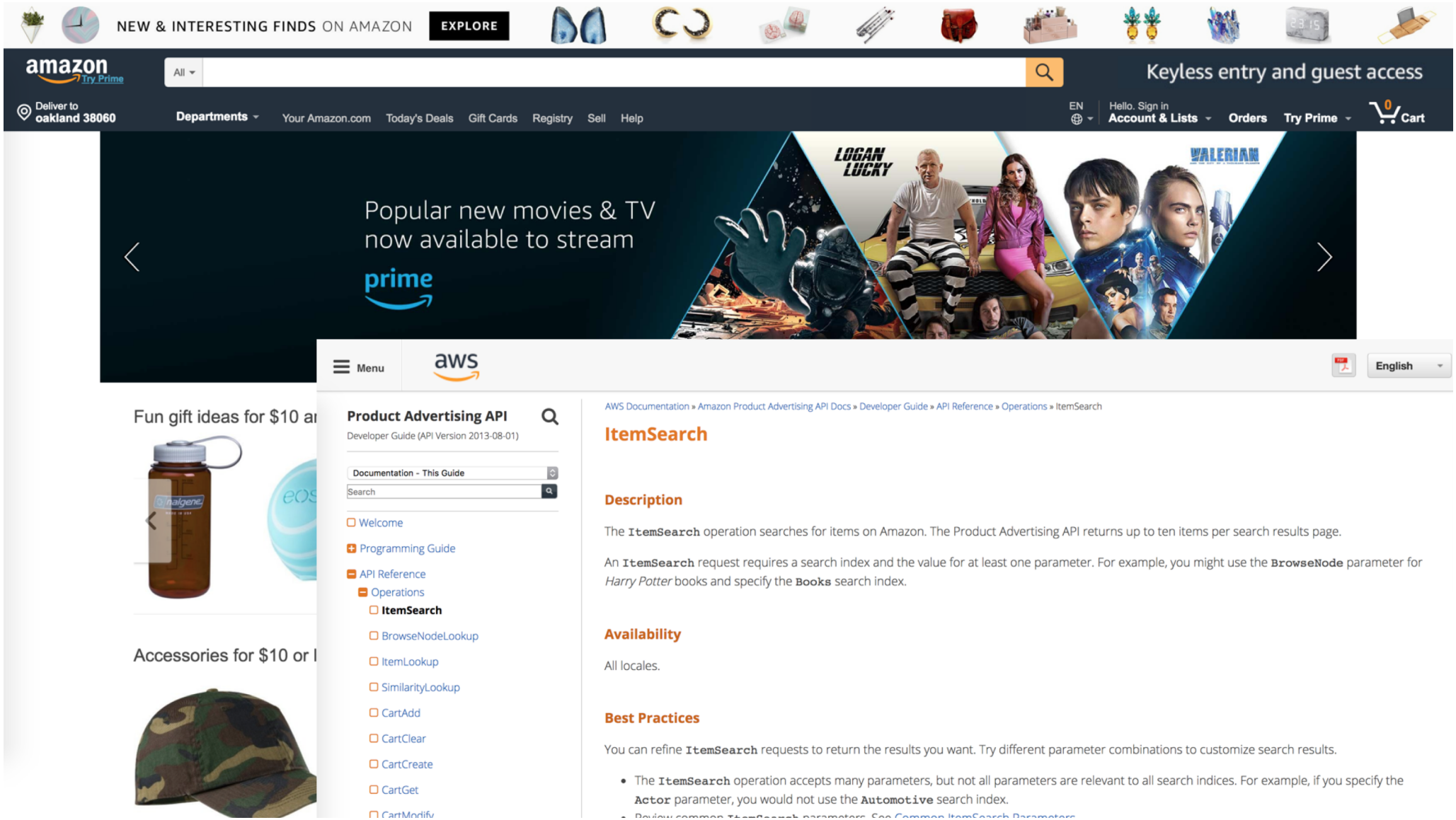
GOOD MORNING

DATA IS WORTHLESS


NEW & INTERESTING FINDS ON AMAZON
EXPLORE
amazon
Try Prime
All
Keyless entry and guest access
Deliver to
oakland 38060
Departments
Your Amazon.com
Today's Deals
Gift Cards
Registry
Sell
Help
EN
Hello. Sign in
Account & Lists
Orders
Try Prime
Cart
Popular new movies & TV
now available to stream
prime
LOGAN LUCKY
VALERIAN
Fun gift ideas for $10 ar
Accessories for $10 or l
Menu
aws
English
Product Advertising API
Developer Guide (API Version 2013-08-01)
Documentation - This Guide
Search
Welcome
Programming Guide
API Reference
Operations
ItemSearch
BrowseNodeLookup
ItemLookup
SimilarityLookup
CartAdd
CartClear
CartCreate
CartGet
CartModify
AWS Documentation » Amazon Product Advertising API Docs » Developer Guide » API Reference » Operations » ItemSearch
ItemSearch
Description
The ItemSearch operation searches for items on Amazon. The Product Advertising API returns up to ten items per search results page.
An ItemSearch request requires a search index and the value for at least one parameter. For example, you might use the BrowseNode parameter for Harry Potter books and specify the Books search index.
Availability
All locales.
Best Practices
You can refine ItemSearch requests to return the results you want. Try different parameter combinations to customize search results.
The ItemSearch operation accepts many parameters, but not all parameters are relevant to all search indices. For example, if you specify the Actor parameter, you would not use the Automotive search index.

Merely **having** more data does not give Amazon a strategic advantage.

Clothing, Shoes & Jewelry best sellers See more

We Have Recommendations for You

Sign in to see personalized recommendations

You don't have any recently viewed items.
View items on Amazon and we'll track them here.

ANALYTICS ARE WORTH PENNIES

NEW & INTERESTING FINDS ON AMAZON EXPLORE

amazon Try Prime | All | presentation clicker | Keyless entry and guest access

Deliver to oakland 38060 | Departments | Your Amazon.com | Today's Deals | Gift Cards | Registry | Sell | Help | EN | Hello. Sign in Account & Lists | Orders | Try Prime | Cart

All Electronics | Deals | Best Sellers | TV & Video | Audio & Home Theater | Computers | Camera & Photo | Wearable Technology | Car Electronics & GPS | Portable Audio | Cell Phones | Office Electronics | Musical Instruments | New Arrivals | Trade-In

‹ Back to search results for "presentation clicker"

Roll over image to zoom in

BEBONCOOL

BEBONCOOL RF 2.4GHz Wireless Presenter Remote Presentation USB Control PowerPoint PPT Clicker

4.2 out of 5 stars 1,230 customer reviews | 141 answered questions

Amazon's Choice for "presentation clicker"

Price: $14.99 prime

FREE Shipping on orders over $25—or get **FREE Two-Day Shipping** with Amazon Prime

In Stock.

Want it Monday, April 16? Order within ~~23 hrs~~ 23 mins and choose **Two-Day Shipping** at checkout. Details

Sold by BEBONCOOL and Fulfilled by Amazon. Gift-wrap available.

receiver; Wireless Technology: 2.4 GHz; 1 x

eas of your slides

room and interact with your audience

Numbers & Pages) etc; For MacBook on

04 option

uch keys easy to control slideshow;

Add to Cart

About pointer: hold down the button to keep the light on

Compare with similar items

New (1) from $14.99 & FREE shipping on orders over $25.00. Details

Share | 660+ Shares

Qty: 1

Add a Protection Plan:

2-Year Protection for $0.58

Add to Cart

Turn on 1-Click ordering for this browser

Deliver to oakland 38060

Add to List

Have one to sell? Sell on Amazon

Super easy to use with no need to be tech savvy

Red star tec wireless powerpoint and keynote presentation remote clicker

4.5 out of 5 stars 520

$24.99 prime

Ad feedback

Frequently bought together

Total price: $19.98

Add both to Cart

Add both to List

Roll over image to zoom in

BEBONCOOL

BEBONCOOL RF 2.4GHz Wireless Presenter Remote Presentation USB Control PowerPoint PPT Clicker

Price: $14.99 prime

FREE Shipping on orders over $25—or get **FREE Two-Day Shipping** with Amazon Prime

In Stock.

Want it Monday, April 16? Order within **23 hrs 23 mins** and choose **Two-Day Shipping** at checkout. Details

Sold by BEBONCOOL and Fulfilled by Amazon. Gift-wrap available.

- Environmentally friendly ABS plastics; Scientific ergonomic design; A plug-and-play wireless receiver; Wireless Technology: 2.4 GHz; 1 x AAABattery Not Included
- A bright red light laser pointer that's easy to see against most backgrounds, highlight key areas of your slides
- Wireless remote control distance range of up to 39-foot, so you can free to move around the room and interact with your audience
- Support options: Supports MS Word, Excel, PowerPoint, ACD See, website, iWork (Keynote & Numbers & Pages) etc; For MacBook on OS, plug the usb receiver into laptop, it will come out a box, and then you can select 101or104 option
- Buttons: light pointer, display of black screen, next, previous, full screen, on/off switch; One-touch keys easy to control slideshow; About pointer: hold down the button to keep the light on

Compare with similar items

New (1) from $14.99 & FREE shipping on orders over $25.00. Details

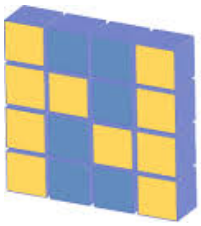
Frequently bought together

Total price: $19.98

Add both to Cart

Add both to List


DECISIONS
ARE
WORTH
DOLLARS



NumPy

matplotlib

pandas

$$y_i t = \beta' x_{it} + \mu_i + \epsilon_{it}$$

jupyter

VS

Data Science with Python and Friends

Douglas Starnes

CodeStock 2018

Polyglot ninja

Memphis, TN area

Co-director of MemPy

Data Science/Machine Learning, Mobile Apps, wannabe Game Designer

Pluralsight Author

@poweredbyaltnet

@poweredbyaltnet


So what does it take to be a data scientist?
@poweredbyaltnet

Ask 10 data scientists what they do

And you'll get at least 20 different answers.

@poweredbyaltnet

Data science is multidisciplinary

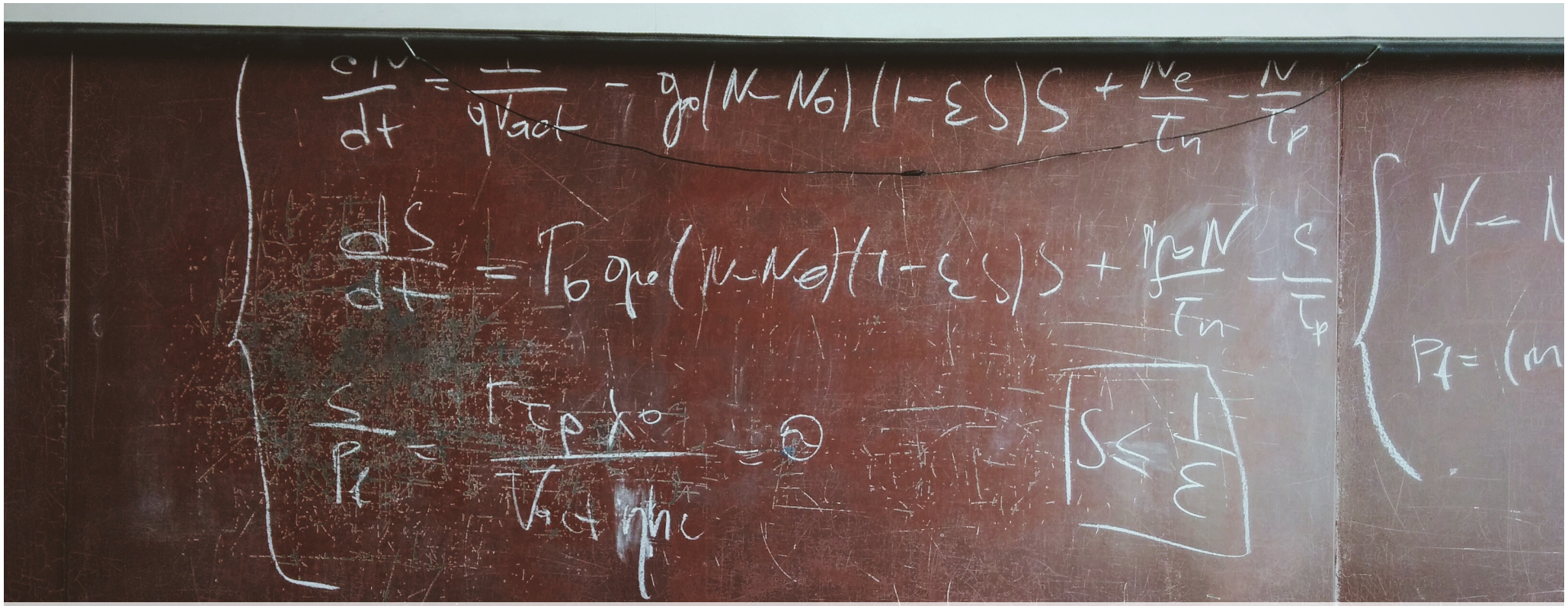
@poweredbyaltnet

Programming

@poweredbyaltnet

Programming

@poweredbyaltnet



Mathematics

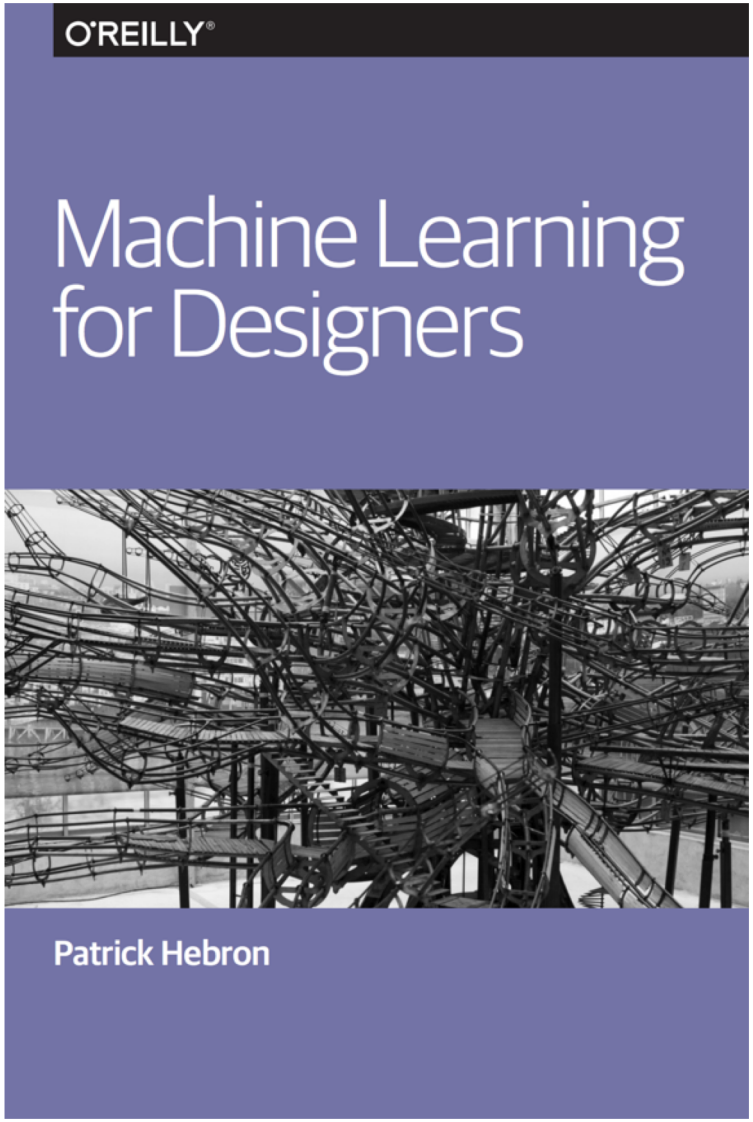
@poweredbyaltnet

Business

@poweredbyaltnet

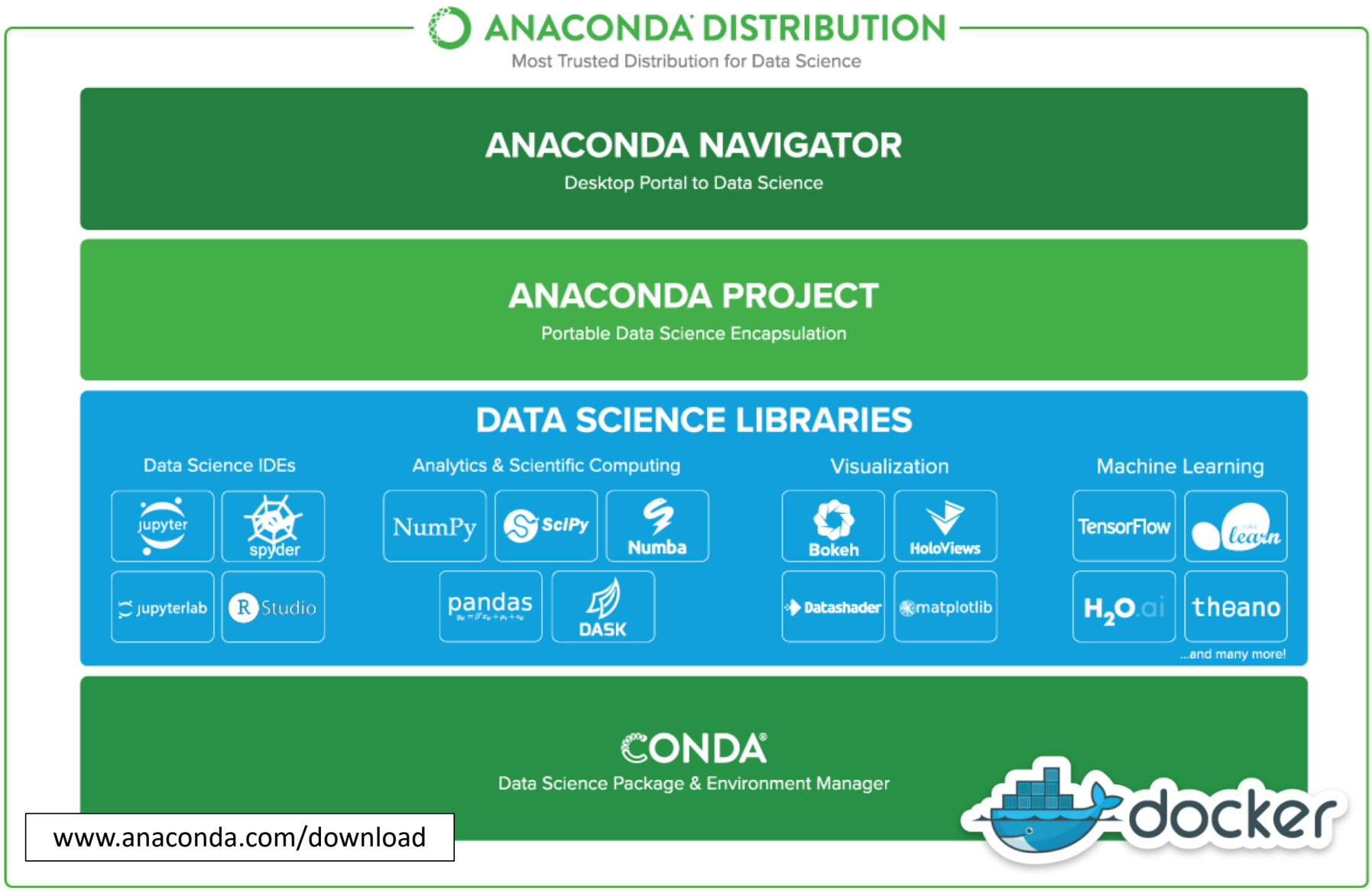
Art

@poweredbyaltnet

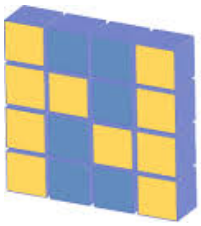

O'REILLY®
Machine Learning for Designers
Patrick Hebron


This is NOT Photoshopped!


Wait! Does this mean I'm not finished with school?



www.anaconda.com/download



NumPy

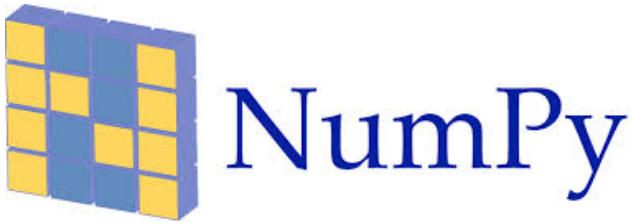
matplotlib

pandas

$$y_i t = \beta' x_{it} + \mu_i + \epsilon_{it}$$

jupyter

@poweredbyaltnet



“NumPy is the fundamental package for scientific computing with Python.”

Arrays

Random number generation

```
>>> r = range(10)
>>> r
range(0, 10)
>>> l = list(r)
>>> l
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> type(l)
list
>>> l[4]
4
>>> l[4:8]
[4, 5, 6, 7]
>>> l[-1]
9
>>> m = list(range(10, 20))
>>> m
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

```
>>> l + m
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
  11, 12, 13, 14, 15, 16, 17, 18, 19]
>>> l.extend(m)
>>> l
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
  11, 12, 13, 14, 15, 16, 17, 18, 19]
>>> l - m
TypeError: unsupported operand type(s)
  for -: 'list' and 'list')
>>> l + 1
TypeError: can only concatenate list
  (not "int") to list
>>> [v + 1 for v in l]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
  11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
```

```
>>> import numpy as np
>>> a = np.arange(10)
>>> a
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
>>> type(a)
numpy.ndarray
>>> a[4]
4
>>> a[4:8]
array([4, 5, 6, 7])
>>> a[-1]
9
>>> b = np.arange(10, 20)
>>> c = a + b
>>> c
array([10, 12, 14, 16, 18,
  20, 22, 24, 26, 28])
```

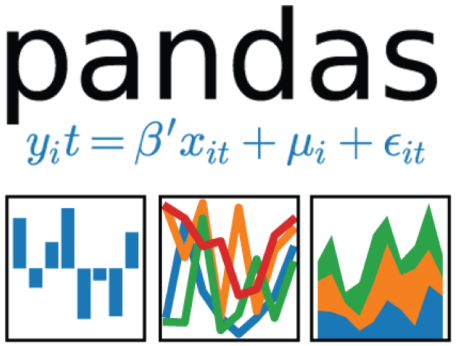
```
>>> c - b
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
>>> a - 1
array([-1,  0,  1,  2,  3,  4,  5,  6,  7,  8])
>>> a < 5
array([ True,  True,  True,  True,  True,
  False, False, False, False, False], dtype=bool)
>>> a[a < 5]
array([0, 1, 2, 3, 4])
```

```
>>> import numpy as np
>>> a = np.arange(20)
>>> a_hat = a.reshape(5, 4)
>>> a_hat
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11],
       [12, 13, 14, 15],
       [16, 17, 18, 19]])
>>> a_hat[1][2]
6
>>> a_hat[1,2]
6
>>> a_hat[:,2]
array([ 2,  6, 10, 14, 18])
```

```
>>> l = list(range(10))
>>> l[:4]
[0, 1, 2, 3]
>>> l[7:]
[7, 8, 9]
>>> l[:]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
>>> import numpy as np
>>> np.random.rand() # randn() uses normal distribution
0.5714800635809371
>>> np.random.rand(5, 4)
array([[ 0.37804532,  0.8301934 ,  0.67764257,  0.23334091],
       [ 0.95522908,  0.40048243,  0.66189479,  0.08995263],
       [ 0.86379148,  0.95826702,  0.38893546,  0.96141803],
       [ 0.49023199,  0.80043109,  0.63809899,  0.52142485],
       [ 0.01152943,  0.48649583,  0.21168751,  0.99074378]])
>>> np.random.rand(5, 4) * 10
array([[ 2.95986735,  0.24069298,  2.6357301 ,  5.4024343 ],
       [ 0.17023176,  2.49200206,  3.92632156,  5.72293003],
       [ 0.44536158,  3.07517235,  6.80317932,  0.20393922],
       [ 9.87761262,  3.24090831,  8.74508052,  4.14903998],
       [ 6.30249775,  9.62854077,  3.70046814,  4.69044164]])
>>> import random
>>> a = [[random.random() * 10 for _ in range(5)] for _ in range(4)]
>>> a.reshape(10, 2)
AttributeError: 'list' object has no attribute 'reshape'
```

```
>>> import numpy as np
>>> np.linspace(0, 10, 25)
array([  0.        ,   0.41666667,   0.83333333,   1.25      ,
         1.66666667,   2.08333333,   2.5       ,   2.91666667,
         3.33333333,   3.75      ,   4.16666667,   4.58333333,
         5.        ,   5.41666667,   5.83333333,   6.25      ,
         6.66666667,   7.08333333,   7.5       ,   7.91666667,
         8.33333333,   8.75      ,   9.16666667,   9.58333333,  10.        ])
>>> x = np.linspace.(0, 2 * np.pi, 361)
>>> y = np.sin(x)
>>> import math
>>> one_degree = 2 * math.pi / 361
>>> x = [i * one_degree for i in range(362)]
>>> y = [math.sin(xi) for xi in x]
>>> y.reshape(36, 10)
AttributeError: 'list' object has no attribute 'reshape'
```



"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."

Data frames

```
>>> import numpy as np
>>> import pandas as pd
>>> df = pd.read_csv('dow_jones_index.csv')
>>> df
```

```
quarter,stock,date,open,high,low,close,volume,percent_change_price,percent_change_volume_over_last_wk,previous_weeks_
volume,next_weeks_open,next_weeks_close,percent_change_next_weeks_price,days_to_next_dividend,percent_return_next_div
idend
1,AA,1/7/2011,$15.82,$16.72,$15.78,$16.42,239655616,3.79267,,,$16.71,$15.97,-4.42849,26,0.182704
1,AA,1/14/2011,$16.71,$16.71,$15.64,$15.97,242963398,-4.42849,1.380223028,239655616,$16.19,$15.79,-2.47066,19,0.18785
2
1,AA,1/21/2011,$16.19,$16.38,$15.60,$15.79,138428495,-2.47066,-43.02495926,242963398,$15.87,$16.13,1.63831,12,0.18999
4
1,AA,1/28/2011,$15.87,$16.63,$15.82,$16.13,151379173,1.63831,9.355500109,138428495,$16.18,$17.14,5.93325,5,0.185989
1,AA,2/4/2011,$16.18,$17.39,$16.18,$17.14,154387761,5.93325,1.987451735,151379173,$17.33,$17.37,0.230814,97,0.175029
1,AA,2/11/2011,$17.33,$17.48,$16.97,$17.37,114691279,0.230814,-25.71219489,154387761,$17.39,$17.28,-0.632547,90,0.172
712
1,AA,2/18/2011,$17.39,$17.68,$17.28,$17.28,80023895,-0.632547,-30.22669579,114691279,$16.98,$16.68,-1.76678,83,0.1736
11
1,AA,2/25/2011,$16.98,$17.15,$15.96,$16.68,132981863,-1.76678,66.17769355,80023895,$16.81,$16.58,-1.36823,76,0.179856
1,AA,3/4/2011,$16.81,$16.94,$16.13,$16.58,109493077,-1.36823,-17.66315005,132981863,$16.58,$16.03,-3.31725,69,0.18094
1
```

```
>>> df.columns
Index(['quarter', 'stock', 'date', 'open', 'high', 'low', 'close', 'volume',
       'percent_change_price', 'percent_change_volume_over_last_wk', 'previous_weeks_volume',
       'next_weeks_open', 'next_weeks_close','percent_change_next_weeks_price',
       'days_to_next_dividend','percent_return_next_dividend'], dtype='object')
```

```
>>> df['stock']
>>> df.columns[1:8]
>>> v = df.loc[:, df.columns[1:8]].copy()
>>> v
 stock       date    open    high     low   close     volume
0      AA   1/7/2011  $15.82  $16.72  $15.78  $16.42  239655616
1      AA  1/14/2011  $16.71  $16.71  $15.64  $15.97  242963398
2      AA  1/21/2011  $16.19  $16.38  $15.60  $15.79  138428495
3      AA  1/28/2011  $15.87  $16.63  $15.82  $16.13  151379173
4      AA   2/4/2011  $16.18  $17.39  $16.18  $17.14  154387761
5      AA  2/11/2011  $17.33  $17.48  $16.97  $17.37  114691279
6      AA  2/18/2011  $17.39  $17.68  $17.28  $17.28   80023895
7      AA  2/25/2011  $16.98  $17.15  $15.96  $16.68  132981863
8      AA   3/4/2011  $16.81  $16.94  $16.13  $16.58  109493077
9      AA  3/11/2011  $16.58  $16.75  $15.42  $16.03  114332562
10     AA  3/18/2011  $15.95  $16.33  $15.43  $16.11  130374108
>>> v.volume.max()
1453438639
>>> v.close[0]
'$16.42'
```

```
>>> for column in v.columns[2:6]:
        v.loc[:, column] = v.loc[:, column].apply(lambda x: float(x[1:]), 1)

>>> v
```

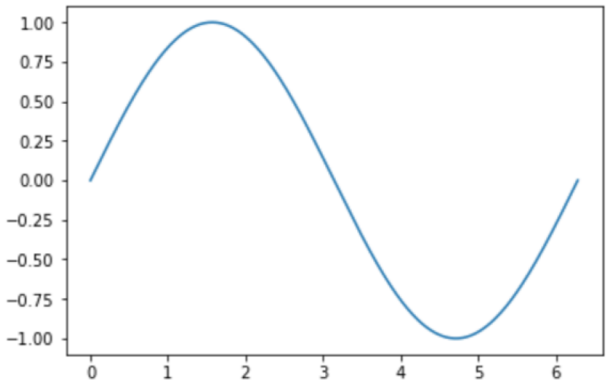
	stock	date	open	high	low	close	volume
0	AA	1/7/2011	15.82	16.72	15.78	16.42	239655616
1	AA	1/14/2011	16.71	16.71	15.64	15.97	242963398
2	AA	1/21/2011	16.19	16.38	15.60	15.79	138428495
3	AA	1/28/2011	15.87	16.63	15.82	16.13	151379173
4	AA	2/4/2011	16.18	17.39	16.18	17.14	154387761
5	AA	2/11/2011	17.33	17.48	16.97	17.37	114691279
6	AA	2/18/2011	17.39	17.68	17.28	17.28	80023895
7	AA	2/25/2011	16.98	17.15	15.96	16.68	132981863
8	AA	3/4/2011	16.81	16.94	16.13	16.58	109493077
9	AA	3/11/2011	16.58	16.75	15.42	16.03	114332562

```
>>> v[v.stock == 'DIS']

>>> v[v.stock == 'DIS']['close']

>>> close_index = v[v.stock == 'DIS']['close'].idxmax()

>>> v.loc[close_index, 'volume']

53096584
```

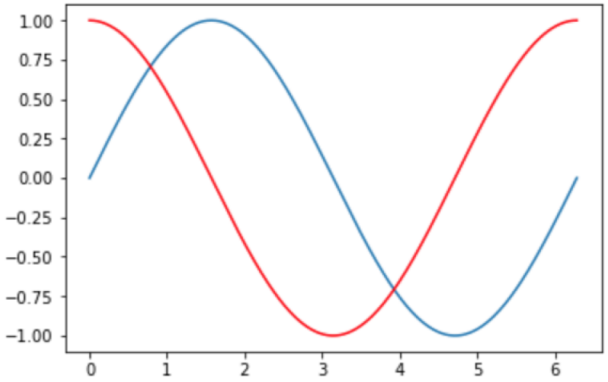
“Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.”

Visualizations

```
>>> import numpy as np
>>> x = np.linspace(0, 2 * np.pi, 361)
>>> y = np.sin(x)
>>> import matplotlib.pyplot as plt
>>> plt.plot(x, y)
```

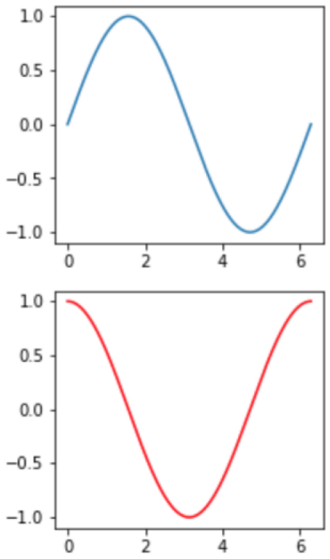



```
>>> y2 = np.cos(x)
>>> plt.plot(x, y)
    plt.plot(x, y2, color='r')
```

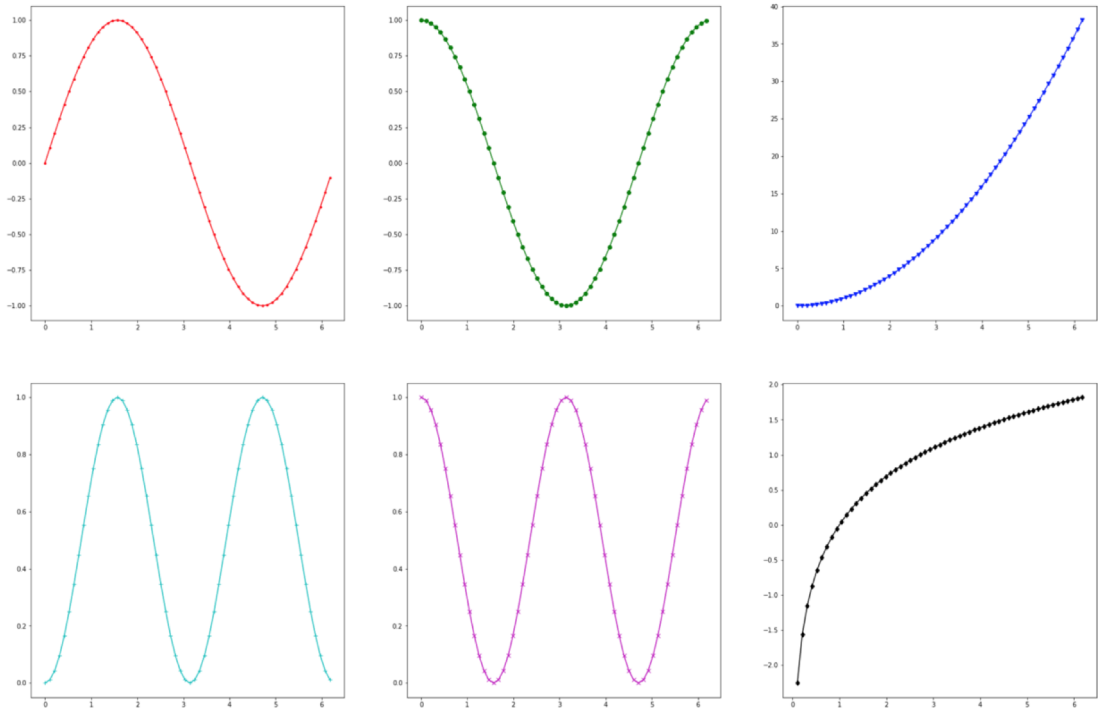


@poweredbyaltnet

```
>>> plt.figure(figsize=(3, 6)) # height is 2x the width
    plt.subplot(2, 1, 1) # 2 rows, 1 column, position 1
    plt.plot(x, y)
    plt.subplot(2, 1, 2) # position 2
    plt.plot(x, y2, color='r')
```

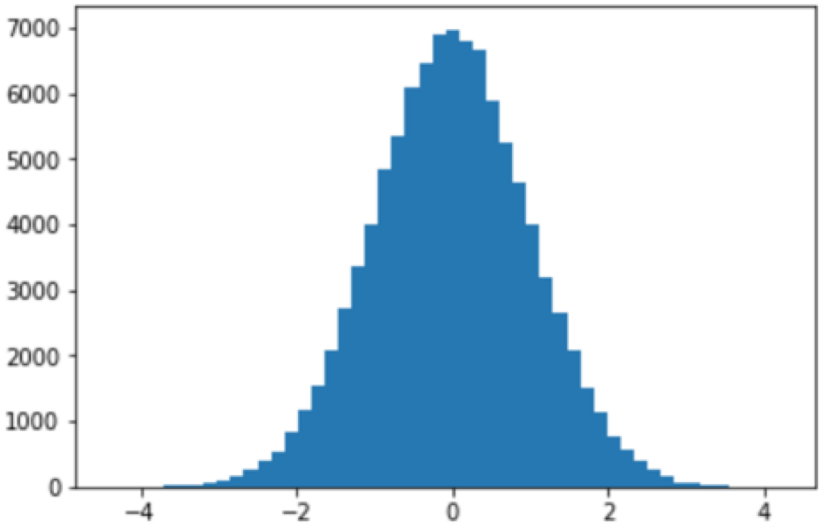


```
fns = [np.sin, np.cos, lambda x: x ** 2, lambda x: np.sin(x) ** 2, lambda x: np.cos(x) ** 2, np.log]
colors = list('rgbcmk')
markers = list('.ov+xd')
data = zip(fns, colors, markers)

plt.figure(figsize=(30, 20))
for i, (fn, color, marker) in enumerate(data):
    plt.subplot(2, 3, i + 1) # 1-3 on first row, 4-6 on second
    plt.plot(x[np.arange(0, 360, 6)], fn(x[np.arange(0, 360, 6)]), color=color, marker=marker)
```

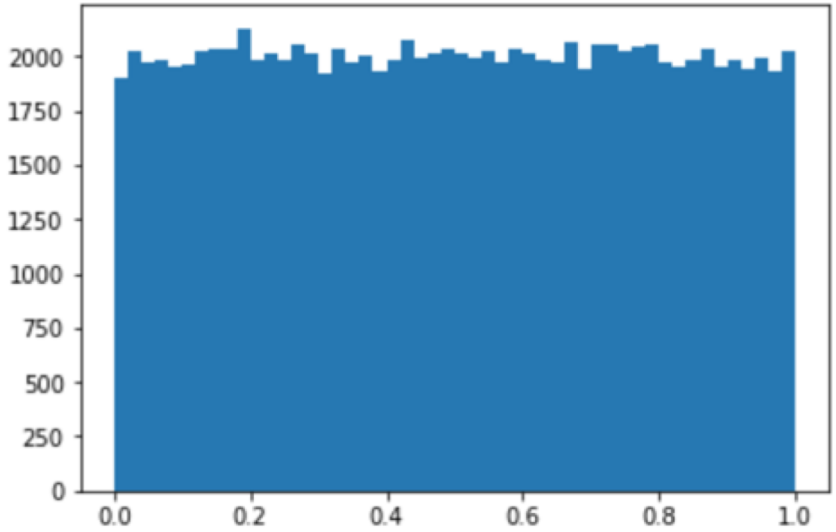


@poweredbyaltnet

```
x = np.random.randn(100000)
plt.hist(x, bins=50)
```

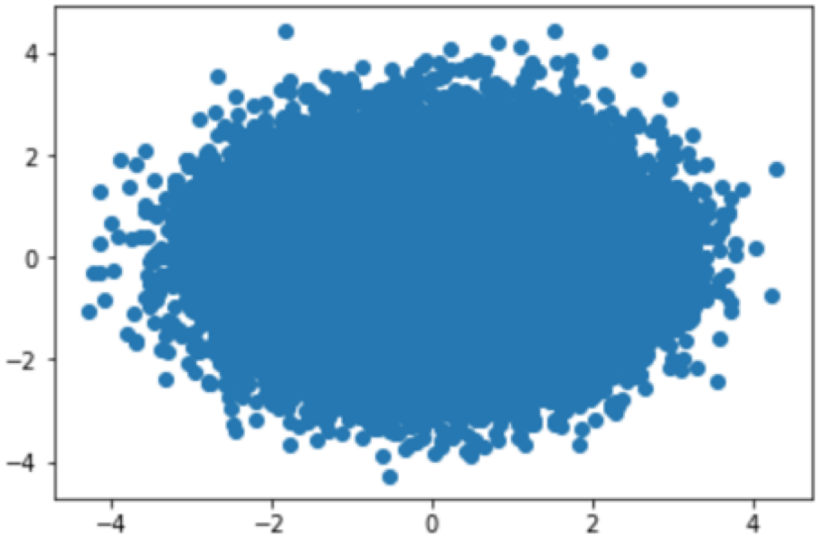



```
x = np.random.rand(100000)
plt.hist(x, bins=50)
```

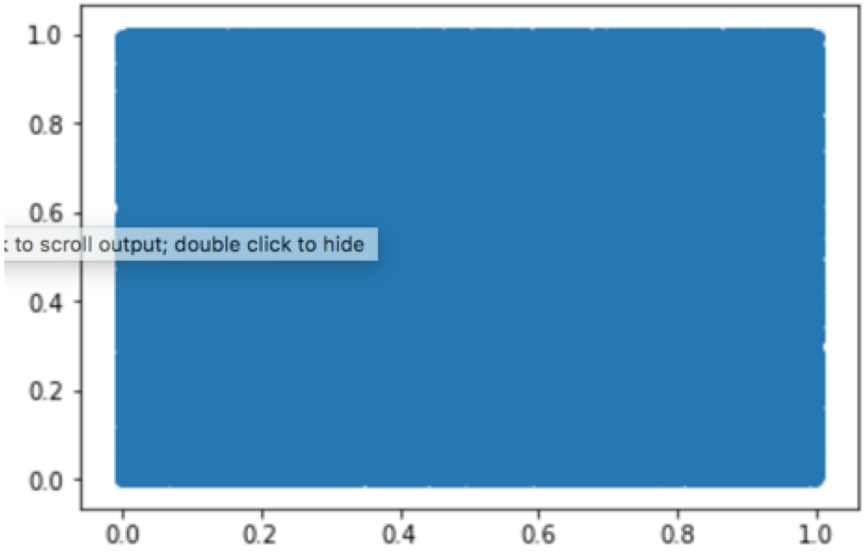


@poweredbyaltnet

```
x = np.random.randn(100000)
y = np.random.randn(100000)
plt.scatter(x, y)
```



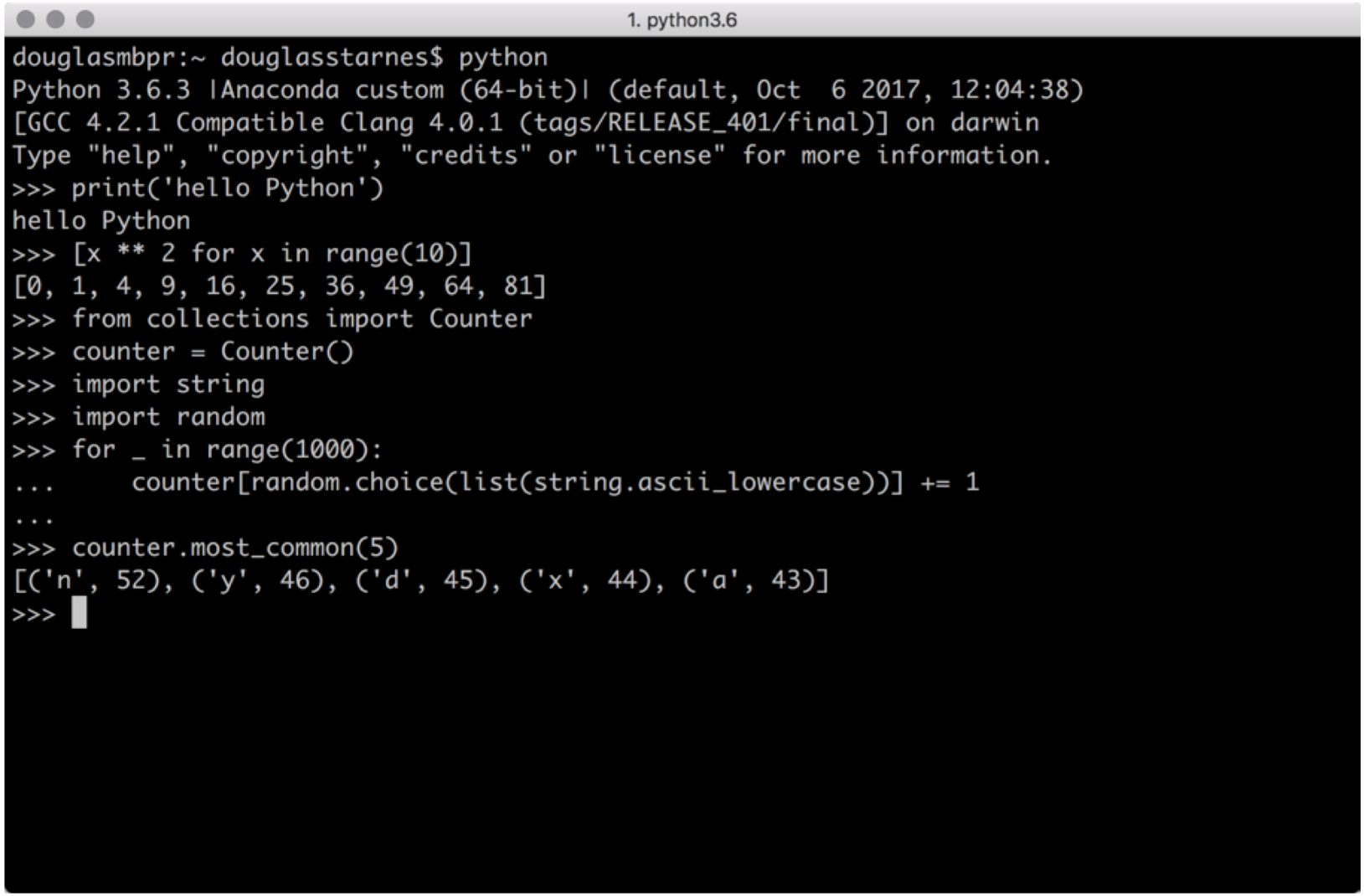

```
x = np.random.rand(100000)
y = np.random.rand(100000)
plt.scatter(x, y)
```

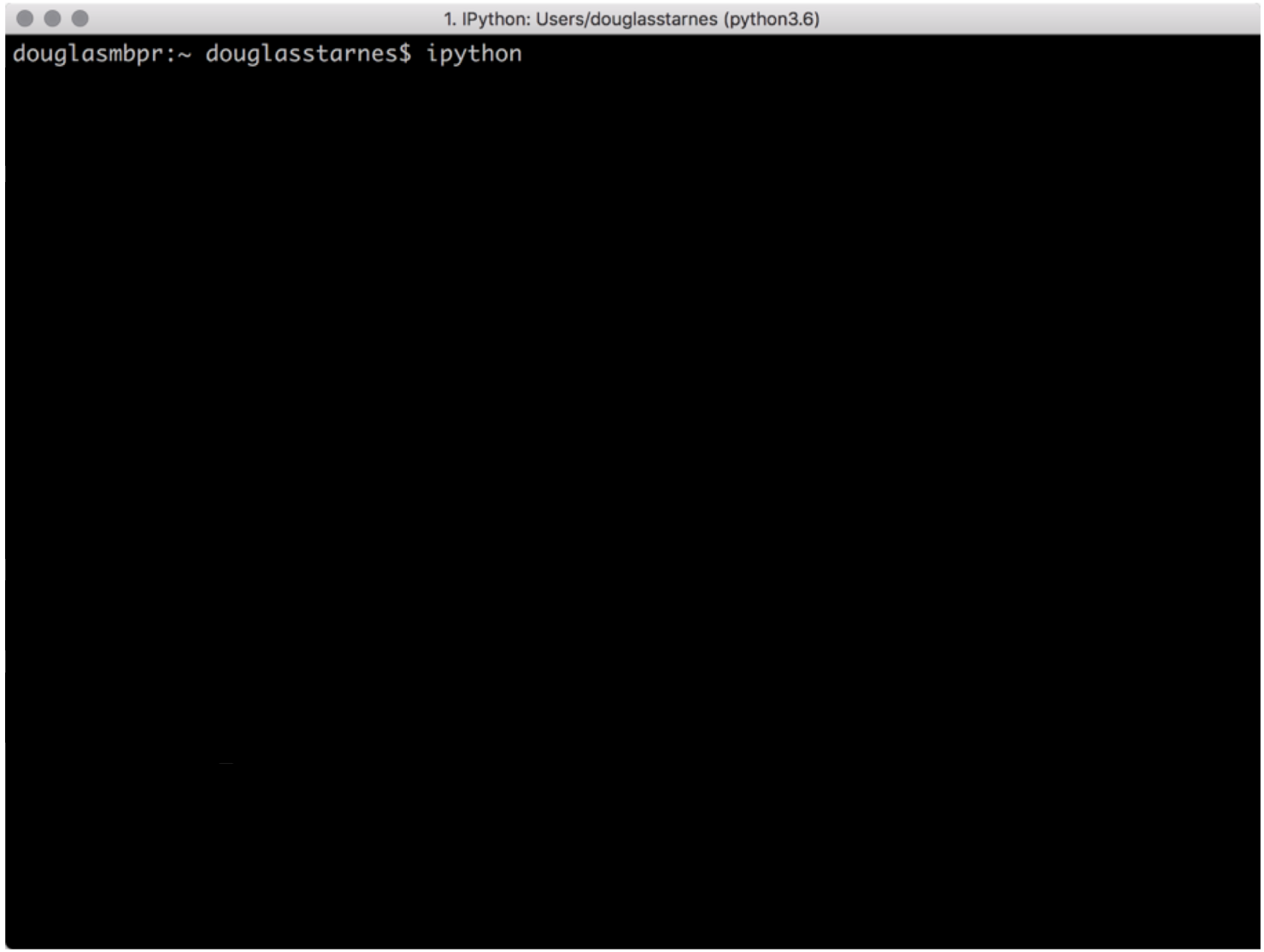


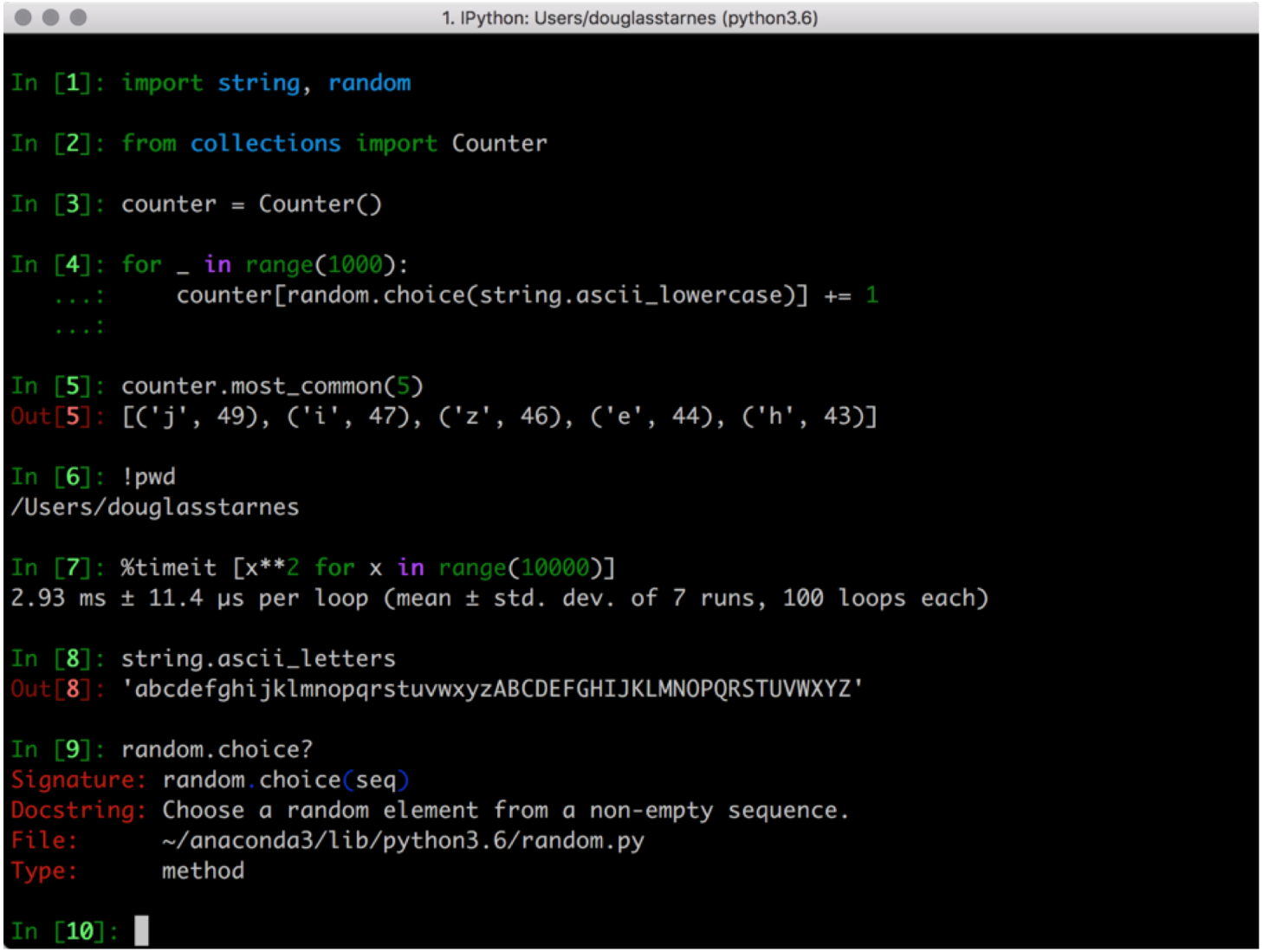
@poweredbyaltnet

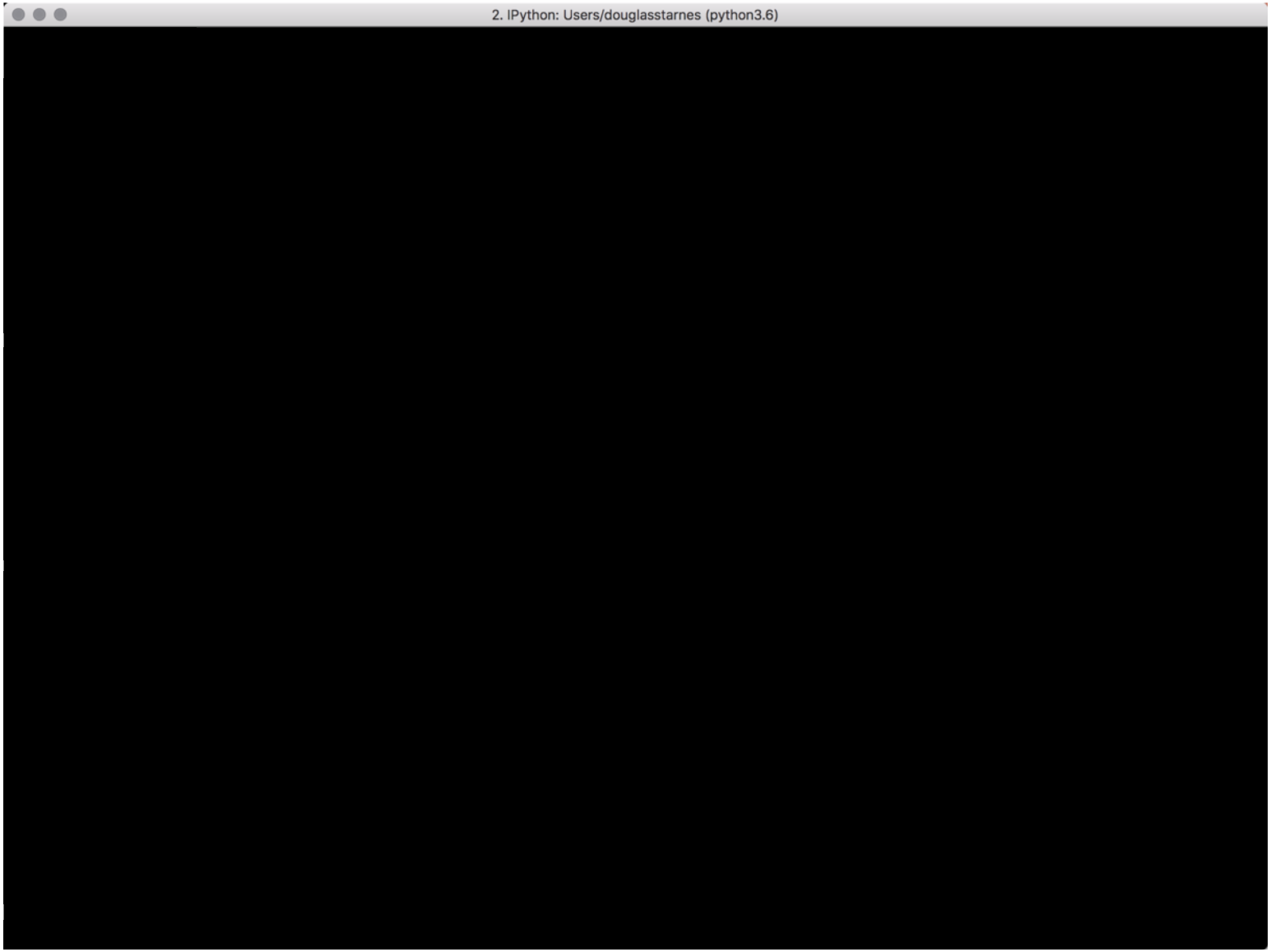
Project Jupyter exists to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.

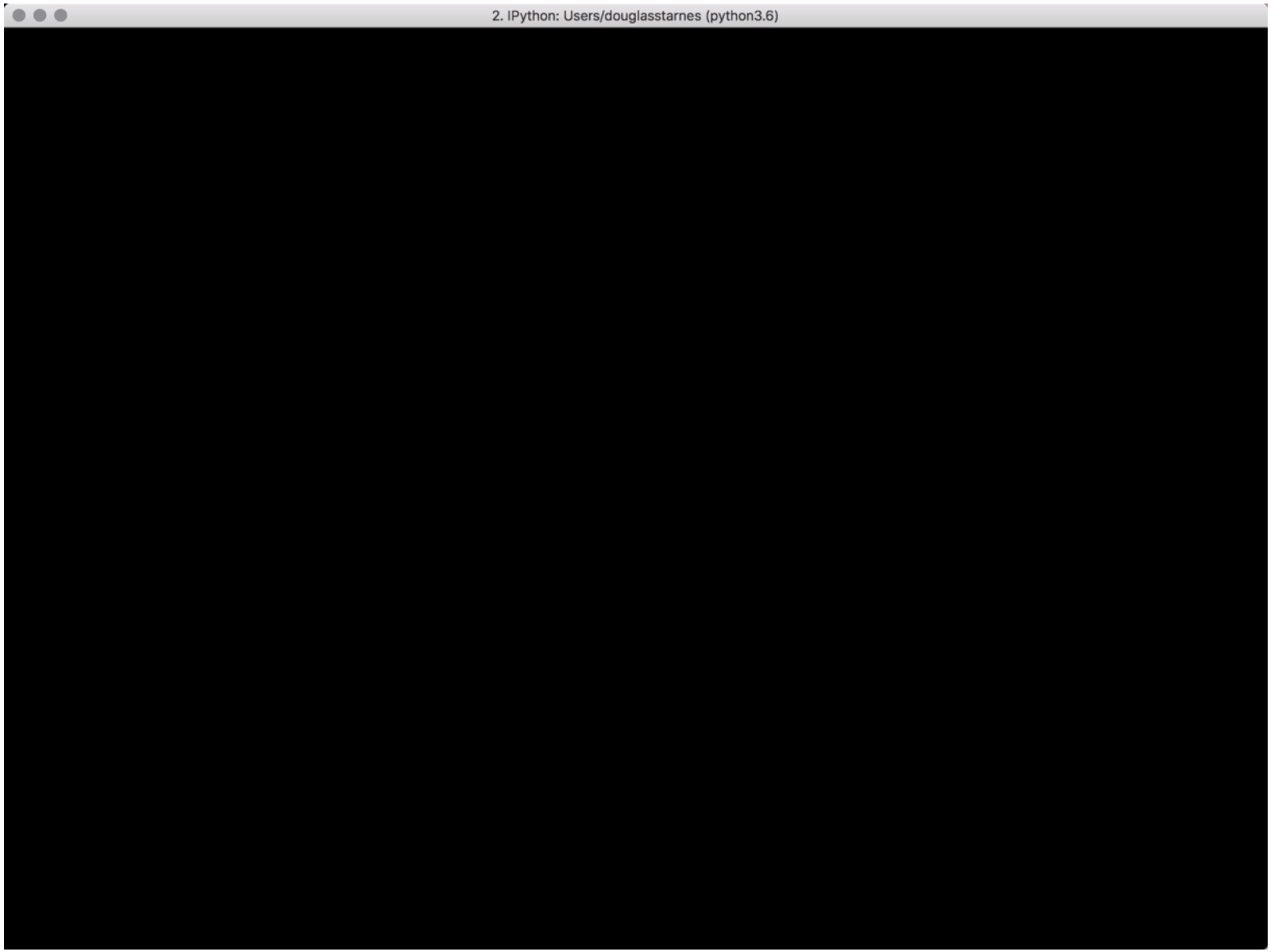
Jupyter notebook


1. python3.6
douglasmbpr:~ douglasstarnes$ python
Python 3.6.3 |Anaconda custom (64-bit)| (default, Oct 6 2017, 12:04:38)
[GCC 4.2.1 Compatible Clang 4.0.1 (tags/RELEASE_401/final)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print('hello Python')
hello Python
>>> [x ** 2 for x in range(10)]
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
>>> from collections import Counter
>>> counter = Counter()
>>> import string
>>> import random
>>> for _ in range(1000):
...   counter[random.choice(list(string.ascii_lowercase))] += 1
...
>>> counter.most_common(5)
[('n', 52), ('y', 46), ('d', 45), ('x', 44), ('a', 43)]
>>>


1. IPython: Users/douglasstarnes (python3.6)
douglasmbpr:~ douglasstarnes$ ipython


1. IPython: Users/douglasstarnes (python3.6)
In [1]: import string, random
In [2]: from collections import Counter
In [3]: counter = Counter()
In [4]: for _ in range(1000):
...: counter[random.choice(string.ascii_lowercase)] += 1
...:
In [5]: counter.most_common(5)
Out[5]: [('j', 49), ('i', 47), ('z', 46), ('e', 44), ('h', 43)]
In [6]: !pwd
/Users/douglasstarnes
In [7]: %timeit [x**2 for x in range(10000)]
2.93 ms ± 11.4 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)
In [8]: string.ascii_letters
Out[8]: 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'
In [9]: random.choice?
Signature: random.choice(seq)
Docstring: Choose a random element from a non-empty sequence.
File: ~/anaconda3/lib/python3.6/random.py
Type: method
In [10]:


2. IPython: Users/douglasstarnes (python3.6)


2. IPython: Users/douglasstarnes (python3.6)

jupyter Widget Demo Last Checkpoint: 03/19/2018 (unsaved changes)

Azure Notebooks | My Libraries | MemPy0318

File Edit View Insert Cell Kernel Data Widgets Help | Not Trusted | Python 3

Run | Code | Enter/Exit RISE Slideshow

The Final Product

This is the complete dashboard.

- Set the endpoints of the slider to the date range
- Check the box to show volume as well as closing price
- Select the stock symbol in the dropdown box

In [33]:

```
date_slider = widgets.SelectionRangeSlider(options=list(stock_dict['AA'].date),
    index=(0, 24),
    description='Dates',
    layout=widgets.Layout(width='500px'))
show_vol2 = widgets.Checkbox(description='Show Volume')
dd_select3 = widgets.Dropdown(options=sorted(list(stock_dict.keys())))
button = widgets.Button(description='Show Graph')

def button_click(b):
    stock_row = stock_dict[dd_select3.value]
    start_index = date_slider.index[0]
    end_index = date_slider.index[1]
    stock_row = stock_row[start_index:(end_index+1)]

    y = stock_row.close
    x = list(range(len(y)))

    plt.xticks(np.arange(len(x)), stock_row.date, rotation='vertical')

    if show_vol2.value == False:
        plt.plot(x, y, color='k')
    else:
        y2 = stock_row.volume
        plt.bar(x, y2, color='#cccccc')
        ax2 = plt.twinx()
        ax2.plot(x, y, color='k')

button.on_click(button_click)

display(date_slider)
display(show_vol2)
display(dd_select3)
display(button)
```

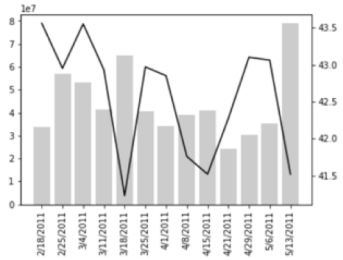
Syntax highlighting
Automatic indentation

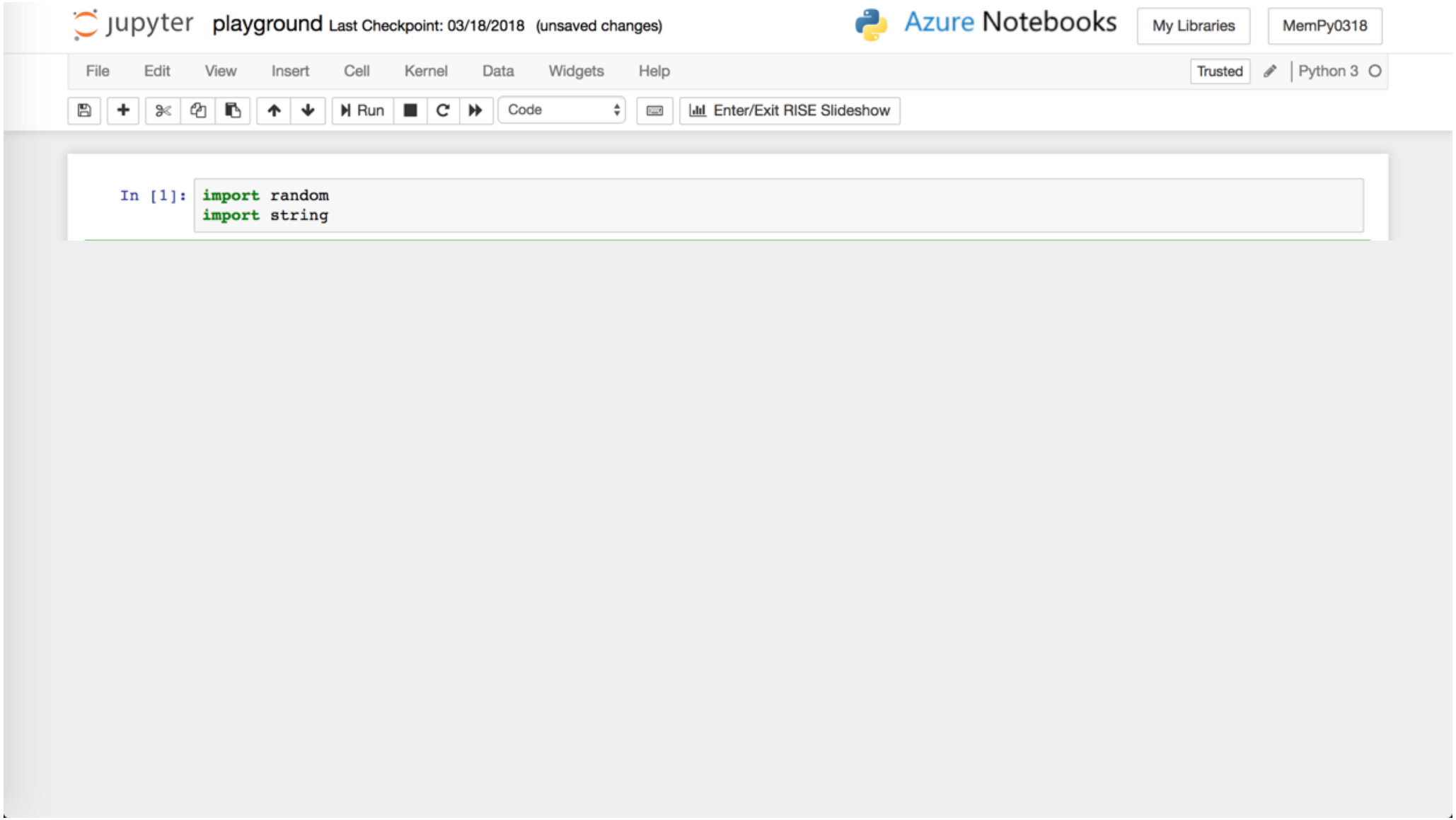
Dates 2/18/2011-5/13/2011

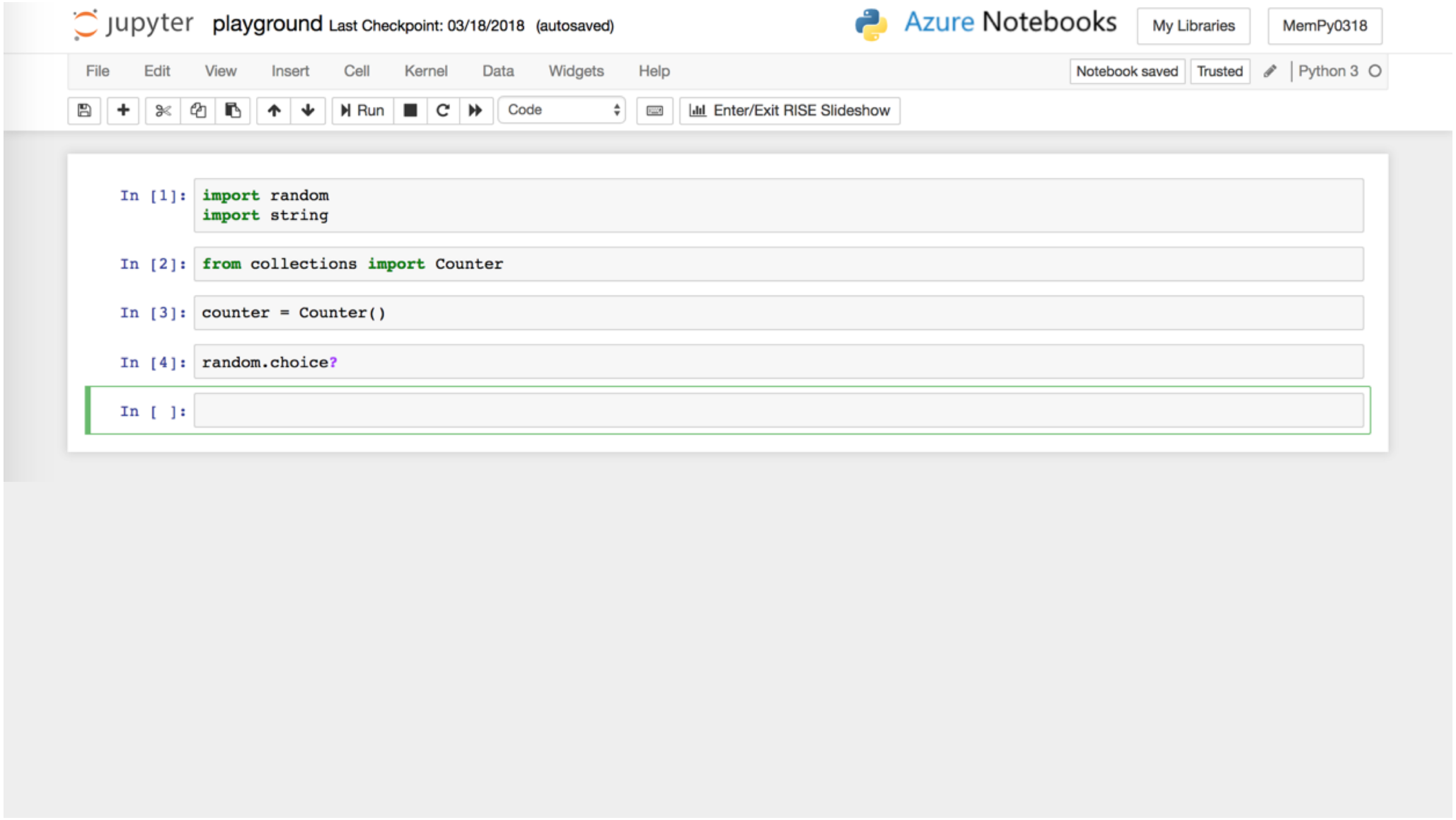
Show Volume

DIS

Show Graph




jupyter playground Last Checkpoint: 03/18/2018 (unsaved changes)
Azure Notebooks
My Libraries
MemPy0318
File Edit View Insert Cell Kernel Data Widgets Help
Trusted
Python 3
Run
Code
Enter/Exit RISE Slideshow
In [1]:
import random
import string


jupyter playground Last Checkpoint: 03/18/2018 (autosaved)
Azure Notebooks
My Libraries
MemPy0318
File Edit View Insert Cell Kernel Data Widgets Help
Notebook saved
Trusted
Python 3
Run
Code
Enter/Exit RISE Slideshow
In [1]:
import random
import string
In [2]:
from collections import Counter
In [3]:
counter = Counter()
In [4]:
random.choice?
In [ ]:

jupyter playground Last Checkpoint: 03/18/2018 (unsaved changes)

Azure Notebooks My Libraries MemPy0318

File Edit View Insert Cell Kernel Data Widgets Help Trusted Python 3

Run Code Enter/Exit RISE Slideshow

In [1]:
```
import random
import string
```

In [2]:
```
from collections import Counter
```

In [3]:
```
counter = Counter()
```

In [4]:
```
random.choice?
```

In [5]:
```
for _ in range(1000):
    counter[random.choice(string.ascii_lowercase)] += 1
```

In [6]:
```
ordered_keys = [i[0] for i in counter.most_common(10)]
ordered_values = [i[1] for i in counter.most_common(10)]
```

In [ ]:

jupyter Widget Demo Last Checkpoint: 03/19/2018 (unsaved changes)

Azure Notebooks | My Libraries | MemPy0318

File Edit View Insert Cell Kernel Data Widgets Help | Not Trusted | Python 3

Run | Code | Enter/Exit RISE Slideshow

The Final Product

This is the complete dashboard.

- Set the endpoints of the slider to the date range
- Check the box to show volume as well as closing price
- Select the stock symbol in the dropdown box

In [33]:

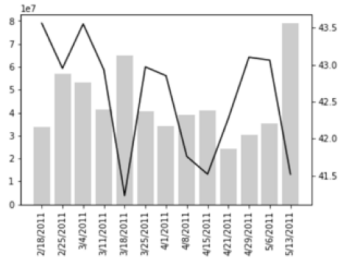
```
date_slider = widgets.SelectionRangeSlider(options=list(stock_dict['AA'].date),
    index=(0, 24),
    description='Dates',
    layout=widgets.Layout(width='500px'))
show_vol2 = widgets.Checkbox(description='Show Volume')
dd_select3 = widgets.Dropdown(options=sorted(list(stock_dict.keys())))
button = widgets.Button(description='Show Graph')

def button_click(b):
    stock_row = stock_dict[dd_select3.value]
    start_index = date_slider.index[0]
    end_index = date_slider.index[1]
    stock_row = stock_row[start_index:(end_index+1)]

    y = stock_row.close
    x = list(range(len(y)))

    plt.xticks(np.arange(len(x)), stock_row.date, rotation='vertical')

    if show_vol2.value == False:
        plt.plot(x, y, color='k')
    else:
        y2 = stock_row.volume
        plt.bar(x, y2, color='#cccccc')
        ax2 = plt.twinx()
        ax2.plot(x, y, color='k')

button.on_click(button_click)

display(date_slider)
display(show_vol2)
display(dd_select3)
display(button)
```

Dates 2/18/2011-5/13/2011

Show Volume

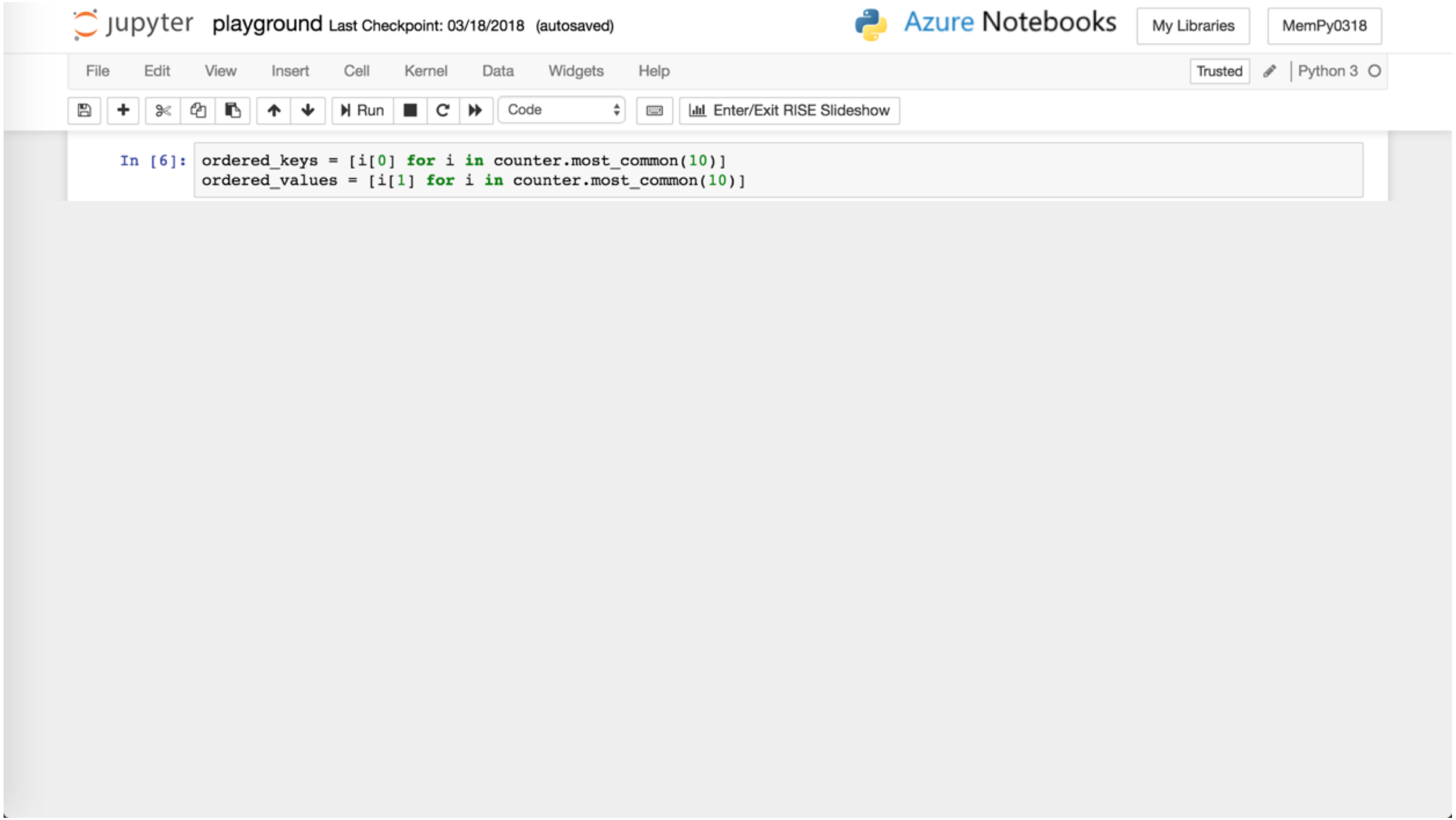
DIS

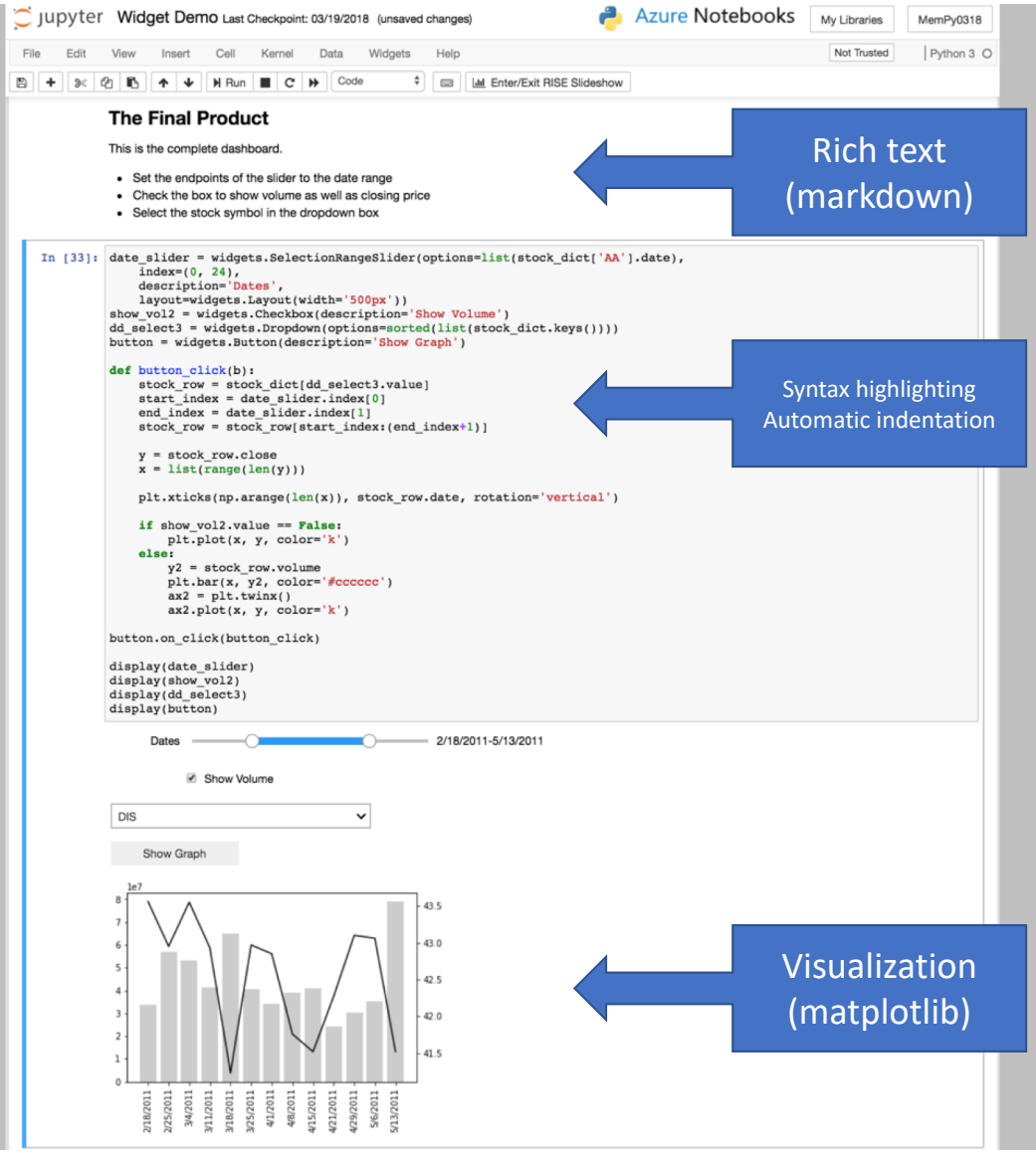
Show Graph



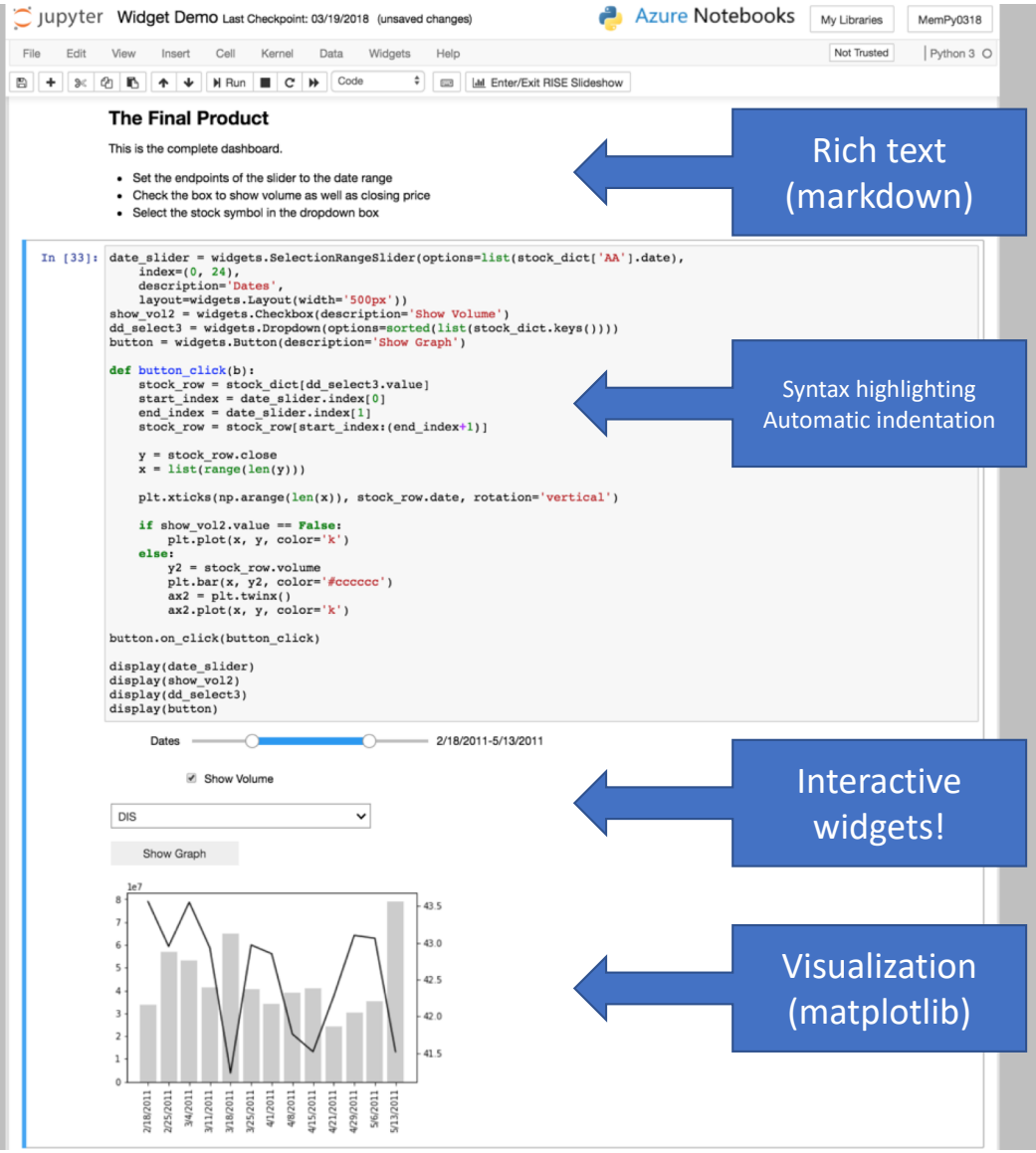
Syntax highlighting
Automatic indentation

Visualization
(matplotlib)

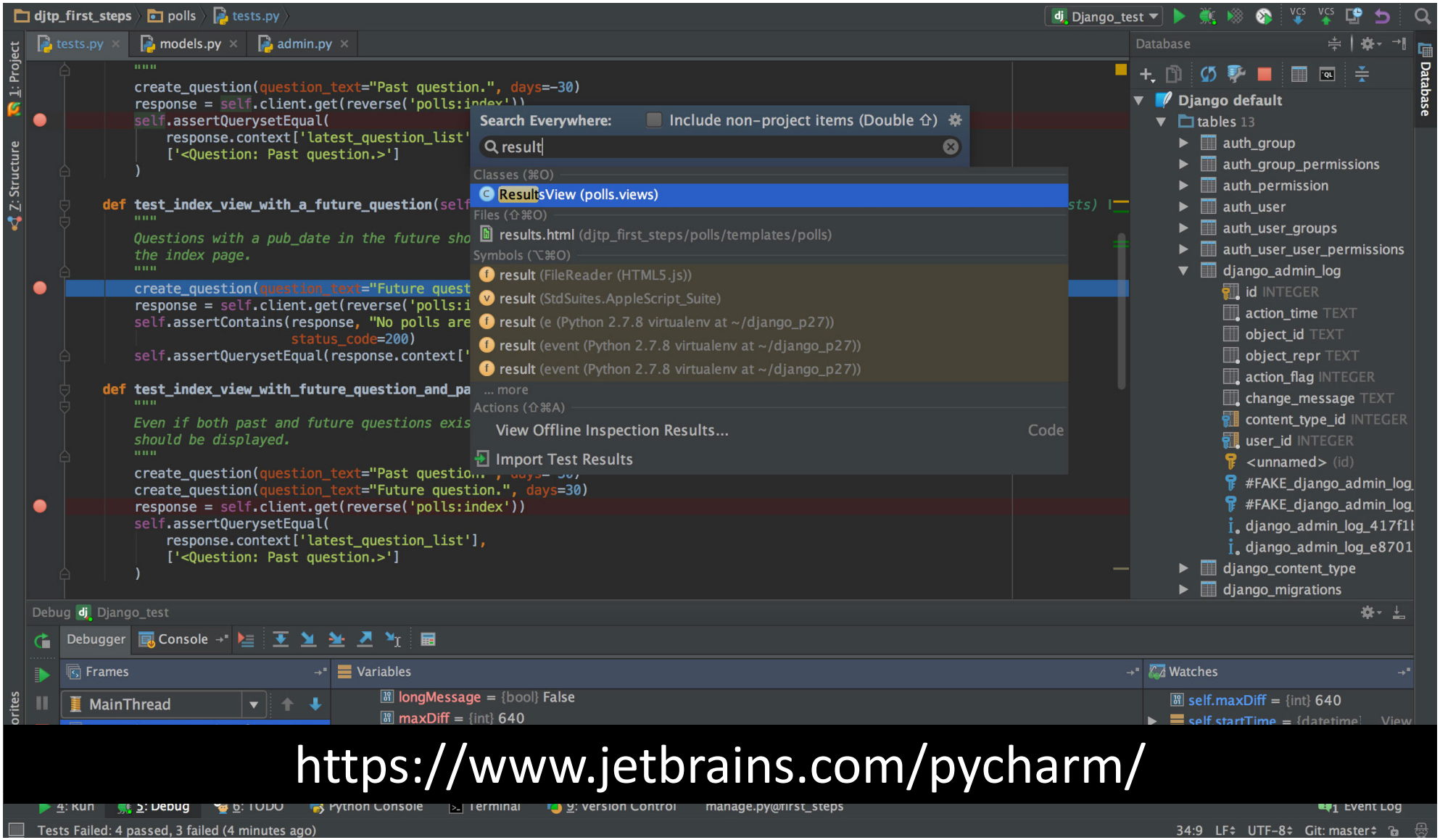

jupyter playground Last Checkpoint: 03/18/2018 (autosaved)
Azure Notebooks
My Libraries
MemPy0318
File Edit View Insert Cell Kernel Data Widgets Help
Trusted
Python 3
Run
Code
Enter/Exit RISE Slideshow
In [6]:
ordered_keys = [i[0] for i in counter.most_common(10)]
ordered_values = [i[1] for i in counter.most_common(10)]


jupyter Widget Demo Last Checkpoint: 03/19/2018 (unsaved changes)
Azure Notebooks
My Libraries
MemPy0318
File Edit View Insert Cell Kernel Data Widgets Help
Not Trusted
Python 3
Run
Code
Enter/Exit RISE Slideshow
The Final Product
This is the complete dashboard.
Set the endpoints of the slider to the date range
Check the box to show volume as well as closing price
Select the stock symbol in the dropdown box
In [33]:
date_slider = widgets.SelectionRangeSlider(options=list(stock_dict['AA'].date),
index=(0, 24),
description='Dates',
layout=widgets.Layout(width='500px'))
show_vol2 = widgets.Checkbox(description='Show Volume')
dd_select3 = widgets.Dropdown(options=sorted(list(stock_dict.keys())))
button = widgets.Button(description='Show Graph')
def button_click(b):
stock_row = stock_dict[dd_select3.value]
start_index = date_slider.index[0]
end_index = date_slider.index[1]
stock_row = stock_row[start_index:(end_index+1)]
y = stock_row.close
x = list(range(len(y)))
plt.xticks(np.arange(len(x)), stock_row.date, rotation='vertical')
if show_vol2.value == False:
plt.plot(x, y, color='k')
else:
y2 = stock_row.volume
plt.bar(x, y2, color='#cccccc')
ax2 = plt.twinx()
ax2.plot(x, y, color='k')
button.on_click(button_click)
display(date_slider)
display(show_vol2)
display(dd_select3)
display(button)
Dates
2/18/2011-5/13/2011
Show Volume
DIS
Show Graph
1e7
2/18/2011
2/25/2011
3/4/2011
3/11/2011
3/18/2011
3/25/2011
4/1/2011
4/8/2011
4/15/2011
4/21/2011
4/29/2011
5/6/2011
5/13/2011
Rich text (markdown)
Syntax highlighting
Automatic indentation
Visualization (matplotlib)

```
## Heading 2
**Bold**
*Italic*

1. Ordered
2. List
3. Items

#### And

* Unordered
* List
* Items

#### And
*  *Formatted*
*  **List**
* Items
```

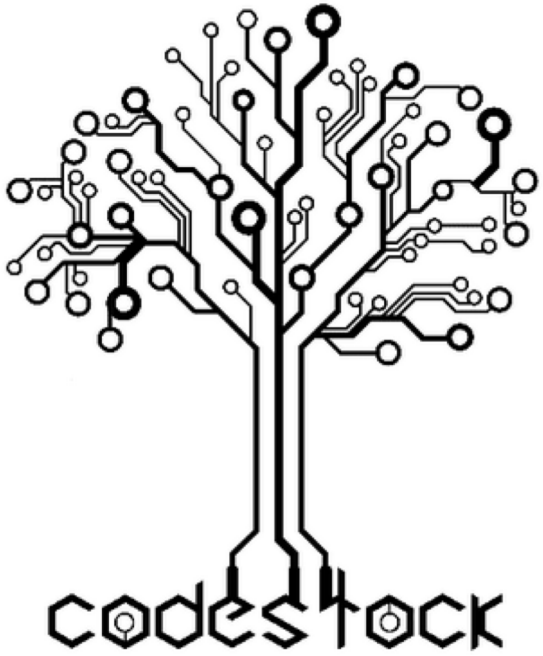

jupyter Widget Demo Last Checkpoint: 03/19/2018 (unsaved changes)
Azure Notebooks
My Libraries
MemPy0318
File Edit View Insert Cell Kernel Data Widgets Help
Not Trusted
Python 3
Run
Code
Enter/Exit RISE Slideshow
The Final Product
This is the complete dashboard.
Set the endpoints of the slider to the date range
Check the box to show volume as well as closing price
Select the stock symbol in the dropdown box
In [33]:
date_slider = widgets.SelectionRangeSlider(options=list(stock_dict['AA'].date),
index=(0, 24),
description='Dates',
layout=widgets.Layout(width='500px'))
show_vol2 = widgets.Checkbox(description='Show Volume')
dd_select3 = widgets.Dropdown(options=sorted(list(stock_dict.keys())))
button = widgets.Button(description='Show Graph')
def button_click(b):
stock_row = stock_dict[dd_select3.value]
start_index = date_slider.index[0]
end_index = date_slider.index[1]
stock_row = stock_row[start_index:(end_index+1)]
y = stock_row.close
x = list(range(len(y)))
plt.xticks(np.arange(len(x)), stock_row.date, rotation='vertical')
if show_vol2.value == False:
plt.plot(x, y, color='k')
else:
y2 = stock_row.volume
plt.bar(x, y2, color='#cccccc')
ax2 = plt.twinx()
ax2.plot(x, y, color='k')
button.on_click(button_click)
display(date_slider)
display(show_vol2)
display(dd_select3)
display(button)
Dates
2/18/2011-5/13/2011
Show Volume
DIS
Show Graph
1e7
2/18/2011
2/25/2011
3/4/2011
3/11/2011
3/18/2011
3/25/2011
4/1/2011
4/8/2011
4/15/2011
4/21/2011
4/29/2011
5/6/2011
5/13/2011
Rich text (markdown)
Syntax highlighting
Automatic indentation
Interactive widgets!
Visualization (matplotlib)

@poweredbyaltnet



https://www.jetbrains.com/pycharm/

I want to thank the organizers, sponsors, staff and volunteers of CodeStock 2018.



@poweredbyaltnet

Don’t forget to fill out the feedback forms for a chance to win a Amazon gift card at the end of the day.

(must be present to win)

@poweredbyaltnet

PLURALSIGHT

Getting Started with Jupyter Notebook and Python

Scan the code or visit the link to get a 10-day FREE trial!

https://bit.ly/jupyter-notebook

@poweredbyaltnet

Thank You!

@poweredbyaltnet

douglas@douglasstarnes.com

http://douglasstarnes.com

https://github.com/douglasstarnes/codestock2018