

# AI 프로그래밍

- 2024



4주차

# 오늘 수업 순서

인하공전 컴퓨터 정보과

- 머신 러닝 개념 복습
- 선형 회귀 실습
- Numpy library

# 선형 회귀 예제 실습 - 당뇨병 예제

인하공전 컴퓨터 정보과

특징(10개)

age	sex	bmi	bp	s1	s2	s3	s4	s5	s6
...									

데이터 개수 (442)

혈당
...

bmi

Bmi와 혈당간의 관계 예측

```
diabetes_X, diabetes_y = datasets.load_diabetes(return_X_y=True)
```

```
print('diabetes_X',diabetes_X.shape )
```

# 하나의 특징(BMI)만 추려내서 2차원 배열로 만든다. BMI 특징의 인덱스가 2이다.

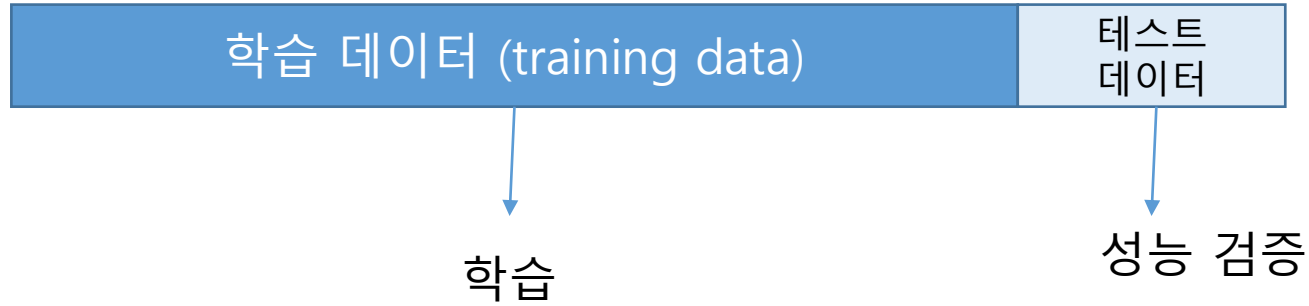
```
diabetes_X_new0 = diabetes_X[:, 2]
```

```
print('diabetes_X_new0',diabetes_X_new0.shape )
```

```
diabetes_X_new = diabetes_X_new0[:, np.newaxis]
```

```
print('diabetes_X_new',diabetes_X_new.shape )
```

```
diabetes_X (442, 10)
diabetes_X_new0 (442,)
diabetes_X_new (442, 1)
```



442개

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(diabetes_X_new, diabetes_y, test_size=0.1, random_state=0)
print('X_train',X_train.shape )
print('X_test',X_test.shape )
print('Y_train',y_train.shape )
print('Y_test',y_test.shape )
```

```
X_train (397, 1)
X_test (45, 1)
Y_train (397,)
Y_test (45,)
```

# 선형 회귀 예제 실습 - 당뇨병 예제

인하공전 컴퓨터 정보과

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn import datasets, linear_model

# 당뇨병 데이터 세트를 적재한다.
diabetes_X, diabetes_y = datasets.load_diabetes(return_X_y=True)

print('diabetes_X,diabetes_X.shape )
# 하나의 특징(BMI)만 추려내서 2차원 배열로 만든다. BMI 특징의 인덱스가 2이다.
diabetes_X_new0 = diabetes_X[:, 2]
print('diabetes_X_new0,diabetes_X_new0.shape )
diabetes_X_new = diabetes_X[:, np.newaxis, 2]
print('diabetes_X_new,diabetes_X_new.shape )

# 학습 데이터와 테스트 데이터를 분리한다.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(diabetes_X_new, diabetes_y, test_size=0.1,
random_state=0)
print('X_train,X_train.shape)
print('X_test,X_test.shape)
print('y_train,y_train.shape)
print('y_test,y_test.shape)

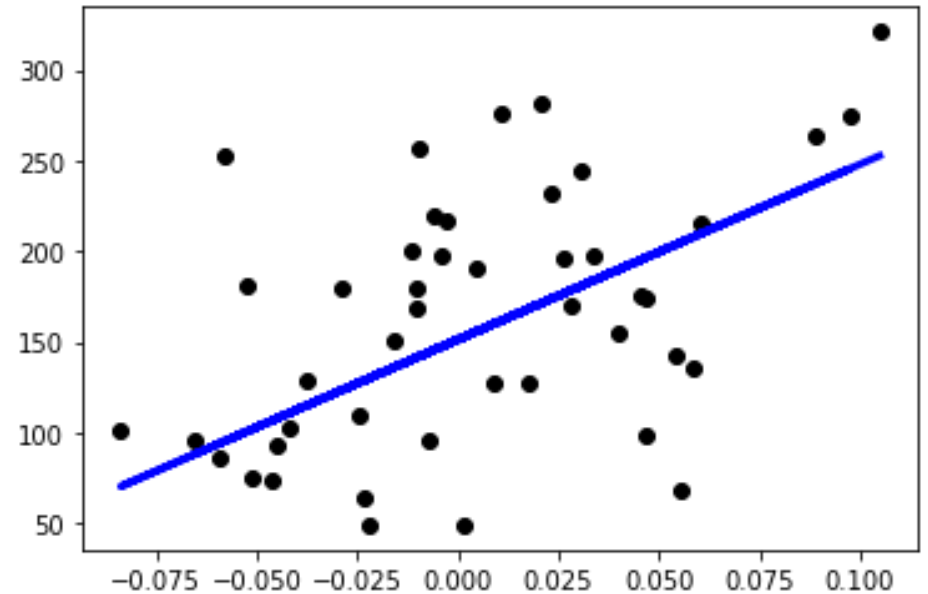
 regr = linear_model.LinearRegression()
 regr.fit(X_train, y_train)

# 테스트 데이터로 예측해보자.
y_pred = regr.predict(X_test)
print(regr.predict([[0.01]])) # bmi가 0.01일때 혈당 예측값

# 실제 데이터와 예측 데이터를 비교해보자.
# plt.plot(y_test, y_pred, '.')

plt.scatter(X_test, y_test, color='black')
plt.plot(X_test, y_pred, color='blue', linewidth=3)
plt.show()
```

과제 4: code : diabetes\_exe.ipynb  
Bmi가 0.025일때의 혈당의 예측 값



- 수치 계산을 위한 라이브러리
- `Import numpy as np`

딥러닝에서 넘파이가 중요한 이유

- 학습 데이터는 2차원 행렬이나 3차원 행렬에 저장된다.

```
>>> import numpy as np

>>> a = np.array([1, 2, 3])
>>> a
array([1, 2, 3])

>>> a[0]
```

1



```
>>> b = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
>>> b
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])

>>> b[0][2]
3
```

b[1,2] ?

b[2,0] ?

# Numpy

```
>>> import numpy as np
>>> a = np.array([[ 0, 1, 2],
                  [ 3, 4, 5],
                  [ 6, 7, 8]])
>>> a.shape           # 배열의 형상
(3, 3)
>>> a.ndim            # 배열의 차원 개수
2
>>> a.dtype           # 요소의 자료형
dtype('int32')
>>> a.itemsize        # 요소 한개의 크기
4                      # 오타
>>> a.size            # 전체 요소의 개수
9
```

```
>>> import numpy as np
>>> a = np.array([[ 0, 1, 2],
                  [ 3, 4, 5],
                  [ 6, 7, 8]])
```

```
>>> np.zeros( (3, 4) )           # (3, 4)는 배열의 형상(행의 개수, 열의 개수)
array([[ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.]])

>>> np.ones((3, 4))
array([[1, 1, 1, 1],
       [1, 1, 1, 1],
       [1, 1, 1, 1]])

>>> np.eye(3)
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

```
>>> import numpy as np
>>> a = np.zeros( (3, 4) )
      a.shape ?
      a.size?
```

# Numpy-arange

```
>>> np.arange(5)
array([0, 1, 2, 3, 4])

>>> np.arange(1, 6)
array([1, 2, 3, 4, 5])

>>> np.arange(1, 10, 2)
array([1, 3, 5, 7, 9])
```

>>> np.arange(1, 11, 2) ?

# Numpy-배열 합치기

인하공전 컴퓨터 정보과

```
>>> x = np.array([[1, 2], [3, 4]])  
>>> y = np.array([[5, 6], [7, 8]])  
  
>>> np.concatenate((x, y), axis=1)  
array([[1, 2, 5, 6],  
       [3, 4, 7, 8]])
```

```
>>> np.vstack((x, y))  
array([[1, 2],  
       [3, 4],  
       [5, 6],  
       [7, 8]])
```

# Numpy-배열 합치기

인하공전 컴퓨터 정보과

```
>>> x = np.array([[1, 2], [3, 4]])
>>> y = np.array([[5, 6], [7, 8]])

>>> np.concatenate((x, y), axis=1)
array([[1, 2, 5, 6],
       [3, 4, 7, 8]])
```

```
>>> np.vstack((x, y))
array([[1, 2],
       [3, 4],
       [5, 6],
       [7, 8]])
```

```
np.hstack((x, y))
```

```
array([[1, 2, 5, 6],
       [3, 4, 7, 8]])
```

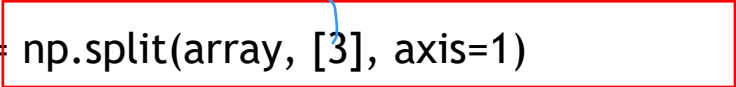
```
>>> a = np.arange(12)
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])

# a에 대하여 reshape(3, 4)를 호출하면 1차원 배열이 2차원 배열로 바뀌게 된다.
>>> a.reshape(3, 4)
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])

>>> a.reshape(6, -1)
array([[ 0,  1],
       [ 2,  3],
       [ 4,  5],
       [ 6,  7],
       [ 8,  9],
       [10, 11]])
```

```
>>> array = np.arange(30).reshape(-1, 10)
>>> array
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, 25, 26, 27, 28, 29]])

>>> arr1, arr2 = np.split(array, [3], axis=1)
>>> arr1
array([[ 0,  1,  2],
       [10, 11, 12],
       [20, 21, 22]])
>>> arr2
array([[ 3,  4,  5,  6,  7,  8,  9],
       [13, 14, 15, 16, 17, 18, 19],
       [23, 24, 25, 26, 27, 28, 29]])
```





```
a = np.array([1, 2, 3, 4, 5, 6])
```

```
a.shape
```

```
(6,)
```

```
a1 = a[np.newaxis, :]
```

```
a1
```

```
array([[1, 2, 3, 4, 5, 6]])
```

```
a1.shape
```

```
(1, 6)
```

```
a2 = a[:, np.newaxis]
```

```
a2.shape
```

```
(6, 1)
```

```
a2
```

```
array([[1],  
       [2],  
       [3],  
       [4],  
       [5],  
       [6]])
```

```
>>> ages = np.array([18, 19, 25, 30, 28])
>>> ages[1:3] # 인덱스 1에서 인덱스 2까지
array([19, 25])
>>> ages[:2] # 인덱스 0에서 인덱스 1까지
array([18, 19])
```

# 논리적인 인덱싱(logical indexing)

```
>>> y = ages > 20
>>> y
array([False, False,  True,  True,  True])
>>> ages[ ages > 20 ]
array([25, 30, 28])
```

# 조건에 맞는 인덱스 찾기

인하공전 컴퓨터 정보과

- `import numpy as np`
- `a=np.array([3, 6, 0, 3, 2, 7, 3, 0, 0, 2])`
- `print(np.where(a!=0))`
- `(array([0, 1, 3, 4, 5, 6, 9]),)`

배열[start : end : step]

-start는 시작 인덱스, end는 끝 인덱스, step은 증가폭/감소폭이다.

-마지막 인덱스 : -1

```
a = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
```

```
print(a[0:3]) => [0 1 2]
```

```
print(a[3:6]) => [3 4 5]
```

```
print(a[1: 8: 2]) => [1 3 5 7]
```

```
print(a[ : : 2]) => [ 0 2 4 6 8 10]
```

```
print(a[3 : 8: 2]) => [3 5 7]
```

```
print(a[3: -1: 2]) => [3 5 7 9]
```

```
print(a[ : :-1]) => [10 9 8 7 6 5 4 3 2 1 0]
```

# Numpy-2차원 배열의 인덱싱

인하공전 컴퓨터 정보과

```
>>> a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
>>> a[0, 2]
3
>>> a[0, 0] = 12
>>> a
array([[12, 2, 3],
       [ 4, 5, 6],
       [ 7, 8, 9]])
```

# Numpy-2차원 배열의 슬라이싱

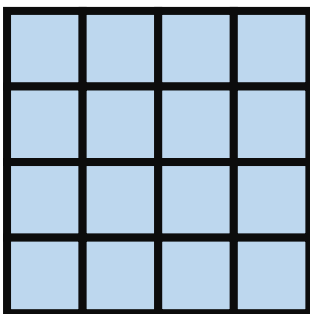
인하공전 컴퓨터 정보과

```
>>> a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])  
>>> a[0:2, 1:3]  
array([[2, 3],  
       [5, 6]])
```

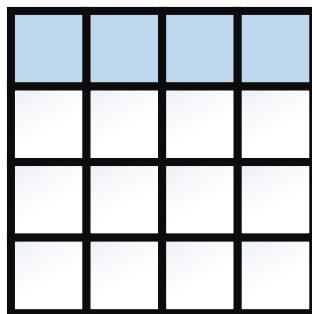
```
a[::2, ::2]
```



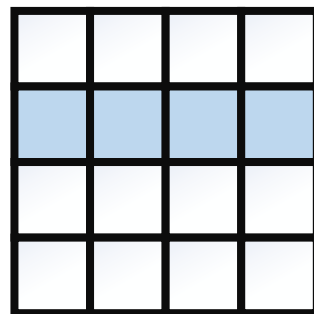
**a**



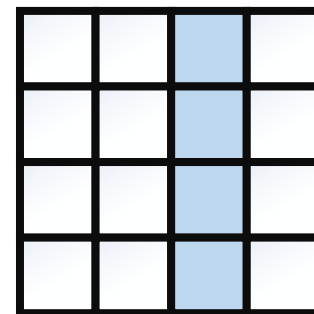
**a[0]**



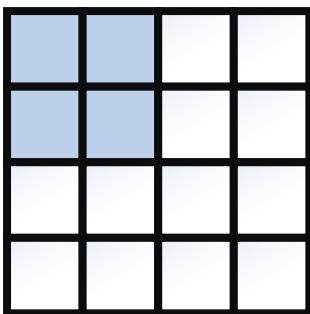
**a[1,:]**



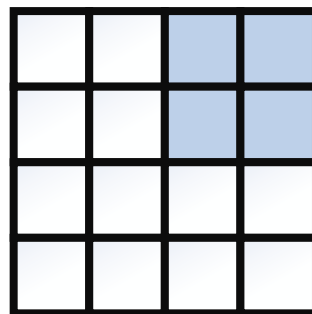
**a[:,2]**



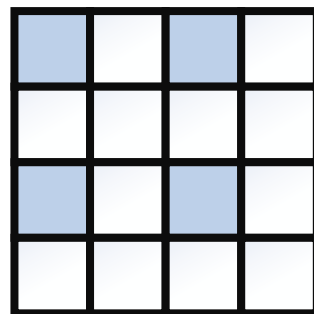
**a[0:2,0:2]**



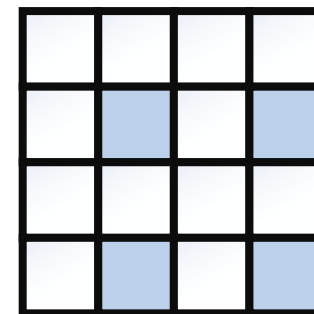
**a[0:2,2:4]**



**a[::2,::2]**



**a[1::2,1::2]**



```
a[::2, ::2]
```

### view

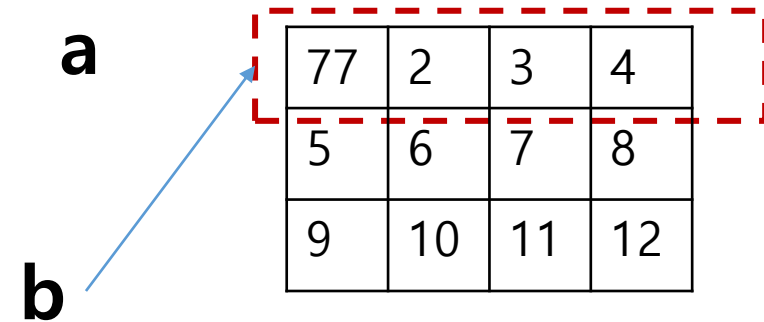
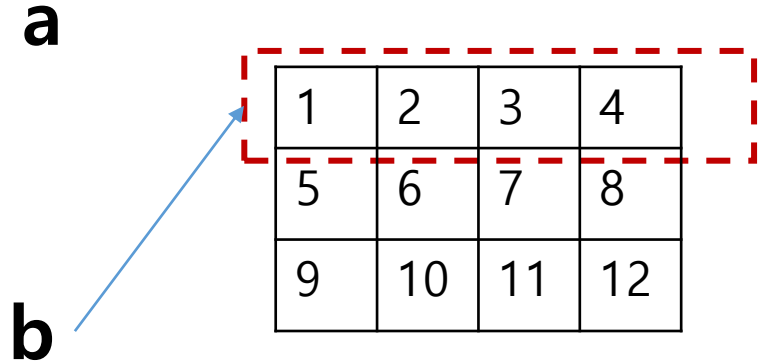
```
>>a=np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12]])  
>>b=a[0,:]
```

**a**

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

## view

```
>>a=np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12]])  
>>b=a[0,:]  
>>b  
array([1,2,3,4])  
b[0]=77  
b  
array([77,2,3,4])  
a  
array ([[77,2,3,4],  
        [5,6,7,8],  
        [9,10,11,12]])
```



## deep copy

```
>>a=np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12]])  
>>b2=a.copy()  
>>b2[0][0]=55  
>>a[0][0] ?
```

**a**

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

**b2**

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

```
>>> arr1 = np.array([[1, 2], [3, 4], [5, 6]])
>>> arr2 = np.array([[1, 1], [1, 1], [1, 1]])
>>> result = arr1 + arr2          # 넘파이 배열에 + 연산이 적용된다.
>>> result
array([[2, 3],
       [4, 5],
       [6, 7]])
```

```
>>> arr1 = np.array([[1, 2], [3, 4], [5, 6]])
>>> arr2 = np.array([[1, 1], [1, 1], [1, 1]])
>>> result = arr1 + arr2          # 넘파이 배열에 + 연산이 적용된다.
>>> result
array([[2, 3],
       [4, 5],
       [6, 7]])
```

```
>>> A = np.array([0, 1, 2, 3])  
>>> 10 * np.sin(A)  
array([0.          , 8.41470985, 9.09297427, 1.41120008])
```



# Numpy-특정한 행과 열을 이용한 연산

인하공전 컴퓨터 정보과

```
>>> scores = np.array([[99, 93, 60], [98, 82, 93],  
...:                   [93, 65, 81], [78, 82, 81]])  
>>> scores.mean(axis=0)  
array([92. , 80.5 , 78.75])
```

```
>>> np.random.seed(100)
>>> np.random.rand(5)
array([0.54340494, 0.27836939, 0.42451759, 0.84477613, 0.00471886])

>>> np.random.rand(5, 3)
array([[0.12156912, 0.67074908, 0.82585276],
       [0.13670659, 0.57509333, 0.89132195],
       [0.20920212, 0.18532822, 0.10837689],
       [0.21969749, 0.97862378, 0.81168315],
       [0.17194101, 0.81622475, 0.27407375]])
```

```
>>> np.random.randn(5)
array([ 0.78148842, -0.65438103,  0.04117247, -0.20191691, -0.87081315])

>>> np.random.randn(5, 4)
array([[ 0.22893207, -0.40803994, -0.10392514,  1.56717879],
       [ 0.49702472,  1.15587233,  1.83861168,  1.53572662],
       [ 0.25499773, -0.84415725, -0.98294346, -0.30609783],
       [ 0.83850061, -1.69084816,  1.15117366, -1.02933685],
       [-0.51099219, -2.36027053,  0.10359513,  1.73881773]])

>>> m, sigma = 10, 2
>>> m + sigma*np.random.randn(5)
array([ 8.56778091, 10.84543531,  9.77559704,  9.09052469,  9.48651379])
```

```
arr=np.array([[1,2],[3,4],[5,6]])  
print(arr.T)
```

```
[[1 3 5]  
 [2 4 6]]
```

```
>>> miles = np.array([1, 2, 3])  
>>> result = miles * 1.6  
>>> result  
array([1.6, 3.2, 4.8])
```

```
>>> arr1 = np.array([[1, 2], [3, 4], [5, 6]])
>>> arr2 = np.array([[2, 2], [2, 2], [2, 2]])
>>> result = arr1 * arr2
>>> result
array([[ 2,  4],
       [ 6,  8],
       [10, 12]])
```

```
>>> arr1 = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
>>> arr2 = np.array([[2, 2], [2, 2], [2, 2]])
>>> result = arr1 @ arr2          # arr1.dot(arr2)로 하여도 된다.
array([[12, 12],
       [30, 30],
       [48, 48]])
```

```
>>> A = np.array([0, 1, 2, 3])  
>>> 10 * np.sin(A)  
array([0.          , 8.41470985, 9.09297427, 1.41120008])
```



```
>>> a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
>>> a.sum()
45
>>> a.min()
1
>>> a.max()
9
```

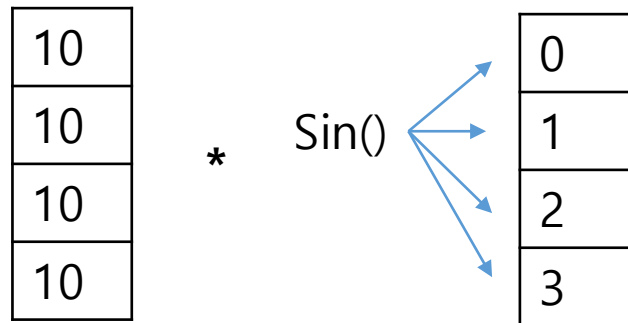
# Numpy-특정한 행과 열을 이용한 연산

인하공전 컴퓨터 정보과

```
>>> scores = np.array([[99, 93, 60], [98, 82, 93],  
...:                   [93, 65, 81], [78, 82, 81]])  
>>> scores.mean(axis=0)  
array([92. , 80.5 , 78.75])
```

```
>>> miles = np.array([1, 2, 3])  
>>> result = miles * 1.6  
>>> result  
array([1.6, 3.2, 4.8])
```

```
>>a=np.array([0,1,2,3])  
>>10*np.sin(A)  
>>array([0.,8.41470985,9.09297427,1.4112008])
```



axis=1 →			
	0	1	2
axis=0 ↓	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9

```
import numpy as np
a = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
a1=a[1:2,0:1]
a2=a[1:2,0]
a3=a[1,0:1]
a4=a[1,0]
print('a1',a1,a1.shape)
print('a2',a2,a2.shape)
print('a3',a3,a3.shape)
print('a4',a4,a4.shape)

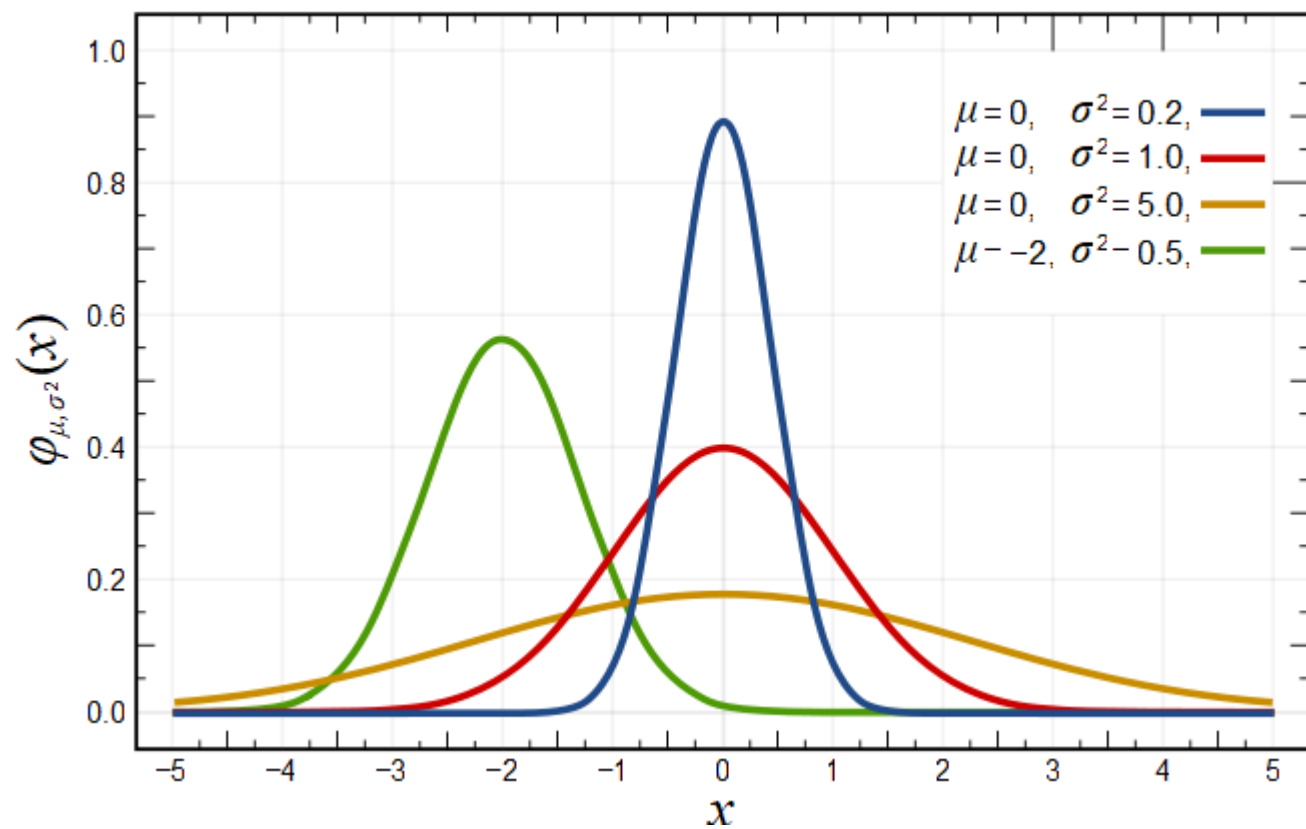
a1 [[4]] (1, 1)
a2 [4] (1,)
a3 [4] (1,)
a4 4 ()
```

```
import numpy as np
a = np.array([1, 2, 3])
a1 = a[np.newaxis, :]
a2 = a[:, np.newaxis]
print('a', a, a.shape)
print('a1', a1, a1.shape)
print('a2', a2, a2.shape)
```

```
a   [1 2 3]           (3,)
a1  [[1 2 3]]        (1, 3)
a2  [[1]
      [2]
      [3]] (3, 1)
```

```
>>> np.random.seed(100)
>>> np.random.rand(5)
array([0.54340494, 0.27836939, 0.42451759, 0.84477613, 0.00471886])

>>> np.random.rand(5, 3)
array([[0.12156912, 0.67074908, 0.82585276],
       [0.13670659, 0.57509333, 0.89132195],
       [0.20920212, 0.18532822, 0.10837689],
       [0.21969749, 0.97862378, 0.81168315],
       [0.17194101, 0.81622475, 0.27407375]])
```





```
>>> np.random.randn(5)
array([ 0.78148842, -0.65438103,  0.04117247, -0.20191691, -0.87081315])

>>> np.random.randn(5, 4)
array([[ 0.22893207, -0.40803994, -0.10392514,  1.56717879],
       [ 0.49702472,  1.15587233,  1.83861168,  1.53572662],
       [ 0.25499773, -0.84415725, -0.98294346, -0.30609783],
       [ 0.83850061, -1.69084816,  1.15117366, -1.02933685],
       [-0.51099219, -2.36027053,  0.10359513,  1.73881773]])

>>> m, sigma = 10, 2
>>> m + sigma*np.random.randn(5)
array([ 8.56778091, 10.84543531,  9.77559704,  9.09052469,  9.48651379])
```

```
import numpy as np
arr=np.array([[1,2],[3,4],[5,6]])
print(arr.T)
```

```
[[1 3 5]
 [2 4 6]]
```

```
x = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
```

```
x.flatten()
```

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

## ■ 그래프를 그리는 라이브러리이다.

```
import matplotlib.pyplot as plt
```

```
X = [ "Mon", "Tue", "Wed", "Thur", "Fri", "Sat", "Sun" ]
```

```
Y1 = [15.6, 14.2, 16.3, 18.2, 17.1, 20.2, 22.4]
```

```
plt.plot(X, Y1, label="Seoul") # 분리시켜서 그려도 됨
```

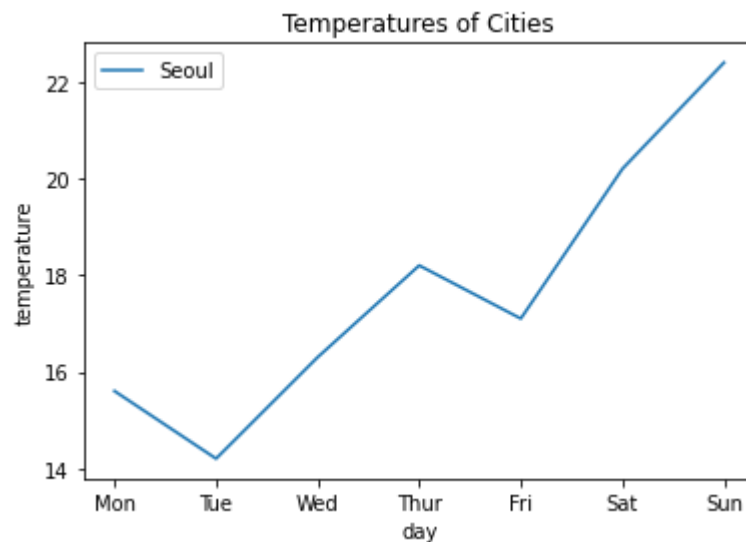
```
plt.xlabel("day")
```

```
plt.ylabel("temperature")
```

```
plt.legend(loc="upper left")
```

```
plt.title("Temperatures of Cities")
```

```
plt.show()
```



- 그래프를 그리는 라이브러리이다.

```
import matplotlib.pyplot as plt
```

```
X = [ "Mon", "Tue", "Wed", "Thur", "Fri", "Sat", "Sun" ]
```

```
Y1 = [15.6, 14.2, 16.3, 18.2, 17.1, 20.2, 22.4]
```

```
Y2 = [20.1, 23.1, 23.8, 25.9, 23.4, 25.1, 26.3]
```

```
plt.plot(X, Y1, label="Seoul")
```

```
plt.plot(X, Y2, label="Busan")
```

```
plt.xlabel("day")
```

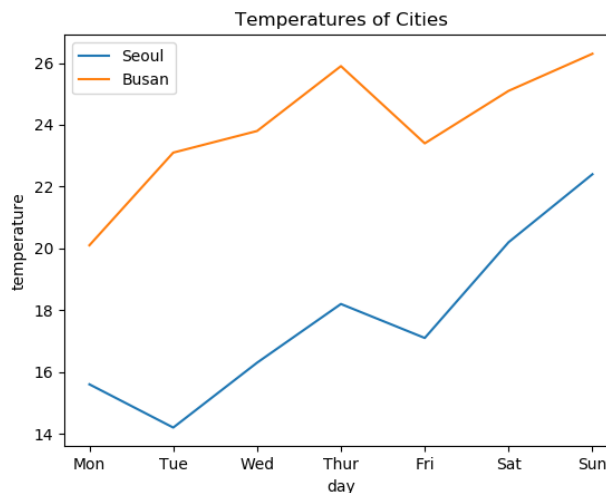
```
plt.ylabel("temperature")
```

```
plt.legend(loc="upper left")
```

```
plt.title("Temperatures of Cities")
```

```
plt.show()
```

# 분리시켜서 그려도 됨  
# 분리시켜서 그려도 됨



# 매트플롯

## 마커속성

character	description
' - '	solid line style
' - - '	dashed line style
' - . '	dash-dot line style
' : '	dotted line style

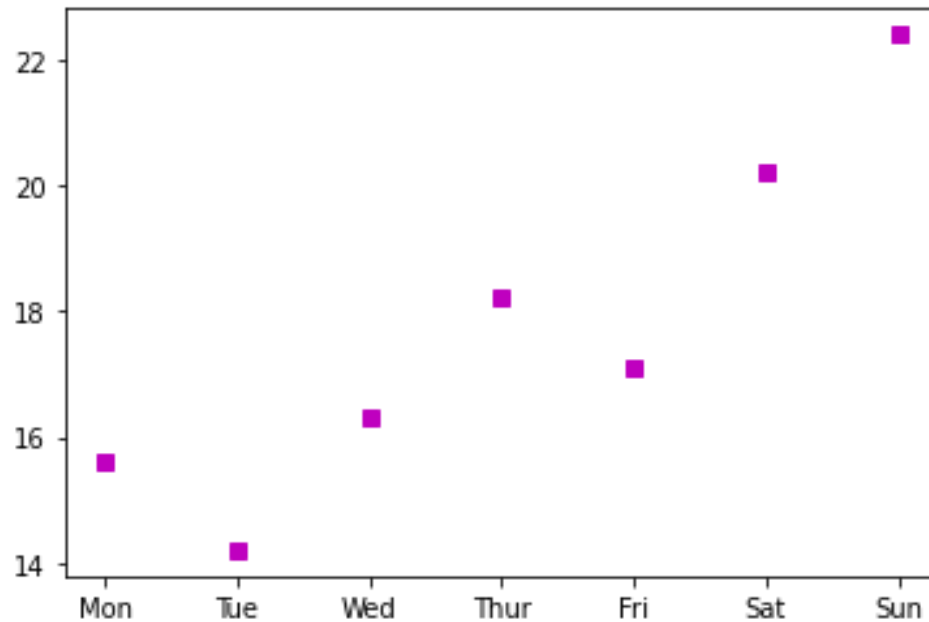
' . '	point marker
' , '	pixel marker
' o '	circle marker
' v '	triangle_down marker
' ^ '	triangle_up marker
' < '	triangle_left marker
' > '	triangle_right marker
' 1 '	tri_down marker
' 2 '	tri_up marker
' 3 '	tri_left marker
' 4 '	tri_right marker
' s '	square marker
' p '	pentagon marker
' * '	star marker
' h '	hexagon1 marker
' H '	hexagon2 marker
' + '	plus marker
' x '	x marker
' D '	diamond marker
' d '	thin_diamond

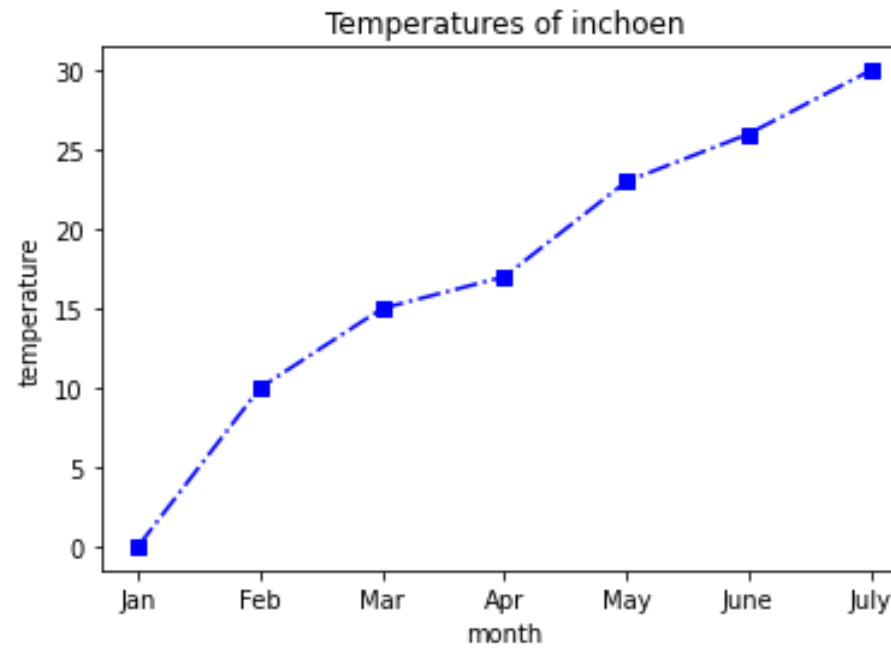
character	color
' b '	blue
' g '	green
' r '	red
' c '	cyan
' m '	magenta
' y '	yellow
' k '	black
' w '	white

"\_4c"

```
import matplotlib.pyplot as plt  
%matplotlib inline
```

```
X = [ "Mon", "Tue", "Wed", "Thur", "Fri", "Sat", "Sun" ]  
plt.plot(X, [15.6, 14.2, 16.3, 18.2, 17.1, 20.2, 22.4], "sm")  
plt.show()
```







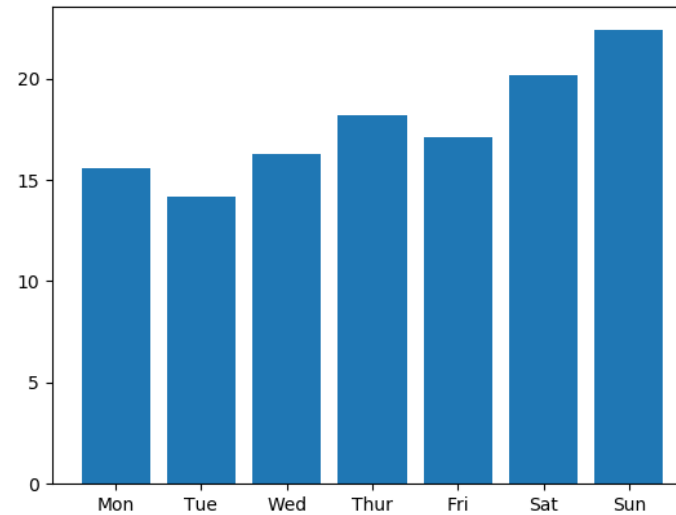
```
import matplotlib.pyplot as plt
```

```
X = [ "Mon", "Tue", "Wed", "Thur", "Fri", "Sat", "Sun" ]
```

```
Y = [15.6, 14.2, 16.3, 18.2, 17.1, 20.2, 22.4]
```

```
plt.bar(X, Y)
```

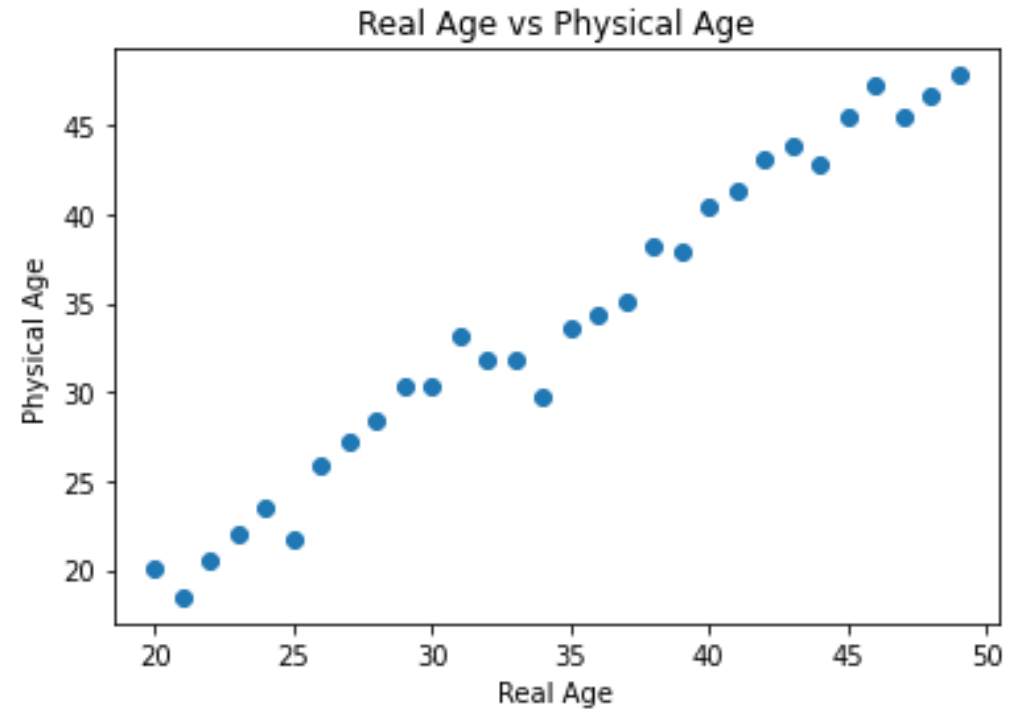
```
plt.show()
```

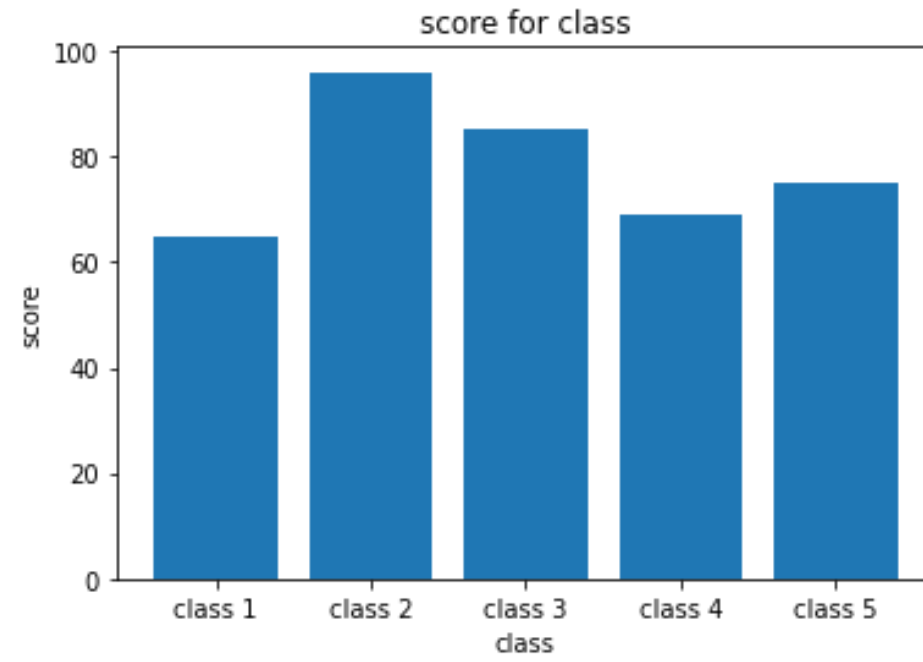


```
import matplotlib.pyplot as plt
import numpy as np

xData=np.arange(20,50)
yData=xData+2*np.random.randn(30)

plt.scatter(xData,yData)
plt.title('Real Age vs Physical Age')
plt.xlabel('Real Age')
plt.ylabel('Physical Age')
plt.show()
```





# 수고하셨습니다

---

[jhmin@inhatech.ac.kr](mailto:jhmin@inhatech.ac.kr)