**Location-Based Personalized Advertisements**

Jiewei Gao

March 2nd , 2019

1. **Introduction/Business Problem**

**1.1 Background**

Marketing is an essential topic for every business. If you want to boost your business, you have to present your products to the audience, or even better, the targeted audience. Commercials, advertisements, coupons are everywhere now in digital world nowadays. People cannot avoid them but also, they cannot be attracted to most of the information because they have limited concentration and would rather allocate more time on things they actually desire. That’s why customers are welcoming to relevant marketing messages rather than being bombarded with a plethora of deals that do not excite them. A customer in the kids’ section is likely to be interested in offers on kid garments. However, a standee displaying offers on kids’ garments is not relevant to most customers walking passed it.

**1.2 Problem**

So now the question lies ahead, if you are business owner and you want to advertise your products, how can your ads be targeted more efficiently on potential customers? if possible? These would be the business problem that we would like to solve in this project.

**1.3 Potential Solution**

Geo-Targeted Mobile Ads could be one of the solutions! Geo-targeting services from Google, Yahoo!, allow advertisers to allocate search campaign resources at a local level. With geo-targeting options available today, audiences can be targeted at the country, state, city, and ZIP code level to determine the best potential ad placements.

1. **Data acquisition and cleaning**

**2.1 Data Source**

We mentioned above that with Geo-Targeted Mobile Ads we can target on potential customers better. Now we need to know the characteristics of each neighborhood in the city so that we can know where to dispose different kinds of mobile Ads. In this case, let’s use **Los Angeles** as our object. There are many great resources we can find online to grab data. We choose 'Los Angeles Times' and use the 'Mapping L.A. Boundaries API' to download neighborhoods covering Los Angeles. This layer is a filtered version of the Los Angeles Times neighborhood boundaries that only includes boundaries of neighborhoods fully or partially within the City of Los Angeles. We use the geojson version of the file.

(Find more info about dataset used in this project: http://remakela-lahub.opendata.arcgis.com/datasets/d6c55385a0e749519f238b77135eafac\_0)

**2.2 Data Cleaning**

Now we can start cleaning data and categorizing them to show neighborhoods and their coordinates. Moreover, we can combine the useful information about venues and category from Foursquare with geolocation data processed before to have a better understanding of which region is more likely to consists of convenience store, for example. In that case we can distribute 7-11 ads or coupons to those areas where more people are interested in convenience store. Thus the money spent on ads are being used more efficiently and are bring more investment returns to business owners.

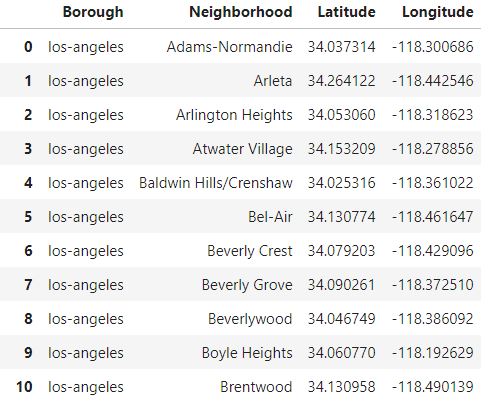
After downloading data, we notice that all the relevant data is in the ‘features’ key. In lists of data under ‘features’, there are many types of data like id, geometry, properties and so on. What we are going to need in our project are properties name, geometry coordinates. Now when we look at the data, we find out that the dataset contains several problems.

* Data not only includes Los Angeles, but also Orange County. Since our object is Los Angeles, we discard Orange County data to clean out unnecessary data.
* There are multiple coordinates under same neighborhood. In the following steps, we need to map out the locations and also the Foursquare API needs to recognize neighborhoods based on coordinates. In order to avoid further misunderstanding from the data standpoint, we group their coordinates based on neighborhoods and calculate the mean of them as the coordinate for the neighborhood location.
* Data are arranged in lists and are not easy to interpret. Before analyzing data, we need to transform dictionaries into pandas dataframe. Then we append ‘Borough’, ‘Neighborhood’, ‘Latitude’, ‘Longitude’ to the dataframe to better format it.

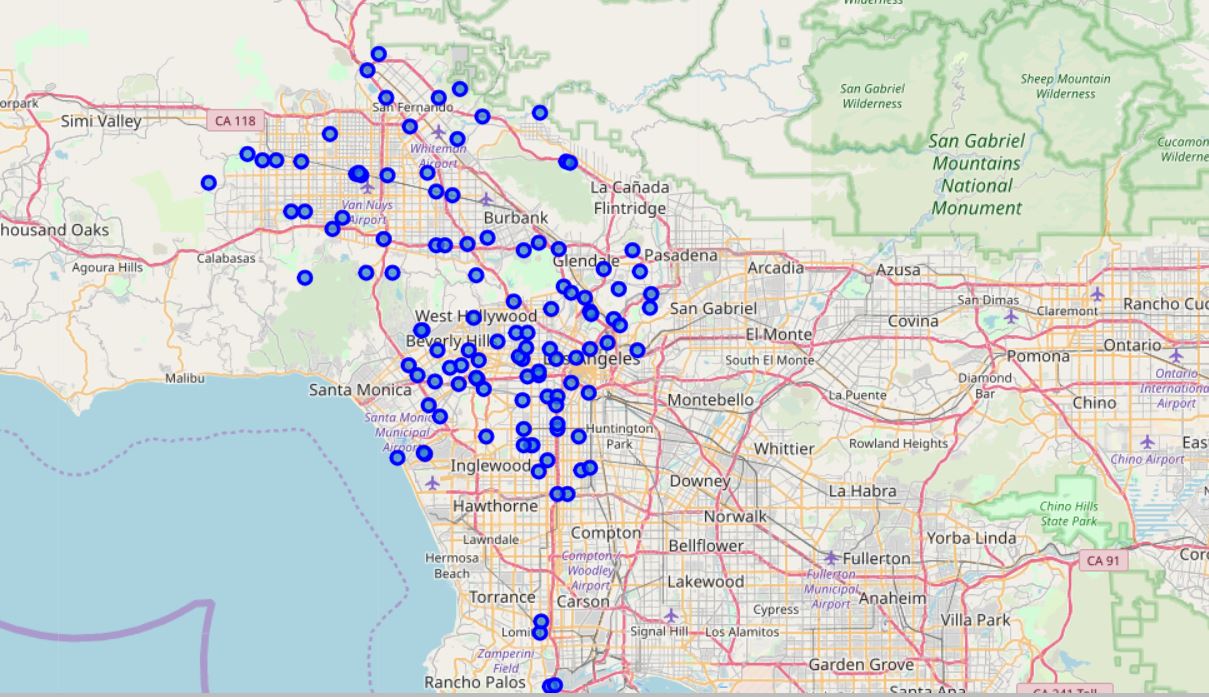
.

1. **Exploratory Data Analysis**

Now that we have the formatted data on hand. Let’s take a quick look at the data. There are 113 different neighborhoods in city of Los Angeles.

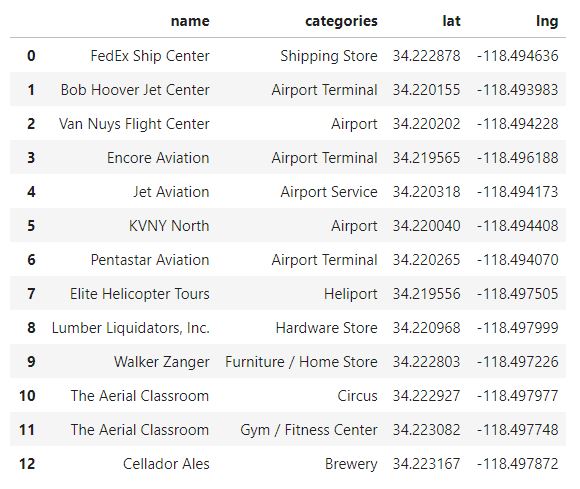


The coordinates of city of Los Angeles needs to be obtained to visualize data. (we need to define a user\_agent in order to define an instance of geocoder) And The geographical coordinate of Los Angeles is 34.0536834, -118.2427669. Now we create a map of Los Angeles with neighborhoods superimposed on top of the map.



To further analyze the data, we choose one random neighborhood and interact with Foursquare API in order to have a better understanding of the dataset combined. We randomly choose the 66th neighborhood in our data and it is ‘North Hills’. Again, we obtain the coordinate of this location, and the latitude and longitude values of North Hills are 34.2202300938775, -118.495248461462.

With Foursquare API, we can also get the top 100 venues that are in ‘North Hills’ within a radius of 500 meters and their coordinates. From Foursquare, we receive data that are returned from our request and we filter out columns that we need, which are ‘name’, ‘categories’, ‘latitude’, ‘longitude’.



In this example, we can clearly see the venues around 'North Hills' in LA. In Foursquare dataset, it returns 12 venues available that are near North Hills of Los Angeles. We have categorized those venues. Most of the venues are venues related to airport and heliport. Also, there are bar, stores, gym, shipping store available. With each venue, we attached the coordinates of them and later we need these datasets to cluster similar characteristics.

1. **Methodology**

Since now we have already got a better hang of the data, let us start utilizing the Foursquare API to explore all the neighborhoods in Los Angeles and segment them properly based on **K-means Clustering**.

First, we need to repeat the process above to all the neighborhoods in Los Angeles. By creating a function that returns only relevant information for each nearby venue, we generate a dataframe that includes all venues nearby of each neighborhood in Los Angeles. The dataframe actually has 274 unique categories.

Next, we are going to turn venues data into numeric data. The reasons why we do this are because K-means Clustering requires data to be numeric and venues data are better for ranking when they are in numeric form. Thus, we use **one-hot encoding** to turn venues data into numeric data. Basically, we list all the venues that appears in our dataframe as columns and if the neighborhood has a venue that fits the column, then the value would be 1, if not then 0.

After restructuring the venues data, we sort out the dataframe that display top 20 venues of each neighborhood. As we can see, for example, the most common venue of ‘Adams-Normandie’ is ‘Café’, and the second venue is ‘Donut Shop’ while the third is ‘Mexican Restaurant’.



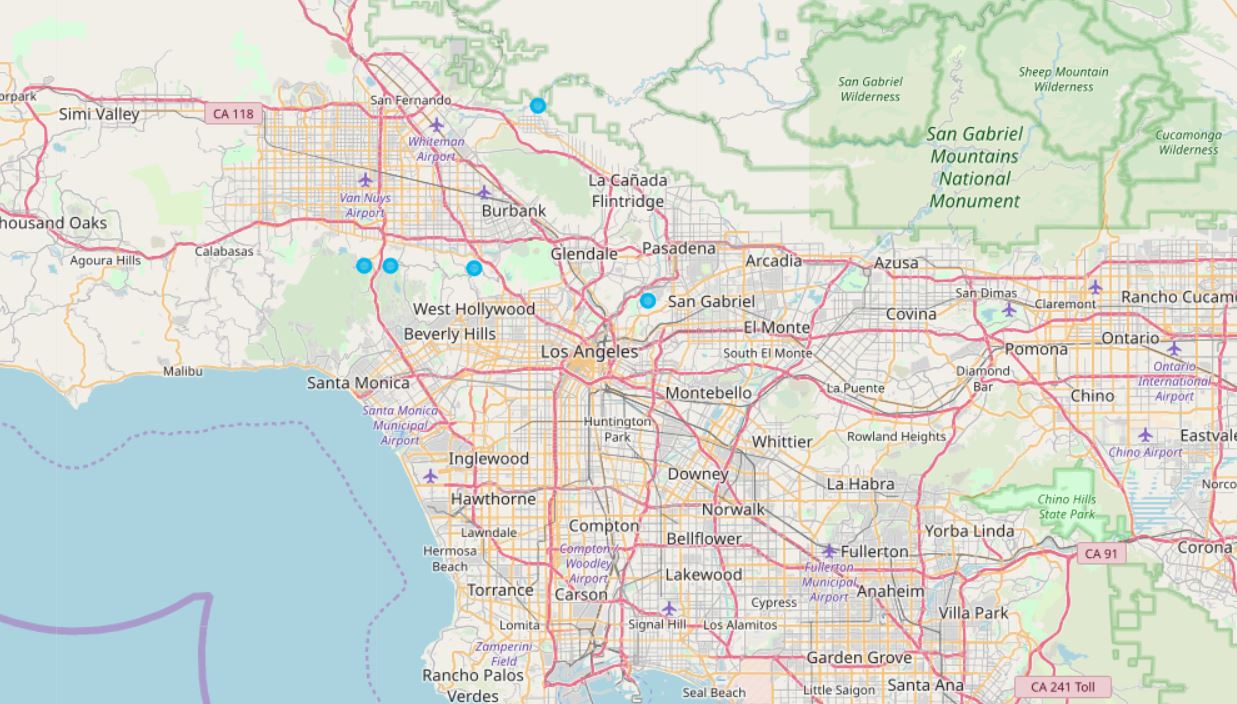
Now it is time for the other usage of one-hot encoding data. For K-mean Clustering, we set that there are 5 clusters in the dataset. With the cluster label, we can plot them on the map using different colors. The interesting thing here is that after clustering, we find out one particular cluster (cluster 3) has almost 70% of the dataset, which consists of a wide range of categories of venues. Since our ads need to be targeted precisely, cluster 3 will not help us that much regarding the precision. **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** can be used to furnish our cluster so that the cluster would be more precise without any additional noise. Also DBSCAN does not require the analyst to select the number of clusters a priori — the algorithm determines this based on the parameters it's given. DBSCAN excels at clustering non-spherical data. As mentioned above, DBSCAN does not force every data point into a cluster — if a data point lies outside of the specified starting parameters, it will be classified as "noise" and not included in any resulting cluster.

1. **Result**

Now with enough outcome of the process, we can finally look into the result. So, there are 5 clusters available in the outcome. In the cluster section below, top 20 venues of each neighborhood under the same cluster will be displayed and the neighborhoods under the same cluster will also be displayed as plots in map.

* **Cluster 1**

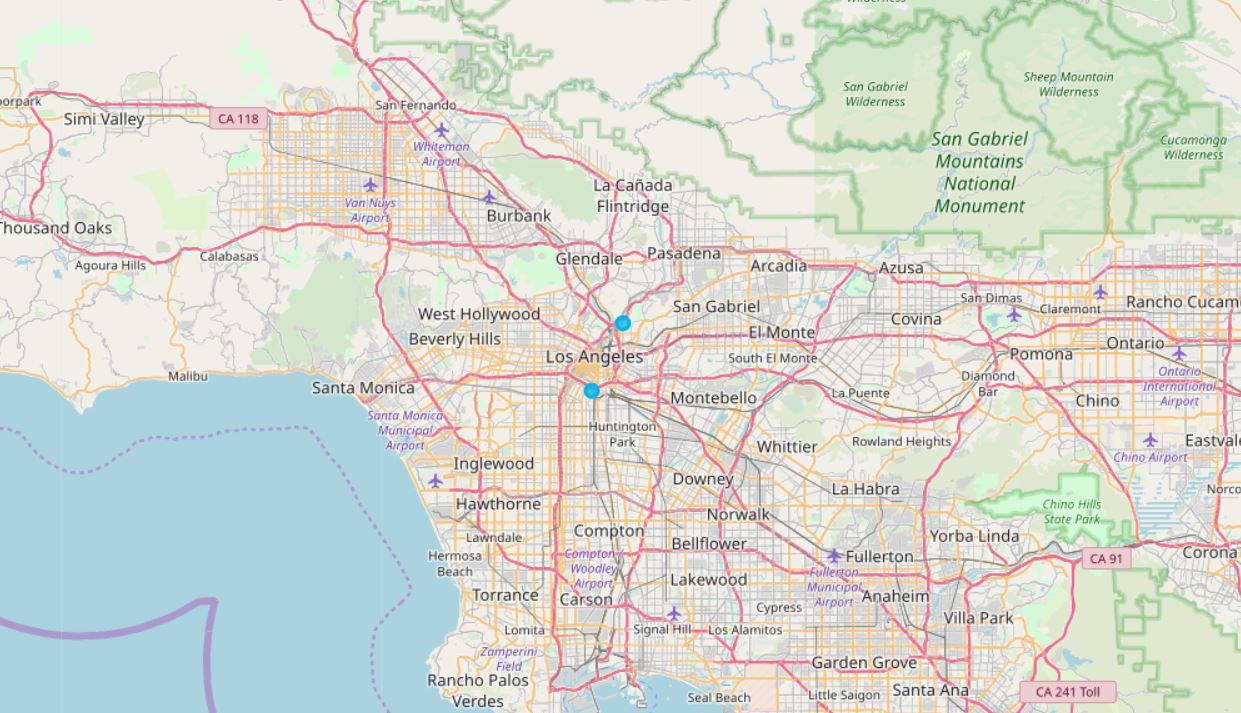




In Cluster 1, there are 5 neighborhoods that are classified together, which are ‘Bel-Air’, ‘Brentwood’, ‘El Sereno’, ‘Hollywood Hills West’, and ‘Sunland’. It is also quite obvious to see that in the top 5 most common list of the cluster, they all share some categories, which are ‘Scenic Lookout’, ‘Zoo’, ‘Empanadas Restaurant’, ‘Donut Shop’. With this information, we can reach a preliminary conclusion that with regarding ads that are related to these categories, these neighborhoods should be the first location to considerate targeting!

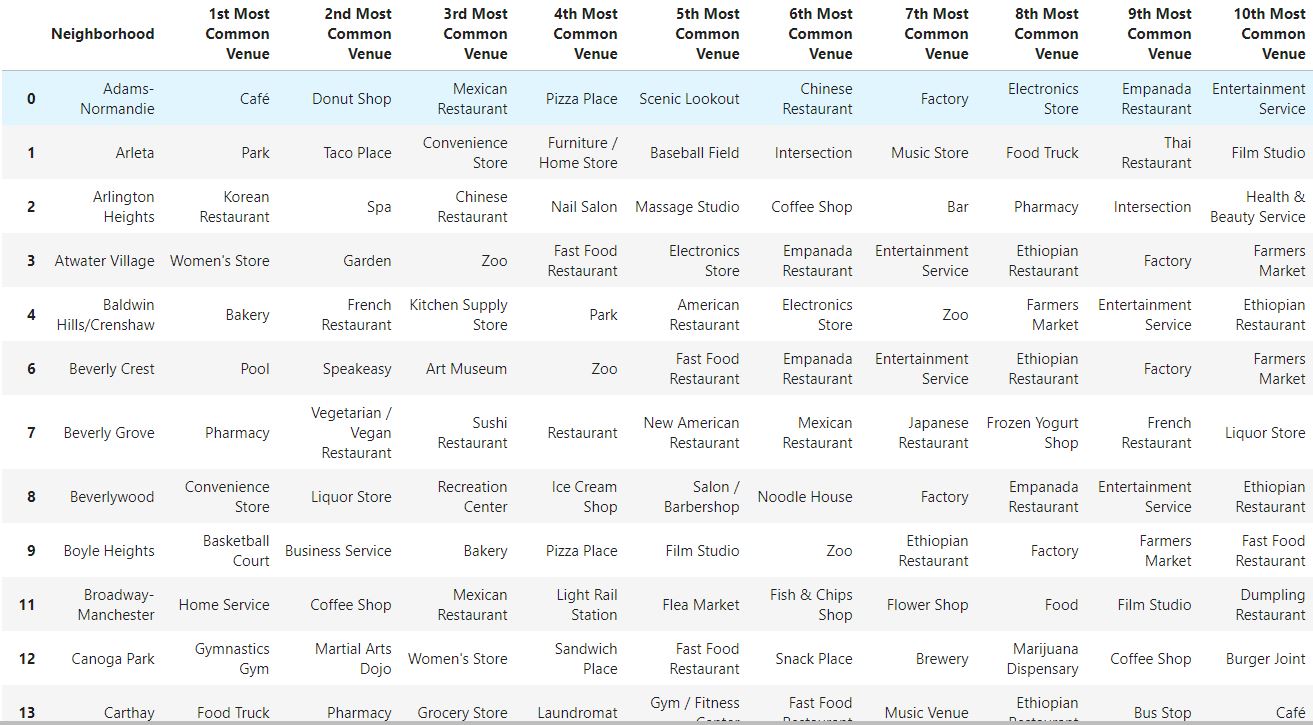
* **Cluster 2**

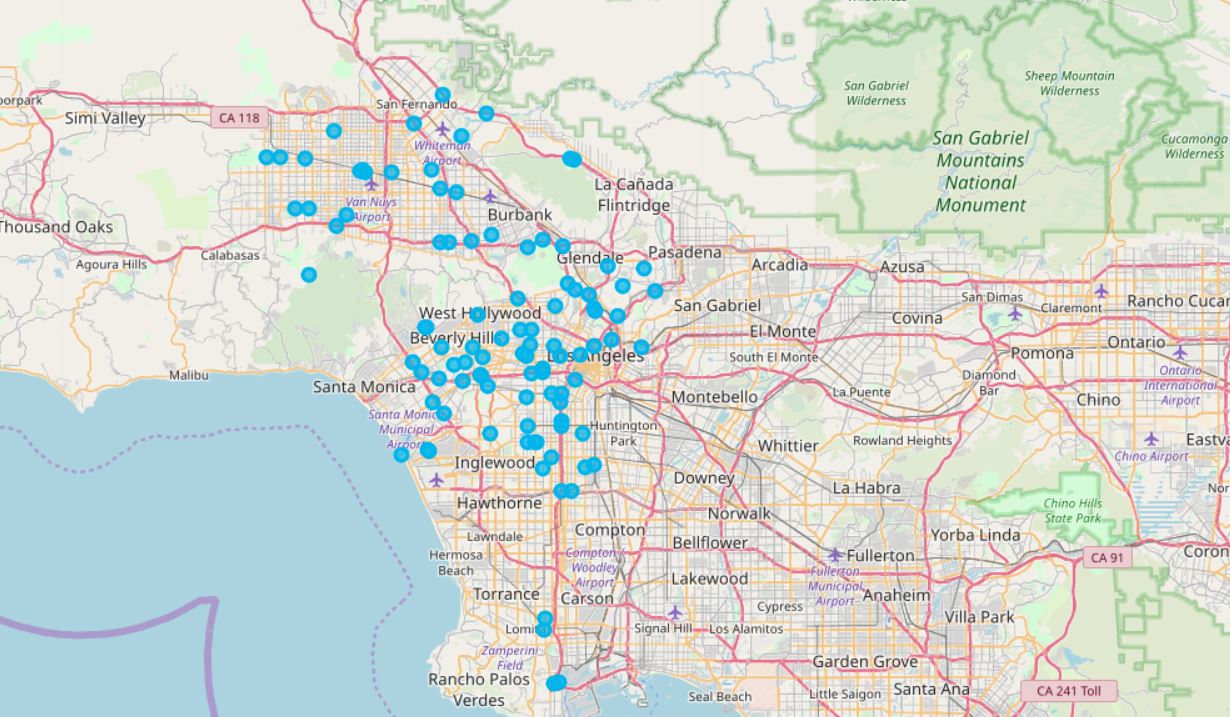




In Cluster 2, there are 2 neighborhoods that are classified together, which are ‘Central-Alameda’ and ‘Lincoln Heights’. The datapoints in this cluster are quite scarce and are also not enough to represent some categories. We would better leave this cluster and save for later uses.

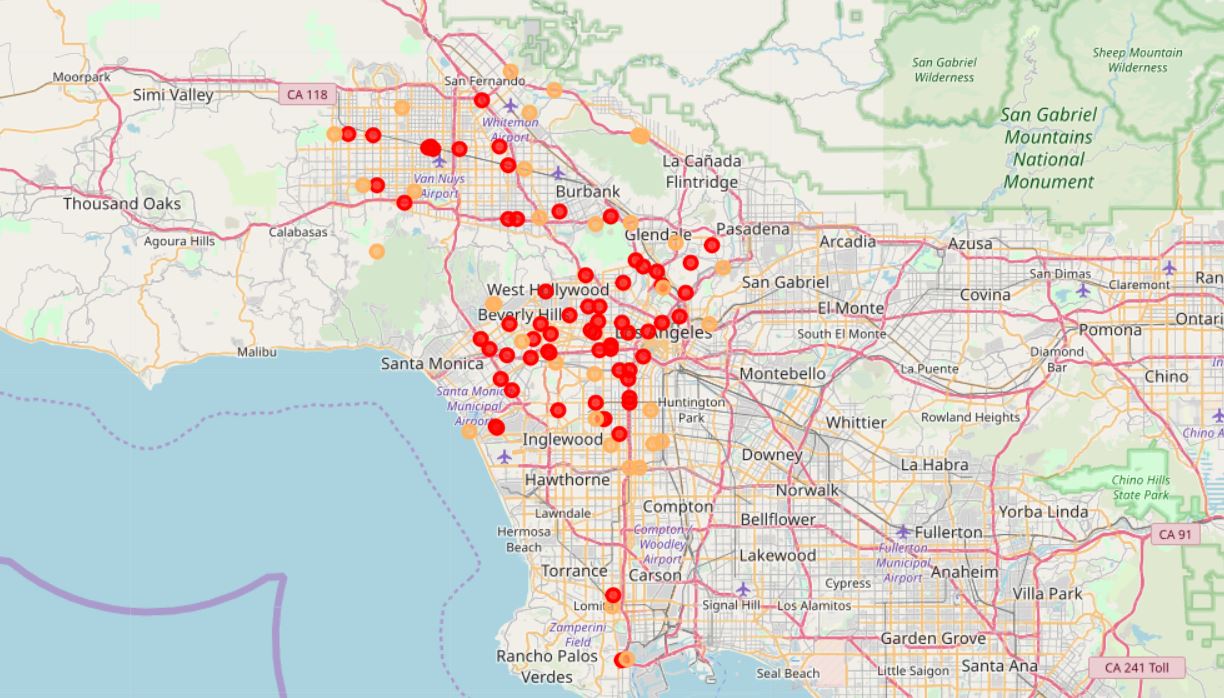
* **Cluster 3**





In Cluster 3, there are 99 neighborhoods that are classified together, and we are not going to show all the neighborhoods here. This cluster is too dense that we can not properly classify. But we also can not just leave them because it includes too much useful data. To solve this dilemma, we use DBSCAN to clear out noises and extract the core useful values inside this cluster.

After trying several parameters for DBSCAN, we decide to use 0.4 as epsilon and 10 as minimal sample number. (Epsilon stands for a measure of radial distance extending from a data point) Clearing the unnecessary noises (33 neighborhoods), we have 66 neighborhoods on hand and they are all classified as one cluster using DBSCAN even after we try to cluster in cluster. The cluster after DBSCAN is shown as below. (Red plots are the new cluster3 while the yellow plots are noises detected by DBSCAN)

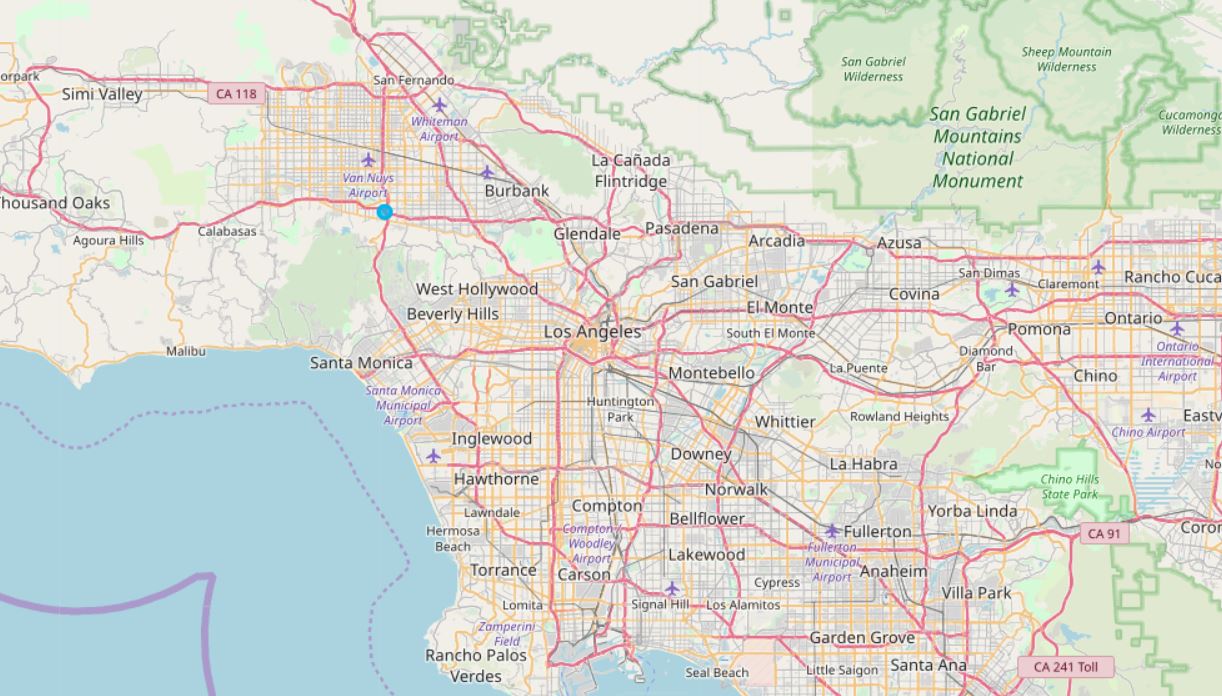




With narrower range of geographical location, ads can be targeted with less cost. Similar categories of venues can be found under neighborhoods of this cluster.

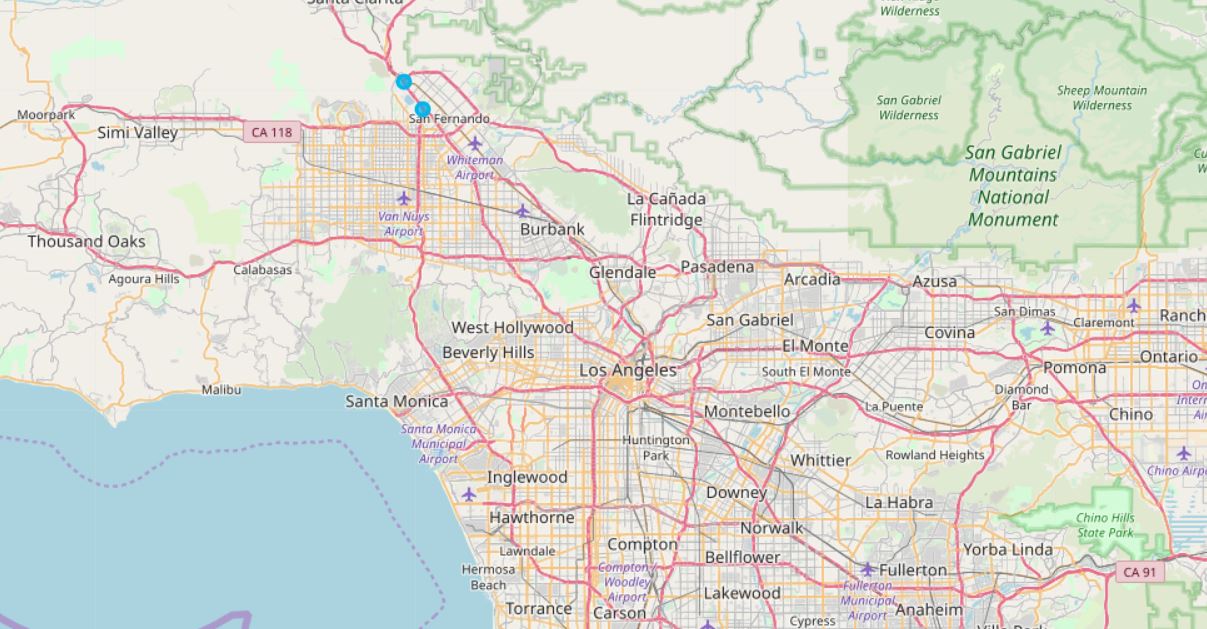
* **Cluster 4**





* **Cluster 5**





In Cluster 4, there are only 1 neighborhood ‘Sepulveda Basin’. In Cluster 5, there are only 2 neighborhoods ‘Granada Hills’, ‘Mission Hills’ that are classified together. Again, like Cluster 2, the datapoints in these clusters are quite scarce and are also not enough to represent some categories. We would better leave these clusters and save for later uses.

1. **Discussion & Conclusion**

Finally, we are reaching at discussion and conclusion. In this project, we analyze the neighborhoods of Los Angeles and use Foursquare API to extract the top 20 venues of each neighborhoods. We use K-mean clustering to generate clusters and then use DBSCAN to further improve the cluster quality. With clustering data, we can have similar neighborhoods that share similar categories of venues. In this way, we can target personalized ads at neighborhoods that have the same category as the ads or neighborhoods that are in the cluster which also has the same category. More precision of clustering leads to less cost and more efficiency of personalized ads targeting. However, there are some clusters that we are not able to utilize. These clusters are more difficult to categorize based on their values but if they can be reassessed, it would bring more accuracy to the ads targeting.

1. **Reference**

Sukup, J. (2018). When K-Means Clustering Fails: Alternatives for Segmenting Noisy Data. https://www.datascience.com/blog/k-means-alternatives

Pinard.(2016) Use scikit-learn for DBSCAN clustering. https://www.cnblogs.com/pinard/p/6217852.html