



**Dana-Farber**  
Cancer Institute

# Intro to Python

**James L**

# James Lindsay

Director, Software @ KSG in  
Department of Data Science

using python since 2007-ish

good programmer !=  
effective programmer

---

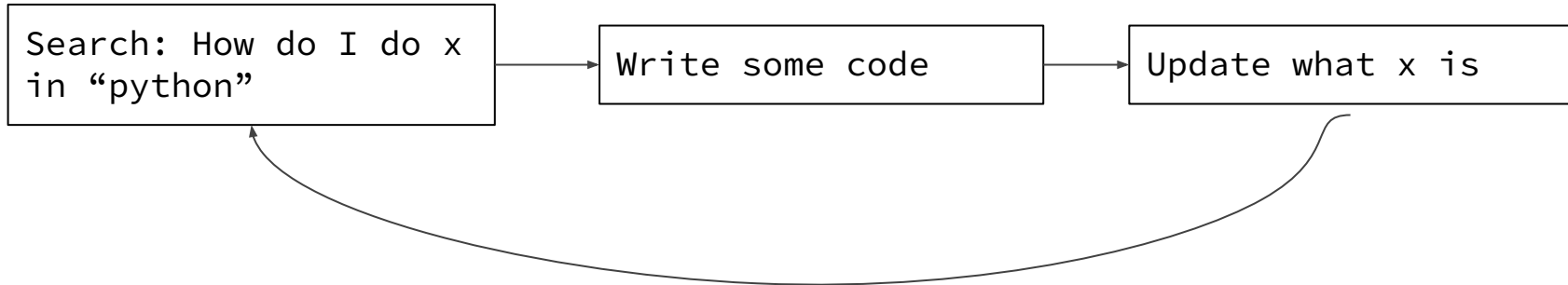
# Assumptions and background

- You must have some exposure to programming concepts (variables, for-loops, print statements). If not: <https://www.py4e.com/>
- You can only learn so much in one sitting, you will need to use Python and find more courses to level up
- Get a copy of these slides and example programs here: {TODO}

# Why python programming is great

You are almost never the first person to have done something so just ask Google / Stackoverflow your question

## Anatomy of writing a python script



# Tutorial overview

1. Hello world
2. Baby steps (python control flow)
3. CSV parsing (the hard way)
4. Libraries (pip/conda)
5. CSV parsing with pandas (the easy way)
6. Stretch goal: Jupyter notebooks

# Project 0: Hello World

- Write a python script that prints “Hello World” to the command line
- Run the program like ````python project0.py````

## **Expected output**

Hello World

# Project 1: Baby steps (control flow)

## Write a program that:

- Uses the `__name__ == "__main__"` (\*) entry point
- Calls a function called `load_data` which first sleeps for 3 seconds, then returns a string
- Print the value returned by `load_data` to stdout
- Include statements with unique outputs:
  - 0: at the very top of the script
  - 1: start of entrypoint
  - 2: start of `load_data`
  - 3: after sleep in `load_data`
  - 4: prints results of `load_data` to stdout

\*`__name__` is a built-in variable which evaluates to the name of the current module

# Project 1: Baby steps (control flow)

## **Expected output**

```
```python project1_babysteps.py```
```

```
0: Hello, I am here!  
1: This is the second print statement  
2: ... now I am here, going to sleep  
3: ..waking up, loaded the data  
4: here is your data:  Data loading
```



# Project 2: CSV parsing

## Problem statement

Calculate the mean mutational and neoantigen load for patients with clinical benefit (CB), no clinical benefit (NCB) or long-term survival (LTS-NCB)

PD=progressive disease  
PR=partial response  
CR=complete response  
SD=stable disease

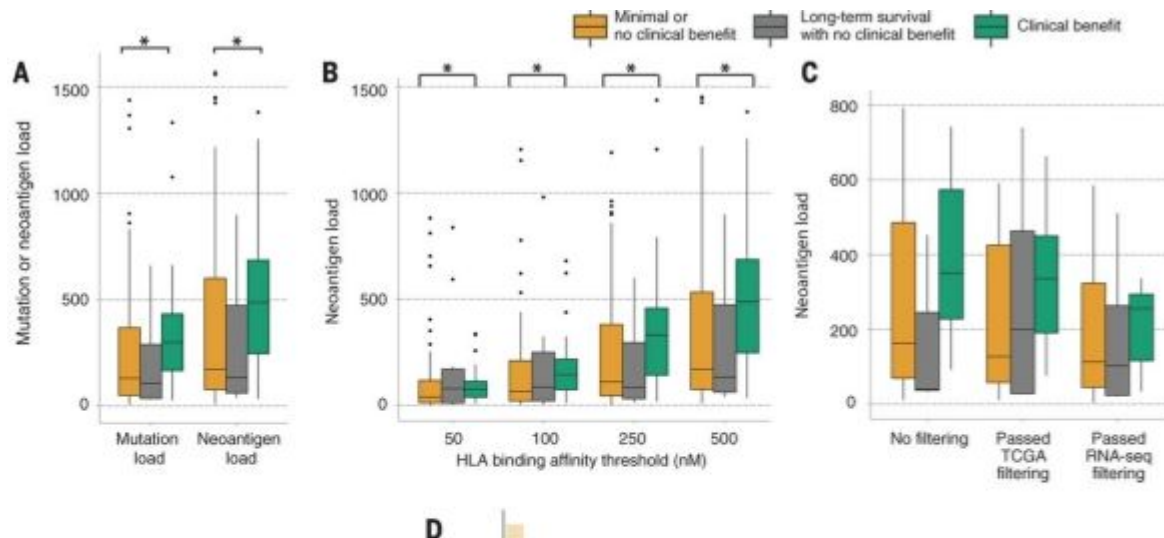


Fig. 2. Overall mutational load, overall neoantigen load, and expression-based neoantigen analysis as predictors of response to ipilimumab (A) Elevated nonsynonymous mutational load and neoantigen load are associated with response to ipilimumab ( $P = 0.0076$  and  $0.027$ , respectively). An additional 20 points are not shown because of outlying high neoantigen loads in a subset of patients.

<https://pubmed.ncbi.nlm.nih.gov/26359337/>

# Project 2: CSV parsing

## Fetch data

[cbioportal.org](https://cbioportal.org), fetch the clinical data from “Metastatic Melanoma (DFCI, Science 2015)”

- Click on study view
- Select all 59 columns
- Click the download (down arrow) to get a TSV

The screenshot shows the cBioPortal interface for the study "Metastatic Melanoma (DFCI, Science 2015)". The interface includes a navigation bar with links like "Data Sets", "Web API", "Tutorials/Webinars", "FAQ", "News", "Visualize Your Data", "About", and "cBioPortal Installations". A search bar is present with the text "Click gene symbols below or enter here". Below the search bar, there are tabs for "Clinical Data" and "CN Segments". A button labeled "Columns" is highlighted with a red circle. A dropdown menu is open, showing a list of columns to be selected. The columns include "Name", "Freq", "Age at Diagnosis", "Cancer Studies", "Cancer Type", "Cancer Type Detailed", "Case Lists", "Durable Clinical Benefit", "HLA-A", "HLA-B", "HLA-C", "KM Plot: Disease Free Survival (months)", "KM Plot: Overall Survival (months)", "M Stage", "Mutation Load", and "Neo-antigen Load". The "Columns" button is also highlighted with a red circle. The table below shows clinical data for 110 patients and 110 samples.

Sample ID	Cancer Study	Cancer Type	Cancer Type Detailed	Number of Samples Per Patient	Mutation Count	Fraction Genome Altered	Age at Diagnosis
Pat39	skcm_dfci_2015	Melanoma	Cutaneous Melanoma	1	73	0.3686	61
Pat47	skcm_dfci_2015	Melanoma	Cutaneous Melanoma	1	135	0.2460	71
Pat63	skcm_dfci_2015	Melanoma	Cutaneous Melanoma	1	208	0.2812	64
Pat03	skcm_dfci_2015	Melanoma	Cutaneous Melanoma	1	367	0.2141	61
Pat06	skcm_dfci_2015	Melanoma	Cutaneous Melanoma	1	170	0.0964	31
Pat08	skcm_dfci_2015	Melanoma	Cutaneous Melanoma	1	251	0.5895	71
Pat100	skcm_dfci_2015	Melanoma	Cutaneous Melanoma	1	503	0.4667	71
Pat101	skcm_dfci_2015	Melanoma	Cutaneous Melanoma	1	50	0.7724	75

[http://www.cbioportal.org/study/clinicalData?id=skcm\\_dfci\\_2015](http://www.cbioportal.org/study/clinicalData?id=skcm_dfci_2015)

# Project 2: CSV parsing

## Expected output

```
```python project2_csv.py```
```

```
clinical_benefit: NCB, 74  
NCB, Mutation Load mean: 396.81  
NCB, Neo-antigen Load mean: 315.99
```

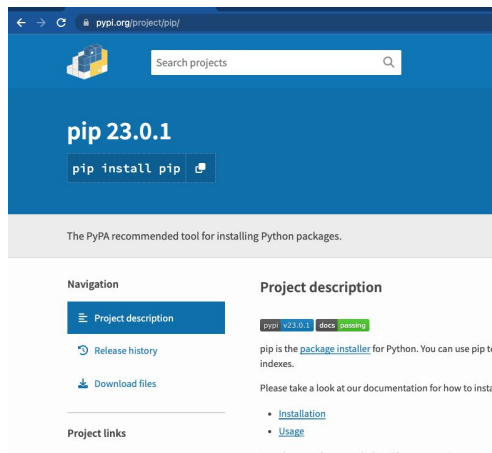
```
clinical_benefit: LTS-NCB, 10  
LTS-NCB, Mutation Load mean: 571.50  
LTS-NCB, Neo-antigen Load mean: 383.00
```

```
clinical_benefit: CB, 26  
CB, Mutation Load mean: 737.15  
CB, Neo-antigen Load mean: 467.69
```

**Note:** These results don't seem to map up with the paper... oops? Bonus if someone can figure out why

# Libraries, and virtual environments

- Best part of python is all the tools already built!
- There are two major repositories of libraries you can use:
  - pypi / pip → the OG
  - Anaconda / conda → packages with pre-built binaries



Products ▾ Pricing Solutions ▾ Resources ▾

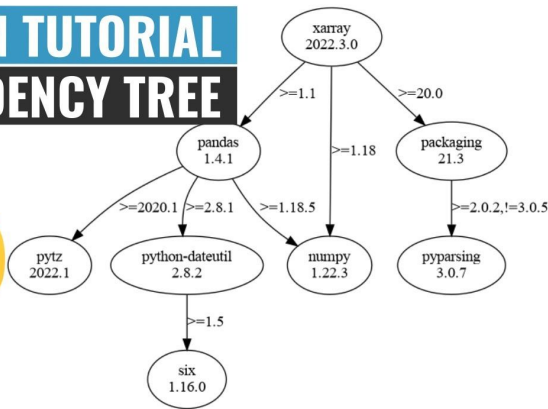
Individual Edition is now

## ANACONDA DISTRIBUTION

The world's most popular open-source Python distribution platform

# Libraries, and virtual environments

## PYTHON TUTORIAL DEPENDENCY TREE



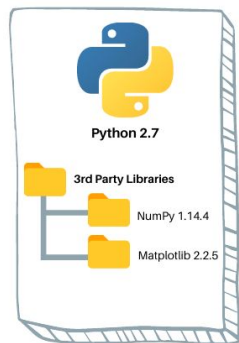
Python projects / libraries are a complicated mix of other python projects / libraries... we call these dependencies!

pip and conda, not only install the tool you want, but also its dependencies

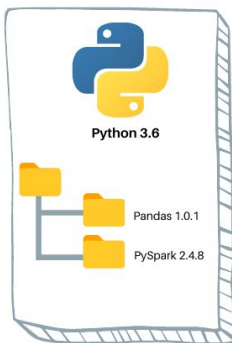
However sometimes those dependencies can clash, hence virtual environments

# Libraries, and virtual environments

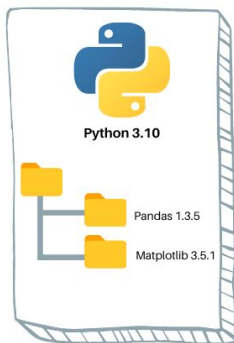
Virtual Environment 1



Virtual Environment 2



Virtual Environment 3



dataquest.io

Virtual environments are stand-alone copies of Python on your computer with (mostly) independent dependencies

Each project gets its own virtual environment

Conda is both a package and environment manager.

Pypi/pip use “virtualenv”

# Libraries, and virtual environments

## Try it (pip/virtualenv):

```
```\npython3 -m pip install --user --upgrade pip\npython3 -m pip --version\npython3 -m pip install --user virtualenv\npython3 -m venv .venv\nsource .venv/bin/activate\n```\n
```

## Try it (Anaconda/conda):

```
```\nconda create --name myenv\nconda activate myenv\n```\n
```

# Libraries, and virtual environments

Installing your dependencies

```
```
```

```
conda install --force-reinstall -y --name myenv -c conda-forge --file  
requirements.txt
```

```
```
```

```
```
```

```
pip install -r requirements.txt
```

```
```
```



# Project 3: CSV parsing with pandas

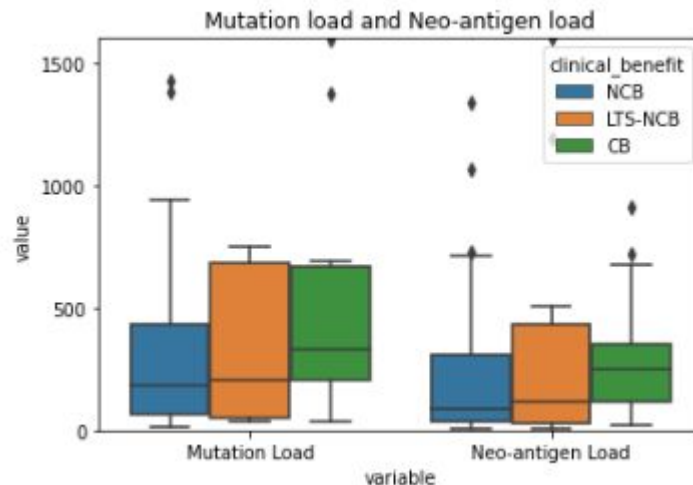
**Do the following with way less code than before**

1. Use the pandas library to load the CSV
2. Prepare the `clinical_benefit` derivative column
3. Calculate the summary stats as before
4. Make a plot like Figure 1A

# Project 3: CSV parsing with pandas

## Expected output

		value
clinical_benefit	variable	
CB	Mutation Load	737.153846
	Neo-antigen Load	467.692308
LTS-NCB	Mutation Load	571.500000
	Neo-antigen Load	383.000000
NCB	Mutation Load	396.810811
	Neo-antigen Load	315.986486



# Project 4: Jupyter Notebook

## Tasks:

- Install jupyter lab
- Modify notebook to point to the right file
- Run `project4_jupyter.ipynb` to generate the plots and stats from project 3.

# Library

- <https://docs.python-guide.org/writing/structure/>
- <https://stackoverflow.com/>
- <https://realpython.com/python-main-function/>
- <https://google.com>
- <https://chat.openai.com/chat>