**CS7070-Big Data Analytics**

**Spring 2020**

**Programming Assignment #1**

**Due Date: March 9th, 2020 (9PM)**

In this programming assignment, you are expected to use and modify the MapReduce programs for computing the TFIDF for terms in a set of documents. Sahil has shared and demonstrated this program with you during one of the classes. The tasks to be accomplished by, as parts of this assignment, you are:

1. (20) Execute all phases of the TFIDF program, on the small sample data shared by Sahil, and submit the following items:
   1. TFIDF for top 18 terms in each document, sorted in descending order of their tfidf values, and formatted for easy readability.

Answer: I have executed all the three phases of the TFIDF Program by providing input as tfidf.txt ,loaded this file into HDFS and keeping three phases map reduce java program into the virtual box machine and after the execution of three phases I have used the command :

hadoop fs -get <HDFS PATH> <Virtual Box Path> and brought the files into virtual box and later written the python program to compute tfidf values by loading individual files into dictionaries and displayed the top 18 terms with high TFIDF values in each document.

Python Script: python\_script\_Q1.py

Output file: First\_question\_output.txt

Input files:

Q1\_part-r-00000\_PH1.txt

Q1\_part-r-00000\_PH2.txt

Q1\_part-r-00000\_PH3.txt

1. (25) Modify the programs to remove from consideration all those words that occur only once or twice in each document. Repeat the task of Q1 above. Comment on any changes in the results of part 1(a). Select at least 3 different words for which there is a change in their tfidf values and explain the reason for the change.

Answer: To remove the words with frequency less than 3 ,I have changed the code in phase one reducer by adding the condition “if(sum>2)” and sent the output of the phase1 to phase2 and phase3 and used the “Hadoop -fs get” command to brought the files into Virtual Box and then sent three files as input to python script for calculating the TFIDF values and displayed the top 18 terms in each document.

Now the phase one output is shrinked and it contains only the words with frequency greater than 2 in each document

Below are the words where I could see a difference in the values.

TFIDF for the word “Words” in the document 0012 for 1(a) is 0.011829

Whereas for the same word in the document 0012 for 2(a) is 0.072952

The reason for the change is because of the change in TF value idf values

Because of the change number of words in the document 12 in question 2.

**Second case:**

TFIDF for the word “a” in 1(a) in the document 0011 is 0.008466

Whereas for the word “a” in 2(a) in the document 0011 is 0.107511 because of the change in total number of the words in the document 0011.

**Third case:**

TFIDF for the word “by in the document 0010 for 1(a) is 0.011240 where as for the same word in same document in 2(a) is almost zero because the frequency of the word by in 0010 document the total number of words in the document 0010 has changed from 47 to 3.

Java file for phase one change: QtwoPhaseOne.java

Python Script: python\_script\_Q2.py

Output file: Second\_question\_output.txt

Input files:

Q2\_part-r-00000\_PH1.txt

Q2\_part-r-00000\_PH2.txt

Q2\_part-r-00000\_PH3.txt

1. (30) Now consider a “Term” to mean a *2*-gram (two words occurring sequentially) in a document. Modify the programs given to you to compute the TFIDF for each 2-gram. Submit the following items:
   1. List of top 20 2-grams for each document, having the highest TFIDF values. The task of selecting the top 20 terms does not need to be done by the MapReduce program.
   2. Which output – obtained in 3(a) or in 2(a) – better characterizes the documents? Give reasons for your answers.

Answer:

For calculating the two grams I changed the code in phase one Mapper to consider the consecutive words

tempStr = (docPart[i]+" "+docPart[i+1]).replaceAll("\\p{P}", ""); //adding the ith and i+1 word with space to form a bigram and removing special character and punctuation from the word .

The output of the phase one is sent as input to phase 2 and phase3 and after that the outputs has been sent to python program for calculating the TFIDFS and the top 20 bigrams with highest TFIDF’S values are displayed.

I would say the output in 3(a) better characterizes the documents than in 2(a) output because bigrams would be sequence of the two words and the retrieving of relevant documents for the query has high probability when we consider bigrams as it will search of the consecutive words instead of a single word and bigram analysis is used for the analyzing the author writing style.

Java file for phase one change: QThreePhaseOne.java

Python Script: python\_script\_Q3.py

Output file: Third\_question\_output.txt

Input files:

Q3\_part-r-00000\_PH1.txt

Q3\_part-r-00000\_PH2.txt

Q3\_part-r-00000\_PH3.txt

1. (20) Once your program is working for the above two parts, run the programs on a larger collection of documents (to be provided to you by March 2nd) and repeat the above task . Discuss the results for 1(a), 2(a), and 3(a) in the context of the new set of documents.

Answer : In this we are passing three documents sequentially to the phase one program in the mapper class and repeated the 1(a) task on the new dataset and the phase one output is sent to phase2 and phase3 as input and then the three files has been sent to python program to find the TFIDF’s and the top 18 terms in each document with highest TFIDF’s values are displayed.

Java program: QFourPhaseOne.java

Input files:

Q4\_part-r-00000\_PH1.txt

Q4\_part-r-00000\_PH2.txt

Q4\_part-r-00000\_PH3.txt

Python file: python\_script\_Q4.py

Output File: Fourth\_question\_output1.txt

Eliminating the words with frequency less than 3 in the collection ,I have added the same condition which I used in 2(a) and sent the collection(3 documents) as input to phase one and the phase one output is sent to phase2 and phase 3 and later the outputs of phase 1,2,3 is passed to python program where I calculated TFIDFS.

Difference words :

1)the word “THE” in document 84-0.txt for 4\_1(a) has the tfidf value 0.023878

Whereas it has 0.001890 in 4\_2(a) because of the change in total word count in the document 84-0.txt

2) the word “I” in document 84-0.txt for 4\_1(a) has the tfidf value 0.006164

Whereas it has 0.018083in 4\_2(a) because of the change in total word count in the document 84-0.txt

3) the word “He” in document 84-0.txt for 4\_1(a) has the tfidf value 0.000776

Whereas it has 0.000840in 4\_2(a) because of the change in total word counts in the document 84-0.txt

Java program: QFourPhaseOneSecond.java

Input files:

Q42\_part-r-00000\_PH1.txt

Q42\_part-r-00000\_PH2.txt

Q42\_part-r-00000\_PH3.txt

Python file: python\_script\_Q42.py

Output File: Fourth\_question\_output2.txt

For Finding Bigrams I have used the same logic which used in 3(a) and changed the code in mapper of the phase1 and the output of phase 1 is sent as i/p to phase 2 and phase 3 and the three outputs files generated in hdfs has been brought to local using hadoop -fs get command and then applied the python program on the bigrams for calculating TF-IDF’s values and displayed the top 20 bigrams in each document.

I would say Bigrams generated in 4\_3(a) characterizes the document efficiently when compared with 4\_2(a) as it looks for the consecutive words in collection and returns the relevant documents from the collection for that bigram

Java program: QFourPhaseOneThree.java.java

Input files:

Q43\_part-r-00000\_PH1.txt

Q43\_part-r-00000\_PH2.txt

Q43\_part-r-00000\_PH3.txt

Python file: python\_script\_Q43.py

Output File: Fourth\_question\_output3.txt

1. (5) Well organized and clearly understandable presentation of results in the submission.

Commands to bring the file to virtual box from HDFS:

hadoop fs -get /tmp/Assignment1/Q43\_output\_ph1/part-r-00000 /home/maria\_dev/Assignment/Q43\_part-r-00000\_PH1.txt

hadoop fs -get /tmp/Assignment1/Q43\_output\_ph2/part-r-00000 /home/maria\_dev/Assignment/Q43\_part-r-00000\_PH2.txt

hadoop fs -get /tmp/Assignment1/Q43\_output\_ph3/part-r-00000 /home/maria\_dev/Assignment/Q43\_part-r-00000\_PH3.txt

command to execute the python script in Virtual Box:

Python <script\_name.py>

Your submission for each part include:

* Listing of source code used
* Output produced by your code
* A description of input files used and any parameters etc. set in your program.
* Interpretation of or comments about the results obtained by your program.