Homework #1

COEN 242 Big data

The goal of hw1 is to practice the RDD programming skill with several basic text analysis tasks, including computing the 1) term frequency (TF), 2) inverse document frequency (IDF), and 3) TF-IDF. Please complete the given TextAnalyzer.py file and follow the instructions below.

Preparation

Data preparation

Download the given zip (hw1.tar.gz) file from the Camino system to your laptop and upload it to your home directory on our AWS cluster system with the following steps:

1. Build a temporary directory on you AWS home directory by: mkdir tmp
2. Open a terminal on your laptop and navigate (by using cd) to the directory where your downloaded file is saved.
3. Upload the file to the cluster system by:
   scp hw1.tar.gz **usrname**@hadoop-aws.engr.scu.edu:/home/**username**/tmp
   Change the usrname as your own.
4. Then by typing the password, you may find the file is uploaded to the tmp directory
5. Unzip the file by: tar -xvzf hw1.tar.gz
6. cd to hw1 and then copy written to the HDFS directory by
   hdfs dfs -copyFromLocal written .

Alternatively, copy the file from /opt/data to your home directory and start from step 5.

Config the spark-submit environment

Open the .bashrc file under the home directory (check it by ls -a ~/), for example, with vim. Edit it by following commands (-> means type enter):

vim ~/.bashrc **->** shit key + G **->** :a **->**

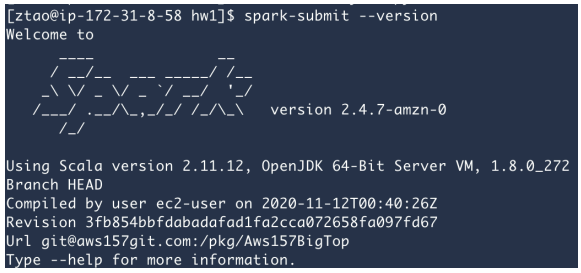
Then, you may use it as a common text editor. Paste the following two lines into this file.

```
export PATH=/usr/lib/hadoop-mapreduce:$PATH
export PATH=/usr/lib/spark/bin:$PATH
```

After pasting, first enter the esc key, then type :wq to exit the file. Type cat ~/.bashrc to check if you have save the file successfully. Try the following command to see if you have added the spark-submit environment.

Spark-submit --version

You should see the version information as shown in the right attached screenshot.



Q1 Compute term-frequency with spark-submit (20 pts)

Use the provided TF(sc, input) function inside the TextAnalyzer.py file to compute the term-frequency values for all the words in the document hotel-california.txt by following:

spark-submit TextAnalyzer.py -m=TF -i="written/fiction/hotel-california.txt" -o=hotel.tf

Try to see if you have saved the hotel.tf under your HDFS directory by

hdfs dfs -ls ./

Start one pyspark interpreter session, read the hotel.tf file and print (you may use pprint by from pprint import pprint) its first **10** lines. Attach the screenshot below.

```
[hxia@linux10621 hw1]$ spark-submit TextAnalyzer.py -m=TF -i="written/fiction/hotel-california.txt" -o=hotel.tf
21/05/10 18:16:05 INFO spark.SparkContext: Running Spark version 2.4.0.cloudera2
21/05/10 18:16:05 INFO spark.SparkContext: Submitted application: Text Analysis
21/05/10 18:16:05 INFO spark.SecurityManager: Changing view acls to: hxia
21/05/10 18:16:05 INFO spark.SecurityManager: Changing modify acls to: hxia
21/05/10 18:16:05 INFO spark.SecurityManager: Changing view acls groups to:
21/05/10 18:16:05 INFO spark.SecurityManager: Changing modify acls groups to:
21/05/10 18:16:05 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(hxia);
 permissions: Set(hxia); groups with modify permissions: Set()
21/05/10 18:16:05 INFO util.Utils: max retries is 16
21/05/10 18:16:05 INFO util.Utils: Successfully started service 'sparkDriver' on port 33160.
```

```
[hxia@linux10621 hw1]$ hdfs dfs -ls ./
Found 4 items
drwx------   - hxia supergroup          0 2021-05-10 18:14 .Trash
drwxrwx---   - hxia supergroup          0 2021-05-10 16:57 .sparkStaging
drwxrwx---   - hxia supergroup          0 2021-05-10 18:16 hotel.tf
drwxrwx---   - hxia supergroup          0 2021-05-08 12:51 written
[hxia@linux10621 hw1]$
```

```
[hxia@linux10621 ~]$ setup cdh-5.16
[hxia@linux10621 ~]$ pyspark
Python 2.7.5 (default, Nov 16 2020, 22:23:17)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-44)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
21/05/10 18:19:47 WARN lineage.LineageWriter: Lineage directory /var/log/spark2/lineage doesn't exist or is not writable. Line
21/05/10 18:19:47 WARN lineage.LineageWriter: Lineage directory /var/log/spark2/lineage doesn't exist or is not writable. Line
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 2.4.0.cloudera2
      /_/

Using Python version 2.7.5 (default, Nov 16 2020 22:23:17)
SparkSession available as 'spark'.
>>> from pprint import pprint
>>> tf_read = sc.textFile('hotel.tf')
>>> pprint(tf_read.take(10))
[u"('', 3)",
 u"(u'all', 48)",
 u"(u'savior', 1)",
 u"(u'dance', 4)",
 u"(u'mattered', 1)",
 u"(u'ephemeral', 1)",
 u"(u'swirls', 1)",
 u"(u'blisters', 1)",
 u"(u'known', 3)",
 u"(u'protest', 1)"]
>>>
```

Q2 Compute inverse document-frequency with spark-submit (30 pts)

Complete the IDF(sc, input) function inside the TextAnalyzer.py file to compute the inverse document-frequency (idf) values for all the words inside the given written corpus. You may submit your spark program after implementing the IDF function by:

spark-submit TextAnalyzer.py -m=IDF -i="written/* " -o=vocab.idf

Start one pyspark interpreter session, read the vocab.idf file and pprint its first **10** lines. Attach the screenshot below. Hint: compute idf by idf(word) = np.log(# documents/(df(word)+1)).

```
[hxia@linux10621 hw1]$ spark-submit TextAnalyzer.py -m=IDF -i="written/*" -o=vocab.idf
21/05/10 18:26:47 INFO spark.SparkContext: Running Spark version 2.4.0.cloudera2
21/05/10 18:26:47 INFO spark.SparkContext: Submitted application: Text Analysis
21/05/10 18:26:47 INFO spark.SecurityManager: Changing view acls to: hxia
21/05/10 18:26:47 INFO spark.SecurityManager: Changing modify acls to: hxia
21/05/10 18:26:47 INFO spark.SecurityManager: Changing view acls groups to:
21/05/10 18:26:47 INFO spark.SecurityManager: Changing modify acls groups to:
21/05/10 18:26:47 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(hxia); groups with view permiss
 permissions: Set(hxia); groups with modify permissions: Set()
21/05/10 18:26:47 INFO util.Utils: max retries is 16
21/05/10 18:26:47 INFO util.Utils: Successfully started service 'sparkDriver' on port 35211.
21/05/10 18:26:47 INFO spark.SparkEnv: Registering MapOutputTracker
21/05/10 18:26:47 INFO spark.SparkEnv: Registering BlockManagerMaster
21/05/10 18:26:47 INFO storage.BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting topology information
21/05/10 18:26:47 INFO storage.BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
21/05/10 18:26:47 INFO storage.DiskBlockManager: Created local directory at /tmp/blockmgr-86947b2b-f3b8-4faa-82e4-5ccb6178500a
21/05/10 18:26:47 INFO memory.MemoryStore: MemoryStore started with capacity 366.3 MB
21/05/10 18:26:47 INFO spark.SparkEnv: Registering OutputCommitCoordinator
21/05/10 18:26:48 INFO util.log: Logging initialized @6452ms
21/05/10 18:26:48 INFO server.Server: jetty-9.3.z-SNAPSHOT, build timestamp: unknown, git hash: unknown
21/05/10 18:26:48 INFO server.Server: Started @6530ms
```

```
[hxia@linux10621 hw1]$ hdfs dfs -ls ./
Found 5 items
drwx------   - hxia supergroup          0 2021-05-10 18:14 .Trash
drwxrwx---   - hxia supergroup          0 2021-05-10 18:19 .sparkStaging
drwxrwx---   - hxia supergroup          0 2021-05-10 18:16 hotel.tf
drwxrwx---   - hxia supergroup          0 2021-05-10 18:27 vocab.idf
drwxrwx---   - hxia supergroup          0 2021-05-08 12:51 written
[hxia@linux10621 hw1]$
```

```
>>> idf_read = sc.textFile('vocab.idf')
>>> pprint(idf_read.take(10))
[u"('', 0.0026702285558788921)",
 u"(u'aided', 4.5406316648505198)",
 u"(u'unscientific', 5.2337788454104652)",
 u"(u'revetts', 5.2337788454104652)",
 u"(u'systematic', 4.8283137373023015)",
 u"(u'pravastatin', 5.2337788454104652)",
 u"(u'moskowitz', 5.2337788454104652)",
 u"(u'yellow', 3.8474844842905749)",
 u"(u'four', 1.9379419794061366)",
 u"(u'gag', 5.2337788454104652)"]
```

Q3 Compute TF-IDF with spark-submit (30 pts)

Complete the TFIDF(sc, TFfile, IDFfile) function inside the TextAnalyzer.py file to compute the tf-idf values for all the words in the document hotel-california.txt. You may submit your spark program after implementing the TFIDF function by:

spark-submit TextAnalyzer.py -m=TFIDF -i=hotel.tf -o=hotel.tfidf --idfvalues=vocab.idf

Start one pyspark interpreter session, read the hotel.tfidf file and pprint its first **10** lines. Attach the screenshot below. Hint: compute TF-IDF by tf-idf(word) = tf(word) * idf(word).

```
[hxia@linux10621 hw1]$ spark-submit TextAnalyzer.py -m=TFIDF -i=hotel.tf -o=hotel.tfidf --idfvalues=vocab.idf
21/05/10 18:30:32 INFO spark.SparkContext: Running Spark version 2.4.0.cloudera2
21/05/10 18:30:32 INFO spark.SparkContext: Submitted application: Text Analysis
21/05/10 18:30:32 INFO spark.SecurityManager: Changing view acls to: hxia
21/05/10 18:30:32 INFO spark.SecurityManager: Changing modify acls to: hxia
21/05/10 18:30:32 INFO spark.SecurityManager: Changing view acls groups to:
21/05/10 18:30:32 INFO spark.SecurityManager: Changing modify acls groups to:
21/05/10 18:30:32 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(hxia); gr
 permissions: Set(hxia); groups with modify permissions: Set()
21/05/10 18:30:32 INFO util.Utils: max retries is 16
21/05/10 18:30:32 INFO util.Utils: Successfully started service 'sparkDriver' on port 41303.
21/05/10 18:30:32 INFO spark.SparkEnv: Registering MapOutputTracker
21/05/10 18:30:32 INFO spark.SparkEnv: Registering BlockManagerMaster
21/05/10 18:30:32 INFO storage.BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting topology information
21/05/10 18:30:32 INFO storage.BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
21/05/10 18:30:32 INFO storage.DiskBlockManager: Created local directory at /tmp/blockmgr-fe5e2537-d0aa-46aa-800f-0de1f039ece1
21/05/10 18:30:32 INFO memory.MemoryStore: MemoryStore started with capacity 366.3 MB
```

```
[hxia@linux10621 hw1]$ hdfs dfs -ls ./
Found 6 items
drwx------   - hxia supergroup          0 2021-05-10 18:14 .Trash
drwxrwx---   - hxia supergroup          0 2021-05-10 18:19 .sparkStaging
drwxrwx---   - hxia supergroup          0 2021-05-10 18:16 hotel.tf
drwxrwx---   - hxia supergroup          0 2021-05-10 18:30 hotel.tfidf
drwxrwx---   - hxia supergroup          0 2021-05-10 18:27 vocab.idf
drwxrwx---   - hxia supergroup          0 2021-05-08 12:51 written
[hxia@linux10621 hw1]$
```

```
>>> tfidf_read = sc.textFile('hotel.tfidf')
>>> pprint(tfidf_read.take(10))
[u"('', 0.008010685667636677)",
 u"(u'looking', 1.9015743352352616)",
 u"(u'malfunctioned', 10.46755769082093)",
 u"(u'contributed', 3.361976668508874)",
 u"(u'hallucinating', 5.233778845410465)",
 u"(u'conversational', 5.233778845410465)",
 u"(u'brought', 2.5257286443082556)",
 u"(u'music', 2.835883572612095)",
 u"(u'machine', 3.4420193761824107)",
 u"(u'hor', 5.233778845410465)"]
>>>
```

Q4 Remove stopwords and query word with pyspark (20 pts)

Start one pyspark interpreter session, read the hotel.tfidf file and finish the following two tasks:

- Remove all the stopwords in the hotel.tfidf file and sort the remaining ones by tf-idf values (**descending**). PPrint the first **10** words and attach the screenshot below.
- Use spark RDD API to query the TF-IDF value for the word '**round**' in the hotel.tfidf file. Give the value and solution (one line spark codes) below.

(1)

```
>>> tfidf_read = sc.textFile('hotel.tfidf')
>>> tfidf_read.count()
1577
>>> new_tfidf = tfidf_read.map(eval)
>>> stopwords = [line.strip('\n') for line in open('english')]
>>> vocab = new_tfidf.filter(lambda x: x[0] not in stopwords)
>>> vocab.count()
1464
>>> sorted_vocab = vocab.sortBy(lambda x:x[1], ascending=False)
>>> pprint(sorted_vocab.take(10))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'pprint' is not defined
>>> from pprint import pprint
>>> pprint(sorted_vocab.take(10))
[(u'adrienne', 177.94848074395583),
 (u'ship', 115.97890985524369),
 (u'zheng', 101.39458848334833),
 (u'ray', 82.70333113484712),
 (u'sarah', 82.70333113484712),
 (u'kishori', 68.03912499033605),
 (u'tiffany', 57.939764847627615),
 (u'captain', 51.75320639989627),
 (u'said', 50.01055825818844),
 (u'jefferson', 49.62199868090828)]
>>>
```

(2)

```
new_tfidf.filter(lambda x:'round' in x).collect()
```

```
>>> new_tfidf.filter(lambda x:'round' in x).collect()
[(u'round', 9.863606089065456)]
```