Homework 2 Part 2

Question 1

Create directory named /user/hadoop/data

- hadoop fs -mkdir /user/hadoop/data/

Transferring logs directory to directory named /user/hadoop/data

- hadoop fs -copyFromLocal logs /user/hadoop/data

Move Web log files from /user/hadoop/data to shared directory /user/share/

- hadoop fs -mv /user/hadoop/data /user/share/data

Listing all the directories and subdirectories in /user/share

- hadoop fs -ls /user/share

Listing all the directories and subdirectories in /user/share

- hadoop fs -ls /user/hadoop

Removing the corrupted file

- hadoop fs -rm /user/share/data/2020-01-01.txt

Upload the new file from logs/new directory to the /user/share/data

- hadoop fs -copyFromLocal logs/new/2020-01-01-new.txt /user/share/data

Renaming the new file

- hadoop fs -mv /user/share/data/2020-01-01-new.txt /user/share/data/2020-01-01.txt

Display of the file 2020-01-01.txt

- hadoop fs -cat /user/share/data/2020-01-01.txt

Display the last kilobyte of the file

- hadoop fs -tail /user/share/data/2020-01-01.txt

Question 2

(a) Car for sale data

- i) Mapper function: Tokenize each line using comma as delimiter to extract the make (column 2), model (column 3), and price (column 5) of each car for sale
- ii) Mapper output: key is the (make, model), value is the price
- iii) Reducer input: key is the (make, model), value is the list of prices
- iv) Reduce function: Calculates the median value of the list of car prices for each combination (make, model)
- v) Reducer output: key is the make and model, value is the median price

(b) Netflix movie rental data

- i) Mapper function: Tokenize each line using white space as delimiter to extract user id (column 1) and movie genre (column 4)
- ii) Mapper output: key is user id, value is movie genre
- iii) Reducer input: key is user id, value is a list of movie genres of the movies streamed by the user
- iv) Reduce function: Find the most frequent value among the list of movie genres streamed by the user
- v) Reducer output: key is the user id, value is the favorite movie genre

(c) YouTube subscriber data

- i) Mapper function: Tokenize each line using white space as the delimiter to extract the user and subscriber
- ii) Mapper output: For each line, output two key-value pairs: (1) Key is user, value is (subscriber, 1) and (2) key is subscriber, value is (user, -1).
- iii) Reducer input: key is the user, value is the list of tuples (user, +/- 1)
- iv) Reduce function: For each user, look for users associated with both +1 and -1. Users that have both + 1 and -1 are subscribers of each other's videos and will be sent to the output
- v) Reducer output: Key is a user (u_1), value is another user (u_2), where $u_1 < u_2$ in terms of lexicographic (alphabetical) order. The ordering prevents duplicate pairs generated as output, eg., (john, mary) and (mary, john)

(d) Loan applicant data

- i) Mapper function: Tokenize each line using comma delimiter to extract all the attributes and class
- ii) Mapper output: key is the attribute, value is (attribute value, class)
- iii) Reducer input: key is the attribute, value is list of (attribute value, class) pairs
- iv) Reduce function: Calculates the joint probability of each pair. Then, calculate the entropy for attributes using the joint probability
- v) Reducer output: key is the attribute, value is the entropy

(e) Document data

i) Mapper function: Processes each line separately
ii) Mapper output: key is document number, value is the word
iii) Reducer input: key is the document number, value is list of the words
iv) Reduce function: Calculates the number of words
v) Reducer output: key is document number, value is list of words and the number of words
vi) Mapper input: key is document number, value is list of words and the number of words
vii) Mapper function: Finds number of similar words between two documents
viii) Mapper output: key is pair of documents, value is the list of number of words in each document and number of similar words
ix) Reducer input: key is pair of documents, value is the list of number of words in each document and number of similar words
x) Reduce function: Calculates the cosine similarity between documents
xi) Reducer output: key is pair of documents, value is cosine similarity