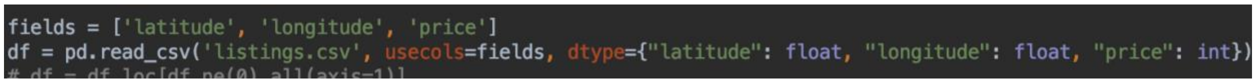
Leyang Shen

CS506 HW1

Q1.

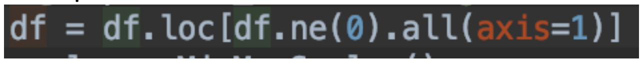
- **Data Filtering and type conversion**

  Obtain the data only from column: “latitude”, “longitude” and “price” and change the data type of each to float and INT accordingly.



```
fields = ['latitude', 'longitude', 'price']
df = pd.read_csv('listings.csv', usecols=fields, dtype={"latitude": float, "longitude": float, "price": int})
# df = df.loc[df.ne(0).all(axis=1)]
```

It is also reasonable to remove the price that is listed as 0 because no one wants to offer their place for free. So we remove the rows that contain 0 for price by:



```
df = df.loc[df.ne(0).all(axis=1)]
```

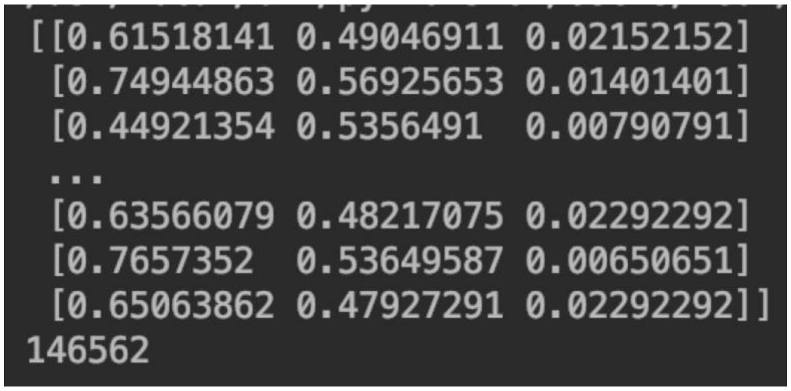
- **Scaling**
- Latitude, longitude and price have different units and the features with high magnitudes will weigh in a lot more than the features with low magnitudes. So in this homework, I choose to use Min-Max scaling for reduce the error in dataset.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

-

- This scaling will bring the value between 0 and 1.

- After preprocessing and scaling, our dataset will look like following (e.g. dataset + total num of data):

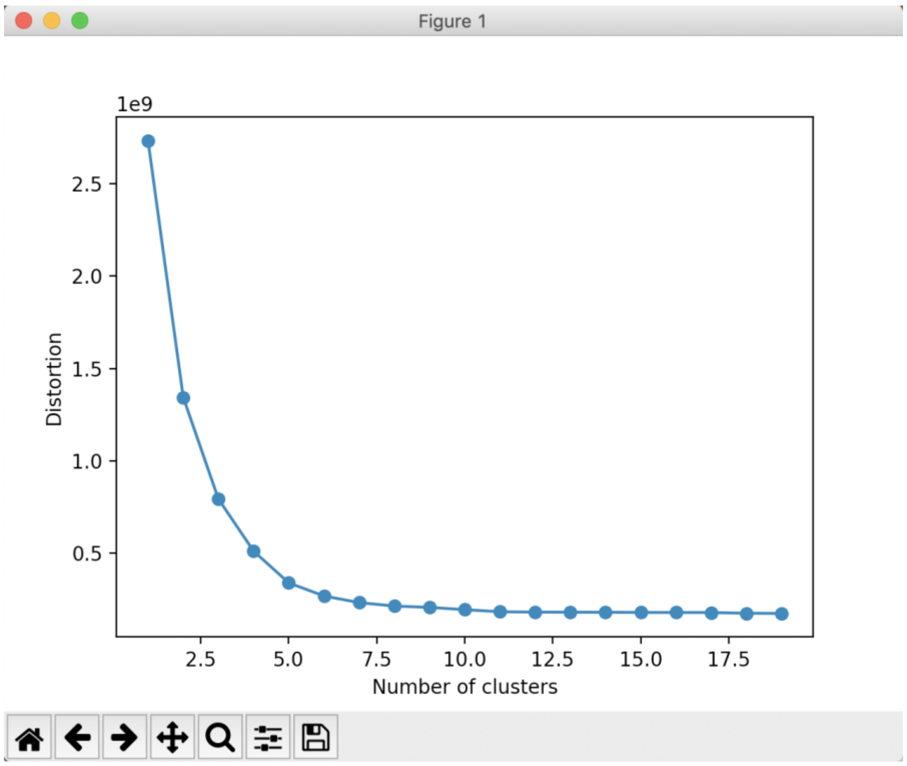


```
[[0.61518141 0.49046911 0.02152152]
 [0.74944863 0.56925653 0.01401401]
 [0.44921354 0.5356491  0.00790791]
 ...
 [0.63566079 0.48217075 0.02292292]
 [0.7657352  0.53649587 0.00650651]
 [0.65063862 0.47927291 0.02292292]]
146562
```

-

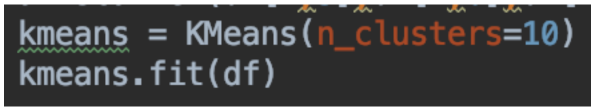
- **Elbow plots (obtain k for k-means++ clustering)**

- We plot number of clusters against inertia (within distance of points). We expect the inertia distance to decrease as number of clusters decreases. For example, when every point belongs to a cluster itself, then distance is 0, but not useful - no meaningful interpretation. Typically, we choose the number of clusters, after which the distance stops increasing rapidly. Most importantly, by doing this, we will get our k for clustering in the future.
-



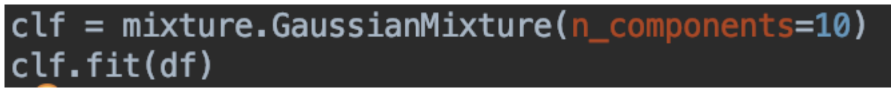
-

The graph become stable when k = 10.
K-means++ method:



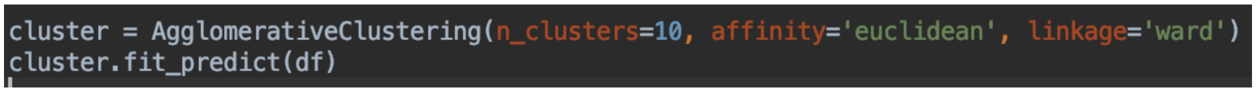
```
kmeans = KMeans(n_clusters=10)
kmeans.fit(df)
```

GMM method:



```
clf = mixture.GaussianMixture(n_components=10)
clf.fit(df)
```

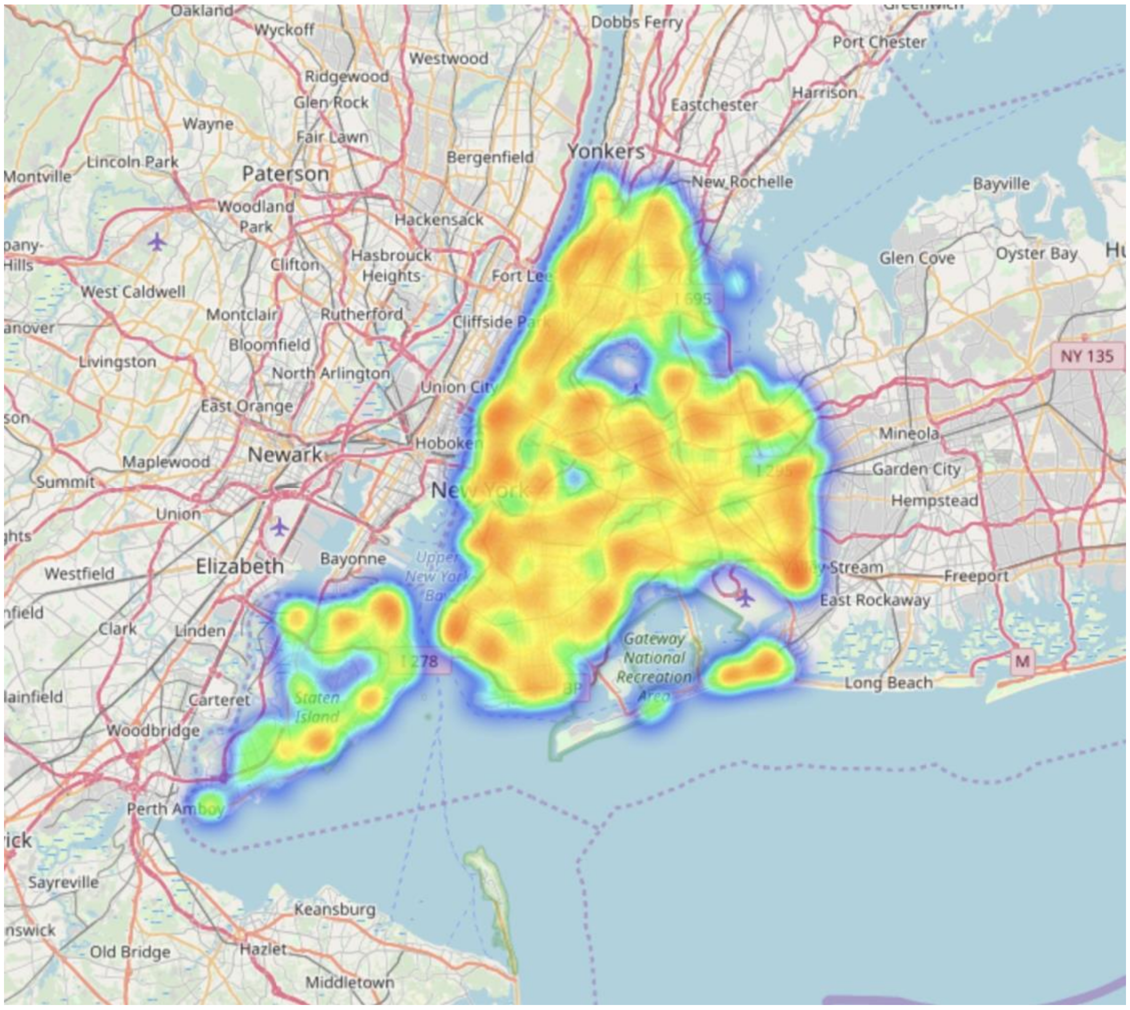
Hierarchical method:



```
cluster = AgglomerativeClustering(n_clusters=10, affinity='euclidean', linkage='ward')
cluster.fit_predict(df)
```

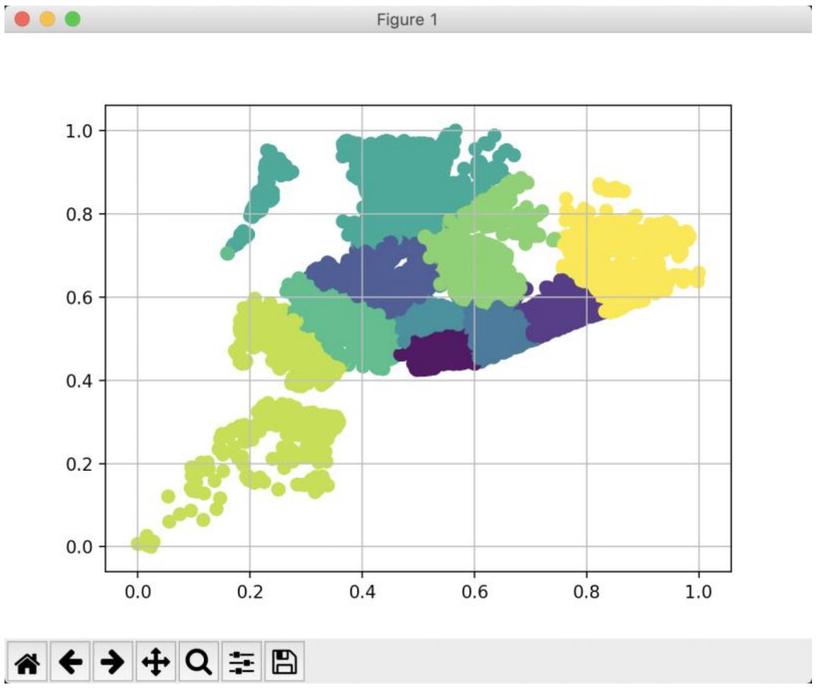
The linkage can be ward, complete, average or single.

Q2.



a. Is this heatmap useful in order to draw conclusions about the expressiveness of areas within NYC? If not, why?

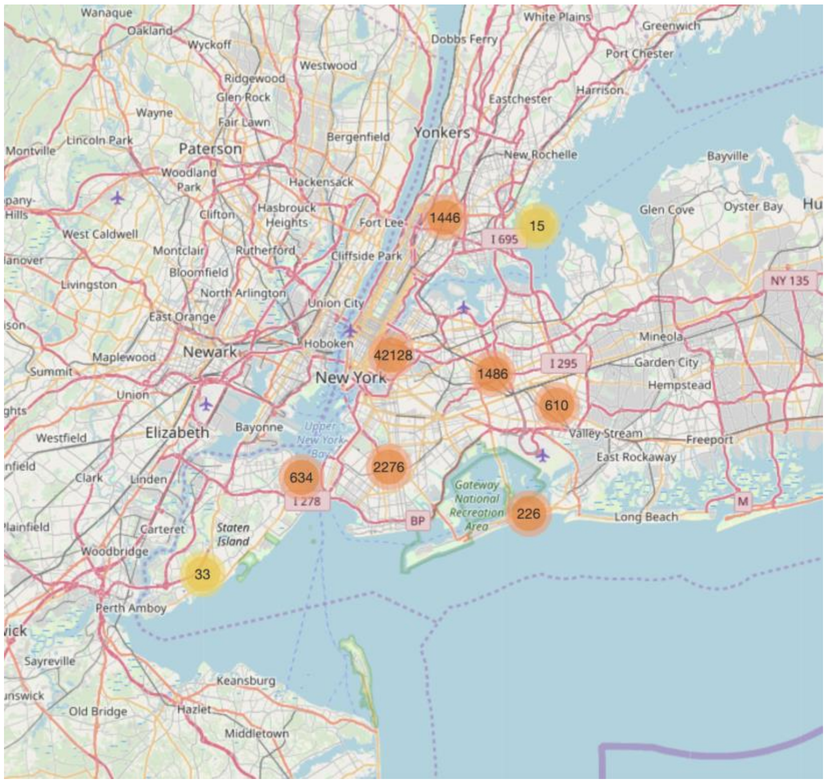
The heatmap is not sufficient to draw conclusions because it only shows preliminary analysis on the Airbnb housing distribution. Price and other detains may not be directly seen from a heatmap.



b.

The clusters in the map with total number of datasets marked on each cluster center point.

c.



e. It agrees with what I know about New York. It would be a little expensive if I choose to live within the city, but on the other hand it would be cheaper to live on the boundary or out of New York city.