Machine Learning Advanced Nanodegree
Capstone Project Report

Kailas Kasar
23rd Jul 2018

I Definition:

Project Overview

Banking today has become one of the major industries, covering almost all aspects of life that we can think of. The various services and products made available are as per requirement of common customers. Bank reaches out to customers primarily through different channels like call centers, direct mail and face-to-face. Though Internet and Mobile marketing is now becoming common marketing tools for banks direct marketing is main strategy of many banks to interact with their customers.

Increase in digitalization has lead generation of gigabytes of data. Finding accurate insight for Marketing campaign using huge amount of data is becoming important for Banks.

It is very easy for Fortune 500 companies to plan as many as 3000 campaigns in a single year but still the effort are insignificant if they are not reaching prospects likely to increase deposit in bank. They can't afford to send direct mails to huge, undifferentiated databases. The frequency and turnaround of campaigns is higher than ever, and so is the expectation for return on investment.

The data mining has been used widely in direct marketing to identify prospective customers for new products, by using purchasing data, a predictive model to measure that a customer is going to respond to the promotion or an offer. Data mining has gained popularity for illustrative and predictive applications in banking processes **[1].**

Problem Statement

Marketing campaigns are designed based on insights and decision derived from analyzing data. Analyzing huge amount of data using manual human efforts is nearly impossible. Machine Learning techniques like Logistic Regression, Decision Trees, and Random Forest become handy for predicting customers that have a higher probability to subscribe term deposit service offered by bank.

We will evaluate the performance of the classification models using statistical metrics like accuracy, sensitivity, precision etc. Model with better prediction with good accuracy is chosen to target the prospective customers.

Evaluation Metrics

Different performance metrics are used to evaluate different Machine Learning algorithms.
In the classification problem True or False refers to classification predictions. Confusion matrix represents actual number of True or False cases predicted by implemented mode so I have used confusion matrix for evaluation of performance of classification problem **[2]**.

Confusion Matrix: Confusion Matrix helps to evaluate the performance by comparing proportion of True or False prediction using Accuracy, Recall, Precision and Specificity.

The **accuracy** is the proportion of the total number of predictions that were correct.
The **recall** or **sensitivity** is proportion of actual positive cases which are correctly identified.
The **precision** is the proportion of the positive cases that were correctly identified.
The **specificity** is the proportion of actual negative that were correctly identified.

II. Analysis

Data Exploration

The data is related with direct marketing campaigns (phone calls) of a Portuguese banking institution. The classification goal is to predict if the client will subscribe a term deposit (variable y) **[3].**

Input variables

Bank client data:
1. age (numeric)
2. job : type of job (categorical: 'admin.','blue-collar','entrepreneur','housemaid', 'management','retired','self-employed','services','student','technician','unemployed','unknown')
3. marital : marital status (categorical: 'divorced','married','single','unknown'; note: 'divorced' means divorced or widowed)
4. education (categorical: 'basic.4y','basic.6y','basic.9y','high.school','illiterate', 'professional.course','university. degree','unknown')
5. default: has credit in default? (categorical: 'no','yes','unknown')
6. housing: has housing loan? (categorical: 'no','yes','unknown')
7. loan: has personal loan? (categorical: 'no','yes','unknown')

Related with the last contact of the current campaign:
1. contact: contact communication type (categorical: 'cellular','telephone')
2. month: last contact month of year (categorical: 'jan', 'feb', 'mar', ..., 'nov', 'dec')
3. day_of_week: last contact day of the week (categorical: 'mon','tue','wed','thu','fri')
4. duration: last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.

Other attributes:
1. campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)
2. pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
3. previous: number of contacts performed before this campaign and for this client (numeric)
4. poutcome: outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')

Social and economic context attributes
1. emp.var.rate: employment variation rate - quarterly indicator (numeric)
2. cons.price.idx: consumer price index - monthly indicator (numeric)
3. cons.conf.idx: consumer confidence index - monthly indicator (numeric)
4. euribor3m: euribor 3 month rate - daily indicator (numeric)
5. nr.employed: number of employees - quarterly indicator (numeric)

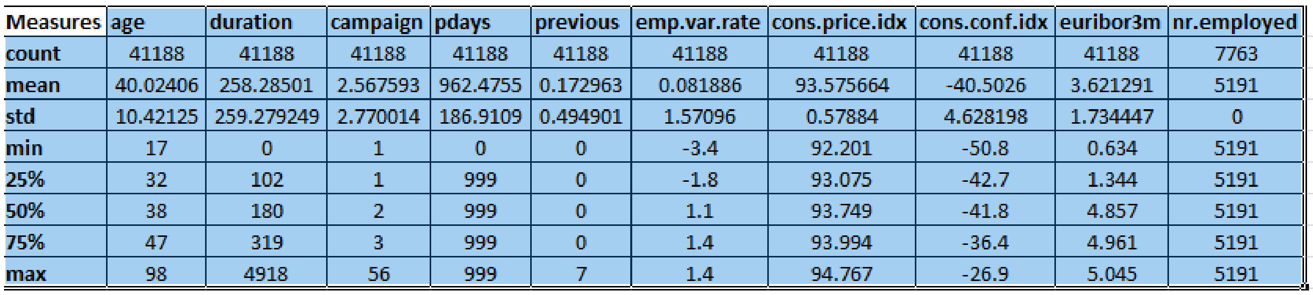
Output variable (desired target):
1. y - has the client subscribed a term deposit? (binary: 'yes','no')

- Dataset has 21 columns and 41188 rows.
- Number of customers who subscribed: 4640
- Number of customers who did not subscribe: 36548
- Percentage of customers who subscribed: 11.265417111780131%

Statistical description of the dataset

Descriptive statistic summary shows that 'duration' is showing variance and potentially can be a good predictor for our model.

Summary shows that 75% of the clients are younger than 47 years old.



Measures	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
count	41188	41188	41188	41188	41188	41188	41188	41188	41188	7763
mean	40.02406	258.28501	2.567593	962.4755	0.172963	0.081886	93.575664	-40.5026	3.621291	5191
std	10.42125	259.279249	2.770014	186.9109	0.494901	1.57096	0.57884	4.628198	1.734447	0
min	17	0	1	0	0	-3.4	92.201	-50.8	0.634	5191
25%	32	102	1	999	0	-1.8	93.075	-42.7	1.344	5191
50%	38	180	2	999	0	1.1	93.749	-41.8	4.857	5191
75%	47	319	3	999	0	1.4	93.994	-36.4	4.961	5191
max	98	4918	56	999	7	1.4	94.767	-26.9	5.045	5191

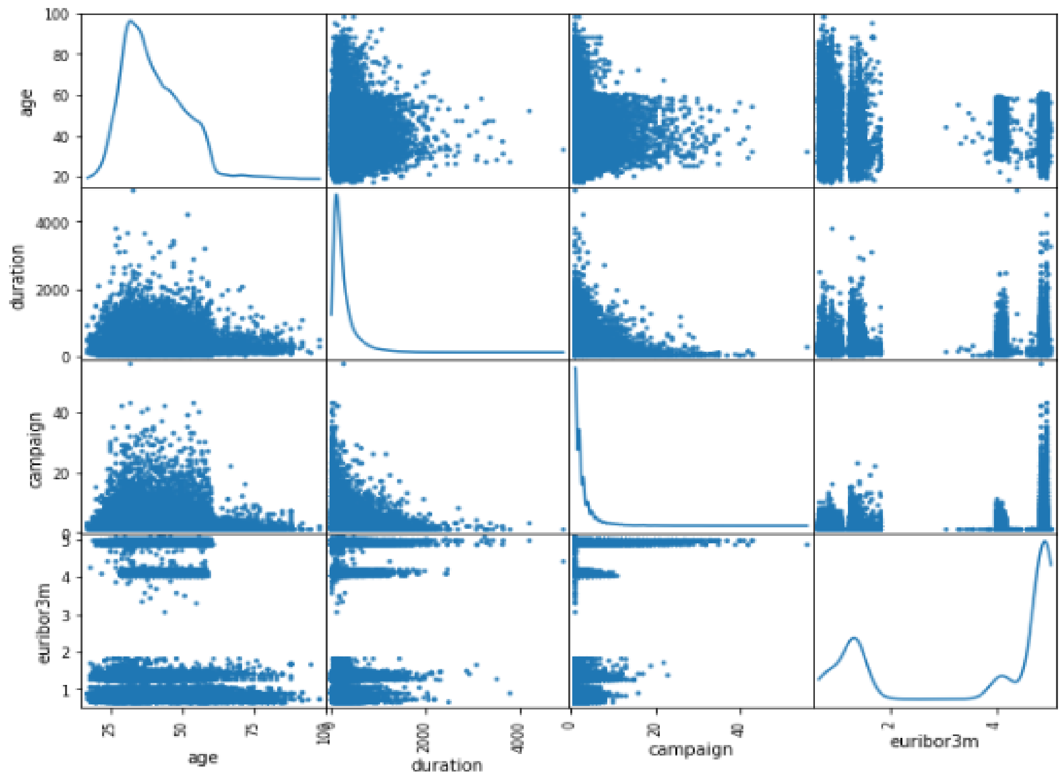
The skew result shows a positive (right) or negative (left) skew. Values closer to zero are less skew.

Below table shows the skew result for data.

Age	0.784697
Duration	3.263141
Campaign	4.762507
Pdays	-4.92219
Previous	3.832042
emp.var.rate	-0.7241
cons.price.idx	-0.23089
cons.conf.idx	0.30318
euribor3m	-0.70919
nr.employed	0

Exploratory Visualization [4]:

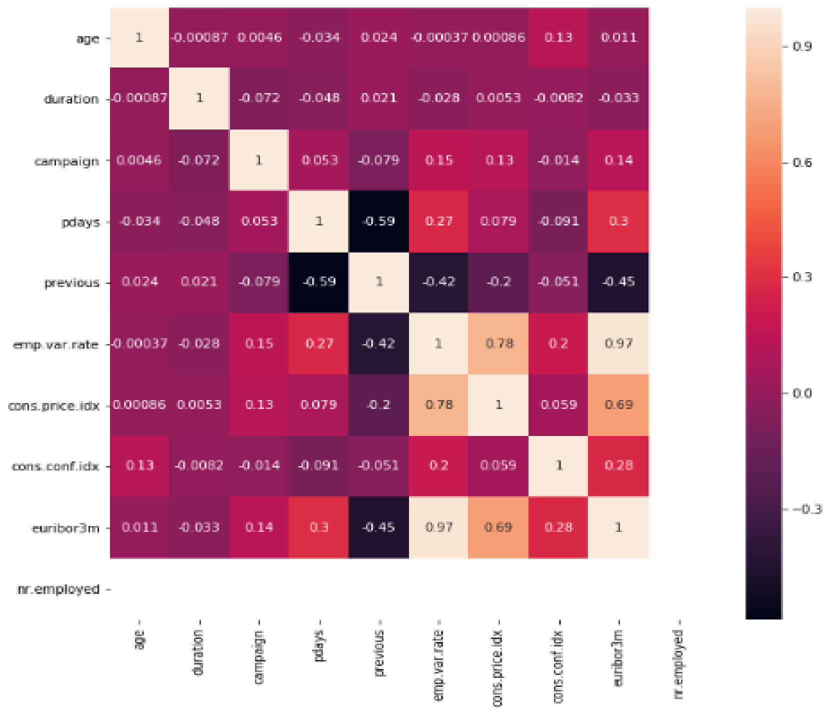
In next graph I has plotted numeric variables using scatter matrix to get more visual information on data.



- ✓ There are splits in data for euribor3m,
- ✓ From plot we can say that duration is smaller for age > 60.
- ✓ From summary and plot we can conclude that 75% of clients were contacted less than three times during this campaign.

Correlation Heatmap

Let’s try to find correlations of features using heatmap.



The correlation among emp.var.rate, euribor3m, and nr.employed are very strong which is consistent with the fact that they are indicators for macro-economic environment , these strong relationships need to be considered when algorithms are evaluated and interpreted. Other than these there was not much of strong relationship in data.

Algorithms and Techniques

Different machine learning techniques are available to solve classification problem. In classification tasks, the machine learning program must draw a conclusion from observed values and determine to what category new observations belong

1) **Logistic Regression:** Logistic regression focuses on estimating the probability of an event occurring based on the previous data provided. It is used to cover a binary dependent variable, that is where only two values, 0 and 1, represent outcomes.

2) **Classification and Regression Trees (CART):** The CART algorithm is structured as a sequence of questions, the answers to which determine what the next question, if any should be. The result of these questions is a tree like structure where the ends are terminal nodes at which point there are no more questions.
3) **Classification Trees:** where the target variable is categorical and the tree is used to identify the "class" within which a target variable would likely fall into.
Regression Trees: where the target variable is continuous and tree is used to predict it's value.

4) Random Forests (RF): Random forests or ‘random decision forests’ is an ensemble learning method, combining multiple algorithms to generate better results for classification, regression and other tasks. Each individual classifier is weak, but when combined with others, can produce excellent results.

5) **Adaptive Boosting (AB):** AdaBoost is a popular boosting technique which helps you combine multiple “weak classifiers” into a single “strong classifier”. A weak classifier is simply a classifier that performs poorly, but performs better than random guessing. AdaBoost can be applied to any classification algorithm, so it’s really a technique that builds on top of other classifiers as opposed to being a classifier itself.

6) **Extreme Gradient Boost (XGB):** XGBoost Algorithm is an implementation of **gradient boosted** decision trees. That was designed for speed and performance. The idea behind GBM is a more sophisticated. Roughly, the idea is to again, combine weak predictors. The trick is to find areas of misclassification and then “boost the importance of those incorrectly predicted data points. And repeat. The result is a single Tree unlike RF.

   XGBoost is a model based on tree ensemble which is a set of classification and regression trees (CART). XGBoost classifies the members of a family into different leaves and assigns scores on corresponding leaf. Usually, a single tree is a weak leaner which is not strong enough to use in practice. Boosted trees are typically shallow which are high in bias and low in variance. A tree ensemble model sums prediction of multiple tree.

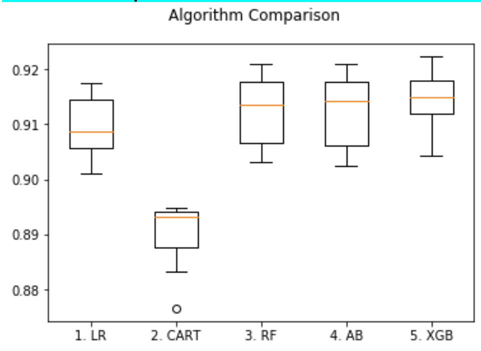
Benchmark Model

The given dataset is a typical supervised learning problem for which tree type models perform a lot better than the rest.

I will pick Extreme Gradient Boosting (XGB) as benchmark and try to beat the benchmark with hyper parameter turning **[5].**

Algorithm	**Accuracy**	**Std. Error**
LR	90.97%	(0.005144)
CART	88.96%	(0.006663)
RF	91.34%	(0.006129)
AB	91.26%	(0.006215)
XGB	91.46%	(0.005547)

Visual comparison of different models using box_plot



Box plots shows that CART is low performer among the models at approx 89% and XGB is best model with range near approx 91%, so XGB is best model we should use for this classification probelm.

III. Methodology

Data Preprocessing

Data preprocessing refers to the transformations applied to your data before feeding it to the algorithm. Data preprocessing is an important step of solving every machine learning problem. Most of the datasets used with Machine Learning problems need to be processed / cleaned / transformed so that a Machine Learning algorithm can be trained on it. Most commonly used preprocessing techniques are very few like - missing value imputation, encoding categorical variables, scaling, etc.

We need to prepare the data by creating dummy variables for each of the categorical columns (since we cannot use textual data to build our model). e.g. housing. these can be reasonably converted into 1/0 (binary) values and other categorical columns like job with more than two values can assign with dummy variables and assign a 1 to one of them and 0 to all others.

It is challenge to decide how to treat the missing values; you can either drop or fill missing values. As columns with missing values may hold relevant information so dropping column may impact model predictions hence I have decided to fill missing values with median statistic.

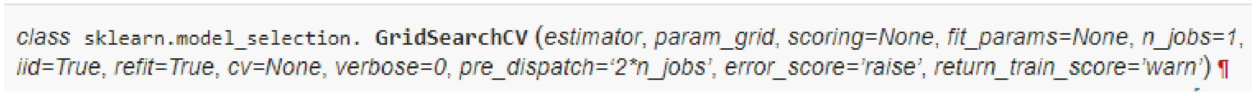
To choose the correct model we need to validate the model performance, To do so it is general practice to split data into training and testing dataset. so that one dataset is used to tune prediction model and another part assess the model's performance. However for this problem we have small number of observations in our dataset so instead of using two separate datasets, I have decided to use cross-validation method would allow us to use one dataset for both purposes.

In cross-validation we divide the dataset in several segments that are used to test the model repeatedly. In a single iteration, all but one of the segments would be used to train a predictive model.

I have used 10 fold cross- validations so my 9 segment are used to train a model and 1 is used to test the model. This process is repeated until each segment has been used as the test segment.

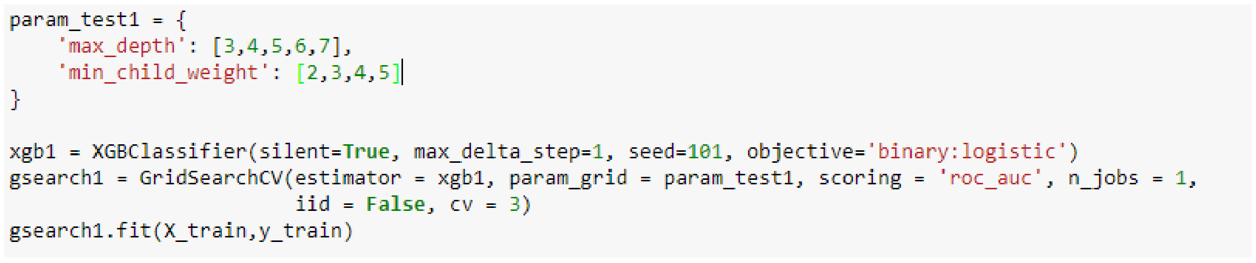
To check if the model I created is any good, I will split the data into training and validation sets to check the accuracy of the best model. We will split the given training data in two ,70% of which will be used to train our models and 30% we will hold back as a validation set.

Scikit-learn library has made code part easy for classification problem. **“**XGBoost provides a wrapper class to allow models to be treated like classifiers**”**. Challenging part is to tune parameters to improve the performance of un-tuned model. Scikit-learn library has hyperparameter optimizer GridSearchCV to search best combination of parameters **[6]**



```
class sklearn.model_selection. GridSearchCV (estimator, param_grid, scoring=None, fit_params=None, n_jobs=1,
iid=True, refit=True, cv=None, verbose=0, pre_dispatch='2*n_jobs', error_score='raise', return_train_score='warn') ¶
```

Below is the code use to tune the parameter. I have passed different sets of parameter to **param_grid** with range of value. I have repeated below code for other selected parameters to get best possible combination of parameters.



```
param_test1 = {
    'max_depth': [3,4,5,6,7],
    'min_child_weight': [2,3,4,5]
}

xgb1 = XGBClassifier(silent=True, max_delta_step=1, seed=101, objective='binary:logistic')
gsearch1 = GridSearchCV(estimator = xgb1, param_grid = param_test1, scoring = 'roc_auc', n_jobs = 1,
                        iid = False, cv = 3)
gsearch1.fit(X_train,y_train)
```

Implementation

The workflow of solving this problem will be in the following order:

- Exploring the Data
    - Loading Libraries and data
    - Peek at the training data
    - Dimensions of data
    - Overview of responses and overall response rate
    - Statistical summary
- Data preprocessing
    - Preprocess feature columns
    - Identify Feature and Target columns
    - Data cleaning
    - Training and Validation data split
    - Feature Scaling - Standardization/Normalizing data
- Evaluate Algorithms
    - Build models
    - Select best model
    - Make predictions on the validation set
    - Feature importance and feature selection
- Model Tuning to Improve Result
- Final conclusion

Refinement

I have performed hyper tuning of below listed parameters of XGBoost wrapper from scikit-learn library [7].

- max_depth : maximum depth of a tree to control overfitting
- min_child_weight : minimum sum of weights of all observations required in a child
- Gamma : minimum loss reduction required to make a split
- reg_alpha : L1 regularization term on Weights
- reg_lambda : L2 regularization term on Weights
- Subsample : Subsample ratio of the training instance
- colsample_bytree : Subsample ratio of columns when constructing each tree
- colsample_bylevel : Subsample ratio of columns for each split, in each level

I have tested different values for parameter to get best value for that parameter.

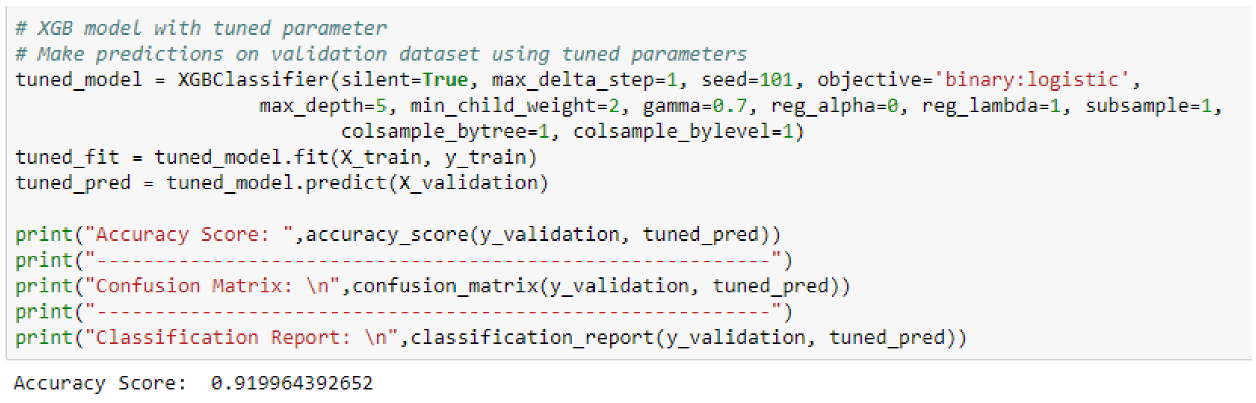
Parameter	**Value Tested**	**Best Value**
max_depth	3,4,5,6,7	5
min_child_weight	2,3,4,5	2
Gamma	(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.9,1.0)	0.7
reg_alpha	(0,1,2,3)	0
reg_lambda	(0,1,2,3)	1
Subsample	(0.5,1)	1
colsample_bytree	(0.5,1)	1
colsample_bylevel	(0.5,1)	1

IV. Results

Model Evaluation and Validation

After comparing different classification models XGBoost is giving better accuracy for classification problem. I tried to improve the XGB model performance by tuning its parameter.

To validate the model implemented with tuned parameters, I have used validation dataset with the help of XGBClassifier, It has resulted Accuracy Score of 91.99%. Below is snapshot of code.



```
# XGB model with tuned parameter
# Make predictions on validation dataset using tuned parameters
tuned_model = XGBClassifier(silent=True, max_delta_step=1, seed=101, objective='binary:logistic',
                     max_depth=5, min_child_weight=2, gamma=0.7, reg_alpha=0, reg_lambda=1, subsample=1,
                            colsample_bytree=1, colsample_bylevel=1)
tuned_fit = tuned_model.fit(X_train, y_train)
tuned_pred = tuned_model.predict(X_validation)

print("Accuracy Score: ",accuracy_score(y_validation, tuned_pred))
print("----------------------------------------------------------")
print("Confusion Matrix: \n",confusion_matrix(y_validation, tuned_pred))
print("----------------------------------------------------------")
print("Classification Report: \n",classification_report(y_validation, tuned_pred))
```

```
Accuracy Score:  0.919964392652
```

Below are statistics given by tuned and un-tuned model.

	Tuned	**Un-tuned**
Accuracy	91.99%	91.84%
precision	0.91	0.91
Recall	0.92	0.92
f1-score	0.92	0.91
TP	10599	10615
FN	366	350
FP	623	658
TN	769	734

Robustness of model for making predictions on unseen data can ensure by using classification accuracy. However accuracy along cannot be used to make the decision. There are two other important measures Precision and Recall can use to evaluate the classification problem.**[8]**

If we improve recall rate our model will become more robust, to improve recall rate we should tune classifier to reduce the false negative predictions. Our dataset is imbalanced so we should improve recall rate for class label 1. It will cost us on accuracy but will improve overall performance of the model.

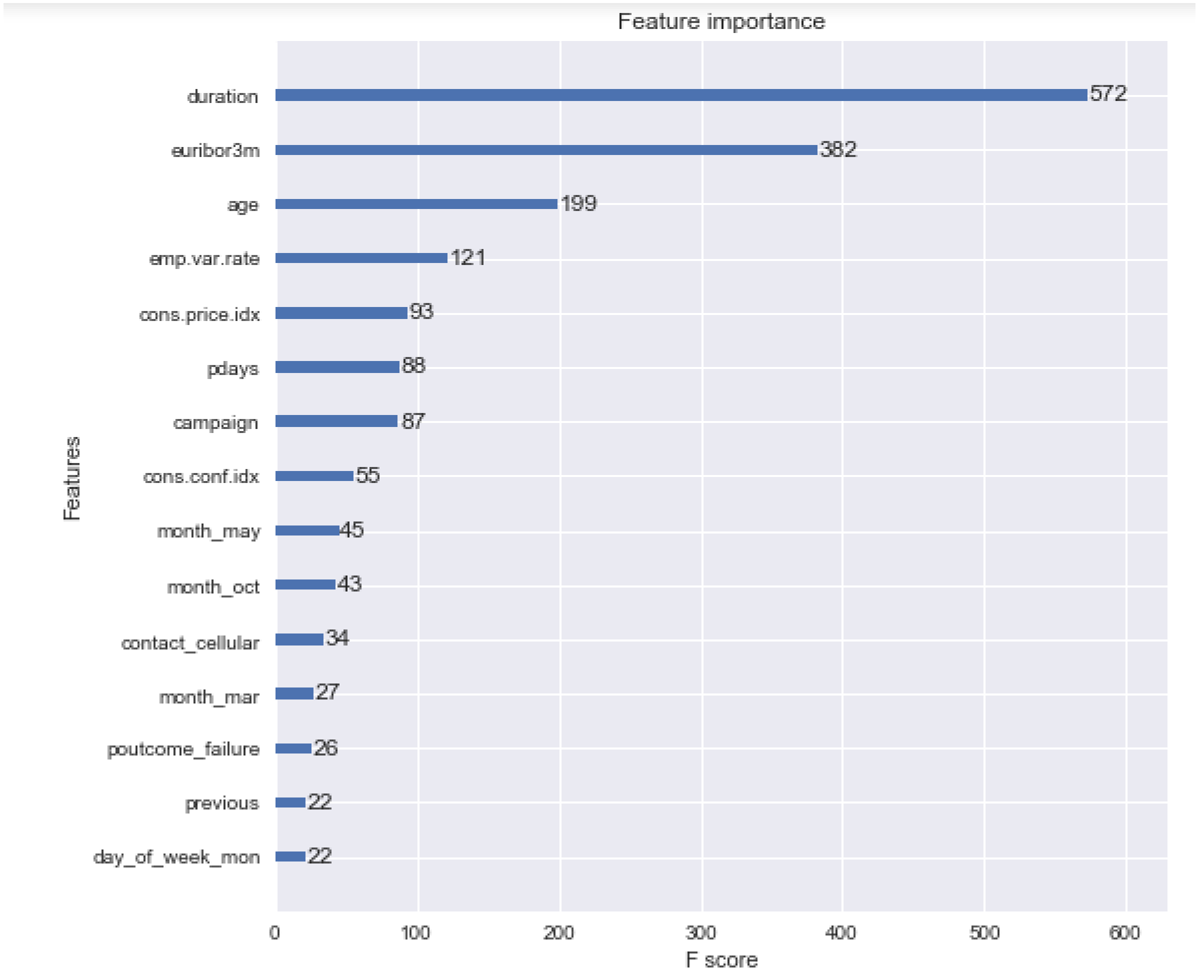
Justification

I have tuned 8 parameters which results in improving model performance by 0.15% also f1-score for tuned model has improve thus parameter tuning has worked for predicting the target customers and I think solution is significant enough to solved the classification problem of Portuguese banking institution.

V. Conclusion

Free-Form Visualization

The importance of each feature was visualized with the following plot:



Feature important plot has shown the importance of different predictors for term deposit service offered by bank.

Reflection

The solution to this classification problem was started with data cleansing and processing as there were multiple columns with null values. Once the data was prepared and ready, the next challenge was to pick an algorithm that could be best suited for the problem. I have selected different Machine Learning algorithms for this problem after comparing performance of different algorithm I observed that XGBoost performed the best out of others. Finally challenge was to improve performance of un-tuned model by tuning the parameters. Parameters are tuned to improve the accuracy of prediction.

Parameter tuning is interesting part of working on the capstone project. Parameter takes different range of values, when you change a single parameter value, best value for other parameter and overall accuracy of model changes accordingly. I have learn that if you spend your time in tuning parameters with range of values you will ultimately find best combinations of parameter for classification problem.

duration is most important feature that means if customer is interested in taking the Term Deposit product then he is taking more information regarding product to take his decision to invest in term deposit. **euribor3m** is another important variable as customer always demands higher interest on his deposits, more interest rate then more willingly customer will spend their money on financial tools. There is no surprise that **Age** is another important variable, it is obvious that customers will invest their money in term deposit while they are young and earning.

Therefor bank should focus on tele-marketing executives with good communication skills who will engage young and earning customers with their quality conversation; also bank should target there customers when interest rates are high.

Improvements

Model has performed well with 91.99% of accuracy so i don't think further parameter tuning is required for this model otherwise it will lead to overfitting.

However feature engineering, feature subsetting, feature importance analysis has a potential to increase the model's performance. Using XGBoost to get a subset of important features allows us to increase the performance of models without feature selection by giving that feature subset to them. Using feature selection based on feature importance can greatly increase the performance of your models **[9] [10] [11]**.

References

[1] https://analyticsindiamag.com/analytics-in-indian-banking-sector-on-a-right-track/

[2] http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/confusion_matrix.html

[3] https://archive.ics.uci.edu/ml/datasets/bank+marketing

[4] https://towardsdatascience.com/the-art-of-effective-visualization-of-multi-dimensional -data-6c7202990c57

[5] https://machinelearningmastery.com/compare-machine-learning-algorithms-python-scikit-learn/

[6] http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[7] https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/

[8] http://www.chishiki.soft.iwate-pu.ac.jp/ISNLP/Chinoukikai-LectureNote/Lecture0.pdf

[9] https://datawhatnow.com/feature-importance/

[10] https://www.cc.gatech.edu/~hic/CS7616/Papers/Kumar-Minz-2014.pdf

[11] https://stats.stackexchange.com/questions/264093/does-feature-selection-help-improve-the-performance-of-machine-learning