Geo-Location Clustering with K-Mean

By:
Praveen Mandadi

Introduction and Motivation

Clustering is a process of grouping a set of data points into clusters so that points that are put within the same cluster are like each other whereas points from different clusters are dissimilar. Clustering has many useful applications for marketing, logistics, and document classification. We clustered geo-location data, which will be naturally visualized.

We implemented the k-means algorithm in Spark to solve the clustering issue in a parallel fashion. The algorithm iteratively updates the position of k cluster centroids as a distance-based approach until the shift in the mean of centroids converges to alpha=0.1 km where alpha is conve rged.

Data Pre-Processing

We went through pre-processing steps before implementing the real algorithm to translate the data for later processing into a standardized format. The pre-processing process for device status data is listed below.

A. Load the dataset.
B. Determine which delimiter to use.
C. Filter out any records which do not parse correctly; each record should have exactly 14 values.
D. Extract the date, model, device ID, and latitude and longitude.
    I. date: 1st field
    II. model: 2nd field
    III. device ID: 3rd field
    IV. latitude: 13th field
    V. longitude: 14th field
E. Store latitude and longitude as the first two fields
F. Filter out locations that have a latitude and longitude of 0.
G. Split the model field that contains the device manufacturer and model name by spaces.
H. Save the extracted data as comma separated values file in the s3://datageo1/results/devicedata.csv directory on AWS S3.
I. Confirm the data in the file(s) was saved correctly.

Device Status dataset

```
In [16]: #Printing the data frame after droping zeros
         devicedata
```

Out[16]:

	latitude	longitude	date	model	device ID
0	33.6894754264	-117.543308253	2014-03-15:10:10:20	F41L	8cc3b47e-bd01-4482-b500-28f2342679af
1	39.3635186767	-119.400334708	2014-03-15:10:10:20	F41L	707daba1-5640-4d60-a6d9-1d6fa0645be0
2	33.1913581092	-116.448242643	2014-03-15:10:10:20	Novelty Note 1	db66fe81-aa55-43b4-9418-fc6e7a00f891
3	33.8343543748	-117.330000857	2014-03-15:10:10:20	F41L	ffa18088-69a0-433e-84b8-006b2b9cc1d0
4	37.3803954321	-121.840756755	2014-03-15:10:10:20	F33L	66d678e6-9c87-48d2-a415-8d5035e54a23
...	...	...	...	...	...
65640	39.4463417571	-114.736213453	2014-03-15:10:49:30	F22L	40e61459-5448-4dc9-bb89-42e73a4e19cf
65641	38.4282665514	-121.25933863	2014-03-15:10:49:30	S2	b13ece99-62ab-4c9f-a366-6a06bd5e877f
65642	33.7778202246	-108.575470704	2014-03-15:10:49:30	F41L	32af1a0b-ca7f-4906-9772-9eb9435e7e4c
65643	38.2596913494	-122.295712621	2014-03-15:10:49:30	S1	a48a5559-d916-481b-84a9-5dce6272cce1
65644	34.2415255221	-118.23526739	2014-03-15:10:49:30	2	d86fbaa6-b71b-435f-a0bf-5304a202a70b

65645 rows × 5 columns

Synthetic Dataset

```
In [26]: #Displaying Pandas Data Frame
         pandas_df.head()
```

Out[26]:

	Latitude	Longitude	LocationID
0	37.77253945	-77.49954987	1
1	42.09013298	-87.68915558	2
2	39.56341754	-75.58753204	3
3	39.45302347	-87.69374084	4
4	38.9537989	-77.01656342	5

DBPedia Dataset

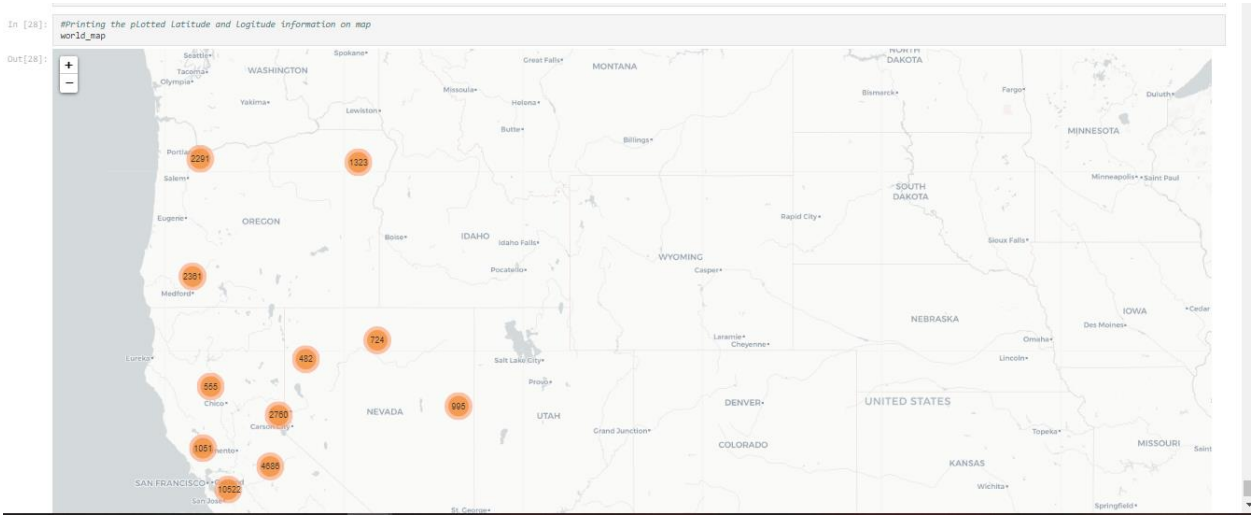
```
In [24]: #Showing pandas data frame
         pandas_df.head()
```

Out[24]:

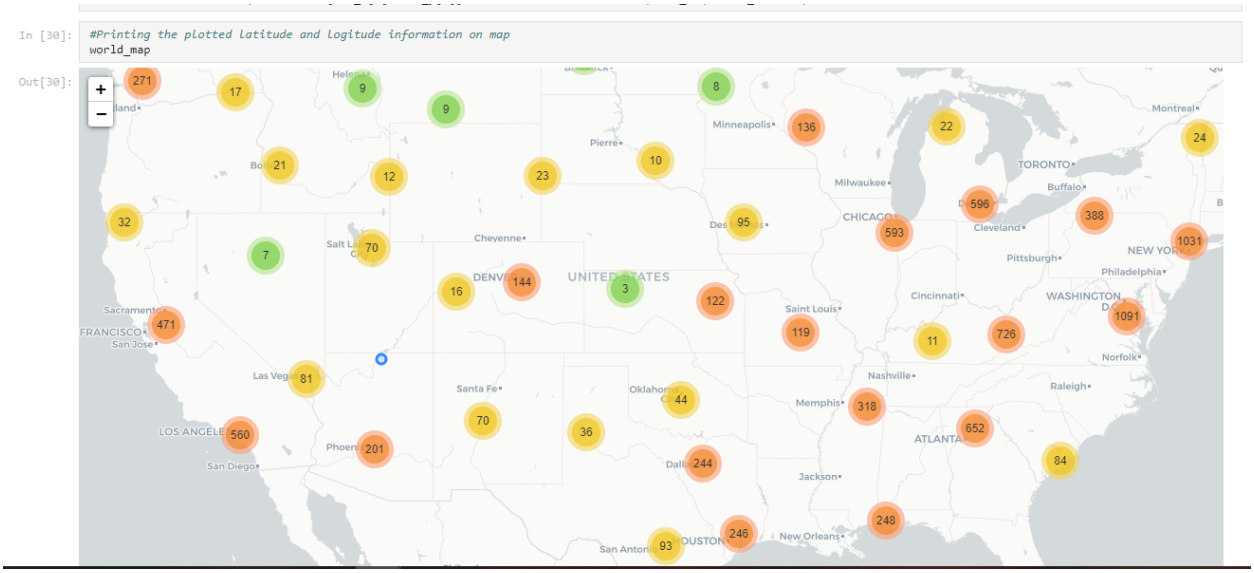
	lat	long	name_of_page
0	36.7	3.216666666666667	<http://dbpedia.org/resource/Algeria>
1	42.5	1.5166666666666666	<http://dbpedia.org/resource/Andorra>
2	12.516666666666667	-70.03333333333333	<http://dbpedia.org/resource/Aruba>
3	-8.833333333333334	13.333333333333334	<http://dbpedia.org/resource/Angola>
4	41.333333333333336	19.8	<http://dbpedia.org/resource/Albania>

Visualization

Mobile Net data



Synthetic Location Data



DBPedia Location Data

Clustering Approach

In the dataset, the initial centroids are a k-sized random sample of all items. The algorithm assigns each point to its nearest centroid for each iteration, then calculates the new centroids by taking the average of all points in the cluster of that centroid. Using either Euclidean distance or Great Circle distance, the distance between points and centroids is determined - the user sets this parameter. The key difference between the two measurements is that the former measures a straight-line distance between the points in 3D space, while the latter measures the distance around the spherical surface of the Planet.

Though a “perfect” algorithm would iterate until the change in centroid locations converges to 0, this algorithm continues iterating until the sum of all changes in centroid locations converges to α=0.1 km. That is, the algorithm calculates the distance between the current position of each centroid and the former location (using the user-specified distance measure) for each iteration and proceeds to iterate until the sum of these distances is less than 0.1 km for all centroids. This requires larger values of k to converge more precisely than smaller values. Because data this algorithm runs on covers at least an entire continent, we determined that the alpha of 0.1 km or 100 m was a small enough value for this purpose.