Leeds University Business School

UNIVERSITY OF LEEDS

Student Name	PRAVEEN GOPAL REDDY
Module Code:	LUBS5990M
Module Title:	Machine learning in Practice
Module Leader:	Xingjie Wei
Declared Word Count:	2997

Please Note:

Your declared word count must be accurate, and should not mislead. Making a fraudulent statement concerning the work submitted for assessment could be considered academic malpractice and investigated as such. If the amount of work submitted is higher than that specified by the word limit or that declared on your word count, this may be reflected in the mark awarded and noted through individual feedback given to you.

It is not acceptable to present matters of substance, which should be included in the main body of the text, in the appendices ("appendix abuse"). It is not acceptable to attempt to hide words in graphs and diagrams; only text which is strictly necessary should be included in graphs and diagrams.

By submitting an assignment you confirm you have read and understood the University of Leeds **Declaration of Academic Integrity** (http://www.leeds.ac.uk/secretariat/documents/academic_integrity.pdf).

Predicting the Success of Initial Coin Offering Project

1 Introduction

In the following document, we will analyze a dataset of historical ICO (initial coin offering) projects fund raising. Our motivation is to build machine learning models that predict whether ICO project campaigns will raise fund successful or not. Initial Coin Offerings (ICOs) have become an increasingly popular way to raise capital for blockchain technology startups. In an ICO, entrepreneurs raise money for their venture by selling newly created cryptocurrency tokens to investors in exchange for fiat currency such as US dollars or cryptocurrencies such as Ethereum or Bitcoin. The cryptocurrency tokens typically act as a digital medium of exchange to access the firm's digital platform and services. After completing the ICO, the tokens may be traded on an online exchange and increase in value with the success of the project.

We analyze the determinants of success for 1606 ICOs undertaken from August 2015 up until the end of December 2018, a period in which the market for ICOs grew to an unprecedented level. Using supervised machine learning models such as random forest, logistic regression, Decision tree and XGBoost. We find common important features from the prediction results of these models. It is observed that social media, team size, overall rating, token number, accepting currency ETH and BTC are successful factors in raising funding. The features of high importance are related to media presence.

The remainder of the paper proceeds as follows. In Section 2, we present Data understanding. Section 3 discusses the Data preparation to get it ready for modelling. In Section 4, we will perform Modelling. Section 5 contains Evaluation and finally we will end our document with Conclusion.

2 Data Understanding

In this topic we will do exploratory data analysis of ICO dataset.

The ICO dataset csv file contains a total of 1606 records with 20 columns. The column names with order are:

Column 1-5	Column 6-10	Column 11-15	Column 16-20
id	categories	tokenName	softcap
success	overallrating	tokenPrice	hardcap
tokenNum	offered_ownership	tokenType	whitepaper
teamSize	enddate	platform	video
country	startdate	acceptingCurrency	socialMedia

2.1 Statistical summary of ICO dataset.

Before we proceed with the visualizations, let us first see a statistical summary of our dataset:

	success	tokenNum	teamSize	overallrating	offered_ownership	softcap	hardcap	whitepaper	video	socialMedia
count	1606.00	1359.00	1606.00	1606.00	1065.00	1577.00	1577.00	1577.00	1577.00	1577.00
mean	0.55	20152680654366.00	11.17	2.85	1.14	0.44	0.69	0.95	0.67	0.64
std	0.50	626466565998403.00	8.04	0.82	18.41	0.50	0.46	0.22	0.47	1.13
min	0.00	0.00	0.00	0.80	0.01	0.00	0.00	0.00	0.00	0.00
25%	0.00	7000000.00	6.00	2.30	0.45	0.00	0.00	1.00	0.00	0.00
50%	1.00	55000000.00	10.00	2.80	0.60	0.00	1.00	1.00	1.00	0.00
75%	1.00	251480400.00	15.00	3.50	0.74	1.00	1.00	1.00	1.00	1.00
max	1.00	22600000000000000.00	67.00	4.80	601.25	1.00	1.00	1.00	1.00	3.00

Figure 1: Summary of numerical columns.

- From above figure we can see some columns count is 1606 and some are different. Those are different to 1606 count potentially these columns have some null values.
- In offered ownership, mean value is too smaller than max value so there can be possibility of presence of outlier and max value of token number is too big to accept as realistic value.
- Majoirty of white paper ad hard cap records has nominal value of 1 and success column distribution of data between 0 and 1 is in ration of 45:55%

	country	categories	tokenName	tokenType	platform	acceptingCurrency
count	1606	1606	1574	1148	1475	1379
unique	117	696	1492	43	63	234
top	USA	Cryptocurrency	SMT	ERC20	Ethereum	ETH
freq	261	97	4	1061	1307	722

Figure 2: Summary for categorical columns

From Figure 2, most of the ICO projects are USA based with 261 as total. Ethereum is widely used platform and most accepting currency for these projects is ETH.

2.2 Graphical summary of ICO dataset.

Let us build visualization report for important variables.

2.2.1 Distribution of important numerical data.

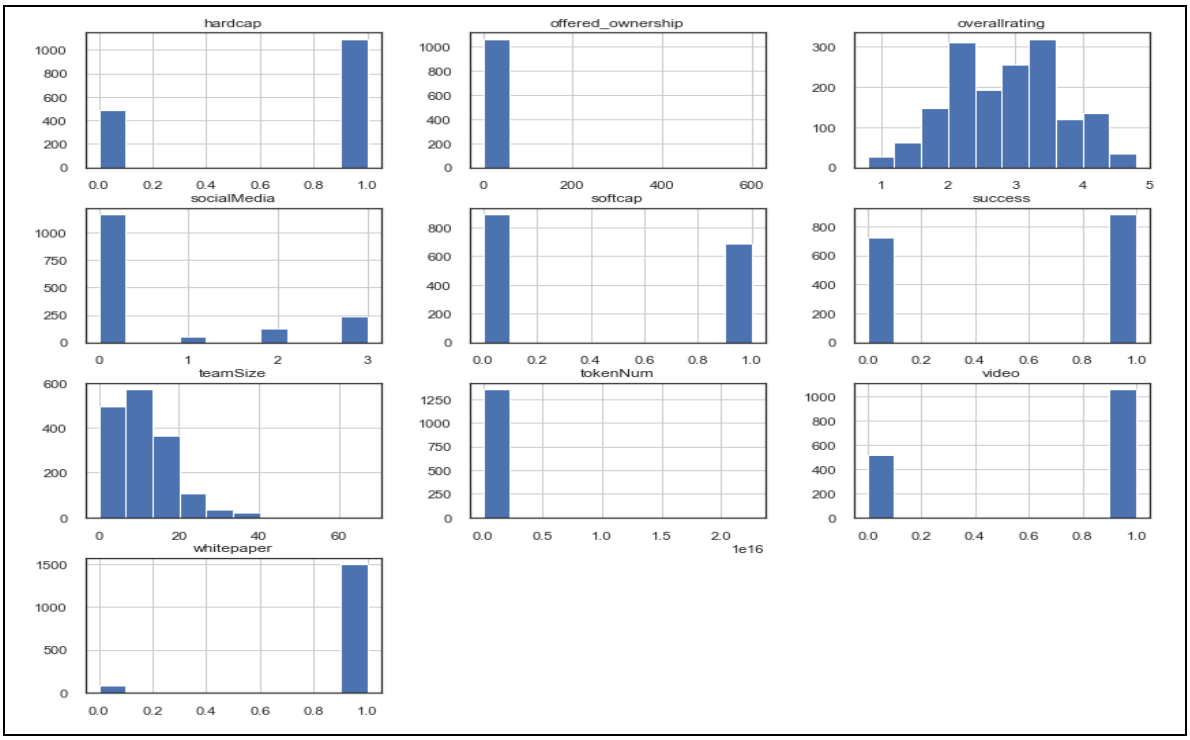


Figure 3: distribution of numerical data

From figure 3 we have following observations:

- more than 1400 rows for whitepaper variable have value 1
- maximum number of social Media ratings are zero
- success variable has more than 700 rows with zero value and more than 800 records with 1 value
- the shape of overall rating looks like normally distributed
- team size data is left skewed

2.2.2 Distribution for categorical data.

The below plot shows number of ICO projects by country.

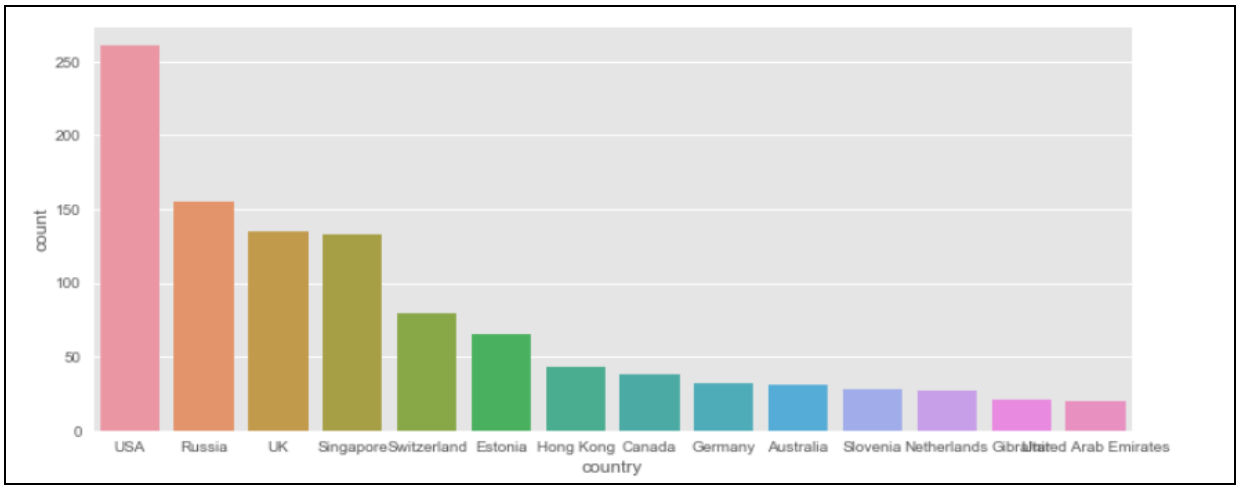


Figure 4: Top Countries with ICO projects

From Figure 4, clearly you can see USA has highest number of ICO projects with more than 250 projects, where as Russia, UK and Singapore are in 2nd,3rd and 4th place respectively.

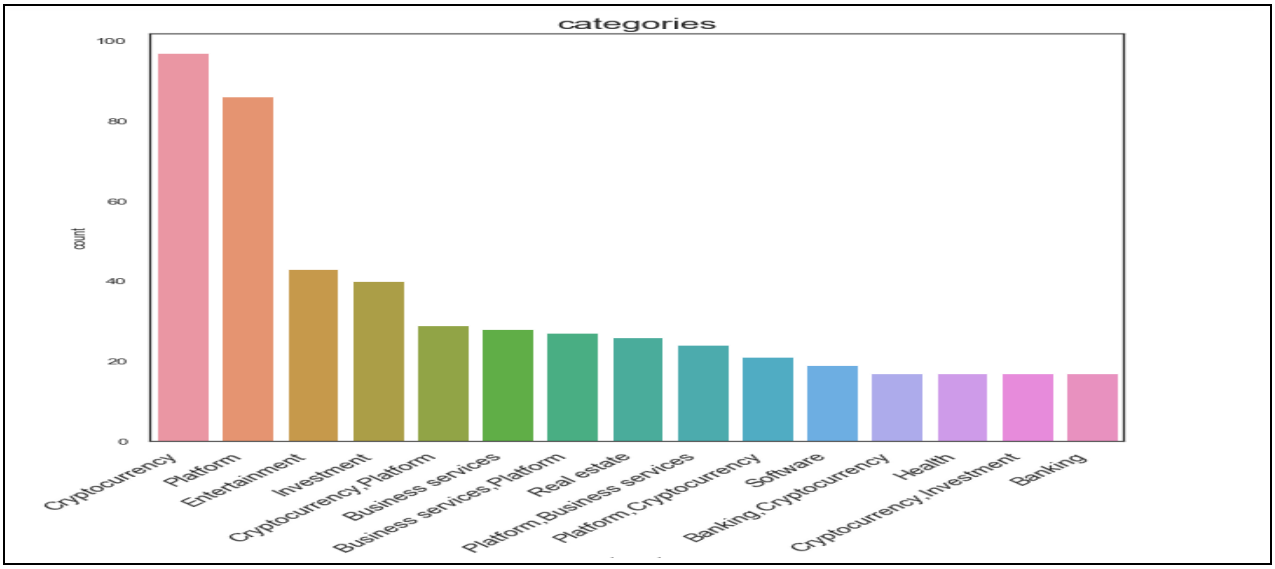


Figure 5: Top Categories data

From above plot it is evident that cyrpocrrency occupies top spot for ICO start up.

3 Data Preparation

In this topic we will prepare our final dataset to be ready for modelling after analysing realtionship between predictors. We will also look at adding new columns and dropping columns based on how strong the influence between dependent variable and predictors.

3.1 Imputation

The below Snapshot shows missing values in dataset for each variable.

```
print(df_org.isnull().sum())
```

```
success                0
tokenNum             247
teamSize               0
country                0
categories             0
overallrating          0
offered_ownership    541
enddate                0
startdate              5
tokenName             32
tokenPrice           182
tokenType            458
platform             131
acceptingCurrency    227
softcap               29
hardcap               29
whitepaper            29
video                 29
socialMedia           29
dtype: int64
```

Figure 6: Missing values in ICO dataset

- The missing values in offered ownership is replaced with column median value.
- The missing values in token number is replaced with median value based on country wise.
- Token number with zero values are removed from dataset.
- The missing values in soft cap,hard cap, white paper,video and social media is replaced by most frequent value of the corresponding variable.
- The missing values in platform is assigned based on token type. For example if token type is ETH/ERC20 etc then we assign platform = Ethereum
- The missing values in accepting currency is assigned either ETH or BTC etc based on platform variable.

3.2 Adding new variables

I. **Duration** :

we added duration variable to the dataset by taking difference between end data and start date

II. **Total fund:**

- we converted each token price variable in terms of USD value by using CoinGeckoAPI and yfinance API in python. for example, 1 ETH = 8,000 WPR, we first convert how many ETH in one WPR i.e., 1 WPR=1/8000 = 0.000125 ETH and then we use above APIs to find 1 ETH is how much in USD price.
- After that we multiplied token price with token number and assigned corresponding value to total fund variable.
- For token price which are null, we assigned mean value of token prize based on the country that it belongs.

III. **Start day**: from start date column we took the day of the week like Monday, Tuesday etc. It is believed that ICO projects success also depends on day of the week when it kick starts.

IV. **Start month**: from start date column we took the month like January, February etc.

3.3 Outliers

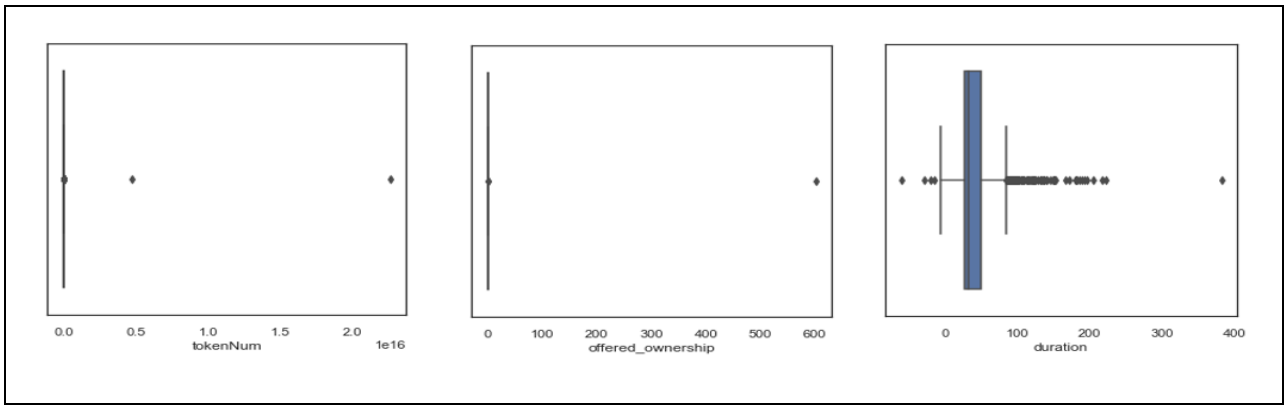


Figure 7 : Outliers

As you see there are some outliers in token number, duration and offered ownership variables. We replaced outliers with median value except for token number. For token number we got the actual correct value from wesbite and manaully updated.

3.4 Visuliaztion after imputation

Lets check the visual representation for some important continuous variables after our imputation and adding new variables.

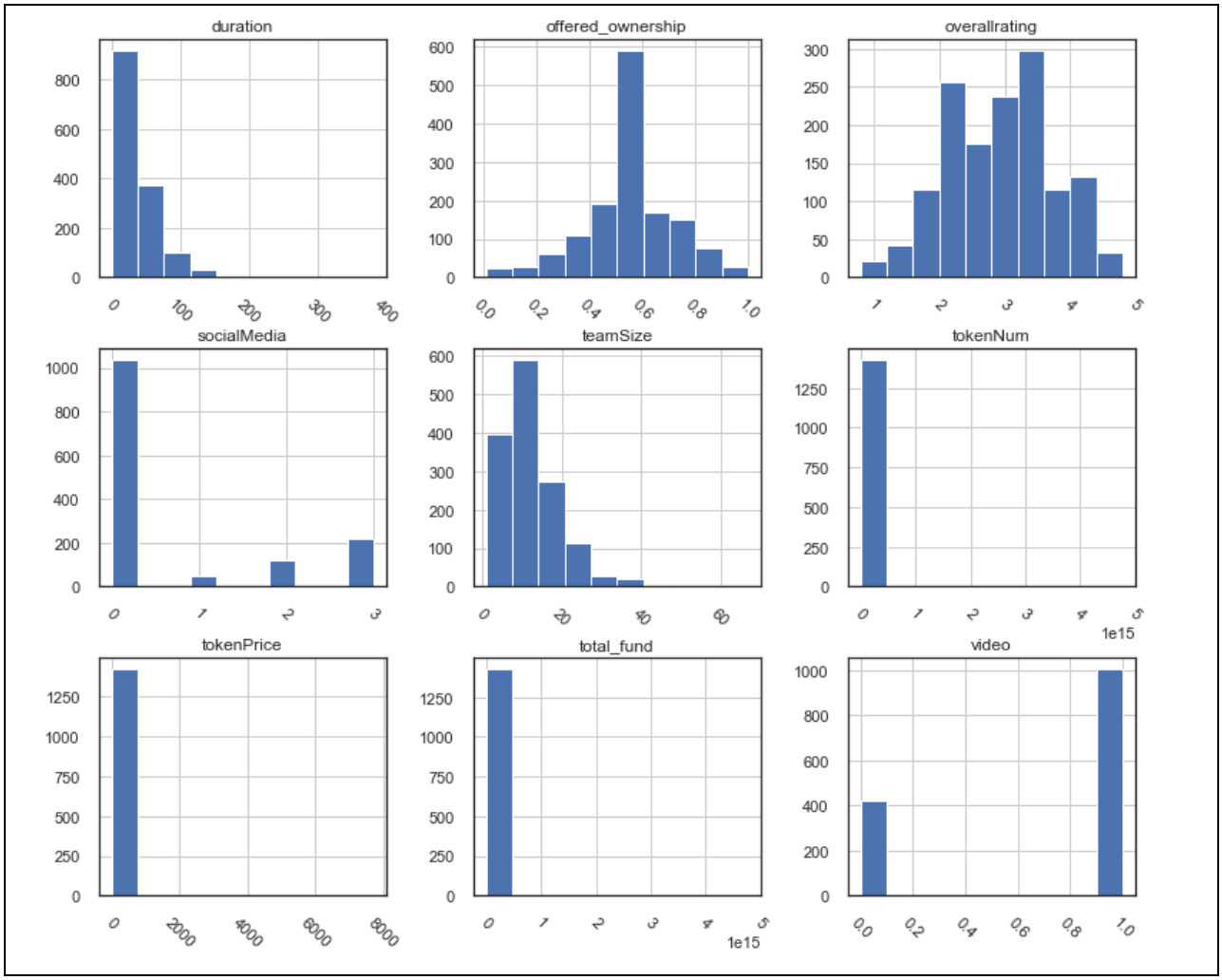


Figure 8 : Visualization after Imputation

1. For duration column, the plot is exponentially distributed, as duration increase number of projects reduce, it clearly shows investors show less interest when duration is above 50 days
2. offered ownership and overall rating data is spread out
3. team size is right skewed

3.5 Feature Selection

1. token name is not an important variable because all are distinct, just like projects having its own name.

2. majority of token type and platform values are uniform so we will drop token type and platform.
3. Start and end date is also not necessary since they are distinct for each record. We will drop these two variables.
4. Total fund has strong positive relation with token number. We will drop total fund.

3.6 Preparing data for machine learning.

- Drop features as stated in feature selection and create new data frame
- Create dummy variables for categorical data
- Separate the dependent (y) and independent (X) features into separate datasets.
- Since the features are on different scales, independent features will be transformed and normalized using StandardScaler in-built function from sklearn kit which removes the mean and scales each feature/variable to unit variance.
- we have used 70/30 train-test data split.

4 Modelling

In this section, four different machine learning models for classification will be applied to the data. Since it is a binary classification problem, where given the set of features, we need to predict if a given project will be successfully funded or not

We have have used Logistic regression, Random Forest, XGBoost and Decision Tree. The reasons for choosing these models will be explained below:

Logistic regression:

- since our target variable is binary output. Logistic regression is an extremely robust and flexible method for dichotomous classification prediction.
- It is useful when the response variable is binary but the explanatory variables are continuous.
- Besides logistic regression is less prone to overfitting and gradient descent typically works very fast and thus makes the training phase of logistic regression quick.

Random Forest:

- We choose this model with regards to accuracy. random forest algorithm is trained through bagging. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms
- We also don't want to use many configurations for modelling. So random forest generates good predictions without requiring many configurations in packages like skikit-learn.
- It can produce a reasonable prediction without hyper-parameter tuning.
- Some of our variables still have some missing data, it provides an effective way of handling missing data

XGBoost:

- We choose this model for many reasons, it is best model for accuracy, can reduce overfitting and fast.

- Since our dataset is structured, XGBoost has won more competitions in the structured data category. It does good job when data is structured and can give good predictions with small dataset also.
- Can Handle missing values using sparse approach and has inbuilt cross validation.
- It is called gradient boosting since, when introducing new models, it uses a gradient descent algorithm to minimize the loss.
- It adds Regularization to avoid overfitting.
- It can do Parallel creation of trees and Out-of-core computing.

Decision Tree:

- We choose this model because in Decision Tree**,** no feature scaling (standardization and normalization) required as it uses rule-based approach instead of distance calculation.
- Training period is less as compared to Random Forest because it generates only one tree unlike forest of trees in the Random Forest
- Decision Tree is usually robust to outliers and can handle them automatically.

4.1 Correlation between predictors

Let's look at the relationships between numeric features and other numeric features.

- Correlation is a value between -1 and 1 that represents how closely values for two separate features move in unison.
- Correlations near -1 or 1 indicate a strong relationship.
- Those closer to 0 indicate a weak relationship.
- 0 indicates no relationship.

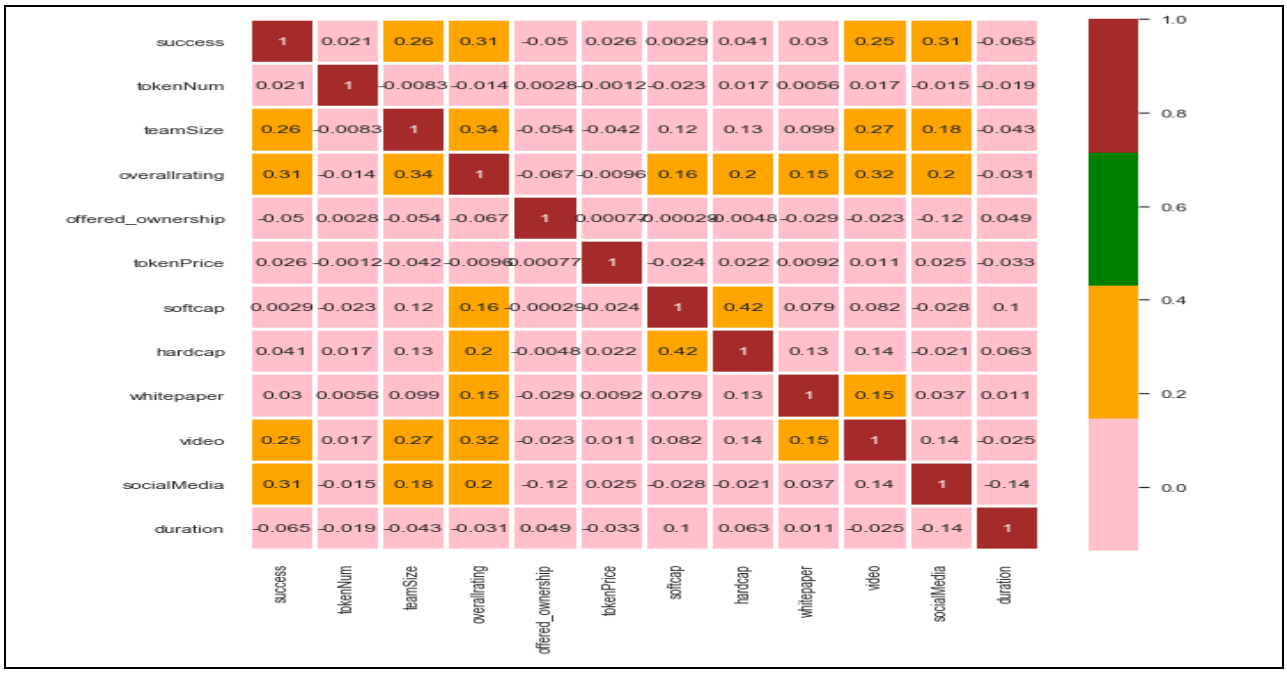


Figure 9: Correlation of numerical data

- From correlation figure we see most of numerical variable are weakly correlated. Some decent postive relation we can see in softcap vs hardcap, video vs overallrating ,teamzsize vs overallrating, success vs overallrating.
- Multi-collinearity will be checked for by assessing correlations between predictor features, as this can cause issues with some models. The multi-collinearity matrix above shows that this is not an issue

4.2 Plotting between predictors and output variable.

Below plot shows median predictors over output(success) variable.

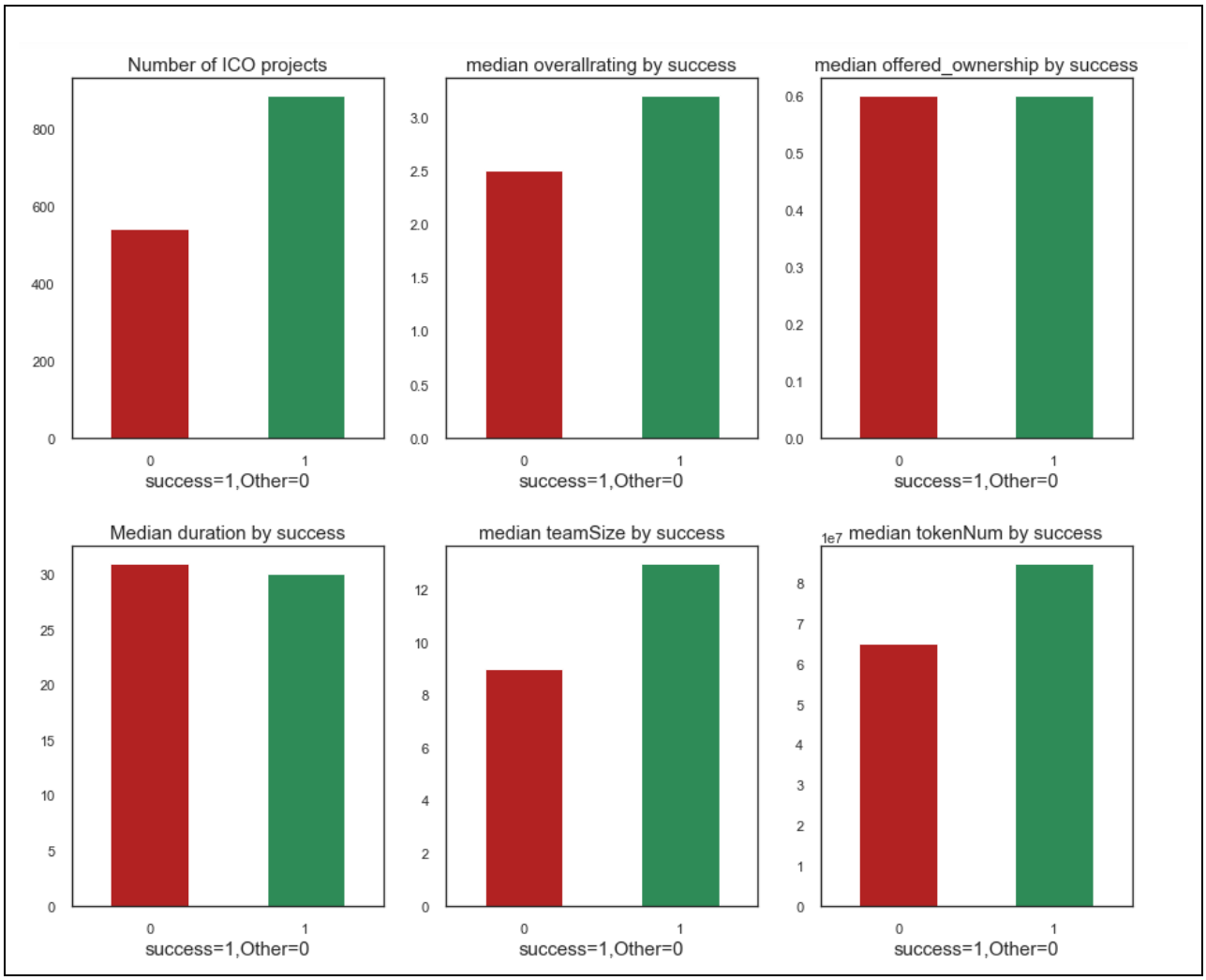


Figure 10: median numerical data over success variable

- For median overall rating , most of the projects are successfully funded if the average overall rating is above 3. Otherswise they are classified as not successful or due to other reasons.
- Intrestingly median offered ownership have equal chances of projects that either can be successfully funded or not funded.

- For medial token number majority of projects are classified success if token num is greater than 0.8*1e8(in scientific notaion) as seen in figure 10.
- Similarly if median team size is above 12 and duration of the project is below 30 days there are high chances that project will be funded.

4.3 Build Models

4.3.1 Logistic Regression

Default Model :

At first we have built logistic regression with default parameters

Build Optimized model with PCA:

There are many features in the dataset after when we created dummies for categorical data. So PCA (Principal Component Analysis) can be used to reduce this into a smaller number of components which still explain as much variation in the data as possible

The best score we got for 596 componets. So we will use this componet number in pipeline. And build the model using best parameters with help of GridSearchCV.

Same code of the model :

```
pipe_best_logreg = Pipeline([('pca', PCA(n_components=596)),
                             ('clf', LogisticRegression(C=0.01, fit_intercept=True, penalty='l2'))])
pipe_best_logreg.fit(X_train, y_train)
```

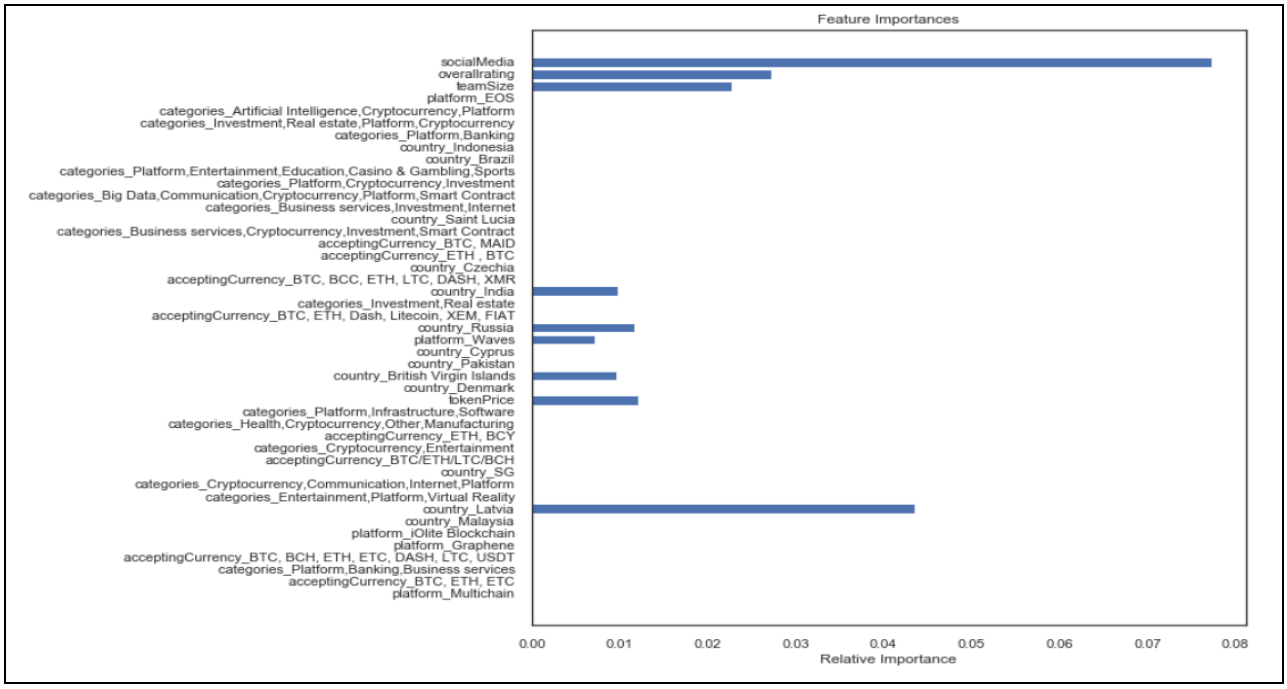


Figure 11: Important features from logistic default model.

As you see from figure 11, social media ,overall rating , team size ,token prize , country lavtia and russia are important features.

4.3.2 Random Forest.

We have used GridSearchCV with help of 5-fold cross validation to select the best parameters from listed hyperparameters. After running GridSearchCV , the best parameters we got are max_depth=35 and n_estimators =400. We then built the model using above parameters.

Sample code of the model:

```
best_rf = RandomForestClassifier(max_depth=35, min_samples_split=0.001, n_estimators=400)

best_rf.fit(X_train, y_train)
```

Below figure shows important features for random forest model . Overall rating, team size, duration, token prize and social media are top 5 important features.

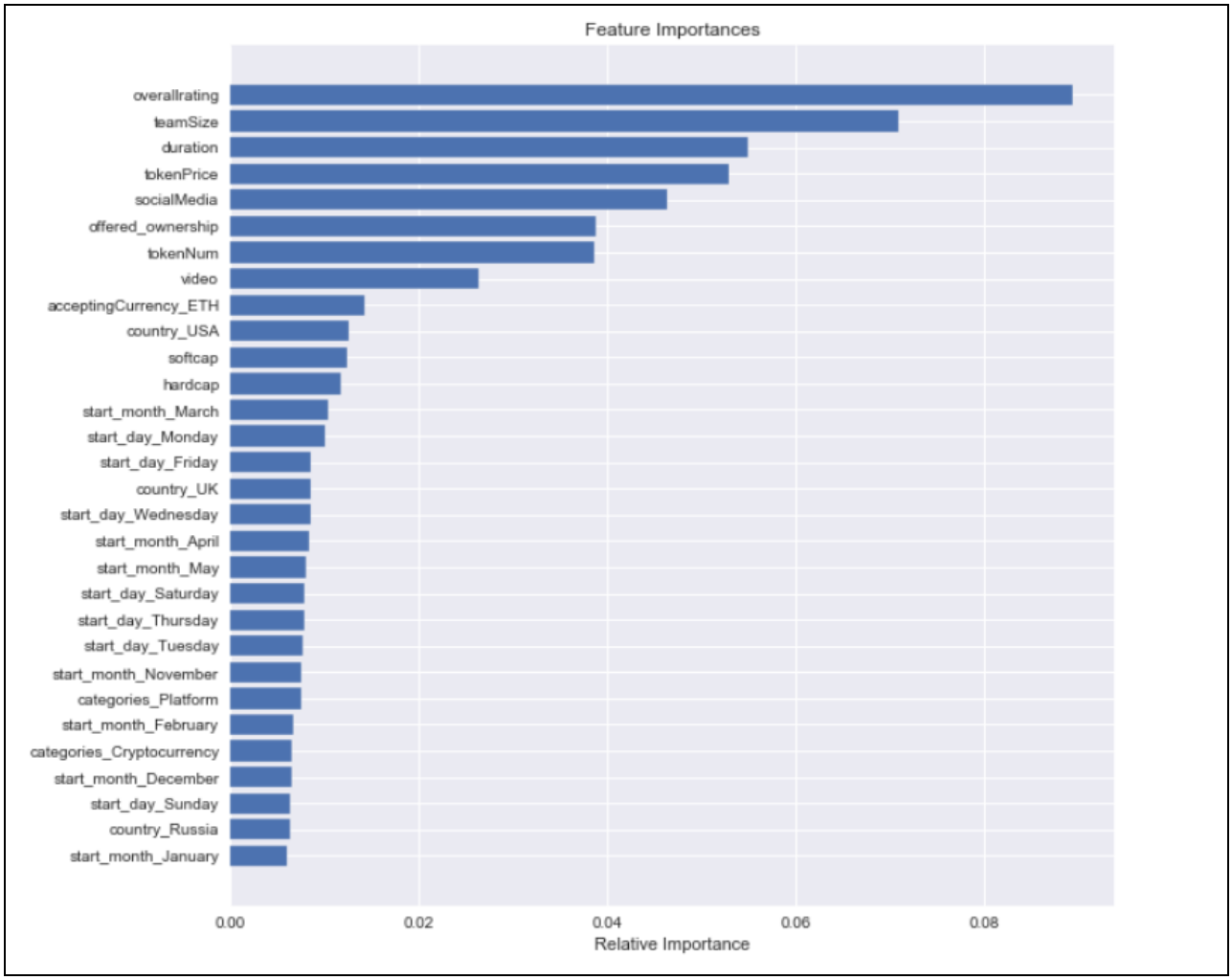


Figure 12: Important features from random forest model.

4.3.3 XGBoost

We have used XGBoost with default parameters. We also observed that optimizing using hyperparameters gave bad result. So we will stick to default parameters.

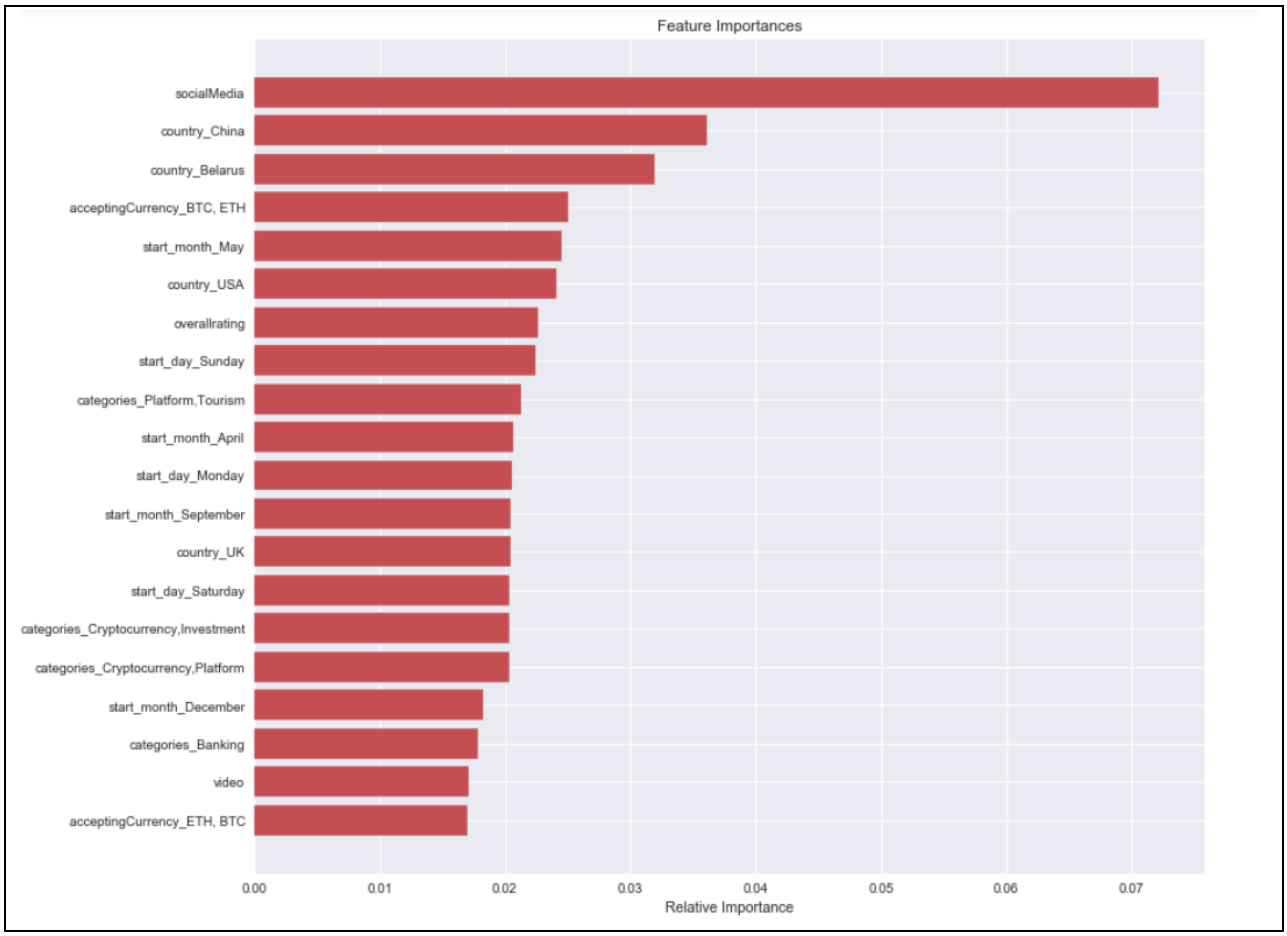


Figure 13: Important features from XGBoost model.

4.3.4 Decsion Tree

We have used GridSearchCV with help of 5-fold cross validation to select the best parameters from listed hyperparameters. After running GridSearchCV , the best parameters we got are max_depth=6, min_samples_leaf=2 and min_samples_split=7. We then built the model using above parameters.

From Figure 14 we can see that overall rating, social media, team size, token number and duration are top 5 important features for random forest model.

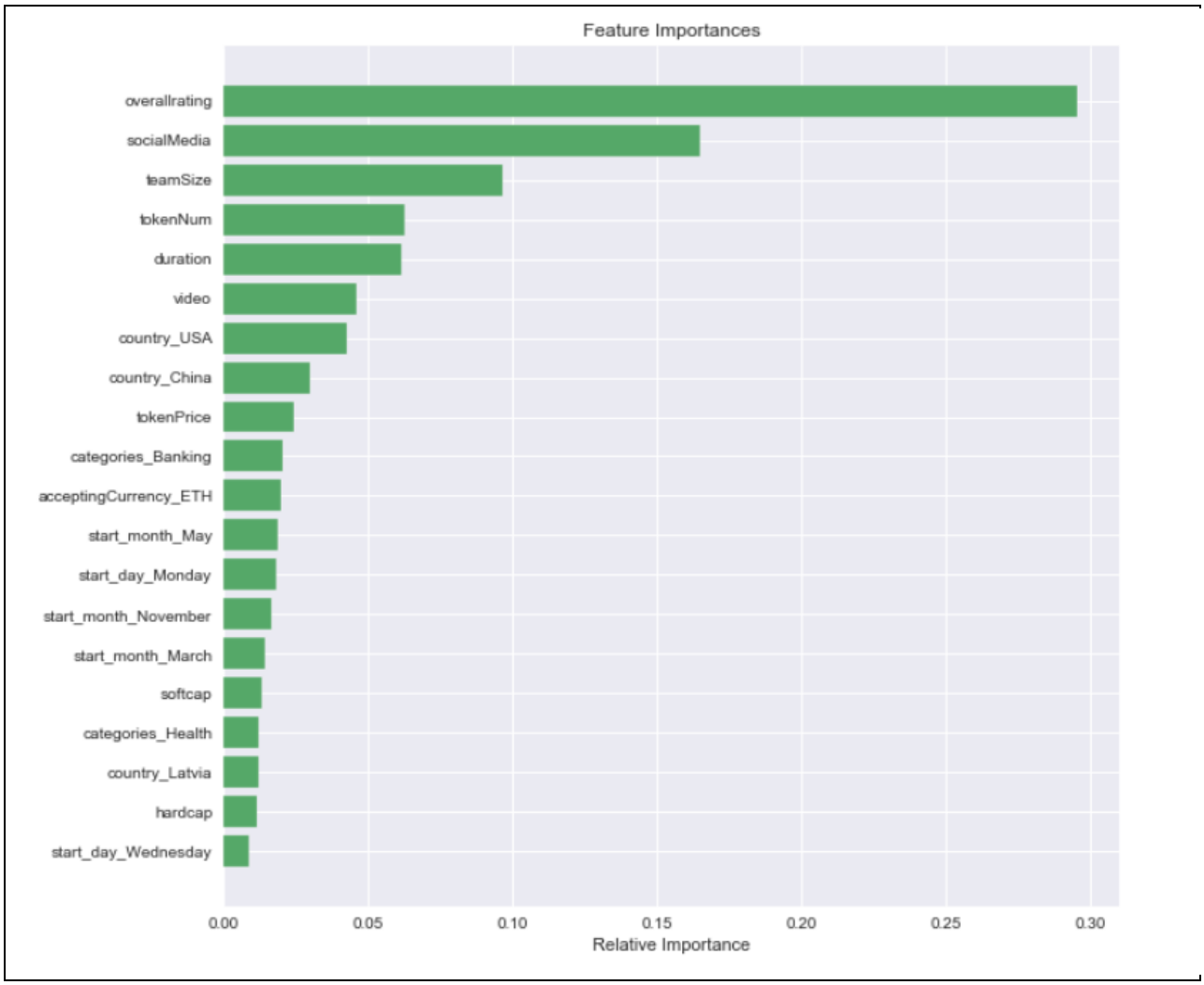


Figure 14: Important features from Decision Tree model.

5 Evaluation

Since this is binary classification problem, we use the following metrics:

- **weighted average F1** - The F1 score calculates the harmonic mean between precision and recall and is a suitable measure because there is no preference for false positives or false negatives in this case.The weighted average will be used because the classes are of slightly different sizes, and we want to be able to predict both successes and failures.
- **Confusion matrix** - For getting a better clarity of the number of correct/incorrect predictions by the model
- **ROC-AUC** - It considers the rank of the output probabilities and intuitively measures the likelihood that model can distinguish between a positive point and a negative point. (**Note:** ROC-AUC is typically used for binary classification only). We will use AUC to select the best model.

5.1 Evaluation of Logistic Regression model

5.1.1 Using default parameter model

We will evaualte using standard model which is explained in 4.3.1 topic.

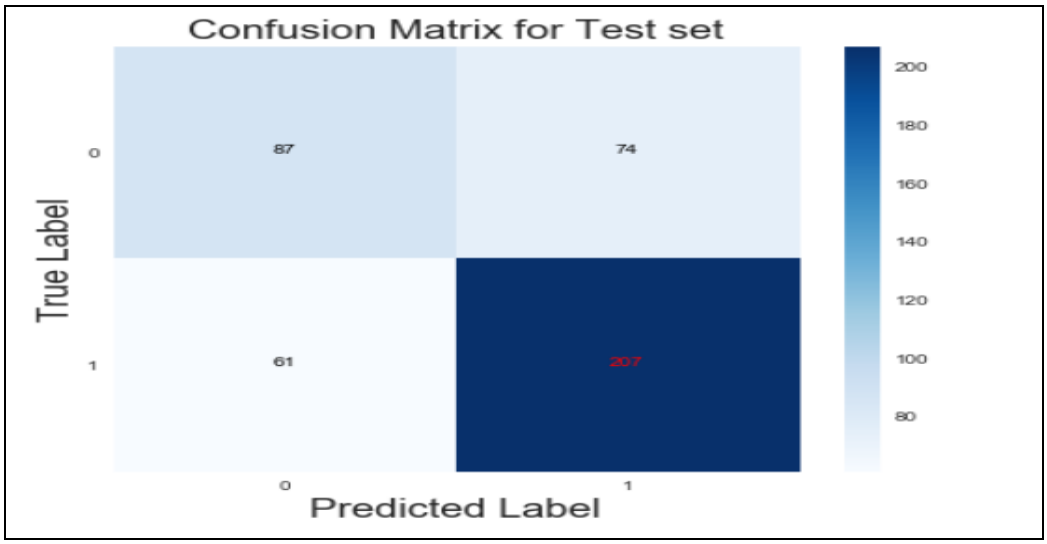
i. Classification report of Training set

	precision	recall	f1-score	support
0	0.91	0.90	0.91	381
1	0.94	0.95	0.94	617
accuracy			0.93	998
macro avg	0.93	0.92	0.92	998
weighted avg	0.93	0.93	0.93	998

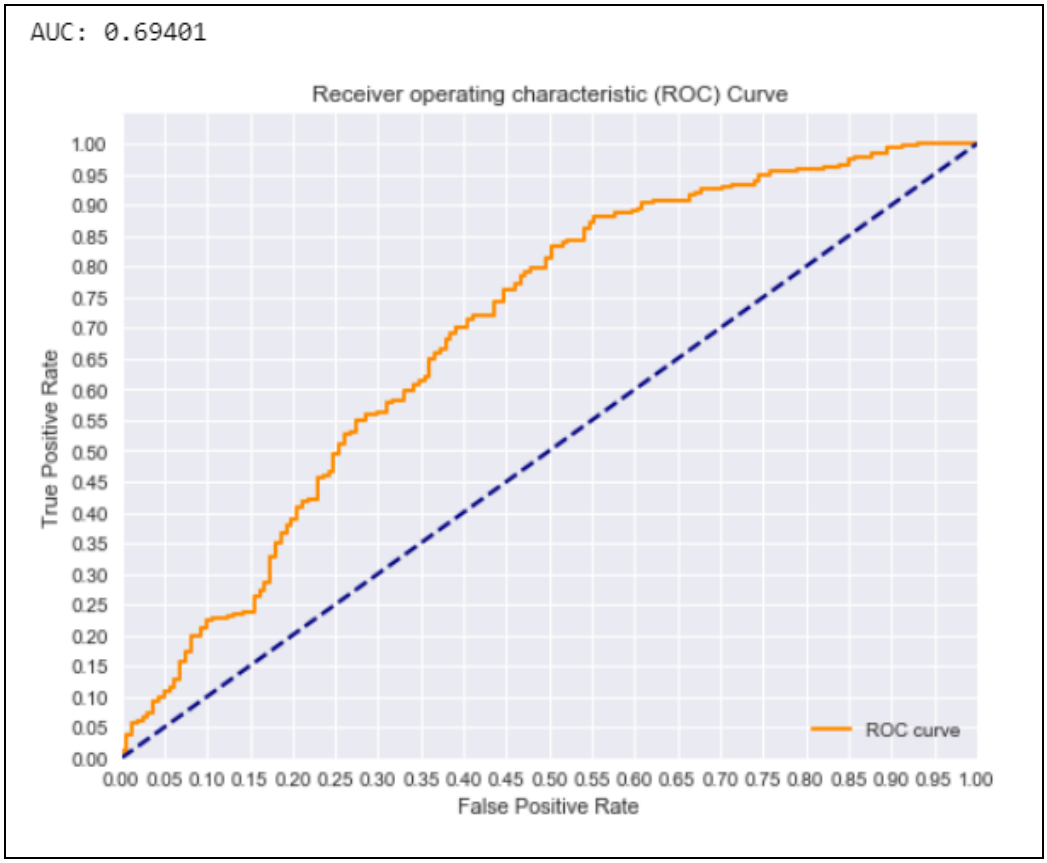
ii. Classification report of Test set

	precision	recall	f1-score	support
0	0.59	0.54	0.56	161
1	0.74	0.77	0.75	268
accuracy			0.69	429
macro avg	0.66	0.66	0.66	429
weighted avg	0.68	0.69	0.68	429

iii. Confusion Matrix for Test set



iv. ROC curve for Test set



The logistic regression model has a good accuracy score of 0.93 on training set (weighted average F1 score) and fair score of 0.68 on test set. However, it is worse at predicting failures compared to successes, and the recall rate (ability to correctly predict positives out of all the actual positives in the data) is notably different between the failure and success categories. The AUC value is decent, and the curve is pulled towards the top left of the graph, which is a positive sign. However, this can probably be improved upon.

5.1.2 Using PCA(optimized)

We will evaualte using the model optimized with PCA and hyper parameters which is explained in 4.3.1 topic.

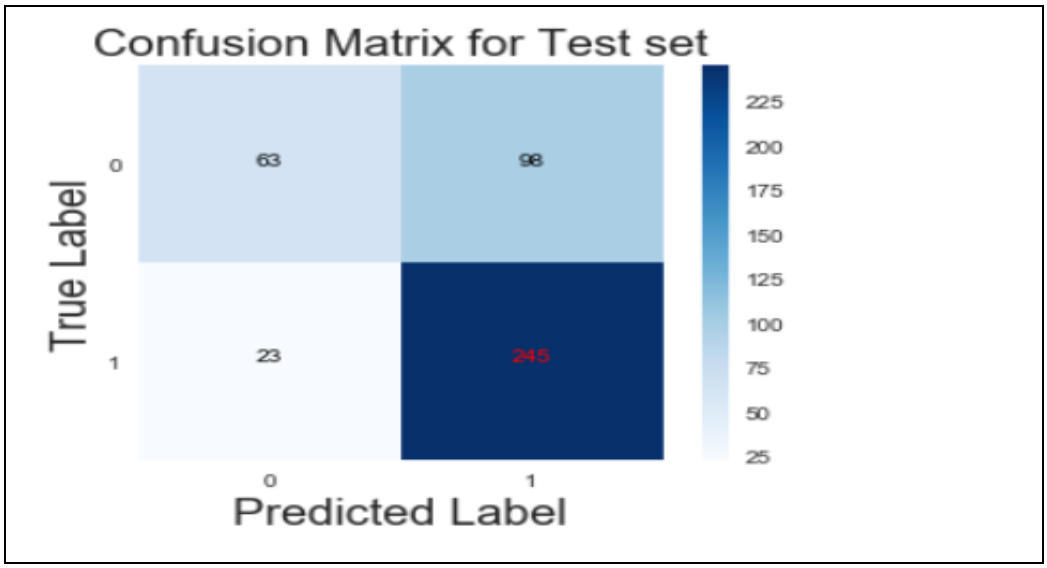
i. Classification Report on Train set

	precision	recall	f1-score	support
0	0.90	0.80	0.85	381
1	0.88	0.95	0.91	617
accuracy			0.89	998
macro avg	0.89	0.87	0.88	998
weighted avg	0.89	0.89	0.89	998

ii. Classification Report on Test set

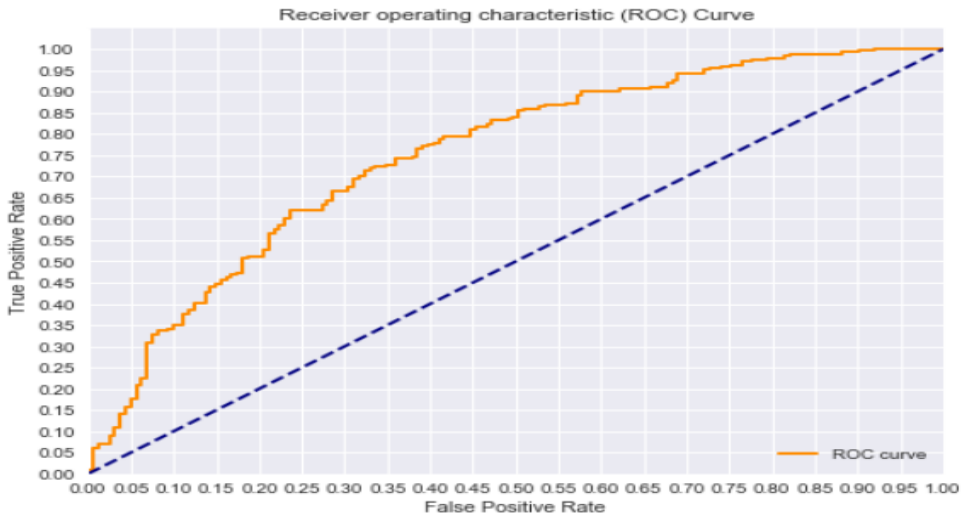
	precision	recall	f1-score	support
0	0.73	0.39	0.51	161
1	0.71	0.91	0.80	268
accuracy			0.72	429
macro avg	0.72	0.65	0.66	429
weighted avg	0.72	0.72	0.69	429

iii. Confusion matrix on Test set



iv. ROC curve for Test set

AUC: 0.75



As you see The AUC value is increased by 6%, and the curve area also improved. The weighted accuracy is increased by 1% and overall accuracy is increased by 3%.

5.2 Evaluation of Random Forest

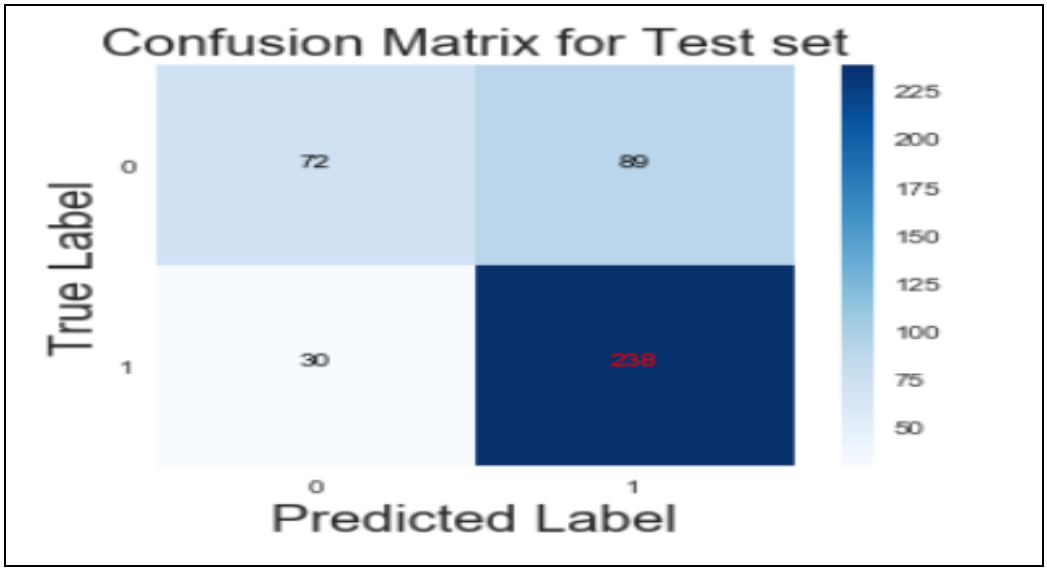
i. Classification Report on Training set

	precision	recall	f1-score	support
0	1.00	1.00	1.00	381
1	1.00	1.00	1.00	617
accuracy			1.00	998
macro avg	1.00	1.00	1.00	998
weighted avg	1.00	1.00	1.00	998

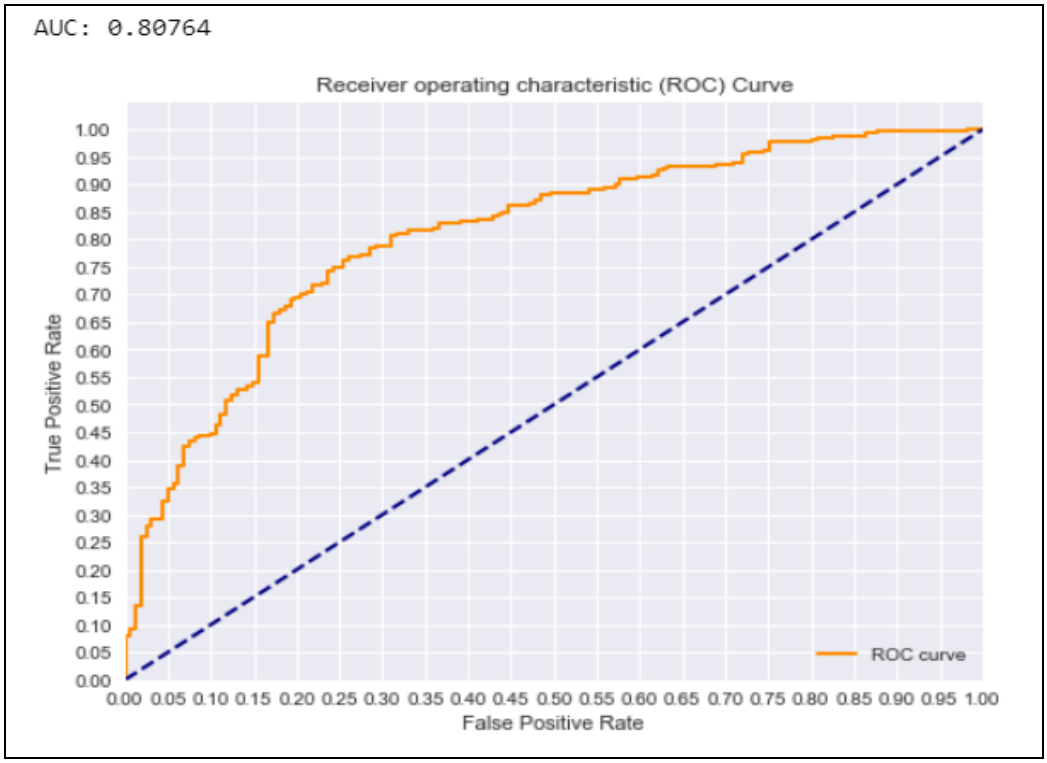
ii. Classification Report on Test set

	precision	recall	f1-score	support
0	0.72	0.46	0.56	161
1	0.73	0.89	0.80	268
accuracy			0.73	429
macro avg	0.73	0.68	0.68	429
weighted avg	0.73	0.73	0.71	429

iii. Confusion Matrix for Test set



iv. ROC curve for Test set



The train set accuracy is 100% and test set weighted accuracy is 71%. The big difference between train set and test accuracy is possibly due to overfitting. The AUC value for test set is pretty good with 80%, which tells us that the model performing good at distinguishing between the positive and negative classes

5.3 Evaluation of XGBoost Model

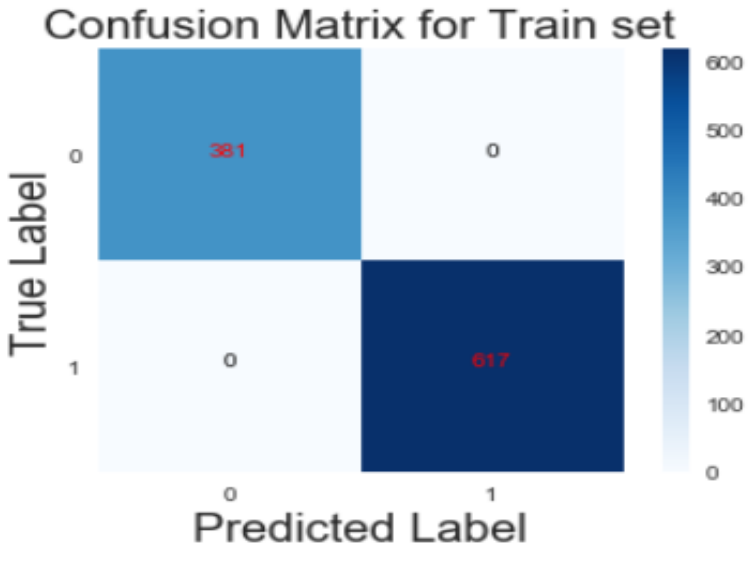
i. Classification Report on Training set

	precision	recall	f1-score	support
0	1.00	1.00	1.00	381
1	1.00	1.00	1.00	617
accuracy			1.00	998
macro avg	1.00	1.00	1.00	998
weighted avg	1.00	1.00	1.00	998

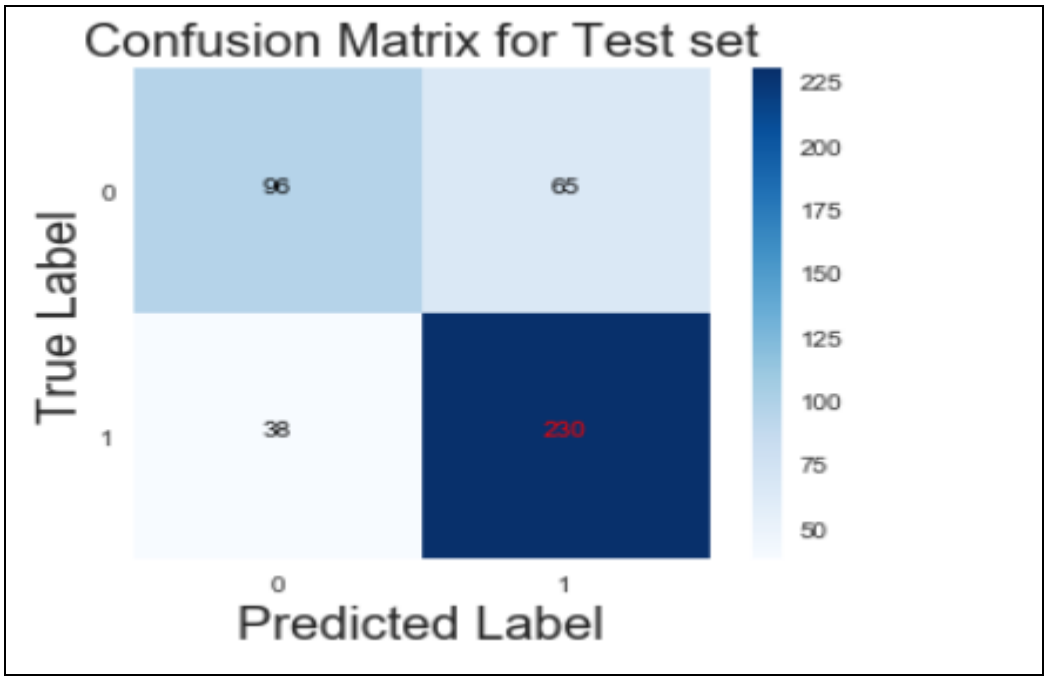
ii. Classification Report on Test set

	precision	recall	f1-score	support
0	0.72	0.60	0.65	161
1	0.78	0.86	0.82	268
accuracy			0.76	429
macro avg	0.75	0.73	0.73	429
weighted avg	0.76	0.76	0.75	429

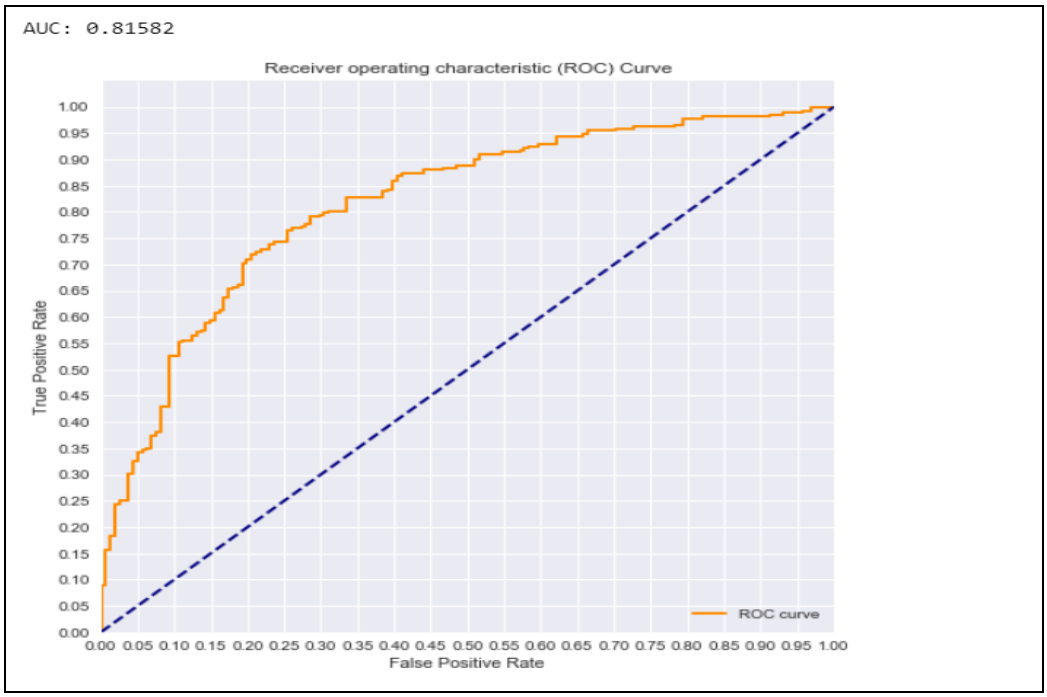
iii. Confusion Matrix for Train set



iv. Confusion Matrix for Test set



v. ROC curve for Test set



The train set accuracy is 100% and test set weighted accuracy is 75%. The AUC value for test set is 81%, which makes it the highest performing model so far.

5.3 Evaluation of Decision Tree Model

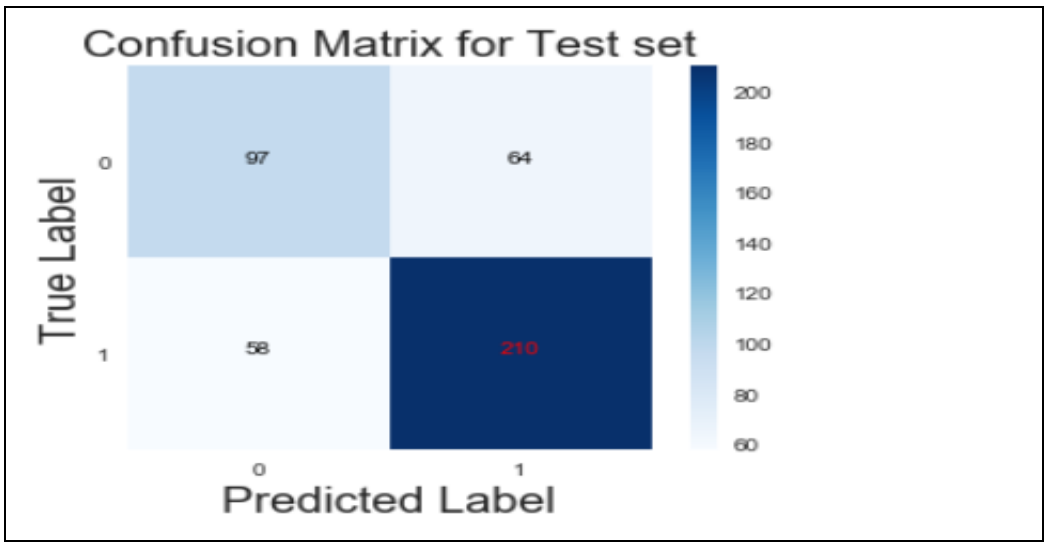
i. Classification Report on Training set

	precision	recall	f1-score	support
0	0.69	0.74	0.72	381
1	0.83	0.80	0.82	617
accuracy			0.78	998
macro avg	0.76	0.77	0.77	998
weighted avg	0.78	0.78	0.78	998

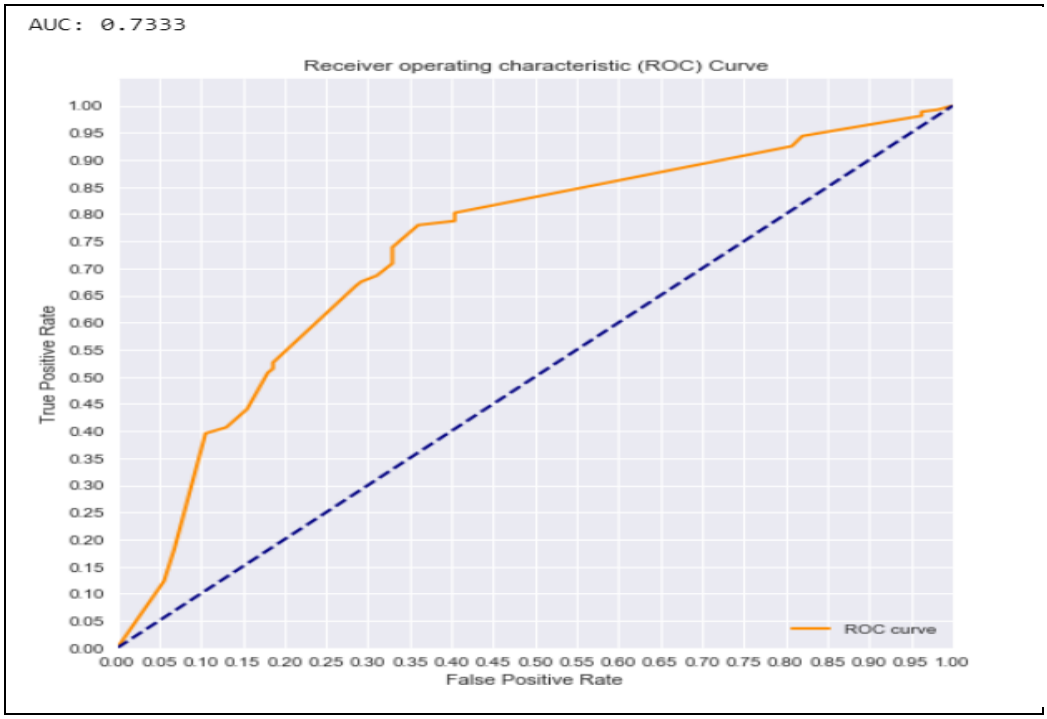
ii. Classification Report on Test set

	precision	recall	f1-score	support
0	0.63	0.60	0.61	161
1	0.76	0.79	0.78	268
accuracy			0.72	429
macro avg	0.70	0.69	0.69	429
weighted avg	0.71	0.72	0.71	429

iii. Confusion Matrix for Test set



vi. ROC curve for Test set



The train set accuracy is less than previous models and test set weighted accuracy is 71%. The AUC value for test set is 73%, which is better than logistic regression default parameter model.

6 Summary

Model	Train set F1 weighted accuracy	Test set F1 weighted accuracy	AUC
Logistic regression (with PCA)	0.89	0.69	0.75
Random Forest	1	0.71	0.80
XGBoost	1	0.75	0.81
Decision Tree	0.78	0.71	0.73

In this paper we have used a total of 4 machine learning models to predict whether ICO projects successfully funded or not. Based on Evaluation results, XGBoost model gave best results with training accuracy of 100% and test accuracy of 75% and with AUC score of 0.81. Both random forest and XGBoost gave training accuracy of 100% but there is big difference between train and test scores in these models, it

is possible that model is overfitting. Random forest is little slower than XGBoost and confusion matrix results and AUC score in XGBoost is better than Random Forest. So, we may have to choose XGBoost as best model.

Interestingly, each model performed worse at predicting failures compared to successes, with a lower true negative rate than true positive rate i.e., it classified quite a few failed projects as successes, but relatively few successful projects as failures.