High FICO is not the only criteria for Loan Pay offs.

Capstone Project

Varun Kumar Chepuri Sept 15 2018

I. Definition

Project Overview

The emergence of peer-to-peer lending companies has significantly risen over the last decade. One such companies in USA is Lending Club (LC). The offerings of this company were registered with the Security and Exchange Commission (SEC) as the first peer-to-peer company [1]. Although, Lending Club had started pretty well with their stock prices rising pretty high from when they started in 2006 and their stock prices went as high as $ 24, 5 years ago.

Unfortunately, due to various reasons they weren't as successful and their stock prices have been significantly falling down, which are currently at $3.84 as of Aug 27th 2018.

If we look back at an article [2] 5 years ago, during the time when the stock prices where pretty high, an article states that the first and the major complaint this company had at that time, is needing a high FICO score for loan acceptance despite having a good salary and job. With the increase in competition in Peer to Peer Lending and better approval rates at other P2P lending companies, the interest in Borrowers to vest money has declined and has been one of reasons for its downfall.

Problem Statement

In this project, I propose to classify a loan into 2 categories: Defaulted Loans and Paid-off Loans. Once I achieve that, I will also analyze that the FICO score in loans that were payed-off vs the ones that were defaulted, to analyze how it varies across these good and bad loans. This problem in the machine learning terms, is a binary classification problem where the model takes 'Loan Status' as the expected output when other features such as 'FICO Score', 'Principal Amount' form the input to this problem. The target variable 'Loan Status' should take into consideration of loans that been completed evaluation and takes values, 'Defaulted' or 'Fully Paid' loans.

Project Design

1. Data Cleaning - Extracting required data - We do not need data about the on-going loans.
2. Feature Engineering
   a. Converting required variables or features into numerical variables.
   b. Removing unrelated columns for classification.
   c. Finding and removing outliers
3. Applying Supervised Algorithms
   a. Benchmark evaluation - Apply logistic regression and random forest models and tune them.
   b. Apply Boosting algorithms like XGBoost and compare the results.
4. Model evaluation
   a. Compare the results with the bench mark model.
   b. Improvements using cost sensitive learning
5. Conclusion
   a. Determine the trend in FICO scores in both the classes.
   b. Future Enhancements.

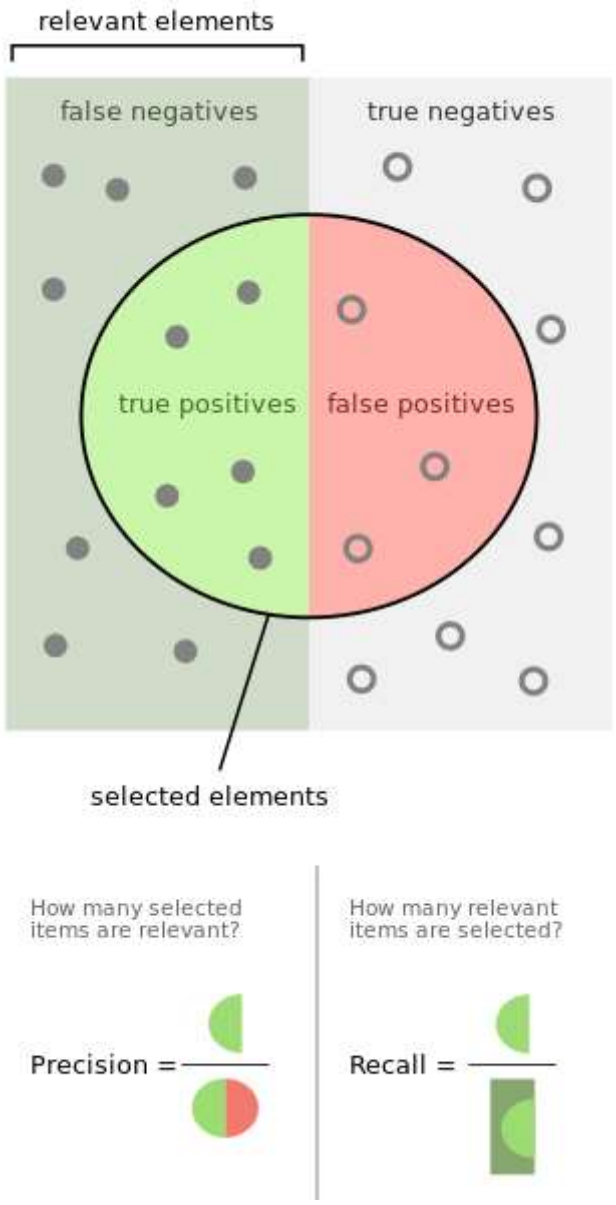
Metrics

For imbalanced classification problems, F1-score is a more suitable metric when compared to accuracy because accuracy can overlook one type of classification: Usually algorithms perform high on true positive cases so accuracy can go high without considering the second type of classification.

Precision and Recall

In general, precision can seen as a measure of exactness or quality, whereas recall is seen as a measure of completeness or quantity. While using precision in measuring an algorithm, high precision determines more relevant results than irrelevant ones and high recall determines most relevant results. [12]

In a classification task, the precision of a class is determined by the ratio of number of true positives to the total number of elements labeled as the positive class. Where as recall, is defined as the number of true positives to the actual number of true positives existing in a particular class. [12]



F1 Score

A general metric used for a binary classification problem is F1 score, also known as F1 measure. It is a measure of test accuracy and computed as follows. A traditional F1 score is a harmonic mean of precision and accuracy. [11]

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

AUC Score

Typically F1-score is evaluated separately for each class and for this binary classification problem, there would be 2 F1-scores. Evaluating the algorithms based on a single score would be easier than considering two different F1-scores. For this reason, we can use AUC (Area under curve) metric for the evaluating the performance of the algorithm as mentioned by an author in his previous attempt [7]. One characteristic of AUC is that it is independent of the fraction of the test population that is from the either classes for this problem.

II. Analysis

Data Exploration

The data-set for Lending Club [3 , 4] is a high dimensional and an imbalanced dataset. It has around 140 features in the original dataset for the years up to 2016 and has rows about 400,000. The data that will be used for this Capstone includes data about only those loans that are accepted by the company. The features like mths_since_last_record with lot of null values, out_prncpl with lots of zero values, suggest that the data includes their ongoing statuses and updated likewise when payment is due.

Out[2]:

	id	member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	installment
0	38098114	NaN	15000.0	15000.0	15000.0	60 months	12.39	336.64
1	36805548	NaN	10400.0	10400.0	10400.0	36 months	6.99	321.08
2	37842129	NaN	21425.0	21425.0	21425.0	60 months	15.59	516.36
3	37612354	NaN	12800.0	12800.0	12800.0	60 months	17.14	319.08
4	37662224	NaN	7650.0	7650.0	7650.0	36 months	13.66	260.20

Curse of Dimensionality

From the below shape, it can be seen that the data-set is of high dimensionality with 150 features and that forms a huge challenge in terms of Data preprocessing. Algorithms like Decision Tree can be extremely sensitive to such data and might need a lot of pruning and won't be ideal for such time of problems [3].

Boosting algorithms like XG-boost or Ada-Boost can be used on such high dimensionality data and they have a clear edge over non-boosting classification problems when the data is of high dimensions. In boosting method, classification is performed using a fitting data by fitting the weight multiple times using the re-weighted data. Although, earlier there was a misconception that boosting algorithms will never over-fit as boosting or functional gradient descent would continue infinitely. But, it is known that one has to stop boosting before the numerical convergence as a way to regularize overfitting.

Although boosting algorithms can also overfit, it can be controlled using a base learner with a weak variance. This property turns out to be beneficial for tuning boosting methods and have an edge over other algorithms when there is a curse of dimensionality [4].

Out[3]: (1646801, 150)

Target variable discrepancy

During the previous attempts [6, 7] made by other authors for the binary classification of this problem, the target variable has been mis-interpreted due to lack of adequate documentation in the data dictionary [5]. The loans with loan statuses such as 'In Grace Period' or 'Late' have been considered as 'Good Loans' or 'Bad Loans' which in reality can be classified either way and one cannot assume to put the data from these variables into one side.

Out[4]:
```
Current                                                788950
Fully Paid                                             646902
Charged Off                                            168084
Late (31-120 days)                                      23763
In Grace Period                                         10474
Late (16-30 days)                                        5786
Does not meet the credit policy. Status:Fully Paid       1988
Does not meet the credit policy. Status:Charged Off       761
Default                                                    70
Name: loan_status, dtype: int64
```

For this binary classification problem, only loan statuses that have a substring of 'Fully Paid' or 'Charged Off' are considered because to determine if a loan is a defaulted or not, only the loans that have been completed needs to be taken into consideration. The loans under payment which have a status of 'Late' or 'In Grace Period' are also current loans but at a different stage of the life cycle of a loan and haven't been completed. One cannot assume such loans to be good or bad without the loan been closed. It is also important to clean the data with null-valued target variables.

Out[5]: 23

Exploratory Visualization

Null valued features

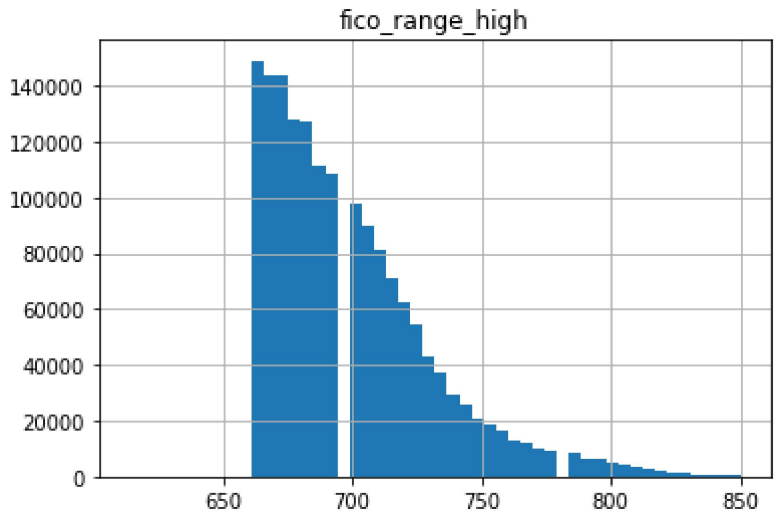
Although there are 150 features, after looking at the data dictionary[5], one can observe that there are a lot of loan information among the 150 features that doesn't apply for binary classification. Features like 'member_id', 'description of loan' etc. are information about the loaner and which doesn't apply for classification. Such features can be examined to removed. Most of those columns are have majority values with null-values, so they can be identified and removed quite easily without having to examine them manually.

Out[6]:
```
member_id                                    100.000000
orig_projected_additional_accrued_interest    99.707190
hardship_length                               99.649138
hardship_reason                               99.649138
hardship_status                               99.649138
deferral_term                                 99.649138
hardship_amount                               99.649138
hardship_start_date                           99.649138
hardship_end_date                             99.649138
payment_plan_start_date                       99.649138
hardship_type                                 99.649138
hardship_dpd                                  99.649138
hardship_payoff_balance_amount                99.649138
hardship_last_payment_amount                  99.649138
hardship_loan_status                          99.649138
sec_app_mths_since_last_major_derog           99.542082
settlement_amount                             99.192981
settlement_percentage                         99.192981
settlement_term                               99.192981
settlement_date                               99.192981
settlement_status                             99.192981
debt_settlement_flag_date                     99.192981
sec_app_revol_util                            98.692192
revol_bal_joint                               98.674703
sec_app_fico_range_low                        98.674703
sec_app_fico_range_high                       98.674703
sec_app_earliest_cr_line                      98.674703
sec_app_inq_last_6mths                        98.674703
sec_app_mort_acc                              98.674703
sec_app_open_acc                              98.674703
sec_app_num_rev_accts                         98.674703
sec_app_chargeoff_within_12_mths              98.674703
sec_app_collections_12_mths_ex_med            98.674703
sec_app_open_act_il                           98.674703
dti_joint                                     97.904422
verification_status_joint                     97.904179
annual_inc_joint                              97.904179
desc                                          92.344734
dtype: float64
```

Out[7]:
```
(1646801, 150)
```

Out[8]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x00000000115D74A8>]],
      dtype=object)
```

Out[9]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x0000000011990240>]],
      dtype=object)
```



FICO score is one of the important features of this analysis. As mentioned earlier, high FICO score is a criteria that the company has insisted on. As we can see above, the history of the FICO scores vs the frequency of those scores has been plotted above. It can be seen that the company has considered FICO scores of at least 670 and most of the customers of Lending Club fall in the category with a FICO score of 670 to 750.

Algorithms and Techniques

Linear Regression

Linear Regression is a linear approach to modeling the relationship between a scalar response with one or more explanatory variables. This is a simple algorithm that works on both classification and regression problems. When the explanatory variable is only one, it would be called as simple linear regression problem. But, when there are multiple explanatory variables it would termed as multiple linear regression.

I have used this algorithm in this classification because linear regression uses the linear predictor functions for modeling the predictions whose unknown model parameters are estimated from the data. There are quite a few parameters such as loan_amt, interest rate etc. that can be used to predict the binary output for the loan classification. While applying this algorithm to classify the data, I would like to use parameters such as fit_intercept so that data can be centered and use for classification. The n_jobs is None by default and I would need to use the default parameter as our target variable consists of only 1 parameter [13].

Decision Tree Classification

Decision Tree Classification is based on a decision tree predictive model that forms branches based on certain decisions at every node to make branches and lead to conclusions about the target value based upon those decisions made. These classification techniques, in the process of classification produce some tree structures where the leaves represents class labels and the branches present conjunctions of features that leads to those class labels.

For this binary classification problem, I have used this algorithm to produce a classifier because the way this algorithm works by taking decisions at different stages of the classification problem would be useful to determine which features would be useful in predicting the classifier. Also, parameters such as minimum sample split and minimum sample leaf would be helpful in producing a good classifier. If we used the default parameters provided by this algorithm, it would overfitting the data.

XG Boosting Algorithm

It is an open source software library which provides an open source software library. It is one of the most used algorithms on Kaggle competitions and it is considered as a winner algorithm on Kaggle. It works on the principle of gradient boosting that is a form of ensemble of weak prediction models. This algorithm work on suitable cost function analysis that is iteratively optimized in the direction of the negative gradient direction [14].

There are various parameters provided by this algorithm that can be used to improve the working of this algorithm for this binary classification problem. Parameters such as max_depth, min_child_weight and subsample can be used for this purpose. These parameters help us not overfit the data, reduce the size of the iterations or the trees and the algorithm can be optimally used for predictions.

Bench Mark Model

In the previous works [6-7] while performing classification on the same dataset, authors have been able to achieve an area under curve for roc curve of around 0.71.

Although, I have considered these as the bench mark models, none of them have used the similar data preprocessing technique and especially some part of the on-going loans have been misclassified into training data by considering them for the training the data. Also, the data-set I have attained is not overly imbalanced like they have considered and few techniques may differ in terms of preprocessing. It would not be completely valid to say that the models presented above completely out perform the work by the previous authors, the preprocessing and effective accountability of the dataset had led to a better accuracy in the above models.

But, it would be worthy to note that the F1 score should also be taken into consideration if one puts forward an argument to consider this as an imbalanced dataset.

III. Methodology

Data Preprocessing

Target Variable

As mentioned earlier, we are considering only the loans that have been closed or that are in completed status. For the case of binary classification, for the case of simplicity we can convert the target variable values into binary by considering '1's as 'Fully Paid' and '0's as 'Defaulted' loans class. Also, null valued variables are now removed.

Out[12]:
```
1    648890
0    168915
Name: loan_status, dtype: int64
```

Remove null valued columns

Secondly, Features like settlement_date, settlement_status, debt_settlement_flag etc. are those features that are introduced after a loan approval and during the settlement process. If we consider such features, we may attain better prediction but such features do not exist when a person is applying for a loan. Theoretically, they would help us improve the performance but practically they would not be valid. So removing such features that are irrelevant.

Removing columns with over 95% of null values as stated above in the Null valued features section.

Irrelavent features removal

Firstly, separating the target variable from the features and reducing unnecessary complexity. Most of these features are a subset of the information that is being served by the other features. For example - zip_code is closely related to the Address State as the last 2 digits of the zip codes are not revealed and doesn't give an exact information.

Converting features into numerical variables

One Hot Encoding

Converting the categorical features into discrete features with binary values. Such encoding makes the life easy while trying to fit various classifiers from sklearn package that require the features to be numerical. As you can see in the below examples that the 'grade' feature is a categorical feature that has 6 categories.

We can see from below shape of the input features that the number of features have increased due to One Hot encoding technique. There are some interesting parameters available for the get_dummies operations [8] like the drop_first that removes the first level of a categorical data, which could be used. But, this may not be needed in this scenario.

Out[16]:
```
(817805, 214)
```

Handling missing values

Imputer transformation

For this highly dimensional dataset, it is difficult to populate all the missing values by considering each column with more accuracy and correctiveness. For this reason, a standard Imputer from the skleran preprocessor is used for filling out the missing values in the dataset. The default value that

is used by the imputer is to replace the missing values is with the mean value of the features and the values that are encoded as np.nan are replace with a string value of "NaN" [9].

Out[19]:

	loan_amnt	funded_amnt_inv	int_rate	installment	annual_inc	dti	delinq_2yrs	fico_range_lo
0	15000.0	15000.0	12.39	336.64	78000.0	12.03	0.0	750.
1	10400.0	10400.0	6.99	321.08	58000.0	14.92	0.0	710.
2	21425.0	21425.0	15.59	516.36	63800.0	18.49	0.0	685.
3	7650.0	7650.0	13.66	260.20	50000.0	34.81	0.0	685.
4	9600.0	9600.0	13.66	326.53	69000.0	25.81	0.0	680.

Train - Test Split

Now that the data is processed, we would need to split the data into test and train data. For this scenario, a standard test split of 20% is considered and a random_state is considered to maintain consistency in randomness.

```
C:\ProgramData\Anaconda2\lib\site-packages\sklearn\cross_validation.py:41: Depr
ecationWarning: This module was deprecated in version 0.18 in favor of the mode
l_selection module into which all the refactored classes and functions are move
d. Also note that the interface of the new CV iterators are different from that
of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
```

Implementation

For this implementation process, it was important to handle the imbalanced data and process the 150 features before applying new techniques and refinement.

1. The results obtained from the pre-processed data were run through the standard metrics of the algorithms to identify if there were any misbehaving features that cannot be accommodated by the algorithm and require some additional processing.
2. Then, identify features that cannot be considered for the classification as they be information populated after closing the loan accounts.
3. It was necessary to identify classifiers that will be suitable for high dimensional data and work with over sampling techniques for the imbalanced data.
4. Evaluating and visualizing the performance of the classifiers.

Linear Regression

Although the name is very deceptive, Linear regression is meant for classification. This would serve well for binary classification using a logistic function. Due to the high number of features, this algorithm might not be the best one to be used and that is one of the reasons for the high values of false positive values in the confusion matrix below compared to other algorithms.

Below is the AUC score for this Linear Regression problem obtained. It is already high than the benchmark model.

Out[25]:
```
0.9210794419173709
```

Decision Tree Classification

Decision Tree Classifiers are a supervised learning method for classification using decision trees. Decision Trees create a model by predicting the value of target variables using certain rules based on the best possible decision to make at each level in a tree and the rules are evaluated until the required model is obtained.

We can see that the AUC Score obtained below is greater than the Linear Regression Model. But, decision tree is known for overfitting the data which needs to be tuned and under go senstivity analysis.

Out[32]:
```
0.9668966347177137
```

Boosting algorithm

Boosting algorithms are popular across data science competitions such as the ones conducted in Kaggle. They require minimum data preprocessing and work on numerical data. It is especially benefical when the data is not well known and is highly dimensional. They require very few additional steps in case when it is hard or impossible to solve data imputation [10].

We can see that the AUC Score obtained below is similar Decision Tree Classifier and in general XG Boost is known perform better with high dimensional data.

```
C:\ProgramData\Anaconda2\lib\site-packages\sklearn\preprocessing\label.py:151:
DeprecationWarning: The truth value of an empty array is ambiguous. Returning F
alse, but in future this will result in an error. Use `array.size > 0` to check
that an array is not empty.
  if diff:
```

Out[35]:
```
0.9649784384279821
```

Refinement

Although the above results are enough to overcome the benchmark mode, there is still scope to improve the accuracy and I found out after tuning the parameters. We have used various refinement techniques before applying the data to the classifiers.

Linear Regression

For Linear Regression algorithm, there were not many parameters to tune but to provide randomness and shuffling the data so that we can analysis the sensitivity of the algorithm, it was used. Normalized the data, shuffled the data and evaluated the feature importances and sensitivity

closely. I can be observed that the score has increased by only 0.5% after tuning the parameters and there is not much that can be tuned with Linear Regression as-is.

Out[36]: 0.9230325542214819

Decision Tree Classification

For Decision Tree Classification, the minimum samples and the minimum split were two parameters that had to be adjusted based on the sensitivity analysis. There had to be a balance maintained between overfitting and covering enough scenarios. We can see that there is almost 3% increase in the accuracy from the implementation stage.

Out[37]: 0.9948640645200802

XGBoost

XGBoosting is renowned for classification while considering any high dimensional data. It requires very less preprocessing, unlike the above algorithms, it was able to achieve higher F1-score without the initial refinement and outperformed other algorithms without any senstivity analysis. Different learning rates were considered and various parameters were tuned to achieve slightly better performance prior to refinement.

We can see the there is almost 3% increase from what the results of the Implementation stage have been able to achieve.

```
C:\ProgramData\Anaconda2\lib\site-packages\sklearn\preprocessing\label.py:151:
DeprecationWarning: The truth value of an empty array is ambiguous. Returning F
alse, but in future this will result in an error. Use `array.size > 0` to check
that an array is not empty.
  if diff:
```

Out[38]: 0.990822714849618

IV. Results

Model Evaluation and Validation

Robustness using Stratified K Fold Cross Validation

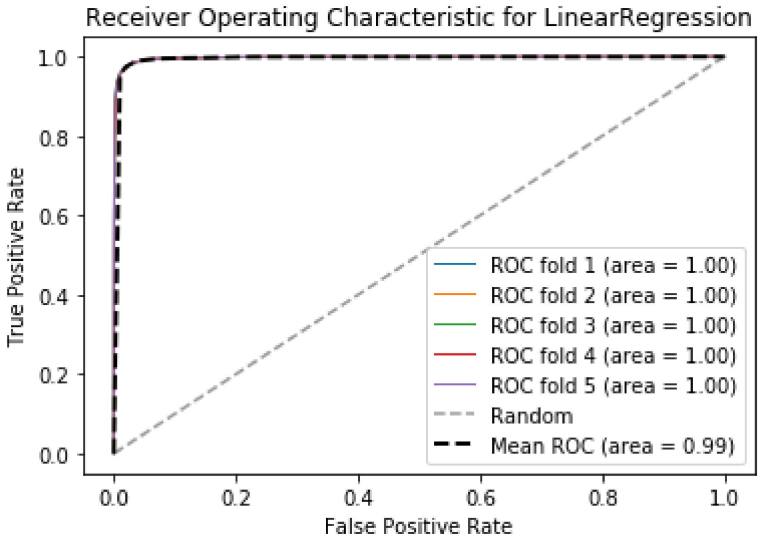
For analyzing the Robustness of the above models, cross validation analysis can be used with different folds and the AUC metric can evaluated for each fold and for each model. Cross validation is a technique for evaluating the robustness of predicting models. K fold cross validation is a type of cross validation technique. Using this technique, the complete data including the test data is divided into 'K' buckets, where 'k' is the number of folds or buckets the data needs to be divided. After dividing the data into 'k' buckets, each bucket is once considered as test data and the remaining 'k-1' buckets are used for training the models. This kind of analysis is usually used for small datasets

where splitting the data into test data can be avoided and the models can be trained and tested using complete data. But, for this problem, we can use this technique for evaluating the robustness of the models.

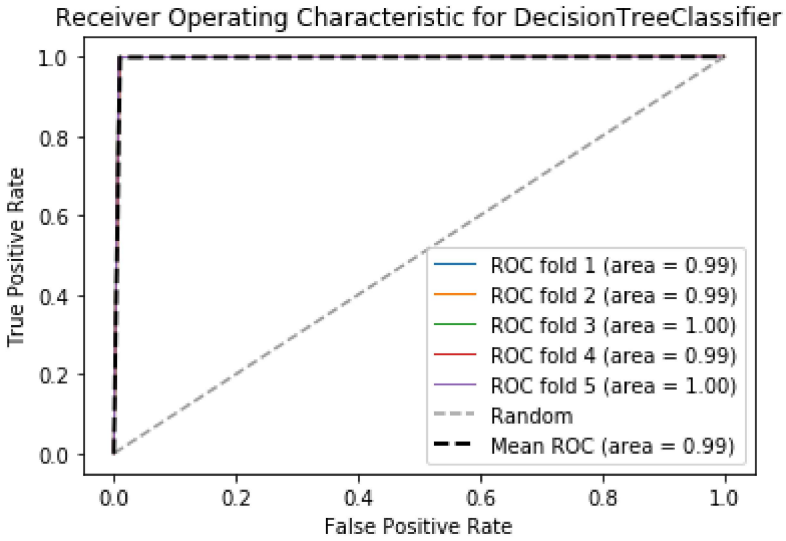
Stratified K fold is similar to K fold technique, but it maintains the percentage of samples from each class. Below are the graphs plotted for 5 fold analysis of the data using Stratified K fold cross validation. For more variation in the folds, I have also used the shuffle parameter to shuffle the data across the folds.

The results show that the data has been performing consistently for different Stratified 5 folds and with almost same AUC values, with a variation of 1%, which a pretty high standard.

Below is the ROC curve for 5 folds using Linear Regression model. The model has been pretty consistent throughout the 5 folds to indicate that it is robust.

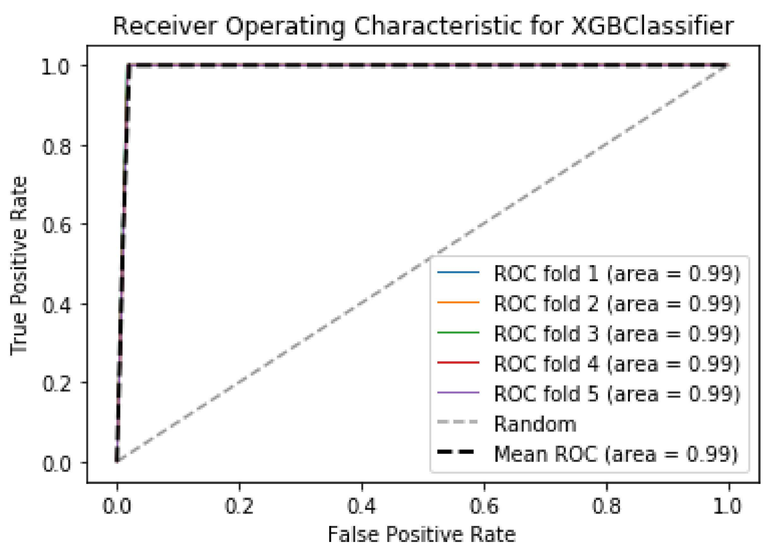


Below is the ROC curve for 5 folds using Decision Tree Classification model. The model has been pretty consistent throughout the 5 folds to indicate that it is robust. One of the drawbacks of Decision Tree Classification makes it is a clear victim during sensitivity analysis but the below results show that this model is robust.



Below is the ROC curve for 5 folds using XG Boost. The model has been pretty consistent throughout the 5 folds to indicate that it is robust. We expect well tuned XG Boost models to perform better than native algorithms and it is evident from the below curve the this model is robust.

```
C:\ProgramData\Anaconda2\lib\site-packages\sklearn\preprocessing\label.py:151:
DeprecationWarning: The truth value of an empty array is ambiguous. Returning F
alse, but in future this will result in an error. Use `array.size > 0` to check
that an array is not empty.
  if diff:
C:\ProgramData\Anaconda2\lib\site-packages\sklearn\preprocessing\label.py:151:
DeprecationWarning: The truth value of an empty array is ambiguous. Returning F
alse, but in future this will result in an error. Use `array.size > 0` to check
that an array is not empty.
  if diff:
C:\ProgramData\Anaconda2\lib\site-packages\sklearn\preprocessing\label.py:151:
DeprecationWarning: The truth value of an empty array is ambiguous. Returning F
alse, but in future this will result in an error. Use `array.size > 0` to check
that an array is not empty.
  if diff:
C:\ProgramData\Anaconda2\lib\site-packages\sklearn\preprocessing\label.py:151:
DeprecationWarning: The truth value of an empty array is ambiguous. Returning F
alse, but in future this will result in an error. Use `array.size > 0` to check
that an array is not empty.
  if diff:
C:\ProgramData\Anaconda2\lib\site-packages\sklearn\preprocessing\label.py:151:
DeprecationWarning: The truth value of an empty array is ambiguous. Returning F
alse, but in future this will result in an error. Use `array.size > 0` to check
that an array is not empty.
  if diff:
```



Confusion Matrix

Since this classification problem has 2 classes, it would be good to examine the number of false positive and false negative cases for the above algorithms. Below is the confusion matrix for predictions from Linear Regression algorithm. Looks like, the number of false positives and false negatives are pretty high for this algorithm.

Out[44]:
```
array([[ 28698,   5128],
       [   303, 129432]], dtype=int64)
```

Below is the confusion matrix for predictions from Decision Tree Classification algorithm. Looks like,

the number of false positives and false negatives are low compared to Linear Regression and we can expect a better AUC score for the predictions of this algorithm.

Out[45]:
```
array([[ 33520,    306],
       [   159, 129576]], dtype=int64)
```

Below is the confusion matrix for predictions from XGBoost algorithm. Looks like, the number of false positives and false negatives are low compared to Linear Regression but seemed to be close to Decision Tree Classification. We can expect a good AUC score for the predictions of this algorithm.

Out[46]:
```
array([[ 33220,    606],
       [    57, 129678]], dtype=int64)
```

Area under curve (AUC) metric

Below is the AUC score for Linear Regression Algorithm and it is already better than what has been proposed by authors in our bench mark model. Although the process that has been used and the dataset that has been considered has been preprocessed very differently, this forms a good score for this classification.

Out[47]:
```
0.9230325542214819
```

Below is the AUC score for Decision Tree Algorithm and it is better than what has been proposed by authors in our bench mark model and also by the Linear Regression Model.

Out[48]:
```
0.9948640645200802
```

Below is the AUC score for XG Boost algorithm and it is better than what has been proposed by authors in our bench mark model and also by the Linear Regression Model. It is almost as close as what Decision Tree Classifier has been able to achieve.

Out[49]:
```
0.990822714849618
```

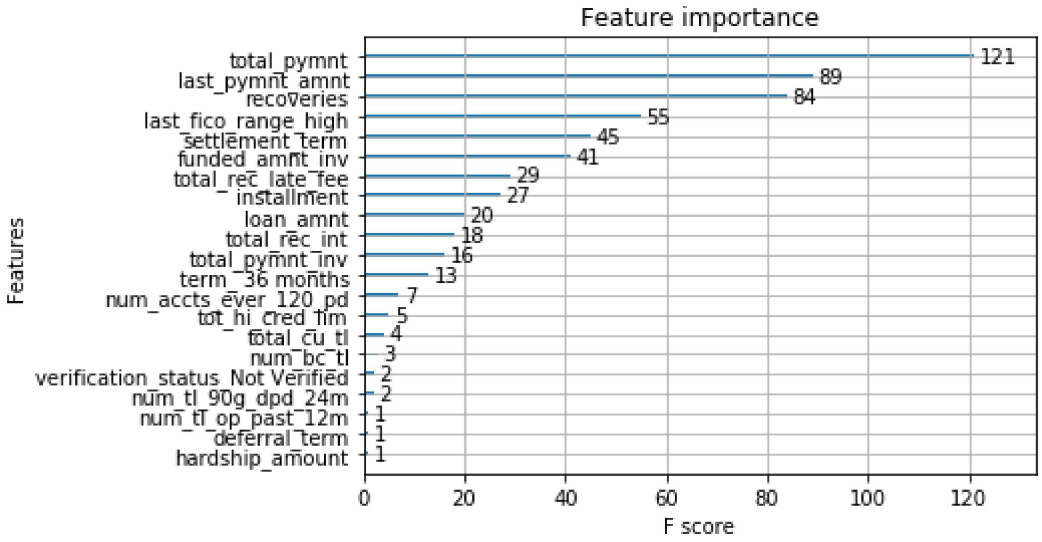
It can be seen from the above scores that the algorithms have produced really high F1 score in terms of prediciting the defaulted loans and the Good loans.

V. Conclusion

Let us consider the feature importances for the XG Boost algorithm to deduce the relation between FICO score and the loans that have been predicted. It would be vital to know if having high FICO scores as a criteria by Lending Club Inc., has helped the company or as stated by the previous articles, if it has been one of the key factors in identifying potential good customers who could pay off the loans.

Free-Form Visualization

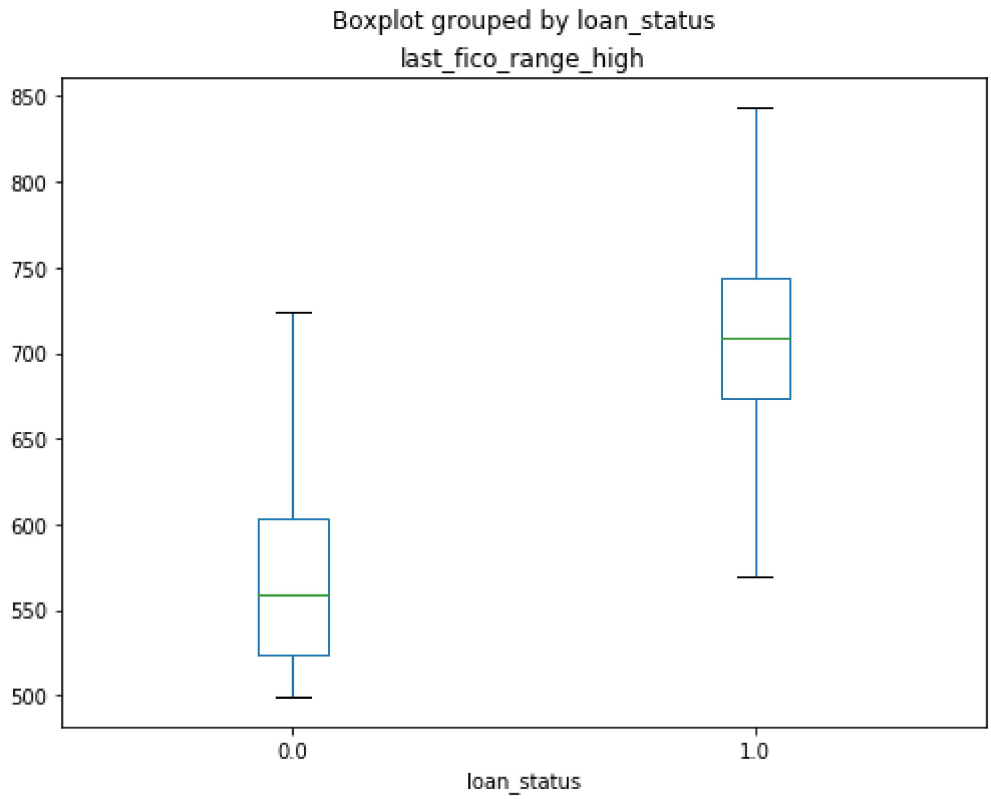
Out[50]: <matplotlib.axes._subplots.AxesSubplot at 0x551b8be0>



From the above graph of F score vs Features, it is clear that the ***FICO range high*** is clearly not the only criteria in classifying a loans into defaulted or otherwise. One should take into important considerations like the funded amount and the term taken to repay the loans as seen in the above graph in parallel with the FICO range high. Having said that FICO range high is one of the top 5 features in determining that a loan is defaulted or not.

Reflection

Out[51]: <matplotlib.axes._subplots.AxesSubplot at 0x617eee80>



Summary

Understanding the box plot

The above box plot, perfectly summarizes a general perspective of FICO values vs. the reality.

General Perspective

It can be seen that, in general 'last_fico_range_high' has been significantly varying in loans that have been completed payed off vs the ones that have been defaulted. The loans that have been defaulted with (X-value as 0) have much lower values compared to the loans that have been completely paid off (X-value as 1) in general. This is similar to the company's approach in insisting in high FICO score.

Introspection

The above criteria seems to be valid for only 50% of the test data that we have considered, which includes the area covered by the 2 boxes above. This constitutes the data for 2 quartiles of the data for the second quartile and the third quartile. However, that doesn't explain the complete picture. The last quartile of the data for the 'Defaulted Loans' has last_fico_range_high going but up as high as 700 which the median of the loans that are paid off. Similar to the first quartile data for 'Paid-off Loans' have FICO scores as low as 600. So, the company may have missed out on misclassifying the another 50% of the data.

From the over all analysis, although FICO score plays an important role in determining the loan defaulters, one should be using it along with other parameters like loan_amount and term etc.

Challenges

The dataset considered for this analysis is very tricky in terms of the features that can be considered for this binary classification problem. It is one of the reasons for the misinterpretation of data by the previous authors. As there are different features in the dataset such as 'settlement_amount' or 'settlement_percentage' that only apply to a specific class of the dataset, such features are exploited by the algorithms to determine the target variable, which is reality is only populated once the target variable is achieved.

Imbalanced data - One of the key challenges of this problem was dealing with imbalancing. This is a typical financial problem in the banking industry and it was necessary to classify both the classes with equal evaluation metric. The challenges faced here are general to an imbalanced problem, it was about accurately classifying both the classes equally, meaning one class had to be oversampled, then, I had to choose metric that are suitable for classifying both the classes.

Improvements

To perform actual analysis on this dataset specific to the binary classification of loans as considered, there needs a lot of domain knowledge. There are quite a few biased parameters that have been cleaned during the preprocessing stage in this analysis. But, there should still be a few out there and can be addressed only be addressed with in-depth domain knowledge and if the authors of the dataset can share more detailed information about the features.

Also, the data for this analysis does not take into considered of the applications that have been rejected by Lending Club Inc due to a lot of inconsistencies between the features between the two datasets. There could be further analysis done to determine if Lending Club Inc. has lost any valid customers based on the current analysis. This falls out of the scope of this project and has not been considered.

VI. References

[1] https://en.wikipedia.org/wiki/Lending_Club (https://en.wikipedia.org/wiki/Lending_Club)
[2] https://github.com/spiningup/udacity-ml-capstone/blob/master/capstone_report.ipynb (https://github.com/spiningup/udacity-ml-capstone/blob/master/capstone_report.ipynb)
[3] https://stats.stackexchange.com/questions/1292/what-is-the-weak-side-of-decision-trees (https://stats.stackexchange.com/questions/1292/what-is-the-weak-side-of-decision-trees)
[4] https://www.r-project.org/conferences/DSC-2003/Proceedings/Buehlmann.pdf (https://www.r-project.org/conferences/DSC-2003/Proceedings/Buehlmann.pdf)
[5] https://resources.lendingclub.com/LCDataDictionary.xlsx (https://resources.lendingclub.com/LCDataDictionary.xlsx)
[6] https://rstudio-pubs-static.s3.amazonaws.com/203258_d20c1a34bc094151a0a1e4f4180c5f6f.html (https://rstudio-pubs-static.s3.amazonaws.com/203258_d20c1a34bc094151a0a1e4f4180c5f6f.html)
[7] https://github.com/spiningup/udacity-ml-capstone/blob/master/capstone_report.ipynb (https://github.com/spiningup/udacity-ml-capstone/blob/master/capstone_report.ipynb)

[8] https://pandas.pydata.org/pandas-docs/stable/generated/pandas.get_dummies.html (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.get_dummies.html)
[9] http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Imputer.html (http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Imputer.html)
[10] http://danielhnyk.cz/how-to-use-xgboost-in-python/ (http://danielhnyk.cz/how-to-use-xgboost-in-python/)
[11] https://en.wikipedia.org/wiki/F1_score (https://en.wikipedia.org/wiki/F1_score)
[12] https://en.wikipedia.org/wiki/Precision_and_recall (https://en.wikipedia.org/wiki/Precision_and_recall)
[13] https://en.wikipedia.org/wiki/Linear_regression (https://en.wikipedia.org/wiki/Linear_regression)
[14] https://xgboost,readthedocs,io/en/latest/python/python_api,html#module-xgboost,sklearn (https://xgboost.readthedocs.io/en/latest/python/python_api.html#module-xgboost.sklearn)