Applied Data Science Capstone : Coursera

Tanish Ghosh

tanishghosh@gmail.com

1. Introduction

The Brihanmumbai Electricity Supply and Transport (BEST) (also known as the Bombay Electric Supply & Transport, the official name until 1995) is a civic transport and electricity provider public body based in Mumbai, Maharashtra, India. It was originally set up in 1873 as a tramway company called "Bombay Tramway Company Limited".
In 1926, BEST became an operator of motor buses. In 1995 the organisation was renamed "Brihanmumbai Electric Supply & Transport (BEST)" alongside Bombay was renamed to Mumbai. It now operates as an autonomous body under the Municipal Corporation.
BEST operates one of India's largest fleets of buses. The bus transport service covers the entire city and also extends its operations outside city limits into neighbouring urban areas.

As of January 2015, the BEST has a fleet of 3337 buses.[13] The fleet comprises 602 single-decker diesel buses, 2694 CNG buses, 40 AC Midi buses,1000 Non Ac Midi Buses, and 120 double-decker buses. All are tagged with a route number and its corresponding destination. BEST bus routes are spread citywide and to neighboring cities. BEST operates inter-city services to three areas beyond the municipal limits of Mumbai City; i.e., into the limits of the bordering corporations of Navi Mumbai, Thane, and Mira-Bhayandar.

Over 88% of the commuters in Mumbai use public transport. Mumbai has the largest organized bus transport network among major Indian cities.
Public buses operated by the Brihanmumbai Electric Supply and Transport Undertaking (BEST) carry over 3.67 million passengers each day.

The fact that all but the highest income households spend more per month on bus than on rail (see Table 4) reflects the fact that bus fares are higher, per kilometer traveled, than rail fares. At the time of our survey, a person commuting 15 km each way to work by bus paid a fare of Rs. 18 per day or Rs. 450 per month, assuming 25 workdays per month. A person commuting 15 km each way to work by rail paid Rs. 75 for a monthly pass—one-sixth the cost of the bus fare.7 Bus fares, per km, remain higher than rail fares today, although monthly bus passes are now available.

Bus stations are ideal locations for small businesses to set up shops, because hundreds or even thousands of people day and night come and go every day. 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 bus or going somewhere after deboarding a bus.

2. Business Problem

Commuters who travel by bus are often in a hurry and are in need of food that can be prepared and consumed fast in order to reach their destination quickly. Foods that attract busy people on the go include egg sandwiches, fries, pizza, burgers,microwaveable or cold prepared meals. The main objective of the project will be to find ideal spots in the city where fast food retail chains can be put up, aiming at the above demographic, thereby helping the owners of the outlets to extract maximum profits out of them.

3. Data

The data required for this project have been acquired from multiple sources. Any error or discrepancy of the data found is credited to the source only.

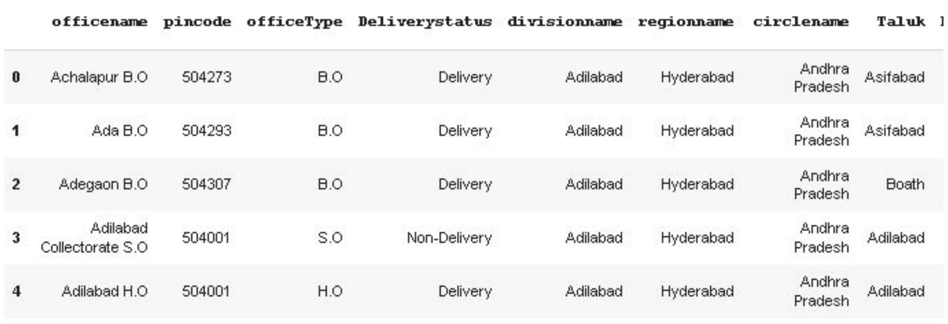
3.1 Neighborhood Data

The neighborhood names of Mumbai are extracted from a .csv file containing all the pincodes of India(Source: https://data.gov.in/resources) along with other details using Pandas library for Python. The neighborhood names and their corresponding pincode is stored in a Pandas DataFrame.

Code

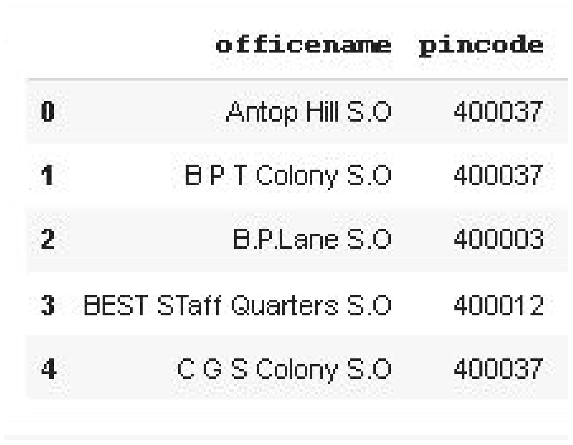
```
import pandas as pd
df = pd.read_csv('PO.csv')
neighborhood=[]
for i in df.index:
  district = df['regionname'][i]
  if district=="Mumbai":
      l1 = [df['officename'][i],df['pincode'][i]]
      neighborhood.append(l1)
neighborhood = pd.DataFrame(neighborhood,
columns=['officename','pincode'])
```

Initial dataset



	officename	pincode	officeType	Deliverystatus	divisionname	regionname	circlename	Taluk
0	Achalapur B.O	504273	B.O	Delivery	Adilabad	Hyderabad	Andhra Pradesh	Asifabad
1	Ada B.O	504293	B.O	Delivery	Adilabad	Hyderabad	Andhra Pradesh	Asifabad
2	Adegaon B.O	504307	B.O	Delivery	Adilabad	Hyderabad	Andhra Pradesh	Boath
3	Adilabad Collectorate S.O	504001	S.O	Non-Delivery	Adilabad	Hyderabad	Andhra Pradesh	Adilabad
4	Adilabad H.O	504001	H.O	Delivery	Adilabad	Hyderabad	Andhra Pradesh	Adilabad

Neighborhoods of Mumbai



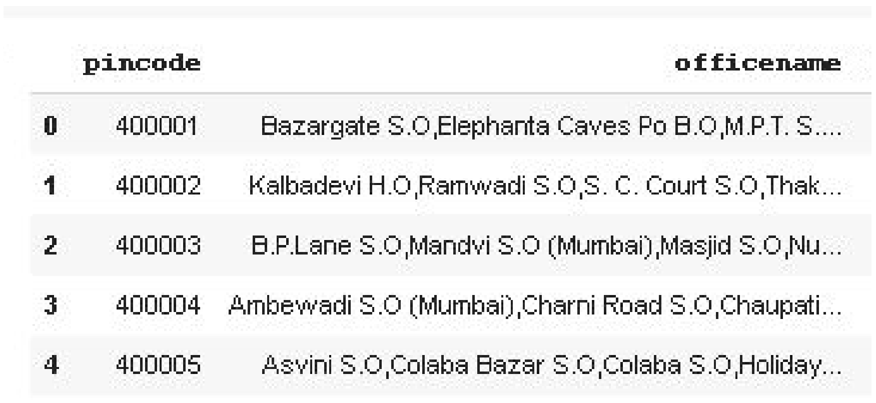
	officename	pincode
0	Antop Hill S.O	400037
1	B P T Colony S.O	400037
2	B.P.Lane S.O	400003
3	BEST STaff Quarters S.O	400012
4	C G S Colony S.O	400037

We then group neighborhoods having same pincode in order to remove redundancies.

Code

```
neighborhood1=neighborhood.groupby(['pincode'])['officename'].apply(lam
bda x:','.join(x.astype(str))).reset_index()
```

Neighborhoods of Mumbai (after grouping by pincode)



	pincode	officename
0	400001	Bazargate S.O,Elephanta Caves Po B.O,M.P.T. S....
1	400002	Kalbadevi H.O,Ramwadi S.O,S. C. Court S.O,Thak...
2	400003	B.P.Lane S.O,Mandvi S.O (Mumbai),Masjid S.O,Nu...
3	400004	Ambewadi S.O (Mumbai),Charni Road S.O,Chaupati...
4	400005	Asvini S.O,Colaba Bazar S.O,Colaba S.O,Holiday...

3.2 Coordinates of Neighborhoods

The latitude and longitude of the neighbourhoods are retrieved using geopy library. Geopy locate the coordinates of addresses, using third-party geocoders and other data sources. The geometric location values are then stored into the initial dataframe.

Code

```
import geopy
from geopy.geocoders import Nominatim
loc = []
pincode = neighborhood1[‘pincode’]
for i in pincode:
      geolocator = Nominatim(user_agent='my-application')
      time.sleep(.45)
      location = geolocator.geocode((str)(i)+"Mumbai India")
      print(i, location.latitude, location.longitude)
      loc.append([i,location.latitude, location.longitude])
temp = pd.DataFrame(loc)
temp.columns = [‘pincode’,’longitude’,’latitude’]
neighborhood1['longitude']=temp['longitude']
neighborhood1['latitude']=temp['latitude']
```

3.3 Venue Data

From the location data, the venue data is found out bypassing in the required parameters to the FourSquare API, and creating another DataFrame to contain all the venue details along with the respective neighborhoods.

Code

```
def getNearbyVenues(names, latitudes, longitudes, radius=500):
    LIMIT = 15
    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)
        url = 
'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secr
et={}&v={}&ll={},{}&radius={}&limit={}'.format(
            CLIENT_ID, 
            CLIENT_SECRET, 
            VERSION, 
            lat, 
            lng, 
            radius, 
            LIMIT)
        results = 
requests.get(url).json()["response"]['groups'][0]['items']

        venues_list.append([(
             name,
             lat,
             lng,
             v['venue']['name'],
             v['venue']['location']['lat'],
             v['venue']['location']['lng'],
             v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list 
for item in venue_list])
    nearby_venues.columns = ['Neighborhood','Neighborhood Latitude', 
'Neighborhood Longitude', 'Venue', 'Venue Latitude','Venue Longitude', 
'Venue Category']

    return(nearby_venues)

mumbai_venues = getNearbyVenues(names=neighborhood1['officename'],
                                   latitudes=neighborhood1['latitude'],

longitudes=neighborhood1['longitude']
                                  )
```

4. Methodology

A detailed analysis of the principles of methods and rules employed have been made in order to ensure the inferences to be made are as accurate as possible. In the initial development stages, the coordinates were being extracted through various third-party API services.

However, they were found to be erroneous and had a large error percentage. After many iterations, geopy package is used which locates the coordinates of a location based on various geocoders which was found to have the least number of collisions in our analysis.

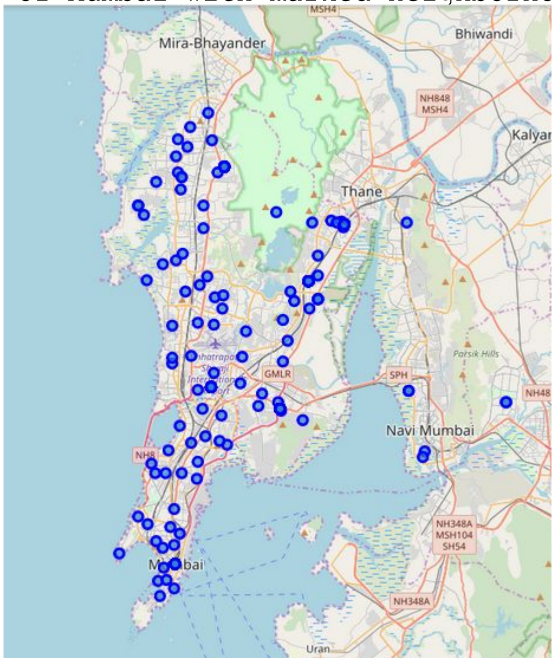
4.1 Folium

folium builds on the data wrangling strengths of the Python ecosystem and the mapping strengths of the leaflet.js library. Manipulate your data in Python, then visualize it in on a Leaflet map via folium.
It enables both the binding of data to a map for choropleth visualizations as well as passing rich vector/raster/HTML visualizations as markers on the map.

Code

```
map_mumbai = folium.Map(location=[latitude, longitude], zoom_start=10)
count = 0
for lat, lng, neighborhood in zip(temp['latitude'], temp['longitude'],
temp['pincode']):
    label = '{}'.format(neighborhood)
    label = folium.Popup(label, parse_html=True)
    li = [lat,lng]
    folium.CircleMarker(
        li,
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_mumbai)
    count = count+1
```

Map of Mumbai with marked neighborhoods



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.

Code

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

```
fixed_columns = [mumbai_onehot.columns[-1]] +
list(mumbai_onehot.columns[:-1])
mumbai_onehot = mumbai_onehot[fixed_columns]
```

4.3 Top 10 most common venues

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

Code

```
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()
```

4.4 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.

Code

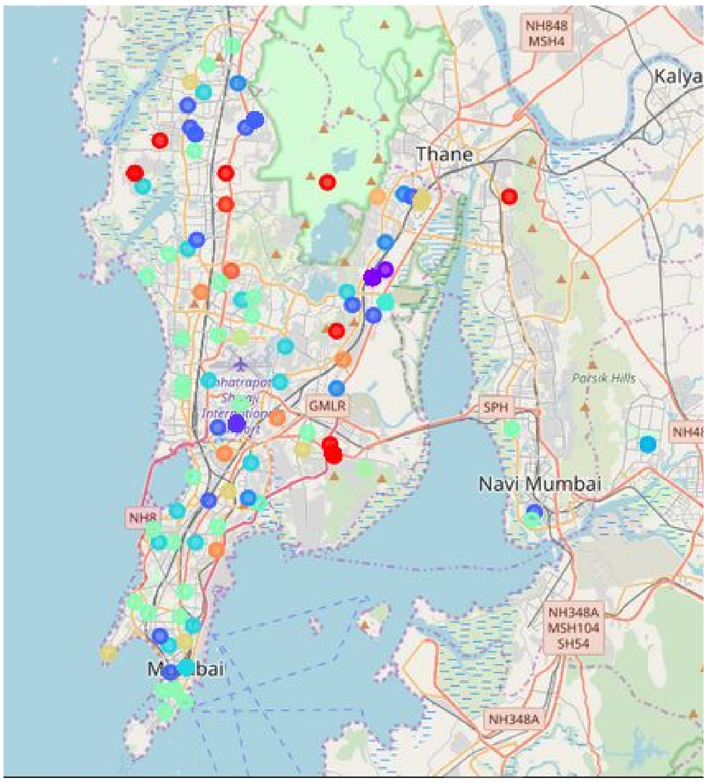
```
kmeans = KMeans(n_clusters=optimal_value,
random_state=0).fit(mumbai_grouped_clustering)
```

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 so as to make them distinguishable.

Code

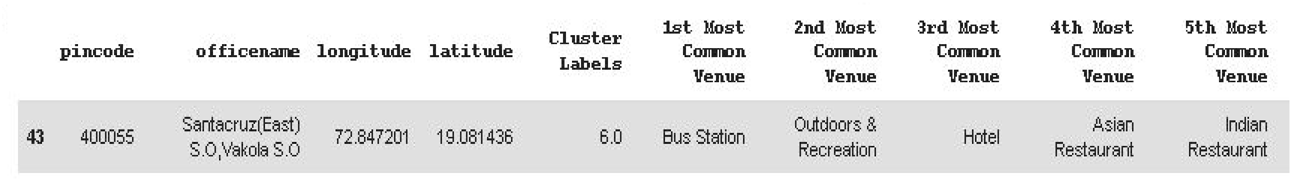
```
map_clusters = folium.Map(location=[latitude, longitude],
zoom_start=11)
x = np.arange(optimal_value)
ys = [i + x + (i*x)**2 for i in range(optimal_value)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(mumbai_merged['latitude'],
mumbai_merged['longitude'], mumbai_merged['officename'],
mumbai_merged['Cluster Labels']):
    if np.isnan(cluster):
        cluster = (int)(np.nan_to_num(cluster))
    else:
        cluster = (int)(cluster)
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster),
parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)
```



6. Discussion

After analyzing the various clusters produced by the Machine learning algorithm, cluster no. 6, is a prime fit to solving the problem of finding a cluster with common venue as a bus station.

Cluster having Train Station as most common venue



	pincode	officename	longitude	latitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue
43	400055	Santacruz(East) S.O,Vakola S.O	72.847201	19.081436	6.0	Bus Station	Outdoors & Recreation	Hotel	Asian Restaurant	Indian Restaurant

The two places mentioned are Santacruz East and Vakola which are at the heart of the city. The demographic of the population in these areas fall under the lower middle class of the society.

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.