**Zomato Restaurant Project**

**Problem Statement:**

Zomato Data Analysis is one of the most useful analysis for foodies who want to taste the best cuisines of every part of the world which lies in their budget. This analysis is also for those who want to find the value for money restaurants in various parts of the country for the cuisines.

Additionally, this analysis caters the needs of people who are striving to get the best cuisine of the country and which locality of that country serves that cuisines with maximum number of restaurants..

**Data Storage:**

This problem statement contains two datasets- Zomato.csv and country\_code.csv.

Country\_code.csv contains two variables:

 Country code

 Country name

The collected data has been stored in the Comma Separated Value file Zomato.csv. Each

restaurant in the dataset is uniquely identified by its Restaurant Id. Every Restaurant contains the

following variables:

• Restaurant Id: Unique id of every restaurant across various cities of the world

• Restaurant Name: Name of the restaurant

• Country Code: Country in which restaurant is located

• City: City in which restaurant is located

• Address: Address of the restaurant

• Locality: Location in the city

• Locality Verbose: Detailed description of the locality

• Longitude: Longitude coordinate of the restaurant&#39;s location

• Latitude: Latitude coordinate of the restaurant&#39;s location

• Cuisines: Cuisines offered by the restaurant

• Average Cost for two: Cost for two people in different currencies ��

• Currency: Currency of the country

• Has Table booking: yes/no

• Has Online delivery: yes/ no

• Is delivering: yes/ no

• Switch to order menu: yes/no

• Price range: range of price of food

• Aggregate Rating: Average rating out of 5

• Rating color: depending upon the average rating color

• Rating text: text on the basis of rating of rating

• Votes: Number of ratings casted by people

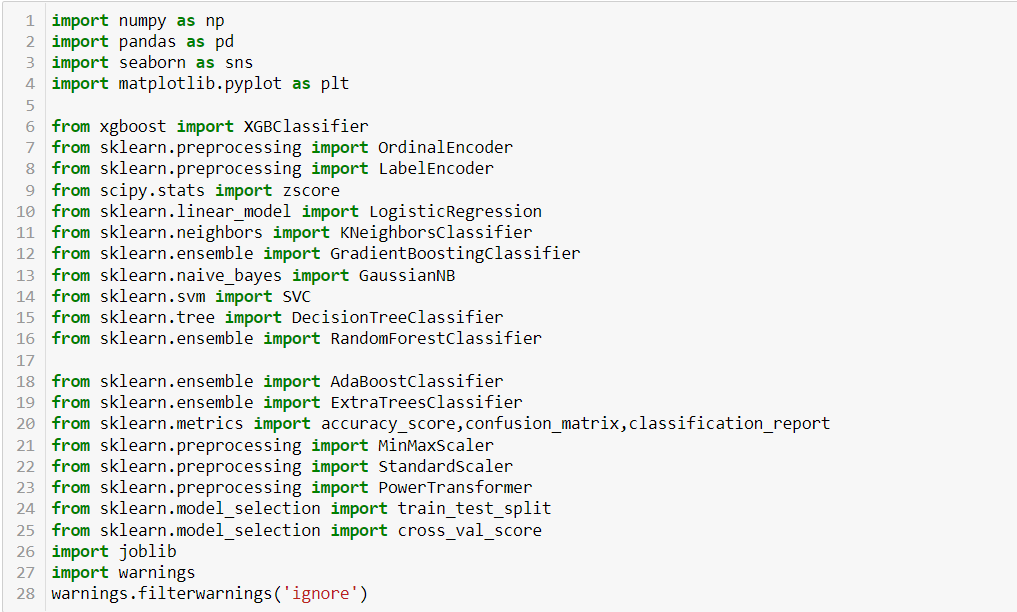
**Label\_column/Target =**  **Price range**

Problem statement : In this dataset predict **Price range**

**Data Analysis:**

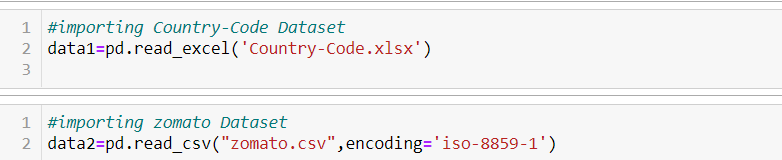
* **Import all required libraries**

We will be requiring NumPy and pandas for mathematical computation and data manipulation. And matplotlib with seaborn to visualize the data with interactive plots. Also, import preprocessing tools and models to deal with pre-processing and model building. We have imported libraries for categorical data as Target value is of different price range. Import warnings to ignore all the warnings.

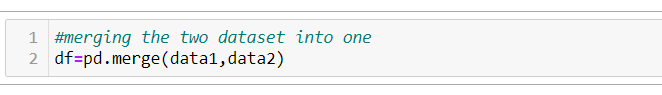
****

* **Load Dataset**

The dataset is in csv format so use the pandas read\_csv method to load the dataset. After loading just see its shape and head of the dataset to have a near view of the data we have.

****

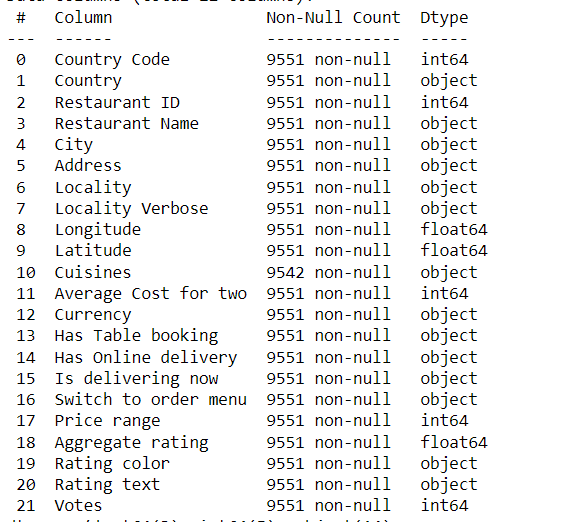
* **Merging the two dataset**

****

* **Checking Datatypes**

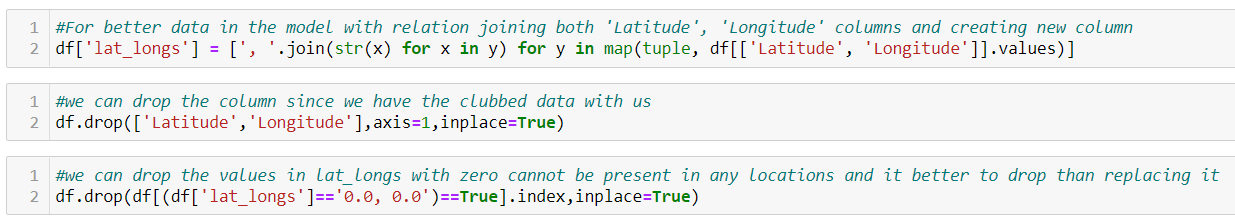
We have int and object datatypes. Label column that is Average Cost for two is of int64.

Country, 'Restaurant Name', 'City', 'Address', 'Locality', 'Locality Verbose', 'Cuisines','Currency', 'Has Table booking', 'Has Online delivery', 'Is delivering now', 'Switch to order menu', 'Rating color', 'Rating text' are of object type. We have to check on this data and remove/encode as per need.



* **Dropping unwanted columns**

Checked for unwanted data. We found that 3 columns which are found to be unwanted the entire dataset. We dropped Country Code, Latitude,Longitude columns.



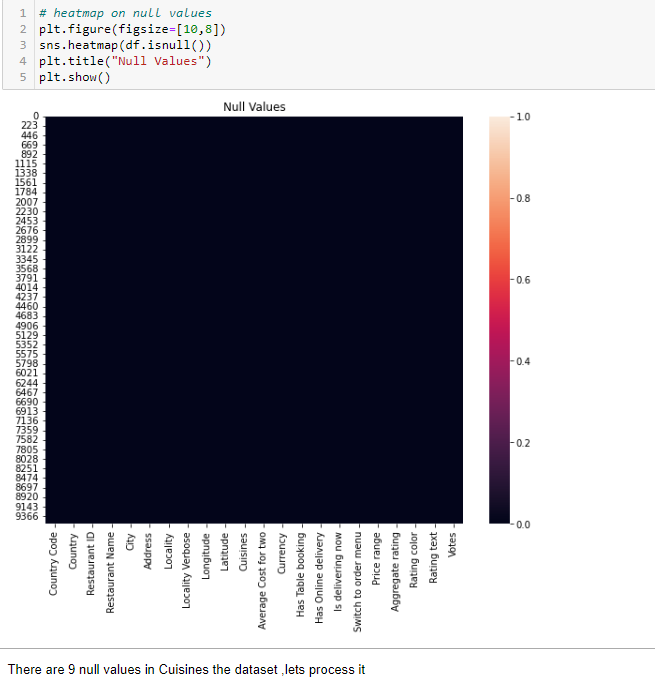
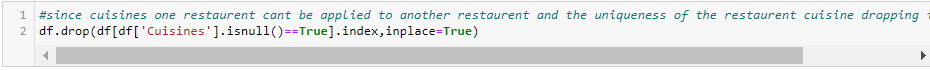


Now we have 1470 rows and 32 columns for Pre-Processing.

**Pre-Processing Pipeline**

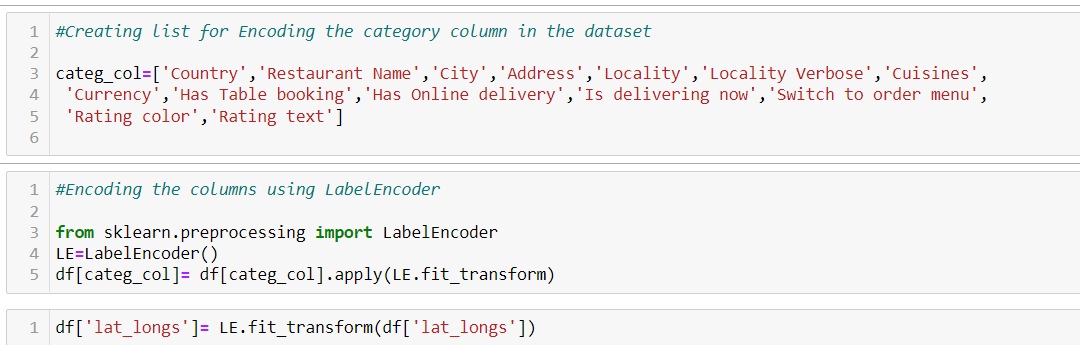
* **Missing Values:**

There are no Missing values/ Null values found in the dataset. The Presence of missing values can be found using data.isnull().

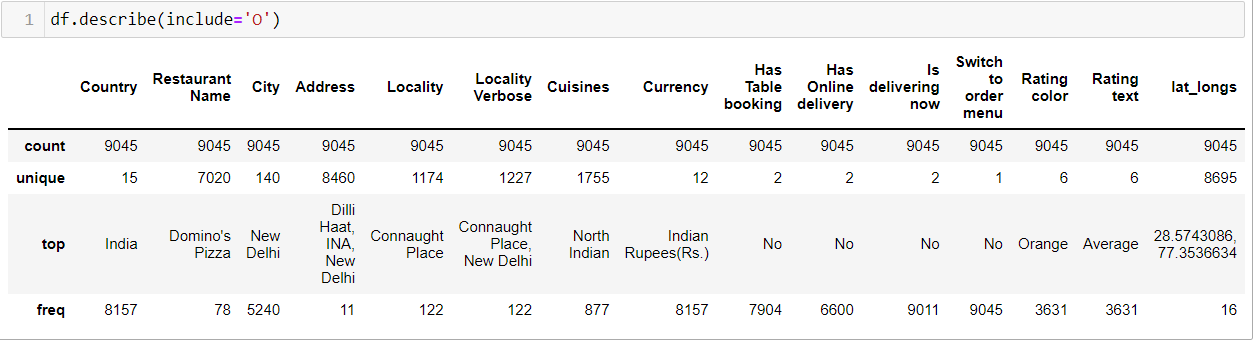
* **Encoding data**

Since we can see a lot of columns are of object data type. we need to encode using label encoder for feature columns. We have identified columns to be encoded as shown in the figure.



* **Statistical summary**

The statistical view of data is found from this summary. We can get the statistical view using data.describe() . The count is same in all data. There are possible marginal outliers in some of the columns. The mean and median look like in range apart from categorical data. The standard deviation of all data is seen clearly. We can further check for skewness and outliers through visualization and can treat efficiently. 



#We can observe india is most data in the dataset

#Domino's Pizza has the highest freq of 78 in the dataset

#Connaught Place has the highest freq in loaclity

#north indaian and Indian Rupees has the highest freq in dataset

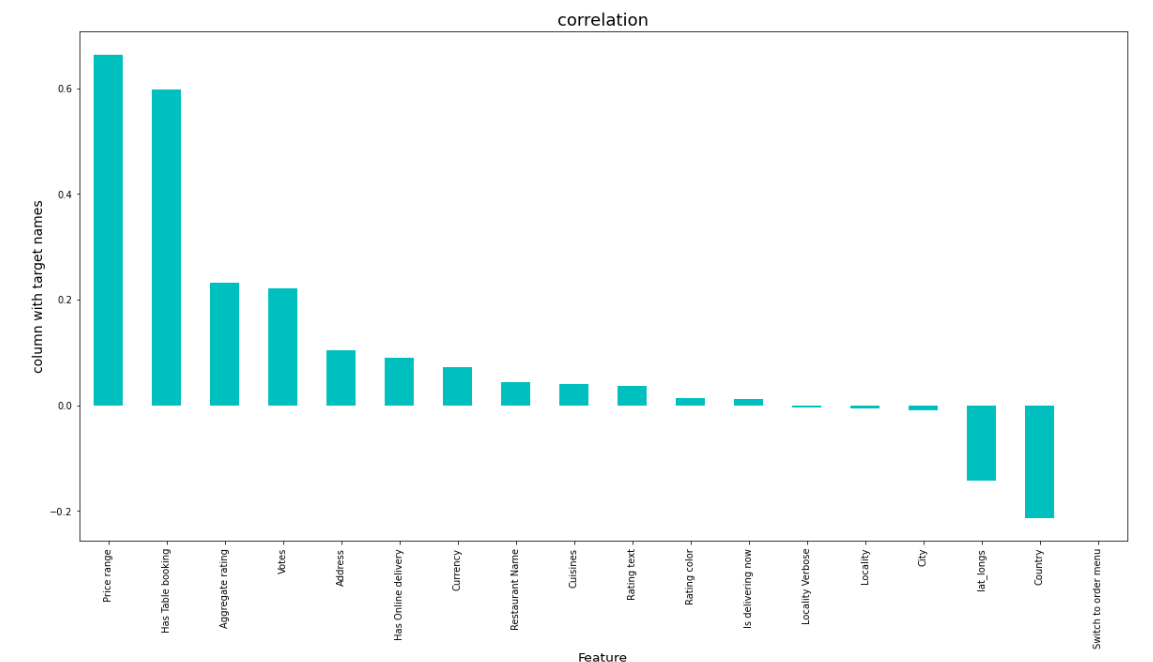
#high freq for No classification in the datset

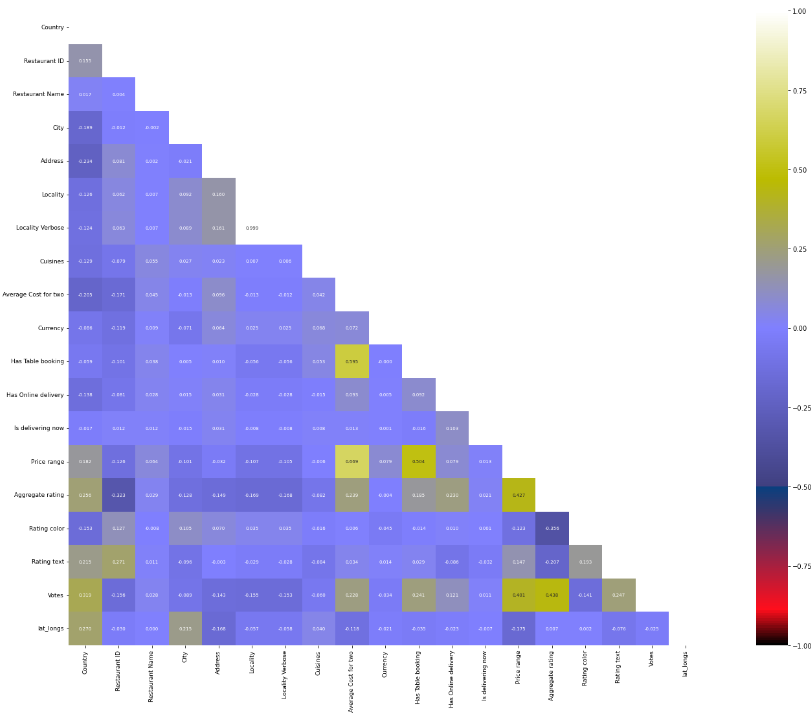
#Highest freq for Average rating color in the dataset

#no outliers present in the dataset

# Correlation Check

Correlation between each column can be viewed using corr(). Locality,Locality verbose are less correlated which can be removed if needed. Multicollinearity possibility is there among columns which we can check and remove if needed.Since Data is is precious lets keep rest of the data and drop the columns further after evaluating the models overfitting/underfiiting Accordingly



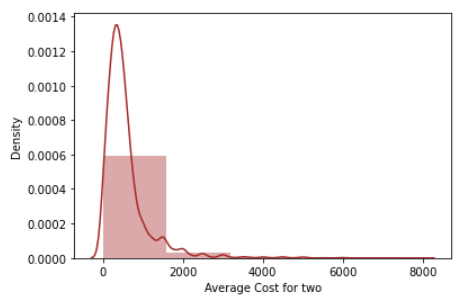
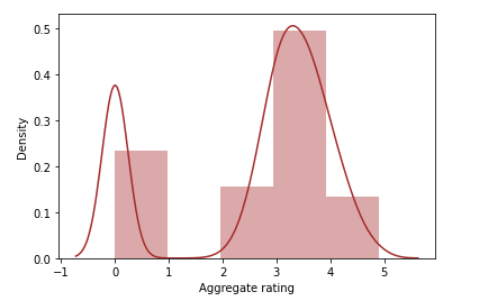


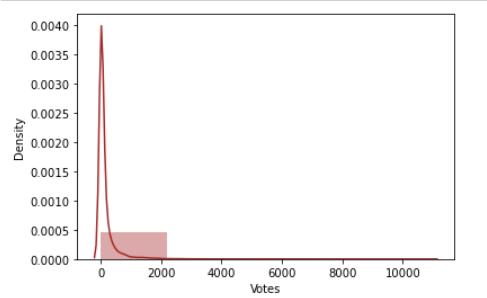
# Distribution of data:

# Skewness

We can identify skewness using distplot. We can interpret whether the data is normally distributed, rightly skewed or left skewed plotting distplot graph.

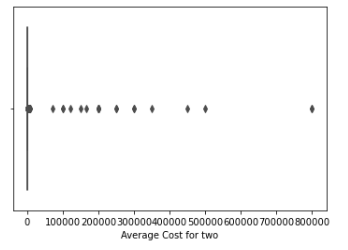
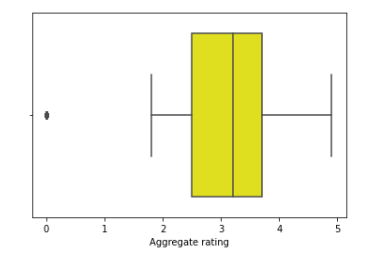
Here Average Cost for two,Aggregate rating,Votes are skewed .

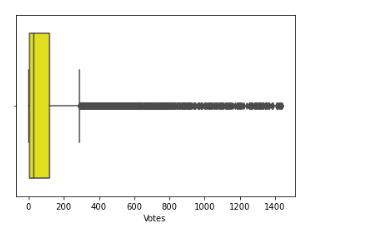
 



* **Detecting Outliers:**

We can detect outliers using boxplot. It is found that there are outliers found in some of the columns like Average Cost for two,Aggregate rating,Votes.



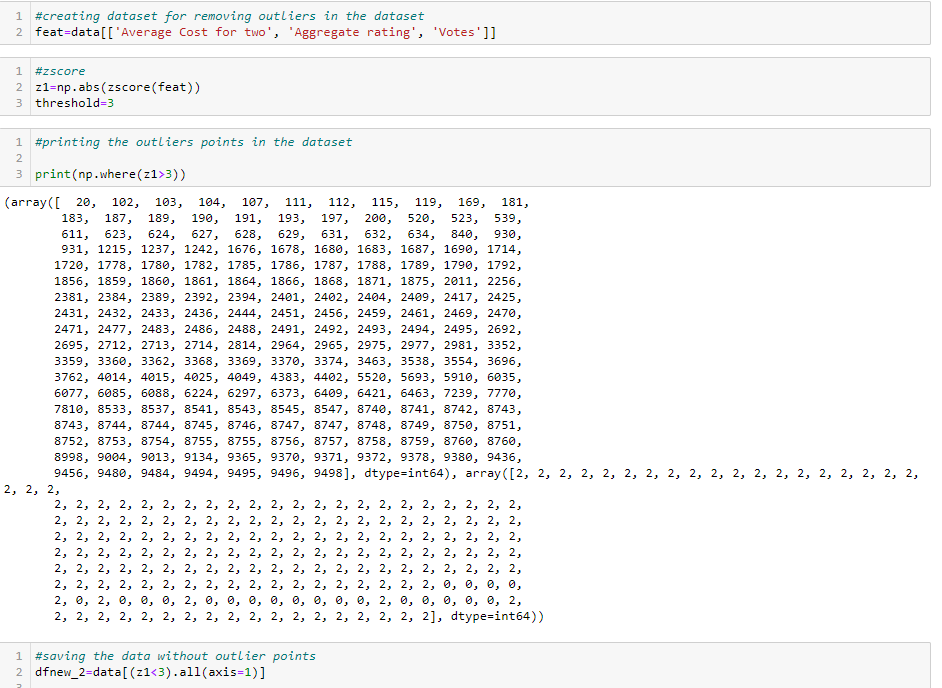
* **Distribution relationship:**

we can see here how the range of data in each column dominates the attrition data. This can be achieved using scatterplot. It is found that each data has either positive or negative relationship with the output data. The correlation value is shown using corr()..

We have identified skewness and outliers in the dataset. We can clean the data now

* **Data Cleaning:**
* **Removing Outliers:**

We are using z-score technique to remove outliers from the entire dataset



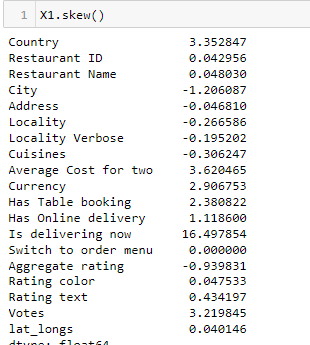
After removing outliers, it is found that we have a data loss of around 5 %.

The new dataset consists of 4850 rows and 20 columns.

We have removed outliers from entire dataset. Now, data is clean with no outliers but we have to check skewness and correct it such the data become efficient for model building. We need target variable to be balanced for better result.

* **Skewness Removal:**

As we have skewness present in many of the columns ignoring categorical columns, we can use power transformer to the entire dataset such that every data will be transformed efficiently. As we need to apply skewness only for feature columns we can classify the data into x and y where x is the feature columns and y is the target column. Skewness value can be determined using skew(). Power Transformer removes the skewness for the entire dataset and keeps the columns skewness value in acceptable range. We also found that Average Cost for two,Aggregate rating,Votes has high.

****

We can see that skewness are corrected and are in range.

The data is ready for model building but we have to check multi collinearity before proceeding.

* **Multicollinear column Removal:**

This helps in removing columns that have same importance to the output to be predicted. If we have variance inflation factor less than 5, it is considered to be a good column without multicollinearity. Locality and Locality Verbose are multicorrelated. Dropping Locality column as it has lowest p value.

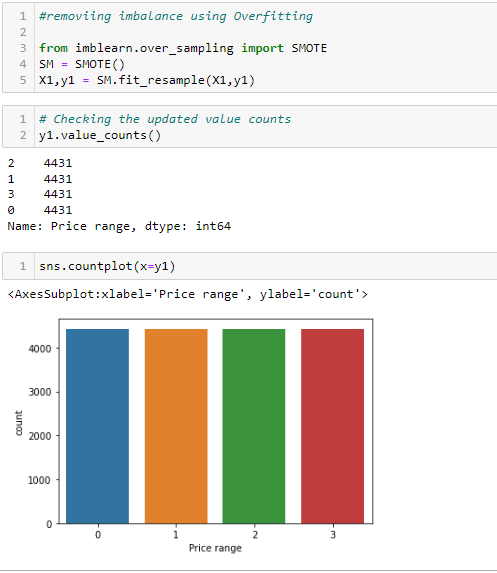
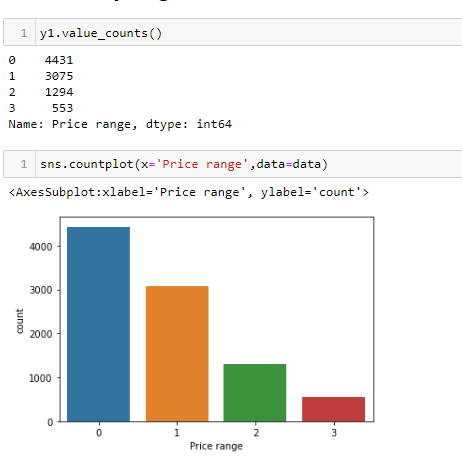
****

We can see all the values below 5. That means multicollinearity of the columns removed.

* **Imbalanced to balanced:**

Taking countplot of target variable, it is found that the data is imbalanced. We can remove the imbalancing by Oversampling using imblearn.over\_sampling technique.

After resampling we got new dataset of 17,724 rows and 19 columns where Price Range column is balanced with equal number of data.



we have equal number of data for efficient model building.

***EDA CONCLUDING REMARKS:***

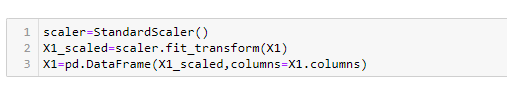
Through EDA we were able to analyse, visualize and clean data for model building, EDA helped in removing outliers and correcting skewness of the entire dataset. We have dropped useless data columns. It also helped in ensuring there is no multi correlation between columns. We were able to clean data efficiently dropping columns that may affect the performance of the model.

After analysing, visualization, computing and cleaning data. The dataset is ready for model building with 17724 rows and 19 columns including target variable.

***Building Machine Learning Models***

* Initializing x and y separately, where x contains of dataset of without target variable that is only feature\_columns and y contains only target variable data.
* **Scaling:**

We can scale the data using Standard Scaler before splitting it into train and predict dataset such that we have scaled data which will make easy for machine to learn . This is specifically done when we have different data type in a dataset.



* **Best Random\_State and train\_test\_split**

We can find best random state first such that we can apply that while splitting the dataset into training and prediction phase. As it is of binary output we can use logistic regression to find the best random state for the model as shown in the figure.

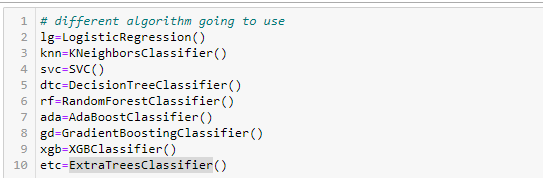


We can find random state as 101 with 74 % accuracy score which we will be using in train and test as shown. The random state is chosen from range 1-200

* **Different Algorithm used:**

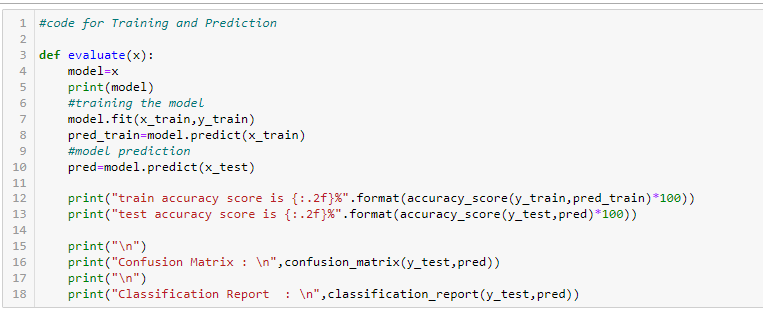
We have used different algorithm to determine the best model for the dataset. The algorithm used in this model are:

* Logistic Regression
* KNeighborsClassifier
* SVC
* DecisionTreeClassifier
* RandomForestClassifier
* AdaBoostClassifier
* GradientBoostingClassifier
* XGBClassifier
* ExtraTreesClassifier



* **Metrics:**

The metrics like accuracy score, confusion matrix and classification report are calculated for each algorithm and chosen the best algorithm for final model. The metrics can be found by,

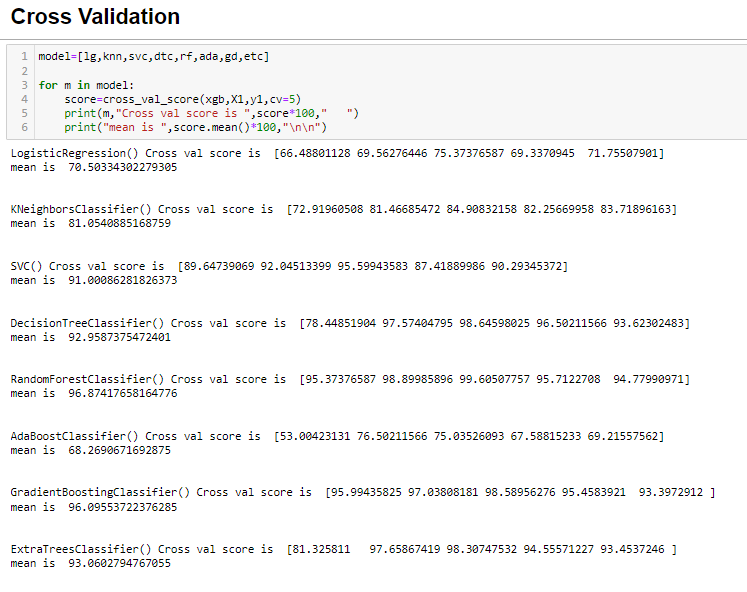
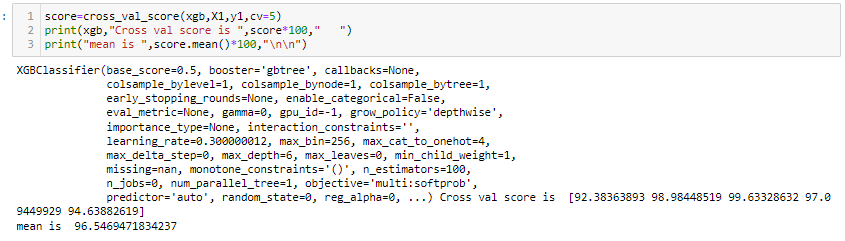


Using this code, we can do training and prediction phase for each algorithm and can find the score through which we can have insights on how good each algorithm performs after data feeding.

* **Accuracy Scores**

The accuracy score of each algorithm is as follows:

|  |  |
| --- | --- |
| **Algorithm** | **Score** |
| * Logistic Regression | 74.50 |
| * KNeighborsClassifier | 85.71 |
| * SVC | 92.78 |
| * DecisionTreeClassifier | 97.89 |
| * RandomForestClassifier | 98.93 |
| * AdaBoostClassifier | 73.82 |
| * GradientBoostingClassifier | 97.39 |
| * XGBClassifier | 99.00 |
| * ExtraTreesClassifier | 97.84 |

* We can see that XGBClassifier and RandomForestClassifier gives the top 2 score for the model. But we have to cross check with cross validation score to finalize the model.
* **Cross Validation Score** 

The minimum value difference between the accuracy score and cross validation score gives us the best model. By comparing both cross validation accuracy and actual accuracy. It is found that Support Vector Classifier is the best model with a difference of ~4%

# Support Vector classifier model is the best model with 92.41% accuracy

We will try to improve accuracy by hyper tuning.

* **Hyper Tuning – GridSearchCV**

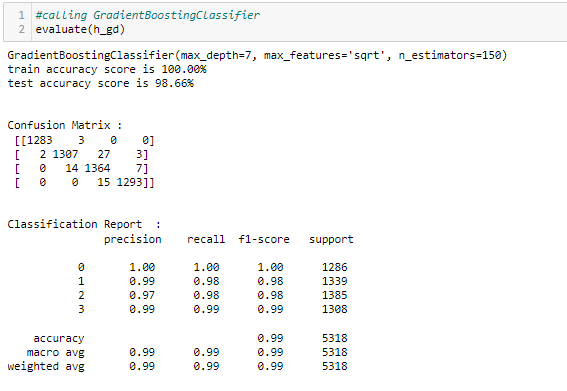
We can use different parameters for GradientBoostingClassifier such as 'max\_features':['auto','sqrt','log2'],

"n\_estimators":[50,100,150,200,250,300],

"max\_depth":[3,5,7,9,11,13], "learning\_rate":[0.001,0.01,0.1,1]

such that we try to increase the accuracy score.





* ***Concluding Remarks***

The hyper parameter tuning of Gradient Boosting Classifier gives actual accuracy of 97.39 % and cross validation score of 96.09 %. Since there is an accuracy increase of 1% for the hyper tuned parameter, the hyper tuned model of Gradient Boosting Classifier algorithm is selected for final output. Since Hyper parameter tuning taking a lot of time, it is performed only for top model to see whether it is improving the accuracy.

# Gradient Boosting Classifier hypertuned model with true accuracy  98.66% is selected as final model for execution

# 

* You can see how we have handled numerical and categorical data and also how we build different machine learning models on the same dataset.
* Using hyper parameter tuning we can improve our model accuracy,.
* Using this machine Learning Model we people can easily predict the Price Range of Restaurents.