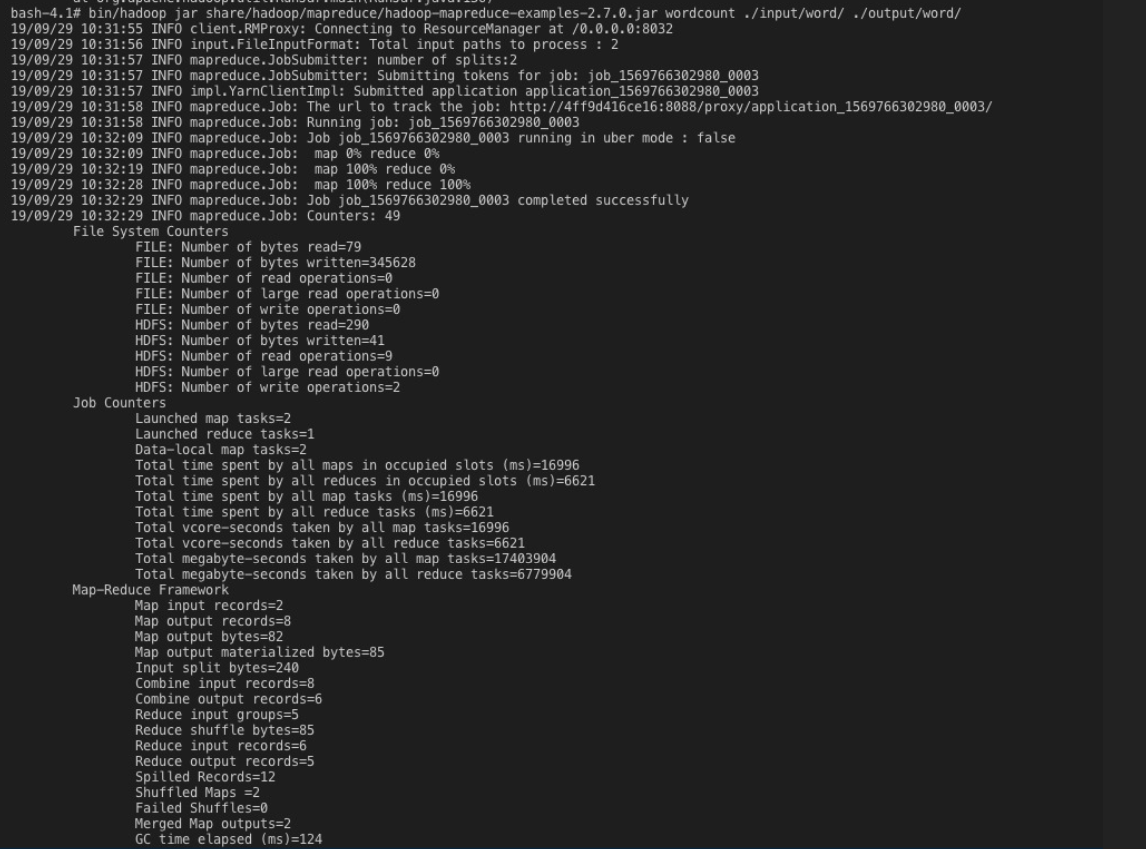
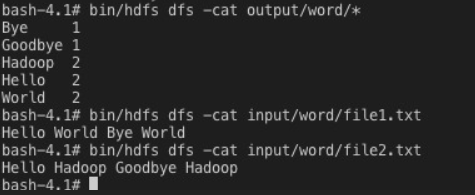
1.1

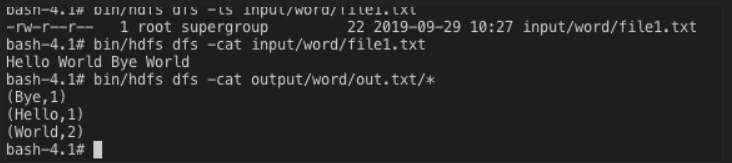
Installed Virtual box and installed cloudera. Created two input files with two sentences and kept them in one directory and then Executed wordcount from share/Hadoop/mapreduce/Hadoop-mapreduce-examples-2.7.0.jar wordcount. Execution was successful creating an output file in hdfs.



1.2



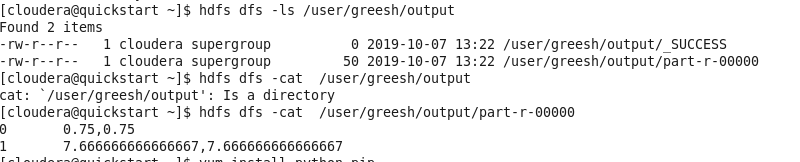
Used Spark shell Scala for running word count. Created input for this and used above commands for execution. Execution was successful creating an output file as below.



2.1



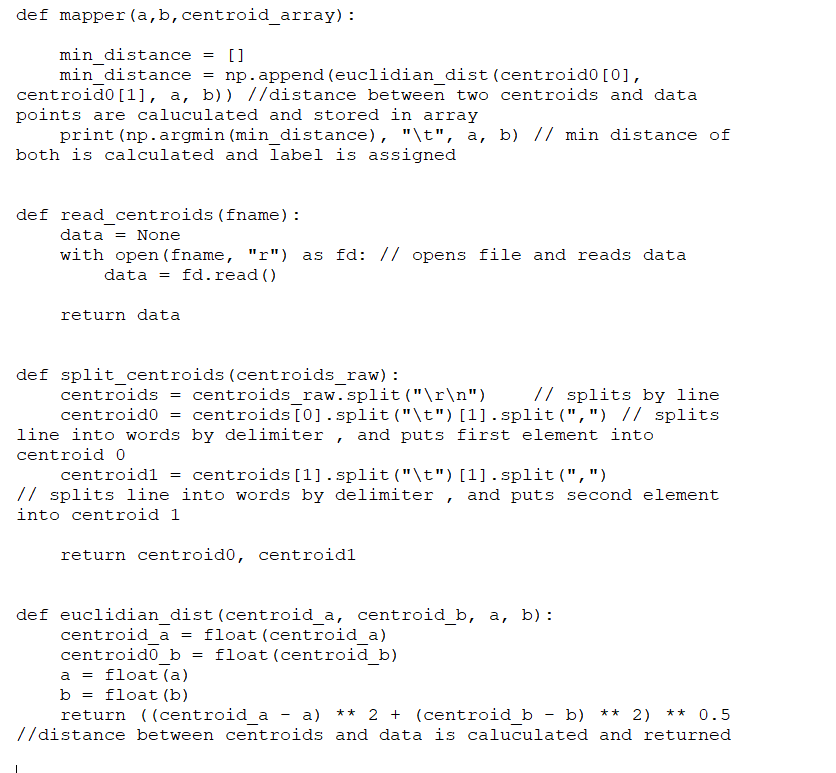
Used For loop for iteration, adding input path of file and setting output path for file. After that updating values of centroid with output and comparing convergence with previous centroids such that it stops if converge() is true. Yes I have successfully implemented iteration for given source code



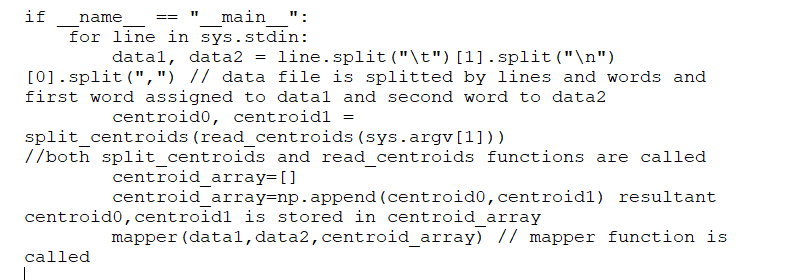
2.2

Combiner class encapsulates the output from map with same key and output of combiner class will sent to actual reduce class as input. Several repeated keys are genrated by maps this basically used to sum-up these large output records from map to less records. It does not have interface instead it uses reducer interface and reduce method in combiner is called on each map output key. It’s input and output key-values should match with reducer class. Combiner increases overall performance of Mapreduce, it also decreases time for data transmission. The big O representation of data passed to reducer phase without combiner is O(NRKM) and with combiner is O(NRK/M)

2.3



I have two methods split\_centroids and read\_centroids for reading and splitting centroids.I am getting data file as system standard input and splitting lines and storing values in x,y.I have Euclidian\_distance calucuting distance between centroids and data points from input file. After calculating distance finding out minimum distance using numpy.argmin and assigning label for the cluster.



**Hadoop commands:**

To make directory and to place input file :

hdfs dfs -mkdir /user/greesh/kmeans\_python/inputfile

hdfs dfs -put /home/cloudera/Downloads/data.hdfs /user/greesh/kmeans\_python/inputfile

hdfs dfs -rm -r /user/greesh/kmeans\_python/output

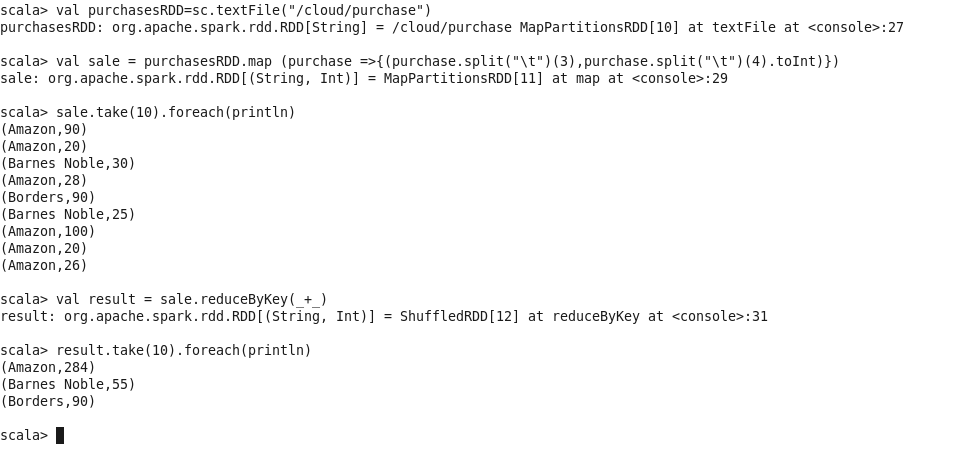
Hadoop streaming:

hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar -input /user/greesh/kmeans\_python/inputfile/data.hdfs -output /user/greesh/kmeans\_python/output -mapper "python mapper.py /user/greesh/centroids" -file /home/cloudera/python\_source/Mapper.py

To check output:

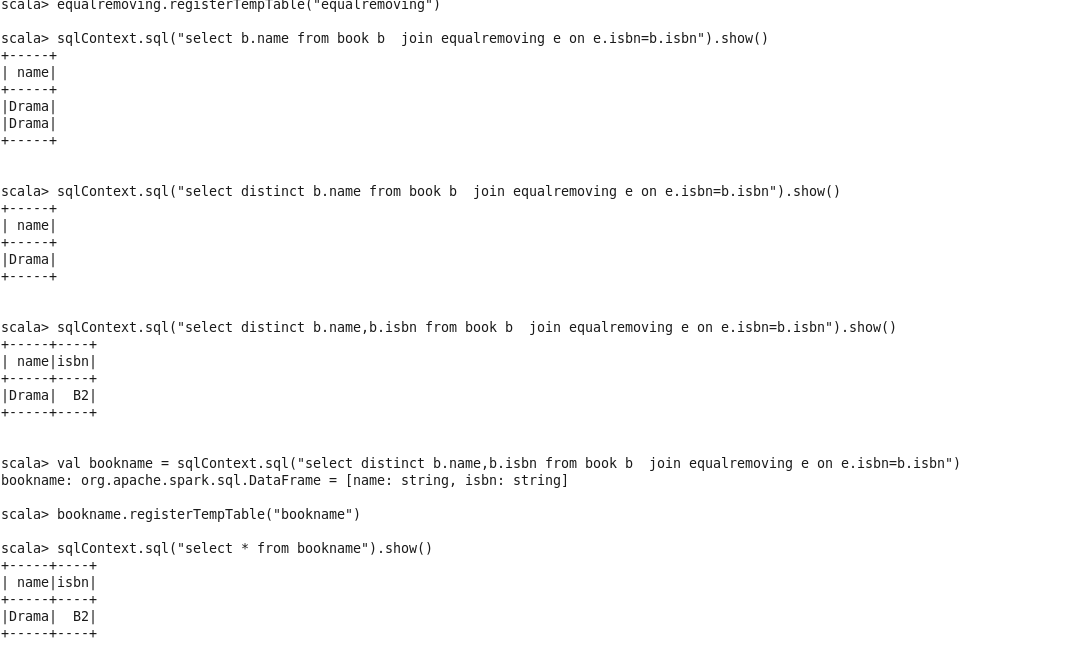
hdfs dfs -cat /user/greesh/kmeans\_python/output/\*

3.1



The total sales of Amazon is 284, Barnes Noble is 55, Borders is 90.Intially loaded file into purchaseRDD then splited by \t (since dataset is seperated by \t) and extracted seller and sales column into sale and then used reduceByKey to get total sales of each seller".

3.2



Drama is the name of book that amazon sells for lowest price compared to all other sellers.