

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | |
|  | SANTANDER CUSTOMER TRANSACTION PREDICTION | | | | | |  |
|  |  | | | | |  | |
|  | | |  |  | | | |
|  | | | santander_logo |  | | | |
|  | | | 11/12/2019—Data science—By Suraj Mohan Raut |  | | | |
|  | |  | | |  | | |

**INDEX**

**Chapter 1 Introduction**

* 1. Problem Statement
  2. Data
  3. Problem Analysis
  4. Programming languages

**Chapter 2 Methodology**

* 1. Business understanding
  2. Data pre-Processing
  3. Model development
  4. Model validation
  5. Model Selection
  6. Prediction of target column for test.csv

**Chapter 3 Pre-processing**

* 1. Understanding the data
  2. Missing value analysis
  3. Outliers analysis
  4. Handling imbalanced data
  5. Feature selection
  6. Feature engineering

**Chapter 4 Modelling**

* 1. Logistic Regression
  2. Decision Tree
  3. Random Forest
  4. Naive Bayes model
  5. SVC
  6. KNN
  7. XGBoost
  8. LightGBM
  9. SGD classifier

**Chapter 5 model validation**

* 1. Model Evaluation and Model Selection
  2. Predicting the target column value test dataset

**Chapter 6 Instructions to deploy and run code**

**Appendix**

**References**

**Chapter 1 Introduction**

**Santander** is one of the best open financial services platforms. Santander operates in 10 main markets where it has high market shares: Spain, Germany, Poland, Portugal, the United Kingdom, Brazil, Mexico, Chile, Argentina and the United States. It also has a consumer financing business in other European countries and a presence in China through wholesale banking and consumer finance business.

So, our project ‘Santander customer transaction’ is a demo how it takes help from data science to know more about the customer and hence can take effective steps so that it improves its services and hence improve in the no. transaction by customers

* 1. **Problem Statement**

Identifying new ways to solve Santander’s most common challenge, binary classification problems such as: is a customer satisfied? Will a customer buy this product? Can a customer pay this loan?

In this challenge, we need to identify which customers will make a specific transaction in the future, irrespective of the amount of money transacted.

* 1. **Data**

We are provided with an anonymized dataset containing numeric feature variables, the binary target column, and a string ID\_code column. The task is to predict the value of target column in the test set.

1. ​​test.csv
2. ​train.csv

train.csv is a comma separated file containing 202 columns and 200k observations, on which we have to train various machine learning models.

test.csv is another comma separated file containing 201 columns and 200k observations for which we have to predict the target column value.

Since the data provided to us is bank details so it is encrypted and converted into anonymized numeric features.

* 1. **Problem Analysis**

This is a binary classification problem under supervised machine learning algorithm. The task is to predict the value of target column in the test set

* 1. **Programming languages**

1. Python
2. R

**Chapter 2 Methodology**

**2.1 Business understanding**

Santander is a big name from banking sector. At ​Santander​, mission is to help people and businesses prosper. They are always looking for ways to help our customers understand their financial health and identify which products and services might help them achieve their monetary goals.

Their main aim is to identify new ways to solve our most common challenge, binary classification problems such as: is a customer satisfied? Will a customer buy this product? Can a customer pay this loan?

**2.2 Data Pre-Processing**

Here our goal is to understand the given data and make the data eligible for model development. For that we use the following data:

1. Understanding the data
2. Descriptive analysis
3. Graphical analysis
4. Missing value analysis
5. Outliers analysis
6. Handling imbalanced data
7. Correlational analysis and Feature selection
8. Feature engineering
9. Splitting the data
   1. **Model development**
10. Implement the following models using **k fold cross validation**
11. Logistic regression
12. Decision tree
13. Random forest
14. Lightgbm
15. XGBoost
16. Naïve bayes
17. SGDClassifier
18. KNeighborsClassifier
19. SVC
20. Check the accuracy and the time taken
21. Implement the one which takes more accuracy and less time separately using hyperparameter tuning
    1. **Model validation**
22. K fold cross validation
23. Using confusion metrics
    1. **Model Selection**
24. Hyper parameter tuning
    1. **Prediction of target column for test.csv**

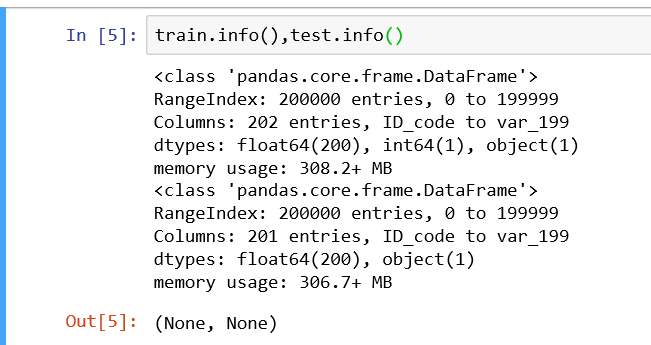
**Chapter 3 Pre-processing**

* 1. **Understanding the data**

1. Descriptive analysis

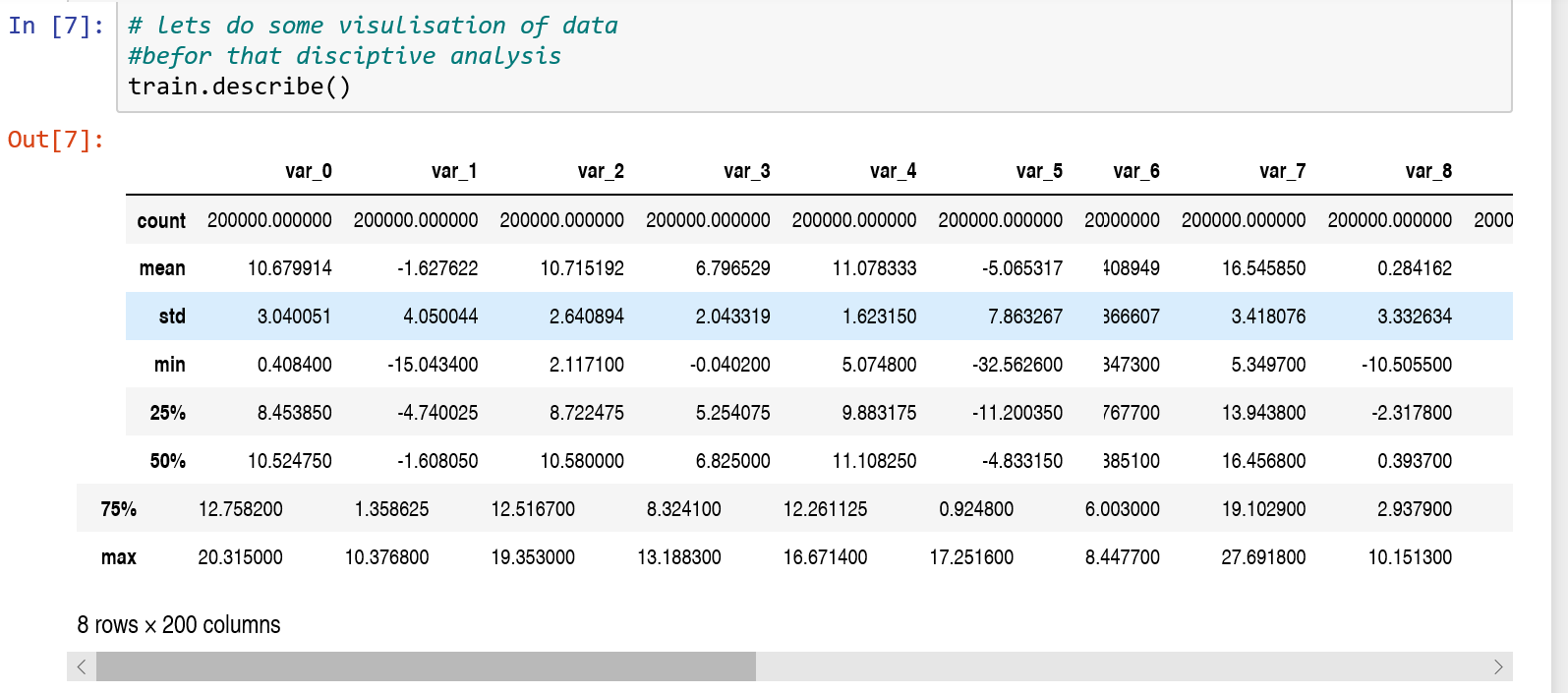
In train data set there are 202 columns and 200k observations, 1st columns is ID\_code which does not give any useful information so we remove it and second column is target column which has data type of int so we have to convert it to categorical data type

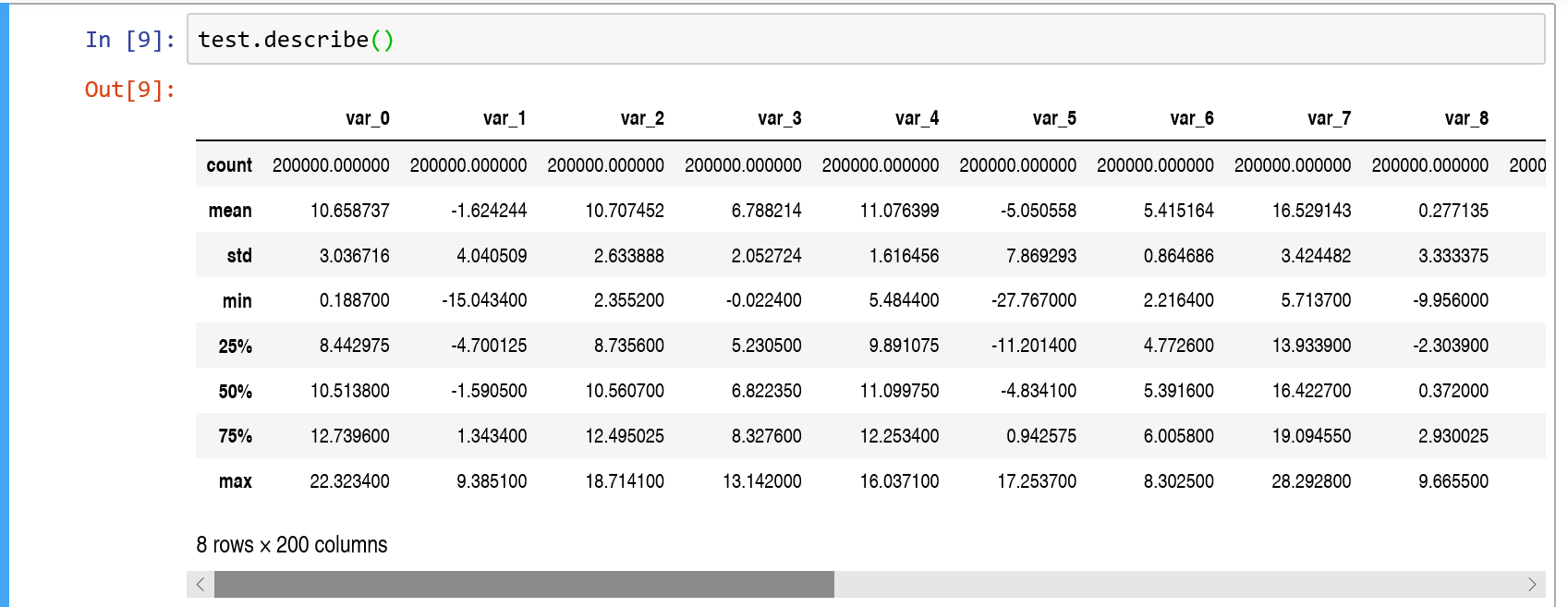
All left 200 columns are of float data type 200k observations



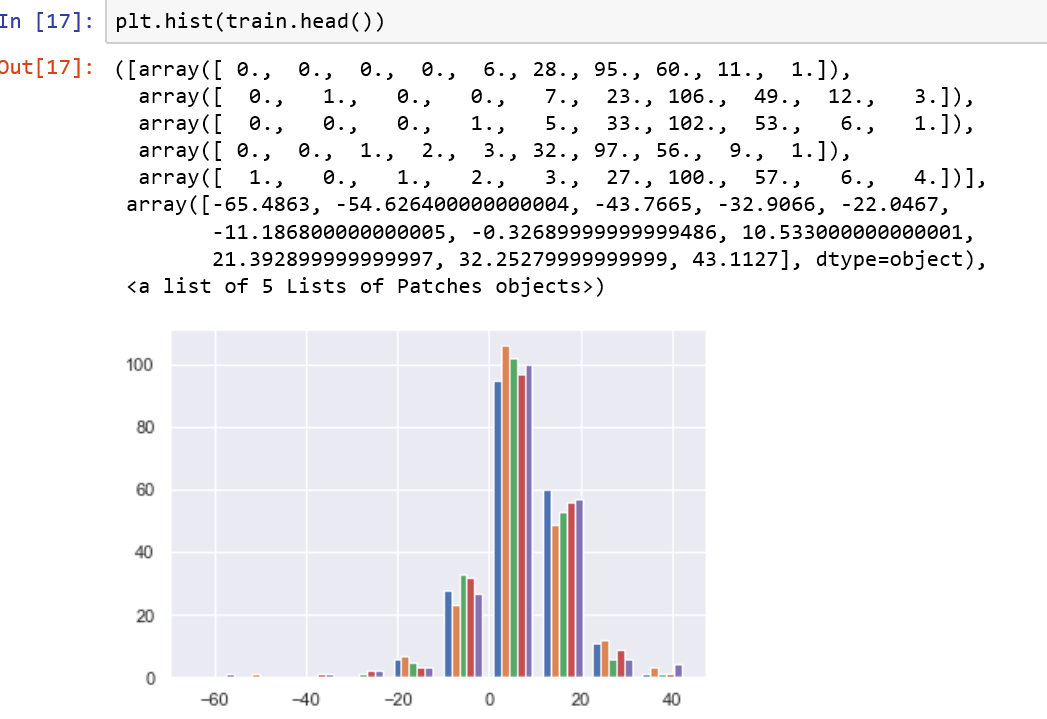
In descriptive statistics we can get the following values for each columns of both train and test data: -

Count ,mean ,standard deviation ,minimum ,maximum ,25th percentile ,75th percentile and median.





1. Graphical analysis



* 1. **Missing value analysis**

There are no missing values present in train data and test data .

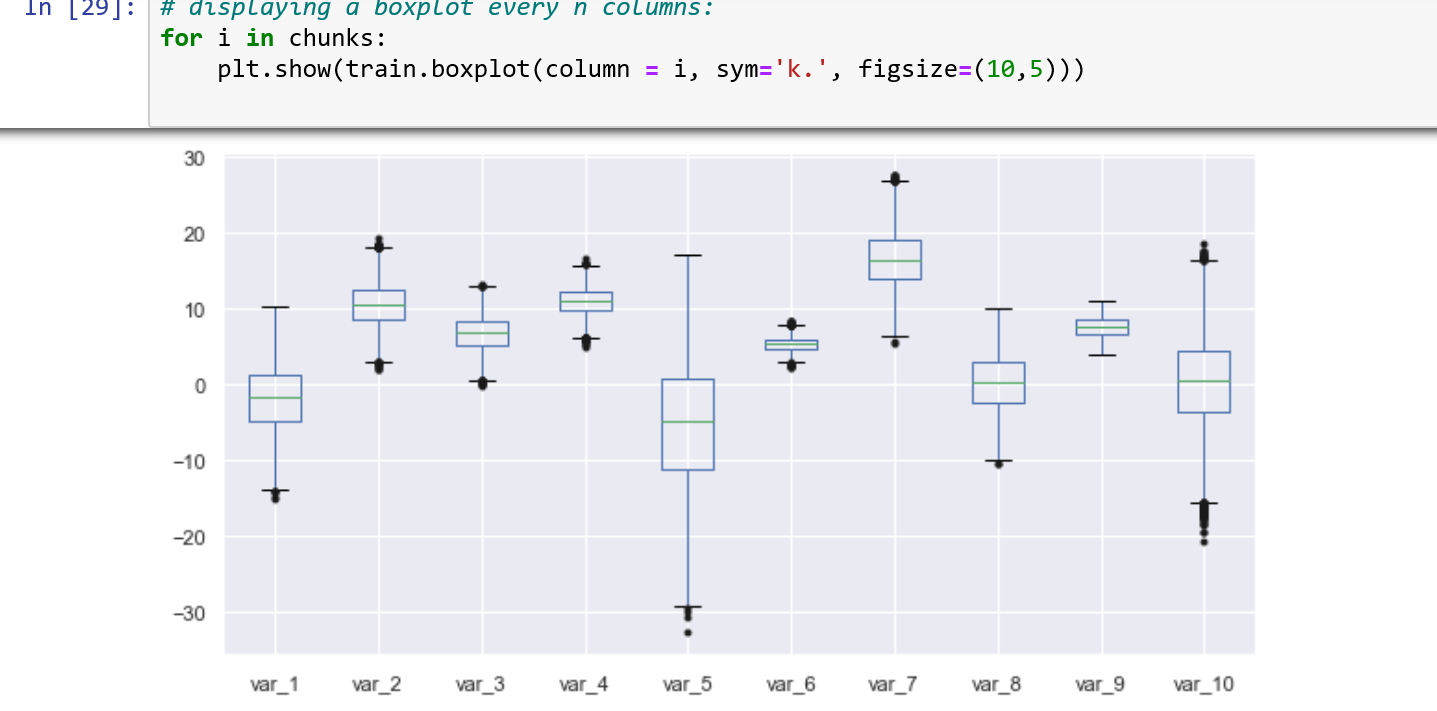


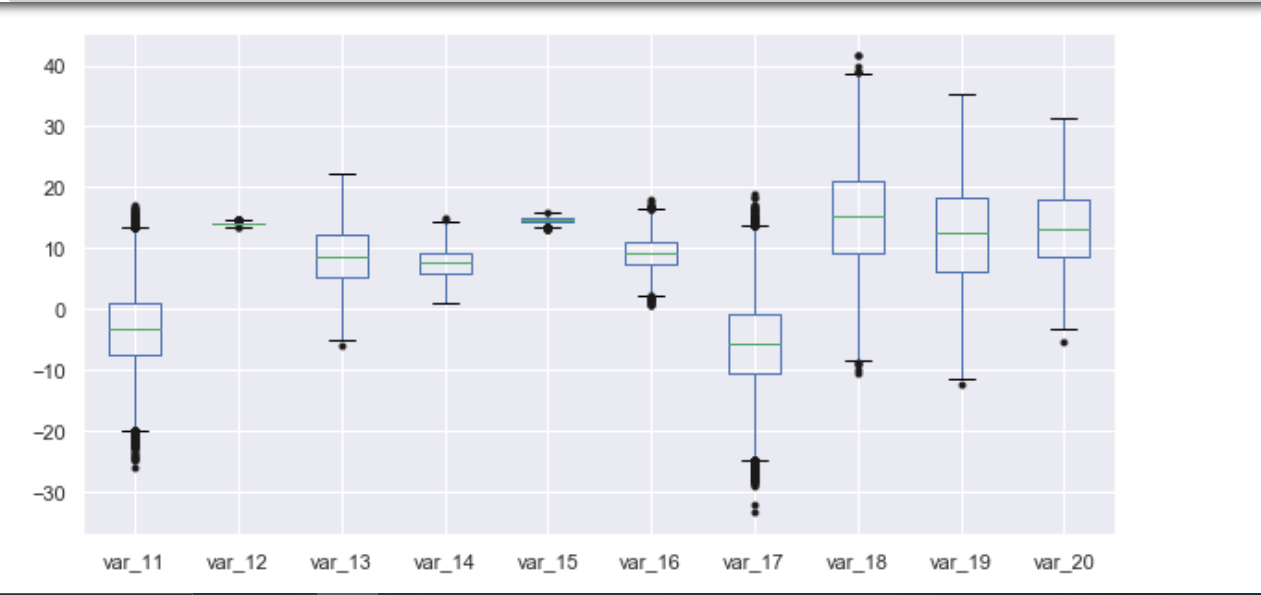
* 1. **Outliers analysis**

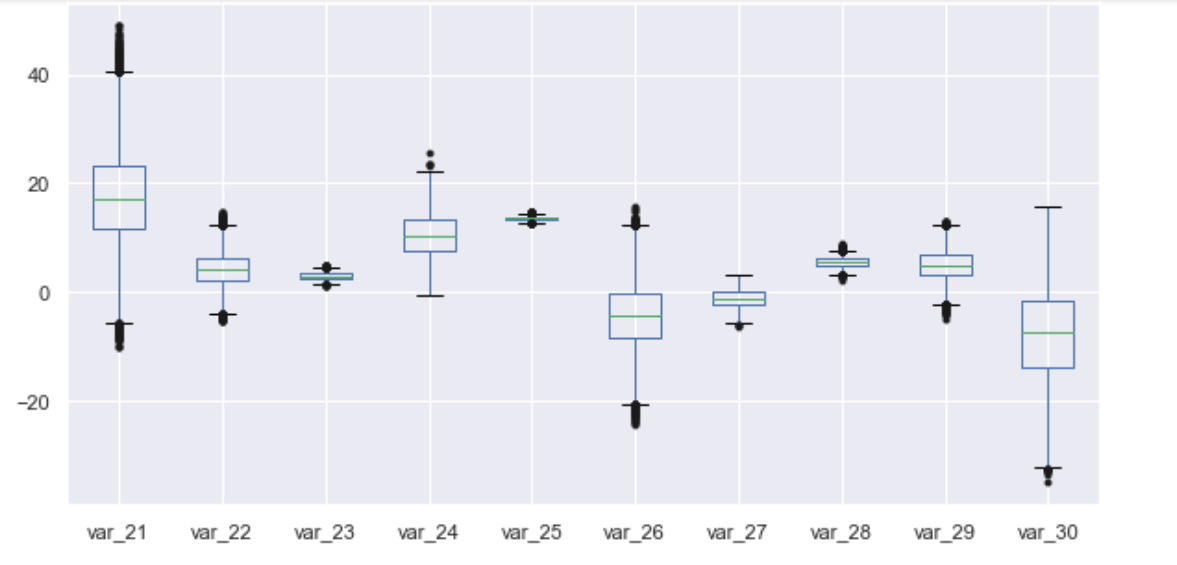
There are 26536 outliers which we can observe using boxplot.

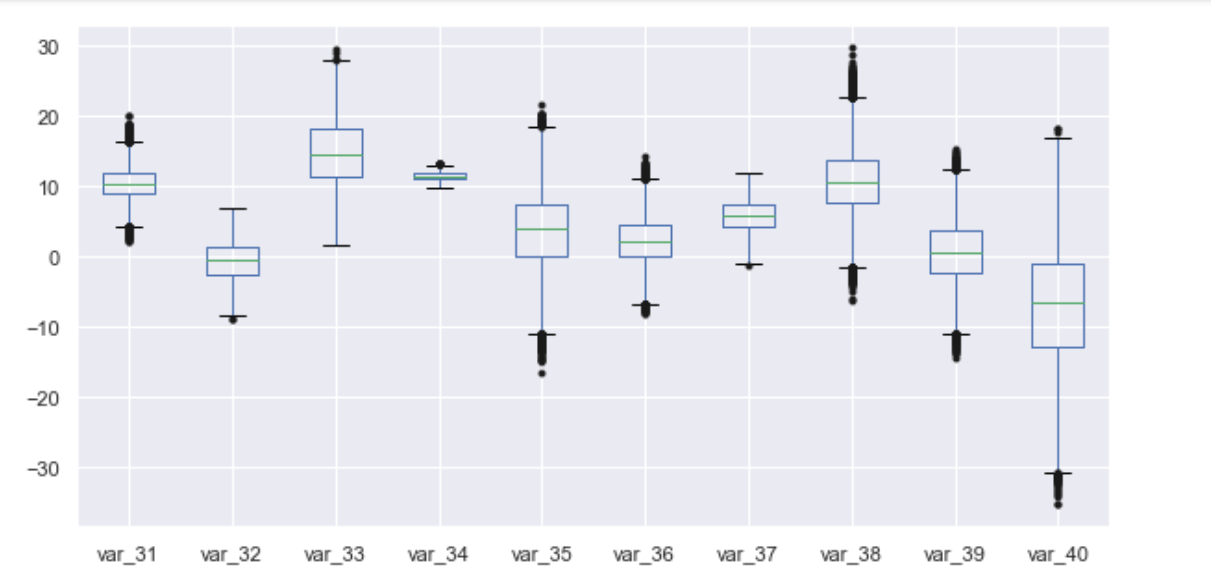
We removed outliers by converting them first to np.nan and the droping the np.nan values using **dropna()** function in python

Image of the python code is shown as follows:









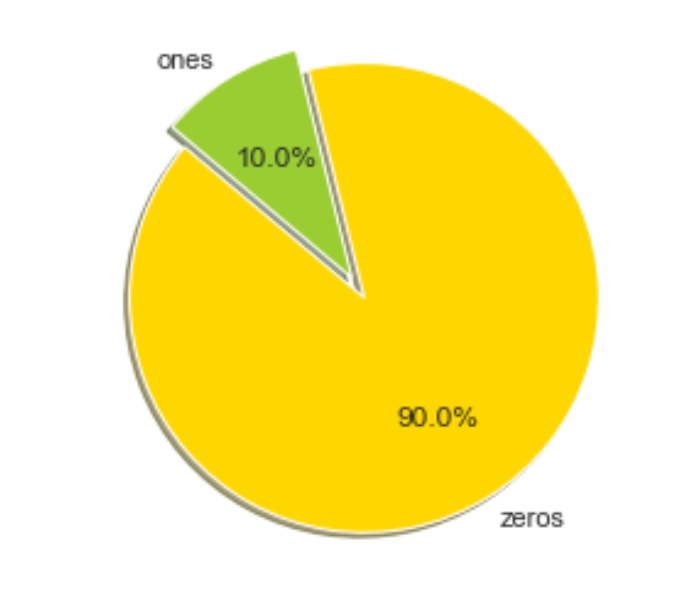
You can see the outlier at the edges of boxplot and 6 more plots are available

* 1. **Handling imbalanced data**

Given train data is highly imbalanced as the target columns has 89.9%zeroes and only 10.1%

Ones, which is displayed using following codes:





**Random Over-Sampling**

Over-Sampling increases the number of instances in the minority class by randomly replicating them in order to present a higher representation of the minority class in the sample.

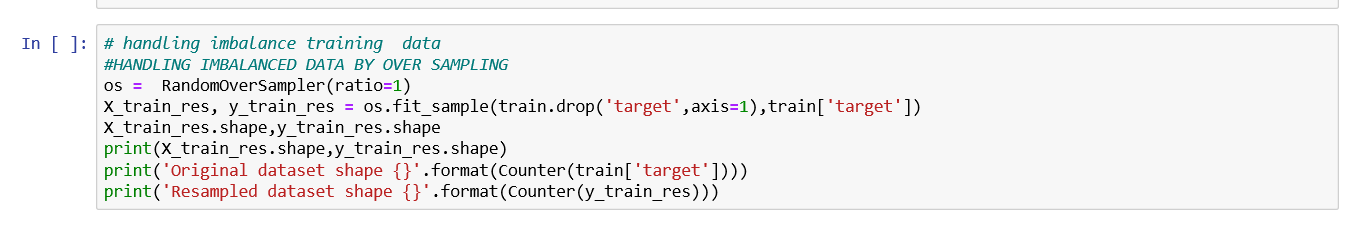
Advantages

● Unlike under sampling this method leads to no information loss.

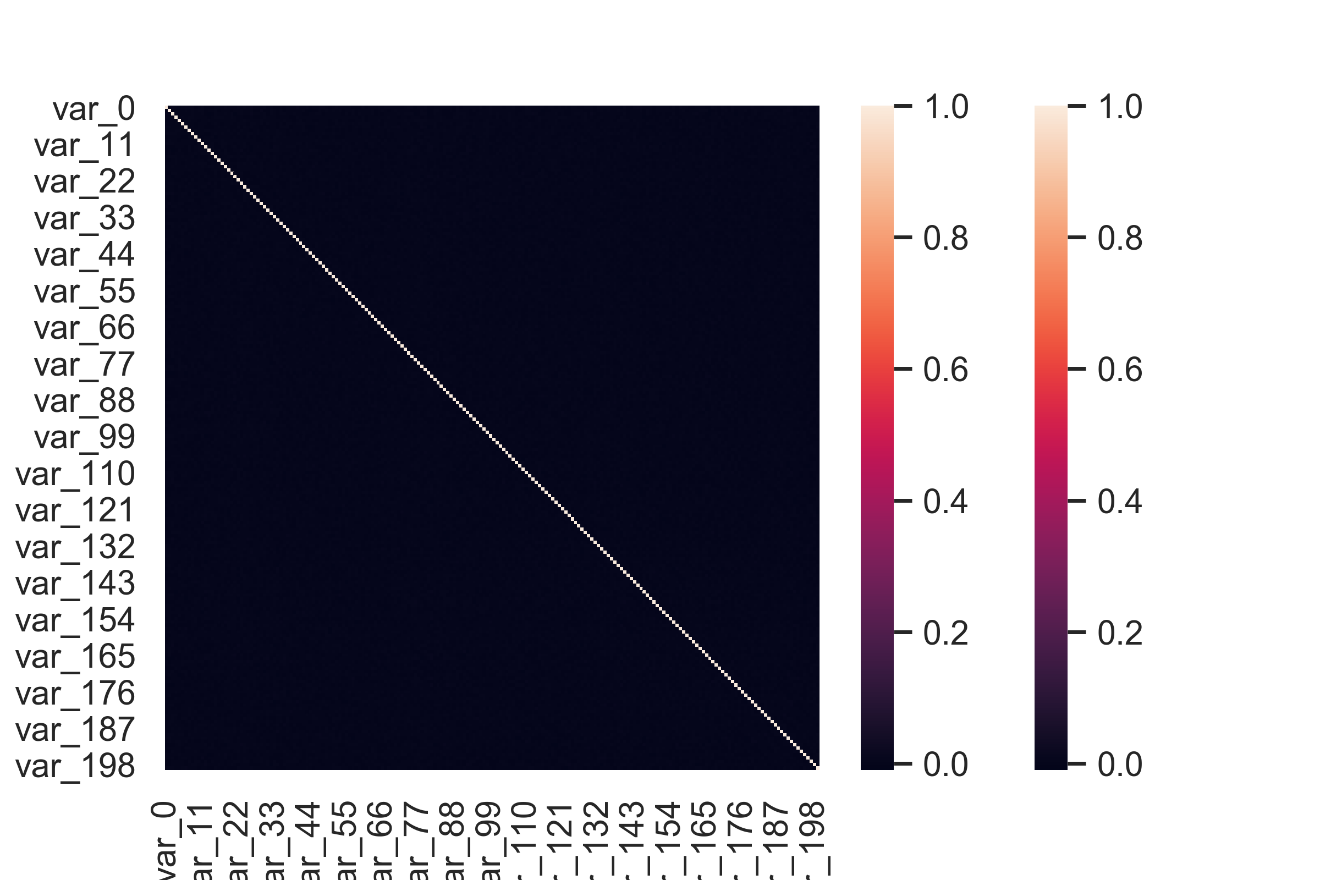
● Outperforms under sampling

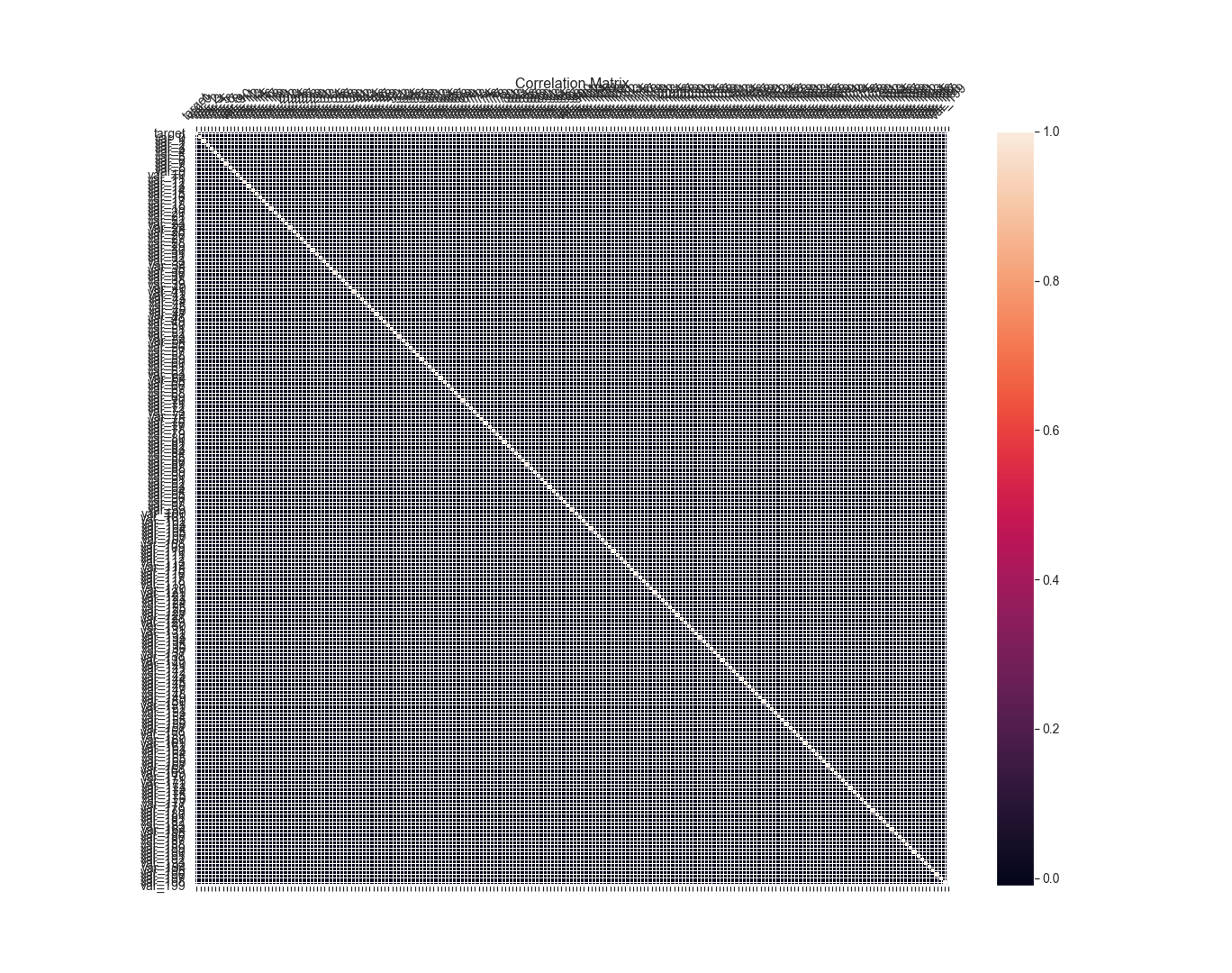
Disadvantages

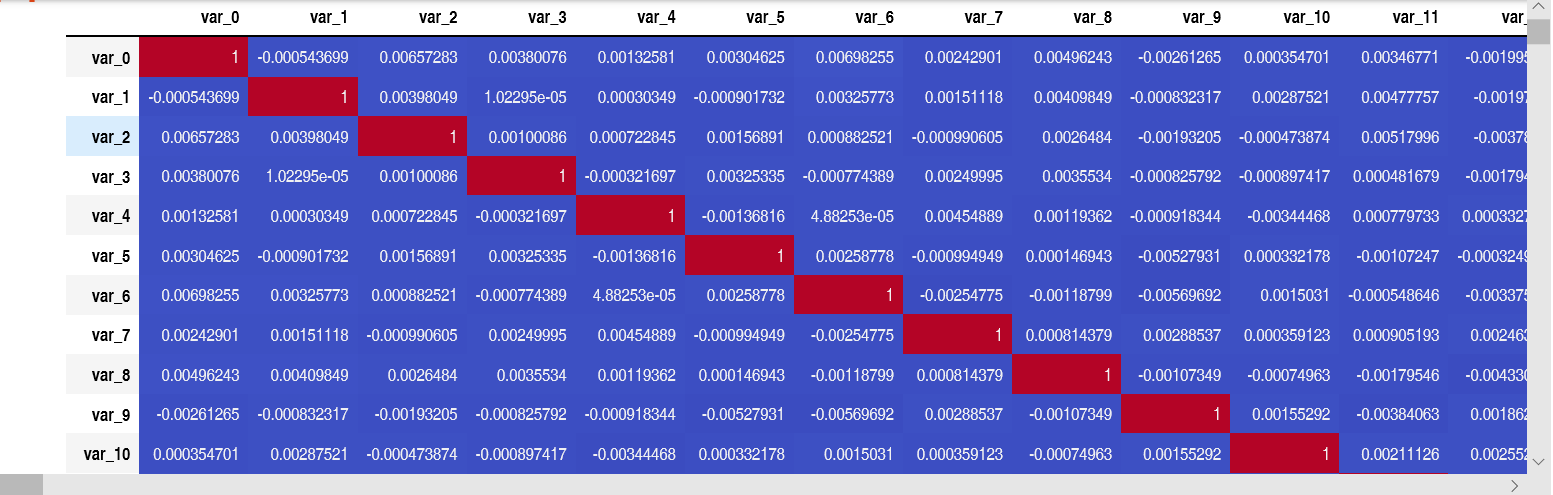
● It increases the likelihood of overfitting since it replicates the minority class  events.



* 1. **Feature selection** using correlation analysis





****

Correlation value of each column is vey less and so the have vey less correlation and hence all can be selected as feature selection step

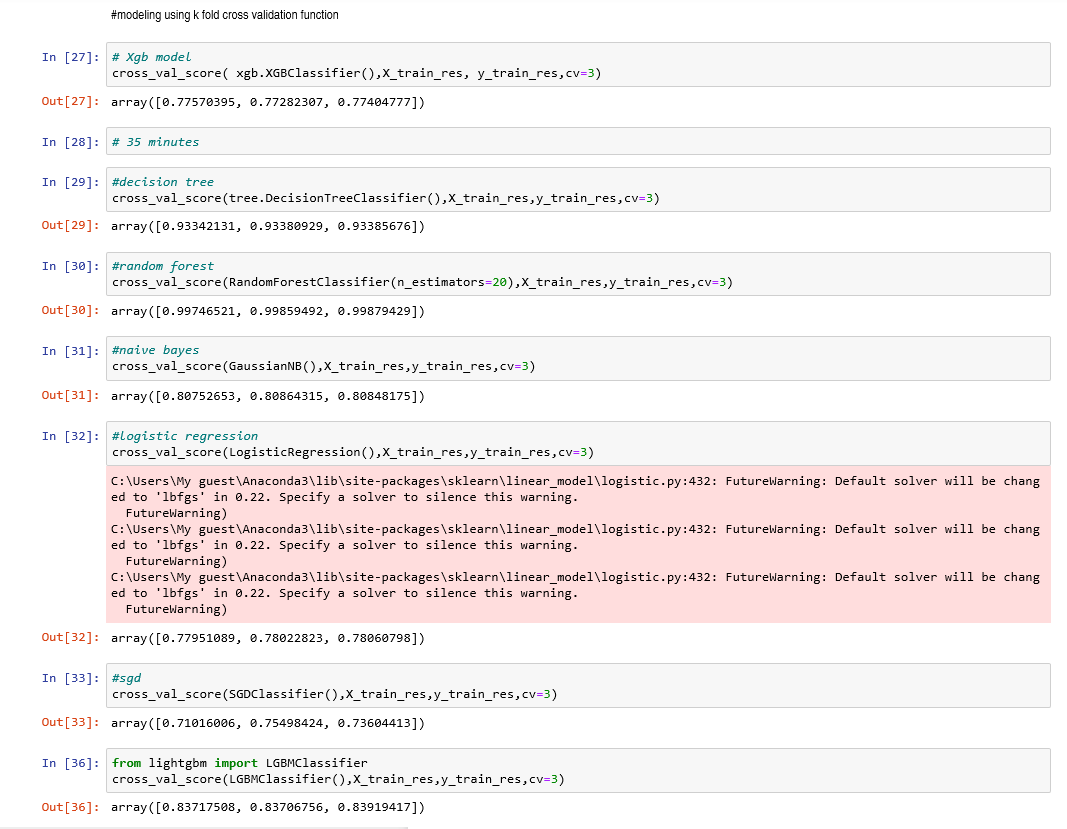
* 1. **Feature engineering**

There is no information about the given features so one can not perform the feature engineering

**Chapter 4 Modelling**

Model implementation using k fold cross validations function for following machine learning algorithms:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Machine learning algorithm** | **Accuracy** | **Execution time** |
| 1 | XGBClassifier | 77% | 35 min |
| 2 | Decision tree | 94% | 15-20 min |
| 3 | Random forest | 99% | 5-10 min |
| 4 | Naïve bayes | 80% | 30 sec |
| 5 | Logistic regression | 78% | 60 min |
| 6 | SGD Classifier | 73% | 45 min |
| 7 | Lightgbm Classifier | 84% | 3 min |
| 8 | SVC | -- | Large execution time |
| 9 | Kneighbourclassifier | -- | Large execution time |

****

So from above results and keeping accuracy and time of execution taking in mind

Three models are selected for further model deployment

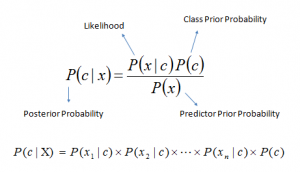
1. Naive Bayes algorithm

It is a [classification technique](https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2/?utm_source=blog&utm_medium=6stepsnaivebayesarticle) based on Bayes’ Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as ‘Naive’.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:

[](https://www.analyticsvidhya.com/wp-content/uploads/2015/09/Bayes_rule-300x172.png)Above,

P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).

P(c) is the prior probability of class.

P(x|c) is the likelihood which is the probability of predictor given class.

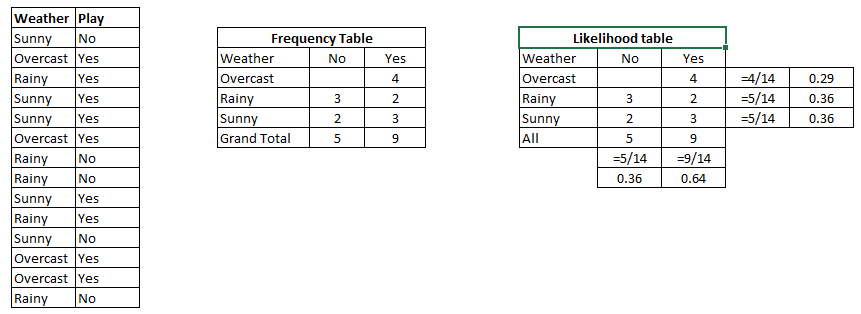
P(x) is the prior probability of predictor.

How Naive Bayes algorithm works?

Let’s understand it using an example. Below I have a training data set of weather and corresponding target variable ‘Play’ (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let’s follow the below steps to perform it.

Step 1: Convert the data set into a frequency table

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

[](https://www.analyticsvidhya.com/wp-content/uploads/2015/08/Bayes_41.png)

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Problem: Players will play if weather is sunny. Is this statement is correct?

We can solve it using above discussed method of posterior probability.

P(Yes | Sunny) = P( Sunny | Yes) \* P(Yes) / P (Sunny)

Here we have P (Sunny |Yes) = 3/9 = 0.33, P(Sunny) = 5/14 = 0.36, P( Yes)= 9/14 = 0.64

Now, P (Yes | Sunny) = 0.33 \* 0.64 / 0.36 = 0.60, which has higher probability.

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.

What are the Pros and Cons of Naive Bayes?

Pros:

It is easy and fast to predict class of test data set. It also perform well in multi class prediction

When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.

It perform well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Cons:

If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.

On the other side naive Bayes is also known as a bad estimator, so the probability outputs from predict\_proba are not to be taken too seriously.

Another limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent.

4 Applications of Naive Bayes Algorithms

Real time Prediction: Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.

Multi class Prediction: This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.

Text classification/ Spam Filtering/ Sentiment Analysis: Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)

Recommendation System: Naive Bayes Classifier and [Collaborative Filtering](https://en.wikipedia.org/wiki/Collaborative_filtering) together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not

1. Random forest

Random forests is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance. Random forests has a variety of applications, such as recommendation engines, image classification and feature selection. It can be used to classify loyal loan applicants, identify fraudulent activity and predict diseases. It lies at the base of the Boruta algorithm, which selects important features in a dataset.

The Random Forests Algorithm

Let’s understand the algorithm in layman’s terms. Suppose you want to go on a trip and you would like to travel to a place which you will enjoy.

So what do you do to find a place that you will like? You can search online, read reviews on travel blogs and portals, or you can also ask your friends.

Let’s suppose you have decided to ask your friends, and talked with them about their past travel experience to various places. You will get some recommendations from every friend. Now you have to make a list of those recommended places. Then, you ask them to vote (or select one best place for the trip) from the list of recommended places you made. The place with the highest number of votes will be your final choice for the trip.

In the above decision process, there are two parts. First, asking your friends about their individual travel experience and getting one recommendation out of multiple places they have visited. This part is like using the decision tree algorithm. Here, each friend makes a selection of the places he or she has visited so far.

The second part, after collecting all the recommendations, is the voting procedure for selecting the best place in the list of recommendations. This whole process of getting recommendations from friends and voting on them to find the best place is known as the random forests algorithm.

It technically is an ensemble method (based on the divide-and-conquer approach) of decision trees generated on a randomly split dataset. This collection of decision tree classifiers is also known as the forest. The individual decision trees are generated using an attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result. In the case of regression, the average of all the tree outputs is considered as the final result. It is simpler and more powerful compared to the other non-linear classification algorithms.

How does the algorithm work?

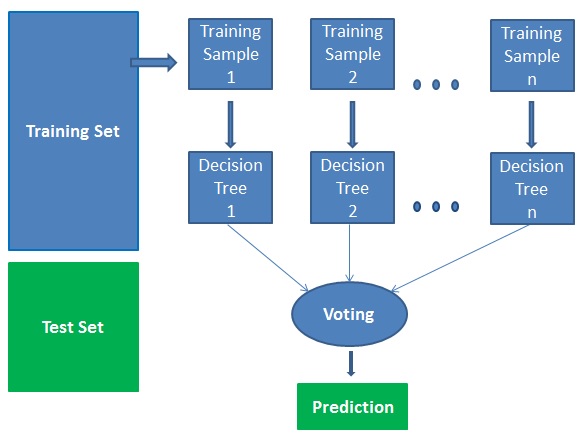
It works in four steps:

Select random samples from a given dataset.

Construct a decision tree for each sample and get a prediction result from each decision tree.

Perform a vote for each predicted result.

Select the prediction result with the most votes as the final prediction.



Advantages:

Random forests is considered as a highly accurate and robust method because of the number of decision trees participating in the process.

It does not suffer from the overfitting problem. The main reason is that it takes the average of all the predictions, which cancels out the biases.

The algorithm can be used in both classification and regression problems.

Random forests can also handle missing values. There are two ways to handle these: using median values to replace continuous variables, and computing the proximity-weighted average of missing values.

You can get the relative feature importance, which helps in selecting the most contributing features for the classifier.

Disadvantages:

Random forests is slow in generating predictions because it has multiple decision trees. Whenever it makes a prediction, all the trees in the forest have to make a prediction for the same given input and then perform voting on it. This whole process is time-consuming.

The model is difficult to interpret compared to a decision tree, where you can easily make a decision by following the path in the tree.

Finding important features

Random forests also offers a good feature selection indicator. Scikit-learn provides an extra variable with the model, which shows the relative importance or contribution of each feature in the prediction. It automatically computes the relevance score of each feature in the training phase. Then it scales the relevance down so that the sum of all scores is 1.

This score will help you choose the most important features and drop the least important ones for model building.

Random forest uses gini importance or mean decrease in impurity (MDI) to calculate the importance of each feature. Gini importance is also known as the total decrease in node impurity. This is how much the model fit or accuracy decreases when you drop a variable. The larger the decrease, the more significant the variable is. Here, the mean decrease is a significant parameter for variable selection. The Gini index can describe the overall explanatory power of the variables.

Random Forests vs Decision Trees

Random forests is a set of multiple decision trees.

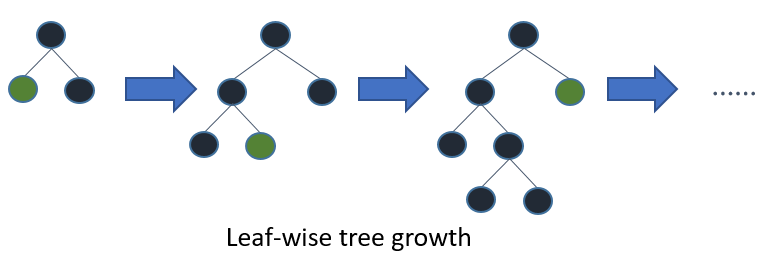
Deep decision trees may suffer from overfitting, but random forests prevents overfitting by creating trees on random subsets.

Decision trees are computationally faster.

Random forests is difficult to interpret, while a decision tree is easily interpretable and can be converted to rules.

1. Light GBM:-

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks. Since it is based on decision tree algorithms, it splits the tree leaf wise with the best fit whereas other boosting algorithms split the tree depth wise or level wise rather than leaf-wise. So when growing on the same leaf in Light GBM, the leaf-wise algorithm can reduce more loss than the level-wise algorithm and hence results in much better accuracy which can rarely be achieved by any of the existing boosting algorithms. Also, it is surprisingly very fast, hence the word ‘Light’. Before is a diagrammatic representation by the makers of the Light GBM to explain the difference clearly.



Leaf wise tree growth in Light GBM.

Leaf wise splits lead to increase in complexity and may lead to overfitting and it can be overcome by specifying another parameter max-depth which specifies the depth to which splitting will occur.

Below, we will see the steps to install Light GBM and run a model using it. We will be comparing the results with XGBOOST results to prove that you should take Light GBM in a ‘LIGHT MANNER’.

Let us look at some of the advantages of Light GBM.

2. Advantages of Light GBM

Faster training speed and higher efficiency: Light GBM use histogram based algorithm i.e it buckets continuous feature values into discrete bins which fasten the training procedure.

Lower memory usage: Replaces continuous values to discrete bins which result in lower memory usage.

Better accuracy than any other boosting algorithm: It produces much more complex trees by following leaf wise split approach rather than a level-wise approach which is the main factor in achieving higher accuracy. However, it can sometimes lead to overfitting which can be avoided by setting the max\_depth parameter.

Compatibility with Large Datasets: It is capable of performing equally good with large datasets with a significant reduction in training time as compared to XGBOOST.Parallel learning supported.

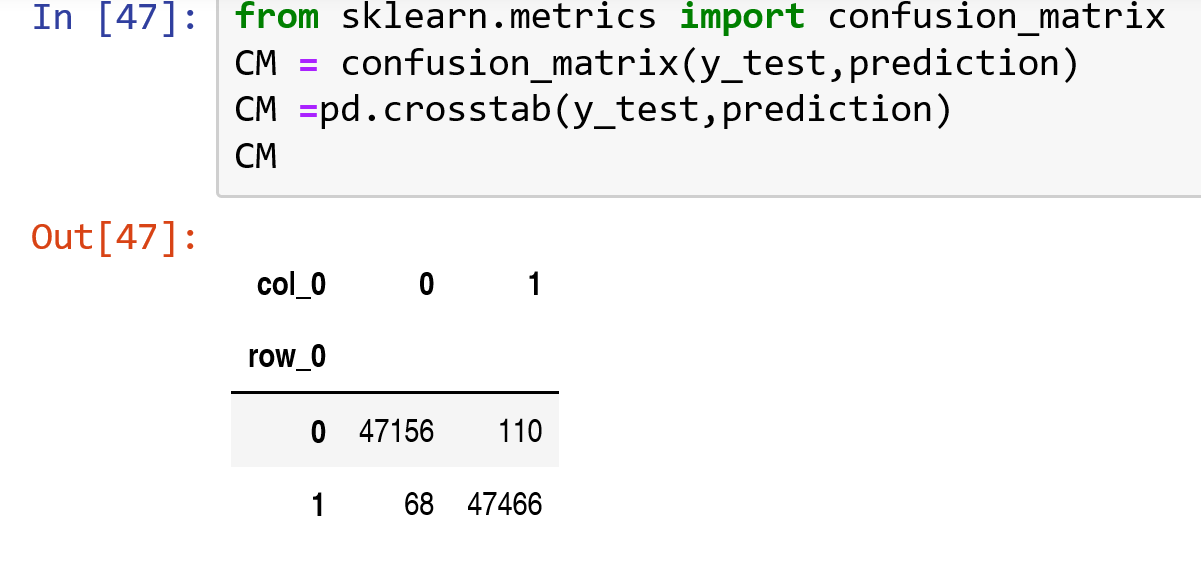
**Chapter 5 Model validation**

**Model evaluation and selection**

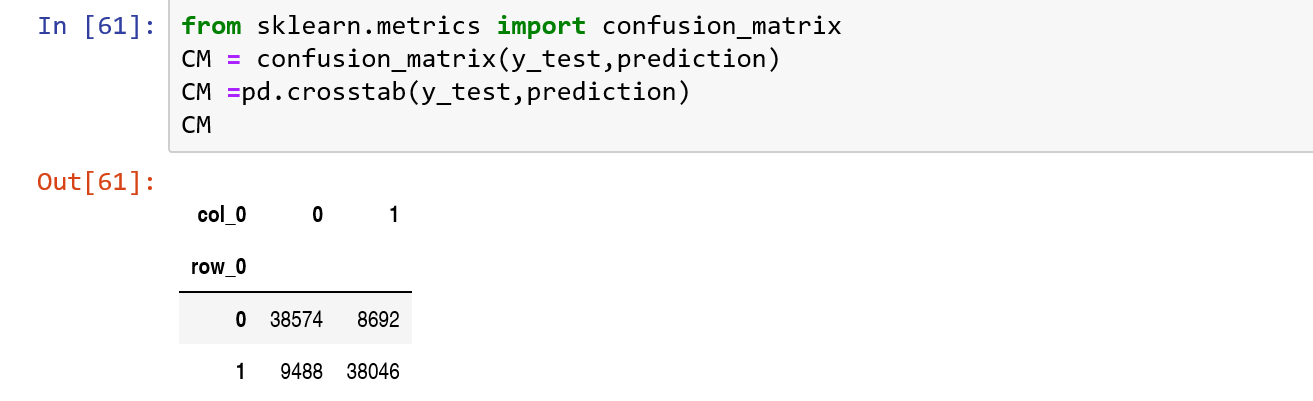
|  |  |  |
| --- | --- | --- |
| **Model** | **accuracy** | **AUC** |
| Random forest | **90%** | **.505** |
| Naïve Bayes | **81.6%** | **.806** |
| Light gbm | **90.4%** | **.720** |

**Since the auc of naives bayes is best among all so I have selected random forest as my final model**

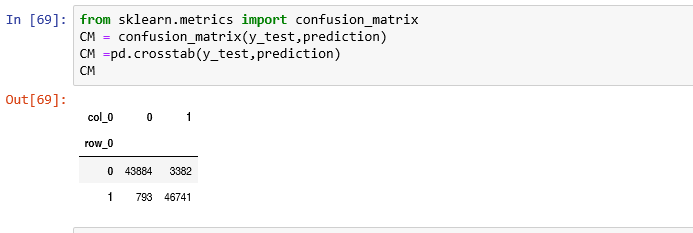
1. Confusion matrix
2. For random forest



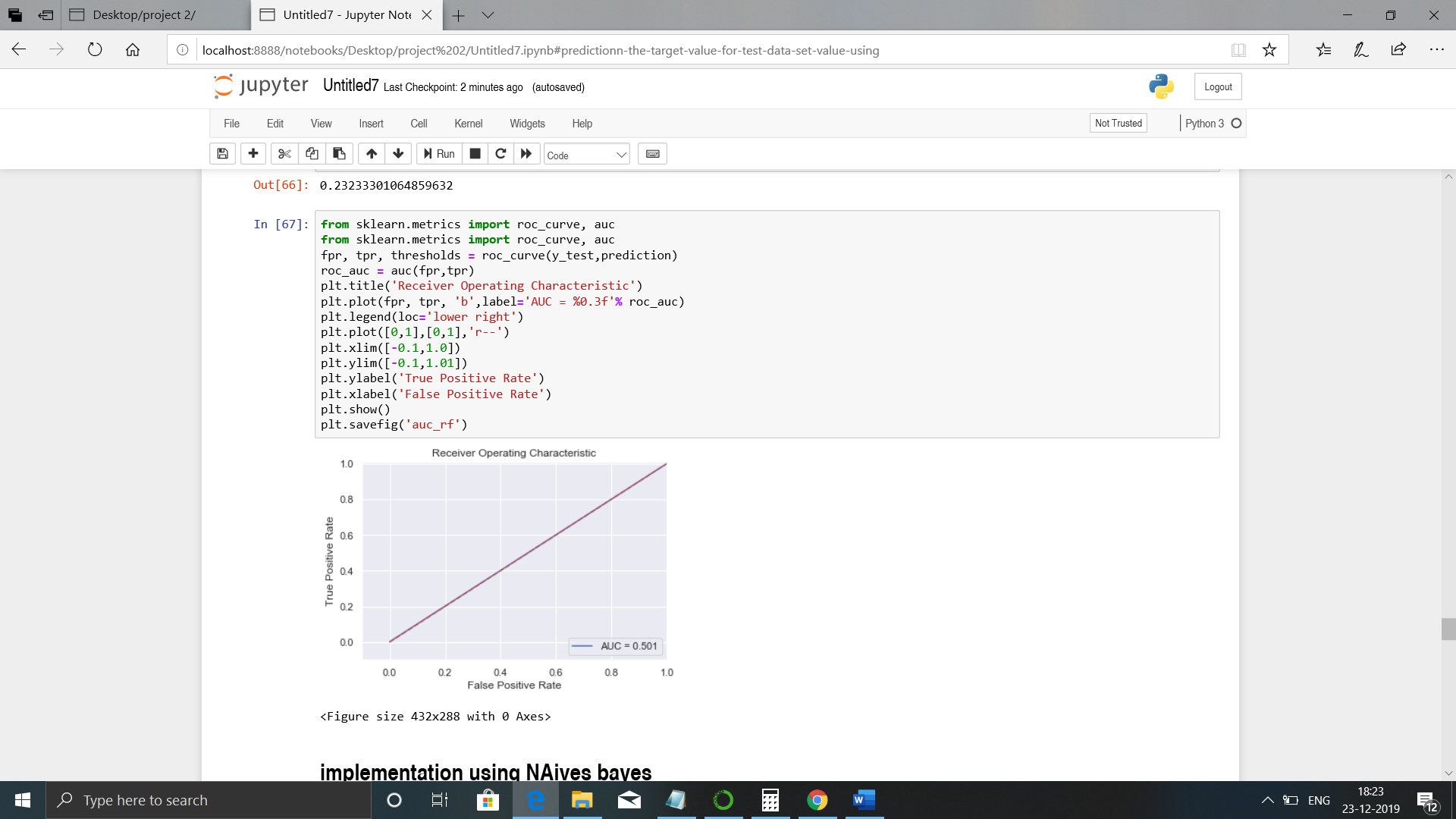
1. For naives bayes



1. For lightgbm



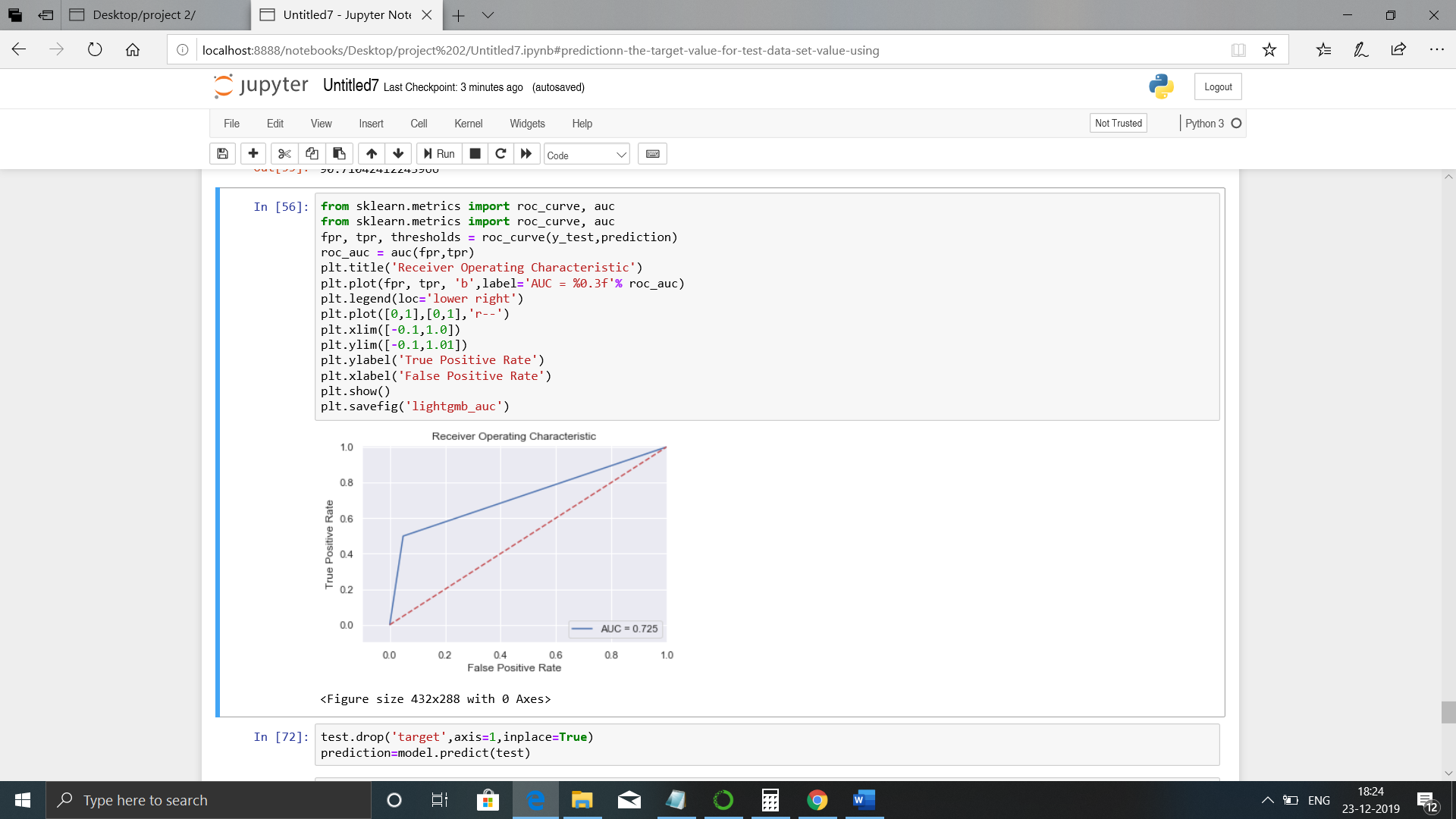
1. Auc curve
2. For random forest



1. For naives bayes



1. For lightgbm



**Chapter 6**

**Instructions to deploy and run code.**

1. **Python instructions**
2. Open anaconda
3. Open jupyter notebook
4. Go to the file location and open finalcab.py file
5. Now open new python console
6. And copy the code from .py to console and
7. Import the data set in console using your system file location
8. Go to cell and click on run all cell at a time
9. The complete python file will be executed without inturuption
10. And result will be displayed of the test file
11. **R instructions**
12. Open r studio
13. Go to file loction and open mo.r file
14. First run first few lines to import the data set in r console using your system file loction
15. Then run complete file code lines one by one
16. The complete project will be executed without inturuption
17. The result will be displayed in console
18. Visualisations will be displayed in plots section

**References**

1. For Data Cleaning and Model Development - <https://edwisor.com/career-datascientist>

2. For other code related queries - [https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principalcomponent-analysis- python/](https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principalcomponent-analysis-%20python/)

3. For Visualization – <https://www.udemy.com/python-for-data-science-andmachine-learning-bootcamp/>

4. <https://towardsdatascience.com/>

5. https://stackoverflow.com/