Exercise 12

Due Friday August 06 2021.

Some files are provided that you need below: E12.zip. As usual, **you may not write any loops** and **all calculations must be done with Spark DataFrames**.

Word Count

Counting words in text files is often used as a "hello world" example for big data tools: it would be odd to not make you do it at some point. The task is: given a collection of (English-like) text files, count the number of occurrences of each word in the text. We expect to find that the words like "the", "to", and "and" are the most frequent.

This task is strangely a little trickier with Spark DataFrames than RDDs, but still not too hard. [You must do this question with DataFrames.] Things that have to be done:

1. Read lines from the files with `spark.read.text`.
2. Split the lines into words with the regular expression below. Use the `split` and `explode` functions. Normalize all of the strings to lower-case (so "word" and "Word" are not counted separately.)
3. Count the number of times each word occurs.
4. Sort by decreasing count (i.e. frequent words first) and alphabetically if there's a tie.
5. Notice that there are likely empty strings being counted: remove them from the output. (They come from spaces at the start/end of lines in the original input.)
6. Write results as CSV files with the **word** in the first column, and **count** in the second (uncompressed: they aren't big enough to worry about).

Here is a regular expression that can be used to split words:

```
import string, re
wordbreak = r'[%s\s]+' % (re.escape(string.punctuation),)  # regex that matches spaces and/or punctuation
```

I have been deliberately vague above: you'll have to look at the Spark DataFrame docs *(https://spark.apache.org/docs/latest/api/python/reference/pyspark.sql.html)* for some details

Create a program `wordcount.py` that takes input and output directories on the command line. Output should be uncompressed CSV as described above.

Explaining Pup Inflation

In Exercise 2, we started looking at the @dog_rates *(https://twitter.com/dog_rates)* Twitter account. There, we did initial analysis of the data and presented some results. In Exercise 7, we determined that there was a statistically-significant trend in the ratings.

In this exercise, will will present our results. **Write a summary** of your analysis of these ratings. You should think of this as **a blog post**: written for the general audience (but the general audience who cares about slightly quirky data science analysis of Twitter accounts). Write with that audience in mind: not too technical; general interest explanations of your findings.

You must include **at least two visualizations** of the data. The scatter plot with fit line will likely be one (perhaps with better labels, etc than you produced originally). Come up with another that helps explain what's happening in the data.

You may want to try the Seaborn *(https://seaborn.pydata.org/)* visualization package (but aren't required to). If nothing else, it will make your matplotlib plots slightly nicer if you start with this:

```
import seaborn
seaborn.set()
```

Total length should be maybe a half page of text (not including the visualizations). Save/export your document as a PDF `pup_inflation.pdf`.

Submitting

Submit your files through CourSys for Exercise 12.

Updated Wed April 07 2021, 09:16 by ggbaker.