Data Processing Using Python

Advanced Data Processing and Visualization of Python

ZHANG Dazhuang

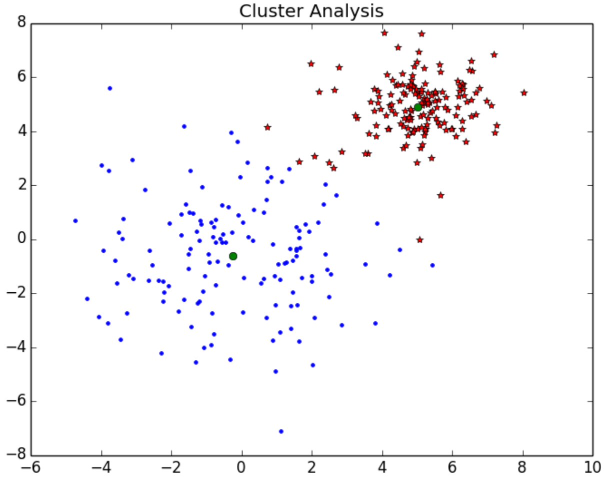
Department of Computer Science and Technology

Department of University Basic Computer Teaching

Data Processing Using Python

1 CLUSTER ANALYSIS

Cluster



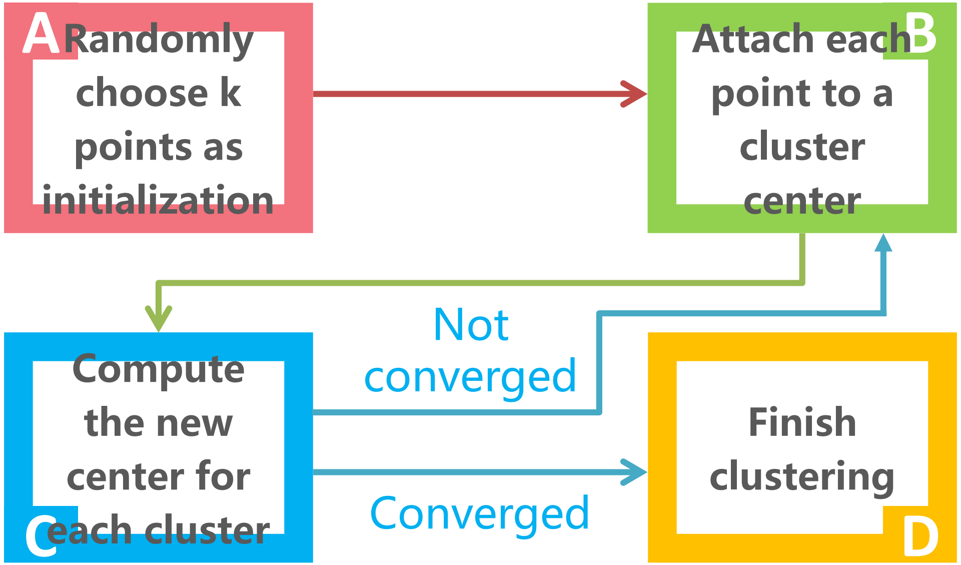
- **cluster analysis**

grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters).

  – Feature
    - Based on similarity
    - Have multiple cluster centers

K-MEANS

K-means algorithm uses k points in space as the centers to cluster all objects.


A
Randomly choose k points as initialization
B
Attach each point to a cluster center
C
Compute the new center for each cluster
Not converged
Converged
D
Finish clustering

A Daily Example

	Math	English	Python	Music
ming	88	64	96	85
MING	92	99	95	94
peng	91	87	99	95
PENG	78	99	97	81
meng	88	78	98	84
MENG	100	95	100	92

File

```python
# Filename: kmeansStu1.py
import numpy as np
from scipy.cluster.vq import vq, kmeans, whiten
list1 = [88.0, 74.0, 96.0, 85.0]
list2 = [92.0, 99.0, 95.0, 94.0]
list3 = [91.0, 87.0, 99.0, 95.0]
list4 = [78.0, 99.0, 97.0, 81.0]
list5 = [88.0, 78.0, 98.0, 84.0]
list6 = [100.0, 95.0, 100.0, 92.0]
data = np.array([list1,list2,list3,list4,list5,list6])
whiten = whiten(data)
centroids,_ = kmeans(whiten, 2)
result,_= vq(whiten, centroids)
print(result)
```

Output:

```
[1 0 0 1 1 0]
```

Solve with Tools

File

```
# Filename: kmeansStu2.py
import numpy as np
from sklearn.cluster import KMeans
list1 = [88.0,74.0,96.0,85.0]
list2 = [92.0,99.0,95.0,94.0]
list3 = [91.0,87.0,99.0,95.0]
list4 = [78.0,99.0,97.0,81.0]
list5 = [88.0,78.0,98.0,84.0]
list6 = [100.0,95.0,100.0,92.0]
X = np.array([list1,list2,list3,list4,list5,list6])
kmeans = KMeans(n_clusters = 2).fit(X)
pred = kmeans.predict(X)
print(pred)
```

```
from sklearn import datasets
from sklearn import svm
clf = svm.SVC(gamma=0.001, C=100.)
digits = datasets.load_digits()
clf.fit(digits.data[:-1], digits.target[:-1])
clf.predict(digits.data[-1])
```

Output:

```
[0 1 1 1 0 1]
```

Another Example

Cluster 10 DJI constituents according to close price trend of every adjacent pair of days.

File ['MMM','AXP','AAPL','BA','CAT','CVX','CSCO','KO','DIS','DD']

```
# Filename: kmeansDJI.py
listDji = ['MMM','AXP','AAPL','BA','CAT','CVX','CSCO','KO','DIS','DD']
listTemp = [0] * len(listDji)
for i in range(len(listTemp)):
    listTemp[i] = create_df(listDji[i]).close      # a function for creating a DataFrame
status = [0] * len(listDji)
for i in range(len(status)):
    status[i] = np.sign(np.diff(listTemp[i]))
kmeans = KMeans(n_clusters = 3).fit(status)
pred = kmeans.predict(status)
print(pred)
```

Output:

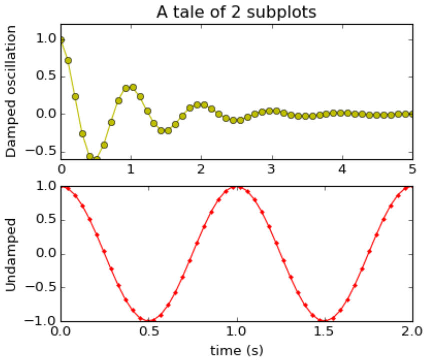
```
[2 0 2 2 0 0 2 2 1 1]
```

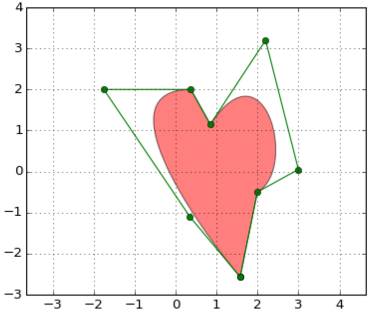
Data Processing Using Python

2

MATPLOTLIB PLOTTING

Matplotlib Plotting





- **Matplotlib Plotting**

Most famous Python 2D plotting library

- – High quality
- – Convenient plotting modules
  - Plotting API——pyplot module
  - Library——pylab module（contains useful functions in NumPy and pyplot）

Data Source

The monthly average of close price of Coca-Cola in the past year

Source

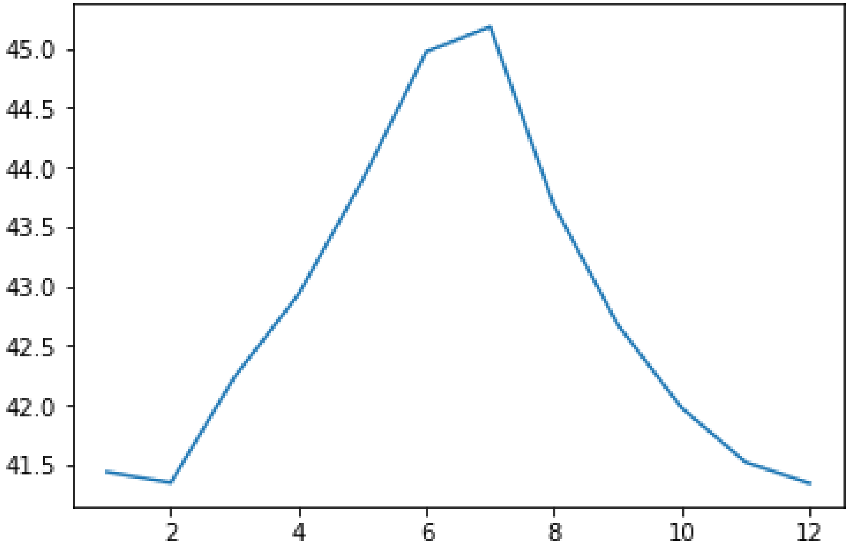
```
>>> closeMeansKO = tempkodf.groupby('month').close.mean()
>>> closeMeansKO
month
1        41.440500
2        41.350526
3        42.241304
4        42.934210
…
10       41.979524
11       41.523809
12       41.345714
```

Line Chart

Plot the monthly average of Coca-Cola’ s close price in the past year as a line chart

File

```
# Filename: plotKO.py
import matplotlib.pyplot as plt
…
x = closeMeansKO.index
y = closeMeansKO.values
plt.plot(x, y)
```

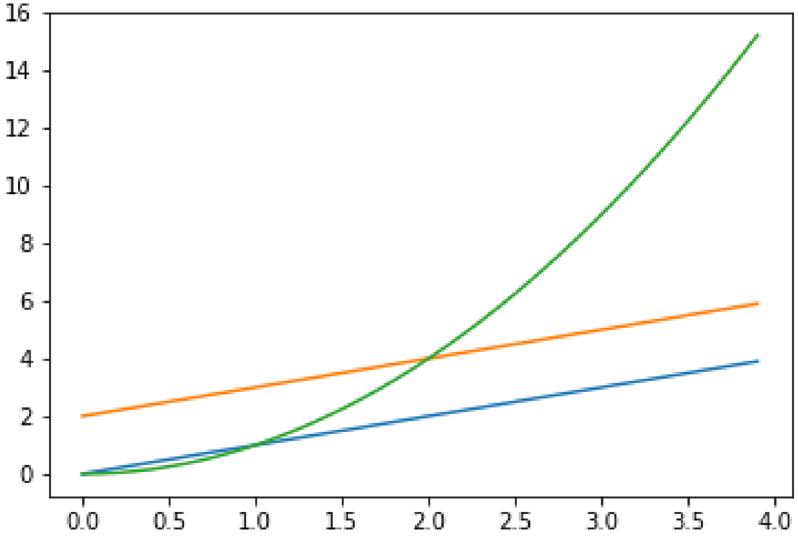


Line Chart

NumPy array can also be used as a parameter of **Matplotlib**

Source

```
>>> import numpy as np
>>> import matplotlib.pyplot as plt
>>> t=np.arange(0.,4.,0.1)
>>> plt.plot(t, t, t, t+2, t, t**2)
```

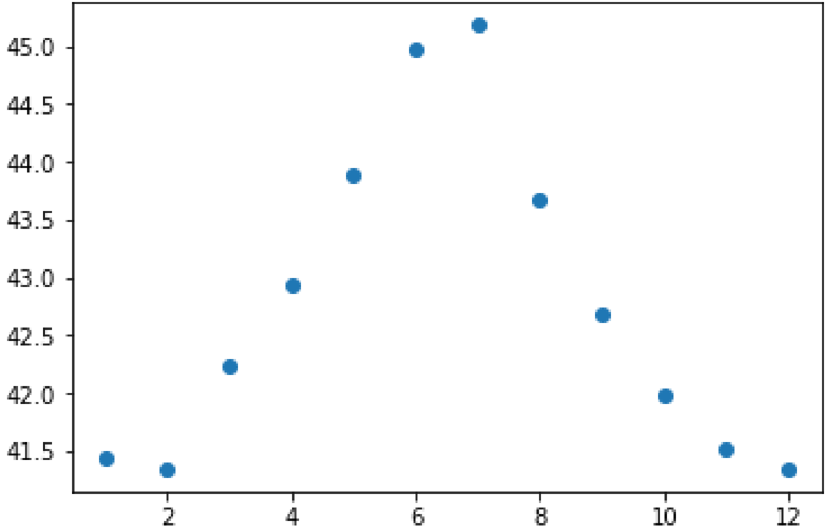


Scatter Graph

? Plot the monthly average of Coca-Cola’ s close price in the past year as a scatter graph

plt.plot(x, y)

plt.plot(x, y, 'o')

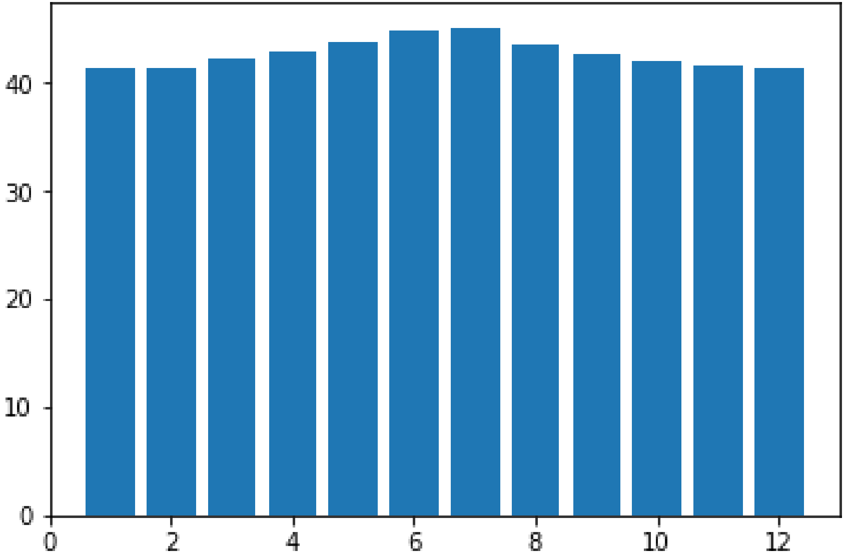


Bar Graph

Plot the monthly average of Coca-Cola’ s close price in the past year as a bar graph.

plt.plot(x, y)

plt.bar(x, y)

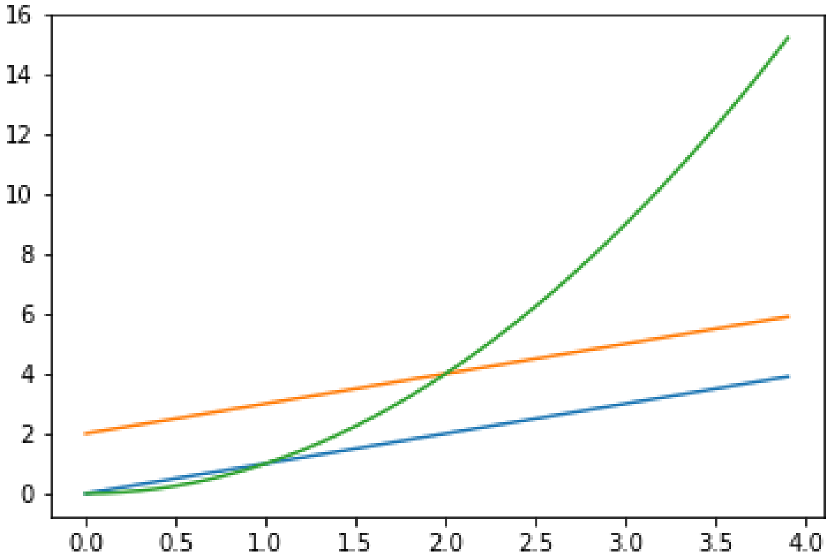


Pylab plotting

NumPy array can also be used as a parameter of **Matplotlib**

Source

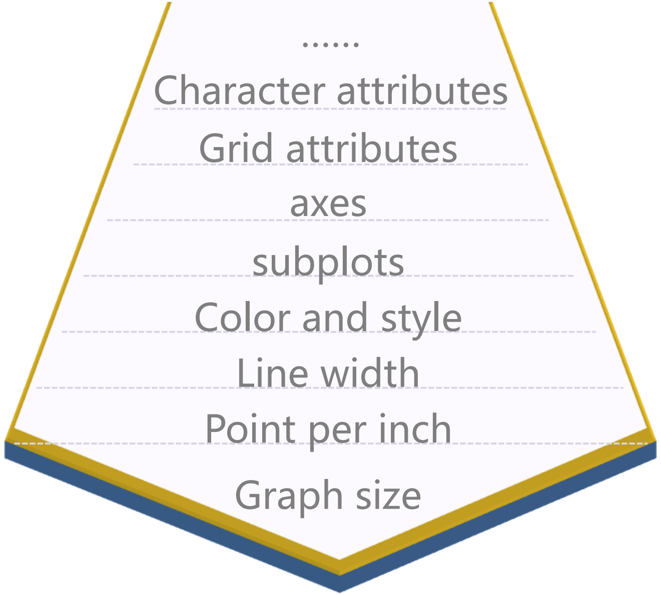
```
>>> import numpy as np
>>> import pylab as pl
>>> t=np.arange(0.,4.,0.1)
>>> pl.plot(t,t,t,t+2,t,t**2)
```



Data Processing Using Python

3 MATPLOTLIB ATTRIBUTE CONTROL

Matplotlib Attributes



……

Character attributes

Grid attributes

axes

subplots

Color and style

Line width

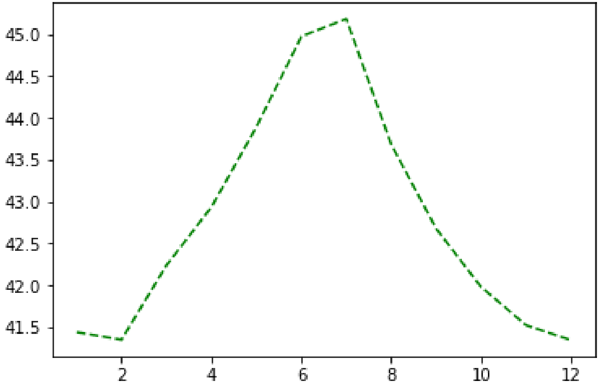
Point per inch

Graph size

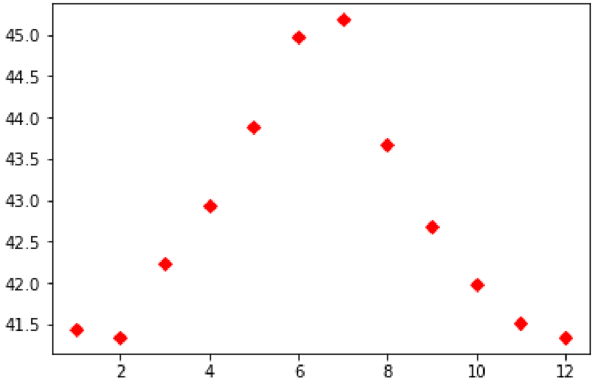
Default attributes Matplotlib can control

Color and Style

Could color, line or style of graph be modified?



```
plt.plot(x, y, 'g--')
```



```
plt.plot(x, y, 'rD')
```

Color and Style

Character	Color
b	blue
g	green
r	red
c	cyan
m	magenta
Y	yellow
k	black
w	white

Type	Description
'-'	solid
'--'	dashed
'-.'	dash_dot
':'	dotted
'None'	draw nothing
' '	draw nothing
''	draw nothing

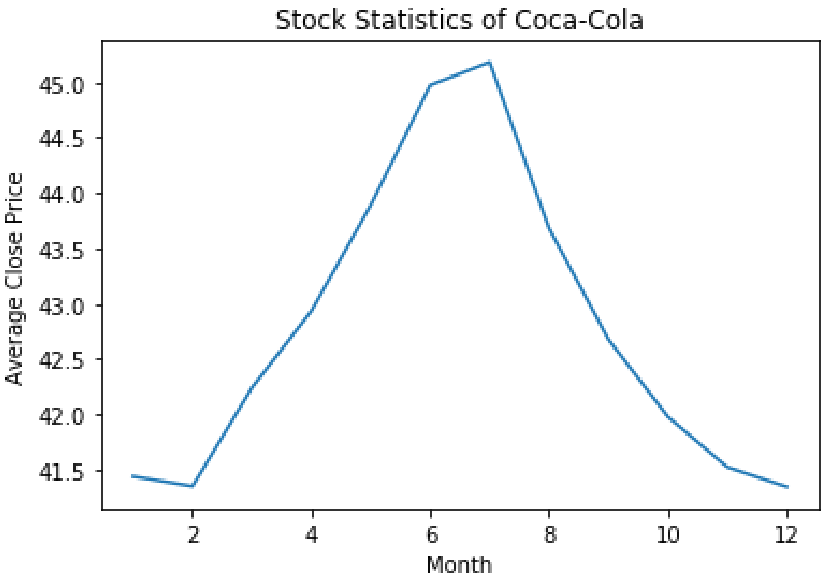
Mark	Description
"o"	circle
"v"	triangle_down
"s"	square
"p"	pentagon
"*"	star
"h"	hexagon1
"+"	plus
"D"	diamond
…	…

Words

Add titles：graph, vertical axis and horizontal axis

F_{ile}

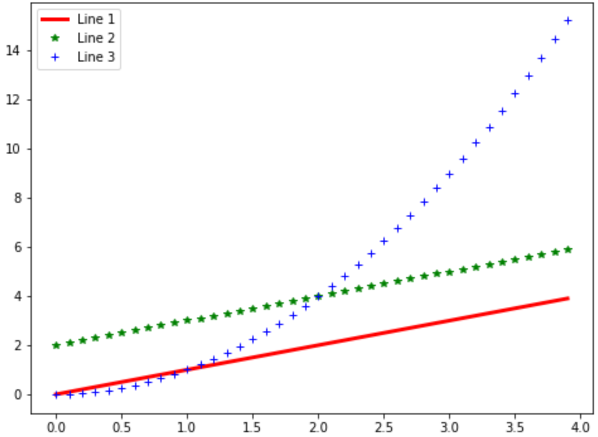
```
# Filename: plotKO.py
import matplotlib.pyplot as plt
…
x = closeMeansKO.index
y = closeMeansKO.values
plt.title('Stock Statistics of Coca-Cola')
plt.xlabel('Month')
plt.ylabel('Average Close Price')
plt.plot(x, y)
```



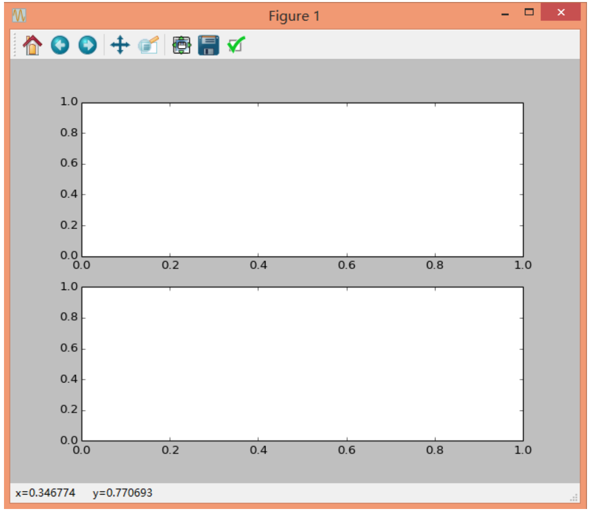
Other Attributes

File

```
# Filename: multilines.py
import pylab as pl
import numpy as np
pl.figure(figsize=(8,6),dpi=100)
t=np.arange(0.,4.,0.1)
pl.plot(t,t,color='red',linestyle='-',linewidth=3,label='Line 1')
pl.plot(t,t+2,color='green',linestyle='',marker='*',linewidth=3,label='Line 2')
pl.plot(t,t**2,color='blue',linestyle='',marker='+',linewidth=3,label='Line 3')
pl.legend(loc='upper left')
```

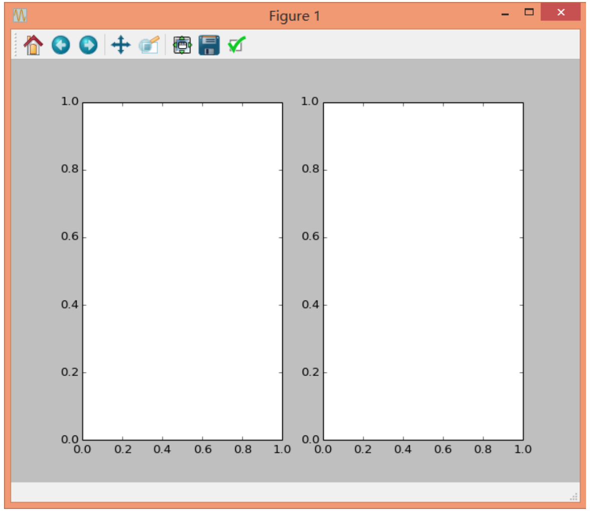


subplots

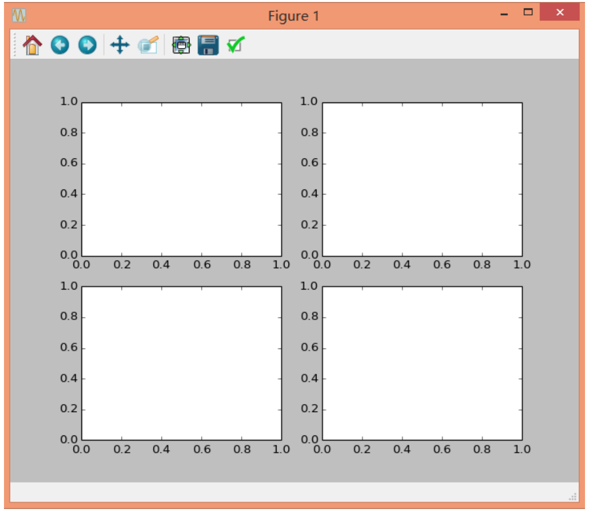


x=0.346774 y=0.770693

```
plt.subplot(211)
plt.subplot(212)
```



```
plt.subplot(121)
plt.subplot(122)
```



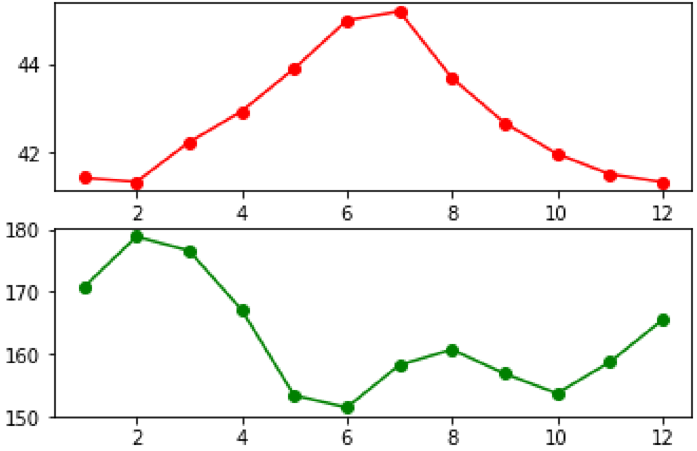
```
plt.subplot(221)
plt.subplot(222)
plt.subplot(223)
plt.subplot(224)
```

subplots

Plot the monthly average close price of Coca-Cola and IBM in the past year into a single graph.

File

```
#The data of Coca-Cola and IBM is ready
plt.subplot(211)
plt.plot(x,y,color='r',marker='o')
plt.subplot(212)
plt.plot(xi,yi,color='green',marker='o')
```

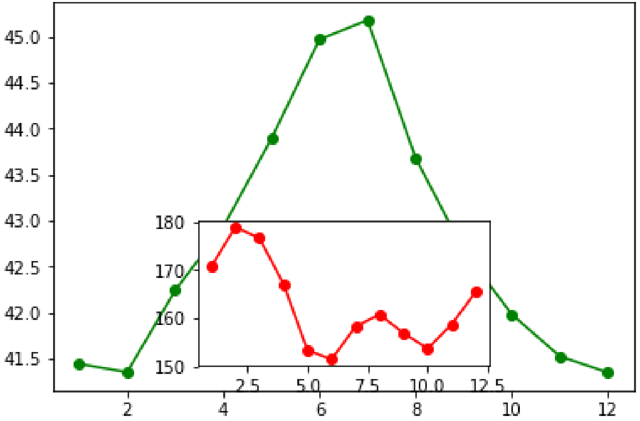


subplots-axes

Plot the monthly average close price of Coca-Cola and IBM in the past year into a single graph.

File

```
#The data of Coca-Cola and IBM is ready
plt.axes([.1,.1,0.8,0.8])
plt.plot(x,y,color='green',marker='o')
plt.axes([.3,.15,0.4,0.3])
plt.plot(xi,yi,color='r',marker='o')
plt.savefig('1.jpg')
```



axes([left,bottom,width,height])

Range of parameter: (0,1)

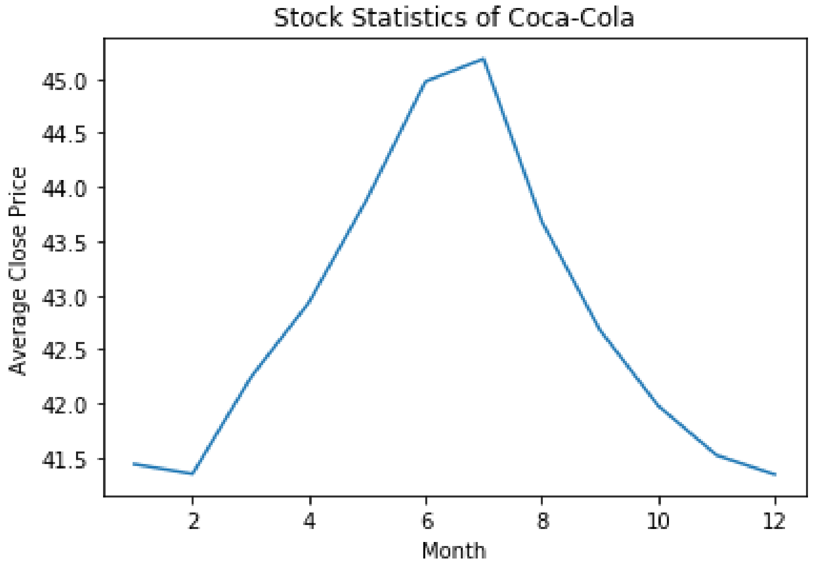
Data Processing Using Python

4 PANDAS PLOTTING

Python Example

Source

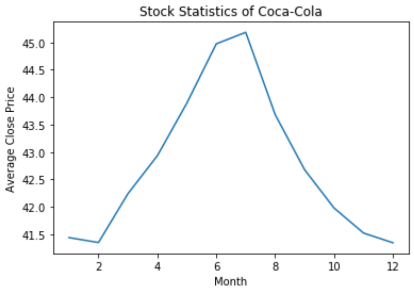
```
>>> plt.title('Stock Statistics of Coca-Cola')
>>> plt.xlabel('Month')
>>> plt.ylabel('Average Close Price')
>>> plt.plot(closeMeansKO)
```

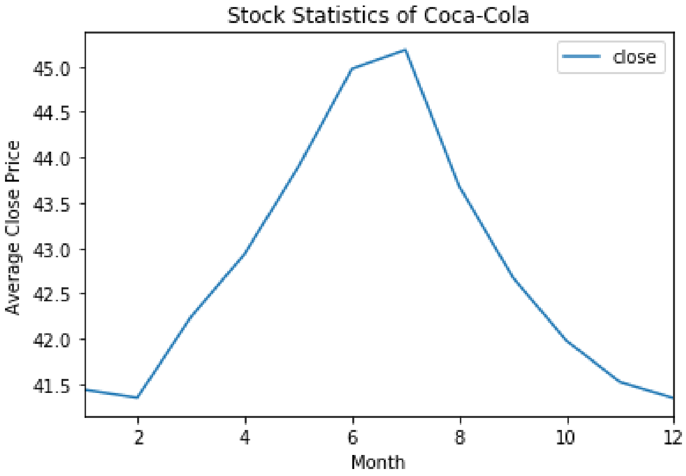


Pandas plotting

Source

```
>>> import pandas as pd

>>> closeMeansKO.plot()

>>> plt.title('Stock Statistics of Coca-Cola')

>>> plt.xlabel('Month')

>>> plt.ylabel('Average Close Price')
```

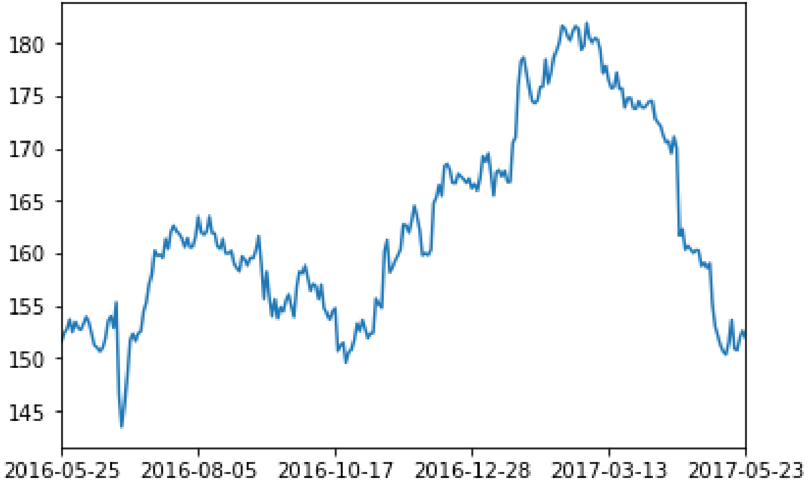




Pandas plotting

Plot the close price of IBM in the past year as line chart

```
# Filename: quotesdfplot.py
...
quotes = retrieve_quotes_historical('IBM')
quotesdfIBM = pd.DataFrame(quotes)
quotesdfIBM.close.plot()
```

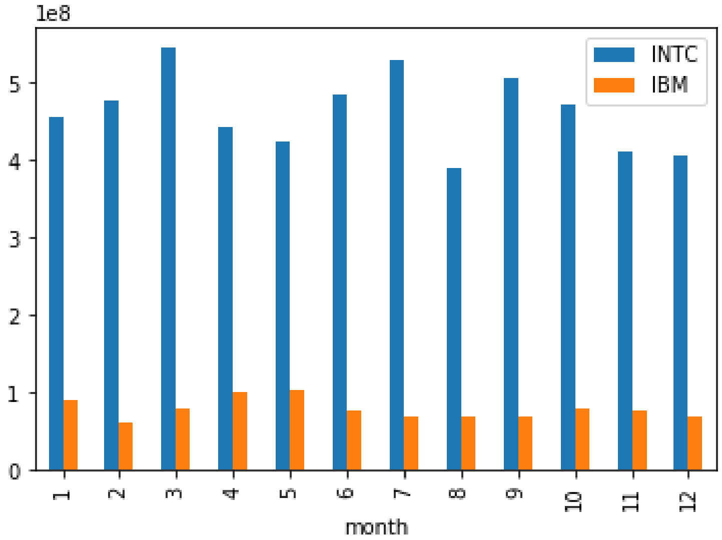


Pandas Format Control

? Use bar graph to compare the volume of Intel and IBM in the past year

File

```
# Filename: plot_volumes.py
…
INTC_volumes = create_volumes('INTC')
IBM_volumes = create_volumes('IBM')
quotesIIdf = pd.DataFrame()
quotesIIdf['INTC'] = INTC_volumes
quotesIIdf['IBM'] = IBM_volumes
quotesIIdf.plot(kind = 'bar')
```

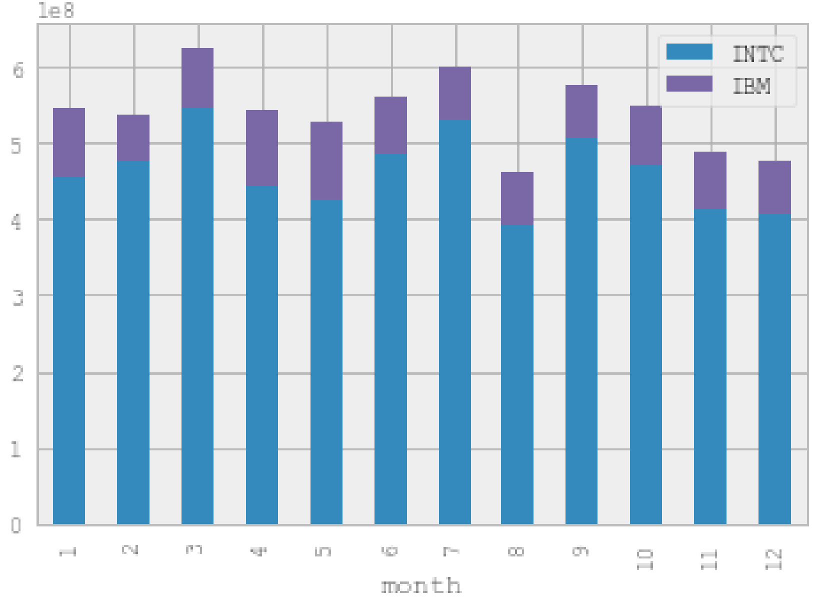


Pandas Format Control

Use bar graph to compare the volume of Intel and IBM in the past year

```
quotesIIdf.plot(kind='bar')
```

```
quotesIIdf.plot(kind='bar',stacked = True)
```

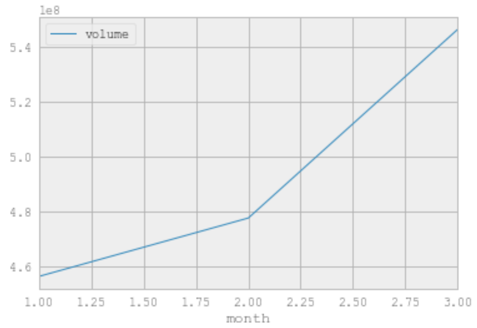


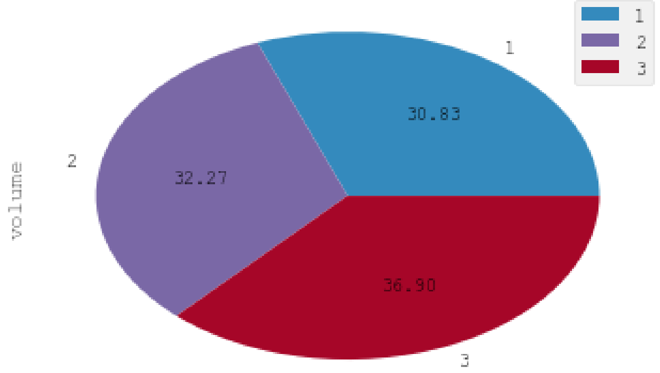
Pandas Format Control

? The ratio comparison of Intel’ s close price in first three months this year

```
quotesINTC.plot()
```

```
quotesINTC.plot(kind = 'pie',
subplots = True, autopct = '%.2f')
```

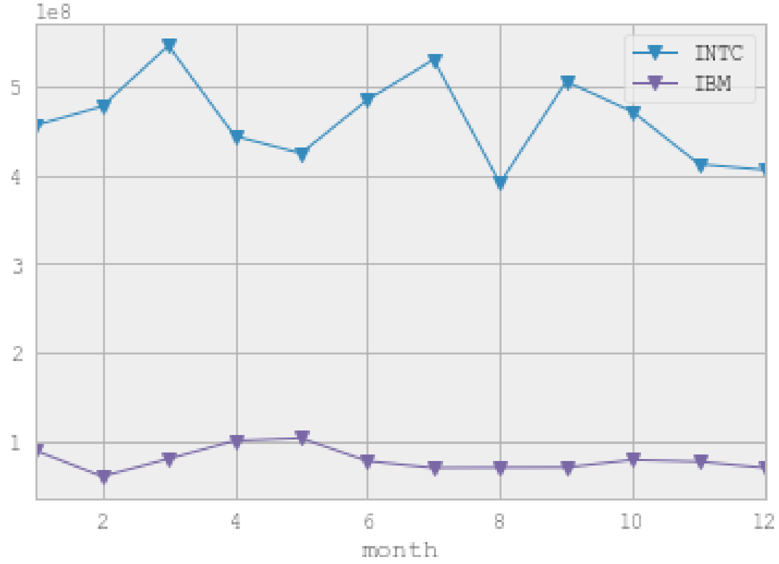




Pandas Format Control

Source

```
#The data of Intel and IBM is ready
>>> quotesIIdf.plot(marker='v')
```



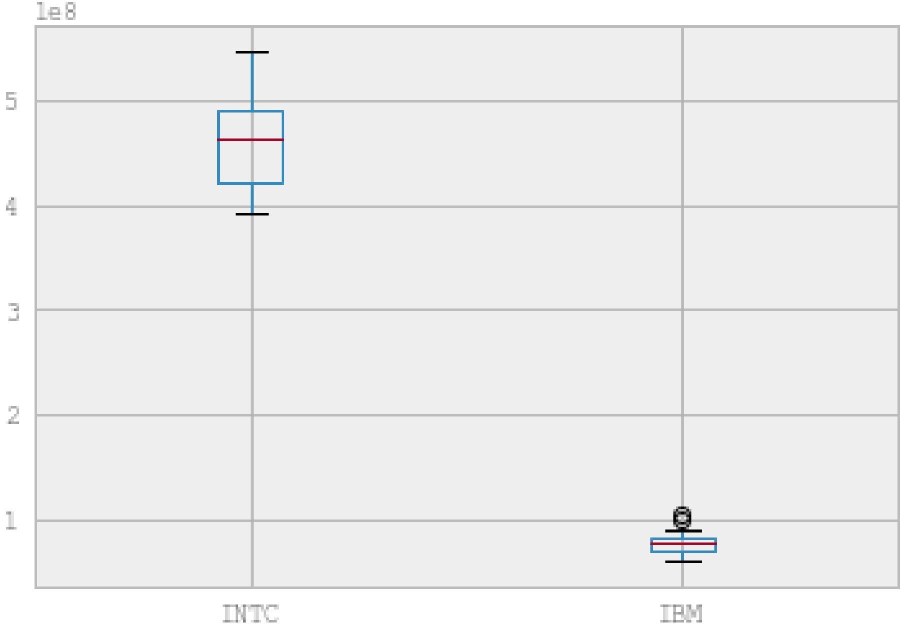
Box Plot

?

Plot the volume of Intel and IBM in the past year with box plot.

quotesIIdf.plot(kind='bar')

quotesIIdf.boxplot()



Maximum，First Quartile，Medium，Third Quartile，Minimum

Data Processing Using Python

5 DATA STORAGE AND FETCH

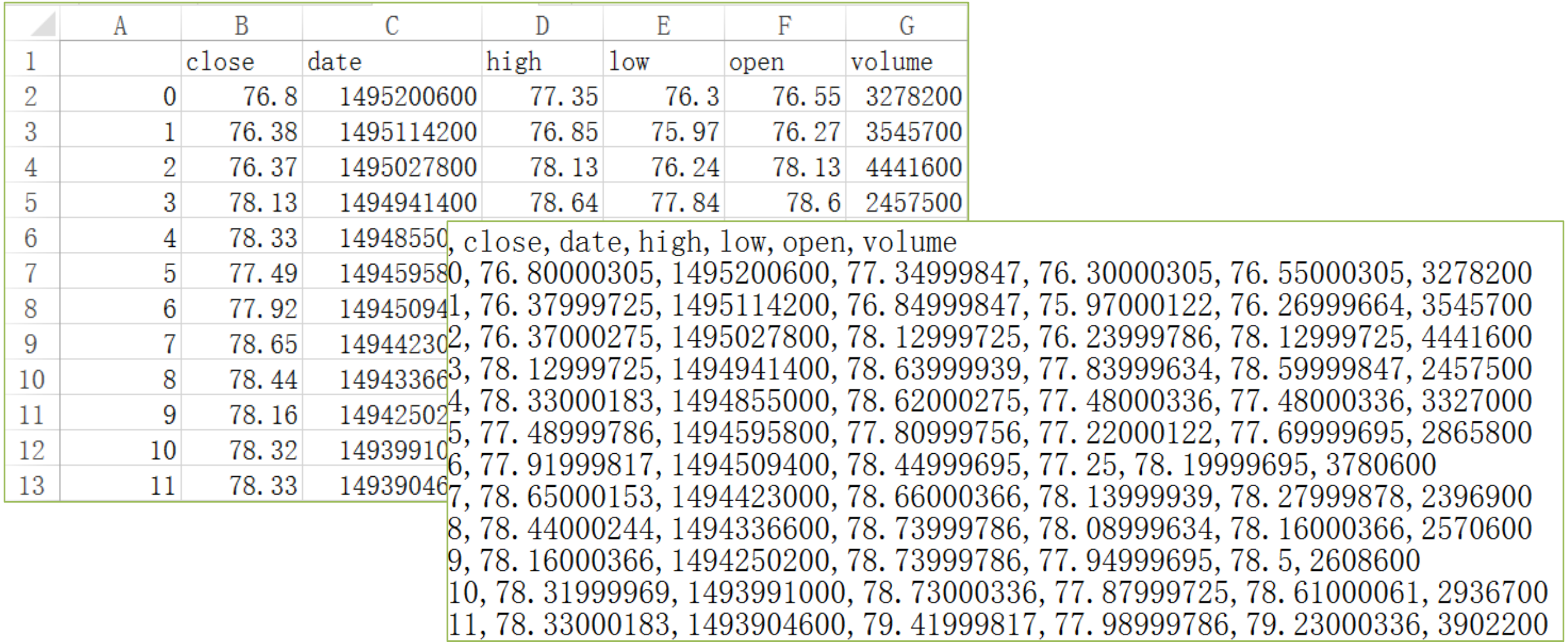
Read and Write of csv Format

? Store the basic stock information of American Express in the past year into stockAXP.csv.

File

```
# Filename: to_csv.py
import pandas as pd
…
quotes = retrieve_quotes_historical('AXP')
df = pd.DataFrame(quotes)
df.to_csv('stockAXP.csv')
```

Read and Write of csv Format



	A	B	C	D	E	F	G
1		close	date	high	low	open	volume
2	0	76.8	1495200600	77.35	76.3	76.55	3278200
3	1	76.38	1495114200	76.85	75.97	76.27	3545700
4	2	76.37	1495027800	78.13	76.24	78.13	4441600
5	3	78.13	1494941400	78.64	77.84	78.6	2457500
6	4	78.33	14948550				
7	5	77.49	14945958				
8	6	77.92	14945094				
9	7	78.65	14944230				
10	8	78.44	14943366				
11	9	78.16	14942502				
12	10	78.32	14939910				
13	11	78.33	14939046				

```
,close,date,high,low,open,volume
0,76.80000305,1495200600,77.34999847,76.30000305,76.55000305,3278200
1,76.37999725,1495114200,76.84999847,75.97000122,76.26999664,3545700
2,76.37000275,1495027800,78.12999725,76.23999786,78.12999725,4441600
3,78.12999725,1494941400,78.63999939,77.83999634,78.59999847,2457500
4,78.33000183,1494855000,78.62000275,77.48000336,77.48000336,3327000
5,77.48999786,1494595800,77.80999756,77.22000122,77.69999695,2865800
6,77.91999817,1494509400,78.44999695,77.25,78.19999695,3780600
7,78.65000153,1494423000,78.66000366,78.13999939,78.27999878,2396900
8,78.44000244,1494336600,78.73999786,78.08999634,78.16000366,2570600
9,78.16000366,1494250200,78.73999786,77.94999695,78.5,2608600
10,78.31999969,1493991000,78.73000336,77.87999725,78.61000061,2936700
11,78.33000183,1493904600,79.41999817,77.98999786,79.23000336,3902200
```

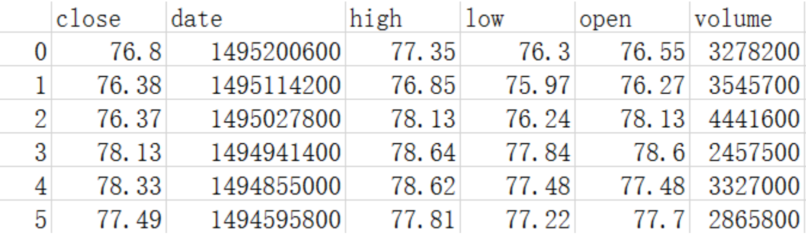
Read and Write of csv Format

Source

```
>>> result = pd.read_csv('stockAXP.csv')
>>> result
  Unnamed: 0      close        date       high        low       open  \
0          0  76.800003  1495200600  77.349998  76.300003  76.550003
1          1  76.379997  1495114200  76.849998  75.970001  76.269997
2          2  76.370003  1495027800  78.129997  76.239998  78.129997
3          3  78.129997  1494941400  78.639999  77.839996  78.599998
…
>>> print(result['close'])
0      76.800003
1      76.379997
2      76.370003
3      78.129997
…
```

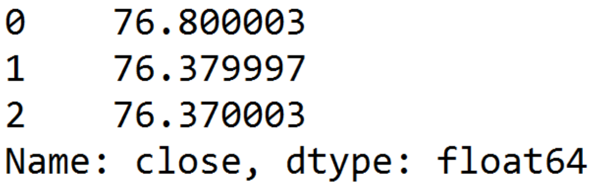
Read and Write of Excel Data

```
# Filename: to_excel.py
…
quotes = retrieve_quotes_historical('AXP')
df = pd.DataFrame(quotes)
df.to_excel('stockAXP.xlsx', sheet_name='AXP')
```



	close	date	high	low	open	volume
0	76.8	1495200600	77.35	76.3	76.55	3278200
1	76.38	1495114200	76.85	75.97	76.27	3545700
2	76.37	1495027800	78.13	76.24	78.13	4441600
3	78.13	1494941400	78.64	77.84	78.6	2457500
4	78.33	1494855000	78.62	77.48	77.48	3327000
5	77.49	1494595800	77.81	77.22	77.7	2865800

```
# Filename: read_excel.py
…
df = pd.read_excel('stockAXP.xlsx')
print(df['close'][:3])
```



```
0    76.800003
1    76.379997
2    76.370003
Name: close, dtype: float64
```

Data Processing Using Python

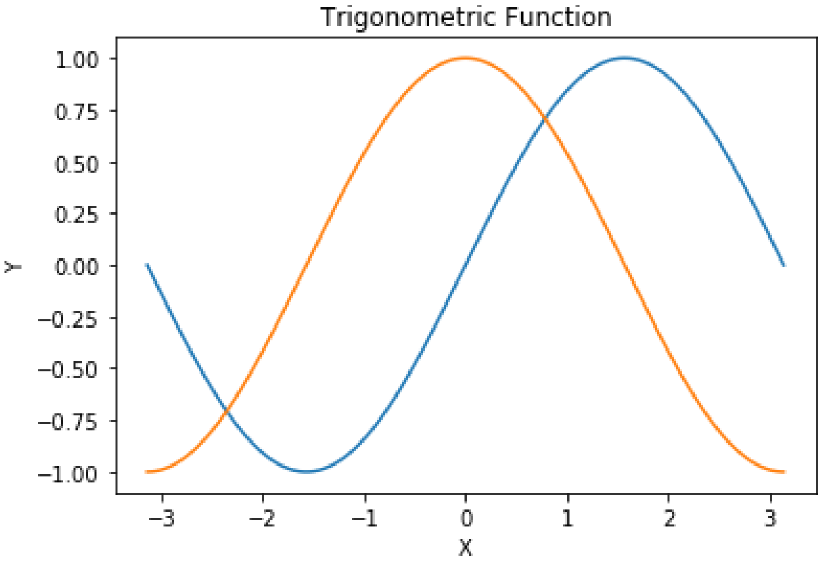
6

SCIENTIFIC APPLICATION OF PYTHON

Computation of Trigonometric Function

File

```
# Filename: mathA.py
import numpy as np
import pylab as pl
x = np.linspace(-np.pi, np.pi, 256)
s = np.sin(x)
c = np.cos(x)
pl.title('Trigonometric Function')
pl.xlabel('X')
pl.ylabel('Y')
pl.plot(x,s)
pl.plot(x,c)
```



Fast Fourier Transformation

Array：[1,1,…,1,-1,-1,…,1,1,1…,1]

File

```
# Filename: mathB.py
import scipy as sp
import pylab as pl
listA = sp.ones(500)
listA[100:300] = -1
f = sp.fft(listA)
pl.plot(f)
```

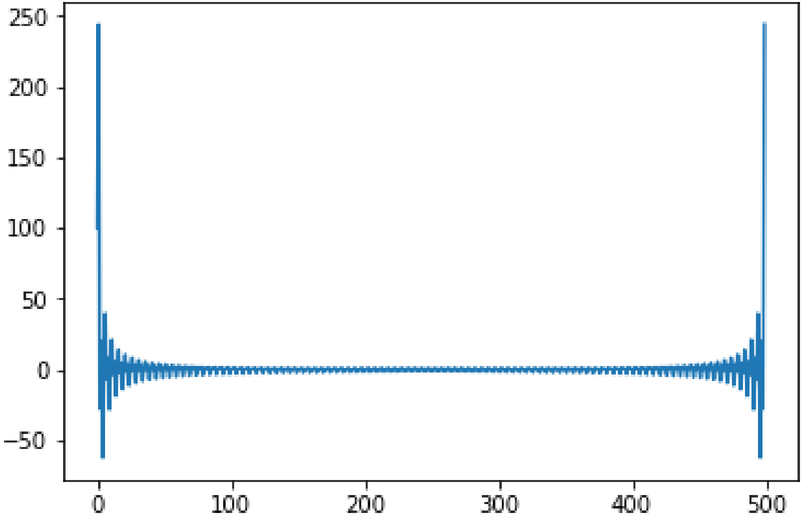
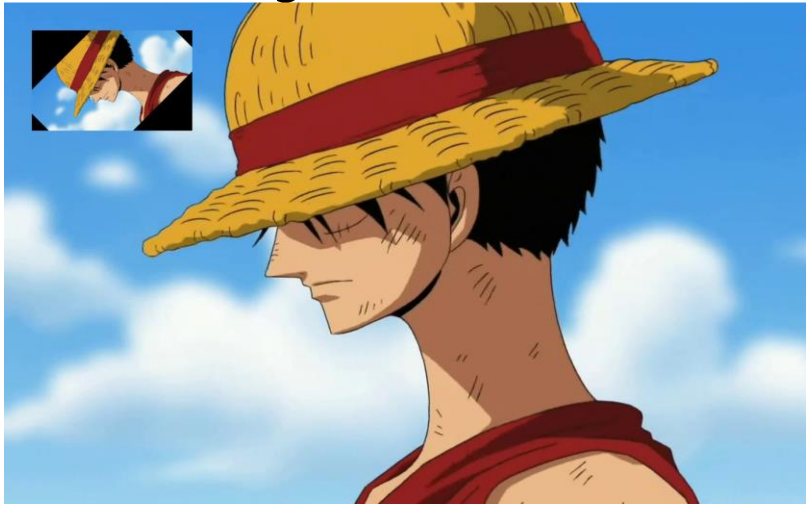


Image Processing

- **Useful Python Library**
  - Pillow(PIL)
  - OpenCV
  - Skimage



File

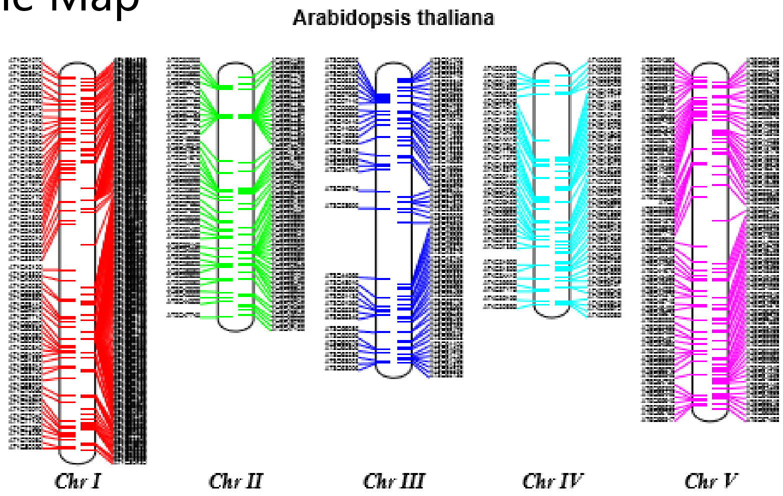
```
# Filename: pasteimg.py
from PIL import Image
im1 = Image.open('1.jpg')
print(im1.size, im1.format, im1.mode)
Image.open('1.jpg').save('2.png')
im2 = Image.open('2.png')
size = (288, 180)
im2.thumbnail(size)
out = im2.rotate(45)
im1.paste(out, (50,50))
```

Biopython

- Developed by Biopython, a group focusing on computational biology with Python
- Sequence, Alphabet and Chromosome Map

Source

```
>>> from Bio.Seq import Seq
>>> my_seq = Seq("AGTACACTGGT")
>>> my_seq.alphabet
Alphabet()
>>> print(my_seq)
AGTACACTGGT
```

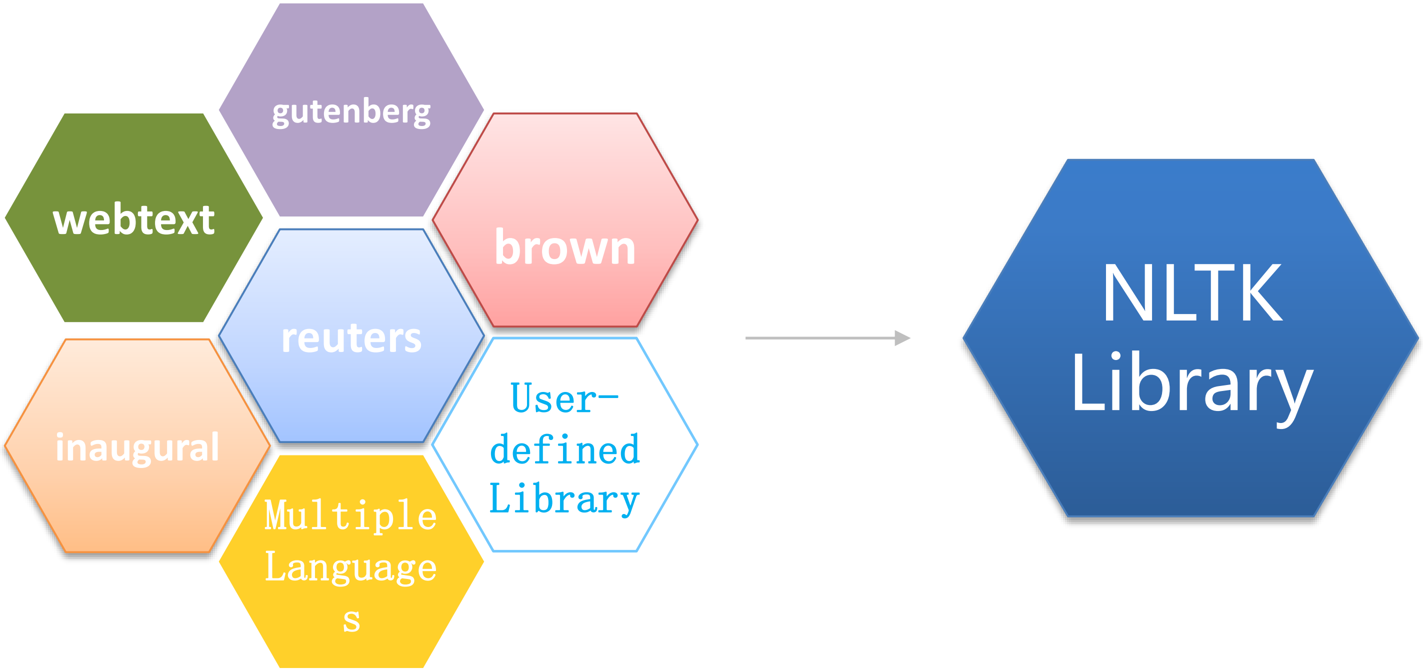


Data Processing Using Python

7

SOCIAL SCIENCE APPLICATION OF PYTHON

NLTK Library



Gutenberg Project

- Count all books currently included in Gutenberg Project

Source

```
>>> from nltk.corpus import gutenberg
>>> gutenberg.fileids()
['austen-emma.txt', 'austen-persuasion.txt', 'austen-sense.txt', 'bible-kjv.txt', 'blake-poems.txt', 'bryant-stories.txt', 'burgess-busterbrown.txt', 'carroll-alice.txt', 'chesterton-ball.txt', 'chesterton-brown.txt', 'chesterton-thursday.txt', 'edgeworth-parents.txt', 'melville-moby_dick.txt', 'milton-paradise.txt', 'shakespeare-caesar.txt', 'shakespeare-hamlet.txt', 'shakespeare-macbeth.txt', 'whitman-leaves.txt']
```

Gutenberg Project

- Some simple calculation

Source

```
>>> from nltk.corpus import gutenberg
>>> allwords = gutenberg.words('shakespeare-hamlet.txt')
>>> len(allwords)
37360
>>> len(set(allwords))
5447
>>> allwords.count('Hamlet')
99
>>> A = set(allwords)
>>> longwords = [w for w in A if len(w) > 12]
>>> print(sorted(longwords))
```

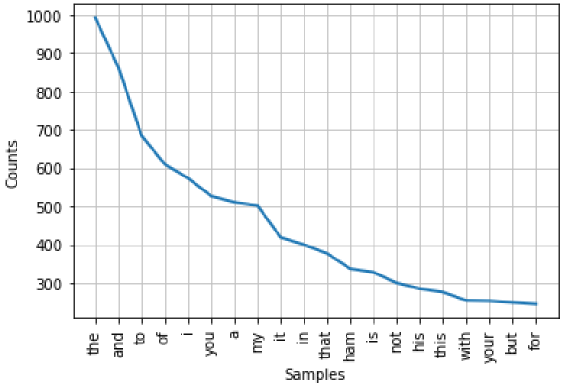
Output:

```
['Circumstances',
'Guildensterne',
'Incontinencie',
'Recognizances',
'Vnderstanding',
'determination',
'encompassement',
'entertainment',
'imperfections',
'indifferently',
'instrumentall',
'reconcilement',
'stubbornnesse',
'transformation',
'vnderstanding']
```

Gutenberg Project

File

```
# Filename: freqG20.py
from nltk.corpus import gutenberg
from nltk.probability import *
fd2 = FreqDist([sx.lower() for sx in allwords if sx.isalpha()])
print(fd2.B())
print(fd2.N())
fd2.tabulate(20)
fd2.plot(20)
fd2.plot(20, cumulative = True)
```



Output:

```
4699
30266
 the  and   to   of    i  you    a   my   it   in that  ham
is  not  his this with your  but  for
 993  863  685  610  574  527  511  502  419  400
377  337  328  300  285  276  254  253  249  245
```

Inaugural Library

Source

```
>>> from nltk.corpus import inaugural
>>> from nltk.probability import *
>>> fd3 = FreqDist([s for s in inaugural.words()])
>>> print(fd3.freq('freedom'))
0.00119394791917
```

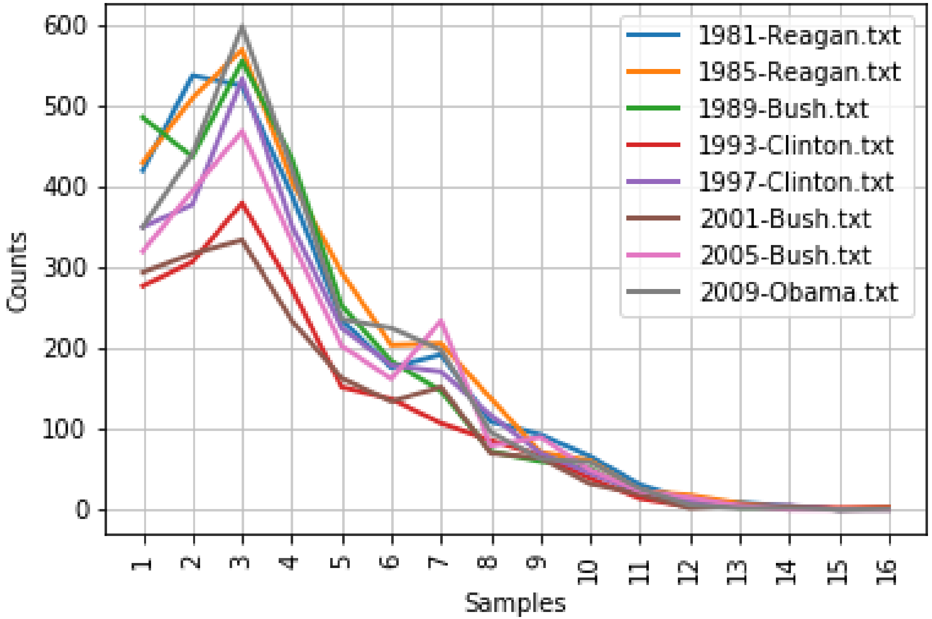
File

```
# Filename: inaugural.py
from nltk.corpus import inaugural
from nltk.probability import *
cfd = ConditionalFreqDist(
            (fileid, len(w))
            for fileid in inaugural.fileids()
            for w in inaugural.words(fileid)
            if fileid > '1980' and fileid < '2010')
print(cfd.items())
cfd.plot()
```

Inaugural Library

Output:

```
dict_items([('1981-Reagan.txt',
FreqDist({2: 538, 3: 525, 1: 420, 4:
390, 5: 235, 7: 192, 6: 176, 8: 109, 9:
93, 10: 66, ...})), ... , ('2005-Bush.txt',
FreqDist({3: 469, 2: 395, 4: 332, 1:
320, 7: 234, 5: 203, 6: 162, 9: 90, 8:
79, 10: 49, ...})), ('2009-Obama.txt',
FreqDist({3: 599, 2: 441, 4: 422, 1:
350, 5: 236, 6: 225, 7: 198, 8: 96, 9:
63, 10: 59, ...}))])
```


Counts
Samples
1981-Reagan.txt
1985-Reagan.txt
1989-Bush.txt
1993-Clinton.txt
1997-Clinton.txt
2001-Bush.txt
2005-Bush.txt
2009-Obama.txt