

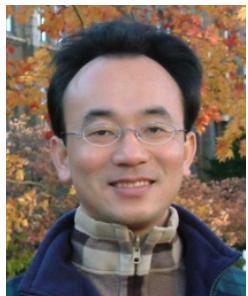
Big Data Analytics in Finance

Python 財務大數據分析基礎 (Foundations of Finance Big Data Analytics in Python)

1061BDAF06

MIS EMBA (M2322) (8605)

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/>

2017-11-02



課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2017/09/21	財務金融大數據分析課程介紹 (Course Orientation for Big Data Analytics in Finance)
2	2017/09/28	金融科技商業模式 (Business Models of Fintech)
3	2017/10/05	人工智慧投資分析與機器人理財顧問 (Artificial Intelligence for Investment Analysis and Robo-Advisors)
4	2017/10/12	金融科技對話式商務與智慧型交談機器人 (Conversational Commerce and Intelligent Chatbots for Fintech)
5	2017/10/19	事件研究法 (Event Study)
6	2017/10/26	財務金融大數據分析個案研究 I (Case Study on Big Data Analytics in Finance I)

課程大綱 (Syllabus)

週次 (Week) 日期 (Date) 內容 (Subject/Topics)

- | | | |
|----|------------|---|
| 7 | 2017/11/02 | <p>Python 財務大數據分析基礎
(Foundations of Finance Big Data Analytics in Python)</p> |
| 8 | 2017/11/09 | <p>Python Numpy大數據分析
(Big Data Analytics with Numpy in Python)</p> |
| 9 | 2017/11/16 | <p>Python Pandas 財務大數據分析
(Finance Big Data Analytics with Pandas in Python)</p> |
| 10 | 2017/11/23 | <p>期中報告 (Midterm Project Report)</p> |
| 11 | 2017/11/30 | <p>文字探勘分析技術與自然語言處理
(Text Mining Techniques and
Natural Language Processing)</p> |
| 12 | 2017/12/07 | <p>Python Keras深度學習
(Deep Learning with Keras in Python)</p> |

課程大綱 (Syllabus)

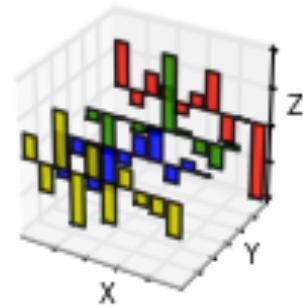
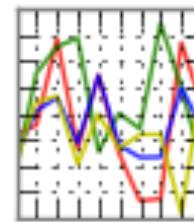
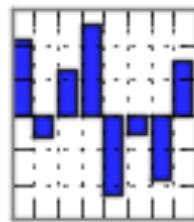
週次 (Week) 日期 (Date) 內容 (Subject/Topics)

- | | | |
|----|------------|--|
| 13 | 2017/12/14 | 財務金融大數據分析個案研究 II
(Case Study on Big Data Analytics in Finance II) |
| 14 | 2017/12/21 | TensorFlow深度學習
(Deep Learning with TensorFlow) |
| 15 | 2017/12/28 | 財務金融大數據深度學習
(Deep Learning for Finance Big Data) |
| 16 | 2018/01/04 | 社會網絡分析 (Social Network Analysis) |
| 17 | 2018/01/11 | 期末報告 I (Final Project Presentation I) |
| 18 | 2018/01/18 | 期末報告 II (Final Project Presentation II) |



pandas

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



Yahoo Finance

Secure | https://finance.yahoo.com

Home Mail Flickr Tumblr News Sports Finance Celebrity Answers Groups Mobile More ▾ Try Yahoo Finance on Firefox »

YAHOO!
FINANCE

Search for news, symbols or companies Sign in 1 ✉

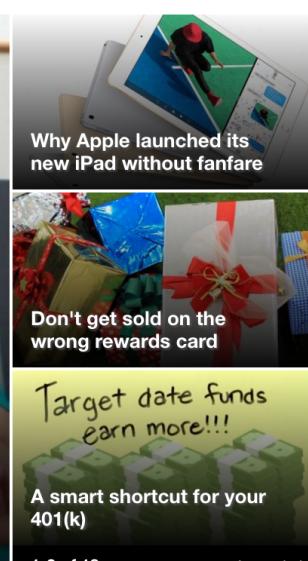
Finance Home Originals Events Personal Finance Technology Markets Industries NEW My Screeners My Portfolio

 Is solar power in your future? YAHOO! Search now

(⌚) US Markets are closed

S&P 500 2,344.02 -29.45 (-1.24%)	Dow 30 20,668.01 -237.85 (-1.14%)	Nasdaq 5,793.83 -107.70 (-1.83%)	Crude Oil 47.50 +0.16 (+0.34%)	Gold 1,244.80 -1.70 (-0.14%)	Silver 17.53 -0.05 (-0.30%)
--	---	--	--------------------------------------	------------------------------------	-----------------------------------

 Facebook head of counterterrorism: We need help
Monika Bickert might have one of the toughest jobs at Facebook right now. As Facebook's (FB) head of counterterrorism efforts and global product policy, she sets community standards and... 27

 Why Apple launched its new iPad without fanfare
Don't get sold on the wrong rewards card
Target date funds earn more!!!
A smart shortcut for your 401(k)

Quote Lookup

My Portfolio & Markets Customize

Recently Viewed >

Symbol	Last Price	Change	% Change
^TWII	9,874.79	-97.70	-0.98%
TSEC weighted index			
^IXIC	5,793.83	-107.70	-1.83%
Nasdaq			
^GSPC	2,344.02	-29.45	-1.24%
S&P 500			
^DJI	20,668.01	-237.85	-1.14%
Dow Jones Industrial Average			
GOOG	830.46	-17.94	-2.11%
Alphabet Inc.			
2330.TW	192.00	-3.00	-1.54%

<http://finance.yahoo.com/>

Apple Inc. (AAPL) - NasdaqGS



Search for news, symbols or companies

Search

Apple Inc. (AAPL)

NasdaqGS - NasdaqGS Delayed Price. Currency in USD

[Add to watchlist](#)

139.84 -1.62 (-1.15%)

At close: 4:00PM EDT

139.35 -0.49 (-0.35%)

After hours: 7:59PM EDT

[Summary](#)

[Conversations](#)

[Statistics](#)

[Profile](#)

[Financials](#)

[Options](#)

[Holders](#)

[Historical Data](#)

[Analysts](#)

Previous Close

141.46

Market Cap

733.68B

1D

5D

1M

6M

YTD

1Y

2Y

5Y

10Y

MAX

[Interactive chart](#)

Open

142.11

Beta

1.45

Bid

139.31 x 100

PE Ratio
(TTM)

16.79

Ask

139.40 x 1300

EPS (TTM)

8.33

Day's Range

139.73 - 142.80

Earnings Date
Apr 24, 2017 - Apr 28, 2017

52 Week Range

89.47 - 142.80

Dividend & Yield

2.28 (1.63%)

Volume

39,529,912

Ex-Dividend Date

N/A

Avg. Volume

26,889,183

1y Target Est

143.29

Trade prices are not sourced from all markets



<http://finance.yahoo.com/quote/AAPL?p=AAPL>

Yahoo Finance Charts: Apple Inc. (AAPL)

YAHOO! FINANCE

Go to Quote Summary Page



S&P 500

2,344.02

-29.45 (-1.24%)

Dow 30

20,668.01

-237.85 (-1.14%)

Nasdaq

5,793.83

-107.70 (-1.83%)

Crude Oil

47.50

+0.16 (+0.34%)

Gold

1,245.50

-1.00 (-0.08%)

Apple Inc. (AAPL) 139.84 -1.62 (-1.15%) As of 4:00PM EDT. Market closed.



AAPL 139.84

Open 142.11

Close 139.84

Low 139.73

High 142.80

Vol 39.53M

% Chg 63.27%

YAHOO!
FINANCE

Jan 7 '13

Jan 6 '14

Jan 5 '15

Jan 4 '16

Jan 2 '17

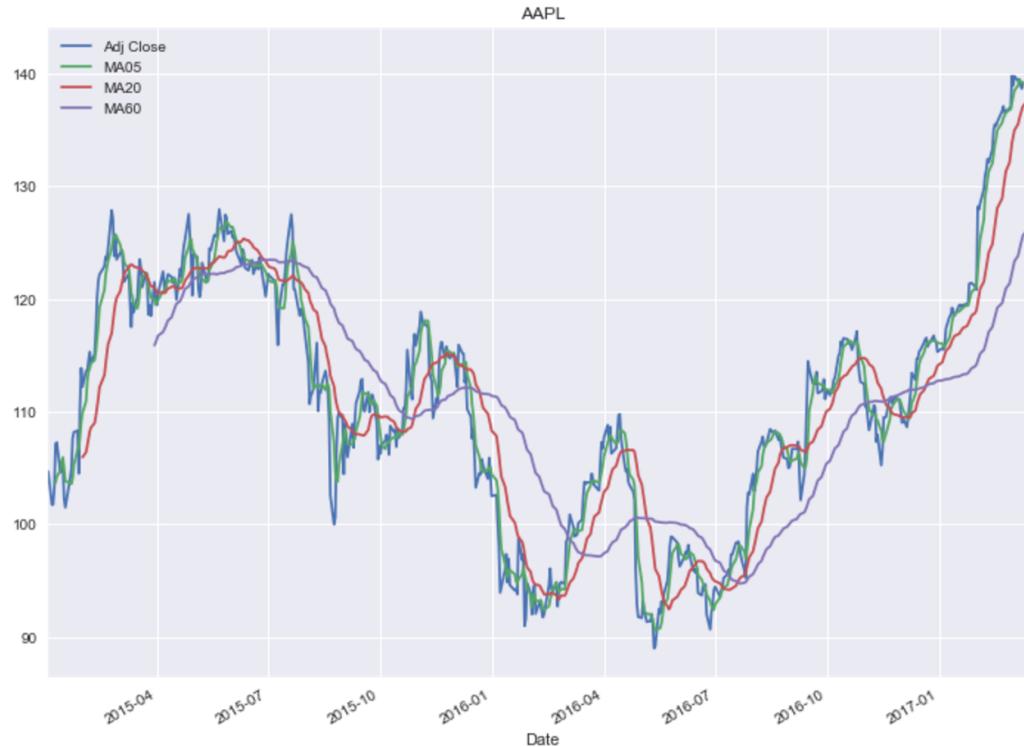
39.53M

<http://finance.yahoo.com/chart/AAPL>

Python Pandas for Finance

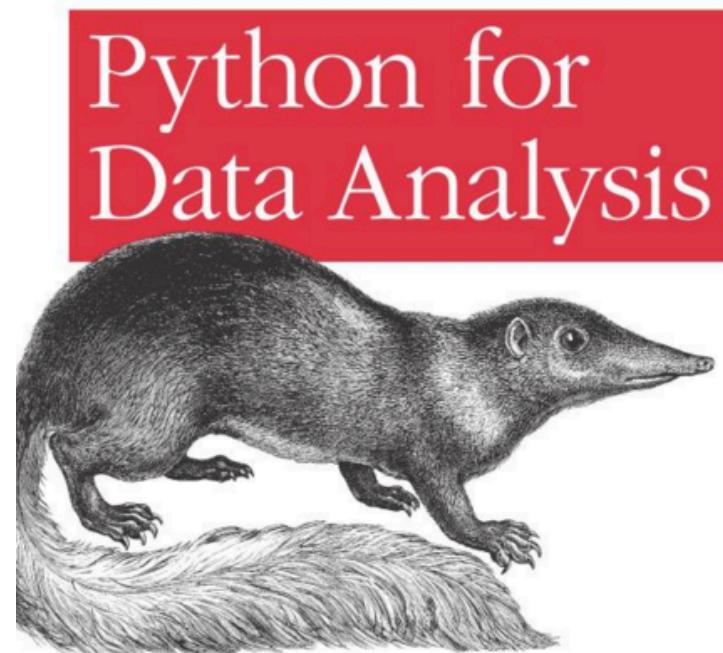
```
# simple moving averages
df['MA05'] = df['Adj Close'].rolling(5).mean() #5 days
df['MA20'] = df['Adj Close'].rolling(20).mean() #20 days
df['MA60'] = df['Adj Close'].rolling(60).mean() #60 days

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)
```



Wes McKinney (2012), Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, O'Reilly Media

Data Wrangling with Pandas, NumPy, and IPython

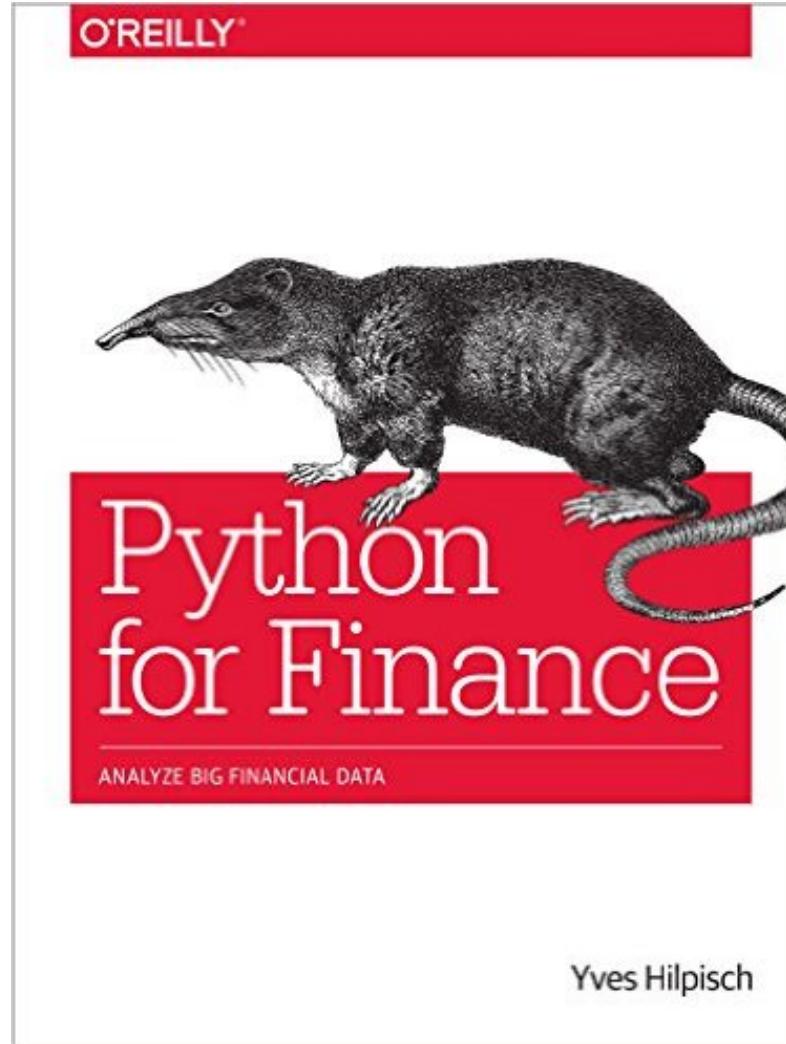


O'REILLY®

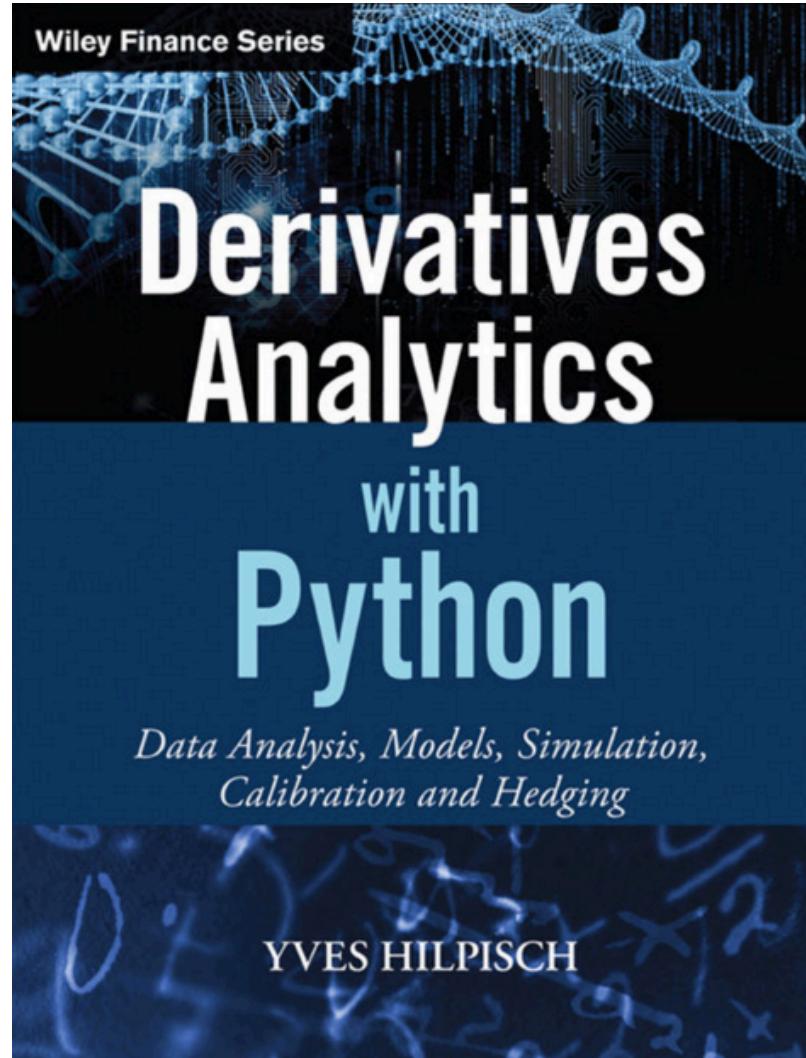
Copyrighted Material

Wes McKinney

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

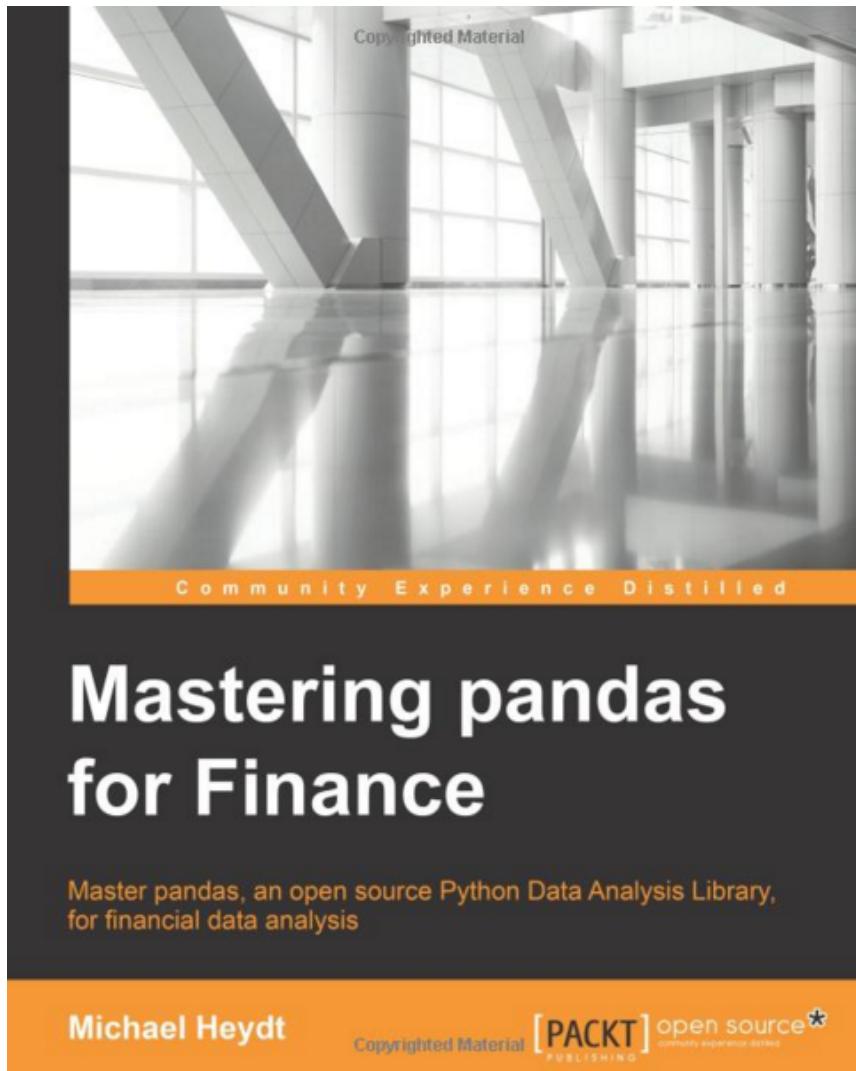


**Yves Hilpisch (2015),
Derivatives Analytics with Python:
Data Analysis, Models, Simulation, Calibration and Hedging, Wiley**



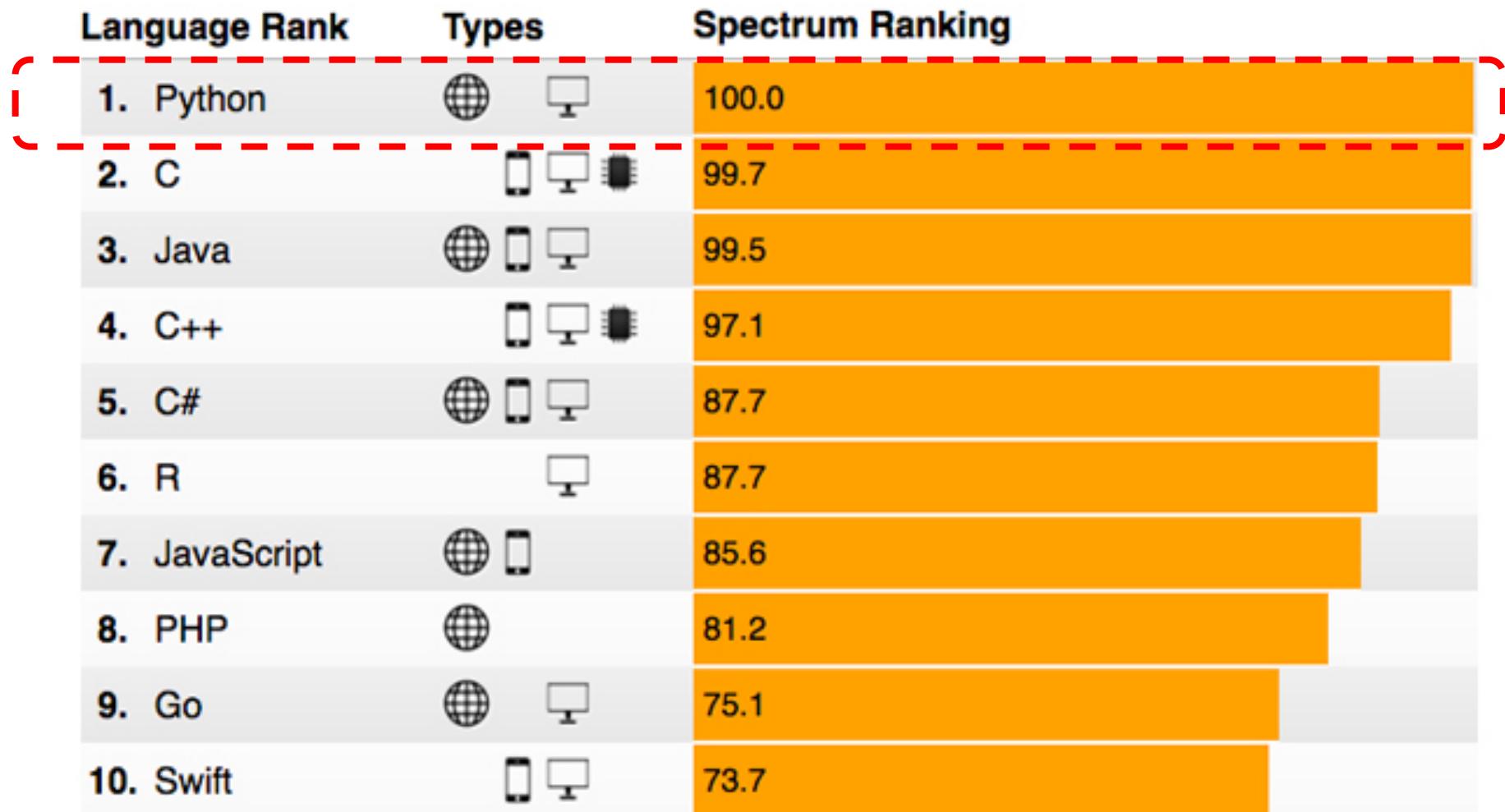
Source: <http://www.amazon.com/Derivatives-Analytics-Python-Simulation-Calibration/dp/1119037999/>

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

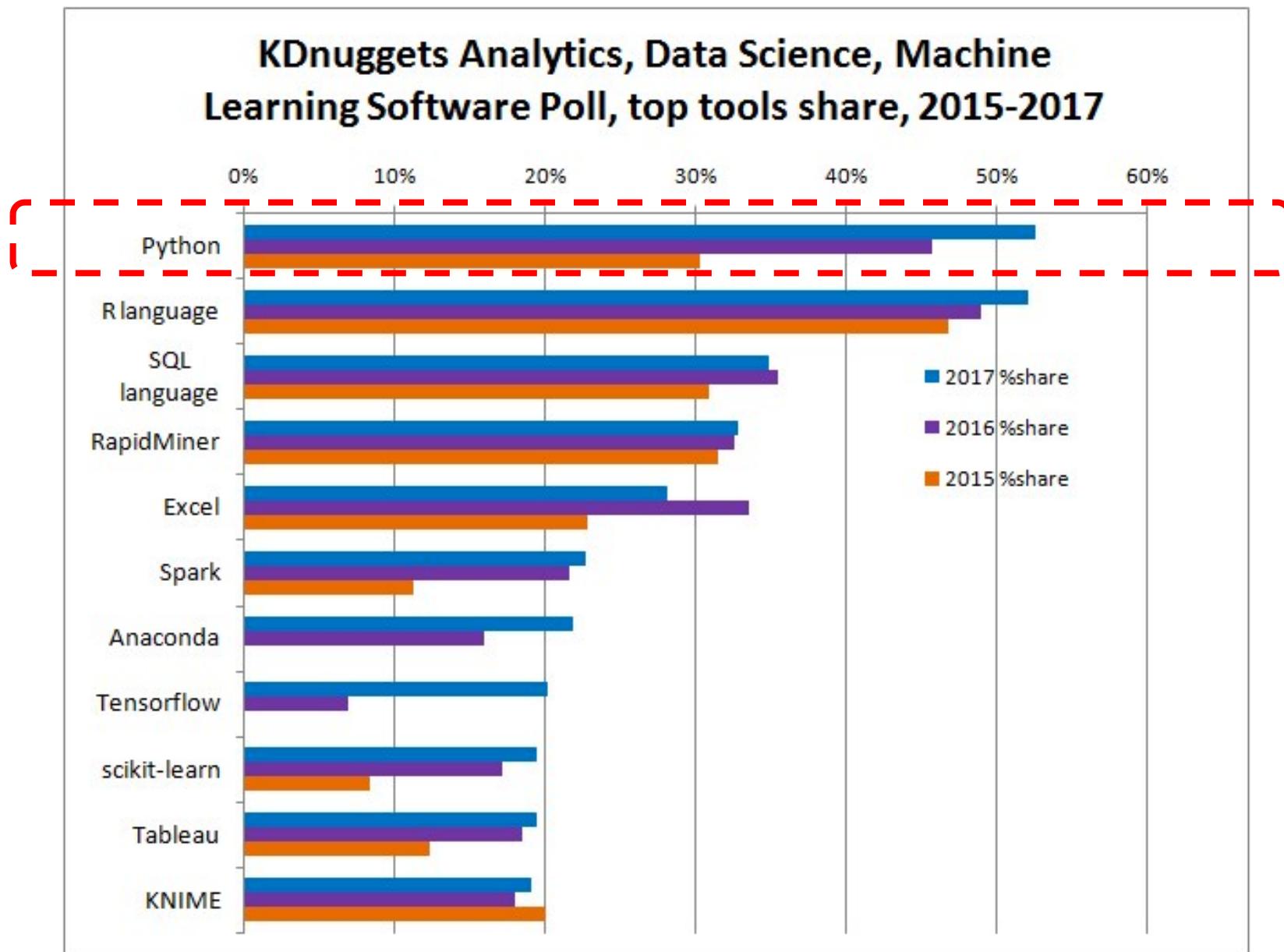


Source: <http://www.amazon.com/Mastering-Pandas-Finance-Michael-Heydt/dp/1783985100>

Python: The 2017 Top Programming Languages



Python: Top Analytics and Data Science Software

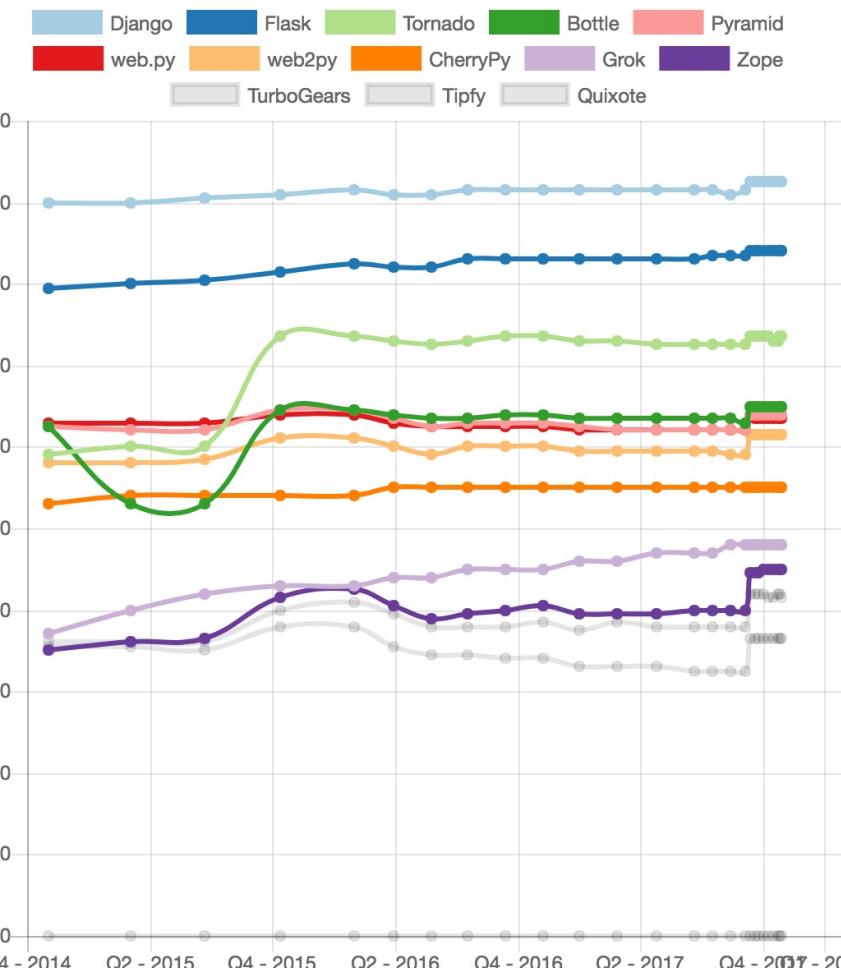


Python Hot Frameworks

[HotFrameworks](#)[Top Frameworks](#)[Rankings](#)[Languages ▾](#)[FAQ](#)

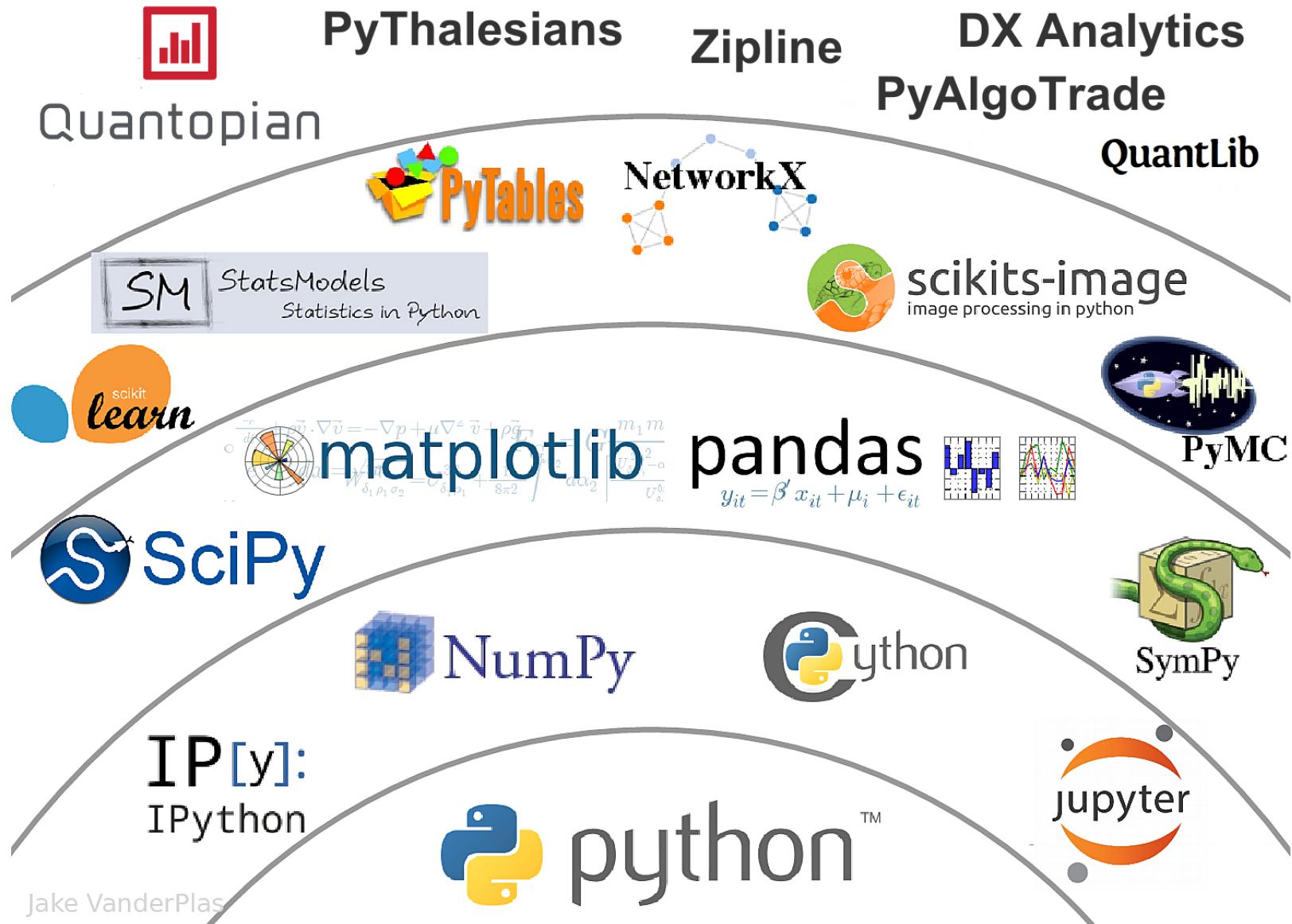
Python

Framework	Score
Django	92
Flask	84
Tornado	73
Bottle	65
Pyramid	64
web.py	63
web2py	61
CherryPy	55
Grok	48
Zope	45
TurboGears	41
Tipfy	36
Quixote	0



Source: <https://hotframeworks.com/languages/python>

The Quant Finance PyData Stack



Anaconda



[Log In](#) [Get Support](#) [Search](#) [Contact](#)

[PRODUCTS](#) [COMMUNITY](#) [CONSULTING](#) [TRAINING](#) [ABOUT](#) [RESOURCES](#)

SUPERPOWERS FOR PEOPLE WHO CHANGE THE WORLD

Leading Open Data Science Platform Powered by Python

[DOWNLOAD ANACONDA](#)



ACCELERATE

Time-to-Value



CONNECT

Data, Analytics & Compute



EMPOWER

Data Science Teams



<https://www.continuum.io/>



**Leading Open
Data Science
Platform
Powered by
Python**

Python

Python

PSF

Docs

PyPI

Jobs

Community



Search

GO

Socialize

Sign In

About

Downloads

Documentation

Community

Success Stories

News

Events

```
# Python 3: Simple output (with Unicode)
>>> print("Hello, I'm Python!")
Hello, I'm Python!

# Input, assignment
>>> name = input('What is your name?\n')
>>> print('Hi, %s.' % name)
What is your name?
Python
Hi, Python.
```



Quick & Easy to Learn

Experienced programmers in any other language can pick up Python very quickly, and beginners find the clean syntax and indentation structure easy to learn. [Whet your appetite](#) with our Python 3 overview.

1 2 3 4 5

Python is a programming language that lets you work quickly and integrate systems more effectively. [» Learn More](#)

Get Started

Download

Docs

Jobs

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

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**

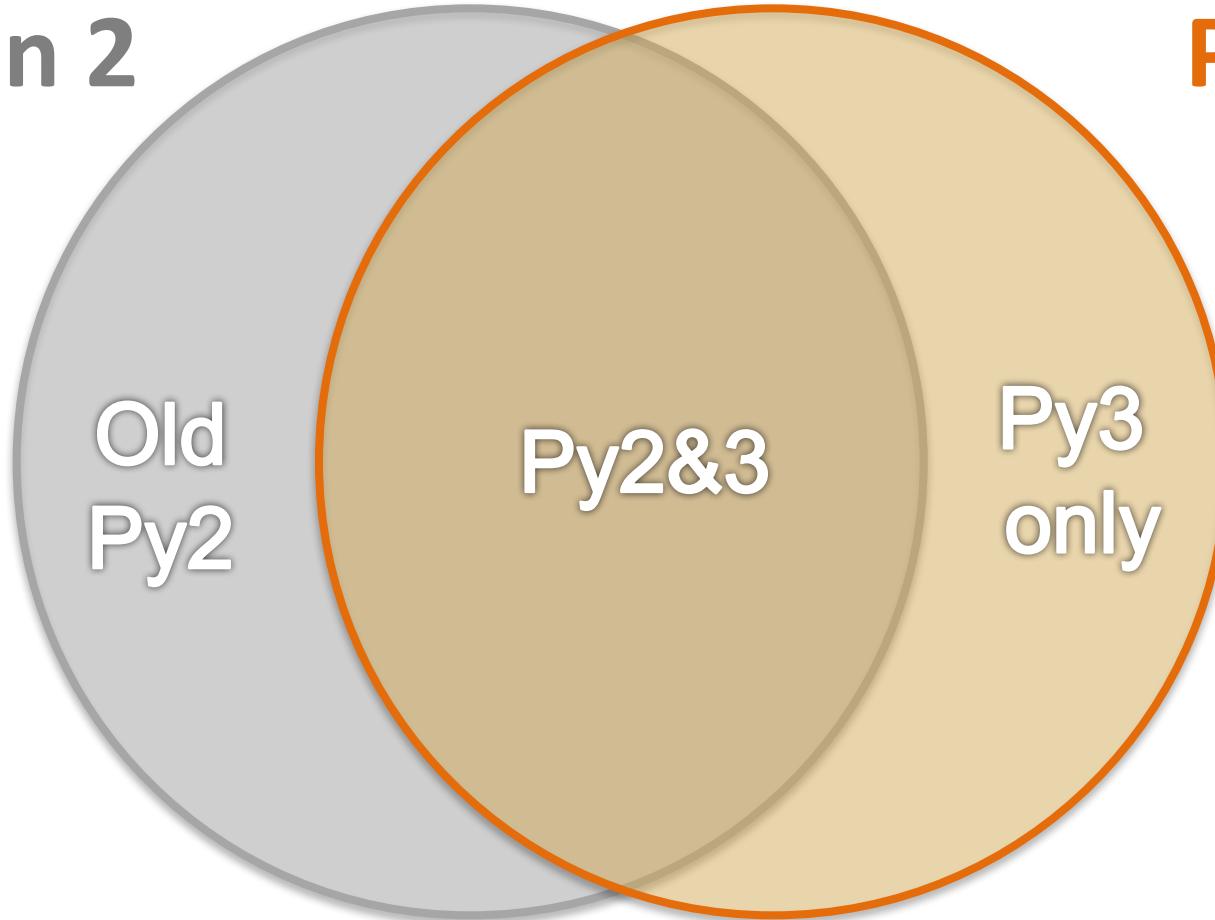
Python (Python 2.7 & Python 3.6)



Standard Syntax

Python 2

Python 3



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield

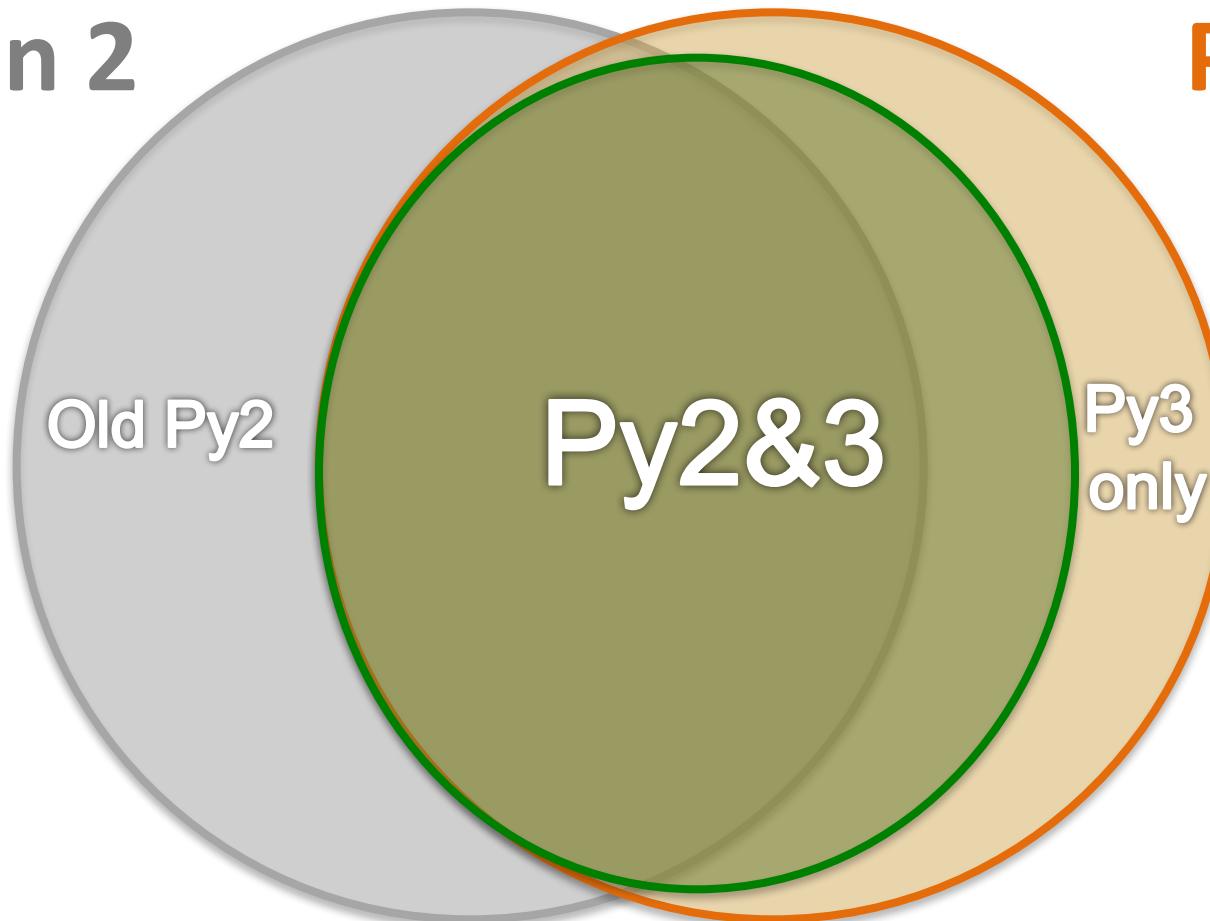
<https://www.youtube.com/watch?v=KOqk8j11aAI>

```
from __future__ import ...
```



Python 2

Python 3



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield

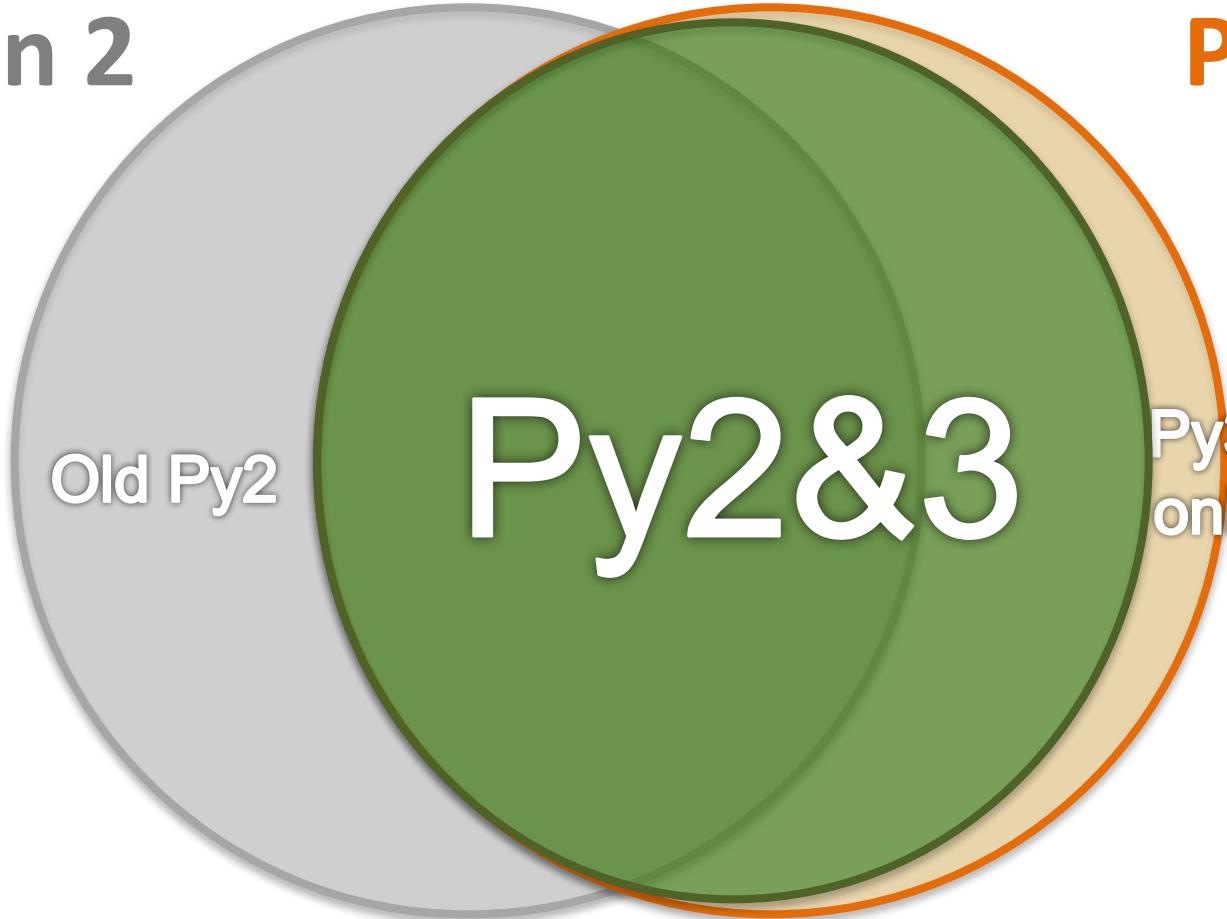
<https://www.youtube.com/watch?v=KOqk8j11aAI>

```
from future.builtins import *
```



Python 2

Python 3



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield

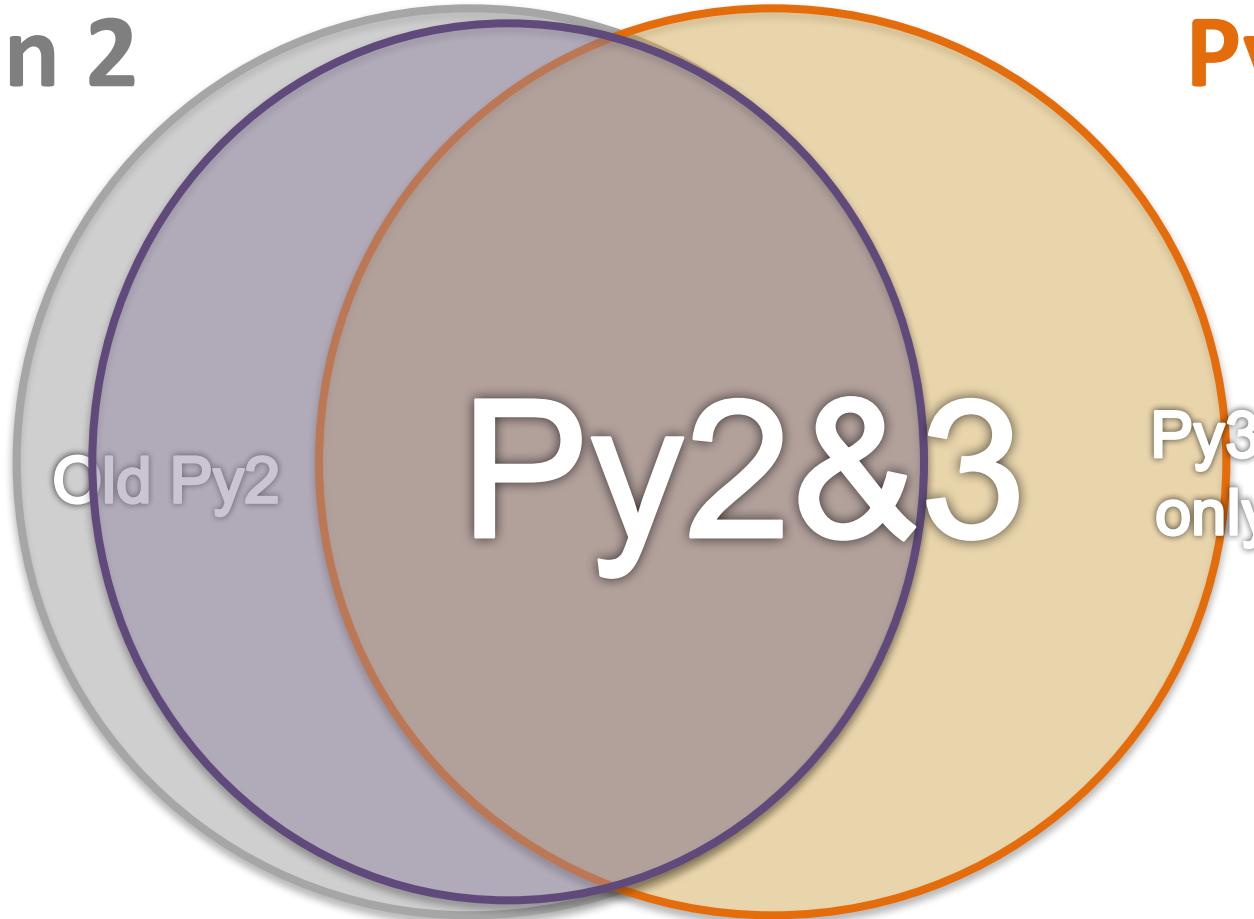
<https://www.youtube.com/watch?v=KOqk8j11aAI>

```
from past.builtins import *
```



Python 2

Python 3



Source: PyCon Australia (2014), Writing Python 2/3 compatible code by Edward Schofield

<https://www.youtube.com/watch?v=KOqk8j11aAI>

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](#)

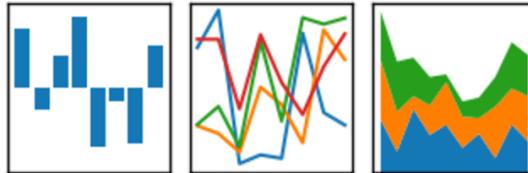
[Old array packages](#)

NumPy
is the
fundamental package
for
scientific computing
with Python.

pandas

pandas

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



[overview](#) // [get pandas](#) // [documentation](#) // [community](#) // [talks](#)

Python Data Analysis Library

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the [Python](#) programming language.

pandas is a [NUMFocus](#) sponsored project. This will help ensure the success of development of *pandas* as a world-class open-source project.

A Fiscally Sponsored Project of



0.19.2 Final (December 24, 2016)

This is a minor bug-fix release in the 0.19.x series and includes some small regression fixes, bug fixes and performance improvements.

Highlights include:

- Compatibility with Python 3.6

<http://pandas.pydata.org/>

VERSIONS

Release

0.19.2 - December 2016
[download](#) // [docs](#) // [pdf](#)

Development

0.20.0 - 2017
[github](#) // [docs](#)

Previous Releases

0.19.1 - [download](#) // [docs](#) // [pdf](#)
0.19.0 - [download](#) // [docs](#) // [pdf](#)
0.18.1 - [download](#) // [docs](#) // [pdf](#)
0.18.0 - [download](#) // [docs](#) // [pdf](#)
0.17.1 - [download](#) // [docs](#) // [pdf](#)
0.17.0 - [download](#) // [docs](#) // [pdf](#)
0.16.2 - [download](#) // [docs](#) // [pdf](#)
0.16.1 - [download](#) // [docs](#) // [pdf](#)
0.16.0 - [download](#) // [docs](#) // [pdf](#)
0.15.2 - [download](#) // [docs](#) // [pdf](#)
0.15.1 - [download](#) // [docs](#) // [pdf](#)
0.15.0 - [download](#) // [docs](#) // [pdf](#)
0.14.1 - [download](#) // [docs](#) // [pdf](#)

pandas

Python Data Analysis

Library

**providing high-performance, easy-to-use
data structures and data analysis tools
for the Python programming language.**

pandas Ecosystem

- Statistics and Machine Learning
 - [Statsmodels](#)
 - [sklearn-pandas](#)
- Visualization
 - [Bokeh](#)
 - [yhat/ggplot](#)
 - [Seaborn](#)
 - [Vincent](#)
 - [IPython Vega](#)
 - [Plotly](#)
 - [Pandas-Qt](#)
- IDE
 - [IPython](#)
 - [quantopian/qgrid](#)
 - [Spyder](#)
- API
 - [pandas-datareader](#)
 - [quandl/Python](#)
 - [pydatastream](#)
 - [pandaSDMX](#)
 - [fredapi](#)
- Domain Specific
 - [Geopandas](#)
 - [xarray](#)
- Out-of-core
 - [Dask](#)
 - [Blaze](#)
 - [Odo](#)

pandas: powerful Python data analysis toolkit

[pandas 0.19.2 documentation »](#)

Table Of Contents

What's New
Installation
Contributing to pandas
Frequently Asked Questions (FAQ)
Package overview
10 Minutes to pandas
Tutorials
Cookbook
Intro to Data Structures
Essential Basic Functionality
Working with Text Data
Options and Settings
Indexing and Selecting Data
MultiIndex / Advanced Indexing
Computational tools
Working with missing data
Group By: split-apply-combine
Merge, join, and concatenate
Reshaping and Pivot Tables
Time Series / Date functionality
Time Deltas
Categorical Data
Visualization
Style
IO Tools (Text, CSV, HDF5, ...)
Remote Data Access
Enhancing Performance
Sparse data structures
Caveats and Gotchas
rpy2 / R interface
pandas Ecosystem
Comparison with R / R libraries
Comparison with SQL
Comparison with SAS
API Reference

pandas: powerful Python data analysis toolkit

[PDF Version](#)

[Zipped HTML](#)

Date: Dec 24, 2016 **Version:** 0.19.2

Binary Installers: <http://pypi.python.org/pypi/pandas>

Source Repository: <http://github.com/pydata/pandas>

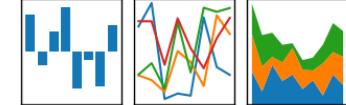
Issues & Ideas: <https://github.com/pydata/pandas/issues>

Q&A Support: <http://stackoverflow.com/questions/tagged/pandas>

Developer Mailing List: <http://groups.google.com/group/pydata>

pandas

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



pandas is a [Python](#) package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, **real world** data analysis in Python. Additionally, it has the broader goal of becoming **the most powerful and flexible open source data analysis / manipulation tool available in any language**. It is already well on its way toward this goal.

pandas is well suited for many different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure

The two primary data structures of pandas, **Series** (1-dimensional) and **DataFrame** (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, **DataFrame** provides everything that R’s `data.frame` provides and much more. pandas is built on top of **NumPy** and is

<http://pandas.pydata.org/pandas-docs/stable/>

pandas: powerful Python data analysis toolkit

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure

Series

DataFrame

- Primary data structures of pandas
 - Series (1-dimensional)
 - DataFrame (2-dimensional)
- Handle the vast majority of typical use cases in **finance**, statistics, social science, and many areas of engineering.

pandas DataFrame

- **DataFrame** provides everything that R's `data.frame` provides and much more.
- pandas is built on top of **NumPy** and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

pandas

Comparison with SAS

pandas	SAS
DataFrame	data set
column	variable
row	observation
groupby	BY-group
NaN	.

Python Pandas Cheat Sheet

Data Wrangling

with pandas
Cheat Sheet

<http://pandas.pydata.org>

Syntax – Creating DataFrames

	a	b	c
1	4	7	10
2	5	8	11
3	6	9	12

```
df = pd.DataFrame(
    {"a" : [4 ,5, 6],
     "b" : [7, 8, 9],
     "c" : [10, 11, 12]},
    index = [1, 2, 3])
Specify values for each column.
```

```
df = pd.DataFrame(
    [[4, 7, 10],
     [5, 8, 11],
     [6, 9, 12]],
    index=[1, 2, 3],
    columns=['a', 'b', 'c'])
Specify values for each row.
```

	a	b	c
n	v		
d	1	4	7
e	2	5	11

```
df = pd.DataFrame(
    {"a" : [4 ,5, 6],
     "b" : [7, 8, 9],
     "c" : [10, 11, 12]},
    index = pd.MultiIndex.from_tuples(
        [('d',1),('d',2),('e',2)],
        names=['n','v']))
Create DataFrame with a MultiIndex
```

Method Chaining

Most pandas methods return a DataFrame so that another pandas method can be applied to the result. This improves readability of code.

```
df = (pd.melt(df)
      .rename(columns={
          'variable' : 'var',
          'value' : 'val'})
      .query('val >= 200')
     )
```

Tidy Data – A foundation for wrangling in pandas

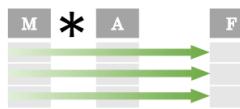
In a tidy data set:

 Each variable is saved in its own column

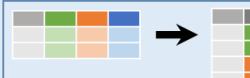
&

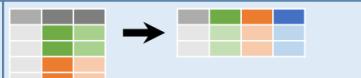
 Each observation is saved in its own row

Tidy data complements pandas's **vectorized operations**. pandas will automatically preserve observations as you manipulate variables. No other format works as intuitively with pandas.


M * A

Reshaping Data – Change the layout of a data set


pd.melt(df)
 Gather columns into rows.


df.pivot(columns='var', values='val')
 Spread rows into columns.


pd.concat([df1,df2])
 Append rows of DataFrames


pd.concat([df1,df2], axis=1)
 Append columns of DataFrames

df=df.sort_values('mpg')
 Order rows by values of a column (low to high).

df=df.sort_values('mpg', ascending=False)
 Order rows by values of a column (high to low).

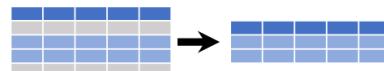
df=df.rename(columns = {'y':'year'})
 Rename the columns of a DataFrame

df=df.sort_index()
 Sort the index of a DataFrame

df=df.reset_index()
 Reset index of DataFrame to row numbers, moving index to columns.

df=df.drop(['Length', 'Height'], axis=1)
 Drop columns from DataFrame

Subset Observations (Rows)



```
df[df.Length > 7]
Extract rows that meet logical criteria.

df.drop_duplicates()
Remove duplicate rows (only considers columns).

df.head(n)
Select first n rows.

df.tail(n)
Select last n rows.
```

```
df.sample(frac=0.5)
Randomly select fraction of rows.

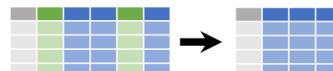
df.sample(n=10)
Randomly select n rows.

df.iloc[10:20]
Select rows by position.

df.nlargest(n, 'value')
Select and order top n entries.

df.nsmallest(n, 'value')
Select and order bottom n entries.
```

Subset Variables (Columns)



```
df[['width', 'length', 'species']]
Select multiple columns with specific names.

df['width'] or df.width
Select single column with specific name.

df.filter(regex='regex')
Select columns whose name matches regular expression regex.
```

regex (Regular Expressions) Examples

'.'	Matches strings containing a period '.'.
'Length\$'	Matches strings ending with word 'Length'
'Sepal'	Matches strings beginning with the word 'Sepal'
'x[1-5]\$'	Matches strings beginning with 'x' and ending with 1,2,3,4,5
'^(?!Species\$).*''	Matches strings except the string 'Species'

Logic in Python (and pandas)

<	Less than	!=	Not equal to
>	Greater than	df.column.isin(values)	Group membership
==	Equals	pd.isnull(obj)	Is NaN
<=	Less than or equals	pd.notnull(obj)	Is not NaN
>=	Greater than or equals	&, , ~, ^, df.any(), df.all()	Logical and, or, not, xor, any, all

```
df.loc[:, 'x2': 'x4']
Select all columns between x2 and x4 (inclusive).

df.iloc[:, 1, 2, 5]
Select columns in positions 1, 2 and 5 (first column is 0).

df.loc[df['a'] > 10, ['a', 'c']]
Select rows meeting logical condition, and only the specific columns.
```



Python

Download Anaconda



[Log In](#) [Get Support](#) [Search](#) [Contact](#)

[PRODUCTS](#) [COMMUNITY](#) [CONSULTING](#) [TRAINING](#) [ABOUT](#) [RESOURCES](#)

DOWNLOAD ANACONDA NOW

Download for



GET SUPERPOWERS WITH ANACONDA

Anaconda is the leading open data science platform powered by Python. The open source version of Anaconda is a high performance distribution of Python and R and includes over 100 of the most popular Python, R and Scala packages for data science.

Which version should I download and install?

With Anaconda you can run multiple versions of Python in isolated environments, so choose the download with the Python version that you use more often, as that will be your default Python version.

<https://www.continuum.io/downloads>

Download Anaconda Python 3.6

Download for Windows

Download for macOS

Download for Linux

Anaconda 4.3.1

For macOS

macOS 10.12.2 users: To prevent permissions problems, we recommend that you upgrade to macOS 10.12.3 or later before installing Anaconda.

Anaconda is BSD licensed which gives you permission to use Anaconda commercially and for redistribution.

Changelog

Graphical Installer

1. Download the graphical installer
2. Double-click the downloaded .pkg file and follow the instructions

Command Line Installer

1. Download the command-line installer
2. Optional: Verify data integrity with [MD5](#) or [SHA-256](#) [More info](#)
3. In your terminal window type one of the below and follow the instructions:
Python 3.6 version

Python 3.6 version

GRAPHICAL INSTALLER (424M)

COMMAND-LINE INSTALLER (363M)

64-Bit

Python 2.7 version

GRAPHICAL INSTALLER (419M)

COMMAND-LINE INSTALLER (358M)

64-Bit

GET ANACONDA SUPPORT

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
```

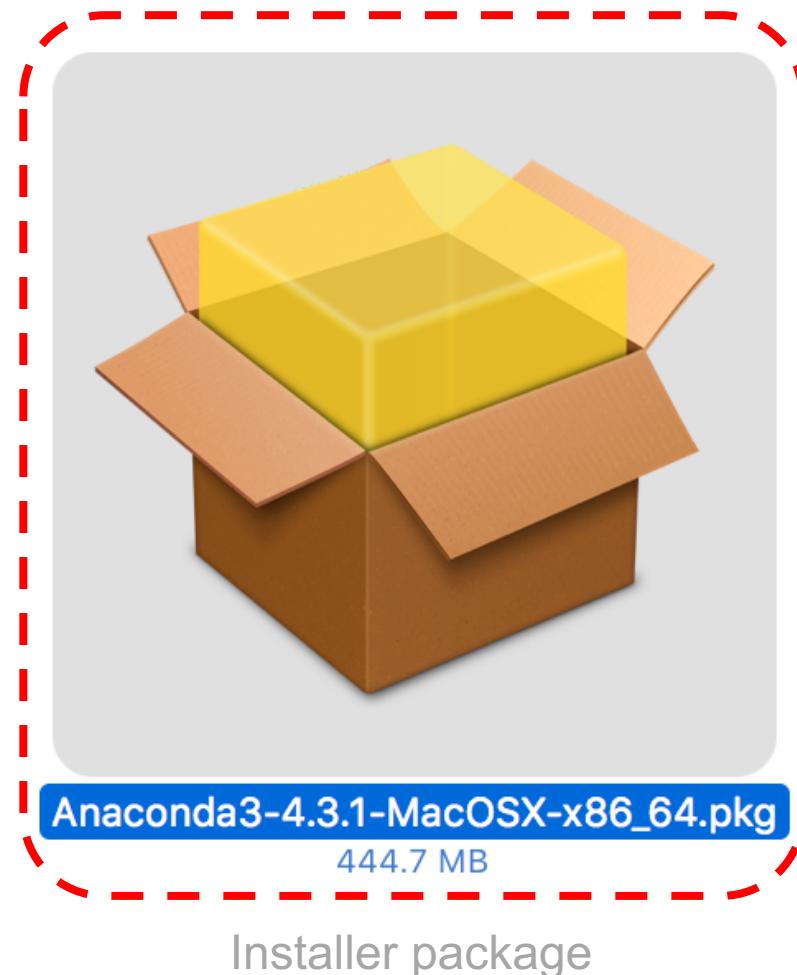
Python 2.7 version

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

OS X Anaconda 3 - 4.3.1

Python 3.6 Installation

Anaconda3-4.3.1-MacOSX-x86_64.pkg



Install Anaconda 3

Install Anaconda3

Welcome to the Anaconda3 Installer

- **Introduction**
- Read Me
- License
- Destination Select
- Installation Type
- Installation
- Summary

You will be guided through the steps necessary to install this software.

 ANACONDA®

Go Back Continue

Install Anaconda 3

Install Anaconda3

Important Information

- Introduction
- Read Me**
- License
- Destination Select
- Installation Type
- Installation
- Summary

 **ANACONDA®**

Anaconda is a modern open source analytics platform powered by Python. See <https://www.continuum.io/downloads/>.

By default, this installer modifies your bash profile to put Anaconda in your PATH. To disable this, choose "Customize" at the "Installation Type" phase, and disable the "Modify PATH" option. If you do not do this, you will need to add `~/anaconda/bin` to your PATH manually to run the commands, or run all anaconda commands explicitly from that path.

To install to a different location, select "Change Install Location..." at the "Installation Type" phase, the choose "Install on a specific disk...", choose the disk you wish to install on, and click "Choose Folder...". The "Install for me only" option will install anaconda to the default location, `~/anaconda`.

The packages included in this installation are:

- alabaster 0.7.9

Print...

Save...

Go Back

Continue

44

Install Anaconda 3

Install Anaconda3

Software License Agreement

=====

Anaconda License

=====

Copyright 2016, Continuum Analytics, Inc.

All rights reserved under the 3-clause BSD License:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of Continuum Analytics, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS

Print... Save... Go Back Continue



ANACONDA

Install Anaconda 3

Install Anaconda3

Software License Agreement

=====

Anaconda License

=====

Copyright 2016, Continuum Analytics, Inc.

To continue installing the software you must agree to the terms of the software license agreement.

Click Agree to continue or click Disagree to cancel the installation and quit the Installer.

Read License Disagree Agree

* Neither the name of Continuum Analytics, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS

Print... Save... Go Back Continue



ANACONDA

Install Anaconda 3

Install Anaconda3

Standard Install on "Macintosh HD"

- Introduction
- Read Me
- License
- Destination Select
- Installation Type**
- Installation
- Summary

This will take 1.4 GB of space on your computer.

Click Install to perform a standard installation of this software in your home folder. Only the current user of this computer will be able to use this software.

Change Install Location...

Customize Go Back Install



ANACONDA®

Install Anaconda 3

Install Anaconda3

Select a Destination

- Introduction
- Read Me
- License
- **Destination Select**
- Installation Type
- Installation
- Summary

How do you want to install this software?

 Install for all users of this computer

 **Install for me only**

 Install on a specific disk...

Installing this software requires 1.4 GB of space.
You have chosen to install this software in your home folder.
Only the current user will be able to use this software.

Go Back Continue



Install Anaconda 3

Install Anaconda3

Standard Install on "Macintosh HD"

- Introduction
- Read Me
- License
- Destination Select
- Installation Type**
- Installation
- Summary

This will take 1.4 GB of space on your computer.

Click Install to perform a standard installation of this software in your home folder. Only the current user of this computer will be able to use this software.

Change Install Location...

Customize Go Back **Install**



ANACONDA®

Install Anaconda 3

Install Anaconda3

Installing Anaconda3

- Introduction
- Read Me
- License
- Destination Select
- Installation Type
- **Installation**
- Summary

Registering updated applications...

Install time remaining: About a minute

Go Back Continue



ANACONDA®

Install Anaconda 3

The installation was completed successfully.

Anaconda is the leading open data science platform powered by Python.

Share your notebooks and packages on Anaconda Cloud!
[Sign up for free](#)

178 python packages included.

**Supported packages:
453**



ANACONDA®

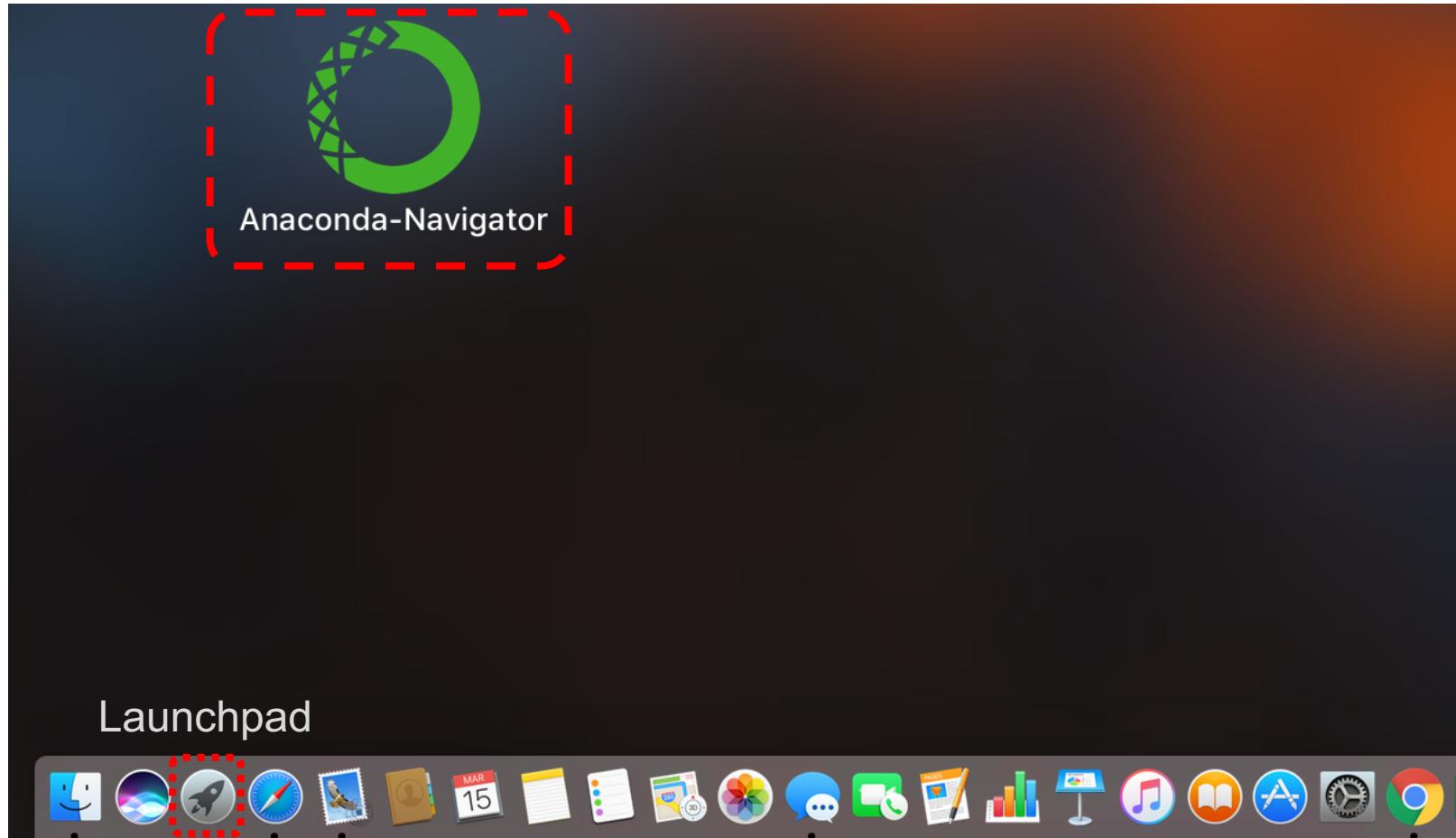
Go Back Close

Install Anaconda 3

1	_license	1.1	51	heapdict	1.0.0	101	partd	0.3.7	151	sip	4.18	py36_0
2	alabaster	0.7.9	52	icu	54.1	102	path.py	10.0	152	six	1.10.0	py36_0
3	anaconda	4.3.1	53	idna	2.2	103	pathlib2	2.2.0	153	snowballstemmer	1.2.1	py36_0
4	anaconda-client	1.6.0	54	imagesize	0.7.1	104	patsy	0.4.1	154	sockjs-tornado	1.0.3	py36_0
5	anaconda-navigator	1.5.0	55	ipykernel	4.5.2	105	pep8	1.7.0	155	sphinx	1.5.1	py36_0
6	anaconda-project	0.4.1	56	ipython	5.1.0	106	pexpect	4.2.1	156	spyder	3.1.2	py36_0
7	appnope	0.1.0	57	ipython_genutils	0.1.0	107	pickleshare	0.7.4	157	sqlalchemy	1.1.5	py36_0
8	appscript	1.0.1	58	ipywidgets	5.2.2	108	pillow	4.0.0	158	sqlite	3.13.0	0
9	astroid	1.4.9	59	isort	4.2.5	109	pip	9.0.1	159	statsmodels	0.6.1	np111py36_1
10	astropy	1.3	60	itsdangerous	0.24	110	ply	3.9	160	sympy	1.0	py36_0
11	babel	2.3.4	61	jbig	2.1	111	prompt_toolkit	1.0.9	161	terminado	0.6	py36_0
12	backports	1.0	62	jdcal	1.3	112	psutil	5.0.1	162	tk	8.5.18	0
13	beautifulsoup4	4.5.3	63	jedi	0.9.0	113	ptyprocess	0.5.1	163	toolz	0.8.2	py36_0
14	bitarray	0.8.1	64	jinja2	2.9.4	114	py	1.4.32	164	tornado	4.4.2	py36_0
15	blaze	0.10.1	65	jpeg	9b	115	pyasn1	0.1.9	165	traitlets	4.3.1	py36_0
16	bokeh	0.12.4	66	jsonschema	2.5.1	116	pycosat	0.6.1	166	unicodecsv	0.14.1	py36_0
17	boto	2.45.0	67	jupyter	1.0.0	117	pycparser	2.17	167	wcwidth	0.1.7	py36_0
18	bottleneck	1.2.0	68	jupyter_client	4.4.0	118	pycrypto	2.6.1	168	werkzeug	0.11.15	py36_0
19	cffi	1.9.1	69	jupyter_console	5.0.0	119	pycurl	7.43.0	169	wheel	0.29.0	py36_0
20	chardet	2.3.0	70	jupyter_core	4.2.1	120	pyflakes	1.5.0	170	widgetsnbextension	1.2.6	py36_0
21	chest	0.2.3	71	lazy-object-proxy	1.2.2	121	pygments	2.1.3	171	wrapt	1.10.8	py36_0
22	click	6.7	72	libiconv	1.14	122	pylint	1.6.4	172	xlrd	1.0.0	py36_0
23	cloudpickle	0.2.2	73	libpng	1.6.27	123	pyopenssl	16.2.0	173	xlsxwriter	0.9.6	py36_0
24	clyent	1.2.2	74	libtiff	4.0.6	124	pyparsing	2.1.4	174	xlwings	0.10.2	py36_0
25	colorama	0.3.7	75	libxml2	2.9.4	125	pyqt	5.6.0	175	xlwt	1.2.0	py36_0
26	conda	4.3.14	76	libxslt	1.1.29	126	pytables	3.3.0	176	xz	5.2.2	1
27	conda-env	2.6.0	77	llvmlite	0.15.0	127	pytest	3.0.5	177	yaml	0.1.6	0
28	configobj	5.0.6	78	locket	0.2.0	128	python	3.6.0	178	zlib	1.2.8	3
29	contextlib2	0.5.4	79	lxml	3.7.2	129	python-dateutil	2.6.0		py36_0		
30	cryptography	1.7.1	80	markupsafe	0.23	130	python.app	1.2		py36_4		
31	curl	7.52.1	81	matplotlib	2.0.0	131	pytz	2016.10		py36_0		
32	cycler	0.10.0	82	mistune	0.7.3	132	pyyaml	3.12		py36_0		
33	cython	0.25.2	83	mkl	2017.0.1	133	pyzmq	16.0.2		py36_0		
34	cytoolz	0.8.2	84	mkl-service	1.1.2	134	qt	5.6.2		0		
35	dask	0.13.0	85	mpmath	0.19	135	qtawesome	0.4.3		py36_0		
36	datashape	0.5.4	86	multipledispatch	0.4.9	136	qtconsole	4.2.1		py36_1		
37	decorator	4.0.11	87	nbconvert	4.2.0	137	qtpy	1.2.1		py36_0		
38	dill	0.2.5	88	nbformat	4.2.0	138	readline	6.2		2		
39	docutils	0.13.1	89	networkx	1.11	139	redis	3.2.0		0		
40	entrypoints	0.2.2	90	nltk	3.2.2	140	redis-py	2.10.5		py36_0		
41	et_xmlfile	1.0.1	91	nose	1.3.7	141	requests	2.12.4		py36_0		
42	fastcache	1.0.2	92	notebook	4.3.1	142	rope	0.9.4		py36_1		
43	flask	0.12	93	numba	0.30.1	143	ruamel_yaml	0.11.14		py36_1		
44	flask-cors	3.0.2	94	numexpr	2.6.1	144	scikit-image	0.12.3		np111py36_1		
45	freetype	2.5.5	95	numpy	1.11.3	145	scikit-learn	0.18.1		np111py36_1		
46	get_terminal_size	1.0.0	96	numpydoc	0.6.0	146	scipy	0.18.1		np111py36_1		
47	gevent	1.2.1	97	odo	0.5.0	147	seaborn	0.7.1		py36_0		
48	greenlet	0.4.11	98	openpyxl	2.4.1	148	setuptools	27.2.0		py36_0		
49	h5py	2.6.0	99	openssl	1.0.2k	149	simplegeneric	0.8.1		py36_1		
50	hdf5	1.8.17	100	pandas	0.19.2	150	singledispatch	3.4.0.3		py36_0		

178
python
package
S
included.⁵²

Anaconda-Navigator



Anaconda-Navigator

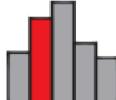
The screenshot shows the Anaconda Navigator application window. At the top, there's a header bar with the Anaconda Navigator logo, a search bar labeled "Applications on root", and buttons for "Upgrade Now" and "Sign in to Anaconda Cloud". On the left, a sidebar menu includes "Home", "Environments", "Projects (beta)", "Learning", "Community", "Documentation", "Developer Blog", and "Feedback", along with social media links for Twitter, YouTube, and GitHub.

The main area displays a grid of application icons. A central modal dialog box is open, titled "ANACONDA NAVIGATOR", with the message "Thanks for installing Anaconda! Anaconda Navigator helps you easily start important Python applications and manage the packages in your local Anaconda installation. It also connects you to online resources for learning and engaging with the Python, SciPy, and PyData community. To help us improve Anaconda Navigator, fix bugs, and make it even easier for everyone to use Python, we gather anonymized usage information, just like most web browsers and mobile apps. To opt out of this, please uncheck below (You can always change this setting in the Preferences menu)." It contains a checked checkbox "Yes, I'd like to help improve Anaconda." and two buttons: "Ok" and "Ok, and don't show again".

The application grid includes:

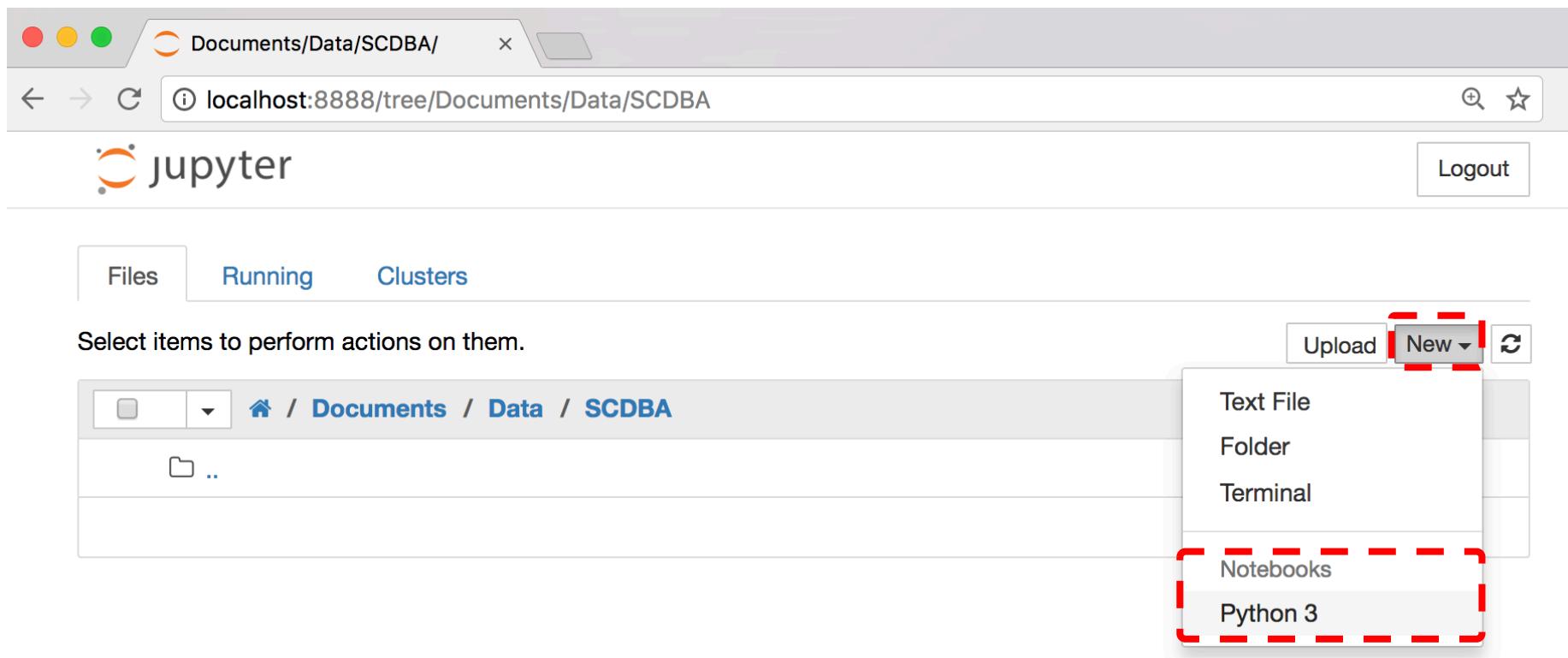
- jupyter notebook**: Web-based, interactive computing environment. Edit and run human docs while describing the data. Version 4.3.1. Buttons: Launch, Install.
- anaconda-fusion**: Integration between Excel® and Anaconda via Notebooks. Run data science functions, interact with results and create advanced visualizations in a code-free app inside Excel. Version 1.0.2. Buttons: Install.
- spyder**: Python Development Environment. Powerful Python IDE with editing, interactive testing, and introspection features. Version 3.1.2. Buttons: Launch, Install.
- glueviz**: Multidimensional data visualization across files. Explore relationships within and among related datasets. Version 0.9.1. Buttons: Install.
- rstudio**: A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks. Version 1.0.136. Buttons: Install.

Jupyter Notebook

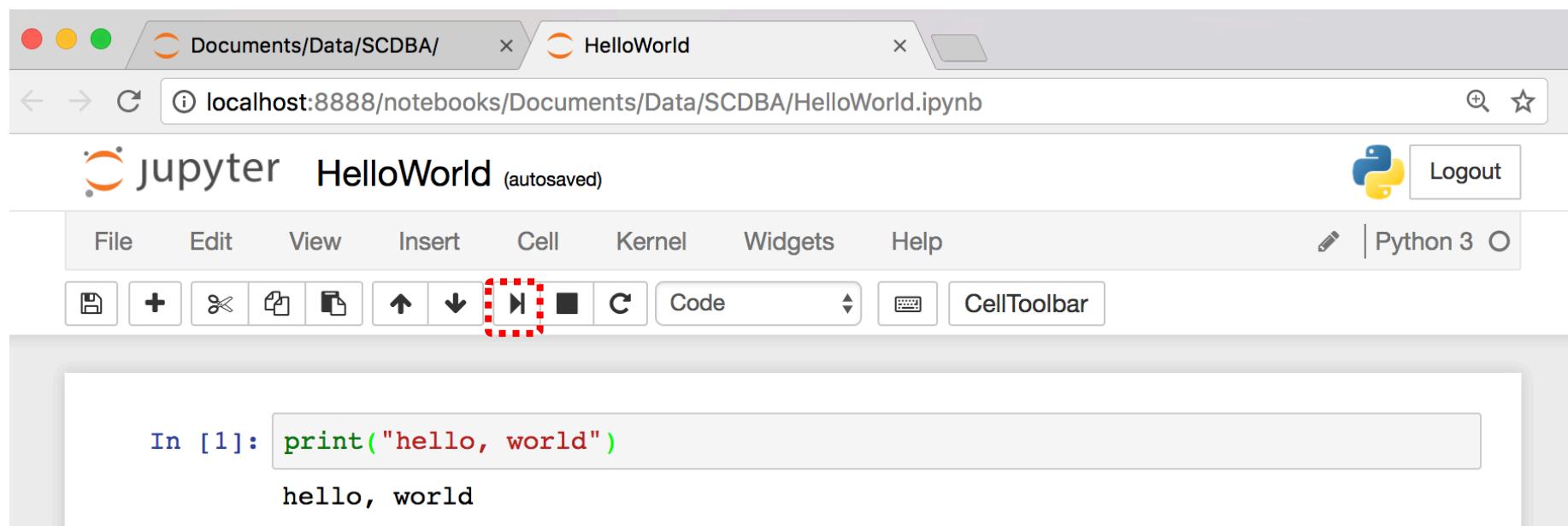
 <p>jupyter notebook  4.3.1</p> <p>Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.</p> <p>Launch</p>	 <p>qtconsole  4.2.1</p> <p>PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.</p> <p>Launch</p>	 <p>spyder  3.1.2</p> <p>Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features</p> <p>Launch</p>
 <p>anaconda-fusion  1.0.2</p> <p>Integration between Excel ® and Anaconda via Notebooks. Run data science functions, interact with results and create advanced visualizations in a code-free app inside Excel</p> <p>Install</p>	 <p>glueviz  0.9.1</p> <p>Multidimensional data visualization across files. Explore relationships within and among related datasets.</p> <p>Install</p>	 <p>rstudio  1.0.136</p> <p>A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.</p> <p>Install</p>

Jupyter Notebook

New Python 3



print("hello, world")



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

The screenshot shows a Jupyter Notebook interface running in a web browser. The title bar indicates the notebook is titled "HelloWorld". The toolbar includes standard file operations like Open, Save, and New, along with a Cell toolbar containing options like "Code" and "CellToolbar".

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

In [2]: `from platform import python_version
print("Python Version:", python_version())`
Python Version: 3.6.0

Create Python Environments with Anaconda

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

Anaconda Create New Python 3.5 Environment (py35)

ANACONDA NAVIGATOR

Sign in to Anaconda Cloud

Home Environments Projects (beta) Learning Community Documentation Developer Blog Feedback

Create Clone Import Remove

Search Environments Installed Channels Update index... Search Packages

root

Create new environment

Environment name: py35

Python version: 3.5

Cancel Create

186 packages available (root)

py35 Python 3.5

The screenshot shows the Anaconda Navigator interface. On the left, there's a sidebar with links for Home, Environments (which is selected and highlighted with a red dashed border), Projects (beta), Learning, Community, Documentation, Developer Blog, and Feedback. At the bottom of the sidebar are social media icons for Twitter, YouTube, and GitHub. The main area has tabs for Search Environments, Installed (selected), Channels, and Update index... There's also a Search Packages bar. A tree view on the left shows a single node 'root'. In the center, a modal dialog titled 'Create new environment' is open, showing fields for 'Environment name' (set to 'py35'), 'Python' (checked), 'R' (unchecked), and 'Python version' (set to '3.5'). Below the modal is a list of 186 packages available under the 'root' environment, including '_license', 'jupyter', 'astroid', 'astropy', 'babel', 'backports', 'backports.shutil-get-terminal-size', and 'beautifulsoup4'. At the bottom of the main area, there are buttons for Create, Clone, Import, and Remove.

Anaconda Create New Python 2.7 Environment (py27)

The screenshot shows the Anaconda Navigator interface. On the left, there's a sidebar with icons for Home, Environments (highlighted with a red dashed box), Projects (beta), Learning, Community, Documentation, Developer Blog, and Feedback. Below the sidebar are social media links for Twitter, YouTube, and GitHub.

In the main area, there's a search bar for environments and a list of environments: root and py35. A large red box highlights the 'py35' environment. To its right, a table lists installed packages: openssl, pip, python, readline, setuptools, sqlite, tk, wheel, xz, and zlib. The 'python' package is highlighted with a blue arrow icon.

On the far right, a modal window titled 'Create new environment' is open. It has fields for 'Environment name' (set to 'py27'), 'Python' (checked), 'R' (unchecked), and 'Python version' (set to '2.7'). There are 'Cancel' and 'Create' buttons at the bottom.

At the bottom of the main pane, it says '10 packages available (/Users/imyday/anaconda/envs/py35)'.

Red annotations are present: 'py35 Python 3.5' is written over the 'py35' environment entry, and 'py27 Python 2.7' is written over the 'Create new environment' modal.

Verify that conda is installed, check current conda version

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

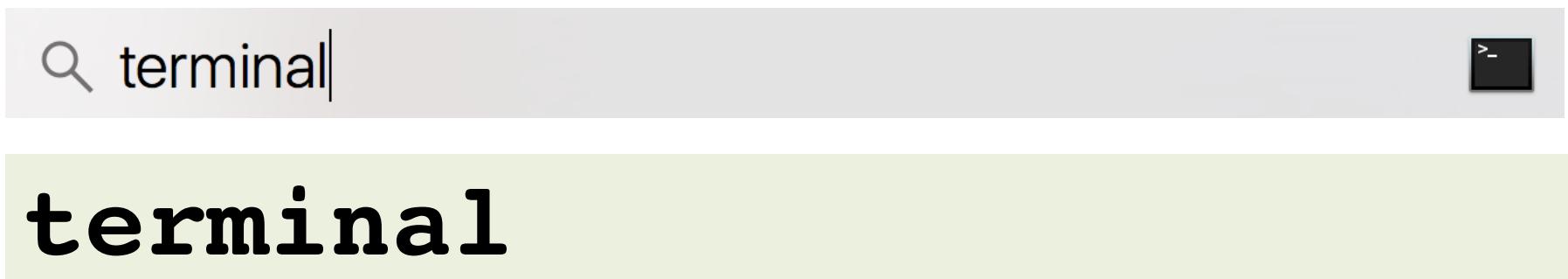
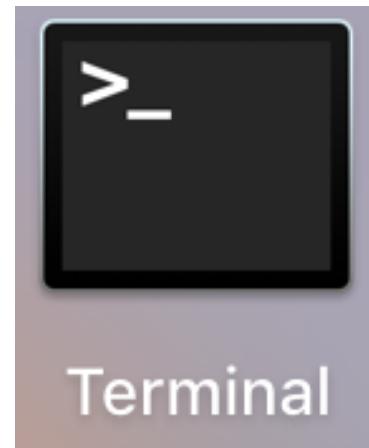
Check current conda version

Check current python version

Check conda environments

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

Terminal



conda list

```
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
```

python --version



A screenshot of a macOS terminal window titled "imyday — -bash — 80x24". The window shows the command "python --version" being run, and the output "Python 3.6.0 :: Anaconda 4.3.1 (x86_64)" is displayed. The command and its output are highlighted with red dashed boxes.

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

conda --version

```
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
```

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

```
[iMyday-MacBook-Pro:~ imyday$ source activate py35  
[(py35) iMyday-MacBook-Pro:~ imyday$ python --version  
Python 3.5.3 :: Continuum Analytics, Inc.
```

source activate py35

```
[(py35) iMyday-MacBook-Pro:~ imyday$ conda --version  
conda 4.3.14
```

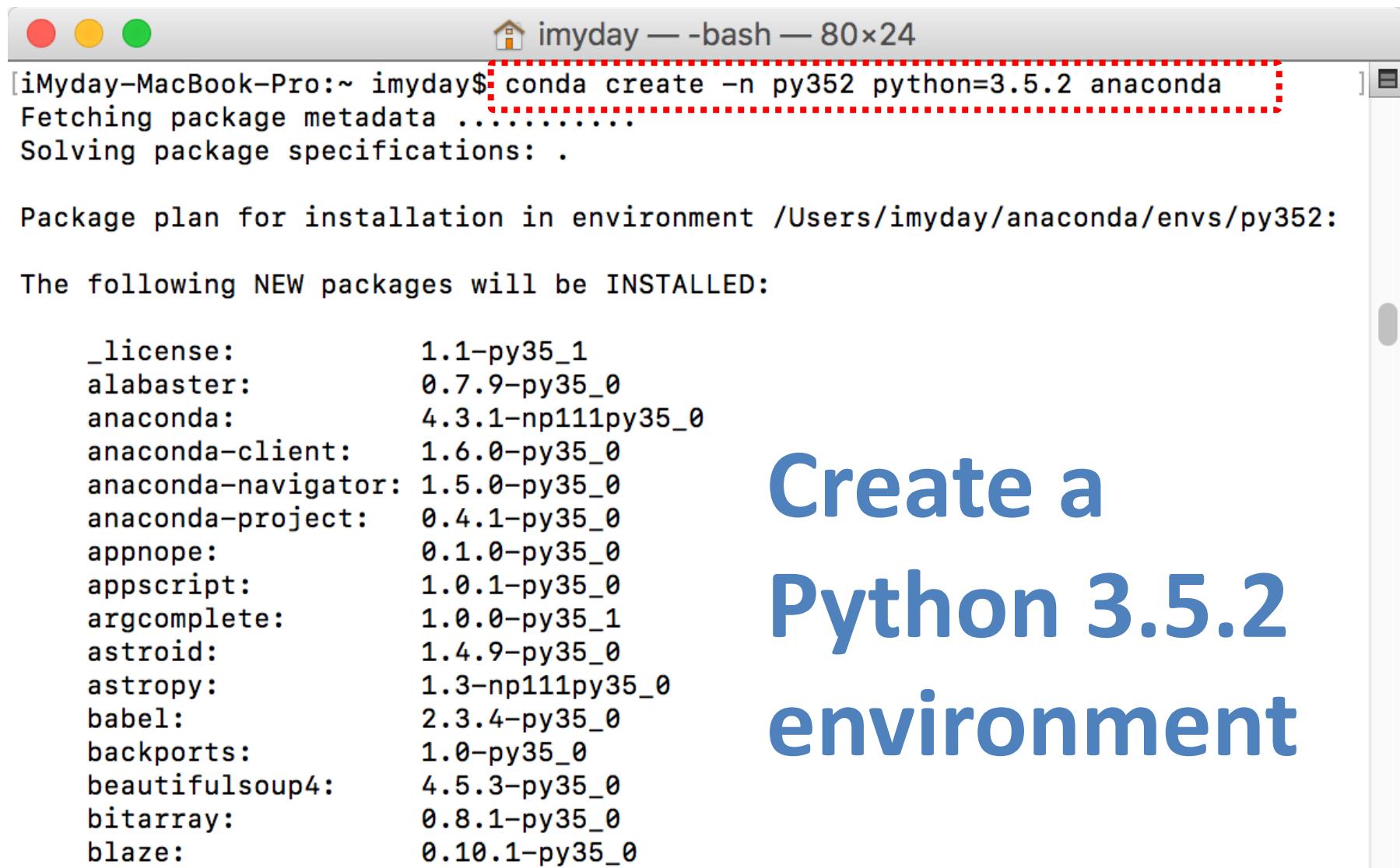
```
[(py35) iMyday-MacBook-Pro:~ imyday$ source deactivate py35
```

```
[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
```

source deactivate py35

```
conda create -n py352 python=3.5.2 anaconda
```



A screenshot of a macOS terminal window titled "imyday — -bash — 80x24". The window shows the command "conda create -n py352 python=3.5.2 anaconda" being run. The output includes package metadata fetching, solving specifications, and a package plan for installation in the environment "/Users/imyday/anaconda/envs/py352". The terminal also lists the new packages that will be installed, including _license, alabaster, anaconda, anaconda-client, anaconda-navigator, anaconda-project, appnope, appscript, argcomplete, astroid, astropy, babel, backports, beautifulsoup4, bitarray, and blaze.

```
[iMyday-MacBook-Pro:~ imyday$ conda create -n py352 python=3.5.2 anaconda
Fetching package metadata .....
Solving package specifications: .

Package plan for installation in environment /Users/imyday/anaconda/envs/py352:

The following NEW packages will be INSTALLED:

_license:          1.1-py35_1
alabaster:         0.7.9-py35_0
anaconda:          4.3.1-np111py35_0
anaconda-client:   1.6.0-py35_0
anaconda-navigator: 1.5.0-py35_0
anaconda-project:  0.4.1-py35_0
appnope:           0.1.0-py35_0
appscript:          1.0.1-py35_0
argcomplete:        1.0.0-py35_1
astroid:            1.4.9-py35_0
astropy:            1.3-np111py35_0
babel:              2.3.4-py35_0
backports:          1.0-py35_0
beautifulsoup4:     4.5.3-py35_0
bitarray:           0.8.1-py35_0
blaze:              0.10.1-py35_0
```

Create a Python 3.5.2 environment

```
conda create -n py352 python=3.5.2 anaconda
```

```
i myday — -bash — 80x24
pyopenssl-16.2 100% | #####| Time: 0:00:00 1.40 MB/s
scikit-image-0 100% | #####| Time: 0:00:17 1.05 MB/s
seaborn-0.7.1- 100% | #####| Time: 0:00:00 1.05 MB/s
statsmodels-0. 100% | #####| Time: 0:00:04 1.06 MB/s
anaconda-navig 100% | #####| Time: 0:00:04 1.05 MB/s
blaze-0.10.1-p 100% | #####| Time: 0:00:00 1.05 MB/s
ipykernel-4.5. 100% | #####| Time: 0:00:00 1.21 MB/s
nbconvert-4.2. 100% | #####| Time: 0:00:00 1.22 MB/s
jupyter_consol 100% | #####| Time: 0:00:00 2.74 MB/s
notebook-4.3.1 100% | #####| Time: 0:00:05 1.05 MB/s
qtconsole-4.2. 100% | #####| Time: 0:00:00 1.03 MB/s
spyder-3.1.2-p 100% | #####| Time: 0:00:03 1.06 MB/s
widgetsnbexten 100% | #####| Time: 0:00:01 1.05 MB/s
ipywidgets-5.2 100% | #####| Time: 0:00:00 1.08 MB/s
jupyter-1.0.0- 100% | #####| Time: 0:00:00 2.53 MB/s
anaconda-4.3.1 100% | #####| Time: 0:00:00 4.49 MB/s
#
# To activate this environment, use:
# > source activate py352
#
# To deactivate this environment, use:
# > source deactivate py352
#
```

```
source activate py352
```

conda info --envs

```
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$ python --version
Python 3.6.0 :: Anaconda 4.3.1 (x86_64)
[iMyday-MacBook-Pro:~ imyday$ source activate py352
(py352) 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

(py352) iMyday-MacBook-Pro:~ imyday$ python --version
Python 3.5.2 :: Anaconda 4.3.1 (x86_64)
(py352) iMyday-MacBook-Pro:~ imyday$ 
```

```
conda info --envs
```

```
source activate py27
```

```
python --version
```

```
conda install notebook ipykernel
```

```
jupyter notebook
```

source activate py27

conda install notebook ipykernel

```
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: .

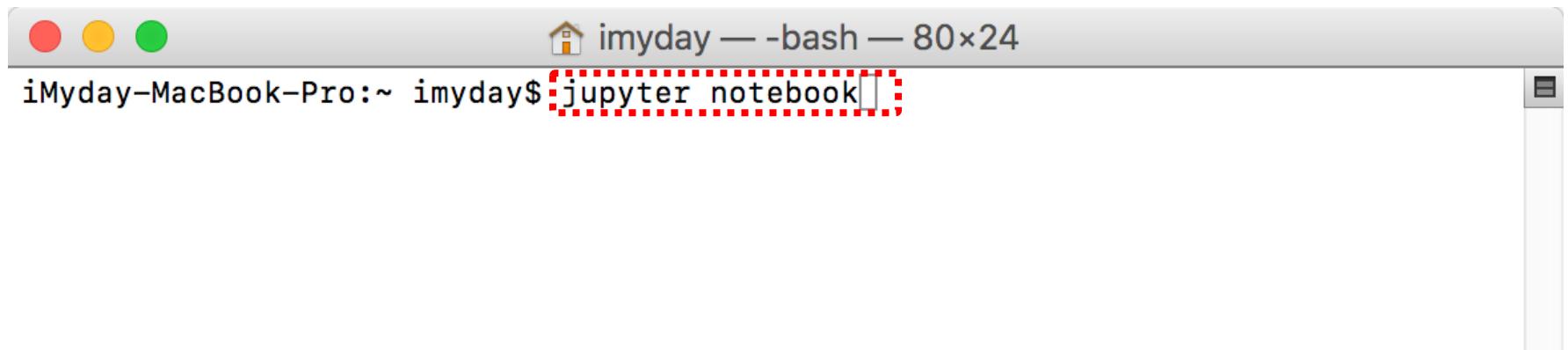
Package plan for installation in environment /Users/imyday/anaconda/envs/py27:

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



A screenshot of a macOS terminal window. The title bar shows the path "imyday — -bash — 80x24". The command "jupyter notebook" is being typed into the terminal, with the last part "notebook" highlighted by a red dashed rectangle. The window has the standard OS X look with red, yellow, and green close buttons.

jupyter notebook
ipython notebook

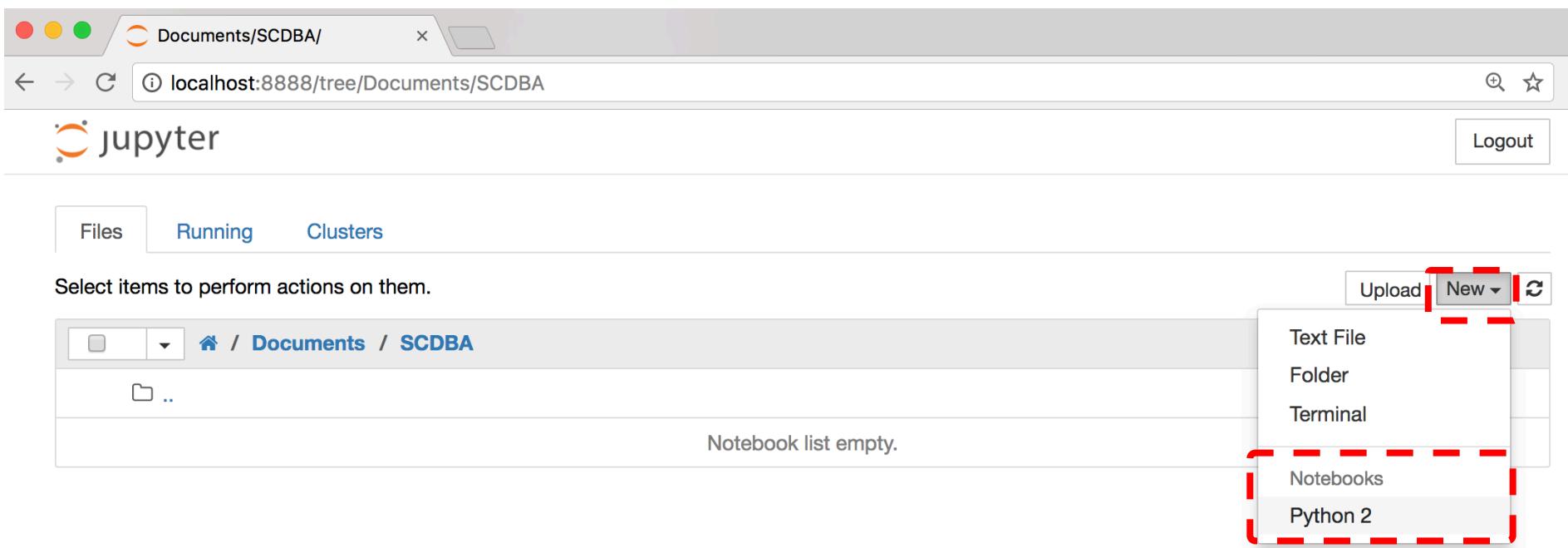
jupyter notebook

```
(py27) iMyday-MacBook-Pro:~ imyday$ jupyter notebook
[W 07:27:29.771 NotebookApp] Widgets are unavailable. Please install widgetsnbextension or ipywidgets 4.0
[I 07:27:29.808 NotebookApp] Serving notebooks from local directory: /Users/imyday
[I 07:27:29.808 NotebookApp] 0 active kernels
[I 07:27:29.808 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888/?token=9cab2ca4b397ce9c4d48a4ef063ff235ffc7a1fc3a9d3ed6
[I 07:27:29.808 NotebookApp] Use Control-C to stop this server and shut down all
kernels (twice to skip confirmation).
[C 07:27:29.810 NotebookApp]

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
http://localhost:8888/?token=9cab2ca4b397ce9c4d48a4ef063ff235ffc7a1fc3a9d3ed6
[I 07:27:30.162 NotebookApp] Accepting one-time-token-authenticated connection from ::1
[W 07:42:52.367 NotebookApp] 404 GET /nbextensions/widgets/notebook/js/extension.js?v=20170315072729 (::1) 13.01ms referer=http://localhost:8888/notebooks/Documents/Data/SCDBA/HelloWorld.ipynb
[I 07:42:52.543 NotebookApp] Kernel started: 390583e8-e01f-448f-8e26-ecd96a630728
```

Jupyter Notebook

New Python 2



print "hello, world"

A screenshot of a Jupyter Notebook interface. At the top, there are three tabs: 'Documents/Data/SCDBA/' and 'HelloPython2'. Below the tabs is a browser-style address bar showing 'localhost:8888/notebooks/Documents/Data/SCDBA/HelloPython2.ipynb'. The main window title is 'jupyter HelloPython2 (unsaved changes)'. On the right side, there is a Python logo icon and a 'Logout' button. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', and 'Help'. The toolbar below the menu has icons for file operations like save, new, cut, copy, paste, and cell controls, followed by 'Code' and 'CellToolbar' buttons. In the bottom-left cell area, the code 'In [1]: print "hello, world"' is written in green, and its output 'hello, world' is displayed in black text.

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

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

jupyter HelloPython2 (unsaved changes)



Logout

File Edit View Insert Cell Kernel Help

Python 2



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

hello, world

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

Python Version: 2.7.13

Jupyter Notebook New Python 3

A screenshot of a Jupyter Notebook interface. At the top, there's a header bar with a back/forward button, a search icon, a star icon, and a three-dot menu icon. Below the header is a navigation bar with icons for 'jupyter' and 'Logout'. The main area has tabs for 'Files', 'Running', and 'Clusters', with 'Files' currently selected. A message says 'Select items to perform actions on them.' Below this is a file tree showing a directory structure: 'Documents / SCDBA / Pandas'. To the right of the file tree is a context menu with options: 'Upload', 'New' (which is highlighted with a red dashed box), 'Text File', 'Folder', 'Terminal', 'Notebooks' (which is also highlighted with a red dashed box), and 'Python 3'. The 'Notebooks' and 'Python 3' options are grouped together by a red dashed box.



Python Fiddle

Python Cloud IDE | Python Fiddle ×

← → ⌂ ⓘ pythonfiddle.com ⋮

Run Reset Share Import Login Language▼

G+1 2.6k

Python Fiddle Python Cloud IDE

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

```
1 print("Hello Python Fiddle")
2
```

Title:

Description:

Tags: A comma-separated list of tags.

Save

Hello Python Fiddle

Text input and output

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

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

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

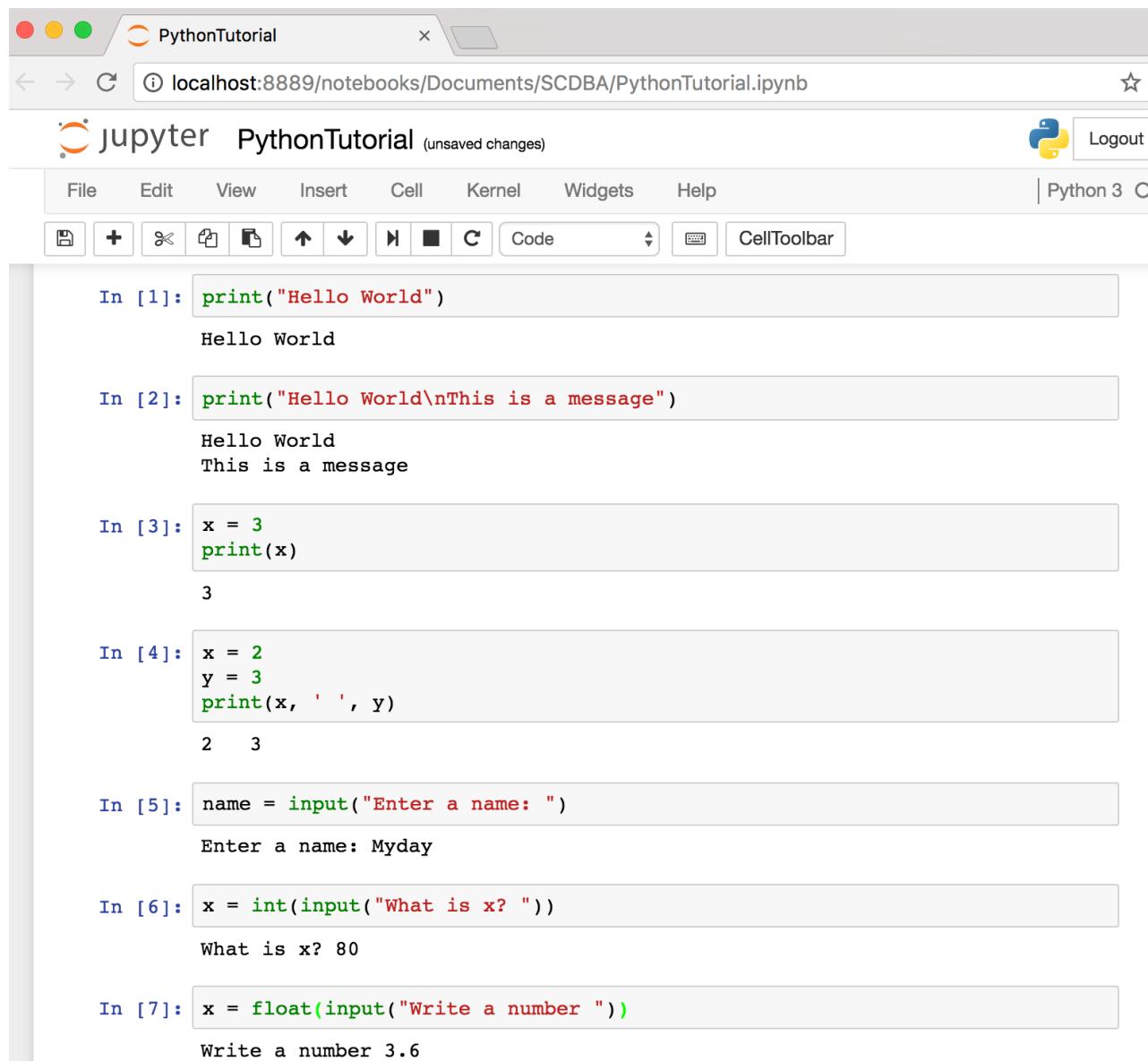
```
x = 2  
y = 3  
print(x, ' ', y)
```

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

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

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

Text input and output



The screenshot shows a Jupyter Notebook interface with the title "PythonTutorial" and the URL "localhost:8889/notebooks/Documents/SCDBA/PythonTutorial.ipynb". The notebook contains the following code and output:

```
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
```

Variables

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

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

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

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)))
```

BMI Calculator in Python

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

File Edit View Insert Cell Kernel Widgets Help

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 []:

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

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)
```

194.871000000012

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

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")
```

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>

for loops

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

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

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
```

while loops

```
age = 10  
  
while age < 20:  
    print(age)  
    age = age + 1
```

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

Functions

```
def convertCMTOM( xcm ) :  
    m = xcm/100  
    return m  
  
cm = 180  
m = convertCMTOM( cm )  
print( str(m) )
```

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

Dictionary

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

Dictionary

'EN' → 'English'

'FR' → 'French'

English

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))
```

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'
```

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)
```

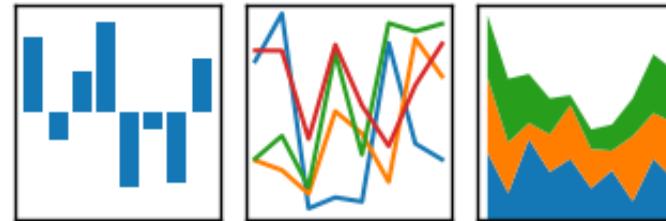
```
with open('myfile.txt', 'a+') as file:  
    file.write('\n' + 'New line')
```

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

Hello World
This is Python File Input Output
New line

pandas

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

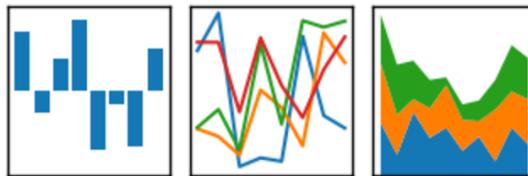


Python Pandas for Finance

pandas

pandas

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



[overview](#) // [get pandas](#) // [documentation](#) // [community](#) // [talks](#)

Python Data Analysis Library

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the [Python](#) programming language.

pandas is a [NUMFocus](#) sponsored project. This will help ensure the success of development of *pandas* as a world-class open-source project.

A Fiscally Sponsored Project of



0.19.2 Final (December 24, 2016)

This is a minor bug-fix release in the 0.19.x series and includes some small regression fixes, bug fixes and performance improvements.

Highlights include:

- Compatibility with Python 3.6

<http://pandas.pydata.org/>

VERSIONS

Release

0.19.2 - December 2016
[download](#) // [docs](#) // [pdf](#)

Development

0.20.0 - 2017
[github](#) // [docs](#)

Previous Releases

0.19.1 - [download](#) // [docs](#) // [pdf](#)
0.19.0 - [download](#) // [docs](#) // [pdf](#)
0.18.1 - [download](#) // [docs](#) // [pdf](#)
0.18.0 - [download](#) // [docs](#) // [pdf](#)
0.17.1 - [download](#) // [docs](#) // [pdf](#)
0.17.0 - [download](#) // [docs](#) // [pdf](#)
0.16.2 - [download](#) // [docs](#) // [pdf](#)
0.16.1 - [download](#) // [docs](#) // [pdf](#)
0.16.0 - [download](#) // [docs](#) // [pdf](#)
0.15.2 - [download](#) // [docs](#) // [pdf](#)
0.15.1 - [download](#) // [docs](#) // [pdf](#)
0.15.0 - [download](#) // [docs](#) // [pdf](#)
0.14.1 - [download](#) // [docs](#) // [pdf](#)

pandas

Python Data Analysis

Library

**providing high-performance, easy-to-use
data structures and data analysis tools
for the Python programming language.**

Jupyter Notebook New Python 3

The screenshot shows the Jupyter Notebook interface at the URL `localhost:8888/tree/Documents/SCDBA/Pandas`. The top navigation bar includes back, forward, and search icons. The logo on the left says "jupyter". The main menu has tabs for "Files", "Running", and "Clusters", with "Files" currently selected. A sidebar on the left shows the file structure: a folder icon, a home icon, and the path `Documents / SCDBA / Pandas`, followed by a ".." button. The main content area displays the message "Notebook list empty." A context menu is open at the top right, with a red dashed box highlighting the "New" dropdown. The "New" dropdown menu lists "Text File", "Folder", "Terminal", "Notebooks", and "Python 3", with "Python 3" also highlighted by a red dashed box.

localhost:8888/tree/Documents/SCDBA/Pandas

jupyter

Logout

Files Running Clusters

Select items to perform actions on them.

Upload New ▾

Text File

Folder

Terminal

Notebooks

Python 3

Creating pd.DataFrame

	a	b	c
1	4	7	10
2	5	8	11
3	6	9	12

In [1]:

```
import numpy as np
import pandas as pd
df = pd.DataFrame({"a": [4, 5, 6],
                    "b": [7, 8, 9],
                    "c": [10, 11, 12]}, 
                    index = [1, 2, 3])
```

Out[1]:

	a	b	c
1	4	7	10
2	5	8	11
3	6	9	12

```
df = pd.DataFrame({"a": [4, 5, 6],
                    "b": [7, 8, 9],
                    "c": [10, 11, 12]},
                    index = [1, 2, 3])
```

Pandas DataFrame

```
type(df)
```

```
type(df)
```

```
pandas.core.frame.DataFrame
```

conda install pandas-datareader

```
iMyday—bash—80x24
[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% |#####
pandas-datarea 100% |#####
[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
```

Jupyter Notebook New Python 3

The screenshot shows the Jupyter Notebook interface running in a web browser at `localhost:8888/tree/Documents/SCDBA/Pandas`. The top navigation bar includes links for Files, Running, and Clusters, along with Logout and other standard browser controls.

The main area displays a file tree under the path `Documents / SCDBA / Pandas`, showing a folder icon and a .. link. A message "Notebook list empty." is centered below the tree.

A context menu is open on the right side, triggered by a red dashed box around the "New" button. The menu options are:

- Upload
- New ▾ (highlighted with a red dashed box)
- Text File
- Folder
- Terminal
- Notebooks (highlighted with a red dashed box)
- Python 3 (highlighted with a red dashed box)

```
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
print('Hello Pandas')
```

```
s = pd.Series([1,3,5,np.nan,6,8])  
s
```

```
dates = pd.date_range('20170301',  
                      periods=6)  
dates
```

localhost:8888/notebooks/Documents/SCDBA/python/PythonPandasFinance.ipynb

jupyter PythonPandasFinance (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help

Code CellToolbar

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
print('Hello Pandas')
```

Hello Pandas

```
In [2]: s = pd.Series([1,3,5,np.nan,6,8])
s
```

```
Out[2]: 0    1.0
1    3.0
2    5.0
3    NaN
4    6.0
5    8.0
dtype: float64
```

```
In [3]: dates = pd.date_range('20170301', periods=6)
dates
```

```
Out[3]: DatetimeIndex(['2017-03-01', '2017-03-02', '2017-03-03', '2017-03-04',
                       '2017-03-05', '2017-03-06'],
                      dtype='datetime64[ns]', freq='D')
```

```
df = pd.DataFrame(np.random.randn(6,4),  
index=dates, columns=list('ABCD'))  
df
```

```
df = pd.DataFrame(np.random.randn(6,4), index=dates, columns=list('ABCD'))  
df
```

	A	B	C	D
2017-03-01	-1.613934	0.168983	0.686730	1.520077
2017-03-02	0.656124	-1.155197	-1.969431	0.444569
2017-03-03	-0.242393	-0.372972	-0.310283	1.102838
2017-03-04	-0.034627	-0.422640	0.489834	1.498832
2017-03-05	-1.022516	-0.841598	-0.136151	-0.767695
2017-03-06	1.208003	3.114586	-0.451570	0.579479

```
df = pd.DataFrame(np.random.randn(4,6),  
index=['student1','student2','student3',  
'student4'], columns=list('ABCDEF'))  
df
```

```
df = pd.DataFrame(np.random.randn(4,6), index=['student1','student2','student3','student4'], columns=list('ABCDEF'))  
df
```

	A	B	C	D	E	F
student1	-0.420406	0.829262	-0.326521	-0.037699	-1.350555	-0.617676
student2	0.310825	-0.356479	0.149704	-0.685609	-0.744307	-0.488782
student3	-1.295312	0.765656	1.701502	-0.415809	-0.454114	0.397702
student4	0.979525	0.367767	1.869465	0.988012	0.916746	0.165911

```
df2 = pd.DataFrame({ 'A' : 1.,
'B' : pd.Timestamp('20170322'),
'C' : pd.Series(2.5,index=list(range(4)),dtype='float32'),
'D' : np.array([3] * 4,dtype='int32'),
'E' : pd.Categorical(["test","train","test","train"]),
'F' : 'foo' })
df2
```

```
df2 = pd.DataFrame({ 'A' : 1.,
'B' : pd.Timestamp('20170322'),
'C' : pd.Series(2.5,index=list(range(4)),dtype='float32'),
'D' : np.array([3] * 4,dtype='int32'),
'E' : pd.Categorical(["test","train","test","train"]),
'F' : 'foo' })
df2
```

	A	B	C	D	E	F
0	1.0	2017-03-22	2.5	3	test	foo
1	1.0	2017-03-22	2.5	3	train	foo
2	1.0	2017-03-22	2.5	3	test	foo
3	1.0	2017-03-22	2.5	3	train	foo

df2.dtypes

```
df2.dtypes
```

```
A          float64
B    datetime64[ns]
C          float32
D          int32
E        category
F        object
dtype: object
```

Yahoo Finance Symbols: AAPL

Apple Inc. (AAPL)

Home Mail Flickr Tumblr News Sports Finance Celebrity Answers Groups Mobile More ▾

YAHOO!
FINANCE

Search for news, symbols or companies

Finance Home Originals Events Personal Finance Technology Markets Industries NEW My Screeners My Portfolio

S&P 500
2,344.02
-29.45 (-1.24%)

Dow 30
20,668.01
-237.85 (-1.14%)

Nasdaq
5,793.83
-107.70 (-1.83%)

Crude Oil
47.50
+0.16 (+0.34%)

Gold
1,245.40
-1.10 (-0.09%)

Quote Lookup

Search for symbols or companies: YHOO, GOOG, DIS



Symbols similar to 'aapl'

All Markets ▾

All (9)

Stocks (6)

Mutual Funds (0)

ETFs (1)

Indices (2)

Futures (0)

Currencies (0)

Symbol	Company Name	Last Price	Industry / Category	Type	Exchange
AAPL	Apple Inc.	139.84	Electronic Equipment	Stocks	NMS
AAPL.SW	Apple Inc.	140.70	N/A	Stocks	EBS
AAPL.MX	Apple Inc.	2678.68	Electronic Equipment	Stocks	MEX
AAPL34.FSA	Apple Inc.	0.00	N/A	Stocks	SAO
AAPL34.SA	Apple Inc.	43.14	Electronic Equipment	Stocks	SAO

<http://finance.yahoo.com/q?s=AAPL>

Apple Inc. (AAPL) - NasdaqGS



Search for news, symbols or companies

Search

Apple Inc. (AAPL)

NasdaqGS - NasdaqGS Delayed Price. Currency in USD

[Add to watchlist](#)

139.84 -1.62 (-1.15%)

At close: 4:00PM EDT

139.35 -0.49 (-0.35%)

After hours: 7:59PM EDT

[Summary](#) [Conversations](#) [Statistics](#) [Profile](#) [Financials](#) [Options](#) [Holders](#) [Historical Data](#) [Analysts](#)

Previous Close	141.46	Market Cap	733.68B
Open	142.11	Beta	1.45
Bid	139.31 x 100	PE Ratio (TTM)	16.79
Ask	139.40 x 1300	EPS (TTM)	8.33
Day's Range	139.73 - 142.80	Earnings Date	Apr 24, 2017 - Apr 28, 2017
52 Week Range	89.47 - 142.80	Dividend & Yield	2.28 (1.63%)
Volume	39,529,912	Ex-Dividend Date	N/A
Avg. Volume	26,889,183	1y Target Est	143.29



Trade prices are not sourced from all markets

<http://finance.yahoo.com/quote/AAPL?p=AAPL>

Yahoo Finance Charts: Apple Inc. (AAPL)

YAHOO! FINANCE

Go to Quote Summary Page



S&P 500

2,344.02

-29.45 (-1.24%)

Dow 30

20,668.01

-237.85 (-1.14%)

Nasdaq

5,793.83

-107.70 (-1.83%)

Crude Oil

47.50

+0.16 (+0.34%)

Gold

1,245.50

-1.00 (-0.08%)

Apple Inc. (AAPL) 139.84 -1.62 (-1.15%) As of 4:00PM EDT. Market closed.



AAPL 139.84

Open 142.11

Close 139.84

Low 139.73

High 142.80

Vol 39.53M

% Chg 63.27%

YAHOO!
FINANCE

Jan 7 '13

Jan 6 '14

Jan 5 '15

Jan 4 '16

Jan 2 '17

60.00

70.00

80.00

90.00

100.00

110.00

120.00

130.00

140.00

139.84

139.84

139.84

139.84

39.53M

Apple Inc. (AAPL) Historical Data

Home Mail Flickr Tumblr News Sports Finance Celebrity Answers Groups Mobile More ▾



Search for news, symbols or companies

Search

Finance Home Originals Events Personal Finance Technology Markets Industries NEW My Screeners My Portfolio

S&P 500
2,344.02
-29.45 (-1.24%)

Dow 30
20,668.01
-237.85 (-1.14%)

Nasdaq
5,793.83
-107.70 (-1.83%)

Crude Oil
47.50
+0.16 (+0.34%)

Gold
1,245.60
-0.90 (-0.07%)

Apple Inc. (AAPL)

NasdaqGS - NasdaqGS Delayed Price. Currency in USD

Add to watchlist

139.84 -1.62 (-1.15%) 139.35 -0.49 (-0.35%)

At close: 4:00PM EDT

After hours: 7:59PM EDT

Summary

Conversations

Statistics

Profile

Financials

Options

Holders

Historical Data

Analysts

Thank you for helping us improve your Yahoo experience

Learn more about your feedback.

Time Period: Mar 22, 2016 - Mar 22, 2017

Show: Historical Prices

Frequency: Daily

Apply

Currency in USD

Download Data

Date	Open	High	Low	Close	Adj Close*	Volume
Mar 21, 2017	142.11	142.80	139.73	139.84	139.84	39,116,800

<http://finance.yahoo.com/q/hp?s=AAPL+Historical+Prices>

Yahoo Finance Historical Prices

Apple Inc. (AAPL)



Search for news, symbols or companies

Search

Time Period: Mar 22, 2016 - Mar 22, 2017

Show: Historical Prices

Frequency: Daily

Apply

Currency in USD



Date	Open	High	Low	Close	Adj Close*	Volume
Mar 21, 2017	142.11	142.80	139.73	139.84	139.84	39,116,800
Mar 20, 2017	140.40	141.50	140.23	141.46	141.46	21,542,000
Mar 17, 2017	141.00	141.00	139.89	139.99	139.99	43,885,000
Mar 16, 2017	140.72	141.02	140.26	140.69	140.69	19,232,000
Mar 15, 2017	139.41	140.75	139.03	140.46	140.46	25,691,800
Mar 14, 2017	139.30	139.65	138.84	138.99	138.99	15,309,100
Mar 13, 2017	138.85	139.43	138.82	139.20	139.20	17,421,700
Mar 10, 2017	139.25	139.36	138.64	139.14	139.14	19,612,800
Mar 09, 2017	138.74	138.79	137.05	138.68	138.68	22,155,900
Mar 08, 2017	138.95	139.80	138.82	139.00	139.00	18,707,200
Mar 07, 2017	139.06	139.98	138.79	139.52	139.52	17,446,300
Mar 06, 2017	139.37	139.77	138.60	139.34	139.34	21,750,000
Mar 03, 2017	138.78	139.83	138.59	139.78	139.78	21,108,100

<http://finance.yahoo.com/quote/AAPL/history>

Yahoo Finance Historical Prices

Apple Inc. (AAPL)



Search for news, symbols or companies

Search

Time Period: Dec 12, 1980 - Mar 22, 2017 ▼

Show: Historical Prices ▼

Frequency: Daily ▼

Apply

Currency: USD

Download Data ▼

1D 5D 3M 6M

YTD 1Y 5Y MAX

Date

Start Date End Date

		High	Low	Close	Adj Close*	Volume
Mar 21, 2017	12/12/1980	12.80	139.73	139.84	139.84	39,116,800
Mar 20, 2017	3/22/2017	41.50	140.23	141.46	141.46	21,542,000
Mar 17, 2017		41.00	139.89	139.99	139.99	43,885,000
Mar 16, 2017	140.72	141.02	140.26	140.69	140.69	19,232,000
Mar 15, 2017	139.41	140.75	139.03	140.46	140.46	25,691,800
Mar 14, 2017	139.30	139.65	138.84	138.99	138.99	15,309,100
Mar 13, 2017	138.85	139.43	138.82	139.20	139.20	17,421,700
Mar 10, 2017	139.25	139.36	138.64	139.14	139.14	19,612,800
Mar 09, 2017	138.74	138.79	137.05	138.68	138.68	22,155,900
Mar 08, 2017	138.95	139.80	138.82	139.00	139.00	18,707,200

Yahoo Finance Historical Prices

Apple Inc. (AAPL)



Search for news, symbols or companies

Search

Time Period: Dec 12, 1980 - Mar 22, 2017

Show: Historical Prices

Frequency: Daily

Apply

Currency in USD

Download Data

Volume

Date	Open	High	Low	Close	Adj Close*	Volume
Mar 21, 2017	142.11	142.80	139.73	139.84	139.84	39,116,800
Mar 20, 2017	140.40	141.50	140.23	141.46	141.46	21,542,000
Mar 17, 2017	141.00	141.00	139.89	139.99	139.99	43,885,000
Mar 16, 2017	140.72	141.02	140.26	140.69	140.69	19,232,000
Mar 15, 2017	139.41	140.75	139.03	140.46	140.46	25,691,800
Mar 14, 2017	139.30	139.65	138.84	138.99	138.99	15,309,100
Mar 13, 2017	138.85	139.43	138.82	139.20	139.20	17,421,700
Mar 10, 2017	139.25	139.36	138.64	139.14	139.14	19,612,800
Mar 09, 2017	138.74	138.79	137.05	138.68	138.68	22,155,900
Mar 08, 2017	138.95	139.80	138.82	139.00	139.00	18,707,200

Yahoo Finance Historical Prices

<http://ichart.finance.yahoo.com/table.csv?s=AAPL>

table.csv

Date	Open	High	Low	Close	Volume	Adj Close
2017-03-21	142.110001	142.800003	139.729996	139.839996	39116800	139.839996
2017-03-20	140.399994	141.50	140.229996	141.460007	20213100	141.460007
2017-03-17	141.00	141.00	139.889999	139.990005	43597400	139.990005
2017-03-16	140.720001	141.020004	140.259995	140.690002	19132500	140.690002
2017-03-15	139.410004	140.75	139.029999	140.460007	25566800	140.460007
2017-03-14	139.300003	139.649994	138.839996	138.990005	15189700	138.990005
2017-03-13	138.850006	139.429993	138.820007	139.199997	17042400	139.199997
2017-03-10	139.25	139.360001	138.639999	139.139999	19488000	139.139999
2017-03-09	138.740005	138.789993	137.050003	138.679993	22065200	138.679993
2017-03-08	138.949997	139.800003	138.820007	139.00	18681800	139.00
2017-03-07	139.059998	139.979996	138.789993	139.520004	17267500	139.520004
2017-03-06	139.369995	139.770004	138.600006	139.339996	21155300	139.339996
2017-03-03	138.779999	139.830002	138.589996	139.779999	21108100	139.779999
2017-03-02	140.00	140.279999	138.759995	138.960007	26153300	138.960007
2017-03-01	137.889999	140.149994	137.600006	139.789993	36272400	139.789993
2017-02-28	137.080002	137.440002	136.699997	136.990005	23403500	136.990005
2017-02-27	137.139999	137.440002	136.279999	136.929993	20196400	136.929993
2017-02-24	135.910004	136.660004	135.279999	136.660004	21690900	136.660004
2017-02-23	137.380005	137.479996	136.300003	136.529999	20704100	136.529999
2017-02-22	136.429993	137.119995	136.110001	137.110001	20745300	137.110001

Yahoo Finance Charts

Alphabet Inc. (GOOG)



Dow Jones Industrial Average (^DJI)

YAHOO! FINANCE

Go to Quote Summary Page



S&P 500
2,344.02
-29.45 (-1.24%)

Dow 30
20,668.01
-237.85 (-1.14%)

Nasdaq
5,793.83
-107.70 (-1.83%)

Crude Oil
47.50
+0.16 (+0.34%)

Gold
1,244.90
-1.60 (-0.13%)

Dow Jones Industrial Average (^DJI) 20,668.01 -237.85 (-1.14%) As of 4:36PM EDT. Market closed.



<http://finance.yahoo.com/chart/^DJI>

TSEC weighted index (^TWII) - Taiwan



S&P 500
2,344.02
-29.45 (-1.24%)

Dow 30
20,668.01
-237.85 (-1.14%)

Nasdaq
5,793.83
-107.70 (-1.83%)

Crude Oil
47.50
+0.16 (+0.34%)

Gold
1,245.10
-1.40 (-0.11%)

TSEC weighted index (^TWII) 9,868.95 -103.54 (-1.04%) As of 10:38AM CST. Taiwan Delayed Price. Market open.



Taiwan Semiconductor Manufacturing Company Limited (2330.TW)

Home Mail Flickr Tumblr News Sports Finance Celebrity Answers Groups Mobile More ▾



Search for news, symbols or companies

Search

Finance Home Originals Events Personal Finance Technology Markets Industries NEW My Screeners My Portfolio

S&P 500

2,344.02

-29.45 (-1.24%)

Dow 30

20,668.01

-237.85 (-1.14%)

Nasdaq

5,793.83

-107.70 (-1.83%)

Crude Oil

47.50

+0.16 (+0.34%)

Gold

1,244.90

-1.60 (-0.13%)

Taiwan Semiconductor Manufacturing Company Limited (2330.TW)

Taiwan - Taiwan Delayed Price. Currency in TWD

Add to watchlist

192.50 **-2.50 (-1.28%)**

As of 9:52AM CST. Market open.

Summary

Conversations

Statistics

Profile

Financials

Options

Holders

Historical Data

Analysts

Previous Close

195.00

Market Cap

4.98T

1D 5D 1M 6M YTD 1Y 2Y 5Y 10Y MAX

Interactive chart

Open

192.50

Beta

N/A

Bid

192.00 x

PE Ratio (TTM)

14.90

Ask

192.50 x

EPS (TTM)

12.89

Day's Range

191.50 - 193.00

Earnings Date

Apr 13, 2017

52 Week Range

154.00 - 193.00

Dividend & Yield

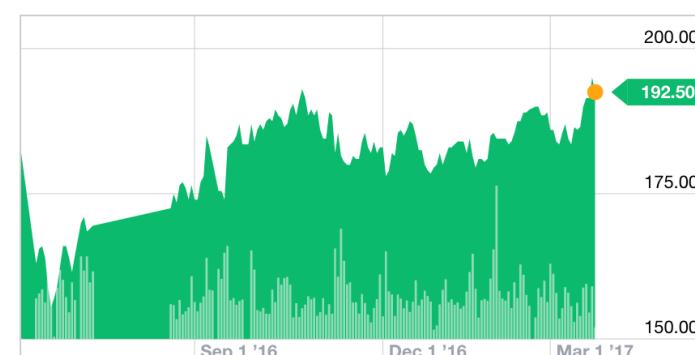
N/A (N/A)

Volume

6,977,000

Ex-Dividend Date

N/A



<http://finance.yahoo.com/q?s=2330.TW>

Yahoo Finance Charts

TSMC (2330.TW)



S&P 500
2,344.02
-29.45 (-1.24%)

Dow 30
20,668.01
-237.85 (-1.14%)

Nasdaq
5,793.83
-107.70 (-1.83%)

Crude Oil
47.50
+0.16 (+0.34%)

Gold
1,245.00
-1.50 (-0.12%)

Taiwan Semiconductor Manufacturing Company Limited (2330.TW) 192.00 -3.00 (-1.54%)

As of 10:29AM CST. Taiwan Delayed Price. Market open.



```

import pandas as pd
import pandas_datareader.data as web
df = web.DataReader('AAPL', data_source='yahoo',
start='1/1/2010', end='3/21/2017')
df.to_csv('AAPL.csv')
df.tail()

```

```

import pandas as pd
import pandas_datareader.data as web
#df = web.DataReader('AAPL', 'yahoo')
df = web.DataReader('AAPL', data_source='yahoo', start='1/1/2010', end='3/21/2017')
#df = web.DataReader('AAPL', data_source='google', start='1/1/2010', end='3/21/2017')
df.to_csv('AAPL.csv')
df.tail()

```

	Open	High	Low	Close	Volume	Adj Close
Date						
2017-03-15	139.410004	140.750000	139.029999	140.460007	25566800	140.460007
2017-03-16	140.720001	141.020004	140.259995	140.690002	19132500	140.690002
2017-03-17	141.000000	141.000000	139.889999	139.990005	43597400	139.990005
2017-03-20	140.399994	141.500000	140.229996	141.460007	20213100	141.460007
2017-03-21	142.110001	142.800003	139.729996	139.839996	39116800	139.839996

```
df = web.DataReader('GOOG',
data_source='yahoo', start='1/1/1980',
end='3/21/2017')
df.head(10)
```

```
df = web.DataReader('GOOG', data_source='yahoo', start='1/1/1980', end='3/21/2017')
df.head(10)
```

	Open	High	Low	Close	Volume	Adj Close
Date						
2004-08-19	100.000168	104.060182	95.960165	100.340176	44871300	50.119968
2004-08-20	101.010175	109.080187	100.500174	108.310183	22942800	54.100990
2004-08-23	110.750191	113.480193	109.050183	109.400185	18342800	54.645447
2004-08-24	111.240189	111.600192	103.570177	104.870176	15319700	52.382705
2004-08-25	104.960181	108.000187	103.880180	106.000184	9232100	52.947145
2004-08-26	104.950180	107.950188	104.660179	107.910182	7128600	53.901190
2004-08-27	108.100185	108.620186	105.690180	106.150181	6241200	53.022069
2004-08-30	105.280178	105.490184	102.010172	102.010172	5221400	50.954132
2004-08-31	102.300173	103.710180	102.160177	102.370175	4941200	51.133953
2004-09-01	102.700174	102.970180	99.670169	100.250171	9181600	50.075011

`df.tail(10)`

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

	Open	High	Low	Close	Volume	Adj Close
Date						
2017-03-08	833.510010	838.150024	831.789978	835.369995	988700	835.369995
2017-03-09	836.000000	842.000000	834.210022	838.679993	1259900	838.679993
2017-03-10	843.280029	844.909973	839.500000	843.250000	1701100	843.250000
2017-03-13	844.000000	848.684998	843.250000	845.539978	1149500	845.539978
2017-03-14	843.640015	847.239990	840.799988	845.619995	779900	845.619995
2017-03-15	847.590027	848.630005	840.770020	847.200012	1379600	847.200012
2017-03-16	849.030029	850.849976	846.130005	848.780029	970400	848.780029
2017-03-17	851.609985	853.400024	847.109985	852.119995	1712300	852.119995
2017-03-20	850.010010	850.219971	845.150024	848.400024	1190300	848.400024
2017-03-21	851.400024	853.500000	829.020020	830.460022	2442900	830.460022

df.count()

```
df.count()
```

Open	3169
High	3169
Low	3169
Close	3169
Volume	3169
Adj Close	3169
dtype:	int64

df.ix['2015-12-31']

```
df.ix['2015-12-31']
```

```
Open          7.695000e+02
High          7.695000e+02
Low           7.583400e+02
Close          7.588800e+02
Volume         1.489600e+06
Adj Close      7.588800e+02
Name: 2015-12-31 00:00:00, dtype: float64
```

```
df.to_csv('2330.TW.Yahoo.Finance.Data.csv')
```

2330.TW.Yahoo.Finance.Data.csv *

	Date	Open	High	Low	Close	Volume	Adj Close
1	2010-01-01	64.5	64.5	64.5	64.5	0	52.8308
2	2010-01-04	65.0	65.0	64.0	64.9	39407000	53.1584
3	2010-01-05	65.0	65.1	63.9	64.5	37138000	52.8308
4	2010-01-06	64.5	64.9	63.7	64.9	49261000	53.1584
5	2010-01-07	64.9	65.0	64.2	64.2	42134000	52.5851
6	2010-01-08	63.5	64.3	63.5	64.0	46076000	52.4213
7	2010-01-11	64.0	64.9	63.5	64.5	36799000	52.8308
8	2010-01-12	64.4	64.4	63.3	63.6	49853000	52.0936
9	2010-01-13	63.0	63.1	62.6	62.8	47976000	51.4384
10	2010-01-14	63.6	63.6	63.0	63.2	36149000	51.766
11	2010-01-15	62.9	63.5	62.8	63.5	47852000	52.0117
12	2010-01-18	62.8	63.1	62.8	62.9	30136000	51.5203
13	2010-01-19	63.0	63.2	62.0	62.5	47202000	51.1926
14	2010-01-20	62.9	63.2	62.2	63.0	52281000	51.6022

Python Pandas for Finance

Python Pandas for Finance

```
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
```

```
#Python for Stocks: 1
#Source: https://mapattack.wordpress.com/2017/02/12/using-python-for-stocks-1/
#Import Packages Required
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|
```

Python Pandas for Finance

```
#Read Stock Data from Yahoo Finance
end = dt.datetime.now()
#start = dt.datetime(end.year-2, end.month, end.day)
start = dt.datetime(2015, 1, 1)
df = web.DataReader("AAPL", 'yahoo', start, end)
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()
```

```
#Read Stock Data from Yahoo Finance
end = dt.datetime.now()
#start = dt.datetime(end.year-2, end.month, end.day)
start = dt.datetime(2015, 1, 1)
df = web.DataReader("AAPL", 'yahoo', start, end)
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()
```

	Open	High	Low	Close	Volume	Adj Close
Date						
2017-03-15	139.410004	140.750000	139.029999	140.460007	25566800	140.460007
2017-03-16	140.720001	141.020004	140.259995	140.690002	19132500	140.690002
2017-03-17	141.000000	141.000000	139.889999	139.990005	43597400	139.990005
2017-03-20	140.399994	141.500000	140.229996	141.460007	20213100	141.460007
2017-03-21	142.110001	142.800003	139.729996	139.839996	39116800	139.839996

Finance Data from Quandl

```
import quandl
```

```
df = quandl.get("WIKI/AAPL", start_date="2015-01-01", end_date="2017-10-31")
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()
```

```
import quandl
df = quandl.get("WIKI/AAPL", start_date="2015-01-01", end_date="2017-10-31")
df.to_csv('AAPL.csv')
df.from_csv('AAPL.csv')
df.tail()
```

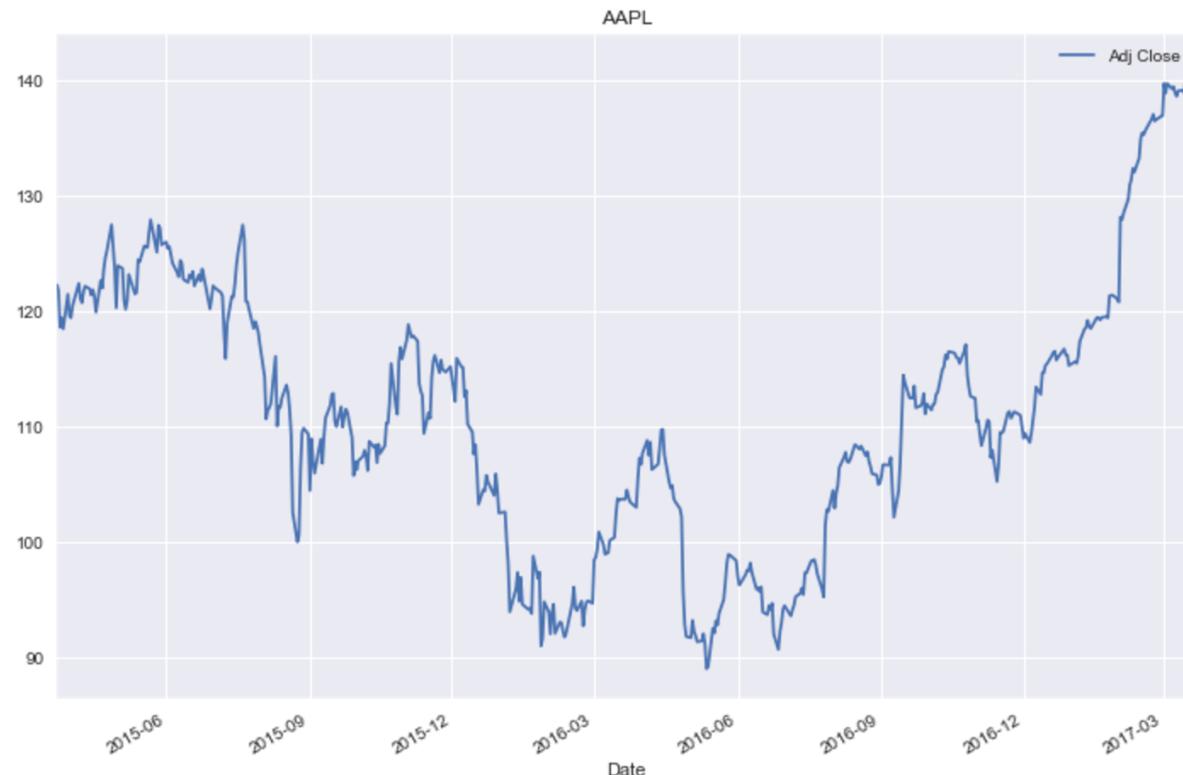
	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio	Adj. Open	Adj. High	Adj. Low	Adj. Close	Adj. Volume
Date												
2017-10-25	156.91	157.5500	155.27	156.405	20126554.0	0.0	1.0	156.91	157.5500	155.27	156.405	20126554.0
2017-10-26	157.23	157.8295	156.78	157.410	16751691.0	0.0	1.0	157.23	157.8295	156.78	157.410	16751691.0
2017-10-27	159.29	163.6000	158.70	163.050	43904150.0	0.0	1.0	159.29	163.6000	158.70	163.050	43904150.0
2017-10-30	163.89	168.0700	163.72	166.720	43923292.0	0.0	1.0	163.89	168.0700	163.72	166.720	43923292.0
2017-10-31	167.90	169.6499	166.94	169.040	35474672.0	0.0	1.0	167.90	169.6499	166.94	169.040	35474672.0

Python Pandas for Finance

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

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

```
<matplotlib.axes._subplots.AxesSubplot at 0x1150bac88>
```



Python Pandas for Finance

```
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')
```

Python Pandas for Finance

<matplotlib.text.Text at 0x115630860>



Python Pandas for Finance

```
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')
```



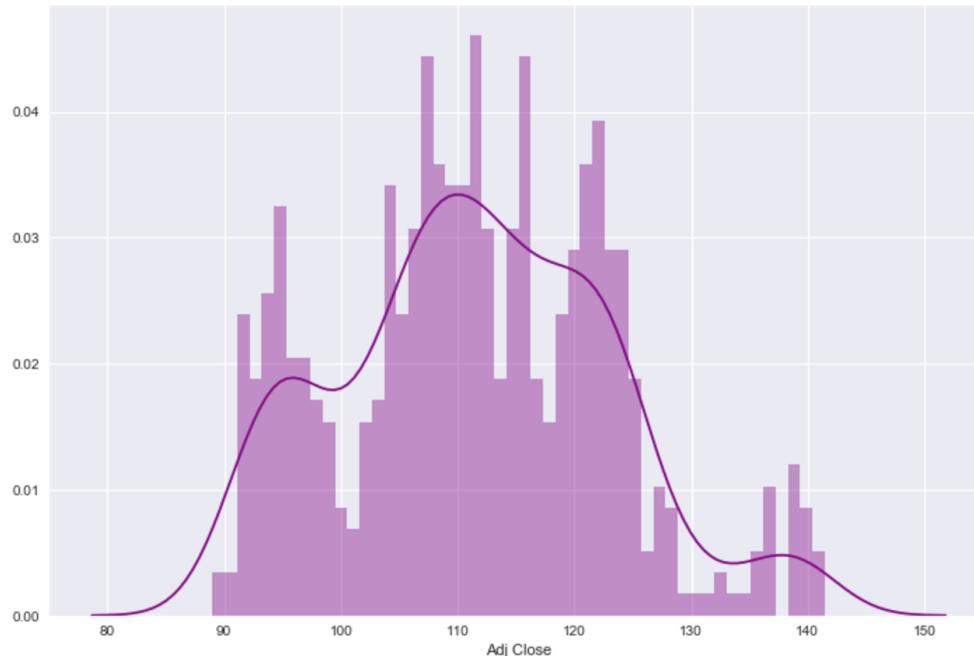
Python Pandas for Finance

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

```
plt.figure(figsize=(12,8))
sns.distplot(df['Adj Close'].dropna(), bins=50, color='purple')

/Users/imyday/anaconda/lib/python3.6/site-packages/statsmodels/nonparametric/kdetools
g: using a non-integer number instead of an integer will result in an error in the fu
    y = X[:m/2+1] + np.r_[0,X[m/2+1:],0]*1j

<matplotlib.axes._subplots.AxesSubplot at 0x116309780>
```

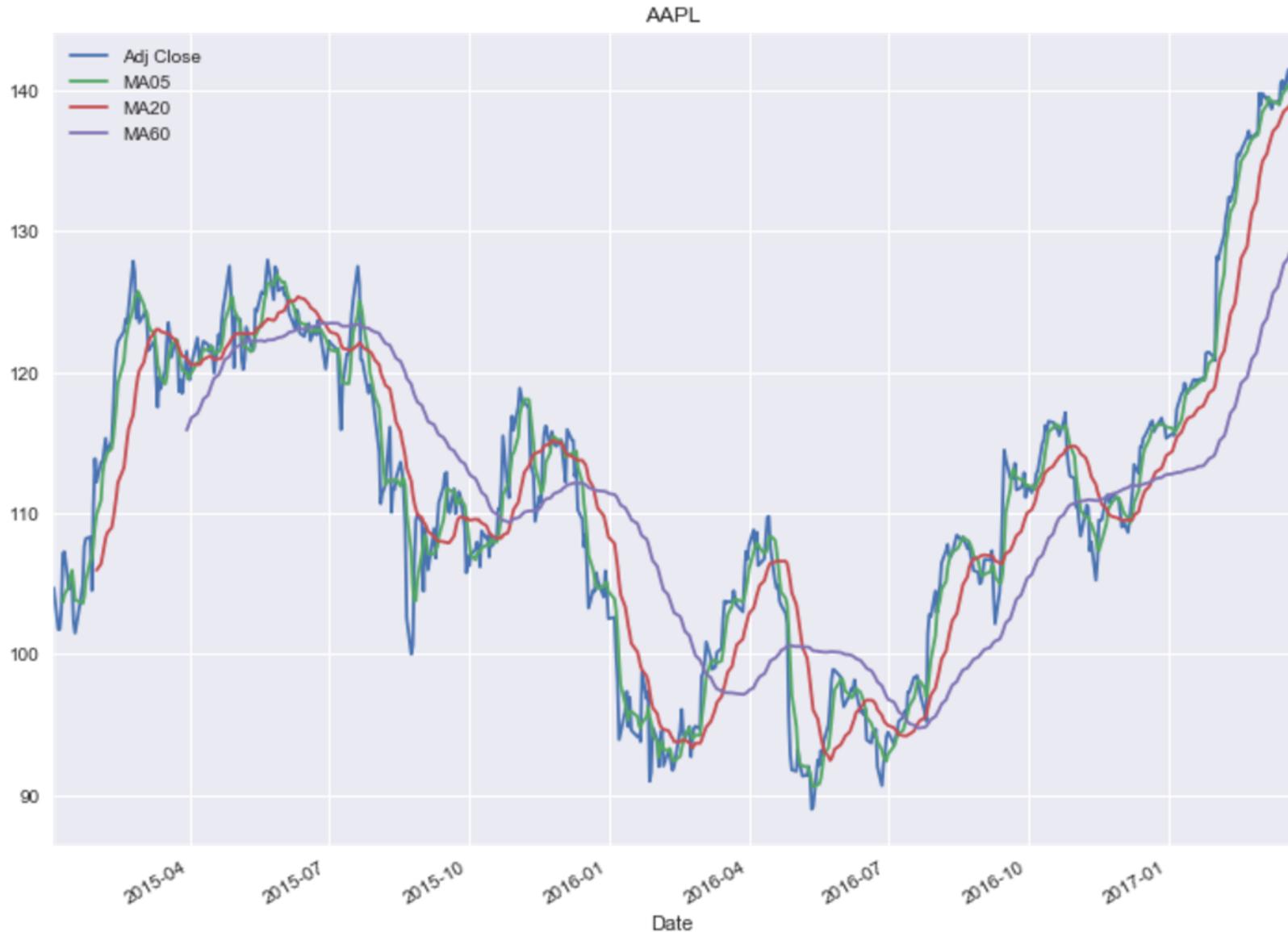


Python Pandas for Finance

```
# simple moving averages
df['MA05'] = df['Adj Close'].rolling(5).mean()
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()
```

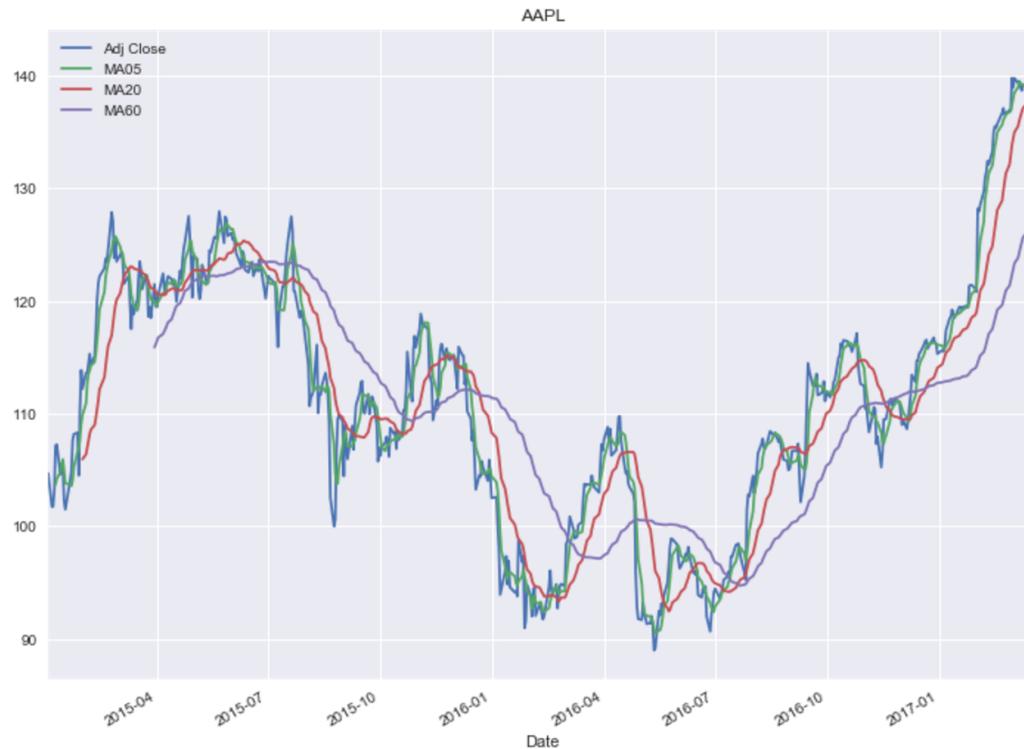
Python Pandas for Finance



Python Pandas for Finance

```
# 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)
```



```

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(2015, 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()

```

Examples: Python Pandas for Finance

```
In [11]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

import pandas.io.data as web
import random

import datetime
import time
import timeit

import io
import os

import re
import codecs
import requests
get_ipython().magic('matplotlib inline')

from scipy import stats

#pd.set_option('display.notebook_repr_html', False)
pd.set_option('display.max_columns', 15)
pd.set_option('display.max_rows', 10)
pd.set_option('precision', 3)

def getDateNow():
    strnow = datetime.datetime.now().strftime("%Y%m%d_%H%M%S")
    return strnow

print('Hello Pandas')
print(getDateNow())
```

```
Hello Pandas
20160323_145708
```

```
sSymbol = "AAPL"
#sSymbol = "GOOG"
#sSymbol = "IBM"
#sSymbol = "MSFT"
#sSymbol = "^TWII"
#sSymbol = "000001.SS"
#sSymbol = "2330.TW"
#sSymbol = "2317.TW"

# sURL = "http://ichart.finance.yahoo.com/table.csv?s=AAPL"
# sBaseUrl = "http://ichart.finance.yahoo.com/table.csv?s="
sURL = "http://ichart.finance.yahoo.com/table.csv?s=" + sSymbol
#req = requests.get("http://ichart.finance.yahoo.com/table.csv?s=2330.TW")
#req = requests.get("http://ichart.finance.yahoo.com/table.csv?s=AAPL")
req = requests.get(sURL)

sText = req.text
#print(sText)
#df = web.DataReader(sSymbol, 'yahoo', starttime, endtime)
#df = web.DataReader("2330.TW", 'yahoo')

sPath = "data/"
sPathFilename = sPath + sSymbol + ".csv"
print(sPathFilename)

f = open(sPathFilename, 'w')
f.write(sText)
f.close()
sIOdata = io.StringIO(sText)
df = pd.DataFrame.from_csv(sIOdata)
df.head(5)
```

```
In [13]: starttime = datetime.datetime(2000, 1, 1)
endtime = datetime.datetime(2015, 12, 31)
sSymbol = "AAPL"
#sSymbol = "GOOG"
#sSymbol = "IBM"
#sSymbol = "MSFT"
#sSymbol = "^TWII"
#sSymbol = "000001.SS"
#sSymbol = "2330.TW"
#sSymbol = "2317.TW"
#"^TWII"
#"AAPL"
#SHA:000016"
#"600000.SS"
#"2330.TW"
df = web.DataReader(sSymbol, 'yahoo', starttime, endtime)
#df_01 = web.DataReader("2330.TW", 'yahoo')
sSymbol = sSymbol.replace(":", "_")
sSymbol = sSymbol.replace("^", "_")
sPath = "data/financedata/"
#/users/imyday/SCDBA/data/ #Mac OS X
#sPath = "C:\data\" #Windows

sPathFilename = sPath + sSymbol + "_Yahoo_Finance.csv"
print(sPathFilename)
df.to_csv(sPathFilename)
df.head(5)

data/financedata/AAPL_Yahoo_Finance.csv
```

Out[13]:

	Open	High	Low	Close	Volume	Adj Close
Date						
2000-01-03	104.875	112.500	101.688	111.938	133949200	3.702
2000-01-04	108.250	110.625	101.188	102.500	128094400	3.390

df.tail(5)

```
In [14]: df.tail(5)
```

Out[14]:

	Open	High	Low	Close	Volume	Adj Close
Date						
2015-12-24	109.00	109.00	107.95	108.03	13596700	107.447
2015-12-28	107.59	107.69	106.18	106.82	26704200	106.243
2015-12-29	106.96	109.43	106.86	108.74	30931200	108.153
2015-12-30	108.58	108.70	107.18	107.32	25213800	106.741
2015-12-31	107.01	107.03	104.82	105.26	40912300	104.692

```
sSymbol = "AAPL"

# sURL = "http://ichart.finance.yahoo.com/table.csv?s=AAPL"
sURL = "http://ichart.finance.yahoo.com/table.csv?s=" + sSymbol
#req = requests.get("http://ichart.finance.yahoo.com/table.csv?s=AAPL")

req = requests.get(sURL)

sText = req.text
#print(sText)

sPath = "data/"
sPathFilename = sPath + sSymbol + ".csv"
print(sPathFilename)

f = open(sPathFilename, 'w')
f.write(sText)
f.close()
sIOdata = io.StringIO(sText)
df = pd.DataFrame.from_csv(sIOdata)
df.head(5)
```

```
In [15]: sSymbol = "AAPL"
#sSymbol = "GOOG"
#sSymbol = "IBM"
#sSymbol = "MSFT"
#sSymbol = "^TWII"
#sSymbol = "000001.SS"
#sSymbol = "2330.TW"
#sSymbol = "2317.TW"

# sURL = "http://ichart.finance.yahoo.com/table.csv?s=AAPL"
# sBaseUrl = "http://ichart.finance.yahoo.com/table.csv?s="
sURL = "http://ichart.finance.yahoo.com/table.csv?s=" + sSymbol
#req = requests.get("http://ichart.finance.yahoo.com/table.csv?s=2330.TW")
#req = requests.get("http://ichart.finance.yahoo.com/table.csv?s=AAPL")
req = requests.get(sURL)

sText = req.text
#print(sText)
#df = web.DataReader(sSymbol, 'yahoo', starttime, endtime)
#df = web.DataReader("2330.TW", 'yahoo')

sPath = "data/"
sPathFilename = sPath + sSymbol + ".csv"
print(sPathFilename)

f = open(sPathFilename, 'w')
f.write(sText)
f.close()
sIOdata = io.StringIO(sText)
df = pd.DataFrame.from_csv(sIOdata)
df.head(5)
```

data/AAPL.csv

Out[15]:

	Open	High	Low	Close	Volume	Adj Close
Date						
2016-03-22	105.25	107.29	105.21	106.72	32232600	106.72
2016-03-21	105.93	107.65	105.14	105.91	35180800	105.91
2016-03-18	106.34	106.50	105.19	105.92	43402300	105.92
2016-03-17	105.52	106.47	104.96	105.80	34244600	105.80
2016-03-16	104.61	106.31	104.59	105.97	37893800	105.97

```
def getYahooFinanceData(sSymbol, starttime, endtime, sDir):
    #GetMarketFinanceData_From_YahooFinance
    # "^TWII"
    # "000001.SS"
    # "AAPL"
    #SHA:000016"
    # "600000.SS"
    # "2330.TW"
    #sSymbol = "^TWII"
    starttime = datetime.datetime(2000, 1, 1)
    endtime = datetime.datetime(2015, 12, 31)
    sPath = sDir
    #sPath = "data/financedata/"
    df_YahooFinance = web.DataReader(sSymbol, 'yahoo', starttime, endtime)
    #df_01 = web.DataReader("2330.TW", 'yahoo')
    sSymbol = sSymbol.replace(":", "_")
    sSymbol = sSymbol.replace("^", "_")
    sPathFilename = sPath + sSymbol + "_Yahoo_Finance.csv"
    df_YahooFinance.to_csv(sPathFilename)
    #df_YahooFinance.head(5)
    return sPathFilename
#End def getYahooFinanceData(sSymbol, starttime, endtime, sDir):
```

```
In [16]: def getYahooFinanceData(sSymbol, starttime, endtime, sDir):
    #GetMarketFinanceData_From_YahooFinance
    #"ATWII"
    #"000001.SS"
    #AAPL
    #SHA:000016"
    #"600000.SS"
    #2330.TW"
    #sSymbol = "ATWII"
    starttime = datetime.datetime(2000, 1, 1)
    endtime = datetime.datetime(2015, 12, 31)
    sPath = sDir
    #sPath = "data/financedata/"
    df_YahooFinance = web.DataReader(sSymbol, 'yahoo', starttime, endtime)
    #df_01 = web.DataReader("2330.TW", 'yahoo')
    sSymbol = sSymbol.replace(":", "_")
    sSymbol = sSymbol.replace("^", "_")
    sPathFilename = sPath + sSymbol + "_Yahoo_Finance.csv"
    df_YahooFinance.to_csv(sPathFilename)
    #df_YahooFinance.head(5)
    return sPathFilename
#End def getYahooFinanceData(sSymbol, starttime, endtime, sDir):
```

```
sSymbol = "AAPL"
starttime = datetime.datetime(2000, 1, 1)
endtime = datetime.datetime(2015, 12, 31)
sDir = "data/financedata/"

sPathFilename = getYahooFinanceData(sSymbol, starttime, endtime,
sDir)
print(sPathFilename)
```

```
In [17]: sSymbol = "AAPL"
#sSymbol = "GOOG"
#sSymbol = "IBM"
#sSymbol = "MSFT"
#sSymbol = "^TWII"
#sSymbol = "000001.SS"
#sSymbol = "2330.TW"
#sSymbol = "2317.TW"
starttime = datetime.datetime(2000, 1, 1)
endtime = datetime.datetime(2015, 12, 31)
sDir = "data/financedata/"

sPathFilename = getYahooFinanceData(sSymbol, starttime, endtime, sDir)
print(sPathFilename)
```

data/financedata/AAPL_Yahoo_Finance.csv

```

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(2015, 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')

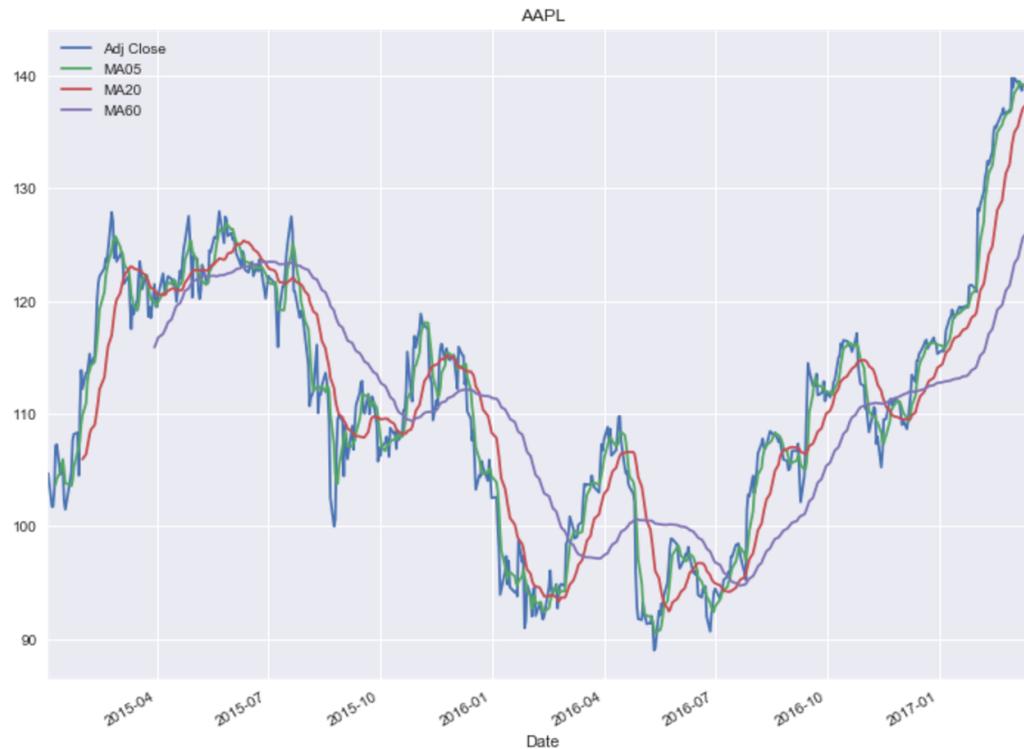
# simple moving averages
df['MA05'] = df['Adj Close'].rolling(5).mean() #5 days
df['MA20'] = df['Adj Close'].rolling(20).mean() #20 days
df['MA60'] = df['Adj Close'].rolling(60).mean() #60 days
df2 = pd.DataFrame({'Adj Close': df['Adj Close'], 'MA05': df['MA05'], 'MA20': df['MA20'], 'MA60': df['MA60']})
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()

```

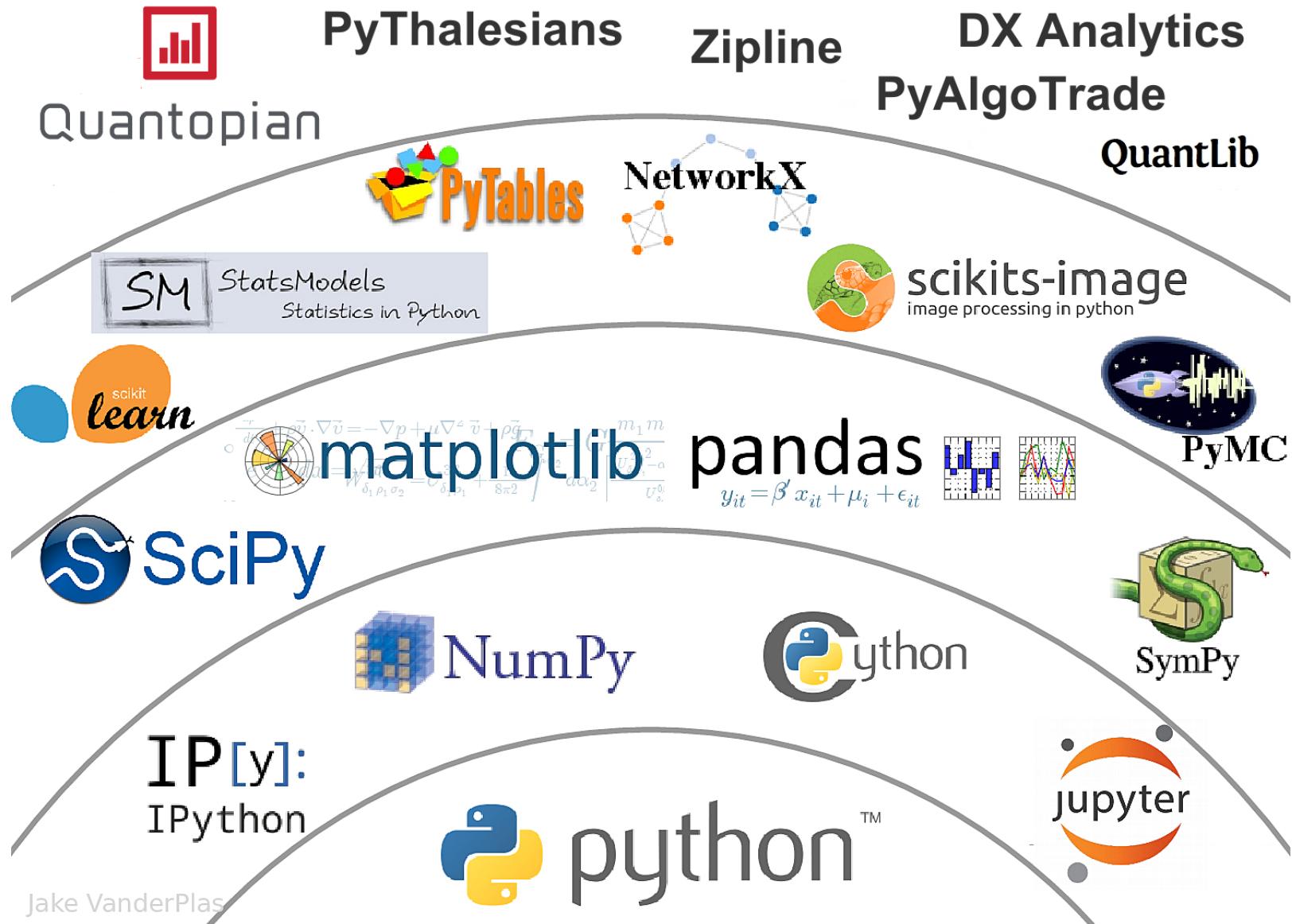
Python Pandas for Finance

```
# 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)
```



The Quant Finance PyData Stack



Quantopian

Q

Investor Relations

Allocations

Research

Community

Learn

Help

Log In

Sign Up

Leveling Wall Street's Playing Field

Quantopian inspires talented people everywhere to write investment algorithms.

Select authors may license their algorithms to us and get paid based on performance.

Start Coding

References

- 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
- Yves Hilpisch (2015), Derivatives Analytics with Python: Data Analysis, Models, Simulation, Calibration and Hedging, Wiley
- 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/>

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>