Clustering Flats in Berlin

June 29, 2019

1 Clustering Exercise

In this exercise you will use a data set crawled from immobilienscout24.de to cluster flats in Berlin.

First let's load the data and install a package for plotting on a map.

```
In [10]: import pandas as pd

         df_is = pd.read_csv('mietwohnungen.csv').dropna(subset=['lat','lng'], how='any')
         # some data cleaning
         df_is['lat'] = df_is['lat'].str.strip(",").astype(float)
         X = df_is[['lat','lng']]
```

Now let's learn a clustering with sklearn's `MiniBatchKMeans` and compute the cluster assignments for all data points.

```
In [11]: from sklearn.cluster import MiniBatchKMeans
         # we'll look at 12 clusters - feel free to change that
         n_clusters = 12
         # IMPLEMENT YOUR CODE HERE
         # TRAIN A KMEANS MODEL
         km_is = MiniBatchKMeans(n_clusters=n_clusters).fit(X)
         # PREDICT CLUSTERS WITH THAT MODEL HERE
         df_is['cluster_assignment'] = km_is.predict(X)
```

Now let's plot the predictions on the Berlin map.

```
In [12]: # uncomment the next line to install the folium package for plotting maps
         #!pip install folium
         import folium
         from matplotlib import colors as mcolors
         # some dark colors without gray/grey
         colors = [c for c in list(mcolors.CSS4_COLORS.keys()) if 'dark' in c][6:]

         # the initial map, centered at Beuth
         m = folium.Map(location=[52.545195, 13.354670], tiles='Stamen Toner', zoom_start=10)

         # add the flats and the cluster centers to the map
```

```
for cluster_id in range(len(df_is['cluster_assignment'].unique())):
    this_cluster_idx = df_is['cluster_assignment']==cluster_id
    this_cluster_lat_lng = df_is.loc[this_cluster_idx, ['lat','lng']].values
    for lat, lng in this_cluster_lat_lng:
        folium.CircleMarker(
        radius=2,
        location=[lat, lng],
        color=colors[cluster_id],
        fill=False,
        ).add_to(m)

    folium.CircleMarker(
        location=km_is.cluster_centers_[cluster_id,:],
        radius=10,
        color=colors[cluster_id],
        fill=True,
    ).add_to(m)

m.save('flat_clusters.html')

from IPython.display import IFrame

IFrame(src='./flat_clusters.html', width=700, height=600)
```

```
Out[12]: <IPython.lib.display.IFrame at 0x222823257b8>
```

```
In [ ]:
```