Bangkok's Metro (BTS, MRT, ARL) and Surrounding Venues

Arrart Kongtaln

June 7, 2020

Introduction

Bangkok is the capital city of Thailand and has a population of over 8 million people. The surrounding area, the Bangkok Metropolitan Region, is home to more than 14 million citizens, myself included. Thailand's Bangkok Metropolitan Region has **3** main urban rail transit systems, the **BTS** (**B**angkok mass **T**ransit **S**ystem or Skytrain), the **MRT** (**M**etropolitan **R**apid **T**ransit), and the **ARL** (**A**irport **R**ail **L**ink). There are currently 111 stations in operation (we will use 109 locations, as Siam station is listed on both BTS lines, and Tao Poon station is listed on two MRT lines):

- 49 BTS Skytrain stations (36 on the Sukhumvit Line and 13 on the Silom Line, with Siam being an interchange station between the two lines)
- 54 MRT stations (38 on the Blue Line and 16 on the Purple Line, with Tao Poon station being an interchange between the two lines)
- 8 Airport Rail Link stations

The combined average ridership of all 3 transit systems is more than 1.28 million people per day. When people look to rent or buy a residence in or near downtown Bangkok, places near a metro station are usually more attractive than those farther away from the Skytrain or MRT. The focus of this data science project is to look at the kinds of venues surrounding the metro stations and classify them based on the types of venues near a station the most, as well as taking into consideration the geographic location of the station. This report can be useful to people searching for a new home and wanting to know more about an area, or giving an answer to a more common question of deciding where to go to have lunch, or even city planners can use it as a brief summary of the current state of the '*Mass Rapid Transit Master Plan in Bangkok Metropolitan Region*' (or M-Map), among other possible creative applications.

Data

The final dataframe used for the data analysis contains 107 rows of Bangkok's metro stations' data (the stations 'Bang Pho' and 'Sirindhorn' were recently completed at the time, and did not yet yield location query results from Google's Geocoding API, and were not used). The list of all the metro (**BTS**, **MRT**, **ARL**) stations we used were retrieved from the *Wikipedia* page (as of May 28, 2020) [1].

We clean this data by removing the 'Photo' column, as it does not contain any data relevant to this project. We also handle cases of station duplication as shortly explained in the introduction (more information is provided in the following steps) with stations 'Tao Poon' and 'Siam', by formatting values to remove ambiguity in the columns 'Station Code', 'English Station Name', and 'Line(s)' for each station row so that there is only one value/name in a cell. The last bit of data cleaning is to remove extra information from some cells in the the 'English Station Name' column, where some names have parentheses in them, with the format of '(RTGS: *alternate_spelling*)'. RTGS is the acronym for the 'Royal Thai General System of Transcription', which is an alternate way to spell names in English. However, for our purposes, we need just the name with the common spelling (before the parentheses), and we also remove from the 'Phraek Sa' station row, the alternate spelling string: 'RTGS: Phraekkasa'.

[10]:

	Station Code	English Station Name	Thai Station Name	Line(s)	First Opened	Has Transfer To
0	N5	Ari	อารีย์	BTS	5 December 1999	NaN
1	E4	Asok	อโศก	BTS	5 December 1999	MRT : Sukhumvit
2	A3	Ban Thap Chang	บ้านทับช้าง	ARL	23 August 2010	NaN
3	E10	Bang Chak	บางจาก	BTS	12 August 2011	NaN
4	BL37	Bang Khae	บางแค	MRT	21 September 2019	NaN
5	BL04	Bang Khun Non	บางขุนนนท์	MRT	23 December 2019	MRT (planned) SRT : Charansanitwong (planned)
6	PP10	Bang Krasor	บางกระสอ	MRT	12 August 2016	NaN
7	E13	Bang Na	บางนา	BTS	12 August 2011	NaN

Fig. 1 : Cleaned and formatted data from the Wikipedia page.

Next, we add latitude and longitude coordinates (as well as the Plus Code if any exist) to each row, by looping through the whole list and creating custom Google Geocoding API queries for each row from their cell values. We then save the dataframe into a .csv file, so that we can use it for repeated testing and data classification, without having to call the Google API each time.

[22]:

```
for index, row in raw_df.iterrows():
    print("Querying: " +row[3] +" -'" +row[1] +"'")
    place_name = row[1].replace(" ", "+")
    search_url = "https://maps.googleapis.com/maps/api/geocode/json?address={}+{}".format(row[3], place_name)
    search_url += "&components=administrative_area:bangkok|country:TH&key=" +G_API_key
    search_req = requests.get(search_url)
    search_json = search_req.json()

    if (not search_json['results']) :
        longlats = {
                  "lat": "Not Available",
                  "lng": "Not Available"
                    }
        plus_code = "Not Available"
    else :
        longlats = search_json['results'][0]['geometry']['location']

        if 'plus_code' in search_json['results'][0].keys():
            plus_code = search_json['results'][0]['plus_code']['global_code']
        else :
            plus_code = ''

        if (len(plus_code) == 0):
            plus_code = "Not Available"

    insert2coord(row[0], plus_code, longlats)
    print("Dataframe updated.")

coord_df.head()
```

Fig. 2 : Python For loop containing Google API call to customize query.

[24]:

	Station Code	English Station Name	Thai Station Name	Line(s)	First Opened	Has Transfer To	Plus Code	Latitude	Longitude
0	N5	Ari	อารีย์	BTS	5 December 1999	NaN	7P52QGHV+VR	13.7797	100.545
1	E4	Asok	อโศก	BTS	5 December 1999	MRT : Sukhumvit	7P52PHP6+M8	13.7367	100.561
2	A3	Ban Thap Chang	บ้านทับช้าง	ARL	23 August 2010	NaN	7P52PMMQ+6F	13.7331	100.689
3	E10	Bang Chak	บางจาก	BTS	12 August 2011	NaN	7P52MJW4+P4	13.6968	100.605
4	BL37	Bang Khae	บางแค	MRT	21 September 2019	NaN	7P52PC65+8Q	13.7108	100.409
5	BL04	Bang Khun Non	บางขุนนนท์	MRT	23 December 2019	MRT (planned) SRT : Charansanitwong (planned)	7P52PGQ8+4V	13.7378	100.517
6	PP10	Bang Krasor	บางกระสอ	MRT	12 August 2016	NaN	Not Available	13.7601	100.566
7	E13	Bang Na	บางนา	BTS	12 August 2011	NaN	7P52MJ93+6V	13.6681	100.605

Fig. 3 : Dataframe with latitude and longitude coordinates from Google API query.

We can visualize the metro station data by plotting the coordinates of each station onto a map of the Bangkok area, where a station location is a drawn circle around the location point (with a radius of 500 meters), with the station type (BTS, MRT, ARL) being mapped to one of three colors :

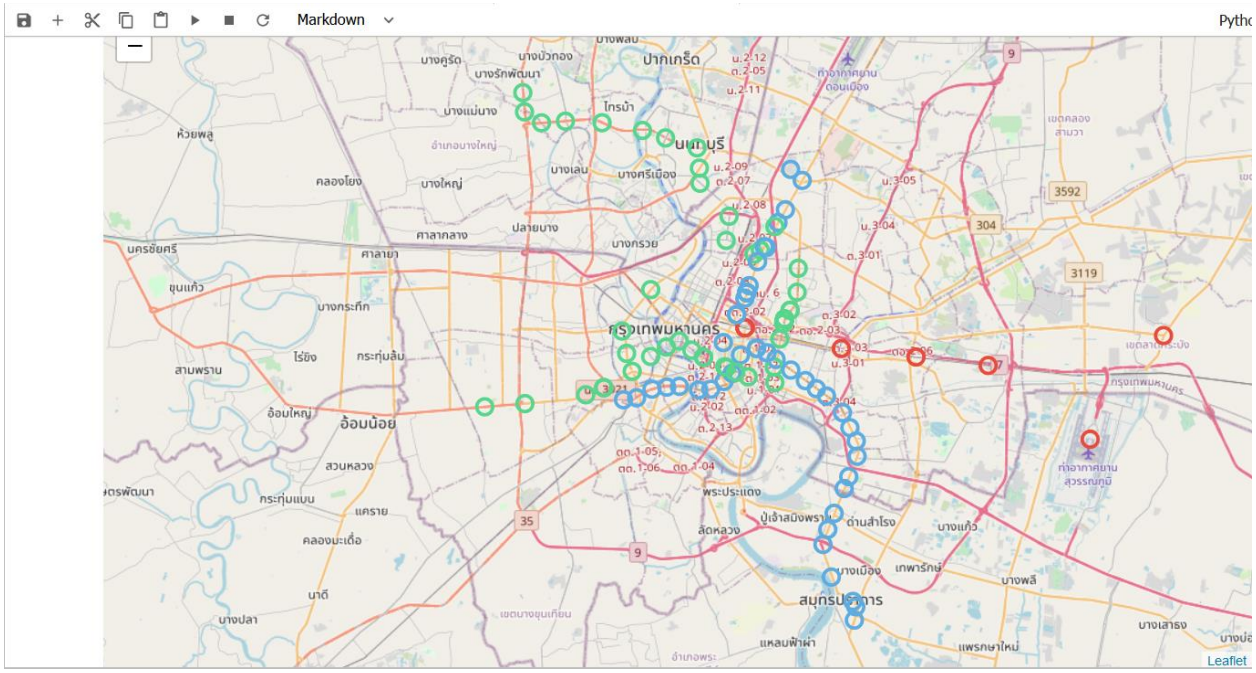

Fig. 4 : Map of the Bangkok Metropolitan Region showing metro station locations.

Using the coordinates we got from querying with the Google API, we can query locations and any nearby venues with the Foursquare Places API. We query each station's location and collect the number of venues, sorted by the top-level categories available from the Foursquare API. The Foursquare **explore** API contains a *categoryId* that you can use to query the number of venues of each category in a specific radius around a location's coordinates. The results have a *totalResults* value for the specified coordinates, radius and category.

[54]:
```
def get_venues_count(vlat, vlng, radius, categoryId):
    ll = "" +str(vlat) +"," +str(vlng)
    explore_url = 'https://api.foursquare.com/v2/venues/explore?client_id={}&client_secret={}&v={}&ll={}&radius={}&categoryId={}'.format(
                CLIENT_ID,
                CLIENT_SECRET,
                VERSION,
                ll,
                radius,
                categoryId)

    # make the GET request
    fs_req = requests.get(explore_url)

    fs_json = fs_req.json()

    total = 0

    #Handle empty response(s) (for locations that do not have venues near them)
    if (not fs_json['response']) :
        pass
    else :
        if (not fs_json['response']['totalResults']) :
            pass
        else :
            #print(fs_json)
            total = fs_json['response']['totalResults']

    return total
```

Fig. 5 : Python function containing Foursquare API call to customize query.

[58]:

	Station Code	English Station Name	Thai Station Name	Line(s)	First Opened	Has Transfer To	Plus Code	Latitude	Longitude	Arts & Entertainment	College & University	Event	Food	Nightlife Spot	Outdoors & Recreation	Professi & O Pl
0	N5	Ari	อารีย์	BTS	5 December 1999	NaN	7P52QGHV+VR	13.7796944	100.544625	8	10	0	90	22	17	
1	E4	Asok	อโศก	BTS	5 December 1999	MRT : Sukhumvit	7P52PHP6+M8	13.7367125	100.5608218	16	16	1	54	28	39	
2	A3	Ban Thap Chang	บ้านทับช้าง	ARL	23 August 2010	NaN	7P52PMMQ+6F	13.7330995	100.688657	1	1	0	4	1	3	
3	E10	Bang Chak	บางจาก	BTS	12 August 2011	NaN	7P52MJW4+P4	13.6967714	100.6053226	8	15	0	27	5	5	
4	BL37	Bang Khae	บางแค	MRT	21 September 2019	NaN	7P52PC65+8Q	13.710825	100.4094956	5	6	0	43	7	5	

Fig. 6 : Dataframe with total venues of each top-level type from Foursquare API query.

We can then classify each station based on the top categories of the total number of nearby venues around a station. Clusters of station locations and their surrounding areas can then be marked on a map of Bangkok and inform users about that part of the city.

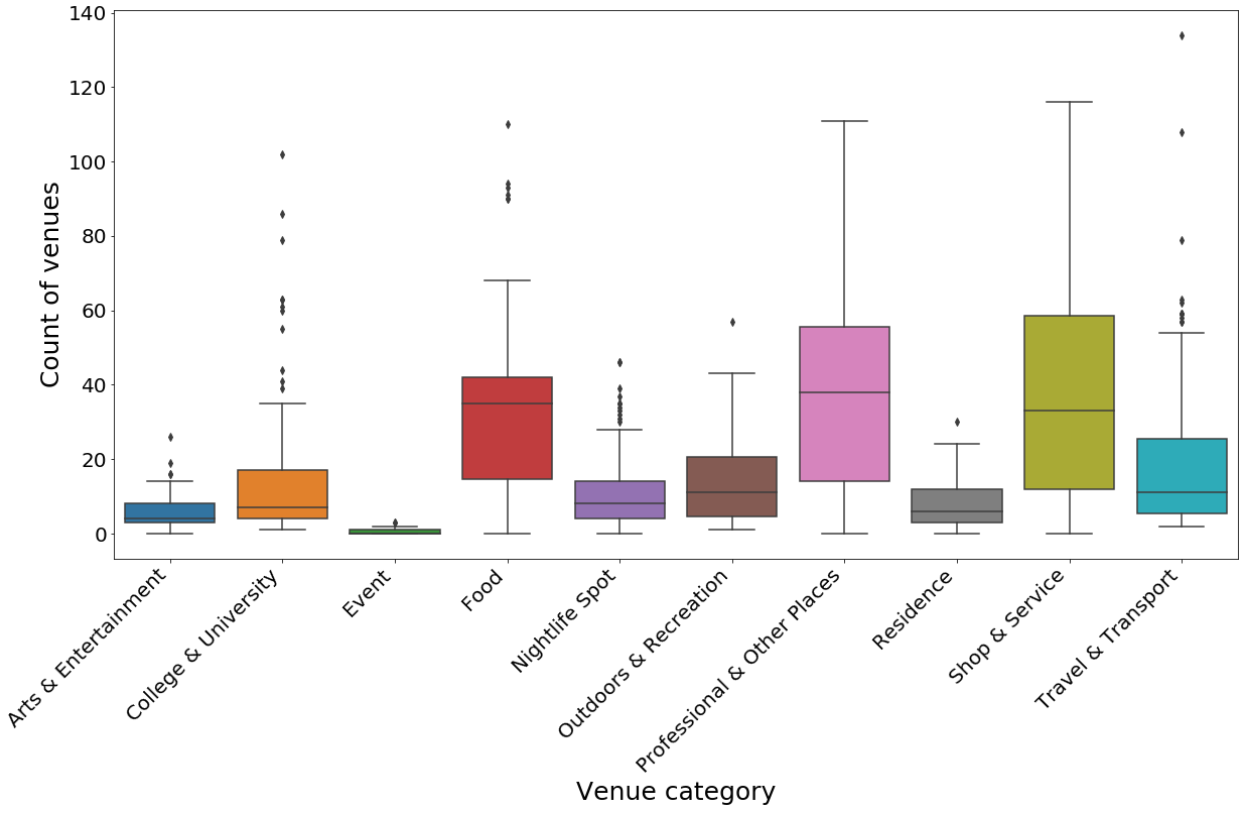

Fig. 7 : Boxplots of the raw total nearby venues around the metro stations.

It looks like the most frequent venue categories are **Shop&Service**, **Professional&OtherPlaces**, and **Food**. The venue category **Event** has very little data, so let's discard it from both the dataframe and the list of categories. Now the data preparation phase is nearly complete. Since the range of values among the categories is significant (noticeable from the outliers on the boxplots), we need to normalize the data to minimize the distortion differences in the ranges of values. Let's normalize the data using *MinMaxScaler* (scale from 0 to 1). This scales the data and provides an easy to interpret score at the same time.

Out[79]:

	Arts & Entertainment	College & University	Food	Nightlife Spot	Outdoors & Recreation	Professional & Other Places	Residence	Shop & Service	Travel & Transport
0	0.307692	0.089109	0.818182	0.478261	0.285714	0.603604	0.666667	0.456897	0.037879
1	0.615385	0.148515	0.490909	0.608696	0.678571	0.000000	0.533333	0.784483	0.583333
2	0.038462	0.000000	0.036364	0.021739	0.035714	0.027027	0.100000	0.025862	0.015152
3	0.307692	0.138614	0.245455	0.108696	0.071429	0.378378	0.300000	0.172414	0.015152
4	0.192308	0.049505	0.390909	0.152174	0.071429	0.225225	0.066667	0.534483	0.068182

Fig. 8 : The dataframe of the normalized total nearby venues around each metro station.

We can now visualize the scaled data :

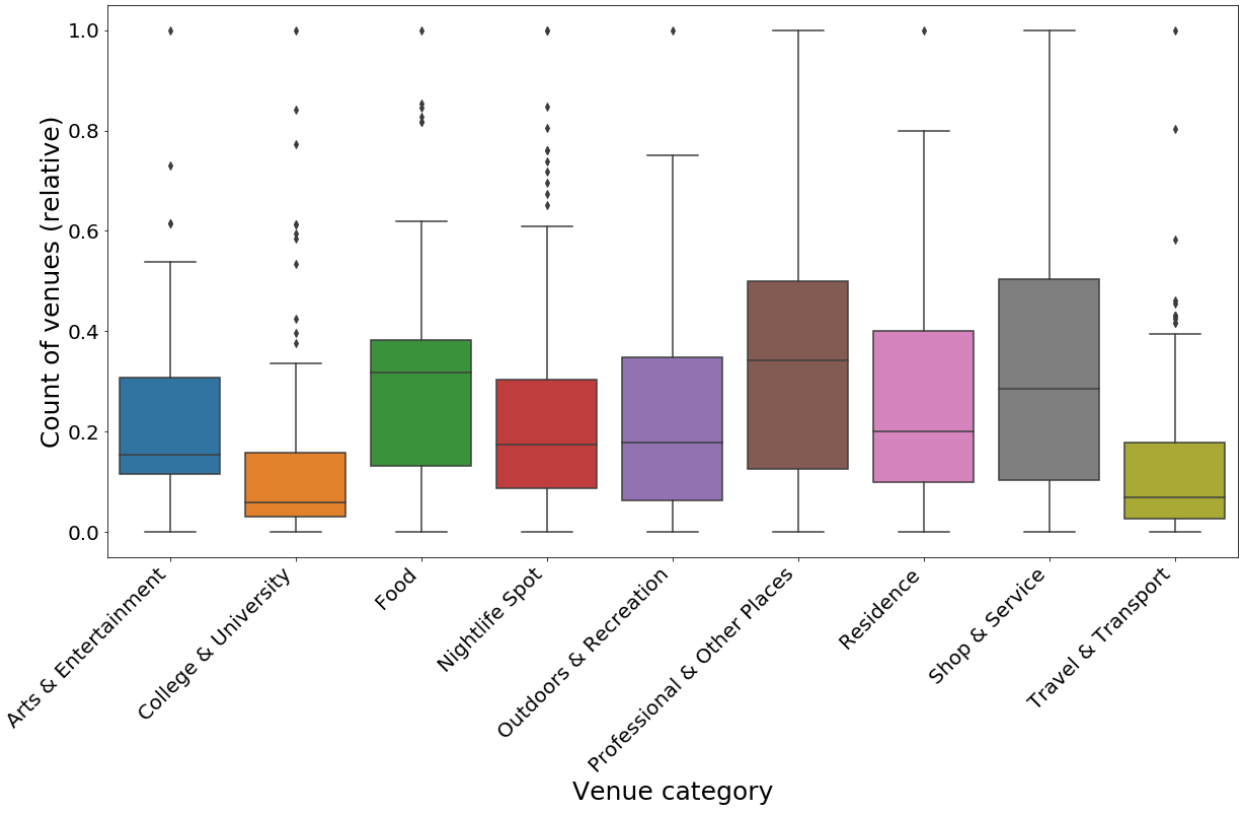

Fig. 9 : Boxplots of the normalized total nearby venues around the metro stations.

Methodology

Our classifier will use K-Means Clustering. Using different numbers of clusters, the initial results can be seen below :

- 2 clusters divided the area into just the downtown central area and the outer city surrounding area
- 3 clusters yield the most intuitive result consisting of high density areas, medium venue density areas, and low density suburbs
- 4 or more clusters are difficult to interpret, or need a more in-depth analysis to explain the cluster results

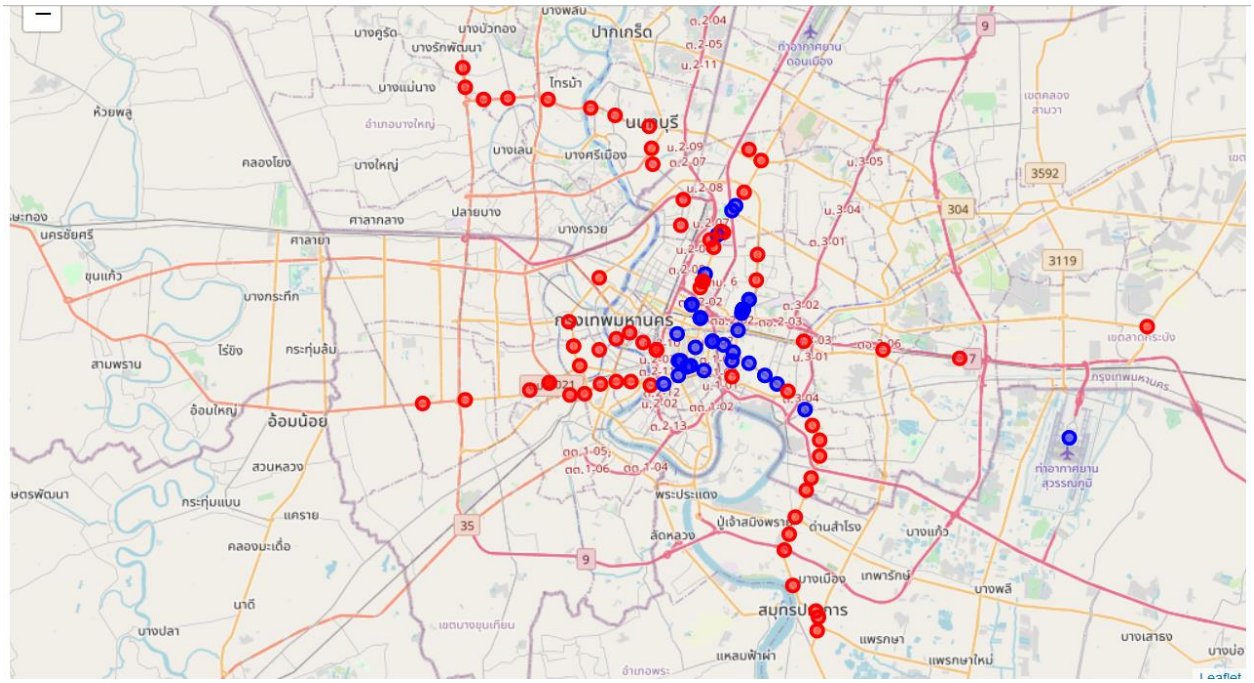

Fig. 10.1 : Image showing the K-Means clustering results from using a k value of 2.

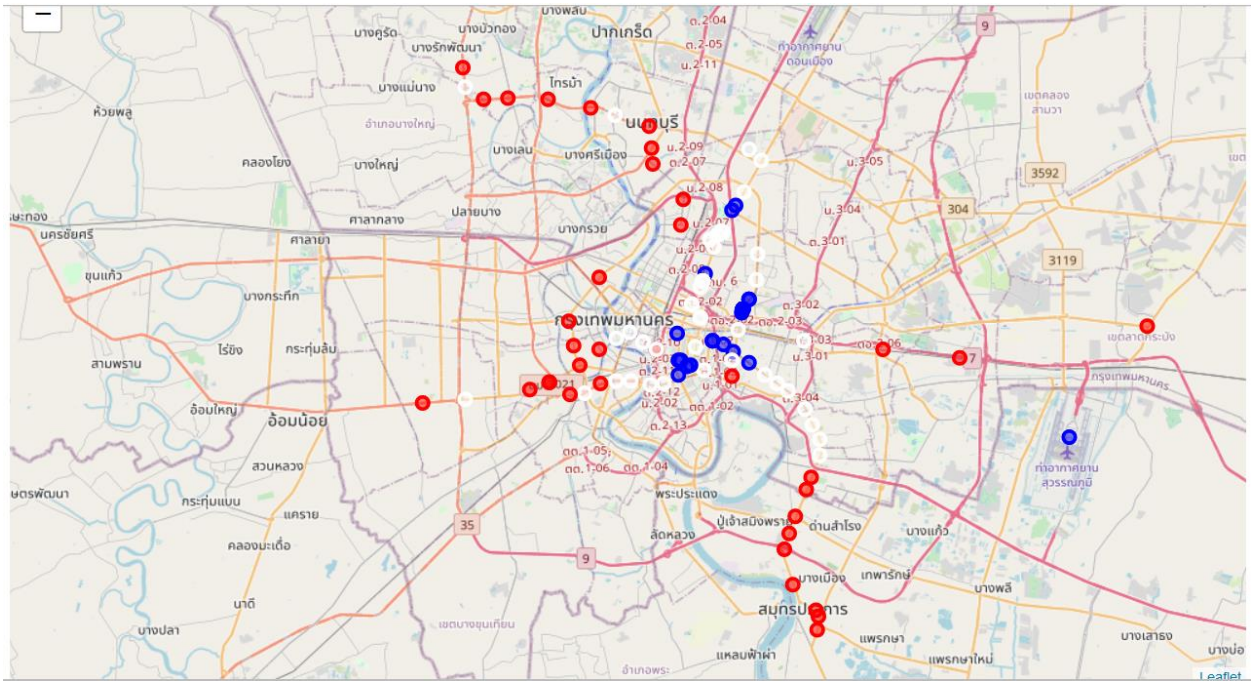

Fig. 10.2 : Image showing the K-Means clustering results from using a k value of 3.

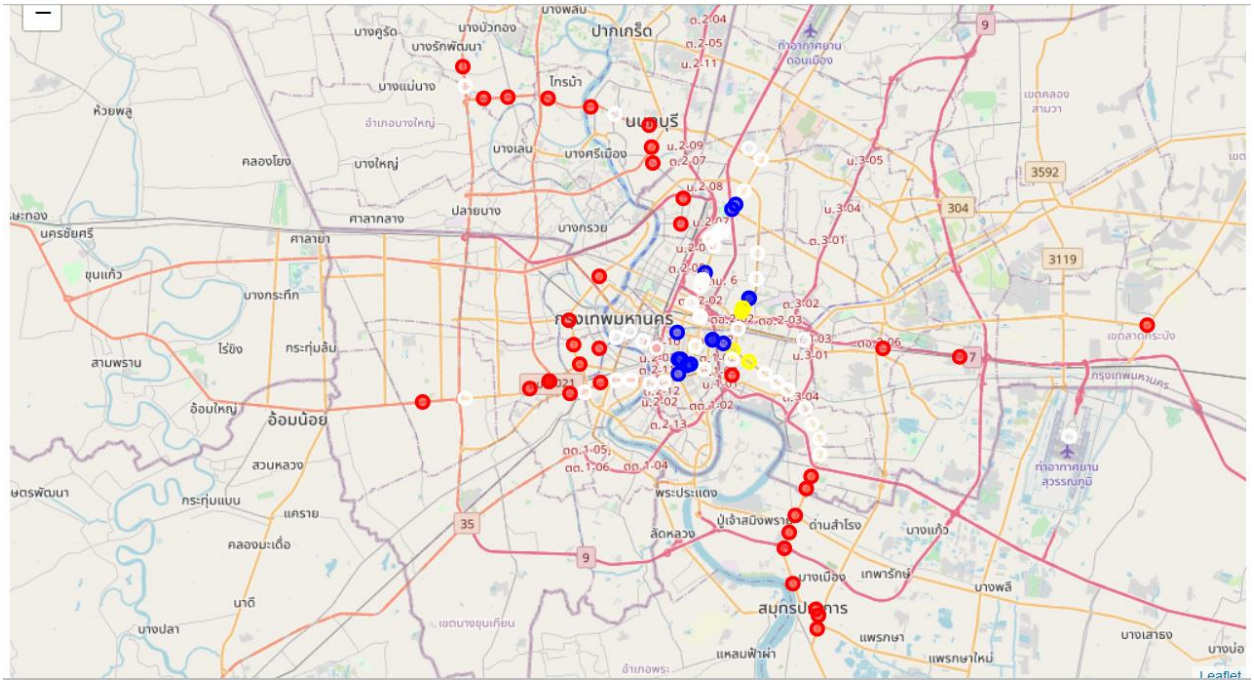

Fig. 10.3 : Image showing the K-Means clustering results from using a k value of 4.

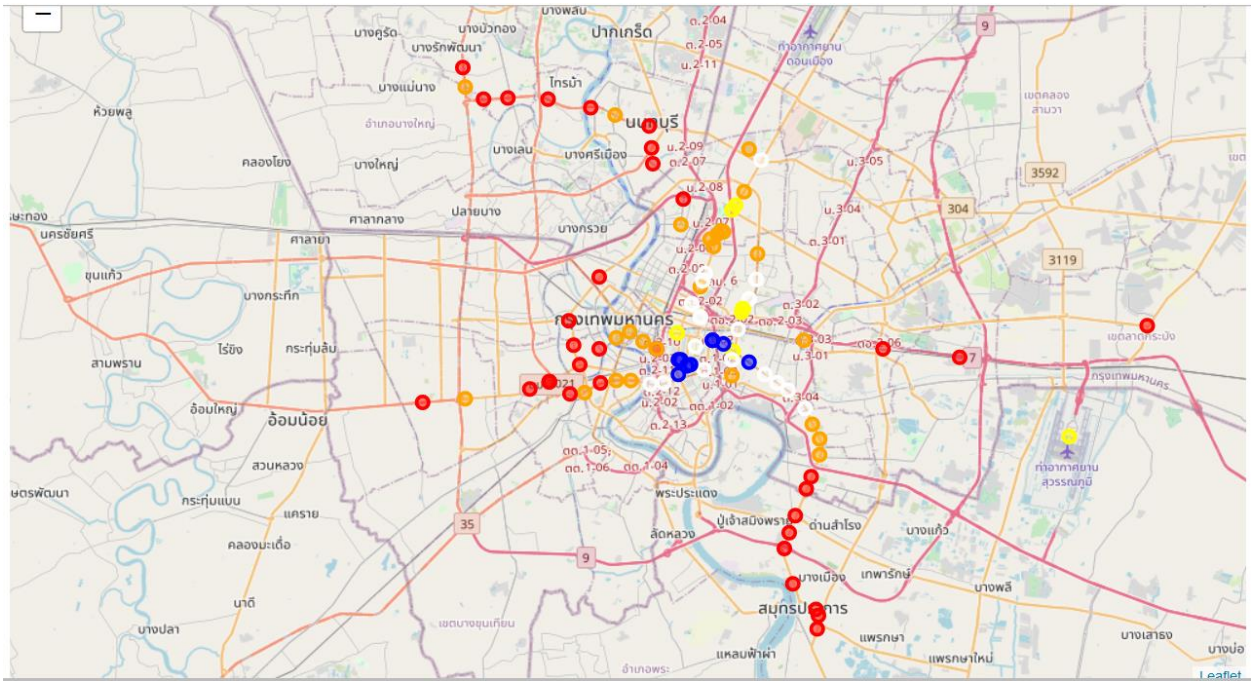

Fig. 10.4 : Image showing the K-Means clustering results from using a k value of 5.

For the scope of this class project, we will use 3 clusters in our analysis. We can use boxplots to view the classified clusters :

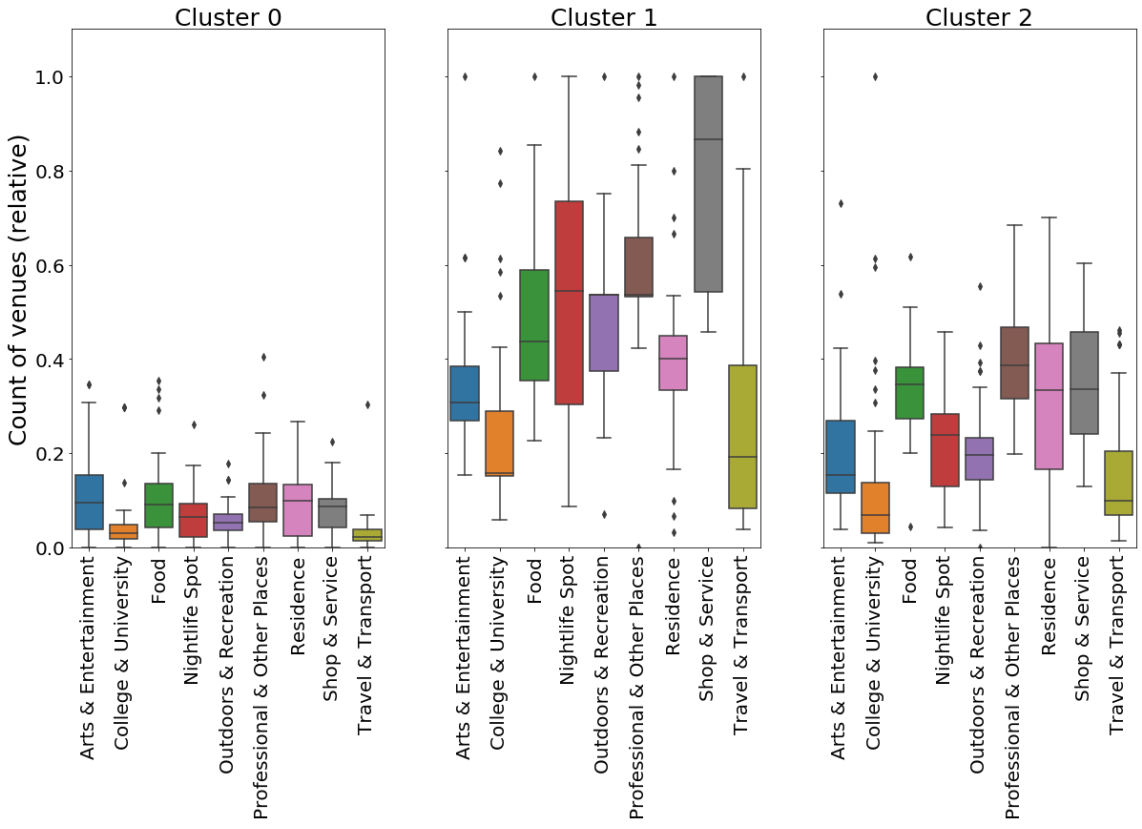

Fig. 11 : Boxplots of clusters 0 to 2 (3 clusters total).

We can draw the clustered locations onto a map of Bangkok. We will use the 3 colors of the Thai national flag for the 3 classified clusters. The clickable popup for each station will show the top 3 venue categories surrounding that area (full interactive map available at [2]).

Results

We can briefly summarize each of our classified clusters by looking at the boxplot showing the normalized values for the venues nearby each group of stations :

- Cluster 0 (Red) on average has the least number of venues near its stations, and appears as the lowest density area
- Cluster 1 (Blue) has the highest number of venues nearby, especially for Shop&Service, Nightlife Spot, and Travel&Transport
- Cluster 2 (White) has the overall highest number of Residence venues, and is between the other 2 clusters in nearby venue density

After coloring and plotting most of the stations (some locations did not render) on a map of the Bangkok Metropolitan Region, we can see that :

- Cluster 1 (Blue) most likely has the highest number of people passing by and creating venues and check-ins, as they are in the densely populated areas of the city (offices and department stores i.e. Si Lom station), or in the case of Suvarnabhumi International Airport, a very high volume of daily traffic as a transport hub
- Cluster 0 (Red) marks stations that are not in areas as developed as in the other 2 clusters, with the average for the Residence category being the highest among all the venue types located nearby
- Cluster 2 (White) seems to mark stations where there are less nearby venues than those stations in blue, but still quite a high number than when compared to the red stations

Some important stations (i.e. BTS Siam station, BTS & ARL Phaya Thai station) could not be drawn on the map when using the Folium library, however, the final map is still quite informative as stations nearby were rendered and appear to be correctly classified, so for our project's scope (looking at the area as a whole) this is still deemed quite satisfactory.

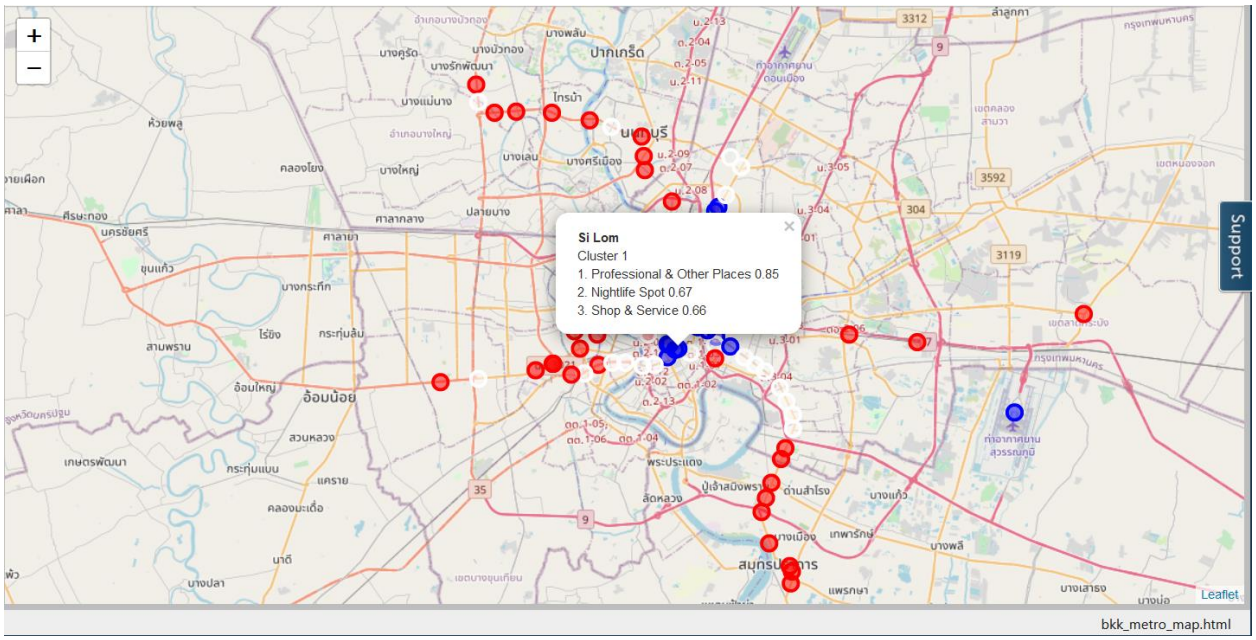

Fig. 12 : Map of a Cluster 1 (blue) station selected, showing the popup of Si Lom.

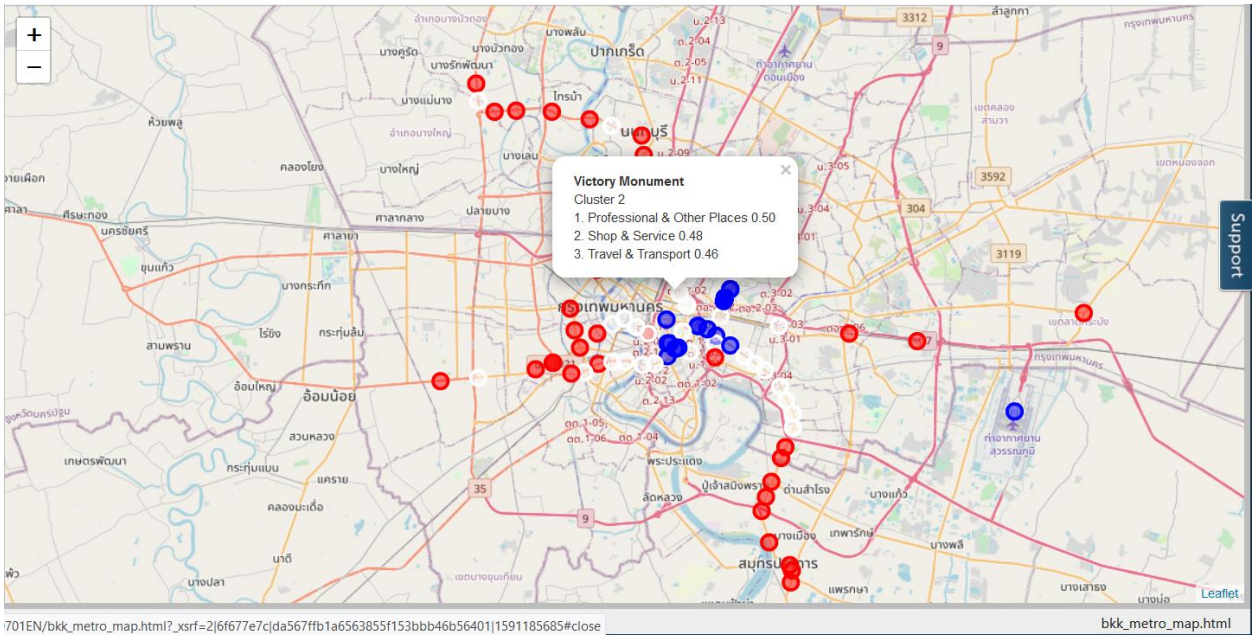

Fig. 13 : Map of a Cluster 2 (white) station selected, showing the popup of Victory Monument.

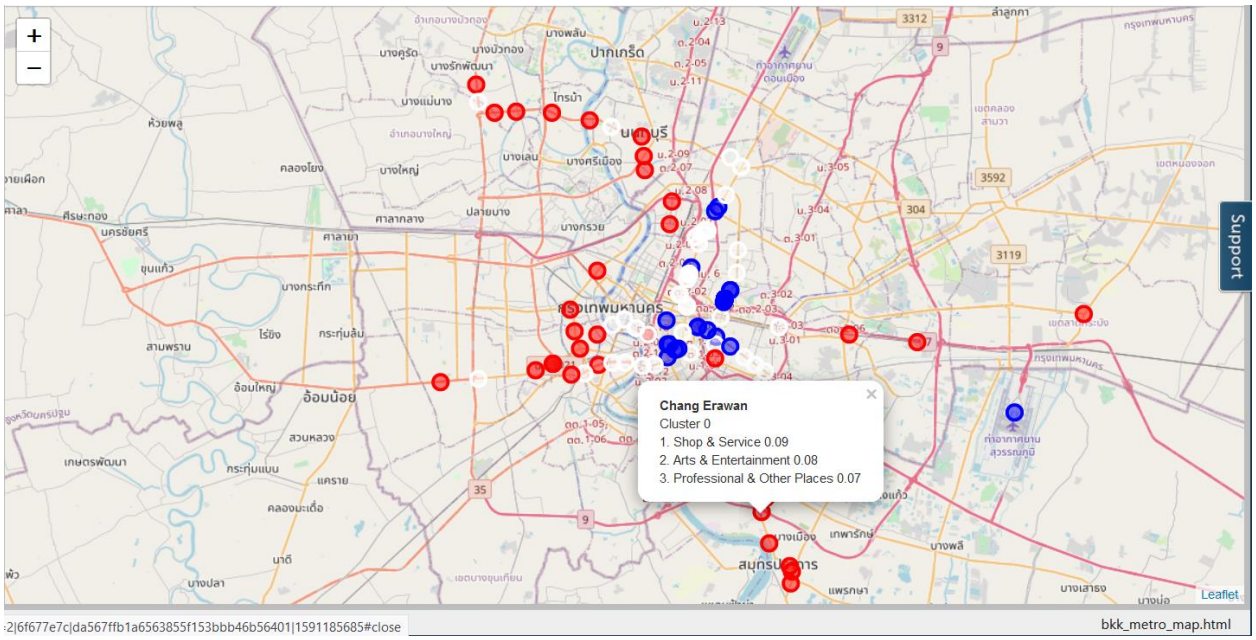

Fig. 14 : Map of a Cluster 0 (red) station selected, showing the popup of Chang Erawan.

Discussion

There are some factors to consider when analyzing the results of this data science project. First, using the Foursquare database to get the number of venues around each station can make our results a bit biased towards the Food and Travel&Transport categories, as these 2 types of locations are the most commented and checked-into places (see the paper at [3] for more details). The significance of a location or building also is not shown, so some key landmarks or important areas might not be highlighted. However, with the main theme being density and having users being able to click and reveal the top 3 categories of each rendered location on the map, we were able to answer the questions and challenges asked in the beginning of the project. Users can use the interactive map of Bangkok's Metro to find out more about the surroundings of each station by clicking on a circled area, and a popup will inform them about the top 3 types of venues around the station. By familiarizing themselves with the color scheme of the map (blue for high-density, white for medium-density, red for sparsely-dense areas), users can view the overall status of the Bangkok Metropolitan Region.

Conclusion

We have shown how to use the Google Geocoding API, the Foursquare Places API, and the Python Folium library to retrieve the locations and nearby number of venues around each of Bangkok's metro (BTS, MRT, ARL) stations, and plot most of them onto an interactive map of the Bangkok Metropolitan Region. The data collected can be useful to others in the future in other areas of research, especially if combined with more data from other sources, such as social media feeds or census data.

References

GitHub link : https://github.com/akongtaln/data-science-capstone

[1] Wikipedia page https://en.wikipedia.org/wiki/List_of_rapid_transit_stations_in_Bangkok

[2] Interactive map https://akongtaln.github.io/bkk_map/bkk_metro_map.html

[3] Foursquare paper https://www.researchgate.net/publication/261060627_Exploring_venue_popularity_in_Foursquare