Case Study 1

Full Stack Analyst

Eshaan Vora

Style Reference: https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf

Predicting Interest Rate from Individual’s Loan Data

(Note: Loan Data represents loans already issued)

Clean Data

```
#Update file path
filePath = "loans_full_schema.csv"
data = read.csv(filePath, stringsAsFactors = TRUE)

#Define function to print variables with missing values
num_missing_val <- function(data_frame){
  print("Count of Variables' Missing Values:")
  for(i in colnames(data_frame)){
    num_missing <- sum(is.na(data_frame %>% select(i)))
    if(num_missing > 0){
      print(paste0(i, " NA Count: ", num_missing))}}
    }
suppress_messages(num_missing_val(data))
```

```
## [1] "Count of Variables' Missing Values:"
## [1] "emp_length NA Count: 817"
## [1] "debt_to_income NA Count: 24"
## [1] "annual_income_joint NA Count: 8505"
## [1] "debt_to_income_joint NA Count: 8505"
## [1] "months_since_last_delinq NA Count: 5658"
## [1] "months_since_90d_late NA Count: 7715"
## [1] "months_since_last_credit_inquiry NA Count: 1271"
## [1] "num_accounts_120d_past_due NA Count: 318"
```

```
#Most of the missing values are because the filer is single and cannot produce joint filer data
#Filter data based on whether the filer is a single filer
data_clean <- subset(data, is.na(annual_income_joint))

#Remove variables with low explanatory power based on variable importance analysis
data_clean <- select(data_clean,-c(emp_title, emp_length, application_type, annual_income_joint, debt_to_income_j
oint, verification_income_joint, current_accounts_delinq, num_accounts_120d_past_due, num_accounts_30d_past_due))

#Impute "N/A" values to the largest value in the "months_since_delinquency" variable
#This is assuming filers with an "N/A" for the variable "months_since_delinquency" have never been delinquent and
so they should be imputed with the largest value
data_clean$months_since_last_delinq <- data_clean$months_since_last_delinq %>% replace_na(max(data_clean$months_s
ince_last_delinq, na.rm=T))
#Impute "N/A" values to the largest value in the "months_since_90d_late" variable
data_clean$months_since_90d_late <- data_clean$months_since_90d_late %>% replace_na(max(data_clean$months_since_9
0d_late, na.rm=T))
#Impute "N/A" values to the largest value in the "months_since_last_credit_inquiry" variable
data_clean$months_since_last_credit_inquiry <- data_clean$months_since_last_credit_inquiry %>% replace_na(max(dat
a_clean$months_since_last_credit_inquiry, na.rm=T))
```

Split Data into Training and Test Data

```
#Split 80% of data for model training and 20% for model testing
set.seed(1999)
split_data = sort(sample(nrow(data_clean), nrow(data_clean)*.8))
train<-data_clean[split_data,]
test<-data_clean[-split_data,]
```

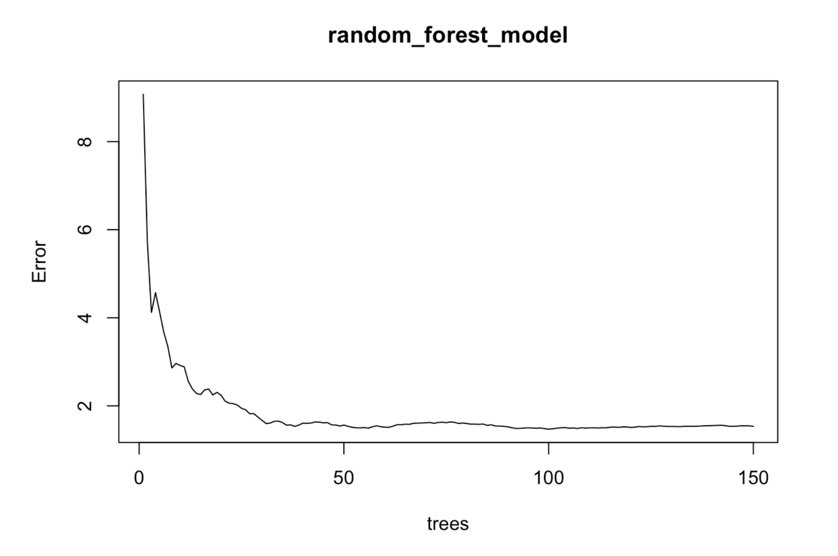
Model Prediction

Random Forest Model

```
#RANDOM FOREST MODEL
#IMPORTANT NOTE: The loan grading variables "grade" and "sub_grade"  and the interest rate given for the loan, ar
e determined on many of the same indicators of credit risk and the loan's sub-grading often determines the additi
onal interest rate adjustment for risk & volatility above the base interest rate, giving the variables an outsize
d explanatory power in interest rate prediction

random_forest_model <- randomForest(interest_rate ~ .
                                    #- grade -sub_grade
                                    -paid_interest -paid_principal -balance -term -total_debit_limit, data = data
_clean, mtry = 5, importance = TRUE, ntree=150)

print(random_forest_model)
```

```
##
## Call:
##  randomForest(formula = interest_rate ~ . - paid_interest - paid_principal -      balance - term - total_debit
_limit, data = data_clean, mtry = 5,      importance = TRUE, ntree = 150)
##                Type of random forest: regression
##                      Number of trees: 150
## No. of variables tried at each split: 5
##
##           Mean of squared residuals: 1.534346
##                     % Var explained: 93.67
```

```
#Error begins to plateau when we use 150 decision trees
#mtry value was tuned starting from n/3 where n is the number of variables (mtry represents number of variables s
ampled per split)
plot(random_forest_model)
```




```
results_random_forest <- data.frame(predict(random_forest_model, test), test$interest_rate) %>% rename(Random_For
est_Predicted_Interest_Rate = 1, Actual_Interest_Rate = 2)

cat("Sample random forest model predictions:\nPredicted Interest Rate vs Actual Interest Rate:")
```

```
## Sample random forest model predictions:
## Predicted Interest Rate vs Actual Interest Rate:
```

```
head(results_random_forest,n=10)
```

```
##    Random_Forest_Predicted_Interest_Rate Actual_Interest_Rate
## 3                              17.247656                17.09
## 9                              13.714717                13.59
## 12                             10.025087                 9.92
## 16                             18.997868                19.03
## 17                             18.806147                19.03
## 23                             13.627758                13.59
## 34                             19.925873                20.39
## 35                              9.902013                 9.93
## 36                              6.857309                 6.08
## 37                             20.504107                21.45
```

```
#Variable importance for random forest model
i_scores <- varImp(random_forest_model, conditional=TRUE) %>% arrange(-Overall)
print("Variable Importance in Predicting Interest Rates: ")
```

```
## [1] "Variable Importance in Predicting Interest Rates: "
```

```
head(i_scores,n=15)
```

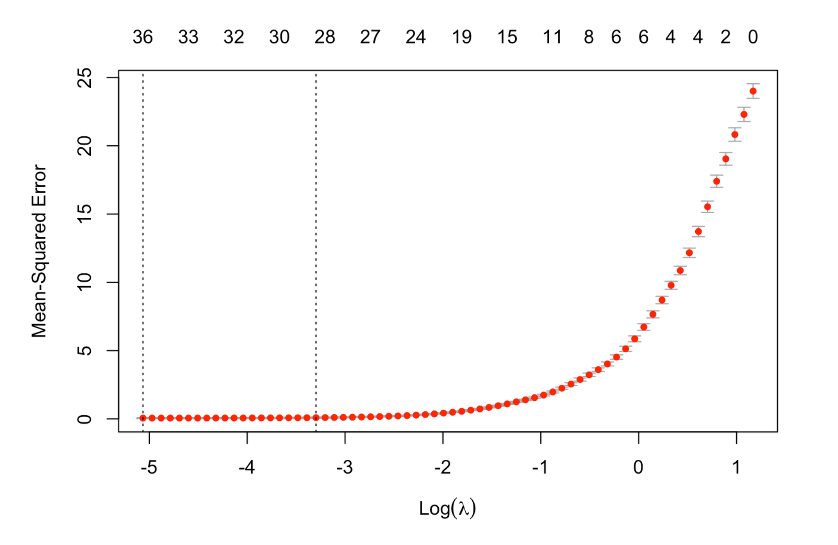
```
##                                Overall
## grade                        25.100389
## sub_grade                    22.427085
## disbursement_method           9.424571
## debt_to_income                8.020390
## verified_income               7.027520
## total_credit_limit            6.979688
## loan_amount                   6.915501
## num_open_cc_accounts          6.538256
## account_never_delinq_percent  6.440258
## open_credit_lines             5.930278
## paid_total                    5.766428
## total_credit_lines            5.519217
## num_cc_carrying_balance       5.267560
## months_since_last_delinq      5.187219
## total_credit_utilized         5.135180
```

Lasso Regression Model

```
#LASSO REGRESSION MODEL
#The explanatory variables exhibit multicollinearity due to the high correlation of credit risk and credit worthi
ness data
#Due to multicollinearity, the model's coefficient estimates will be confounded, and so we will add a high shrink
age penalty
lasso_model <- cv.glmnet(interest_rate ~ .
                         #-grade -sub_grade
                         , data = train, alpha = 1)

print(lasso_model)
```

```
## Call:
## cv.glmnet.formula(formula = interest_rate ~ ., data = train,
##     alpha = 1)
##
## Model fitting options:
##     Sparse model matrix: FALSE
##     Use model.frame: FALSE
##     Number of crossvalidation folds: 10
##     Alpha: 1
##     Deviance-minimizing lambda: 0.006317304  (+1 SE): 0.03700058
```

```
#The MSE plateaus at 58 variables, indicating the lasso model has reduced the coefficient to 0 for 46 variables
#Next we will determine which variables to discard from future modeling
plot(lasso_model)
```




```
results_lasso <- data.frame(predict(lasso_model, test), test$interest_rate) %>% rename(Lasso_Model_Predicted_Inte
rest_Rate = 1, Actual_Interest_Rate = 2)

print("Sample model predictions:")
```

```
## [1] "Sample model predictions:"
```

```
head(results_lasso,n = 10)
```

```
##    Lasso_Model_Predicted_Interest_Rate Actual_Interest_Rate
## 3                            17.431328                17.09
## 9                            13.683321                13.59
## 12                           10.082157                 9.92
## 16                           18.982609                19.03
## 17                           18.975946                19.03
## 23                           13.689552                13.59
## 34                           19.808391                20.39
## 35                           10.060775                 9.93
## 36                            6.234545                 6.08
## 37                           21.352695                21.45
```

```
#Determine variable importance, including the factor levels within string variables
#Determine which variables affect prediction the most at lambda.1se (or 1 standard error away from lambda value w
ith minimum MSE)
#Reference:https://localcoder.org/glmnet-variable-importance
coefList <- coef(lasso_model, s='lambda.1se')
coefList <- data.frame(coefList@Dimnames[[1]][coefList@i+1],coefList@x) %>% arrange(-coefList.x)
names(coefList) <- c('Variable','Coefficient')
print("Variable Importance in Predicting Interest Rates:")
```

```
## [1] "Variable Importance in Predicting Interest Rates:"
```

```
head(coefList,n=10)
```

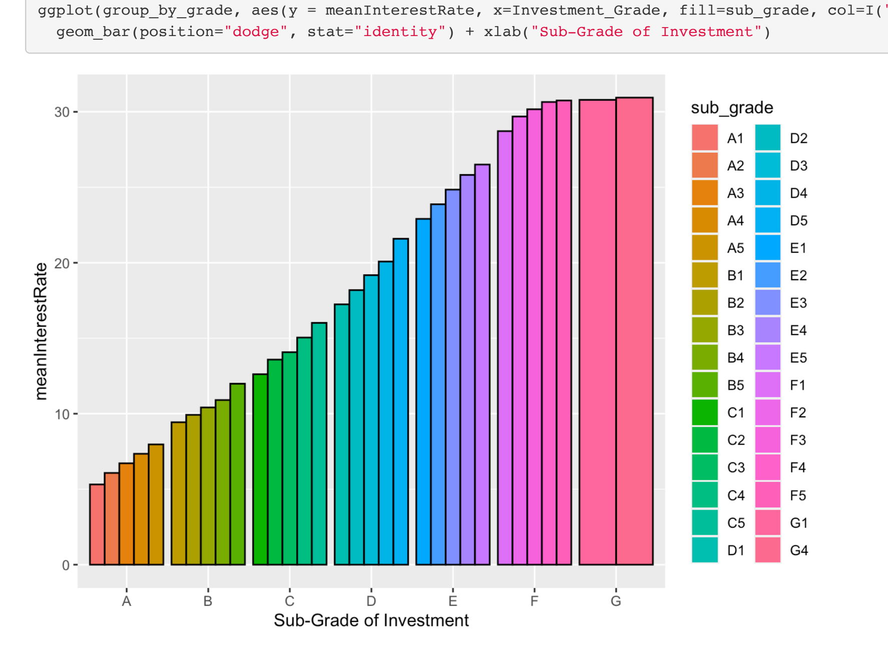
```
##       Variable Coefficient
## 1       gradeG   16.074274
## 2       gradeF   14.951280
## 3  (Intercept)   13.655754
## 4       gradeE   10.250034
## 5       gradeD    4.440544
## 6  sub_gradeD5    3.229437
## 7  sub_gradeC5    2.165247
## 8  sub_gradeE5    2.127242
## 9  sub_gradeD4    1.668061
## 10 sub_gradeE4    1.418022
```

Data Visualizations

Mean Interest Rate vs. Investment Quality

Note: There are no subgrades “G2”,“G3”,“G5” in the dataset

```
#The lower-grade the investment, the riskier and therefore the higher the interest rates
group_by_grade <- data_clean %>% group_by(grade, sub_grade) %>% summarise(meanInterestRate = mean(interest_rate),
.groups='rowwise') %>% rename(Investment_Grade = grade)

ggplot(group_by_grade, aes(y = meanInterestRate, x=Investment_Grade, fill=sub_grade, col=I("black"))) +
  geom_bar(position="dodge", stat="identity") + xlab("Sub-Grade of Investment")
```



Interest Rate by State (Are certain state potentially more expensive to borrow in?)

Note: There is no data available for the state of Iowa (IA)

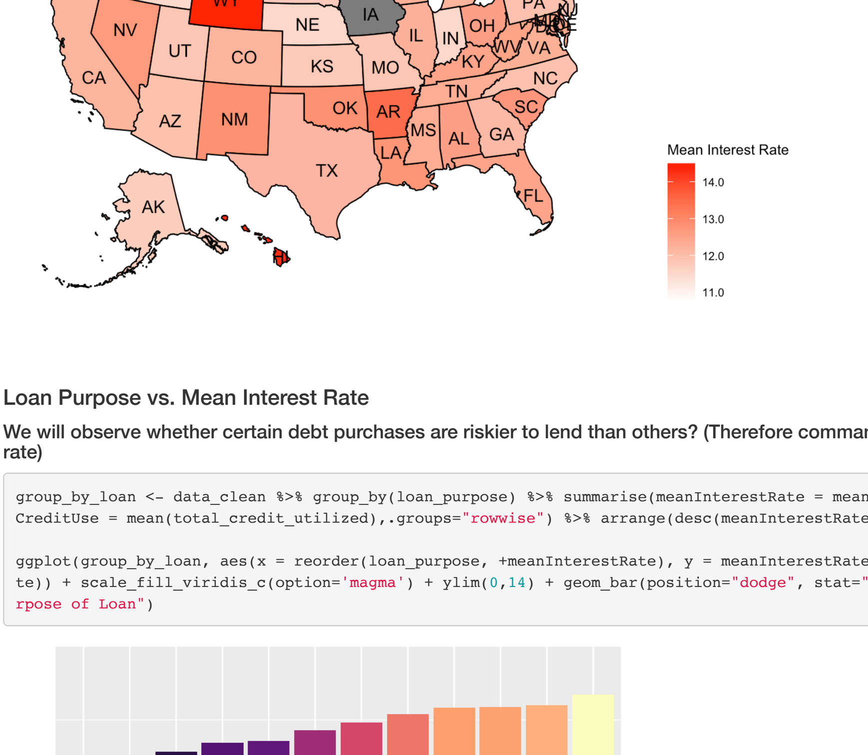
```
group_by_state <- data_clean %>% group_by(state) %>% summarise(meanInterestRate = mean(interest_rate), .groups='r
owwise') %>% arrange(desc(meanInterestRate))

#Top 5 States with highest Interest Rate
head(group_by_state,n=5)
```

```
## # A tibble: 5 x 2
## # Rowwise:  state
##   state meanInterestRate
##   <fct>            <dbl>
## 1 WY                14.5
## 2 HI                14.3
## 3 ND                14.1
## 4 AR                13.4
## 5 DC                13.1
```

```
#Bottom 5 States with lowest Interest Rate
tail(group_by_state,n=5)
```

```
## # A tibble: 5 x 2
## # Rowwise:  state
##   state meanInterestRate
##   <fct>            <dbl>
## 1 AK                11.7
## 2 NE                11.5
## 3 IN                11.5
## 4 ID                11.5
## 5 ME                10.8
```

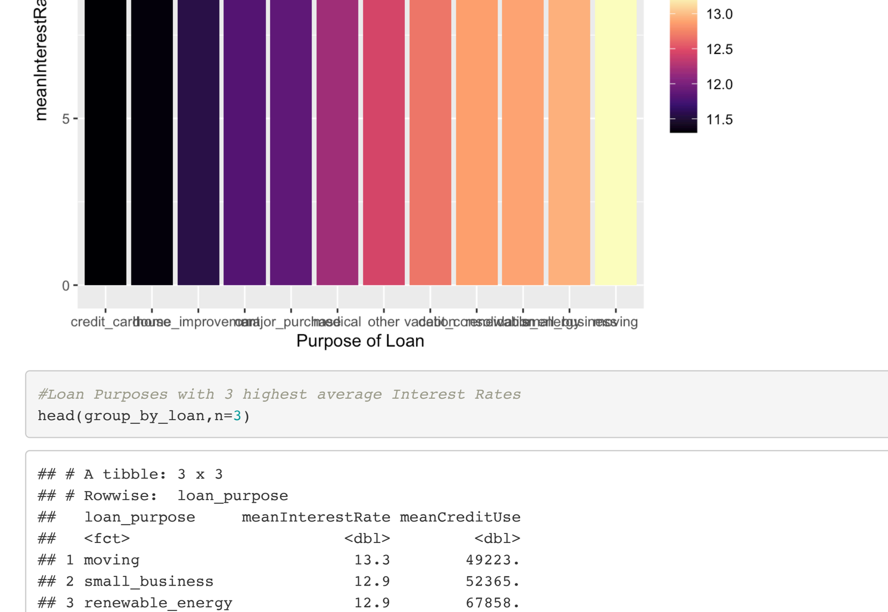
```
#Wyoming, Hawaii, and North Dakota have the highest average interest rates in the country, while Maine has the lo
west average interest rate
plot_usmap(data = group_by_state, values = "meanInterestRate", labels=TRUE) +
  scale_fill_continuous(low = "white", high = "red", name = "Mean Interest Rate", label = scales::comma
  ) + theme(legend.position = "right")
```



Loan Purpose vs. Mean Interest Rate

We will observe whether certain debt purchases are riskier to lend than others? (Therefore commanding a higher interest rate)

```
group_by_loan <- data_clean %>% group_by(loan_purpose) %>% summarise(meanInterestRate = mean(interest_rate), mean
CreditUse = mean(total_credit_utilized),.groups="rowwise") %>% arrange(desc(meanInterestRate))

ggplot(group_by_loan, aes(x = reorder(loan_purpose, +meanInterestRate), y = meanInterestRate, fill=meanInterestRa
te)) + scale_fill_viridis_c(option='magma') + ylim(0,14) + geom_bar(position="dodge", stat="identity") + xlab("Pu
rpose of Loan")
```




```
#Loan Purposes with 3 highest average Interest Rates
head(group_by_loan,n=3)
```

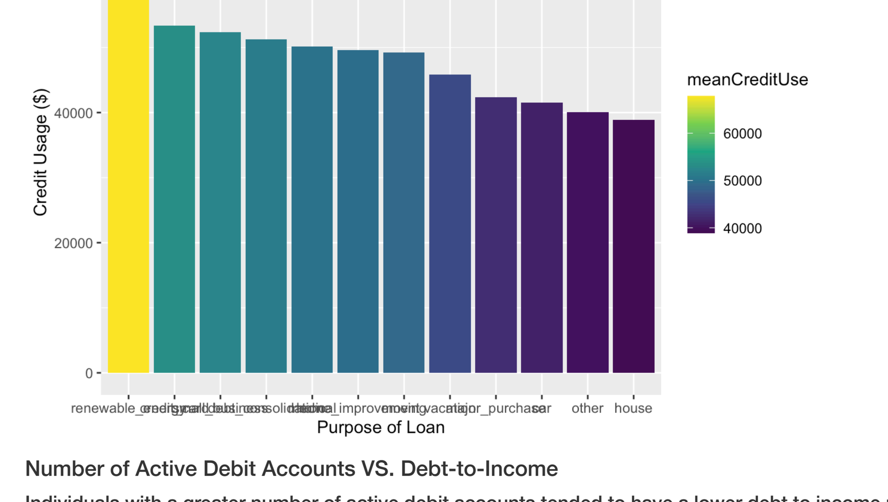
```
## # A tibble: 3 x 3
## # Rowwise:  loan_purpose
##   loan_purpose     meanInterestRate meanCreditUse
##   <fct>                       <dbl>         <dbl>
## 1 moving                       13.3        49223.
## 2 small_business               12.9        52365.
## 3 renewable_energy             12.9        67858.
```

```
#Loan Purposes with the 3 lowest average Interest Rates
tail(group_by_loan,n=3)
```

```
## # A tibble: 3 x 3
## # Rowwise:  loan_purpose
##   loan_purpose     meanInterestRate meanCreditUse
##   <fct>                       <dbl>         <dbl>
## 1 home_improvement             11.5        49587.
## 2 house                        11.3        38896
## 3 credit_card                  11.3        53330.
```

Loan Purpose vs. Credit Utilization

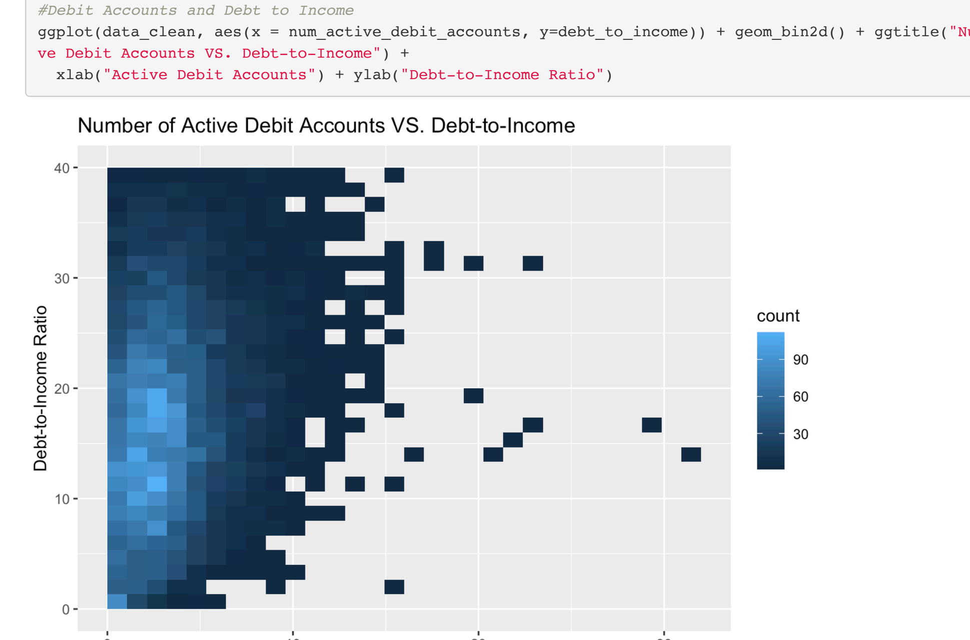
```
#Credit Usage represents loan balance based on revolving credit and excluding mortgages
#Customers who sought funding for Renewable Energy had the highest level of credit debt among customers seeking a
ll type of loans
ggplot(group_by_loan, aes(x = reorder(loan_purpose, -meanCreditUse), y = meanCreditUse, fill=meanCreditUse)) + sc
ale_fill_viridis_c() + geom_bar(position="dodge", stat="identity") + ggtitle("Purpose of Loan VS. Credit Utilizat
ion") + xlab("Purpose of Loan") + ylab("Credit Usage ($)")
```



Number of Active Debit Accounts VS. Debt-to-Income

Individuals with a greater number of active debit accounts tended to have a lower debt to income ratio

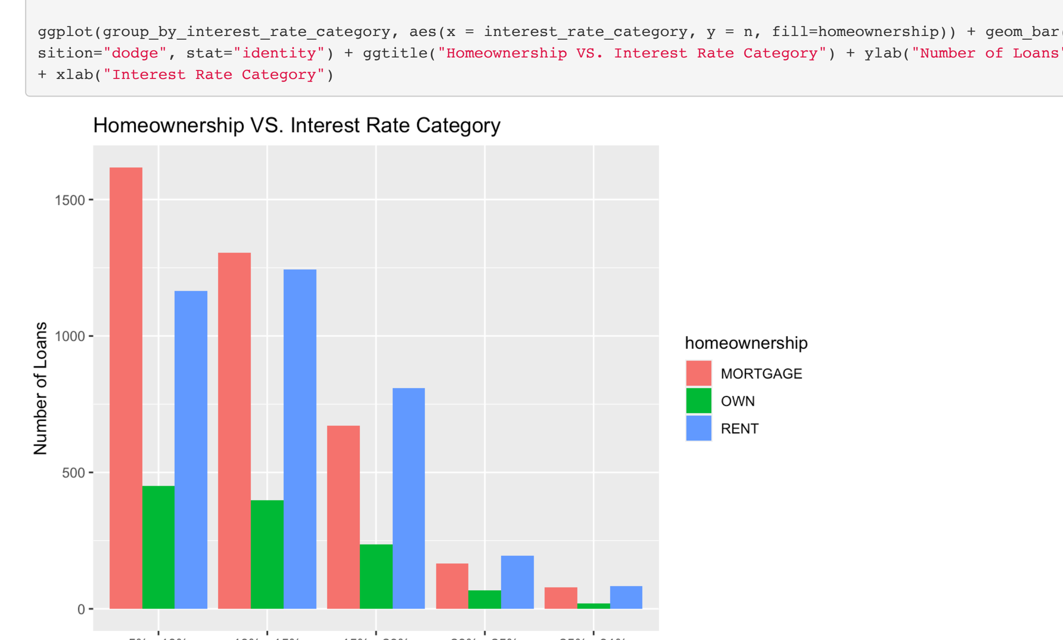
```
#Debit Accounts and Debt to Income
ggplot(data_clean, aes(x = num_active_debit_accounts, y=debt_to_income)) + geom_bin2d() + ggtitle("Number of Acti
ve Debit Accounts VS. Debt-to-Income") +
  xlab("Active Debit Accounts") + ylab("Debt-to-Income Ratio")
```



Homeownership vs. Mean Interest Rate

Renters had interest rates, most often, between 10-15%, unlike mortgagers and owners who had more loans fall between 5%-10%

```
#Break the continuous variable "interest_rate" into
data_clean$interest_rate_category <- cut(data_clean$interest_rate,
              breaks=c(5, 10, 15, 20, 25, 31),
              labels=c('5% - 10%', '10% - 15%', '15% - 20%', '20% - 25%' , '25% - 31%'))

group_by_interest_rate_category <- data_clean %>% group_by(interest_rate_category) %>% count(homeownership)

ggplot(group_by_interest_rate_category, aes(x = interest_rate_category, y = n, fill=homeownership)) + geom_bar(po
sition="dodge", stat="identity") + ggtitle("Homeownership VS. Interest Rate Category") + ylab("Number of Loans")
+ xlab("Interest Rate Category")
```




```
group_by_ownership <- data_clean %>% group_by(homeownership) %>% summarise(meanInterestRate = mean(interest_rate)
, .groups="rowwise") %>% arrange(desc(meanInterestRate))
```

Case Study #1

Below is a data set that represents thousands of loans made through the Lending Club platform, which is a platform that allows individuals to lend to other individuals.

We would like you to perform the following using the language of your choice:

- Describe the dataset and any issues with it.
- Generate a minimum of 5 unique visualizations using the data and write a brief description of your observations. Additionally, all attempts should be made to make the visualizations visually appealing
- Create a feature set and create a model which predicts *interest_rate* using at least 2 algorithms. Describe any data cleansing that must be performed and analysis when examining the data.
- Visualize the test results and propose enhancements to the model, what would you do if you had more time. Also describe assumptions you made and your approach.

Dataset

https://www.openintro.org/data/index.php?data=loans_full_schema

Output

An HTML website hosting all visualizations and documenting all visualizations and descriptions. All code hosted on GitHub for viewing. Please provide URL's to both the output and the GitHub repo.

* If you submit a jupyter notebook, also submit the accompanying python file. You may use python(.py), R, and RMD(knit to HTML) files. Other languages are acceptable as well.

Case Study 1 NOTES:

Using R:

Generate visualizations, group data by relevant levels for ggplot

-Time Series Forecast Graph

-Plot interest rates per demographic or geographical labels

Modeling Objectives: Build accurate model, identify most important variables, test model fit

-Create Random Forest Model (Try gradient boosting model too, but observe if data is overfit)

-Create Elastic Net model or lasso regression model

Use R markdown to format results into HTML