Analysis of Zomato Restaurants Data

BootCamp on Data Science and Tools

Submitted

To

CRC-Training

By

Abhay Kumar Pandey
Roll Number: 2002901520004

Naman Tiwari
Roll Number: 2002901550019

Priyank Awasthi
Roll Number: 2002901520082

Under the guidance of
Mr. Gaurav Kansal / Mr. Gopal Gupta

TABLE OF CONTENTS

	Page No.
DECLARATION	ii
CERTIFICATE	iii
ACKNOWLEDGEMENTS	iv
ABSTRACT	v
LIST OF FIGURES	vi
CHAPTER-1 INTRODUCTION	**1**
1.1 Problem Definition	1
1.2 Motivation	1
1.3 Objective of the Project	1
1.4 Scope of the Project	1
1.5 Need of Work	2
CHAPTER 2 RELATED WORK	**3**
CHAPTER 3 PROPOSED METHODOLOGY	**4**
3.1 Dataset Description	4
3.2 Methods	5
3.3 Hardware / Software Requirements	9
3.3 Our Methodology	10
CHAPTER 4 CONCLUSION	**54**
5.1 Discussion	54
5.2 Future Work	54
REFERENCES	**55**

CERTIFICATE

This is to certify that Project Report entitled **"Analysis of Zomato Restaurant Data"** which is submitted by **Abhay Kumar Pandey, Naman Tiwari** and **Priyank Awasthi** in partial fulfillment of the requirement for the "Bootcamp on Data Science and Tools" in Department of CRC-Training of ABES Institute of Technology, is a record of the candidate own work carried out by him under my/our supervision.

Mr. Gaurav Kansal

Mr. Gopal Gupta

Date: 28-04-2023

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the "Bootcamp on Data Science and Tools" *undertaken during B.Tech, 3rd Year. We owe special debt of gratitude to Mr. Gopal Gupta and Mr. Gaurav Kansal for his constant support and guidance throughout the course of our work. His constant motivation has been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.*

We would like to thank head of department of computer science and engineering (Artificial intelligence/IoT) to provide us the opportunity and help us throughout the project.

We also take the opportunity to acknowledge the contribution of team members of CRC-Training for their full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the motivation of Department Of Computer Science And Engineering(AI/IoT), ABES Institute of Technology to provide us the opportunity to undergo training at CRC-Training.

Signature:

Name : Abhay Kumar Pandey

Roll No.:2002901520004

Date : 28-04-2023

Signature:

Name : Naman Tiwari

Roll No.: 2002901550019

Date : 28-04-2023

Signature:

Name : Priyank Awasthi

Roll No.: 2002901520082

Date : 28-04-2023

ABSTRACT

The aim of this project is to analyse the Zomato restaurants data and visualize the insights obtained from it. The report focuses on the analysis of various factors that affect the success of a restaurant, such as location, cuisines, ratings, and cost. The data is pre-processed before analysis.

Several visualizations are used to showcase the findings of the analysis. These include bar charts, scatter plots, heat maps, and maps. The report highlights the popular cuisines in different regions, the correlation between ratings and cost, the number of restaurants in a particular location, and the most preferred type of restaurant based on user ratings.

This analysis offers a comprehensive understanding of the factors that affect the success of a restaurant. Overall, the project provides valuable insights for restaurant owners, investors, and food enthusiasts looking to explore the restaurant scene in different cities.

LIST OF FIGURES

Fig. No.	Fig. Name	Page No.
3.1	Installing NumPy	**5**
3.2	Installing Seaborn	**7**
3.3	Importing Libraries	**9**
3.4	read_excel()	**10**
3.5	describe() function	**10**
3.6	info() function	**11**
3.7	head() function	**12**
3.8	All columns	**12**
3.9	duplicated().sum()	**12**
3.10	isnull().sum() function	**13**
3.11	Rows with “NaN”	**14**
3.12	Removing rows with "NaN"	**15**
3.13	isnull().sum() function	**15**
3.14	No Of Restaurants in country	**16**
3.15	Restaurants In Different Country	**17**
3.16	Indian Restaurants	**18**
3.17	No Of Restaurants in city (bar-plot)	**19**
3.18	No Of Restaurants in city(Table)	**20**
3.19	No of Locality	**21**
3.20	Online order service (bar-plot)	**21**
3.21	Online order service (pie-plot)	**22**
3.22	Online order in Locality	**23**
3.23	Table Booking Service(bar-plot)	**24**
3.24	Table Booking Service(pie-plot)	**25**
3.25	Table booking facility location wise	**27**
3.26	Average rating on Locality	**29**
3.27	Best Restaurant	**29**
3.28	Votes on Locality	**31**
3.29	No of cuisines	**33**
3.30	Top rated cuisines	**35**
3.31	Most served cuisines	**37**
3.32	Locality by price range	**39**
3.33	Online order Vs Rate	**40**
3.34	Table booking Vs Rate	**41**
3.35	Type of Rating text	**41**
3.36	Text Rating on Locality	**44**
3.37	Popular Restaurants	**45**
3.38	Restaurants with highest vote	**47**
3.39	Relation b/w rating and cost	**49**
3.40	Top 10 cuisines with ‘price range’ and ‘agg. rating’	**50**

CHAPTER 1

INTRODUCTION

1.1. Problem Definition: -

- The analysis of Zomato restaurants data aims to examine and understand various aspects of restaurants listed on the Zomato platform, such as the types of cuisines, pricing, ratings, and reviews.
- The goal of this analysis is to gain insights into the restaurant industry and consumer behavior, which can be used to inform business decisions for restaurant owners and Zomato.
- The analysis may involve descriptive statistics, data visualization techniques to uncover patterns and relationships in the data.

1.2. Motivation:

- The motivation for analyzing Zomato restaurants data is to gain insights into various aspects of the restaurant industry, such as consumer preferences, pricing strategies, popular restaurants, geographic trends, and areas for improvement in Zomato's services.
- By analyzing this data, restaurant owners and Zomato can make informed business decisions to better meet the needs of consumers and remain competitive in the market.

1.3. Objective of the Project:

- The objective of a project analyzing Zomato restaurant data is to provide valuable insights into various aspects of the restaurant industry.
- To understand customer behavior and preferences, such as their preferred cuisine type, price range, location, or specific dishes. By analyzing this data, restaurant owners can make more informed decisions about their menus, pricing, and marketing strategies, leading to improved performance, profitability, and customer satisfaction.

1.4. Scope of the Project:

- The project could involve analyzing customer behavior and preferences, such as their preferred cuisine type, price range, location, or specific dishes. It could

also involve competitor analysis, identifying the top competitors of a particular restaurant and analyzing their menus, pricing, customer reviews, and overall performance.

- Market research is another area of focus, where trends in customer behavior and market dynamics can be identified and analyzed. This could include identifying emerging food trends or changes in consumer preferences.

1.5. Need of Work:

- The analysis of Zomato restaurant data has become increasingly important due to the growth of the restaurant industry and the rise of online food ordering platforms.
- The analysis of customer behaviour and preferences can provide insights into the types of cuisine, price ranges, and locations that are most popular, enabling restaurant owners to make informed decisions about their menus, pricing, and marketing strategies. This can lead to improved performance, profitability, and customer satisfaction.

CHAPTER 2

RELATED WORK

Zomato is a platform that provides information and reviews on restaurants across the world. It is a valuable source of data for data science and analytics enthusiasts who want to explore various aspects of the restaurant industry, such as customer preferences, ratings, cuisines, locations, prices, and more. There have been numerous studies and analyses performed on Zomato data, ranging from descriptive statistics to predictive modeling. Some of the common questions that can be answered using Zomato data are:

- What are the most popular cuisines and restaurants in a given city or country?
- How do ratings and reviews affect the popularity and profitability of restaurants?
- What are the factors that influence the price range and delivery time of restaurants?
- How can restaurants improve their service quality and customer satisfaction?
- How can Zomato data be used to recommend restaurants to users based on their preferences and behavior?

These are some of the examples of how Zomato data can be analyzed and utilized for various purposes. Zomato data is a rich and diverse dataset that offers many opportunities for data science and analytics projects.

We have read many articles on websites about our projects. We got much useful information from it about data set and attributes in our data sets. This section will contain reference about two such papers. After reading the papers we got these basic ideas about dataset:

1. "Exploratory Data Analysis and Sentiment Analysis of Zomato Restaurant Reviews" by Kavya Datta and Rajiv Kumar. This study analyzed the reviews and ratings of restaurants on Zomato to understand the sentiments of the customers. The study used data visualization and natural language processing techniques to identify the key factors that influence the customers' opinions.
2. "Predicting restaurant ratings using Machine Learning" by Siddhant Jain and Prateek Jain. This study used machine learning algorithms to predict the ratings of restaurants on Zomato based on their location, cuisine, and other features. The study found that the accuracy of the predictions was around 80%.
3. "An Empirical Study of Online Restaurant Ratings and Reviews" by Xiangliang Zhang and Yun Fu. This study analyzed the data from multiple online platforms, including Zomato, to understand the relationship between the reviews and ratings of restaurants. The study found that the sentiment of the reviews and the rating of the restaurant were strongly correlated.
4. "Understanding Customer Preferences in Restaurant Selection Using Zomato Dataset" by Venkata Sai Manoj and Rama Sushma G. This study analyzed the data on Zomato to understand the preferences of customers in selecting restaurants. The study found that the location and the cuisine of the restaurant were the most important factors in the decision-making process.

CHAPTER 3

PROPOSED METHODOLOGY

3.1 Dataset Description

- Zomato is an Indian multinational restaurant aggregator and food delivery company.
- Zomato provides information, menus and user-reviews of restaurants as well as food delivery options from partner restaurants in select cities.
- This dataset contains information of food restaurants who are working with Zomato.
- The data set has 9551 rows and 22 columns.
- It has a total of 22 attributes. They are as follows: -

1. **Restaurant Id:** Unique id of every restaurant across various cities of the world.
2. **Restaurant Name:** Name of the restaurant.
3. **Country Code:** Country in which the restaurant is located.
4. **City:** City in which restaurant is located.
5. **Address:** Address of the restaurant.
6. **Locality:** Location in the city.
7. **Locality Verbose:** Detailed description of the locality.
8. **Longitude:** Longitude coordinate of the restaurant's location.
9. **Latitude:** Latitude coordinate of the restaurant's location.
10. **Cuisines:** Cuisines offered by the restaurant.
11. **Average Cost for two:** Cost for two people in different currencies.
12. **Currency:** Currency of the country.
13. **Has Table booking:** Is restaurant is giving table booking service or not.
14. **Has Online delivery:** Is restaurant is giving online delivery service or not.
15. **Is delivering:** is delivery service available at the restaurant or not.
16. **Switch to order menu:** yes/no.
17. **Price range:** range of price of food.
18. **Aggregate Rating:** Average rating out of 5.
19. **Rating color:** depending upon the average rating color.
20. **Rating text:** text based on rating of rating.
21. **Votes:** Number of ratings cast by people.
22. **Country:** Name of country.

3.2 Methods:

- After downloading the data set, we begin by analyzing the data set's attributes, then preprocessing it.
- With the help of matplotlib and seaborn we visual each attribute with target variable.

NumPy:

NumPy is a commonly used Python data analysis package. By using NumPy, you can speed up your workflow, and interface with other packages in the Python ecosystem, like scikit-learn, that use NumPy under the hood. NumPy was originally developed in the mid-2000s and arose from an even older package called Numeric. This longevity means that almost every data analysis or machine learning package for Python leverages NumPy in some way.

In this tutorial, we'll walk through using NumPy to analyze data of Zomato. The data contains information on various attributes of restaurants, such as cuisine and location, along with a quality score between 0 and 5 for each restaurant.

Installing NumPy:

```
# pip2 install numpy
Collecting numpy
Using cached numpy-1.12.1-cp27-cp27mu-manylinux1_x86_64.whl
Installing collected packages: numpy
Successfully installed numpy-1.12.1
```

Fig. 3.1: Installing NumPy

It's highly recommended to install Python using Anaconda distribution to make sure all underlying dependencies (such as Linear Algebra libraries) all sync up with the use of a Conda install.

Pandas:

Pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open-source

data analysis / manipulation tool available in any language. It is already well on its way towards this goal.

Main Features: -

- Easy handling of missing data (represented as Nan, NA, or Nat) in floating point as well as non-floating-point data.
- Size mutability: columns can be inserted and deleted from Data Frame and higher dimensional objects.
- Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, Data Frame, etc. automatically align the data for you in computations.
- Powerful, flexible group by functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data.
- Make it easy to convert ragged, differently indexed data in other Python and NumPy data structures into Data Frame objects.
- Intelligent label-based slicing, fancy indexing, and sub setting of large data sets
- Intuitive merging and joining data sets
- Flexible reshaping and pivoting of data sets
- Hierarchical labeling of axes (possible to have multiple labels per tick)
- Robust IO tools for loading data from flat files (CSV and delimited), Excel files, databases, and saving/loading data from the ultrafast HDF5 format
- Time series-specific functionality: date range generation and frequency conversion, moving window statistics, date shifting and lagging.

Matplotlib:

Matplotlib is an open-source drawing library that supports various drawing types .You can generate plots, histograms, bar charts, and other types of charts with just a few lines of code. It's often used in web application servers, shells, and Python scripts.

Getting Started with Pyplot:

Pyplot is a Matplotlib module that provides simple functions for adding plot elements, such as lines, images, text, etc. to the axes in the current figure.

Matplotlib Subplots

You can use the subplot() method to add more than one plot in a figure.

Syntax: plt.subplots(nrows, ncols, index)

The three-integer arguments specify the number of rows and columns and the index of the subplot grid.

Important Types of Plots:

- Bar graphs
- Histograms
- Scatter plots

Seaborn:

Seaborn is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis. Seaborn works easily with data frames and the Pandas library. The graphs created can also be customized easily. Below are a few benefits of Data Visualization. Graphs can help us find data trends that are useful in any machine learning or forecasting project. Graphs make it easier to explain your data to non-technical people. Visually attractive graphs can make presentations and reports much more appealing to the reader.

Installing Seaborn:

If you are working on Google Collab, you can skip the installation step. However, if you are working on your local machine, you will need to install Seaborn. I highly recommend creating a virtual environment to better manage your packages.

```
python -m venv venv
venv/Scripts/activate
pip install pandas, matplotlib, seaborn
```

Fig. 3.2: Installing Seaborn

Seaborn Plot types:

Seaborn provides a wide range of plot types that can be used for data visualization and exploratory data analysis. Broadly speaking, any visualization can fall into one of the three categories.

a) Univariate – x only (contains only one axis of information)
b) Bivariate – x and y (contains two axis of information)
c) Trivariate – x, y, z (contains three axis of information)

Here are some of the most used plot types in Seaborn:

- **Scatter Plot.** A scatter plot is used to visualize the relationship between two variables. Seaborn's scatterplot() function provides a simple way to create scatter plots.
- **Line Plot.** A line plot is used to visualize the trend of a variable over time. Seaborn's lineplot() function provides a simple way to create line plots.
- **Histogram.** A histogram is used to visualize the distribution of a variable. Seaborn's histplot() function provides a simple way to create histograms.
- **Box Plot.** A box plot is used to visualize the distribution of a variable. Seaborn's boxplot () function provides a simple way to create box plots.
- **Violin Plot.** A violin plot is like a box plot but provides a more detailed view of the distribution of the data. Seaborn's violinplot() function provides a simple way to create violin plots.
- **Heatmap.** A heatmap is used to visualize the correlation between different variables. Seaborn's heatmap () function provides a simple way to create heatmaps.
- **Pairplot.** A pairplot is used to visualize the relationship between multiple variables. Seaborn's pairplot() function provides a simple way to create pairplots.

Best practices for Seaborn visualization:

Choose the right plot type for your data:

Seaborn provides a wide range of plot types, each designed for different types of data and analysis. It's important to choose the right plot type for your data to effectively communicate your findings. For example, a scatter plot may be more appropriate for visualizing the relationship between two continuous variables, while a bar plot may be more appropriate for visualizing categorical data.

Use color effectively:

Color can be a powerful tool for data visualization, but it's important to use it effectively. Avoid using too many colors or overly bright colors, as this can make the visualization difficult to read. Instead, use color to highlight important information or to group similar data points.

Label your axes and use clear titles:

Labels and titles are essential for effective data visualization. Make sure to label your axes clearly and provide a descriptive title for your visualization. This will help your audience understand the message you are trying to convey.

Consider the audience

When creating visualizations, it's important to consider the audience and the message you are trying to communicate. If your audience is non-technical, use clear and concise language, avoid technical jargon, and provide clear explanations of any statistical concepts.

Use appropriate statistical analysis:

Seaborn provides a range of statistical functions that you can use to analyze your data. When choosing a statistical function, make sure to choose the one that is most appropriate for your data and research question.

Customize your visualizations:

You'll find a wide range of customization options in Seaborn that you can use to enhance your visualizations. Experiment with different fonts, styles, and colors to find the one that best communicates your message.

3.3 Hardware / Software Requirements

- Minimums Hardware Requirements:
  - RAM: 2 GB
  - Processor: Intel Core 2 Duo
  - Hard disk: 50GB
- Minimums Software Requirements:
  - Pandas: 2.0.1
  - Numpy; 1.24.3
  - Matplotlib: 3.7.1
  - Seaborn: 11.2
  - Python: 3.8.16

3.4 Our Methodology:

- **Setup of Environment -**

- Firstly, we obtain the data set from Kaggle "**Zomato Restaurants Data**" of world, which contains information about various restaurants across the globe.
- The next step in our data analysis project is to set up a jupyter notebook and import the necessary libraries.
- Jupyter notebooks are interactive environments that allow us to write and execute code, as well as document our findings.
- To import the libraries, we use the following code snippet:

```
In [1]:
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Fig. 3.3: Importing libraries

Importing necessary libraries.

- import pandas as pd: load a library for data analysis
- import numpy as np: load a library for numerical computation
- import seaborn as sns: load a library for data visualization
- import matplotlib.pyplot as plt: load another library for data visualization

➢ **Reading the data:**

To read the data set, we use the following code snippet:

```
In [2]:
df=pd.read_excel("zomatoWorld.xlsx")
country = pd.read_excel('Country-Code.xlsx')
zmt = pd.merge(df, country, on='Country Code')
```

Fig. 3.4: read_excel()

➢ **Describing the data:**

```
In [62]: zmt.describe()
```

Out[62]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	Votes
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666370	156.909748
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378	430.169145
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	0.000000
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500000	5.000000
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200000	31.000000
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000	131.000000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	10934.000000

Fig. 3.5: describe() function

- The main task of describe () in data analysis is to get a summary of the numerical variables in a data set.
- The **describe()** function in Python can help with this task by returning some basic statistics such as the mean, median, standard deviation, minimum and maximum values of each numerical column.

```
In [8]: zmt.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9551 entries, 0 to 9550
Data columns (total 22 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Restaurant ID         9551 non-null   int64  
 1   Restaurant Name       9551 non-null   object 
 2   Country Code          9551 non-null   int64  
 3   City                  9551 non-null   object 
 4   Address               9551 non-null   object 
 5   Locality              9551 non-null   object 
 6   Locality Verbose      9551 non-null   object 
 7   Longitude             9551 non-null   float64
 8   Latitude              9551 non-null   float64
 9   Cuisines              9542 non-null   object 
 10  Average Cost for two  9551 non-null   int64  
 11  Currency              9551 non-null   object 
 12  Has Table booking     9551 non-null   object 
 13  Has Online delivery   9551 non-null   object 
 14  Is delivering now     9551 non-null   object 
 15  Switch to order menu  9551 non-null   object 
 16  Price range           9551 non-null   int64  
 17  Aggregate rating      9551 non-null   float64
 18  Rating color          9551 non-null   object 
 19  Rating text           9551 non-null   object 
 20  Votes                 9551 non-null   int64  
 21  Country               9551 non-null   object 
dtypes: float64(3), int64(5), object(14)
memory usage: 1.7+ MB
```

Fig. 3.6: info() function

- The **info()** function is a useful tool for exploring the attributes of an object. It displays the name, type, and value of each attribute.
- The data frame has 22 columns with different data types. And these 22 are as follows-
  - **Three** columns have floating-point numbers, which can store decimal values.
  - **Five** columns have integers, which can store whole numbers.
  - **Fourteen** columns have objects, which can store strings or other types of data.
- The dataset has 9551 entries, which means it has 9551 unique integer labels. The labels range from 0 to 9550, inclusive.

```
In [3]: #Top 5 row of dataset from upper side.(by default)
        zmt.head()
```

Out[3]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Has Table booking	Has Online delivery	Is delivering now	Swi or m
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts	...	Yes	No	No	
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese	...	Yes	No	No	
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian	...	Yes	No	No	
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi	...	No	No	No	
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean	...	Yes	No	No	

5 rows × 22 columns

Fig. 3.7: head() function

Using the **head() function** to get a basic idea about how the data is entered.

- **All COLUMN:-**

```
In [7]: zmt.columns
Out[7]: Index(['Restaurant ID', 'Restaurant Name', 'Country Code', 'Cit
        y', 'Address',
               'Locality', 'Locality Verbose', 'Longitude', 'Latitude',
        'Cuisines',
               'Average Cost for two', 'Currency', 'Has Table booking',
               'Has Online delivery', 'Is delivering now', 'Switch to o
        rder menu',
               'Price range', 'Aggregate rating', 'Rating color', 'Rati
        ng text',
               'Votes', 'Country'],
              dtype='object')
```

Fig. 3.8: All columns

- **CHECKING DUPLICATE VALUES FOR EACH COLUMN**

```
In [11]: #total number of duplicate value in dataset
         zmt.duplicated().sum()
Out[11]: 0
```

Fig. 3.9: duplicated().sum()

- There is no duplicate value in our dataset. This means that each row of data is unique and represents a distinct observation. Having no duplicate value in our dataset is important for data quality and analysis accuracy.

➢ CHECKING MISSING VALUES FOR EACH COLUMN

```
In [12]: zmt.isnull().sum()

Out[12]: Restaurant ID         0
         Restaurant Name       0
         Country Code          0
         City                  0
         Address               0
         Locality              0
         Locality Verbose      0
         Longitude             0
         Latitude              0
         Cuisines              9
         Average Cost for two  0
         Currency              0
         Has Table booking     0
         Has Online delivery   0
         Is delivering now     0
         Switch to order menu  0
         Price range           0
         Aggregate rating      0
         Rating color          0
         Rating text           0
         Votes                 0
         Country               0
         dtype: int64
```

Fig. 3.10: isnull().sum() function

- In the data set, column Cuisines contains information about the types of food served by different restaurants. However, some of the rows in this column have missing values, which are represented by NaN.
- So, the rows with NaN values are-

```
In [63]: zmt[zmt['Cuisines'].isna()]
```

Out[63]:

City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Has Table booking	H Onli delive
Albany	115 N Jackson St, Albany, GA 31701	Albany	Albany, Albany	-84.154000	31.577200	NaN	...	No	
Albany	814 N Slappey Blvd, Albany, GA 31701	Albany	Albany, Albany	-84.175900	31.588200	NaN	...	No	
Albany	204 S Jackson St, Albany, GA 31701	Albany	Albany, Albany	-84.153400	31.575100	NaN	...	No	
Gainesville	51 W Main St, Dahlonega, GA 30533	Dahlonega	Dahlonega, Gainesville	-83.985800	34.531800	NaN	...	No	
Macon	543 Cherry St, Macon, GA 31201	Macon	Macon, Macon	-83.627979	32.836410	NaN	...	No	
Miller	109 N Broadway Ave, Miller, SD 57362	Miller	Miller, Miller	-98.989100	44.515800	NaN	...	No	
Orlando	215 South Orlando Avenue, Winter Park, FL 32789	Winter Park	Winter Park, Orlando	-81.365260	28.596682	NaN	...	No	
Rest of Hawaii	933 Kapahulu Ave, Honolulu, HI 96816	Kaimuki	Kaimuki, Rest of Hawaii	-157.813432	21.284586	NaN	...	No	
Savannah	1311 Butler Ave, Tybee Island, GA 31328	Tybee Island	Tybee Island, Savannah	-80.848297	31.995810	NaN	...	No	

Fig. 3.11: Rows with "NaN"

- To deal with this issue, we will drop the rows with NaN values.

➢ **REMOVE ROWS WITH MISSING VALUES**

- Then using **dropna()** function we remove all Rows with NaN values.
- It can be seen that total row is reduced to 9542 from 9551.

```
In [67]: zmt=zmt.dropna()
         zmt
```

Out[67]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Has Table booking	Has Online delivery	deliver r
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts	...	Yes	No	
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese	...	Yes	No	
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian	...	Yes	No	
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi	...	No	No	
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean	...	Yes	No	

Fig. 3.12: Removing rows with "NaN"

```
In [68]: zmt.isnull().sum()
```

```
Out[68]: Restaurant ID         0
         Restaurant Name       0
         Country Code          0
         City                  0
         Address               0
         Locality              0
         Locality Verbose      0
         Longitude             0
         Latitude              0
         Cuisines              0
         Average Cost for two  0
         Currency              0
         Has Table booking     0
         Has Online delivery   0
         Is delivering now     0
         Switch to order menu  0
         Price range           0
         Aggregate rating      0
         Rating color          0
         Rating text           0
         Votes                 0
         Country               0
         dtype: int64
```

Fig. 3.13: isnull().sum()

- The dataset does not have any null values, which means there is no missing or unknown data. This makes the dataset clean and ready for further analysis.

➢ ANALYSING EACH ATTRIBUTE

○ Which country have more restaurant?

```
In [15]: #groupby country code
zmt_country = zmt.groupby(['Country'], as_index=False).count()
[['Country', 'Restaurant ID']]
zmt_country.columns = ['Country', 'No of Restaurant']
zmt_country
```

Out[15]:

	Country	No of Restaurant
0	Australia	24
1	Brazil	60
2	Canada	4
3	India	8652
4	Indonesia	21
5	New Zealand	40
6	Phillipines	22
7	Qatar	20
8	Singapore	20
9	South Africa	60
10	Sri Lanka	20
11	Turkey	34
12	UAE	60
13	United Kingdom	80
14	United States	434

Fig. 3.14: No Of Restaurants in country

```
In [16]: # No of restaurant that are on zomato in each country
         plt.bar(zmt_country['Country'], zmt_country['No of Restaurant'])
         plt.xlabel('Country', fontsize=20)
         plt.rcParams['figure.figsize']=(30,20)
         plt.ylabel('No of Restaurant', fontsize=20)
         plt.title('Restaurant in different country', fontsize=30)
         plt.xticks(rotation = 60)
```

```
Out[16]: ([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],
          [Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, ''),
           Text(0, 0, '')])
```

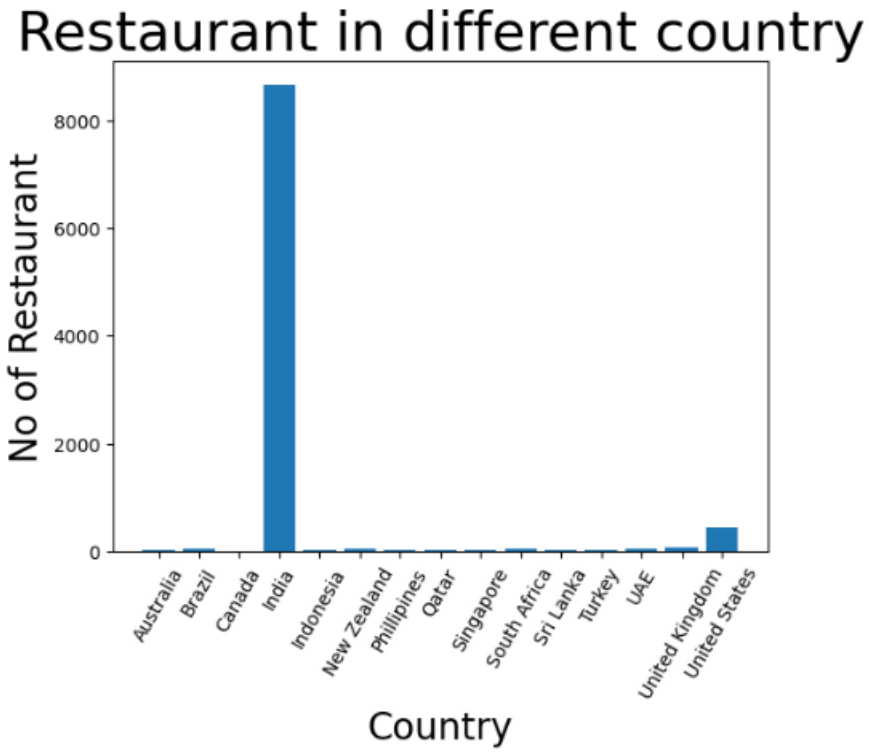


Fig. 3.15: Restaurants In Different Country

- Above Visualization, shows the Restaurants distribution among the country in our dataset.
- According to dataset India have maximum number of restaurants.

Let's separate the data for restaurants from India for ease of analysis. Here in this data set number of restaurant of India is maximum so all conclusion will be based on Indian restaurant.

```
In [17]: zmt_india = zmt.loc[zmt['Country']=='India']
         zmt_india.head()
```

Out[17]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	C
624	3400025	Jahanpanah	1	Agra	E 23, Shopping Arcade, Sadar Bazaar, Agra Cant...	Agra Cantt	Agra Cantt, Agra	78.011544	27.161661	
625	3400341	Rangrezz Restaurant	1	Agra	E-20, Shopping Arcade, Sadar Bazaar, Agra Cant...	Agra Cantt	Agra Cantt, Agra	0.000000	0.000000	
626	3400005	Time2Eat - Mama Chicken	1	Agra	Main Market, Sadar Bazaar, Agra Cantt, Agra	Agra Cantt	Agra Cantt, Agra	78.011608	27.160832	
627	3400021	Chokho Jeeman Marwari Jain Bhojanalya	1	Agra	1/48, Delhi Gate, Station Road, Raja Mandi, Ci...	Civil Lines	Civil Lines, Agra	77.998092	27.195928	Ra

Fig. 3.16: Indian Restaurants

- **Which city have maximum number of restaurants?**

```
In [18]: zmt_City = zmt[zmt['Country'] =='India']
         Total_city =zmt_City['City'].value_counts()
         Total_city.plot.bar(figsize=(14, 7),fontsize=14)
         plt.title('Restaurants by City', fontsize=30)
         plt.xlabel('City', fontsize=20)
         plt.ylabel('No of Restaurants', fontsize=20)
         plt.show()
```

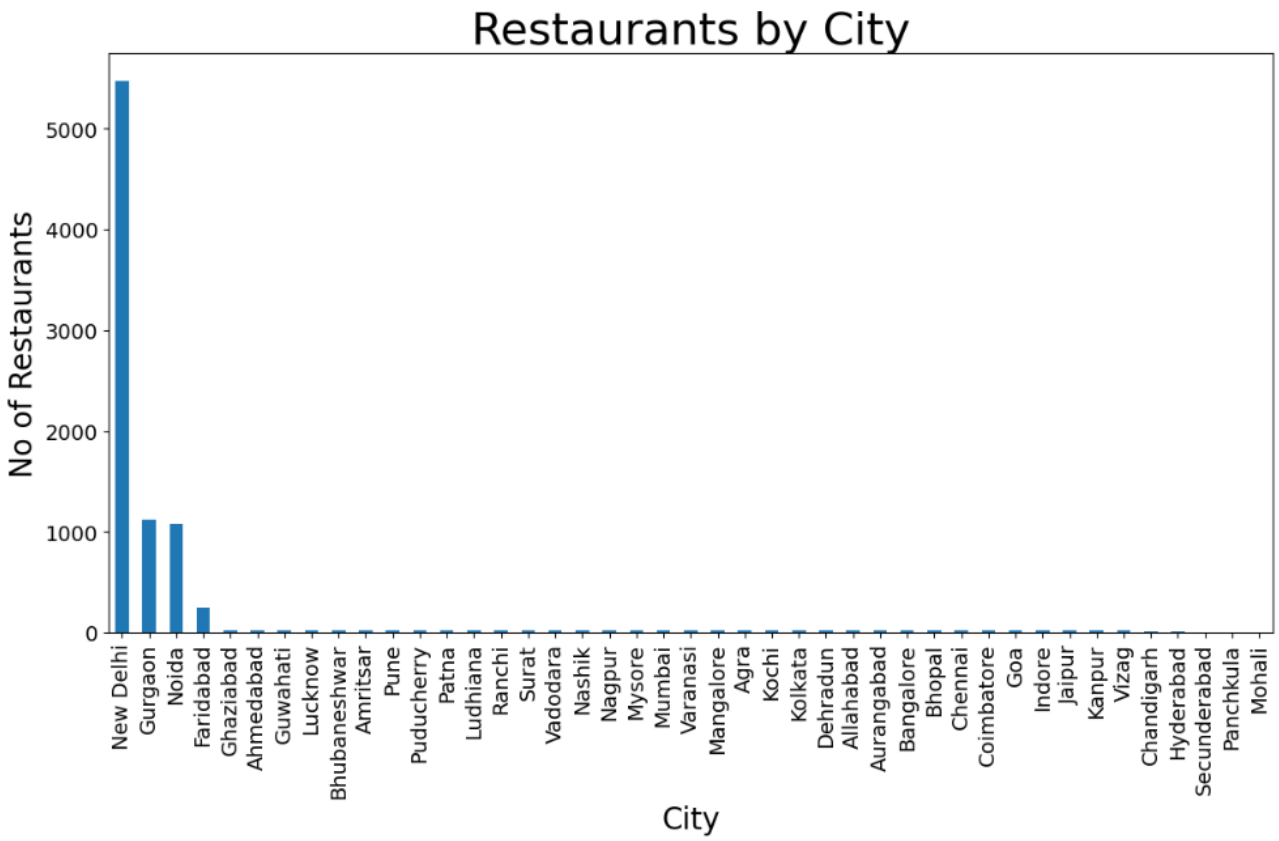


Fig. 3.17: No of Restaurants in city(bar plot)

```
In [19]: zmt_City['City'].value_counts()

Out[19]: New Delhi       5473
         Gurgaon         1118
         Noida           1080
         Faridabad        251
         Ghaziabad         25
         Ahmedabad         21
         Guwahati          21
         Lucknow           21
         Bhubaneshwar      21
         Amritsar          21
         Pune              20
         Puducherry        20
         Patna             20
         Ludhiana          20
         Ranchi            20
         Surat             20
         Vadodara          20
         Nashik            20
         Nagpur            20
         Mysore            20
         Mumbai            20
         Varanasi          20
         Mangalore         20
         Agra              20
         Kochi             20
         Kolkata           20
         Dehradun          20
         Allahabad         20
         Aurangabad        20
         Bangalore         20
         Bhopal            20
         Chennai           20
         Coimbatore        20
         Goa               20
         Indore            20
         Jaipur            20
         Kanpur            20
         Vizag             20
         Chandigarh        18
         Hyderabad         18
         Secunderabad       2
         Panchkula          1
         Mohali             1
         Name: City, dtype: int64
```

Fig. 3.18: No of Restaurants in city(Table)

New Delhi has maximum number of restaurants. Clearly this data has maximum number of restaurants listed in Delhi NCR region (Comprising of Gurgaon, Noida, Faridabad). Other cities have almost equal distribution but significantly less as compared to Delhi NCR region.

So, we take only restaurants located in a New Delhi. To narrow down our search, we focus on restaurants that are situated in New Delhi. This city is the capital of India and has a rich culinary heritage. We want to explore the different cuisines and dishes that New Delhi has to offer.

```
In [20]: zmt1 = zmt_india[zmt_india['City'] =='New Delhi']
         zmt1['Locality'].unique()
```

```
In [21]: zmt1['Locality'].nunique()
Out[21]: 254
```

Fig. 3.19: No of Locality

- The dataset contains 254 distinct values for the location feature, meaning that there are 254 unique locations in the data. This indicates that the dataset is diverse and covers a wide range of geographical areas.

- **Which restaurants are available for online orders?**

```
In [22]: fig, ax = plt.subplots(figsize=(10, 6))
ax = sns.countplot(y="Has Online delivery", data=zmt1)
plt.title('Restaurants available for online orders',size=20)
plt.xlabel('No of Restaurants ',size=20)

total = len(zmt1['Has Online delivery'])
for p in ax.patches:
        percentage = '{:.1f}%'.format(100 * p.get_width()/total)
        x = p.get_x() + p.get_width() + 0.02
        y = p.get_y() + p.get_height()/2
        ax.annotate(percentage, (x, y))

plt.show()
```

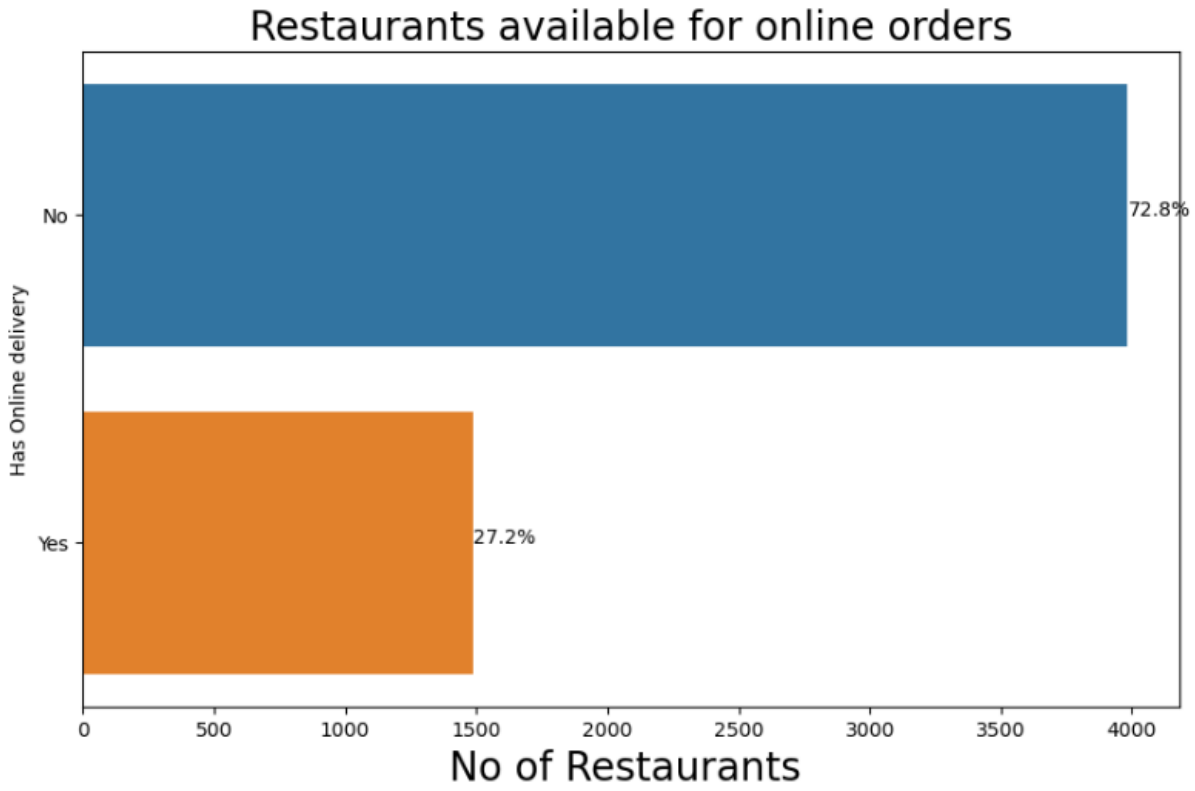


Fig. 3.20: Online order service(bar-plot)

```
In [23]: labels = ['Not Accepted',"Accepted"]
values = zmt1['Has Online delivery'].value_counts().values
fig1, ax1 = plt.subplots(figsize=(10, 6))
colors = ['red', 'yellow']
ax1.pie(values, labels=labels, autopct='%1.1f%%',shadow=True,
        startangle=90,colors=colors)
plt.title('Online order')
plt.show()
```

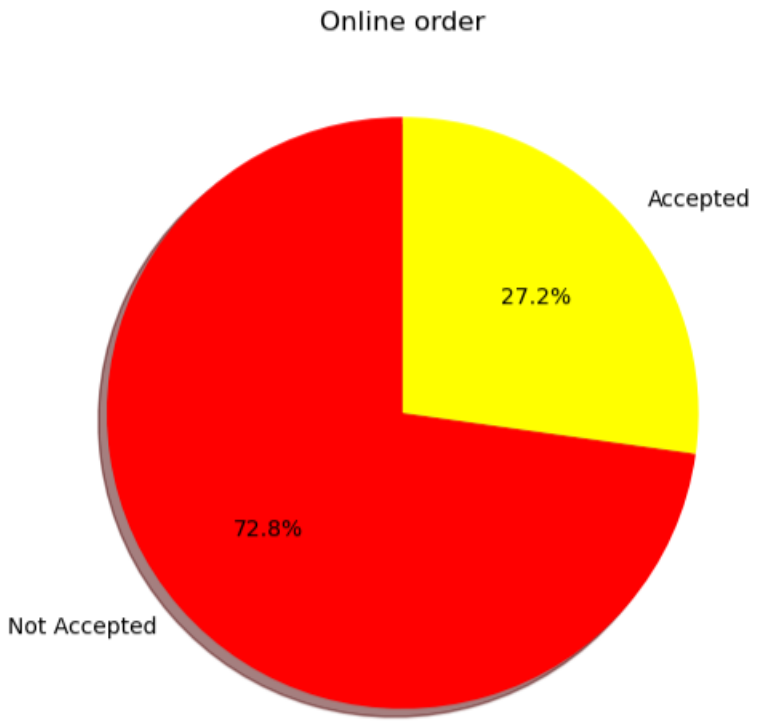


Fig. 3.21: Online order service(pie-plot)

- **Visualizing Online delivery, Location Wise**

```
In [24]: df = zmt1.groupby(['Locality','Has Online delivery'])['Restaurant Name'].
         count()
df.to_csv('location_online_delivery.csv')
df =pd.read_csv('location_online_delivery.csv')
df = pd.pivot_table(df,values=None,index=['Locality'],
              columns=['Has Online delivery'],fill_value=0,aggfunc=np.sum)
df1=df.head(10)
df1
```

Out[24]:

Has Online delivery	Restaurant Name	
	No	Yes
Locality		
ARSS Mall, Paschim Vihar	1	0
Aaya Nagar	1	0
Adchini	5	8
Aditya Mega Mall, Karkardooma	2	2
Aerocity	4	0
Aggarwal City Mall, Pitampura	1	2
Aggarwal City Plaza, Rohini	2	3
Alaknanda	23	9
Ambience Mall, Vasant Kunj	6	6
Anand Lok	1	1

In [25]:
```
df1.plot(kind = 'bar',figsize=(20,12))
```

Out[25]: <AxesSubplot:xlabel='Locality'>

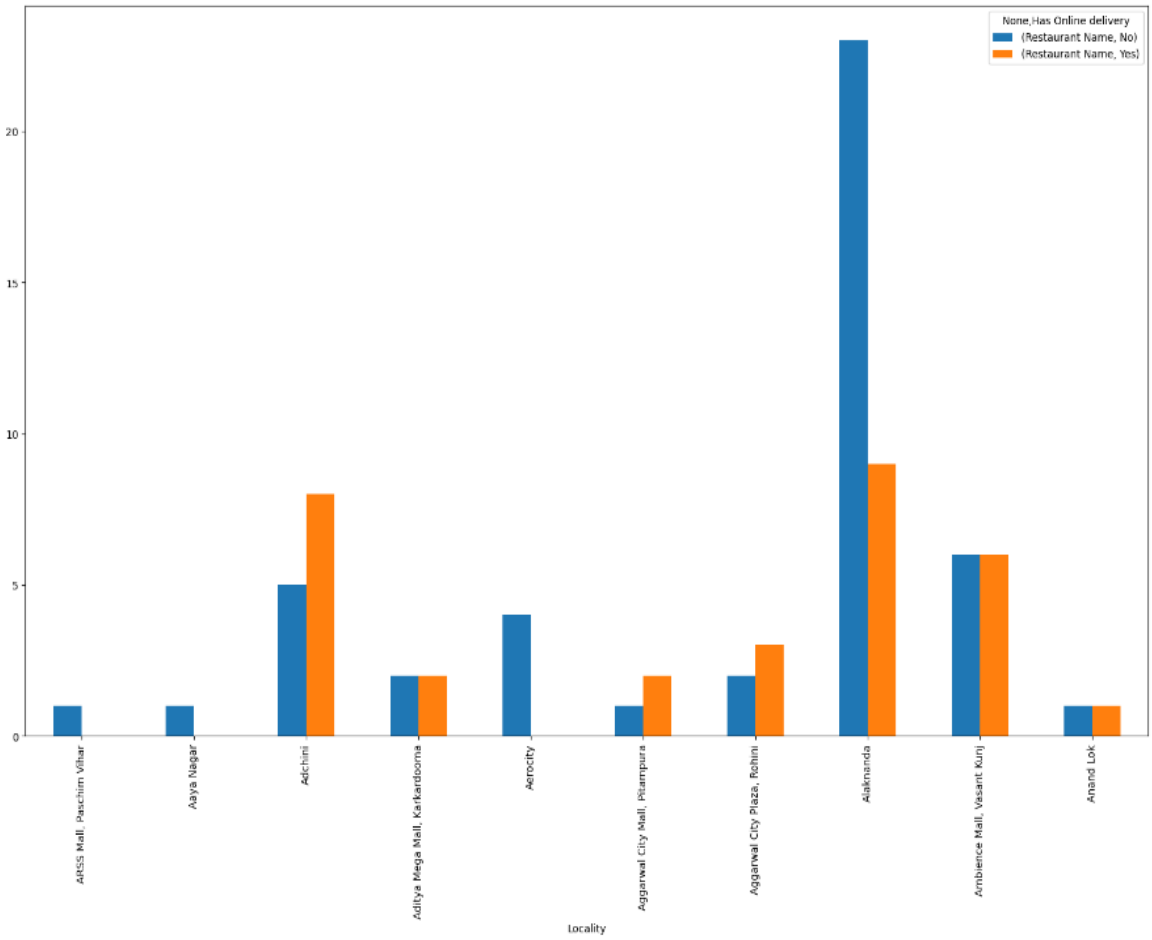


Fig. 3.22: Online order in locality

- **How many restaurants allow table booking?**

In [26]:
```
fig, ax = plt.subplots(figsize=(10, 6))
ax = sns.countplot(y="Has Table booking", data=zmt1)
plt.title('Restaurants offer book_table or not',size=20)
plt.xlabel('No of Restaurants ',size=15)

total = len(zmt1['Has Table booking'])
for p in ax.patches:
        percentage = '{:.1f}%'.format(100 * p.get_width()/total)
        x = p.get_x() + p.get_width() + 0.02
        y = p.get_y() + p.get_height()/2
        ax.annotate(percentage, (x, y))

plt.show()
```

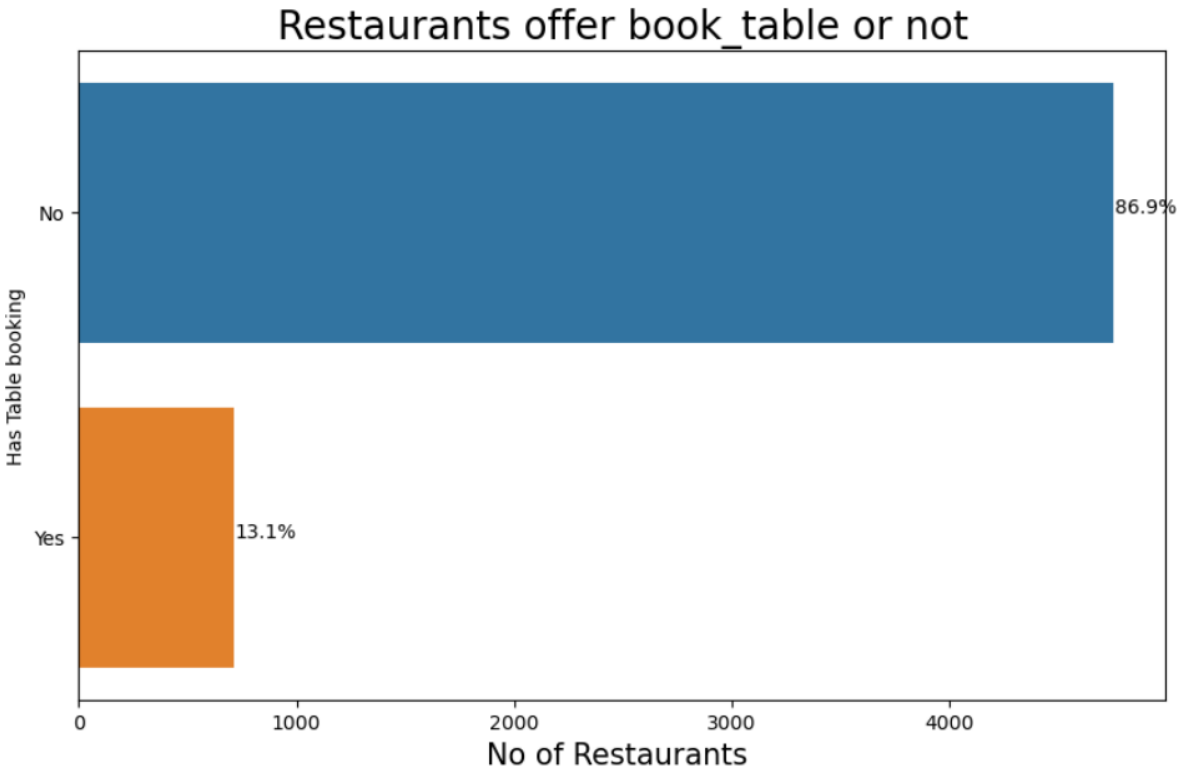


Fig. 3.23: Table Booking Service(bar-plot)

```
In [27]: labels = ['Not Accepted',"Accepted"]
values = zmt1['Has Table booking'].value_counts().values
fig1, ax1 = plt.subplots(figsize=(10, 6))
colors = ['red', 'yellow']
ax1.pie(values, labels=labels, autopct='%1.1f%%',shadow=True,
        startangle=90,colors=colors)
plt.title('Table booking')
plt.show()
```

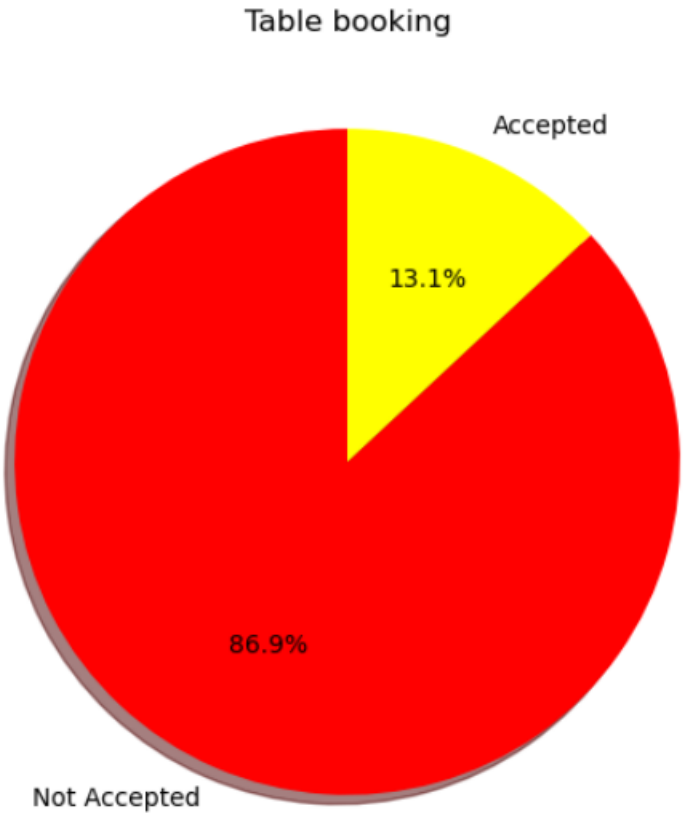


Fig. 3.24: Table Booking Service(pie-plot)

In our dataset 86.9% restaurants do not allow book table.

- **Visualizing Book Table Facility, Location Wise**

```
In [28]: df_table = zmt1.groupby(['Locality','Has Table booking'])['Restaurant Name']
.count()
df_table .to_csv('location_book_table.csv')
df_table  =pd.read_csv('location_book_table.csv')
df_table  = pd.pivot_table(df_table ,values=None,index=['Locality'],
                           columns=['Has Table booking'],fill_value=0,aggfunc=np.sum)
df_table1=df.head(20)
df_table1
```

Out[28]:

	Restaurant Name	
Has Online delivery	No	Yes
Locality		
ARSS Mall, Paschim Vihar	1	0
Aaya Nagar	1	0
Adchini	5	8
Aditya Mega Mall, Karkardooma	2	2
Aerocity	4	0
Aggarwal City Mall, Pitampura	1	2
Aggarwal City Plaza, Rohini	2	3
Alaknanda	23	9
Ambience Mall, Vasant Kunj	6	6
Anand Lok	1	1
Anand Vihar	35	7
Andaz Delhi, Aerocity	1	0
Ansal Plaza Mall, Khel Gaon Marg	5	3
Asaf Ali Road	3	0
Ashok Vihar Phase 1	31	5
Ashok Vihar Phase 2	6	8
Ashok Vihar Phase 3	9	2
Barakhamba Road	10	6
Basant Lok Market, Vasant Vihar	6	9
Bellagio, Ashok Vihar Phase 2	0	3

In [29]:
```
df_table1.plot(kind = 'bar',figsize=(20,12))
```

Out[29]:
```
<AxesSubplot:xlabel='Locality'>
```

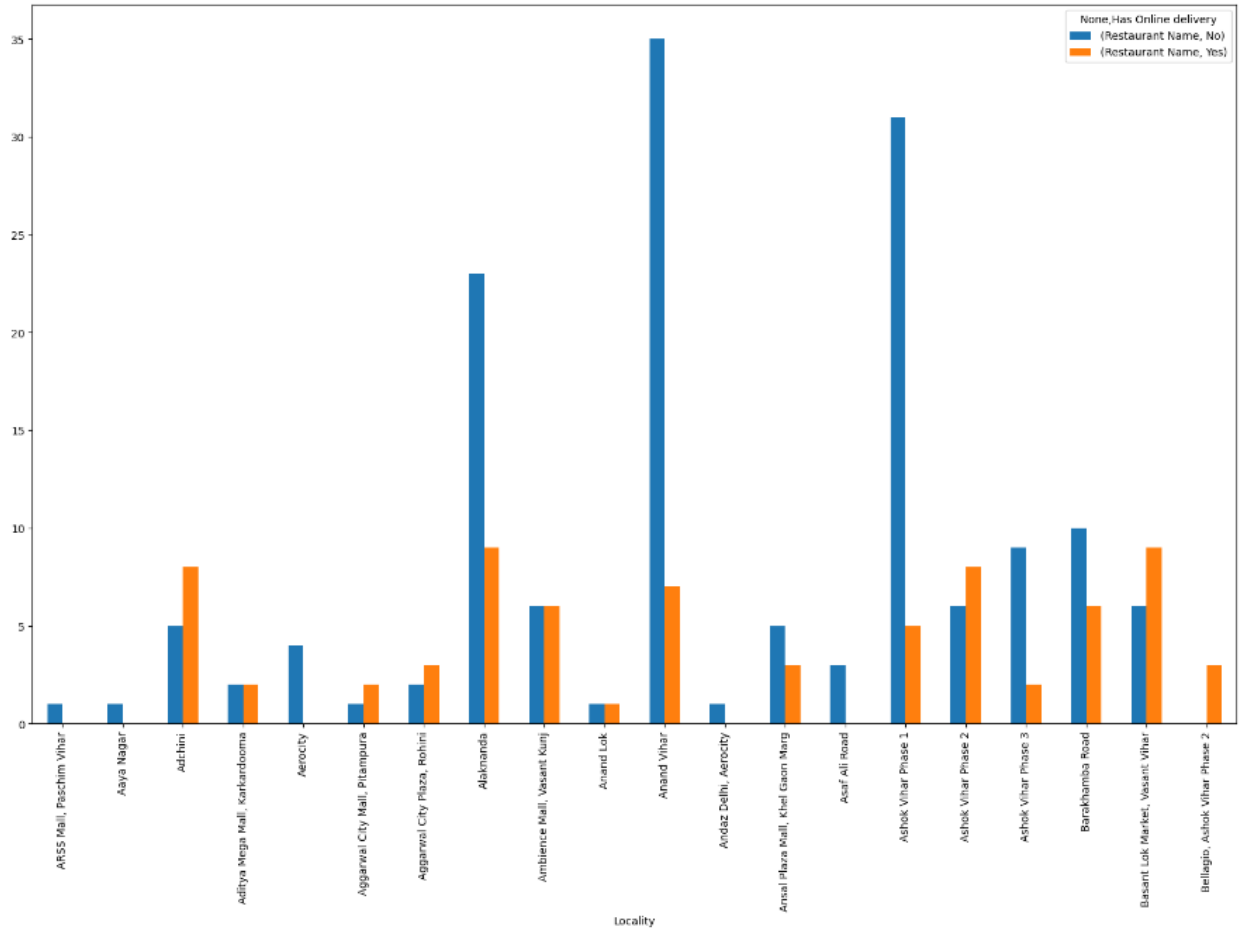


Fig. 3.25: Table booking facility location wise

- **Highest average Ratings on Locality**

```
In [30]: average_ratings = zmt1.groupby(['Locality'], as_index=False)
         average_ratings_agg = average_ratings['Aggregate rating'].agg(np.mean)
         average_ratings_agg_sort = average_ratings_agg.sort_values
         (by='Aggregate rating', ascending=False)
         df_rating=average_ratings_agg_sort.head(20)
         df_rating
```

Out[30]:

	Locality	Aggregate rating
106	Kasbah, Greater Kailash (GK) 1	4.500000
63	Garden of Five Senses, Saket	4.200000
202	Sheraton New Delhi Hotel, Saket	4.066667
60	Friends Colony	4.050000
177	Radisson Blu Plaza Delhi, Mahipalpur	3.950000
168	Pragati Maidan	3.900000
198	Shangri La's - Eros hotel, Janpath	3.880000
188	Sangam Courtyard, RK Puram	3.871429
162	Pandara Road Market	3.860000
73	Hauz Khas Village	3.845833
228	The Village Restaurant Complex, Khel Gaon Marg	3.833333
81	Hyatt Regency, Bhikaji Cama Place	3.800000
26	Chawri Bazar	3.800000
114	Lajpat Nagar 4	3.800000
138	Model Town 1	3.800000
226	The Taj Palace Hotel, Chanakyapuri	3.800000
9	Anand Lok	3.800000
225	The Taj Mahal Hotel, Mansingh Road	3.785714
221	The Leela Palace, Chanakyapuri	3.780000
220	The Leela Ambience Convention Hotel	3.775000

In [31]:
```
df_rating.plot(kind = 'bar',figsize=(20,12))
plt.ylabel('Rating',size=15)
plt.xlabel('Index of locality ',size=15)
```

Out[31]:
```
Text(0.5, 0, 'Index of locality ')
```

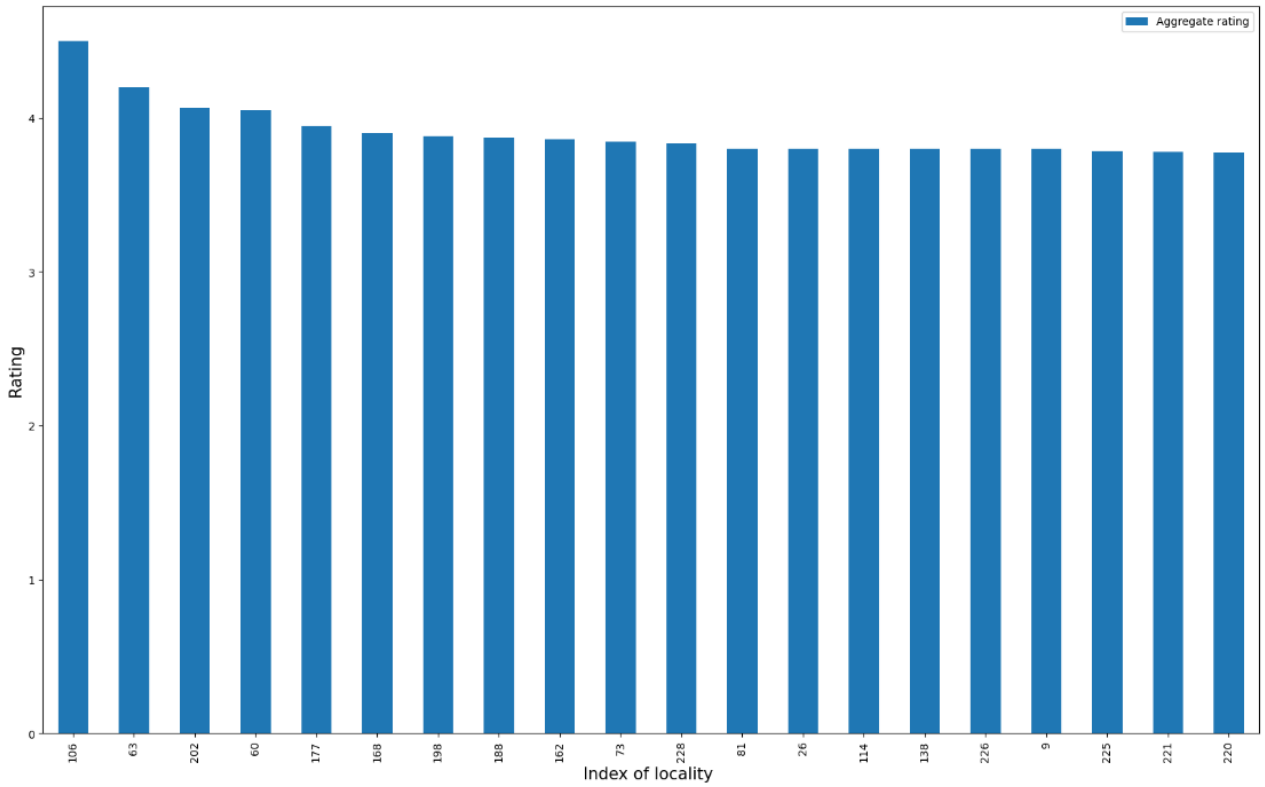


Fig. 3.26: Average rating on locality

From the above analysis, We can say that the restaurant located in "Kasbah, Greater Kailash (GK) 1" has the highest average rating It means that customers are satisfied with the service of a restaurants at that location.

- **Best Restaurant**

```
In [32]: Best_res= zmt1[zmt1['Locality'] =='Kasbah, Greater Kailash (GK) 1']
         # zmt1['Locality'].unique()
         Best_res['Restaurant Name']

Out[32]: 4631    Zaffran
         Name: Restaurant Name, dtype: object
```

Fig. 3.27: Best restaurant

The mentioned restaurant is best in "Kasbah, Greater Kailash (GK) 1" on the basis of rating.

- **Highest average Votes on Locality**

```
In [33]: average_Votes = zmt1.groupby(['Locality'], as_index=False)
         average_Votes_agg = average_Votes['Votes'].agg(np.mean)
         average_Votes_agg_sort = average_Votes_agg.sort_values
         (by='Votes', ascending=False)
         df_Votes=average_Votes_agg_sort.head(20)
         df_Votes
```

Out[33]:

	Locality	Votes
63	Garden of Five Senses, Saket	1561.000000
59	Feroze Shah Road	1505.500000
73	Hauz Khas Village	1357.208333
9	Anand Lok	1205.500000
168	Pragati Maidan	1156.500000
33	Connaught Place	1050.073770
60	Friends Colony	974.500000
162	Pandara Road Market	797.800000
138	Model Town 1	727.000000
108	Khan Market	646.931818
114	Lajpat Nagar 4	546.000000
106	Kasbah, Greater Kailash (GK) 1	524.000000
84	ITC Maurya, Chanakyapuri	520.166667
42	DLF Place Mall, Saket	493.333333
30	City Square Mall, Rajouri Garden	405.666667
241	Vijay Nagar	395.838710
123	MGF Metropolitan Mall, Saket	389.000000
223	The Park, Connaught Place	380.333333
65	Ginger Hotel, Vivek Vihar	378.500000
228	The Village Restaurant Complex, Khel Gaon Marg	377.666667

In [34]:
```
df_Votes.plot(kind = 'bar',figsize=(20,12))
plt.ylabel('Votes',size=15)
plt.xlabel('Index of locality ',size=15)
```

Out[34]:
```
Text(0.5, 0, 'Index of locality ')
```

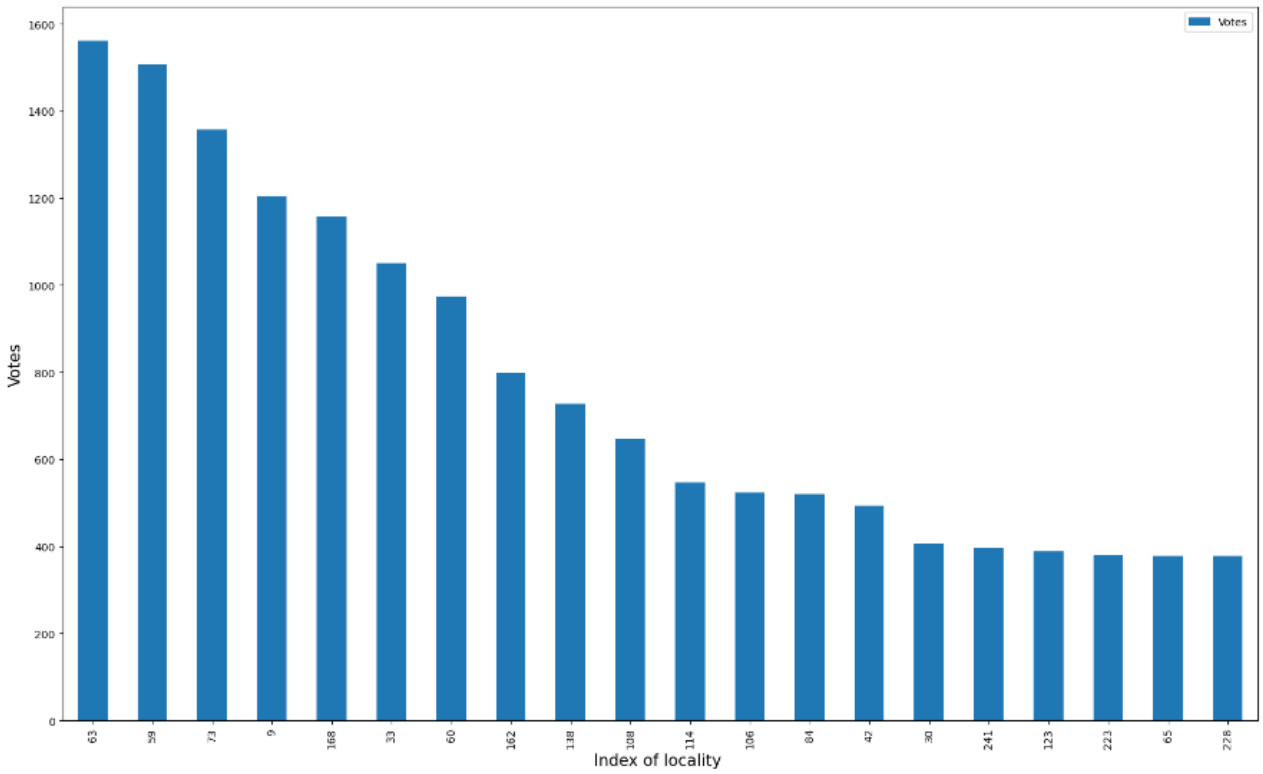


Fig. 3.28: Votes on locality

From above analysis, It shows that "Garden of Five Senses, Saket" has more customer.

- **Most visited restaurant.**

```
In [35]: M_visit_res= zmt1[zmt1['Locality'] =='Garden of Five Senses, Saket']
         # zmt1['Locality'].unique()
         M_visit_res['Restaurant Name']

Out[35]: 3605    FIO Country Kitchen and Bar
         Name: Restaurant Name, dtype: object
```

```
In [36]: Cuisine_data =zmt1.groupby(['Cuisines'], as_index=False)['Restaurant ID'].
         count()
         Cuisine_data.columns = ['Cuisines', 'Number of Resturants']
         Top20= (Cuisine_data.sort_values(['Number of Resturants'],ascending=False)).
         head(20)
         Top20
```

- **No of Cuisines on Zomato**

Out[36]:

	Cuisines	Number of Resturants
588	North Indian	658
600	North Indian, Chinese	284
369	Fast Food	242
184	Chinese	228
722	North Indian, Mughlai	207
105	Cafe	158
844	Street Food	123
48	Bakery	122
727	North Indian, Mughlai, Chinese	120
52	Bakery, Desserts	117
197	Chinese, Fast Food	99
789	Pizza, Fast Food	92
558	Mithai, Street Food	90
566	Mughlai	86
815	South Indian	81
62	Bakery, Fast Food	80
223	Chinese, North Indian	70
458	Ice Cream, Desserts	64
542	Mithai	59
673	North Indian, Fast Food	48

In [37]:
```
Cuisine_data =zmt1.groupby(['Cuisines'], as_index=False)['Restaurant ID'].
count()
Cuisine_data.columns = ['Cuisines', 'Number of Resturants']
Top20= (Cuisine_data.sort_values(['Number of Resturants'],ascending=False)).
head(20)
sns.set(rc={'figure.figsize':(11.7,8.27)})
sns.barplot(Top20['Cuisines'], Top20['Number of Resturants'])
plt.xlabel('Cuisines', fontsize=20)
plt.ylabel('Number of Resturants', fontsize=20)
plt.title('Top 20 Cuisines on Zomato', fontsize=30)
plt.xticks(rotation = 90)
plt.show()
```

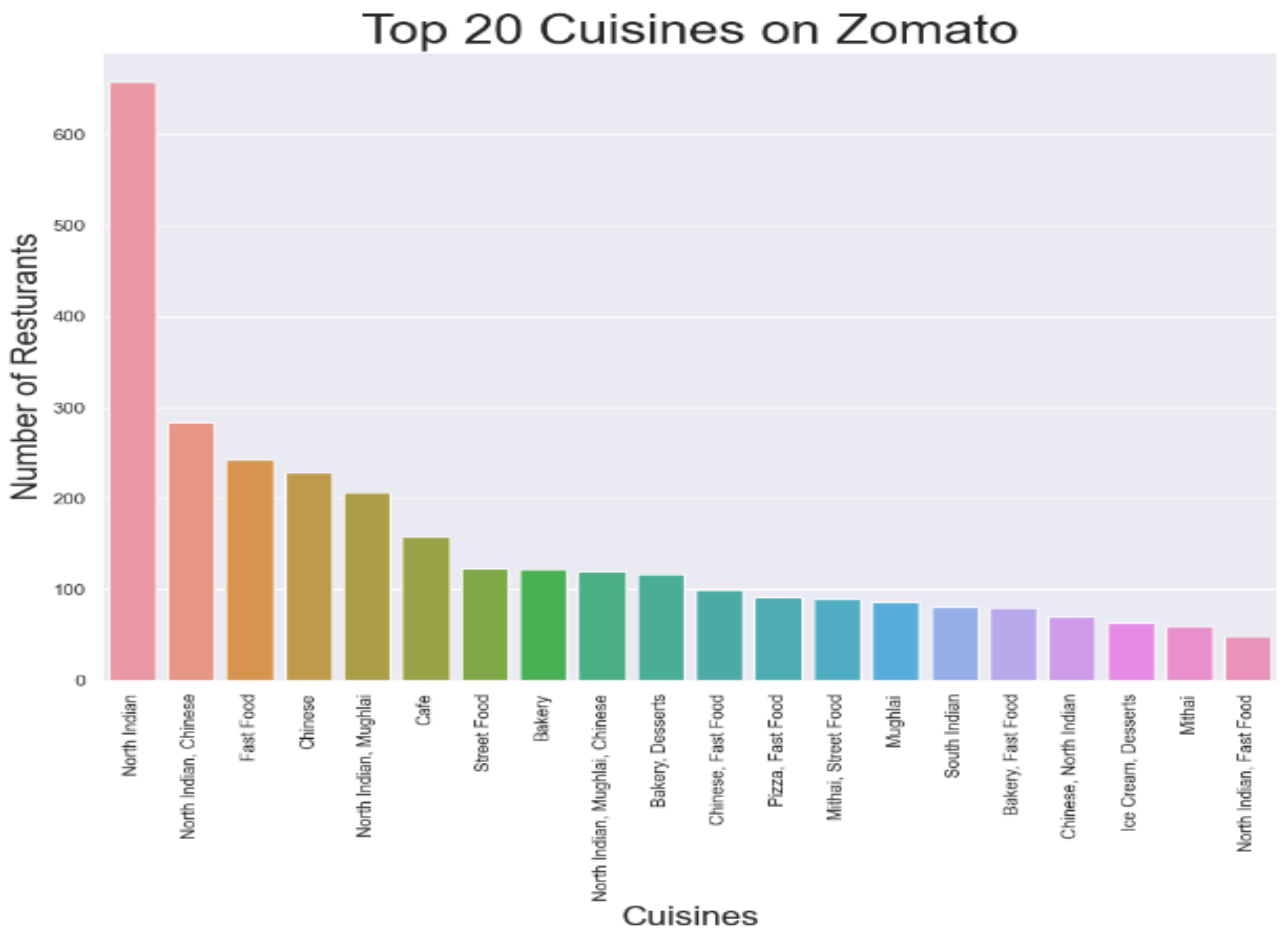


Fig. 3.29: No. of cuisines

- **Top Rated Cuisines on Zomato**

```
In [38]: Cuisine_data_rating=(zmt1.groupby(['Cuisines'], as_index=False)
                              ['Aggregate rating'].mean())
         Cuisine_data_rating.columns = ['Cuisines', 'Rating']
         Top30_ratings= (Cuisine_data_rating.sort_values(['Rating'],
                                                         ascending=False)).head(30)
         Top30_ratings
Out[38]:
```

	Cuisines	Rating
128	Cafe, Continental, Italian, Mexican, Chinese, ...	4.700
296	Continental, Italian, Asian, Indian	4.700
30	Asian, Chinese, Thai, Japanese	4.700
307	Continental, Mexican	4.600
4	American, Asian, European, Seafood	4.600
564	Modern Indian	4.580
305	Continental, Kerala	4.500
880	Thai, European, Mexican, North Indian, Chinese...	4.500
747	North Indian, South Indian, Bihari	4.500
98	Burger, American, Fast Food, Italian, Pizza	4.500
471	Italian, Continental, European, Cafe	4.475
527	Malaysian, Indonesian	4.400
78	Bengali, Seafood	4.400
578	Mughlai, Street Food	4.400
694	North Indian, Italian, Asian, American	4.400
155	Cafe, Italian, Continental, Chinese	4.400
622	North Indian, Chinese, Mexican, Italian, Thai,...	4.400
510	Kashmiri, North Indian, Mughlai, South Indian,...	4.400
29	Asian, Chinese, Thai	4.400
313	Continental, North Indian, American, Italian, ...	4.300
159	Cafe, Italian, Fast Food	4.300
714	North Indian, Mediterranean, Asian, Fast Food	4.300
65	Bakery, Fast Food, Desserts	4.300
655	North Indian, Continental, European, Chinese, ...	4.300
280	Continental, Chinese, Italian, Finger Food	4.300
590	North Indian, Afghani, Mughlai	4.300
156	Cafe, Italian, Continental, Fast Food	4.300
890	Turkish, Mediterranean, Middle Eastern	4.300
463	Ice Cream, Mithai, North Indian, Street Food	4.300
843	Spanish, Italian	4.300

```
In [39]: Cuisine_data_rating=(zmt1.groupby(['Cuisines'], as_index=False)
                              ['Aggregate rating'].mean())
         Cuisine_data_rating.columns = ['Cuisines', 'Rating']
         Top30_ratings= (Cuisine_data_rating.sort_values(['Rating'],ascending=False))
         .head(30)
         sns.set(rc={'figure.figsize':(11.7,8.27)})
         sns.barplot(Top30_ratings['Cuisines'], Top30_ratings['Rating'])
         plt.title('Top Rated Cuisines on Zomato', fontsize=30)
         plt.xlabel('Cuisines', fontsize=20)
         plt.ylabel('Rating', fontsize=20)
         plt.xticks(rotation = 90)
         plt.show()
```

Most served cuisines

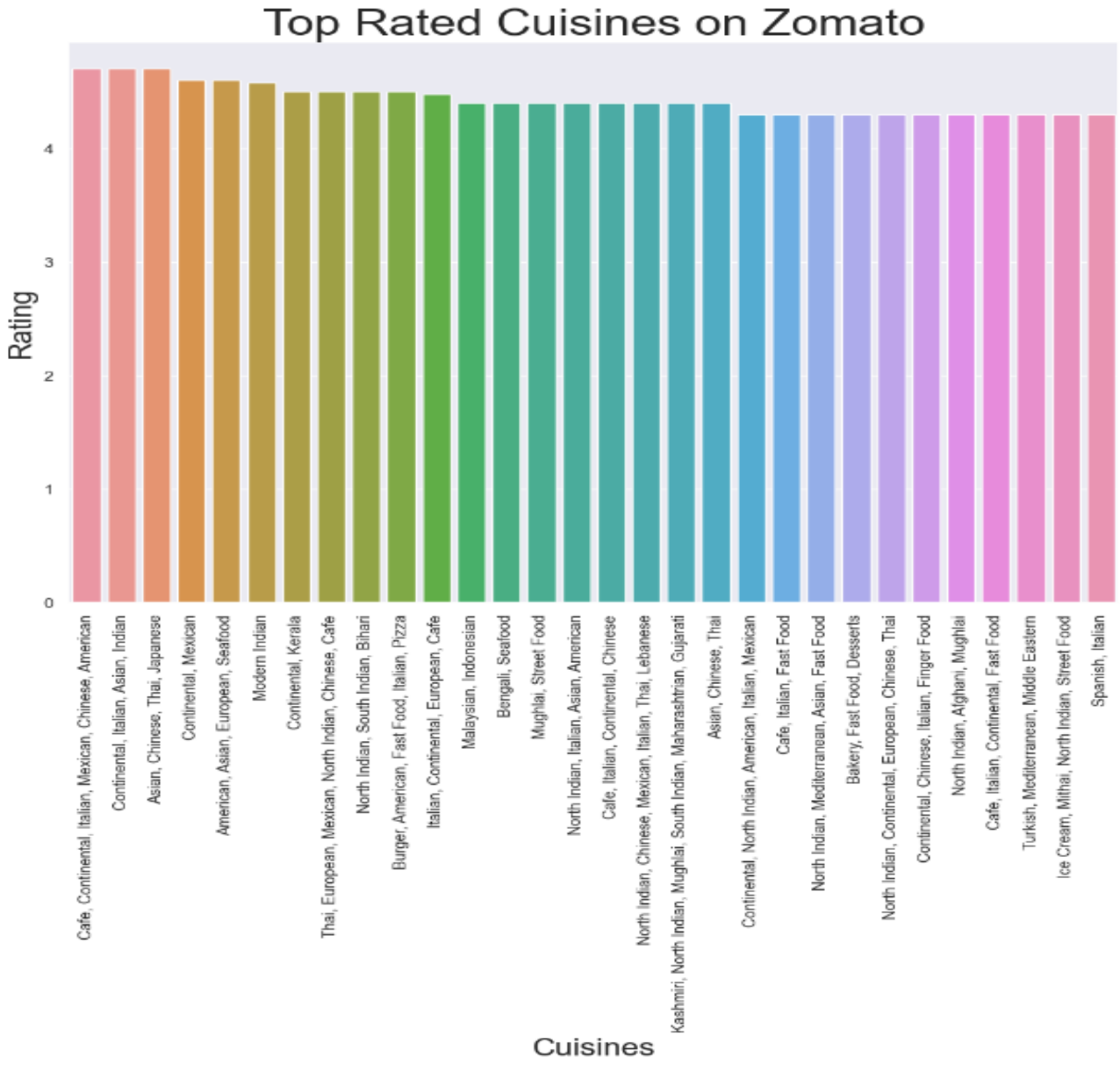


Fig. 3.30: Top rated cuisines

- **Most served cuisines**

In [40]:
```
df_served_cuis = zmt1[['Cuisines', 'Votes']]
df_served_cuis.drop_duplicates()
df_served_cuis1 = df_served_cuis.groupby(['Cuisines'])['Votes'].sum()
df_served_cuis1 = df_served_cuis1.to_frame()
df_served_cuis1 = df_served_cuis1.sort_values('Votes', ascending=False)
df_served_cuis2=df_served_cuis1.head(20)
df_served_cuis2
```

Out[40]:

Cuisines	Votes
North Indian, Mughlai	27951
North Indian	25673
North Indian, Chinese	19381
Cafe	16305
North Indian, Mughlai, Chinese	16003
South Indian	13820
Mughlai, North Indian	12872
Italian, Continental, European, Cafe	10853
Chinese	10771
Continental, American, Asian, North Indian	10688
Fast Food	9533
Street Food	8155
Chinese, Thai	7603
North Indian, Chinese, Italian, Continental	7156
Bakery, Desserts, Fast Food	6926
Pizza, Fast Food	6181
North Indian, Chinese, Mughlai	6005
Bakery, Desserts	5586
American, Fast Food	5537
Burger, Fast Food	5479

In [41]:
```
df_served_cuis2.plot(kind = 'bar',figsize=(20,12))
plt.ylabel('Votes',size=15)
plt.xlabel('Cuisines ',size=15)
```

Out[41]:
```
Text(0.5, 0, 'Cuisines ')
```

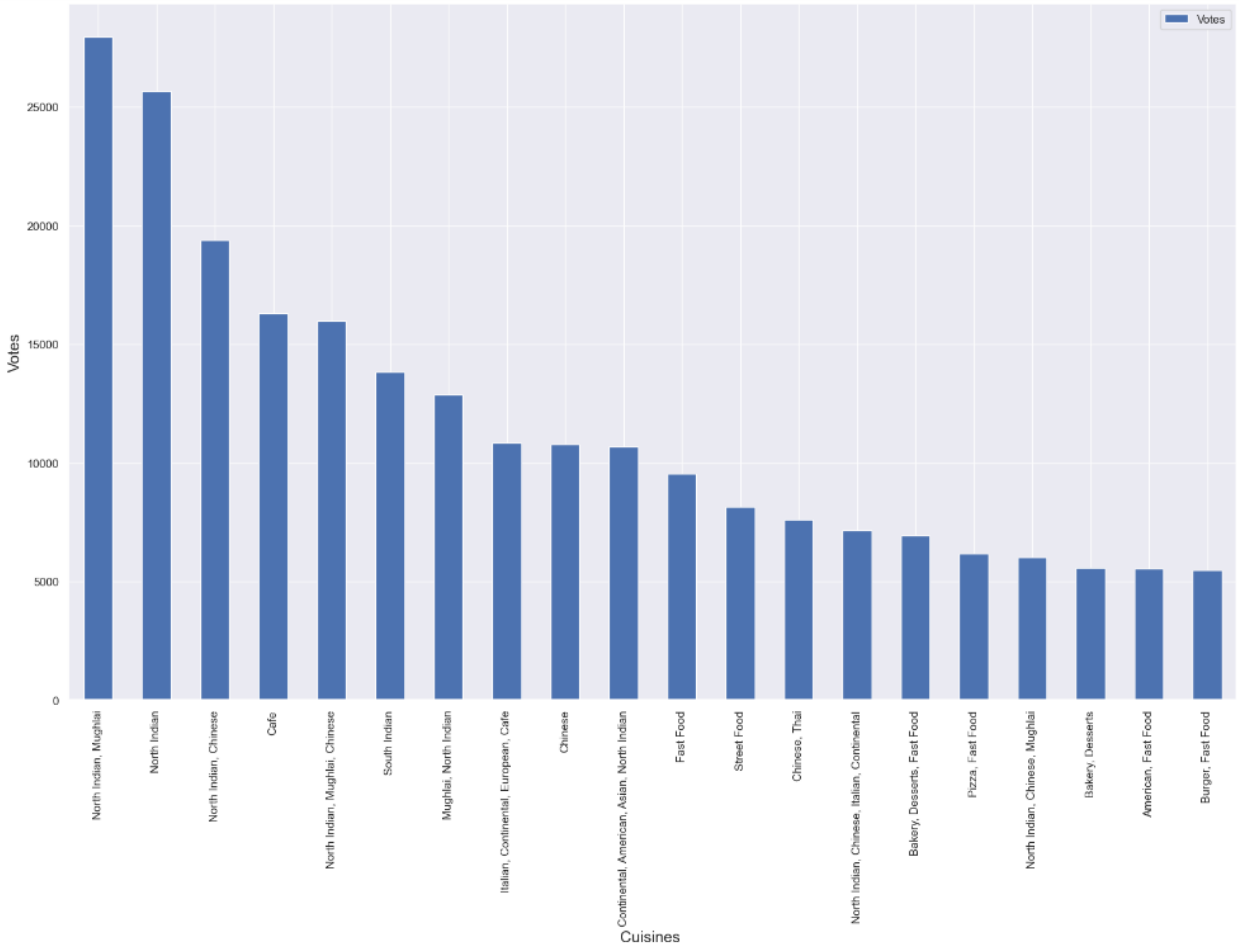


Fig. 3.31: Most served cuisines

From the above graph, we can clearly see that 'North Indian, Mughlai' cuisine is the most popular cuisine and it makes sense as well since the maximum data has restaurants listed from North India.

- **Locality by Price Range on Zomato**

In [42]:
```
df_Loc_price=(zmt1.groupby(['Locality'], as_index=False)['Price range'].mean())
df_Loc_price.columns = ['Locality', 'Price range' ]
df_Loc_price1=df_Loc_price.sort_values(['Price range'],ascending=False).head(30)
df_Loc_price1
```

Out[42]:

	Locality	Price range
198	Shangri La's - Eros hotel, Janpath	4.000000
171	Premier Inn, Shalimar Bagh	4.000000
220	The Leela Ambience Convention Hotel	4.000000
221	The Leela Palace, Chanakyapuri	4.000000
216	The Grand New Delhi, Vasant Kunj	4.000000
223	The Park, Connaught Place	4.000000
37	Crowne Plaza, Mayur Vihar Phase 1	4.000000
213	Taj Vivanta, Khan Market	4.000000
202	Sheraton New Delhi Hotel, Saket	4.000000
96	Jaypee Siddharth, Rajendra Place	4.000000
225	The Taj Mahal Hotel, Mansingh Road	4.000000
178	Radisson Blu, Paschim Vihar	4.000000
63	Garden of Five Senses, Saket	4.000000
166	Piccadily Hotel, Janakpuri	4.000000
77	Hotel City Park, Pitampura	4.000000
80	Hotel The Royal Plaza, Janpath	4.000000
126	Maidens Hotel, Civil Lines	4.000000
117	Le Meridien, Janpath	4.000000
89	JW Marriott New Delhi	4.000000
106	Kasbah, Greater Kailash (GK) 1	4.000000
215	The Claridges, Aurangzeb Road	3.857143
81	Hyatt Regency, Bhikaji Cama Place	3.833333
217	The Imperial, Janpath	3.777778
226	The Taj Palace Hotel, Chanakyapuri	3.750000
177	Radisson Blu Plaza Delhi, Mahipalpur	3.666667
183	Roseate House, Aerocity	3.666667
84	ITC Maurya, Chanakyapuri	3.666667
219	The Lalit New Delhi, Barakhamba Road	3.625000
97	Jaypee Vasant Continental, Vasant Vihar	3.600000
205	Southern Park Mall, Saket	3.500000

```
In [43]: df_Loc_price1.plot(kind = 'bar',figsize=(20,12))
         plt.ylabel('Price range',size=15)
         plt.xlabel('Index of locality',size=15)

Out[43]: Text(0.5, 0, 'Index of locality')
```

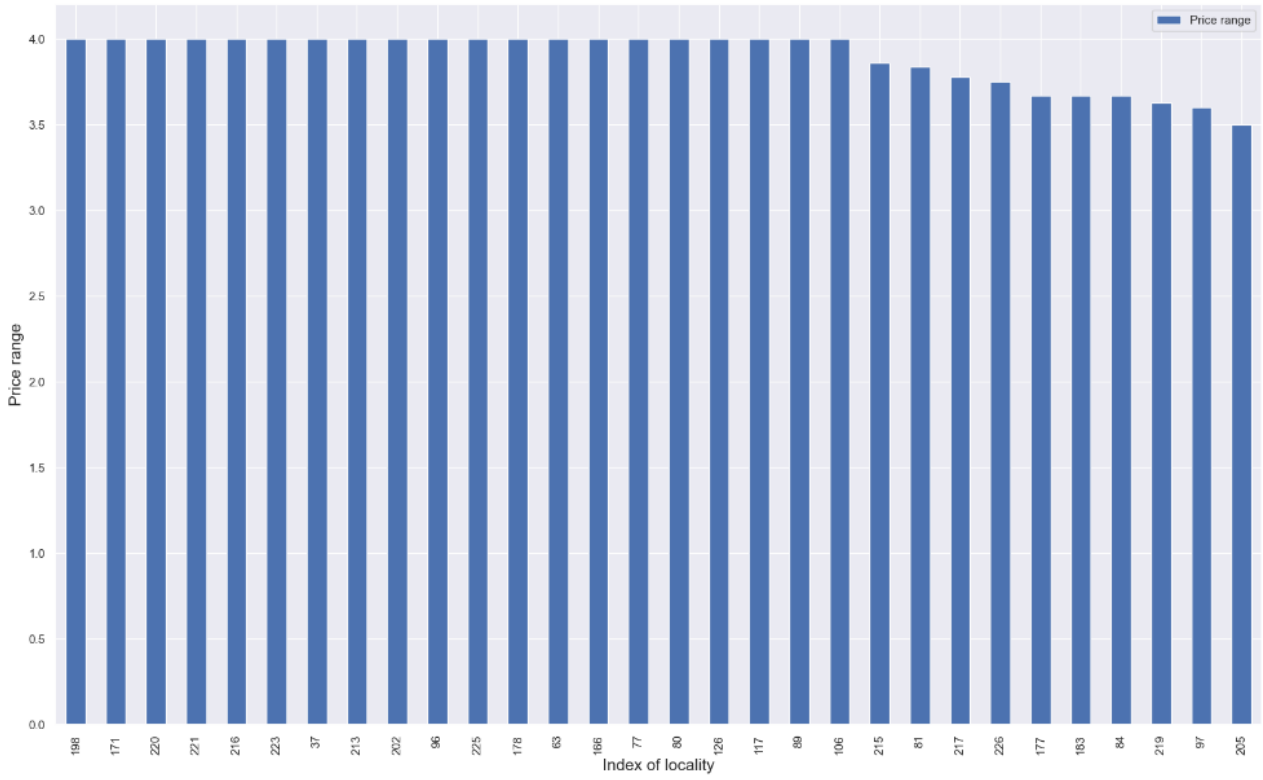


Fig. 3.32: Locality by price range

- **Visualizing Online Orders Vs Rate**

```
In [44]: plt.figure(figsize = (6,6))
         sns.boxplot(x='Has Online delivery', y='Aggregate rating',
                     data=zmt1)

Out[44]: <AxesSubplot:xlabel='Has Online delivery', ylabel='Aggregate r
         ating'>
```

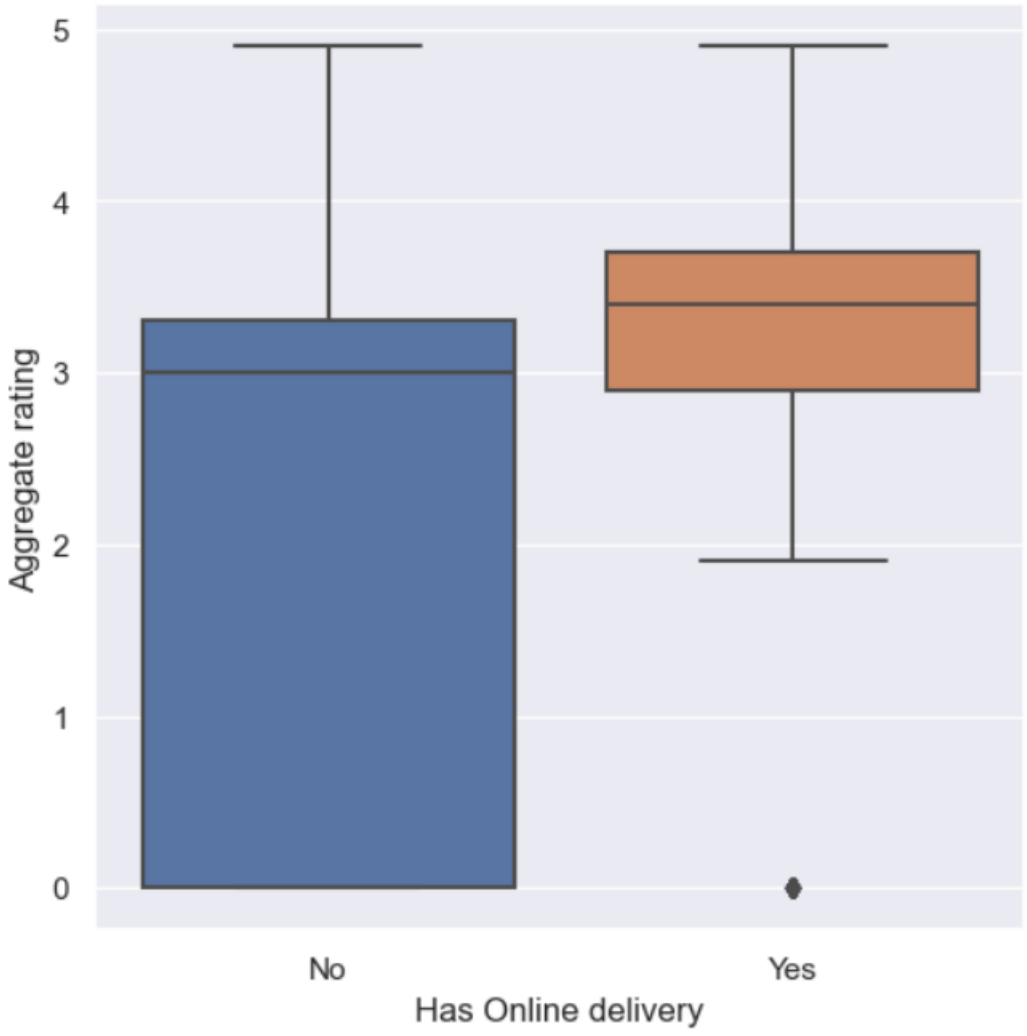


Fig. 3.33: Online order Vs Rate

Conclusion: The resturants which are having the online order facility, their maximum rate is higher than the restuarants which are not having online order facility.

- **Visualizing Table booking Vs Rate**

```
In [45]: plt.figure(figsize = (6,6))
         sns.boxplot(x='Has Table booking', y='Aggregate rating',
                     data=zmt1)

Out[45]: <AxesSubplot:xlabel='Has Table booking', ylabel='Aggregate rating'>
```

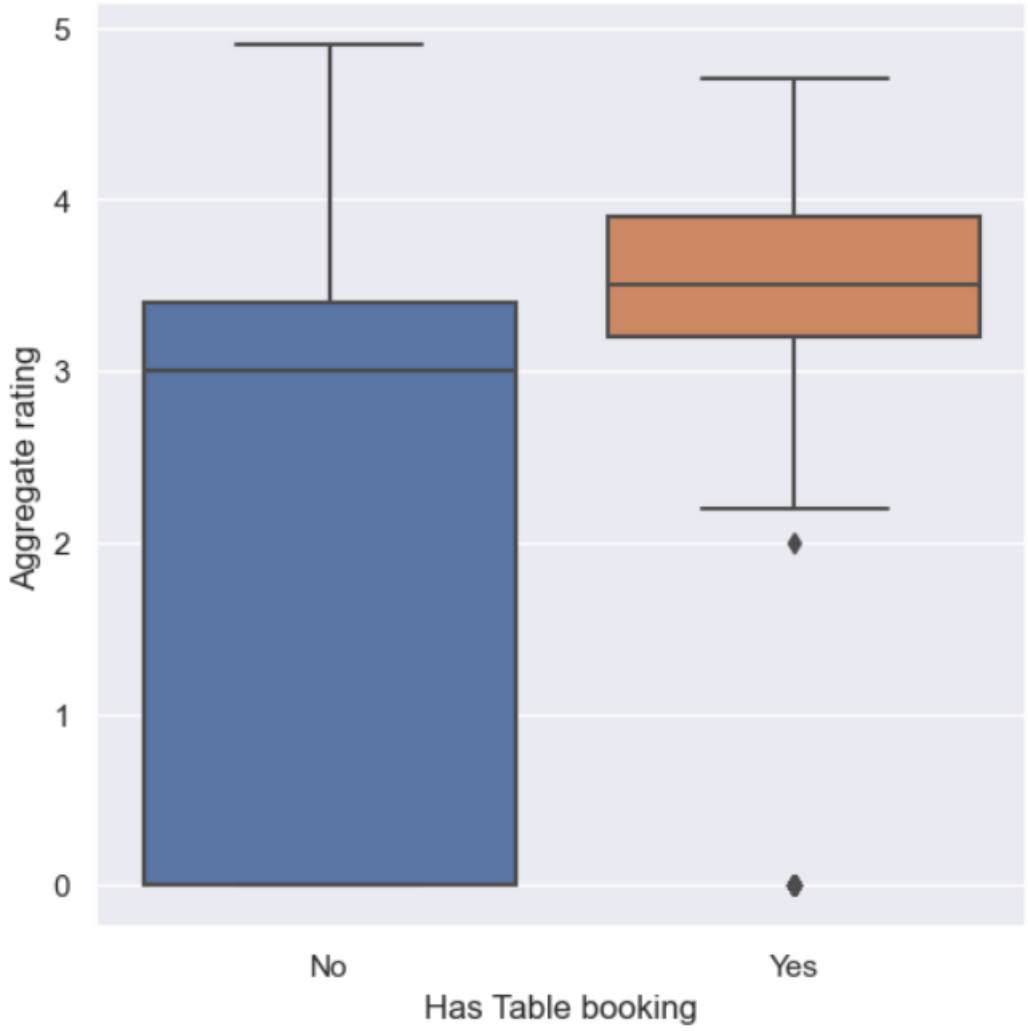


Fig. 3.34: Table booking Vs Rate

Conclusion: The resturants which are having the book table facility, their maximum rate is higher than the restuarants which are not having book table facility.

- **No of Rating text on Zomato**

```
In [47]: zmt1['Rating text'].value_counts()

Out[47]: Average      2495
         Not rated    1425
         Good         1128
         Very Good     300
         Poor           97
         Excellent      28
         Name: Rating text, dtype: int64
```

Fig. 3.35: Type of Rating Text

- **No of Rating text on Locality**

In [48]:
```
Text_Rating = zmt1.groupby(['Locality'], as_index=False).count()
[['Locality', 'Restaurant ID']]
Text_Rating.columns = ['Locality', 'No of Restaurant']
Restaurant_text_rating=zmt1.groupby(['Locality', 'Rating text'],
                     as_index=False)['Restaurant ID'].count()
Total_Restaurant_text_rating = pd.merge(Text_Rating,
                         Restaurant_text_rating, on='Locality')
Total_Restaurant_text_rating['Percentage'] =
(Total_Restaurant_text_rating['Restaurant ID']/
 Total_Restaurant_text_rating['No of Restaurant'])*100
Total_Restaurant_text_rating1=Total_Restaurant_text_rating.
head(20)
Total_Restaurant_text_rating1
```

Out[48]:

	Locality	No of Restaurant	Rating text	Restaurant ID	Percentage
0	ARSS Mall, Paschim Vihar	1	Average	1	100.000000
1	Aaya Nagar	1	Not rated	1	100.000000
2	Adchini	13	Average	7	53.846154
3	Adchini	13	Good	2	15.384615
4	Adchini	13	Poor	1	7.692308
5	Adchini	13	Very Good	3	23.076923
6	Aditya Mega Mall, Karkardooma	4	Average	3	75.000000
7	Aditya Mega Mall, Karkardooma	4	Good	1	25.000000
8	Aerocity	4	Average	2	50.000000
9	Aerocity	4	Not rated	2	50.000000
10	Aggarwal City Mall, Pitampura	3	Average	3	100.000000

11	Aggarwal City Plaza, Rohini	5	Average	3	60.000000
12	Aggarwal City Plaza, Rohini	5	Good	1	20.000000
13	Aggarwal City Plaza, Rohini	5	Poor	1	20.000000
14	Alaknanda	32	Average	19	59.375000
15	Alaknanda	32	Good	4	12.500000
16	Alaknanda	32	Not rated	8	25.000000
17	Alaknanda	32	Poor	1	3.125000
18	Ambience Mall, Vasant Kunj	12	Average	4	33.333333
19	Ambience Mall, Vasant Kunj	12	Good	7	58.333333

```
In [49]: sns.set(rc={'figure.figsize':(20,11)})
         sns.barplot('Locality', 'Percentage',
             data=Total_Restaurant_text_rating1, hue='Rating text')
         plt.xticks(rotation = 90)
         plt.xlabel('Locality', fontsize=20)
         plt.title('No of Rating text', fontsize=30)
         plt.ylabel('No of Ratings', fontsize=20)
```

```
Out[49]: Text(0, 0.5, 'No of Ratings')
```

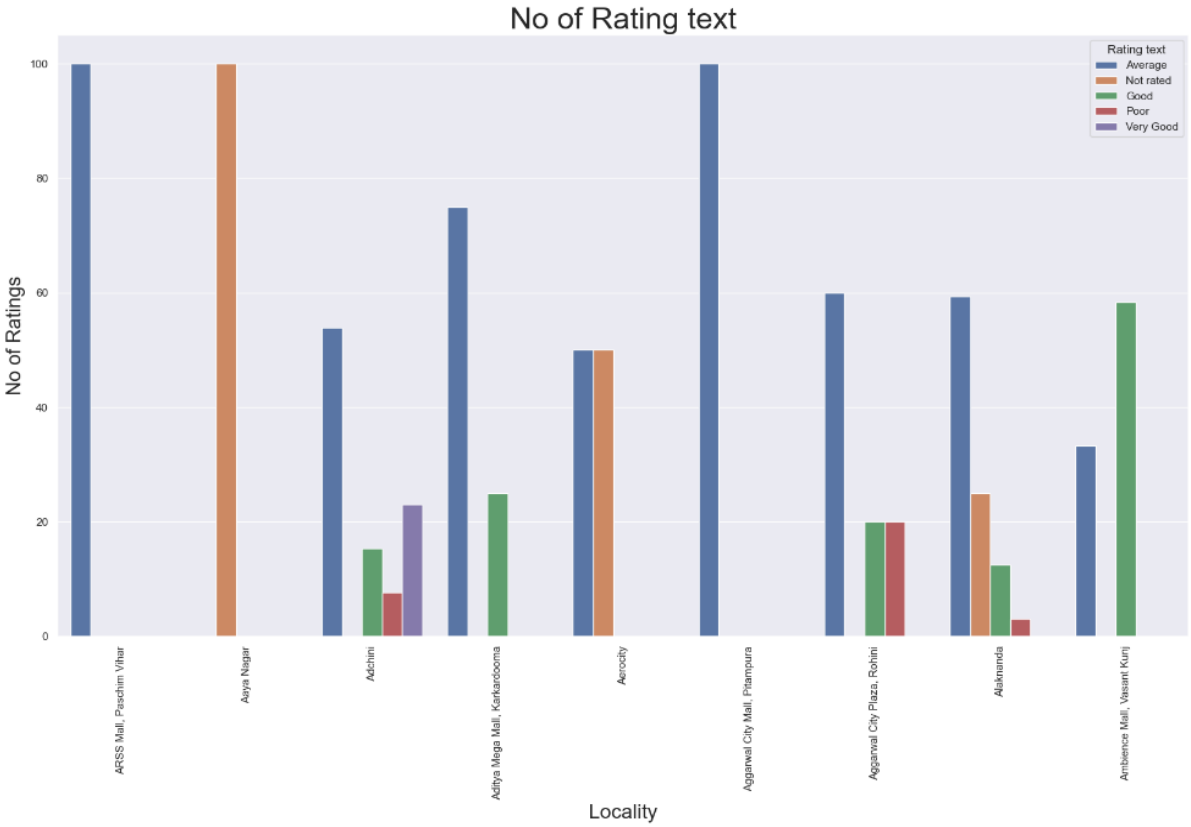
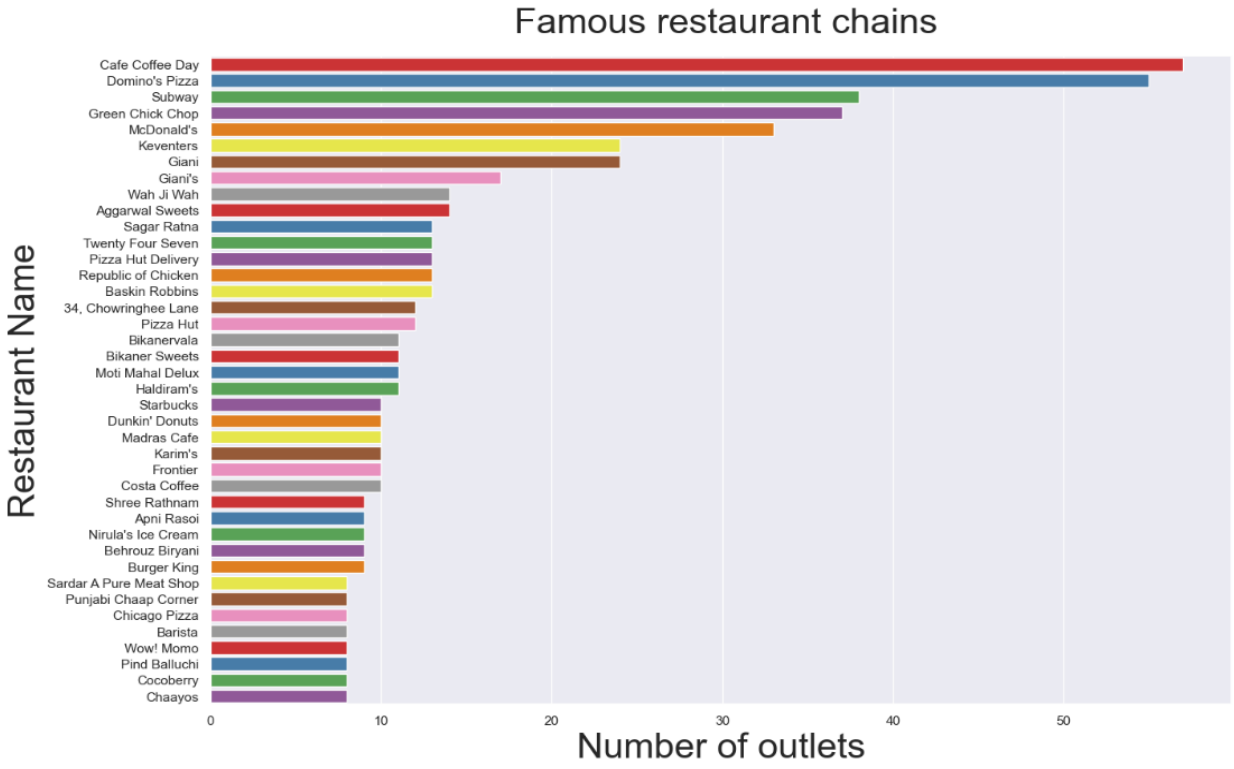


Fig. 3.36:Text rating on locality

- **Popular Restaurants**

In [50]:
```
plt.figure(figsize=(15,10))
chains=zmt1['Restaurant Name'].value_counts()[:40]
sns.barplot(x=chains,y=chains.index,palette='Set1')
plt.title(" Famous restaurant chains",size=30,pad=20)
plt.xlabel("Number of outlets",size=30)
plt.ylabel("Restaurant Name",size=30)
```

Out[50]:
```
Text(0, 0.5, 'Restaurant Name')
```



In [53]:
```
#Top 20 Restro with highest no. of votes
max_votes =zmt1.Votes.sort_values(ascending=False).head(30)
zmt1.loc[zmt1['Votes'].isin(max_votes)][['Restaurant Name','Votes']]
zmt1.loc[zmt1['Votes'].isin(max_votes)][['Restaurant Name','Votes']].
plot.bar(x='Restaurant Name', y='Votes',
figsize = (10,6), color='purple')
```

Out[53]:
```
<AxesSubplot:xlabel='Restaurant Name'>
```

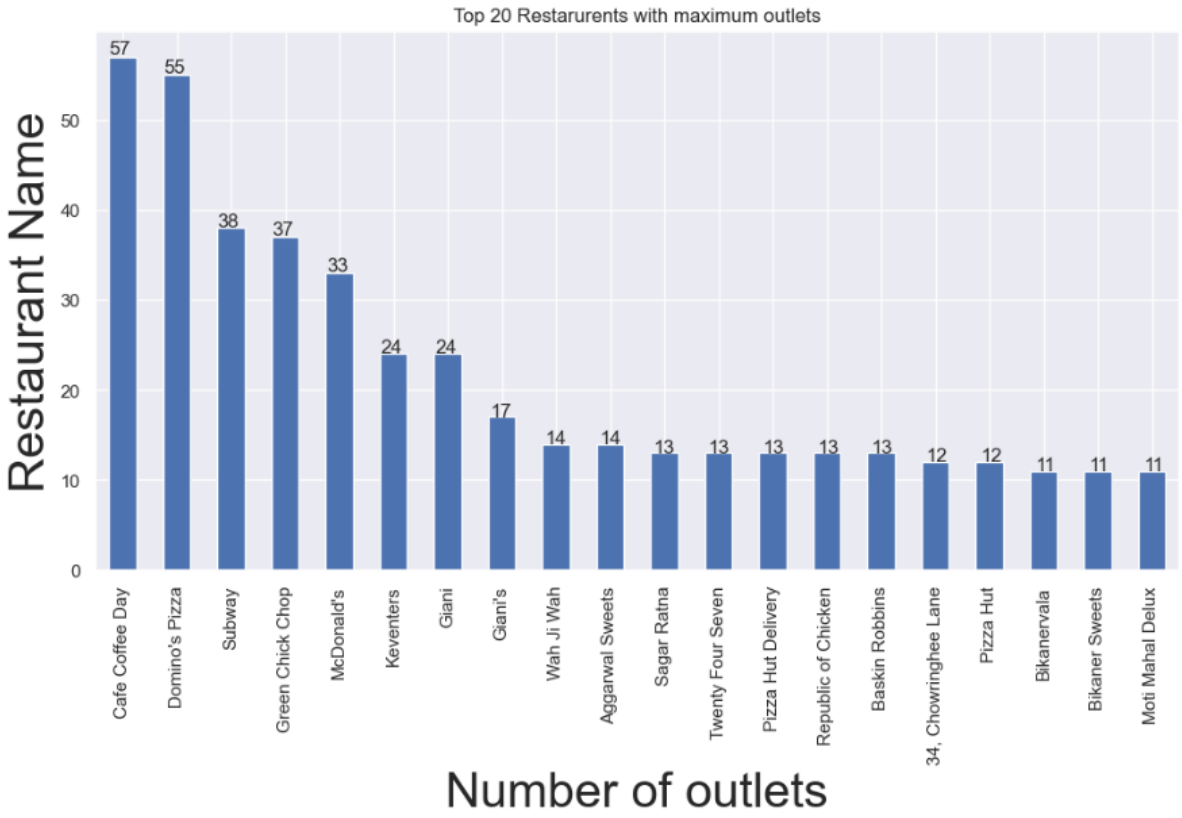


Fig. 3.37: Popular Restaurants

We can see Cafe Coffee Day, Dominos and Subway are the clear winners here.

- **Restaurants in the data with highest number of votes.**

```
In [52]: max_votes =zmt1.Votes.sort_values(ascending=False).head(30)
         zmt1.loc[zmt1['Votes'].isin(max_votes)][['Restaurant Name','Votes']]
```

Out[52]:

	Restaurant Name	Votes
3008	My Bar Lounge & Restaurant	2460
3013	Naturals Ice Cream	2620
3016	Berco's	2639
3026	Cha Bar	3206
3049	Lord of the Drinks	2589
3055	My Bar Headquarters	3010
3060	Parikrama - The Revolving Restaurant	2689
3083	The Vault Cafe	3413
3085	Warehouse Cafe	4914
3095	Barbeque Nation	3164
3110	Saravana Bhavan	5172
3119	Wenger's	3591
3316	QD's Restaurant	2724
3336	Ricos	4085
3438	Big Chill	2777
3589	The Flying Saucer Cafe	3002
3599	Andhra Bhavan	3010
3992	Out Of The Box	3697
3994	Hauz Khas Social	7931
4087	Bukhara - ITC Maurya	2826
4178	Karim's	4689
4638	Big Chill	4986
4649	Khan Chacha	2860
5007	The All American Diner	3495
5026	Lodi - The Garden Restaurant	2549
6144	Gulati	4373
6712	Pirates of Grill	2514
6848	Rajinder Da Dhaba	3530
7033	Big Yellow Door	3311
7863	Big Yellow Door	3986

In [53]:
```
#Top 20 Restro with highest no. of votes
max_votes =zmt1.Votes.sort_values(ascending=False).head(30)
zmt1.loc[zmt1['Votes'].isin(max_votes)][['Restaurant Name',
                                         'Votes']]
zmt1.loc[zmt1['Votes'].isin(max_votes)][['Restaurant Name',
                                         'Votes']].
plot.bar(x='Restaurant Name', y='Votes',
figsize = (10,6), color='purple')
```

Out[53]:
```
<AxesSubplot:xlabel='Restaurant Name'>
```

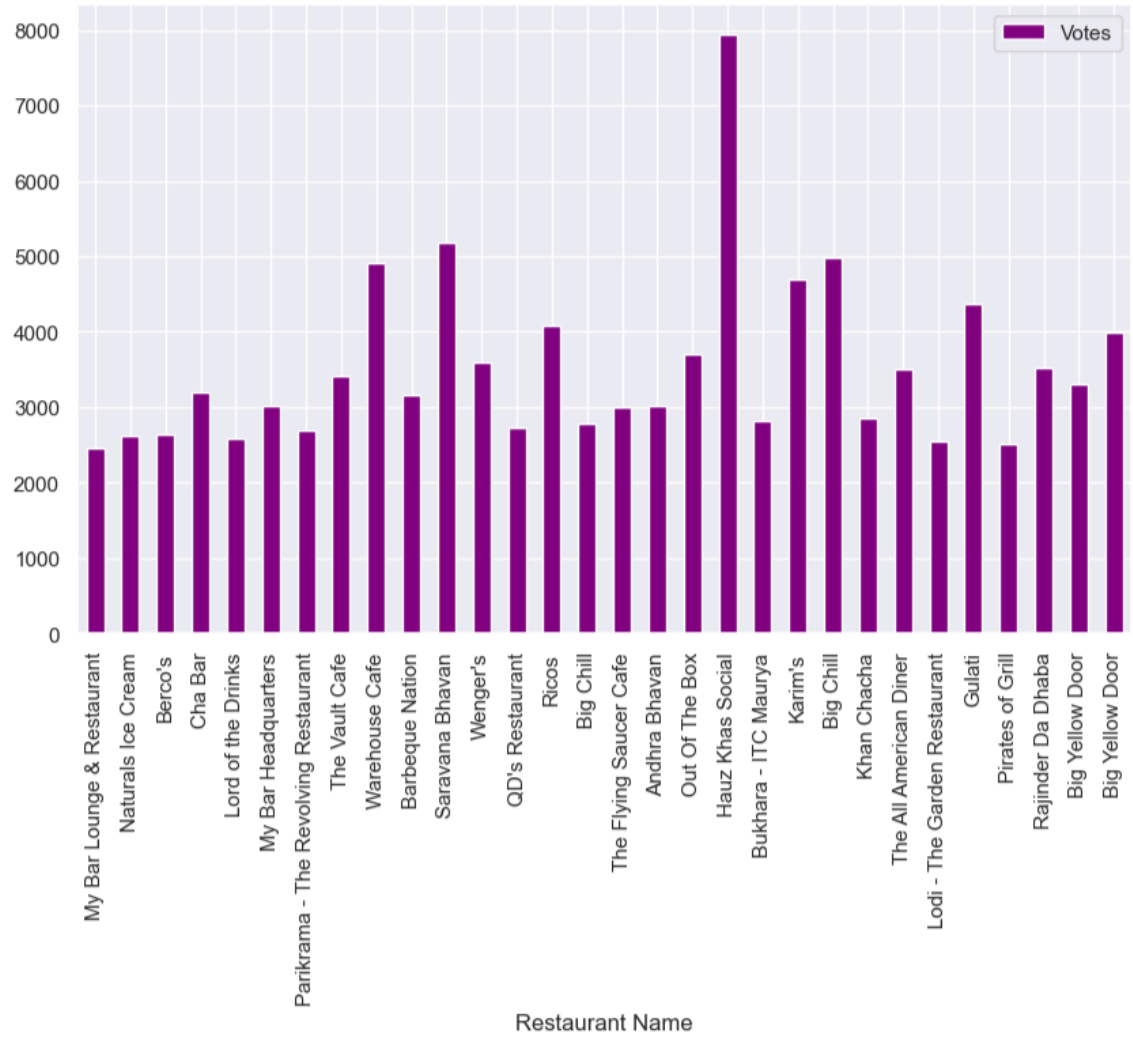


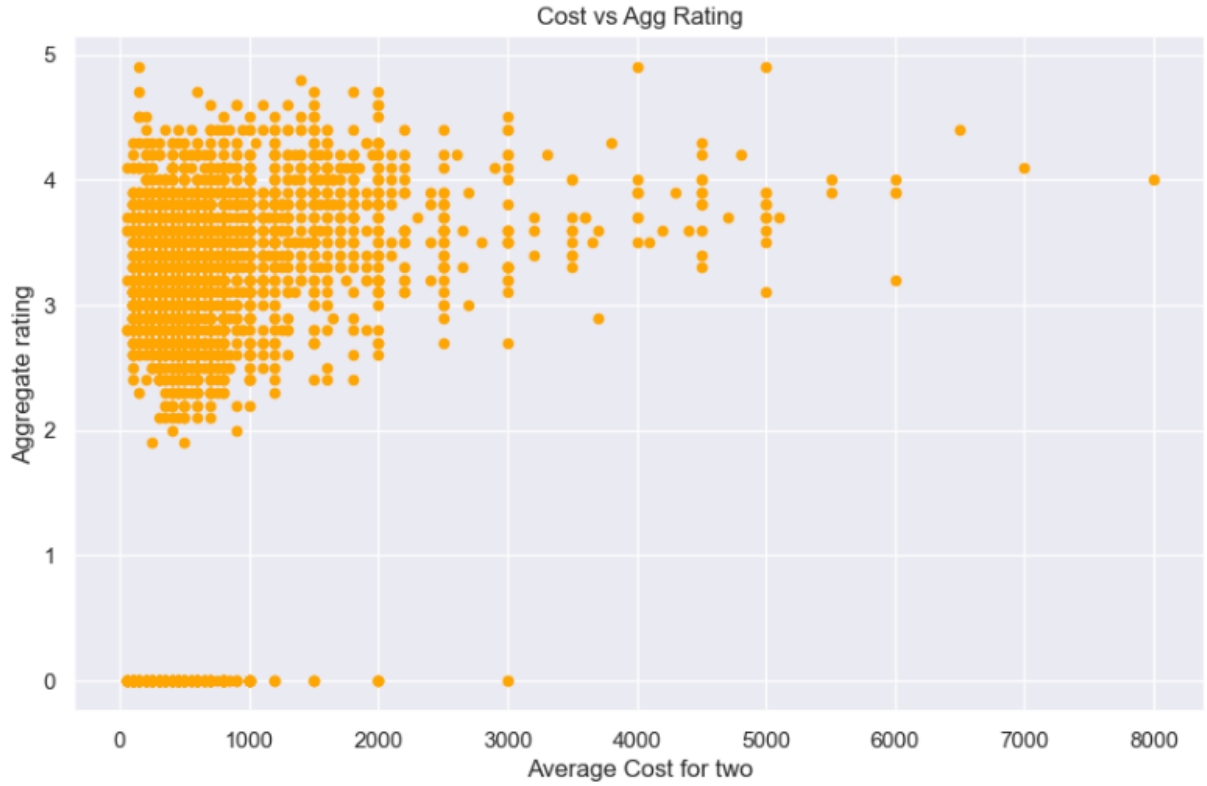
Fig. 3.38: Restaurants with highest vote

We can see in the above graph that the restaurants with maximum number of outlets are not the one which have highest number of votes. The above list is totally different that our list of top 10 restaurants with maximum number of outlets.

- **Relation between average cost for two and aggregate rating of restaurants.**

```
In [54]: #Is there any relation between average cost for two and aggregate rating of
         restaurants
         zmt1.plot.scatter(x='Average Cost for two',y='Aggregate rating',
                           figsize=(10,6), color='orange', title="Cost vs Agg Rating")

Out[54]: <AxesSubplot:title={'center':'Cost vs Agg Rating'}, xlabel='Average Cost for tw
         o', ylabel='Aggregate rating'>
```



From the above graph, we can see that most of the data is clustered around cost upto 2000 and rating values from 2 to 4.5 approximately. There are few restaurants with cost range between 2500 to 6000.

There are some outliers in the data where cost is listed as 0 because it might not be captured for those restaurants. A more concise and detailed view of above data can be via hex plot as shown below.

In [55]:
```
#Better view of relation between average cost for two and aggregate rating of restaurants
sns.jointplot(x='Average Cost for two',y='Aggregate rating',kind ='hex',gridsize=18,
              data =zmt1,color='green')
```

Out[55]:
```
<seaborn.axisgrid.JointGrid at 0x26c30ce28b0>
```

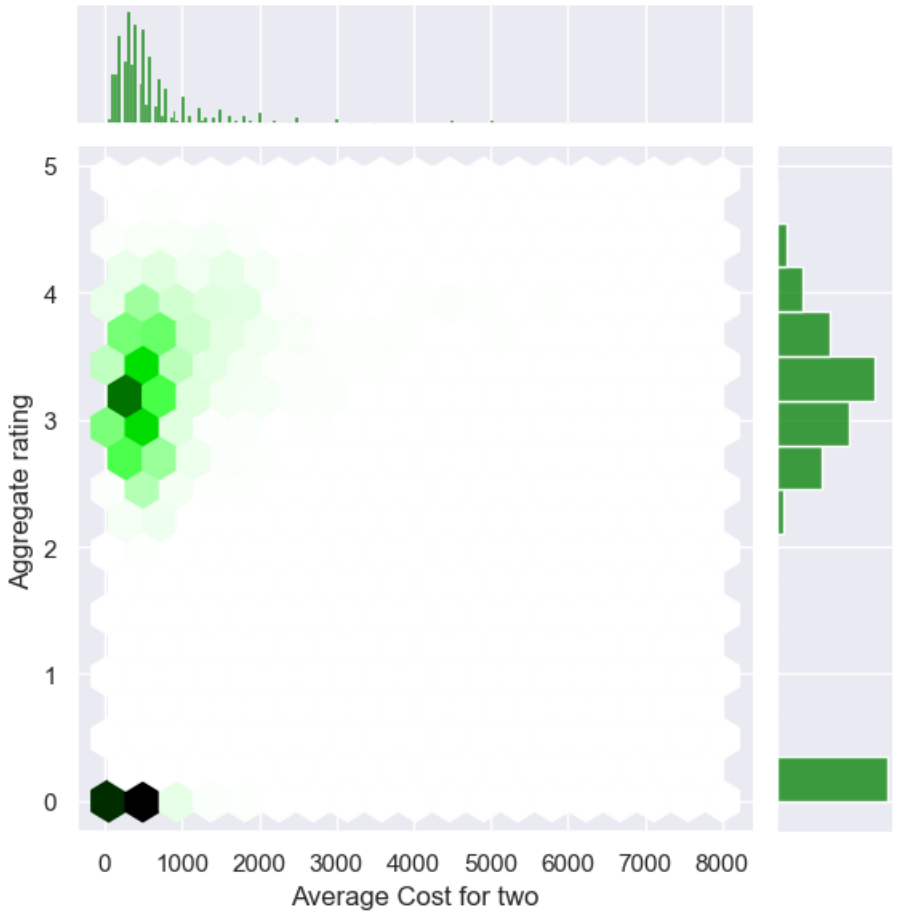


Fig. 3.39:Relation b/w rating and cost

In above graph, we can see more clearly that the maximum number of rating values are around 3 to 3.5 and the 'Avg cost for two, for maximum data is also up to 1000.
From the above graph, we can clearly see that 'North Indian' cuisine is the most popular cuisine and it makes sense as well since the maximum data has restaurants listed from North India.

- **Analyze top 10 cuisines with 'price range' and 'agg rating'**

In [56]:
```
#ANalysis of top 10 cuisines with price range and agg rating
top10cuisines_list=["North Indian, Mughlai","North Indian","North Indian, Chinese","Cafe",
                    "North Indian, Mughlai, Chinese","South Indian","Mughlai, North Indian"
,"Italian, Continental, European, Cafe","Chinese","Continental, American, Asian, North Indian" ]
zmt_cuisines = zmt.loc[zmt['Cuisines'].isin(top10cuisines_list)]
zmt_cuisines_data = zmt_cuisines[['Average Cost for two', 'Price range', 'Aggregate rating',
                                  'Cuisines']]

fig, axx =plt.subplots(figsize=(16,8))
sns.barplot(x='Cuisines', y='Aggregate rating', hue='Price range', data=zmt_cuisines_data,
            palette="Set1")
axx.set_title("Analysis of Top10 Cuisines with price range and Agg. rating ")
plt.xticks(rotation = 90)
```

Out[56]:
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'Chinese'),
  Text(1, 0, 'Cafe'),
  Text(2, 0, 'South Indian'),
  Text(3, 0, 'North Indian, Mughlai'),
  Text(4, 0, 'North Indian'),
  Text(5, 0, 'North Indian, Chinese'),
  Text(6, 0, 'North Indian, Mughlai, Chinese'),
  Text(7, 0, 'Mughlai, North Indian'),
  Text(8, 0, 'Continental, American, Asian, North Indian'),
  Text(9, 0, 'Italian, Continental, European, Cafe')])
```

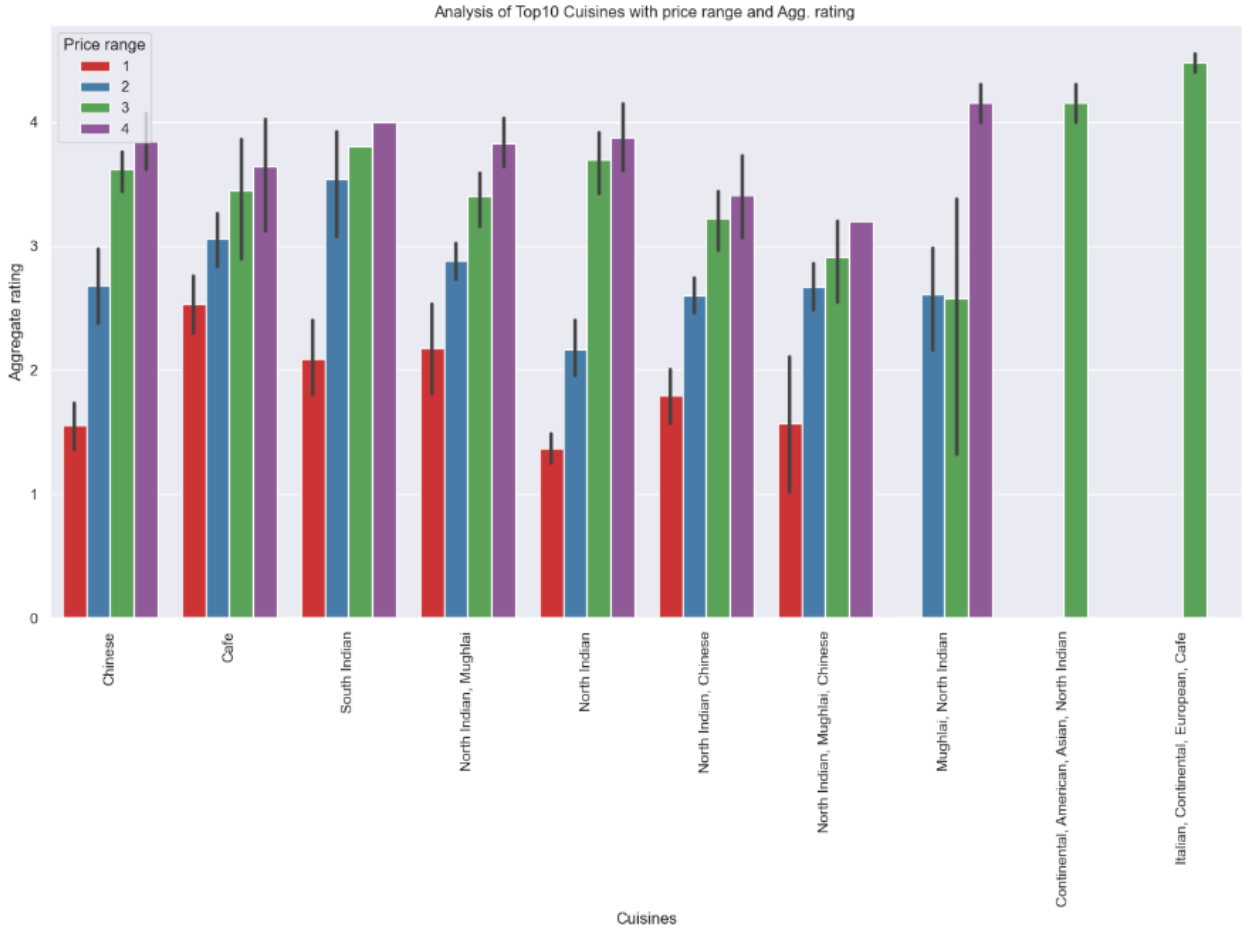


Fig. 3.40: top 10 cuisines with 'price range' and 'agg rating'

From the above graph, we can clearly see that among the top 10 cuisines, price range 4 has got the highest rating!! May be we can assume that the high end restaurants (with higher price range) serves the best and hence the higher 'Agg rating' in all cuisines.

Similar is the case with price range 1 which has lowest 'Agg rating' amoung all cuisines

In all the cuisines, we can see the trend, higher the price range, better is the 'Agg rating' .

- **Percentage rating of restaurants in our top 10 cities**

In [60]:
```
#Restaurant Percentage wise rating in top 5 cities
top10_indian_cities = ['Connaught Place','Rajouri Garden','Shahdara',  'Defence Colony' ,
                       'Pitampura',"Malviya Nagar","Mayur Vihar Phase 1","Rajinder Nagar","Safdarjung","Satyaniketan"]
zmt_rate = zmt1.loc[zmt1['Locality'].isin(top10_indian_cities)]

#Find total number of restaurants
total_restro = zmt_rate.groupby(['Locality'], as_index=False).count()[['Locality','Restaurant ID']]
total_restro.columns=['Locality','Total Restaurants']

#Find total rating count of each type
top10rest = zmt_rate.groupby(['Locality','Rating text'], as_index=False)[['Restaurant Name']].count()
top10rest.columns=['Locality','Rating text', 'Total Ratings']

#Merge both the dataframes and calculate percentage
top10restro_rating_percent = pd.merge(total_restro, top10rest, on='Locality')
top10restro_rating_percent['Percentage']= (top10restro_rating_percent['Total Ratings']/top10restro_rating_percent
                                           ['Total Restaurants'])*100

top10restro_rating_percent
```

Out[60]:

	Locality	Total Restaurants	Rating text	Total Ratings	Percentage
0	Connaught Place	122	Average	13	10.655738
1	Connaught Place	122	Excellent	3	2.459016
2	Connaught Place	122	Good	74	60.655738
3	Connaught Place	122	Not rated	3	2.459016
4	Connaught Place	122	Very Good	29	23.770492
5	Defence Colony	86	Average	35	40.697674
6	Defence Colony	86	Good	30	34.883721
7	Defence Colony	86	Not rated	7	8.139535
8	Defence Colony	86	Poor	1	1.162791
9	Defence Colony	86	Very Good	13	15.116279
10	Malviya Nagar	84	Average	35	41.666667
11	Malviya Nagar	84	Good	35	41.666667
12	Malviya Nagar	84	Poor	5	5.952381
13	Malviya Nagar	84	Very Good	9	10.714286
14	Mayur Vihar Phase 1	84	Average	48	57.142857
15	Mayur Vihar Phase 1	84	Good	6	7.142857
16	Mayur Vihar Phase 1	84	Not rated	27	32.142857
17	Mayur Vihar Phase 1	84	Poor	3	3.571429
18	Pitampura	85	Average	55	64.705882
19	Pitampura	85	Good	19	22.352941
20	Pitampura	85	Not rated	8	9.411765
21	Pitampura	85	Poor	3	3.529412
22	Rajinder Nagar	81	Average	63	77.777778
23	Rajinder Nagar	81	Good	10	12.345679

24	Rajinder Nagar	81	Not rated	3	3.703704
25	Rajinder Nagar	81	Poor	1	1.234568
26	Rajinder Nagar	81	Very Good	4	4.938272
27	Rajouri Garden	99	Average	34	34.343434
28	Rajouri Garden	99	Excellent	5	5.050505
29	Rajouri Garden	99	Good	39	39.393939
30	Rajouri Garden	99	Not rated	1	1.010101
31	Rajouri Garden	99	Poor	1	1.010101
32	Rajouri Garden	99	Very Good	19	19.191919
33	Safdarjung	80	Average	46	57.500000
34	Safdarjung	80	Good	14	17.500000
35	Safdarjung	80	Not rated	11	13.750000
36	Safdarjung	80	Poor	2	2.500000
37	Safdarjung	80	Very Good	7	8.750000
38	Satyaniketan	79	Average	29	36.708861
39	Satyaniketan	79	Excellent	1	1.265823
40	Satyaniketan	79	Good	32	40.506329
41	Satyaniketan	79	Not rated	1	1.265823
42	Satyaniketan	79	Poor	2	2.531646
43	Satyaniketan	79	Very Good	14	17.721519
44	Shahdara	87	Average	40	45.977011
45	Shahdara	87	Good	1	1.149425
46	Shahdara	87	Not rated	46	52.873563

In [61]:
```
#Plot Rating percentage of restaurants in top 5 cities
fig, axx =plt.subplots(figsize=(12,6))
axx.set_title("Percentage Rating of Restaurants in Top 10 Cities")
sns.barplot(x='Locality', y='Percentage',hue='Rating text', data=top10restro_rating_percent,
            palette='Set3')
plt.xticks(rotation = 90)
```

Out[61]:
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'Connaught Place'),
  Text(1, 0, 'Defence Colony'),
  Text(2, 0, 'Malviya Nagar'),
  Text(3, 0, 'Mayur Vihar Phase 1'),
  Text(4, 0, 'Pitampura'),
  Text(5, 0, 'Rajinder Nagar'),
  Text(6, 0, 'Rajouri Garden'),
  Text(7, 0, 'Safdarjung'),
  Text(8, 0, 'Satyaniketan'),
  Text(9, 0, 'Shahdara')])
```

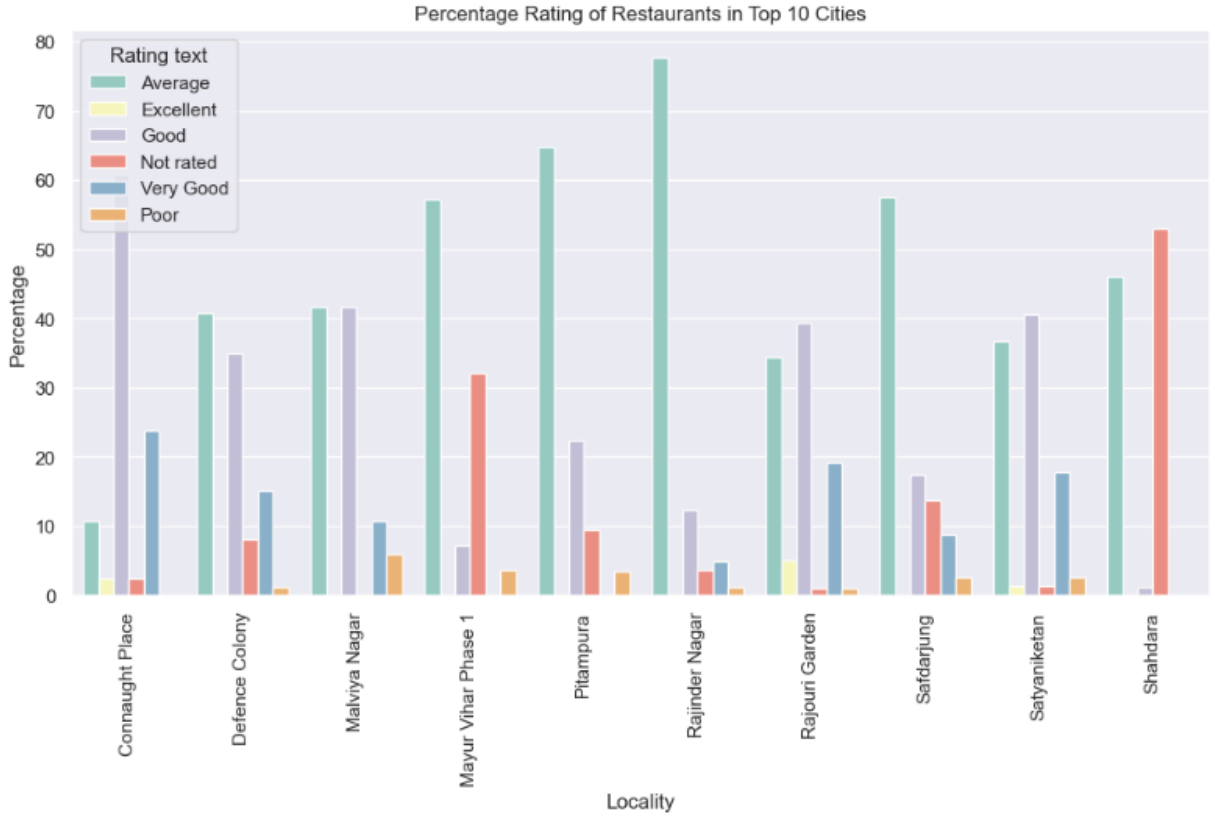


Fig. 3.41: Percentage rating of restaurants in top 10 cities

From the above graph, we can see that maximum number of restaurants in top 10 cities have the rating 'Average'

Excellent rating is almost negligible.

CHAPTER 4
CONCLUSION

5.1 Discussion

After comprehensively analyzing Zomato's restaurant data, we can conclude that there are several factors that influence customer preferences when choosing a restaurant. Some of the most important factors are location, food, cost, atmosphere and customer service.

We also found a high correlation between review opinions and restaurant ratings. This suggests that customers tend to rate restaurants based on their overall experience. Additionally, machine learning algorithms can be used to predict restaurant ratings with reasonable accuracy.

Overall, analyzing Zomato's restaurant data provides valuable insight into customer preferences, which restaurant owners and managers can use to improve their business strategy. By focusing on the key factors that influence customer decisions, you can attract more customers and improve your ratings and reviews on your platform. Additionally, this analysis helps customers make more informed decisions when choosing restaurants based on their preferences.

5.2 Future Work

Future work on our project are as follows:

- Increase the accuracy to give more comprehensive result on the prediction.
- Make the machine learning model to predict the success of the new restaurant in given area.
- Make a web version of the project and easy to use.

REFERENCES

Downloading data set:

https://www.kaggle.com/code/bansodesandeep/random-forrest-extratree-regressor-pickle/notebook

Loading and analyzing

https://pandas.pydata.org/

https://numpy.org/

Pre-processing

https://towardsdatascience.com/data-preprocessing-in-python-b52b652e37d5

https://neptune.ai/blog/data-preprocessing-guide

https://cloud.google.com/architecture/data-preprocessing-for-ml-with-tf-transform-pt1

https://www.analyticsvidhya.com/blog/2021/08/data-preprocessing-in-data-mining-a-hands-on-guide/

Data Visualization

https://matplotlib.org/tutorials/index.html

https://python-graph-gallery.com/seaborn/