EDA for zomato dataset

In [3]:
```
# import all the libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [7]:
```
# Read the zomato dataset
df = pd.read_csv('/Users/madhu/Desktop/Data Science/iNeuron/EDA/zomato.csv', encoding='latin-1')
df.head()
```

Out[7]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency	Has Table booking	Has Online delivery	delive
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	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	...	Botswana Pula(P)	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	...	Botswana Pula(P)	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	...	Botswana Pula(P)	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	...	Botswana Pula(P)	Yes	No

5 rows × 21 columns

In [8]:
```
# Get the columns of df
df.columns
```

Out[8]:
```
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]:
```
# gives the basic info about data
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]:
```
# gives basic stats value of numerical variable
df.describe()
```

Out[11]:

	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 Data Analysis What All Things We Do

1. Missing Values
2. Explore About the Numerical Variables
3. Explore About categorical Variables
4. Finding Relationship between features

In [15]:
```
# Fins if there is any null value in any of the column.
# From the data we can say "cuisines" feature is having 9 null value
df.isnull().sum()
```

Out[15]:
```
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 [19]:
```
# Find only that particular column/feature which is having null value using list comprehension.
[features for features in df.columns if df[features].isnull().sum()>0]
```

Out[19]:
```
['Cuisines']
```

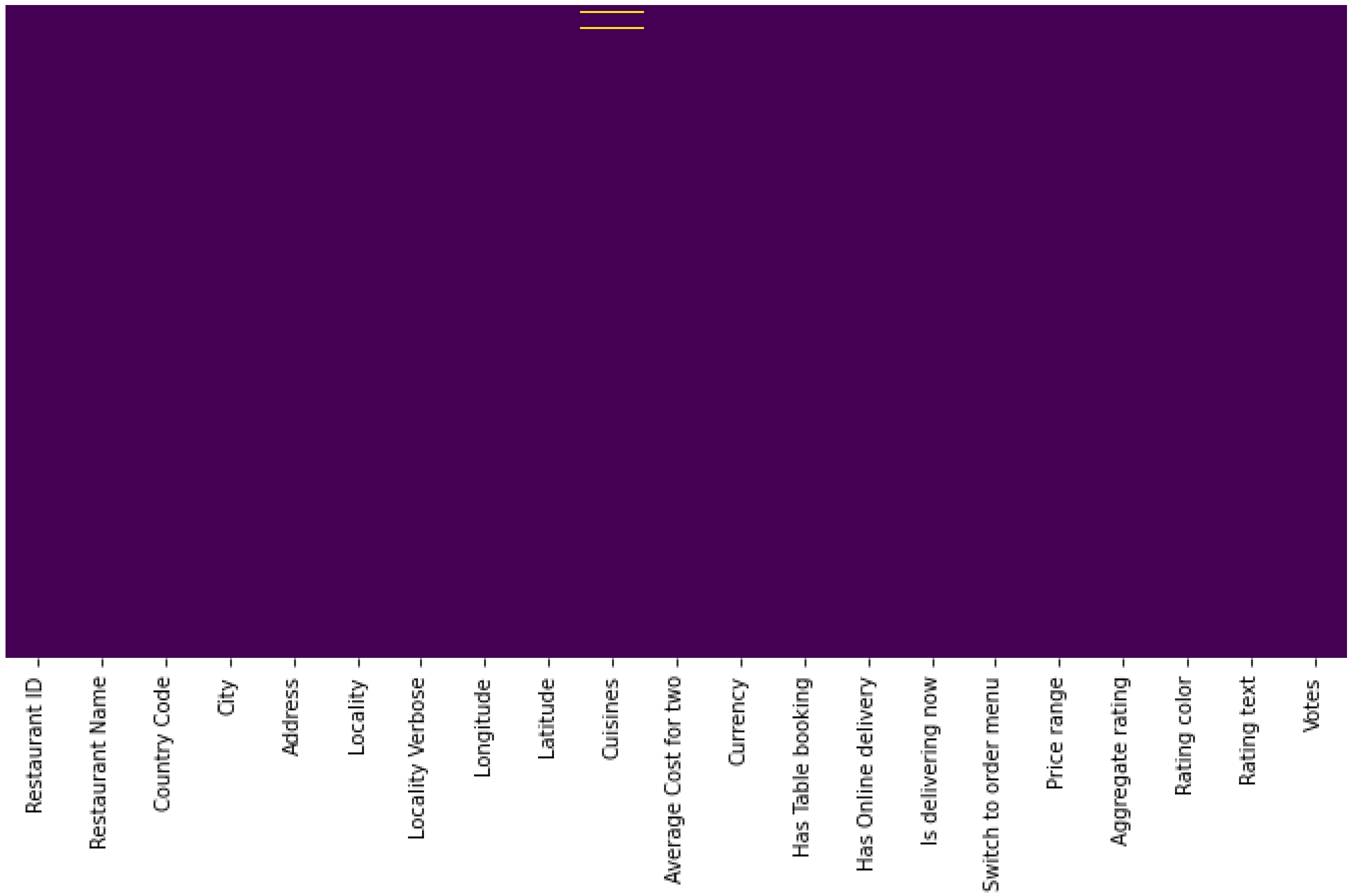
In [16]:
```
df.columns
```

Out[16]:
```
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')
```

Heatmap for zomato dataset

In [84]:
```
# 9551 rows are there so it didn't come properly.
plt.rcParams['figure.figsize'] = (12, 6)
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[84]:
```
<AxesSubplot:>
```



In [24]:
```
df.shape
```

Out[24]:
```
(9551, 21)
```

In [26]:
```
# Read Country-code dataset
df_country = pd.read_excel('/Users/madhu/Desktop/Data Science/iNeuron/EDA/country-code.xlsx')
df_country.head()
```

Out[26]:

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

In [30]:

```
# Since we have country-code column in both the dataset(i.e zomato and country-code) also hence try to combine both the dataset
df.columns
```

Out[30]:

```
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')
```

Merge both the dataset

In [35]:

```
final_df = pd.merge(df, df_country, on = 'Country Code', how = 'left'  )
final_df.head(2)
```

Out[35]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Has Table booking	Has Online delivery	Is delivering now	Switch to order menu	Price range	A
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	No	3	
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	No	3	

2 rows × 22 columns

In [37]:

```
# Check the data type
final_df.dtypes
```

Out[37]:

```
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 [38]:
```
final_df.columns
```

Out[38]:
```
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')
```

Find out how many countries are there and w.r.t country how many records are there?

In [43]:
```
# Observation:- Zomato is mainly in India
final_df.Country.value_counts()
```

Out[43]:
```
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
```

Get all the clountry name seperately

```
In [47]: country_names = final_df.Country.value_counts().index
         country_names
```

```
Out[47]: Index(['India', 'United States', 'United Kingdom', 'Brazil', 'UAE',
                'South Africa', 'New Zealand', 'Turkey', 'Australia', 'Phillipines',
                'Indonesia', 'Singapore', 'Qatar', 'Sri Lanka', 'Canada'],
               dtype='object')
```

Get all the values seperately

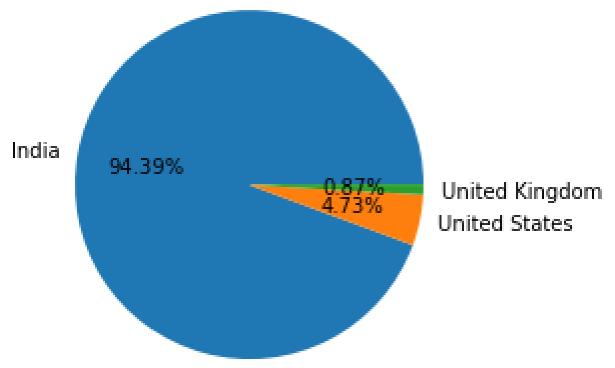
```
In [52]: country_val = final_df.Country.value_counts().values
         country_val
```

```
Out[52]: array([8652,  434,   80,   60,   60,   60,   40,   34,   24,   22,   21,
                  20,   20,   20,    4])
```

pie chart between Top 3 country_name and country_value

```
In [57]: # we use autopct= '%1.2f%%' to get the percentage
         plt.pie(country_val[:3], labels=country_names[:3], autopct= '%1.2f%%')
```

```
Out[57]: ([<matplotlib.patches.Wedge at 0x7fbae3c810d0>,
           <matplotlib.patches.Wedge at 0x7fbae3c8b700>,
           <matplotlib.patches.Wedge at 0x7fbae3c8bca0>],
          [Text(-1.0829742700952103, 0.19278674827836725, 'India'),
           Text(1.077281715838356, -0.22240527134123297, 'United States'),
           Text(1.0995865153823035, -0.030157837943120734, 'United Kingdom')],
          [Text(-0.590713238233751, 0.10515640815183668, '94.39%'),
           Text(0.5876082086391032, -0.12131196618612707, '4.73%'),
           Text(0.5997744629358018, -0.016449729787156763, '0.87%')])
```



Observation:Zomato maximum records or transaction are from India After that USA and then United Kingdoms

```
In [58]: final_df.columns
```

```
Out[58]: 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 [69]: # perform group by operation on the Aggregate Rating, Rating color, Rating text
         # reset_index() resets the index and converts it into a data frame.

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

```
In [70]: ratings
```

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
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

1. When Rating is between 4.5 to 4.9---> Excellent
2. When Rating are 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
6. When Reating is 0 ----> not rated

In [75]:

```
ratings.head()
```

Out[75]:

	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

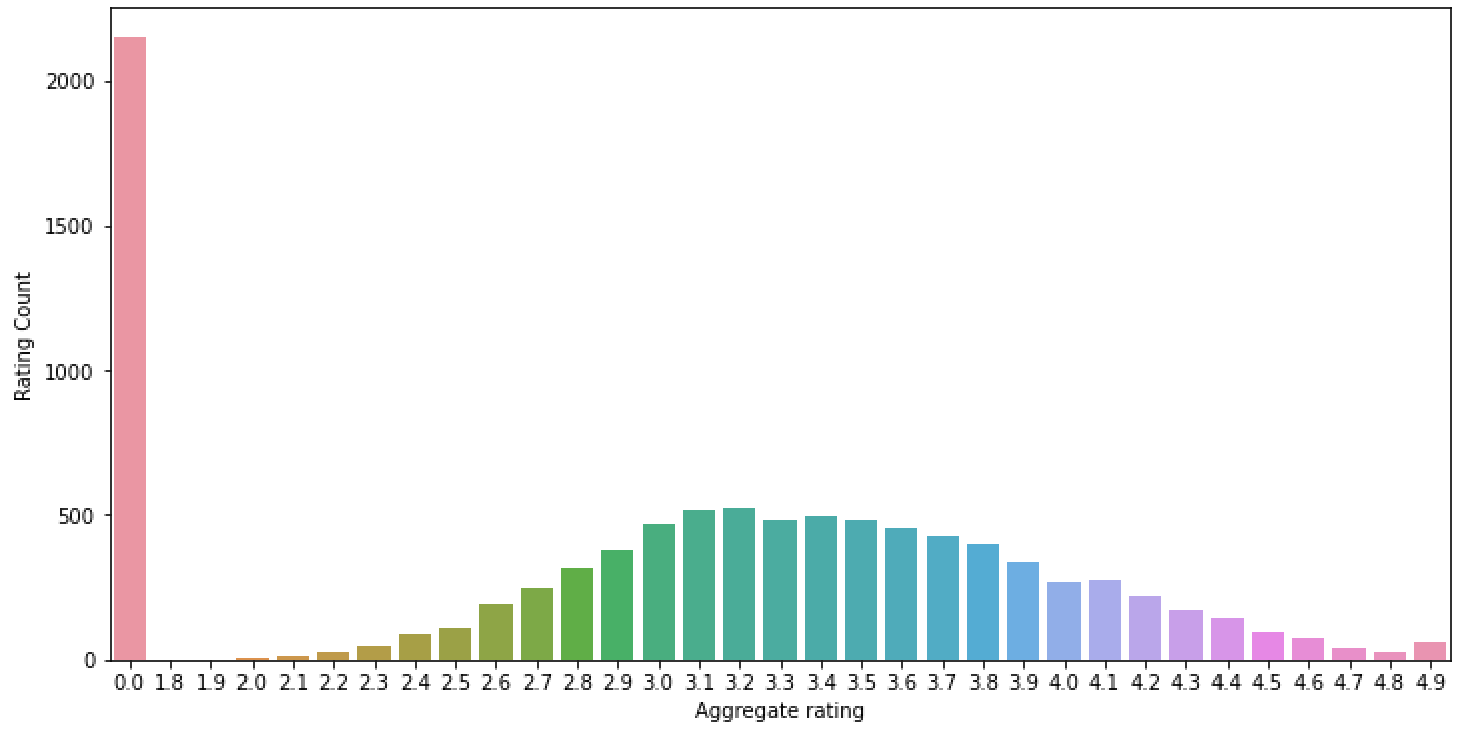
Bar plot

In [83]:

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

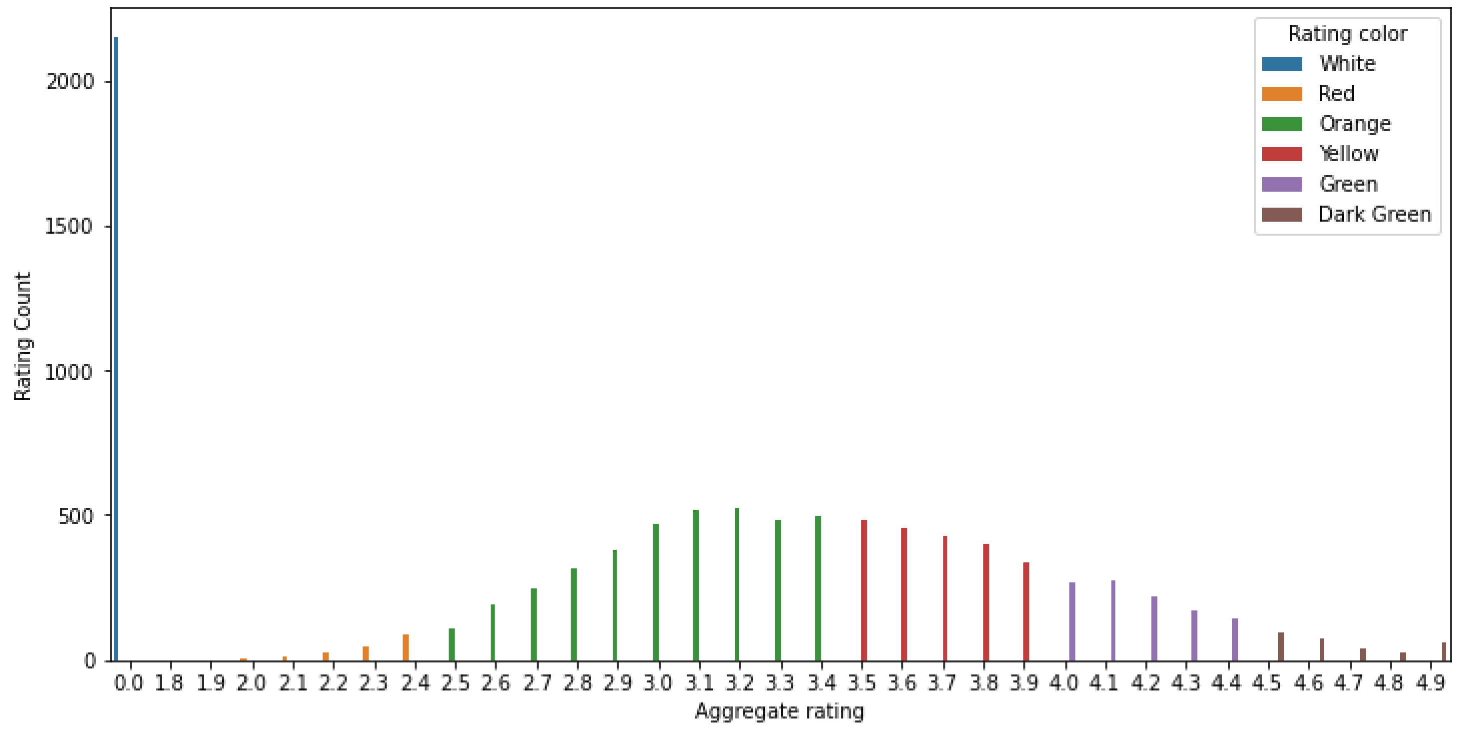
Out[83]:

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



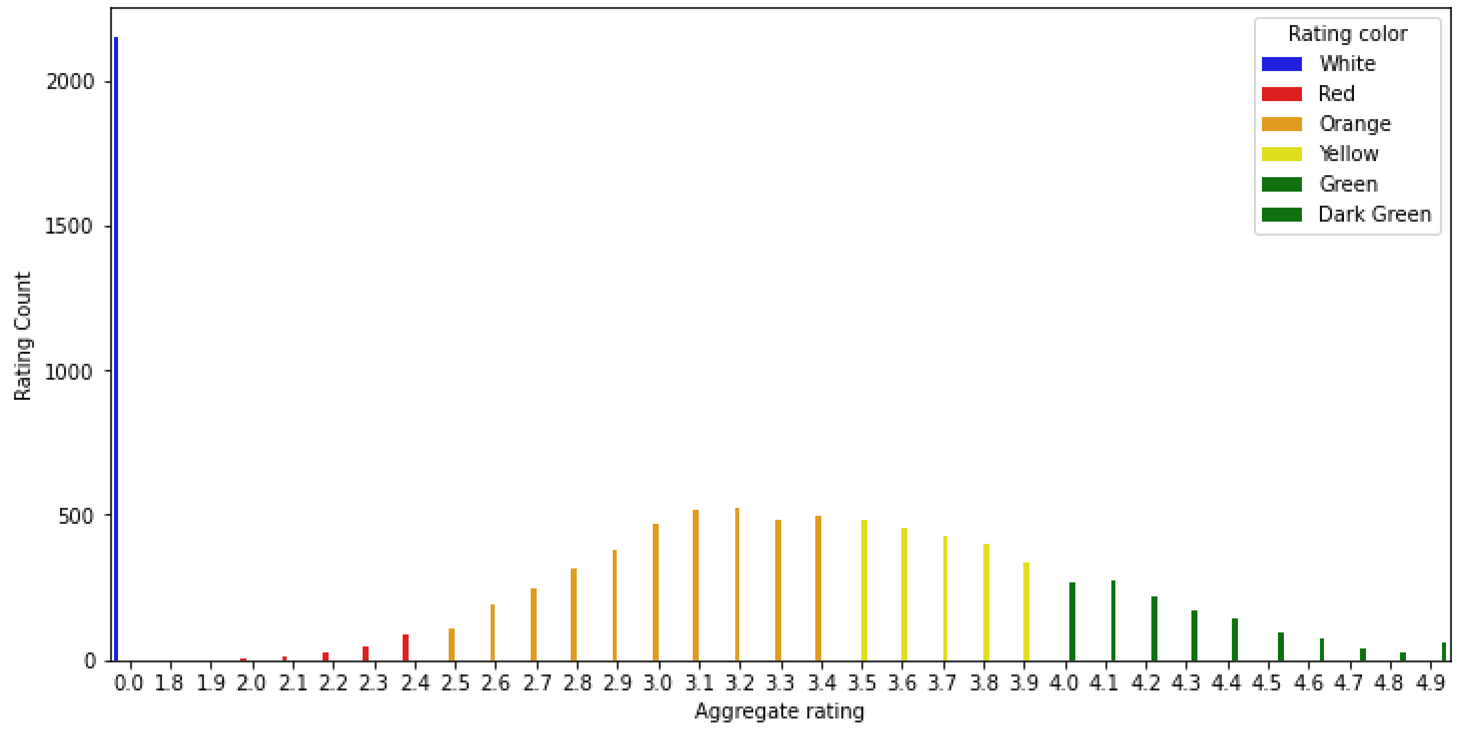
In [85]:
```
# hue = 'Rating color' It will catogrise it into different colors
plt.rcParams['figure.figsize'] = (12, 6)
sns.barplot(x = 'Aggregate rating', y = 'Rating Count', hue = 'Rating color', data = ratings)
```

Out[85]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>
```



```
In [106…
# Map the colours
sns.barplot(x = 'Aggregate rating', y = 'Rating Count', hue = 'Rating color', data= ratings, palette=['blue', 'Red', 'Orange',
```

```
Out[106]: <AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>
```



Observation:

1. Not Rated count is very high
2. Maximum number of rating are between 2.5 to 3.4

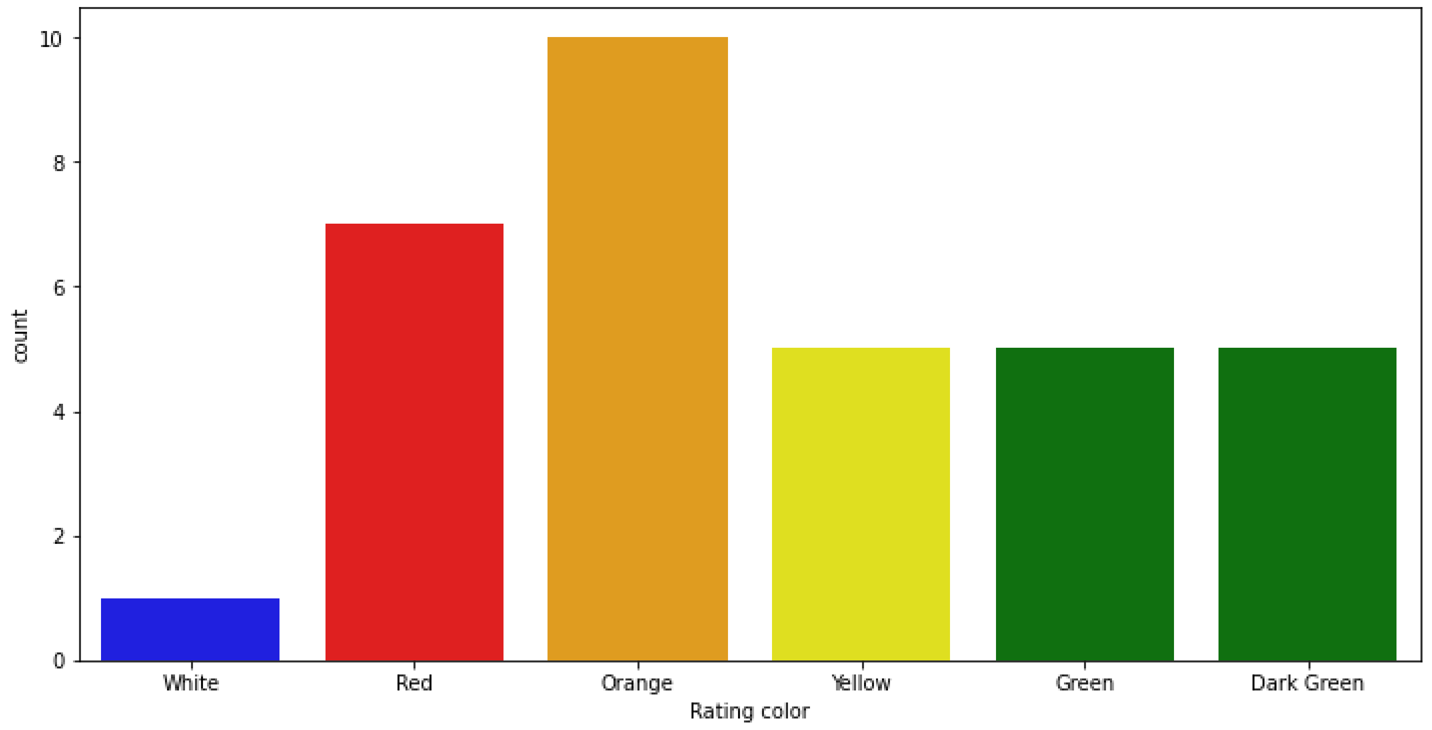
Count Plot

In [105…

```
# Basically we use it for plotting catogrical variables.
# On y axis we have frequency. white-1, red- 7, yellow-5 , orange-10 and so on..
sns.countplot(x = 'Rating color', data = ratings, palette =['blue', 'Red', 'Orange', 'Yellow', 'Green','Green'] )
```

Out[105]:

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



```
In [107…  ratings
```

Out[107]:

	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

Find the countries name that has given 0 rating

In [115…

```
final_df.columns
```

Out[115]:

```
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 [143…

```
final_df[final_df['Aggregate rating'] ==0].groupby('Country').size().reset_index()
```

Out[143]:

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

In [139…

```
final_df.groupby(['Aggregate rating','Country']).size().reset_index().head(5)
```

Out[139]:

	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

Observations Maximum number of 0 ratings are from Indian customers

find out which currency is used by which country?

In [146…

```
final_df.columns
```

Out[146]:

```
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 [150…

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

Out[150]:

	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 deliveries option??

In [170…
```
final_df.columns
```

Out[170]:
```
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 [176…
```
final_df[final_df['Has Online delivery'] =="Yes"].Country.value_counts()
```

Out[176]:
```
India    2423
UAE        28
Name: Country, dtype: int64
```

In [182…
```
final_df.groupby(['Has Online delivery','Country']).size().reset_index()
```

Out[182]:

	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

Observations:

Online Deliveries are available in India and UAE

Create a pie chart for top 5 cities distribution

In [183…
```
final_df.columns
```

Out[183]:
```
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')
```

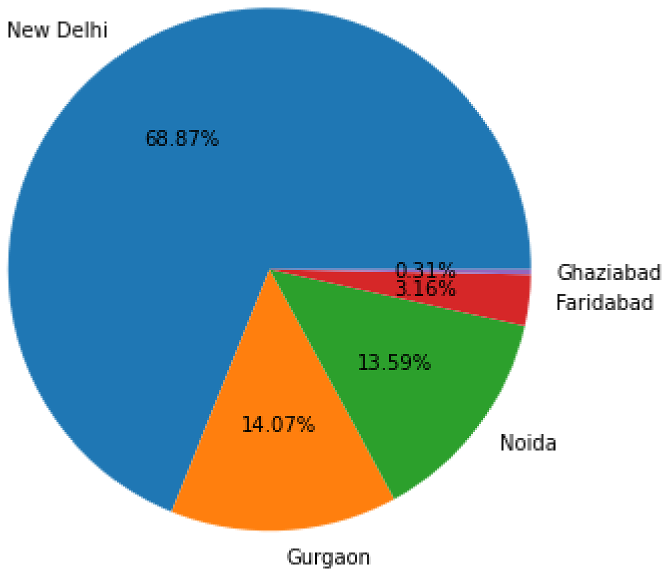
In [193…
```
city_labels = final_df['City'].value_counts().index
city_labels
```

Out[193]:
```
Index(['New Delhi', 'Gurgaon', 'Noida', 'Faridabad', 'Ghaziabad',
       'Bhubaneshwar', 'Amritsar', 'Ahmedabad', 'Lucknow', 'Guwahati',
       ...
       'Ojo Caliente', 'Montville', 'Monroe', 'Miller', 'Middleton Beach',
       'Panchkula', 'Mc Millan', 'Mayfield', 'Macedon', 'Vineland Station'],
      dtype='object', length=141)
```

In [206…
```
city_values = final_df['City'].value_counts().values
```

In [210…
```
plt.pie(city_values[:5], labels = city_labels[:5],autopct='%1.2f%%')
```

Out[210]:
```
([<matplotlib.patches.Wedge at 0x7fbac29b8be0>,
  <matplotlib.patches.Wedge at 0x7fbac29b8400>,
  <matplotlib.patches.Wedge at 0x7fbac1f34370>,
  <matplotlib.patches.Wedge at 0x7fbac1f34be0>,
  <matplotlib.patches.Wedge at 0x7fbac29bbdc0>],
 [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, -0.010871113182029922, '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, -0.005929698099289048, '0.31%')])
```



Find the top 10 cuisines

```
In [212…  final_df.columns

Out[212]: 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 [217…  final_df['Cuisines'].value_counts()[:10]

Out[217]: North Indian                        936
          North Indian, Chinese               511
          Chinese                             354
          Fast Food                           354
          North Indian, Mughlai               334
          Cafe                                299
          Bakery                              218
          North Indian, Mughlai, Chinese      197
          Bakery, Desserts                    170
          Street Food                         149
          Name: Cuisines, dtype: int64
```

```
In [ ]:
```