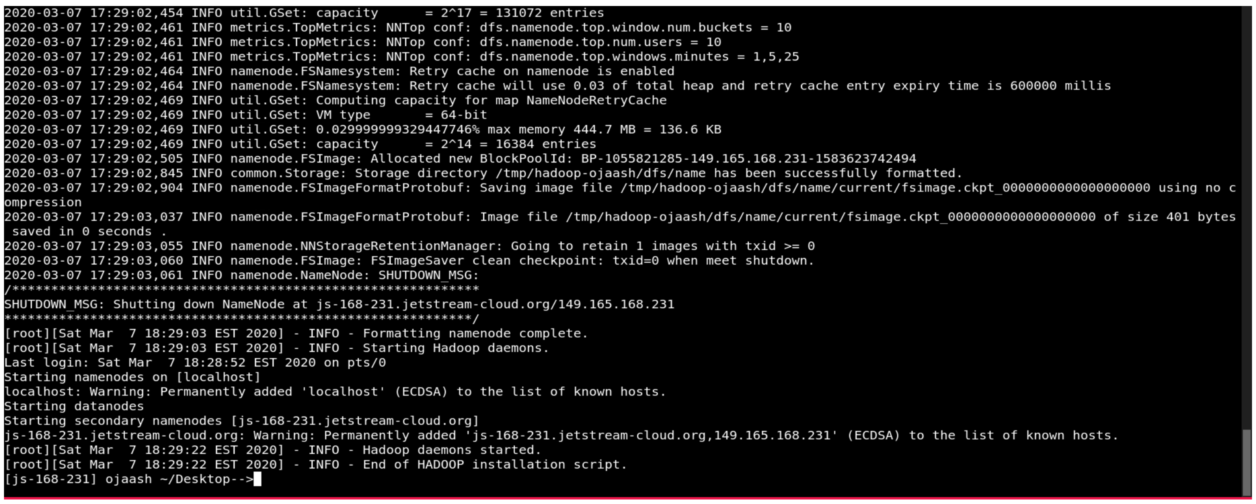
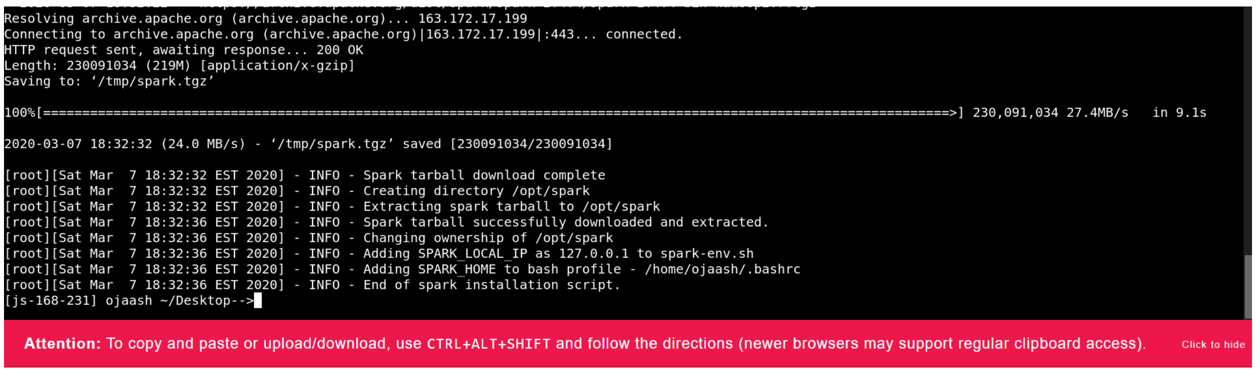
Installation of Hadoop and Spark did not cause me any major hiccups. I have used the steps mentioned in the assignment canvas page to do the installation for both the software. HDFS is initialized in a pseudo-distributed form while spark is used on a local machine here instead of the cluster-based installation. The installation process of the software was easy due to the installer.sh file that was made available to us. I tried my hand at installing Hadoop and spark without the given scripts as well, but I kept running into an error that I was not the root user of the system and permission was denied on most of the commands that I entered in the web shell.

Installing Hadoop:



```
2020-03-07 17:29:02,454 INFO util.GSet: capacity      = 2^17 = 131072 entries
2020-03-07 17:29:02,461 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
2020-03-07 17:29:02,461 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
2020-03-07 17:29:02,461 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
2020-03-07 17:29:02,464 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
2020-03-07 17:29:02,464 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
2020-03-07 17:29:02,469 INFO util.GSet: Computing capacity for map NameNodeRetryCache
2020-03-07 17:29:02,469 INFO util.GSet: VM type       = 64-bit
2020-03-07 17:29:02,469 INFO util.GSet: 0.029999999329447746% max memory 444.7 MB = 136.6 KB
2020-03-07 17:29:02,469 INFO util.GSet: capacity      = 2^14 = 16384 entries
2020-03-07 17:29:02,505 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1055821285-149.165.168.231-1583623742494
2020-03-07 17:29:02,845 INFO common.Storage: Storage directory /tmp/hadoop-ojaash/dfs/name has been successfully formatted.
2020-03-07 17:29:02,904 INFO namenode.FSImageFormatProtobuf: Saving image file /tmp/hadoop-ojaash/dfs/name/current/fsimage.ckpt_0000000000000000000 using no c
ompression
2020-03-07 17:29:03,037 INFO namenode.FSImageFormatProtobuf: Image file /tmp/hadoop-ojaash/dfs/name/current/fsimage.ckpt_0000000000000000000 of size 401 bytes
 saved in 0 seconds .
2020-03-07 17:29:03,055 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
2020-03-07 17:29:03,060 INFO namenode.FSImage: FSImageSaver clean checkpoint: txid=0 when meet shutdown.
2020-03-07 17:29:03,061 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at js-168-231.jetstream-cloud.org/149.165.168.231
************************************************************/
[root][Sat Mar  7 18:29:03 EST 2020] - INFO - Formatting namenode complete.
[root][Sat Mar  7 18:29:03 EST 2020] - INFO - Starting Hadoop daemons.
Last login: Sat Mar  7 18:28:52 EST 2020 on pts/0
Starting namenodes on [localhost]
localhost: Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Starting datanodes
Starting secondary namenodes [js-168-231.jetstream-cloud.org]
js-168-231.jetstream-cloud.org: Warning: Permanently added 'js-168-231.jetstream-cloud.org,149.165.168.231' (ECDSA) to the list of known hosts.
[root][Sat Mar  7 18:29:22 EST 2020] - INFO - Hadoop daemons started.
[root][Sat Mar  7 18:29:22 EST 2020] - INFO - End of HADOOP installation script.
[js-168-231] ojaash ~/Desktop-->
```

Installing Apache Spark:

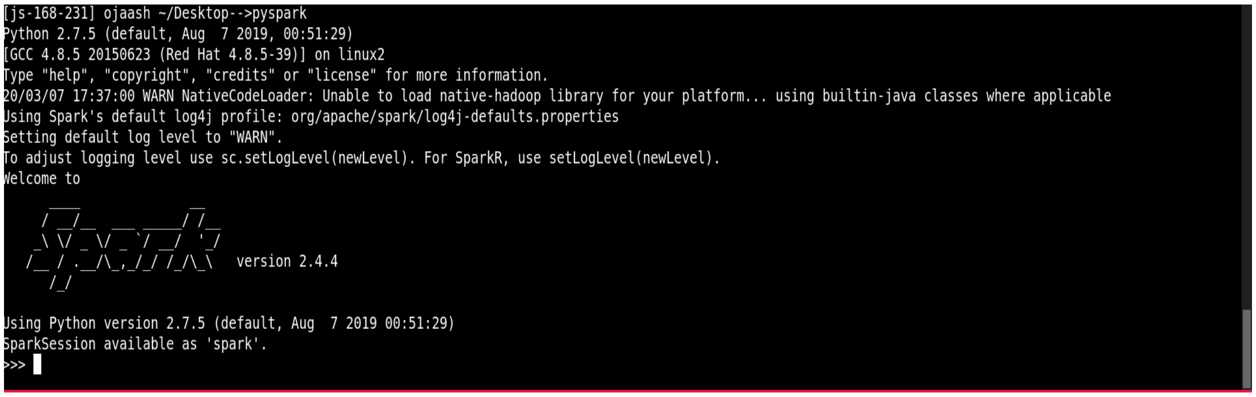


```
Resolving archive.apache.org (archive.apache.org)... 163.172.17.199
Connecting to archive.apache.org (archive.apache.org)|163.172.17.199|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 230091034 (219M) [application/x-gzip]
Saving to: '/tmp/spark.tgz'

100%[=========================================================================================================>] 230,091,034 27.4MB/s   in 9.1s

2020-03-07 18:32:32 (24.0 MB/s) - '/tmp/spark.tgz' saved [230091034/230091034]

[root][Sat Mar  7 18:32:32 EST 2020] - INFO - Spark tarball download complete
[root][Sat Mar  7 18:32:32 EST 2020] - INFO - Creating directory /opt/spark
[root][Sat Mar  7 18:32:32 EST 2020] - INFO - Extracting spark tarball to /opt/spark
[root][Sat Mar  7 18:32:36 EST 2020] - INFO - Spark tarball successfully downloaded and extracted.
[root][Sat Mar  7 18:32:36 EST 2020] - INFO - Changing ownership of /opt/spark
[root][Sat Mar  7 18:32:36 EST 2020] - INFO - Adding SPARK_LOCAL_IP as 127.0.0.1 to spark-env.sh
[root][Sat Mar  7 18:32:36 EST 2020] - INFO - Adding SPARK_HOME to bash profile - /home/ojaash/.bashrc
[root][Sat Mar  7 18:32:36 EST 2020] - INFO - End of spark installation script.
[js-168-231] ojaash ~/Desktop-->
```

Attention: To copy and paste or upload/download, use CTRL+ALT+SHIFT and follow the directions (newer browsers may support regular clipboard access).
Click to hide

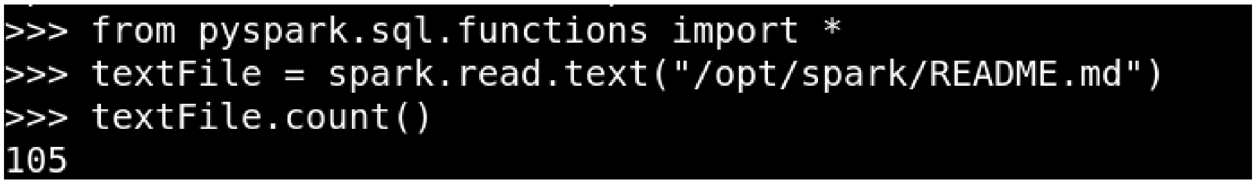
Pyspark has a lot of functions inbuilt which help us to find the total word count in a file, the frequency of occurrence of the words. But here we use a SQL resembling module to get the relevant results from the file. In case of the HDFS, when we run the mapper and reducer py files, we get the word frequencies using a dictionary that is stored in a separate output file.

Opening PySpark:



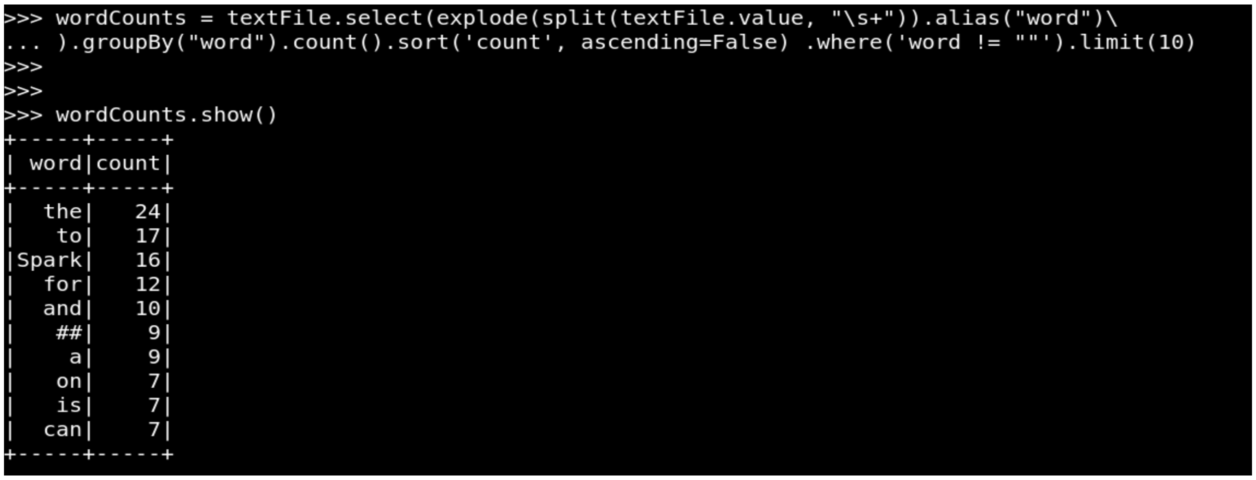
```
[js-168-231] ojaash ~/Desktop-->pyspark
Python 2.7.5 (default, Aug  7 2019, 00:51:29)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-39)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
20/03/07 17:37:00 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 2.4.4
      /_/

Using Python version 2.7.5 (default, Aug  7 2019 00:51:29)
SparkSession available as 'spark'.
>>> 
```

Using Spark to parse a file:



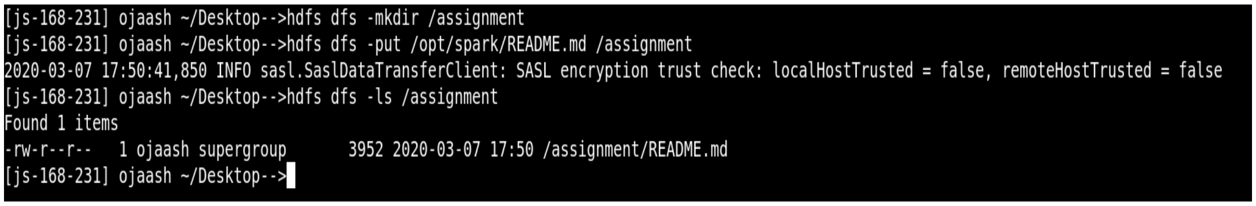
```
>>> from pyspark.sql.functions import *
>>> textFile = spark.read.text("/opt/spark/README.md")
>>> textFile.count()
105
```

Word Count using Spark:



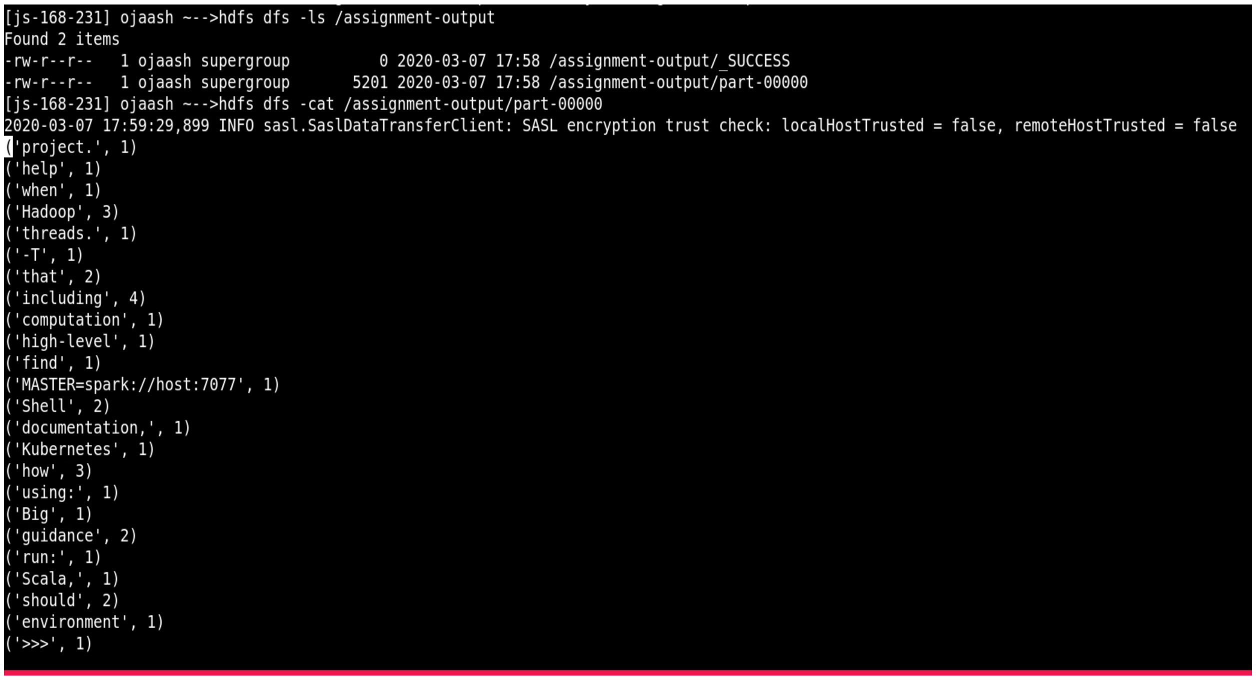
```
>>> wordCounts = textFile.select(explode(split(textFile.value, "\s+")).alias("word")\
... ).groupBy("word").count().sort('count', ascending=False) .where('word != ""').limit(10)
>>>
>>>
>>> wordCounts.show()
+-----+-----+
| word|count|
+-----+-----+
|  the|   24|
|   to|   17|
|Spark|   16|
|  for|   12|
|  and|   10|
|   ##|    9|
|    a|    9|
|   on|    7|
|   is|    7|
|  can|    7|
+-----+-----+
```

Transfer the file to Hadoop HDFS:



```
[js-168-231] ojaash ~/Desktop-->hdfs dfs -mkdir /assignment
[js-168-231] ojaash ~/Desktop-->hdfs dfs -put /opt/spark/README.md /assignment
2020-03-07 17:50:41,850 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
[js-168-231] ojaash ~/Desktop-->hdfs dfs -ls /assignment
Found 1 items
-rw-r--r--   1 ojaash supergroup       3952 2020-03-07 17:50 /assignment/README.md
[js-168-231] ojaash ~/Desktop-->
```

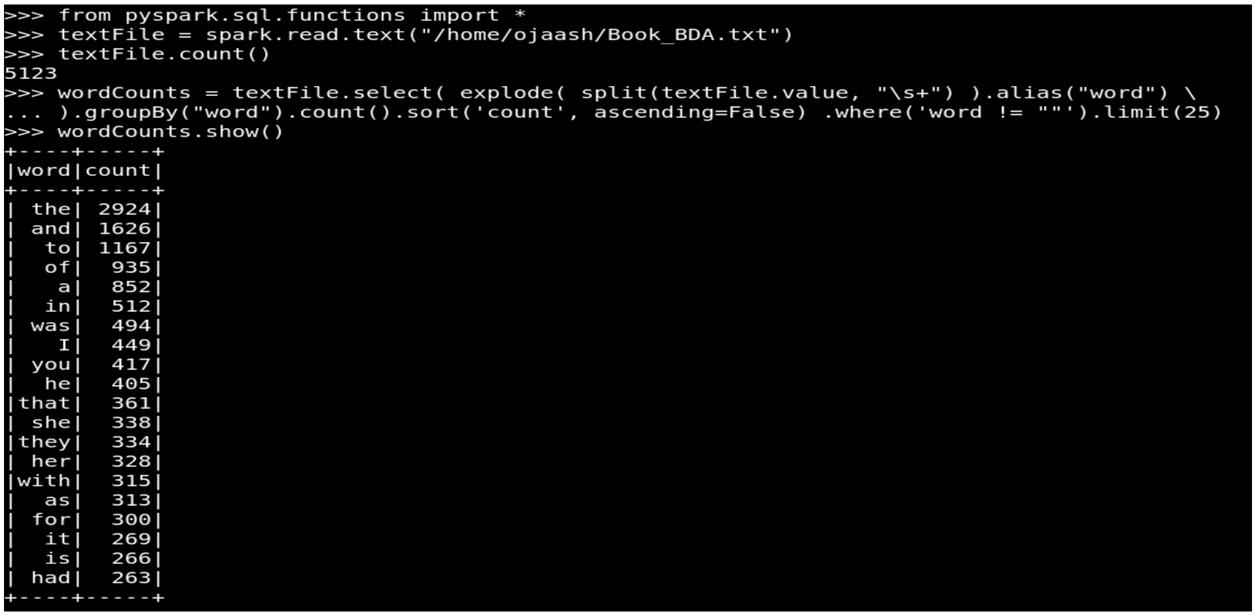
Output of Mapper and Reducer:



```
[js-168-231] ojaash ~-->hdfs dfs -ls /assignment-output
Found 2 items
-rw-r--r--   1 ojaash supergroup          0 2020-03-07 17:58 /assignment-output/_SUCCESS
-rw-r--r--   1 ojaash supergroup       5201 2020-03-07 17:58 /assignment-output/part-00000
[js-168-231] ojaash ~-->hdfs dfs -cat /assignment-output/part-00000
2020-03-07 17:59:29,899 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
('project.', 1)
('help', 1)
('when', 1)
('Hadoop', 3)
('threads.', 1)
('-T', 1)
('that', 2)
('including', 4)
('computation', 1)
('high-level', 1)
('find', 1)
('MASTER=spark://host:7077', 1)
('Shell', 2)
('documentation,', 1)
('Kubernetes', 1)
('how', 3)
('using:', 1)
('Big', 1)
('guidance', 2)
('run:', 1)
('Scala,', 1)
('should', 2)
('environment', 1)
('>>>', 1)
```

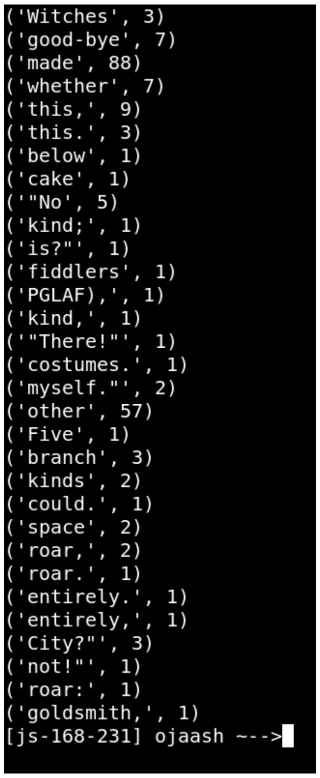
The same set of functionalities for spark an HDFS was tested on a text file named “Book_BDA.txt” which was downloaded randomly from “The Gutenberg Project”. The word count for the book was tested using both the methods as well as the frequencies of the words were verified.

Finding the word counts of the “Book_BDA.txt” file using Spark:



```
>>> from pyspark.sql.functions import *
>>> textFile = spark.read.text("/home/ojaash/Book_BDA.txt")
>>> textFile.count()
5123
>>> wordCounts = textFile.select( explode( split(textFile.value, "\s+") ).alias("word") \
... ).groupBy("word").count().sort('count', ascending=False) .where('word != ""').limit(25)
>>> wordCounts.show()
+----+-----+
|word|count|
+----+-----+
| the| 2924|
| and| 1626|
|  to| 1167|
|  of|  935|
|   a|  852|
|  in|  512|
| was|  494|
|   I|  449|
| you|  417|
|  he|  405|
|that|  361|
| she|  338|
|they|  334|
| her|  328|
|with|  315|
|  as|  313|
| for|  300|
|  it|  269|
|  is|  266|
| had|  263|
+----+-----+
```

The output for Book_BDA.txt using HDFS (mapper.py and reducer.py):



```
('Witches', 3)
('good-bye', 7)
('made', 88)
('whether', 7)
('this,', 9)
('this.', 3)
('below', 1)
('cake', 1)
('"No', 5)
('kind;', 1)
('is?"', 1)
('fiddlers', 1)
('PGLAF),', 1)
('kind,', 1)
('"There!"', 1)
('costumes.', 1)
('myself."', 2)
('other', 57)
('Five', 1)
('branch', 3)
('kinds', 2)
('could.', 1)
('space', 2)
('roar,', 2)
('roar.', 1)
('entirely.', 1)
('entirely,', 1)
('City?"', 3)
('not!"', 1)
('roar:', 1)
('goldsmith,', 1)
[js-168-231] ojaash ~-->
```

The installation of Spark and HDFS is not that complicated to follow. We get the instruction for the same on the respective web pages. However, the installation was done more easily due to the shell script that was provided on canvas. The script covered all the major steps that included in the installation instructions into aa single file and running that single file executes the software for us. Overall, the application of Spark and HDFS to distributed computing and storage systems can be realized very easily from this assignment.