Big Data Computing
Lab - IV

August 24, 2021

I. Netflix Movie Recommendation

Netflix, an online DVD-rental and video streaming service, held an open competition, Netflix Prize, for choosing the best algorithm to predict the user ratings for films, depending upon prior ratings without any other information about the users or films, i.e. without the users or the films being identified except by numbers assigned for the contest. The training data set comprises 100,480,507 ratings that 480,189 users gave to 17,770 movies (>2GB file size). Each tuple of the dataset holds the information concerning following attributes:

Column	Description
CustomerID	Unique integer IDs for identifying users ranging between 1 to 2649429
MovieID	Unique integer Movie identifier ranging from 1 to 17,770 sequentially.
Rating date	Date of movie rating by user in YYYY-MM-DD format
Rating	On a five star (integral) scale from 1 to 5

Given the huge volume of the input dataset, we will work with a trimmed-version-dataset comprising attributes including **<CustomerID, MovieID, Rating>** as part of an input file: '**input.txt**' . Now utilizing **map-reduce** programs, output the consolidated ratings for every potential movie that can get recommended to each user. Afterwards, extracting the highest ranking rated movies can be suggested to respective viewers accordingly.

A. You are provided with the instructions & source code of several map-reduce programs which performs the following task:

a. **RatingDataReader.java** : Runs map-reduce jobs to read the raw rating data and output the aggregated rating data on the basis of user-id.

b. **CooccurrenceMatrixGenerator.java :** Runs map-reduce jobs to read the aggregated rating data on the basis of user-id and generate the cooccurence of each movieID pair.

c. CooccurrenceMatrixNormalization.java : Runs map-reduce jobs to read the unnormalized co-occurrence matrix and normalize the row of movieIdi.

d. **UserRatingAveraging.java** : Runs map-reduce jobs to read the raw rating data and calculate the average rating given by a user.

e. MatrixCellMultiplication.java : Runs multiple mappers (first for: reading the normalized co-occurrence matrix and second for: generate the rating matrix by reading the raw rating data) and a reduce job for multiplying a cell of co-occurrence matrix with the corresponding cell of rating matrix.

f. **MatrixCellSum.java** : Runs map-reduce jobs to read the sub product of each cell and sum up the sub rating to the final rating input.

II. K-means Clustering

You are provided with the source code (Map-Reduce program) which performs the following tasks: Implement k-means algorithm using MapReduce on a sample dataset of points.

Input: a text file containing a set of points

8, 14, 3,12,21,3,10,30,29,48,31,6,29,44,32,12,44,44,32,7,37,25,36,13,14,33,29,1,4,31

Output: number of points assigned to each cluster depending upon set number of clusters (Ex- k = 5)

Cluster centers: 8.0, 21.0, 30.0, 41.0, 48.0

Clustered data:	Cluster center	Data point in each cluster
	8.0	12 8 3 14 6 13 10 14 4 3 7 1 12
	21.0	21
	30.0	31 29 33 25 32 29 31 29 30 32
	41.0	44 44 44 37 36
	48.0	48