ZOMATO DATASET EXPLORATORY DATA ANALYSIS

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

```
Matplotlib is building the font cache; this may take a moment.
```

In [4]:
```
df = pd.read_csv("C:\\Users\\Sonali Thakur.DESKTOP-T4FPVTD.000\\Downloads\\zomatc
df.head()
```

Out[4]:

City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency	H Tab booki
kati 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)	Y
kati 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)	Y
aluyong 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)	Y
aluyong 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)	N
aluyong 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)	Y

In [5]:
```
df.columns
```

Out[5]:
```
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 [6]:
```
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
```

- here int64 basically means that for integer variable
- objects in pandas means strings and it can also be categorical variable or a text variable or may be integer variable

In [7]:
```
df.describe()
```

Out[7]:

	Restaurant ID	Country Code	Longitude	Latitude	Average Cost for two	Price range	Aggreg rat
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000	9551.000000	9551.000
mean	9.051128e+06	18.365616	64.126574	25.854381	1199.210763	1.804837	2.666
std	8.791521e+06	56.750546	41.467058	11.007935	16121.183073	0.905609	1.516
min	5.300000e+01	1.000000	-157.948486	-41.330428	0.000000	1.000000	0.000
25%	3.019625e+05	1.000000	77.081343	28.478713	250.000000	1.000000	2.500
50%	6.004089e+06	1.000000	77.191964	28.570469	400.000000	2.000000	3.200
75%	1.835229e+07	1.000000	77.282006	28.642758	700.000000	2.000000	3.700
max	1.850065e+07	216.000000	174.832089	55.976980	800000.000000	4.000000	4.900

- the features which are taken inside the describe function are integer features

IN DATA ANALYSIS THE THINGS WE KNOW

1. Missing Values
2. Explore about numerical variables
3. Explore about categorical variables
4. Finding relationship between features.

- IN ORDER TO FIND MISSING VALUES WE DO

In [8]:
```
df.isnull().sum()
```

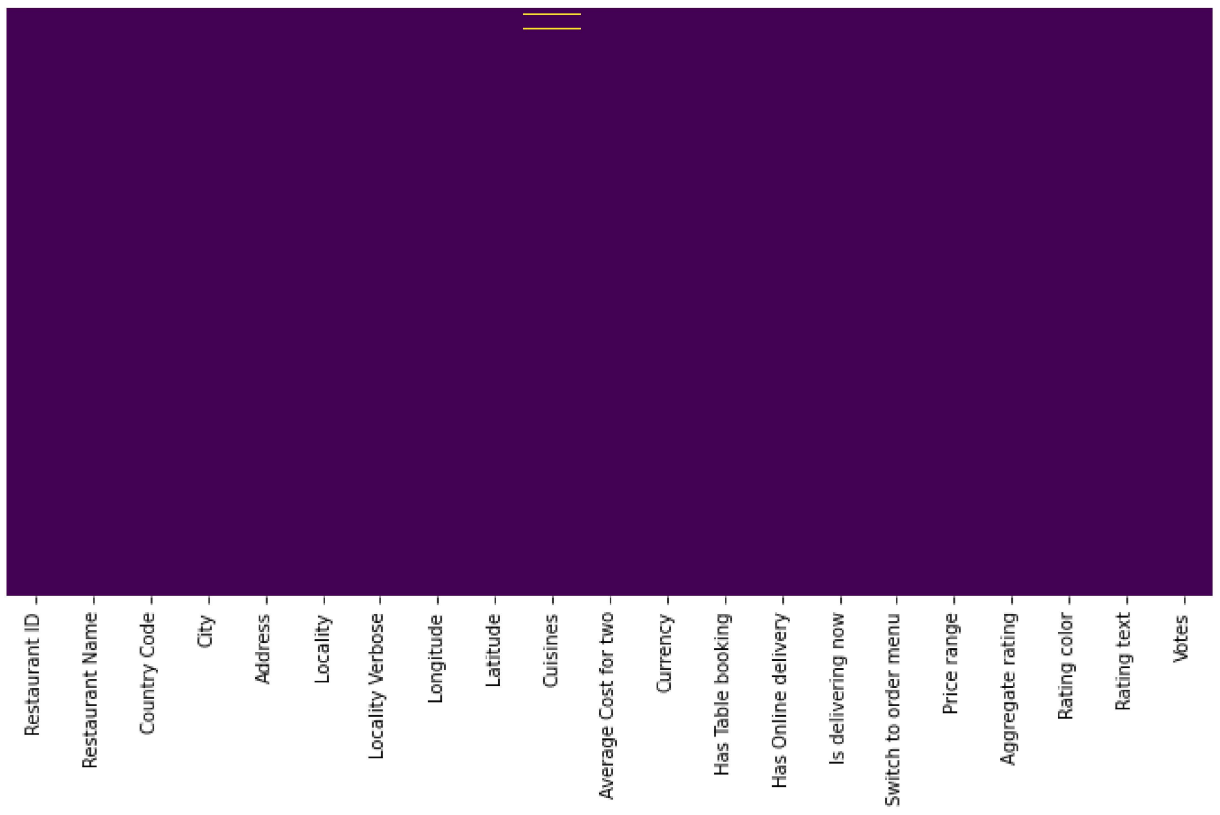
Out[8]:
```
Restaurant ID           0
Restaurant Name         0
Country Code            0
City                    0
Address                 0
Locality                0
Locality Verbose        0
Longitude               0
Latitude                0
Cuisines                9
Average Cost for two    0
Currency                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 [9]:
```
[features for features in df.columns if df[features].isnull().sum()>0]
```

Out[9]:
```
['Cuisines']
```

In [47]:
```
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[47]:
```
<AxesSubplot:>
```



In [12]:
```
df_country = pd.read_excel('C:\\Users\\Sonali Thakur.DESKTOP-T4FPVTD.000\\Downloa
df_country.head()
```

Out[12]:

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

```
In [15]: df.columns
```

```
Out[15]: 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 df_country ALONG WITH df WITH THIS COUNTRYCODE

```
In [18]: final_df = pd.merge(df,df_country,on='Country Code',how='left')
         final_df.head(2)
```

Out[18]:

Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Has Table booking	Has Online delivery	Is delivering now
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
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

In [19]:
```
## TO CHECK DATA TYPES

final_df.dtypes
```

Out[19]:
```
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 [20]:
```
final_df.columns
```

Out[20]:
```
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 [26]:
```
country_names = final_df.Country.value_counts().index
country_names
```

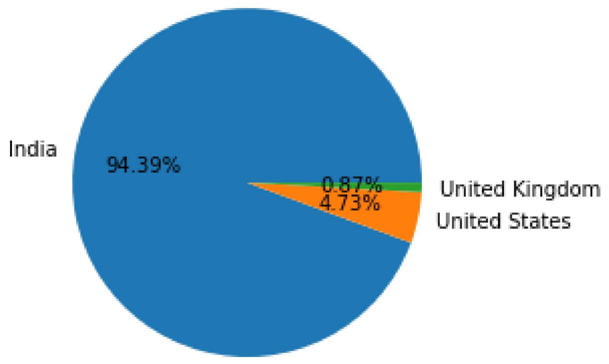
Out[26]:
```
Index(['India', 'United States', 'United Kingdom', 'Brazil', 'UAE',
       'South Africa', 'New Zealand', 'Turkey', 'Australia', 'Phillipines',
       'Indonesia', 'Singapore', 'Qatar', 'Sri Lanka', 'Canada'],
      dtype='object')
```

In [28]:
```
country_val=final_df.Country.value_counts().values
country_val
```

Out[28]:
```
array([8652,  434,   80,   60,   60,   60,   40,   34,   24,   22,   21,
         20,   20,   20,    4], dtype=int64)
```

In [34]:
```
## Pie Chart - Top 3 countries that uses zomato

plt.pie(country_val[:3],labels=country_names[:3],autopct='%1.2f%%')
```

Out[34]:
```
([<matplotlib.patches.Wedge at 0x2616bbe1b20>,
  <matplotlib.patches.Wedge at 0x2616bbec280>,
  <matplotlib.patches.Wedge at 0x2616bbec9a0>],
 [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

1. ZOMATO's MAXIMUM TRANSCATIONS ARE FROM INDIA , AFTER THAT U.S.A.
2. THE COMPANY IS GETTING MORE PROFIT FROM INDIA
3. MAJOR BUSINESS IS HAPPENING IN INDIA

In [35]:
```
final_df.columns
```

Out[35]:
```
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 [43]:
```
tings=final_df.groupby(['Aggregate rating','Rating color','Rating text']).size().r
```

In [44]:
```
ratings
```

Out[44]:

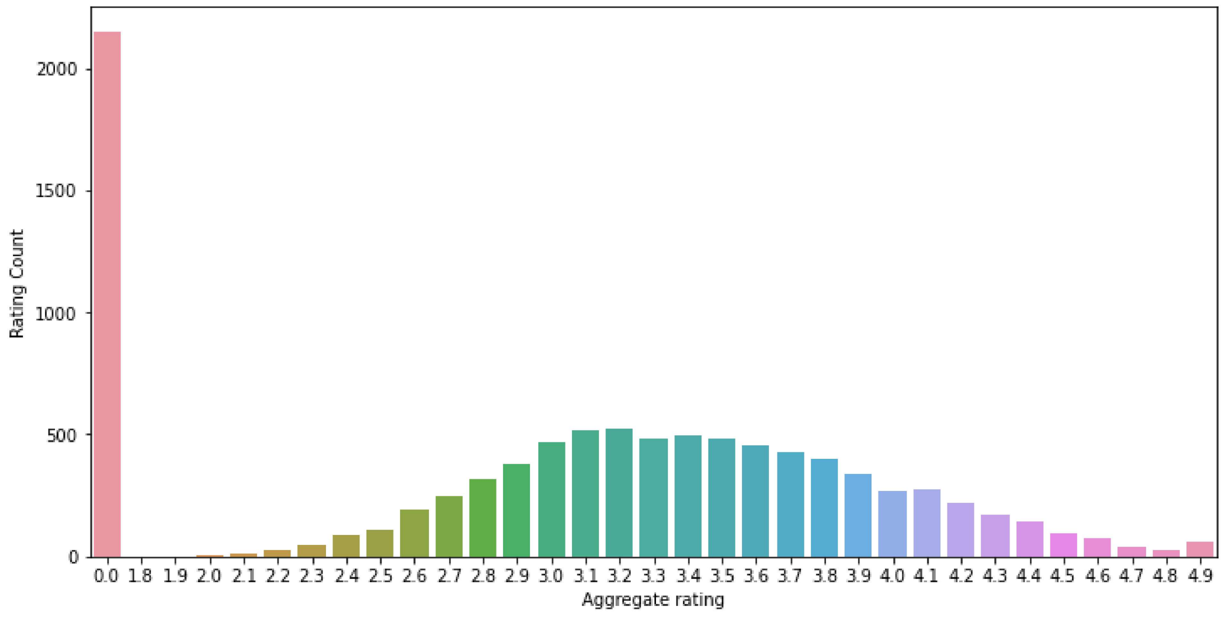
	Aggregate rating	Rating color	Rating text	Rating Count
0	0.0	White	Not rated	2148
1	1.8	Red	Poor	1
2	1.9	Red	Poor	2
3	2.0	Red	Poor	7
4	2.1	Red	Poor	15
5	2.2	Red	Poor	27
6	2.3	Red	Poor	47
7	2.4	Red	Poor	87
8	2.5	Orange	Average	110
9	2.6	Orange	Average	191
10	2.7	Orange	Average	250
11	2.8	Orange	Average	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 FROM 4.5 TO 4.9 THE RATING WERE EXCELLENT
2. WHEN RATING IS BETWEEN FROM 4.0 TO 4.4 THE RATING WERE VERY GOOD
3. WHEN RATING IS BETWEEN FROM 3.5 TO 3.9 THE RATING WERE GOOD
4. WHEN RATING IS BETWEEN FROM 2.5 TO 3.4 THE RATING WERE AVERAGE
5. WHEN RATING IS BETWEEN FROM 1.8 TO 2.4 THE RATING WERE POOR
6. WHEN RATING IS 0 IT MEANS PEOPLE HAS NOT GIVEN ANY RATING

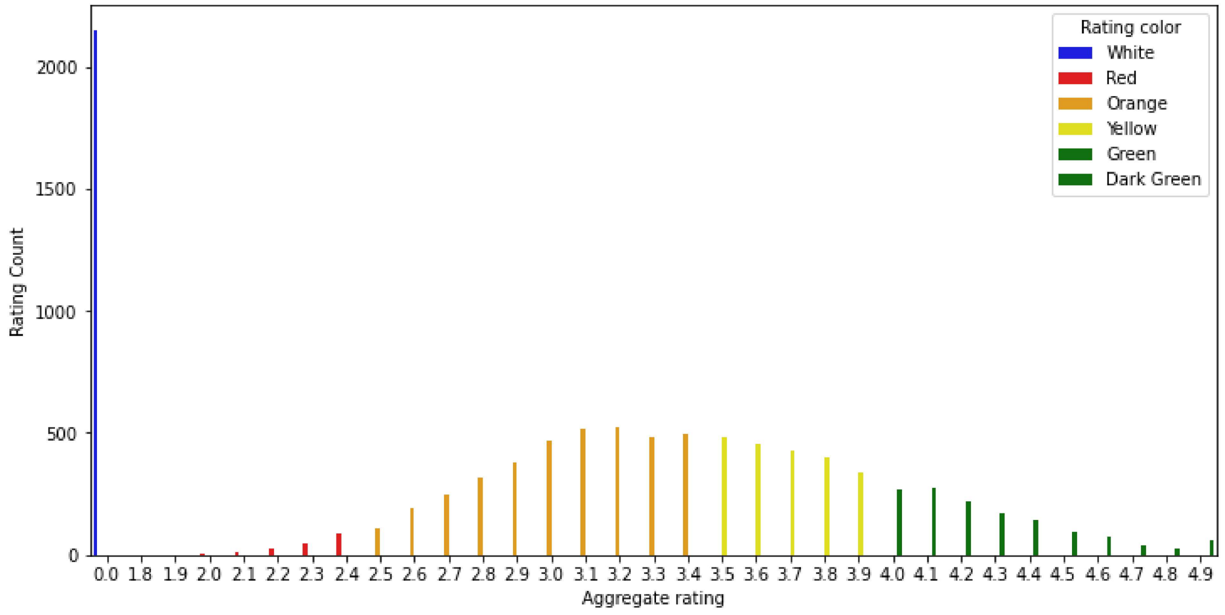
In [46]:
```
import matplotlib
matplotlib.rcParams['figure.figsize']=(12,6)

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

Out[46]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>
```



In [51]:
```
sns.barplot(x="Aggregate rating",y="Rating Count",hue='Rating color',data=ratings
```

Out[51]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating Count'>
```

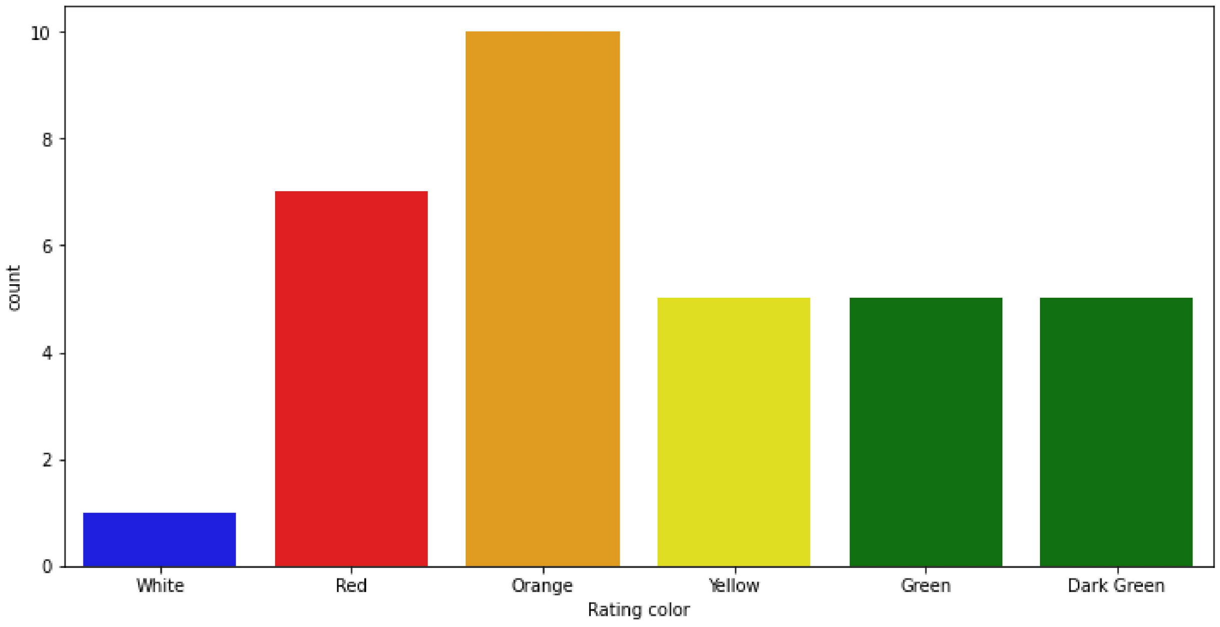


OBSERVATION

1. NOT RATED COUNT IS VERY HIGH
2. MAXIMUM NUMBER OF RATINGS ARE BETWEEN 2.5 TO 3.4

In [52]:
```
## COUNT PLOT

sns.countplot(x="Rating color",data=ratings,palette=['blue','red','orange','yello
```

Out[52]:
```
<AxesSubplot:xlabel='Rating color', ylabel='count'>
```



FIND THE COUNTRIES NAME THAT HAS GIVEN ZERO RATING

In [73]:
```
final_df.groupby(["Aggregate rating","Country"]).size().reset_index().head(4)
```

Out[73]:

	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

In [82]:
```
final_df[final_df["Aggregate rating"]==0].Country.value_counts()
```

Out[82]:
```
India             2139
Brazil               5
United States        3
United Kingdom       1
Name: Country, dtype: int64
```

OBSERVATION

MAXIMUM NUMBER OF ZERO RATING ARE FROM INDIA

FIND OUT WHICH CURRENCY IS USED BY WHICH COUNTRY

In [75]:
```
final_df.columns
```

Out[75]:
```
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 [76]:
```
final_df.groupby(["Country","Currency"]).size().reset_index()
```

Out[76]:

	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 DELIEVERY OPTIONS

In [77]:
```
final_df.columns
```

Out[77]:
```
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 [78]:
```
final_df.groupby(["Country","Has Online delivery"]).size().reset_index()
```

Out[78]:

	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

In [79]:
```
final_df[final_df['Has Online delivery']=='Yes'].Country.value_counts()
```

Out[79]:
```
India    2423
UAE        28
Name: Country, dtype: int64
```

OBSEVATIONS

1. ONLINE DELIVERIES ARE AVAILABLE IN INDIA AND UAE

CREATE A PIE CHART FOR TOP 5 CITIES DISTRIBUTION

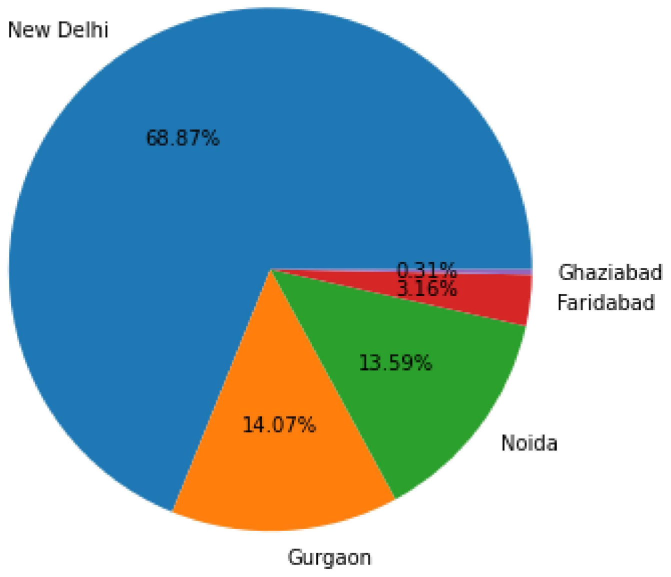
In [83]:
```
final_df.columns
```

Out[83]:
```
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]:
```
city_values = final_df.City.value_counts().values
city_labels = final_df.City.value_counts().index
```

In [90]:
```
plt.pie(city_values[:5],labels=city_labels[:5],autopct='%1.2f%%')
```

Out[90]:
```
([<matplotlib.patches.Wedge at 0x26173713e80>,
  <matplotlib.patches.Wedge at 0x2617371b550>,
  <matplotlib.patches.Wedge at 0x2617371bc70>,
  <matplotlib.patches.Wedge at 0x261737273d0>,
  <matplotlib.patches.Wedge at 0x26173727af0>],
 [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%')])
```

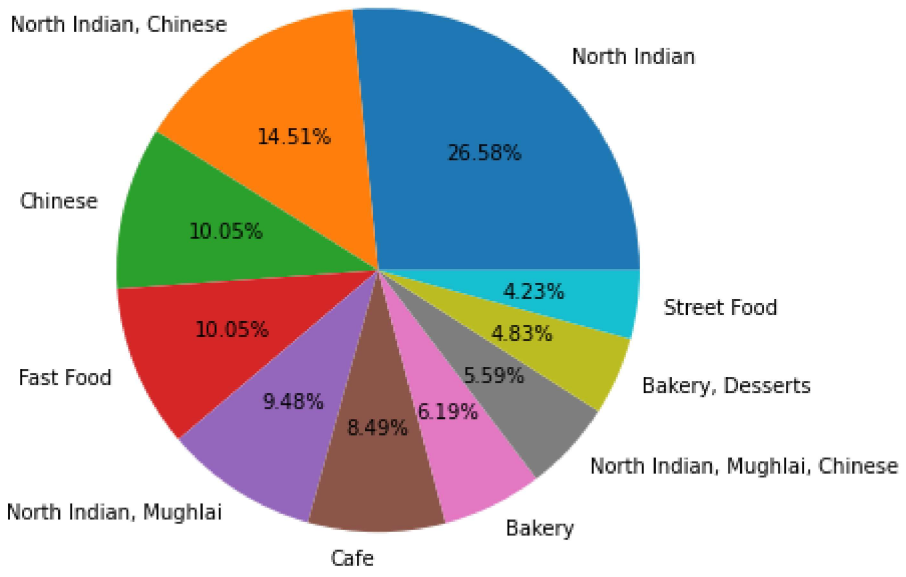


FIND THE TOP 10 CUISINES

In [100]:
```
cuisines_values = final_df.Cuisines.value_counts().values
cuisines_labels = final_df.Cuisines.value_counts().index
```

In [102]:
```
plt.pie(cuisines_values[:10],labels=cuisines_labels[:10],autopct='%1.2f%%')
```

Out[102]:
```
([<matplotlib.patches.Wedge at 0x26173946d60>,
  <matplotlib.patches.Wedge at 0x261739484c0>,
  <matplotlib.patches.Wedge at 0x26173948be0>,
  <matplotlib.patches.Wedge at 0x26173912340>,
  <matplotlib.patches.Wedge at 0x26173912a60>,
  <matplotlib.patches.Wedge at 0x2617396d1c0>,
  <matplotlib.patches.Wedge at 0x2617396d8e0>,
  <matplotlib.patches.Wedge at 0x26173966040>,
  <matplotlib.patches.Wedge at 0x26173966760>,
  <matplotlib.patches.Wedge at 0x26173966e80>],
 [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, Chines
e'),
  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

1. THE MOST DEMANDED CUISINES IS NORTH INDIAN.
2. THE SECOND MOST DEMANDED CUISINES IS NORTH INDIAN,CHINESE.

```
In [ ]:
```