Setup PySpark environment

Jia Yu (jiayuasu.github.io)

Washington State University

Last updated: Sept 7 2021

1. Install a Linux Operating System

In this class, if you will install a new Linux OS or use WSL, you are required to use Ubuntu 20.04 with an OS username "ubuntu". No capitalization.

If you already have a computer with MacOS, Ubuntu or other Unix-like OS, you can jump to Section 1.2.

If you are using a Windows operating system, you have two options:

1. (Not recommended) Use VirtualBox to create a Virtual Machine with Ubuntu Desktop 20.04 installed.
    - After trying all solutions, if you still cannot make Spark work on your Windows or Mac, just don't bother to do it anymore.
    - Use VirtualBox to install a brand-new Ubuntu Desktop 20.04 VirtualMachine and install the necessary software, this will fix everything. When you create your VM, make sure you give AT LEAST 4GB RAM and 30GB hard drive. Here is a youtube video about how to install it: https://www.youtube.com/watch?v=x5MhydijWmc
2. (Recommended) Use Windows Subsystem for Linux (WSL) available on Windows 10 and newer.

1.1 Install WSL on Windows 10 and newer

Follow "Manual Installation Steps" Step 1 and Step 6 on Windows official WSL doc to install WSL 1 with Ubuntu 20.04: https://docs.microsoft.com/en-us/windows/wsl/install-win10#manual-installation-steps

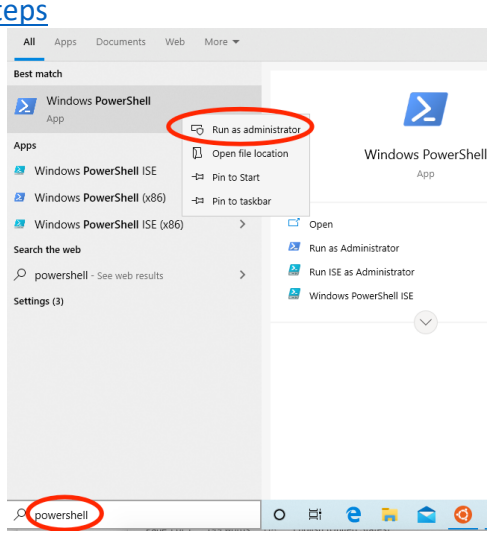

Figure 1 Start Windows Powershell with Admin role

Note:

- You need to start Powershell with Admin privilege.

- You need to restart your Windows after Step 1, before Step 6.

After Step 6, you will be able to launch Ubuntu from your Windows Start menu.

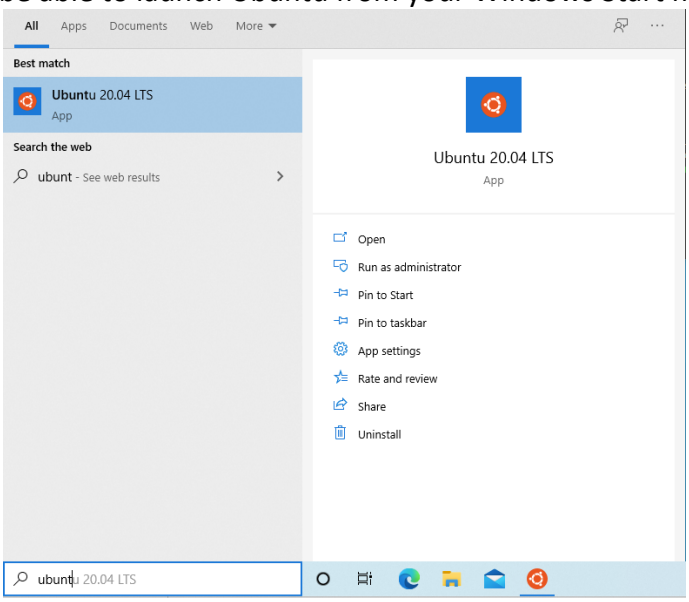

Figure 2 Start Ubuntu from Windows start menu

1.2 Install necessary software on Ubuntu 20.04

Your Ubuntu probably already has these software installed. If so, don't need to install them again. The installation steps must be done in an Ubuntu terminal.

On MacOS, you already have SSH / Java / ifconfig installed. For other software, you will need to install Homebrew (https://brew.sh/) and use "brew" instead of "apt" command to install these software. Note that: brew does not need "sudo" for installation.

1.2.1 Update software listing

This will update the software listing information on your local machine. This is required because many software information may be out-of-date and your installation might fail.

sudo apt update

1.2.2 ssh server

- On WSL ubuntu, you must remove and re-install openssh-server

sudo apt remove openssh-server
sudo apt install openssh-server

- On regular ubuntu:

sudo apt install openssh-server

- Check the status of the ssh service:

service ssh status

- If you see: * sshd is not running, then run this command:

sudo service ssh start

On WSL, Windows will prompt a dialog window to ask you to give permission to ssh the first time when you run this command. Please check all checkbox and allow it. **You need to run this command everytime when you restart your Windows.**

If by any chance, you missed that dialog window, you can do "Enable Port 22 in Windows Firewall" as in this blog post: https://www.illuminiastudios.com/dev-diaries/ssh-on-windows-subsystem-for-linux/

- On a Mac

ssh is already installed by default. But you need to enable ssh first.

In a terminal, you need to run "sudo systemsetup -setremotelogin on". If it shows "Turning Remote Login on or off requires Full Disk Access privileges", go to Mac "System Prefs>Security & Privacy>Privacy tab>Full Disk Access" and give Full Disk Access to "Terminal".

1.2.3 ifconfig

sudo apt install net-tools

Confirm the installation in the terminal: ifconfig
You should be able to see your IP address

1.2.4 Java Runtime Environment (JRE)

sudo apt install openjdk-11-jre-headless

Confirm the installation in the terminal: java -version
As long as the Java version is >= 1.8 (Java 8) and < 1.16 (Java 16). You should be fine. Mac already has Java there.

If you are using the latest MacOS Big Sur which comes with Java 16 by default, you won't be able to run Spark. You will have to downgrade it to Java 11. But just don't bother. Use the VirtualBox option mentioned before.

2. Install Spark environment in a real computer cluster

2.1 Key points

- A Spark cluster consists of a master machine and a number of worker machines.
- The master coordinates the tasks across different workers.
- The master talks to each worker via no-password ssh.
- Each worker talks to the master via no-password ssh.
- The OS username, Spark installation path and input/output data paths must be exactly same on all machines including master and workers.

2.2 Setup no password ssh on all machines including master and workers

- Generate public key and private key for this machine

ssh-keygen

Then keep pressing “enter” until the command stops. You will find “id_rsa” and “id_rsa.pub” in ~/.ssh folder.

- Share the content of public key “id_rsa.pub” of the master machine with each worker machine. Append the content to ~/.ssh/authorized_keys file on each worker machine.
- Share the content of public key “id_rsa.pub” of each worker machine with the master machine. Append the content to ~/.ssh/authorized_keys file on the master machine.
- If you want to use the master as a worker machine, do the same to itself.
    - cat id_rsa.pub > authorized_keys
- Verify no-password ssh is enabled by
    - ssh from master to each worker
    - ssh from each worker to the master
    - ssh localhost on the master machine if you want to use the master as a worker

You should be able to log into the remote computer without providing a password

2.3 Download Spark on all machines including master and workers

- Go to https://spark.apache.org/downloads.html
- Select the Spark version. In this tutorial, use Spark 3.0.3
- Package type: Pre-built for Apache Hadoop 3.2 and later
- Download spark-3.0.3-bin-hadoop3.2.tgz
  wget https://mirrors.advancedhosters.com/apache/spark/spark-3.0.3/spark-3.0.3-bin-hadoop3.2.tgz
- Unzip the tgz file
  tar -zxvf spark-3.0.3-bin-hadoop3.2.tgz

2.4 Start your Spark cluster now

- On the master machine, go to spark-3.0.3-bin-hadoop3.2 folder. Let's assume its path is /home/ubuntu/spark-3.0.3-bin-hadoop3.2

- In the conf folder, copy conf/slaves.template to conf/slaves file

cp conf/slaves.template conf/slaves

- Add the ip addresses of all workers to conf/slaves file. If you want to use the master itself as a worker, just use localhost. By default, conf/slave has localhost inside

  If you use WSL Ubuntu instead of a real Unix-like system such as MacOS or Ubuntu, you won't be able to connect to other workers. Just use the master itself as the worker.

- Start the Spark cluster
  ./sbin/start-all.sh
- Confirm your Spark cluster is ready. On the master machine, open the browser and go to localhost:8080. You should be able to see a web page which lists all registered workers.

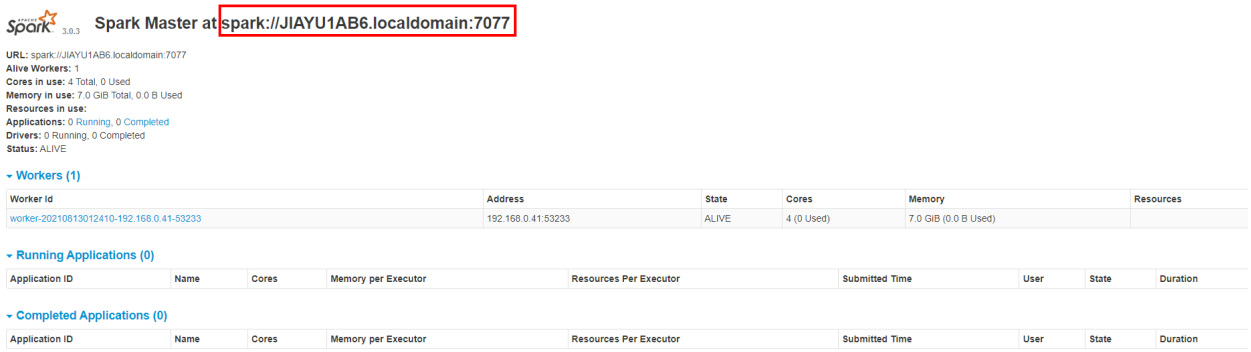

Figure 3 Spark Web UI at localhost:8080

3. Setup Python Jupyter notebook environment

The operations below should be done on the master machine of the cluster.

3.1 Install Python3 and Pip3

Python3 is installed by Ubuntu by default. Confirm the installation in the terminal: python3

Install Pip3: sudo apt install python3-pip

3.2 Install PySpark, Jupyter notebook and Pandas using pip

pip3 install pyspark==3.0.3

pip3 install notebook

pip3 install pandas

3.3 Add the notebook commands to PATH

On WSL, add /home/ubuntu/.local/bin to PATH environment variable

echo "PATH=\$PATH:/home/ubuntu/.local/bin" >> ~/.bashrc

source ~/.bashrc

3.3 Setup environment variables

export SPARK_HOME=/home/ubuntu/spark-3.0.3-bin-hadoop3.2
export PYTHONPATH=/home/ubuntu/spark-3.0.3-bin-hadoop3.2/PYTHON

These environment variables only work for the current terminal session. This means: if you start your notebook from a different Linux terminal session, you need to re-enter these commands. This usually happen if you open another terminal or restart your machine.

Here is a method to set these variables permanently but I don't recommend it:
https://askubuntu.com/questions/643585/how-to-set-a-permanent-environment-variable

3.4 Start your first Spark Python Jupyter Notebook

1. Start your notebook: jupyter-notebook --no-browser
2. Keep this terminal open. Make sure you copy the token URL shown in the terminal. For example:
   http://localhost:8888/?token=ecd392ec6a2f5a16805aad95c84f8d76f60b88eb2ab5af1c
3. Then in your Windows browser, go to the token URL you just copied:
   http://localhost:8888/?token=ecd392ec6a2f5a16805aad95c84f8d76f60b88eb2ab5af1c
4. Then try the notebook example here: https://github.com/DataOceanLab/CPTS-415-Project-Examples/blob/main/pyspark-example.ipynb

5. Change the Master IP in the notebook to switch between the pseudo mode and cluster mode.

6. When use the cluster mode, you should be able to see your notebook application on master IP address (or localhost):8080

3.5 How to access the Ubuntu files from your Windows and vice versa

Make sure your Windows can show hidden files: https://support.microsoft.com/en-us/windows/view-hidden-files-and-folders-in-windows-97fbc472-c603-9d90-91d0-1166d1d9f4b5

- Access Ubuntu files from Windows:

C:\Users\<username>\AppData\Local\Packages\CanonicalGroupLimited.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\home\ubuntu

- Access Windows files from Ubuntu:

cd /mnt/c/

CAUTION

Creating/modifying any files within the Linux subsystem using Windows apps & tools can cause data corruption and data loss in Ubuntu subsystem. This is absolutely not supported.

You should only copy/paste files if necessary but never create or modify an Ubuntu file from Windows and vice versa!