**Project – Implementing a Recommender System using the Hadoop ecosystem**

Step 1: Introduction to Hadoop Commands and experiments with the Hadoop commands

users-MacBook-Pro-2163:~ etanan$ ssh hadoop@172.16.230.133

hadoop@172.16.230.133's password:

Last login: Fri Oct 19 14:56:38 2018 from 1.qarestr.sub-172-16-230.myvzw.com

1. **Prepare to store the data into the HDFS file system by creating directories for it.**

[hadoop@133 ~]$ hadoop fs -mkdir -p /user/data/movielens/raw

[hadoop@133 ~]$ hadoop fs -mkdir /user/data/movielens/raw/users

[hadoop@133 ~]$ hadoop fs -mkdir /user/data/movielens/raw/movies

[hadoop@133 ~]$ hadoop fs -mkdir /user/data/movielens/raw/ratings

1. **Put the data into HDFS**

Make sure the terminal code is running from /home/cloudera/Exercises/ml-1m

[hadoop@133 ~]$ cd Exercises

[hadoop@133 Exercises]$ ls –la ##list the files in the directory to locate ml-1m

total 8

drwxr-xr-x. 4 hadoop hadoop 34 Oct 10 12:44 .

drwx------. 10 hadoop hadoop 4096 Oct 19 02:07 ..

drwxr-x---. 2 hadoop hadoop 173 Oct 19 02:51 ml-1m

drwxr-xr-x. 3 hadoop hadoop 4096 Oct 10 12:45 scripts

[hadoop@133 Exercises]$ cd ml-1m

[hadoop@133 ml-1m]$ hadoop fs -put users.dat /user/data/movielens/raw/users

[hadoop@133 ml-1m]$ hadoop fs -put movies.dat /user/data/movielens/raw/movies

[hadoop@133 ml-1m]$ hadoop fs -put ratings.dat /user/data/movielens/raw/ratings

1. **Confirming the data is present**

[hadoop@133 ml-1m]$ hadoop fs -ls -R /user/data/movielens

drwxr-xr-x - hadoop supergroup 0 2018-10-18 21:21 /user/data/movielens/raw

drwxr-xr-x - hadoop supergroup 0 2018-10-18 21:22 /user/data/movielens/raw/movies

-rw-r--r-- 1 hadoop supergroup 171308 2018-10-18 21:22 /user/data/movielens/raw/movies/movies.dat

drwxr-xr-x - hadoop supergroup 0 2018-10-18 21:23 /user/data/movielens/raw/ratings

-rw-r--r-- 1 hadoop supergroup 24594131 2018-10-18 21:23 /user/data/movielens/raw/ratings/ratings.dat

drwxr-xr-x - hadoop supergroup 0 2018-10-18 21:22 /user/data/movielens/raw/users

-rw-r--r-- 1 hadoop supergroup 134368 2018-10-18 21:22 /user/data/movielens/raw/users/users.dat

1. Now make a subdirectory to contain the “cleaned” data that we will create via mapreduce from the step 2

[hadoop@133 ml-1m]$ hadoop fs -mkdir /user/data/movielens/cleaned

1. **cat examines the content of the raw files**

[hadoop@133 ml-1m]$ hadoop fs -cat /user/data/movielens/raw/ratings/ratings.dat



1. **tail – display last kB of the file to standard out**

[hadoop@133 scripts]$ hadoop fs -tail /user/data/movielens/raw/users/users.dat



1. **get – copies the HDFS file to the local directory**

[hadoop@133 ml-1m]$ hadoop fs -get /user/data/movielens/raw/ratings/ratings.dat ratings\_cpy.dat

[hadoop@133 ml-1m]$ ls –la ##The HDFS file is now in the local directory highlighted as ratings\_cpy.dat

total 48364

drwxr-x---. 2 hadoop hadoop 173 Oct 19 02:51 .

drwxr-xr-x. 4 hadoop hadoop 34 Oct 10 12:44 ..

-rw-r-----. 1 hadoop hadoop 212 Oct 10 12:44 .\_movies.dat

-rw-r-----. 1 hadoop hadoop 171308 Oct 10 12:44 movies.dat

-rw-r--r--. 1 hadoop hadoop 24594131 Oct 19 02:51 ratings\_cpy.dat

-rw-r-----. 1 hadoop hadoop 212 Oct 10 12:44 .\_ratings.dat

-rw-r-----. 1 hadoop hadoop 24594131 Oct 10 12:44 ratings.dat

-rw-r-----. 1 hadoop hadoop 5189 Oct 10 12:44 README

-rw-r-----. 1 hadoop hadoop 259 Oct 10 12:44 .\_README

-rw-r-----. 1 hadoop hadoop 212 Oct 10 12:44 .\_users.dat

-rw-r-----. 1 hadoop hadoop 134368 Oct 10 12:44 users.dat

**2. Processing Data using Python**

Working directory is cd /home/cloudera/Exercises/scripts

Therefore, get into scripts folder

[hadoop@131 ml-1m]$ cd ..

[hadoop@131 Exercises]$ cd scripts

[hadoop@133 scripts]$ ./do\_mapreduce.sh users ##The following lines are the last 5 lines of what returned

File Input Format Counters

Bytes Read=171308

File Output Format Counters

Bytes Written=150482

2018-10-16 05:05:53,358 INFO streaming.StreamJob: Output directory: /user/data/movielens/cleaned/movies

[hadoop@133 scripts]$ ./do\_mapreduce.sh movies ##The following lines are the last 5 lines of what returned

File Input Format Counters

Bytes Read=171308

File Output Format Counters

Bytes Written=150482

2018-10-16 05:05:53,358 INFO streaming.StreamJob: Output directory: /user/data/movielens/cleaned/movies

[hadoop@133 scripts]$ ./do\_mapreduce.sh ratings ##The following lines are the last 5 lines of what returned

File Input Format Counters

Bytes Read=24594131

File Output Format Counters

Bytes Written=21593504

2018-10-16 05:06:17,607 INFO streaming.StreamJob: Output directory: /user/data/movielens/cleaned/ratings

**Viewing do\_mapreduce.sh shell script**

[hadoop@133 scripts]$ more do\_mapreduce.sh

#!/bin/bash

STREAMING=/home/hadoop/hadoop-3.1.1/share/hadoop/tools/lib/hadoop-streaming-3.1.1.jar ##Defines STREAMIN and calls a Java Archive file; the same file we used on the first Hadoop exercise to mapreduce Shakespeare files.

SCRIPTS=/home/hadoop/Exercises/scripts ##Defines Scripts as the parent directory

HDFS\_LOC=/user/data/movielens ##Defines HDFS\_LOC as the hdfs files directory which is /user/data/movielens

if [ "${1}" == "" ] || [ ! -f ${SCRIPTS}/clean\_${1}\_dat.py ]; then

echo -e "\nSpecify users, movies, or ratings with ${0}.\nFor example:\n ${0} users\n"

exit 1

fi

##The four lines of code above return a string “Specify users, movies, or ratings with ./do\_mapreduce.sh. For example: ./do\_mapreduce.sh users” where ${0} is the shell file and if ${1}—which is the standard input file—is not specified

echo "Removing previous results directory if it exists..." ##returns this string if ${1} is specified.

hadoop fs -rm -r ${HDFS\_LOC}/cleaned/${1} ##Deletes the directory /user/data/movielens/cleaned/${1}

hadoop jar ${STREAMING} \ ##Runs the Java Archive file

-Dmapred.reduce.tasks=0 \ ##Only map function will be used to process data

-input ${HDFS\_LOC}/raw/${1} \ ##Input data will be used from the /user/data/movielens/raw/${1} directory

-output ${HDFS\_LOC}/cleaned/${1} \ ##Output data will be placed in the /user/data/movielens/cleaned/${1} directory

-mapper ${SCRIPTS}/clean\_${1}\_dat.py \ ##Mapper specifies which python file to run from /scripts folder

-file ${SCRIPTS}/clean\_${1}\_dat.py ##File copies the clean data file and stores it in /scripts folder per ${1}

**Step 3: Creating Hive Tables**

[hadoop@133 scripts]$ hive -f create\_users\_table.hql

[hadoop@133 scripts]$ hive -f create\_movies\_table.hql

[hadoop@133 scripts]$ hive -f create\_ratings\_table.hql

##The above lines create three dbase table and import data from /users/data/movielens/cleaned/(users or movies or ratings)

[hadoop@133 scripts]$ hive -e "describe users" ##The next lines are the returned values

OK

id int

gender char(1)

age int

occupation int

zip char(5)

Time taken: 2.216 seconds, Fetched: 5 row(s)

[hadoop@133 scripts]$ hive -e "select count(\*) from ratings" ##The next lines are the returned values.

Total MapReduce CPU Time Spent: 0 msec

OK

1000209

Time taken: 13.416 seconds, Fetched: 1 row(s)

**Can you do queries for these questions?**

In order to do this, the fields in the users and ratings data set need to be identified.

[hadoop@133 scripts]$ hive ##Initiates the Hive consoles.

hive> describe users; ##Displays the fields in the users data set.

OK

id int

gender char(1)

age int

occupation int

zip char(5)

Time taken: 0.126 seconds, Fetched: 5 row(s)

hive> describe ratings; ##Displays the fields in the ratings data set.

OK

userid int

movieid int

rating int

tstamp int

Time taken: 0.086 seconds, Fetched: 4 row(s)

1. **How many users are under 35 years old?**

hive> SELECT count(\*) FROM users WHERE age <35; ##The returned values is 3421

OK

3421

Time taken: 11.387 seconds, Fetched: 1 row(s)

1. **The top three users who did the most ratings?**

hive> SELECT userid, count(\*) FROM ratings GROUP BY userid ORDER BY count(\*) DESC LIMIT 0,5;

##This query returned five userids along with the number of times they rated a moved.

OK

4169 2314

1680 1850

4277 1743

1941 1595

1181 1521

Time taken: 3.247 seconds, Fetched: 5 row(s)

1. **How many users in each occupation?**

hive> SELECT occupation, count(\*) FROM users GROUP BY occupation;

##This query returned 21 lists of occupations (from 0 to 20) users provided.

OK

0 711

1 528

2 267

3 173

4 759

5 112

6 236

7 679

8 17

9 92

10 195

11 129

12 388

13 142

14 302

15 144

16 241

17 502

18 70

19 72

20 281

Time taken: 1.776 seconds, Fetched: 21 row(s)

**Step 4: Build the basic recommender system using Mahout**

1. Prepare data file for Mahout

[hadoop@133 scripts]$ ./do\_mapreduce\_mahout.sh

File Input Format Counters

Bytes Read=24594131

File Output Format Counters

Bytes Written=12553665

2018-10-16 05:09:16,557 INFO streaming.StreamJob: Output directory: /user/data/movielens/mahout/input

1. Specify users for recommendation

[hadoop@133 scripts]$ hadoop fs -mkdir /user/data/movielens/mahout

mkdir: `/user/data/movielens/mahout': File exists ##File was already in the directory

[hadoop@133 scripts]$ hadoop fs -put mahout\_users.dat /user/data/movielens/mahout ##Copies mahout\_users.dat from /scripts foler to HDFS folder in /user/data/movielens/mahout

[hadoop@133 scripts]$ ./do\_mahout.sh

File Input Format Counters

Bytes Read=2184965

File Output Format Counters

Bytes Written=277

18/10/19 03:05:23 INFO MahoutDriver: Program took 174033 ms (Minutes: 2.90055)

[hadoop@133 scripts]$ hadoop fs -cat /user/data/movielens/mahout/output/part-r-00000

1 [529:5.0,2636:5.0,2638:5.0,3034:5.0,3296:5.0,1124:5.0,1253:5.0,594:5.0,1187:5.0,2637:5.0]

2 [529:5.0,2375:5.0,3035:5.0,3431:5.0,2376:5.0,2243:5.0,1715:5.0,3430:5.0,1187:5.0,527:5.0]

3 [3629:5.0,3168:5.0,3035:5.0,3034:5.0,3429:5.0,2375:5.0,3760:5.0,2045:5.0,1187:5.0,200:5.0]