Big Data Algorithms

Lab_1 WANG Wanqing

4.1 TFIDF

In this pre-process, the intuition is to use file “defoe-robinson-103.txt”and “callwild”, and the files produced “stopwords.txt” with the class “StopWords.java”.
And the process is to 1) remove all stop words and special characters, just keep numbers and alphabets, then keep each unique word in each line without empty; 2) store the number of records on HDFS; 3) Ascendingly order the tokens with frequency and store them on HDFS.
For this pre-processing part, the main intuition is to create a new project, then implement 2 classes that one is implemented to make the word count, the other is to remove all stop words and count the frequency.
To finish the pre-processing part, the file “stopwords.txt” will be used to remove the stop words from the files. So, the 2 files need to be input into Hadoop HDFS for the next usage.

The code shows here in terminal:
Create folders in the project and put the files need to be used into HDFS

```
hadoop fs -mkdir input
wget http://www.textfiles.com/etext/FICTION/defoe-robinson-103.txt
hadoop fs -copyFromLocal defoe-robinson-103.txt input
wget http://www.textfiles.com/etext/FICTION/callwild
hadoop fs -copyFromLocal callwild input
Hadoop fs –mkdir output
```

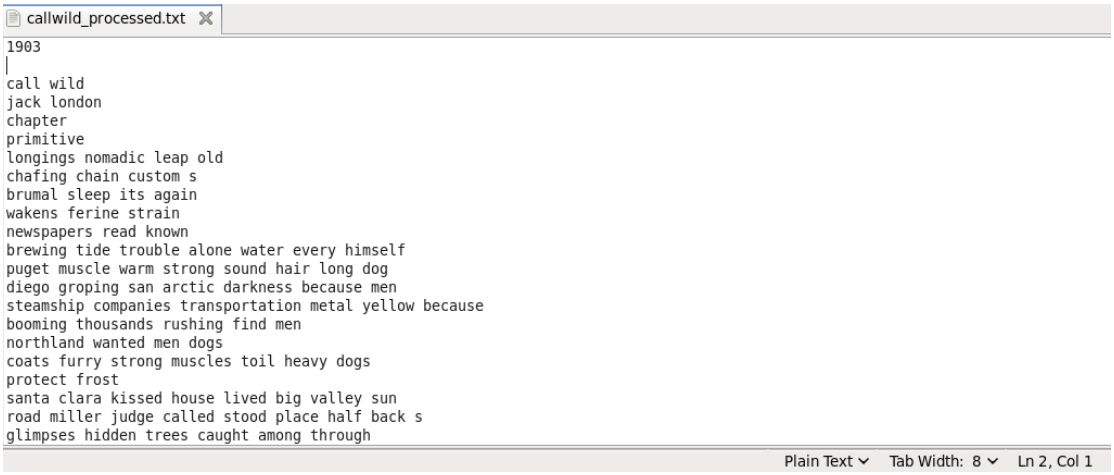
After creating the input and output folders

```
hadoop jar tfidf.jar tfidf.StopWords input output
hadoop fs -getmerge output workspace/tfidf/output/stopwords.txt
hadoop fs –put workspace/tfidf/output/stopwords.txt input1
```

The files stopwords.txt are now in HDFS. After implementing 3 classes(SkipStopWords.java and TFIDF related classes) in Eclipse, export the jar file of the project named tfidf.jar. We can now run the project in terminal.

```
hadoop jar tfidf.jar tfidf.SkipStopWords input/defoe-robinson-103.txt output/defoe -skip input1/stopwords.txt
hadoop fs -getmerge output/defoe workspace/tfidf/output/defoe_processed.txt

hadoop jar tfidf.jar tfidf.SkipStopWords input/callwild output/callwild -skip input1/stopwords.txt
hadoop fs -getmerge output/callwild workspace/tfidf/output/callwild_processed.txt
```



After those above, we can get two processed files without tab, space, stopwords and punctuations in the context.

Put the 2 files into a new input1:

```
hadoop fs -mkdir input1
hadoop fs –put defoe_processed.txt input1
hadoop fs –put callwild_processed.txt input1
```

Then we can run the TFIDF class to get the results:

```
hadoop jar tfidf.jar tfidf.TFIDF  input1 output1
```

```
                Reduce shuffle bytes=441246
                Reduce input records=10921
                Reduce output records=10921
                Spilled Records=21842
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=215
                CPU time spent (ms)=6570
                Physical memory (bytes) snapshot=370810880
                Virtual memory (bytes) snapshot=5490692096
                Total committed heap usage (bytes)=226365440
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=419398
        File Output Format Counters
                Bytes Written=499270
cloudera@quickstart ~]$
```

```
hadoop fs -getmerge output1 workspace/tfidf/output/tfidf.txt
```

tfidf.txt

```
acceptable_defoe-robinson-103.txt       0.47712125471966244
accepted_callwild.txt   0.3010299956639812
accepted_defoe-robinson-103.txt 0.4822681122428943
accepting_callwild.txt  0.6207490639591157
access_defoe-robinson-103.txt   0.47712125471966244
accident_defoe-robinson-103.txt 0.535276863192891
accident_callwild.txt   0.3010299956639812
accidental_defoe-robinson-103.txt       0.47712125471966244
accidents_defoe-robinson-103.txt        0.6207490639591157
accommodate_defoe-robinson-103.txt      0.47712125471966244
accommodated_callwild.txt       0.47712125471966244
accommodations_defoe-robinson-103.txt   0.47712125471966244
accomodate_defoe-robinson-103.txt       0.6207490639591157
accompanied_callwild.txt        0.3010299956639812
accompanied_defoe-robinson-103.txt      0.39164905395343774
accompanying_defoe-robinson-103.txt     0.47712125471966244
accomplish_defoe-robinson-103.txt       0.47712125471966244
accomplished_callwild.txt       0.6207490639591157
accomplishing_defoe-robinson-103.txt    0.47712125471966244
according_callwild.txt  0.3010299956639812
according_defoe-robinson-103.txt        0.535276863192891
accordingly_defoe-robinson-103.txt      1.1600553298354574
```

Plain Text Tab Width: 8 Ln 74, Col 51

The format of the the results is:

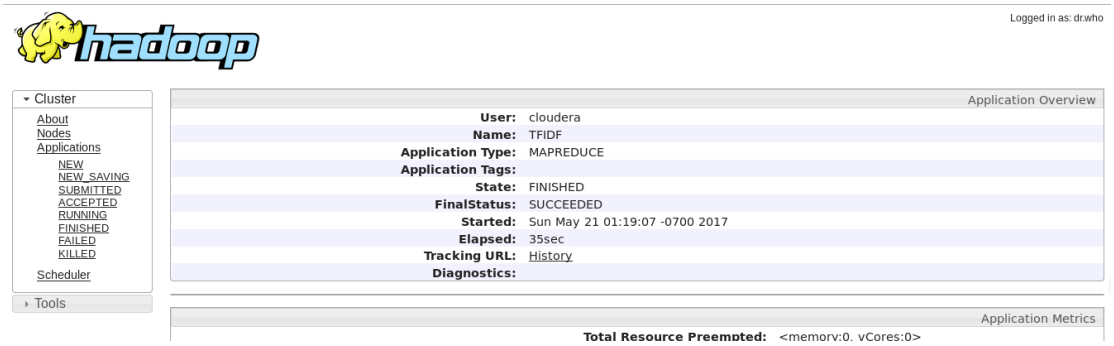
Word_document tfidf socre

In this process, I failed to use EMR job flow(It`s impossible to use google because I`m in China and I cannot see Chinese characters with firebox in cloudera). So I tracked the job application with YARN ResourceManager to see the status.

**There are still some bugs need to be fixed.*

Show 20 entries

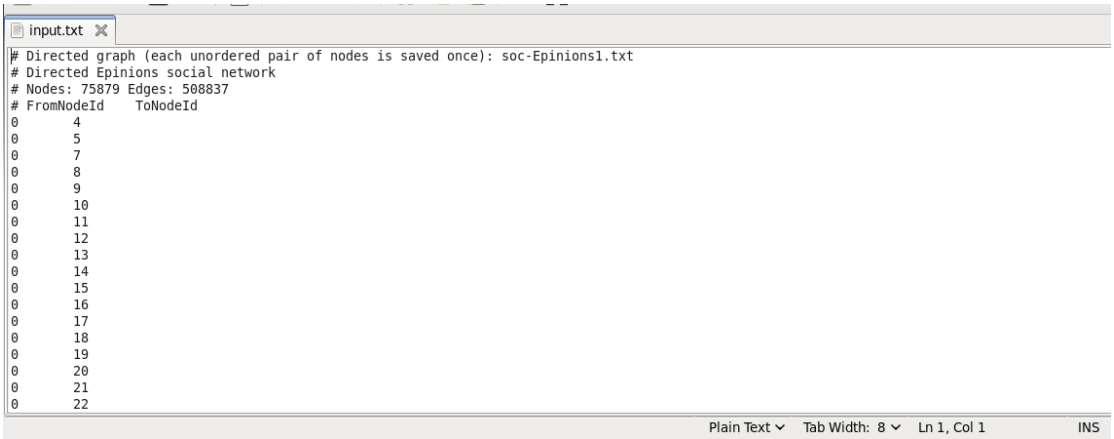
ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Runnin Containe
application_1493977914828_0021	cloudera	TFIDF	MAPREDUCE	root.cloudera	Sun May 21 01:19:07 -0700 2017	Sun May 21 01:19:42 -0700 2017	FINISHED	SUCCEEDED	N/A
application_1493977914828_0020	cloudera	TermFrequency	MAPREDUCE	root.cloudera	Sun May 21 01:18:17 -0700 2017	Sun May 21 01:19:05 -0700 2017	FINISHED	SUCCEEDED	N/A



4.1 PageRank

In this project, we created 4 classes: “Main”, “Map”, “Reduce”, “Node”.

The input is:



The final output is: (Top 10)