ZOMATO RESTAURANT RATINGS

Main Objective:

1. Get intuition about the data.
2. Import Zomato Restaurant Ratings csv file and perform EDA (Exploratory Data Analysis).
3. Point out conclusions in every step of analysis.
4. Use graphical modules (matplotlib and seaborn) to answer questions.
5. Build a machine learning model that helps to predict various restaurant rating based on certain features.

Abstract:

About Zomato Restaurant csv file

Zomato is one of the best online food delivery apps which gives the users the ratings and the reviews on restaurants all over india. These ratings and the Reviews are considered as one of the most important deciding factors which determine how good a restaurant is.

We will therefore use the real time Data set with various features a user would look into regarding a restaurant. We will be considering Bangalore City in this analysis.

Content The basic idea of analysing the Zomato dataset is to get a fair idea about the factors affecting the establishment of different types of restaurants at different places in Bengaluru, aggregate rating of each restaurant, Bengaluru being one such city has more than 12,000 restaurants with restaurants serving dishes from all over the world.

With each day new restaurants opening the industry hasn’t been saturated yet and the demand is increasing day by day. Inspite of increasing demand it however has become difficult for new restaurants to compete with established restaurants. Most of them serving the same food. Bengaluru being an IT capital of India. Most of the people here are dependent mainly on the restaurant food as they don’t have time to cook for themselves.

With such an overwhelming demand of restaurants it has therefore become important to study the demography of a location. What kind of a food is more popular in a locality? Do the entire locality loves vegetarian food. If yes, then is that locality populated by a particular sect of people for eg. Jain, Marwaris, Gujaratis who are mostly vegetarian. This kind of analysis can be done using the data, by studying the factors such as

- Location of the restaurant
- Approx Price of food
- Theme based restaurant or not
- Which locality of that city serves those cuisines with maximum number of restaurants
- The needs of people who are striving to get the best cuisine of the neighbourhood
- Is a particular neighbourhood famous for its own kind of food.

“Just so that you have a good meal the next time you step out”

The data is accurate to that available on the zomato website until 15 March 2019. The data was scraped from Zomato in two phase. After going through the structure of the website I found that for each neighbourhood there are 6-7 category of restaurants viz. Buffet, Cafes, Delivery, Desserts, Dine-out, Drinks & nightlife, Pubs and bars.

Phase I,

In Phase I of extraction only the URL, name and address of the restaurant were extracted which were visible on the front page. The URl's for each of the restaurants on the Zomato were recorded in the csv file so that later the data can be extracted individually for each restaurant. This made the extraction process easier and reduced the extra load on my machine. The data for each neighbourhood and each category can be found here

Phase II,

In Phase II the recorded data for each restaurant and each category was read and data for each restaurant was scraped individually. 15 variables were scraped in this phase. For each of the neighbourhood and for each category their onlineorder, booktable, rate, votes, phone, location, resttype, dishliked, cuisines, approxcost(for two people), reviewslist, menu_item was extracted. See section 5 for more details about the variables.

Acknowledgements The data scraped was entirely for educational purposes only. Note that I don’t claim any copyright for the data. All copyrights for the data is owned by Zomato Media Pvt. Ltd..

Source: Kaggle(

1. EDA (Exploratory Data Analysis) – Step 1

Reading and Exploring the Data

By looking at the dataset, it's possible to see the meaning of each columns of the data. Just to allign the concepts, we have:

URL : Link of restaurant in Zomato website

ADDRESS : Address of restaurant

name : restaurant Name

online_order : Weather restaurant has online ordering or not

book_table : Weather restaurant has table booking or not

rate : overall rating of the restaurant

votes : contains total numbers of ratings

phone : Mobile number of restaurant

location : contains city in which the Restaurant is located rest_type : Restaurant Type

dish_liked : Dishes most liked by the customers

cuisines : Type of food styles prepared

approx_cost : The approx cost for two people

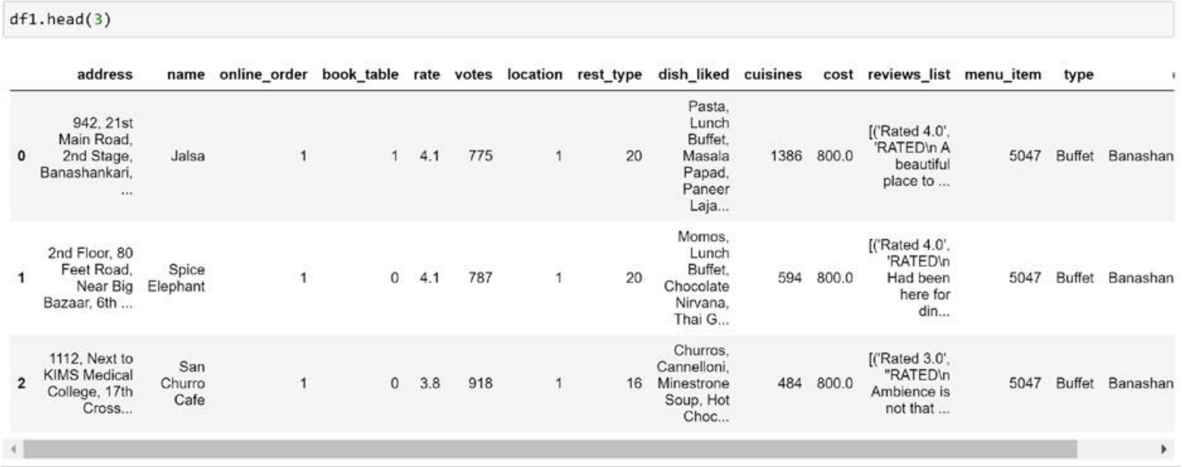
reviews_list : contains list of Reviews for the restaurant

menu_item : contains restaurant food menu list

listed_in : contains the neighbourhood in which the restaurant is listed

Well, after reading the data and taking a look at its characteristics, we can now do a hands on to apply some preparation steps. As we said right above, there are some columns that could be threated better. Let's point in topics we will do right now:

A. approx_cost(for two people):
   Change the data type from object to float, rate:
B. Let's eliminate the "/5" text and change data type from object to float



```
df1.head(3)
```

	address	name	online_order	book_table	rate	votes	location	rest_type	dish_liked	cuisines	cost	reviews_list	menu_item	type	
0	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	1	1	4.1	775	1	20	Pasta, Lunch Buffet, Masala Papad, Paneer Laja...	1386	800.0	[('Rated 4.0', 'RATED\n A beautiful place to ...	5047	Buffet	Banashan
1	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	1	0	4.1	787	1	20	Momos, Lunch Buffet, Chocolate Nirvana, Thai G...	594	800.0	[('Rated 4.0', 'RATED\n Had been here for din...	5047	Buffet	Banashan
2	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	1	0	3.8	918	1	16	Churros, Cannelloni, Minestrone Soup, Hot Choc...	484	800.0	[('Rated 3.0', "RATED\n Ambience is not that ...	5047	Buffet	Banashan

2. Graphical Exploration

In this session, let's present useful charts that could be used to get intuition about the data. We will divide the exploratory data analysis into topics to help the users take their own conclusions. In each topic, we will also try to purpose questions that can be answered with some graphic or table analysis.

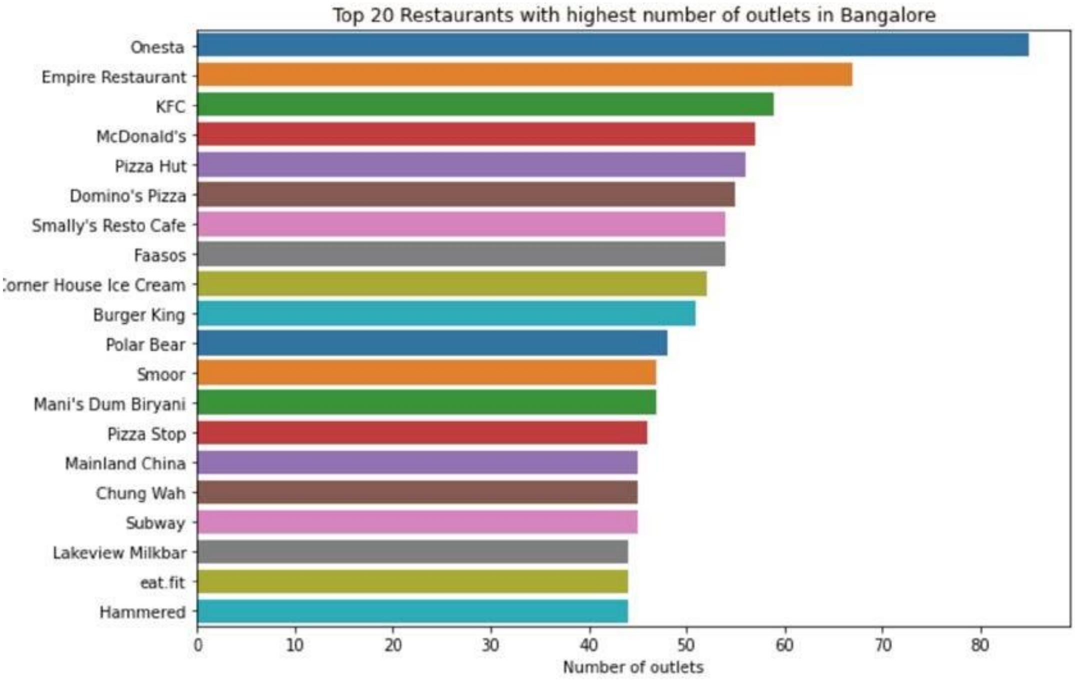
In this topic, let's make an initial analysis on our data looking at restaurant's features presented in it.

2.1 Top 20 Restaurants with highest number of outlets in Bangalore

For answering the question above, let's group our data into some numerical variables that could probably show a good overview from restaurant's indicators like total votes, mean approx cost, rate and others. By the end, we will have in hands a new DataFrame with grouped information about all the restaurant's franchise in the dataset.

It's important to say that here we are grouping by restaurant name and, as long as there are restaurants with the same name in the dataset

(same name but different locations, for example), we will also see how many unities each "franchise" have.

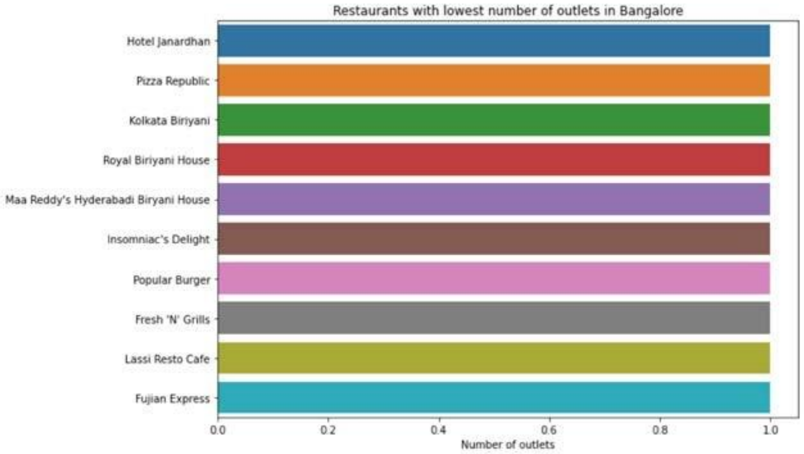


2.1 Top 20 Restaurants with highest number of outlets in Bangalore

2.2 Ten Restaurants with lowest number of outlets in Bangalore

For answering the question above, let's group our data into some numerical variables that could probably show a good overview from restaurant's indicators like total votes, mean approx cost, rate and others. By the end, we will have in hands a new DataFrame with grouped information about all the restaurant's franchise in the dataset.

It's important to say that here we are grouping by restaurant name and, as long as there are restaurants with the same name in the dataset (same name but different locations, for example), we will also see how many unities each "franchise" have. And take the lowest count of the Restaurants.

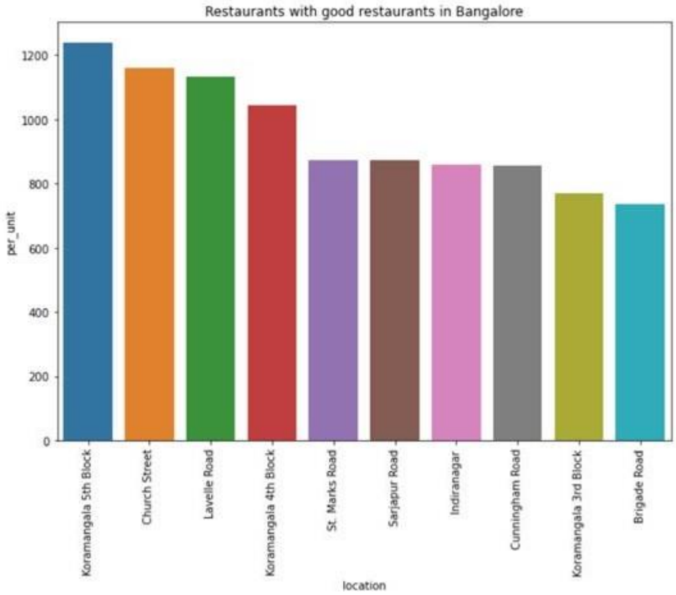


2.2 Ten Restaurants with lowest number of outlets in Bangalore

2.3 Good restaurants in Bangalore according to votes

For answering the question above, let's group our data into some numerical variables that could probably show a good overview from restaurant's indicators like total votes, mean approx cost, rate and others. By the end, we will have in hands a new DataFrame with grouped information about all the restaurant's franchise in the dataset.

It's important to say that here we are grouping by restaurant name and, as long as there are restaurants with 'per unit' in the dataset. we will also see how many votes each "unit" have. And take the best count of votes per each unit of the Restaurants.

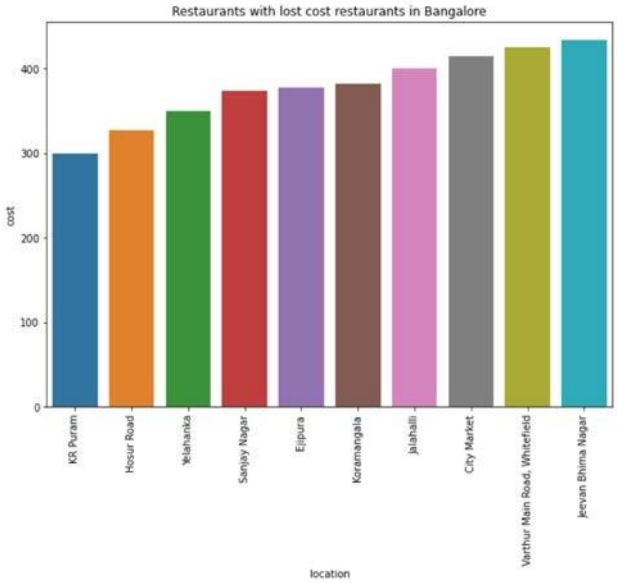


2.3 Good restaurants in Bangalore according to votes

2.4 Check the areas with lowest cost Restaurants for two people to have food

we are grouping by restaurant location and, as long as there are restaurants with ‘cost’ in the dataset. We take the mean cost of restaurants and sort the values.

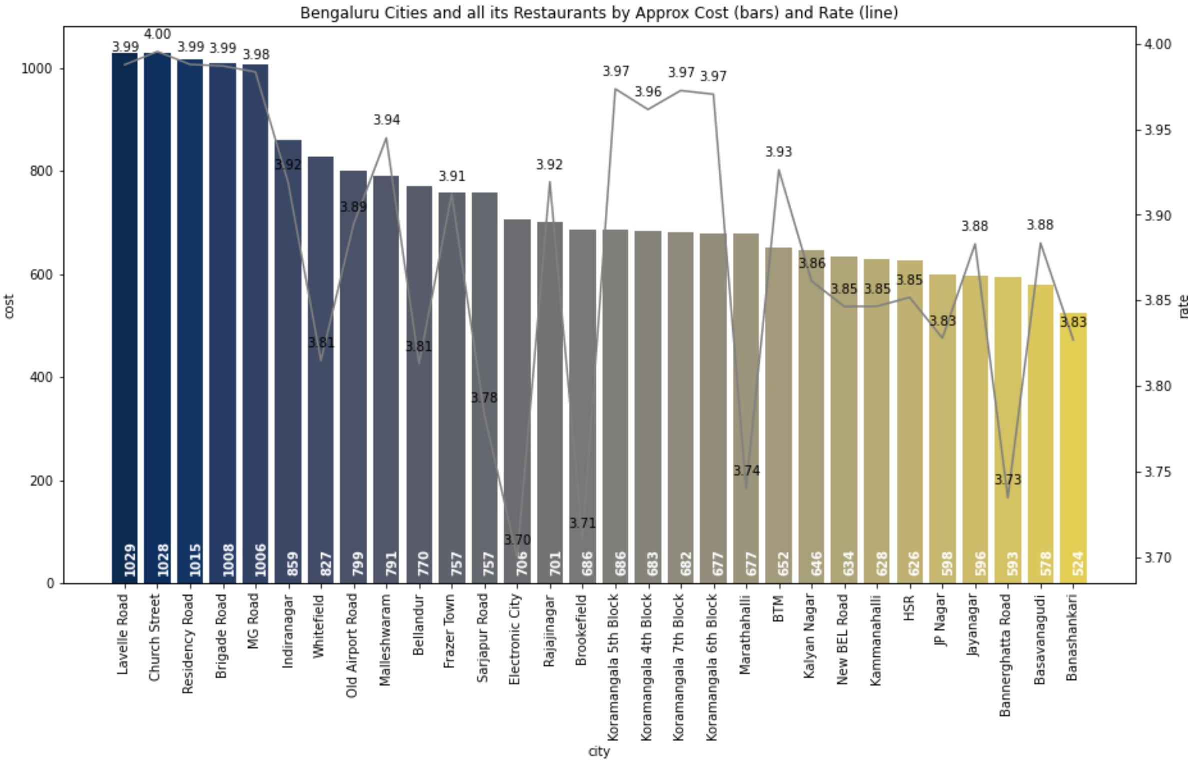
Now we check the locations with lowest cost for two people and map



2.4 Locations where restaurants with lowest cost food in Bangalore

2.5 Bengaluru City Restaurants by Approx Cost (bars) and Rate (line)

Now let's purpose a clearly view for cities with restaurants in Bengaluru and take a look at the average cost and rate of restaurants in each one. The idea is to see how these two variables are related and to present customers the insight of choosing the best city to visit (eating purposes) in Bengaluru.

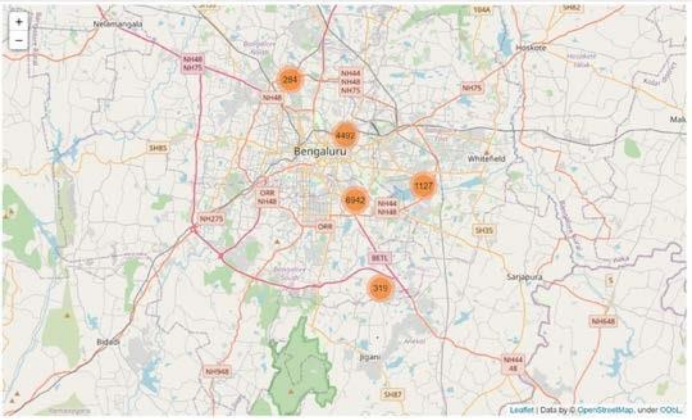


2.6 Bengaluru City Restaurants by Approx Cost (bars) and Rate (line)

2.7 Locate the Restaurants located in Bengaluru using geopy

In the following steps, the real idea was to extract lat and long features from the address in the dataset, but there are huge differences between address formats that made it difficult. So, maybe the best we can do here is to extract geolocation features based in the restaurant city.

Of course, this is not the best approach in terms of location precision, but I think is the best we can do regarding on the problems in the dataset address column construction. Let's use the Nominatim API to help us doing this job. I'll let my user_agent open

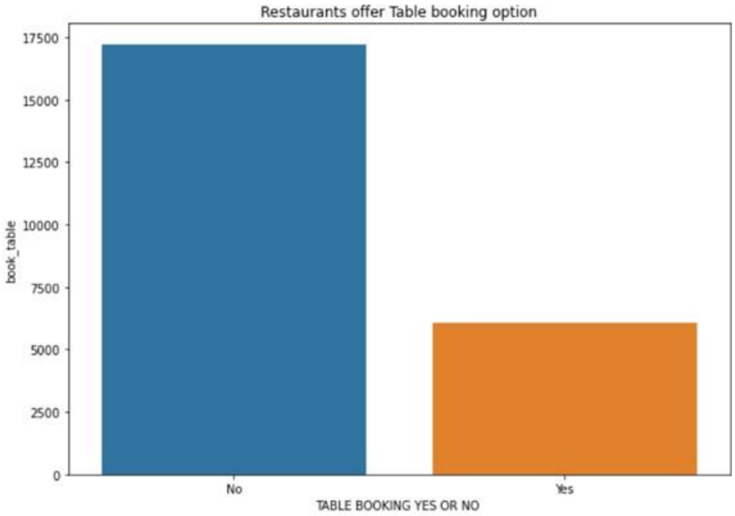


2.7 Geo Locations of restaurants located in Bengaluru

2.8 Restaurants Services offered

2.8.1 How many restaurants offer Book Table service?

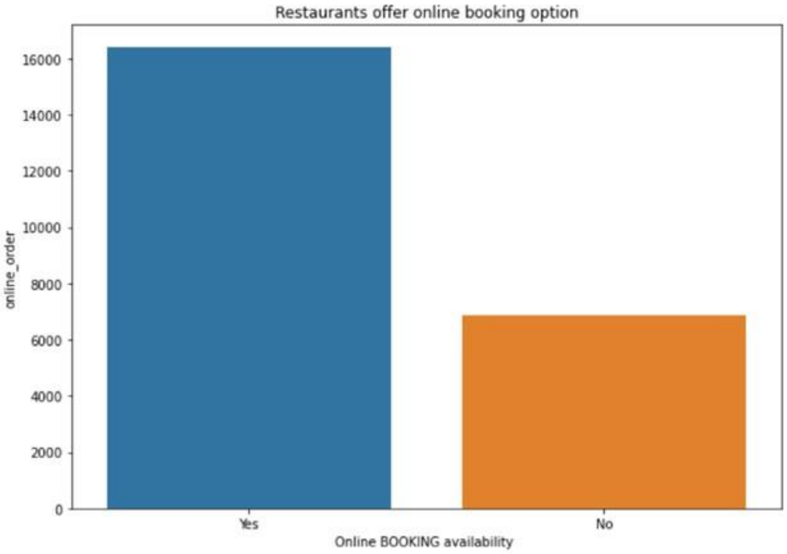
Well, with the chart above we purpose a good look at the Bengaluru scenario: we chose three numerical features (total votes, mean approx cost and rate). Now let's use other useful features of our dataset to understand how these numerical features can be changed along the services restaurants offers, like book table and online order.



- We can now see most of the restaurants donot offer table booking option

2.8.1 how many restaurants offer Book Table service

2.8.2 How many Restaurants offer online booking facility

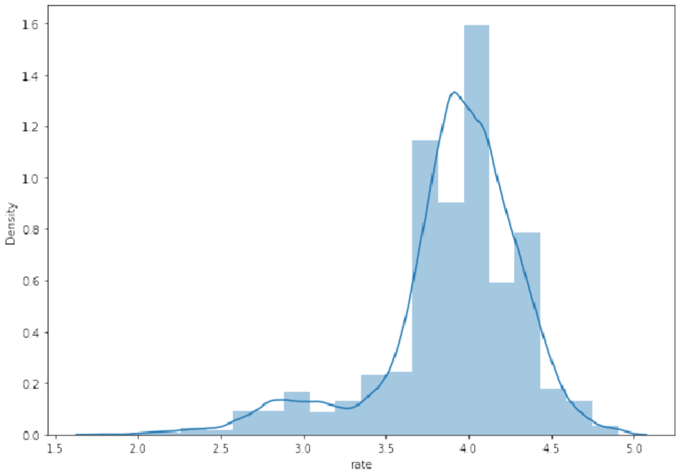


- we can observe that most of the Restaurants offer online delivering option

2.8.2 How many Restaurants offer online booking facility

2.9 Plot Restaurants Ratings distribution to know how ratings are in Bengaluru

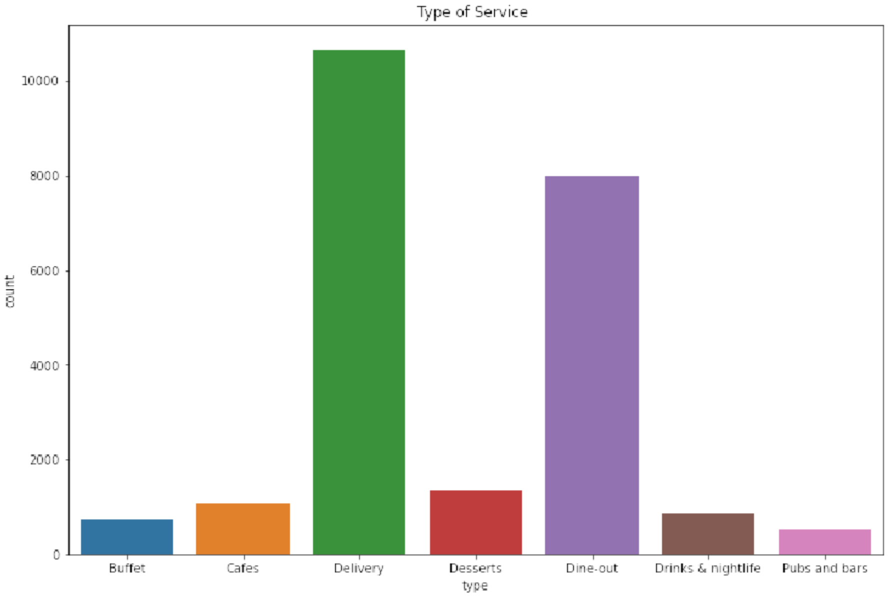
Let's take a closer look into the type of restaurants in this zomato dataset. The idea is to see if there is some preference in zomato orders.



2.9 Ratings for Restaurants in Bengaluru

2.10 Different service Types offered in Restaurants

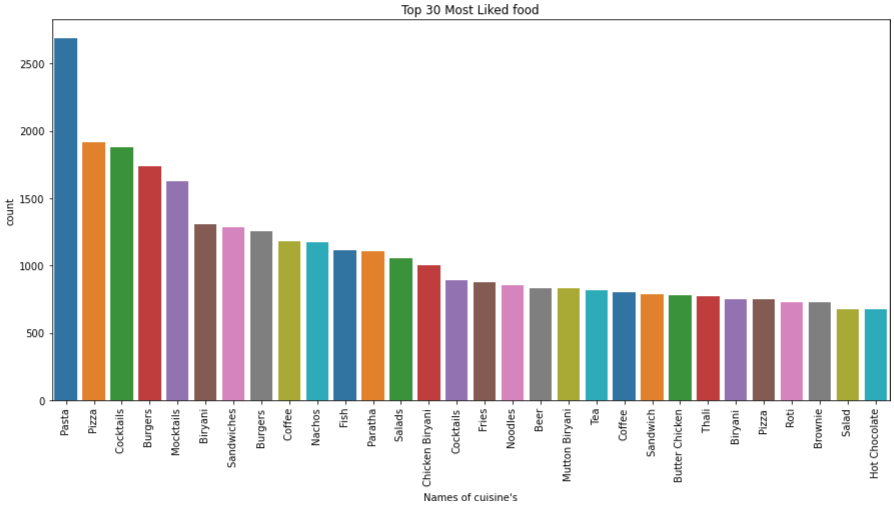
We use data frame df1['type'] column with unique feature to check the services offered and count the restaurants with highest type of service provides in Bengaluru.



2.10 service Types offered in Restaurants

2.11 Most liked dishes on Bengaluru Restaurants

For answering this question, let's do a text mining on cuisine column fo extract single options for each restaurant. After that, we will plot a bar to see trends on food options.



2.11 Thirty Most liked dishes on Bengaluru Restaurants

Wow! There are a huge amount of Pasta, Pizza, Burgers and Cocktails. We can also see Biryani, Coffee, Sandwiches and others.

3.0 Building Model Predicting the Success of a Restaurant:

our main task on this notebook: predict the success of a restaurant in Bengaluru using the data provided and extracting some additional features. At this point probably you're asking: but how we will do it? That's what a thought for this task:

Let's keep the first one aside for now and let's work only with the training and validation set. The next step is to create our target variable to be used in this classification task.

The main point here is to define a fair threshold for splitting the restaurants into good and bad ones. It would be a really experimental decision and we must keep in mind that this approach is not the best one. Probably it would let margin for classification errors. Even so, let's try!

The next step is to prepare some features for training our classification model.

After defining the target and splitting the data into train+val and test sets, let's define the features to be used on training. Here we will take a look at the raw data to select valuable features and apply some steps to create another ones.

The initial set of selected features inclue: - online_order; - book_table; - location; - rest_type; - cuisines; - listed_in(type); - listed_in(city); - approx_cost

We created the types, dishes and cuisines features in a way of counting the food services offered by the restaurant. This is information can be gotten before the launching of the establishment.

Looking at the data we prepared for training our classification model, it's important to point the need to apply some encoding technique on categorical features. Here we can define the "global reach" of your restaurant success predictor: using features like location and listed_in(city) for training will restrict the use of our model to Bengaluru area only. Let's keep those ones for a while pointing that, at least on this first moment, our classification model could only be used to predict success of restaurants in Bengaluru.

For the last act on this topic, let's split the data into training and validation sets once and for all.

Training a Model

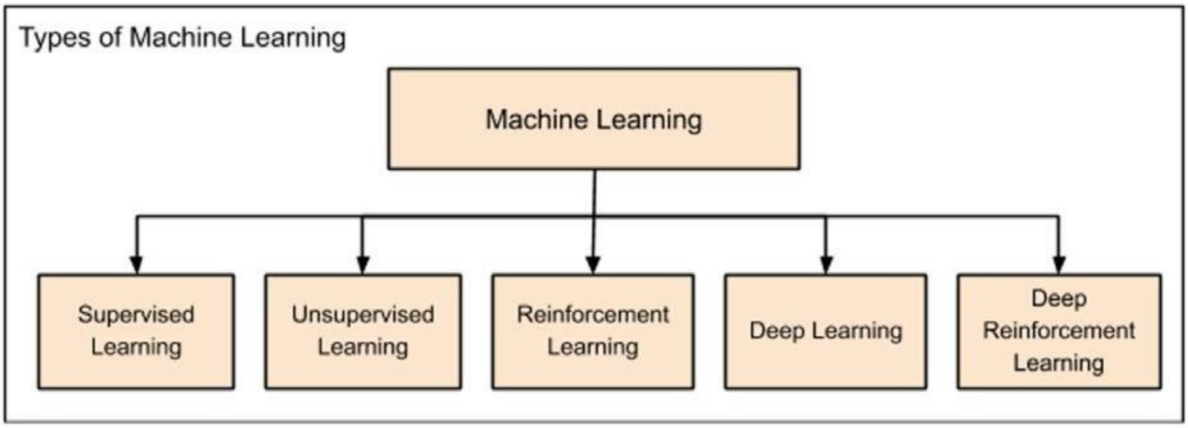
For this task, let's train a LinearRegression, RandomForest and ExtraTreesRegressor

The above are Machine learning models, Machine Learning is broadly categorized under the following

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning
- Deep Learning
- Deep Reinforcement Learning

Source url:

https://www.tutorialspoint.com/machine_learning_with_python/index.htm



Supervised Learning:

Supervised learning is analogous to training a child to walk. You will hold the child's hand, show him how to take his foot forward, walk yourself for a demonstration and so on, until the child learns to walk on his own.

Regression: The technique helps in finding the interrelationship that links the variables and enables us to predict the uninterrupted output variable based on the one or more predictor variables.

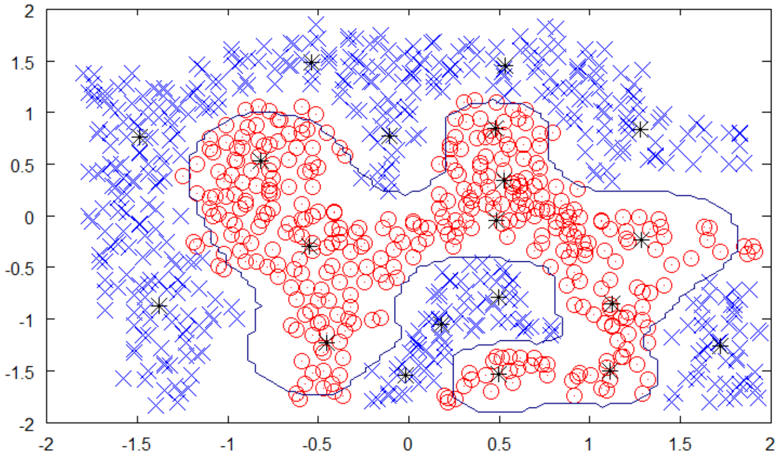
Classification: Classification is to determine which category an observation belongs to, and this is done by understanding the relationship between the dependent variable and the independent variables. Here the dependent variable is categorical, while the independent variables can be numerical or categorical. As there is a dependent variable that allows us to establish the relationship between the input variables.

Unsupervised Learning:

In unsupervised learning, we do not specify a target variable to the machine, rather we ask machine "What can you tell me about X?". More specifically, we may ask questions such as given a huge data set X, "What are the five best groups we can make out of X?" or "What feature occur together most frequently in X?". To arrive at the answers to such questions, you can understand that the number of data points that the machine would require to deduce a strategy would be very large.

The unsupervised learning has shown a great success in many modern AI applic such as face detection, object detection, and so on.

Source url: https://chrisjmccormick.files.wordpress.com/2013/08/approx_decision_boundary.png

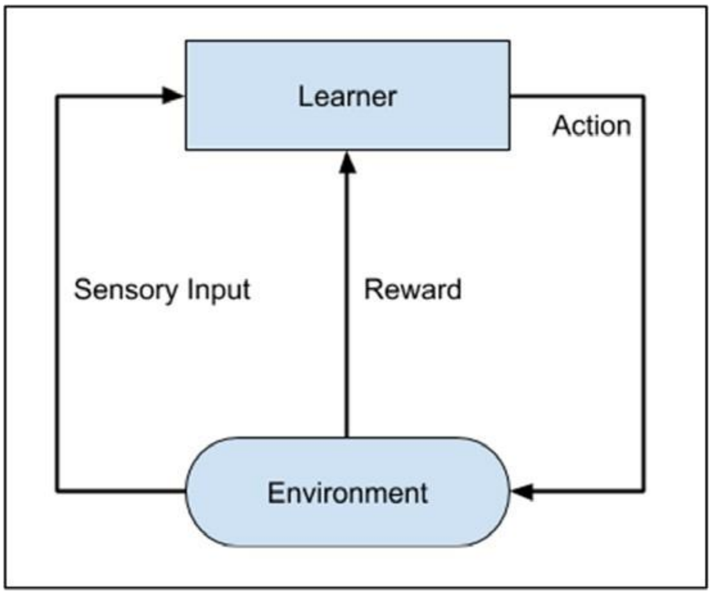


Reinforcement Learning:

Consider training a pet dog, we train our pet to bring a ball to us. We throw the ball at a certain distance and ask the dog to fetch it back to us. Every time the dog does this right, we reward the dog. Slowly, the dog learns that doing the job rightly gives him a reward and then the dog starts doing the job right way every time in future. Exactly, this concept is applied in "Reinforcement" type of learning. The technique was initially developed for machines to play games. The machine is given an algorithm to analyses all possible moves at each stage of the game. The machine may select one of the moves at random.

Source url:

https://www.tutorialspoint.com/machine_learning_with_python/index.htm



Deep Learning:

The deep learning is a model based on Artificial Neural Networks (ANN), more specifically Convolutional Neural Networks (CNN)s. There are several architectures used in deep learning such as deep neural networks, deep belief networks, recurrent neural networks, and convolutional neural networks.

Deep Reinforcement Learning:

The Deep Reinforcement Learning (DRL) combines the techniques of both deep and reinforcement learning. The reinforcement learning algorithms like Q-learning are now combined with deep learning to create a powerful DRL model. The technique has been with a great success in the fields of robotics, video games, finance and healthcare.

Now Discuss about Linear Regression, Random Forest and Extra Trees Regressor models

The development of today's AI applications started with using the age-old traditional statistical techniques. You must have used straight-line interpolation in schools to predict a future value. There are several other such statistical techniques which are successfully applied in developing so-called AI programs.

Some of the examples of statistical techniques that are used for developing AI applications in those days and are still in practice are listed here:

- Regression
- Classification
- Clustering
- Probability Theories
- Decision Trees

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

Advantages of Random Forest

- Random Forest can perform both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks

Extra Trees Regressor models:

The Extra Trees algorithm works by creating many unpruned decision trees from the training dataset. Predictions are made by averaging the prediction of the decision trees in the case of regression or using majority voting in the case of classification.

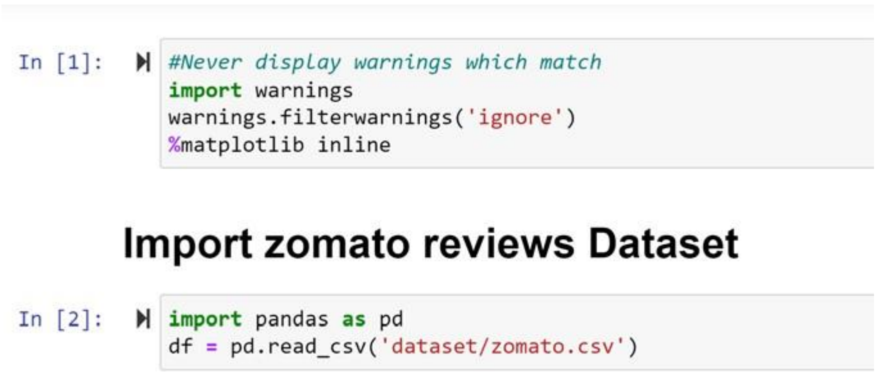
Regression: Predictions made by averaging predictions from decision trees.

Classification: Predictions made by majority voting from decision trees.

We can also use the Extra Trees model as a final model and make predictions for classification.

Implementation Results

First, we imported the necessary libraries.

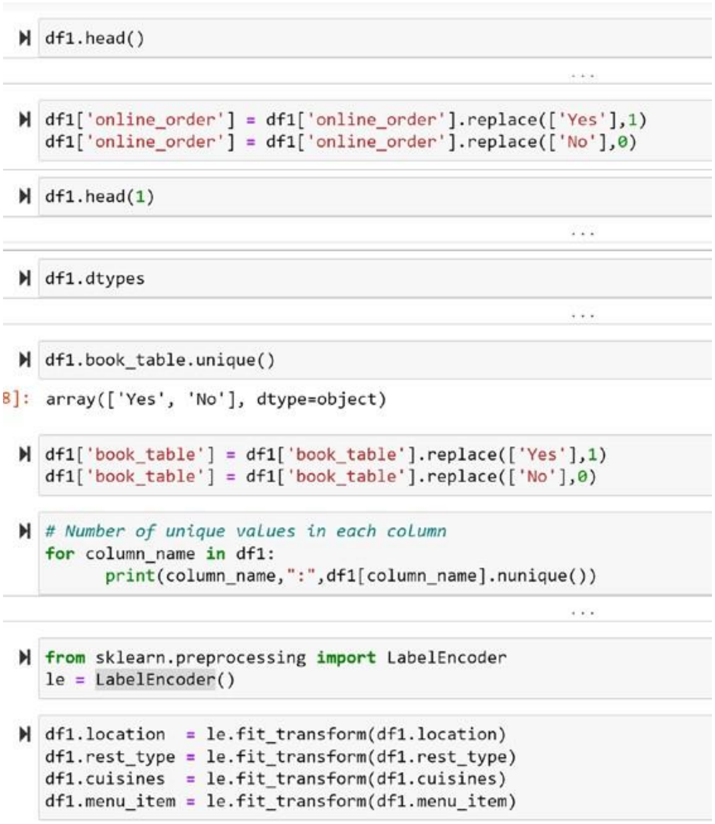


```
In [1]:    #Never display warnings which match
           import warnings
           warnings.filterwarnings('ignore')
           %matplotlib inline
```

Import zomato reviews Dataset

```
In [2]:    import pandas as pd
           df = pd.read_csv('dataset/zomato.csv')
```

Then we make the required changes to the columns in the data frame, Like removing null values from the data frame, dropping the un-required columns, renaming the column names, changing the object datatype to categorical data by making them unique values as per the column name meaning, then used LabelEncoder to convert the categorical values into numerical values that helps to ease the model development.

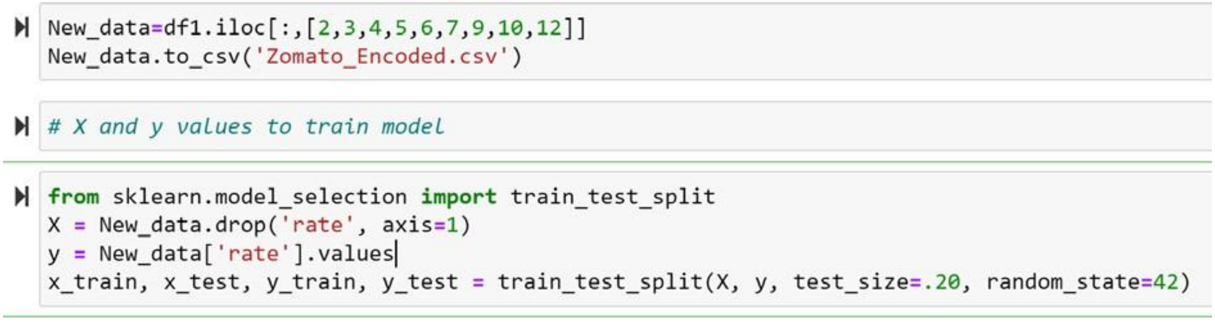


```
df1.head()
...
df1['online_order'] = df1['online_order'].replace(['Yes'],1)
df1['online_order'] = df1['online_order'].replace(['No'],0)

df1.head(1)
...
df1.dtypes
...
df1.book_table.unique()

8]:  array(['Yes', 'No'], dtype=object)

df1['book_table'] = df1['book_table'].replace(['Yes'],1)
df1['book_table'] = df1['book_table'].replace(['No'],0)

# Number of unique values in each column
for column_name in df1:
    print(column_name,":",df1[column_name].nunique())
...
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

df1.location  = le.fit_transform(df1.location)
df1.rest_type = le.fit_transform(df1.rest_type)
df1.cuisines  = le.fit_transform(df1.cuisines)
df1.menu_item = le.fit_transform(df1.menu_item)
```

Make the new Dataframe with updated columns and save the file.

Use the New Dataframe and create X (input variables) and y (output variable) data.

Split the data into train and test using ‘train_test_split’ module.

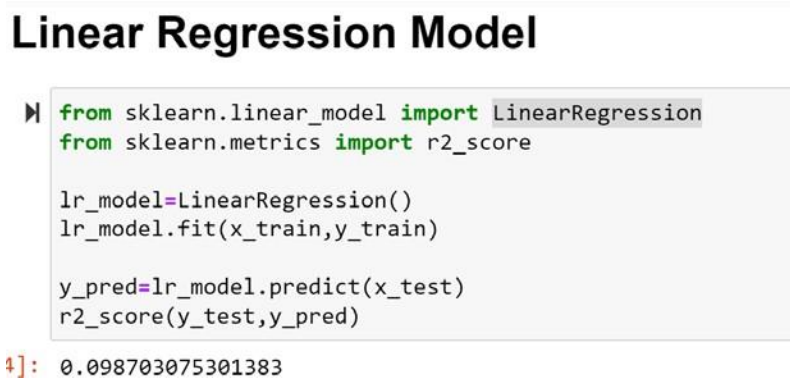


```
New_data=df1.iloc[:,[2,3,4,5,6,7,9,10,12]]
New_data.to_csv('Zomato_Encoded.csv')
```

```
# X and y values to train model
```

```
from sklearn.model_selection import train_test_split
X = New_data.drop('rate', axis=1)
y = New_data['rate'].values
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=.20, random_state=42)
```

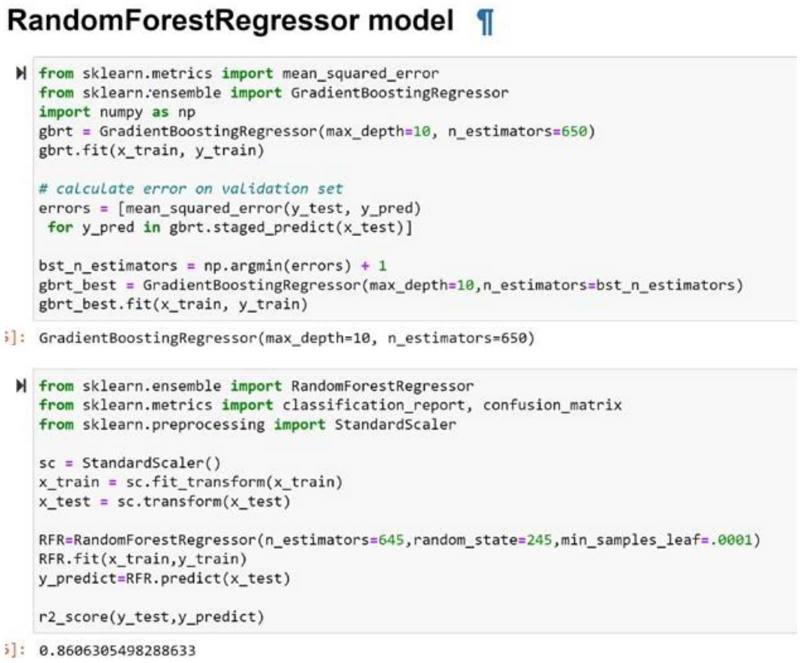
We then created and parameterized the Sequential the model into which we’d feed the training and testing data:



Linear Regression Model

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

lr_model=LinearRegression()
lr_model.fit(x_train,y_train)

y_pred=lr_model.predict(x_test)
r2_score(y_test,y_pred)
```

```
4]:  0.098703075301383
```



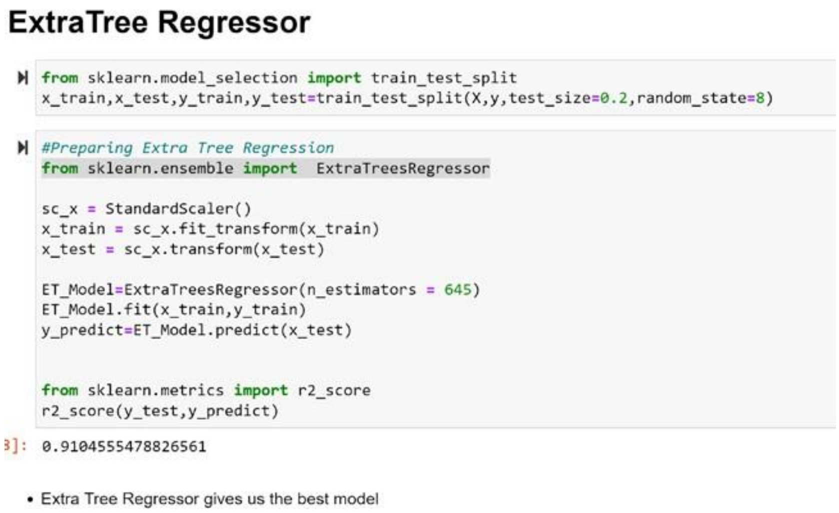
RandomForestRegressor model ¶

```
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import GradientBoostingRegressor
import numpy as np
gbrt = GradientBoostingRegressor(max_depth=10, n_estimators=650)
gbrt.fit(x_train, y_train)

# calculate error on validation set
errors = [mean_squared_error(y_test, y_pred)
 for y_pred in gbrt.staged_predict(x_test)]

bst_n_estimators = np.argmin(errors) + 1
gbrt_best = GradientBoostingRegressor(max_depth=10,n_estimators=bst_n_estimators)
gbrt_best.fit(x_train, y_train)
```

```
5]:  GradientBoostingRegressor(max_depth=10, n_estimators=650)
```

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)

RFR=RandomForestRegressor(n_estimators=645,random_state=245,min_samples_leaf=.0001)
RFR.fit(x_train,y_train)
y_predict=RFR.predict(x_test)

r2_score(y_test,y_predict)
```

```
5]:  0.8606305498288633
```



ExtraTree Regressor

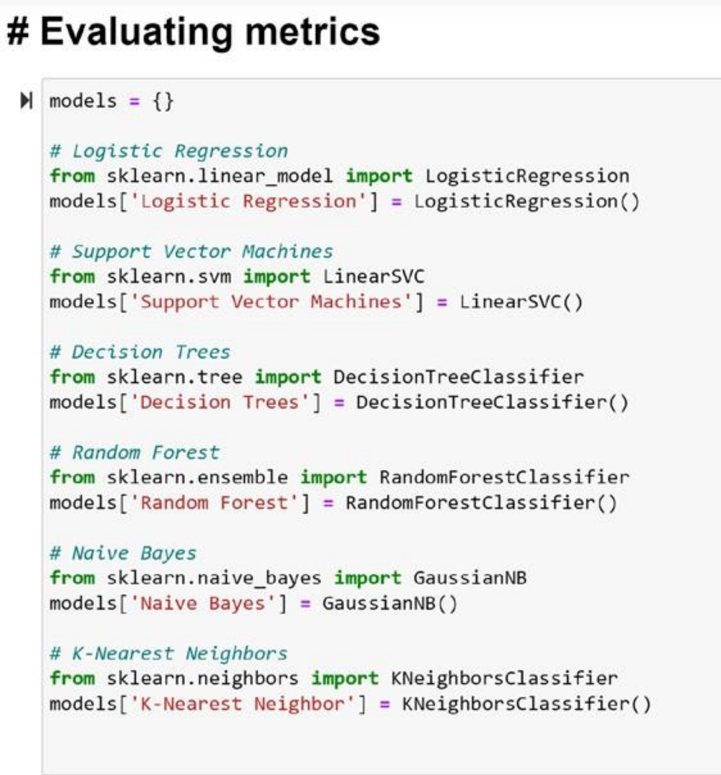
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=8)
```

```
#Preparing Extra Tree Regression
from sklearn.ensemble import  ExtraTreesRegressor

sc_x = StandardScaler()
x_train = sc_x.fit_transform(x_train)
x_test = sc_x.transform(x_test)

ET_Model=ExtraTreesRegressor(n_estimators = 645)
ET_Model.fit(x_train,y_train)
y_predict=ET_Model.predict(x_test)


from sklearn.metrics import r2_score
r2_score(y_test,y_predict)
```

```
3]:  0.9104555478826561
```

- Extra Tree Regressor gives us the best model

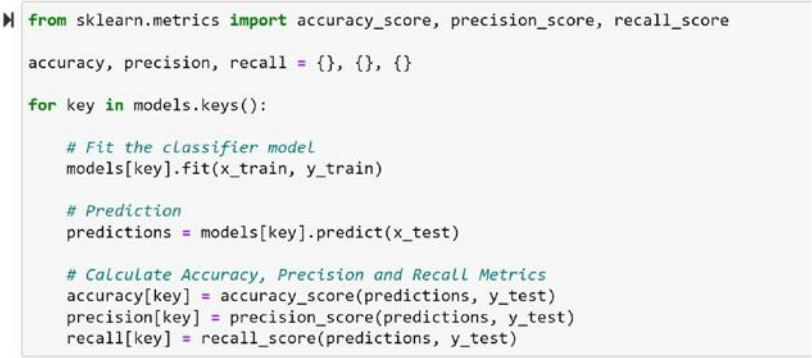
Here we created three models and the best r2 score from the above models is 0.91 form ExtraTreeRegressor.



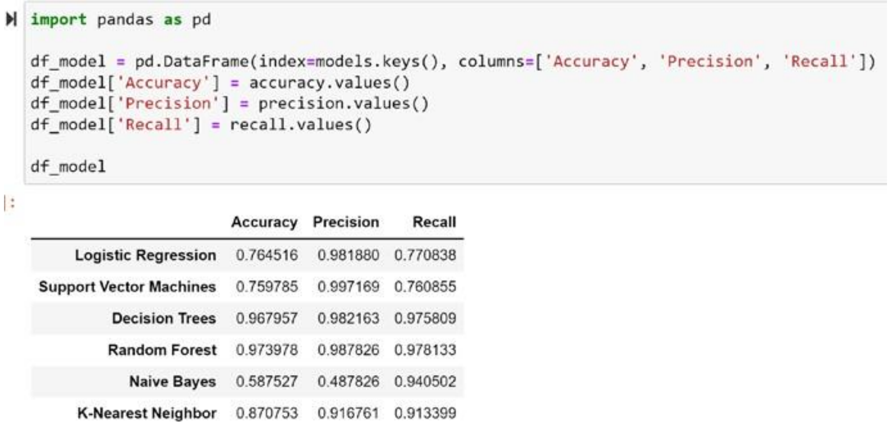
Evaluating metrics

```
models = {}

# Logistic Regression
from sklearn.linear_model import LogisticRegression
models['Logistic Regression'] = LogisticRegression()

# Support Vector Machines
from sklearn.svm import LinearSVC
models['Support Vector Machines'] = LinearSVC()

# Decision Trees
from sklearn.tree import DecisionTreeClassifier
models['Decision Trees'] = DecisionTreeClassifier()

# Random Forest
from sklearn.ensemble import RandomForestClassifier
models['Random Forest'] = RandomForestClassifier()

# Naive Bayes
from sklearn.naive_bayes import GaussianNB
models['Naive Bayes'] = GaussianNB()

# K-Nearest Neighbors
from sklearn.neighbors import KNeighborsClassifier
models['K-Nearest Neighbor'] = KNeighborsClassifier()
```

(



```
from sklearn.metrics import accuracy_score, precision_score, recall_score

accuracy, precision, recall = {}, {}, {}

for key in models.keys():

    # Fit the classifier model
    models[key].fit(x_train, y_train)

    # Prediction
    predictions = models[key].predict(x_test)

    # Calculate Accuracy, Precision and Recall Metrics
    accuracy[key] = accuracy_score(predictions, y_test)
    precision[key] = precision_score(predictions, y_test)
    recall[key] = recall_score(predictions, y_test)
```



```
import pandas as pd

df_model = pd.DataFrame(index=models.keys(), columns=['Accuracy', 'Precision', 'Recall'])
df_model['Accuracy'] = accuracy.values()
df_model['Precision'] = precision.values()
df_model['Recall'] = recall.values()

df_model
```

	Accuracy	Precision	Recall
Logistic Regression	0.764516	0.981880	0.770838
Support Vector Machines	0.759785	0.997169	0.760855
Decision Trees	0.967957	0.982163	0.975809
Random Forest	0.973978	0.987826	0.978133
Naive Bayes	0.587527	0.487826	0.940502
K-Nearest Neighbor	0.870753	0.916761	0.913399

With this implementation it was possible to delivery useful information for Business areas and also for Zomato customers on choosing the best restaurant for ordering (specially the new ones). This is an example of what predictive models can do in practice.

Presented by

BADAM VINAY KUMAR(U00847415)