



**POLITECNICO  
DI TORINO**



# Data Science Lab

Introduction to Python

Database and Data Mining Group

Andrea Pasini, Elena Baralis



## Summary

- **Python engine**
  - Basic components and setup
- **Python language**
  - Data types, object oriented programming
- **Numpy library**
  - Computation with multi-dimensional arrays
- **Pandas library**
  - Tabular data and data preprocessing
- **Scikit-Learn library**
  - Machine learning and data science tools



# Introduction to Python



- Python language

- Clean and concise syntax
  - No semi-colons to end instructions
  - No braces to define if clauses and for loops
  - No need to specify variable types
  - ...

Java

```
List<String> l = new LinkedList<>();  
for (int i=0; i<10; i++) {  
    l.add(i);  
}
```

Python

```
l = []  
for i in range(0,10):  
    l.append(i)
```

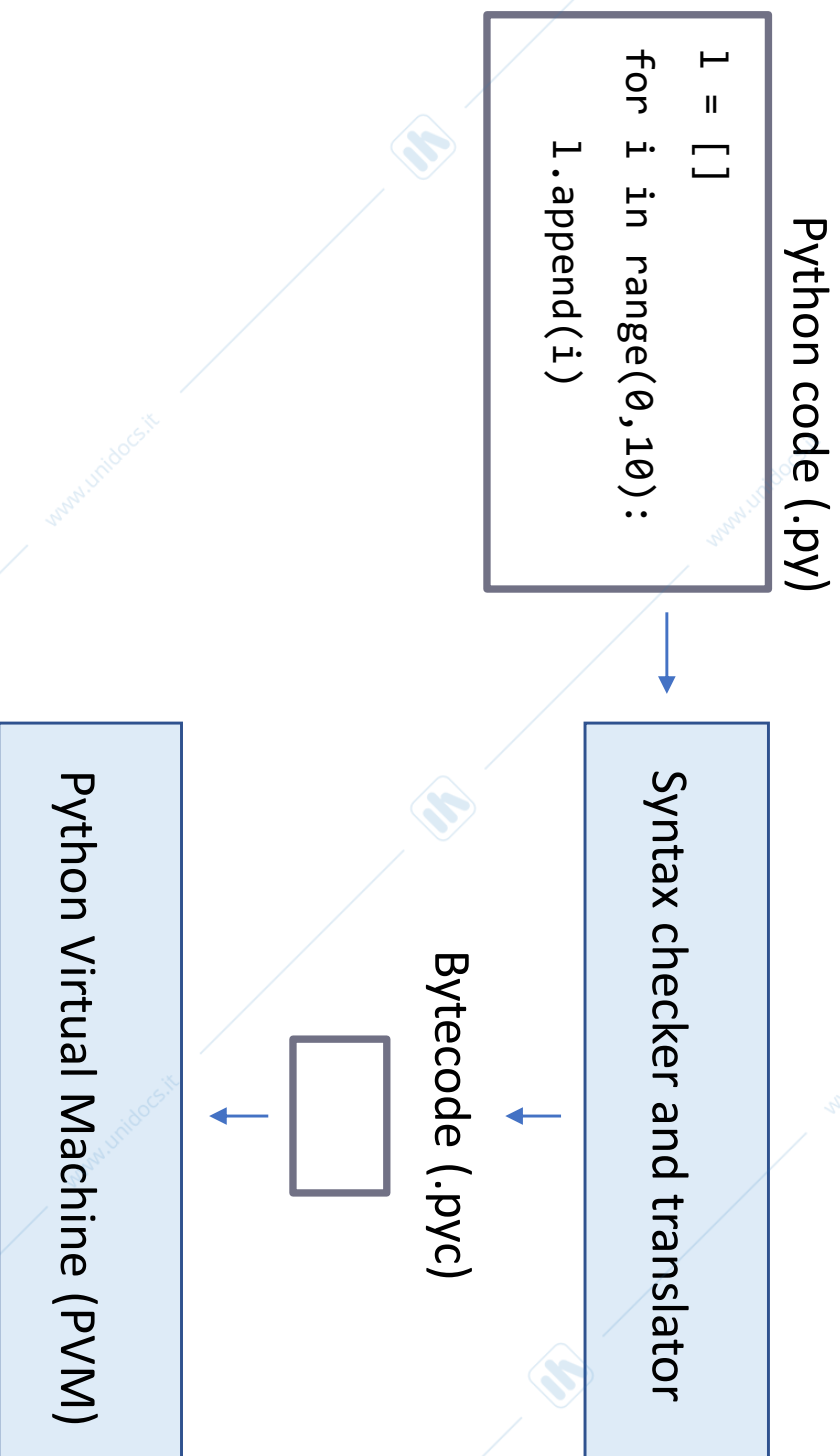


# Introduction to Python

- Python is an **interpreted** language
  - Code is not compiled to machine language
  - However the source code is compiled to an intermediate level, called **bytecode**
  - For this reason, to run Python programs, you need an **interpreter** that is able to execute the bytecode



- Sequence of operations executed by the interpreter





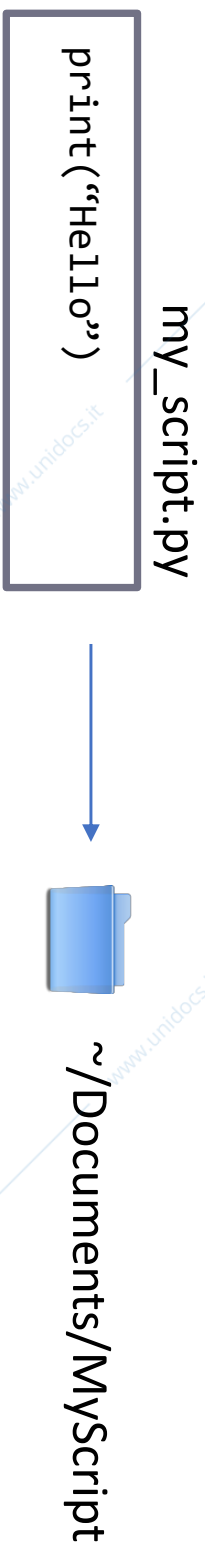
# Introduction to Python

- A common Python 3 setup on a **Linux System**
- Typically in the `usr/bin` folder:
  - “**python3**” executable: run Python programs
  - “**pip3**” executable: install Python packages
  - “**ipython3**” executable: run programs line by line
  - “**jupyter**” executable: run a jupyter notebook



# Introduction to Python

- Executing a Python program



- Type in your terminal:
  - `cd ~/Documents/MyScript`
  - `python3 my_script.py`



# Introduction to Python



- Running Python line by line with iPython
- Type in your terminal:
  - ipython3 (or ipython, depending on your installation)

```
IPython: home/andrea
File Modifica Visualizza Cerca Terminale Aiuto
andrea@andrea:~$ ipython3
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
Type "copyright", "credits" or "license()" for more information.
```



# Introduction to Python



- Write your program line by line to see the results step by step...

```
IPython: home/andrea
File Modifica Visualizza Cerca Terminale Aiuto
andrea@andrea:~$ ipython3
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
Type "copyright", "credits" or "license()" for more information.

IPython 5.5.0 -- An enhanced Interactive Python.
?                -> Introduction and overview of IPython's features.
%quickref        -> Quick reference.
help             -> Python's own help system.
object?         -> Details about 'object'? for extra details.

In [1]: mystring = "hello"

In [2]: print(mystring)
hello

In [3]:
```



# Introduction to Python



- Python and **iPython** programs are the core for executing scripts, but...
- There are two typical scenarios:
  1. Develop your Python **project** with an **IDE**
    - Example: Visual Studio Code, PyCharm
    - **Debug** and **run** your code inside the IDE
  2. Develop and test a Python **script** with **Jupyter notebook**
    - Inspect **step by step** the results
    - Keep the history of the output of the script



# Introduction to Python



## Scenario 1: PyCharm (IDE)

Run/Debug commands

The screenshot shows the PyCharm IDE interface. The top toolbar contains icons for File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The main window is divided into three panes: Project, Run, and Code. The Project pane on the left shows a tree view of the project structure, including folders like 'classes', 'deplab', 'maskrcnn', 'maskrcnn\_matterport', 'outdir', 'panopticiapi', 'semantic\_analysis', and files like 'algorithms.pyx', 'master.zip', 'main\_inference.py', 'main\_inspection.py', 'main\_merge.py', and 'main\_sad.py'. A blue box labeled 'Project overview' points to this pane. The Run pane in the middle shows the execution progress of 'main\_sad.py'. The Code pane on the right shows the source code of 'main\_sad.py'. A blue box labeled 'Code' points to this pane. The code includes a main function that prints the duration of the execution. A blue arrow points from the 'Run/Debug commands' text to the Run/Debug icon in the toolbar.

```
end_time = datetime.now()
print("Done.")
print('Duration: ' + str(end_time - start_time))

if __name__ == "__main__":
    start_time = datetime.now()
    chunk_size = 10 # number of images processed for each task
    num_processes = 10 # number of processes where scheduling tasks

    #TODO: use training images, instead of validation
    input_images = './COCO/images/val2017/'
    run_tasks('./COCO/annotations/panoptic_val2017.json', './COCO/annot
end_time = datetime.now()
print("Done.")
print('Duration: ' + str(end_time - start_time))

if __name__ == "__main__"
```

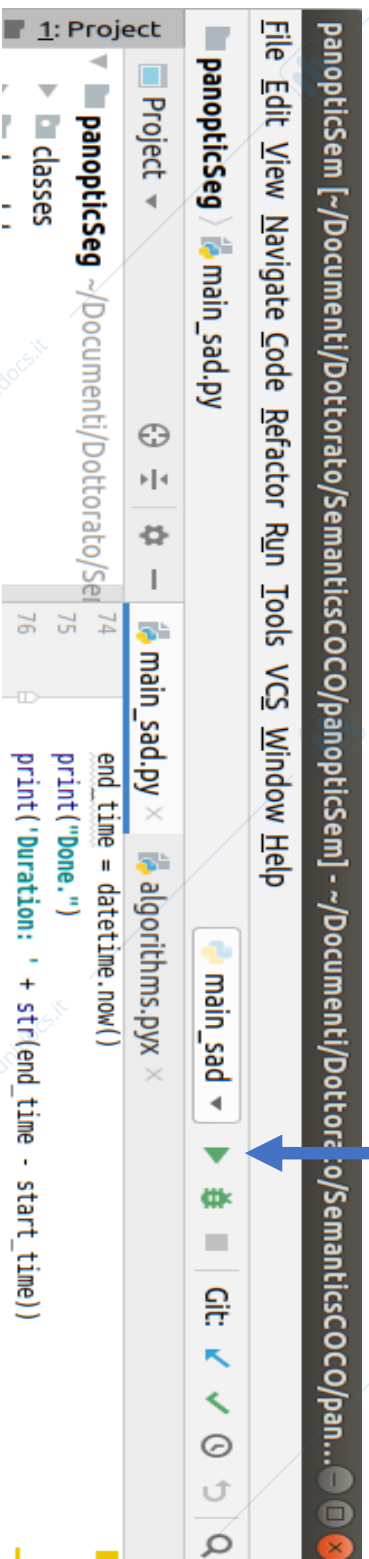


# Introduction to Python



- **Scenario 1: PyCharm (IDE)**
  - When you click on the run button, the IDE will automatically call the “python” command to execute your script

Run command

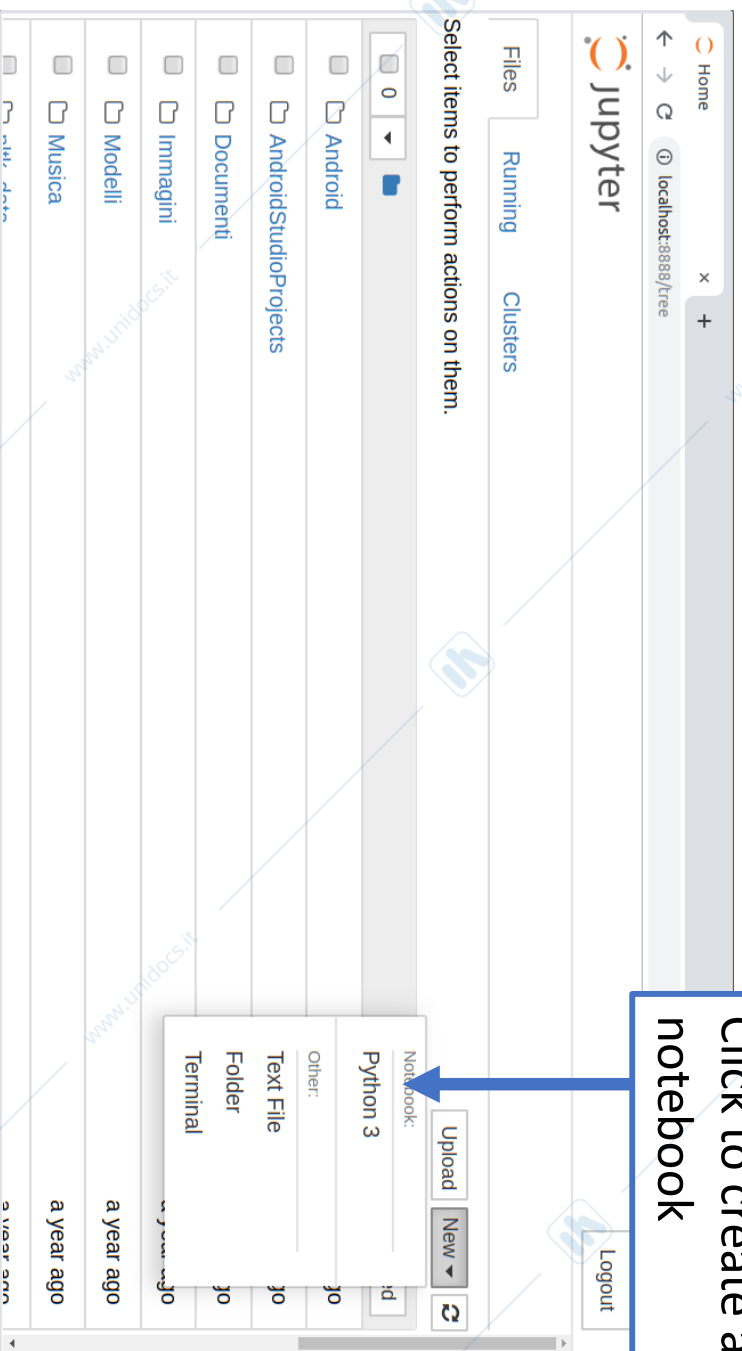




# Introduction to Python



- **Scenario 2: Jupyter notebook**
- Type in your terminal
- jupyter notebook
- Jupyter will open on your browser



Click to create a new notebook



# Introduction to Python



## ■ Scenario 2: Jupyter notebook



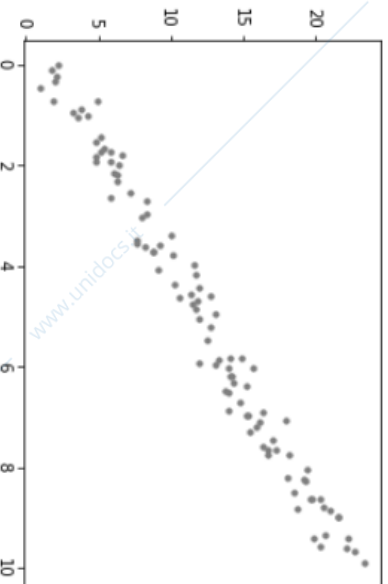
Markdown cell

### 1. Simple linear regression Generating a dataset

Code cell

```
In [26]:  
# Make dataset  
err = np.random.normal(0,1, 100) # gaussian data, mean=0, std=1  
x = 10*np.random.rand(100) # 100 data points in [0, 10]  
y = (2*x + 2) + err # target is a linear function of the input with some  
  
In [27]:  
# Plots  
plt.scatter(x, y, s=10, c='grey')  
plt.show()
```

Result cell





- **Scenario 2: Jupyter notebook**
  - Based on **iPython** command
  - Each code **cell** can be executed **separately** by pressing **CTR + ENTER**

```
In [261]:  
# Make dataset  
err = np.random.normal(0,1, 100)  
x = 10*np.random.rand(100)  
y = (2*x + 2) + err  
  
# gaussian data, mean=0, std=1  
# 100 data points in [0, 10]  
# target is a linear function of the i  
  
In [271]:  
# Plots  
plt.scatter(x, y, s=10, c='grey')  
plt.show()
```

Code cell 1

Code cell 2



# Introduction to Python

## IDE vs Jupyter notebook

### ■ IDE

- For more **complex** projects (many files)
- More powerful debug commands
- More powerful code editing tools

### ■ Jupyter notebook

- For simple scripts and prototypes
- Great **visualization** tool
  - Example: **report** with Python code and text for explanations



## ■ Installing libraries

- Python language is provided with many useful libraries:
  - Numpy, Pandas, Matplotlib, Scikit-learn, SciPy, ...
- To use any of them you first have to install it with the **pip** command:
  - pip3 install numpy
  - pip3 install pandas

```
andrea@andrea  
File Modifica Visualizza Cerca Terminale Aiuto  
andrea@andrea:~$ pip3 install numpy
```



# Introduction to Python



## ■ Virtual environments

- The pip command will associate the libraries to your **default Python installation**
- A more powerful way of managing libraries is to use a **Python environment (virtualenv)**
  - Designed when you want to design **different projects** that use different libraries and **configurations (e.g. versions)**
  - Each projects is associated to a virtual environment



# Introduction to Python

- **Virtual environments**

- To create a new environment:
  - `cd ~/Documents/My_project`
  - `virtualenv myenv`
- It will create a new environment in your project folder

The screenshot shows a terminal window titled "Documenti MyProject". The user is in the directory ~/Documents/MyProject. The terminal output shows the following commands and their results:

```
andrea@andrea-XPS-13-9360: ~/Documents/MyProject
andrea@andrea:~/Documents/MyProject$ virtualenv myenv
Running virtualenv with interpreter /usr/bin/python2
New python executable in /home/andrea/Documents/MyProject/myenv/bin/python2
Also creating executable in /home/andrea/Documents/MyProject/myenv/bin/python
Installing setuptools, pkg_resources, pip, wheel...done.
andrea@andrea:~/Documents/MyProject$
```



# Introduction to Python



- **Virtual environments**

- **To activate the created environment:**
  - `cd ~/Documents/My_project`
  - `source myenv/bin/activate`

```
File Modifica Visualizza Cerca Terminale Aiuto
andrea@andrea:~/Documenti/myProject$
(myenv) andrea@andrea:~/Documenti/myProject$
```



# Introduction to Python

## ■ Virtual environments

- After activation you can use the terminal to work within the environment

```
File Modifica Visualizza Cerca Terminale Aiuto
andrea@andrea:~/Documenti/MyProject$
(myenv) andrea@andrea:~/Documenti/MyProject$
```

- Install libraries to the current environment
  - pip3 install my\_library
- Execute a script/notebook within the environment
  - python3 my\_script.py
  - jupyter notebook