Final project report

Geo-location Clustering using the k-means Algorithm

By

Hyder Iqbal

Under Guidance

Of

Vahid Behzadan

(Distributed and Scalable Data)

Motivation:

In this project Iam using clustering algorithm in which I Implement k-means in Spark and use it for geo-location clustering. Clustering has many useful Applications such as finding a group of consumers with common preferences, grouping documents based on the similarity of their contents, or finding spatial clusters of customers to improve logistics. I will cluster geo-location data based on the device location. This leads in many aspects of successful business opening in the given geo-location. It not only helps many organizations but also different governments to know the preferences of civilians residing that particular area, which helps to make the market of that preference and boost the economy.

Approach:

1. Uploading data into s3 bucket.

2. Creating EMR cluster on m4.xlarge.

3. Creating a notebook on the cluster.

4. Making a function to read the data using spark context

(Here the problem arises of knowing the delimiters. Every function every file has different implementation to take care of delimiters).

5. Making the spark data frame to store the data using `spark.createDataFrame.`

Then filtering the data out locations that have a latitude and longitude of 0.

6. Uploading parsed file to s3 bucket using `rdd.coalesce ().`

7. Then calculating k-means on all the data sequentially.

8. Calculating Euclidean distance and great circle distance

9. Visualization the graph after every file processing.

System Configuration:

Before creating this cluster we have to make key-pair pem file. I used m4.xlarge emr cluster which gives 1 master and 2 slave nodes by default. In the configuration please select SPARK instead of Hadoop.

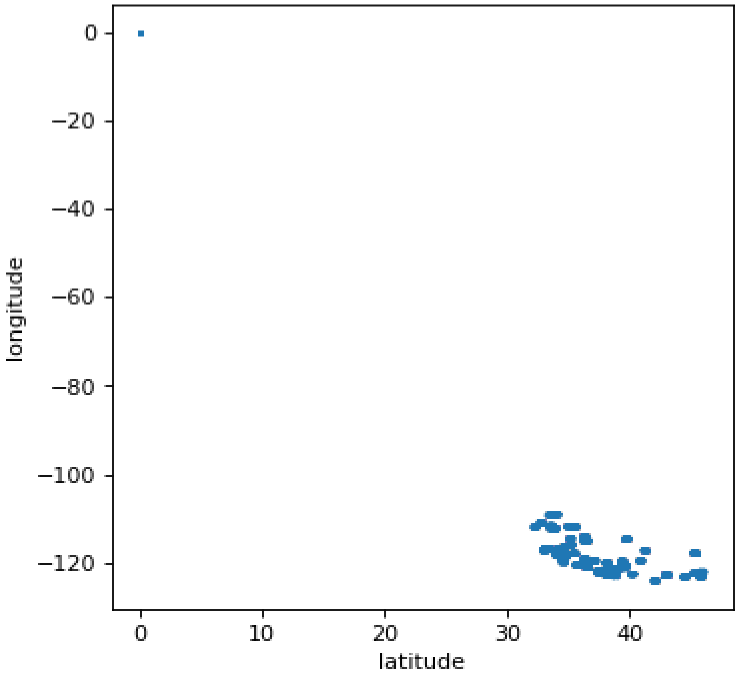
Then connecting it with ssh terminal and there we have to install different packages then after connecting with Jupyter it give address which helps in getting to Jupyter notebook.

Big data application/dataset:

K-means clustering is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. We successfully implemented k-means algorithm with k=2,4,5,6 (k gives the no. of clusters to be made). In this step we can analyse different task completed through graphs.

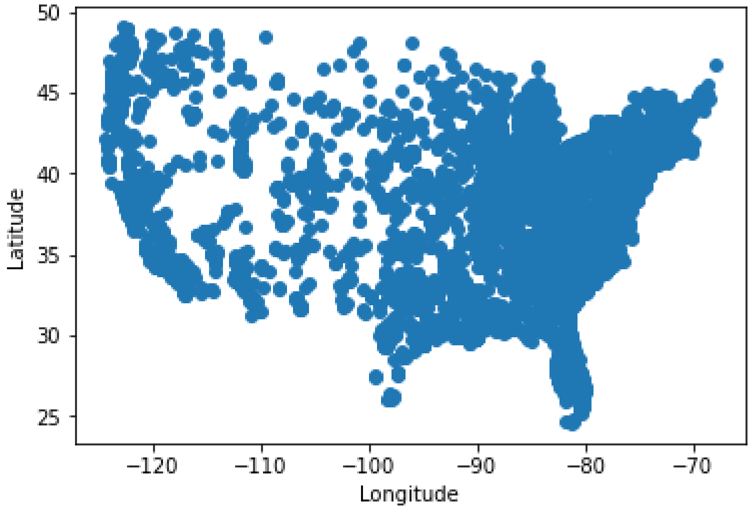
File wise graphs:

1) Analysing the devicestatus file using longitude and latitude



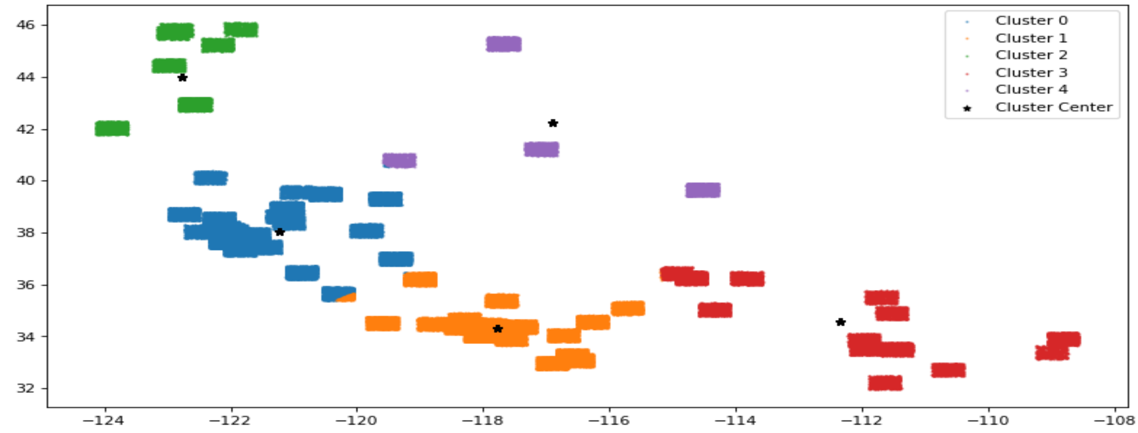
2) Analysing sample_geo text file

```
Text(0, 0.5, 'Latitude')
```

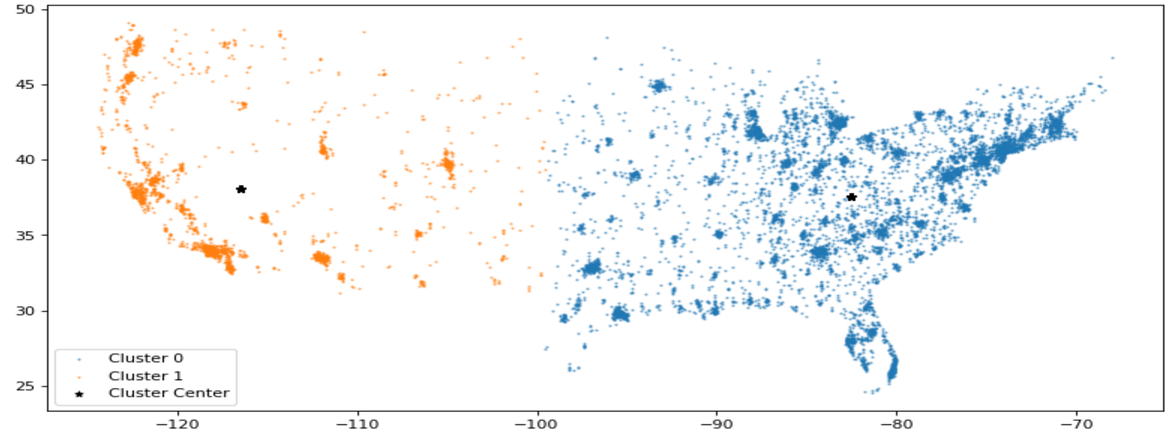


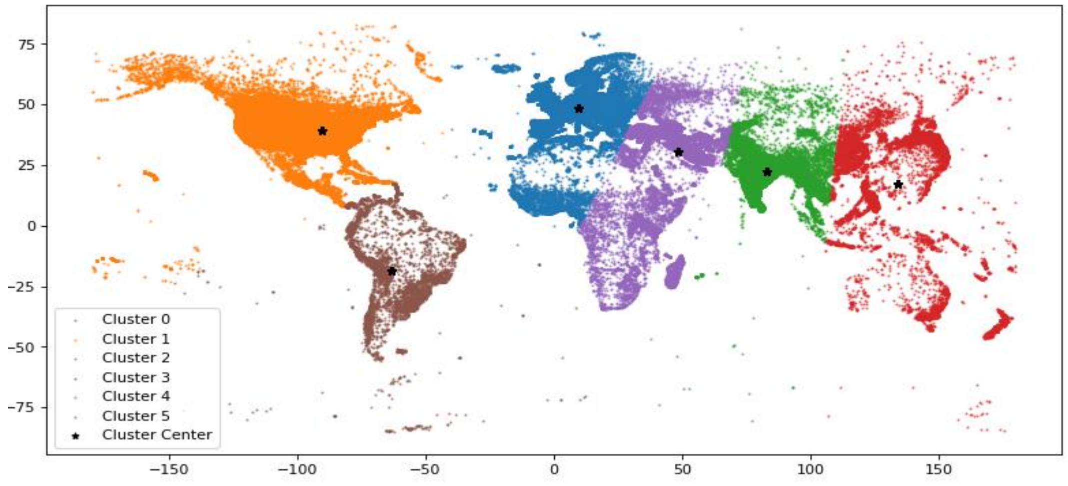
Below is the visuals of different no. of clusters:

```
<matplotlib.legend.Legend at 0x7f9976257f28>
```



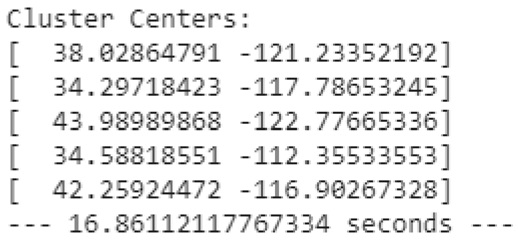
```
<matplotlib.legend.Legend at 0x7f9979b152e8>
```





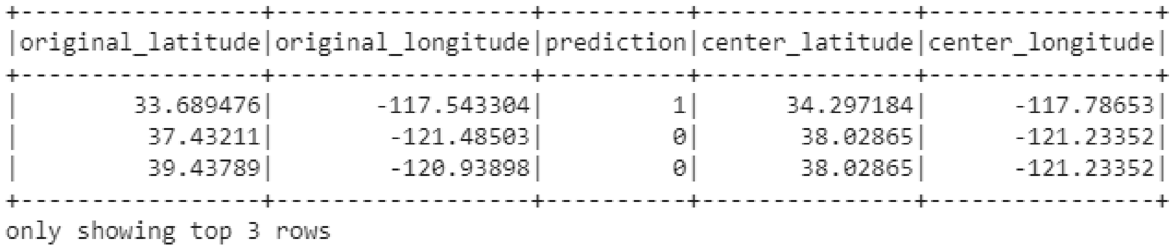
RESULTS:

Clusters Centre



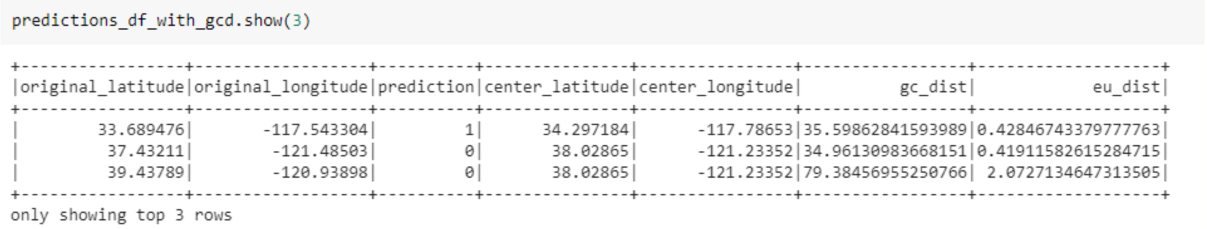
```
Cluster Centers:
[  38.02864791 -121.23352192]
[  34.29718423 -117.78653245]
[  43.98989868 -122.77665336]
[  34.58818551 -112.35533553]
[  42.25924472 -116.90267328]
--- 16.86112117767334 seconds ---
```

Spark data-frame after parsing the data



```
+-----------------+------------------+----------+---------------+----------------+
|original_latitude|original_longitude|prediction|center_latitude|center_longitude|
+-----------------+------------------+----------+---------------+----------------+
|        33.689476|       -117.543304|         1|      34.297184|      -117.78653|
|         37.43211|        -121.48503|         0|       38.02865|      -121.23352|
|         39.43789|        -120.93898|         0|       38.02865|      -121.23352|
+-----------------+------------------+----------+---------------+----------------+
only showing top 3 rows
```

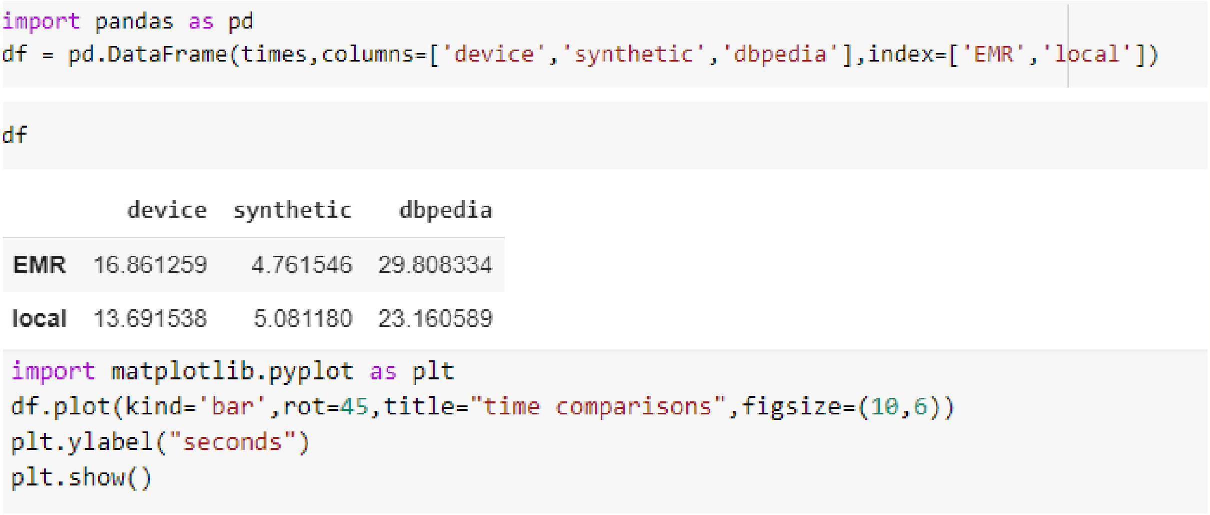
Spark data-frame after calculating Euclidean and GCD:



```
predictions_df_with_gcd.show(3)
```

```
+-----------------+------------------+----------+---------------+----------------+-----------------+------------------+
|original_latitude|original_longitude|prediction|center_latitude|center_longitude|          gc_dist|           eu_dist|
+-----------------+------------------+----------+---------------+----------------+-----------------+------------------+
|        33.689476|       -117.543304|         1|      34.297184|      -117.78653|35.59862841593989|0.4284674337977763|
|         37.43211|        -121.48503|         0|       38.02865|      -121.23352|34.96130983668151|0.4191158261528471|
|         39.43789|        -120.93898|         0|       38.02865|      -121.23352|79.38456955250766| 2.072713464731350|
+-----------------+------------------+----------+---------------+----------------+-----------------+------------------+
only showing top 3 rows
```

Runtime analysis:

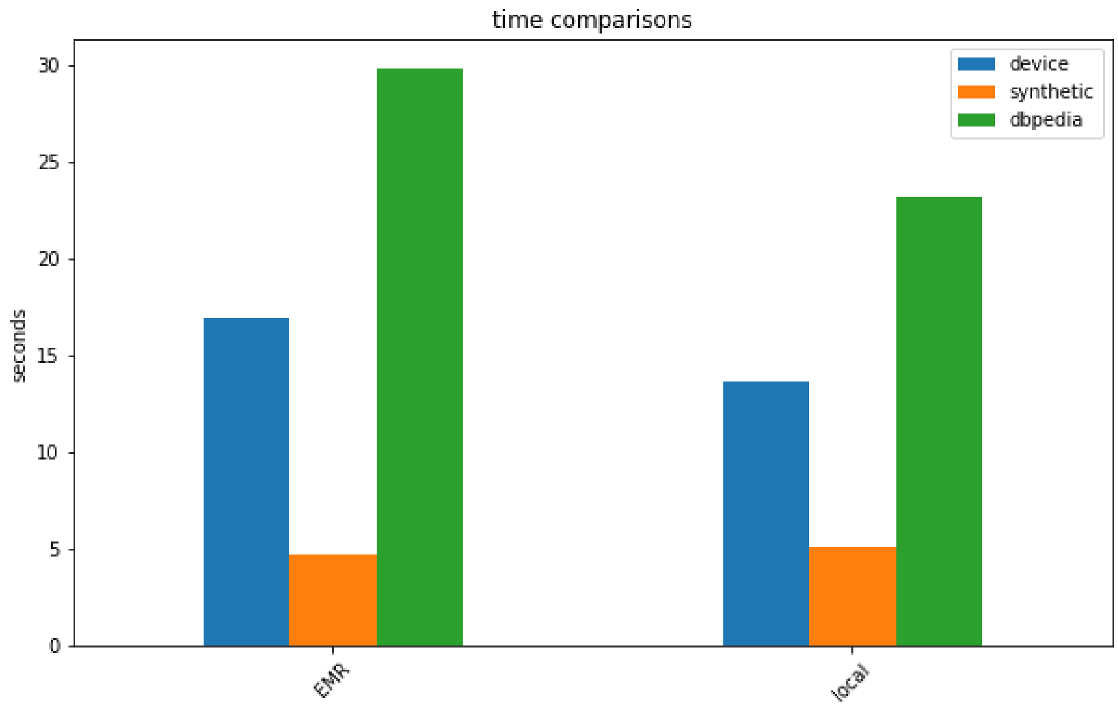


```python
import pandas as pd
df = pd.DataFrame(times,columns=['device','synthetic','dbpedia'],index=['EMR','local'])
```

```python
df
```

	device	synthetic	dbpedia
EMR	16.861259	4.761546	29.808334
local	13.691538	5.081180	23.160589

```python
import matplotlib.pyplot as plt
df.plot(kind='bar',rot=45,title="time comparisons",figsize=(10,6))
plt.ylabel("seconds")
plt.show()
```



CONCLUSION/FUTURE WORKS

This project helped to know the geo-locations of devices under the service of Loudacre's network in which different part of the word. We can generate data from other networks and some other features (eg- malicious activities) to detect the activities. Apart from that it can also be used to detect densely populated areas. The more data the more work can be done. But however we have to be careful in handling the data and its usage including its privacy.