DSCI 6007 - Distributed & Scalable Data Engineering Final Project

Geo-location Clustering using *k*-means in Spark

May 11, 2020

Shivani Mathur, University of New Haven

I. Project Motivation

In this project we will use SPARK to implement *k*-means algorithm that solves the clustering problem in an efficient distributed fashion. Spark is a unified analytics engine for large-scale data processing. It provides an optimized engine that supports general computation graphs for data analysis. Spark is widely used for fast processing of Big Data. It is not only limited to interactive queries and data analysis, but can also be handy with visualization and Machine Learning. In this project we will get hands on PySpark and will get familiarised with the concepts of RDDs (Resilient Distributed Datasets).

Another motivational factor for this project has been the implementation of k-means algorithm from scratch. *k*-means group the similar data points together and discover underlying patterns.

To achieve this objective, *k*-means looks for a fixed number (*k*) of clusters in a dataset. To process the dataset, *k*-means algorithm starts with the first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative calculations to optimize the positions of the centroids. It halts creating and optimizing clusters when either there is no change in centoriod values because the clustering has been successful or a defined number of iterations has been achieved.

Data visualisation, as we use and learn through this project, is one of the most important skills for a Data Science. Data visualizations makes big or small data easier to understand, and visualization also makes it easier to detect patterns, trends, and outliers in groups of data. Representation of the result of *k*-mean algorithm with help in better understanding of the dataset and further analysis.

II. Approach

As instructed, I reviewed the Clustering and *k*-means algorithm from *MMDS chapter-7 infolab.stanford.edu* and other available sources. Also, I explored the use of PySpark for data analysis. My knowledge and working experience on AWS, EC2, EMR and S3 from the course and previous assignment helped me with the configuration of clusters. Before implementing into the algorithm, I delved into retrieving various files from s3 and working with RDDs on Jupyter Notebook.

III. System Configuration

- Created a S3 (Simple Storage Service) bucket, EC2 (Elastic Compute Cloud) instance and EMR (Elastic MapReduce) cluster on AWS.
- Connected to the instance using ssh.
- Installed Python, Jupyter, Spark, Basemap and other packegs to be used in the project.
- Provided AWS CLI credentials to establish connection from Jupyter to AWS.

IV. Dataset Description

Datasets used in this project:

- **Device Location** Data is the information collected from mobile devices on Loudacre's network, including device ID, current status, location and so on. These data records have different formats in that they use different delimiters to separate the fields. These data only contain the location information of western USA.

- **Synthetic Location** Data is the dataset of the whole USA, and it only contains the latitude, longitude and location ID. The records are tab delimited.

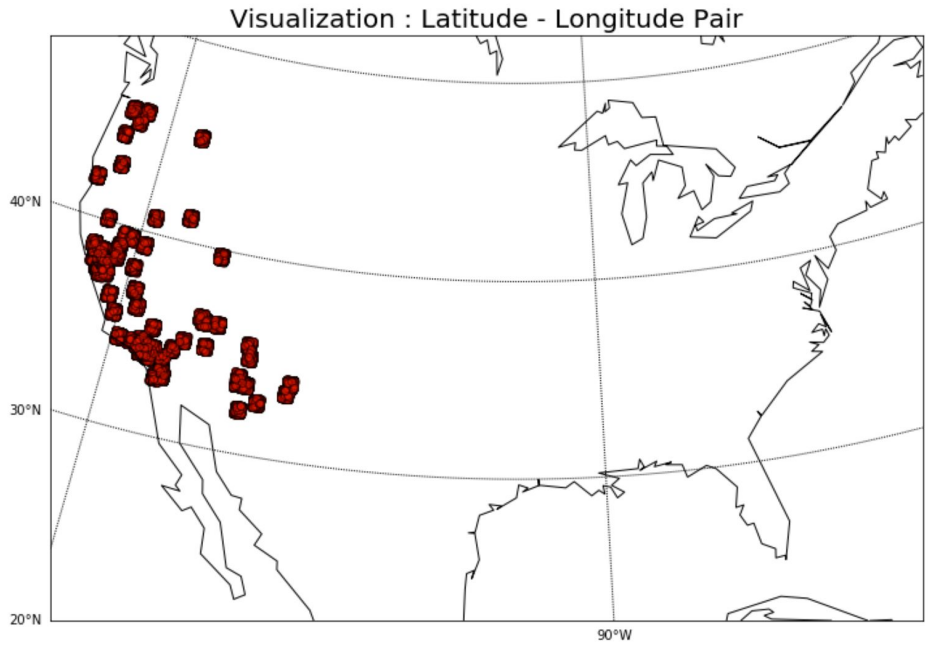
- **DBpedia** Location Data is the large-scale clustering data extracted from DB-pedia. Each record represents a location or place with a Wikipedia link and latitude, longitude information. The records are space delimited.

We did not perform any pre-processing on the Synthetic Location Data and DBpedia Location data, but we did pre-processing on the Device Location Data. Each record was separated using three different delimiters and those which did not parsed correctly were filtered out. Date, model, device ID and latitude and longitude were extracted from the original 14 featured dataset and model was separated into manufacturer and model Id. The csv file was then uploaded to S3 bucket for future use.

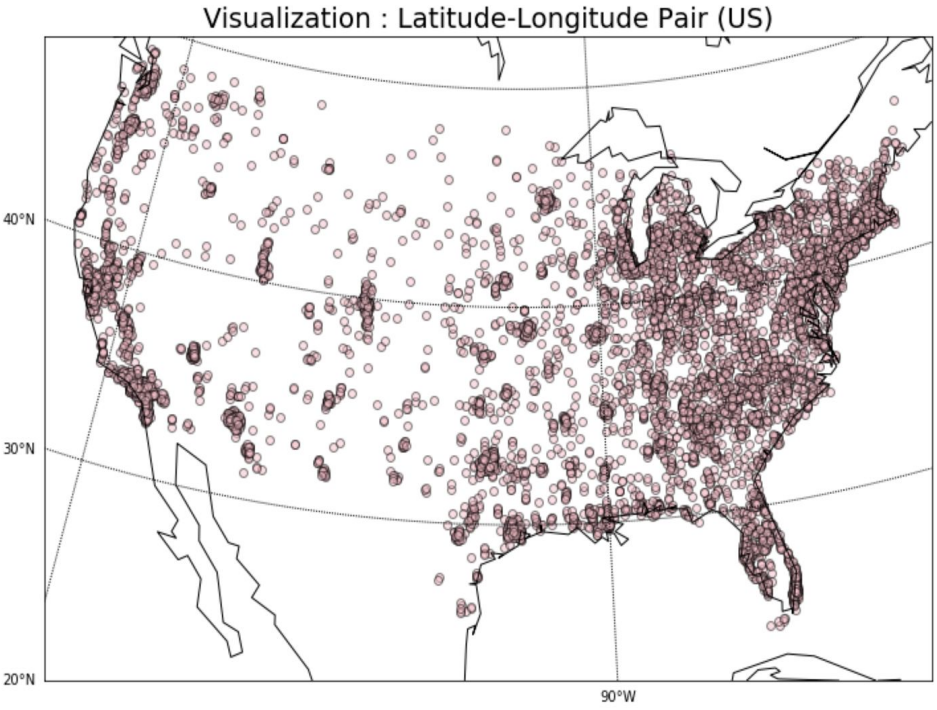
Number of records in each dataset:

- Device Location Data: 431,857
- Synthetic Location Data: 9,970
- DBpedia Location Data: 450,151

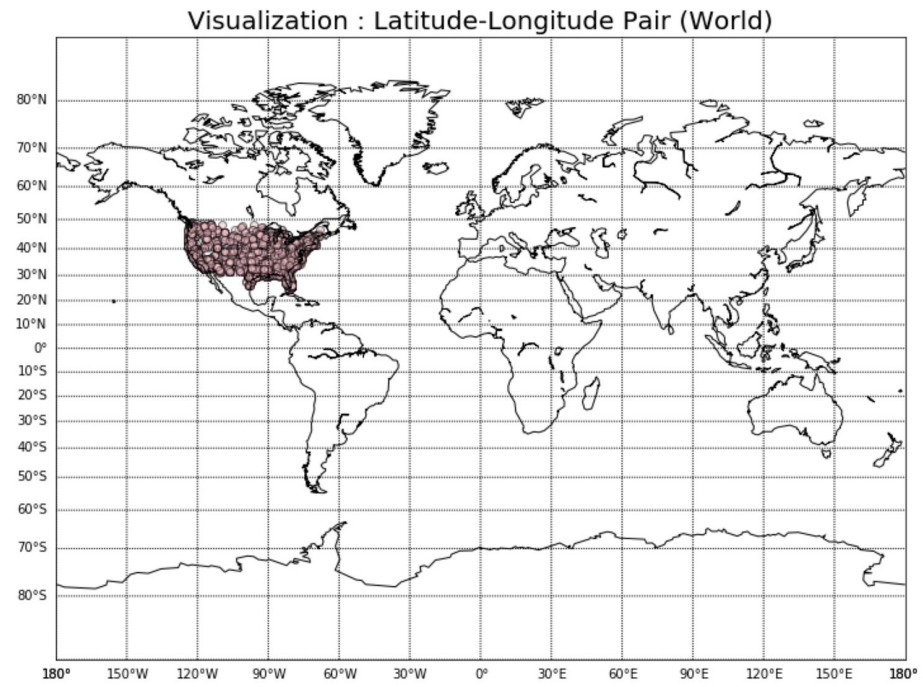
Visualisation of Latitude-Longitude pair for each dataset :



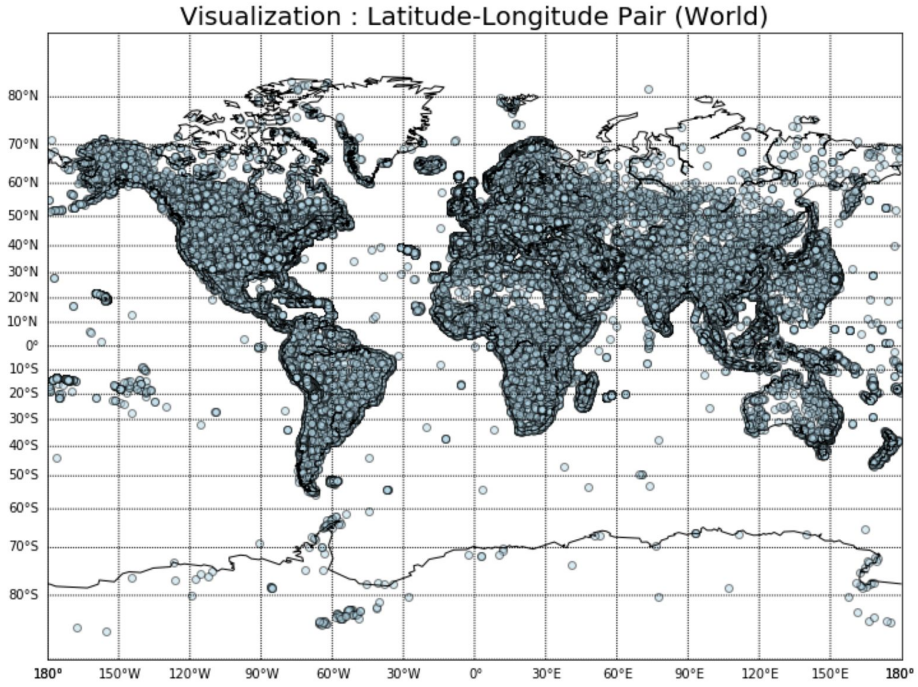
Latitude-Longitude pair for Device Data



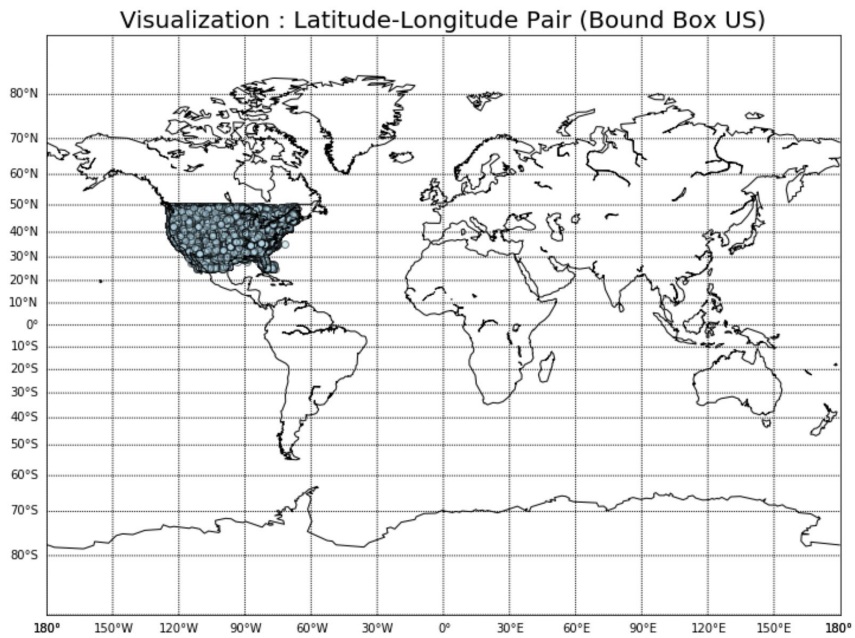
Latitude-Longitude pair for Synthetic Data



Latitude-Longitude pair for Synthetic Data



Latitude-Longitude pair for DBpediaData



Latitude-Longitude pair for DBpediaData : Bounding Box US

V. Implementation

Two distance measurement methodologies that have been used to implement *k*-means in this project :

Euclidean Distance - The Euclidean distance between two points in either the plane or 3-dimensional space measures the length of a segment connecting the two points.

$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

Great Circle Distance - It is the shortest distance between two points on the surface of a sphere, measured along the surface of the sphere (as opposed to a straight line through the sphere's interior).

$$\begin{aligned}
\Delta X &= \cos\phi_2 \cos\lambda_2 - \cos\phi_1 \cos\lambda_1; \\
\Delta Y &= \cos\phi_2 \sin\lambda_2 - \cos\phi_1 \sin\lambda_1; \\
\Delta Z &= \sin\phi_2 - \sin\phi_1; \\
C &= \sqrt{(\Delta X)^2 + (\Delta Y)^2 + (\Delta Z)^2}
\end{aligned}$$

***K*-means Algorithm** is implemented using following steps -

1. Randomly select *'c/k'* cluster centers.
2. Calculate the distance between each data point and cluster centers.
3. Assign the data points to the cluster center whose distance from the cluster center is the minimum of all the cluster centers..
4. Recalculate the new cluster center using:

$$v_i = (1/c_i) \sum_{j=1}^{c_i} x_i$$

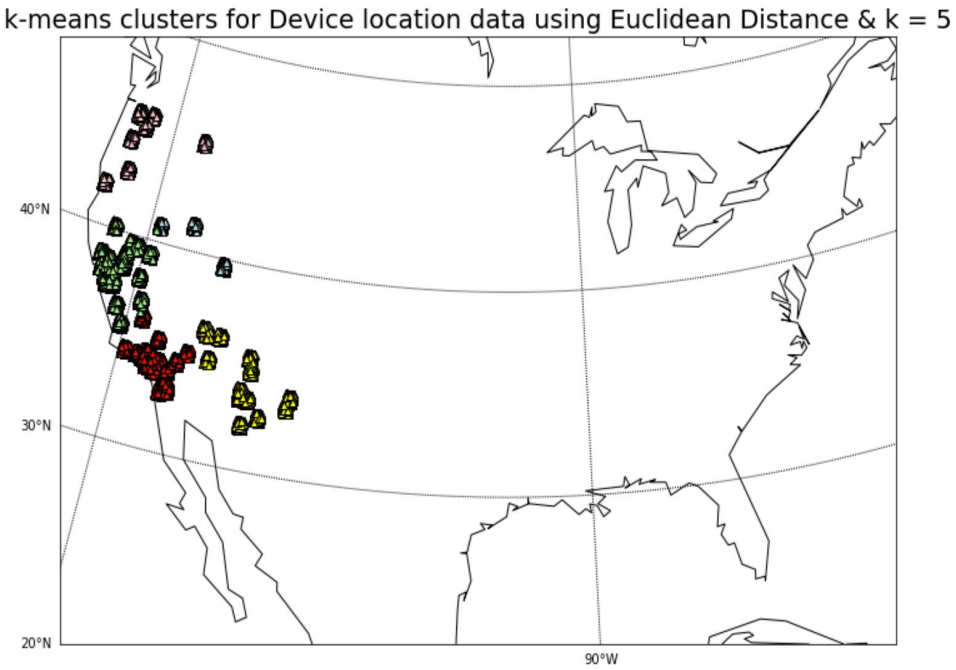
where, 'c_i' represents the number of data points in i^{th} cluster.

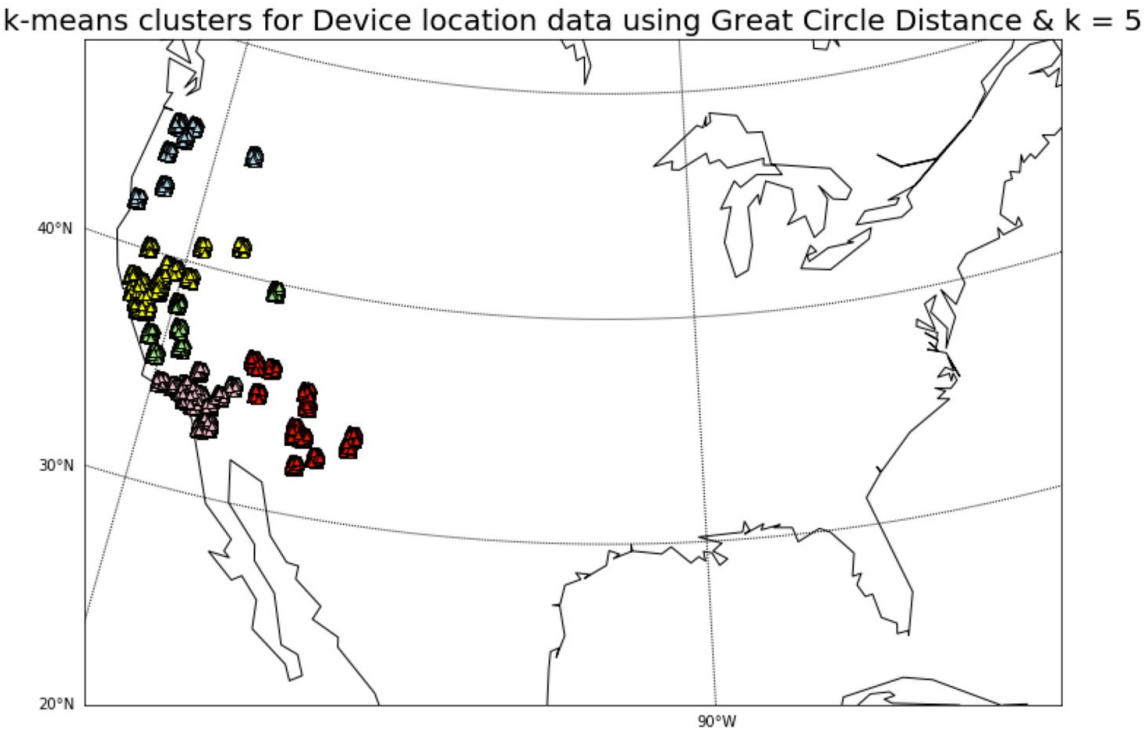
5. Recalculate the distance between each data point and new obtained cluster centers.
6. Calculate the average distance difference between the old centers and the new centers, using the distance measurement specified by the user;
7. If the mean distance difference is larger than converge distance, repeat 2-6

The output format of the program will be a comma delimited file with records (latitude, longitude, cluster number).

VI. Results

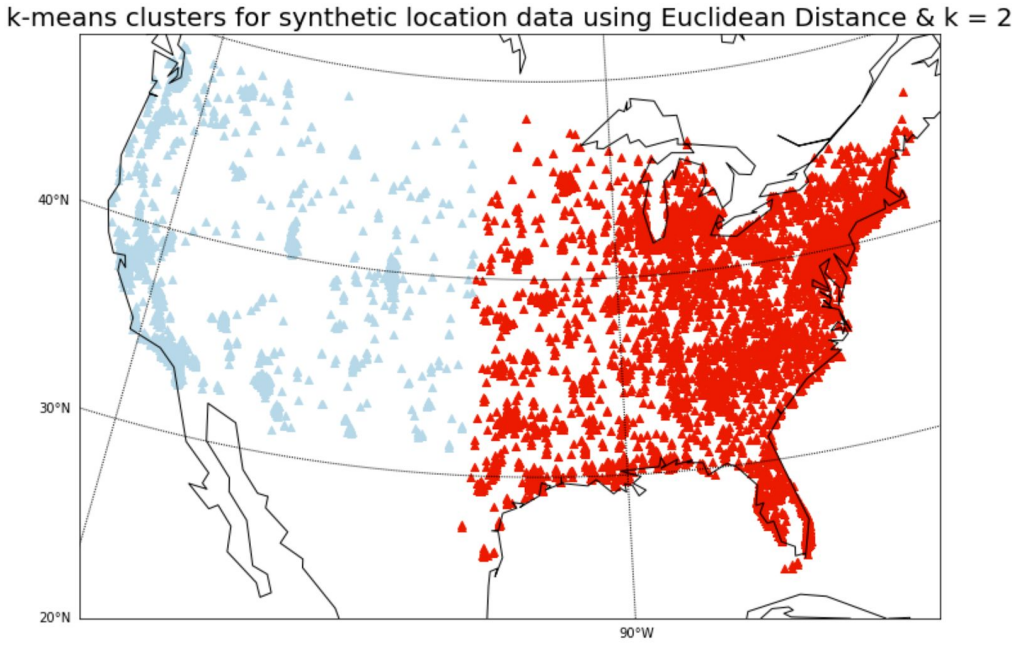
Device Location data (*k* = 5) :

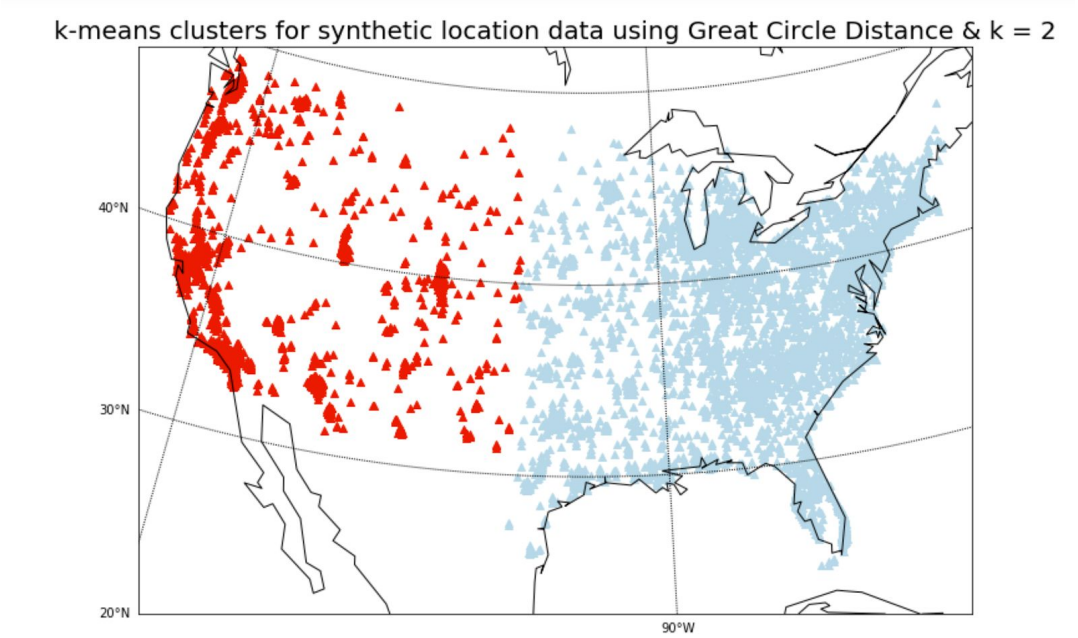




We can observe that the using Device Data with k = 5, results from Euclidean Distance and Great Circle Distance is on the same lines. Except for a few outlier points which were chosen in different clusters in both the measures.

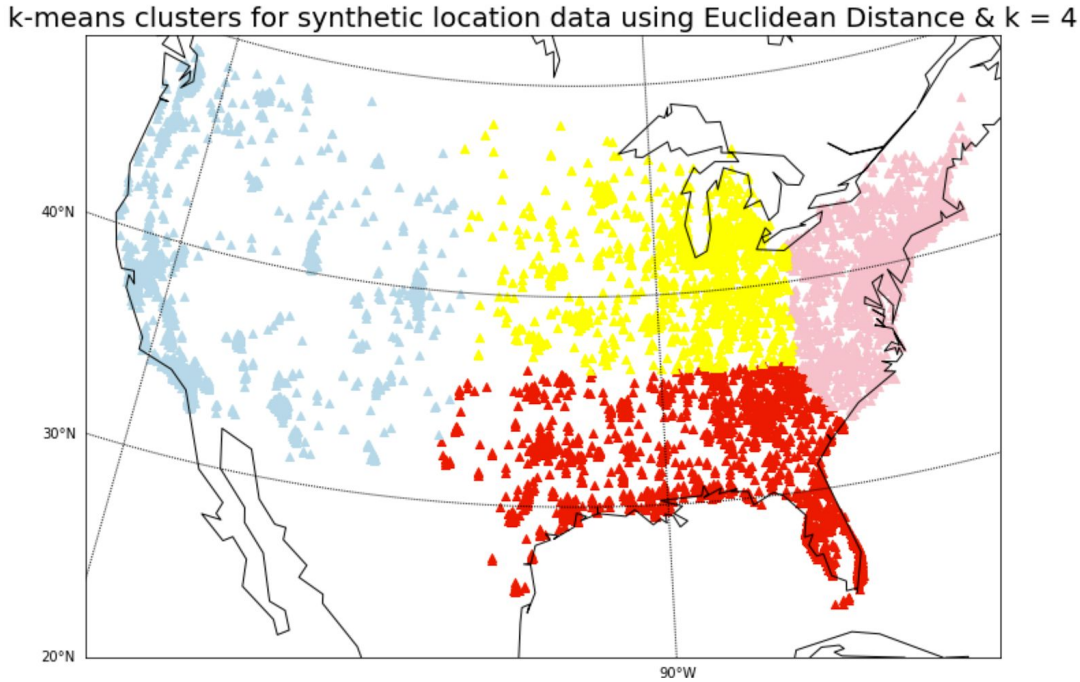
Synthetic Location Data (k = 2) :

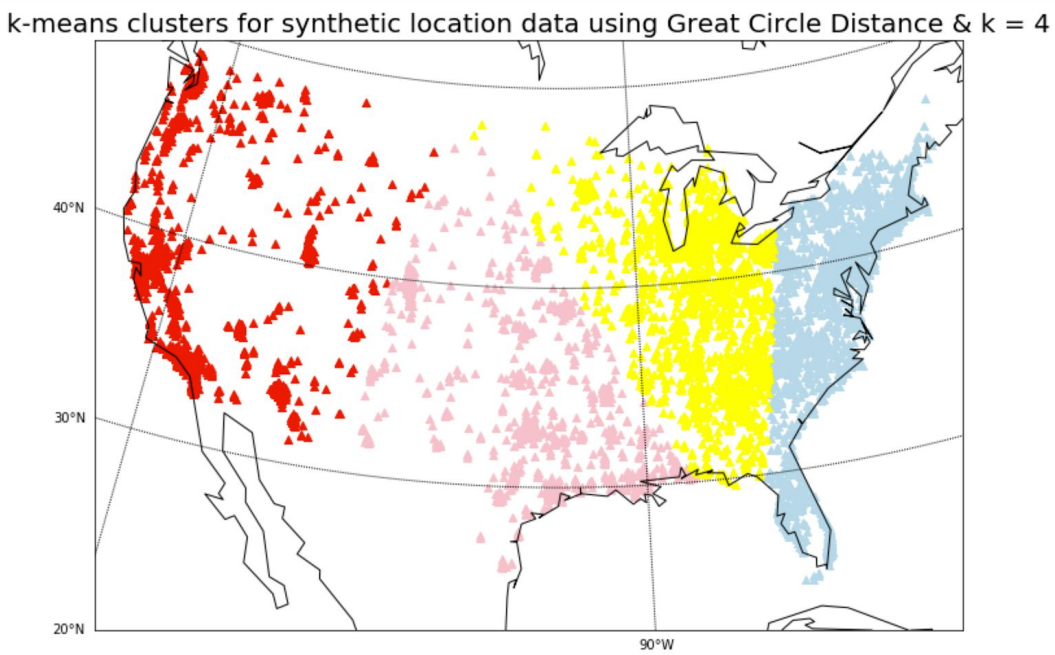




Here we can observe that the using Synthetic Data with k = 2, results from Euclidean Distance and Great Circle Distance is exactly the same.

Synthetic Location Data (k = 4) :





Using Synthetic Data with $k = 4$, results from Euclidean Distance and Great Circle Distance has shown a lot of variance. Clusters using GCD show the points clustered along the surface of the sphere.

VII. Conclusion

- For the same Device location dataset, with "*k* = 5", the two distance calculation measure, generated a very similar result, with exception of a view outlying points.
- For the Synthetic location dataset, if we have "*k* = 2", the two distance calculation measures generated practically the same results. However, for "*k* = 4" the resulting clusters are a lot different.
- We can conclude that while dealing with clustering of a Latitude Longitude dataset through *k*-means, for smaller values of *k,* such as 2 or 3, both the distance measures will produce a similar result, however, for higher the values of clusters or *k*, Great Circle Distance measure should produce more logical result since our earth is circular (spherical).