CINF/CSCI 5931 Big Data Analytics

Lab Tutorial—Install Spark and Python on AWS EC2

Objective: Sep up Python, Spark, and Jupyter Notebook on AWS EC2.

Note: When you sign up AWS, you have 12-month access to services in the "Free tier" including micro instances. So you will NOT be charged. However, it is your responsibility to monitor your bills to make sure you will not go over your credit.

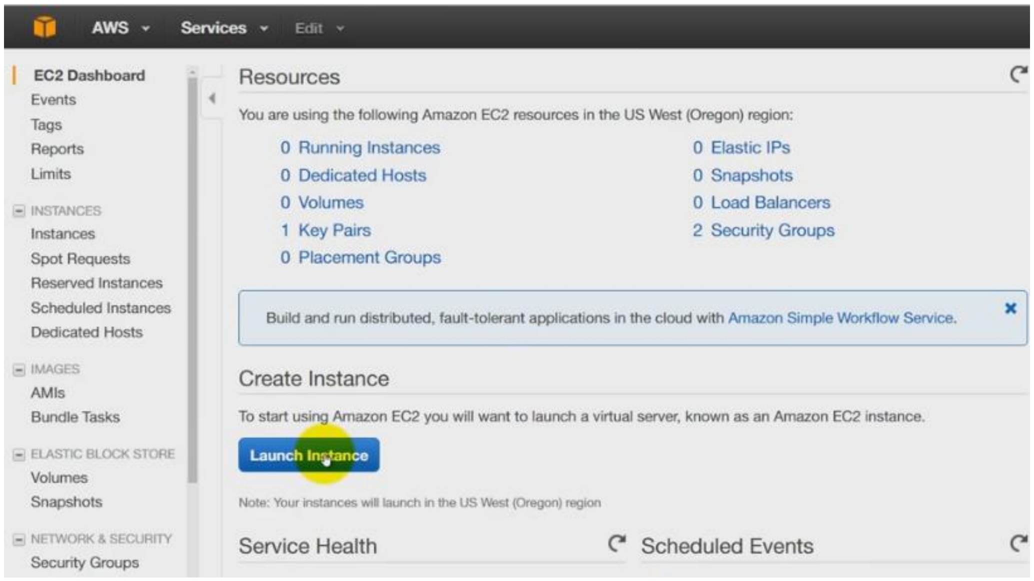
Because not all of you have easy access to a computer with the desired level of specification (such as at least 8GB memory) to set up a local system. We will employ AWS to have access to a virtual computer that you can access through the Internet.

Step 1. Set up AWS account

By now, you should have completed this step for Assignment 1.

Step 2. Create an EC2 instance

1. Go to aws.amazon.com and log into your console.
2. From the list of services, click on 'EC2' and you will see something as below.



You are using the following Amazon EC2 resources in the US West (Oregon) region:

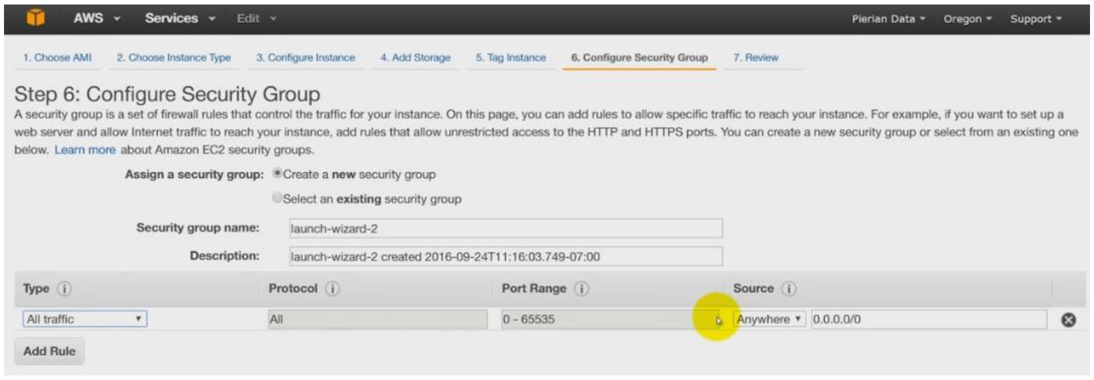
Build and run distributed, fault-tolerant applications in the cloud with Amazon Simple Workflow Service.

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Note: Your instances will launch in the US West (Oregon) region

3. Click on 'Launch Instance', you will be guided through the process.
   a. For Step 1 'Choose an Amazon Machine Image (AMI)', select the 'Ubuntu Server' option, make sure you see the 'Free tier eligible' underneath the Ubuntu icon.
   b. For Step 2 'Choose an Instance Type', select 'General purpose' (Family) and t2.micro (Type), notice that there is a 'Free tier eligible' label for t2.micro. (Note: you may try other more powerful instances later with charge for the "real thing").
   c. For Step 3 'Configure Instance Details', keep all the parameters as they are to avoid charges.
   d. For Step 4 'Add Storage', keep the default.
   e. For Step 5 'Tag Instance', create a meaningful name for your instance for future references.

f. For Step 6 ‘Configure Security Group’, click ‘Create a new security group’ for ‘Assign a security group’, the choose ‘All traffic’ for ‘Type’ and keep the others as default. This is the easiest way to enable inbound SSH traffic from your IP address to your instance on EC2. (But this basically leaves your instance open to the world).



Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group: (●) Create a **new** security group
( ) Select an **existing** security group

Security group name: launch-wizard-2
Description: launch-wizard-2 created 2016-09-24T11:16:03.749-07:00

Type	Protocol	Port Range	Source
All traffic	All	0 - 65535	Anywhere 0.0.0.0/0

Note that you will get a warning message about the insecurity of your instance. We are doing this for easy access to non-critical instance to practice your skills. For production environment, of course you are going to assign much more strict security policies to safeguard data and information on your instances.

g. Launch the instance and you will see a pop-up window ‘Select an existing key pair or create a new key pair’. We have gone through this process for Assignment 1. Make sure you download the key pair somewhere you will remember because you will have to use it to access your virtual machine remotely through SSH or something similar.

h. Go to your console to make sure your instance is correctly launched and up running. Like in Assignment 1, pay attention to the Public DNS of your instance. (Make it a habit terminate your instance once you are done with your task).

Step 3. Connect to Command Line of your EC2 Instance

Go to http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html, following the instructions to connect your Linus Instance on EC2 (You should have done this for Assignment 1). The major steps are:

1. Download PuTTY and PuTTYgen.
2. Use PuTTYgen to convert your key pair in .pem format into a .ppk file.
3. Start a PuTTY session, your Host Name is ‘ubuntu@<your_public_DNS_name>, Port is ‘22’ and Connection type is ‘SSH’. Then in the left pane, click Category->Connection->SSH->Auth, for the textbox, browse to where your .ppk file is and click ‘Open’ to start a PuTTY session.
4. You should see your virtual machine up running now. You can play with the command line to see what’s there.

If you are a Mac or Linux user, this step is going to be different. You basically do the same thing through your terminal as follows:

1. Change directory to where your key pair file is.
2. Do ‘`chmod 400 <your key pair file name>`’.
3. Do ‘`ssh -i <your key pair file name> ubuntu@<your public DNS name>`’.

4. You should be able to connect to your instance.

Step 4. Use the Command Line to Set Up the Environment on Your EC2 Instance

In this step, you are going to download and install all the software you need on your EC2 instance (that is your virtual machine on AWS).

1. Open your ubuntu command line, issue the following commands (step by step):

```
sudo apt-get update
```

(this updates the package lists from a server on the Internet. The list can be used to determine which software to install when given a command to install)

```
sudo apt install python3-pip
```

(this installs pip3, we will use pip3 to install other python packages)

```
pip3 install jupyter
```

(this installs jupyter. For more information on jupyter, please visit http://jupyter.org/. In a nutshell, it is a web application that allows you to create and share documents that contain live code, equations, visualizations and comments. It supports many programming languages including python).

```
sudo apt-get install default-jre
```

(this installs java because we need scala)

```
sudo apt-get install scala
```

(this installs scala since we need to install Spark)

```
pip3 install py4j
```

(this installs py4j, Py4J enables Python programs running in a Python interpreter to dynamically access Java objects in a Java Virtual Machine).

```
wget http://archive.apache.org/dist/spark/spark-2.1.1/spark-2.1.1-bin-hadoop2.7.tgz
```

(this downloads the Spark library from Apache, the most updated version is Spark 2.2.0)

```
sudo tar -zxvf spark-2.1.1-bin-hadoop2.7.tgz
```

(this unzips and installs the Spark and Hadoop)

Note: please use 'pwd' to locate the working directory for your Spark.

2. Other tasks:

As we know, Apache Spark supports several languages and we are suing Python for this course. It is true that Scala is the language that Spark comes in, but we choose Python because its wide application in and power of handling complex data analytics. PySpark is an API developed in Python for spark programming and writing spark applications in Python style. However, PySpark is not on sys.path by default, so we need to add PySpark to sys.path at runtime, and we use 'findspark' to do it by issuing the following command.

```
pip3 install findspark
```

Then go to your home directory and issuing the following commands:

```
jupyter notebook –generate-config
```

And then issue the following commands (step by step) to create some certification files for the configuration of jupyter notebook.

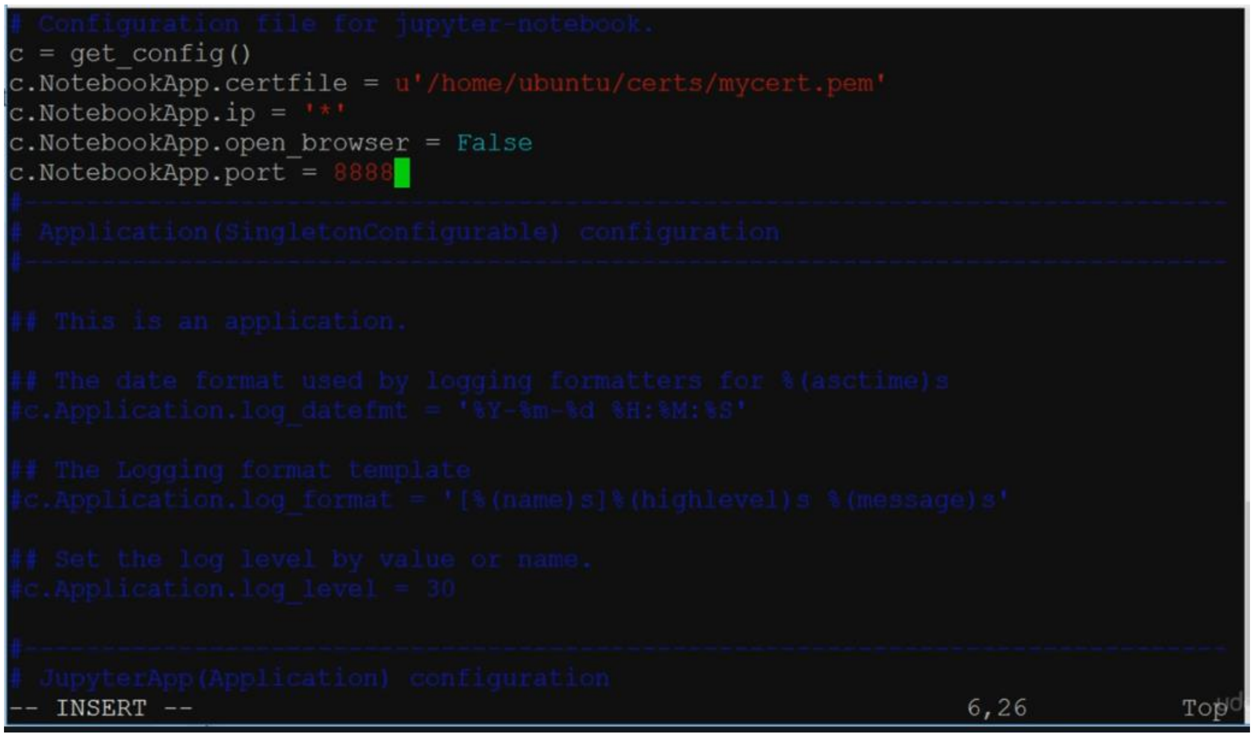
`mkdir certs` (make a new directory called certs)
`cd certs` (change directory to certs and then issue the following command)
```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout mycert.pem -out mycert.pem
```

Follow the prompts and go through the questions and return to the 'certs' directory and then editing the configuration file:

`cd ~/.jupyter/` (go into the hidden directory)

`vi jupyter_notebook_config.py` (start a text editor to edit the configuration file for jupyter in Python)

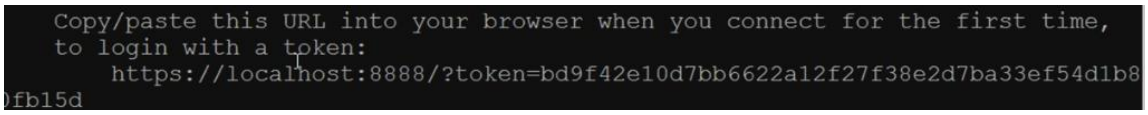
Then click 'i' to get into the insert mode and use the text editor to insert the following codes to the file



```
# Configuration file for jupyter-notebook.
c = get_config()
c.NotebookApp.certfile = u'/home/ubuntu/certs/mycert.pem'
c.NotebookApp.ip = '*'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 8888
#------------------------------------------------------------------------------
# Application(SingletonConfigurable) configuration
#------------------------------------------------------------------------------

## This is an application.

## The date format used by logging formatters for %(asctime)s
#c.Application.log_datefmt = '%Y-%m-%d %H:%M:%S'

## The Logging format template
#c.Application.log_format = '[%(name)s]%(highlevel)s %(message)s'

## Set the log level by value or name.
#c.Application.log_level = 30

#------------------------------------------------------------------------------
# JupyterApp(Application) configuration
-- INSERT --                                              6,26          Top
```

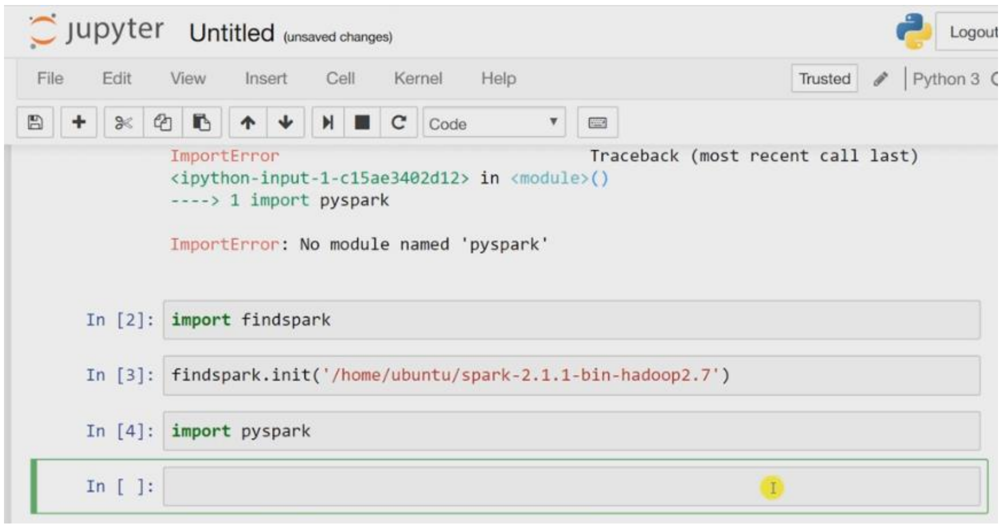
After the insertion, use ':wq!' to exit.

3. Go back to home directory and issue '`jupyter notebook`'. And you should see a message similar to this:



```
    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        https://localhost:8888/?token=bd9f42e10d7bb6622a12f27f38e2d7ba33ef54d1b8
fb15d
```

   Copy and paste this link into the address bar of a browser, replace 'localhost' with your actual EC2 Public DNS name (skip the security warning), and then you should see your jupyter notebook systems.
4. Using jupyter, start a new notebook, and import findspark and pyspark as follow.



```
ImportError                               Traceback (most recent call last)
<ipython-input-1-c15ae3402d12> in <module>()
----> 1 import pyspark

ImportError: No module named 'pyspark'


In [2]: import findspark

In [3]: findspark.init('/home/ubuntu/spark-2.1.1-bin-hadoop2.7')

In [4]: import pyspark

In [ ]:
```

So, if you have gone so far, you have successfully set up your working environment on EC2. You have Spark and you can program Spark application in Python using PySpark as the API. In addition, you can access your codes (and other stuff) through jupyter notebook systems.