Network and Algorithms Final Project
Providing food supply to network of restaurants

Rahim Sharifov, Nihad Shukur , Elmar Hajizade

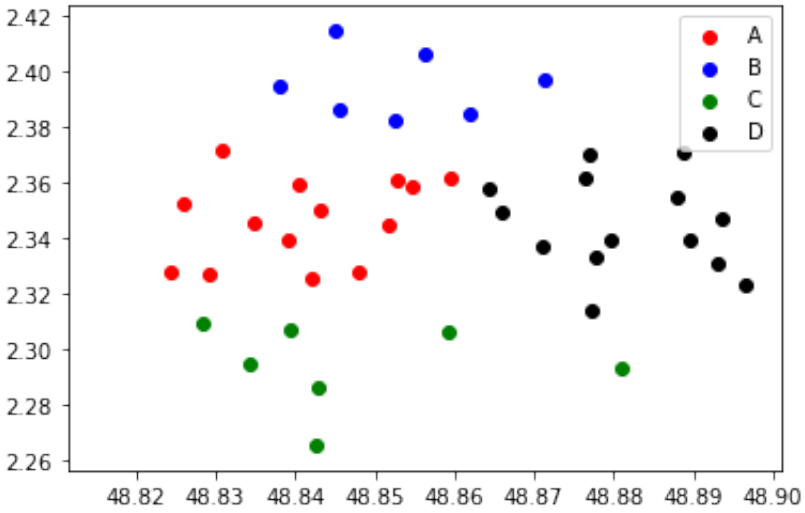
January 2020

Objective

In every city there are some warehouses of fruits, products and etc. Restaurants or supermarkets order some daily,weekly needs from these warehouses and there is some delivery service companies who delivers these orders from warehouses to supermarkets or restaurants. Each company has some limited cars for shipping. The company divide restaurants addresses between cars. But most of time they do it due to their logic or feelings. In this project we will analyze and try to find best way for delivery services. We will consider dividing process of restaurants between cars. Then create a best route for cars. What we mean by saying best route? Shortest path? No! Because shortest way does not mean best way. Maybe shortest way have high traffic. As we know if there is 10 restaurants, there are 10^8 route for delivery. Driver choose his path according to his logic or feelings. But bad(long or high traffic) route means more cost(gasoline and other costs). So in this project we also will consider traffic and distance between restaurants and try to find the best route and minimize cost of delivery services.

Analysing Data

In this project, we will work with Paris restaurants data. Data have 42 records. Initially we have zip-codes, restaurant names, addresses, city names, longitude and latitude coordinates of restaurants and type of restaurant. But except Restaurant names, longitude and latitude coordinates we don't need other columns so we drop them. As we mentioned above, each deliver service company have some cars for delivery. In this project we assume have 4 cars. Now we should divide 42 restaurants between 4 cars. Thanks to sklearn python library help us find best way for dividing restaurants between cars. We will use sklearn library in order to use K-Means machine learning algorithm to find best clusters of 42 restaurants.



Each color represent addresses of each car as mentioned right-above . As a result of K-Means: car A , car B , car C and car D responsible deliver 14,7,7,14 addresses with respectively. After applying K-Means algorithm y_kmeans variable contains list of 0,1,2,3 which represent addresses of A, B, C , D respectively, but we need a data frame for each car which contains the addresses of each car responsible. So we create 4 data frame and name them car_A, car_B, car_C, car_D. And as mentioned below append each data point into appropriate dataframe.

```
j=0
for i in y_kmeans:
  if i==0:
    car_A=car_A.append(df.iloc[j,:])
  if i==1:
    car_B=car_B.append(df.iloc[j,:])
  if i==2:
    car_C=car_C.append(df.iloc[j,:])
  if i==3:
    car_D=car_D.append(df.iloc[j,:])
  j+=1
```

As a next step we create a dictionary which each restaurant has own longitude and latitude coordinates. Then we multiply each coordinate by 1000. Because initial coordinates took from maps so they are very close each other and it's hard to see edges and nodes in visualization. Now we have 4 independent dataframes and from this point we don't need longitude and latitude coordinates in our dataframes so we create simple Delete_X_and_Y_coors which take dataframe as input, drop 2 mentioned columns and return updated dataframe. When we

create these 4 new dataframes we also take their indexes. But indexes are not sorted and they may be problem for us, so in order to reindex our dataframes we create another simple reIndexColumns which takes dataframe as input reindex datframe and return it. The next function is MakeCompleteGraph. Actually there are ways from restaurant to any restaurant in a map. So we decided create complete graph where every node(restaurants) have edge others. So this function responsible to create this dataframe. Photo mentioned below is a print of car_A dataframe show each restaurant have path to others.

```
         Names                dest
0    AVE MARIA               MAINE
1    AVE MARIA            BEAUNIER
2    AVE MARIA          MOUFFETARD
3    AVE MARIA        LES ARTISTES
4    AVE MARIA    MADELEINE BEJART
..         ...                 ...
177    CHARCOT            TOLBIAC
178    CHARCOT       ANDRE MALRAUX
179    CHARCOT           BOUTEBRIE
180    CHARCOT          PORT ROYAL
181    CHARCOT  JARDIN DES PLANTES

[182 rows x 2 columns]
```

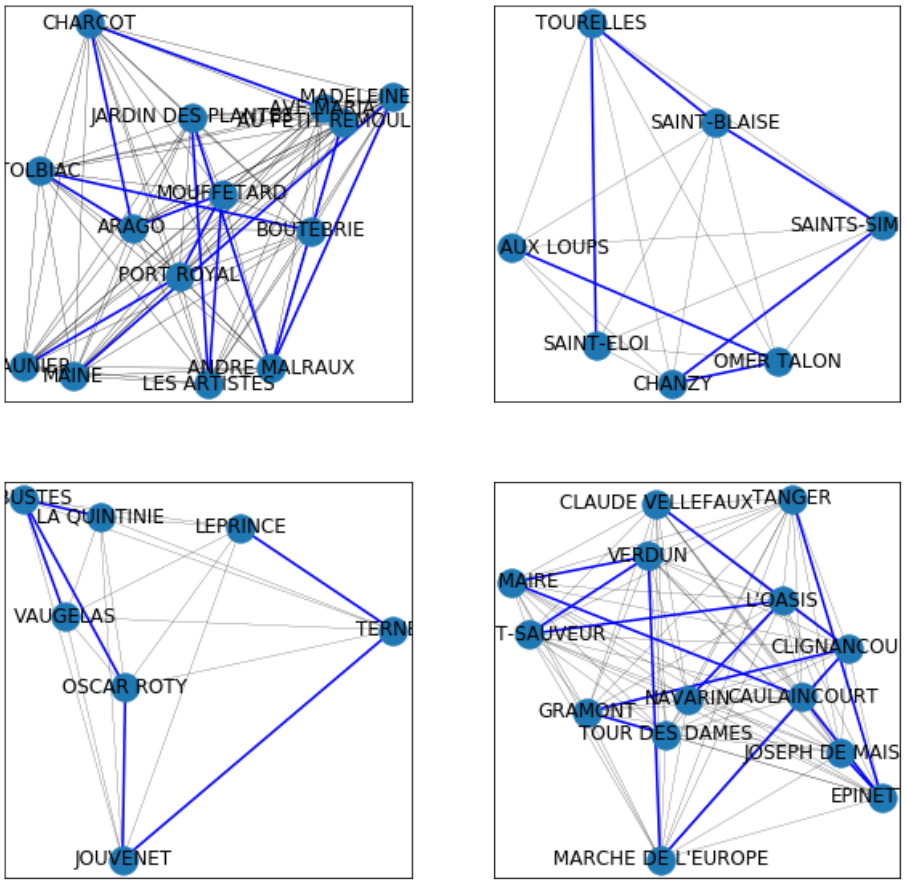
Unfortunately we could not find a dataset with traffic scale in ways and distance between them so we will generate them pseudo-randomly. Generate_distance_and_traffic is responsible to generate distance (between 100-500) and traffic scale(between 0-10) for each record. How we implement minimum spanning tree algorithm? We know spanning tree algorithms works with weight. But what is our weights? We think weight should not be neither distance nor traffic scale. Because if we consider weight as traffic, the algorithm find for us route which has minimum traffic, but maybe with longest way and if we consider weight as distance it will found the shortest path but maybe with the highest traffic. So what to do? We generate new parameter which contains both traffic scale and distance. How? We multiply traffic scale by 20 and add it with distance. So this new parameter called weight. Thanks to Generate_Weight function for doing these.

Organizing Paths

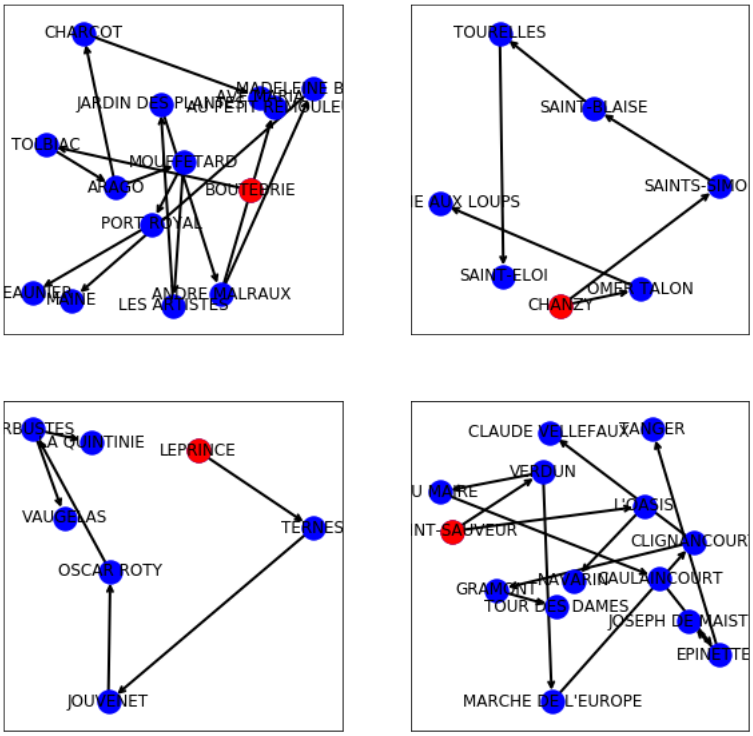
After splitting our network into four clusters, path that each truck will follow should be defined. Our way to do this is to find minimum spanning tree, as it is spans the given network with regard to keeping the weight the least. At to weight, it is the weighted sum of traffic scale and distance, according to their

importance in our network.

With calculated weight, using Prim's algorithm, we easily achieve graph of this tree. minimum_spanning_tree() method of Networkx, take name of algorithm,which is 'prim', and weight,on which our tree will be based, as arguments and returns the tree. As a result following spanning trees obtained.

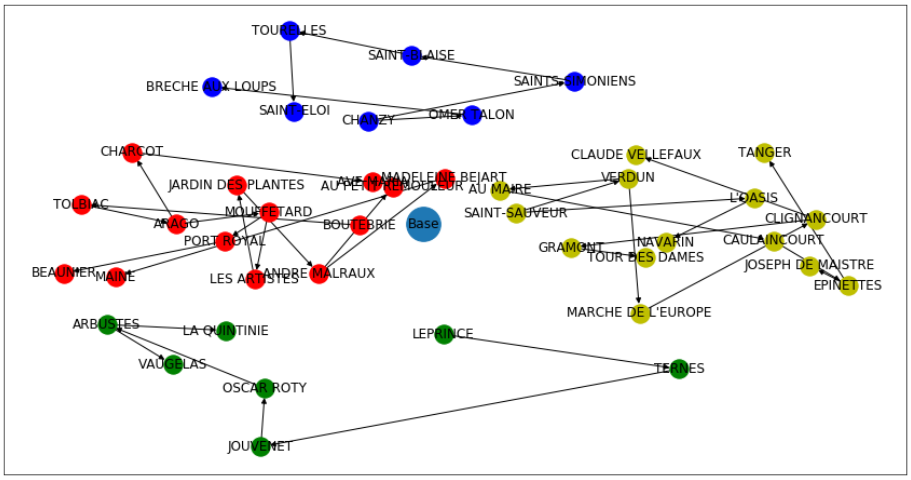


With obtained spanning trees, we use Depth First Search algorithm to traverse all the restaurants that we have. We take 'BOUTEBRIE', 'CHANZY', 'LEPRINCE', 'SAINT-SAUVEUR' as starting nodes. The reason we take these as starting nodes is that later on we find which restaurant should each truck start based on location of warehouse.Thus, using dfs_tree() algorithm of Networkx,which takes graph,and starting node as argument and returns directed graph which is following



As we have all directed graph of these clusters, now we back to objective of this project, to providing supplies to all these restaurants. We assume our warehouse, is in the middle of our network.

So, we find the coordinates of our base by summing x and y locations of all restaurants,and dividing to number of restaurants. We get average coordinate, which is middle of our graph.



As we have the location of our base, now we find 4 closest restaurants for each cluster to begin supplement. Using basic mathematical equation, finding distance between two points.

$$l = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

4 closest restaurants are 'BOUTERIE','LEPRINCE','CHANZY','SAINT-SAUVEUR'.

Conclusion

As a conclusion, our graph provides, optimal path that a warehouse can provide supply with 4 trucks. Final graph is