R-view on prepayment and interest rate risk in Mortgage Backed Securities

Smitha Shivakumar

Mortgage Backed Securities

Mortgage-backed securities (MBS), are bonds that represent an investment in a group of home loans. These asset-backed securities are formed when lending banks bundle their mortgages into pools and sell them to investment banks or government agencies in the form of a bond.

The banks categorize the loans according to credit ratings and sell them to investors. These tradable asset-backed securities are built around a collection of mortgages and then made available to the public for trade. In essence, it’s a way for individual investors to invest in mortgages without having to actually issue or purchase them.

Interest Rate Risk

Prices of securities are exposed to fluctuation of interest rate and so does MBS, particularly for the fixed rate MBS which is included in the following valuation part. When interest rate increases, the investors of MBS will receive lower return for the unchanged interest rate, inversely, when the interest rate drops, investors will benefit from the fixed rate of loans. In order to avoid such kind of risk, we need to simulate the path of market interest rate and to give a reasonable discount rate of the loans.

Prepayment Risk

Prepayment refers to the activity that borrower of mortgage loans pay more than the scheduled monthly payment amount, where the excess part will be used to pay the remaining outstanding balance principle which leads principle be paid faster than original scheduled amortization. Prepayment is one of the risk factor that will interrupt the scheduled cash flow of MBS influencing the structure and profitability.

Using a model

Black Derman Toy Model

- Black-Derman-Toy model (a short rate model) is a model of the evolution of the yield curve.
- It is a single stochastic factor (the short rate) determines the future evolution of all interest rates.
- The parameters can be calibrated to fit the current term structure of interest rates (yield curve) and volatility structure as derived from implied prices (Black-76) for interest rate caps.
- The model was introduced by Fischer Black, Emanuel Derman, and Bill Toy in the Financial Analysts Journal in 1990.

BDT from Models for Financial Economics (m4fe) library

```
library(m4fe)

bdt(yields=c(0.10, 0.11, 0.12, 0.125), volatilities=c(NA, 0.10, 0.15, 0.14))
```

`yields`	The historical zero-coupon bond yields
`volatilities`	The volatilities in force for each of the zero-coupon bonds at time 1. Note that volatilites[1] is undefined

Data Sources

Mortgage Rates:

- http://www.federalreserve.gov/releases/h15/data/Weekly_Friday_/H15_MORTG_NA.txt

Treasury Yields:

- http://www.ustreas.gov/offices/domestic-finance/ debt-management/interest-rate/yield_historical_huge.shtml

PSA model

- Prepayment influences the cash flow of MBS significantly as it is a crucial part in pricing. PSA will be the model used to calculate prepayment cash flow in the paper because the model fits the reality better.
- In PSA100 model, we assume that the prepayment rate starts from 0.2%, increase evenly during the next 29 months to 6% and remain constant at 6% in the rest of loan life, which can be summarized as:

$$CPR = t * 6\%/30, \quad t \leq 30 \qquad\qquad CPR = 6\%, \quad t > 30$$

- The 100 in PSA100 indicates a multiplier of the loan prepayment speed. 150% PSA would assume 0.3% (1.5 x 0.2%) increases to a peak of 9%, and 200% PSA would assume 0.4% (2 x 0.2%) increases to a peak of a 12% prepayment rate

PSA model packages

```
install.packages("remotes")
remotes::install_github("glennmschultz/BondLab")
```

BondLab’s prepayment model

```
signature("PrepaymentModel"),
function(.Object,
         PrepaymentAssumption = "character",
         PPCStart = numeric(),
         PPCEnd = numeric(),
         PPCSeasoning = numeric(),
         FirstPmtDate = "character",
         LastPmtDate = "character",
         FinalPmtDate = "character",
         PmtDate = "character",
         LoanAge = numeric(),
         Period = numeric(),
         NoteRate = numeric(),
         MtgRateFwd = numeric(),
         Incentive = numeric(),
         SMM = numeric(),
         MDR = numeric(),
         Severity = numeric())
```

Building a model

Data

Loan data for all loans issued through the 2007-2015, including the current loan status (Current, Late, Fully Paid, etc.) and latest payment information. The file containing loan data through the "present" contains complete loan data for all loans issued through the previous completed calendar quarter. Additional features include credit scores, number of finance inquiries, address including zip codes, and state, and collections among others.

The file is a matrix of about 890 thousand observations and 75 variables.

Feature Selection & Engineering

- The dataset consists of information of age, annual income, grade of employee, home ownership that affect the probability of default of the borrower. The columns we are going to use are :

loan_status : Variable with multiple levels (e.g. Charged off, Current, Default, Fully Paid …)	**loan_amnt :** Total amount of loan taken
int_rate : Loan interest rate	**grade :** Grade of employment
emp_length : Duration of employment	**home_ownership :** Type of ownership of house
annual_inc : Total annual income	**term :** 36-month or 60-month period

Modelling Process

- Created the binary loan_outcome which will be our response variable.
- Excluded some independent variables in order to make the model simpler.
- Split the dataset to training set(75%) and testing set(25%) for the validation.
- Trained a model to predict the probability of default.
- Because of the binary response variable we can use logistic regression. Rather than modelling the response Y directly, logistic regression models the probability that Y belongs to a particular category, in our case the probability of a non-performing loan. This probability can be computed by the logistic function,

$$P = \exp(b0 + b1X1 + \ldots + bNXN) / [\, 1 + \exp(b0 + b1X1 + \ldots + bNXN) \,]$$

where, P is the probability of default

b0 , b1 , … , bN are the coefficient estimates

N the number of observations

X1 , … , XN are the independent variables

Running a logistic regression

```r
# Split dataset
loan2$loan_outcome = as.numeric(loan2$loan_outcome)
idx = sample(dim(loan2)[1] , 0.75*dim(loan2)[1] , replace = F)
trainset = loan2[idx , ]
testset = loan2[-idx , ]

# Fit logistic regression
glm.model = glm(loan_outcome ~ . , trainset , family = binomial(link = 'logit'))
summary(glm.model)
```

```
##
## Call:
## glm(formula = loan_outcome ~ ., family = binomial(link = "logit"),
##     data = trainset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4571  -0.7102  -0.5325  -0.3196   6.9090
##
## Coefficients:
##                          Estimate Std. Error  z value Pr(>|z|)
## (Intercept)         -3.167e+00  2.007e-02 -157.824  < 2e-16 ***
## loan_amnt            1.166e-05  3.806e-07   30.643  < 2e-16 ***
## int_rate             3.962e-02  1.759e-03   22.522  < 2e-16 ***
## gradeB               6.685e-01  1.371e-02   48.747  < 2e-16 ***
## gradeC               1.071e+00  1.694e-02   63.256  < 2e-16 ***
## gradeD               1.283e+00  2.231e-02   57.509  < 2e-16 ***
## gradeE               1.402e+00  2.814e-02   49.825  < 2e-16 ***
## gradeF               1.447e+00  3.581e-02   40.408  < 2e-16 ***
## gradeG               1.488e+00  4.542e-02   32.768  < 2e-16 ***
## emp_length1 year    -4.485e-03  1.371e-02   -0.327 0.743644
## emp_length10+ years -5.825e-02  1.045e-02   -5.574 2.49e-08 ***
## emp_length2 years   -3.859e-02  1.271e-02   -3.037 0.002387 **
## emp_length3 years   -1.910e-02  1.311e-02   -1.457 0.145007
## emp_length4 years   -3.271e-02  1.421e-02   -2.301 0.021384 *
## emp_length5 years   -4.212e-02  1.406e-02   -2.996 0.002735 **
## emp_length6 years   -4.607e-02  1.538e-02   -2.995 0.002747 **
## emp_length7 years   -5.386e-02  1.563e-02   -3.447 0.000568 ***
## emp_length8 years    1.073e-02  1.543e-02    0.696 0.486728
## emp_length9 years   -1.128e-02  1.643e-02   -0.687 0.492122
## home_ownershipOTHER -1.628e-02  2.946e-01   -0.055 0.955927
## home_ownershipOWN    1.890e-01  9.532e-03   19.824  < 2e-16 ***
## home_ownershipRENT   3.522e-01  6.098e-03   57.751  < 2e-16 ***
## annual_inc          -2.280e-06  7.227e-08  -31.544  < 2e-16 ***
## term60 months        4.221e-01  6.848e-03   61.649  < 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: 912161  on 920912  degrees of freedom
## Residual deviance: 836605  on 920889  degrees of freedom
## AIC: 836653
##
## Number of Fisher Scoring iterations: 5
```

Coefficient Significance

The coefficients of the following features are **positive**:

- Loan Amount
- Interest Rate
- Home Ownership - Other
- Term
- Grade

The better the grade the more difficult to default. This means the probability of defaulting on the given credit varies directly with these factors. For example more the given amount of the loan, more the risk of losing credit.

Coefficient Significance

The coefficients of the following features are **negative**:

- Annual Income
- Home Ownership - Own
- Home Ownership - Rent
- Borrowers with 10+ years of experience are more likely to pay their debt

There is no significant difference in the early years of employment. This means that the probability of defaulting is inversely proportional to the factors mentioned above.

Accuracy, Sensitivity and Specificity

The accuracy, sensitivity and specificity are transformed for given threshold. We can use a threshold of 50% for the posterior probability of default in order to assign an observation to the default class. However, if we are concerned about incorrectly predicting the default status for individuals who default, then we can consider lowering this threshold. So we will consider these three metrics for threshold levels from 1% up to 50%.

```
k = 0
accuracy = c()
sensitivity = c()
specificity = c()
for(i in seq(from = 0.01 , to = 0.5 , by = 0.01)){
        k = k + 1
        preds_binomial = ifelse(preds > i , 1 , 0)
        confmat = table(testset$loan_outcome , preds_binomial)
        accuracy[k] = sum(diag(confmat)) / sum(confmat)
        sensitivity[k] = confmat[1 , 1] / sum(confmat[ , 1])
        specificity[k] = confmat[2 , 2] / sum(confmat[ , 2])
}
```

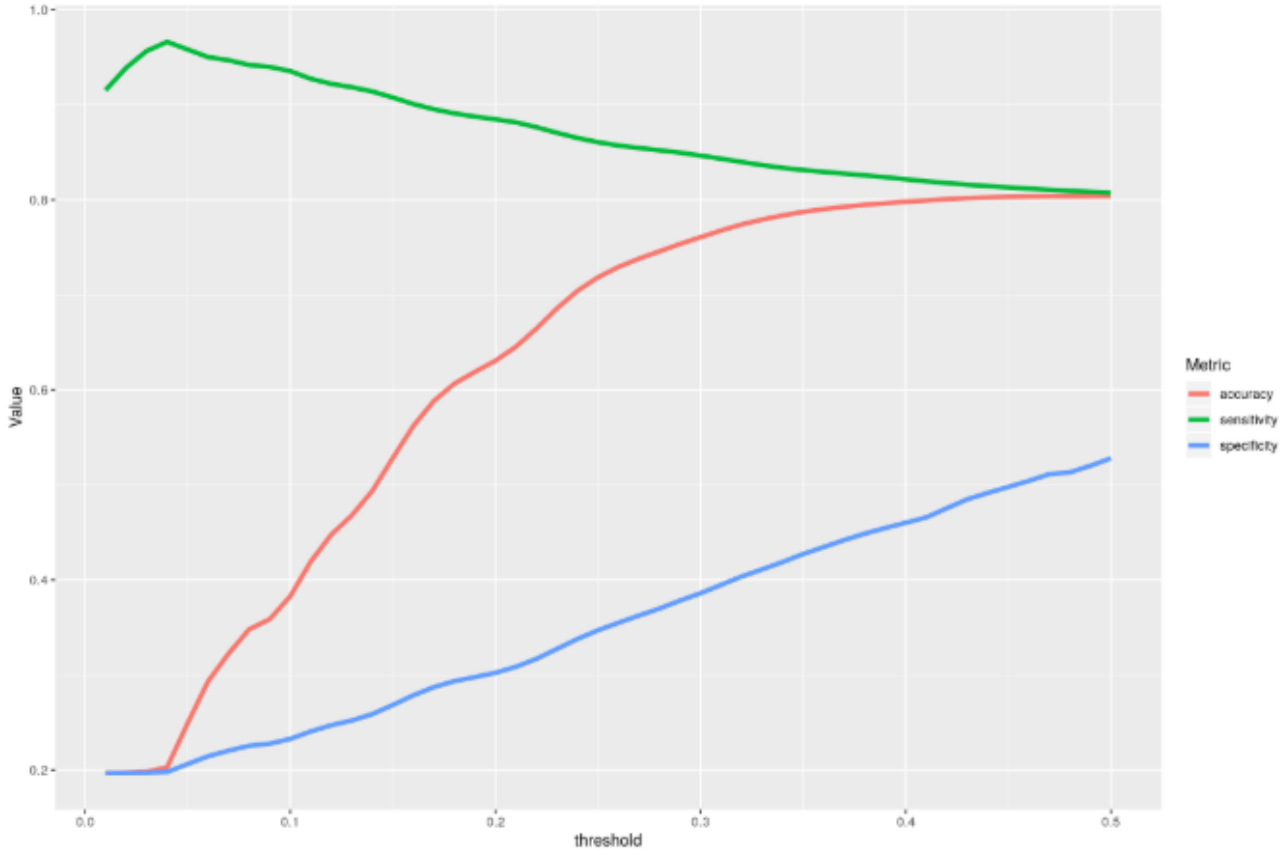
```
threshold = seq(from = 0.01 , to = 0.5 , by = 0.01)

data = data.frame(threshold , accuracy , sensitivity , specificity)
head(data)
```

```
##   threshold  accuracy sensitivity specificity
## 1      0.01 0.1970733   0.9152542   0.1969353
## 2      0.02 0.1973340   0.9387755   0.1969787
## 3      0.03 0.1980767   0.9565217   0.1971094
## 4      0.04 0.2029208   0.9661274   0.1979711
## 5      0.05 0.2491270   0.9580999   0.2062817
## 6      0.06 0.2935251   0.9500713   0.2145863
```

Accuracy, Sensitivity and Specificity

```
# Gather accuracy , sensitivity and specificity in one column
ggplot(gather(data , key = 'Metric' , value = 'Value' , 2:4) ,
       aes(x = threshold , y = Value , color = Metric)) +
        geom_line(size = 1.5)
```



Confusion Matrix and ROC

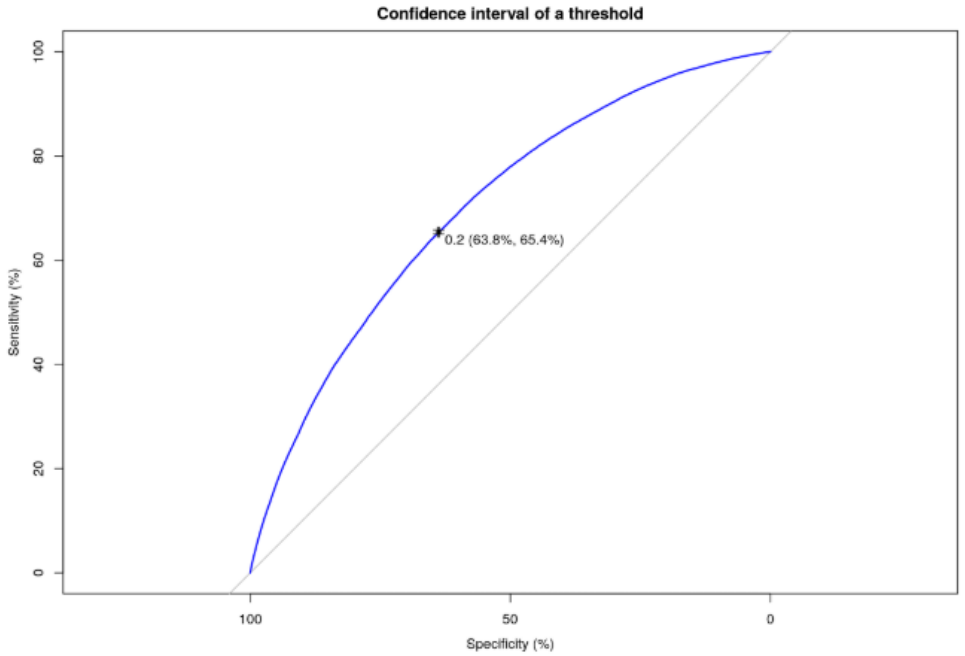
A threshold of 25% - 30% seems ideal cause further increase of the cut off percentage does not have significant impact on the accuracy of the model.
The Confusion Matrix for cut off point at 30% will be this

```
preds.for.30 = ifelse(preds > 0.3 , 1 , 0)
confusion_matrix_30 = table(Predicted = preds.for.30 , Actual = testset$loan_outcome)
confusion_matrix_30
```

```
##          Actual
## Predicted      0      1
##         0 211382  38351
##         1  35143  22096
```

```
## [1] "Accuracy : 0.7606"
```

```
# Plot ROC curve
plot.roc(testset$loan_outcome , preds , main = "Confidence interval of a threshold" , percent = TRUE ,
         ci = TRUE , of = "thresholds" , thresholds = "best" , print.thres = "best" , col = 'blue')
```



The *ROC (Receiver Operating Characteristics) curve* is a popular graphic for simultaneously displaying the two types of errors for all possible thresholds.

```
library(pROC)

# Area Under Curve
auc(roc(testset$loan_outcome , preds))
```

```
## Area under the curve: 0.7011
```

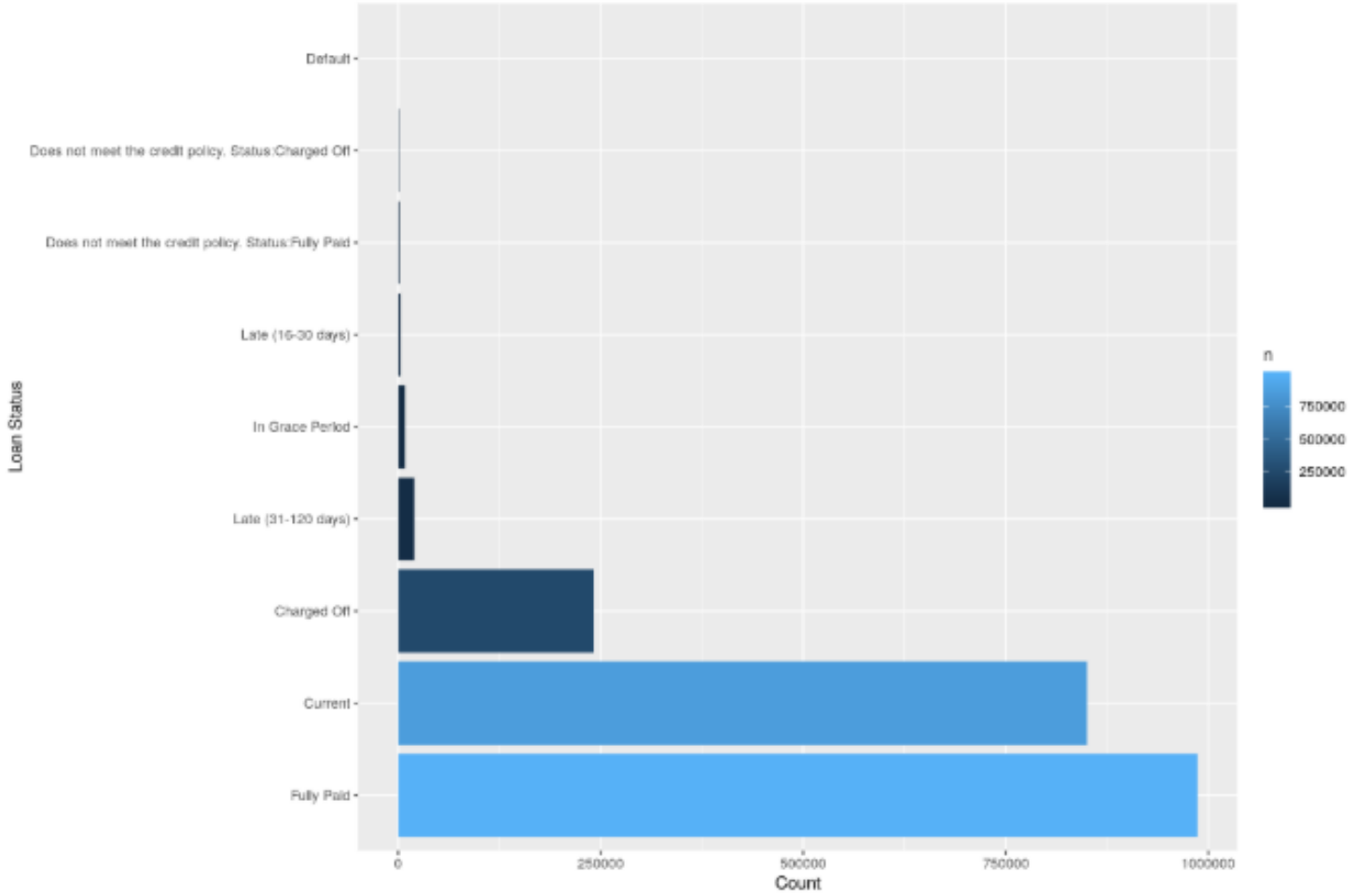
Model takeaways

A logistic regression model was used to predict the loan status. Different cut off's were used to decide if the loan should be granted or not. Cut off of 30% gave a good accuracy of 76.06%. The decision to set a cut off is arbitrary and higher levels of threshold increases the risk. The Area Under Curve also gives a measure of accuracy, which came out to be 70.11%.

APPENDIX

Loan status

```
loan %>%
        count(loan_status) %>%
        ggplot(aes(x = reorder(loan_status , desc(n)) , y = n , fill = n)) +
        geom_col() +
        coord_flip() +
        labs(x = 'Loan Status' , y = 'Count')
```



Checking loan status for Default and Fully Paid

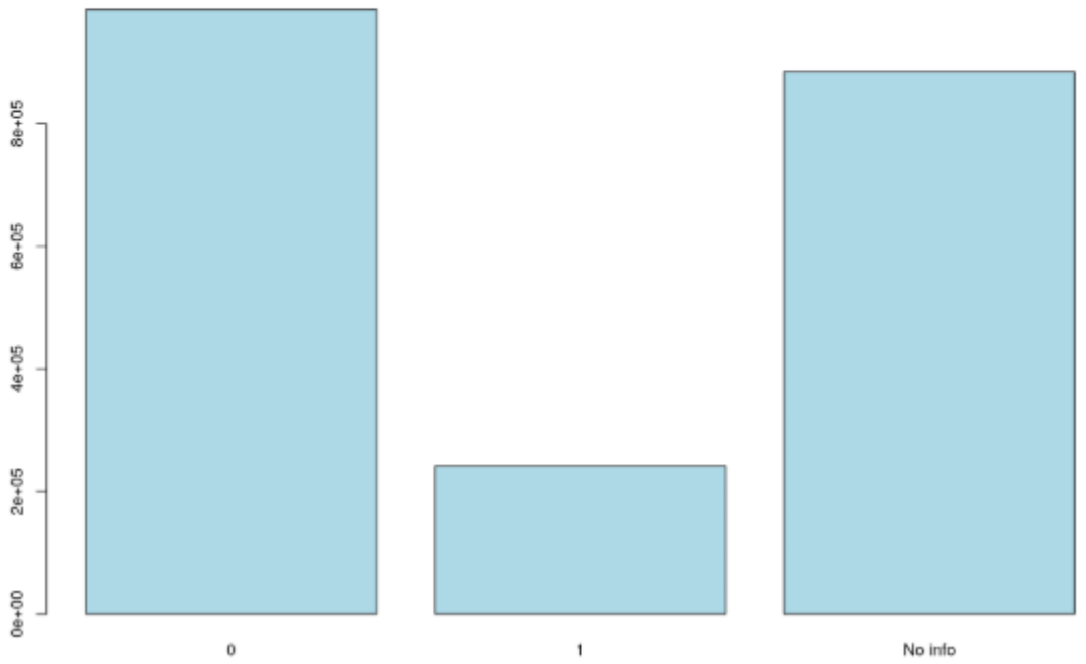
Converting this variable to binary (1 for default and 0 for non-default) but we have 10 different levels.

Loans with status Current, Late payments, In grace period need to be removed. Therefore, we create a new variable called loan_outcome where

loan_outcome -> 1 if loan_status = ‘Charged Off’ or ‘Default’ loan_outcome -> 0 if loan_status = ‘Fully Paid’

```
loan = loan %>%
       mutate(loan_outcome = ifelse(loan_status %in% c('Charged Off' , 'Default') ,
                                    1,
                                    ifelse(loan_status == 'Fully Paid' , 0 , 'No info')
                                    ))

barplot(table(loan$loan_outcome) , col = 'lightblue')
```



Relation between the interest rates and Grades

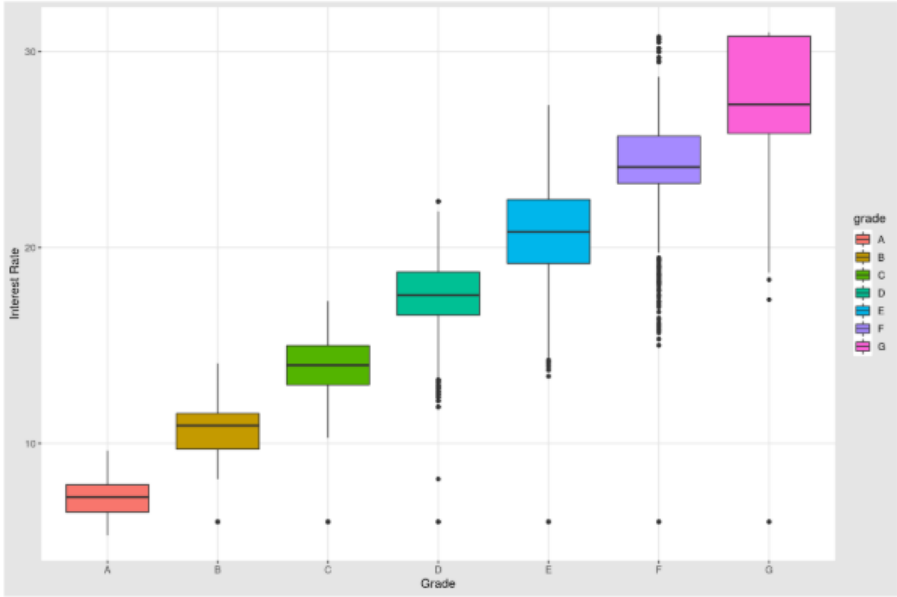
creating a new dataset which contains only rows with 0 or 1 in loan_outcome feature for better modelling.

Our new dataset contains of **1227885 rows**.

Let’s observe how useful these variables would be for credit risk modelling. It is known that the better the grade the lowest the interest rate. We can nicely visualise this with boxplots.

```
# Create the new dataset by filtering 0's and 1's in the loan_outcome column and remove loan_status colu
mn for the modelling
loan2 = loan %>%
        select(-loan_status) %>%
        filter(loan_outcome %in% c(0 , 1))
```

```
ggplot(loan2 , aes(x = grade , y = int_rate , fill = grade)) +
        geom_boxplot() +
        theme_igray() +
        labs(y = 'Interest Rate' , x = 'Grade')
```



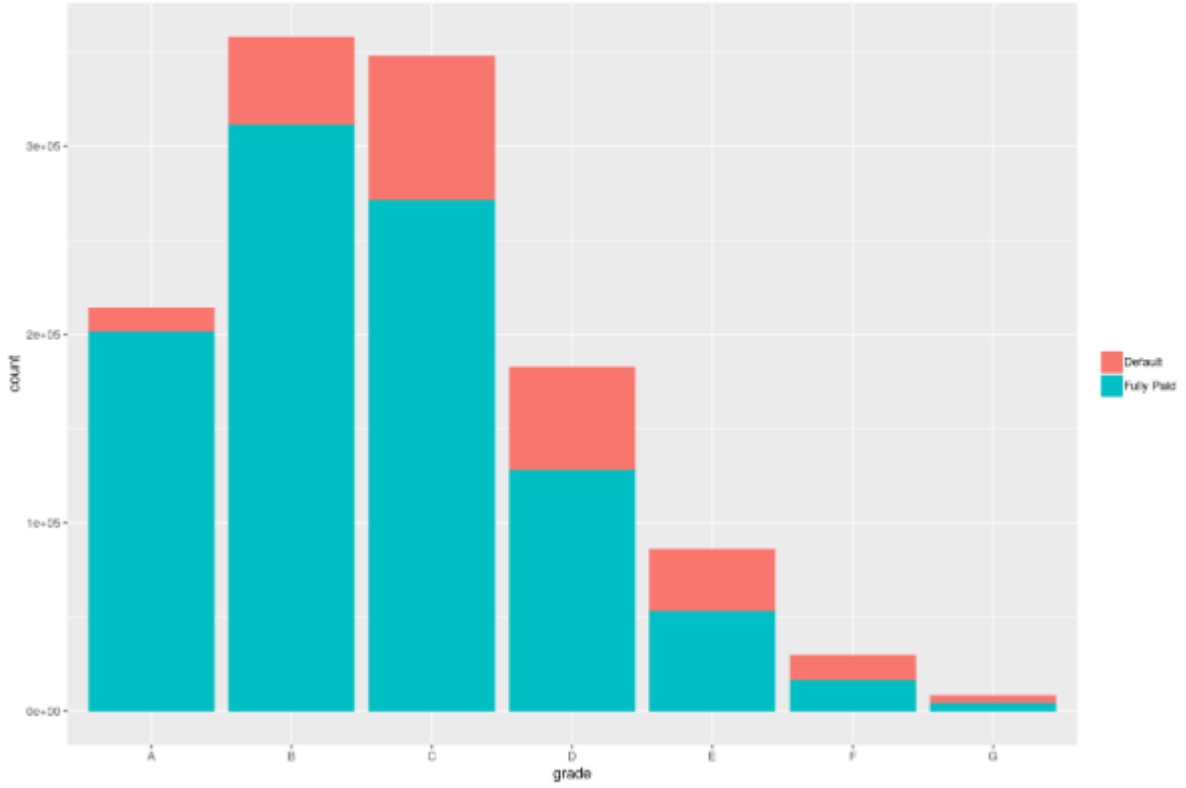
Relation between the loan status and the credit grade

We assume that grade is a great predictor for the volume of non-performing loans. But how many of them did not performed grouped by grade?

```
ggplot(loan2 , aes(x = grade , y = ..count.. , fill = factor(loan_outcome , c(1 , 0) , c('Default' ,
'Fully Paid')))) +
        geom_bar() +
        theme(legend.title = element_blank())
```

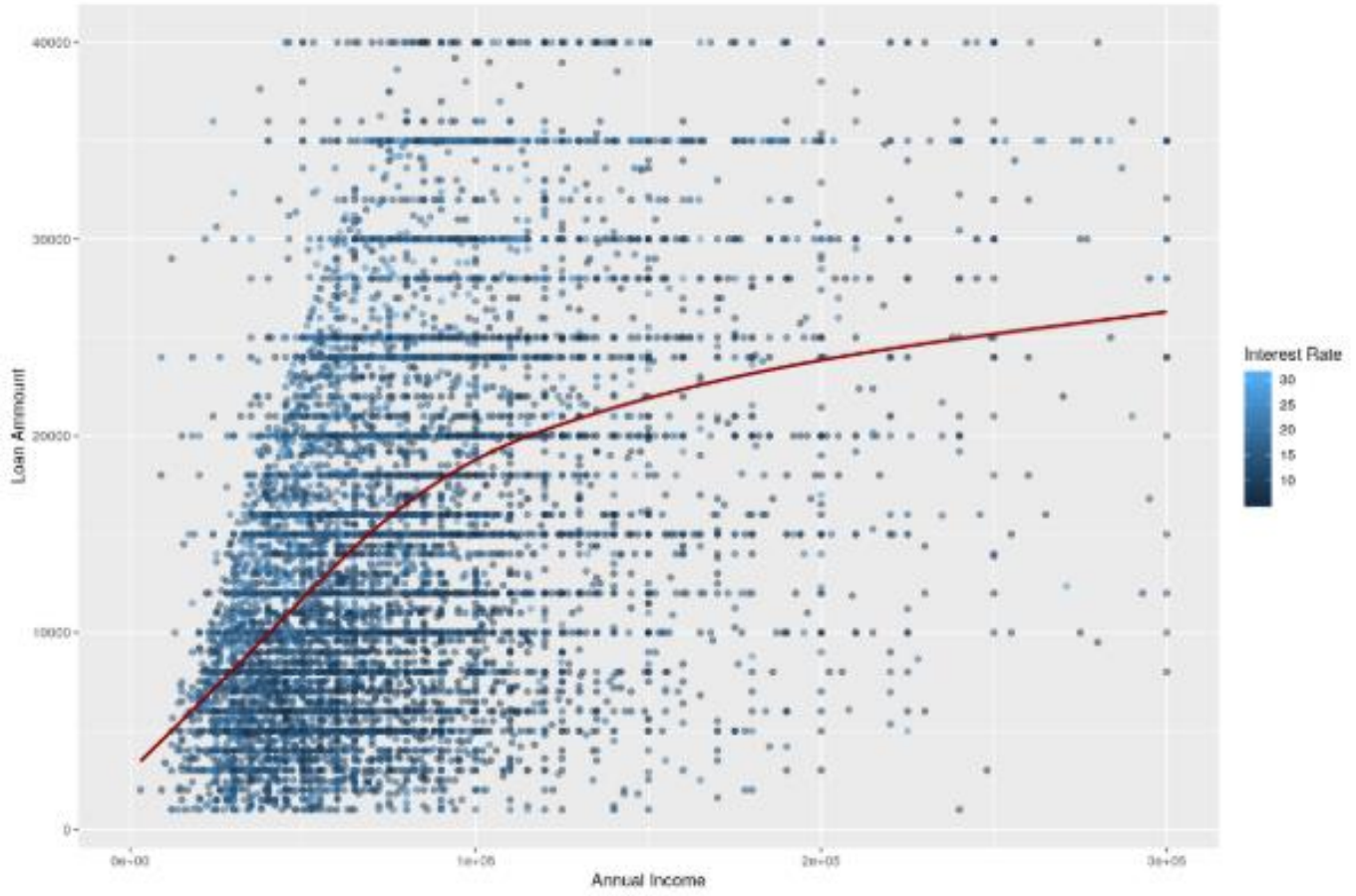
```
table(loan2$grade , factor(loan2$loan_outcome , c(0 , 1) , c('Fully Paid' , 'Default')))
```

```
##
##     Fully Paid Default
##   A     201708   12424
##   B     311584   46655
##   C     271318   76680
##   D     128042   54771
##   E      53267   33064
##   F      16447   13413
##   G       4285    4227
```

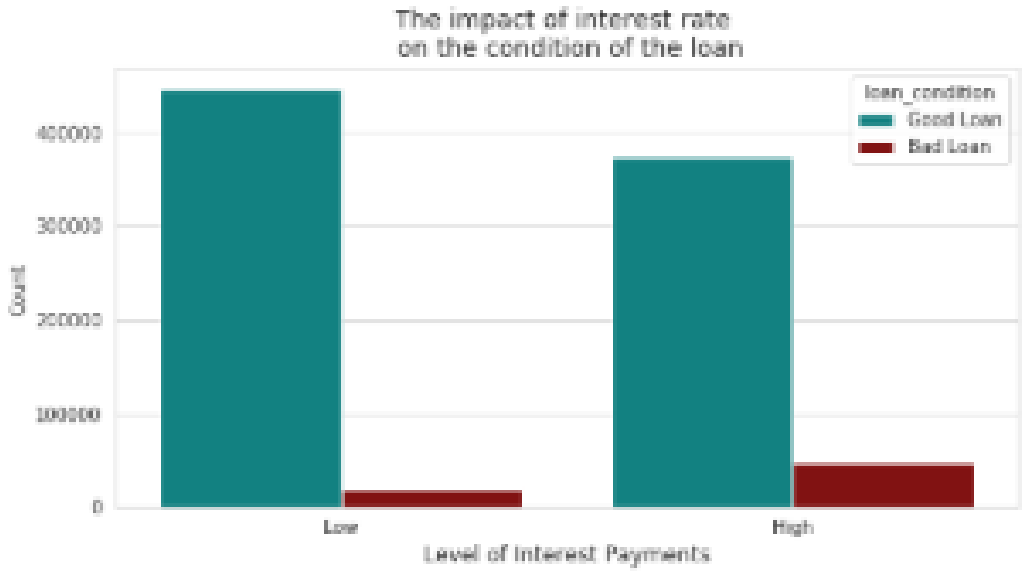


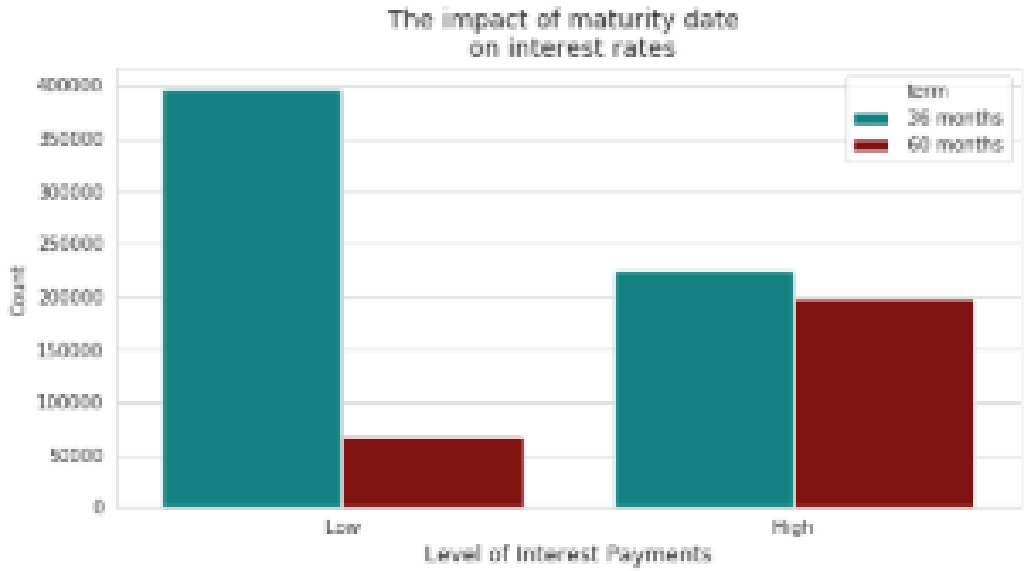
Relationship between Annual Income, Loan amount and the Interest rates

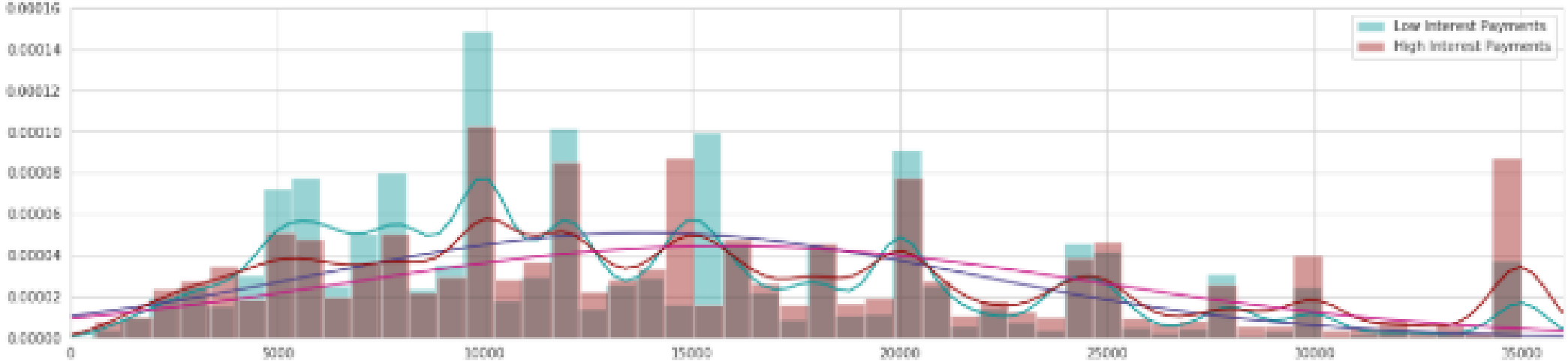
the larger the annual income the larger the demanded amount by the borrower.



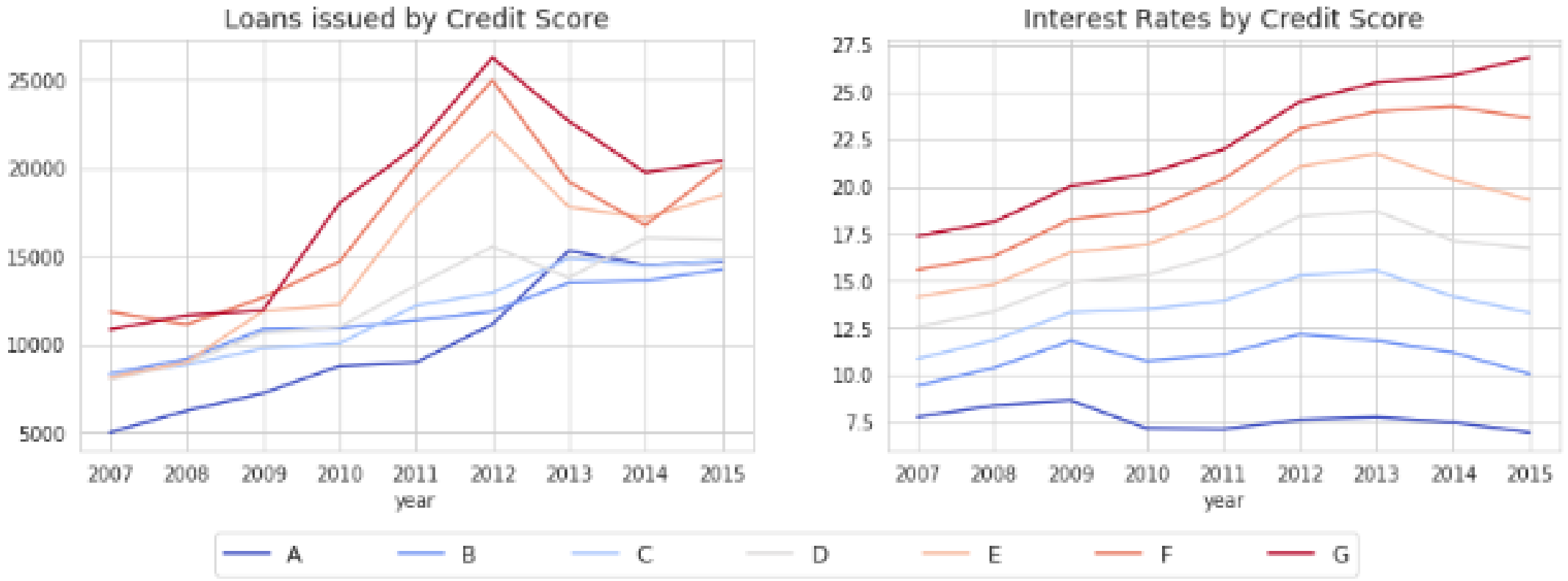
Impact of Interest rates and maturity dates



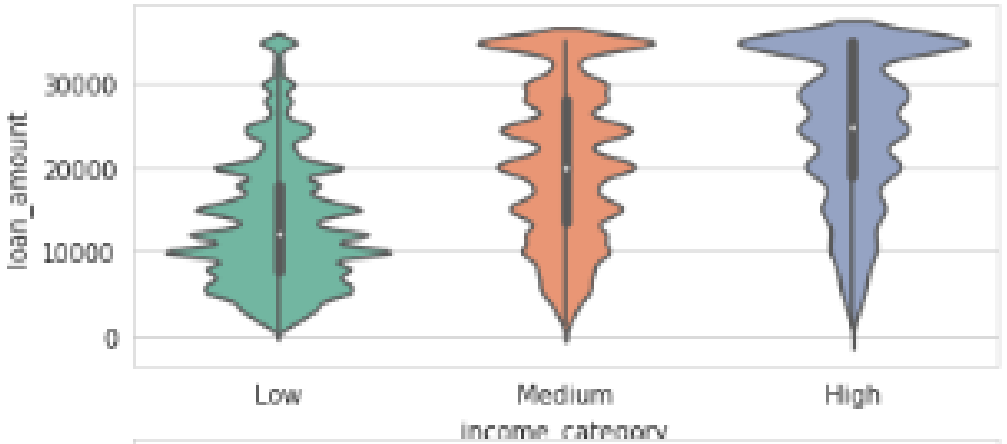


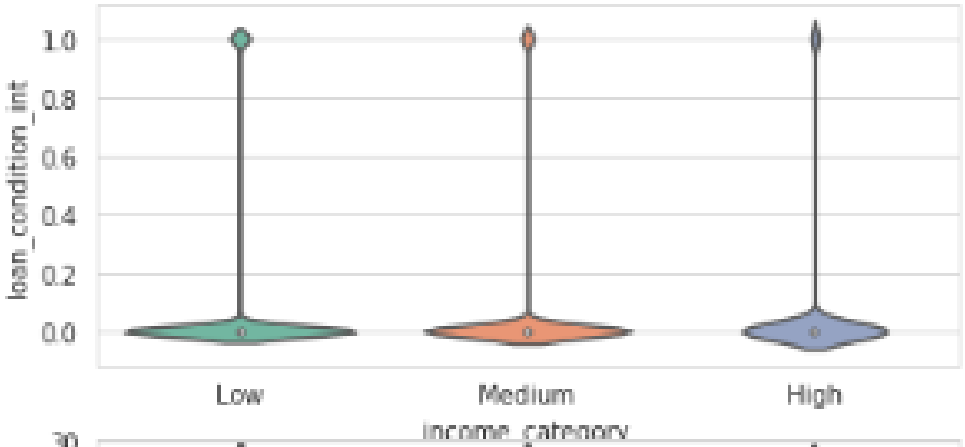


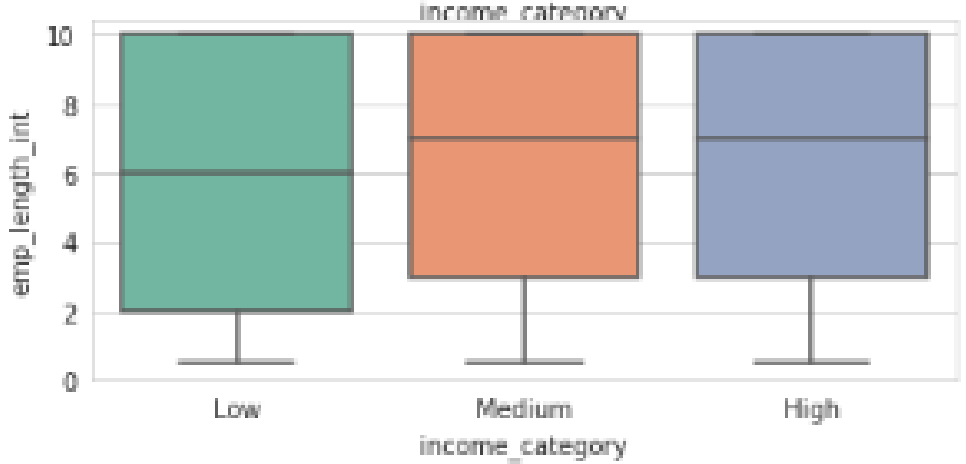
Loans and Interest Rates vs Credit Score

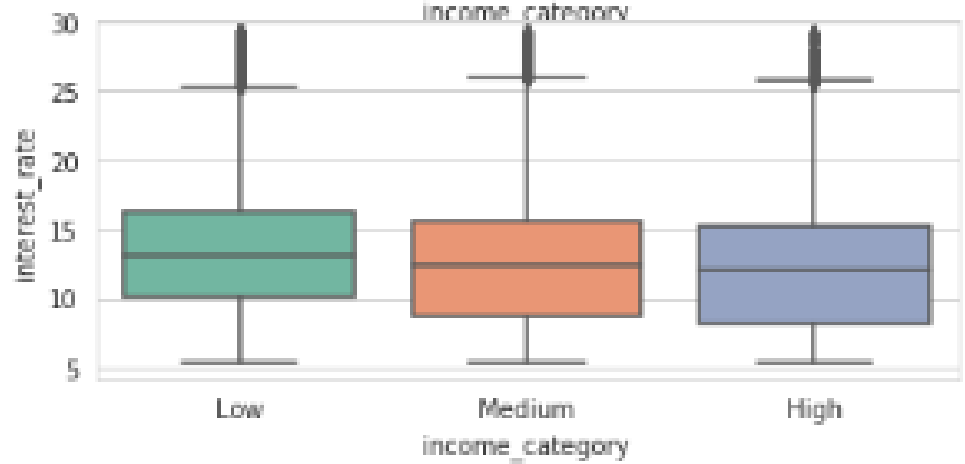


Analysis by Income Category





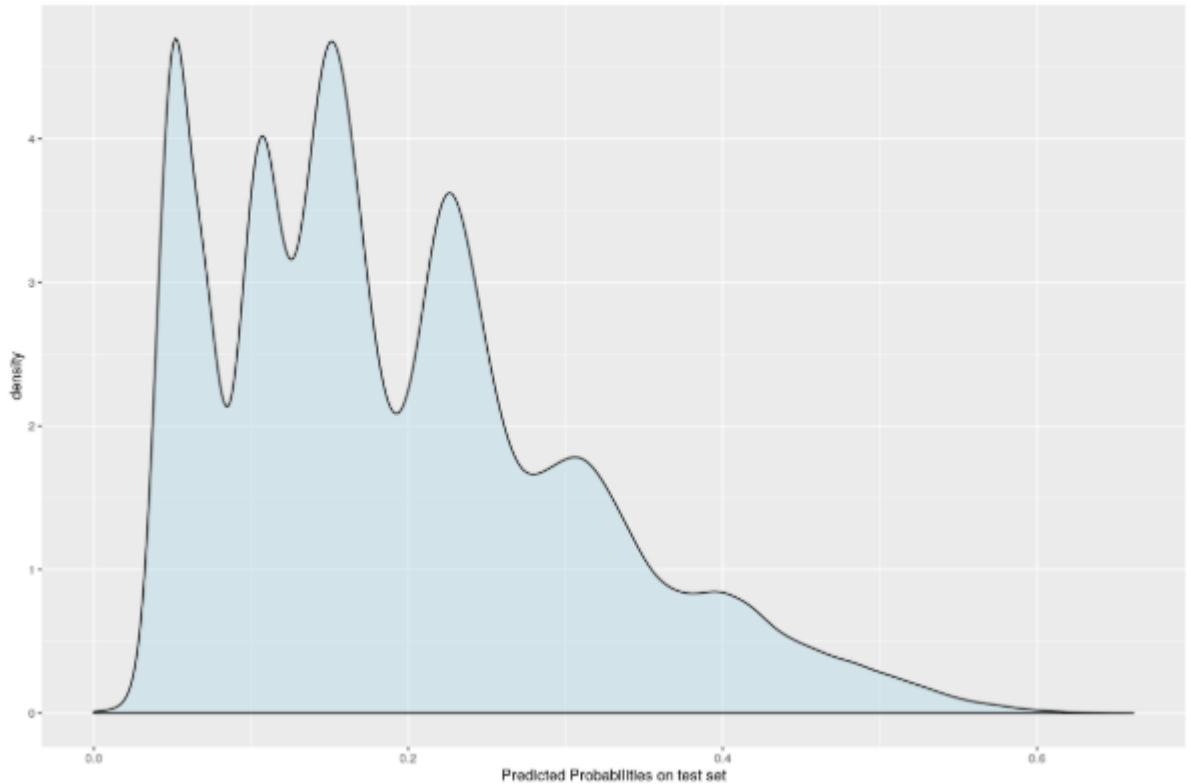




Predicting Loan Default Probability – Visualizing its density

```
# Prediction on test set
preds = predict(glm.model , testset , type = 'response')

# Density of probabilities
ggplot(data.frame(preds) , aes(preds)) +
        geom_density(fill = 'lightblue' , alpha = 0.4) +
        labs(x = 'Predicted Probabilities on test set')
```



Thank you