TON DUC THANG UNIVERSITY
Faculty of Information Technology
Computer Science

Multi-class classification with Softmax function

Lê Anh Cường

Outline

- Repeat logistic function
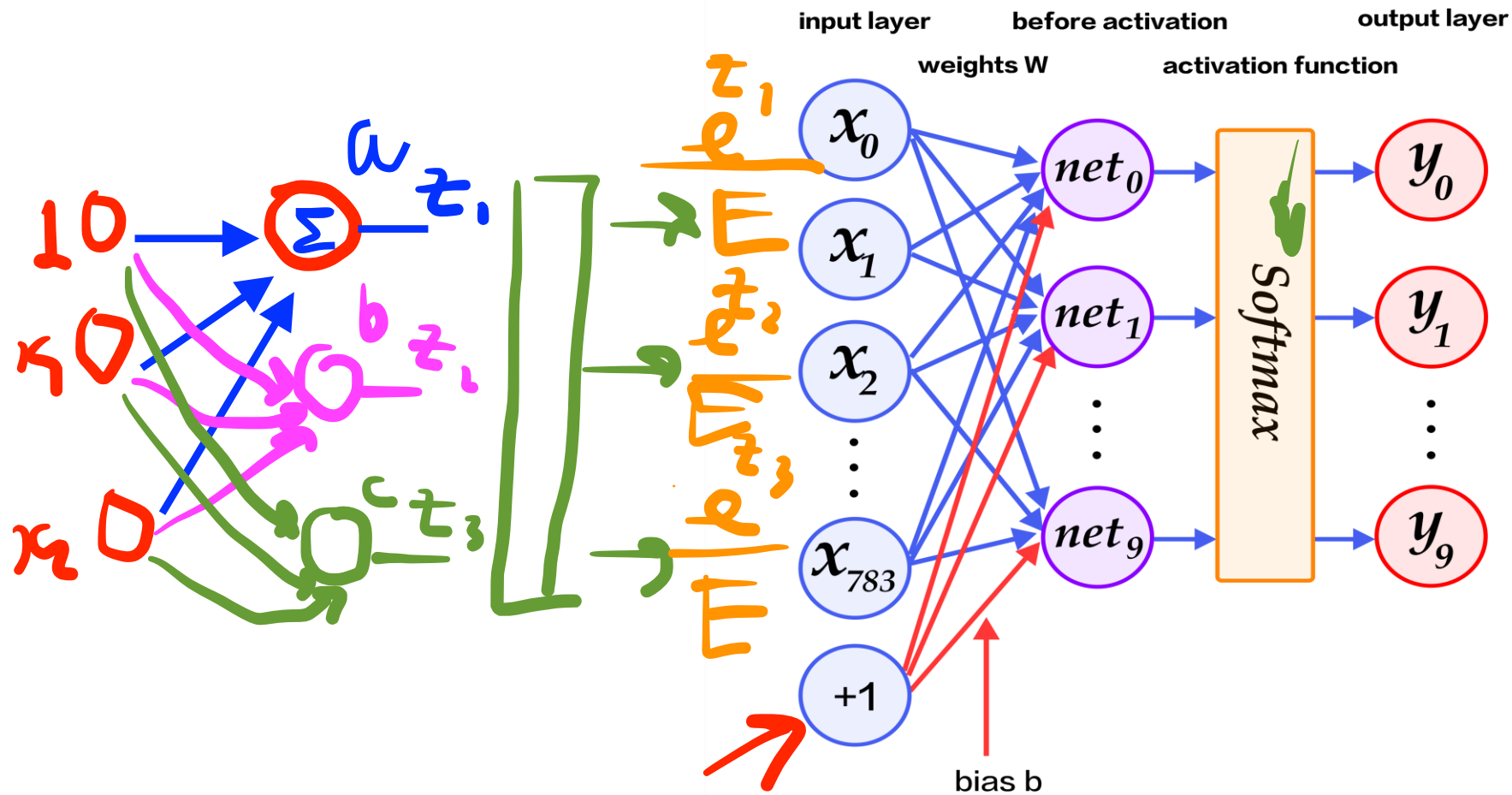- Softmax function
- Derivative of the softmax function

IT701010 - Chapter 8.3 - Multi-class classification with softmax function

Logistic function

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

$$h_\theta(x) = P(y = 1 | x; \theta) = \frac{1}{1 + e^{-\theta^T x}}$$

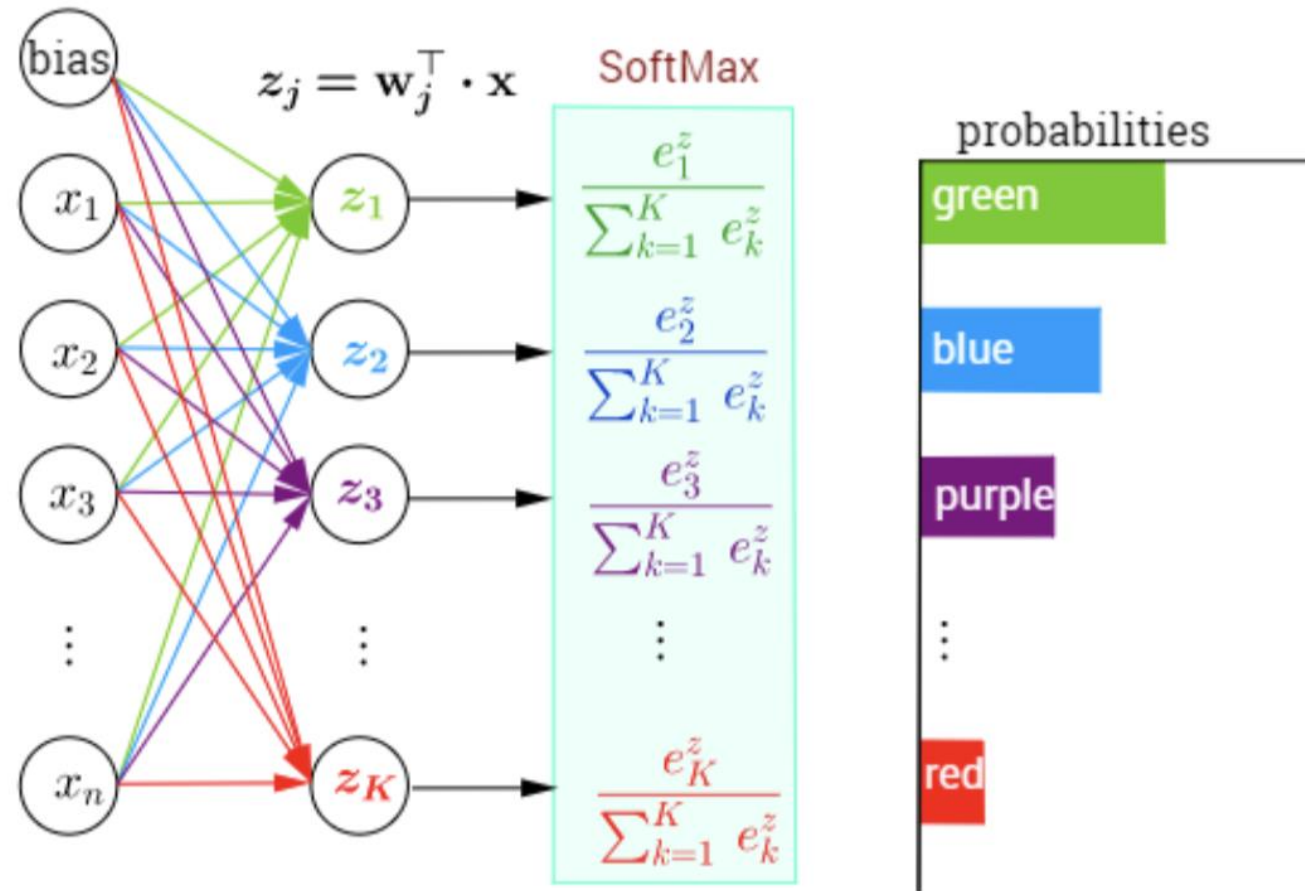$$P(y = 0 | x; \theta) = 1 - P(y = 1 | x; \theta) = \frac{e^{-\theta^T x}}{1 + e^{-\theta^T x}}$$

IT701010 - Chapter 8.3 - Multi-class classification with softmax function

NN with Softmax layer

IT701010 - Chapter 8.3 - Multi-class classification with softmax function

NN with Softmax layer

IT701010 - Chapter 8.3 - Multi-class classification with softmax function

Softmax function

The denominator $\sum_{d=1}^{C} e^{z_d}$ acts as a regularizer to make sure that $\sum_{c=1}^{C} y_c = 1$. As the output layer of a neural network, the softmax function can be represented graphically as a layer with C neurons.

We can write the probabilities that the class is $t = c$ for $c = 1 \ldots C$ given input $\mathbf{z}$ as:

$$\begin{bmatrix} P(t=1|\mathbf{z}) \\ \vdots \\ P(t=C|\mathbf{z}) \end{bmatrix} = \begin{bmatrix} \varsigma(\mathbf{z})_1 \\ \vdots \\ \varsigma(\mathbf{z})_C \end{bmatrix} = \frac{1}{\sum_{d=1}^{C} e^{z_d}} \begin{bmatrix} e^{z_1} \\ \vdots \\ e^{z_C} \end{bmatrix}$$

Where $P(t=c|\mathbf{z})$ is thus the probability that that the class is c given the input $\mathbf{z}$.

$$y_c = \varsigma(\mathbf{z})_c = \frac{e^{z_c}}{\sum_{d=1}^{C} e^{z_d}} \quad \text{for } c = 1 \cdots C$$

IT701010 - Chapter 8.3 - Multi-class classification with softmax function

Likelihood

The likelihood $\mathcal{L}(\theta|\mathbf{t}, \mathbf{z})$ can be rewritten as the joint probability of generating $\mathbf{t}$ and $\mathbf{z}$ given the parameters θ: $P(\mathbf{t}, \mathbf{z}|\theta)$. Which can be written as a conditional distribution:

$$P(\mathbf{t}, \mathbf{z}|\theta) = P(\mathbf{t}|\mathbf{z}, \theta)P(\mathbf{z}|\theta)$$

Since we are not interested in the probability of $\mathbf{z}$ we can reduce this to: $\mathcal{L}(\theta|\mathbf{t}, \mathbf{z}) = P(\mathbf{t}|\mathbf{z}, \theta)$. Which can be written as $P(\mathbf{t}|\mathbf{z})$ for fixed θ. Since each t_c is dependent on the full $\mathbf{z}$, and only 1 class can be activated in the $\mathbf{t}$ we can write

$$P(\mathbf{t}|\mathbf{z}) = \prod_{i=c}^{C} P(t_c|\mathbf{z})^{t_c} = \prod_{c=1}^{C} \varsigma(\mathbf{z})_c^{t_c} = \prod_{c=1}^{C} y_c^{t_c}$$

IT701010 - Chapter 8.3 - Multi-class classification with softmax function

Likelihood

$$P(\mathbf{t}|\mathbf{z}) = \prod_{i=c}^{C} P(t_c|\mathbf{z})^{t_c} = \prod_{c=1}^{C} \varsigma(\mathbf{z})_c^{t_c} = \prod_{c=1}^{C} y_c^{t_c}$$

As was noted during the derivation of the loss function of the logistic function, maximizing this likelihood can also be done by minimizing the negative log-likelihood:

$$-\log \mathcal{L}(\theta|\mathbf{t}, \mathbf{z}) = \xi(\mathbf{t}, \mathbf{z}) = -\log \prod_{c=1}^{C} y_c^{t_c} = \sum_{c=1}^{C} t_c \cdot \log(y_c)$$

IT701010 - Chapter 8.3 - Multi-class classification with softmax function

Cross-entropy loss function

$$H(p, q) = -\sum_{\forall x} p(x) \log(q(x))$$

$$H(p, q) = -\sum_i p_i \log q_i$$

The cross-entropy error function over a batch of multiple samples of size n can be calculated as:

$$\xi(T, Y) = \sum_{i=1}^{n} \xi(\mathbf{t}_i, \mathbf{y}_i) = -\sum_{i=1}^{n} \sum_{c=1}^{C} t_{ic} \cdot \log(y_{ic})$$

Where t_{ic} is 1 if and only if sample i belongs to class c, and y_{ic} is the output probability that sample i belongs to class c.

Derivative of the softmax function

To use the softmax function in neural networks, we need to compute its derivative. If we define $\Sigma_C = \sum_{d=1}^{C} e^{z_d}$ for $c = 1 \cdots C$ so that $y_c = e^{z_c}/\Sigma_C$, then this derivative $\partial y_i / \partial z_j$ of the output $\mathbf{y}$ of the softmax function with respect to its input $\mathbf{z}$ can be calculated as:

$$\text{if } i = j : \frac{\partial y_i}{\partial z_i} = \frac{\partial \frac{e^{z_i}}{\Sigma_C}}{\partial z_i} = \frac{e^{z_i}\Sigma_C - e^{z_i}e^{z_i}}{\Sigma_C^2} = \frac{e^{z_i}}{\Sigma_C}\frac{\Sigma_C - e^{z_i}}{\Sigma_C} = \frac{e^{z_i}}{\Sigma_C}(1 - \frac{e^{z_i}}{\Sigma_C}) = y_i(1 - y_i)$$

$$\text{if } i \neq j : \frac{\partial y_i}{\partial z_j} = \frac{\partial \frac{e^{z_i}}{\Sigma_C}}{\partial z_j} = \frac{0 - e^{z_i}e^{z_j}}{\Sigma_C^2} = -\frac{e^{z_i}}{\Sigma_C}\frac{e^{z_j}}{\Sigma_C} = -y_i y_j$$

Note that if $i = j$ this derivative is similar to the derivative of the logistic function.

Derivative of the cross-entropy loss function for the softmax function

$$\xi(T, Y) = \sum_{i=1}^{n} \xi(\mathbf{t}_i, \mathbf{y}_i) = -\sum_{i=1}^{n} \sum_{c=1}^{C} t_{ic} \cdot \log(y_{ic})$$

$$\frac{\partial \xi}{\partial \theta_j} = \frac{\partial \xi}{\partial y_c} \cdot \frac{\partial y_c}{\partial z_c} \cdot \frac{\partial z_c}{\partial \theta_j}$$

$$y_c = \varsigma(\mathbf{z})_c = \frac{e^{z_c}}{\sum_{d=1}^{C} e^{z_d}} \quad \text{for } c = 1 \cdots C$$

IT701010 - Chapter 8.3 - Multi-class classification with softmax function

Derivative of the cross-entropy loss function for the softmax function

$$\xi(T, Y) = \sum_{i=1}^{n} \xi(\mathbf{t}_i, \mathbf{y}_i) = -\sum_{i=1}^{n} \sum_{c=1}^{C} t_{ic} \cdot \log(y_{ic})$$

$$\frac{\partial \log(y_i)}{\partial y_c} = \frac{1}{y_c}$$

$$(\ln x)' = \frac{1}{x}$$

$$\frac{\partial \xi}{\partial \theta_j} = \frac{\partial \xi}{\partial y_c} \cdot \frac{\partial y_c}{\partial z_c} \cdot \frac{\partial z_c}{\partial \theta_j}$$

$$y_c = \varsigma(\mathbf{z})_c = \frac{e^{z_c}}{\sum_{d=1}^{C} e^{z_d}} \quad \text{for } c = 1 \cdots C$$

$$\text{if } i = j : \frac{\partial y_i}{\partial z_i} = \frac{\partial \frac{e^{z_i}}{\Sigma_C}}{\partial z_i} = \frac{e^{z_i} \Sigma_C - e^{z_i} e^{z_i}}{\Sigma_C^2} = \frac{e^{z_i}}{\Sigma_C} \frac{\Sigma_C - e^{z_i}}{\Sigma_C} = \frac{e^{z_i}}{\Sigma_C} \left(1 - \frac{e^{z_i}}{\Sigma_C}\right) = y_i(1 - y_i)$$

$$\text{if } i \neq j : \frac{\partial y_i}{\partial z_j} = \frac{\partial \frac{e^{z_i}}{\Sigma_C}}{\partial z_j} = \frac{0 - e^{z_i} e^{z_j}}{\Sigma_C^2} = -\frac{e^{z_i}}{\Sigma_C} \frac{e^{z_j}}{\Sigma_C} = -y_i y_j$$

$$\left(\frac{u}{v}\right)' = \frac{u'v - uv'}{v^2}$$

Derivative of the cross-entropy loss function for the softmax function

$$\frac{\partial \xi}{\partial z_i} = \ ?$$

Derivative of the cross-entropy loss function for the softmax function

The derivative $\partial\xi/\partial z_i$ of the loss function with respect to the softmax input z_i can be calculated as:

$$\frac{\partial\xi}{\partial z_i} = -\sum_{j=1}^{C} \frac{\partial t_j \log(y_j)}{\partial z_i} = -\sum_{j=1}^{C} t_j \frac{\partial \log(y_j)}{\partial z_i} = -\sum_{j=1}^{C} t_j \frac{1}{y_j} \frac{\partial y_j}{\partial z_i}$$

$$= -\frac{t_i}{y_i} \frac{\partial y_i}{\partial z_i} - \sum_{j\neq i}^{C} \frac{t_j}{y_j} \frac{\partial y_j}{\partial z_i} = -\frac{t_i}{y_i} y_i(1-y_i) - \sum_{j\neq i}^{C} \frac{t_j}{y_j}(-y_j y_i)$$

$$= -t_i + t_i y_i + \sum_{j\neq i}^{C} t_j y_i = -t_i + \sum_{j=1}^{C} t_j y_i = -t_i + y_i \sum_{j=1}^{C} t_j$$

$$= y_i - t_i$$

Gradient Descent?

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)\right]$$

Want $\min_\theta J(\theta)$:

Repeat $\{$

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

(simultaneously update all θ_j)

$\}$

Example of Softmax and classification

- https://medium.com/@awjuliani/simple-softmax-in-python-tutorial-d6b4c4ed5c16

- https://www.kaggle.com/saksham219/softmax-regression-for-iris-classification

- https://scikit-learn.org/stable/auto_examples/linear_model/plot_iris_logistic.html