Credit risk model for forecasting loan default

2024-01-06

```
#importing data

data <-
read.csv('/home/vaasala/Desktop/submission/loan2.csv')#importing the
dataframe
head(data)#visualizing the dataframe

##   X loan_amnt      term int_rate installment grade emp_length
home_ownership
## 1 1        NA 60 months   100.99          NA     E  10+ years
MORTGAGE
## 2 2         0 36 months     0.00        0.00     A   < 1 year
RENT
## 3 3      2500 36 months    13.98       85.42     C    4 years
RENT
## 4 4      5000 36 months    15.95      175.67     D    4 years
RENT
## 5 5      7000 36 months     9.91      225.58     B  10+ years
MORTGAGE
## 6 6      2000 36 months     5.42       60.32     A  10+ years
RENT
##   annual_inc                                        loan_status
## 1         NA                                            Current
## 2          0                                        Charged Off
## 3      20004 Does not meet the credit policy. Status:Fully Paid
## 4      59000                                        Charged Off
## 5      53796                                         Fully Paid
## 6      30000                                         Fully Paid
##              purpose    dti delinq_2yrs open_acc pub_rec revol_util
total_acc
## 1        credit_card 100.00          NA       NA      NA  ﷿100.00%﷿
NA
## 2    major_purchase   0.00           0        0       0    ﷿0.00%﷿
1
## 3              other  19.86           0        7       0      0.213
10
## 4 debt_consolidation  19.57           0        7       0      0.999
15
## 5              other  10.80           3        7       0      0.472
20
## 6 debt_consolidation   3.60           0        7       0          0
15
##   total_pymnt_inv total_rec_prncp total_rec_int repay_fail
## 1              NA              NA            NA          0
```

```
## 2            0.00            0.00          0.00          1
## 3         3075.29         2500.00        575.29          0
## 4         2948.76         1909.02        873.81          1
## 5         8082.39         7000.00       1082.39          0
## 6         2161.66         2000.00        161.66          0

#(part1)data cleaning

#number of rows before cleaning
nrow(data)

## [1] 38480

#removing missing values
data2 <- na.omit(data) #function removes rows with NA
head(data2)#visualizing the dataframe

##   X loan_amnt      term int_rate installment grade emp_length
home_ownership
## 2 2         0 36 months     0.00        0.00     A   < 1 year
RENT
## 3 3      2500 36 months    13.98       85.42     C    4 years
RENT
## 4 4      5000 36 months    15.95      175.67     D    4 years
RENT
## 5 5      7000 36 months     9.91      225.58     B  10+ years
MORTGAGE
## 6 6      2000 36 months     5.42       60.32     A  10+ years
RENT
## 7 7      3600 36 months    10.25      116.59     B  10+ years
MORTGAGE
##   annual_inc                                          loan_status
## 2          0                                          Charged Off
## 3      20004 Does not meet the credit policy. Status:Fully Paid
## 4      59000                                          Charged Off
## 5      53796                                           Fully Paid
## 6      30000                                           Fully Paid
## 7     675048 Does not meet the credit policy. Status:Fully Paid
##              purpose   dti delinq_2yrs open_acc pub_rec revol_util
total_acc
## 2     major_purchase  0.00           0        0       0    0.00%
1
## 3              other 19.86           0        7       0      0.213
10
## 4 debt_consolidation 19.57           0        7       0      0.999
15
## 5              other 10.80           3        7       0      0.472
20
## 6 debt_consolidation  3.60           0        7       0          0
15
## 7              other  1.55           0        8       0          0
```

```
25
##   total_pymnt_inv total_rec_prncp total_rec_int repay_fail
## 2            0.00            0.00          0.00          1
## 3         3075.29         2500.00        575.29          0
## 4         2948.76         1909.02        873.81          1
## 5         8082.39         7000.00       1082.39          0
## 6         2161.66         2000.00        161.66          0
## 7         4206.03         3600.00        606.03          0
```

```
#number of rows after cleaning
nrow(data2)
```

```
## [1] 38419
```

```
#the number of rows removed during cleaning
dif <- nrow(data)-nrow(data2)
sprintf('the number of rows removed is %d',dif)
```

```
## [1] "the number of rows removed is 61"
```

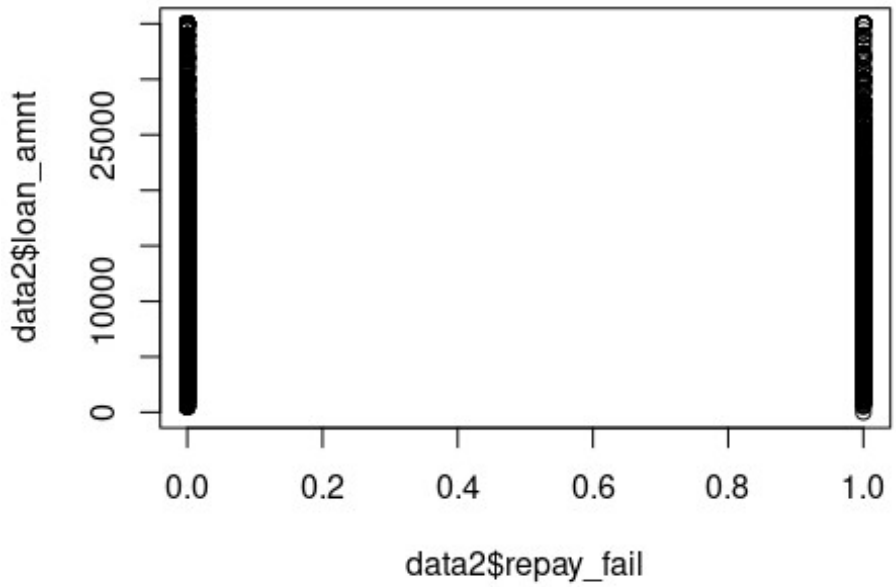
```
#number of columns in data2
ncol(data2)
```

```
## [1] 21
```

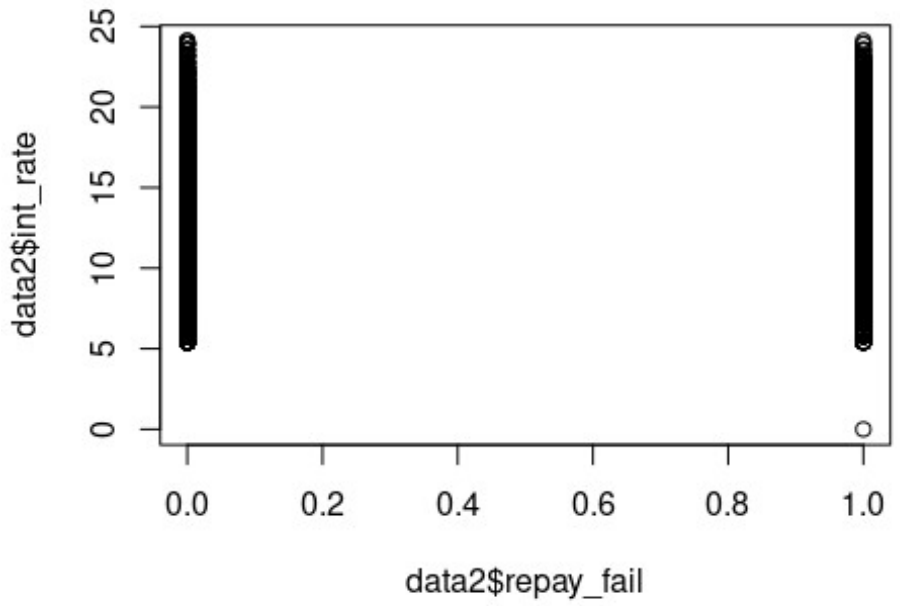
```
#(part2)feature selection
#scatter plot approach
str(data2)
```

```
## 'data.frame':    38419 obs. of  21 variables:
##  $ X              : int  2 3 4 5 6 7 8 9 10 11 ...
##  $ loan_amnt      : int  0 2500 5000 7000 2000 3600 8000 6000 25600
19750 ...
##  $ term           : chr  "36 months" "36 months" "36 months" "36
months" ...
##  $ int_rate       : num  0 13.98 15.95 9.91 5.42 ...
##  $ installment    : num  0 85.4 175.7 225.6 60.3 ...
##  $ grade          : chr  "A" "C" "D" "B" ...
##  $ emp_length     : chr  "< 1 year" "4 years" "4 years" "10+ years"
...
##  $ home_ownership : chr  "RENT" "RENT" "RENT" "MORTGAGE" ...
##  $ annual_inc     : num  0 20004 59000 53796 30000 ...
##  $ loan_status    : chr  "Charged Off" "Does not meet the credit
policy. Status:Fully Paid" "Charged Off" "Fully Paid" ...
##  $ purpose        : chr  "major_purchase" "other"
"debt_consolidation" "other" ...
##  $ dti            : num  0 19.9 19.6 10.8 3.6 ...
##  $ delinq_2yrs    : int  0 0 0 3 0 0 0 0 0 0 ...
##  $ open_acc       : int  0 7 7 7 7 8 12 5 16 15 ...
##  $ pub_rec        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ revol_util     : chr  "0.00%" "0.213" "0.999" "0.472" ...
##  $ total_acc      : int  1 10 15 20 15 25 49 9 32 44 ...
```

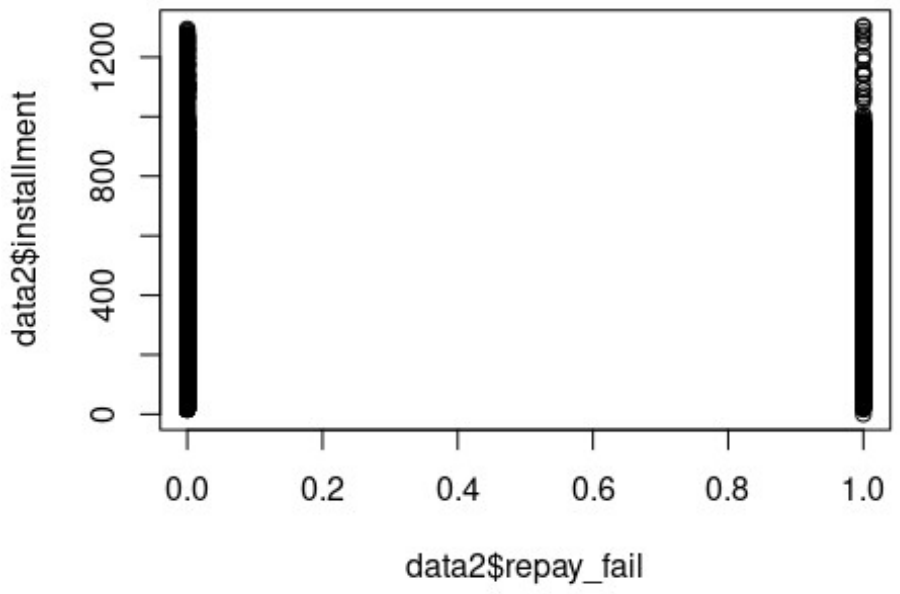
```
##  $ total_pymnt_inv: num  0 3075 2949 8082 2162 ...
##  $ total_rec_prncp: num  0 2500 1909 7000 2000 ...
##  $ total_rec_int  : num  0 575 874 1082 162 ...
##  $ repay_fail     : int  1 0 1 0 0 0 0 0 0 0 ...
##  - attr(*, "na.action")= 'omit' Named int [1:61] 1 1124 3070 3541
3647 3669 3887 4261 5132 6115 ...
##   ..- attr(*, "names")= chr [1:61] "1" "1124" "3070" "3541" ...

#relationship between independent variables and repay_fail
plot(data2$repay_fail,data2$loan_amnt) #loan amount vs repay fail
```



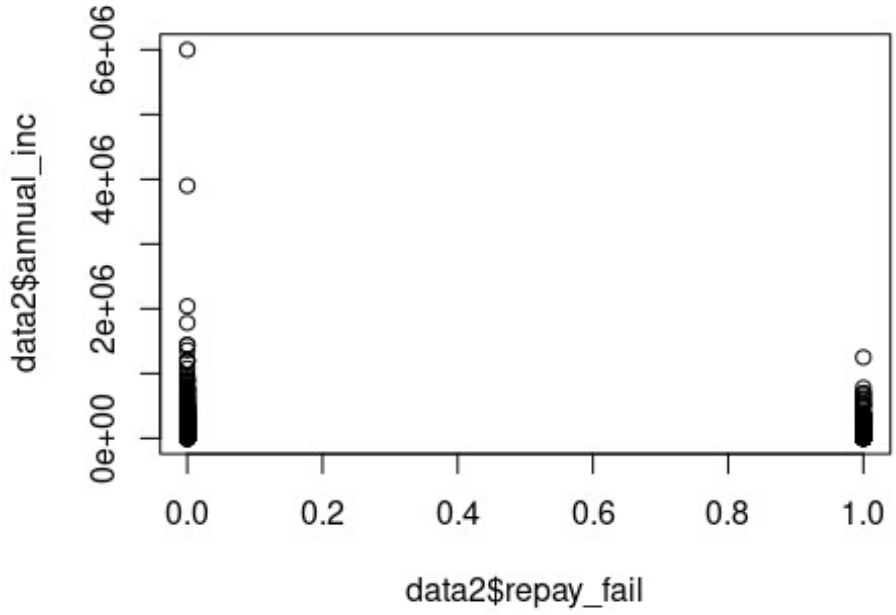

```
plot(data2$repay_fail,data2$int_rate) #int rate vs repay fail
```



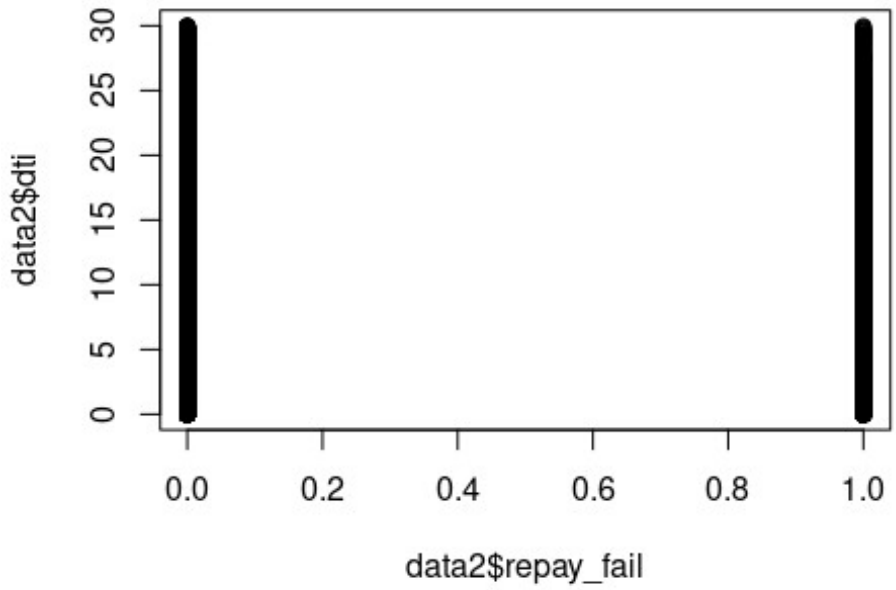

```r
plot(data2$repay_fail,data2$installment) #installment vs repay fail
```



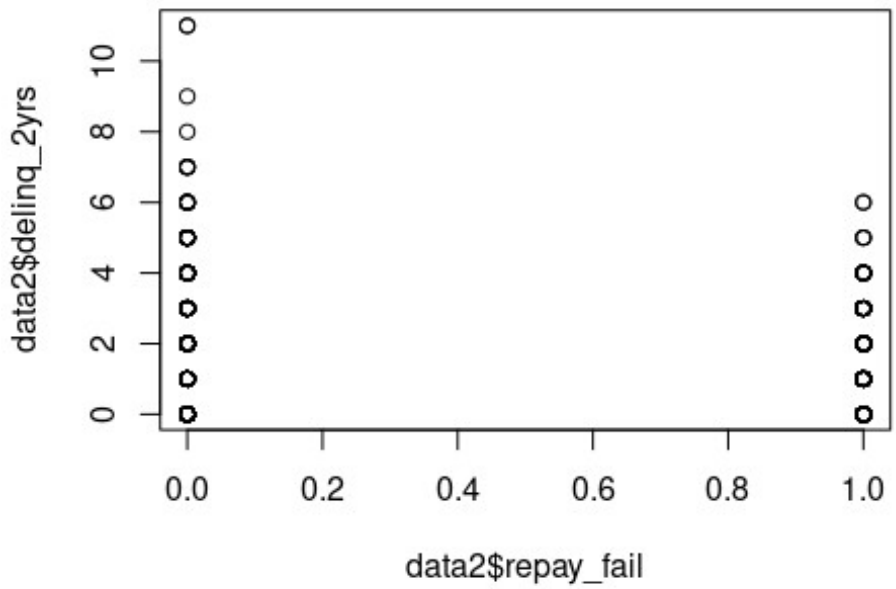

```r
plot(data2$repay_fail,data2$annual_inc) #annual inc vs repay fail
```



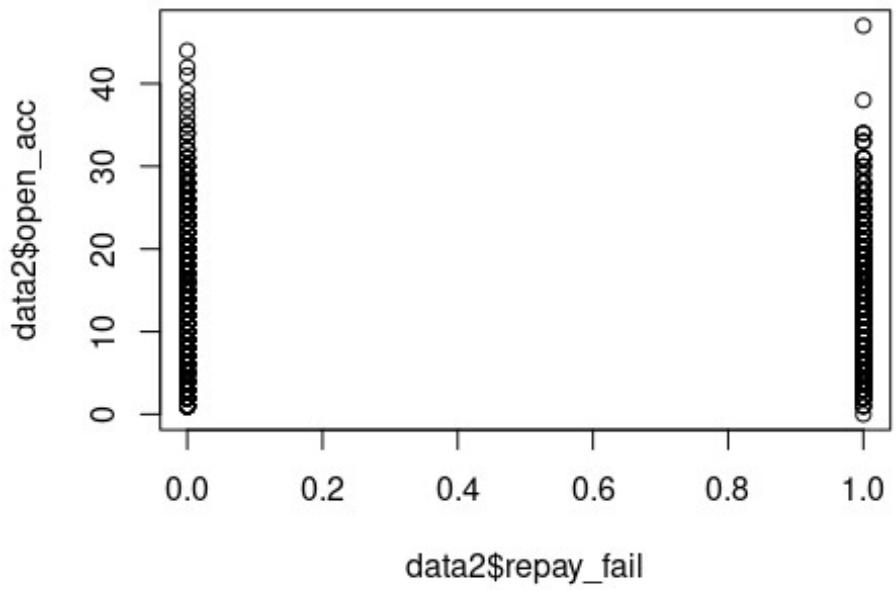

```r
plot(data2$repay_fail,data2$dti) #dti vs repay fail
```



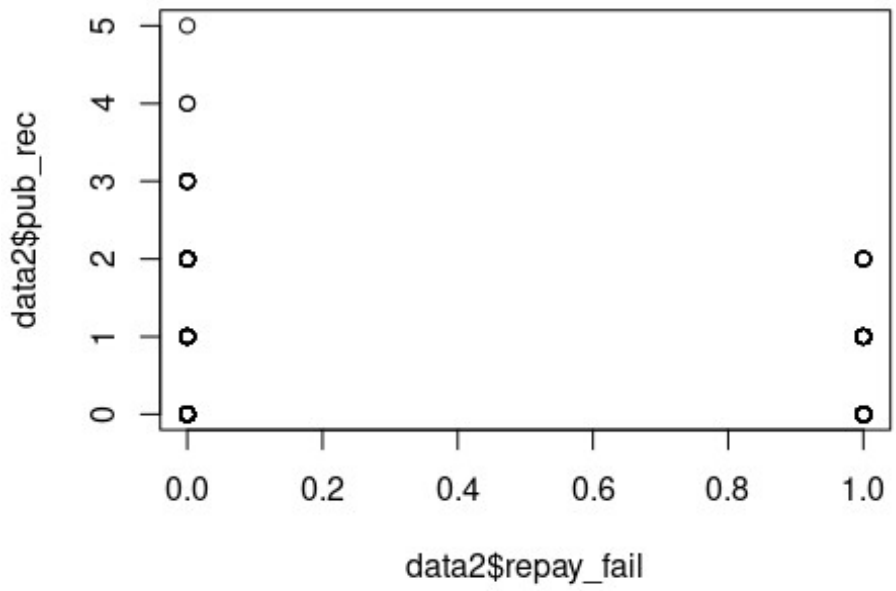

```r
plot(data2$repay_fail,data2$delinq_2yrs) #deling 2years vs repay fail
```



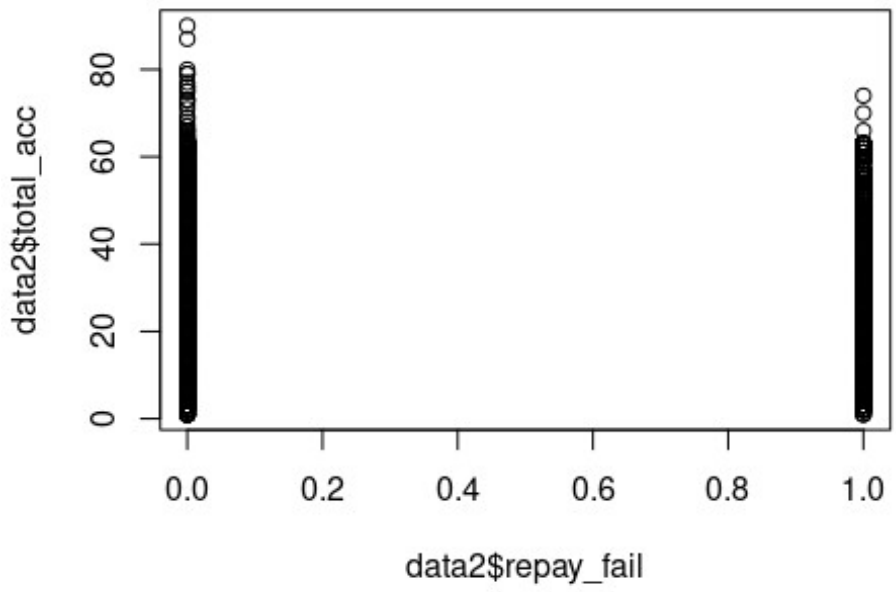

```r
plot(data2$repay_fail,data2$open_acc) #open acc vs repay fail
```



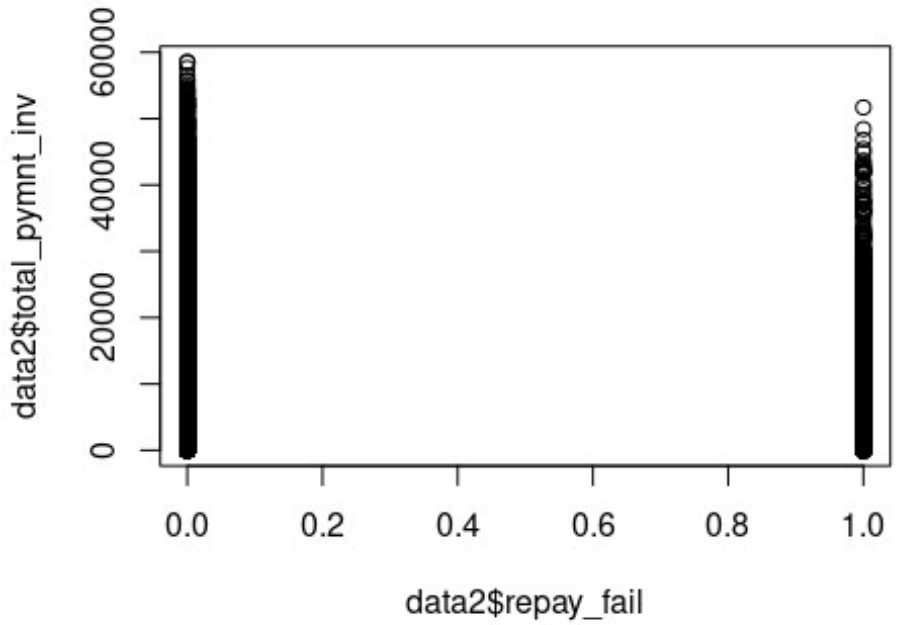

```r
plot(data2$repay_fail,data2$pub_rec) #pub rec vs repay fail
```



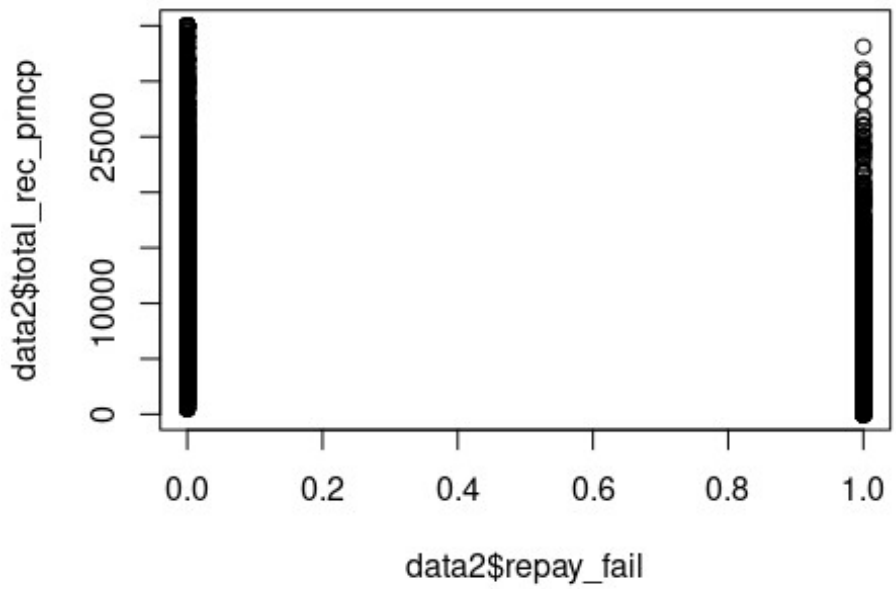

```r
plot(data2$repay_fail,data2$total_acc) #total acc vs repay fail
```



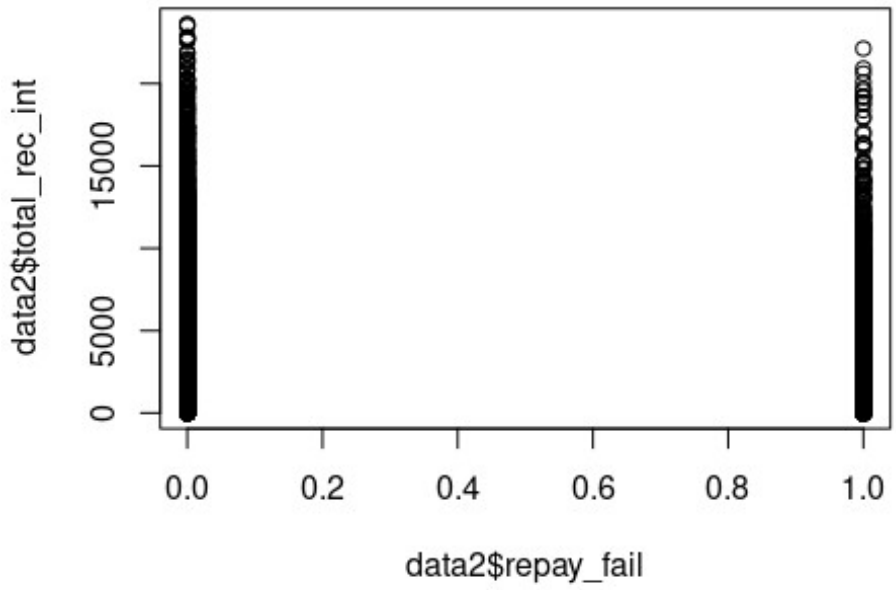
```r
plot(data2$repay_fail,data2$total_pymnt_inv) #total payment vs repay fail
```




```r
plot(data2$repay_fail,data2$total_rec_prncp) #total_rec_prncp vs repau_fail
```




```
plot(data2$repay_fail,data2$total_rec_int) #total_rec_int vs
repay_fail
```



```r
#since the output is binary it is not possible to use a scatter plot
for feature selection

#(part2)feature selection

#######################principal componenet analysis for feature
selection##########################################
install.packages("corrr")#installing of package for creating and
handling dataframes
```

```
## Installing package into '/home/vaasala/R/x86_64-pc-linux-gnu-
library/4.3'
## (as 'lib' is unspecified)
```

```r
library('corrr')#importing the library

install.packages("ggcorrplot")#package for visualization of
correlation matrix
```

```
## Installing package into '/home/vaasala/R/x86_64-pc-linux-gnu-
library/4.3'
## (as 'lib' is unspecified)
```

```r
library(ggcorrplot)
```

```
## Loading required package: ggplot2
```

```r
install.packages("FactoMineR")#package used for multivariate data
analsyis
```

```
## Installing package into '/home/vaasala/R/x86_64-pc-linux-gnu-
library/4.3'
## (as 'lib' is unspecified)
```

```r
library("FactoMineR")

str(data2)#the column and data types
```

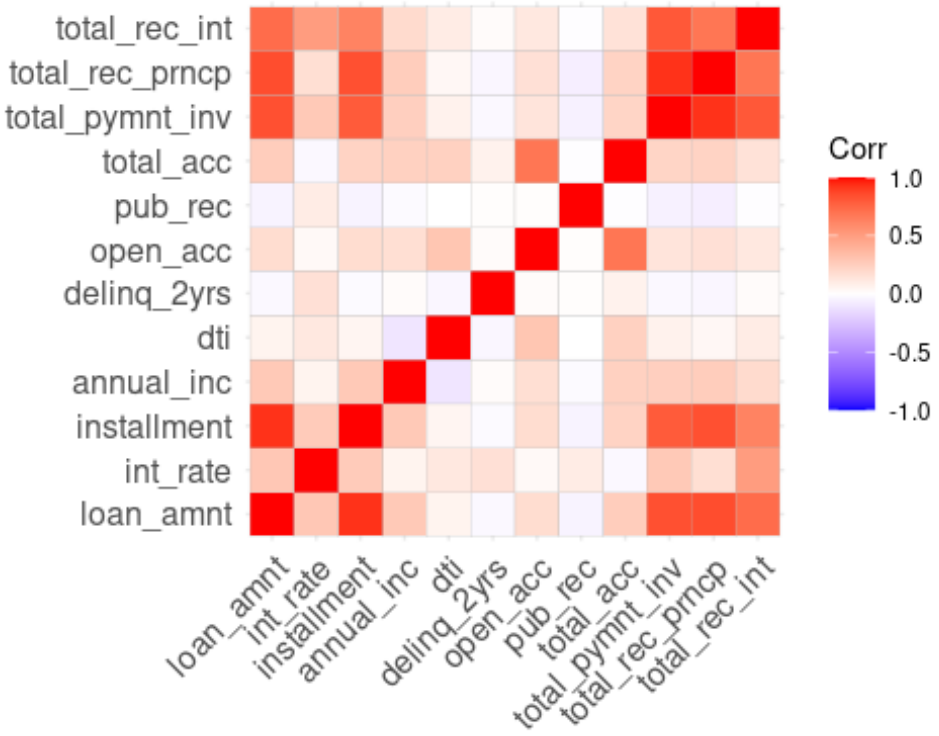
```
## 'data.frame':    38419 obs. of  21 variables:
##  $ X              : int  2 3 4 5 6 7 8 9 10 11 ...
##  $ loan_amnt      : int  0 2500 5000 7000 2000 3600 8000 6000 25600
19750 ...
##  $ term           : chr  "36 months" "36 months" "36 months" "36
months" ...
##  $ int_rate       : num  0 13.98 15.95 9.91 5.42 ...
##  $ installment    : num  0 85.4 175.7 225.6 60.3 ...
##  $ grade          : chr  "A" "C" "D" "B" ...
##  $ emp_length     : chr  "< 1 year" "4 years" "4 years" "10+ years"
...
##  $ home_ownership : chr  "RENT" "RENT" "RENT" "MORTGAGE" ...
##  $ annual_inc     : num  0 20004 59000 53796 30000 ...
##  $ loan_status    : chr  "Charged Off" "Does not meet the credit
```

```
policy. Status:Fully Paid" "Charged Off" "Fully Paid" ...
##  $ purpose        : chr  "major_purchase" "other"
"debt_consolidation" "other" ...
##  $ dti            : num  0 19.9 19.6 10.8 3.6 ...
##  $ delinq_2yrs    : int  0 0 0 3 0 0 0 0 0 0 ...
##  $ open_acc       : int  0 7 7 7 7 8 12 5 16 15 ...
##  $ pub_rec        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ revol_util     : chr  "0.00%" "0.213" "0.999" "0.472" ...
##  $ total_acc      : int  1 10 15 20 15 25 49 9 32 44 ...
##  $ total_pymnt_inv: num  0 3075 2949 8082 2162 ...
##  $ total_rec_prncp: num  0 2500 1909 7000 2000 ...
##  $ total_rec_int  : num  0 575 874 1082 162 ...
##  $ repay_fail     : int  1 0 1 0 0 0 0 0 0 0 ...
##  - attr(*, "na.action")= 'omit' Named int [1:61] 1 1124 3070 3541
3647 3669 3887 4261 5132 6115 ...
##   ..- attr(*, "names")= chr [1:61] "1" "1124" "3070" "3541" ...

colSums(is.na(data2))#number of missing values present in dataset

##               X       loan_amnt            term        int_rate
installment
##               0               0               0               0
0
##           grade      emp_length  home_ownership      annual_inc
loan_status
##               0               0               0               0
0
##         purpose             dti     delinq_2yrs        open_acc
pub_rec
##               0               0               0               0
0
##      revol_util       total_acc total_pymnt_inv total_rec_prncp
total_rec_int
##               0               0               0               0
0
##      repay_fail
##               0

#there are no missing values

#principal component analysis can only be applied to numerical data so
the categorical columns must be removed.

data3 <- subset(data2, select = -c(X, term,
grade,emp_length,home_ownership,loan_status,purpose,revol_util,repay_f
ail))
#repay_fail column also removed because it is the dependent variable
str(data3)

## 'data.frame':    38419 obs. of  12 variables:
##  $ loan_amnt      : int  0 2500 5000 7000 2000 3600 8000 6000 25600
```

```
19750 ...
##  $ int_rate       : num  0 13.98 15.95 9.91 5.42 ...
##  $ installment    : num  0 85.4 175.7 225.6 60.3 ...
##  $ annual_inc     : num  0 20004 59000 53796 30000 ...
##  $ dti            : num  0 19.9 19.6 10.8 3.6 ...
##  $ delinq_2yrs    : int  0 0 0 3 0 0 0 0 0 0 ...
##  $ open_acc       : int  0 7 7 7 7 8 12 5 16 15 ...
##  $ pub_rec        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ total_acc      : int  1 10 15 20 15 25 49 9 32 44 ...
##  $ total_pymnt_inv: num  0 3075 2949 8082 2162 ...
##  $ total_rec_prncp: num  0 2500 1909 7000 2000 ...
##  $ total_rec_int  : num  0 575 874 1082 162 ...

head(data3)

##   loan_amnt int_rate installment annual_inc   dti delinq_2yrs
open_acc pub_rec
## 2         0     0.00        0.00          0  0.00           0
0       0
## 3      2500    13.98       85.42      20004 19.86           0
7       0
## 4      5000    15.95      175.67      59000 19.57           0
7       0
## 5      7000     9.91      225.58      53796 10.80           3
7       0
## 6      2000     5.42       60.32      30000  3.60           0
7       0
## 7      3600    10.25      116.59     675048  1.55           0
8       0
##   total_acc total_pymnt_inv total_rec_prncp total_rec_int
## 2         1            0.00            0.00          0.00
## 3        10         3075.29         2500.00        575.29
## 4        15         2948.76         1909.02        873.81
## 5        20         8082.39         7000.00       1082.39
## 6        15         2161.66         2000.00        161.66
## 7        25         4206.03         3600.00        606.03

#normalizing the data
data3_normalized <- scale(data3)
head(data3_normalized)

##    loan_amnt   int_rate installment annual_inc        dti
delinq_2yrs
## 2 -1.4994346 -3.2776629   -1.546672 -1.0707622 -1.9882696  -
0.2994498
## 3 -1.1618471  0.4914804   -1.138147 -0.7604690  0.9630919  -
0.2994498
## 4 -0.8242597  1.0226115   -0.706522 -0.1555803  0.9199954  -
0.2994498
## 5 -0.5541897 -0.6058310   -0.467825 -0.2363025 -0.3832996
5.6124525
```

```
## 6 -1.2293646 -1.8163784   -1.258189 -0.6054155 -1.4532796  -
0.2994498
## 7 -1.0133087 -0.5141637   -0.989075  9.4002834 -1.7579267  -
0.2994498
##     open_acc    pub_rec  total_acc total_pymnt_inv total_rec_prncp
## 2 -2.0799101 -0.2357255 -1.8239688      -1.2614572      -1.3691975
## 3 -0.5228984 -0.2357255 -1.0469715      -0.9177389      -1.0146818
## 4 -0.5228984 -0.2357255 -0.6153063      -0.9318809      -1.0984865
## 5 -0.5228984 -0.2357255 -0.1836411      -0.3581065      -0.3765534
## 6 -0.5228984 -0.2357255 -0.6153063      -1.0198533      -1.0855849
## 7 -0.3004681 -0.2357255  0.2480241      -0.7913586      -0.8586948
##   total_rec_int
## 2    -0.8690454
## 3    -0.6453254
## 4    -0.5292362
## 5    -0.4481232
## 6    -0.8061787
## 7    -0.6333711
```

```
#plotting of the covariance matrix
corr_matrix <- cor(data3_normalized) #calculation of correlations for
the covariance matrix
ggcorrplot(corr_matrix) #covariance plot of the features
```




```
#applying principle component analysis
data3_pca <- princomp(corr_matrix)
summary(data3_pca)
```

```
## Importance of components:
##                             Comp.1    Comp.2     Comp.3     Comp.4
Comp.5
## Standard deviation     0.8521109 0.4534905 0.33037942 0.28757739
0.22327073
## Proportion of Variance 0.5877865 0.1664807 0.08835952 0.06694789
0.04035441
## Cumulative Proportion  0.5877865 0.7542672 0.84262671 0.90957460
0.94992900
##                             Comp.6      Comp.7      Comp.8
Comp.9
## Standard deviation     0.20177284 0.102337390 0.084900170
0.055093005
## Proportion of Variance 0.03295739 0.008478052 0.005835049
0.002457086
## Cumulative Proportion  0.98288639 0.991364444 0.997199493
0.999656579
##                              Comp.10      Comp.11 Comp.12
## Standard deviation     0.0161308844 0.0128071282       0
## Proportion of Variance 0.0002106414 0.0001327795       0
## Cumulative Proportion  0.9998672205 1.0000000000       1
```

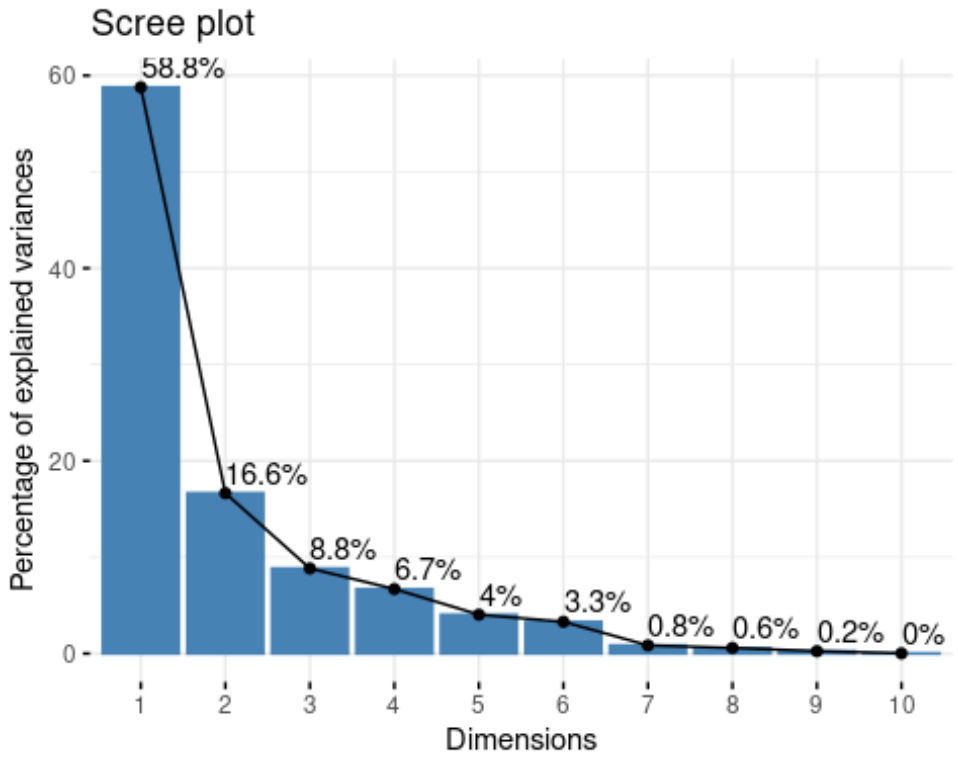
```
#the first six components can accurately present 95.8 percent of the
data, so we select the 1st 6 components
install.packages("factoextra")
```

```
## Installing package into '/home/vaasala/R/x86_64-pc-linux-gnu-
library/4.3'
## (as 'lib' is unspecified)
```

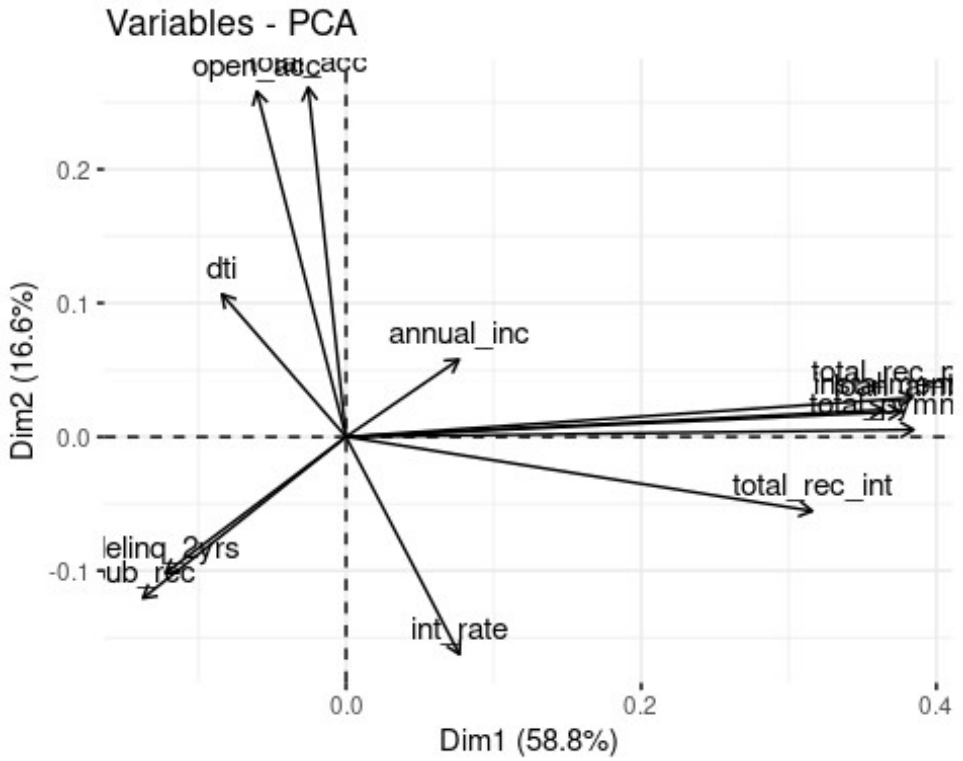
```
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at
https://goo.gl/ve3WBa
```

```
fviz_eig(data3_pca, addlabels = TRUE)#visualization of principal
components
```



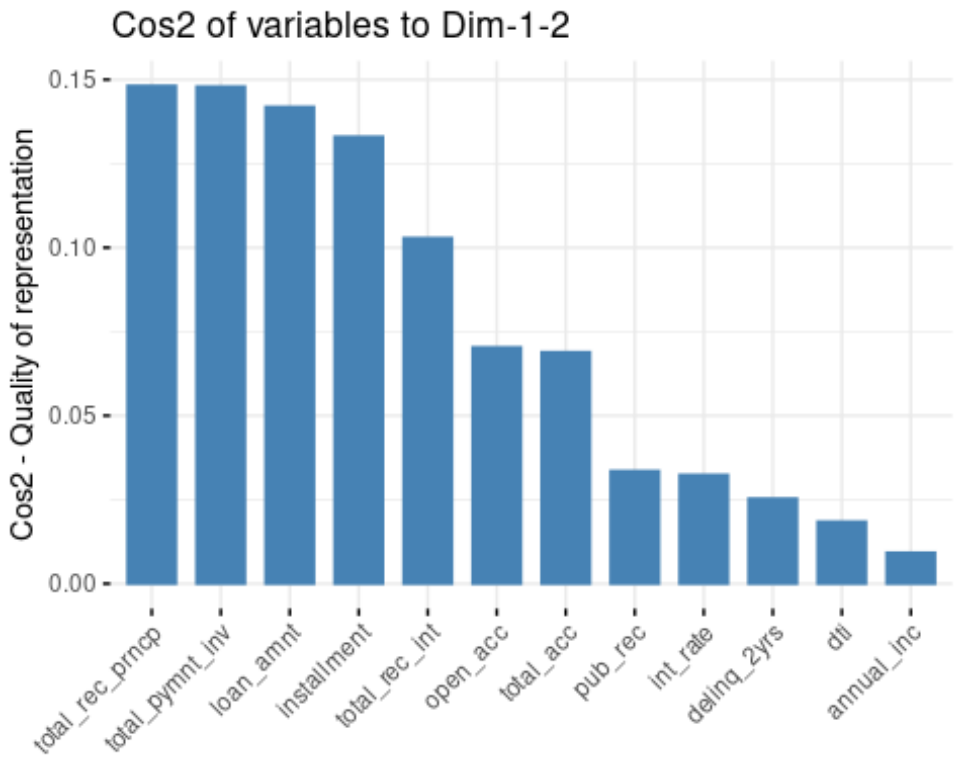

```r
# Graph of the variables
fviz_pca_var(data3_pca, col.var = "black")
```



```
head(data3)
```

```
##   loan_amnt int_rate installment annual_inc   dti delinq_2yrs
open_acc pub_rec
## 2         0     0.00        0.00          0  0.00           0
0       0
## 3      2500    13.98       85.42      20004 19.86           0
7       0
## 4      5000    15.95      175.67      59000 19.57           0
7       0
## 5      7000     9.91      225.58      53796 10.80           3
7       0
## 6      2000     5.42       60.32      30000  3.60           0
7       0
## 7      3600    10.25      116.59     675048  1.55           0
8       0
##   total_acc total_pymnt_inv total_rec_prncp total_rec_int
## 2         1            0.00            0.00          0.00
## 3        10         3075.29         2500.00        575.29
## 4        15         2948.76         1909.02        873.81
## 5        20         8082.39         7000.00       1082.39
## 6        15         2161.66         2000.00        161.66
## 7        25         4206.03         3600.00        606.03
```

```
#contribution of each variable to the major principle components
fviz_cos2(data3_pca, choice = "var", axes = 1:2)
```



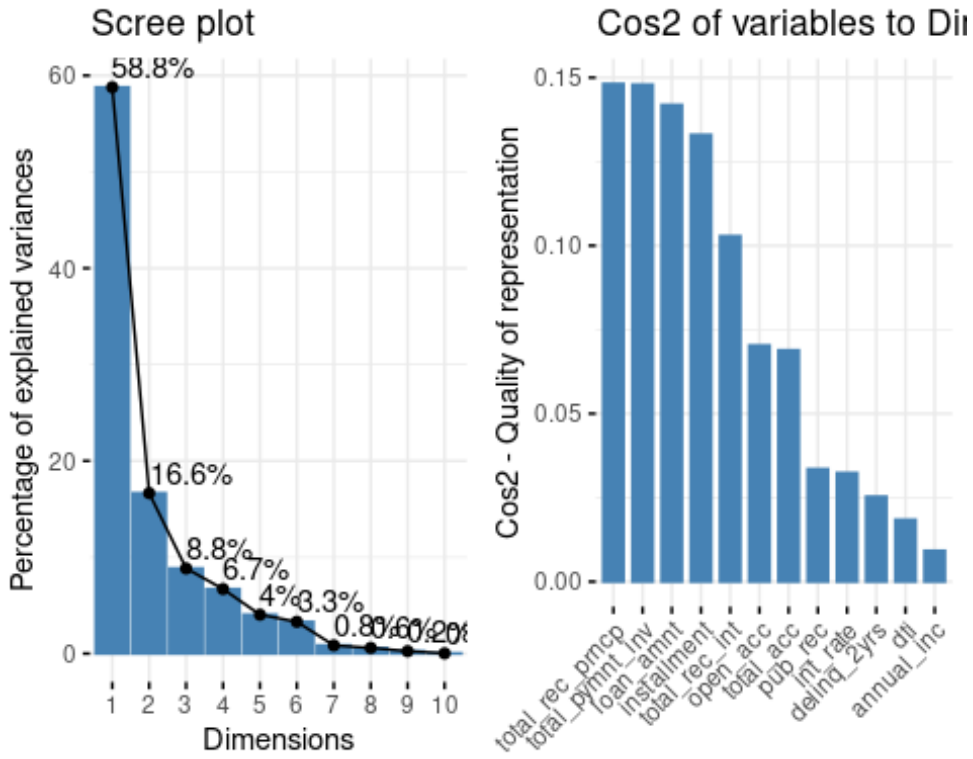
```r
#removing the columns with the least contribution
data4 <- subset(data3, select = -
c(annual_inc,dti,int_rate,pub_rec,total_acc,open_acc,delinq_2yrs))
head(data4)
```

```
##   loan_amnt installment total_pymnt_inv total_rec_prncp
total_rec_int
## 2         0        0.00            0.00            0.00
0.00
## 3      2500       85.42         3075.29         2500.00
575.29
## 4      5000      175.67         2948.76         1909.02
873.81
## 5      7000      225.58         8082.39         7000.00
1082.39
## 6      2000       60.32         2161.66         2000.00
161.66
## 7      3600      116.59         4206.03         3600.00
606.03
```

```r
# 3 view plot of principle componenet analysis
install.packages("gridExtra")
```

```
## Installing package into '/home/vaasala/R/x86_64-pc-linux-gnu-
library/4.3'
## (as 'lib' is unspecified)
```

```r
library("gridExtra")
grid.arrange(fviz_eig(data3_pca, addlabels = TRUE),
             fviz_cos2(data3_pca, choice = "var", axes = 1:2),
             ncol = 2)
```




```
#(part3) fitting the model(pca)
str(data2)

## 'data.frame':    38419 obs. of  21 variables:
##  $ X              : int  2 3 4 5 6 7 8 9 10 11 ...
##  $ loan_amnt      : int  0 2500 5000 7000 2000 3600 8000 6000 25600
19750 ...
##  $ term           : chr  "36 months" "36 months" "36 months" "36
months" ...
##  $ int_rate       : num  0 13.98 15.95 9.91 5.42 ...
##  $ installment    : num  0 85.4 175.7 225.6 60.3 ...
##  $ grade          : chr  "A" "C" "D" "B" ...
##  $ emp_length     : chr  "< 1 year" "4 years" "4 years" "10+ years"
...
##  $ home_ownership : chr  "RENT" "RENT" "RENT" "MORTGAGE" ...
##  $ annual_inc     : num  0 20004 59000 53796 30000 ...
##  $ loan_status    : chr  "Charged Off" "Does not meet the credit
policy. Status:Fully Paid" "Charged Off" "Fully Paid" ...
##  $ purpose        : chr  "major_purchase" "other"
"debt_consolidation" "other" ...
##  $ dti            : num  0 19.9 19.6 10.8 3.6 ...
##  $ delinq_2yrs    : int  0 0 0 3 0 0 0 0 0 0 ...
##  $ open_acc       : int  0 7 7 7 7 8 12 5 16 15 ...
##  $ pub_rec        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ revol_util     : chr  "0.00%" "0.213" "0.999" "0.472" ...
##  $ total_acc      : int  1 10 15 20 15 25 49 9 32 44 ...
##  $ total_pymnt_inv: num  0 3075 2949 8082 2162 ...
```

```
##  $ total_rec_prncp: num  0 2500 1909 7000 2000 ...
##  $ total_rec_int  : num  0 575 874 1082 162 ...
##  $ repay_fail     : int  1 0 1 0 0 0 0 0 0 0 ...
##  - attr(*, "na.action")= 'omit' Named int [1:61] 1 1124 3070 3541
3647 3669 3887 4261 5132 6115 ...
##   ..- attr(*, "names")= chr [1:61] "1" "1124" "3070" "3541" ...

data4$repay_fail=data2["repay_fail"]
# changing repay_fail column to be not nested
str(data4)

## 'data.frame':    38419 obs. of  6 variables:
##  $ loan_amnt      : int  0 2500 5000 7000 2000 3600 8000 6000 25600
19750 ...
##  $ installment    : num  0 85.4 175.7 225.6 60.3 ...
##  $ total_pymnt_inv: num  0 3075 2949 8082 2162 ...
##  $ total_rec_prncp: num  0 2500 1909 7000 2000 ...
##  $ total_rec_int  : num  0 575 874 1082 162 ...
##  $ repay_fail     :'data.frame': 38419 obs. of  1 variable:
##   ..$ repay_fail: int  1 0 1 0 0 0 0 0 0 0 ...

data4$repay_fail <- unlist(data4$repay_fail$repay_fail)

head(data4)

##   loan_amnt installment total_pymnt_inv total_rec_prncp
total_rec_int
## 2         0        0.00            0.00            0.00
0.00
## 3      2500       85.42         3075.29         2500.00
575.29
## 4      5000      175.67         2948.76         1909.02
873.81
## 5      7000      225.58         8082.39         7000.00
1082.39
## 6      2000       60.32         2161.66         2000.00
161.66
## 7      3600      116.59         4206.03         3600.00
606.03
##   repay_fail
## 2          1
## 3          0
## 4          1
## 5          0
## 6          0
## 7          0

data4 <- data.frame(data4)
str(data4)
```

```
## 'data.frame':    38419 obs. of  6 variables:
##  $ loan_amnt      : int  0 2500 5000 7000 2000 3600 8000 6000 25600
19750 ...
##  $ installment    : num  0 85.4 175.7 225.6 60.3 ...
##  $ total_pymnt_inv: num  0 3075 2949 8082 2162 ...
##  $ total_rec_prncp: num  0 2500 1909 7000 2000 ...
##  $ total_rec_int  : num  0 575 874 1082 162 ...
##  $ repay_fail     : int  1 0 1 0 0 0 0 0 0 0 ...
```

```
#splitting the data into training and testing- the caret package
install.packages("caret")
```

```
## Installing package into '/home/vaasala/R/x86_64-pc-linux-gnu-
library/4.3'
## (as 'lib' is unspecified)
```

```
library("caret")
```

```
## Loading required package: lattice
```

```
#partitioning data frame into training and testing sets
head(data4)
```

```
##   loan_amnt installment total_pymnt_inv total_rec_prncp
total_rec_int
## 2         0        0.00            0.00            0.00
0.00
## 3      2500       85.42         3075.29         2500.00
575.29
## 4      5000      175.67         2948.76         1909.02
873.81
## 5      7000      225.58         8082.39         7000.00
1082.39
## 6      2000       60.32         2161.66         2000.00
161.66
## 7      3600      116.59         4206.03         3600.00
606.03
##   repay_fail
## 2          1
## 3          0
## 4          1
## 5          0
## 6          0
## 7          0
```

```
col <- c("loan_amnt","installment","total_pymnt_inv",
"total_rec_prncp", "total_rec_int", "repay_fail")#assigning values to
col object for iteration
all_indices <- NULL #assigning the all_indices vector as null object
for (i in col) { #for loop for partitioning each column
  train_indices <- createDataPartition(unlist(data4[i]), times=1,
p=.6, list=FALSE) #partition each column 80% as training set
```

```r
  all_indices <- union(all_indices,train_indices) #combine train
indices columns together in each iteration of i
}
#create training set
data4_train <- data4[all_indices , ]#selecting the rows for train
indices

#create testing set
data4_test  <- data4[-all_indices, ]#selecting the rows for test
indices by reducing train indices

#view number of rows in each set
data4_train <- data.frame(data4_train)
data4_test <- data.frame(data4_test)

#observation of data4_train
head(data4_train)

##    loan_amnt installment total_pymnt_inv total_rec_prncp
total_rec_int
## 3       2500       85.42         3075.29         2500.00
575.29
## 4       5000      175.67         2948.76         1909.02
873.81
## 6       2000       60.32         2161.66         2000.00
161.66
## 8       8000      243.49         8724.97         8000.00
724.97
## 9       6000      186.61         6717.95         6000.00
717.95
## 10     25600      599.26        32659.13        25600.00
7240.06
##    repay_fail
## 3           0
## 4           1
## 6           0
## 8           0
## 9           0
## 10          0

nrow(data4_train)

## [1] 38226

str(data4_train)

## 'data.frame':    38226 obs. of  6 variables:
##  $ loan_amnt      : int  2500 5000 2000 8000 6000 25600 6250 9000
4500 10500 ...
##  $ installment    : num  85.4 175.7 60.3 243.5 186.6 ...
##  $ total_pymnt_inv: num  3075 2949 2162 8725 6718 ...
```

```
##  $ total_rec_prncp: num  2500 1909 2000 8000 6000 ...
##  $ total_rec_int  : num  575 874 162 725 718 ...
##  $ repay_fail     : int  0 1 0 0 0 0 1 0 0 0 ...

#observation of data4_test
head(data4_test)

##      loan_amnt installment total_pymnt_inv total_rec_prncp
total_rec_int
## 140      12800      427.72        11334.69         8357.55
2318.62
## 502       2000       42.87         2411.67         2000.00
411.67
## 590       8900      189.86         6742.15         6225.00
523.23
## 797      35000      961.50        15297.39         6005.16
9355.32
## 1213     22750      577.09        20355.34        16775.00
3702.41
## 1351      1500       49.75         1731.54         1500.00
290.82
##      repay_fail
## 140           1
## 502           0
## 590           0
## 797           1
## 1213          0
## 1351          0

nrow(data4_test)

## [1] 193

str(data4_test)

## 'data.frame':    193 obs. of  6 variables:
##  $ loan_amnt      : int  12800 2000 8900 35000 22750 1500 6000 5000
7500 12500 ...
##  $ installment    : num  427.7 42.9 189.9 961.5 577.1 ...
##  $ total_pymnt_inv: num  11335 2412 6742 15297 20355 ...
##  $ total_rec_prncp: num  8358 2000 6225 6005 16775 ...
##  $ total_rec_int  : num  2319 412 523 9355 3702 ...
##  $ repay_fail     : int  1 0 0 1 0 0 0 0 0 0 ...

#utilizing the logit link function
model_logit <- glm(repay_fail ~ loan_amnt + installment +
total_pymnt_inv + total_rec_prncp + total_rec_int,
                   family = binomial(link = "logit"), #linkage
                   data = data4_train) #data

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
#utilizing the probit link function
model_probit <- glm(repay_fail ~ loan_amnt + installment +
total_pymnt_inv + total_rec_prncp + total_rec_int,
                    family = binomial(link = "probit"), #linkage
                    data = data4_train) #data

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

#utilizing the complementary log log(cloglog) link function
model_cloglog <- glm(repay_fail ~ loan_amnt + installment +
total_pymnt_inv + total_rec_prncp + total_rec_int,
                     family = binomial(link = "cloglog"), #linkage
                     data = data4_train) #data

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

#(part4)validation

##########################################
#holdout validation for model selection using testing dataset


#logit


predictions1 <- predict(model_logit, data4_test)

# predicting the target variable
print(predictions1)

##         140         502         590         797        1213
1351
##   9.2706443  -4.1247190  -4.1977578  75.4194935  -2.5305342  -
2.3202532
##        2393        2501        2671        2893        3113
3264
##  -3.8355939  -4.4756823  -7.1224448  -8.4502704  -3.0417280 -
14.6862220
##        3411        3632        3731        3785        3789
4142
##  -5.7426216  -4.8254636  -2.6341456 -20.8732451  -3.2200314  -
9.3868562
##        4586        4711        4808        5064        5099
5177
##  -2.9599825  -3.2083643  -2.7551233  36.5691842  -3.9667741 -
13.1517186
##        5266        5662        5753        5927        5973
5988
##  24.9684368  -2.1579897 -11.8860876  12.8660614  -9.8336269  -
```

```
5.3995148
##        6232        6329        6371        6385        7253
7457
##  -4.1874755  14.9728796  -5.8130321  -7.6542991 -11.7676160
31.8676479
##        7459        7715        7781        7854        7980
8293
##  -4.1032283  -6.3663807  -6.2065095  -5.5005031  11.6619882  -
7.6098219
##        8360        9111        9333        9353        9359
9507
##  -7.0332830   2.5099091  -6.8369178 -13.6839433 -14.7071295  -
6.5089092
##        9629       10077       10145       10191       10497
10652
##  -5.4471197 -19.2966049  -8.1615038  -5.7192716  -6.3748982 -
10.9668723
##       11062       11138       11718       11734       11807
12045
##  -9.0646340   4.0911873  20.4418892  19.6929665  -5.2910459  -
3.7554922
##       12062       13249       13281       13509       13526
13532
##  -5.9715559  -5.6505304  -6.0845491  -8.8592058  -5.3011798  -
5.4852887
##       13744       13771       13786       13939       14127
15106
##  -6.1107177  -2.8892770  -6.9566397  -2.3657356 -10.2917890 -
32.7661981
##       15473       15519       15893       16039       16133
16142
##  -7.5580775  -2.9198963  -6.7257195   7.1272177  15.0577252  -
4.5370417
##       16507       16640       16730       16738       17005
17293
##   6.0865451  54.5216955  -4.3291498  -8.0398538 -13.2834373  -
3.2587144
##       17770       17835       17907       17980       18248
18603
##  -2.9212912  -9.8152079 -31.5221487   1.7823059  -5.0785401  -
3.4359626
##       18650       18857       19040       19295       19711
19753
##  -4.1877446 -17.7964633  -2.6826255  -4.7655060  -7.1258086  -
3.1961072
##       20296       20406       20688       20726       20746
20769
##  -6.7716413 -10.0837292  -0.2909281  -4.5646156 -12.8276629  -
6.0469728
##       20864       20961       21294       21900       22098
```

```
22209
##  -9.2346810 -12.5620184  -2.1755154  -8.9068577 -20.4480020  -
3.0138975
##       22338       22633       22668       23182       23510
23642
## -10.4411664  -5.9832080  -7.1609497 -12.6277678  -3.2641054  -
6.1525730
##       23927       23943       23976       24031       24125
24184
##  -7.0473776 -14.7992124  18.8723099 -10.8473878  -5.2375477  -
3.4615096
##       24839       25041       25219       25284       25515
25529
##  13.1826037  -2.6522048   3.2898882  -2.4639177  -5.3157536 -
10.8020503
##       25618       25879       26036       26210       26422
27102
## -13.6364586  -2.2668199  -4.7341246 -17.9008372  11.9504404 -
10.1641774
##       27336       27550       27581       27668       28048
28120
##  -3.3756442  -7.6874712  45.7040574  -2.5090754  -3.6373153  -
6.1856847
##       28444       28565       28575       28613       29328
29501
##   0.1603372  -8.7133223  -8.0734225  -4.2603834  -3.4415838  -
4.2167028
##       29834       30017       30064       30140       30445
30616
##  -0.2592817  -3.1994937  -4.5554383  26.5415713  -8.9730748
8.0307697
##       30777       30895       31048       31113       31824
33031
## -34.9147717 -10.9410051 -15.9210982  16.9183806  -7.3483869 -
12.6014628
##       33051       33066       33172       33403       33428
33627
##  20.6089644  -2.7571037 -11.8502944  -3.3333153  -4.8225988  -
4.8386027
##       34183       34579       34629       34816       35011
35056
##  -3.3582746  -8.4094921  -9.4867425  -5.6742250 -12.6753112  -
3.3942424
##       35090       35166       35187       35237       35467
35545
##  -6.3601079  -4.2817417  -4.4647454   5.6451791  -3.8392457
33.4433036
##       35691       35814       36003       36267       36355
36415
##  -6.8015948  -3.0920879  -4.3098488 -11.1287057  20.5431051 -
```

```
12.8322915
##       36565       36626       36658       36961       37019
37251
##  -4.8376705  -6.6126002  -7.9529577  37.4084673 -17.7354229  -
6.1178985
##       37327       37362       37414       37735       37825
37866
##  -6.0380503  -5.2187241  -2.8799227  38.4190636  -8.7903540 -
16.1596858
##       37894
##  -9.6877487

#omitting any missing values
predictions1 <- na.omit(unlist(predictions1))
data4_testvec <- na.omit(unlist(data4_test$repay_fail))

#normalization of predictions1
predictions1 <- scale(predictions1)
print(predictions1)

##                [,1]
## 140    9.094876e-01
## 502   -9.328351e-02
## 590   -9.875116e-02
## 797    5.861362e+00
## 1213   2.605647e-02
## 1351   4.179802e-02
## 2393  -7.163973e-02
## 2501  -1.195565e-01
## 2671  -3.176920e-01
## 2893  -4.170924e-01
## 3113  -1.221127e-02
## 3264  -8.839130e-01
## 3411  -2.143990e-01
## 3632  -1.457409e-01
## 3731   1.830017e-02
## 3785  -1.347071e+00
## 3789  -2.555898e-02
## 4142  -4.872048e-01
## 4586  -6.091839e-03
## 4711  -2.468559e-02
## 4808   9.243829e-03
## 5064   2.953044e+00
## 5099  -8.145982e-02
## 5177  -7.690408e-01
## 5266   2.084618e+00
## 5662   5.394500e-02
## 5753  -6.742962e-01
## 5927   1.178639e+00
## 5973  -5.206499e-01
```

```
## 5988  -1.887142e-01
## 6232  -9.798143e-02
## 6329   1.336354e+00
## 6371  -2.196699e-01
## 6385  -3.575063e-01
## 7253  -6.654275e-01
## 7457   2.601089e+00
## 7459  -9.167472e-02
## 7715  -2.610933e-01
## 7781  -2.491254e-01
## 7854  -1.962741e-01
## 7980   1.088503e+00
## 8293  -3.541768e-01
## 8360  -3.110173e-01
## 9111   4.033819e-01
## 9333  -2.963175e-01
## 9353  -8.088829e-01
## 9359  -8.854782e-01
## 9507  -2.717630e-01
## 9629  -1.922779e-01
## 10077 -1.229044e+00
## 10145 -3.954755e-01
## 10191 -2.126510e-01
## 10497 -2.617309e-01
## 10652 -6.054841e-01
## 11062 -4.630834e-01
## 11138  5.217557e-01
## 11718  1.745762e+00
## 11734  1.689698e+00
## 11807 -1.805942e-01
## 12045 -6.564335e-02
## 12062 -2.315369e-01
## 13249 -2.075051e-01
## 13281 -2.399955e-01
## 13509 -4.477051e-01
## 13526 -1.813529e-01
## 13532 -1.951352e-01
## 13744 -2.419545e-01
## 13771 -7.988529e-04
## 13786 -3.052799e-01
## 13939  3.839323e-02
## 14127 -5.549477e-01
## 15106 -2.237372e+00
## 15473 -3.503032e-01
## 15519 -3.091001e-03
## 15893 -2.879933e-01
## 16039  7.490316e-01
## 16133  1.342706e+00
## 16142 -1.241498e-01
## 16507  6.711273e-01
```

```
## 16640  4.296962e+00
## 16730 -1.085871e-01
## 16738 -3.863688e-01
## 17005 -7.789012e-01
## 17293 -2.845478e-02
## 17770 -3.195426e-03
## 17835 -5.192710e-01
## 17907 -2.144243e+00
## 17980  3.489138e-01
## 18248 -1.646861e-01
## 18603 -4.172350e-02
## 18650 -9.800157e-02
## 18857 -1.116744e+00
## 19040  1.467098e-02
## 19295 -1.412525e-01
## 19711 -3.179438e-01
## 19753 -2.376803e-02
## 20296 -2.914310e-01
## 20406 -5.393724e-01
## 20688  1.937124e-01
## 20726 -1.262140e-01
## 20746 -7.447821e-01
## 20769 -2.371826e-01
## 20864 -4.758130e-01
## 20961 -7.248961e-01
## 21294  5.263303e-02
## 21900 -4.512723e-01
## 22098 -1.315237e+00
## 22209 -1.012790e-02
## 22338 -5.661300e-01
## 22633 -2.324092e-01
## 22668 -3.205744e-01
## 23182 -7.298181e-01
## 23510 -2.885835e-02
## 23642 -2.450878e-01
## 23927 -3.120725e-01
## 23943 -8.923715e-01
## 23976  1.628264e+00
## 24031 -5.965396e-01
## 24125 -1.765894e-01
## 24184 -4.363594e-02
## 24839  1.202335e+00
## 25041  1.694827e-02
## 25219  4.617708e-01
## 25284  3.104336e-02
## 25515 -1.824438e-01
## 25529 -5.931456e-01
## 25618 -8.053282e-01
## 25879  4.579801e-02
## 26036 -1.389033e-01
```

```
## 26210 -1.124558e+00
## 26422  1.110096e+00
## 27102 -5.453947e-01
## 27336 -3.720809e-02
## 27550 -3.599896e-01
## 27581  3.636877e+00
## 27668  2.766287e-02
## 28048 -5.679668e-02
## 28120 -2.475665e-01
## 28444  2.274939e-01
## 28565 -4.367844e-01
## 28575 -3.888817e-01
## 28613 -1.034393e-01
## 29328 -4.214430e-02
## 29501 -1.001694e-01
## 29834  1.960815e-01
## 30017 -2.402154e-02
## 30064 -1.255270e-01
## 30140  2.202382e+00
## 30445 -4.562293e-01
## 30616  8.166711e-01
## 30777 -2.398214e+00
## 30895 -6.035477e-01
## 31048 -9.763553e-01
## 31113  1.481994e+00
## 31824 -3.346059e-01
## 33031 -7.278489e-01
## 33051  1.758269e+00
## 33066  9.095583e-03
## 33172 -6.716167e-01
## 33403 -3.403937e-02
## 33428 -1.455265e-01
## 33627 -1.467245e-01
## 34183 -3.590781e-02
## 34579 -4.140398e-01
## 34629 -4.946823e-01
## 34816 -2.092789e-01
## 35011 -7.333771e-01
## 35056 -3.860035e-02
## 35090 -2.606238e-01
## 35166 -1.050382e-01
## 35187 -1.187377e-01
## 35237  6.380868e-01
## 35467 -7.191310e-02
## 35545  2.719042e+00
## 35691 -2.936733e-01
## 35814 -1.598119e-02
## 36003 -1.071422e-01
## 36267 -6.175989e-01
## 36355  1.753339e+00
```

```
## 36415 -7.451286e-01
## 36565 -1.466547e-01
## 36626 -2.795252e-01
## 36658 -3.798638e-01
## 36961  3.015873e+00
## 37019 -1.112175e+00
## 37251 -2.424921e-01
## 37327 -2.365147e-01
## 37362 -1.751803e-01
## 37414 -9.859517e-05
## 37735  3.091526e+00
## 37825 -4.425509e-01
## 37866 -9.942159e-01
## 37894 -5.097295e-01
## attr(,"scaled:center")
## [1] -2.878606
## attr(,"scaled:scale")
## [1] 13.35835
```

```
print(data4_testvec)
```

```
##   [1] 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0
1 0 0 0 1 0
##  [38] 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
##  [75] 0 1 1 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0
## [112] 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0
0 0 1 0 0 1
## [149] 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1
0 0 0 0 1 0
## [186] 0 0 0 0 1 0 0 0
```

```
# computing model performance metrics
df1 <- data.frame( R21 = R2(predictions1, data4_testvec), #R2 error
            RMSE1 = RMSE(predictions1, data4_testvec), #root mean
square error
            MAE1 = MAE(predictions1,data4_testvec)) #mean absolute
error

#probit

predictions2 <- predict(model_probit, data4_test)

# predicting the target variable
print(predictions2)
```

```
##         140         502         590         797        1213
1351
##   3.3606134  -2.0025601  -1.9548521  30.2375564  -1.1000199  -
1.3133606
```

```
##        2393        2501        2671        2893        3113
3264
##  -1.8954779  -2.1555498  -3.1391215  -3.4332836  -1.5276791  -
5.9503800
##        3411        3632        3731        3785        3789
4142
##  -2.6530670  -2.2391024  -1.4420511  -8.6242870  -1.6447702  -
4.0853621
##        4586        4711        4808        5064        5099
5177
##  -1.5573364  -1.6565415  -1.4849874  14.5117762  -1.9849868  -
5.1266065
##        5266        5662        5753        5927        5973
5988
##   9.8695769  -1.2443177  -5.0609323   5.3108723  -4.2485708  -
2.5537367
##        6232        6329        6371        6385        7253
7457
##  -2.0510432   5.7117019  -2.6902352  -3.3915957  -4.9920671
12.5823465
##        7459        7715        7781        7854        7980
8293
##  -2.0117827  -2.8298786  -2.4210284  -2.5204546   4.3443734  -
3.3513434
##        8360        9111        9333        9353        9359
9507
##  -3.1327875   0.6186157  -3.0574481  -5.6262095  -5.9695226  -
2.9430104
##        9629       10077       10145       10191       10497
10652
##  -2.4675527  -7.6519881  -3.6032145  -2.5083516  -2.8173813  -
4.5083605
##       11062       11138       11718       11734       11807
12045
##  -3.9075928   1.2175886   7.7681095   7.4488037  -2.4278898  -
1.8566631
##       12062       13249       13281       13509       13526
13532
##  -2.7365915  -2.6045224  -2.7666250  -3.8425584  -2.4821862  -
2.5501599
##       13744       13771       13786       13939       14127
15106
##  -2.7821393  -1.5152783  -3.0823136  -1.3281516  -3.7679372 -
12.5186943
##       15473       15519       15893       16039       16133
16142
##  -3.3642338  -1.5276478  -3.1068620   2.4424928   5.6332357  -
2.0903097
##       16507       16640       16730       16738       17005
17293
```

```
##   2.0702607  21.6325960  -2.0401271  -3.6004230  -5.3335519  -
1.6793899
##       17770       17835       17907       17980       18248
18603
##  -1.5486811  -4.2075858 -12.4768870   0.5143090  -2.4016021  -
1.3742938
##       18650       18857       19040       19295       19711
19753
##  -2.0377036  -7.3436263  -1.4562871  -2.2721703  -3.1813154  -
1.6747090
##       20296       20406       20688       20726       20746
20769
##  -3.0162363  -4.3202299  -0.5073647  -2.1606673  -5.4291544  -
2.7563238
##       20864       20961       21294       21900       22098
22209
##  -3.9843040  -5.1474388  -1.2597980  -3.8518920  -7.7911133  -
1.5820175
##       22338       22633       22668       23182       23510
23642
##  -4.4283505  -2.7034969  -3.2224761  -5.2813734  -1.7008881  -
2.7197497
##       23927       23943       23976       24031       24125
24184
##  -2.7656793  -6.0230825   7.4336944  -4.3783757  -2.4434664  -
1.7670749
##       24839       25041       25219       25284       25515
25529
##   5.1178082  -1.4369446   0.9751478  -1.3654999  -2.4553541  -
4.6250685
##       25618       25879       26036       26210       26422
27102
##  -5.4269025  -1.2943200  -2.2292964  -7.0792633   4.5271157  -
4.1295462
##       27336       27550       27581       27668       28048
28120
##  -1.7282715  -3.4295952  18.9102025  -1.3861471  -1.8258755  -
2.8317208
##       28444       28565       28575       28613       29328
29501
##  -0.3313231  -3.8648051  -3.5807350  -2.0375927  -1.7453305  -
2.0259157
##       29834       30017       30064       30140       30445
30616
##  -0.4836376  -1.6155967  -2.1995972  10.6548489  -3.5209371
2.8151716
##       30777       30895       31048       31113       31824
33031
## -13.7328516  -4.7599069  -6.2688158   6.6595229  -3.2481085  -
5.2764352
```

```
##       33051       33066       33172       33403       33428
33627
##   8.5219986  -1.4943869  -5.0385359  -1.7248602  -2.3010657  -
2.3049761
##       34183       34579       34629       34816       35011
35056
##  -1.6578375  -3.6717601  -3.8324092  -2.5664486  -5.3858372  -
1.7469905
##       35090       35166       35187       35237       35467
35545
##  -2.8807878  -1.9222126  -2.0689516   1.8526874  -1.8896492
12.9009978
##       35691       35814       36003       36267       36355
36415
##  -3.0884087  -1.5931817  -2.1223639  -4.3318204   7.8117998  -
5.2721408
##       36565       36626       36658       36961       37019
37251
##  -2.1945410  -2.9771034  -3.5017370  15.0127094  -7.1819950  -
2.8050722
##       37327       37362       37414       37735       37825
37866
##  -2.7239878  -2.4736459  -1.5196627  15.0992629  -3.8242873  -
6.6756813
##       37894
##  -3.8147708

#omitting any missing values
predictions2 <- na.omit(unlist(predictions2))
data4_testvec <- na.omit(unlist(data4_test$repay_fail))

#normalization of predictions2
predictions2 <- scale(predictions2)
print(predictions2)

##               [,1]
## 140    0.891785097
## 502   -0.111230754
## 590   -0.102308444
## 797    5.918286617
## 1213   0.057561500
## 1351   0.017662716
## 2393  -0.091204332
## 2501  -0.139842752
## 2671  -0.323789435
## 2893  -0.378803365
## 3113  -0.022418946
## 3264  -0.849548451
## 3411  -0.232887961
## 3632  -0.155468671
```

```
## 3731  -0.006404857
## 3785  -1.349620123
## 3789  -0.044317218
## 4142  -0.500754499
## 4586  -0.027965418
## 4711  -0.046518672
## 4808  -0.014434768
## 5064   2.977265455
## 5099  -0.107944201
## 5177  -0.695487082
## 5266   2.109085545
## 5662   0.030575047
## 5753  -0.683204746
## 5927   1.256520739
## 5973  -0.531277639
## 5988  -0.214311303
## 6232  -0.120298010
## 6329   1.331483539
## 6371  -0.239839135
## 6385  -0.371006936
## 7253  -0.670325634
## 7457   2.616425264
## 7459  -0.112955549
## 7715  -0.265955104
## 7781  -0.189492319
## 7854  -0.208086914
## 7980   1.075767003
## 8293  -0.363478989
## 8360  -0.322604847
## 9111   0.378979179
## 9333  -0.308514940
## 9353  -0.788922380
## 9359  -0.853128492
## 9507  -0.287112922
## 9629  -0.198193239
## 10077 -1.167781659
## 10145 -0.410583687
## 10191 -0.205823418
## 10497 -0.263617878
## 10652 -0.579863268
## 11062 -0.467508243
## 11138  0.490998541
## 11718  1.716071026
## 11734  1.656354734
## 11807 -0.190775525
## 12045 -0.083945224
## 12062 -0.248508634
## 13249 -0.223809203
## 13281 -0.254125470
## 13509 -0.455345569
```

```
## 13526 -0.200929992
## 13532 -0.213642361
## 13744 -0.257026956
## 13771 -0.020099758
## 13786 -0.313165281
## 13939  0.014896506
## 14127 -0.441389971
## 15106 -2.077948642
## 15473 -0.365889725
## 15519 -0.022413083
## 15893 -0.317756296
## 16039  0.720079010
## 16133  1.316808857
## 16142 -0.127641610
## 16507  0.650464498
## 16640  4.308994730
## 16730 -0.118256502
## 16738 -0.410061631
## 17005 -0.734189821
## 17293 -0.050791761
## 17770 -0.026346720
## 17835 -0.523612662
## 17907 -2.070129877
## 17980  0.359471818
## 18248 -0.185859216
## 18603  0.006267047
## 18650 -0.117803255
## 18857 -1.110112112
## 19040 -0.009067267
## 19295 -0.161653003
## 19711 -0.331680502
## 19753 -0.049916333
## 20296 -0.300807558
## 20406 -0.544679259
## 20688  0.168399341
## 20726 -0.140799813
## 20746 -0.752069314
## 20769 -0.252198952
## 20864 -0.481854704
## 20961 -0.699383111
## 21294  0.027679951
## 21900 -0.457091134
## 22098 -1.193800723
## 22209 -0.032581253
## 22338 -0.564899874
## 22633 -0.242319322
## 22668 -0.339378342
## 23182 -0.724431450
## 23510 -0.054812330
## 23642 -0.245358909
```

```
## 23927 -0.253948608
## 23943 -0.863145203
## 23976  1.653529021
## 24031 -0.555553625
## 24125 -0.193688657
## 24184 -0.067190531
## 24839  1.220414081
## 25041 -0.005449851
## 25219  0.445657477
## 25284  0.007911677
## 25515 -0.195911866
## 25529 -0.601689895
## 25618 -0.751648164
## 25879  0.021223675
## 26036 -0.153634775
## 26210 -1.060671182
## 26422  1.109943296
## 27102 -0.509017769
## 27336 -0.059933542
## 27550 -0.378113569
## 27581  3.799855152
## 27668  0.004050242
## 28048 -0.078187359
## 28120 -0.266299637
## 28444  0.201322475
## 28565 -0.459506120
## 28575 -0.406379600
## 28613 -0.117782522
## 29328 -0.063123899
## 29501 -0.115598692
## 29834  0.172836762
## 30017 -0.038861216
## 30064 -0.148080448
## 30140  2.255946408
## 30445 -0.395196244
## 30616  0.789777061
## 30777 -2.305019245
## 30895 -0.626907256
## 31048 -0.909102028
## 31113  1.508744170
## 31824 -0.344172083
## 33031 -0.723507899
## 33051  1.857062679
## 33066 -0.016192669
## 33172 -0.679016187
## 33403 -0.059295572
## 33428 -0.167056996
## 33627 -0.167788323
## 34183 -0.046761036
## 34579 -0.423403031
```

```
## 34629 -0.453447463
## 34816 -0.216688675
## 35011 -0.743968165
## 35056 -0.063434351
## 35090 -0.275476100
## 35166 -0.096204241
## 35187 -0.123647227
## 35237  0.609774140
## 35467 -0.090114256
## 35545  2.676019136
## 35691 -0.314305171
## 35814 -0.034669176
## 36003 -0.133636356
## 36267 -0.546846896
## 36355  1.724241944
## 36415 -0.722704779
## 36565 -0.147134848
## 36626 -0.293488949
## 36658 -0.391605458
## 36961  3.070949526
## 37019 -1.079883980
## 37251 -0.261315832
## 37327 -0.246151509
## 37362 -0.199332785
## 37414 -0.020919716
## 37735  3.087136682
## 37825 -0.451928514
## 37866 -0.985193645
## 37894 -0.450148759
## attr(,"scaled:center")
## [1] -1.407804
## attr(,"scaled:scale")
## [1] 5.347048

print(data4_testvec)

##   [1] 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0
1 0 0 0 1 0
##  [38] 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
##  [75] 0 1 1 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0
## [112] 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0
0 0 1 0 0 1
## [149] 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1
0 0 0 0 1 0
## [186] 0 0 0 0 1 0 0 0

# computing model performance metrics
df2 <- data.frame( R22 = R2(predictions2, data4_testvec), #R2 error
            RMSE2 = RMSE(predictions2, data4_testvec), #root mean
```

```
square error
            MAE2 = MAE(predictions2,data4_testvec)) #mean absolute
error

#cloglog
predictions3 <- predict(model_cloglog, data4_test)

# predicting the target variable
print(predictions3)
```

```
##           140           502           590           797
1213
##  1.065591e+15 -3.188628e+14 -5.654685e+14  1.120646e+16 -
1.031412e+15
##          1351          2393          2501          2671
2893
## -5.736446e+13 -6.386008e+14 -6.413651e+14 -1.257551e+15 -
1.594137e+15
##          3113          3264          3411          3632
3731
## -7.783694e+14 -2.423272e+15 -9.481514e+14 -1.266676e+15 -
1.017901e+14
##          3785          3789          4142          4586
4711
## -4.807300e+15 -5.351019e+14 -1.891959e+15 -3.118149e+14 -
3.431986e+14
##          4808          5064          5099          5177
5266
## -1.927856e+14  5.532038e+15 -5.090162e+14 -2.281858e+15
3.945045e+15
##          5662          5753          5927          5973
5988
## -7.542826e+13 -2.487148e+15  1.657695e+15 -2.116887e+15 -
1.051046e+15
##          6232          6329          6371          6385
7253
## -6.095581e+14  2.532420e+15 -1.221875e+15 -1.657618e+15 -
2.970304e+15
##          7457          7459          7715          7781
7854
##  5.011655e+15 -5.762673e+14 -1.801182e+15 -1.248308e+15 -
1.079832e+15
##          7980          8293          8360          9111
9333
##  9.358578e+14 -1.986602e+15 -1.566415e+15  5.551240e+14 -
1.475310e+15
##          9353          9359          9507          9629
10077
## -2.157820e+15 -2.421152e+15 -1.274149e+15 -1.591914e+15 -
3.625126e+15
```

```
##         10145         10191         10497         10652
11062
## -1.423796e+15 -1.665786e+15 -1.965337e+15 -1.767167e+15 -
2.338999e+15
##         11138         11718         11734         11807
12045
##  9.640831e+14  3.073827e+15  3.118433e+15 -1.325855e+15 -
6.231003e+14
##         12062         13249         13281         13509
13526
## -1.092857e+15 -1.049664e+15 -1.240494e+15 -1.944788e+15 -
8.367113e+14
##         13532         13744         13771         13786
13939
## -9.659951e+14 -1.280820e+15 -4.215975e+14 -1.333934e+15 -
1.041993e+14
##         14127         15106         15473         15519
15893
## -1.521757e+15 -6.357688e+15 -1.477230e+15 -4.258233e+14 -
1.423904e+15
##         16039         16133         16142         16507
16640
##  1.373007e+15  2.265235e+15 -1.466840e+15  8.754568e+14
8.393751e+15
##         16730         16738         17005         17293
17770
## -1.192350e+15 -1.578638e+15 -2.353447e+15 -3.101897e+14 -
2.218541e+14
##         17835         17907         17980         18248
18603
## -2.112815e+15 -5.663589e+15  4.259427e+14 -7.692393e+14 -
2.013629e+15
##         18650         18857         19040         19295
19711
## -6.024410e+14 -4.881139e+15 -1.480266e+14 -6.799081e+14 -
1.374386e+15
##         19753         20296         20406         20688
20726
## -2.494320e+14 -1.524864e+15 -2.696422e+15  2.879471e+14 -
9.717139e+14
##         20746         20769         20864         20961
21294
## -2.154895e+15 -1.207123e+15 -1.313990e+15 -2.158838e+15
3.482846e+12
##         21900         22098         22209         22338
22633
## -2.213101e+15 -4.010734e+15 -2.754824e+14 -2.699680e+15 -
1.883403e+15
##         22668         23182         23510         23642
23927
```

```
## -1.261182e+15 -2.040043e+15 -3.238319e+14 -2.016257e+15 -
1.372335e+15
##         23943         23976         24031         24125
24184
## -2.533775e+15  2.978016e+15 -1.862872e+15 -9.795661e+14 -
2.965336e+14
##         24839         25041         25219         25284
25515
##  2.169404e+15 -2.263741e+14  3.481447e+14 -1.399882e+14 -
1.165406e+15
##         25529         25618         25879         26036
26210
## -2.382497e+15 -2.451547e+15 -2.929831e+13 -9.022399e+14 -
3.430145e+15
##         26422         27102         27336         27550
27581
##  2.102450e+15 -1.720092e+15 -3.254985e+14 -1.373218e+15
6.368548e+15
##         27668         28048         28120         28444
28565
## -1.238815e+14 -4.433414e+14 -1.040592e+15  3.942395e+14 -
1.841114e+15
##         28575         28613         29328         29501
29834
## -1.819512e+15 -9.915565e+14 -4.236759e+14 -6.961049e+14
1.470168e+14
##         30017         30064         30140         30445
30616
## -7.097815e+14 -5.722446e+14  3.949163e+15 -1.208777e+15
1.417900e+15
##         30777         30895         31048         31113
31824
## -6.502047e+15 -2.275046e+15 -2.967996e+15  3.010196e+15 -
1.811196e+15
##         33031         33051         33066         33172
33403
## -2.076040e+15  2.732910e+15 -1.205163e+14 -2.631614e+15 -
4.183108e+14
##         33428         33627         34183         34579
34629
## -6.952269e+14 -6.747441e+14 -8.900155e+14 -1.438041e+15 -
1.642654e+15
##         34816         35011         35056         35090
35166
## -1.250288e+15 -2.855319e+15 -2.927560e+14 -1.277079e+15 -
1.028165e+15
##         35187         35237         35467         35545
35691
## -1.456772e+15  1.169571e+15 -6.892853e+14  5.136591e+15 -
1.400267e+15
```

```
##         35814         36003         36267         36355
36415
## -4.904718e+14 -6.880526e+14 -2.129474e+15  3.161623e+15 -
2.035450e+15
##         36565         36626         36658         36961
37019
## -6.361604e+14 -1.346006e+15 -1.800704e+15  6.250768e+15 -
3.887758e+15
##         37251         37327         37362         37414
37735
## -1.032187e+15 -1.452964e+15 -7.886487e+14 -3.386227e+14
6.054572e+15
##         37825         37866         37894
## -1.268683e+15 -4.297927e+15 -1.365341e+15

#omitting any missing values
predictions3 <- na.omit(unlist(predictions3))
data4_testvec <- na.omit(unlist(data4_test$repay_fail))

#normalization of predictions3
predictions3 <- scale(predictions3)
print(predictions3)

##                [,1]
## 140    0.7756029578
## 502    0.1597417989
## 590    0.0500415475
## 797    5.2866752713
## 1213  -0.1572289925
## 1351   0.2760668870
## 2393   0.0175093447
## 2501   0.0162796767
## 2671  -0.2578248579
## 2893  -0.4075522062
## 3113  -0.0446654025
## 3264  -0.7763850113
## 3411  -0.1201913802
## 3632  -0.2618840426
## 3731   0.2563045378
## 3785  -1.8368976444
## 3789   0.0635498330
## 4142  -0.5400353200
## 4586   0.1628770000
## 4711   0.1489162615
## 4808   0.2158260640
## 5064   2.7624601365
## 5099   0.0751538362
## 5177  -0.7134783661
## 5266   2.0565013297
## 5662   0.2680313763
```

```
## 5753  -0.8047994940
## 5927   1.0389952752
## 5973  -0.6400923797
## 5988  -0.1659628990
## 6232   0.0304287097
## 6329   1.4281083784
## 6371  -0.2419547667
## 6385  -0.4357910745
## 7253  -1.0197269926
## 7457   2.5309728684
## 7459   0.0452378263
## 7715  -0.4996538946
## 7781  -0.2537132695
## 7854  -0.1787683358
## 7980   0.7178925333
## 8293  -0.5821365450
## 8360  -0.3952203455
## 9111   0.5485266701
## 9333  -0.3546927940
## 9353  -0.6583009392
## 9359  -0.7754417231
## 9507  -0.2652081623
## 9629  -0.4065630212
## 10077 -1.3110183793
## 10145 -0.3317772942
## 10191 -0.4394244274
## 10497 -0.5726768961
## 10652 -0.4845227576
## 11062 -0.7388968701
## 11138  0.7304482790
## 11718  1.6689481627
## 11734  1.6887906126
## 11807 -0.2882090650
## 12045  0.0244046032
## 12062 -0.1845623209
## 13249 -0.1653481877
## 13281 -0.2502370706
## 13509 -0.5635356653
## 13526 -0.0706182660
## 13532 -0.1281289589
## 13744 -0.2681760969
## 13771  0.1140412256
## 13786 -0.2918033672
## 13939  0.2552328472
## 14127 -0.3753542547
## 15106 -2.5265733631
## 15473 -0.3555469254
## 15519  0.1121614117
## 15893 -0.3318253061
## 16039  0.9123543989
```

```
## 16133  1.3092535265
## 16142 -0.3509250702
## 16507  0.6910237142
## 16640  4.0354667707
## 16730 -0.2288208894
## 16738 -0.4006576367
## 17005 -0.7453238899
## 17293  0.1635999580
## 17770  0.2028952014
## 17835 -0.6382809706
## 17907 -2.2178099000
## 17980  0.4910615764
## 18248 -0.0406040038
## 18603 -0.5941589542
## 18650  0.0335946665
## 18857 -1.8697440761
## 19040  0.2357366852
## 19295 -0.0008658143
## 19711 -0.3097978370
## 19753  0.1906274449
## 20296 -0.3767364084
## 20406 -0.8978930428
## 20688  0.4296755092
## 20726 -0.1306728993
## 20746 -0.6570001088
## 20769 -0.2353926491
## 20864 -0.2829311073
## 20961 -0.6587541897
## 21294  0.3031342438
## 21900 -0.6828922974
## 22098 -1.4825525197
## 22209  0.1790391558
## 22338 -0.8993426256
## 22633 -0.5362292491
## 22668 -0.2594400361
## 23182 -0.6059093253
## 23510  0.1575313249
## 23642 -0.5953283464
## 23927 -0.3088854581
## 23943 -0.8255414223
## 23976  1.6263275421
## 24031 -0.5270961462
## 24125 -0.1341658815
## 24184  0.1696747085
## 24839  1.2666239797
## 25041  0.2008845443
## 25219  0.4564538544
## 25284  0.2393124754
## 25515 -0.2168351711
## 25529 -0.7582467512
```

```
## 25618 -0.7889630678
## 25879  0.2885518548
## 26036 -0.0997680418
## 26210 -1.2242832431
## 26422  1.2368403118
## 27102 -0.4635820128
## 27336  0.1567899495
## 27550 -0.3092781680
## 27581  3.1345738838
## 27668  0.2464773915
## 28048  0.1043686486
## 28120 -0.1613124947
## 28444  0.4769586779
## 28565 -0.5174175289
## 28575 -0.5078078700
## 28613 -0.1394997063
## 29328  0.1131166566
## 29501 -0.0080708300
## 29834  0.3669839711
## 30017 -0.0141547225
## 30064  0.0470272487
## 30140  2.0583330281
## 30445 -0.2361283403
## 30616  0.9323242968
## 30777 -2.5907897991
## 30895 -0.7104478333
## 31048 -1.0187004694
## 31113  1.6406424268
## 31824 -0.5041087020
## 33031 -0.6219220718
## 33051  1.5172944645
## 33066  0.2479744013
## 33172 -0.8690639097
## 33403  0.1155032660
## 33428 -0.0076802601
## 33627  0.0014313297
## 34183 -0.0943301663
## 34579 -0.3381139886
## 34629 -0.4291342587
## 34816 -0.2545940128
## 35011 -0.9685770430
## 35056  0.1713551447
## 35090 -0.2665116636
## 35166 -0.1557847717
## 35187 -0.3464463289
## 35237  0.8218576036
## 35467 -0.0050371988
## 35545  2.5865496340
## 35691 -0.3213107815
## 35814  0.0834031316
```

```
## 36003 -0.0044888599
## 36267 -0.6456914937
## 36355  1.7080031932
## 36415 -0.6038661521
## 36565  0.0185949360
## 36626 -0.2971731533
## 36658 -0.4994414789
## 36961  3.0821806179
## 37019 -1.4278479465
## 37251 -0.1575736603
## 37327 -0.3447525726
## 37362 -0.0492380963
## 37414  0.1509518078
## 37735  2.9949048763
## 37825 -0.2627768229
## 37866 -1.6103077193
## 37894 -0.3057740435
## attr(,"scaled:center")
## [1] -6.779617e+14
## attr(,"scaled:scale")
## [1] 2.247996e+15
```

```
print(data4_testvec)
```

```
##   [1] 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0
1 0 0 0 1 0
##  [38] 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
##  [75] 0 1 1 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0
## [112] 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0
0 0 1 0 0 1
## [149] 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1
0 0 0 0 1 0
## [186] 0 0 0 0 1 0 0 0
```

```
# computing model performance metrics
df3 <- data.frame( R23 = R2(predictions3, data4_testvec), #R2 error
            RMSE3 = RMSE(predictions3, data4_testvec), #root mean
square error
            MAE3 = MAE(predictions3,data4_testvec)) #mean absolute
error

#observing results
print(df1)
```

```
##         R21     RMSE1      MAE1
## 1 0.5813747 0.7691173 0.4680241
```

```
print(df2)
```

```
##         R22     RMSE2      MAE2
## 1 0.5857539 0.7676999 0.4708263

print(df3)

##         R23     RMSE3      MAE3
## 1 0.5626744 0.7752011 0.5054075

#results
#R22>R21>R23
#RMSE3>RMSE1>RMSE2
#MAE2>MAE1>MAE3

#considering the 3 error metrics the model 1 seems to be the most
plausible, the logistic regression model is suitable for this task

#cross validation for model selection
library(caret)
head(data4)

##   loan_amnt installment total_pymnt_inv total_rec_prncp
total_rec_int
## 2         0        0.00            0.00            0.00
0.00
## 3      2500       85.42         3075.29         2500.00
575.29
## 4      5000      175.67         2948.76         1909.02
873.81
## 5      7000      225.58         8082.39         7000.00
1082.39
## 6      2000       60.32         2161.66         2000.00
161.66
## 7      3600      116.59         4206.03         3600.00
606.03
##   repay_fail
## 2          1
## 3          0
## 4          1
## 5          0
## 6          0
## 7          0

#assinging the equation
eqn <- unlist(repay_fail) ~ loan_amnt + installment + total_pymnt_inv
+ total_rec_prncp + total_rec_int
#fitting model(logit)
model_logit_crossval <- train(
  eqn,
  data = data4, #dataset
  method = "glm", #model for validation
  trControl = trainControl(method = "cv", number = 10), # 10-fold
```

```
cross-validation
  metric = "RMSE", #metric of evaluation
  family=binomial(link = "logit"), #family of link for glm
)

## Warning in train.default(x, y, weights = w, ...): You are trying to
do
## regression and your outcome only has two possible values Are you
trying to do
## classification? If so, use a 2 level factor as your outcome column.

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

print(model_logit_crossval)

## Generalized Linear Model
##
## 38419 samples
##     5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 34577, 34577, 34577, 34577, 34577,
34578, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.1120959  0.9018555  0.02860973
```

```r
#fitting model(probit)
model_probit_crossval <- train(
  eqn,
  data = data4, #dataset
  method = "glm", #model for validation
  trControl = trainControl(method = "cv", number = 10), # 10-fold
cross-validation
  metric = "RMSE", #metric of evaluation
  family=binomial(link = "probit"), #family of link for glm
)
```

```
## Warning in train.default(x, y, weights = w, ...): You are trying to
do
## regression and your outcome only has two possible values Are you
trying to do
## classification? If so, use a 2 level factor as your outcome column.

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
```

```
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred

## Warning in train.default(x, y, weights = w, ...): glm.fit: fitted
probabilities
## numerically 0 or 1 occurred
```

```
print(model_probit_crossval)
```

```
## Generalized Linear Model
##
## 38419 samples
##     5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 34577, 34577, 34577, 34577, 34577,
34578, ...
## Resampling results:
##
##   RMSE     Rsquared   MAE
##   0.12302  0.8826403  0.03597402
```

```
#fitting model(cloglog)
model_cloglog_crossval <- train(
  eqn,
  data = data4, #dataset
  method = "glm", #model for validation
  trControl = trainControl(method = "cv", number = 10), # 10-fold
cross-validation
  metric = "RMSE", #metric of evaluation
  family=binomial(link = "cloglog"), #family of link for glm
)
```

```
## Warning in train.default(x, y, weights = w, ...): You are trying to
do
## regression and your outcome only has two possible values Are you
trying to do
## classification? If so, use a 2 level factor as your outcome column.

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
print(model_cloglog_crossval)
```

```
## Generalized Linear Model
##
## 38419 samples
##     5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 34577, 34577, 34578, 34577, 34577,
34577, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.1367265  0.8526693  0.01952194
```

```
#goodness of fit using receiver operating characteristic curve(ROC)


library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

#logit
# Create an ROC object
predictions1 <- na.omit(unlist(predictions1))
data4_testvec <- na.omit(unlist(data4_test$repay_fail))

print(data4_testvec)

##   [1] 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0
1 0 0 0 1 0
##  [38] 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
##  [75] 0 1 1 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0
## [112] 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0
0 0 1 0 0 1
## [149] 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1
0 0 0 0 1 0
## [186] 0 0 0 0 1 0 0 0

print(predictions1)

##                [,1]
## 140    9.094876e-01
## 502   -9.328351e-02
## 590   -9.875116e-02
## 797    5.861362e+00
## 1213   2.605647e-02
## 1351   4.179802e-02
## 2393  -7.163973e-02
## 2501  -1.195565e-01
## 2671  -3.176920e-01
## 2893  -4.170924e-01
## 3113  -1.221127e-02
## 3264  -8.839130e-01
## 3411  -2.143990e-01
## 3632  -1.457409e-01
## 3731   1.830017e-02
## 3785  -1.347071e+00
## 3789  -2.555898e-02
## 4142  -4.872048e-01
## 4586  -6.091839e-03
## 4711  -2.468559e-02
```

```
## 4808   9.243829e-03
## 5064   2.953044e+00
## 5099  -8.145982e-02
## 5177  -7.690408e-01
## 5266   2.084618e+00
## 5662   5.394500e-02
## 5753  -6.742962e-01
## 5927   1.178639e+00
## 5973  -5.206499e-01
## 5988  -1.887142e-01
## 6232  -9.798143e-02
## 6329   1.336354e+00
## 6371  -2.196699e-01
## 6385  -3.575063e-01
## 7253  -6.654275e-01
## 7457   2.601089e+00
## 7459  -9.167472e-02
## 7715  -2.610933e-01
## 7781  -2.491254e-01
## 7854  -1.962741e-01
## 7980   1.088503e+00
## 8293  -3.541768e-01
## 8360  -3.110173e-01
## 9111   4.033819e-01
## 9333  -2.963175e-01
## 9353  -8.088829e-01
## 9359  -8.854782e-01
## 9507  -2.717630e-01
## 9629  -1.922779e-01
## 10077 -1.229044e+00
## 10145 -3.954755e-01
## 10191 -2.126510e-01
## 10497 -2.617309e-01
## 10652 -6.054841e-01
## 11062 -4.630834e-01
## 11138  5.217557e-01
## 11718  1.745762e+00
## 11734  1.689698e+00
## 11807 -1.805942e-01
## 12045 -6.564335e-02
## 12062 -2.315369e-01
## 13249 -2.075051e-01
## 13281 -2.399955e-01
## 13509 -4.477051e-01
## 13526 -1.813529e-01
## 13532 -1.951352e-01
## 13744 -2.419545e-01
## 13771 -7.988529e-04
## 13786 -3.052799e-01
## 13939  3.839323e-02
```

```
## 14127 -5.549477e-01
## 15106 -2.237372e+00
## 15473 -3.503032e-01
## 15519 -3.091001e-03
## 15893 -2.879933e-01
## 16039  7.490316e-01
## 16133  1.342706e+00
## 16142 -1.241498e-01
## 16507  6.711273e-01
## 16640  4.296962e+00
## 16730 -1.085871e-01
## 16738 -3.863688e-01
## 17005 -7.789012e-01
## 17293 -2.845478e-02
## 17770 -3.195426e-03
## 17835 -5.192710e-01
## 17907 -2.144243e+00
## 17980  3.489138e-01
## 18248 -1.646861e-01
## 18603 -4.172350e-02
## 18650 -9.800157e-02
## 18857 -1.116744e+00
## 19040  1.467098e-02
## 19295 -1.412525e-01
## 19711 -3.179438e-01
## 19753 -2.376803e-02
## 20296 -2.914310e-01
## 20406 -5.393724e-01
## 20688  1.937124e-01
## 20726 -1.262140e-01
## 20746 -7.447821e-01
## 20769 -2.371826e-01
## 20864 -4.758130e-01
## 20961 -7.248961e-01
## 21294  5.263303e-02
## 21900 -4.512723e-01
## 22098 -1.315237e+00
## 22209 -1.012790e-02
## 22338 -5.661300e-01
## 22633 -2.324092e-01
## 22668 -3.205744e-01
## 23182 -7.298181e-01
## 23510 -2.885835e-02
## 23642 -2.450878e-01
## 23927 -3.120725e-01
## 23943 -8.923715e-01
## 23976  1.628264e+00
## 24031 -5.965396e-01
## 24125 -1.765894e-01
## 24184 -4.363594e-02
```

```
## 24839  1.202335e+00
## 25041  1.694827e-02
## 25219  4.617708e-01
## 25284  3.104336e-02
## 25515 -1.824438e-01
## 25529 -5.931456e-01
## 25618 -8.053282e-01
## 25879  4.579801e-02
## 26036 -1.389033e-01
## 26210 -1.124558e+00
## 26422  1.110096e+00
## 27102 -5.453947e-01
## 27336 -3.720809e-02
## 27550 -3.599896e-01
## 27581  3.636877e+00
## 27668  2.766287e-02
## 28048 -5.679668e-02
## 28120 -2.475665e-01
## 28444  2.274939e-01
## 28565 -4.367844e-01
## 28575 -3.888817e-01
## 28613 -1.034393e-01
## 29328 -4.214430e-02
## 29501 -1.001694e-01
## 29834  1.960815e-01
## 30017 -2.402154e-02
## 30064 -1.255270e-01
## 30140  2.202382e+00
## 30445 -4.562293e-01
## 30616  8.166711e-01
## 30777 -2.398214e+00
## 30895 -6.035477e-01
## 31048 -9.763553e-01
## 31113  1.481994e+00
## 31824 -3.346059e-01
## 33031 -7.278489e-01
## 33051  1.758269e+00
## 33066  9.095583e-03
## 33172 -6.716167e-01
## 33403 -3.403937e-02
## 33428 -1.455265e-01
## 33627 -1.467245e-01
## 34183 -3.590781e-02
## 34579 -4.140398e-01
## 34629 -4.946823e-01
## 34816 -2.092789e-01
## 35011 -7.333771e-01
## 35056 -3.860035e-02
## 35090 -2.606238e-01
## 35166 -1.050382e-01
```

```
## 35187 -1.187377e-01
## 35237  6.380868e-01
## 35467 -7.191310e-02
## 35545  2.719042e+00
## 35691 -2.936733e-01
## 35814 -1.598119e-02
## 36003 -1.071422e-01
## 36267 -6.175989e-01
## 36355  1.753339e+00
## 36415 -7.451286e-01
## 36565 -1.466547e-01
## 36626 -2.795252e-01
## 36658 -3.798638e-01
## 36961  3.015873e+00
## 37019 -1.112175e+00
## 37251 -2.424921e-01
## 37327 -2.365147e-01
## 37362 -1.751803e-01
## 37414 -9.859517e-05
## 37735  3.091526e+00
## 37825 -4.425509e-01
## 37866 -9.942159e-01
## 37894 -5.097295e-01
## attr(,"scaled:center")
## [1] -2.878606
## attr(,"scaled:scale")
## [1] 13.35835
```

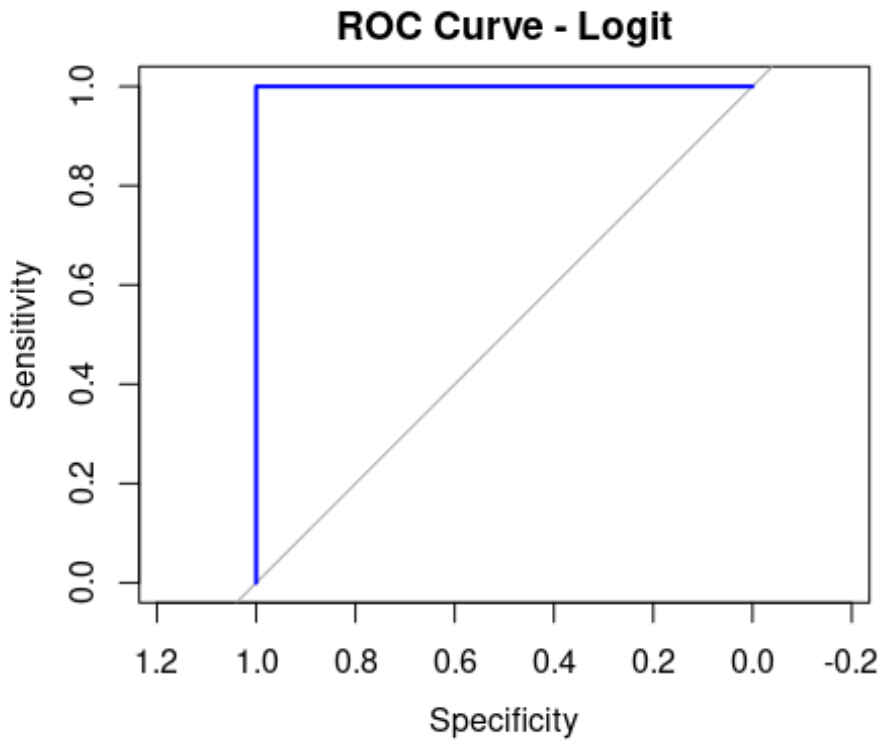
```
roc_obj_logit <- roc(response = data4_testvec, predictor =
predictions1)
```

```
## Setting levels: control = 0, case = 1
```

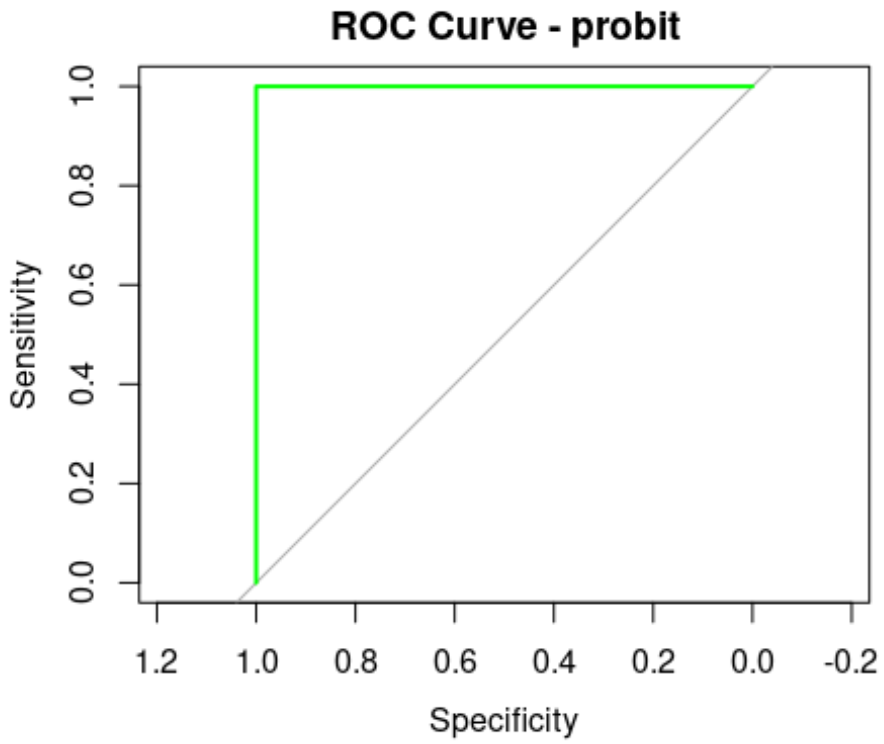
```
## Warning in roc.default(response = data4_testvec, predictor =
predictions1):
## Deprecated use a matrix as predictor. Unexpected results may be
produced,
## please pass a numeric vector.
```

```
## Setting direction: controls < cases
```

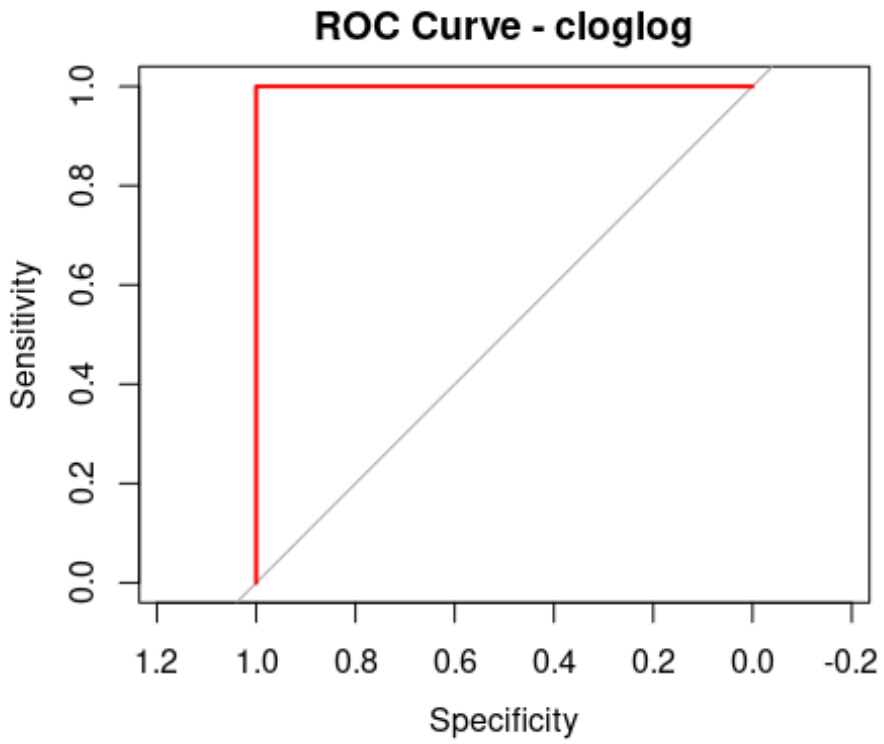
```
# AUC value
auc_value_logit <- auc(roc_obj_logit)

#ROC curve
plot(roc_obj_logit, main = "ROC Curve - Logit", col = "blue")
```



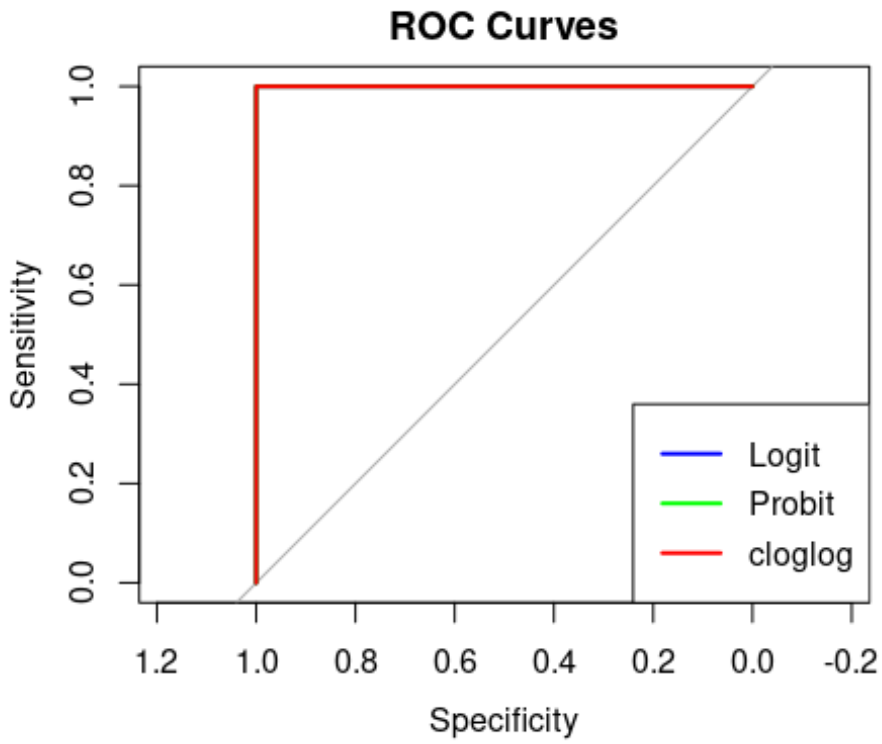

```
#AUC values
print(auc_value_logit)

## Area under the curve: 1

#probit
# Create an ROC object
predictions2 <- na.omit(unlist(predictions2))
data4_testvec <- na.omit(unlist(data4_test$repay_fail))
roc_obj_probit <- roc(response = data4_testvec, predictor =
predictions2)

## Setting levels: control = 0, case = 1

## Warning in roc.default(response = data4_testvec, predictor =
predictions2):
## Deprecated use a matrix as predictor. Unexpected results may be
produced,
## please pass a numeric vector.

## Setting direction: controls < cases

# AUC value
auc_value_probit <- auc(roc_obj_probit)

#ROC curve
plot(roc_obj_probit, main = "ROC Curve - probit", col = "green")
```




```
#AUC values
print(auc_value_probit)

## Area under the curve: 1

#cloglog
# Create an ROC object
predictions3 <- na.omit(unlist(predictions3))
data4_testvec <- na.omit(unlist(data4_test$repay_fail))
roc_obj_cloglog <- roc(response = data4_testvec, predictor =
predictions3)

## Setting levels: control = 0, case = 1

## Warning in roc.default(response = data4_testvec, predictor =
predictions3):
## Deprecated use a matrix as predictor. Unexpected results may be
produced,
## please pass a numeric vector.

## Setting direction: controls < cases

# AUC value
auc_value_cloglog <- auc(roc_obj_cloglog)

#ROC curve
plot(roc_obj_cloglog, main = "ROC Curve - cloglog", col = "red")
```




```
#AUC values
print(auc_value_cloglog)

## Area under the curve: 1

#all curves in one plot
plot(roc_obj_logit, col = "blue", main = "ROC Curves", lwd = 2)
plot(roc_obj_probit, col = "green", add = TRUE, lwd = 2)
plot(roc_obj_cloglog, col = "red", add = TRUE, lwd = 2)
legend("bottomright", legend = c("Logit", "Probit","cloglog"), col =
c("blue","green", "red"), lwd = 2)
```




```
#validation using AIC(akike information criterion)

# Calculation of AIC
aic_1 <- AIC(model_logit)
aic_2 <- AIC(model_probit)
aic_3 <- AIC(model_cloglog)

# Comparing AIC values

if (aic_1 < aic_2 & aic_1 < aic_3) {
  cat("Model 1 most suitable.\n")
} else if (aic_2 < aic_1 & aic_2 < aic_3) {
  cat("Model 2 is most suitable.\n")
} else {
  cat("Model 3 is most suitable.\n")
}

## Model 1 most suitable.

#using the AIC we have determined that model 1 is the best

#####################################

#using the 2 seperate metric we have determined and selected the model
1 to be the most suitable
```