[SWCON253] Machine Learning – Lec.11

Multiclass & Softmax

Fall 2025

김휘용

경희대학교
KYUNG HEE UNIVERSITY

Contents

1. **Multiclass Classification**
2. **Softmax Classifier**
3. ***Summary: Gradient of MSE & CE Losses***

References

- *http://stat.wisc.edu/~sraschka/teaching*
- *https://en.wikipedia.org/wiki/Multiclass_classification*
- *https://en.wikipedia.org/wiki/One-hot*
- *기계 학습* by 오일석

1. Multi-Class Classification

- ✓ Categorical Data
- ✓ Multiclass Classification
- ✓ (Recap) Single Neuron Models for a Binary Classifier
- ✓ Multiclass Classification with Multiple Binary Classifiers
- ✓ One-vs-Rest (One-vs-All) Method
- ✓ One-vs-One Method

Categorical Data

- **Numerical vs. Categorical Variables**
  - **Numerical** (quantitative) variables:
    - ★ e.g. price, height, weight, image pixel values
    - ★ it has some *order*
  - **Categorical** (qualitative) variables
    - ★ e.g. object category, blood type, roll of a die
    - ★ to assign each individual observation to a particular *nominal category* on the basis of some qualitative property
    - ★ take on one of a limited number of possible values (i.e., *numerical indices*)
    - ★ it does *not* have order

- **Examples**
  - IRIS dataset
    - ★ in: length, width → numerical
    - ★ out: species → categorical (nominal)
  - MNIST dataset
    - ★ in: pixel values → numerical
    - ★ out: digits → categorical (nominal)
  - ImageNet dataset
    - ★ in: pixel values → numerical
    - ★ out: object categories → categorical (nominal)

Multiclass Classification

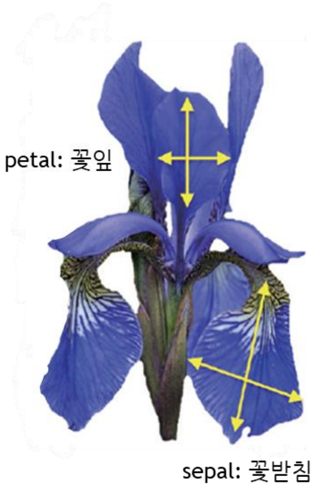
◆ Multiclass (Multinomial) Classification

- The problem of classifying instances into one of three or more ***classes***.
  - ★ The output is categorical (e.g., Iris, MNIST, ImageNet)
- The label is no longer binary. It is multinomial: e.g. $y \in \{1, 2, 3, \ldots, c\}$.

C = 21,841

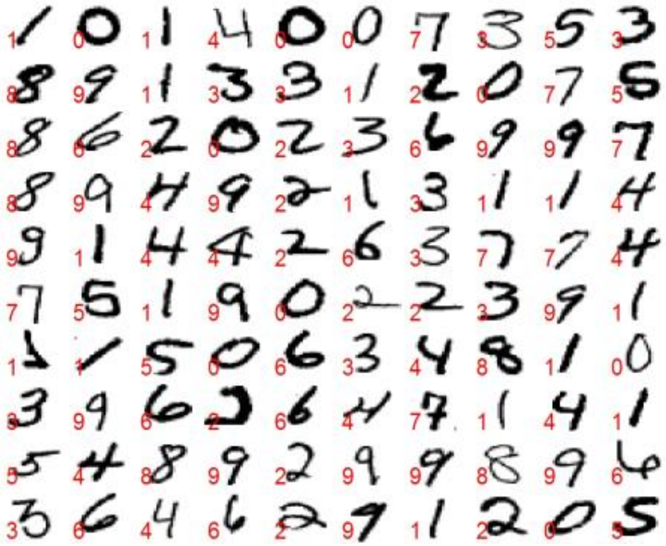
(a) 'swing' 부류

(b) 'Great white shark' 부류



Classes: {Virginica, Versicolor, Setosa}

C = 3



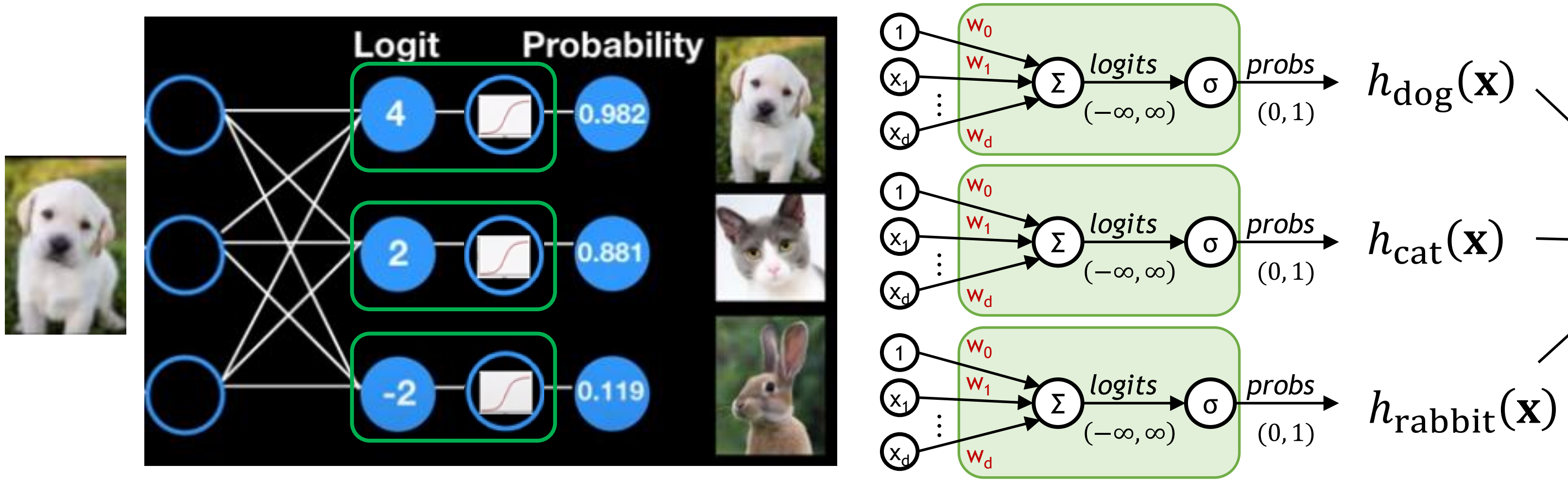
Classes: ?

C = 10

Multiclass Classification with Multiple Binary Classifiers

One-versus-Rest (One-versus-All) Method

- 이진 분류기 c개를 독립적으로 사용하여 class k와 나머지 c-1개 class를 분류 (1 : c-1)
- Class k에 대한 이진 분류기를 h_k라 하면, $h_k(\mathbf{x})$가 가장 큰 값을 갖는 k로 분류함

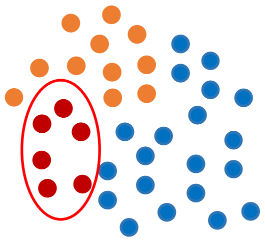


$$\boxed{\hat{k} = \arg\max_k h_k(\mathbf{x})}$$

Multiple binary classifiers (e.g., logistic regression)

Remarks

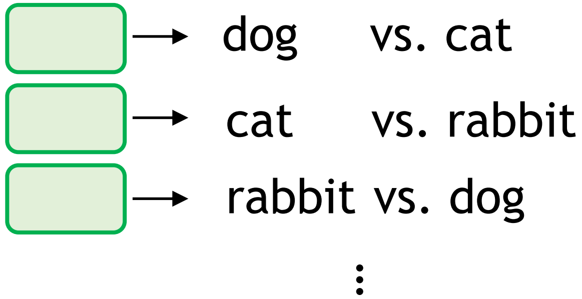
- 필요한 이진 분류기의 개수: c개
- 각 이진 분류기에 대해 훈련집합의 불균형을 일으킴 (class k 샘플수 ≪ 나머지 샘플수)



Multiclass Classification with Multiple Binary Classifiers (cont'd)

One-versus-One Method

- 이진 분류기 $C(c,2)$개를 독립적으로 사용하여 class k와 class l을 분류 (1 : 1)
  - ★ $C(c,2) = \dfrac{c!}{(c-2)!\,2!} = c(c-1)/2$



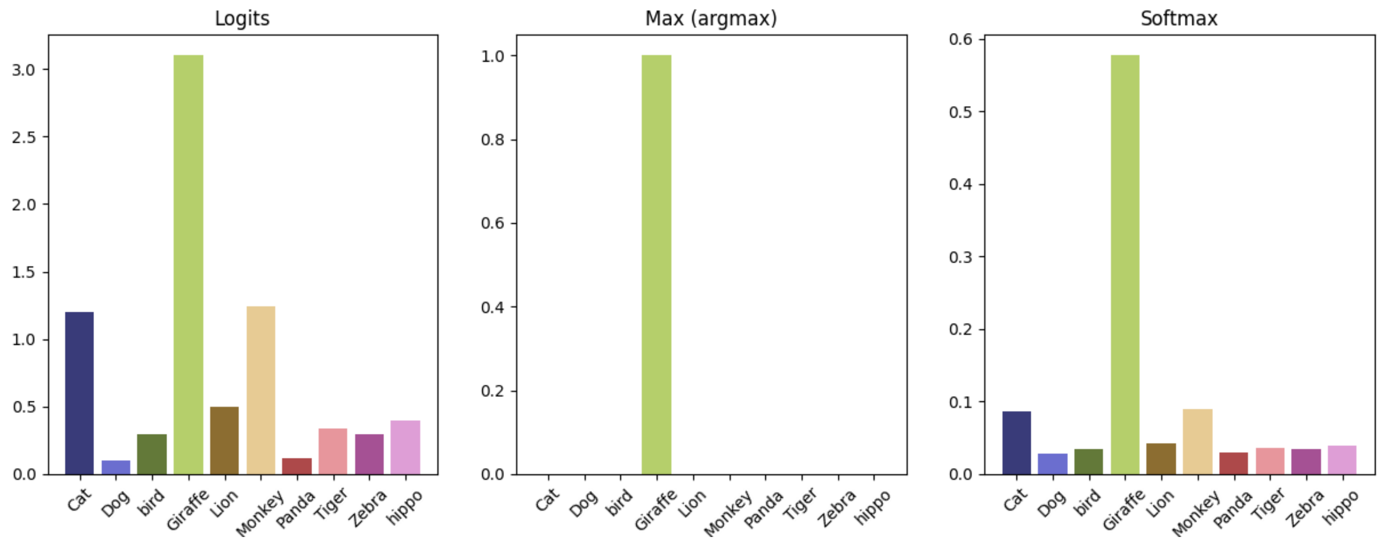
- 가장 많은 이진 분류기가 선택(투표)한 class를 최종 결과로 결정
  - ★ Class k와 l을 비교하는 이진 분류기를 $h_{(k,l)}(\mathbf{x})$라 하자.
  - ★ $h_{(k,l)}(\mathbf{x})$가 class k(또는 l)를 출력하면, class k(또는 l)에 한 표를 추가.
  - ★ C(c, 2)개 이진 분류기에 대해 가장 많은 표를 획득한 class를 최종 결과로 결정 (최대 표의 개수: c-1)
    - 비유) 야구나 축구 리그에서 가장 승리를 많이 한 팀이 우승

Remarks

- 훈련집합의 불균형을 일으키지 않음: class k 샘플수 ≈ class l 샘플수
- 사용되는 이진 분류기의 개수: $c(c-1)/2$ → c^2에 비례: 높은 training/testing 복잡도

2. Softmax Classifier

- ✓ Softmax – Motivation & Definition
- ✓ One-Hot Encoding
- ✓ Cross-Entropy Loss
- ✓ Training Softmax Classifier with CE



(Recap.) One-Hot Encoding

- **One-Hot Code (One-Hot Vector)**
  - A group of bits among which the legal combinations of values are only those with ***a single high (1) bit*** and all the others low (0)
    - ★ A similar implementation in which all bits are '1' except one '0' is sometimes called "***one-cold code***".
  - One-hot Encoding is frequently used to deal with ***categorical data***
    - ★ because many ML models need variables to be numeric

Binary	Gray code	One-hot
000	000	00000001
001	001	00000010
010	011	00000100
011	010	00001000
100	110	00010000
101	111	00100000
110	101	01000000
111	100	10000000

- **One-Hot Encoding in ML**
  - k번째 class의 target vector를 k번째 자리는 1, 나머지는 0이 되도록 설정
  - Cross Entropy 계산에 적합해 짐: target **y**의 원소들의 합이 1이 되므로 각 원소를 그 class의 정답확률로 볼 수 있다.

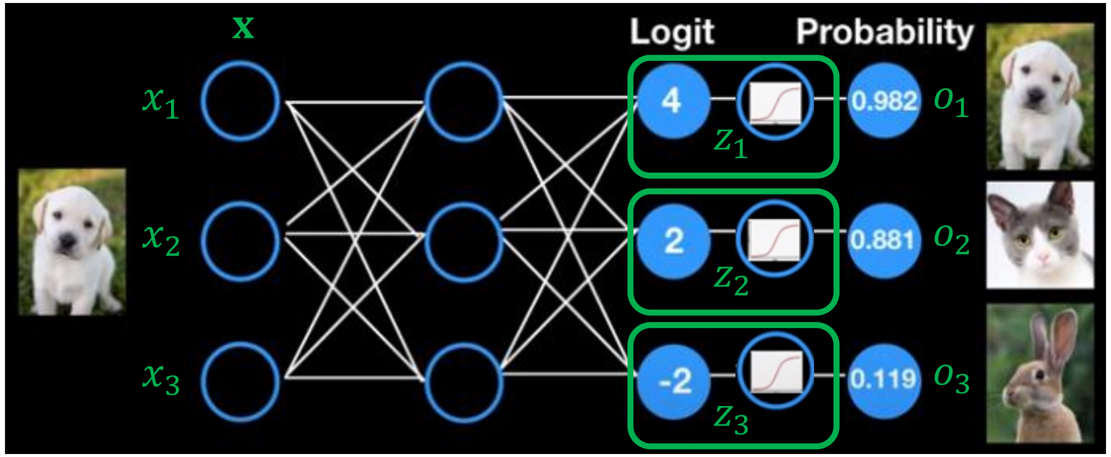
Index	Job
1	Police
2	Doctor
3	Student
4	Teacher
5	Driver

⇒

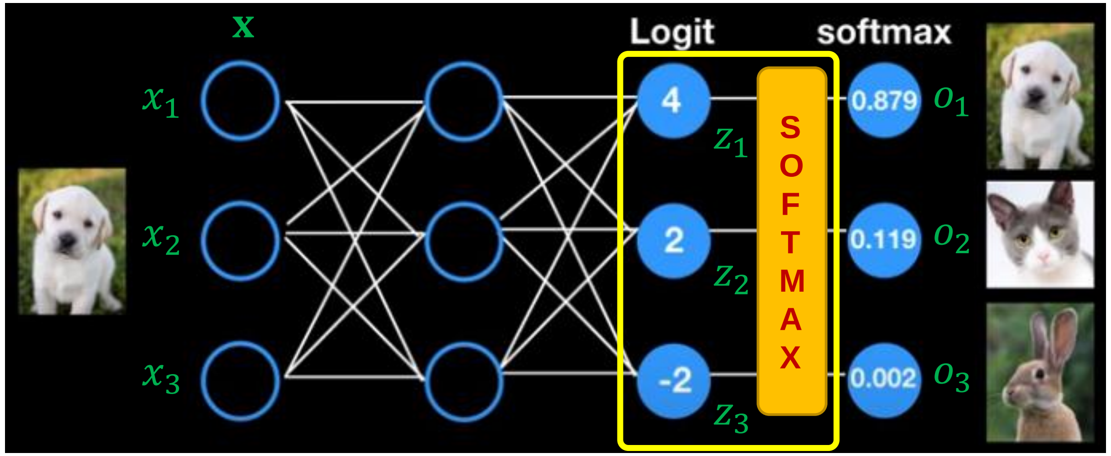
One hot encoded data
[1 0 0 0 0]
[0 1 0 0 0]
[0 0 1 0 0]
[0 0 0 1 0]
[0 0 0 0 1]

Softmax – *Motivation*

- What is the best way to convert (-∞, ∞) to probability for multiclass classification?
  - ***What we want*** in the output layer
    - ★ conditional probabilities
      - $o_i = p(y_i = 1|\mathbf{x})$
  - ***Sigmoid*** activations in the output layer
    do ***not*** sum up to 1 ($\sum_1^K o_i \neq 1$)
    - ★ Is it suitable for *Cross-Entropy Loss*?
  - ***Softmax*** activations in the output layer
    do sum up to 1 ($\sum_1^K o_i = 1$)
    - ★ 출력벡터의 원소들의 합이 1이 되므로
      각 원소를 그 class의 확률 추정치로 볼 수 있다.
    - ★ Suits well to *Cross-Entropy Loss*



Multiple binary classifiers



A single multinomial classifier

Figures (modified): https://www.youtube.com/watch?v=K7HTd_Zgr3w

Softmax – *Definition*

(Remind) Sigmoid $\sigma : \mathbb{R} \mapsto (0,1) \quad \sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$

Softmax Function

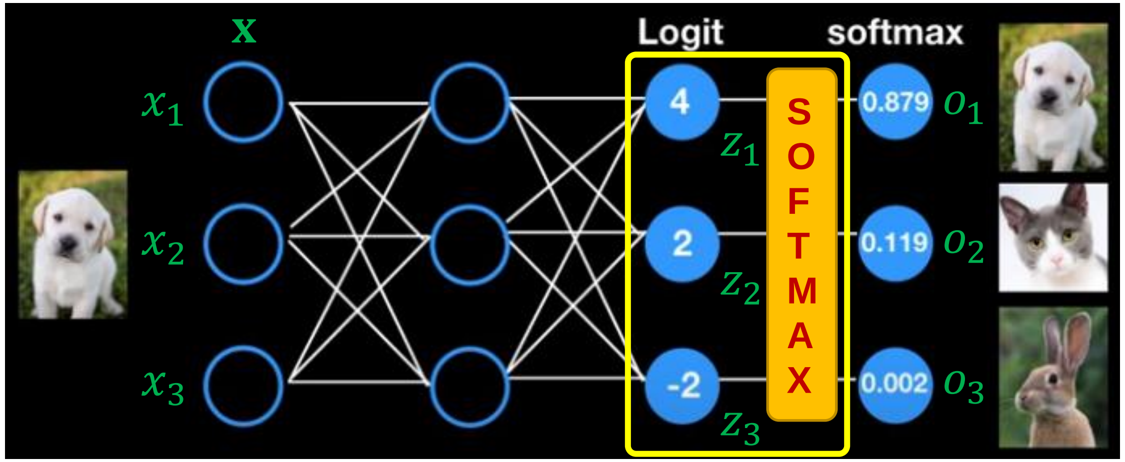
- Takes as input a vector **z** of *K* real numbers,
- and normalizes it into a probability distribution consisting of *K* probabilities

$$\sigma : \mathbb{R}^K \to (0,1)^K \qquad \sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \qquad \text{for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$$

$$\Rightarrow \sum_{i=1}^{K} \sigma_i(\underline{z}) = 1$$

$$\begin{bmatrix} 1.20 \\ 0.10 \\ 0.30 \\ \mathbf{3.10} \end{bmatrix} \xrightarrow{\textbf{Softmax}} \begin{bmatrix} 0.119 \\ 0.040 \\ 0.048 \\ \mathbf{0.794} \end{bmatrix}$$

$$\frac{e^{3.10}}{e^{1.20} + e^{0.10} + e^{0.30} + e^{3.10}}$$



Cross-Entropy Loss

- **Assume One-Hot Encoding & Softmax at Output Layer**
  - one-hot-encoded target (y_k's are either 0 or 1) → y_k can be treated as ground-truth probability of class-k
  - softmax at output layer (o_k's are summed to 1) → o_k can be treated as predicted probability of class-k

- ***Binary CE*** **(for binary classification)**

$$\frac{1}{N}\sum_{n=1}^{N}\{-y^{(n)}\log o^{(n)} - (1-y^{(n)})\log(1-o^{(n)})\} = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{2} -y_i^{(n)}\log o_i^{(n)}$$

$$o_1^{(n)} = o^{(n)}$$
$$o_2^{(n)} = (1-o^{(n)})$$
$$y_1^{(n)} = y^{(n)}$$
$$y_2^{(n)} = (1-y^{(n)})$$

- ***Multinomial CE*** **(for multiclass classification)**

$$\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{K} -y_i^{(n)}\log o_i^{(n)}$$

For example,

$$\begin{pmatrix} o_1 \\ o_2 \\ o_3 \\ o_4 \end{pmatrix} = \begin{pmatrix} 0.1 \\ 0.4 \\ 0.2 \\ 0.3 \end{pmatrix} \quad \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

$$\sum_{i=1}^{K} -y_i \log o_i = -0\cdot\log(0.1) - 1\cdot\log(0.4) - 0\cdot\log(0.2) - 0\cdot\log(0.3) = -\log(0.4)$$

Training Softmax Classifier with CE

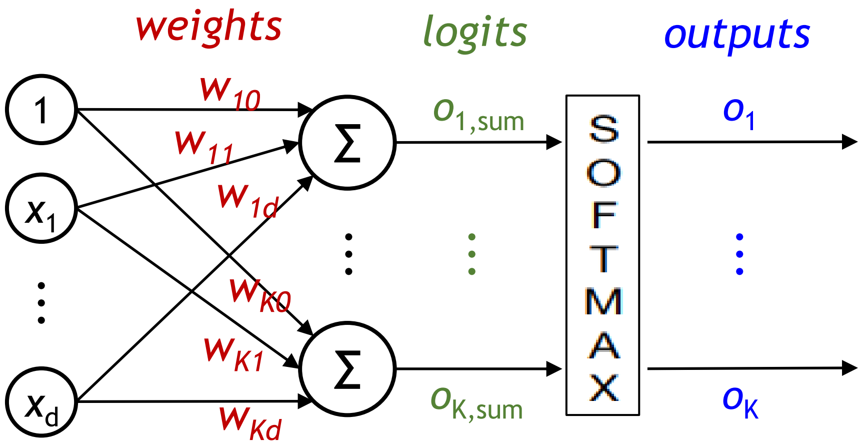
Gradient of the Cross-Entropy Loss at the Output Layer

- Cross-Entropy Loss

$$J(\mathbf{W}) = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{K} -y_i^{(n)} \log o_i^{(n)}$$

$$= \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{K} -y_i^{(n)} \log \frac{\exp(\mathbf{w}_i^T \mathbf{x}^{(n)})}{\sum_{j=1}^{K}\exp(\mathbf{w}_j^T \mathbf{x}^{(n)})}$$

$z_k = o_{k,sum}$



- It's Gradient

$$\frac{\partial J(\mathbf{W})}{\partial \mathbf{w}_k} = \frac{1}{\ln 2}\frac{1}{N}\sum_{n=1}^{N}\left(o_k^{(n)} - y_k^{(n)}\right)\mathbf{x}^{(n)}$$

$$= \frac{1}{\ln 2}\frac{1}{N}\sum_{n=1}^{N}\left(\frac{\exp(\mathbf{w}_k^T \mathbf{x}^{(n)})}{\sum_{j=1}^{K}\exp(\mathbf{w}_j^T \mathbf{x}^{(n)})} - y_k^{(n)}\right)\mathbf{x}^{(n)}$$

base=2 일 경우

$$\frac{\partial \log o_i}{\partial z_k} = \frac{1}{\ln 2}\frac{\partial}{\partial z_k}\ln\frac{\exp(z_i)}{\sum_{j=1}^{K}\exp(z_j)} \qquad z_k = o_{k,sum}$$

$$= \frac{1}{\ln 2}\frac{\partial}{\partial z_k}\left(\log\exp(z_i) - \log\textstyle\sum_{j=1}^{K}\exp(z_j)\right)$$

$$= \frac{1}{\ln 2}\left(1\{i=k\} - \frac{\exp(z_k)}{\sum_{j=1}^{K}\exp(z_j)}\right) = \frac{1}{\ln 2}\left(1\{i=k\} - o_k\right)$$

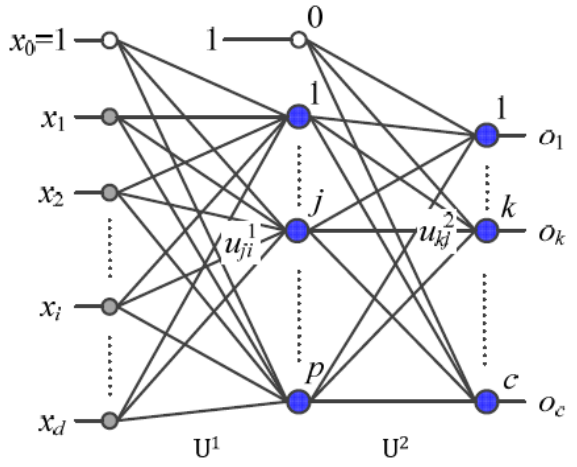
$$\therefore \sum_{i=1}^{K} -y_i\frac{\partial \log o_i}{\partial z_k} = \frac{1}{\ln 2}\sum_{i=1}^{K} -y_i\left(1\{i=k\} - o_k\right)$$

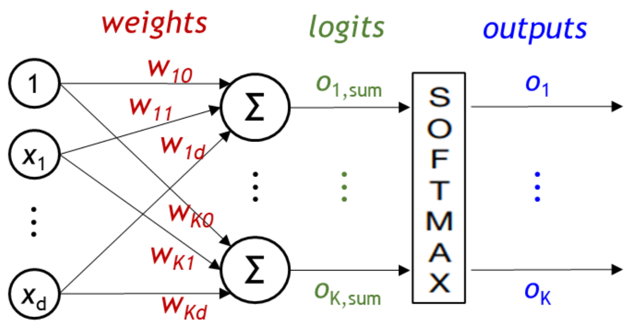
$$= -\frac{1}{\ln 2}\left(y_k - \left(\textstyle\sum_{i=1}^{K} y_i\right)o_k\right) = \frac{1}{\ln 2}\left(o_k - y_k\right)$$

3. Gradient of MSE & CE Losses

Gradient of MSE & CE Losses (Summary)

- For a <u>*linear*</u> neuron (i.e., neuron without activation = Linear Regressor):
  - Gradient of **MSE**: (output – label) ·(input$_i$)
- For a <u>*non-linear*</u> neuron (e.g., Logistic Regression or Perceptron)
  - Gradient of **MSE**: (output – label) · $\sigma'()$ ·(input$_i$)
  - Gradient of **CE**: (output – label) · (input$_i$)
- For a neural network (of <u>*non-linear*</u> neurons)
  - Gradient of **MSE**
    - ★ For k^{th} <u>output</u> neuron: (output$_k$ – label$_k$) · $\sigma_k'()$ ·(input$_j$) = (delta$_k$)·(input$_j$)
    - ★ For j^{th} <u>intermediate</u> neuron: $\left(\sum_{\forall k \in \{\text{next layer neurons}\}} (\text{delta}_k)\cdot(\text{weight}_{kj})\right) \cdot \sigma_j'()$ (input$_i$) = (eta$_j$)·(input$_i$)
  - Gradient of **CE** (with Softmax output)
    - ★ For k^{th} <u>Softmax output</u>: (output$_k$ – label$_k$) ·(input$_j$)





Q & A