Python Training

at Python Predictions

Hugo Herter, 2017

hello@hugoherter.com

Program

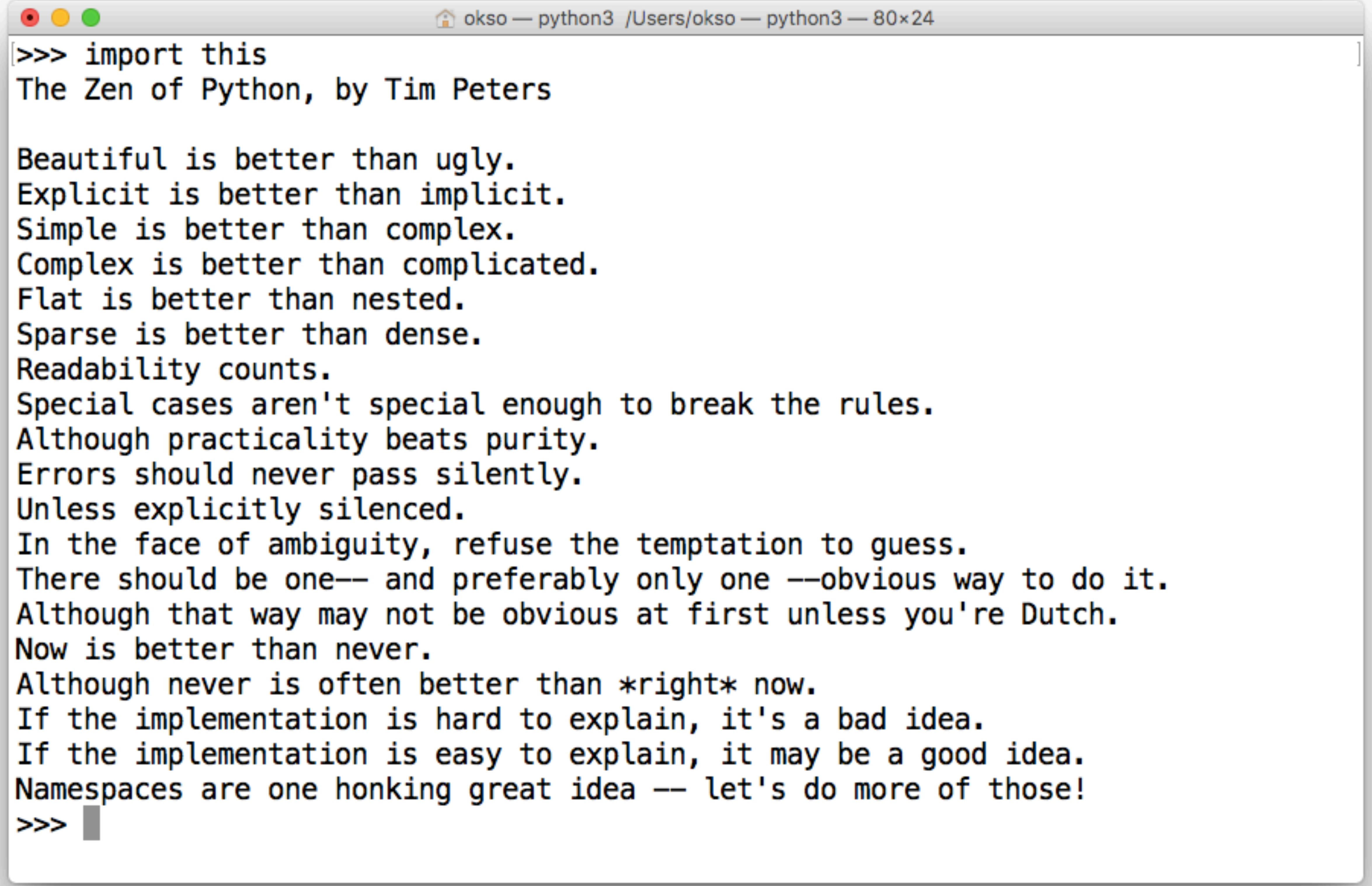
Fundamentals of Python

- Start and run python programs interactively with python CLI
- Use an IDE to write programs and execute them, including command line arguments
- Create notebooks locally and on a server
- Import libraries
- Store data in variables and understand their reach
- Know the standard operators
- Control the flow of a program
- Perform common string operations such as concatenation, substring, replace
- Use the correct data structures
- Use functions to structure your program

Statistical and Machine Learning Packages

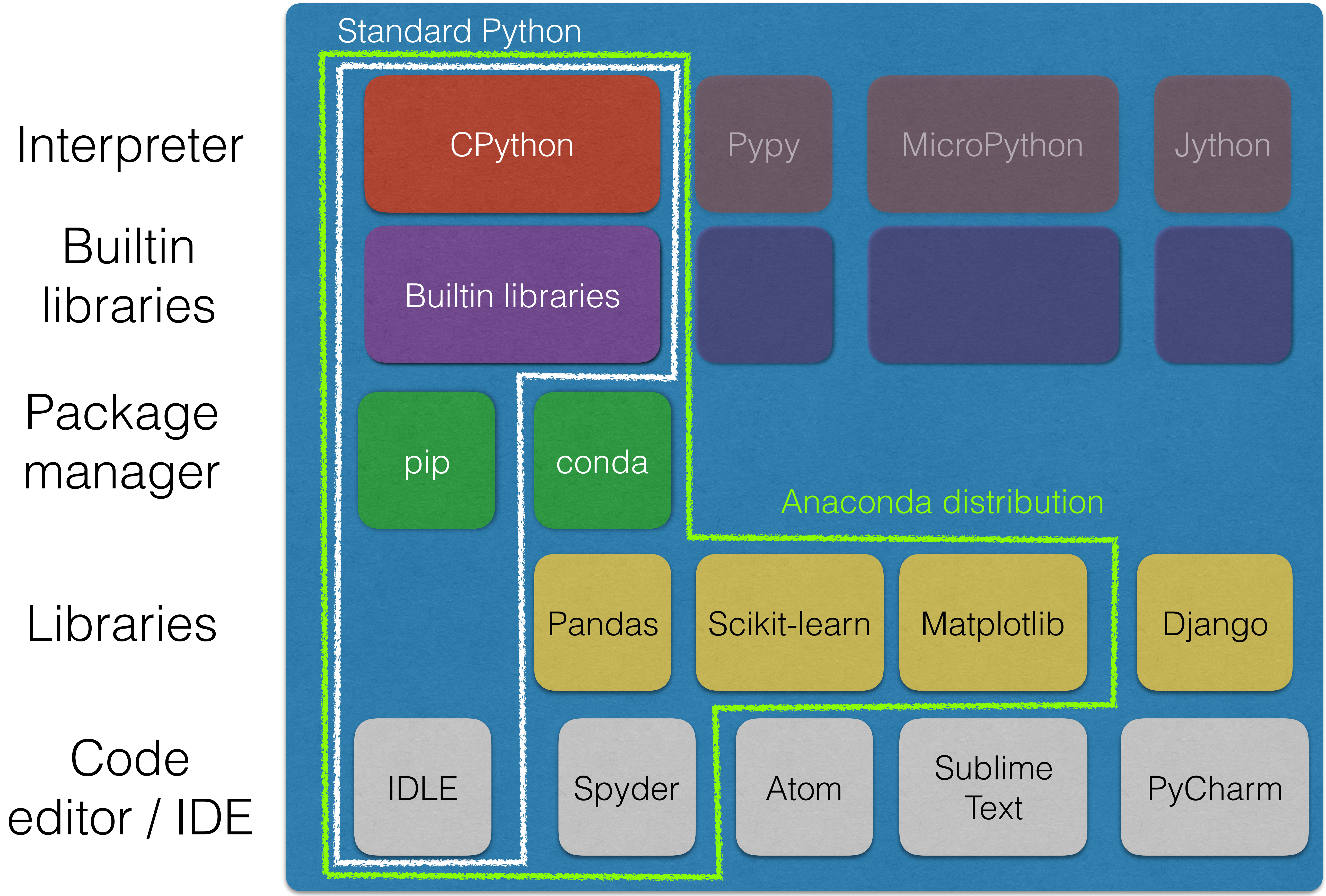
- Import and export data in csv, with dates and special formats
- Use numpy/scipy to perform mathematical computations Please focus on statistics here. Mathematics (integrals, optimization is less relevant for us)
- Slice and dice data
- Use pandas to wrangle data
- Plot data and perform exploratory analysis
- Use scikit-learn If possible, focus on logistic regression, decision trees (+visualisation)
- Perform regression analysis in Python
- Perform classification analysis in Python

The Zen of Python



```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
>>>
```

Python ecosystem



Python Interpreters

- CPython: default, reference
- Pypy: JIT, high performance but lack of compatibility with many C libraries
- MicroPython: microcontrollers, IoT
- Jython: Java, deprecated

Libraries

- `import myfile`
- `from myfile import variable`
- Can import a <u>module</u> or a <u>package</u>

Module

- **File** containing Python definitions and statements
- Usually .py extension (when bytecode .pyc or .pyo)
- Every script is a module if you can import it

https://docs.python.org/3/tutorial/modules.html

Package

- *Packages are a way of structuring Python’s module namespace by using “dotted module names”.*

- **Directory** containing at least a file `__init__.py`

- Can be executed if contains `__main__.py`

- Can be put in a zip with .pyz extension

https://docs.python.org/3/tutorial/modules.html

Package

```
sound/                          Top-level package
      __init__.py               Initialize the sound package
      formats/                  Subpackage for file format conversions
              __init__.py
              wavread.py
              wavwrite.py
              aiffread.py
              ...
      effects/                  Subpackage for sound effects
              __init__.py
              echo.py
              surround.py
              reverse.py
              ...
      filters/                  Subpackage for filters
              __init__.py
              equalizer.py
              vocoder.py
              karaoke.py
              ...
```

Tips with libraries

- Don’t name your files after the name of libraries you will use

- Don’t forget the __init__.py file

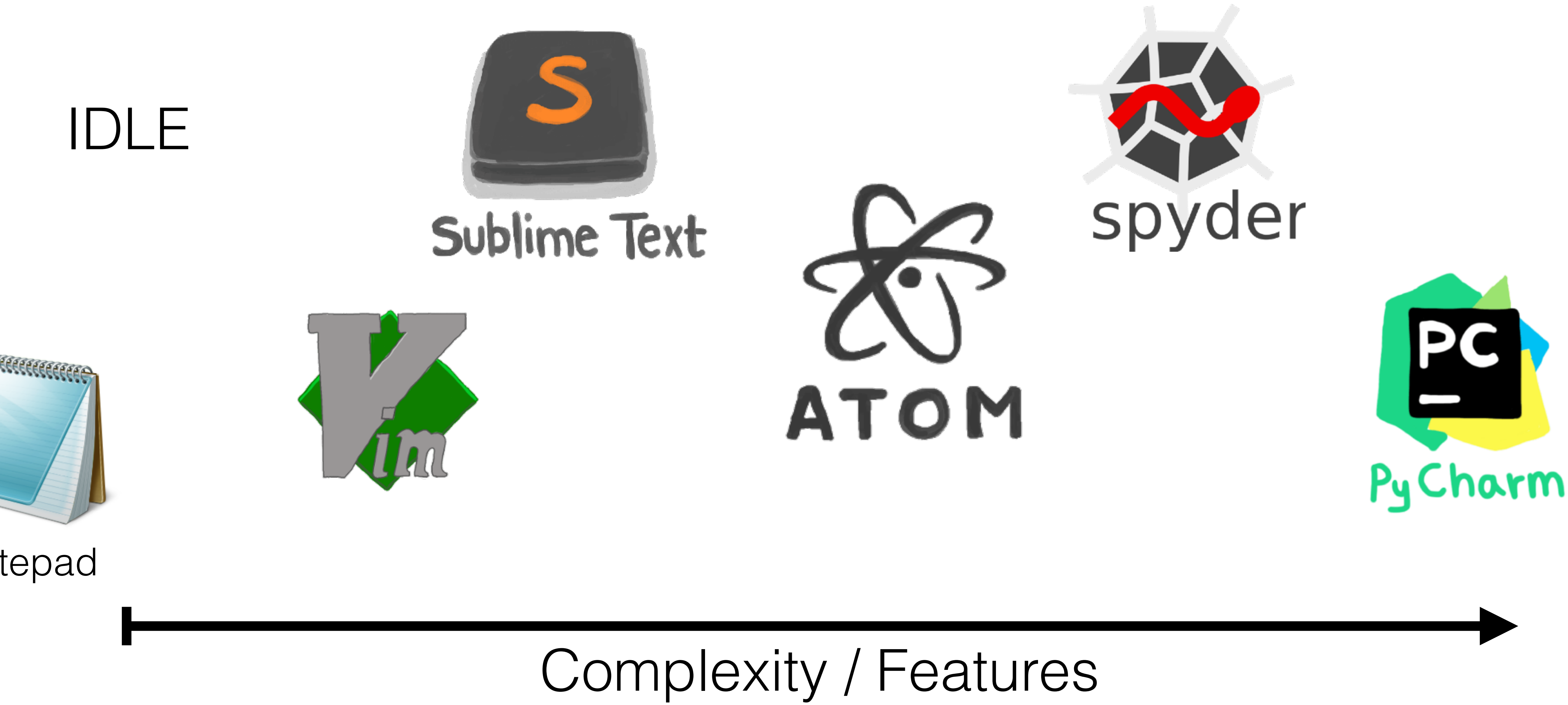
- Python2: Beware if you delete a .py file but not the .pyc

Package managers

- **pip:** download and install new libraries
- **conda:** download and install libraries within Anaconda
- **virtualenv:** create isolated *environment* (collection of libraries & specific version of Python)

https://virtualenv.pypa.io/en/stable/

Code Editors / IDE

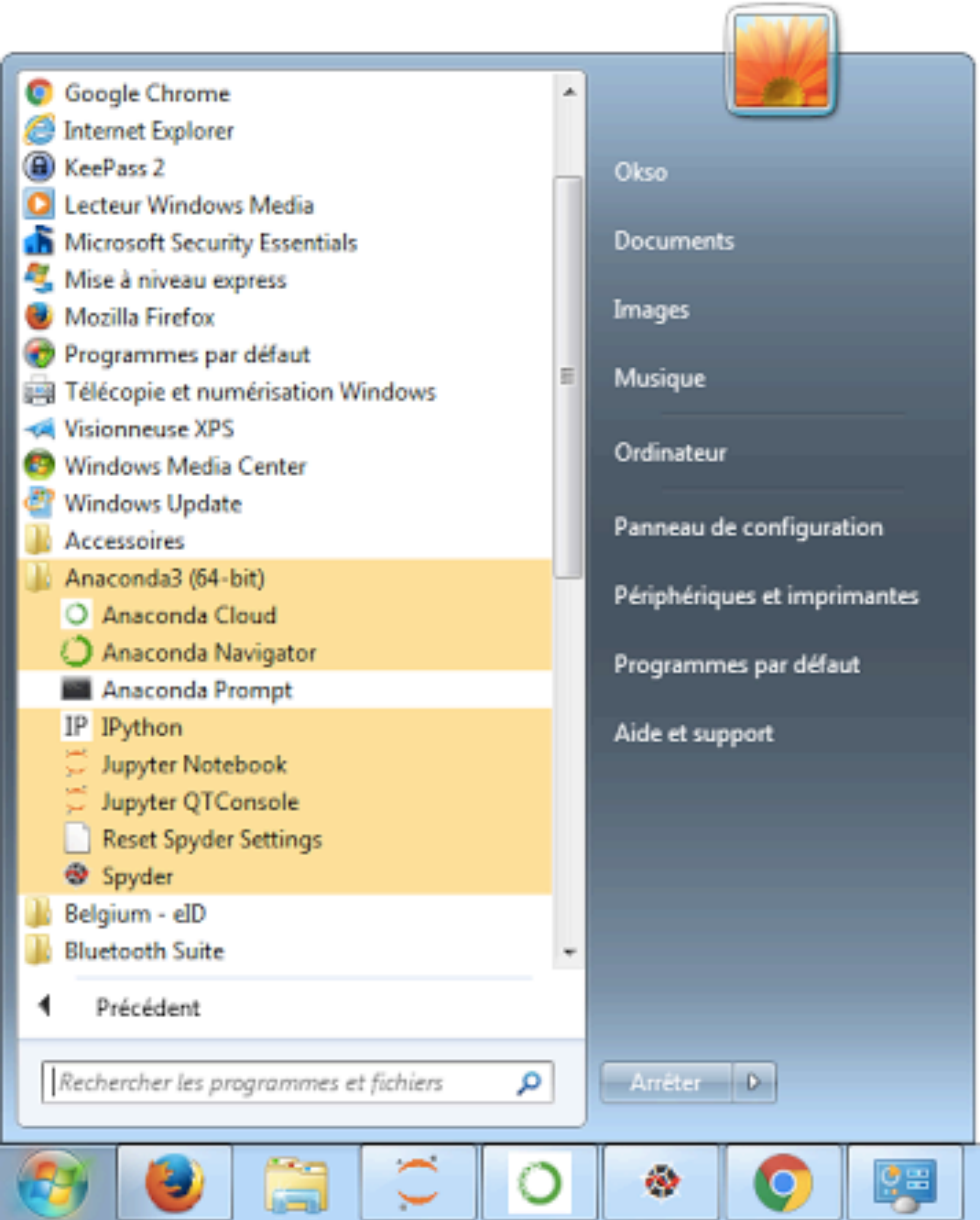


IDLE

tepad

Complexity / Features

Let’s get started

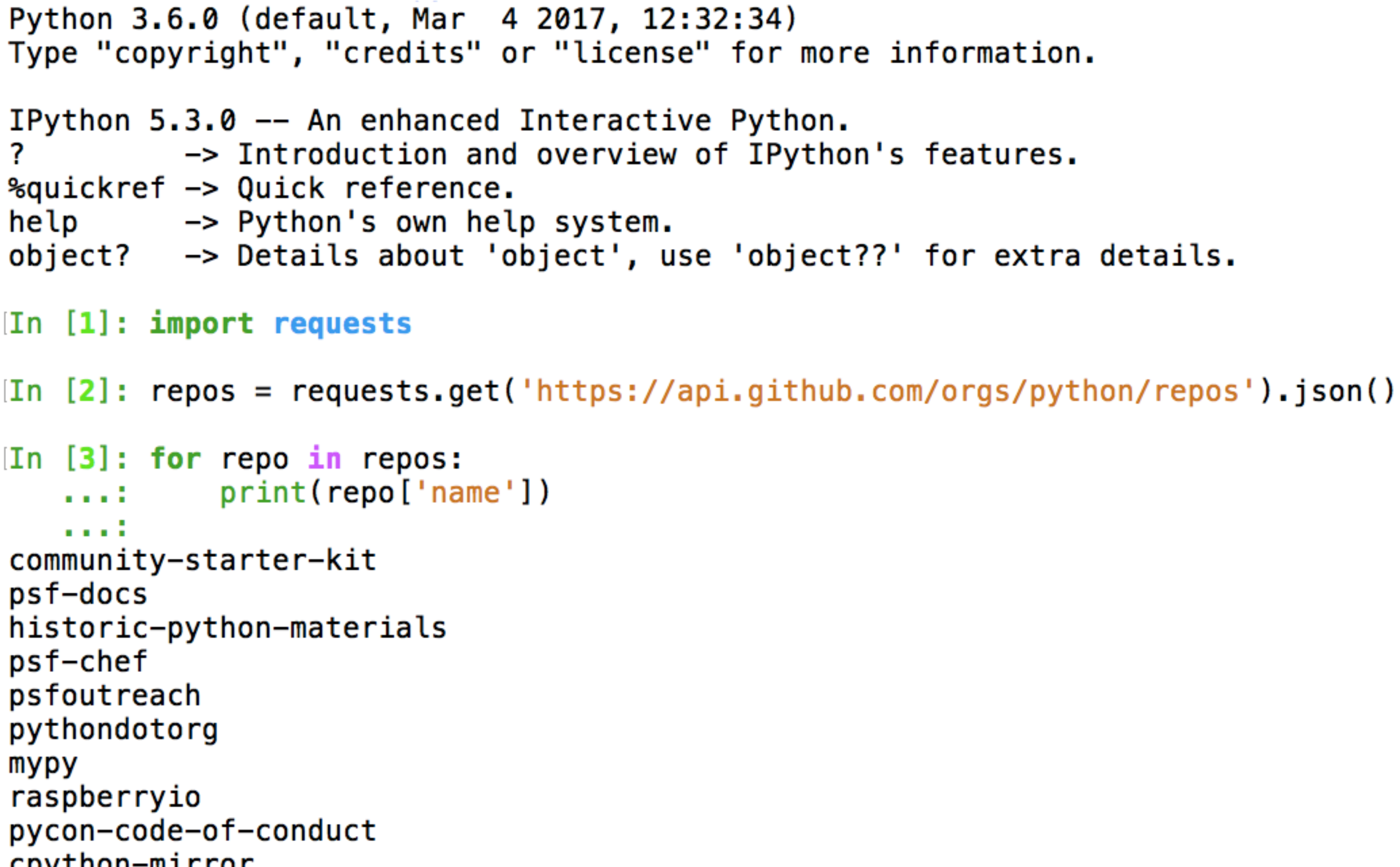


1. Anaconda Prompt
2. IPython
3. Spyder

https://docs.python.org/3/tutorial/

https://learnpythonthehardway.org/python3/

Example using IPython

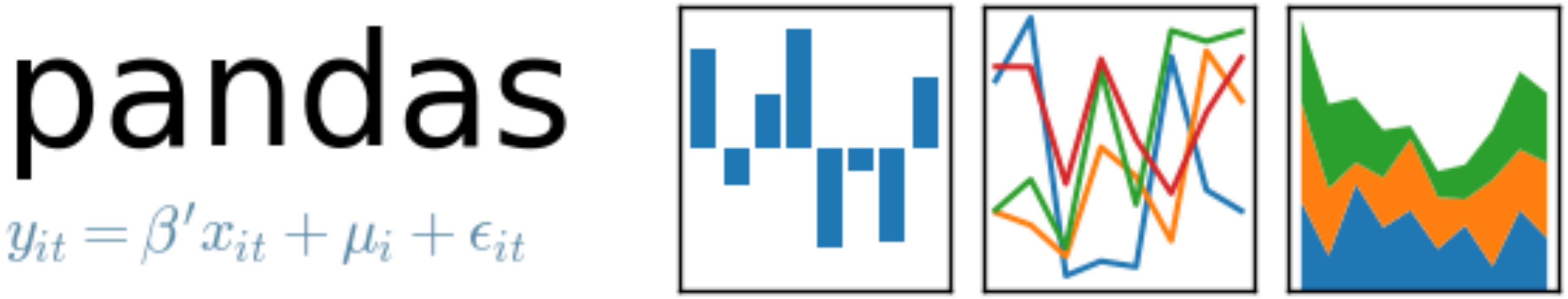


```
Python 3.6.0 (default, Mar  4 2017, 12:32:34)
Type "copyright", "credits" or "license" for more information.

IPython 5.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]: import requests

In [2]: repos = requests.get('https://api.github.com/orgs/python/repos').json()

In [3]: for repo in repos:
   ...:     print(repo['name'])
   ...:
community-starter-kit
psf-docs
historic-python-materials
psf-chef
psfoutreach
pythondotorg
mypy
raspberryio
pycon-code-of-conduct
cpython-mirror
```

Jupyter Notebook

NumPy

- Native Python types are high level
- Provides fast arrays for numerical data
  - Multi-dimensions -> including images
- Provides functions to manipulate that data

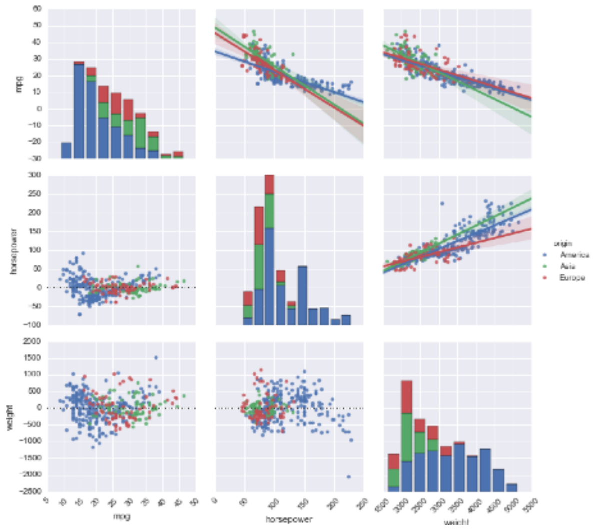


- **goal:** high-performance, easy-to-use data structures and data analysis tools
- Similar to R DataFrames
- Built on top of Numpy

CSV import

- Goal: reproduce the results from this notebook to read a CSV file with Pandas
- http://goo.gl/peFDZ2
  - http://nbviewer.jupyter.org/github/jvns/pandas-cookbook/blob/master/cookbook/Chapter%201%20-%20Reading%20from%20a%20CSV.ipynb

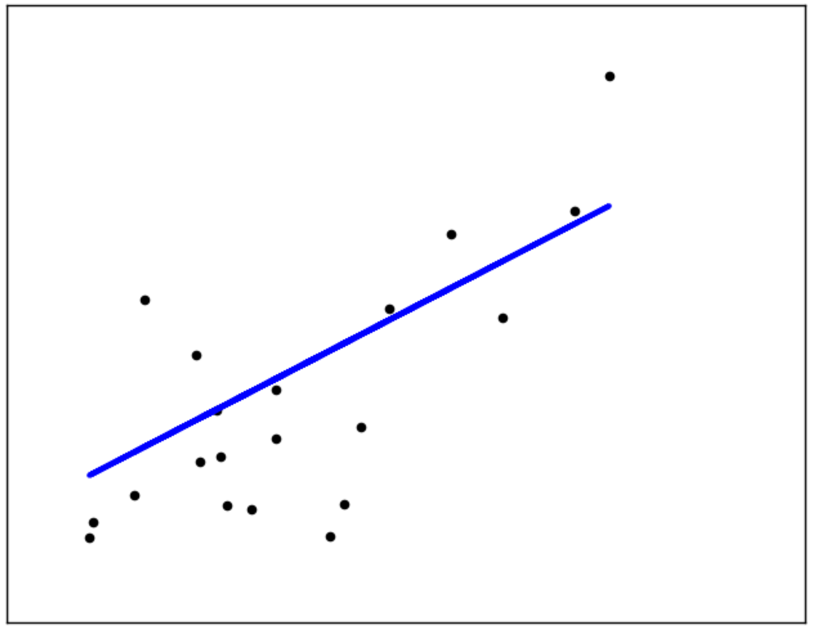
Visualising data



- Matplotlib: the old standard
- Seaborn: relooking on top of Matplotlib
- Plotly, Bokeh: advanced interactive graphs (using Javascript)

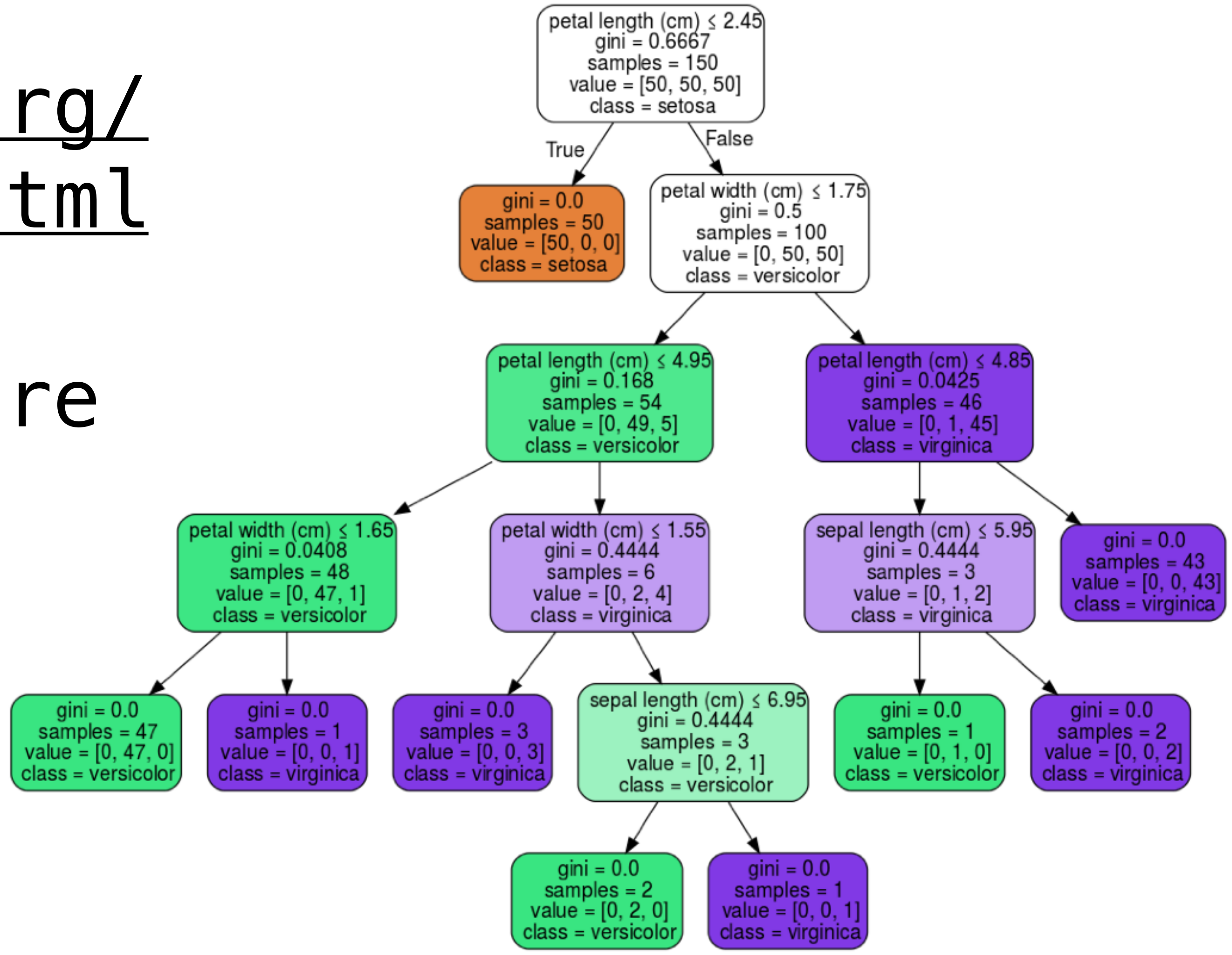
Scikit-learn: linear regression

- Official documentation follows a *“notebook”* approach
- http://scikit-learn.org/stable/modules/linear_model.html



Scikit-learn: Decision tree

- Official documentation follows a *“notebook”* approach
- http://scikit-learn.org/stable/modules/tree.html
- Note: Requires software *Graphviz for images*



Resources

- https://docs.python.org/3/tutorial/ : Official Python tutorial

- https://learnpythonthehardway.org/book/ : Example-based book to learn the Python language, free online access

- https://github.com/pandas-dev/pandas/blob/master/doc/cheatsheet/Pandas_Cheat_Sheet.pdf : Pandas Cheatsheet

- http://nbviewer.jupyter.org/ : Collection of Notebooks illustrating many use cases

- https://github.com/jupyter/jupyter/wiki/A-gallery-of-interesting-Jupyter-Notebooks : Gallery of interesting Notebooks