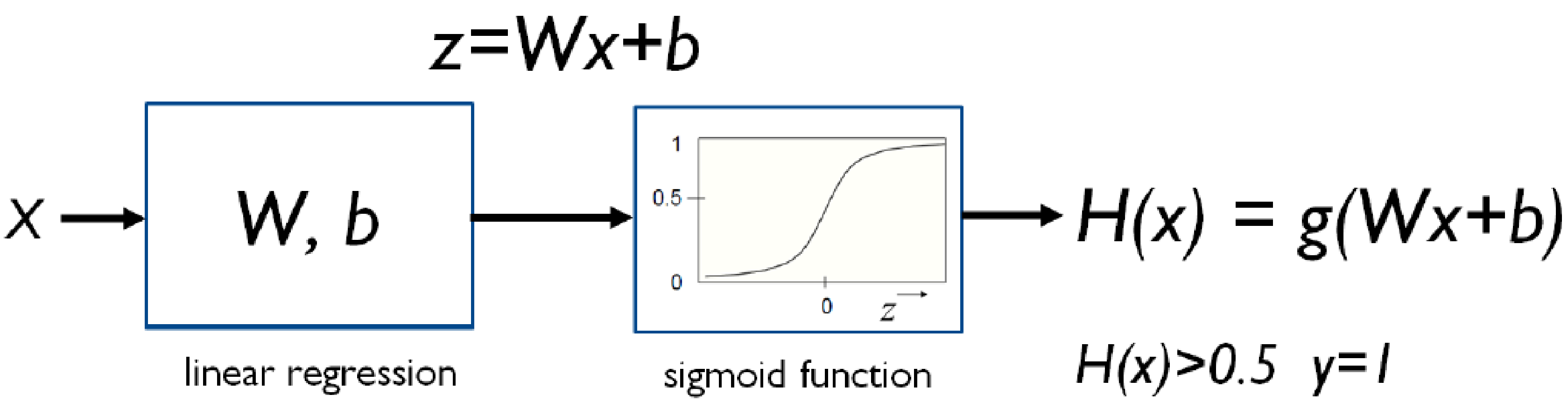
Lecture 5

Softmax classification:
Multinominal (Multi-class) classification

Dong Kook Kim

Logistic regression : Review

- Binary classification

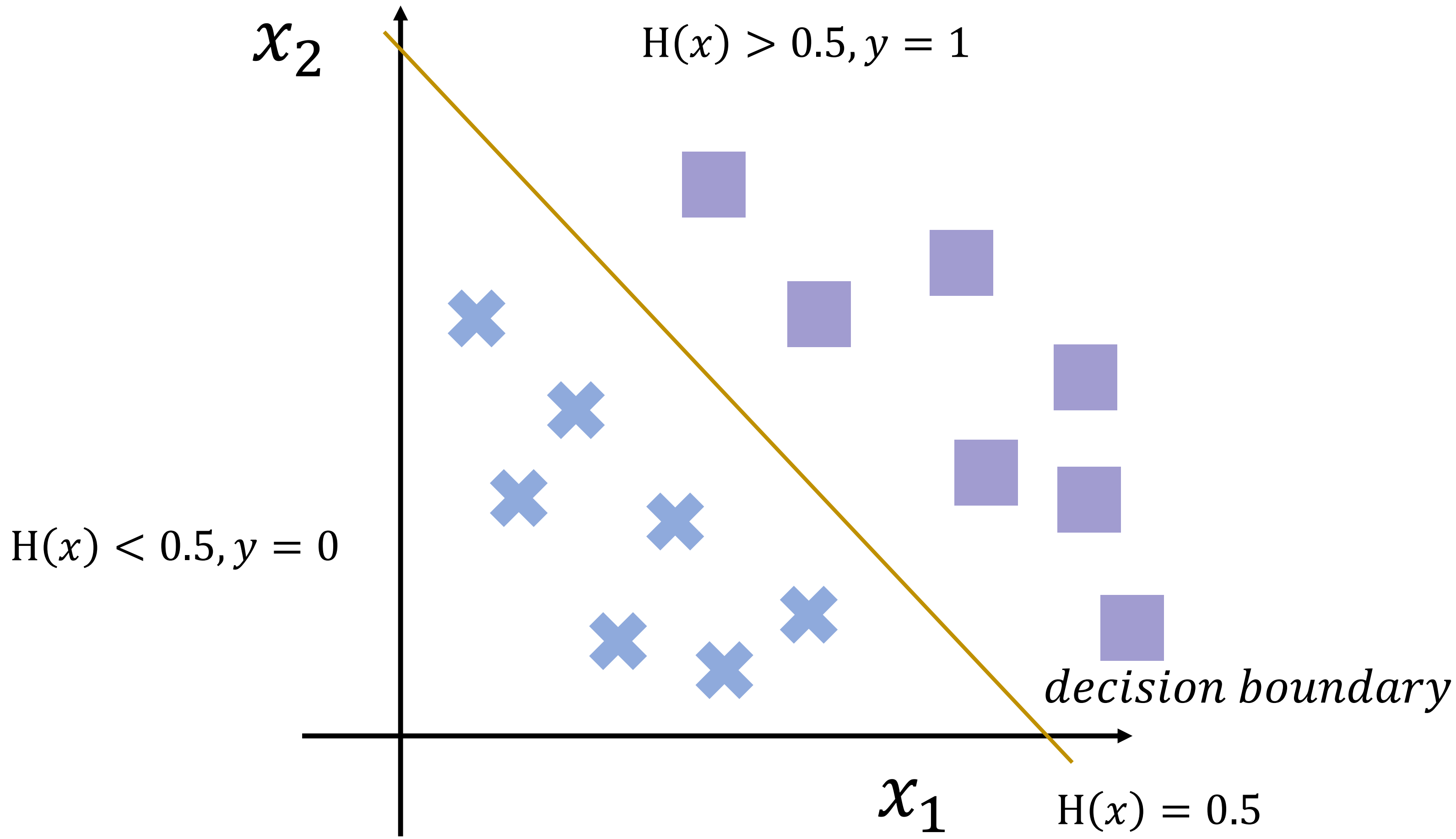


$H(x)>0.5 \quad y=1$

$H(x)<0.5, \; y=0$

$$\sigma(z) = \frac{1}{1+\exp(-z)}$$

Logistic regression : Review



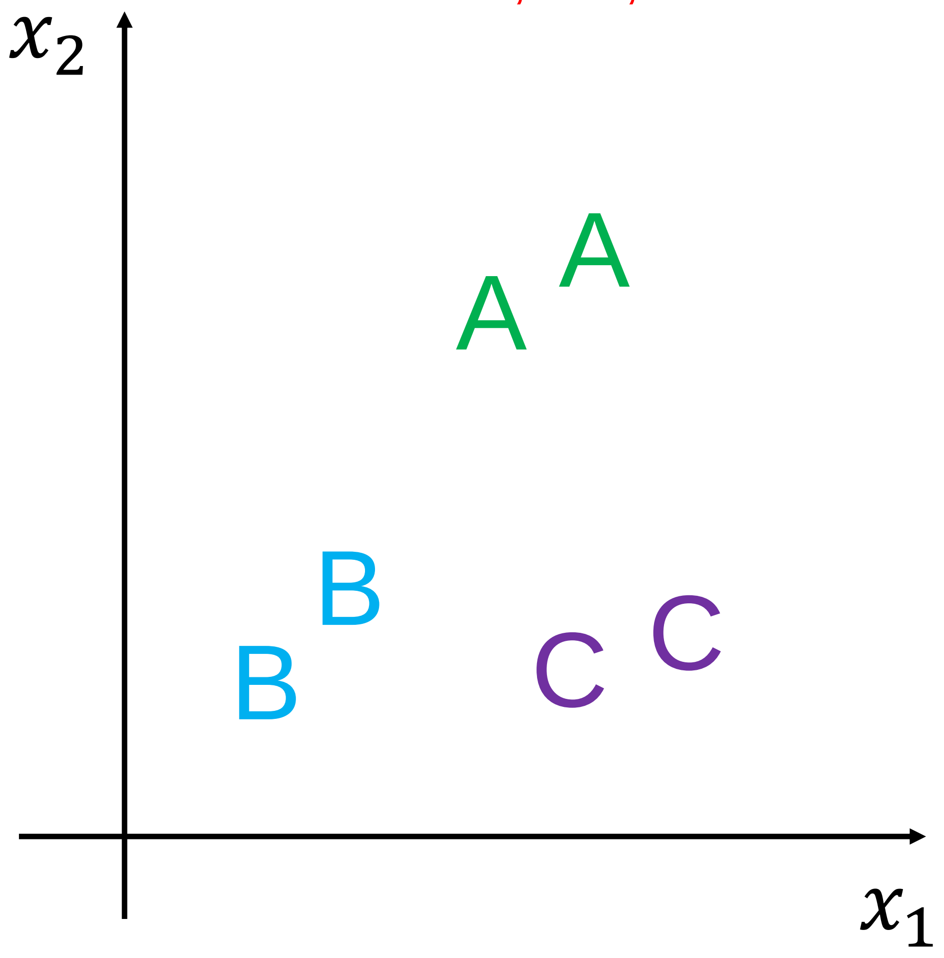
Multi-class classification

- Target: 1,2 ….K (integer), K-class

x1 (hours)	x2 (attendance)	y (grade)
10	5	A
9	5	A
3	2	B
2	4	B
11	1	C

input, target

3-class: 1-A, 2-B, 3-C



Integer Target : one-hot encoding

- one-hot or 1-of-K encoding:

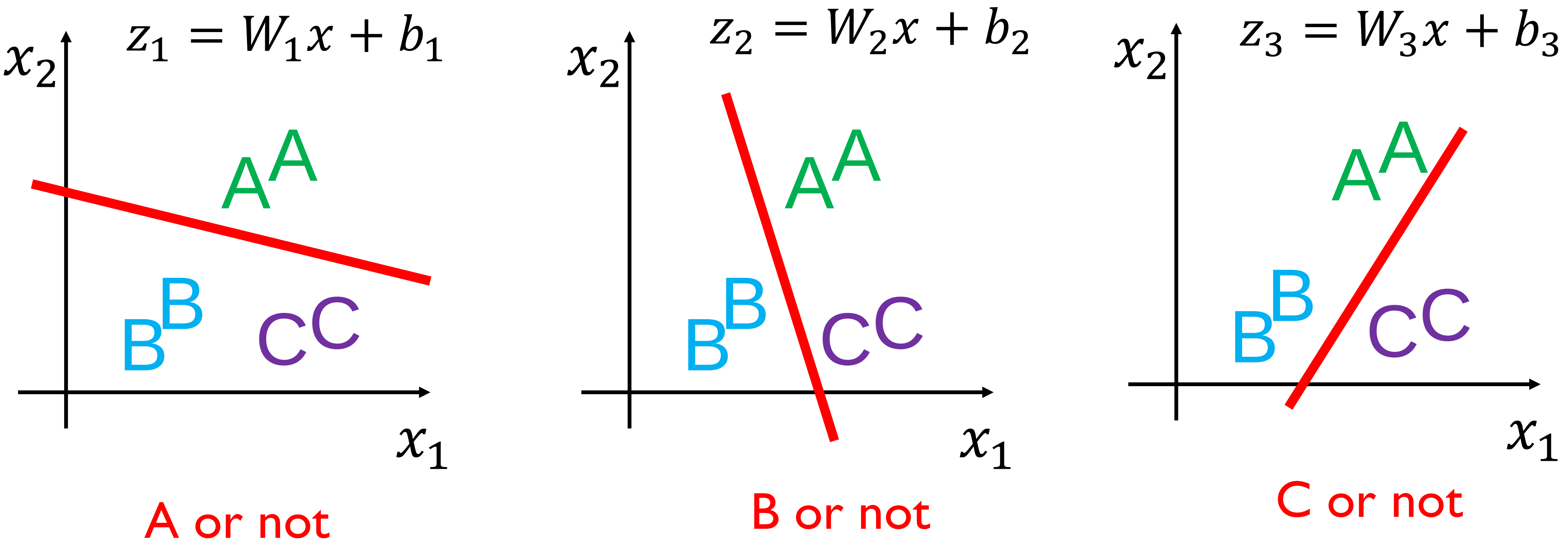
For multi-class problems (with K classes), instead of using t = k (target has label k) we often use a 1-of-K encoding, i.e., a vector of K target values containing a single 1 for the correct class and zeros elsewhere

Example: For a 4-class problem, we would write a target with class label 1,2,3,4 as:

$t = [1, 0, 0, 0]^T$	1
$t = [0, 1, 0, 0]^T$	2
$t = [0, 0, 1, 0]^T$	3
$t = [0, 0, 0, 1]^T$	4

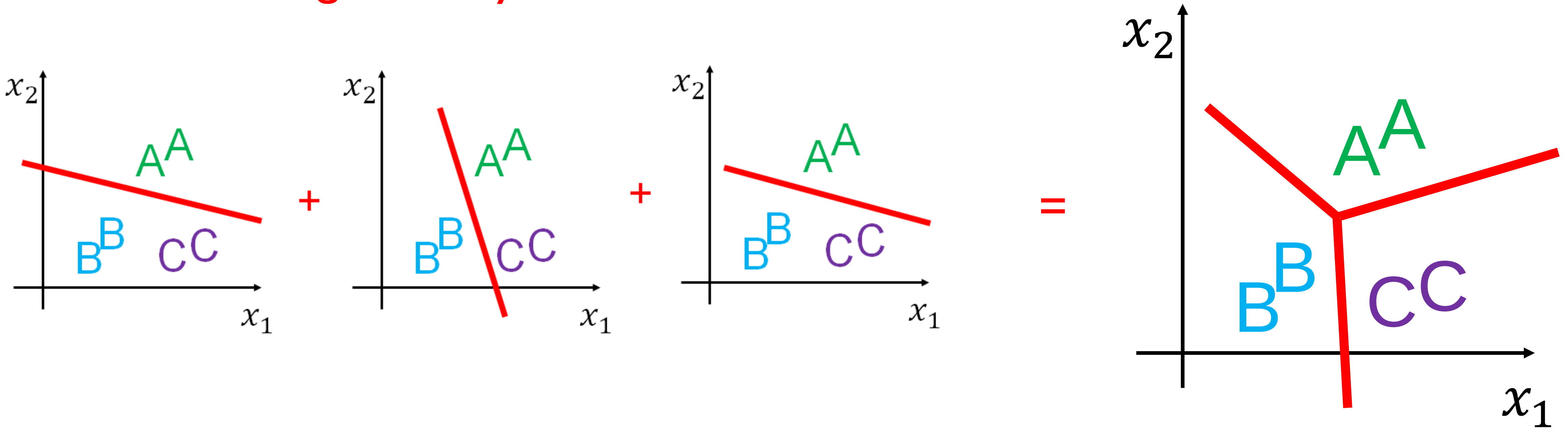
Multi-class Classification Approach

- Using 3 Binary classifier

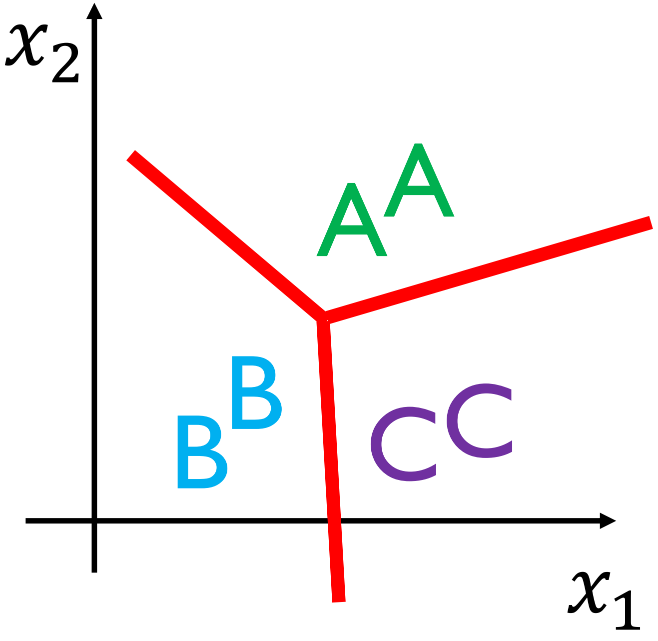


Multi-class Classification Approach

combing 3 binary classifier



Multi-class Classification Approach



$$\left.\begin{aligned} z_1 &= W_1 x + b_1 \\ z_2 &= W_2 x + b_2 \\ z_3 &= W_3 x + b_3 \end{aligned}\right\} \quad z = Wx + b$$

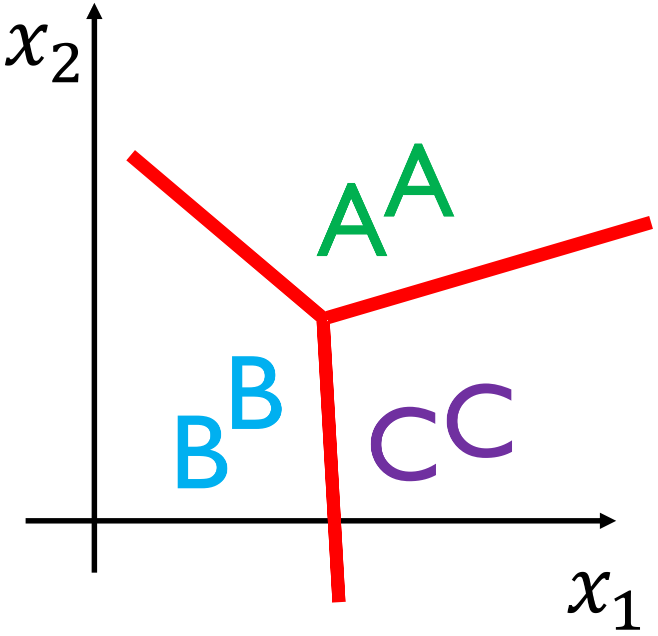
K linear regression

Multi-variable

linear regression

$$W = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \cdots & \ddots & \cdots \\ w_{k1} & \cdots & w_{kn} \end{bmatrix} \qquad b = \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix}$$

Multi-class Classification Approach



$$z = Wx + b$$

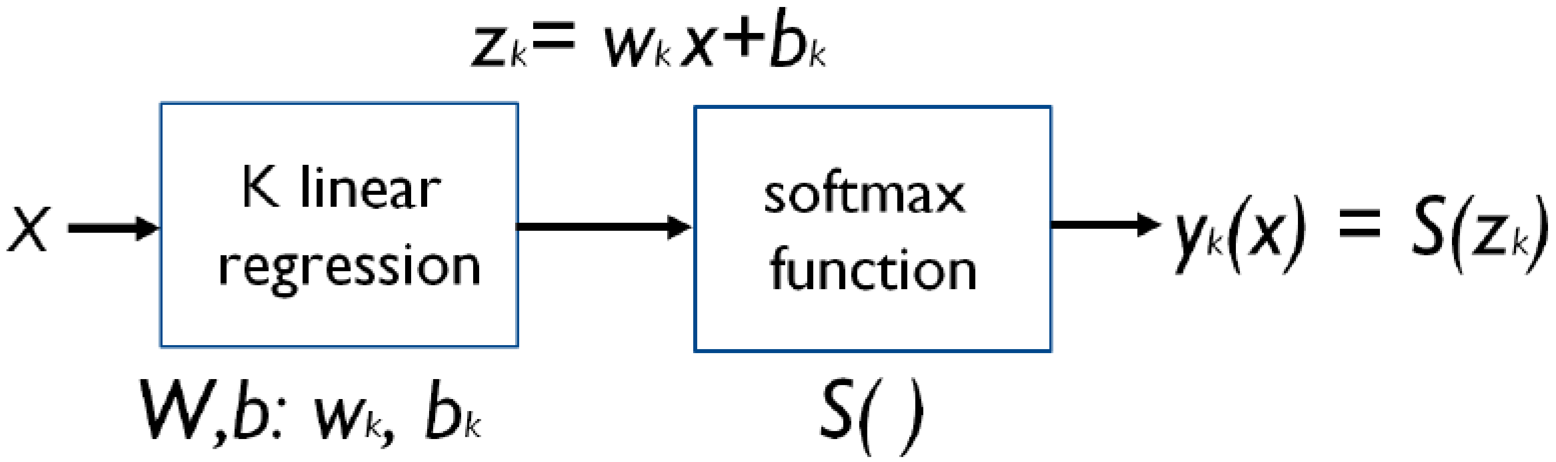
$$\begin{bmatrix} z_1 \\ \vdots \\ z_k \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \cdots & \ddots & \cdots \\ w_{k1} & \cdots & w_{kn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_1 \\ \vdots \\ b_k \end{bmatrix}$$

K linear regression: $z_k = W_k x + b_k$

Classify x to class $y = argmax_j \, S_j(z)$

Softmax Classification

- New approach : K linear regression + softmax function



Softmax

- Softmax Function : convert scores to probabilities

$$z_k = \mathbf{w}_k^T \mathbf{x}$$

scores

$$y_k(\mathbf{x}) = \frac{\exp(z_k)}{\sum_j \exp(z_j)}$$

: softmax function

probabilities

Softmax Classification : Example

$$\begin{bmatrix} W_{A1} & W_{A2} & W_{A3} \\ W_{B1} & W_{B2} & W_{B3} \\ W_{C1} & W_{C2} & W_{C3} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} W_{A1}x_1 + W_{A2}x_2 + W_{A3}x_3 \\ W_{B1}x_1 + W_{B2}x_2 + W_{B3}x_3 \\ W_{C1}x_1 + W_{C2}x_2 + W_{C3}x_3 \end{bmatrix} = \begin{bmatrix} z_A \\ z_B \\ z_C \end{bmatrix} \begin{array}{c} 2.0 \\ 1.0 \\ 0.1 \end{array}$$

Softmax Function

$$z \begin{bmatrix} 2.0 \\ 1.0 \\ 0.1 \end{bmatrix} \rightarrow S(z_i) = \frac{e^{z_i}}{\sum_j e^{z_i}} \begin{matrix} \rightarrow 0.7 \\ \rightarrow 0.2 \\ \rightarrow 0.1 \end{matrix}$$

$$Scores\ (logit) \rightarrow \quad Probabilities$$

Softmax Function

$$z \left[\begin{matrix} 2.0 \\ 1.0 \\ 0.1 \end{matrix}\right. \xrightarrow{S(z)} \begin{matrix} 0.7 \\ 0.2 \\ 0.1 \end{matrix} \xleftrightarrow{argmax} \begin{matrix} 1.0 \\ 0.0 \\ 0.0 \end{matrix}$$

'One-Hot' Encoding Target

Loss Function : Softmax Classification

- Cross-Entropy

Output (0~1)

'One-Hot' Encoding Target

$$\begin{bmatrix}0.7\\0.2\\0.1\end{bmatrix} \quad D(S,L) = -\sum_i L_i \log(S_i) \quad \begin{bmatrix}1.0\\0.0\\0.0\end{bmatrix}$$

$S(z)$

$L = Y$

- Keras : categorical_crossentropy
- model.compile(loss='categorical_crossentropy',optimizer='adam', metrics=['accuracy'])

Cross-Entropy Loss Fucntion

- **3-class**

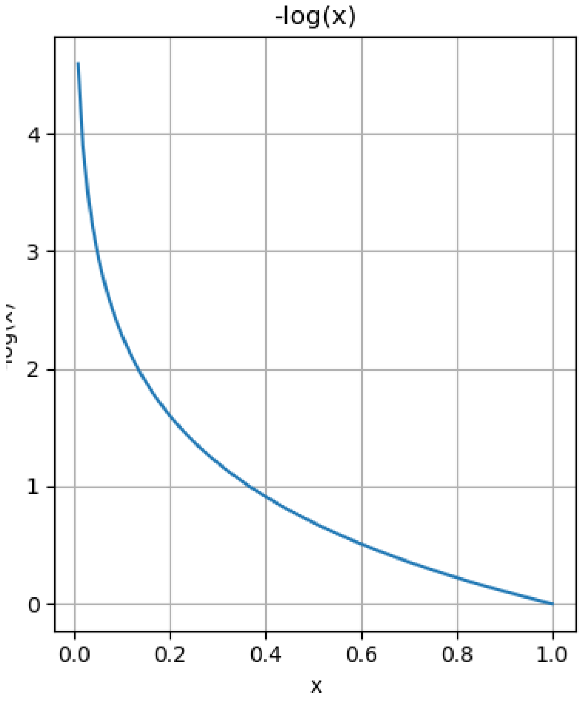
$$D(S,L) = -\sum_i L_i \log(S_i)$$

$$L = \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \end{bmatrix}$$

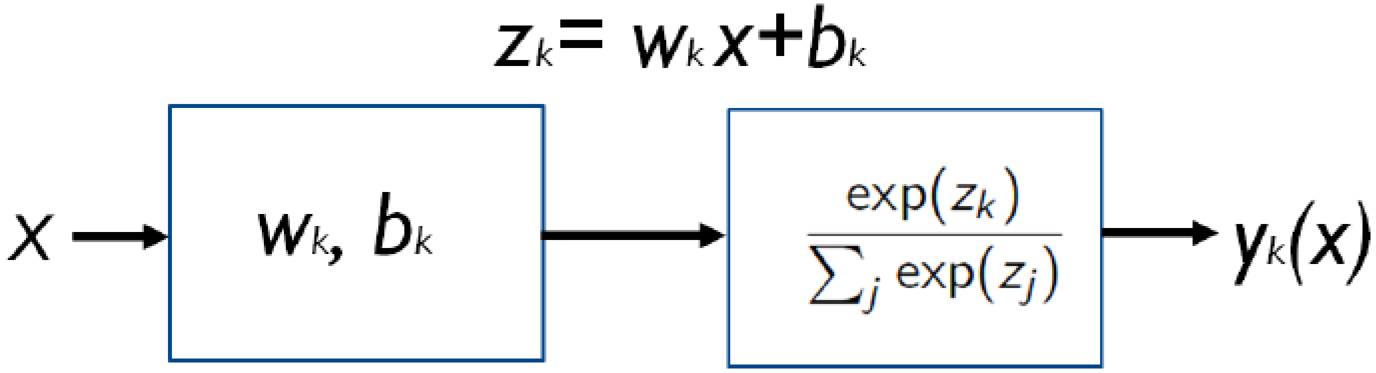
$$S = \begin{bmatrix} 0.0 \\ 1.0 \\ 0.0 \end{bmatrix} \quad D(S,L) = \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \end{bmatrix} \circ log \begin{bmatrix} 0.0 \\ 1.0 \\ 0.0 \end{bmatrix} = \infty$$

$$S = \begin{bmatrix} 0.0 \\ 0.0 \\ 1.0 \end{bmatrix} \quad D(S,L) = \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \end{bmatrix} \circ log \begin{bmatrix} 0.0 \\ 0.0 \\ 1.0 \end{bmatrix} = \infty$$

$$S = \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \end{bmatrix} \quad D(S,L) = \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \end{bmatrix} \circ log \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \end{bmatrix} = 0$$



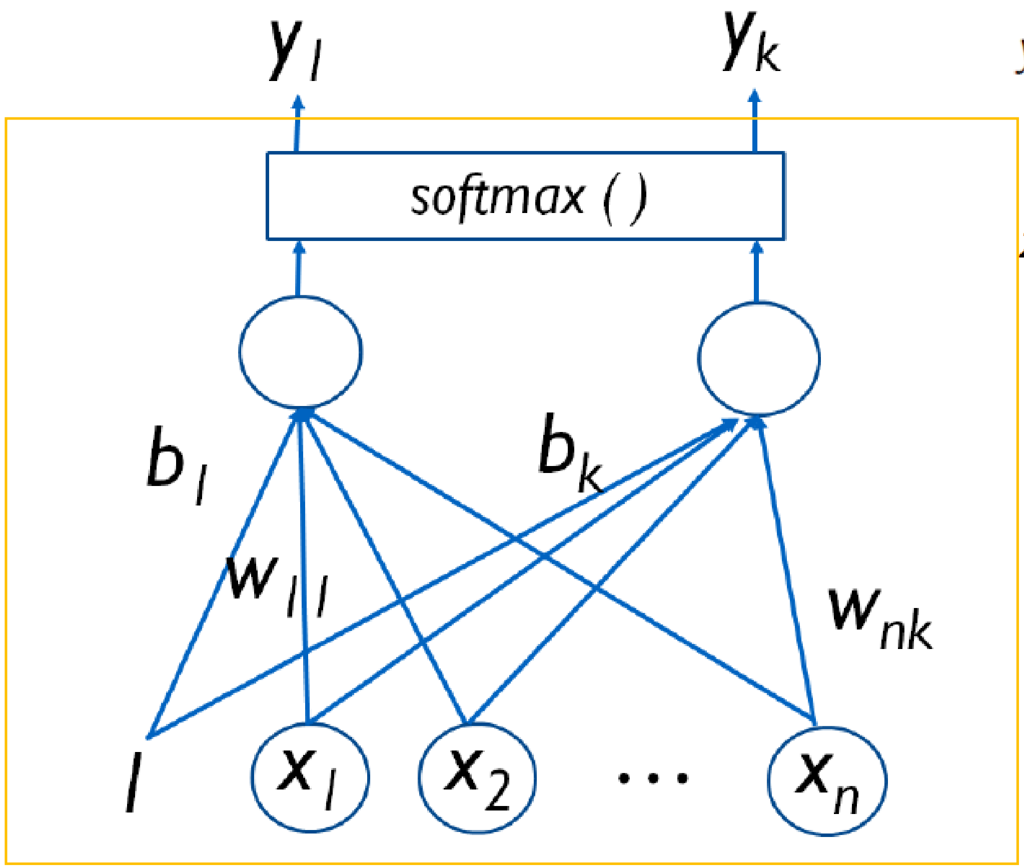
Softmax Classification : Decision



- Decision Rule for multi-class classification

$$Decide\ jth\ class, j = argmax_k\ S_k(x)$$

Graph for Softmax Classification



$$y_k(\mathbf{x}) = \frac{\exp(z_k)}{\sum_j \exp(z_j)}$$

$$z_k = \mathbf{w}_k^T \mathbf{x}$$

Keras : Dense(k, input_dim=n, activation='softmax')

Softmax Classification : Summary

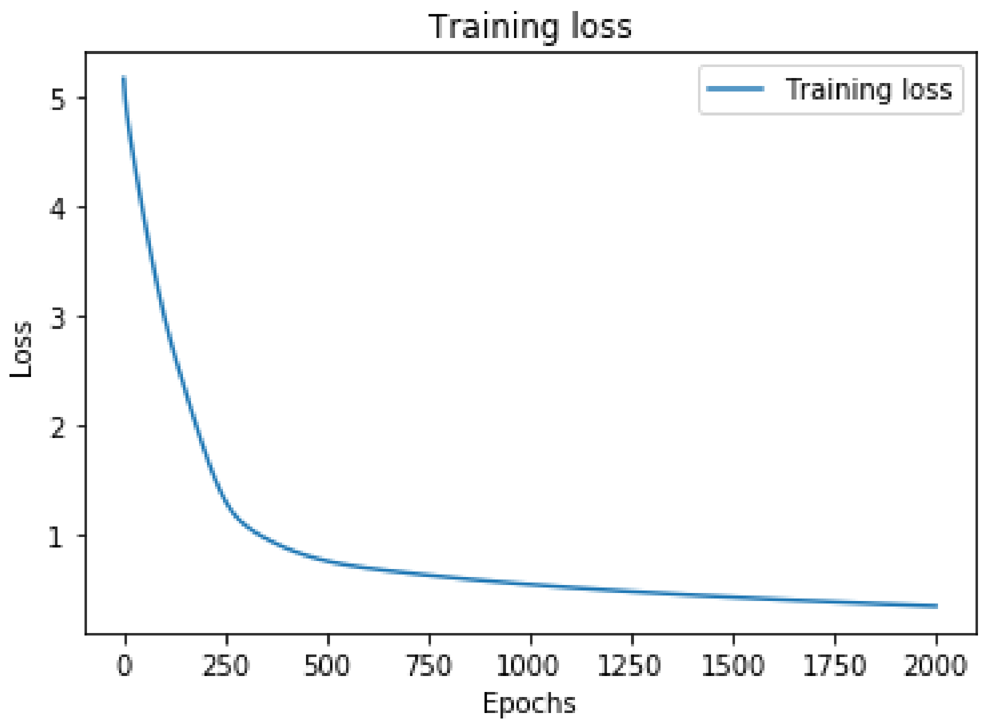
Data Set :	$x^{(i)}, y^{(i)}, i = 1, \dots, m$	Input vector, $y^{(i)}$: one-hot encoding vector
Model :	$z_k = W_k x + b_k, \qquad S_k(x) = \frac{\exp(z_k)}{\sum_j \exp(z_j)}$	Softmax function W : weight matrix b : bias vector
Cost Function	$cost(W, b) = -\frac{1}{m} \sum_{i,k} {y_k}^{(i)} \log(S_k(x^{(i)}))$	CE
Optimization	$W := W - \alpha \frac{\partial cost(W, b)}{\partial W}$	GD
Testing	Given x & $w_k, b_k, decide\ jth\ class, j = argmax_k\, S_k(x)$	Metric : Accuracy(%)

Exercise 04-1.

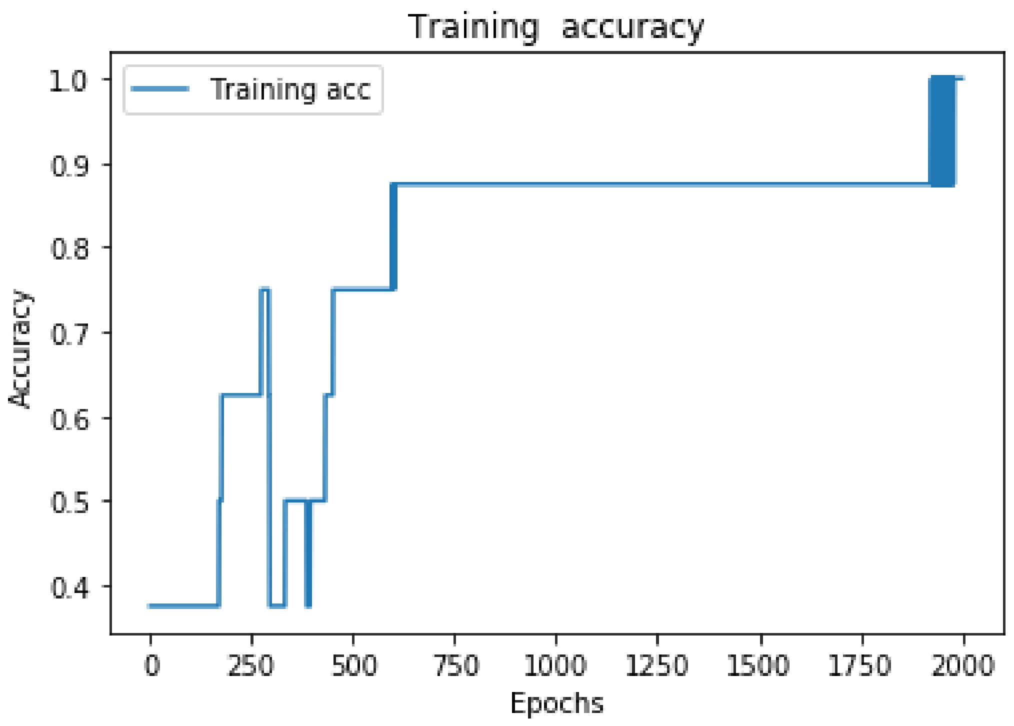
tf2-04-1-softmax.py

Exercise 04-1.

- Training loss



- Training accuracy

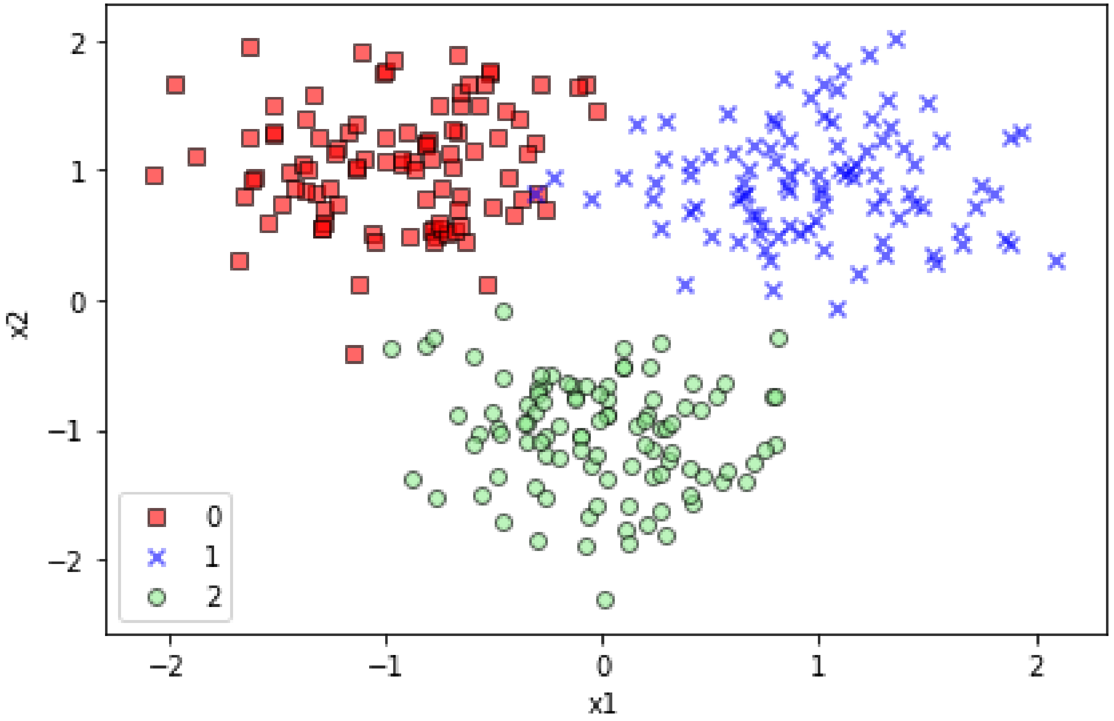


Exercise 04-2.

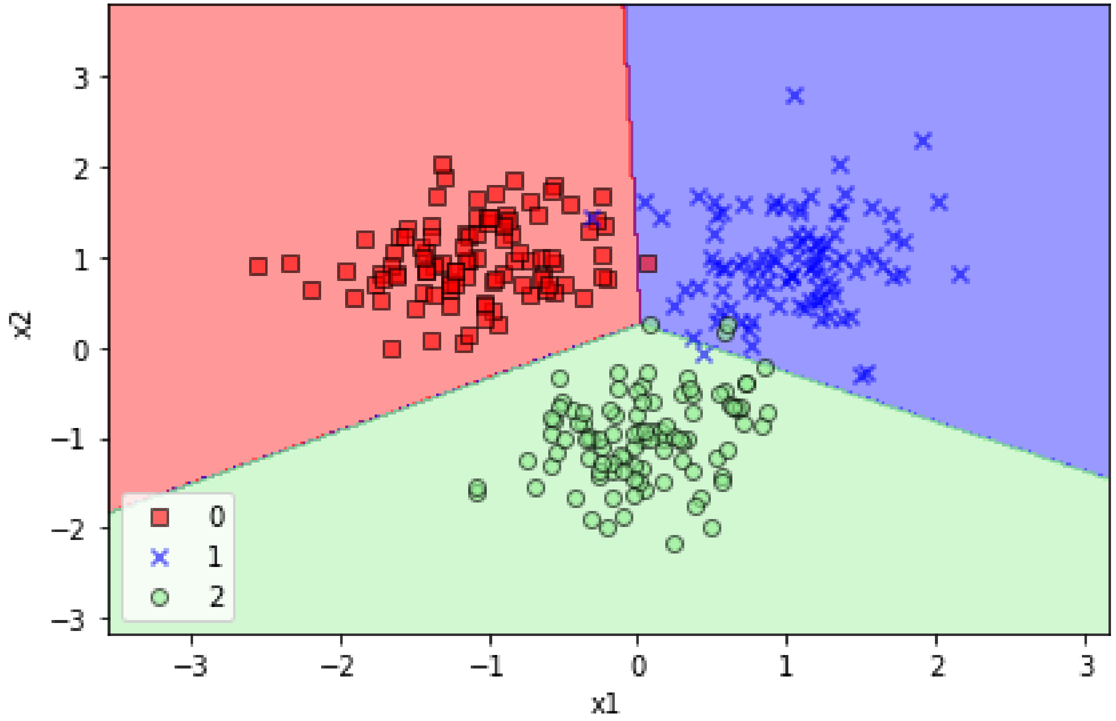
tf2-04-2-softmax_3classes.py

Exercise 04-2.

- Input and target



- Classification results

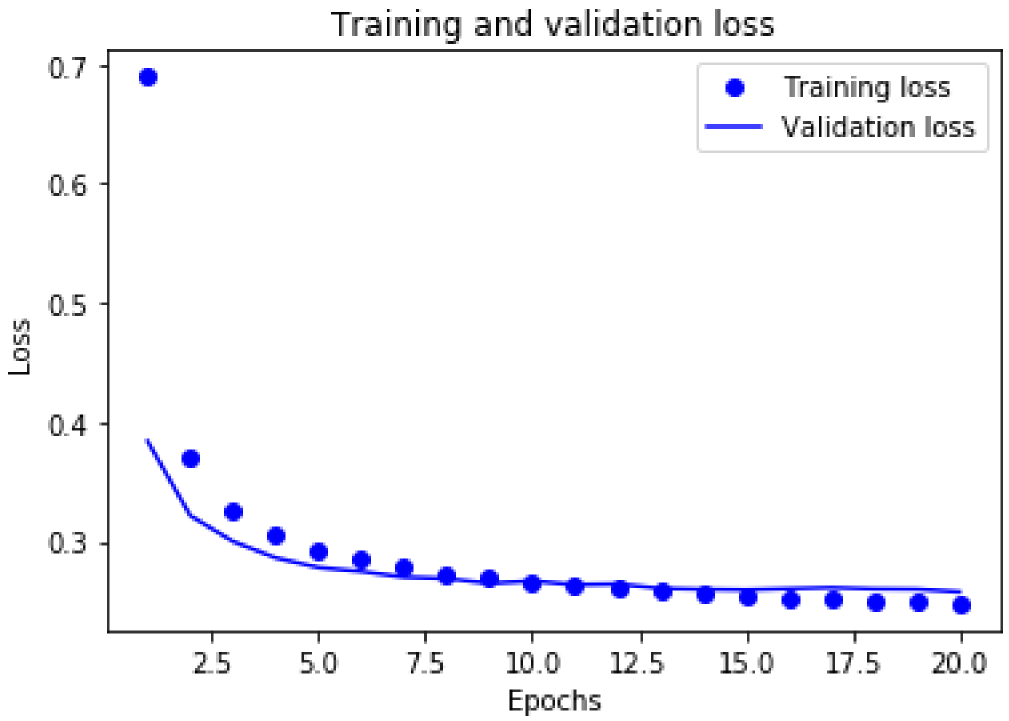


Exercise 04-3.

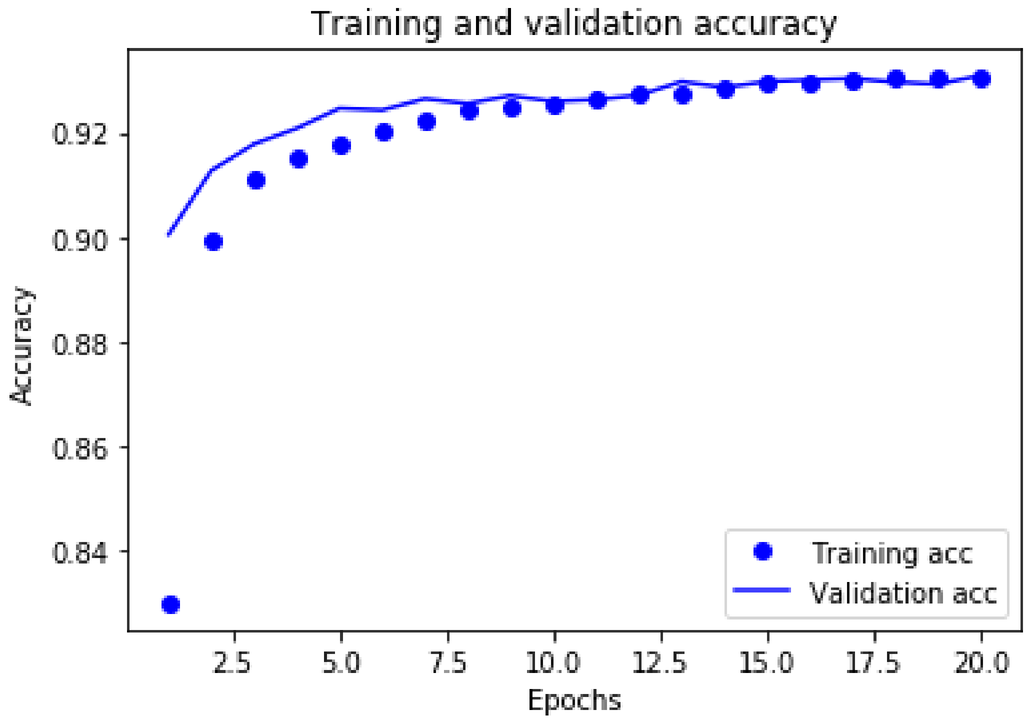
tf2-04-3-softmax_mnist.py

Exercise 04-3.

- Training and validation loss



- Training and validation accuracy



- Test accuracy : 0.9268

Linear Models : Summary

Supervised Learning	Input x	Target y	Distribution for Target	Classifier p(y|x)	Linear models	Loss function
Regression	vector binary integer real	real	Gaussian	Gaussian	Linear regression	MSE
Binary Classification		binary	Bernoulli	sigmoid	Logistic classification	Binary Cross-entropy
Multi-class Classification		integer	multinoulli	softmax	Softmax classification	Multi-class Cross-entropy