**Final Project on Loan\_Defaulting**

Akhilesh Keerthi

03-19-2023

**R Markdown**

*# Suppress dplyr summarise grouping warning messages*  
options(dplyr.summarise.inform = FALSE)  
  
***## Add R libraries here***  
library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.4.0 ✔ purrr 1.0.1  
## ✔ tibble 3.1.8 ✔ dplyr 1.1.0  
## ✔ tidyr 1.3.0 ✔ stringr 1.5.0  
## ✔ readr 2.1.3 ✔ forcats 1.0.0  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(tidymodels)

## ── Attaching packages ────────────────────────────────────── tidymodels 1.0.0 ──  
## ✔ broom 1.0.3 ✔ rsample 1.1.1  
## ✔ dials 1.1.0 ✔ tune 1.0.1  
## ✔ infer 1.0.4 ✔ workflows 1.1.3  
## ✔ modeldata 1.1.0 ✔ workflowsets 1.0.0  
## ✔ parsnip 1.0.4 ✔ yardstick 1.1.0  
## ✔ recipes 1.0.5   
## ── Conflicts ───────────────────────────────────────── tidymodels\_conflicts() ──  
## ✖ scales::discard() masks purrr::discard()  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ recipes::fixed() masks stringr::fixed()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ yardstick::spec() masks readr::spec()  
## ✖ recipes::step() masks stats::step()  
## • Use tidymodels\_prefer() to resolve common conflicts.

library(dplyr)  
library(rlang)

##   
## Attaching package: 'rlang'  
##   
## The following objects are masked from 'package:purrr':  
##   
## %@%, flatten, flatten\_chr, flatten\_dbl, flatten\_int, flatten\_lgl,  
## flatten\_raw, invoke, splice

library(ggplot2)  
library(skimr)  
library(vip)

##   
## Attaching package: 'vip'  
##   
## The following object is masked from 'package:utils':  
##   
## vi

library(rpart.plot)

## Loading required package: rpart  
##   
## Attaching package: 'rpart'  
##   
## The following object is masked from 'package:dials':  
##   
## prune

library(parsnip)  
  
*# Load data*  
loans\_df <- read\_rds("C:/Users/sun/Desktop/shareit/loan\_data.rds")  
  
*#-------------------------------------------------------------------------------*  
*# Data Analysis [30 Points]*  
  
*#Question 1*  
*# Are there any dependencies between loan takers and loan defaulters seen in their credit lines?*  
  
*#Ans:*  
*# Using visualizations to compare the credit lines taken by loan defaulters and non-defaulters in order to determine whether the credit lines variable has an impact on loan defaulters.*  
*# This plot demonstrates how skewed the data are, and step YeoJohnson is used to balance the distribution. The number of credit lines taken is lower for loan defaulters, with the data being skewed to the left.*  
  
*#For Loan-Defaulters as No, the number of credit lines are more in number*  
*# with the data being skewed to the right side*  
*#Code starts*  
loan\_default\_yes <- loans\_df %>% filter(loan\_default == 'yes')  
*#print(loan\_default\_yes)*  
*#Code starts*  
loan\_default\_no <- loans\_df %>% filter(loan\_default == 'no')  
*#print(loan\_default\_no)*  
  
  
ggplot(data = loan\_default\_yes, mapping = aes(x = total\_credit\_lines)) +  
 geom\_histogram(fill = '#006EA1', color = 'white', bins = 15) +  
 labs(title = 'Distribution of Total Credit Lines for Loan-Defaulters',  
 x = 'Total Credit Lines',  
 y = 'Number of Loan-Defaulters')



ggplot(data = loan\_default\_no, mapping = aes(x = total\_credit\_lines)) +  
 geom\_histogram(fill = '#006EA1', color = 'white', bins = 15) +  
 labs(title = 'Distribution of Total Credit Lines for Loan-Non-Defaulters',  
 x = 'Total credit Lines',  
 y = 'Number of Loan-Non-Defaulters')



*# Splitting dataset into Training and Test Data*  
set.seed(1)  
loan\_split <- initial\_split(loans\_df, prop = 0.75, strata = loan\_default)  
  
*# Generate a training data frame*  
loan\_df\_training <- loan\_split %>% training()  
  
*# View results*  
loan\_df\_training

## # A tibble: 3,082 × 16  
## loan\_…¹ loan\_…² insta…³ inter…⁴ loan\_…⁵ appli…⁶ term homeo…⁷ annua…⁸ curre…⁹  
## <fct> <int> <dbl> <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl>  
## 1 no 28800 942. 8.97 debt\_c… indivi… thre… rent 160000 10  
## 2 no 26000 619. 12.0 debt\_c… indivi… five… mortga… 125000 5  
## 3 no 5500 176. 7.97 debt\_c… indivi… thre… rent 70000 4  
## 4 no 40000 952. 11.0 home\_i… indivi… five… mortga… 70000 3  
## 5 no 36000 764. 7.22 small\_… indivi… five… mortga… 185000 10  
## 6 no 6000 195. 11.2 debt\_c… indivi… thre… own 31000 10  
## 7 no 10000 226. 13.2 debt\_c… indivi… five… rent 25000 1  
## 8 no 10000 301. 7.22 small\_… indivi… thre… own 60000 10  
## 9 no 6500 204. 10.5 debt\_c… indivi… thre… rent 52000 3  
## 10 no 10000 308. 8.47 medical indivi… thre… rent 85000 10  
## # … with 3,072 more rows, 6 more variables: debt\_to\_income <dbl>,  
## # total\_credit\_lines <int>, years\_credit\_history <dbl>,  
## # missed\_payment\_2\_yr <fct>, history\_bankruptcy <fct>,  
## # history\_tax\_liens <fct>, and abbreviated variable names ¹​loan\_default,  
## # ²​loan\_amount, ³​installment, ⁴​interest\_rate, ⁵​loan\_purpose,  
## # ⁶​application\_type, ⁷​homeownership, ⁸​annual\_income, ⁹​current\_job\_years

*# Generate a training data frame*  
loan\_df\_test <- loan\_split %>% testing()  
  
*# View results*  
loan\_df\_test

## # A tibble: 1,028 × 16  
## loan\_…¹ loan\_…² insta…³ inter…⁴ loan\_…⁵ appli…⁶ term homeo…⁷ annua…⁸ curre…⁹  
## <fct> <int> <dbl> <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl>  
## 1 no 3600 111. 9.72 medical indivi… thre… mortga… 72000 4  
## 2 no 15000 452. 11.2 small\_… indivi… thre… rent 42000 10  
## 3 yes 9600 366. 11.2 home\_i… indivi… thre… rent 60000 10  
## 4 no 11500 241. 7.72 debt\_c… indivi… five… rent 34416 5  
## 5 yes 28000 1000. 10.8 medical indivi… thre… mortga… 75000 10  
## 6 yes 19200 509. 15.8 small\_… indivi… five… rent 43500 3  
## 7 no 6000 186. 13.2 small\_… indivi… thre… own 180000 10  
## 8 yes 8500 309. 18 debt\_c… indivi… thre… mortga… 103500 4  
## 9 yes 3000 109. 10.2 small\_… indivi… thre… mortga… 65000 10  
## 10 no 20000 609. 9.22 small\_… indivi… thre… own 85000 9  
## # … with 1,018 more rows, 6 more variables: debt\_to\_income <dbl>,  
## # total\_credit\_lines <int>, years\_credit\_history <dbl>,  
## # missed\_payment\_2\_yr <fct>, history\_bankruptcy <fct>,  
## # history\_tax\_liens <fct>, and abbreviated variable names ¹​loan\_default,  
## # ²​loan\_amount, ³​installment, ⁴​interest\_rate, ⁵​loan\_purpose,  
## # ⁶​application\_type, ⁷​homeownership, ⁸​annual\_income, ⁹​current\_job\_years

summary(loan\_df\_training)

## loan\_default loan\_amount installment interest\_rate   
## yes:1147 Min. : 1000 Min. : 32.01 Min. : 4.72   
## no :1935 1st Qu.: 9219 1st Qu.: 273.79 1st Qu.: 8.22   
## Median :15000 Median : 418.90 Median :11.25   
## Mean :16687 Mean : 489.56 Mean :11.42   
## 3rd Qu.:24000 3rd Qu.: 662.53 3rd Qu.:13.75   
## Max. :40000 Max. :1566.59 Max. :20.00   
## loan\_purpose application\_type term homeownership   
## debt\_consolidation:915 individual:2628 three\_year:1946 mortgage:1456   
## credit\_card :676 joint : 454 five\_year :1136 rent :1242   
## medical :472 own : 384   
## small\_business :634   
## home\_improvement :385   
##   
## annual\_income current\_job\_years debt\_to\_income total\_credit\_lines  
## Min. : 3000 Min. : 0.00 Min. : 0.00 Min. : 2.00   
## 1st Qu.: 45000 1st Qu.: 2.00 1st Qu.: 11.66 1st Qu.:13.00   
## Median : 65000 Median : 5.00 Median : 18.57 Median :21.00   
## Mean : 72608 Mean : 5.77 Mean : 20.01 Mean :22.59   
## 3rd Qu.: 91000 3rd Qu.:10.00 3rd Qu.: 26.02 3rd Qu.:29.00   
## Max. :200000 Max. :10.00 Max. :437.61 Max. :87.00   
## years\_credit\_history missed\_payment\_2\_yr history\_bankruptcy history\_tax\_liens  
## Min. : 3.00 yes: 362 yes: 358 yes: 48   
## 1st Qu.:11.00 no :2720 no :2724 no :3034   
## Median :14.00   
## Mean :15.75   
## 3rd Qu.:19.00   
## Max. :49.00

*#Comparing the number of Credit lines to the loan*  
loaners\_recipe <- recipe(loan\_default ~ .,  
 data = loan\_df\_training)  
print(loaners\_recipe)

##   
## ── Recipe ──────────────────────────────────────────────────────────────────────  
##   
## ── Inputs   
## Number of variables by role  
## outcome: 1  
## predictor: 15

summary(loaners\_recipe)

## # A tibble: 16 × 4  
## variable type role source   
## <chr> <list> <chr> <chr>   
## 1 loan\_amount <chr [2]> predictor original  
## 2 installment <chr [2]> predictor original  
## 3 interest\_rate <chr [2]> predictor original  
## 4 loan\_purpose <chr [3]> predictor original  
## 5 application\_type <chr [3]> predictor original  
## 6 term <chr [3]> predictor original  
## 7 homeownership <chr [3]> predictor original  
## 8 annual\_income <chr [2]> predictor original  
## 9 current\_job\_years <chr [2]> predictor original  
## 10 debt\_to\_income <chr [2]> predictor original  
## 11 total\_credit\_lines <chr [2]> predictor original  
## 12 years\_credit\_history <chr [2]> predictor original  
## 13 missed\_payment\_2\_yr <chr [3]> predictor original  
## 14 history\_bankruptcy <chr [3]> predictor original  
## 15 history\_tax\_liens <chr [3]> predictor original  
## 16 loan\_default <chr [3]> outcome original

loaners\_recipe %>%  
 step\_normalize(all\_numeric(), -all\_outcomes()) %>%  
 prep() %>%  
 bake(new\_data = loan\_df\_test)

## # A tibble: 1,028 × 16  
## loan\_…¹ insta…² inter…³ loan\_…⁴ appli…⁵ term homeo…⁶ annua…⁷ curre…⁸ debt\_…⁹  
## <dbl> <dbl> <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl> <dbl>  
## 1 -1.30 -1.31 -0.436 medical indivi… thre… mortga… -0.0164 -0.480 0.181   
## 2 -0.168 -0.131 -0.0522 small\_… indivi… thre… rent -0.825 1.15 0.524   
## 3 -0.705 -0.427 -0.0446 home\_i… indivi… thre… rent -0.340 1.15 -0.0302  
## 4 -0.516 -0.857 -0.947 debt\_c… indivi… five… rent -1.03 -0.209 1.21   
## 5 1.12 1.76 -0.172 medical indivi… thre… mortga… 0.0645 1.15 0.724   
## 6 0.250 0.0660 1.11 small\_… indivi… five… rent -0.785 -0.752 0.456   
## 7 -1.06 -1.05 0.459 small\_… indivi… thre… own 2.89 1.15 -0.498   
## 8 -0.814 -0.622 1.68 debt\_c… indivi… thre… mortga… 0.833 -0.480 0.765   
## 9 -1.36 -1.31 -0.300 small\_… indivi… thre… mortga… -0.205 1.15 -0.439   
## 10 0.329 0.413 -0.563 small\_… indivi… thre… own 0.334 0.877 -0.597   
## # … with 1,018 more rows, 6 more variables: total\_credit\_lines <dbl>,  
## # years\_credit\_history <dbl>, missed\_payment\_2\_yr <fct>,  
## # history\_bankruptcy <fct>, history\_tax\_liens <fct>, loan\_default <fct>, and  
## # abbreviated variable names ¹​loan\_amount, ²​installment, ³​interest\_rate,  
## # ⁴​loan\_purpose, ⁵​application\_type, ⁶​homeownership, ⁷​annual\_income,  
## # ⁸​current\_job\_years, ⁹​debt\_to\_income

print(loaners\_recipe)

##   
## ── Recipe ──────────────────────────────────────────────────────────────────────  
##   
## ── Inputs   
## Number of variables by role  
## outcome: 1  
## predictor: 15

*#b.step\_corr() Highly Correlated Predictors cause problem in model fitting,*  
loaners\_recipe %>%  
 step\_corr(all\_numeric(), -all\_outcomes()) %>%  
 prep() %>%  
 bake(new\_data = loan\_df\_test)

## # A tibble: 1,028 × 15  
## insta…¹ inter…² loan\_…³ appli…⁴ term homeo…⁵ annua…⁶ curre…⁷ debt\_…⁸ total…⁹  
## <dbl> <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl> <dbl> <int>  
## 1 111. 9.72 medical indivi… thre… mortga… 72000 4 22.7 35  
## 2 452. 11.2 small\_… indivi… thre… rent 42000 10 27.7 16  
## 3 366. 11.2 home\_i… indivi… thre… rent 60000 10 19.6 13  
## 4 241. 7.72 debt\_c… indivi… five… rent 34416 5 37.8 27  
## 5 1000. 10.8 medical indivi… thre… mortga… 75000 10 30.7 25  
## 6 509. 15.8 small\_… indivi… five… rent 43500 3 26.7 28  
## 7 186. 13.2 small\_… indivi… thre… own 180000 10 12.7 26  
## 8 309. 18 debt\_c… indivi… thre… mortga… 103500 4 31.3 24  
## 9 109. 10.2 small\_… indivi… thre… mortga… 65000 10 13.5 16  
## 10 609. 9.22 small\_… indivi… thre… own 85000 9 11.2 22  
## # … with 1,018 more rows, 5 more variables: years\_credit\_history <dbl>,  
## # missed\_payment\_2\_yr <fct>, history\_bankruptcy <fct>,  
## # history\_tax\_liens <fct>, loan\_default <fct>, and abbreviated variable names  
## # ¹​installment, ²​interest\_rate, ³​loan\_purpose, ⁴​application\_type,  
## # ⁵​homeownership, ⁶​annual\_income, ⁷​current\_job\_years, ⁸​debt\_to\_income,  
## # ⁹​total\_credit\_lines

print(loaners\_recipe)

##   
## ── Recipe ──────────────────────────────────────────────────────────────────────  
##   
## ── Inputs   
## Number of variables by role  
## outcome: 1  
## predictor: 15

summary(loaners\_recipe)

## # A tibble: 16 × 4  
## variable type role source   
## <chr> <list> <chr> <chr>   
## 1 loan\_amount <chr [2]> predictor original  
## 2 installment <chr [2]> predictor original  
## 3 interest\_rate <chr [2]> predictor original  
## 4 loan\_purpose <chr [3]> predictor original  
## 5 application\_type <chr [3]> predictor original  
## 6 term <chr [3]> predictor original  
## 7 homeownership <chr [3]> predictor original  
## 8 annual\_income <chr [2]> predictor original  
## 9 current\_job\_years <chr [2]> predictor original  
## 10 debt\_to\_income <chr [2]> predictor original  
## 11 total\_credit\_lines <chr [2]> predictor original  
## 12 years\_credit\_history <chr [2]> predictor original  
## 13 missed\_payment\_2\_yr <chr [3]> predictor original  
## 14 history\_bankruptcy <chr [3]> predictor original  
## 15 history\_tax\_liens <chr [3]> predictor original  
## 16 loan\_default <chr [3]> outcome original

*#To increase the recall for the regression analysis with the threshold value of 0.3*  
*# Recall measures the proportion of events occurring in the domain that are “captured” by the models.*  
*# After implementing the correlation with threshold of 0.3 to increase the recall for the model*  
*# 2 variables of are removed from the new data*  
loaners\_recipe %>%  
 step\_corr(all\_numeric(), -all\_outcomes(), -has\_role('id variable'), threshold = 0.3) %>%  
 prep() %>%  
 bake(new\_data = loan\_df\_test)

## # A tibble: 1,028 × 13  
## inter…¹ loan\_…² appli…³ term homeo…⁴ annua…⁵ curre…⁶ debt\_…⁷ years…⁸ misse…⁹  
## <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl> <dbl> <dbl> <fct>   
## 1 9.72 medical indivi… thre… mortga… 72000 4 22.7 11 no   
## 2 11.2 small\_… indivi… thre… rent 42000 10 27.7 22 no   
## 3 11.2 home\_i… indivi… thre… rent 60000 10 19.6 12 no   
## 4 7.72 debt\_c… indivi… five… rent 34416 5 37.8 14 no   
## 5 10.8 medical indivi… thre… mortga… 75000 10 30.7 14 no   
## 6 15.8 small\_… indivi… five… rent 43500 3 26.7 8 no   
## 7 13.2 small\_… indivi… thre… own 180000 10 12.7 16 no   
## 8 18 debt\_c… indivi… thre… mortga… 103500 4 31.3 14 no   
## 9 10.2 small\_… indivi… thre… mortga… 65000 10 13.5 21 no   
## 10 9.22 small\_… indivi… thre… own 85000 9 11.2 20 no   
## # … with 1,018 more rows, 3 more variables: history\_bankruptcy <fct>,  
## # history\_tax\_liens <fct>, loan\_default <fct>, and abbreviated variable names  
## # ¹​interest\_rate, ²​loan\_purpose, ³​application\_type, ⁴​homeownership,  
## # ⁵​annual\_income, ⁶​current\_job\_years, ⁷​debt\_to\_income, ⁸​years\_credit\_history,  
## # ⁹​missed\_payment\_2\_yr

print(loaners\_recipe)

##   
## ── Recipe ──────────────────────────────────────────────────────────────────────  
##   
## ── Inputs   
## Number of variables by role  
## outcome: 1  
## predictor: 15

loaners\_recipe %>%  
 step\_YeoJohnson(total\_credit\_lines) %>%  
 prep() %>%  
 bake(new\_data = loan\_default\_yes) %>%  
  
 ggplot(mapping = aes(x = total\_credit\_lines)) +  
 geom\_histogram(fill = '#006EA1', color = 'white', bins = 8) +  
 labs(title = 'Distribution of Total Credit Lines for Loan-Defaulters',  
 x = 'Total Credit Lines',  
 y = 'Number of Loan-Defaulters')



loaners\_recipe %>%  
 step\_YeoJohnson(total\_credit\_lines) %>%  
 prep() %>%  
 bake(new\_data = loan\_default\_no) %>%  
  
 ggplot(mapping = aes(x = total\_credit\_lines)) +  
 geom\_histogram(fill = '#006EA1', color = 'white', bins = 15) +  
 labs(title = 'Distribution of Total Credit Lines for Loan-Non-Defaulters',  
 x = 'Total Credit Lines',  
 y = 'Number of Loan-Non-Defaulters')



*#Question 2*  
*#a.Is there a relationship between the years credit history, total credit history, duration, loan amount, and loan default?*  
  
*#Ans:*   
*# There are fewer people taking out loans for medical purposes than defaulters.*  
*# And those who have medical debt default more frequently on their loans.*  
*# If the borrowers are in the medical industry, the bank must take further measures to ensure that they won't default on their loans.*  
ggplot(loan\_default\_no, aes(x=loan\_purpose, y=loan\_amount)) + geom\_point()



ggplot(loan\_default\_yes, aes(x=loan\_purpose, y=loan\_amount)) + geom\_point()



ggplot(loans\_df, aes(x=years\_credit\_history, y=loan\_amount, color= loan\_default)) + geom\_point() + scale\_x\_log10() + facet\_wrap(~ term)



*#Ans: As we can see that the Loan defaulting are more in between 10-30 years of credit history*  
*# The top default is 40000 with more than 30 years of credit history*  
ggplot(loans\_df, aes(x=total\_credit\_lines, y=loan\_amount, color= loan\_default)) + geom\_point() + scale\_x\_log10() + facet\_wrap(~ term)



*#Ans: As we can observe that the total credit lines are more in number between 10-30 creditlines*  
*# for both loan\_defaulting and loan-non-defaulting*  
*# The top loan defaulting is 40000 for five year term, with 4 loan-defaulters*  
  
  
*#b.Does the correlation seen between years of credit history, total credit history, homeownership, loan amount, and loan default exist?*  
ggplot(loans\_df, aes(x=years\_credit\_history, y=loan\_amount, color= loan\_default))+ geom\_point() + scale\_x\_log10() + facet\_wrap(~ homeownership)



*#Ans: There is few loan defaulters in homeownership of type own and more in mortagage*  
*# Loan amount of around 0-20000 are higher to default loan in the ownership type rent*  
ggplot(loans\_df, aes(x=total\_credit\_lines, y=loan\_amount, color= loan\_default)) + geom\_point() + scale\_x\_log10() + facet\_wrap(~ homeownership)



*#Ans : The data is pointed more in the center of mortgage of total credit lines at 10-30 with more number of loan defaulters in rent with loan*  
*# amount ranging from 0-20000 with more loan-defaulters*  
  
*#Question 3*  
*# Are the dependencies between current job years and loan purpose & Interest rate for the loan-defaulters*  
current\_jy\_10 <- loan\_default\_yes %>%  
 filter(current\_job\_years == 10)  
  
ggplot(current\_jy\_10, aes(x = loan\_purpose, y = interest\_rate)) +  
 geom\_boxplot()



*#Ans: For different loan\_purposes like credit\_card, medical the median is 15% interest\_rate*  
*# For small\_business and improvements the median is just below 15 i.e 14.8% interest\_rate min/ max are also similar*  
*# The first and third quartiles for each of the purpose is dissimilar*

*#-------------------------------------------------------------------------------*  
*# Predictive Modelling 70 points*  
  
*#2-Modelling techniques,*   
*#Using loan\_df with train and test data set to set the seed, while considering variables for model use recipe*  
  
*#Logistic Regressions*  
  
  
  
*#Training and running model for all the dependent variables*  
*# Training model*  
logistic\_model <- glm(loan\_default ~ loan\_amount+installment+interest\_rate+annual\_income+current\_job\_years+debt\_to\_income+total\_credit\_lines+ years\_credit\_history,  
 data = loan\_df\_training,  
 family = "binomial")  
logistic\_model

##   
## Call: glm(formula = loan\_default ~ loan\_amount + installment + interest\_rate +   
## annual\_income + current\_job\_years + debt\_to\_income + total\_credit\_lines +   
## years\_credit\_history, family = "binomial", data = loan\_df\_training)  
##   
## Coefficients:  
## (Intercept) loan\_amount installment   
## 8.794e+00 7.765e-05 -3.596e-03   
## interest\_rate annual\_income current\_job\_years   
## -6.833e-01 7.410e-06 -1.816e-02   
## debt\_to\_income total\_credit\_lines years\_credit\_history   
## -1.207e-02 -4.275e-03 2.528e-02   
##   
## Degrees of Freedom: 3081 Total (i.e. Null); 3073 Residual  
## Null Deviance: 4069   
## Residual Deviance: 2105 AIC: 2123

*# Summary*  
summary(logistic\_model)

##   
## Call:  
## glm(formula = loan\_default ~ loan\_amount + installment + interest\_rate +   
## annual\_income + current\_job\_years + debt\_to\_income + total\_credit\_lines +   
## years\_credit\_history, family = "binomial", data = loan\_df\_training)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.4044 -0.3590 0.1477 0.5366 2.8782   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 8.794e+00 3.739e-01 23.518 < 2e-16 \*\*\*  
## loan\_amount 7.765e-05 1.748e-05 4.441 8.95e-06 \*\*\*  
## installment -3.596e-03 5.996e-04 -5.997 2.01e-09 \*\*\*  
## interest\_rate -6.833e-01 2.624e-02 -26.036 < 2e-16 \*\*\*  
## annual\_income 7.410e-06 1.824e-06 4.062 4.86e-05 \*\*\*  
## current\_job\_years -1.816e-02 1.560e-02 -1.164 0.24430   
## debt\_to\_income -1.207e-02 4.701e-03 -2.568 0.01023 \*   
## total\_credit\_lines -4.275e-03 5.057e-03 -0.845 0.39783   
## years\_credit\_history 2.528e-02 8.545e-03 2.959 0.00309 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 4068.8 on 3081 degrees of freedom  
## Residual deviance: 2105.2 on 3073 degrees of freedom  
## AIC: 2123.2  
##   
## Number of Fisher Scoring iterations: 6

newdata = data.frame(loan\_amount = 4200, installment= 900, interest\_rate = 15, annual\_income = 90000, current\_job\_years = 6, debt\_to\_income= 21, total\_credit\_lines = 9, years\_credit\_history = 9)  
  
*# Predict test data based on model*  
predict\_reg <- predict(logistic\_model,   
 newdata, type = "response")  
predict\_reg

## 1   
## 0.02038335

*#Ans: By using the logistic regression, the predictions based on the variables,*  
*# show that the final prediction outcome is genuine and helps the bank to evaluate their customers.*  
*#-------------------------------------------------------------------*  
  
*#Decision Trees*  
set.seed(1)  
loan\_split <- initial\_split(loans\_df, prop = 0.75, strata = loan\_default)  
  
*# Generate a training data frame*  
loan\_df\_training <- loan\_split %>% training()  
  
*# View results*  
loan\_df\_training

## # A tibble: 3,082 × 16  
## loan\_…¹ loan\_…² insta…³ inter…⁴ loan\_…⁵ appli…⁶ term homeo…⁷ annua…⁸ curre…⁹  
## <fct> <int> <dbl> <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl>  
## 1 no 28800 942. 8.97 debt\_c… indivi… thre… rent 160000 10  
## 2 no 26000 619. 12.0 debt\_c… indivi… five… mortga… 125000 5  
## 3 no 5500 176. 7.97 debt\_c… indivi… thre… rent 70000 4  
## 4 no 40000 952. 11.0 home\_i… indivi… five… mortga… 70000 3  
## 5 no 36000 764. 7.22 small\_… indivi… five… mortga… 185000 10  
## 6 no 6000 195. 11.2 debt\_c… indivi… thre… own 31000 10  
## 7 no 10000 226. 13.2 debt\_c… indivi… five… rent 25000 1  
## 8 no 10000 301. 7.22 small\_… indivi… thre… own 60000 10  
## 9 no 6500 204. 10.5 debt\_c… indivi… thre… rent 52000 3  
## 10 no 10000 308. 8.47 medical indivi… thre… rent 85000 10  
## # … with 3,072 more rows, 6 more variables: debt\_to\_income <dbl>,  
## # total\_credit\_lines <int>, years\_credit\_history <dbl>,  
## # missed\_payment\_2\_yr <fct>, history\_bankruptcy <fct>,  
## # history\_tax\_liens <fct>, and abbreviated variable names ¹​loan\_default,  
## # ²​loan\_amount, ³​installment, ⁴​interest\_rate, ⁵​loan\_purpose,  
## # ⁶​application\_type, ⁷​homeownership, ⁸​annual\_income, ⁹​current\_job\_years

*# Generate a training data frame*  
loan\_df\_test <- loan\_split %>% testing()  
  
*# View results*  
loan\_df\_test

## # A tibble: 1,028 × 16  
## loan\_…¹ loan\_…² insta…³ inter…⁴ loan\_…⁵ appli…⁶ term homeo…⁷ annua…⁸ curre…⁹  
## <fct> <int> <dbl> <dbl> <fct> <fct> <fct> <fct> <dbl> <dbl>  
## 1 no 3600 111. 9.72 medical indivi… thre… mortga… 72000 4  
## 2 no 15000 452. 11.2 small\_… indivi… thre… rent 42000 10  
## 3 yes 9600 366. 11.2 home\_i… indivi… thre… rent 60000 10  
## 4 no 11500 241. 7.72 debt\_c… indivi… five… rent 34416 5  
## 5 yes 28000 1000. 10.8 medical indivi… thre… mortga… 75000 10  
## 6 yes 19200 509. 15.8 small\_… indivi… five… rent 43500 3  
## 7 no 6000 186. 13.2 small\_… indivi… thre… own 180000 10  
## 8 yes 8500 309. 18 debt\_c… indivi… thre… mortga… 103500 4  
## 9 yes 3000 109. 10.2 small\_… indivi… thre… mortga… 65000 10  
## 10 no 20000 609. 9.22 small\_… indivi… thre… own 85000 9  
## # … with 1,018 more rows, 6 more variables: debt\_to\_income <dbl>,  
## # total\_credit\_lines <int>, years\_credit\_history <dbl>,  
## # missed\_payment\_2\_yr <fct>, history\_bankruptcy <fct>,  
## # history\_tax\_liens <fct>, and abbreviated variable names ¹​loan\_default,  
## # ²​loan\_amount, ³​installment, ⁴​interest\_rate, ⁵​loan\_purpose,  
## # ⁶​application\_type, ⁷​homeownership, ⁸​annual\_income, ⁹​current\_job\_years

set.seed(1)  
churn\_folds <- vfold\_cv(loan\_df\_training, v = 5)  
  
loan\_transformations <- recipe(loan\_default ~ .,  
 data = loan\_df\_training) %>%  
 *# Transformation steps*  
 step\_YeoJohnson(all\_numeric(), -all\_outcomes()) %>%  
 step\_normalize(all\_numeric(), -all\_outcomes()) %>%  
 step\_dummy(all\_nominal(), -all\_outcomes()) %>%  
  
 prep()  
  
skim(loans\_df)

*Data summary*

|  |  |
| --- | --- |
| Name | loans\_df |
| Number of rows | 4110 |
| Number of columns | 16 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| factor | 8 |
| numeric | 8 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: factor**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| skim\_variable | n\_missing | complete\_rate | ordered | n\_unique | top\_counts |
| loan\_default | 0 | 1 | FALSE | 2 | no: 2580, yes: 1530 |
| loan\_purpose | 0 | 1 | FALSE | 5 | deb: 1218, cre: 879, sma: 853, med: 635 |
| application\_type | 0 | 1 | FALSE | 2 | ind: 3494, joi: 616 |
| term | 0 | 1 | FALSE | 2 | thr: 2588, fiv: 1522 |
| homeownership | 0 | 1 | FALSE | 3 | mor: 1937, ren: 1666, own: 507 |
| missed\_payment\_2\_yr | 0 | 1 | FALSE | 2 | no: 3640, yes: 470 |
| history\_bankruptcy | 0 | 1 | FALSE | 2 | no: 3624, yes: 486 |
| history\_tax\_liens | 0 | 1 | FALSE | 2 | no: 4050, yes: 60 |

**Variable type: numeric**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| loan\_amount | 0 | 1 | 16692.79 | 10038.89 | 1000.00 | 9600.00 | 15000.00 | 24000.00 | 40000.00 | ▆▇▅▃▂ |
| installment | 0 | 1 | 489.42 | 289.50 | 31.04 | 274.82 | 421.97 | 663.98 | 1566.59 | ▇▇▅▂▁ |
| interest\_rate | 0 | 1 | 11.38 | 3.92 | 4.72 | 8.22 | 11.25 | 13.75 | 20.00 | ▆▆▇▃▃ |
| annual\_income | 0 | 1 | 73015.01 | 37203.11 | 3000.00 | 45000.00 | 65000.00 | 92000.00 | 200000.00 | ▃▇▃▁▁ |
| current\_job\_years | 0 | 1 | 5.80 | 3.69 | 0.00 | 2.00 | 5.00 | 10.00 | 10.00 | ▆▃▂▂▇ |
| debt\_to\_income | 0 | 1 | 20.04 | 14.23 | 0.00 | 11.85 | 18.59 | 26.13 | 437.61 | ▇▁▁▁▁ |
| total\_credit\_lines | 0 | 1 | 22.47 | 12.03 | 2.00 | 14.00 | 20.00 | 29.00 | 87.00 | ▇▇▂▁▁ |
| years\_credit\_history | 0 | 1 | 15.76 | 7.22 | 3.00 | 11.00 | 14.00 | 19.00 | 51.00 | ▆▇▂▁▁ |

tree\_model <- decision\_tree(cost\_complexity = tune(),  
tree\_depth = tune(), min\_n = tune()) %>%  
set\_engine('rpart') %>%  
set\_mode('classification')  
  
*#Model Creation*  
tree\_workflow <- workflow() %>%  
 add\_model(tree\_model) %>%  
add\_recipe(loan\_transformations)  
  
*#Hyperparameter Tuning*  
*#hyperparameter values to test*  
tree\_grid<- grid\_regular(cost\_complexity(), tree\_depth(), min\_n(), levels = 2)  
  
*#View Grid*  
tree\_grid

## # A tibble: 8 × 3  
## cost\_complexity tree\_depth min\_n  
## <dbl> <int> <int>  
## 1 0.0000000001 1 2  
## 2 0.1 1 2  
## 3 0.0000000001 15 2  
## 4 0.1 15 2  
## 5 0.0000000001 1 40  
## 6 0.1 1 40  
## 7 0.0000000001 15 40  
## 8 0.1 15 40

*#Tuning with tune\_grid()*  
*#Tune decision*  
set.seed(1)  
tree\_tuning <- tree\_workflow %>%  
tune\_grid(resamples = churn\_folds, grid=tree\_grid)  
  
*#Show top 5 best models*  
tree\_tuning %>% show\_best('roc\_auc')

## # A tibble: 5 × 9  
## cost\_complexity tree\_depth min\_n .metric .estima…¹ mean n std\_err .config  
## <dbl> <int> <int> <chr> <chr> <dbl> <int> <dbl> <chr>   
## 1 0.0000000001 15 40 roc\_auc binary 0.968 5 0.00292 Prepro…  
## 2 0.0000000001 15 2 roc\_auc binary 0.905 5 0.00292 Prepro…  
## 3 0.0000000001 1 2 roc\_auc binary 0.810 5 0.00639 Prepro…  
## 4 0.1 1 2 roc\_auc binary 0.810 5 0.00639 Prepro…  
## 5 0.1 15 2 roc\_auc binary 0.810 5 0.00639 Prepro…  
## # … with abbreviated variable name ¹​.estimator

*#Select best model*  
best\_tree <- tree\_tuning %>% select\_best(metric = 'roc\_auc')  
  
*#View best tree parametrs*  
best\_tree

## # A tibble: 1 × 4  
## cost\_complexity tree\_depth min\_n .config   
## <dbl> <int> <int> <chr>   
## 1 0.0000000001 15 40 Preprocessor1\_Model7

*#Finalizing Workflow*  
final\_tree\_workflow <- tree\_workflow %>%  
 finalize\_workflow(best\_tree)  
  
*#Fit the Model*  
tree\_wf\_fit <- final\_tree\_workflow %>%  
 fit(data = loan\_df\_training)  
  
*#Explore trained model*  
tree\_fit <- tree\_wf\_fit %>% pull\_workflow\_fit()

## Warning: `pull\_workflow\_fit()` was deprecated in workflows 0.2.3.  
## ℹ Please use `extract\_fit\_parsnip()` instead.

*#Variable\_Importance*  
vip(tree\_fit)



*#Tree Plot*  
rpart.plot(tree\_fit$fit, roundint = FALSE)



*#Train and Evaluate the last\_fit()*  
tree\_last\_fit <- final\_tree\_workflow %>%  
 last\_fit(loan\_split)  
  
*#Accuracy and Area*  
tree\_last\_fit %>%collect\_metrics()

## # A tibble: 2 × 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.897 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.964 Preprocessor1\_Model1

*#collect Predictions*  
*#Estimated Probabilites*  
tree\_predictions <- tree\_last\_fit %>% collect\_predictions()  
  
*#ROC Curve*  
tree\_predictions %>% roc\_curve(truth = loan\_default, estimate = .pred\_yes) %>% autoplot()

## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in  
## dplyr 1.1.0.  
## ℹ Please use `reframe()` instead.  
## ℹ When switching from `summarise()` to `reframe()`, remember that `reframe()`  
## always returns an ungrouped data frame and adjust accordingly.  
## ℹ The deprecated feature was likely used in the yardstick package.  
## Please report the issue at <]8;;https://github.com/tidymodels/yardstick/issueshttps://github.com/tidymodels/yardstick/issues]8;;>.



*#confusion Matrix*  
conf\_mat(tree\_predictions, truth = loan\_default, estimate = .pred\_class)

## Truth  
## Prediction yes no  
## yes 306 29  
## no 77 616

*#Conclusion:*  
*# Decision Tree*  
*# After implementing the decision tree algorithm for tuning the decision to make based on*  
*# the variables that are available in the project, we can observe that there are 5 best models based on*  
*# the roc\_auc method.*  
*# The best model has the cost\_complexity of 0.000000001 with the tree\_depth of 15.*  
*# The tree's last fit's accuracy's estimate is 0.897 and roc\_auc's estimate is 0.964*  
  
*# The Predicition shows that*  
*# Model correctly classified 306 True Positive class data points.*  
*# Model correctly classified 616 True Negative class data points.*  
*# Model incorrectly classified 29 False Positive class data points.*  
*# Model incorrectly classified 77 False Negative class data points.*  
*# The predictions show that Given the comparatively higher proportion of true positive and true negative values in our dataset, this proved to be a rather good classifier*  
  
*#Random Forest*  
*# Model Specification*  
rf\_model <- rand\_forest(mtry = tune(), trees = tune(), min\_n = tune()) %>%   
 set\_engine('ranger', importance = "impurity") %>%  
 set\_mode('classification')  
  
*# Specify the workflow*   
rf\_workflow <- workflow() %>%   
 add\_model(rf\_model) %>%   
 add\_recipe(loan\_transformations)  
  
rf\_workflow

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 3 Recipe Steps  
##   
## • step\_YeoJohnson()  
## • step\_normalize()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = tune()  
## trees = tune()  
## min\_n = tune()  
##   
## Engine-Specific Arguments:  
## importance = impurity  
##   
## Computational engine: ranger

***## Create a grid of hyperparameter values to test***  
set.seed(12)  
rf\_grid <- grid\_random(mtry() %>%   
 range\_set(c(4, 12)),  
 trees(),  
 min\_n(),   
 size = 10)  
  
*# Tune random forest workflow*   
set.seed(314)  
rf\_tuning <- rf\_workflow %>%  
 tune\_grid(resamples = churn\_folds, grid = rf\_grid)  
  
***## Show the top 5 best models based on roc\_auc metric***   
rf\_tuning %>%show\_best('roc\_auc')

## # A tibble: 5 × 9  
## mtry trees min\_n .metric .estimator mean n std\_err .config   
## <int> <int> <int> <chr> <chr> <dbl> <int> <dbl> <chr>   
## 1 11 1464 15 roc\_auc binary 0.976 5 0.00256 Preprocessor1\_Model06  
## 2 9 72 21 roc\_auc binary 0.975 5 0.00291 Preprocessor1\_Model10  
## 3 10 618 31 roc\_auc binary 0.974 5 0.00262 Preprocessor1\_Model02  
## 4 12 313 33 roc\_auc binary 0.974 5 0.00255 Preprocessor1\_Model09  
## 5 8 432 35 roc\_auc binary 0.973 5 0.00241 Preprocessor1\_Model04

*# Select best model based on*   
best\_rf <- rf\_tuning %>% select\_best(metric = 'roc\_auc')  
  
*# View the best parameters*  
best\_rf

## # A tibble: 1 × 4  
## mtry trees min\_n .config   
## <int> <int> <int> <chr>   
## 1 11 1464 15 Preprocessor1\_Model06

*# Finalize workflow*   
final\_rf\_workflow <- rf\_workflow %>%   
finalize\_workflow(best\_rf)   
final\_rf\_workflow

## ══ Workflow ════════════════════════════════════════════════════════════════════  
## Preprocessor: Recipe  
## Model: rand\_forest()  
##   
## ── Preprocessor ────────────────────────────────────────────────────────────────  
## 3 Recipe Steps  
##   
## • step\_YeoJohnson()  
## • step\_normalize()  
## • step\_dummy()  
##   
## ── Model ───────────────────────────────────────────────────────────────────────  
## Random Forest Model Specification (classification)  
##   
## Main Arguments:  
## mtry = 11  
## trees = 1464  
## min\_n = 15  
##   
## Engine-Specific Arguments:  
## importance = impurity  
##   
## Computational engine: ranger

*# Fit the model*  
rf\_wf\_fit <- final\_rf\_workflow %>%   
fit(data = loan\_df\_training)  
  
*# Extract trained model*  
rf\_fit <- rf\_wf\_fit %>%  
pull\_workflow\_fit()  
  
*# Variable Importance*   
vip(rf\_fit)



*# Train and evaluate with last\_fit()*  
rf\_last\_fit <- final\_rf\_workflow %>%  
 last\_fit(loan\_split)  
  
*# Accuracy and Area under the ROC curve*   
print('printing accuracy rate and roc curve for the random forests')

## [1] "printing accuracy rate and roc curve for the random forests"

rf\_last\_fit %>%  
 collect\_metrics()

## # A tibble: 2 × 4  
## .metric .estimator .estimate .config   
## <chr> <chr> <dbl> <chr>   
## 1 accuracy binary 0.901 Preprocessor1\_Model1  
## 2 roc\_auc binary 0.966 Preprocessor1\_Model1

*# collect\_predictions()*   
rf\_predictions <- rf\_last\_fit %>% collect\_predictions()  
rf\_predictions

## # A tibble: 1,028 × 7  
## id .pred\_yes .pred\_no .row .pred\_class loan\_default .config   
## <chr> <dbl> <dbl> <int> <fct> <fct> <chr>   
## 1 train/test split 0.0286 0.971 5 no no Preproces…  
## 2 train/test split 0.0734 0.927 11 no no Preproces…  
## 3 train/test split 0.186 0.814 15 no yes Preproces…  
## 4 train/test split 0.00216 0.998 18 no no Preproces…  
## 5 train/test split 0.630 0.370 21 yes yes Preproces…  
## 6 train/test split 1 0 34 yes yes Preproces…  
## 7 train/test split 0.0835 0.916 37 no no Preproces…  
## 8 train/test split 1.00 0.000342 43 yes yes Preproces…  
## 9 train/test split 0.222 0.778 44 no yes Preproces…  
## 10 train/test split 0 1 45 no no Preproces…  
## # … with 1,018 more rows

*# ROC Curve*   
rf\_predictions %>%  
 roc\_curve(truth = loan\_default, estimate = .pred\_yes) %>%   
 autoplot()



*# Confusion matrix*  
conf\_mat(rf\_predictions, truth = loan\_default, estimate = .pred\_class)

## Truth  
## Prediction yes no  
## yes 306 25  
## no 77 620

*#Theory for the project's outputs is as below:-*  
  
*# 1. The data talks about the bank's loan defaulting issue from their customers*  
*#*   
*# - The Company is trying to understand the dependent reasons for the loan-defaulting*  
*# By knowing the main dependencies that cause this problem, the company can solve the loan-defaulting problem*  
*#*   
*# - The Goal of my analysis is to predict the future out-comings of loan-defaulting*   
*# To predict this implementing logistic regression, Decision Trees, Random Forests*  
*# The Main Questions that help to understand the dependencies for this is final analysis are as below*  
  
*# 1. Are there any dependencies between loan takers and loan defaulters seen in their credit lines?*  
*# 2. a.Is there a relationship between the years credit history, total credit history, duration, loan amount, and loan default?*  
*# b.Does the correlation seen between years of credit history, total credit history, home-ownership, loan amount, and loan default exist?*  
*# 3. Are the dependencies between current job years and loan purpose & Interest rate for the loan-defaulters*  
*# 4. Using Logistic Regression to analysis and predict the outcome for a loantaker in future*  
*# 5. Using Decision Tree to find out the best tree and options.*  
*# 6. Using Random Forest to fine tune the best tree that analysises the dependencies of the variables*  
   
*# 2. Highlights and key findings from your Exploratory Data Analysis section*   
*#*   
*# - What were the interesting findings from your analysis and \*\*why are they important for the business\*\*?*  
*# The Interesting findings from the analysis are that the Those who take out loans for medical*   
*# expenses are less likely to default than those who do, and those with medical debt do so more*   
*# frequently. The bank must take further steps to ensure compliance if the borrowers are in the*   
*# medical sector.*  
  
*# - The median interest rate for various loan reasons, such as credit cards and medical, is 15%.*   
*# The minimum and maximum interest rates for small businesses and upgrades are likewise similar.*   
*# The first and third quartiles for each purpose are different.*   
  
*# - The results of the logistic regression demonstrate that the predictions for future events are correct, assisting the bank organization in properly deciding whether to extend credit to consumers and ultimately influencing loan default or non-default rates.*  
  
*# 3. Your “best” classification model and an analysis of its performance*   
*# The three models of logistic regression, Decision Tree and Random Forest have*  
*# all proved to be worthy and valuable for the predictions making.*  
*# But the best of three can either be Decision Tree or Random Forest as they provide the*   
*# True Positives and True Negatives greater than the False Positives and False Negatives as shown in the outputs*  
*#*   
*# - The expected error for the model of Decision tree is around 0.103 and for Random Forests is 0.99 which is better.*   
*#*   
*# - After performing the predictions on the train and test split data it shows that random forest derives accurate outcomes as shown the output of rf\_predicitons*  
*#*   
*# - The Performance metric of ROC Curve for the Random Forest is 0.966 and Decision Tree is 0.964.*  
*#*   
*# Here's the information that is valuable to the executives of the bank, the Random Forest can be 96.6% times dependent and accurate for all the predicitons*  
*# that has to be made.'*  
*# The Decision Tree makes a ROC CUrve of 0.964 that means it is almost 97% times reliable for the predictions and accuracy of the model.*  
  
*# 4. Your recommendations to the company on how to reduce loan default rates*   
*#*   
*# - Recommending the company to take preventive measures by providing loan in more number to homeownership purpose than other if possible*  
*# and less to those for the purpose of mortgage*  
*# - Less homeowners who own their homes default on their loans than those who mortgage them. # Loan amounts of between 0 and 20,000 are more likely to default than those who own their homes but rent them.*  
  
*# Potential Impact and benefits*   
*# - The potential impact of this might be that you might miss some valuable customers that might not loan-default but pay back in time, this is conuter-feited by the matter of fact that the bank organisation has to lessen the amount of loan that can be awared to them to lesser than 5000, as it will save money to the organisation and would not risk the time & money into such scenarios.\*  
  
  
*# 5. Conclusion*  
  
*# Using visualizations to compare the credit lines taken by loan defaulters and non-defaulters in order to determine whether the credit lines variable has an impact on loan defaulters.*  
*# This plot demonstrates how skewed the data are, and step YeoJohnson is used to balance the distribution. The number of credit lines taken is lower for loan defaulters, with the data being skewed to the left.*  
*# For Loan-Defaulters as No, the number of credit lines are more in number*  
*# with the data being skewed to the right side*  
  
*# To increase the recall for the regression analysis with the threshold value of 0.3*  
*# Recall measures the proportion of events occurring in the domain that are “captured” by the models.*  
*# After implementing the correlation with threshold of 0.3 to increase the recall for the model*  
*# 2 variables of are removed from the new data*  
  
*# There are fewer people taking out loans for medical purposes than defaulters.*  
*# And those who have medical debt default more frequently on their loans.*  
*# If the borrowers are in the medical industry, the bank must take further measures to ensure that they won't default on their loans.*  
  
*# As we can see that the Loan defaulting are more in between 10-30 years of credit history*  
*# The top default is 40000 with more than 30 years of credit history*  
  
*# As we can observe that the total credit lines are more in number between 10-30 creditlines*  
*# for both loan\_defaulting and loan-non-defaulting*  
*# The top loan defaulting is 40000 for five year term, with 4 loan-defaulters*  
  
*# There is few loan defaulters in homeownership of type own and more in mortagage*  
*# Loan amount of around 0-20000 are higher to default loan in the ownership type rent*  
  
*# The data is pointed more in the center of mortgage of total credit lines at 10-30 with more number of loan defaulters in rent with loan*  
*# amount ranging from 0-20000 with more loan-defaulters*  
  
*# For different loan\_purposes like credit\_card, medical the median is 15% interest\_rate*  
*# For small\_business and improvements the median is just below 15 i.e 14.8% interest\_rate min/ max are also similar*  
*# The first and third quartiles for each of the purpose is dissimilar*  
  
*# By using the logistic regression, the predictions based on the variables,*  
*# show that the final prediction outcome is genuine and helps the bank to evaluate their customers.*  
  
*# Decision Tree*  
*# After implementing the decision tree algorithm for tuning the decision to make based on*  
*# the variables that are available in the project, we can observe that there are 5 best models based on*  
*# the roc\_auc method.*  
*# The best model has the cost\_complexity of 0.000000001 with the tree\_depth of 15.*  
*# The tree's last fit's accuracy's estimate is 0.897 and roc\_auc's estimate is 0.964*  
  
*# The Prediction shows that*  
*# Model correctly classified 306 True Positive class data points.*  
*# Model correctly classified 616 True Negative class data points.*  
*# Model incorrectly classified 29 False Positive class data points.*  
*# Model incorrectly classified 77 False Negative class data points.*  
*# The predictions show that Given the comparatively higher proportion of true positive and true negative values in our dataset, this proved to be a rather good classifier*  
  
*#Random Forest*  
*# On implementing the Random Forest Algorithm for fine-tuning, there is one best tree with the right combinations for analysis.*  
*# The Prediction shows that*  
*# Model correctly classified 306 True Positive class data points.*  
*# Model correctly classified 620 True Negative class data points.*  
*# Model incorrectly classified 29 False Positive class data points.*  
*# Model incorrectly classified 77 False Negative class data points.*  
*# The predictions show that Given the comparatively higher proportion of true positive and true negative values in our dataset, this proved to be a rather good classifier*  
  
*#After working on both Decision Tree and Random Forest, it is clear that these can be*   
*# trusted for the purpose of decision making in this case.*  
  
*# The predictions from the Random Forest fitted model shows that almost all the predictions*   
*# display great value of efficiency by making the output genuine(i.e.matching exactly the final prediction with the loan-default's value for that instance)*