Tutorial 4

Spark wordcount

Step 0: If you have not done so already: *claim a virtual machine from infomdssLabA*

Login to the azure portal (https://portal.azure.com/),

Go to **Dashboard → infomdssLabA → infomdssLabA** and claim a virtual machine.

Login to your lab B vm (the default password is the same as in tutorial 1) and **change your password**.

```
$ passwd # choose your new password wisely
```

These actions are explained in tutorial 1.

Step 1: Fetch the python script

Within your Azure VM, go to the hadoop folder, create a new directory named `wordcount_spark` and download `wordcount_mapreduce.py` from github.

```
$ mkdir ~/wordcount_spark && cd "$_" # shortcut for mkdir and cd
$ wget -O wordcount_mapreduce.py \
https://raw.githubusercontent.com/Infomdss2018/infomdss/master/wordcount_mapreduce.py
```

Step 2: Make the script executable

These commands should be familiar by now:

```
$ chmod +x wordcount_mapreduce.py
```

You also have to give spark a little help in creating a folder to store information while processing its jobs. Spark usually looks for a folder named scratch in the `.spark` directory (if this folder exists spark tries to store temp files, so spark needs writing permission). Another way of doing this is by setting a global variable.

```
$ mkdir -p /home/labuser/.spark/scratch/
```

```
$ sudo chmod 777 /home/labuser/.spark/scratch/
```

Step 3: create text file

Also not very exciting, create a file named `testfile` with some simple content:

```
$ echo "A long long long time ago,
in a galaxy far far away..." > testfile
```

Step 4: Execute the python script in Spark

Here is where you call the `Spark` command to do its magic, note that we only have to feed it one script which contains a mapper, and reducer. This approach gives us more flexibility in chaining our mapreduce actions.

```
$ spark-submit wordcount_mapreduce.py
```

Expected output:

```
2018-10-08 16:34:13 WARN  NativeCodeLoader:62 - Unable to load
native-hadoop library for your platform... using builtin-java classes
where applicable
2018-10-08 16:34:15 INFO  SparkContext:54 - Running Spark version 2.3.1
[...]
2018-10-08 16:34:22 INFO  SparkContext:54 - Successfully stopped
SparkContext
2018-10-08 16:34:22 INFO  ShutdownHookManager:54 - Shutdown hook called
2018-10-08 16:34:22 INFO  ShutdownHookManager:54 - Deleting directory
/data/home/labuser/.spark/scratch/spark-2bd7a83d-fd63-4e47-b6d6-42283a6
1b392/pyspark-321ac5ed-f1ce-4347-bb97-f7fb7425cf87
2018-10-08 16:34:22 INFO  ShutdownHookManager:54 - Deleting directory
/tmp/spark-19d9943e-c88a-4d96-859d-bd6c34cb777e
2018-10-08 16:34:22 INFO  ShutdownHookManager:54 - Deleting directory
/data/home/labuser/.spark/scratch/spark-2bd7a83d-fd63-4e47-b6d6-42283a6
1b392
```

Step 5: collect the output

Also not very existing, the `more` command is already been used in previous tutorials.

```
$ more output/part-00000
```

Expected output:

```
(u'A', 1)
(u'ago,', 1)
(u'far', 2)
(u'in', 1)
(u'long', 3)
(u'away...', 1)
(u'a', 1)
(u'time', 1)
(u'galaxy', 1)
```

Step 6: Remove your output directory

This is not a necessary step, but good practise to clean-up after your done. Note that it is wise to remove your output directory only if you do not need its content anymore.

```
$ rm -r ./output
```

Further reading and practising

You can find a lot of python-tutorials at http://learnpython.org/ and more about Spark at https://spark.apache.org/docs/latest/api/python/pyspark.html.

Stop your virtual machine at the end of the workshop