Lab 0: Creating Single Node Spark/ PySpark Cluster

Sanjana Mohile [002123793]

Northeastern University, Toronto

ALY 6110: Data Management and Big Data

Dr. Mohammad Shafiqul Islam

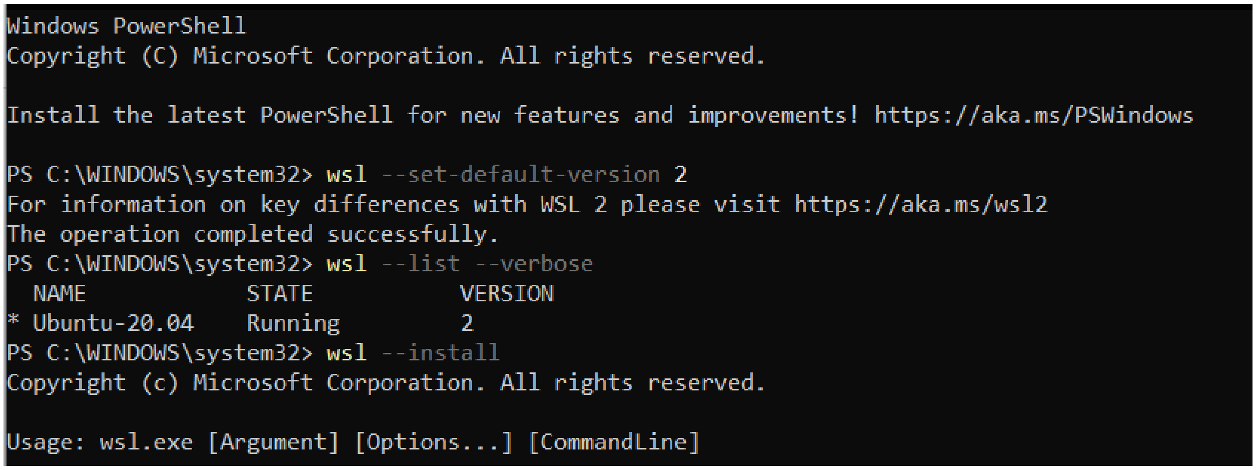
November 11th, 2022

Introduction

The main aim of Lab 0 is to help us in creating an environment, where we can gain a hands-on experience with our big data knowledge. We would first install Windows Subsystem for Linux also commonly known as WSL2 for creating a single node Spark Cluster. To create this cluster we would install Java, Python3, create Jupyter Notebook, Scala, Spark with Hadoop, and finally activate spark. We would also test Spark and close the cluster. This lab gives an overall understanding of how to create environments and would help us further in learning this course.

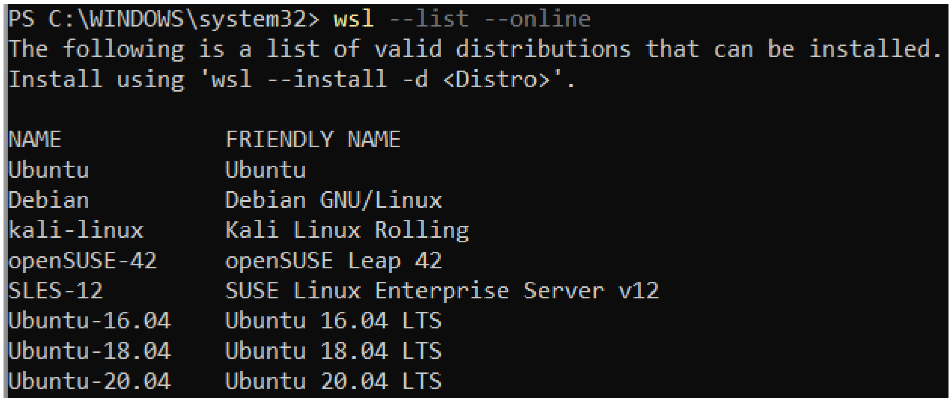
Installing WSL2

We would install the WSL2 using Windows Powershell. As we can see from figure 1 and figure 2, we were able to install the Ubuntu – 20.04 version for creating our environment.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> wsl --set-default-version 2
For information on key differences with WSL 2 please visit https://aka.ms/wsl2
The operation completed successfully.
PS C:\WINDOWS\system32> wsl --list --verbose
  NAME            STATE           VERSION
* Ubuntu-20.04    Running         2
PS C:\WINDOWS\system32> wsl --install
Copyright (c) Microsoft Corporation. All rights reserved.

Usage: wsl.exe [Argument] [Options...] [CommandLine]
```

Figure 1: Installing WSL2



```
PS C:\WINDOWS\system32> wsl --list --online
The following is a list of valid distributions that can be installed.
Install using 'wsl --install -d <Distro>'.

NAME            FRIENDLY NAME
Ubuntu          Ubuntu
Debian          Debian GNU/Linux
kali-linux      Kali Linux Rolling
openSUSE-42     openSUSE Leap 42
SLES-12         SUSE Linux Enterprise Server v12
Ubuntu-16.04    Ubuntu 16.04 LTS
Ubuntu-18.04    Ubuntu 18.04 LTS
Ubuntu-20.04    Ubuntu 20.04 LTS
```

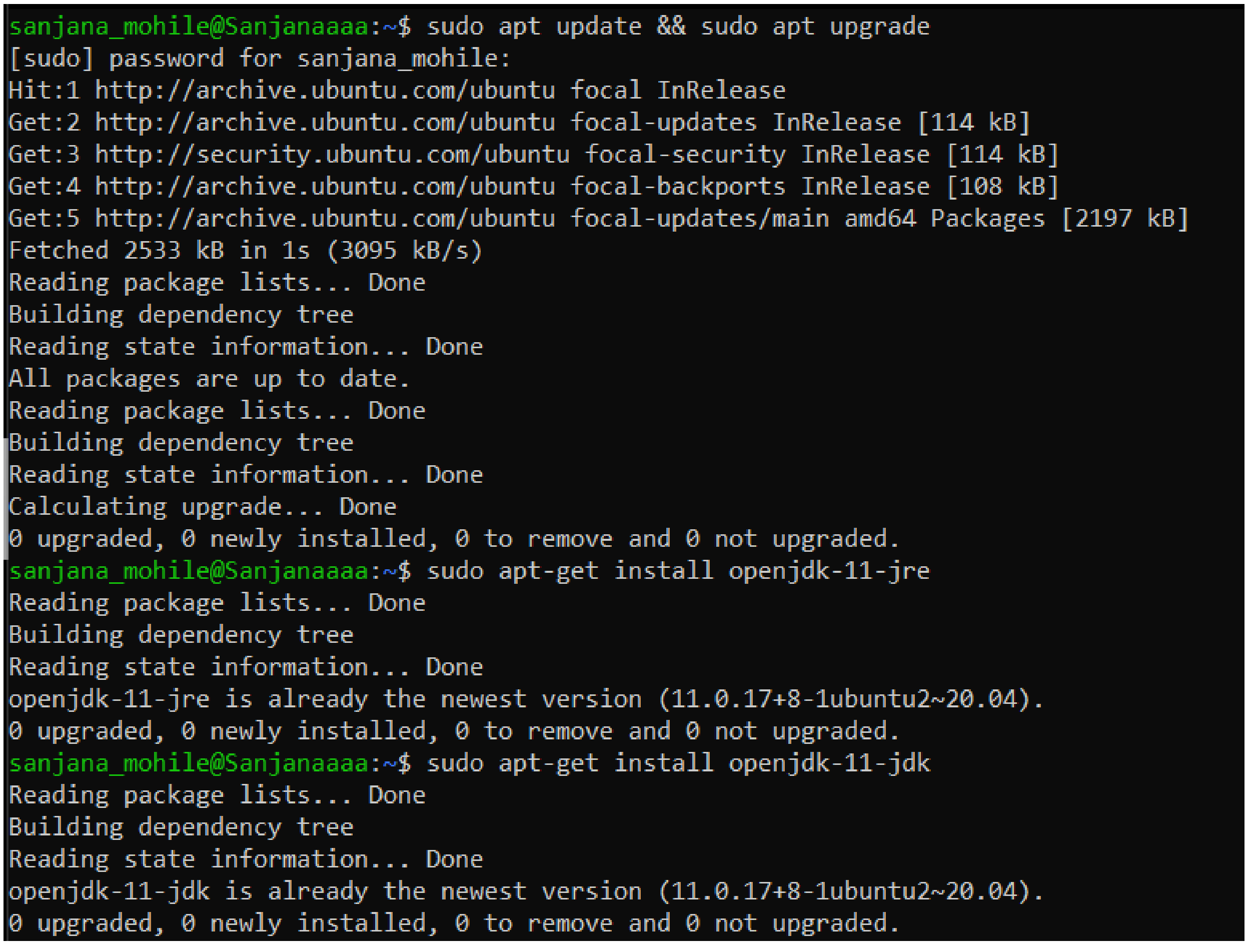
Figure 2: Successful installation of WSL2

Creating Cluster Spark

1. Install Java

   We would be installing the Java Development Kit(JDK) for creating our environment. JDK is a cross-platform software development environment that has a collection of libraries needed to develop Java-based software applications. In our case, using only JRE (Java Runtime Environment) is not a wise choice as we are not interested in running only Java programs.

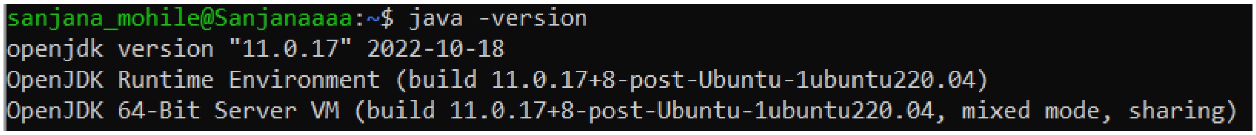
   The codes in figure 3 would be used to install Java –



```
sanjana_mohile@Sanjanaaaa:~$ sudo apt update && sudo apt upgrade
[sudo] password for sanjana_mohile:
Hit:1 http://archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:4 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [2197 kB]
Fetched 2533 kB in 1s (3095 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
sanjana_mohile@Sanjanaaaa:~$ sudo apt-get install openjdk-11-jre
Reading package lists... Done
Building dependency tree
Reading state information... Done
openjdk-11-jre is already the newest version (11.0.17+8-1ubuntu2~20.04).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
sanjana_mohile@Sanjanaaaa:~$ sudo apt-get install openjdk-11-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
openjdk-11-jdk is already the newest version (11.0.17+8-1ubuntu2~20.04).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Figure 3: Installation of Java

Now, we check the version of Java. This can be seen in figure 4.



```
sanjana_mohile@Sanjanaaaa:~$ java -version
openjdk version "11.0.17" 2022-10-18
OpenJDK Runtime Environment (build 11.0.17+8-post-Ubuntu-1ubuntu220.04)
OpenJDK 64-Bit Server VM (build 11.0.17+8-post-Ubuntu-1ubuntu220.04, mixed mode, sharing)
```

Figure 4: Version of Java

2. Installing Python

PySpark is a Spark library that is written in Python, to run Python Applications along with Apache Spark capabilities. We can run multiple nodes parallelly using PySpark.

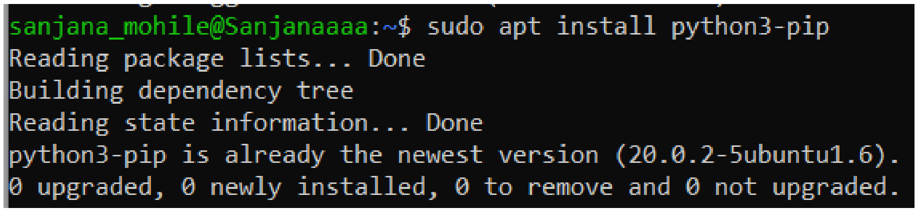
We use the following code to install python using the Ubuntu terminal –

$sudo apt update && upgrade

$sudo apt install python3 python3-pip ipython3

$sudo apt install python3-pip

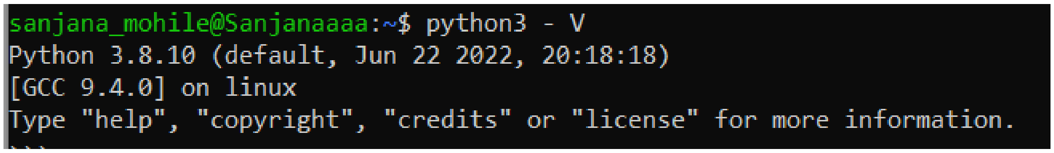
$pip3 install jupyter py4j pyspark



```
sanjana_mohile@Sanjanaaaa:~$ sudo apt install python3-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3-pip is already the newest version (20.0.2-5ubuntu1.6).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Figure 5: Python Installation

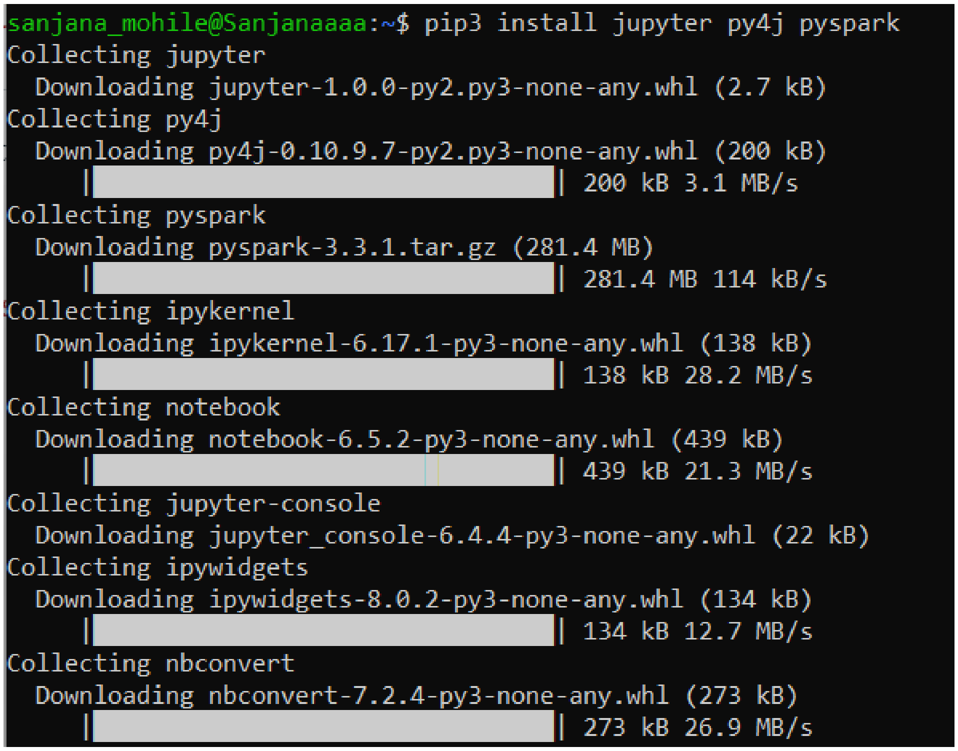
Figure 5 shows the installation of Python3. This is done in the Ubuntu terminal.



```
sanjana_mohile@Sanjanaaaa:~$ python3 - V
Python 3.8.10 (default, Jun 22 2022, 20:18:18)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
```

Figure 6: Version of Python

Figure 6 shows the version of Python installed on our device. As we can see, we installed the latest version of Python which was 3.8.10

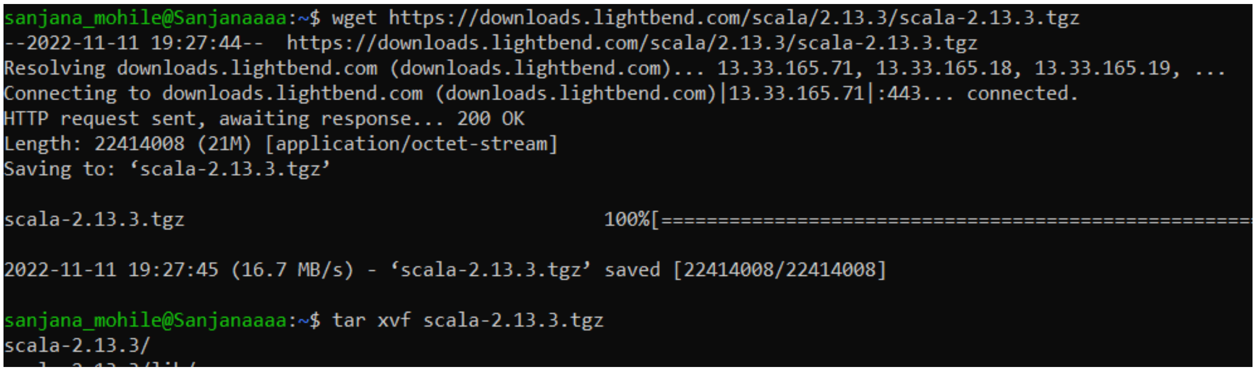


```
sanjana_mohile@Sanjanaaaa:~$ pip3 install jupyter py4j pyspark
Collecting jupyter
  Downloading jupyter-1.0.0-py2.py3-none-any.whl (2.7 kB)
Collecting py4j
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
     |████████████████████████████████| 200 kB 3.1 MB/s
Collecting pyspark
  Downloading pyspark-3.3.1.tar.gz (281.4 MB)
     |████████████████████████████████| 281.4 MB 114 kB/s
Collecting ipykernel
  Downloading ipykernel-6.17.1-py3-none-any.whl (138 kB)
     |████████████████████████████████| 138 kB 28.2 MB/s
Collecting notebook
  Downloading notebook-6.5.2-py3-none-any.whl (439 kB)
     |████████████████████████████████| 439 kB 21.3 MB/s
Collecting jupyter-console
  Downloading jupyter_console-6.4.4-py3-none-any.whl (22 kB)
Collecting ipywidgets
  Downloading ipywidgets-8.0.2-py3-none-any.whl (134 kB)
     |████████████████████████████████| 134 kB 12.7 MB/s
Collecting nbconvert
  Downloading nbconvert-7.2.4-py3-none-any.whl (273 kB)
     |████████████████████████████████| 273 kB 26.9 MB/s
```

Figure 7: Installation of Jupyter notebook

After the successful installation of python, we would now install the jupyter notebook. We use the command shown in figure 7 for its installation.

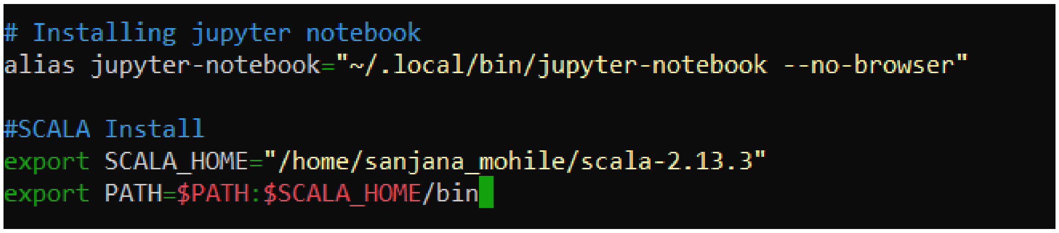
3. Installation of Scala



```
sanjana_mohile@Sanjanaaaa:~$ wget https://downloads.lightbend.com/scala/2.13.3/scala-2.13.3.tgz
--2022-11-11 19:27:44--  https://downloads.lightbend.com/scala/2.13.3/scala-2.13.3.tgz
Resolving downloads.lightbend.com (downloads.lightbend.com)... 13.33.165.71, 13.33.165.18, 13.33.165.19, ...
Connecting to downloads.lightbend.com (downloads.lightbend.com)|13.33.165.71|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 22414008 (21M) [application/octet-stream]
Saving to: ‘scala-2.13.3.tgz’

scala-2.13.3.tgz                          100%[=====================================================

2022-11-11 19:27:45 (16.7 MB/s) - ‘scala-2.13.3.tgz’ saved [22414008/22414008]

sanjana_mohile@Sanjanaaaa:~$ tar xvf scala-2.13.3.tgz
scala-2.13.3/
```

Figure 8: Installing and unzipping Scala

Figure 8 shows the command used to install Scala. Scala is an object-oriented as well as a functional programming language. Apache Spark is the most well-known open-source cluster-computing solution that is written in Scala. After the successful installation of Scala, we needed to extract the files from Scala as it was in TAR archive file (.tgz) format. This type of file is most used in Unix and Linux systems.



```
# Installing jupyter notebook
alias jupyter-notebook="~/.local/bin/jupyter-notebook --no-browser"

#SCALA Install
export SCALA_HOME="/home/sanjana_mohile/scala-2.13.3"
export PATH=$PATH:$SCALA_HOME/bin
```

Figure 9: Glimpse of bashrc file

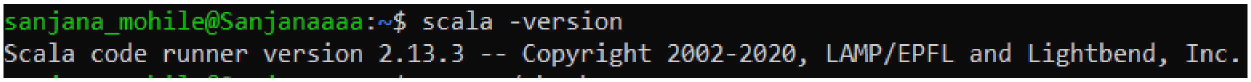
The bashrc file is a popular Linux distribution. We use "nano" to edit our bashrc file and add Scala to it using the following command –

$nano ~/.bashrc

Any changes we make in this file would automatically show when we open the file next time. Because we needed the file immediately, we use the following command -

$source ~/.bashrc

We can see figure 9, which shows how to add Scala in the bashrc file.



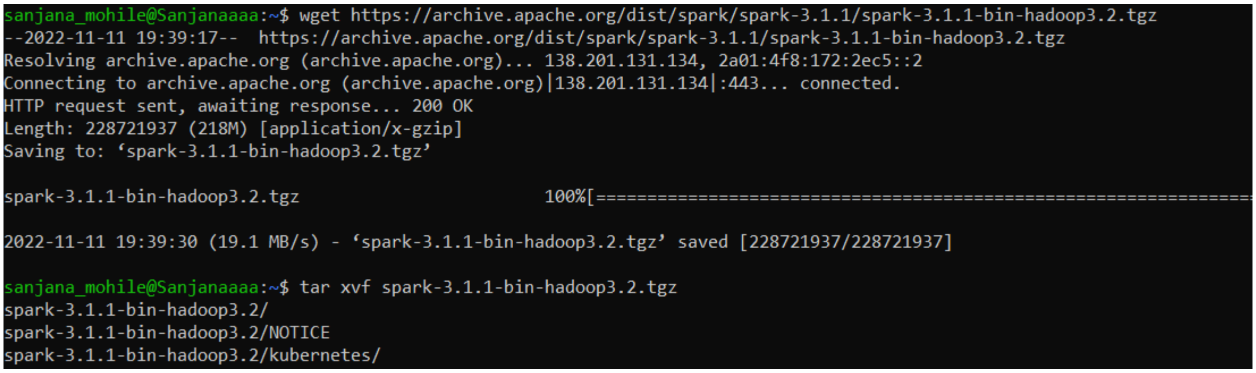
```
sanjana_mohile@Sanjanaaaa:~$ scala -version
Scala code runner version 2.13.3 -- Copyright 2002-2020, LAMP/EPFL and Lightbend, Inc.
```

Figure 10: Scala Version

Now, our Scala has been successfully installed and is ready to use. As we can see, figure 10 shows the version of Scala installed in our bashrc file.

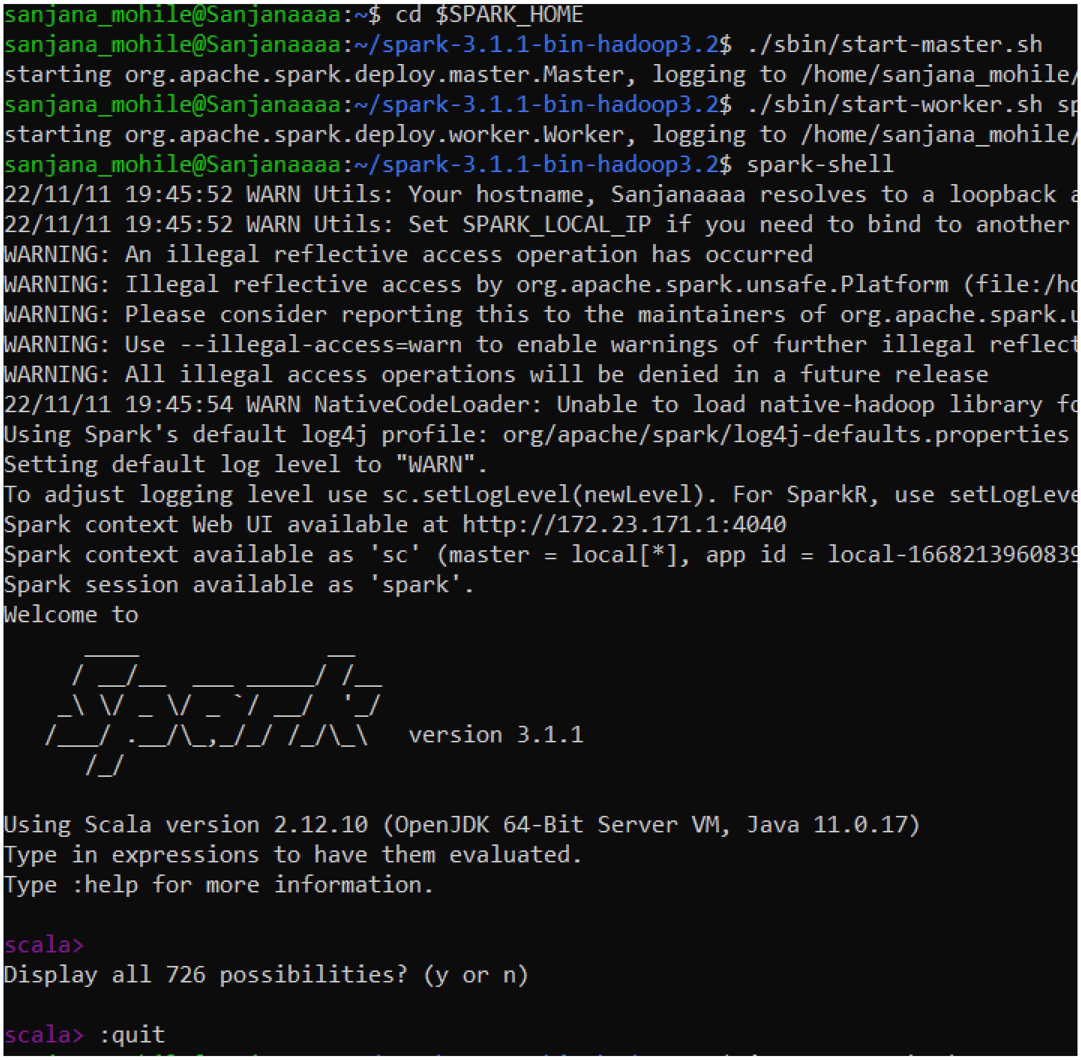
4. Install Spark with Hadoop

   We repeat the same steps we performed while installing Scala to install Spark with Hadoop. We download the TAR file of Spark with Hadoop and then unzip it. Then, we open our bashrc file again and add the same lines again to install Spark with Hadoop in the bashrc files.



```
sanjana_mohile@Sanjanaaaa:~$ wget https://archive.apache.org/dist/spark/spark-3.1.1/spark-3.1.1-bin-hadoop3.2.tgz
--2022-11-11 19:39:17--  https://archive.apache.org/dist/spark/spark-3.1.1/spark-3.1.1-bin-hadoop3.2.tgz
Resolving archive.apache.org (archive.apache.org)... 138.201.131.134, 2a01:4f8:172:2ec5::2
Connecting to archive.apache.org (archive.apache.org)|138.201.131.134|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 228721937 (218M) [application/x-gzip]
Saving to: ‘spark-3.1.1-bin-hadoop3.2.tgz’

spark-3.1.1-bin-hadoop3.2.tgz                        100%[=================================================================

2022-11-11 19:39:30 (19.1 MB/s) - ‘spark-3.1.1-bin-hadoop3.2.tgz’ saved [228721937/228721937]

sanjana_mohile@Sanjanaaaa:~$ tar xvf spark-3.1.1-bin-hadoop3.2.tgz
spark-3.1.1-bin-hadoop3.2/
spark-3.1.1-bin-hadoop3.2/NOTICE
spark-3.1.1-bin-hadoop3.2/kubernetes/
```

Figure 11: Downloading the TAR file and unzipping it

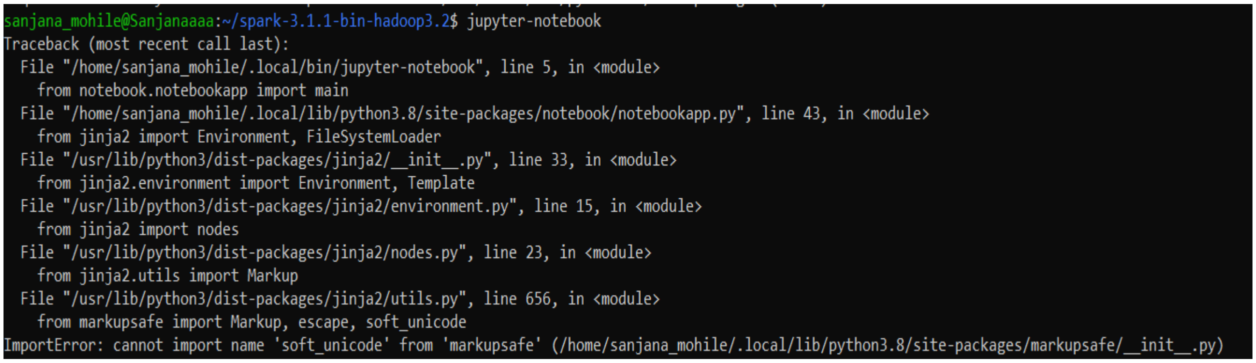


```
sanjana_mohile@Sanjanaaaa:~$ cd $SPARK_HOME
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ ./sbin/start-master.sh
starting org.apache.spark.deploy.master.Master, logging to /home/sanjana_mohile/
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ ./sbin/start-worker.sh sp
starting org.apache.spark.deploy.worker.Worker, logging to /home/sanjana_mohile/
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ spark-shell
22/11/11 19:45:52 WARN Utils: Your hostname, Sanjanaaaa resolves to a loopback a
22/11/11 19:45:52 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/ho
WARNING: Please consider reporting this to the maintainers of org.apache.spark.u
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflect
WARNING: All illegal access operations will be denied in a future release
22/11/11 19:45:54 WARN NativeCodeLoader: Unable to load native-hadoop library fo
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLeve
Spark context Web UI available at http://172.23.171.1:4040
Spark context available as 'sc' (master = local[*], app id = local-166821396083
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.1.1
      /_/

Using Scala version 2.12.10 (OpenJDK 64-Bit Server VM, Java 11.0.17)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
Display all 726 possibilities? (y or n)

scala> :quit
```

Figure 12: Successful installation of Spark with Hadoop.

5. Installation of Jupyter Notebook

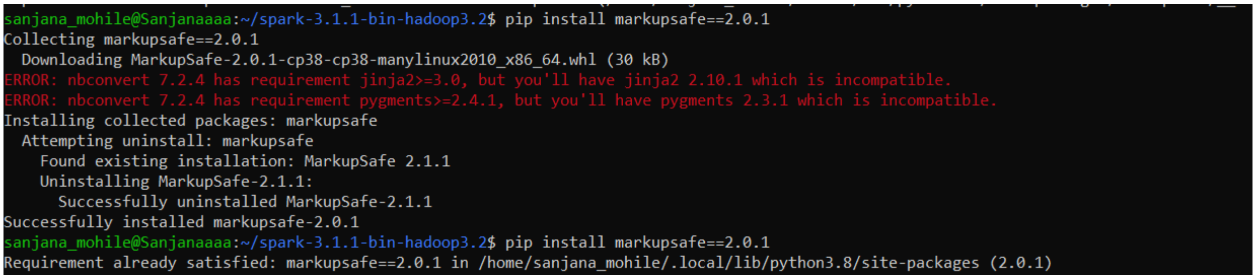
   While installing Jupyter Notebook, I encountered an error. I was able to solve this problem and move ahead for creating clusters.



```
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ jupyter-notebook
Traceback (most recent call last):
  File "/home/sanjana_mohile/.local/bin/jupyter-notebook", line 5, in <module>
    from notebook.notebookapp import main
  File "/home/sanjana_mohile/.local/lib/python3.8/site-packages/notebook/notebookapp.py", line 43, in <module>
    from jinja2 import Environment, FileSystemLoader
  File "/usr/lib/python3/dist-packages/jinja2/__init__.py", line 33, in <module>
    from jinja2.environment import Environment, Template
  File "/usr/lib/python3/dist-packages/jinja2/environment.py", line 15, in <module>
    from jinja2 import nodes
  File "/usr/lib/python3/dist-packages/jinja2/nodes.py", line 23, in <module>
    from jinja2.utils import Markup
  File "/usr/lib/python3/dist-packages/jinja2/utils.py", line 656, in <module>
    from markupsafe import Markup, escape, soft_unicode
ImportError: cannot import name 'soft_unicode' from 'markupsafe' (/home/sanjana_mohile/.local/lib/python3.8/site-packages/markupsafe/__init__.py)
```

Figure 13: Error in installation of Jupyter Notebook

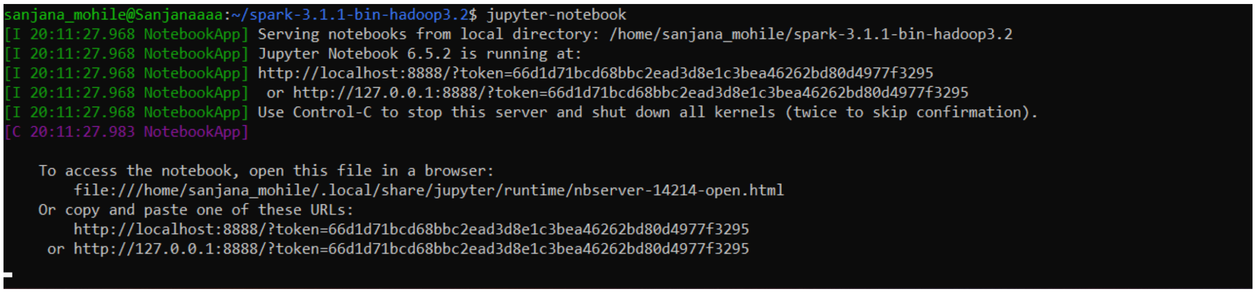
After qualitative research, I understood the error and was able to install new libraries needed to support the command and get rid of the error.



```
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ pip install markupsafe==2.0.1
Collecting markupsafe==2.0.1
  Downloading MarkupSafe-2.0.1-cp38-cp38-manylinux2010_x86_64.whl (30 kB)
ERROR: nbconvert 7.2.4 has requirement jinja2>=3.0, but you'll have jinja2 2.10.1 which is incompatible.
ERROR: nbconvert 7.2.4 has requirement pygments>=2.4.1, but you'll have pygments 2.3.1 which is incompatible.
Installing collected packages: markupsafe
  Attempting uninstall: markupsafe
    Found existing installation: MarkupSafe 2.1.1
    Uninstalling MarkupSafe-2.1.1:
      Successfully uninstalled MarkupSafe-2.1.1
Successfully installed markupsafe-2.0.1
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ pip install markupsafe==2.0.1
Requirement already satisfied: markupsafe==2.0.1 in /home/sanjana_mohile/.local/lib/python3.8/site-packages (2.0.1)
```

Figure 14: Solution of the error

As we can see in figure 14, installing “markupsafe” helped in removing the error.



```
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ jupyter-notebook
[I 20:11:27.968 NotebookApp] Serving notebooks from local directory: /home/sanjana_mohile/spark-3.1.1-bin-hadoop3.2
[I 20:11:27.968 NotebookApp] Jupyter Notebook 6.5.2 is running at:
[I 20:11:27.968 NotebookApp] http://localhost:8888/?token=66d1d71bcd68bbc2ead3d8e1c3bea46262bd80d4977f3295
[I 20:11:27.968 NotebookApp]  or http://127.0.0.1:8888/?token=66d1d71bcd68bbc2ead3d8e1c3bea46262bd80d4977f3295
[I 20:11:27.968 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 20:11:27.983 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/sanjana_mohile/.local/share/jupyter/runtime/nbserver-14214-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=66d1d71bcd68bbc2ead3d8e1c3bea46262bd80d4977f3295
     or http://127.0.0.1:8888/?token=66d1d71bcd68bbc2ead3d8e1c3bea46262bd80d4977f3295
```

Figure 15: Successful installation of jupyter notebook

Figure 15 shows the successful implementation of the jupyter notebook. Using any one of the three links mentioned in the output we can open our jupyter notebook and activate clusters thereafter.

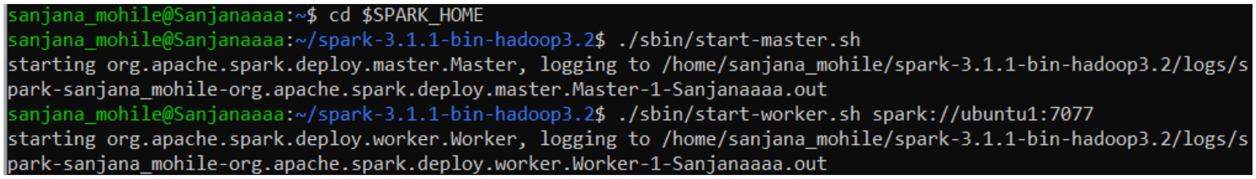
Our system is set up with all the needed libraries and tools to run Spark. We would now use the Jupyter Notebook to activate the cluster and test it.

Activate Spark

To activate the cluster, we will use the following commands-

```
$cd $SPARK_HOME
$./sbin/start-master.sh
$./sbin/start-worker.sh spark://ubuntu1:7077
```

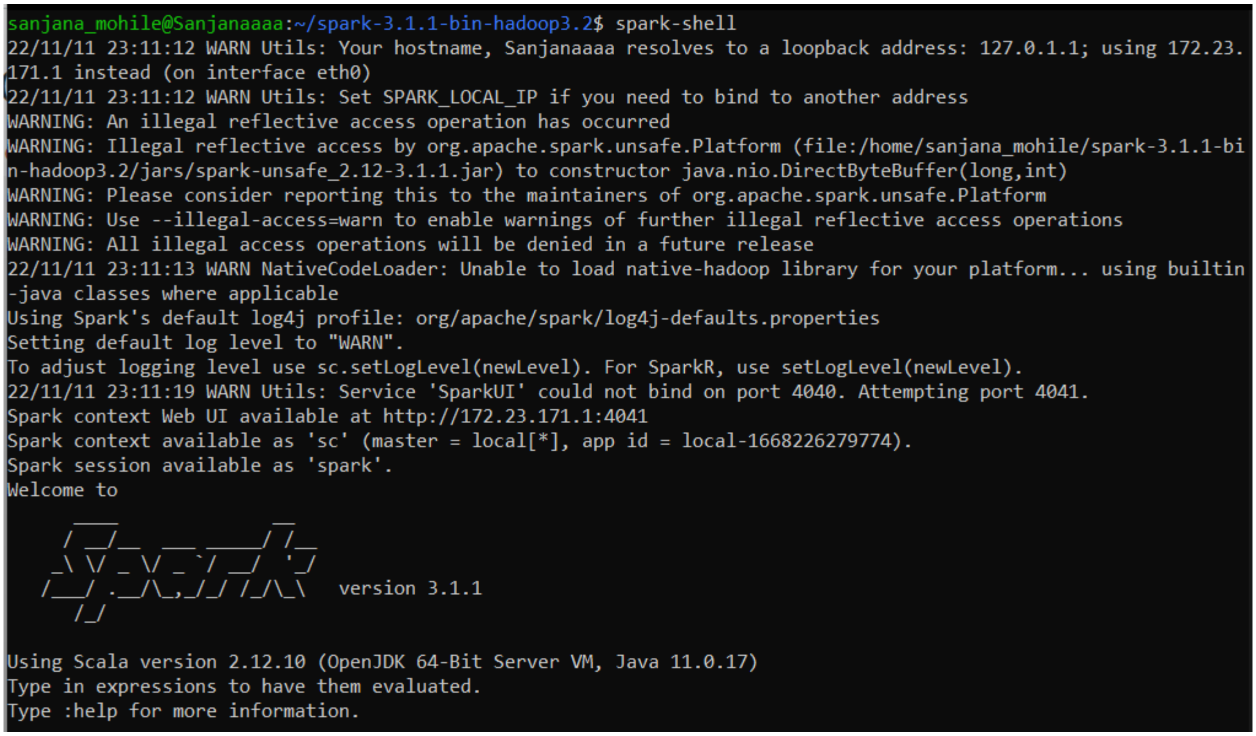
Figure 16 shows how to successfully start the clusters. We first enter Spark with Hadoop and start the mast and worker clusters. Apache spark helps us to create and stop clusters.



```
sanjana_mohile@Sanjanaaaa:~$ cd $SPARK_HOME
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ ./sbin/start-master.sh
starting org.apache.spark.deploy.master.Master, logging to /home/sanjana_mohile/spark-3.1.1-bin-hadoop3.2/logs/s
park-sanjana_mohile-org.apache.spark.deploy.master.Master-1-Sanjanaaaa.out
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ ./sbin/start-worker.sh spark://ubuntu1:7077
starting org.apache.spark.deploy.worker.Worker, logging to /home/sanjana_mohile/spark-3.1.1-bin-hadoop3.2/logs/s
park-sanjana_mohile-org.apache.spark.deploy.worker.Worker-1-Sanjanaaaa.out
```

Figure 16: Starting the cluster

Testing

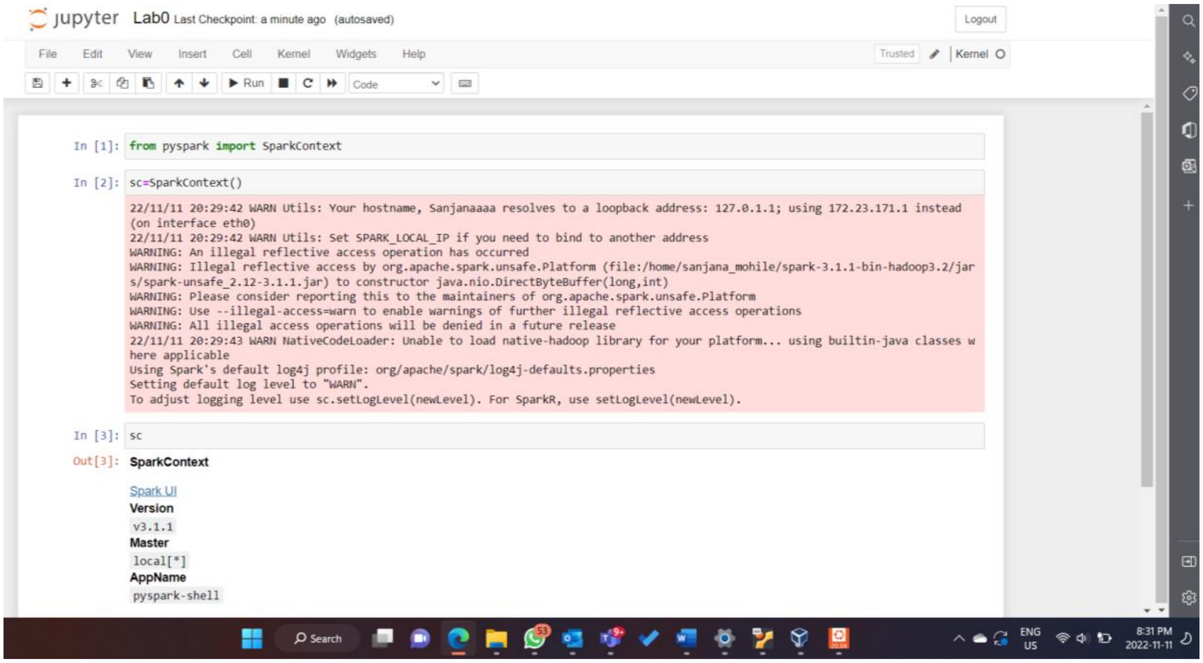


```
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ spark-shell
22/11/11 23:11:12 WARN Utils: Your hostname, Sanjanaaaa resolves to a loopback address: 127.0.1.1; using 172.23.
171.1 instead (on interface eth0)
22/11/11 23:11:12 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/home/sanjana_mohile/spark-3.1.1-bi
n-hadoop3.2/jars/spark-unsafe_2.12-3.1.1.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
22/11/11 23:11:13 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin
-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
22/11/11 23:11:19 WARN Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
Spark context Web UI available at http://172.23.171.1:4041
Spark context available as 'sc' (master = local[*], app id = local-1668226279774).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.1.1
      /_/

Using Scala version 2.12.10 (OpenJDK 64-Bit Server VM, Java 11.0.17)
Type in expressions to have them evaluated.
Type :help for more information.
```

Figure 17: Testing the Spark

The $spark-shell command is used to check if the spark is installed and to know its version. Although it is done in the earlier parts of the installation, it is a good practice to check before we start the process.

We will now test our PySpark using Jupyter Notebook. The command $jupyter-notebook will be used to test our PySpark.



```
In [1]: from pyspark import SparkContext

In [2]: sc=SparkContext()

22/11/11 20:29:42 WARN Utils: Your hostname, Sanjanaaaa resolves to a loopback address: 127.0.1.1; using 172.23.171.1 instead
(on interface eth0)
22/11/11 20:29:42 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/home/sanjana_mohile/spark-3.1.1-bin-hadoop3.2/jar
s/spark-unsafe_2.12-3.1.1.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
22/11/11 20:29:43 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes w
here applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).

In [3]: sc

Out[3]: SparkContext

        Spark UI
        Version
        v3.1.1
        Master
        local[*]
        AppName
        pyspark-shell
```

Figure 18: Jupyter Notebook

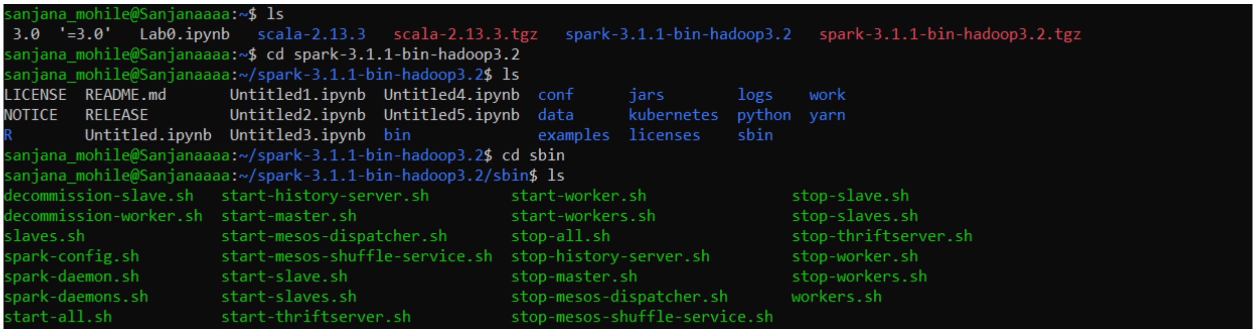
Figure 18 shows the successful installation of PySpark through Jupyter Notebook. We import the library SparkContext from PySpark first. Then we obtain the Spark UI version, Master, and AppName that would confirm the successful working of PySpark.

Stop the cluster

Our final step would be to stop the cluster after testing it. We can not leave the clustered unattended. Once we complete the testing of our cluster, we stop it to avoid heavy billings. We would use the following command to stop the cluster –

$./sbin/stop-master.sh

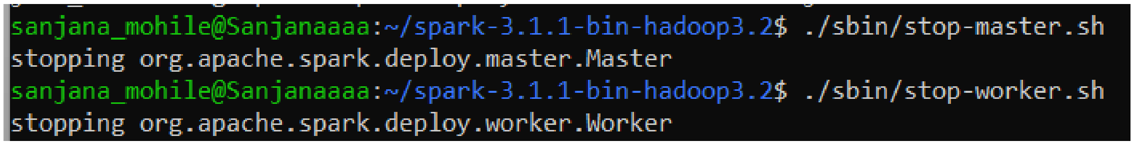
$./sbin/stop-worker.sh



```
sanjana_mohile@Sanjanaaaa:~$ ls
 3.0  '=3.0'   Lab0.ipynb   scala-2.13.3   scala-2.13.3.tgz   spark-3.1.1-bin-hadoop3.2   spark-3.1.1-bin-hadoop3.2.tgz
sanjana_mohile@Sanjanaaaa:~$ cd spark-3.1.1-bin-hadoop3.2
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ ls
LICENSE  README.md       Untitled1.ipynb  Untitled4.ipynb  conf      jars        logs    work
NOTICE   RELEASE         Untitled2.ipynb  Untitled5.ipynb  data      kubernetes  python  yarn
R        Untitled.ipynb  Untitled3.ipynb  bin              examples  licenses    sbin
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ cd sbin
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2/sbin$ ls
decommission-slave.sh   start-history-server.sh         start-worker.sh               stop-slave.sh
decommission-worker.sh  start-master.sh                 start-workers.sh              stop-slaves.sh
slaves.sh               start-mesos-dispatcher.sh       stop-all.sh                   stop-thriftserver.sh
spark-config.sh         start-mesos-shuffle-service.sh  stop-history-server.sh        stop-worker.sh
spark-daemon.sh         start-slave.sh                  stop-master.sh                stop-workers.sh
spark-daemons.sh        start-slaves.sh                 stop-mesos-dispatcher.sh      workers.sh
start-all.sh            start-thriftserver.sh           stop-mesos-shuffle-service.sh
```

Figure 19: Checking the files

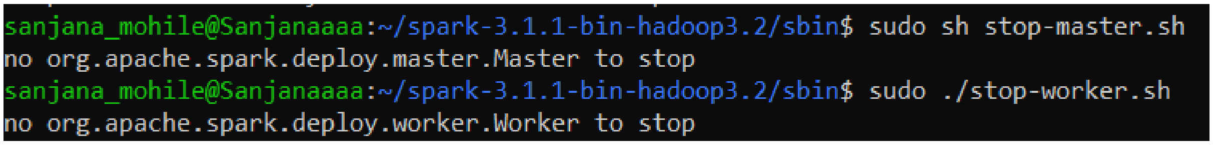
Before we stop the clusters, it is important to know which and how many clusters are open. The "ls" command shows the list of files in that folder.



```
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ ./sbin/stop-master.sh
stopping org.apache.spark.deploy.master.Master
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2$ ./sbin/stop-worker.sh
stopping org.apache.spark.deploy.worker.Worker
```

Figure 20: Stopping the cluster

Figure 20 shows the attempt of stopping our cluster. To be sure of it, we double-run the command of stopping the cluster.



```
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2/sbin$ sudo sh stop-master.sh
no org.apache.spark.deploy.master.Master to stop
sanjana_mohile@Sanjanaaaa:~/spark-3.1.1-bin-hadoop3.2/sbin$ sudo ./stop-worker.sh
no org.apache.spark.deploy.worker.Worker to stop
```

Figure 21: Checking if the clusters have stopped

We now check if the clusters are still running or are stopped by using the commands shown in figure 21. "Sudo" means "superuser do". Hence, we are now sure that our clusters have successfully stopped.

Conclusion

We have now successfully created an environment that will help us in practicing the various concepts of Big Data Analytics. The installation of different tools and libraries was interesting to learn and understand. We also were able to create, test, and close the clusters. We also learned about the basic steps of installing WSL2 in our systems.

References

I. Banerjee, P. (June 17, 2022) JDK in Java. Geeks for geeks. Retrieved November 11, 2022, from https://www.geeksforgeeks.org/jdk-in-java

II. Fox, A. (January 23, 2018) What is Bashrc and why should you edit it? Make Tech Easier. Retrieved November 11, 2022, from https://www.maketecheasier.com/what-is-bashrc

III. Pedamkar, P. (September 22, 2021) Spark Shell Commands. Educba. Retrieved November 11, 2022, from https://www.educba.com/spark-shell-commands

IV. My Repository - https://github.com/SanjanaMohile/ALY-6110---Data-Management-and-Big-Data