12/01 Simple Linear Regression

정규화 (Normalization)

① Min-Max Normalization (Scaling) → 0~1 사이값으로 변환

이상치에 민감

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

② (z-score Normalization) Standardization

$$x_{scaled} = \frac{x - x_{평균}}{x_{표준편차}}$$

이상치에 둔감

Simple Linear Regression → $\hat{y} = \beta_0 + \sum_{i=1}^{n} \beta_i x_i$

독립변수 1개

연속적인 수치값을 예측하기 위한 Regression

- ① Python → 사실은 생능적인 면에서 쉬울 오늘이요!
- ② Tensorflow → 코드는 Sklearn이 비해서 어렵지만 deep learning 구현이 가능
- ③ Sklearn → 초보자용. deep learning에서는 불편

데이터 전처리

① 결측치처리 → 삭제 / Imputation

② 이상치처리 → Tukey's Fence / Z-Score (정규분포)

③ 정규화처리 → MinMax Normalization (이상치에 취약, 0~1값) / Z-Score Normalization (Standardization)

* Multiple Linear Regression

↳ 독립변수가 여러개

↳ 우리의 target (t_data) 값 종속변수 가 연속적인 숫자값

Temp → Ozone

Temp, 태양광, 바람 → Ozone *

$$\hat{y} = \beta_0 + \sum_{i=1}^{n} \beta_i x_i$$

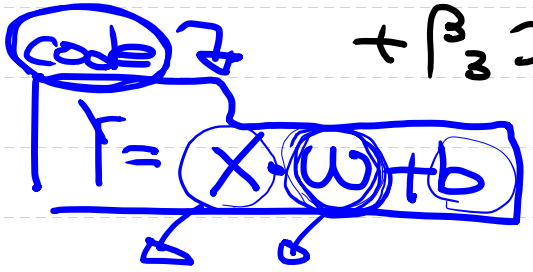
if n=3

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

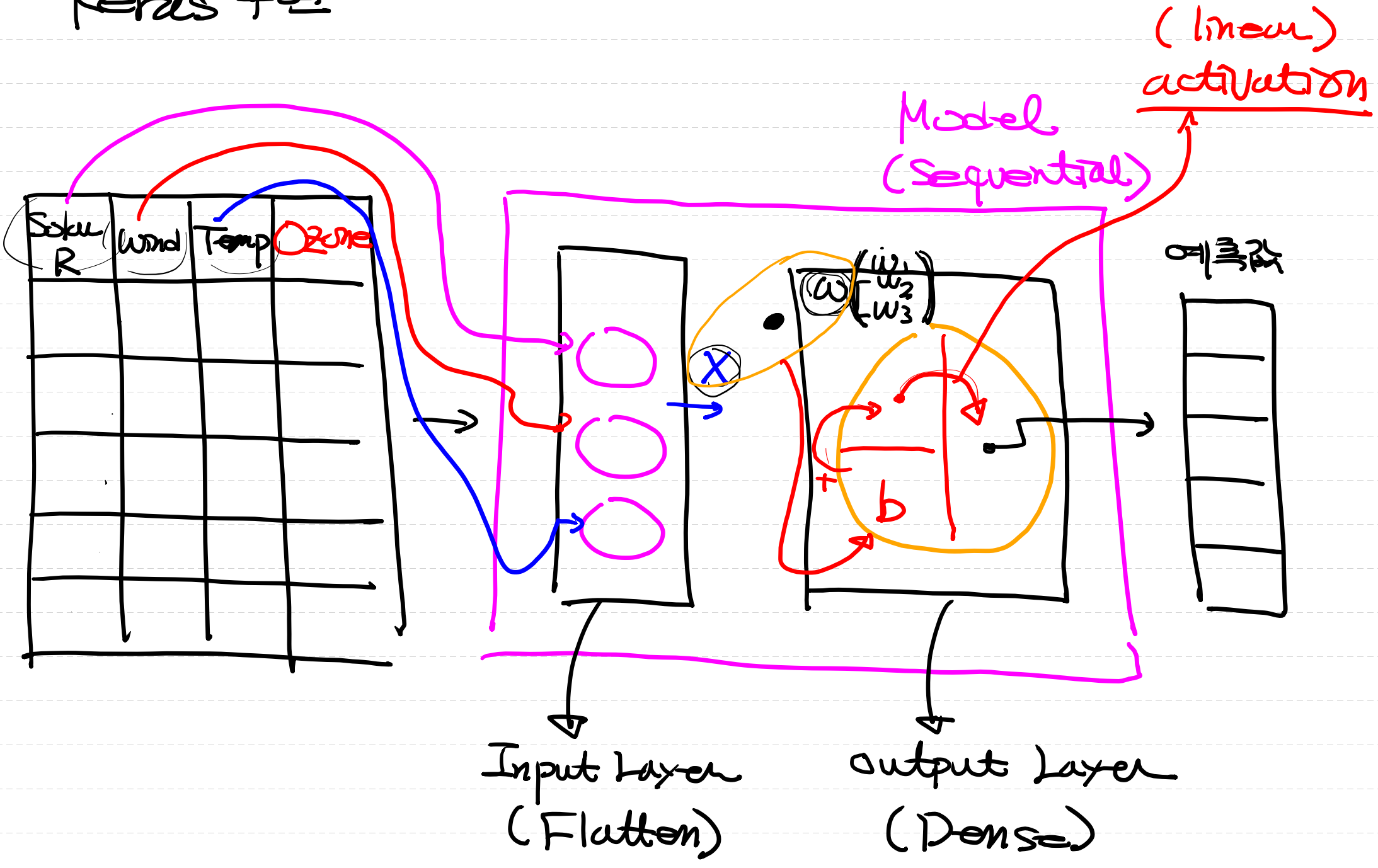
우리의 Model

$$y = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

$Wx+b$



"keras 구현"


(linear)
activation
Model
(Sequential)
Solar R
Wind
Temp
Ozone
X
W
b
예측값
Input Layer
(Flatten)
output Layer
(Dense)

Simple Linear Regression

Multiple Linear Regression

연속적인 수치값을 예측

Multiple Logistic Regression → 이산적인 분류값을 예측

↳ 이항분류 (이진분류)

→ "binary classification"

"deep Learning의 기본 component"

- Classification (분류)

ex) MRI사진 → 병이 있다 / 없다

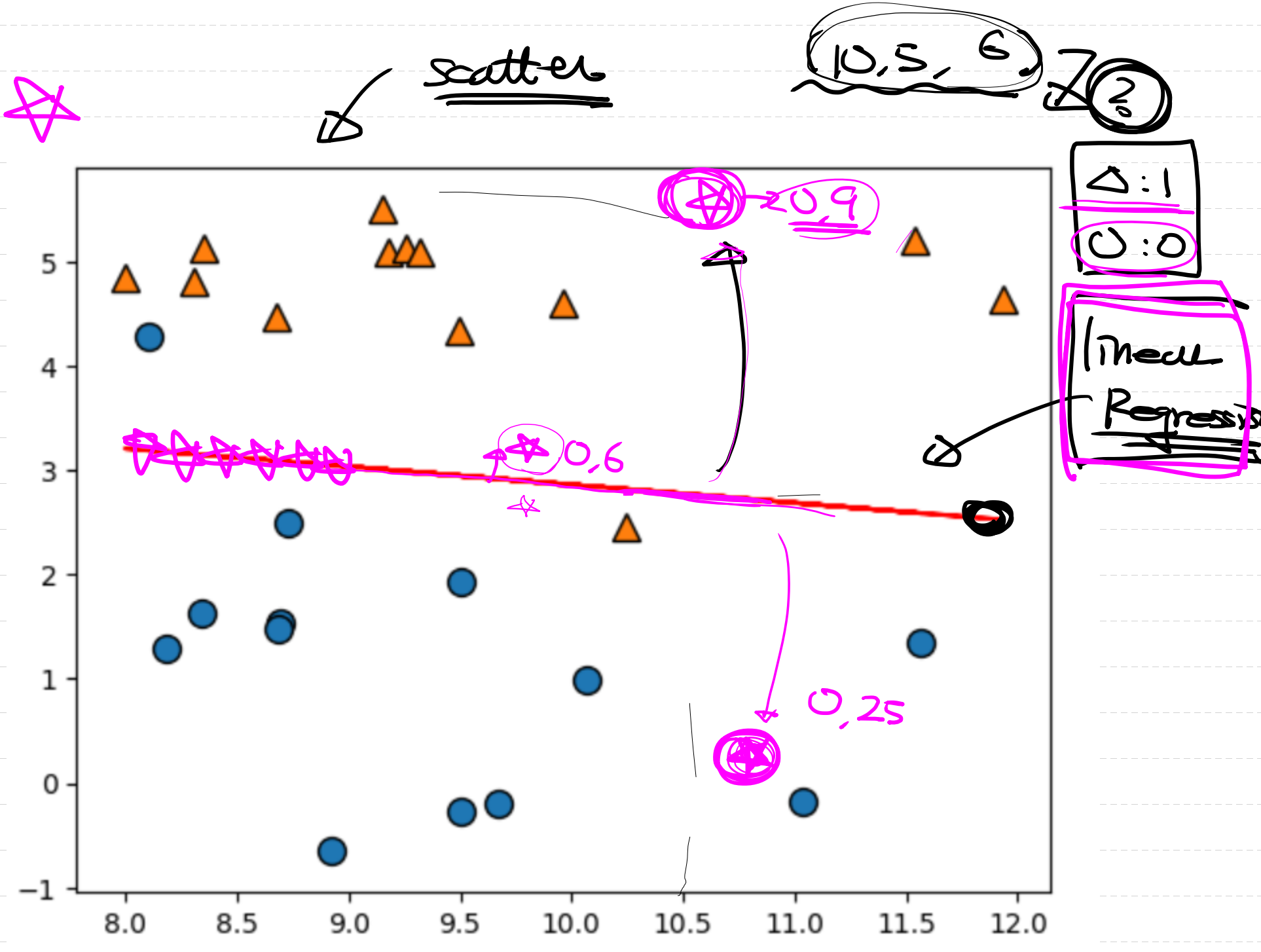
✻ 신용카드 → 정상 / 비정상.

공부시간	합격여부
x	t
1	0
3	0
5	0
7	1
9	1

6시간? → ML → ✻ 0.63 (확률)

0.5 기준 ↑ ① ↓ ⓪

"코드로 이해"


scatter
10,5, 6
?
△:1
○:0
0,9
0,6
0,25
8.0
8.5
9.0
9.5
10.0
10.5
11.0
11.5
12.0
5
4
3
2
1
0
−1

logistic Regression

Training DataSet

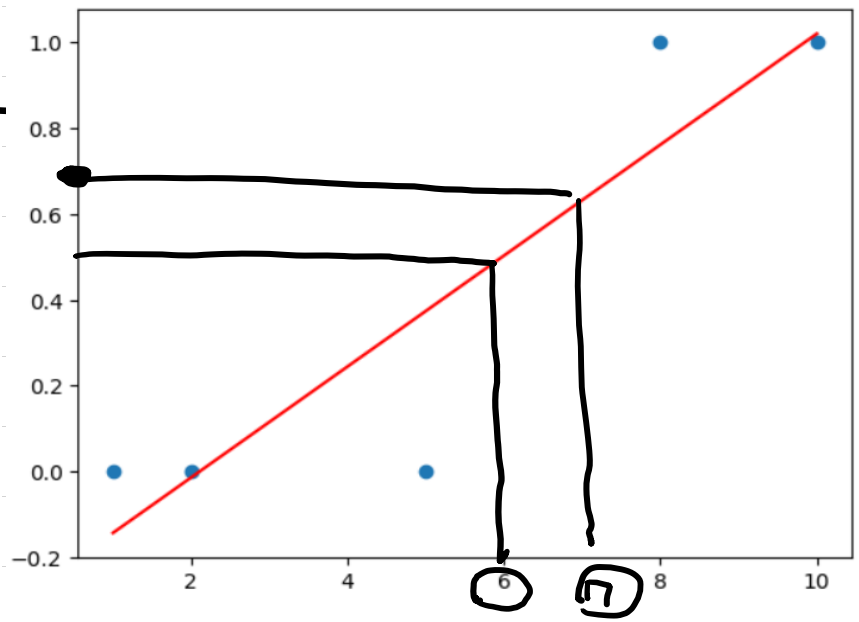
Model

Linear Regression

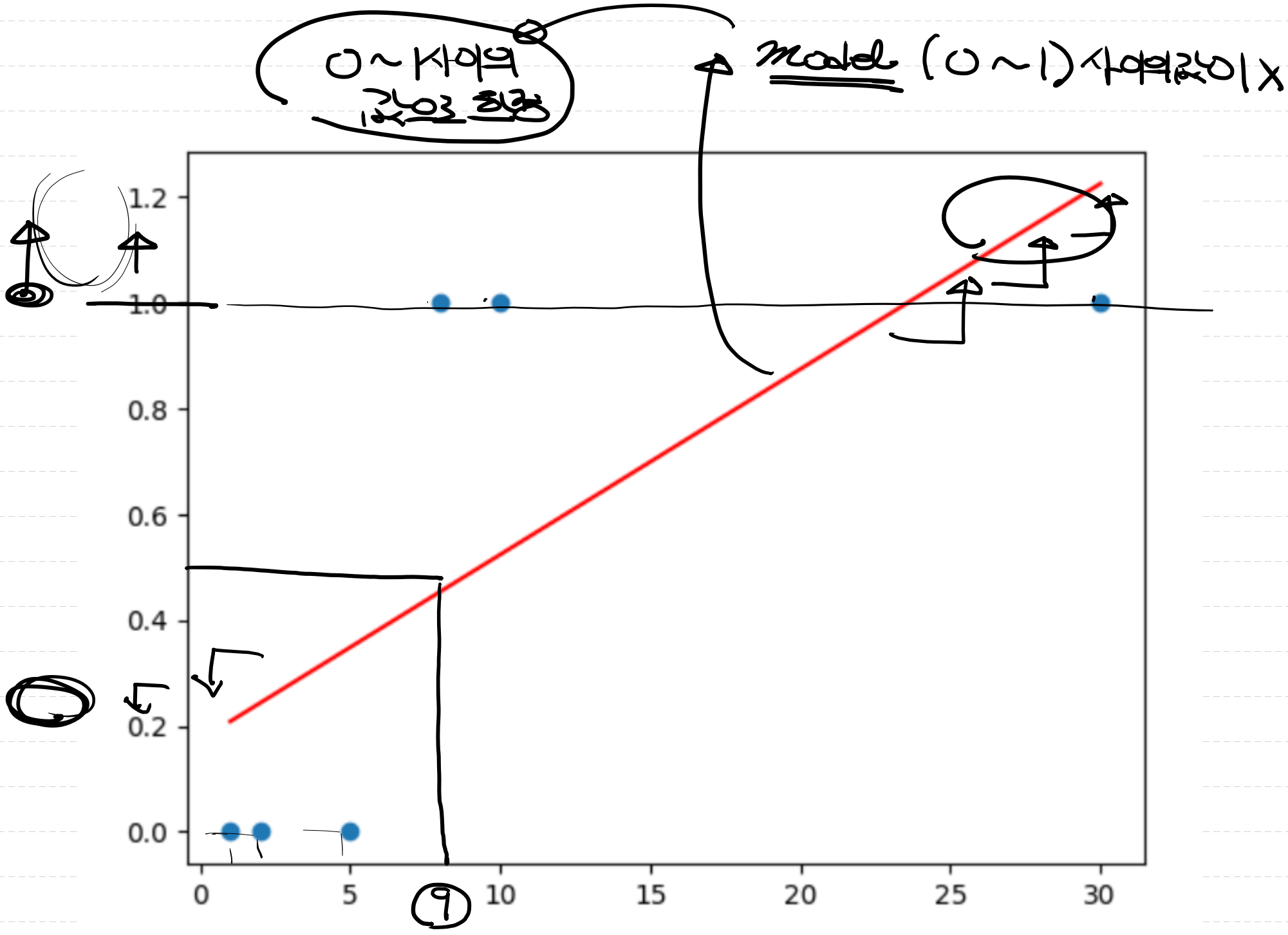
Classification

예측값

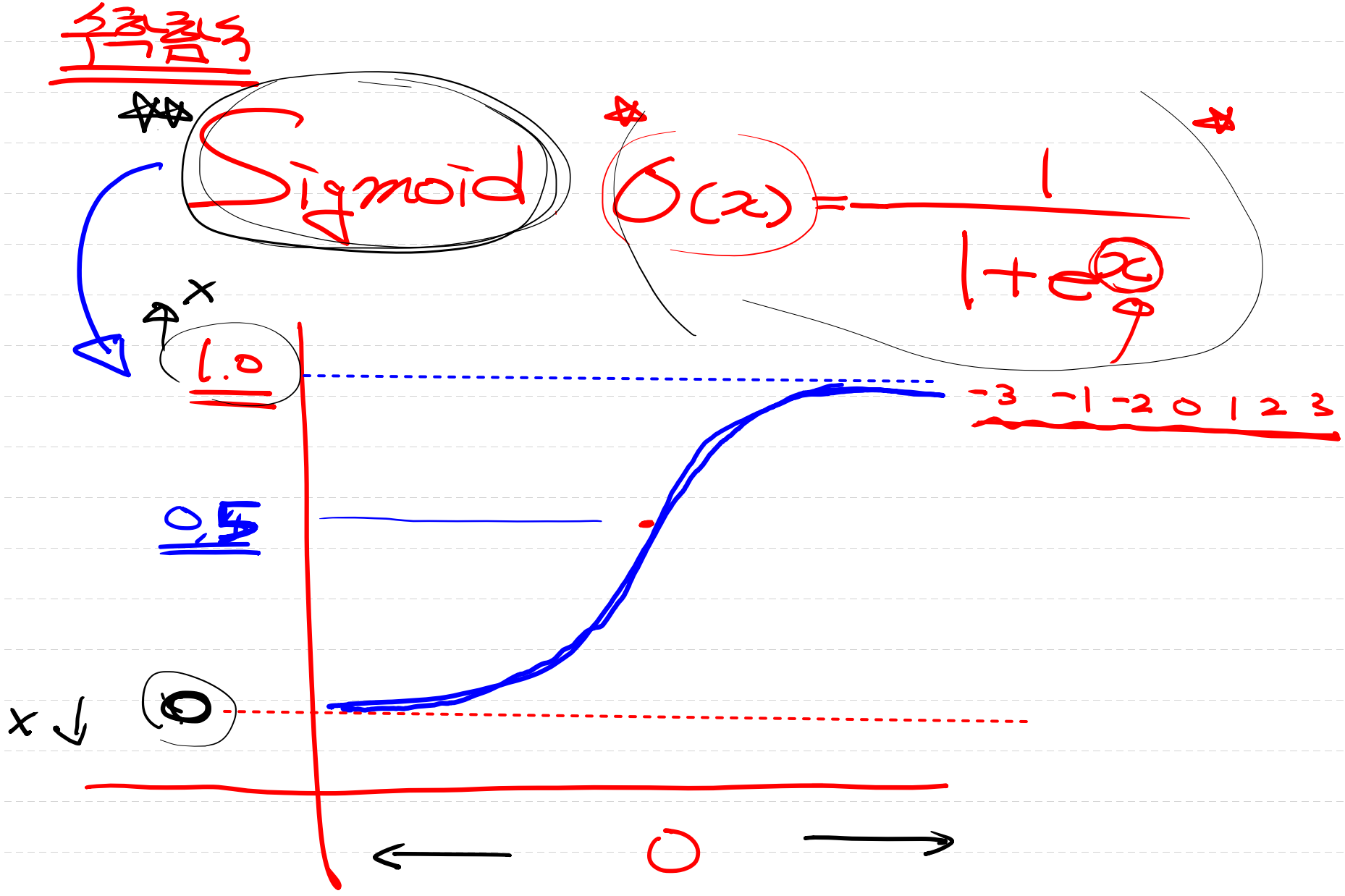
0 ~ 1



0.63 !!.


0~사이의
값을 출력
model (0~1)사이값이X
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
5
9
10
15
20
25
30

그럼 어떻게 model을 만들까요??



우리의 linear Regression Model

↳ Sigmoid 함수를 적용

* logistic Regression Model

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$y = Wx + b \rightarrow \text{linear regression Model}$$

$$y = \frac{1}{1+e^{-(Wx+b)}} \rightarrow \text{logistic regression model}$$

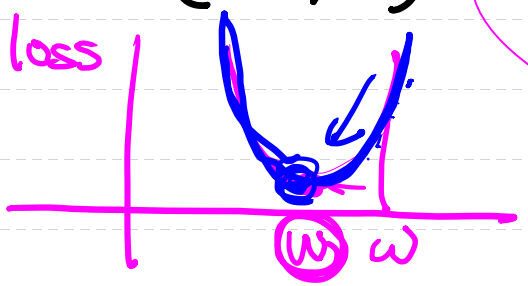
결론적으로 Model의 바뀐거죠!!!

Linear Regression의 loss

평균제곱오차(MSE)

$$E(w,b) = \frac{1}{n}\sum_{i=1}^{n}\left[t_i - (wx_i+b)\right]^2$$

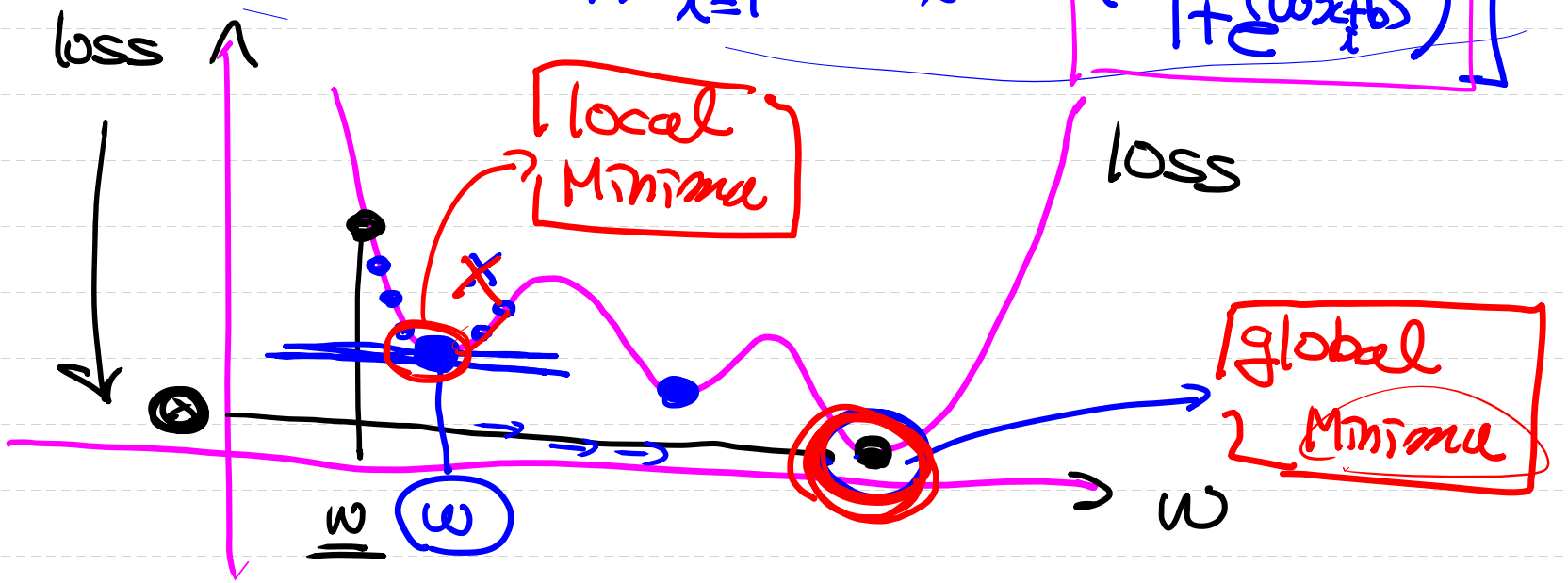
Convex



만약 Logistic Regression의 loss를 MSE를 이용한다고 가정한다면

$$E(w,b) = \frac{1}{n}\sum_{i=1}^{n}\left[t_i - \left(\frac{1}{1+e^{(wx_i+b)}}\right)\right]^2$$

loss를 바꿔야해요



loss의 식을 다시 만들어요

(MSE를 쓰세요)

$$E(w,b) = -\sum_{i=1}^{n} (t_i \log y_i + (1-y_i) \log (1-y_i))$$

Cross Entropy

Linear Regression에서

Logistic Regression으로

(종속변수, target)

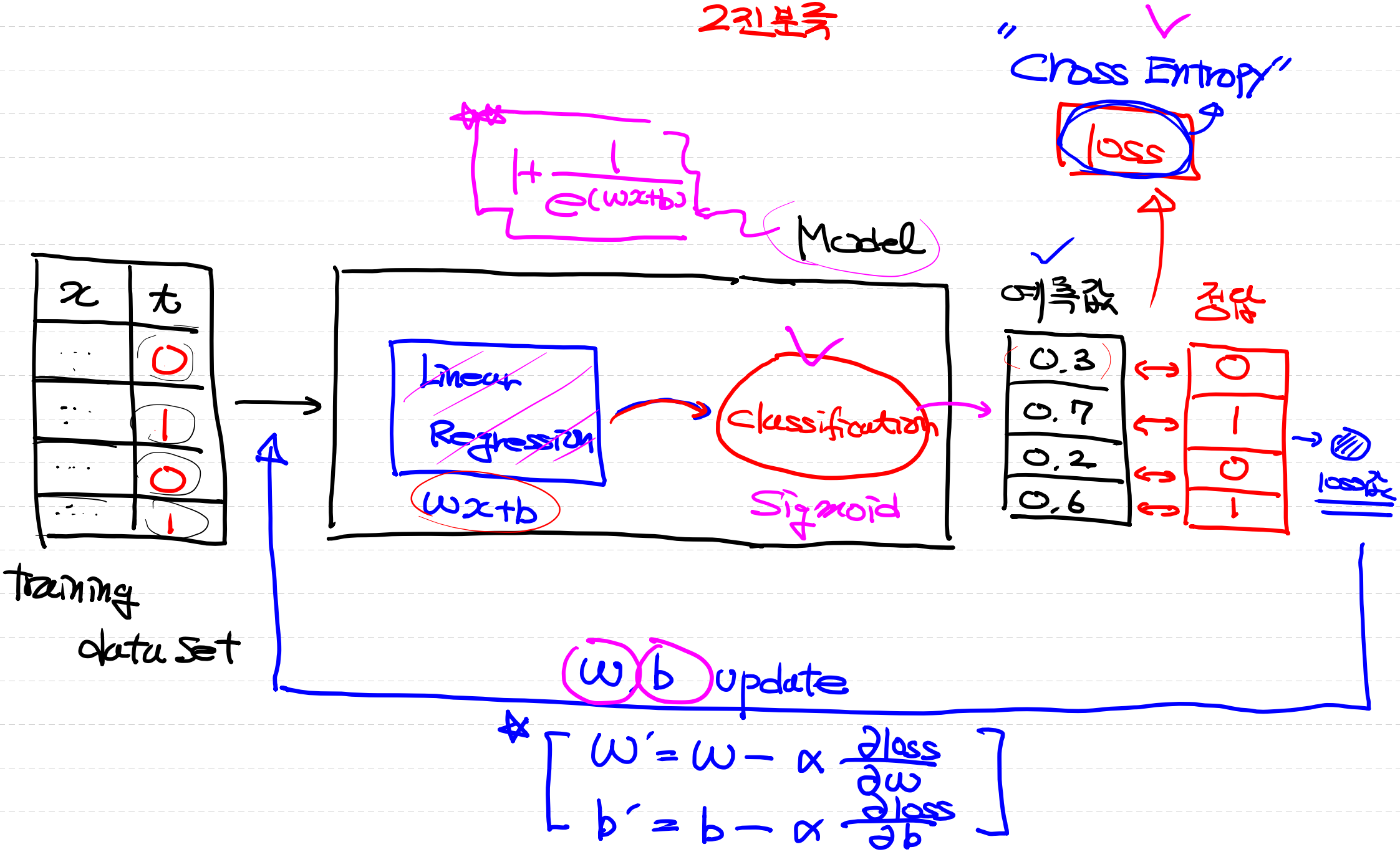
① 데이터가 바뀌어요 (0, 1)

② Model이 변경 (sigmoid)

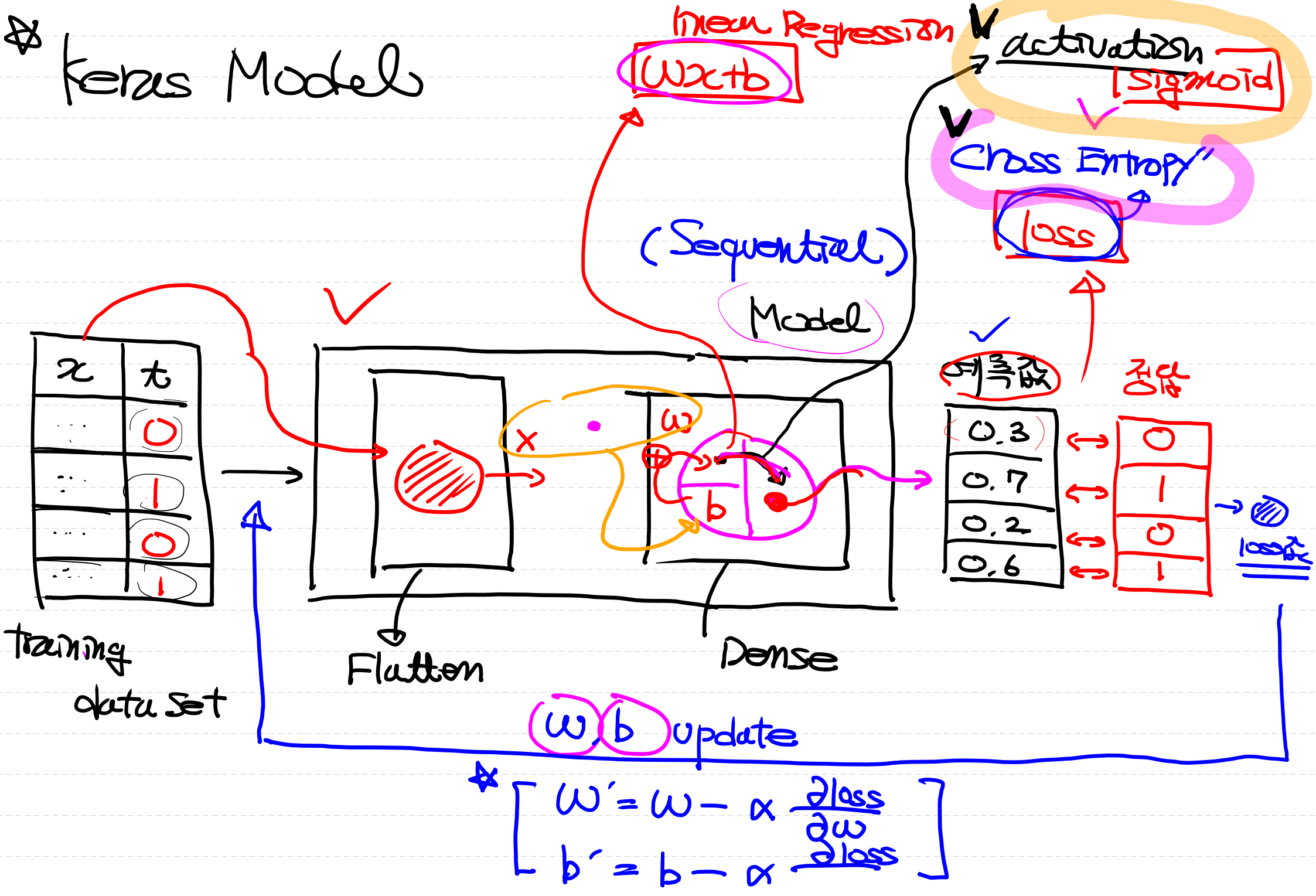
③ loss가 변경 (MSE → cross Entropy)

Logistic Regression (binary classification)

2진분류



keras Model


linear Regression
Wxtb
activation
sigmoid
Cross Entropy
loss
(Sequential)
Model
x
t
0
1
0
1
x
w
b
Flatten
Dense
예측값
0.3
0.7
0.2
0.6
정답
0
1
0
1
loss값
training
data set
w, b update
$w' = w - \alpha \frac{\partial loss}{\partial w}$
$b' = b - \alpha \frac{\partial loss}{\partial b}$

구현한 결과가 둘다 달라요!!

- Sklearn → 0 (0.4999)
- Tensorflow → 1 (0.59)

?

① Sklearn을 믿고가야합니요? (X)

② 검증방법이 필요!! → 내가 만든 model이 잘 만든 model인지 확인하는 과정!!

평가방법 + 모델 평가를 진행하기 위해서 평가지표(Metrics)를 이용해요 (많은 평가 지표가 존재해요)