Restaurant Rating Prediction

Technologies : Machine Learning Technology

Domain : E-commerce

Project Difficulties level : Intermediate

The main goal of this project is to perform extensive Exploratory Data Analysis(EDA) on the Zomato Dataset and build an appropriate Machine Learning Model that will help various Zomato Restaurants to predict their respective Ratings based on certain features.

A Look into Database

The basic idea of analyzing the Zomato dataset is to get a fair idea about the factors affecting the aggregate rating of each restaurant, establishment of different types of restaurant at different places, 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 has'nt 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. These kind of analysis can be done using the data, by studying different factors.

```
#importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.graph_objs as go
import plotly.offline as py
import seaborn as sns

import matplotlib.ticker as mtick
plt.style.use('fivethirtyeight')
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import  ExtraTreesRegressor
from sklearn.model_selection import train_test_split

import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

load and reading the dataset

```
data=pd.read_csv('/kaggle/input/zomato-bangalore-restaurants/zomato.csv')
```

```
data.head()
```

	url	address	name	online_order	bool
0	https://www.zomato.com/bangalore/jalsa-banasha...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	
1	https://www.zomato.com/bangalore/spice-elephan...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	
2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	
3	https://www.zomato.com/bangalore/addhuri-udupi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	
4	https://www.zomato.com/bangalore/grand-village...	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	

```
data.shape
```

```
(51717, 17)
```

```
data.dtypes
```

```
url                            object
address                        object
name                           object
online_order                   object
book_table                     object
rate                           object
votes                           int64
phone                          object
location                       object
rest_type                      object
dish_liked                     object
cuisines                       object
approx_cost(for two people)    object
reviews_list                   object
menu_item                      object
listed_in(type)                object
listed_in(city)                object
dtype: object
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51717 entries, 0 to 51716
Data columns (total 17 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   url                          51717 non-null  object
 1   address                      51717 non-null  object
 2   name                         51717 non-null  object
 3   online_order                 51717 non-null  object
 4   book_table                   51717 non-null  object
 5   rate                         43942 non-null  object
 6   votes                        51717 non-null  int64 
 7   phone                        50509 non-null  object
 8   location                     51696 non-null  object
 9   rest_type                    51490 non-null  object
 10  dish_liked                   23639 non-null  object
 11  cuisines                     51672 non-null  object
 12  approx_cost(for two people)  51371 non-null  object
 13  reviews_list                 51717 non-null  object
 14  menu_item                    51717 non-null  object
 15  listed_in(type)              51717 non-null  object
 16  listed_in(city)              51717 non-null  object
dtypes: int64(1), object(16)
memory usage: 6.7+ MB
```

looks like we got some null values

Null values in the dataset

```
data.isnull().sum()
```

```
url                                0
address                            0
```

```
name                            0
online_order                    0
book_table                      0
rate                         7775
votes                           0
phone                        1208
location                       21
rest_type                     227
dish_liked                  28078
cuisines                       45
approx_cost(for two people)   346
reviews_list                    0
menu_item                       0
listed_in(type)                 0
listed_in(city)                 0
dtype: int64
```

Remove unnecessary features(columns)

```
df=data.drop(['url','phone'],axis=1) #Dropping the column like "phone" and "url" and savir
```

Duplicate values in the dataset

```
df.duplicated().sum()
```

```
43
```

Remove duplicates in dataset

```
df.drop_duplicates(inplace=True)
```

Lets check if the duplicates now are zero or not

```
df.duplicated().sum()
```

```
0
```

Remove Null values in dataset

```
df.dropna(how='any',inplace=True)
df.isnull().sum()
```

```
address                     0
name                        0
online_order                0
book_table                  0
rate                        0
```

```
votes                          0
location                       0
rest_type                      0
dish_liked                     0
cuisines                       0
approx_cost(for two people)    0
reviews_list                   0
menu_item                      0
listed_in(type)                0
listed_in(city)                0
dtype: int64
```

Renaming columns according to our convienence

```
df.columns
```

```
Index(['address', 'name', 'online_order', 'book_table', 'rate', 'votes',
       'location', 'rest_type', 'dish_liked', 'cuisines',
       'approx_cost(for two people)', 'reviews_list', 'menu_item',
       'listed_in(type)', 'listed_in(city)'],
      dtype='object')
```

```
df = df.rename(columns={'approx_cost(for two people)':'cost','listed_in(type)':'type',
                        'listed_in(city)':'city'})
df.columns
```

```
Index(['address', 'name', 'online_order', 'book_table', 'rate', 'votes',
       'location', 'rest_type', 'dish_liked', 'cuisines', 'cost',
       'reviews_list', 'menu_item', 'type', 'city'],
      dtype='object')
```

```
df.head()
```

	address	name	online_order	book_table	rate	votes	location	rest_
0	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1/5	775	Banashankari	Ca D
1	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1/5	787	Banashankari	Ca D
2	1112, Next to KIMS Medical College, 17th	San Churro	Yes	No	3.8/5	918	Banashankari	Ca

Changing datatype of cost as float

```
df['cost'].unique()
```

```
array(['800', '300', '600', '700', '550', '500', '450', '650', '400',
       '750', '200', '850', '1,200', '150', '350', '250', '1,500',
       '1,300', '1,000', '100', '900', '1,100', '1,600', '950', '230',
       '1,700', '1,400', '1,350', '2,200', '2,000', '1,800', '1,900',
       '180', '330', '2,500', '2,100', '3,000', '2,800', '3,400', '40',
       '1,250', '3,500', '4,000', '2,400', '1,450', '3,200', '6,000',
       '1,050', '4,100', '2,300', '120', '2,600', '5,000', '3,700',
       '1,650', '2,700', '4,500'], dtype=object)
```

```
df['cost'] = df['cost'].apply(lambda x: x.replace(',','')) #Using lambda function to repla
df['cost'] = df['cost'].astype(float)
##replacing the "," with nothing and converting the results to float
```

```
print(df['cost'].unique())
df.dtypes
```

```
[ 800.  300.  600.  700.  550.  500.  450.  650.  400.  750.  200.  850.
 1200.  150.  350.  250. 1500. 1300. 1000.  100.  900. 1100. 1600.  950.
  230. 1700. 1400. 1350. 2200. 2000. 1800. 1900.  180.  330. 2500. 2100.
 3000. 2800. 3400.   40. 1250. 3500. 4000. 2400. 1450. 3200. 6000. 1050.
 4100. 2300.  120. 2600. 5000. 3700. 1650. 2700. 4500.]
address          object
name             object
online_order     object
book_table       object
rate             object
votes             int64
location         object
rest_type        object
dish_liked       object
cuisines         object
cost            float64
reviews_list     object
menu_item        object
type             object
city             object
dtype: object
```

Removing '/5' from Rates and getting rid of 'NEW' as an entry

```
df['rate'].unique()
```

```
array(['4.1/5', '3.8/5', '3.7/5', '4.6/5', '4.0/5', '4.2/5', '3.9/5',
       '3.0/5', '3.6/5', '2.8/5', '4.4/5', '3.1/5', '4.3/5', '2.6/5',
       '3.3/5', '3.5/5', '3.8 /5', '3.2/5', '4.5/5', '2.5/5', '2.9/5',
       '3.4/5', '2.7/5', '4.7/5', 'NEW', '2.4/5', '2.2/5', '2.3/5',
       '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5', '4.1 /5', '2.9 /5',
       '2.7 /5', '2.5 /5', '2.6 /5', '4.5 /5', '4.3 /5', '3.7 /5',
```

```
            '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '3.4 /5', '3.6 /5',
            '3.3 /5', '4.6 /5', '4.9 /5', '3.2 /5', '3.0 /5', '2.8 /5',
            '3.5 /5', '3.1 /5', '4.8 /5', '2.3 /5', '4.7 /5', '2.4 /5',
            '2.1 /5', '2.2 /5', '2.0 /5', '1.8 /5'], dtype=object)
```

```
df = df.loc[df.rate !='NEW']
```

```
df['rate'].unique()
```

```
array(['4.1/5', '3.8/5', '3.7/5', '4.6/5', '4.0/5', '4.2/5', '3.9/5',
       '3.0/5', '3.6/5', '2.8/5', '4.4/5', '3.1/5', '4.3/5', '2.6/5',
       '3.3/5', '3.5/5', '3.8 /5', '3.2/5', '4.5/5', '2.5/5', '2.9/5',
       '3.4/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5', '4.8/5',
       '3.9 /5', '4.2 /5', '4.0 /5', '4.1 /5', '2.9 /5', '2.7 /5',
       '2.5 /5', '2.6 /5', '4.5 /5', '4.3 /5', '3.7 /5', '4.4 /5',
       '4.9/5', '2.1/5', '2.0/5', '1.8/5', '3.4 /5', '3.6 /5', '3.3 /5',
       '4.6 /5', '4.9 /5', '3.2 /5', '3.0 /5', '2.8 /5', '3.5 /5',
       '3.1 /5', '4.8 /5', '2.3 /5', '4.7 /5', '2.4 /5', '2.1 /5',
       '2.2 /5', '2.0 /5', '1.8 /5'], dtype=object)
```

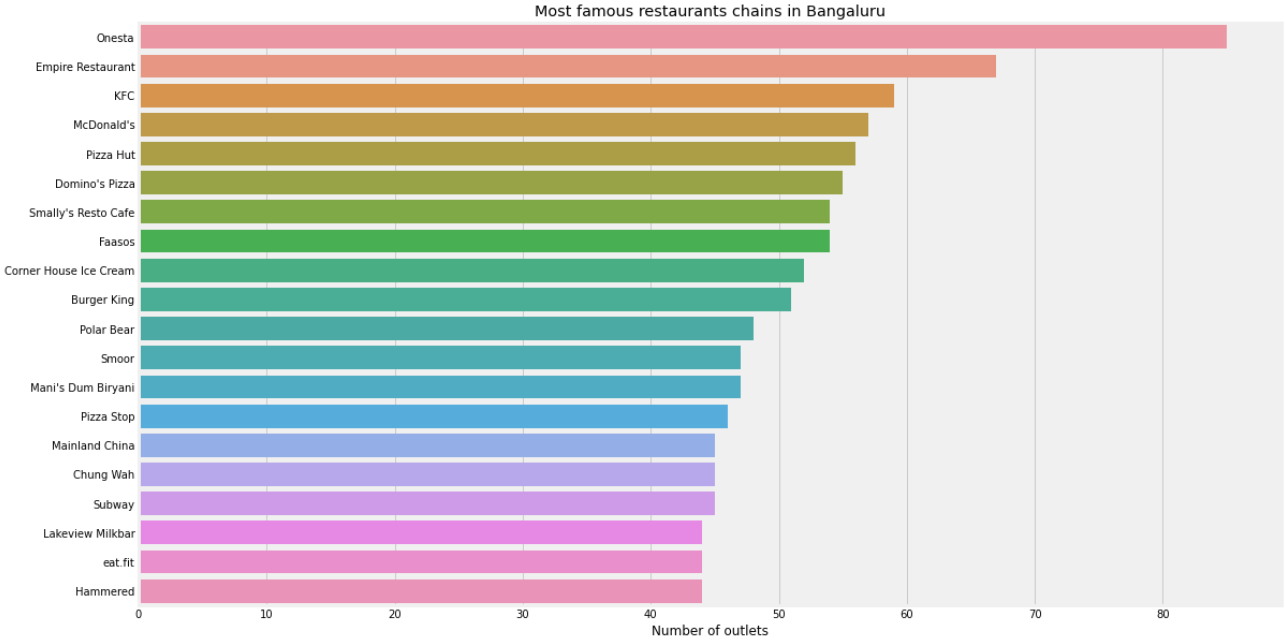
```
df['rate'] = df['rate'].apply(lambda x: x.replace('/5',''))
```

```
df['rate'].unique()
```

```
array(['4.1', '3.8', '3.7', '4.6', '4.0', '4.2', '3.9', '3.0', '3.6',
       '2.8', '4.4', '3.1', '4.3', '2.6', '3.3', '3.5', '3.8 ', '3.2',
       '4.5', '2.5', '2.9', '3.4', '2.7', '4.7', '2.4', '2.2', '2.3',
       '4.8', '3.9 ', '4.2 ', '4.0 ', '4.1 ', '2.9 ', '2.7 ', '2.5 ',
       '2.6 ', '4.5 ', '4.3 ', '3.7 ', '4.4 ', '4.9', '2.1', '2.0', '1.8',
       '3.4 ', '3.6 ', '3.3 ', '4.6 ', '4.9 ', '3.2 ', '3.0 ', '2.8 ',
       '3.5 ', '3.1 ', '4.8 ', '2.3 ', '4.7 ', '2.4 ', '2.1 ', '2.2 ',
       '2.0 ', '1.8 '], dtype=object)
```

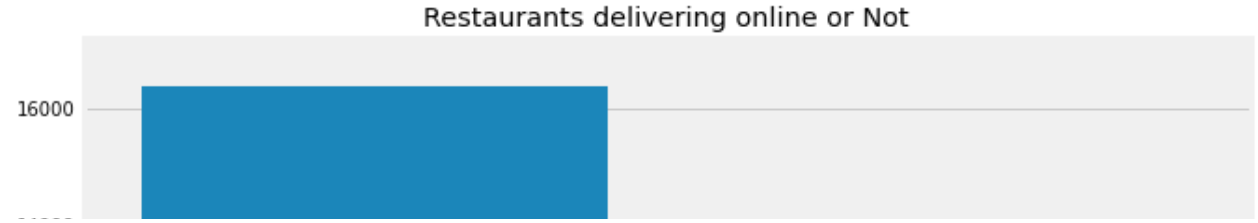
Exploratory Data Analysis

```
plt.figure(figsize=(17,10))
chains=df['name'].value_counts()[:20]
sns.barplot(x=chains,y=chains.index)
plt.title("Most famous restaurants chains in Bangaluru")
plt.xlabel("Number of outlets")
plt.show()
```



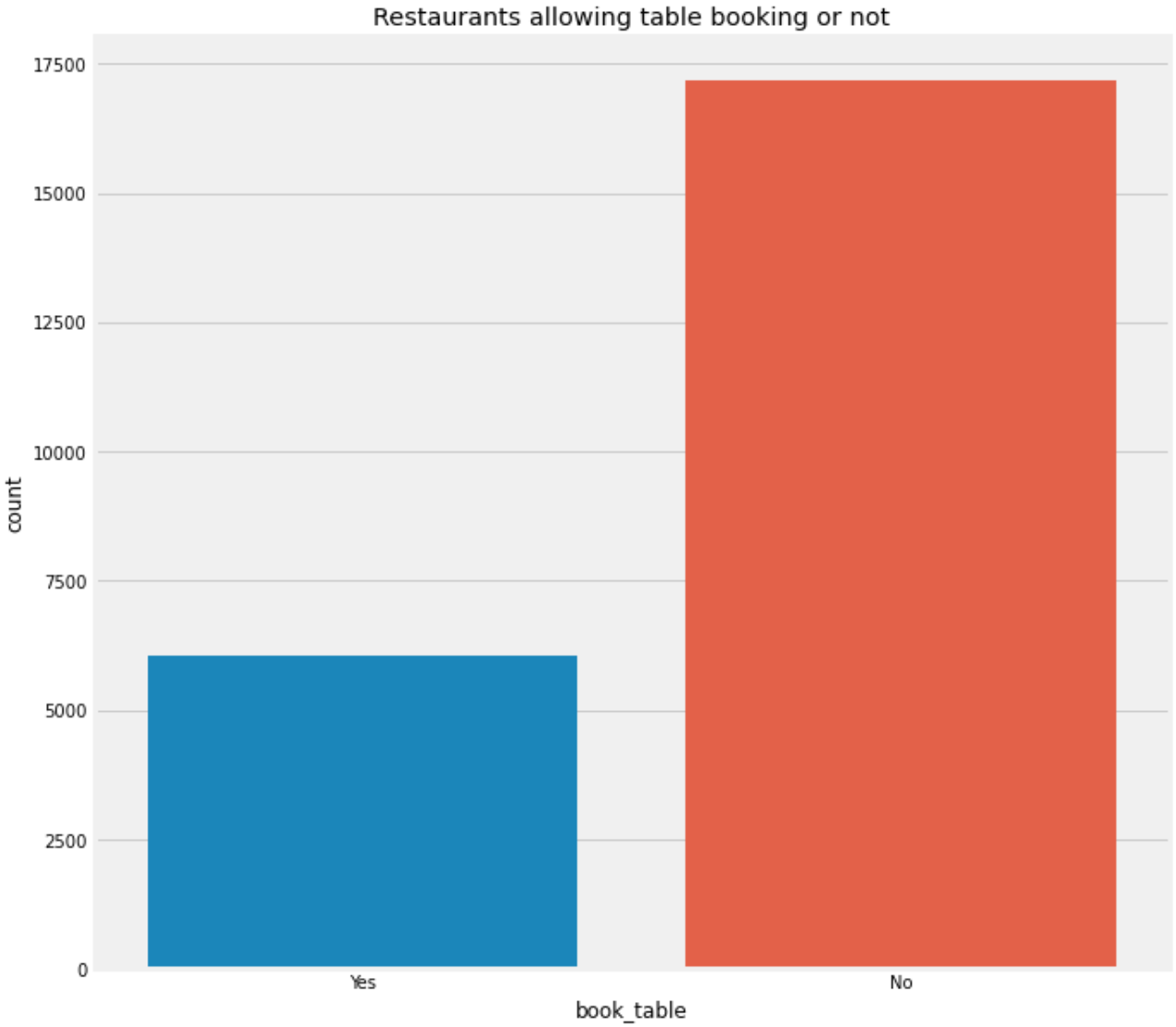

```
sns.countplot(df['online_order'])
fig = plt.gcf()
fig.set_size_inches(10,10)
plt.title('Restaurants delivering online or Not')
```

```
Text(0.5, 1.0, 'Restaurants delivering online or Not')
```

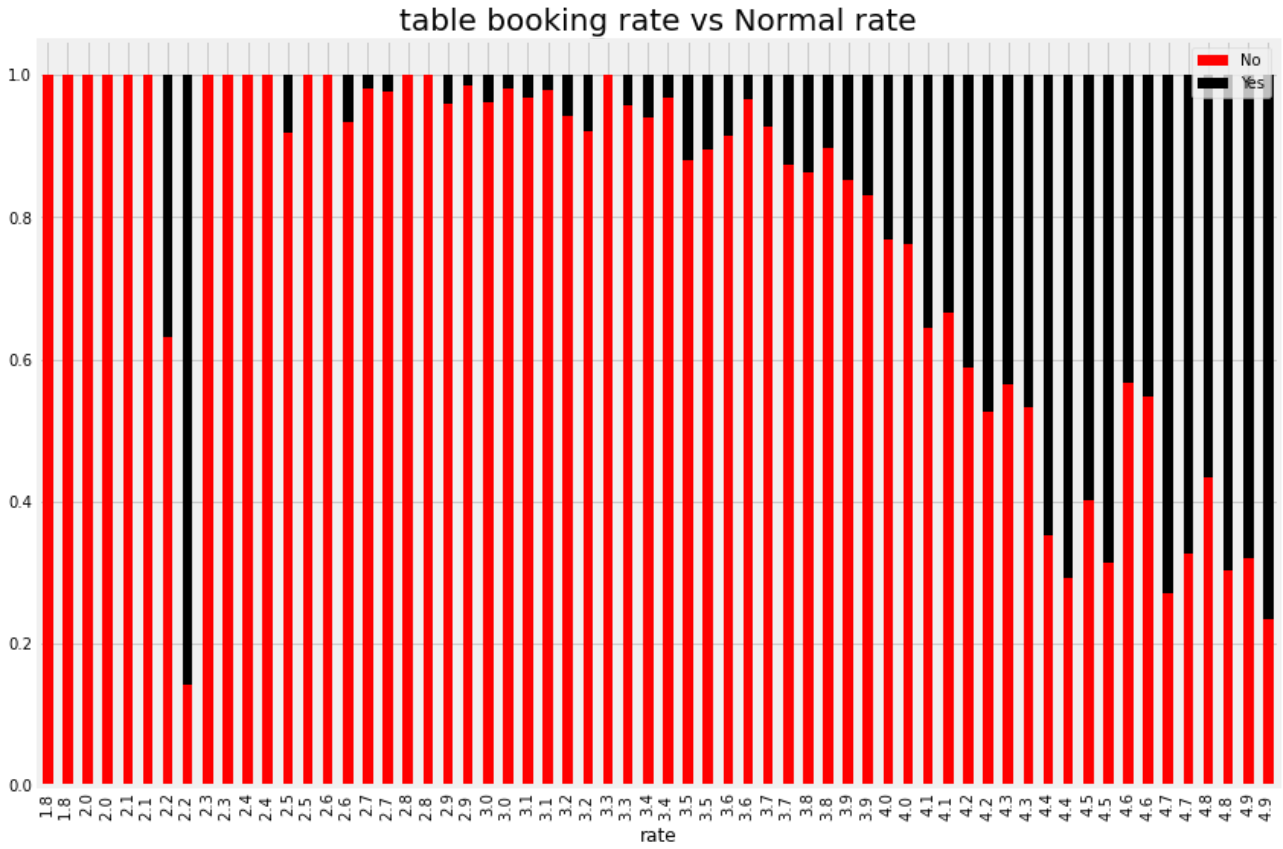



```
sns.countplot(df['book_table'])
fig = plt.gcf()
fig.set_size_inches(10,10)
plt.title('Restaurants allowing table booking or not')
```

```
Text(0.5, 1.0, 'Restaurants allowing table booking or not')
```

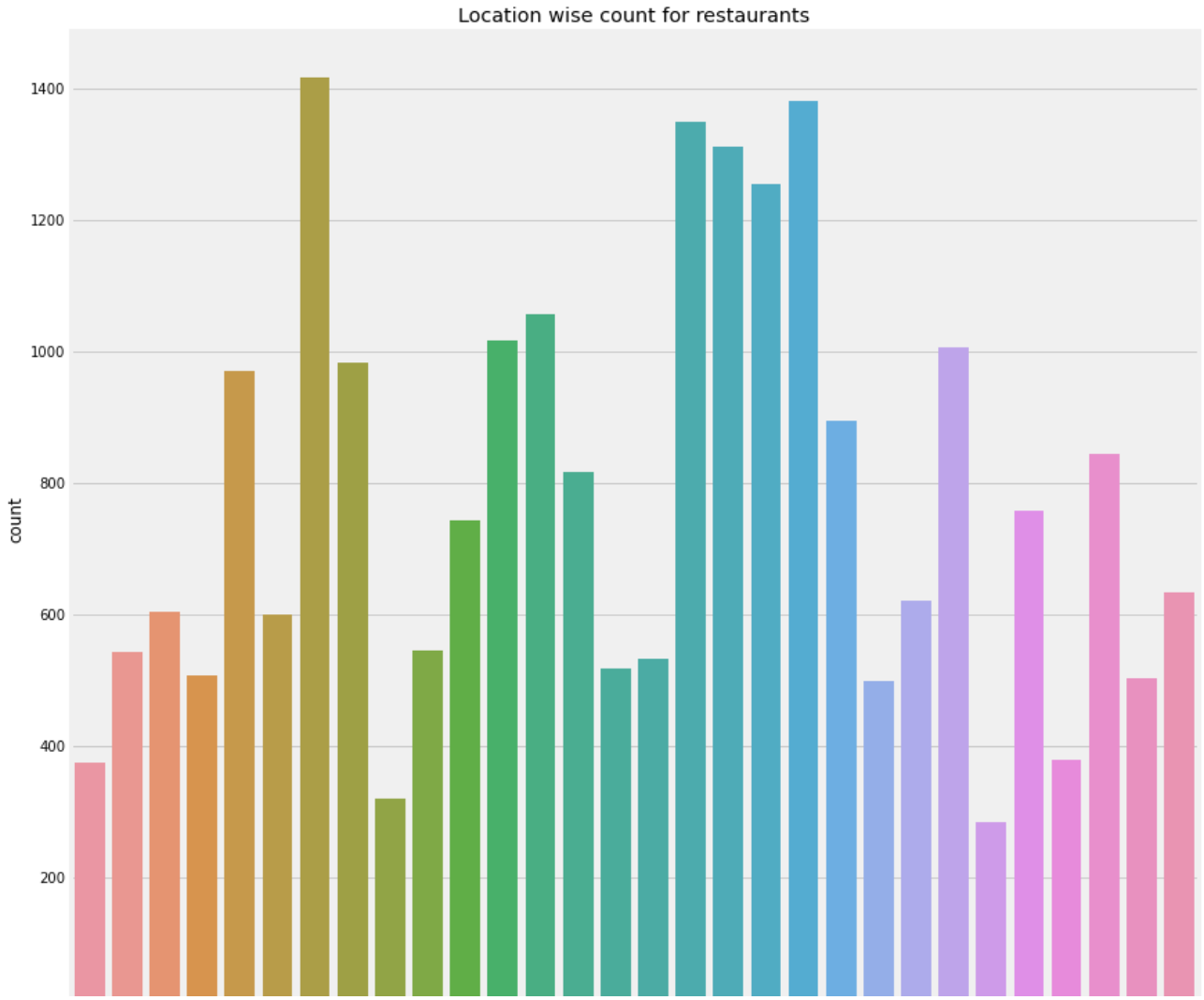



```
plt.rcParams['figure.figsize'] = (13, 9)
Y = pd.crosstab(df['rate'], df['book_table'])
Y.div(Y.sum(1).astype(float), axis = 0).plot(kind = 'bar', stacked = True,color=['red','bl
plt.title('table booking rate vs Normal rate', fontweight = 30, fontsize = 20)
plt.legend(loc="upper right")
plt.show()
```




```
sns.countplot(df['city'])
sns.countplot(df['city']).set_xticklabels(sns.countplot(df['city']).get_xticklabels(), rot
fig = plt.gcf()
fig.set_size_inches(13,13)
plt.title('Location wise count for restaurants')
```

```
Text(0.5, 1.0, 'Location wise count for restaurants')
```



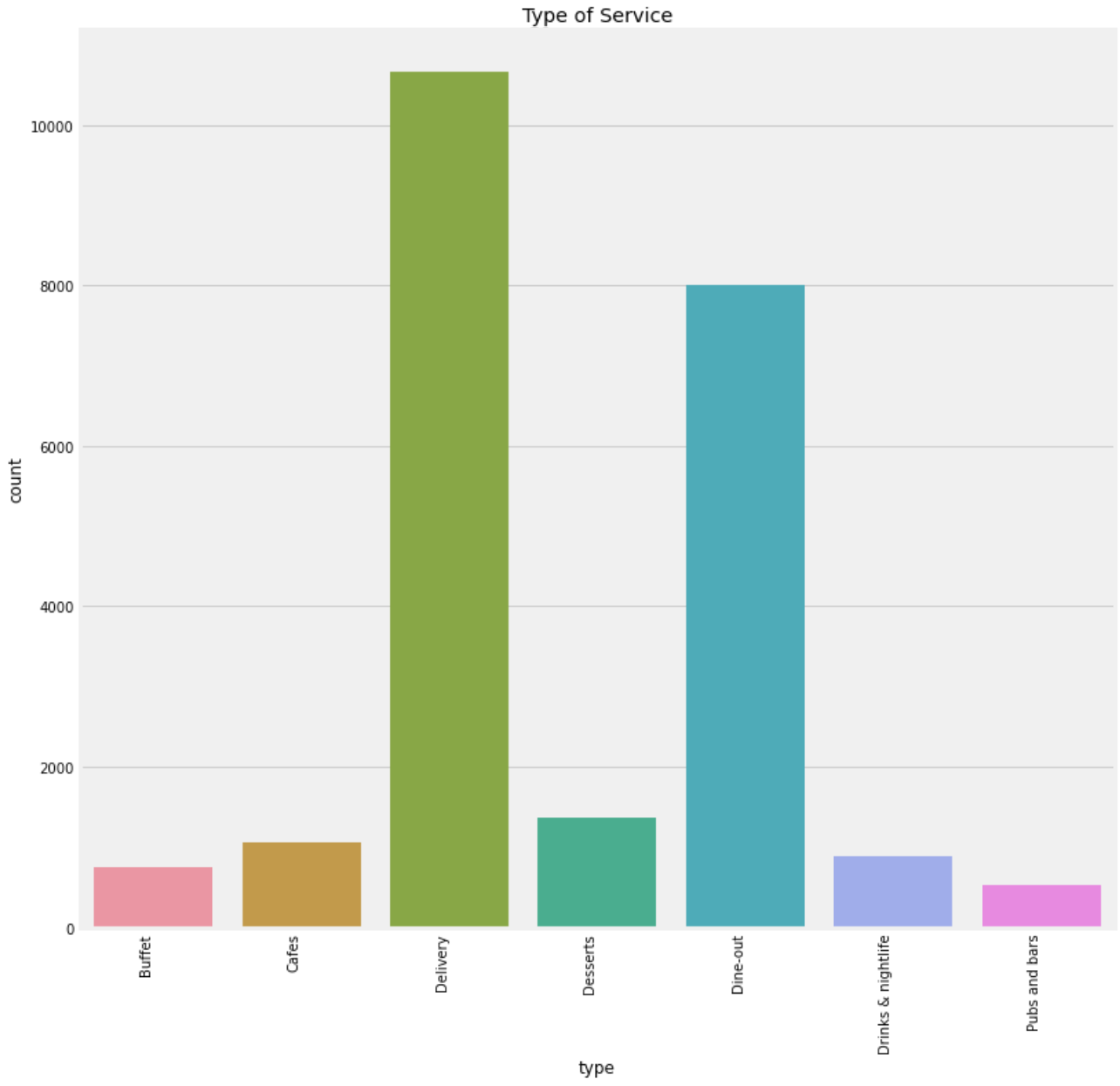

```
plt.figure(figsize=(9,7))
sns.distplot(df['rate'],bins=20)
```

```
<AxesSubplot:xlabel='rate', ylabel='Density'>
```

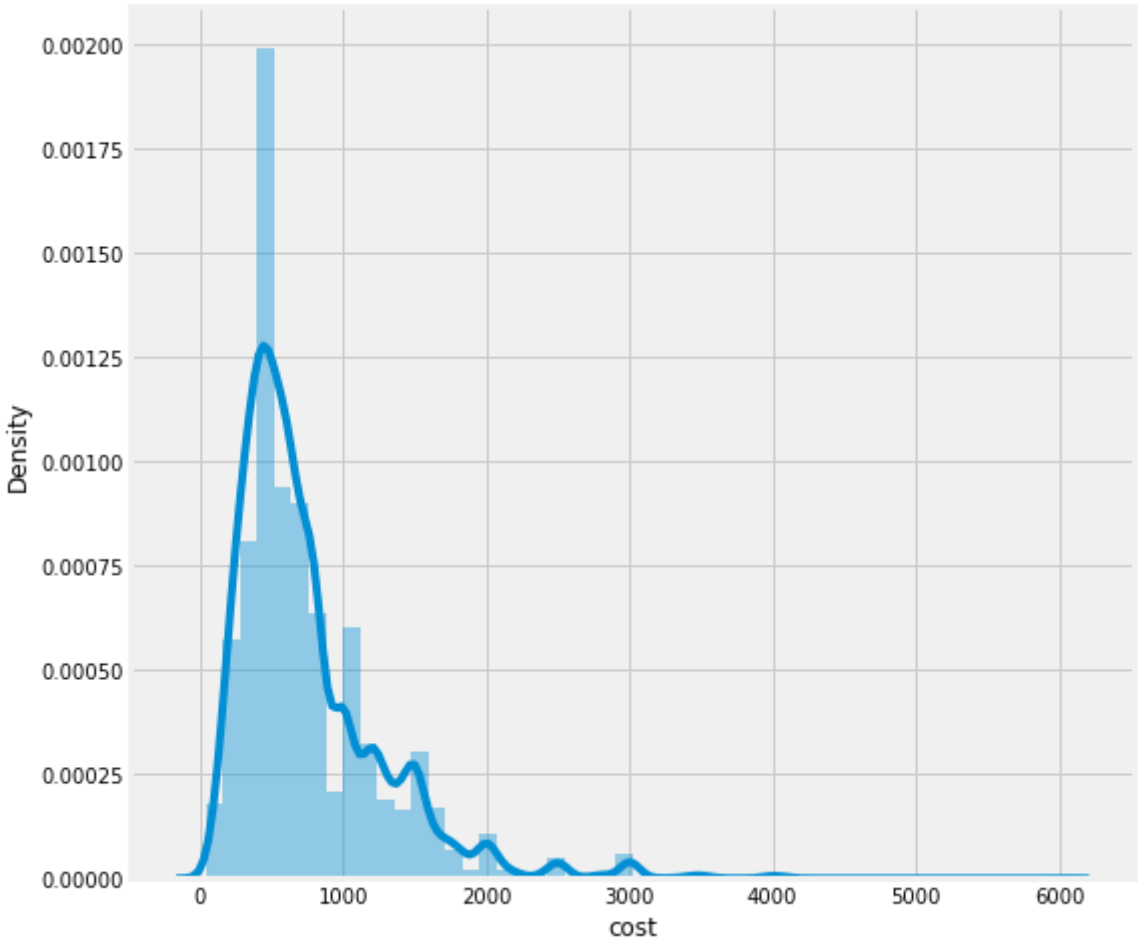
most of the ratings are within 3.5 and 4.5

```
sns.countplot(df['type']).set_xticklabels(sns.countplot(df['type']).get_xticklabels(), rot
fig = plt.gcf()
fig.set_size_inches(12,12)
plt.title('Type of Service')
```

```
Text(0.5, 1.0, 'Type of Service')
```



```
#distribution of charges
plt.figure(figsize=(8,8))
sns.distplot(df['cost'])
plt.show()
```



Most liked dishes

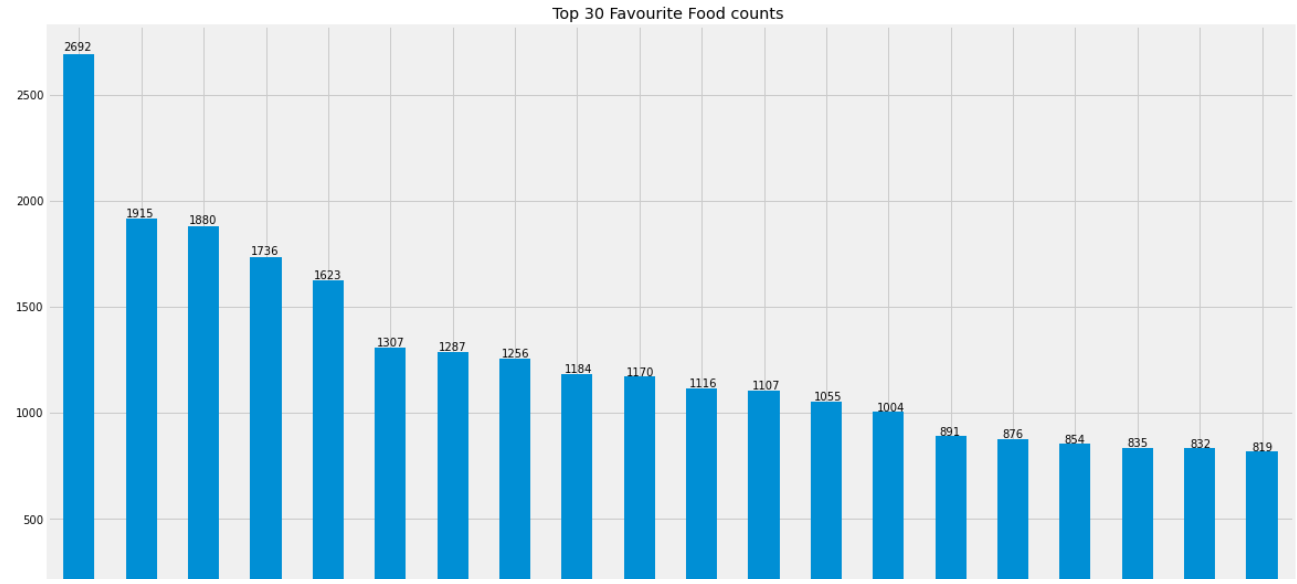
```
import re

df.index=range(df.shape[0])
likes=[]
for i in range(df.shape[0]):
    array_split=re.split(',',df['dish_liked'][i])
    for item in array_split:
        likes.append(item)


df.index=range(df.shape[0])
df.index
```

```
RangeIndex(start=0, stop=23248, step=1)
```

```
print("Count of Most liked dishes in Bangalore")
favourite_food = pd.Series(likes).value_counts()
favourite_food.head(30)
```

```
Count of Most liked dishes in Bangalore
 Pasta              2692
 Pizza              1915
 Cocktails          1880
 Burgers            1736
 Mocktails          1623
 Biryani            1307
 Sandwiches         1287
Burgers             1256
 Coffee             1184
 Nachos             1170
 Fish               1116
 Paratha            1107
 Salads             1055
 Chicken Biryani    1004
Cocktails            891
 Fries               876
 Noodles             854
 Beer                835
 Mutton Biryani      832
 Tea                 819
Coffee               801
 Sandwich            788
 Butter Chicken      782
 Thali               770
Biryani              749
Pizza                747
 Roti                729
 Brownie             726
 Salad               677
 Hot Chocolate       672
dtype: int64
```

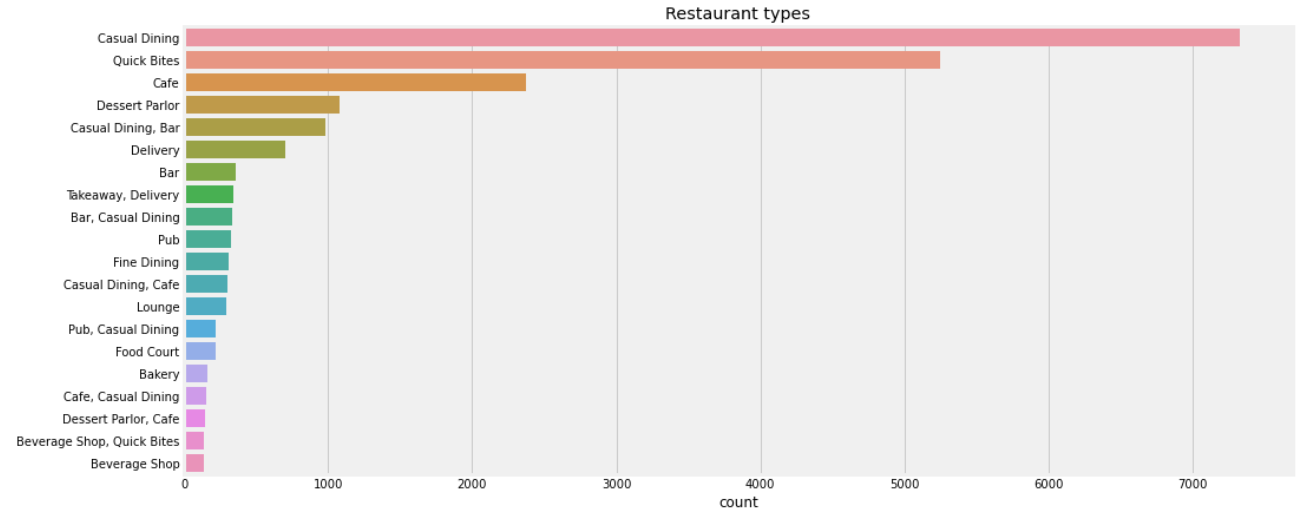
```
ax = favourite_food.nlargest(n=20, keep='first').plot(kind='bar',figsize=(18,10),title = '
for i in ax.patches:
    ax.annotate(str(i.get_height()), (i.get_x() * 1.005, i.get_height() * 1.005))
```



The 5 most liked dishes are Pasta,Pizza,Cocktails,Burgers,and Mocktails

```
plt.figure(figsize=(15,7))
rest=df['rest_type'].value_counts()[:20]
sns.barplot(rest,rest.index)
plt.title("Restaurant types")
plt.xlabel("count")
```

```
Text(0.5, 0, 'count')
```



Casual Dining, Quick Bites and Cafe are the 3 most common types of Restaurants in Bangalore

Convert the online categorical variables into a numeric format

```
df.online_order[df.online_order == 'Yes'] = 1
df.online_order[df.online_order == 'No'] = 0
df.online_order.value_counts()
```

```
1    16378
0     6870
Name: online_order, dtype: int64
```

```
df.online_order = pd.to_numeric(df.online_order)
df.online_order
```

```
0        1
1        1
2        1
3        0
4        0
        ..
23243    1
23244    0
23245    0
23246    0
23247    0
Name: online_order, Length: 23248, dtype: int64
```

Convert the String categorical variables into a numeric format

```
df.book_table[df.book_table == 'Yes'] = 1
df.book_table[df.book_table == 'No'] = 0
df.book_table.value_counts()
```

```
0    17191
1     6057
Name: book_table, dtype: int64
```

```
df.book_table = pd.to_numeric(df.book_table)
df.book_table
```

```
0        1
1        0
2        0
3        0
4        0
        ..
23243    1
23244    0
23245    0
23246    0
23247    1
Name: book_table, Length: 23248, dtype: int64
```

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

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

```
df.head()
```

	address	name	online_order	book_table	rate	votes	location	rest_ty
0	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	1	1	4.1	775	1	
1	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	1	0	4.1	787	1	
	1112, Next to							

```
my_data=df.iloc[:,[2,3,4,5,6,7,9,10,12]]
my_data.to_csv('Zomato_df_komal.csv')
```

Train and test splits of data

```
x = df.iloc[:,[2,3,5,6,7,9,10,12]]
x.head()
```

	online_order	book_table	votes	location	rest_type	cuisines	cost	menu_ite
0	1	1	775	1	20	1386	800.0	504
1	1	0	787	1	20	594	800.0	504
2	1	0	918	1	16	484	800.0	504
3	0	0	88	1	62	1587	300.0	504
4	0	0	166	4	20	1406	600.0	504

```
y = df['rate']
y
```

```
0       4.1
1       4.1
2       3.8
```

```
3        3.7
4        3.8
        ...
23243    3.8
23244    3.9
23245    2.8
23246    2.5
23247    4.3
Name: rate, Length: 23248, dtype: float64
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.3,random_state=10)
```

Building Models

Linear Regression

```
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)
```

```
0.22818828522967072
```

Random Forest

```
from sklearn.ensemble import RandomForestRegressor
RF_Model=RandomForestRegressor(n_estimators=650,random_state=245,min_samples_leaf=.0001)
RF_Model.fit(x_train,y_train)
y_predict=RF_Model.predict(x_test)
r2_score(y_test,y_predict)
```

```
0.8809706960047533
```

ExtraTree Regressor

```
from sklearn.ensemble import  ExtraTreesRegressor
ET_Model=ExtraTreesRegressor(n_estimators = 120)
ET_Model.fit(x_train,y_train)
y_predict=ET_Model.predict(x_test)
r2_score(y_test,y_predict)
```

```
0.9325151755043354
```

r2_score of Extra Tree Regressor is highest which gives us the best model

Save the model

```
import pickle
# Saving model to disk
pickle.dump(ET_Model, open('model.pkl','wb'))
model=pickle.load(open('model.pkl','rb'))
```