stat5010_final_project

Siri Devarapalli

Credit risk is defined as the risk of loss resulting from the failure by a borrower to repay the principal and interest owed to the leader. The lender uses the interest payments from the loan to compensate for the risk of potential losses. When the borrower defaults on his/her obligations, it causes an interruption in the cash flows of the lender.

Performing credit risk analysis helps the lender determine the borrower's ability to meet debt obligations in order to cushion itself from loss of cash flows and reduce the severity of losses. Borrowers who present a high level of credit risk are charged a high interest rate on the loan to compensate the lender for the high risk of default.

```
library(gmodels)
library(fastDummies)
library(lares)
library(car)
```

```
## Loading required package: carData
```

Preprocessing

```
data <- read.csv("C:/UCB/stat - 5010/application_train.csv/credit_risk_dataset.csv")
```

```
str(data)
```

```
## 'data.frame':    32581 obs. of  12 variables:
##  $ person_age                : int  22 21 25 23 24 21 26 24 24 21 ...
##  $ person_income             : int  59000 9600 9600 65500 54400 9900 77100 78956 83000 10000 ...
##  $ person_home_ownership     : chr  "RENT" "OWN" "MORTGAGE" "RENT" ...
##  $ person_emp_length         : num  123 5 1 4 8 2 8 5 8 6 ...
##  $ loan_intent               : chr  "PERSONAL" "EDUCATION" "MEDICAL" "MEDICAL" ...
##  $ loan_grade                : chr  "D" "B" "C" "C" ...
##  $ loan_amnt                 : int  35000 1000 5500 35000 35000 2500 35000 35000 35000 1600 ...
##  $ loan_int_rate             : num  16 11.1 12.9 15.2 14.3 ...
##  $ loan_status               : int  1 0 1 1 1 1 1 1 1 1 ...
##  $ loan_percent_income       : num  0.59 0.1 0.57 0.53 0.55 0.25 0.45 0.44 0.42 0.16 ...
##  $ cb_person_default_on_file : chr  "Y" "N" "N" "N" ...
##  $ cb_person_cred_hist_length: int  3 2 3 2 4 2 3 4 2 3 ...
```

The dataset appears to have 12 columns/variables

1. person_age - age of borrower.

2. person_income - income of the borrower.
3. person_home_ownership - a categorical variable which indicates whether the borrower has a home or not.
4. person_emp_length - number of years the borrower has been employed.
5. loan_intent - reason of taking a loan.
6. loan_grade - a classification system that involves assigning a quality score to a loan based on a borrower's credit history, quality of the collateral, and the likelihood of repayment of the principal and interest.
7. loan_amnt - The loan amount
8. loan_int_rate - Interest rate of the loan
9. loan_status - Loan approved or not (1 for approved loan and 0 for declined loan)
10. loan_percent_income - Loan to income ratio
11. cb_person_default_on_file - If the person defaulted in the past.
12. cb_person_cred_hist_length - The credit history length.

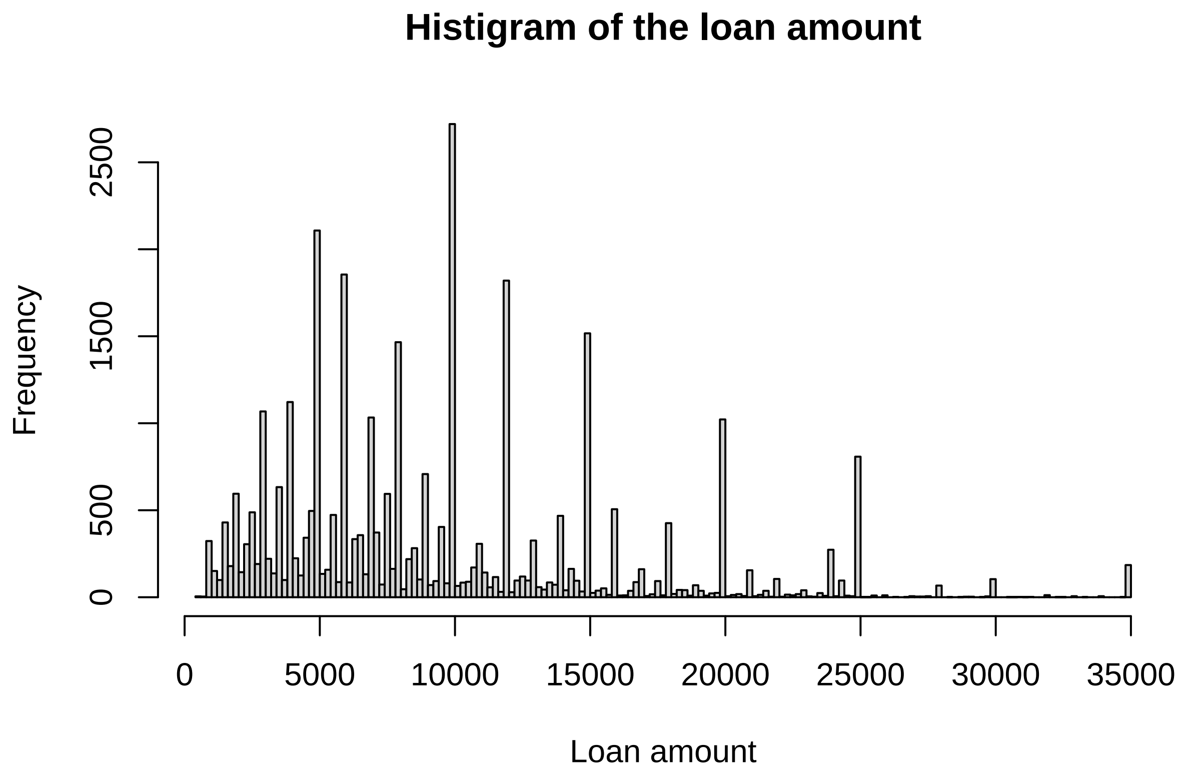
```
CrossTable(data$loan_status)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  32581
##
##
##           |         0 |         1 |
##           |-----------|-----------|
##           |     25473 |      7108 |
##           |     0.782 |     0.218 |
##           |-----------|-----------|
##
##
##
##
```

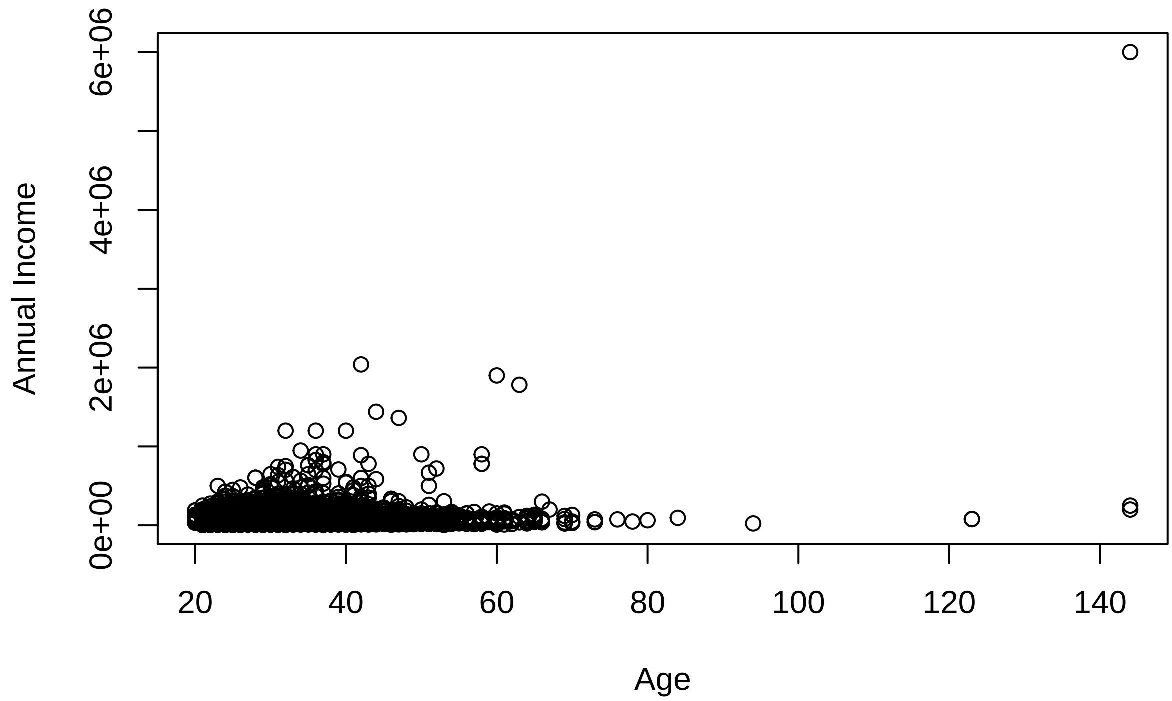
In the given data set 0.218 percent of the loans were approved.

Visualisations

```r
hist_1 <- hist(data$loan_amnt, breaks = 200, xlab = "Loan amount",
               main = "Histigram of the loan amount")
```



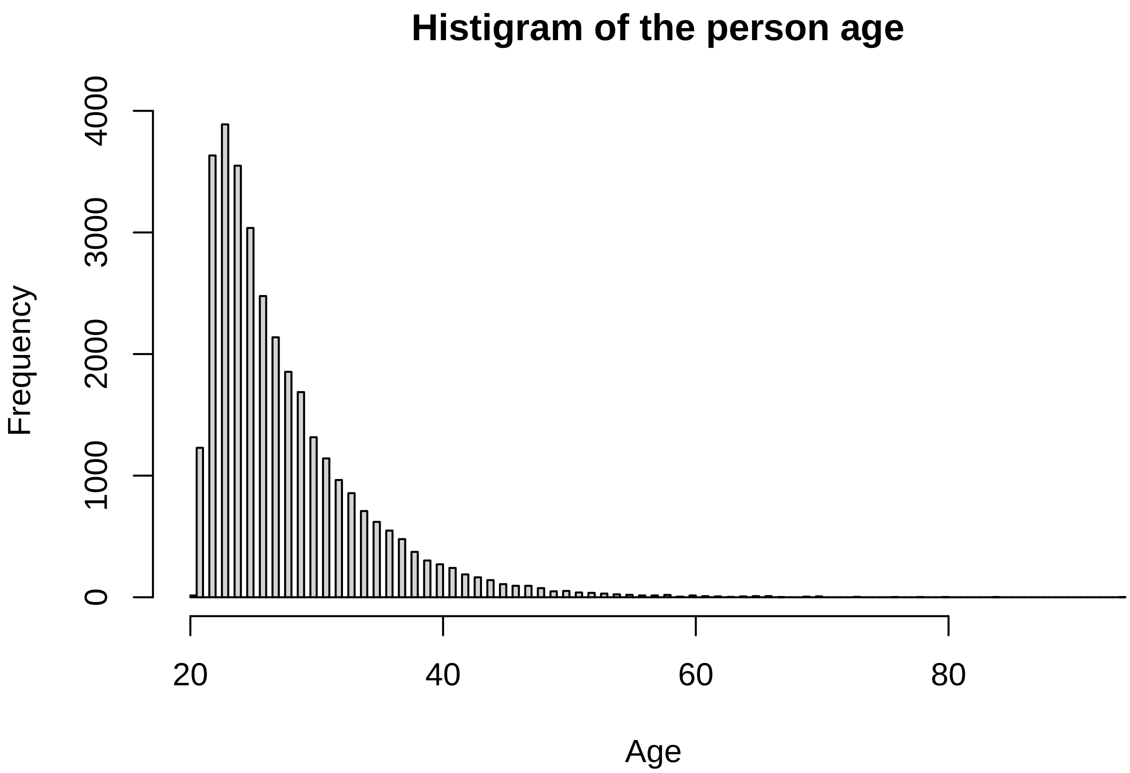

```r
plot(x = data$person_age, y = data$person_income, xlab = "Age",
     ylab = "Annual Income")
```



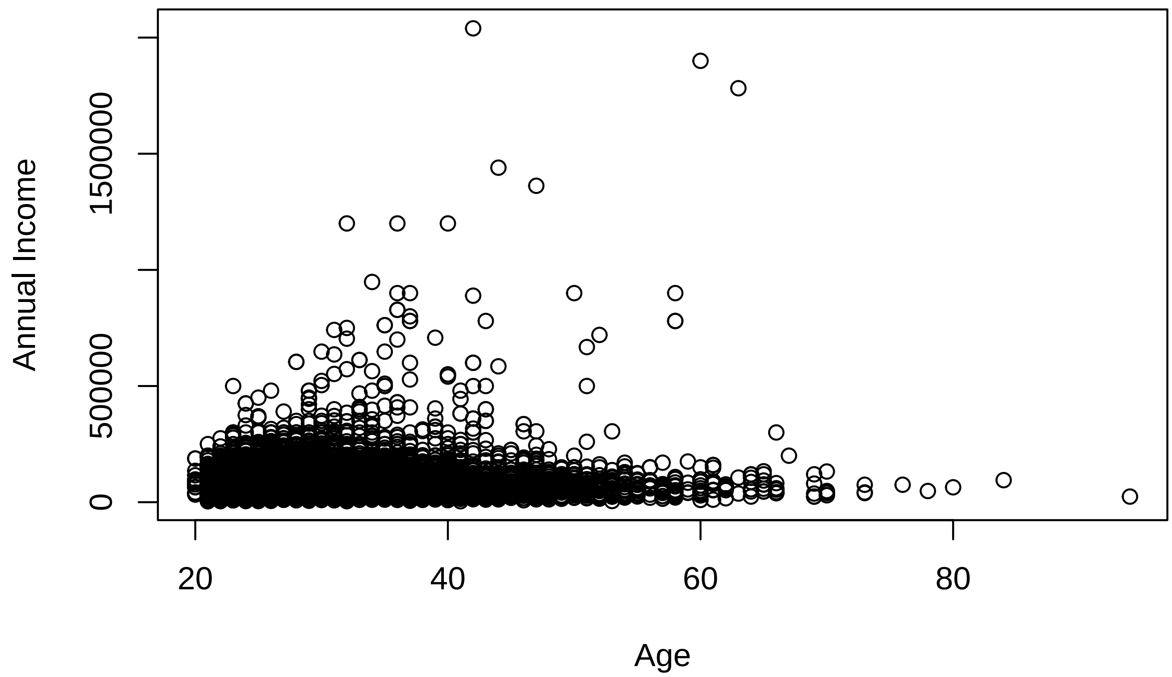
The person who has an annual income of 6 million is more than 140 years. This must be a mistake so we can take it out from the data along with some other outliers . All the datapoints which are greater than 120 can be taken out.

```
index_highage = which(data$person_age >120)
data <- data[-index_highage, ]
hist(x = data$person_age, breaks = 200, xlab = "Age ",
     main = "Histogram of the person age")
```


```r
hist(x = data$person_age, breaks = 200, xlab = "Age ",
     main = "Histigram of the person age")
```




```
plot(x = data$person_age, y = data$person_income,
     xlab = "Age", ylab = "Annual Income")
```




```
summary(data)
```

```
##    person_age    person_income     person_home_ownership person_emp_length
##  Min.   :20.00   Min.   :   4000   Length:32576          Min.   :  0.00
##  1st Qu.:23.00   1st Qu.:  38500   Class :character      1st Qu.:  2.00
##  Median :26.00   Median :  55000   Mode  :character      Median :  4.00
##  Mean   :27.72   Mean   :  65882                         Mean   :  4.79
##  3rd Qu.:30.00   3rd Qu.:  79200                         3rd Qu.:  7.00
##  Max.   :94.00   Max.   :2039784                         Max.   :123.00
##                                                          NA's   :895
##  loan_intent         loan_grade          loan_amnt     loan_int_rate
##  Length:32576       Length:32576       Min.   :  500   Min.   : 5.42
##  Class :character   Class :character   1st Qu.: 5000   1st Qu.: 7.90
##  Mode  :character   Mode  :character   Median : 8000   Median :10.99
##                                        Mean   : 9589   Mean   :11.01
##                                        3rd Qu.:12200   3rd Qu.:13.47
##                                        Max.   :35000   Max.   :23.22
##                                                        NA's   :3115
##   loan_status     loan_percent_income cb_person_default_on_file
##  Min.   :0.0000   Min.   :0.0000      Length:32576
##  1st Qu.:0.0000   1st Qu.:0.0900      Class :character
##  Median :0.0000   Median :0.1500      Mode  :character
##  Mean   :0.2182   Mean   :0.1702
##  3rd Qu.:0.0000   3rd Qu.:0.2300
##  Max.   :1.0000   Max.   :0.8300
```

```
##
##  cb_person_cred_hist_length
##  Min.   : 2.000
##  1st Qu.: 3.000
##  Median : 4.000
##  Mean   : 5.804
##  3rd Qu.: 8.000
##  Max.   :30.000
##
```

There are many NA values in the interest rate column. As the number of records which we might loose by deleting the records with NA values is 9.56% we can replace these values with the median interest rate.

```
na_index <- which(is.na(data$loan_int_rate))
median_ir <- median(data$loan_int_rate, na.rm = TRUE)

data$loan_int_rate[na_index] <- median_ir
summary(data$loan_int_rate)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.42    8.49   10.99   11.01   13.11   23.22
```

To evaluate categorical variables we need to create dummy variables.

```
data <- dummy_cols(data, select_columns = c('person_home_ownership',
                                            'loan_intent', 'loan_grade', 'cb_person_default_on_file'),
                   remove_first_dummy = TRUE)
data[,c('person_home_ownership','loan_intent', 'loan_grade', 'cb_person_default_on_file')]<-list(NULL)
summary(data)
```

```
##    person_age    person_income     person_emp_length   loan_amnt
##  Min.   :20.00   Min.   :   4000   Min.   :  0.00    Min.   :  500
##  1st Qu.:23.00   1st Qu.:  38500   1st Qu.:  2.00    1st Qu.: 5000
##  Median :26.00   Median :  55000   Median :  4.00    Median : 8000
##  Mean   :27.72   Mean   :  65882   Mean   :  4.79    Mean   : 9589
##  3rd Qu.:30.00   3rd Qu.:  79200   3rd Qu.:  7.00    3rd Qu.:12200
##  Max.   :94.00   Max.   :2039784   Max.   :123.00    Max.   :35000
##                                    NA's   :895
##  loan_int_rate    loan_status     loan_percent_income
##  Min.   : 5.42   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 8.49   1st Qu.:0.0000   1st Qu.:0.0900
##  Median :10.99   Median :0.0000   Median :0.1500
##  Mean   :11.01   Mean   :0.2182   Mean   :0.1702
##  3rd Qu.:13.11   3rd Qu.:0.0000   3rd Qu.:0.2300
##  Max.   :23.22   Max.   :1.0000   Max.   :0.8300
##
##  cb_person_cred_hist_length person_home_ownership_OTHER
##  Min.   : 2.000             Min.   :0.000000
##  1st Qu.: 3.000             1st Qu.:0.000000
##  Median : 4.000             Median :0.000000
##  Mean   : 5.804             Mean   :0.003285
##  3rd Qu.: 8.000             3rd Qu.:0.000000
```

```
##  Max.   :30.000             Max.   :1.000000
##
##  person_home_ownership_OWN person_home_ownership_RENT loan_intent_EDUCATION
##  Min.   :0.00000           Min.   :0.0000            Min.   :0.000
##  1st Qu.:0.00000           1st Qu.:0.0000            1st Qu.:0.000
##  Median :0.00000           Median :1.0000            Median :0.000
##  Mean   :0.07932           Mean   :0.5048            Mean   :0.198
##  3rd Qu.:0.00000           3rd Qu.:1.0000            3rd Qu.:0.000
##  Max.   :1.00000           Max.   :1.0000            Max.   :1.000
##
##  loan_intent_HOMEIMPROVEMENT loan_intent_MEDICAL loan_intent_PERSONAL
##  Min.   :0.0000              Min.   :0.0000      Min.   :0.0000
##  1st Qu.:0.0000              1st Qu.:0.0000      1st Qu.:0.0000
##  Median :0.0000              Median :0.0000      Median :0.0000
##  Mean   :0.1107              Mean   :0.1864      Mean   :0.1694
##  3rd Qu.:0.0000              3rd Qu.:0.0000      3rd Qu.:0.0000
##  Max.   :1.0000              Max.   :1.0000      Max.   :1.0000
##
##  loan_intent_VENTURE  loan_grade_B     loan_grade_C     loan_grade_D
##  Min.   :0.0000      Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000      1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000      Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.1755      Mean   :0.3207   Mean   :0.1982   Mean   :0.1113
##  3rd Qu.:0.0000      3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :1.0000      Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##
##   loan_grade_E      loan_grade_F       loan_grade_G
##  Min.   :0.00000   Min.   :0.000000   Min.   :0.000000
##  1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:0.000000
##  Median :0.00000   Median :0.000000   Median :0.000000
##  Mean   :0.02959   Mean   :0.007398   Mean   :0.001965
##  3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:0.000000
##  Max.   :1.00000   Max.   :1.000000   Max.   :1.000000
##
##  cb_person_default_on_file_Y
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.1764
##  3rd Qu.:0.0000
##  Max.   :1.0000
##
```

Modelling

Split the data set into training and test data.

```r
index_train <- sample(1:nrow(data),2 / 3 * nrow(data) )
training_set <- data[index_train, ]
test_set <- data[-index_train,]
test_set <- na.omit(test_set)
```

Let us start with the full model

```r
log_model_all <- glm(loan_status~. ,family = "binomial" ,
                     data = training_set)
summary(log_model_all)
```

```
##
## Call:
## glm(formula = loan_status ~ ., family = "binomial", data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7929  -0.5115  -0.2937  -0.1249   3.4034
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -4.095e+00  2.147e-01 -19.073  < 2e-16 ***
## person_age                 -4.361e-03  7.293e-03  -0.598 0.549833
## person_income               2.022e-06  5.553e-07   3.641 0.000271 ***
## person_emp_length          -1.324e-02  5.838e-03  -2.268 0.023339 *
## loan_amnt                  -1.096e-04  5.450e-06 -20.105  < 2e-16 ***
## loan_int_rate               4.254e-02  1.607e-02   2.648 0.008098 **
## loan_percent_income         1.362e+01  3.208e-01  42.455  < 2e-16 ***
## cb_person_cred_hist_length  1.051e-03  1.115e-02   0.094 0.924886
## person_home_ownership_OTHER  2.455e-01  3.594e-01   0.683 0.494448
## person_home_ownership_OWN  -1.813e+00  1.310e-01 -13.843  < 2e-16 ***
## person_home_ownership_RENT  8.435e-01  5.040e-02  16.736  < 2e-16 ***
## loan_intent_EDUCATION      -9.195e-01  7.195e-02 -12.780  < 2e-16 ***
## loan_intent_HOMEIMPROVEMENT  6.170e-03  7.956e-02   0.078 0.938183
## loan_intent_MEDICAL        -1.299e-01  6.710e-02  -1.937 0.052805 .
## loan_intent_PERSONAL       -6.610e-01  7.315e-02  -9.037  < 2e-16 ***
## loan_intent_VENTURE        -1.117e+00  7.766e-02 -14.384  < 2e-16 ***
## loan_grade_B                3.251e-01  8.046e-02   4.040 5.34e-05 ***
## loan_grade_C                6.198e-01  1.132e-01   5.478 4.31e-08 ***
## loan_grade_D                2.806e+00  1.391e-01  20.180  < 2e-16 ***
## loan_grade_E                3.138e+00  1.812e-01  17.316  < 2e-16 ***
## loan_grade_F                3.546e+00  2.694e-01  13.164  < 2e-16 ***
## loan_grade_G                6.778e+00  1.055e+00   6.427 1.30e-10 ***
## cb_person_default_on_file_Y  1.091e-02  6.234e-02   0.175 0.861051
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 21963  on 21113  degrees of freedom
## Residual deviance: 13974  on 21091  degrees of freedom
##   (603 observations deleted due to missingness)
## AIC: 14020
##
## Number of Fisher Scoring iterations: 6
```

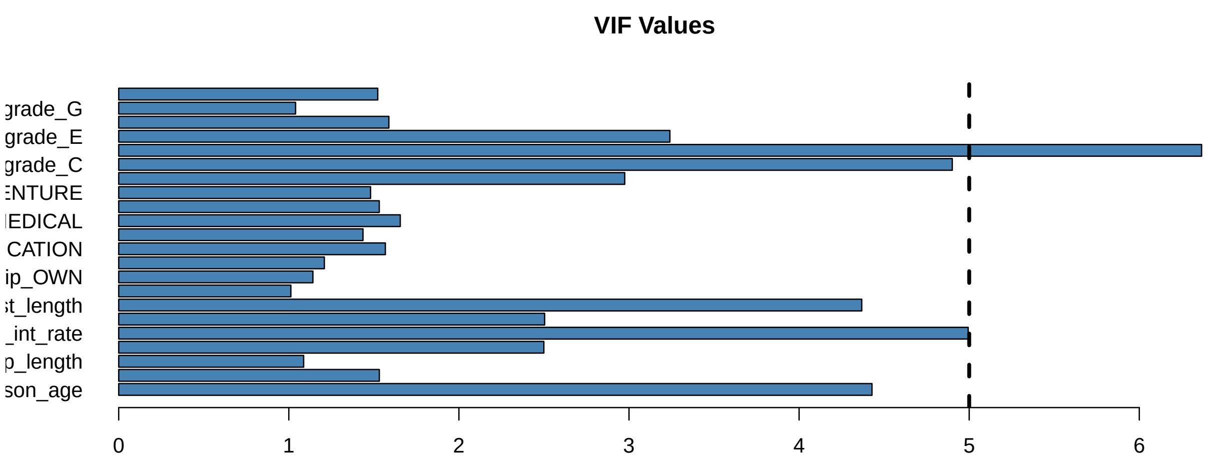
```r
predictions_all <- predict(log_model_all, newdata = test_set,
                           type = "response")
```

We can find out the vif values to detect multicollinearity

```r
vif_values <- vif(log_model_all)

#create horizontal bar chart to display each VIF value
barplot(vif_values, main = "VIF Values", horiz = TRUE,
        col = "steelblue", names = labels(vif_values), las = 1)

#add vertical line at 5
abline(v = 5, lwd = 3, lty = 2)
```



Remove the parameters which have high p-values

```r
log_model_multi <- glm(loan_status~person_age+loan_int_rate+
                      loan_grade_B+loan_grade_C+loan_grade_D+
                      loan_grade_E+loan_amnt+person_income+
                      person_home_ownership_OWN+
                      person_home_ownership_RENT+loan_percent_income+
                        loan_intent_PERSONAL+loan_intent_VENTURE,
                      family = "binomial" ,data = training_set)
summary(log_model_multi)
```

```
##
## Call:
## glm(formula = loan_status ~ person_age + loan_int_rate + loan_grade_B +
##     loan_grade_C + loan_grade_D + loan_grade_E + loan_amnt +
##     person_income + person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE,
##     family = "binomial", data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6192  -0.5307  -0.3142  -0.1327   3.3992
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -5.613e+00  1.557e-01 -36.044  < 2e-16 ***
```

```
## person_age                  1.230e-03  3.340e-03   0.368  0.71263
## loan_int_rate               2.047e-01  1.162e-02  17.625  < 2e-16 ***
## loan_grade_B               -3.370e-01  6.373e-02  -5.288 1.24e-07 ***
## loan_grade_C               -4.127e-01  8.191e-02  -5.039 4.69e-07 ***
## loan_grade_D                1.458e+00  9.760e-02  14.940  < 2e-16 ***
## loan_grade_E                1.531e+00  1.396e-01  10.969  < 2e-16 ***
## loan_amnt                  -1.070e-04  5.236e-06 -20.433  < 2e-16 ***
## person_income               1.713e-06  5.599e-07   3.060  0.00221 **
## person_home_ownership_OWN  -1.725e+00  1.191e-01 -14.487  < 2e-16 ***
## person_home_ownership_RENT  8.274e-01  4.740e-02  17.456  < 2e-16 ***
## loan_percent_income         1.329e+01  3.072e-01  43.257  < 2e-16 ***
## loan_intent_PERSONAL       -3.619e-01  5.800e-02  -6.239 4.40e-10 ***
## loan_intent_VENTURE        -8.088e-01  6.289e-02 -12.860  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22746  on 21716  degrees of freedom
## Residual deviance: 15024  on 21703  degrees of freedom
## AIC: 15052
##
## Number of Fisher Scoring iterations: 6
```

Remove age as it has the highest p value

```
log_model_multi_1 <- glm(loan_status~loan_int_rate+loan_grade_B+
                      loan_grade_C+loan_grade_D+loan_grade_E+loan_amnt+
                      person_income+person_home_ownership_OWN+
                      person_home_ownership_RENT+loan_percent_income+
                        loan_intent_PERSONAL+loan_intent_VENTURE ,
                      family = "binomial" ,data = training_set)
summary(log_model_multi_1)
```

```
##
## Call:
## glm(formula = loan_status ~ loan_int_rate + loan_grade_B + loan_grade_C +
##     loan_grade_D + loan_grade_E + loan_amnt + person_income +
##     person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE,
##     family = "binomial", data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6210  -0.5309  -0.3141  -0.1330   3.4013
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -5.581e+00  1.287e-01 -43.357  < 2e-16 ***
## loan_int_rate               2.048e-01  1.162e-02  17.631  < 2e-16 ***
## loan_grade_B               -3.373e-01  6.373e-02  -5.293 1.21e-07 ***
## loan_grade_C               -4.130e-01  8.191e-02  -5.042 4.61e-07 ***
## loan_grade_D                1.458e+00  9.760e-02  14.940  < 2e-16 ***
```

```
## loan_grade_E                1.531e+00  1.396e-01  10.966  < 2e-16 ***
## loan_amnt                  -1.069e-04  5.234e-06 -20.433  < 2e-16 ***
## person_income               1.730e-06  5.578e-07   3.102  0.00192 **
## person_home_ownership_OWN  -1.725e+00  1.191e-01 -14.486  < 2e-16 ***
## person_home_ownership_RENT  8.275e-01  4.740e-02  17.458  < 2e-16 ***
## loan_percent_income         1.329e+01  3.071e-01  43.261  < 2e-16 ***
## loan_intent_PERSONAL       -3.615e-01  5.799e-02  -6.233 4.58e-10 ***
## loan_intent_VENTURE        -8.090e-01  6.289e-02 -12.862  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22746  on 21716  degrees of freedom
## Residual deviance: 15024  on 21704  degrees of freedom
## AIC: 15050
##
## Number of Fisher Scoring iterations: 6
```

Remove the predictor loan grade as it might be causing multicollinearity

```
log_model_multi_2 <- glm(loan_status~loan_int_rate+loan_grade_B+
                      loan_grade_C+loan_grade_E+loan_amnt+person_income+
                      person_home_ownership_OWN+person_home_ownership_RENT+
                        loan_percent_income+loan_intent_PERSONAL+
                        loan_intent_VENTURE ,family = "binomial" ,
                      data = training_set)
summary(log_model_multi_2)
```

```
##
## Call:
## glm(formula = loan_status ~ loan_int_rate + loan_grade_B + loan_grade_C +
##     loan_grade_E + loan_amnt + person_income + person_home_ownership_OWN +
##     person_home_ownership_RENT + loan_percent_income + loan_intent_PERSONAL +
##     loan_intent_VENTURE, family = "binomial", data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7936  -0.5427  -0.3209  -0.1326   3.4166
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -6.443e+00  1.173e-01 -54.919  < 2e-16 ***
## loan_int_rate               3.362e-01  7.996e-03  42.049  < 2e-16 ***
## loan_grade_B               -9.256e-01  4.972e-02 -18.618  < 2e-16 ***
## loan_grade_C               -1.300e+00  5.734e-02 -22.665  < 2e-16 ***
## loan_grade_E                2.388e-01  1.117e-01   2.137  0.03259 *
## loan_amnt                  -1.069e-04  5.231e-06 -20.434  < 2e-16 ***
## person_income               1.643e-06  5.596e-07   2.935  0.00333 **
## person_home_ownership_OWN  -1.690e+00  1.189e-01 -14.209  < 2e-16 ***
## person_home_ownership_RENT  8.458e-01  4.697e-02  18.005  < 2e-16 ***
## loan_percent_income         1.320e+01  3.058e-01  43.150  < 2e-16 ***
## loan_intent_PERSONAL       -3.418e-01  5.737e-02  -5.958 2.56e-09 ***
```

```
## loan_intent_VENTURE        -8.041e-01  6.233e-02 -12.900  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22746  on 21716  degrees of freedom
## Residual deviance: 15248  on 21705  degrees of freedom
## AIC: 15272
##
## Number of Fisher Scoring iterations: 6
```

Remove loan grade E as it has a high p - value. It is insignificant.

```
log_model_multi_3 <- glm(loan_status~loan_int_rate+loan_grade_B+
                      loan_grade_C+loan_amnt+person_income+
                      person_home_ownership_OWN+
                      person_home_ownership_RENT+loan_percent_income+
                      loan_intent_PERSONAL+loan_intent_VENTURE ,
                      family = "binomial" ,data = training_set)
summary(log_model_multi_3)
```

```
##
## Call:
## glm(formula = loan_status ~ loan_int_rate + loan_grade_B + loan_grade_C +
##     loan_amnt + person_income + person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE,
##     family = "binomial", data = training_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8153  -0.5431  -0.3201  -0.1324   3.4146
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -6.495e+00  1.150e-01 -56.494  < 2e-16 ***
## loan_int_rate               3.421e-01  7.530e-03  45.429  < 2e-16 ***
## loan_grade_B               -9.438e-01  4.902e-02 -19.255  < 2e-16 ***
## loan_grade_C               -1.331e+00  5.551e-02 -23.972  < 2e-16 ***
## loan_amnt                  -1.065e-04  5.218e-06 -20.405  < 2e-16 ***
## person_income               1.667e-06  5.568e-07   2.994  0.00276 **
## person_home_ownership_OWN  -1.688e+00  1.188e-01 -14.213  < 2e-16 ***
## person_home_ownership_RENT  8.457e-01  4.695e-02  18.013  < 2e-16 ***
## loan_percent_income         1.319e+01  3.055e-01  43.189  < 2e-16 ***
## loan_intent_PERSONAL       -3.429e-01  5.735e-02  -5.979 2.24e-09 ***
## loan_intent_VENTURE        -8.031e-01  6.227e-02 -12.896  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22746  on 21716  degrees of freedom
## Residual deviance: 15252  on 21706  degrees of freedom
```

```
## AIC: 15274
##
## Number of Fisher Scoring iterations: 6
```

We can compare the models by using chi - square test

```r
anova(log_model_multi,log_model_multi_1, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: loan_status ~ person_age + loan_int_rate + loan_grade_B + loan_grade_C +
##     loan_grade_D + loan_grade_E + loan_amnt + person_income +
##     person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE
## Model 2: loan_status ~ loan_int_rate + loan_grade_B + loan_grade_C + loan_grade_D +
##     loan_grade_E + loan_amnt + person_income + person_home_ownership_OWN +
##     person_home_ownership_RENT + loan_percent_income + loan_intent_PERSONAL +
##     loan_intent_VENTURE
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1     21703      15024
## 2     21704      15024 -1 -0.13545   0.7128
```

As the p-value is significant, by removing the loan grade D parameter we will reject model log_model_multi_1.

```r
anova(log_model_multi,log_model_multi_2, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: loan_status ~ person_age + loan_int_rate + loan_grade_B + loan_grade_C +
##     loan_grade_D + loan_grade_E + loan_amnt + person_income +
##     person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE
## Model 2: loan_status ~ loan_int_rate + loan_grade_B + loan_grade_C + loan_grade_E +
##     loan_amnt + person_income + person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1     21703      15024
## 2     21705      15248 -2  -223.67 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As removing person_age helps improve the fit over the first model we should accept log_model_multi_2.

```r
anova(log_model_multi_2,log_model_multi_3, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: loan_status ~ loan_int_rate + loan_grade_B + loan_grade_C + loan_grade_E +
##     loan_amnt + person_income + person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE
## Model 2: loan_status ~ loan_int_rate + loan_grade_B + loan_grade_C + loan_amnt +
```

```
##     person_income + person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1     21705      15248
## 2     21706      15252 -1  -4.6052  0.03187 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As the p - value is significant removing the loan grade E parameter does not help improve the fit of the model. We will reject model log_model_multi_3.

log_model_multi_2 turns out to be the best model of all the others.

Predictions

We will calculate predictions on the above models. The predictions will be floating numbers. Each result indicates the probability of default. But banks do not need this measure, they just want to know what percentage of the loans are supposed to be approved to make sure the risk factor does not cross the threshold imposed by the banks. So, we will assign all the values which are 0.15 can be assigned to 0 and all the values which are grater than 0.15 are assigned to 1.

```r
predictions_multi <- predict(log_model_multi_1, newdata = test_set,
                             type = "response")
range(predictions_multi)
```

```
## [1] 0.0003658958 0.9978839410
```

```r
pred_cutoff_15 = ifelse(predictions_multi>0.15,1,0)
```

```r
tab_multi_1 <- table(test_set$loan_status,pred_cutoff_15)
acc_multi_model <- sum(diag(tab_multi_1)) / nrow(test_set)
acc_multi_model
```

```
## [1] 0.7344563
```

```r
sensitivity_multi_model = tab_multi_1[2,2]/sum(tab_multi_1[2,])
specificity_multi_model = tab_multi_1[1,1]/sum(tab_multi_1[1,])
sensitivity_multi_model
```

```
## [1] 0.8268644
```

```r
specificity_multi_model
```

```
## [1] 0.708847
```

The accuracy of the full model is 74%

```r
predictions_multi <- predict(log_model_multi_2, newdata = test_set,
                             type = "response")
pred_cutoff_15 = ifelse(predictions_multi>0.15,1,0)
tab_multi_2 <- table(test_set$loan_status,pred_cutoff_15)
sum(diag(tab_multi_2)) / nrow(test_set)
```

```
## [1] 0.7243305
```

```
tab_multi_2[2,2]/sum(tab_multi_2[2,])
```

```
## [1] 0.833406
```

```
tab_multi_2[1,1]/sum(tab_multi_2[1,])
```

```
## [1] 0.694102
```

The accuracy of the multiple regression model is 72%, sensitivity is 83%, specificity is 69%.

Using link functions

Generalized linear models include a link function that relates the expected value of the response to the linear predictors in the model. A link function transforms the probabilities of the levels of a categorical response variable to a continuous scale that is unbounded. Once the transformation is complete, the relationship between the predictors and the response can be modeled with linear regression. For example, a binary response variable can have two unique values. Conversion of these values to probabilities makes the response variable range from 0 to 1.

Logit The purpose of the logit link is to take a linear combination of the covariate values (which may take any value between - infinity to + infinity) and convert those values to the scale of a probability, i.e., between 0 and 1.

```
log_model_logit <- glm(loan_status ~ person_age+loan_int_rate +
                         loan_grade_B + loan_grade_C + loan_grade_E +
                         loan_amnt + person_income +
                         person_home_ownership_OWN +
                         person_home_ownership_RENT +
                         loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE,family = binor
predictions_logit <- predict(log_model_logit, newdata = test_set,
                             type = "response")
class_pred_logit <- ifelse(predictions_logit > 0.15, 1, 0)
tab_class_logit <- table(test_set$loan_status,class_pred_logit)
acc_logit <- sum(diag(tab_class_logit)) / nrow(test_set)
acc_logit
```

```
## [1] 0.7241412
```

Logit model has an accuracy of 73.2%

Probit Probit regression, also called a probit model, is used to model dichotomous or binary outcome variables. In the probit model, the inverse standard normal distribution of the probability is modeled as a linear combination of the predictors.

```
log_model_probit <- glm(loan_status ~ loan_int_rate + loan_grade_B +
                         loan_grade_C + loan_grade_E + loan_amnt +
                         person_income + person_home_ownership_OWN + person_home_ownership_RENT + loan_
predictions_probit <- predict(log_model_probit, newdata = test_set,
                             type = "response")
class_pred_probit <- ifelse(predictions_probit > 0.15, 1, 0)
tab_class_probit <- table(test_set$loan_status,class_pred_probit)
acc_probit <- sum(diag(tab_class_probit)) / nrow(test_set)
acc_probit
```

```
## [1] 0.708148
```

Probit link model has an accuracy of 71%

Cloglog Probit regression, also called a probit model, is used to model dichotomous or binary outcome variables. In the probit model, the inverse standard normal distribution of the probability is modeled as a linear combination of the predictors.

```
log_model_cloglog <- glm(loan_status ~ loan_int_rate + loan_grade_B +
                          loan_grade_C + loan_grade_E + loan_amnt +
                          person_income + person_home_ownership_OWN + person_home_ownership_RENT + loan
                        data = training_set)
```

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

```
predictions_cloglog <- predict(log_model_cloglog, newdata = test_set,
                              type = "response")
class_pred_cloglog <- ifelse(predictions_cloglog > 0.15, 1, 0)
tab_class_cloglog <- table(test_set$loan_status,class_pred_cloglog)
acc_cloglog <- sum(diag(tab_class_cloglog)) / nrow(test_set)
acc_cloglog
```

```
## [1] 0.7181792
```

Cloglog model has an accuracy of 72.3%

```
anova(log_model_multi_2,log_model_logit, test = "Chisq")
```

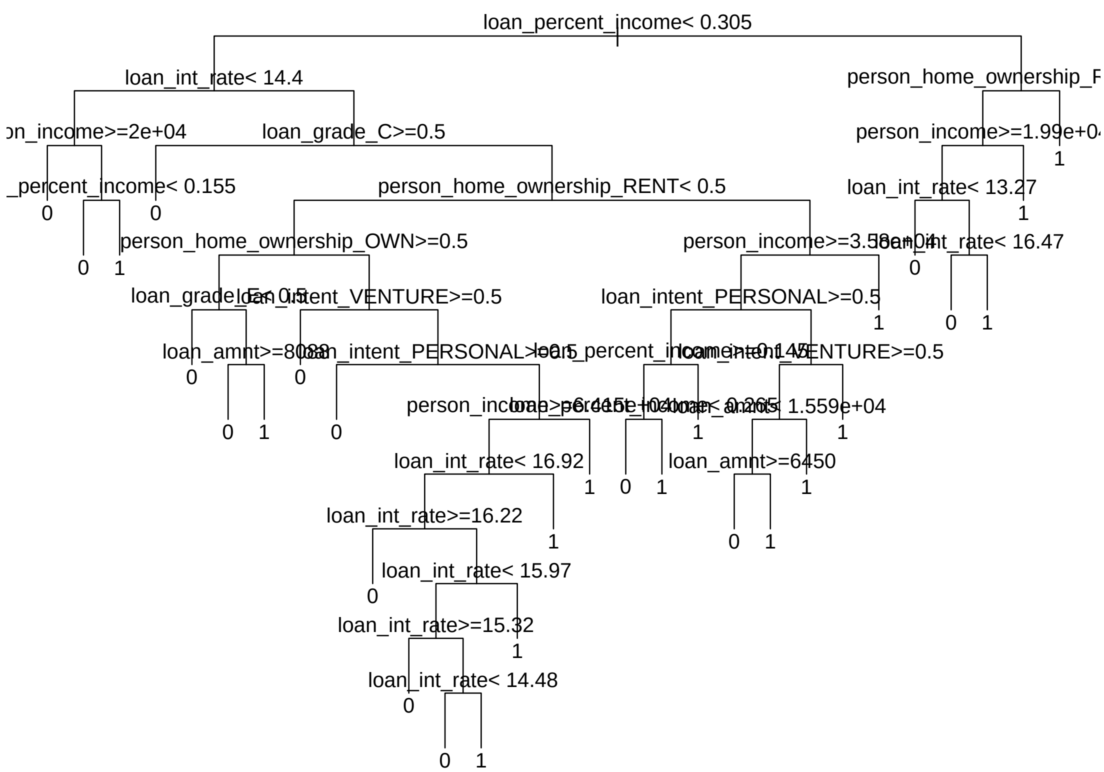
```
## Analysis of Deviance Table
##
## Model 1: loan_status ~ loan_int_rate + loan_grade_B + loan_grade_C + loan_grade_E +
##     loan_amnt + person_income + person_home_ownership_OWN + person_home_ownership_RENT +
##     loan_percent_income + loan_intent_PERSONAL + loan_intent_VENTURE
## Model 2: loan_status ~ person_age + loan_int_rate + loan_grade_B + loan_grade_C +
##     loan_grade_E + loan_amnt + person_income + person_home_ownership_OWN +
##     person_home_ownership_RENT + loan_percent_income + loan_intent_PERSONAL +
##     loan_intent_VENTURE
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1     21705      15248
## 2     21704      15248  1  0.13813   0.7102
```

So when we compare logit model which is the best model in the above link function models with the best model in our initial analysis

Decision Trees

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

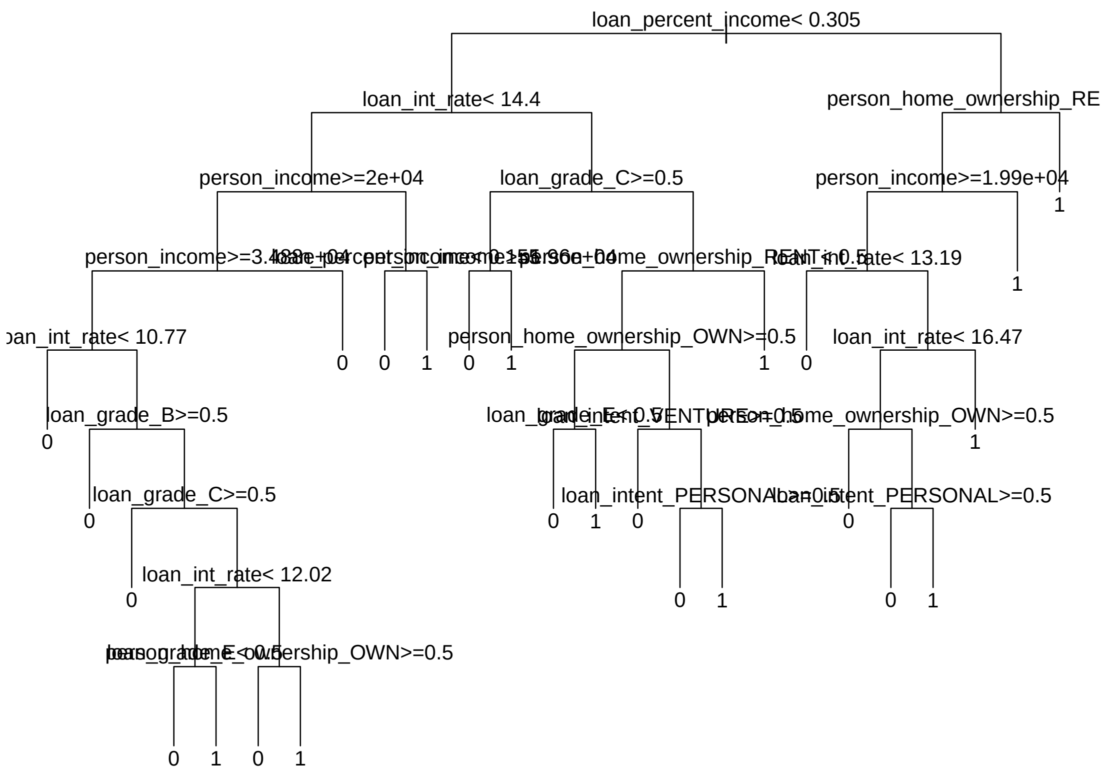
```
library(rpart)
tree_undersample <- rpart(loan_status ~ loan_int_rate + loan_grade_B +
                             loan_grade_C + loan_grade_E + loan_amnt +
                             person_income + person_home_ownership_OWN + person_home_ownership_RENT + loa
                           control = rpart.control(cp = 0.001))
plot(tree_undersample,uniform = TRUE)
text(tree_undersample)
```



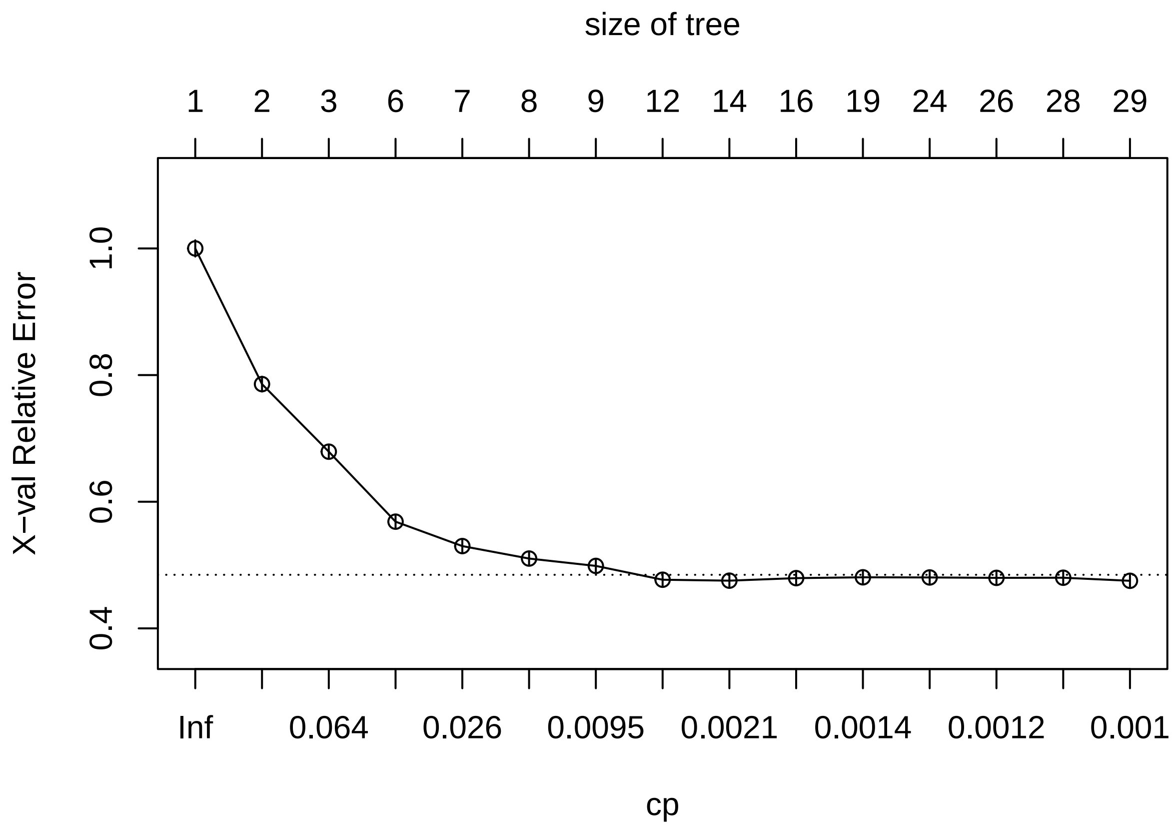
We can have a splitting index to improve our model for classification problems.

```
library(rpart)
tree_undersample_1 <- rpart(loan_status ~ loan_int_rate + loan_grade_B +
                               loan_grade_C + loan_grade_E + loan_amnt +
                               person_income + person_home_ownership_OWN + person_home_ownership_RENT + l
                             parms = list(prior = c(.7,.3)),
                             control = rpart.control(cp = 0.001))
```

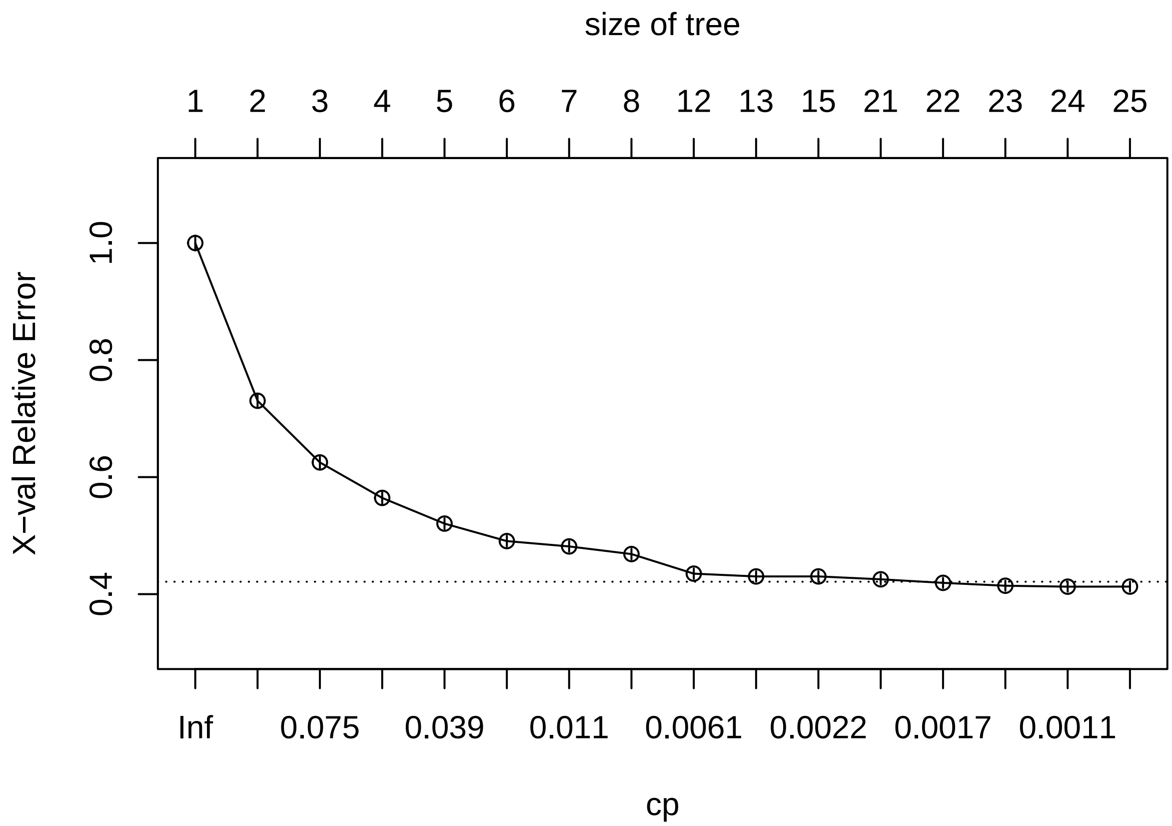
```
plot(tree_undersample_1,uniform = TRUE)
text(tree_undersample_1)
```




```
plotcp(tree_undersample)
```

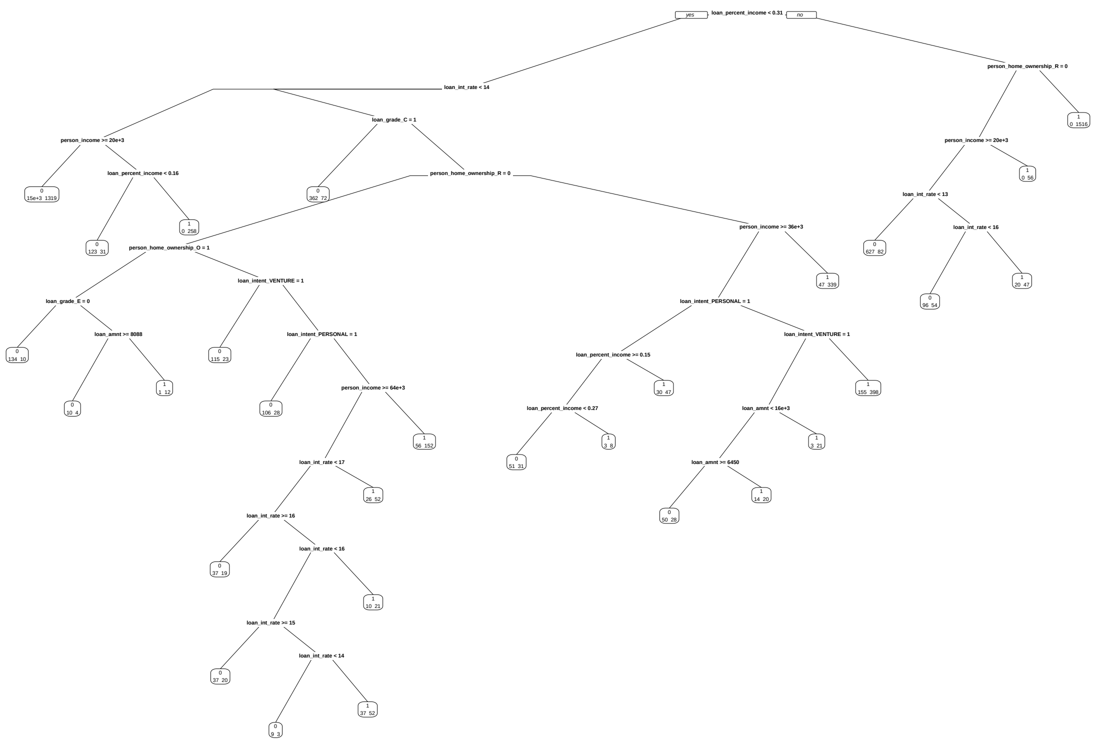



```
plotcp(tree_undersample_1)
```

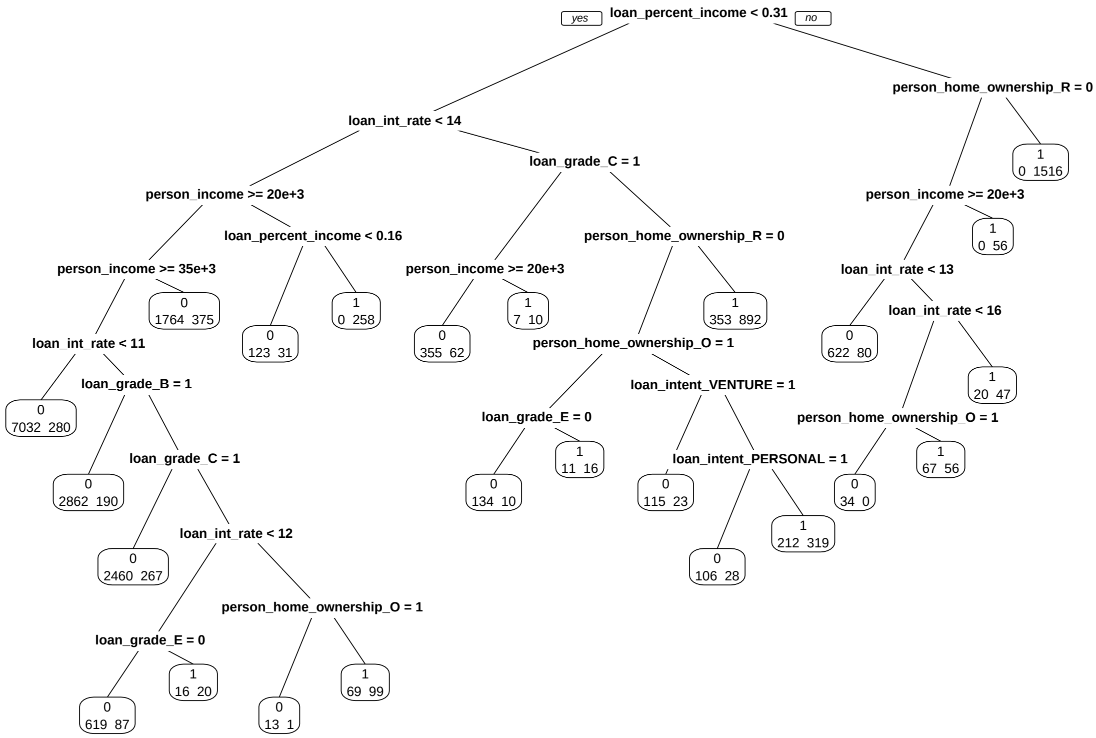


Let us prune the tree to get clear understanding of the tree.

```
library(rpart.plot)
index <- which.min(tree_undersample$cptable[ , "xerror"])
tree_min <- tree_undersample$cptable[index, "CP"]
ptree_undersample <- prune(tree_undersample, cp = tree_min)
prp(ptree_undersample,extra =1)
```




```r
index <- which.min(tree_undersample_1$cptable[ , "xerror"])
tree_min <- tree_undersample_1$cptable[index, "CP"]
ptree_undersample_1 <- prune(tree_undersample_1, cp = tree_min)
prp(ptree_undersample_1,extra =1)
```



We can predct the test set to find the acuracy of the ecision trees.

```
pred_undersample <- predict(ptree_undersample, newdata = test_set,
                            type = "class")
pred_undersample_1 <- predict(ptree_undersample_1, newdata = test_set,
                              type = "class")

confmat_undersample <- table(test_set$loan_status, pred_undersample)
confmat_undersample_1 <- table(test_set$loan_status, pred_undersample_1)

acc_undersample <- sum(diag(confmat_undersample)) / nrow(test_set)
acc_undersample_1 <- sum(diag(confmat_undersample_1)) / nrow(test_set)

acc_undersample
```

```
## [1] 0.8958077
```

```
acc_undersample_1
```

```
## [1] 0.8941043
```

The undersampletree_1 has a higher accuracy. Now, we can find the bad_rate which is the percentage of accounts that perform in an unsatisfactory manner as defined by the good/bad definition that was used in the scorecard development.

```r
prob_default_undersample <- predict(ptree_undersample_1,
                                    newdata = test_set)[ ,2]
cutoff_undersample = quantile(prob_default_undersample,
                              probs = 0.8)
bin_pred_us_80 <- ifelse(prob_default_undersample > cutoff_undersample,
                         1, 0)
accepted_status_us_80 <- test_set$loan_status[bin_pred_us_80 == 0]
bad_rate <- sum(accepted_status_us_80)/length(accepted_status_us_80)

bad_rate
```

```
## [1] 0.0783736
```

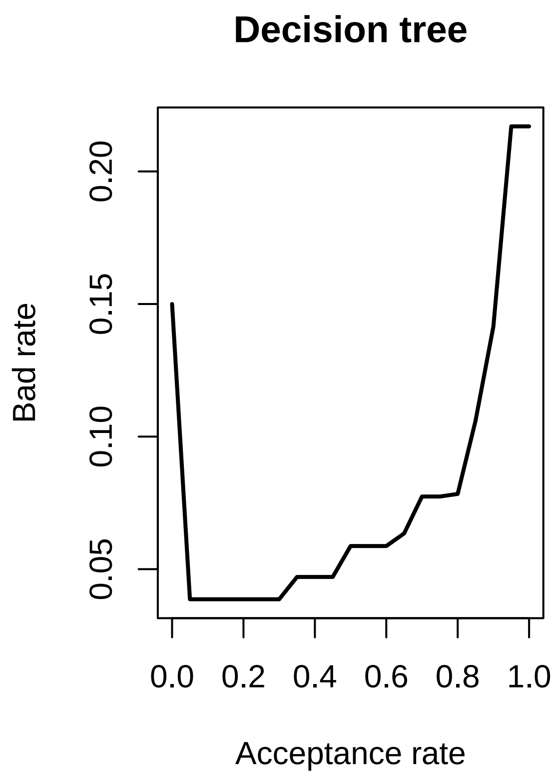
We can calculate the acceptance rates abd bad rates for all probabilities

```r
strategy_bank <- function(prob_of_def){
cutoff=rep(NA, 21)
bad_rate=rep(NA, 21)
accept_rate=seq(1,0,by=-0.05)
for (i in 1:21){
  cutoff[i]=quantile(prob_of_def,accept_rate[i])
  pred_i=ifelse(prob_of_def> cutoff[i], 1, 0)
  pred_as_good=test_set$loan_status[pred_i==0]
  bad_rate[i]=sum(pred_as_good)/length(pred_as_good)}
table=cbind(accept_rate,cutoff=round(cutoff,4),bad_rate=round(bad_rate,4))
return(list(table=table,bad_rate=bad_rate, accept_rate=accept_rate, cutoff=cutoff))
}
```

```r
strategy_undersample <- strategy_bank(prob_default_undersample)
strategy_undersample$table
```

```
##       accept_rate cutoff bad_rate
##  [1,]        1.00 1.0000   0.2170
##  [2,]        0.95 1.0000   0.2170
##  [3,]        0.90 0.7958   0.1415
##  [4,]        0.85 0.7837   0.1060
##  [5,]        0.80 0.2894   0.0784
##  [6,]        0.75 0.2469   0.0774
##  [7,]        0.70 0.2469   0.0774
##  [8,]        0.65 0.1781   0.0635
##  [9,]        0.60 0.1434   0.0587
## [10,]        0.55 0.1434   0.0587
## [11,]        0.50 0.1434   0.0587
## [12,]        0.45 0.0929   0.0471
## [13,]        0.40 0.0929   0.0471
## [14,]        0.35 0.0929   0.0471
## [15,]        0.30 0.0578   0.0387
## [16,]        0.25 0.0578   0.0387
## [17,]        0.20 0.0578   0.0387
## [18,]        0.15 0.0578   0.0387
## [19,]        0.10 0.0578   0.0387
## [20,]        0.05 0.0578   0.0387
## [21,]        0.00 0.0000   0.1500
```

```r
par(mfrow = c(1,2))
plot(strategy_undersample$accept_rate, strategy_undersample$bad_rate,
     type = "l", xlab = "Acceptance rate", ylab = "Bad rate",
     lwd = 2, main = "Decision tree")
```



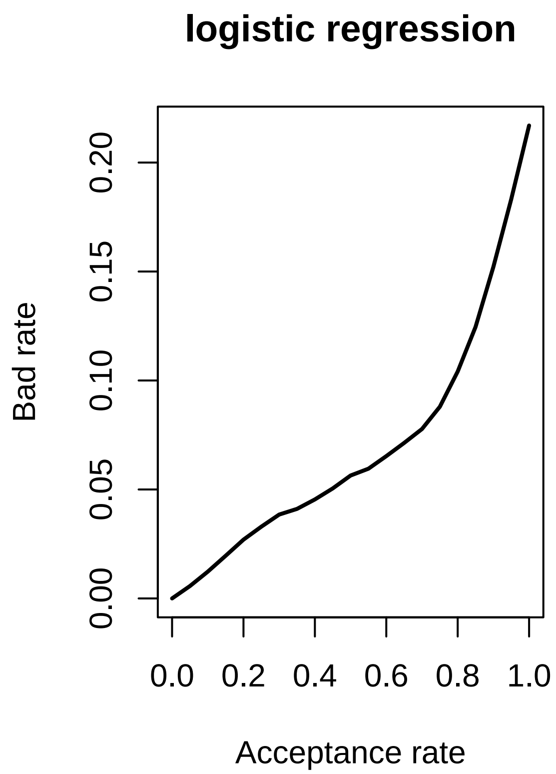
If 80% of all loan applications are accepted then 5% of them will default.

```r
strategy_logit <- strategy_bank(predictions_logit)
strategy_logit$table
```

```
##       accept_rate cutoff bad_rate
##  [1,]        1.00 0.9975   0.2170
##  [2,]        0.95 0.8098   0.1832
##  [3,]        0.90 0.6600   0.1522
##  [4,]        0.85 0.5179   0.1246
##  [5,]        0.80 0.4018   0.1041
##  [6,]        0.75 0.3147   0.0879
##  [7,]        0.70 0.2460   0.0777
##  [8,]        0.65 0.1992   0.0713
##  [9,]        0.60 0.1627   0.0653
## [10,]        0.55 0.1333   0.0595
## [11,]        0.50 0.1110   0.0564
## [12,]        0.45 0.0941   0.0505
## [13,]        0.40 0.0787   0.0454
## [14,]        0.35 0.0655   0.0411
## [15,]        0.30 0.0538   0.0385
```

```
## [16,]        0.25 0.0443   0.0329
## [17,]        0.20 0.0348   0.0270
## [18,]        0.15 0.0271   0.0196
## [19,]        0.10 0.0200   0.0123
## [20,]        0.05 0.0126   0.0057
## [21,]        0.00 0.0003   0.0000
```

```
par(mfrow = c(1,2))
plot(strategy_logit$accept_rate, strategy_logit$bad_rate,
     type = "l", xlab = "Acceptance rate", ylab = "Bad rate",
     lwd = 2, main = "logistic regression")
```



The logit model predicts that 5% of the loans will be defaulted if 45% of all loans are accepted.

We can compare all the link models and undersample_1 decision tree at once using the criteria of area under the curve.

```
library(pROC)
```

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

```
##
## Attaching package: 'pROC'
```

```
## The following object is masked from 'package:gmodels':
##
##     ci
```

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

```
ROC_logit <- roc(test_set$loan_status, predictions_logit)
```

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

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

```
ROC_probit <- roc(test_set$loan_status, predictions_probit)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

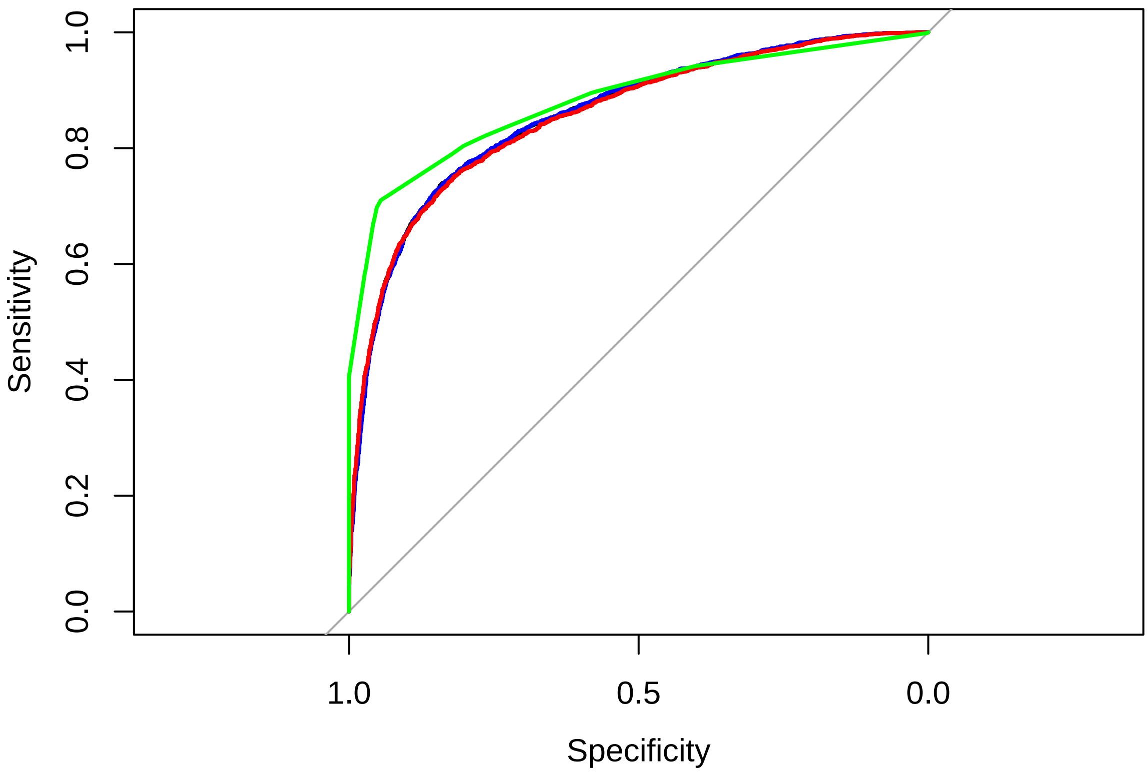
```
ROC_cloglog <-roc(test_set$loan_status, predictions_cloglog)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
ROC_tree <-roc(test_set$loan_status, prob_default_undersample)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
plot(ROC_logit)
lines(ROC_probit, col="blue")
lines(ROC_cloglog, col="red")
lines(ROC_tree,col = "green")
```




```
auc(ROC_logit)
```

```
## Area under the curve: 0.8575
```

```
auc(ROC_probit)
```

```
## Area under the curve: 0.8579
```

```
auc(ROC_cloglog)
```

```
## Area under the curve: 0.8554
```

```
auc(ROC_tree)
```

```
## Area under the curve: 0.8825
```

Logit regression has the highest area under the curve so it is the best model here.

GAM's

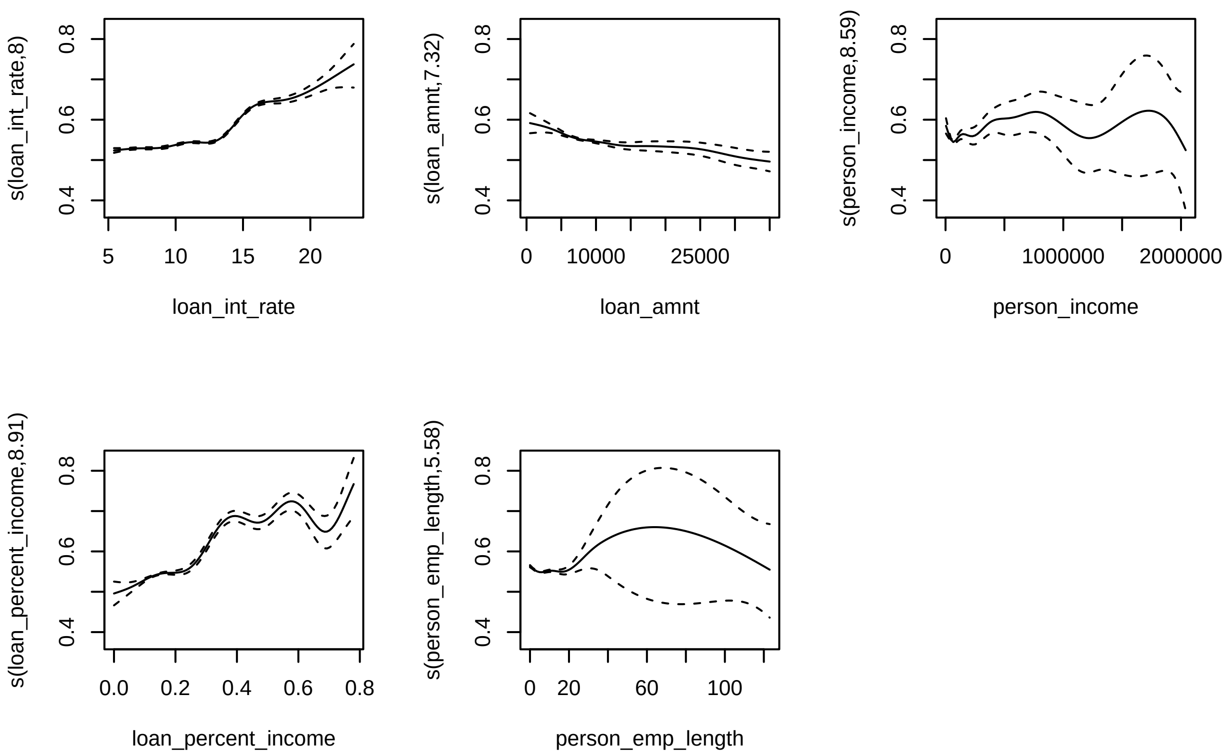
GAM is an additive modeling technique where the impact of the predictive variables is captured through smooth functions which—depending on the underlying patterns in the data—can be nonlinear. GAM's are used for interpretability, flexibility and regularization.

```r
library(mgcv)
```

```
## Loading required package: nlme

## This is mgcv 1.8-38. For overview type 'help("mgcv-package")'.
```

```r
gam_mod<- gam(loan_status ~ s(loan_int_rate)+ s(loan_amnt) +
                          s(person_income)+s(loan_percent_income)+
                s(person_emp_length), data = training_set, na.rm =TRUE)
plot(gam_mod, pages = 1, trans = plogis,shift = coef(gam_mod)[1], seWithMean = TRUE)
```




```r
summary(gam_mod)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## loan_status ~ s(loan_int_rate) + s(loan_amnt) + s(person_income) +
##     s(loan_percent_income) + s(person_emp_length)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 0.214692   0.002316   92.71   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                          edf Ref.df       F p-value
## s(loan_int_rate)       7.997  8.585 407.784  <2e-16 ***
## s(loan_amnt)           7.315  8.330   5.677  <2e-16 ***
## s(person_income)       8.595  8.941  21.876  <2e-16 ***
## s(loan_percent_income) 8.906  8.994 245.559  <2e-16 ***
## s(person_emp_length)   5.577  6.338  12.917  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.328   Deviance explained =   33%
## GCV = 0.11345  Scale est. = 0.11324   n = 21114
```

The plots show how each variable is related to the response variable (loan_status). We apply the plogis transformation to get model the data into 0-1 bound as we have a binary response variable. We can find predictions using the model.

```
test_set <- na.omit(test_set)
test_predict <- predict(gam_mod, type="terms",newdata = test_set, se.fit = TRUE)
high_pred <- test_predict$fit + 2*test_predict$se.fit
low_pred <- test_predict$fit - 2*test_predict$se.fit
high_prob <- plogis(high_pred)
low_prob <- plogis(low_pred)
head(high_prob)
```

```
##    s(loan_int_rate) s(loan_amnt) s(person_income) s(loan_percent_income)
## 9         0.4798057    0.4668714        0.4995163              0.6499112
## 13        0.4779965    0.4668714        0.5054206              0.6470371
## 15        0.4779965    0.4668714        0.5149835              0.5693816
## 18        0.6430150    0.4668714        0.5039466              0.5965727
## 20        0.4922471    0.5525591        0.5423037              0.4950388
## 21        0.4780241    0.4668714        0.5251660              0.5025187
##    s(person_emp_length)
## 9             0.4986939
## 13            0.5030850
## 15            0.5030850
## 18            0.4972927
## 20            0.4986939
## 21            0.5030850
```

```
head(low_prob)
```

```
##    s(loan_int_rate) s(loan_amnt) s(person_income) s(loan_percent_income)
## 9         0.4739133    0.4189580        0.4882320              0.6195523
## 13        0.4730752    0.4189580        0.4910523              0.6213895
## 15        0.4730752    0.4189580        0.4964129              0.5482840
## 18        0.6128288    0.4189580        0.4902664              0.5743950
## 20        0.4885348    0.5150059        0.5090356              0.4896401
```

```
## 21        0.4729592    0.4189580        0.4961559              0.4891505
##    s(person_emp_length)
## 9             0.4944282
## 13            0.5008280
## 15            0.5008280
## 18            0.4937268
## 20            0.4944282
## 21            0.5008280
```

So, the percentage of income which has to be used from income is the most influential factor. Additional analysis can be done using gams but due to time constraint I couldnt explore GAMS in detail. But we will continue to surivival analysis.

Survival analysis

Survival analysis is modelling of the time to death. But survival analysis has a much broader use in statistics. Any event can be defined as death. For example, age for marriage, time for the customer to buy his first product after visiting the website for the first time, time to attrition of an employee etc. All can be modeled as survival analysis. In this project we will use survival analysis to find out when each person would default based on their credit history. For this project we can define death as the time for the customer to default.

```r
library(survival)
library(survminer)
```

```
## Warning: package 'survminer' was built under R version 4.1.3
```

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

```
## Loading required package: ggpubr
```

```
## Warning: package 'ggpubr' was built under R version 4.1.3
```

```
##
## Attaching package: 'survminer'
```

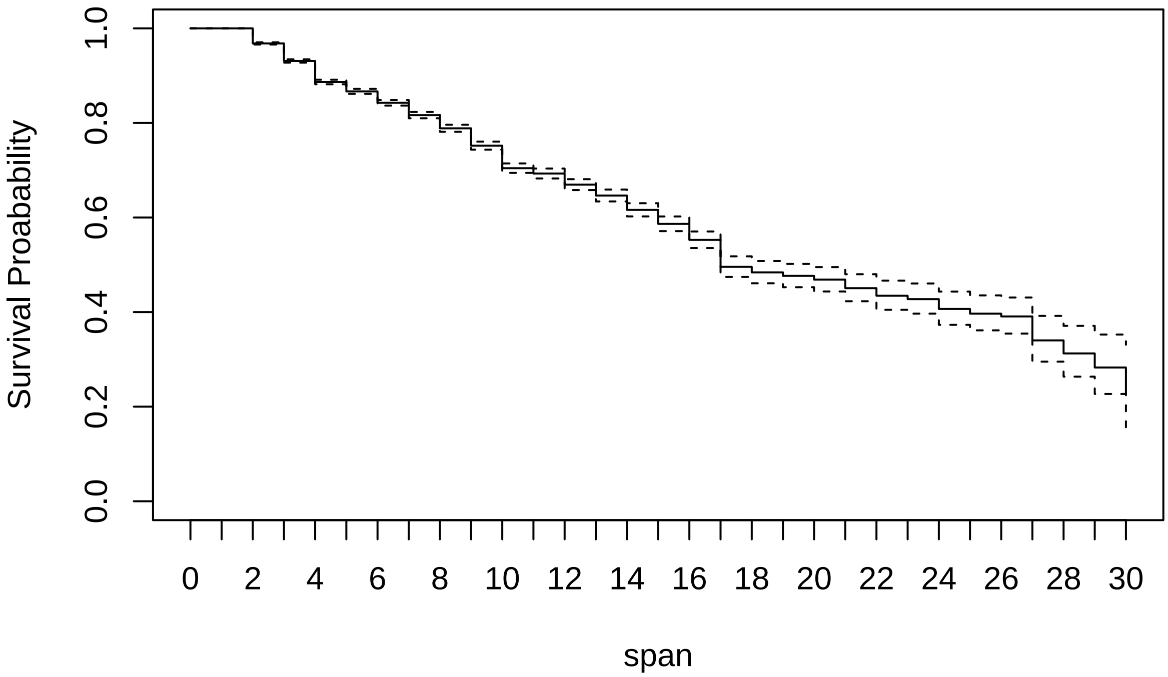
```
## The following object is masked from 'package:survival':
##
##     myeloma
```

```r
kmsurv <- survfit(Surv(training_set$cb_person_cred_hist_length,
                       training_set$cb_person_default_on_file_Y ) ~ 1)
summary(kmsurv)
```

```
## Call: survfit(formula = Surv(training_set$cb_person_cred_hist_length,
##     training_set$cb_person_default_on_file_Y) ~ 1)
##
##  time n.risk n.event survival std.err lower 95% CI upper 95% CI
##     2  21717     691    0.968 0.00119        0.966        0.971
##     3  17707     680    0.931 0.00181        0.927        0.935
##     4  13779     658    0.887 0.00241        0.882        0.891
```

```
##      5   9825     220     0.867 0.00270         0.861         0.872
##      6   8571     239     0.843 0.00305         0.837         0.849
##      7   7332     226     0.817 0.00341         0.810         0.823
##      8   6052     207     0.789 0.00380         0.781         0.796
##      9   4824     225     0.752 0.00435         0.743         0.760
##     10   3558     225     0.704 0.00510         0.694         0.714
##     11   2298      37     0.693 0.00535         0.683         0.704
##     12   2007      68     0.669 0.00587         0.658         0.681
##     13   1682      58     0.646 0.00641         0.634         0.659
##     14   1388      65     0.616 0.00712         0.602         0.630
##     15   1061      51     0.587 0.00790         0.571         0.602
##     16    780      45     0.553 0.00891         0.535         0.570
##     17    475      49     0.496 0.01110         0.474         0.518
##     18    212       5     0.484 0.01201         0.461         0.508
##     19    197       3     0.477 0.01256         0.453         0.502
##     20    180       3     0.469 0.01316         0.444         0.495
##     21    156       6     0.451 0.01457         0.423         0.480
##     22    140       5     0.435 0.01572         0.405         0.466
##     23    122       2     0.427 0.01625         0.397         0.460
##     24    103       5     0.407 0.01792         0.373         0.443
##     25     81       2     0.397 0.01883         0.361         0.435
##     26     68       1     0.391 0.01944         0.354         0.431
##     27     54       7     0.340 0.02460         0.295         0.392
##     28     37       3     0.313 0.02728         0.263         0.371
##     29     21       2     0.283 0.03178         0.227         0.352
##     30     10       2     0.226 0.04388         0.155         0.331
```

```
plot(kmsurv,xlab = "span",ylab="Survival Proabability",xaxp = c(0, 30,30))
```



Facts about the population :

1. All customers who have a credit history of less than 2 years (age of customers would be 20) do not default at all.
2. 59% of the population has gone into debt by the age of 32 (14 years of credit history)
3. 31% of the population survives even after 30 years of credit history (around 48 years)

References

1. campus.datacamp.com
2. noamross.github.io
3. cran.r-project.org
4. ideas.repec.org
5. www.programmingr.com
6. stackoverflow.com (for errors)
7. www.sthda.com
8. www.r-bloggers.com
9. stats.stackexchange.com
10. www.analyticsvidhya.com
11. www.analyticsvidhya.com