Tamkang University
淡江大學

Social Computing and Big Data Analytics
社群運算與大數據分析

Tamkang University

Big Data Analytics with Numpy in Python
(Python Numpy 大數據分析)

1052SCBDA05
MIS MBA (M2226) (8606)
Wed, 8,9, (15:10-17:00) (L206)

Min-Yuh Day
戴敏育
Assistant Professor
專任助理教授
Dept. of Information Management, Tamkang University
淡江大學 資訊管理學系
http://mail. tku.edu.tw/myday/

2017-03-15

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2017/02/15	Course Orientation for Social Computing and Big Data Analytics (社群運算與大數據分析課程介紹)
2	2017/02/22	Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data (資料科學與大數據分析：探索、分析、視覺化與呈現資料)
3	2017/03/01	Fundamental Big Data: MapReduce Paradigm, Hadoop and Spark Ecosystem (大數據基礎：MapReduce典範、Hadoop與Spark生態系統)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
4	2017/03/08	Big Data Processing Platforms with SMACK: Spark, Mesos, Akka, Cassandra and Kafka (大數據處理平台SMACK：Spark, Mesos, Akka, Cassandra, Kafka)
5	2017/03/15	Big Data Analytics with Numpy in Python (Python Numpy 大數據分析)
6	2017/03/22	Finance Big Data Analytics with Pandas in Python (Python Pandas 財務大數據分析)
7	2017/03/29	Text Mining Techniques and Natural Language Processing (文字探勘分析技術與自然語言處理)
8	2017/04/05	Off-campus study (教學行政觀摩日)

課程大綱 (Syllabus)

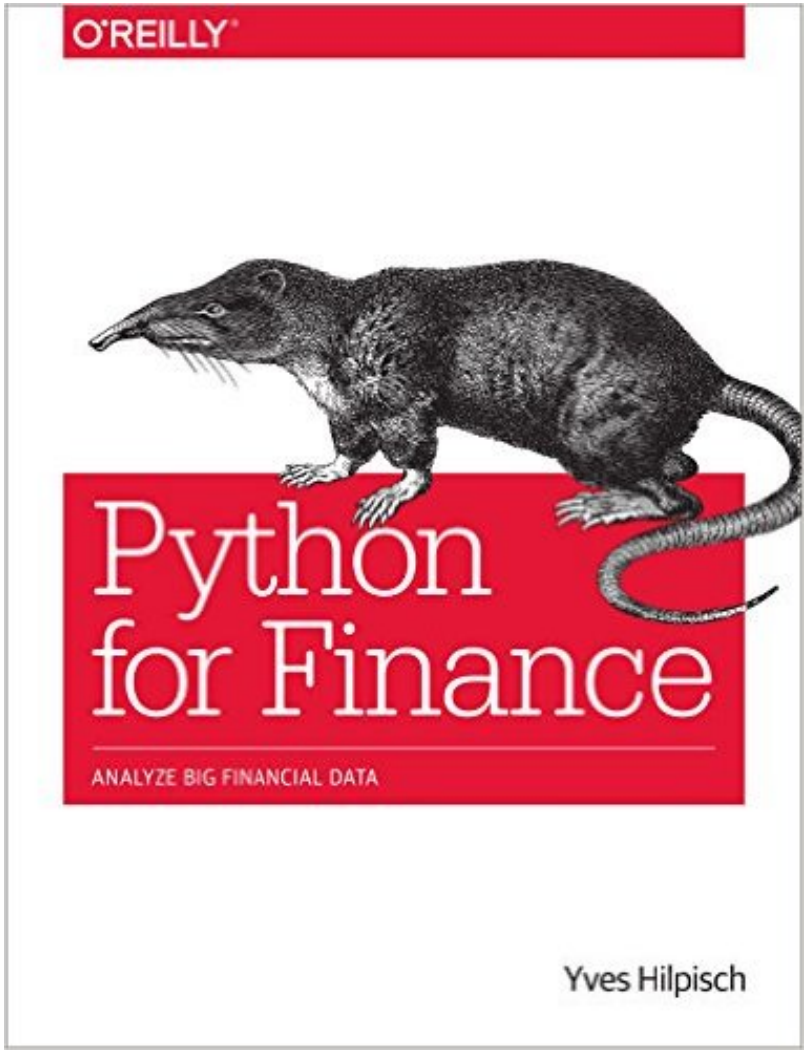
週次 (Week)	日期 (Date)	內容 (Subject/Topics)
9	2017/04/12	Social Media Marketing Analytics (社群媒體行銷分析)
10	2017/04/19	期中報告 (Midterm Project Report)
11	2017/04/26	Deep Learning with Theano and Keras in Python (Python Theano 和 Keras 深度學習)
12	2017/05/03	Deep Learning with Google TensorFlow (Google TensorFlow 深度學習)
13	2017/05/10	Sentiment Analysis on Social Media with Deep Learning (深度學習社群媒體情感分析)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
14	2017/05/17	Social Network Analysis (社會網絡分析)
15	2017/05/24	Measurements of Social Network (社會網絡量測)
16	2017/05/31	Tools of Social Network Analysis (社會網絡分析工具)
17	2017/06/07	Final Project Presentation I (期末報告 I)
18	2017/06/14	Final Project Presentation II (期末報告 II)

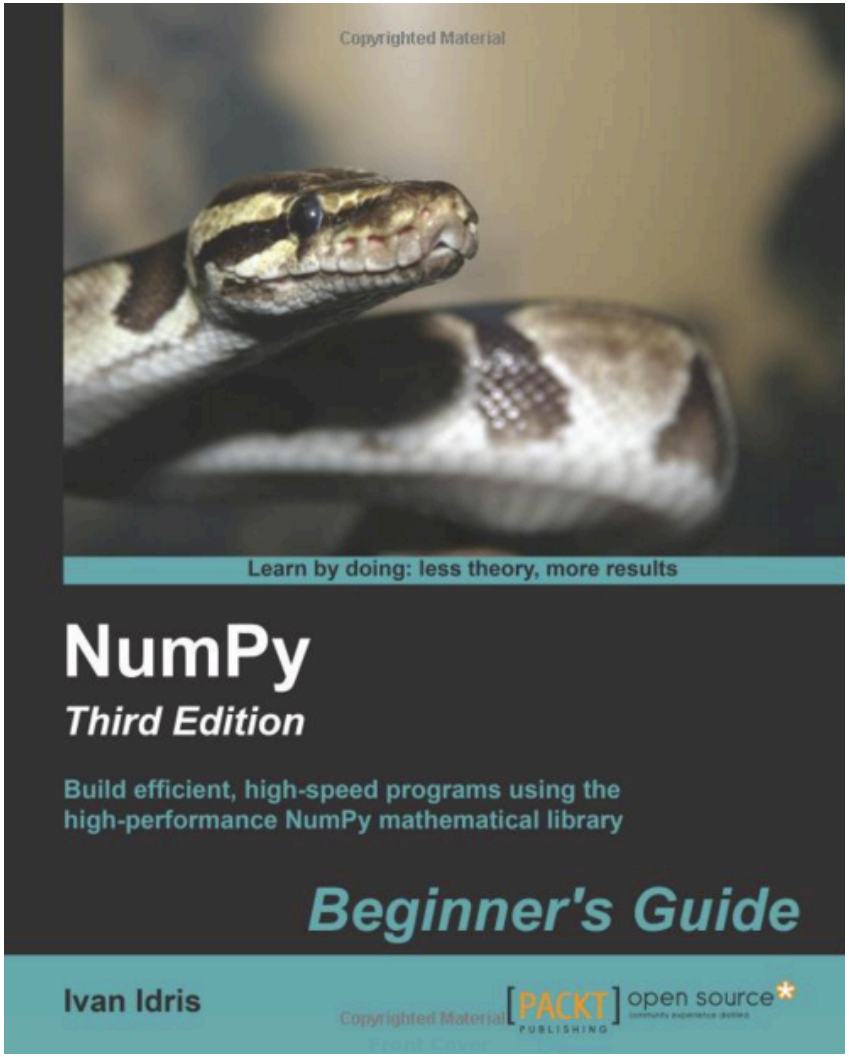
Source: https://www.python.org/community/logos/

Yves Hilpisch, Python for Finance: Analyze Big Financial Data, O'Reilly, 2014



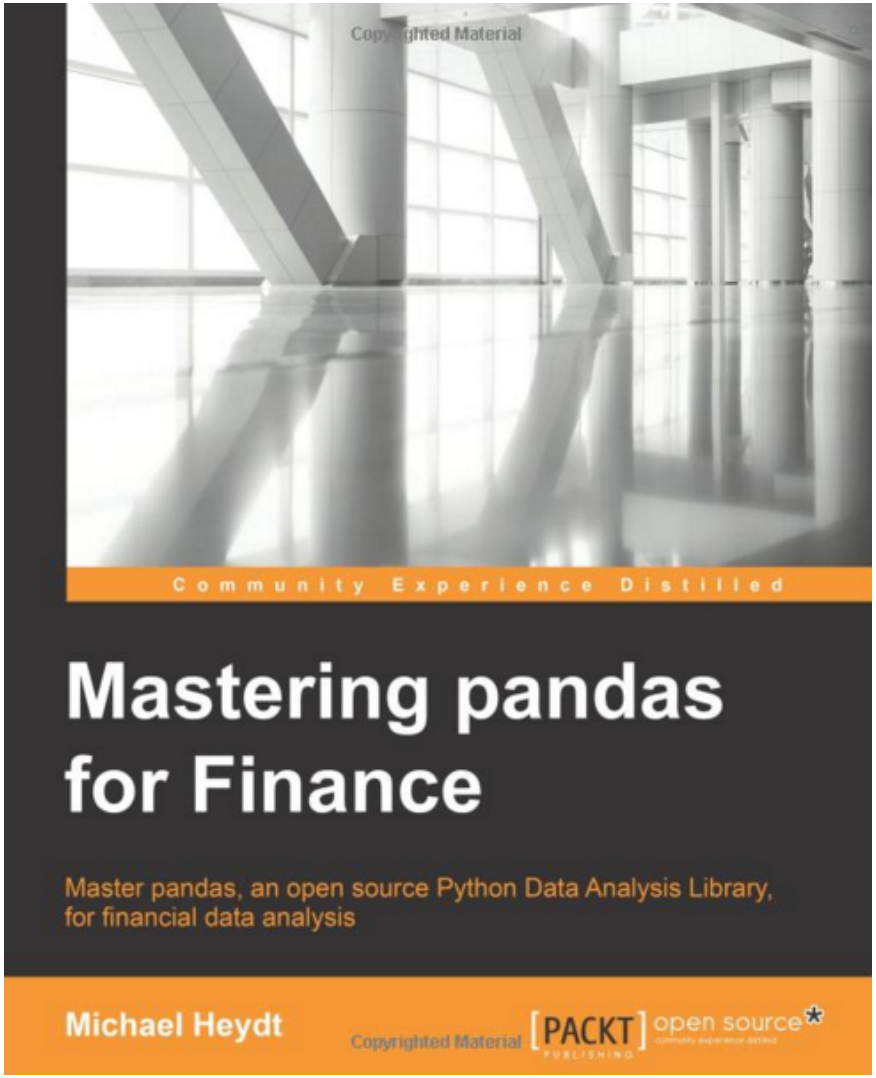
Source: http://www.amazon.com/Python-Finance-Analyze-Financial-Data/dp/1491945281

Ivan Idris,
Numpy Beginner's Guide, Third Edition
Packt Publishing, 2015



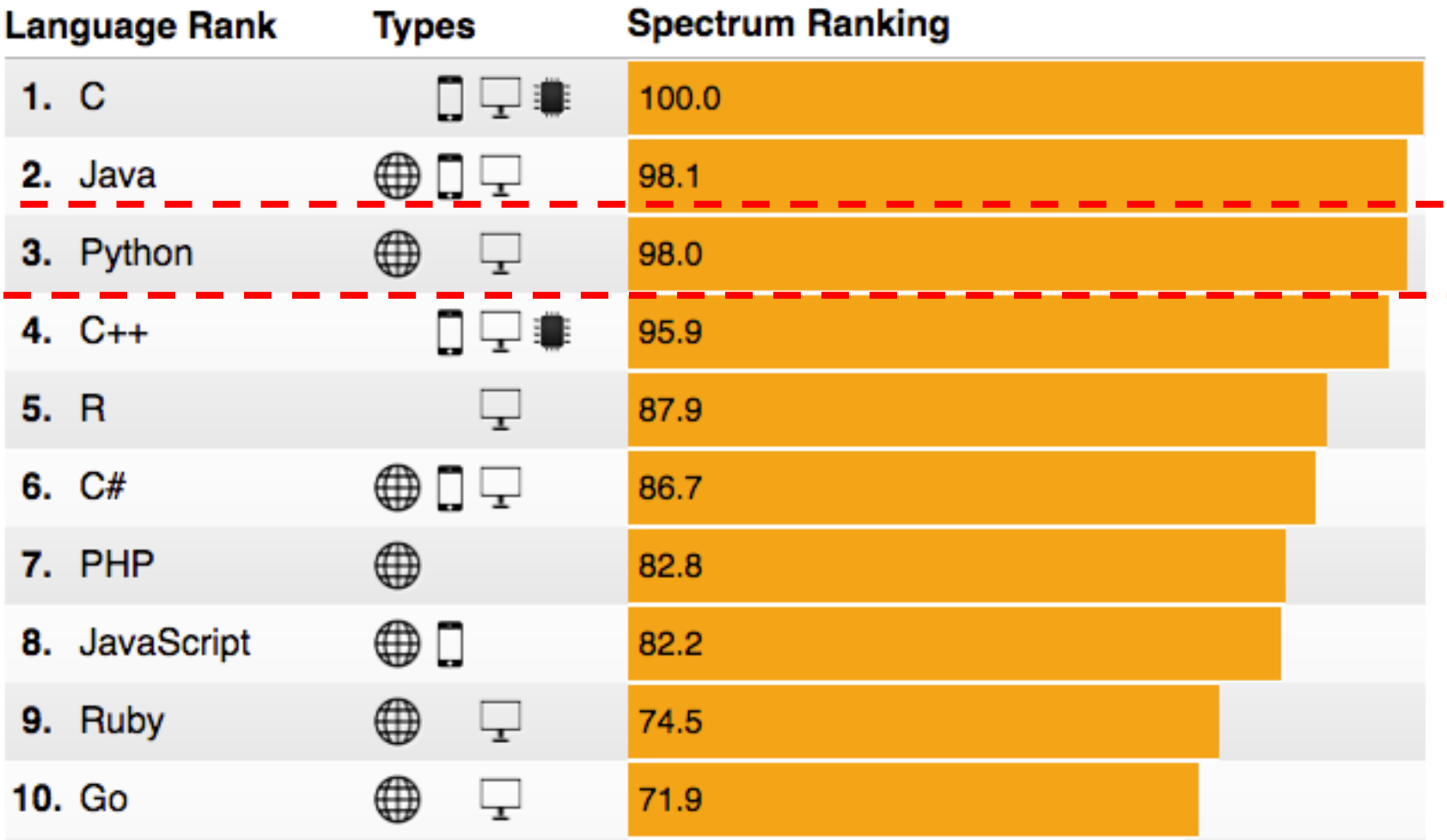
Source: http://www.amazon.com/Numpy-Beginners-Guide-Ivan-Idris/dp/1785281968

Michael Heydt ,
Mastering Pandas for Finance,
Packt Publishing, 2015



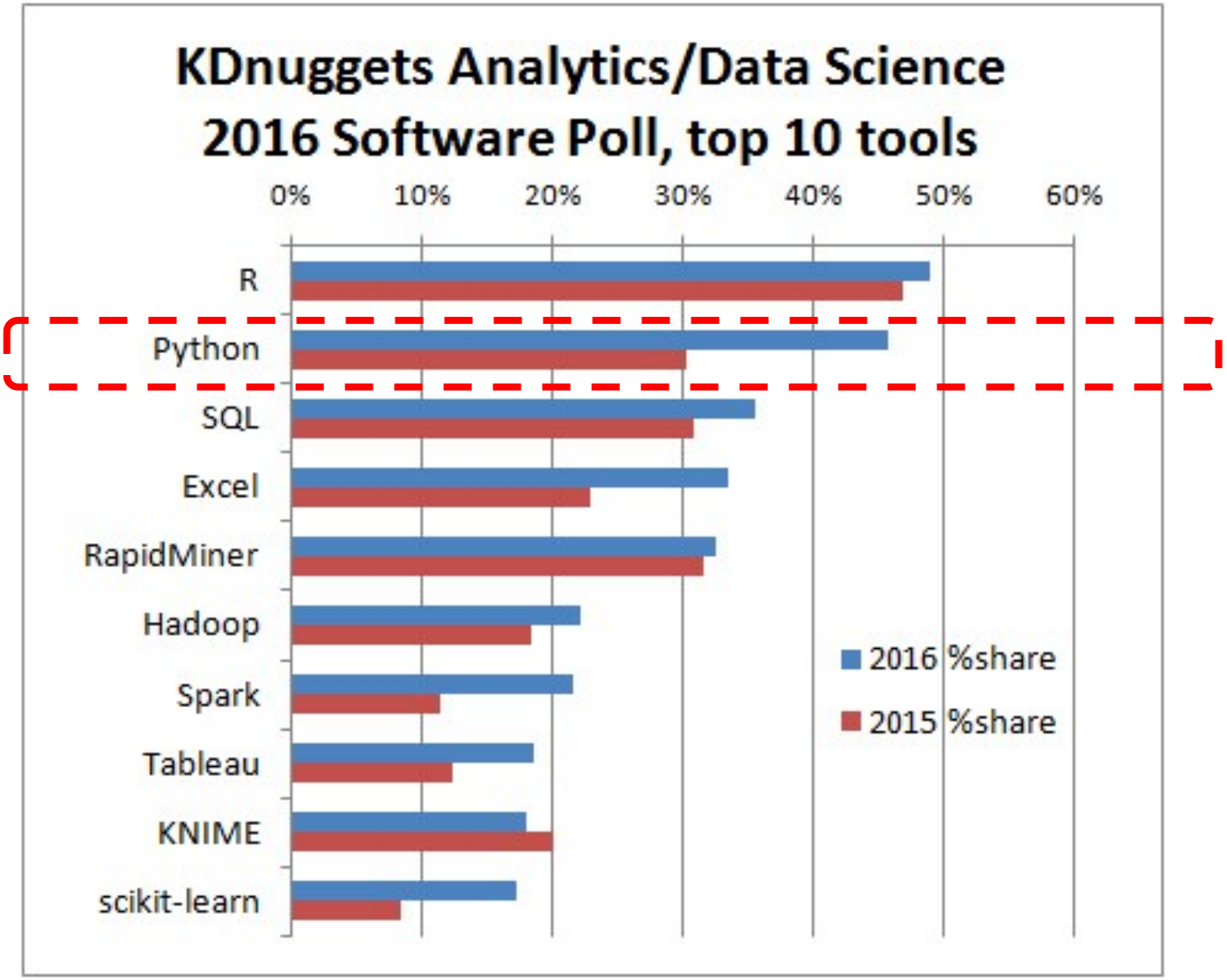
Source: http://www.amazon.com/Mastering-Pandas-Finance-Michael-Heydt/dp/1783985100

Python for Big Data Analytics



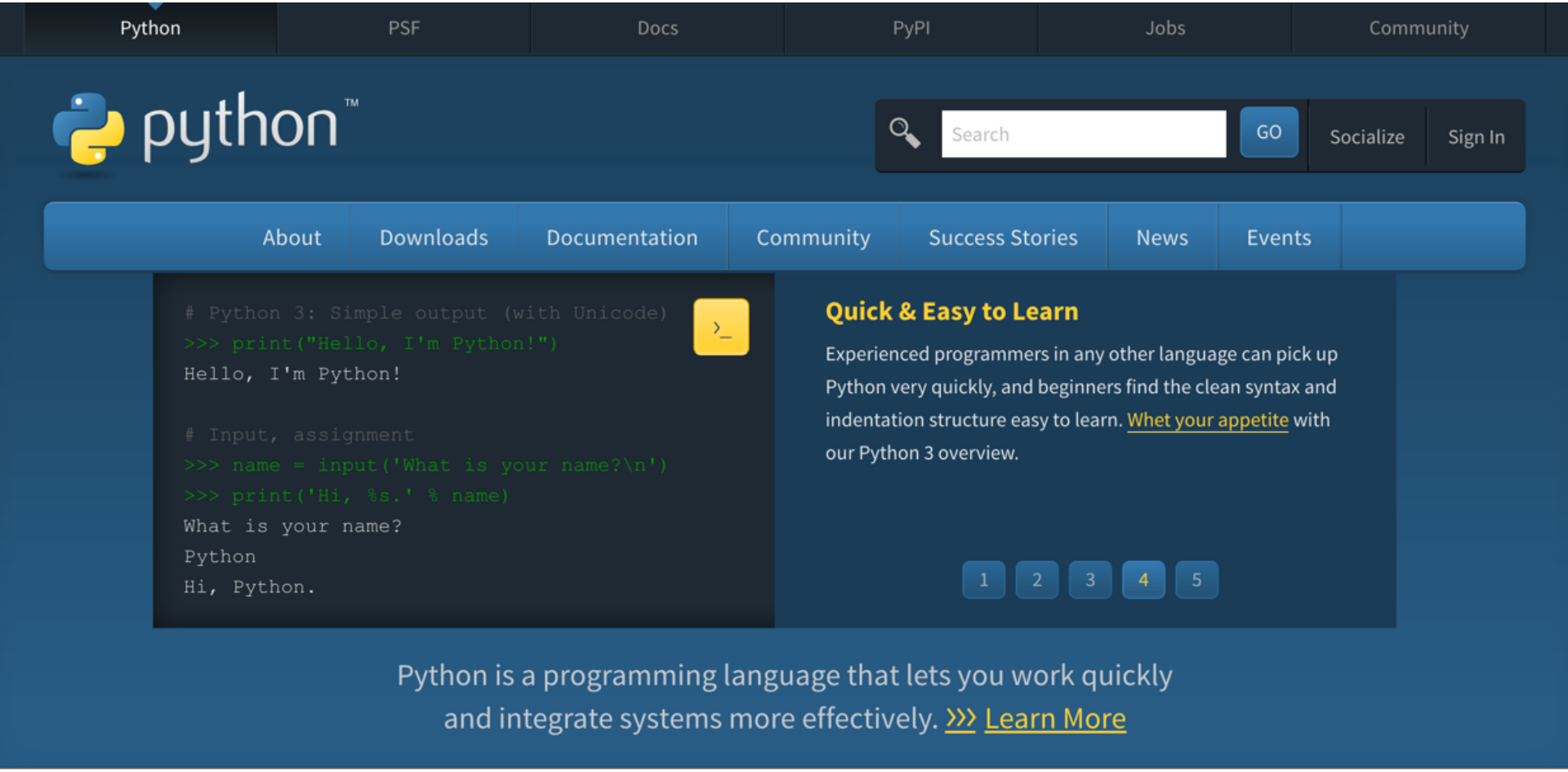
Source: http://spectrum.ieee.org/computing/software/the-2016-top-programming-languages

Python: Analytics and Data Science Software



Source: http://www.kdnuggets.com/2016/06/r-python-top-analytics-data-mining-data-science-software.html

Python



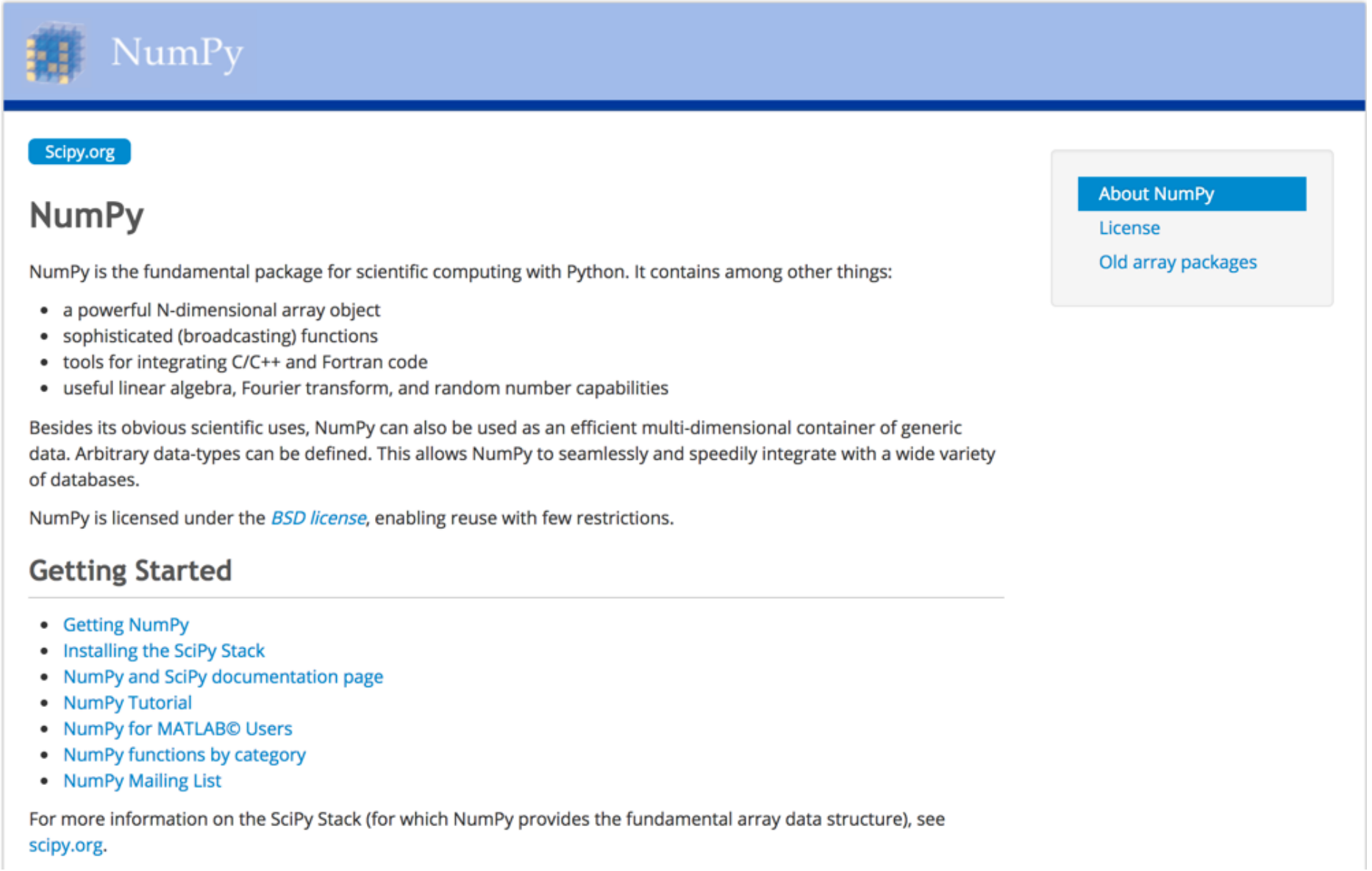
https://www.python.org/

python™

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.

Source: https://www.python.org/doc/essays/blurb/

NumPy



http://www.numpy.org/

NumPy

is the

fundamental package

for

scientific computing

with Python.

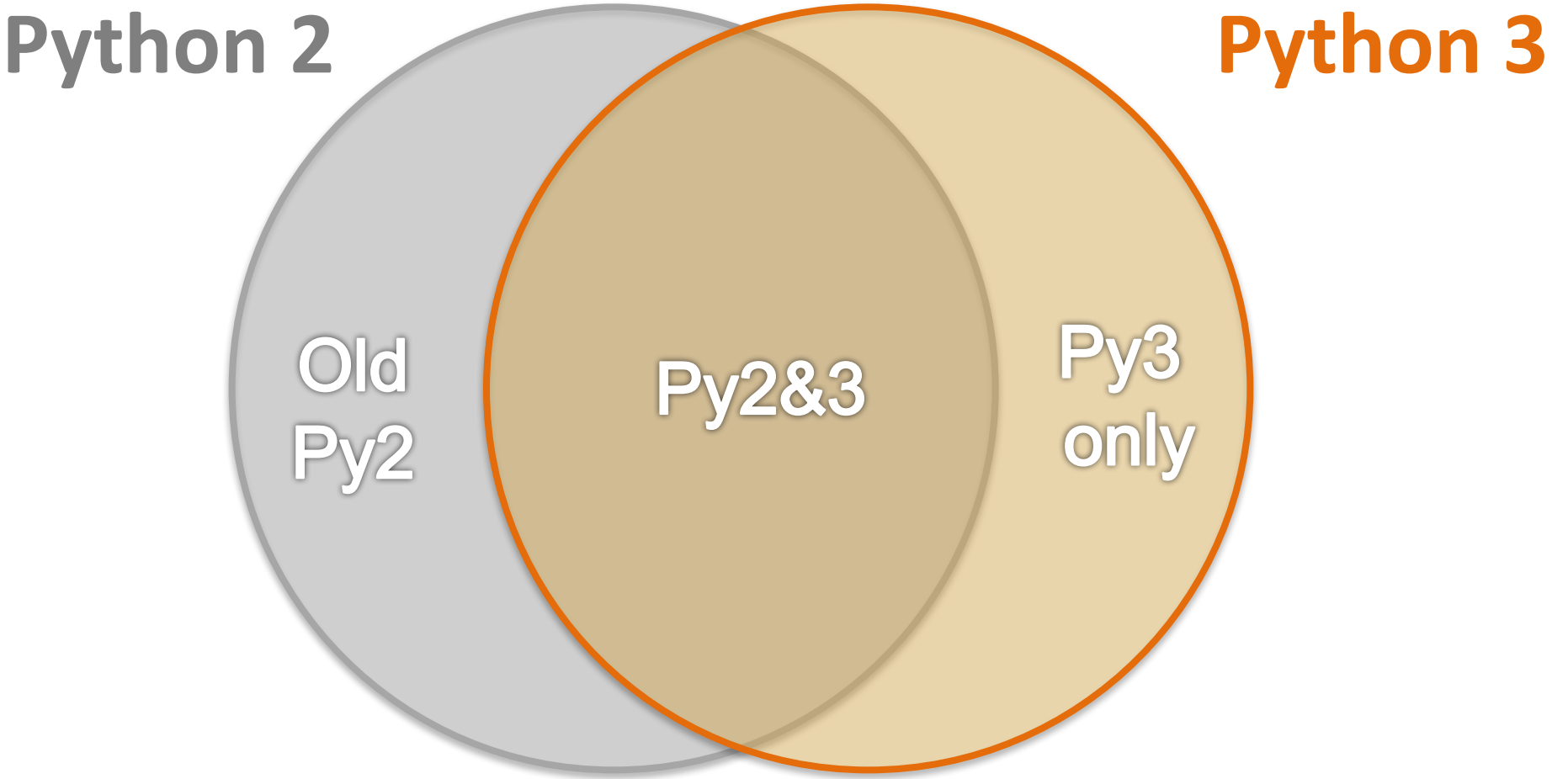
Source: http://www.numpy.org/

Python versions (py2 and py3)

- Python 0.9.0 released in 1991 (first release)
- Python 1.0 released in 1994
- Python 2.0 released in 2000
- Python 2.6 released in 2008
- **Python 2.7 released in 2010**
- **Python 3.0 released in 2008**
- Python 3.3 released in 2010
- Python 3.4 released in 2014
- **Python 3.5 released in 2015**
- **Python 3.6 released in 2016**

Source: Yves Hilpisch (2014), Python for Finance: Analyze Big Financial Data, O'Reilly

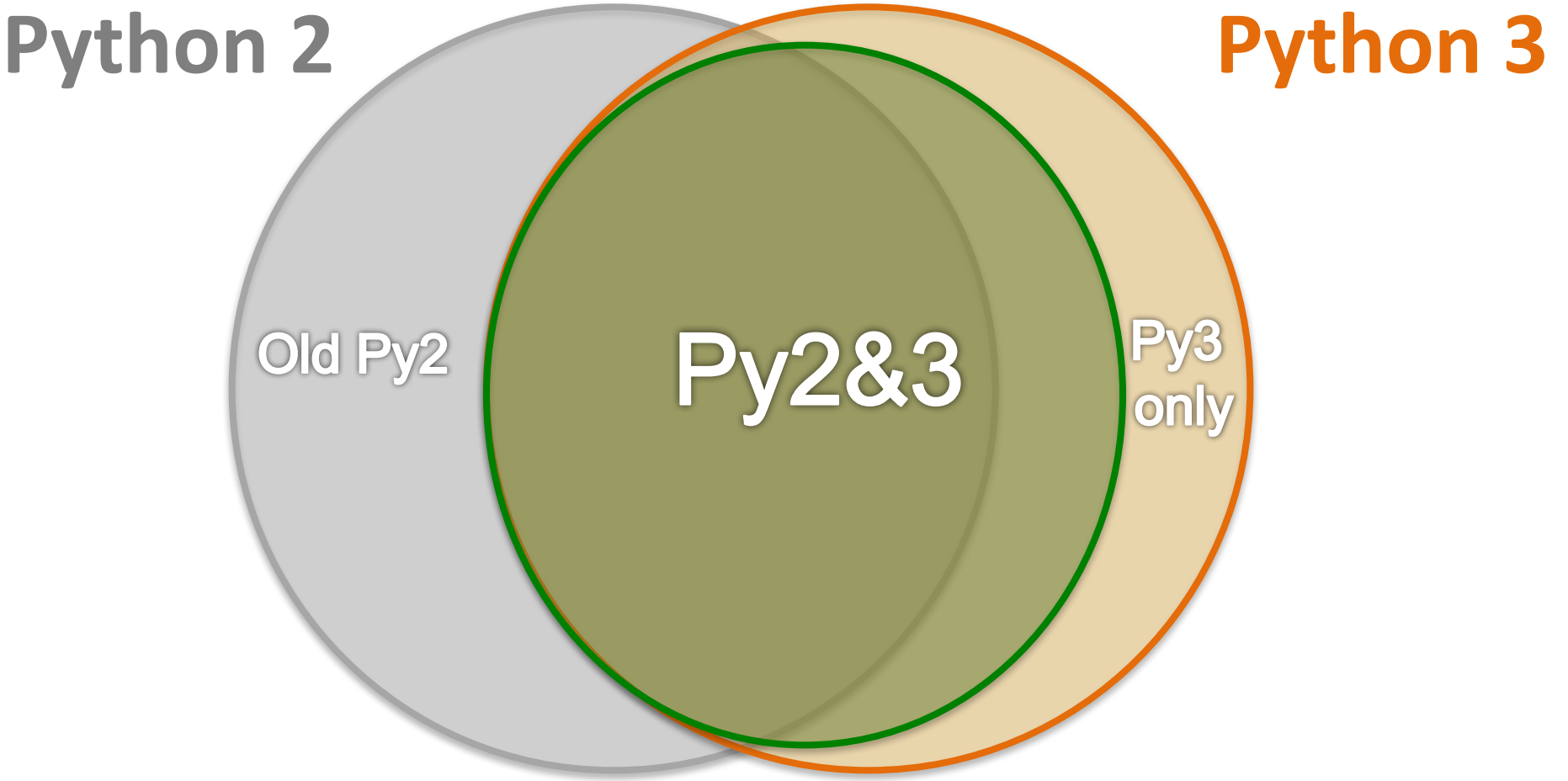
Python (Python 2.7 & Python 3.6) Standard Syntax



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield
https://www.youtube.com/watch?v=KOqk8j11aAI

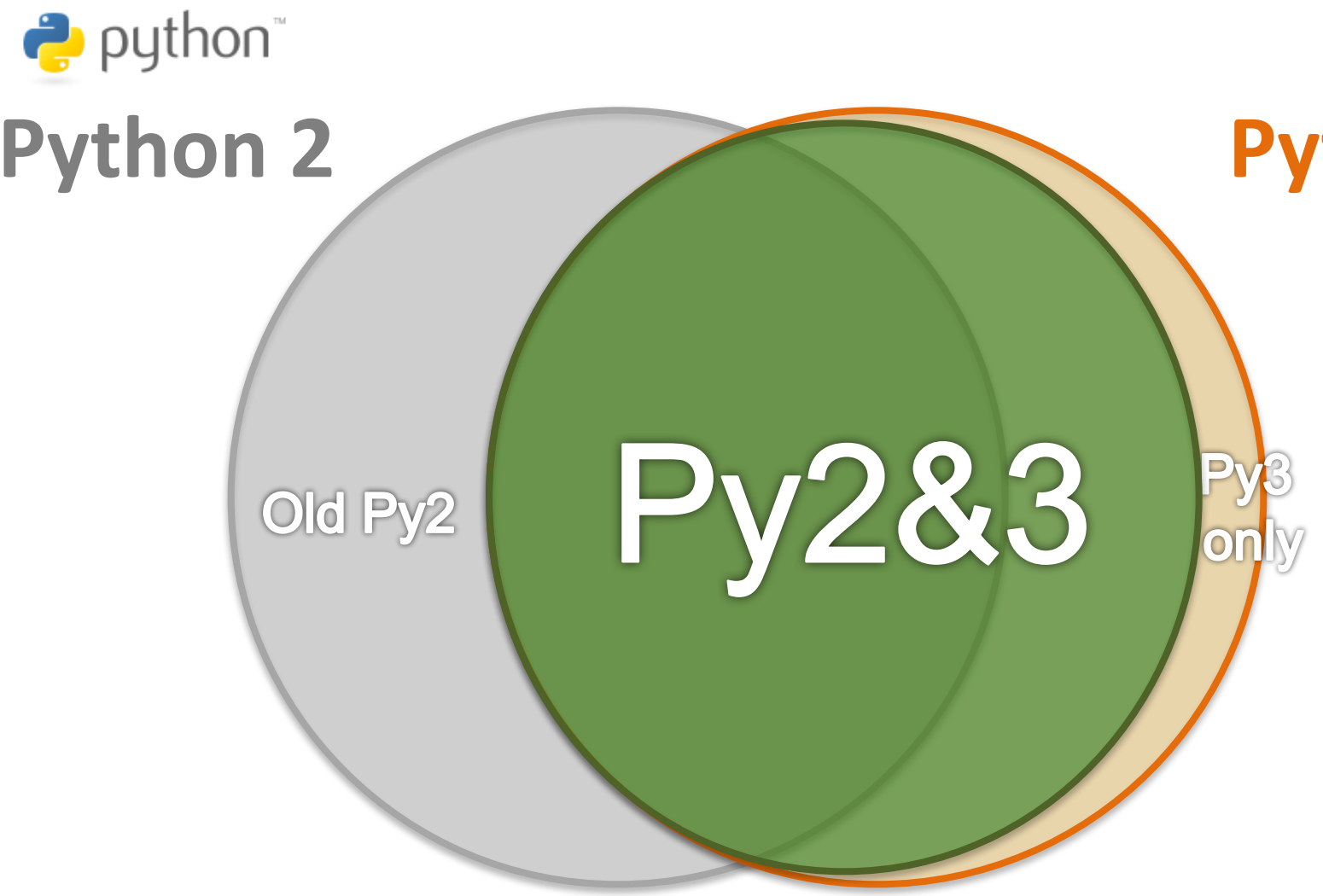
`from __ future __ import ...`

python™



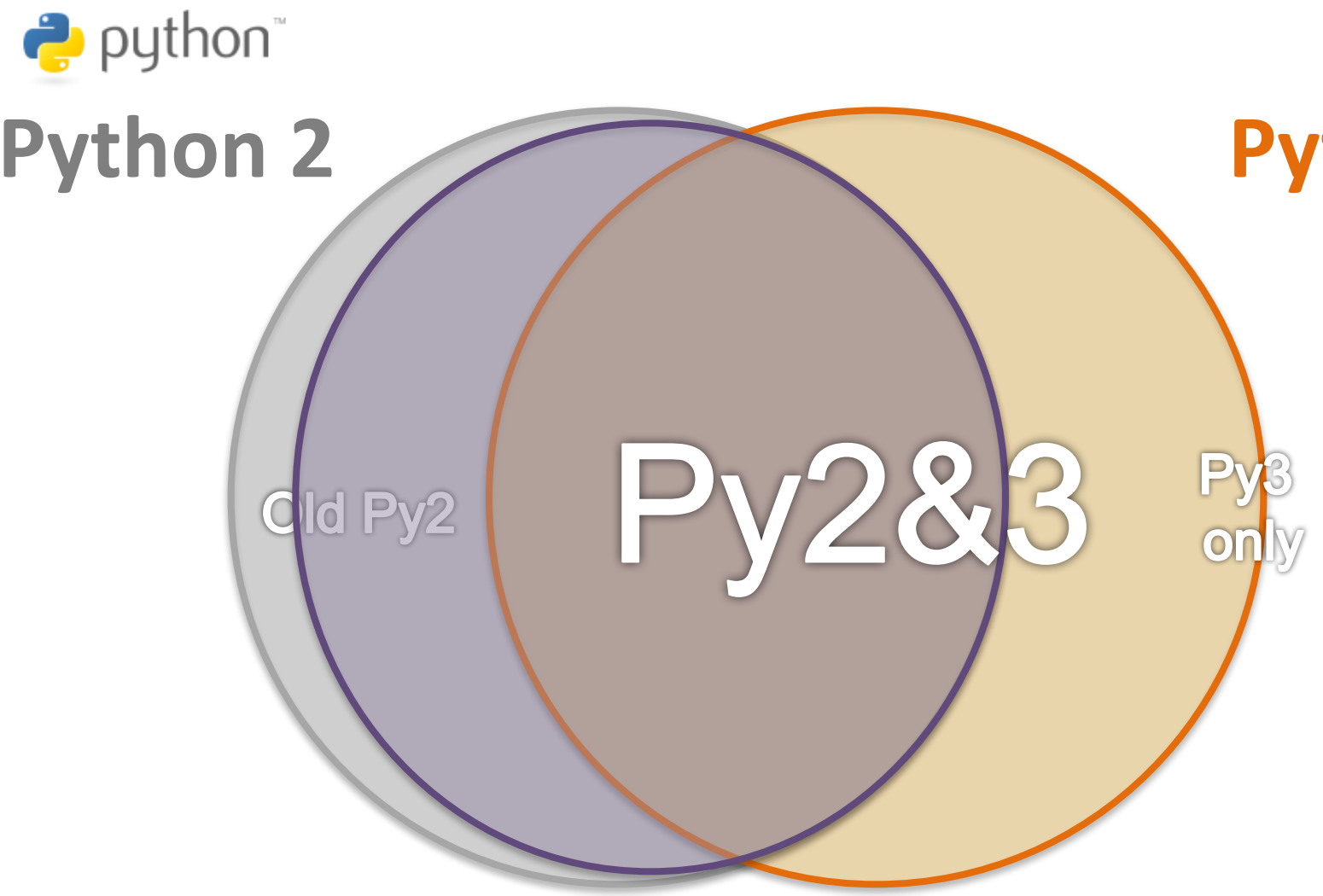
Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield
https://www.youtube.com/watch?v=KOqk8j11aAI

`from future.builtins import *`



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield
https://www.youtube.com/watch?v=KOqk8j11aAI

`from past.builtins import *`

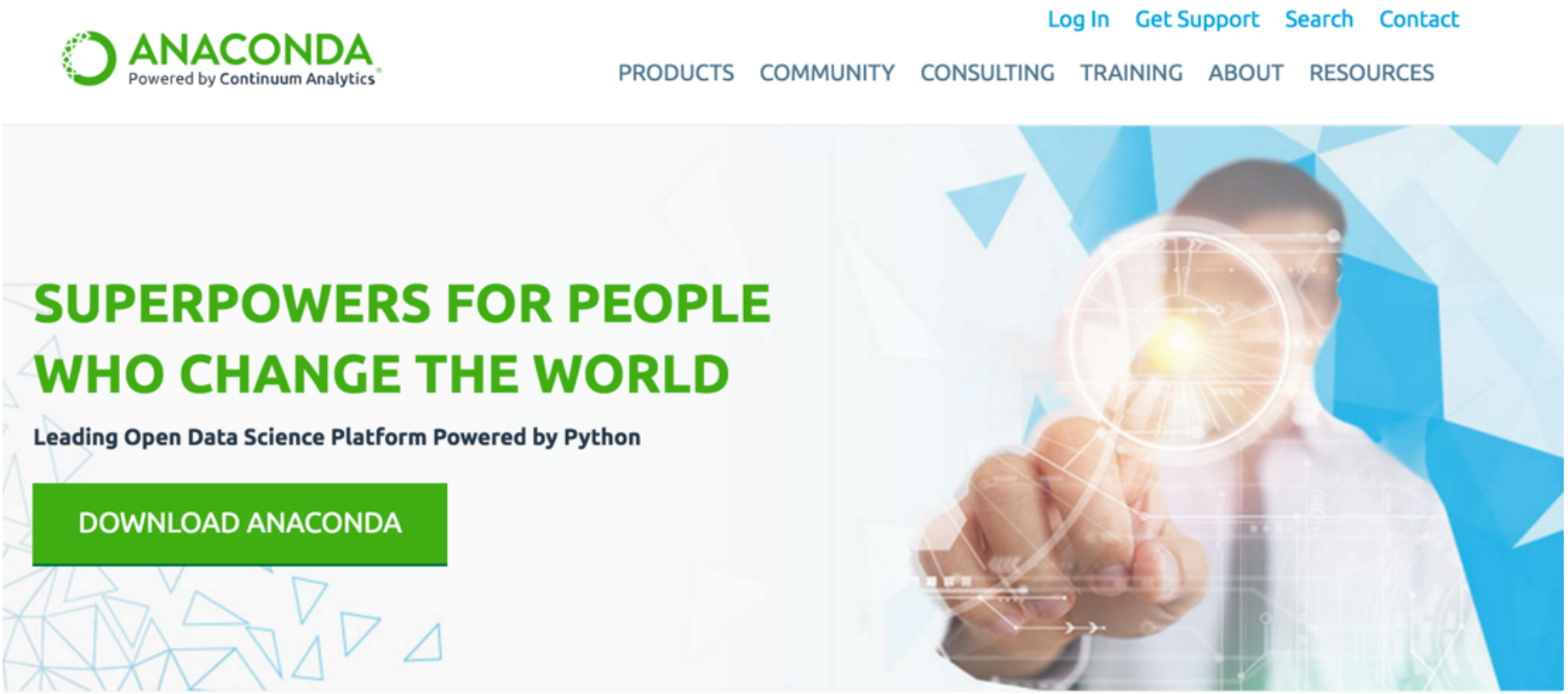


Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield
https://www.youtube.com/watch?v=KOqk8j11aAI

Leading Open Data Science Platform Powered by Python

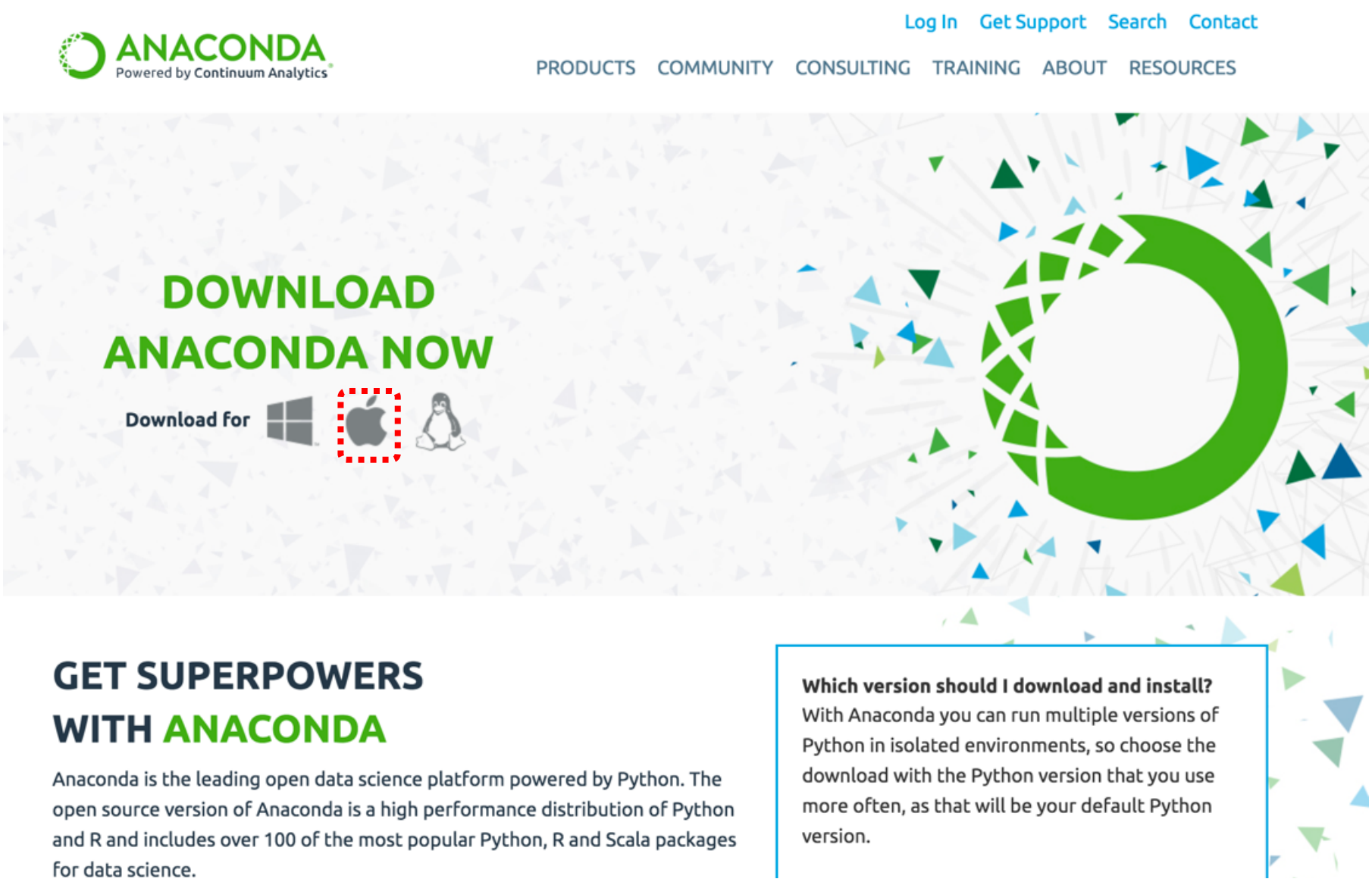
Source: https://www.continuum.io/

Anaconda



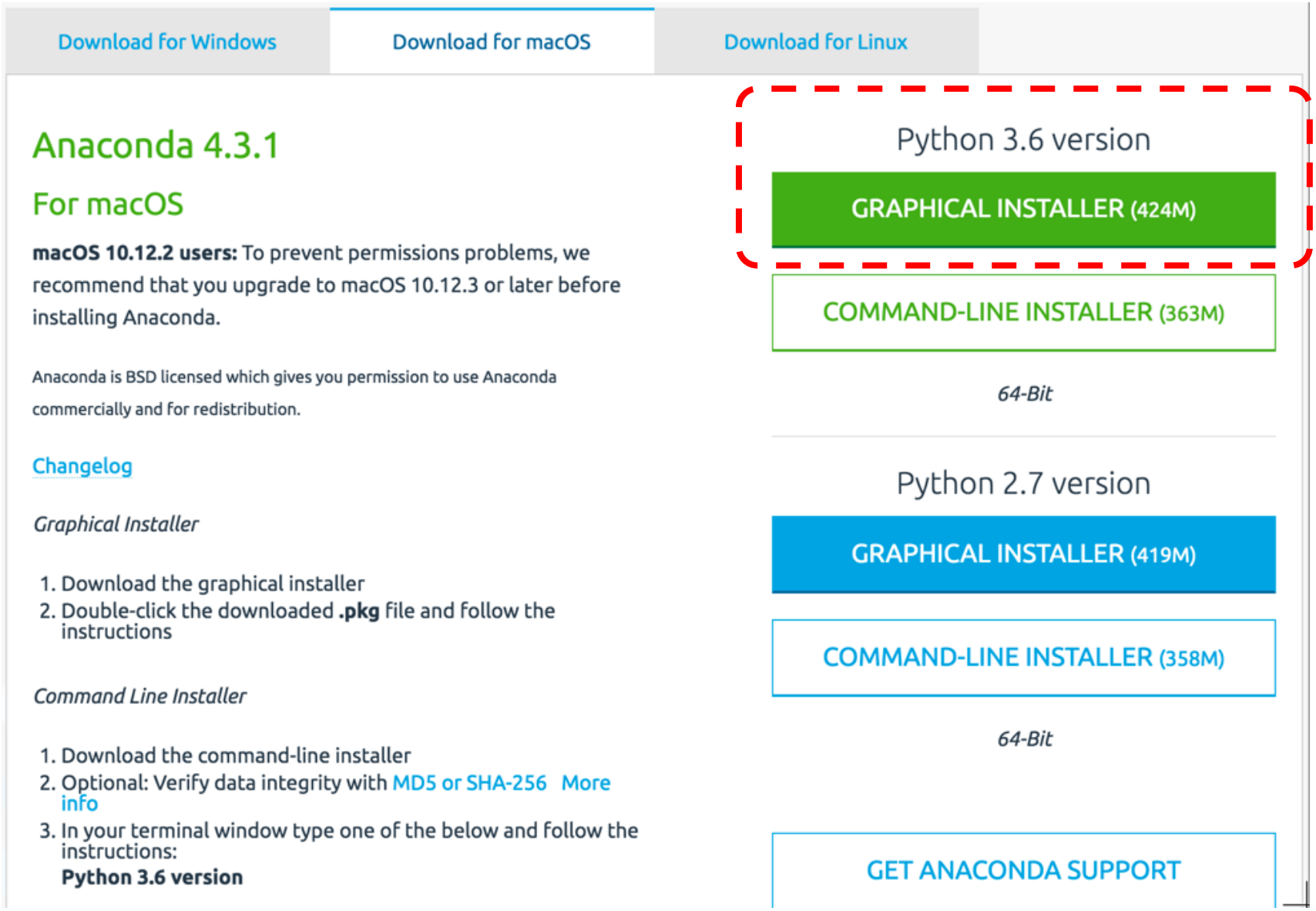
https://www.continuum.io/

Download Anaconda



https://www.continuum.io/downloads

Download Anaconda Python 3.6



https://www.continuum.io/downloads

OS X Anaconda Python 3.6 Installation

Command Line Installer

Download the command-line installer

In your terminal window type one of the below and follow the instructions:

Python 3.6 version

```
bash Anaconda3-4.3.1-MacOSX-x86_64.sh
```

Python 2.7 version

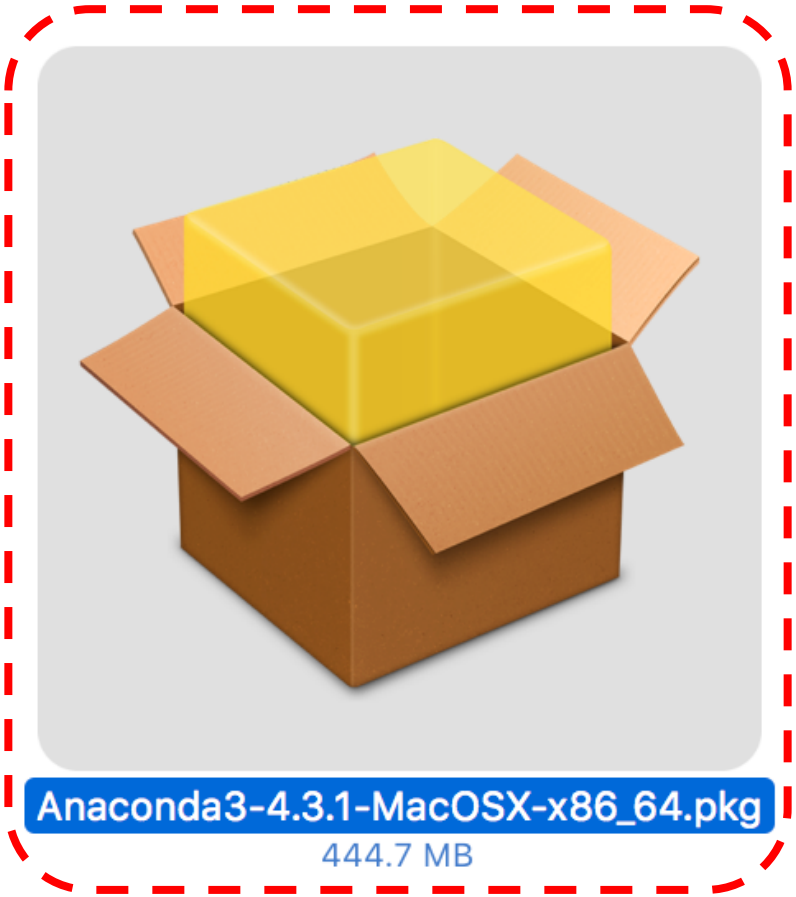
```
bash Anaconda2-4.3.1-MacOSX-x86_64.sh
```

https://www.continuum.io/downloads

OS X Anaconda 3 - 4.3.1

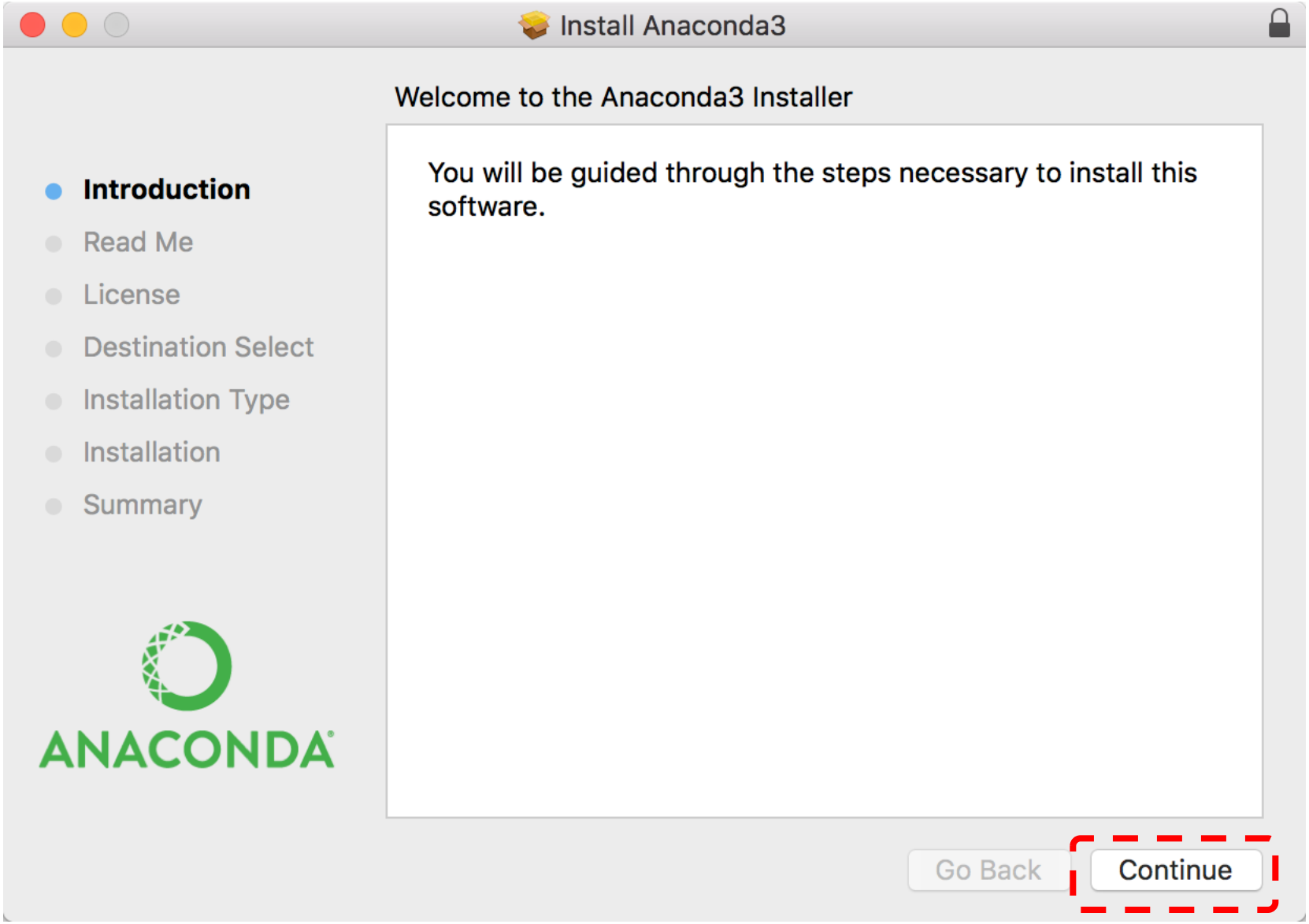
Python 3.6 Installation

Anaconda3-4.3.1-MacOSX-x86_64.pkg

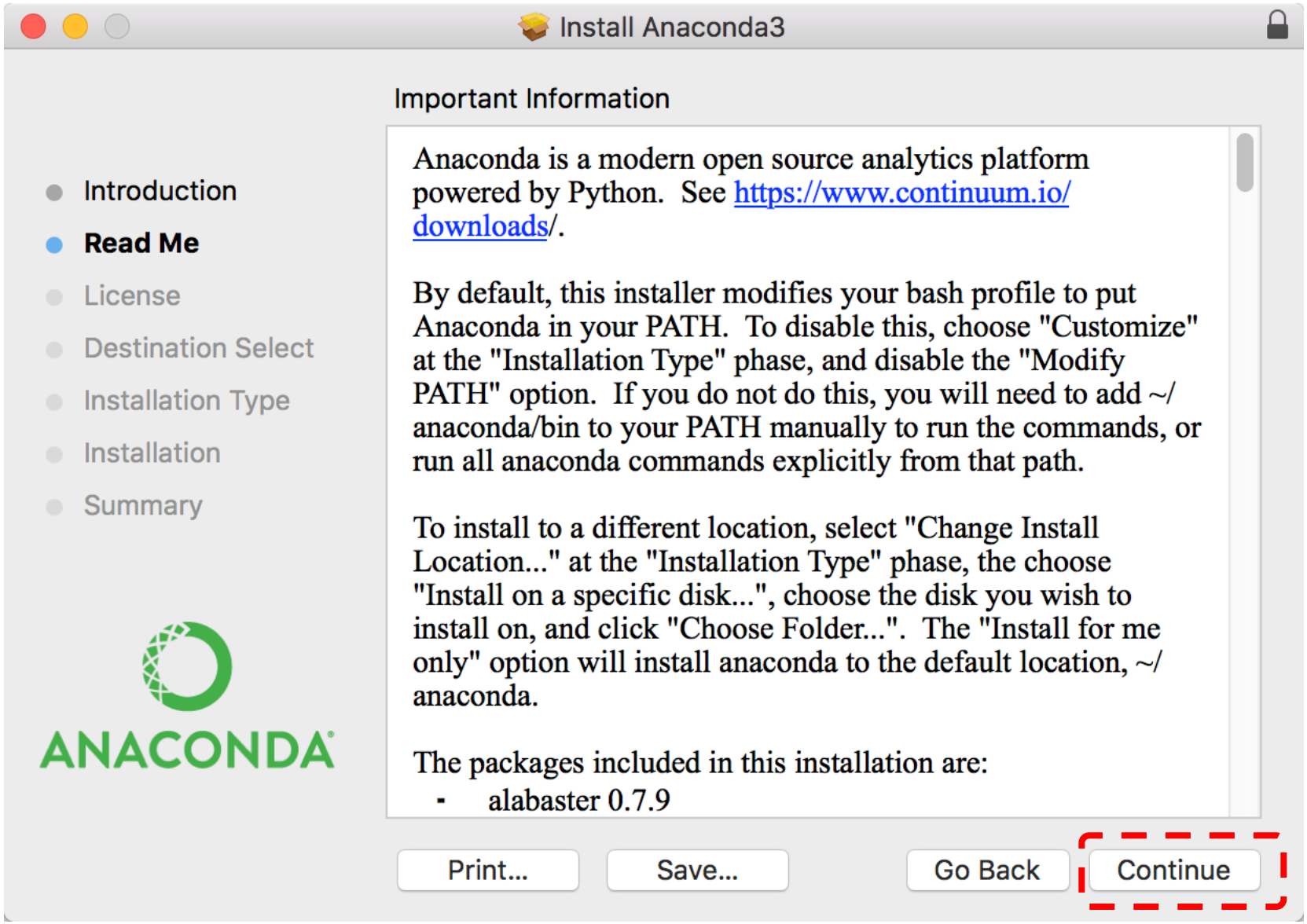


Installer package

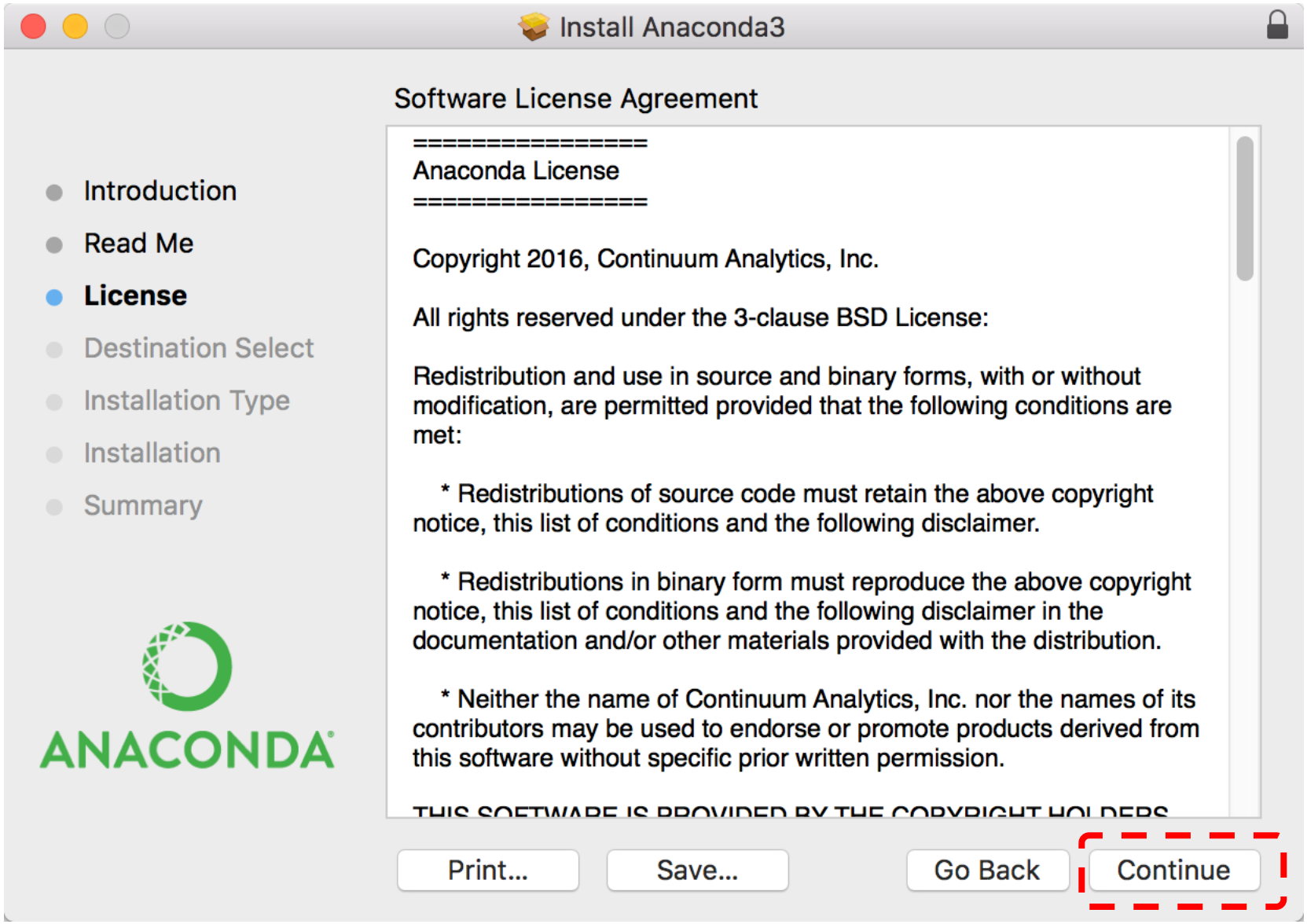
Install Anaconda 3



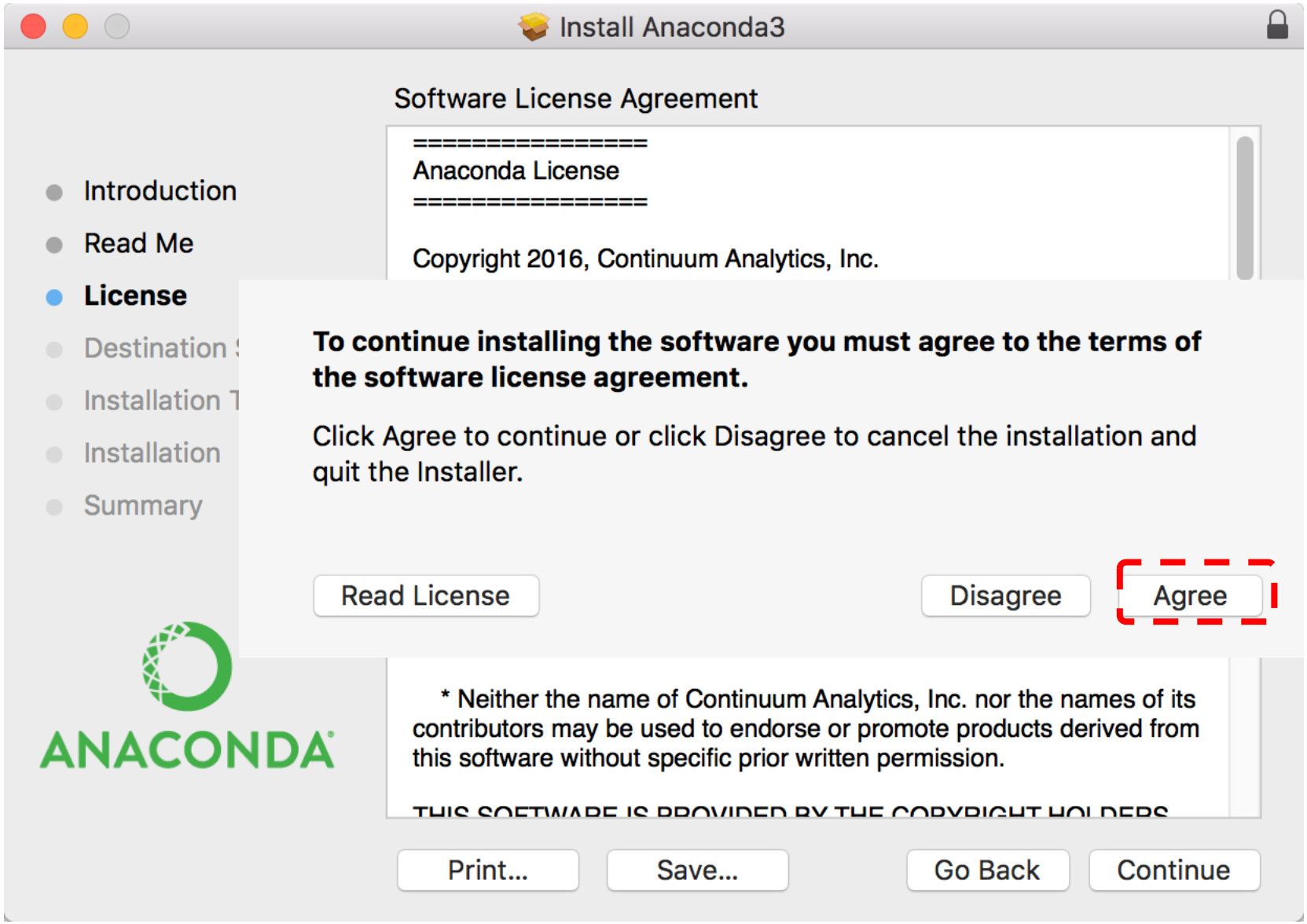
Install Anaconda 3



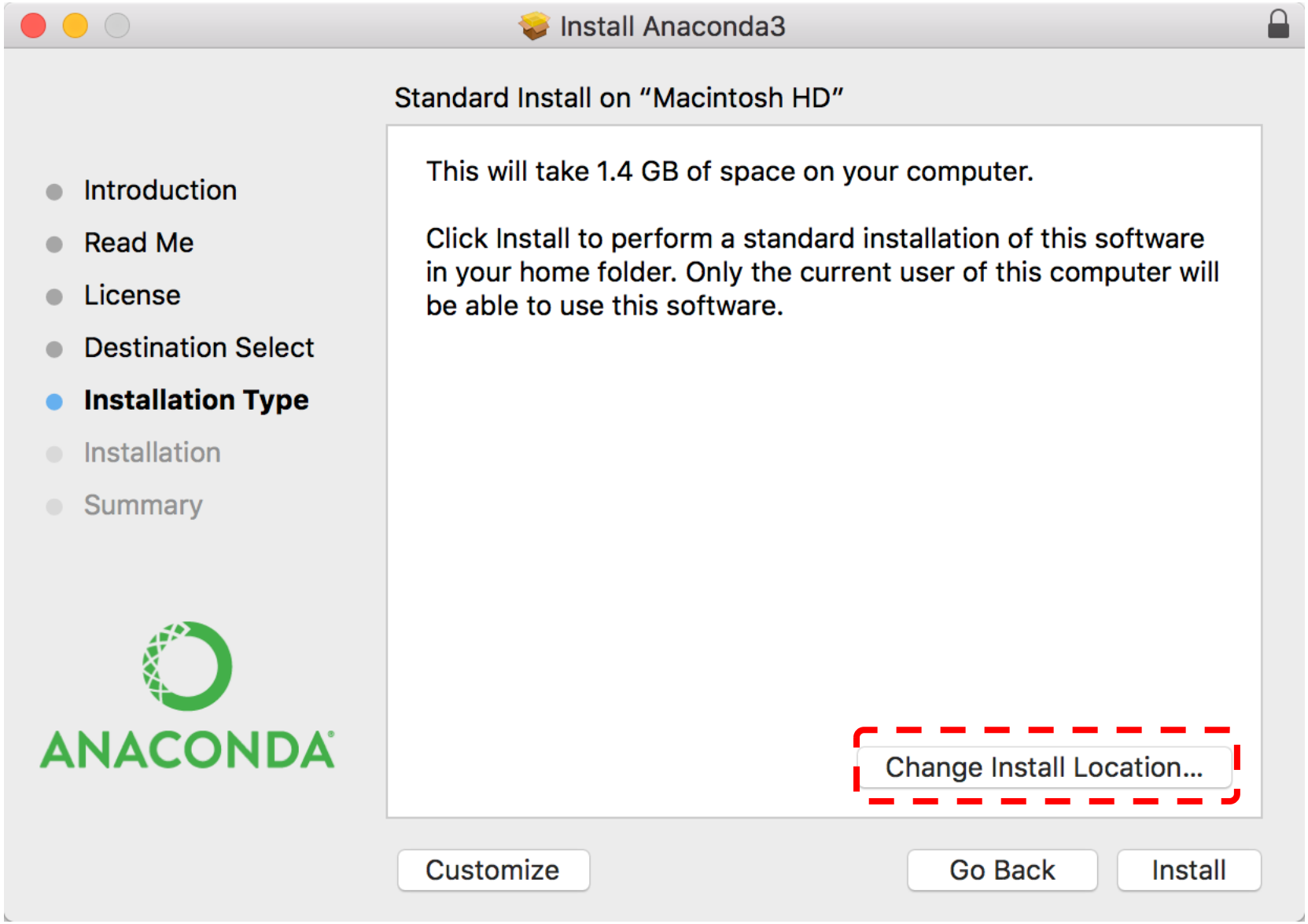
Install Anaconda 3



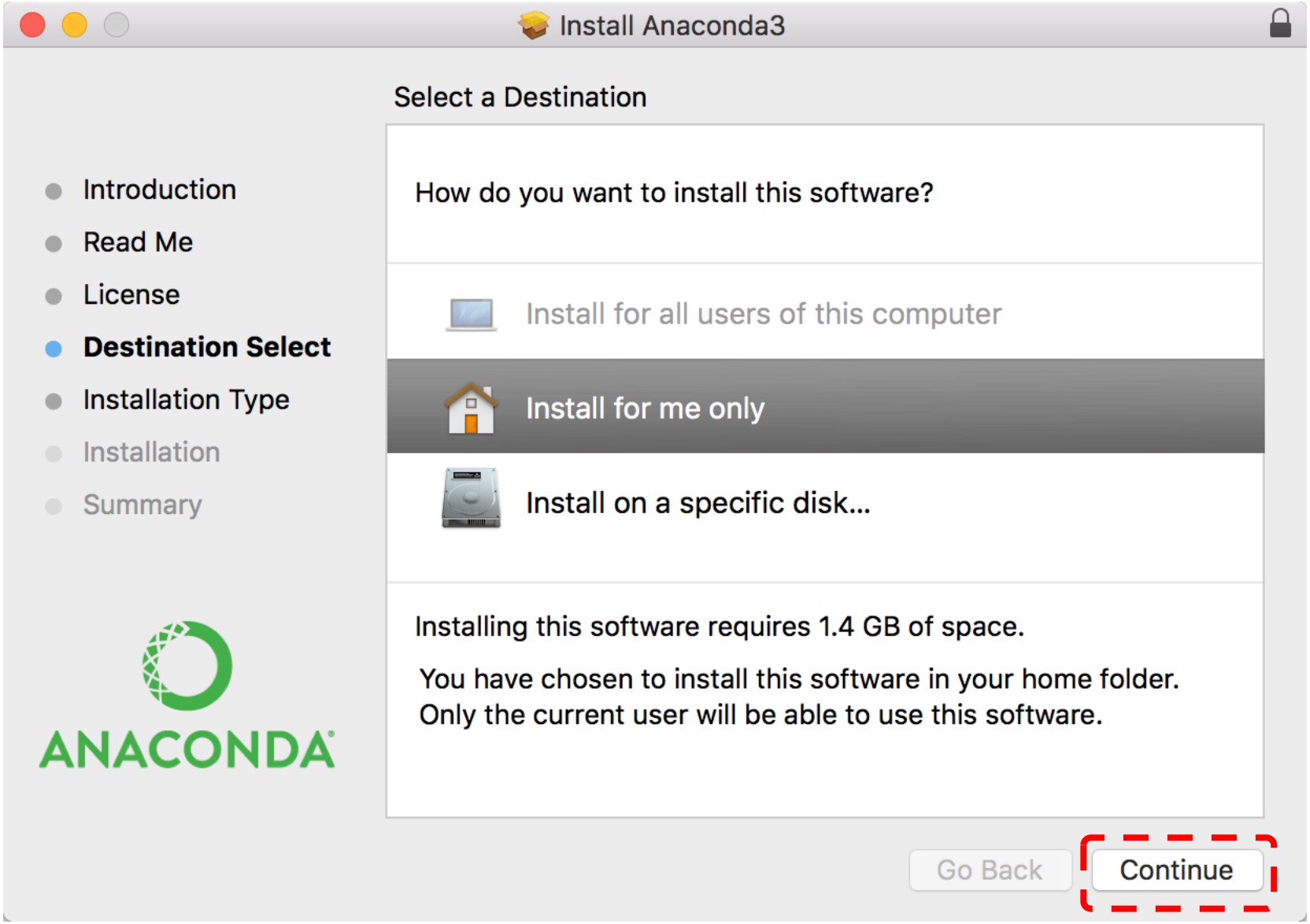
Install Anaconda 3



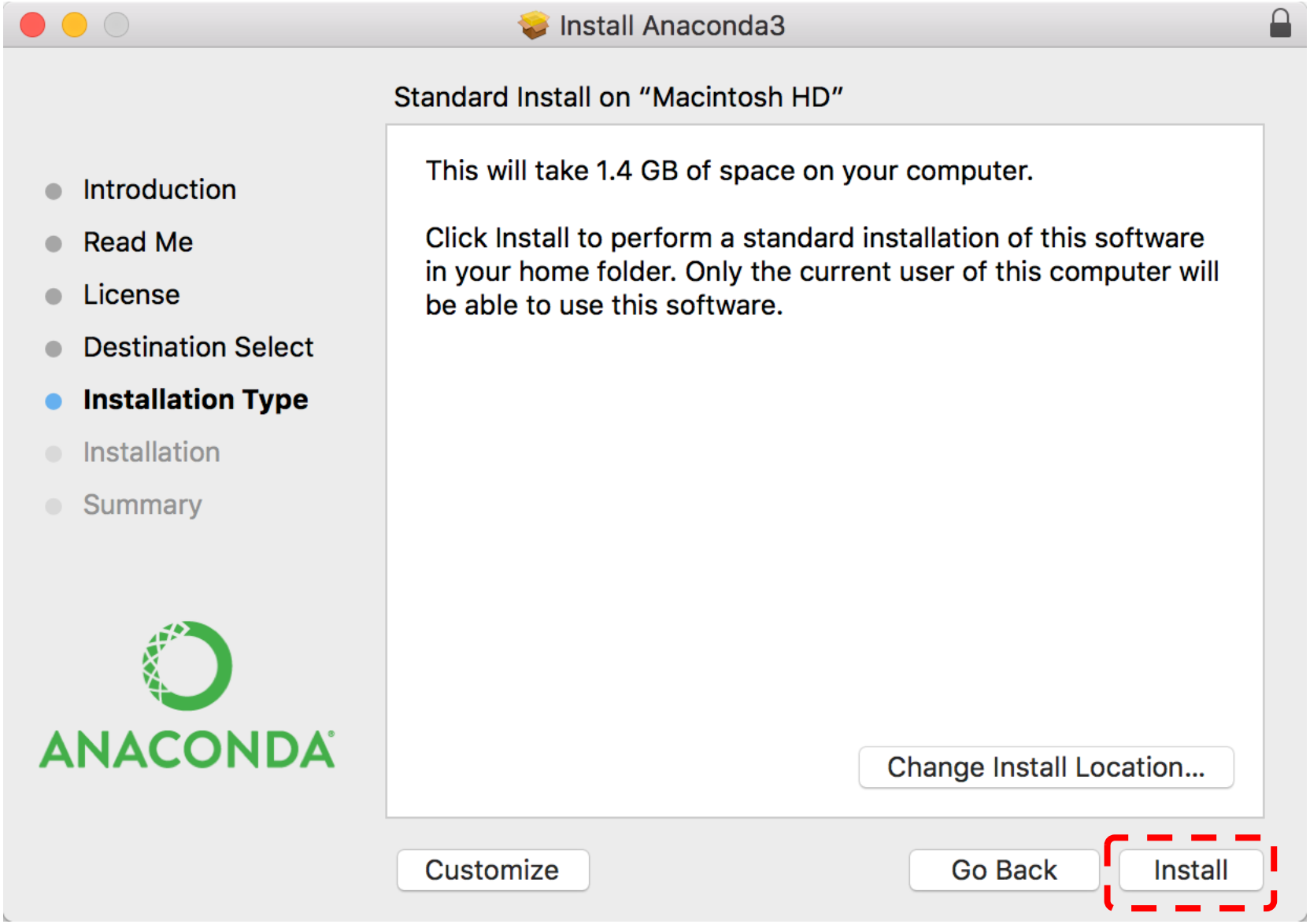
Install Anaconda 3



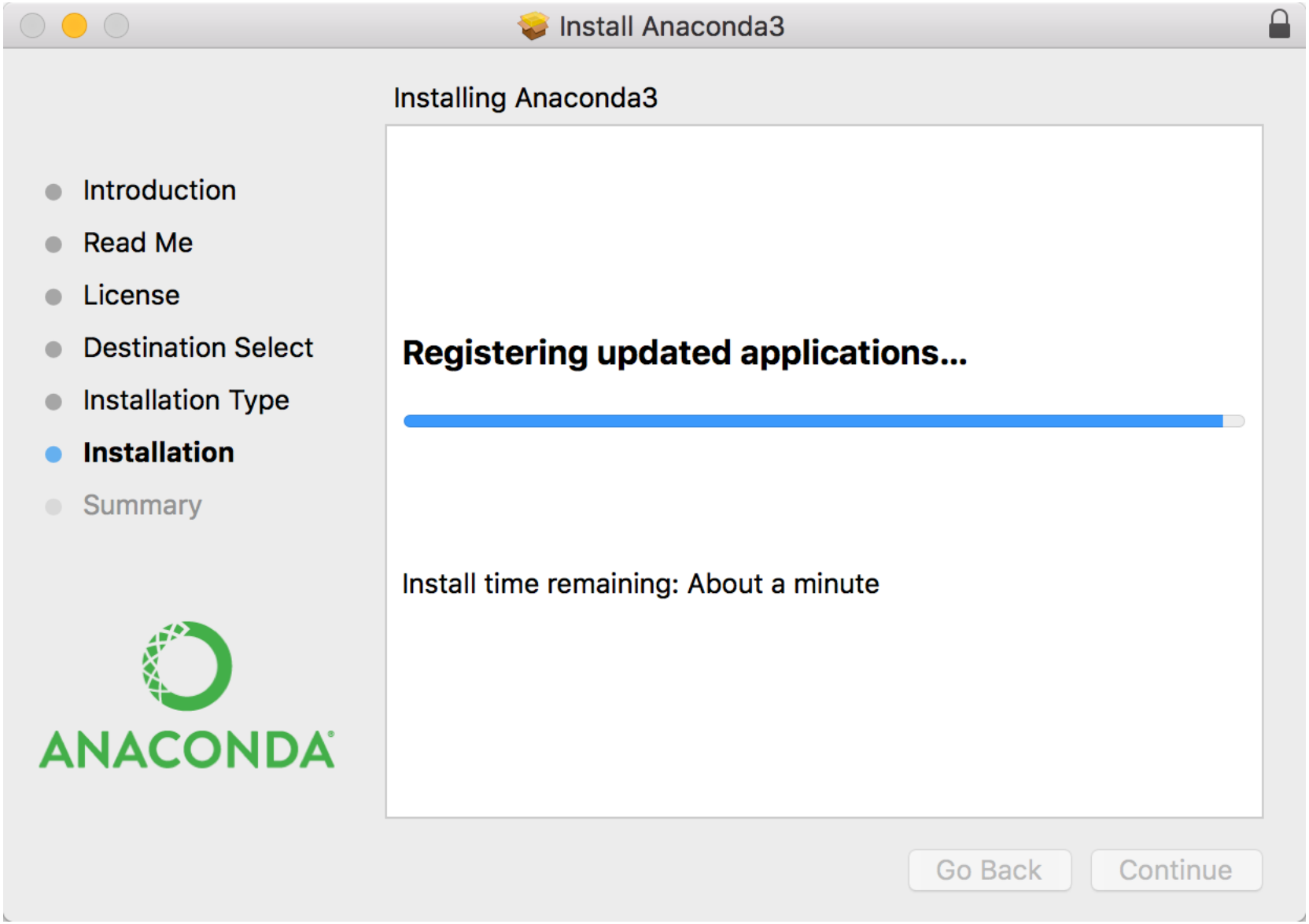
Install Anaconda 3



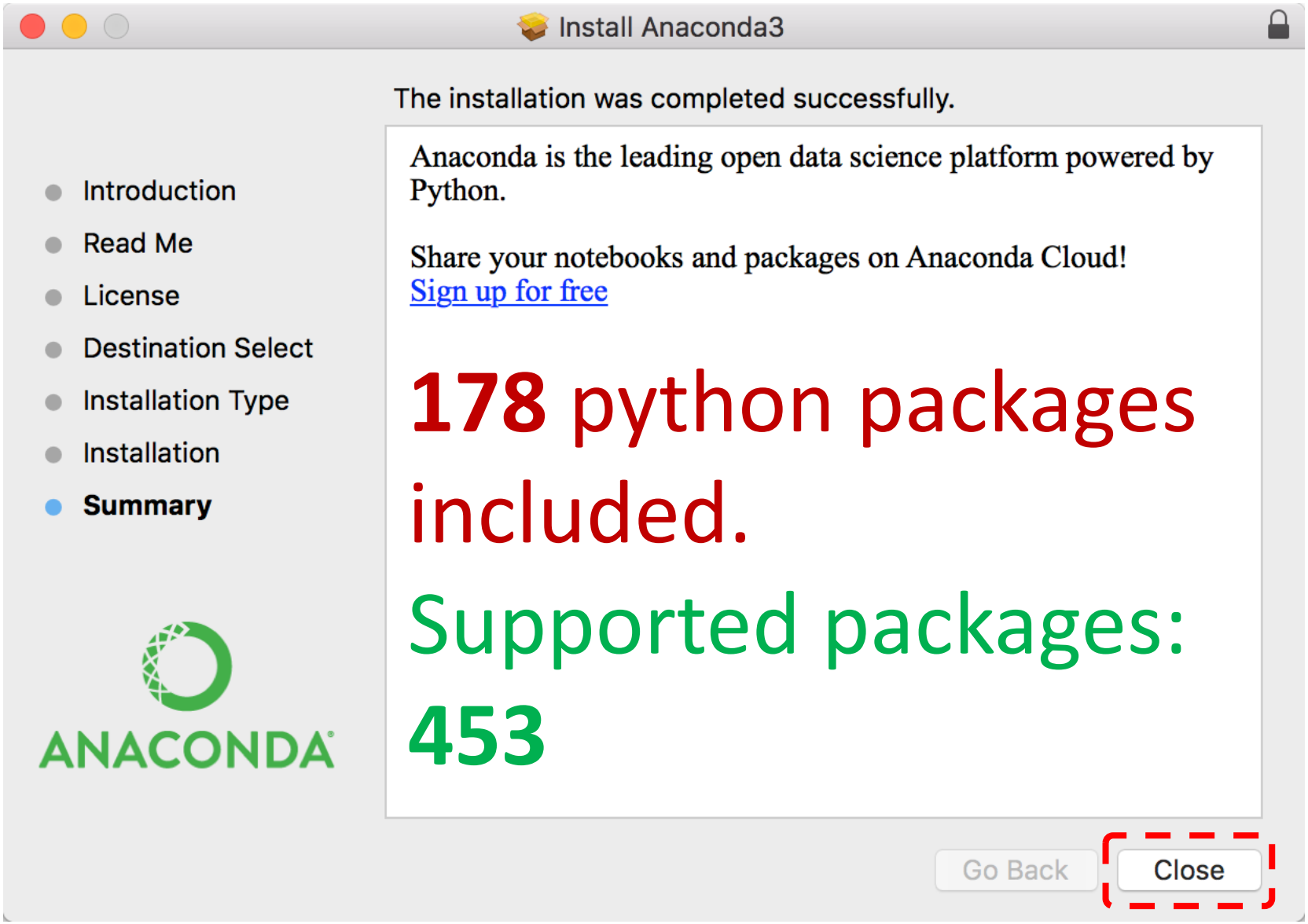
Install Anaconda 3



Install Anaconda 3



Install Anaconda 3



Source: https://docs.continuum.io/anaconda/pkg-docs

Install Anaconda 3

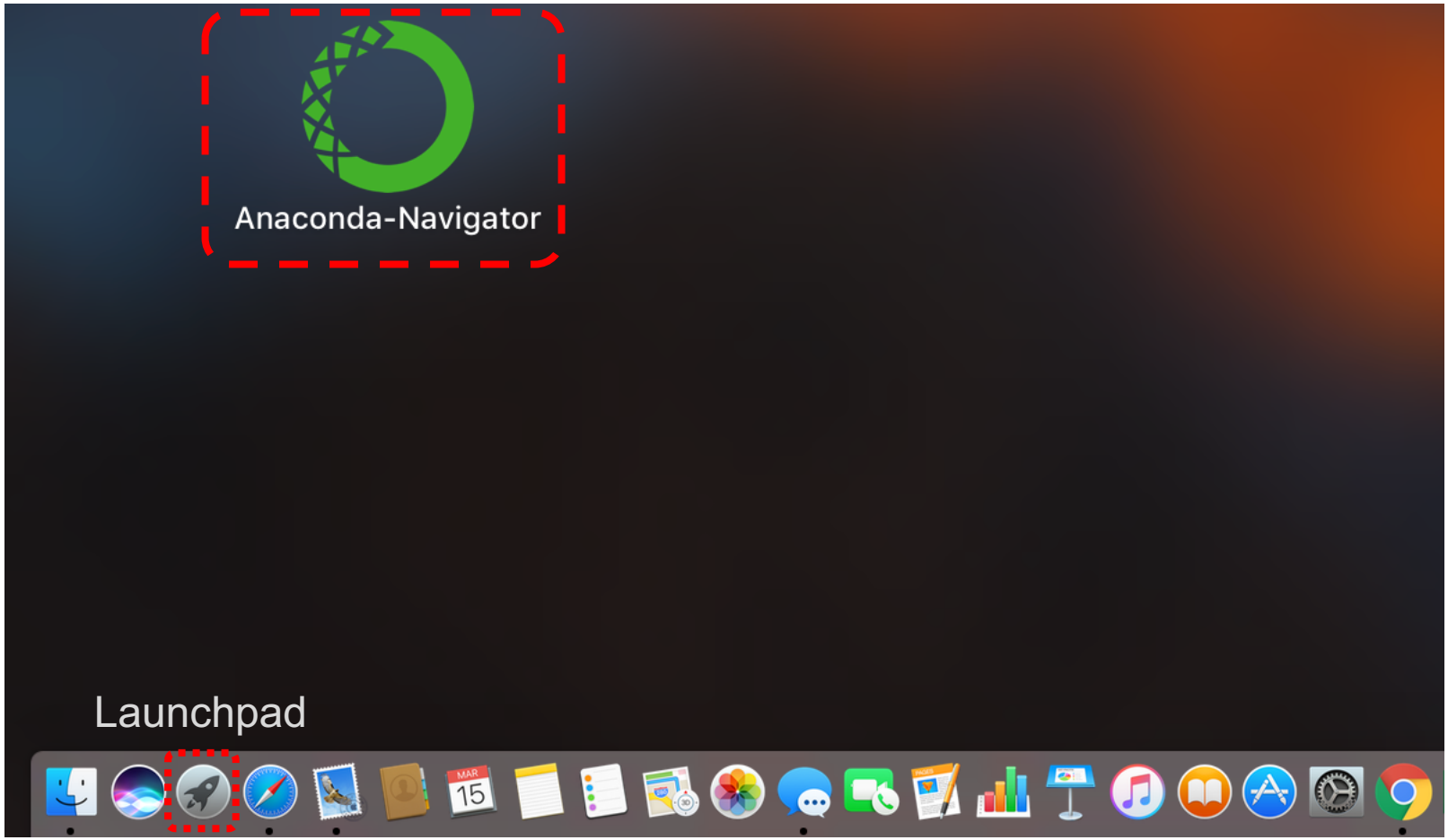
1	_license	1.1
2	alabaster	0.7.9
3	anaconda	4.3.1
4	anaconda-client	1.6.0
5	anaconda-navigator	1.5.0
6	anaconda-project	0.4.1
7	appnope	0.1.0
8	appscript	1.0.1
9	astroid	1.4.9
10	astropy	1.3
11	babel	2.3.4
12	backports	1.0
13	beautifulsoup4	4.5.3
14	bitarray	0.8.1
15	blaze	0.10.1
16	bokeh	0.12.4
17	boto	2.45.0
18	bottleneck	1.2.0
19	cffi	1.9.1
20	chardet	2.3.0
21	chest	0.2.3
22	click	6.7
23	cloudpickle	0.2.2
24	clyent	1.2.2
25	colorama	0.3.7
26	conda	4.3.14
27	conda-env	2.6.0
28	configobj	5.0.6
29	contextlib2	0.5.4
30	cryptography	1.7.1
31	curl	7.52.1
32	cycler	0.10.0
33	cython	0.25.2
34	cytoolz	0.8.2
35	dask	0.13.0
36	datashape	0.5.4
37	decorator	4.0.11
38	dill	0.2.5
39	docutils	0.13.1
40	entrypoints	0.2.2
41	et_xmlfile	1.0.1
42	fastcache	1.0.2
43	flask	0.12
44	flask-cors	3.0.2
45	freetype	2.5.5
46	get_terminal_size	1.0.0
47	gevent	1.2.1
48	greenlet	0.4.11
49	h5py	2.6.0
50	hdf5	1.8.17
51	heapdict	1.0.0
52	icu	54.1
53	idna	2.2
54	imagesize	0.7.1
55	ipykernel	4.5.2
56	ipython	5.1.0
57	ipython_genutils	0.1.0
58	ipywidgets	5.2.2
59	isort	4.2.5
60	itsdangerous	0.24
61	jbig	2.1
62	jdcal	1.3
63	jedi	0.9.0
64	jinja2	2.9.4
65	jpeg	9b
66	jsonschema	2.5.1
67	jupyter	1.0.0
68	jupyter_client	4.4.0
69	jupyter_console	5.0.0
70	jupyter_core	4.2.1
71	lazy-object-proxy	1.2.2
72	libiconv	1.14
73	libpng	1.6.27
74	libtiff	4.0.6
75	libxml2	2.9.4
76	libxslt	1.1.29
77	llvmlite	0.15.0
78	locket	0.2.0
79	lxml	3.7.2
80	markupsafe	0.23
81	matplotlib	2.0.0
82	mistune	0.7.3
83	mkl	2017.0.1
84	mkl-service	1.1.2
85	mpmath	0.19
86	multipledispatch	0.4.9
87	nbconvert	4.2.0
88	nbformat	4.2.0
89	networkx	1.11
90	nltk	3.2.2
91	nose	1.3.7
92	notebook	4.3.1
93	numba	0.30.1
94	numexpr	2.6.1
95	numpy	1.11.3
96	numpydoc	0.6.0
97	odo	0.5.0
98	openpyxl	2.4.1
99	openssl	1.0.2k
100	pandas	0.19.2
101	partd	0.3.7
102	path.py	10.0
103	pathlib2	2.2.0
104	patsy	0.4.1
105	pep8	1.7.0
106	pexpect	4.2.1
107	pickleshare	0.7.4
108	pillow	4.0.0
109	pip	9.0.1
110	ply	3.9
111	prompt_toolkit	1.0.9
112	psutil	5.0.1
113	ptyprocess	0.5.1
114	py	1.4.32
115	pyasn1	0.1.9
116	pycosat	0.6.1
117	pycparser	2.17
118	pycrypto	2.6.1
119	pycurl	7.43.0
120	pyflakes	1.5.0
121	pygments	2.1.3
122	pylint	1.6.4
123	pyopenssl	16.2.0
124	pyparsing	2.1.4
125	pyqt	5.6.0
126	pytables	3.3.0
127	pytest	3.0.5
128	python	3.6.0
129	python-dateutil	2.6.0
130	python.app	1.2
131	pytz	2016.10
132	pyyaml	3.12
133	pyzmq	16.0.2
134	qt	5.6.2
135	qtawesome	0.4.3
136	qtconsole	4.2.1
137	qtpy	1.2.1
138	readline	6.2
139	redis	3.2.0
140	redis-py	2.10.5
141	requests	2.12.4
142	rope	0.9.4
143	ruamel_yaml	0.11.14
144	scikit-image	0.12.3
145	scikit-learn	0.18.1
146	scipy	0.18.1
147	seaborn	0.7.1
148	setuptools	27.2.0
149	simplegeneric	0.8.1
150	singledispatch	3.4.0.3
151	sip	4.18
152	six	1.10.0
153	snowballstemmer	1.2.1
154	sockjs-tornado	1.0.3
155	sphinx	1.5.1
156	spyder	3.1.2
157	sqlalchemy	1.1.5
158	sqlite	3.13.0
159	statsmodels	0.6.1
160	sympy	1.0
161	terminado	0.6
162	tk	8.5.18
163	toolz	0.8.2
164	tornado	4.4.2
165	traitlets	4.3.1
166	unicodecsv	0.14.1
167	wcwidth	0.1.7
168	werkzeug	0.11.15
169	wheel	0.29.0
170	widgetsnbextension	1.2.6
171	wrapt	1.10.8
172	xlrd	1.0.0
173	xlsxwriter	0.9.6
174	xlwings	0.10.2
175	xlwt	1.2.0
176	xz	5.2.2
177	yaml	0.1.6
178	zlib	1.2.8

	py36_0
	py36_4
	py36_0
	py36_0
	py36_0
	0
	py36_0
	py36_1
	py36_0
	2
	0
	py36_0
	py36_0
	py36_1
	py36_1
np111py36_1	
np111py36_1	
np111py36_1	
	py36_0
	py36_0
	py36_1
	py36_0

	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	0
np111py36_1	
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	py36_0
	1
	0
	3

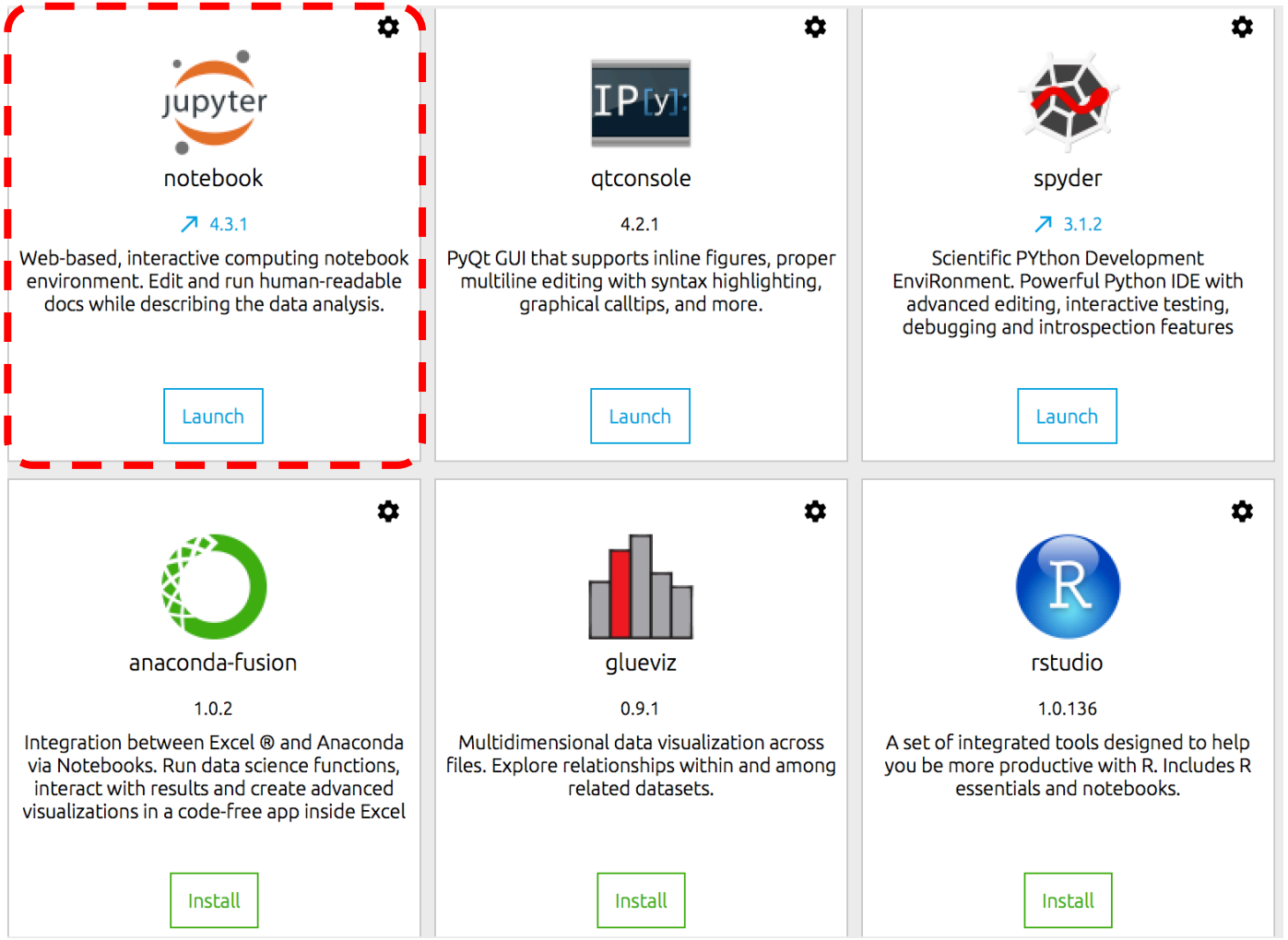
178 python packages included.

Anaconda-Navigator

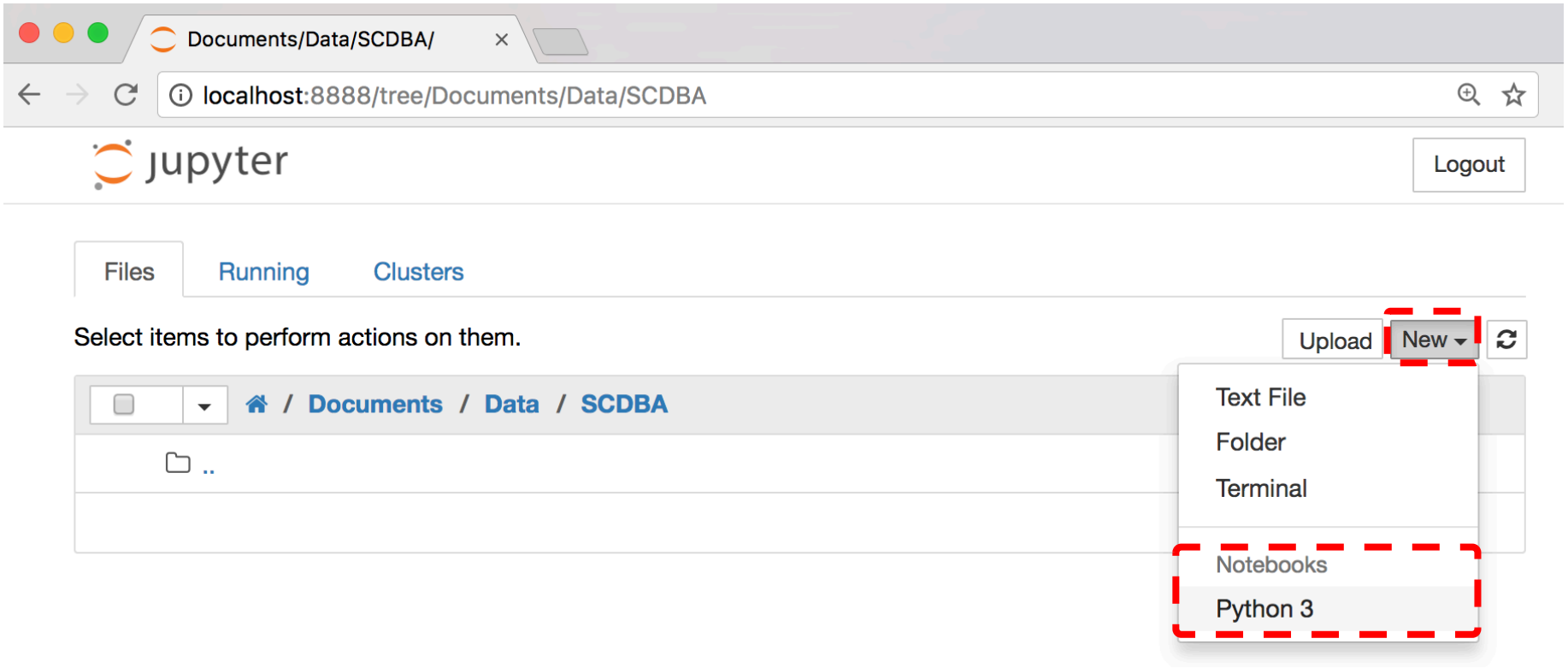


Anaconda-Navigator

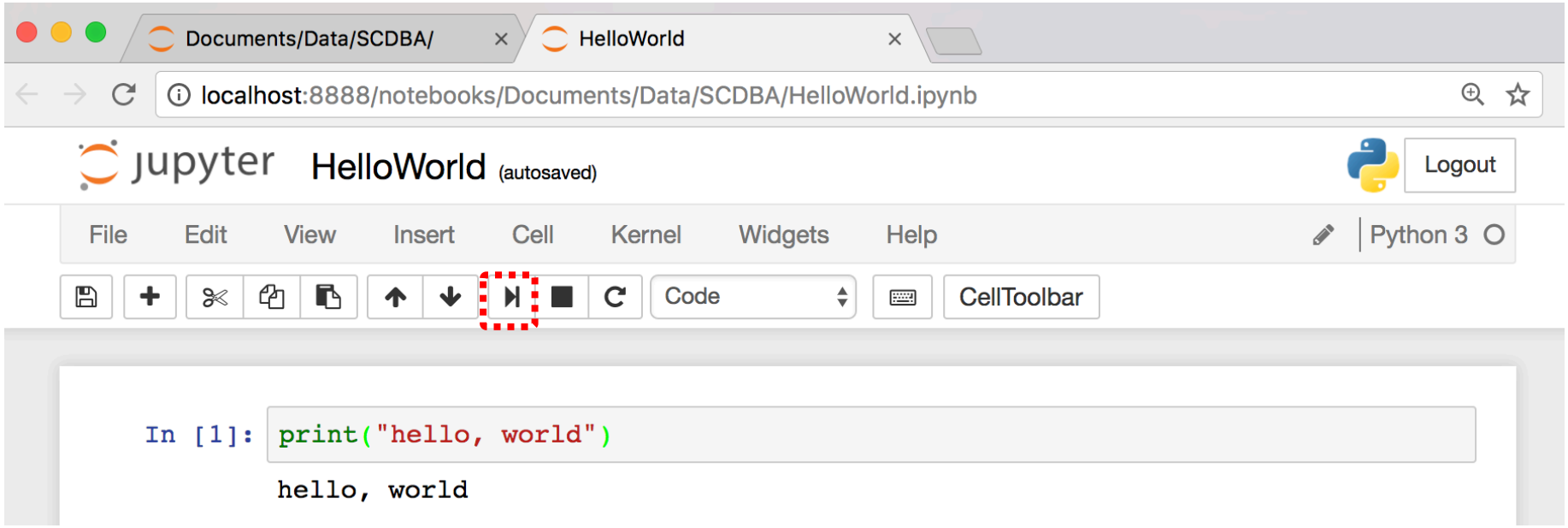
Jupyter Notebook



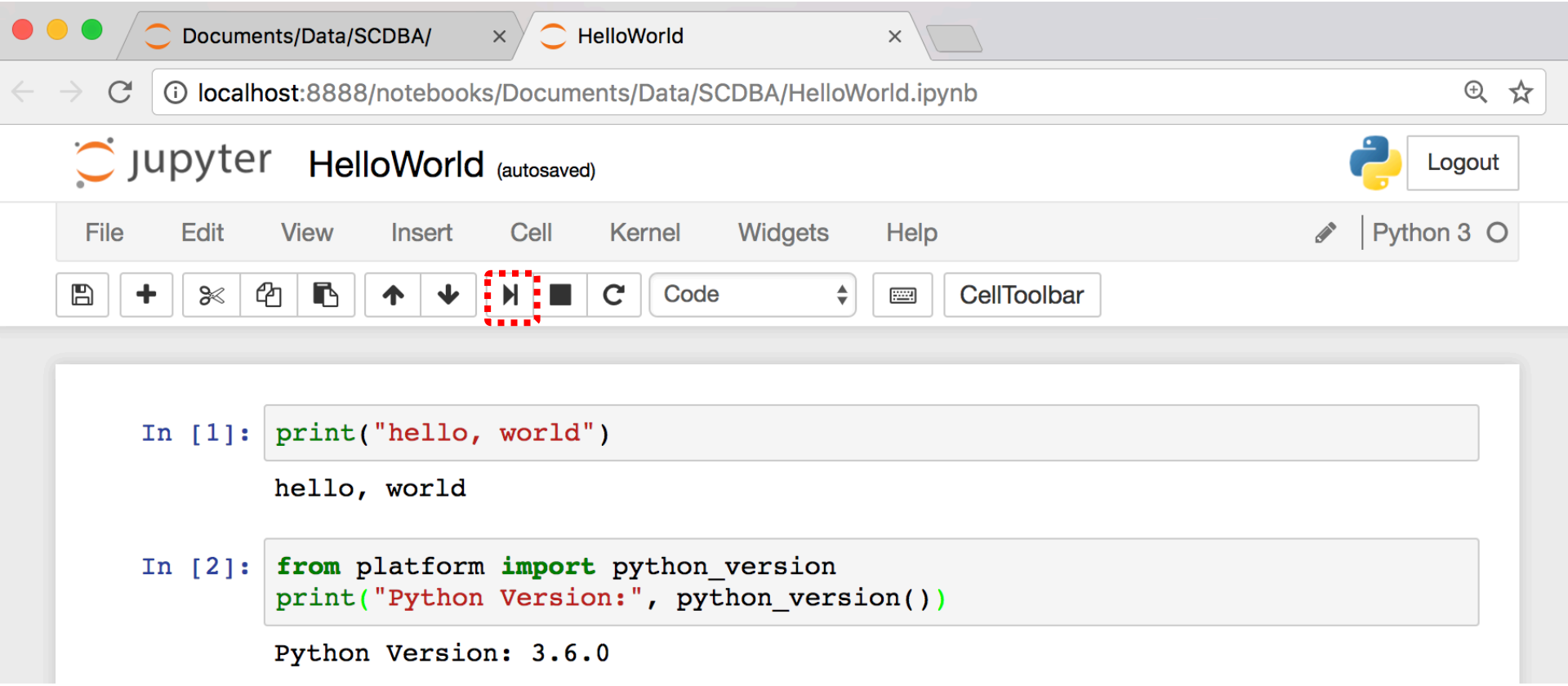
Jupyter Notebook
New Python 3



print("hello, world")



```
from platform import python_version
print("Python Version:", python_version())
```

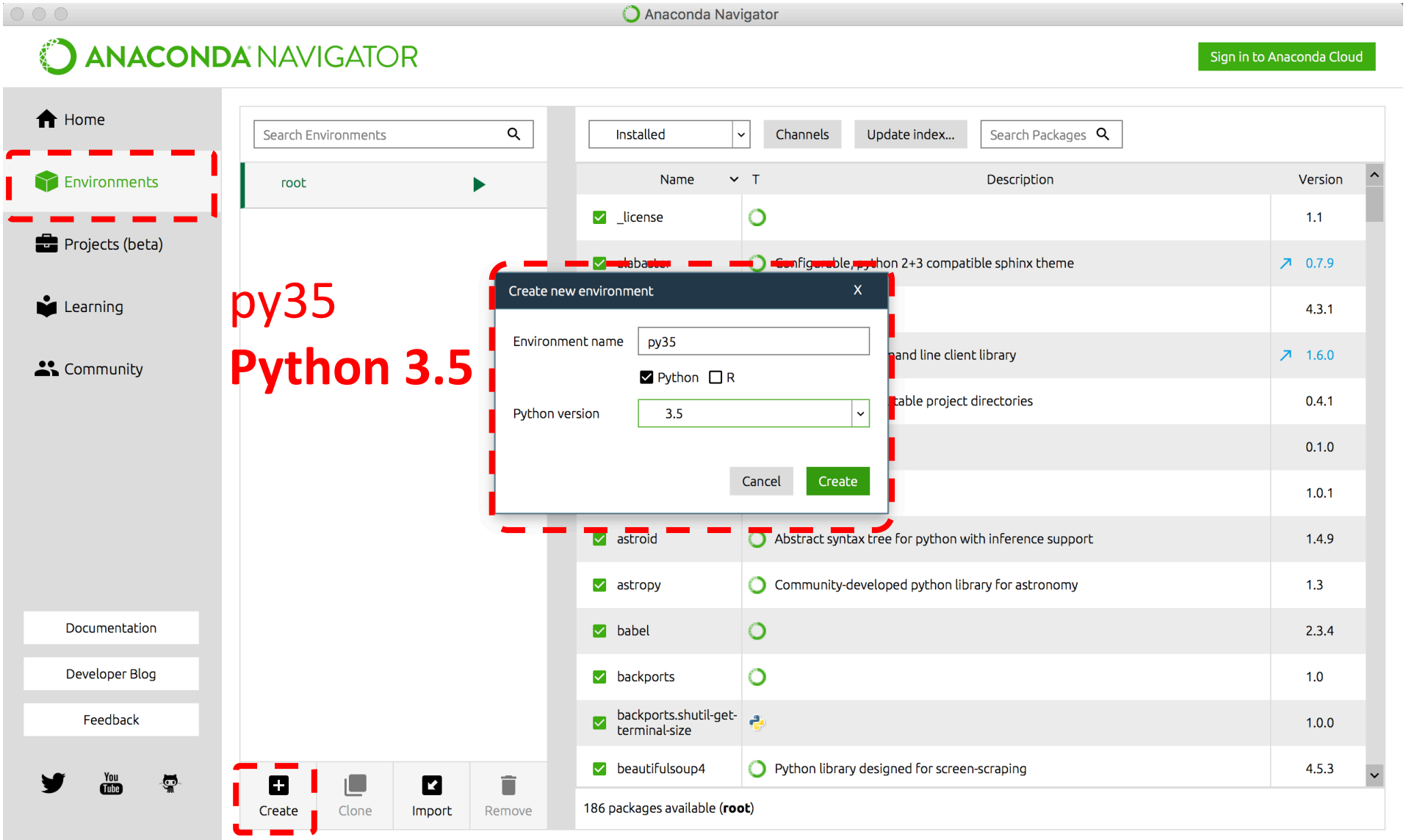


Create Python Environments with Anaconda

- Python 3.6
- Python 3.5
  - Python 3.5.3
  - Python 3.5.2
- Python 2.7

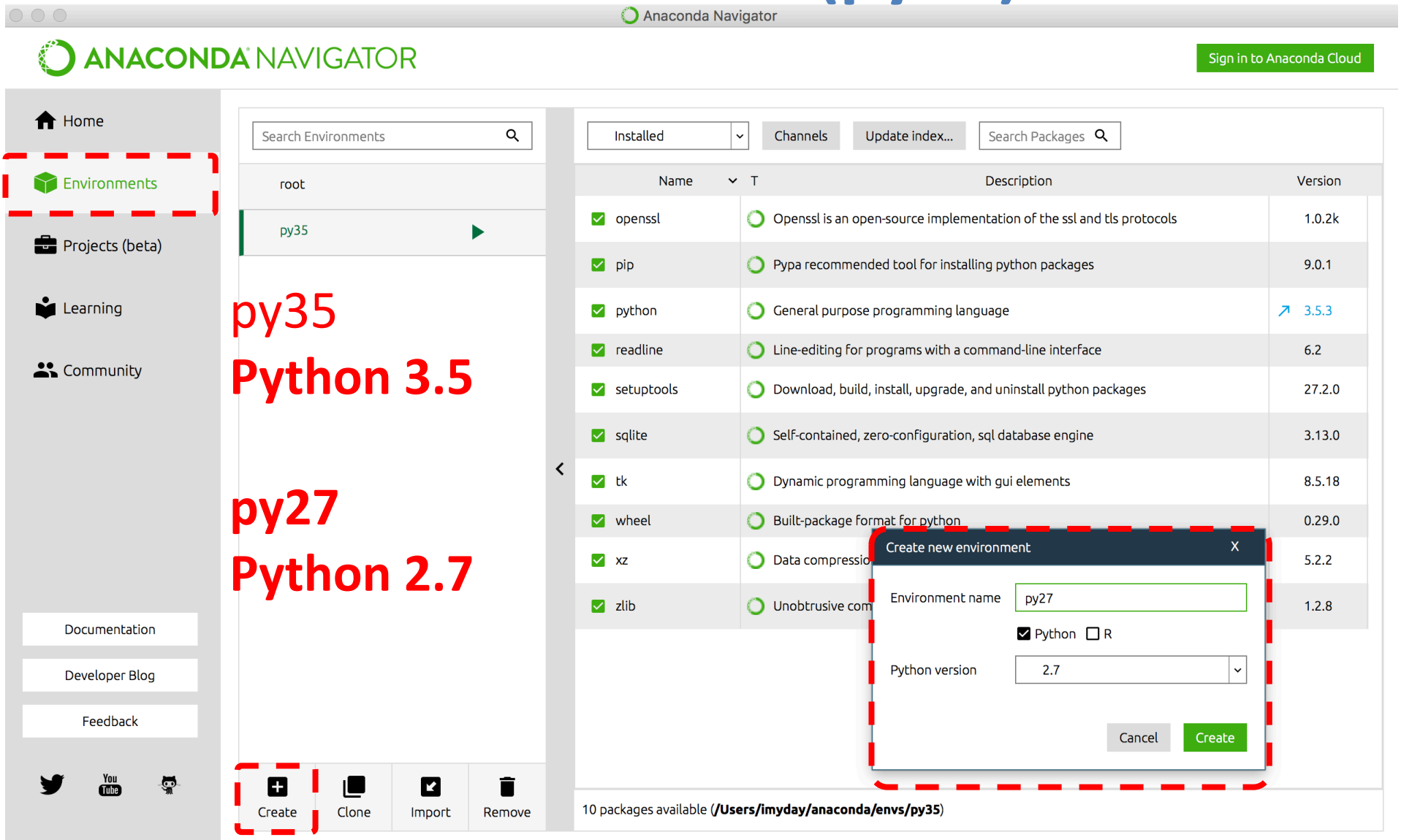
https://conda.io/docs/py2or3.html

Anaconda Create New Python 3.5 Environment (py35)



Source: http://conda.pydata.org/docs/py2or3.html

Anaconda Create New Python 2.7 Environment (py27)



Source: http://conda.pydata.org/docs/py2or3.html

Verify that conda is installed, check current conda version

- **`conda --version`**

- Update conda to the current version
  - **`conda update conda`**

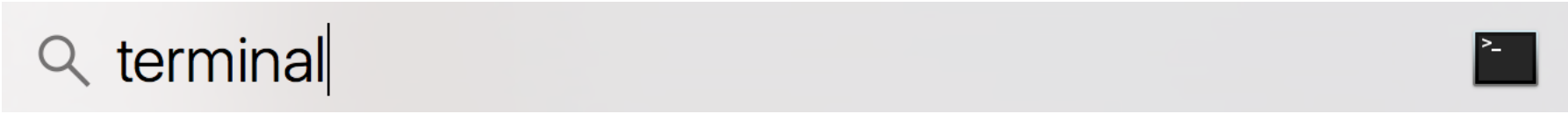
http://conda.pydata.org/docs/using/using.html#verify-that-conda-is-installed-check-current-conda-version

Check current conda version
Check current python version
Check conda environments

- `conda --version`
- `python --version`
- `conda info --envs`

http://conda.pydata.org/docs/using/using.html#verify-that-conda-is-installed-check-current-conda-version

Terminal




```
terminal
```

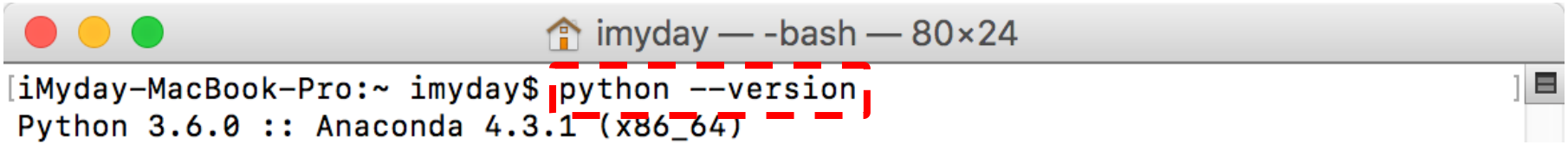
conda list

imyday — -bash — 80×24

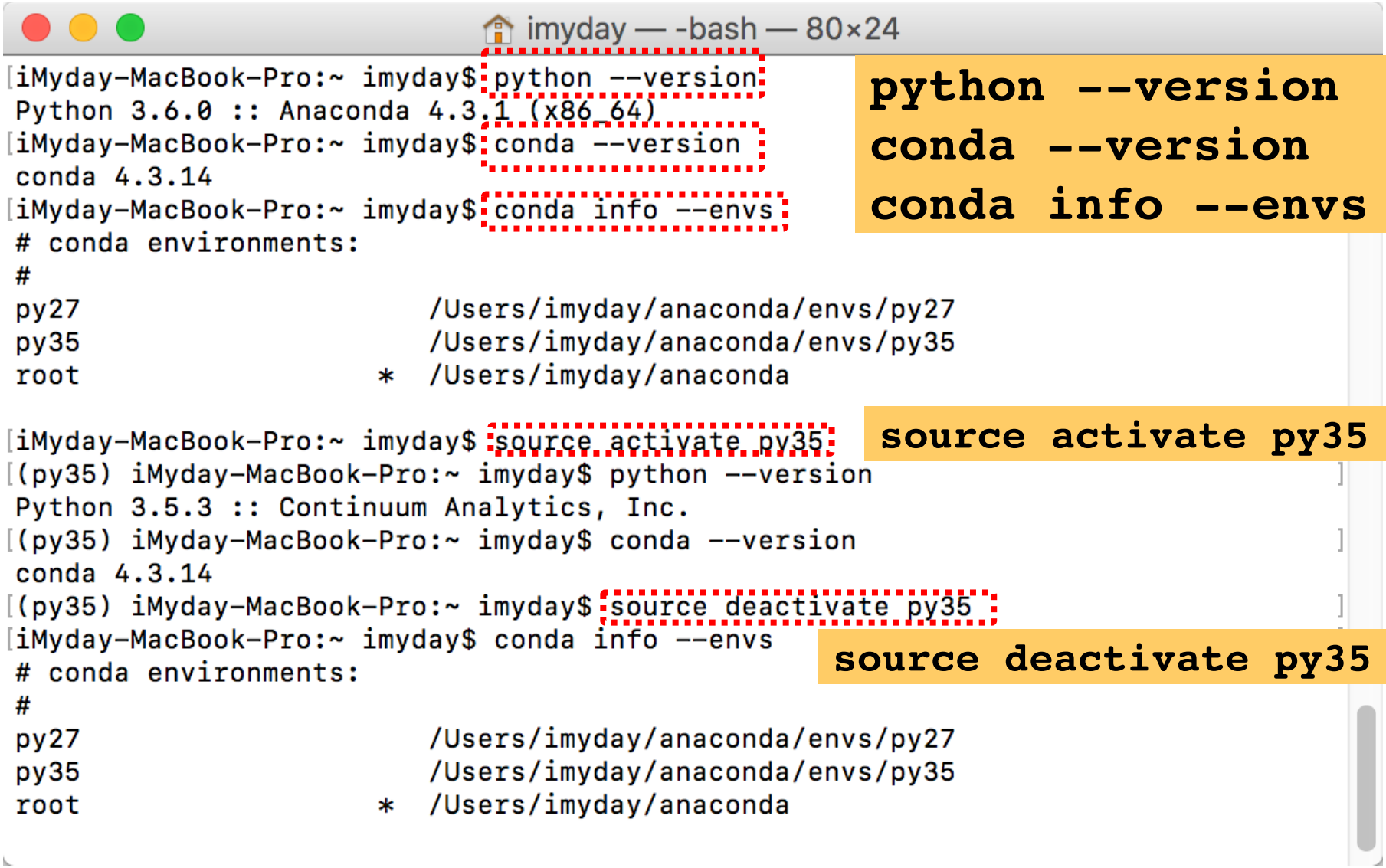
```
iMyday-MacBook-Pro:~ imyday$ conda list
# packages in environment at /Users/imyday/anaconda:
#
_license                  1.1                      py36_1
alabaster                 0.7.9                    py36_0
anaconda                  4.3.1                 np111py36_0
anaconda-client           1.6.0                    py36_0
anaconda-navigator        1.5.0                    py36_0
anaconda-project          0.4.1                    py36_0
appnope                   0.1.0                    py36_0
appscript                 1.0.1                    py36_0
astroid                   1.4.9                    py36_0
astropy                   1.3                   np111py36_0
babel                     2.3.4                    py36_0
backports                 1.0                      py36_0
beautifulsoup4            4.5.3                    py36_0
bitarray                  0.8.1                    py36_0
blaze                     0.10.1                   py36_0
bokeh                     0.12.4                   py36_0
boto                      2.45.0                   py36_0
bottleneck                1.2.0                 np111py36_0
cffi                      1.9.1                    py36_0
chardet                   2.3.0                    py36_0
chest                     0.2.3                    py36_0
```

http://conda.pydata.org/docs/using/using.html#verify-that-conda-is-installed-check-current-conda-version

python --version



conda --version

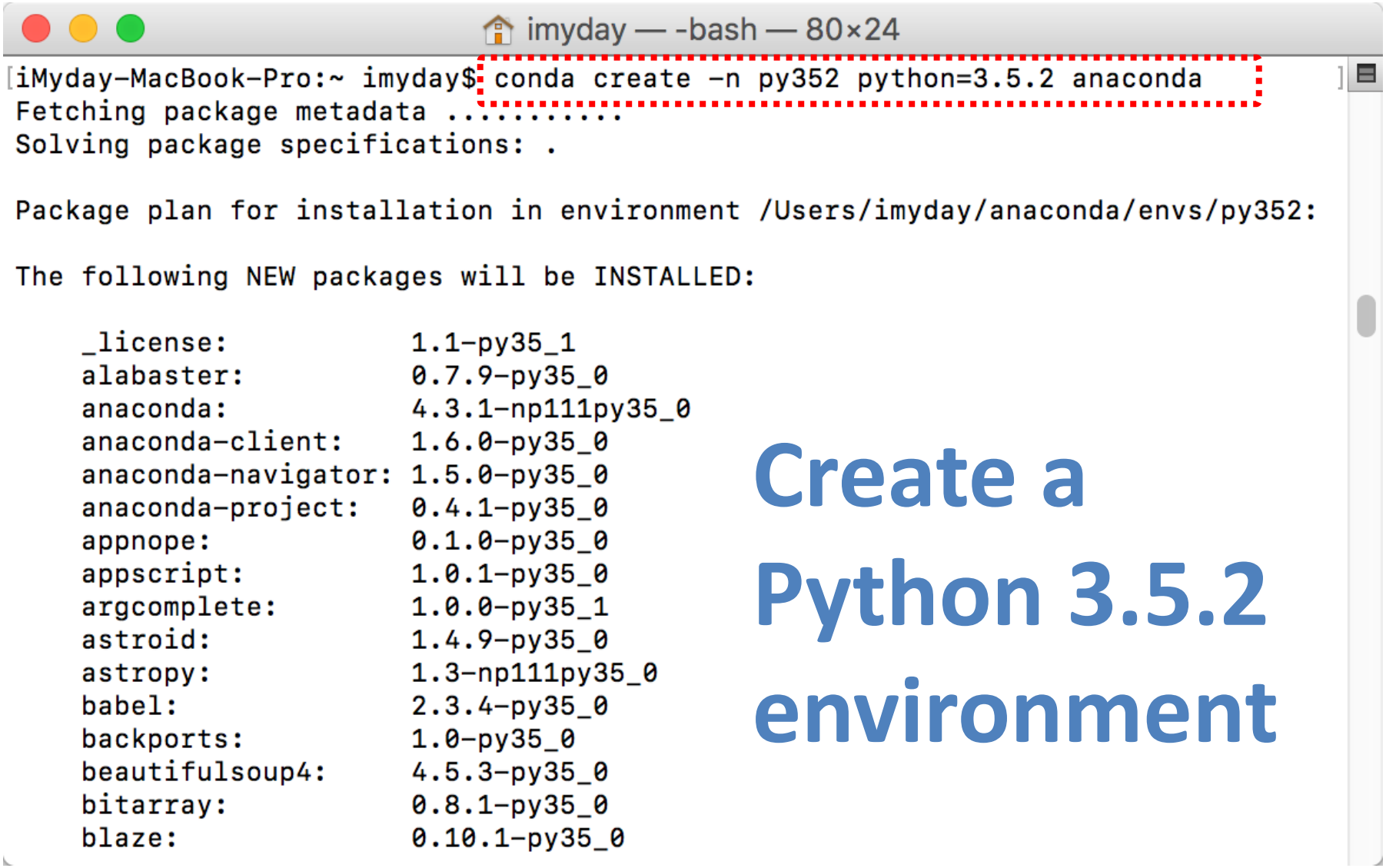


python --version
conda --version
conda info --envs

source activate py35

source deactivate py35

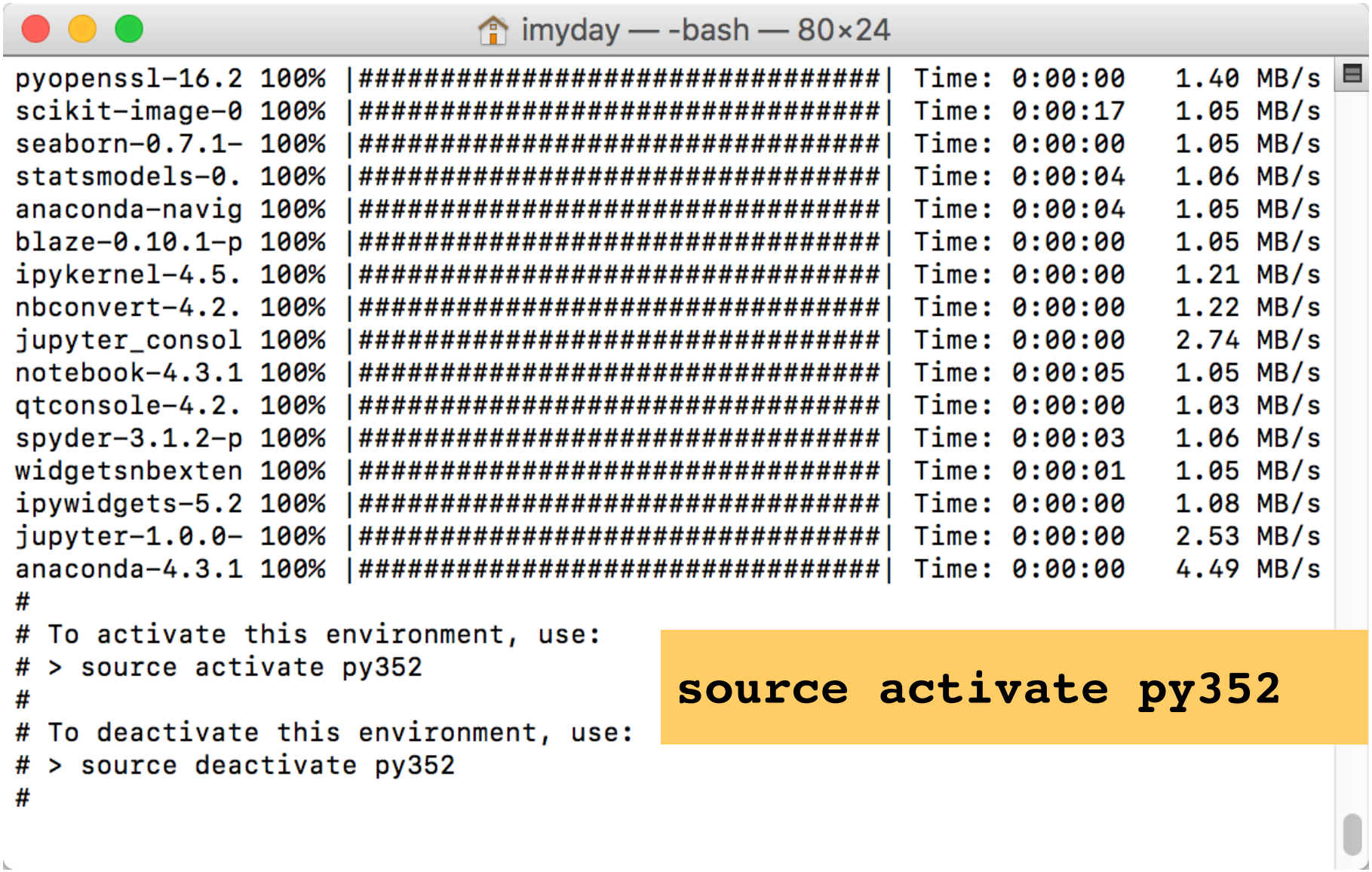
conda create -n py352 python=3.5.2 anaconda



Create a Python 3.5.2 environment

Source: http://conda.pydata.org/docs/py2or3.html

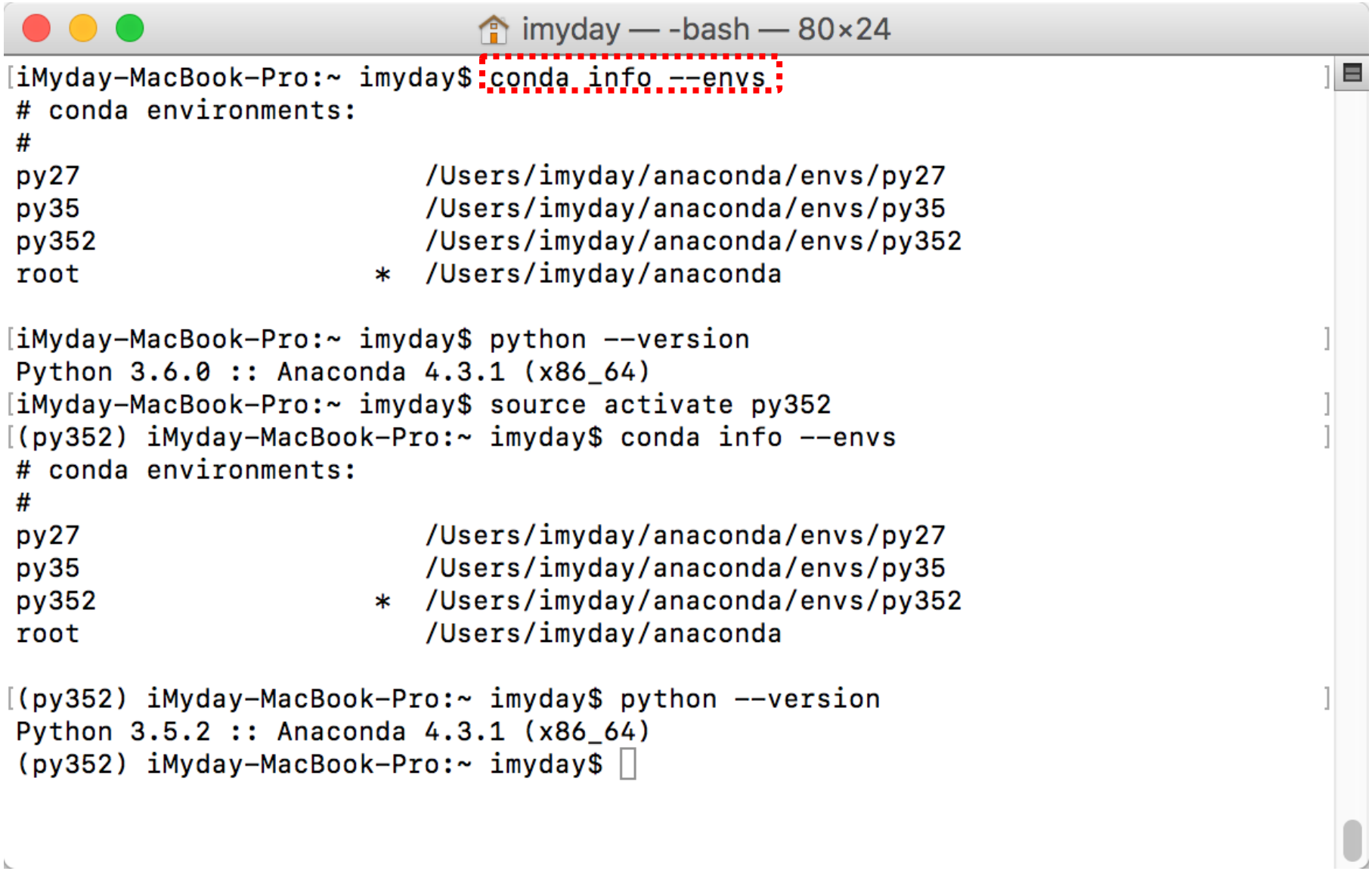
conda create -n py352 python=3.5.2 anaconda



source activate py352

Source: http://conda.pydata.org/docs/py2or3.html

conda info --envs



```
conda info --envs
source activate py27
python --version

conda install notebook ipykernel

jupyter notebook
```

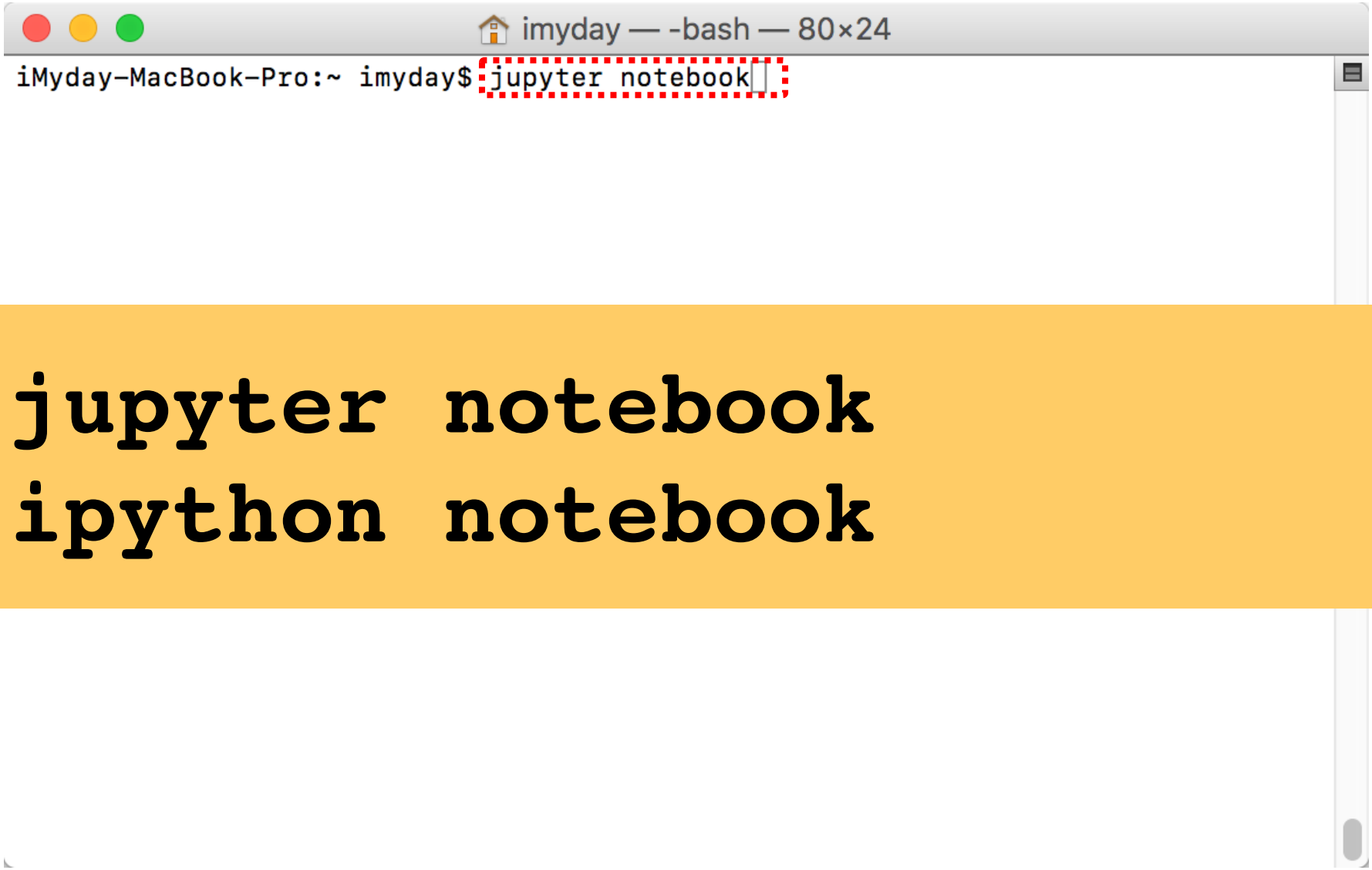
source activate py27
conda install notebook ipykernel

imyday — -bash — 80×24

```
iMyday-MacBook-Pro:~ imyday$ conda info --envs
# conda environments:
#
py27                     /Users/imyday/anaconda/envs/py27
py35                     /Users/imyday/anaconda/envs/py35
py352                    /Users/imyday/anaconda/envs/py352
root                  *  /Users/imyday/anaconda

iMyday-MacBook-Pro:~ imyday$ source activate py27
(py27) iMyday-MacBook-Pro:~ imyday$ python --version
Python 2.7.13 :: Continuum Analytics, Inc.
(py27) iMyday-MacBook-Pro:~ imyday$ conda install notebook ipykernel
Fetching package metadata ...........
Solving package specifications: .

Package plan for installation in environment /Users/imyday/anaconda/envs/py27:

The following NEW packages will be INSTALLED:

    appnope:             0.1.0-py27_0
    backports:           1.0-py27_0
    backports_abc:       0.5-py27_0
    bleach:              1.5.0-py27_0
    configparser:        3.5.0-py27_0
```

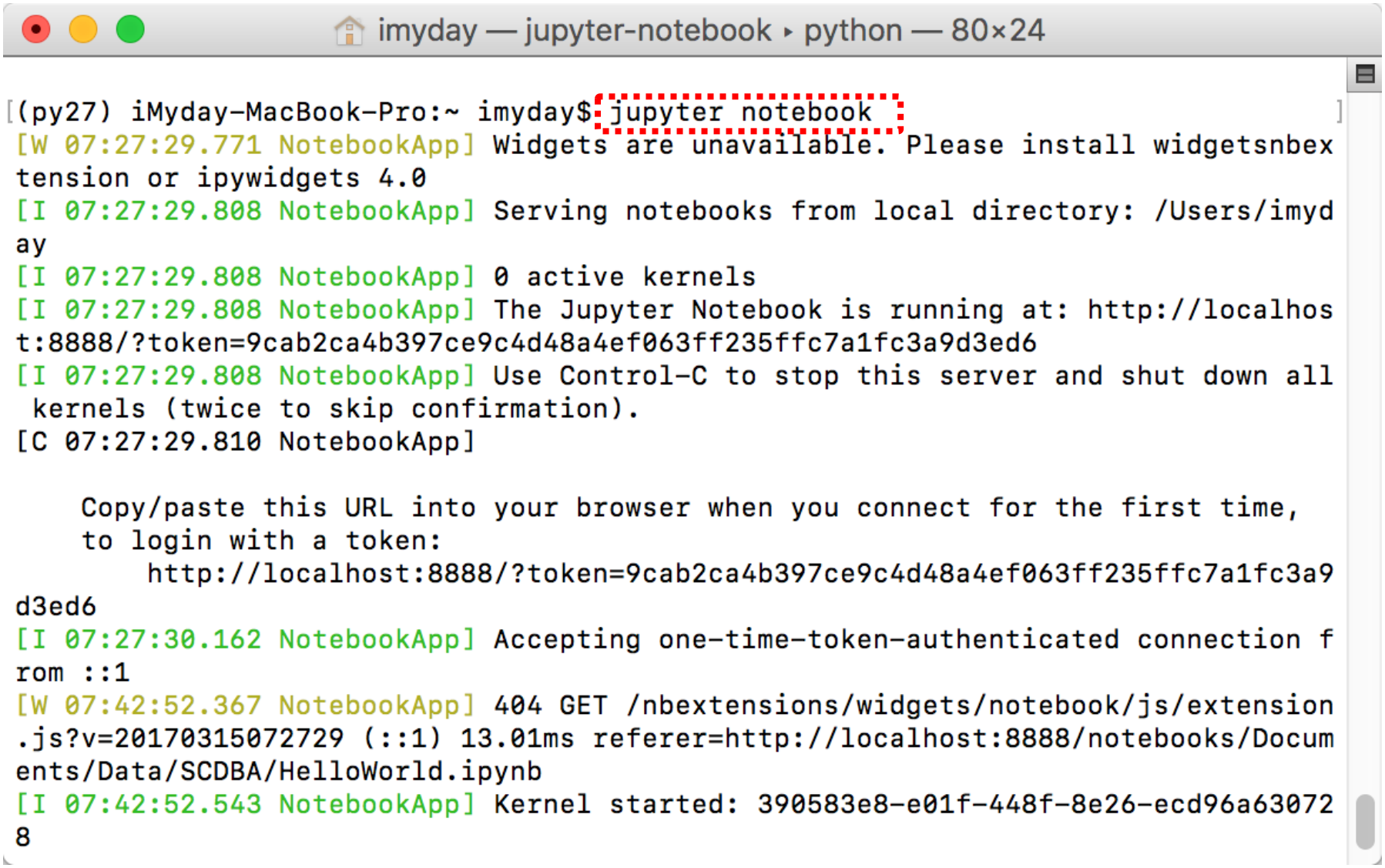
conda info --envs
source activate py27
python --version
conda install notebook ipykernel
jupyter notebook

jupyter notebook

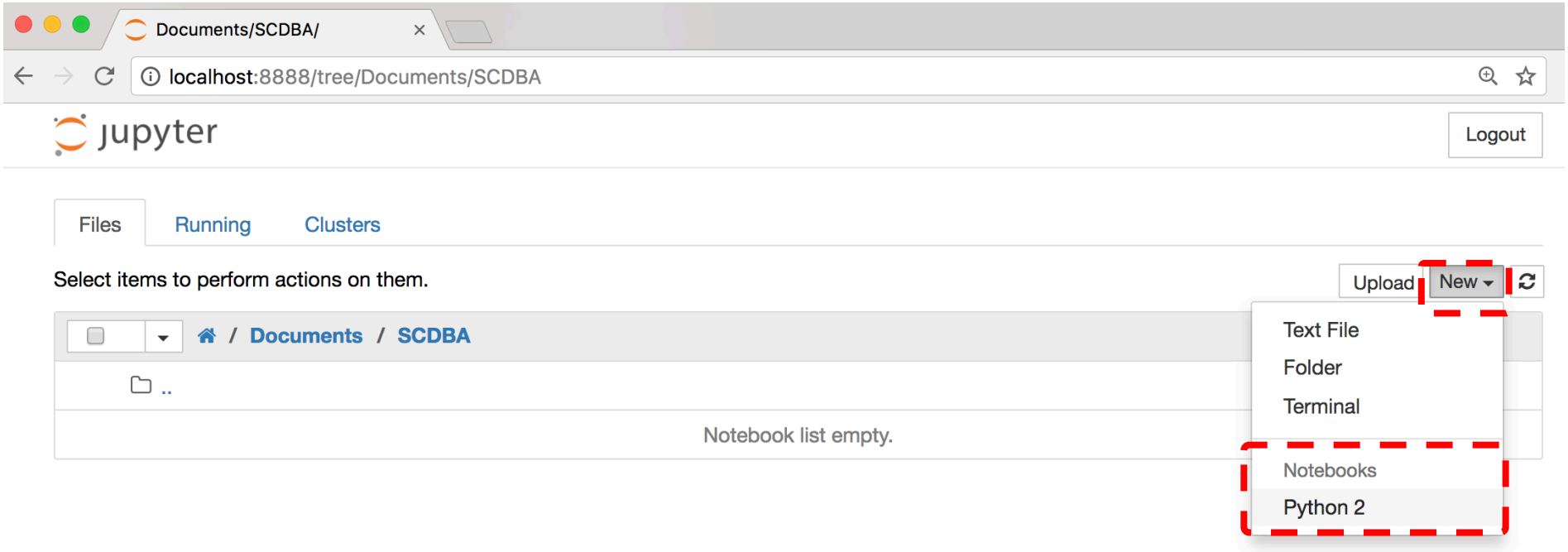


jupyter notebook
ipython notebook

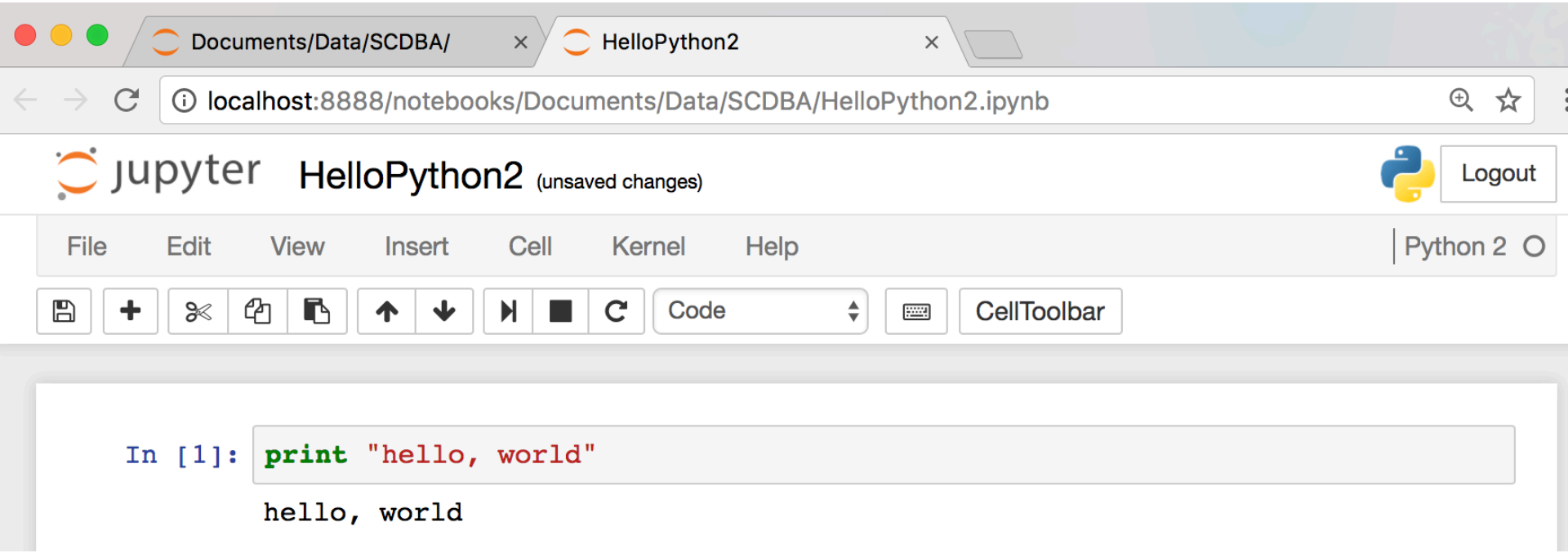
jupyter notebook



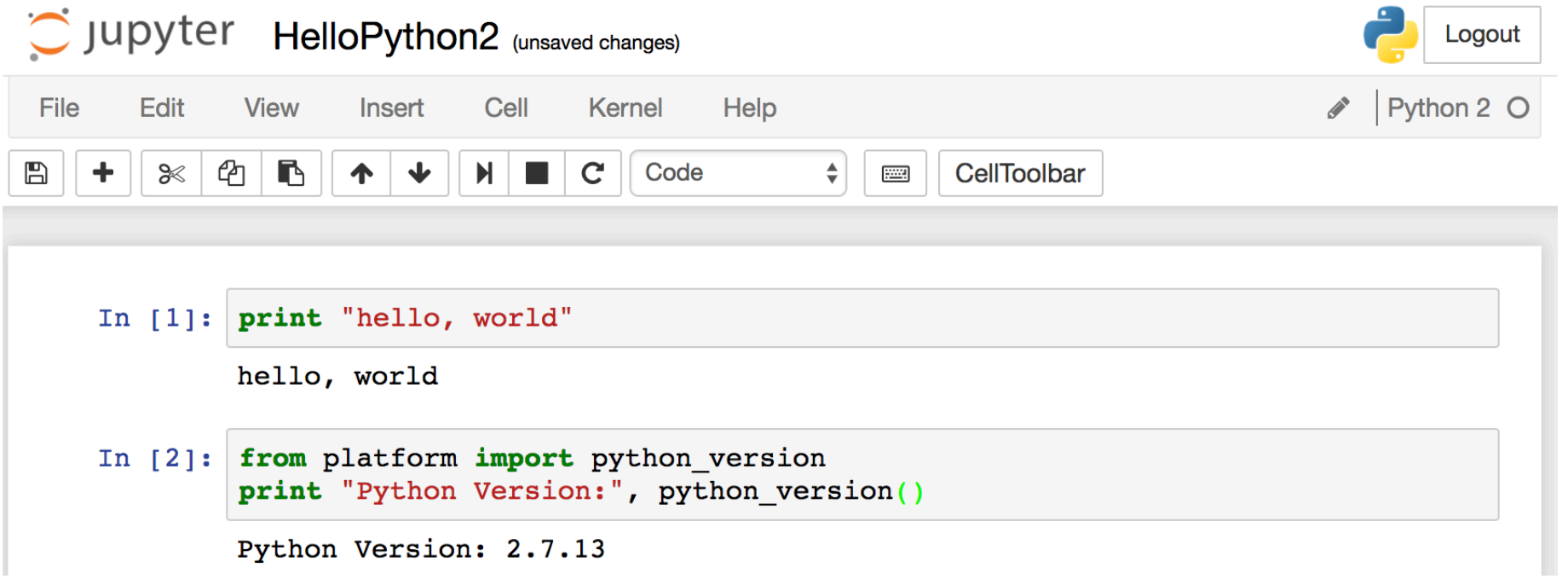
Jupyter Notebook
New Python 2



print “hello, world”



```
from platform import python_version
print "Python Version:", python_version()
```



```
jupyter notebook
```

```
ipython notebook
```

Source: https://www.python.org/community/logos/

Text input and output

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

```
print("Hello World\nThis is a message")
```

```
x = 3
print(x)
```

```
x = 2
y = 3
print(x, ' ', y)
```

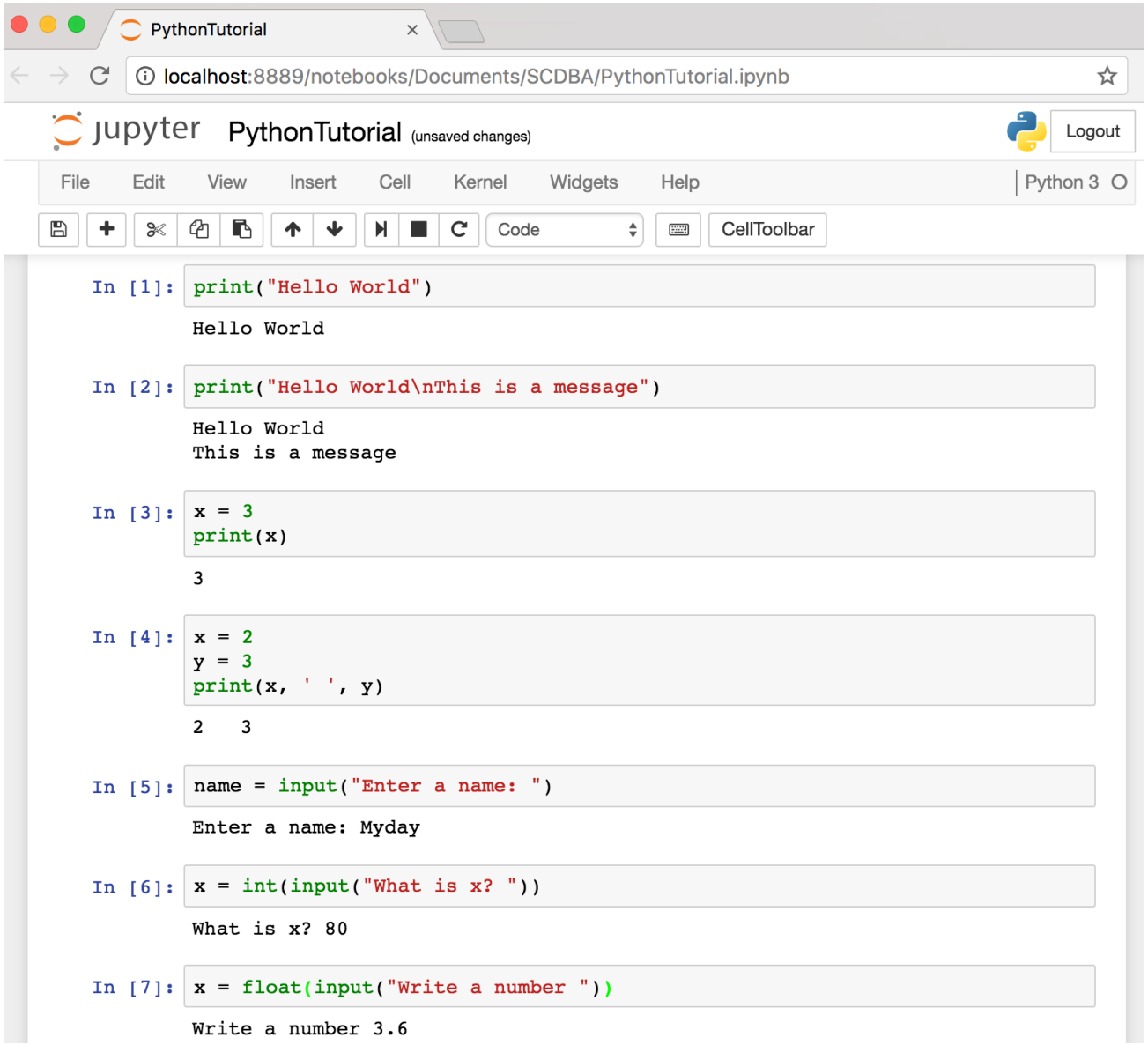
```
name = input("Enter a name: ")
```

```
x = int(input("What is x? "))
```

```
x = float(input("Write a number "))
```

Source: http://pythonprogramminglanguage.com/text-input-and-output/

Text input and output



Source: http://pythonprogramminglanguage.com/text-input-and-output/

Variables

```
x = 2
price = 2.5
word = 'Hello'
```

```
word = 'Hello'
word = "Hello"
word = '''Hello'''
```

```
x = 2
x = x + 1
x = 5
```

Source: http://pythonprogramminglanguage.com/

Python Basic Operators

```
print('7 + 2 =', 7 + 2)
print('7 - 2 =', 7 - 2)
print('7 * 2 =', 7 * 2)
print('7 / 2 =', 7 / 2)
print('7 // 2 =', 7 // 2)
print('7 % 2 =', 7 % 2)
print('7 ** 2 =', 7 ** 2)
```

```
print('7 + 2 =', 7 + 2)
print('7 - 2 =', 7 - 2)
print('7 * 2 =', 7 * 2)
print('7 / 2 =', 7 / 2)
print('7 // 2 =', 7 // 2)
print('7 % 2 =', 7 % 2)
print('7 ** 2 =', 7 ** 2)
```

```
7 + 2 = 9
7 - 2 = 5
7 * 2 = 14
7 / 2 = 3.5
7 // 2 = 3
7 % 2 = 1
7 ** 2 = 49
```

BMI Calculator in Python

```
height_cm = float(input("Enter your height in cm: "))
weight_kg = float(input("Enter your weight in kg: "))

height_m = height_cm/100
BMI = (weight_kg/(height_m**2))

print("Your BMI is: " + str(round(BMI,1)))
```

Source: http://code.activestate.com/recipes/580615-bmi-code/

BMI Calculator in Python

jupyter PythonTutorial Last Checkpoint: a minute ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Python 3

Code CellToolbar

In [1]:
```
height_cm = float(input("Enter your height in cm: "))
weight_kg = float(input("Enter your weight in kg: "))

height_m = height_cm/100
BMI = (weight_kg/(height_m**2))

print("Your BMI is: " + str(round(BMI,1)))
```

```
Enter your height in cm: 170
Enter your weight in kg: 60
Your BMI is: 20.8
```

In [ ]:

Source: http://code.activestate.com/recipes/580615-bmi-code/

If statements

> greater than
< smaller than
== equals
!= is not

```
score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail")
```

```
Pass
```


```
score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail")
```


Source: http://pythonprogramminglanguage.com/

For loops

```
for i in range(1,11):
    print(i)
```

```
1
2
3
4
5
6
7
8
9
10
```

Source: http://pythonprogramminglanguage.com/

For loops

```
for i in range(1,10):
    for j in range(1,10):
        print(i, ' * ' , j , ' = ', i*j)
```

```
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
```

Source: http://pythonprogramminglanguage.com/

Functions

```
def convertCMtoM(xcm):
    m = xcm/100
    return m

cm = 180
m = convertCMtoM(cm)
print(str(m))
```

1.8

Lists

```
x = [60, 70, 80, 90]
print(len(x))
print(x[0])
print(x[1])
print(x[-1])
```

```
4
60
70
90
```

Tuples

A tuple in Python is a collection that cannot be modified.

A tuple is defined using parenthesis.

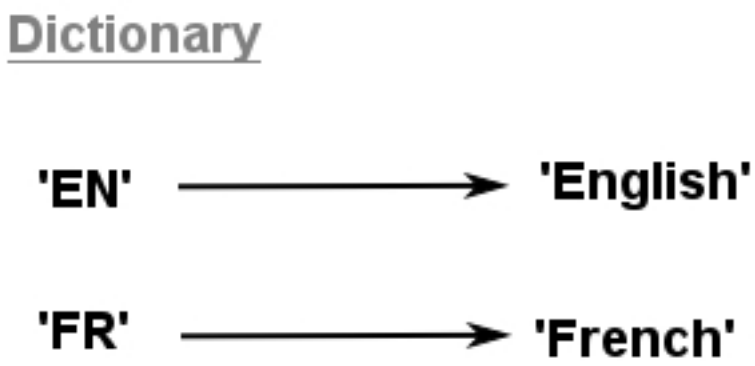
```
x = (10, 20, 30, 40, 50)
print(x[0])
print(x[1])
print(x[2])
print(x[-1])
```

```
10
20
30
50
```

Source: http://pythonprogramminglanguage.com/tuples/

Dictionary

```
k = { 'EN':'English', 'FR':'French' }
print(k['EN'])
```



English

Source: http://pythonprogramminglanguage.com/dictionary/

python™

Sets

```
animals = {'cat', 'dog'}
```

```
animals = {'cat', 'dog'}
print('cat' in animals)  # Check if an element is in a set; prints "True"
print('fish' in animals) # prints "False"
animals.add('fish')      # Add an element to a set
print('fish' in animals) # Prints "True"
print(len(animals))      # Number of elements in a set; prints "3"
animals.add('cat')       # Adding an element that is already in the set does nothing
print(len(animals))      # Prints "3"
animals.remove('cat')    # Remove an element from a set
print(len(animals))      # Prints "2"
```

```
True
False
True
3
3
2
```

```
animals = {'cat', 'dog'}
print('cat' in animals)
print('fish' in animals)
animals.add('fish')
print('fish' in animals)
print(len(animals))
animals.add('cat')
print(len(animals))
animals.remove('cat')
print(len(animals))
```

Source: http://cs231n.github.io/python-numpy-tutorial/

Python Ecosystem

Python Ecosystem
import math

```
x = log(1)
print(x)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-64-55d85b4998db> in <module>()
----> 1 x = log(1)
      2 print(x)

NameError: name 'log' is not defined
```

math.log?

```
import math
x = math.log(1)
print(x)
```

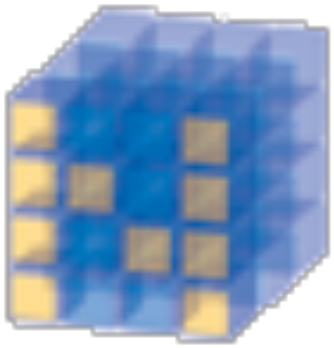
```
0.0
```

```
math.log(8,2)
```

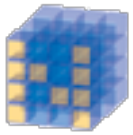
```
3.0
```

```
Docstring:
log(x[, base])

Return the logarithm of x to the given base.
If the base not specified, returns the natural logarithm (base e) of x.
Type:      builtin_function_or_method
```

Numpy



NumPy
Base
N-dimensional array
package

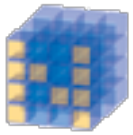


NumPy

NumPy

- NumPy provides a multidimensional array object to store homogenous or heterogeneous data; it also provides optimized functions/methods to operate on this array object.

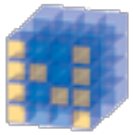
Source: Yves Hilpisch (2014), Python for Finance: Analyze Big Financial Data, O'Reilly



NumPy

NumPy

```
v = range(1, 6)
print(v)
2 * v
import numpy as np
v = np.arange(1, 6)
v
2 * v
```



NumPy

Base
N-dimensional
array package

```
v = range (1, 6)
print(v)
```

```
[1, 2, 3, 4, 5]
```

```
2 * v
```

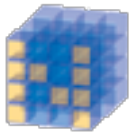
```
[1, 2, 3, 4, 5, 1, 2, 3, 4, 5]
```

```
import numpy as np
v = np.arange(1, 6)
v
```

```
array([1, 2, 3, 4, 5])
```

```
2 * v
```

```
array([ 2,  4,  6,  8, 10])
```



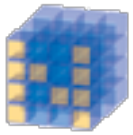
NumPy

NumPy

```
import numpy as np
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
c = a * b
c
```

```
import numpy as np
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
c = a * b
c
```

```
array([ 4, 10, 18])
```



NumPy

NumPy

```python
import numpy as np

a = np.zeros((2,2))  # Create an array of all zeros
print(a)             # Prints "[[ 0.  0.]
                     #          [ 0.  0.]]"

b = np.ones((1,2))   # Create an array of all ones
print(b)             # Prints "[[ 1.  1.]]"

c = np.full((2,2), 7) # Create a constant array
print(c)               # Prints "[[ 7.  7.]
                       #          [ 7.  7.]]"

d = np.eye(2)        # Create a 2x2 identity matrix
print(d)             # Prints "[[ 1.  0.]
                     #          [ 0.  1.]]"

e = np.random.random((2,2)) # Create an array filled with random values
print(e)                    # Might print "[[ 0.91940167  0.08143941]
                            #               [ 0.68744134  0.87236687]]"
```

```
[[ 0.  0.]
 [ 0.  0.]]
[[ 1.  1.]]
[[ 7.  7.]
 [ 7.  7.]]
[[ 1.  0.]
 [ 0.  1.]]
[[ 0.22886991  0.68473232]
 [ 0.20683825  0.16589995]]
```

Source: http://cs231n.github.io/python-numpy-tutorial/

Compatible Python 2 and Python 3 Code

- print()
- Exceptions
- Division
- Unicode strings
- Bad imports

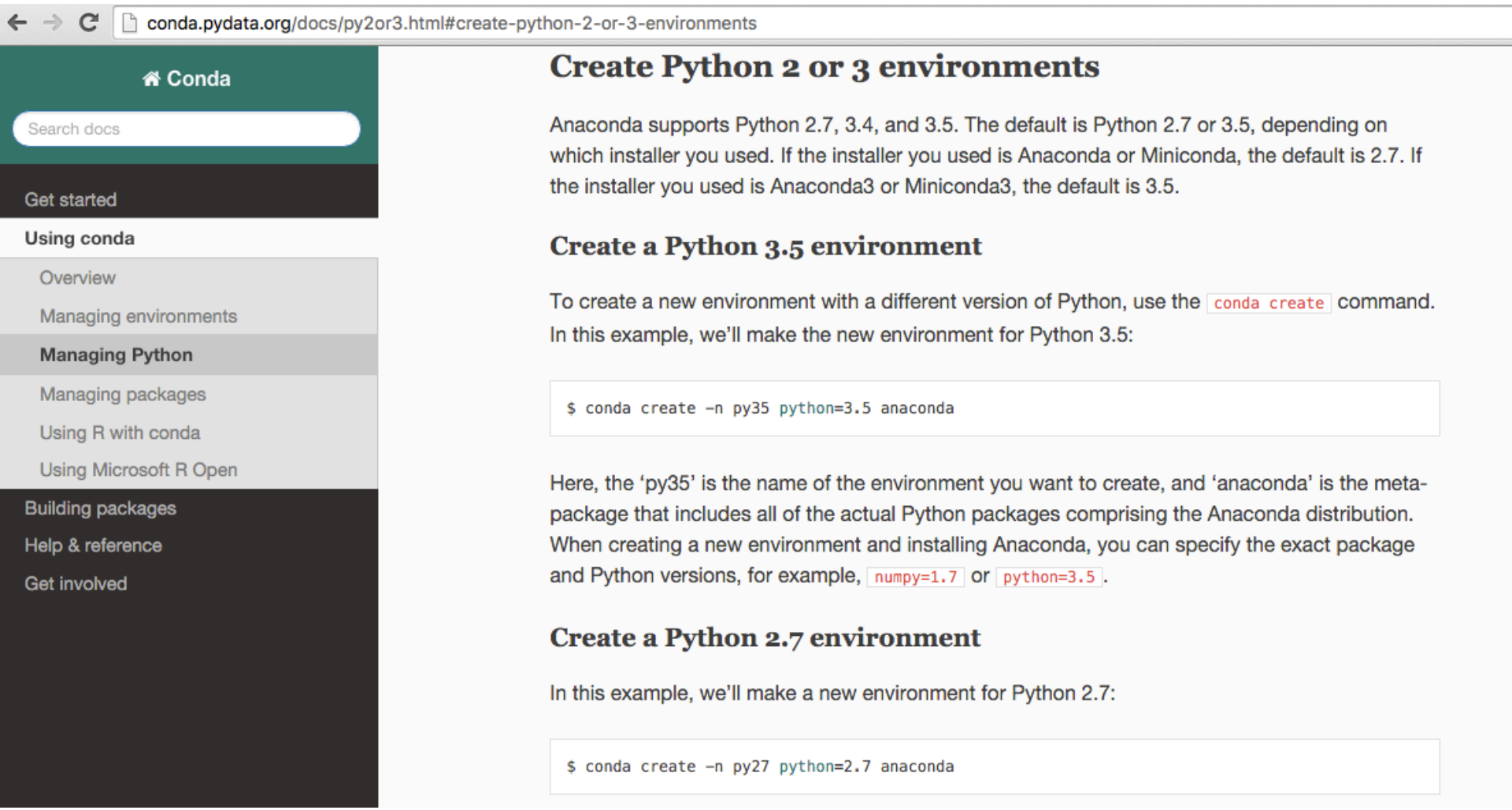
Source: DrapsTV (2016), Compatible Python 2 & 3 Code
https://www.youtube.com/watch?v=5Pwc-Rd4qJA

Compatible Python 2 and Python 3 Code

```
print()
print("This works in py2 and py3")

from __future__ import print_function
print("Hello", "World")
```

Source: DrapsTV (2016), Compatible Python 2 & 3 Code
https://www.youtube.com/watch?v=5Pwc-Rd4qJA

Create Python 2 or 3 environments



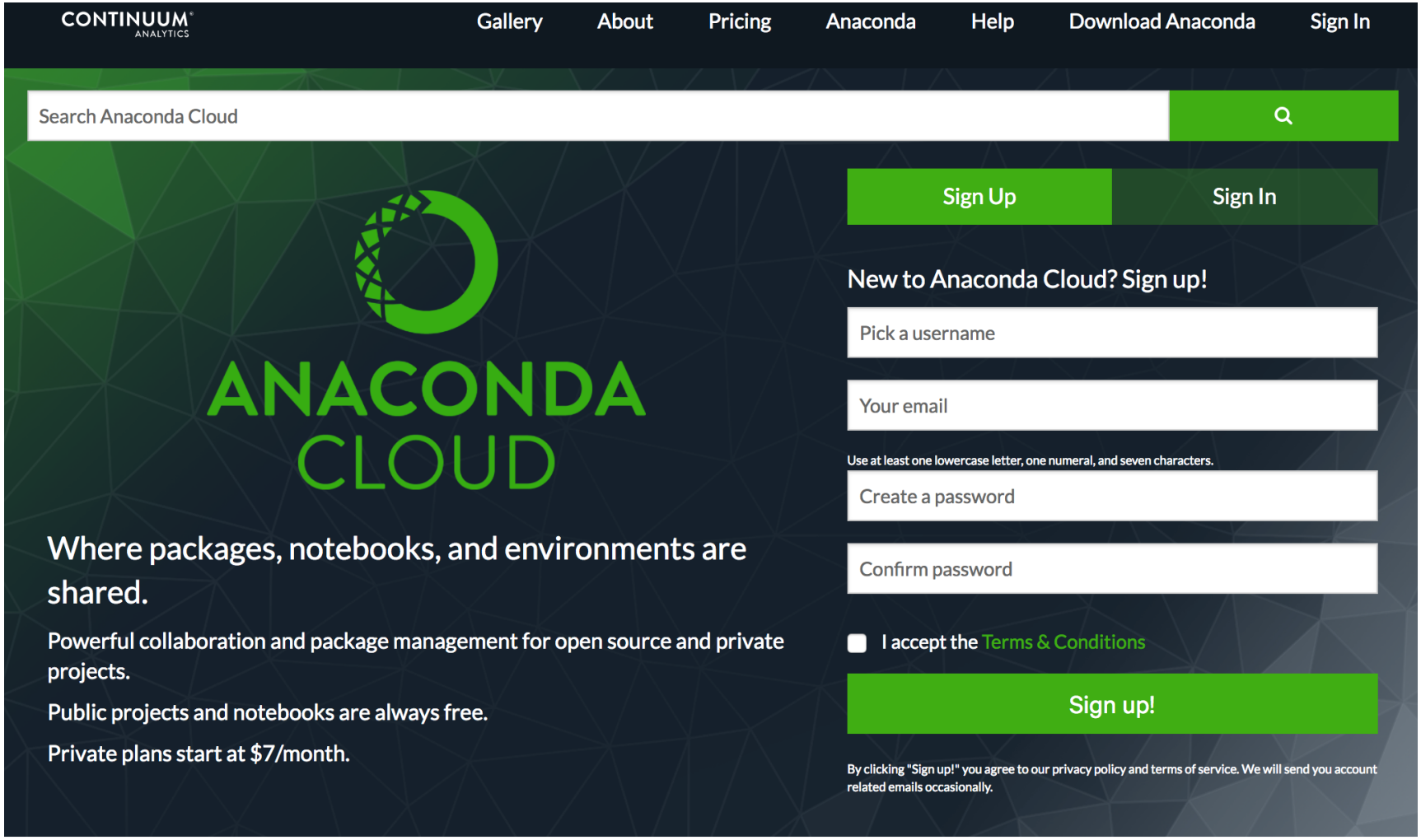
Source: http://conda.pydata.org/docs/py2or3.html#create-python-2-or-3-environments

File IO with open()

```
# Python 2 only
f = open('myfile.txt')
data = f.read()                 # as a byte string
text = data.decode('utf-8')

# Python 2 and 3: alternative 1
from io import open
f = open('myfile.txt', 'rb')
data = f.read()                 # as bytes
text = data.decode('utf-8')  # unicode, not bytes

# Python 2 and 3: alternative 2
from io import open
f = open('myfile.txt', encoding='utf-8')
text = f.read()    # unicode, not bytes
```

https://github.com/PythonCharmers/python-future/blob/master/docs/notebooks/Writing%20Python%202-3%20compatible%20code.ipynb

Anaconda Cloud

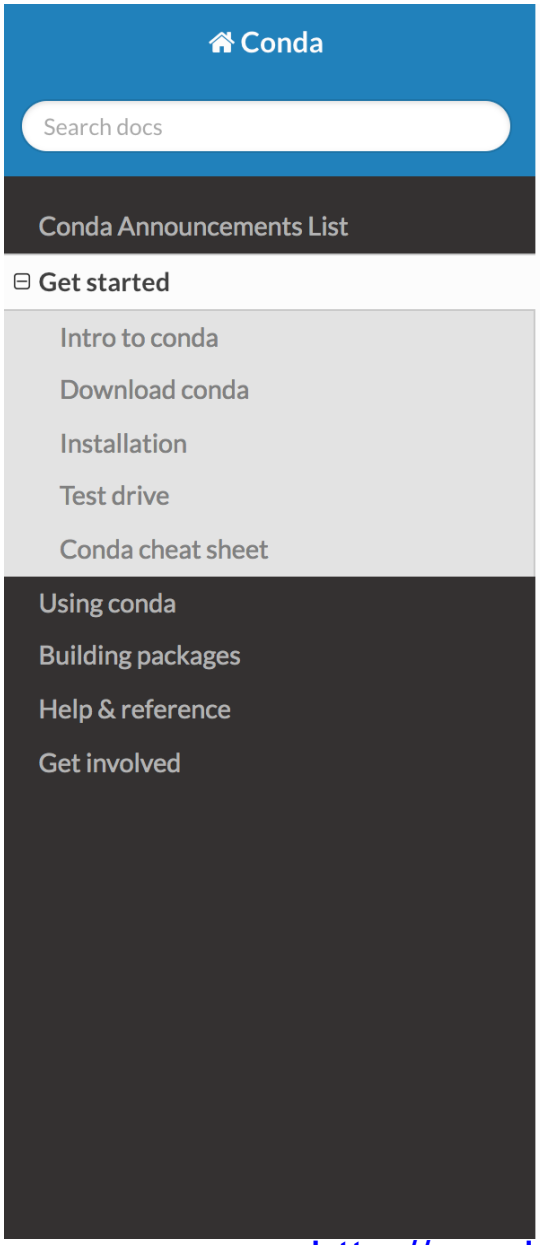


Why you'll love Anaconda Cloud

Making it easy to share packages, notebooks, and environments to be more collaborative.

https://anaconda.org/

Conda Get-Started

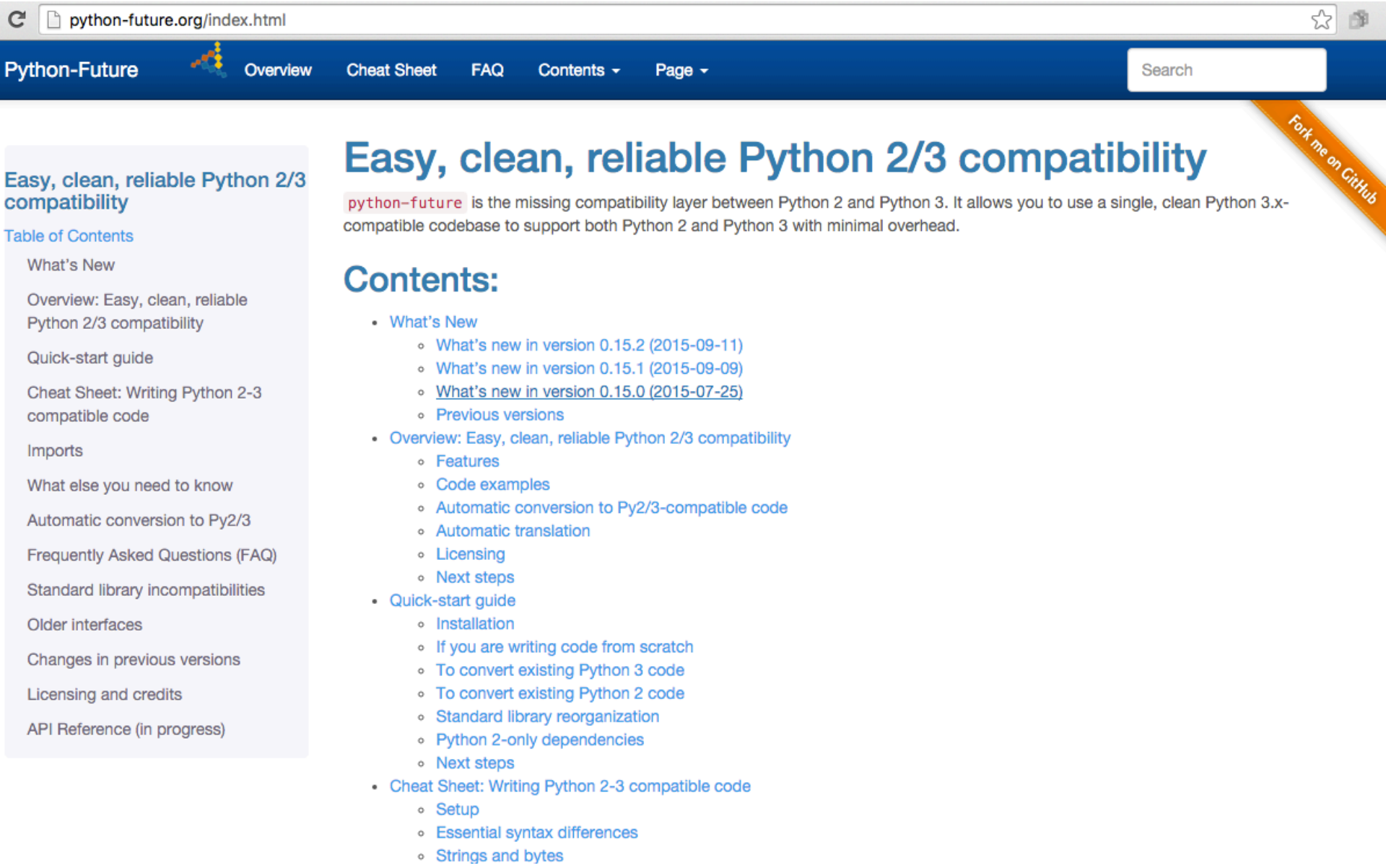


Get started

- Intro to conda
- Download conda
  - Should I download Anaconda or Miniconda?
  - Which version of Anaconda or Miniconda should I choose?
  - Should I choose GUI installer or command line installer?
  - What version of Python should I choose?
  - What about cryptographic hash verification?
- Installation
  - Quick install
    - Miniconda quick install requirements
    - Windows Miniconda install
    - OS X Miniconda install
    - Linux Miniconda install
  - Full installation
    - Anaconda requirements
    - Install instructions
      - Windows Anaconda install
      - OS X Anaconda install
      - Linux Anaconda install
  - Configuration
    - The conda configuration file (.condarc)
    - General configuration

http://conda.pydata.org/docs/get-started.html

Python–Future



http://python-future.org/index.html

pip install future
pip install six

The imports below refer to these pip-installable packages on PyPI:

```
import future           # pip install future
import builtins         # pip install future
import past             # pip install future
import six              # pip install six
```

```
futurize                # pip install future
pasteurize              # pip install future
```

http://python-future.org/compatible_idioms.html

print

```
# Python 2 only:
print 'Hello'
```

```
# Python 2 and 3:
print('Hello')
```

```
# Python 2 only:
print 'Hello', 'Guido'
```

```
# Python 2 and 3:
from __future__ import print_function #(at top of module)

print('Hello', 'Guido')
```

http://python-future.org/compatible_idioms.html

Writing Python 2-3 compatible code
Essential syntax differences

print

```
# Python 2 only:
print 'Hello'
```

```
# Python 2 and 3:
print('Hello')
```

To print multiple strings, import `print_function` to prevent Py2 from interpreting it as a tuple:

```
# Python 2 only:
print 'Hello', 'Guido'
```

```
# Python 2 and 3:
from __future__ import print_function    # (at top of module)

print('Hello', 'Guido')
```

http://python-future.org/compatible_idioms.html

Unicode (text) string literals

```
# Python 2 only
s1 = 'The Zen of Python'
s2 = u'きたないのよりきれいな方がいい\n'

# Python 2 and 3
s1 = u'The Zen of Python'
s2 = u'きたないのよりきれいな方がいい\n'
```

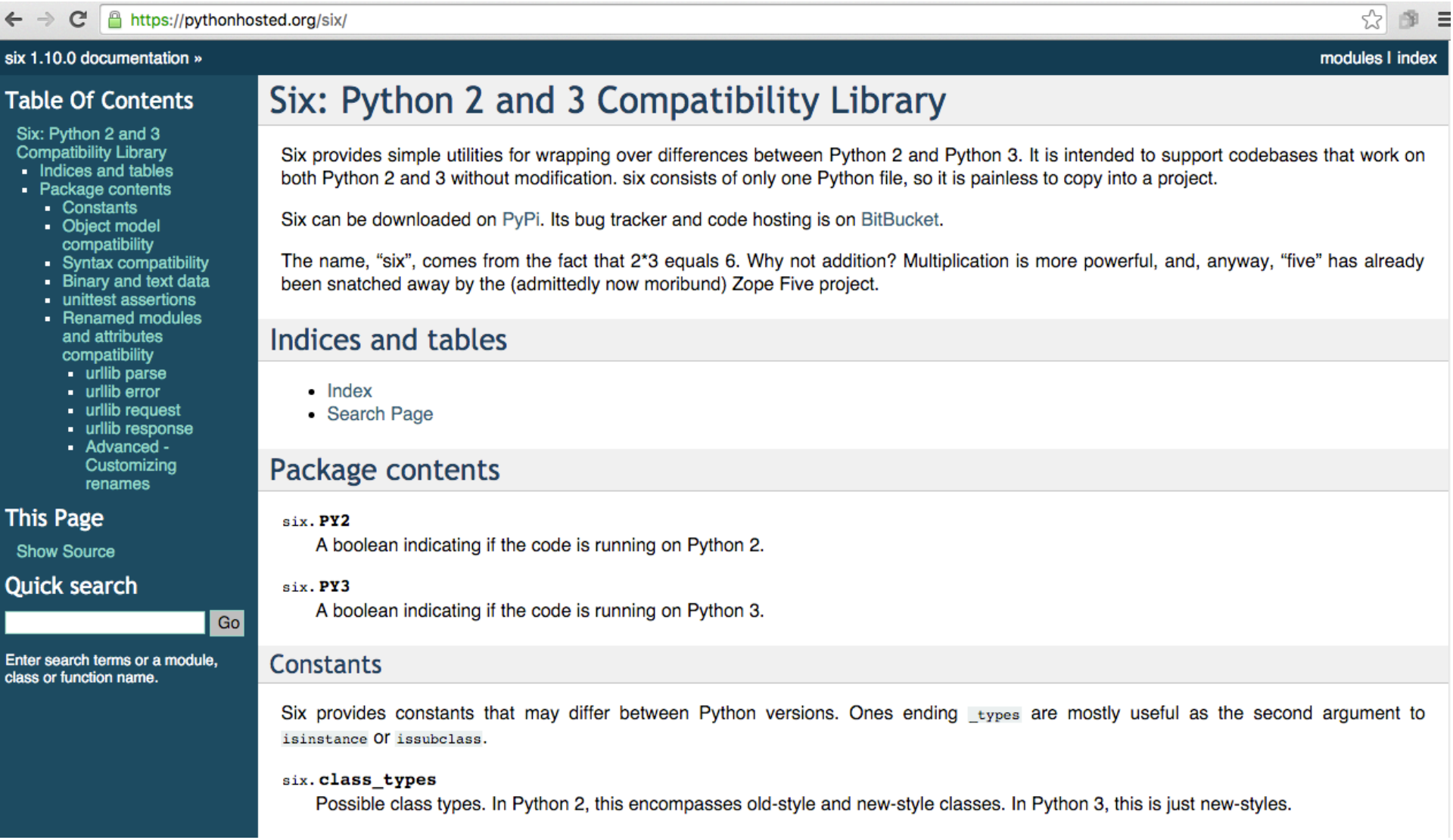
http://python-future.org/compatible_idioms.html

Unicode (text) string literals

```
# Python 2 and 3
from __future__ import unicode_literals # at top of module

s1 = 'The Zen of Python'
s2 = 'きたないのよりきれいな方がいい\n'
```

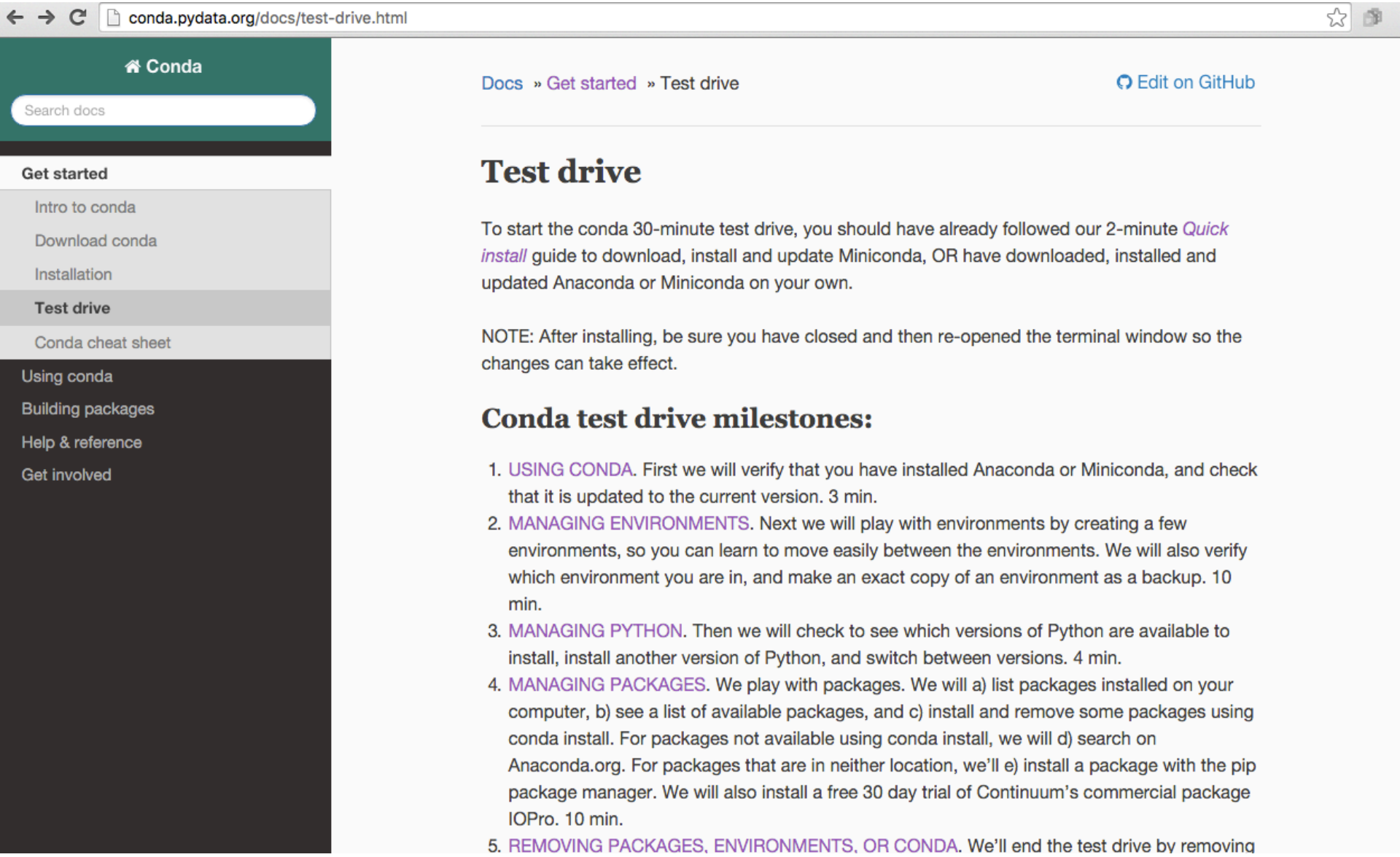
http://python-future.org/compatible_idioms.html

Six: Python 2 and 3 Compatibility Library



https://pythonhosted.org/six/

Conda Test Drive



http://conda.pydata.org/docs/test-drive.html

Managing Conda and Anaconda

Managing conda and anaconda

`conda info`	Verify conda is installed, check version #
`conda update conda`	Update conda package and environment manager to current version
`conda update an aconda`	Update the anaconda meta package (the library of packages ready to install with `conda` command)

http://conda.pydata.org/docs/_downloads/conda-cheatsheet.pdf

Managing environments

Managing environments

conda info --envs *or* **conda info -e**	Get a list of all my environments, active environment shown with *
conda create --name snowflakes biopython *or* **conda create -n snowflakes biopython**	Create an environment and install program(s) ***TIP:*** *To avoid dependency conflicts, install all programs in the environment (snowflakes) at the same time.* ***TIP:*** *Environments install by default into the envs directory in your conda directory. You can specify a different path; see* **conda create --help** *for details.*
source activate snowflakes *(Linux, Mac)* **activate snowflakes** *(Windows)*	Activate the new environment to use it ***TIP:*** *Activate prepends the path to the snowflakes environment.*
conda create -n bunnies python=3.4 astroid	Create a new environment, specify Python version
conda create -n flowers --clone snowflakes	Make exact copy of an environment
conda remove -n flowers --all	Delete an environment
conda env export > puppies.yml	Save current envirinment to a file
conda env create -f puppies.yml	Load environment fromm a file

http://conda.pydata.org/docs/_downloads/conda-cheatsheet.pdf

Managing Python

Managing Python

conda search --full-name python *or* **conda search -f python**	Check versions of Python available to install
conda create -n snakes python=3.4	Install different version of Python in new environment
source activate snakes *(Linux, Mac)* **activate snakes** *(Windows)*	Switch to the new environment that has a different version of Python ***TIP:*** *Activate prepends the path to the snakes environment.*

http://conda.pydata.org/docs/_downloads/conda-cheatsheet.pdf

Managing Packages in Python

Managing packages, including Python

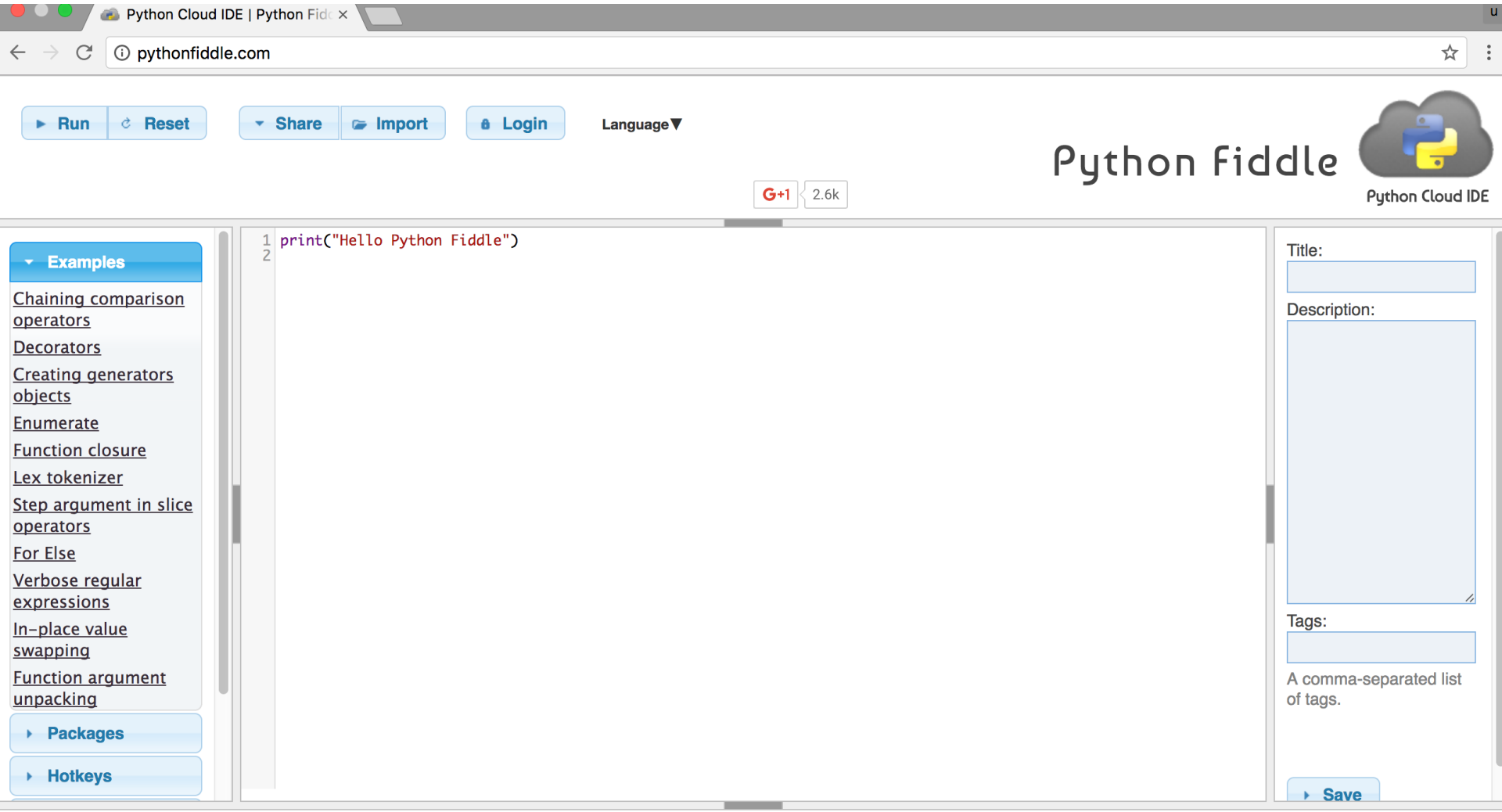
`conda list`	View list of packages and versions installed in active environment
`conda search beautiful-soup`	Search for a package to see if it is available to conda install
`conda install -n bunnies beautiful-soup`	Install a new package ***NOTE:*** *If you do not include the name of the new environment (`-n bunnies`) it will install in the current active environment.* ***TIP:*** *To view list of all packages available through* **`conda install`**, *visit http://docs.continuum.io/anaconda/pkg-docs.html.*
`conda update beautiful-soup`	Update a package in the current environment
`conda search --override-channels -c pandas bottleneck`	Search for a package in a specific location (i.e. the pandas channel on Anaconda.org) ***NOTE:*** *Or go to Anaconda.org in the browser and search by package name. This will show the specific channel (owner) through which it is available.*
`conda install -c pandas bottleneck`	Install a package from a specific channel
`conda search --override-channels -c defaults beautiful-soup`	Search for a package to see if it is available from the Anaconda repository
`source activate bunnies` *(Linux, Mac)* `activate bunnies` *(Windows)* `pip install see`	Activate the environment where you want to install a package and install it with pip (included with Anaconda and Miniconda)
`conda install iopro accelerate`	Install commercial Continuum packages
`conda skeleton pypi pyinstrument` `conda build pyinstrument`	Build a Conda package from a Python Package Index (PyPI) Package

http://conda.pydata.org/docs/_downloads/conda-cheatsheet.pdf

PyCharm: Python IDE

http://www.jetbrains.com/pycharm/

Python Fiddle



http://pythonfiddle.com/

Python 2 or 3

Create Python 2 or 3 environments

Anaconda supports Python 2.7, 3.4, and 3.5. The default is Python 2.7 or 3.5, depending on which installer you used. If the installer you used is Anaconda or Miniconda, the default is 2.7. If the installer you used is Anaconda3 or Miniconda3, the default is 3.5.

Create a Python 3.5 environment

To create a new environment with a different version of Python, use the `conda create` command. In this example, we'll make the new environment for Python 3.5:

```
$ conda create -n py35 python=3.5 anaconda
```

Here, the 'py35' is the name of the environment you want to create, and 'anaconda' is the meta-package that includes all of the actual Python packages comprising the Anaconda distribution. When creating a new environment and installing Anaconda, you can specify the exact package and Python versions, for example, `numpy=1.7` or `python=3.5`.

Create a Python 2.7 environment

In this example, we'll make a new environment for Python 2.7:

```
$ conda create -n py27 python=2.7 anaconda
```

Source: https://conda.io/docs/py2or3.html

References

- Yves Hilpisch (2014), Python for Finance: Analyze Big Financial Data, O'Reilly
- Ivan Idris (2015), Numpy Beginner's Guide, Third Edition, Packt Publishing
- Python, https://www.python.org/
- Python Programming Language, http://pythonprogramminglanguage.com/
- Numpy, http://www.numpy.org/