Zomato API – II

Name : Shobhit Bansal

Email id : bshobhit67@gmail.com

PART 1 of 3 : Analysing the given dataset on the basis of NCR region and the Rest of India :

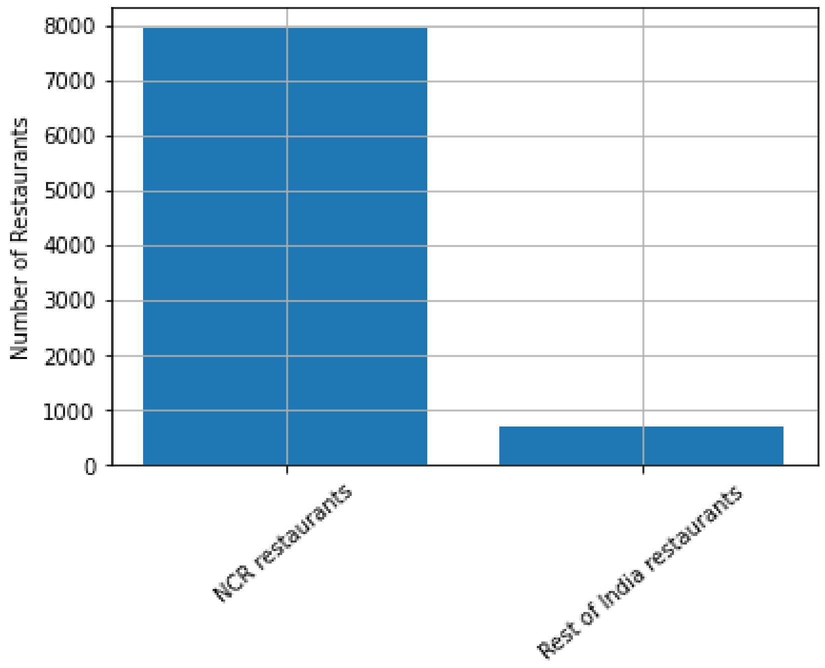
1. *To compare the number of restaurants in the NCR region against the rest of India.*

   Here, after loading the Dataset in a pandas data frame, the first step is to replace all the cities in the NCR region to the name "NCR". This includes New Delhi, Noida, Gurgaon, Ghaziabad and Faridabad. Then using value_counts(), we found that the data for NCR region was far more than other parts of the country. So for the rest of India we subtracted the NCR number from the total restaurant number of the country. And then we plotted a bar graph that made the visualisation.

   Conclusion : The Dataset is highly skewed towards NCR region as evident from the number of restaurants as well as the bar plot.

   Output and Graph :

```
NCR restaurants :  7947
Rest of India restaurants :  705
```



2. *Find the Cuisines that are served in restaurants other than the NCR region. Also verify if this is due to the incomplete dataset or not.*

First of all, we remove all the spaces in the Cuisine as they are comma separated and would become hectic as some would have an added space in the beginning or at the end. Doing this does not leave the data pretty, but improves accuracy of the result. The data frame is split into two data frames each for NCR region and the rest of the country. Let's define two similar functions that will split the cuisines by the "," delimiter and maintaining two Boolean hashmaps which will store if the corresponding cuisine is served in a restaurant. Now, we have two Boolean hashmaps that contain every cuisine served in NCR and Rest of India(ROI) respectively. We traverse over the ROI hashmap and check if the NCR hashmap returns true or False. If False is returned, we add that cuisine to a list naming it as Not_in_NCR. This list contains the data about all the cuisines served in ROI but not in NCR (obviously according to the dataset given).

To check if this result is due to the incomplete dataset, we can make an API call having parameters as city_id (that is 1 in case of NCR) and query and try to generate the cuisine id for the elements of the Not_in_NCR list. If a cuisine id is generated and returned then it implies that such a cuisine is served in some restaurant in the NCR region.

Conclusion : We found that there were 4 cuisines served in other parts of the country but not in NCR as per the given the data. One of the 4, BBQ however, was found to be served in some restaurants of the NCR region as the API call returned a cuisine id. To verify this result even further, we even found the list of 10 restaurants serving BBQ that are closest to Coding Ninjas Campus. In conclusion, **the given dataset is Incomplete.**

Output :

Part 1 : (Cuisines not in NCR as per dataset)

```
German
Malwani
BBQ
Cajun
```

Part 2 : (API call to find cuisine id)

```
BBQ 193
```

Part 3 : (10 restaurants closest to Coding Ninjas Campus serving BBQ)

```
Peshawari Kohat Wale
Tandoori Nights
UBQ by Barbeque Nation
Barbeque Nation
Jungal Raaj
Anytime Chicken
UBQ by Barbeque Nation
Moet's Barbeque
The Tangdi Club
BBQ Express By The Barbeque Company
```

3. *Find the top 10 cuisines served the most in Delhi NCR and Rest of India.*

The initial code logic is not very different from the problem just above. Maintaining two hashmaps, defining two similar functions and applying on the cuisines column header of the data frame. The hashmaps in this case do not need to be Boolean but, they will maintain the number of restaurants serving the corresponding cuisines. To make a comparison, I felt that using percentages is most effective. So we converted number of restaurants serving a cuisine to the percent of restaurants serving the said cuisine. Finally, the top 10 were printed.

Conclusion : The top 10 cuisines served the most were printed separately for NCR region and the Rest of India. Note that I have removed all the spaces in the cuisine names, so readability is a bit reduced however the result is accurate.

Output :

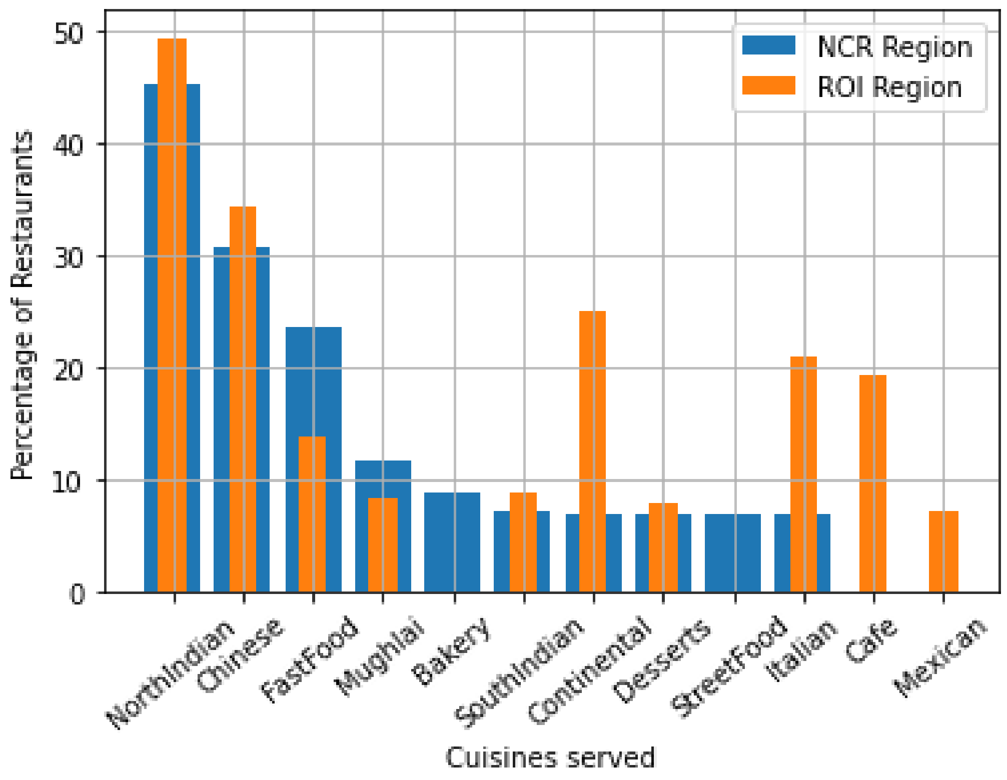
```
The Top 10 cuisines served in the restaurants of NCR :

NorthIndian 45.26
Chinese 30.8
FastFood 23.48
Mughlai 11.74
Bakery 8.77
SouthIndian 7.16
Continental 6.88
Desserts 6.82
StreetFood 6.77
Italian 6.73

The Top 10 cuisines served in the restaurants of the rest of India :

NorthIndian 49.5
Chinese 34.33
Continental 25.11
Italian 20.85
Cafe 19.29
FastFood 13.76
SouthIndian 8.79
Mughlai 8.37
Desserts 7.8
Mexican 7.09
```

4. *Short detailed note and inference of the result found above.*

Finally, I plotted a bar plots showing the percentage of restaurants serving a cuisine for NCR region and other Indian restaurants separately. It is very intuitive to use Pie Chart, however the Pie Chart will not be an accurate representation as there are many restaurants that serve more than one cuisine. So a Bar plot has made the interpretation easy.

Conclusion and Inference : The first observation made is that the North Indian and Chinese are the most popular all over the country. They are even more popular in parts of the country other than the NCR region. Looking at the rest 8, these trends behave quite differently in the NCR region than other parts of the country. However 6 of the cuisines are same, though their percentages are quite different. Delhi NCR has more percentages of restaurants serving Bakery and Street Food but Café and Mexican Cuisines are not that popular. Delhi NCR has preferred Fast Food the most. The difference of percentages of restaurants is quite noticeable. Delhi NCR has more Fast Food than other parts of the country.

Output and Graph Plot :

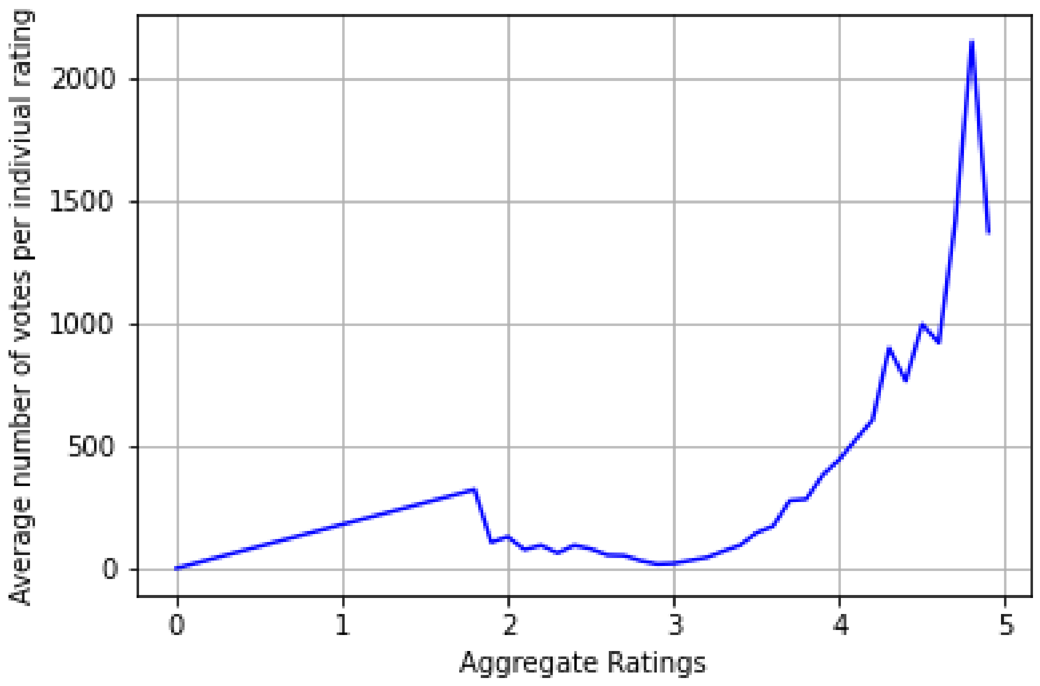


PART 2.1 of 3 : Analysing the trends of User ratings with different parameters.

1. *User Rating against number of votes*

First of all, I must convey what the graph will imply. The Graph implies the average number of votes given per individual rating value. Let's say there are two restaurants with rating 4.1. One restaurant got 10, the other got 20 votes. So to get a 4.1 rating, a restaurant should have on an average (10 + 20)/ 2 = 15 users reviewing a restaurant. The logic behind is quite simple. Apply value_counts() to the "Aggregate rating" column of the data frame. Add the total votes for each rating separately, divide it by the frequency of each rating and store it in a hashmap. Finally, sorting it based on the rating value and then plotting the simple line graph.

Line Graph :

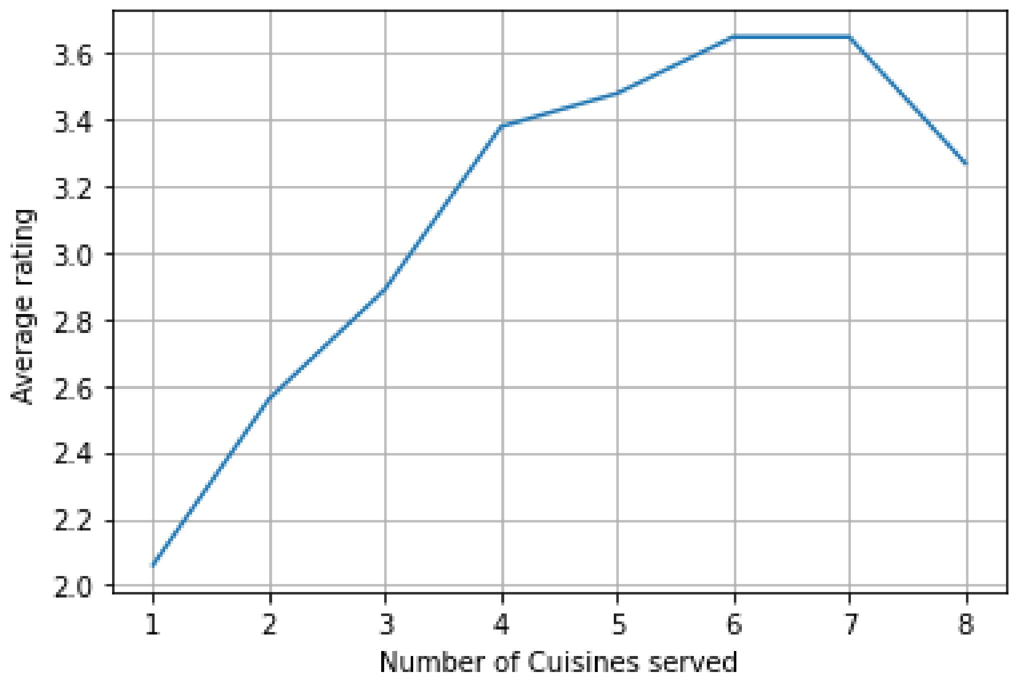


Inference and Conclusion : The first and foremost observation is the linear part of the curve between 0 and a value just less than 2. This does not have any significance apart from the fact that there is no data on restaurants who got ratings in this range in the given dataset. The Zero rating and Zero votes corresponds to the unrated restaurants. Coming to the more interesting and informative part of the curve. The curve takes a somewhat smooth dip towards rating of 3 but the number of votes increase almost exponentially moving towards 4 and 5. Ignoring the final dip (Being linear, only reflects about one point therefore insignificant) and keeping in mind the limitations of the given dataset, **the Average number of votes affect a restaurant rating directly.** A restaurant aiming to get a higher user rating should encourage Users to Review and vote for their restaurant.

2. *User Rating v/s Number of Cuisines served*

Firstly, I need to convey what the graph would depict. The graph depicts the trend of average user rating to the number of cuisines served. Let's say there are 3 restaurants serving 2 cuisines with ratings 4.1, 3.6 and 4.6. So on average, a restaurant serving 2 cuisines gets (4.1 + 3.6 + 4.6) / 3 = 4.1. The Logic here again, is similar to the one used to compare the trend of cuisines in NCR restaurants and in Other Indian Restaurants. Firstly, removing spaces so as to avoid and trailing or leading spaces in different names on the Dataset. Then applying another user defined function. Here, the name of the cuisines is split by the ',' and the length is noted. I maintained a Hashmap where number of cuisines served is the key and each value is a list of size 2. It holds the sum of user rating and count of the restaurants serving the cuisine. Then finally, separating key and the division of the two values. I felt that the line graph is the most informative here so I plotted a line graph.

Line Graph :

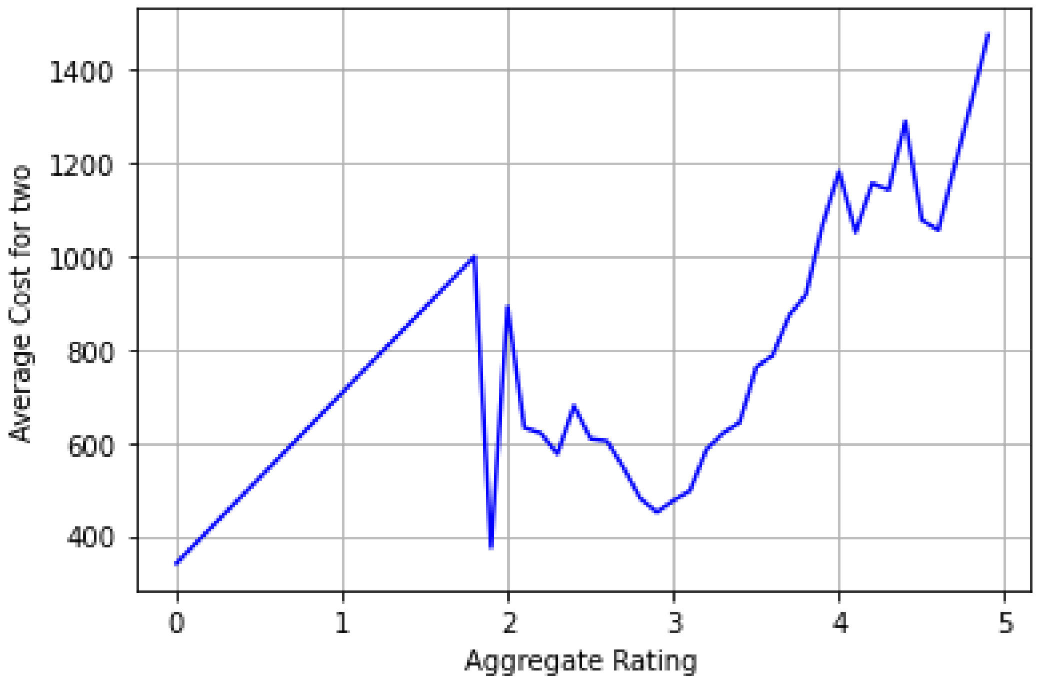


Inference and Conclusion : First of all, there are at max eight cuisines served at a single restaurant as per the given dataset. The inference can be made by breaking the curve into two parts. Number one is that restaurants serving four or less cuisines, the rise is high, that is users are expecting restaurants to serve multiple cuisines. However, moving forward, the rise become less steeper, then plateauing and finally falling, This implies that either restaurants are serving too many cuisines that they are not able to keep up with other important factors or the users expect variety to an extent. So in conclusion **for a restaurant, 4 – 7 is an ideal range of the number of cuisines that it should serve to get better approval ratings.**

3. *User Rating v/s Average Cost*

Again, it is important to convey what the graph depicts. The graph depicts the average cost for two at a restaurant of a specific individual rating. This means the average amount a user or customer pays for sitting in a restaurant of a specific rating. Let's say there are 2 restaurants with rating 3.5 having average cost of two as 1000 and 1500 respectively. So at a restaurant with rating 3.5, the average cost for two is (1000 + 1500) / 2 = 1250. The Code logic again is simple and quite similar to the comparison of user rating to the Number of votes. Apply value_counts() to the "Aggregate rating" column of the data frame. Add the Average cost for two for each rating separately, divide it by the frequency of each rating and store it in a hashmap. Finally, sorting it based on the rating value and then plotting the simple line graph.

Line Graph :

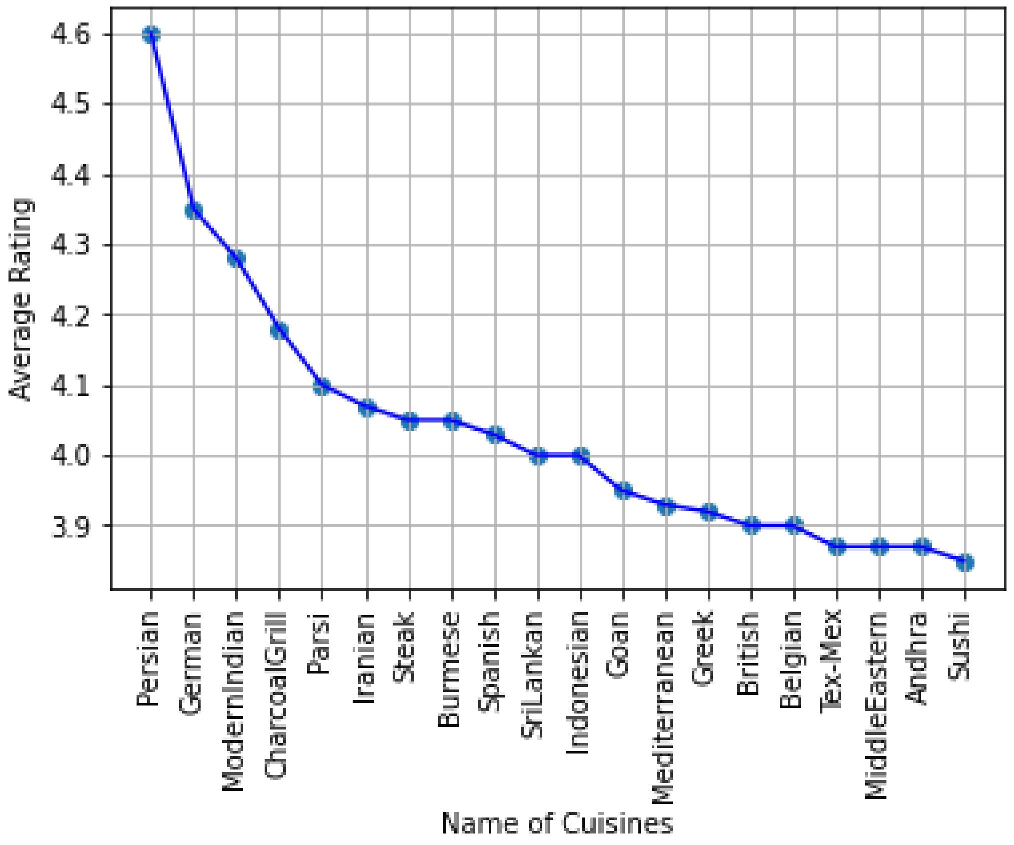


Conclusion and Inference : Here again, the initial linear part is insignificant apart from conveying the fact that there is no data available for the said interval of ratings in the dataset. Despite certain irregularities the general trend is quite straight forward. The rating and the average cost of two are directly dependent when we talk about restaurants having a user of 3 and above. However, for a restaurant of a rating between two and three, there is a dip in the average cost of two, converse to the general trend. **All in all, for a restaurant rated 4 or above, the average cost for two is in the range of 1000 to 1500.**

4. *User Rating v/s Specific Cuisines*

Here the graph would imply the average rating for a restaurant serving a specific cuisine. The logic here will again draw parallels with the trend analysis for user rating and Number of cuisines. As discussed many times before, I removed spaces in the Cuisine. Then applied a user defined function that splits the string on a comma. Here is what's different. Now we traverse on each element of the splitted string and maintain a hashmap that holds the sum of ratings and count of restaurants in a list as value and name of the cuisine as key. Then dividing the two values in the hashmap to get average rating and then sorting based on the ratings. I plotted a line graph as I could fit more data into the chart area. However a bar plot would also suffice.

Line Graph :



Conclusion and Inference : There is not much to analyse here. It is basically a list of cuisines that are the most popular and fetch the highest rating for a restaurant. Persian is the most rated cuisine served as per the dataset, followed by German, Modern Indian and Charcoal Grill.

PART 2.2 of 3 : Finding the Weighted rating for each locality in the dataset.

The Code logic is quite simple here as well. Firstly, I created another column by the name Weight that is the product of rating and number of votes. Then, all different locality names are found using unique(). Traversing over each element in this list of unique localities, I found the weighted rating of each by use of Boolean indexing and the .sum() pandas functionality and maintaining a hashmap that holds the weighted rating of each locality. Then converted the hashmap to a pandas data frame and sorted it in descending order finally printing the top 10.

Conclusion : The top 10 localities of India with the highest weighted ratings were printed. I considered only Indian localities here as the lobal data was not interesting and did not convey any information of value

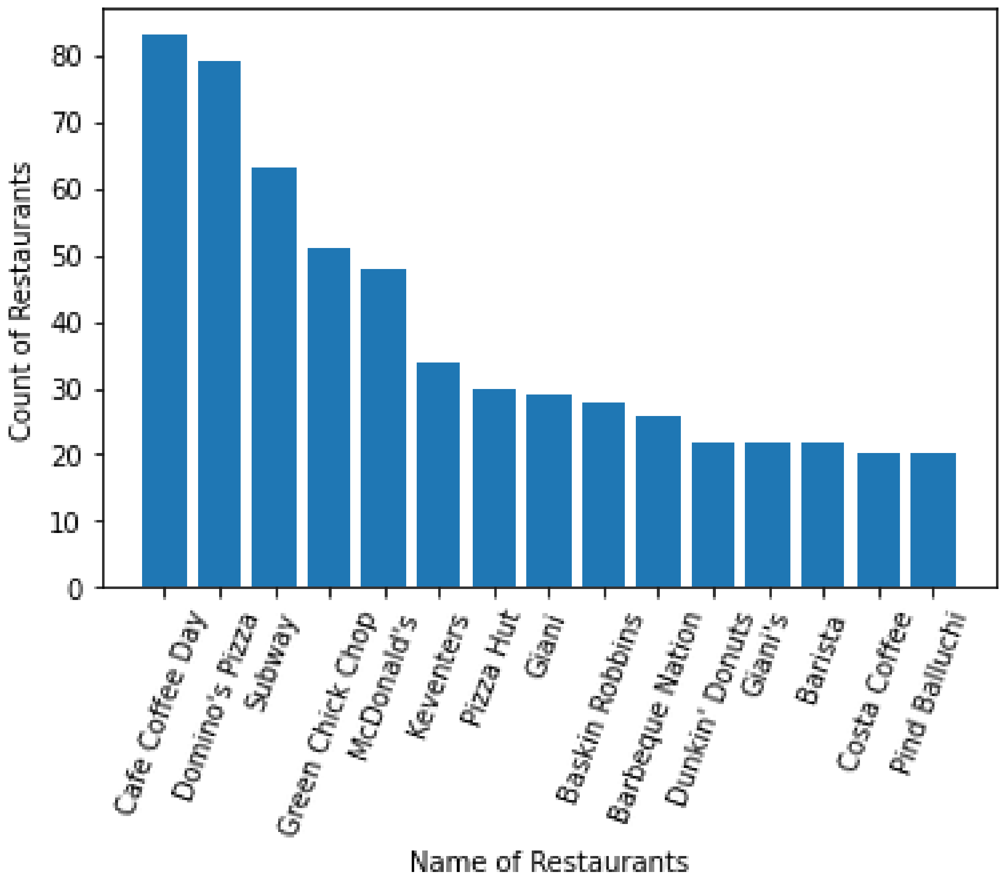
Output :

```
Aminabad 4.9
Hotel Clarks Amer, Malviya Nagar 4.9
Friends Colony 4.89
Powai 4.84
Kirlampudi Layout 4.82
Deccan Gymkhana 4.8
Express Avenue Mall,  Royapettah 4.8
Banjara Hills 4.72
Sector 5, Salt Lake 4.71
Riverside Mall, Gomti Nagar 4.7
```

PART 3 of 3 : Visualisation

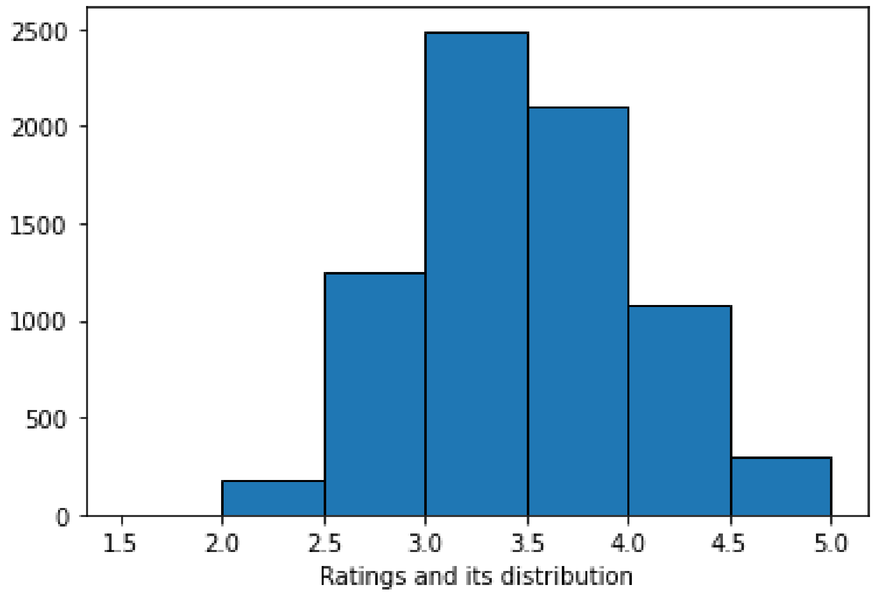
1. *Bar graph of the top 15 restaurants have a maximum number of outlets.*

   These problems are simply to visualize and are very straight forward. I simply applied the Value_counts() to the restaurant name and sliced the returned list for the top 15 and then simply plotted the bar graph.



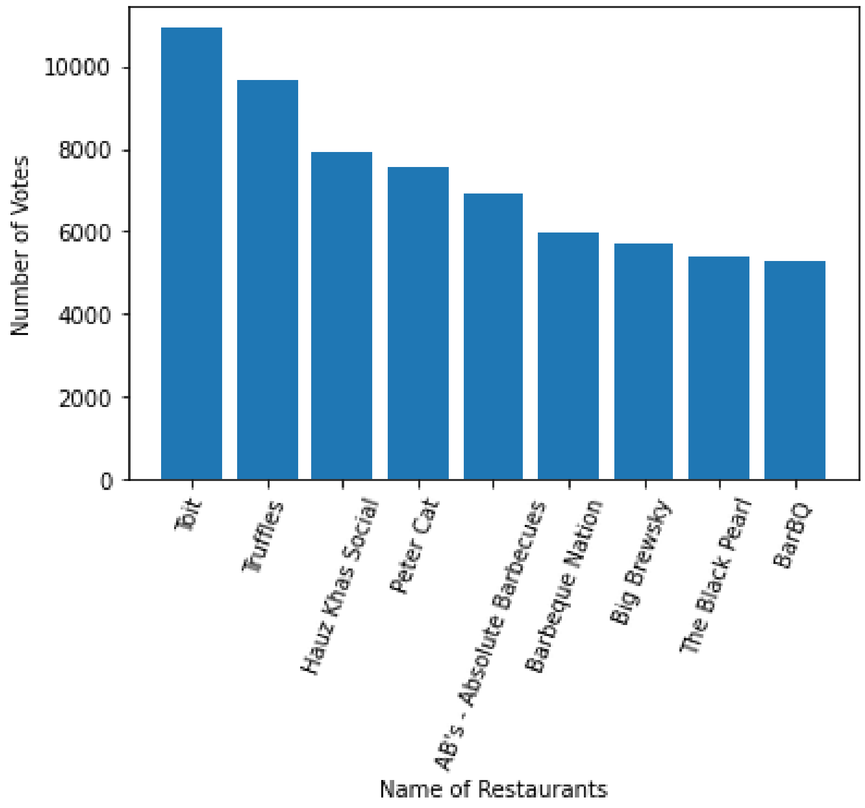
2. *Histogram of aggregate rating of restaurant(drop the unrated restaurant)*

The histogram plotting is also quite simple here. Firstly, I dropped the unrated restaurants by using Boolean Indexing that is only that data was kept in the data frame whose aggregate rating was not zero. I made a bin array from 1.5 to 5 and then plotted the Histogram.



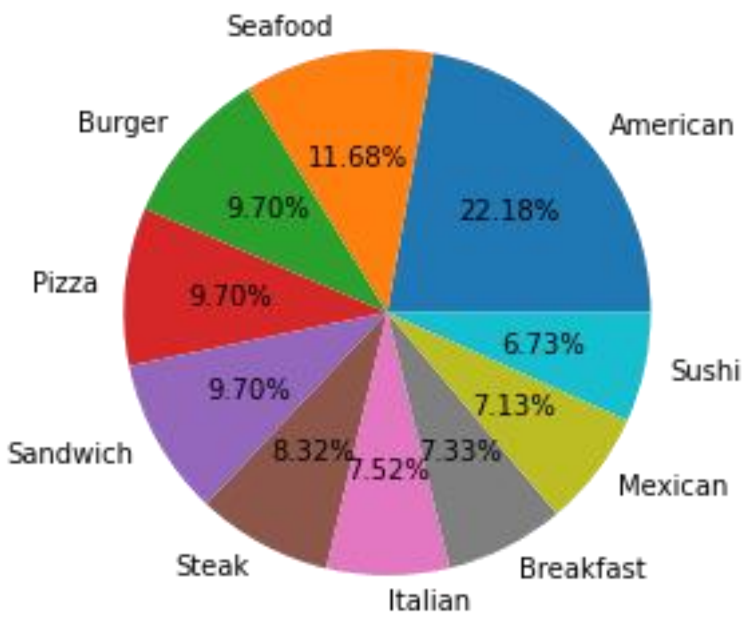
3. *Bar graph of the top 10 restaurants in the data with the highest number of votes.*

The only major step here is to sort the data frame by the number of votes which was taken care by sort_values. Then slicing the array of the Restaurant column and votes column to get the top 10 values and then plotting the bar graph.



4. *Pie graph of top 10 cuisines present in restaurants in the USA*

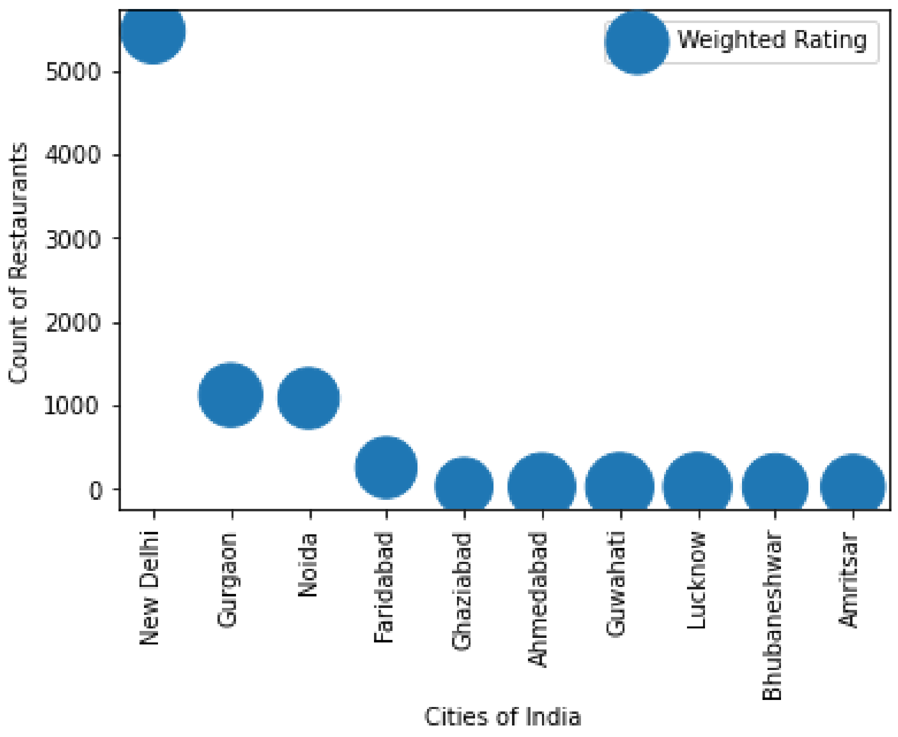
Firstly, I selected the data for USA only by applying Boolean Indexing on the Country Code. Then, removing spaces in the cuisine column and applying a user defined function that maintains a hashmap for the count of each cuisine. Then we sorted the hashmap by converting it into the pandas data frame and applying sort_values(). Then creating a list each for cuisine name and count for the top 10. Using these two lists, the pie chart was plotted.



5. *Bubble graph of a number of Restaurants present in the city of India and keeping the weighted restaurant rating of the city in a bubble.*

   Note : Since the desired number of cities has not been mentioned, I have plotted the graph for the top 10.

   This problem is solved using one of the problems above. I found the weighted rating for city in the exact same way I used in Part 2.2 of 3. These are the sizes of the bubbles. In the same user defined function, I also maintained the count of restaurants in a city in another hashmap. Now I sorted the count of restaurants in a city hashmap by converting it to a pandas data frame and sort_values(). Now I traversed the top 10 of the city restaurant count and maintained three lists, one for the city name, one for the count and one for the sizes of the bubbles that is the weighted rating of the city. Note that I multiplied the rating by 200 because if I considered the original values, the bubble sizes would have been so small that they would not convey much and the reader would not be able to differentiate on the ratings.



Weighted Rating
5000
4000
3000
2000
1000
0
Count of Restaurants
New Delhi
Gurgaon
Noida
Faridabad
Ghaziabad
Ahmedabad
Guwahati
Lucknow
Bhubaneshwar
Amritsar
Cities of India