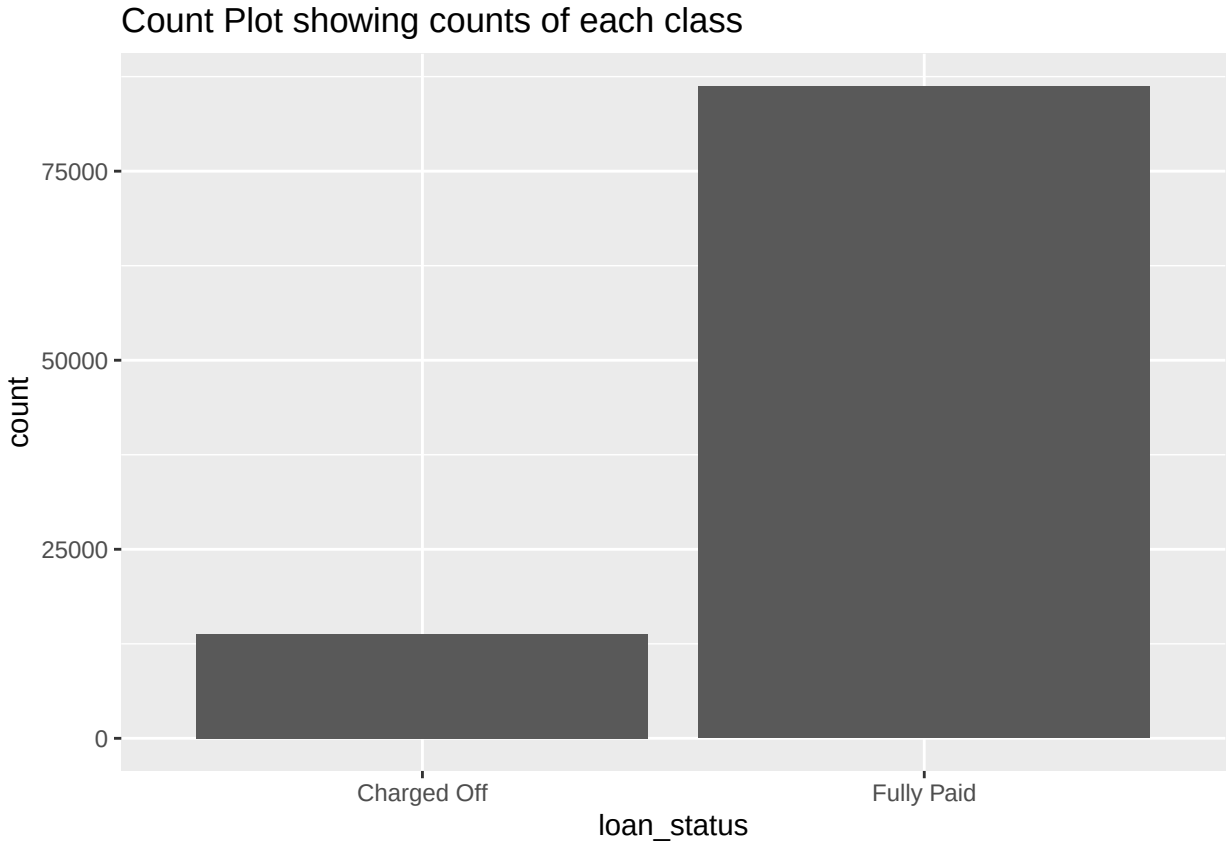
Lending Club Dataset Analysis

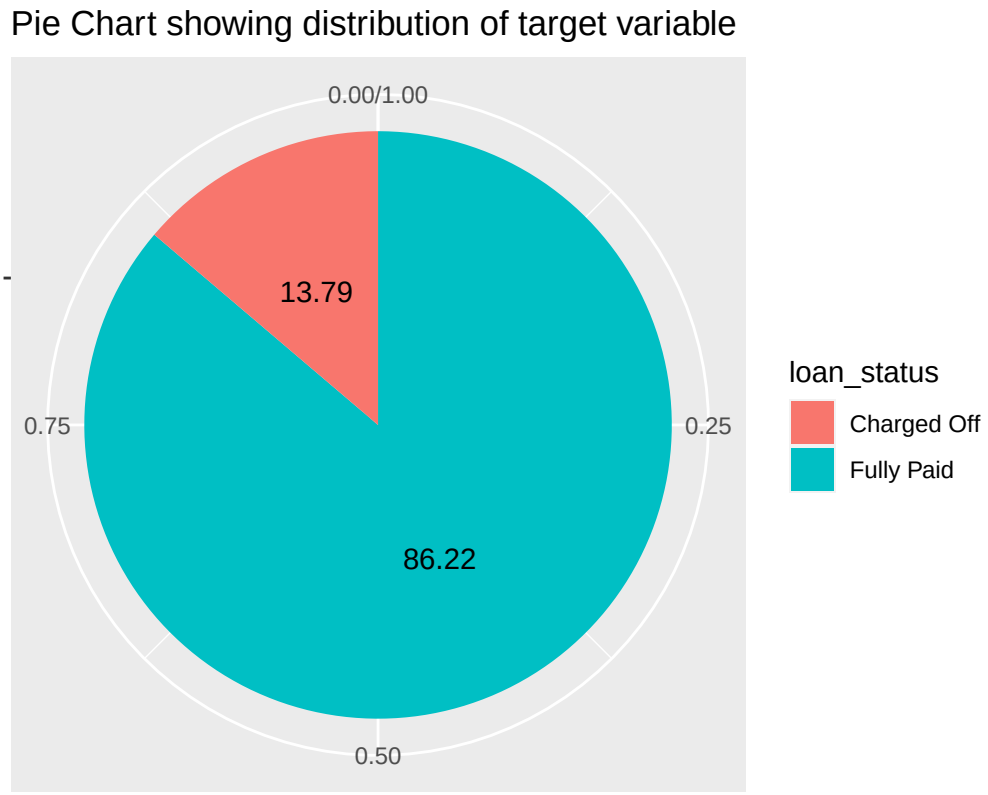
Vamsy Tammineedi

9/20/2021

Data Exploration:

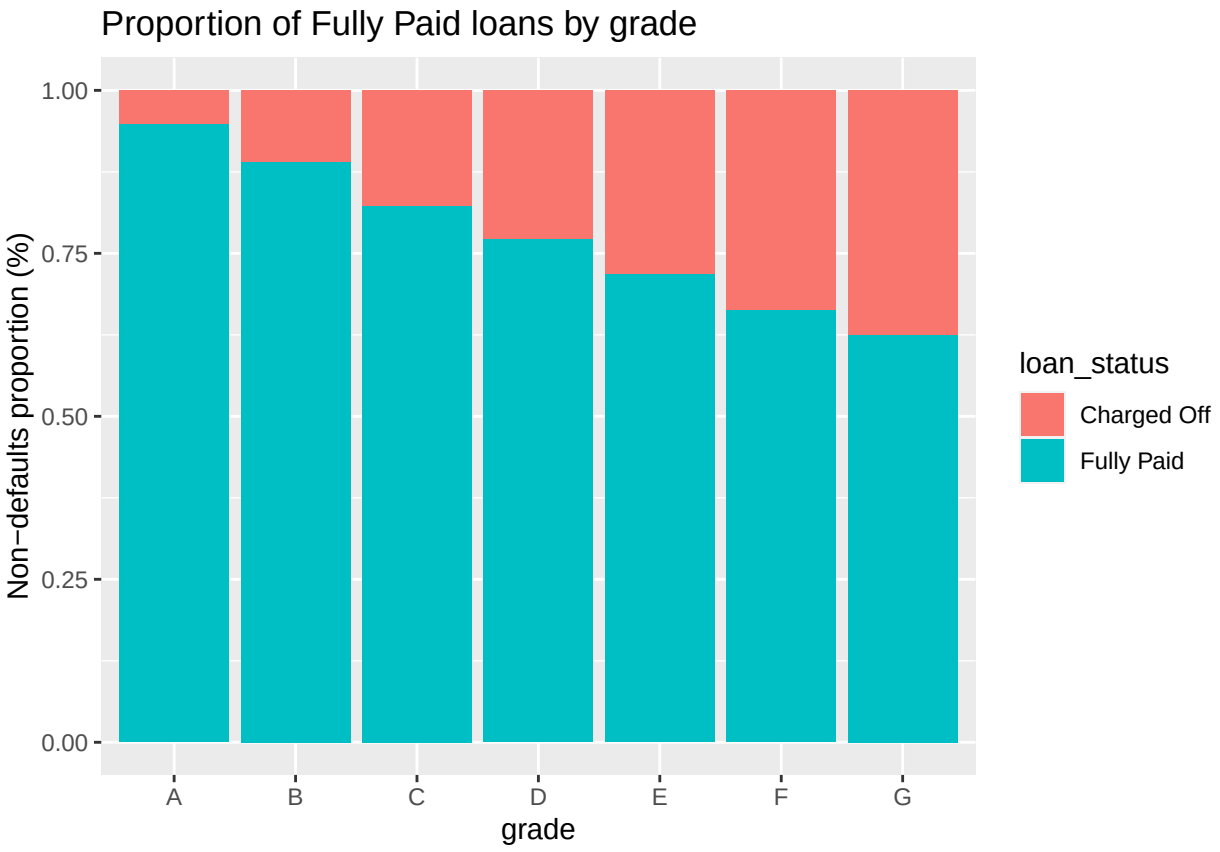
There are a total of 100000 rows and 144 variables in our data and about 13.79% of loans are charged off and 86.22% of loans are fully paid.



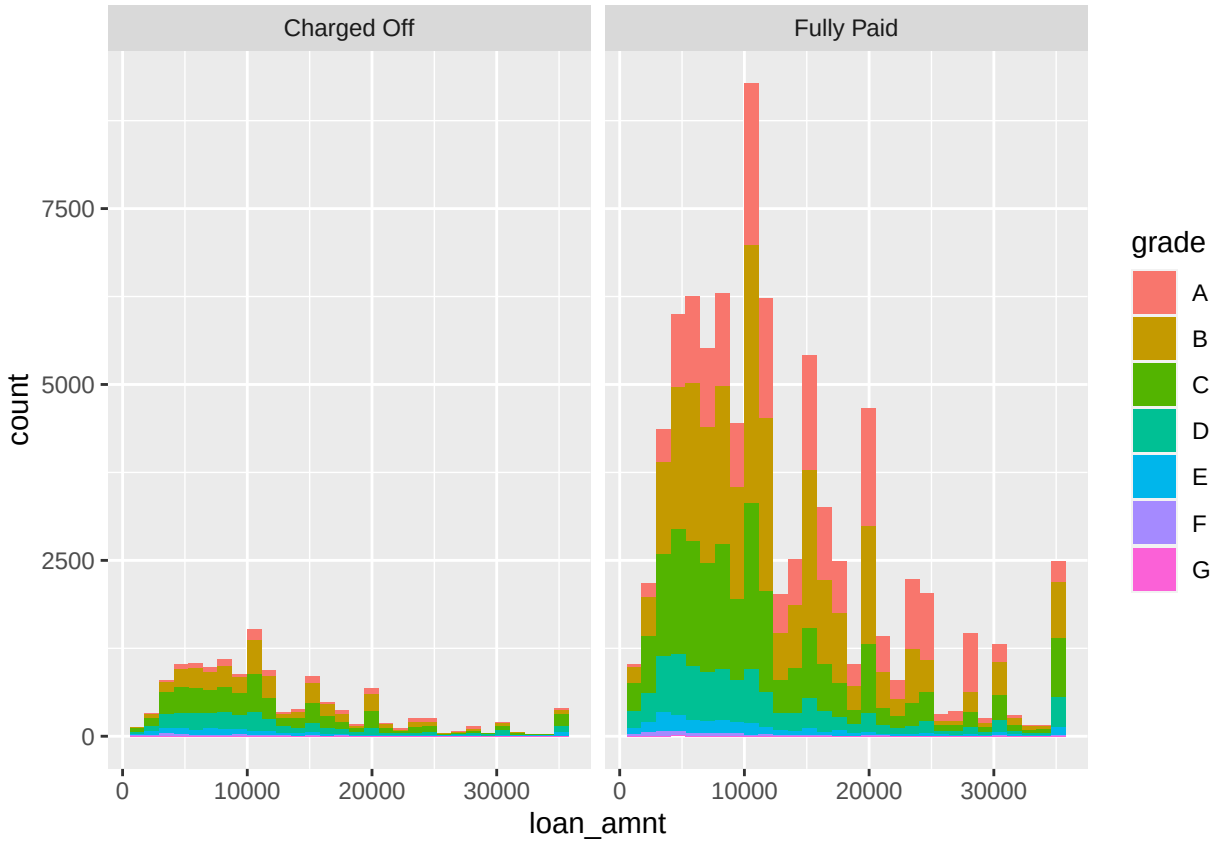


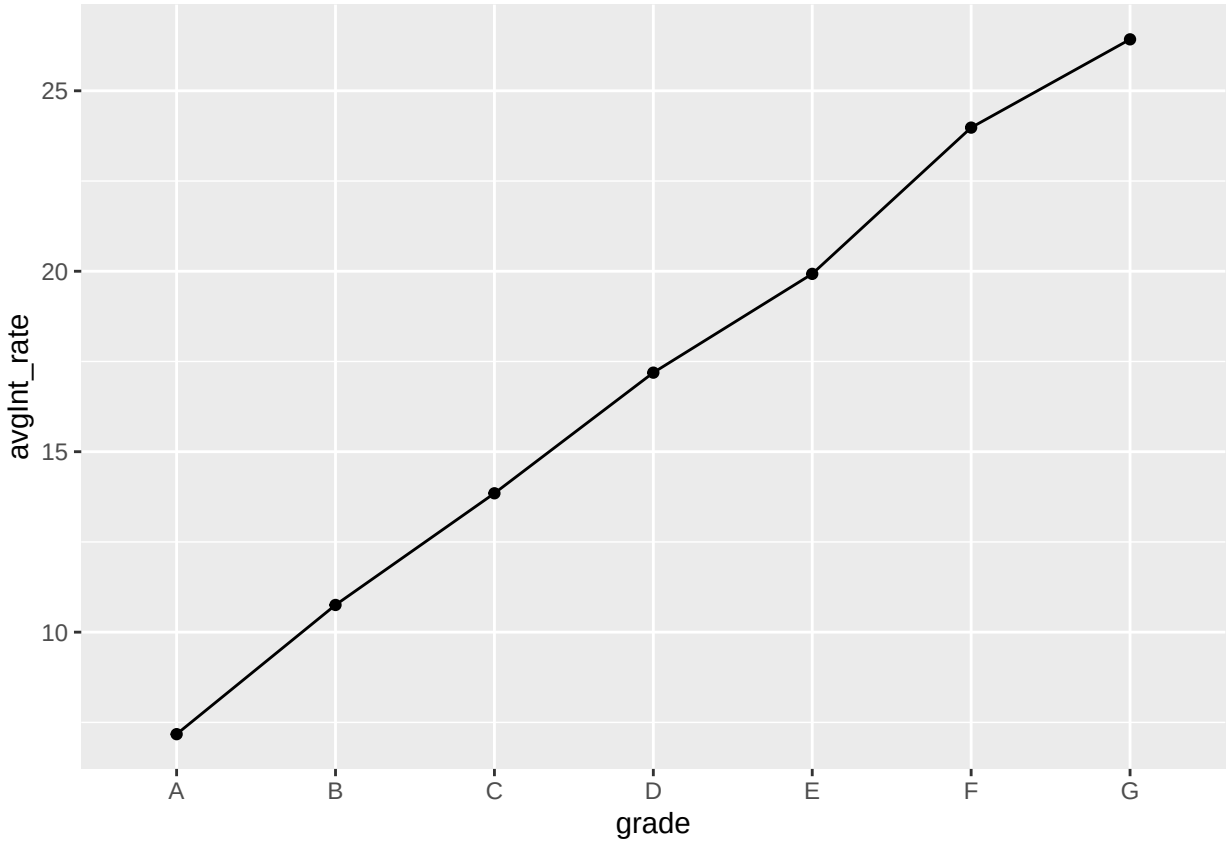
Lending Club categorizes borrowers into seven different loan grades: A through G. Within each loan grade there are five sub-grades from A1 to G5. A borrower is graded depending on many factors like credit report, borrowing amount, debt-income ratio. . . etc. The chance of loan getting approved will be decreasing from A1 thru G5.

```
##
##                     A     B     C     D     E     F     G
##   Charged Off  1187  3723  4738  2858  1010   239    30
##   Fully Paid  21401 30184 21907  9635  2569   469    50
```



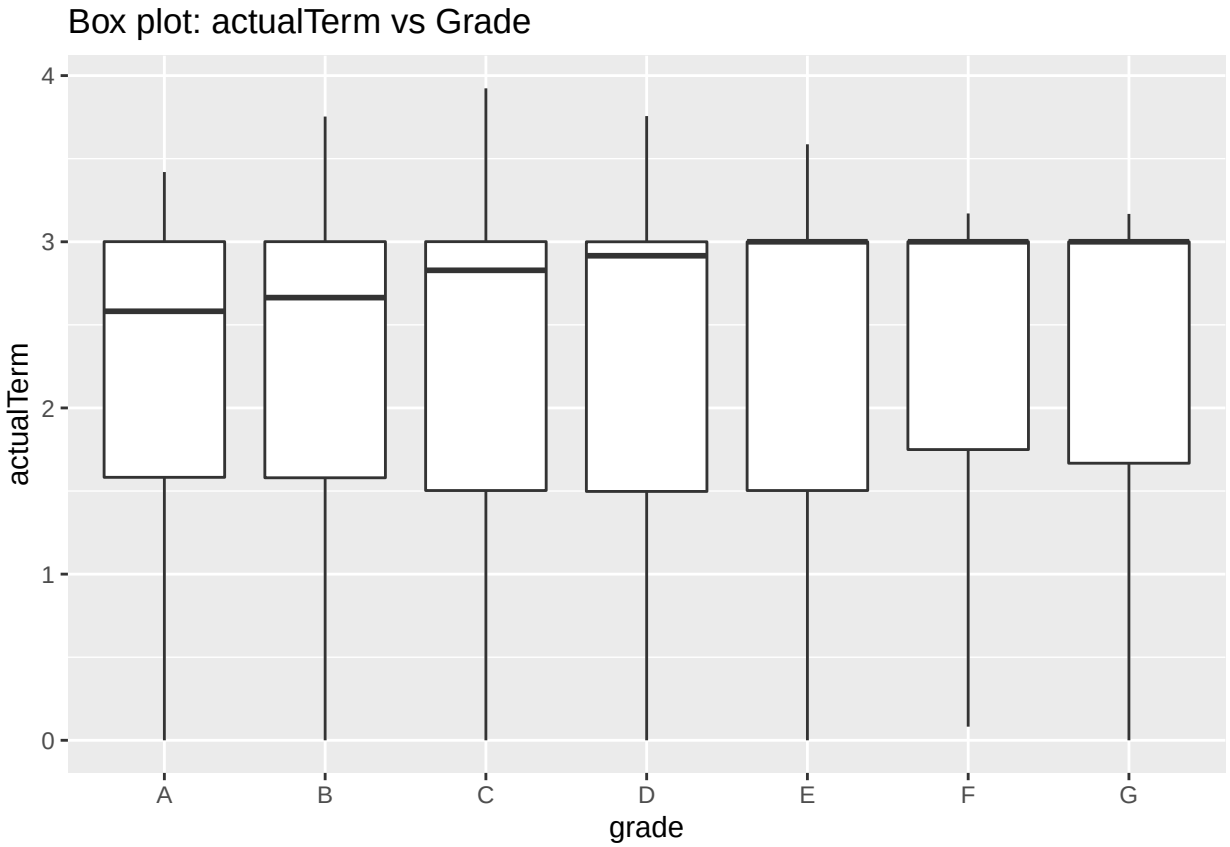
We observe that not many people have grade A, Since credit score is one of the factors effecting the grade and many people didnt have a good credit score back in 2013-2015 it could be one of the reasons for higher people having grades B and C (as per FICO data https://www.fico.com/blogs/average-us-fico-score-ticks-706)


Charged Off
Fully Paid
count
7500
5000
2500
0
0
10000
20000
30000
loan_amnt
grade
A
B
C
D
E
F
G



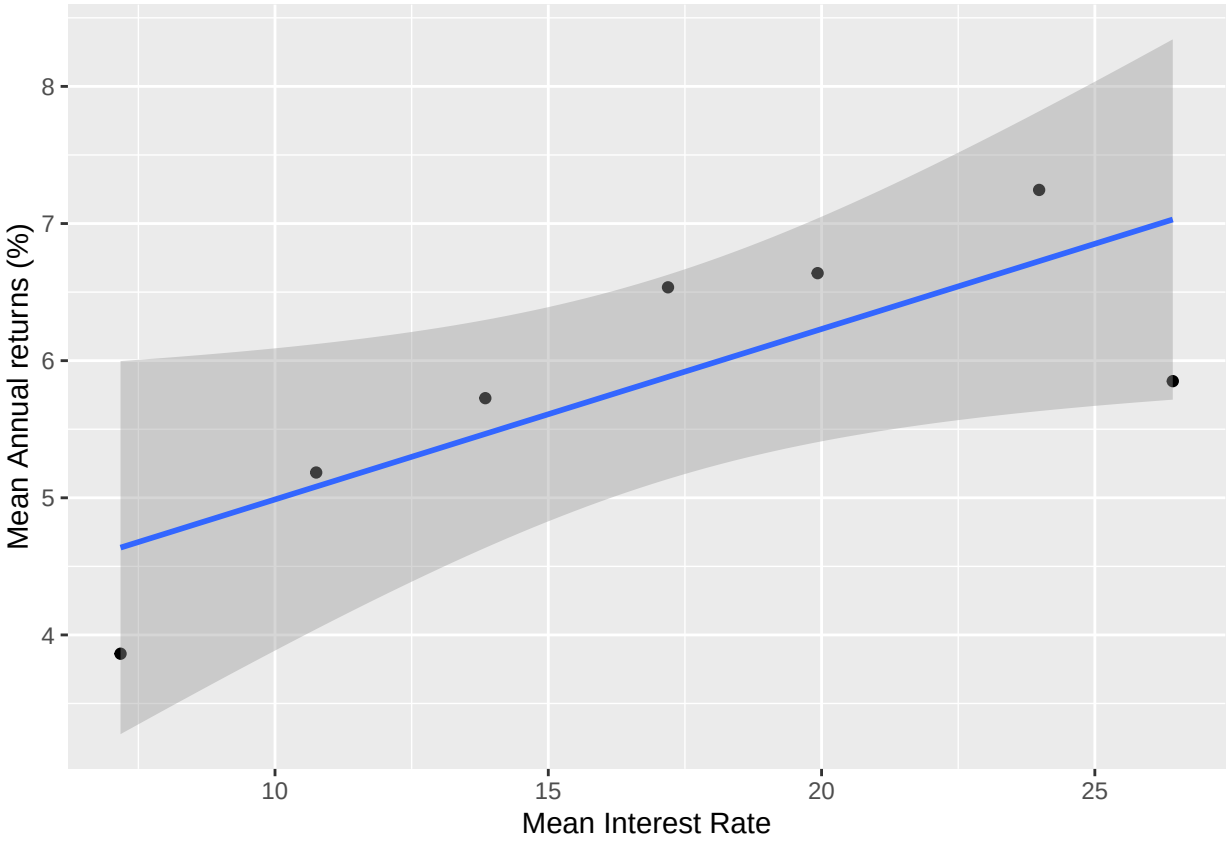
There is a relationship between interest rate and grades which is expected - lower the grade higher the interest rate.

Now we will calculate the annual returns of investors based on the actual term. For a fully paid loan actual term is obtained by subtracting the loan issue date and the last payment date. But in case of a charged off loan last payment date is not available for a few observations in the data. Since all the loans have either a 36- or 60-month term, with fixed interest rates and equal payments. We set 3 years as the actual term for charged off loans.




```
##     loan_amnt total_pymnt int_rate installment actualReturn
## 25       5600     6114.48    12.99      188.66    3.0623810
## 29       3500     3710.88    16.55      124.01    2.0083810
## 204      5000     6033.28    15.61      174.83    6.8885333
## 357      2500     2875.74    19.99       92.90    5.0098667
## 378     10000    10879.32     9.49      320.29    2.9310667
## 398     12000    12267.07    13.67      408.22    0.7418611
```

There are few positive annual returns for charged off loans. This could be because they charged off after paying few installments. For example, a borrower has installment of $188.66 at int_rate of 12.99 with 36 months term. He paid $6114.48 and then charged off on the interest the needs to be paid. So, there are few returns in case of this borrower.



We can see that the annual returns rate % is almost same for grade A,B and slightly decreases as the grade lowers to G. Also the mean return % is having a positive relationship with interest rate. If I were the investor I would probably look for grade B,C loans to invest since they have low default rate and high returns.

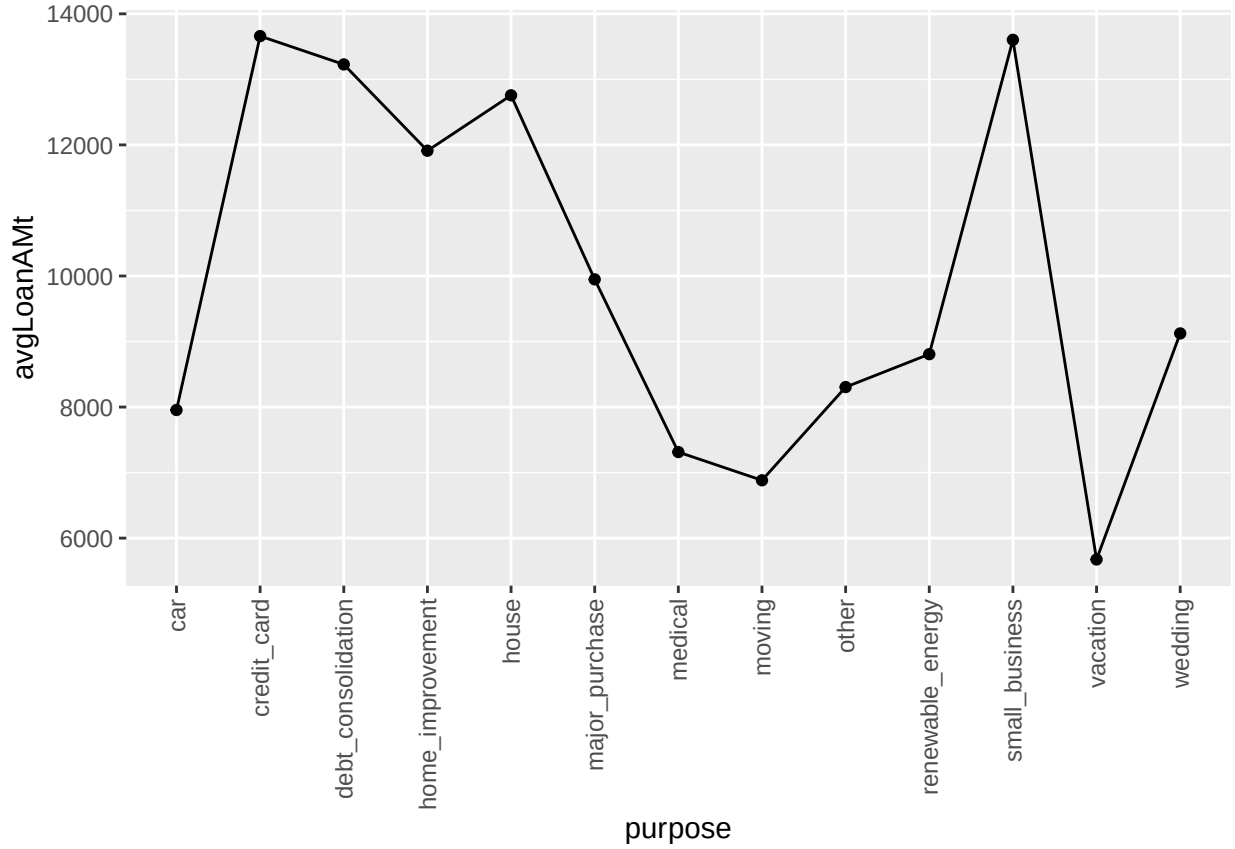
Let's explore few columns

Purpose Column:

```
## # A tibble: 13 x 8
##    purpose        nLoans defaults defaultRate avgInterest stdInterest avgLoanAMt
##    <chr>           <int>    <int>       <dbl>       <dbl>       <dbl>      <dbl>
##  1 car               928      107       0.115        11.5        3.96      7955.
##  2 credit_card     24989     2865       0.115        10.6        3.41     13660.
##  3 debt_consolid~  57622     8319       0.144        12.2        3.71     13228.
##  4 home_improvem~   5654      682       0.121        11.8        3.95     11911.
##  5 house             354       63       0.178        15.3        4.97     12757.
##  6 major_purchase   1823      266       0.146        12.1        4.12      9948.
##  7 medical          1119      172       0.154        14.3        4.09      7313.
##  8 moving            691      144       0.208        16.1        3.93      6882.
##  9 other            5091      838       0.165        14.7        4.08      8305.
## 10 renewable_ene~     58       11       0.190        15.7        3.60      8807.
## 11 small_business    893      203       0.227        16.8        3.93     13603.
## 12 vacation          678      101       0.149        14.5        3.86      5674.
## 13 wedding           100       14       0.14         18.0        3.14      9124.
## # ... with 1 more variable: avgPmnt <dbl>
```

The above table shows the number of loans, defaults, average Interest, average loan amount, and average

payment for all the purposes.




```
##
##                          A     B     C     D     E     F     G
##   car                  253   306   238    92    27     8     4
##   credit_card         8349  9809  5008  1518   266    37     2
##   debt_consolidation 11573 19745 16497  7534  1954   292    27
##   home_improvement    1457  1777  1496   673   215    33     3
##   house                 37    74    83    74    48    27    11
##   major_purchase       441   553   479   252    70    26     2
##   medical               84   270   382   251    97    34     1
##   moving                10    96   207   234   108    32     4
##   other                324  1036  1702  1321   551   139    18
##   renewable_energy       3     5    22    18     8     2     0
##   small_business        15   100   249   300   159    62     8
##   vacation              42   127   257   180    59    13     0
##   wedding                0     9    25    46    17     3     0
```

We can see that the average loan amount is high for a credit card, debt consolidations, house, small business loans and low for vacation, moving and medical loans and also the number of loans for debt consolidation irrespective of loan grade and as stated earlier overall number of loans are high for grades B and C.

```
## [1] 0
```

```
##  chr [1:100000] "9 years" "6 years" "3 years" "n/a" "1 year" "10+ years" ...
```

Employement Length Column: The employment length column has missing information but running is.na() function doesn't show any missing values because the missing information is entered as "n/a". Lets

also convert the employement length feature into factor variable

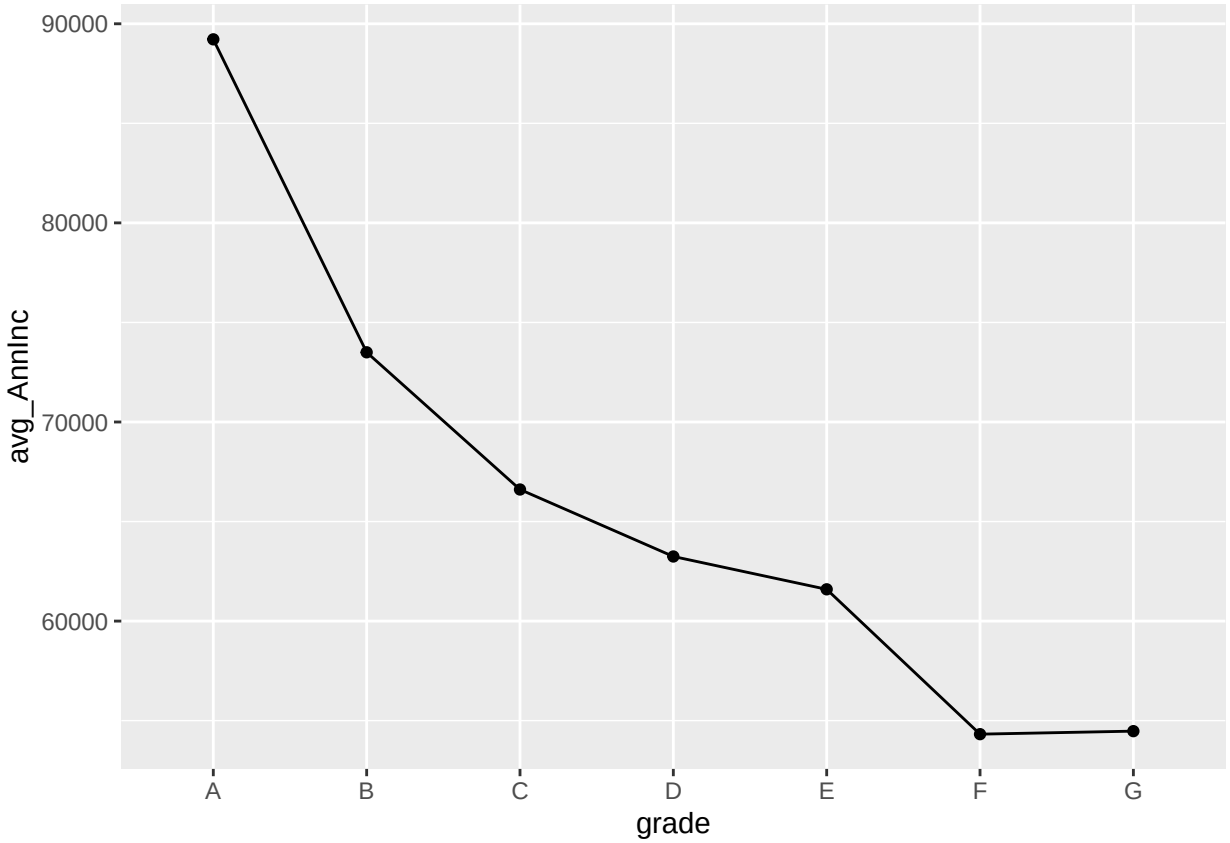
```
## # A tibble: 12 x 7
##    emp_length nLoans defaults defaultRate avg_loan_amnt avg_act_ret avg_act_term
##    <fct>       <int>    <int>       <dbl>         <dbl>       <dbl>        <dbl>
##  1 n/a          6148     1296       0.211        10152.        3.95         2.43
##  2 < 1 year     8104     1204       0.149        12171.        5.01         2.23
##  3 1 year       6649      960       0.144        12137.        5.10         2.25
##  4 2 years      8987     1206       0.134        12252.        5.29         2.22
##  5 3 years      8046     1088       0.135        12433.        5.31         2.25
##  6 4 years      5892      775       0.132        12556.        5.37         2.24
##  7 5 years      6046      841       0.139        12658.        5.28         2.23
##  8 6 years      4712      632       0.134        12589.        5.48         2.23
##  9 7 years      5124      712       0.139        12563.        5.39         2.23
## 10 8 years      4990      698       0.140        13029.        5.17         2.24
## 11 9 years      3908      522       0.134        13077.        5.19         2.24
## 12 10+ years   31394     3851       0.123        13741.        5.56         2.24
```

```
##
##             Charged Off Fully Paid
##   n/a              1296       4852
##   < 1 year         1204       6900
##   1 year            960       5689
##   2 years          1206       7781
##   3 years          1088       6958
##   4 years           775       5117
##   5 years           841       5205
##   6 years           632       4080
##   7 years           712       4412
##   8 years           698       4292
##   9 years           522       3386
##   10+ years        3851      27543
```

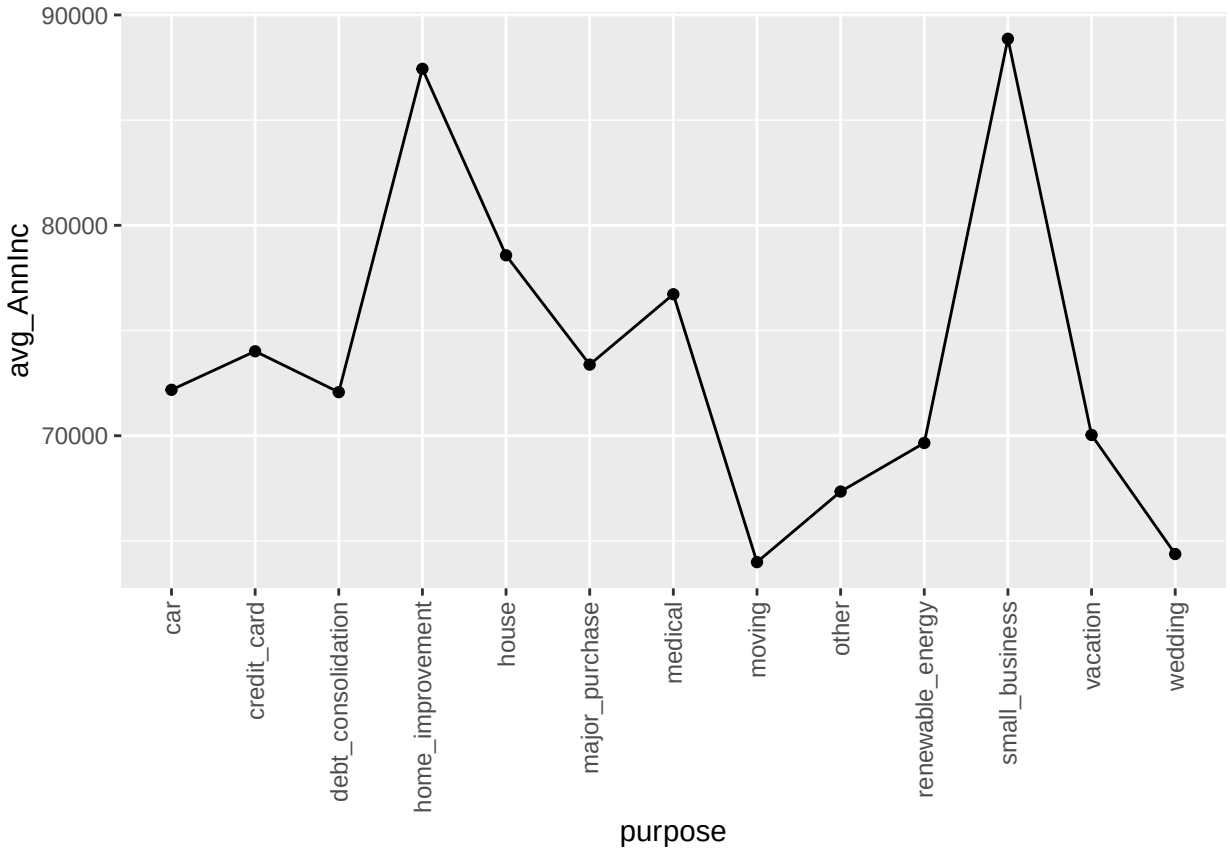
The average returns, average loan amount are high in case of cusotmer with more than 10 years of employement length. Also default rate seems to be decreasing with increase in employement length

Annual Income Column:

```
## # A tibble: 2 x 2
##   loan_status avg_AnnInc
##   <chr>            <dbl>
## 1 Charged Off     64678.
## 2 Fully Paid      74744.
```

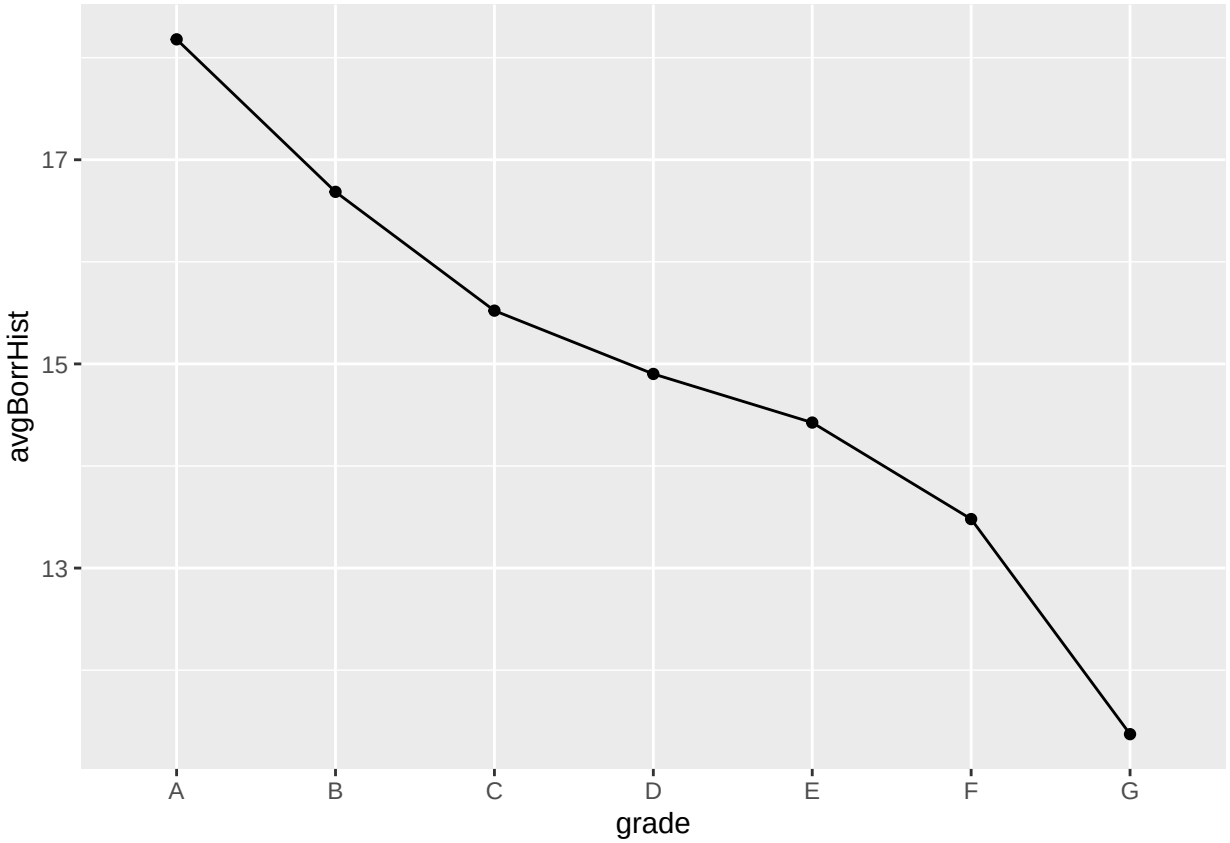


As expected, the average annual income of fully paid loans is higher than that of defaulted loans. Also the average income level keeps increasing as we move from higher grades to lower which can be one of the reasons for high default rates for lower grades.



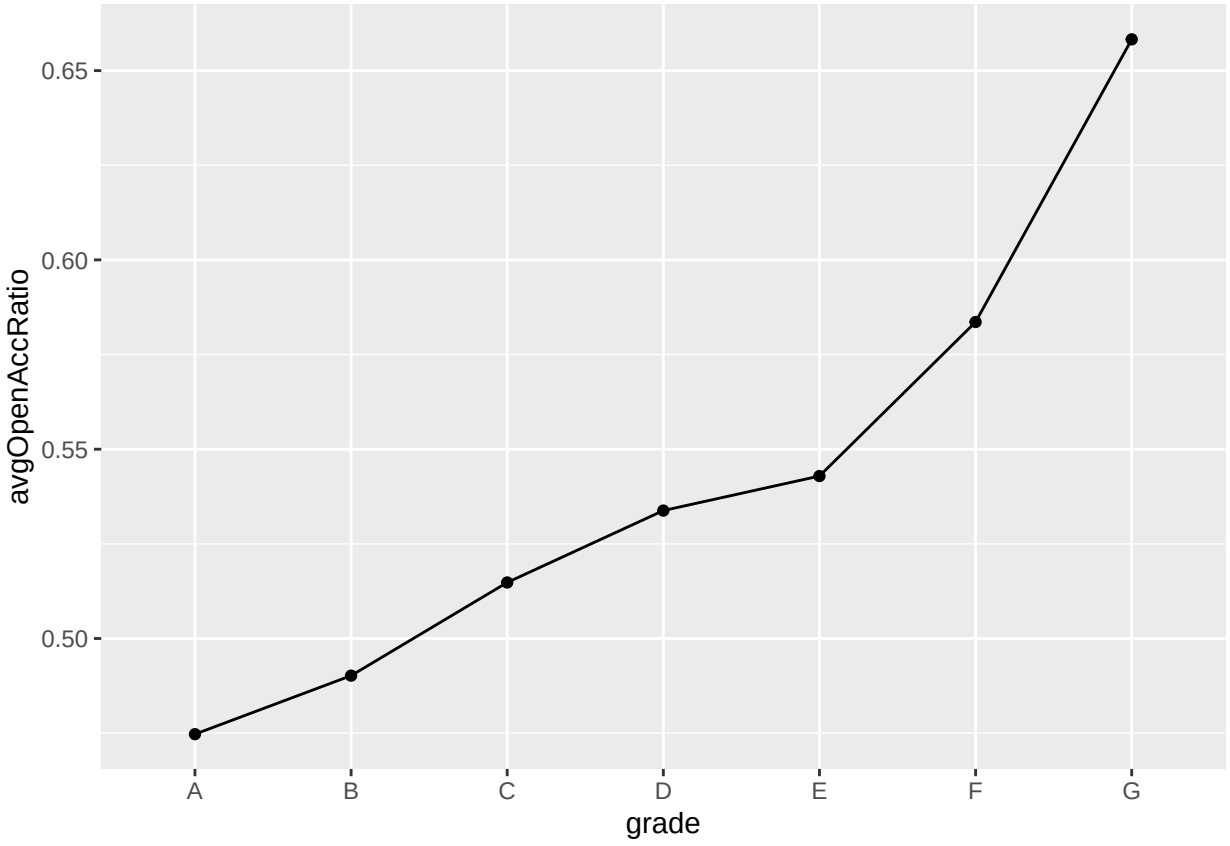
Also, the average annual income is high for cusotmer applying loan for home improvements, small businesses

Let's look at some more columns and see if we can derive some new variables out of them. We have a few columns which gives us details about a peron's backgroud info like the date the borrower's earliest reported credit line was opened, the number of open credit lines and the total number of credit lines in the borrower's credit file, the number of satisfactory bankcard accounts, and the total number of bankcard accounts.



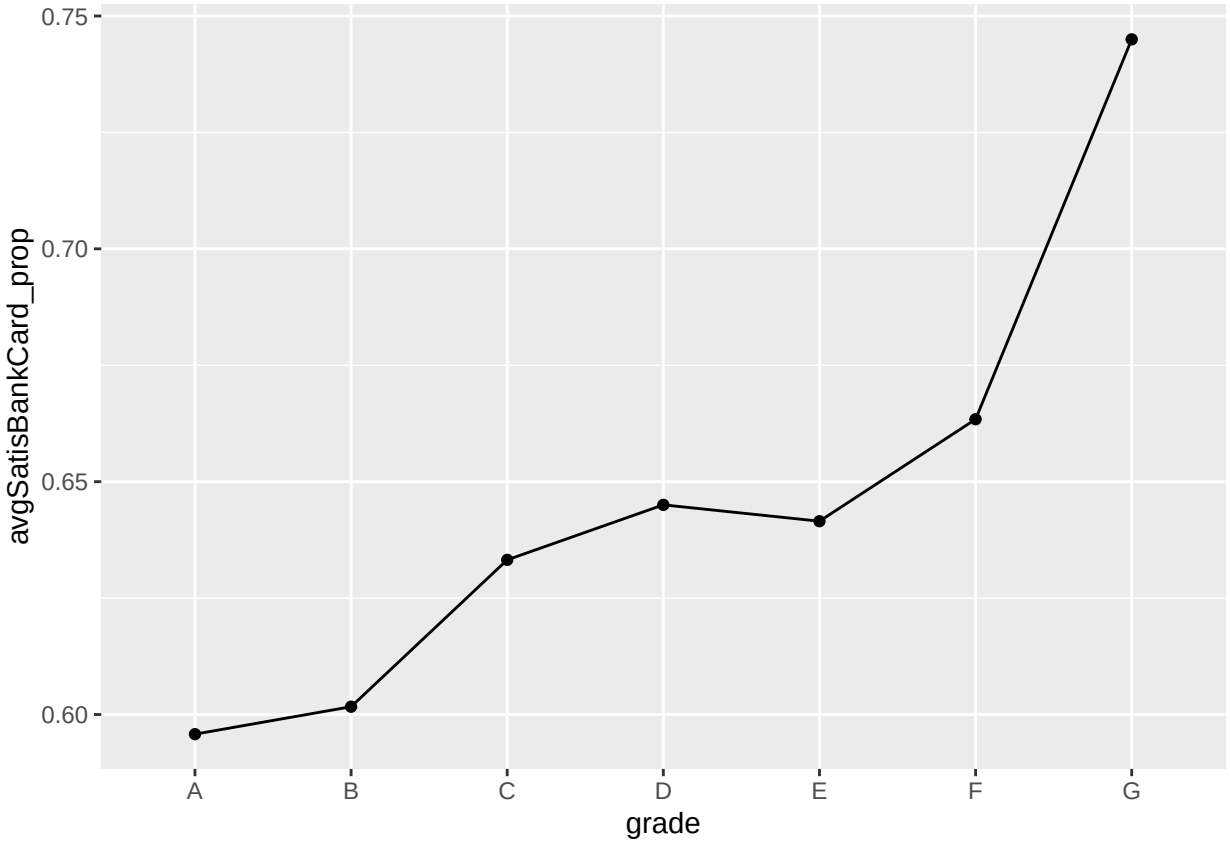

```
## # A tibble: 2 x 2
##   loan_status avgBorrHist
##   <chr>             <dbl>
## 1 Charged Off        15.6
## 2 Fully Paid         16.5
```

Borrower's History is calculated by subtracting the earliest credit line date and the loan issue date.We can see that for grade A the borrower's history is high and it keeps decreasing as we move to lower grades.




```
## # A tibble: 2 x 2
##   loan_status avgOpenAccRatio
##   <chr>                 <dbl>
## 1 Charged Off           0.522
## 2 Fully Paid            0.498
```

Open account ratio is calculated by taking the ratio of the number of open credit lines to the total number of credit lines in the borrower's credit file. We can see that as the grade moves from A to G the open account ratio keeps increasing.

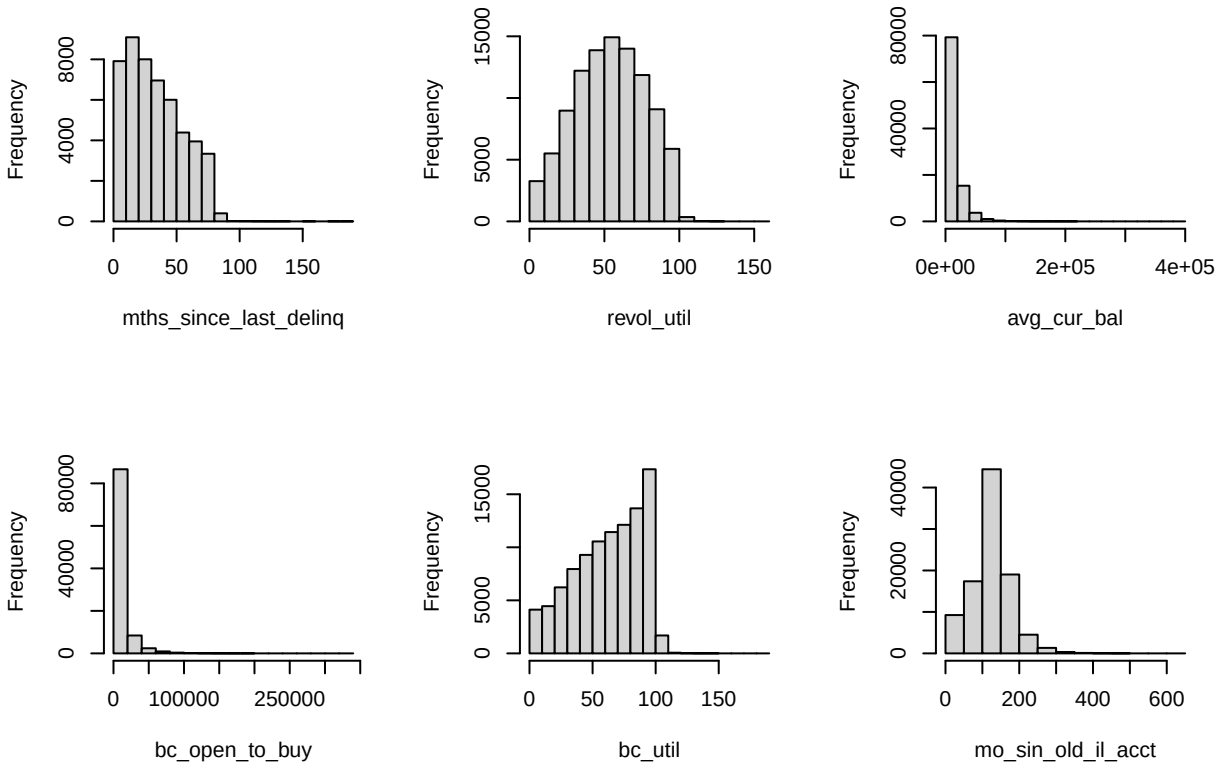


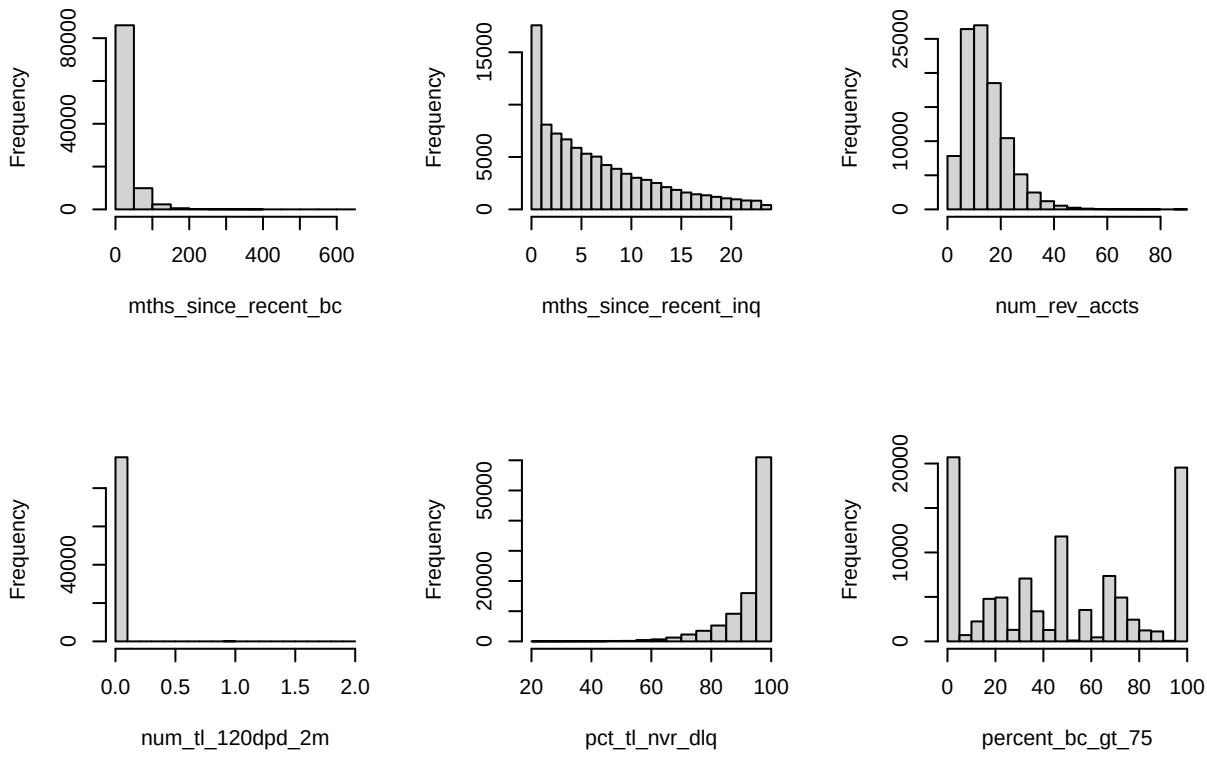

```
## # A tibble: 2 x 2
##   loan_status avgSatisBankCard_prop
##   <chr>                       <dbl>
## 1 Charged Off                 0.635
## 2 Fully Paid                  0.613
```

Satisfactory Bankcard accounts ratio is calculated by dividing the number of satisfactory bankcard accounts by the total number of bankcard accounts. We can see that the Satisfactory Bankcard accounts ratio is increasing as the grade moves from A to G

Let's look at the distributions of missing value columns. Most of the columns are right skewed distributions. We impute all those columns with median of its respective column.

```
##  [1] "mths_since_last_delinq" "revol_util"             "avg_cur_bal"
##  [4] "bc_open_to_buy"         "bc_util"                "mo_sin_old_il_acct"
##  [7] "mths_since_recent_bc"   "mths_since_recent_inq"  "num_rev_accts"
## [10] "num_tl_120dpd_2m"       "pct_tl_nvr_dlq"         "percent_bc_gt_75"
```


Frequency
0
4000
8000
0
50
100
150
mths_since_last_delinq
Frequency
0
5000
15000
0
50
100
150
revol_util
Frequency
0
40000
80000
0e+00
2e+05
4e+05
avg_cur_bal
Frequency
0
40000
80000
0
100000
250000
bc_open_to_buy
Frequency
0
5000
15000
0
50
100
150
bc_util
Frequency
0
20000
40000
0
200
400
600
mo_sin_old_il_acct



There are many columns with all data points as NAs so first let's drop such columns. Later remove columns which were having more than 60% missing values.

Let's also remove variables that can be a potential for data leakage. Though these variables were present at the time of creating and training the data model, when a new borrower enrolls into the lending club these values will not be present. Hence, creating a model based on these variables will result in inaccurate predictions.

The variables that have been excluded from the model are funded_amnt,funded_amnt_inv, term, installment, grade, emp_title, pymnt_plan, title, zip_code, addr_state, out_prncp, out_prncp_inv, total_pymnt_inv, total_rec_prncp, total_rec_int,total_rec_late_fee,recoveries, collection_recovery_fee, last_credit_pull_d, policy_code, disbursement_method, debt_settlement_flag, hardship_flag,chargeoff_within_12_mths, collections_12_mths_ex_med, application_type,last_pymnt_d,last_pymnt_amnt,total_pymnt,issue_d

Let's measure the AUC score of each variable on loan status using auc() function. Below is the final table with sorted scores in descending order. Below is the rankings table. (actualReturn, total_pymnt, actualTerm columns will be omitted later when building the model)

```
## # A tibble: 64 x 2
##    names                    x
##    <chr>                <dbl>
##  1 actualReturn         0.986
##  2 total_pymnt          0.756
##  3 sub_grade            0.666
##  4 actualTerm           0.664
##  5 int_rate             0.658
##  6 acc_open_past_24mths 0.583
##  7 annual_inc           0.577
```

```
##  8 bc_open_to_buy       0.574
##  9 tot_hi_cred_lim      0.574
## 10 total_bc_limit       0.573
## # ... with 54 more rows
```

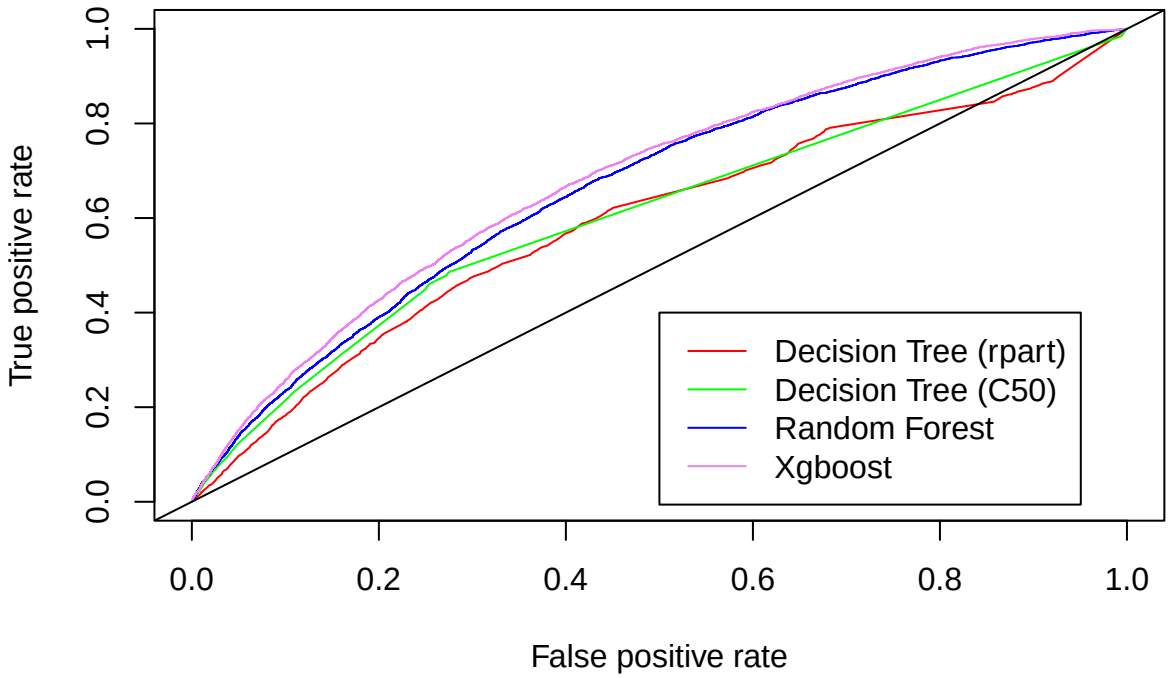
Let's build few models

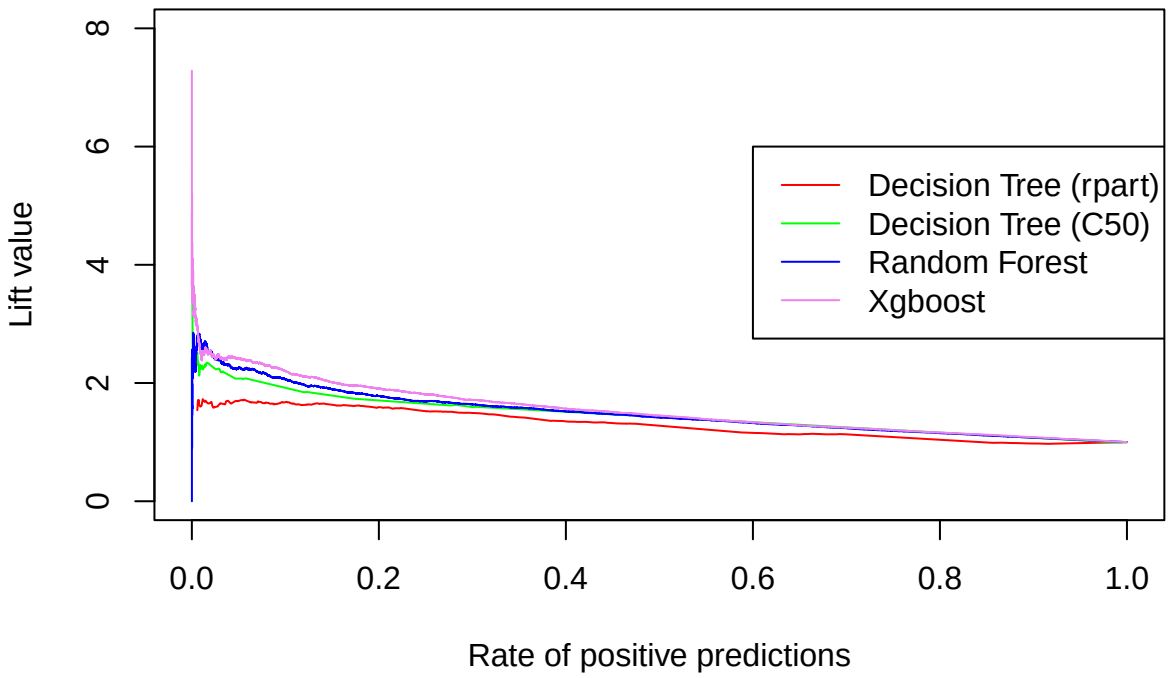
We have a total of 100K observations in our data set. Let's use 70% of the data as a training set and remaining as a test set.

Multiple classification models were built - Decision Tree using both rpart and C50 libraries, Random Forests, Gradient Boosting and are compared. Sensitivity score is used as a evaluation metric to compare models, as shown below

```
##                  Models Accuracy Sensitivity  AUC
## 1               XGBoost     0.48        0.81 0.68
## 2 Decision Tree (rpart)     0.82        0.14 0.59
## 3   Decision Tree (C50)     0.86         0.2 0.61
## 4         Random Forest     0.86         0.4 0.66
```

Let's look at the ROC curves and lift charts for above classification models





Now lets try building regression models for predicting actual returns as well and later combine this model with that of classification model to see if we can improve accuracy

Annualized Returns is calculated by summing all loan payments received net of principal repayment, credit losses, and servicing costs. LendingClub charges an investor service fee of 1% of the number of borrower payments received by the payment due date or during applicable grace periods.

A random forest, a linear model and an xgboost model are developed to predict the best returns.Below table shows the RMSE scores for the regression models developed.

```
##     Data Set XGBoost Linear Model Random Forest
## 1      Train    3.69         8.24          7.97
## 2 Validation    8.23         8.17          8.16
```

Let's combine best models that we built previously - a regression model that predicts actualreturns, classification model to predict default rate

If we had used just the classification model, we would have had more emphasis on capital preservation but have a low returns. However, by incorporating the probability scores from the classification model into our regression model for predicting return, we are able to invest in the loans most likely to be paid with the highest predicted return.

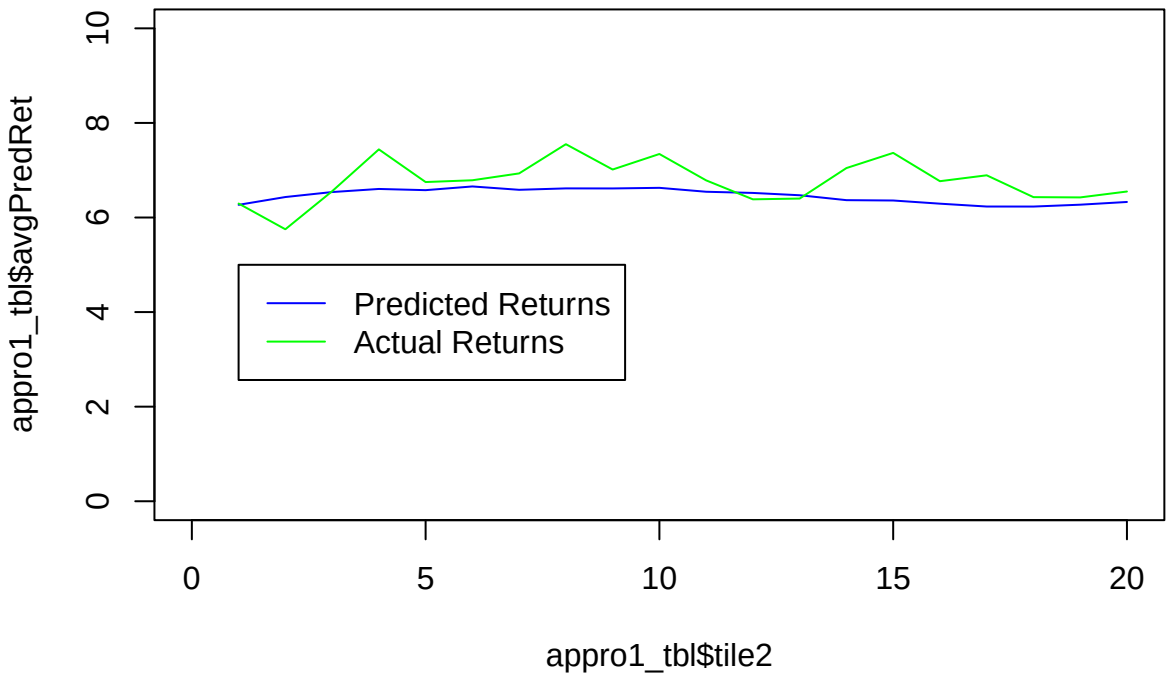
Approach 1 -> Consider regression model based loans with high predicted returns, and sort these based on high classification model scores and then select the top (4%) loans

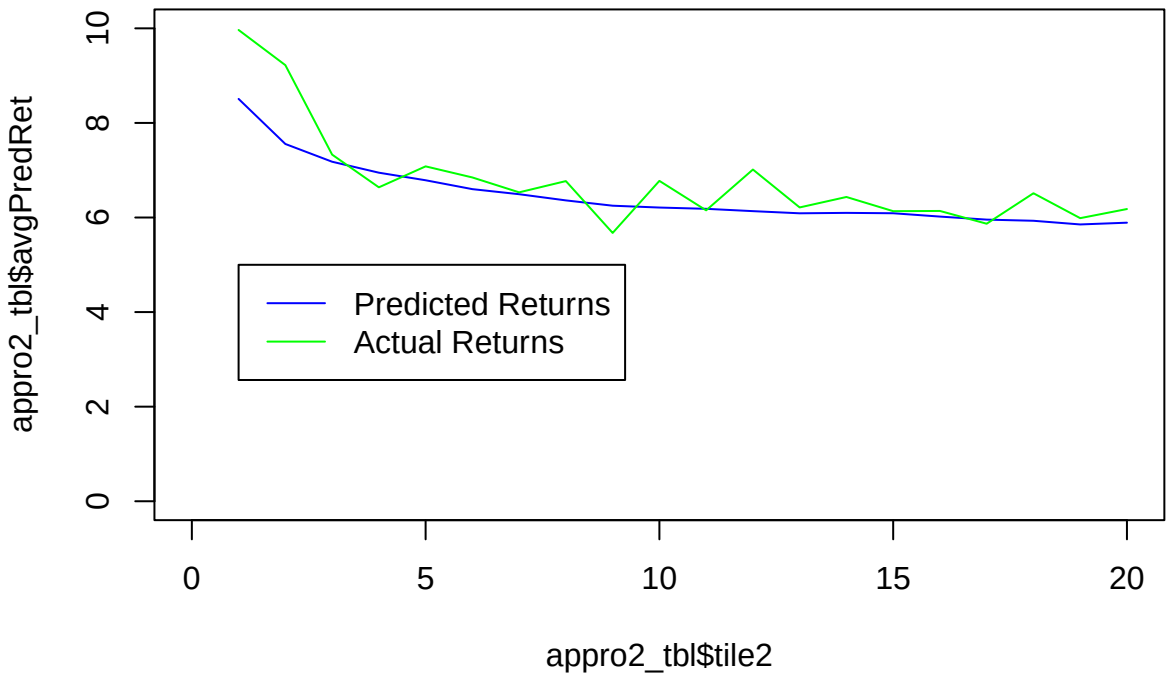
Approach 2 -> Calculating the expected returns by (Predicted returns from a loan) * (prob. of being FullyPaid) and select top fraction of loans based on this product of scores and sort these based on high those calculated scores and then select the top (4%) loans

```
## # A tibble: 20 x 15
```

```
##    tile2 count avgPredRet numDefaults avgActRet minRet maxRet avgTer  totA  totB
##    <int> <int>      <dbl>       <int>     <dbl>  <dbl>  <dbl>  <dbl> <int> <int>
##  1     1   600       6.27         183      6.30  -32.1   44.4   2.29     0     0
##  2     2   600       6.43         167      5.75  -33.3   33.6   2.31     0     0
##  3     3   600       6.54         144      6.56  -31.1   26.0   2.21     0     0
##  4     4   600       6.60         118      7.44  -31.4   28.9   2.20     0     4
##  5     5   600       6.58         129      6.75  -28.4   24.5   2.23     0    11
##  6     6   600       6.66         129      6.79  -30.4   34.2   2.08     0    10
##  7     7   600       6.59         112      6.93  -32.2   26.4   2.14     0    33
##  8     8   600       6.62          91      7.55  -27.3   25.1   2.16     0    29
##  9     9   600       6.61         101      7.01  -29.2   27.0   2.16     0    37
## 10    10   600       6.63          89      7.34  -29.2   26.8   2.17     0    60
## 11    11   600       6.54          86      6.79  -30.0   20.7   2.21     0    91
## 12    12   600       6.52          95      6.38  -31.1   29.2   2.22     0   124
## 13    13   600       6.47          88      6.40  -31.1   21.3   2.24     0   144
## 14    14   600       6.37          65      7.05  -32.2   23.7   2.15     0   203
## 15    15   600       6.36          59      7.37  -31.0   26.2   2.09     0   254
## 16    16   600       6.29          59      6.77  -30.0   20.3   2.18     0   302
## 17    17   600       6.23          55      6.89  -26.9   23.0   2.19     0   357
## 18    18   600       6.23          57      6.43  -30.1   20.3   2.10     0   413
## 19    19   600       6.27          60      6.43  -32.2   25.4   2.16     0   436
## 20    20   600       6.33          46      6.55  -30.8   22.1   2.09     2   472
## # ... with 5 more variables: totC <int>, totD <int>, totE <int>, totF <int>,
## #   totG <int>
```

```
## # A tibble: 20 x 15
##    tile2 count avgPredRet numDefaults avgActRet minRet maxRet avgTer  totA  totB
##    <int> <int>      <dbl>       <int>     <dbl>  <dbl>  <dbl>  <dbl> <int> <int>
##  1     1   600       8.51         119      9.97  -28.3   32.0   2.13     0     0
##  2     2   600       7.56         111      9.22  -31.4   26.0   2.05     0     0
##  3     3   600       7.18         135      7.33  -29.8   44.4   2.18     0     0
##  4     4   600       6.95         137      6.64  -30.2   29.5   2.24     0     1
##  5     5   600       6.79         120      7.08  -29.8   24.5   2.17     0     3
##  6     6   600       6.60         120      6.85  -33.3   24.6   2.17     0     8
##  7     7   600       6.49         120      6.53  -31.1   21.1   2.21     0    11
##  8     8   600       6.36         116      6.77  -29.2   22.0   2.26     0    24
##  9     9   600       6.25         135      5.67  -32.1   22.0   2.20     0    28
## 10    10   600       6.21          92      6.77  -31.1   27.0   2.16     0    42
## 11    11   600       6.18          99      6.15  -30.4   21.7   2.24     0    50
## 12    12   600       6.13          78      7.01  -32.2   23.7   2.23     0    85
## 13    13   600       6.09          93      6.21  -31.0   29.2   2.19     0   116
## 14    14   600       6.10          81      6.43  -30.0   20.0   2.20     0   195
## 15    15   600       6.09          75      6.13  -32.2   17.0   2.20     0   224
## 16    16   600       6.02          79      6.14  -31.1   22.0   2.15     0   304
## 17    17   600       5.96          68      5.87  -32.2   16.5   2.21     0   379
## 18    18   600       5.93          55      6.51  -30.0   23.0   2.21     0   438
## 19    19   600       5.85          57      5.99  -32.2   25.4   2.11     0   518
## 20    20   600       5.89          43      6.18  -30.1   17.3   2.08     2   554
## # ... with 5 more variables: totC <int>, totD <int>, totE <int>, totF <int>,
## #   totG <int>
```


appro1_tbl$avgPredRet
0
2
4
6
8
10
0
5
10
15
20
appro1_tbl$tile2
Predicted Returns
Actual Returns



The above table shows the predicted returns,by xgb model, sorted by the expected returns (calculated in approach 2). We observe that there are almost no grade A loans which give us the best returns and with approach 2 we have loans with higher returns - low defaults compared to the first approach.

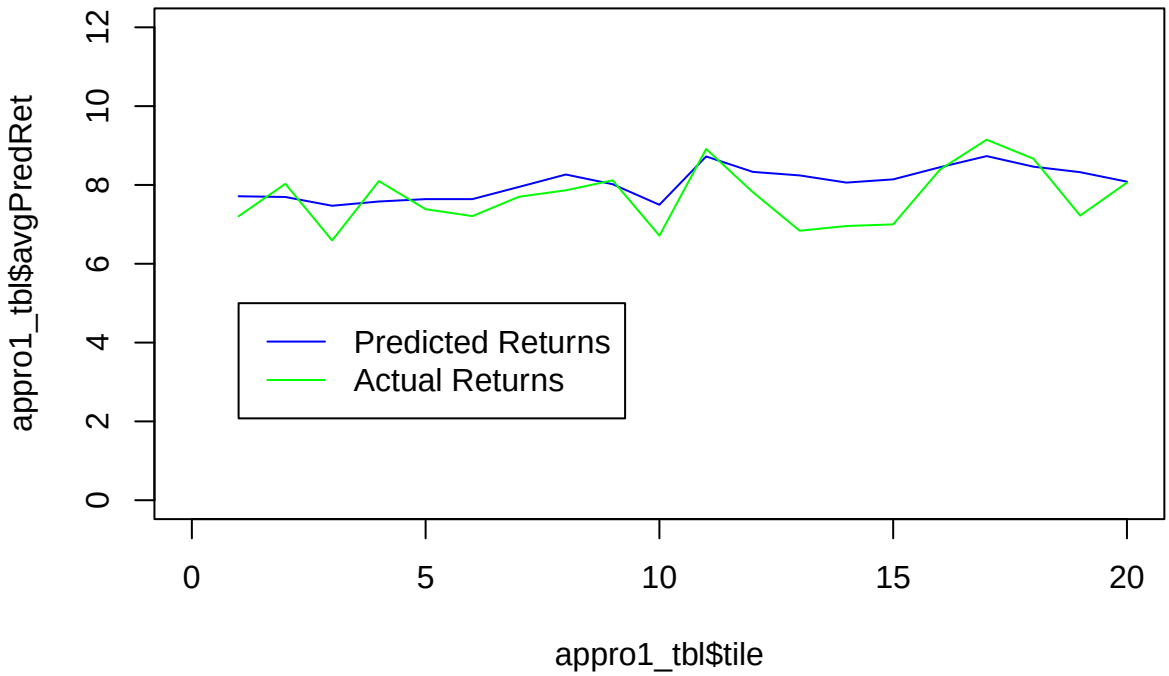
Compared to single models (and approach 1 results) the combined model through approach 2 performs better having good returns and low defaulted loans.

Let's focus on low grade loans now, because higher grade loans are less likely to default but also carry lower interest rates; many lower grad loans are fully paid, and these can yield higher returns. One approach may be to focus on lower grade loans (C and below), and try to identify those which are likely to be paid off.

Now we develop models from the data on lower grade loans, and check if this can provide an effective investment approach.

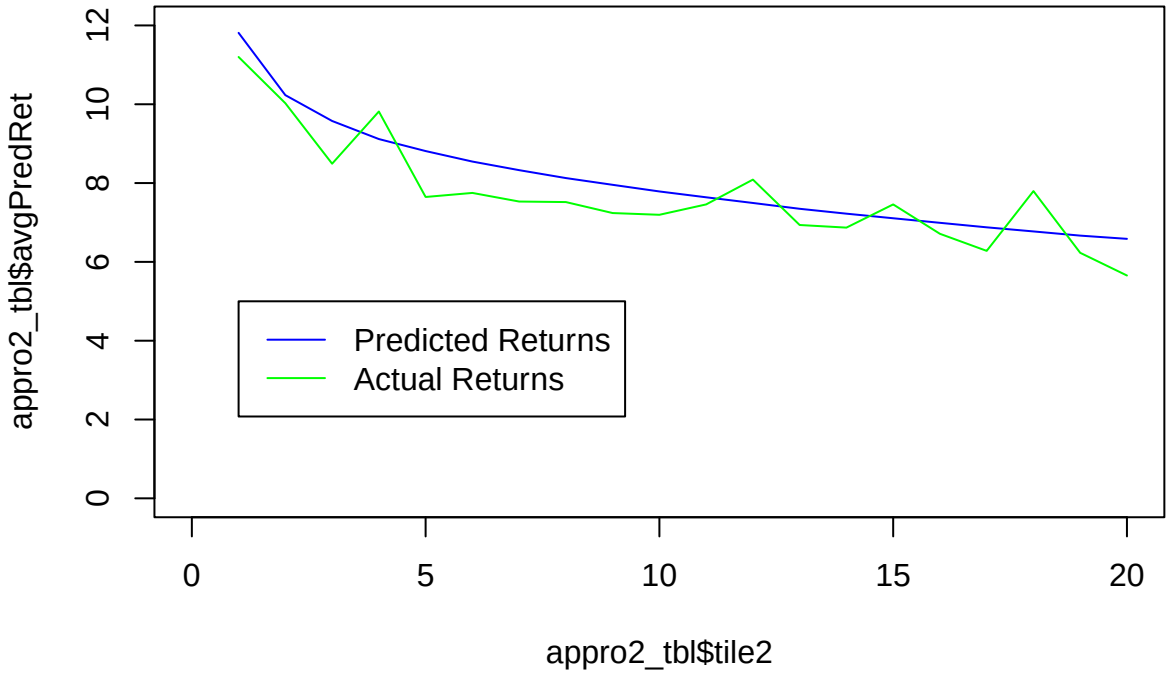
```
## # A tibble: 20 x 15
##     tile count avgPredRet numDefaults avgActRet minRet maxRet avgTer  totA  totB
##    <int> <int>      <dbl>       <int>     <dbl>  <dbl>  <dbl>  <dbl> <int> <int>
##  1     1   263       7.71          31      7.21  -32.2   17.0   2.27     0     0
##  2     2   263       7.69          25      8.03  -23.1   20.9   2.27     0     0
##  3     3   263       7.47          35      6.59  -29.8   19.3   2.38     0     0
##  4     4   263       7.58          26      8.10  -26.3   20.3   2.07     0     0
##  5     5   263       7.64          29      7.38  -26.3   19.2   2.10     0     0
##  6     6   263       7.64          31      7.21  -28.9   17.7   2.13     0     0
##  7     7   263       7.95          40      7.70  -27.3   29.5   2.02     0     0
##  8     8   263       8.27          44      7.86  -30.2   24.6   2.14     0     0
##  9     9   263       8.02          36      8.12  -26.0   23.7   2.11     0     0
## 10    10   263       7.50          47      6.71  -27.6   16.9   2.01     0     0
## 11    11   263       8.72          37      8.91  -25.2   23.2   2.04     0     0
```

```
## 12    12   263       8.33          50      7.81  -27.5   26.8   2.15     0     0
## 13    13   263       8.24          56      6.84  -31.4   21.7   2.30     0     0
## 14    14   263       8.06          49      6.96  -31.0   19.4   2.21     0     0
## 15    15   263       8.14          60      7.00  -27.9   21.1   2.21     0     0
## 16    16   263       8.45          51      8.39  -30.8   24.1   2.16     0     0
## 17    17   262       8.73          49      9.15  -29.8   26.4   2.06     0     0
## 18    18   262       8.46          54      8.66  -26.1   28.8   2.20     0     0
## 19    19   262       8.32          68      7.22  -26.9   28.9   2.17     0     0
## 20    20   262       8.08          68      8.06  -25.7   32.0   2.26     0     0
## # ... with 5 more variables: totC <int>, totD <int>, totE <int>, totF <int>,
## #   totG <int>
```




```
## # A tibble: 20 x 15
##    tile2 count avgPredRet numDefaults avgActRet minRet maxRet avgTer  totA  totB
##    <int> <int>      <dbl>       <int>     <dbl>  <dbl>  <dbl>  <dbl> <int> <int>
##  1     1   263      11.8           34     11.2   -27.9   26.2   2.04     0     0
##  2     2   263      10.2           35     10.0   -27.4   26.4   2.07     0     0
##  3     3   263       9.58          46      8.49  -30.8   22.3   2.23     0     0
##  4     4   263       9.12          28      9.82  -27.2   28.6   2.00     0     0
##  5     5   263       8.81          47      7.65  -31.4   29.5   2.09     0     0
##  6     6   263       8.55          44      7.75  -31.0   21.0   2.01     0     0
##  7     7   263       8.33          48      7.53  -27.2   24.2   2.16     0     0
##  8     8   263       8.13          43      7.52  -27.5   26.8   2.21     0     0
##  9     9   263       7.96          51      7.24  -26.4   24.6   2.16     0     0
## 10    10   263       7.79          47      7.19  -28.6   29.2   2.09     0     0
## 11    11   263       7.64          44      7.46  -28.6   23.4   2.18     0     0
## 12    12   263       7.49          38      8.09  -29.8   28.9   2.14     0     0
```

```
## 13    13   263       7.35          47      6.94  -28.4   21.9   2.24     0     0
## 14    14   263       7.22          41      6.87  -28.5   19.2   2.26     0     0
## 15    15   263       7.11          45      7.46  -32.2   23.7   2.14     0     0
## 16    16   263       6.99          47      6.71  -26.3   21.1   2.28     0     0
## 17    17   262       6.88          51      6.28  -25.5   19.3   2.22     0     0
## 18    18   262       6.77          38      7.80  -29.2   24.5   2.16     0     0
## 19    19   262       6.67          49      6.23  -27.3   32.0   2.26     0     0
## 20    20   262       6.58          63      5.65  -26.1   29.5   2.32     0     0
## # ... with 5 more variables: totC <int>, totD <int>, totE <int>, totF <int>,
## #   totG <int>
```



Conslusion:

Considering all the approaches, we conclude that combined models are better than single models and also we observed that we can effectively invest in the riskier loan grades (C and below) and can get better returns at higher interest rates compared to that of investing in grade A,B loans with low interest rates, low returns and low risk.

We observed that when all the loan grades are considered most of the loans in the top 1% are from high risk grades D,E,F,G. So we had high defaults. However, when we consider only the lower grades (C and below) the top 1% loans are from grade C which are having low risk rate compared to grades D,E,F,G. Also we observe that we get better returns when considering lower grade loans due to the high interest rates.