Exploratory data analysis

Zomato Data Set

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [9]:
```
df=pd.read_csv("zomato.csv",encoding='latin-1')
df.head()
```

Out[9]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	I
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
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
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
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
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong	121.057508	14

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	
					Megamall, Ortigas...		City, Mandal...		

5 rows × 21 columns

In [10]:
```
df.columns
```

Out[10]:
```
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 [11]:
```
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 [12]:
```
df.describe()
```

Out[12]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggregate rating	
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	

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

Data Analysis

missing value

Explore numrical and categorical values

Relationship between features

In [20]:
```
df.shape
```

Out[20]:
```
(9551, 21)
```

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
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
```

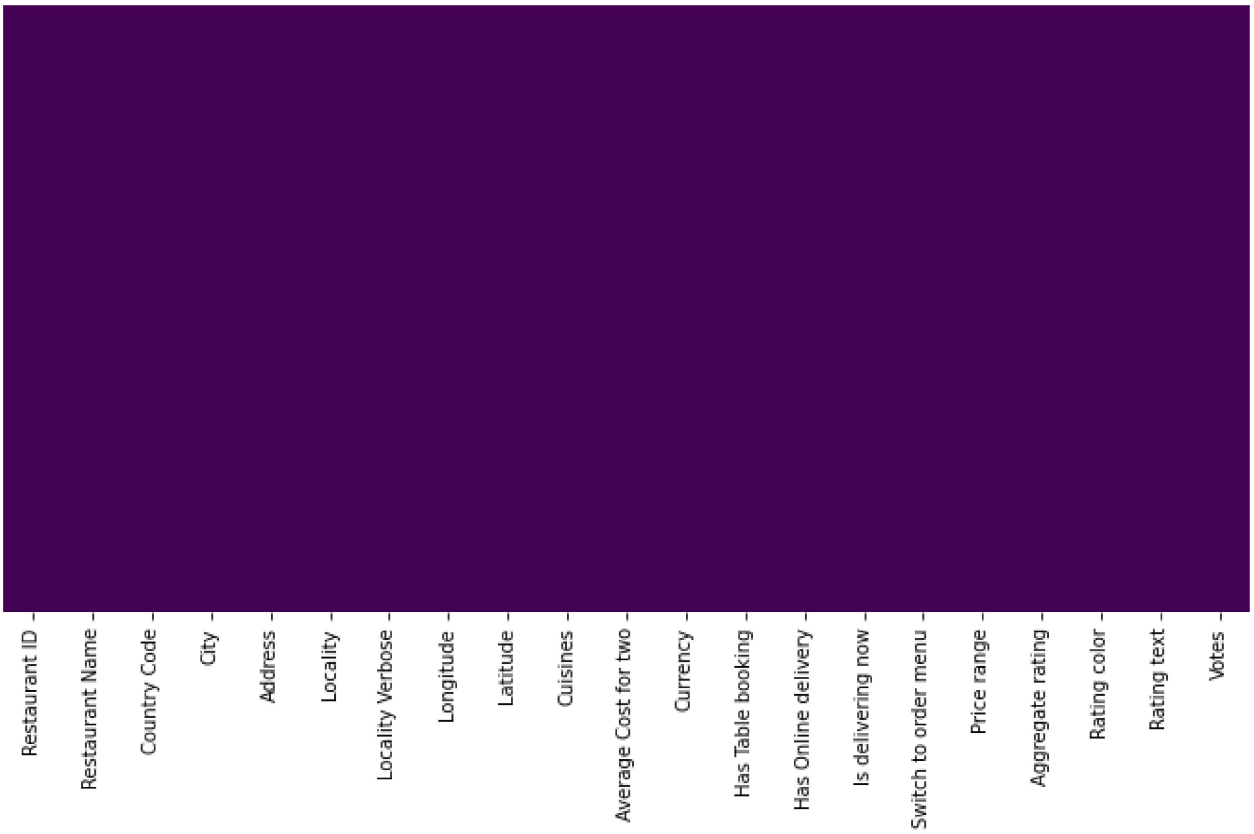
```
Votes                 0
dtype: int64
```

In [18]:
```
[features for features in df.columns if df[features].isnull().sum()>0]
#List comprehension
```

Out[18]:
```
['Cuisines']
```

In [73]:
```
#Plotting for Heat Map
sns.heatmap(df.isnull(), yticklabels=False, cbar=False, cmap='viridis')
```

Out[73]:
```
<AxesSubplot:>
```



In [24]:
```
df_country=pd.read_excel('Country-code.xlsx')
df_country.head()
```

Out[24]:

	Country Code	Country
0	1	India
1	14	Australia
2	30	Brazil
3	37	Canada
4	94	Indonesia

In [25]:
```
df.columns
```

Out[25]:
```
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')
```

Combining datasets zomato.csv and Country-code.xlsx because they have a common column named Country Code. joing both data sets using 'merge' and putting it in df_final dataset.

In [27]:
```
df_final=pd.merge(df,df_country, on='Country Code', how='left')
```

In [29]:
```
df_final.head(2)
```

Out[29]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuis
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	Fre Japar Des
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	Japa

2 rows × 22 columns

In [30]:
```
# To check data types
df_final.dtypes
```

Out[30]:
```
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]:
```
df_final.columns
```

Out[31]:
```
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 [36]:
```
df_final.Country.value_counts()
```

Out[36]:
```
India             8652
United States      434
United Kingdom      80
Brazil              60
South Africa        60
UAE                 60
New Zealand         40
Turkey              34
Australia           24
Phillipines         22
Indonesia           21
Sri Lanka           20
Singapore           20
Qatar               20
Canada               4
Name: Country, dtype: int64
```

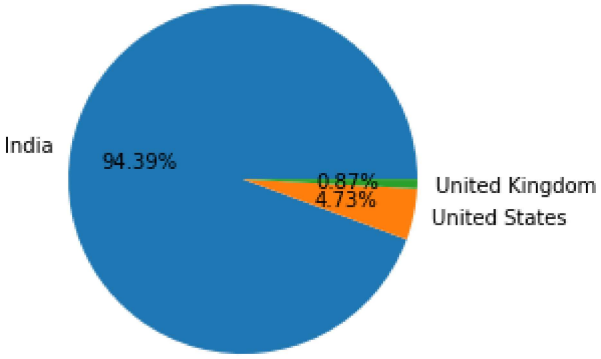
In [37]:
```
Country_names=df_final.Country.value_counts().index #y-axis for pie chart
```

In [40]:
```
Country_val=df_final.Country.value_counts().values #x-axis for pie chart
```

In [51]:
```
# Pie Chart
plt.pie(Country_val[:3],labels=Country_names[:3],autopct='%1.2f%%') #Top 3 countries th
```

Out[51]:
```
([<matplotlib.patches.Wedge at 0x1778d295b80>,
  <matplotlib.patches.Wedge at 0x1778d2a3340>,
  <matplotlib.patches.Wedge at 0x1778d2a3a00>],
 [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: The country with maximum % records of transactions are from India, then United States and 3rd is United Kingdom.

In [53]:
```
df_final.columns
```

Out[53]:
```
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 [65]:
```
ratings=df_final.groupby(['Aggregate rating', 'Rating color', 'Rating text']).size().re
#to chande it to a data frame
#and doing indexing and where column is 0 change it to Rating Count.
```

In [66]:
```
ratings
```

Out[66]:

	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

	Aggregate rating	Rating color	Rating text	Rating Count
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

1.When rating is between 4.5 to 4.9....> Excellent 2.When rating is between 4.0 to 4.4....> very good
3.When rating is between 3.5 to 3.9....> good 4.When rating is between 2.5 to 3.4.....> average
5.When rating is between 1.8 to 2.4.....> poor

In [67]:
```
ratings.head()
```

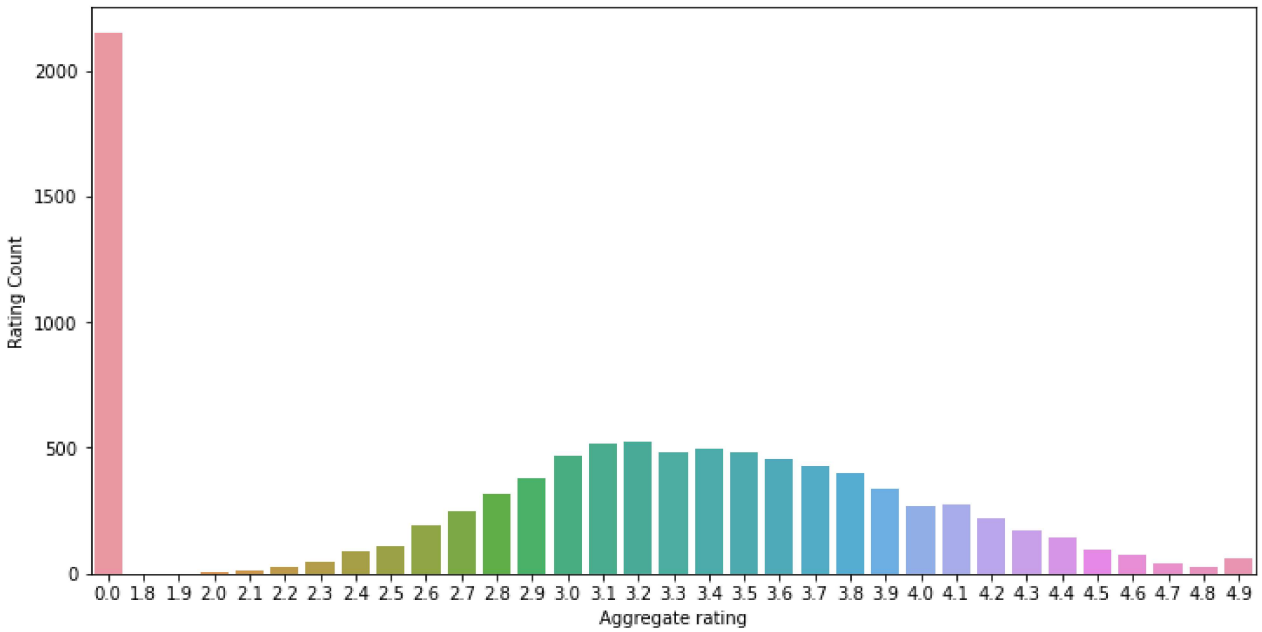
Out[67]:

	Aggregate rating	Rating color	Rating text	Rating Count
0	0.0	White	Not rated	2148
1	1.8	Red	Poor	1

	Aggregate rating	Rating color	Rating text	Rating Count
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15

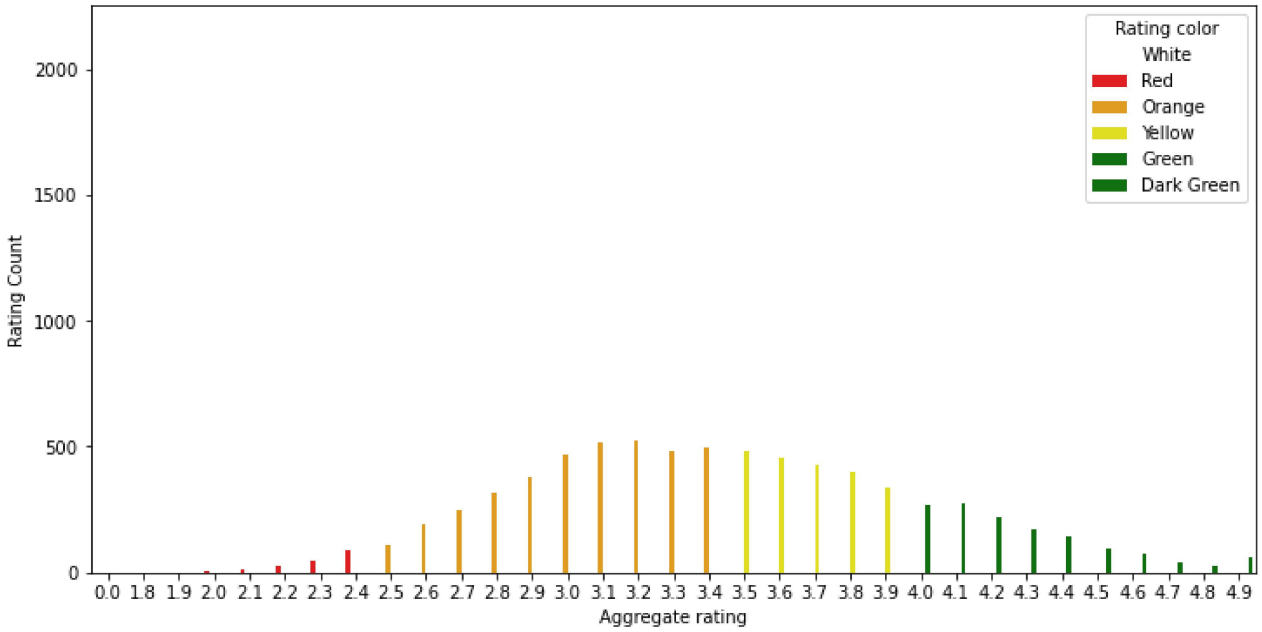
In [75]:
```
import matplotlib
matplotlib.rcParams['figure.figsize']= (12, 6)
sns.barplot(x='Aggregate rating',y='Rating Count', data=ratings)
```

Out[75]: <AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>



In [79]:
```
# Mapping color according to the rating
sns.barplot(x='Aggregate rating',y='Rating Count',hue='Rating color', data=ratings, pal
```

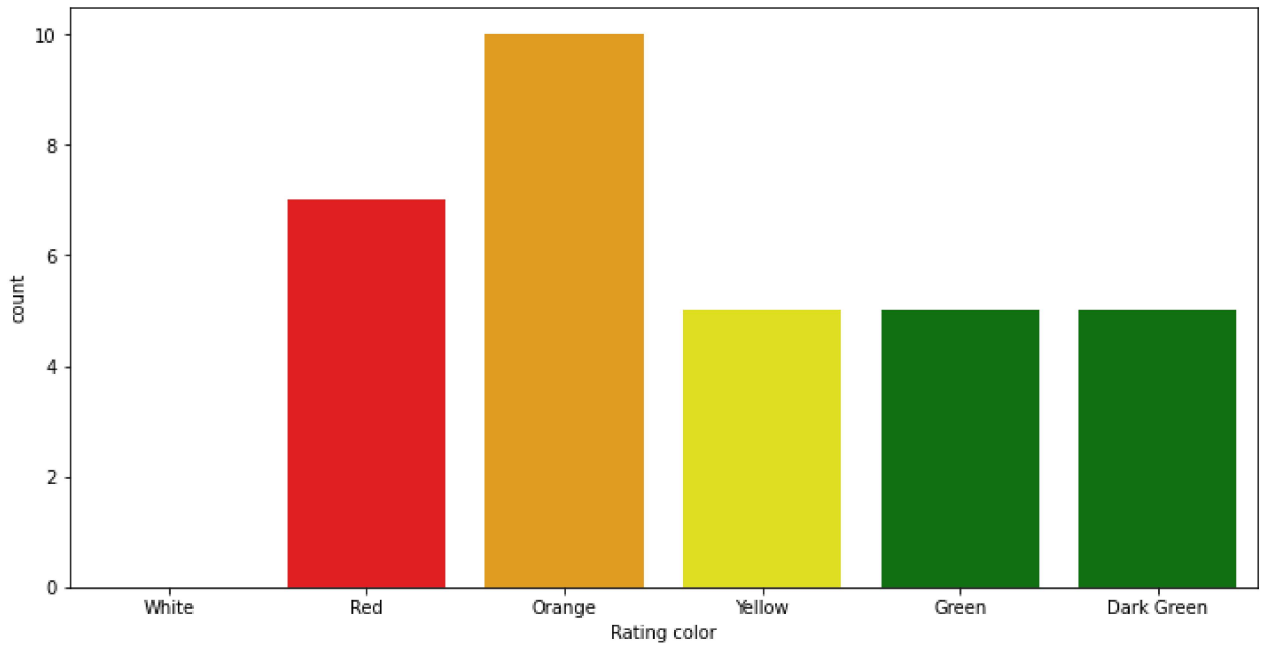
Out[79]: <AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>



In [ ]:
```
# Observation
1. Not rated count is very high
2. Maximum rating is between 2.5 to 3.4
```

In [140…
```
# Count plot
sns.countplot(x='Rating color', data=ratings, palette=['white', 'red', 'orange', 'yello
```

Out[140… <AxesSubplot:xlabel='Rating color', ylabel='count'>



In [81]:
```
ratings
```

Out[81]:

	Aggregate rating	Rating color	Rating text	Rating Count
0	0.0	White	Not rated	2148

	Aggregate rating	Rating color	Rating text	Rating Count
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 [ ]:
```
# Countries that has given 0 ratings
```

In [85]:
```
df_final[df_final['Rating color']=='White'].groupby('Country').size().reset_index()
```

Out[85]:

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

In [91]:
```
df_final.groupby(['Aggregate rating', 'Country']).size().reset_index().head(5)
```

Out[91]:

	Aggregate rating	Country	0
0	0.0	Brazil	5
1	0.0	India	2139
2	0.0	United Kingdom	1
3	0.0	United States	3
4	1.8	India	1

Observation Maximum number of 0 ratings is from India

In [93]:
```
# Which currency is used by which country?
df_final.columns
```

Out[93]:
```
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 [96]:
```
df_final.groupby(['Country', 'Currency']).size().reset_index()
```

Out[96]:

	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

	Country	Currency	0
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

In [101…

```
# Another method to see the country and currency is:
df_final[['Country', 'Currency']].groupby(['Country','Currency']).size().reset_index()
```

Out[101…

	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

In [109…

```
# Which Country do have online delivery options
df_final[df_final['Has Online delivery']=='Yes'].Country.value_counts()
```

Out[109…

```
India    2423
UAE        28
Name: Country, dtype: int64
```

In [110…

```
# Another method to see the countries which has online delivery
df_final.groupby(['Has Online delivery' ,'Country']).size().reset_index()
```

Out[110...

	Has Online delivery	Country	0
0	No	Australia	24
1	No	Brazil	60
2	No	Canada	4
3	No	India	6229
4	No	Indonesia	21
5	No	New Zealand	40
6	No	Phillipines	22
7	No	Qatar	20
8	No	Singapore	20
9	No	South Africa	60
10	No	Sri Lanka	20
11	No	Turkey	34
12	No	UAE	32
13	No	United Kingdom	80
14	No	United States	434
15	Yes	India	2423
16	Yes	UAE	28

Observation

1.Online deliveries are available in India and UAE

In [112...

```
# Create a pie chart for cities distribution
df_final.columns
```

Out[112...

```
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 [115...

```
df_final.City.value_counts().reset_index()
```

Out[115...

	index	City
0	New Delhi	5473
1	Gurgaon	1118

	index	City
2	Noida	1080
3	Faridabad	251
4	Ghaziabad	25
...	...	...
136	Yorkton	1
137	Paynesville	1
138	Victor Harbor	1
139	Armidale	1
140	Montville	1

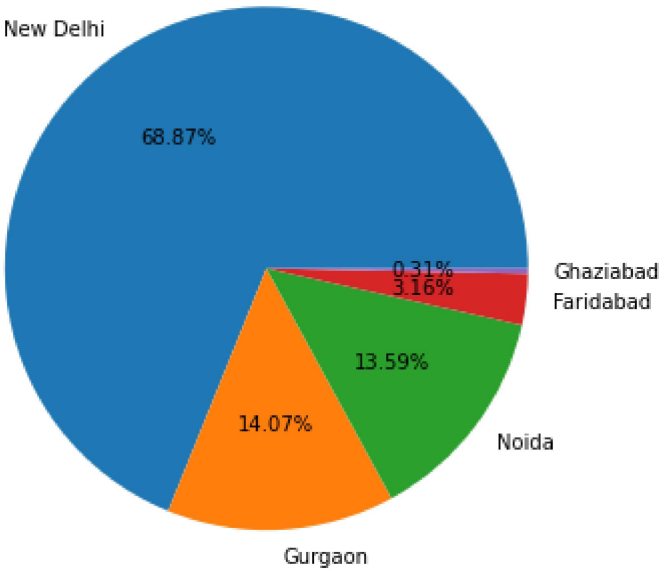
141 rows × 2 columns

In [122...
```
City_labels=df_final.City.value_counts().index
```

In [123...
```
City_val=df_final.City.value_counts().values
```

In [127...
```
# Top 5 City distribution
plt.pie(City_val[:5],labels=City_labels[:5],autopct='%1.2f%%')
```

Out[127...
```
([<matplotlib.patches.Wedge at 0x17791ef5dc0>,
  <matplotlib.patches.Wedge at 0x17791f044f0>,
  <matplotlib.patches.Wedge at 0x17791f04af0>,
  <matplotlib.patches.Wedge at 0x17791f13130>,
  <matplotlib.patches.Wedge at 0x17791f13730>],
 [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%')])
```



In [128…

```
# Find Top 10 cuisines
df_final.columns
```

Out[128…

```
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 [139…

```
df_final.Cuisines.value_counts().reset_index().head(10)
```

Out[139…

	index	Cuisines
0	North Indian	936
1	North Indian, Chinese	511
2	Chinese	354
3	Fast Food	354
4	North Indian, Mughlai	334
5	Cafe	299
6	Bakery	218
7	North Indian, Mughlai, Chinese	197
8	Bakery, Desserts	170
9	Street Food	149

In [ ]: