



First Steps with Python

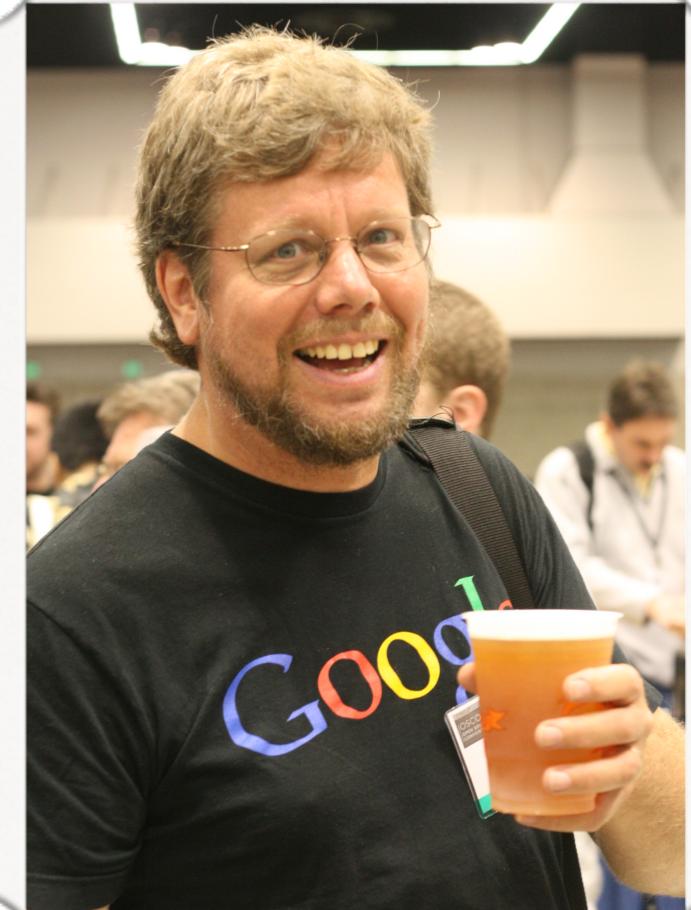
Mon 28th October 2019 5pm - 6:30pm



WHAT IS PYTHON?

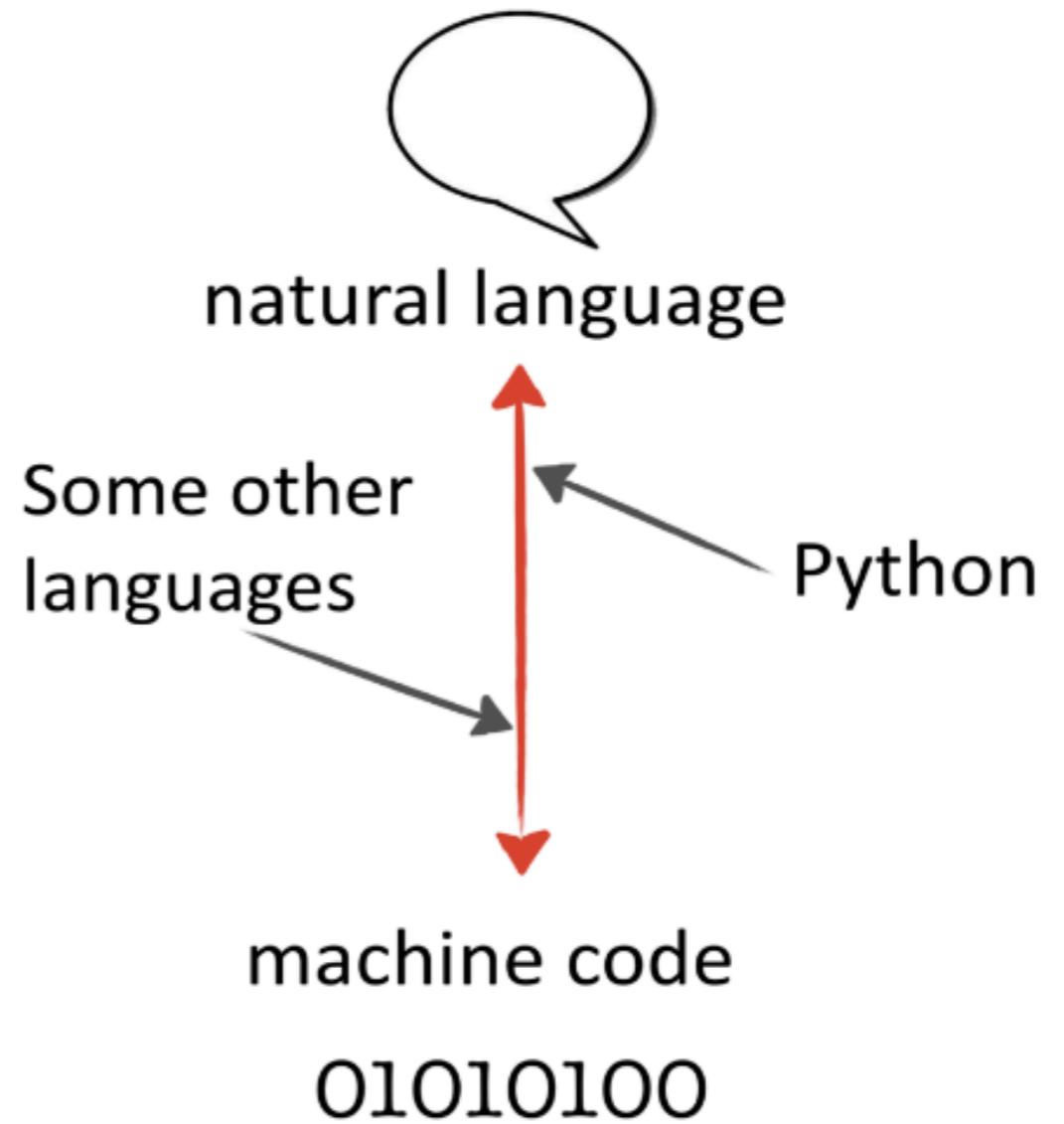
What is Python?

- ▶ Created by Guido Van Rossum in 1991
- ▶ Emphasises **productivity** and **code readability**
- ▶ The language is **easy to pick up** and learn
- ▶ **Readable code** means that almost anyone can pick up a piece of code and understand what it is doing



Why Python?

- ▶ Human readable
- ▶ High level & general
- ▶ Cross-platform
- ▶ Free & open source
- ▶ Strong & friendly community



Why Python?

- ▶ **Simple & clean** syntax
- ▶ Extremely fun to develop in :)
- ▶ Basically **everything** can be done with Python: many purposes outside data science
- ▶ Things can be done **quickly!**
 - ▶ For example a program that takes you weeks to write in C++, might take you a day in Python.

Where is Python used?

- ▶ Everywhere!
- ▶ Industry
- ▶ Academia & research
- ▶ Web Development / Web Apps
- ▶ Art & music
- ▶ Game Development
- ▶ Windows Applications

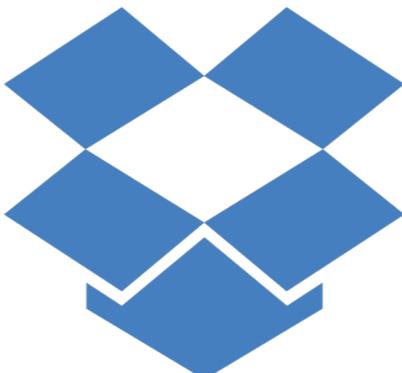
Powered by Python



Instagram



Quora



Dropbox



Spotify®



YouTube

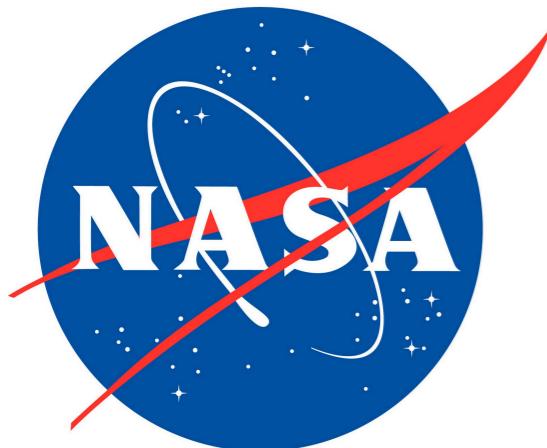
Powered by Python



Gartner®

Google

Honeywell



BNP PARIBAS



DISQUS



TOYOTA



The Washington Post

Eventbrite®

the ONION®

splunk®



LAB:

TOOLS OF THE TRADE

The Python Shell (February 1991)

- ▶ Search your computer for "**Anaconda Prompt**". You are now using the command line interface (CLI)!
- ▶ A python shell is similar to a Command Line Terminal and it can be launched by typing:
 - ▶ **python**

```
(base) ➔ ~ python
Python 3.6.6 | packaged by conda-forge | (default, Jul 26 2018, 09:55:02)
[GCC 4.2.1 Compatible Apple LLVM 6.1.0 (clang-602.0.53)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 1 + 2
3
>>> x = 3
>>> x
3
>>> quit() _
```

IPython Shell (2001)

- ▶ An IPython shell is more interesting than a plain terminal, providing syntax colouring and shortcuts to interact with our code. It can be launched with:
 - ▶ ipython

```
(base) ➔ ~ ipython
Python 3.6.6 | packaged by conda-forge | (default, Jul 26 2018, 09:55:02)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.0.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: 1+1
Out[1]: 2

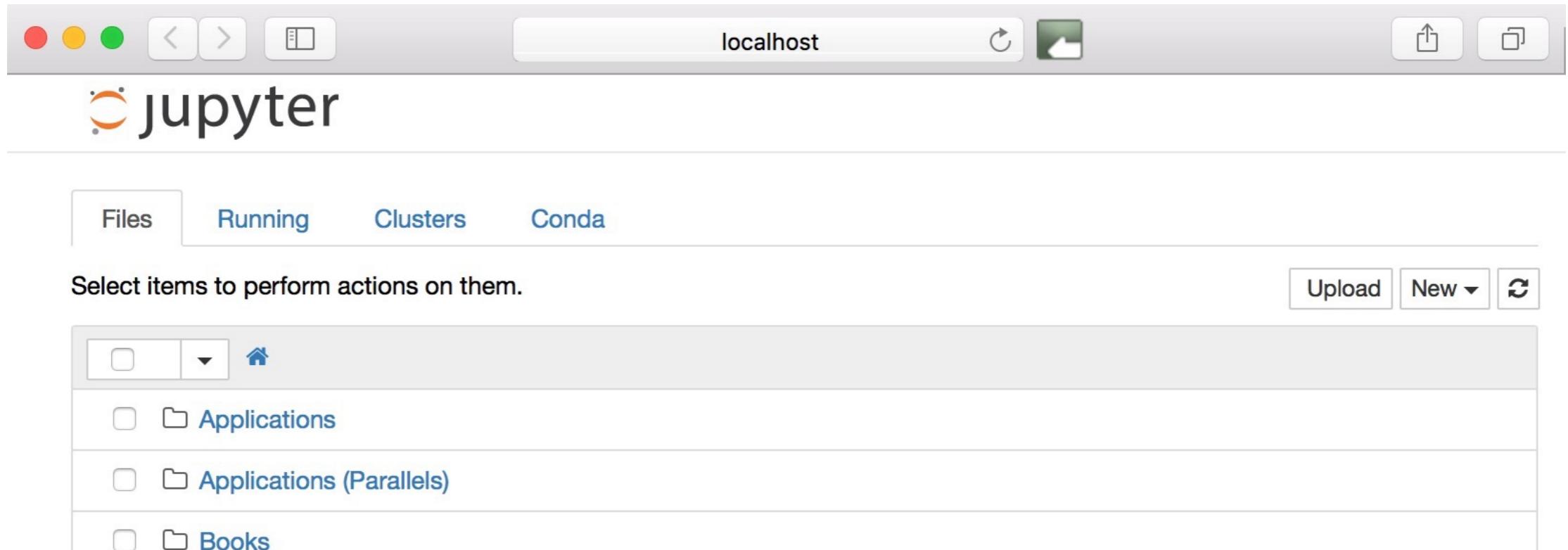
In [2]: x = "Hello"

In [3]: x
Out[3]: 'Hello'

In [4]: quit()
```

Jupyter Notebook (2014)

- ▶ Jupyter notebook is a web interface that let's us use formatting along side our code. It is the extremely common and very useful! You can launch it by typing:
 - ▶ **jupyter notebook**
- ▶ Click **Desktop** to navigate into that folder.
- ▶ On the top-right-hand-side click "New" then "Python 3".





LAB:

JUPYTER NOTEBOOK



LAB:

LET'S WRITE

SOME PYTHON

Python as a calculator

Try each line inside **separate** Jupyter Notebook cells
(type Shift-Enter to run):

```
1 + 1
```

```
50 - 2 * 4
```

```
8 / 5
```

```
3 ** 2
```

"Hello World"

- ▶ Compare this typical "Hello World!" program in Python...

```
print("Hello World")
```

- ▶ ...with the same program in C++

C++

```
1 #include<iostream>
2 using namespace std;
3
4 int main()
5 {
6     cout << "Hello World!";
7     return 0;
8 }
```

Variables

- ▶ Variables stores values under a specified name:

```
cup = "coffee"  
print(cup)
```

```
cup = 5  
print(cup)
```

```
cup = 3.14  
print(cup)
```

Operators

- ▶ Try the following:

```
# This is a comment

# Comparing numbers
2 > 1

# Testing for equality
2 == 2

# "Boolean comparisons"
(8 > 0) and (3 != 3)
(5 >= 5) or (3 != 3)
```

Lists

- ▶ Lists are collections of objects

```
my_list = ["a", "b", "c", "d"]
```

```
type(my_list)
```

```
my_list
```

```
my_list[1:3]
```

- ▶ Python starts counting from zero!

```
my_list[0]
```

Lists

- ▶ Lists can be changed

```
my_list
```

```
my_list[3] = "changed!"
```

```
my_list
```

Dictionaries

- ▶ Dictionaries combine keys with values in pairs
- ▶ Like in a dictionary, you can search the keys to obtain their corresponding value

```
my_dict = {'x': 'hello', 'y': 10}

my_dict

my_dict['x']

my_dict['z'] = [1, 2, 3]

my_dict

my_dict['z'][0]
```

Other data structures

- ▶ "Data structures" are a particular way of storing and organising data in a computer so that it can be used efficiently. Some examples include:
 - ▶ Tuples
 - ▶ Arrays
 - ▶ Matrices
 - ▶ Dataframes
- ▶ We will see some examples of these later.



PYTHON PROGRAMMING FUNDAMENTALS

Control structures

- ▶ **Control structures:** A block of programming that analyses variables and chooses a direction in which to go based on given parameters.
- ▶ The term flow control details the direction the program takes (which way program control “flows”). It determines how a computer will respond when given certain conditions and parameters. Some typical structures include:
 - ▶ **If statement**
 - ▶ **For loop**
 - ▶ **Functions**

if

- ▶ An **if** statement is a conditional structure that, if proved true, performs a function or displays information.
- ▶ Think of this as a decision that moves the flow of your program depending on the answer to a TRUE-FALSE question.

- ▶ In pseudocode:

```
1 | IF a person is older than 18  
2 | THEN they can drive  
3 | ELSE they cannot drive
```

- ▶ In Python:

```
if age_person > 18:  
    print("They can drive")  
else:  
    print("They cannot drive")
```

if

► Try this:

```
A = 10  
B = 100  
if A > B:  
    print("A is bigger")  
elif A == B:  
    print("same")  
else:  
    print("B is bigger")
```

if

► Try this:

```
A = 10
B = 100
if A > B:
    print("A is bigger")
elif A == B:
    print("same")
else:
    print("B is bigger")
```

White space is
very important in
Python!

You can indent
this with four
spaces, or a tab
(some text editors
convert tabs to
four spaces)

if

► Try this:

```
A = 10
B = 100
if A > B:
    print("A is bigger")
elif A == B:
    print("same")
else:
    print("B is bigger")
```

Don't forget the
colon (:)

for loop

- ▶ A loop statement in programming performs a predefined set of instructions or tasks while or until a predetermined condition is met.
- ▶ Think of this as a repetitive action that has to be performed until further notice.

- ▶ In pseudocode:

```
1 | FOR each user of a service in a list  
2 |   PRINT greet them
```

- ▶ In Python:

```
people = ["Boris", "Jeremy", "Jo"]  
  
for person in people:  
    print("Hello " + person)
```

for loop

► Try this:

```
for x in [1, 2, 3]:  
    print(x)
```

```
for x in range(10):  
    print(x**2)
```

Functions

- ▶ A function is a group of instructions used by programming languages to return a single result or a set of results.
- ▶ Functions are a convenient way to divide our code into useful blocks, providing us with order, making the code more readable and reusable.
- ▶ Here is how you define a function in Python:

Python

```
1 | def function_name(input1, input2...):  
2 |     1st block of instructions  
3 |     2nd block of instructions  
4 |     ...
```

Functions

```
def square(x):  
    return x**2  
  
def add(x, y):  
    return x + y  
  
output = square(7)  
print(output)  
print(add(output, 3))
```



LAB:

WRITE A PYTHON PROGRAM

Activity: Write a Python program

- Write a Python script to solve one (or all!) of the following tasks:
- 1. Create a function that will calculate the area of a circle with radius r.**
 - 2. For a given number, calculate and print the square of a number. If the given number is larger than 10 also calculate the cube.**
 - 3. List the letters in the sentence "Python is fun".**

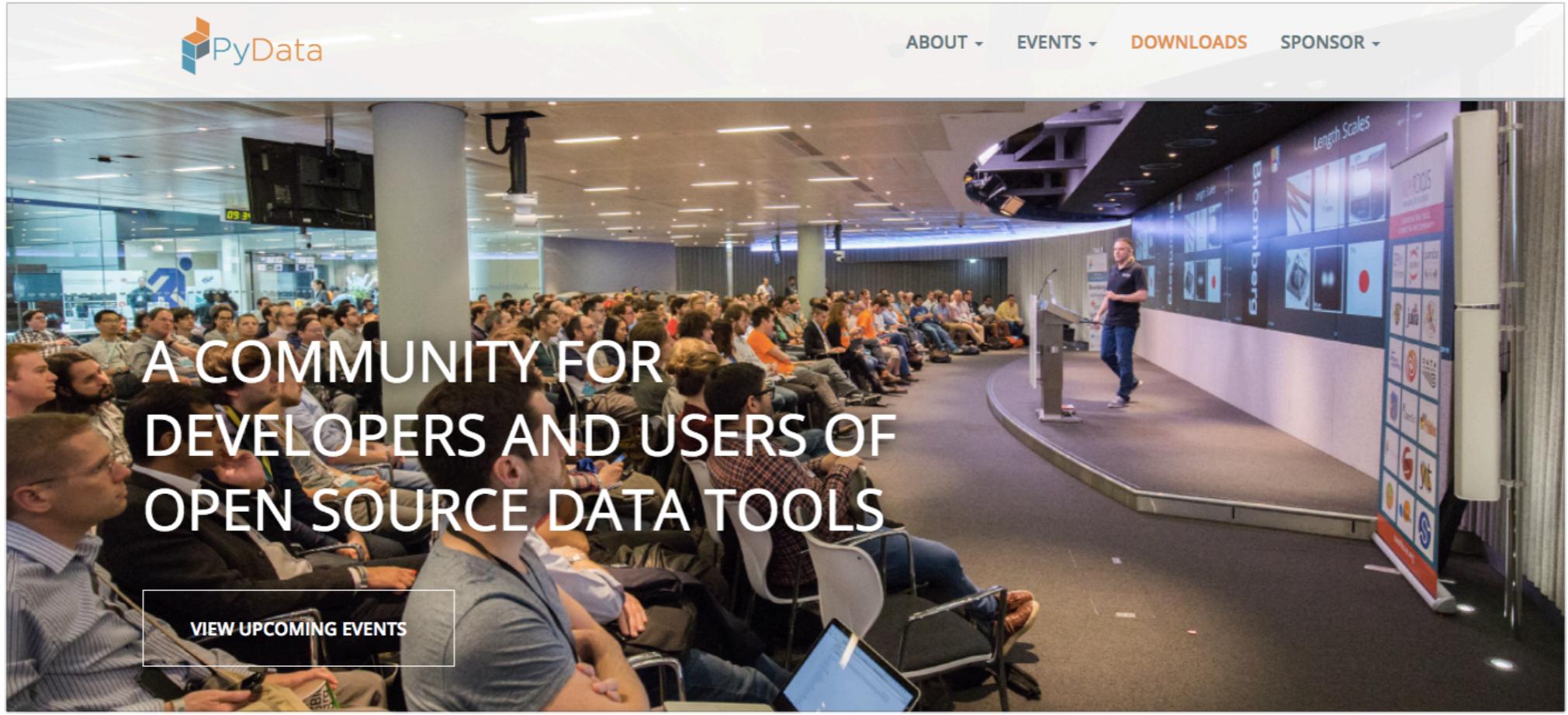


LAB:

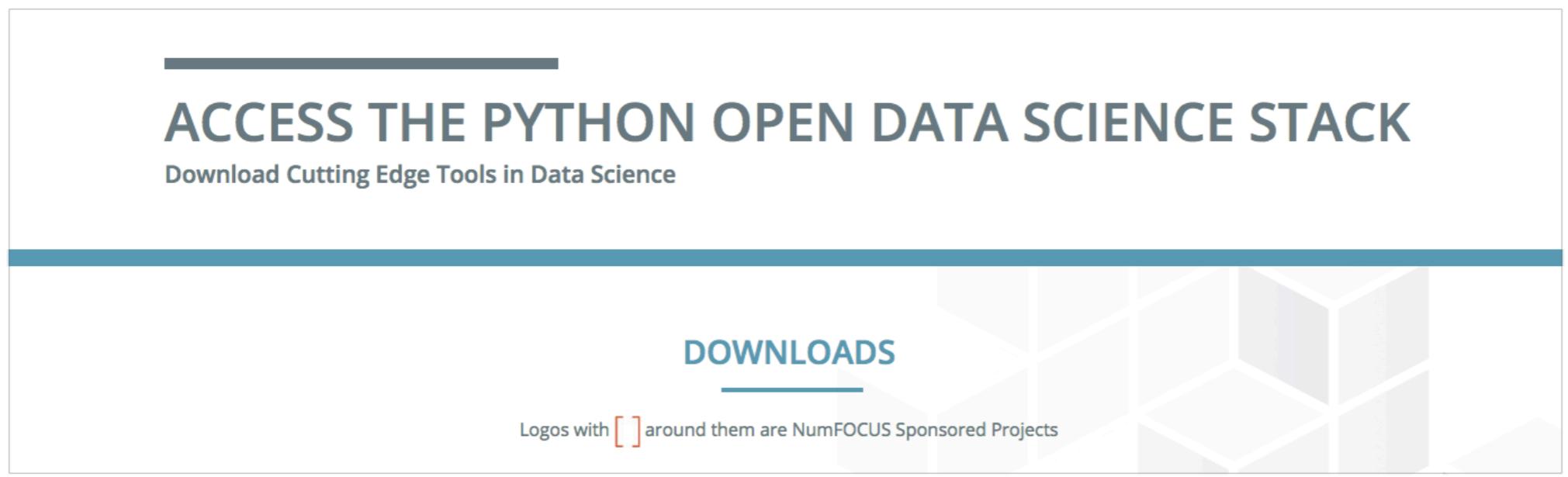
PYTHON

FUNDAMENTALS

The PyData Ecosystem



The image shows the homepage of the PyData website. At the top left is the PyData logo. At the top right are navigation links: ABOUT, EVENTS, DOWNLOADS (which is highlighted in orange), and SPONSOR. Below the header is a large photograph of a conference room. A man in a dark t-shirt is standing at a podium on a stage, speaking to a large audience seated in rows of chairs. To the right of the stage, there's a wall with various logos and images related to data science. Overlaid on the left side of the photo is white text: "A COMMUNITY FOR DEVELOPERS AND USERS OF OPEN SOURCE DATA TOOLS". At the bottom left of this overlay is a button labeled "VIEW UPCOMING EVENTS".



The image shows the "DOWNLOADS" section of the PyData website. At the top, there's a horizontal bar with the text "ACCESS THE PYTHON OPEN DATA SCIENCE STACK" and a subtext "Download Cutting Edge Tools in Data Science". Below this is a blue decorative bar. Further down, there's a section titled "DOWNLOADS" with a subtext: "Logos with [] around them are NumFOCUS Sponsored Projects". To the right of the text is a graphic of three interlocking 3D cubes.

The PyData Ecosystem



FENICS'18
MARCH 21-23, 2018
Oxford, UK



PYDATA KAUNAS @ PYCON LITHUANIA
MAY 5, 2018



PYDATA EDINBURGH @ EUROPYTHON
JULY 25-29, 2018



PYDATA LOS ANGELES
OCTOBER 22-24, 2018



PYDATA FLORENCE @ PYCON ITALY
APRIL 20-22, 2018



ROPENSCI UNCONF
MAY 21-22, 2018
Seattle, WA, USA



JULIACON
AUGUST 7-11, 2018
London, UK



PYDATA KARLSRUHE & PYCON DE
OCTOBER 24-28, 2018



PYDATA LONDON
APRIL 27-29, 2018



PYDATA AMSTERDAM
MAY 25-27, 2018



JUPYTERCON
AUGUST 21-24, 2018
New York, NY, USA



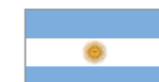
PYDATA WARSAW
NOVEMBER 19-20, 2018



PYTHON IN ASTRONOMY
APRIL 30 - MAY 4, 2018
New York, NY, USA



PYDATA BERLIN
JULY 6-8, 2018



PYDATA CÓRDOBA
OCTOBER 1-2, 2018



PYDATA NYC
NOVEMBER 2018



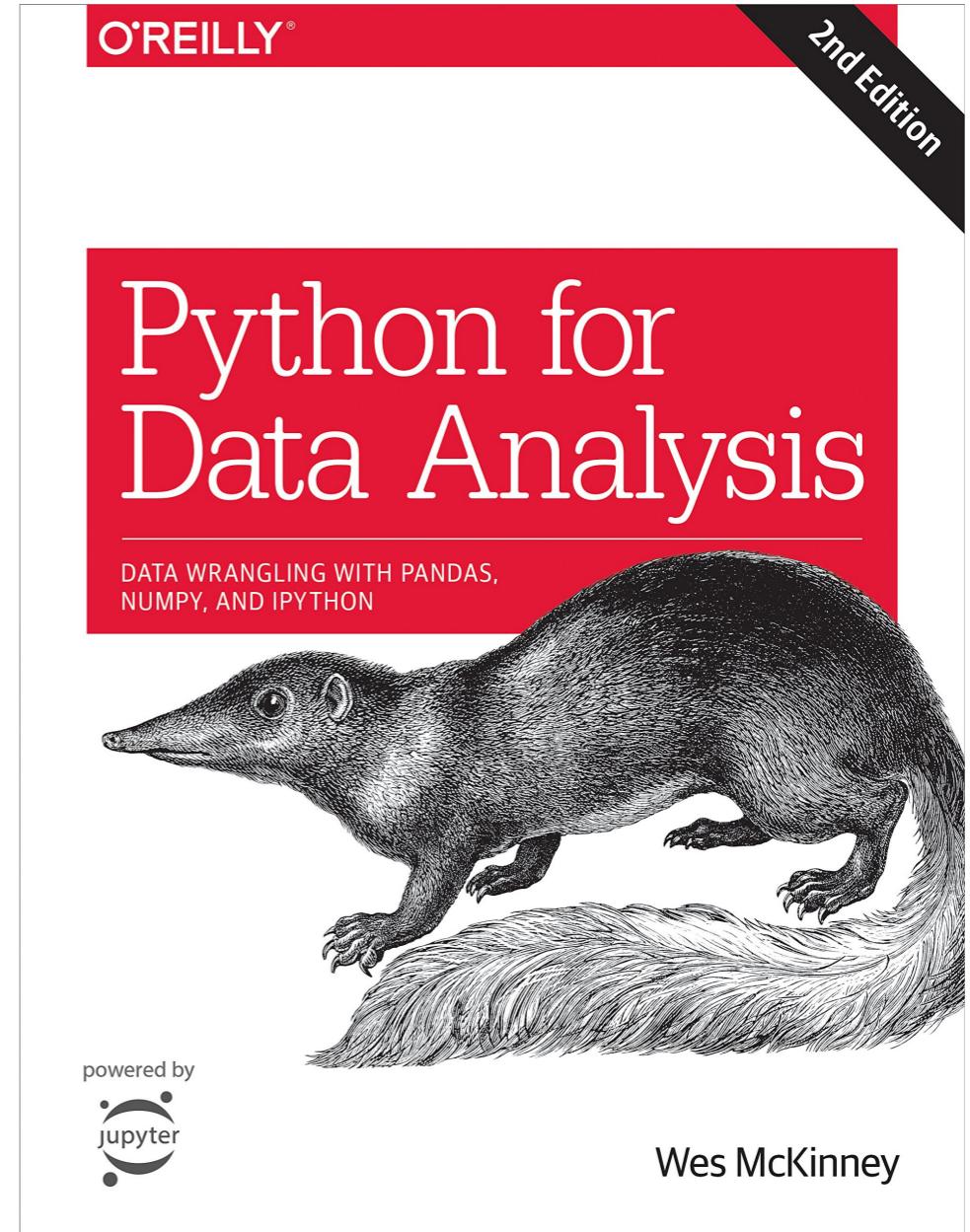
Learn More!

Data Skeptic Podcast

The screenshot shows a podcast player interface. On the left is a dark sidebar with the 'DATA SKEPTIC' logo in yellow. The main area has a white background. At the top, it says 'MARCH 10, 2017' and ' [MINI] The Perceptron'. Below that is a button with '▶ PLAY 14:46' and a 'Download' button with a download icon. The episode summary text reads: 'Today's episode overviews the perceptron algorithm. It is characterized by a few particular features. It updates every example, rather than as a batch. It uses a step function. It's only appropriate for linearly separable data. It finds a solution if the data meets these criteria. Being a fairly simple algorithm, it can be implemented very efficiently. Although we don't discuss it in this episode, the backpropagation algorithm and perceptron networks are what makes this technique work.'

dataskeptic.com

kaggle.com/learn



THANK YOU

@john_sandall

