Exploratory Data Analysis - Zomato Dataset

In [1]:

```
# importing necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [67]:

```
import matplotlib
matplotlib.rcParams['figure.figsize']=(12, 6)
```

In [3]:

```
# importing the zomato dataset
df = pd.read_csv('zomato.csv',encoding='latin-1')
```

In [5]:

```
df.head()
```

Out[5]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude
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
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
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
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
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

5 rows × 21 columns

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

Out[6]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude
9546	5915730	NamlÛ± Gurme	208	ÛÁstanbul	Kemanke□ô Karamustafa Pa□ôa Mahallesi, RÛ±htÛ±...	Karakí_y	Karakí_y, ÛÁstanbul	28.977392	41.022793
9547	5908749	Ceviz AÛôacÛ±	208	ÛÁstanbul	Ko□ôuyolu Mahallesi, Muhittin îìstí_ndaÛô Cadd...	Ko□ôuyolu	Ko□ôuyolu, ÛÁstanbul	29.041297	41.009847
9548	5915807	Huqqa	208	ÛÁstanbul	Kuruí_e□ôme Mahallesi, Muallim Naci Caddesi, N...	Kuruí_e□ôme	Kuruí_e□ôme, ÛÁstanbul	29.034640	41.055817
9549	5916112	A□ô□ôk Kahve	208	ÛÁstanbul	Kuruí_e□ôme Mahallesi, Muallim Naci Caddesi, N...	Kuruí_e□ôme	Kuruí_e□ôme, ÛÁstanbul	29.036019	41.057979
9550	5927402	Walter's Coffee Roastery	208	ÛÁstanbul	CafeaÛôa Mahallesi, BademaltÛ± Sokak, No 21/B,...	Moda	Moda, ÛÁstanbul	29.026016	40.984776

5 rows × 21 columns

In [9]:
```
df.columns
```

Out[9]:
```
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 [10]:
```
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 [11]:

```
df.describe()
```

Out[11]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	Vot
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.0000
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666370	156.9097
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516378	430.1691
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000000	0.0000
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500000	5.0000
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200000	31.0000
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700000	131.0000
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900000	10934.0000

In Data Analysis what all things we do?

1. Missing Values
2. Explore about the Numerical Veriables
3. Explore about the Categorical Veriables
4. Finding Relationships between features

In [12]:

```
# Checking the null values
df.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
dtype: int64
```

In [17]:

```
# Finding features that has missing values
[feature for feature in df.columns if df[feature].isnull().sum()>0]
```

Out[17]:

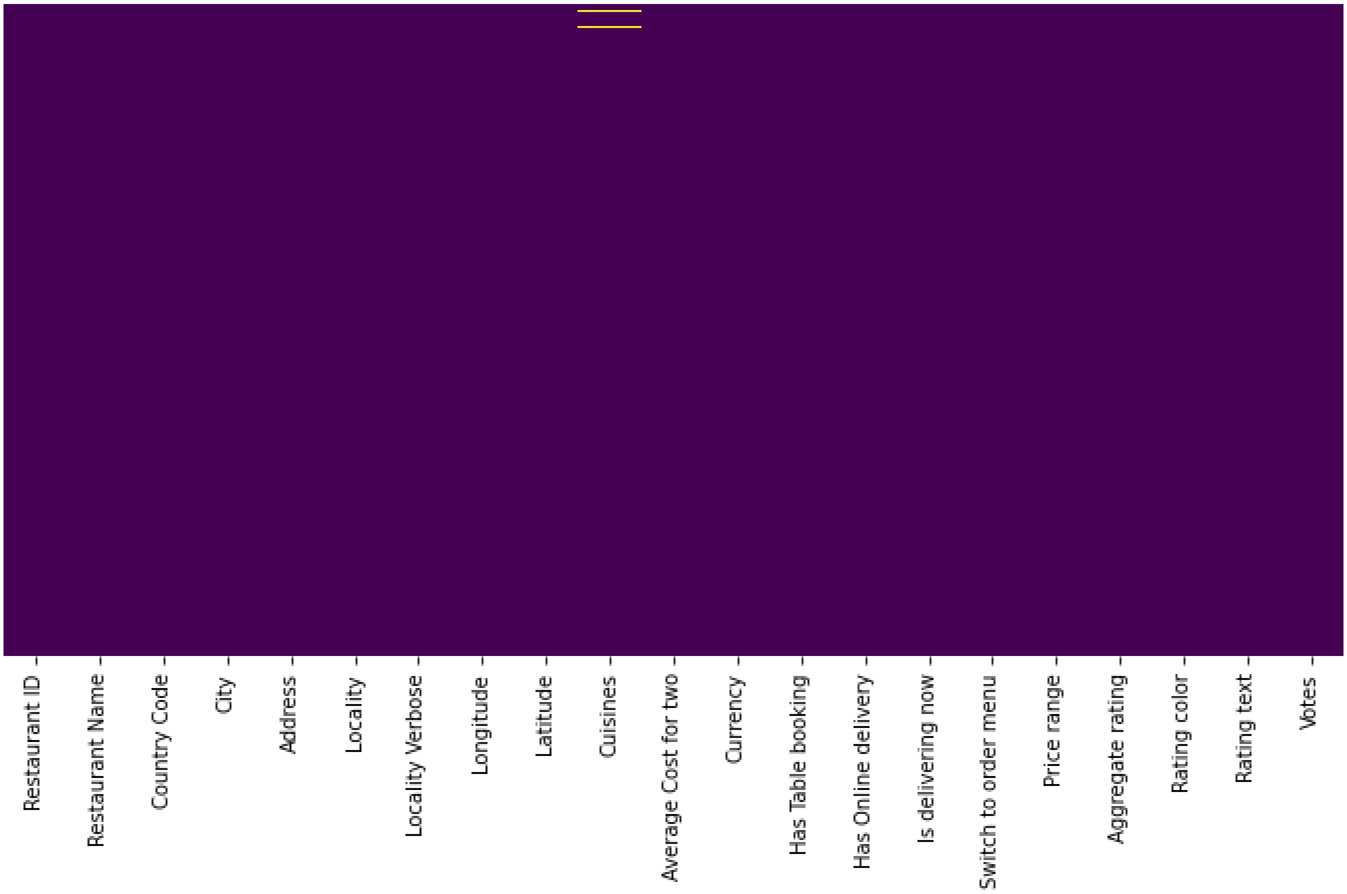
```
['Cuisines']
```

In [68]:

```
# Seeing the null values using Heatmap
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[68]:

```
<AxesSubplot:>
```



In [20]:

```
df.shape
```

Out[20]:

```
(9551, 21)
```

Null values are not visible because total rows= 9551, out of only 9 values are null

In [21]:

```
# Importing the other dataset
df_cc = pd.read_excel('Country-Code.xlsx')
```

In [22]:

```
df_cc.head()
```

Out[22]:

	Country Code	Country
0	1	India
1	14	Australia
2	30	Brazil

3	37	Canada
4	94	Indonesia

In [23]:
```
df.columns
```

Out[23]:
```
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 [24]:
```
df_cc.columns
```

Out[24]:
```
Index(['Country Code', 'Country'], dtype='object')
```

In [25]:
```
# Both the dataframe has common column 'Country Code'
# Let's combine both the data frame
pd.merge(df, df_cc, on='Country Code', how='left')
```

Out[25]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Lati
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	14.56
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	14.55
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	14.58
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	14.58
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	14.58
...	...	...	...	...	...	...	...	...	...
9546	5915730	NamlÛ± Gurme	208	ÛÁstanbul	Kemanke□ô Karamustafa Pa□ôa Mahallesi, RÛ±htÛ±...	Karakí_y	Karakí_y, ÛÁstanbul	28.977392	41.02

9547	5908749	Ceviz AÛôacÛ±	208	ÛÁstanbul	Ko□ôuyolu Mahallesi, Muhittin îistí_ndaÛô Cadd...	Ko□ôuyolu	Ko□ôuyolu, ÛÁstanbul	29.041297	41.00
9548	5915807	Huqqa	208	ÛÁstanbul	Kuruí_e□ôme Mahallesi, Muallim Naci Caddesi, N...	Kuruí_e□ôme	Kuruí_e□ôme, ÛÁstanbul	29.034640	41.05
9549	5916112	A□ô□ôk Kahve	208	ÛÁstanbul	Kuruí_e□ôme Mahallesi, Muallim Naci Caddesi, N...	Kuruí_e□ôme	Kuruí_e□ôme, ÛÁstanbul	29.036019	41.05
9550	5927402	Walter's Coffee Roastery	208	ÛÁstanbul	CafeaÛôa Mahallesi, BademaltÛ± Sokak, No 21/B,...	Moda	Moda, ÛÁstanbul	29.026016	40.98

9551 rows × 22 columns

In [26]:

```
final_df = pd.merge(df, df_cc, on='Country Code', how='left')
```

In [28]:

```
final_df.head(2)
```

Out[28]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...
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	...
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	...

2 rows × 22 columns

In [29]:

```
# Checking datatypes
final_df.dtypes
```

Out[29]:

```
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
Country                  object
dtype: object
```

In [31]:

```python
# Countrtywise total records
final_df.Country.value_counts()
```

Out[31]:

```
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
```

Conclusion: Here we can conclude like, most of the business of Zomato is having in India only, then few in U.S, rest are very less

In [37]:

```python
country_names = final_df.Country.value_counts().index
```

In [38]:

```python
country_names
```

Out[38]:

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

In [41]:

```python
country_values = final_df.Country.value_counts().values
```

In [42]:

```python
country_values
```

Out[42]:

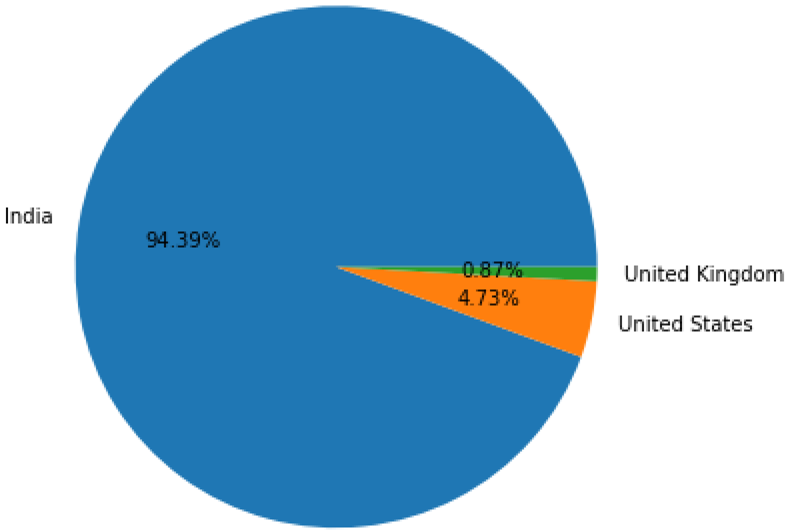
```
array([8652,   434,    80,    60,    60,    60,    40,    34,    24,    22,    21,
         20,    20,    20,     4], dtype=int64)
```

In [69]:

```python
# Top 3 countries where zomato is there
plt.pie(country_values[:3], labels=country_names[:3],autopct='%1.2f%%')
```

Out[69]:

```
([<matplotlib.patches.Wedge at 0x26aa7c00100>,
  <matplotlib.patches.Wedge at 0x26aa7c00460>,
  <matplotlib.patches.Wedge at 0x26aa7c11940>],
 [Text(-1.0829742700952103, 0.19278674827836725, 'India'),
  Text(1.077281715838356, -0.22240527134123297, 'United States'),
  Text(1.0995865153823035, -0.03015783794312073, 'United Kingdom')],
 [Text(-0.590713238233751, 0.10515640815183668, '94.39%'),
  Text(0.5876082086391032, -0.12131196618612707, '4.73%'),
  Text(0.5997744629358018, -0.01644972978715676, '0.87%')])
```



Observation: Zomato has having most number of transection from India, then US, and finally few from UK

In [46]:
```
final_df.columns
```

Out[46]:
```
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', 'Country'],
      dtype='object')
```

In [55]:
```
# Grouping all the records based on 'Aggregate rating', 'Rating color', 'Rating text' co
final_df.groupby(['Aggregate rating', 'Rating color', 'Rating text']).size().reset_index
```

Out[55]:

	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

In [66]:

```
ratings = final_df.groupby(['Aggregate rating', 'Rating color', 'Rating text']).size().r
```

Observation:

1. 4.5 to 4.9 = Excellent
2. 4.0 to 4.4 = Very Good
3. 3.5 to 3.9 = Good
4. 2.5 to 3.4 = Average
5. 1.8 to 2.4 = Poor

In [70]:

```
ratings.head()
```

Out[70]:

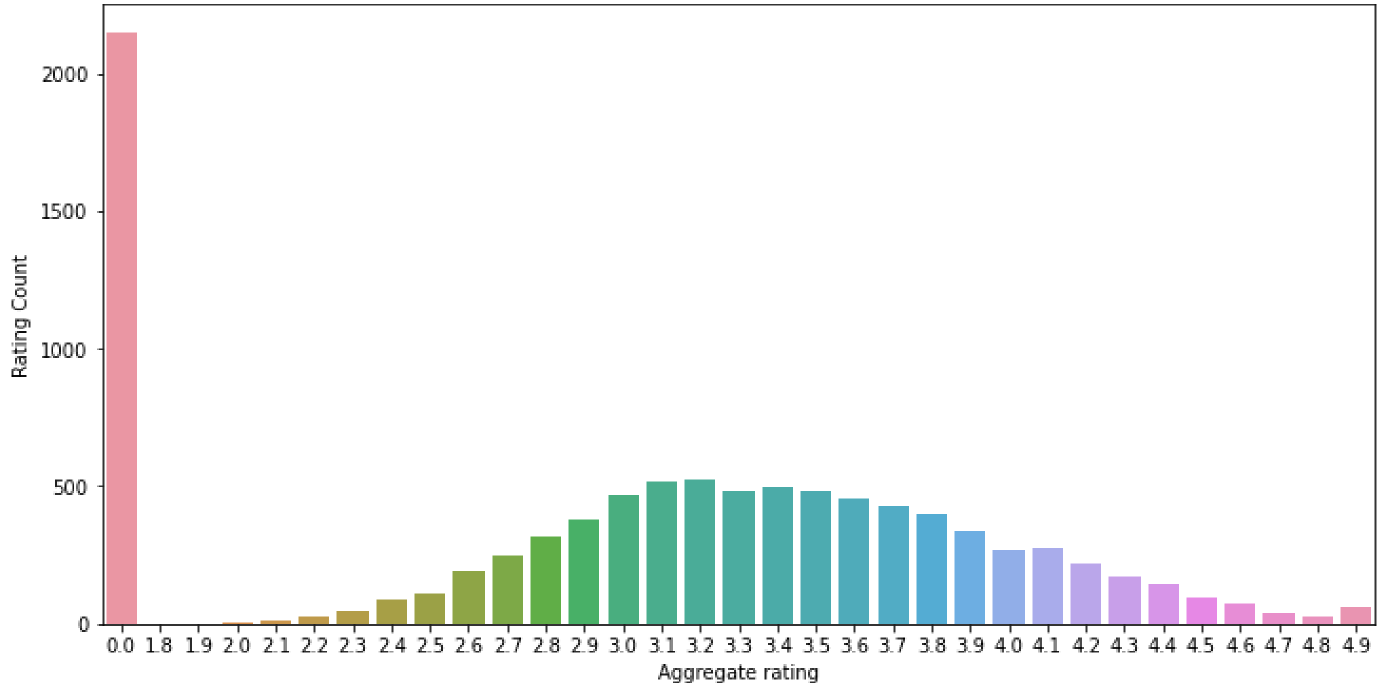
	Aggregate rating	Rating color	Rating text	Rating Count
0	0.0	White	Not rated	2148
1	1.8	Red	Poor	1
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15

In [73]:

```
sns.barplot(x= 'Aggregate rating', y= 'Rating Count', data= ratings)
```

Out[73]:

```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>
```

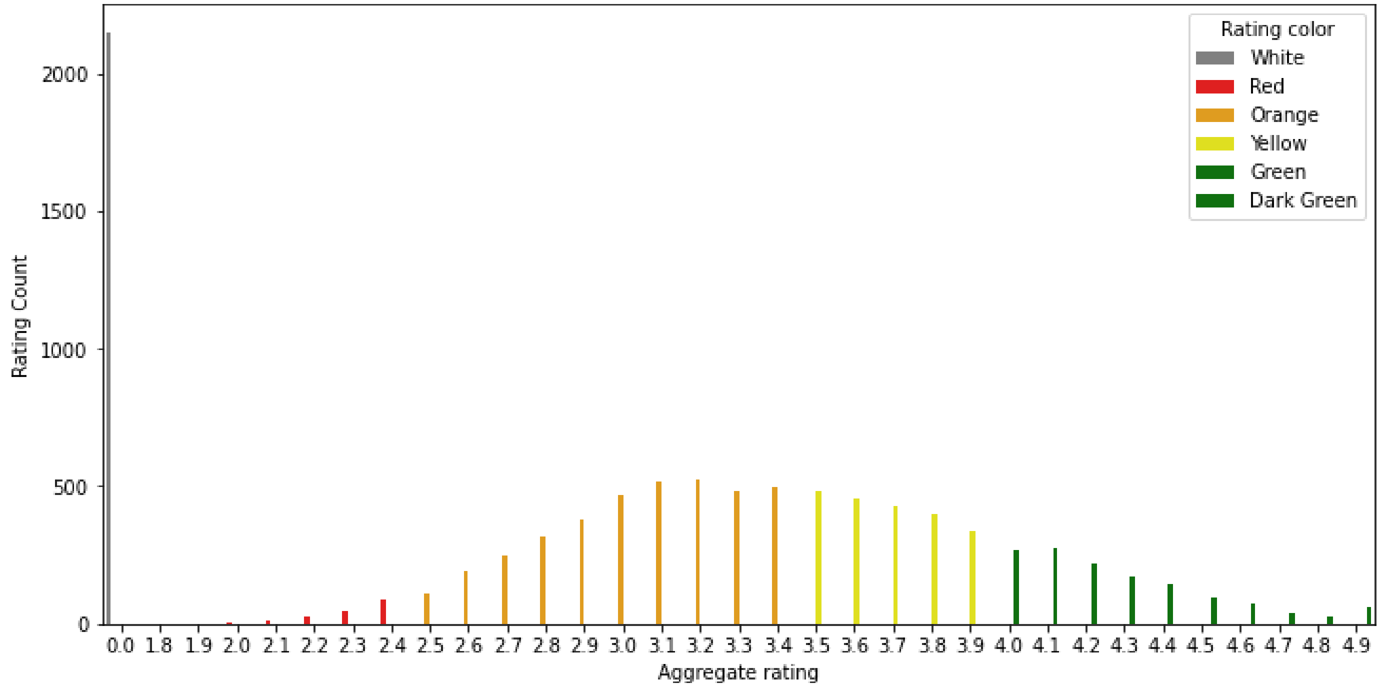


In [89]:

```
# plotting barplot with matching with 'Rating color' column
sns.barplot(x= 'Aggregate rating', y= 'Rating Count', hue='Rating color',data = ratings,
```

Out[89]:

```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>
```



Observation:

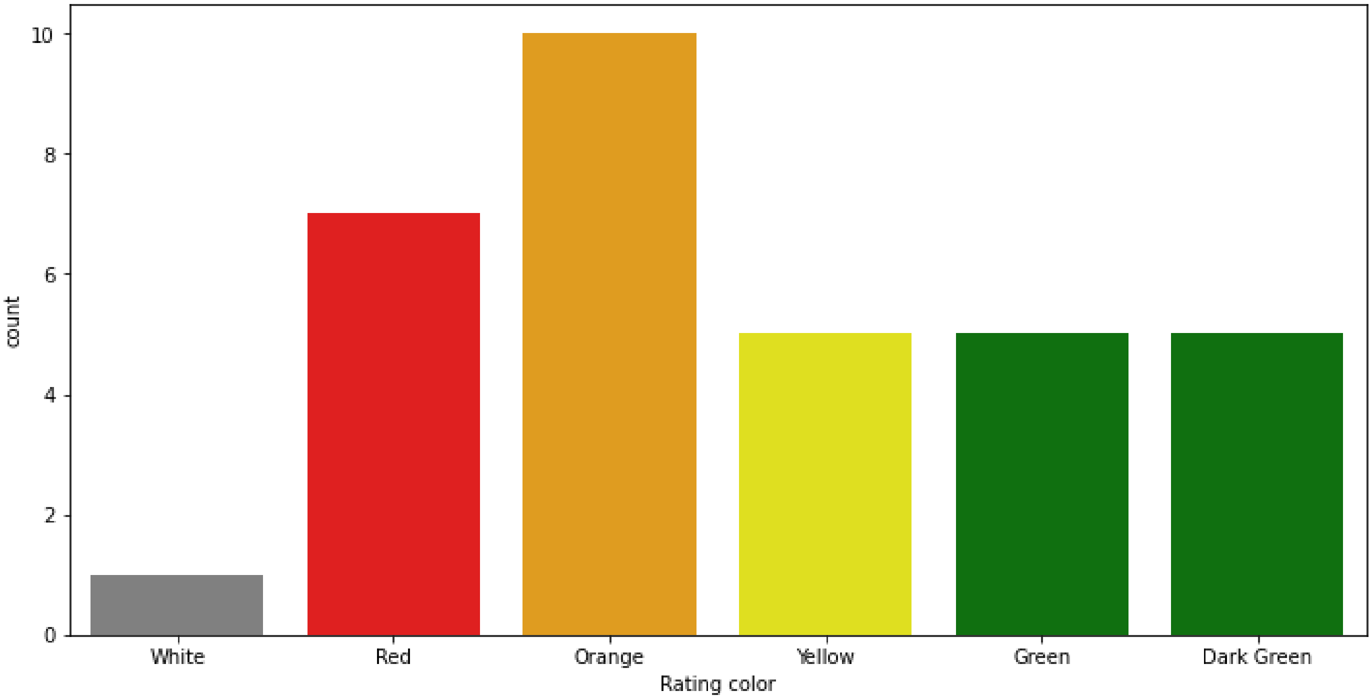
1. 0 rating count is very high
2. Most of the ratings are in between 2.6 to 3.9

In [80]:

```
# Countplot (this is working with categorical variables)
sns.countplot(x='Rating color', data=ratings, palette=['gray','red','orange','yellow','g
```

Out[80]:

```
<AxesSubplot:xlabel='Rating color', ylabel='count'>
```



Here we got the frequency of each category of 'Rating color' categorical feature

Find the Country names that has given 0 ratings

```
In [82]: final_df.columns
```

```
Out[82]: 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', 'Country'],
               dtype='object')
```

```
In [88]: final_df[final_df['Rating color']=='White'].groupby('Country').size().reset_index()
```

Out[88]:

	Country	0
0	Brazil	5
1	India	2139
2	United Kingdom	1
3	United States	3

Observation: Maximum number of 0 ratings are from India

Find out which currency is used by which country

```
In [90]: final_df.columns
```

```
Out[90]: 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', 'Country'],
               dtype='object')
```

In [103…

```
final_df[['Country','Currency']].drop_duplicates()
```

Out[103]:

	Country	Currency
0	Phillipines	Botswana Pula(P)
22	Brazil	Brazilian Real(R$)
82	United States	Dollar($)
102	Australia	Dollar($)
185	Canada	Dollar($)
456	Singapore	Dollar($)
564	UAE	Emirati Diram(AED)
624	India	Indian Rupees(Rs.)
9276	Indonesia	Indonesian Rupiah(IDR)
9297	New Zealand	NewZealand($)
9337	United Kingdom	Pounds(□£)
9417	Qatar	Qatari Rial(QR)
9437	South Africa	Rand(R)
9497	Sri Lanka	Sri Lankan Rupee(LKR)
9517	Turkey	Turkish Lira(TL)

In [106…

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

Out[106]:

	Country	Currency	0
0	Australia	Dollar($)	24
1	Brazil	Brazilian Real(R$)	60
2	Canada	Dollar($)	4
3	India	Indian Rupees(Rs.)	8652
4	Indonesia	Indonesian Rupiah(IDR)	21
5	New Zealand	NewZealand($)	40
6	Phillipines	Botswana Pula(P)	22
7	Qatar	Qatari Rial(QR)	20
8	Singapore	Dollar($)	20
9	South Africa	Rand(R)	60
10	Sri Lanka	Sri Lankan Rupee(LKR)	20
11	Turkey	Turkish Lira(TL)	34
12	UAE	Emirati Diram(AED)	60
13	United Kingdom	Pounds(□£)	80
14	United States	Dollar($)	434

Which countries do have online delivery options

In [107…

```
final_df.columns
```

Out[107]:

```
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', 'Country'],
      dtype='object')
```

In [130…

```
final_df[final_df['Has Online delivery']=='Yes'].value_counts().values
```

Out[130]:

```
array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
```

In [132…

```
final_df[['Country','Has Online delivery']].drop_duplicates().sort_values('Has Online de
```

Out[132]:

	Country	Has Online delivery
0	Phillipines	No
9437	South Africa	No
9417	Qatar	No
9337	United Kingdom	No
9297	New Zealand	No
9276	Indonesia	No
9497	Sri Lanka	No
624	India	No
564	UAE	No
456	Singapore	No
185	Canada	No
102	Australia	No
82	United States	No
22	Brazil	No
9517	Turkey	No
646	India	Yes
565	UAE	Yes

In [131…

```
final_df[['Country','Has Online delivery']].groupby(['Country','Has Online delivery']).s
```

Out[131]:

	Country	Has Online delivery	0
0	Australia	No	24
1	Brazil	No	60
2	Canada	No	4
3	India	No	6229
4	India	Yes	2423
5	Indonesia	No	21
6	New Zealand	No	40

7	Phillipines	No	22
8	Qatar	No	20
9	Singapore	No	20
10	South Africa	No	60
11	Sri Lanka	No	20
12	Turkey	No	34
13	UAE	No	32
14	UAE	Yes	28
15	United Kingdom	No	80
16	United States	No	434

Observation: Only in India and UAE is having online delivery facility

Pie chart distribution with respect to city (Top 5 city)

In [140…

```
city_values = final_df['City'].value_counts().values
```

In [141…

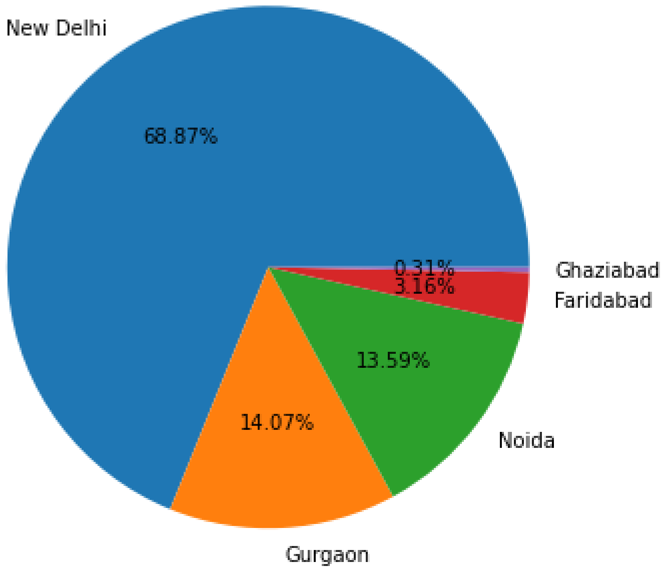
```
city_labels = final_df['City'].value_counts().index
```

In [144…

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

Out[144]:

```
([<matplotlib.patches.Wedge at 0x26aaf480df0>,
  <matplotlib.patches.Wedge at 0x26ab0636850>,
  <matplotlib.patches.Wedge at 0x26ab0636640>,
  <matplotlib.patches.Wedge at 0x26ab04b9310>,
  <matplotlib.patches.Wedge at 0x26ab0684700>],
 [Text(-0.6145352824185932, 0.9123301960708633, 'New Delhi'),
  Text(0.0623675251198054, -1.0982305276263407, 'Gurgaon'),
  Text(0.8789045225625368, -0.6614581167535246, 'Noida'),
  Text(1.0922218418223437, -0.13058119407559224, 'Faridabad'),
  Text(1.099946280005612, -0.010871113182029924, 'Ghaziabad')],
 [Text(-0.3352010631374145, 0.497634652402289, '68.87%'),
  Text(0.0340186500653484, -0.5990348332507311, '14.07%'),
  Text(0.47940246685229276, -0.36079533641101336, '13.59%'),
  Text(0.5957573682667329, -0.07122610585941394, '3.16%'),
  Text(0.5999706981848791, -0.005929698099289049, '0.31%')])
```



Observation:

1. These are the top 5 city which is having more number of transections

Find out Top 10 Cuisines

```
In [145…  final_df.columns
```

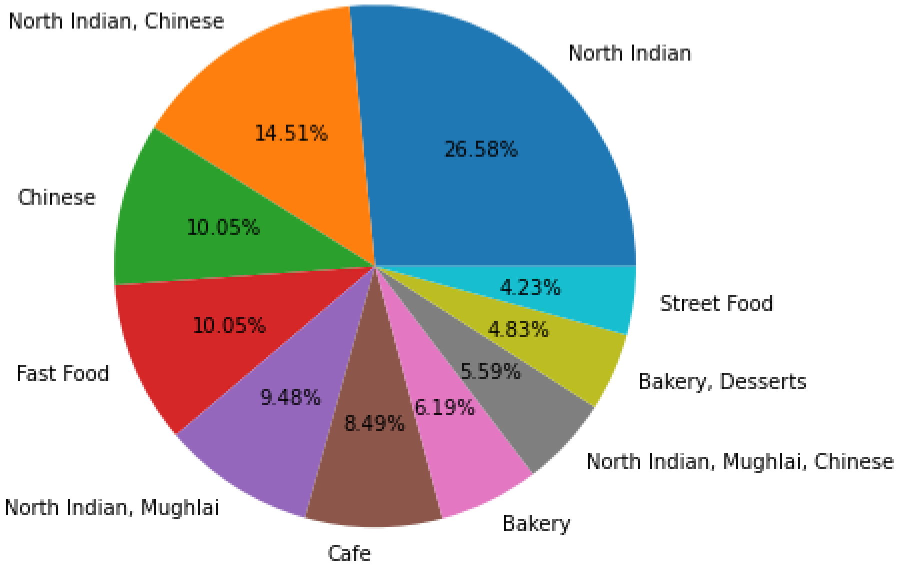
```
Out[145]: Index(['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address',
                 'Locality', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines',
                 'Average Cost for two', 'Currency', 'Has Table booking',
                 'Has Online delivery', 'Is delivering now', 'Switch to order menu',
                 'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
                 'Votes', 'Country'],
                dtype='object')
```

```
In [146…  cuisines_values = final_df['Cuisines'].value_counts().values
```

```
In [148…  cuisines_labels = final_df['Cuisines'].value_counts().index
```

```
In [150…  plt.pie(cuisines_values[:10],labels=cuisines_labels[:10],autopct='%1.2f%%')
```

```
Out[150]: ([<matplotlib.patches.Wedge at 0x26ab75787c0>,
             <matplotlib.patches.Wedge at 0x26ab7578ee0>,
             <matplotlib.patches.Wedge at 0x26ab7581640>,
             <matplotlib.patches.Wedge at 0x26ab7581d60>,
             <matplotlib.patches.Wedge at 0x26ab75894c0>,
             <matplotlib.patches.Wedge at 0x26ab7589be0>,
             <matplotlib.patches.Wedge at 0x26ab7590340>,
             <matplotlib.patches.Wedge at 0x26ab7590a60>,
             <matplotlib.patches.Wedge at 0x26ab75971c0>,
             <matplotlib.patches.Wedge at 0x26ab75978e0>],
            [Text(0.7383739846958008, 0.8153550507137645, 'North Indian'),
             Text(-0.5794679314239953, 0.9349956772366362, 'North Indian, Chinese'),
             Text(-1.067309479615702, 0.26617752482593154, 'Chinese'),
             Text(-1.0185984499802057, -0.4152796620326146, 'Fast Food'),
             Text(-0.5935788454809928, -0.9261015895664211, 'North Indian, Mughlai'),
             Text(-0.005887079599915552, -1.0999842463843672, 'Cafe'),
             Text(0.4842062514572988, -0.9876964645323336, 'Bakery'),
             Text(0.808736477166136, -0.7456174022251013, 'North Indian, Mughlai, Chinese'),
             Text(1.0055375294202338, -0.44597564611473206, 'Bakery, Desserts'),
             Text(1.090298995560443, -0.14576728123927227, 'Street Food')],
            [Text(0.4027494461977095, 0.4447391185711442, '26.58%'),
             Text(-0.316073417140361, 0.5099976421290743, '14.51%'),
             Text(-0.5821688070631101, 0.14518774081414446, '10.05%'),
             Text(-0.5555991545346576, -0.22651617929051704, '10.05%'),
             Text(-0.32377027935326874, -0.5051463215816842, '9.48%'),
             Text(-0.003211134327226664, -0.5999914071187457, '8.49%'),
             Text(0.26411250079489024, -0.5387435261085456, '6.19%'),
             Text(0.441128987545165, -0.40670040121369155, '5.59%'),
             Text(0.5484750160474001, -0.24325944333530836, '4.83%'),
             Text(0.5947085430329688, -0.07950942613051214, '4.23%')])
```



Observation: These are the top 10 Cuisines

```
In [ ]:
```