CPSC 5330 Spring 2023

Lab 4: Hadoop on AWS

The main goal of this lab is to get you running Map Reduce code on AWS. The code you will implement will do a better job of tokenizing the words in the corpus.

Recall that the previous policy of (a) split on whitespace, then (b) lowercase all letters, then (c) filtering out all non-letters leads to some tokens that don't intuitively look like words that would be useful in a search engine. Maybe we can do better.

You will write Hadoop streaming program(s) to process the text corpus, which will be stored in a common location on S3 (see below). You will run the streaming job in EMR and the output will be stored in S3.

You will use the following rules to tokenize the text.
For each text line do the following steps *in order*:

1. Convert letters to lower case
2. Split on whitespace and punctuation to get a set of *words*. So the line
   `"hello world's best_ever fair"`
   would produce this list of words:
   `['hello', 'world', 's', 'best', 'ever', 'fair']`

   HINT: the underscore character is tricky. The regular expression metacharacter `\W` matches letters, digits, *and the underscore character*, so it isn't exactly what you need. You want to consider underscore to be a punctuation character too.
3. Remove words that have 1 or fewer characters
4. Remove words that are sequences of digits (Filter words that are only digits, like `'123'`, but not strings that contain digits like `'a1b2c3'`)
5. Remove "stop words"

Stop words are common words like `'a'` and `'the'` that appear frequently in most documents and are rarely useful in determining query relevance. The common method for processing stop words is to have a stop word list, and simply remove tokens that appear in the list. Stopwords for this assignment are in a file `stopwords.py` appearing in the course repository. You can insert the code in this file directly into your mapper code.

The resulting words, after processing and filtering, are known as the *tokens* in the document.

You will then gather some statistics about the document tokenization. Your reducer output will have one record per document, with the following tab-separated fields

1. Document ID -- same as previous lab. For example, `'austen-persuasion'`
2. The total number of *words* in the document, prior to any processing (i.e. after splitting on whitespace and punctuation, but before any filtering)
3. The number of stopword tokens in the document
4. The total number of tokens in the document after all processing
5. The number of *unique* tokens in the document after all processing

Here is a sample output record. There should be one record per document in your reducer output file(s)

```
shakespeare-hamlet      30271   15122   13509   4409
```

AWS Details

- The set of text files is here -- `s3://cpsc5330s23/books/`
- Create a folder `lab4` to hold your solution
- Create a folder `code` inside of `lab4` to hold your mapper and reducer
- When you run the EMR job, have the output go to a folder output inside of `lab4`

Submission

We will review your submission directly on AWS, so you will submit a single PDF file containing the following information

1. The ID of the EMR cluster that produced your output
2. A retrospective report a reflection on the assignment, with the following components
    a. Your name
    b. How much time you spent on the assignment
    c. If parts of the assignments are not fully working, which parts and what the problem(s) are
    d. Were there aspects of the assignment that were particularly challenging? Particularly confusing?
    e. What were the main learning take-aways from this lab – that is, did it introduce particular concepts or techniques that might help you as an analyst or engineer in the future?