Group: Dewar Tan, Paul Cabrera, Alexios Ladikos, Cameron Cho
Cosi129a
Due: Friday, November 18, 2016

Assignment 3: Using Mahout for Naïve-Bayes Model Training and Testing

Code Design

The assignment was to use Mahout's utilities to train and test a Naïve-Bayes Classifier Model and see how accurate it was. The main task, then, was to pre-process the given data so that Mahout could use it effectively. The pre-processing program, in concept, is fairly straightforward. It first takes in two files—one file that contains an index of key-value pairs (in form of <Name, List<StringInteger>>) that state what words appear in a document and with what frequency, and another file that has a list of people with their profession. The code parses the two files before placing all of the words from the index into a "dictionary" vector that holds all of the words that appear across all documents. Once the dictionary is formed, the program then assigns every document a vector equal in length to the dictionary and fills it with the word frequencies in each document. The key of every vector is a profession matching found in profession.txt that cross-matches with the name found in lemma-index. For example "Robert Frost : Poet" would match with "Robert Frost <word, frequency> …" and from there the vector generated will be as such "Poet <word, frequency>… " of all words from the dictionary. Word frequencies that are not found in the dictionary are given a value of zero-index and people with multiple professions will have multiple vectors. With every document assigned a vector, completing the pre-processing. We then classified the rest of the unclassified people by looking at the top 3 professions by frequency counts for professions in which obtained from the list in professions.txt. Now ready, the data is then split into training and testing sets before being passed into Mahout, using the appropriate training and testing commands on the data set and evaluating the classifier's effectiveness.

Challenges

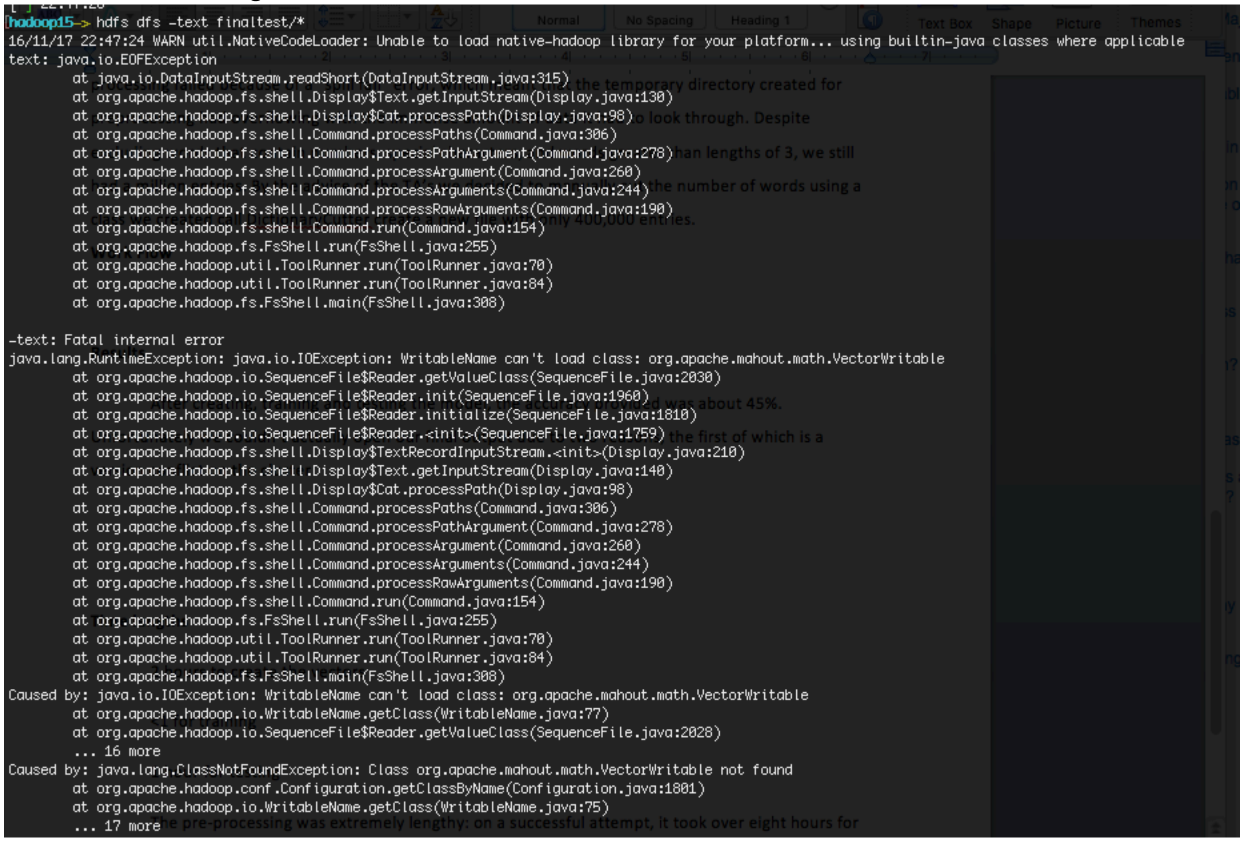
The first issue encountered was the parsing of the lemma-index data. In our previous assignment we took care to parse and remove any special characters that might alter our splitting methods. However, because we decided to use the lemma file given for the assignment we were thrown off initially on overlooking special word frequencies such as <shift-5,-4,1> which we parsed out the frequency due to the comma at -4. Unfortunately, we didn't find this error until we had moved onto training a model and in which case threw a non-descriptive error with no documentation on how to solve it. We luckily realized this might have been the case when we found out that some word frequencies had multiple commas in them.

The second issue encountered was dictionary size. On the cluster, the compiling phase of pre-processing failed because of a "spill fail" error, which meant that the temporary directory created for preprocessing had overflowing with the immense amount of data it had to look through. Despite excluding words that contain numbers, special characters, and words greater than lengths of 3, we still had a million entries. By the advice of the TA's we decided to manually cut the number of words using a class we created call DictionaryCutter create a new file with only 400,000 entries.

Lastly, the final issue we encountered was running the entire lemma-index set as a whole. We had difficulties parsing between classifiable, the ones in which names are present from profession.txt, and the unclassifiable when creating the vector. Hence we created different classes to split them up first.

Results

After creating, training and testing the model, the accuracy provided was about 45%. Unfortunately we couldn't actually open our final output due to two reasons, the first of which is a version conflict on the cluster seen below in **Figure 1.** Running “hdfs dfs –text finaltest/*” would have allowed us to peer into the file in its plain format however here we see that the VectorWritable class is not found even though mahout is available on the cluster.

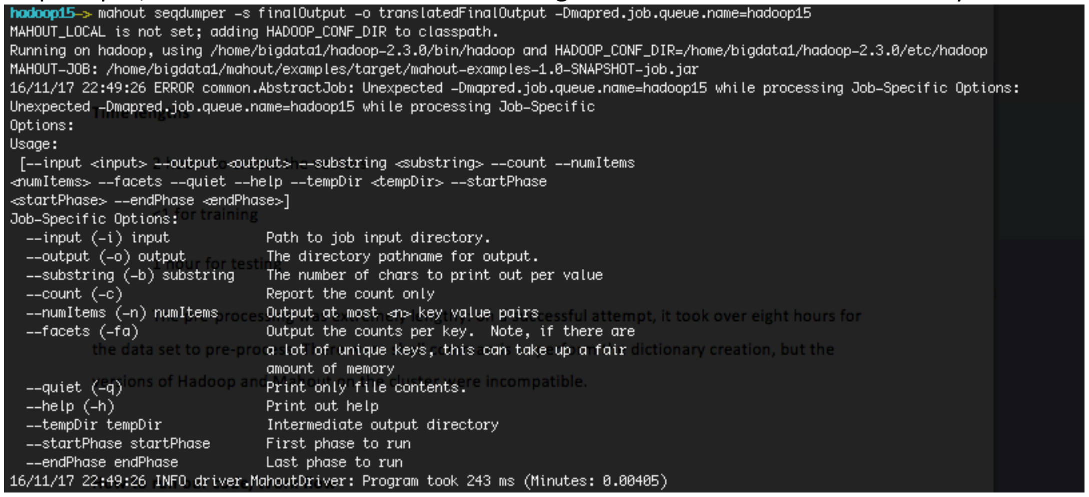


```
hadoop15-> hdfs dfs -text finaltest/*
16/11/17 22:47:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
text: java.io.EOFException
        at java.io.DataInputStream.readShort(DataInputStream.java:315)
        at org.apache.hadoop.fs.shell.Display$Text.getInputStream(Display.java:130)
        at org.apache.hadoop.fs.shell.Display$Cat.processPath(Display.java:98)
        at org.apache.hadoop.fs.shell.Command.processPaths(Command.java:306)
        at org.apache.hadoop.fs.shell.Command.processPathArgument(Command.java:278)
        at org.apache.hadoop.fs.shell.Command.processArgument(Command.java:260)
        at org.apache.hadoop.fs.shell.Command.processArguments(Command.java:244)
        at org.apache.hadoop.fs.shell.Command.processRawArguments(Command.java:190)
        at org.apache.hadoop.fs.shell.Command.run(Command.java:154)
        at org.apache.hadoop.fs.FsShell.run(FsShell.java:255)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:70)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:84)
        at org.apache.hadoop.fs.FsShell.main(FsShell.java:308)

-text: Fatal internal error
java.lang.RuntimeException: java.io.IOException: WritableName can't load class: org.apache.mahout.math.VectorWritable
        at org.apache.hadoop.io.SequenceFile$Reader.getValueClass(SequenceFile.java:2030)
        at org.apache.hadoop.io.SequenceFile$Reader.init(SequenceFile.java:1960)
        at org.apache.hadoop.io.SequenceFile$Reader.initialize(SequenceFile.java:1810)
        at org.apache.hadoop.io.SequenceFile$Reader.<init>(SequenceFile.java:1759)
        at org.apache.hadoop.fs.shell.Display$TextRecordInputStream.<init>(Display.java:210)
        at org.apache.hadoop.fs.shell.Display$Text.getInputStream(Display.java:140)
        at org.apache.hadoop.fs.shell.Display$Cat.processPath(Display.java:98)
        at org.apache.hadoop.fs.shell.Command.processPaths(Command.java:306)
        at org.apache.hadoop.fs.shell.Command.processPathArgument(Command.java:278)
        at org.apache.hadoop.fs.shell.Command.processArgument(Command.java:260)
        at org.apache.hadoop.fs.shell.Command.processArguments(Command.java:244)
        at org.apache.hadoop.fs.shell.Command.processRawArguments(Command.java:190)
        at org.apache.hadoop.fs.shell.Command.run(Command.java:154)
        at org.apache.hadoop.fs.FsShell.run(FsShell.java:255)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:70)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:84)
        at org.apache.hadoop.fs.FsShell.main(FsShell.java:308)
Caused by: java.io.IOException: WritableName can't load class: org.apache.mahout.math.VectorWritable
        at org.apache.hadoop.io.WritableName.getClass(WritableName.java:77)
        at org.apache.hadoop.io.SequenceFile$Reader.getValueClass(SequenceFile.java:2028)
        ... 16 more
Caused by: java.lang.ClassNotFoundException: Class org.apache.mahout.math.VectorWritable not found
        at org.apache.hadoop.conf.Configuration.getClassByName(Configuration.java:1801)
        at org.apache.hadoop.io.WritableName.getClass(WritableName.java:75)
        ... 17 more
```

Figure 1. Conflict on the cluster for VectorWritable

A quick Google search on how to solve this issue recommended we use a mahout utility called seqdumper, however the cluster has the wrong version of mahout and the utility does not exist **Figure 2.**



```
hadoop15-> mahout seqdumper -s finalOutput -o translatedFinalOutput -Dmapred.job.queue.name=hadoop15
MAHOUT_LOCAL is not set; adding HADOOP_CONF_DIR to classpath.
Running on hadoop, using /home/bigdata1/hadoop-2.3.0/bin/hadoop and HADOOP_CONF_DIR=/home/bigdata1/hadoop-2.3.0/etc/hadoop
MAHOUT-JOB: /home/bigdata1/mahout/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar
16/11/17 22:49:26 ERROR common.AbstractJob: Unexpected -Dmapred.job.queue.name=hadoop15 while processing Job-Specific Options:
Unexpected -Dmapred.job.queue.name=hadoop15 while processing Job-Specific
Options:
Usage:
 [--input <input> --output <output> --substring <substring> --count --numItems
<numItems> --facets --quiet --help --tempDir <tempDir> --startPhase
<startPhase> --endPhase <endPhase>]
Job-Specific Options:
  --input (-i) input            Path to job input directory.
  --output (-o) output          The directory pathname for output.
  --substring (-b) substring    The number of chars to print out per value
  --count (-c)                  Report the count only
  --numItems (-n) numItems      Output at most <n> key value pairs
  --facets (-fa)                Output the counts per key.  Note, if there are
                                a lot of unique keys, this can take up a fair
                                amount of memory
  --quiet (-q)                  Print only file contents.
  --help (-h)                   Print out help
  --tempDir tempDir             Intermediate output directory
  --startPhase startPhase       First phase to run
  --endPhase endPhase           Last phase to run
16/11/17 22:49:26 INFO driver.MahoutDriver: Program took 243 ms (Minutes: 0.00405)
```

Figure 2. Seqdumper command not found

Several other group members have tried consulting with the TA's on the matter but there was no avail in how to solve the issue at hand. Our last attempt was to try to export the file and installing a local version of mahout but installation itself had its own set of issues and at this point we felt we were headed down a rabbit hole for this assignment given that we had no remaining instruction.

Time lengths

~3 hours to create the vectors

< 1 hour for training

1 hour for testing

How to run our code/Work flow

1. First we ran CreateDictionary to produce the dictionary and we moved the output from hdfs to the local in ssh using –copyToLocal hdfs command.

yarn jar pa3.jar CreateDictionary -Dmapred.job.queue.name=hadoop15 /shared/lemma-index/part-r-00000 outputPath

2. Then we ran Dictionarycutter to cut down the number of entries and produce's a dictionaryMod.txt file. We then place this file back into the hdfs

3. We place the profession.txt file into the hdfs and separate the classified using SeparateClassifiable and the unclassifiable using SeparateUnclassifiable. We put in the following arguments:

yarn jar pa3.jar SeparateClassifiable -Dmapred.job.queue.name=hadoop15 profession.txt /shared/lemma-index/part-r-00000 outputPath

4. We create the vector seq file with the classifiable people using CreateVector with the following arguments

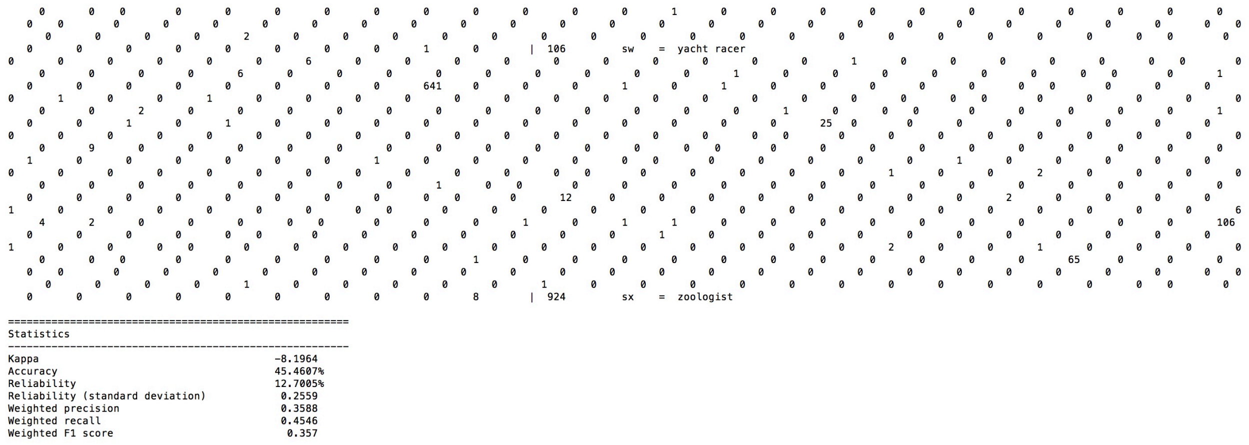
yarn jar pa3.jar CreateVector -Dmapred.job.queue.name=hadoop15 profession.txt dictionaryMod.txt classifiable outputPath

5. We then classified the rest of the unknowns against profession.txt

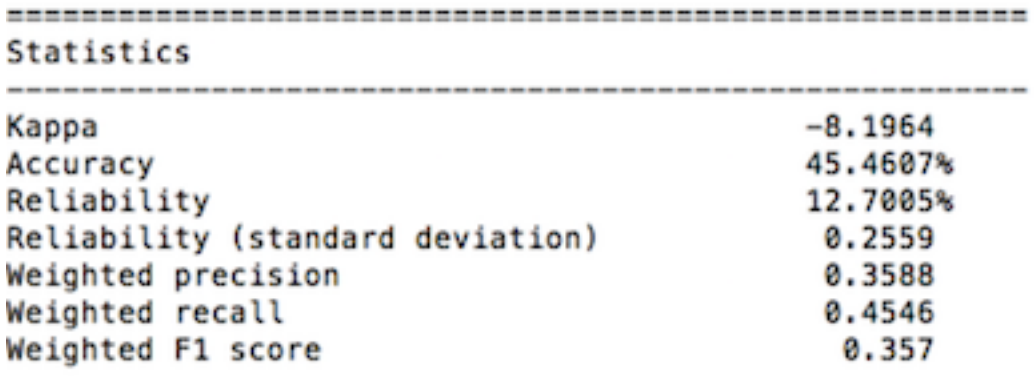
yarn jar pa3.jar ClassifyUnknown -Dmapred.job.queue.name=hadoop15 profession.txt unclassifiable classifyUnknown

6. From here out we used all mahout commands. We placed all of them in a directory in hdfs and run the mahout commands for seq2directory and seq2sparse following the commands from this source: https://mahout.apache.org/users/classification/twenty-newsgroups.html

7. We split and trained (using trainnb) and tested (testnb) the classifier and got the final result of 45% accuracy seen below in **Figure 3a. &3b**



```
     0     0  0     0     0     0     0     0     0     0     0     0     0     1     0     0     0     0     0     0     0     0     0     0     0
   0  0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0  0     0
     0     0     0     0     2     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0  0     0
   0     0     0     0     0     0     0     0     1     0        |  106        sw    =  yacht racer
0     0     0     0     0     0     6     0     0     0     0     0     0     0     0     0     0     1     0     0     0     0     0     0  0     0
     0     0     0     0     6     0     0     0     0     0     0     0     0     0     1     0     0     0     0     0     0     0  0     0     1
   0     0     0     0     0     0     0     0   641     0     0     0     1     0     1     0     0     0     0     0     0     0     0     0     0
0     1     0     0     1     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0  0     0     0     0     0     0
     0     0     2     0     0     0     0     0     0     0     0     0     0     0     0     1     0     0  0     0     0     0     0     0     1
   0     0     1     0     1     0     0     0     0     0     0     0     0     0     0     0    25  0     0     0     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0  0     0     0     0     0     0     0     0     0     0
     0     9     0     0     0     0     0     0     0     0     0     0     0     0  0     0     0     0     0     0     0     0     0     0     0
   1     0     0     0     0     0     0     1     0     0     0     0     0  0     0     0     0     0     0     1     0     0     0     0     0
0     0     0     0     0     0     0     0     0     0     0     0  0     0     0     0     0     0     1     0     0     2     0     0     0     0
     0     0     0     0     0     0     0     0     1     0  0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0  0     0    12     0     0     0     0     0     0     0     0     2     0     0     0     0
1     0     0     0     0     0     0     0  0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     6
     4     2     0     0     0     0  0     0     0     0     1     0     1     1     0     0     0     0     0     0     0     0     0     0   106
   0     0     0     0     0  0     0     0     0     0     0     0     0     1     0     0     0     0     0     0     0     0     0     0     0
1     0     0     0  0     0     0     0     0     0     0     0     0     0     0     0     0     0     2     0     0     1     0     0     0     0
     0     0  0     0     0     0     0     0     0     1     0     0     0     0     0     0     0     0     0     0     0    65     0     0     0
   0  0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0  0     0
     0     0     0     0     1     0     0     0     0     0     1     0     0     0     0     0     0     0     0     0     0     0     0  0     0
   0     0     0     0     0     0     0     0     0     8        |  924        sx    =  zoologist
=======================================================
Statistics
-------------------------------------------------------
Kappa                                          -8.1964
Accuracy                                       45.4607%
Reliability                                    12.7005%
Reliability (standard deviation)                0.2559
Weighted precision                              0.3588
Weighted recall                                 0.4546
Weighted F1 score                               0.357
```



```
=======================================================
Statistics
-------------------------------------------------------
Kappa                                          -8.1964
Accuracy                                       45.4607%
Reliability                                    12.7005%
Reliability (standard deviation)                0.2559
Weighted precision                              0.3588
Weighted recall                                 0.4546
Weighted F1 score                               0.357
```

Figure 3a & 3b. Final output percentage