

UCLA ECE 232E S2021 Project #4

Graph Algorithms

Lin Fan - 505627503

Sunay Bhat - 905629072

Yi-chun Hung - 705428593

Tu Yu-Hsien - 405627283

Table of Contents

1. Stock market

1. Return correlation

Q1)

2. Constructing correlation graphs

Q2)

3. Minimum spanning tree (MST)

Q3)

4. Sector clustering in MST's

Q4)

5. Correlation graphs for weekly data

Q5)

2. Let's Help Santa!

2. Build Your Graph

Q6)

3. Traveling Salesman Problem

Q7)

Q8)

Q9)

Q10)

4. Analysing Traffic Flow

5. Estimate the Roads

Q11)

6. Calculate Road Traffic Flows

Q12)

7. Calculate Max Flow

Q13)

8. Prune Your Graph

Q14)

Q15)

9. Define Your Own Task

1. Stock market

1. Return correlation

Q1)

The return correlation coefficient is below as defined in the project prompt.

$$\rho_{ij} = \frac{\langle r_i(t)r_j(t) \rangle - \langle r_i(t) \rangle \langle r_j(t) \rangle}{\sqrt{(\langle r_i(t)^2 \rangle - \langle r_i(t) \rangle^2)(\langle r_j(t)^2 \rangle - \langle r_j(t) \rangle^2)}}$$

Like a standard correlation coefficient, **the upper and lower bounds are 1 and -1 respectively**. The derivation below shows how ρ_{ij} is **1 at most if $r_i(t)$ and $r_j(t)$ are perfectly positively correlated**, and **-1 at least if $r_i(t)$ and $r_j(t)$ are perfectly negatively correlated**. ρ_{ij} is **0 if there is no correlation** ie. $r_i(t)$ and $r_j(t)$ are orthogonal.

If $r_i(t)$ and $r_j(t)$ are correlated, then:

$$r_i(t) = \alpha r_j(t)$$

Thus:

$$\begin{aligned}\langle r_i(t)r_j(t) \rangle &= \alpha \langle r_j(t)^2 \rangle \\ \langle r_i(t) \rangle \langle r_j(t) \rangle &= \alpha \langle r_j(t) \rangle^2\end{aligned}$$

In Addition:

$$(\langle r_i(t)^2 \rangle - \langle r_i(t) \rangle^2) = (\alpha^2 \langle r_j(t)^2 \rangle - \alpha^2 \langle r_j(t) \rangle^2) = \alpha^2 (\langle r_j(t)^2 \rangle - \langle r_j(t) \rangle^2)$$

Thus the denominator of ρ_{ij} is:

$$\sqrt{(\langle r_i(t)^2 \rangle - \langle r_i(t) \rangle^2)(\langle r_j(t)^2 \rangle - \langle r_j(t) \rangle^2)} = \sqrt{\alpha^2 (\langle r_j(t)^2 \rangle - \langle r_j(t) \rangle^2)^2}$$

And the numerator of ρ_{ij} is:

$$\langle r_i(t)r_j(t) \rangle - \langle r_i(t) \rangle \langle r_j(t) \rangle = \alpha \langle r_j(t)^2 \rangle - \alpha \langle r_j(t) \rangle^2 = \alpha (\langle r_j(t)^2 \rangle - \langle r_j(t) \rangle^2)$$

And finally:

$$\rho_{ij} = \frac{\alpha (\langle r_j(t)^2 \rangle - \langle r_j(t) \rangle^2)}{\sqrt{\alpha^2 (\langle r_j(t)^2 \rangle - \langle r_j(t) \rangle^2)^2}} = \frac{\alpha}{\sqrt{\alpha^2}}$$

Meaning that if $\alpha > 0$, $\rho_{ij} = 1$ and if $\alpha < 0$, $\rho_{ij} = -1$

Log normalization is commonly used in stock market analysis because it **reduces the variance of time-series data**. There are multiple mathematical benefits of log normalization, but the most important is the log identity that allows **normally distributed compounding returns to be calculated simply**:

$$\begin{aligned} \log[1 + q_i(t)] &= \\ \log\left[\frac{p_1(t-1)}{p_1(t-1)} + \frac{(p_1(t) - p_1(t-1))}{p_1(t-1)}\right] &= \\ \log\left[\frac{p_1(t)}{p_1(t-1)}\right] &= \\ \log[p_1(t)] - \log[p_1(t-1)] \end{aligned}$$

More generally, “t-1” could be replaced with any range, and the **log normalized return of this time period is simply the difference in logs of the initial and final price of the time period**. This allows for much more convenient price analysis for compounded growth.

2. Constructing correlation graphs

Q2)

The edge weights are calculated from the return correlation coefficient of the two vertices. The equation below is “**shifting**” the edge weights such that the **strongest negative correlation** ($\rho_{ij} = -1$) is the **largest weight value** and the **strongest positive correlation** ($\rho_{ij} = 1$) produces the **lowest weight value**:

$$w_{ij} = \sqrt{2(1 - \rho_{ij})}$$

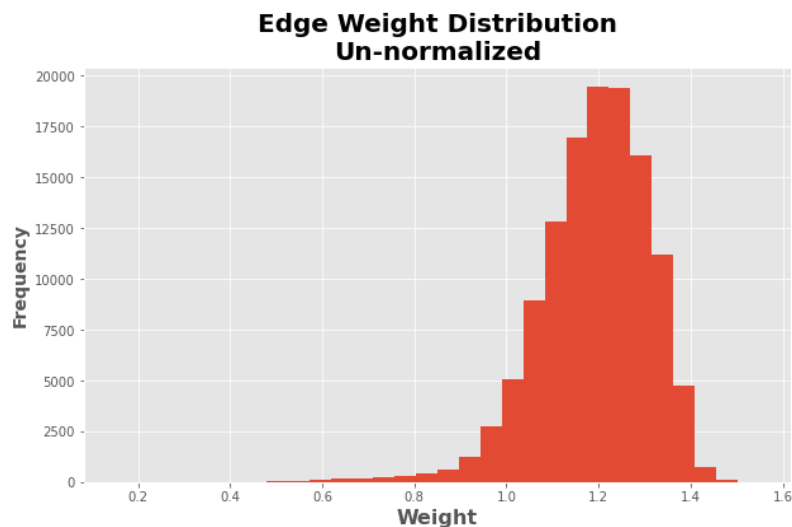


Figure 1: Stock correlation graph edge weight distributions

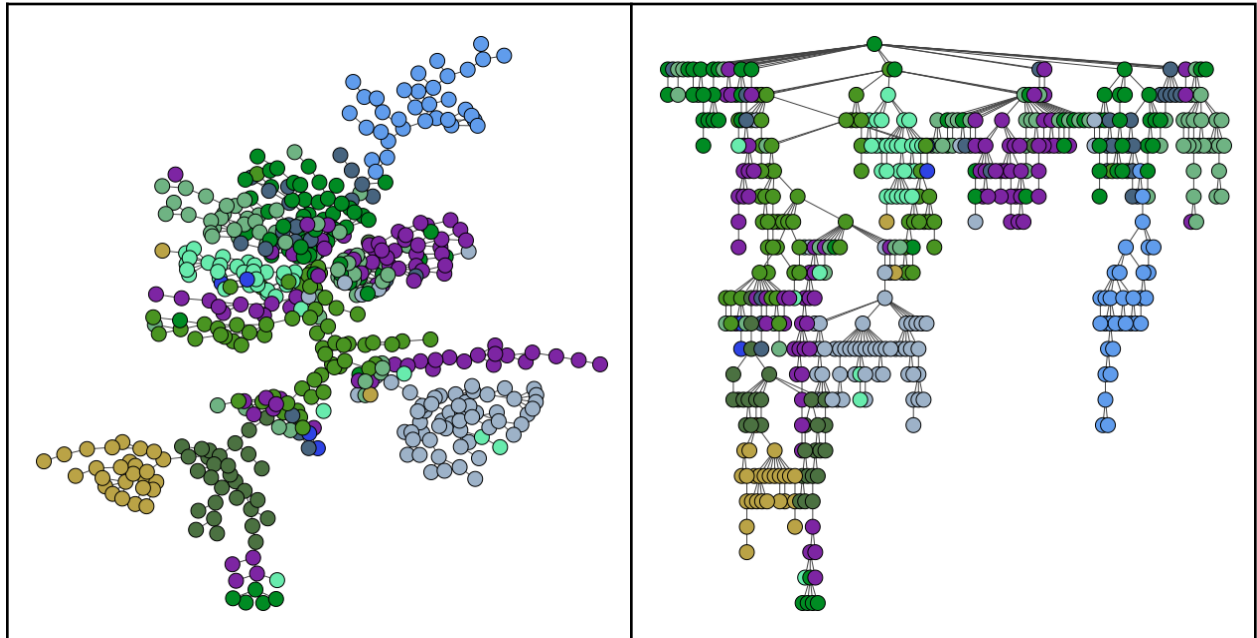
The plot above shows the distribution of edge weights in the correlation graph. The distribution is essentially **unimodal**, shaped somewhat like a **gamma distribution centered around a mean of 1.1-1.2**. **Due to the bounds on ρ_{ij} , the weights w_{ij} are also bounded** between 0 and 2, with 2 representing the highest negative correlation, $\sqrt{2}$ representing a zero correlation, and 0 representing a positive correlation. **Based on this, we can interpret the graph as there being far more ‘weak’ positive correlations with a small tail in the direction of some stronger positive correlations.**

3. Minimum spanning tree (MST)

Q3)

The **minimum spanning tree** is generated by finding the **tree that minimizes the cumulative edge weights**. Because of how our edge weight value is calculated, as stated in the previous problem, it is **maximizing the correlation representation of the graph** (or choosing the edge weights with the highest positive correlation while meeting the criteria of a tree). When we assign colors by sector the vertices, we can then visualize how highly correlated the sectors are. We should see “communities” or Vine clusters in the tree of the same color if the sectors are highly correlated.

We use two layouts below to help visualize any structure: the **“Kamada Kawai” (left)** and the **“Reingold Tilford” (right)** layouts. **There is certainly some structure, as colors are loosely clustered together.** But in both layouts we see **this is not close to strictly true**, as there is much overlap and many sectors appear to be loosely correlated as well. **This aligns with the intuition based on the distribution of edge weights where loose positive correlations seem to dominate, implying some weak correlation structure.**



4. Sector clustering in MST's

Q4)

In this question, we explore performance metrics for two methods of predicting the market sector of a stock. Both are summarized by an α value:

$$\alpha = \frac{1}{|V|} \sum_{v_i \in V} P(v_i \in S_i)$$

The first method utilizes the minimum spanning tree (MST) generated in the previous question. Q_i is the set of neighbors from the MST that share the same sector, while N_i is the set of all neighbors in the MST:

$$P(v_i \in S_i) = \frac{|Q_i|}{|N_i|}$$

The second method effectively considers the entire graph. S_i is the total number of stocks in the same sector, while V is the set of all stocks (vertices):

$$P(v_i \in S_i) = \frac{|S_i|}{|V|}$$

Note that if we take the **first method of the full graph**, these two values are equivalent. Since the full graph is fully connected:

$$Q_i = S_i, N_i = V$$

Thus,

$$\alpha_1 = \alpha_2$$

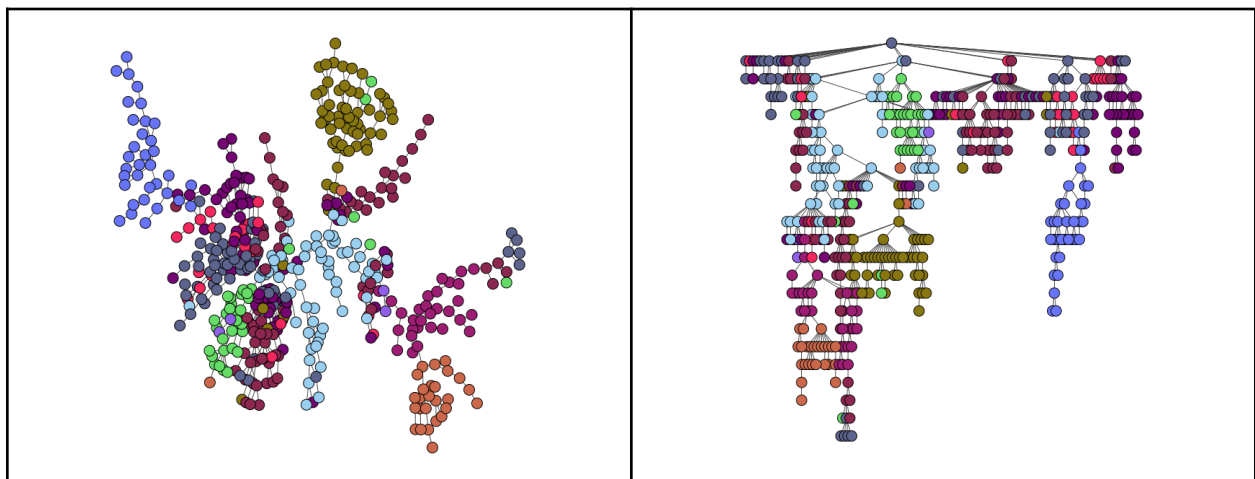
When comparing the two methods, we get the following results. **It is clear the first method is superior, since it considers the correlation via the MST structure. Since we have a large amount of weak positive correlations, this proves to be an effective way to predict the stock sector. On the other hand, the base probability (second method), does not use any correlation information to improve its prediction.**

α_1	α_2
0.8981	0.1142

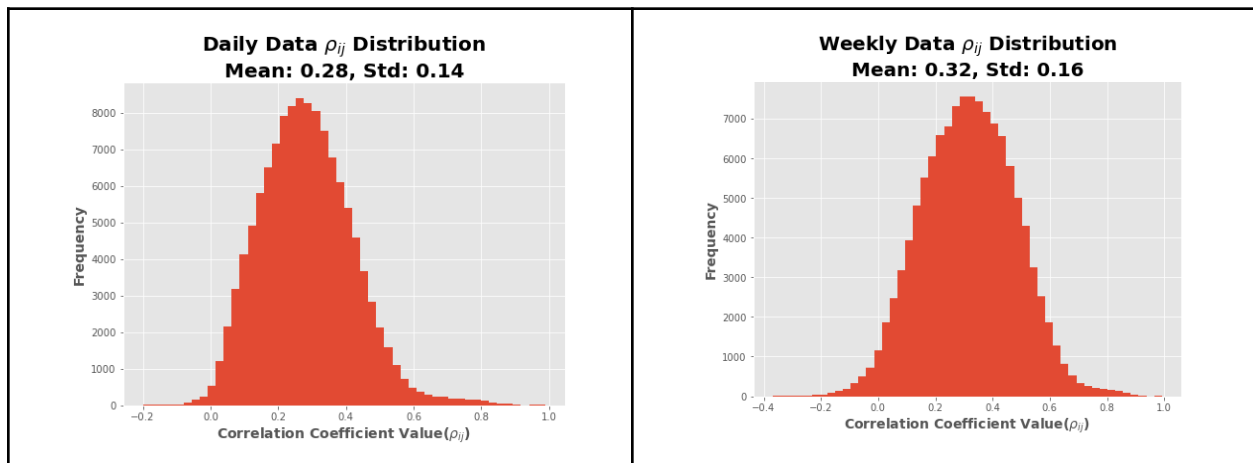
5. Correlation graphs for weekly data

Q5)

The MST's for the **weekly data is generated**, again with the two layouts below to help visualize any structure: the “**Kamada Kawai**” (left) and the “**Reingold Tilford**” (right) layouts. It is nearly very difficult to distinguish any obvious structural differences from this graph alone. **It does appear there is greater correlation, and hence more distinguishable vine clusters of the same color, but this is hard to be certain. One intuitive reason could be that weekly information will have a bigger impact on entire sectors, whereas daily performance is highly variable** and a vast amount of other variables might have greater impact on the companies' stock values in the short term.



In order to confirm our intuition, we compared the distributions of the correlation coefficient themselves for the daily and weekly. **Indeed the distribution of coefficients has shifted slightly higher, but also with greater variance as confirmed by the increase in the mean and standard deviation. This is indicating correlation is marginally stronger on the weekly data, but with slightly more spread.**



2. Let's Help Santa!

2. Build Your Graph

Q6)

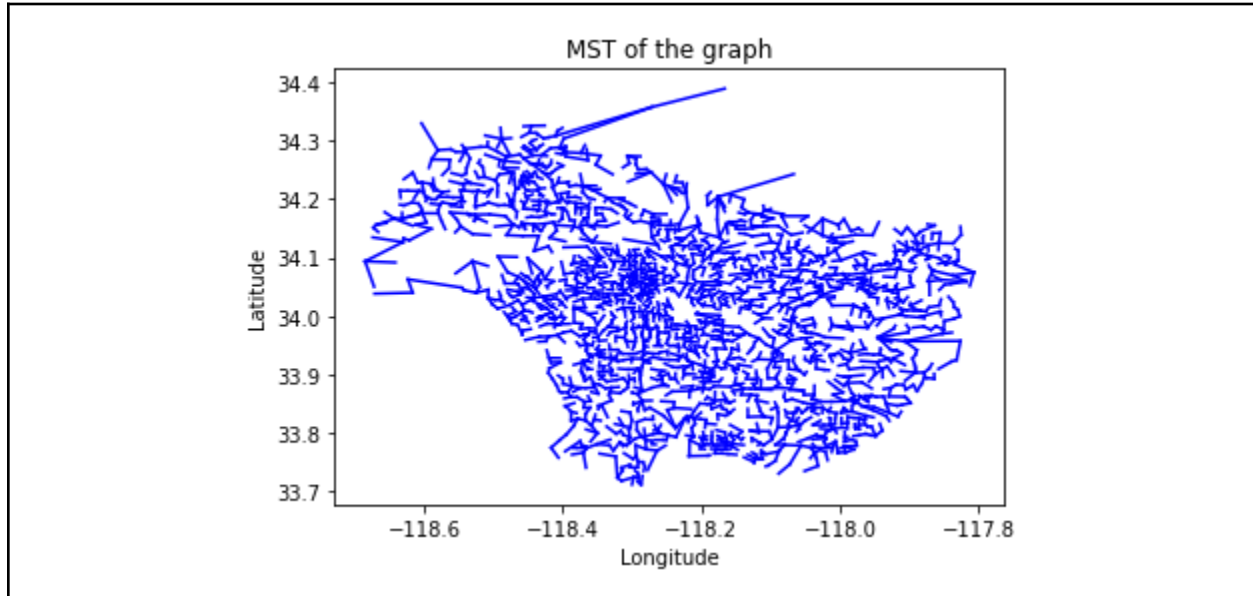
Since we only need the December data and the largest connected component of the graph, the subgraph has fewer edges and nodes than the original graph. The following information shows the number of edges and nodes of the subgraph.

```
number of edges: 1004955
number of nodes: 2649
```

3. Traveling Salesman Problem

Q7)

A minimum spanning tree of graph G is a tree whose nodes cover all the nodes in the graph. When the tree is found, we can use it to address the traveling salesman problem efficiently.



Display Name	Street Address	Display Name	Street Address	Distance
Census Tract 480302	823 E Grand Ave, Alhambra, CA 91801	Census Tract 481002	308 S Cordova St, Alhambra, CA 91801	0.6 miles driving

Census Tract 480302	823 E Grand Ave, Alhambra, CA 91801	Census Tract 480400	902 N Garfield Ave, Alhambra, CA 91801	1.0 miles driving
Census Tract 480303	400 N Marguerita Ave, Alhambra, CA 91801	Census Tract 480304	211-299 W Woodward Ave, Alhambra, CA 91801	0.7 miles driving
Census Tract 480303	400 N Marguerita Ave, Alhambra, CA 91801	Census Tract 480400	902 N Garfield Ave, Alhambra, CA 91801	0.9 miles driving

From the figure and the table, we know that the distance between node and node does not vary a lot. Normally, they only take a few minutes driving. Based on this property, we can use MST to address the problem of traveling salesmen. The intuition is that forming a tree by connecting the neighbor nodes could lead to the good estimation of a shortest traveling path to every node.

Q8)

We randomly pick 1000 sets of 3 points on the map to see if they satisfy the triangle inequality. In mathematics, the triangle inequality means that the sum of the lengths of any two sides of a triangle is greater than the third side.

Suppose there are 3 points A,B,C, then their mean_travel_time a for A to B, b for B to C, and c for A to C, then these times shall satisfy $a + b > c$ or $b + c > a$ or $a + c > b$ to meet the triangle inequality. We get about 89% of the triangles in this graph to pass the test.

Q9)

By the relationship introduced in the lecture note, we know that

$$\begin{aligned}
 \text{Length of the tour returned by the algorithm} &\leq \text{Distance covered by Euler cycle} \\
 &\leq 2 \times \text{Weight of minimum spanning tree} \\
 &\leq 2 \times \text{Weight of minimum tour length}
 \end{aligned}$$

So, an upper bound on the performance of the approximate algorithm is 2.
We can also compute the performance of the algorithm in the given graph.

The cost of MST: 260551.600000000033

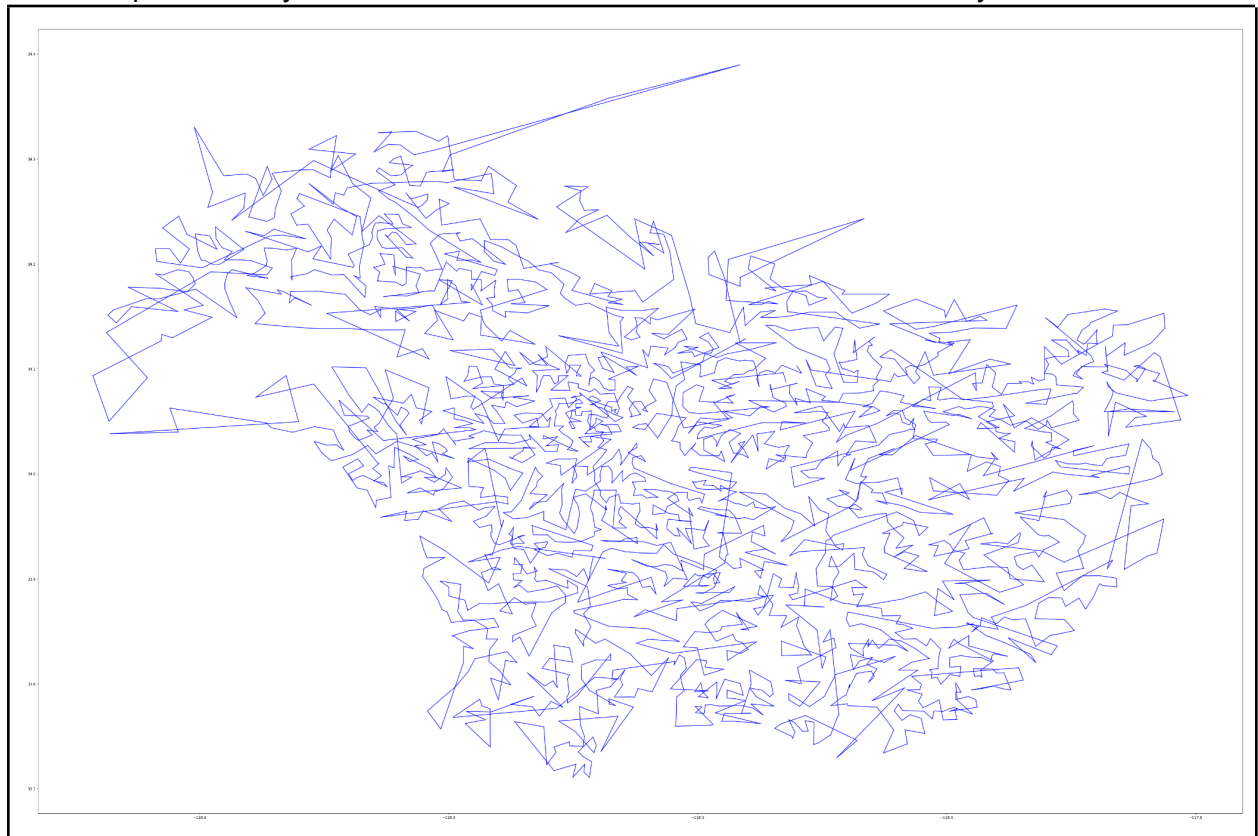
The cost by the approximation algorithm: 447269.33999999993

The two times cost of MST (upper bound): 521103.200000000065

Based on the calculation, the algorithm performs 1.71 times of MST cost, which is smaller than the theoretical upper bound.

Q10)

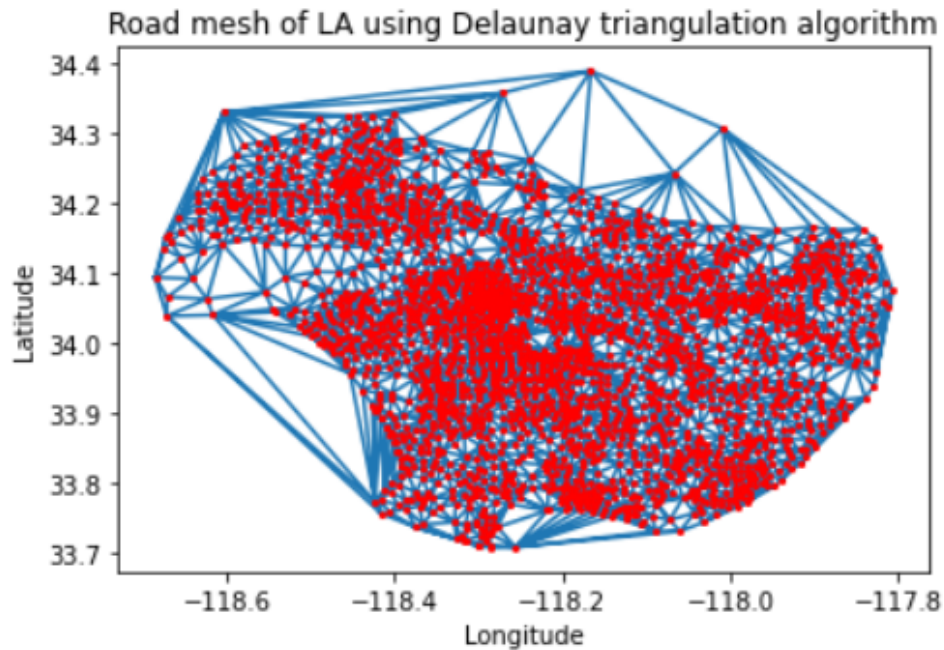
From the plot, we may see that the salesman travels to the locations exactly once.



5. Estimate the Roads

Q11)

Delaunay triangulation is an algorithm that takes a set of points and discovers connecting triangles with the condition that no points lie in the circumcircle of any derived triangles.



The figure above is the road mesh we obtain after doing Delaunay triangulation. We observe the red points representing the road structure of Los Angeles. There are more nodes in the urban area and much less in the countryside. If a place has more data, that place would be divided into more parts for the algorithm to analyze. However, there are many edges in the road mesh that actually don't exist, added by the Delaunay algorithm. Thus, we need to do other processes to make the road mesh more realistic.

6. Calculate Road Traffic Flows

Q12)

$$\text{Distance} = v * (\text{mean travel time} * \frac{1}{3600}) \text{ miles}$$

$$\text{Car length} + \text{gap} = 0.003 + (v * 2 * \frac{1}{3600}) \text{ miles}$$

$$\text{Number of cars} = \frac{v * (\frac{\text{mean travel time}}{3600})}{0.003 + (\frac{v}{1800})} * 4 = \frac{2 * v * \text{mean travel time}}{5.4 + v}$$

$$\text{Max traffic flow} = \frac{2 * v * \text{mean travel time}}{5.4 + v} / \frac{\text{mean travel time}}{3600} = \frac{7200 * v}{5.4 + v} \text{ cars/hr}$$

The mean_travel_time in the “los angeles-censustracts-2019-4-All-MonthlyAggregate.csv” is given in seconds. In order to calculate traffic flow, we need to convert it to hours. Distance is converted to velocity times mean travel time in hours. The car length is 0.003 miles and the gap length is velocity multiplied by 2 seconds of safety distance time (converted to hours). The number of cars equals the total distance divided by car length+gap. Note that it is multiplied by 4 because each road has 2 directions and each direction has 2 lanes. At last, we can calculate the traffic flow using the number of cars divided by travel time with units in cars/hr.

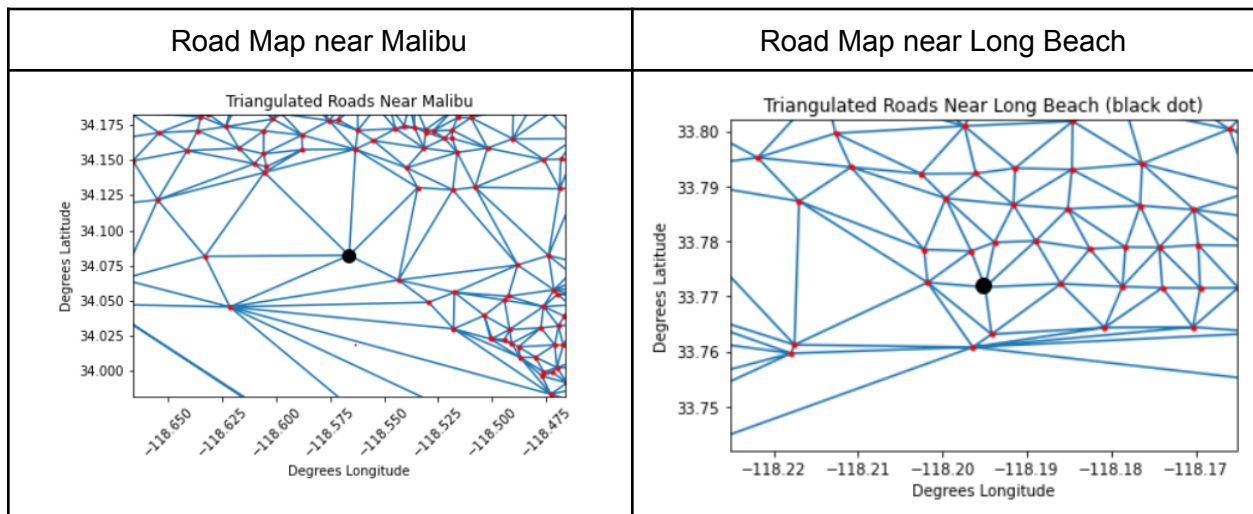
7. Calculate Max Flow

Q13)

For calculating the maximum number of traffic flows, we use the graph from Delaunay triangulation and the location to location commute times provided by Uber data. From Malibu to Long Beach, we get an average of 67 minutes commute time. Then by combining these informations and applying the traffic flow equation we derived in Q12, we got the maximum flow. In ideal circumstances, the mean_travel_time we need is nearly 0, so approximately 7200 Cars/hr we can get for the flow. However, in the real world we have to consider the impact of the traffic time thus reducing the flow. Our results are shown below:

Maximum flow from Malibu -> Long Beach	Number of Edge Disjoint Paths
6060.17 cars/hr	6

To get the edge disjoint point above, we obtain the result by looking at the zoom version of the road map at both Malibu and Long Beach. But since these two coordinates given by the project don't actually exist in our graph, thus our way is to find the nearest location of these two spots. We use the argmin function to find the coordinates that have the smallest distance with Malibu and Long Beach and use those two spots for the zoom version of the map to see how many vertices are there in each node. We get (34.082256, -118.566222) for Malibu and (33.76310184, -118.19416296) for Long Beach. The result is shown below:

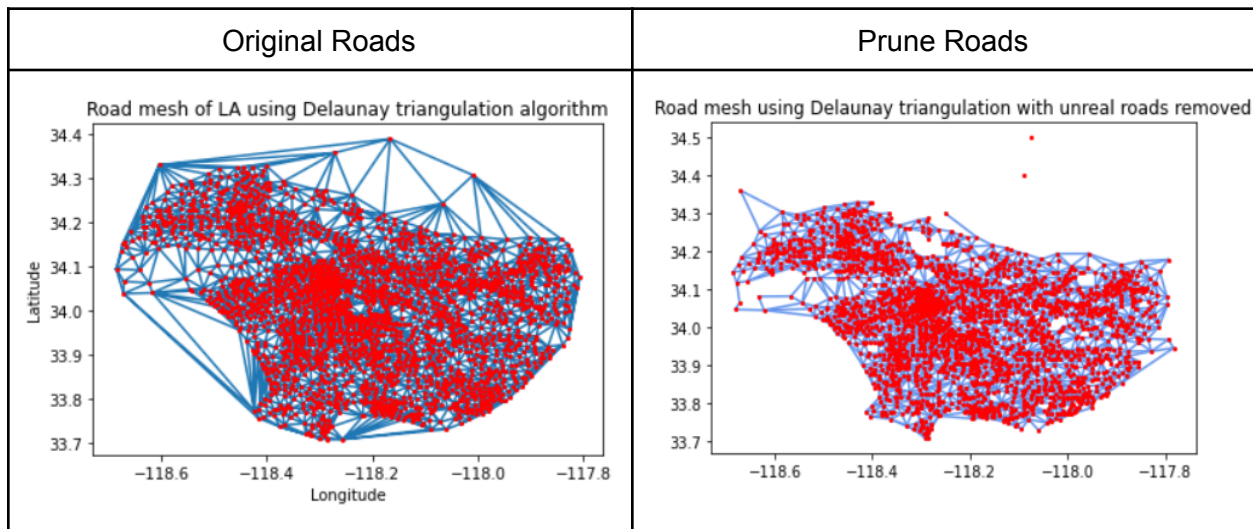


We can see that the number of nodes pointing out from Malibu are 6, and Long Beach also gets 6, thus we get 6 edge disjoint paths in this case.

8. Prune Your Graph

Q14)

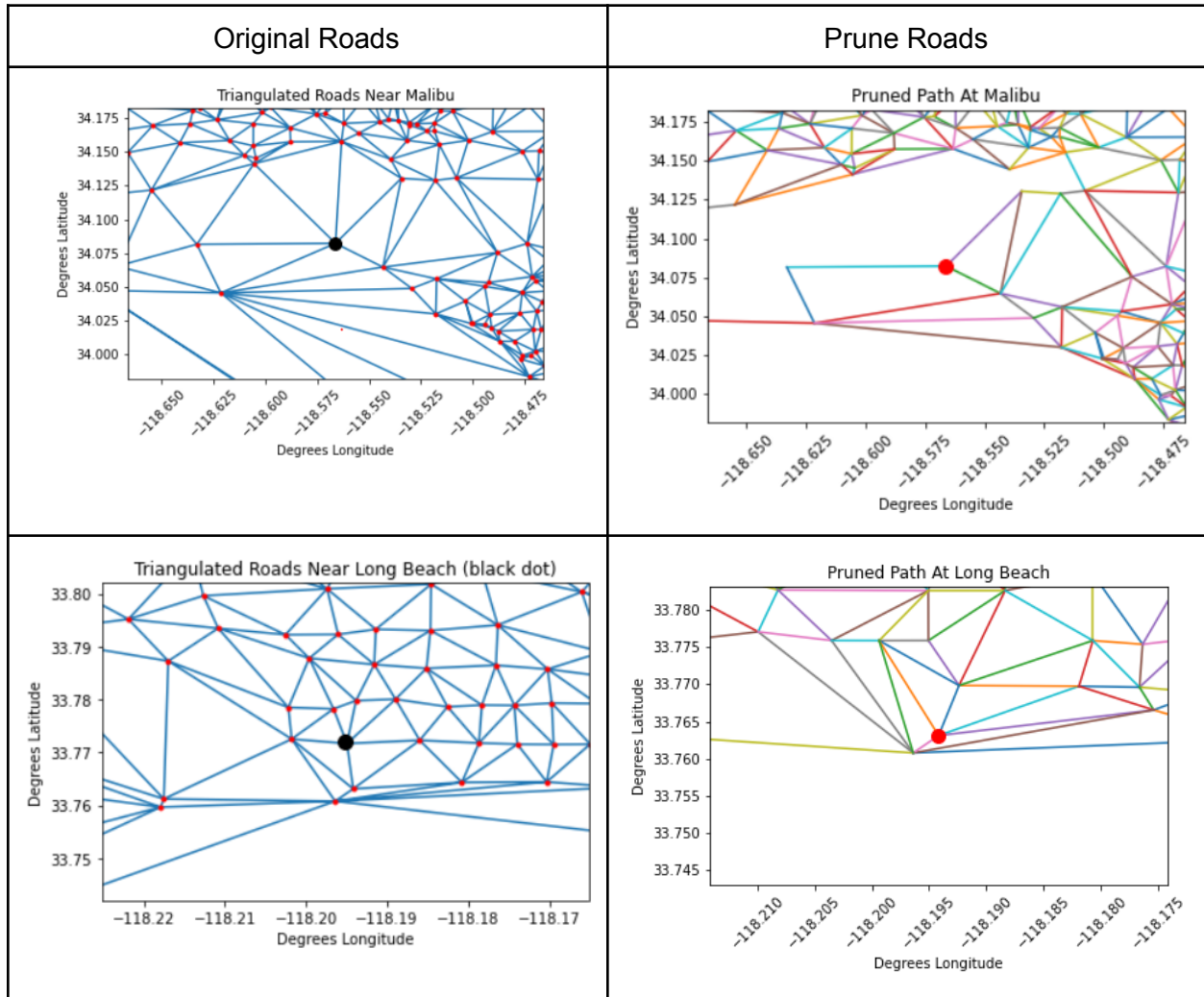
Since we are doing Delaunay triangulation, there will be some large edges created by the algorithm. These edges might be one of the edges crossing the concavities of the beach, as well as crossing the hills of Topanga. These fake edges will have large mean_travel_time so they should be eliminated to get the real graph. To remove the fake roads while preserving most of the real ones, we drop the edge that has a time threshold of 12 minutes. This results in the removal of most non-exist edges in those two areas. Below we show the road mesh of LA before and after pruning:



From the figure above, we can see that we remove the edges of the top and the left bottom corner of the graph, which are the hills of Topanga and the concavities of the beach. Since they have larger distance, thus leading to a longer commute time and potentially easier to get pruned.

Q15)

After pruning the fake roads of those two areas, we reconstructed the new pruned graph to see whether there are any differences. And we also re-computed the maximum flow and the number of edge disjoint paths between Malibu and Long Beach. The results are shown below:



	Maximum Flow	Edge Disjoint Paths
Before Pruning	6060.17 cars/hr	6
After Pruning	5136.5 cars/hr	3

We can see that after pruning, since the fake roads have been removed, one can no longer travel from Malibu to Long Beach using these paths, thus the number of edge disjoint roads diminished from 6 to 3. And due to this fact, people have less options to choose, thus causing the maximum flow of the remaining paths also diminished.

9. Define Your Own Task

Introduction

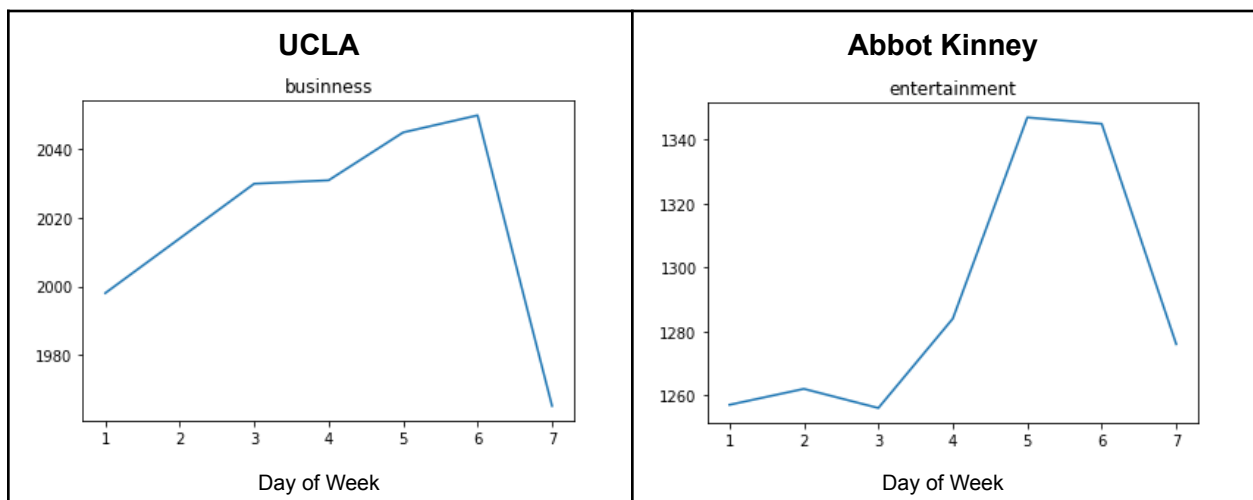
We decided to use the same style of traffic data from **Uber to perform a classification task of locations based on their traffic patterns**. In order to better understand the results of an unsupervised classification method, we first generated a test case list of **two classes: entertainment locations and business/academic locations** (see appendix data). The intuition being that regions dominated with entertainment will have fundamentally different traffic patterns, with greater traffic in the evenings and on weekends.

Method - Feature Selection

The table below shows our selection of locations along these two categories. **We used the “Quarter 3 2019” datasets, analyzing the traffic patterns for day of week and hour of day. For every node in the graph we took the ‘neighbor’ graph, and used the minimum edge weight (traffic time) as the node’s traffic time.** We initially tried mean and a few other methods across all edges, but we believe the min edge seems to best represent local traffic patterns for this study.

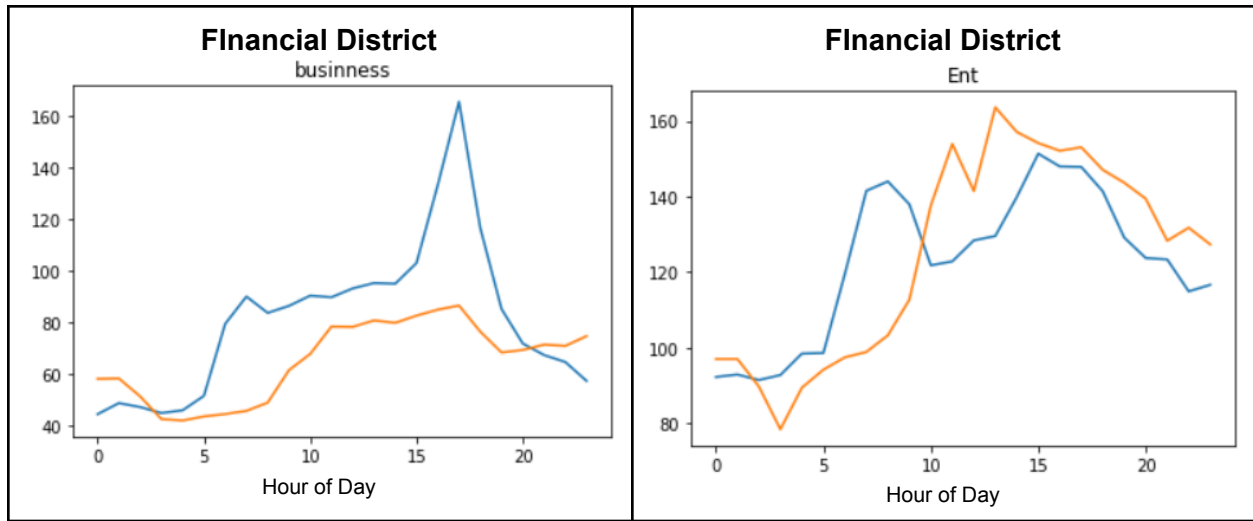
Day of Week

When we used the day of week data, we found that **averaging across the day did not distinguish our locations well enough**. The plots below show Abbot Kinney (majority entertainment) vs UCLA (majority business/academic) and we can see some change in the width of peak traffic. But the overall shape of the curve is tough to distinguish. We believe that this is due to the average over the day. Since the traffic only gets bad in a few hours of a day (i.e. before/after work hours), taking the average operation in the whole day will vanish this feature. To this sense, we start to seek if there is any weekday and weekend dataset.



Combining Day of Week with Hour of Day:

Based on the previous results, we start to generate the features from the dataset of hour of day in weekdays and weekends. In the feature extraction, we chose the mean travel time from a node to the nearest neighbor since it somehow reflects the local traffic status. If some locations have the worst traffic in some time, that will increase the traveling time to the nearby locations. According to this intuition, **We generated an hour of day curve for both the weekend and the weekday (resulting in 48 dimensional features per node)**. The plot below shows these features. We can see **business locations have a clear “after-work” traffic spike on the weekdays** while the entertainment locations stay the same on both weekdays and weekends. We believe that this feature space offers us the best chance to cluster our locations.



--- Weekday
--- Weekend

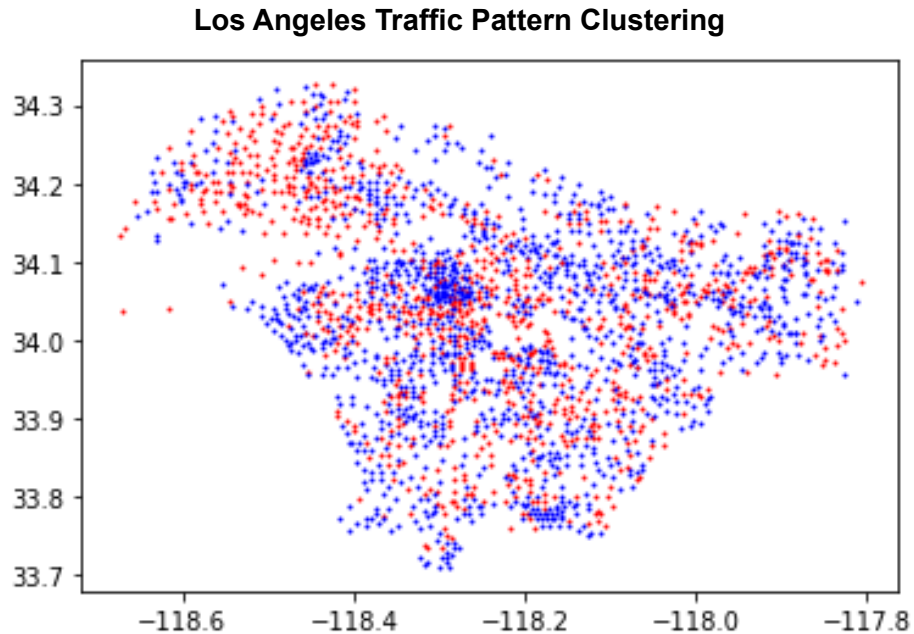
Method - K-means Clustering

After collecting our features, we utilized a **standard k-means clustering algorithm** to process the data into two distinct clusters. The k-means algorithm seeks to optimize the minimum means distance of nodes from 'k' centroids that are classified as across the feature space:

$$\arg \min_{\mathbf{S}} \sum_{i=1}^k \frac{1}{2|S_i|} \sum_{\mathbf{x}, \mathbf{y} \in S_i} \|\mathbf{x} - \mathbf{y}\|^2$$

Results

The following plot shows the results of the clustering algorithm across our chosen feature space. **From a high-level, and detailed interpretation is certainly tricky** as there is major crossover in business and entertainment locations through LA county. We notice a **higher density of blue locations along the coast which we would predict as being more entertainment type traffic patterns**. **The red locations seem to overlap more with major city centers**, but these city centers often have a high density of both types of locations.



Due to the granularity of the plot, our full set of test locations reduced down to 12 entertainment nodes and 13 business nodes.

	Classification Accuracy
Entertainment (Blue)	83.33% (10/12)
Business (Red)	46.15% (6/13)

The average accuracy is 64.74%.

Conclusions

The entertainment clustering ended up having a reasonably high accuracy, but the business areas were not nearly as accurate. In general, a lot of our business locations are in big city centers where there is a large mix of both types of locations. **We believe this algorithm would perform significantly better if we had greater location granularity, and added more features from the traffic dataset to further distinguish locations by their associated traffic patterns.**

While the accuracy is not quite high-enough to make a definitive statement, one could find many practical uses of an algorithm that determines the types of establishments in a location based solely on traffic data. **Improving recommendation engines for locations, traffic avoidance, and other similar uses come to mind as viable applications of such an algorithm.** Further research could better develop the idea into its numerous practical uses.

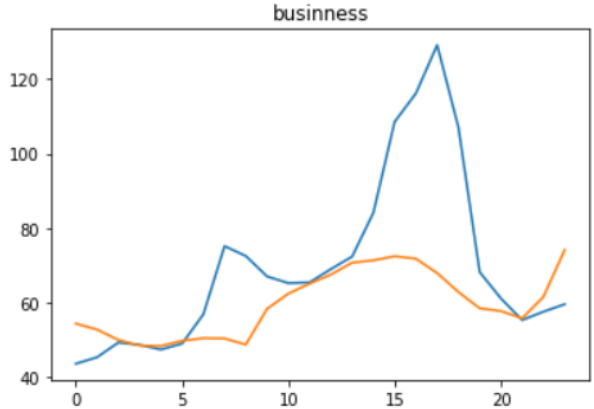
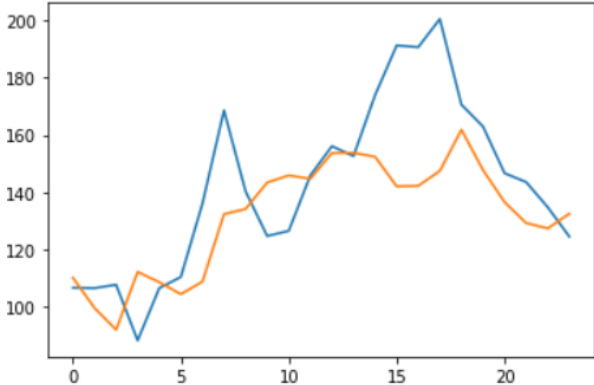
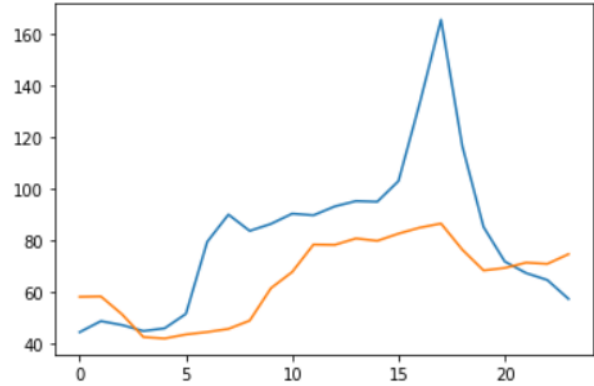
Appendix Data

Test Locations

Entertainment		Business and Academic	
Location Name	GPS Coordinates	Location Name	GPS Coordinates
Getty Museum	34.079109, 118.474469	Santa Monica Business Park	34.021496, 118.452710
Venice Beach	33.98749212549136, -118.47403102521139	West Hollywood (1)	34.08249493100406, -118.38444281055526
Hermosa Pier	33.861797244139616, -118.40073478159646	West Hollywood (2)	34.092702697895795, -118.3657804462937
Griffith Observatory	34.11875810198704, -118.30035313864202	LAX	33.94147646179464, -118.41899180718153
Hollywood Bowl (concert)	34.11206605736903, -118.33896447974048	UCLA	34.06858869409452, -118.4438194761533
Dodger Stadium (baseball)	34.07397740801106, -118.23991286415264	Financial District LA	34.05177388672339, -118.25487549746245
Abbot Kinney:	33.993083, 118.474597	USC	34.06040131113, -118.20437039121283
Santa Monica Promenade:	34.024360, 118.496276	Loyola	33.97038093926426, -118.4163851772907
Manhattan Pier:	33.88476125173915, -118.41072212325572	Culver City Station	34.02740213271951, -118.3888228586567
SM Pier:	34.009484536400336, -118.49382892991356	Northrup	33.8946124592154, -118.37476741447813
Universal Studios:	34.13731918288014, -118.35211683810894	Disney Studios	34.1570145961555, -118.32552272232894
Long Beach Marina Area:	33.76418814975836, -118.19112300405536	Cedars Sinai Medical Center	34.07484806375239, -118.3809572429272
LA Bar Area:	34.04798926686432, -118.25696906506356	CalTech	34.13777854469987, -118.12475934679715
Art Walk:	34.047601645757254, -118.24836109606531	Fashion District	34.03538568743437, -118.2546454616027

Redondo Pier:	33.839999109936, -118.38945764306173	CSLB	33.78544763931632, -118.11373827993155
---------------	---	------	---

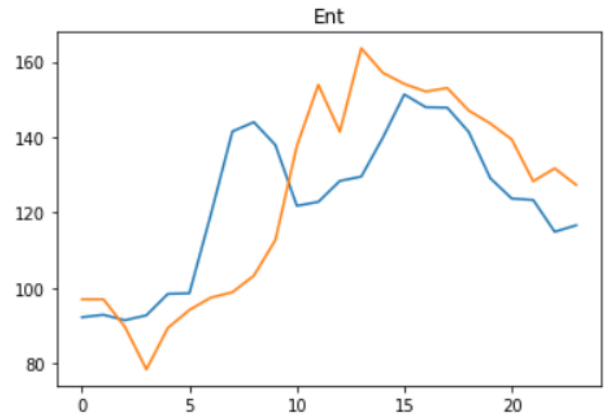
Additional Feature Plots

Locations ID	Distribution of workdays and weekends for commuting time
ID:1254 Financial district [34.03538568743437, -118.2546454616027]	
ID:1535 UCLA [34.06858869409452, -118.4438194761533]	
ID:1232 Financial District LA [34.05177388672339, -118.25487549746245]	

ID:1956

Santa Monica

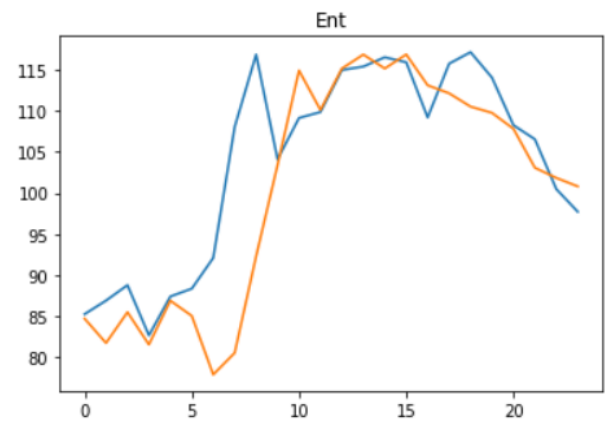
[34.024360, 118.496276]



ID:1594

Abbot Kinney Venice

[33.993083, 118.474597]



ID:2164

Universal Studios

[34.13731918288014, -118.35211683810894]

