intership-task

November 23, 2024

0.1 Level 1

Task 1 : Data Exploration and Preprocessing

[1]:
```
import pandas as pd
import numpy as np
```

[2]:
```
df=pd.read_csv('Dataset .csv')
```

[3]:
```
df.head(2)
```

[3]:
```
   Restaurant ID   Restaurant Name  Country Code         City  \
0        6317637  Le Petit Souffle           162  Makati City
1        6304287  Izakaya Kikufuji           162  Makati City

                                             Address  \
0  Third Floor, Century City Mall, Kalayaan Avenu…
1  Little Tokyo, 2277 Chino Roces Avenue, Legaspi…

                                    Locality  \
0   Century City Mall, Poblacion, Makati City
1  Little Tokyo, Legaspi Village, Makati City

                                     Locality Verbose   Longitude   Latitude  \
0  Century City Mall, Poblacion, Makati City, Mak…  121.027535  14.565443
1  Little Tokyo, Legaspi Village, Makati City, Ma…  121.014101  14.553708

                     Cuisines  …          Currency Has Table booking  \
0  French, Japanese, Desserts  …  Botswana Pula(P)               Yes
1                    Japanese  …  Botswana Pula(P)               Yes

  Has Online delivery Is delivering now Switch to order menu Price range  \
0                  No                No                   No           3
1                  No                No                   No           3

   Aggregate rating  Rating color Rating text Votes
0               4.8    Dark Green   Excellent   314
1               4.5    Dark Green   Excellent   591
```

```
[2 rows x 21 columns]
```

[4]:
```
df.shape
```

[4]:
```
(9551, 21)
```

[5]:
```
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
```

[6]:
```
print(f'Number of rows:{df.shape[0]}')
print(f'Number of columns:{df.shape[1]}')
```

```
Number of rows:9551
Number of columns:21
```

[7]:
```
miss_value=df.isnull().sum()
print("Missing Values:\n",miss_value)
```

```
Missing Values:
 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
```

[9]:
```
df['Cuisines'].fillna('Unkown',inplace=True)
```

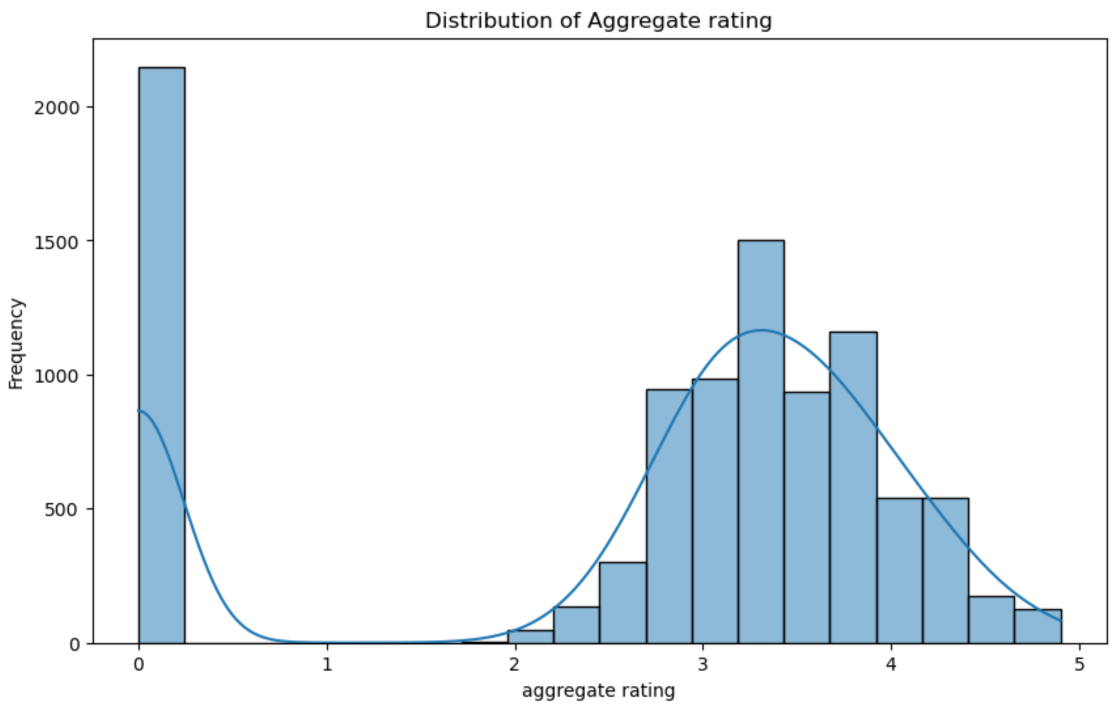
```
C:\Users\jeeva\AppData\Local\Temp\ipykernel_14924\4047093895.py:1:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  df['Cuisines'].fillna('Unkown',inplace=True)
```

[16]:
```
df.isnull().sum()
```

[16]:
```
Restaurant ID            0
Restaurant Name          0
Country Code             0
City                     0
Address                  0
Locality                 0
Locality Verbose        0
Longitude               0
Latitude                0
Cuisines                0
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
```

[18]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

[20]:
```
plt.figure(figsize=(10,6))
sns.histplot(df['Aggregate rating'],bins=20,kde=True)
plt.title('Distribution of Aggregate rating')
plt.xlabel('aggregate rating')
plt.ylabel('Frequency')
plt.show()
```



Task 2 : Descriptive Analysis

[22]:
```
class_count=df['Aggregate rating'].value_counts()
print(class_count)
```

```
Aggregate rating
0.0    2148
3.2     522
3.1     519
3.4     498
3.3     483
3.5     480
3.0     468
3.6     458
3.7     427
3.8     400
2.9     381
3.9     335
2.8     315
4.1     274
4.0     266
2.7     250
4.2     221
2.6     191
4.3     174
4.4     144
2.5     110
4.5      95
2.4      87
4.6      78
4.9      61
2.3      47
4.7      42
2.2      27
4.8      25
2.1      15
2.0       7
1.9       2
1.8       1
Name: count, dtype: int64
```

[23]:
```
df.describe()
```

[23]:
```
       Restaurant ID  Country Code    Longitude     Latitude  \
count   9.551000e+03   9551.000000  9551.000000  9551.000000
```

```
mean    9.051128e+06     18.365616    64.126574    25.854381
std     8.791521e+06     56.750546    41.467058    11.007935
min     5.300000e+01      1.000000  -157.948486   -41.330428
25%     3.019625e+05      1.000000    77.081343    28.478713
50%     6.004089e+06      1.000000    77.191964    28.570469
75%     1.835229e+07      1.000000    77.282006    28.642758
max     1.850065e+07    216.000000   174.832089    55.976980

       Average Cost for two  Price range  Aggregate rating         Votes
count           9551.000000  9551.000000       9551.000000   9551.000000
mean            1199.210763     1.804837          2.666370    156.909748
std            16121.183073     0.905609          1.516378    430.169145
min                0.000000     1.000000          0.000000      0.000000
25%              250.000000     1.000000          2.500000      5.000000
50%              400.000000     2.000000          3.200000     31.000000
75%              700.000000     2.000000          3.700000    131.000000
max           800000.000000     4.000000          4.900000  10934.000000
```

[24]:
```
country_distribution=df['Country Code'].value_counts()
print(country_distribution)
```

```
Country Code
1      8652
216     434
215      80
30       60
214      60
189      60
148      40
208      34
14       24
162      22
94       21
184      20
166      20
191      20
37        4
Name: count, dtype: int64
```

[25]:
```
city_distribution=df['City'].value_counts()
print(city_distribution)
```

```
City
New Delhi           5473
Gurgaon             1118
Noida               1080
Faridabad            251
Ghaziabad             25
```

```
                    …
Panchkula              1
Mc Millan              1
Mayfield               1
Macedon                1
Vineland Station       1
Name: count, Length: 141, dtype: int64
```

[26]:
```
cusine_distribution=df['Cuisines'].value_counts()
print(cusine_distribution)
```

```
Cuisines
North Indian                                               936
North Indian, Chinese                                      511
Chinese                                                    354
Fast Food                                                  354
North Indian, Mughlai                                      334
                                                          …
Bengali, Fast Food                                           1
North Indian, Rajasthani, Asian                              1
Chinese, Thai, Malaysian, Indonesian                         1
Bakery, Desserts, North Indian, Bengali, South Indian        1
Italian, World Cuisine                                       1
Name: count, Length: 1826, dtype: int64
```

[27]:
```
print("Top Cuisines:")
print(cusine_distribution.head(10))
```

```
Top Cuisines:
Cuisines
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: count, dtype: int64
```

[28]:
```
print("Top Citis:")
print(city_distribution.head(10))
```

```
Top Citis:
City
New Delhi       5473
```

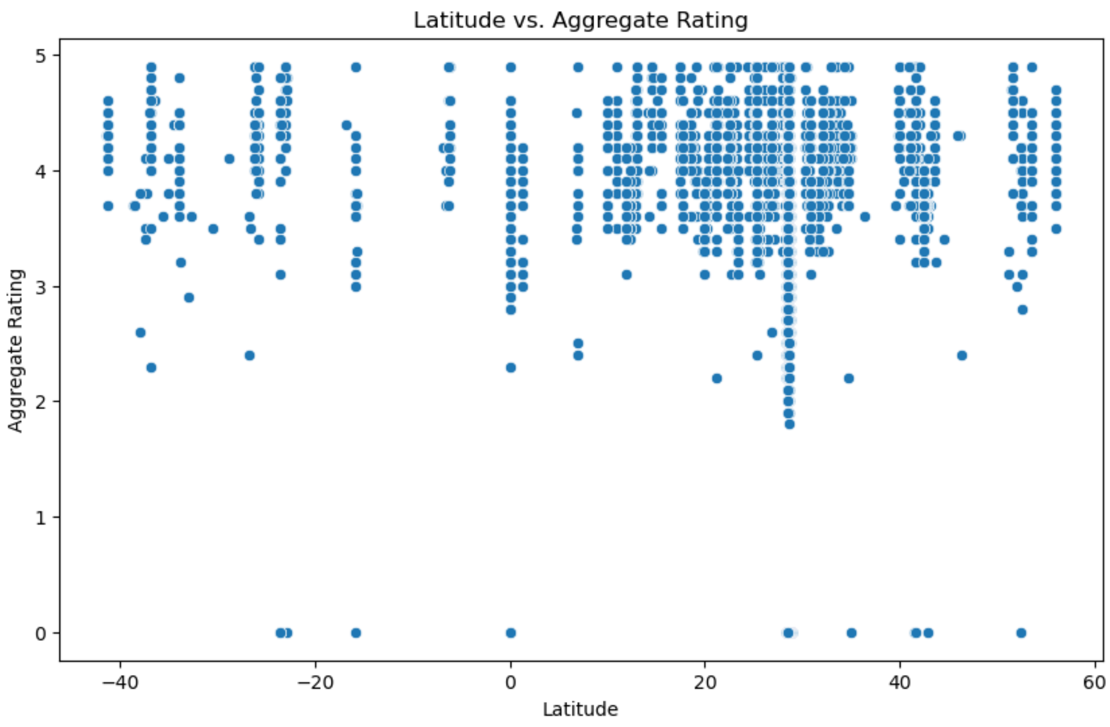
```
Gurgaon          1118
Noida            1080
Faridabad         251
Ghaziabad          25
Bhubaneshwar       21
Amritsar           21
Ahmedabad          21
Lucknow            21
Guwahati           21
Name: count, dtype: int64
```

Task 3 : Geospatial Analysis

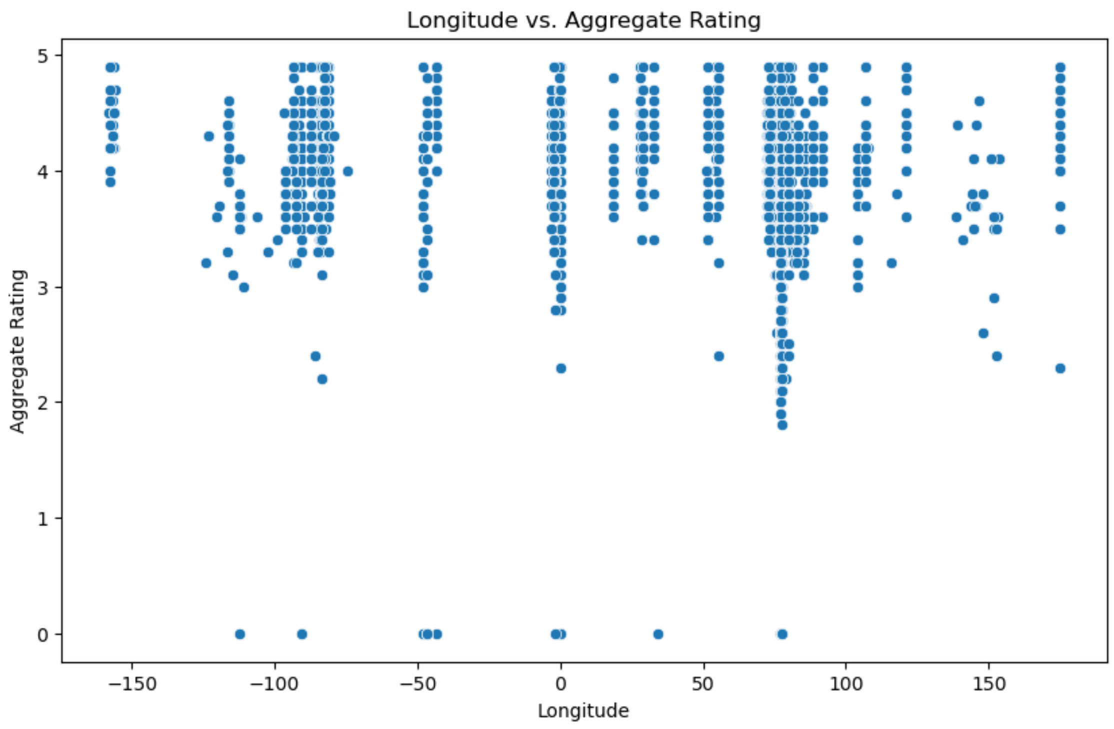
[30]:
```
import folium
map_center=[df['Latitude'].mean(),df['Longitude'].mean()]
restaurant_map=folium.Map(location=map_center,zoom_start=12)

for _,row in df.iterrows():
    folium.
 ↪Marker(location=[row['Latitude'],row['Longitude']],popup=row['Restaurant␣
 ↪Name'],).add_to(restaurant_map)
```

[31]:
```
restaurant_map.save('restaurant_map.html')
```

[32]:
```
plt.figure(figsize=(10,6))
sns.scatterplot(data=df,x='Latitude',y='Aggregate rating')
plt.title('Latitude vs. Aggregate Rating')
plt.xlabel('Latitude')
plt.ylabel('Aggregate Rating')
plt.show()
```



[33]:
```
plt.figure(figsize=(10,6))
sns.scatterplot(data=df,x='Longitude',y='Aggregate rating')
plt.title('Longitude vs. Aggregate Rating')
plt.xlabel('Longitude')
plt.ylabel('Aggregate Rating')
plt.show()
```



```
[34]: corr_matrix=df[['Latitude','Longitude','Aggregate rating']].corr()
      print(corr_matrix)
```

```
                 Latitude  Longitude  Aggregate rating
Latitude         1.000000   0.043207          0.000516
Longitude        0.043207   1.000000         -0.116818
Aggregate rating 0.000516  -0.116818          1.000000
```

0.2 Level 2

Task 1 : Table Booking and Online Delivery

```
[38]: tabel_booking_percentage=(df['Has Table booking'].
        ↪value_counts(normalize=True)*100).get('yes',0)
      print(f'Percentage of restaurant offering booking:{tabel_booking_percentage:.
        ↪2f}%')
```

```
Percentage of restaurant offering booking:0.00%
```

```
[39]: online_percentage=(df['Has Online delivery'].value_counts(normalize=True)*100).
        ↪get('yes',0)
      print(f'Percentage of restaurant offering booking:{online_percentage:.2f}%')
```

```
Percentage of restaurant offering booking:0.00%
```

Task 2 : Price Range Analysis

[49]:
```
ava_rating_with_booking=df[df['Has Table booking']=='Yes']['Aggregate rating'].
 ↪mean()
ava_rating_without_booking=df[df['Has Table booking']=='No']['Aggregate␣
 ↪rating'].mean()
```

[50]:
```
print(f'Avarage rating of with table booking:{ava_rating_with_booking:.2f}')
print(f'Avarage rating of without table booking:{ava_rating_without_booking:.
 ↪2f}')
```

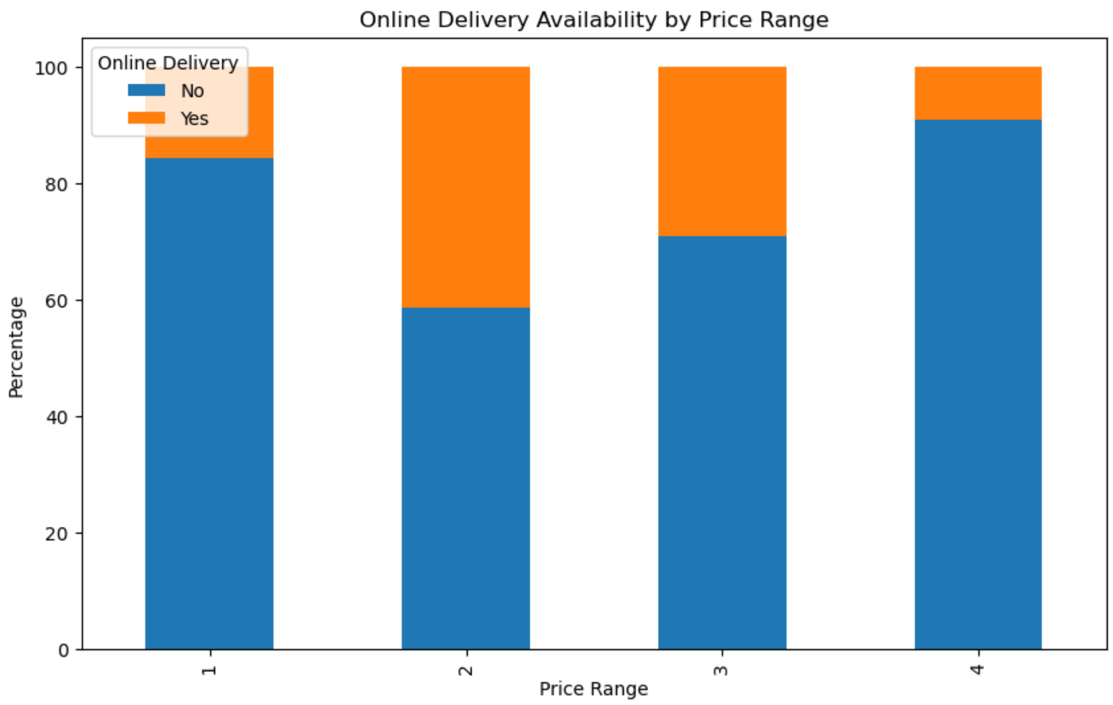
```
Avarage rating of with table booking:3.44
Avarage rating of without table booking:2.56
```

[51]:
```
online_delivery_price=df.groupby('Price range')['Has Online delivery'].
 ↪value_counts(normalize=True).unstack().fillna(0)*100
```

[52]:
```
print(online_delivery_price)
```

```
Has Online delivery         No        Yes
Price range
1                    84.225923  15.774077
2                    58.689367  41.310633
3                    70.809659  29.190341
4                    90.955631   9.044369
```

[53]:
```
online_delivery_price.plot(kind='bar',stacked=True,figsize=(10,6))
plt.title('Online Delivery Availability by Price Range')
plt.xlabel("Price Range")
plt.ylabel('Percentage')
plt.legend(title='Online Delivery',loc='upper left')
plt.show()
```



```
[68]: df.head(2)
```

```
[68]:    Restaurant ID   Restaurant Name  Country Code         City  \
      0        6317637  Le Petit Souffle           162  Makati City
      1        6304287  Izakaya Kikufuji           162  Makati City

                                                  Address  \
      0  Third Floor, Century City Mall, Kalayaan Avenu…
      1  Little Tokyo, 2277 Chino Roces Avenue, Legaspi…

                                        Locality  \
      0   Century City Mall, Poblacion, Makati City
      1  Little Tokyo, Legaspi Village, Makati City

                                       Locality Verbose   Longitude   Latitude  \
      0  Century City Mall, Poblacion, Makati City, Mak…  121.027535  14.565443
      1  Little Tokyo, Legaspi Village, Makati City, Ma…  121.014101  14.553708

                           Cuisines  …         Currency Has Table booking  \
      0  French, Japanese, Desserts  …  Botswana Pula(P)               Yes
      1                    Japanese  …  Botswana Pula(P)               Yes

        Has Online delivery Is delivering now Switch to order menu Price range  \
      0                  No                No                   No           3
```

```
1                 No                No                   No            3

   Aggregate rating  Rating color Rating text Votes
0               4.8    Dark Green   Excellent   314
1               4.5    Dark Green   Excellent   591

[2 rows x 21 columns]
```

Task 3 : Feature Engineering

[70]:
```
df['Restaturant_Name_length']=df['Restaurant Name'].apply(len)
```

[76]:
```
df['Address_length']=df['Address'].apply(len)
```

[84]:
```
df[['Restaurant Name','Restaturant_Name_length','Address','Address_length']].
↪head()
```

[84]:
```
         Restaurant Name  Restaturant_Name_length  \
0        Le Petit Souffle                       16
1        Izakaya Kikufuji                       16
2  Heat - Edsa Shangri-La                       22
3                    Ooma                        4
4             Sambo Kojin                       11

                                           Address  Address_length
0  Third Floor, Century City Mall, Kalayaan Avenu…              71
1  Little Tokyo, 2277 Chino Roces Avenue, Legaspi…              67
2  Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal…              56
3  Third Floor, Mega Fashion Hall, SM Megamall, O…              70
4  Third Floor, Mega Atrium, SM Megamall, Ortigas…              64
```

[90]:
```
df['Has_Table_Booking']=df['Has Table booking'].apply(lambda x: 1 if x == 'Yes'␣
↪else 0)
df['Has_Online_Delivery']=df['Has Online delivery'].apply(lambda x: 1 if x ==␣
↪'Yes' else 0)
```

[92]:
```
df[['Has Table booking','Has_Table_Booking','Has Online␣
↪delivery','Has_Online_Delivery']].head()
```

[92]:
```
  Has Table booking  Has_Table_Booking Has Online delivery  \
0               Yes                  1                  No
1               Yes                  1                  No
2               Yes                  1                  No
3                No                  0                  No
4               Yes                  1                  No

   Has_Online_Delivery
```

```
0                    0
1                    0
2                    0
3                    0
4                    0
```

0.3 Level 3

Task 1: Predictive Modeling

[102]:
```
feature=['Restaturant_Name_length','Address_length','Has_Table_Booking','Has_Online_Delivery']
target = ['Aggregate rating']
```

[104]:
```
X=df[feature]
y=df[target]
```

[106]:
```
from sklearn.model_selection import train_test_split
```

[108]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.
↪2,random_state=42)
```

Linear Regression

[111]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error,r2_score
```

[113]:
```
lr=LinearRegression()
lr.fit(X_train,y_train)
```

[113]:
```
LinearRegression()
```

[115]:
```
y_pred_lr=lr.predict(X_test)
```

[117]:
```
mae_lr=mean_absolute_error(y_test,y_pred_lr)
msse_lr=mean_squared_error(y_test,y_pred_lr)
r2_lr=r2_score(y_test,y_pred_lr)
```

[119]:
```
print(f"Linear Regression - MAE:{mae_lr}, MSE:{msse_lr},R2:{r2_lr}")
```

```
Linear Regression - MAE:1.193199733256996,
MSE:2.1032980983357454,R2:0.07592382118320307
```

Decision Tree

[125]:
```
from sklearn.tree import DecisionTreeRegressor
```

[127]:
```
dt=DecisionTreeRegressor()
dt.fit(X_train,y_train)
```

```
[127]: DecisionTreeRegressor()
```

```
[129]: y_pred_dt=dt.predict(X_test)
```

```
[131]: mae_dt=mean_absolute_error(y_test,y_pred_dt)
       msse_dt=mean_squared_error(y_test,y_pred_dt)
       r2_dt=r2_score(y_test,y_pred_dt)
```

```
[133]: print(f"DecisionTreeRegressor - MAE:{mae_dt}, MSE:{msse_dt},R2:{r2_dt}")
```

```
DecisionTreeRegressor - MAE:1.3223126575185862,
MSE:3.0066380393776058,R2:-0.32095521443759667
```

Random Forest

```
[136]: from sklearn.ensemble import RandomForestRegressor
```

```
[138]: rf=RandomForestRegressor(random_state=42)
       rf.fit(X_train,y_train)
```

```
C:\Users\jeeva\anaconda3\Lib\site-packages\sklearn\base.py:1474:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples,), for example using
ravel().
  return fit_method(estimator, *args, **kwargs)
```

```
[138]: RandomForestRegressor(random_state=42)
```

```
[140]: y_pred_rf=rf.predict(X_test)
```

```
[142]: mae_rf=mean_absolute_error(y_test,y_pred_rf)
       msse_rf=mean_squared_error(y_test,y_pred_rf)
       r2_rf=r2_score(y_test,y_pred_rf)
```

```
[144]: print(f"Random Forest - MAE:{mae_rf}, MSE:{msse_rf},R2:{r2_rf}")
```

```
Random Forest - MAE:1.2526774928676685,
MSE:2.5957098373762104,R2:-0.14041544074885026
```

Task 2: Customer Preference Analysis

```
[149]: cusine_rating=df.groupby('Cuisines')['Aggregate rating'].mean().
         ↪sort_values(ascending=False)
```

```
[155]: cusine_rating.head(10)
```

```
[155]: Cuisines
       Continental, Indian             4.9
       BBQ, Breakfast, Southern        4.9
```

```
Italian, Deli                   4.9
American, Caribbean, Seafood    4.9
Burger, Bar Food, Steak         4.9
American, Burger, Grill         4.9
Italian, Bakery, Continental    4.9
European, Asian, Indian         4.9
European, Contemporary          4.9
American, Coffee and Tea        4.9
Name: Aggregate rating, dtype: float64
```

[157]:
```
cusine_votes = df.groupby('Cuisines')['Votes'].sum().
 ↪sort_values(ascending=False)
```

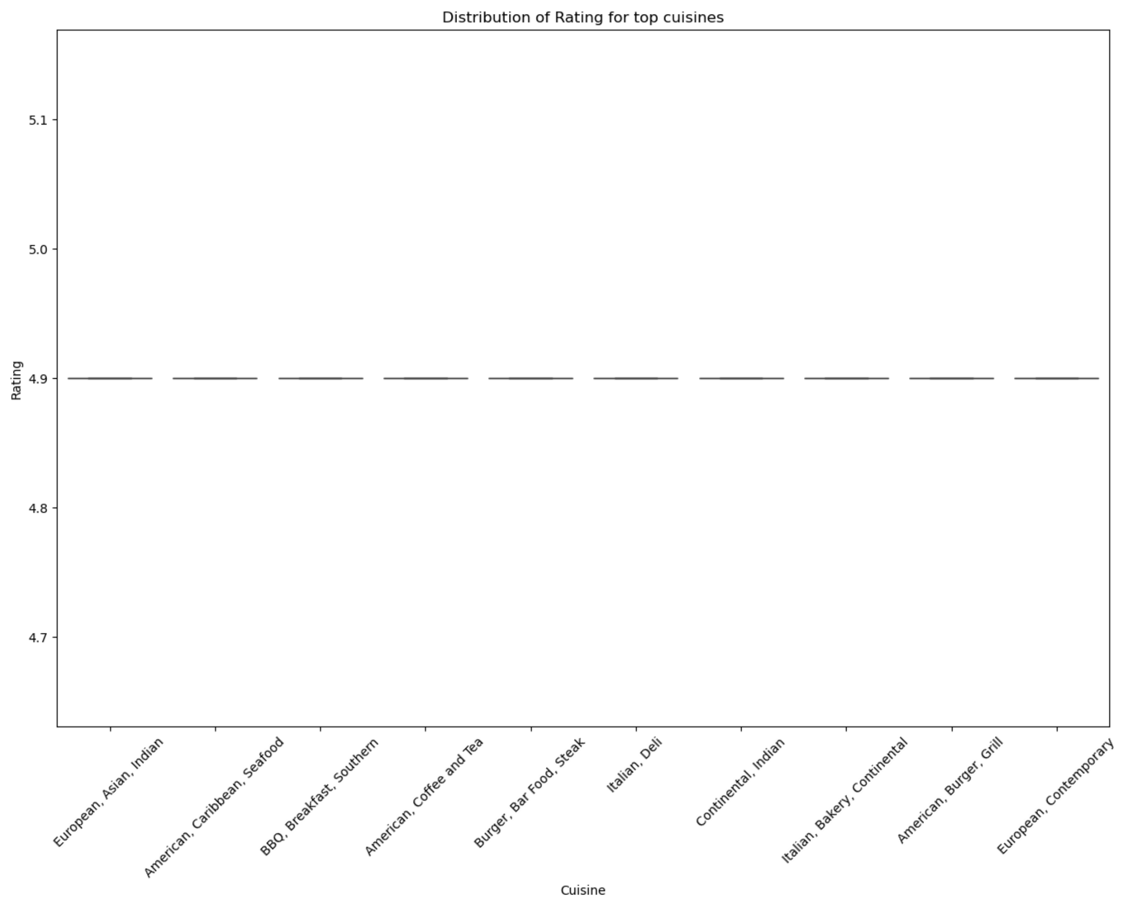
[161]:
```
cusine_votes.head(10)
```

[161]:
```
Cuisines
North Indian, Mughlai               53747
North Indian                        46241
North Indian, Chinese               42012
Cafe                                30657
Chinese                             21925
North Indian, Mughlai, Chinese      20115
Fast Food                           17852
South Indian                        16433
Mughlai, North Indian               15275
Italian                             14799
Name: Votes, dtype: int64
```

[163]:
```
top_cusine=cusine_rating.head(10).index
```

[165]:
```
df_top_cusines=df[df['Cuisines'].isin(top_cusine)]
```

[169]:
```
plt.figure(figsize=(15,10))
sns.boxplot(x='Cuisines',y='Aggregate rating', data=df_top_cusines)
plt.xticks(rotation=45)
plt.title('Distribution of Rating for top cuisines')
plt.xlabel('Cuisine')
plt.ylabel('Rating')
plt.show()
```

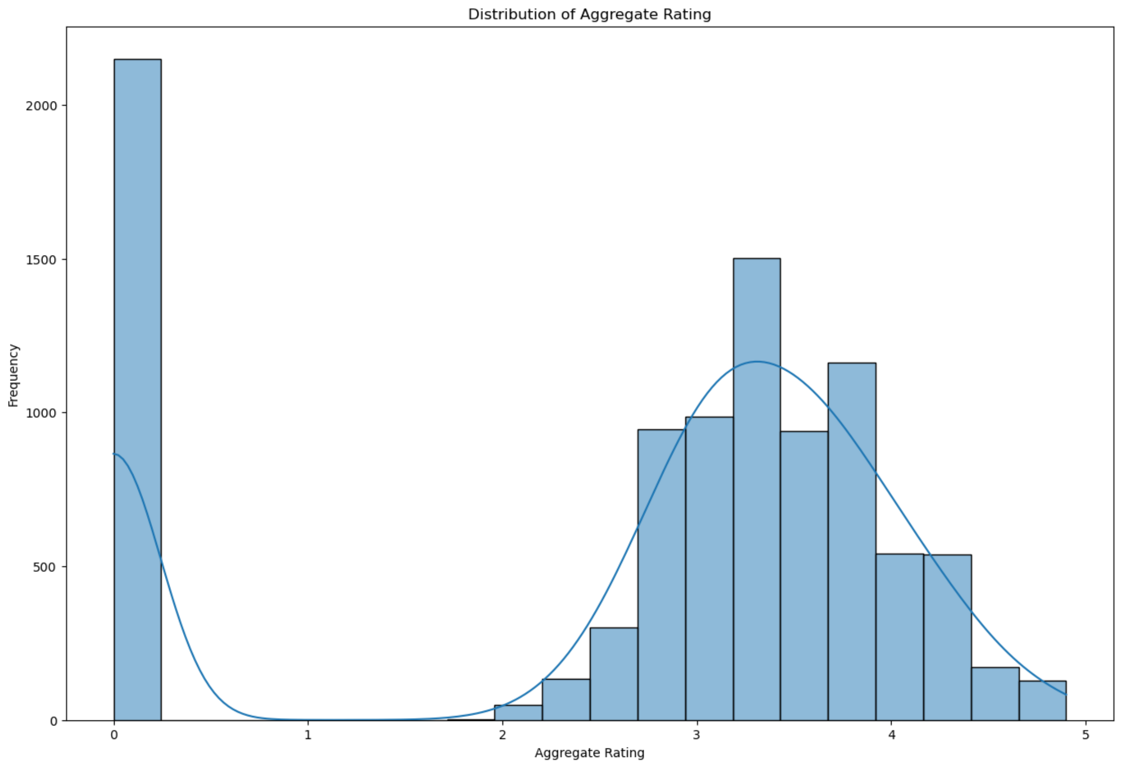


Task 3: Data Visualization

1.Distribution of Rating

Histogram of Rating

[174]:
```
plt.figure(figsize=(15,10))
sns.histplot(df['Aggregate rating'],bins=20,kde=True)
plt.title('Distribution of Aggregate Rating')
plt.xlabel('Aggregate Rating')
plt.ylabel('Frequency')
plt.show()
```

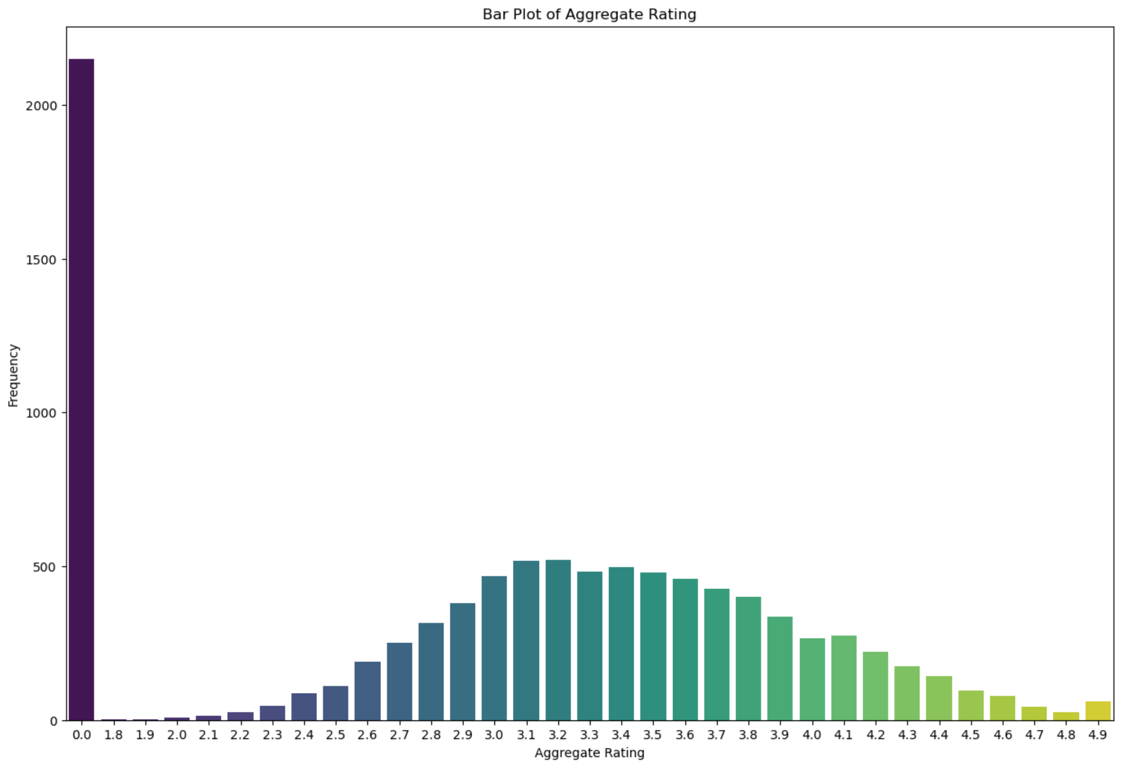


Bar Plot of Rating

```
[179]: rating_counts=df['Aggregate rating'].value_counts().sort_index()
```

```
[181]: plt.figure(figsize=(15,10))
       sns.barplot(x=rating_counts.index,y=rating_counts.values,palette='viridis')
       plt.title('Bar Plot of Aggregate Rating')
       plt.xlabel('Aggregate Rating')
       plt.ylabel('Frequency')
       plt.show()
```

```
C:\Users\jeeva\AppData\Local\Temp\ipykernel_14924\3444833412.py:2:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=rating_counts.index,y=rating_counts.values,palette='viridis')
```

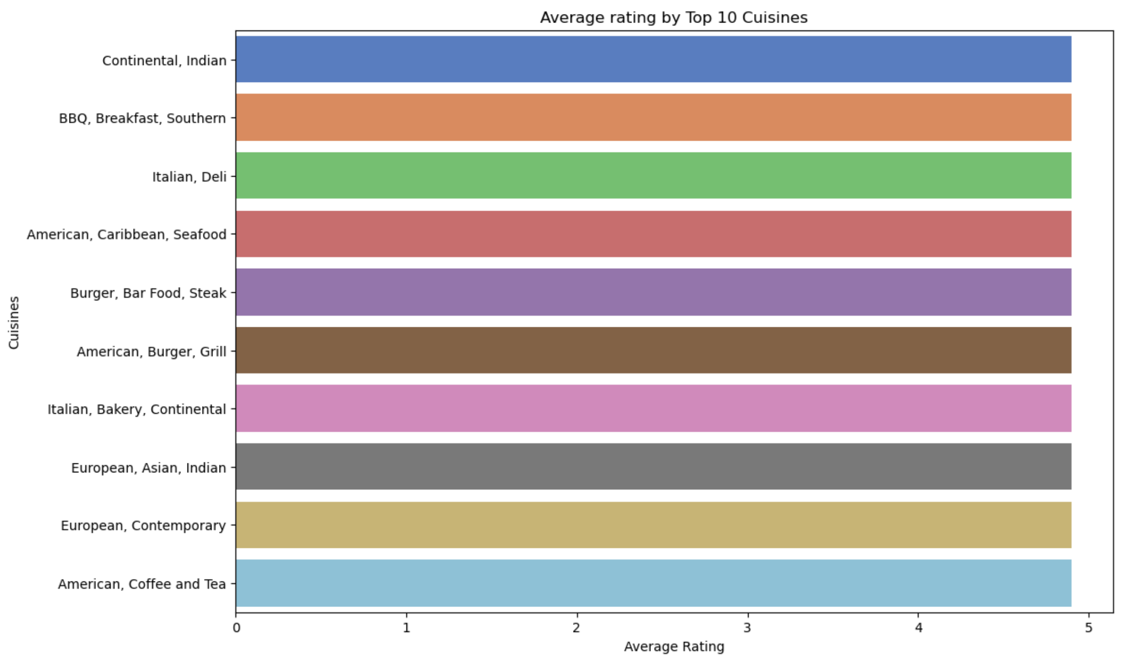


Average Rating by Cuisine

[184]:
```
ava_rating_cuisine=df.groupby('Cuisines')['Aggregate rating'].mean().
  ↪sort_values(ascending=False).head(10)
```

[188]:
```
plt.figure(figsize=(12,8))
sns.barplot(y=ava_rating_cuisine.index,x=ava_rating_cuisine.
  ↪values,palette='muted')
plt.title('Average rating by Top 10 Cuisines')
plt.xlabel('Average Rating')
plt.ylabel('Cuisines')
plt.show()
```

```
C:\Users\jeeva\AppData\Local\Temp\ipykernel_14924\106019245.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(y=ava_rating_cuisine.index,x=ava_rating_cuisine.values,palette='mu
ted')
```

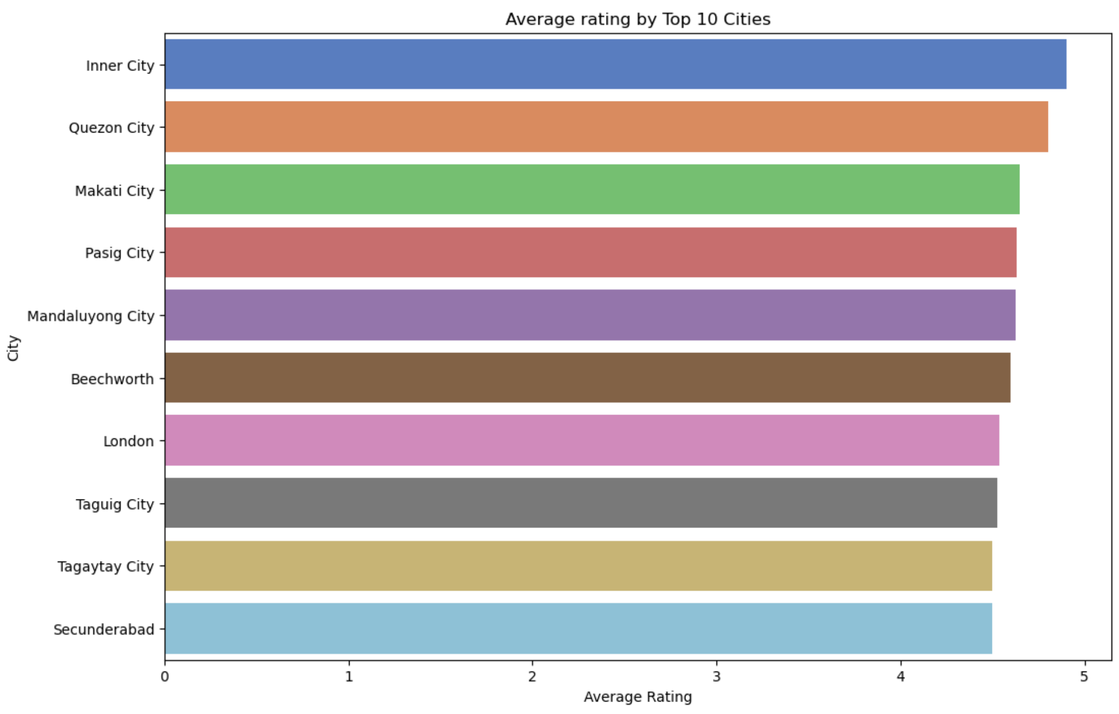


Average Rating by city

[191]:
```
ava_rating_city =df.groupby('City')['Aggregate rating'].mean().
 ↪sort_values(ascending=False).head(10)
```

[193]:
```
plt.figure(figsize=(12,8))
sns.barplot(y=ava_rating_city.index,x=ava_rating_city.values,palette='muted')
plt.title('Average rating by Top 10 Cities')
plt.xlabel('Average Rating')
plt.ylabel('City')
plt.show()
```

```
C:\Users\jeeva\AppData\Local\Temp\ipykernel_14924\2114028040.py:2:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(y=ava_rating_city.index,x=ava_rating_city.values,palette='muted')
```

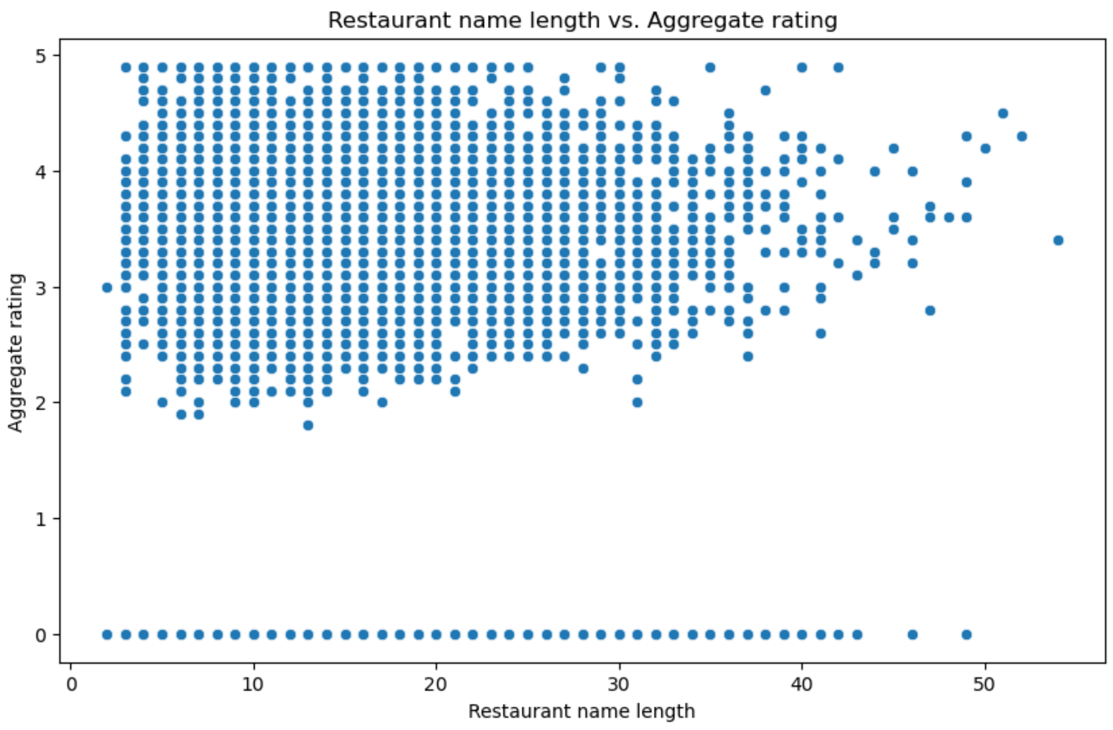


Scatter plot: Restaurant Name Length vs. Aggregate Rating

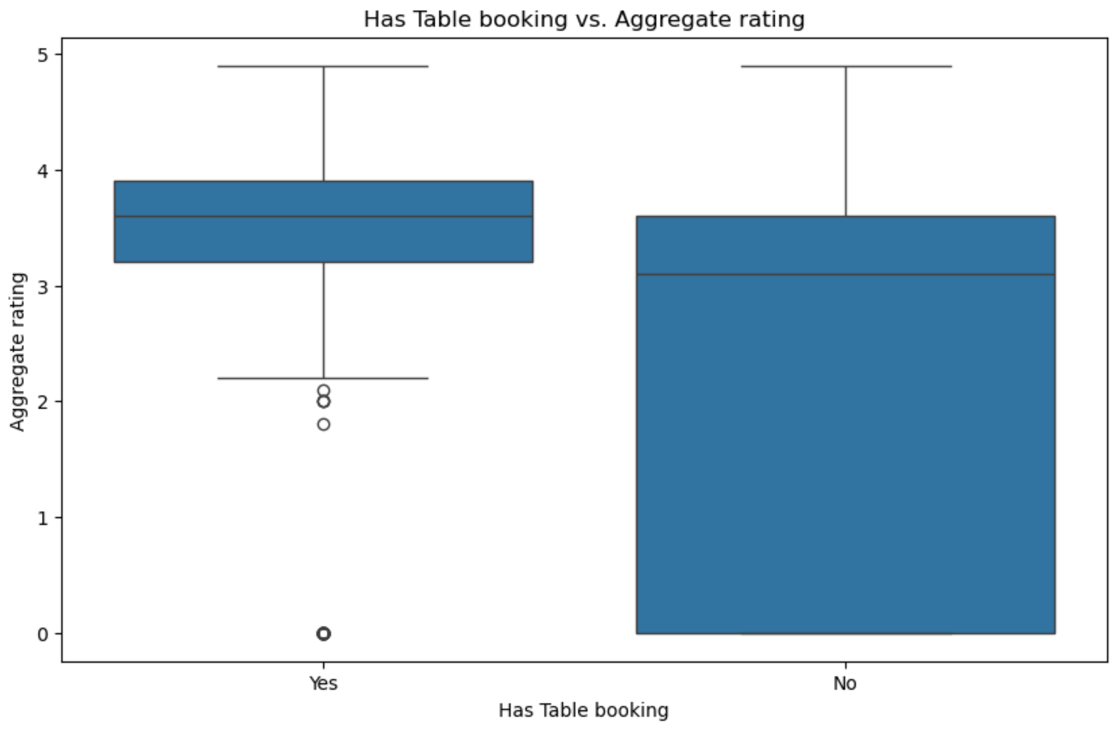
[200]:
```
df.head(3)
```

[200]:
```
   Restaurant ID         Restaurant Name  Country Code              City  \
0        6317637        Le Petit Souffle           162       Makati City
1        6304287        Izakaya Kikufuji           162       Makati City
2        6300002  Heat - Edsa Shangri-La           162  Mandaluyong City

                                             Address  \
0  Third Floor, Century City Mall, Kalayaan Avenu…
1  Little Tokyo, 2277 Chino Roces Avenue, Legaspi…
2  Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal…

                                    Locality  \
0   Century City Mall, Poblacion, Makati City
1  Little Tokyo, Legaspi Village, Makati City
2  Edsa Shangri-La, Ortigas, Mandaluyong City

                                    Locality Verbose   Longitude   Latitude  \
0  Century City Mall, Poblacion, Makati City, Mak…  121.027535  14.565443
1  Little Tokyo, Legaspi Village, Makati City, Ma…  121.014101  14.553708
2  Edsa Shangri-La, Ortigas, Mandaluyong City, Ma…  121.056831  14.581404

                          Cuisines  … Switch to order menu Price range  \
0        French, Japanese, Desserts  …                        No          3
1                          Japanese  …                        No          3
2  Seafood, Asian, Filipino, Indian  …                        No          4

  Aggregate rating Rating color Rating text Votes  Restaurant_Name_length  \
0              4.8   Dark Green   Excellent   314                      16
1              4.5   Dark Green   Excellent   591                      16
2              4.4        Green   Very Good   270                      22

   Address_length Has_Table_Booking Has_Online_Delivery
0              71                 1                   0
1              67                 1                   0
2              56                 1                   0

[3 rows x 25 columns]
```

[202]:
```
plt.figure(figsize=(10,6))
sns.scatterplot(data=df,x='Restaturant_Name_length',y='Aggregate rating')
plt.title('Restaurant name length vs. Aggregate rating')
plt.xlabel('Restaurant name length')
plt.ylabel('Aggregate rating')
plt.show()
```



[206]:
```python
plt.figure(figsize=(10,6))
sns.boxplot(data=df,x='Has Table booking',y='Aggregate rating')
plt.title('Has Table booking vs. Aggregate rating')
plt.xlabel('Has Table booking')
plt.ylabel('Aggregate rating')
plt.show()
```



[ ]:
```
```