#Zomato sales Analysis

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

df=pd.read_csv('/content/drive/MyDrive/zomato.csv',encoding='latin-1')
df2=df.copy() #making backup file
pd.set_option('display.max_columns', None)
df.tail()

{"type":"dataframe"}

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

df.describe()

{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"Restaurant ID\",\n      \"properties\": {\n
```

```
\"dtype\": \"number\",\n        \"std\": 7645150.642496776,\n
\"min\": 53.0,\n        \"max\": 18500652.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
9051128.349178096,\n          6004089.0,\n          9551.0\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Country Code\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 3362.6853318315943,\n        \"min\":
1.0,\n        \"max\": 9551.0,\n        \"num_unique_values\": 5,\n
\"samples\": [\n          18.365616165846507,\n          216.0,\n
56.750545600949856\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Longitude\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 3360.2106786503405,\n        \"min\": -
157.948486,\n        \"max\": 9551.0,\n        \"num_unique_values\":
8,\n        \"samples\": [\n          64.12657446168706,\n
77.1919642,\n          9551.0\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Latitude\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 3369.972988568076,\n        \"min\": -
41.330428,\n        \"max\": 9551.0,\n        \"num_unique_values\":
8,\n        \"samples\": [\n          25.854380700074756,\n
28.57046888,\n          9551.0\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Average Cost for two\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 281478.0961029089,\n
\"min\": 0.0,\n        \"max\": 800000.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
1199.2107632708617,\n          400.0,\n          9551.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Price range\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
3376.1466071461127,\n        \"min\": 0.9056088473975366,\n
\"max\": 9551.0,\n        \"num_unique_values\": 6,\n
\"samples\": [\n          9551.0,\n          1.804837189823055,\n
4.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Aggregate rating\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 3375.855226922187,\n        \"min\":
0.0,\n        \"max\": 9551.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          2.66637001361114,\n          3.2,\n
9551.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Votes\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 4699.7638410944965,\n        \"min\": 0.0,\n        \"max\":
10934.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n
156.909747670401,\n          31.0,\n          9551.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```

```
df.isna().sum()
```

```
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
```

```
#Checking duplicates
df['Restaurant ID'].duplicated().sum()
```

```
np.int64(0)
```

```
df_country=pd.read_excel('/content/drive/MyDrive/zomato/Country-
Code.xlsx')
df_country
```


```
{"summary":"{\n  \"name\": \"df_country\",\n  \"rows\": 15,\n
\"fields\": [\n    {\n      \"column\": \"Country Code\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
80,\n        \"min\": 1,\n        \"max\": 216,\n
\"num_unique_values\": 15,\n        \"samples\": [\n          189,\n
208,\n          1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Country\",\n      \"properties\": {\n        \"dtype\": \"string\",\
n        \"num_unique_values\": 15,\n        \"samples\": [\n
\"South Africa\",\n          \"Turkey\",\n          \"India\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"df_country"}
```


#Data Prec-Processing & Modeling

```
#Merging two tables
f_df=pd.merge(df,df_country,on='Country Code',how='left')
```

```
f_df.dtypes
```

```
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
```

```
f_df.isnull().sum()
```

```
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
Country                   0
dtype: int64
```

```
#checking duplicates in merged data frame
f_df['Restaurant ID'].duplicated().sum()
```

```
np.int64(0)
```

```
#Converting Currencies of each country into INR
f_df.set_index('Currency')
g=f_df.groupby(['Country','Currency']).size().reset_index().rename(col
umns={0:'No. of Resturaunts'})
g
```


```
{"summary":"{\n  \"name\": \"g\",\n  \"rows\": 15,\n  \"fields\": [\n
{\n      \"column\": \"Country\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 15,\n
\"samples\": [\n          \"South Africa\",\n          \"Turkey\",\n
\"Australia\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Currency\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 12,\n        \"samples\":
[\n          \"Emirati Diram(AED)\",\n          \"Turkish Lira(TL)\",\
n          \"Dollar($)\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"No. of Resturaunts\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 2219,\n        \"min\": 4,\n
\"max\": 8652,\n        \"num_unique_values\": 11,\n
\"samples\": [\n          40,\n          24,\n          80\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"g"}
```


```
#Conveting Botswana Pula(P) to INR
import numpy as np
f_df['Average Cost for two']=np.where(f_df['Currency']=='Botswana
Pula(P)',
                                      f_df['Average Cost for
two']*6.30,
                                      f_df['Average Cost for two']
)
```

```
#Conveting Dollar($) to INR
import numpy as np
f_df['Average Cost for two']=np.where(f_df['Currency']=='Dollar($)',
                                      f_df['Average Cost for
two']*84.97,
                                      f_df['Average Cost for two']
)
```

```
f_df['Currency'].unique()
```

```
array(['Botswana Pula(P)', 'Brazilian Real(R$)', 'Dollar($)',
       'Emirati Diram(AED)', 'Indian Rupees(Rs.)',
       'Indonesian Rupiah(IDR)', 'NewZealand($)', 'Pounds(\x8c£)',
       'Qatari Rial(QR)', 'Rand(R)', 'Sri Lankan Rupee(LKR)',
       'Turkish Lira(TL)'], dtype=object)
```

```
#To automate the task i use for loop with np.where
# Define a dictionary with currency as keys and corrected exchange
rates as values
exchange_rates = {
    'Brazilian Real(R$)': 14.91,
    'Emirati Diram(AED)': 23.12,
    'Indian Rupees(Rs.)': 1,  # No conversion needed for INR
    'Indonesian Rupiah(IDR)': 0.0051,
    'NewZealand($)': 49.67,
    'Pounds(\x8c£)': 111.88,
    'Qatari Rial(QR)': 23.32,
    'Rand(R)': 4.64,
    'Sri Lankan Rupee(LKR)': 0.28,
    'Turkish Lira(TL)': 2.19
}

# Loop through the exchange rates dictionary
for currency, rate in exchange_rates.items():
    # Apply the conversion using np.where
    f_df['Average Cost for two'] = np.where(f_df['Currency'] ==
currency,
                                            f_df['Average Cost for two']
* rate,
                                            f_df['Average Cost for
two'])
```

```
f_df
```

```
{"type":"dataframe","variable_name":"f_df"}
```

```
f_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 22 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   float64
 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
 21  Country               9551 non-null   object
dtypes: float64(4), int64(4), object(14)
memory usage: 1.6+ MB
```

```
# Split the 'Cuisines' column by ', '
f_df['Cuisines'] = f_df['Cuisines'].str.split(', ')
# Explode the 'Cuisines' column to create new rows for each cuisine
f_df = f_df.explode('Cuisines').reset_index(drop=True)

f_df.head()
```


```
{"type":"dataframe","variable_name":"f_df"}
```


#Data Visualisation

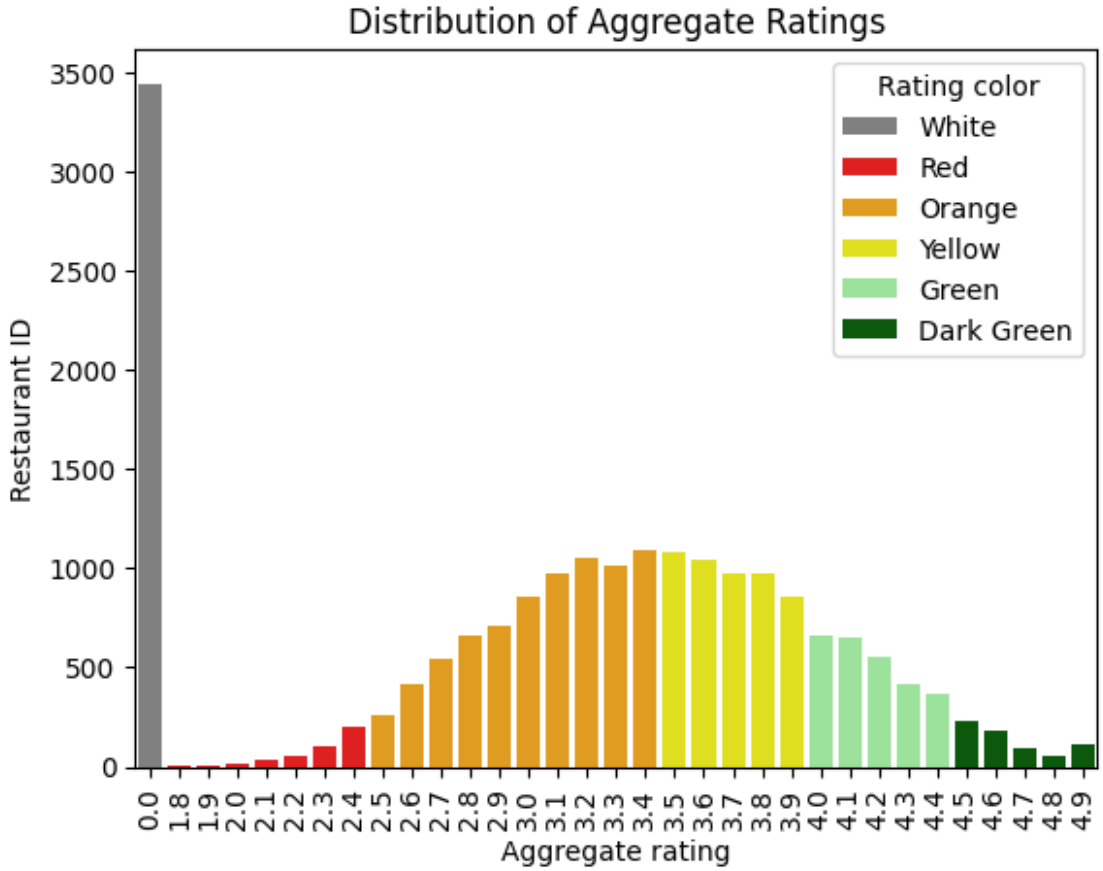
Q1. What is the Distribution of the aggregate rating

```
g=f_df.groupby(['Aggregate rating','Rating color'])['Restaurant
ID'].count().reset_index()
g
```


```
{"summary":"{\n  \"name\": \"g\",\n  \"rows\": 33,\n  \"fields\": [\n
{\n      \"column\": \"Aggregate rating\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1.092051169852291,\n
\"min\": 0.0,\n        \"max\": 4.9,\n        \"num_unique_values\":
33,\n        \"samples\": [\n          4.8,\n          3.2,\n
4.3\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Rating color\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 6,\n        \"samples\":
[\n          \"White\",\n          \"Red\",\n          \"Dark Green\"\
n        ],\n        \"semantic_type\": \"\",\n
```

```
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Restaurant ID\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 639,\n        \"min\": 2,\n
\"max\": 3443,\n        \"num_unique_values\": 33,\n
\"samples\": [\n          52,\n          1053,\n          412\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"g"}

#visualization
import matplotlib.pyplot as plt
import seaborn as sns
sns.barplot(data=g,x='Aggregate rating',y='Restaurant ID',hue='Rating
color',palette=['Gray','red','orange','yellow','lightgreen','darkgreen
'])
plt.xticks(rotation=90)
plt.title('Distribution of Aggregate Ratings')
plt.show()
```



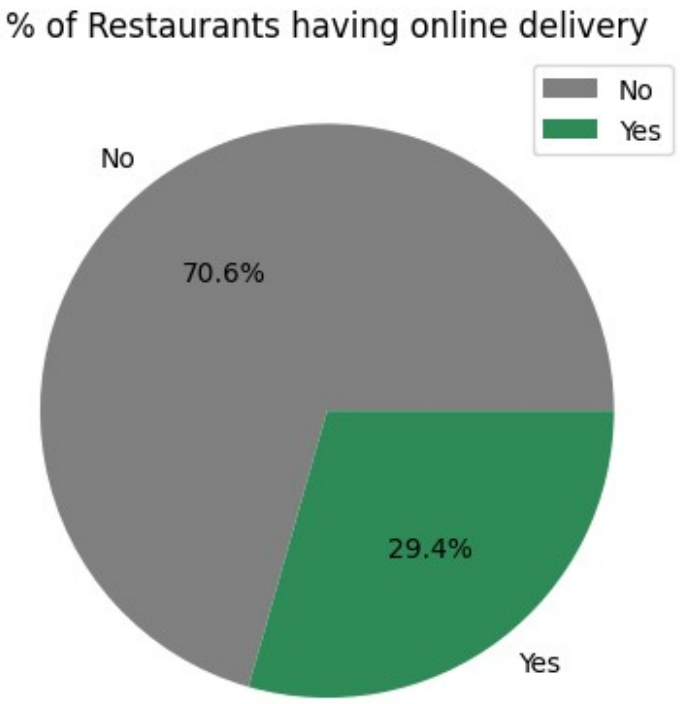
While the majority of restaurants exhibit aggregate ratings between 3.0 and 4.0, it is important to note that a substantial number of entries have no assigned rating, suggesting a significant portion of customer feedback is not captured in the aggregate score.

Q2. how many restaurants offer online delivery

```
restaurants_offer_OD=f_df.groupby('Has Online delivery')['Restaurant
ID'].count().reset_index()
restaurants_offer_OD
```


```
{"summary":"{\n  \"name\": \"restaurants_offer_OD\",\n  \"rows\": 2,\n
\"fields\": [\n    {\n      \"column\": \"Has Online delivery\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          \"Yes\",\
n          \"No\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Restaurant ID\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 5739,\n        \"min\": 5801,\n
\"max\": 13918,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          5801,\n          13918\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\
n}","type":"dataframe","variable_name":"restaurants_offer_OD"}
```


```
#visualization
colors = ['gray', 'seagreen']
plt.pie(restaurants_offer_OD['Restaurant
ID'],labels=restaurants_offer_OD['Has Online delivery'],autopct='%1.1f
%%',colors=colors)
plt.title('% of Restaurants having online delivery')
plt.legend()
plt.show()
```



- *29.4% of analyzed restaurants offer online delivery.*
- *70.6% of analyzed restaurants do not currently offer online delivery.*

Q3. what is the average amount of two accross different cities

```
#calculate Average Cost for two across top 10 cities
av=f_df.groupby('City').agg({'Average Cost for
two':'mean'}).reset_index().head(10).sort_values(by='Average Cost for
two',ascending=False)
print(av.to_string())
```

```
        City  Average Cost for two
0  Abu Dhabi           3920.347826
9   Auckland           3342.081429
3     Albany           2066.315909
7   Armidale           1699.400000
8     Athens           1590.717442
1       Agra            953.061224
2  Ahmedabad            866.447368
5   Amritsar            545.714286
4  Allahabad            512.765957
6     Ankara            179.580000
```

by analyzing the dataset we Concluded that across the top 10 cities, Restaurants of Abu Dhabi has the highest average cost for two.

#Q4. Identify the Top 5 cuisines across different restaurants

```
# identify the Top 5 cuisines across different restaurants
top_cuisines=f_df.groupby('Cuisines').size().reset_index().sort_values
(by=0,ascending=False).rename(columns={0:'frequency'}).head()
top_cuisines
```


```
{"summary":"{\n  \"name\": \"top_cuisines\",\n  \"rows\": 5,\n
\"fields\": [\n    {\n      \"column\": \"Cuisines\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 5,\n        \"samples\": [\n
\"Chinese\",\n          \"Italian\",\n          \"Fast Food\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"frequency\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1311,\n        \"min\": 764,\n        \"max\": 3960,\n
\"num_unique_values\": 5,\n        \"samples\": [\n          2735,\n
764,\n          1986\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"top_cuisines"}
```


Analysis of the dataset reveals that North Indian cuisine is the most prevalent among the top five identified cuisines.

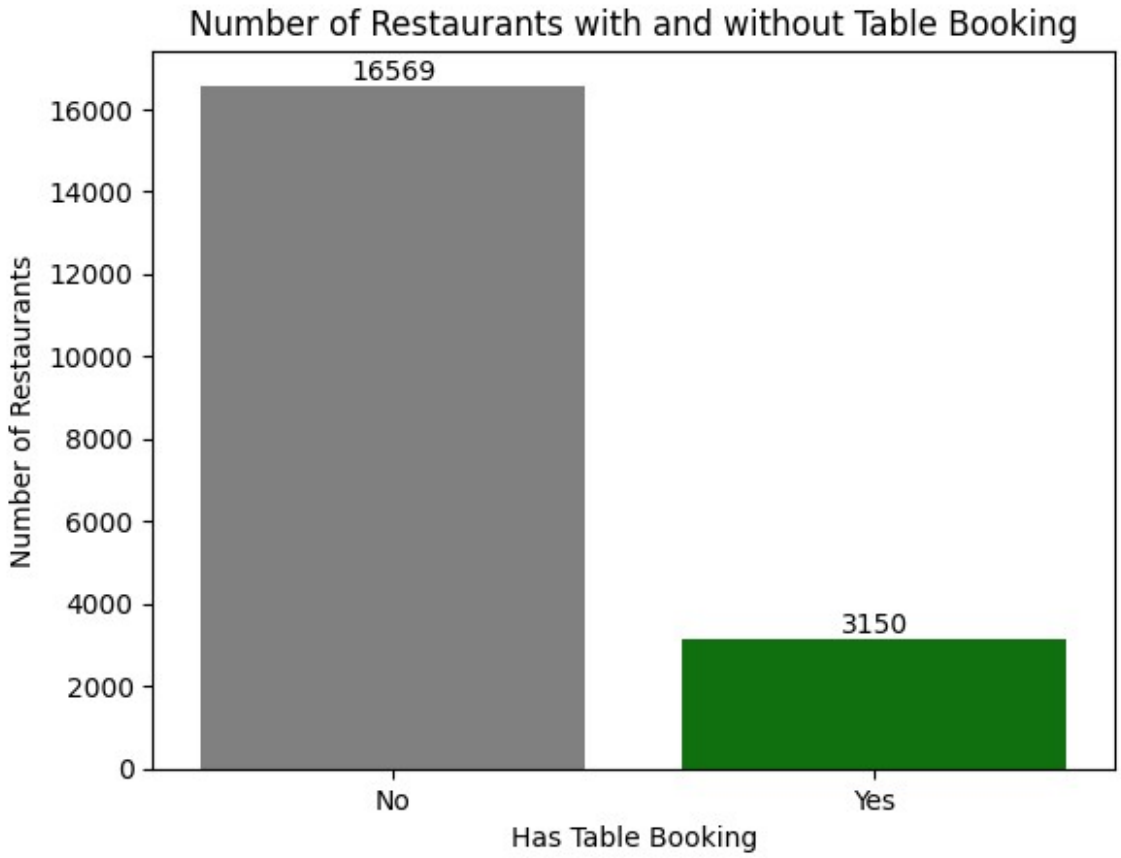
#Q5. Compare the number of restaurant that allows table booking versus those that do not

```
allow_TB=f_df['Has Table
booking'].value_counts().reset_index().rename(columns={'count':'No. of
restaurants'})
allow_TB
```


```
{"summary":"{\n  \"name\": \"allow_TB\",\n  \"rows\": 2,\n
\"fields\": [\n    {\n      \"column\": \"Has Table booking\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          \"Yes\",\
n          \"No\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"No.
of restaurants\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 9488,\n        \"min\": 3150,\n
\"max\": 16569,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          3150,\n          16569\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"allow_TB"}
```


```
#visualization
import matplotlib.pyplot as plt
import seaborn as sns
```

```
ax=sns.barplot(x='Has Table booking', y='No. of restaurants',
data=allow_TB,palette=['gray', 'green'], hue='Has Table booking')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.title('Number of Restaurants with and without Table Booking')
plt.xlabel('Has Table Booking')
plt.ylabel('Number of Restaurants')
plt.show()
```



A significant majority of restaurants in the dataset do not offer table booking services[16596] , in contrast to the smaller proportion that do[3150].

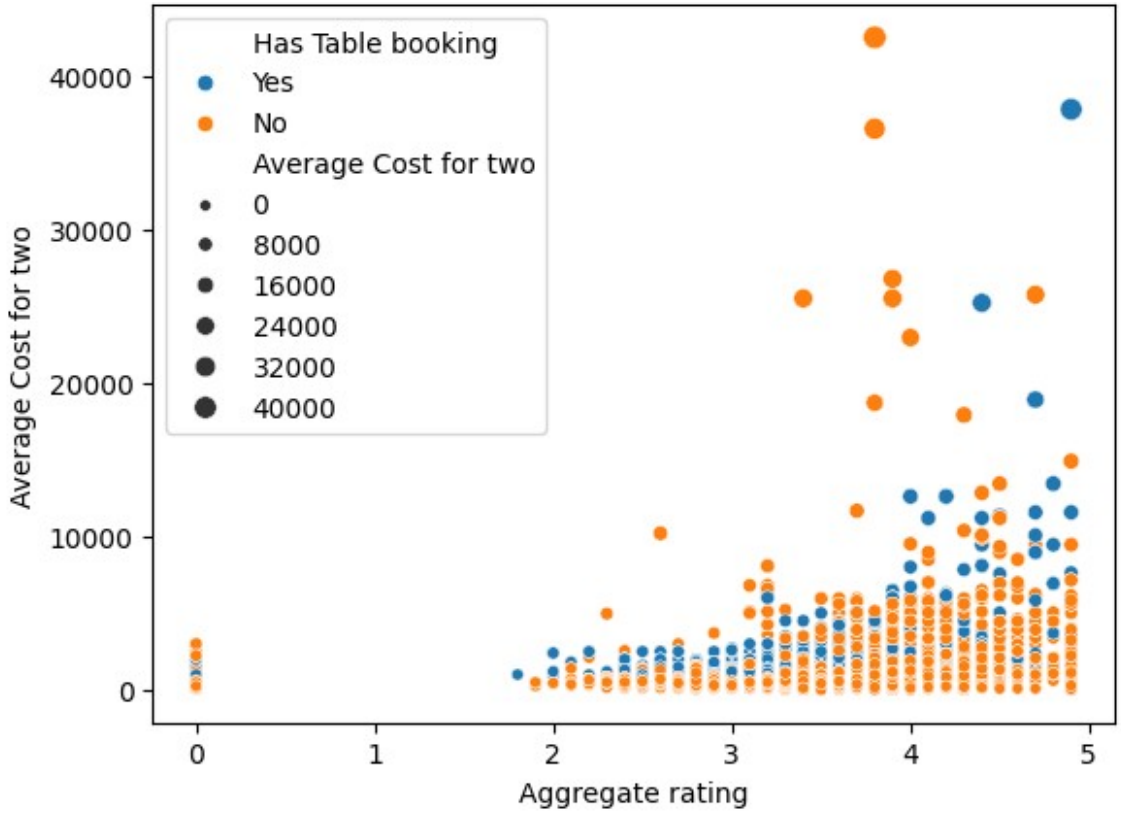
#Q6. Analyze the correlation between average cost for two aggregate rating and how many of them provide table booking.

```
provide_TB=f_df['Has Table
booking'].value_counts().reset_index().rename(columns={'count':'provid
e table booking'})
provide_TB
```


```
{"summary":"{\n  \"name\": \"provide_TB\",\n  \"rows\": 2,\n
\"fields\": [\n    {\n      \"column\": \"Has Table booking\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
```

```
\"num_unique_values\": 2,\n        \"samples\": [\n          \"Yes\",\
n          \"No\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"provide table booking\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 9488,\n        \"min\":
3150,\n        \"max\": 16569,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          3150,\n          16569\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"provide_TB"}
```


```
#visualization
import seaborn as sns
import matplotlib.pyplot as plt
sns.scatterplot(data=f_df,x='Aggregate rating',y='Average Cost for
two',size='Average Cost for two',markers='o',hue='Has Table booking')
plt.show()
```



there's a positive correlation between aggregate rating and average cost for two. restaurant with bettr ratings generally cost more, and those allow table bookings are often higher-rated and costly.

#Q7. identify the top 5 restaurants with highest number of votes

```
g=f_df.groupby('Restaurant
Name').agg({'Votes':'sum'}).reset_index().sort_values(by='Votes',ascen
ding=False).head()
g
```


```
{"summary":"{\n  \"name\": \"g\",\n  \"rows\": 5,\n  \"fields\": [\n
{\n      \"column\": \"Restaurant Name\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 5,\n
\"samples\": [\n          \"Big Chill\",\n          \"Toit\",\n
\"AB's - Absolute Barbecues\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Votes\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 10328,\n        \"min\": 32802,\n
\"max\": 58631,\n        \"num_unique_values\": 5,\n
\"samples\": [\n          43412,\n          32802,\n          40200\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"g"}
```


Top 5 restaurants with highest votes are:-

- "Barbeque Nation" with 58631 votes
- "Big Chill" with 43412 votes
- "AB's - Absolute Barbecues" with 40200 votes
- "Big Brewsky" with 34230 votes
- "Toit" with 32802 votes

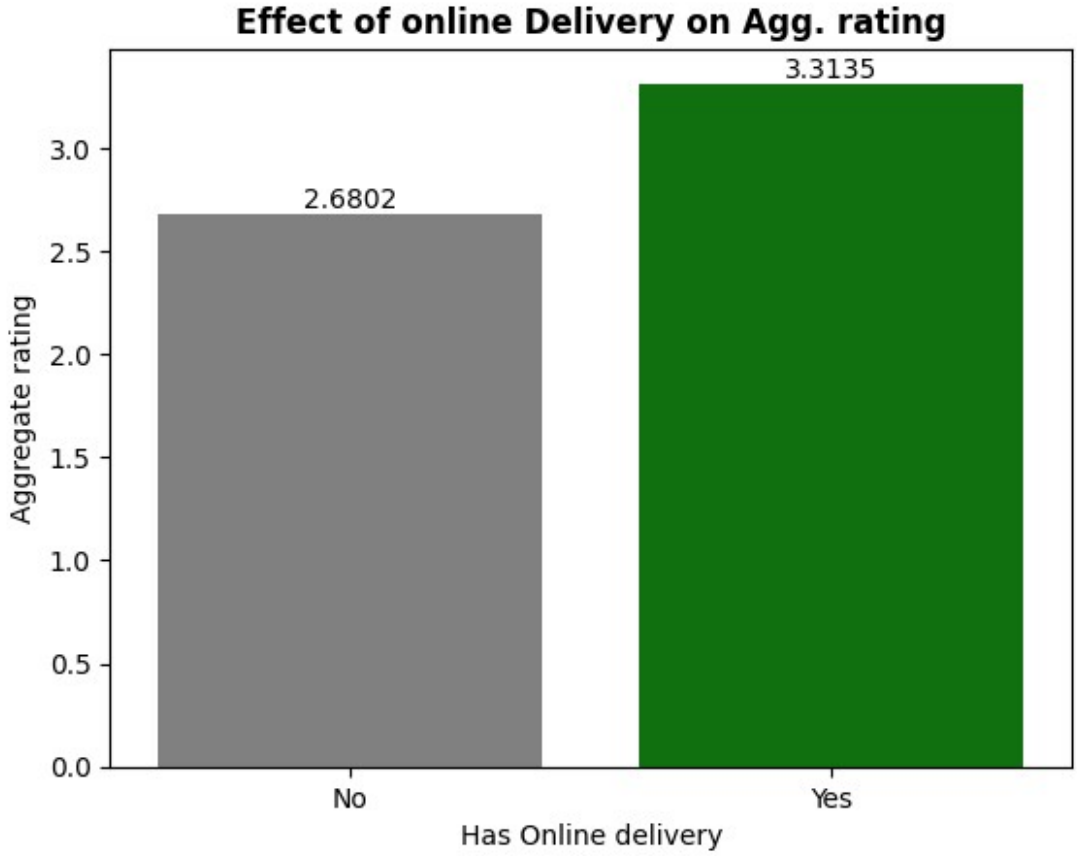
#Q8. how do restaurant rating vary between restaurants that offer online delivery and those that do not.

```
g=f_df.groupby('Has Online delivery')['Aggregate
rating'].mean().reset_index()
g
```


```
{"summary":"{\n  \"name\": \"g\",\n  \"rows\": 2,\n  \"fields\": [\n
{\n      \"column\": \"Has Online delivery\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"Yes\",\n          \"No\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Aggregate rating\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.44781027795990125,\n        \"min\": 2.680198304354074,\n
\"max\": 3.3134976728150316,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          3.3134976728150316,\n
2.680198304354074\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"g"}
```


```
#visualization
ax=sns.barplot(data=g,x='Has Online delivery',y='Aggregate
```

```
rating',palette=['gray', 'green'], hue='Has Online delivery')
ax.bar_label(ax.containers[0])
ax.bar_label(ax.containers[1])
plt.title('Effect of online Delivery on Agg.
rating',fontweight='bold')
plt.show()
```



Analysis reveals that the average aggregate rating of those restaurants who offer's online dilevery service[3.313498] is slightly higher than those that do not[2.6802].

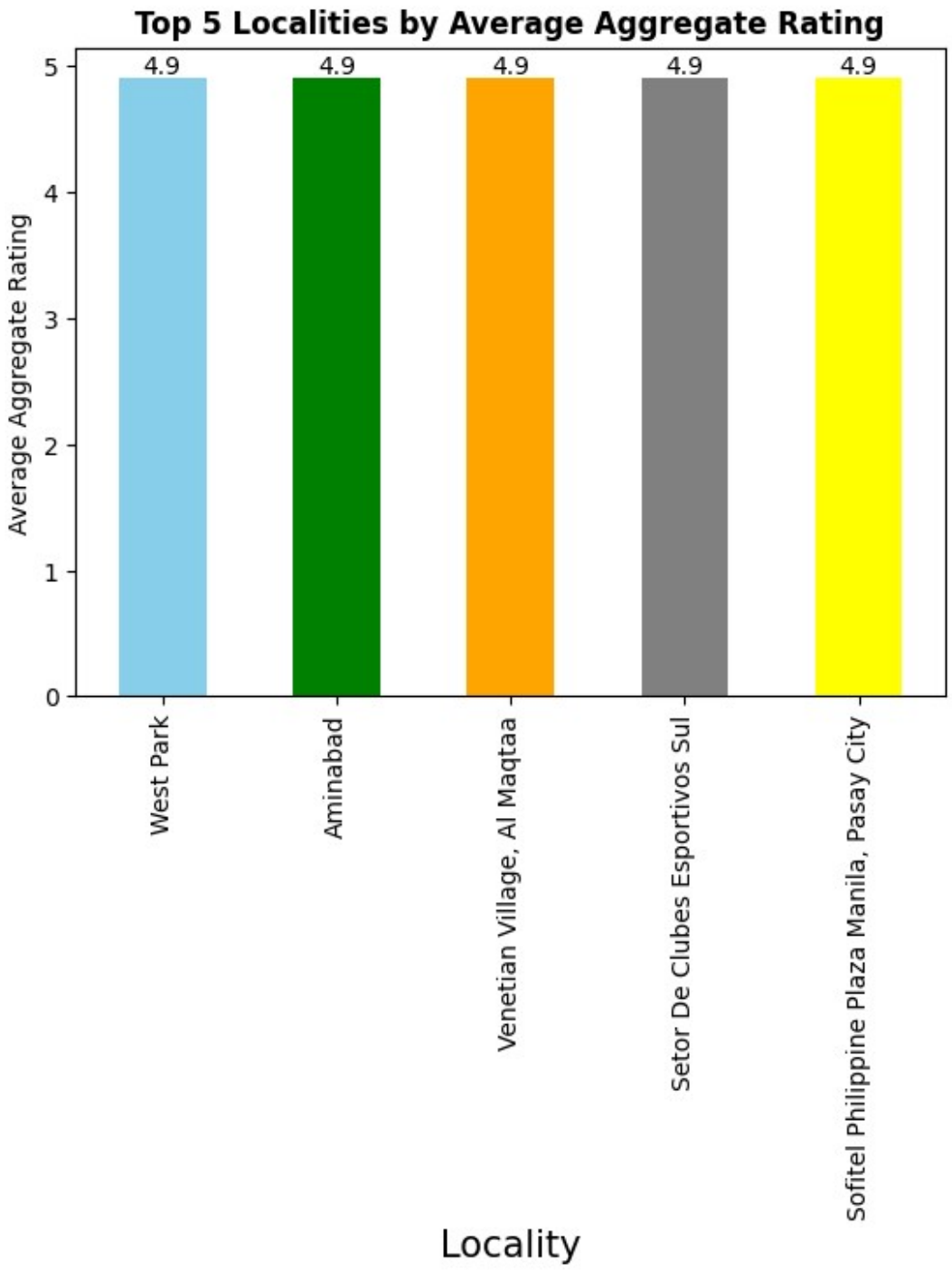
#Q9. Determine the top 5 localities with the highest average restaurant rating.

```
g=f_df.groupby('Locality')['Aggregate
rating'].mean().reset_index().sort_values(by='Aggregate
rating',ascending=False).head(5)
g
```

```
{"summary":"{\n  \"name\": \"g\",\n  \"rows\": 5,\n  \"fields\": [\n
{\n      \"column\": \"Locality\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 5,\n
\"samples\": [\n          \"Aminabad\",\n          \"Sofitel
Philippine Plaza Manila, Pasay City\",\n          \"Venetian Village,
Al Maqtaa\"\n        ],\n        \"semantic_type\": \"\",\n
```

```
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Aggregate rating\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.0,\n        \"min\": 4.9,\n
\"max\": 4.9,\n        \"num_unique_values\": 1,\n        \"samples\":
[\n          4.9\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"g"}

#visualization
colors = ['skyblue', 'green', 'orange', 'gray', 'yellow']
ax=g.plot(kind='bar', x='Locality', y='Aggregate
rating',color=colors,legend=False)
plt.xlabel('Locality',size=15)
plt.ylabel('Average Aggregate Rating')
plt.title('Top 5 Localities by Average Aggregate
Rating',fontweight='bold')
for container in ax.containers:
    ax.bar_label(container, fmt='%.1f')
plt.show()
```



*By analyzing dataset, we have concluded that Top 5 localities with highest restaurant ratings are :-

1. West Park
2. Aminabad
3. Venetian Village, Al Maqtaa
4. Setor De Clubes Esportivos Sul

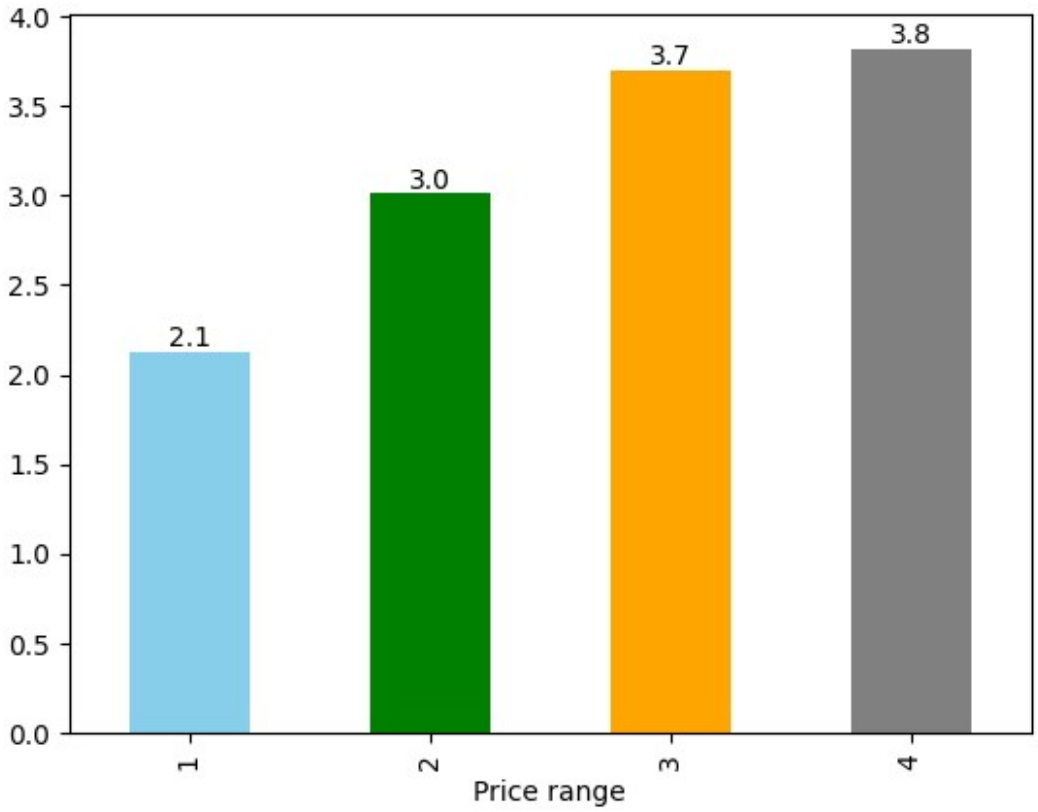
5. Sofitel Philippine Plaza Manila, Pasay City

#Q10. what is the relationship between price range and aggregate rating

```
g=f_df.groupby('Price range')['Aggregate rating'].mean().reset_index()
g
```


```
{"summary":"{\n  \"name\": \"g\",\n  \"rows\": 4,\n  \"fields\": [\n
{\n      \"column\": \"Price range\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1,\n        \"min\": 1,\n
\"max\": 4,\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          2,\n          4,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Aggregate rating\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.7797105807529038,\n        \"min\": 2.1248991121872476,\n
\"max\": 3.8199137311286844,\n        \"num_unique_values\": 4,\n
\"samples\": [\n          3.0122529090144403,\n
3.8199137311286844,\n          2.1248991121872476\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"g"}
```


```
#visualization
colors = ['skyblue', 'green', 'orange', 'gray', 'blue']
ax=g.plot(kind='bar',x='Price range',y='Aggregate
rating',legend=False,color=colors)
for container in ax.containers:
    ax.bar_label(container, fmt='%.1f')
plt.show()
```



Analysis reveals that Restaurants with Price range 4 got highest average rating of 3.4 among those with lower price range.

[Note:- Price range shows the affordability.]