Big Data Technologies
2024/2025
Lab 4

Context:

Suppose we have a set of text documents and we wish to determine which document is most relevant to a given query, for example “The Big Data with Spark”. A simple way to start out is by eliminating documents that do not contain all the words “The”, “Big”, “Data”, “with” and “Spark” but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document.

However, because the terms “The” and “with” are so common, this will tend to incorrectly emphasize documents which happen to use the words “The” or “with” more frequently, without giving enough weight to the more meaningful terms “Big”, “Data” and “Spark”. The terms “The” and “with” are not good keywords to distinguish relevant and non-relevant documents and terms, unlike the less used words “Big”, “Data” and “Spark”. Hence an inverse document frequency factor should be used to diminish the weight of terms that occur very frequently in the document set and to increase the weight of terms that occur rarely. This leads to the definition of the “tfidf” value where “tfidf” means “term frequency-inverse document frequency”.

Some notations are useful to compute the expected value denoted tfidf(t,d,D) in the following:

- D: the corpus, i.e., the set of all documents. It contains N documents.
- tf(t,d,D): it denotes the Term Frequency, i.e., the total number of occurrences of the term t in the document d of the corpus D.
- df(t,D): it denotes the Document Frequency, i.e., the total number of documents which contain the term t in the corpus D.
- idf(t,D) : it denotes inverse document frequency computed as idf(t,D)=log[N/ df(t,D)]
- tfidf(t,d,D) : this feature is finally defined by tfidf(t,d,D)= tf(t,d,D) x idf(t,D).

Questions:

Perform the following tasks. For each task, you must **describe carefully the type** returned by each command (and also the type of each element within the command).
You must manipulate RDD only.

1. Build the image “spark:lab4” after downloading the Dockerfile from the Moodle website
   ```
   docker build -t spark:lab4 .
   ```

2. Run a container from the image “spark:lab4”
   ```
   docker run -it spark:lab4
   ```

3. The container starts the spark-shell and you directly get the spark-shell prompt. Run the following command that will execute the list of commands of the file “commands.scala”:
   ```
   :load commands.scala
   ```
   The content of this file is described below (you can read it on the GitHub website).

4. The container contains three text files: “fruit.txt”, “color.txt” and “vegetable.txt”. You can see the content of the file “color.txt” with the command
   `"cat /opt/spark/work-dir/lab4/color.txt".!`
   This command is already written in “commands.scala”.

5. Calculate by hand tfidf(‘green’, ‘fruit.txt’,D), tfidf(‘orange’, ‘color.txt’,D) and tfidf(‘potato’, ‘vegetable.txt’,D) where D={fruit.txt, vegetable.txt,color.txt}. Detail the calculation.

6. Explain briefly the role of the following commands and precise the type of all the variables. These commands are already written in “commands.scala”.

```
// first command
val files = sc.wholeTextFiles("/opt/spark/work-dir/lab4/*.txt").collect.toList

// second command
var ch = "hello a b  c   bye"
ch.split("\\s+")
ch.split(" ")

// third command
var fruit = Array(("apple","2"), ("orange","3"), ("pear","1"))
fruit(2)._2

// fourth command
val a=Array((1,2),(3,4),(3,6))
val b=Array((3,9))
val A=sc.parallelize(a)
val B=sc.parallelize(b)
val J=A.join(B).collect
val L=A.leftOuterJoin(B).collect
```

7. Compute the number N of “.txt” documents in the directory “/opt/spark/work-dir/lab4”.

8. Compute tf(t,d,D) for all t and all d. The result must be stored in the RDD named “tf”. What is the advantage of using a “map” function when using a Big Data ecosystem like Hadoop?

9. Compute df(t,D) for all t. The result must be stored in the RDD “df”. Which commands are lazy and which commands are not lazy when calculating df(t,D)?

10. Compute idf(t,D) for all t. The result must be stored in the RDD “idf”.

11. Compute tfidf(t,d,D) for all t and all d. The result must be stored in the RDD “tfidf”.

12. Print the results on the screen.

13. Interpret the “tfidf” scores. What is the interest of these scores when we analyze a huge number of text files?

14. Save the results as text files in "/opt/spark/work-dir/lab4/tfidf.score".