# Emmanuel Aboah Assignment 01 CSCI-E63 Big Data Analytics

Issued January 27th, 2023 Due: Saturday, February 04th, 2023, 11:59 AM EST

**Note: This is windows WSL2 setup**

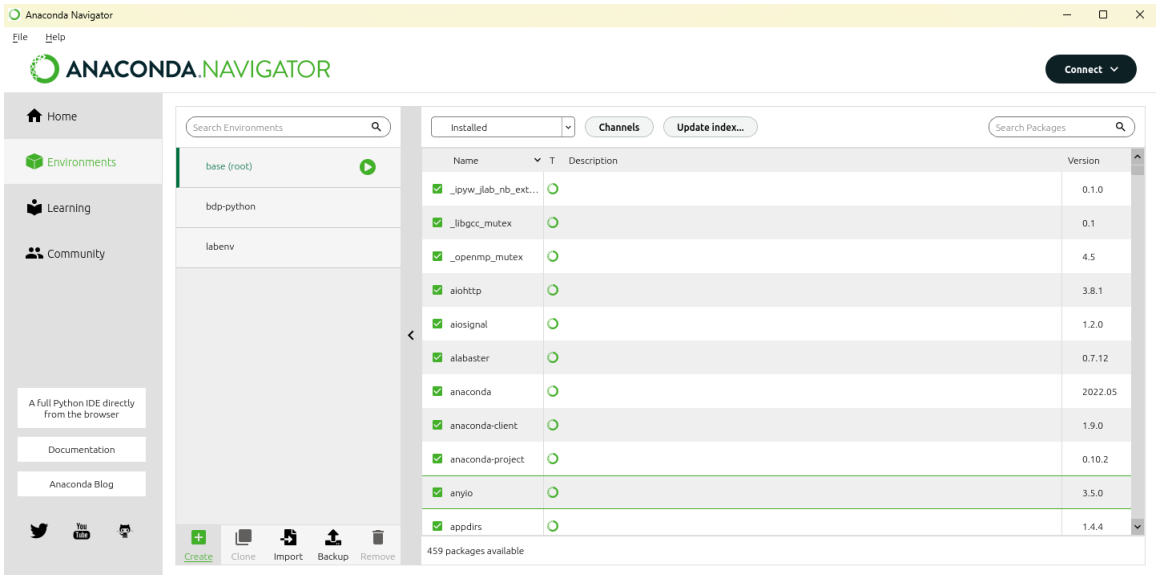
**Problem 1**.

Install the latest version of Anaconda on your system. Keep track of what folder your Anaconda will be installed in. Launch Anaconda and take a snapshot of your Anaconda Navigator and paste below.

Display the 4 Anaconda environmental variables from your install and paste below. From your terminal window display your Python version and your `conda` version and paste below. Open a Jupyter notebook and add your name to a comment (Markup cell) and run the cell. Add a Python print statement in a cell that prints something of your choice. Then run the cell to display the results. Display a snapshot of your Jupyter Notebook with your Python print statement and results and print below. Create a virtual environment for your conda called BDA2023 and show us a display of how you activated it. Display a listing of your conda virtual environments and paste below. [20%]

========== **Display your screen shots here:**

**Anaconda Navigator (Via MobaXterm):**



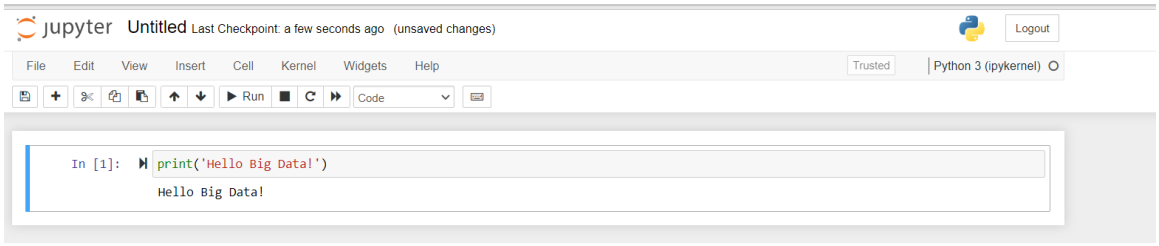

**Anaconda environment variables:**

```
# >>> conda initialize >>>
# !! Contents within this block are managed by 'conda init' !!
__conda_setup="$('/home/manny/anaconda3/bin/conda' 'shell.bash' 'hook' 2> /dev/null)"
if [ $? -eq 0 ]; then
    eval "$__conda_setup"
else
    if [ -f "/home/manny/anaconda3/etc/profile.d/conda.sh" ]; then
        . "/home/manny/anaconda3/etc/profile.d/conda.sh"
    else
        export PATH="/home/manny/anaconda3/bin:$PATH"
    fi
fi
unset __conda_setup
# <<< conda initialize <<<

# Java
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64

# Spark
export SPARK_HOME="/home/manny/dev/spark-3.3.1-bin-hadoop3"
export PATH=$PATH:$SPARK_HOME/bin

# Hadoop
export HADOOP_HOME=~/dev/hadoop/hadoop-3.3.2
export PATH=$PATH:$HADOOP_HOME/bin
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
```

**Conda and Python versions:**

```
(base) manny@LAPTOP-85L1BUVJ:~/dev$ conda -V
conda 23.1.0
(base) manny@LAPTOP-85L1BUVJ:~/dev$ python -V
Python 3.9.12
(base) manny@LAPTOP-85L1BUVJ:~/dev$
```

**Jupyter python printout:**




**conda create -n BDA2023 && conda activate BDA2023 && conda env list:**

```
(base) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$ conda create -n BDA2023 && conda activate BDA2023 && conda env list
Retrieving notices: ...working... done
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: /home/manny/anaconda3/envs/BDA2023



Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate BDA2023
#
# To deactivate an active environment, use
#
#     $ conda deactivate

# conda environments:
#
base                     /home/manny/anaconda3
BDA2023               *  /home/manny/anaconda3/envs/BDA2023
bdp-python               /home/manny/anaconda3/envs/bdp-python
labenv                   /home/manny/anaconda3/envs/labenv
spark3.3                 /home/manny/anaconda3/envs/spark3.3

(BDA2023) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$
```

**Problem 2:**

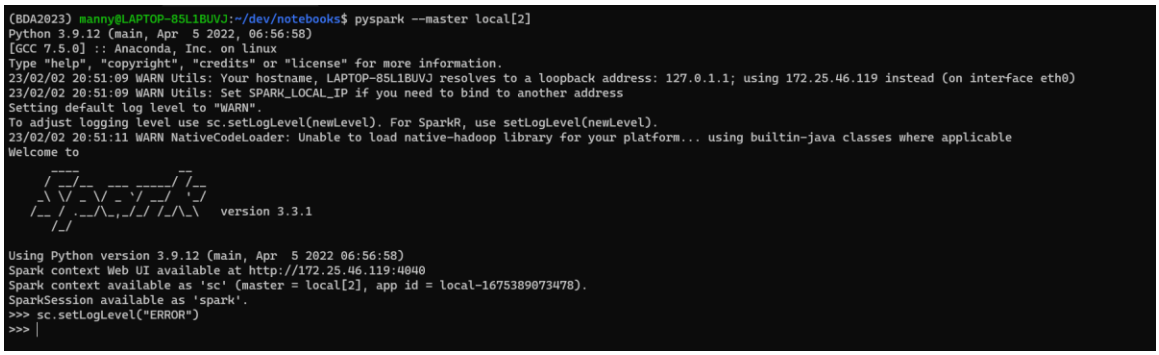
Install the latest version of Spark on your operating system. You could install either version 3.3.1 or 3.2.3. If running Anaconda Python, before the installation, make sure that you create and activate a virtual environment. Perform Python (`pip` and `pyspark`) portion of that installation in that environment. Display your Hadoop Home, JAVA Home, Spark Home and Path environment variables and paste below.

Create an env variable in your System Variables for anaconda home called ANACONDA_HOME. Use the path in problem 1 of your Anaconda install for ANACONDA_HOME. Display your ANACONDA_HOME env variable. Demonstrate that you can successfully open `pyspark` and that you can eliminate "most" of the `WARN`ing messages. You are not expected to document every step of the installation process. [20%]

**Environment variables:**

```
(base) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$ vi ~/.bashrc
(base) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$ source ~/.bashrc
(base) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$ echo $JAVA_HOME
/usr/lib/jvm/java-11-openjdk-amd64
(base) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$ echo $SPARK_HOME
/home/manny/dev/spark-3.3.1-bin-hadoop3
(base) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$ echo $HADOOP_HOME
/home/manny/dev/hadoop/hadoop-3.3.2
(base) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$ echo $ANACONDA_HOME
/home/manny/anaconda3
(base) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$
```

**Pyspark:**


(BDA2023) manny@LAPTOP-85L1BUVJ:~/dev/notebooks$ pyspark --master local[2]
Python 3.9.12 (main, Apr 5 2022, 06:56:58)
[GCC 7.5.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
23/02/02 20:51:09 WARN Utils: Your hostname, LAPTOP-85L1BUVJ resolves to a loopback address: 127.0.1.1; using 172.25.46.119 instead (on interface eth0)
23/02/02 20:51:09 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/02/02 20:51:11 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Welcome to
version 3.3.1
Using Python version 3.9.12 (main, Apr 5 2022 06:56:58)
Spark context Web UI available at http://172.25.46.119:4040
Spark context available as 'sc' (master = local[2], app id = local-1675389073478).
SparkSession available as 'spark'.
>>> sc.setLogLevel("ERROR")
>>>

========== **Display your screen shots here:**

**Problem 3:**
Display that your Spark shell works. Run the following 6 lines in your pyspark session and show us your session with your code and results. [20%]

```
myData1 = "The Project Gutenberg Etext of Ulysses, by James
Joyce".split(" ")
result1 = sc.parallelize(myData1, 2)
myData2 = "The Project Gutenberg eBook of The Time Machine, by H. G.
Wells".split(" " )
result2 = sc.parallelize(myData2,2)
result1.collect()
result2.collect()
```

========== **Display your screen shots here:**

```
>>> myData1 = "The Project Gutenberg Etext of Ulysses, by James Joyce".split(" ")
>>> result1 = sc.parallelize(myData1, 2)
>>> myData2 = "The Project Gutenberg eBook of The Time Machine, by H. G. Wells".split(" ")
>>> result2 = sc.parallelize(myData2, 2)
>>> result1.collect()
['The', 'Project', 'Gutenberg', 'Etext', 'of', 'Ulysses,', 'by', 'James', 'Joyce']
>>> result2.collect()
['The', 'Project', 'Gutenberg', 'eBook', 'of', 'The', 'Time', 'Machine,', 'by', 'H.', 'G.', 'Wells']
>>>
```

**Problem 4**.
Use RDD similar to the one used in the lecture to read file `TheTimeMachine.txt`. Count ALL words. Find the count of words: `Time Traveler, clambering` and `Morlocks`. What is the count for '`time traveler`'?

Then Create a standalone Python script that will count **ALL the words in the text and** 3 counts for "`Time Traveler`", "`clambering`" and "`Morlocks`" in file `TheTimeMachine.txt`. Execute that script using `spark-submit`.). [%20]
========== **Display your screen shots here:**

**All Words:**

```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.3.1
      /_/

Using Python version 3.6.9 (default, Nov 25 2022 14:10:45)
Spark context Web UI available at http://172.25.46.119:4040
Spark context available as 'sc' (master = local[2], app id = local-1675471508410).
SparkSession available as 'spark'.
>>> sc.setLogLevel("ERROR")
>>> tmrdd = sc.textFile("TheTimeMachine.txt")
>>> tmrdd.count()
3557
>>>
```

**Time Traveler count:**

```
>>> time_traveler_upper = tmrdd.filter(lambda x: 'Time Traveler' in x).collect()
>>> len(time_traveler_upper)
0
```

**Clambering count:**

```
>>> clambering = tmrdd.filter(lambda x: 'clambering' in x).collect()
>>> len(clambering)
3
```

**Morlocks:**

```
>>> morlocks = tmrdd.filter(lambda x: 'Morlocks' in x).collect()
>>> len(morlocks)
48
```

**‘Time traveler’:**

```
>>> time_traveler_lower = tmrdd.filter(lambda x: "time" in x and "traveler" in x).collect()
>>> len(time_traveler_lower)
0
>>> time_traveler_upper = tmrdd.filter(lambda x: 'Time Traveler' in x).collect()
>>> len(time_traveler_upper)
0
>>> time_traveler_upper = tmrdd.filter(lambda x: 'Time' in x and 'Traveler' in x).collect()
>>> len(time_traveler_upper)
0
>>> time_traveler_lower = tmrdd.filter(lambda x: "time traveler" in x).collect()
>>> len(time_traveler_lower)
0
>>>
```

**Standalone App:**

```python
import findspark

findspark.init()

from pyspark import SparkConf, SparkContext

conf = SparkConf().setMaster("local").setAppName("Hw01App")
sc = SparkContext(conf=conf)

sc.setLogLevel("ERROR")

lines_rdd = sc.textFile("TheTimeMachine.txt")

time_traveler_upper = lines_rdd.filter(lambda x: "Time Traveler" in
x).collect()
print(f"Time Traveler count: {len(time_traveler_upper)}")

time_traveler_lower = lines_rdd.filter(lambda x: "time traveler" in
x).collect()
print(f"time traveler count: {len(time_traveler_lower)}")

clambering = lines_rdd.filter(lambda x: "clambering" in x).collect()
print(f"clambering count: {len(clambering)}")

morlocks = lines_rdd.filter(lambda x: "Morlocks" in x).collect()
print(f"Morlocks: {len(morlocks)}")
```

**Output:**

```
23/02/03 21:47:47 INFO BlockManagerMaster: Registered BlockManager BlockManagerId(driver, 172.25.46.119, 43747, None)
23/02/03 21:47:47 INFO BlockManager: Initialized BlockManager: BlockManagerId(driver, 172.25.46.119, 43747, None)
Time Traveler count: 0
time traveler count: 0
clambering count: 3
Morlocks: 48
```

**Problem 5:** Open the Spark Master in the Browser and show us the screen shot.
[%20]
========== **Display your screen shots here:**


Your submission will contain this MS Word document modified with various screen shots and code snippets. Please, describe every step of your work and present all intermediate and final results in this MS Word document. Please, copy past text version of all essential command and snippets of results into this Word document with explanations of the purpose of those commands. We cannot retype text that is in JPG images. Please, always submit a separate copy of the original, working scripts you used. Sometimes we need to run your code and retyping is too costly. Please include in your MS Word document only relevant portions of the console output or output files. Sometime either console output or the result file is too long and including it into the MS Word document makes that document too hard to read. **PLEASE COPY Snippets of your Code but DO NOT EMBED files into your MS Word document**. For issues and comments visit the class Discussion Board on Piazza.