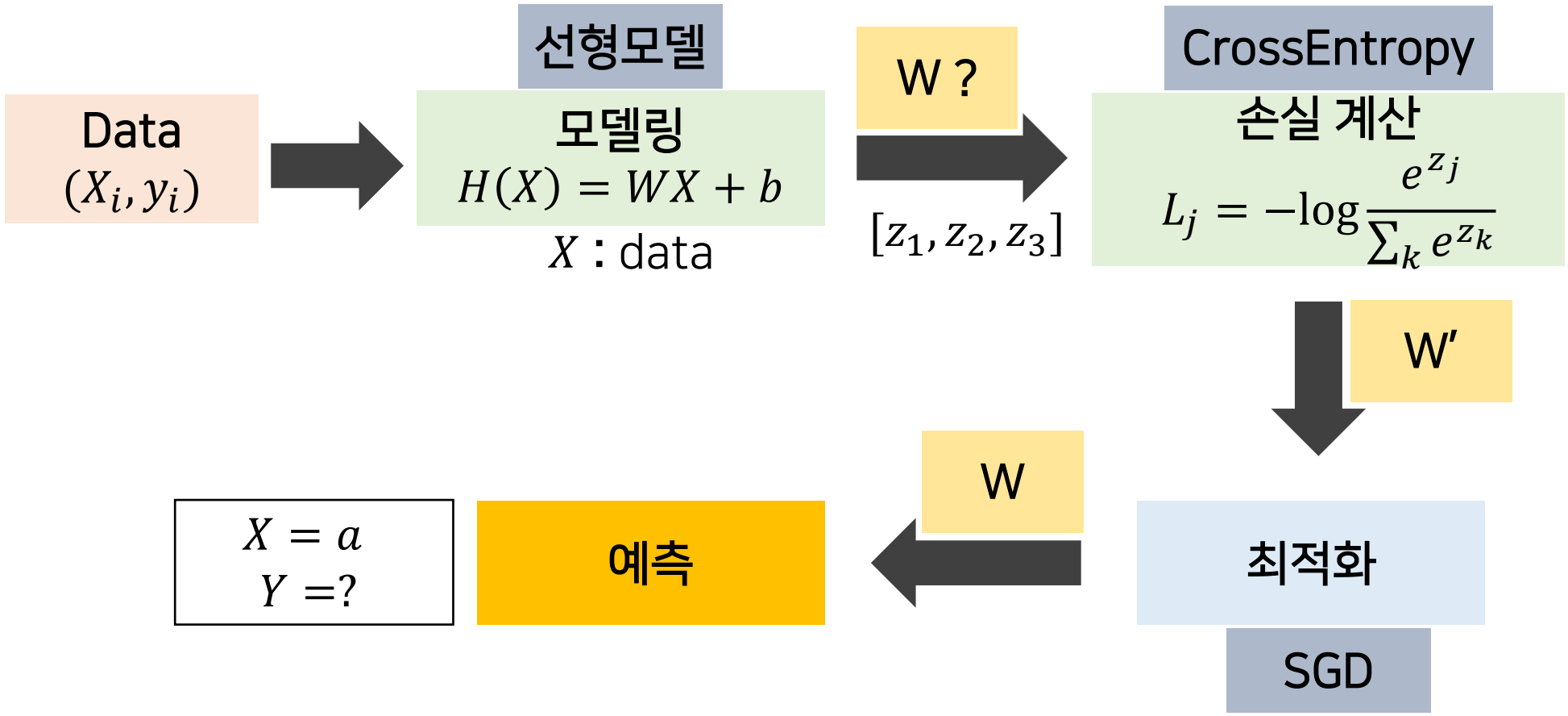
Contents

3장 기본 흐름 파악하기

선형 분류 과정


Data
(X_i, y_i)
선형모델
모델링
$H(X) = WX + b$
X : data
W ?
$[z_1, z_2, z_3]$
CrossEntropy
손실 계산
$L_j = -\log \frac{e^{z_j}}{\sum_k e^{z_k}}$
W'
최적화
SGD
W
예측
$X = a$
$Y = ?$

Contents

3장 기본 흐름 파악하기

Iris Dataset

	x1	x2	x3	x4	y
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

데이터 종류 : 정형 데이터

레이블 종류 : 다중 분류 (0,1,2)

문제 분류 : 분류 문제 (3 class)

독립변수(input) : x1, x2, x3, x4

종속변수(label) : y

0: Setosa

1: Versicolor

2: Virginica

데이터 불러오기

(a) 기본 라이브러리 불러오기

```
import numpy as np
```

(b) 데이터 불러오기

```
from sklearn.datasets import load_iris
```

#sklearn 프레임워크의 datasets 모듈에서 load_iris 함수 불러오기(import)

```
X,y = load_iris(return_X_y=True)
```

return_X_y=True 데이터를 X, y로 분리하여 불러오기

```
X.shape, y.shape
```

데이터 크기(shape) 확인

데이터 분석

numpy

```
X.mean(axis=0), X.std(axis=0), X.max(axis=0), X.min(axis=0)
```

pandas

```
import pandas as pd

data=pd.DataFrame(X)

data.describe()
```

	0	1	2	3
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

데이터 분리

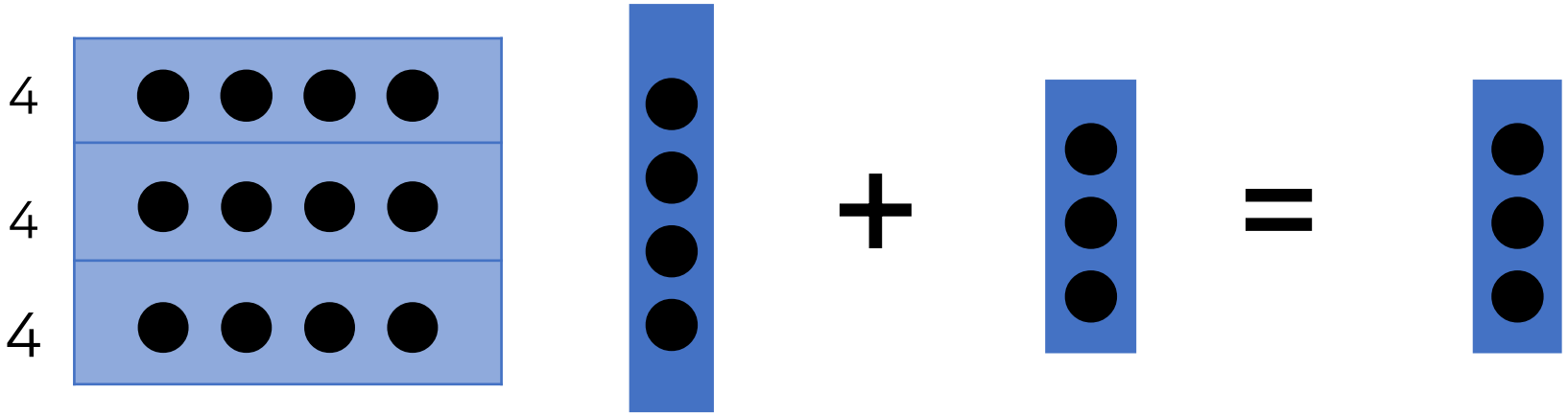
```
from sklearn.model_selection import train_test_split
train_x, test_x, train_y, test_y = train_test_split(X, y, test_size=0.3,
                                    random_state=42, stratify=y)
```

test_size	Test data 비율
random_state	Random seed
stratify	Data 정렬 기준

Contents

3장 기본 흐름 파악하기

기본 모델



W	x	b	Wx+b
가중치	입력 이미지	편향	스코어

선형 층

```
Dense(output, activation, input_shape)
```

output : 출력값의 개수

input_shape : 입력 벡터 형태

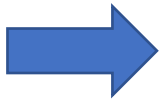
activation: 해당되는 경우에만 설정

class 개수인 3

4개의 변수, input_shape=(4,)

선형회귀의 경우: 설정 X

선형분류인 경우: softmax



```
Dense(3, activation='softmax', input_shape=(4,))
```

모델 설정

라이브러리 불러오기

```
from keras.models import Sequential
from keras.layers import Dense
```

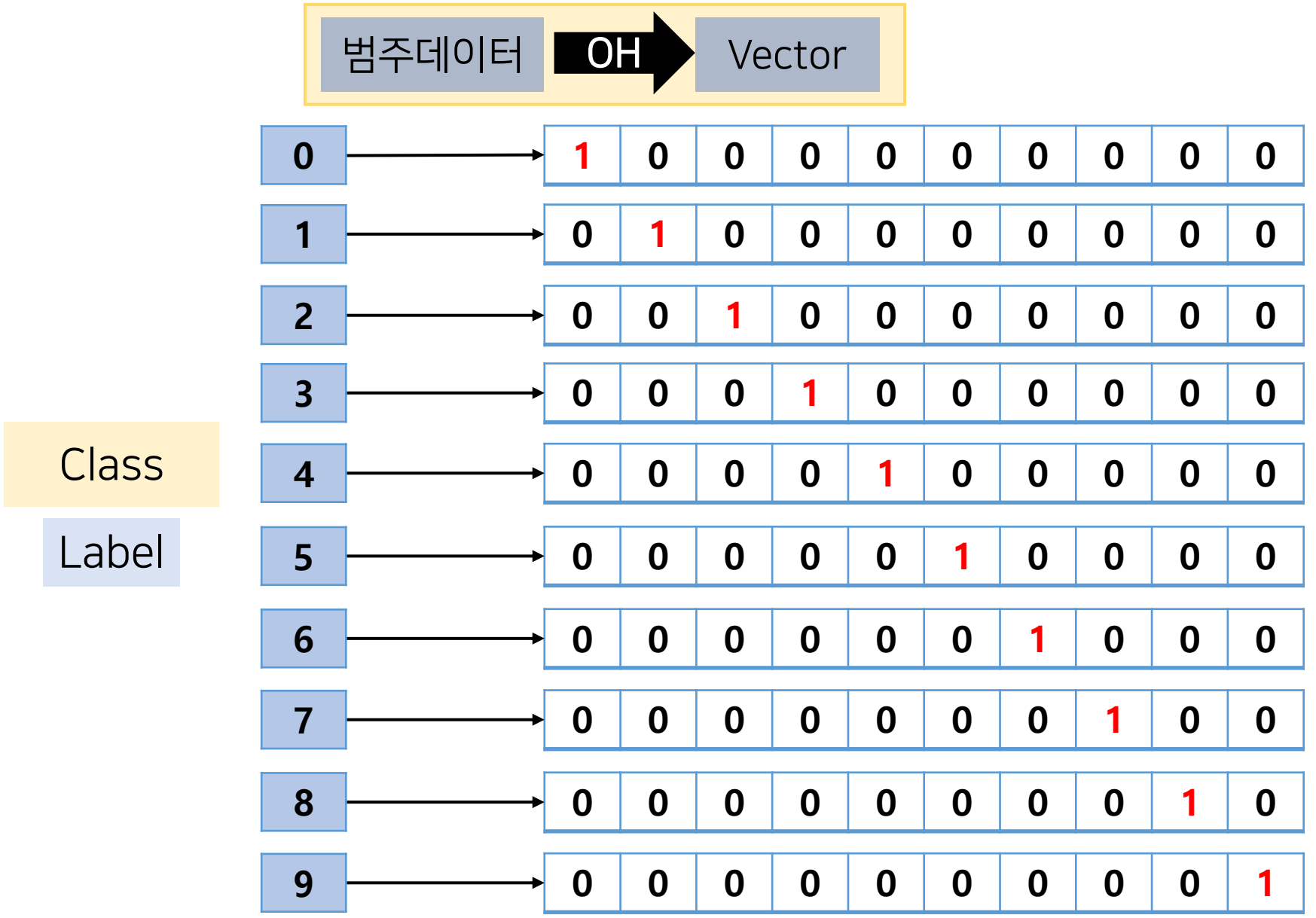
모델 설정

```
md=Sequential()  # 모델명을 md로 정의
md.add(Dense(3, activation='softmax', input_shape=(4,)))
md.summary()
```

Contents

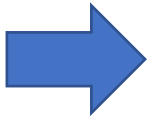
3장 기본 흐름 파악하기

*One-hot vector



손실 함수 설정

문제 종류	Label 차원(형태)	activation	Loss 함수
이진분류	1 (0 또는 1)	sigmoid	binary_crossentropy
다중분류	n (One-hot 벡터)	softmax	categorical_crossentropy
다중분류	n (숫자 0,1,2,…)	softmax	sparse_categorical_crossentropy
회귀	1 (실수)	-	mse / mae



```
Dense : activation='softmax'
compile : loss='sparse_categorical_crossentropy'
```

Contents

3장 기본 흐름 파악하기

compile

compile : loss, optimizer, metrics

loss	손실 함수 설정	loss='sparse_categorical_crossentropy'
optimizer	최적화 함수 설정	optimizer='sgd'
metrics	출력할 값 설정	metrics='acc'

fit

fit : input, output, epochs, batch_size, validation_data, validation_split

input	입력 데이터	X
output	출력 데이터(label)	y
epochs	학습회수	Epochs=200
batch_size	학습단위	Batch_size=100
validation_data	검증 데이터	Validation_data=(val_x,val_y)
validation_split	검증 데이터 분리	Validation_split=0.3

학습 설정

```
md.compile(loss='sparse_categorical_crossentropy',  optimizer='sgd’,
              metrics='acc')
```

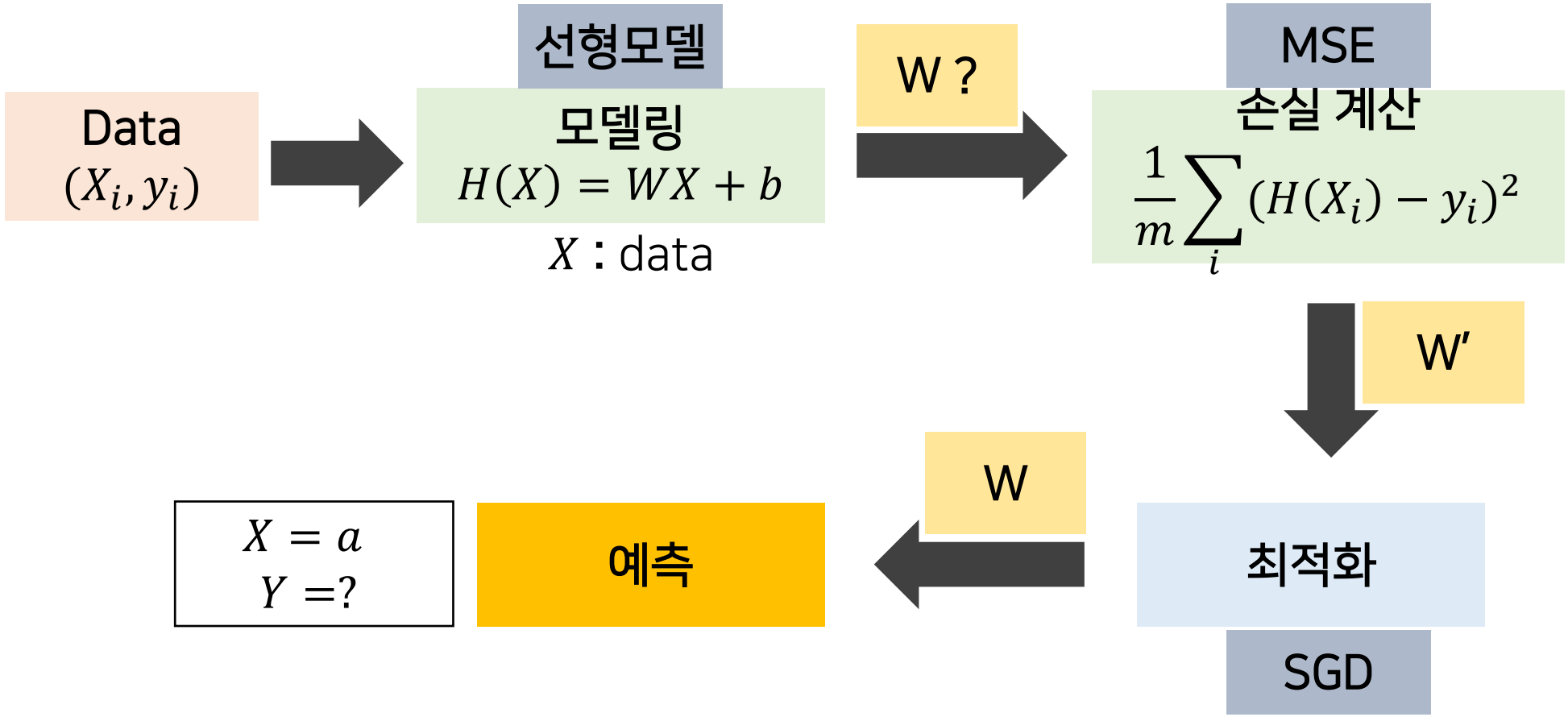
```
hist=md.fit(train_x, train_y, epochs=200)
```

학습진행 상황을 hist에 저장

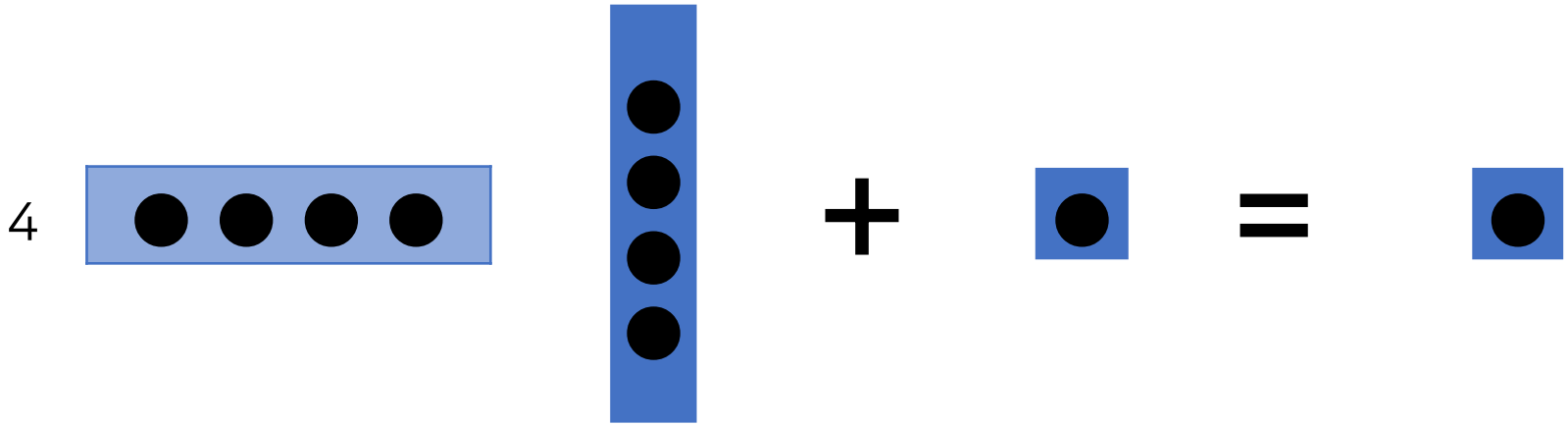
Contents

3장 기본 흐름 파악하기

선형회귀 과정


Data
(X_i, y_i)
선형모델
모델링
$H(X) = WX + b$
X : data
W ?
MSE
손실 계산
$\frac{1}{m}\sum_i (H(X_i) - y_i)^2$
W′
최적화
SGD
W
예측
$X = a$
$Y = ?$

선형회귀 모델 설정




```
Dense(1, input_shape=(4,))
```

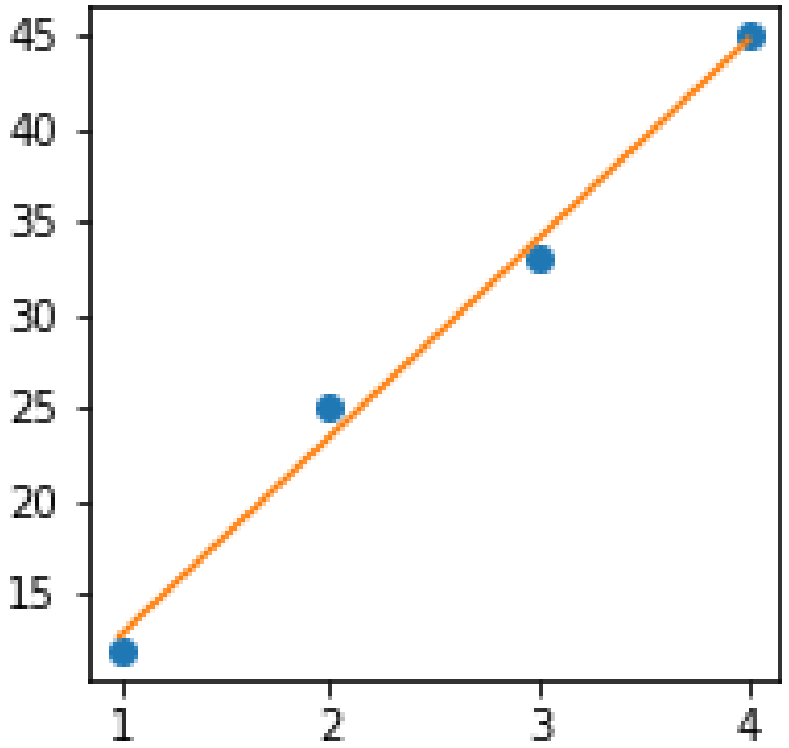
선형회귀 예제

Data

x(시간)	y(생산량)
1	12
2	25
3	33
4	45

모델

$$H(x) = ax + b$$



손실

MSE

$$\frac{1}{m}\sum_{i}(H(x_i) - y_i)^2$$

선형회귀 예제 학습 설정

```
Dense(1, input_shape=(1,))
```

```
md.compile(loss='mse', optimizer='sgd', metrics='acc')

hist=md.fit(train_x, train_y, epochs=50)
```

선형회귀 손실 계산

머신 러닝 과정

모델 : $H(x) = ax + b$

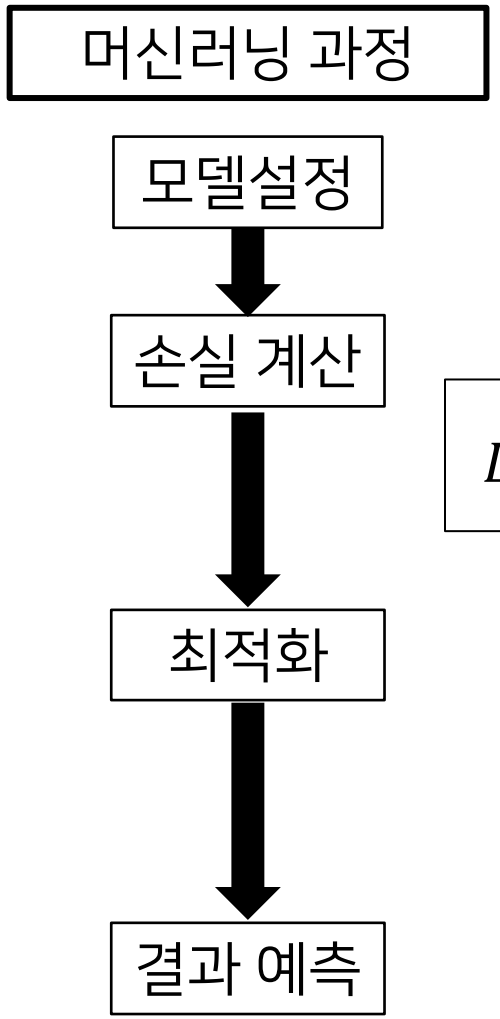
손실 :
$\frac{1}{m}\sum_i (H(x_i) - y_i)^2$

x(시간)	y(생산량)
1	12
2	25
3	33
4	45

$$L(a,b) = \frac{1}{4}\left((a+b-12)^2+(2a+b-25)^2+(3a+b-33)^2+(4a+b-45)^2\right)$$

$$L(a,b) = \frac{1}{4}(30a^2 + 20ab - 682a + 4b^2 - 230b + 3883)$$

선형회귀 최적화 과정



모델 : $H(x) = ax + b$

손실 :
$\frac{1}{m}\sum_i (H(x_i) - y_i)^2$

$$L(a,b) = \frac{1}{4}(30a^2 + 20ab - 682a + 4b^2 - 230b + 3883)$$

미분 : $\nabla_W L$

경사하강 : $W \leftarrow W - h\nabla_W L$

$H(5)??$

선형회귀 최적화

함수 최적화

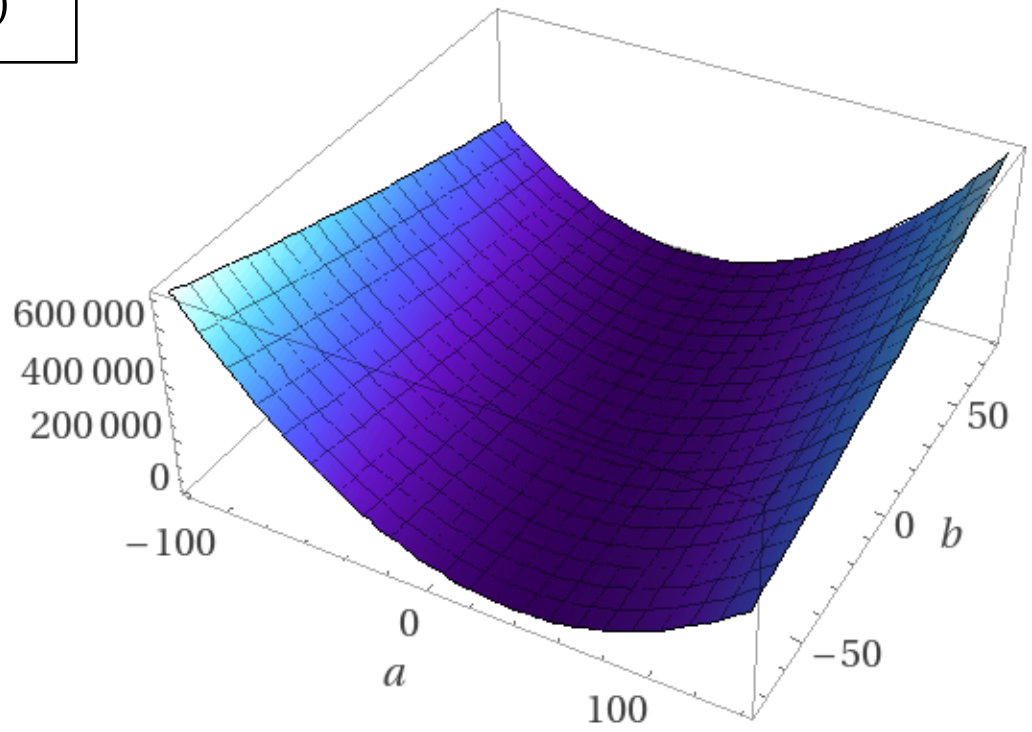
$$L(a,b) = \frac{1}{4}(30a^2 + 20ab - 682a + 4b^2 - 230b + 3883)$$

$$\nabla_W L = \left[\frac{1}{4}(60a + 20b - 682), \frac{1}{4}(20a + 8b - 230)\right] = 0$$

⇨ $(a,b) = (10.7, 2)$ 임계점

⇨ $L(10.7, 2) \approx 1.1$ minimum

⇨ $H(5) = 10.7 \times 5 + 2 = 55.5$



Computed by Wolfram|Alpha

선형회귀 최적화

경사하강법

Weight update 1

$$L(a,b) = (30a^2 + 20ab - 682a + 4b^2 - 230b + 3883)/4$$

$$\nabla_W L = [(60a + 20b - 682)/4, (20a + 8b - 230)/4]$$

$$W^{(1)} = W^{(0)} - h\,\nabla_W L \qquad h = 0.01$$

$W^{(0)} = [20,10]$ ⇨ $\nabla_W L = [5, 2.1]$ ⇨ $W^{(1)} = [12.8,7.5]$

초기값은 random

Loss = 1089

Loss= 198.3

선형회귀 최적화

경사하강법

Weight update n

$$L(a,b) = (30a^2 + 20ab - 682a + 4b^2 - 230b + 3883)/4$$

$$\nabla_W L = [(60a + 20b - 682)/4, (20a + 8b - 230)/4]$$

$$W^{(2)} = W^{(1)} - h\,\nabla_W L \qquad h = 0.01$$

$W^{(1)} = [12.8, 7.5]$ ⇨ $W^{(2)} = [10.4, 6.6]$ ⇨ $W^{(n)} = [\cdots, \cdots]$

Loss= 198.3

Loss= 17.2

Optimal Solution

$W = [10.7, 2]$ ⇨ $y(5) \approx 55.5$ predict

Contents

3장 기본흐름 파악하기

실습예제 선형분류

Dense

Dense()

```
Dense(n,activation=None)
```

-n : 출력층 개수 설정

-activation : 활성화 함수 설정

```
Dense=(3, activation='softmax', input_shape=(4,))
```

#input_shape : input data shape 설정(첫번째 층인 경우에 설정)

Sequential

Sequential()

```
md=Sequential()
md.add(…)
```

-add() : 층을 추가

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
md=Sequential()
md.add(Dense=(3, activation='softmax', input_shape=(4,)))
```

선형분류 coding data loading

numpy 호출

```
import numpy as np
```

data load

```
from sklearn.datasets import load_iris
```

```
X,y = load_iris(return_X_y=True)
```

```
X.shape, y.shape
```

data 분리

```
from sklearn.model_selection import train_test_split
train_x, test_x, train_y, test_y = train_test_split(X, y, test_size=0.3,
                                   random_state=42, stratify=y)
```

선형분류 coding 모델 설정

모델 설정

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

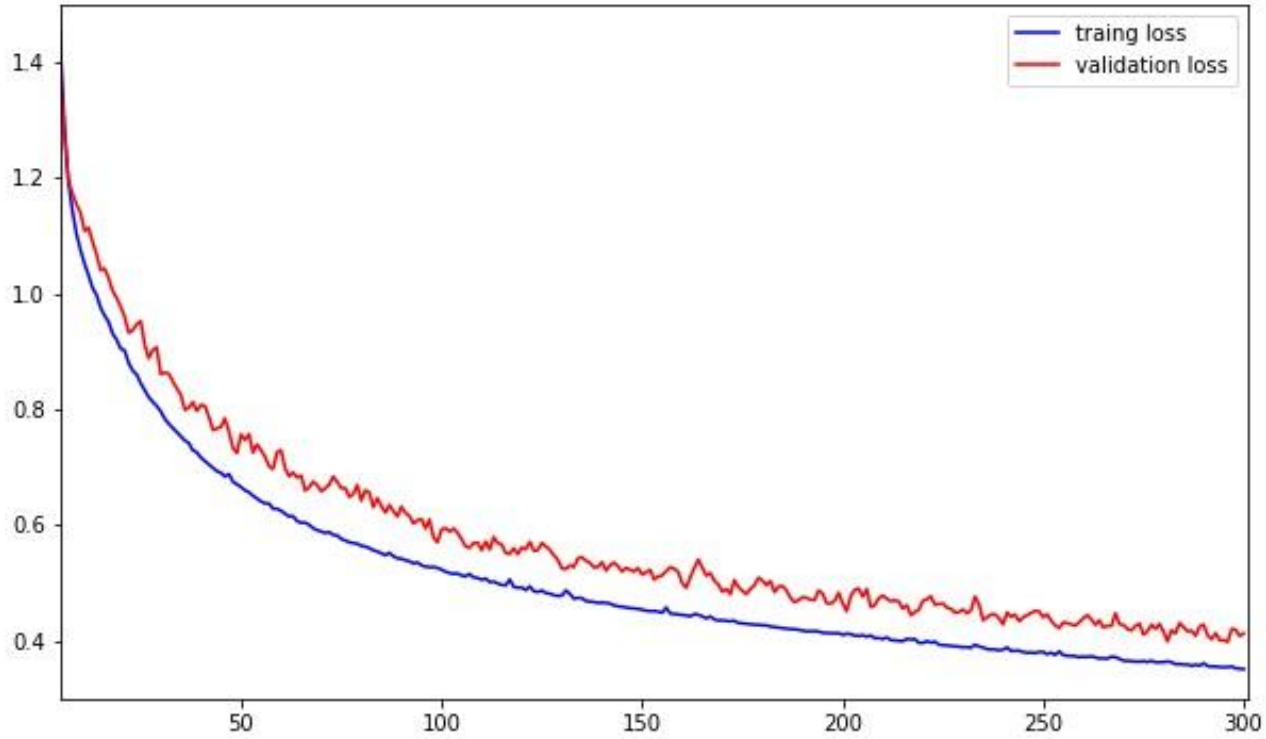
```
md=Sequential()  # 모델명을 md로 정의
md.add(Dense(3, activation='softmax', input_shape=(4,)))
md.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_7 (Dense)              (None, 3)                 15
=================================================================
Total params: 15
Trainable params: 15
Non-trainable params: 0
_________________________________________________________________
```

선형분류 coding 학습

compile/fit

```
md.compile(loss='sparse_categorical_crossentropy',  optimizer='sgd’,
            metrics='acc')
hist=md.fit(train_x, train_y, epochs=200)
```



선형분류 coding 평가

평가

```
md.evaluate(test_x, test_y)
```

```
2/2 [==============================] - 0s 2ms/step - loss: 0.3850 - acc: 0.9000
[0.3850276470184326, 0.8999999761581421]
```