Environment Setup

1. Install Python Environment

For data science, we recommend installing and using the Anaconda distribution of Python. This section details the installation of the Anaconda distribution of Python on Windows OS, Linux OS and MacOS. Anaconda is free (although the download is large which can take time) and can be installed on school or work computers where you don't have administrator access or the ability to install new programs. Anaconda comes bundled with about 600 packages pre-installed including **Pandas**, **NumPy**, **Matplotlib** and **Scikit-Learn**. These packages are very useful for data science.

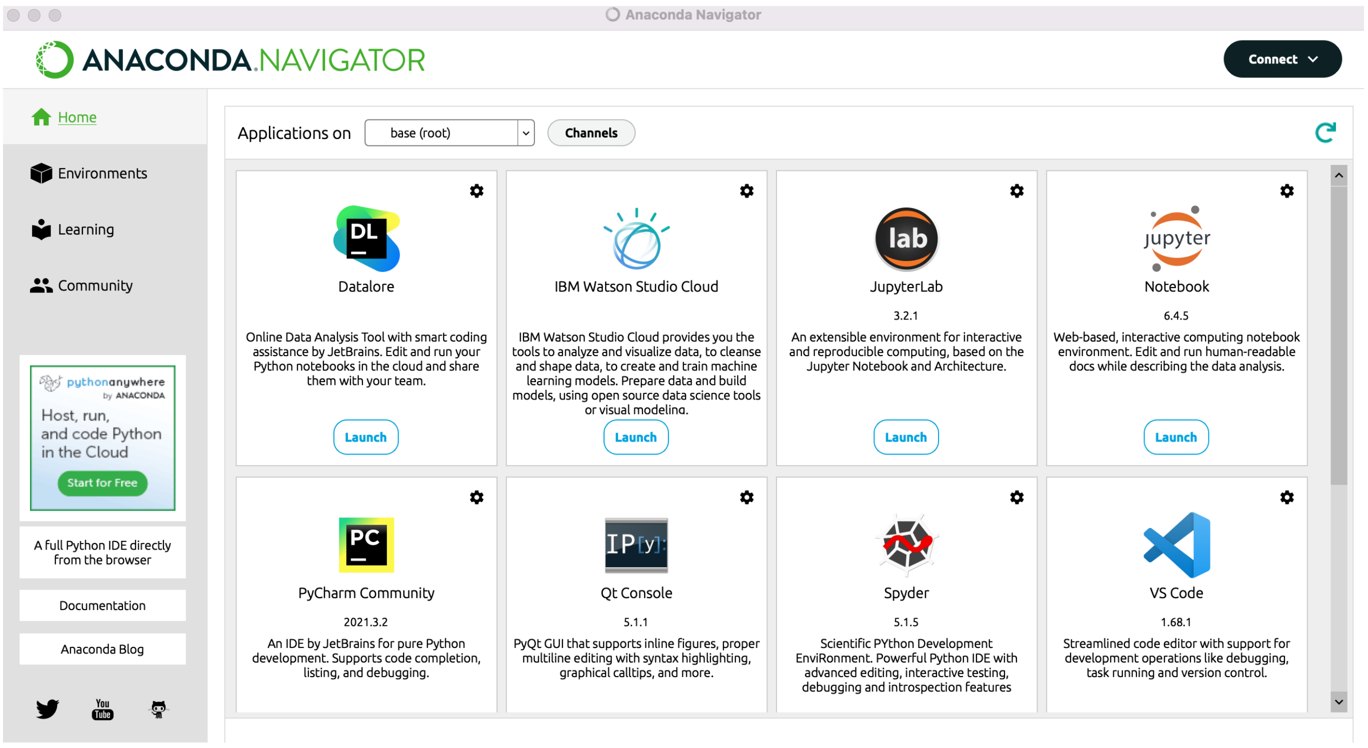
1.1. Install Anaconda Python on MacOS

This section details the installation of the Anaconda Distribution of Python on MacOS. Most versions of MacOS come pre-installed with legacy Python (Version 2.x). You can confirm the legacy version of Python is installed on MacOS by opening and running a command at the MacOS terminal. To open the MacOS terminal use [command]+[Space Bar] and type terminal in the Spotlight Search bar.

In the MacOS Terminal type (note: the dollar sign $ is used to indicate the terminal prompt. The dollar sign $ does not need to be typed): $ python. You will most likely see Python version 2.x is installed. Follow the steps below to install the Anaconda distribution of Python on MacOS.

Steps:

- Visit Anaconda
- Select MacOS and Download the **.dmg** installer (Python 3.9 version)
- Open the **.dmg** installer
- Follow the installation instructions
- Open a **Terminal** and type python and run some code



Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

An IDE by JetBrains for pure Python development. Supports code completion, listing, and debugging.

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Streamlined code editor with support for development operations like debugging, task running and version control.

1.2. Install Anaconda Python on Linux OS

To install Anaconda in Ubuntu 22.04, you must follow the below-given step-by-step instructions.

Steps:

- Visit Anaconda
- Select MacOS and Download the **.sh** installer (Python 3.9 version)
- Verify the Data Integrity of the installer with the command

> $ sha256sum Anaconda3-2022.05-Linux-x86-64.sh

- Run the Anaconda Script

> $ bash Anaconda3-2022.05-Linux-x86-64.sh

- Selection options (It is recommended that you type yes to use the conda command)
- Activate installation

> $ source ~/.bashrc

- Open a terminal and type python and run some code

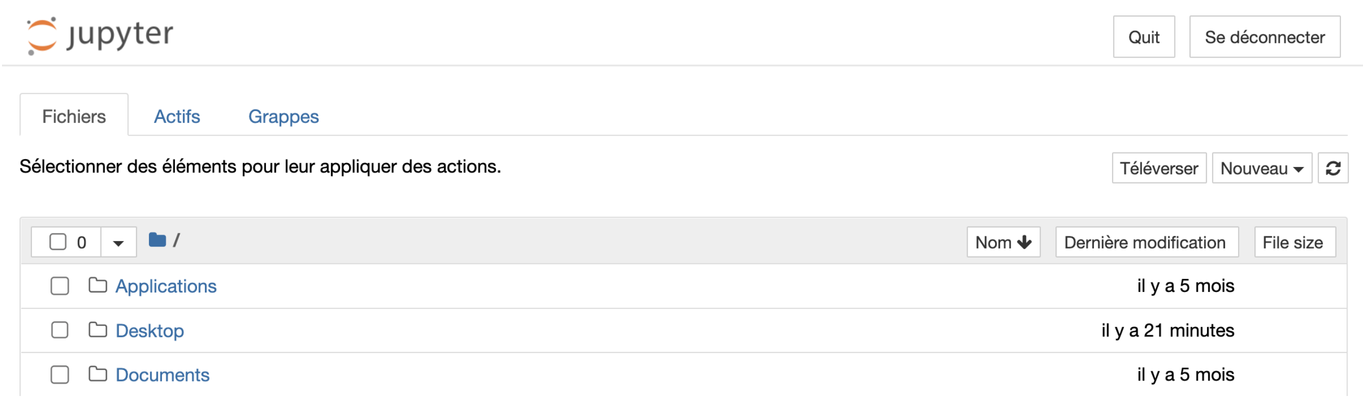
2. A Beginner’s Tutorial to Jupyter Notebook

A Jupyter Notebook is a powerful tool for interactively developing and presenting Data Science projects. Jupyter Notebooks integrate your code and its output into a single document. That document will contain the text, mathematical equations, and visualisations that the code produces directly in the same page.

2.1. Setting up a Jupyter Notebook

A Jupyter Notebook is installed by default with the Anaconda Python environment. To get started, use your terminal to move into the folder you would like to work from using the cd command (Linux or Mac). Then start up Jupyter with the following command: $ jupyter notebook

This will start up a Jupyter server and your browser will open up a new tab to the following URL: localhost. It’ll look a little something like this:



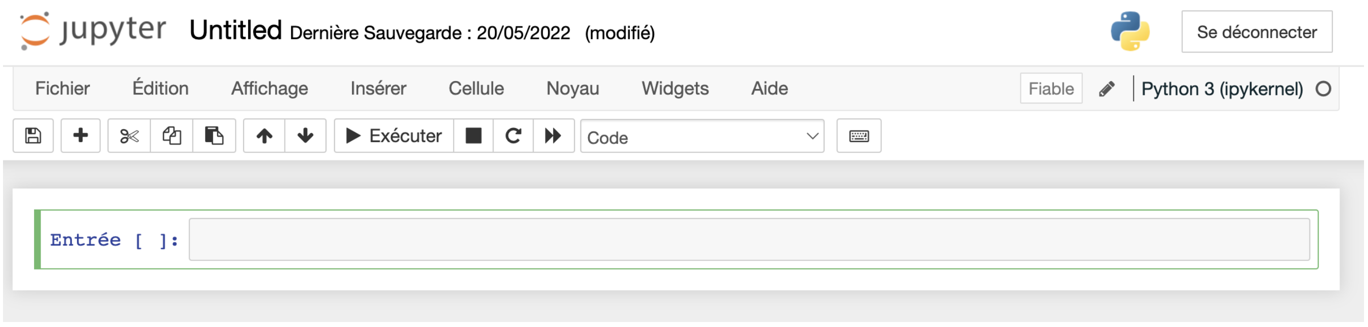
Sélectionner des éléments pour leur appliquer des actions.

Nom	Dernière modification
Applications	il y a 5 mois
Desktop	il y a 21 minutes
Documents	il y a 5 mois

Great! We’ve got our Jupyter server up and running. Now we can start building our notebook and filling it up with code!

2.2 Basics of Jupyter Notebook

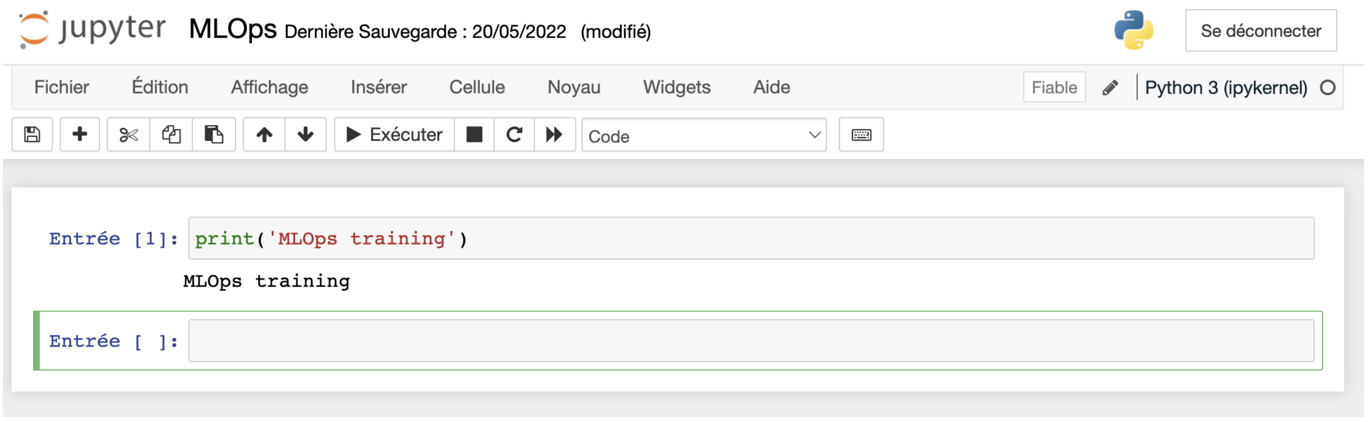
To create a notebook, click on the "new" menu in the top right and select "Python 3". At this point your web-page will look similar to this:


jupyter Untitled Dernière Sauvegarde : 20/05/2022 (modifié)
Se déconnecter
Fichier Édition Affichage Insérer Cellule Noyau Widgets Aide
Fiable
Python 3 (ipykernel)
Exécuter
Code
Entrée [ ]:

You'll notice that at the top of your page is the word Untitled next to the Jupyter icon; this is the title of your Notebook. Let's change it to something a little more descriptive. Just move your mouse over the word Untitled and click on the text. You should now see an in-browser dialog where you can rename your Notebook. We calling this notebook MLOps.

Let's start writing some code!

Notice how the first line of your Notebook is marked with an In [] next to it. That keyword specifies that what you are going to type is an input. Let's try writing a simply print statement there. Recall that your print statement must have Python 3 syntax since this is a Python 3 Notebook. Once you write your print statement in the cell, press the Run button.


jupyter MLOps Dernière Sauvegarde : 20/05/2022 (modifié)
Se déconnecter
Fichier Édition Affichage Insérer Cellule Noyau Widgets Aide
Fiable
Python 3 (ipykernel)
Exécuter
Code
Entrée [1]:
Entrée [ ]:

```
print('MLOps training')
```

```
MLOps training
```

2.3. Adding Descriptive Text to Your Notebook

Jupyter Notebooks come with a great set of tools for adding descriptive text to your notebooks. Not only can you write comments, but you can also add titles, lists, bold, and italics. All of this is done in the super easy Markdown format.

The first thing to do is to change the cell type – click the drop down menu that says "Code" on it and change it to "Markdown". This changes the type of cell we are working with.

Let's try out a couple of the options. We can create titles using the # symbol. A single # will make the biggest title and adding more #s will create a smaller and smaller title.

We can use italic in our text using a single star on either side or bold it using a double star. Creating a list is easy with a simple dash – and space beside each list item.

```
# super big title
## big title
### medium title
#### small title
```

```
*italics*
**blod**
```

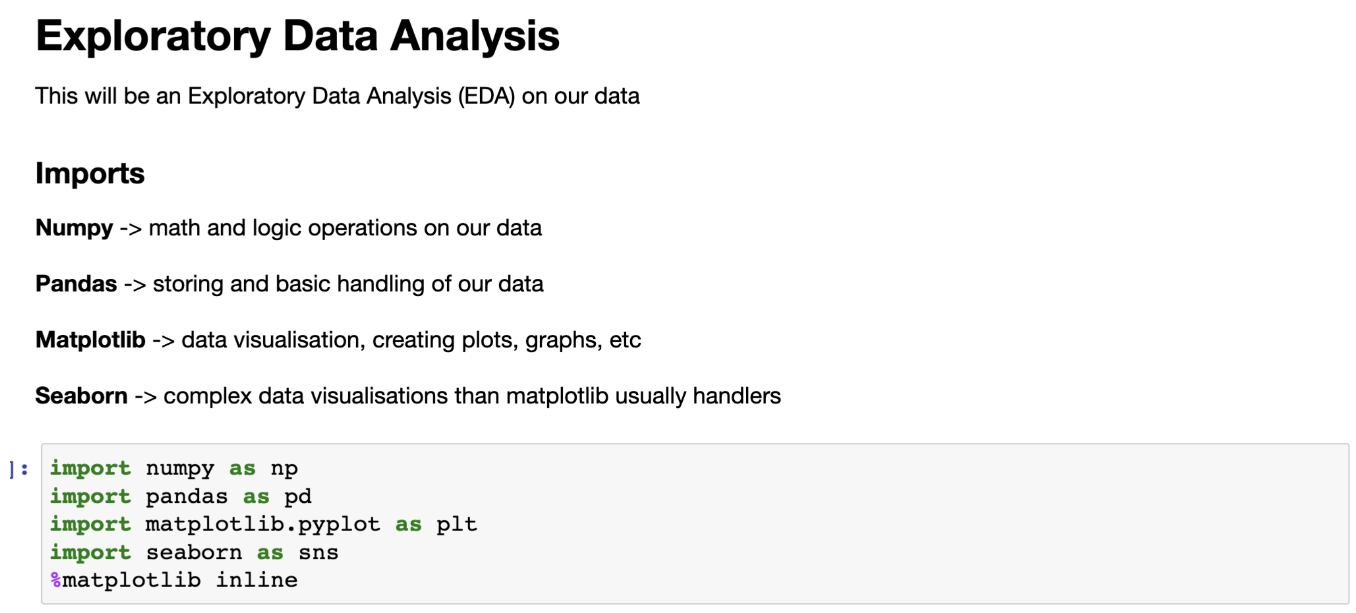
```
- item 1
- item 2
    - subitem 1
```

2.4. Interactive Data Science

Let's do a quick running example of how to create an interactive Data Science project. This notebook and code comes from an actual project.

We start out with a Markdown cell and put up a title with the biggest header by using a single # . We then create a list and description of all the important libraries about to import.

Next comes the first code cell which imports all of the relevant libraries. This will be standard Python Data Science code except for 1 additional item: in-order to see your Matplotlib visualisations directly within the notebook, you'll need to add the following line: %matplotlib inline.



Exploratory Data Analysis

This will be an Exploratory Data Analysis (EDA) on our data

Imports

Numpy -> math and logic operations on our data

Pandas -> storing and basic handling of our data

Matplotlib -> data visualisation, creating plots, graphs, etc

Seaborn -> complex data visualisations than matplotlib usually handlers

```
]: import numpy as np
   import pandas as pd
   import matplotlib.pyplot as plt
   import seaborn as sns
   %matplotlib inline
```

3. Create Python virtual environment

The venv module provides support for creating lightweight "virtual environments" with their own site directories, optionally isolated from system site directories. Each virtual environment has its own Python binary (which matches the version of the binary that was used to create this environment) and can have its own independent set of installed Python packages in its site directories

3.1. Create virtual environment on Mac Os

Install pip

Usually Python3 comes with pip preinstalled. If you get an error "pip command not found", use the following command to install pip:

```
$ sudo easy_install pip
```

Install virtualenv

Enter this command into terminal

```
$ sudo pip install virtualenv
```

or if you get an error

```
$ sudo -H pip install virtualenv
```

Start virtualenv

Navigate to where you want to store your code. Create new directory

```
$ mkdir my_project && cd my_project
```

Inside my_project folder create a new virtualenv

```
$ virtualenv env
```

Activate virtualenv

```
$ source env/bin/activate
```

3.2. Create virtual environment on Linux Os

Update system

```
$sudo apt update sudo apt upgrade
```

Install pip for python3

```
$ apt install python3-pip
```

Create virtual environment

```
$ apt install python3-venv

$ python3 -m venv mlops_project
```

The above command creates a directory named 'mlops_project' in the current directory, which contains pip, interpreter, scripts, and libraries

```
$ ls my_env_project/ bin include lib lib64 pyvenv.cfg share
```

Activate it

```
$ source lmops_project/bin/activate
```

4. Install Git Environment

For any programmer, today Version controlling is becoming an essential tool.

Version control systems are a category of software tools that help a software team manage changes to source code over time. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members.

The most widely used version control system today is Git. Git is a Distributed Version Control System. That means rather than having only one single place for the full version history (centralized repository) of the software, every developer's working copy also contains the full history of all changes. Git has been designed with performance, security, and flexibility in mind. Thus, the most popular and most used.

Before you can start using Git, it must be available on your computer. Even if it's already installed, it's probably a good idea to use the latest version available. You can install it either as a package or with an installer, or by downloading the code and compiling it yourself.

4.1. Install Git on Mac Os

There are several ways to install Git on a Mac. The easiest is probably to install Xcode Command Line Tools. On Mavericks (10.9) or later, you can simply try running git in the terminal the first time.

> $ git –version

If it is not already installed, it will ask you to do so.

If you want a more up-to-date version, you can also install it from the binary installer. A Git installer for macOS is maintained and available for download from the Git website at git-scm.

4.2. Install Git on Linux OS

If you want to install the basic Git tools on Linux via a binary installer, you can usually do so using the package management tool that comes with your distribution. On a Debian-based distribution, such as Ubuntu, try apt:

> $ sudo apt install git-all

For more options, instructions for installing on different Unix versions are available on the Git website at git-scm.

5. A Beginner's Tutorial to Git

This section features the most important and commonly used Git commands for easy reference

git config

- You must first configure Git before you can use it
- The username aand email address that will be used with your commits can be specified using this command

> sets up Git with your name
>
> $ git config --global user.name "mention username"

```
sets up Git with your email

$ git config --global user.email "mention email"
```

git init

- A git repository must first be created before you can make commits or do anything else with it
- This command is used to initiate a new git repository (create .git directory)

```
$ git init
```

git clone

- git clone command line utility used to create a local copy of a remote repository
- It uses a remote URL to access the repository
- The original repository is usually hosted on aremote server, usually through a Git service such a GitHubn Bitbucket, or Gitlab

```
// git clone [repo url]

$ git clone https://MLOps/mlops.git
```

git add

- Use the git add command to move files from the Working Directory to the Staging Index
- The git add command stores your modifications to the staging area in afile, allowing you to compare your local version to the version in the remote repository
- Use the git add command to add your new or update file to the staging area before commiting it

```
// to add specific files

$ git add file1, file2

// to add all files

$ git add .
```

git commit

- This command recors a log message as well as the commit id of the git repository's modifications
- With git commit, you can save your changes to your local repository
- Every time you commit code changes, you must include a concise summary of the changes. This commit message assists others in understanding the chages that have been made

```
git commit -m "Type your descriptive commit message here"
```

git push

- The git push command is used to upload rlocal epository content to remote repository
- Pushing is how you transfer commits from your local repository to remote repo
- Here git push origin (origin means remote) master (main branch name)

```
$ git push origin master
```

git branch

The git branch command lets you create, list, rename, and delate branches

```
// To view a list of your local Git branches, you can run

$ git branch

// To view all remote branch

$ git branch -r

// To create a new branch

$ git branch ["branch-name"]

// To delate any branch

$ git branch -d ["branch name that need to be deleted"]
```

git checkout

- We can use the git chekout command to switch to an existing branch
- It can also used to created and switch to a new one branch (then no need to fisrt use git branch to creat new branch, git checkout will do both)

```
// switch from current branch to develop branch

$ git checkout develop

// create and switch branch to develop in single command

$ git checkout -b develop
```

git log

- The git log command is used to show all of a repository's commits
- This comamnd doispalys a log of all commits made to current branch so far or it is used to list the version history for the current branch

```
$ git log
```

git merge

- git merge is used to combine the changes of two branches
- If there is no conflict while merging it will combine the changes with commit message
- In case of conflict, developer need to resolve the conflict first then need to merge the changes
- You should be on the branch you want to merge with your parent branch before running the git merge command

```
$ git merge "name-of-branch-to-merge-in"
```

6. Install Docker Engine

Docker Engine is an open source containerization technology for building and containerizing your applications. Docker Engine acts as a client-server application with:

- A server with a long-running daemon process dockerd
- APIs which specify interfaces that programs can use to talk to and instruct the Docker daemon
- A command line interface (CLI) client docker

6.1. Install Docker on Mac Os

Docker Desktop helps you build, share, and run containers easily on Mac.

Download Docker Desktop for Mac (with Intel chip or Apple chip)

After downloading Docker.dmg, run the following commands in a terminal to install Docker Desktop in the Applications folder:

```
$ sudo hdiutil attach Docker.dmg

$ sudo /Volumes/Docker/Docker.app/Contents/MacOS/install

$ sudo hdiutil detach /Volumes/Docker
```

As macOS typically performs security checks the first time an application is used, the install command can take several minutes to run

6.2. Install Docker on Linux Os

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker repository. Afterward, you can install and update Docker from the repository

Set up the repository

Update the apt package index and install packages to allow apt to use a repository over HTTPS:

```
$ sudo apt-get update

$ sudo apt-get install \ ca-certificates \ curl \ gnupg \ lsb-release
```

Add Docker's official GPG key:

```
$ sudo mkdir -p /etc/apt/keyrings

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
```

Use the following command to set up the repository:

```
echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \

$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Install Docker Engine

Update the apt package index, and install the latest version of Docker Engine, containerd, and Docker Compose, or go to the next step to install a specific version:

> $ sudo apt-get update
>
> $ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin

Install a specific version

> $ sudo apt-get install docker-ce=<VERSION_STRING> docker-ce-cli=<VERSION_STRING> containerd.io docker-compose-plugin

Verify that Docker Engine is installed correctly by running the hello-world image

> $ sudo docker run hello-world