

## 03/11 Tensorflow 2.x

→ 지금까지는 TF 1.15 버전을 이용

↳ Low Level API를 이용하기 구현

내용을 이해하기 힘들어요.

⇒ 2.x 버전으로 넘어오면서 많은 변화점이 생겼어요.

- Session 삭제
- 초기화 코드 삭제
- Placeholder 삭제

Tensorflow는 google의 딥러닝 라이브러리

- TensorFlow → google
- PYTorch → Facebook
- Keras → 프랑스인 팀
- MXNet
- ...

TF 2.0 '2019년 9월 30일'

2.4.1

- Eager Execution (즉석실행) → 코드의 직관성이 좋아졌어요
- keras가 High-level API로 공식적으로 지원.

6

- Machine Learning의 raise and fall (흥망성쇠)

→ 1986년 체프지 알고리즘 (Back propagation) (Neural Network)  
알고리즘이 의해 Machine Learning이 다시 각광

→ 2010년 부터 Deep Learning이 불꽃시작. ↗ 대학 연구실과  
기초학회으로 Deep Learning Library가  
만들어지기 시작했습니다.

5)

- 2010 → 브이오의 "Theano" ← 예전 수학과에서 고등학교의  
언어
- 2013 → 버클리대학 중국 국립대 (기아천) → Caffe → 2017년도 개발 종지!!
- 2014 → 코널 Torch3

2015 →  
카네기멜론대학 MXNet  
구글 Tensorflow  
프랑스의 툴킷 keras (쉽게 사용가능)

- 2016 → MS의CNTk
- 2019 → TF 2.0으로 개발

## • Simple Linear Regression

✓  $\hat{y} = \beta_0 + \sum_{i=1}^p \beta_i x_i \rightarrow \text{일반적}$   
 (Classical Linear  
Regression Model)

✓  $y = \beta_0 + \beta_1 x_1$

✓  $y = Wx + b$  label

$x(\text{입력})$	$t(\text{정답})$
1	3
2	5
3	7
4	9

$x_1 w + b = y_1$

$x_2 w + b = y_2$

$x_3 w + b = y_3$

$x_4 w + b = y_4$

$X \cdot W + b = T$

$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \cdot (W + b) = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}$   
 $(4 \times 1) \quad (1 \times 1)$

✓ loss function

$$E(W, b) = \frac{1}{n} \sum_{i=1}^n (t_i - (w_i x_i + b))^2$$

$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \quad \begin{pmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{pmatrix}$   
비교

$\begin{matrix} \uparrow & \uparrow \\ \text{예측값} & \text{정답} \\ (4 \times 1) & (4 \times 1) \end{matrix}$

## • Multiple Linear Regression

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

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

$x_1$	$x_2$	$x_3$	$t$ ( <small>target</small> )
73	62	80	153
:			
:			
:			
:			

$$X \cdot (w + b) = Y$$

행렬식

$$\begin{pmatrix} 73 & 62 & 80 \\ : & : & : \\ : & : & : \\ : & : & : \\ (4 \times 3) \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} + \begin{pmatrix} b \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \quad \text{예측}$$

loss function

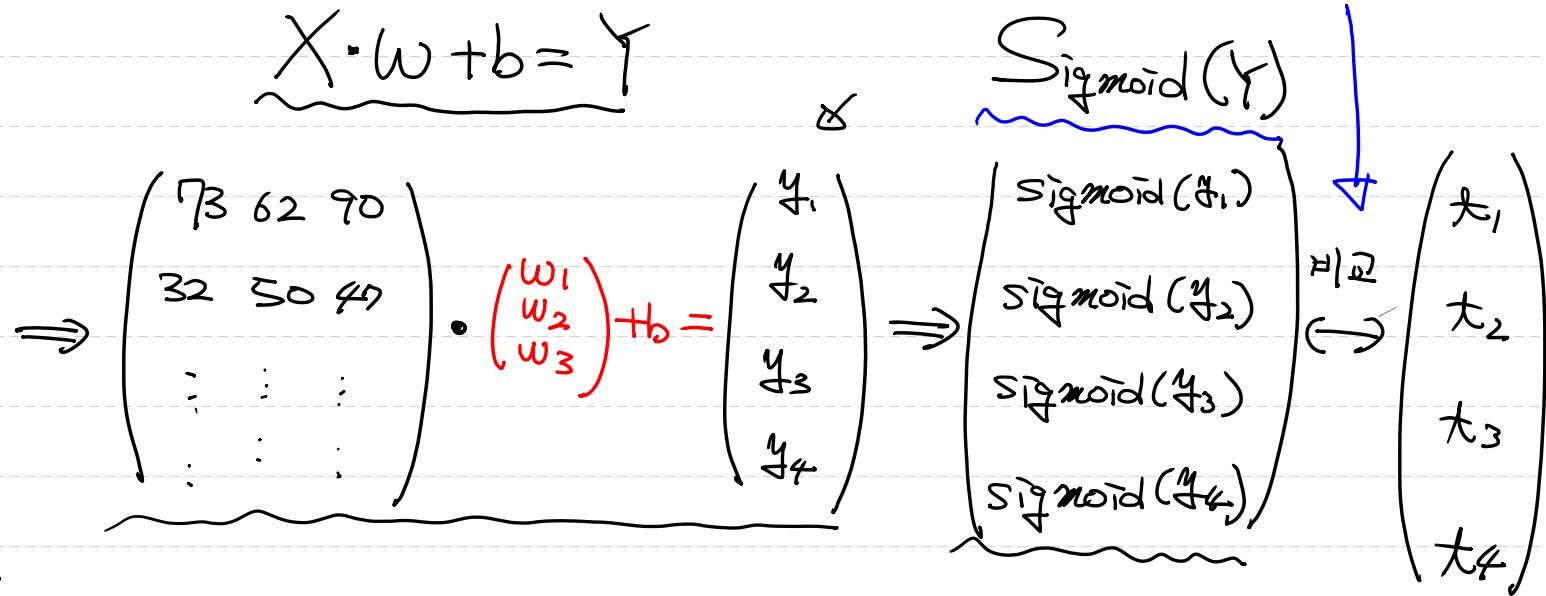
$$E(w, b) = \frac{1}{n} \sum_{i=1}^n [t_i - (w \cdot x_i + b)]^2$$

• Multiple Logistic Regression

Cross Entropy =

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

$x_1$	$x_2$	$x_3$	$t(x)$
73	62	90	0
32	50	47	1
:	:		0
:	:		1



## - Multinomial Classification

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

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

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

$$y = Wx + b$$

$x_1$	$x_2$	$x_3$	$t$ (target)
73	62	90	A
80	20	50	B
23	33	70	B
20	10	50	C

한정부제

$$\begin{pmatrix} 73 & 62 & 90 \\ 80 & 20 & 50 \\ 23 & 33 & 70 \\ 20 & 10 & 50 \end{pmatrix}$$

$$(4 \times 3) \quad (3 \times 3)$$

$$\begin{pmatrix} w_{A1} & w_{B1} & w_{C1} \\ w_{A2} & w_{B2} & w_{C2} \\ w_{A3} & w_{B3} & w_{C3} \end{pmatrix}$$

$$(b_A \ b_B \ b_C)$$

$$\begin{pmatrix} y_{A1} & y_{B1} & y_{C1} \\ y_{A2} & y_{B2} & y_{C2} \\ y_{A3} & y_{B3} & y_{C3} \\ y_{A4} & y_{B4} & y_{C4} \end{pmatrix}$$

Softmax(Y)

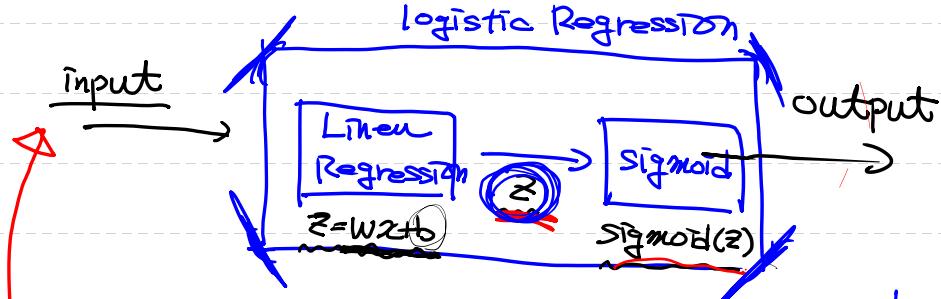
$$\begin{pmatrix} y_{A1}' & y_{B1}' & y_{C1}' \\ y_{A2}' & y_{B2}' & y_{C2}' \\ y_{A3}' & y_{B3}' & y_{C3}' \\ y_{A4}' & y_{B4}' & y_{C4}' \end{pmatrix}$$

One-hot

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

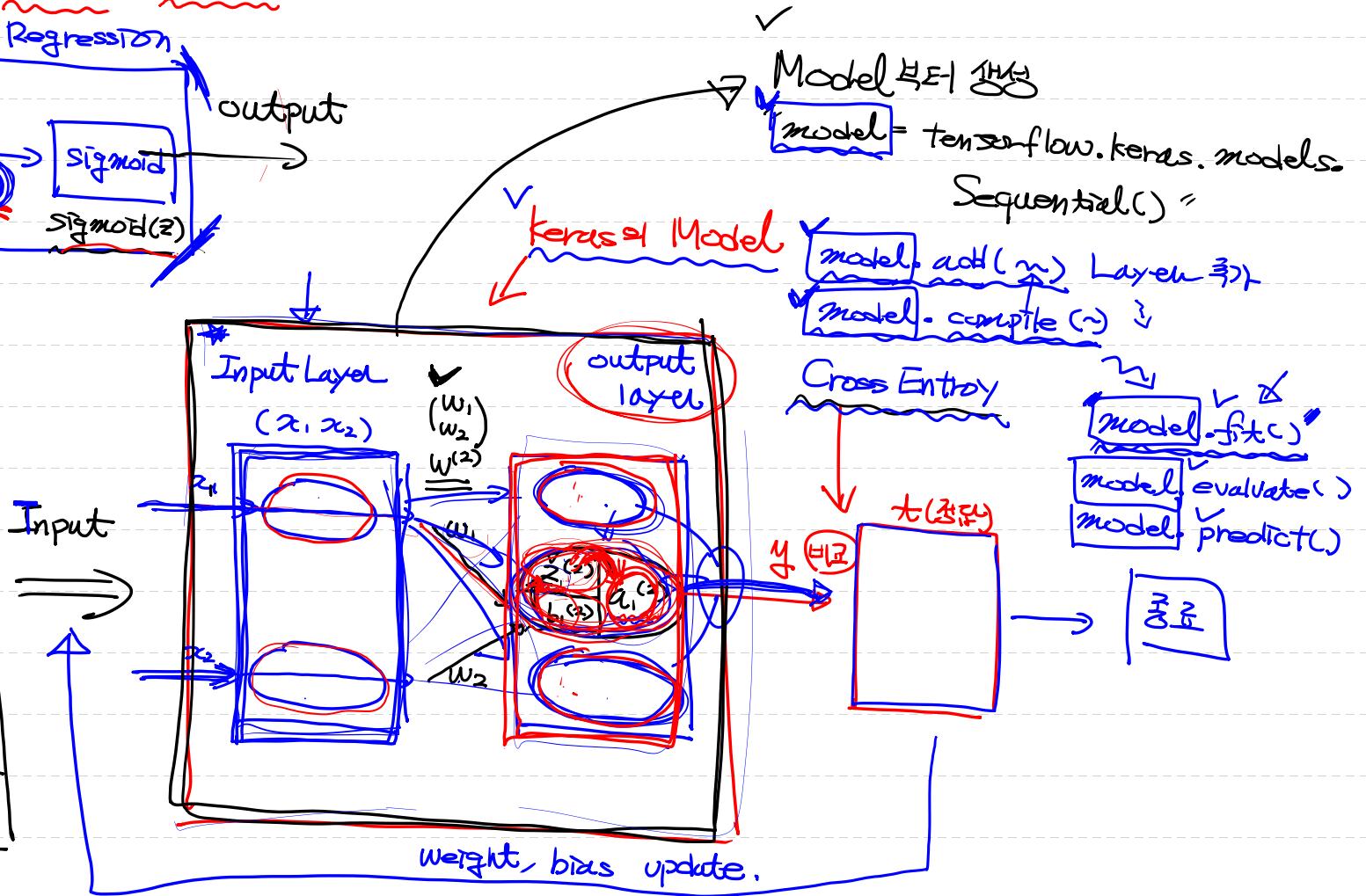
Cross Entropy

- Logistic Regression은 keras의 Model을 이용하여 표현



Training Data Set

$x_1$	$x_2$	$t$ (정답)
$x_{11}$	$x_{12}$	O/A
$x_{21}$	$x_{22}$	O/B
$x_{31}$	$x_{32}$	I/C
$x_{41}$	$x_{42}$	O/A
:	:	: B



## ① KNN (k-Nearest Neighbor)

KNN 알고리즘 K최근접이웃이라고 하기도 해요.

"근접을 선형회귀 모델"

K를 기준으로

$K=1 \rightarrow$  소장

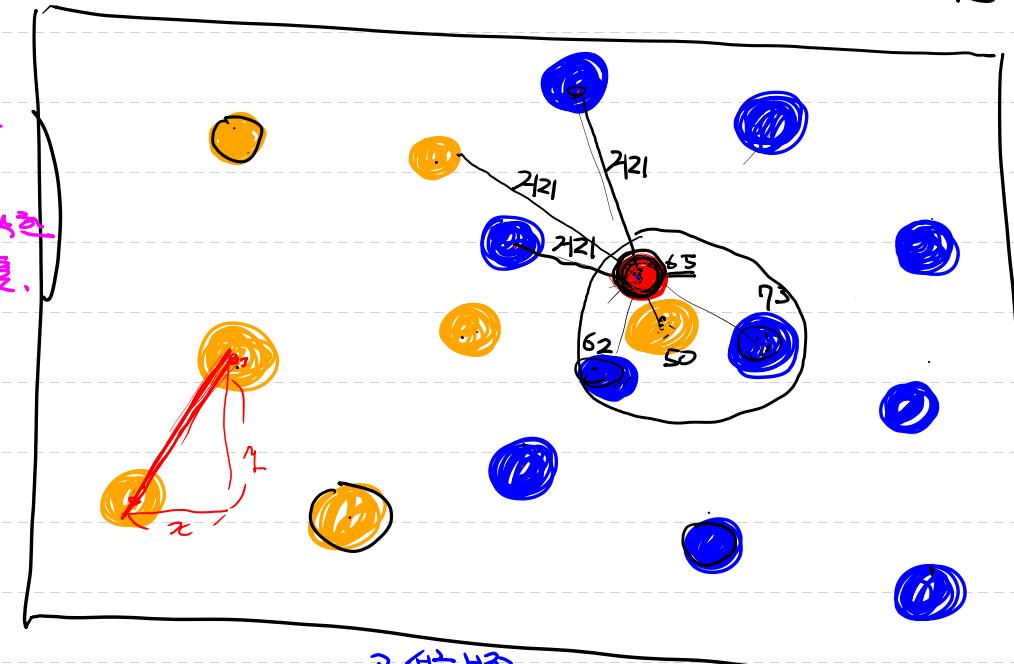
$K=3 \rightarrow$  판정

"반지 전처리"

종점: 데이터가

많으면  
선형회귀식의  
결과를 도출.

단점: 시간이 오래  
걸려요!!



" $K=1$ "

"1-NN 모자법칙이  
선형보조  
이상적인 모자법칙의 2배보다  
간거가 짧아요 → 설명"

분류쪽으로 설명

Regression에 기본으로  
KNN을 활용하는

KNN은 확률 학습이라는 틀로 X

→ 새로운 데이터가 들어왔을 때 기존 data들간의  
거리를 계산해 이웃을 뽑아서 예측을 수행.

\* Instance-based Learning  
Lazy Model

KNN의 두개의 hyperparameter

(1) 이웃의 수 ( $k$ ) →  $k$ 가 적을경우

(2) 거리 측정방식

지역적 특성을 너무 많이  
반영하는 Overfitting

$k$ 가 너무면  
Underfitting

① Euclidean distance

② Manhattan distance

③ Mahalanobis distance



"좌표축 방향으로만 이동"