

Breakdown of this notebook:

1. **Loading the dataset:** Load the data and import the libraries.

2. **Data Cleaning:**

- Deleting redundant columns.
- Renaming the columns.
- Dropping duplicates.
- Cleaning individual columns.
- Remove the NaN values from the dataset
- #Some Transformations

3. **Regression Analysis**

- Linear Regression
- Decision Tree Regression
- Random Forest Regression

4. **Data Visualization:** Using plots to find relations between the features.

- Restaurants delivering Online or not
- Restaurants allowing table booking or not
- Table booking Rate vs Rate
- Best Location
- Relation between Location and Rating
- Restaurant Type
- Gaussian Rest type and Rating
- Types of Services
- Relation between Type and Rating
- Cost of Restuarant
- No. of restaurants in a Location
- Restaurant type
- Most famous restaurant chains in Bengaluru

The basic idea is analyzing the **Buisness Problem of Zomato** to get a fair idea about the factors affecting the establishment of different types of restaurant at different places in Bengaluru, aggregate rating of each restaurant and many more. I have provided the link to download the dataset at the end of this notebook.

```
In [1]: #Importing Libraries
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import r2_score
```

```
In [2]: #reading the dataset
zomato_real=pd.read_csv("zomato.csv")
zomato_real.head() # prints the first 5 rows of a DataFrame
```

Out[2]:

		url	address	name	online_order	book_table	rating
0		https://www.zomato.com/bangalore/jalsa-banashankari...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1.
1		https://www.zomato.com/bangalore/spice-elephant...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1.
2		https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No	3.8.
3		https://www.zomato.com/bangalore/addhuri-udipi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No	3.7.
4		https://www.zomato.com/bangalore/grand-village...	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	No	3.8.

In [3]: `zomato_real.info() # Looking at the information about the dataset, datatypes of t`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51717 entries, 0 to 51716
Data columns (total 17 columns):
url                      51717 non-null object
address                  51717 non-null object
name                     51717 non-null object
online_order              51717 non-null object
book_table                51717 non-null object
rate                      43942 non-null object
votes                     51717 non-null int64
phone                     50509 non-null object
location                  51696 non-null object
rest_type                 51490 non-null object
dish_liked                23639 non-null object
cuisines                  51672 non-null object
approx_cost(for two people) 51371 non-null object
reviews_list               51717 non-null object
menu_item                 51717 non-null object
listed_in(type)            51717 non-null object
listed_in(city)             51717 non-null object
dtypes: int64(1), object(16)
memory usage: 6.7+ MB
```

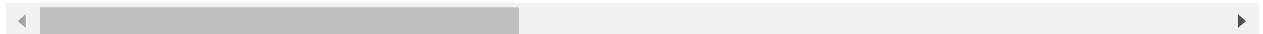
In [4]: `#Deleting Unnecessary Columns`

```
zomato=zomato_real.drop(['url','dish_liked','phone'],axis=1) #Dropping the column
```

In [5]: `zomato_real.head() # Looking at the dataset after transformation`

Out[5]:

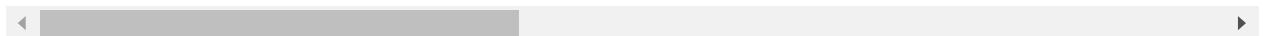
		url	address	name	online_order	book_table	rat
0	https://www.zomato.com/bangalore/jalsa-banashankari	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1.	
1	https://www.zomato.com/bangalore/spice-elephant-bazaar	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1.	
2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No	3.8.	
3	https://www.zomato.com/bangalore/addhuri-udipi	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No	3.7.	
4	https://www.zomato.com/bangalore/grand-village	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	No	3.8.	



```
In [6]: #Removing the Duplicates
zomato.duplicated().sum()
zomato.drop_duplicates(inplace=True)
zomato_real.head() # Looking at the dataset after transformation
```

Out[6]:

		url	address	name	online_order	book_table	rating
0	https://www.zomato.com/bangalore/jalsabana... ...		942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1.
1	https://www.zomato.com/bangalore/spice-elephan... ...		2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1.
2	https://www.zomato.com/SanchurroBangalore? cont...		1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No	3.8.
3	https://www.zomato.com/bangalore/addhuri-udipi... ...		1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No	3.7.
4	https://www.zomato.com/bangalore/grand-village... ...		10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	No	3.8.



In [7]: #Remove the NaN values from the dataset
`zomato.isnull().sum()
zomato.dropna(how='any', inplace=True)
zomato.info() #.info() function is used to get a concise summary of the dataframe`

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 43499 entries, 0 to 51716
Data columns (total 14 columns):
address                  43499 non-null object
name                     43499 non-null object
online_order              43499 non-null object
book_table                43499 non-null object
rate                      43499 non-null object
votes                     43499 non-null int64
location                 43499 non-null object
rest_type                 43499 non-null object
cuisines                  43499 non-null object
approx_cost(for two people) 43499 non-null object
reviews_list               43499 non-null object
menu_item                 43499 non-null object
listed_in(type)            43499 non-null object
listed_in(city)             43499 non-null object
dtypes: int64(1), object(13)
memory usage: 5.0+ MB
```

In [8]: #Reading Column Names
`zomato.columns`

Out[8]: Index(['address', 'name', 'online_order', 'book_table', 'rate', 'votes',
'location', 'rest_type', 'cuisines', 'approx_cost(for two people)',
'reviews_list', 'menu_item', 'listed_in(type)', 'listed_in(city)'),
dtype='object')

In [9]: #Changing the column names
`zomato = zomato.rename(columns={'approx_cost(for two people)':'cost', 'listed_in(type)':'listed_in(city)'})
zomato.columns`

Out[9]: Index(['address', 'name', 'online_order', 'book_table', 'rate', 'votes',
'location', 'rest_type', 'cuisines', 'cost', 'reviews_list',
'menu_item', 'type', 'city'],
dtype='object')

In [10]: #Some Transformations

```
zomato['cost'] = zomato['cost'].astype(str) #Changing the cost to string
zomato['cost'] = zomato['cost'].apply(lambda x: x.replace(',', '.')) #Using Lambda
zomato['cost'] = zomato['cost'].astype(float) # Changing the cost to Float
zomato.info() # Looking at the dataset information after transformation
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 43499 entries, 0 to 51716
Data columns (total 14 columns):
address          43499 non-null object
name             43499 non-null object
online_order     43499 non-null object
book_table       43499 non-null object
rate              43499 non-null object
votes             43499 non-null int64
location          43499 non-null object
rest_type         43499 non-null object
cuisines          43499 non-null object
cost              43499 non-null float64
reviews_list      43499 non-null object
menu_item         43499 non-null object
type              43499 non-null object
city              43499 non-null object
dtypes: float64(1), int64(1), object(12)
memory usage: 5.0+ MB
```

In [11]: #Reading unique values from the Rate column

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

Out[11]: array(['4.1/5', '3.8/5', '3.7/5', '3.6/5', '4.6/5', '4.0/5', '4.2/5',
 '3.9/5', '3.1/5', '3.0/5', '3.2/5', '3.3/5', '2.8/5', '4.4/5',
 '4.3/5', 'NEW', '2.9/5', '3.5/5', '2.6/5', '3.8 /5', '3.4/5',
 '4.5/5', '2.5/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5',
 '3.4 /5', '-', '3.6 /5', '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5',
 '4.1 /5', '3.7 /5', '3.1 /5', '2.9 /5', '3.3 /5', '2.8 /5',
 '3.5 /5', '2.7 /5', '2.5 /5', '3.2 /5', '2.6 /5', '4.5 /5',
 '4.3 /5', '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '4.6 /5',
 '4.9 /5', '3.0 /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)

In [12]: #Removing '/5' from Rates

```
zomato = zomato.loc[zomato.rate != 'NEW']
zomato = zomato.loc[zomato.rate != '-'].reset_index(drop=True)
remove_slash = lambda x: x.replace('/5', '') if type(x) == np.str else x
zomato.rate = zomato.rate.apply(remove_slash).str.strip().astype('float')
zomato['rate'].head() # Looking at the dataset after transformation
```

Out[12]: 0 4.1

1 4.1

2 3.8

3 3.7

4 3.8

Name: rate, dtype: float64

In [13]: # Adjust the column names

```
zomato.name = zomato.name.apply(lambda x:x.title())
zomato.online_order.replace(('Yes','No'),(True, False),inplace=True)
zomato.book_table.replace(('Yes','No'),(True, False),inplace=True)
zomato_real.head() # Looking at the dataset after transformation
```

Out[13]:

		url	address	name	online_order	book_table	rat
0	https://www.zomato.com/bangalore/jalsa-banashankari...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1.	
1	https://www.zomato.com/bangalore/spice-elephant...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1.	
2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No	3.8.	
3	https://www.zomato.com/bangalore/addhuri-udipi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No	3.7.	
4	https://www.zomato.com/bangalore/grand-village...	10, 3rd Floor, Lakshmi Associates, Gandhi Baza...	Grand Village	No	No	3.8.	

In [15]: zomato.cost.unique() # cheking the unique costs

```
Out[15]: array([800. , 300. , 600. , 700. , 550. , 500. , 450. , 650. ,
   400. , 900. , 200. , 750. , 150. , 850. , 100. , 1.2 ,
   350. , 250. , 950. , 1. , 1.5 , 1.3 , 199. , 1.1 ,
   1.6 , 230. , 130. , 1.7 , 1.35, 2.2 , 1.4 , 2. ,
   1.8 , 1.9 , 180. , 330. , 2.5 , 2.1 , 3. , 2.8 ,
   3.4 , 50. , 40. , 1.25, 3.5 , 4. , 2.4 , 2.6 ,
   1.45, 70. , 3.2 , 240. , 6. , 1.05, 2.3 , 4.1 ,
   120. , 5. , 3.7 , 1.65, 2.7 , 4.5 , 80. ])
```

In [16]: #Encode the input Variables

```
def Encode(zomato):
    for column in zomato.columns[~zomato.columns.isin(['rate', 'cost', 'votes'])]:
        zomato[column] = zomato[column].factorize()[0]
    return zomato

zomato_en = Encode(zomato.copy())
zomato_en.head() # Looking at the dataset after transformation
```

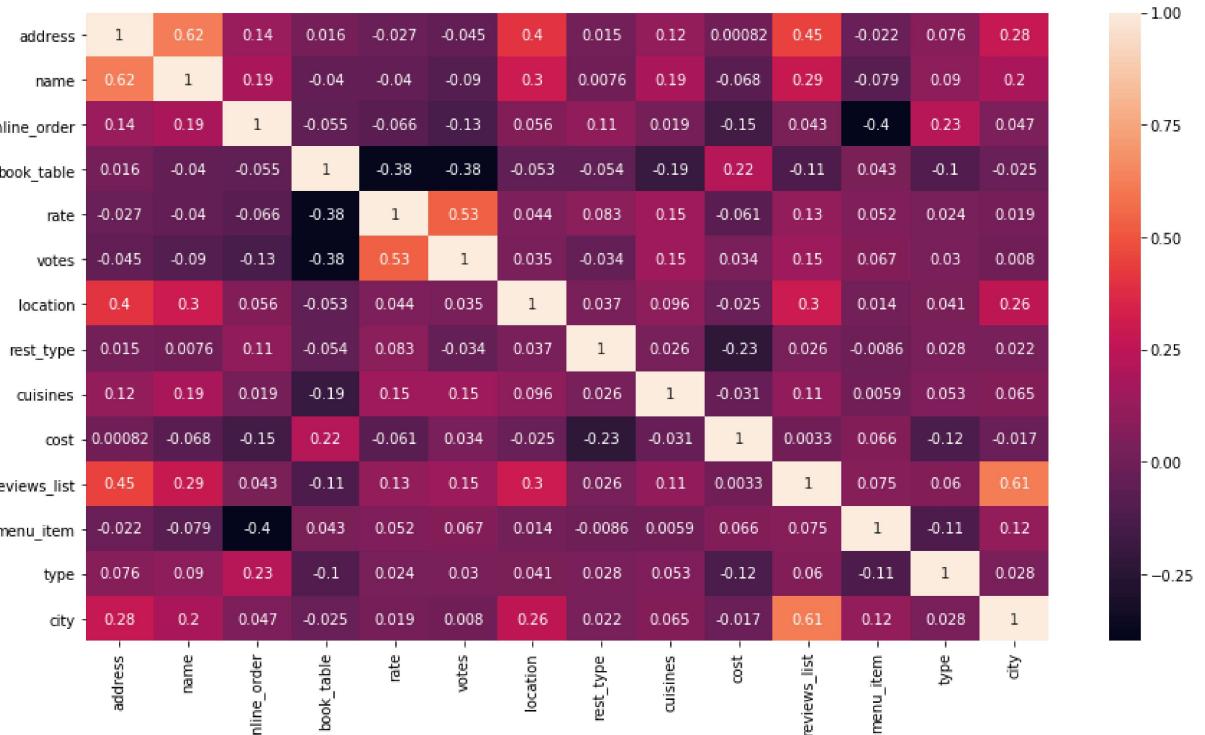
Out[16]:

	address	name	online_order	book_table	rate	votes	location	rest_type	cuisines	cost	revi
0	0	0		0	0	4.1	775	0	0	0	800.0
1	1	1		0	1	4.1	787	0	0	1	800.0
2	2	2		0	1	3.8	918	0	1	2	800.0
3	3	3		1	1	3.7	88	0	2	3	300.0
4	4	4		1	1	3.8	166	1	0	4	600.0

In [17]: #Get Correlation between different variables

```
corr = zomato_en.corr(method='kendall')
plt.figure(figsize=(15,8))
sns.heatmap(corr, annot=True)
zomato_en.columns
```

Out[17]: Index(['address', 'name', 'online_order', 'book_table', 'rate', 'votes', 'location', 'rest_type', 'cuisines', 'cost', 'reviews_list', 'menu_item', 'type', 'city'], dtype='object')



The highest correlation is between name and address which is 0.62 which is not of very much concern

Regression Analysis

Splitting the Dataset

```
In [18]: #Defining the independent variables and dependent variables
x = zomato_en.iloc[:,[2,3,5,6,7,8,9,11]]
y = zomato_en['rate']
#Getting Test and Training Set
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.1,random_state=353)
x_train.head()
```

Out[18]:

	online_order	book_table	votes	location	rest_type	cuisines	cost	menu_item
16950	0	1	0	8	2	5	250.0	0
767	0	1	131	8	4	278	400.0	190
6750	0	1	137	45	2	1295	250.0	0
9471	0	1	74	16	0	537	1.0	0
25162	0	1	61	12	2	1860	350.0	0

In [19]: y_train.head()

Out[19]:

16950	3.9
767	3.7
6750	4.0
9471	3.8
25162	3.7

Name: rate, dtype: float64

In [20]: zomato_en['menu_item'].unique() # seeing the unique values in 'menu_item'

Out[20]: array([0, 1, 2, ..., 8240, 8241, 8242], dtype=int64)

In [21]: zomato_en['location'].unique() # seeing the unique values in 'Location'

Out[21]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91], dtype=int64)

In [22]: zomato_en['cuisines'].unique() # seeing the unique values in 'cuisines'

Out[22]: array([0, 1, 2, ..., 2364, 2365, 2366], dtype=int64)

In [23]: `zomato_en['rest_type'].unique() # seeing the unique values in 'rest_type'`

Out[23]: `array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86], dtype=int64)`

In [24]: `x.head()`

Out[24]:

	online_order	book_table	votes	location	rest_type	cuisines	cost	menu_item
0	0	0	775	0	0	0	800.0	0
1	0	1	787	0	0	1	800.0	0
2	0	1	918	0	1	2	800.0	0
3	1	1	88	0	2	3	300.0	0
4	1	1	166	1	0	4	600.0	0

In [25]: `y.head()`

Out[25]:

0	4.1
1	4.1
2	3.8
3	3.7
4	3.8

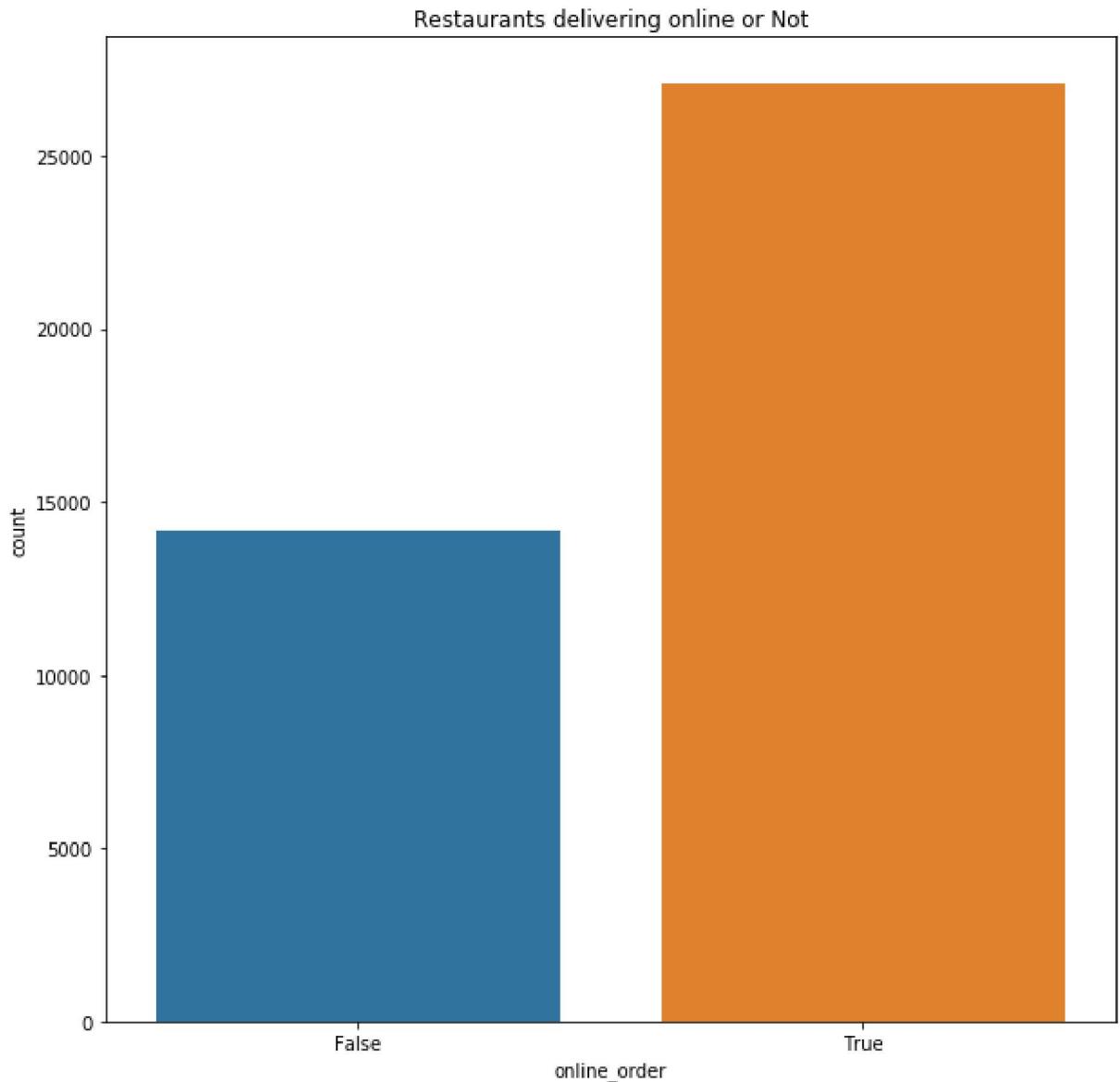
Name: rate, dtype: float64

Data Visualization

Restaurants delivering Online or not

```
In [33]: #Restaurants delivering Online or not  
sns.countplot(zomato['online_order'])  
fig = plt.gcf()  
fig.set_size_inches(10,10)  
plt.title('Restaurants delivering online or Not')
```

```
Out[33]: Text(0.5, 1.0, 'Restaurants delivering online or Not')
```



Restaurants allowing table booking or not

```
In [34]: sns.countplot(zomato['book_table'])
fig = plt.gcf()
fig.set_size_inches(10,10)
plt.title('Restaurants allowing table booking or not')
```

```
Out[34]: Text(0.5, 1.0, 'Restaurants allowing table booking or not')
```

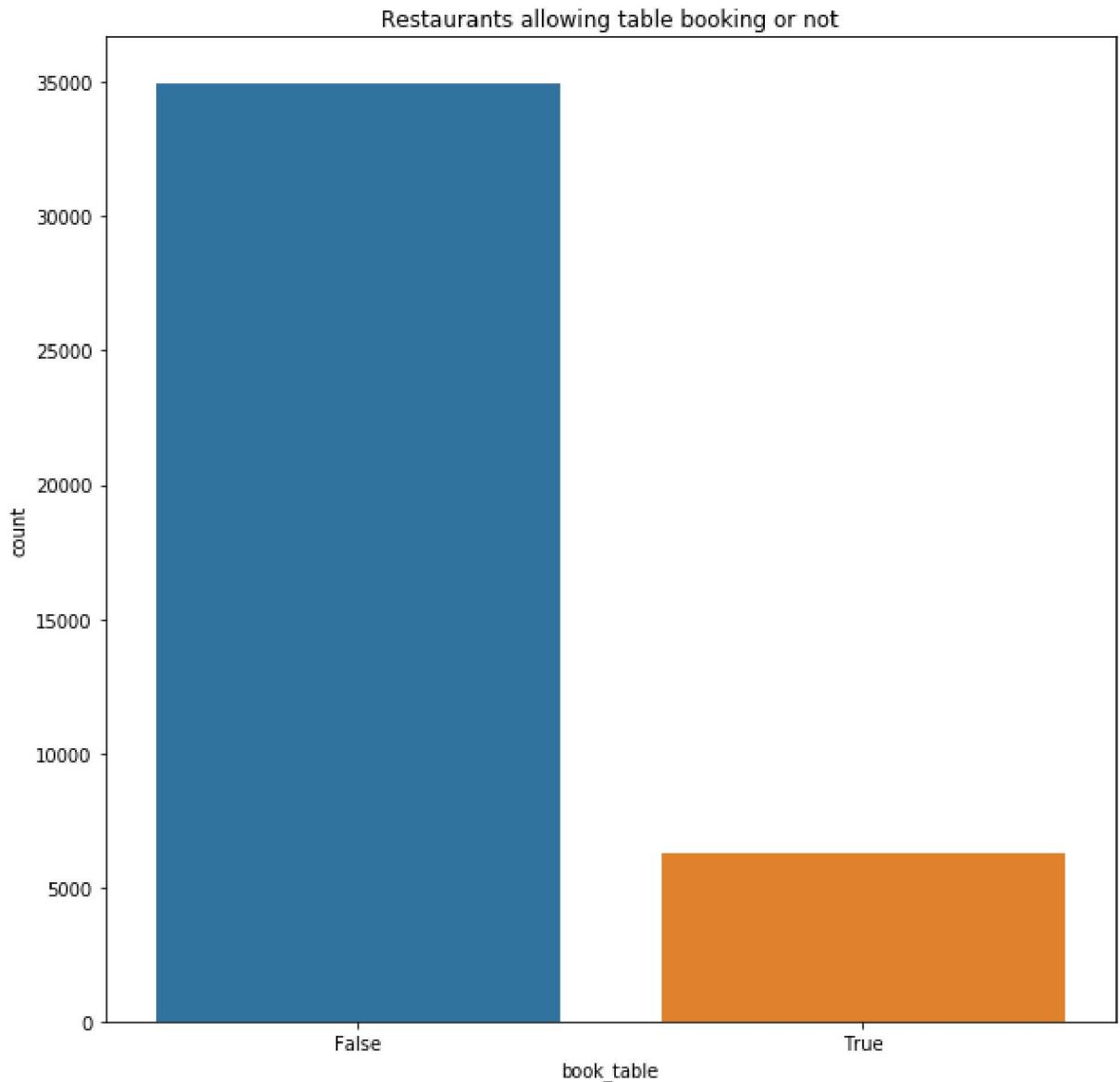


Table booking Rate vs Normal Rate

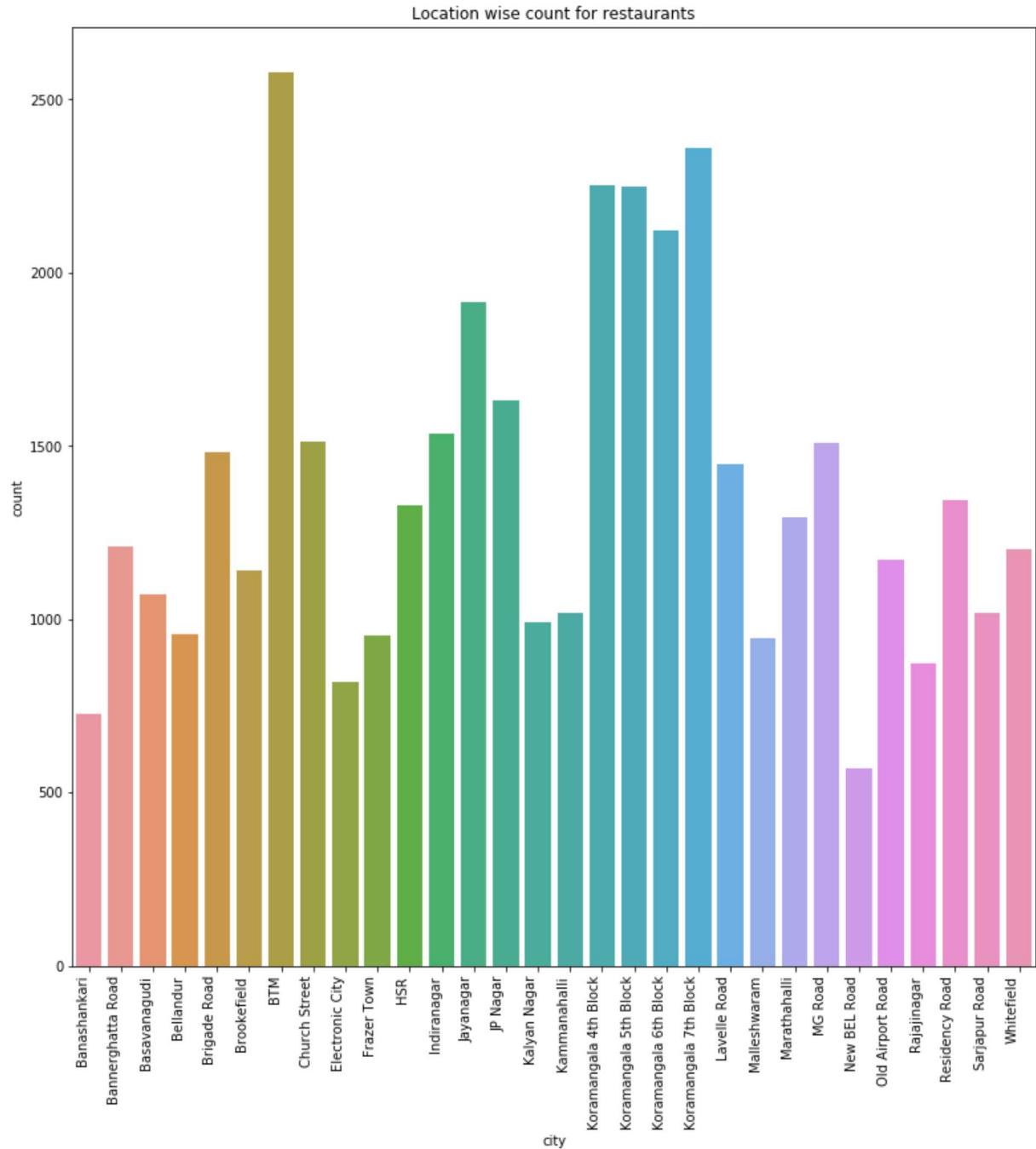
```
In [36]: plt.rcParams['figure.figsize'] = (13, 9)
Y = pd.crosstab(zomato['rate'], zomato['book_table'])
Y.div(Y.sum(1).astype(float), axis = 0).plot(kind = 'bar', stacked = True,color=[plt.cm.Reds(0),plt.cm.YlGnBu(0)])
plt.title('table booking vs Normal rate', fontweight = 30, fontsize = 20)
plt.legend(loc="upper right")
plt.show()
```



Location

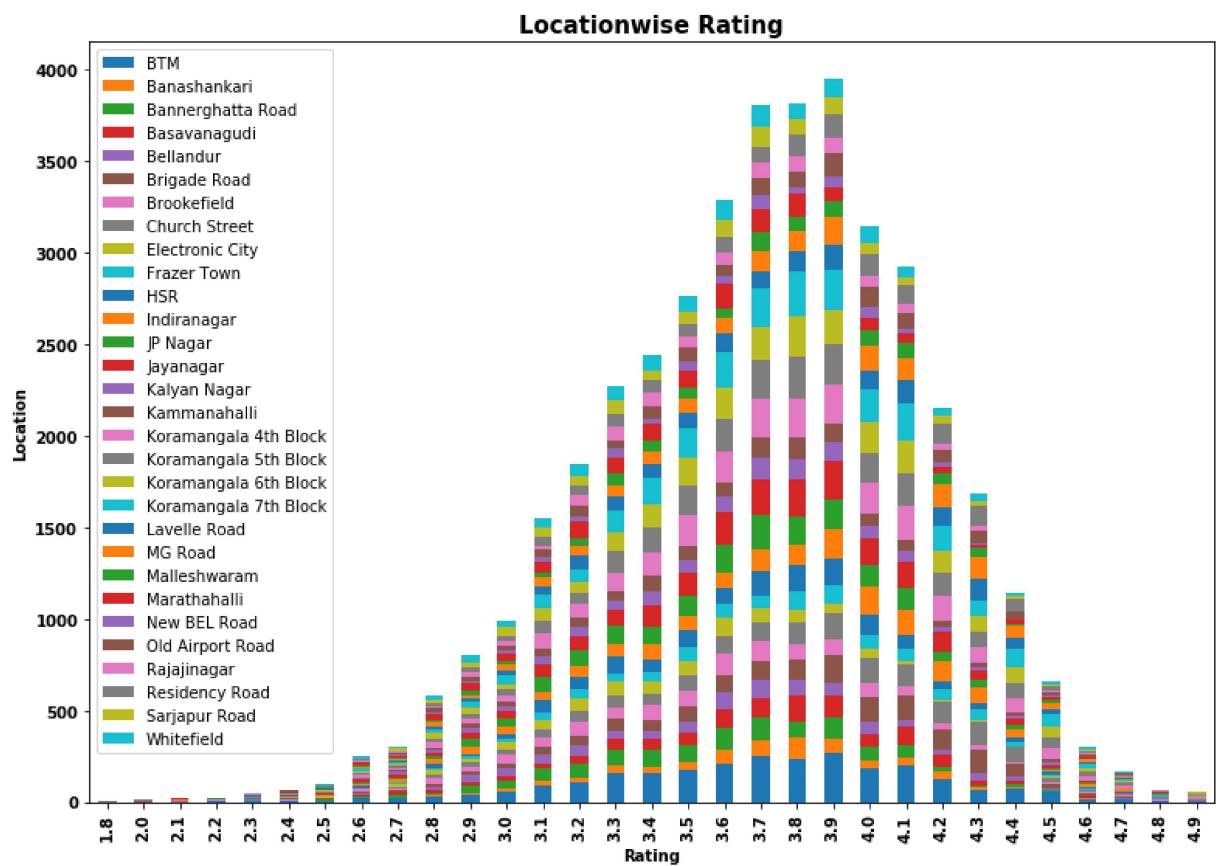
```
In [38]: sns.countplot(zomato['city'])
sns.countplot(zomato['city']).set_xticklabels(sns.countplot(zomato['city']).get_xticklabels(), rotation=90)
fig = plt.gcf()
fig.set_size_inches(13,13)
plt.title('Location wise count for restaurants')
```

Out[38]: Text(0.5, 1.0, 'Location wise count for restaurants')



Location and Rating

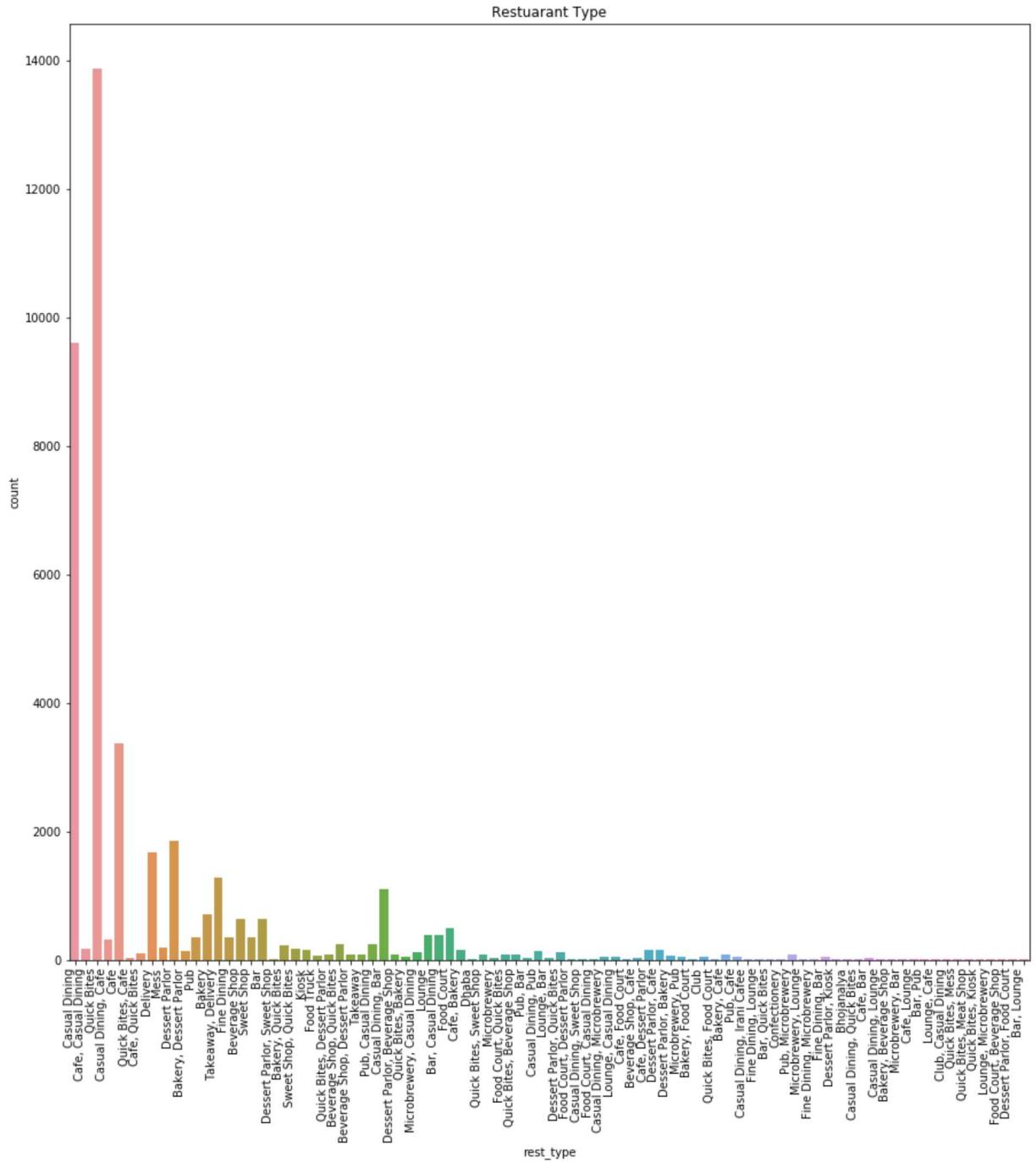
```
In [40]: loc_plt=pd.crosstab(zomato['rate'],zomato['city'])
loc_plt.plot(kind='bar',stacked=True);
plt.title('Locationwise Rating',fontsize=15,fontweight='bold')
plt.ylabel('Location',fontsize=10,fontweight='bold')
plt.xlabel('Rating',fontsize=10,fontweight='bold')
plt.xticks(fontsize=10,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold');
plt.legend();
```



Restaurant Type

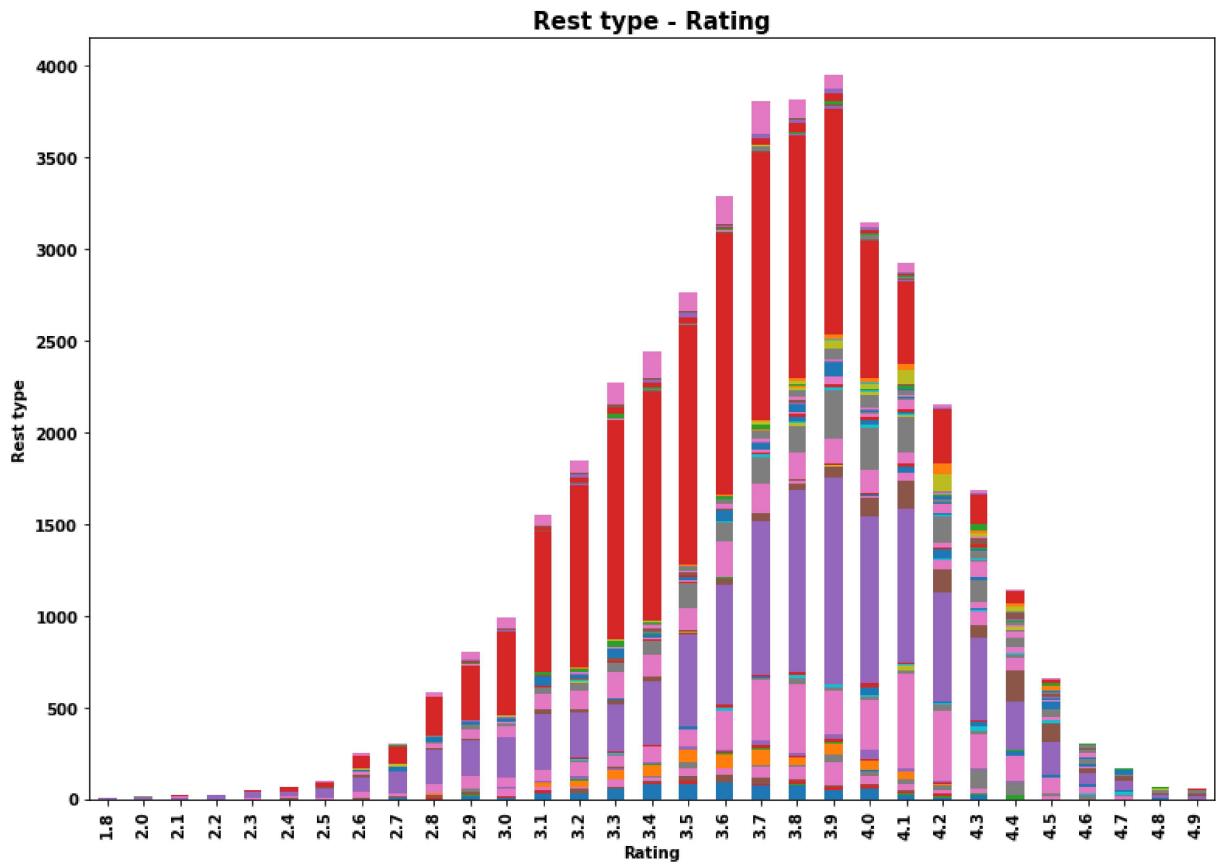
```
In [41]: sns.countplot(zomato['rest_type'])
sns.countplot(zomato['rest_type']).set_xticklabels(sns.countplot(zomato['rest_type']).get_xticklabels(), rotation=90)
fig = plt.gcf()
fig.set_size_inches(15,15)
plt.title('Restuarant Type')
```

```
Out[41]: Text(0.5, 1.0, 'Restuarant Type')
```



Gaussian Rest type and Rating

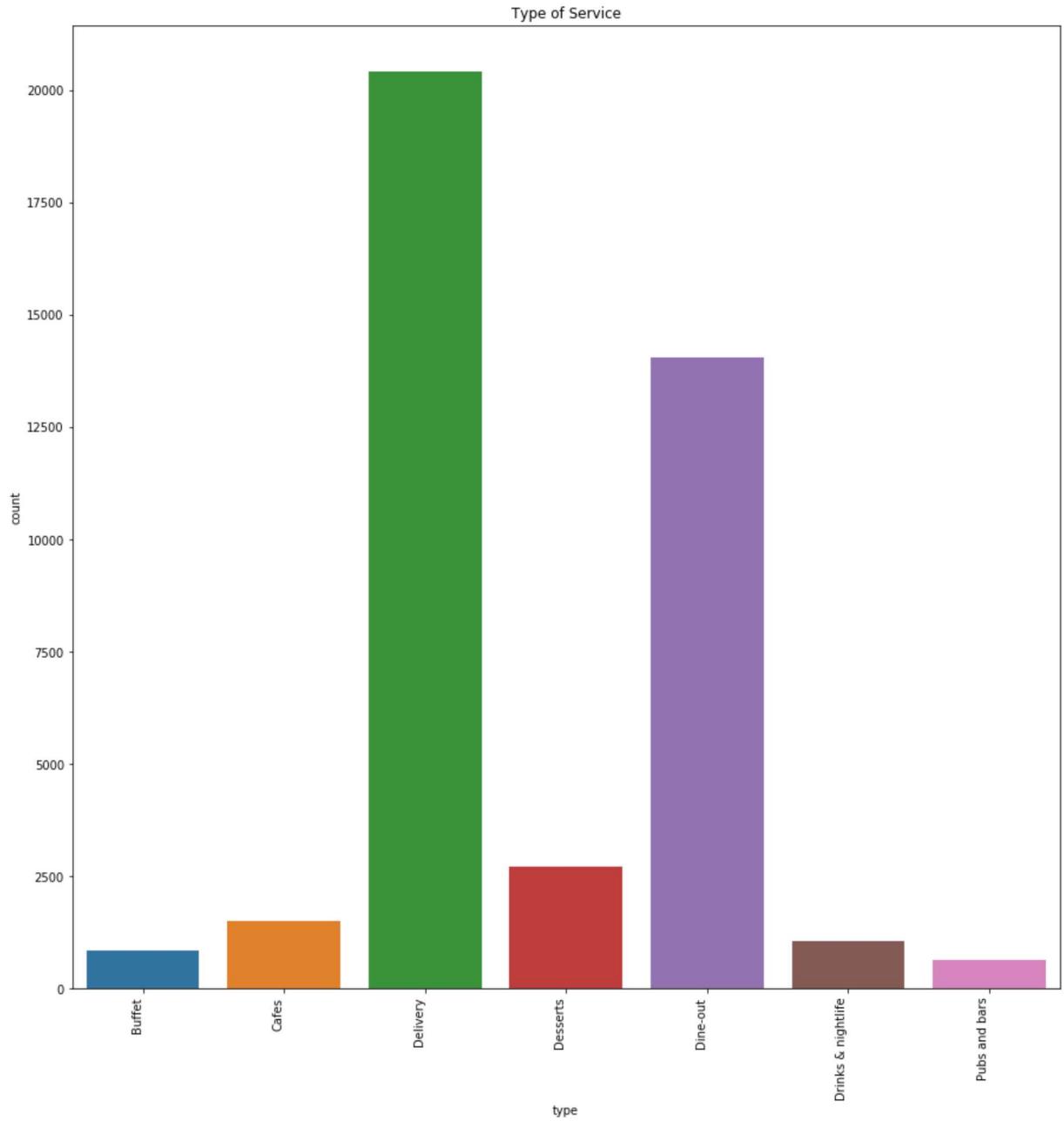
```
In [42]: loc_plt=pd.crosstab(zomato['rate'],zomato['rest_type'])
loc_plt.plot(kind='bar',stacked=True);
plt.title('Rest type - Rating',fontsize=15,fontweight='bold')
plt.ylabel('Rest type',fontsize=10,fontweight='bold')
plt.xlabel('Rating',fontsize=10,fontweight='bold')
plt.xticks(fontsize=10,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold');
plt.legend().remove();
```



Types of Services

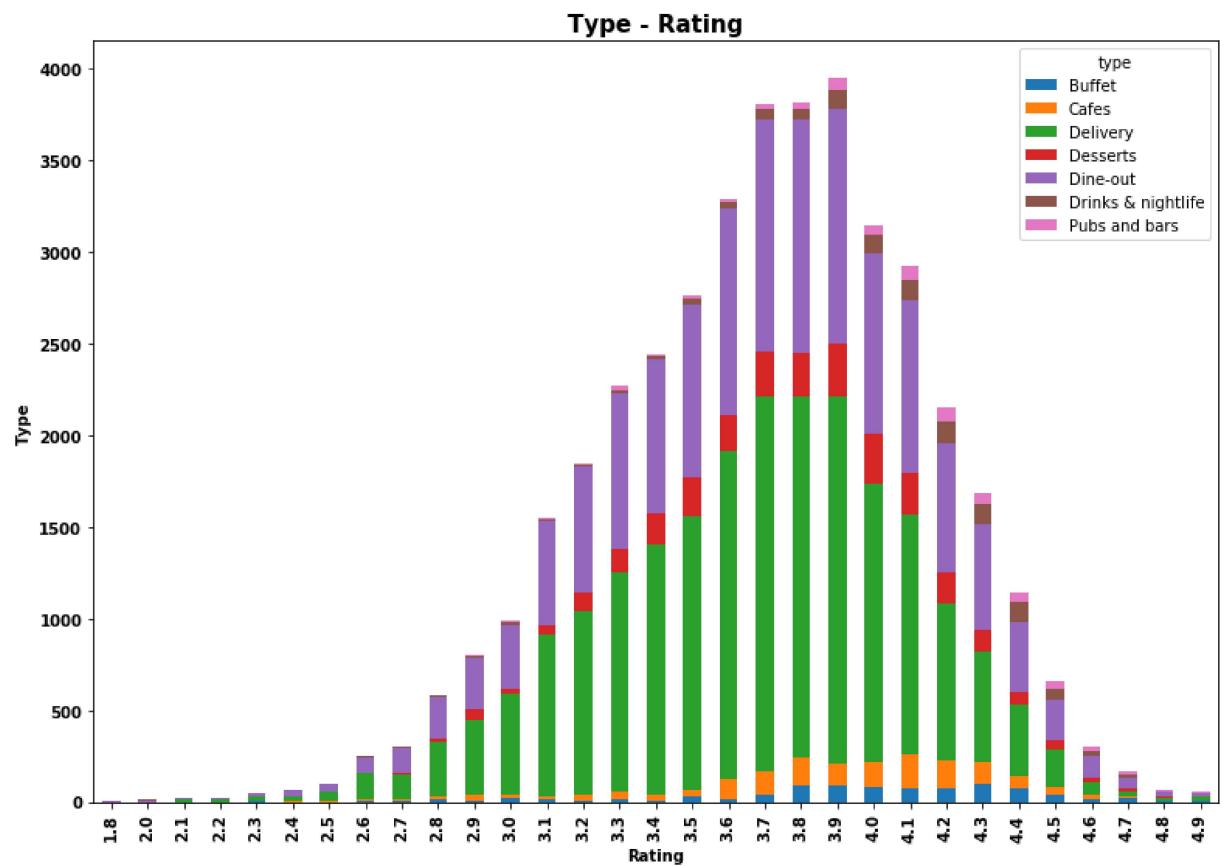
```
In [43]: sns.countplot(zomato['type'])
sns.countplot(zomato['type']).set_xticklabels(sns.countplot(zomato['type']).get_
fig = plt.gcf()
fig.set_size_inches(15,15)
plt.title('Type of Service')
```

Out[43]: Text(0.5, 1.0, 'Type of Service')



Type and Rating

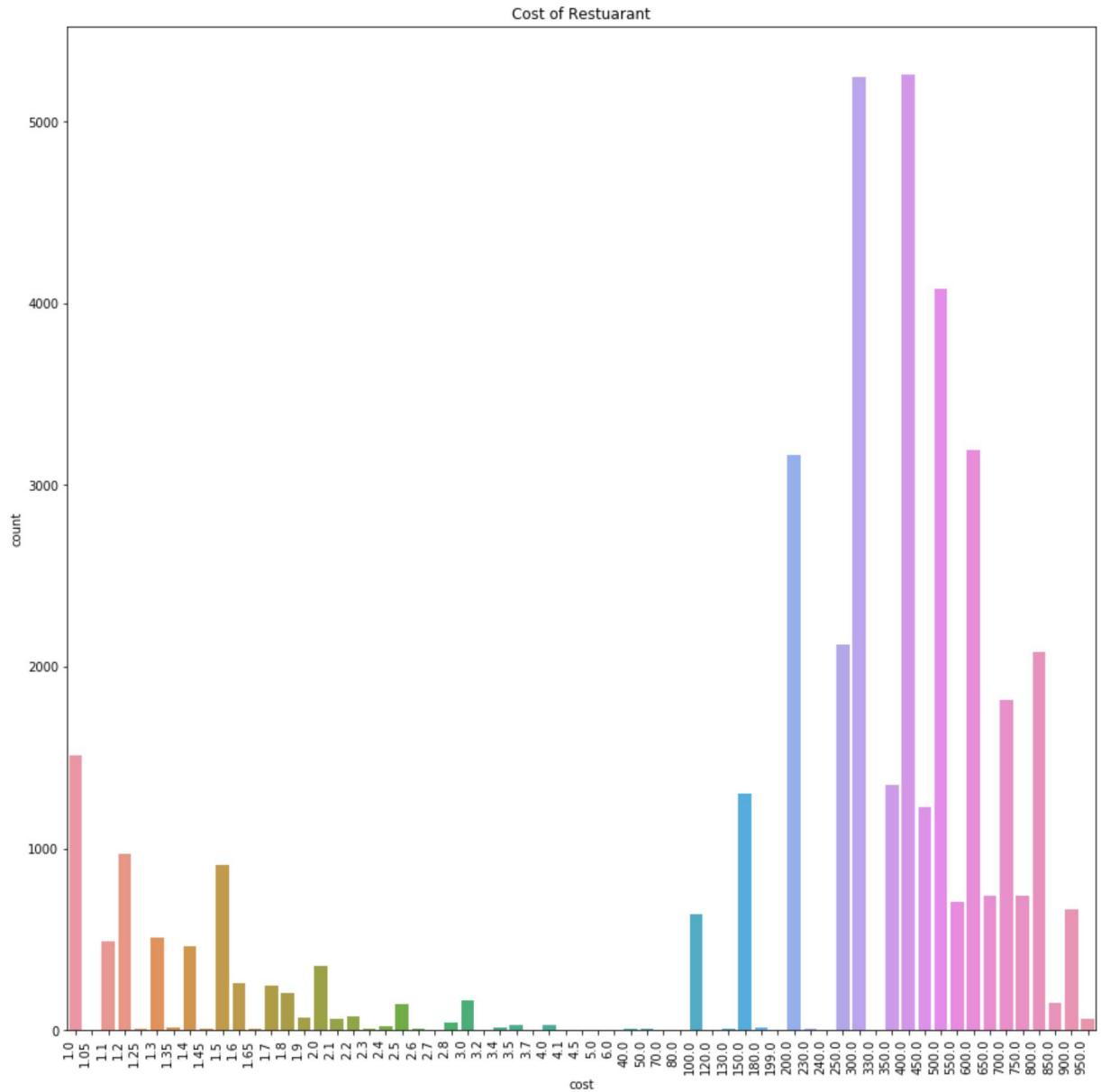
```
In [45]: type_plt=pd.crosstab(zomato['rate'],zomato['type'])
type_plt.plot(kind='bar',stacked=True);
plt.title('Type - Rating',fontsize=15,fontweight='bold')
plt.ylabel('Type',fontsize=10,fontweight='bold')
plt.xlabel('Rating',fontsize=10,fontweight='bold')
plt.xticks(fontsize=10,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold');
```



Cost of Restaurant

```
In [27]: sns.countplot(zomato['cost'])
sns.countplot(zomato['cost']).set_xticklabels(sns.countplot(zomato['cost']).get_
fig = plt.gcf()
fig.set_size_inches(15,15)
plt.title('Cost of Restuarant')
```

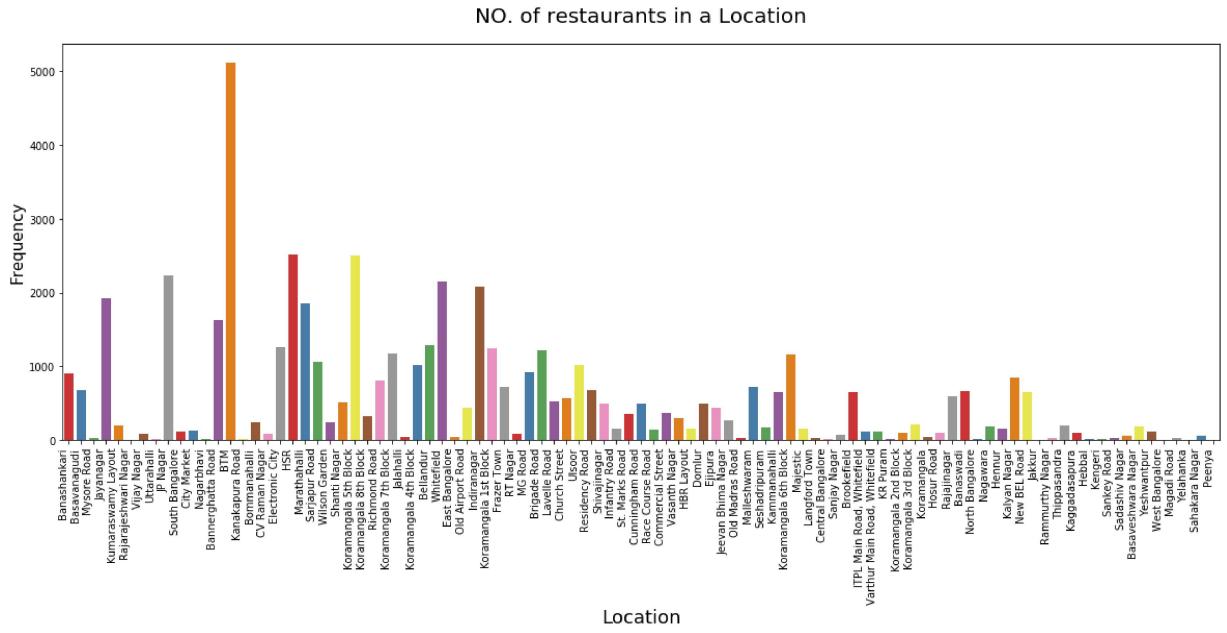
Out[27]: Text(0.5, 1.0, 'Cost of Restuarant')



No. of Restaurants in a Location

```
In [46]: fig = plt.figure(figsize=(20,7))
loc = sns.countplot(x="location", data=zomato_real, palette = "Set1")
loc.set_xticklabels(loc.get_xticklabels(), rotation=90, ha="right")
plt.ylabel("Frequency", size=15)
plt.xlabel("Location", size=18)
loc
plt.title('NO. of restaurants in a Location', size = 20, pad=20)
```

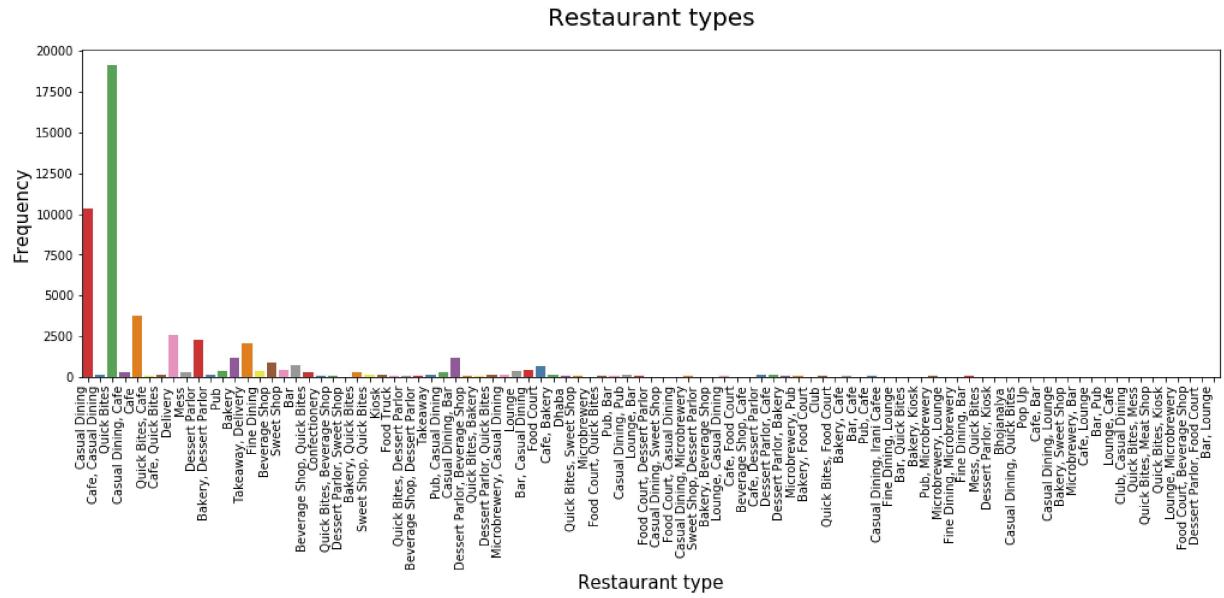
Out[46]: Text(0.5, 1.0, 'NO. of restaurants in a Location')



Restaurant type

```
In [47]: fig = plt.figure(figsize=(17,5))
rest = sns.countplot(x="rest_type", data=zomato_real, palette = "Set1")
rest.set_xticklabels(rest.get_xticklabels(), rotation=90, ha="right")
plt.ylabel("Frequency", size=15)
plt.xlabel("Restaurant type", size=15)
rest
plt.title('Restaurant types', fontsize = 20, pad=20)
```

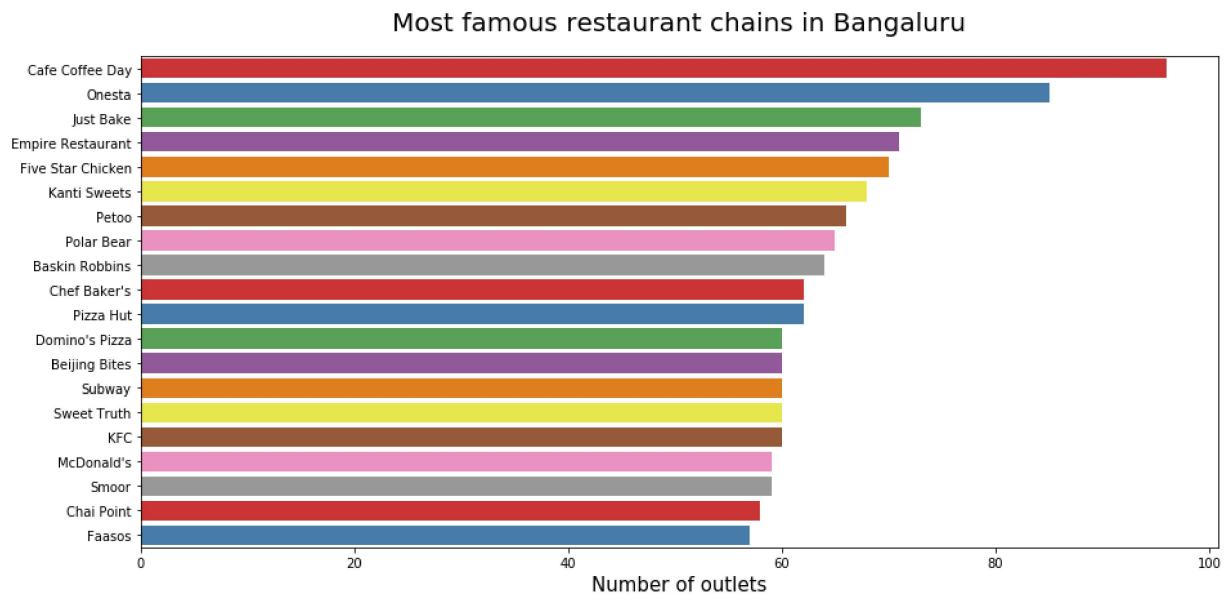
Out[47]: Text(0.5, 1.0, 'Restaurant types')



Most famous Restaurant chains in Bengaluru

```
In [48]: plt.figure(figsize=(15,7))
chains=zomato_real['name'].value_counts()[:20]
sns.barplot(x=chains,y=chains.index,palette='Set1')
plt.title("Most famous restaurant chains in Bangalore",size=20,pad=20)
plt.xlabel("Number of outlets",size=15)
```

Out[48]: Text(0.5, 0, 'Number of outlets')



Linear Regression

```
In [49]: #Prepare a Linear Regression Model
reg=LinearRegression()
reg.fit(x_train,y_train)
y_pred=reg.predict(x_test)
from sklearn.metrics import r2_score
r2_score(y_test,y_pred)
```

Out[49]: 0.27351655171030775

Decision Tree Regression

```
In [50]: #Prepairng a Decision Tree Regression
from sklearn.tree import DecisionTreeRegressor
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.1,random_state=105)
DTree=DecisionTreeRegressor(min_samples_leaf=.0001)
DTree.fit(x_train,y_train)
y_predict=DTree.predict(x_test)
from sklearn.metrics import r2_score
r2_score(y_test,y_predict)
```

Out[50]: 0.8534852172188889

Random Forest Regression

In [51]: #Preparing Random Forest REgression

```
from sklearn.ensemble import RandomForestRegressor
RForest=RandomForestRegressor(n_estimators=500,random_state=329,min_samples_leaf=1)
RForest.fit(x_train,y_train)
y_predict=RForest.predict(x_test)
from sklearn.metrics import r2_score
r2_score(y_test,y_predict)
```

Out[51]: 0.8773808619238765

Extra Tree Regressor

In [52]: #Preparing Extra Tree Regression

```
from sklearn.ensemble import ExtraTreesRegressor
ETree=ExtraTreesRegressor(n_estimators = 100)
ETree.fit(x_train,y_train)
y_predict=ETree.predict(x_test)

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

Out[52]: 0.9399168889035364

In []: import pickle

```
# Saving model to disk
pickle.dump(ETree, open('model.pkl','wb'))
```

It can be observed that we have got the best accuracy for Extra tree regressor

In []: