Capstone Project I
Lending Club loan status prediction

Synopsis

Lending Club is a US peer-to-peer lending company, which operates an online lending platform that enables borrowers to obtain a loan, and investors to purchase notes backed by payments made on loans. Lending Club is the world's largest peer-to-peer lending platform. The company claims that $15.98 billion in loans had been originated through its platform up to December 31, 2015.

To find good borrowers is key for the investors in Lending Club to make profit. How to find a good borrowers? In other word, how to predict the loan status according to the information of the loan applicants? A predictive model of loan status can help the investors to make a wise decision for the loan applicants and decrease the risk of bad loans.

In this report, we attempt to predict the risk of the loan being default based on the past loan data. We obtained data from Lending Club’s website. We also get external data resource for the unemployment rate in United States by zip codes from Kaggle. The assumption is if a borrowers from a zip code area with high unemployment rate he is more likely to have a bad loan. The investors can make a judgement according to the zip code of the borrowers for they correlate with Unemployment rate.

We use loan data from year 2015. The target variable in the project is loan status. There are 7 types of loan status. Among them 'Current', 'Fully Paid' and 'in grace period' are called 'good loan'. And the rest four status are called 'bad loan'.

In the project we performed data wrangling, explanatory data analysis and statistical hypothesis testing. We also split the data set into training and test data sets and apply tree based methods to build predictive models for the two loan status. We found that, among multiple machine learning algorithms that we tried, Gradient boosting method provided a reasonable trade-off performance, and a higher return than the naive loan picking strategy can be achieved.

Data wrangling

a. Data wrangling of unemployment rate data sets 2015

1. Data resources:
a. https://www.kaggle.com/jayrav13/unemployment-by-county-us
b. https://www.gaslampmedia.com/download-zip-code-latitude-longitude-city-state-county-csv/
c. https://github.com/liyepeng/Spring-Board-Data-Science-Track/blob/master/Lending%20Club%20project/Three_state_unemploy.csv

2. Data set brief introduction:

- Data 'a' is about unemployment rate in counties of 47 states in USA in 2015.
- Data 'b' is the information of counties and corresponding zip codes.
- Data 'c' includes the three states unemployment rates which are not in data 'c'.
- What we want is unemployment rate in zip codes. So we need to merge 'a' with 'c' by counties. For data 'a' only contains 47 states' unemployment rate it should concatenate with data 'c' before it merge with data 'b'.

3. Data 'a' cleaning job:

- Changing states name to abbreviation.
- Adding missing states unemployment information (3 states) from data 'c'.

4. Data 'b' cleaning job:

- This data set contains 62 states among them there are associate states, military base. Keeping 50 states information to match with data 'a'.
- Some county names are different from that in data 'a'. Making the counties' name match in both data 'a' and data 'b' are the majority part of the cleaning work.

5. Data set merging:
The last step is to combine data 'a' and data 'b' together. One county may has several zip code (XXX00 format) and one zip code may correspond with several county.

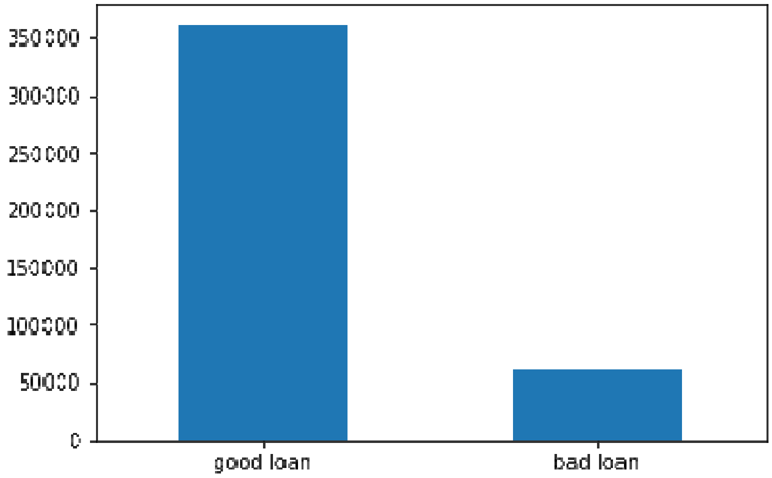
b. Data wrangling for lending club data set

Data resources: https://www.lendingclub.com/info/download-data.action

1. Filling blank with 'NaN'. Because the tree based algorithm doesn't matter the missing values I keep these values in original status.
2. Deleting blank columns, deleting post loan and hard ship variables. Blank columns don't provide any information. Post loan and hard ship variables are not the factors for the investors to make a decision. So we excluded these three types of variables.
3. My target variable is loan status and try to keep the variables it help to make the decision for the loan.
4. Data format wrangling: get rid of '%', change the strings to lower case, check if there are duplicate records. There is no duplicate records in the data set.
5. Feature engineering: There are more than 100000 categories for 'emp_title' variable. I choose top ten titles and create 10 dummy variables so it can be used in EDA and model building.
6. Concatenating unemployment rate in different zip code file with lending club data set on zip code. My aim is to check if the unemployment rate has connection with loan status.
7. There are some outliers in the part of the numerical variables. The tree based method are robust to outliers and they are kept as original in the data set.

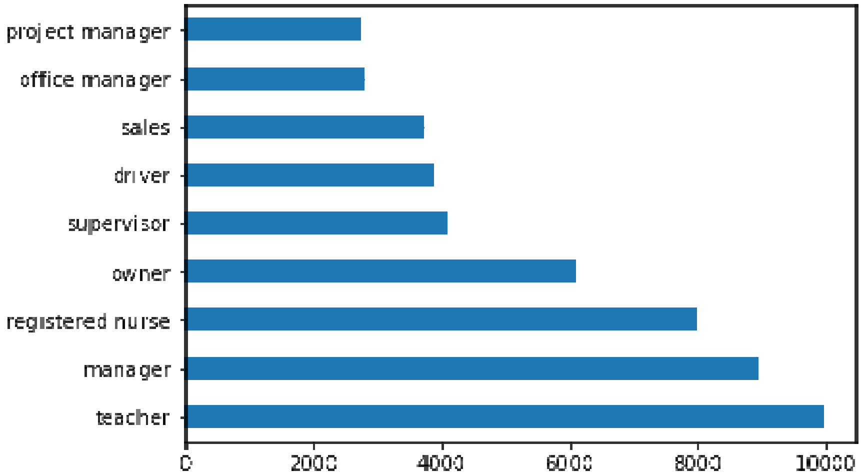
Explanatory Data Analysis

How many good loans and bad loans are there?



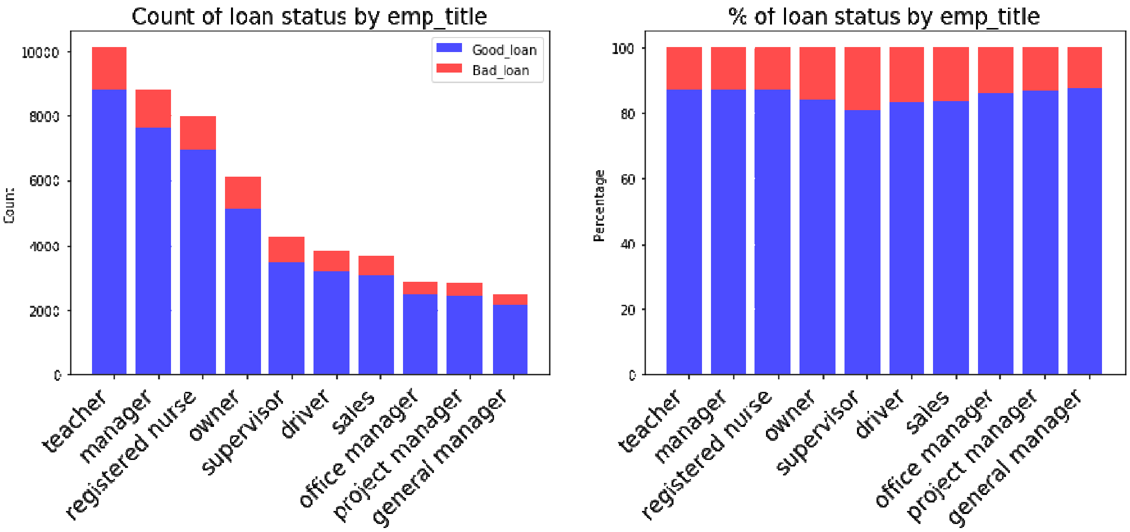
Among all the loans about 20% is bad loans and 80% is good loans.

The loan amount and employment title



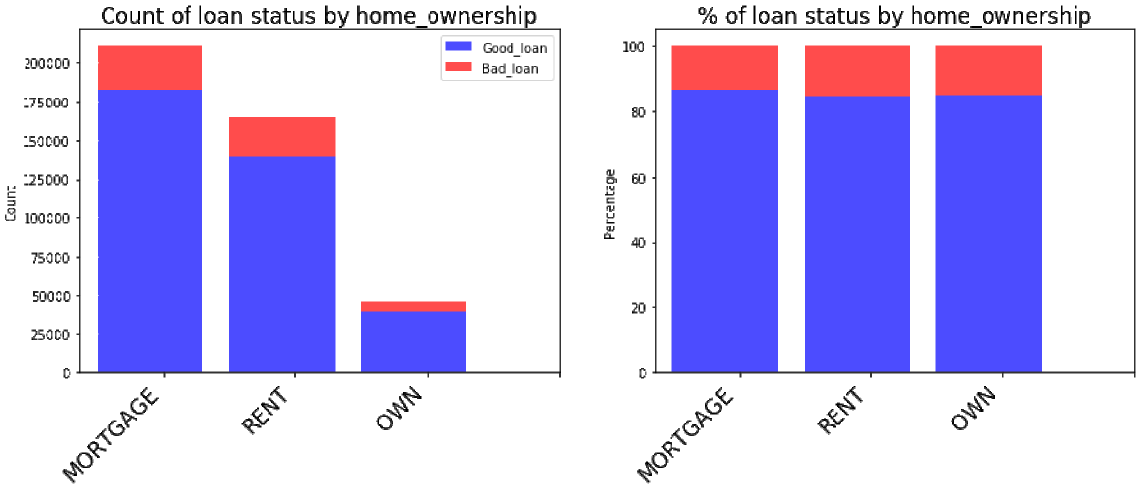
Teacher is the occupation which has the largest number of loans. Manager and registered nurse rank the second and third in the number of loans.

The bad loan percentage for different occupations



Although teachers have largest number of loans they have relatively low bad loan percentage. And manager and registered nurse also have low bad loan ratio.

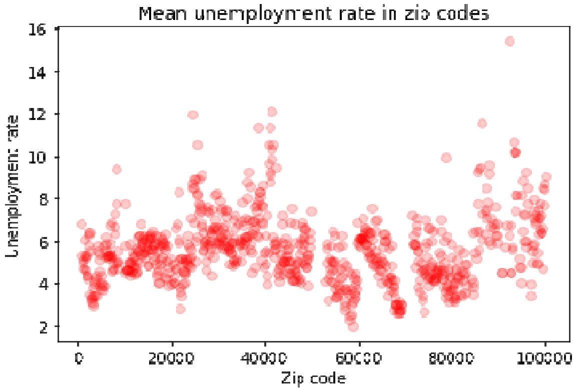
The relationship between loan status and home ownership

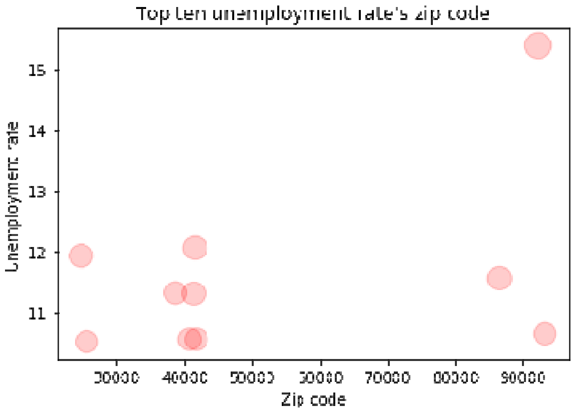


The people who have mortgage have largest number of loans while their bad loan percentage relatively low compare to the renters/home owners.

The relationship of zip code, unemployment rate and bad loan percentage

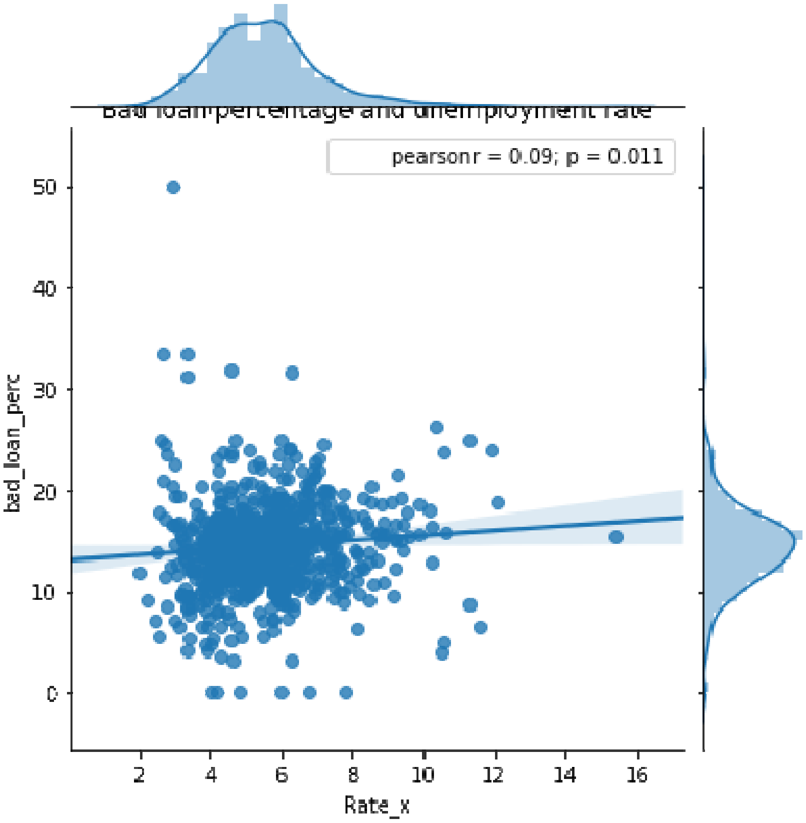
What is the mean unemployment rate in zip codes for bad/good loans?





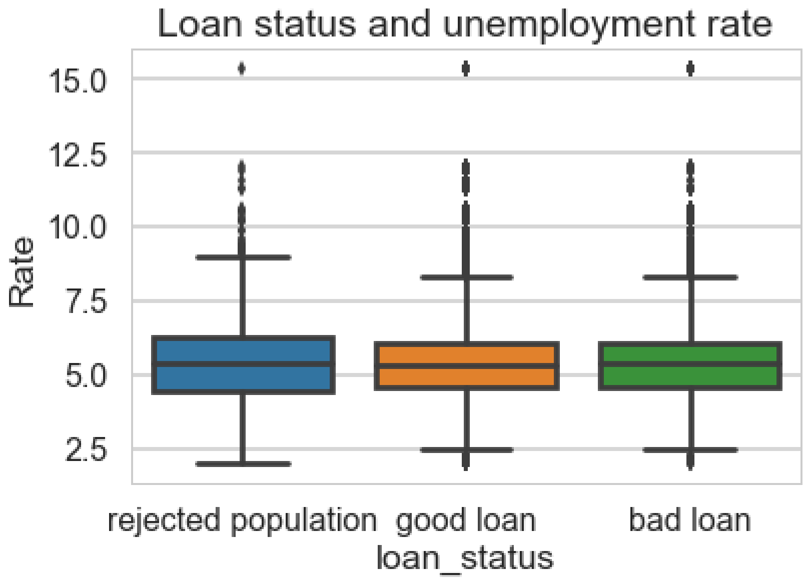
The top ten unemployment rates locate in three zip codes area (20000, 40000, 90000). States CA and AZ's zip codes are around 90000, KY is around 40000, WV and MS are around 20000.

The top ten unemployment rates locate in the zip code areas which are in this five states: CA, AZ, KY, WV and MS. Will the loan lenders from the five states also have high bad loan percentage?



From the above plot we can see that the higher the unemployment rate is the higher the bad loan percentage. So we should pay attention to (not restrict to) the loan applicants from the five states (CA, AZ, KY, WV and MS) which have high unemployment rates. Higher unemployment rates may mean higher bad loan ratio. High bad loan ratio will decrease the profit of the investors.

What are the relationships of the mean unemployment rate for people without loan, in good loan status and in bad loan status in the same zip code area?

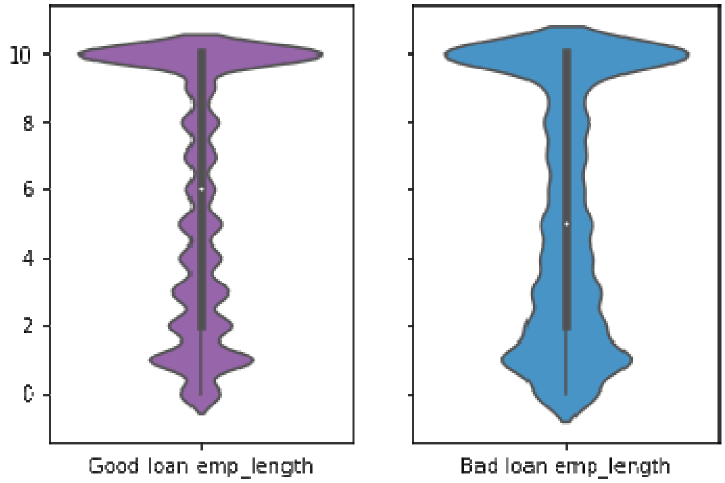


```
Rejected population' mean unemployment rate: 5.523099415204683

Good loan lenders' mean unemployment rate: 5.4941326246224715

Bad loan lenders' mean unemployment rate: 5.554777603913014
```

From the above figure we can see the mean unemployment rate for good loan lenders is lower than that of rejected population and bad loan lenders. The mean calculation shows the mean unemployment rate for rejected population is lower than that of bad loan lenders.

In other words, people that are accepted for a loan and pay back a loan on time typically more easily employed than the rest of the population. While people that are accepted for a loan and don't pay back a loan typically more likely to be unemployed than the rest of the population.

The relationship of employment length and bad loan percentage

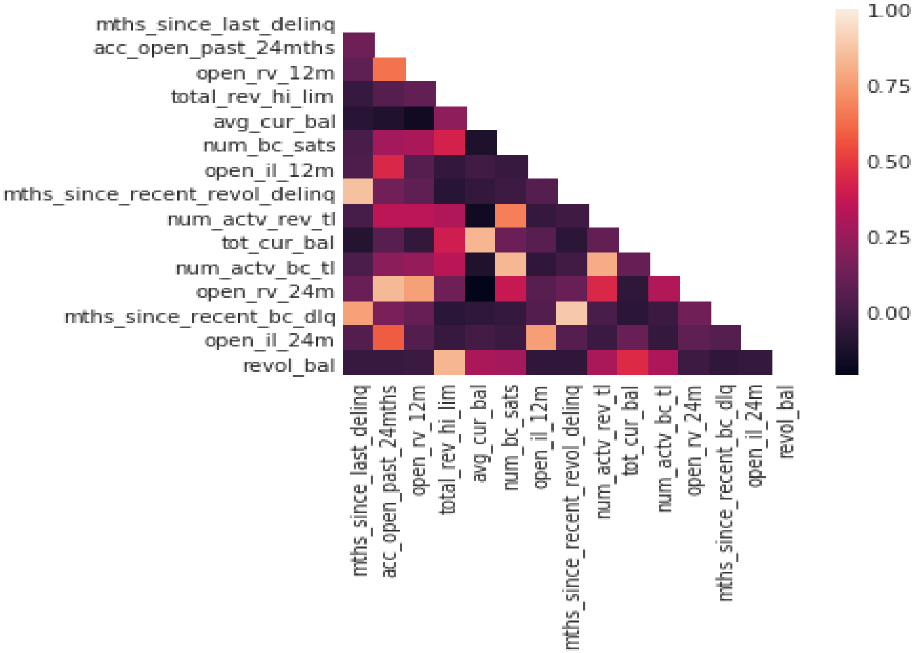


The median employment length of good loan lenders is higher than that of the bad loan lenders. Longer employment length means better financial condition and also the ability for paying the debt on time.

Statistical Analysis report

1. Correlation analysis
There are more than 30 pairs of numeric variables highly correlate and the correlation coefficient are over 0.75.

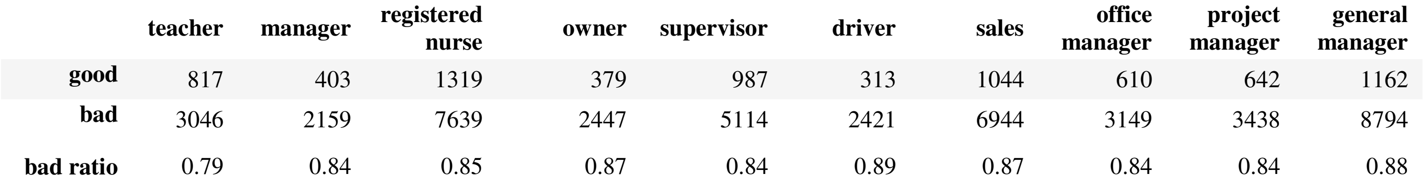
Correlation hot map for the selected variables:



2. Whether people with different employment titles have a different bad loan rate?

Chi square test hypothesis
H_0: In the population, variable 'loan_status' and variable 'emp_title' are independent.
H_1: In the population, variable 'loan_status' and variable 'emp_title' are dependent.

Frequency table



	teacher	manager	registered nurse	owner	supervisor	driver	sales	office manager	project manager	general manager
good	817	403	1319	379	987	313	1044	610	642	1162
bad	3046	2159	7639	2447	5114	2421	6944	3149	3438	8794
bad ratio	0.79	0.84	0.85	0.87	0.84	0.89	0.87	0.84	0.84	0.88

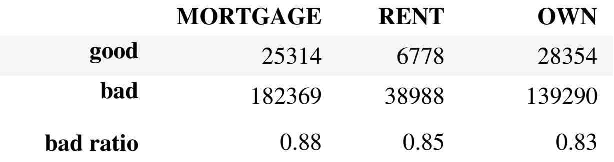
Test statistics
chi_squared_stat: 269.3
Critical value: 21.0260698175
P value: 0.0

Conclusion
We reject H_0 and consider variable 'loan_status' and variable 'emp_title are dependent. Teacher have lowest bad loan percentage than that of other top ten occupations.

3. If people with different home ownership have a different bad loan rate?

Chi square test hypothesis
H_0: In the population, variable 'loan_status' and variable 'home_ownership' are independent.
H_1: In the population, variable 'loan_status' and variable 'home_ownership' are dependent.

Frequency table



	MORTGAGE	RENT	OWN
good	25314	6778	28354
bad	182369	38988	139290
bad ratio	0.88	0.85	0.83

Test Statistics
chi_squared_stat: 1692.8
Critical value: 11.0704976935
P value: 0.0

Conclusion
We reject H_0 and consider variable'loan_status' and variable 'home_ownership' are dependent. It means people who are owing a home have lower bad loan rate than the people who have mortgage or rent a home.

4. Are the medians of employment length of different loan status same?
The Kruskal-Wallis H-test tests the null hypothesis that the population median of all of the groups are equal. It is a non-parametric version of ANOVA. The test works on 2 or more independent samples, which may have different sizes. Note that rejecting the null hypothesis does not indicate which of the groups differs. Post-hoc comparisons between groups are required to determine which groups are different.

Hypothesis
H_0: The population median of employment length in good loan borrowers and bad loan borrowers are equal.
H_1: The population median of employment length in good loan borrowers and bad loan borrowers are not equal.

Test statistics
H-statistic: 189435635.4
P-Value: 0.0

Conclusion
We reject H_0 and consider the population median of employment length in good loan borrowers and bad loan borrowers are equal. The median of good loan lenders' employment length is higher than that of bad loan lenders'.

5. What is the relationship for the mean of unemployment rate in good loan lenders and bad loan lenders?

Two sample t-test hypothesis

H_0: The population mean of unemployment rate of good loan borrowers and bad loan borrowers are equal.
H_1: The population mean of unemployment rate of good loan borrowers and bad loan borrowers are not equal.

Test statistics
t = -9.58285
p = 9.71499e-22

Conclusions:
We reject H_0 and consider the mean of unemployment rate of good loan lenders is lower than that of the bad loan lenders.

Model building

I. Decision Tree

Decision tree learning uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). Tree models where the target variable can take a discrete set of values are called classification trees.

Decision tree is a method **Easy to Understand**, **Useful in Data exploration** and **Less data cleaning required**. But it can easily overfit. What are the key parameters of tree modeling and how can we avoid over-fitting in decision trees? Setting constraints on tree size and Tree pruning are two ways to avoid the over-fitting of decision trees.

In this project using cross validation we tuned **Maximum depth of tree (vertical depth)** which is used to control over-fitting as higher depth will allow model to learn relations very specific to a particular sample.

For the hyper parameter **Minimum samples leaves** which defines the minimum samples (or observations) required in a terminal node or leaf. Used to control over-fitting similar to min_samples_split. Generally lower values should be chosen for imbalanced class problems because the regions in which the minority class will be in majority will be very small. In this project we choose 50 as the value of **Minimum samples leaves.**

Hyper parameter tuning for max depth

```
[(3, 0.85669108733167099), (4, 0.85671483517706548), (5,
0.85673066729330494), (6, 0.85673066729330494), (7,
0.85663567777590433), (8, 0.85623987914651389), (9,
0.85583221575084278)]
```

From the scores we obtained from the tuning process max_depth equaling 5 or 6 is the best choice.

Model score using ‘gini index’ and ‘entropy’

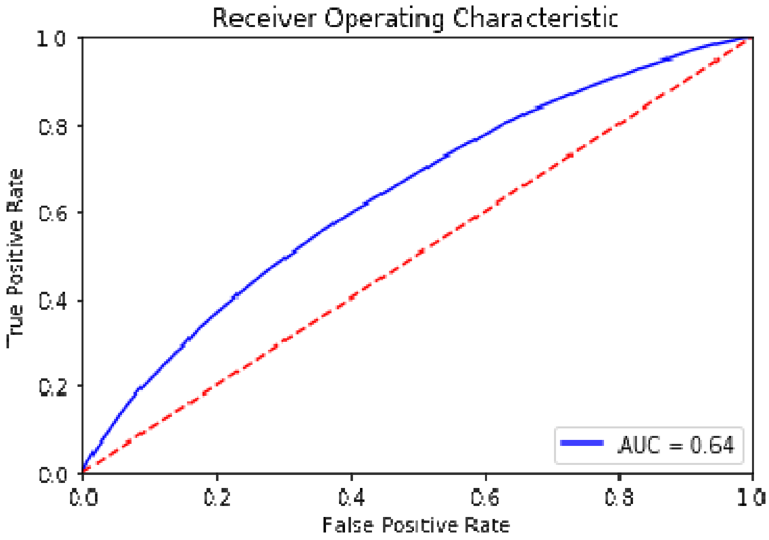
```
DecisionTreeClassifier(class_weight=None, criterion='gini',
max_depth=6, max_features=None, max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=50, min_samples_split=2,
min_weight_fraction_leaf=0.0, presort=False, random_state=0,
splitter='best')
```

```
AUC - ROC gini : 0.636
```

```
AUC - ROC entropy : 0.637
```

The score of the model using 'entropy' is a little bit higher than that using 'gini index'.

ROC_AUC Curve of Decision tree model



II. Random Forest

Random forest is an ensemble tool which takes a subset of observations and a subset of variables to build a decision trees. It builds multiple such decision tree and amalgamate them together to get a more accurate and stable prediction.

There are primarily 3 features which can be tuned to improve the predictive power of the model:

1. max_features:

These are the maximum number of features Random Forest is allowed to try in individual tree. Increasing max_features generally improves the performance of the model as at each node now we have a higher number of options to be considered. However, this is not necessarily true as this decreases the diversity of individual tree which is the USP of random forest. But, for sure, you decrease the speed of algorithm by increasing the max_features.

2. n_estimators:

This is the number of trees you want to build before taking the maximum voting or averages of predictions. Higher number of trees give you better performance but makes your code slower. You should choose as high value as your processor can handle because this makes your predictions stronger and more stable.

3. min_sample_leaf:

If you have built a decision tree before, you can appreciate the importance of minimum sample leaf size. Leaf is the end node of a decision tree. A smaller leaf makes the model more prone to capturing noise in train data. Generally I prefer a minimum leaf size of more than 50.

Features which will make the model training easier:

1. n_jobs:

This parameter tells the engine how many processors are allowed to use. A value of “-1” means there is no restriction whereas a value of “1” means it can only use one processor.

2. oob_score:

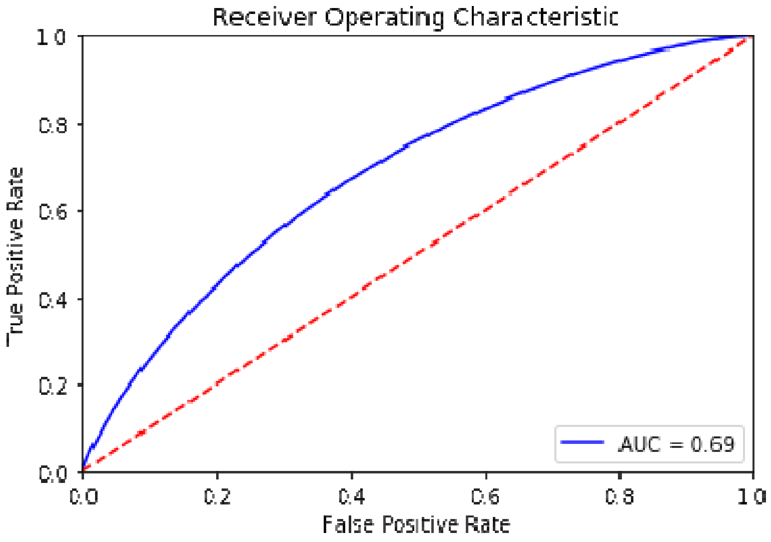
This is a random forest cross validation method. It is very similar to leave one out validation technique, however, this is so much faster. This method simply tags every observation used in different tress. And then it finds out a maximum vote score for every observation based on only trees which did not use this particular observation to train itself.

From the above rules we choose the following Random forest model:

```
Forest = RandomForestClassifier (n_estimators=1000, oob_score = True, random_state=0,
            n_jobs=-1, max_features = "auto", min_samples_leaf = 50)
```

Score the random forest model and plot roc curve

```
AUC - ROC:  0.685
```



Decision tree model's auc_roc score is 0.637 and random forest model get the score 0.685.

III. Gradient Boosting Method

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

Let's consider the important GBM parameters used to improve model performance in Python:

1. learning_rate

This determines the impact of each tree on the final outcome. GBM works by starting with an initial estimate which is updated using the output of each tree. The learning parameter controls the magnitude of this change in the estimates.
Lower values are generally preferred as they make the model robust to the specific characteristics of tree and thus allowing it to generalize well. Lower values would require higher number of trees to model all the relations and will be computationally expensive.

2. n_estimators

The number of sequential trees to be modeled. Though GBM is fairly robust at higher number of trees but it can still over fit at a point. Hence, this should be tuned using CV for a particular learning rate.

3. subsample

The fraction of observations to be selected for each tree. Selection is done by random sampling. Values slightly less than 1 make the model robust by reducing the variance. Typical values ~0.8 generally work fine but can be fine-tuned further.

Tuning the learning rate

```
learning rate 0.0001 mean scores: 0.8567
learning rate 0.001 mean scores: 0.8567
learning rate 0.005 mean scores: 0.8567
learning rate 0.01 mean scores: 0.8567
learning rate 0.05 mean scores: 0.8569
learning rate 0.1 mean scores: 0.8571
```

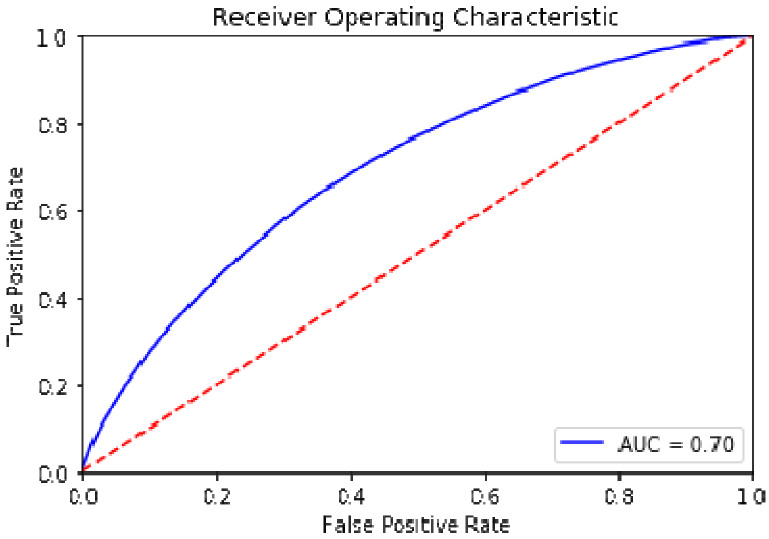
Learning rate decides the number of trees. The lower the learning rate the more of trees needed. Learning rate 0.1 performs better than other smaller rates.

Final models of GBM:

```
GradientBoostingClassifier(criterion='friedman_mse', init=None,
learning_rate=0.2, loss='deviance', max_depth=2, max_features=None,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0, n_estimators=200, presort='auto',
random_state=None, subsample=1.0, verbose=0, warm_start=False)
```

Scoring the GBM model

```
AUC - ROC: 0.695
```

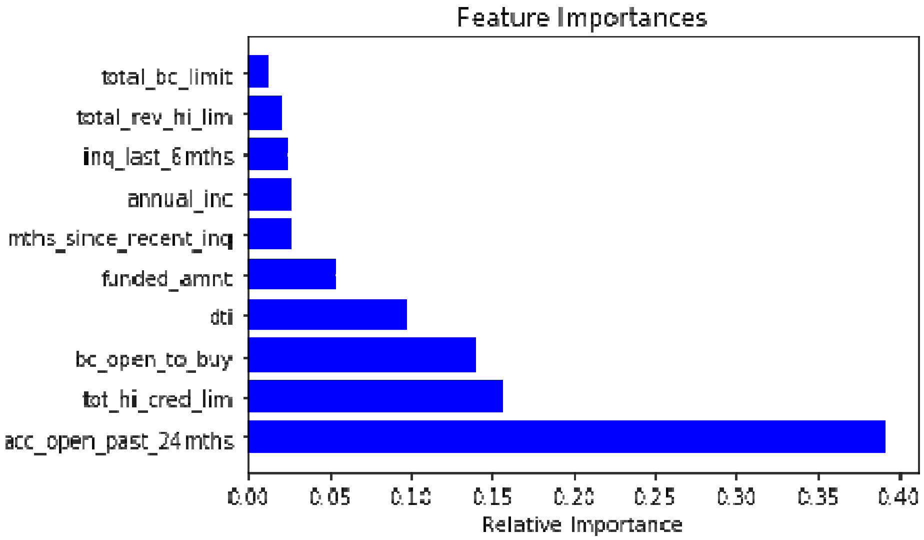


GBM model gets a score of 0.695 and a little bit improvement than Random Forest 0.685.

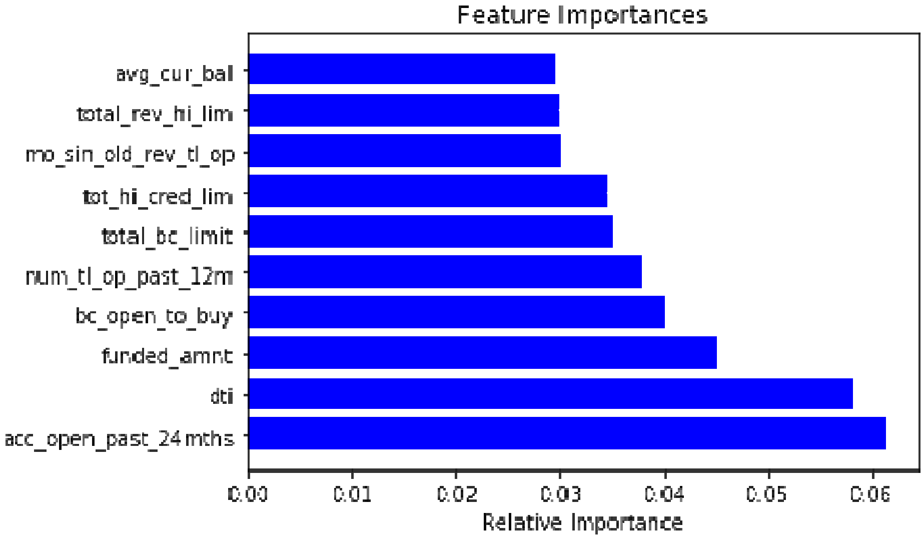
To predict the loan status we tried Decision Tree, Random Forest and Gradient Boost method. We tuned hyper parameters and used cross validation. Finally we increase the ROC_AUC score from 0.637 to 0.695. Gradient Boost method proved to be the best method for the predictive model.

IV. Comparing feature importance in the tree based methods

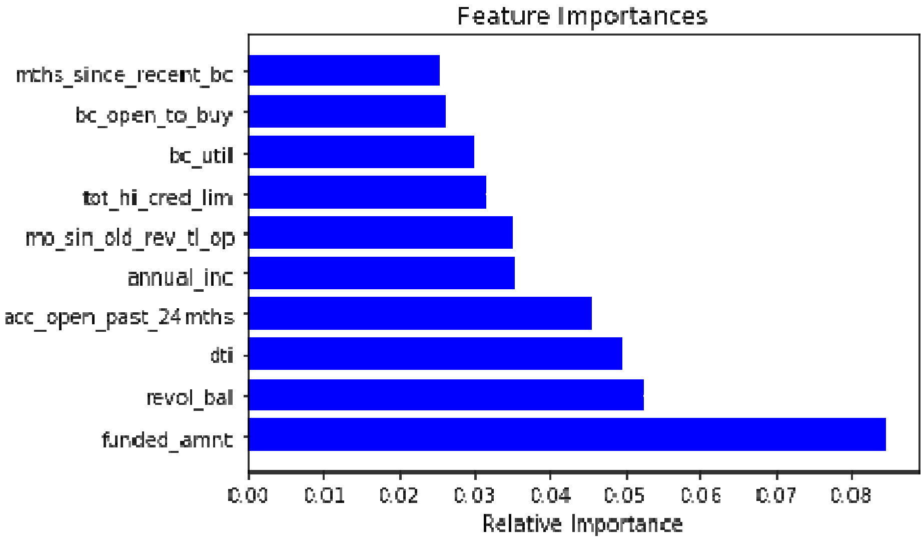
Top 10 important features from Decision tree model



Top 10 important features from Random Forest model



Top 10 important features from Gradient Boost model



V. Discussion of three models' top ten important features

`Common part:`

1. The common important features in the three models are **'acc_open_past_24mths', 'bc_open_to_buy', 'dti', 'funded_amnt', and 'tot_hi_cred_lim'**.
2. Among the five common important features of the three models **'acc_open_past_24mths'**, **'bc_open_to_buy'** and **'dti'** are variables reflect the bank account information of the lenders.
3. **'tot_hi_cred_lim'** shows the credit information and **'funded_amnt'** is the total amount committed to that loan at that point in time. Another common important feature is **'annual_inc'** which is in both decision tree model and GBM. And it ranks 11 in random forest model.

Different part:

Decision tree:

4 bank variabls: acc_open_past_24mths, bc_open_to_buy, dti, inq_last_6mths
4 credit variables: tot_hi_cred_lim mths_since_recent_inq total_rev_hi_lim total_bc_limit
1: funded_amnt
1: annual_inc

Random Forest:

6 bank variables: acc_open_past_24mths, bc_open_to_buy, dti, num_tl_op_past_12m, mo_sin_old_rev_tl_op , avg_cur_bal
3 credit varaibles: tot_hi_cred_lim total_bc_limit total_rev_hi_lim
1: funded_amnt

Gradient boost method:

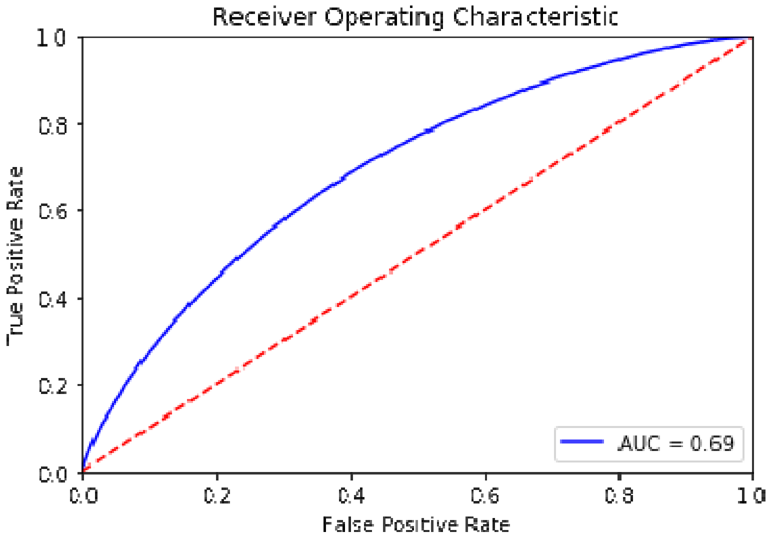
6 bank variables: acc_open_past_24mths, bc_open_to_buy, dti , mo_sin_old_rev_tl_op, mths_sincerecent, bc_util
2 credit variables: revol_bal, tot_hi_cred_lim
1: funded_amnt
1: annual_inc

Except the order of common important features are different, the most interesting thing is that Random Forest and GBM are focus more on bank account information than the Decision tree (0.6 vs 0.4). It shows bank account is a very import factor for the investors to make decision no matter the account is for saving money or spending money.

VI. Build a model without 'Rate' variable

The accuracy of GBM model without 'Rate' Variable is 0.6949 and that of model with 'Rate' is 0.6952. There is a little change for the model accuracy with or without 'Rate' variable. The reason may be that original unemployment rate we get is by counties not zip codes. One county has several corresponding zip codes. If we use census data in U.S and recalculate the unemployment rate by zip code the 'Rate' variable may have more weight in the model.

AUC - ROC: 0.6949



There is still room for the improvement of model performance, however. In the EDA and statistical part we analyzed four features which are employment length, unemployment rate, employment title and home ownership. They all have impacts on the bad loan ratio and statistical testing shows the impacts have statistical significance. But from the feather importance analysis none of them is among the top ten important features. The employment title's importance is zero. It is worth to dig deeper for the feature "employment title". For example we can explore more than 10 employment titles and see if it will increase the importance of the feature. Or we can connect employment title with employment length and create a new variable, because the income for a teacher working for ten years will have great difference with a manager working for ten years.

VII. Random Oversampling in building predictive model

This data set has 80% good loan and 20% bad loan. It is imbalanced data set. We tried SMOTE (Synthetic Minority Over-sampling) Technique in random oversampling method to balance the data set.

Oversampling is a well-known way to potentially improve models trained on imbalanced data. Advantages of oversampling is that unlike under sampling this method leads to no information loss and outperforms under sampling. But oversampling increases the likelihood of overfitting since it replicates the minority class events. Gradient boost method is great because the model architecture reduces overfitting.

GBM model performance for balanced data set:

Prevision score: 0.704

The model performance of Gradient boost in imbalanced data set is 0.695 and that in balanced data set is 0.704. A little bit improvement of model performance has been made by applying oversampling method to balance the data set. There are lots of methods to balance the imbalanced data set including **under sampling, over sampling**, **over sampling** followed by **under sampling** and **Ensemble sampling**. It is worth trying other method to balance the data set and build models in the future work.