

# Analysis on Approval or Denial of Loan

## Table of Contents

<b>1. Introduction.....</b>	<b>2</b>
<b>2. Data.....</b>	<b>2</b>
<b>3. Problems to be Solved .....</b>	<b>3</b>
<b>4. Solutions .....</b>	<b>3</b>
<b>5. Experiments and Results.....</b>	<b>4</b>
5.1. Methods and Process .....	4
5.2. Descriptive Statistics.....	6
5.2.1 Bar Graph.....	6
5.2.3. Pie Chart.....	7
5.2.4 Histogram.....	8
5.3. Hypothesis Testing 1.....	9
5.4. Hypothesis Test 2.....	10
5.5. Classification Models .....	11
5.5.1. K- Nearest Neighbor .....	11
5.5.2. Naïve Bayes .....	14
5.5.3 Logistic Regression.....	18
5.5.4. Decision Tree .....	22
5.6. Evaluations and Results .....	24
5.7. Findings .....	24
<b>6. Conclusions and Future Work.....</b>	<b>25</b>
6.1. Conclusions .....	25
6.2. Limitations .....	25
6.3. Potential Improvements or Future Work .....	25

## 1. Introduction

A small-scale business or large-scale business may require a loan at any point of time, starting from creating the initial investment or in order to pay the debts or for growing the business. Loans can be savior for any business. But in order to get this loan the business has to go through a rigorous process which is done by the bank who gives the loan. As the loan is lending the money to a party, a various analysis is done on which the acceptance of the loan and even the amount of the loan can be predicted and further approved. So, in this project we would do the analysis on the data set of the companies from 1947 to 2014, who had applied for the for loans.

The data set consists of the factors on which the approval or denial of the loan depends.

## 2. Data

The Data set is collected from U.S. Small Business Administrator.

Number of Rows: 899164

Number of Columns: 27

The columns defines the various factors on which our final outcome would depend. With all this factor we would do creation of the best fitted model which would help us to reach the final outcome.

Variable Name	Data type	Description of variable
LoanNr_ChkDgt	Text	Identifier – Primary Key
Name	Text	Borrower Name
City	Text	Borrower City
State	Text	Borrower State
Zip	Text	Borrower Zip Code
Bank	Text	Bank Name
BankState	Text	Bank State
NAICS	Text	North American Industry Classification System code
ApprovalDate	Date/Time	Date SBA Commitment Issued
ApprovalFY	Text	Fiscal Year of Commitment
Term	Number	Loan term in months
NoEmp	Number	Number of Business Employees
NewExist	Text	1 = Existing Business, 2 = New Business
CreateJob	Number	Number of jobs created
RetainedJob	Number	Number of jobs retained
FranchiseCode	Text	Franchise Code 00000 or 00001 = No Franchise
UrbanRural	Text	1= Urban, 2= Rural, 0 = Undefined
RevLineCr	Text	Revolving Line of Credit : Y = Yes
LowDoc	Text	LowDoc Loan Program: Y = Yes, N = No

ChgOffDate	Date/Time	The date when a loan is declared to be in default
DisbursementDate	Date/Time	Disbursement Date
DisbursementGross	Currency	Amount Disbursed
BalanceGross	Currency	Gross amount outstanding
MIS_Status	Text	Loan Status
ChgOffPrinGr	Currency	Charged-off Amount
GrAppv	Currency	Gross Amount of Loan Approved by Bank
SBA_Appv	Currency	SBA's Guaranteed Amount of Approved Loan

But after some brain storming is found the below column the most informative and factors that may be most impactful for approval or denial of the loan

		Variable Name	Description of variable
<b>Independent Variable (X)</b>	}	ApprovalFY	Fiscal Year of Commitment
		Term	Loan term in months
		NoEmp	Number of Business Employees
		NewExist	Existence of the business
		CreateJob	Number of jobs created
		RetainedJob	Number of jobs retained
		UrbanRural	Urban or Rural
<b>Dependent Variable (Y)</b>	}	DisbursementGross	Amount Disbursed
		MIS_Status	Loan Status

### 3. Problems to be Solved

1. Identify which explanatory variables may be good “predictors” or risk indicators of the level of risk associated with approval of the loan;
2. Work through the stages in model building and validation;

### 4. Solutions

In this project I would the one sampled and two sampled hypothesis testing along with that I would built the various classification models like the K-Nearest Neighbor, Naïve Bayes, Logistic Regression and Decision Tree. From all this model I would able to get the best factors in order to get the loan approved.

## 5. Experiments and Results

### 5.1. Methods and Process

First Loading up the data into the R

```
Source
Console Terminal R Markdown
~/Downloads/
> mydata=data.frame(LoanNo,Name,City,State,Zip,BankName,BankState,NAICS,AppDate,AppFY,LoanTerm,NumbrEmp,NewExist,JobCreated,JobRetained,FranchiseCode,UrbanRural,RevLineCr,LoxDoc,ChgoffDate,DisbursementDate,DisbursementGross,BalanceGross,MIS_status,ChgOffPrinGr,GrAppv,SBA_Appv)
> summary(mydata)
  LoanNo      Name      City      State
Min.   :1.000e+09 SUBWAY      : 1266 LOS ANGELES: 11547 CA      :130365
1st Qu.:2.592e+09 QUIZNO'S SUBS : 432 HOUSTON    : 10220 TX      : 70339
Median :4.363e+09 COLD STONE CREAMERY: 365 NEW YORK   : 7828 NY      : 57313
Mean   :4.774e+09 QUIZNO'S      : 344 CHICAGO    : 5998 FL      : 41173
3rd Qu.:6.909e+09 DAIRY QUEEN   : 328 MIAMI      : 5592 PA      : 34599
Max.   :9.996e+09 (Other)       :893265 SAN DIEGO  : 5353 OH      : 32474
              NA's      :      5 (Other)   :849467 (Other):529742

  Zip      BankName      BankState      NAICS
Min.   :      0 BANK OF AMERICA NATL ASSOC : 86733 CA      :117933 Min.   :      0
1st Qu.:27614 WELLS FARGO BANK NATL ASSOC : 63401 NC      : 79405 1st Qu.:235210
Median :55416 JPMORGAN CHASE BANK NATL ASSOC: 48091 IL      : 65805 Median :445310
Mean   :53866 U.S. BANK NATIONAL ASSOCIATION: 35086 OH      : 58362 Mean   :398561
3rd Qu.:83706 CITIZENS BANK NATL ASSOC      : 33764 SD      : 51022 3rd Qu.:561730
Max.   :99999 PNC BANK, NATIONAL ASSOCIATION: 27312 TX      : 47712 Max.   :928120
              (Other)      :601618 (Other):475766

  AppDate      AppFY      LoanTerm      NumbrEmp      NewExist
7-Jul-93 : 1129 Min.   :1969 Min.   : 0.0 Min.   : 0.0 Min.   :1.000
30-Jan-04: 1027 1st Qu.:1997 1st Qu.: 60.0 1st Qu.: 2.0 1st Qu.:1.000
8-Jul-93 : 780 Median :2002 Median : 84.0 Median : 4.0 Median :1.000
4-Oct-04 : 658 Mean   :2001 Mean   :110.8 Mean   : 11.4 Mean   :1.282
30-Sep-03: 605 3rd Qu.:2006 3rd Qu.:120.0 3rd Qu.: 10.0 3rd Qu.:2.000

  LoanNo      Name      City      State
Median : 0.000 Median : 1.00 Median : 1 Median :1.0000 Y      :200588
Mean   : 8.419 Mean   : 10.79 Mean   : 2757 Mean : 0.7573 T      :15232
3rd Qu.: 1.000 3rd Qu.: 4.00 3rd Qu.: 1 3rd Qu.:1.0000 : 4517
Max.   :8800.000 Max.   :9500.00 Max.   :99999 Max.   :2.0000 1      : 22
              (Other): 42

  LoxDoc      ChgoffDate      DisbursementDate      DisbursementGross
N      :779969 :733648 31-Jul-95: 10345 Min.   : 4000
Y      :110046 13-Mar-10: 733 30-Apr-95: 10299 1st Qu.: 42399
      : 2575 20-Feb-10: 611 31-Jan-95: 9725 Median : 100000
      : 1489 30-Jan-10: 517 31-Oct-94: 8876 Mean   : 201447
      : 754 6-Feb-10 : 460 31-Oct-95: 8150 3rd Qu.: 238300
      : 602 6-Mar-10 : 420 30-Apr-96: 8066 Max.   :11446325
(Other): 570 (Other) :159616 (Other) :840544

  BalanceGross      MIS_status      ChgOffPrinGr      GrAppv
$0.00 :895991 CHGOFF:157481 $0.00 :734329 $50,000.00 : 69300
$1,760.00 : 1 P I F :738524 $50,000.00 : 2108 $25,000.00 : 51173
$115,820.00 : 1 $10,000.00 : 1865 $100,000.00 : 50905
$12,750.00 : 1 $25,000.00 : 1371 $10,000.00 : 38258
$25,000.00 : 1 $35,000.00 : 1344 $150,000.00 : 27572
$37,100.00 : 1 $100,000.00 : 1028 $20,000.00 : 23374
(Other) : 9 (Other) :153960 (Other) :635423

  SBA_Appv
$25,000.00 : 49528
$12,500.00 : 40079
$5,000.00 : 31032
$50,000.00 : 25030
$10,000.00 : 16979
$17,500.00 : 16135
```

## Creating the data frame

```
Console Terminal x R Markdown x
~/Downloads/ ➔
> #Creating the data frame
> mydata_1 = data.frame(AppFY,LoanTerm,NumbrEmp,NewExist,JobCreated,JobRetained,UrbanRural,DisbursementGross,MIS_status)
> str(mydata_1)
'data.frame': 896005 obs. of 9 variables:
 $ AppFY      : int  1997 1997 1997 1997 1997 1997 1980 1997 1997 1997 ...
 $ LoanTerm   : int  84 60 180 60 240 120 45 84 297 84 ...
 $ NumbrEmp   : int  4 2 7 2 14 19 45 1 2 3 ...
 $ NewExist   : int  2 2 1 1 1 1 2 2 2 2 ...
 $ JobCreated : int  0 0 0 0 7 0 0 0 0 0 ...
 $ JobRetained : int  0 0 0 0 7 0 0 0 0 0 ...
 $ UrbanRural : int  0 0 0 0 0 0 0 0 0 0 ...
 $ DisbursementGross: int  60000 40000 287000 35000 229000 517000 600000 45000 305000 70000 .
 ..
 $ MIS_status  : Factor w/ 2 levels "CHGOFF","P I F": 2 2 2 2 2 2 1 2 2 2 ...
> |
```

## Shuffling the data first

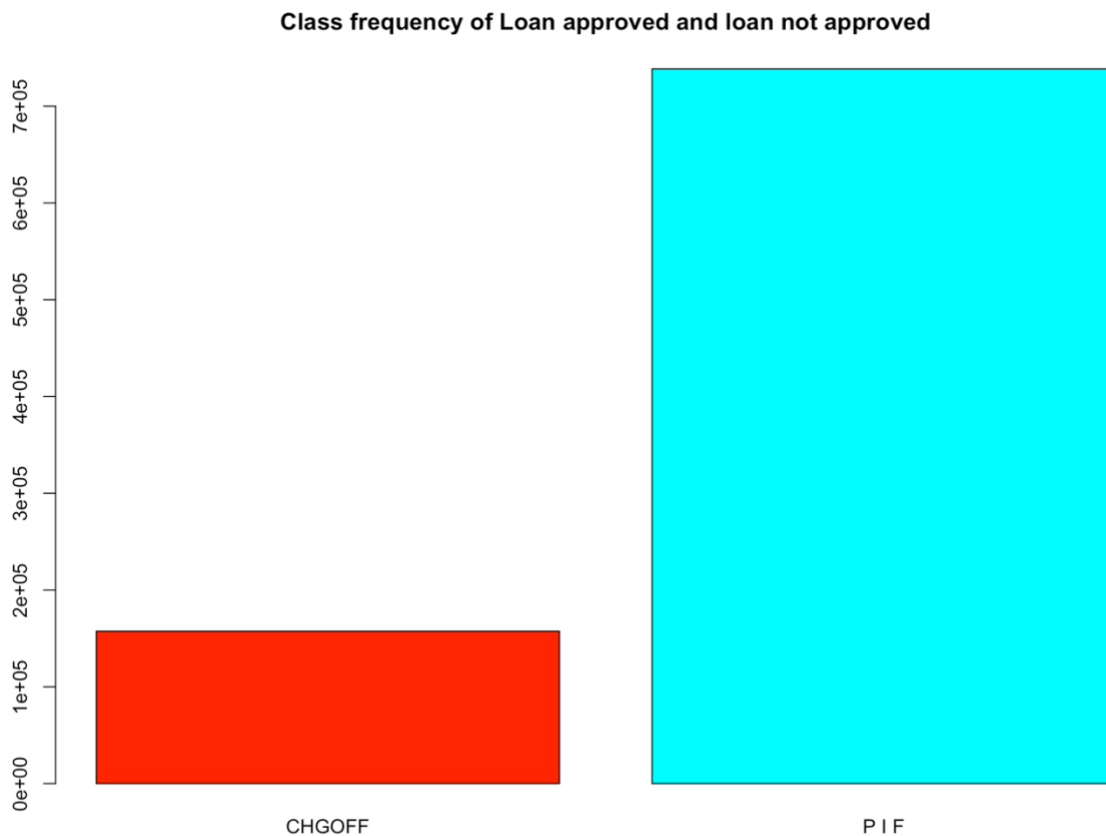
```
Console Terminal x R Markdown x
~/Downloads/ ➔
> head(mydata_1)
  AppFY LoanTerm NumbrEmp NewExist JobCreated JobRetained UrbanRural
237898  2007      84      30        1          0          30         2
333425  2000      42       4        2          0           0         1
513279  2002      90       5        2          0           0         1
813756  2005      94      15        1          0           0         1
180708  2006      46       1        2          0           1         2
804958  2005     125      12        2         12           0         1
  DisbursementGross MIS_status
237898          90157      P I F
333425          16900      P I F
513279          87594      P I F
813756         168000    CHGOFF
180708          15000    CHGOFF
804958         424616      P I F
> |
```

## 5.2. Descriptive Statistics

### 5.2.1 Bar Graph

Here, the bar graph is drawn between the frequency of the CHGOFF values and PIF values present in the data. This gives overall percentage of loan approved since 1947 to 2014.

```
Console Terminal R Markdown
~/Downloads/
> library(plyr)
> opt=count(MIS_status)
> cf=opt$freq
> labels=opt$x
> opt
      x      freq
1 CHGOFF 157481
2 P I F 738524
> barplot(cf,main="Class frequency of Loan approved and loan not approved",names.arg=labels,col = rainbow(length(cf)))
>
```



### 5.2.3. Pie Chart

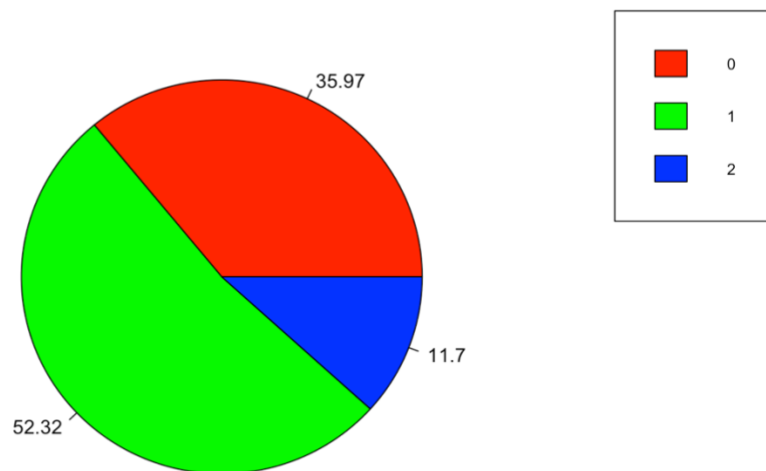
The pie chart describes the percentage division of the companies present in data in undefined, urban and rural areas.

```
Console Terminal R Markdown
~/Downloads/
> #pie chart
> opt1=count(UrbanRural)
> cf1=opt1$freq
> labels1=opt1$x
> crf=table(my_data$UrbanRural)/nrow(my_data)
> piepercent=round(100*crf/sum(crf),2)
> crf

      0      1      2
0.3597279 0.5232236 0.1170485
> pie(crf,label=piepercent,main="Pie Chart for Type of Area", col = rainbow(length(crf)))
> legend("topright", c("0","1","2"),cex=0.8,fill=rainbow(length(crf)))
> piepercent

      0      1      2
35.97 52.32 11.70
>
```

Pie Chart for Type of Area

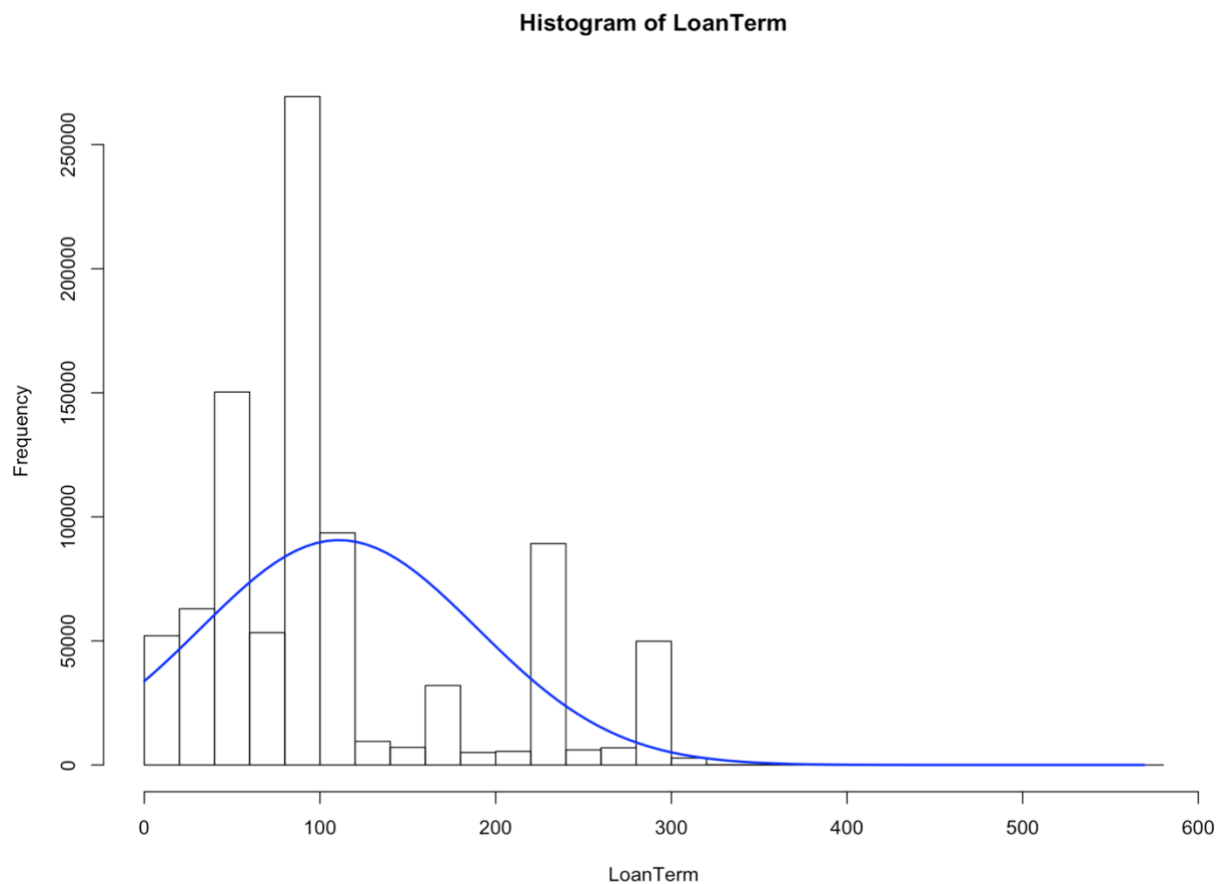


0-Undefined, 1-Urban, 2-Rural

### 5.2.4 Histogram

The histogram here tells different terms in month for which the loan was required by the companies.

```
> #histogram  
> h=hist(LoanTerm)  
> xfit<-seq(min(LoanTerm),max(LoanTerm),length=100)  
> yfit<-dnorm(xfit,mean=mean(LoanTerm),sd=sd(LoanTerm))  
> yfit <- yfit*diff(h$mids[1:2])*length(LoanTerm)  
> lines(xfit, yfit, col="blue", lwd=2)  
>
```





### 5.3. Hypothesis Testing 1

This is a one sampled hypothesis where

$H_0$ : The sample disbursement mean is equal to 216153

$H_1$ : The sample disbursement mean is greater than 216153

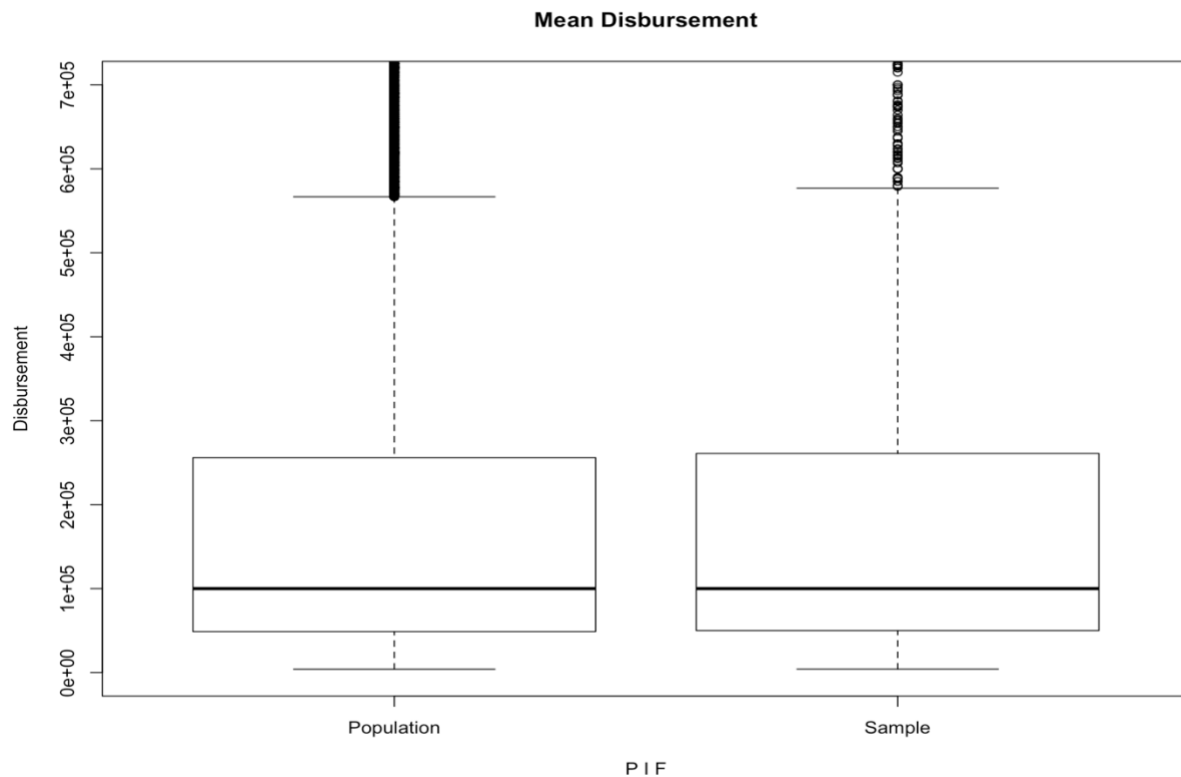
Confidence interval: 95%

```
> z.test(ne_sample$DisbursementGross, NULL, alternative = "greater", mu=216153, sigma.x=sd(ne_sample$DisbursementGross), conf.level = 0.95)
```

One-sample z-Test

```
data: ne_sample$DisbursementGross
z = 1.0018, p-value = 0.1582
alternative hypothesis: true mean is greater than 216153
95 percent confidence interval:
 212162      NA
sample estimates:
mean of x
 222371.1
```

Since the p-value is greater than 0.05, we accept the null hypothesis that the sample distribution mean is equal to 216153 at 95% confidence level interval



With the help of the box plot, we can visualize the null hypothesis being accepted.

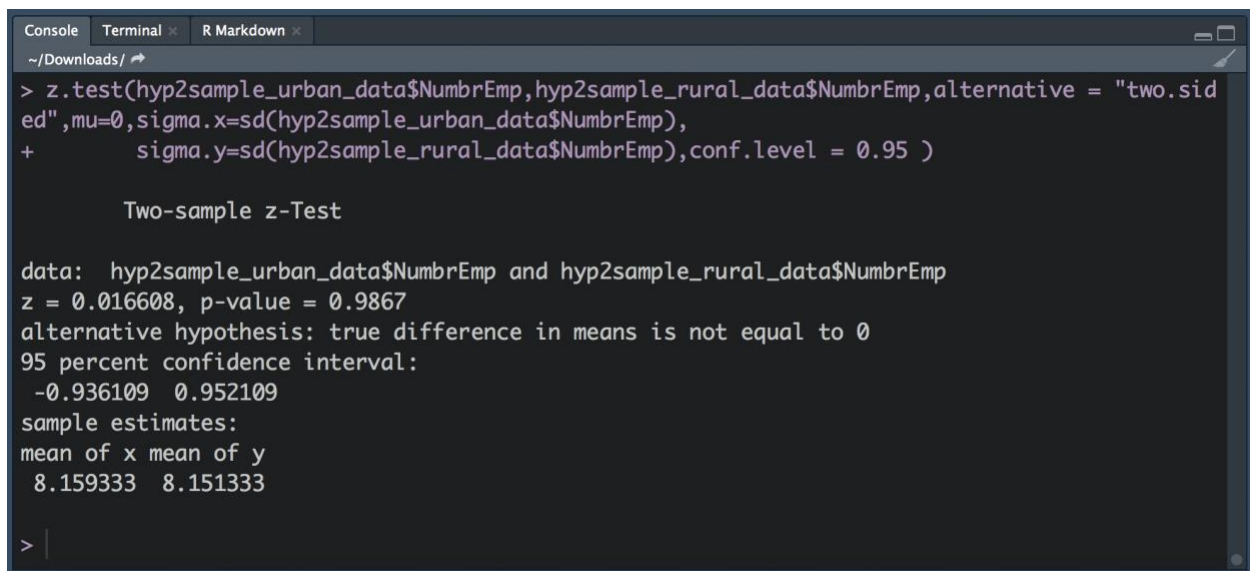
## 5.4. Hypothesis Test 2

This is a Two sampled hypothesis testing in which one of the samples is the number of employees in urban area and the second is the number of employees in rural area. Here,

$H_0$ : The difference between the mean of number of employees in urban and in rural is 0

$H_1$ : The difference between the mean of number of employees in urban and in rural is not 0

Confidence interval: 95%



```
Console Terminal R Markdown
~/Downloads/
> z.test(hyp2sample_urban_data$NumbrEmp, hyp2sample_rural_data$NumbrEmp, alternative = "two.sided", mu=0, sigma.x=sd(hyp2sample_urban_data$NumbrEmp),
+       sigma.y=sd(hyp2sample_rural_data$NumbrEmp), conf.level = 0.95 )

      Two-sample z-Test

data:  hyp2sample_urban_data$NumbrEmp and hyp2sample_rural_data$NumbrEmp
z = 0.016608, p-value = 0.9867
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.936109  0.952109
sample estimates:
mean of x mean of y
 8.159333  8.151333

>
```

Since, the p-value is greater than 0.05, we accept the null hypothesis that the difference between the mean number of employees in urban and in rural is 0 at a confidence interval of 95%.

## 5.5. Classification Models

### 5.5.1. K- Nearest Neighbor

Predicting the model using the K-Nearest Neighbor

--> First the data is pre-processed

-> The X variables are converted to numeric

-> The Y variable is kept as categorical or factored variable

-> The whole data is brought into same scale

--> The data is then divided into testing and training

--> The Model is built with different K values (here: 200,300,499) on training data

--> Accuracy is found for all the models on the testing data

Pre-processing the data

```
Console Terminal R Markdown
~/Downloads/
> #Preprocessing for KNN model
> knn_data=mydata_1
> knn_data$NewExist=as.character(knn_data$NewExist)
> library(plyr)
> knn_data$NewExist=revalue(knn_data$NewExist, c("1"= "0"))
> knn_data$NewExist=revalue(knn_data$NewExist, c("2"= "1"))
> knn_data$NewExist=as.integer(knn_data$NewExist)
> library(dummies)
> knn_data = dummy.data.frame(knn_data,names="UrbanRural")
> knn_data$MIS_status=revalue(knn_data$MIS_status, c("CHGOFF"= "0"))
> knn_data$MIS_status=revalue(knn_data$MIS_status, c("P I F"= "1"))
> str(knn_data)
'data.frame': 896005 obs. of 11 variables:
 $ AppFY : int 2007 2000 2002 2005 2006 2005 1996 2003 1993 2005 ...
 $ LoanTerm : int 84 42 90 94 46 125 84 84 120 71 ...
 $ NumbrEmp : int 30 4 5 15 1 12 8 7 10 3 ...
 $ NewExist : int 0 1 1 0 1 1 0 1 0 0 ...
 $ JobCreated : int 0 0 0 0 0 12 0 0 0 0 ...
 $ JobRetained : int 30 0 0 0 1 0 0 0 0 3 ...
 $ UrbanRural0 : int 0 0 0 0 0 0 1 0 1 0 ...
 $ UrbanRural1 : int 0 1 1 1 0 1 0 1 0 1 ...
 $ UrbanRural2 : int 1 0 0 0 1 0 0 0 0 0 ...
 $ DisbursementGross: int 90157 16900 87594 168000 15000 424616 30000 144000 222500 25000 ..
 $ MIS_status : Factor w/ 2 levels "0","1": 2 2 2 1 1 2 2 2 1 ...
- attr(*, "dummies")=List of 1
 $ UrbanRural: int 7 9 0
```

## Scaling the data

```
Console Terminal R Markdown
~/Downloads/
> #Scaling the preprocessed knn data
> #extracting numerical data
> num.vars.knn <- sapply(knn_data, is.numeric)
> #normalizing the knn data
> knn_data[num.vars.knn] = lapply(knn_data[num.vars.knn], scale)
> head(knn_data)
      AppFY  LoanTerm  NumbrEmp  NewExist  JobCreated  JobRetained  UrbanRural0
237898  0.9910201 -0.3399390  0.252576382 -0.6265058 -0.03559242  0.08106884 -0.7495568
333425 -0.1930129 -0.8723731 -0.100506890  1.5961525 -0.03559242 -0.04551841 -0.7495568
513279  0.1452822 -0.2638770 -0.086926764  1.5961525 -0.03559242 -0.04551841 -0.7495568
813756  0.6527249 -0.2131690  0.048874494 -0.6265058 -0.03559242 -0.04551841 -0.7495568
180708  0.8218725 -0.8216651 -0.141247268  1.5961525 -0.03559242 -0.04129883 -0.7495568
804958  0.6527249  0.1798180  0.008134117  1.5961525  0.01514011 -0.04551841 -0.7495568
      UrbanRural1 UrbanRural2 DisbursementGross MIS_status
237898 -1.0475773  2.7465364      -0.3869226          1
333425  0.9545824 -0.3640945      -0.6416164          1
513279  0.9545824 -0.3640945      -0.3958335          1
813756  0.9545824 -0.3640945      -0.1162846          0
180708 -1.0475773  2.7465364      -0.6482222          0
804958  0.9545824 -0.3640945       0.7758963          1
>
```

## Dividing the pre-processed data into training and testing by setting index

```
Console Terminal R Markdown
~/Downloads/
> #dividing the pre-processed data into training and testing
> set.seed(1)
> train=1:716804
> knn.train.data <- knn_data[train,]      #train data
> knn.tst.data = knn_data[-train,]      #test data
> str(knn.train.data)
'data.frame': 716804 obs. of 11 variables:
 $ AppFY      : num [1:716804, 1] 0.991 -0.193 0.145 0.653 0.822 ...
 $ LoanTerm   : num [1:716804, 1] -0.34 -0.872 -0.264 -0.213 -0.822 ...
 $ NumbrEmp   : num [1:716804, 1] 0.2526 -0.1005 -0.0869 0.0489 -0.1412 ...
 $ NewExist   : num [1:716804, 1] -0.627 1.596 1.596 -0.627 1.596 ...
 $ JobCreated : num [1:716804, 1] -0.0356 -0.0356 -0.0356 -0.0356 -0.0356 ...
 $ JobRetained: num [1:716804, 1] 0.0811 -0.0455 -0.0455 -0.0455 -0.0413 ...
 $ UrbanRural0: num [1:716804, 1] -0.75 -0.75 -0.75 -0.75 -0.75 ...
 $ UrbanRural1: num [1:716804, 1] -1.048 0.955 0.955 0.955 -1.048 ...
 $ UrbanRural2: num [1:716804, 1] 2.747 -0.364 -0.364 -0.364 2.747 ...
 $ DisbursementGross: num [1:716804, 1] -0.387 -0.642 -0.396 -0.116 -0.648 ...
 $ MIS_status : Factor w/ 2 levels "0","1": 2 2 2 1 1 2 2 2 1 ...
 - attr(*, "dummies")=List of 1
 ..$ UrbanRural: int 7 8 9
> str(knn.tst.data)
'data.frame': 179201 obs. of 11 variables:
 $ AppFY      : num [1:179201, 1] 0.484 1.16 0.653 -1.884 -1.208 ...
 $ LoanTerm   : num [1:179201, 1] -0.34 -0.758 0.75 1.638 1.448 ...
 $ NumbrEmp   : num [1:179201, 1] -0.141 -0.128 -0.114 -0.114 -0.128 ...
 $ NewExist   : num [1:179201, 1] -0.627 1.596 1.596 -0.627 1.596 ...
 $ JobCreated : num [1:179201, 1] -0.0356 -0.0187 -0.0356 -0.0356 -0.0356 ...
 $ JobRetained: num [1:179201, 1] -0.0413 -0.0371 -0.0455 -0.0455 -0.0455 ...
```

For building the models with different Ks

```
> knn.200 = knn(knn.train.data,knn.tst.data,knn.train.MIS,k=200)
> #finding the accuracy of the models built on testing data
> accuracy(knn.tst.MIS, knn.200)
[1] 0.8252856
> knn.300 = knn(knn.train.data,knn.tst.data,knn.train.MIS,k=300)
> accuracy(knn.tst.MIS, knn.300)
[1] 0.8252856
> accuracy(knn.tst.MIS, knn.499)
Error in mean(actual != predicted) : object 'knn.499' not found
> knn.499 = knn(knn.train.data,knn.tst.data,knn.train.MIS,k=499)
> accuracy(knn.tst.MIS, knn.499)
[1] 0.8252856
```

The Accuracy at K= 200: 82.52%

The Accuracy at K= 300: 82.52%

The Accuracy at K= 499: 82.52%



### 5.5.2. Naïve Bayes

Predicting the model using the Naïve Bayes Classifier

→ First the data is pre-processed

-> The X variables are converted to categorical

-> The Y variable is kept as categorical or factored variable

→ The data is then divided into testing and training

→ The Model is built with on training data

→ Accuracy is found for all the models on the testing data

Pre-processing the data

```
Console Terminal R Markdown
~/Downloads/
> #Naive Baye's
> #shuffled raw data
> naive_data=mydata_1
> #pre-processing the data
> naive_data[,1] = cut(naive_data[,1], breaks = c(1968,2007,2009,2014),
+                       labels = c("non-recession1", "recession", "non-recession2"))
> naive_data[,2] = cut(naive_data[,2], breaks = c(-1,60,84,120,569),
+                       labels = c("very-short term", "short-term", "long-term", "very-long term"))
> naive_data[,3] = cut(naive_data[,3], breaks = c(-1,2500,7500,9999),
+                       labels = c("small-sized", "medium-sized", "large-sized"))
> naive_data[,4] = factor(naive_data[,4])
> naive_data$JobCreated = cut(naive_data$JobCreated,3)
> naive_data[,6] = cut(naive_data[,6],3)
> naive_data[,7] = factor(naive_data[,7])
> naive_data[,8] = cut(naive_data[,8], breaks = c(3999,42400,10000,238390,11446325),
+                       labels = c("very-low-requirement", "low-requirement", "medium-requirement", "high-requirement"))
> str(naive_data)
'data.frame': 896005 obs. of 9 variables:
 $ AppFY      : Factor w/ 3 levels "non-recession1",...: 1 1 1 1 1 1 1 1 1 1 ...
 $ LoanTerm   : Factor w/ 4 levels "very-short term",...: 2 1 3 3 1 4 2 2 3 2 ...
 $ NumbrEmp   : Factor w/ 3 levels "small-sized",...: 1 1 1 1 1 1 1 1 1 1 ...
 $ NewExist   : Factor w/ 2 levels "1", "2": 1 2 2 1 2 2 1 2 1 1 ...
 $ JobCreated : Factor w/ 3 levels "(-8.8,2.93e+03]",...: 1 1 1 1 1 1 1 1 1 1 ...
 $ JobRetained : Factor w/ 3 levels "(-9.5,3.17e+03]",...: 1 1 1 1 1 1 1 1 1 1 ...
 $ UrbanRural : Factor w/ 3 levels "0", "1", "2": 3 2 2 2 3 2 1 2 1 2 ...
 $ DisbursementGross: Factor w/ 4 levels "very-low-requirement",...: 3 2 3 3 2 4 2 3 3 2 ...
 $ MIS_status : Factor w/ 2 levels "CHGOFF", "P I F": 2 2 2 1 1 2 2 2 2 1 ...
>
```

Dividing the pre-processed data into training and testing by setting index

```
Source
Console Terminal R Markdown
~/Downloads/
$ MIS_status : Factor w/ 2 levels "CHGOFF", "P I F": 2 2 2 1 1 2 2 2 1 ...
> #dividing the pre-processed data into training and testing
> set.seed(1)
> naive.train.data <- naive_data[train,]      #train data
> naive.tst.data = naive_data[-train,]        #test data
> str(naive.train.data)
'data.frame': 716804 obs. of 9 variables:
 $ AppFY      : Factor w/ 3 levels "non-recession1",...: 1 1 1 1 1 1 1 1 1 ...
 $ LoanTerm   : Factor w/ 4 levels "very-short term",...: 2 1 3 3 1 4 2 2 3 2 ...
 $ NumbrEmp   : Factor w/ 3 levels "small-sized",...: 1 1 1 1 1 1 1 1 1 ...
 $ NewExist   : Factor w/ 2 levels "1", "2": 1 2 2 1 2 2 1 2 1 1 ...
 $ JobCreated : Factor w/ 3 levels "(-8.8,2.93e+03]",...: 1 1 1 1 1 1 1 1 1 ...
 $ JobRetained: Factor w/ 3 levels "(-9.5,3.17e+03]",...: 1 1 1 1 1 1 1 1 1 ...
 $ UrbanRural : Factor w/ 3 levels "0", "1", "2": 3 2 2 2 3 2 1 2 1 2 ...
 $ DisbursementGross: Factor w/ 4 levels "very-low-requirment",...: 3 2 3 3 2 4 2 3 3 2 ...
 $ MIS_status : Factor w/ 2 levels "CHGOFF", "P I F": 2 2 2 1 1 2 2 2 1 ...
> str(naive.tst.data)
'data.frame': 179201 obs. of 9 variables:
 $ AppFY      : Factor w/ 3 levels "non-recession1",...: 1 2 1 1 1 1 1 1 1 ...
 $ LoanTerm   : Factor w/ 4 levels "very-short term",...: 2 1 4 4 4 2 4 2 3 4 ...
 $ NumbrEmp   : Factor w/ 3 levels "small-sized",...: 1 1 1 1 1 1 1 1 1 ...
 $ NewExist   : Factor w/ 2 levels "1", "2": 1 2 2 1 2 2 2 2 2 1 ...
 $ JobCreated : Factor w/ 3 levels "(-8.8,2.93e+03]",...: 1 1 1 1 1 1 1 1 1 ...
 $ JobRetained: Factor w/ 3 levels "(-9.5,3.17e+03]",...: 1 1 1 1 1 1 1 1 1 ...
 $ UrbanRural : Factor w/ 3 levels "0", "1", "2": 2 3 2 1 1 1 2 1 1 2 ...
 $ DisbursementGross: Factor w/ 4 levels "very-low-requirment",...: 3 3 2 3 4 3 4 3 4 3 ...
 $ MIS_status : Factor w/ 2 levels "CHGOFF", "P I F": 2 1 2 2 2 1 2 2 2 2 ...
> |
```

## Creating the model on the training data set

```
Source
Console Terminal R Markdown
~/Downloads/
> library(naivebayes)
> nb_model=naive_bayes(MIS_status ~ .,naive.train.data)
> nb_model

===== Naive Bayes =====
=
Call:
naive_bayes.formula(formula = MIS_status ~ ., data = naive.train.data)

A priori probabilities:

      CHGOFF      P I F
0.1758542 0.8241458

Tables:

AppFY      CHGOFF      P I F
non-recession1 0.84763552 0.90162183
recession      0.12850944 0.05193897
non-recession2 0.02385505 0.04643919

LoanTerm      CHGOFF      P I F
very-short term 0.64614091 0.22103221
short-term      0.20142321 0.35009843
long-term       0.09158846 0.15122107
very-long term  0.06084742 0.27764828

NumbreEmp      CHGOFF      P I F
small-sized 9.999207e-01 9.998510e-01
medium-sized 5.553220e-05 1.252643e-04
large-sized 2.379951e-05 2.369865e-05

NewExist      CHGOFF      P I F
1 0.6992694 0.7224871
2 0.3007306 0.2775129

JobCreated      CHGOFF      P I F
(-8.8,2.93e+03] 9.985641e-01 9.994177e-01
(2.93e+03,5.87e+03] 0.000000e+00 1.354208e-05
(5.87e+03,8.81e+03] 1.435904e-03 5.687676e-04

# ... and 3 more tables
>
```



## Evaluating the accuracy on testing data

```
> head(predict(nb_model, naive.tst.data, type = "prob"))
      CHGOFF      P I F
[1,] 0.147893373 0.8521066
[2,] 0.651007390 0.3489926
[3,] 0.092508537 0.9074915
[4,] 0.015353258 0.9846467
[5,] 0.008609267 0.9913907
[6,] 0.043823834 0.9561762
> library(Metrics)
> accuracy(naive.tst.data[,9], pred)
[1] 0.8237063
>
```

### 5.5.3 Logistic Regression

Predicting the model using the Logistic Classifier

→ First the data is pre-processed

-> The X variables are converted to numerical

-> The Y variable is kept as categorical or factored variable

→ The data is then divided into testing and training

→ The Model is built with on training data

→ Accuracy is found for all the models on the testing data

Pre-processing the data

```
> #logistic regression
> log_data=mydata_1
> #pre-processing the data
> log_data=dummy.data.frame(log_data, names = c("NewExist","UrbanRural"))
> log_data=dummy.data.frame(log_data, names = c("NewExist","UrbanRural"))
> log_data$MIS_status=revalue(log_data$MIS_status, c("CHGOFF"= "0"))
> log_data$MIS_status=revalue(log_data$MIS_status, c("P I F"= "1"))
> str(log_data)
'data.frame': 896005 obs. of 12 variables:
 $ AppFY      : int  2007 2000 2002 2005 2006 2005 1996 2003 1993 2005 ...
 $ LoanTerm   : int  84 42 90 94 46 125 84 84 120 71 ...
 $ NumbrEmp   : int  30 4 5 15 1 12 8 7 10 3 ...
 $ NewExist1  : int  1 0 0 1 0 0 1 0 1 1 ...
 $ NewExist2  : int  0 1 1 0 1 1 0 1 0 0 ...
 $ JobCreated : int  0 0 0 0 0 12 0 0 0 0 ...
 $ JobRetained: int  30 0 0 0 1 0 0 0 0 3 ...
 $ UrbanRural0: int  0 0 0 0 0 0 1 0 1 0 ...
 $ UrbanRural1: int  0 1 1 1 0 1 0 1 0 1 ...
 $ UrbanRural2: int  1 0 0 0 1 0 0 0 0 0 ...
 $ DisbursementGross: int  90157 16900 87594 168000 15000 424616 30000 144000 222500 25000 ..
 .
 $ MIS_status : Factor w/ 2 levels "0","1": 2 2 2 1 1 2 2 2 2 1 ...
 - attr(*, "dummies")= list()
>
```

## Dividing the data into training and testing data

```
Console Terminal R Markdown
~/Downloads/

> #dividing the pre-processed data into training and testing
> set.seed(1)
> log.train.data <- log_data[train,]          #train data
> log.tst.data = log_data[-train,]           #test data
> str(log.train.data)
'data.frame':  716804 obs. of  12 variables:
 $ AppFY      : int  2007 2000 2002 2005 2006 2005 1996 2003 1993 2005 ...
 $ LoanTerm    : int  84 42 90 94 46 125 84 84 120 71 ...
 $ NumbrEmp    : int  30 4 5 15 1 12 8 7 10 3 ...
 $ NewExist1   : int  1 0 0 1 0 0 1 0 1 1 ...
 $ NewExist2   : int  0 1 1 0 1 1 0 1 0 0 ...
 $ JobCreated  : int  0 0 0 0 0 12 0 0 0 0 ...
 $ JobRetained : int  30 0 0 0 1 0 0 0 0 3 ...
 $ UrbanRural0 : int  0 0 0 0 0 0 1 0 1 0 ...
 $ UrbanRural1 : int  0 1 1 1 0 1 0 1 0 1 ...
 $ UrbanRural2 : int  1 0 0 0 1 0 0 0 0 0 ...
 $ DisbursementGross: int  90157 16900 87594 168000 15000 424616 30000 144000 222500 25000 ..
.
 $ MIS_status  : Factor w/ 2 levels "0","1": 2 2 2 1 1 2 2 2 2 1 ...
- attr(*, "dummies")= list()
> str(log.tst.data)
'data.frame':  179201 obs. of  12 variables:
 $ AppFY      : int  2004 2008 2005 1990 1994 1979 2002 1998 1993 2001 ...
 $ LoanTerm    : int  84 51 170 240 225 80 183 84 120 180 ...
 $ NumbrEmp    : int  1 2 3 3 2 35 10 6 30 3 ...
 $ NewExist1   : int  1 0 0 1 0 0 0 0 0 1 ...
 $ NewExist2   : int  0 1 1 0 1 1 1 1 1 0 ...
 $ JobCreated  : int  0 4 0 0 0 0 0 0 15 0 ...
```

Building the model using Step wise backward and stepwise both feature selection

```
Source
Console Terminal R Markdown
~/Downloads/
> #stepwise backward elimination
> m1 = step(full, direction = "backward", trace=T)
Start: AIC=536882.3
MIS_status ~ AppFY + LoanTerm + NumbrEmp + NewExist1 + NewExist2 +
  JobCreated + JobRetained + UrbanRural0 + UrbanRural1 + UrbanRural2 +
  DisbursementGross

Step: AIC=536882.3
MIS_status ~ AppFY + LoanTerm + NumbrEmp + NewExist1 + NewExist2 +
  JobCreated + JobRetained + UrbanRural0 + UrbanRural1 + DisbursementGross

Step: AIC=536882.3
MIS_status ~ AppFY + LoanTerm + NumbrEmp + NewExist1 + JobCreated +
  JobRetained + UrbanRural0 + UrbanRural1 + DisbursementGross

      Df Deviance    AIC
<none>      536862 536882
- JobRetained    1  536907 536925
- JobCreated     1  536940 536958
- AppFY          1  536966 536984
- NewExist1      1  537088 537106
- NumbrEmp       1  537644 537662
- UrbanRural1    1  537971 537989
- DisbursementGross 1  538070 538088
- UrbanRural0    1  538281 538299
- LoanTerm       1  618434 618452
```

```
Console Terminal R Markdown
~/Downloads/
> #stepwise both
> m2 = step(base,scope=list(upper=full, lower=~1), direction="both",trace=T)
Start: AIC=654825.7
MIS_status ~ DisbursementGross

      Df Deviance   AIC
+ LoanTerm      1  556606 556612
+ UrbanRural0    1  623062 623068
+ AppFY          1  624629 624635
+ UrbanRural1    1  629867 629873
+ NumbrEmp       1  652819 652825
+ JobRetained    1  654708 654714
+ JobCreated     1  654719 654725
+ UrbanRural2    1  654746 654752
+ NewExist1      1  654760 654766
+ NewExist2      1  654760 654766
<none>          654822 654826
- DisbursementGross 1  666697 666699

Step: AIC=556611.7
MIS_status ~ DisbursementGross + LoanTerm

      Df Deviance   AIC
+ UrbanRural0    1  540250 540258
+ UrbanRural1    1  543140 543148
+ AppFY          1  544721 544729
+ NumbrEmp       1  553697 553705
+ JobCreated     1  556134 556142
```

Evaluating the accuracy from the built model, on testing data

```
Console Terminal R Markdown
~/Downloads/
> #accuracy of the model built stepwise backward elimination
> predict(m1, type="response", newdata = log.tst.data)
728681 301447 842177 363666 612327 885021 574713 104787 577671
0.7896916 0.6851742 0.9653859 0.9980562 0.9961322 0.9313318 0.9730202 0.9060290 0.9542696
417193 738408 188886 772372 355083 798644 773313 329283 593215
0.9748247 0.6363417 0.9107818 0.9247818 0.8828600 0.9265232 0.5034838 0.9096678 0.9350523
804895 6071 327121 593583 861817 376248 682371 600925 71378
0.7199917 0.8070550 0.9961934 0.9979384 0.9981911 0.7679044 0.9992800 0.9663982 0.9241337
480336 369347 293003 616685 511848 330204 83353 367738 843005
0.5835051 0.9982130 0.7942671 0.9979225 0.9983760 0.9708874 0.3557181 0.7073985 0.9992636
169520 460851 807116 589480 410142 749127 715791 493195 163201
0.8704741 0.9920700 0.8979081 0.9730017 0.9402468 0.9163382 0.6654706 0.9159298 0.8040284
81077 146733 317157 629261 886471 349024 599672 39717 568717
0.9730954 0.7786693 0.5233934 0.7890969 0.8599462 0.9265038 0.8919144 0.8975737 0.6602562
693781 274204 575639 127267 811106 224229 549005 336254 471505
0.7793836 0.5640858 0.9193494 0.8028245 0.9980577 0.6119973 0.9965617 0.7899614 0.9737902
383859 189619 289890 814449 442498 356073 839278 236500 663521
0.9976358 0.7988605 0.7824101 0.8871543 0.9909096 0.9266328 0.8732773 0.1159151 0.9609691
227877 774393 859035 852602 412011 716195 233112 155661 670352
0.7806704 0.8876991 0.9546603 0.9652563 0.8101046 0.9344260 0.7278260 0.7998073 0.8178499
227108 686903 667811 241552 796932 607242 90979 571311 640899
0.7764496 0.8016275 0.8237563 0.7780392 0.9976275 0.6098806 0.9651068 0.8907215 0.9895693
97134 686884 695873 427698 730652 195545 13634 658787 396524
0.7769539 0.5839777 0.7293566 0.9994142 0.7968936 0.9981035 0.6737933 0.9891459 0.8601824
722650 807515 31890 287987 575193 436293 99355 165412 687537
```



```
Console Terminal R Markdown
~/Downloads/
Error in mean(actual != predicted) : object 'log.test.MIS' not found
> accuracy(log.tst.MIS,prob1)      #accuracy of model 1
Error in mean(actual != predicted) : object 'log.tst.MIS' not found
> #dividing the label into testing and training
> log.train.MIS=log.train.data$MIS_status
> log.tst.MIS=log.tst.data$MIS_status
> accuracy(log.tst.MIS,prob1)      #accuracy of model 1
[1] 0.8374563
```

#### 5.5.4. Decision Tree

- No need to perform the pre-processing
- Split the data into testing and training
- Built the decision tree on training data set
- Evaluate the accuracy with the model built on testing data

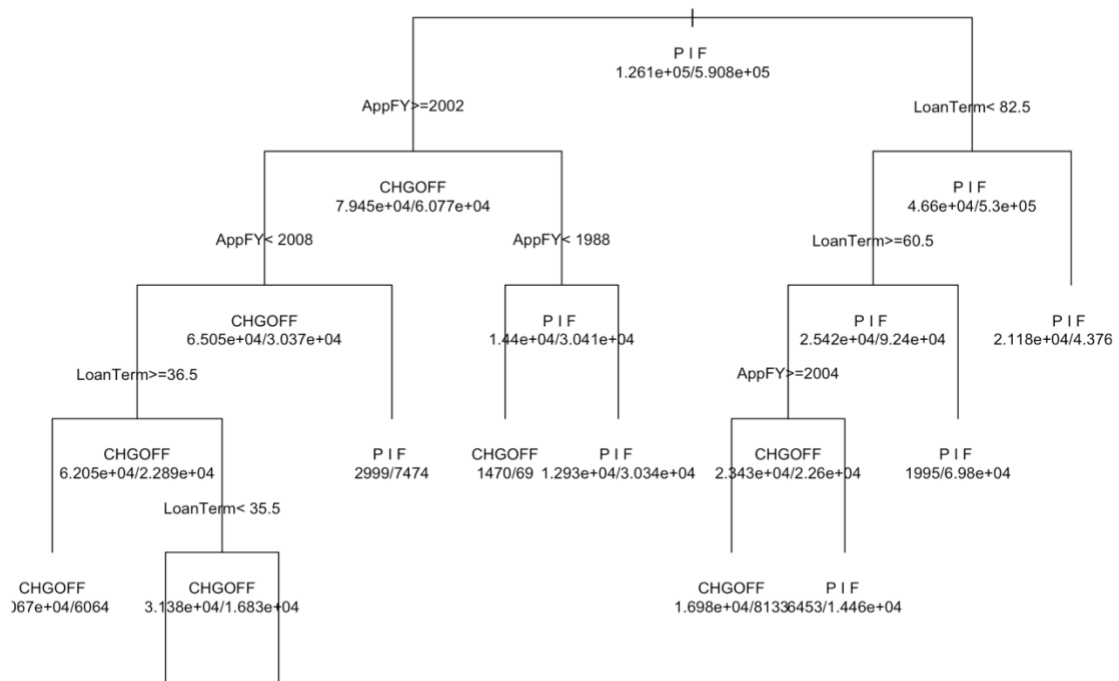
```
Console Terminal R Markdown
~/Downloads/
> #decision tree
> dec_data=mydata_1
> #dividing the pre-processed data into training and testing
> set.seed(1)
> dec.train.data <- dec_data[train,]      #train data
> dec.tst.data = dec_data[-train,]      #test data
> str(dec.train.data)
'data.frame':  716804 obs. of  9 variables:
 $ AppFY      : int  2007 2000 2002 2005 2006 2005 1996 2003 1993 2005 ...
 $ LoanTerm   : int  84 42 90 94 46 125 84 84 120 71 ...
 $ NumbrEmp   : int  30 4 5 15 1 12 8 7 10 3 ...
 $ NewExist   : int  1 2 2 1 2 2 1 2 1 1 ...
 $ JobCreated : int  0 0 0 0 0 12 0 0 0 0 ...
 $ JobRetained: int  30 0 0 0 1 0 0 0 0 3 ...
 $ UrbanRural : int  2 1 1 1 2 1 0 1 0 1 ...
 $ DisbursementGross: int  90157 16900 87594 168000 15000 424616 30000 144000 222500 25000 ..
.
 $ MIS_status : Factor w/ 2 levels "CHGOFF","P I F": 2 2 2 1 1 2 2 2 2 1 ...
> str(dec.tst.data)
'data.frame':  179201 obs. of  9 variables:
 $ AppFY      : int  2004 2008 2005 1990 1994 1979 2002 1998 1993 2001 ...
 $ LoanTerm   : int  84 51 170 240 225 80 183 84 120 180 ...
 $ NumbrEmp   : int  1 2 3 3 2 35 10 6 30 3 ...
 $ NewExist   : int  1 2 2 1 2 2 2 2 2 1 ...
 $ JobCreated : int  0 4 0 0 0 0 0 0 15 0 ...
 $ JobRetained: int  1 2 0 0 0 0 0 0 15 3 ...
 $ UrbanRural : int  1 2 1 0 0 0 1 0 0 1 ...
 $ DisbursementGross: int  180916 55500 37000 115000 395000 200000 275000 230000 750000 18900
```

```

> #dividing the label in training and testing
> dec.train.MIS=dec.train.data$MIS_status
> dec.tst.MIS=dec.tst.data$MIS_status
> #building the decision tree model
> dec = rpart(MIS_status~.,method="class", data=dec.train.data)
> #plotting the decision tree
> plot(dec, uniform=TRUE, main="Classification Tree")
> text(dec, use.n=TRUE, all=TRUE, cex=.8)
>

```

Classification Tree



Decision Tree

## Evaluating the accuracy of the decision tree model built on the testing data

```
Console Terminal R Markdown
~/Downloads/
> #evaluating the accuracy of the decision tree on testing data
> library(caret)
> pred_dec <- predict(dec, newdata=dec.tst.data, type="class")
> confusionMatrix(pred_dec, dec.tst.MIS)
Confusion Matrix and Statistics

              Reference
Prediction CHGOFF P I F
CHGOFF    19754  6218
P I F     11674 141555

      Accuracy : 0.9002
      95% CI : (0.8988, 0.9015)
No Information Rate : 0.8246
P-Value [Acc > NIR] : < 2.2e-16

      Kappa : 0.6295

McNemar's Test P-Value : < 2.2e-16

Sensitivity : 0.6285
Specificity : 0.9579
Pos Pred Value : 0.7606
Neg Pred Value : 0.9238
Prevalence : 0.1754
Detection Rate : 0.1102
Detection Prevalence : 0.1449
Balanced Accuracy : 0.7932
```

## 5.6. Evaluations and Results

Classification Model	Accuracy
K-nearest neighbor	82.52%
Naïve Bayes	82.37%
Logistic Regression	83.74%
Decision Tree	90.02%

## 5.7. Findings

On comparing all the above classification models Decision Tree gives the best accuracy of 90.02%.



## 6. Conclusions and Future Work

### 6.1. Conclusions

From the above analysis I can conclude that with the help Decision Tree you can predict the best fitted model in order to predict best factors that are most responsible for the approval of the loan.

From this prediction the analysis team will get to know which factors can lead to approval of the loan thus making their task easy in order to predict the approval or denial of the loan for any future application.

### 6.2. Limitations

The limitation which I found performing the analysis was while the K-Nearest classification. R Studio supports only a maximum of 499 K values. So, since the data selected here is too large, I could have used more K values in order to get some better accuracy.

### 6.3. Potential Improvements or Future Work

As for the future work I would test for the overfitting issue in the decision tree, and if found I would do the pruning to solve that issue.

Also, I would perform the random forest on this to know some of the more accuracies.

Also, I would perform linear regression on the approved amount by the bank to the company, in order to predict the dependency of approved amount on various independent factors.

Variable Name	Data type	Description of variable
LoanNr_ChkDgt	Text	Identifier – Primary Key
Name	Text	Borrower Name
City	Text	Borrower City
State	Text	Borrower State
Zip	Text	Borrower Zip Code
Bank	Text	Bank Name
BankState	Text	Bank State
NAICS	Text	North American Industry Classification System code
ApprovalDate	Date/Time	Date SBA Commitment Issued
ApprovalFY	Text	Fiscal Year of Commitment
Term	Number	Loan term in months
NoEmp	Number	Number of Business Employees
NewExist	Text	1 = Existing Business, 2 = New Business
CreateJob	Number	Number of jobs created
RetainedJob	Number	Number of jobs retained
FranchiseCode	Text	Franchise Code 00000 or 00001 = No Franchise
UrbanRural	Text	1= Urban, 2= Rural, 0 = Undefined
RevLineCr	Text	Revolving Line of Credit : Y = Yes
LowDoc	Text	LowDoc Loan Program: Y = Yes, N = No
ChgOffDate	Date/Time	The date when a loan is declared to be in default
DisbursementDate	Date/Time	Disbursement Date
DisbursementGross	Currency	Amount Disbursed
BalanceGross	Currency	Gross amount outstanding
MIS_Status	Text	Loan Status
ChgOffPrinGr	Currency	Charged-off Amount
GrAppv	Currency	Gross Amount of Loan Approved by Bank
SBA_Appv	Currency	SBA's Guaranteed Amount of Approved Loan