數據分析、資料科學

matplotlib

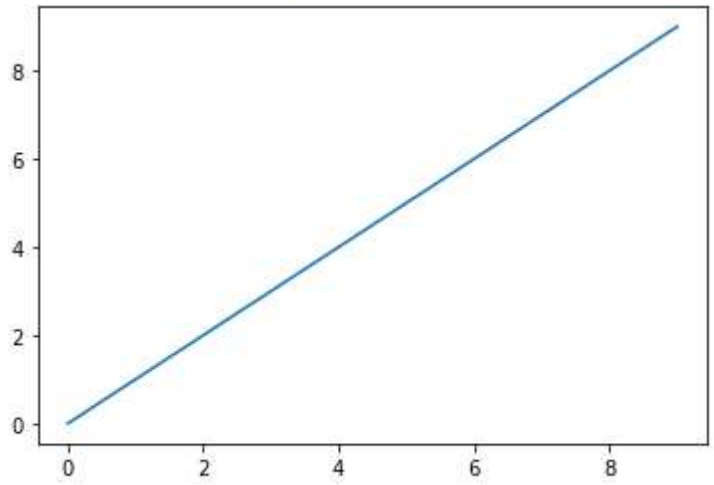
Matplotlib

- Python中最常用來繪圖的套件
- 所使用的函式名稱和matlab很像
- MatPlotLib 是 Matrix Plotting Libray 的縮寫。它是 Python 最主要的 2D 繪圖套件，是用 Python 來畫圖的首選套件。
- Matplotlib。

Matplotlib

```
[5]: x = np.arange(10)
     plt.plot(x)
```

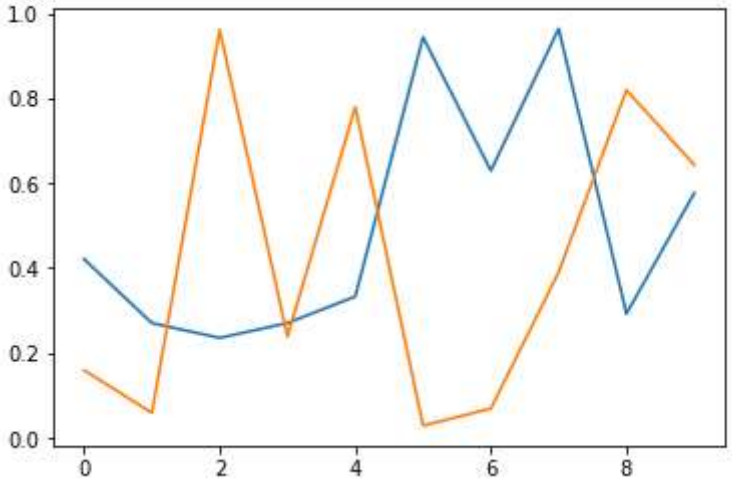
```
[5]: [<matplotlib.lines.Line2D at 0x17f7ffa23a0>]
```



Matplotlib

```
x = np.random.rand(10, 2)
plt.plot(x)
```

```
[<matplotlib.lines.Line2D at 0x17f017300d0>,
 <matplotlib.lines.Line2D at 0x17f01730100>]
```

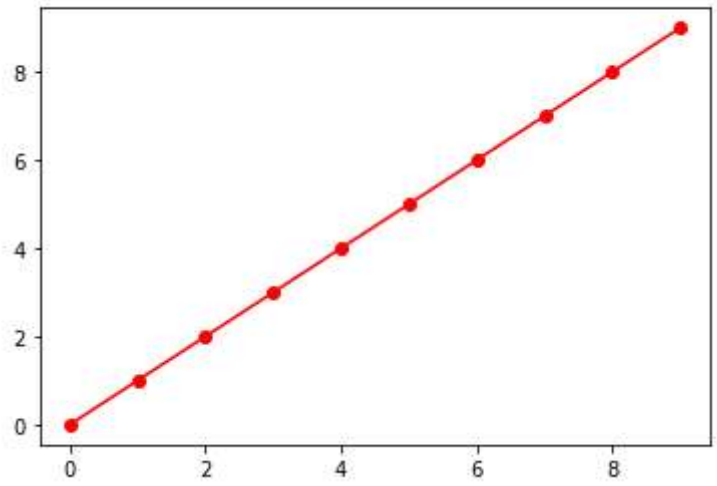


Matplotlib

- 控制線的顏色、線型
- https://matplotlib.org/2.0.2/api/pyplot_api.htr

```
x = np.arange(10)
plt.plot(x, 'ro-')
```

```
[<matplotlib.lines.Line2D at 0x17f018b5760>]
```

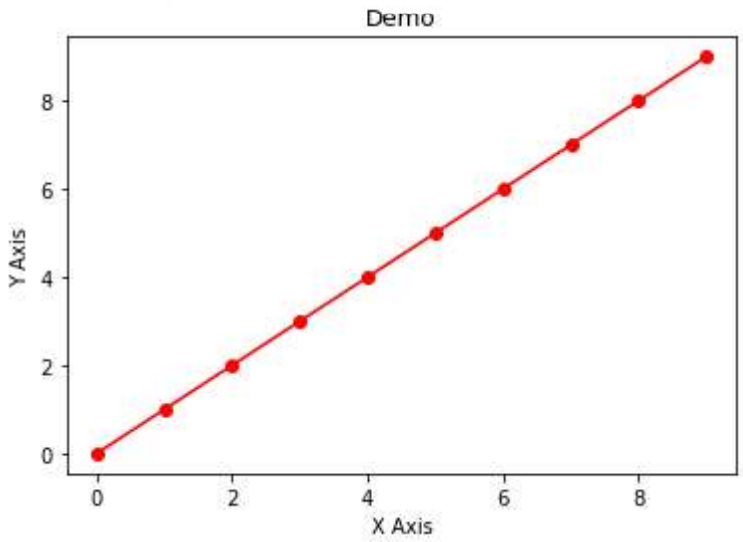


Matplotlib

- 控制圖的標題，X, Y軸標示文字

```
x = np.arange(10)
plt.plot(x, 'ro-')
plt.title('Demo')
plt.xlabel('X Axis')
plt.ylabel('Y Axis')
```

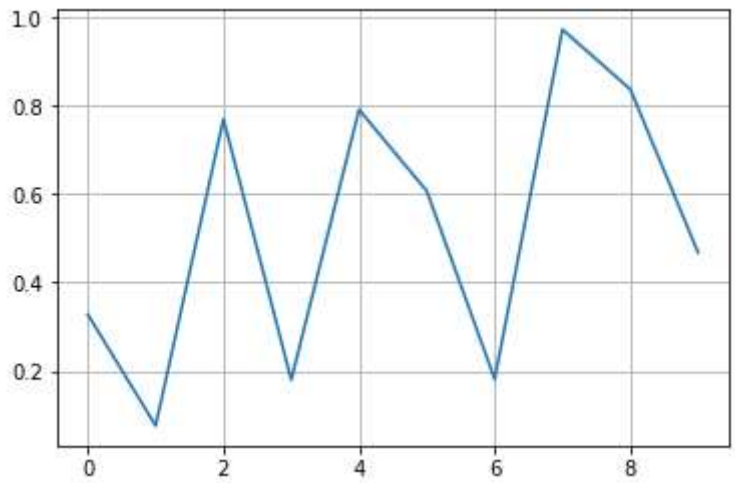
```
Text(0,0.5,'Y Axis')
```



Matplotlib

- 加入格線

```
x = np.random.rand(10)
plt.plot(x)
plt.grid(True)
```

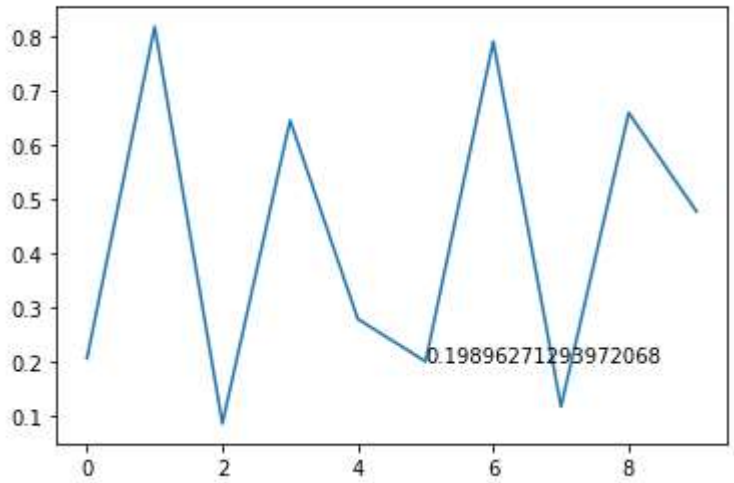


Matplotlib

- 在圖上標註文字

```
x = np.random.rand(10)
plt.plot(x)
i = 5
plt.text(i, x[i], str(x[i]))
```

```
Text(5, 0.19896271293972068, '0.19896271293972068')
```

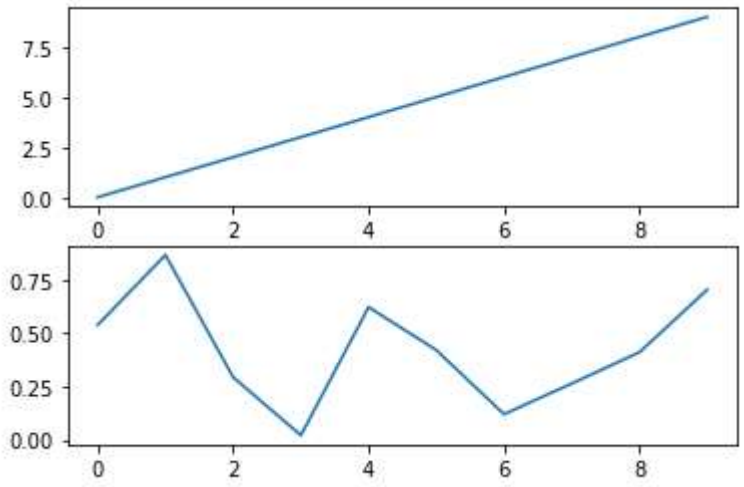


Matplotlib

- 繪製子圖

```
plt.subplot(211)
x = np.arange(10)
plt.plot(x)
plt.subplot(212)
y = np.random.rand(10)
plt.plot(y)
```

```
[<matplotlib.lines.Line2D at 0x17f019b8df0>]
```

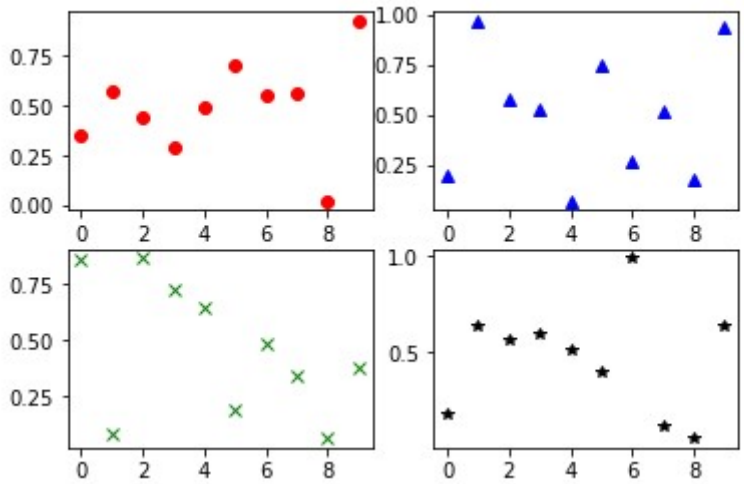


Matplotlib

- 繪製子圖
- ref: https://matplotlib.org/2.0.2/examples/pyl

```
plt.subplot(221)
x1 = np.random.rand(10)
plt.plot(x1, 'ro')
plt.subplot(222)
x2 = np.random.rand(10)
plt.plot(x2, 'b^')
plt.subplot(223)
x3 = np.random.rand(10)
plt.plot(x3, 'gx')
plt.subplot(224)
x4 = np.random.rand(10)
plt.plot(x4, 'k*')
```

```
[<matplotlib.lines.Line2D at 0x17f01ae8f40>]
```



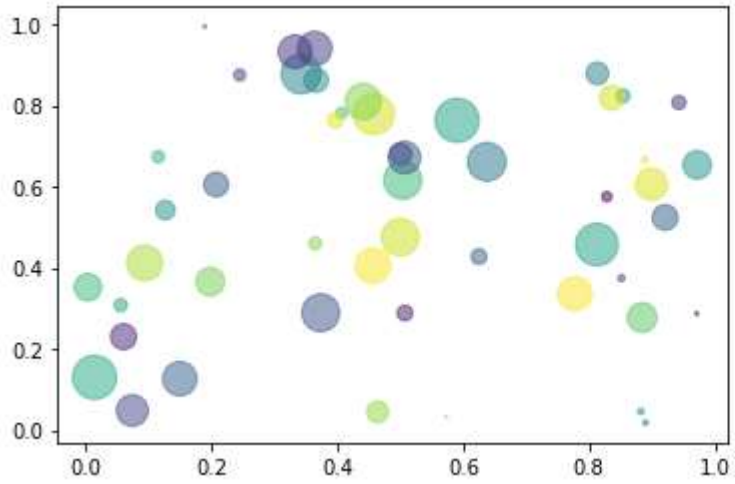
Matplotlib

- 畫散布圖

```
# 請參考：http://matplotlib.org/examples/shapes_and_collections/scatter_demo.html
N = 50
x = np.random.rand(N)
y = np.random.rand(N)
colors = np.random.rand(N)
area = np.pi * (15 * np.random.rand(N))**2  # 0 to 15 point radii

plt.scatter(x, y, s=area, c=colors, alpha=0.5)
```

```
<matplotlib.collections.PathCollection at 0x17f01b90700>
```

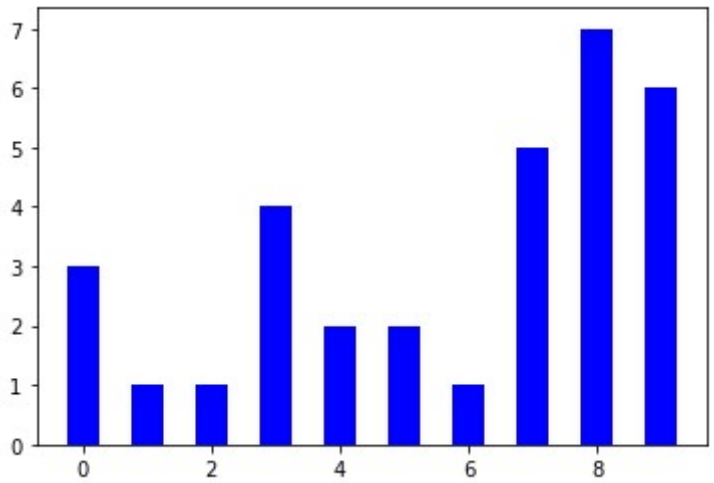


Matplotlib

- 直方圖

```
# bar chart - 1
N = 10
y = np.random.randint(1, 10, size=N)
x = range(N)
width = 0.5
plt.bar(x, y, width, color="blue")
```

```
<BarContainer object of 10 artists>
```

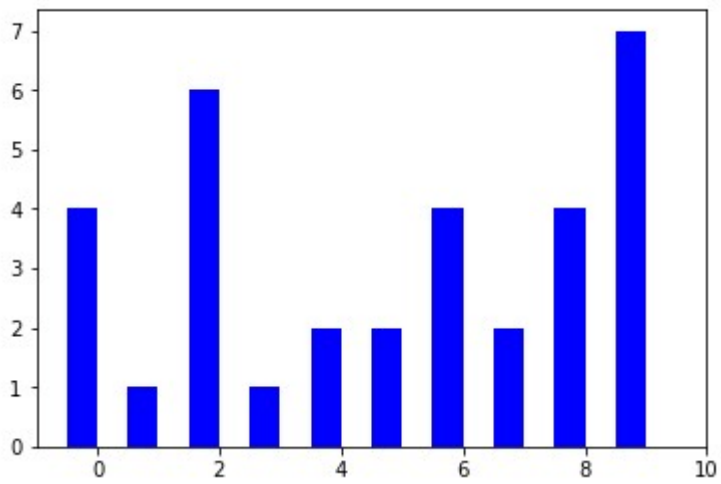


Matplotlib

- 直方圖

```
# bar chart - 2
N = 10
y = np.random.randint(1, 10, size=N)
x = np.arange(N)
width = 0.5
plt.bar(x - 0.25, y, width, color="blue")
plt.xlim(-1, 10)
```

```
(-1, 10)
```



numpy

Numpy: 數學運算的基礎套件

- https://numpy.org/
- 向量運算
- 速度，底層都是C實作的
- 很多工程、科學計算都使用此套件為基礎往上蓋房子

Numpy(2) 常用功能

函式	說明
sum	總和
mean	平均
std	標準差
var	變異數
min, max	最小值、最大值
argmin, argmax	最小值的索引、最大值的索引
cumsum	和的累計值
cumprod	積的累計值

Numpy

```
X = [1, 2, 3]
Y = np.array(X)
Y
```

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

```
X
```

```
[1, 2, 3]
```

```
Z = np.array([4, 5, 6])
Z
```

```
array([4, 5, 6])
```

Numpy

```
I = np.eye(4)
I
```

```
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
```

```
I.shape
```

```
(4, 4)
```

```
G = np.arange(25)
G
```

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24])
```

```
G.shape
```

```
(25,)
```

Numpy

- 重新塑形

```
F = G.reshape(5, 5)
F
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

```
G.shape
```

```
(5, 5)
```

Numpy

- 方便的布林運算方式

```
G
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

```
G > 10
```

```
array([[False, False, False, False, False],
       [False, False, False, False, False],
       [False,  True,  True,  True,  True],
       [ True,  True,  True,  True,  True],
       [ True,  True,  True,  True,  True]])
```

Numpy

- 列出合乎條件的index

```
G[G > 10]
```

```
array([11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24])
```

Numpy

- 列出合乎條件的進行運算

```
np.where(G > 10, G, 0)
```

```
array([[ 0,  0,  0,  0,  0],
       [ 0,  0,  0,  0,  0],
       [ 0, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

Pandas

Pandas

- 常見的資料分析及整理的套件
- 重要的資料元件和操作
  - Series, Dataframe
- https://pandas.pydata.org/
- https://pandas.pydata.org/docs/user_guide/index.html

Pandas

- 引入套件

```
import pandas as pd

pd.__version__
```

```
'1.3.4'
```

```
%matplotlib inline

import numpy as np
from datetime import datetime
```

Pandas

- Series的使用

Series

```
s = pd.Series([1, 2, 3, 4, 5])
s
```

```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

```
u = pd.Series([1, 2, 3, 4, 5])
```

```
s+u
```

```
0     2
1     4
2     6
3     8
4    10
dtype: int64
```

Pandas

- 指定index

```
s = pd.Series(range(5), index=list('abcde'))
```

```
s
```

```
a    0
b    1
c    2
d    3
e    4
dtype: int64
```

```
u.index = list('bcdef')
```

```
u
```

```
b    1
c    2
d    3
e    4
f    5
dtype: int64
```

Pandas

- 指定index

```
s = pd.Series(range(5), index=list('abcde'))
```

```
s
```

```
a    0
b    1
c    2
d    3
e    4
dtype: int64
```

```
u.index = list('bcdef')
```

```
u
```

```
b    1
c    2
d    3
e    4
f    5
dtype: int64
```

Pandas

- 有index時，會有不同的運算結果

```
s+u
```

```
a    NaN
b    2.0
c    4.0
d    6.0
e    8.0
f    NaN
dtype: float64
```

Pandas

- 超好用的DataFrame

```
data = np.random.randn(10, 4)
```

```
df = pd.DataFrame(data)
df
```

	0	1	2	3
0	1.982231	0.620001	0.158268	0.962065
1	-0.400261	-0.576531	0.980601	0.317331
2	0.272691	-0.657567	-0.275869	0.815325
3	0.714238	-1.095269	1.137020	-1.157445
4	0.873165	0.036381	0.190656	-0.254953
5	-0.440756	-0.055469	0.045051	1.969517
6	-0.276417	-0.487820	1.392265	0.160565
7	-1.730829	0.002668	-0.276775	-0.568242
8	-0.332232	-0.791057	0.416602	-0.106758
9	0.160832	-0.565030	0.110545	-0.779896

Pandas

- 設定column

```
df.columns = ['No1', 'No2', 'No3', 'No4']
df
```

	No1	No2	No3	No4
0	1.982231	0.620001	0.158268	0.962065
1	-0.400261	-0.576531	0.980601	0.317331
2	0.272691	-0.657567	-0.275869	0.815325
3	0.714238	-1.095269	1.137020	-1.157445
4	0.873165	0.036381	0.190656	-0.254953
5	-0.440756	-0.055469	0.045051	1.969517
6	-0.276417	-0.487820	1.392265	0.160565
7	-1.730829	0.002668	-0.276775	-0.568242
8	-0.332232	-0.791057	0.416602	-0.106758
9	0.160832	-0.565030	0.110545	-0.779896

Pandas

- 為row加上時間為index

```
df.index = pd.date_range('2016-01-01', periods=10)
df
```

	No1	No2	No3	No4
2016-01-01	1.982231	0.620001	0.158268	0.962065
2016-01-02	-0.400261	-0.576531	0.980601	0.317331
2016-01-03	0.272691	-0.657567	-0.275869	0.815325
2016-01-04	0.714238	-1.095269	1.137020	-1.157445
2016-01-05	0.873165	0.036381	0.190656	-0.254953
2016-01-06	-0.440756	-0.055469	0.045051	1.969517
2016-01-07	-0.276417	-0.487820	1.392265	0.160565
2016-01-08	-1.730829	0.002668	-0.276775	-0.568242
2016-01-09	-0.332232	-0.791057	0.416602	-0.106758
2016-01-10	0.160832	-0.565030	0.110545	-0.779896

Pandas

- 定位並移除資料

```
df.loc['2016-01-06']
```

```
No1   -0.440756
No2   -0.055469
No3    0.045051
No4    1.969517
Name: 2016-01-06 00:00:00, dtype: float64
```

```
df.drop(datetime(2016, 1, 3), inplace=True)
df
```

	No1	No2	No3	No4
2016-01-01	1.982231	0.620001	0.158268	0.962065
2016-01-02	-0.400261	-0.576531	0.980601	0.317331
2016-01-04	0.714238	-1.095269	1.137020	-1.157445
2016-01-05	0.873165	0.036381	0.190656	-0.254953
2016-01-06	-0.440756	-0.055469	0.045051	1.969517
2016-01-07	-0.276417	-0.487820	1.392265	0.160565
2016-01-08	-1.730829	0.002668	-0.276775	-0.568242
2016-01-09	-0.332232	-0.791057	0.416602	-0.106758
2016-01-10	0.160832	-0.565030	0.110545	-0.779896

Pandas

- 重新再設定index為數字

```
df.index=range(9)
df
```

	No1	No2	No3	No4
0	1.982231	0.620001	0.158268	0.962065
1	-0.400261	-0.576531	0.980601	0.317331
2	0.714238	-1.095269	1.137020	-1.157445
3	0.873165	0.036381	0.190656	-0.254953
4	-0.440756	-0.055469	0.045051	1.969517
5	-0.276417	-0.487820	1.392265	0.160565
6	-1.730829	0.002668	-0.276775	-0.568242
7	-0.332232	-0.791057	0.416602	-0.106758
8	0.160832	-0.565030	0.110545	-0.779896

Pandas

- 取一個row
- 刪除一筆資料

```
df.iloc[1]
```

```
No1   -0.400261
No2   -0.576531
No3    0.980601
No4    0.317331
Name: 1, dtype: float64
```

```
df.drop(4)
```

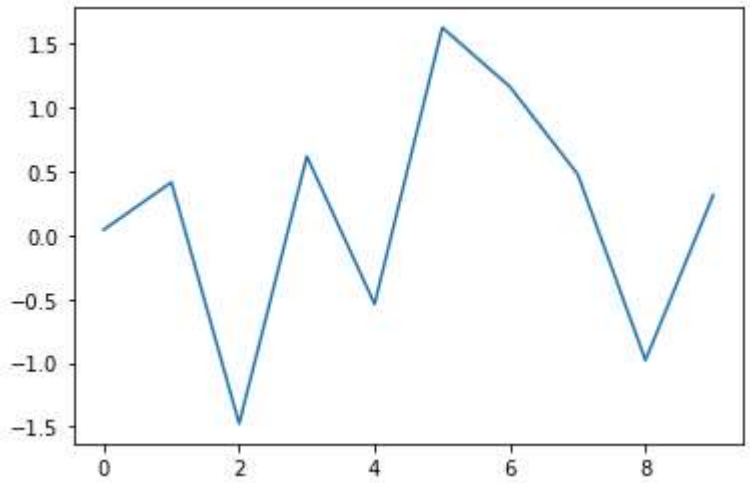
	No1	No2	No3	No4
0	1.982231	0.620001	0.158268	0.962065
1	-0.400261	-0.576531	0.980601	0.317331
2	0.714238	-1.095269	1.137020	-1.157445
3	0.873165	0.036381	0.190656	-0.254953
5	-0.276417	-0.487820	1.392265	0.160565
6	-1.730829	0.002668	-0.276775	-0.568242
7	-0.332232	-0.791057	0.416602	-0.106758
8	0.160832	-0.565030	0.110545	-0.779896

Pandas

- Series，和dataframe都有內建的繪圖函式

```
s = pd.Series(np.random.randn(10), index=np.arange(10))
s.plot()
```

```
<AxesSubplot:>
```

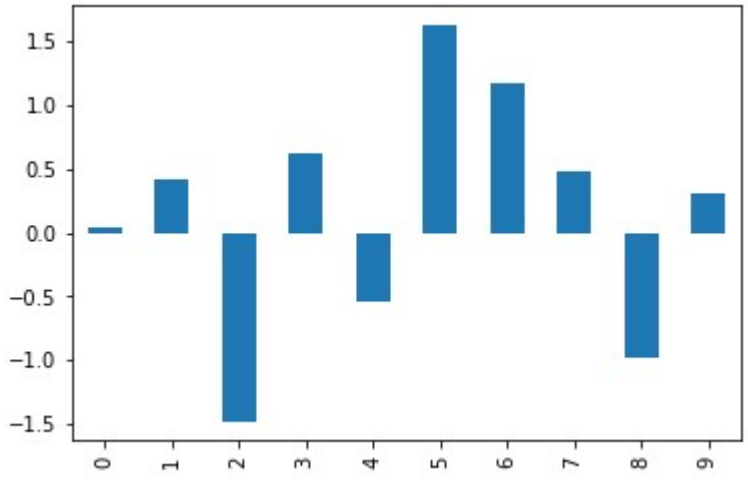


Pandas

- Series，和dataframe都有內建的繪圖函式

```
s.plot(kind="bar")
```

```
<AxesSubplot:>
```

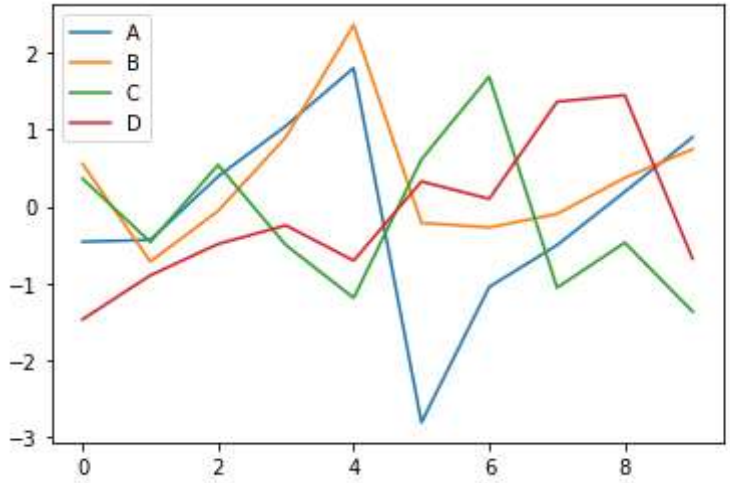


Pandas

- Series，和dataframe都有內建的繪圖函式

```
df = pd.DataFrame(np.random.randn(10, 4), columns=list('ABCD'))
df.plot()
```

```
<AxesSubplot:>
```



Pandas

- Series，和dataframe都有內建的繪圖函式

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

```
<AxesSubplot:>
```