Laboratory 2 - Spark RDD

The objective of this laboratory is to start playing around with Apache Spark, processing large text files.

Write your code using the `jupyter.polito.it` web interface.

- You can use **Jupyter notebooks** by using the **PySpark (Local)** kernel to run it locally on the gateway or **PySpark (YARN)** kernel to run it on the cluster.
- You can create **scripts** by generating a text file with the extension **.py** and then run it with the shell command `spark-submit`, either locally (`--master local`) or on the cluster (`--master yarn`). For more details see lecture slides and part 1.3 of Lab 1.

1. Problem Specification

On HDFS, at the path `/data/students/bigdata_internet/lab2/word_frequency.tsv` you have a large text file with the word frequencies in food reviews in Amazon, in the format `word\tfreq`, where freq is an integer. Your task is to write a Spark application (the combination of steps 1.1, 1.2 and 1.3) to filter these results, analyze the filtered data and compute some statistics on them.

Start analyzing what is inside the file.

Questions: How can you take a look at some of the lines of the file? How many words does it contain? **Hint**: you can use the the **take** action and the **count** action.

1.1 Filter words starting with a specified prefix

The first step you should implement the following task:

- Keep only the lines containing words that start with a prefix (a string) that is a parameter of the script, i.e., a global variable `PREFIX`.

The result is the set of lines (`word\tfreq`) that satisfy the filtering operation.

Print on the standard output:

- The number of selected lines
- The maximum frequency (`maxfreq`) among the ones of the selected lines (i.e., the maximum value of `freq` in the lines obtained by applying the filter).

1.2 Filter most frequent words

In the second part of your application, among the lines selected by the first filter, you have to apply another filter to select only the most frequent words. Specifically, your application must select those lines that contain words with a frequency (`freq`) greater than 80% of the maximum frequency (`maxfreq`) computed before. Hence, implement the following filter:

- Keep only the lines with a frequency freq greater than `0.8*maxfreq`.

1.3 Count the remaining words and save the output

Finally, perform the following operations on the selected lines (the ones selected by applying both filters):

- Count the number of selected lines and print this number on the standard output
- Save the selected words (without frequency) in an output folder for inspecting the results

2. Run the application in different ways

1. Run you application locally on Jupyter web interface by using the **PySpark (Local)** kernel and select only the lines containing the words starting with `ho` from the the HDFS file `/data/students/bigdata_internet/lab2/word_frequency.tsv` .
2. Create a Python script to be executed with the `spark-submit` command. **Note:** the `prefix` and the `output_folder` must be specified by two command line arguments (Hint: use `sys.argv[]`). Run your script and select only the lines containing the words starting with `ho` from the content of the HDFS file `/data/students/bigdata_internet/lab2/word_frequency.tsv` .

Your script must be run:

- locally by using the `--master local` option of `spark submit`
- on the cluster by using the `--master yarn` option of `spark submit`

Question: Take note of the time that it takes to run the applications in the two different submission modes. Is there a difference in time? Can you give a plausible explanation?

Question: In this application, would **caching** an RDD increase the performance? If yes, which RDD would you cache?

3. Bonus Task

On HDFS, at the path `/data/students/bigdata_internet/lab2/finefoods_text.txt` you have the original huge file that was used to produce the word frequencies that you used in step 1 and 2 of previous tasks (`/data/students/bigdata_internet/lab2/word_frequency.tsv`)

Question: How many words (with repetition) does it contain? Consider a word all the characters between spaces. Hint: you can use the standard split( ) method of strings in python3. You do not need to care about punctuation marks.

Question: Can you write the code to obtain the word frequency file starting from the original file? Compare the number of words in the file you created and the ones in the word frequency file (`/data/students/bigdata_internet/lab2/word_frequency.tsv`).

How to write the report

In your report, you must include all the code that you have written along with comments explaining what the code does. Include also the command lines that you used to execute it with `spark-submit` , as well as an example of input and output files. Include answers to all questions. You must follow the order in which question and exercise are posed.

Your report must be a PDF generated from a Jupyter notebook. Go on `File-> Export Notebook As ... -> Export Notebook As PDF` . It must be submitted through the Teaching Portal course page (didattica.polito.it), deadline is 10 days before the exam, but you can submit it now and then update the file as many times as you want.

Naming convention: The file must be named with the following schema:

```
s<id>_lab<n>.pdf
```

For example, if you student ID is 123456 and you are submitting lab 2, the file must be: `s123456_lab2.pdf` Reports with a wrong file name will **NOT** be considered.

Report length: the report length must not exceed 10 pages, otherwise it will **NOT** be considered valid.