Business Development New Ice Cream shop venue – A Report

RAGHUNATH NAIR

IBM

Introduction

A BUSINESS DEVELOPMENT CONSULTANT HAS BEEN TASKED TO MAKE RECOMMENDATIONS TO A VENTURE CAPITALIST ON OPENING A NEW ICE CREAM SHOP IN BANGALORE.

THE CLIENT WOULD LIKE A NEIGHBOURHOOD WITH OTHER INDIAN RESTAURANTS, BUT A LOCATION HAVING NOT MUCH COMPETITION IN THE AREA

THE PROJECT PROVIDES AN EXPLORATORY DATA ANALYSIS AND VISUALIZATIONS TO EVENTUALLY COME TO A RECOMMENDED LOCATION(S).

Data

The data to be used for this project consists the Foursquare location data for the City of Bangalore. More precisely the dataset would use the following features for exploratory data analysis and necessary machine learning involved.

Neighbourhood, Neighbourhood Latitude, Neighbourhood Longitude, Venue, Venue Latitude, Venue Longitude, Venue Categories

Methodology

Get Neighbourhoods in Bangalore and assign variable source and initializing beautifulsoup object to soup. Beautiful Soup is a Python library for pulling data out of HTML and XML files.

128 neighbourhoods were retrieved by the API

Get Neighbourhoods in Bangalore and assigning variable `source` and iniatilizing `beautifulsoup` object to `soup`

Beautiful Soup is a Python library for pulling data out of HTML and XML files

In [31]:
```
source = requests.get('https://en.wikipedia.org/wiki/Category:Neighbourhoods_in_Bangalore').text
soup = BeautifulSoup(source, 'lxml')
```

Initialize `csv_writer` object and write name of columns on it as the first row

In [32]:
```
csv_file = open('bangalore.csv', 'w')
csv_writer = csv.writer(csv_file)
csv_writer.writerow(['Neighbourhood'])
```

Out[32]: 15

Scrape the page to extract the list of neighbourhoods in Bangalore

In [33]:
```
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()
```

Read the csv into a pandas dataframe

In [34]:
```
df = pd.read_csv('bangalore.csv')
df.head()
```

Out[34]:

	Neighbourhood
0	Adugodi
1	Agara, Bangalore
2	Ananthnagar
3	Anjanapura
4	Arekere

Methodology

Get coordinates for each neighbourhood

Use GEOPY library to get the latitude and longitude values of the above neighbourhoods in Bangalore City

Get the number of neighbourhoods in Bangalore

In [35]:
```
df.shape
```

Out[35]: (128, 1)

Use geopy library to get the latitude and longitude values of the neighbourhoods in Bangalore City.

In order to define an instance of the geocoder, we need to define a user_agent. We will name our agent ny_explorer, as shown below.

In [20]:
```
import json

latitudes = [] # Initializing the latitude array
longitudes = [] # Initializing the longitude array
geolocator = Nominatim(user_agent="ny_explorer1")
cnt=1

for nbd in df["Neighbourhood"] :
    address = nbd + ",Bangalore,India" # Formats the place name
    # giving a time out of 15 seconds so that it waits before quitting
    location = geolocator.geocode(address, timeout=15)
    if (location):
        print("{} - {} - {}".format(cnt,address,location))
        lat = location.latitude
        lng = location.longitude
        latitudes.append(lat) # Appending to the list of latitudes
        longitudes.append(lng) # Appending to the list of longitudes
        cnt=cnt + 1
    else:
        latitudes.append(0.0) # Appending to the list of latitudes
        longitudes.append(0.0) # Appending to the list of longitudes
        print("location not found for {}".format(address))
```

```
1 - Adugodi,Bangalore,India - Adugodi, South Zone, Bengaluru, Bangalore Urban, Karnataka, 560095, India
2 - Agara, Bangalore,Bangalore,India - Agara, Kanakapura taluk, Ramanagara district, Karnataka, India
location not found for Ananthnagar,Bangalore,India
3 - Anjanapura,Bangalore,India - Anjanapura, Bommanahalli Zone, Bengaluru, Bangalore Urban, Karnataka, India
4 - Arekere,Bangalore,India - Arekere, Bommanahalli Zone, Bengaluru, Bangalore Urban, Karnataka, 560076, India
5 - Austin Town,Bangalore,India - Austin Town, East Zone, Bengaluru, Bangalore Urban, Karnataka, - 560095, India
location not found for Babusapalya,Bangalore,India
6 - Bagalur, Bangalore Urban,Bangalore,India - Bagaluru, Bangalore Urban, Karnataka, IAM IN BANG, India
location not found for Bahubalinagar,Bangalore,India
7 - Banashankari,Bangalore,India - Banashankari, Kanakapura Road, Banashankari Temple ward, South Zone, Bengaluru, Bangalore Urban, Karnataka, 560070, India
8 - Banaswadi,Bangalore,India - Banaswadi, East Zone, Bengaluru, Bangalore Urban, Karnataka, 560043, India
```

Methodology

Visualize using Folium

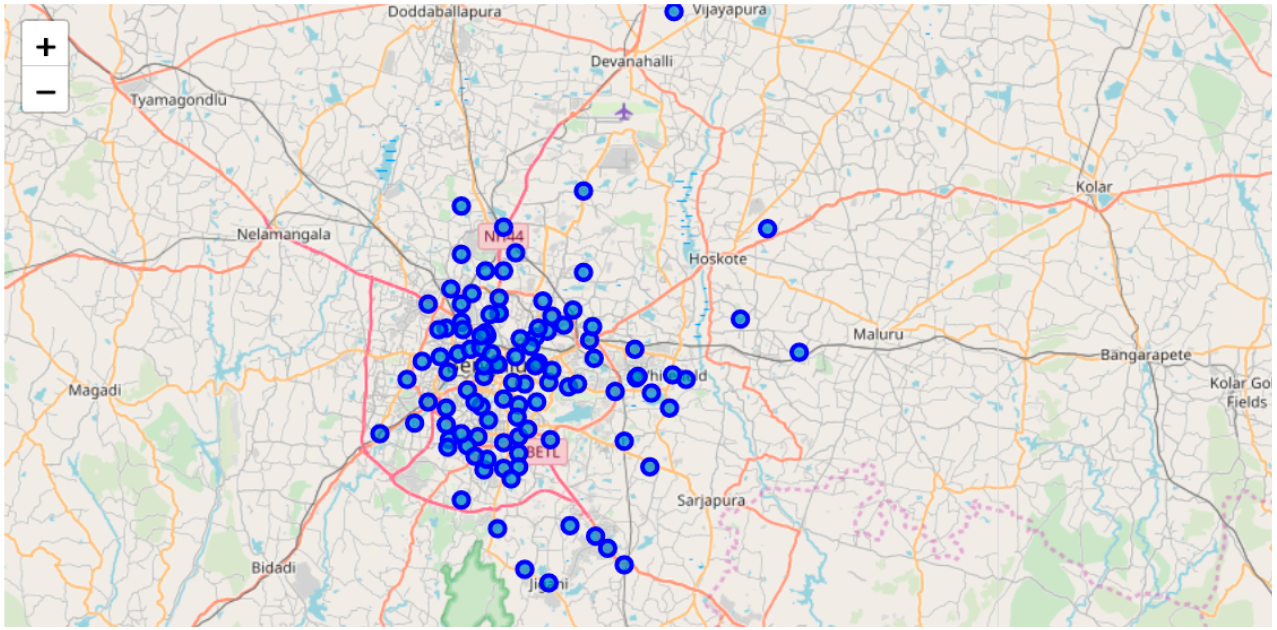
It makes sense to visualize the datasets at each stage of the data wrangling as it could stimulate new ideas and provide different perspectives. Then we create a map of Bangalore using latitude and longitude values

Creating a folium map of Bangalore

In [41]:

```
# Bangalore latitude and longitude using Google search
blr_lat = 12.9116225
blr_lng = 77.6388622

# Creates map of Bangalore using latitude and longitude values
map_bangalore = folium.Map(location=[blr_lat, blr_lng], zoom_start=10)

# 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_bangalore)

map_bangalore
```

Out[41]:



Methodology

Top 10 most venue categories

Use the foursquare APIs to get the nearest X venues and venue categories, which will be used for clustering the neighbourhoods. Since there are so many venue categories, only Top 15 were chosen for this exercise

Creating a new dataframe to get the top 10 venues

In [49]:

```
num_top_venues = 15
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'] = bangalore_grouped['Neighbourhood']

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

neighbourhoods_venues_sorted.to_csv("blr_neighbourhoods_venues_sorted.csv")
print(neighbourhoods_venues_sorted.shape)
neighbourhoods_venues_sorted.head()
```

```
(88, 16)
```

Out[49]:

	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	11th Most Common Venue	12th Most Common Venue	13th Most Common Venue	14th Most Common Venue	15th Most Common Venue
0	Adugodi	Indian Restaurant	Lounge	Dessert Shop	Coffee Shop	Multiplex	Café	Donut Shop	Brewery	Creperie	Shopping Mall	Bookstore	Men's Store	Juice Bar	Punjabi Restaurant	Gaming Cafe
1	Anjanapura	ATM	Pool	Campground	Creperie	Cricket Ground	Field	Fast Food Restaurant	Farmers Market	Falafel Restaurant	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Donut Shop	Dive Bar	Diner
2	Arekere	Indian Restaurant	Department Store	Pizza Place	Sporting Goods Shop	Ice Cream Shop	Fast Food Restaurant	Chinese Restaurant	Fish Market	Café	Business Service	BBQ Joint	South Indian Restaurant	Andhra Restaurant	Mughlai Restaurant	Burger Joint
3	Austin Town	Tea Room	Indian Restaurant	Cocktail Bar	Indie Movie Theater	Clothing Store	Middle Eastern Restaurant	Donut Shop	Café	Mediterranean Restaurant	Market	Shopping Mall	Soccer Stadium	Italian Restaurant	Boutique	Toy / Game Store
4	BTM Layout	Bakery	Snack Place	Ice Cream Shop	Vegetarian / Vegan Restaurant	Café	Pizza Place	Indian Restaurant	Chinese Restaurant	BBQ Joint	Sandwich Place	Dim Sum Restaurant	Italian Restaurant	Restaurant	Fast Food Restaurant	Park

In [50]:

```
print(neighbourhoods_venues_sorted.shape)
```

```
(88, 16)
```

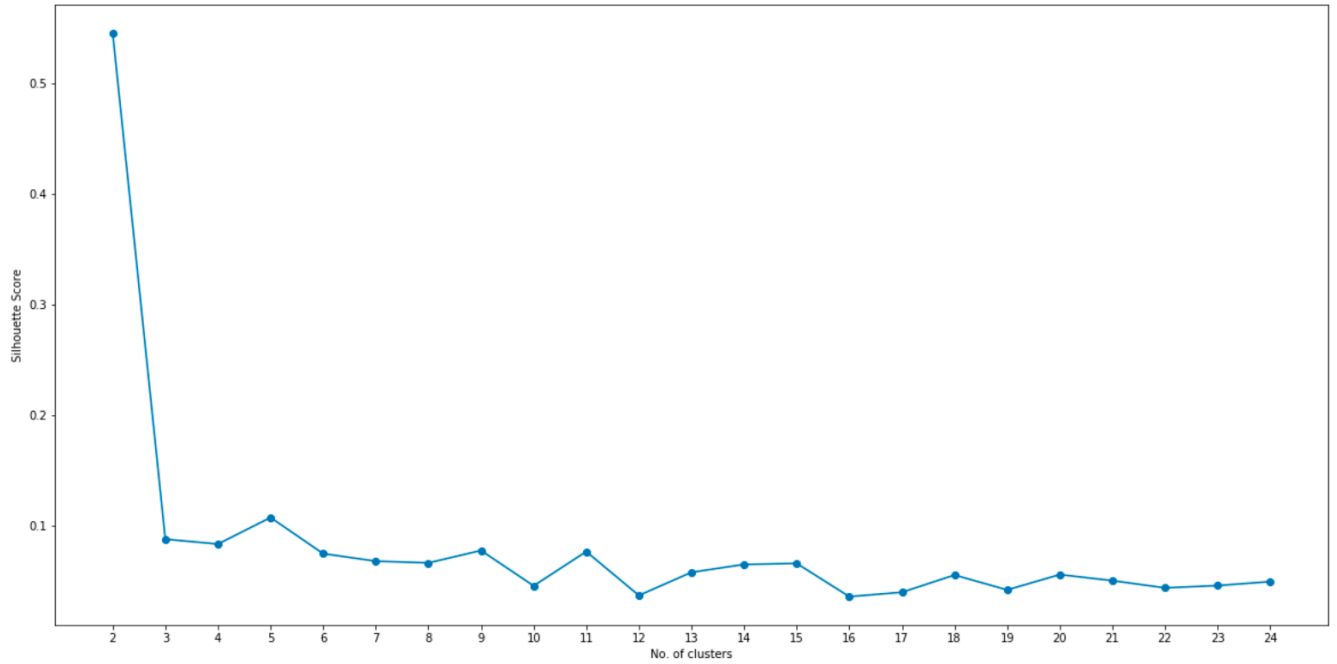
Methodology

Optimal number of clusters

The optimal number of clusters is found out by the silhouette score whose **value** 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.

In [54]:
```
from sklearn.metrics import silhouette_samples, silhouette_score

indices = []
scores = []

for kclusters in range(2, max_range) :

    # Run k-means clustering
    kgc = bangalore_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)
```

In [55]:
```
plot(max_range, scores, "No. of clusters", "Silhouette Score")
```

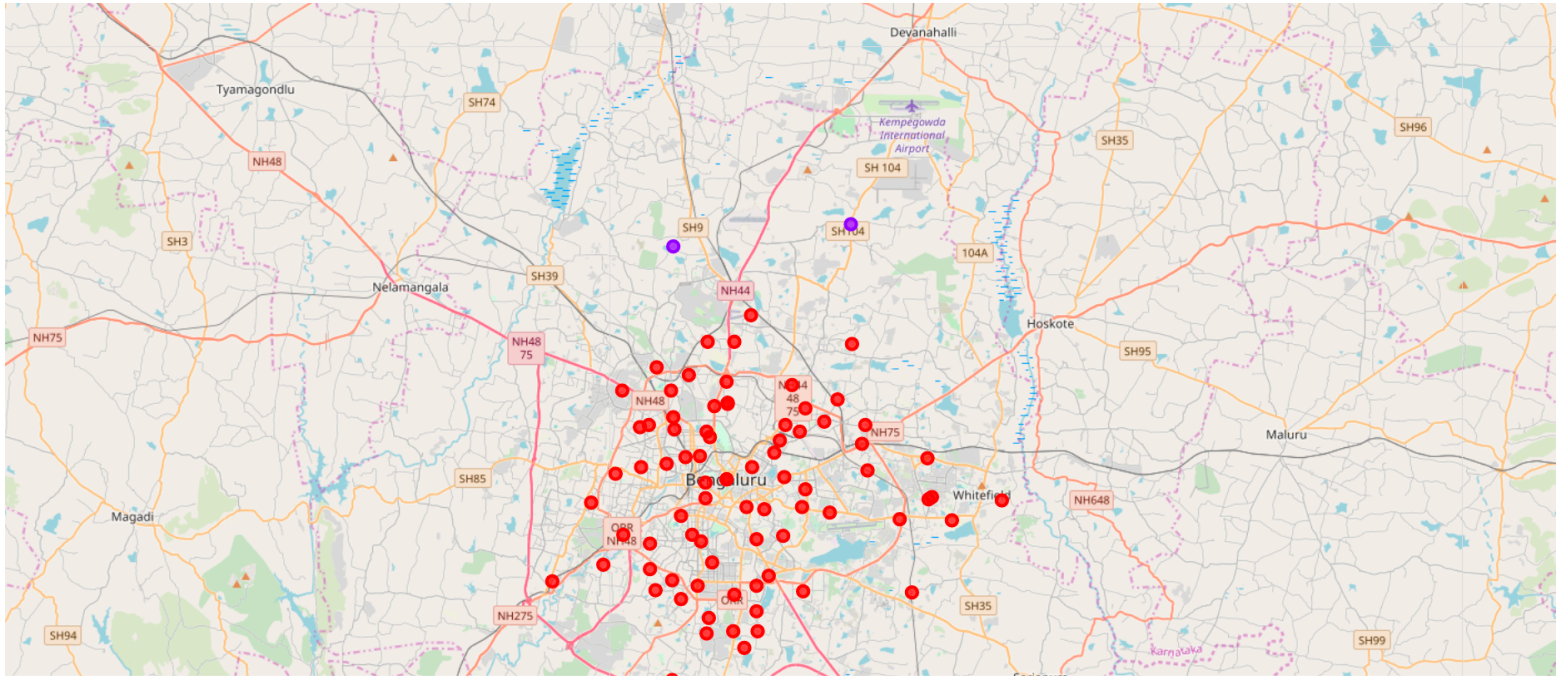


Methodology

K-means clustering

The Bangalore neighbourhoods and venue data is trained using K-means Clustering Algorithm to get the desired clusters to base the analysis on.

K-means was chosen since the variables(categories in this case) are large, the **K-Means** most of the times is computationally faster.



From the above clustering, we can see that majority of the clusters are falling under the same cluster (cluter1) with most common venue being eateries('Indian Restaurant', 'Kerala Restaurant' etc), apart from the remote areas like Bagalur, Ramagondanahalli etc.

Methodology

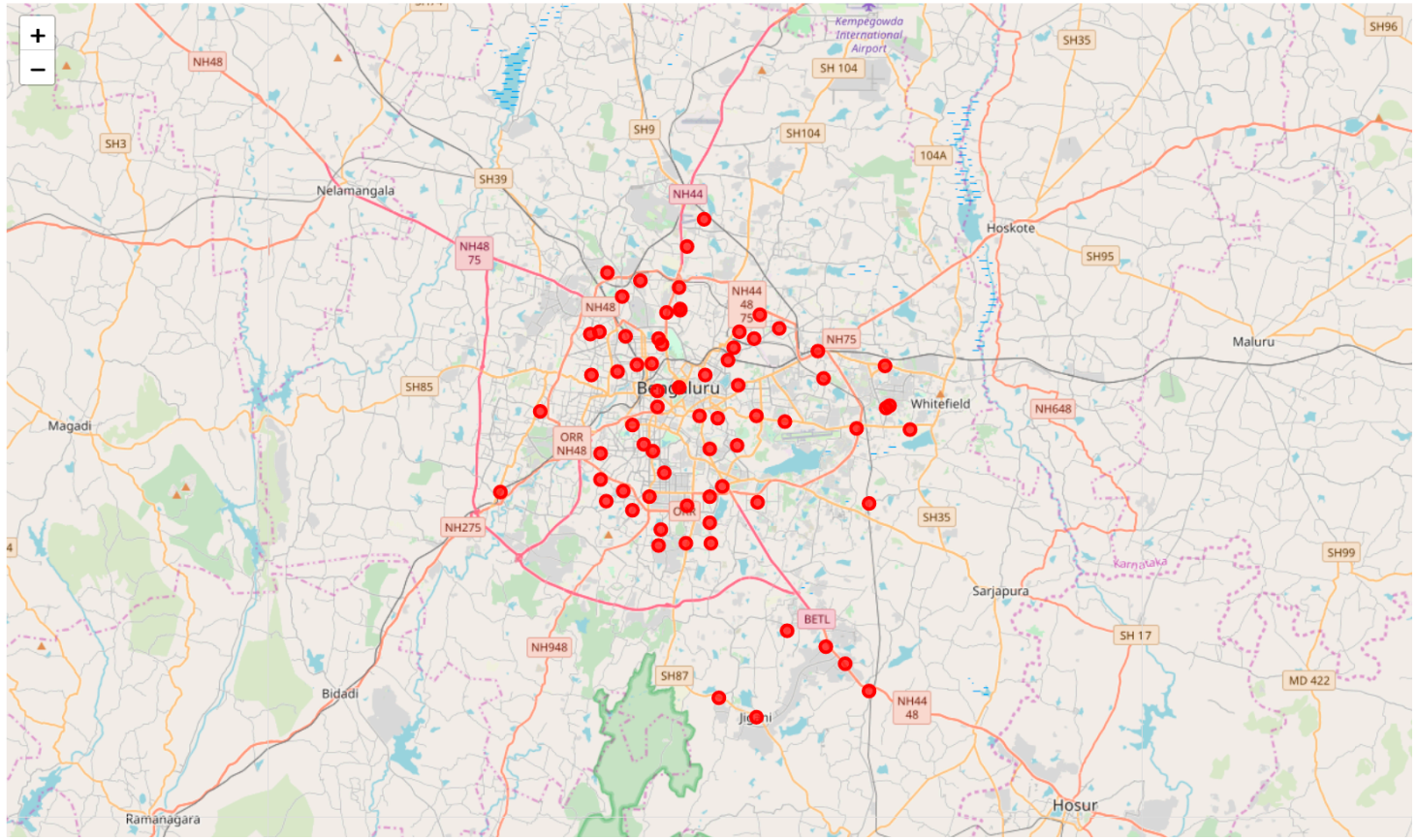
Narrowing Down

Subsequently we narrow down the cluster step-by-step to just a few points based on certain conditions like, the venue should have restaurants as common venue

Visualize locations of the clusters in bangalore which have most common venues as Restaurants

In [65]:
```
1 mapclust = visualize(blr_lat, blr_lng, kclusters,filterinfDataframe)
2 mapclust
```

Out[65]:



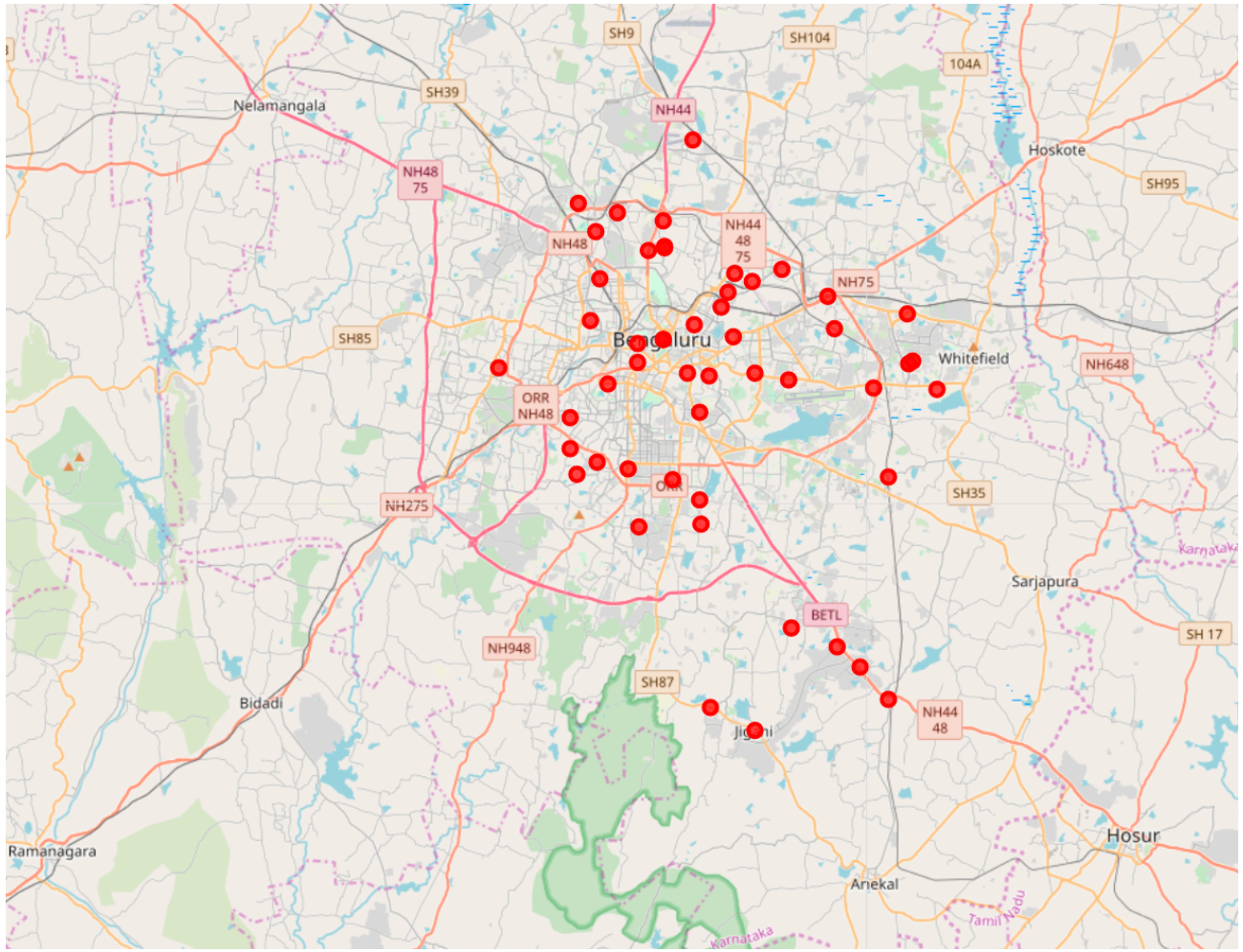
Methodology

Narrowing Down

Subsequently we narrow down the cluster step-by-step to just a few points based on certain conditions like, the venue should have restaurants *but not many ice cream parlours*

bourhoods with Indian Restaurants BUT which do not have ice Cream shops ¶

```
visualize(blr_lat, blr_lng, kclusters,filterinfDataframe_rest_ice)
```



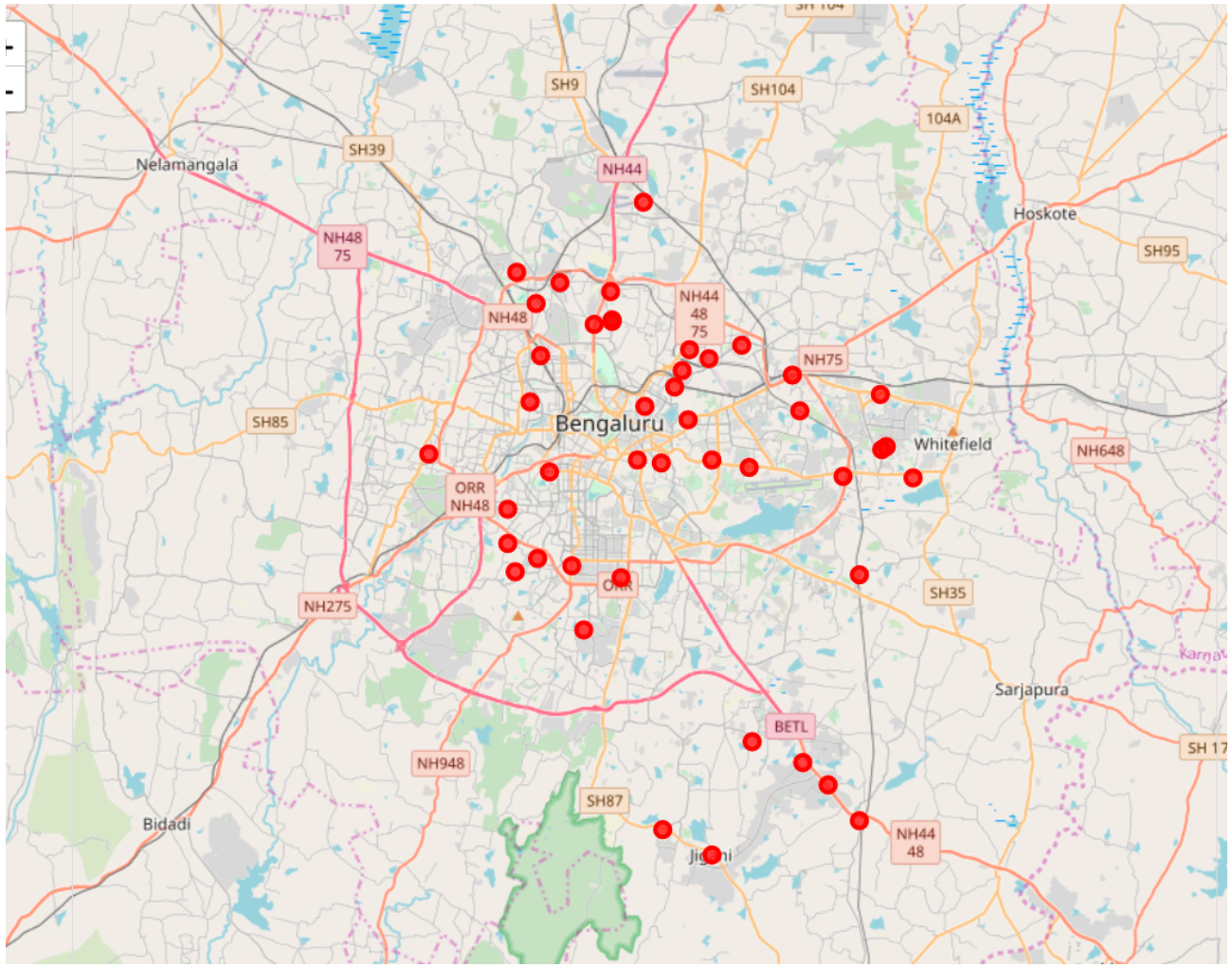
Methodology

Narrowing Down

Subsequently we narrow down the cluster step-by-step to just a few points based on certain conditions like, the venue should have restaurants but not many ice cream parlours *and desserts in its vicinity*

UALIZE neighbourhoods with Indian Restaurants BUT which do not have Ice Cream and Dessert

```
mapclust = visualize(blr_lat, blr_lng, kclusters,filterinfDataframe_rest_noiceDessert)
mapclust
```



Narrowing Down further, since we are looking for a neoghbourhood which families might visit, we might as look for areas with Clothing Store

```
dfRestCloth_noiceDessert = filterinfDataframe_rest_noiceDessert[(filterinfDataframe_rest_noiceDessert['1st Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['2nd Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['3rd Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['4th Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['5th Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['6th Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['7th Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['8th Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['9th Most Common Venue'] == 'Clothing Store') |
                                (filterinfDataframe_rest_noiceDessert['10th Most Common Venue'] == 'Clothing Store')]
print(dfRestCloth_noiceDessert.shape)
dfRestCloth_noiceDessert.to_csv("IndianRestaurants_clothing_noICSDessert.csv")
```

```
(9, 19)
```

Methodology

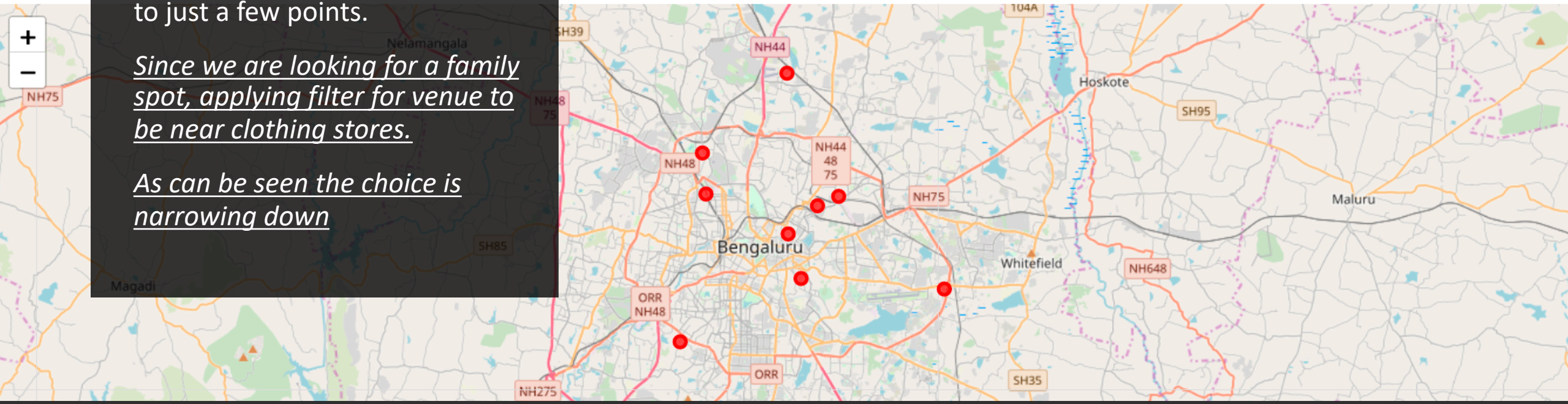
Narrowing Down

Narrow down further the cluster to just a few points.

Since we are looking for a family spot, applying filter for venue to be near clothing stores.

As can be seen the choice is narrowing down

```
mapclust = visualize(blr_lat, blr_lng, kclusters,dfRestCloth_noiceDessert)
mapclust
```



Assuming that Juice bars will also serve desserts and ice creams, eliminate the same from the target neighbourhood list

In [72]:
```
dfRestCloth_noiceDessertJuice = dfRestCloth_noiceDessert[(dfRestCloth_noiceDessert['1st Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['2nd Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['3rd Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['4th Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['5th Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['6th Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['7th Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['8th Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['9th Most Common Venue'] != 'Juice Bar') &
                                (dfRestCloth_noiceDessert['10th Most Common Venue'] != 'Juice Bar')]
print(dfRestCloth_noiceDessertJuice.shape)
dfRestCloth_noiceDessertJuice.to_csv("IndianRestaurants_clothing_noICSDessertJuice.csv")
```

(8, 19)

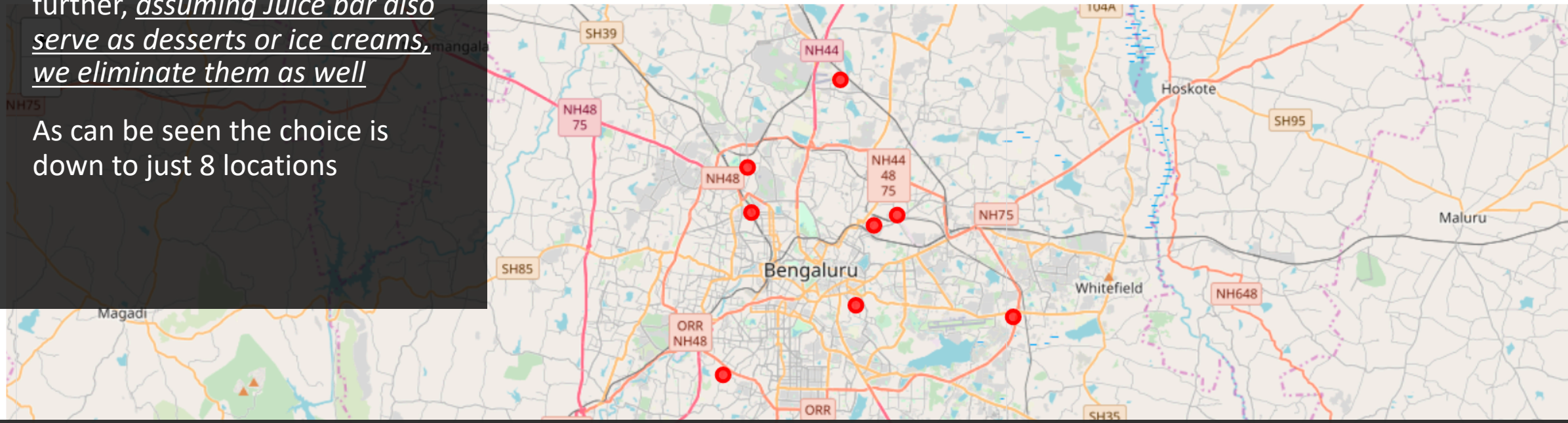
Methodology

In [73]:
```
mapclust = visualize(blr_lat, blr_lng, kclusters,dfRestCloth_noiceDessertJuice)
```

Narrowing Down the cluster further, *assuming Juice bar also serve as desserts or ice creams, we eliminate them as well*

As can be seen the choice is down to just 8 locations

Out[73]:

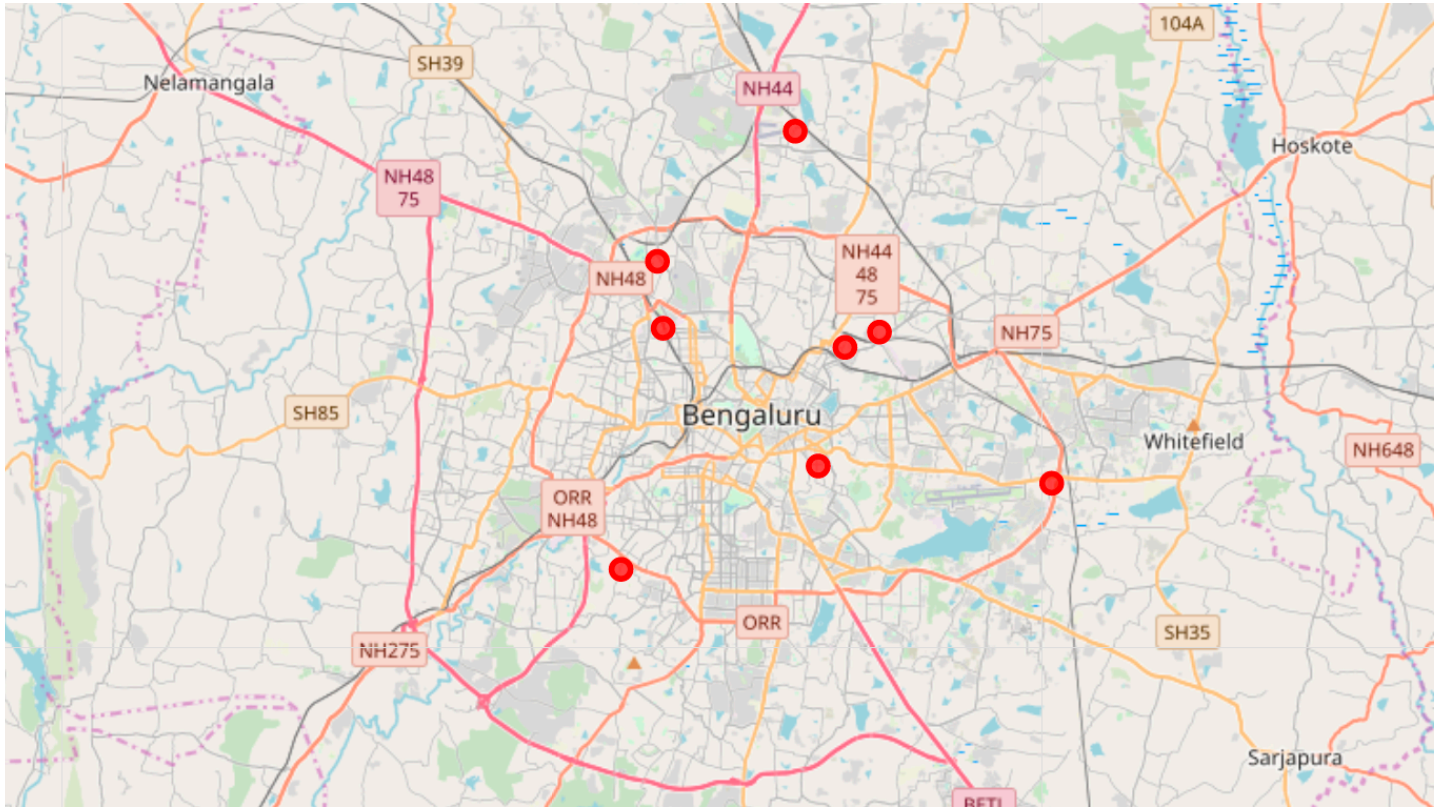


Results

To find an Ice cream storefront site accessible from main roads, with easy access, and with sufficient customer parking, we need to look for locations near businesses, like children's clothing stores or Indian restaurants.

To narrow down our search and visualize at each step helps the Stakeholders to make business decisions as per company strategy

On applying the filter conditions we can see that the target locations meeting the required conditions were narrowed down to just 8



Discussion

From the exploratory analysis, clustering and filtration process based on venue categories, we see that there are lots of neighbourhoods which have Restaurants among the top 10 venues.

However we found from the above exercise that localities where we have Restaurants and clothing Stores as most common venues, BUT which do not have ice cream shops, desserts or juice bars are very few.

This entire process provides interesting insights into the problem allowing stake holders to make easier decision making which are aligned to the business strategy

Conclusion

A possible recommendation could bee that - since *Jakkur* is emerging as an upcoming residential property market and not reached full potential, this could possibly be a good candidate for opening an Ice cream shop, and it also meets all the necessary conditions.

Reference

HTTPS://GITHUB.COM/RAGHUNATH-NAIR/COURSERA_CAPSTONE_W4