Exploratory Data Analysis of Zomato's Restaurant Dataset

About Dataset

- Zomato API Analysis is one of the most useful analysis for foodies who want to taste the best cuisines of every part of the world which lies in their budget. This analysis is also for those who want to find value-for-money restaurants in various parts of the country for cuisines. Additionally, this analysis caters to the needs of people who are striving to get the best cuisine of the country and which locality of that country serves that cuisine with the maximum number of restaurants

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from plotly.offline import init_notebook_mode, plot, iplot
import plotly.graph_objs as go
%matplotlib inline
```

In [2]:
```
# df = pd.read_csv('../Python/DataSet/Zomatodataset/zomato.csv')
# shows error of utf-8 so we do encoding
```

In [3]:
```
df = pd.read_csv('../Python/DataSet/Zomatodataset/zomato.csv',encoding= 'latin1')
df.head()
```

Out[3]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency	Has Table booking
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	...	Botswana Pula(P)	Yes
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	...	Botswana Pula(P)	Yes
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	...	Botswana Pula(P)	Yes
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	...	Botswana Pula(P)	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	...	Botswana Pula(P)	Yes

5 rows × 21 columns

Dataset Details

- Restaurant Id: Unique id of every restaurant across various cities of the world
- Restaurant Name: Name of the restaurant
- Country Code: Country in which restaurant is located
- City: City in which restaurant is located
- Address: Address of the restaurant
- Locality: Location in the city
- Locality Verbose: Detailed description of the locality -_Longitude: Longitude coordinate of the restaurant's location
- Latitude: Latitude coordinate of the restaurant's location
- Cuisines: Cuisines offered by the restaurant
- Average Cost for two: Cost for two people in different currencies
- Currency: Currency of the country

- Has Table booking: yes/no
- Has Online delivery: yes/ no
- Is delivering: yes/ no
- Switch to order menu: yes/no
- Price range: range of price of food
- Aggregate Rating: Average rating out of 5
- Rating color: depending upon the average rating color
- Rating text: text on the basis of rating of rating
- Votes: Number of ratings casted by people

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address',
       'Locality', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines',
       'Average Cost for two', 'Currency', 'Has Table booking',
       'Has Online delivery', 'Is delivering now', 'Switch to order menu',
       'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
       'Votes'],
      dtype='object')
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 21 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  
dtypes: float64(3), int64(5), object(13)
memory usage: 1.5+ MB
```

In [6]:
```
df.describe()
```

Out[6]:

	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

In [7]:
```
df.isnull().sum().sum()
```

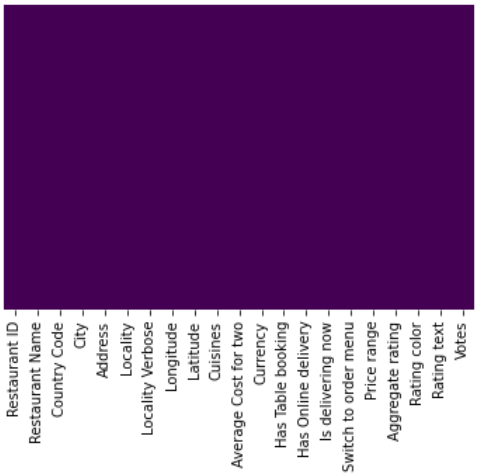
Out[7]:
```
9
```

finding null val function

In [8]:
```
[features for features in df.columns if df[features].isnull().sum()>0]
```

Out[8]:
```
['Cuisines']
```

In [9]:
```
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
plt.show()
```



In [10]:
```
df_country = pd.read_excel('../Python/DataSet/Zomatodataset/Country-Code.xlsx')
df_country
```

Out[10]:

	Country Code	Country
0	1	India
1	14	Australia
2	30	Brazil
3	37	Canada
4	94	Indonesia
5	148	New Zealand
6	162	Phillipines
7	166	Qatar
8	184	Singapore
9	189	South Africa
10	191	Sri Lanka
11	208	Turkey
12	214	UAE
13	215	United Kingdom
14	216	United States

In [11]:
```
final_df = pd.merge(df,df_country,on = 'Country Code',how = 'left')
```

In [12]:
```
final_df.head(2)
```

Out[12]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Has Table booking	Has Online delivery	Is delivering now
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 rows × 22 columns

In [13]:
```
final_df.columns.tolist()
```

Out[13]:
```
['Restaurant ID',
 'Restaurant Name',
 'Country Code',
 'City',
 'Address',
 'Locality',
 'Locality Verbose',
 'Longitude',
 'Latitude',
 'Cuisines',
 'Average Cost for two',
 'Currency',
 'Has Table booking',
 'Has Online delivery',
 'Is delivering now',
 'Switch to order menu',
 'Price range',
 'Aggregate rating',
 'Rating color',
 'Rating text',
 'Votes',
 'Country']
```

In [14]:
```
# to check datatypes
df.dtypes
```

Out[14]:
```
Restaurant ID             int64
Restaurant Name          object
Country Code              int64
City                     object
Address                  object
Locality                 object
Locality Verbose         object
Longitude               float64
Latitude                float64
Cuisines                 object
Average Cost for two      int64
Currency                 object
Has Table booking        object
Has Online delivery      object
Is delivering now        object
Switch to order menu     object
Price range               int64
Aggregate rating        float64
Rating color              object
Rating text               object
Votes                     int64
dtype: object
```

In [15]:
```
final_df.Country.value_counts()
```

Out[15]:
```
India             8652
United States      434
United Kingdom      80
Brazil              60
UAE                 60
South Africa        60
New Zealand         40
Turkey              34
Australia           24
Phillipines         22
Indonesia           21
Singapore           20
Qatar               20
Sri Lanka           20
Canada               4
Name: Country, dtype: int64
```

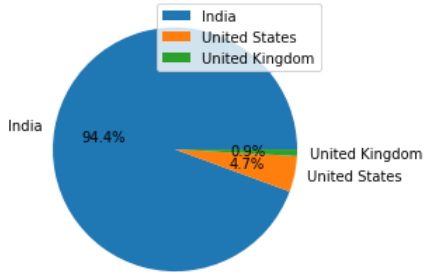
In [16]:
```
Country_names= final_df.Country.value_counts().index
Country_names
```

Out[16]:
```
Index(['India', 'United States', 'United Kingdom', 'Brazil', 'UAE',
       'South Africa', 'New Zealand', 'Turkey', 'Australia', 'Phillipines',
       'Indonesia', 'Singapore', 'Qatar', 'Sri Lanka', 'Canada'],
      dtype='object')
```

In [17]:
```
Country_val= final_df.Country.value_counts().values
```

Top 3 countries that uses zomato

In [18]:
```
plt.pie(x= Country_val[:3], labels= Country_names[:3],autopct='%1.1f%%')
plt.legend()
plt.show()
```



Observation :

- zomato maximum records / transaction are from INDIA and then USA and UK

In [19]:
```
df.describe(include= 'all')
```

Out[19]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency	T bool
count	9.551000e+03	9551	9551.000000	9551	9551	9551	9551	9551.000000	9551.000000	9542	...	9551	9
unique	NaN	7446	NaN	141	8918	1208	1265	NaN	NaN	1825	...	12	
top	NaN	Cafe Coffee Day	NaN	New Delhi	Dilli Haat, INA, New Delhi	Connaught Place	Connaught Place, New Delhi	NaN	NaN	North Indian	...	Indian Rupees(Rs.)	
freq	NaN	83	NaN	5473	11	122	122	NaN	NaN	936	...	8652	8
mean	9.051128e+06	NaN	18.365616	NaN	NaN	NaN	NaN	64.126574	25.854381	NaN	...	NaN	
std	8.791521e+06	NaN	56.750546	NaN	NaN	NaN	NaN	41.467058	11.007935	NaN	...	NaN	
min	5.300000e+01	NaN	1.000000	NaN	NaN	NaN	NaN	-157.948486	-41.330428	NaN	...	NaN	
25%	3.019625e+05	NaN	1.000000	NaN	NaN	NaN	NaN	77.081343	28.478713	NaN	...	NaN	
50%	6.004089e+06	NaN	1.000000	NaN	NaN	NaN	NaN	77.191964	28.570469	NaN	...	NaN	
75%	1.835229e+07	NaN	1.000000	NaN	NaN	NaN	NaN	77.282006	28.642758	NaN	...	NaN	
max	1.850065e+07	NaN	216.000000	NaN	NaN	NaN	NaN	174.832089	55.976980	NaN	...	NaN	

11 rows × 21 columns

In [20]:
```
ratings = final_df.groupby(['Aggregate rating','Rating color','Rating text']).size(). \
    reset_index().rename(columns={0:'Rating Count'})
```

In [21]:
```
ratings
```

Out[21]:

	Aggregate rating	Rating color	Rating text	Rating Count
0	0.0	White	Not rated	2148
1	1.8	Red	Poor	1
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15
5	2.2	Red	Poor	27
6	2.3	Red	Poor	47
7	2.4	Red	Poor	87
8	2.5	Orange	Average	110
9	2.6	Orange	Average	191
10	2.7	Orange	Average	250
11	2.8	Orange	Average	315
12	2.9	Orange	Average	381
13	3.0	Orange	Average	468
14	3.1	Orange	Average	519
15	3.2	Orange	Average	522
16	3.3	Orange	Average	483
17	3.4	Orange	Average	498
18	3.5	Yellow	Good	480
19	3.6	Yellow	Good	458
20	3.7	Yellow	Good	427
21	3.8	Yellow	Good	400
22	3.9	Yellow	Good	335
23	4.0	Green	Very Good	266
24	4.1	Green	Very Good	274
25	4.2	Green	Very Good	221
26	4.3	Green	Very Good	174
27	4.4	Green	Very Good	144
28	4.5	Dark Green	Excellent	95
29	4.6	Dark Green	Excellent	78
30	4.7	Dark Green	Excellent	42
31	4.8	Dark Green	Excellent	25
32	4.9	Dark Green	Excellent	61

OBSERVATION

- WHEN THE RATING IS BETWEEN 4.5 TO 4.9 --> EXCELLENT
- WHEN THE RATING IS BETWEEN 4.0 TO 4.4 --> VERY GOOD
- WHEN THE RATING IS BETWEEN 3.5 TO 3.9 --> GOOD
- WHEN THE RATING IS BETWEEN 3.0 TO 2.9 --> AVERAGE
- WHEN THE RATING IS BETWEEN 2.5 TO 3.4 --> AVERAGE
- WHEN THE RATING IS BETWEEN 2.0 TO 2.4 --> POOR

In [22]:
```
matplotlib.rcParams['figure.figsize']=(12,6)
sns.barplot(x='Aggregate rating',y='Rating Count',data = ratings)
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [22], in <cell line: 1>()
----> 1 matplotlib.rcParams['figure.figsize']=(12,6)
      2 sns.barplot(x='Aggregate rating',y='Rating Count',data = ratings)
      3 plt.show()

NameError: name 'matplotlib' is not defined
```

In [ ]:
```
matplotlib.rcParams['figure.figsize']=(12,6)
sns.barplot(x='Aggregate rating',y='Rating Count',hue = "Rating color",data = ratings,palette = ['white','red','orange
plt.show()
```

In [ ]:
```
sns.countplot(x= 'Rating color',data = ratings,palette= ['blue','red','orange','yellow'])
plt.show()
```

In [ ]:
```
final_df.head(2)
```

In [ ]:
```
final_df[final_df['Rating color']=='White'].groupby('Country').size()
```

In [ ]:
```
final_df[final_df['Aggregate rating']==0.0].groupby('Country').size()
```

find out which currency is used by which country?

In [ ]:
```
final_df[['Country','Currency']].groupby(['Country','Currency']).size().reset_index()
```

which country do have online delivery options?

In [ ]:
```
final_df[final_df['Has Online delivery']=='Yes'].Country.value_counts()
```

In [ ]:
```
city_values= final_df.City.value_counts().values
city_index= final_df.City.value_counts().index
```

In [ ]:
```
plt.pie(city_values[:5], labels = city_index[:5],autopct ='%1.2f%%')
```

find top 10 cuisins

In [23]:
```
final_df.Cuisines[:10]
```

Out[23]:
```
0           French, Japanese, Desserts
1                             Japanese
2     Seafood, Asian, Filipino, Indian
3                      Japanese, Sushi
4                     Japanese, Korean
5                              Chinese
6                      Asian, European
7    Seafood, Filipino, Asian, European
8               European, Asian, Indian
9                             Filipino
Name: Cuisines, dtype: object
```

In [24]:
```
final_df[final_df['Country']=='India']
```

Out[24]:

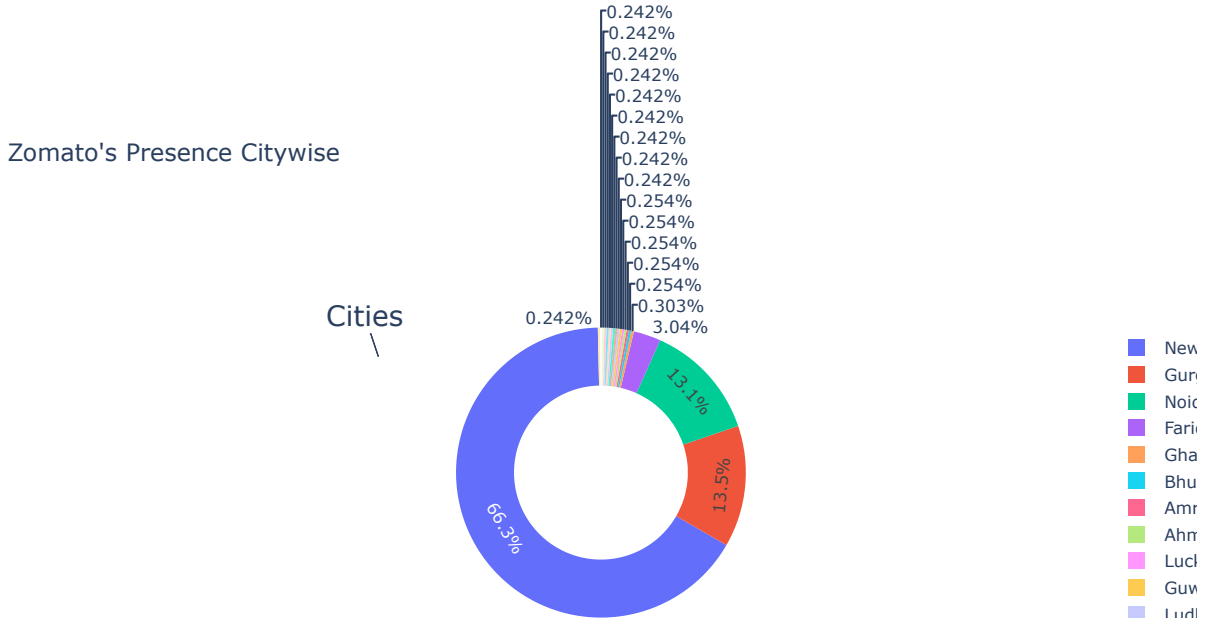
	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Has Table booking	Has Online delivery	Is delivering nov
624	3400025	Jahanpanah	1	Agra	E 23, Shopping Arcade, Sadar Bazaar, Agra Cant...	Agra Cantt	Agra Cantt, Agra	78.011544	27.161661	North Indian, Mughlai	...	No	No	No
625	3400341	Rangrezz Restaurant	1	Agra	E-20, Shopping Arcade, Sadar Bazaar, Agra Cant...	Agra Cantt	Agra Cantt, Agra	0.000000	0.000000	North Indian, Mughlai	...	No	No	No
626	3400005	Time2Eat - Mama Chicken	1	Agra	Main Market, Sadar Bazaar, Agra Cantt, Agra	Agra Cantt	Agra Cantt, Agra	78.011608	27.160832	North Indian	...	No	No	No
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	Rajasthani	...	No	No	No
628	3400017	Pinch Of Spice	1	Agra	23/453, Opposite Sanjay Cinema, Wazipura Road,...	Civil Lines	Civil Lines, Agra	78.007553	27.201725	North Indian, Chinese, Mughlai	...	No	No	No
...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
9271	2800100	D Cabana	1	Vizag	Beach Road, Near Bus Stop, Sagar Nagar, Visakh...	Sagar Nagar	Sagar Nagar, Vizag	83.361377	17.764287	Continental, Seafood, Chinese, North Indian, B...	...	No	No	No
9272	2800418	Kaloreez	1	Vizag	Plot 95, Opposite St. Lukes Nursing School, Da...	Siripuram	Siripuram, Vizag	0.000000	0.000000	Cafe, North Indian, Chinese	...	No	No	No
9273	2800881	Plot 17	1	Vizag	Plot 17, Gangapur Layout, Siripuram, Vizag	Siripuram	Siripuram, Vizag	83.315281	17.719539	Burger, Pizza, Biryani	...	No	No	No
9274	2800042	Vista - The Park	1	Vizag	The Park, Beach Road, Pedda Waltair, Lawsons B...	The Park, Lawsons Bay	The Park, Lawsons Bay, Vizag	83.336840	17.721182	American, North Indian, Thai, Continental	...	No	No	No
9275	2800019	Flying Spaghetti Monster	1	Vizag	10-50-12/F2, Sai Dakshata Complex, Beside Leno...	Waltair Uplands	Waltair Uplands, Vizag	83.314942	17.721119	Italian	...	No	No	No

8652 rows × 22 columns

In [43]:
```
# 1st way
final_df['Country'].groupby([final_df['Cuisines']]).size().sort_values().tail(10)

#  2 way
# final_df['Cuisines'].value_counts()[:10]
```

Out[43]:
```
Cuisines
Street Food                       149
Bakery, Desserts                  170
North Indian, Mughlai, Chinese    197
Bakery                            218
Cafe                              299
North Indian, Mughlai             334
Chinese                           354
Fast Food                         354
North Indian, Chinese             511
North Indian                      936
Name: Country, dtype: int64
```

In [27]:
```
plt.figure(figsize=(12,6))
# import plotly.plotly as py

labels = list(final_df.City.value_counts().head(20).index)
values = list(final_df.City.value_counts().head(20).values)

fig = {
    "data":[
        {
            "labels" : labels,
            "values" : values,
            "hoverinfo" : 'label+percent',
            "domain": {"x": [0, .9]},
            "hole" : 0.6,
            "type" : "pie",
            "rotation":120,
        },
    ],
    "layout": {
        "title" : "Zomato's Presence Citywise",
        "annotations": [
            {
                "font": {"size":20},
                "showarrow": True,
                "text": "Cities",
                "x":0.2,
                "y":0.9,
            },
        ]
    }
}

iplot(fig)
plt.show()
```

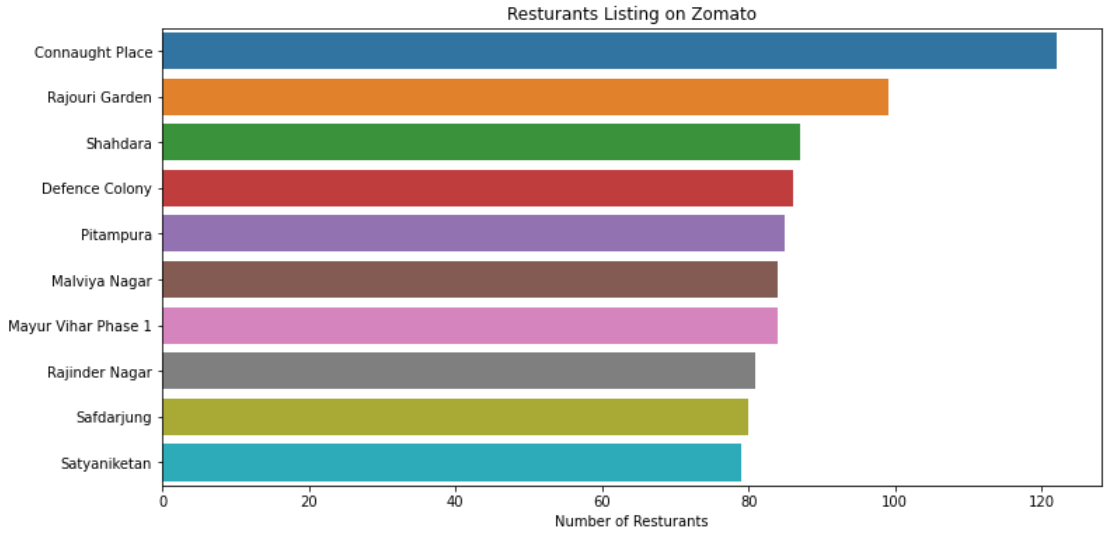



```
<Figure size 864x432 with 0 Axes>
```

- The data seems to be skewed towards New Delhi, Gurgaon, and Noida. I see minimal data for other cities. Hence I would do my analysis predominantly on New Delhi.

From which Locality maximum hotels are listed in Zomato

In [28]:
```
Delhi = final_df[(final_df.City == 'New Delhi')]
plt.figure(figsize=(12,6))
sns.barplot(x=Delhi.Locality.value_counts().head(10), y=Delhi.Locality.value_counts().head(10).index)

plt.ylabel(None);
plt.xlabel('Number of Resturants')
plt.title('Resturants Listing on Zomato');
```



What kind of Cuisine do these highly-rated restaurants offer

In [29]:
```
## Fetching the resturants having 'Excellent' and 'Very Good' rating
ConnaughtPlace = Delhi[(Delhi.Locality.isin(['Connaught Place'])) & (Delhi['Rating text'].isin(['Excellent','Very Good

ConnaughtPlace = ConnaughtPlace.Cuisines.value_counts().reset_index()

## Extracing all the cuisens in a single list
Cuisines = []
for x in ConnaughtPlace['index']:
    Cuisines.append(x)

# cuisines = '[%s]'%', '.join(map(str, cuisien))
Cuisines
```

Out[29]:
```
['North Indian, Chinese, Italian, Continental',
 'Continental, Italian, Asian, Indian',
 'Continental, Mediterranean, Italian, North Indian',
 'Bakery, Desserts, Fast Food',
 'North Indian, Continental',
 'North Indian, European, Asian, Mediterranean',
 'Continental, North Indian, Italian, Asian',
 'North Indian, Afghani, Mughlai',
 'North Indian, European',
 'Cafe',
 'Continental, Mexican, Burger, American, Pizza, Tex-Mex',
 'South Indian',
 'Asian, North Indian',
 'Italian, Mexican, Continental, North Indian, Finger Food',
 'Continental, American, Asian, North Indian',
 'Fast Food, American, Burger',
 'North Indian, Mediterranean, Asian, Fast Food',
 'Ice Cream',
 'Healthy Food, Continental, Italian',
 'Japanese',
 'Modern Indian',
 'Chinese',
 'Continental, North Indian, Chinese, Mediterranean',
 'North Indian, Chinese, Italian, American, Middle Eastern',
 'Biryani, Hyderabadi',
 'Biryani, North Indian, Hyderabadi',
 'North Indian, Chinese',
 'North Indian, Chinese, Continental, Italian',
 'North Indian, Italian, Asian, American',
 'North Indian',
 'Bakery, Fast Food, Desserts']
```

we try some new visualization via wordcloud library

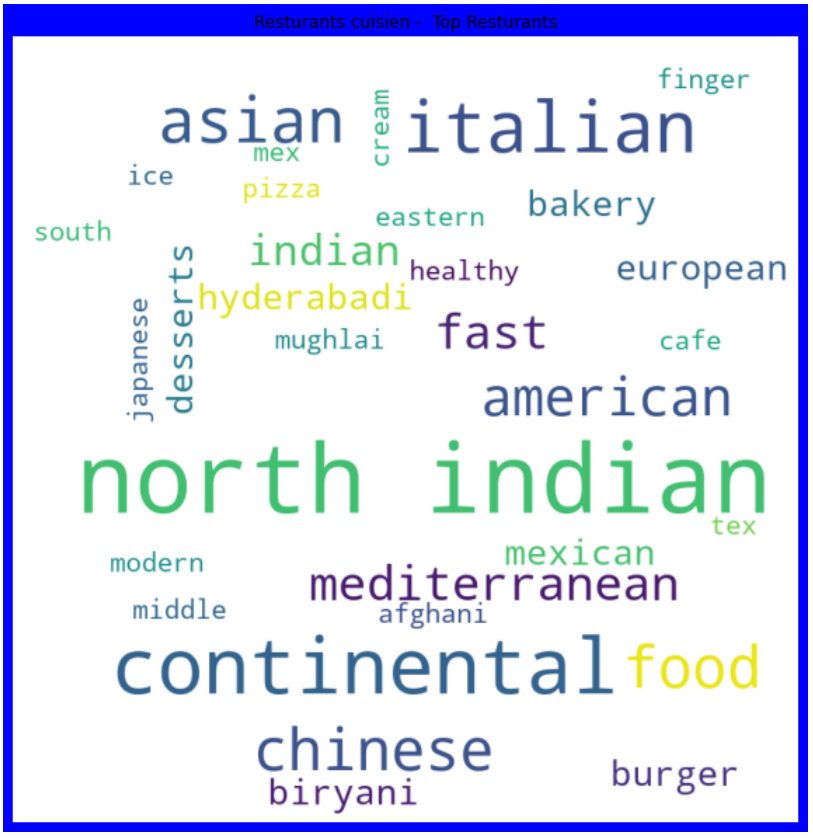
- Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analyzing data from social network websites.

In [46]:
```
pip install wordcloud
```

```
Collecting wordcloud
  Downloading wordcloud-1.8.2.2-cp39-cp39-win_amd64.whl (153 kB)
     ------------------------------------ 153.1/153.1 kB 2.3 MB/s eta 0:00:00
Requirement already satisfied: matplotlib in c:\users\keyur prajapati\appdata\local\programs\python\python39\lib\sit
e-packages (from wordcloud) (3.5.1)
Requirement already satisfied: numpy>=1.6.1 in c:\users\keyur prajapati\appdata\local\programs\python\python39\lib\s
ite-packages (from wordcloud) (1.22.3)
Requirement already satisfied: pillow in c:\users\keyur prajapati\appdata\local\programs\python\python39\lib\site-pa
ckages (from wordcloud) (9.0.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\keyur prajapati\appdata\local\programs\python\python39
\lib\site-packages (from matplotlib->wordcloud) (1.3.2)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\keyur prajapati\appdata\local\programs\python\python39
\lib\site-packages (from matplotlib->wordcloud) (4.30.0)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\keyur prajapati\appdata\local\programs\python\python39\l
ib\site-packages (from matplotlib->wordcloud) (3.0.7)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\keyur prajapati\appdata\local\programs\python\python
39\lib\site-packages (from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: packaging>=20.0 in c:\users\keyur prajapati\appdata\local\programs\python\python39\li
b\site-packages (from matplotlib->wordcloud) (21.3)
Requirement already satisfied: cycler>=0.10 in c:\users\keyur prajapati\appdata\local\programs\python\python39\lib\s
ite-packages (from matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: six>=1.5 in c:\users\keyur prajapati\appdata\local\programs\python\python39\lib\site-
packages (from python-dateutil>=2.7->matplotlib->wordcloud) (1.16.0)
Installing collected packages: wordcloud
Successfully installed wordcloud-1.8.2.2

[notice] A new release of pip available: 22.2 -> 22.2.2
[notice] To update, run: c:\users\keyur prajapati\appdata\local\programs\python\python39\python.exe -m pip install -
-upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [30]:
```
from wordcloud import WordCloud, STOPWORDS
```

In [31]:
```
comment_words = ''
stopwords = set(STOPWORDS)

# iterate through the csv file
for val in Cuisines:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()

    # Converts each token into lowercase
    for i in range(len(tokens)):
        tokens[i] = tokens[i].lower()

    comment_words += " ".join(tokens)+" "

wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (8, 8), facecolor = 'b', edgecolor='g')
plt.title('Resturants cuisien -  Top Resturants')
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

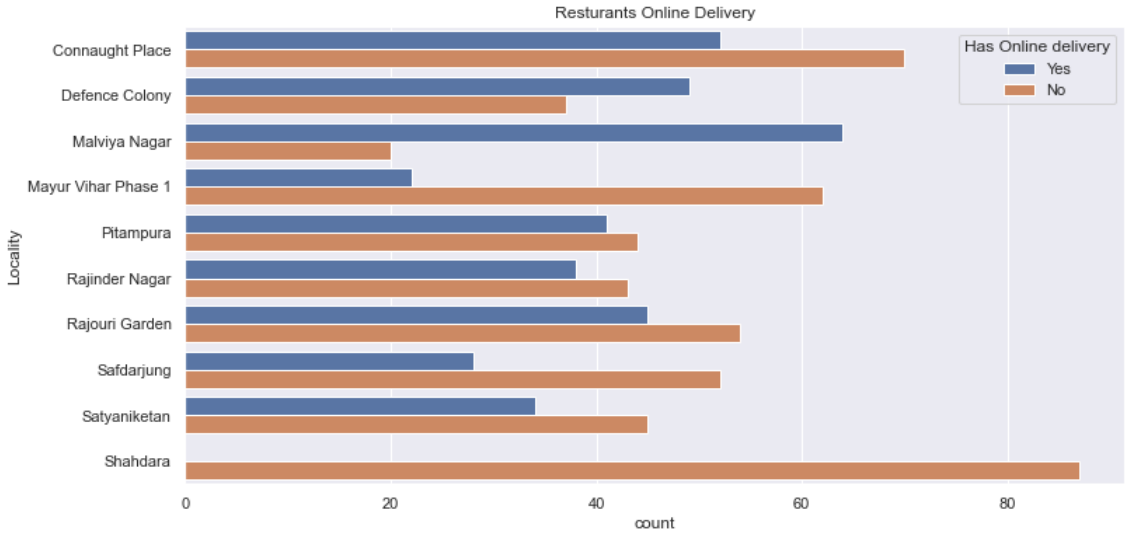


Top-rated restaurants seem to be doing well in the following cuisine

- North Indian
- Chinese
- Italian
- American

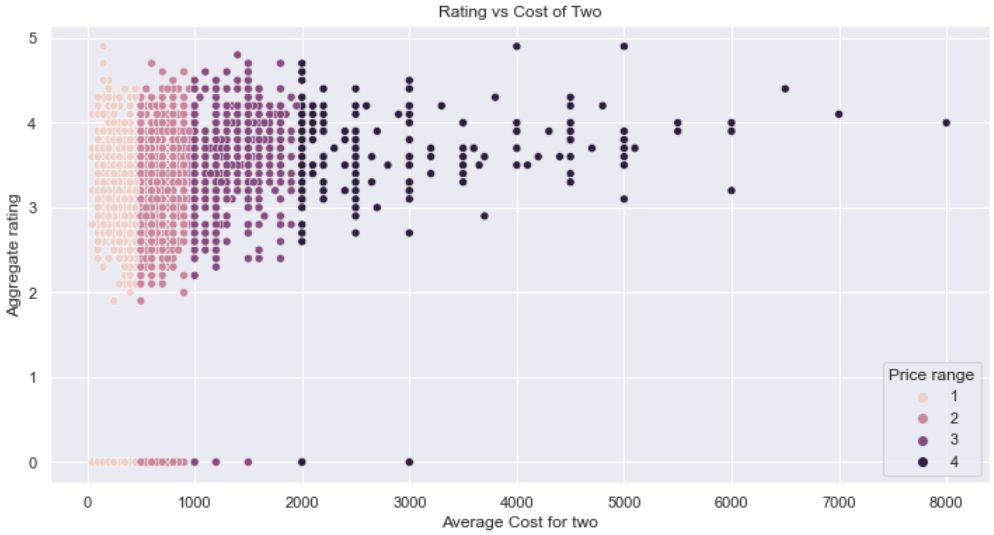
How many of such restaurants accept online delivery

In [32]:
```
top_locality = Delhi.Locality.value_counts().head(10)
sns.set_theme(style="darkgrid")
plt.figure(figsize=(12,6))
ax = sns.countplot(y= "Locality", hue="Has Online delivery", data=Delhi[Delhi.Locality.isin(top_locality.index)])
plt.title('Resturants Online Delivery');
```



Rating VS Cost of dinning

In [33]:
```
plt.figure(figsize=(12,6))
sns.scatterplot(x="Average Cost for two", y="Aggregate rating", hue='Price range', data=Delhi)

plt.xlabel("Average Cost for two")
plt.ylabel("Aggregate rating")
plt.title('Rating vs Cost of Two');
```

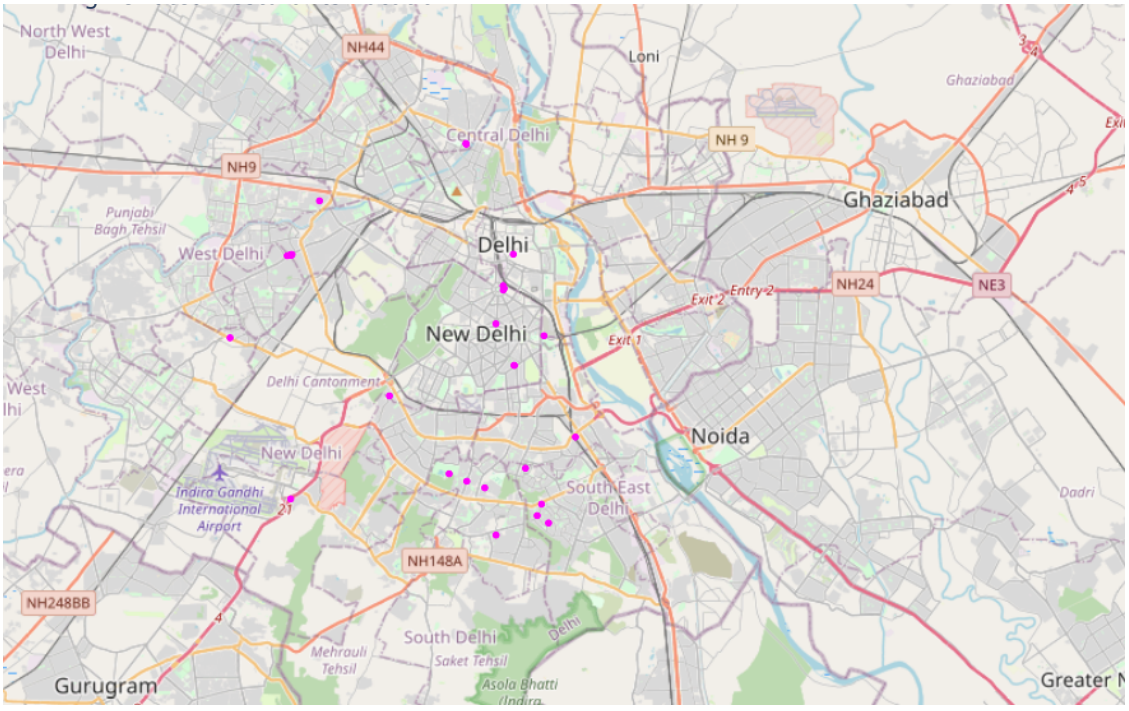


Location of Highly rated restaurants across New Delhi

In [34]:
```
Delhi['Rating text'].value_counts()
```

Out[34]:
```
Average      2495
Not rated    1425
Good         1128
Very Good     300
Poor           97
Excellent      28
Name: Rating text, dtype: int64
```

In [35]:
```
import plotly.express as px
Highly_rated = Delhi[Delhi['Rating text'].isin(['Excellent'])]

fig = px.scatter_mapbox(Highly_rated, lat="Latitude", lon="Longitude", hover_name="City", hover_data=["Aggregate ratir
                        color_discrete_sequence=["fuchsia"], zoom=10, height=300)
fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.update_layout(title='Highle rated Resturants Location',
                  autosize=True,
                  hovermode='closest',
                  showlegend=False)
fig.update_layout(
    autosize=False,
    width=800,
    height=500,)

fig.show()
```



Common Eateries

1 : Breakfast and Coffee locations

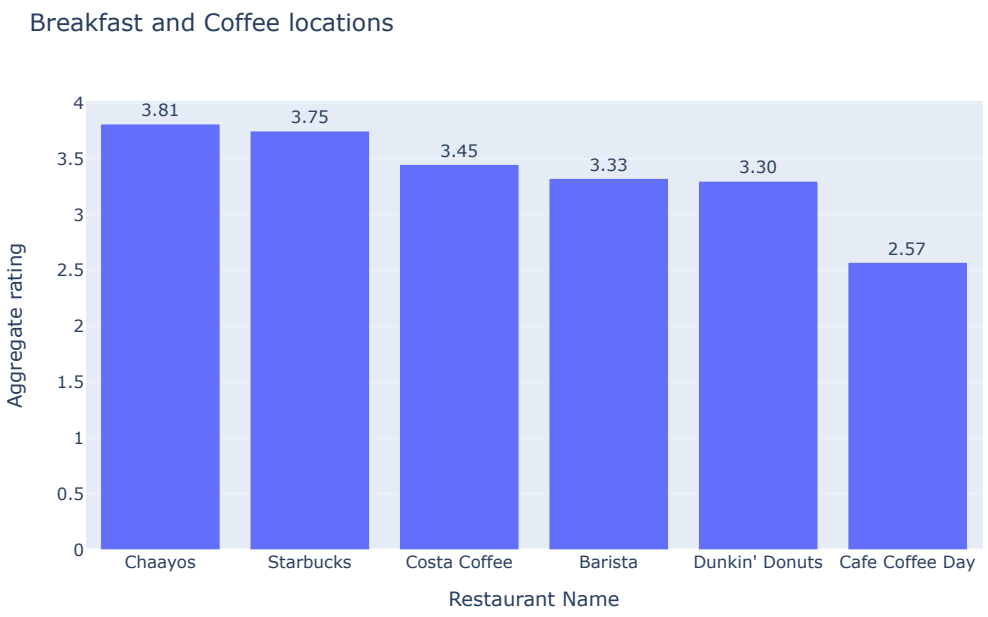
In [36]:
```
types = {
    "Breakfast and Coffee" : ["Cafe Coffee Day", "Starbucks", "Barista", "Costa Coffee", "Chaayos", "Dunkin' Donuts"],
    "American": ["Domino's Pizza", "McDonald's", "Burger King", "Subway", "Dunkin' Donuts", "Pizza Hut"],
    "Ice Creams and Shakes": ["Keventers", "Giani", "Giani's", "Starbucks", "Baskin Robbins", "Nirula's Ice Cream"]
}

breakfast = Delhi[Delhi['Restaurant Name'].isin(types['Breakfast and Coffee'])]
american = Delhi[Delhi['Restaurant Name'].isin(types['American'])]
ice_cream = Delhi[Delhi['Restaurant Name'].isin(types['Ice Creams and Shakes'])]
```

In [37]:
```
breakfast = breakfast[['Restaurant Name','Aggregate rating']].groupby('Restaurant Name').mean().reset_index().sort_val
breakfast
```

Out[37]:

	Restaurant Name	Aggregate rating
2	Chaayos	3.812500
5	Starbucks	3.750000
3	Costa Coffee	3.450000
0	Barista	3.325000
4	Dunkin' Donuts	3.300000
1	Cafe Coffee Day	2.573684

In [38]:
```
df= breakfast
fig = px.bar(df, y='Aggregate rating', x='Restaurant Name', text='Aggregate rating', title="Breakfast and Coffee locat
fig.update_traces(texttemplate='%{text:.3s}', textposition='outside')
fig.update_layout(
    autosize=False,
    width=800,
    height=500,)
fig.show()
```



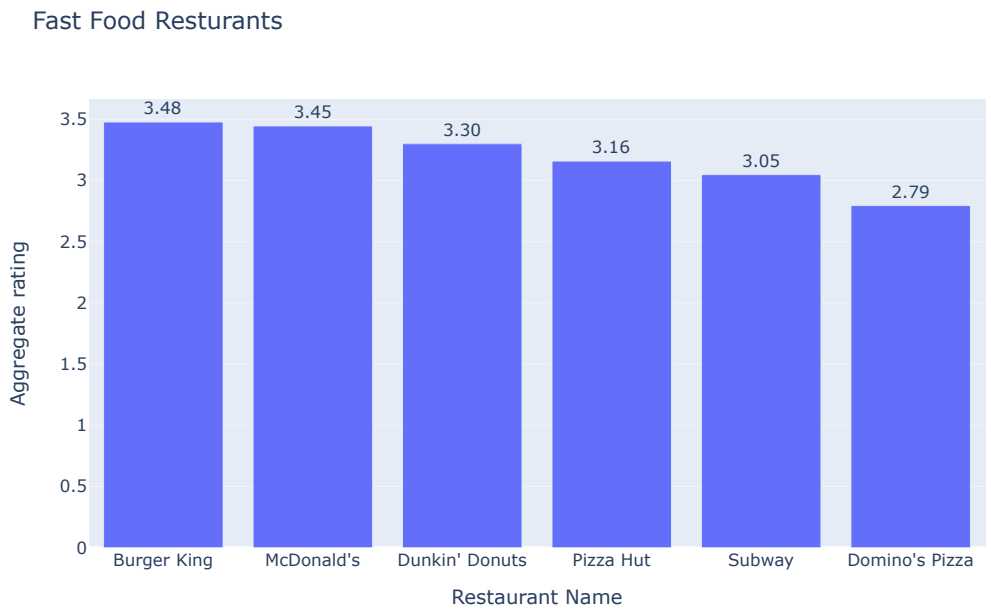
2 : Fast Food Restaurants

In [39]:
```
american = american[['Restaurant Name','Aggregate rating']].groupby('Restaurant Name').mean().reset_index().sort_value
american
```

Out[39]:

	Restaurant Name	Aggregate rating
0	Burger King	3.477778
3	McDonald's	3.445455
2	Dunkin' Donuts	3.300000
4	Pizza Hut	3.158333
5	Subway	3.047368
1	Domino's Pizza	2.794545

In [40]:
```
df= american
fig = px.bar(df, y='Aggregate rating', x='Restaurant Name', text='Aggregate rating', title="Fast Food Resturants")
fig.update_traces(texttemplate='%{text:.3s}', textposition='outside')
fig.update_layout(
    autosize=False,
    width=800,
    height=500,)

fig.show()
```



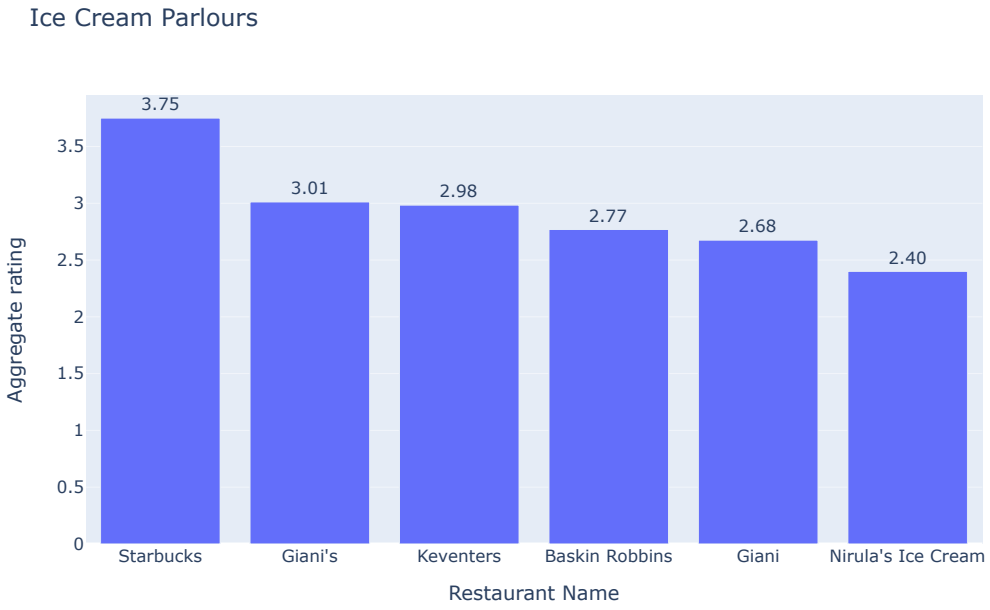
3: Ice Cream Parlors

In [41]:
```
ice_cream = ice_cream[['Restaurant Name','Aggregate rating']].groupby('Restaurant Name').mean().reset_index().sort_val
ice_cream
```

Out[41]:

	Restaurant Name	Aggregate rating
5	Starbucks	3.750000
2	Giani's	3.011765
3	Keventers	2.983333
0	Baskin Robbins	2.769231
1	Giani	2.675000
4	Nirula's Ice Cream	2.400000

In [42]:
```
df= ice_cream
fig = px.bar(df, y='Aggregate rating', x='Restaurant Name', text='Aggregate rating', title="Ice Cream Parlours")
fig.update_traces(texttemplate='%{text:.3s}', textposition='outside')
fig.update_layout(
    autosize=False,
    width=800,
    height=500,)
fig.show()
```



Conclusions

We've drawn many inferences from the survey. Here's a summary of a few of them:

- The dataset is skewed towards India and doesn't represent the complete data of restaurants worldwide.

Restaurants rating is categorized in six categories

- Not Rated
- Average
- Good
- Very Good
- Excellent

Connaught Palace has maximum restaurants listed on Zomato but in terms of online delivery acceptance Defence colony and Malviya Nagar seems to be doing better.

The top-rated restaurants seem to be getting a better rating on the following cuisine

- North Indian
- Chinese
- American
- Italian

There is no relation between cost and rating. Some of the best-rated restaurants are low on cost and vice versa.

On common Eateries, For Breakfast and Coffee location, Indian restaurants seem to be better rated but for Fast food chain and Ice cream parlors, American restaurants seem to be doing better.

In [ ]: