Uber Trips

Unsupervised learning

Automatic video captions available.

Workflow

Back to Agenda Page

The Uber Data

.

Visualization

EDA and Streamlit

Prediction Models

Our Dataset

Time data: July 2014 hour per hour uber pick ups

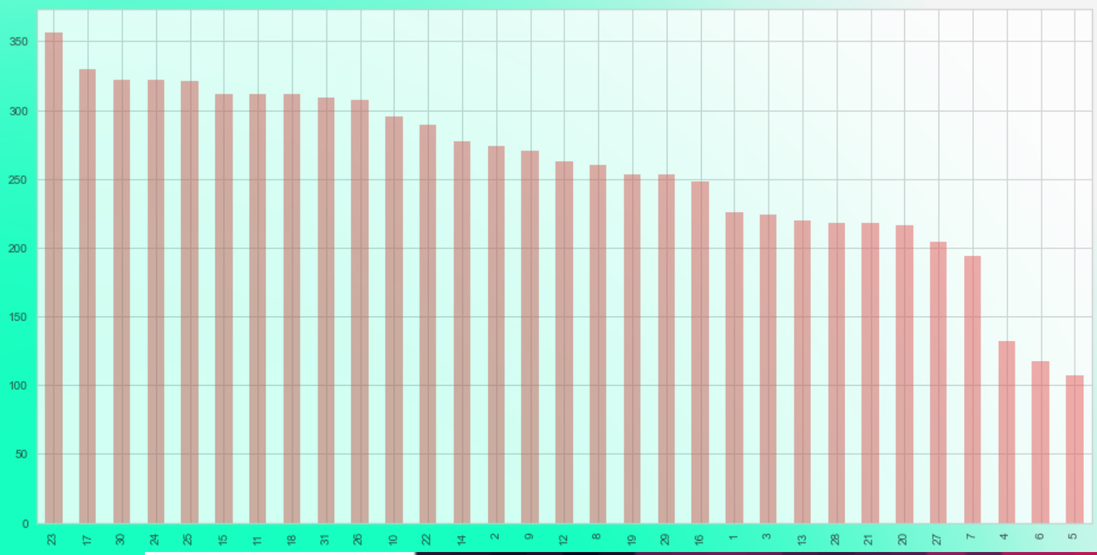
Use Case

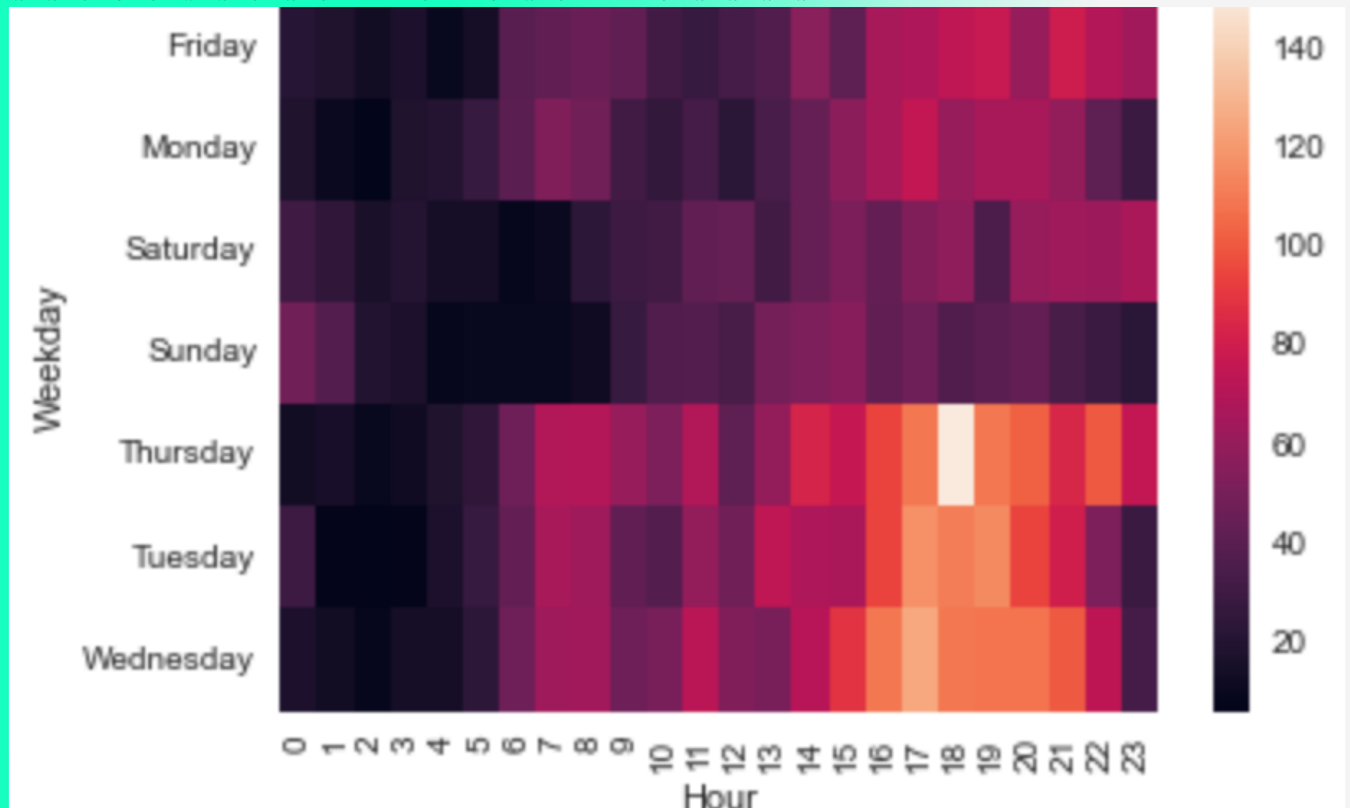
Busiest time for uber drivers

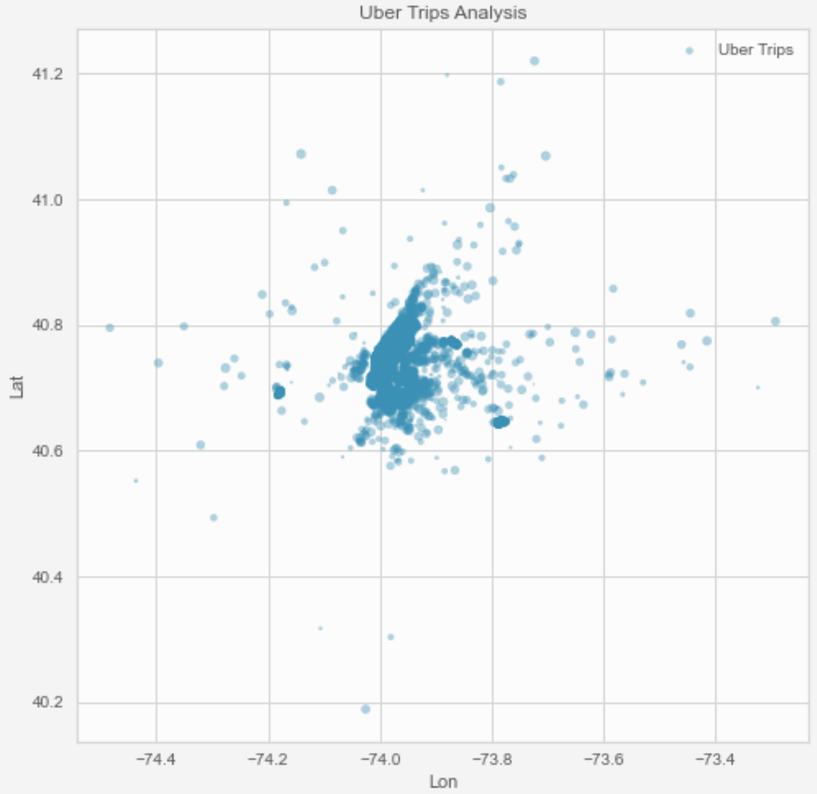
Back to Agenda Page

Sampling

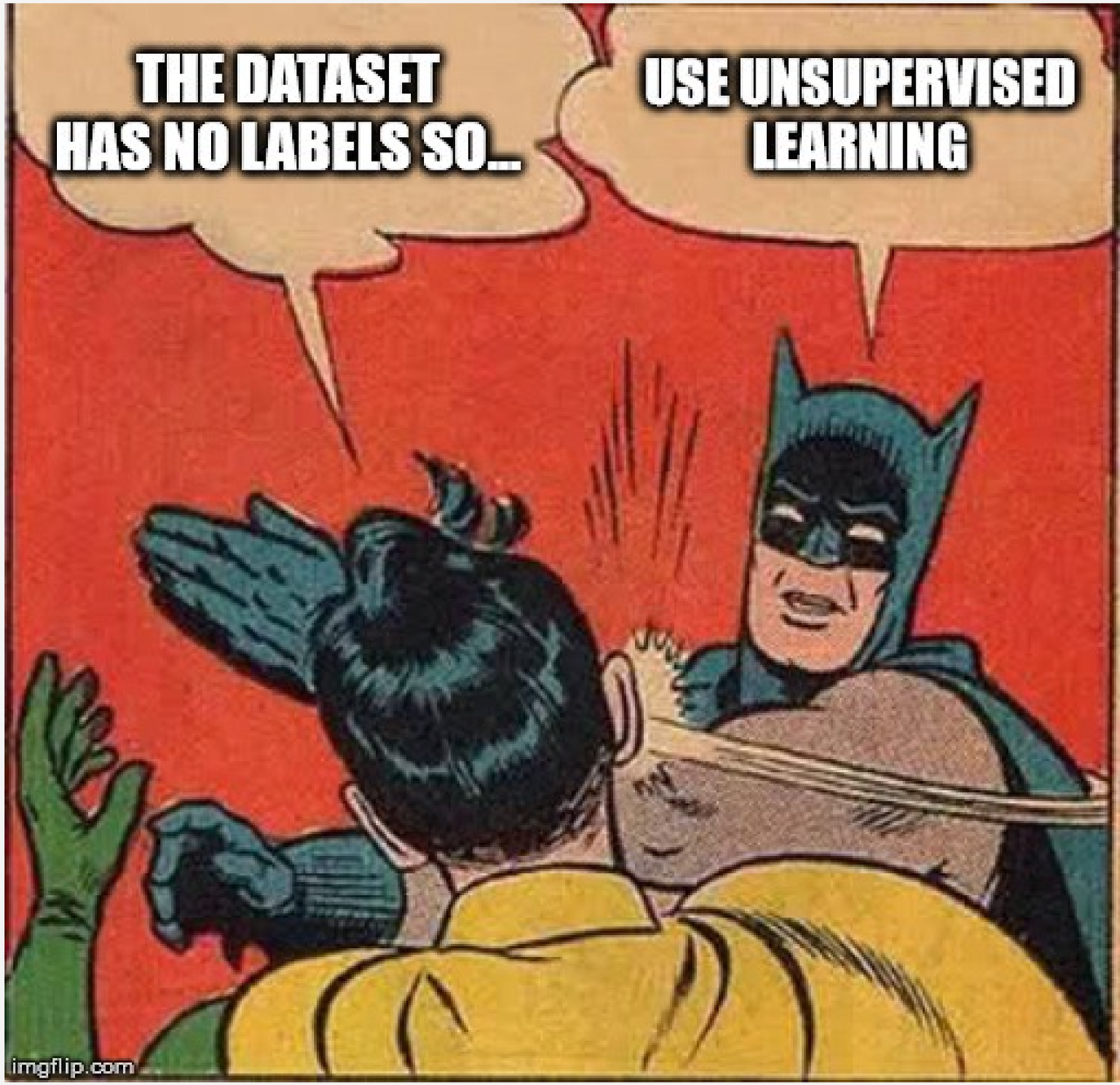
EDA


350
300
250
200
150
100
50
0
23 17 30 24 25 15 11 18 31 26 10 22 14 2 9 12 8 19 29 16 1 3 13 28 21 20 27 7 4 6 5


Friday
Monday
Saturday
Sunday
Thursday
Tuesday
Wednesday
Weekday
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
Hour
140
120
100
80
60
40
20


Uber Trips Analysis
Uber Trips
41.2
41.0
40.8
40.6
40.4
40.2
Lat
−74.4
−74.2
−74.0
−73.8
−73.6
−73.4
Lon

Supervised vs unsupervised Leaning



Back to Agenda Page

SUPERVISED LEARNING

When the training set contains labels (i.e. outputs/target)

Example: Predict the price of an apartment

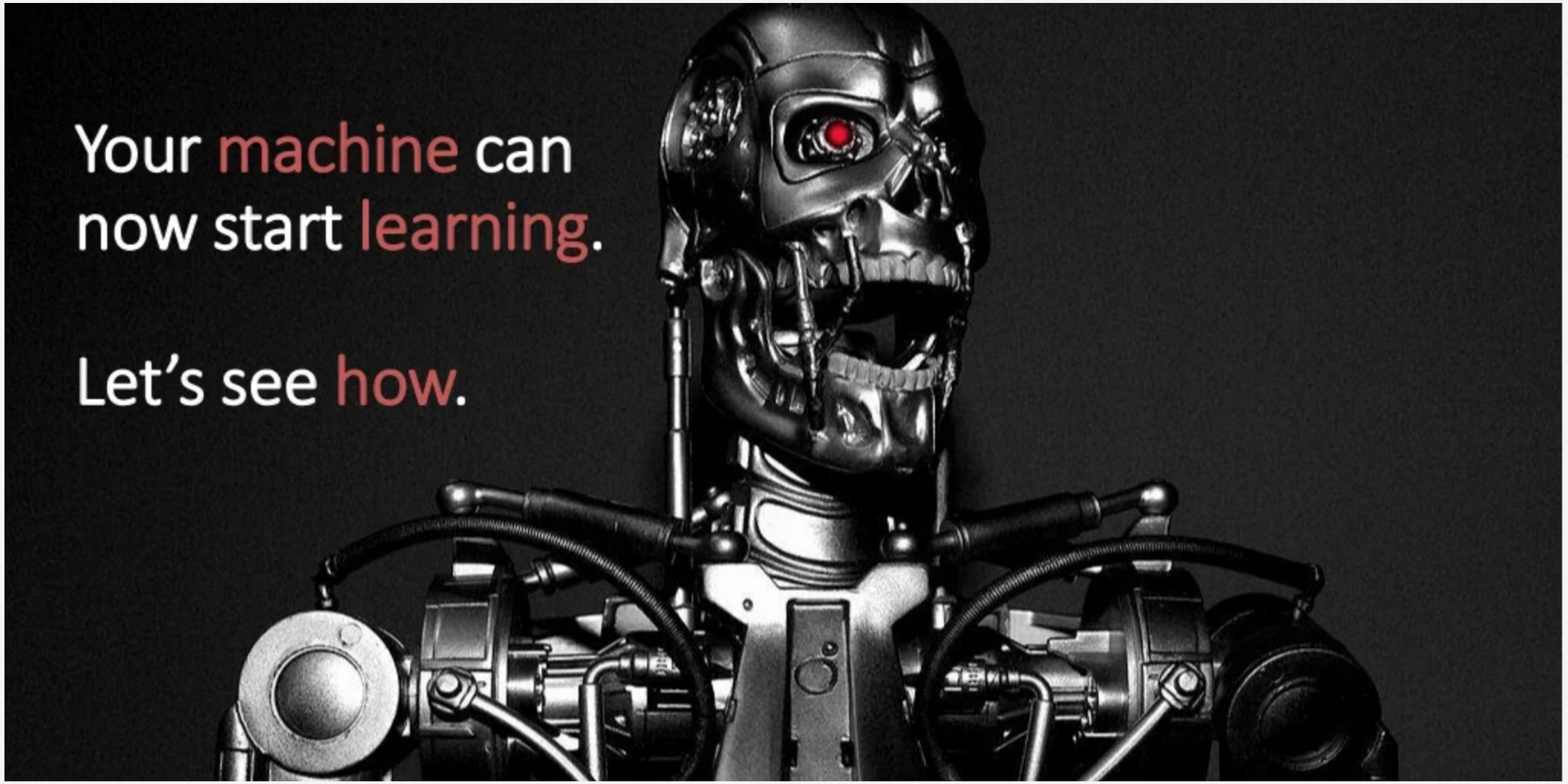
FEATURES					LABEL
Size (m^2)	# rooms	Location	Floor	Elevator	Price (k€)
62	3	Paris	3	Yes	500
92	4	Lyon	4	No	400
43	2	Lille	5	Yes	200

UNSUPERVISED LEARNING

When the training set contains no label, only features

Example: Define client segments within a customer base

FEATURES					LABEL
Name	Gender	Age	Location	Married	
John	M	46	New-York	Yes	
Sarah	F	42	San Francisco	No	
Michael	M	18	Los Angeles	Yes	
Danielle	F	54	Atlanta	Yes	


Your machine can
now start learning.
Let's see how.

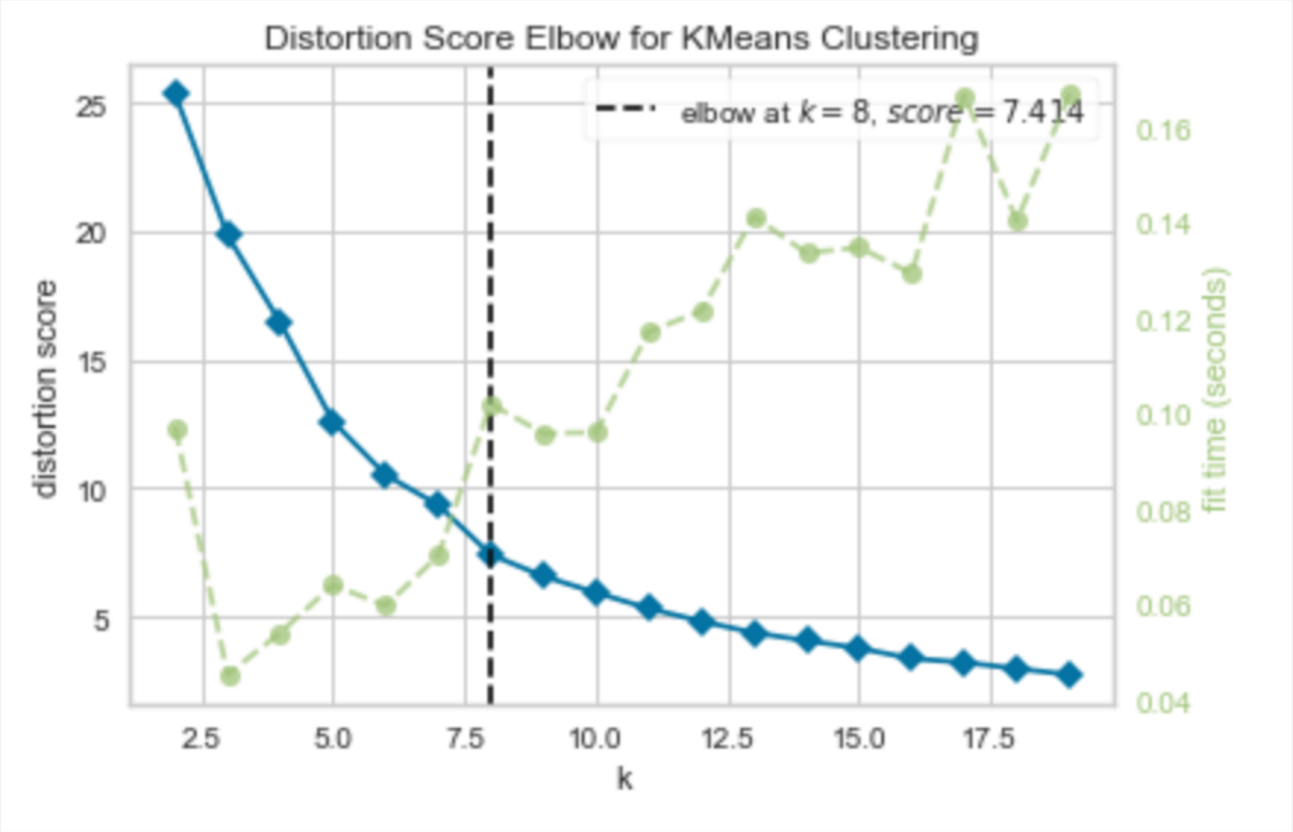
Back to Agenda Page

```
x = uber_sample[['Lat','Lon']]
kmeans = KMeans(n_clusters=7)
labels = kmeans.fit(x)


uber_sample['labels']= labels.labels_
centroids = kmeans.cluster_centers_
centroidss = pd.DataFrame(data = centroids, columns = [['Lat', 'Lon']])
```

Kmeans

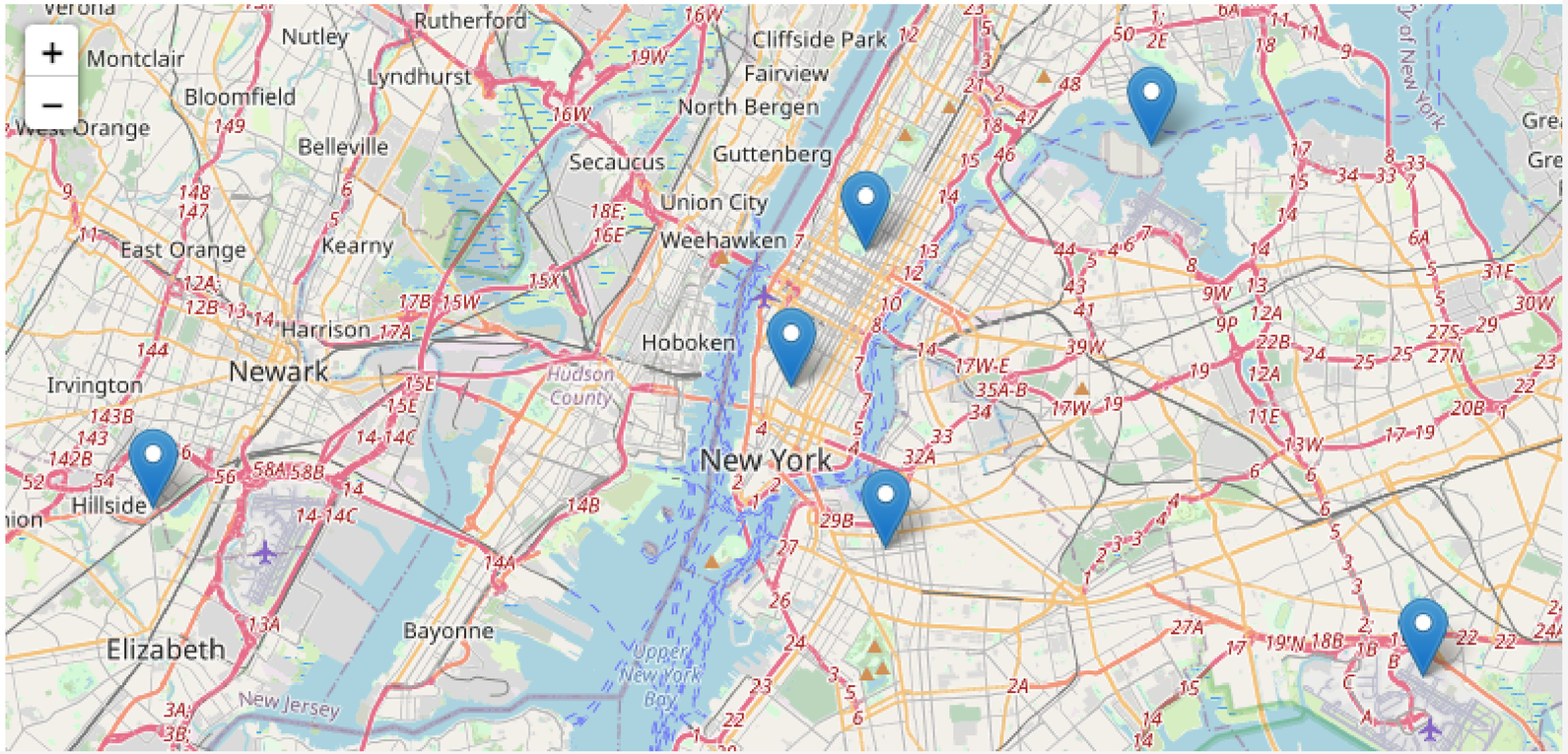
```
x = uber_sample[['Lat', 'Lon']]
model = KMeans()
visualizer = KElbowVisualizer(estimator = model, k = (2,20))
visualizer.fit(x)
visualizer.poof()
```



```
]: centroid = centroidss.values.tolist()

mapp = folium.Map(location=[40.7128, -74.0060], zoom_start=10,tiles = "openstreetmap")
for point in range(0, len(centroid)):
    folium.Marker(centroid[point], popup = centroid[point]).add_to(mapp)

mapp
```



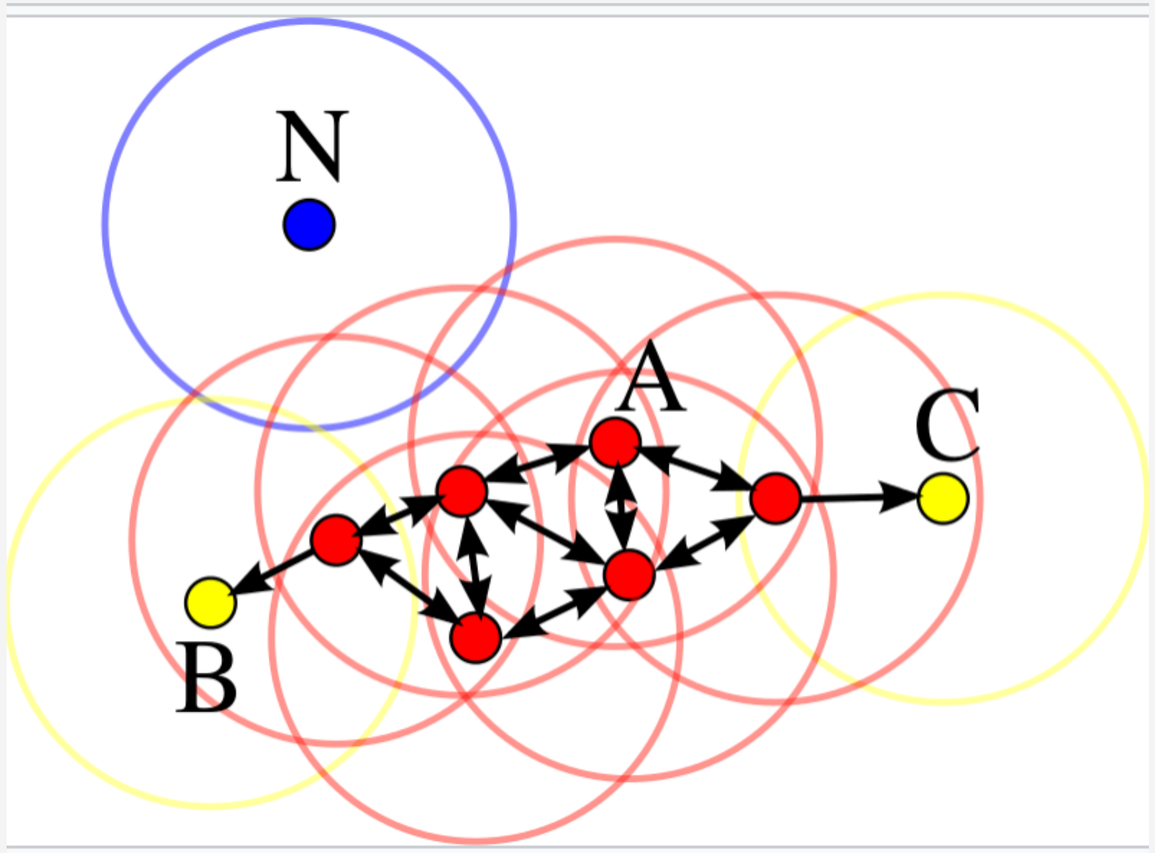
DBSCAN

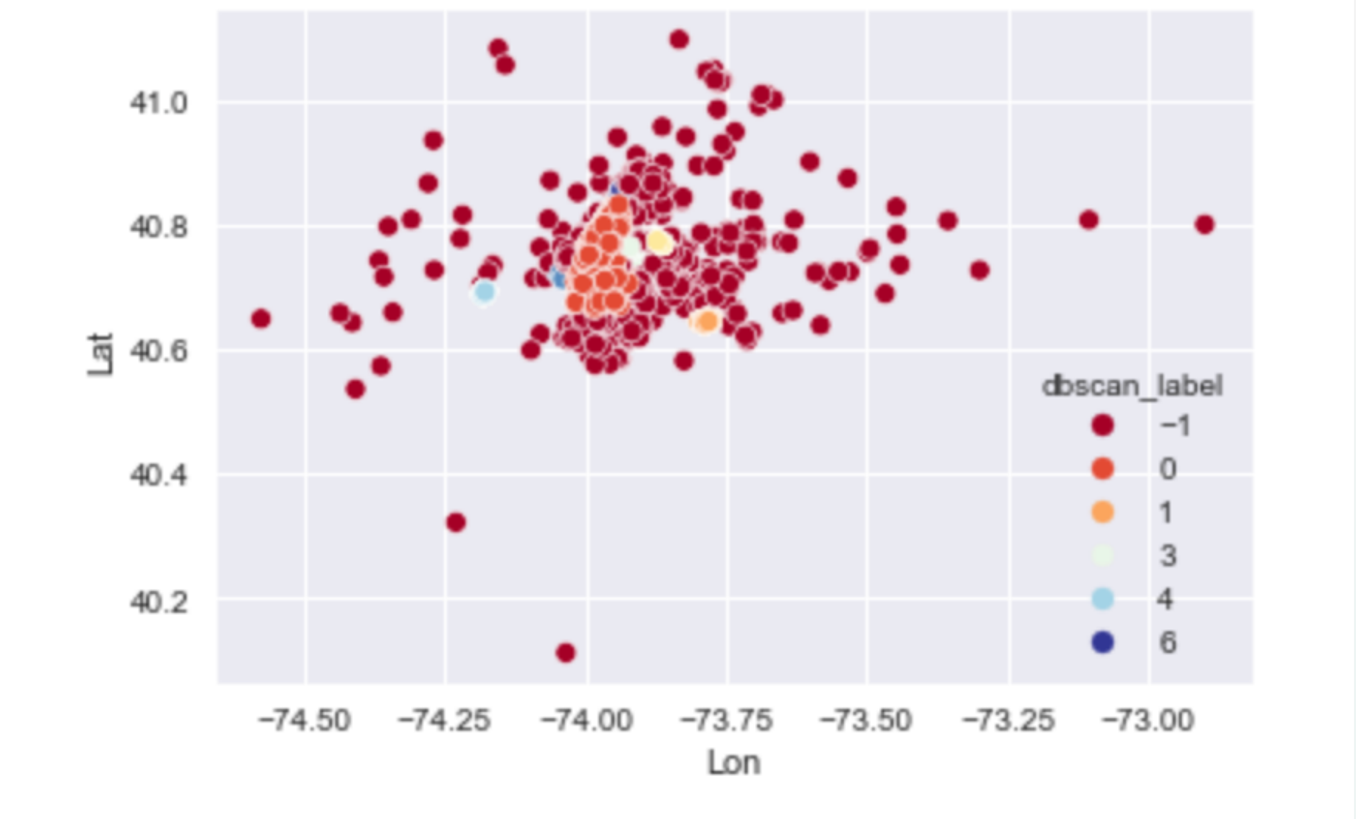
```
dbscan = DBSCAN(eps=0.009, min_samples=20,n_jobs=-1)
dbscan.fit(x)
y_dbscan = dbscan.fit_predict(x)
y_dbscan
```

```
uber_sample['dbscan_label'] =dbscan.labels_

sns.set_style('darkgrid')
sns.scatterplot(data=uber_sample, x='Lon', y='Lat', hue='dbscan_label', palette='RdYlBu')
plt.show()
```

How DBSCAN functions




41.0
40.8
40.6
40.4
40.2
Lat
−74.50
−74.25
−74.00
−73.75
−73.50
−73.25
−73.00
Lon
dbscan_label
−1
0
1
3
4
6

Possible Applications

Marketing: finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records;
Biology: classification of plants and animals given their features;
Insurance: identifying groups of motor insurance policy holders with a high average claim cost; identifying frauds;
Earthquake studies: clustering observed earthquake epicenters to identify dangerous zones;
World Wide Web: document classification; clustering weblog data to discover groups of similar access patterns.

Netflix Prize

COMPLETED

In 2006, Netflix offered a $1m prize for anyone who would **improve the accuracy of their recommendation system by 10%.**

The 2nd team, which achieved a 8.43% improvement, reported **more than 2000 hours of** work to come up with a combination of **107 algorithms!**

The 10% improvement was **only achieved in 2009**, and the algorithm **never went into production…**

Learn more