



**A PROJECT REPORT ON
ZOMATO RESTAURANT RATING PREDICTION.
BY**

CHAVAN YOGITA KAILAS

Roll No: 1312

SALVI RUTUJA SANJAY

Roll No: 1343

Under the Guidance of

Mr.AKSHAY TILEKAR

**POST GRADUATE DIPLOMA IN BIG DATA ANALYTICS
FEB-2020**

**INSTITUTE FOR ADVANCED COMPUTING AND
SOFTWARE DEVELOPMENT, AKURDI, PUNE**



CERTIFICATE

This is to certify that the Project Entitled

ZOMATO RESTAURANT RATING PREDICTION

Submitted by

CHAVAN YOGITA KAILAS

Roll No: 1312

SALVI RUTUJA SANJAY

Roll No: 1343

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of Mr. AKSHAY TILEKAR and it is approved for the partial fulfillment of the requirement of Post Graduate Diploma in Big Data Analytics.

Mr. AKSHAY TILEKAR
Project Guide
PG-DBDA

Mr. PRASHANTKARHALE
Centre Coordinator
IACSD, AKURDI

ACKNOWLEDGEMENT

It gives us great pleasure in presenting the preliminary project report on 'ZOMATO RESTAURANT RATING PREDICTION'.

I would like to take this opportunity to thank my internal guide Mr. AKSHAY TILEKAR for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.

I am also grateful to Mr. PRASHANT KARHALE, Centre Coordinator, Akurdi, Pune for his indispensable support, suggestions.

In the end our special thanks to IACSD for providing various resources such as laboratory with all needed software platforms, continuous Internet connection, for our Project.

CHAVAN YOGITA KAILAS

SALVI RUTUJA SANJAY

ABSTRACT

Some start-ups in the Indian Food & Beverages industry have revolutionized the way we look around for places to dine. The scenario has changed from previous practice when we had to actually go to restaurants to select the restaurant and have the experience. Today, choice for the best places to have food is just a click away. One can choose the best rated place and then decide to enjoy there with friends and family.

Contents

1. INTRODUCTION

1.1 Purpose	1
1.2 Problem Description	1
1.3. Goal and Objective.....	2

2 OVERALL DESCRIPTION

2.1 Data	3
2.2 Import	4

3. DATA PRE-PROCESSING AND ANALYSIS

3.1 Data Cleaning	5
3.1.1 Screenshot before cleaning	6
3.1.2 Screenshot after cleaning	7
3.2 Pre-processing on Data	7

4. SYSTEM DESIGN

4.1 Flow chart of System	8
--------------------------------	---

5. TESTING

5.1 Saving Model	24
5.2 Testing on dummy data	25
5.3 Predicting Rating of new restaurants	26

6. CONCLUSION

7. FUTURE SCOPE.....

List of Figures

3.1 Data Before Cleaning	5
3.2 Data After Cleaning	6
4.1 Flow chart of System.	8
4.2 Name Vs Number of Restaurants	10
4.3 Number of Restaurants accepting online order	10
4.4 Number of Restaurants with book table facility.....	11
4.5 Percentage of Restaurants present in that city.....	11
4.6 Top 10 Restaurants type.....	12
4.7 Top 10 most liked dish.	12
4.8 Top 10 rest Serving pasta.....	13
4.9 Top 10 cuisines.....	13
4.10 Average Rating Of Restaurants	14
4.11 Rating Vs online order.....	14
4.12 Heatmap.....	15
4.13 Decision Tree.....	16
4.14 Random Forest Regressor.....	17
4.15 XGBoost.....	18
4.16 Linear Regression without Tuning	19
4.17 Linear Regression with Tuning.....	20
4.18 Decision Tree Without Tuning.....	20
4.19 Decision Tree with Tuning.....	21
4.20 Random Forest Regressor without tuning.....	22
4.21 Random Forest Regressor with tuning.....	23
4.22 XGBoost Regressor without tuning.....	24
4.23 XGBoost Regressor with tuning.....	25
4.24 r2 score plot of all model.....	24
5.1 Saving model.....	25
5.2 Testing and prediction on dummy data.....	25
5.3 Actual data of new restaurants.	26
5.4 Pre-process data of new restaurants.....	26
5.5 Predicted rating.....	26

Chapter 1

INTRODUCTION

Zomato is a restaurant search and discovery website/application which operates in many major cities spread across 24 countries. Zomato helps land customers over the restaurants through many search options based on restaurants location, restaurant type, cuisines they serve and helps discover better using filters and sort by options for advance searches. Bangalore, India is continuously urbanization and stretching its boundaries, the city has more than 8,000 different restaurants, bar, pubs, cafes and has potential to have more successful running food restaurants. Currently 51,000 restaurants are spread across 30 listed location in Bengaluru. Which has 2,000 new restaurants listed on Zomato. The Dataset is in csv file provided in Kaggle by Himanshu Poddar All copyrights for the data is owned by Zomato Media Pvt. Ltd. And the data is for education purpose only Some start-ups in the Indian Food & Beverages industry have revolutionized the way we look around for places to dine. The scenario has changed from previous practice when we had to actually go to restaurants to select the restaurant and have the experience. Today, choice for the best places to have food is just a click away. One can choose the best rated place and then decide to enjoy there with friends and family.

1.1 Purpose

- Zomato charges less commission for higher rating restaurants
- Zomato has filter options (rating 3.5+) which displays better rated restaurants in the search
- Restaurant shows on top over the other in search
- Attracts more footfall to the Restaurants.

1.2 Problem Description

Restaurants from all over the world can be found here in Bengaluru. From United States to Japan, Russia to Antarctica, you get all type of cuisines here. Delivery, Dineout, Pubs, Bars, Drinks, Buffet, Desserts you name it and Bengaluru has it. Bengaluru is best place for foodies. The number of restaurants is increasing day by day. Currently which stands at approximately 12,000 restaurants. With such a high number of restaurants. This industry hasn't been saturated yet. And new restaurants are opening every day. However, it has become difficult for them to compete with already established restaurants. The key issues that continue to pose a challenge to them include high real estate costs, rising food costs, shortage of quality

manpower, fragmented supply chain and over-licensing. This Zomato data aims at analyzing demography of the location. Most importantly it will help new restaurants in deciding their theme, menus, cuisine, cost etc. for a particular location. It also aims at finding similarity between neighborhoods of Bengaluru on the basis of food.

- Does demography of area matters?
- Does location of particular type of restaurant depends on people living in that area?
- Does theme of restaurant matter?

Is food chain category restaurant likely to have more customers than its counterpart?

- Are any neighborhood on similar based on the type of food?
- Are particular neighbors being famous for its own kind of food?
- If two neighbors are similar does that mean these are related or particular group of people live in neighborhood or these are places to eat.
- What kind of food is famous in locality?
- Do entire locality loves veg food, if yes then locality populated by particular set of people e.g., Jain, Gujarati, Marwari who are basically veg.

1.3 Goals and Objectives

- Complete analysis of existing restaurants.
- To find out the rating of newly opened restaurants

Chapter2

OVERALL DESCRIPTION

2.1 Data

Dataset contains information of 51,717(8,792 unique) restaurants in Bengaluru city by 15th March 2019. It contains 16 features and one target column for each restaurant.

• **About the columns:**

1.url(object)– This is simply a string with tells the URL which is almost common for all as it is starting with Zomato.com/xyz/abc.

2. Address(object)– Complete address of the restaurant. Contains the full address which is not required for the analysis, can be deleted.

3.Name(object)– Name of the restaurant, the name is unique and is different languages such as Gabar(Hindi Word), Asha(Hindi word), Twist (English) – it will be difficult to categories 51,717 values in the name column. For these reasons the column can be dropped.

4.Online order (Object)– Yes or No, useful for predicting by converting it to binary.

5. Book Table (Object)– Restaurant provides online table booking service (yes or no), useful for predicting by converting it to binary.

6. Rate (Target)– Overall rating of the restaurant between” 1 – 5” up to one decimal point example 4.2. “New” for new restaurants.

7.Votes(int)– Total number of reviews/votes, not useful for predicting the target variable, needs to be dropped.

8.Phone(object)– Phone number of the restaurant. The counts of phone numbers provided could help in predicting target.

9. Location(object)–Neighborhood of the Restaurant in Bengaluru, important for predicting rating. One- hot encode to change it to dummies.

10. Rest_type(object)– Type of the restaurant, important for predicting rating. One-hot encode to change it to dummies.

11. Dish_liked(object)– Dish liked by people in the restaurant.

12. Cuisines(object)– Cuisines types in the restaurants. Types are comma separated

13. Approx_cost(object)– expected cost for meal of two in Rupees, important for prediction

14. Reviews_list(object)- list of tuples containing reviews for the restaurant, each tuple consists of two values, rating and review by the customer. Needs to be dropped for the prediction for new restaurant.

15. Menu_item(Object)– List of menus available in restaurants. Contains more than 54% missing values, needs to be dropped.

16. Listed_in(type)- type of meal

17. Listed_in(city) - contains the neighborhood in which the restaurant is listed duplicate column can be dropped.

About Dataset:

- Dataset size: 51,717 rows x 17 columns
- All columns are of type "object" except for votes
- 7775 ratings are missing in the rating column
- 1208 phone numbers are missing in the phone column
- 21 are missing location
- 28,078 dished liked are missing
- 45 cuisines are missing
- 346 approx. cost for two persons is missing

2.2 Imports

• **Matplotlib:** Matplotlib is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure. We have used it to show visualizations of analysis.

• **Pandas:** Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. We have used pandas for perform operation on data.

• **Numpy:** Numpy is used to for mathematical operations. This package provides easy use of mathematical function.

• **Seaborn:** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

• **Sklearn:** Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines.

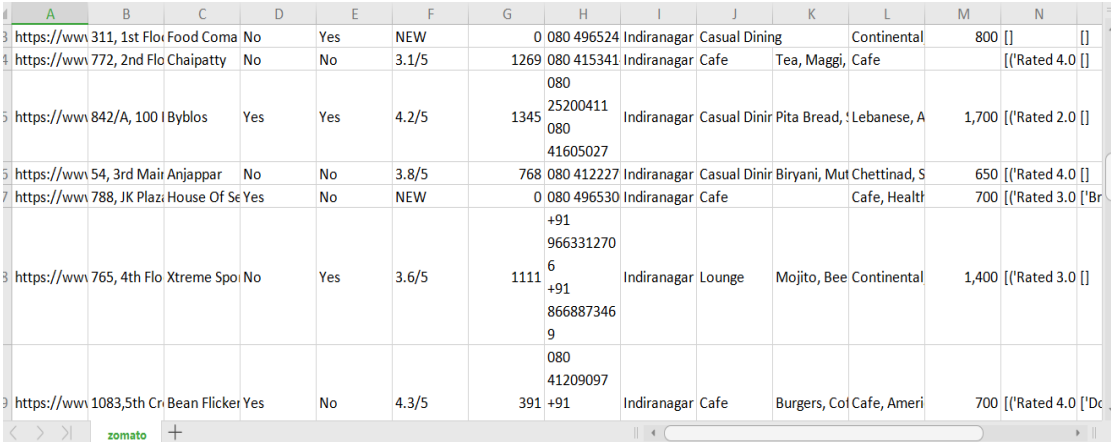
Chapter 3

DATA PRE-PROCESSING AND ANALYSIS

3.1 Data Cleaning

Data cleaning is the process of preparing data for analysis by removing or changing data that is wrong, incomplete, unrelated/unimportant, copied, or improperly formatted. This data is usually not necessary or helpful when it comes to carefully studying data because it may interfere with the process or provide incorrect results. There are (more than two, but not a lot of) methods for cleaning data depending on how it is stored along with the answers being searched for/tried to get.

3.1.1 Screenshot before cleaning



#	A	B	C	D	E	F	G	H	I	J	K	L	M	N
3	https://www.311, 1st Floor Food Coma	No	Yes	NEW			0 080 496524	Indiranagar	Casual Dining	Continental		800	[]	[]
4	https://www.772, 2nd Floor Chaipatty	No	No	3.1/5			1269 080 415341	Indiranagar	Cafe	Tea, Maggi, Cafe				[('Rated 4.0 [])
5	https://www.842/A, 100 Byblos	Yes	Yes	4.2/5			1345 25200411 080 41605027	Indiranagar	Casual Dining	Pita Bread, Lebanese, A		1,700	[('Rated 2.0 [])	
5	https://www.54, 3rd Floor Anjappar	No	No	3.8/5			768 080 412227	Indiranagar	Casual Dining	Biryani, Mut Chettinad, S		650	[('Rated 4.0 [])	
7	https://www.788, JK Plaza House Of Se	Yes	No	NEW			0 080 496530	Indiranagar	Cafe	Cafe, Health		700	[('Rated 3.0 ['Br	
8	https://www.765, 4th Floor Xtreme Spo	No	Yes	3.6/5			+91 9663312706 +91 8668873469 080 41209097	Indiranagar	Lounge	Mojito, Bee Continental		1,400	[('Rated 3.0 [])	
9	https://www.1083, 5th Floor Bean Flicker	Yes	No	4.3/5			391 +91	Indiranagar	Cafe	Burgers, Col Cafe, Ameri		700	[('Rated 4.0 ['Dc	

Figure 3.1: Data before Cleaning

In this Figure the data has Nan values.

3.1.2 Screenshot after cleaning

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	online_order	booked_table	rating	phone	rest_type	cuisines	average_cost	listed_in(type)	listed_in(city)						
2	0	1	1	4.1	2	casualdining north indian	800	Buffet	Banashankari						
3	1	1	0	4.1	1	casualdining chinese, north indian	800	Buffet	Banashankari						
4	2	1	0	3.8	1	cafe,casual cafe, mexican	800	Buffet	Banashankari						
5	3	0	0	3.7	0	quickbites south indian	300	Buffet	Banashankari						
6	4	0	0	3.8	1	casualdining north indian	600	Buffet	Banashankari						
7	5	1	0	3.8	0	casualdining north indian	600	Buffet	Banashankari						
8	6	0	0	3.6	1	casualdining north indian	800	Buffet	Banashankari						
9	7	1	1	4.6	2	casualdining pizza, cafe, italian	600	Cafes	Banashankari						
10	8	1	0	4	2	cafe cafe, italian	700	Cafes	Banashankari						
11	9	1	0	4.2	2	cafe cafe, mexican	550	Cafes	Banashankari						
12	10	1	0	4.1	2	cafe cafe	500	Cafes	Banashankari						
13	11	1	1	4.2	1	cafe cafe, italian	600	Cafes	Banashankari						
14	12	1	1	4.2	1	cafe cafe, chinese	500	Cafes	Banashankari						
15	13	0	0	4	1	cafe cafe, continental	450	Cafes	Banashankari						
16	14	1	0	3.8	1	cafe,casual cafe, mexican	800	Cafes	Banashankari						
17	15	1	0	3.8	2	cafe cafe	650	Cafes	Banashankari						
18	16	1	0	3.9	2	cafe cafe, fast food	800	Cafes	Banashankari						
19	17	1	0	3.8	2	cafe chinese, cafe	700	Cafes	Banashankari						
20	18	0	0	3.9	1	cafe cafe, italian	300	Cafes	Banashankari						
21	19	1	0	3.1	1	cafe cafe, chinese	400	Cafes	Banashankari						

Figure 3.2: Data after Cleaning

In this Figure the data has Nan values gets removed.

3.2 Pre-Processing on the data

Renaming columns

- approx_cost (for two people) to average_cost
- listed_in(city) to city

Dropping Unwanted columns

- The URL
- **Name**- most of the names have their short address with their names. The number of words cannot be used to analyze the data
- **Address**- The locality and city in Bangalore is mentioned in another column.
- **Votes**- The new restaurants will not have any prior vote counts
- **dish_liked**- More than 50% of the data is missing, hence cannot be used for analysis and machine learning
- **menu_item**- More than 75% of data is not provided, this column cannot be used for analysis or prediction
- **review_list**- The new restaurant won't have a review list as it is a new restaurant on Zomato
 - listed_in(type): This column is similar to rest_type
 - listed_in(city): This column is similar location

Dropping missing rows

- Removing rows with missing ratings and “-“ no rating provided.
- Replacing missing phone values with “0” indicating it is not provided. Assuming

missing phone numbers are not provided.

Non-Numerical categorical columns

- **online_order** - yes or no for online order booking
- **book_table** - yes or no for table booking Created dummy variables for both these columns which gave 2 columns with binary values.

Numerical, float in form of string columns

- **Rate** - 1,1.1,1. 2,4.9,5

Ratings are given in the form of a string “1.5 /5”. A function is defined in file which removes spaces separates rates where “/” and returns 1.5 in form of float.

- Phone –
- 0 for no phone numbers
- 1 for one phone number provided
- 2 for two phone number provided

Phones are in the form of str “+91 9865745477\n\r 080 67577578” which are either 8 digits for landline or 10 digits for mobile. A function is created which checks for continuous 8 numerical numbers after removing any space in the string and returns the count of phone numbers 0 for missing phone numbers, 1 or 2.

- **cost_for_two** - Approx. cost for two in the restaurants We first renamed it to **average_cost** for easier access. Cost for two is given in string with commas “1,700”. A function on the column removes comma and returns the integer value example “1,700” to 1700

- **location**- We had 93 unique locations Most of them were listed in the **listed_in_city** column. So, we drop **listed_in_city** column. Created dummy columns for this column

Multi-level categorical columns

We have two columns with multilevel categories i.e. (**rest_type**, **cuisines**)

- **rest_type**–In this column we are creating the bins of similar **rest_type**
- **Cuisines**–In this column also we are creating the bins of similar **cuisines_type**

Chapter 4

SYSTEM DESIGN

4.1 Flowchart of the System:

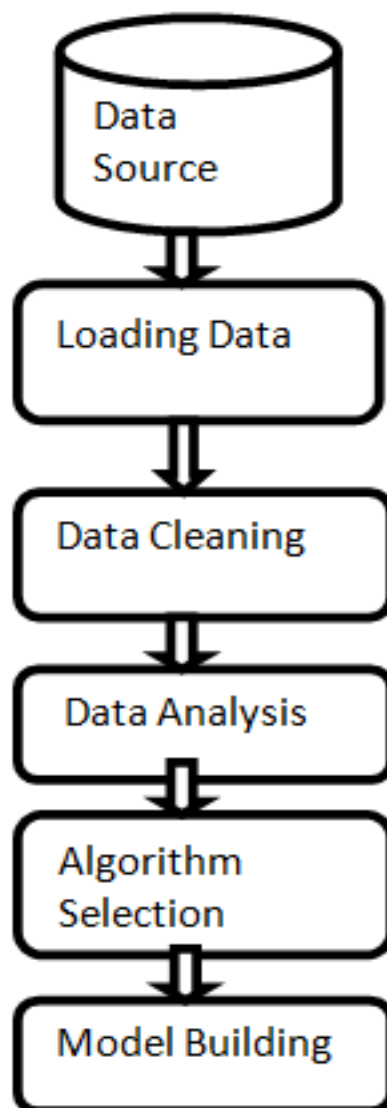


Figure 4.1 Flow chart of system

The above flowchart describes the working flow of the project.

Data Source:

We have collected data from Kaggle from the following link

<https://www.kaggle.com/himanshupoddar/zomato-bangalore-restaurants>

Loading Data:

We loaded our data in Jupyter notebook.

Data Cleaning:

Our data set initially comprised of 17 columns.

They are as follow:URL, Address, Name, online_order, Book_table, Rate, Votes, phone, location, rest_type, dish_liked, Cuisines, approx_cost, review_list, menu_items, list_in(type) and list_in(city).

Missing data in each column were:

7775 missing in ratings

1208 missing in phone numbers

21 missing locations

28,078 missing dish_liked

227 missing rest_type

45 missing cuisines

346 missing approx._cost

Dropped columns:

We dropped the dish_liked column as it contained large number of null values. Then we also dropped menu_item column as it did not have relevant information in it. We dropped Votes column as when a new restaurant is opened and we have to predict ratings for it, that restaurant obviously won't have any votes so we dropped it from our data set. We also dropped list_in(type) and list_in(city) as it contained similar data that was present in rest_type and location.

Dropped Rows:

No of Duplicates in dataset were 15055, so we removed the duplicate rows. We removed the rows which had nulls in rest_type, cuisines and rate.

Handling null values:

In phone no we had nulls but in that with the help of a function we counted no of phone no provided by each restaurant and maintained a count of it and where ever there is null we replaced it by 0, indicating no phone number was provided for that restaurant. In approx._cost we did KNN imputations to replace null values.

Data Analysis:

a) Name vs Number of Restaurant

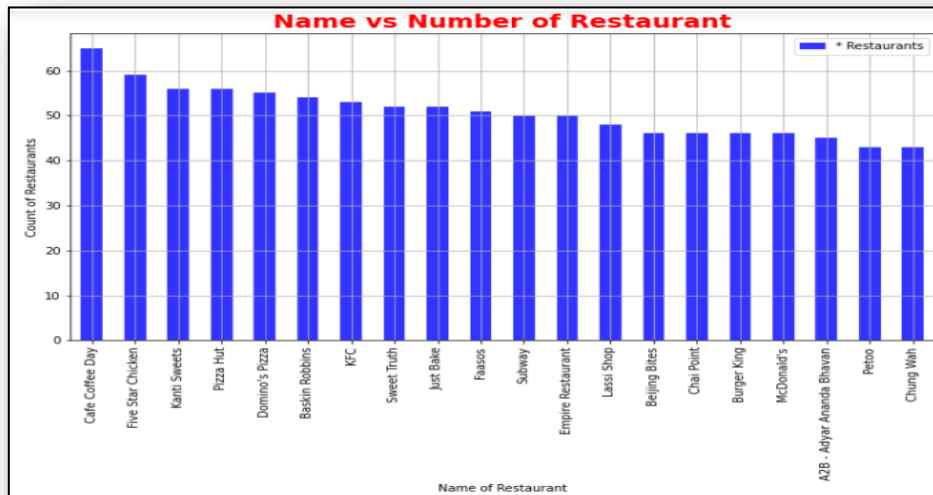


Figure 4.2 Name VsNumber of Restaurants

Observation

- we can say that 'Cafe Coffee Day' day has highest count among all

b) Number of Restaurants accepting online orders



Figure 4.3 Number of restaurants accepting online order

Observation:

- Most of order are online.
- no missing values in online order column

c) No of Restaurant with Book Table Facility

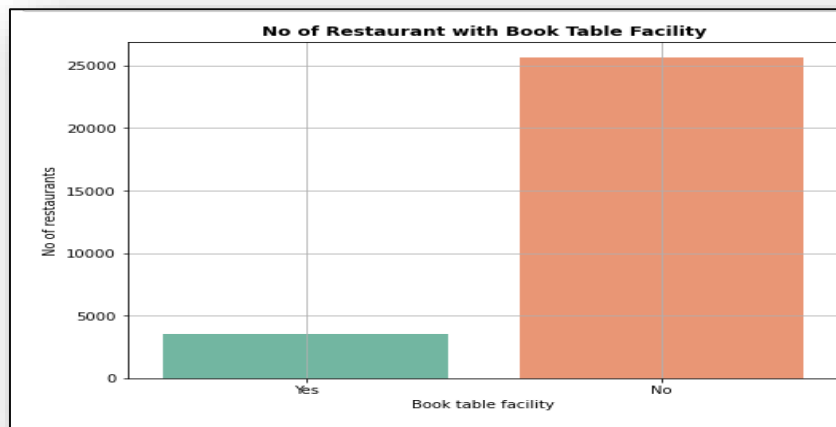


Figure 4.4 Number of restaurants with Book Table Facility

Observation:

- Most of restaurant do not have book table facility
- no missing values in book_tablecolumn

d) Percentage of restaurants present in that city

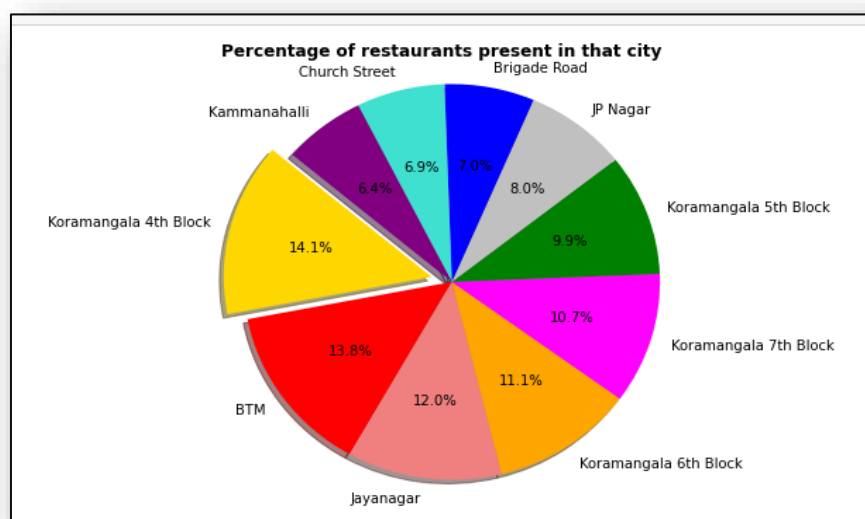


Figure 4.5 Percentage of restaurants present in that city

Observation

- We can say that Koramangala 4th Block location, is where most of restaurant are available

e) Top 10 Restaurant Type

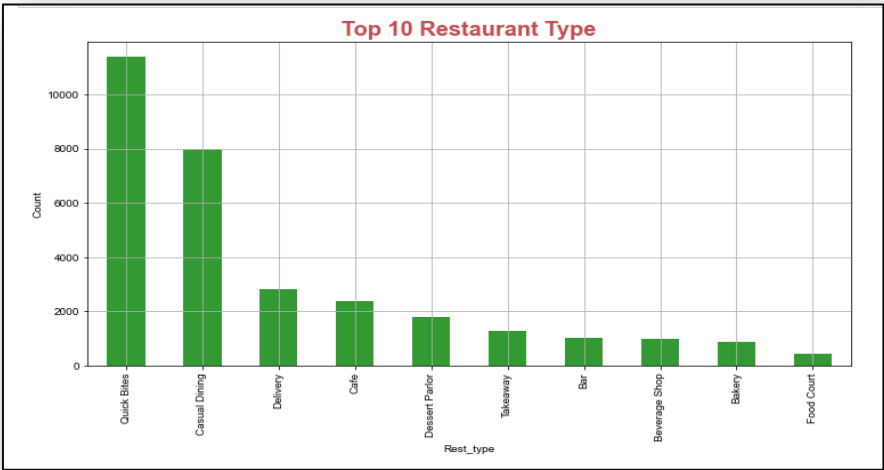


Figure 4.6 Top 10 Restaurant Type

Observation

- We can say that rest_type Quick bites is most popular in Bangalore.

f) Top 10 most liked dishes

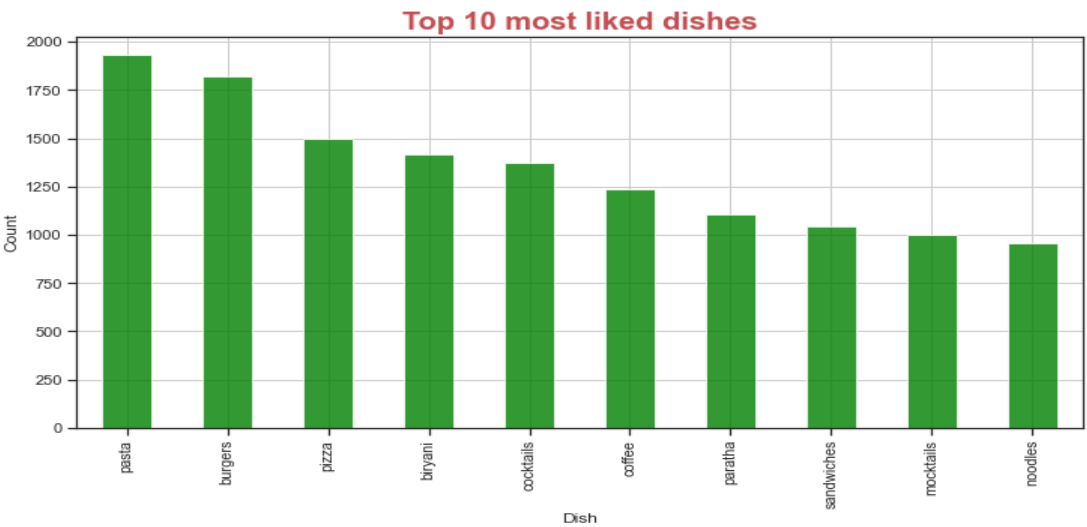


Figure 4.7 Top 10 Most Liked Dish

Observation

- Pasta is the most liked dish followed by burgers

g) Top 10 restaurants serving pasta

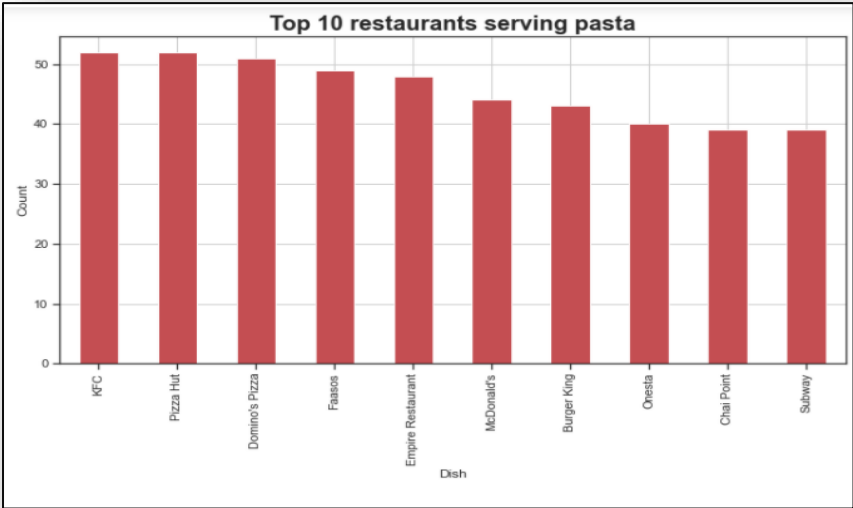


Figure 4.8 Top 10 Restaurants Serving Pasta

Observation:

- Pizza hut followed by KFC serves good Pasta

h) Top 10 cuisines

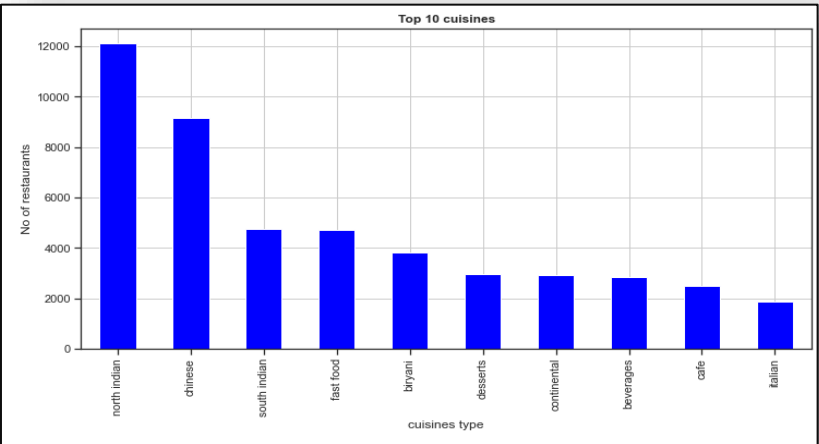


Figure 4.9 Top 10 Cuisines

Observation:

- North Indian food is at top, followed by Chinese and so on.

i) Average Rating of Restaurants

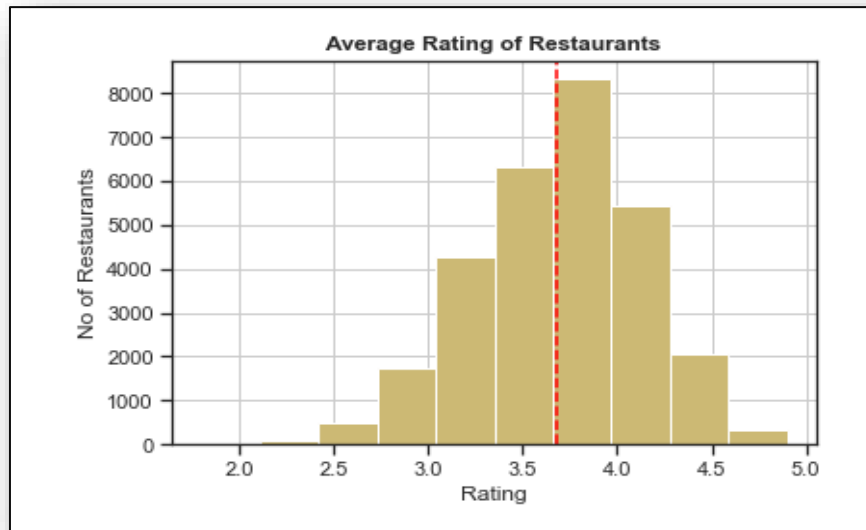


Figure 4.10 Average Rating of Restaurants

Observation :

- Average rating is 3.6

j) rating vs online order

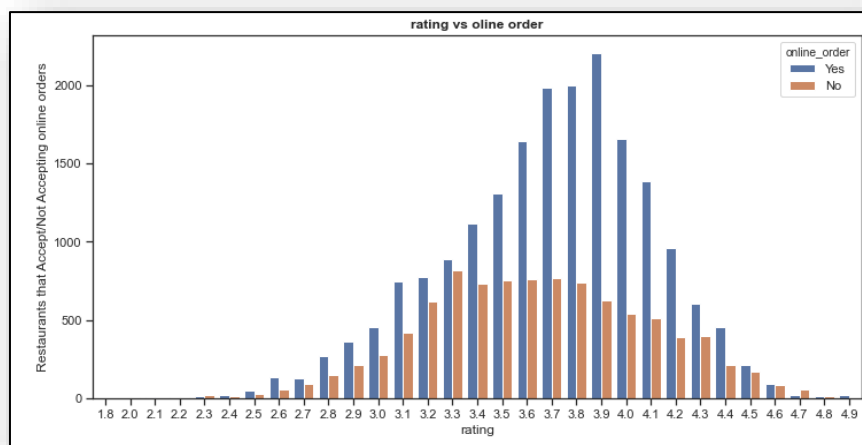


Figure 4.11 Rating Vs online order

Observation:

- It is seen that restaurants accepting online order has comparatively good rating than the restaurant which doesn't.

k) Heatmap

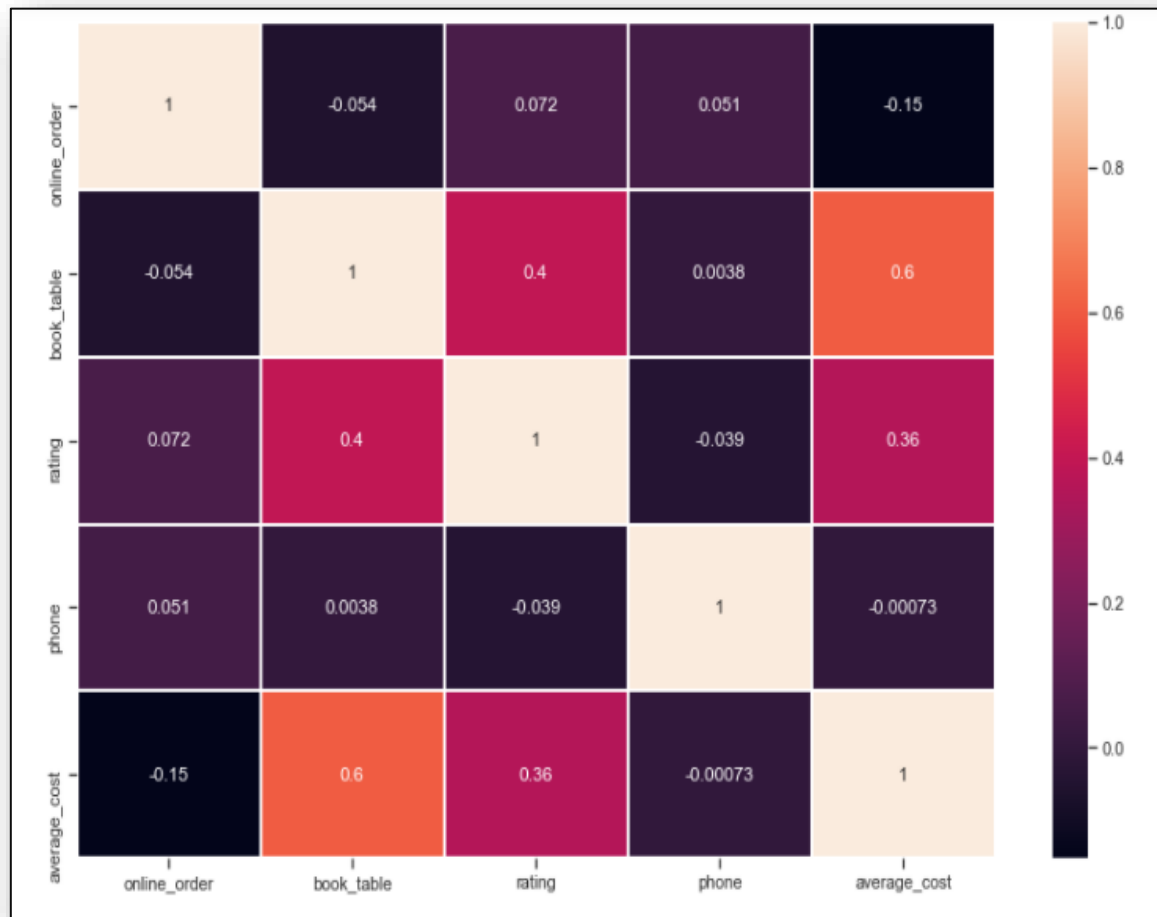


Figure 4.12 Heatmap

Observation:

- we can see that average_cost and book_table have larger impact on rating

Algorithm Selection:

We planned to apply 4 machine learning algorithms on our data set. They were

- i) Linear Regression
- ii) Decision Tree Regressor
- iii) Random Forest Regressor
- iv) XGBoost

❖ Linear Regression

Linear Regression is the oldest, simple and widely used supervised machine learning algorithm for predictive analysis. It's a method to predict a **target variable** by fitting the best linear relationship between the dependent and independent variable.

$$Y = X_1 + X_2 + X_3$$

The diagram illustrates the components of the linear regression equation $Y = X_1 + X_2 + X_3$. A downward arrow points from Y to the label 'Dependent Variable'. A bracket underneath $X_1 + X_2 + X_3$ has a downward arrow pointing to the label 'Independent Variable'.

Dependent Variable	Independent Variable
Outcome Variable	Predictor Variable
Response Variable	Explanatory Variable

What is the Best Fit?

It can be of any shape depending on the number of independent variables (a point on the axis, a line in two dimensions, a plane in three dimensions, or a hyperplane in higher dimensions).

Least Squares Method:

The *best fit* is done by making sure that the sum of all the distances between the shape and the actual observations at each point is as small as possible. The fit of the shape is “best” in the sense that no other position would produce less error given the choice of shape.

❖ Decision Tree Regressor:

The decision tree model is very good at handling tabular data with numerical features, or categorical features with fewer than hundreds of categories. Unlike linear models, decision trees are able to capture non-linear interaction between the features and the target. Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node has two or more branches, each representing values for the attribute tested. Leaf node represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called **root node**.

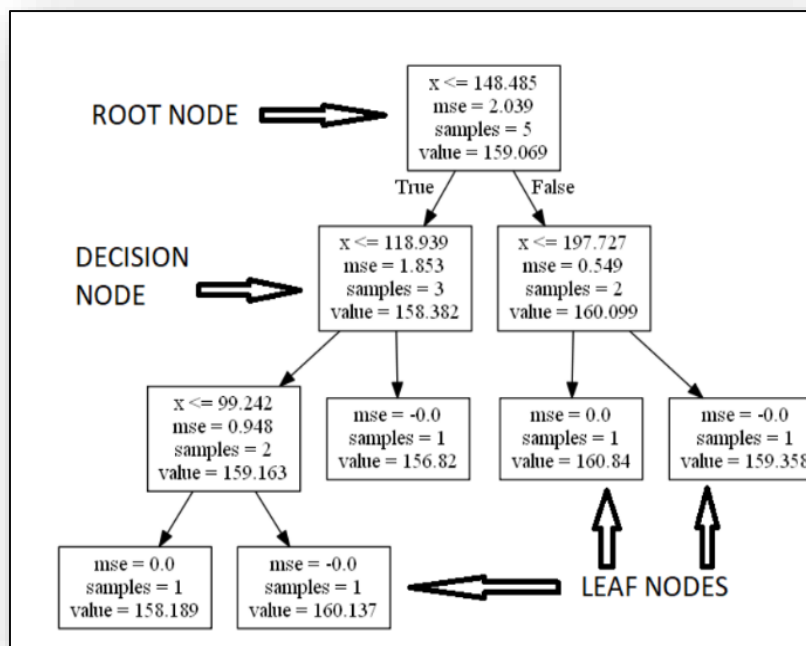


Figure 4.13 Decision Tree

❖ Random Forest:

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called **Bootstrap Aggregation**, commonly known as **bagging**. Bagging, in the Random Forest method, involves training each decision tree on a different data sample where sampling is done with replacement. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

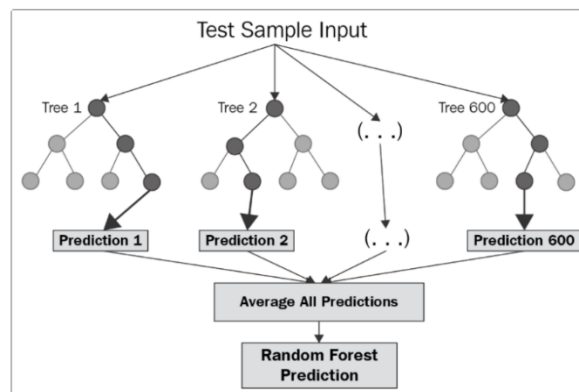


Figure 4.14 Random Forest Regressor

❖ XGBoost:

XGBoost is a popular and efficient open-source implementation of the gradient boosted trees algorithm. Gradient boosting is a supervised learning algorithm, which attempts to accurately predict a target variable by combining the estimates of a set of simpler, weaker models. **Gradient Boosting** specifically is an approach where new models are trained to predict the residuals (i.e., errors) of prior models.

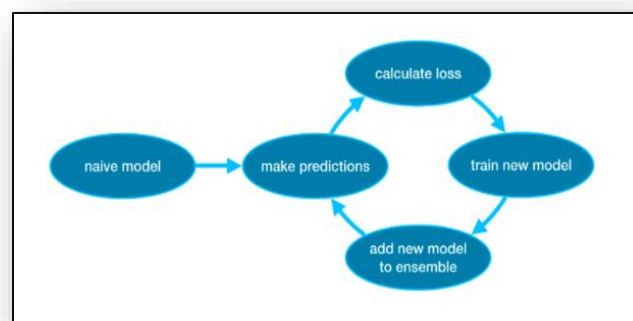


Figure 4.15 XGBoost

Model Building:

```
Linear Regression

In [93]: regressor = LinearRegression()

regressor.fit(X_train,y_train)
y_pred = regressor.predict(X_test)
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
r2score.append(r2_score(y_test, y_pred))
algo.append("linear_reg")
tune.append("without")
print("r2 score: ",r2_score(y_test, y_pred))
print("MSE : ",mean_squared_error(y_test, y_pred) )

r2 score: 0.3299737655132876
MSE : 0.1283790487099604
```

Figure 4.16 Linear Regression without tuning

Linear Regression without Tuning :

- R2 Score:0.32
- MSE:0.12

```
In [94]: #Linear regression using cross validation
regressor = LinearRegression()

kfold = KFold(n_splits=5, random_state=2021,shuffle=True)
results = cross_val_score(regressor, X, y, cv=kfold,
                           scoring='r2')

R2 = results
print(R2)
print("R-Squared: %.2f" % (R2.mean()))
r2score.append(R2.mean())
algo.append("linear_reg")
tune.append("with")

[0.32857244 0.31616622 0.32689985 0.31896078 0.2998541 ]
R-Squared: 0.32
```

Figure 4.17 Linear Regression with tuning

Linear Regression With Tuning using cross validation:

- R2 score: 0.32

Decision tree

```
In [95]: #Decision tree
from sklearn.tree import DecisionTreeRegressor
clf = DecisionTreeRegressor(random_state=2020)
clf2 = clf.fit(X_train, y_train)
y_pred = clf2.predict(X_test)

from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
print("r2 score: ", r2_score(y_test, y_pred))
print("MSE : ", mean_squared_error(y_test, y_pred) )

algo.append("decision_tree")
tune.append("without")
r2score.append(r2_score(y_test, y_pred))

r2 score: 0.7289471360191726
MSE : 0.051934546793718124
```

Figure 4.18 Decision Tree without tuning

Decision Tree Without Tuning:

- R2 Score: 0.72
- MSE: 0.05

```
In [96]: #tuning Decision tree
depth_range = np.linspace(10,1000,10)

parameters = dict(max_depth=depth_range)
from sklearn.model_selection import KFold
kfold = KFold(n_splits=5, random_state=2021, shuffle=True)

clf = DecisionTreeRegressor(random_state=2020)
cv = RandomizedSearchCV(clf, param_distributions=parameters,
                        cv=kfold, scoring='r2', n_iter=20)

cv.fit(X,y)
# Best Parameters
print(cv.best_params_)
print(cv.best_score_)
cv.best_estimator_
algo.append("decision_tree")
tune.append("with")
r2score.append(cv.best_score_)

{'max_depth': 120.0}
0.7637340621733845
```

Figure 4.19 Decision Tree with tuning

Decision Tree With Tuning using Randomized Search CV:

- R2 Score: 0.76

```
Random Forest Regressor

In [97]: #Random Forest
model_rf = RandomForestRegressor(random_state=2021)
model_rf.fit( X_train , y_train )
y_pred = model_rf.predict(X_test)

from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
print("r2 score: ",r2_score(y_test, y_pred))
print("MSE : ",mean_squared_error(y_test, y_pred))
algo.append("Random_Forest")
tune.append("without")
r2score.append(r2_score(y_test, y_pred))

r2 score: 0.7953052907093271
MSE : 0.039220124081902084
```

Figure 4.20 Random Forest Regressor without tuning

Random Forest Regressor Without Tuning:

- R2 score : 0.79
- MSE : 0.03

```
In [98]: #hyper parameter tuning for Random Forest

scores = ["r2","neg_mean_squared_error"]
n_est_range = [100,200,300,350,400,500]

parameters = {'n_estimators':n_est_range,
              'max_features': np.arange(10,30)
              }

from sklearn.model_selection import KFold
kfold = KFold(n_splits=5, random_state=2021,shuffle=True)
model_rf = RandomForestRegressor(random_state=2021)

cv = RandomizedSearchCV(model_rf, param_distributions=parameters,cv=kfold,scoring='r2',n_iter=20)
cv.fit( X, y )
results_df = pd.DataFrame(cv.cv_results_ )
print(cv.best_params_)
print(cv.best_score_)
print(cv.best_estimator_)

algo.append("Random_Forest")
tune.append("with")
r2score.append(cv.best_score_)

{'n_estimators': 400, 'max_features': 14}
0.829941818061946
```

Figure 4.21 Random Forest Regressor with tuning

Random Forest Regressor with Tuning using Randomized Search CV:

- R2 score : 0.82

```

XGBoost

In [99]: #XGBoost
from xgboost import XGBRegressor
clf = XGBRegressor(random_state=2021)
clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)

from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
print("r2 score: ",r2_score(y_test, y_pred))
print("MSE : ",mean_squared_error(y_test, y_pred))

algo.append("XGBoost")
tune.append("without")
r2score.append(r2_score(y_test, y_pred))

[02:30:24] WARNING: src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
r2 score: 0.3478854877922186
MSE : 0.12510839107703788

```

Figure 4.22 XGBoost Regressor without tuning

XGboost Regressor Without Tuning :

- $r2_score = 0.34$
- $MSE = 0.12$

```

In [100]: #xgboost tuning
lr_range = [0.1, 0.2, 0.25, 0.3]
n_est_range = [10, 20, 30, 50, 100]
md_range = [8, 10, 15, 20]

parameters = dict(learning_rate=lr_range,
                   n_estimators=n_est_range,
                   max_depth=md_range)

xg = XGBRegressor(random_state=2021, silent = True)
cv = RandomizedSearchCV(xg, param_distributions=parameters,
                       cv=kfold, scoring='r2', n_iter=20, verbose=1)

cv.fit(X,y)
print(cv.best_params_)
print(cv.best_score_)

algo.append("XGBoost")
tune.append("with")
r2score.append(cv.best_score_)

Fitting 5 folds for each of 20 candidates, totalling 100 fits
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done 100 out of 100 | elapsed: 292.5min finished
{'n_estimators': 100, 'max_depth': 20, 'learning_rate': 0.25}
0.8112592054348223

```

Figure 4.23 XGBoost Regressor with tuning

XGBoost Regressor With Tuning :

- $R2\ score : 0.81$

As observed, we are getting lower MSE i.e., 0.03 and higher r2 score i.e., 0.79 for random forest regressor. So, we went ahead and did hyper parameter tuning for random forest regressor then we are getting r2 score i.e., 0.82 So we decided to stop here.

Result :

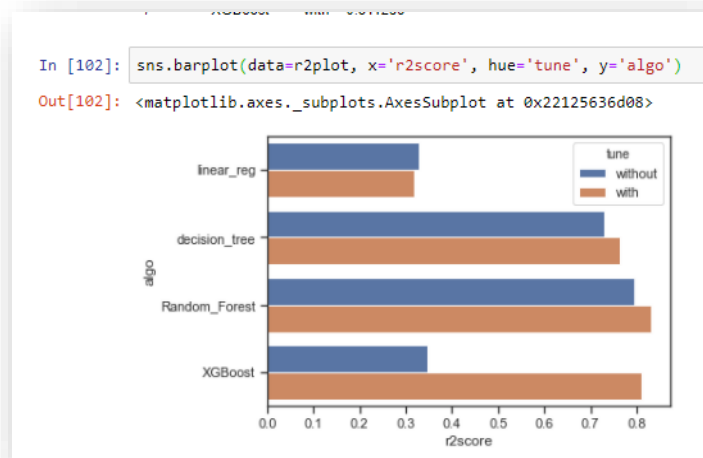


Figure 4.24 r2 score plot of all model

Observation:

- Without any tuning Random forest is giving r2 score as 0.79 and MSE as 0.03 So went ahead and tuned random forest and it is giving r2score: 0.82

Chapter 5

TESTING

5.1 Saving the model

```
Testing

In [119]: #Random Forest
model_rf = RandomForestRegressor(max_features=20, n_estimators=350, random_state=2021)
model_rf.fit(X_train, y_train)
y_pred = model_rf.predict(X_test)

In [120]: #saving the model
import pickle
filename='model_save'
pickle.dump(model_rf,open(filename,'wb'))

In [121]: model=pickle.load(open('model_save','rb'))
```

Figure 5.1 saving model

5.2 Testing on dummy data

```
Testing on Dummy Data

In [136]: testing= [1,1,2,800,"northindian,mughlai,chinese","casualdining","Banashankari"]

In [174]: r=model.predict(test_df.transpose())

In [175]: print("rating is : ",round(r[0],2))
rating is : 3.93
```

Figure 5.2 Testing and prediction on dummy data

5.3 Predicting the ratings of new Restaurants

Actual data of new restaurants:

Predicting the rating of New Restaurants

```
In [118]: #displaying restaurants where rating is NEW
new_rest.head(3)
```

Out[118]:

	name	online_order	book_table	rating	phone	location	rest_type	dish_liked	cuisines	average_cost	city
72	Spicy Tandoor	No	No	NEW	+91 8050884222	Banashankari	Quick Bites	NaN	North Indian	150	Banashankari
75	Om Sri Vinayaka Chats	No	No	NEW	+91 8553208035	Banashankari	Takeaway, Delivery	NaN	Street Food, Fast Food	500	Banashankari
110	Hari Super Sandwich	No	No	NEW	+91 9886722163	Banashankari	Takeaway, Delivery	NaN	Sandwich, Pizza, Beverages	200	Banashankari

Figure 5.3 actual data of new Restaurants

Pre-processed data of new restaurants:

To fit in the model, we have to transform the data according to the trained data

```
In [149]: #pre-processed data
test_data.head(3)
```

Out[149]:

	0	1	2	3	4	5	6	7	8	9	...	140	141	142	1
0	-0.870681	-0.603570	0.613631	0.376592	-0.324316	-0.118826	0.527635	-0.335573	-0.115829	-0.501381	...	-2.587035e-16	-1.330656e-15	1.099619e-16	1.101974
1	-0.411652	0.243378	-2.385461	-1.764152	0.144246	-0.180677	0.874484	0.406472	-0.091535	-0.811792	...	3.590056e-16	-5.796521e-16	1.033578e-16	1.227445
2	-0.528059	1.367496	-2.605673	-1.309503	0.862405	-0.735032	1.708646	1.048035	-1.204609	-0.680684	...	7.135759e-17	-5.393813e-16	-3.949920e-16	-2.911185

3 rows × 150 columns

Figure 5.4 Preprocess data of new restaurants

Predictions:

```
In [151]: #predictions on test data
r=model.predict(test_data)
print("rating is : ",r)

rating is : [3.55878171 3.73476213 3.76396136 ... 3.52348401 3.72130485 3.53591743]

In [169]: for name,rating in zip(names,r):
           print(name.ljust(50), "Predicted Rating : ",round(rating,1))
```

Spicy Tandoor	Predicted Rating : 3.6
Om Sri Vinayaka Chats	Predicted Rating : 3.7
Hari Super Sandwich	Predicted Rating : 3.8
Roll Magic Fast Food	Predicted Rating : 3.7
Foodlieious Multi Cuisine	Predicted Rating : 3.5
Thanishka Nati And Karavali Style	Predicted Rating : 3.6
Melting Melodies	Predicted Rating : 3.8
New Indraprasta	Predicted Rating : 3.6
Bitez	Predicted Rating : 3.6
SSV Upahar	Predicted Rating : 3.5
Sai Super Sandwich	Predicted Rating : 3.6
Right Pizza	Predicted Rating : 3.7
The Sip Shop	Predicted Rating : 3.7
Sri Banashankari Upahara	Predicted Rating : 3.5
Hyderabad Express	Predicted Rating : 3.6
Biryani Royals Garden	Predicted Rating : 3.4
Multi Snackv Huh	Predicted Rating : 3.6

Figure 5.5 Predicted Rating

Chapter 6

CONCLUSION

This project predicts the rating of a newly opened restaurant based on factors such as online order, booking facility, cuisines, restaurant type, location etc. Data is analysed with the help of graphs. The dataset has 51717 rows and 17 columns. Some columns are multi-valued columns for e.g. in cuisines we have multiple cuisines such as Chinese, Thai, continental so split these columns into unique bins. Finally, after complete pre-processing the dataset had 29024 rows and 180 columns. For prediction various machine learning algorithms were used, such as linear regression, Decision tree regressor, Random forest regressor and XGBoost. For evaluation, metrics r2score and MSE are used. Random Forest is chosen as the final model as it performed better than the rest by giving an r2score of 0.83 and MSE of 0.03.

Chapter 7

FUTURE SCOPE

- User interface can be made which takes inputs for a new restaurant
- Neural network can be used to check if it performs better than Random Forest
- Can design a similar system by taking data of all the cities.

REFERENCES

1. <https://www.kaggle.com/himanshupoddar/zomato-bangalore-restaurants>
2. <https://towardsdatascience.com>