# Coursera Capstone Project: Applied Data Science

**Zoya Fahad Khan**

zoyashaikh.0894@gmail.com

# 1) Introduction:

The Mumbai Suburban Railway (colloquially called local trains or simply locals) consists of exclusive inner suburban railway lines augmented by commuter rail on main lines serving outlying suburbs to serve the Mumbai Metropolitan Region. Spread over 390 kilometers (240 mi), the suburban railway operates 2,342 train services and carries more than 7.5 million commuters daily. By annual ridership (2.64 billion), the Mumbai Suburban Railway is one of the busiest commuter rail systems in the world and it has the most severe overcrowding in the world. Trains run from 04:00 until 01:00, and some trains also run up to 02:30. It is the second largest suburban rail network in terms of route length after the Kolkata Suburban Railway.

Train stations are ideal locations for small businesses to set up shops, because they are hubs of human interaction where hundreds or even thousands of people day and night come and go. Each person in this flow of foot traffic is a potential customer who might need a specific item or purchase on impulse while waiting for a train. To succeed with retail at a train station, one must provide an accessible and affordable shopping experience offering merchandise or services that travellers might not quickly find elsewhere in the route while travelling.

# 2) Business Problem:

Train passengers as well as station and train employees need to have breakfast, lunch, dinner or snacks while travelling. Although food sales are forbidden in some railway stations, many do offer merchants the opportunity to sell food. Foods that attract busy people on the go include egg sandwiches, fries, pizza, burgers, microwaveable or cold prepared meals. Beverages such as coffee, tea, wraps, bottled water, soda and juice also sell well. Thus, the main objective of the project will be to find ideal spots in the city where a new restaurant chain can be put up and also finding out the category of the same, aiming at the above demographic, thereby helping the owners of the outlets to extract maximum profits out of them.
Using Foursquare API, I will cluster most common venues together. Based on the output from foursquare, user can easily determine what location is best suitable to set up a new restaurant business and the category of the business.

# 3) Data:

For this assignment, I will be utilizing the Foursquare API to pull the following location data on top 10 venues in different locations in Mumbai, India.

- Venue Name
- Venue Location
- Venue Category

### 3.1 Neighbourhoods:

The data of the neighbourhoods in Mumbai is extracted out by web scraping using Beautiful Soup library for Python. The neighbourhood data is scraped from a Wikipedia webpage.

```
source = requests.get('https://en.wikipedia.org/wiki/List_of_neighbourhoods_in_Mumbai').text

soup = BeautifulSoup(source, 'lxml')

csv_file = open('mumbai.csv', 'w')

csv_writer = csv.writer(csv_file)

csv_writer.writerow(['Neighborhood'])

mwcg = soup.find_all(class_ = "mw-headline")

length = len(mwcg)

for i in range(1, length):

    lists = mwcg [i].find_all('a')

    for list in lists:

        nbd = list.get('title')

        csv_writer.writerow([nbd])

csv_file.close()
```

### 3.2 Geocoding:

The file contents from mumbai.csv is retrieved into a Pandas Data Frame. The latitude and longitude of the neighbourhoods are retrieved using Google Maps Geocoding API. The geometric location values are then stored into the initial data frame.

```
df['Latitude'] = None # Initializing the latitude array

df['Longitude'] = None# Initializing the longitude array


for i in range(0,len(df),1):
    results = geocoder.geocode(df.iat[i,0] + ",Mumbai,India")
    try:
        lat = results[0]['geometry']['lat'] # Extracts the latitude value
        lng = results[0]['geometry']['lng']
        df.iat[i,df.columns.get_loc('Latitude')] =lat
        df.iat[i,df.columns.get_loc('Longitude')] =lng
    except:
        lat = None
        lng = None
```

### 3.3 Venue Data

From the location data obtained after Web Scraping and Geocoding, the venue data is found out by passing in the required parameters to the Foursquare API and creating another Data Frame to contain all the venue details along with the respective neighbourhoods.

```
explore_df_list = []
for i, nbd_name in enumerate(df['Neighborhood']):
    try :
        ### Getting the data of neighborhood
        nbd_name = df.loc[i, 'Neighborhood']
        nbd_lat = df.loc[i, 'Latitude']
        nbd_lng = df.loc[i, 'Longitude']
        radius = 1000 # Setting the radius as 1000 metres
        LIMIT = 30 # Getting the top 30 venues
        url = 'https://api.foursquare.com/v2/venues/explore?client_id={} \
        &client_secret={}&ll={},{}&v={}&radius={}&limit={}'\
        .format(CLIENT_ID, CLIENT_SECRET, nbd_lat, nbd_lng, VERSION, radius, LIMIT)
        results = json.loads(requests.get(url).text)
        results = results['response']['groups'][0]['items']
        nearby = json_normalize(results) # Flattens JSON
        filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat',
'venue.location.lng']
        nearby = nearby.loc[:, filtered_columns]
        columns = ['Name', 'Category', 'Latitude', 'Longitude']
        nearby.columns = columns
        nearby['Category'] = nearby.apply(get_category_type, axis=1)
        for i, name in enumerate(nearby['Name']):
            s_list = nearby.loc[i, :].values.tolist()  # Converts the numpy array to a python list
            f_list = [nbd_name, nbd_lat, nbd_lng] + s_list
            explore_df_list.append(f_list)

    except Exception as e:
        pass
```

# 4) Methodology

A thorough analysis of the principles of methods, rules, and postulates employed have been made in order to ensure the inferences to be made are as accurate as possible.

### 4.1 Folium

Folium builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the leaflet.js library. All cluster visualization is done with help of Folium which in turn generates a Leaflet map made using OpenStreetMap technology.

```
map_mumbai = folium.Map(location=[latitude, longitude], zoom_start=10)

for lat, lng, neighborhood in zip(df['Latitude'], df['Longitude'], df['Neighborhood']):
    label = '{}'.format(neighborhood)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_mumbai)
map_mumbai
```

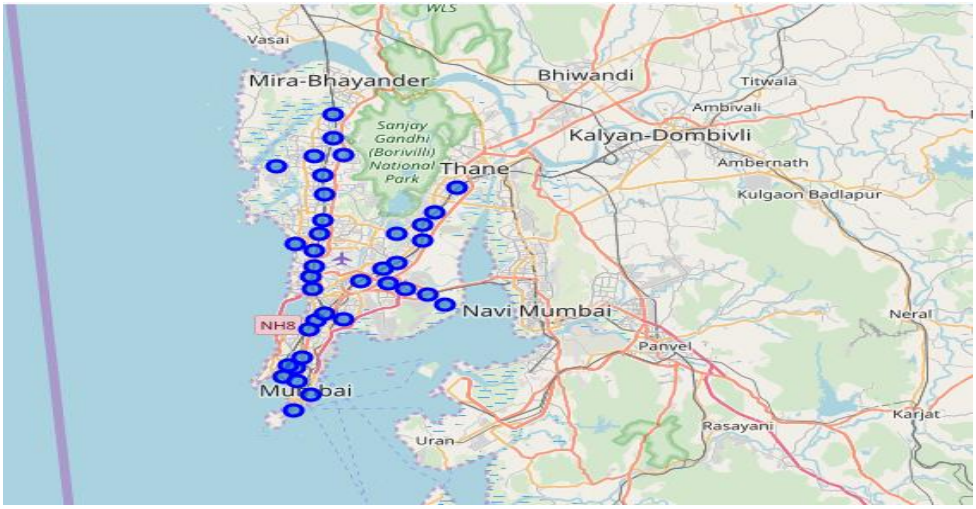


Figure 2: Neighbourhoods of Mumbai.

### 4.2 One hot encoding

One hot encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction. For the K-means Clustering Algorithm, all unique items under Venue Category are one-hot encoded.

```
mumbai_onehot = pd.get_dummies(explore_df[['Venue Category']], prefix="", prefix_sep="")

mumbai_onehot['Neighborhood'] = explore_df['Neighborhood']

fixed_columns = [mumbai_onehot.columns[-1]] + mumbai_onehot.columns[:-1].values.tolist()

mumbai_onehot = mumbai_onehot[fixed_columns]

mumbai_onehot.head()
```

### 4.4 Top 10 most common venues

Due to high variety in the venues, only the top 10 common venues are selected, and a new Data Frame is made, which is used to train the K-means Clustering Algorithm.

```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)
    return row_categories_sorted.index.values[0:num_top_venues]


num_top_venues = 10
indicators = ['st', 'nd', 'rd']
columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = mumbai_grouped['Neighborhood']


for ind in np.arange(mumbai_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] =
return_most_common_venues(mumbai_grouped.iloc[ind, :], num_top_venues)


neighborhoods_venues_sorted.head()
```

### 4.5 Optimal number of clusters

Silhouette Score is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from -1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighbouring clusters. Based on the Silhouette Score of various clusters below 20, the optimal cluster size is determined.

```
import matplotlib.pyplot as plt

%matplotlib inline

def plot(x, y, xlabel, ylabel):

    plt.figure(figsize=(20,10))

    plt.plot(np.arange(2, x), y, 'o-')

    plt.xlabel(xlabel)

    plt.ylabel(ylabel)

    plt.xticks(np.arange(2, x))

    plt.show()
```

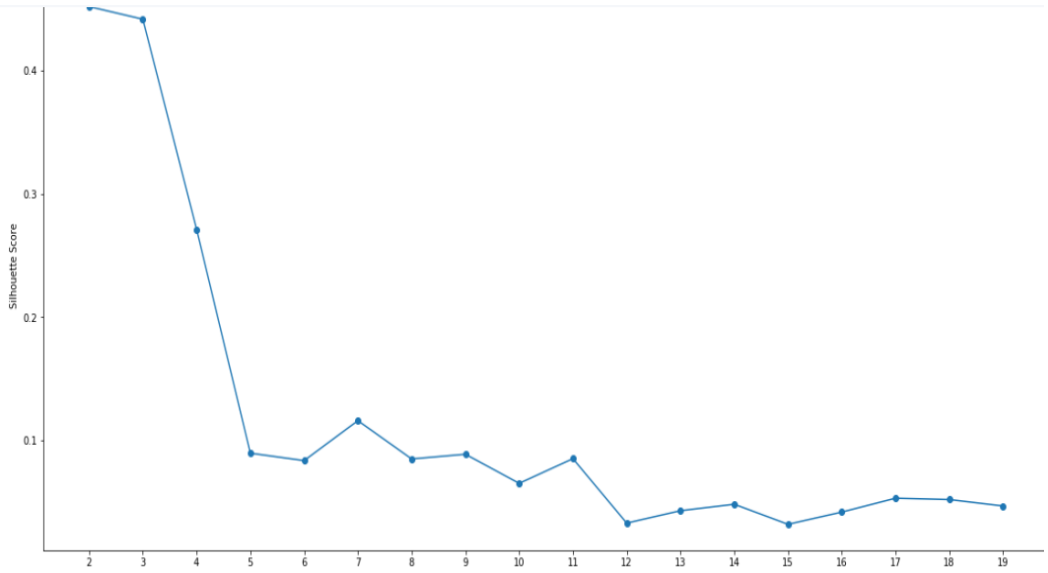


Figure 2: Silhouette score vs Number of clusters.

### 4.6 K-means clustering

The venue data is then trained using K-means Clustering Algorithm to get the desired clusters to base the analysis on. K-means was chosen as the variables (Venue Categories) are huge, and in such situations K-means will be computationally faster than other clustering algorithms.

```
kclusters = optimal

mgc = mumbai_grouped_clustering

kmeans = KMeans(n_clusters = kclusters, init = 'k-means++', random_state = 0).fit(mgc)
```

# 5) Results

The neighbourhoods are divided into n clusters where n is the number of clusters found using the optimal approach. The clustered neighbourhoods are visualized using different colours to make them distinguishable.

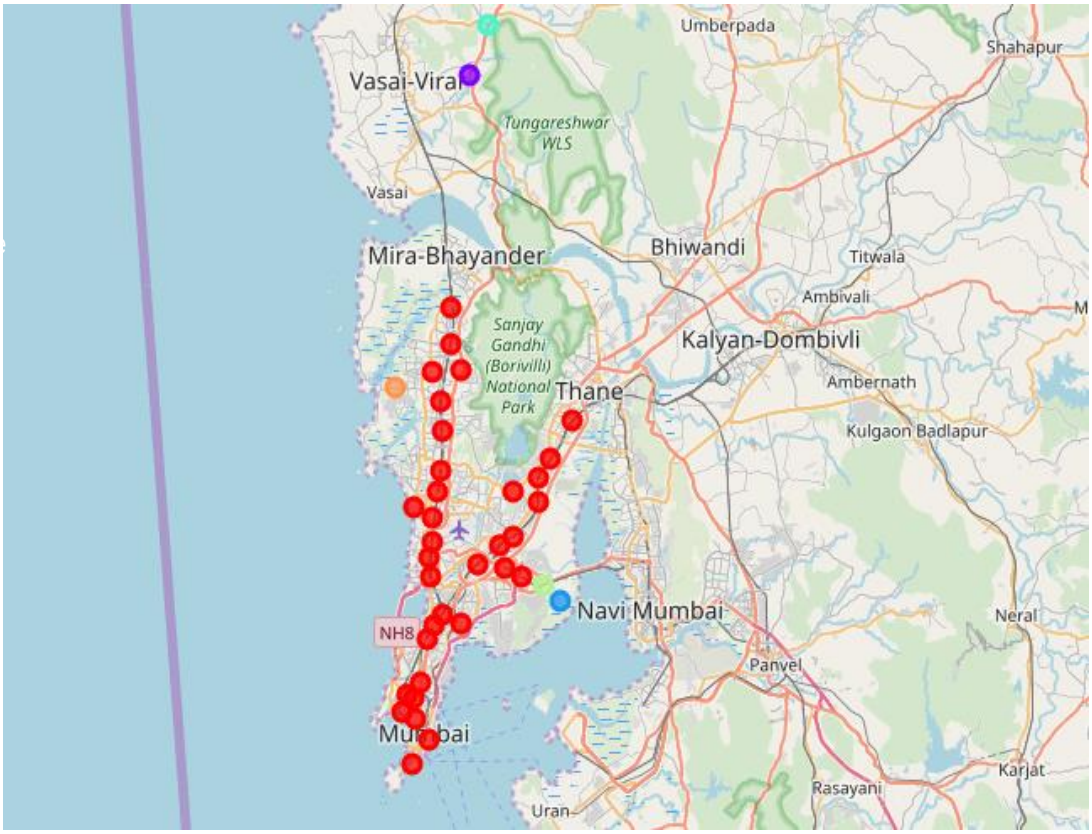


Figure 3: Neighbourhoods of Mumbai (Clustered).

# 6) Discussion:

After analysing the various clusters produced by the Machine learning algorithm, cluster no.1, is a prime fit to solving the problem of finding a cluster with common venue as an Indian Restaurant.

| | Neighborhood | Latitude | Longitude | Cluster Labels | 1st Most Common Venue | 2nd Most Common Venue | 3rd Most Common Venue | 4th Most Common Venue | 5th Most Common Venue | 6th Most Common Venue | 7th Most Common Venue | 8th Most Common Venue | 9th Most Common Venue | 10th Most Common Venue |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Andheri | 19.1197 | 72.8464 | 0 | Indian Restaurant | Fast Food Restaurant | Café | Ice Cream Shop | Sandwich Place | Camera Store | Falafel Restaurant | Coffee Shop | Pizza Place | Burger Joint |
| 1 | Bhayandar | 19.1972 | 72.8114 | 5 | Hot Dog Joint | Diner | Resort | Dance Studio | Dessert Shop | Design Studio | Donut Shop | Electronics Store | Department Store | Food & Drink Shop |
| 2 | Bandra | 19.055 | 72.8402 | 0 | Indian Restaurant | Snack Place | Gourmet Shop | Café | Seafood Restaurant | Park | Indie Movie Theater | Food Truck | Korean Restaurant | Lounge |
| 3 | Borivali | 19.2291 | 72.8574 | 0 | Ice Cream Shop | Chinese Restaurant | Indian Restaurant | Clothing Store | Park | Pizza Place | Scenic Lookout | Sandwich Place | Restaurant | Café |
| 4 | Dahisar | 19.2572 | 72.8575 | 0 | Indian Restaurant | Fast Food Restaurant | Department Store | Café | Coffee Shop | BBQ Joint | Soccer Field | Gym | Restaurant | Sandwich Place |
| 5 | Goregaon | 19.1648 | 72.85 | 0 | Indian Restaurant | Fast Food Restaurant | Snack Place | Design Studio | Indie Movie Theater | Restaurant | Sandwich Place | Seafood Restaurant | Lounge | Bookstore |
| 6 | Jogeshwari | 19.1349 | 72.8488 | 0 | Ice Cream Shop | Indian Restaurant | Coffee Shop | Mughlai Restaurant | Fast Food Restaurant | Smoke Shop | Sandwich Place | Asian Restaurant | Pharmacy | Pizza Place |
| 7 | Juhu | 19.107 | 72.8275 | 0 | Hotel | Café | Chinese Restaurant | Lounge | Seafood Restaurant | Cocktail Bar | Bar | Indian Restaurant | Spanish Restaurant | Restaurant |
| 8 | Kandivali West | 19.2084 | 72.8422 | 0 | Electronics Store | Fast Food Restaurant | Ice Cream Shop | Pizza Place | Coffee Shop | Gym / Fitness Center | Food | Snack Place | Restaurant | Chinese Restaurant |
| 9 | Kandivali East | 19.2102 | 72.8649 | 0 | Pizza Place | Fast Food Restaurant | Indian Restaurant | Lounge | Department Store | Restaurant | Café | Sports Bar | Multiplex | Chinese Restaurant |
| 10 | Khar, Mumbai | 19.0697 | 72.8399 | 0 | Bar | Gym / Fitness Center | Ice Cream Shop | Pub | Café | Seafood Restaurant | Gym | Frozen Yogurt Shop | Fish & Chips Shop | Coffee Shop |

Figure 4: Clusters showing Indian Restaurant as the most common venue

According to most organizations, like the World Bank and the Organization for the Economic Cooperation and Development (OECD), people living on less than US $2 a day are considered poor. For those in the middle classes, the earnings typically lie in the range of US $10 to $100 per day, as expressed in the 2015 purchasing power parities.

India is expected to see a dramatic growth in the middle class, from 5 to 10 percent of the population in 2005 to 90 percent in 2039, by which time a billion people will be added to this group. In 2005, the mean per capita household expenditure was just US $3.20 per day, and very few households exceeded incomes of US $5 per day. Yet, by 2015, half the population had crossed this threshold. By 2025, half the Indian population is expected to surpass US $10 per day.

# 7) Conclusion

As the middle class will grow at a rapid rate in the next upcoming years, opening food outlets catered for that section of the society will see a massive increase in footfall, which would lead to a further increase in business.

If the food outlets have an average rate of US $0.5 equivalent to 15 percent of the per capita household expenditure, for their items, then profits can be expected to be high as the food rates are neither too low or too high for a person of the concerned demographic to spend. Assuming a footfall of 30 people getting off at these stations for each station, 100 trains passing through these stations, and a conversion rate of 20 percent, ordering only one meal, a daily turnover of around US $300 can be expected from these outlets per station.