人工智慧投資分析

Artificial Intelligence for Investment Analysis

Python AI投資分析基礎

(Foundations of AI Investment Analysis in Python)

1071AIIA06
EMBA, IMTKU (M2399) (8540)
Thu 12,13,14 (19:20-22:10) (D503)

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/

2018-10-25

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2018/09/13	人工智慧投資分析課程介紹 (Course Orientation on Artificial Intelligence for Investment Analysis)
2	2018/09/20	AI 金融科技: 金融服務創新應用 (AI in FinTech: Financial Services Innovation and Application)
3	2018/09/27	機器人理財顧問與AI交談機器人 (Robo-Advisors and AI Chatbots)
4	2018/10/04	投資心理學與行為財務學 (Investing Psychology and Behavioral Finance)
5	2018/10/11	財務金融事件研究法 (Event Studies in Finance)
6	2018/10/18	人工智慧投資分析個案研究 I (Case Study on Artificial Intelligence for Investment Analysis I)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
7	2018/10/25	Python AI投資分析基礎 (Foundations of AI Investment Analysis in Python)
8	2018/11/01	Python Pandas量化投資分析 (Quantitative Investing with Pandas in Python)
9	2018/11/08	Python Scikit-Learn 機器學習 (Machine Learning with Scikit-Learn In Python)
10	2018/11/15	期中報告 (Midterm Project Report)
11	2018/11/22	TensorFlow 深度學習財務時間序列預測 I (Deep Learning for Financial Time Series Forecasting with TensorFlow I)
12	2018/11/29	TensorFlow 深度學習財務時間序列預測 II (Deep Learning for Financial Time Series Forecasting with TensorFlow II)

課程大綱 (Syllabus)

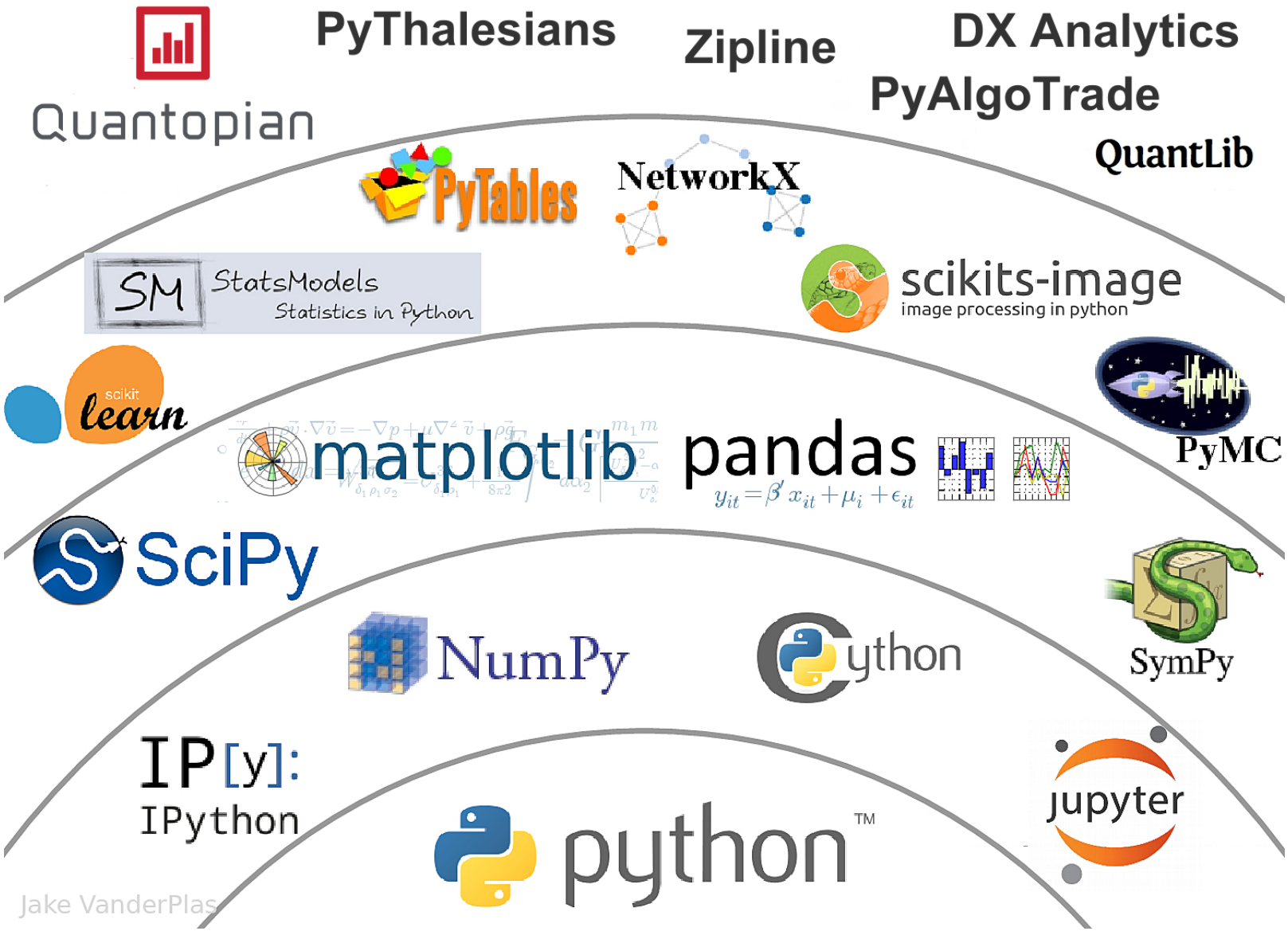
週次 (Week)	日期 (Date)	內容 (Subject/Topics)
13	2018/12/06	人工智慧投資分析個案研究 II (Case Study on Artificial Intelligence for Investment Analysis II)
14	2018/12/13	TensorFlow 深度學習財務時間序列預測 III (Deep Learning for Financial Time Series Forecasting with TensorFlow III)
15	2018/12/20	投資組合最佳化與程式交易 (Portfolio Optimization and Algorithmic Trading)
16	2018/12/27	自然語言處理 (Natural Language Processing)
17	2019/01/03	期末報告 I (Final Project Presentation I)
18	2019/01/10	期末報告 II (Final Project Presentation II)

Foundations of AI Investment Analysis in Python

Outline

- **Foundations of AI Investment Analysis in Python**
  - **Python**
    - Programming language
  - **Numpy**
    - Scientific computing
  - **Pandas**
    - Data structures and data analysis tools

The Quant Finance PyData Stack



Source: http://nbviewer.jupyter.org/format/slides/github/quantopian/pyfolio/blob/master/pyfolio/examples/overview_slides.ipynb#/5

AAPL
Adj Close
MA05
MA20
MA60
220
200
180
160
140
120
100
80
2016-02
2016-06
2016-10
2017-02
2017-06
2017-10
2018-02
2018-06
2018-10
Date

Python

python™

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

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

Google Colab

Hello, Colaboratory - Colabora

Secure https://colab.research.google.com/notebooks/welcome.ipynb

Hello, Colaboratory

File Edit View Insert Runtime Tools Help

SHARE

CODE TEXT CELL CELL COPY TO DRIVE CONNECT EDITING

Table of contents Code snippets Files

Getting Started

Highlighted Features

TensorFlow execution

GitHub

Visualization

Forms

Examples

Local runtime support

SECTION

Welcome to Colaboratory!

Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. See our FAQ for more info.

Getting Started

- Overview of Colaboratory
- Loading and saving data: Local files, Drive, Sheets, Google Cloud Storage
- Importing libraries and installing dependencies
- Using Google Cloud BigQuery
- Forms, Charts, Markdown, & Widgets
- TensorFlow with GPU
- Machine Learning Crash Course: Intro to Pandas & First Steps with TensorFlow

Highlighted Features

Seedbank

Looking for Colab notebooks to learn from? Check out Seedbank, a place to discover interactive machine learning examples.

TensorFlow execution

Colaboratory allows you to execute TensorFlow code in your browser with a single click. The example below adds two matrices.

$$\begin{bmatrix} 1. & 1. & 1. \end{bmatrix} + \begin{bmatrix} 1. & 2. & 3. \end{bmatrix} = \begin{bmatrix} 2. & 3. & 4. \end{bmatrix}$$

https://colab.research.google.com/notebooks/welcome.ipynb

Python in Google Colab

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

python101.ipynb - Colaboratory

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT?authuser=2#scrollTo=wsh36fLxDKC3

python101.ipynb

File Edit View Insert Runtime Tools Help

COMMENT SHARE

CODE TEXT CELL CELL CONNECTED EDITING

```python
# Future Value
pv = 100
r = 0.1
n = 7
fv = pv * ((1 + (r)) ** n)
print(round(fv, 2))
```

```
194.87
```

[11]

```python
amount = 100
interest = 10 #10% = 0.01 * 10
years = 7

future_value = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

```
194.87
```

[12]

```python
# Python Function def
def getfv(pv, r, n):
    fv = pv * ((1 + (r)) ** n)
    return fv
fv = getfv(100, 0.1, 7)
print(round(fv, 2))
```

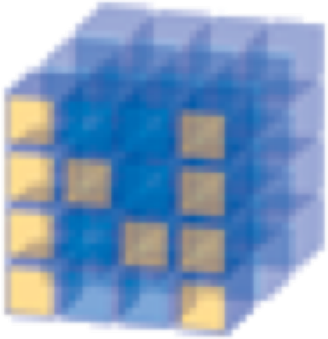
```
194.87
```

[13]

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

```
Pass
```

Numpy



NumPy
Base
N-dimensional array
package

Python matplotlib

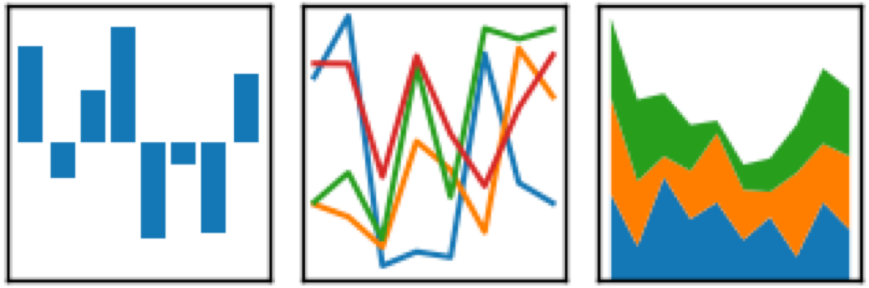
matplotlib

Source: https://matplotlib.org/

Python Pandas

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



http://pandas.pydata.org/

Iris flower data set

setosa

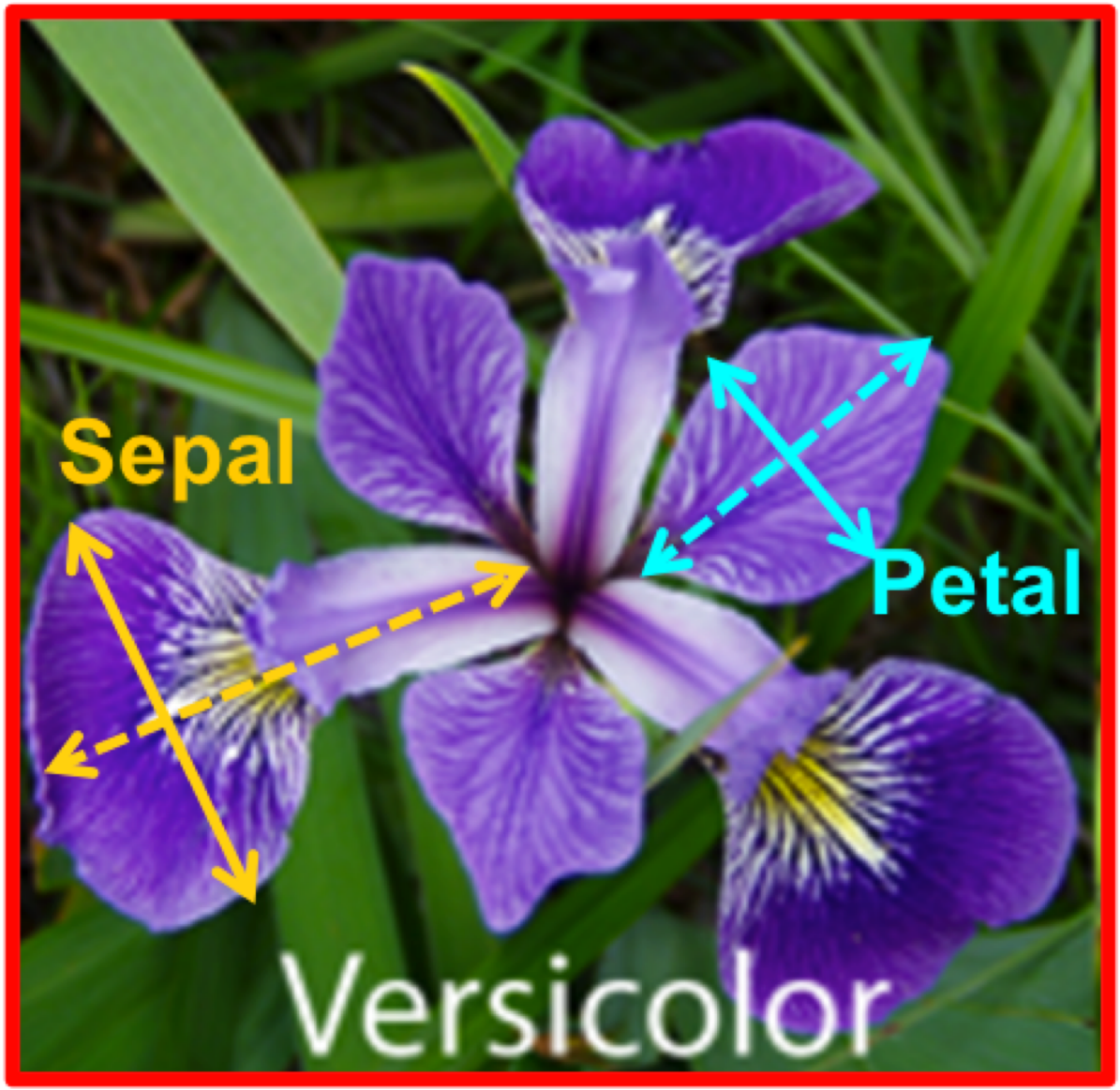
versicolor

virginica

Source: https://en.wikipedia.org/wiki/Iris_flower_data_set

Source: http://suruchifialoke.com/2016-10-13-machine-learning-tutorial-iris-classification/

Iris Classfication



Source: http://suruchifialoke.com/2016-10-13-machine-learning-tutorial-iris-classification/

iris.data

https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data

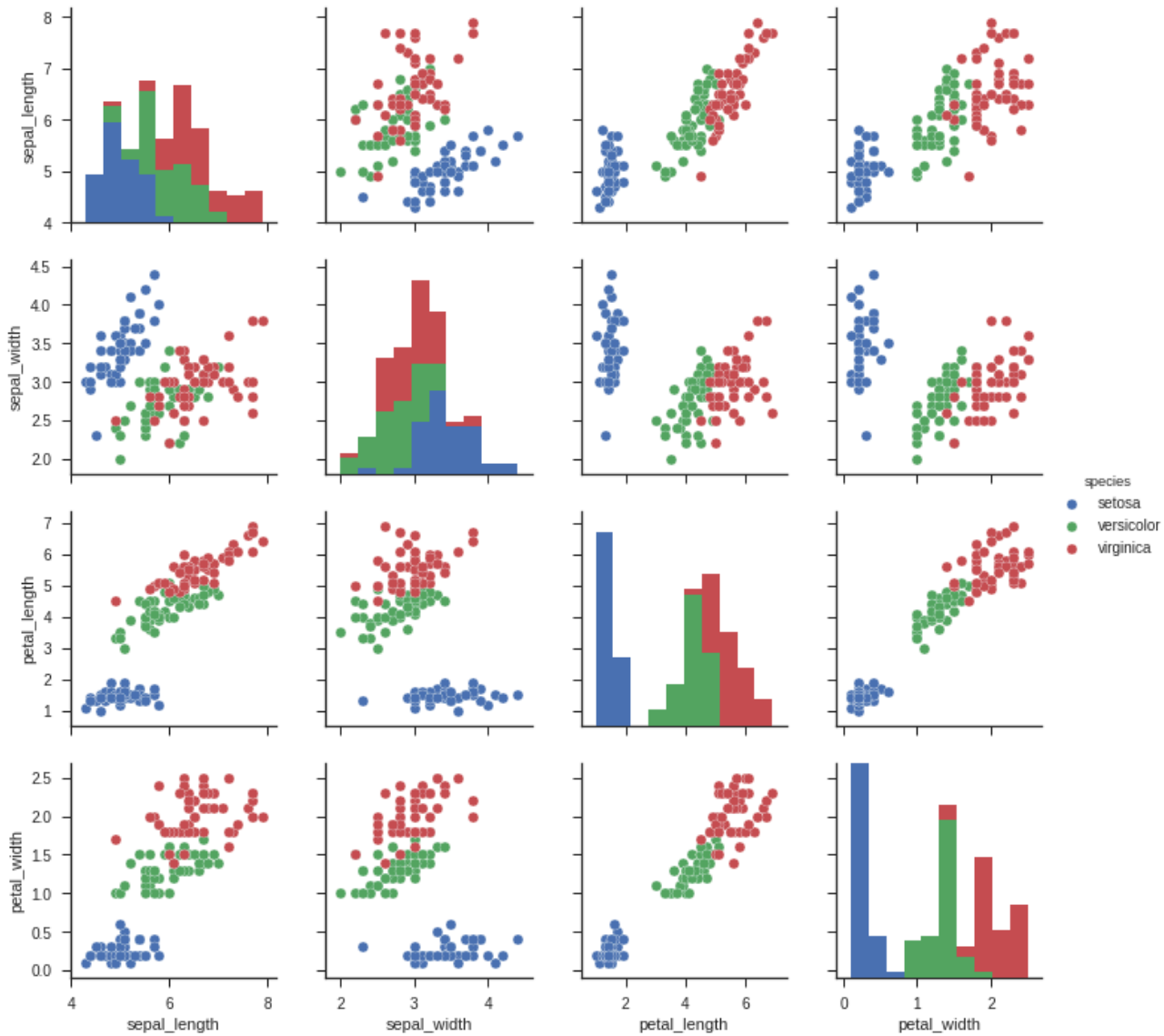
```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
5.4,3.9,1.3,0.4,Iris-setosa
5.1,3.5,1.4,0.3,Iris-setosa
5.7,3.8,1.7,0.3,Iris-setosa
5.1,3.8,1.5,0.3,Iris-setosa
5.4,3.4,1.7,0.2,Iris-setosa
5.1,3.7,1.5,0.4,Iris-setosa
4.6,3.6,1.0,0.2,Iris-setosa
5.1,3.3,1.7,0.5,Iris-setosa
4.8,3.4,1.9,0.2,Iris-setosa
5.0,3.0,1.6,0.2,Iris-setosa
```

setosa

virginica

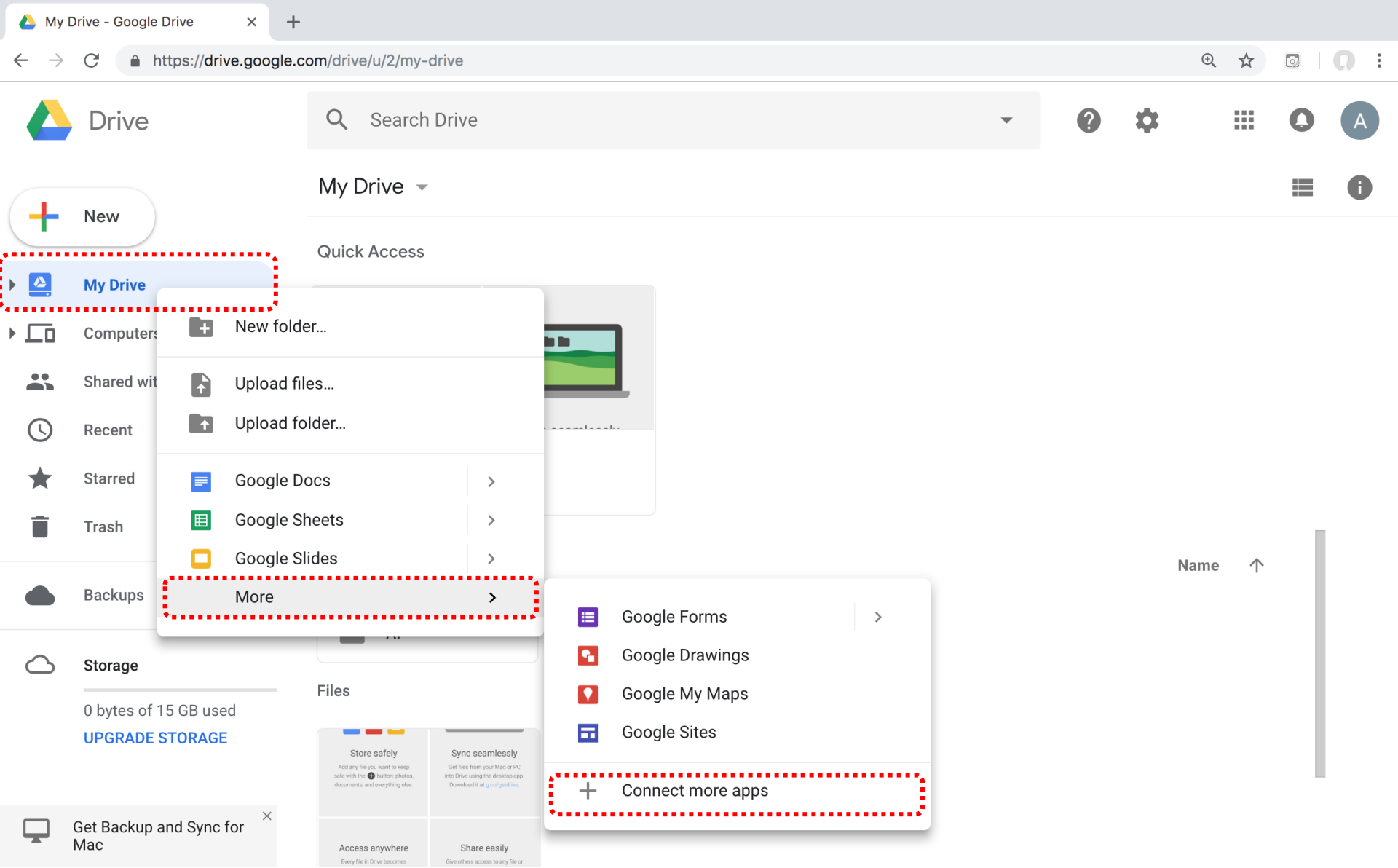
versicolor

Iris Data Visualization

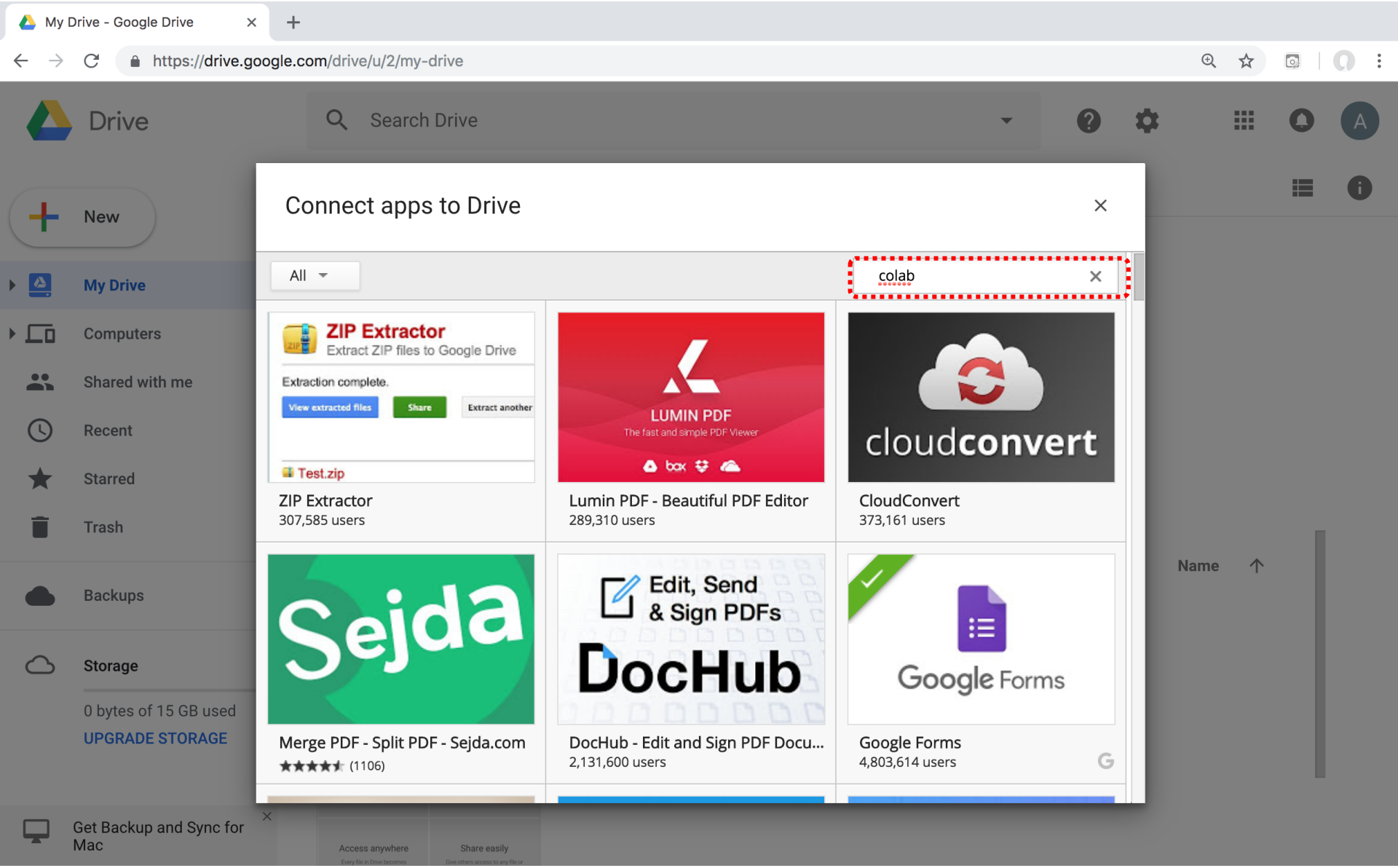


Source: https://seaborn.pydata.org/generated/seaborn.pairplot.html

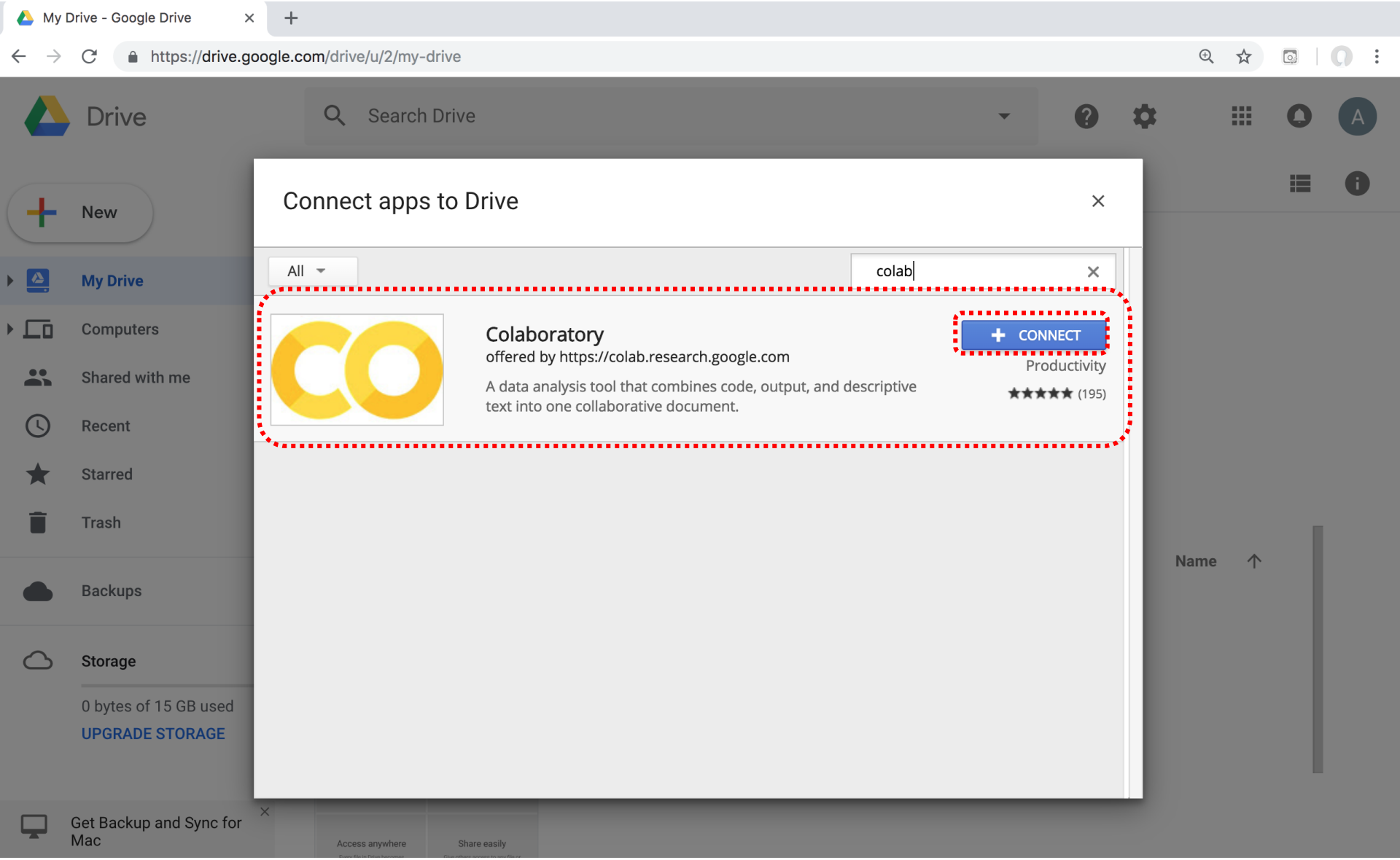
Connect Google Colab in Google Drive



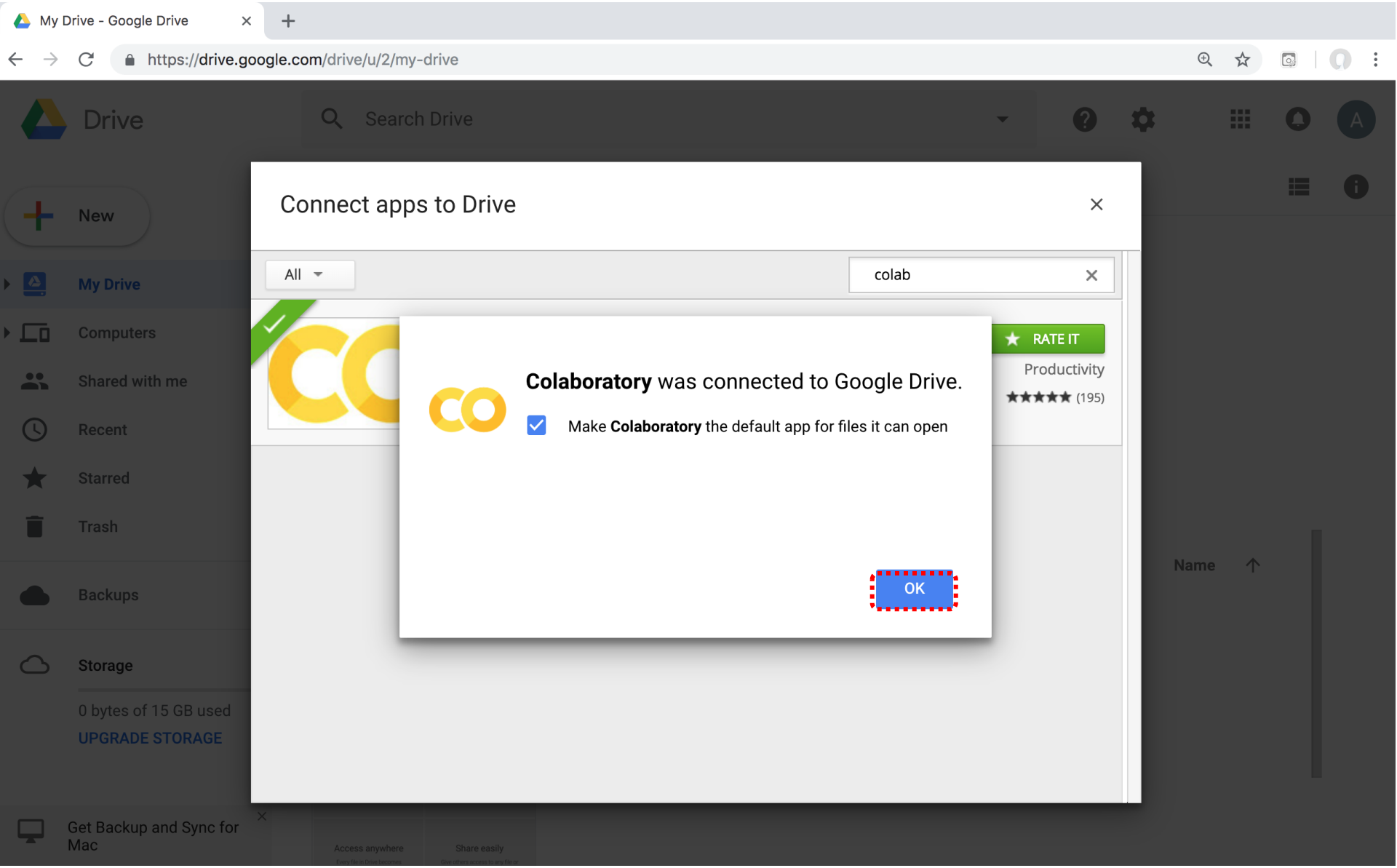
Google Colab



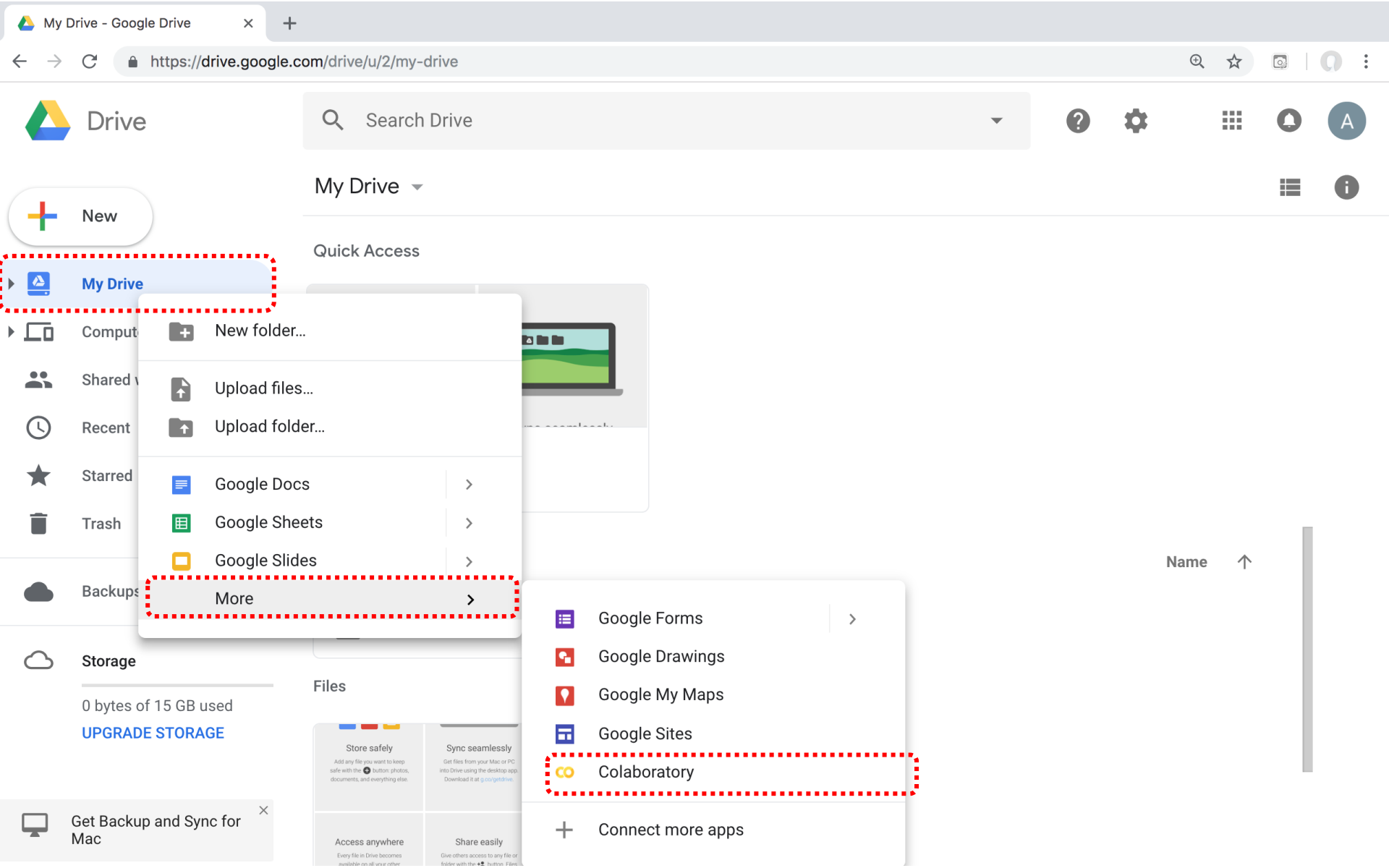
Google Colab



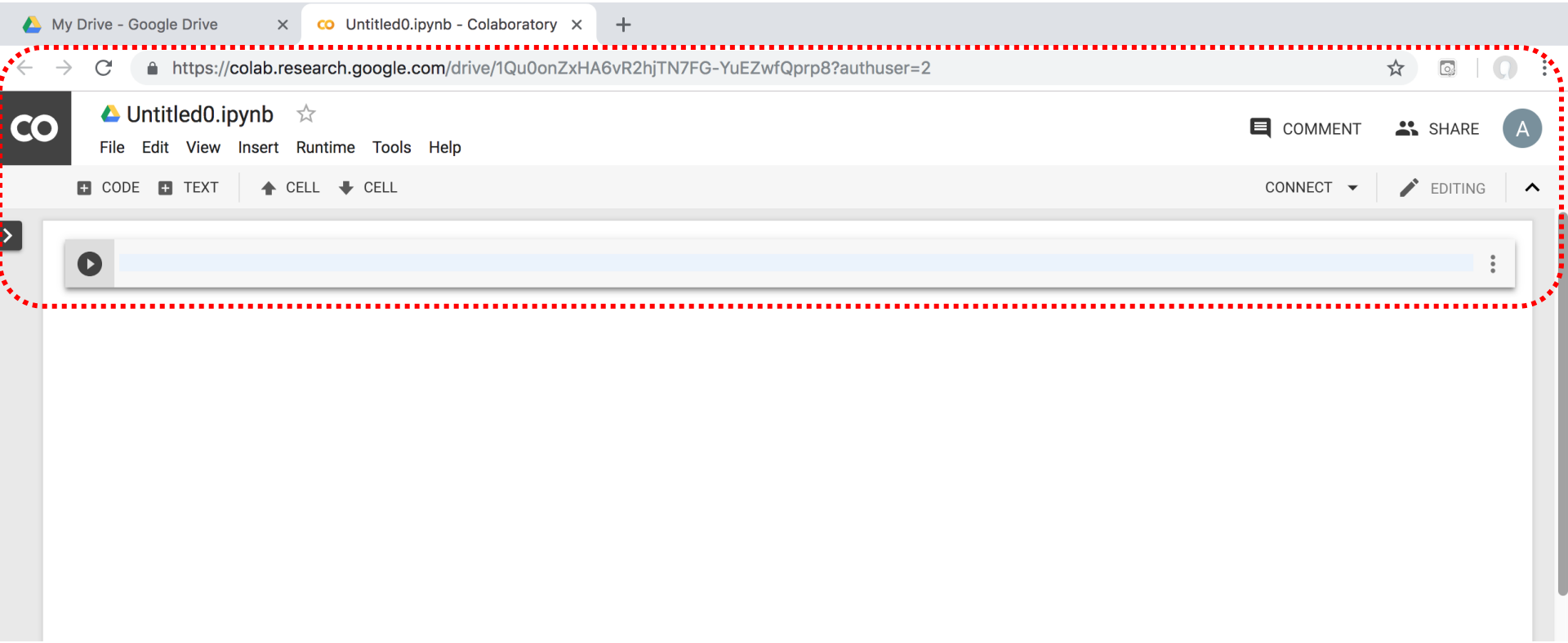
Connect Colaboratory to Google Drive



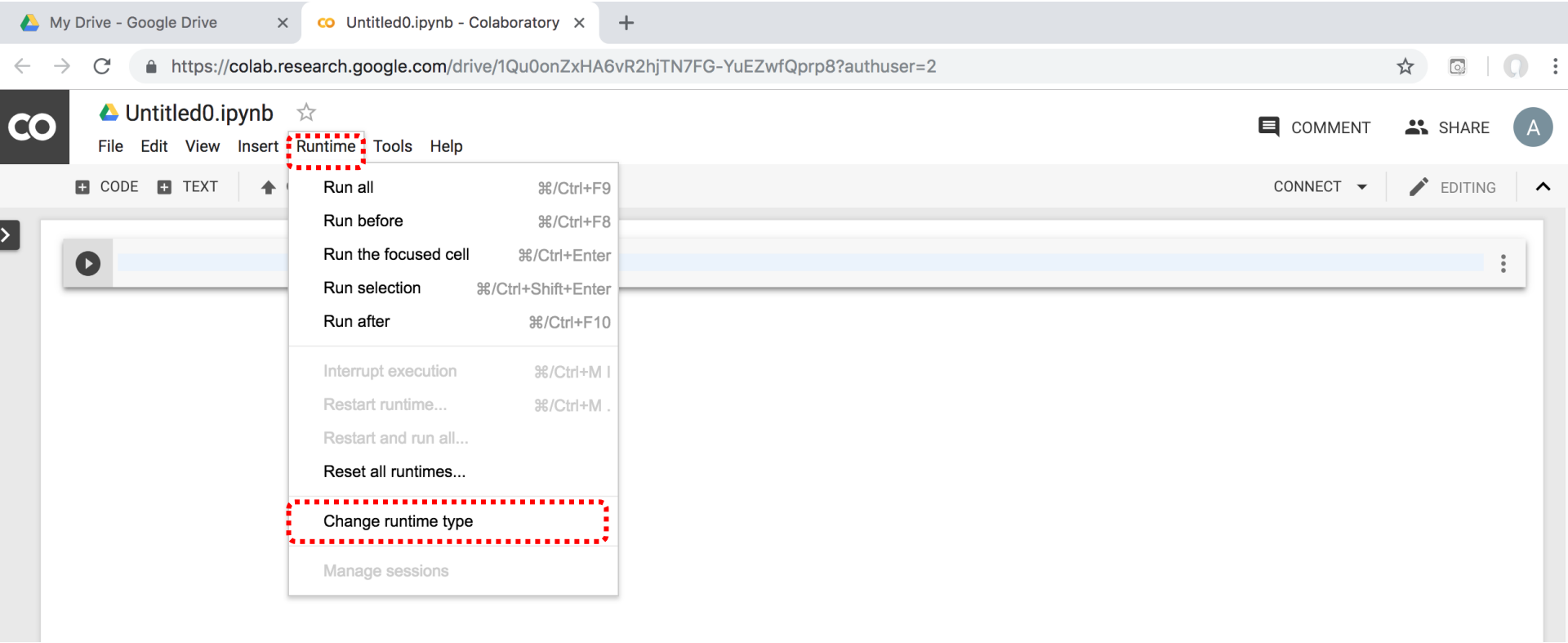
Google Colab



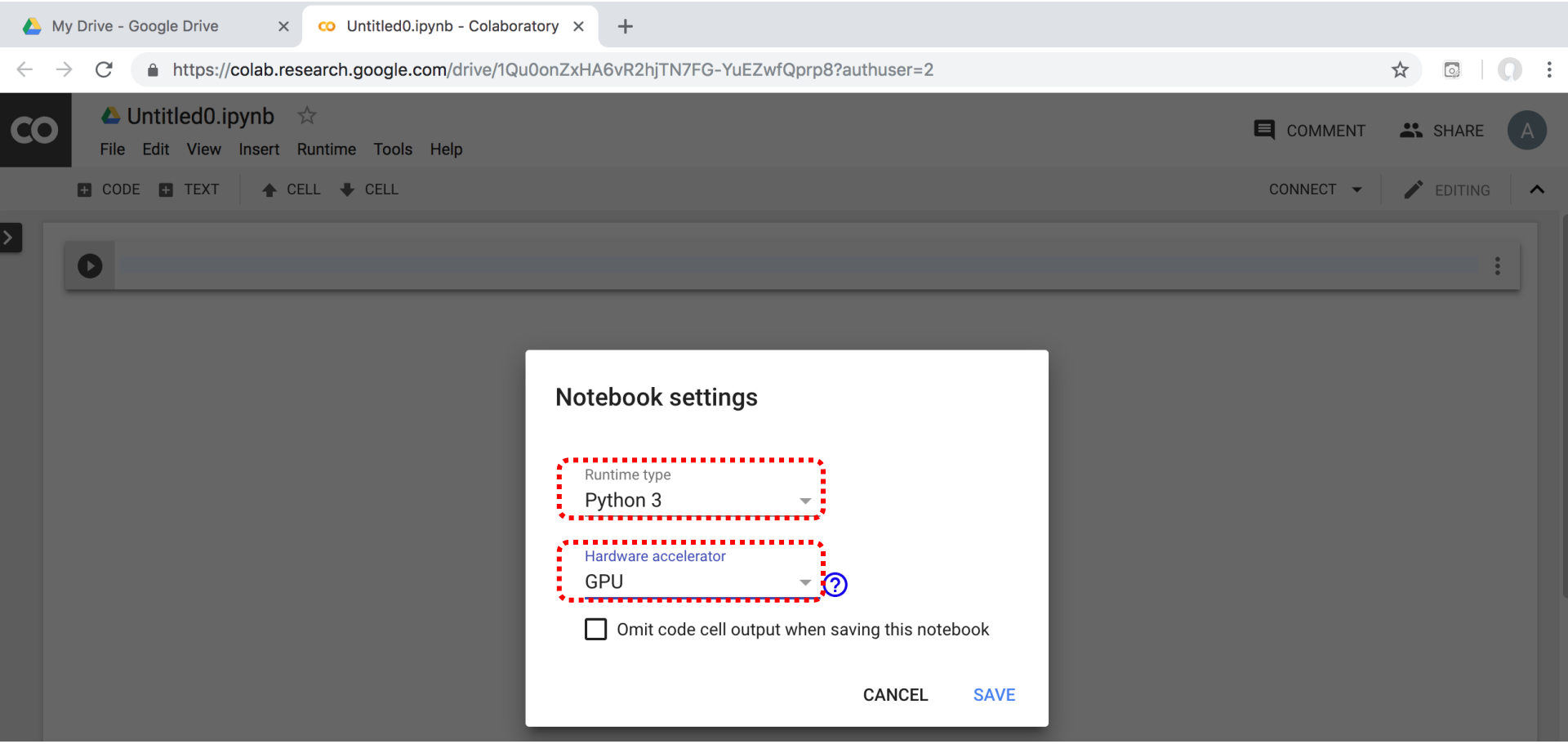
Google Colab



Google Colab



Run Jupyter Notebook Python3 GPU Google Colab



Google Colab Python Hello World print('Hello World')

Data Visualization in Google Colab

https://colab.research.google.com/drive/1KRqtEUd2Hg4dM2au9bfVQKrxWnWN3O9-

datav.ipynb - Colaboratory

https://colab.research.google.com/drive/1KRqtEUd2Hg4dM2au9bfVQKrxWnWN3O9-?authuser=2

datav.ipynb

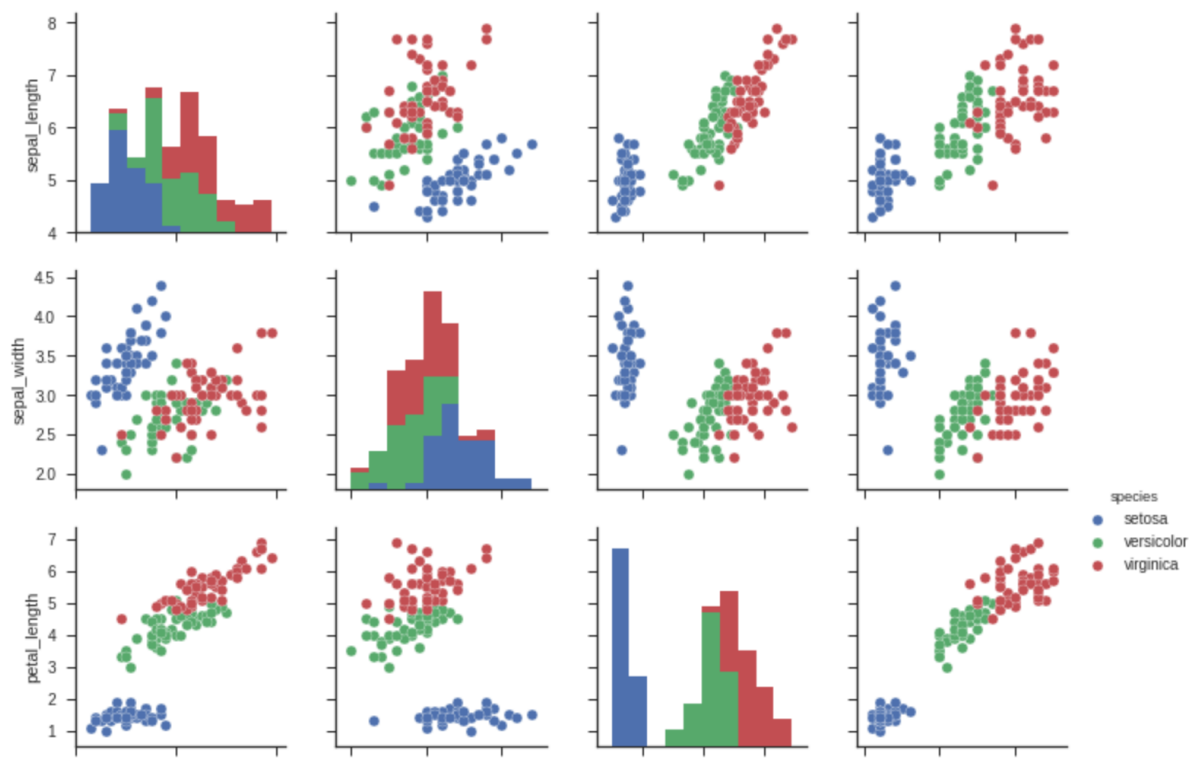
File Edit View Insert Runtime Tools Help

COMMENT SHARE

CODE TEXT CELL CELL

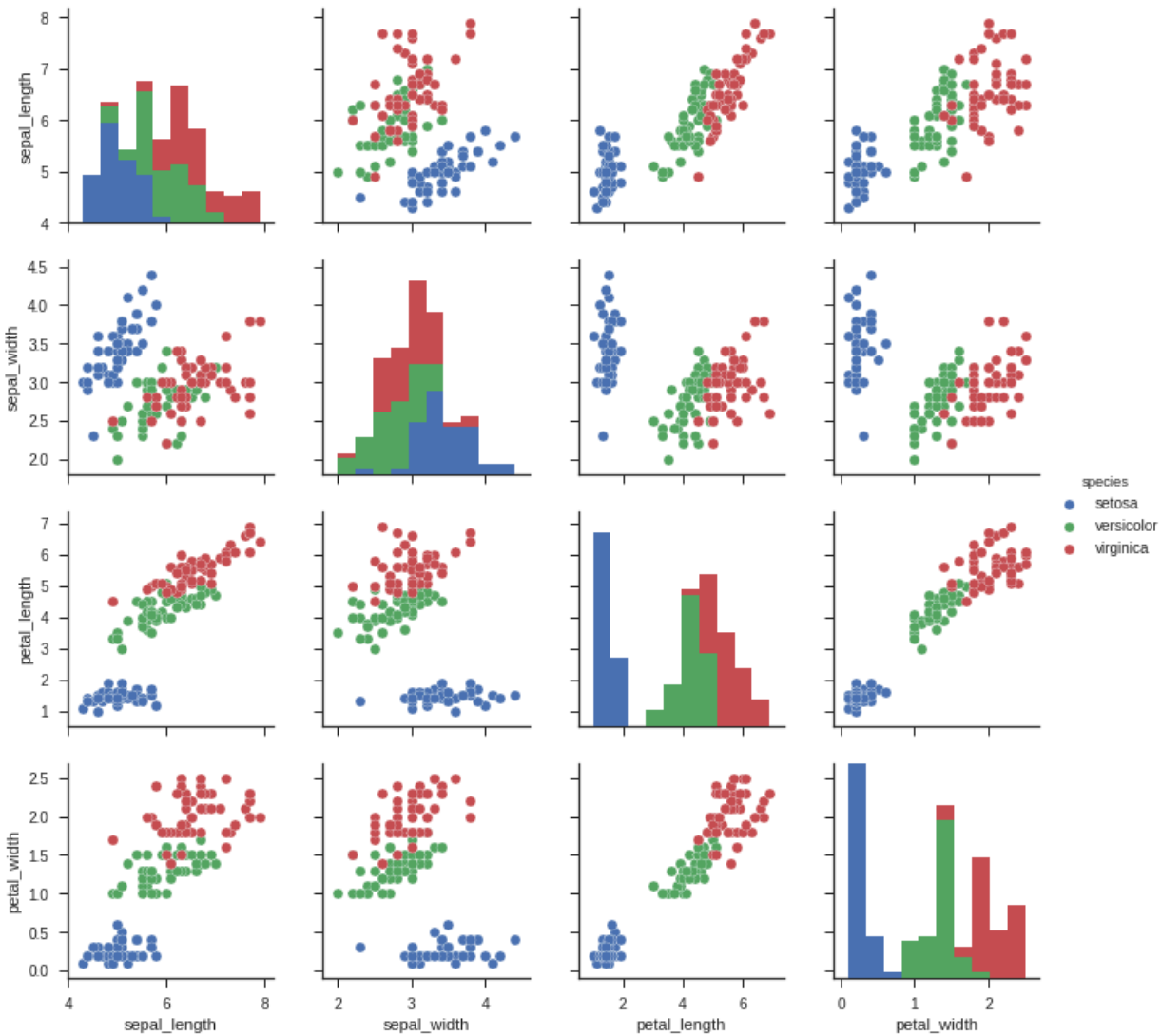
CONNECTED EDITING

```
import seaborn as sns
sns.set(style="ticks", color_codes=True)
iris = sns.load_dataset("iris")
g = sns.pairplot(iris, hue="species")
```



Source: https://seaborn.pydata.org/generated/seaborn.pairplot.html

```
import seaborn as sns
sns.set(style="ticks", color_codes=True)
iris = sns.load_dataset("iris")
g = sns.pairplot(iris, hue="species")
```



Source: https://seaborn.pydata.org/generated/seaborn.pairplot.html

https://colab.research.google.com/drive/1KRqtEUd2Hg4dM2au9bfVQKrxWnWN3O9-

```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix

# Load dataset
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)

print(df.head(10))
print(df.tail(10))
print(df.describe())
print(df.info())
print(df.shape)
print(df.groupby('class').size())

plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()

df.hist()
plt.show()

scatter_matrix(df)
plt.show()

sns.pairplot(df, hue="class", size=2)
```

Source: https://machinelearningmastery.com/machine-learning-in-python-step-by-step/

```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix
```

```
# Import Libraries
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix
print('imported')
```

```
imported
```

```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)
print(df.head(10))
```

```
# Load dataset
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)
print(df.head(10))
```

```
   sepal-length  sepal-width  petal-length  petal-width        class
0           5.1          3.5           1.4          0.2  Iris-setosa
1           4.9          3.0           1.4          0.2  Iris-setosa
2           4.7          3.2           1.3          0.2  Iris-setosa
3           4.6          3.1           1.5          0.2  Iris-setosa
4           5.0          3.6           1.4          0.2  Iris-setosa
5           5.4          3.9           1.7          0.4  Iris-setosa
6           4.6          3.4           1.4          0.3  Iris-setosa
7           5.0          3.4           1.5          0.2  Iris-setosa
8           4.4          2.9           1.4          0.2  Iris-setosa
9           4.9          3.1           1.5          0.1  Iris-setosa
```

df.tail(10)

```
print(df.tail(10))
```

```
     sepal-length  sepal-width  petal-length  petal-width            class
140           6.7          3.1           5.6          2.4  Iris-virginica
141           6.9          3.1           5.1          2.3  Iris-virginica
142           5.8          2.7           5.1          1.9  Iris-virginica
143           6.8          3.2           5.9          2.3  Iris-virginica
144           6.7          3.3           5.7          2.5  Iris-virginica
145           6.7          3.0           5.2          2.3  Iris-virginica
146           6.3          2.5           5.0          1.9  Iris-virginica
147           6.5          3.0           5.2          2.0  Iris-virginica
148           6.2          3.4           5.4          2.3  Iris-virginica
149           5.9          3.0           5.1          1.8  Iris-virginica
```

df.describe()

```
print(df.describe())
```

```
       sepal-length  sepal-width  petal-length  petal-width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000
```

print(df.info())
print(df.shape)

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepal-length    150 non-null float64
sepal-width     150 non-null float64
petal-length    150 non-null float64
petal-width     150 non-null float64
class           150 non-null object
dtypes: float64(4), object(1)
memory usage: 5.9+ KB
None
```

```
print(df.shape)
```

```
(150, 5)
```

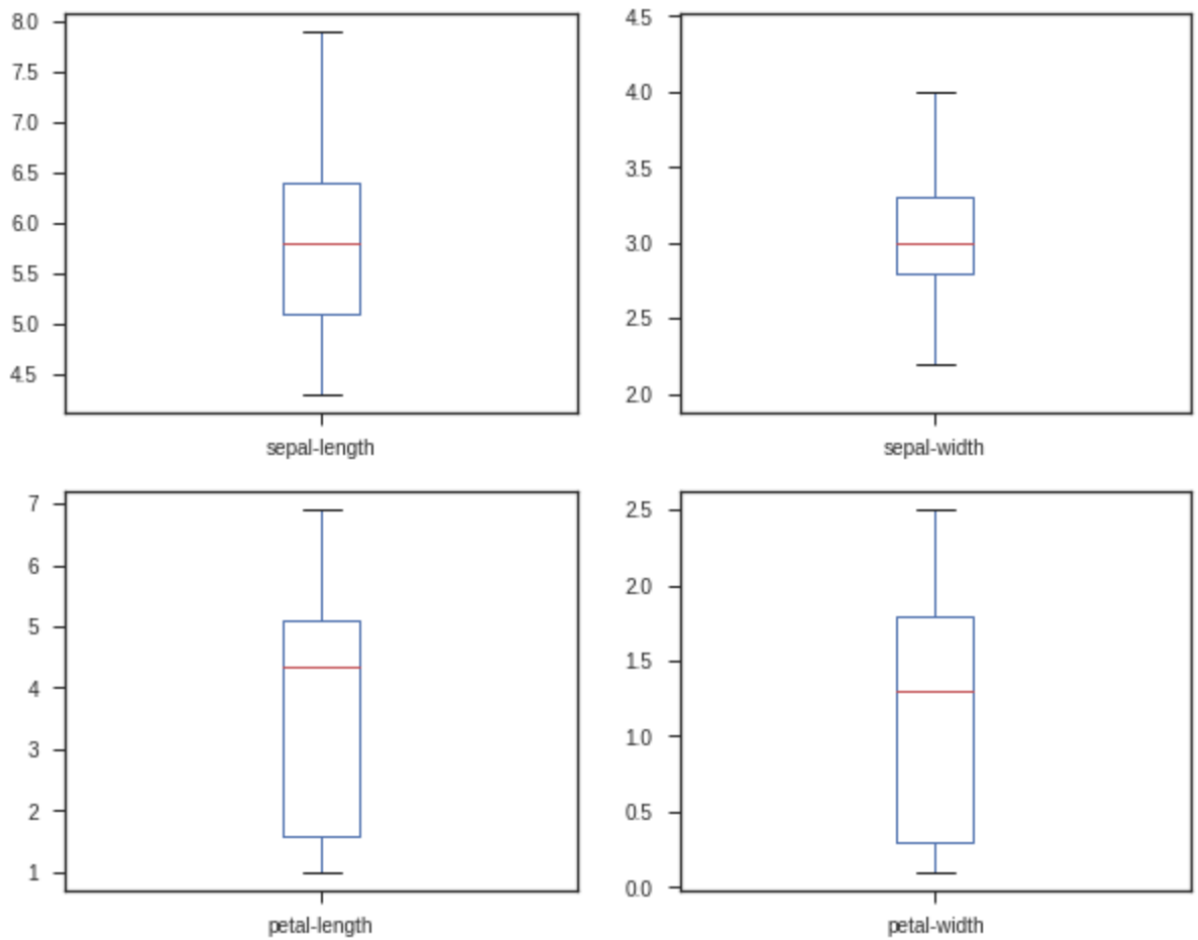
df.groupby('class').size()

```
print(df.groupby('class').size())
```

```
class
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

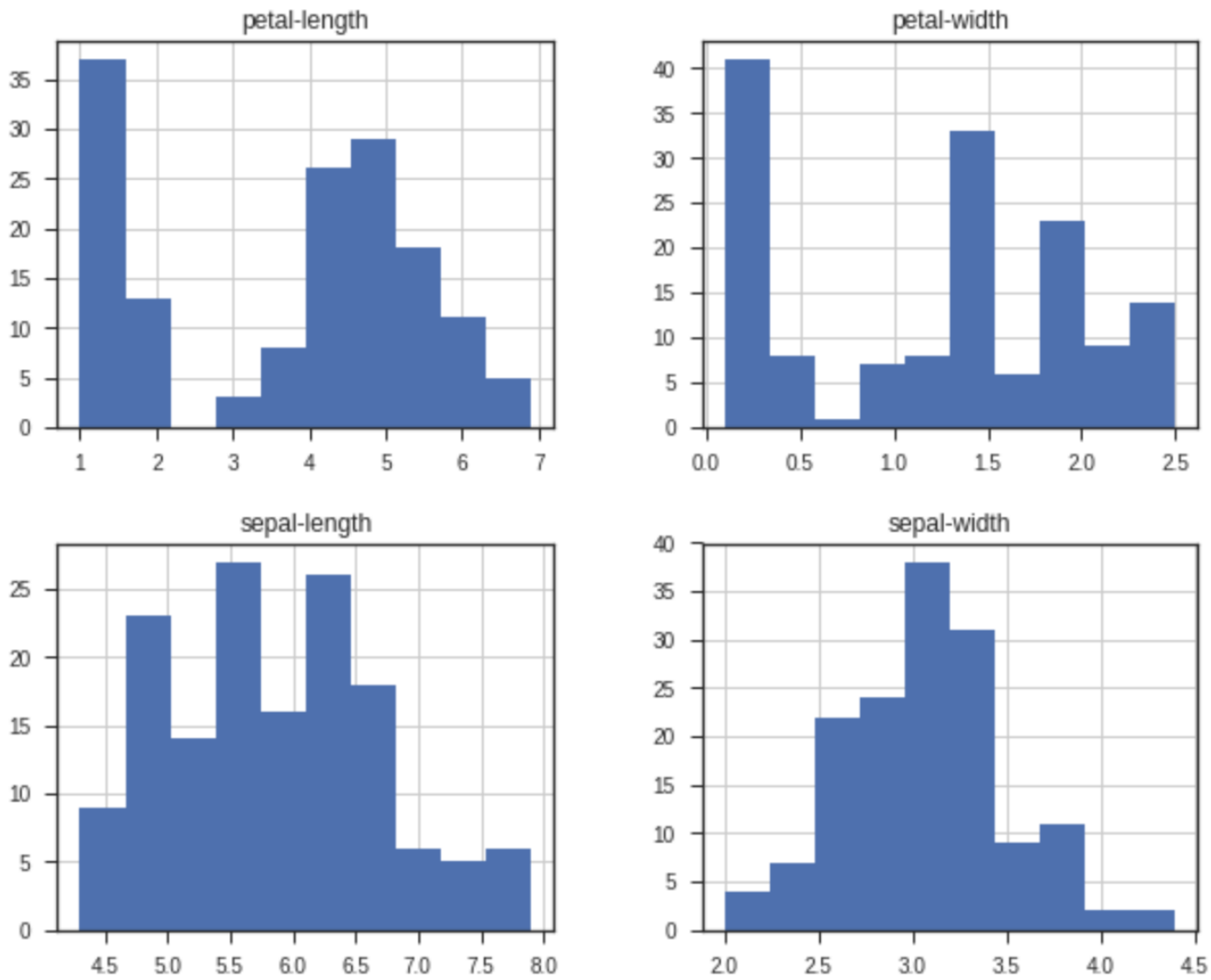
```
plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()
```


```
plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()
```

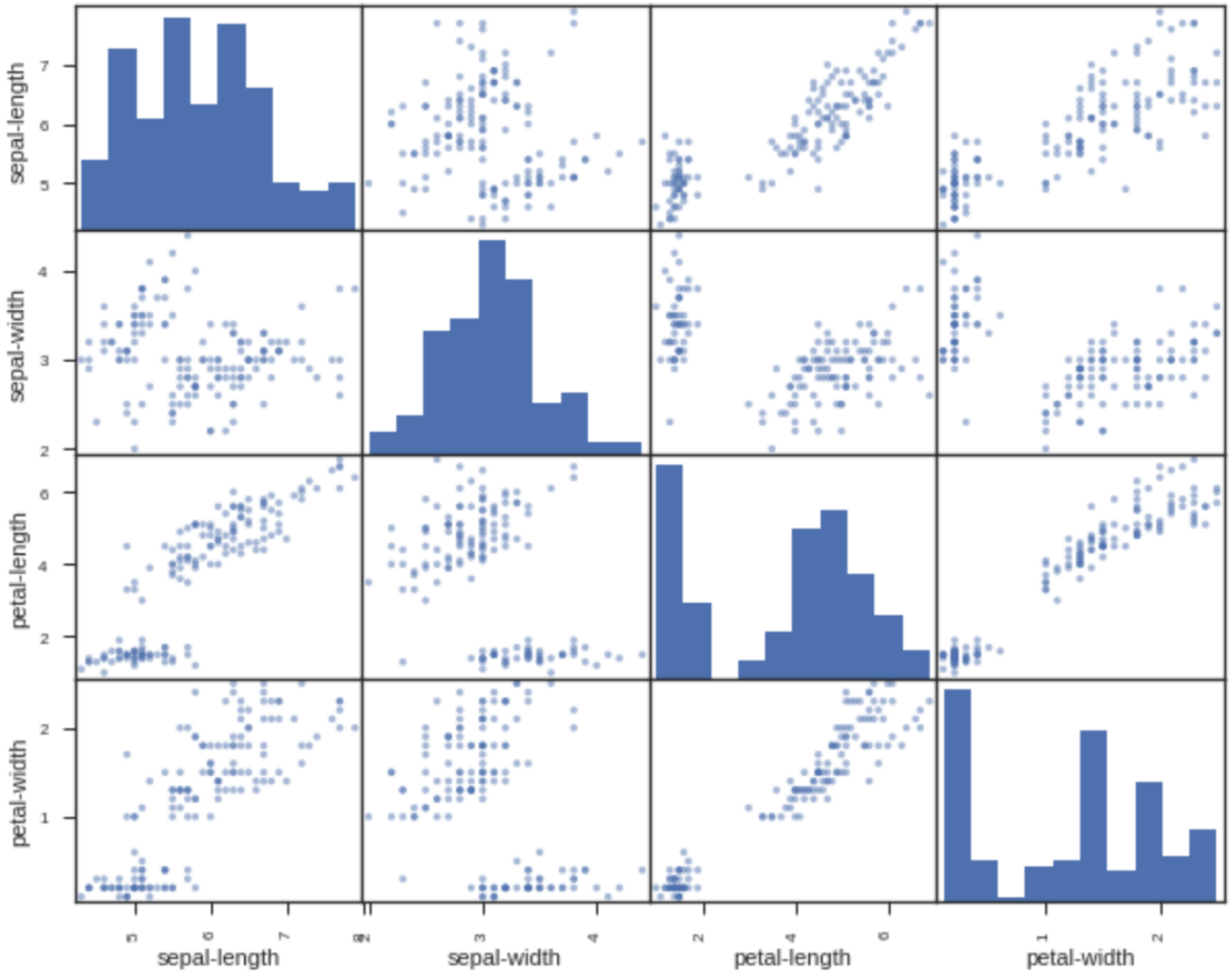
df.hist()
plt.show()

```
df.hist()
plt.show()
```



scatter_matrix(df)
plt.show()

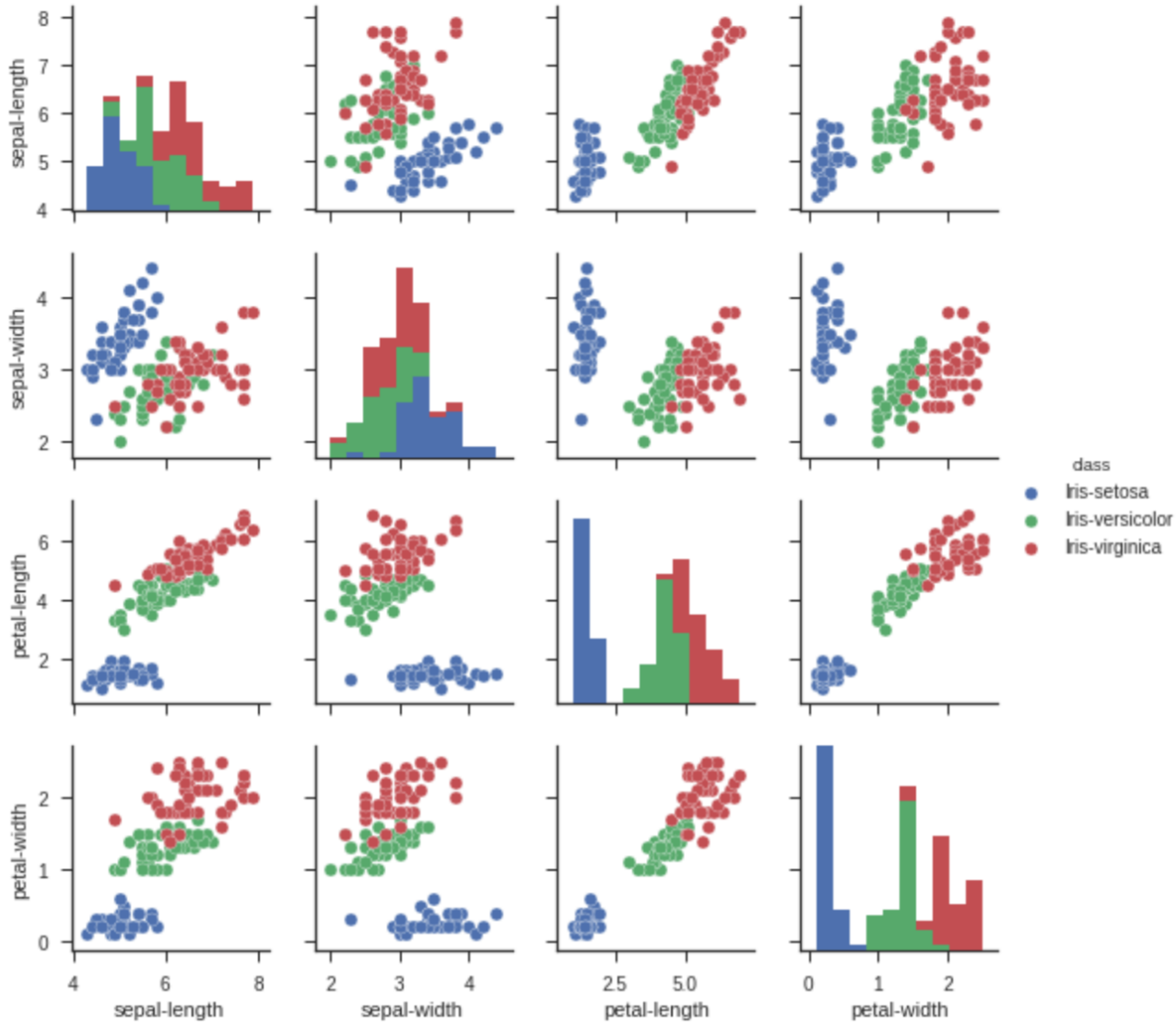
```
scatter_matrix(df)
plt.show()
```



sns.pairplot(df, hue="class", size=2)

```
sns.pairplot(df, hue="class", size=2)
```

```
<seaborn.axisgrid.PairGrid at 0x7f1d21267390>
```



ANACONDA.

Anaconda

The Most Popular

Python

Data Science Platform

Source: https://www.anaconda.com/

Download Anaconda

Documentation Blog Contact Anaconda Cloud

ANACONDA.

What is Anaconda? Products Support Community Resources About Download

Don't Miss AnacondaCon Apr 8-11 Austin TX!

Download Anaconda Distribution

Version 5.1 | Release Date: February 15, 2018

Download For:

High-Performance Distribution

Easily install 1,000+ data science packages

Package Management

Manage packages, dependencies and environments with conda

Portal to Data Science

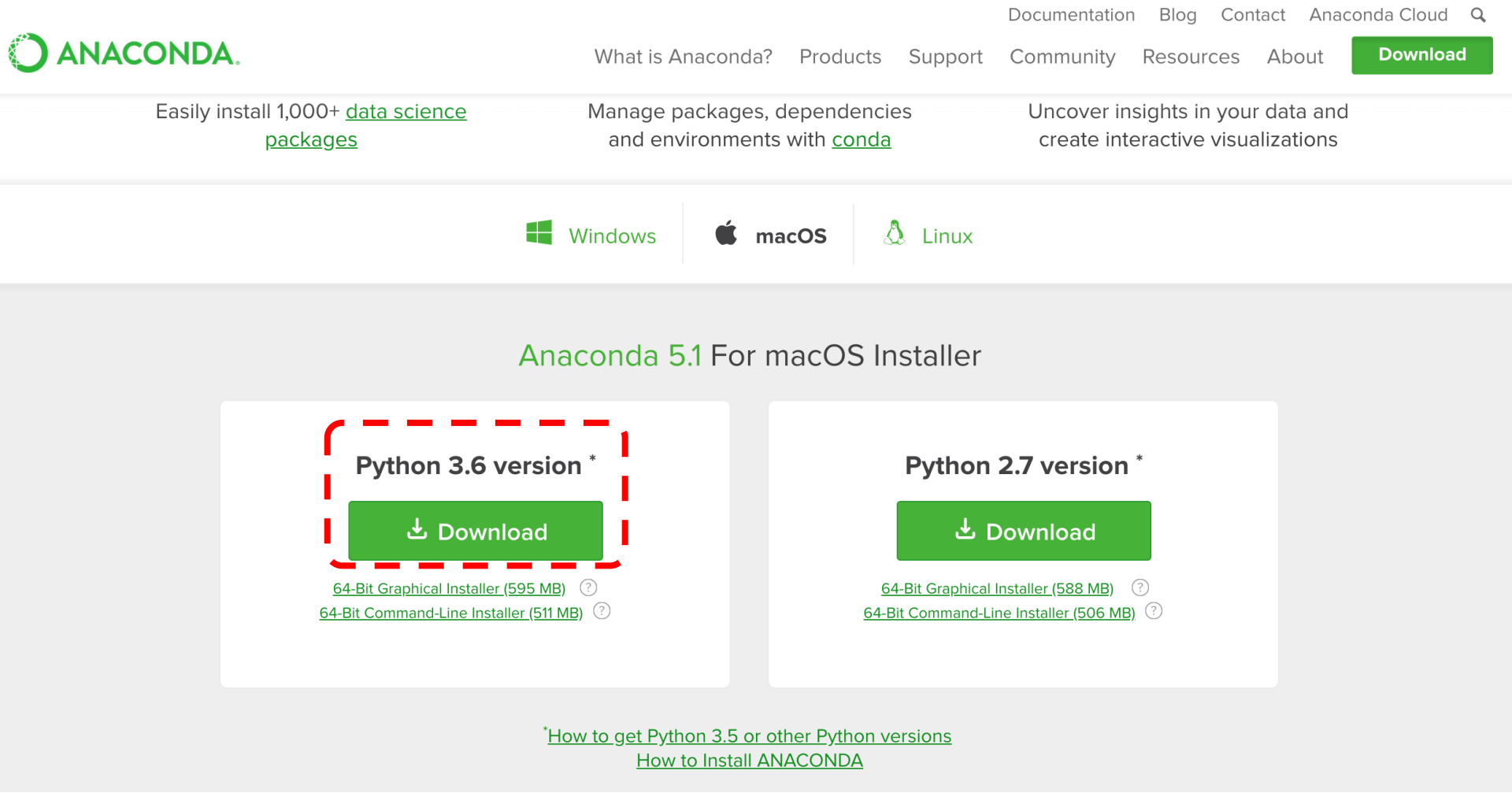
Uncover insights in your data and create interactive visualizations

Windows macOS Linux

Anaconda 5.1 For macOS Installer

https://www.anaconda.com/download

Download Anaconda Python 3.6 version

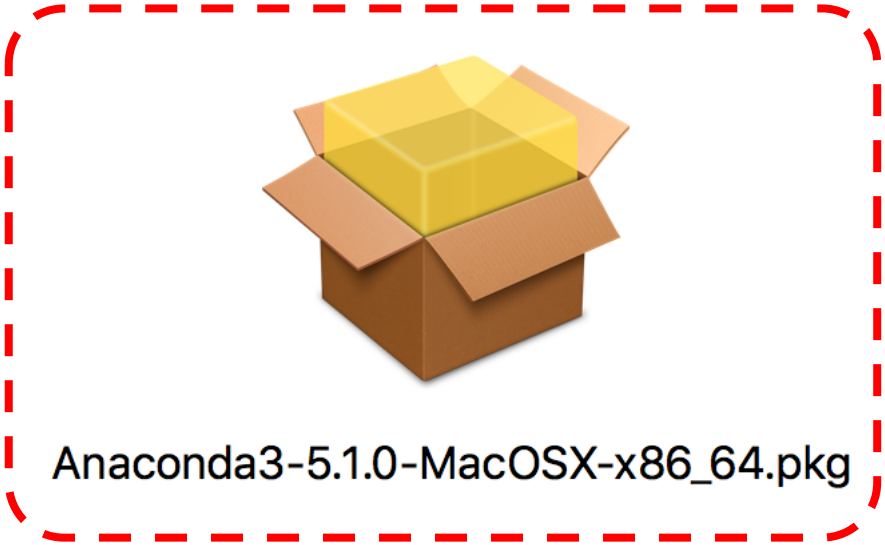


https://www.anaconda.com/download/#macos

Install Anaconda

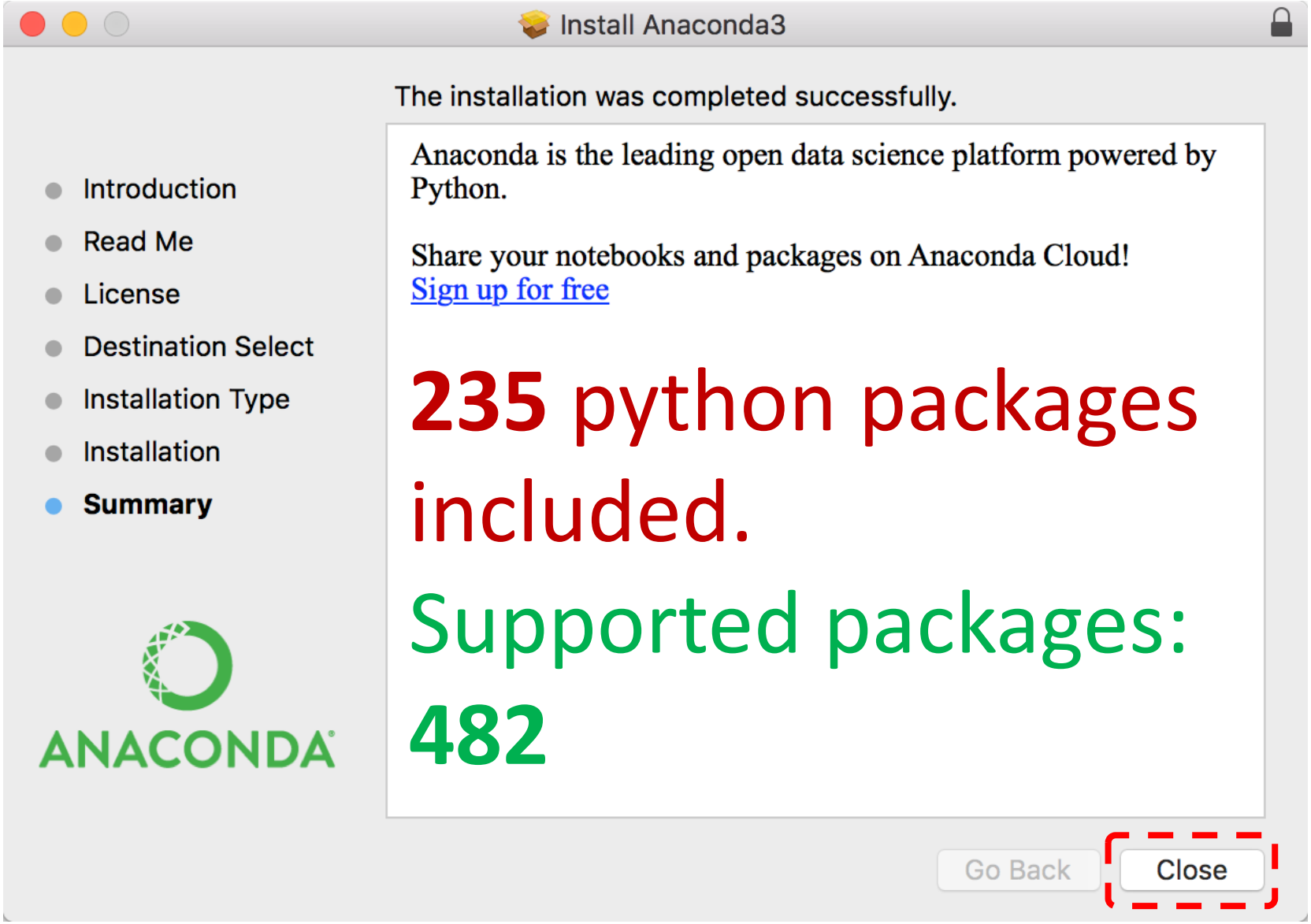
Python 3.6

Anaconda3-5.1.0-MacOSX-x86_64.pkg



Installer package - 623.6 MB

Install Anaconda 3



Source: https://docs.anaconda.com/anaconda/packages/pkg-docs

Install Anaconda 3

```
# Name
_ipyw_jlab_nb_ext_conf
alabaster
anaconda-client
anaconda-navigator
anaconda-project
appnope
appscript
asn1crypto
astroid
astropy
attrs
babel
backports
backports.shutil_get_term
beautifulsoup4
bitarray
bkcharts
blaze
bleach
bokeh
boto
bottleneck
bzip2
ca-certificates
certifi
cffi
chardet
click
cloudpickle
clyent
colorama
conda
conda-build
conda-env
conda-verify
contextlib2
cryptography
curl
cycler
cython
cytoolz
dask
dask-core
datashape
dbus
decorator
distributed
docutils
entrypoints
et_xmlfile
expat
fastcache
filelock
flask
flask-cors
freetype
get_terminal_size
gettext
gevent
glib
glob2
gmp
gmpy2
greenlet
h5py
hdf5
heapdict
html5lib
icu
idna
imageio
imagesize
intel-openmp
ipykernel
ipython
ipython_genutils
ipywidgets
isort
itsdangerous
jbig
jdcal
jedi
jinja2
jpeg
jsonschema
jupyter
jupyter_client
jupyter_console
jupyter_core
jupyterlab
jupyterlab_launcher
lazy-object-proxy
libcurl
libcxx
libcxxabi
libedit
libffi
libgfortran
libiconv
libpng
libsodium
libssh2
libtiff
libxml2
libxslt
llvmlite
locket
lxml
lzo
markupsafe
matplotlib
mccabe
mistune
mkl
mkl-service
mpc
mpfr
mpmath
msgpack-python
multipledispatch
navigator-updater
nbconvert
nbformat
ncurses
networkx
nltk
nose
notebook
numba
numexpr
numpy
numpydoc
odo
olefile
openpyxl
openssl
packaging
pandas
pandoc
pandocfilters
parso
partd
path.py
pathlib2
patsy
pcre
pep8
pexpect
pickleshare
pillow
pip
pkginfo
pluggy
ply
prompt_toolkit
psutil
ptyprocess
py
pycodestyle
pycosat
pycparser
pycrypto
pycurl
pyflakes
pygments
pylint
pyodbc
pyopenssl
pyparsing
pyqt
pysocks
pytables
pytest
python
python-dateutil
python.app
pytz
pywavelets
pyyaml
pyzmq
qt
qtawesome
qtconsole
qtpy
readline
requests
rope
ruamel_yaml
scikit-image
scikit-learn
scipy
seaborn
send2trash
setuptools
simplegeneric
singledispatch
sip
six
snowballstemmer
sortedcollections
sortedcontainers
sphinx
sphinxcontrib
sphinxcontrib-websupport
spyder
sqlalchemy
sqlite
statsmodels
sympy
tblib
terminado
testpath
tk
toolz
tornado
traitlets
typing
unicodecsv
unixodbc
urllib3
wcwidth
webencodings
werkzeug
wheel
widgetsnbextension
wrapt
xlrd
xlsxwriter
xlwings
xlwt
xz
yaml
zeromq
zict
zlib
```

		py36_0
		py36_0
		py36h9364dc8_1
		py36h92f4a7a_1
		py36_0
		py36h1de35cc_0
		h3efe00b_0
		py36h9c68fc9_0
		py36h7f3cf04_0
		py36hda67792_0
		py36_1
		py36h625a49b_0
		h35a86e2_3
		py36_0
		py36_0
		py36h65bd3ce_0
		py36haa2d9ef_0
		py36he531d66_0
		h4cb4dde_1
		py36h68b9469_0
		py36h8c6ec74_0
		py36h3b9701d_1
		py36_0
		py36h5eb2c71_1
		py36_0
		py36hc29e774_0
		py36h336f4a2_1
		py36h3736301_0
		py36_0
		py36h5ad1178_0
		h0278029_2
		4_2
		0_2
		4_0
		0_2

	0.10.7
	0.15.35
	0.13.1
	0.19.1
	1.0.0
	0.8.1
	1.4.2
	38.4.0
	0.8.1
	3.4.0.3
	4.18.1
	1.11.0
	1.2.1
	0.5.3

235 python packages included.

Verify that conda is installed, check current conda version

- **`conda --version`**

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

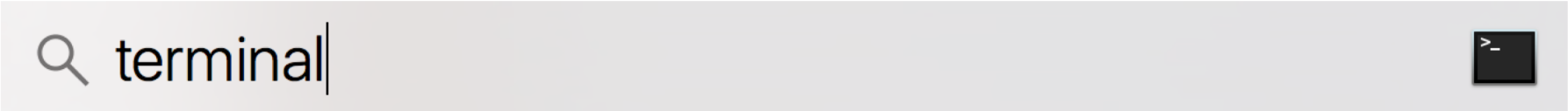
http://conda.pydata.org/docs/using/using.html#verify-that-conda-is-installed-check-current-conda-version

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

imyday — -bash — 80×24

```
iMyday-MacBook-Pro:~ imyday$ python --version
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)
```

conda --version

imyday — -bash — 80×24

```
iMyday-MacBook-Pro:~ imyday$ python --version
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)
iMyday-MacBook-Pro:~ imyday$ conda --version
conda 4.3.14
iMyday-MacBook-Pro:~ imyday$ conda info --envs
# conda environments:
#
py27                     /Users/imyday/anaconda/envs/py27
py35                     /Users/imyday/anaconda/envs/py35
root                  *  /Users/imyday/anaconda

iMyday-MacBook-Pro:~ imyday$ source activate py35
(py35) iMyday-MacBook-Pro:~ imyday$ python --version
Python 3.5.3 :: Continuum Analytics, Inc.
(py35) iMyday-MacBook-Pro:~ imyday$ conda --version
conda 4.3.14
(py35) iMyday-MacBook-Pro:~ imyday$ source deactivate py35
iMyday-MacBook-Pro:~ imyday$ conda info --envs
# conda environments:
#
py27                     /Users/imyday/anaconda/envs/py27
py35                     /Users/imyday/anaconda/envs/py35
root                  *  /Users/imyday/anaconda
```

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

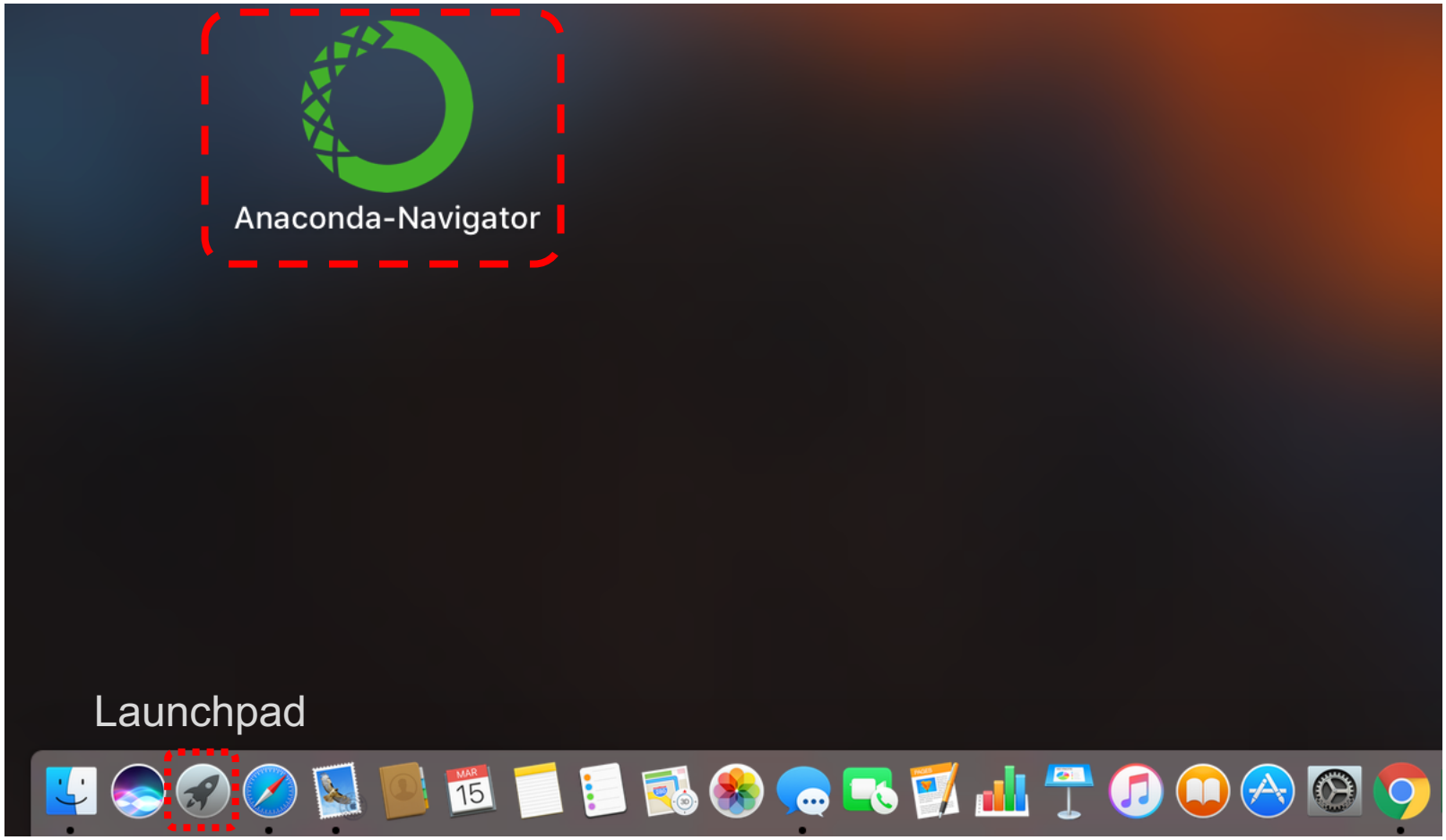
source activate py35

source deactivate py35

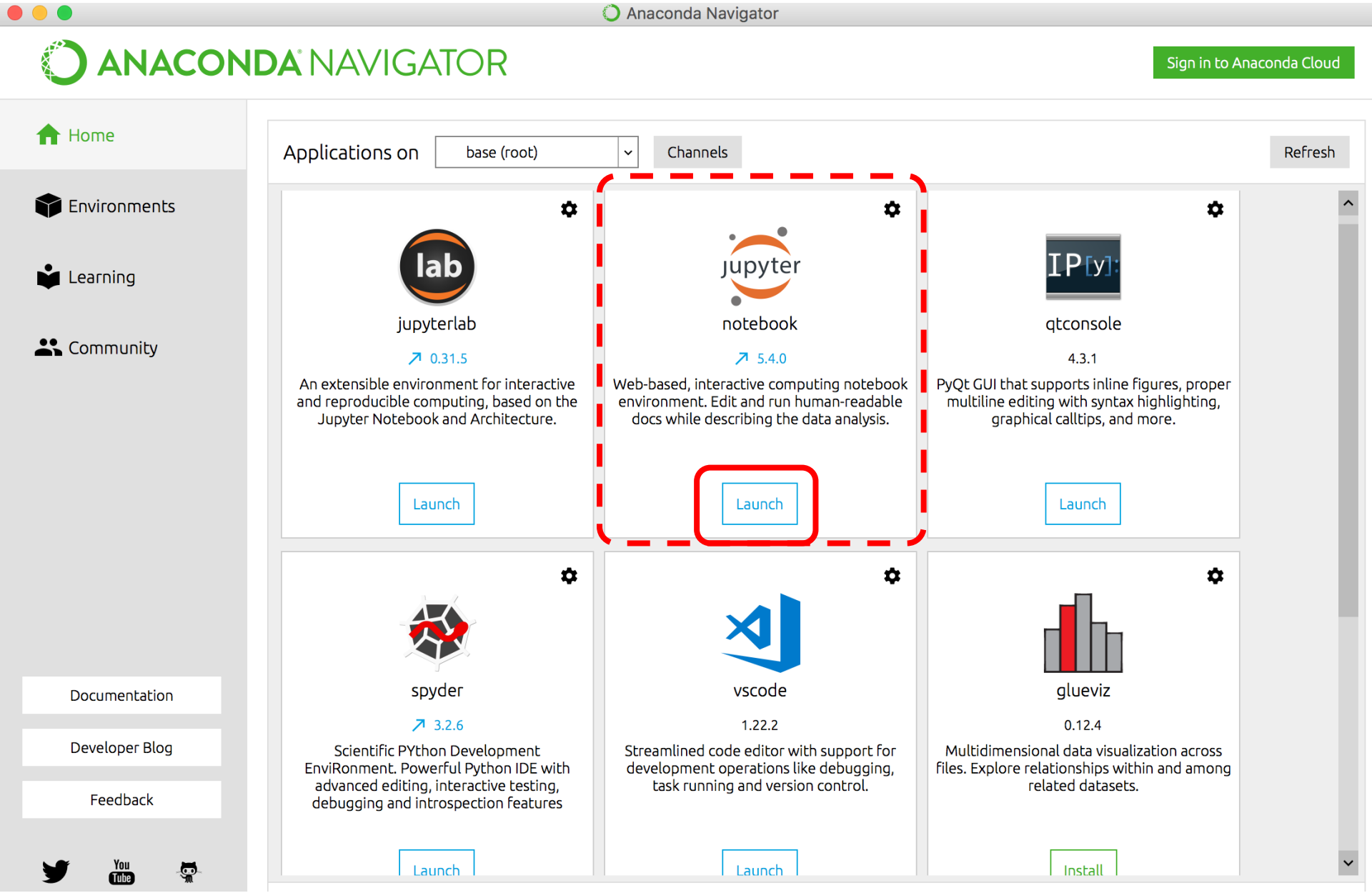
Python

HelloWorld

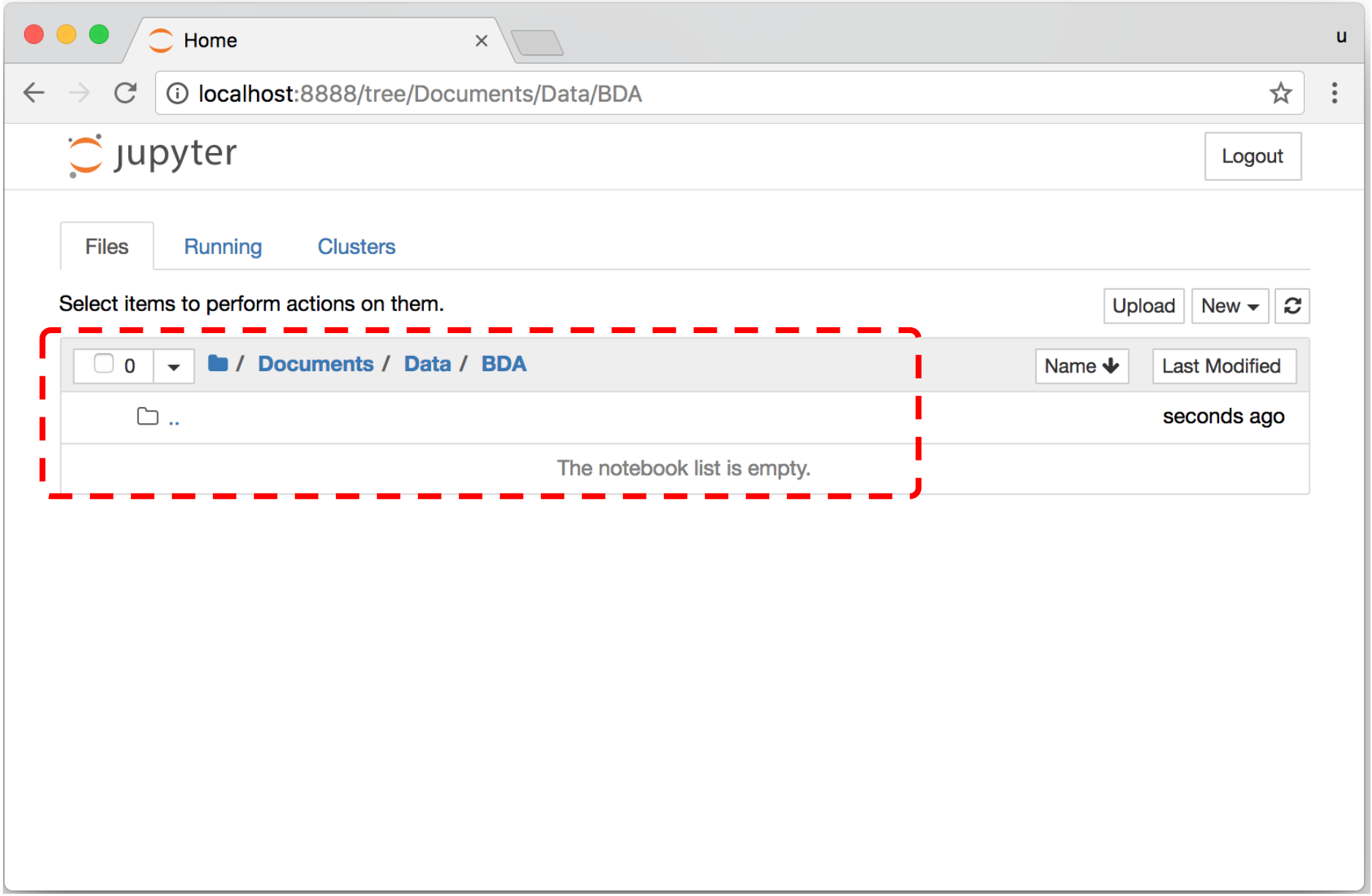
Anaconda-Navigator



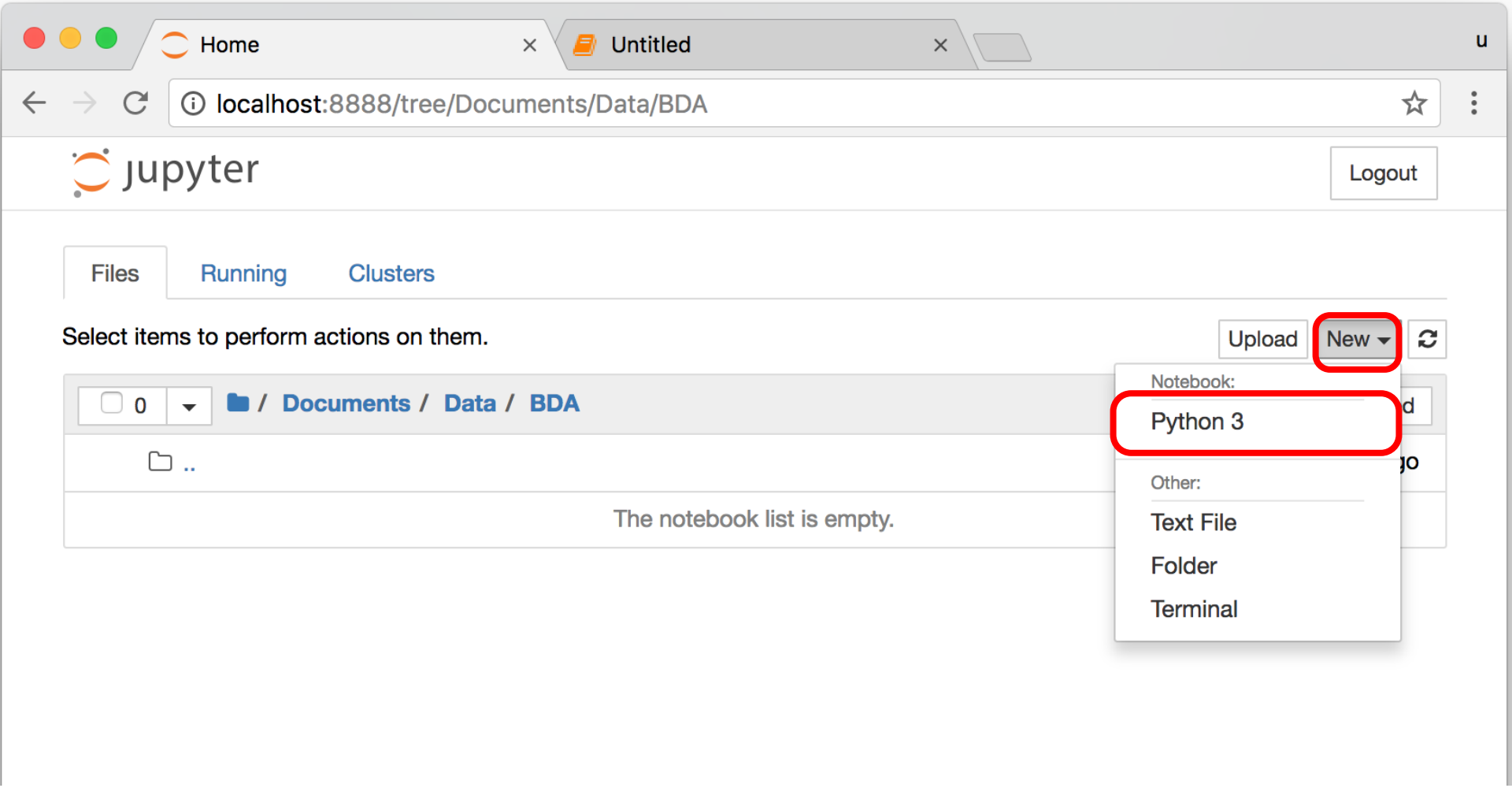
Anaconda Navigator



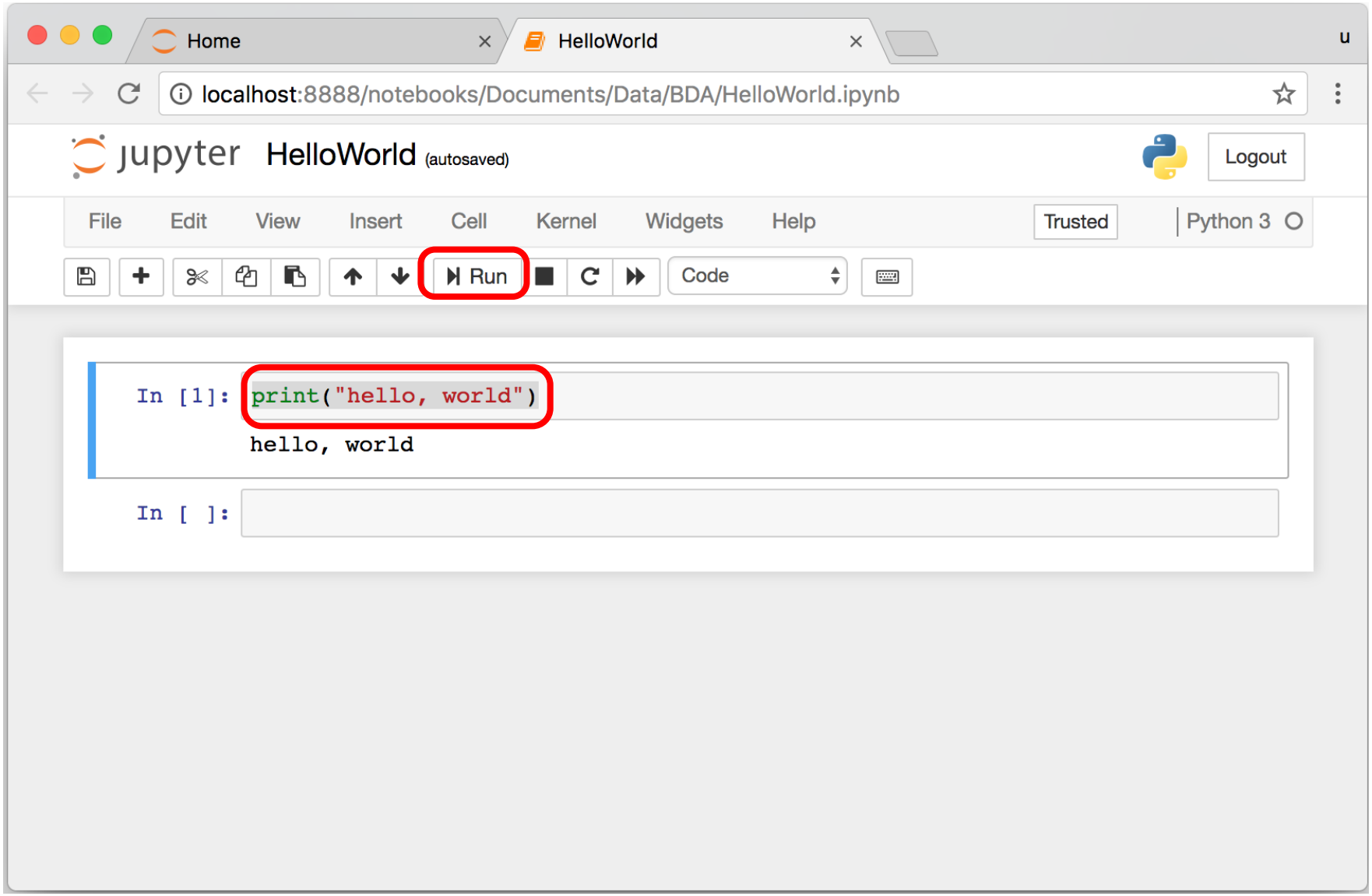
Jupyter Notebook



Jupyter Notebook
New Python 3



print("hello, world")



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

Home | HelloWorld

localhost:8888/notebooks/Documents/Data/BDA/HelloWorld.ipynb

jupyter HelloWorld (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run | Code

In [1]:
```
print("hello, world")
```

```
hello, world
```

In [2]:
```
from platform import python_version
print("Python Version:", python_version())
```

```
Python Version: 3.6.5
```

In [ ]:

Python

Programming

python™

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

Python Fiddle

Python Cloud IDE | Python Fid

pythonfiddle.com

Run | Reset | Share | Import | Login | Language

Python Fiddle

Python Cloud IDE

G+1 2.6k

Examples

- Chaining comparison operators
- Decorators
- Creating generators objects
- Enumerate
- Function closure
- Lex tokenizer
- Step argument in slice operators
- For Else
- Verbose regular expressions
- In-place value swapping
- Function argument unpacking

Packages

Hotkeys

```
print("Hello Python Fiddle")
```

Title:

Description:

Tags:

A comma-separated list of tags.

Save

Hello Python Fiddle

http://pythonfiddle.com/

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/

Python in Google Colab

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT

python101.ipynb - Colaboratory

https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT?authuser=2#scrollTo=LSNfVqJO-7-U

python101.ipynb

File Edit View Insert Runtime Tools Help

COMMENT SHARE A

CODE TEXT CELL CELL CONNECTED EDITING

```
print("hello, world")
```

```
hello, world
```

[2]

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

```
Python Version: 3.6.6
```

[3]

```
# https://www.learnpython.org/en/
# LearnPython.org interactive Python tutorial
print("Hello World")
print("Hello World\nThis is a message")
x = 3
print(x)
x = 2
y = 3
print(x, ' ', y)
```

```
Hello World
Hello World
This is a message
3
2   3
```

[4]

```
# Python Variables
x = 2
price = 2.5
word = 'Hello'

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

Text input and output

PythonTutorial

localhost:8889/notebooks/Documents/SCDBA/PythonTutorial.ipynb

jupyter PythonTutorial (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Python 3

Code CellToolbar

```
In [1]: print("Hello World")

        Hello World

In [2]: print("Hello World\nThis is a message")

        Hello World
        This is a message

In [3]: x = 3
        print(x)

        3

In [4]: x = 2
        y = 3
        print(x, ' ', y)

        2   3

In [5]: name = input("Enter a name: ")

        Enter a name: Myday

In [6]: x = int(input("What is x? "))

        What is x? 80

In [7]: x = float(input("Write a number "))

        Write a number 3.6
```

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/

Future value of a specified principal amount, rate of interest, and a number of years

Source: https://www.w3resource.com/python-exercises/python-basic-exercise-39.php

Future Value (FV)

```
# How much is your $100 worth after 7 years?
print(100 * 1.1 ** 7)
# output = 194.87
```

```
print(100 * 1.1 ** 7)
```

```
194.87171000000012
```

Source: https://www.w3resource.com/python-exercises/python-basic-exercise-39.php

Future Value (FV)

```
pv = 100
r = 0.1
n = 7

fv = pv * ((1 + (r)) ** n)
print(round(fv, 2))
```

```
pv = 100
r = 0.1
n = 7

fv = pv * ((1 + (r)) ** n)
print(round(fv, 2))
```

```
194.87
```

Future Value (FV)

```
amount = 100
interest = 10 #10% = 0.01 * 10
years = 7

future_value = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

```
amount = 100
interest = 10 #10% = 0.01 * 10
years = 7

future_value  = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

```
194.87
```

Source: https://www.w3resource.com/python-exercises/python-basic-exercise-39.php

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/

if elif else

```
score = 90
grade = ""
if score >=90:
    grade = "A"
elif score >= 80:
    grade = "B"
elif score >= 70:
    grade = "C"
elif score >= 60:
    grade = "D"
else:
    grade = "E"
print(grade)
# grade = "A"
```

```
A
```

http://pythontutor.com/visualize.html
https://goo.gl/E6w5ph

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/

while loops

```
age = 10

while age < 20:
    print(age)
    age = age + 1
```

```
10
11
12
13
14
15
16
17
18
19
```

Source: https://learnpython.trinket.io/learn-python-part-8-loops#/while-loops/about-while-loops

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.

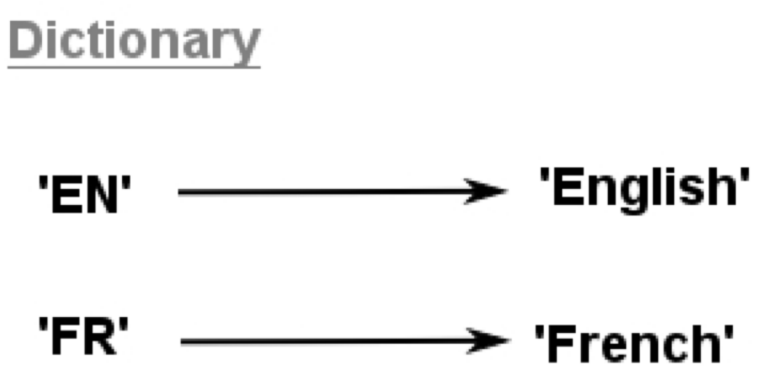
```
x = (10, 20, 30, 40, 50)
print(x[0])
print(x[1])
print(x[2])
print(x[-1])
```

```
10
20
30
50
```

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

Dictionary

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



English

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

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/

File Input / Output

```
with open('myfile.txt', 'w') as file:
    file.write('Hello World\nThis is Python File Input Output')

with open('myfile.txt', 'r') as file:
    text = file.read()
print(text)
```

```
with open('myfile.txt', 'w') as file:
    file.write('Hello World\nThis is Python File Input Output')
```

```
with open('myfile.txt', 'r') as file:
    text = file.read()
print(text)
```

```
Hello World
This is Python File Input Output
```

```
text
```

```
'Hello World\nThis is Python File Input Output'
```

Source: https://github.com/TiesdeKok/LearnPythonforResearch/blob/master/0_python_basics.ipynb

File Input / Output

```
with open('myfile.txt', 'a+') as file:
    file.write('\n' + 'New line')

with open('myfile.txt', 'r') as file:
    text = file.read()
print(text)
```

```
with open('myfile.txt', 'a+') as file:
    file.write('\n' + 'New line')
```

```
with open('myfile.txt', 'r') as file:
    text = file.read()
print(text)
```

```
Hello World
This is Python File Input Output
New line
```

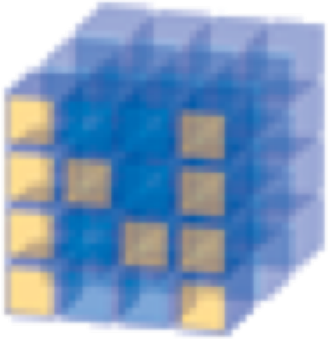
Source: https://github.com/TiesdeKok/LearnPythonforResearch/blob/master/0_python_basics.ipynb

Big Data Analytics with

Numpy

in Python

Numpy



NumPy
Base
N-dimensional array
package

NumPy

is the

fundamental package

for

scientific computing

with Python.

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

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

Scipy.org

NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

NumPy is licensed under the *BSD license*, enabling reuse with few restrictions.

Getting Started

- Getting NumPy
- Installing the SciPy Stack
- NumPy and SciPy documentation page
- NumPy Tutorial
- NumPy for MATLAB© Users
- NumPy functions by category
- NumPy Mailing List

For more information on the SciPy Stack (for which NumPy provides the fundamental array data structure), see scipy.org.

- About NumPy
- License
- Old array packages

http://www.numpy.org/

NumPy

NumPy ndarray

One-dimensional Array (1-D Array)

0	1			n-1
1	**2**	**3**	**4**	**5**

Two-dimensional Array (2-D Array)

	0	1			n-1
0	**1**	**2**	**3**	**4**	**5**
1	**6**	**7**	**8**	**9**	**10**
	11	**12**	**13**	**14**	**15**
m-1	**16**	**17**	**18**	**19**	**20**

NumPy

NumPy

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

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

NumPy

Base
N-dimensional
array package

```
v = list(range(1, 6))
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 Create Array

```
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])
```

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

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.]]
[[0.66258211 0.65552598]
 [0.00429934 0.21695824]]
```

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

Numpy Quickstart Tutorial

SciPy.org Sponsored By ENTHOUGHT

Scipy.org | Docs | NumPy v1.13.dev0 Manual | NumPy User Guide

index | next | previous

Quickstart tutorial

Prerequisites

Before reading this tutorial you should know a bit of Python. If you would like to refresh your memory, take a look at the Python tutorial.

If you wish to work the examples in this tutorial, you must also have some software installed on your computer. Please see http://scipy.org/install.html for instructions.

The Basics

NumPy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called *axes*. The number of axes is *rank*.

For example, the coordinates of a point in 3D space `[1, 2, 1]` is an array of rank 1, because it has one axis. That axis has a length of 3. In the example pictured below, the array has rank 2 (it is 2-dimensional). The first dimension (axis) has a length of 2, the second dimension has a length of 3.

```
[[ 1., 0., 0.],
 [ 0., 1., 2.]]
```

NumPy's array class is called `ndarray`. It is also known by the alias `array`. Note that `numpy.array` is not the same as the Standard Python Library class `array.array`, which only handles one-dimensional arrays and offers less functionality. The more important attributes of an `ndarray` object are:

ndarray.ndim

the number of axes (dimensions) of the array. In the Python world, the number of dimensions is referred to as *rank*.

ndarray.shape

Table Of Contents

- Quickstart tutorial
  - Prerequisites
  - The Basics
    - An example
    - Array Creation
    - Printing Arrays
    - Basic Operations
    - Universal Functions
    - Indexing, Slicing and Iterating
  - Shape Manipulation
    - Changing the shape of an array
    - Stacking together different arrays
    - Splitting one array into several smaller ones
  - Copies and Views
    - No Copy at All
    - View or Shallow Copy
    - Deep Copy
    - Functions and Methods Overview
  - Less Basic
    - Broadcasting rules
  - Fancy indexing and index tricks
    - Indexing with Arrays of

https://docs.scipy.org/doc/numpy-dev/user/quickstart.html

```
import numpy as np
a = np.arange(15).reshape(3, 5)
```

```
a.shape
a.ndim
a.dtype.name
```

```
import numpy as np
a = np.arange(15).reshape(3, 5)
a
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14]])
```

```
print(a.shape)
```

```
(3, 5)
```

```
a.ndim
```

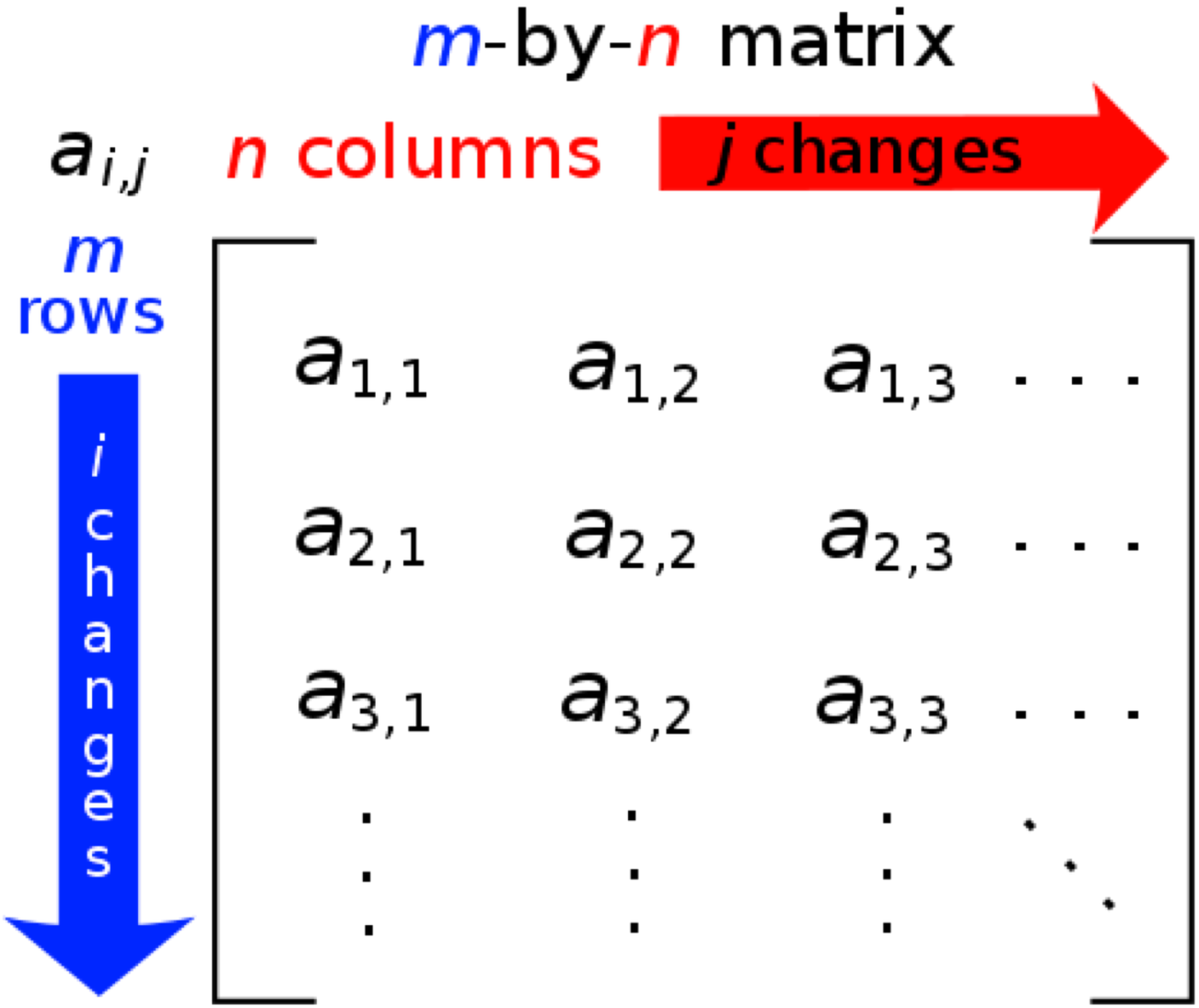
```
2
```

```
a.dtype.name
```

```
'int64'
```

Source: https://docs.scipy.org/doc/numpy-dev/user/quickstart.html

Matrix



Source: https://simple.wikipedia.org/wiki/Matrix_(mathematics)

NumPy ndarray: Multidimensional Array Object

NumPy ndarray

One-dimensional Array (1-D Array)

0	1			n-1
1	**2**	**3**	**4**	**5**

Two-dimensional Array (2-D Array)

	0	1			n-1
0	**1**	**2**	**3**	**4**	**5**
1	**6**	**7**	**8**	**9**	**10**
	11	**12**	**13**	**14**	**15**
m-1	**16**	**17**	**18**	**19**	**20**

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

One-dimensional Array (1-D Array)

0	1			n-1
1	**2**	**3**	**4**	**5**

```
a = np.array([1,2,3,4,5])
a
```

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

```
a = np.array([[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15],[16,17,18,19,20]])
```

Two-dimensional Array (2-D Array)

	0	1			n-1
0	**1**	**2**	**3**	**4**	**5**
1	**6**	**7**	**8**	**9**	**10**
	11	**12**	**13**	**14**	**15**
m-1	**16**	**17**	**18**	**19**	**20**

```
a = np.array([[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15],[16,17,18,19,20]])
a
```

```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15],
       [16, 17, 18, 19, 20]])
```

```
import numpy as np
a = np.array([[0, 1, 2, 3],
[10, 11, 12, 13],
[20, 21, 22, 23]])
a
```

0	1	2	3
10	11	12	13
20	21	22	23

a = np.array([[0, 1, 2, 3], [10, 11, 12, 13], [20, 21, 22, 23]])

```
a = np.array([[0, 1, 2, 3], [10, 11, 12, 13], [20, 21, 22, 23]])
a
```

```
array([[ 0,  1,  2,  3],
       [10, 11, 12, 13],
       [20, 21, 22, 23]])
```

```
print(a.ndim)
```

```
2
```

```
print(a.shape)
```

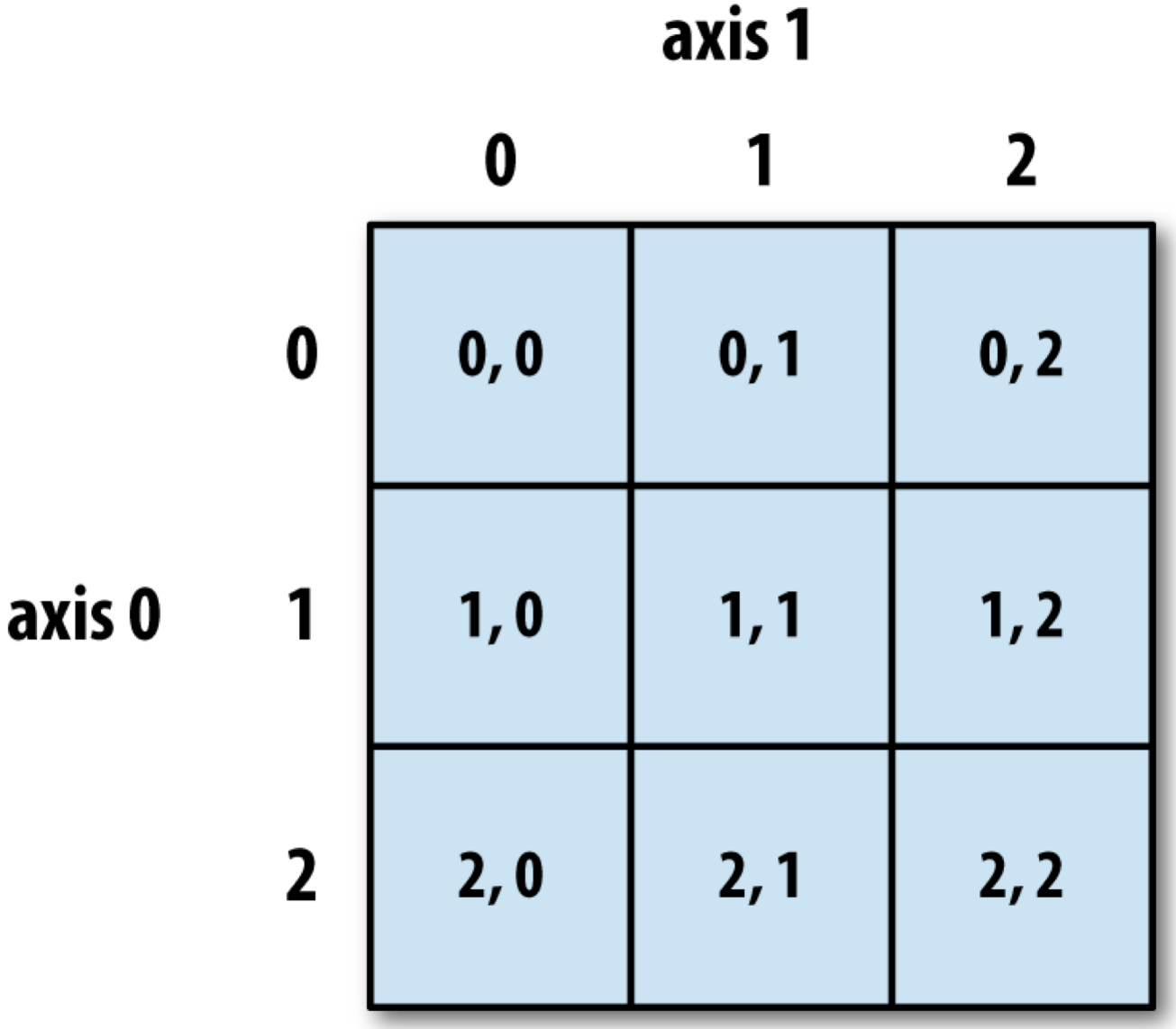
```
(3, 4)
```

0	1	2	3
10	11	12	13
20	21	22	23

NumPy Basics: Arrays and Vectorized Computation

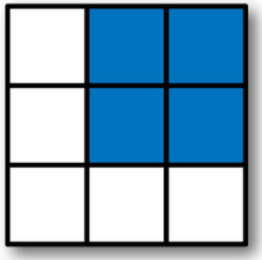
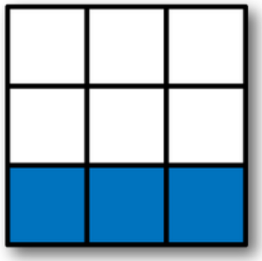
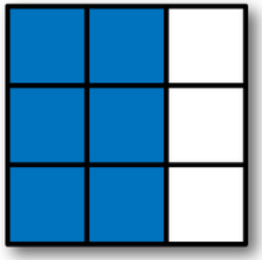
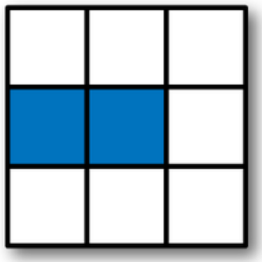
Source: https://www.safaribooksonline.com/library/view/python-for-data/9781449323592/ch04.html

NumPy Array



Source: https://www.safaribooksonline.com/library/view/python-for-data/9781449323592/ch04.html

Numpy Array

	Expression	Shape
	`arr[:2, 1:]`	`(2, 2)`
	`arr[2]`	`(3,)`
	`arr[2, :]`	`(3,)`
	`arr[2:, :]`	`(1, 3)`
	`arr[:, :2]`	`(3, 2)`
	`arr[1, :2]`	`(2,)`
	`arr[1:2, :2]`	`(1, 2)`

Source: https://www.safaribooksonline.com/library/view/python-for-data/9781449323592/ch04.html

Wes McKinney (2017), "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython", 2nd Edition, O'Reilly Media.

Materials and IPython notebooks for "Python for Data Analysis" by Wes McKinney, published by O'Reilly Media

52 commits | 2 branches | 0 releases | 6 contributors

Branch: 2nd-edition | New pull request | Find file | Clone or download

betatim committed with **wesm** Add requirements (#71) — Latest commit ea47998 5 days ago

datasets	Add Kaggle titanic dataset	5 months ago
examples	Remove sex column from tips dataset	4 months ago
.gitignore	Add gitignore	2 years ago
COPYING	Use MIT license for code examples	a month ago
README.md	Add launch in Azure Notebooks button (#70)	19 days ago
appa.ipynb	Make more cells markdown instead of raw	a month ago
ch02.ipynb	Make more cells markdown instead of raw	a month ago
ch03.ipynb	Make more cells markdown instead of raw	a month ago
ch04.ipynb	Convert all notebooks to v4 format	a month ago
ch05.ipynb	Make more cells markdown instead of raw	a month ago
ch06.ipynb	Make more cells markdown instead of raw	a month ago
ch07.ipynb	Convert all notebooks to v4 format	a month ago
ch08.ipynb	Make more cells markdown instead of raw	a month ago
ch09.ipynb	Make more cells markdown instead of raw	a month ago
ch10.ipynb	Make more cells markdown instead of raw	a month ago

https://github.com/wesm/pydata-book

Wes McKinney (2017), "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython", 2nd Edition, O'Reilly Media.

wesm / pydata-book

Watch 640 | Star 4,031 | Fork 4,348

Code | Issues 2 | Pull requests 0 | Projects 0 | Insights

Branch: 2nd-edition | pydata-book / ch04.ipynb | Find file | Copy path

wesm Convert all notebooks to v4 format c2780a0 on Sep 27

2 contributors

1857 lines (1856 sloc) | 32.6 KB | Raw | Blame | History

NumPy Basics: Arrays and

In [ ]:
```
import numpy as np
np.random.seed(12345)
import matplotlib.pyplot as plt
plt.rc('figure', figsize=(10, 6))
np.set_printoptions(precision=4, suppress=True)
```

In [ ]:
```
import numpy as np
my_arr = np.arange(1000000)
my_list = list(range(1000000))
```

In [ ]:
```
%time for _ in range(10): my_arr2 = my_arr * 2
%time for _ in range(10): my_list2 = [x * 2 for x in my_list]
```

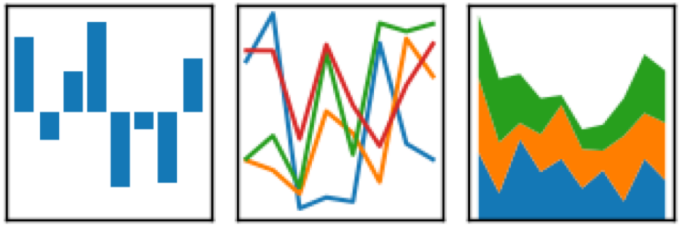
The NumPy ndarray: A Multidimensional Array Object

In [ ]:
```
import numpy as np
# Generate some random data
data = np.random.randn(2, 3)
data
```

Source: https://github.com/wesm/pydata-book/blob/2nd-edition/ch04.ipynb

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



Python Pandas

Python Pandas for Finance

Source: https://mapattack.wordpress.com/2017/02/12/using-python-for-stocks-1/

! pip install pandas_datareader

```
1 ! pip install pandas_datareader
```

```
Collecting pandas_datareader
  Downloading https://files.pythonhosted.org/packages/cc/5c/ea5b6dcfd0f55c5fb1e37fb45335ec01cceca199b8a79339137f5ed269e0/pandas_datarea
    100% |████████████████████████████████| 112kB 2.7MB/s
Collecting lxml (from pandas_datareader)
  Downloading https://files.pythonhosted.org/packages/03/a4/9eea8035fc7c7670e5eab97f34ff2ef0ddd78a491bf96df5accedb0e63f5/lxml-4.2.5-cp3
    100% |████████████████████████████████| 5.8MB 7.5MB/s
Requirement already satisfied: pandas>=0.19.2 in /usr/local/lib/python3.6/dist-packages (from pandas_datareader) (0.22.0)
Requirement already satisfied: requests>=2.3.0 in /usr/local/lib/python3.6/dist-packages (from pandas_datareader) (2.18.4)
Requirement already satisfied: wrapt in /usr/local/lib/python3.6/dist-packages (from pandas_datareader) (1.10.11)
Requirement already satisfied: python-dateutil>=2 in /usr/local/lib/python3.6/dist-packages (from pandas>=0.19.2->pandas_datareader) (2
Requirement already satisfied: numpy>=1.9.0 in /usr/local/lib/python3.6/dist-packages (from pandas>=0.19.2->pandas_datareader) (1.14.6)
Requirement already satisfied: pytz>=2011k in /usr/local/lib/python3.6/dist-packages (from pandas>=0.19.2->pandas_datareader) (2018.5)
Requirement already satisfied: idna<2.7,>=2.5 in /usr/local/lib/python3.6/dist-packages (from requests>=2.3.0->pandas_datareader) (2.6)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /usr/local/lib/python3.6/dist-packages (from requests>=2.3.0->pandas_datareader
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist-packages (from requests>=2.3.0->pandas_datareader) (
Requirement already satisfied: urllib3<1.23,>=1.21.1 in /usr/local/lib/python3.6/dist-packages (from requests>=2.3.0->pandas_datareader
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from python-dateutil>=2->pandas>=0.19.2->pandas_data
Installing collected packages: lxml, pandas-datareader
Successfully installed lxml-4.2.5 pandas-datareader-0.7.0
```

conda install pandas-datareader

imyday — -bash — 80×24

```
iMyday-MacBook-Pro:~ imyday$ conda install pandas-datareader
Fetching package metadata ...........
Solving package specifications: .

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

The following NEW packages will be INSTALLED:

    pandas-datareader: 0.2.1-py36_0
    requests-file:     1.4.1-py36_0

Proceed ([y]/n)? y

requests-file- 100% |################################| Time: 0:00:00   1.55 MB/s
pandas-datarea 100% |################################| Time: 0:00:00 409.66 kB/s
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
```

AAPL
Adj Close
MA05
MA20
MA60
220
200
180
160
140
120
100
80
2016-02
2016-06
2016-10
2017-02
2017-06
2017-10
2018-02
2018-06
2018-10
Date

Finance Data from Yahoo Finance

```
# !pip install pandas_datareader
import pandas_datareader.data as web
import datetime as dt
#Read Stock Data from Yahoo Finance
end = dt.datetime(2017, 12, 31)
start = dt.datetime(2016, 1, 1)
df = web.DataReader("AAPL", 'yahoo', start, end)
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()
```

```
# !pip install pandas_datareader
import pandas as pd
import pandas_datareader.data as web
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
%matplotlib inline

#Read Stock Data from Yahoo Finance
end = dt.datetime.now()
#start = dt.datetime(end.year-2, end.month,
end.day)
start = dt.datetime(2016, 1, 1)
df = web.DataReader("AAPL", 'yahoo', start, end)
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()
```

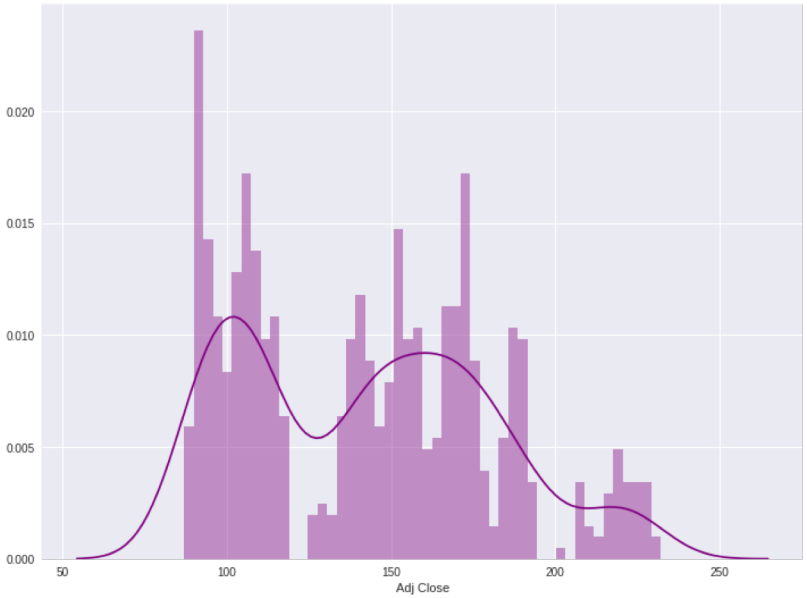
```
df['Adj Close'].plot(legend=True, figsize=(12, 8), title='AAPL’, label='Adj Close')
```

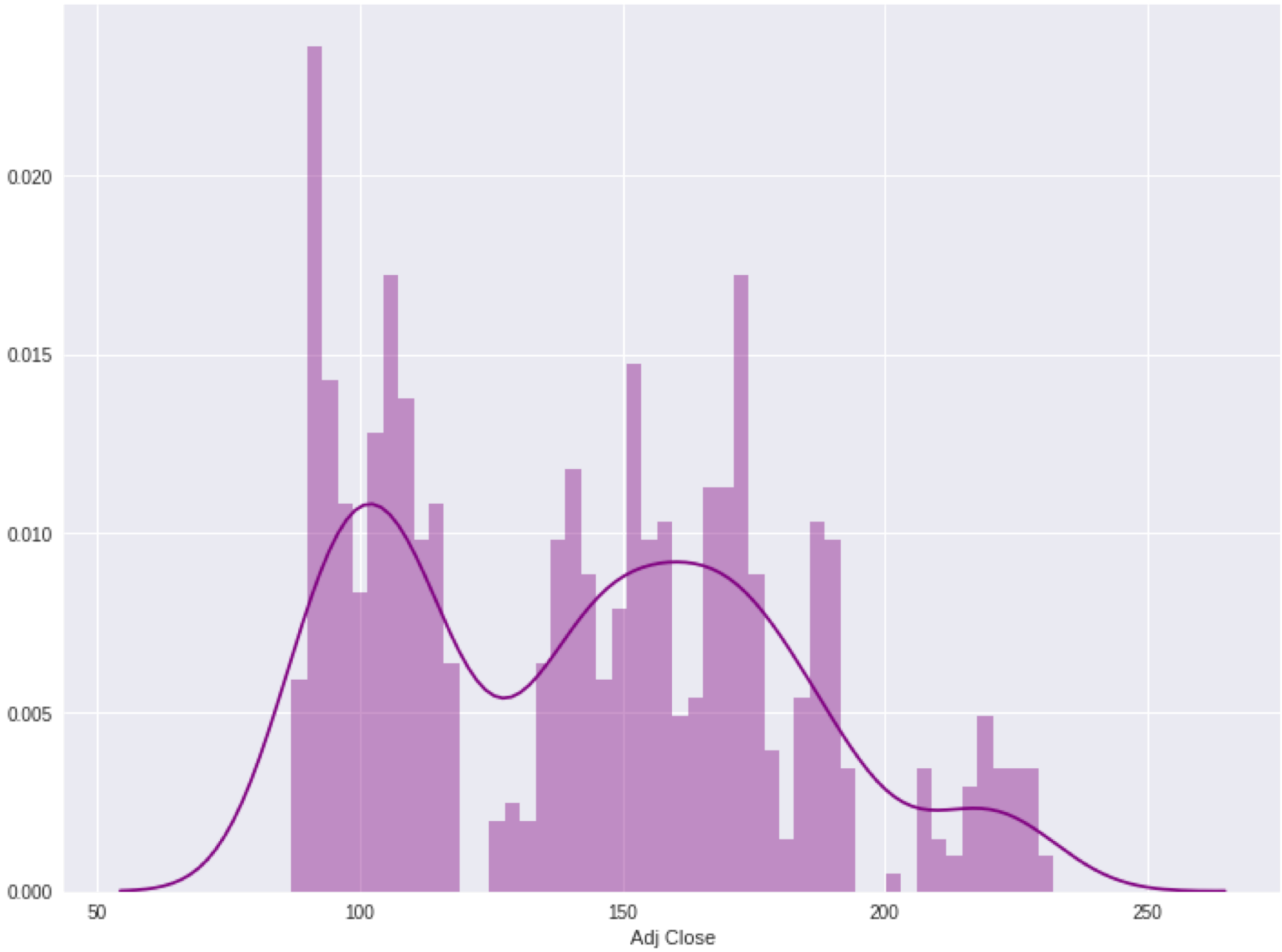
```
plt.figure(figsize=(12,9))
top = plt.subplot2grid((12,9), (0, 0),
rowspan=10, colspan=9)
bottom = plt.subplot2grid((12,9), (10,0),
rowspan=2, colspan=9)
top.plot(df.index, df['Adj Close'],
color='blue') #df.index gives the dates
bottom.bar(df.index, df['Volume'])
```

AAPL
Adj Close
220
200
180
160
140
120
100
80
1e8
Volume
1.0
0.5
0.0
2016-02
2016-06
2016-10
2017-02
2017-06
2017-10
2018-02
2018-06
2018-10

```
 # set the labels
top.axes.get_xaxis().set_visible(False)
top.set_title('AAPL')
top.set_ylabel('Adj Close')
bottom.set_ylabel('Volume')

plt.figure(figsize=(12,9))
sns.distplot(df['Adj Close'].dropna(),
bins=50, color='purple')
```




0.020
0.015
0.010
0.005
0.000
50
100
150
200
250
Adj Close

```
# simple moving averages
df['MA05'] = df['Adj Close'].rolling(5).mean()
#5 days
df['MA20'] = df['Adj
Close'].rolling(20).mean() #20 days
df['MA60'] = df['Adj
Close'].rolling(60).mean() #60 days
df2 = pd.DataFrame({'Adj Close': df['Adj
Close'],'MA05': df['MA05'],'MA20': df['MA20'],
'MA60': df['MA60']})
df2.plot(figsize=(12, 9), legend=True,
title='AAPL')
df2.to_csv('AAPL_MA.csv')
fig = plt.gcf()
fig.set_size_inches(12, 9)
fig.savefig('AAPL_plot.png', dpi=300)
plt.show()
```

AAPL
Adj Close
MA05
MA20
MA60
220
200
180
160
140
120
100
80
2016-02
2016-06
2016-10
2017-02
2017-06
2017-10
2018-02
2018-06
2018-10
Date

```
# !pip install pandas_datareader
import pandas as pd
import pandas_datareader.data as web
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
%matplotlib inline

#Read Stock Data from Yahoo Finance
end = dt.datetime.now()
#start = dt.datetime(end.year-2, end.month, end.day)
start = dt.datetime(2016, 1, 1)
df = web.DataReader("AAPL", 'yahoo', start, end)
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()

df['Adj Close'].plot(legend=True, figsize=(12, 8), title='AAPL', label='Adj Close')
plt.figure(figsize=(12,9))
top = plt.subplot2grid((12,9), (0, 0), rowspan=10, colspan=9)
bottom = plt.subplot2grid((12,9), (10,0), rowspan=2, colspan=9)
top.plot(df.index, df['Adj Close'], color='blue') #df.index gives the dates
bottom.bar(df.index, df['Volume'])

# set the labels
top.axes.get_xaxis().set_visible(False)
top.set_title('AAPL')
top.set_ylabel('Adj Close')
bottom.set_ylabel('Volume')

plt.figure(figsize=(12,9))
sns.distplot(df['Adj Close'].dropna(), bins=50, color='purple')

# simple moving averages
df['MA05'] = df['Adj Close'].rolling(5).mean() #5 days
df['MA20'] = df['Adj Close'].rolling(20).mean() #20 days
df['MA60'] = df['Adj Close'].rolling(60).mean() #60 days
df2 = pd.DataFrame({'Adj Close': df['Adj Close'],'MA05': df['MA05'],'MA20': df['MA20'], 'MA60': df['MA60']})
df2.plot(figsize=(12, 9), legend=True, title='AAPL')
df2.to_csv('AAPL_MA.csv')
fig = plt.gcf()
fig.set_size_inches(12, 9)
fig.savefig('AAPL_plot.png', dpi=300)
plt.show()
```

```python
# !pip install pandas_datareader
import pandas as pd
import pandas_datareader.data as web
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
%matplotlib inline

#Read Stock Data from Yahoo Finance
end = dt.datetime.now()
#start = dt.datetime(end.year-2, end.month, end.day)
start = dt.datetime(2016, 1, 1)
df = web.DataReader("AAPL", 'yahoo', start, end)
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()

df['Adj Close'].plot(legend=True, figsize=(12, 8), title='AAPL', label='Adj Close')
plt.figure(figsize=(12,9))
top = plt.subplot2grid((12,9), (0, 0), rowspan=10, colspan=9)
bottom = plt.subplot2grid((12,9), (10,0), rowspan=2, colspan=9)
top.plot(df.index, df['Adj Close'], color='blue') #df.index gives the dates
bottom.bar(df.index, df['Volume'])

# set the labels
top.axes.get_xaxis().set_visible(False)
top.set_title('AAPL')
top.set_ylabel('Adj Close')
bottom.set_ylabel('Volume')

plt.figure(figsize=(12,9))
sns.distplot(df['Adj Close'].dropna(), bins=50, color='purple')

# simple moving averages
df['MA05'] = df['Adj Close'].rolling(5).mean() #5 days
df['MA20'] = df['Adj Close'].rolling(20).mean() #20 days
df['MA60'] = df['Adj Close'].rolling(60).mean() #60 days
df2 = pd.DataFrame({'Adj Close': df['Adj Close'],'MA05': df['MA05'],'MA20': df['MA20'], 'MA60': df['MA60']})
df2.plot(figsize=(12, 9), legend=True, title='AAPL')
df2.to_csv('AAPL_MA.csv')
fig = plt.gcf()
fig.set_size_inches(12, 9)
fig.savefig('AAPL_plot.png', dpi=300)
plt.show()
```

Finance Data from Quandl

```
# ! pip install quandl
import quandl
# quandl.ApiConfig.api_key = "YOURAPIKEY"
df = quandl.get("WIKI/AAPL", start_date="2016-01-01", end_date="2017-12-31")
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()
```

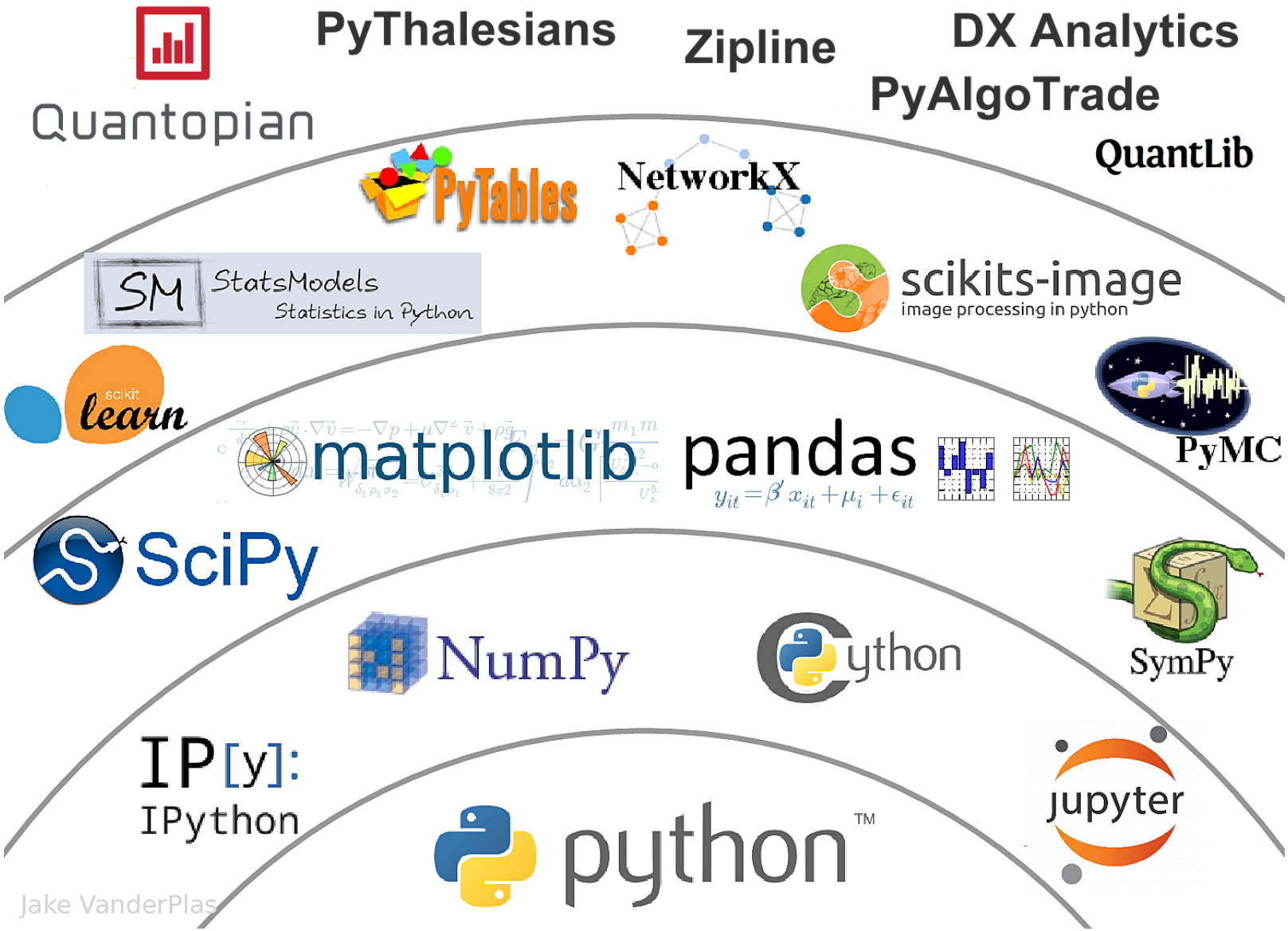
```
# ! pip install quandl
import quandl
# quandl.ApiConfig.api_key = "YOURAPIKEY"
df = quandl.get("WIKI/AAPL", start_date="2016-01-01", end_date="2017-12-31")
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:5: FutureWarning: from_csv is deprecated. Please use read_csv(...) instead
  """
```

	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close	Adj. Volume
Date												
2017-12-22	174.68	175.424	174.500	175.01	16052615.0	0.0	1.0	174.68	175.424	174.500	175.01	16052615.0
2017-12-26	170.80	171.470	169.679	170.57	32968167.0	0.0	1.0	170.80	171.470	169.679	170.57	32968167.0
2017-12-27	170.10	170.780	169.710	170.60	21672062.0	0.0	1.0	170.10	170.780	169.710	170.60	21672062.0
2017-12-28	171.00	171.850	170.480	171.08	15997739.0	0.0	1.0	171.00	171.850	170.480	171.08	15997739.0
2017-12-29	170.52	170.590	169.220	169.23	25643711.0	0.0	1.0	170.52	170.590	169.220	169.23	25643711.0

Source: https://www.quandl.com/tools/python

The Quant Finance PyData Stack



Source: http://nbviewer.jupyter.org/format/slides/github/quantopian/pyfolio/blob/master/pyfolio/examples/overview_slides.ipynb#/5

Summary

- **Foundations of AI Investment Analysis in Python**
  - **Python**
    - Programming language
  - **Numpy**
    - Scientific computing
  - **Pandas**
    - Data structures and data analysis tools

References

- Wes McKinney (2017), "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython", 2nd Edition, O'Reilly Media. https://github.com/wesm/pydata-book
- Avinash Jain (2017), Introduction To Python Programming, Udemy, https://www.udemy.com/pythonforbeginnersintro/
- Alfred Essa (2015), Awesome Data Science: 1.0 Jupyter Notebook Tour, https://www.youtube.com/watch?v=e9cSF3eVQv0
- Ties de Kok (2017), Learn Python for Research, https://github.com/TiesdeKok/LearnPythonforResearch
- Ivan Idris (2015), Numpy Beginner's Guide, Third Edition, Packt Publishing
- Numpy Tutorial, https://docs.scipy.org/doc/numpy-dev/user/quickstart.html
- Wes McKinney (2012), Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, O'Reilly Media
- Yves Hilpisch (2014), Python for Finance: Analyze Big Financial Data, O'Reilly
- Michael Heydt (2015) , Mastering Pandas for Finance, Packt Publishing
- Michael Heydt (2015), Learning Pandas - Python Data Discovery and Analysis Made Easy, Packt Publishing
- James Ma Weiming (2015), Mastering Python for Finance, Packt Publishing
- Fabio Nelli (2015), Python Data Analytics: Data Analysis and Science using PANDAs, matplotlib and the Python Programming Language, Apress
- Wes McKinney (2013), 10-minute tour of pandas, https://vimeo.com/59324550
- Jason Wirth (2015), A Visual Guide To Pandas, https://www.youtube.com/watch?v=9d5-Ti6onew
- Edward Schofield (2013), Modern scientific computing and big data analytics in Python, PyCon Australia, https://www.youtube.com/watch?v=hqOsfS3dP9w
- Python Programming, https://pythonprogramming.net/
- Python, https://www.python.org/
- Python Programming Language, http://pythonprogramminglanguage.com/
- Numpy, http://www.numpy.org/
- Pandas, http://pandas.pydata.org/
- Skikit-learn, http://scikit-learn.org/
- Data School (2015), Machine learning in Python with scikit-learn, https://www.youtube.com/playlist?list=PL5-da3qGB5ICeMbQuqbbCOQWcS6OYBr5A
- Jason Brownlee (2016), Your First Machine Learning Project in Python Step-By-Step, https://machinelearningmastery.com/machine-learning-in-python-step-by-step/