0.1609   3.020  0.00253 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 585.14  on 425  degrees of freedom
Residual deviance: 293.45  on 418  degrees of freedom
AIC: 309.45

Number of Fisher Scoring iterations: 5
```

The results of confusion matrix of this model are

```
  FALSE TRUE
0   201   36
1    63  126
```

The model shows the accuracy of 77% from 327 correctly predicted results which is better than baseline model.

Revised Model: Using the revised model by considering significant variables to predict values on training dataset. The predicted values are found using the threshold of 50% which gives the confusion matrix

```
   FALSE TRUE
0    190   47
1     17  172
```

The accuracy of this model has significantly improved to 85% from 77% compared to the previous model. The model can be simplified using step() which selects the significant variables based on the low values of AIC. The result shows CreditScoreLog as the significant variable instead of PriorDefault and rest remain the same. But the accuracy 77% of this model is like first model developed. So, the second model proves much more efficient in terms of accuracy and can be used to apply to test dataset.

Applying the model

The second revised model is applied to the test dataset and the result of the confusion matrix shows 85% accuracy in predicting 226 observations correctly out of 264.

```
> test.pred<-predict(fitmodel, newdata=Test,type="response")
> table(Test$Approved, test.pred>0.5)

   FALSE TRUE
0    120   26
1     12  106
```

The misclassification error is 14.3 % wit this model. Therefore, we can summarize the significant variables are Income, PriorDefault, YearsEmployed and Credit Score from the developed models.

```
-----------Misclassification Error---------
 [1] 0.1439394
```

Visual result of Logistic Regression on Revised Model

The plot below contains the points of training and test dataset considering attributes IncomeLog and YearsEmployedLog. A line showing separation between the classes with the misclassified cases highlighted with triangle in blue colour for both training and test cases. The test cases are shown by '*'. There are 38 misclassified cases which are found and shown on graph as well validating confusion matrix results.

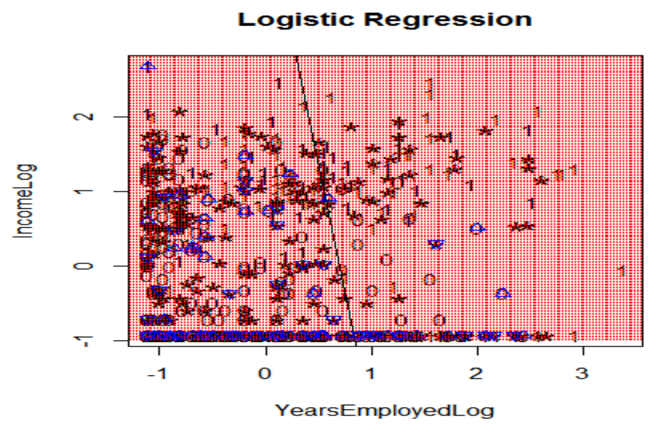
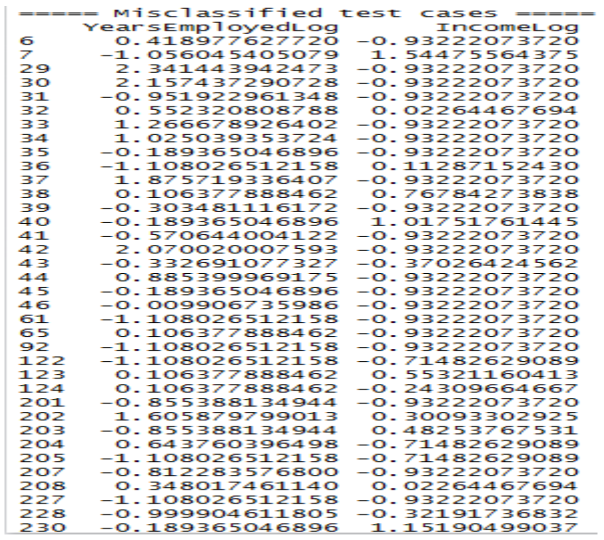


Fig 21: Logistic Regression on Revised Model



```
===== Misclassified test cases =====
     YearsEmployedLog        IncomeLog
6      0.418977627720 -0.93222073720
7     -1.056045405079  1.54475564375
29     2.341443942473 -0.93222073720
30     2.157437290728 -0.93222073720
31    -0.951922961348 -0.93222073720
32     0.552320808788  0.02264467694
33     1.266678926402 -0.93222073720
34     1.025039353724 -0.93222073720
35    -0.189365046896 -0.93222073720
36    -1.108026512158  0.11287152430
37     1.875719336407 -0.93222073720
38     0.106377888462  0.76784273838
39    -0.303481116172 -0.93222073720
40    -0.189365046896  1.01751761445
41    -0.570644004122 -0.93222073720
42     2.070020007593 -0.93222073720
43    -0.332691077327 -0.37026424562
44     0.885399969175 -0.93222073720
45    -0.189365046896 -0.93222073720
46    -0.009906735986 -0.93222073720
61    -1.108026512158 -0.93222073720
65     0.106377888462 -0.93222073720
92    -1.108026512158 -0.93222073720
122   -1.108026512158 -0.71482629089
123    0.106377888462  0.55321160413
124    0.106377888462 -0.24309664667
201   -0.855388134944 -0.93222073720
202    1.605879799013  0.30093302925
203   -0.855388134944  0.48253767531
204    0.643760396498 -0.71482629089
205   -1.108026512158 -0.71482629089
207   -0.812283576800 -0.93222073720
208    0.348017461140  0.02264467694
227   -1.108026512158 -0.93222073720
228   -0.999904611805 -0.32191736832
230   -0.189365046896  1.15190499037
```

```
Area Under the Curve:  0.8327142
```

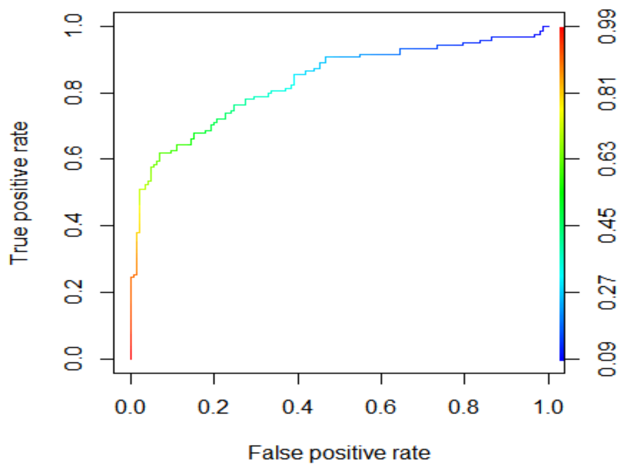


Fig 23: Area under the curve for initial model

```
Area Under the Curve:  0.9044579
```

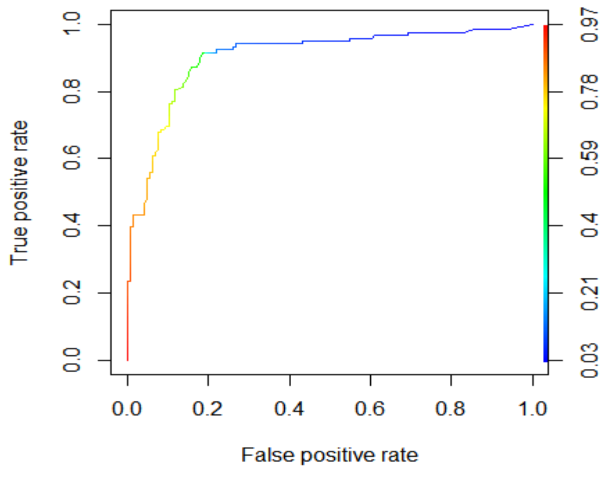


Fig 24 AUC of Revised Model

Area Under the curve/ROC is another performance metrics used to evaluate the trade off between sensitivity (True Positive Rate) and Specificity (False Positive Rate). It can be noticed that the area under curve for revised model is more than the first model created using all continuous and important discrete attributes. Higher the AUC, better is the model. It is important for a model to predict defaulters correctly as the misclassification cost of not predicting defaulters is much more than not predicting non-defaults correctly as it may affect business.

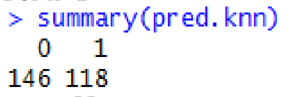
K-Nearest Neighbor Method

k-NN algorithm predicts the values of test dataset from the training datasets following distance (Euclidean or Hamming) similarity measure. k-NN is a non-parametric classification method which does not make assumptions about the distribution of the data and is distribution independent. This classifier when given with unknown test case finds pattern from training cases and these cases acts as neighbors to test cases. The similar instances of prediction attribute to unseen test cases are summarized and acts as prediction for the unseen cases. It is also referred as lazy algorithm. The selection of the value of k for the algorithm is important and can be done experimentally by starting with k=1. Smaller value will have noise and higher value will be computationally expensive. The value of k with least error rate is optimal and can be found by incrementing it or using cross validation.

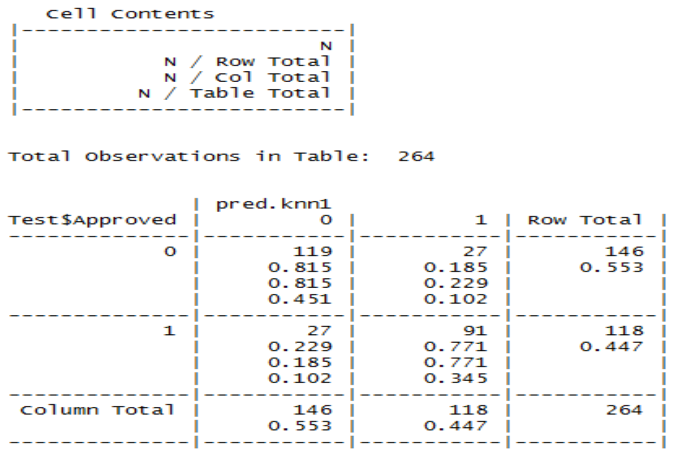
Analysis on Dataset

The attributes in the training and test dataset are chosen based on the significance it holds, normally contains all continuous attributes. Using library(class) for k-NN function with initial value of k=1 is applied on test and training dataset. The values of k are varied to find the least error rate and more accuracy. knn() returns the predicted values of classes and can be verified with actual outcomes.

For k=1



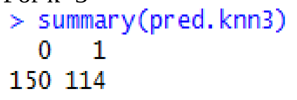
```
> summary(pred.knn)
  0   1
146 118
```



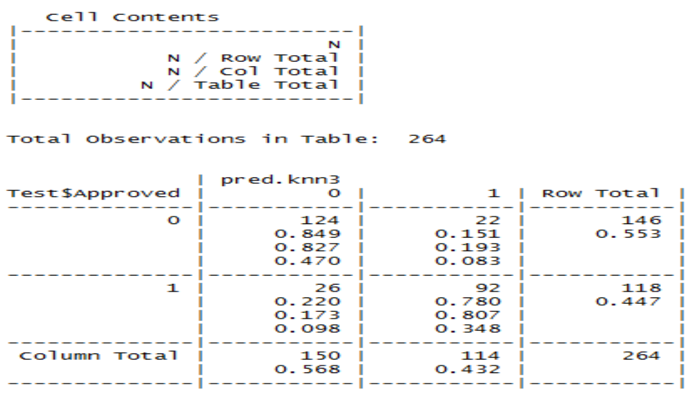
```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  264


              | pred.knn1
Test$Approved |         0 |         1 | Row Total |
--------------|-----------|-----------|-----------|
            0 |       119 |        27 |       146 |
              |     0.815 |     0.185 |     0.553 |
              |     0.815 |     0.229 |           |
              |     0.451 |     0.102 |           |
--------------|-----------|-----------|-----------|
            1 |        27 |        91 |       118 |
              |     0.229 |     0.771 |     0.447 |
              |     0.185 |     0.771 |           |
              |     0.102 |     0.345 |           |
--------------|-----------|-----------|-----------|
 Column Total |       146 |       118 |       264 |
              |     0.553 |     0.447 |           |
--------------|-----------|-----------|-----------|
```

The results of the crosstable show that out of 146, 119 are true negative and 91 are true positive out of 118 giving accuracy 79%.

For k=3



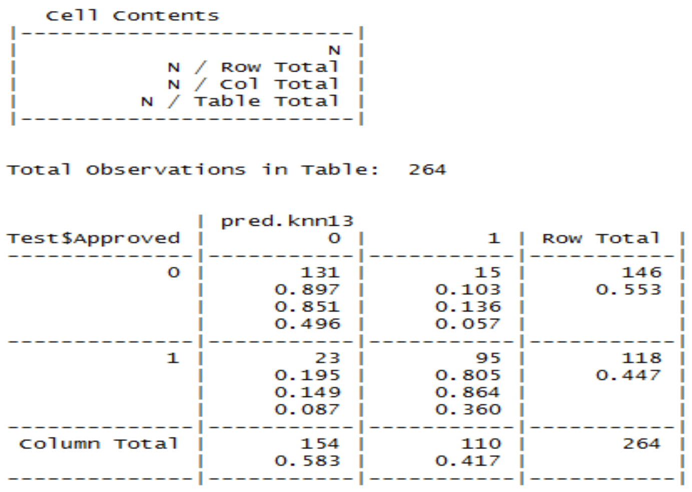
```
> summary(pred.knn3)
  0   1
150 114
```



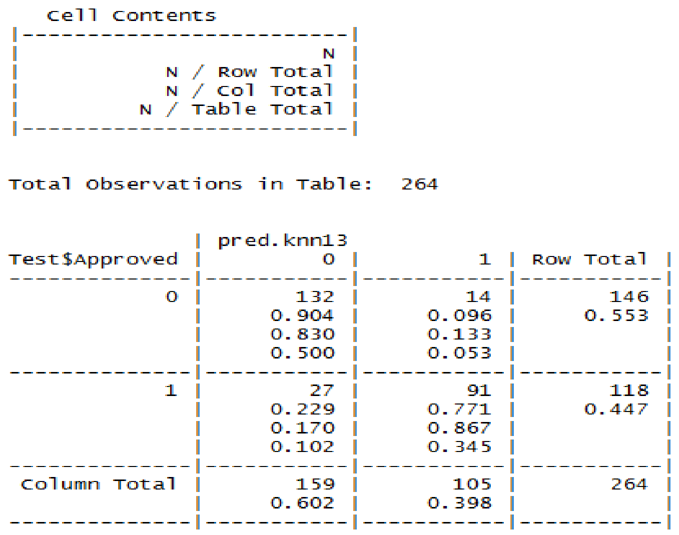
```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  264


              | pred.knn3
Test$Approved |         0 |         1 | Row Total |
--------------|-----------|-----------|-----------|
            0 |       124 |        22 |       146 |
              |     0.849 |     0.151 |     0.553 |
              |     0.827 |     0.193 |           |
              |     0.470 |     0.083 |           |
--------------|-----------|-----------|-----------|
            1 |        26 |        92 |       118 |
              |     0.220 |     0.780 |     0.447 |
              |     0.173 |     0.807 |           |
              |     0.098 |     0.348 |           |
--------------|-----------|-----------|-----------|
 Column Total |       150 |       114 |       264 |
              |     0.568 |     0.432 |           |
--------------|-----------|-----------|-----------|
```

The result of the confusion matrix for k=3 is better than k=1. The classification for true positive is 92 and true negative is 124 giving an overall accuracy of 82%. Similarly, for k=5, the overall accuracy improved to 86%. The values of k are further incremented to test the accuracy and the classification rate. It can be noticed that with the increase in the value of k, the classification rate stays within the range of 82 to 85% past k=5. Further, the sensitivity (true positive) reduces and false negative increases with higher values of k. This is worse for a practical situation where a customer is classified as good when he is at bad credit risk than predicting good customer bad. The confusion matrix for k=5 and k=11 is shown below.

For k=5 with classification rate of 86%
K=11 with Classification rate of 84%



```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  264


              | pred.knn13
Test$Approved |         0 |         1 | Row Total |
--------------|-----------|-----------|-----------|
            0 |       131 |        15 |       146 |
              |     0.897 |     0.103 |     0.553 |
              |     0.851 |     0.136 |           |
              |     0.496 |     0.057 |           |
--------------|-----------|-----------|-----------|
            1 |        23 |        95 |       118 |
              |     0.195 |     0.805 |     0.447 |
              |     0.149 |     0.864 |           |
              |     0.087 |     0.360 |           |
--------------|-----------|-----------|-----------|
 Column Total |       154 |       110 |       264 |
              |     0.583 |     0.417 |           |
--------------|-----------|-----------|-----------|
```



```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  264


              | pred.knn13
Test$Approved |         0 |         1 | Row Total |
--------------|-----------|-----------|-----------|
            0 |       132 |        14 |       146 |
              |     0.904 |     0.096 |     0.553 |
              |     0.830 |     0.133 |           |
              |     0.500 |     0.053 |           |
--------------|-----------|-----------|-----------|
            1 |        27 |        91 |       118 |
              |     0.229 |     0.771 |     0.447 |
              |     0.170 |     0.867 |           |
              |     0.102 |     0.345 |           |
--------------|-----------|-----------|-----------|
 Column Total |       159 |       105 |       264 |
              |     0.602 |     0.398 |           |
--------------|-----------|-----------|-----------|
```

The optimal value of k= 5 for classification of dataset using k-NN algorithm.

Linear and Quadratic Discriminant Analysis

LDA is a classification algorithm that finds the linear combination of predictors to best separate two or more classes. It can be useful for multi class classification. LDA is like PCA but it focuses more on maximizing the seperability among the known categories. It uses a single axis to project the data onto new axis in such a way that maximize the separation between categories and minimize the variation. LDA considers the means and co variance for multivariate data to make predictions. The assumption it makes is that data is Gaussian and the values of each variable vary around mean meaning each attribute has same variance. This makes a model of the distribution of predictors for each class and applies Bayes' theorem to turn them into predictions.

LDA assumes the conditional probability p(x/y=1) and p(x/y=2) are normally distributed having means and covariance parameters (μ1, Σ1) and (μ2, Σ2). Equation is given by

$$p(X|y=k) = \frac{1}{(2\pi)^n|\Sigma_k|^{1/2}}\exp\left(-\frac{1}{2}(X-\mu_k)^t\Sigma_k^{-1}(X-\mu_k)\right)$$

The LDA classifier has equal covariance matrix Σ1=Σ2=Σ and is given by $x^T\Sigma^{-1}\mu_l - 0.5\mu_l^T\Sigma^{-1}\mu_l + \log(\pi(y=C_l))$.

Whereas in QDA, there are no assumptions in considering covariance matrix that each class has its own estimate of covariance and is given by

$$-0.5\log|\Sigma_l| - 0.5(x-\mu_l)^T\Sigma_l^{-1}(x-\mu_l) + \log(\pi(y=C_l)).$$

Model building

In this analysis, the purpose is to find the linear combinations of original variables which are 15 in this case (except class variable) that separates the group (approved) in best possible way. We have two groups (1 and 0) in class variable and the maximum number of discriminant function is minimum number of variables and number of groups-1. So, in this case there will be 1 LDA function. The model is built using 5 variables which are continuous and further normalized, with others as categorical are not considered.

The summary of the lda applied on training set is shown below indicating the LD1 function which is a liner combination of specified attributes along with group means of each attribute.

```
Call:
lda(Train$Approved ~ ., data = Cred.data)

Prior probabilities of groups:
       0        1 
0.556338 0.443662 

Group means:
   IncomeLog CreditScoreLog     AgeLog YearsEmployedLog    DebtLog
0 -0.2532779     -0.4653142 -0.1226674       -0.3877497 -0.1557985
1  0.4008964      0.5619177  0.1397391        0.4629560  0.2667739

Coefficients of linear discriminants:
                         LD1
IncomeLog         0.30986239
CreditScoreLog    0.69129337
AgeLog           -0.02442371
YearsEmployedLog  0.56310426
DebtLog           0.03710201
```

Evaluation of Model

The result of the confusion matrix using LDA shows that 331 cases are correctly classified out of 426 cases in a training set.

```
> ct
   predict.exp
      0   1
  0 211  26
  1  69 120
```

Therefore, giving the classification accuracy of 77%.

```
> sum(diag(prop.table(ct)))
[1] 0.7769953
```

Applying the Model: The model is then applied to test dataset constituting 264 cases. The result of the confusion matrix is not an improvement but close to the results obtained for training dataset. 202 cases out of 264 are correctly classified giving an accuracy of 76%

```
> ct2
   predict.exp2
      0   1
  0 125  21
  1  41  77
> diag(prop.table(ct2, 1))
        0         1 
0.8561644 0.6525424 
> sum(diag(prop.table(ct2)))
[1] 0.7651515
```

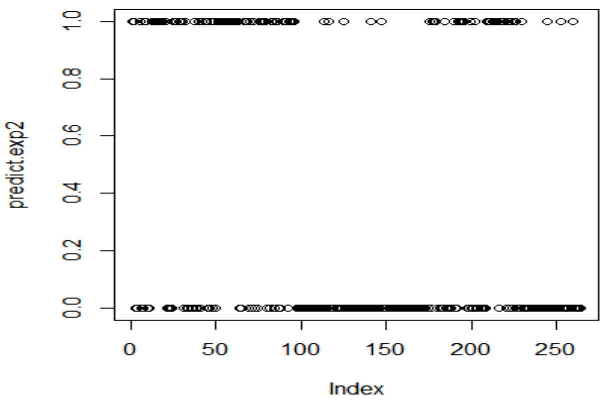


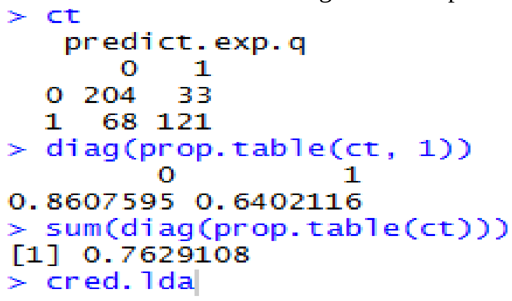
Fig 25: Plot of prediccted results from LDA

The plot of the predicted results from lda using test data set is shown in fig above demonstrating more number of applications are disapproved.

On Building model for QDA, following is the summary of attributes with group means and prior probabilities of groups of class variable.

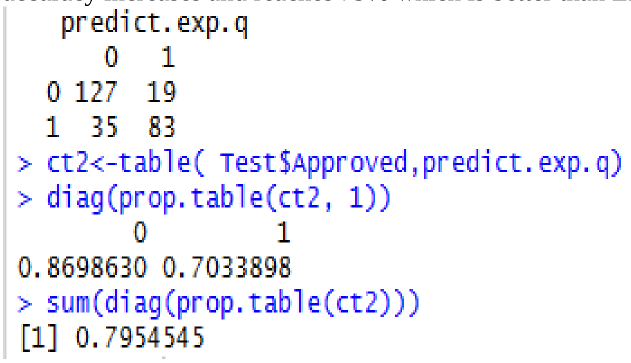
```
Call:
qda(Train$Approved ~ ., data = Cred.data)

Prior probabilities of groups:
       0        1 
0.556338 0.443662 

Group means:
   IncomeLog CreditScoreLog     AgeLog YearsEmployedLog    DebtLog
0 -0.2532779     -0.4653142 -0.1226674       -0.3877497 -0.1557985
1  0.4008964      0.5619177  0.1397391        0.4629560  0.2667739
```

Similarly, the model is applied to training dataset and the result of the confusion matrix shows the classification accuracy of 76% which is close to LDA. NO significant improvement can be seen.



```
> ct
   predict.exp.q
      0    1
  0 204   33
  1  68 121
> diag(prop.table(ct, 1))
        0         1
0.8607595 0.6402116
> sum(diag(prop.table(ct)))
[1] 0.7629108
> cred.lda
```

On applying the model to test dataset, the classification accuracy increases and reaches 79% which is better than LDA.



```
   predict.exp.q
     0   1
  0 127  19
  1  35  83
> ct2<-table( Test$Approved,predict.exp.q)
> diag(prop.table(ct2, 1))
        0         1
0.8698630 0.7033898
> sum(diag(prop.table(ct2)))
[1] 0.7954545
```

Therefore, QDA separates the group of class attributes considering other attributes better than LDA.

Partition Plot using LDA.

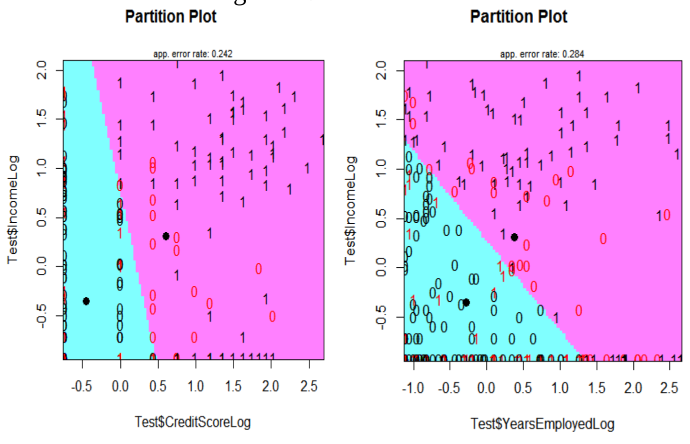


Fig 26: Plots partitioning classes using LDA

Partition Plot Using QDA

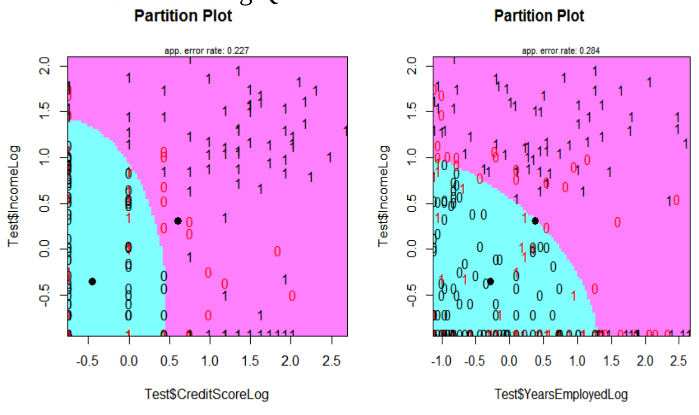


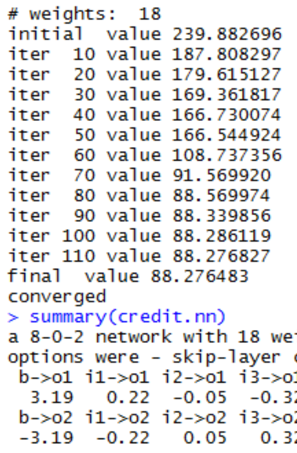
Fig. 27: Fig 26: Plots partitioning classes using QDA

As we can observe, QDA segregate two classes efficiently with least misclassification while cross validating with test set.

Classification using Neural Networks

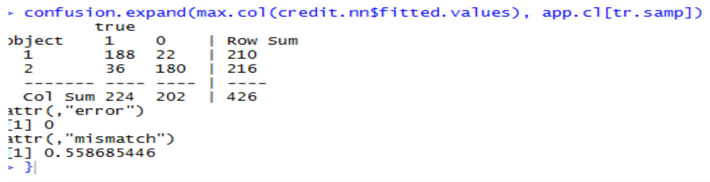
We used the library(nnet) to build this model. The parameters chosen for building model are decay=0.001, range=0.1 with maximum of 500 iterations. The samples for training and test are extracted using function get.train() and saved for the reproducibility of results.

The model is prepared by comparing specific attributes(non-characters) with class variable and initially, no hidden layer is assumed. Attributes with character type are ignored as they were introducing error in the nnet function of NA/NaN/Inf during function call in arg2. 18 weights are computed with the final value of 88.2. The summary of the network is shown below. There are 8 inputs and 2 outputs which indicate group values of class attribute.



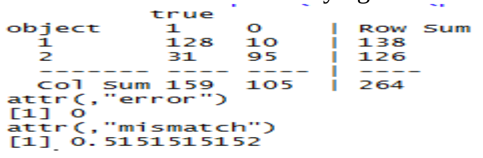
```
# weights:  18
initial  value 239.882696
iter  10 value 187.808297
iter  20 value 179.615127
iter  30 value 169.361817
iter  40 value 166.730074
iter  50 value 166.544924
iter  60 value 108.737356
iter  70 value 91.569920
iter  80 value 88.569974
iter  90 value 88.339856
iter 100 value 88.286119
iter 110 value 88.276827
final  value 88.276483
converged
> summary(credit.nn)
a 8-0-2 network with 18 weights
options were - skip-layer connections  decay=0.001
 b->o1 i1->o1 i2->o1 i3->o1 i4->o1 i5->o1 i6->o1 i7->o1 i8->o1
  3.19   0.22  -0.05  -0.32  -0.78  -0.48  -4.45  -0.54   0.08
 b->o2 i1->o2 i2->o2 i3->o2 i4->o2 i5->o2 i6->o2 i7->o2 i8->o2
 -3.19  -0.22   0.05   0.32   0.78   0.48   4.45   0.54  -0.08
```

Using the fitted values of the model and checking the accuracy, it can be found that object 1 refers '1' and 2 refers value '0' from Approved dataset. There is a mismatch in classification where 188 are correctly classified as 1 (approved) and 180 are classified as 0 denied on training dataset giving an error of 13.6%.



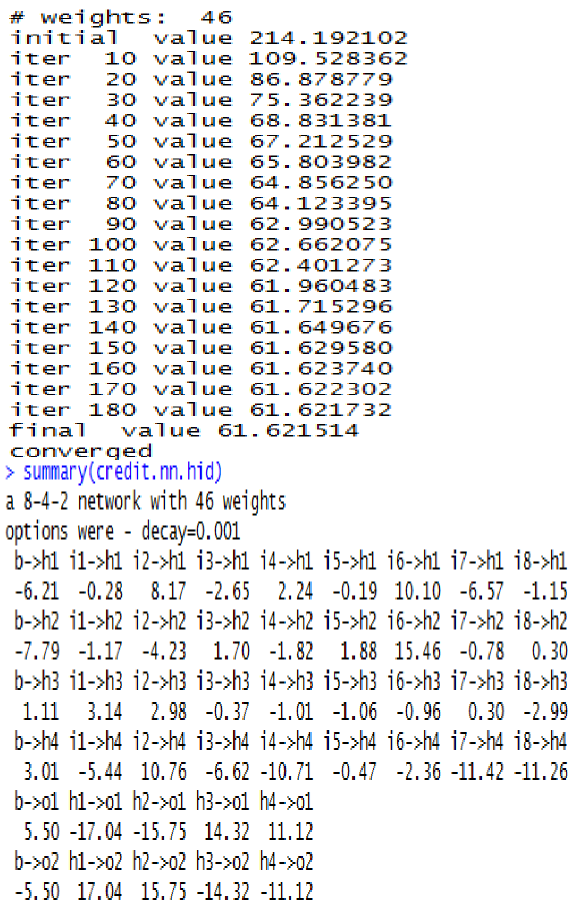
```
> confusion.expand(max.col(credit.nn$fitted.values), app.cl[tr.samp])
          true
object     1     0    | Row Sum
  1        188   22   | 210
  2        36    180  | 216
  ------- ---- ----  | ----
  col Sum 224   202   | 426
attr(,"error")
[1] 0
attr(,"mismatch")
[1] 0.558685446
> }
```

Now, using the predict method to find the correct classification of our dataset by the model. The function is applied on test dataset and the result of confusion matrix shows that the classification done is pretty much close in accuracy when done on training set. The error rate is 15% in classifying test dataset.



```
            true
object      1     0     | Row Sum
   1        128   10    | 138
   2        31    95    | 126
   -------  ----  ----  | ----
   col Sum  159   105   | 264
attr(,"error")
[1] 0
attr(,"mismatch")
[1] 0.5151515152
```

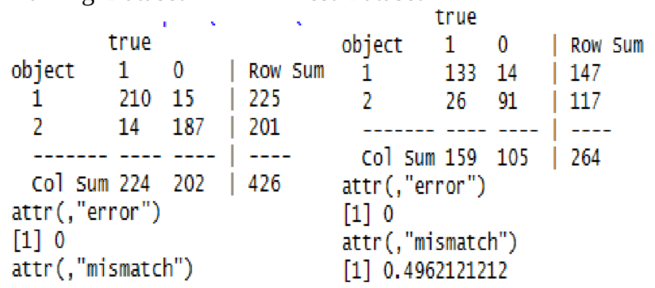
Now, the model is further improved by introducing hidden layer in the neural network. We have not followed any rule for selecting the number of hidden layers. We ran the code for different values of size in the function to get better classification. Choosing the value of size =4, the summary of the updated model is shown below. There are 46 weights with 8 inputs, 4 hidden layers and 2 output nodes.



```
# weights:  46
initial  value 214.192102
iter  10 value 109.528362
iter  20 value 86.878779
iter  30 value 75.362239
iter  40 value 68.831381
iter  50 value 67.212529
iter  60 value 65.803982
iter  70 value 64.856250
iter  80 value 64.123395
iter  90 value 62.990523
iter 100 value 62.662075
iter 110 value 62.401273
iter 120 value 61.960483
iter 130 value 61.715296
iter 140 value 61.649676
iter 150 value 61.629580
iter 160 value 61.623740
iter 170 value 61.622302
iter 180 value 61.621732
final  value 61.621514
converged
> summary(credit.nn.hid)
a 8-4-2 network with 46 weights
options were - decay=0.001
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1 i5->h1 i6->h1 i7->h1 i8->h1
 -6.21  -0.28   8.17  -2.65   2.24  -0.19  10.10  -6.57  -1.15
 b->h2 i1->h2 i2->h2 i3->h2 i4->h2 i5->h2 i6->h2 i7->h2 i8->h2
 -7.79  -1.17  -4.23   1.70  -1.82   1.88  15.46  -0.78   0.30
 b->h3 i1->h3 i2->h3 i3->h3 i4->h3 i5->h3 i6->h3 i7->h3 i8->h3
  1.11   3.14   2.98  -0.37  -1.01  -1.06  -0.96   0.30  -2.99
 b->h4 i1->h4 i2->h4 i3->h4 i4->h4 i5->h4 i6->h4 i7->h4 i8->h4
  3.01  -5.44  10.76  -6.62 -10.71  -0.47  -2.36 -11.42 -11.26
 b->o1 h1->o1 h2->o1 h3->o1 h4->o1
  5.50 -17.04 -15.75  14.32  11.12
 b->o2 h1->o2 h2->o2 h3->o2 h4->o2
 -5.50  17.04  15.75 -14.32 -11.12
```

The result of the confusion matrix using training set shows 93% classification accuracy. The model is tested on test dataset and achieves the classification accuracy of 85%. The result of accuracy keeps on varying with each iteration.

Training Dataset   Test Dataset



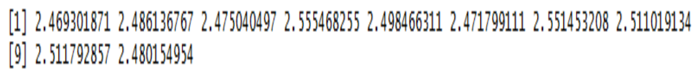
```
          true
object    1    0    | Row Sum
  1       210  15   | 225
  2       14   187  | 201
  ------- ---- ---- | ----
  Col Sum 224  202  | 426
attr(,"error")
[1] 0
attr(,"mismatch")
```

```
           true
object     1    0    | Row Sum
1          133  14   | 147
2          26   91   | 117
-------    ---- ---- | ----
Col Sum    159  105  | 264
attr(,"error")
[1] 0
attr(,"mismatch")
[1] 0.4962121212
```

It was concluded that the model is showing high classification accuracy when the number of observations in the dataset are large. As our test dataset have around 35% of original dataset components, the cases are less and the misclassification rate is high on applying this model.

We chose the value of size =4 and applied a loop 10 times to find the Mean Square error for every time the neural net gives result. The progress is shown below with different MSE.

Error for 10 iterations are



```
[1] 2.469301871 2.486136767 2.475040497 2.555468255 2.498466311 2.471799111 2.551453208 2.511019134
[9] 2.511792857 2.480154954
```

The plot showing the MSE

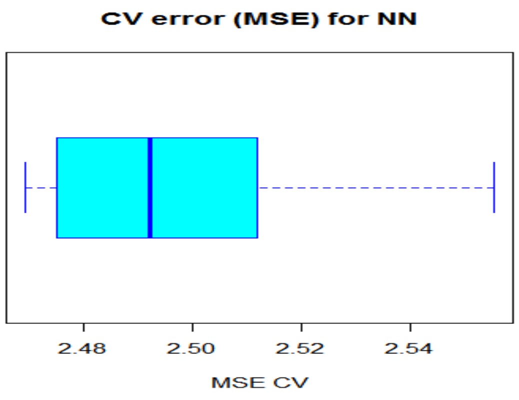


Fig. 29: Boxplot of MSE on NN

Unsupervised Learning

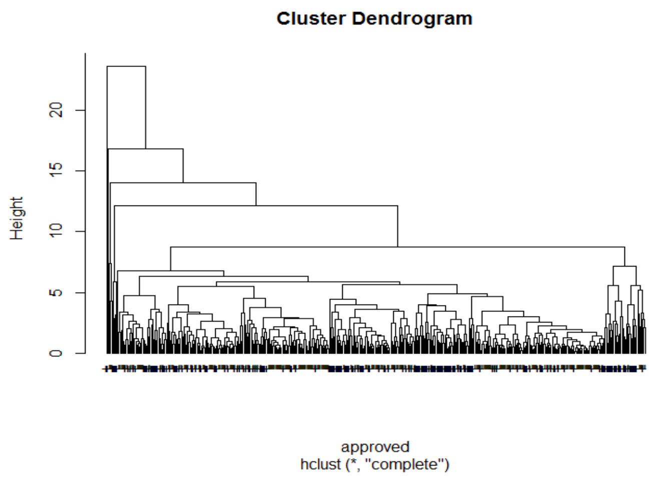


Fig. 30: Dendrogram showing clusters based on classes

Kmeans	-	+	Actual
-	118	21	139
+	68	35	103
Guess	200	42	242

Unsupervised learning by clustering does not provide much help since cases overlap a lot. The cases split by + for approved and by – for not approved. The data then shuffled and then split into 65% for training and 35% for testing. The kmeans of two clusters has an error of about 37% from (68+21)/242.

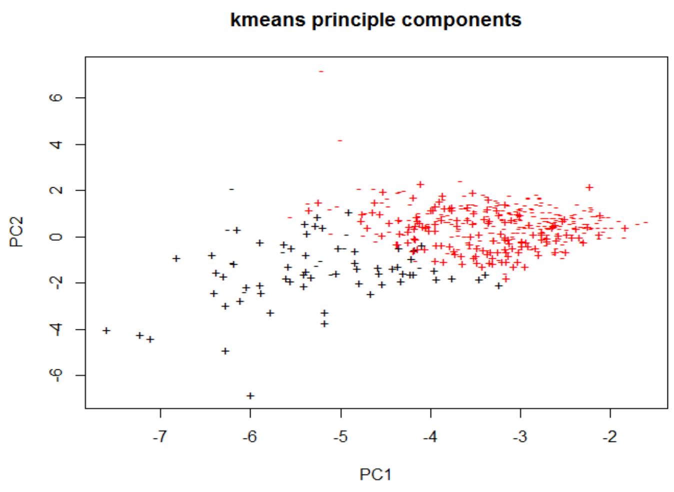


Fig. 31: Result of K-Means on Principle Components

Supervised Learning Through Classification Tree

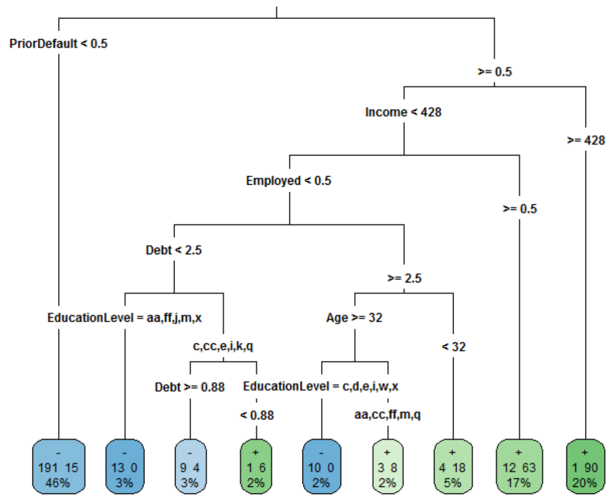


Fig. 32: Classification Tree using Entropy

Entropy	-	+	Actual
-	121	18	139
+	13	90	103
Guess	134	108	242

Classification trees make a much better prediction then kmeans, with an error of about 13%, from entropy measure (total*ln(total)-∑class*log(class)). The top three predictors are Prior Default covering 46%, then Income covering 20%, the Employed covering 17%, for a total of covering of 83%.

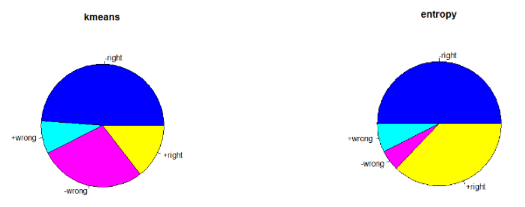


Fig. 33: Comparing K-Means to Binary Trees

Above is a comparison of kmeans on 2 clusters to entropy classification with errors of 37% to 13%.

Results and Comparison of Classifiers

The results of the comparison between classifiers are shown in the table. The classification rate is deduced using test dataset on the developed model. The results of prediction of the classes are found using predict() and demonstrated using confusion matrix in order to determine classification accuracy. From the results obtained, Classification and Regression Tree method outperforms all the other classifiers with highest accuracy of 87.2%. Logistic Regression performs separation much efficiently for when the number of classes are two. On the other hand, LDA and QDA have poor classification accuracy for the chosen dataset. Neural Network and K-NN performs satisfactorily well but they require persistent testing of codes using hit and trial to find optimum values for the desired model.

Classifier using Method	Classification Accuracy in % for Test Set
Logistic Regression	85.6
k-Nearest Neighbors	85.6
Linear Discriminant Analysis	76.5
Quadratic Discriminant Analysis	79.5
Neural networks	84.8
Classification and Regression Tree(CART)	87.2

Table 3: Comparison of Classifiers

IV. SUMMARY OF FINDINGS

Visualization

Using the visualization techniques outlined above, we were able to make the following deductions:

i. Prior default, Employment status and Credit score are the strongest predictors of whether an application will be approved; applicants with high credit scores are almost always approved for credit while those with prior default are denied about 90% of the time.

ii. A higher income increases an applicant's chances of being approved for credit.

iii. The measurable attributes of declined applications have a smaller range of values than those of approved applications.

iv. Education level, though not a conventional predictor of credit approval, does show some correlation with the rate of approval. Applicants with one level of education have much higher approval rates than applicants with other levels of education. One possible explanation could be that with a higher level of education, a person will have a higher income, thereby increasing the chance of approval.

v. Even though ethnicity is a protected attribute and does not factor in the credit approval decision, applicants identifying with one ethnicity received a significantly high rate of approval while applicants another had a higher than average rate of denial

Supervised Learning

a. We found that making a Classification Tree using Entropy proves to be most efficient technique for this dataset. The attributes chosen for classification were generally continuous variables and Prior Default being factor variable has significant contribution.

b. The missing values if not treated properly would create hindrance in the analysis for classification.

V. CONCLUSION

From our analysis, we are able to conclude that the most significant attributes in determining the outcome of a credit application are Income, Years of Employment, Credit Score and whether or not the applicant had defaulted on a prior credit account. This result was consistent over all the techniques applied – visualization, dimension and data reduction, supervised and unsupervised learning. Correspondingly, the other variables in the dataset – Age, Debt, Citizenship, Marital Status, Gender, Education level, Ethnicity, Zip code, Drivers licence and Bank customer – did not have a significant effect on whether an application was approved.

Of the models built to predict credit application outcomes, classification and regression trees provided the most accurate results and had a misclassification rate of only 12.8%. Future work on this and similar datasets could include combination of two or more techniques to produce a classification model with a higher degree of accuracy. Used by creditors, an improved model can greatly reduce the risk of granting credit to potential defaulters.

REFERENCES

1. Bekhet, Hussain Ali, and Shorouq Fathi Kamel Eletter. "Credit Risk Assessment Model For Jordanian Commercial Banks: Neural Scoring Approach". *Review of Development Finance* 4.1 (2014): 20-28. Web.
2. Song, Jungwoo. "A comparison of Classification Methods For Credit Card Approval Using R"
3. Fu,Zhoutong, and Liu Zhedi. "Classifier Comparisons On Credit Approval Prediction".
4. Chitra, Dr. K., Subashini,Mrs. B."Automatic Credit Approval using Classification Method".
5. Aida Krichene, Abdelmoula."Bank credit risk analysis with k-nearestneighbor classifier: Case of Tunisian banks"
6. http://rstudio-pubs-static.s3.amazonaws.com/73039_9946de135c0a49daa7a0a9eda4a67a72.html
7. https://www.r-bloggers.com/classification-on-the-german-credit-database/
8. https://github.com