Lab 01: Getting Started

Installation and Setup

1. Install Python v3.9 from https://www.python.org/downloads/release/python-3912/ . Select a downloadable file based on your operating system. Use custom installation (install it in C:\Python39) and mark 'add to path'.
2. Install stable build VSCode from https://code.visualstudio.com/.
3. In VSCode press Ctrl+Shift+X to open the Extensions tab.
4. Install Jupyter extension (released by Microsoft). The one you select should have around 43M downloads.
5. Install Python extension (released by Microsoft). The one you select should have over 62M downloads.
6. Go to View->Command Palette (or use Ctrl + Shift + P).
7. Start typing Python:Select Interpreter.
8. Choose the Python Interpreter ending in C:\Python39\python.exe (if you already have another python version installed, you may stick to that).
9. Hover on view -> Command Palette -> just type jupyter then you will be given an option out of many to Create a new blank notebook.
10. Enter print("Hello World!") and run it to test your installation (Ctrl + Alt + Enter). 11. We are done with the installation of the software. Now we need to create an environment to run a code.
12. Open a terminal in VSCode from view-> terminal or use (Ctrl + `).
13. Navigate to the directory where you have the Python Interpreter (python.exe) (see page 2 for navigation tips in the terminal)
14. Run this command: py -m pip install numpy
15. Run the same command replacing numpy with pandas, matplotlib, sklearn 16. You can also do that in jupyter notebook using "pip install matplotlib".
17. Test matplotlib:

    ```
    import matplotlib.pyplot as plt
    import numpy as np
    plt.plot(np.random.randn(50))
    ```

    For additional information see:
    https://code.visualstudio.com/docs/datascience/jupyter-notebooks

Description of the Command MacOS/LINUX Windows

For Navigating Directories

List all the files and directories in the current working directory ls ls List of all open files on the system lsof lsof To change/ the current directory* cd cd To print the path of the current working directory pwd pwd To save current location path and go to a new location pushd pushd To return to the saved location popd popd

To Modify Files and directories

To create/make a new directory at the current directory mkdir mkdir To remove a directory from the current directory rmdir rmdir To move/rename a file/directory in the current directory mv move To copy a file/directory cp cp
To remove a file from the directory (CAUTION: cannot undo it) rm rm

To View a File in the Terminal

To page through a file less more To display the content of an existing file cat type To create an empty new file touch echo To find a file in the current directory find dir -r To search inside a file grep findstr

Authority and helps

Become superuser of the system sudo runas Read the manual page man help The network used by the computer hostname hostname To change the access permission or mode of permission chmod To change/transfer the file ownership chown To download files from the internet wget wget To terminate unresponsive program/task kill taskkill To check all the jobs with their statuses jobs To clear the terminal clear cls To exit the shell/terminal exit exit

Some important editing command

To go to insert mode – insert at cursor -i -i To go to insert mode – insert after cursor -a -a
To forcefully delete a directory (CAUTION: Do-rf -rf
no use if there is other ways)

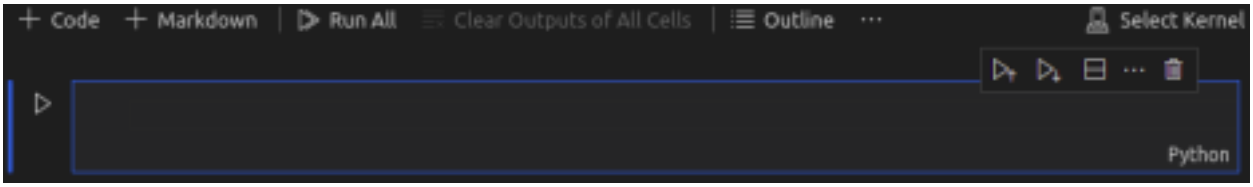
Important Symbols

Wild card/ equivalent to any number of symbol or character * * Current directory . . Go on level up directory form the current directory .. .. Return to root directory ~ ~

Quick Guide: Jupyter Notebook (VSCode)

After the successful installation of the VSCode and extensions (Jupyter and Python) as illustrated in the lab section, the command **Ctrl + Shift + P** comes handy to create a new Jupyter Notebook File.

The first thing a user notices on the Jupyter Notebook is the cell section.



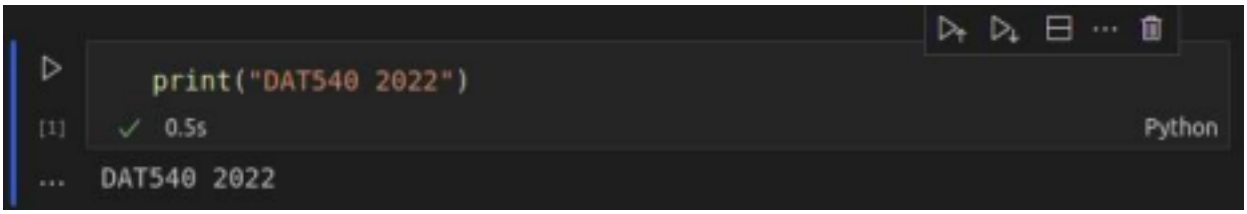
A cell is a container for text to be displayed in the notebook or code to be executed by the notebook's kernel.

Other than that, a *kernel* is the computational engine that executes the code contained in the notebook document.

The *kernel* can be selected or changed with the **Select Kernel** section as displayed in the image on the right where you can see multiple options and you can choose with your version of Python.

Cells

Cells form the body of the notebook. In the screenshot above, the blue marked section is the code shell. There are two types of cells which we will cover and use on a regular basis.



The **code cell** is the section where we write the code and execute it. When the code is executed, the result is seen below the cell.

You can add a code cell with **Esc + a (above), Esc + b (below)**.

The **markdown cell** is the section where we write our markdown and execute which outputs the text in *Markdown Format.*

Quick Hints

Whenever a new notebook is opened, the first cell is always a code cell.

To run a cell, we can click the button to the left of the cell or either press **Ctrl + Enter**.

The **+ Code** button generally adds up a new cell in the notebook. Similarly, the **+ Markdown** button adds a new markdown cell in the notebook.

The **Run All** button runs every cell in the notebook from the start. It stops when the error is encountered and the errors are printed right below the cell where the error is.

The **Clear Output of all Cells** button clears the output of all the cells and the clean code and markdown format is shown.

The **Restart** button restarts the kernel. This button is handy when we face kernel crashes and problems.

The **Variables** button is handy when debugging. The button displays all the names of the variables assigned in the notebook with the type of the value, the size and the value itself.