ml-project1

August 21, 2024

importing the dependencies

[ ]:
```
!pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages
(3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages
(from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages
(from nltk) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in
/usr/local/lib/python3.10/dist-packages (from nltk) (2024.5.15)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages
(from nltk) (4.66.4)
```

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import accuracy_score
```

#Data Preprocessing and Exploration

[ ]:
```
#importing the dataset
df=pd.read_csv('/content/drive/MyDrive/Cognifyz/Dataset 2.csv')
df.head()
```

[ ]:
```
   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
3        6318506                    Ooma           162  Mandaluyong City
4        6314302             Sambo Kojin           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…
3  Third Floor, Mega Fashion Hall, SM Megamall, O…
4  Third Floor, Mega Atrium, SM Megamall, Ortigas…

                                    Locality  \
0   Century City Mall, Poblacion, Makati City
1  Little Tokyo, Legaspi Village, Makati City
2  Edsa Shangri-La, Ortigas, Mandaluyong City
3      SM Megamall, Ortigas, Mandaluyong City
4      SM Megamall, 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
3  SM Megamall, Ortigas, Mandaluyong City, Mandal…  121.056475  14.585318
4  SM Megamall, Ortigas, Mandaluyong City, Mandal…  121.057508  14.584450

                          Cuisines  …          Currency Has Table booking  \
0       French, Japanese, Desserts  …  Botswana Pula(P)               Yes
1                         Japanese  …  Botswana Pula(P)               Yes
2  Seafood, Asian, Filipino, Indian  …  Botswana Pula(P)               Yes
3                  Japanese, Sushi  …  Botswana Pula(P)                No
4                 Japanese, Korean  …  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
2                  No                No                   No           4
3                  No                No                   No           4
4                  No                No                   No           4

   Aggregate rating  Rating color Rating text Votes
0               4.8    Dark Green   Excellent   314
1               4.5    Dark Green   Excellent   591
2               4.4         Green   Very Good   270
3               4.9    Dark Green   Excellent   365
4               4.8    Dark Green   Excellent   229

[5 rows x 21 columns]
```

[1]:
```python
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

Checking for the number of rows and columns and also for the presence of null values and dropping them and re-evaluating the number of rows and columns

[ ]:
```python
#checking for null values and rows and columns
df.describe()
print('num_rows, num_columns= ',df.shape)
df.isnull().sum()
```

```
num_rows, num_columns=  (9551, 21)
```

[ ]:
```
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
```

[ ]:
```python
drop=df.dropna()
print('num_rows, num_columns= ',drop.shape)
```

```
num_rows, num_columns=  (9542, 21)
```

Descriptive Analysis

[ ]:
```python
#Measures of central tendency and dispersion among numerical columns
df.cleaned=df.drop(columns=['Latitude','Longitude','Restaurant ID','Country␣
↪Code'])
```

```
df.cleaned.describe()
```

```
<ipython-input-132-b5746366f825>:2: UserWarning:

Pandas doesn't allow columns to be created via a new attribute name - see
https://pandas.pydata.org/pandas-docs/stable/indexing.html#attribute-access
```

[ ]:

	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

Distribution of Categorical Variables

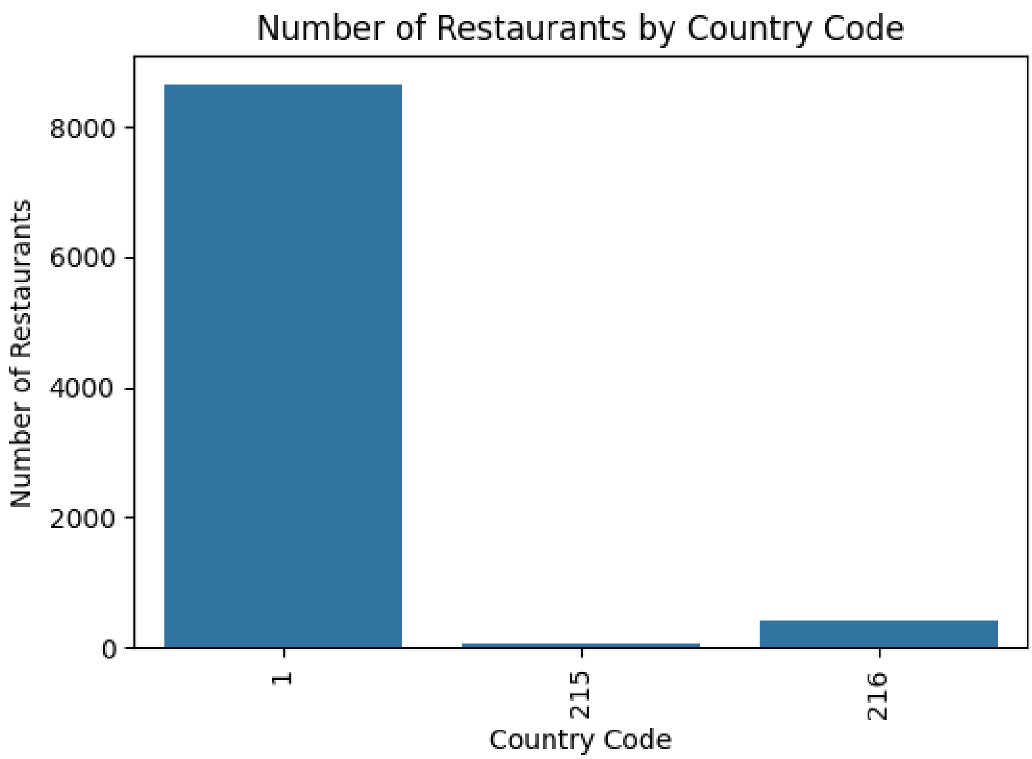
[ ]:

```
country_distribution=df['Country Code'].value_counts()
print(country_distribution)
country_distribution=df['Country Code'].value_counts().head(3)
print('top 3 Country Codes with highest number of restaurants:␣
↪',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
top 3 Country Codes with highest number of restaurants:  Country Code
1      8652
216     434
215      80
Name: count, dtype: int64
```

```
[ ]: plt.figure(figsize=(6, 4))
     sns.barplot(x=country_distribution.index, y=country_distribution.values)
     plt.xlabel('Country Code')
     plt.ylabel('Number of Restaurants')
     plt.title('Number of Restaurants by Country Code')
     plt.xticks(rotation=90)
```

```
[ ]: ([0, 1, 2], [Text(0, 0, '1'), Text(1, 0, '215'), Text(2, 0, '216')])
```



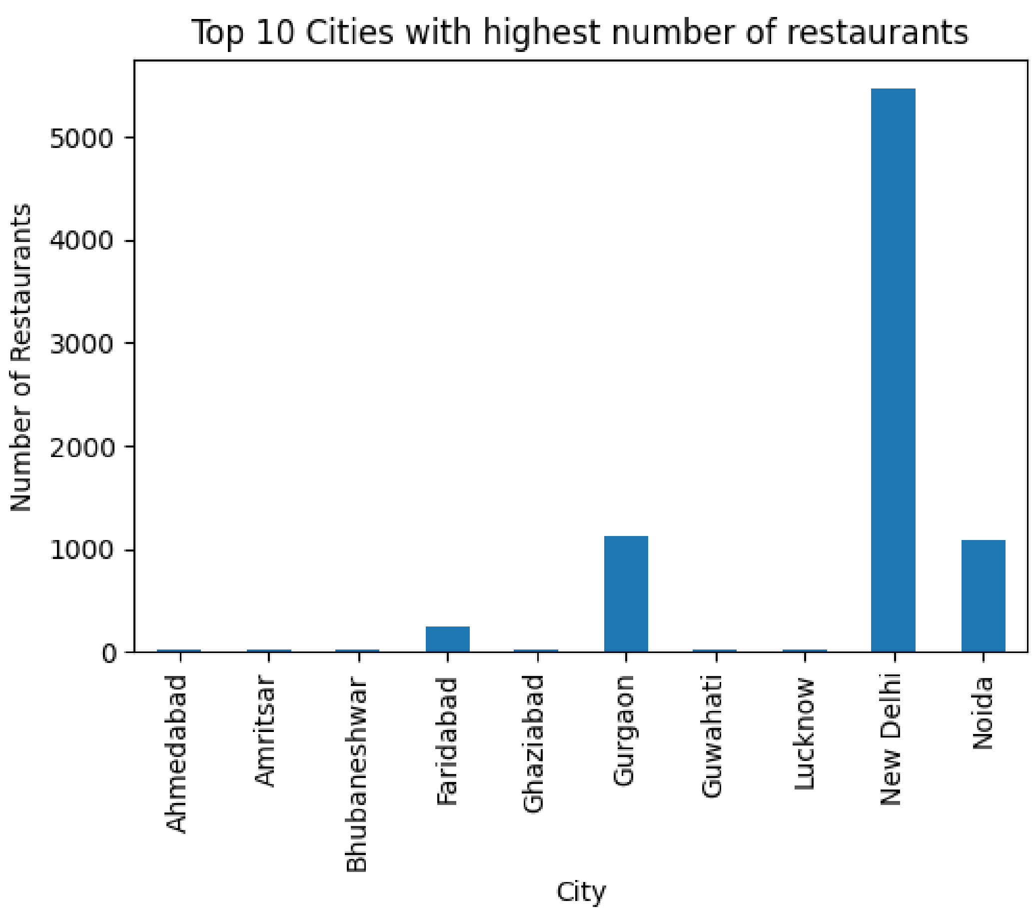

```
[ ]: City_Count= df['City'].value_counts()
     print('Count_of_Cities', City_Count)
     City_Count=df['City'].value_counts().head(3)
     print('Top 3 Cities with highest number of restaurants: ',City_Count)
```

```
Count_of_Cities 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
Top 3 Cities with highest number of restaurants:  City
New Delhi     5473
Gurgaon       1118
Noida         1080
Name: count, dtype: int64
```

[ ]:
```
City_Count=df['City'].value_counts().head(10).sort_index().
  ↪plot(kind='bar',title='Top 10 Cities with highest number of restaurants',␣
  ↪figsize=(6,4))
plt.xlabel('City')
plt.ylabel('Number of Restaurants')
```

[ ]:
```
Text(0, 0.5, 'Number of Restaurants')
```

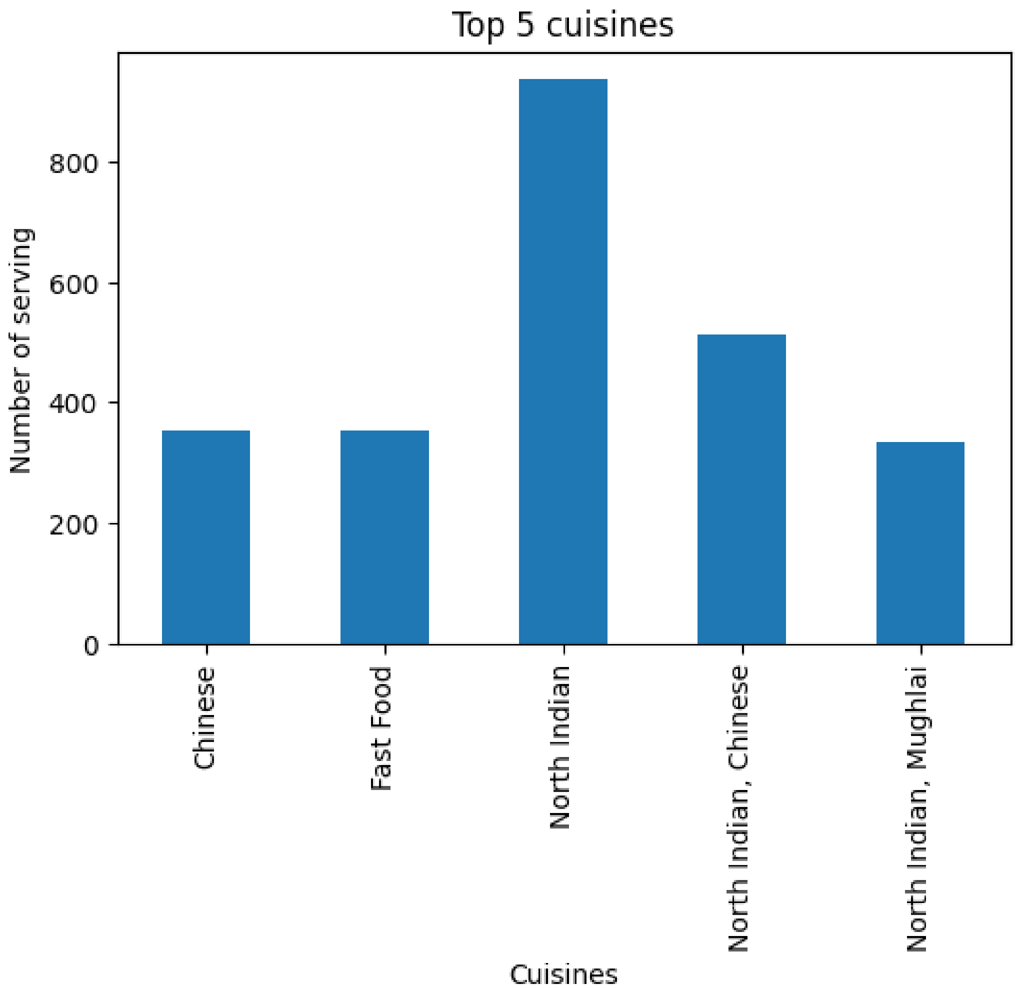


[ ]:
```
Cuisines=df['Cuisines'].value_counts()
print(Cuisines)
Cuisines=df['Cuisines'].value_counts().head(5)
print('Top 5 Cuisines that they serve: ',Cuisines)
```

```
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: 1825, dtype: int64
Top 5 Cuisines that they serve:  Cuisines
North Indian             936
North Indian, Chinese    511
Chinese                  354
Fast Food                354
North Indian, Mughlai    334
Name: count, dtype: int64
```

[ ]:
```
Cuisines=df['Cuisines'].value_counts().head(5).sort_index().
  ↪plot(kind='bar',title='Top 5 cuisines', figsize=(6,4))
plt.xlabel('Cuisines')
plt.ylabel('Number of serving')
```

[ ]:
```
Text(0, 0.5, 'Number of serving')
```



[ ]:
```
df['Rating color'].value_counts()
```

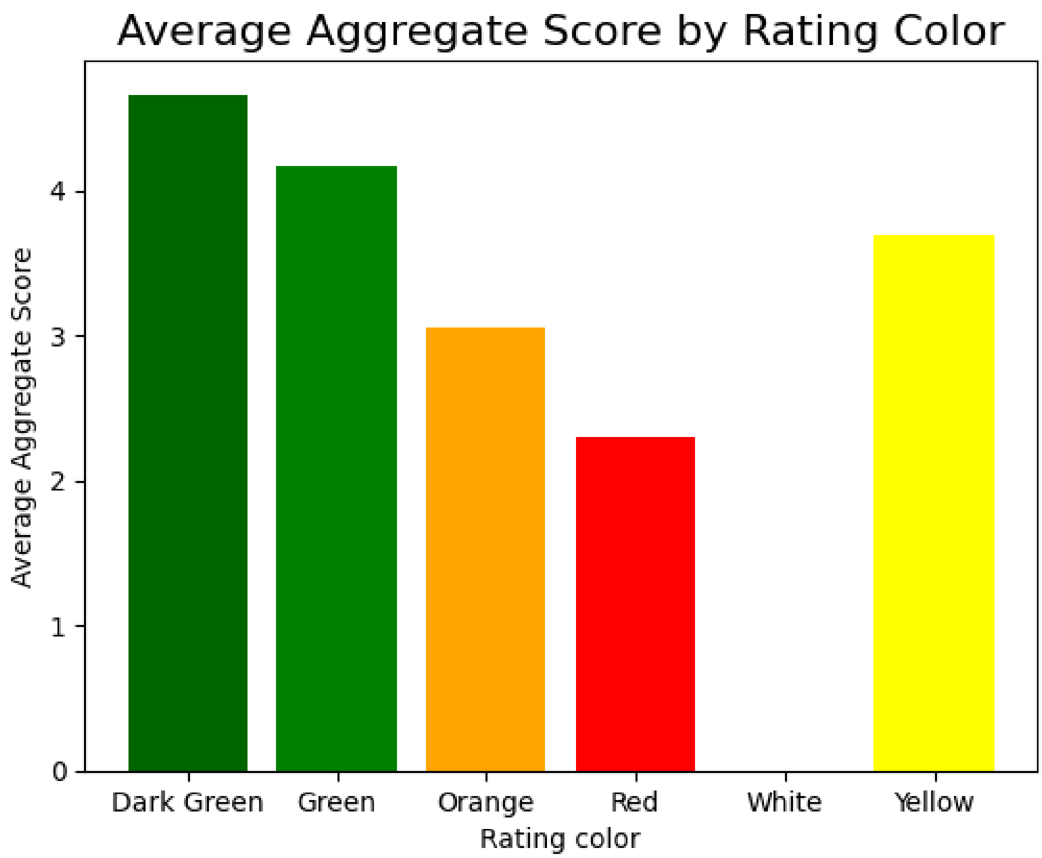
[ ]:
```
Rating color
Orange        3737
White         2148
Yellow        2100
Green         1079
Dark Green     301
Red            186
Name: count, dtype: int64
```

[ ]:
```
Rating_Avg=df.groupby('Rating color')['Aggregate rating'].mean()
print(Rating_Avg)
Rating_Avg=Rating_Avg.sort_index()
color_map = {
    'Red': 'red',
```

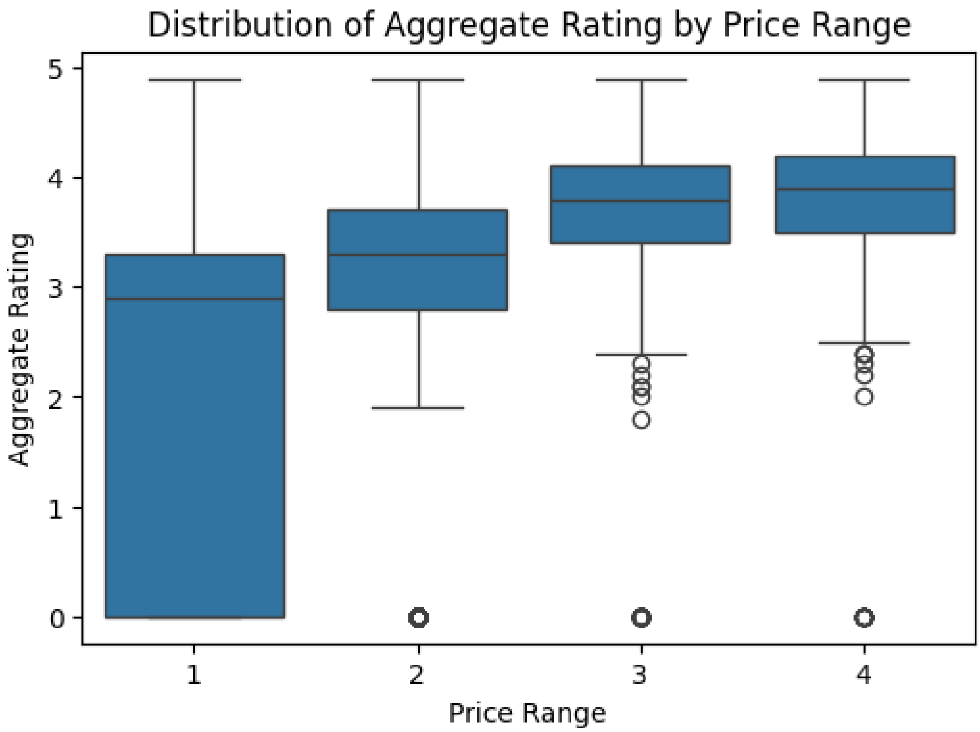
```python
    'Green': 'green',
    'Blue': 'blue',
    'Yellow': 'yellow',
    'Orange': 'orange',
    'White': 'white',
    'Dark Green': 'darkgreen'
}

plt.xlabel('Rating color', fontsize=10)
plt.ylabel('Average Aggregate Score', fontsize=10)
plt.title('Average Aggregate Score by Rating Color', fontsize=16)
plt.bar(Rating_Avg.index, Rating_Avg.values, color=[color_map[color] for color␣
 ↪in Rating_Avg.index])
plt.show()
```

```
Rating color
Dark Green    4.659801
Green         4.168119
Orange        3.051619
Red           2.297849
White         0.000000
Yellow        3.683429
Name: Aggregate rating, dtype: float64
```



[ ]:
```
# Visualize the distribution of 'Aggregate rating' for different 'Price range'␣
↪categories
plt.figure(figsize=(6, 4))
sns.boxplot(x='Price range', y='Aggregate rating', data=df)
plt.title('Distribution of Aggregate Rating by Price Range')
plt.xlabel('Price Range')
plt.ylabel('Aggregate Rating')
plt.show()
```




```
[ ]: # Calculating the correlation coefficient
correlation = df['Votes'].corr(df['Aggregate rating'])

print("Correlation between Votes and Aggregate rating:", correlation)

# Creating a scatter plot to visualize the relationship
fig=px.scatter(df,y='Votes', x='Aggregate rating', trendline='ols')
fig.update_layout(title='Correlation between Votes and Aggregate␣
↪rating',width=800, height=600)
fig.update_yaxes(title_text='Number of Votes')
fig.update_xaxes(title_text='Aggregate Rating')
fig.show()
```

```
Correlation between Votes and Aggregate rating: 0.31369058419541157
```

```
[ ]: import plotly.express as px
fig=px.scatter_mapbox(
        df,
    lat="Latitude",
    lon="Longitude",
    hover_name="Restaurant Name",
    hover_data=["City", "Country Code", "Aggregate rating"],
    color="Aggregate rating",
```

```
    zoom=3,
    height=400,
        width=800
    )


fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

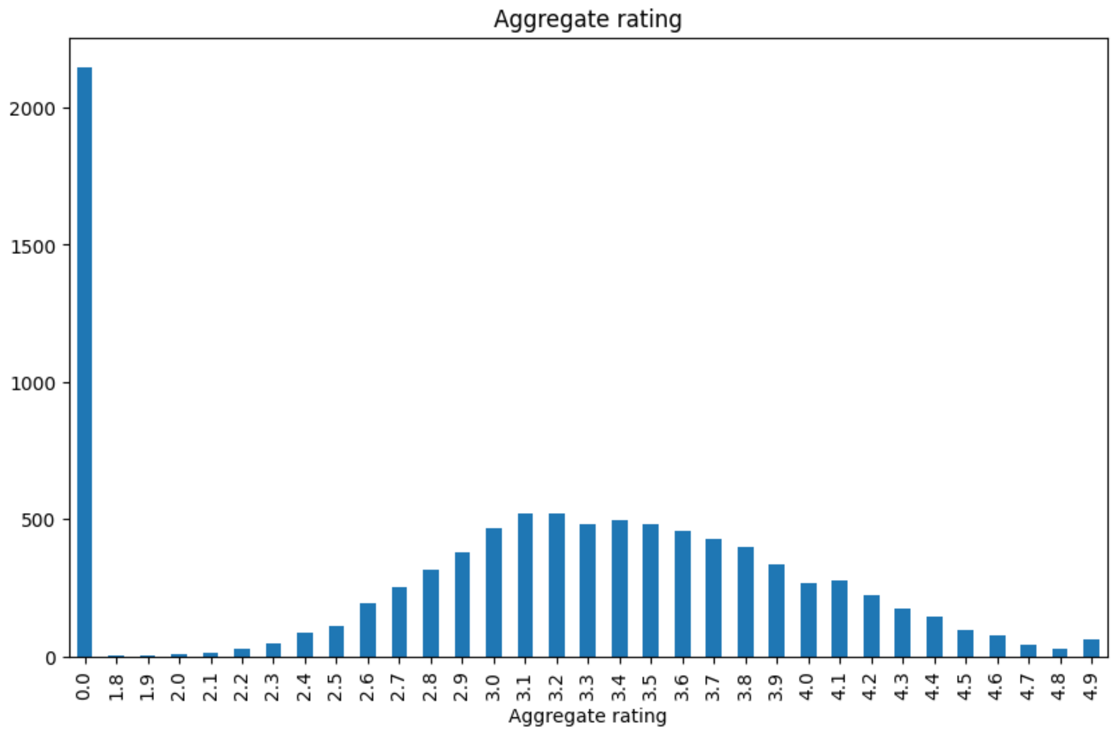
#**Sentiment Analysis**

[ ]:
```
df['Rating text'].value_counts()
```

[ ]:
```
Rating text
Average      3737
Not rated    2148
Good         2100
Very Good    1079
Excellent     301
Poor          186
Name: count, dtype: int64
```

[ ]:
```
ax=df['Aggregate rating'].value_counts().sort_index().
 ↪plot(kind='bar',title='Aggregate rating', figsize=(10,6))
plt.xlabel('Aggregate rating')
```

[ ]:
```
Text(0.5, 0, 'Aggregate rating')
```



Aggregate Ratings form somewhat of a bell shaped distribution with aratings typically clustering around the range of 3-4

[ ]:
```
df['Rating text'].value_counts()
```

[ ]:
```
Rating text
Average      3737
Not rated    2148
Good         2100
Very Good    1079
Excellent     301
Poor          186
Name: count, dtype: int64
```

[ ]:
```
ax=df['Rating text'].value_counts().sort_index().plot(kind='bar',title='Rating␣
 ↪text        ', figsize=(6,4))
plt.xlabel('Rating text')
plt.ylabel('Number of restaurants')
```

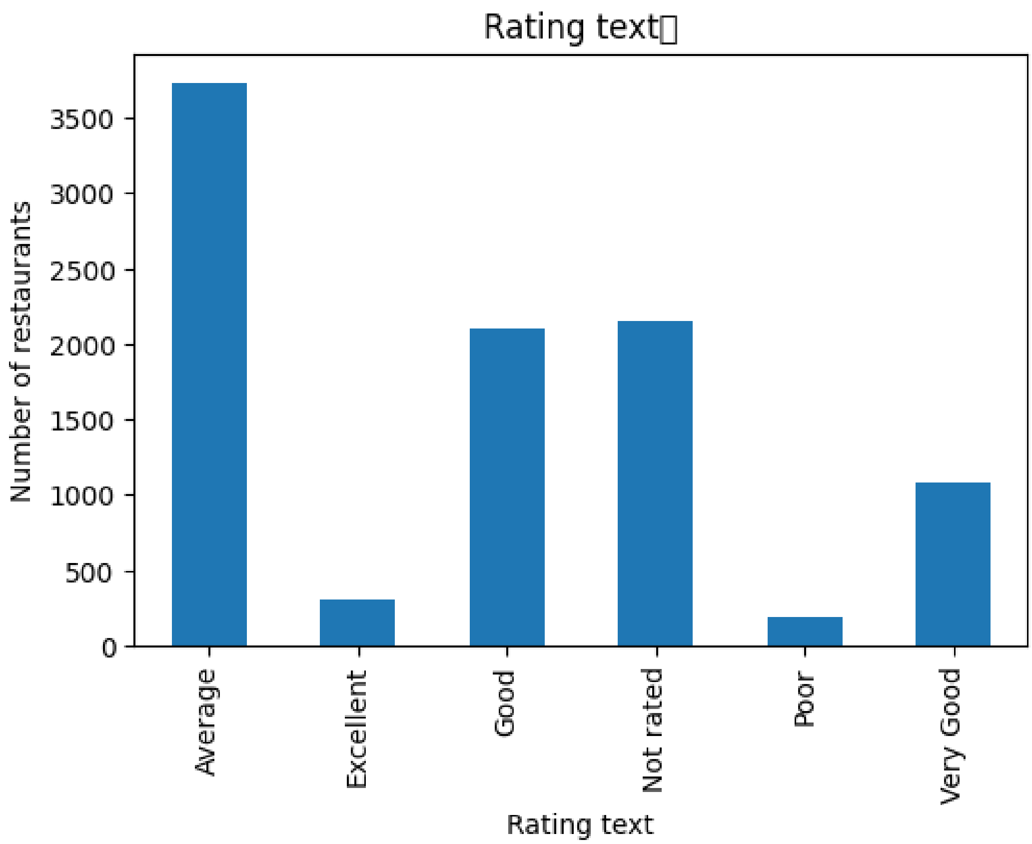
[ ]:
```
Text(0, 0.5, 'Number of restaurants')
```

```
/usr/local/lib/python3.10/dist-packages/IPython/core/events.py:89: UserWarning:

Glyph 9 (       ) missing from current font.
```

```
/usr/local/lib/python3.10/dist-packages/IPython/core/pylabtools.py:151:
UserWarning:

Glyph 9 (       ) missing from current font.
```



[ ]:
```python
import nltk
nltk.download('vader_lexicon')
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm
sia=SentimentIntensityAnalyzer()
```

```
[nltk_data] Downloading package vader_lexicon to /root/nltk_data…
[nltk_data]   Package vader_lexicon is already up-to-date!
```

[ ]:
```python
sia.polarity_scores('Excellent')
```

[ ]:
```
{'neg': 0.0, 'neu': 0.0, 'pos': 1.0, 'compound': 0.5719}
```

[ ]:
```
sia.polarity_scores('Not Rated')
```

[ ]:
```
{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
```

[ ]:
```
outcome={}
for i, row in tqdm(df.iterrows(),total=len(df)):
  text=row['Rating text']
  Id=row['Restaurant ID']
  outcome[Id]=sia.polarity_scores(text)
```

```
  0%|          | 0/9551 [00:00<?, ?it/s]
```

[ ]:
```
Sentiment_Analysis_df=pd.DataFrame(outcome).T
Sentiment_Analysis_df=Sentiment_Analysis_df.reset_index().
 ↪rename(columns={'index':'Restaurant ID'})
Sentiment_Analysis_df=Sentiment_Analysis_df.merge(df,how='left')
```

[ ]:
```
Sentiment_Analysis_df.head()
```

[ ]:
```
   Restaurant ID  neg    neu    pos  compound         Restaurant Name  \
0        6317637  0.0  0.000  1.000    0.5719        Le Petit Souffle
1        6304287  0.0  0.000  1.000    0.5719        Izakaya Kikufuji
2        6300002  0.0  0.238  0.762    0.4927  Heat - Edsa Shangri-La
3        6318506  0.0  0.000  1.000    0.5719                    Ooma
4        6314302  0.0  0.000  1.000    0.5719             Sambo Kojin

   Country Code              City  \
0           162       Makati City
1           162       Makati City
2           162  Mandaluyong City
3           162  Mandaluyong City
4           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…
3  Third Floor, Mega Fashion Hall, SM Megamall, O…
4  Third Floor, Mega Atrium, SM Megamall, Ortigas…

                                    Locality  …          Currency  \
0   Century City Mall, Poblacion, Makati City  …  Botswana Pula(P)
1  Little Tokyo, Legaspi Village, Makati City  …  Botswana Pula(P)
2  Edsa Shangri-La, Ortigas, Mandaluyong City  …  Botswana Pula(P)
3      SM Megamall, Ortigas, Mandaluyong City  …  Botswana Pula(P)
4      SM Megamall, Ortigas, Mandaluyong City  …  Botswana Pula(P)
```

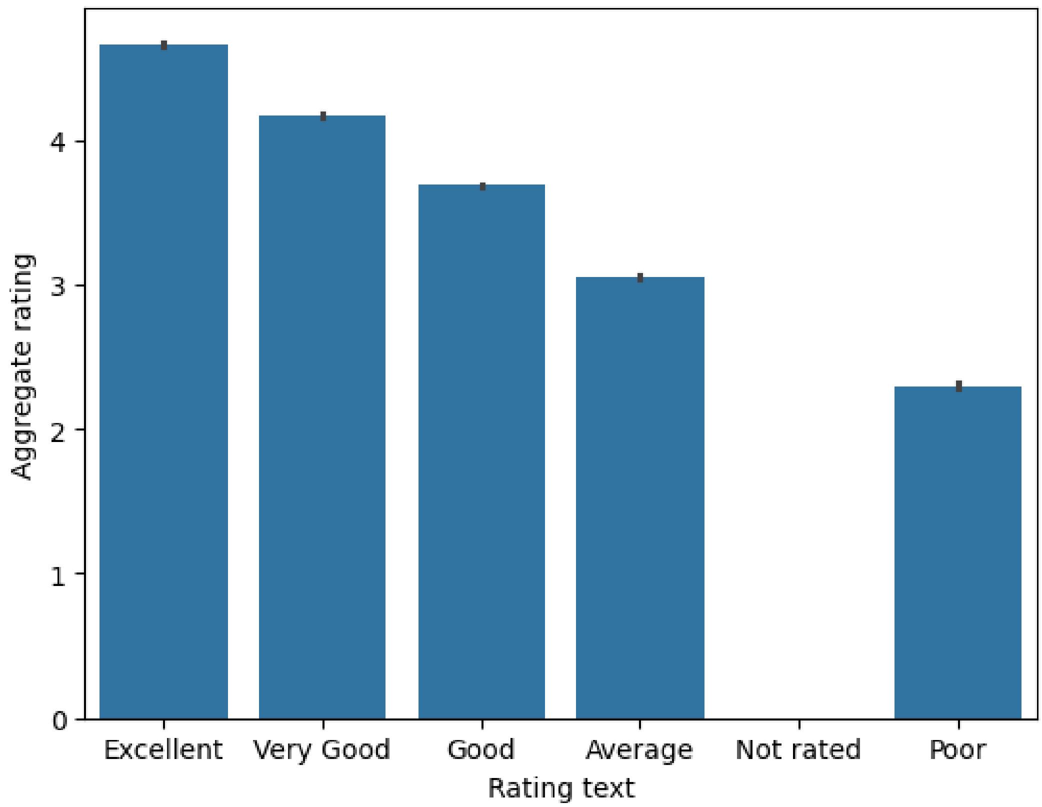
```
   Has Table booking  Has Online delivery Is delivering now  \
0               Yes                  No                No
1               Yes                  No                No
2               Yes                  No                No
3                No                  No                No
4               Yes                  No                No

   Switch to order menu Price range Aggregate rating Rating color Rating text  \
0                    No           3              4.8   Dark Green   Excellent
1                    No           3              4.5   Dark Green   Excellent
2                    No           4              4.4        Green   Very Good
3                    No           4              4.9   Dark Green   Excellent
4                    No           4              4.8   Dark Green   Excellent

  Votes
0   314
1   591
2   270
3   365
4   229

[5 rows x 25 columns]
```

[ ]:
```
px=sns.barplot(data=Sentiment_Analysis_df,x='Rating text',y='Aggregate rating')
```



The reviews with most positive texts are associated with the highest ratings and hence satisfies the virtues of sentiment analysis

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

[ ]:
```
   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
3        6318506                     Ooma           162  Mandaluyong City
4        6314302              Sambo Kojin           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…
3  Third Floor, Mega Fashion Hall, SM Megamall, O…
4  Third Floor, Mega Atrium, SM Megamall, Ortigas…

                                    Locality  \
0   Century City Mall, Poblacion, Makati City
```

```
1  Little Tokyo, Legaspi Village, Makati City
2  Edsa Shangri-La, Ortigas, Mandaluyong City
3      SM Megamall, Ortigas, Mandaluyong City
4      SM Megamall, 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
3  SM Megamall, Ortigas, Mandaluyong City, Mandal…  121.056475  14.585318
4  SM Megamall, Ortigas, Mandaluyong City, Mandal…  121.057508  14.584450

                          Cuisines  …         Currency Has Table booking  \
0       French, Japanese, Desserts  …  Botswana Pula(P)               Yes
1                         Japanese  …  Botswana Pula(P)               Yes
2  Seafood, Asian, Filipino, Indian  …  Botswana Pula(P)               Yes
3                  Japanese, Sushi  …  Botswana Pula(P)                No
4                 Japanese, Korean  …  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
2                  No                No                   No           4
3                  No                No                   No           4
4                  No                No                   No           4

   Aggregate rating  Rating color Rating text Votes
0               4.8    Dark Green   Excellent   314
1               4.5    Dark Green   Excellent   591
2               4.4         Green   Very Good   270
3               4.9    Dark Green   Excellent   365
4               4.8    Dark Green   Excellent   229

[5 rows x 21 columns]
```

[ ]:
```python
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from collections import Counter
import re
import nltk
nltk.download('stopwords')
nltk.download('punkt')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Package punkt is already up-to-date!
```

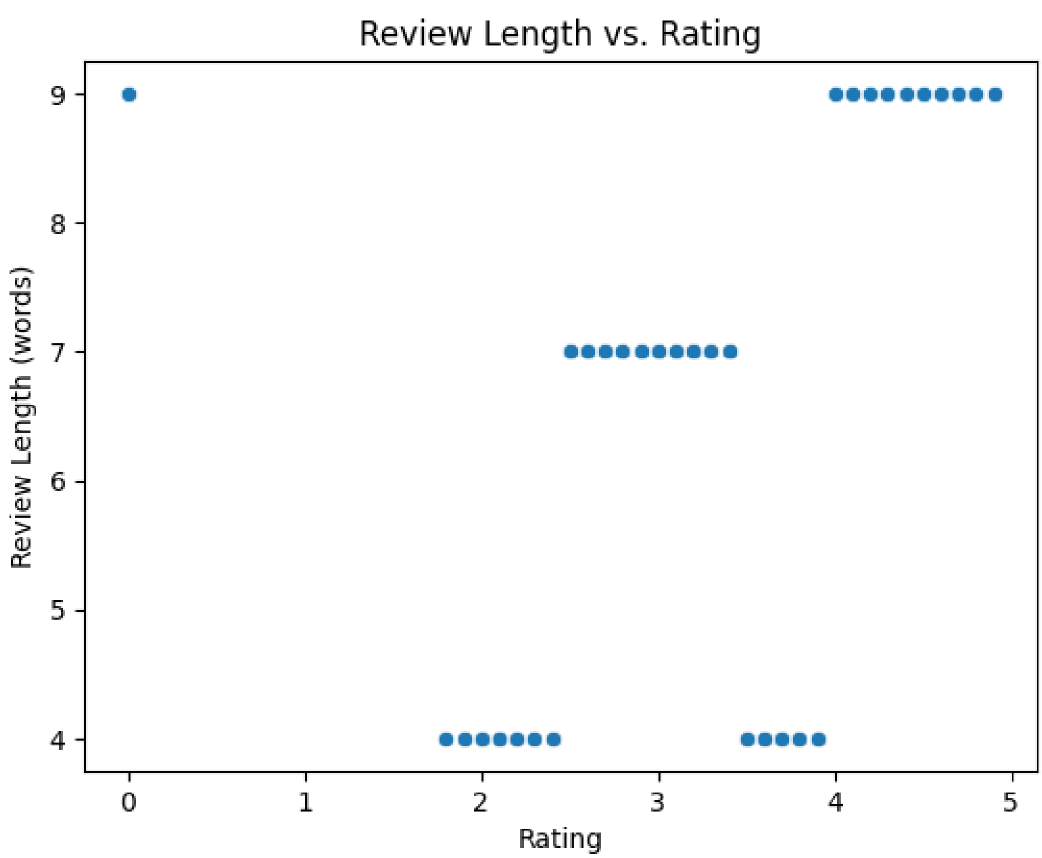
[ ]: True

```
[ ]: df['rating_length'] = df['Rating text'].apply(len)
```

```
[ ]: average_length = df['rating_length'].mean()
         print(f"Average review length: {average_length:.2f} words")
         correlation = df['Aggregate rating'].corr(df['rating_length'])
         correlation_coefficient=np.corrcoef(df['Aggregate rating'],␣
       ↪df['rating_length'])[0, 1]
         print(f"Correlation coefficient: {correlation_coefficient:.2f}")
```

```
Average review length: 7.02 words
Correlation coefficient: -0.48
```

```
[ ]: sns.scatterplot(data=df, x='Aggregate rating', y='rating_length')
     plt.xlabel('Rating')
     plt.ylabel('Review Length (words)')
     plt.title('Review Length vs. Rating')
     plt.show()
```



Hence there exists a significant negative correlation between review length and the reiews the restaurant recieves

#**Model Training**

[ ]:

```

```

[ ]:

```
#feature selection
df.cleaned=df.drop(columns=['Latitude','Longitude','Restaurant ID','Country␣
 ↪Code','Address','Locality','Locality Verbose','Switch to order␣
 ↪menu','Restaurant Name'])
df.cleaned.head()
```

[ ]:

```
              City                          Cuisines  Average Cost for two  \
0      Makati City        French, Japanese, Desserts                  1100
1      Makati City                          Japanese                  1200
2  Mandaluyong City  Seafood, Asian, Filipino, Indian                  4000
3  Mandaluyong City                   Japanese, Sushi                  1500
4  Mandaluyong City                  Japanese, Korean                  1500

           Currency Has Table booking Has Online delivery Is delivering now  \
0  Botswana Pula(P)               Yes                  No                No
1  Botswana Pula(P)               Yes                  No                No
2  Botswana Pula(P)               Yes                  No                No
3  Botswana Pula(P)                No                  No                No
4  Botswana Pula(P)               Yes                  No                No

   Price range  Aggregate rating Rating color Rating text  Votes  \
0            3               4.8   Dark Green   Excellent    314
1            3               4.5   Dark Green   Excellent    591
2            4               4.4        Green   Very Good    270
3            4               4.9   Dark Green   Excellent    365
4            4               4.8   Dark Green   Excellent    229

   rating_length
0              9
1              9
2              9
3              9
4              9
```

[ ]:

```
#creating dummies for categorical variables
dummies=pd.get_dummies(df.cleaned,columns=['Cuisines','City','Has Table␣
 ↪booking','Has Online delivery','Is delivering now','Rating color','Rating␣
 ↪text','Price range'])
dummies.head()
```

```
[ ]:    Average Cost for two          Currency  Aggregate rating  Votes  \
    0                   1100  Botswana Pula(P)               4.8    314
    1                   1200  Botswana Pula(P)               4.5    591
    2                   4000  Botswana Pula(P)               4.4    270
    3                   1500  Botswana Pula(P)               4.9    365
    4                   1500  Botswana Pula(P)               4.8    229

       rating_length  Cuisines_Afghani  Cuisines_Afghani, Mughlai, Chinese  \
    0              9             False                               False
    1              9             False                               False
    2              9             False                               False
    3              9             False                               False
    4              9             False                               False

       Cuisines_Afghani, North Indian  \
    0                           False
    1                           False
    2                           False
    3                           False
    4                           False

       Cuisines_Afghani, North Indian, Pakistani, Arabian  Cuisines_African  …  \
    0                                              False             False  …
    1                                              False             False  …
    2                                              False             False  …
    3                                              False             False  …
    4                                              False             False  …

       Rating text_Average  Rating text_Excellent  Rating text_Good  \
    0                False                   True             False
    1                False                   True             False
    2                False                  False             False
    3                False                   True             False
    4                False                   True             False

       Rating text_Not rated  Rating text_Poor  Rating text_Very Good  \
    0                  False             False                  False
    1                  False             False                  False
    2                  False             False                   True
    3                  False             False                  False
    4                  False             False                  False

       Price range_1  Price range_2  Price range_3  Price range_4
    0          False          False           True          False
    1          False          False           True          False
    2          False          False          False           True
    3          False          False          False           True
```

```
4          False          False          False          True

[5 rows x 1993 columns]
```

dropping dummy variables in order to deal with the problem of dummy variable trap

```
[ ]: dummies_drop=dummies.drop(['Currency','Cuisines_Afghani','Rating␣
      ↪text_Average','City_Ankara','Price range_1','Rating color_Yellow'],axis=1)
     dummies_drop.head()
```

```
[ ]:   Average Cost for two  Aggregate rating  Votes  rating_length  \
     0                 1100               4.8    314              9
     1                 1200               4.5    591              9
     2                 4000               4.4    270              9
     3                 1500               4.9    365              9
     4                 1500               4.8    229              9

        Cuisines_Afghani, Mughlai, Chinese  Cuisines_Afghani, North Indian  \
     0                               False                           False
     1                               False                           False
     2                               False                           False
     3                               False                           False
     4                               False                           False

        Cuisines_Afghani, North Indian, Pakistani, Arabian  Cuisines_African  \
     0                                              False             False
     1                                              False             False
     2                                              False             False
     3                                              False             False
     4                                              False             False

        Cuisines_African, Portuguese  Cuisines_American  …  Rating color_Red  \
     0                         False              False  …             False
     1                         False              False  …             False
     2                         False              False  …             False
     3                         False              False  …             False
     4                         False              False  …             False

        Rating color_White  Rating text_Excellent  Rating text_Good  \
     0               False                   True             False
     1               False                   True             False
     2               False                  False             False
     3               False                   True             False
     4               False                   True             False

        Rating text_Not rated  Rating text_Poor  Rating text_Very Good  \
     0                  False             False                  False
```

```
1                  False             False                      False
2                  False             False                       True
3                  False             False                      False
4                  False             False                      False

   Price range_2  Price range_3  Price range_4
0          False           True          False
1          False           True          False
2          False          False           True
3          False          False           True
4          False          False           True

[5 rows x 1987 columns]
```

Dropping Dummies to solve the problem of Dummy Variable Trap

[ ]:
```
#splitting the model into training and test data
model=LinearRegression()
X=dummies_drop.drop(columns=['Aggregate rating'])
Y=dummies_drop['Aggregate rating']
```

[ ]:
```
X.head()
```

[ ]:
```
   Average Cost for two  Votes  rating_length  \
0                  1100    314              9
1                  1200    591              9
2                  4000    270              9
3                  1500    365              9
4                  1500    229              9

   Cuisines_Afghani, Mughlai, Chinese  Cuisines_Afghani, North Indian  \
0                               False                           False
1                               False                           False
2                               False                           False
3                               False                           False
4                               False                           False

   Cuisines_Afghani, North Indian, Pakistani, Arabian  Cuisines_African  \
0                                              False             False
1                                              False             False
2                                              False             False
3                                              False             False
4                                              False             False

   Cuisines_African, Portuguese  Cuisines_American  \
0                         False              False
1                         False              False
```

```
2                    False           False
3                    False           False
4                    False           False

   Cuisines_American, Asian, Burger  …  Rating color_Red  \
0                              False  …             False
1                              False  …             False
2                              False  …             False
3                              False  …             False
4                              False  …             False

   Rating color_White  Rating text_Excellent  Rating text_Good  \
0               False                   True             False
1               False                   True             False
2               False                  False             False
3               False                   True             False
4               False                   True             False

   Rating text_Not rated  Rating text_Poor  Rating text_Very Good  \
0                  False             False                  False
1                  False             False                  False
2                  False             False                   True
3                  False             False                  False
4                  False             False                  False

   Price range_2  Price range_3  Price range_4
0          False           True          False
1          False           True          False
2          False          False           True
3          False          False           True
4          False          False           True

[5 rows x 1986 columns]
```

[ ]:
```
Y.head()
```

[ ]:
```
0    4.8
1    4.5
2    4.4
3    4.9
4    4.8
Name: Aggregate rating, dtype: float64
```

[ ]:
```
#Dropping the values which have very low frequency in order to perform␣
↪stratified sampling i.e ensuring that our sample is split into homogenous␣
↪group
class_counts = Y.value_counts()
```

```
print(class_counts)
classes_to_keep = class_counts[class_counts > 7].index
df_filtered= dummies_drop[dummies_drop['Aggregate rating'].
 ↪isin(classes_to_keep)]
```

```
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
```

[ ]:
```
model=LinearRegression()
X_filtered=df_filtered.drop(columns=['Aggregate rating'])
Y_filtered=df_filtered['Aggregate rating']
```

[ ]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X_filtered,Y_filtered,␣
 ↪test_size=0.2, stratify=Y_filtered, random_state=3)
```

[ ]:
```
models=[LinearRegression(),DecisionTreeRegressor(),RandomForestRegressor()]
def compare_models_train_test():
 for model in models:
  model.fit(X_train,Y_train)
  test_data_prediction=model.predict(X_test)
  r2=r2_score(Y_test,test_data_prediction)
  print('r2 score of the ',model,' = ',r2)
```

[ ]:
```
compare_models_train_test()
```

```
r2 score of the  LinearRegression()  =  0.9856632694820852
r2 score of the  DecisionTreeRegressor()  =  0.9763927518471383
r2 score of the  RandomForestRegressor()  =  0.9860800631110995
```

From the R2 score it can be said that all the models perform significantly well for prediction however the r2 score depends on the values of randdom state and hence no conclusive result can be made

#**K Fold Cross Validation**

[ ]:
```
cv_lnr=cross_val_score(LinearRegression(),X_filtered,Y_filtered,scoring='r2',␣
 ↪cv=5)
```

[ ]:
```
print(cv_lnr)
mean_r2=sum(cv_lnr)/len(cv_lnr)
mean_r2=round(mean_r2,2)
mean_r2=mean_r2*100
print('mean r2 score: ',mean_r2)
```

```
[-2.01140081e+12  9.81740668e-01  9.84619068e-01 -1.06623296e+09
 -2.32970366e+13]
mean r2 score:  -506190072703361.06
```

[ ]:
```
cv_dec_tree=cross_val_score(DecisionTreeRegressor(),X_filtered,Y_filtered,scoring='r2',␣
 ↪cv=5)
```

[ ]:
```
print(cv_dec_tree)
print(cv_dec_tree)
mean_r2=sum(cv_dec_tree)/len(cv_dec_tree)
mean_r2=round(mean_r2,2)
mean_r2=mean_r2*100
print('mean r2 score: ',mean_r2)
```

```
[0.97155321 0.97003268 0.97588078 0.97495803 0.978779  ]
[0.97155321 0.97003268 0.97588078 0.97495803 0.978779  ]
mean r2 score:  97.0
```

[ ]:
```
cv_random_forest=cross_val_score(RandomForestRegressor(),X_filtered,Y_filtered,scoring='r2',␣
 ↪cv=5)
print(cv_random_forest)
```

```
[0.9826951  0.98335811 0.98546034 0.98555582 0.9863826 ]
```

[ ]:
```
mean_r2=sum(cv_random_forest)/len(cv_random_forest)
mean_r2=round(mean_r2,2)
mean_r2=mean_r2*100
print('mean r2 score: ',mean_r2)
```

```
mean r2 score:  98.0
```

The Random Forrest Regression model is found to be performing the best out of all with it predicting 98% of the variation of data

[ ]:
```
rfr=RandomForestRegressor(random_state=10)
rfr.fit(X_train,Y_train)
```

[ ]:
```
RandomForestRegressor(random_state=10)
```

[ ]:
```
Y_pred=rfr.predict(X_test)
```

[ ]:
```
mean_squared_error(Y_test,Y_pred)
```

[ ]:
```
0.03233801408003268
```

[ ]:
```
r2_score(Y_test,Y_pred)
```

[ ]:
```
0.9859493502202655
```

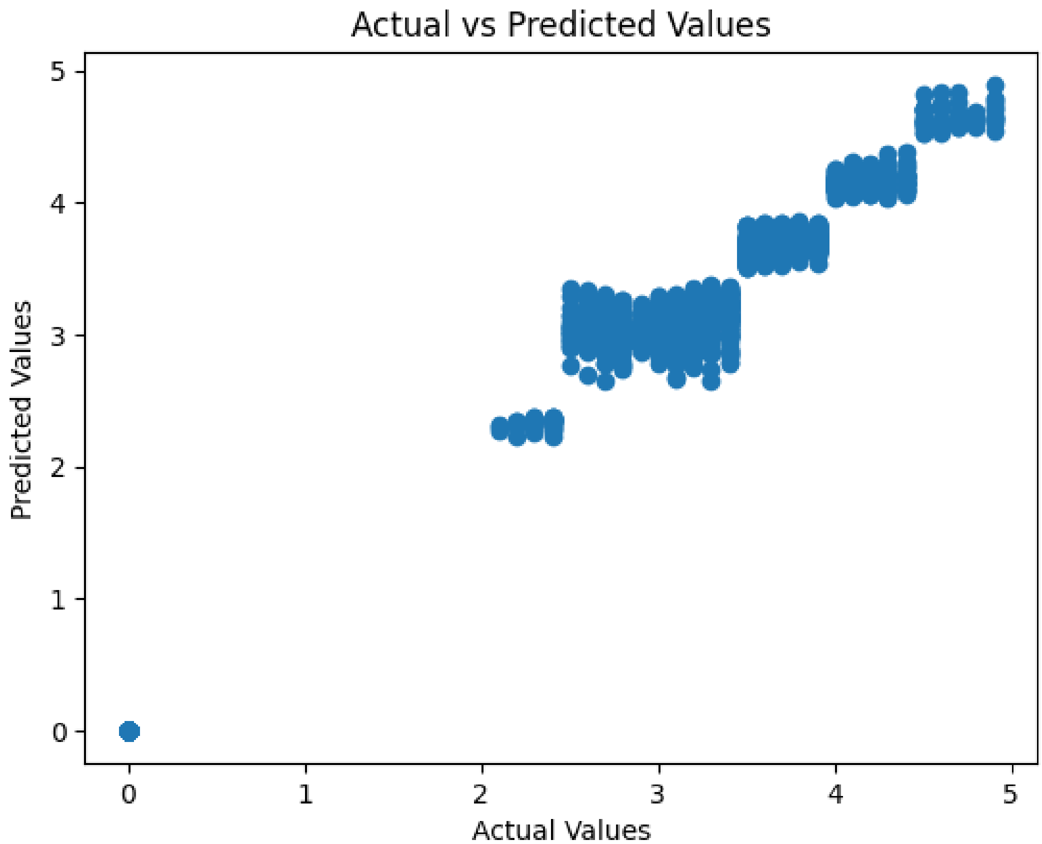
[ ]:
```
train_rmse = np.sqrt(mean_squared_error(Y_train, y_train_pred))
test_rmse = np.sqrt(mean_squared_error(Y_test, y_test_pred))

train_r2 = r2_score(Y_train, y_train_pred)
test_r2 = r2_score(Y_test, y_test_pred)

print("Train RMSE:", train_rmse)
print("Test RMSE:", test_rmse)
```

```
Train RMSE: 0.06964499085935906
Test RMSE: 0.1798277344572652
```

[ ]:
```
plt.scatter(Y_test,Y_pred)
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Actual vs Predicted Values')
plt.show()
```


Actual vs Predicted Values
Predicted Values
Actual Values
0
1
2
3
4
5

```
[ ]:
```