Cognifyz
Where Data Meets Intelligence

Nagpur, Maharashtra, India
cognifyztechnologies@gmail.com
www.cognifyz.com

Data Analysis Project Report

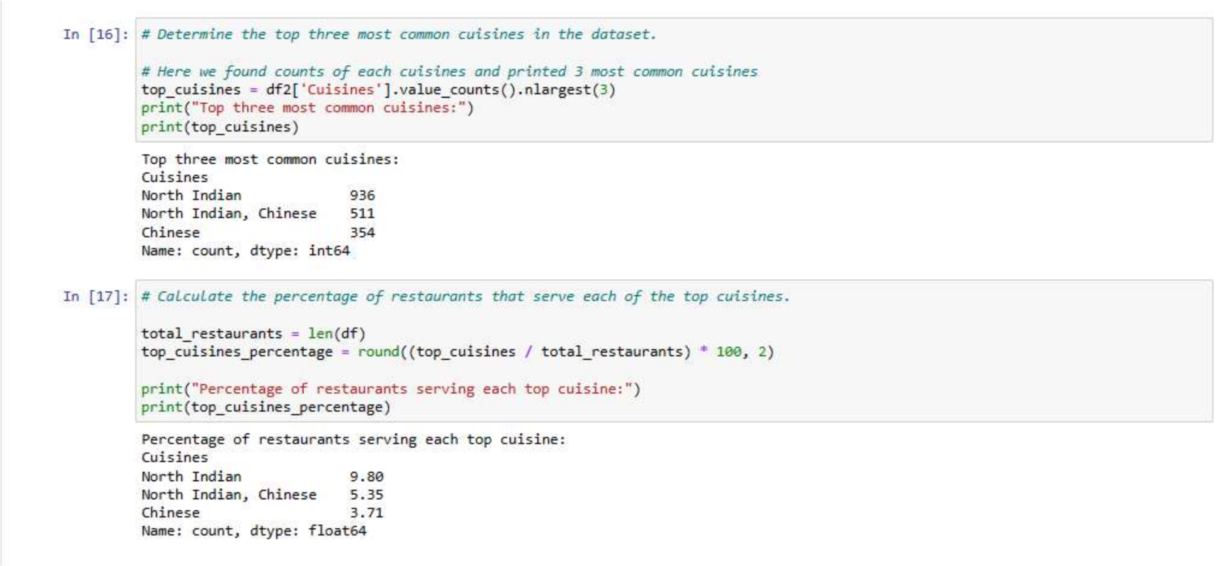
- Dataset: **Restaurant dataset**
- Task: Performing Data Analysis using python and python libraries
- Tools: Jupyter Notebook, Python and libraries – Pandas, Matplotlib, Pyplot, Numpy, Seaborn, Folium, scipy

Intern Details: -

- Name: - Ashwin Kadu
- Ref.: CTI/A1/C19147

Level 1 Task 1: Top Cuisines

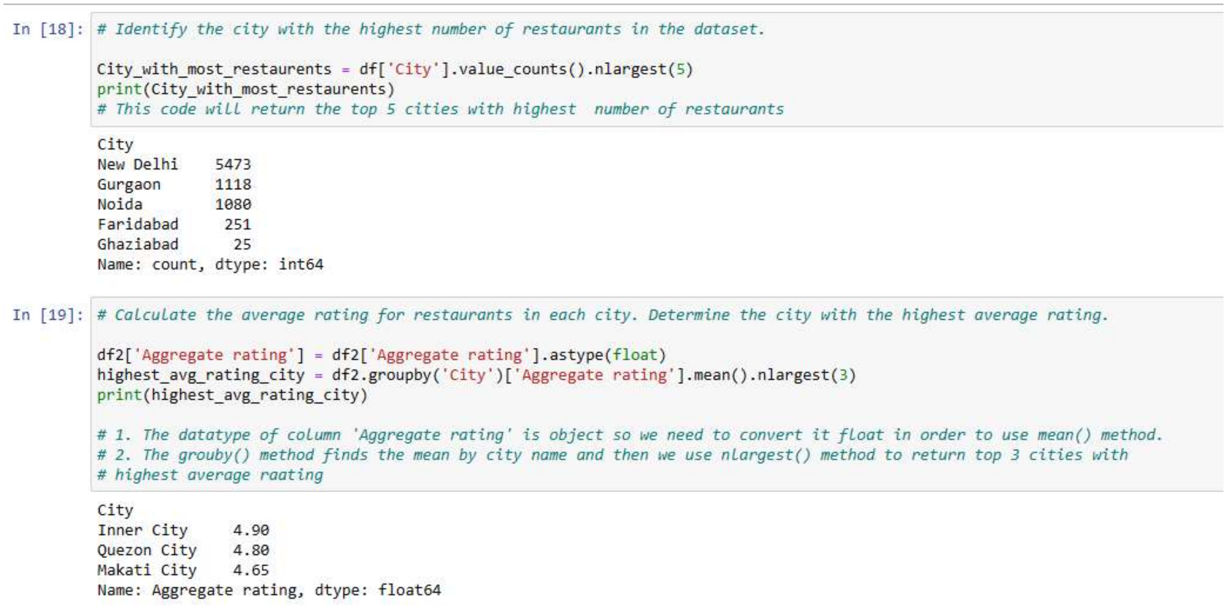
- Determine the top three most common cuisines in the dataset.
- Calculate the percentage of restaurants that serve each of the top cuisines



```
In [16]: # Determine the top three most common cuisines in the dataset.

         # Here we found counts of each cuisines and printed 3 most common cuisines
         top_cuisines = df2['Cuisines'].value_counts().nlargest(3)
         print("Top three most common cuisines:")
         print(top_cuisines)

         Top three most common cuisines:
         Cuisines
         North Indian             936
         North Indian, Chinese    511
         Chinese                  354
         Name: count, dtype: int64

In [17]: # Calculate the percentage of restaurants that serve each of the top cuisines.

         total_restaurants = len(df)
         top_cuisines_percentage = round((top_cuisines / total_restaurants) * 100, 2)

         print("Percentage of restaurants serving each top cuisine:")
         print(top_cuisines_percentage)

         Percentage of restaurants serving each top cuisine:
         Cuisines
         North Indian             9.80
         North Indian, Chinese    5.35
         Chinese                  3.71
         Name: count, dtype: float64
```

Level 1 Task 2: City Analysis

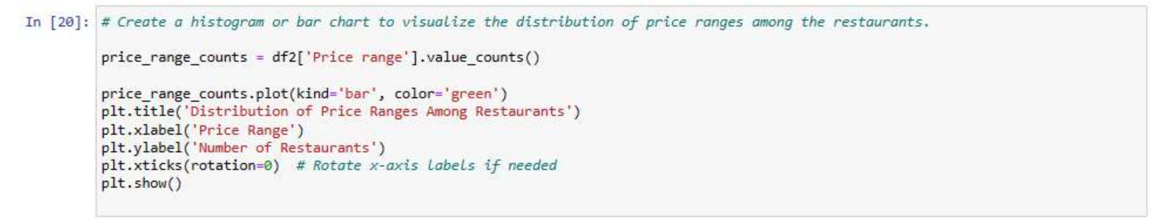
- Identify the city with the highest number of restaurants in the dataset.
- Calculate the average rating for restaurants in each city. Determine the city with the highest average rating.



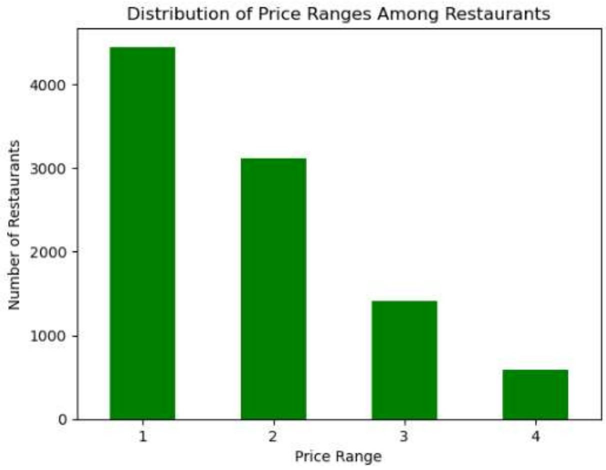
```
In [18]: # Identify the city with the highest number of restaurants in the dataset.

         City_with_most_restaurents = df['City'].value_counts().nlargest(5)
         print(City_with_most_restaurents)
         # This code will return the top 5 cities with highest  number of restaurants

         City
         New Delhi    5473
         Gurgaon      1118
         Noida        1080
         Faridabad     251
         Ghaziabad      25
         Name: count, dtype: int64

In [19]: # Calculate the average rating for restaurants in each city. Determine the city with the highest average rating.

         df2['Aggregate rating'] = df2['Aggregate rating'].astype(float)
         highest_avg_rating_city = df2.groupby('City')['Aggregate rating'].mean().nlargest(3)
         print(highest_avg_rating_city)

         # 1. The datatype of column 'Aggregate rating' is object so we need to convert it float in order to use mean() method.
         # 2. The grouby() method finds the mean by city name and then we use nlargest() method to return top 3 cities with
         # highest average raating

         City
         Inner City     4.90
         Quezon City    4.80
         Makati City    4.65
         Name: Aggregate rating, dtype: float64
```

Level 1 Task 3: Price Range Distribution

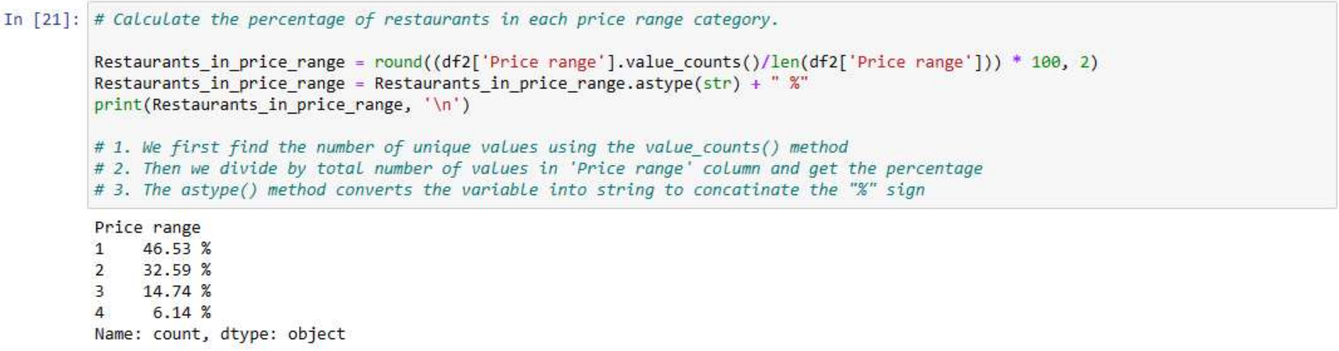
- Create a histogram or bar chart to visualize the distribution of price ranges among the restaurants.



```
In [20]: # Create a histogram or bar chart to visualize the distribution of price ranges among the restaurants.

         price_range_counts = df2['Price range'].value_counts()

         price_range_counts.plot(kind='bar', color='green')
         plt.title('Distribution of Price Ranges Among Restaurants')
         plt.xlabel('Price Range')
         plt.ylabel('Number of Restaurants')
         plt.xticks(rotation=0)  # Rotate x-axis labels if needed
         plt.show()
```



- Calculate the percentage of restaurants in each price range category.
-

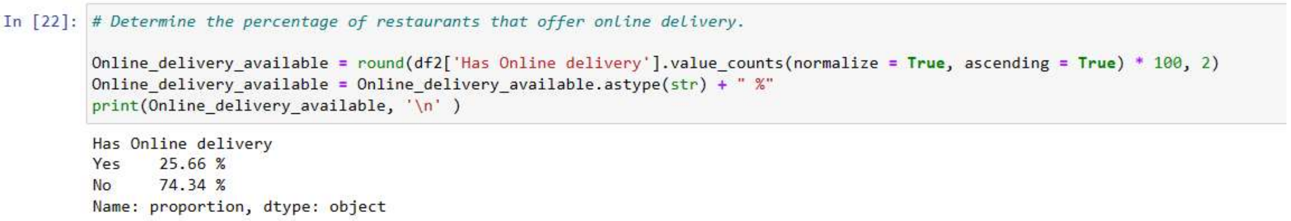


```
In [21]: # Calculate the percentage of restaurants in each price range category.

         Restaurants_in_price_range = round((df2['Price range'].value_counts()/len(df2['Price range'])) * 100, 2)
         Restaurants_in_price_range = Restaurants_in_price_range.astype(str) + " %"
         print(Restaurants_in_price_range, '\n')

         # 1. We first find the number of unique values using the value_counts() method
         # 2. Then we divide by total number of values in 'Price range' column and get the percentage
         # 3. The astype() method converts the variable into string to concatinate the "%" sign
```

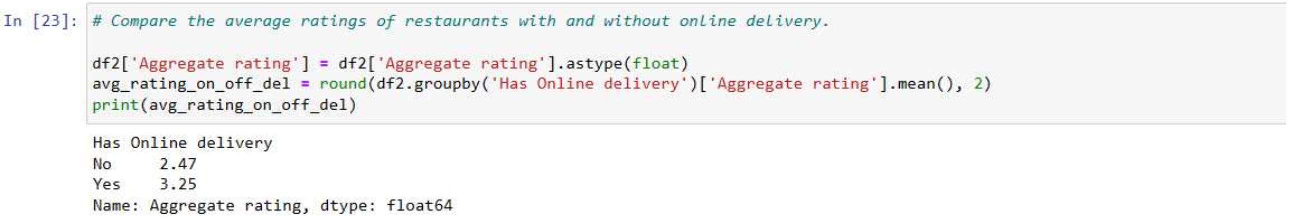
```
Price range
1    46.53 %
2    32.59 %
3    14.74 %
4     6.14 %
Name: count, dtype: object
```

Level 1 Task 4: Online Delivery

- Determine the percentage of restaurants that offer online delivery.
- Compare the average ratings of restaurants with and without online delivery.



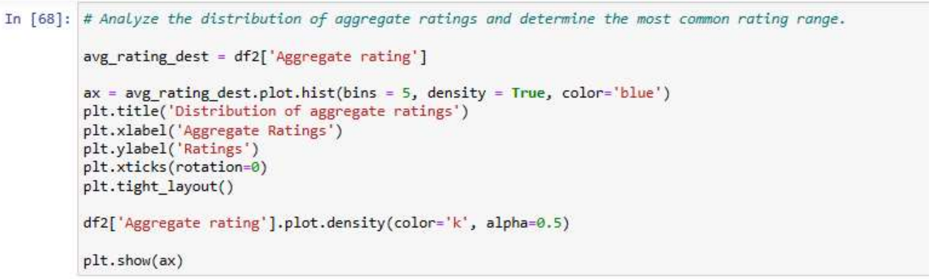
```
In [22]: # Determine the percentage of restaurants that offer online delivery.

         Online_delivery_available = round(df2['Has Online delivery'].value_counts(normalize = True, ascending = True) * 100, 2)
         Online_delivery_available = Online_delivery_available.astype(str) + " %"
         print(Online_delivery_available, '\n' )

         Has Online delivery
         Yes    25.66 %
         No     74.34 %
         Name: proportion, dtype: object
```



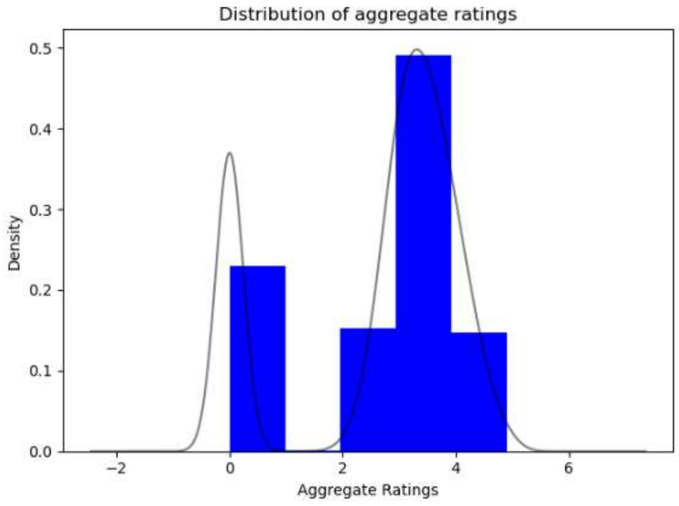
```
In [23]: # Compare the average ratings of restaurants with and without online delivery.

         df2['Aggregate rating'] = df2['Aggregate rating'].astype(float)
         avg_rating_on_off_del = round(df2.groupby('Has Online delivery')['Aggregate rating'].mean(), 2)
         print(avg_rating_on_off_del)

         Has Online delivery
         No     2.47
         Yes    3.25
         Name: Aggregate rating, dtype: float64
```

Level 2 Task 1: Restaurant Ratings

- Analyze the distribution of aggregate ratings and determine the most common rating range.



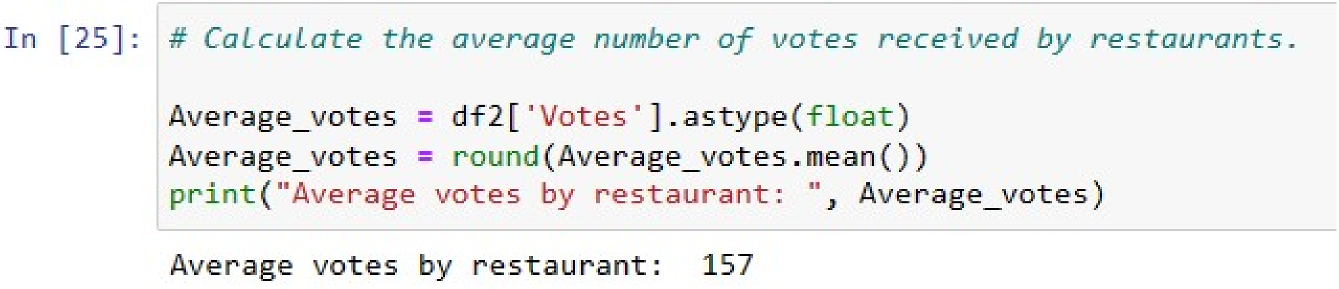
```
In [68]: # Analyze the distribution of aggregate ratings and determine the most common rating range.

         avg_rating_dest = df2['Aggregate rating']

         ax = avg_rating_dest.plot.hist(bins = 5, density = True, color='blue')
         plt.title('Distribution of aggregate ratings')
         plt.xlabel('Aggregate Ratings')
         plt.ylabel('Ratings')
         plt.xticks(rotation=0)
         plt.tight_layout()

         df2['Aggregate rating'].plot.density(color='k', alpha=0.5)

         plt.show(ax)
```



Conclusion: -

1. The distribution in above graph is of BIMODAL Distribution. As there are two peaks, this shows that ratings from 0 to 1 are making peak 1 and ratings between 3 to 4 are making peak 2.
2. Also, Ratings from 3 to 4 are the highest.

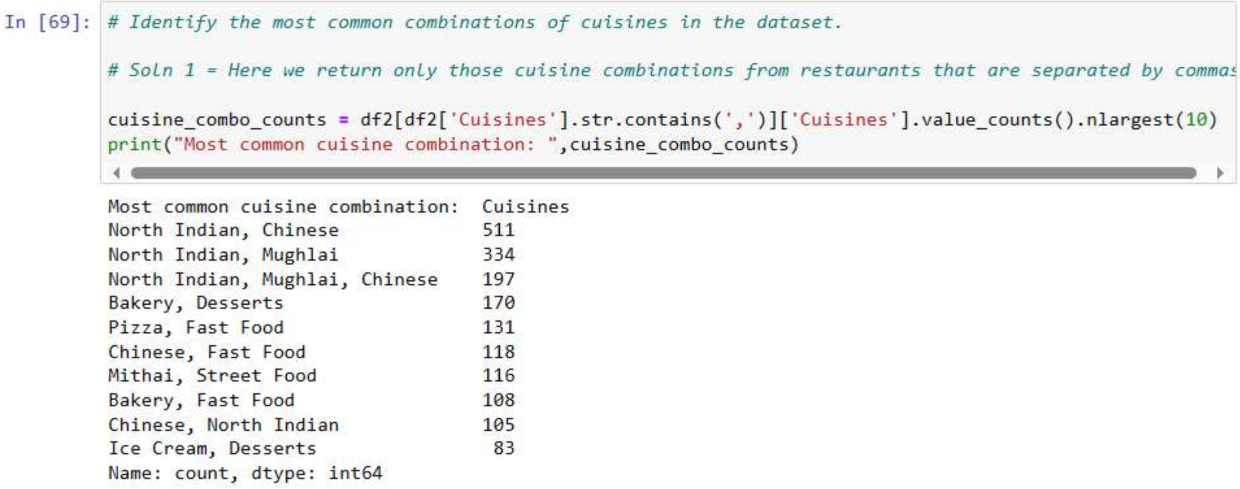
- Calculate the average number of votes received by restaurants



```
In [25]: # Calculate the average number of votes received by restaurants.

         Average_votes = df2['Votes'].astype(float)
         Average_votes = round(Average_votes.mean())
         print("Average votes by restaurant: ", Average_votes)

         Average votes by restaurant:  157
```

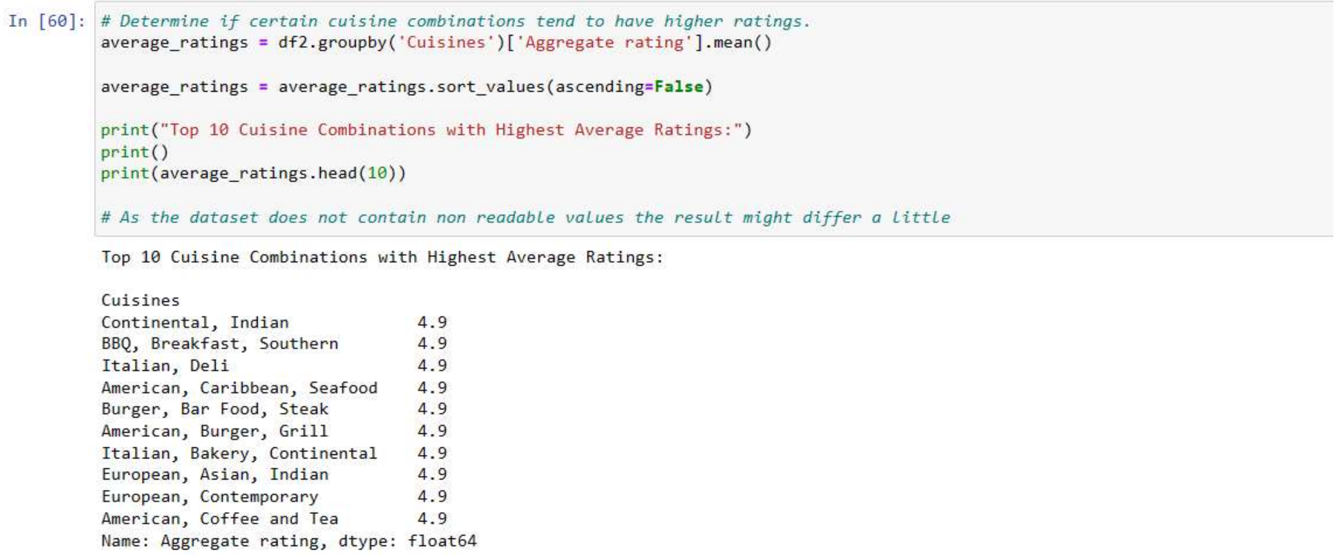
Level 2 Task 2: Cuisine Combination

- Identify the most common combinations of cuisines in the dataset.



```
In [69]: # Identify the most common combinations of cuisines in the dataset.

         # Soln 1 = Here we return only those cuisine combinations from restaurants that are separated by commas

         cuisine_combo_counts = df2[df2['Cuisines'].str.contains(',')]['Cuisines'].value_counts().nlargest(10)
         print("Most common cuisine combination: ",cuisine_combo_counts)

         Most common cuisine combination:  Cuisines
         North Indian, Chinese              511
         North Indian, Mughlai              334
         North Indian, Mughlai, Chinese     197
         Bakery, Desserts                   170
         Pizza, Fast Food                   131
         Chinese, Fast Food                 118
         Mithai, Street Food                116
         Bakery, Fast Food                  108
         Chinese, North Indian              105
         Ice Cream, Desserts                 83
         Name: count, dtype: int64
```

- Determine if certain cuisine combinations tend to have higher ratings.

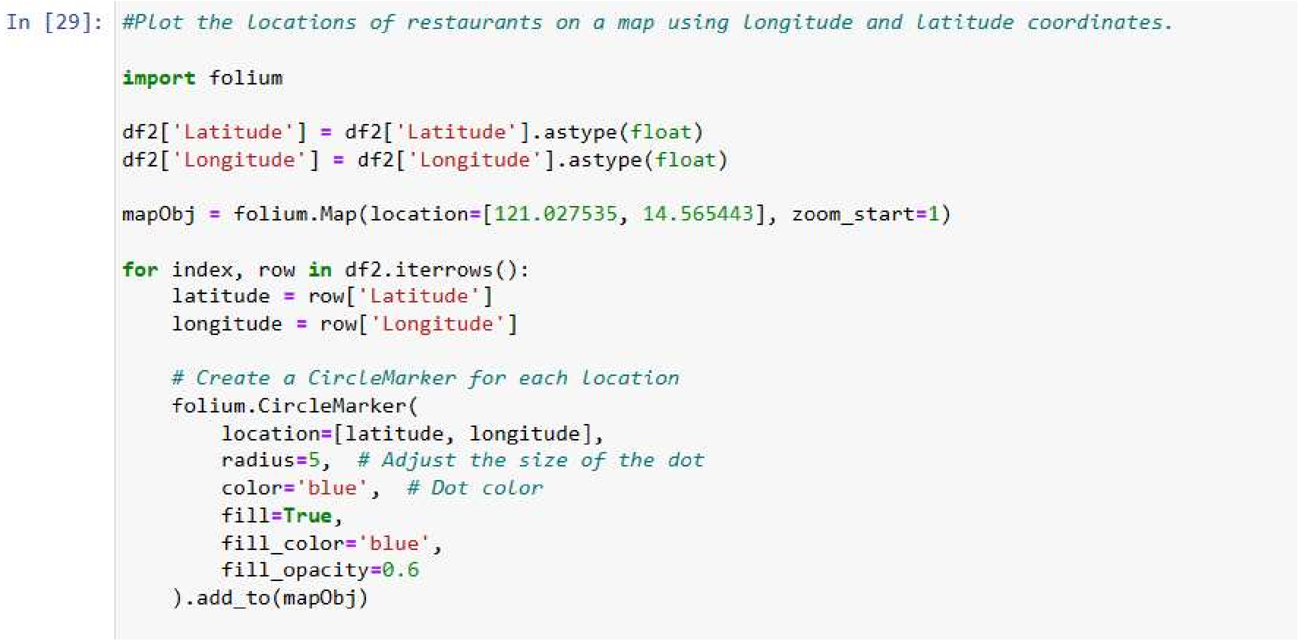


```
In [60]: # Determine if certain cuisine combinations tend to have higher ratings.
         average_ratings = df2.groupby('Cuisines')['Aggregate rating'].mean()

         average_ratings = average_ratings.sort_values(ascending=False)

         print("Top 10 Cuisine Combinations with Highest Average Ratings:")
         print()
         print(average_ratings.head(10))

         # As the dataset does not contain non readable values the result might differ a little
```

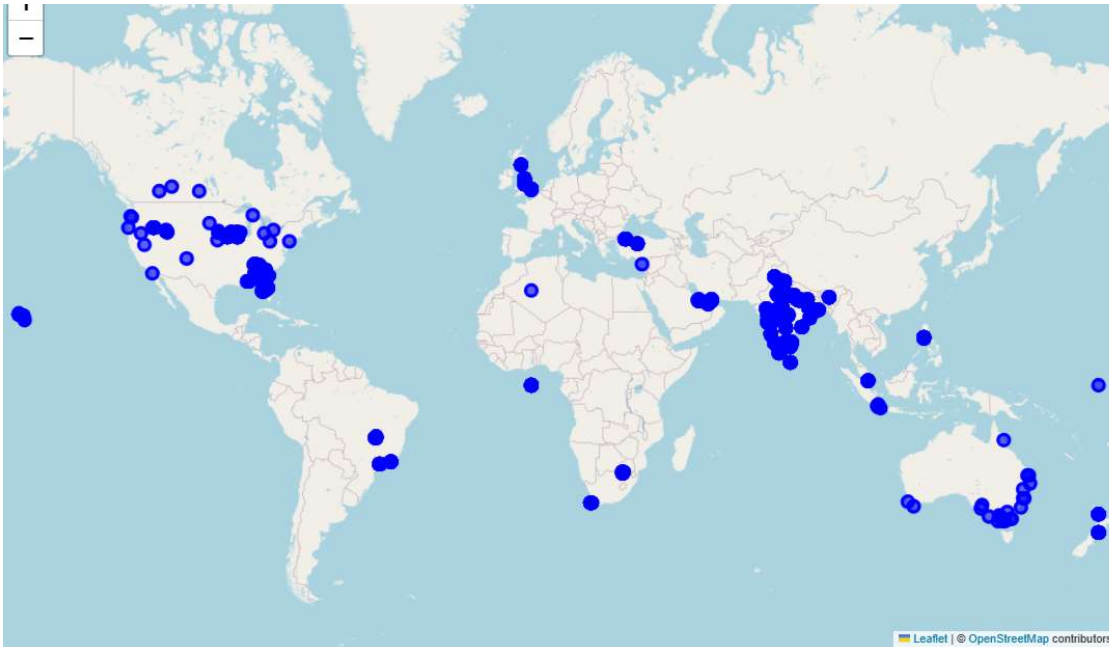
```
Top 10 Cuisine Combinations with Highest Average Ratings:

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
```

Level 2 Task 3: Geographic Analysis

- Plot the locations of restaurants on a map using longitude and latitude coordinates.



```
In [29]: #Plot the locations of restaurants on a map using longitude and latitude coordinates.

         import folium

         df2['Latitude'] = df2['Latitude'].astype(float)
         df2['Longitude'] = df2['Longitude'].astype(float)

         mapObj = folium.Map(location=[121.027535, 14.565443], zoom_start=1)

         for index, row in df2.iterrows():
             latitude = row['Latitude']
             longitude = row['Longitude']

             # Create a CircleMarker for each location
             folium.CircleMarker(
                 location=[latitude, longitude],
                 radius=5,  # Adjust the size of the dot
                 color='blue',  # Dot color
                 fill=True,
                 fill_color='blue',
                 fill_opacity=0.6
             ).add_to(mapObj)
```

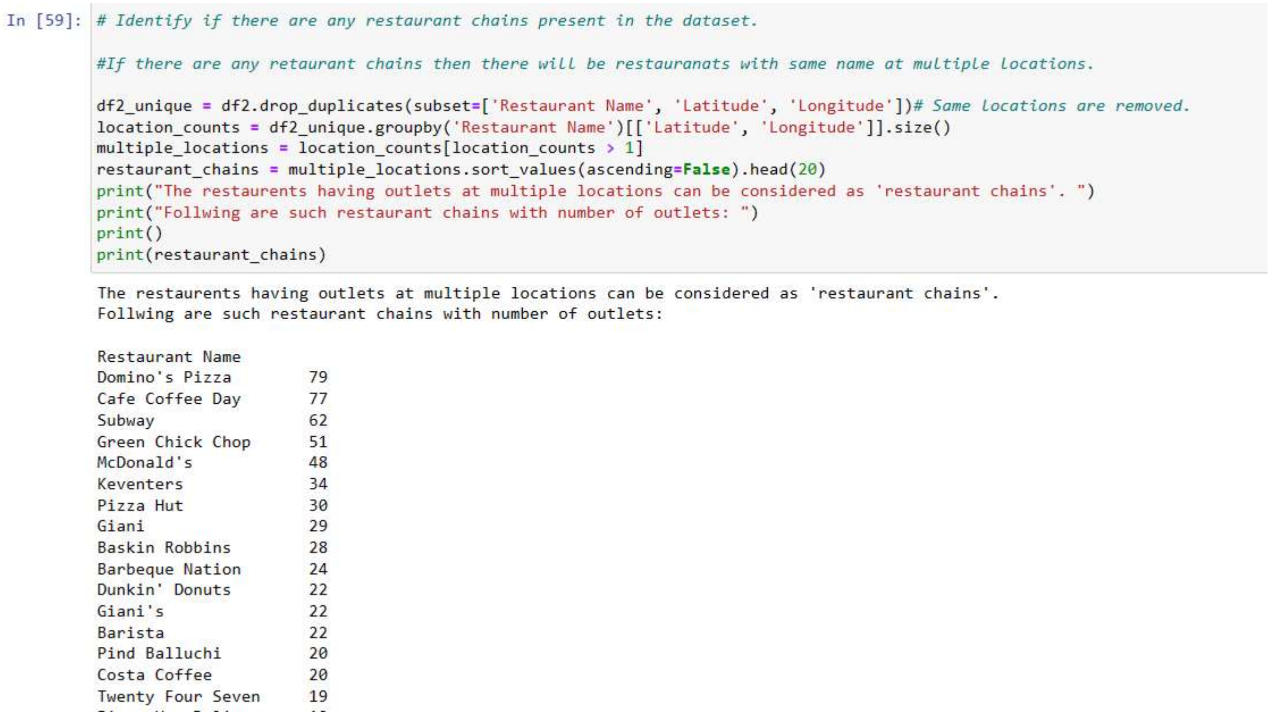


- Identify any patterns or clusters of restaurants in specific areas.

1. By observing the map above we can see clusters of locations of restaurants across the globe
2. It is very clear that the most of the clusters of restaurants are in big cities
3. Cities like Delhim, Mumbai, New York, etc. have even higher number of restaurents.

Level 2 Task 4: Restaurant Chains

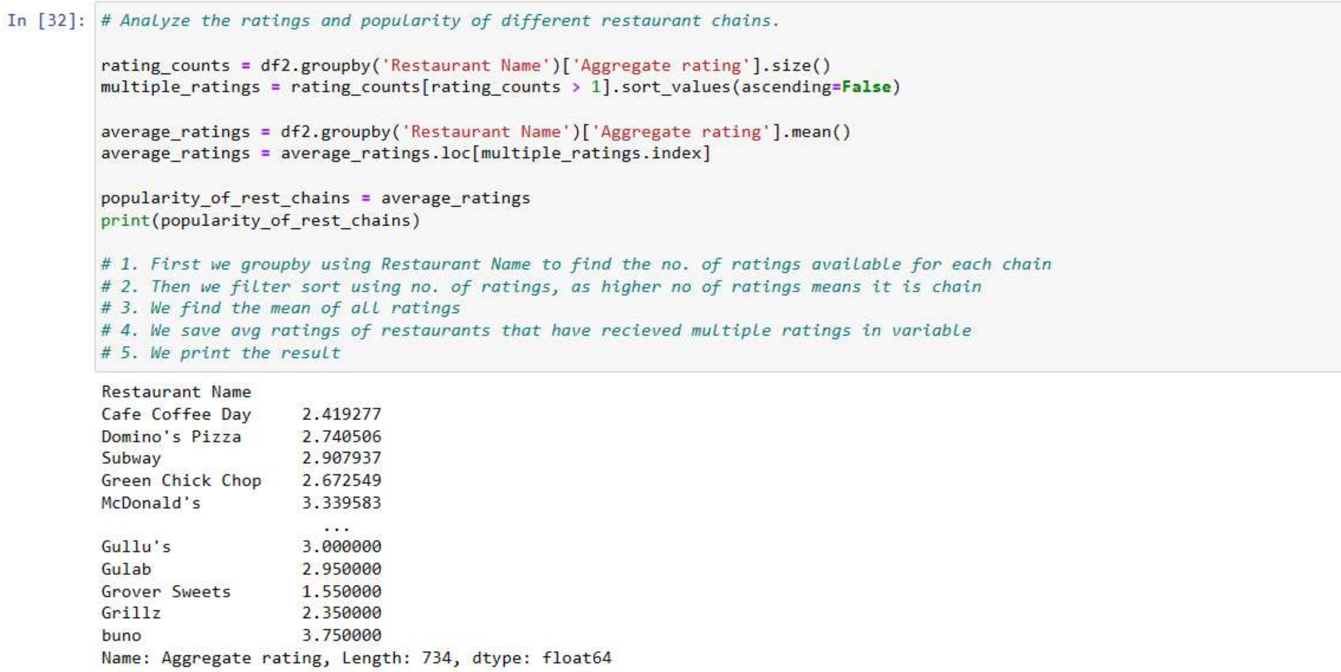
- Identify if there are any restaurant chains present in the dataset.



```
In [59]: # Identify if there are any restaurant chains present in the dataset.

         #If there are any retaurant chains then there will be restauranats with same name at multiple locations.

         df2_unique = df2.drop_duplicates(subset=['Restaurant Name', 'Latitude', 'Longitude'])# Same locations are removed.
         location_counts = df2_unique.groupby('Restaurant Name')[['Latitude', 'Longitude']].size()
         multiple_locations = location_counts[location_counts > 1]
         restaurant_chains = multiple_locations.sort_values(ascending=False).head(20)
         print("The restaurents having outlets at multiple locations can be considered as 'restaurant chains'. ")
         print("Follwing are such restaurant chains with number of outlets: ")
         print()
         print(restaurant_chains)
```

```
The restaurents having outlets at multiple locations can be considered as 'restaurant chains'.
Follwing are such restaurant chains with number of outlets:

Restaurant Name
Domino's Pizza        79
Cafe Coffee Day       77
Subway                62
Green Chick Chop      51
McDonald's            48
Keventers             34
Pizza Hut             30
Giani                 29
Baskin Robbins        28
Barbeque Nation       24
Dunkin' Donuts        22
Giani's               22
Barista               22
Pind Balluchi         20
Costa Coffee          20
Twenty Four Seven     19
```

- Analyze the ratings and popularity of different restaurant chains.

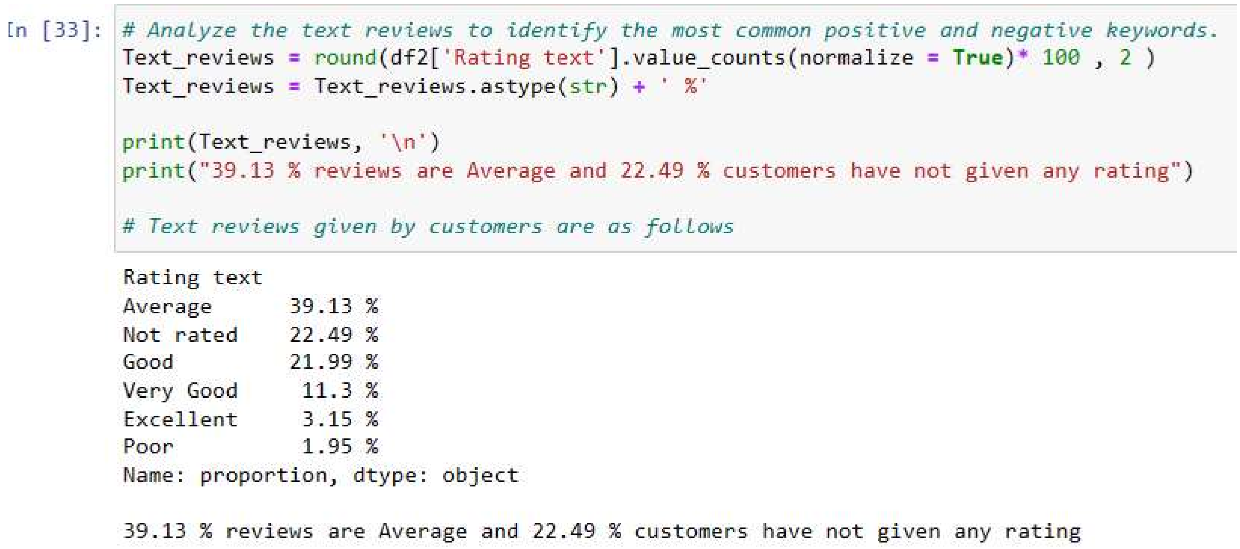


```
In [32]: # Analyze the ratings and popularity of different restaurant chains.

         rating_counts = df2.groupby('Restaurant Name')['Aggregate rating'].size()
         multiple_ratings = rating_counts[rating_counts > 1].sort_values(ascending=False)

         average_ratings = df2.groupby('Restaurant Name')['Aggregate rating'].mean()
         average_ratings = average_ratings.loc[multiple_ratings.index]

         popularity_of_rest_chains = average_ratings
         print(popularity_of_rest_chains)

         # 1. First we groupby using Restaurant Name to find the no. of ratings available for each chain
         # 2. Then we filter sort using no. of ratings, as higher no of ratings means it is chain
         # 3. We find the mean of all ratings
         # 4. We save avg ratings of restaurants that have recieved multiple ratings in variable
         # 5. We print the result
```

```
Restaurant Name
Cafe Coffee Day     2.419277
Domino's Pizza      2.740506
Subway              2.907937
Green Chick Chop    2.672549
McDonald's          3.339583
                      ...
Gullu's             3.000000
Gulab               2.950000
Grover Sweets       1.550000
Grillz              2.350000
buno                3.750000
Name: Aggregate rating, Length: 734, dtype: float64
```

Level 3 Task 1: Restaurant Reviews

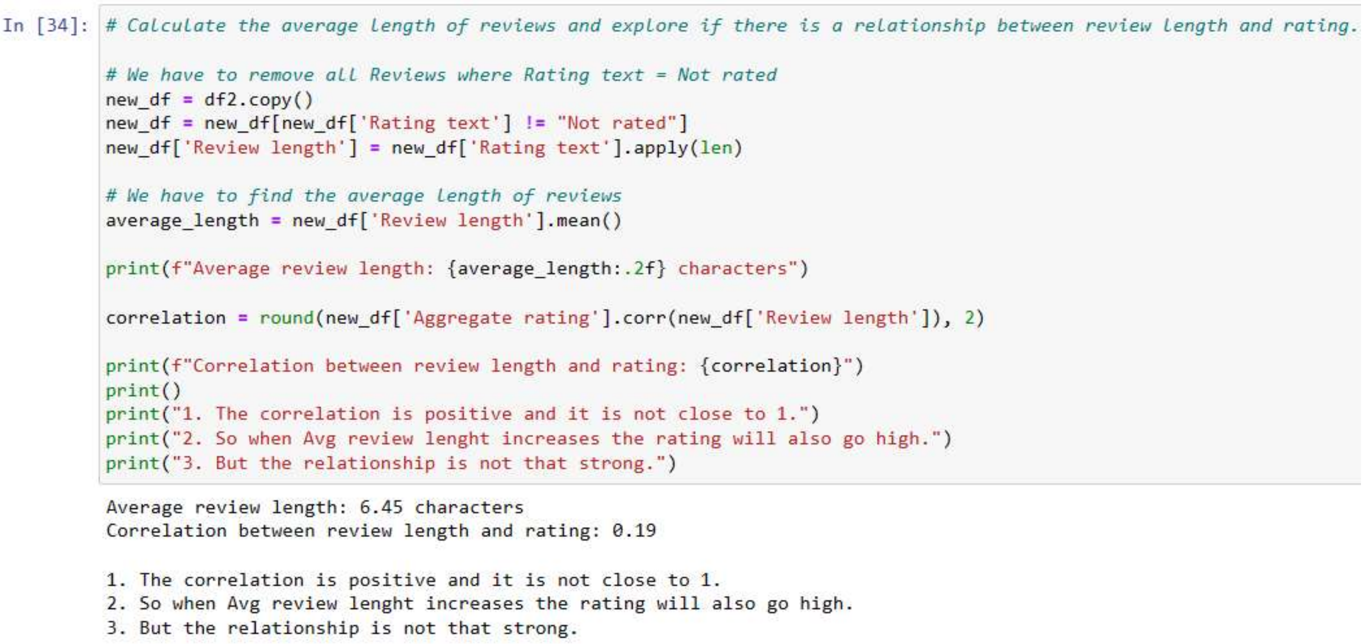
- Analyze the text reviews to identify the most common positive and negative keywords.



```
In [33]: # Analyze the text reviews to identify the most common positive and negative keywords.
         Text_reviews = round(df2['Rating text'].value_counts(normalize = True)* 100 , 2 )
         Text_reviews = Text_reviews.astype(str) + ' %'

         print(Text_reviews, '\n')
         print("39.13 % reviews are Average and 22.49 % customers have not given any rating")

         # Text reviews given by customers are as follows
```

```
Rating text
Average      39.13 %
Not rated    22.49 %
Good         21.99 %
Very Good     11.3 %
Excellent     3.15 %
Poor          1.95 %
Name: proportion, dtype: object

39.13 % reviews are Average and 22.49 % customers have not given any rating
```

- Calculate the average length of reviews and explore if there is a relationship between review length and rating.

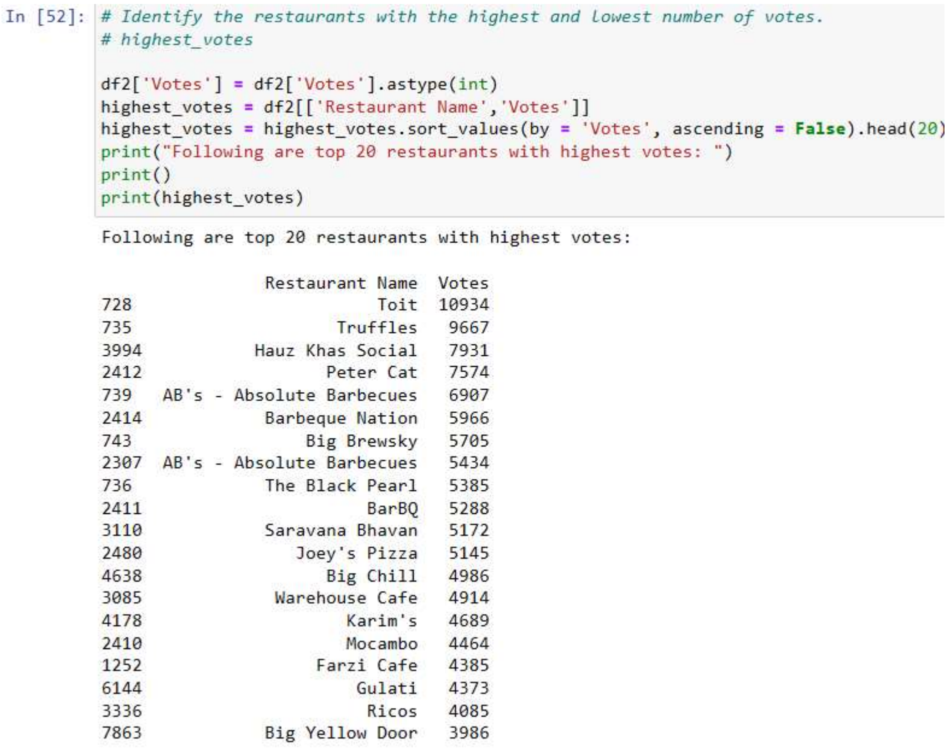


```
In [34]: # Calculate the average length of reviews and explore if there is a relationship between review length and rating.

         # We have to remove all Reviews where Rating text = Not rated
         new_df = df2.copy()
         new_df = new_df[new_df['Rating text'] != "Not rated"]
         new_df['Review length'] = new_df['Rating text'].apply(len)

         # We have to find the average length of reviews
         average_length = new_df['Review length'].mean()

         print(f"Average review length: {average_length:.2f} characters")

         correlation = round(new_df['Aggregate rating'].corr(new_df['Review length']), 2)

         print(f"Correlation between review length and rating: {correlation}")
         print()
         print("1. The correlation is positive and it is not close to 1.")
         print("2. So when Avg review lenght increases the rating will also go high.")
         print("3. But the relationship is not that strong.")
```

```
Average review length: 6.45 characters
Correlation between review length and rating: 0.19

1. The correlation is positive and it is not close to 1.
2. So when Avg review lenght increases the rating will also go high.
3. But the relationship is not that strong.
```

Level 3 Task 2: Votes Analysis

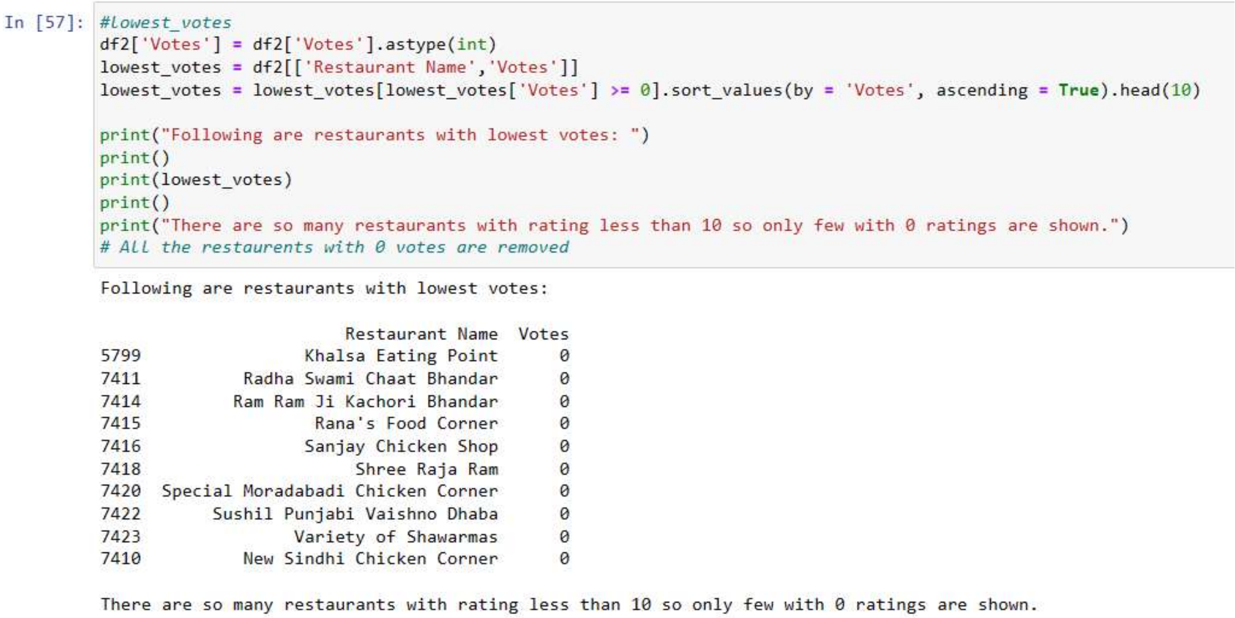
- Identify the restaurants with the highest and lowest number of votes.
    1. highest number of votes



```
In [52]: # Identify the restaurants with the highest and lowest number of votes.
         # highest_votes

         df2['Votes'] = df2['Votes'].astype(int)
         highest_votes = df2[['Restaurant Name','Votes']]
         highest_votes = highest_votes.sort_values(by = 'Votes', ascending = False).head(20)
         print("Following are top 20 restaurants with highest votes: ")
         print()
         print(highest_votes)
```

```
Following are top 20 restaurants with highest votes: 

                Restaurant Name  Votes
728                        Toit  10934
735                    Truffles   9667
3994           Hauz Khas Social   7931
2412                  Peter Cat   7574
739   AB's - Absolute Barbecues   6907
2414            Barbeque Nation   5966
743                 Big Brewsky   5705
2307  AB's - Absolute Barbecues   5434
736             The Black Pearl   5385
2411                      BarBQ   5288
3110            Saravana Bhavan   5172
2480                Joey's Pizza   5145
4638                  Big Chill   4986
3085             Warehouse Cafe   4914
4178                    Karim's   4689
2410                    Mocambo   4464
1252                 Farzi Cafe   4385
6144                     Gulati   4373
3336                      Ricos   4085
7863            Big Yellow Door   3986
```

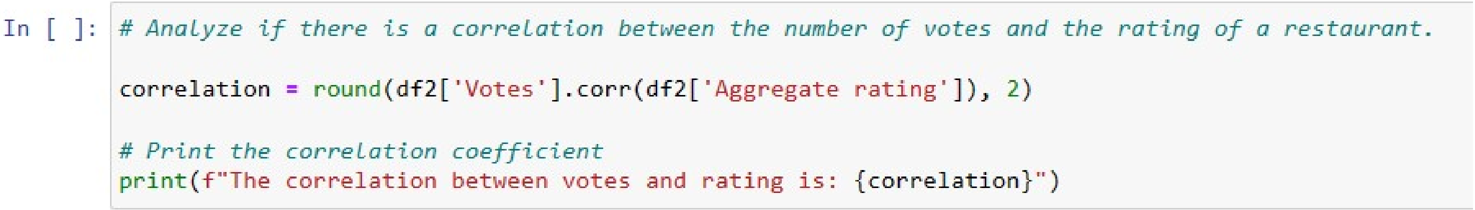
    2. lowest number of votes.



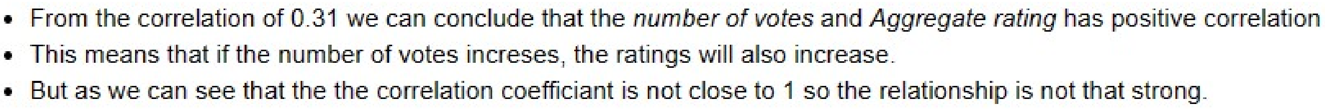
```
In [57]: #Lowest_votes
         df2['Votes'] = df2['Votes'].astype(int)
         lowest_votes = df2[['Restaurant Name','Votes']]
         lowest_votes = lowest_votes[lowest_votes['Votes'] >= 0].sort_values(by = 'Votes', ascending = True).head(10)

         print("Following are restaurants with lowest votes: ")
         print()
         print(lowest_votes)
         print()
         print("There are so many restaurants with rating less than 10 so only few with 0 ratings are shown.")
         # All the restaurents with 0 votes are removed
```

```
Following are restaurants with lowest votes: 

                      Restaurant Name  Votes
5799               Khalsa Eating Point      0
7411          Radha Swami Chaat Bhandar      0
7414          Ram Ram Ji Kachori Bhandar      0
7415                 Rana's Food Corner      0
7416                Sanjay Chicken Shop      0
7418                   Shree Raja Ram      0
7420  Special Moradabadi Chicken Corner      0
7422        Sushil Punjabi Vaishno Dhaba      0
7423              Variety of Shawarmas      0
7410             New Sindhi Chicken Corner      0

There are so many restaurants with rating less than 10 so only few with 0 ratings are shown.
```

- Analyze if there is a correlation between the number of votes and the rating of a restaurant.



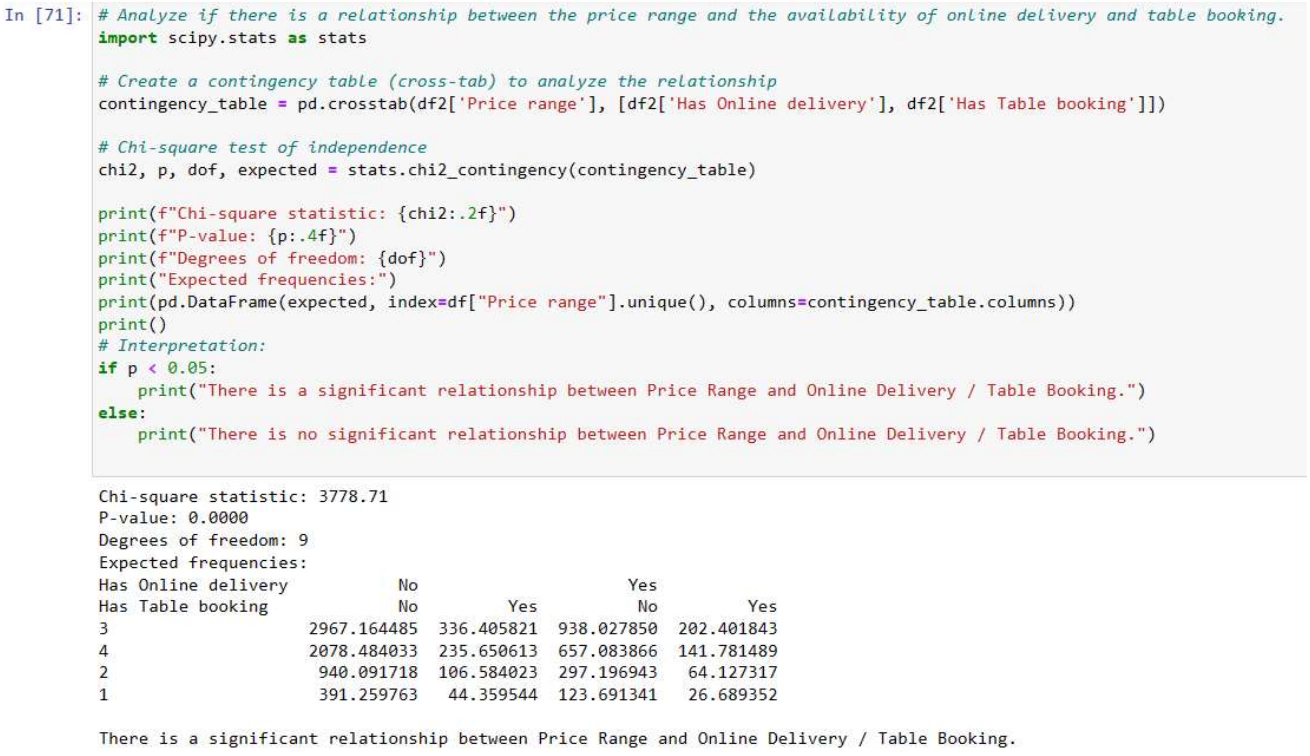
```
In [ ]: # Analyze if there is a correlation between the number of votes and the rating of a restaurant.

        correlation = round(df2['Votes'].corr(df2['Aggregate rating']), 2)

        # Print the correlation coefficient
        print(f"The correlation between votes and rating is: {correlation}")
```



- From the correlation of 0.31 we can conclude that the *number of votes* and *Aggregate rating* has positive correlation
- This means that if the number of votes increses, the ratings will also increase.
- But as we can see that the the correlation coefficiant is not close to 1 so the relationship is not that strong.

Level 3 Task 3: Price Range vs. Online Delivery and Table Booking

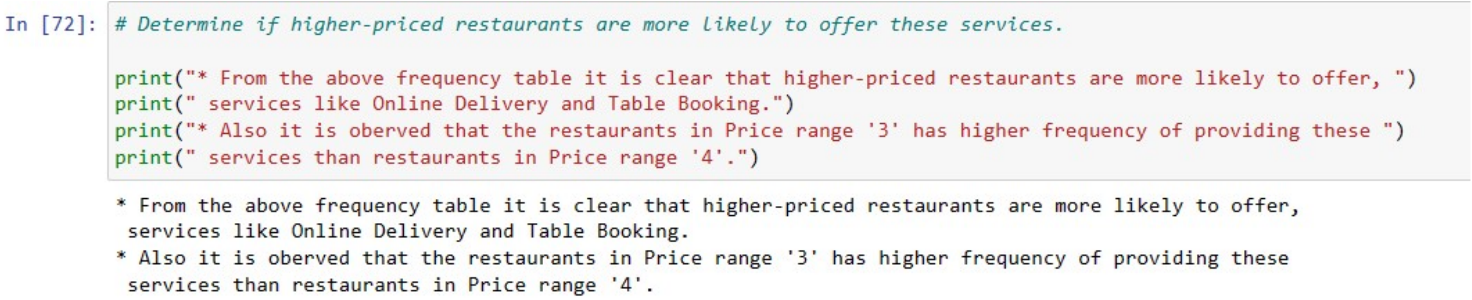
- Analyze if there is a relationship between the price range and the availability of online delivery and table booking.



```
In [71]: # Analyze if there is a relationship between the price range and the availability of online delivery and table booking.
         import scipy.stats as stats

         # Create a contingency table (cross-tab) to analyze the relationship
         contingency_table = pd.crosstab(df2['Price range'], [df2['Has Online delivery'], df2['Has Table booking']])

         # Chi-square test of independence
         chi2, p, dof, expected = stats.chi2_contingency(contingency_table)

         print(f"Chi-square statistic: {chi2:.2f}")
         print(f"P-value: {p:.4f}")
         print(f"Degrees of freedom: {dof}")
         print("Expected frequencies:")
         print(pd.DataFrame(expected, index=df["Price range"].unique(), columns=contingency_table.columns))
         print()
         # Interpretation:
         if p < 0.05:
             print("There is a significant relationship between Price Range and Online Delivery / Table Booking.")
         else:
             print("There is no significant relationship between Price Range and Online Delivery / Table Booking.")
```

```
Chi-square statistic: 3778.71
P-value: 0.0000
Degrees of freedom: 9
Expected frequencies:
Has Online delivery           No                      Yes
Has Table booking             No         Yes          No         Yes
3                    2967.164485  336.405821  938.027850  202.401843
4                    2078.484033  235.650613  657.083866  141.781489
2                     940.091718  106.584023  297.196943   64.127317
1                     391.259763   44.359544  123.691341   26.689352

There is a significant relationship between Price Range and Online Delivery / Table Booking.
```

- Determine if higher-priced restaurants are more likely to offer these services.



In [72]:

```
# Determine if higher-priced restaurants are more likely to offer these services.

print("* From the above frequency table it is clear that higher-priced restaurants are more likely to offer, ")
print(" services like Online Delivery and Table Booking.")
print("* Also it is oberved that the restaurants in Price range '3' has higher frequency of providing these ")
print(" services than restaurants in Price range '4'.")
```

```
* From the above frequency table it is clear that higher-priced restaurants are more likely to offer, 
 services like Online Delivery and Table Booking.
* Also it is oberved that the restaurants in Price range '3' has higher frequency of providing these 
 services than restaurants in Price range '4'.
```