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

In [2]:
```
df=pd.read_csv("zomato.csv",encoding="latin-1")
```

In [3]:
```
df.head()
```

Out[3]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency	Has Table booking	Has Online delivery	Is delivering now	Switch to order menu	Price range	Aggregate rating	Rating color	Rating text	Votes
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	No	No	3	4.8	Dark Green	Excellent	314
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	No	No	3	4.5	Dark Green	Excellent	591
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	No	No	4	4.4	Green	Very Good	270
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	No	No	4	4.9	Dark Green	Excellent	365
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	No	No	4	4.8	Dark Green	Excellent	229

5 rows × 21 columns

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.shape
```

Out[5]:
```
(9551, 21)
```

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
```

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

Out[7]:

	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

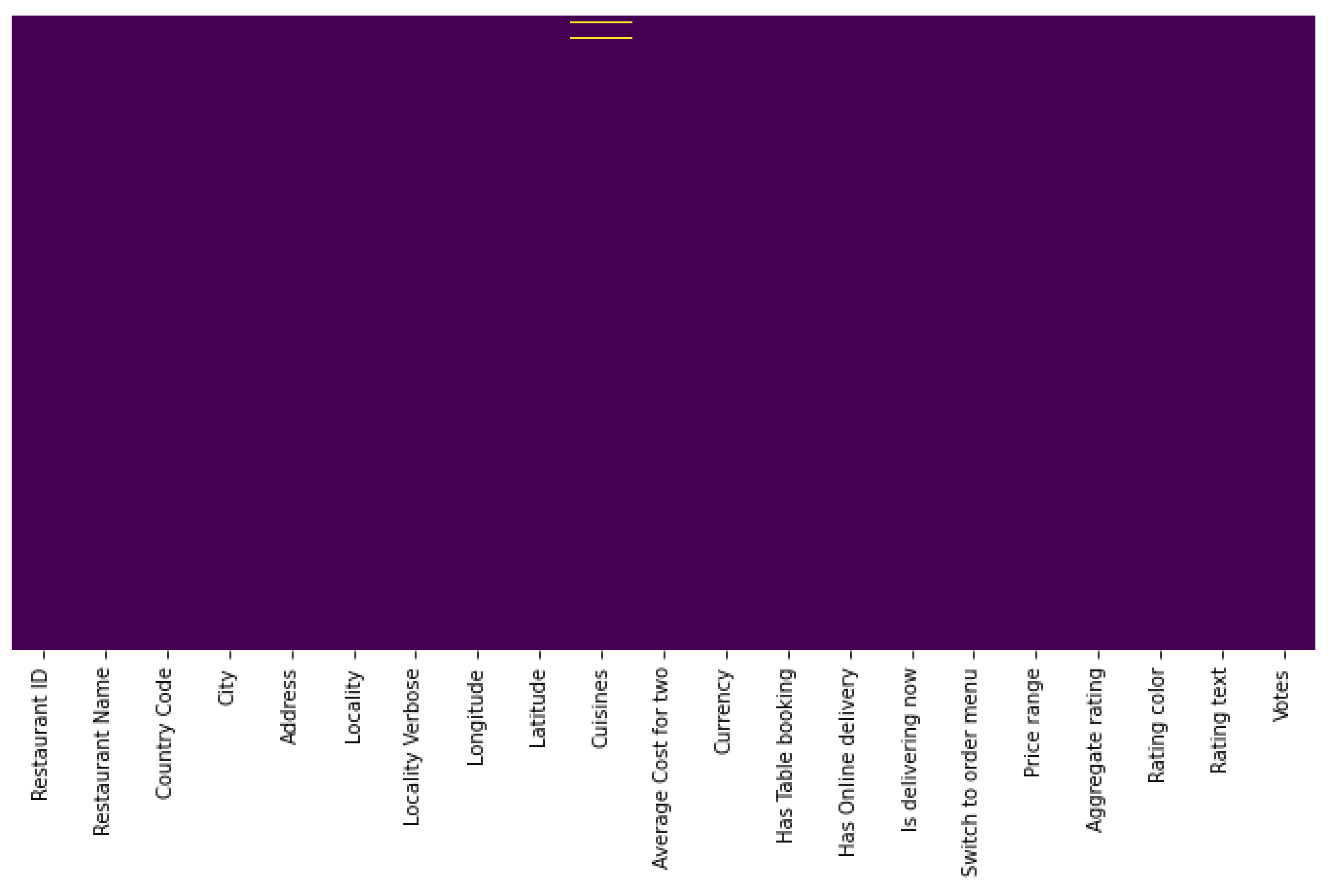
1. Missing Values
2. Explore About the Numerical Variables
3. Explore About categorical Variables
4. Finding Relationship between features

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

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency	Has Table booking	Has Online delivery	Is delivering now	Switch to order menu	Price range	Aggregate rating	Rating color	Rating text	Votes
0	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9546	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
9547	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
9548	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
9549	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False
9550	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	False	False	False	False	False	False

9551 rows × 21 columns

Ways to see null values

In [9]:
```
df.isnull().sum()
```
Out[9]:
```
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 [10]:
```
[features for features in df.columns if df[features].isnull().sum()>0]
#for every features in df.columns check the condition.
#random variable in features is iterated throughout the column
```
Out[10]:
```
['Cuisines']
```
In [11]:
```
matplotlib.rcParams['figure.figsize']=(12,6)
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis') #small value so cant see
```
Out[11]:
```
<AxesSubplot:>
```


In [12]:
```
df_country=pd.read_excel("Country-code.xlsx")
```
In [13]:
```
df_country.head()
```
Out[13]:

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

In [14]:
```
final_df=pd.merge(df,df_country,on='Country Code', how="left" )
#combining 2 dataframes
#on=On which feature we combine the 2 tables
#how=what type of join
```

In [15]:
```
final_df.head()
```
Out[15]:

	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	Aggregate rating	Rating color	Rating text	Votes	Country
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	4.8	Dark Green	Excellent	314	Phillipines
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	4.5	Dark Green	Excellent	591	Phillipines
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	...	Yes	No	No	No	4	4.4	Green	Very Good	270	Phillipines
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	...	No	No	No	No	4	4.9	Dark Green	Excellent	365	Phillipines
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	...	Yes	No	No	No	4	4.8	Dark Green	Excellent	229	Phillipines

5 rows × 22 columns

In [16]:
```
final_df.dtypes
```
Out[16]:
```
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
```

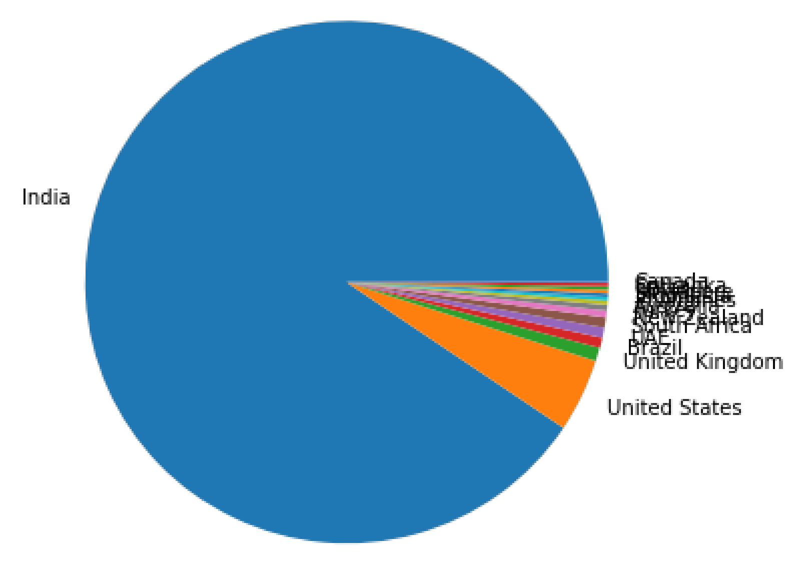
In [17]:
```
final_df.Country.value_counts()
```
Out[17]:
```
India             8652
United States      434
United Kingdom      80
Brazil              60
UAE                 60
South Africa        60
New Zealand         40
Turkey              34
Australia           24
Phillipines         22
Indonesia           21
Singapore           20
Qatar               20
Sri Lanka           20
Canada               4
Name: Country, dtype: int64
```

In [18]:
```
country_names=final_df.Country.value_counts().index
country_names
```
Out[18]:
```
Index(['India', 'United States', 'United Kingdom', 'Brazil', 'UAE',
       'South Africa', 'New Zealand', 'Turkey', 'Australia', 'Phillipines',
       'Indonesia', 'Singapore', 'Qatar', 'Sri Lanka', 'Canada'],
      dtype='object')
```

In [19]:
```
country_val=final_df.Country.value_counts().values
country_val
```
Out[19]:
```
array([8652,  434,   80,   60,   60,   60,   40,   34,   24,   22,   21,
         20,   20,   20,    4], dtype=int64)
```

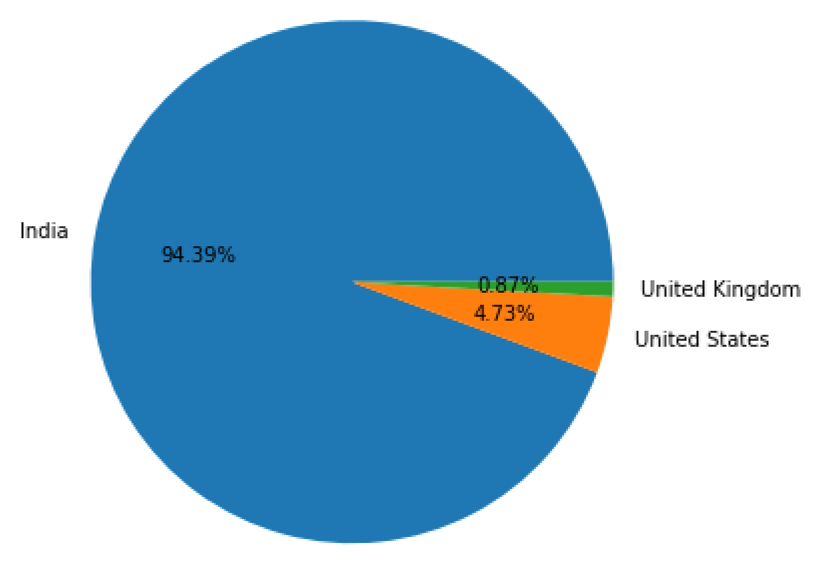
In [20]:
```
plt.pie(country_val,labels=country_names)
```

Out[20]:
```
([<matplotlib.patches.Wedge at 0xe274faa10>,
  <matplotlib.patches.Wedge at 0xe274faf20>,
  <matplotlib.patches.Wedge at 0xe274fb400>,
  <matplotlib.patches.Wedge at 0xe274fb8e0>,
  <matplotlib.patches.Wedge at 0xe274fbdc0>,
  <matplotlib.patches.Wedge at 0xe275242e0>,
  <matplotlib.patches.Wedge at 0xe275247c0>,
  <matplotlib.patches.Wedge at 0xe27524ca0>,
  <matplotlib.patches.Wedge at 0xe27525180>,
  <matplotlib.patches.Wedge at 0xe27525660>,
  <matplotlib.patches.Wedge at 0xe274fa9e0>,
  <matplotlib.patches.Wedge at 0xe27525ff0>,
  <matplotlib.patches.Wedge at 0xe275264d0>,
  <matplotlib.patches.Wedge at 0xe275269b0>,
  <matplotlib.patches.Wedge at 0xe27526e90>],
 [Text(-1.052256163793291, 0.3205572737577906, 'India'),
  Text(0.9911329812843455, -0.477132490415823, 'United States'),
  Text(1.0572858296119743, -0.3035567072257165, 'United Kingdom'),
  Text(1.070138816916019, -0.2545641619112621, 'Brazil'),
  Text(1.0793506814479759, -0.21213699926648824, 'UAE'),
  Text(1.086881147244973, -0.16937937230799818, 'South Africa'),
  Text(1.0918635911832035, -0.1335436192729486, 'New Zealand'),
  Text(1.0947903814016446, -0.10692998078388304, 'Turkey'),
  Text(1.096631023945382, -0.08602556201794338, 'Australia'),
  Text(1.0978070729776455, -0.06942355882735218, 'Phillipines'),
  Text(1.0986791544015209, -0.05388984768543213, 'Indonesia'),
  Text(1.0993059848742366, -0.039068550263413035, 'Singapore'),
  Text(1.0997248508282123, -0.02460187941736628, 'Qatar'),
  Text(1.0999533462179636, -0.010130949802716446, 'Sri Lanka'),
  Text(1.0999990477553414, -0.0014473898376707638, 'Canada')])
```



In [21]:
```
## Pie Chart- Top 3 countries that uses zomato
plt.pie(country_val[:3],labels=country_names[:3],autopct='%1.2f%%')
```

Out[21]:
```
([<matplotlib.patches.Wedge at 0xe27591150>,
  <matplotlib.patches.Wedge at 0xe27591870>,
  <matplotlib.patches.Wedge at 0xe27591f90>],
 [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 maximum records or transaction are from India After that USA and then United Kingdoms

In [22]:
```
final_df.columns
```

Out[22]:
```
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 [23]:
```
final_df.groupby(['Aggregate rating','Rating color','Rating text'])
#This is dataframe groupby object
```

Out[23]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000000E2749BD30>
```

In [24]:
```
final_df.groupby(['Aggregate rating','Rating color','Rating text']).size()
```
Out[24]:
```
Aggregate rating  Rating color  Rating text
0.0               White         Not rated      2148
1.8               Red           Poor              1
1.9               Red           Poor              2
2.0               Red           Poor              7
2.1               Red           Poor             15
2.2               Red           Poor             27
2.3               Red           Poor             47
2.4               Red           Poor             87
2.5               Orange        Average         110
2.6               Orange        Average         191
2.7               Orange        Average         250
2.8               Orange        Average         315
2.9               Orange        Average         381
3.0               Orange        Average         468
3.1               Orange        Average         519
3.2               Orange        Average         522
3.3               Orange        Average         483
3.4               Orange        Average         498
3.5               Yellow        Good            480
3.6               Yellow        Good            458
3.7               Yellow        Good            427
3.8               Yellow        Good            400
3.9               Yellow        Good            335
4.0               Green         Very Good       266
4.1               Green         Very Good       274
4.2               Green         Very Good       221
4.3               Green         Very Good       174
4.4               Green         Very Good       144
4.5               Dark Green    Excellent        95
4.6               Dark Green    Excellent        78
4.7               Dark Green    Excellent        42
4.8               Dark Green    Excellent        25
4.9               Dark Green    Excellent        61
dtype: int64
```

In [25]:
```
final_df.groupby(['Aggregate rating','Rating color','Rating text']).size().reset_index()
```
Out[25]:

	Aggregate rating	Rating color	Rating text	0
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 [26]:
```python
ratings=final_df.groupby(['Aggregate rating','Rating color','Rating text']).size().reset_index().rename(columns={0:'Rating_count'})
#reset_index=resetting index & converting to dataframe.
ratings
```

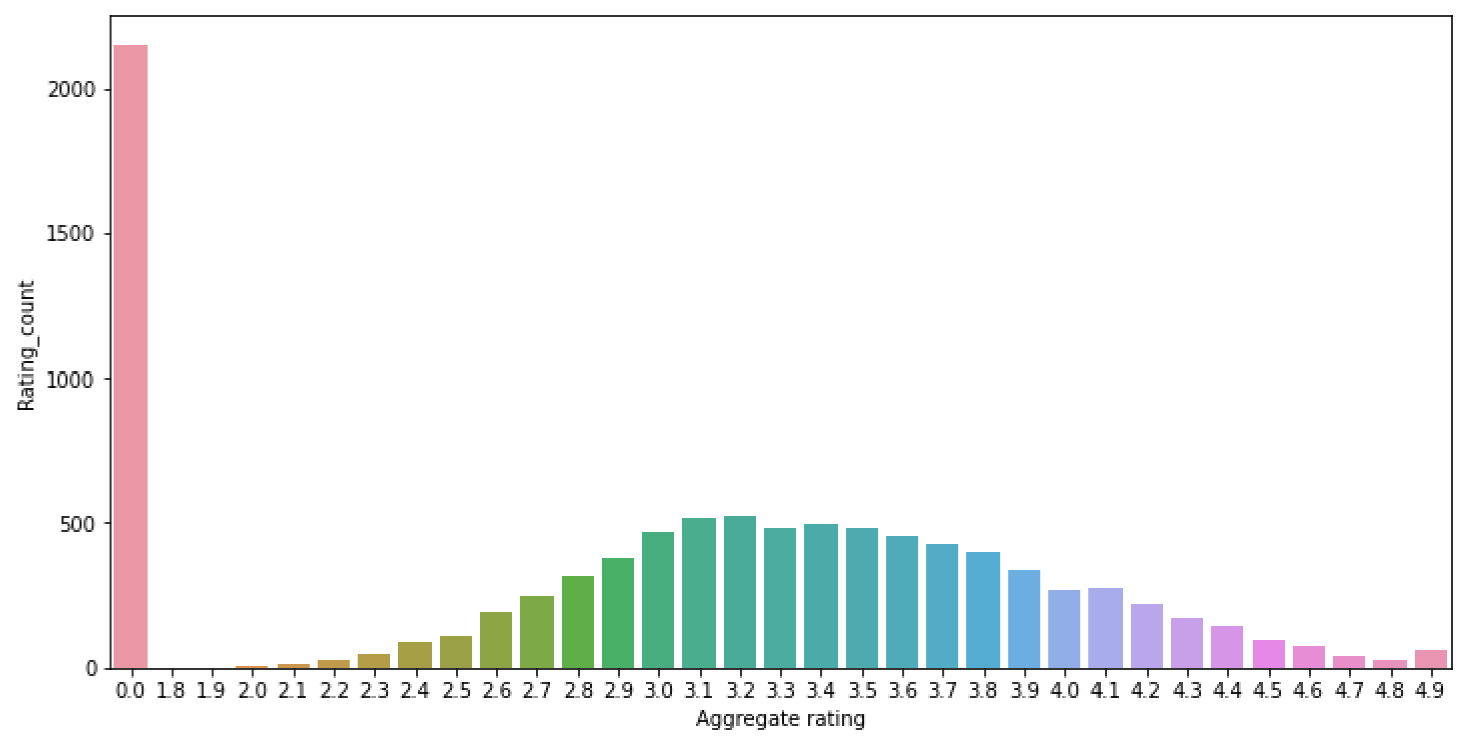
Out[26]:

	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 Rating is between 4.5 to 4.9---> Excellent
When Rating are between 4.0 to 3.4--->very good
when Rating is between 3.5 to 3.9----> good
when Rating is between 3.0 to 3.4----> average
when Rating is between 2.5 to 2.9----> average
when Rating is between 2.0 to 2.4----> Poor
```

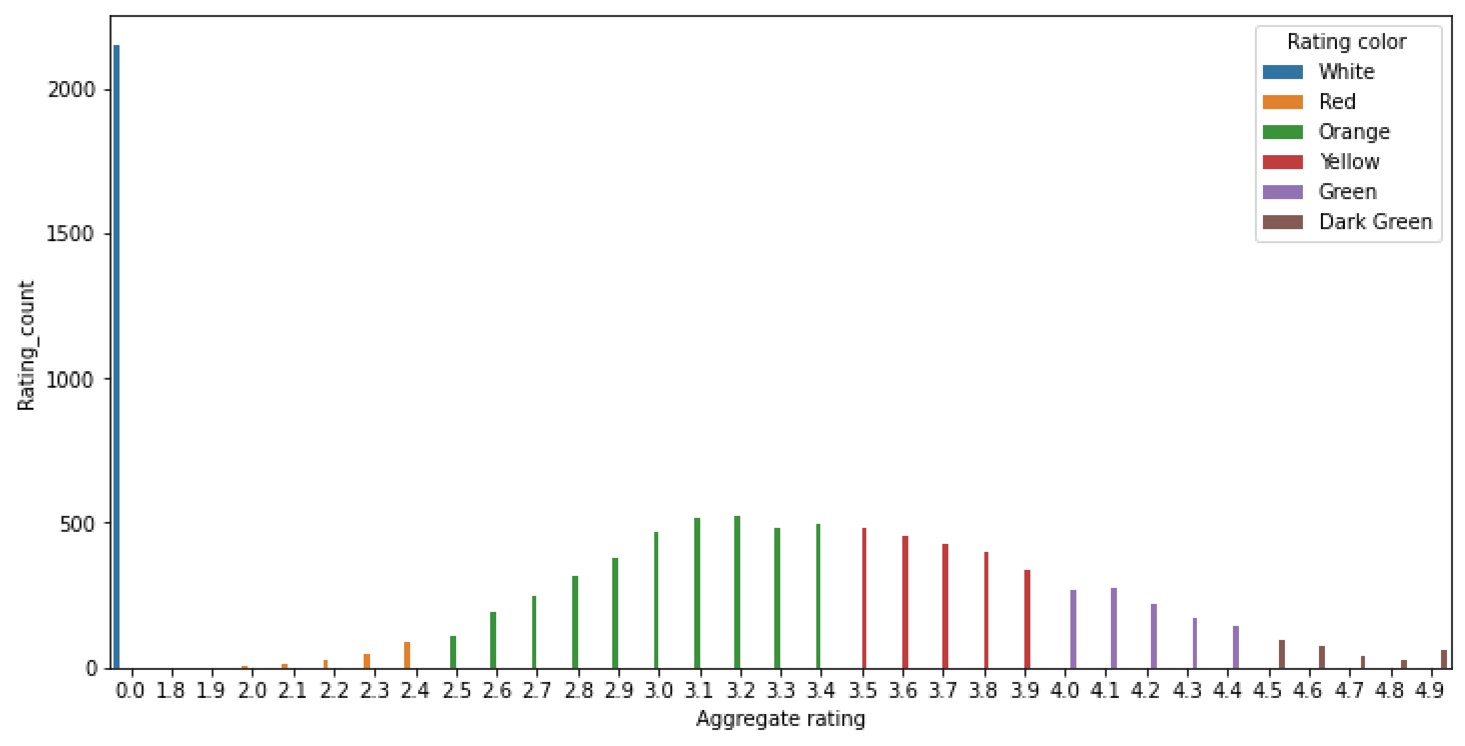
In [27]:
```python
matplotlib.rcParams['figure.figsize']=(12,6)
sns.barplot(x="Aggregate rating",y="Rating_count",data=ratings)
#It looks like gaussian curve.
```

Out[27]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating_count'>
```



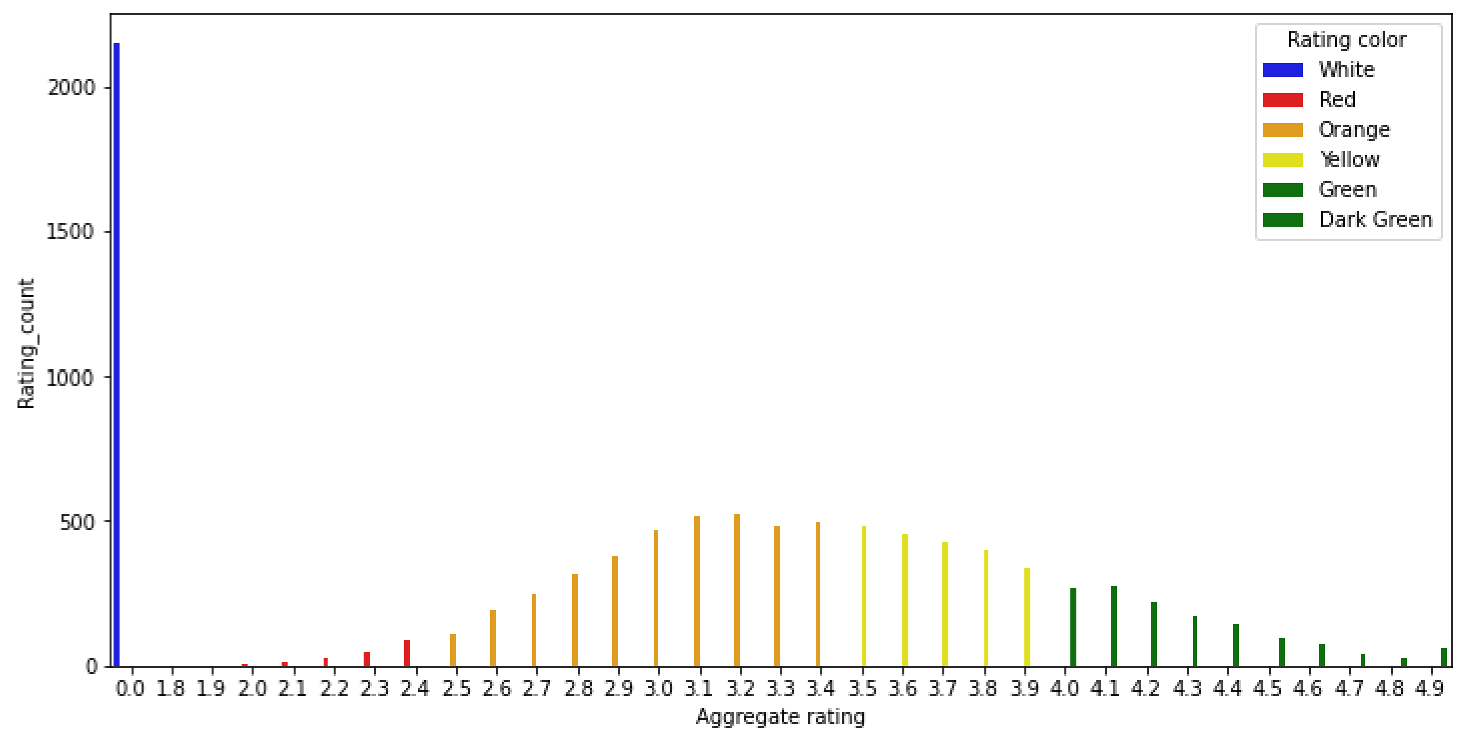
In [28]:
```python
sns.barplot(x="Aggregate rating",y="Rating_count",hue='Rating color',data=ratings)
#HUE ATTRIBUTE DOESN'T GIVE CORRECT COLOR
```

Out[28]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating_count'>
```



In [29]:
```
sns.barplot(x="Aggregate rating",y="Rating_count",hue='Rating color',data=ratings,palette=['blue','red','orange','yellow','green','green'])
```

Out[29]:
```
<AxesSubplot:xlabel='Aggregate rating', ylabel='Rating_count'>
```



Observation:

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

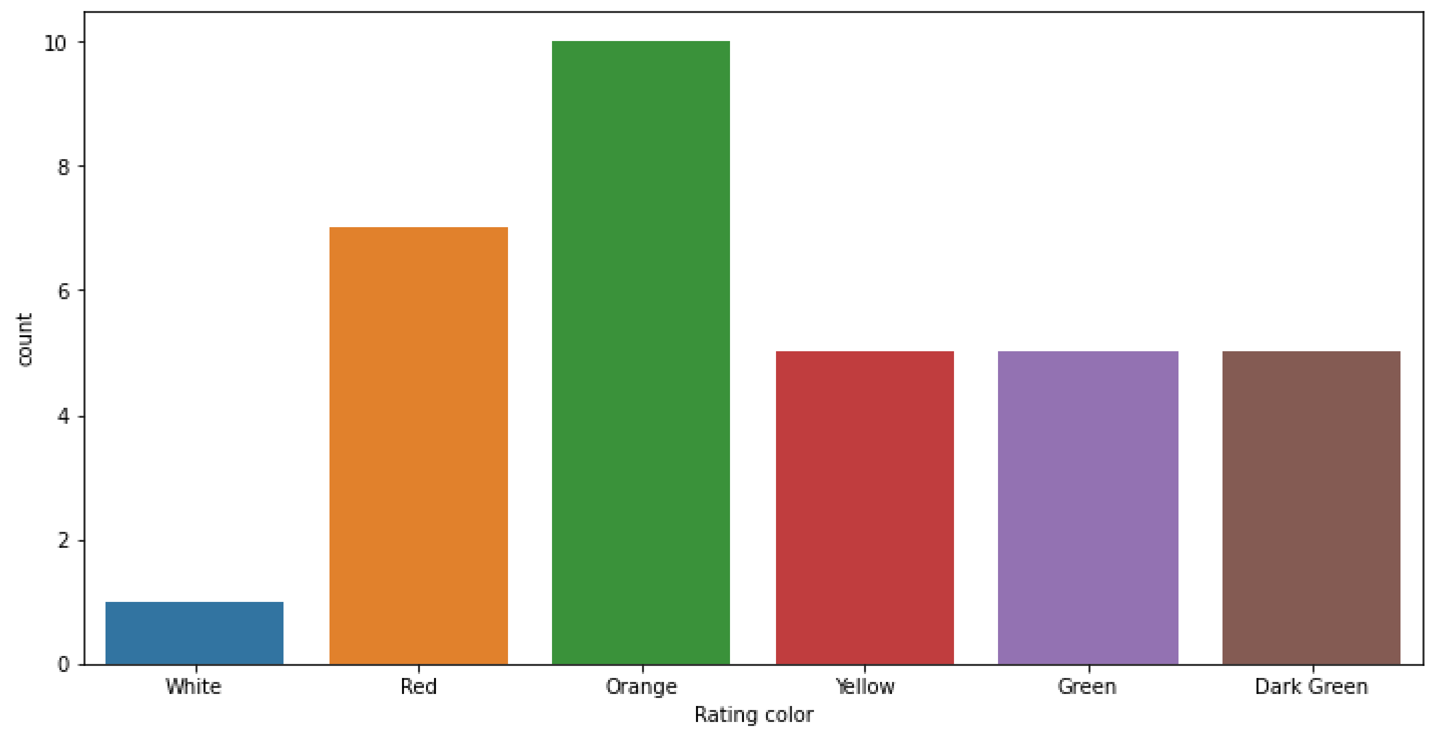
In [30]:
```
ratings
```

Out[30]:

	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 [31]:
```
sns.countplot(x="Rating color",data=ratings)
#Countplot gives the frequency of this rating color
```

Out[31]:
```
<AxesSubplot:xlabel='Rating color', ylabel='count'>
```



In [32]:
```python
### Find the countries name that has given 0 rating
final_df[final_df['Rating color']=='White'].groupby('Country').size().reset_index()
```

Out[32]:

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

Observations Maximum number of 0 ratings are from Indian customers

In [33]:
```python
##find out which currency is used by which country?
final_df.groupby(["Country","Currency"]).size().reset_index()
```

Out[33]:

	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($)	20
9	South Africa	Rand(R)	60
10	Sri Lanka	Sri Lankan Rupee(LKR)	20
11	Turkey	Turkish Lira(TL)	34
12	UAE	Emirati Diram(AED)	60
13	United Kingdom	Pounds(Œ£)	80
14	United States	Dollar($)	434

Which Countries do have online deliveries option

In [34]:
```python
final_df[final_df['Has Online delivery']=='Yes'].groupby(['Has Online delivery','Country']).size().reset_index()
# final_df[final_df['Rating color']=='White'].groupby('Country').size().reset_index()
```

Out[34]:

	Has Online delivery	Country	0
0	Yes	India	2423
1	Yes	UAE	28

Observations:

Online Deliveries are available in India and UAE

In [35]:
```python
## Create a pie chart for top 5 cities distribution
```

In [36]:
```python
final_df.City.value_counts().index
```

Out[36]:
```
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)
```

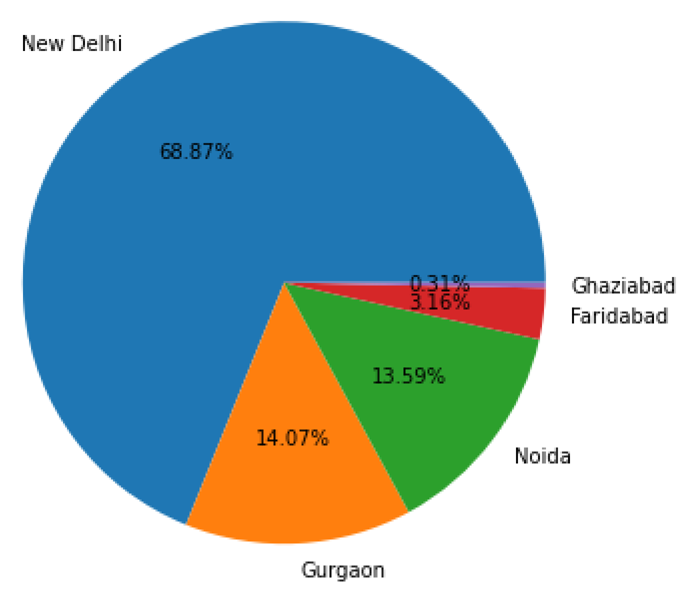
In [37]:
```python
final_df.City.value_counts().values
```

Out[37]:
```
array([5473, 1118, 1080,  251,   25,   21,   21,   21,   21,   21,   20,
         20,   20,   20,   20,   20,   20,   20,   20,   20,   20,   20,
         20,   20,   20,   20,   20,   20,   20,   20,   20,   20,   20,
         20,   20,   20,   20,   20,   20,   20,   20,   20,   20,   20,
         20,   20,   20,   20,   20,   20,   20,   20,   20,   20,   20,
         20,   20,   20,   20,   20,   20,   20,   20,   20,   20,   20,
         20,   20,   20,   20,   20,   20,   20,   20,   20,   20,   20,
         18,   18,   16,   14,   11,    6,    4,    4,    3,    3,    2,
          2,    2,    2,    2,    2,    2,    2,    1,    1,    1,    1,
          1,    1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
          1,    1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
          1,    1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
          1,    1,    1,    1,    1,    1,    1,    1,    1], dtype=int64)
```

In [38]:
```python
city_values=final_df.City.value_counts().values
city_labels=final_df.City.value_counts().index
```

In [39]:
```python
plt.pie(city_values[:5],labels=city_labels[:5],autopct='%1.2f%%')
```

Out[39]:
```
([<matplotlib.patches.Wedge at 0xe29bb6e00>,
  <matplotlib.patches.Wedge at 0xe29bb7550>,
  <matplotlib.patches.Wedge at 0xe29bb7c70>,
  <matplotlib.patches.Wedge at 0xe29bec3d0>,
  <matplotlib.patches.Wedge at 0xe29becaf0>],
 [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%')])
```


New Delhi
Ghaziabad
Faridabad
Noida
Gurgaon