

5주차(2/3)

기계학습 작업 흐름 2

파이썬으로 배우는 기계학습

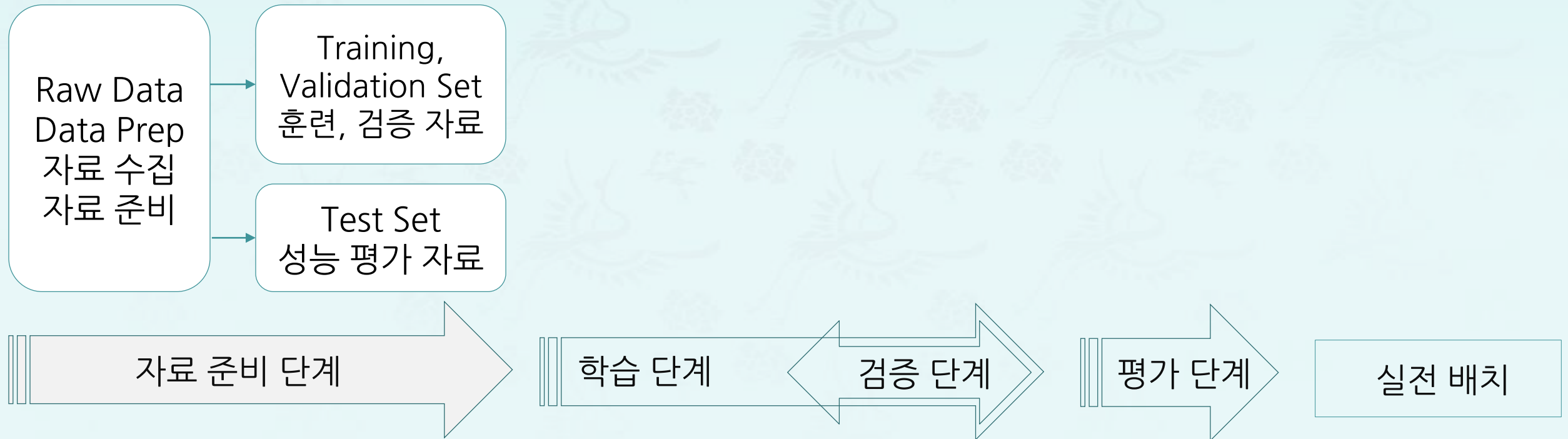
한동대학교
김영섭 교수

기계학습 작업 흐름 2

- 학습 목표
 - 기계학습의 전반적 작업의 흐름을 이해한다.
- 학습 내용
 - 학습 자료 준비
 - 학습 자료 전처리
 - 오차/정확도 측정

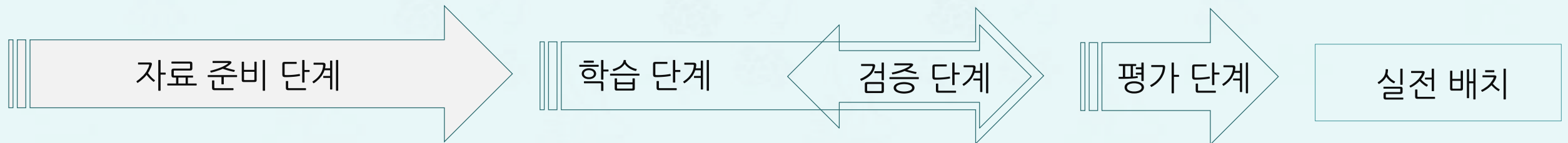
1. 학습 자료의 준비 단계: 자료읽기

훈련 자료 Training Set		평가 자료 Test Set
훈련 자료 Training	검증 자료 Validation	



1. 학습 자료의 준비 단계: 자료읽기

- 학습 자료 준비
 - **joydata.txt**



1. 학습 자료의 준비 단계: 자료읽기

- 학습 자료 준비
 - joydata.txt

```
!cat data/joydata.txt
```

-1.72	-3.12	1
0.31	1.85	1
1.56	2.85	1
2.64	2.41	1
1.23	2.54	1
1.33	2.03	1
1.26	2.68	1
2.58	1.79	1
2.40	0.91	1
0.51	2.44	1

2. 학습 자료 전처리: 자료의 편향성

- 자료 전처리

2. 학습 자료 전처리: 자료의 편향성

- 자료 전처리

```
import numpy as np
data = np.genfromtxt('data/joydata.txt')
x, y = data[:, :2], data[:, 2]
y = y.astype(np.int)
print(x[:5], y[:5])
print(x[-5:], y[-5:])
```

```
[[ -1.72  -3.12]
 [  0.31   1.85]
 [  1.56   2.85]
 [  2.64   2.41]
 [  1.23   2.54]] [1 1 1 1 1]
[[-2.26   0.01]
 [-1.41  -0.23]
 [-1.2   -0.71]
 [-1.69   0.7 ]
 [-1.52  -1.14]] [0 0 0 0 0]
```

2. 학습 자료 전처리: 자료의 편향성

- 자료 전처리

```
import numpy as np
data = np.genfromtxt('data/joydata.txt')
x, y = data[:, :2], data[:, 2]
y = y.astype(np.int)
print(x[:5], y[:5])
print(x[-5:], y[-5:])
```

```
[[ -1.72  -3.12]
 [  0.31   1.85]
 [  1.56   2.85]
 [  2.64   2.41]
 [  1.23   2.54]] [1 1 1 1 1]
[[-2.26   0.01]
 [-1.41  -0.23]
 [-1.2   -0.71]
 [-1.69   0.7 ]
 [-1.52  -1.14]] [0 0 0 0 0]
```


2. 학습 자료 전처리: Shuffling

- 자료 전처리
 - 셔플링(shuffling)


```
import numpy as np
data = np.genfromtxt('data/joydata.txt')
x, y = data[:, :2], data[:, 2]
y = y.astype(np.int)
print(x[:5], y[:5])
print(x[-5:], y[-5:])
```

```
[[ -1.72  -3.12]
 [  0.31   1.85]
 [  1.56   2.85]
 [  2.64   2.41]
 [  1.23   2.54]] [1 1 1 1 1]
[[ -2.26   0.01]
 [ -1.41  -0.23]
 [ -1.2   -0.71]
 [ -1.69   0.7 ]
 [ -1.52  -1.14]] [0 0 0 0 0]
```

2. 학습 자료 전처리: Shuffling

- 자료 전처리
 - 셔플링(shuffling)

```
import numpy as np
data = np.genfromtxt('data/joydata.txt')
x, y = data[:, :2], data[:, 2]
y = y.astype(np.int)
np.random.shuffle(x)
np.random.shuffle(y)
print(x[:5], y[:5])
print(x[-5:], y[-5:])
```




```
[[-5.27 -1.78]
 [ 1.33  2.03]
 [ 1.    0.46]
 [-1.48 -1.17]
 [ 1.14  3.01]] [1 0 1 1 0]
[[-3.45 -0.62]
 [-1.26 -2.9 ]
 [ 1.9   1.34]
 [-1.08 -1.23]
 [ 2.52  1.83]] [1 1 0 1 0]
```

2. 학습 자료 전처리: Shuffling

- 자료 전처리
 - 셔플링(shuffling)

```
import numpy as np
data = np.genfromtxt('data/joydata.txt')
x, y = data[:, :2], data[:, 2]
y = y.astype(np.int)
np.random.shuffle(x)
np.random.shuffle(y)
print(x[:5], y[:5])
print(x[-5:], y[-5:])
```




```
[[-5.27 -1.78]
 [ 1.33  2.03]
 [ 1.     0.46]
 [-1.48 -1.17]
 [ 1.14  3.01]] [1 0 1 1 0]
[[-3.45 -0.62]
 [-1.26 -2.9 ]
 [ 1.9   1.34]
 [-1.08 -1.23]
 [ 2.52  1.83]] [1 1 0 1 0]
```

2. 학습 자료 전처리: Shuffling

- 자료 전처리
 - 셔플링(shuffling)

```
import numpy as np
data = np.genfromtxt('data/joydata.txt')
np.random.shuffle(data)
x, y = data[:, :2], data[:, 2]
y = y.astype(np.int)
print(x[:5], y[:5])
print(x[-5:], y[-5:])
```



```
[[ 0.68  1.43]
 [-3.07 -2.09]
 [ 3.87  2.91]
 [-1.2   -0.71]
 [-3.08 -1.05]] [1 0 1 0 0]
[[-1.41 -0.23]
 [ 1.26  2.68]
 [ 1.9   1.34]
 [ 0.9   2.05]
 [ 1.26  1.17]] [0 1 1 1 1]
```

2. 학습 자료 전처리: Feature Scaling의 종류

- 자료 전처리
 - 셔플링(shuffling)
 - 피쳐 스케일링(feature scaling)
 - 정규화(normalization)
 - 표준화(standardization)

2. 학습 자료 전처리: 정규화

- 자료 전처리
 - 셔플링(shuffling)
 - 피쳐 스케일링(feature scaling)
 - 정규화(normalization)
 - min-max scaling
 - 자료 범위: 0 부터 1사이
 - 계산 방법:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

2. 학습 자료 전처리: 표준화

■ 자료 전처리


- 셔플링(shuffling)
- 피쳐 스케일링(feature scaling)
 - 정규화(normalization)

- min-max scaling
- 자료 범위: 0 부터 1 사이
- 계산 방법:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

■ 표준화(standardization)

- 특이값에 영향을 덜 받음
- 자료 범위: 평균값 0, 표준편차 1
- 계산 방법:


$$\mathbf{x}_j := \frac{\mathbf{x}_j - \mu_j}{\sigma_j}$$
$$\mu_j = \frac{1}{m} \sum_{i=1}^m x_j^{(i)}$$
$$\sigma_j^2 = \frac{1}{m} \sum_{i=1}^m (x_j^{(i)} - \mu_j)^2$$

- x_j : 입력 \mathbf{x} 의 j 번째 특성
- μ_j : 입력 \mathbf{x} 의 j 번째 특성의 평균값
- σ_j : 입력 \mathbf{x} 의 j 번째 특성의 표준편차

2. 학습 자료 전처리: 정규화 코드

- 정규화(normalization) 코딩

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- 표준화(standardization) 코딩

```
xmax = np.max(x)
xmin = np.min(x)
x = (x - xmin)/(xmax - xmin)
```


2. 학습 자료 전처리: 표준화 코드

- 정규화(normalization) 코딩

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- 표준화(standardization) 코딩

$$\begin{aligned} \mathbf{x}_j &:= \frac{\mathbf{x}_j - \mu_j}{\sigma_j} \\ \mu_j &= \frac{1}{m} \sum_{i=1}^m x_j^{(i)} \\ \sigma_j^2 &= \frac{1}{m} \sum_{i=1}^m (x_j^{(i)} - \mu_j)^2 \end{aligned}$$

```
xmax = np.max(x)
xmin = np.min(x)
x = (x - xmin)/(xmax - xmin)
```


```
mu = x.mean(axis=0)
sigma = x.std(axis=0)
x = (x - mu) / sigma
```

2. 학습 자료 전처리: 자료분리

- 자료 전처리
 - 셔플링(shuffling)
 - 피쳐 스케일링(feature scaling)
 - 정규화(normalization)
 - 표준화(standardization)
- 자료 분리
 - 훈련 자료 vs 테스트 자료
 - 8:2 혹은 7:3

2. 학습 자료 전처리: 자료분리

- 자료 전처리
 - 셔플링(shuffling)
 - 피쳐 스케일링(feature scaling)
 - 정규화(normalization)
 - 표준화(standardization)
- 자료 분리
 - 훈련 자료 vs 테스트 자료
 - 8:2 혹은 7:3

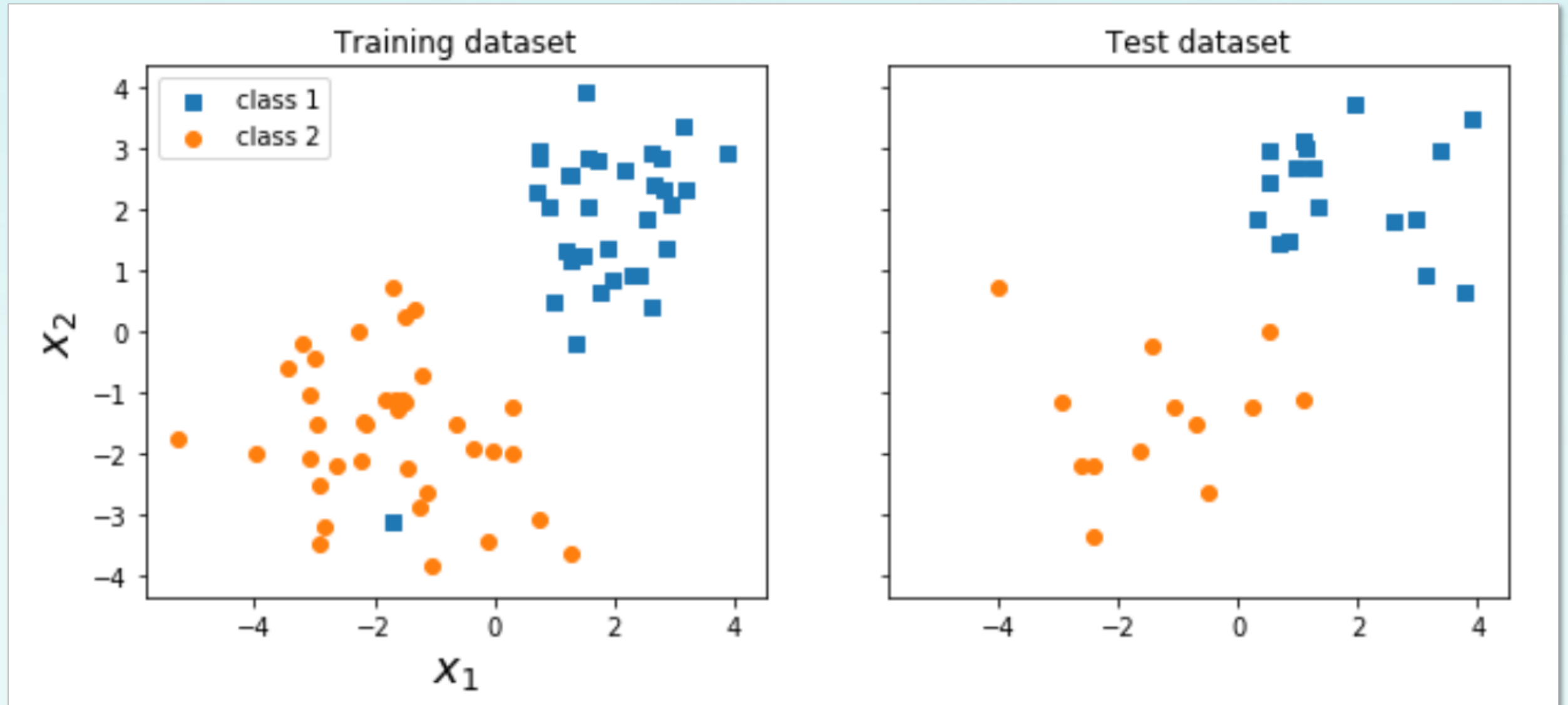


```
1 import numpy as np
2 data = np.genfromtxt('data/joydata.txt')
3 np.random.seed(1)
4 np.random.shuffle(data)
5 x, y = data[:, :2], data[:, 2]
6 y = y.astype(np.int)
7
8 num = int(x.shape[0] * 0.8) ## percentage
9 x_train, x_test = x[:num], x[num:]
10 y_train, y_test = y[:num], y[num:]
```

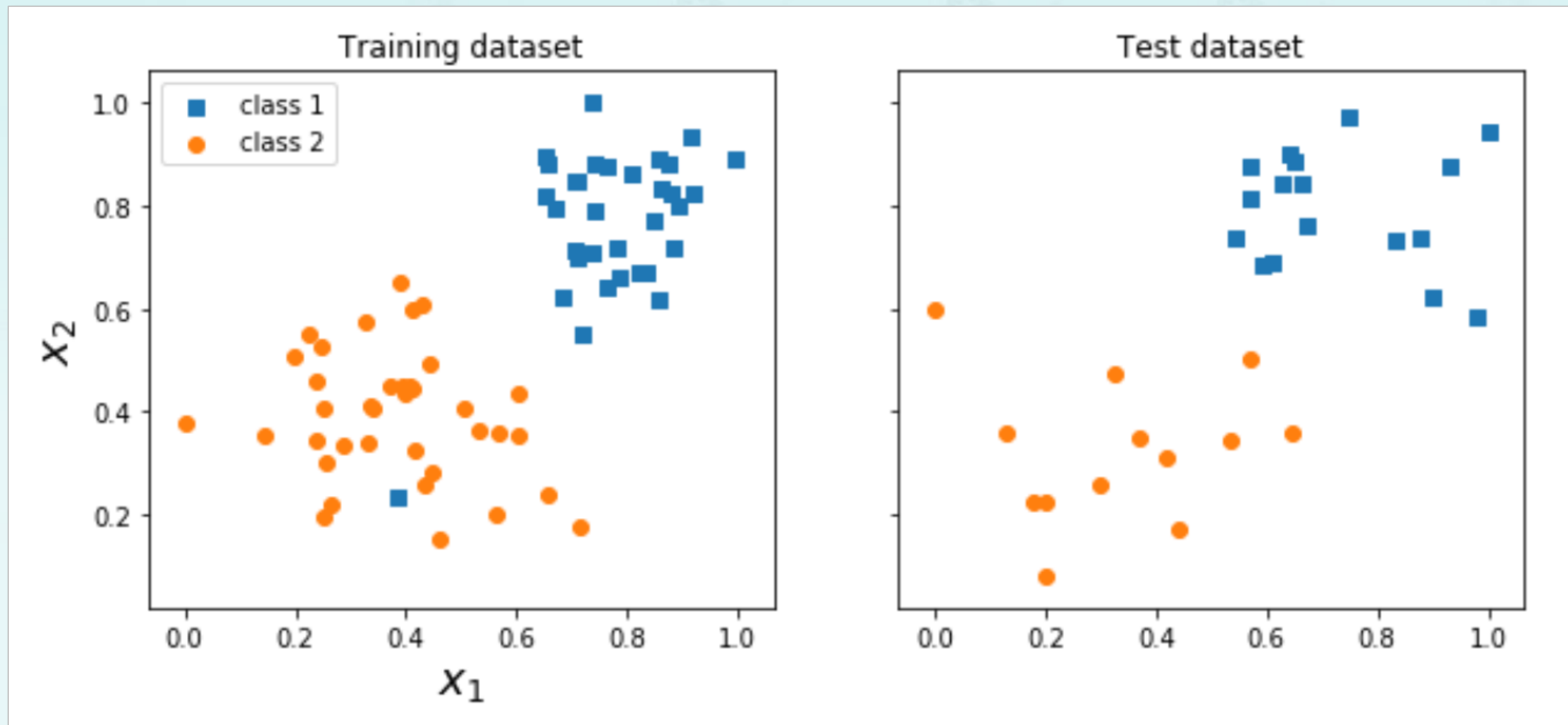
2. 학습 자료 전처리: 자료분리

- 자료 전처리
 - 셔플링(shuffling)
 - 피쳐 스케일링(feature scaling)
 - 정규화(normalization)
 - 표준화(standardization)
- 자료 분리
 - 훈련 자료 vs 테스트 자료
 - 8:2 혹은 7:3

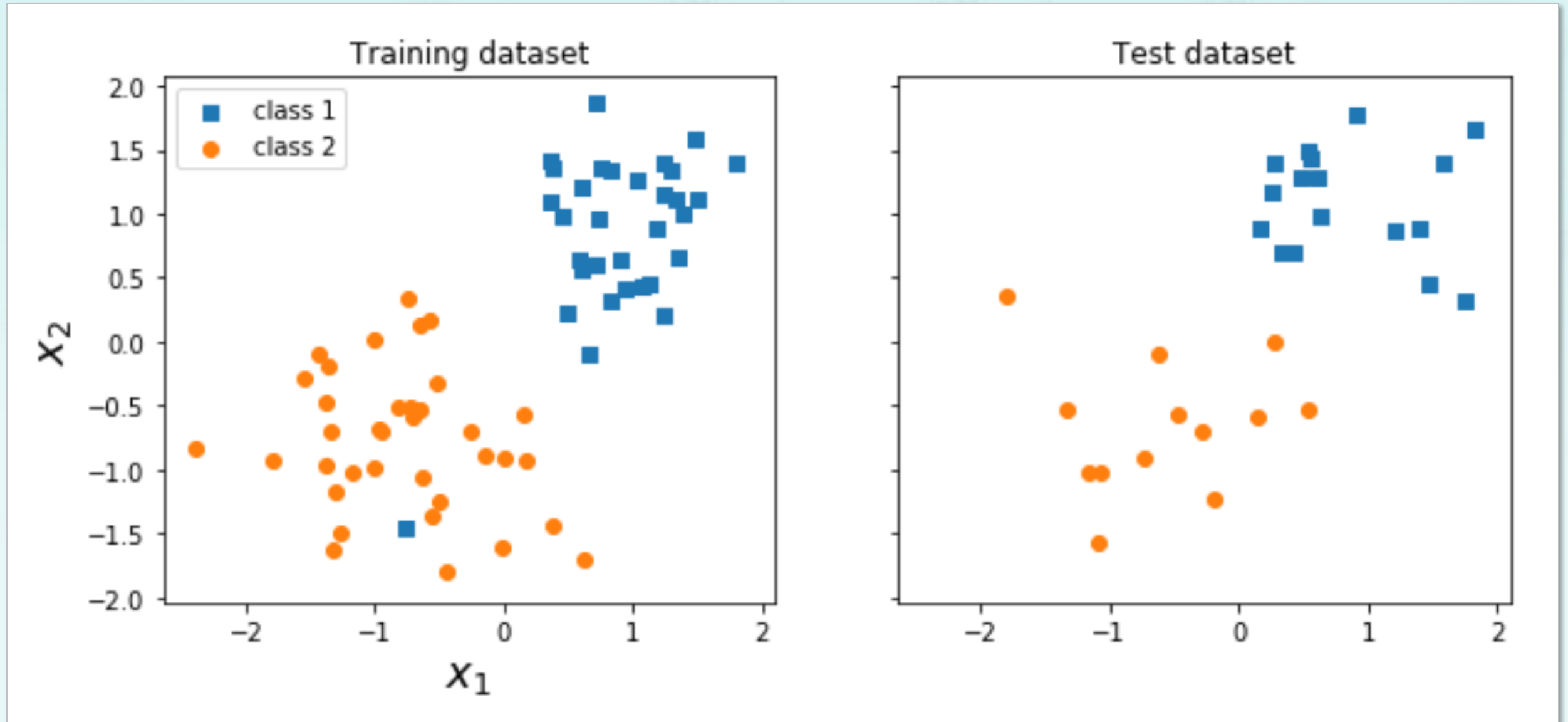
2. 학습 자료 전처리: 훈련자료 시각화(Shuffling)



2. 학습 자료 전처리: 훈련자료 시각화(정규화)



2. 학습 자료 전처리: 훈련자료 시각화(표준화)



2. 학습 자료 전처리: 편향을 포함하는 자료 구조

- 편향을 포함한 자료 구조
- 편향을 포함하지 않은 자료 구조

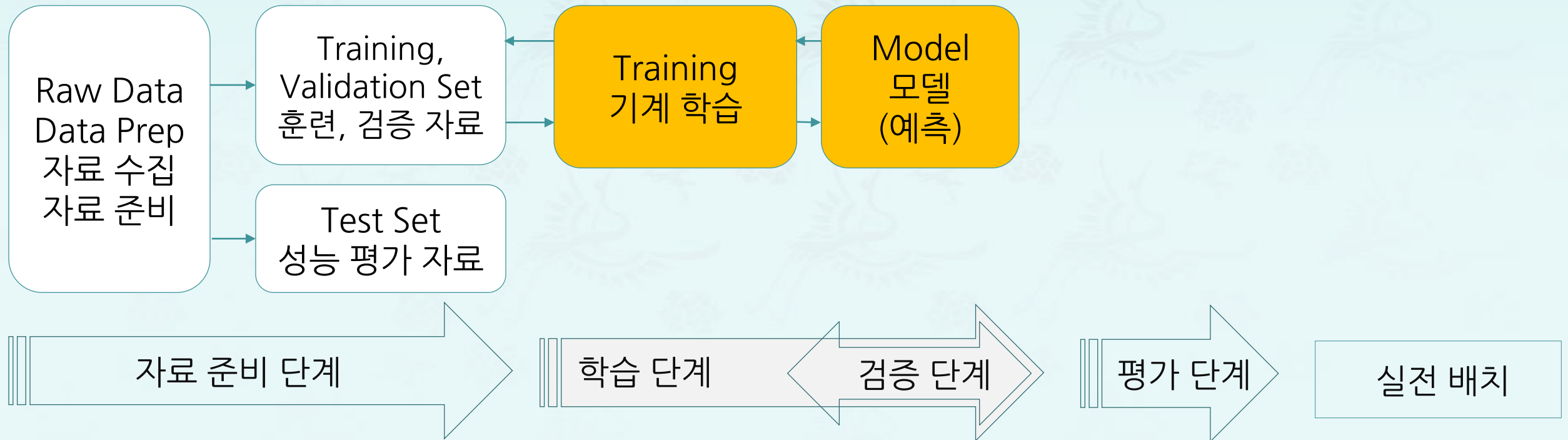
$$\begin{aligned} z &= \mathbf{w}^T \mathbf{x} \\ &= w_0 x_0 + w_1 x_1 + \dots + w_n x_n \\ &= \sum_{j=0}^n x_j w_j \end{aligned}$$

방법 1

$$\begin{aligned} z &= \mathbf{w}^T \mathbf{x} + b \\ &= w_1 x_1 + w_2 x_2 + \dots + w_n x_n \\ &= \sum_{j=1}^n x_j w_j + b \end{aligned}$$

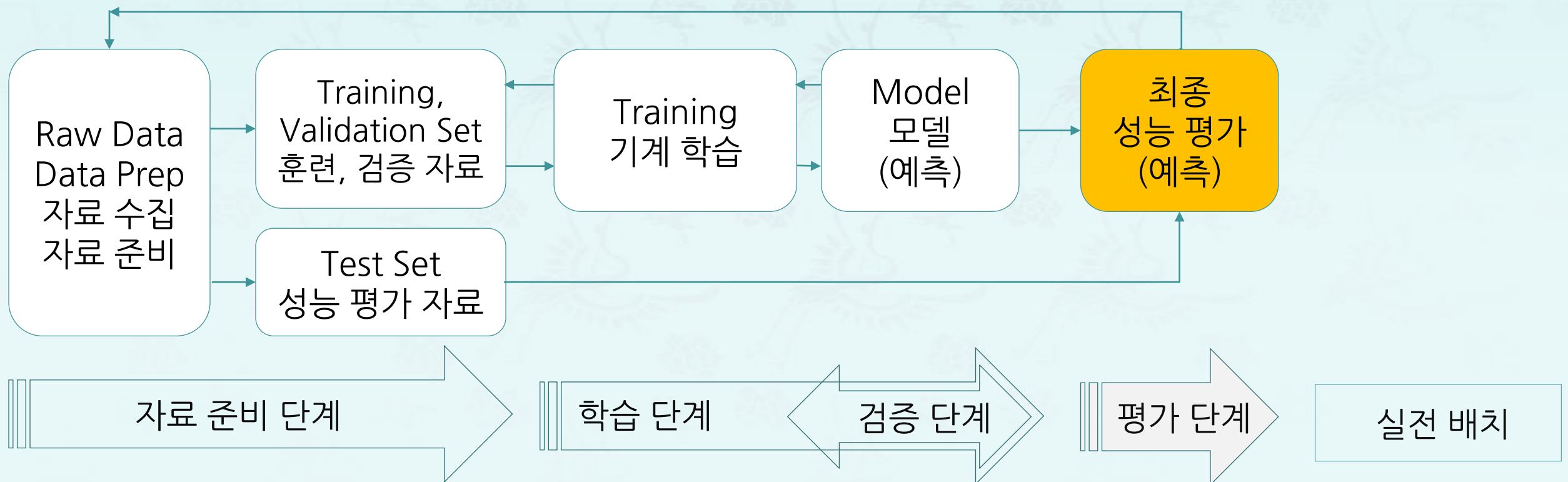
방법 2

3. 예측 및 평가 단계



3. 예측 및 평가 단계

1. 자료 준비 단계: 수집과 전처리
2. 학습 단계: 모델 훈련과 완성
3. 검증 단계: 하이퍼 파라미터 조정
4. 평가 단계: 최종 실전 배치 여부 결정



3. 예측 및 평가 단계: 이진분류 퍼셉트론 예측

- 함수 **perceptron()** 분별기
- 예측 함수
 - **perceptron_predict(X, w)**
 - **X**: 예측할 입력 자료
(검증자료, 테스트 자료)
 - **w**: 가중치
 - **yhat**: 함수 반환 값
 - 형상: (m samples, 1)
 - 형식: 0, 1

3. 예측 및 평가 단계: 이진분류 퍼셉트론 예측

- 함수 **perceptron()** 분별기
- 예측 함수
 - **perceptron_predict(X, w)**
 - **X**: 예측할 입력 자료
(검증자료, 테스트 자료)
 - **w**: 가중치
 - **yhat**: 함수 반환 값
 - 형상: (m samples, 1)
 - 형식: 0, 1

```
1 def perceptron_predict(X, w):  
2     z = np.dot(X, w)  
3     yhat = np.where(z > 0., 1, 0)  
4     return yhat
```

3. 예측 및 평가 단계: 이진분류 퍼셉트론 평가

- 평가 단계
 - **y**: 클래스 레이블(주어진 값, 정답)
 - **yhat**: 예측값

```
1 def perceptron_predict(X, w):  
2     z = np.dot(X, w)  
3     yhat = np.where(z > 0., 1, 0)  
4     return yhat
```

3. 예측 및 평가 단계: 이진분류 퍼셉트론 평가

- 평가 단계
 - **y**: 클래스 레이블(주어진 값, 정답)
 - **yhat**: 예측값

```
1 def perceptron_predict(X, w):  
2     z = np.dot(X, w)  
3     yhat = np.where(z > 0., 1, 0)  
4     return yhat
```

```
1 #version 0.1  
2 yhat = perceptron_predict(X_train, w)  
3 missed = 0 # misclassified count  
4 m_samples = len(y_train)  
5 for m in range(m_samples):  
6     if yhat[m] != y_train[m]:  
7         missed += 1  
8 print('Misclassified:{}/{}'.  
9       format(missed, m_samples))
```

Misclassified:1/80

3. 예측 및 평가 단계: 이진분류 퍼셉트론 평가

- 평가 단계
 - **y**: 클래스 레이블(주어진 값, 정답)
 - **yhat**: 예측값

```
1 def perceptron_predict(X, w):  
2     z = np.dot(X, w)  
3     yhat = np.where(z > 0., 1, 0)  
4     return yhat
```


```
1 #version 0.1  
2 yhat = perceptron_predict(X_train, w)  
3 missed = 0 # misclassified count  
4 m_samples = len(y_train)  
5 for m in range(m_samples):  
6     if yhat[m] != y_train[m]:  
7         missed += 1  
8 print('Misclassified:{}/{}'.  
9       format(missed, m_samples))
```

Misclassified:1/80

3. 예측 및 평가 단계: 이진분류 퍼셉트론 평가


- 평가 단계
 - **y**: 클래스 레이블(주어진 값, 정답)
 - **yhat**: 예측값

```
1 def perceptron_predict(X, w):  
2     z = np.dot(X, w)  
3     yhat = np.where(z > 0., 1, 0)  
4     return yhat
```



```
1 #version 0.1  
2 yhat = perceptron_predict(X_test, w)  
3 missed = 0  
4 m_samples = len(y_test)  
5 for m in range(m_samples):  
6     if yhat[m] != y_test[m]:  
7         missed += 1  
8 print('Misclassified: {}/{}'.  
9       format(missed, m_samples))
```

Misclassified:1/20



```
1 #version 0.1  
2 yhat = perceptron_predict(X_train, w)  
3 missed = 0 # misclassified count  
4 m_samples = len(y_train)  
5 for m in range(m_samples):  
6     if yhat[m] != y_train[m]:  
7         missed += 1  
8 print('Misclassified: {}/{}'.  
9       format(missed, m_samples))
```

Misclassified:1/80

3. 예측 및 평가 단계: 이진분류 퍼셉트론 평가

- 평가 단계
 - **y**: 클래스 레이블(주어진 값, 정답)
 - **yhat**: 예측값

```
1 #version 0.1
2 yhat = perceptron_predict(X_test, w)
3 missed = 0
4 m_samples = len(y_test)
5 for m in range(m_samples):
6     if yhat[m] != y_test[m]:
7         missed += 1
8 print('Misclassified: {}/{}'.format(missed, m_samples))
9
```

Misclassified:1/20



```
#version 0.2
yhat = perceptron_predict(X_test, w)
missed = np.sum(yhat.flatten() != y_test)
print('Misclassified: {}/{}'.format(missed, m_samples))
```

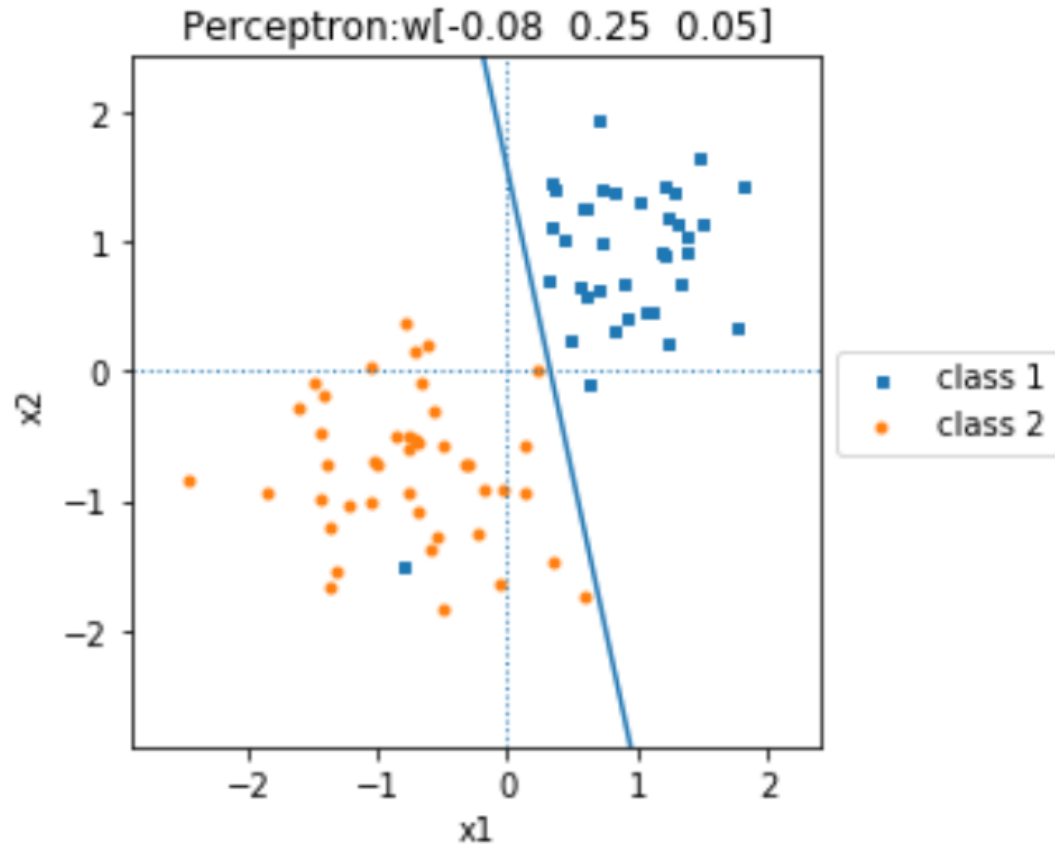
Misclassified:1/20

3. 예측 및 평가 단계: 시각화

- 훈련 자료 분류 결과

- 테스트 자료 분류 결과

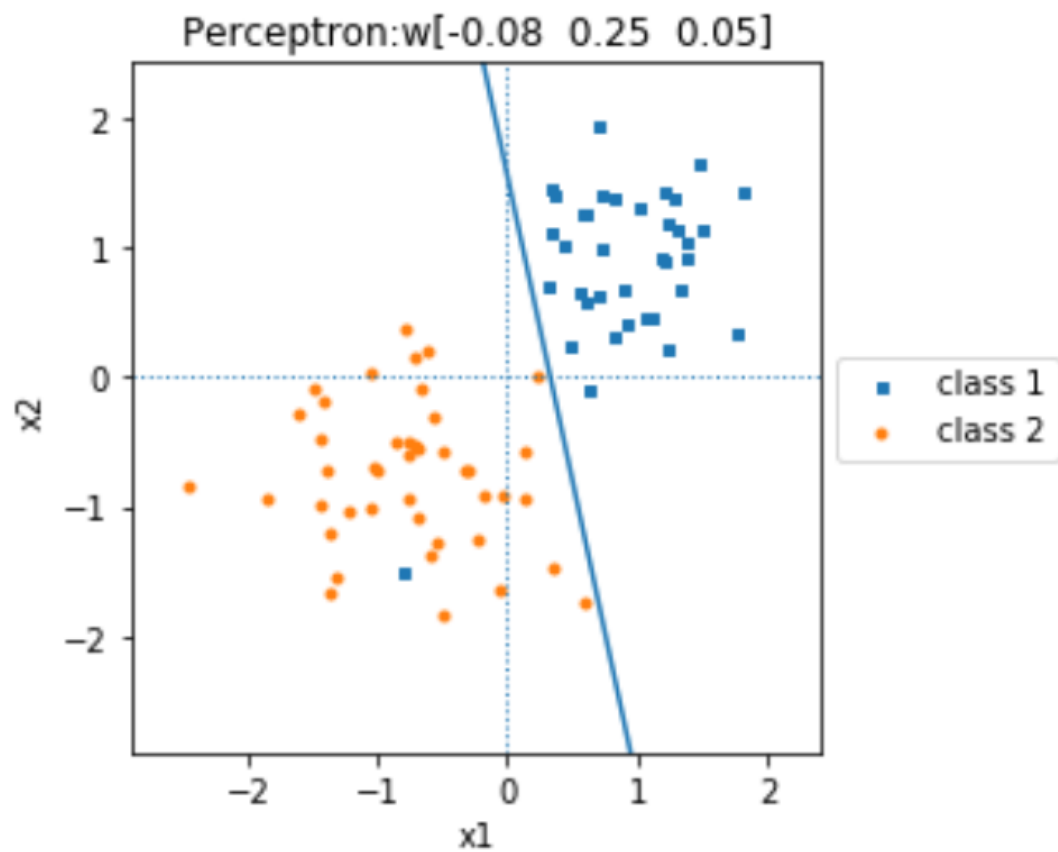
```
%run code/plot_xyw.py  
plot_xyw(X_train, y_train, w.flatten(), X0=True)
```



3. 예측 및 평가 단계: 시각화

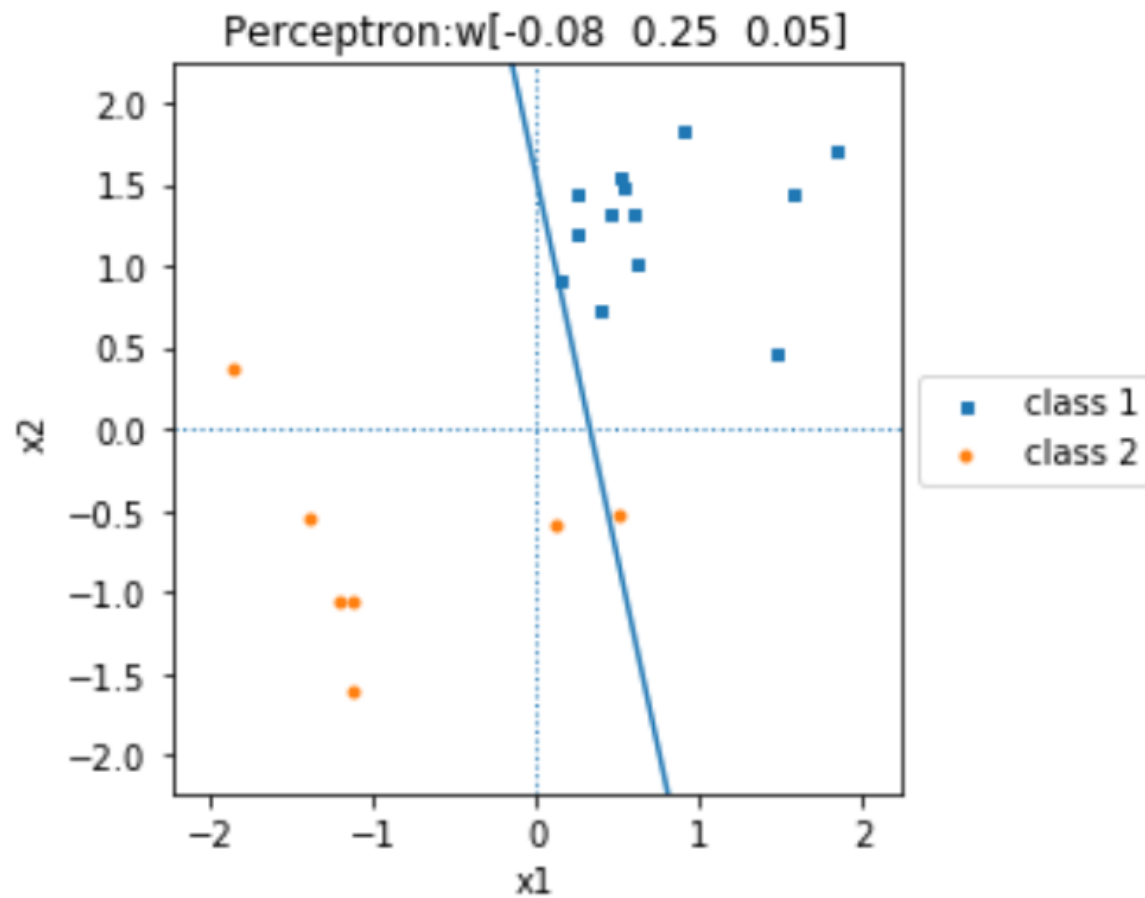
- 훈련 자료 분류 결과

```
%run code/plot_xyw.py  
plot_xyw(X_train, y_train, w.flatten(), X0=True)
```



- 테스트 자료 분류 결과

```
plot_xyw(X_test, y_test, w.flatten(), X0=True)
```



기계학습 작업 흐름 2

- 학습 정리
 - 기계학습을 적용하는데 필요한 작업 흐름도를 이해하기
 - 자료 준비와 전처리 과정 필요성과 역할
 - 모델의 정확도 평가
- 차시 예고
 - **5-3** 객체 지향 퍼셉트론

5주차(2/3)

기계학습 작업 흐름 2

파이썬으로 배우는 기계학습

한동대학교
김영섭 교수

여러분 곁에 항상 열려 있는 K-MOOC 강의실에서 만나 뵙기를 바랍니다.