COURSERA CAPSTONE PROJECT: APPLIED DATA SCIENCE

A perspective view on local businesses in Dhaka, Bangladesh

MD ISLAM
SEATTLE, WASHINGTON, USA
tomislam86@gmail.com

Table of Contents

1 Introduction

Dhaka is the capitol city of Bangladesh. I picked this city because I lived there for 4 years and I am familiar with the demography of this part of the world. Dhaka's 2020 population is now estimated at 21,005,860. In 1950, the population of Dhaka was 335,760. Dhaka has grown by 3,408,684 since 2015, which represents a 3.60% annual change. These population estimates and projections come from the latest revision of the UN World Urbanization Prospects. These estimates represent the Urban agglomeration of Dhaka, which typically includes Dhaka's population in addition to adjacent suburban areas. [Source]

Dhaka is a diverse city located in central Bangladesh along the Buriganga River. Not only is it the capital city, but it is also the largest in the country.

Dhaka is the most populated city in Bangladesh, and it is also one of the most populated cities in the world. The Greater Dhaka Area has a population of over 18 million as of 2016, while the city itself has a population estimated at 8.5 million.

2 Business Problem

Migration from rural areas of the country to urban Dhaka is an active contributor to the population growth. This rural migration accounted for 60% population growth throughout the 1960s and 1970s. While this growth has slowed since that time, Dhaka continues to show steady growth, with estimates placing the 2020 population at almost 21 million, while 2030 may see as many as 27.3 million residents.

With this huge population comes a lot of business opportunities. But since this is no secret to every business owner, there are a lot of competition. The ideal opportunity might come out with exploratory data analysis and data visualization.

2.1 Problem and approach

To find the answers to the following questions:

Q1) Pick an area of interest,
Q2) Explorer businesses in this location,
Q3) What different business are offered in this region,
Q4) Is it possible to do segmentation and clustering based on number of similar businesses?
Q5) What business ideas can come up with the most common venues in these locations?

3 Data

The data for this project has been retrieved and processed after careful consideration of the accuracy and availability of the data.

3.1 Neighbourhoods

The data of the neighbourhoods in Dhaka was scrapped BeautifulSoup from a Wikipedia page.

```
Code
```

```
source =

requests.get('https://en.wikipedia.org/wiki/Category:Neighbourhoods_in_Dhaka').text

soup = BeautifulSoup(source, 'lxml')
csv_file = open('dhaka.csv', 'w')
csv_writer = csv.writer(csv_file)
csv_writer.writerow(['Neighbourhood'])

mwcg = soup.find_all(class_ = "mw-category-group")

length = len(mwcg) # Gets the length of number of `mw-category-groups` present

for i in range(1, length):  # Gets all the neighbourhoods
    lists = mwcg [i].find_all('a')
    for list in lists:
        nbd = list.get('title') # Gets the title of the neighbourhood
        csv_writer.writerow([nbd]) # Writes the name of the neighbourhood in the csv file
csv_file.close()
```

3.2 Geocoding

At this point we can generate dhaka.csv using Pandas DataFrame. We will use this DataFrame to retrieve the longitude and latitude of the neighbourhoods using Google Maps Geocoding API and store them into initial DataFrame.

Code

```
import json
from pandas.io.json import json_normalize

latitudes = [] # Initializing the latitude array
longitudes = [] # Initializing the longitude array

for nbd in df["Neighbourhood"] :
    place_name = nbd + ",Dhaka,Bangladesh" # Formats the place name
    url =  'https://maps.googleapis.com/maps/api/geocode/json?address={}&key={}'.format(place_name,
API_KEY) # Gets the proper url to make the API call
    obj = json.loads(requests.get(url).text) # Loads the JSON file in the form of a python dictionary

    results = obj['results'] # Extracts the results information out of the JSON file
    lat = results[0]['geometry']['location']['lat'] # Extracts the latitude value
    lng = results[0]['geometry']['location']['lng'] # Extracts the longitude value

    latitudes.append(lat) # Appending to the list of latitudes
    longitudes.append(lng) # Appending to the list of longitudes
```

Code (continued)

```
df['Latitude'] = latitudes
df['Longitude'] = longitudes

df.head()
```

Our initial DataFrame looks like this:

	Neighbourhood	Latitude	Longitude
0	Bailey Road, Dhaka	23.741785	90.405797
1	Banani DOHS	23.793263	90.398972
2	Banasree	23.761935	90.433141
3	Bangla Bazar	23.706556	90.411265
4	Baridhara	23.799898	90.420766

Figure 1: Initial DataFrame.

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 DataFrame to contain all the venue details along with the respective neighbourhoods.

Code

```
explore_df_list = []

for i, nbd_name in enumerate(df['Neighbourhood']):

   try :
      ### Getting the data of neighbourhood
      nbd_name = df.loc[i, 'Neighbourhood']
      nbd_lat = df.loc[i, 'Latitude']
      nbd_lng = df.loc[i, 'Longitude']

      radius = 500 # Setting the radius as 500 metres
      LIMIT = 30 # Getting the top 30 venues
```

Code (continued)

```
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
        CLIENT_ID,
        CLIENT_SECRET,
        VERSION,
        nbd_lat,
        nbd_lng,
        radius,
        LIMIT)

    results = json.loads(requests.get(url).text)
    results = results['response']['groups'][0]['items']

    nearby = io.json.json_normalize(results) # Flattens JSON

    # Filtering the columns
    filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng']
    nearby = nearby.loc[:, filtered_columns]

    # Renaming the columns
    columns = ['Name', 'Category', 'Latitude', 'Longitude']
    nearby.columns = columns

    # Gets the categories
    nearby['Category'] = nearby.apply(get_category_type, axis=1)

    # Gets the data required
    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

The approach consists of different stages of a Data Science project in accordance with the course guidelines. The principles of methods and rules were maintained in order to infer our business goal as accurately as possible..

4.1 Accuracy of the Geocoding API

After much research and different factors into consideration, Google Geocoding API was ahead of any available APIs for this project. This was also extremely easy to use and less prone to error. A collision error was analyzed to find out the accuracy of our API.

Code

```
col = 0
explored_lat_lng = []
for lat, lng, neighbourhood in zip(df['Latitude'], df['Longitude'], df['Neighbourhood']):
    if (lat, lng) in explored_lat_lng:
        col = col + 1
    else:
        explored_lat_lng.append((lat, lng))

print("Collisions : ", col)
```

4.2 Folium

All cluster visualization are done with help of Folium which in turn generates a Leaflet map made using OpenStreetMap technology.

Code

```
# Dhaka latitude and longitude using Google search
dhaka_lat = 23.8103
dhaka_lng = 90.4125

# Creates map of Dhaka using latitude and longitude values
map_dhaka = folium.Map(location=[dhaka_lat, dhaka_lng], zoom_start=12)

# Add markers to map
for lat, lng, neighbourhood in zip(df['Latitude'], df['Longitude'], df['Neighbourhood']):
    label = '{}'.format(neighbourhood)
    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_dhaka)

map_dhaka
```

This gives us the locations superimposed on the map.

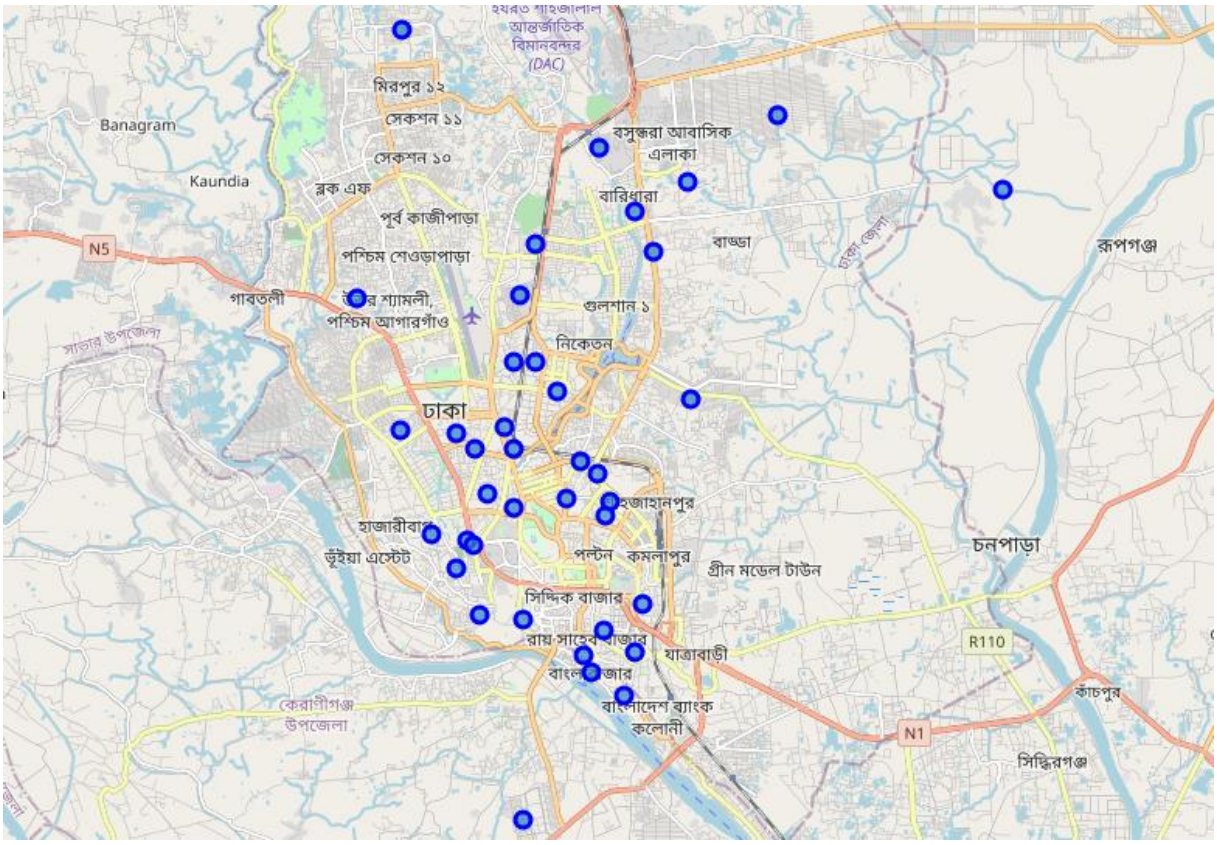

Figure 2: Neighbourhoods of Dhaka.

4.3 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

```
# One hot encoding
dhaka_onehot = pd.get_dummies(explore_df[['Venue Category']], prefix="", prefix_sep="")

# Add neighbourhood column back to dataframe
dhaka_onehot['Neighbourhood'] = explore_df['Neighbourhood']

# Move neighbourhood column to the first column
fixed_columns = [dhaka_onehot.columns[-1]] + dhaka_onehot.columns[:-1].values.tolist()
dhaka_onehot = dhaka_onehot[fixed_columns]

dhaka_onehot.head()
```

4.4 Top 10 most common venues

We will try to determine the top 10 most common venues using our DataFrame. This region has a variety of venues hence only top 10 were selected to use it in K-means clustering algorithm.

Code

```
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']

# Create columns according to number of top venues
columns = ['Neighbourhood']
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))

# Create a new dataframe
neighbourhoods_venues_sorted = pd.DataFrame(columns=columns)
neighbourhoods_venues_sorted['Neighbourhood'] =dhaka_grouped['Neighbourhood']

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

neighbourhoods_venues_sorted.head()
```

4.5 Optimal number of clusters

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

Code

```
max_range = 20 # Maximum range of clusters
from sklearn.metrics import silhouette_samples, silhouette_score

indices = []
scores = []

for kclusters in range(2, max_range) :

    # Run k-means clustering
    kgc = dhaka_grouped_clustering
    kmeans = KMeans(n_clusters = kclusters, init = 'k-means++', random_state = 0).fit_predict(kgc)
```

```
    # Gets the score for the clustering operation performed
    score = silhouette_score(kgc, kmeans)

    # Appending the index and score to the respective lists
    indices.append(kclusters)
    scores.append(score)

plot(max_range, scores, "No. of clusters", "Silhouette Score")
```

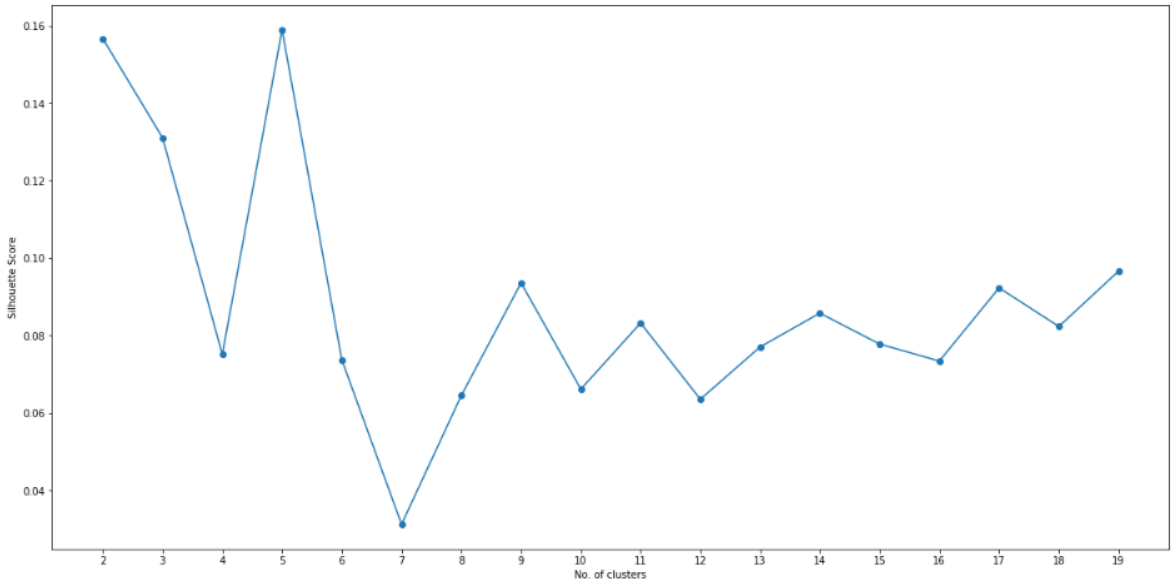

Figure 3: Silhouette score vs Number of clusters.

4.6 K-means clustering

From the graph the optimal number is found out to be considered for our clustering:

```
In [55]: opt = np.argmax(scores) + 2 # Finds the optimal value
         opt

Out[55]: 5
```

Figure 4: Optimum value of k-clusters.

At this point the venue data is trained using K-means Clustering Algorithm. This algorithm was chosen because K-means will be computationally faster than other clustering algorithms in our scenario.

Code

```
kclusters = opt

# Run k-means clustering
kgc = dhaka_grouped_clustering
kmeans = KMeans(n_clusters = kclusters, init = 'k-means++', random_state = 0).fit(kgc)
```

5 Results

The neighbourhood are color coded in n clusters to separate from one another.

Code

```
# Create map
map_clusters = folium.Map(location=[dhaka_lat, dhaka_lng], zoom_start=11.5)

# Set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
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(dhaka_merged['Latitude'], dhaka_merged['Longitude'], dhaka_merged['Neighbourhood'], dhaka_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' (Cluster ' + str(cluster + 1) + ')', parse_html=True)
    map_clusters.add_child(
        folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7))

map_clusters
```

We get the clustering below.

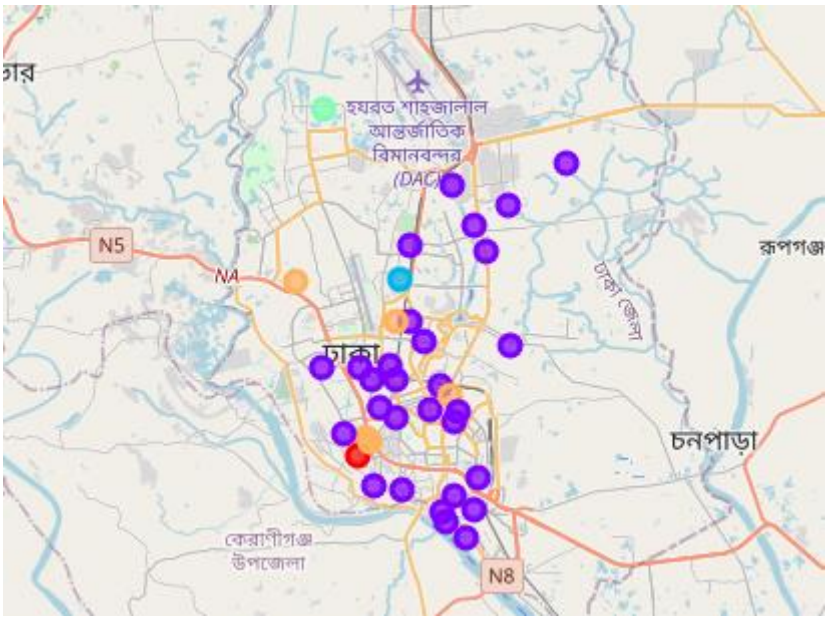

Figure 5: Neighbourhoods of Dhaka (Clustered).

6 Discussion

We analyzed the K-Means Clustering and it appears Shopping Mall/Markets are the most common venue. This gives us idea that a range of business services can be setup within a Mall which also can be close to a Bus Station.

	Neighbourhood	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
15	Kallyanpur	Bus Station	Theater	Diner	Comfort Food Restaurant	Convenience Store	Cosmetics Shop	Cupcake Shop	Department Store	Dessert Shop	Electronics Store
20	Malibagh	Shopping Mall	Market	BBQ Joint	Theater	Diner	Comfort Food Restaurant	Convenience Store	Cosmetics Shop	Cupcake Shop	Department Store
23	Nakhalpara	Shopping Mall	Bookstore	Theater	Diner	Comfort Food Restaurant	Convenience Store	Cosmetics Shop	Cupcake Shop	Department Store	Dessert Shop
25	New Market, Dhaka	Market	Bus Station	Bookstore	Theater	Coffee Shop	Convenience Store	Cosmetics Shop	Cupcake Shop	Department Store	Dessert Shop
27	Nilkhet	Bus Station	Market	Bookstore	Theater	Coffee Shop	Convenience Store	Cosmetics Shop	Cupcake Shop	Department Store	Dessert Shop

Figure 6: Cluster having most common venue

An interesting finding is the demography of these 5 locations. It ranges in the middle-class income families who can be the target audience for any product or service. This can be invaluable information for someone with a business plan who is looking for a location of the business.

Area	Demography
Kallyanpur	Middle Class
Malibagh	Upper Middle Class
Nakhalpara	Lower Middle Class
New Market	Middle Class
Nilkhet	Middle Class

7 Conclusion

As the current pandemic looms on us, a big part of the families who frequently commute to locations like Shopping malls or markets, will stay back at home. This is another opportunity to provide them with what they need with online services. A delivery service in these areas will prove to be extremely profitable among other possibilities. This is just an overview. This project has a lot of scopes for future growth and based on that we can further analyze and explore other possibilities.