Ukrainian Catholic University

Applied Sciences Faculty

Final project: Distributed K-means

Distributed Algorithms

Project report

Author: Dmytro Maliarenko

maliarenko@ucu.edu.ua

github link:
https://github.com/arxitekton/DistributedAlgorithms/tree/master/TriangleCounting_v2/src/main/java/kmeans

1 Introduction

K-MEANS: A CENTROID-BASED CLUSTERING TECHNIQUE

This partitioning clustering is most popular and fundamental technique [1]. It is vastly used clustering technique which requires user specified parameters like number of clusters k, cluster initialisation and cluster metric [4].

First it needs to define initial clusters which makes subsets (or groups) of nearest points (from centroid) inside the data set and these subsets (or groups) called clusters [1]. Secondly, it finds means value for each cluster and define new centroid to allocate data points to this new centroid and this iterative process will goes on until centroid [5] does not changes. The simplest algorithm for the traditional Kmeans [4] is as follows;

Input: D = {d1, d2, d3,.....dn} // set of n numbers of data points

K // The number of desire Clusters

Output: A set of k clusters [4]

1. Select k points as initial centroids.
2. Repeat
3. From K clusters by assigning each data point to its nearest centroid.
4. Recompute the centroid for each cluster until centroid does not change.

However, the algorithm has its own pros and cons, which is as follows;

PROs:

1. It is relatively faster clustering technique [1].
2. It works fast with the Large data set since the time complexity is O(nkl) where n is the number of patterns, k is the number of clusters and l is the number of the iterations [5, 6].
3. It relies on Euclidian distance which makes it works well with numeric values with interesting geometrical and statistic meaning [6].

CONs:

1. The initial assumption of value for K is very important but there isn't any proper description on assuming value for K and hence for different values for K will generate the different numbers of clusters [6].
2. The initial cluster centroids are very important but if the centroid is far from the cluster center of the data itself then it results into infinite iterations which sometimes lead to incorrect clustering [6].
3. The K-means clustering is not good enough with clustering data set with noise [6].

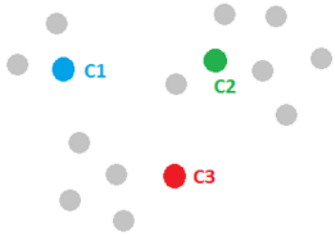
2 Steps of algorithm:

K-means follows the four steps listed below [7].

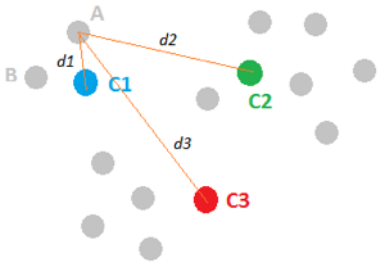
We have next points:

Step one: Initialize cluster centers

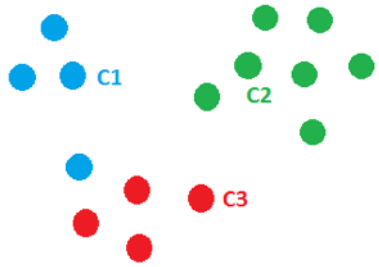
We randomly pick three points C1, C2 and C3, and label them with blue, green and red color separately to represent the cluster centers.



Step two: Assign observations to the closest cluster center

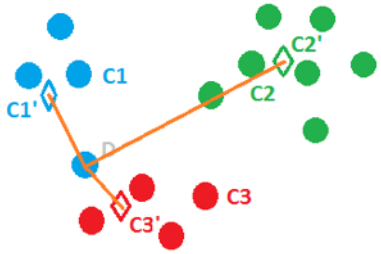


Once we have these cluster centers, we can assign each point to the clusters based on the minimum distance to the cluster center. For the gray point A, compute its distance to C1, C2 and C3, respectively. And after comparing the lengths of d1, d2 and d3, we figure out that d1 is the smallest, therefore, we assign point A to the blue cluster and label it with blue. We then move to point B and follow the same procedure. This process can assign all the points and leads to the following figure.



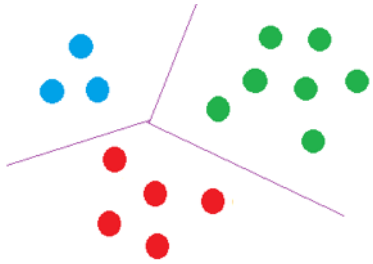
Step three: Revise cluster centers as mean of assigned observations

Now we've assigned all the points based on which cluster center they were closest to. Next, we need to update the cluster centers based on the points assigned to them. For instance, we can find the center mass of the blue cluster by summing over all the blue points and dividing by the total number of points, which is four here. And the resulted center mass C1', represented by a blue diamond, is our new center for the blue cluster. Similarly, we can find the new centers C2' and C3' for the green and red clusters.



Step four: Repeat step 2 and step 3 until convergence

The last step of k-means is just to repeat the above two steps. For example, in this case, once C1', C2' and C3' are assigned as the new cluster centers, point D becomes closer to C3' and thus can be assigned to the red cluster. We keep on iterating between assigning points to cluster centers, and updating the cluster centers until convergence. Finally, we may get a solution like the following figure. Well done!



Some Additional Remarks about K-means

The k-means algorithm converges to local optimum. Therefore, the result found by K-means is not necessarily the most optimal one.

The initialization of the centers is critical to the quality of the solution found. There is a smarter initialization method called K-means++ that provides a more reliable solution for clustering.

The user has to select the number of clusters ahead of time.

3 Data

Source: Kaggle (https://www.kaggle.com/c/nyc-taxi-trip-duration/data) from Competition: New York City Taxi Trip Duration

Data:

The dataset is based on the 2016 NYC Yellow Cab trip record data made available in Big Query on Google Cloud Platform. The data was originally published by the NYC Taxi and Limousine Commission (TLC). The data was sampled and cleaned for the purposes of this playground competition. Based on individual trip attributes, participants should predict the duration of each trip in the test set.

File descriptions:

train.csv - the training set (contains 1458644 trip records)

Data fields:

- id - a unique identifier for each trip
- vendor_id - a code indicating the provider associated with the trip record
- pickup_datetime - date and time when the meter was engaged
- dropoff_datetime - date and time when the meter was disengaged
- passenger_count - the number of passengers in the vehicle (driver entered value)
- pickup_longitude - the longitude where the meter was engaged
- pickup_latitude - the latitude where the meter was engaged
- dropoff_longitude - the longitude where the meter was disengaged
- dropoff_latitude - the latitude where the meter was disengaged
- store_and_fwd_flag - This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server - Y=store and forward; N=not a store and forward trip
- trip_duration - duration of the trip in seconds

Used for clustering:

- **pickup_longitude** - the longitude where the meter was engaged
- **pickup_latitude** - the latitude where the meter was engaged

4 The goal:

Clustering of Pickup Locations

5 Tools:

Apache Spark is a fast and general-purpose cluster computing system. It provides high-level APIs in Java, Scala, Python and R, and an optimized engine that supports general execution graphs. It also supports a rich set of higher-level tools including Spark SQL for SQL and structured data processing, MLlib for machine learning, GraphX for graph processing, and Spark Streaming.

6 Language:

7 Implementation:

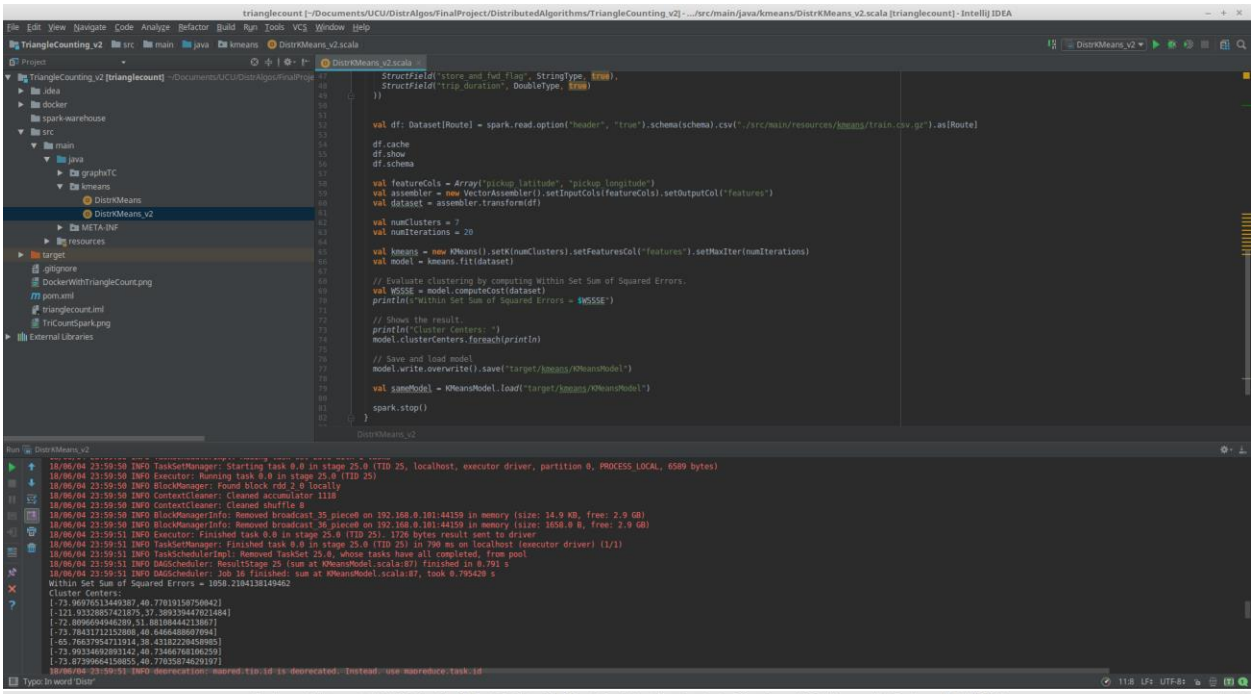
I used KMeans from Spark's machine learning (ML) library (`org.apache.spark.ml.clustering.KMeans`)

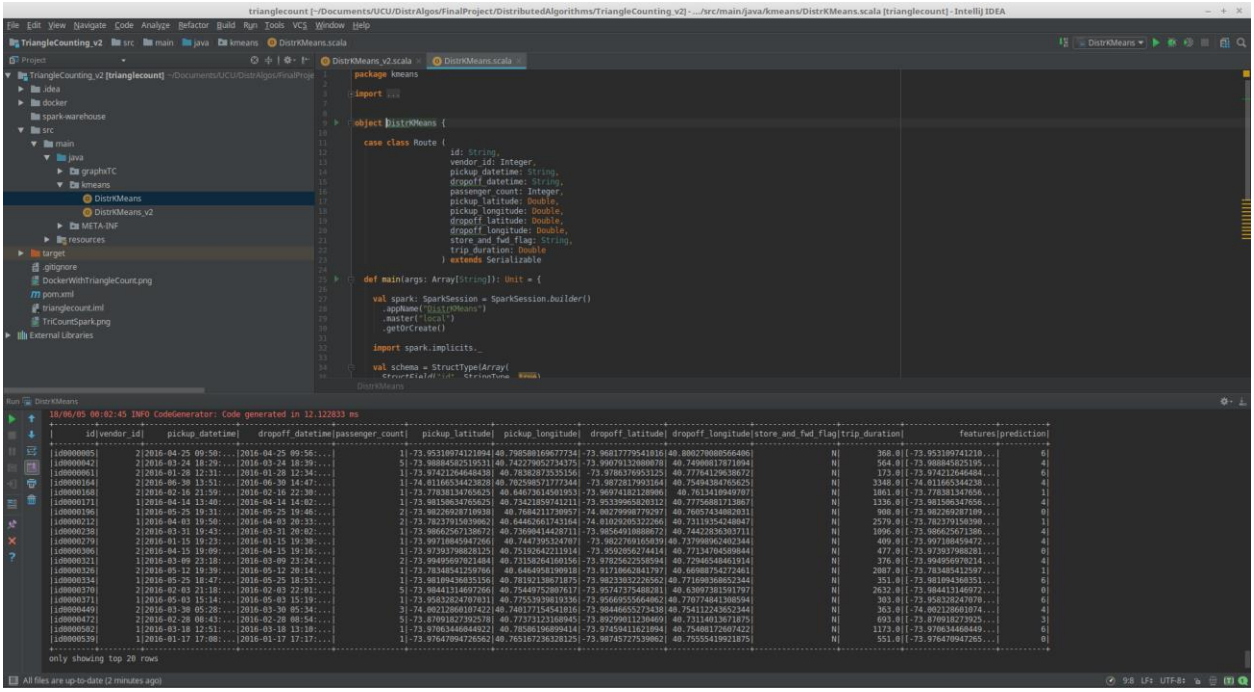
github link:
https://github.com/arxitekton/DistributedAlgorithms/tree/master/TriangleCounting_v2/src/main/java/kmeans

8 Result:

Cluster prediction for Pickup Locations

9 Screenshots:





Literature

[1] H. Jiawei,M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, San Francisco California, Morgan Kaufmann Publishers, 2012.

[4] P. Rai, and S. Sing, “A survey of clustering techniques”, International Journal of computer Applications, vol. 7(12), pp.1-5, 2010.

[5] T. Soni Madhulatha, “An overview on clustering methods”, IOSR Journal of engineering, vol. 2(4), pp.719-725, 2012.

[6] C. Zhang, and Z. Fang, “An improved k-means clustering algorithm”, Journal of Information & Computational Science, vol. 10(1), pp.193-199, 2013.

[7] Hanlei Zhu. Step by Step to K-Means Clustering [Blog post]. Retrieved from Data Science Blog

https://healthcare.ai/step-step-k-means-clustering/