Cross-Entropy and Softmax Function

Người thực hiện: Trần Ngọc Bảo Duy

Người hướng dẫn: PGS TS Lê Anh Cường

1. Cross entropy

Entropy là độ đo bất xác định khi dự đoán trạng thái của một biến ngẫu nhiên X. Entropy thông tin của biến ngẫu nhiên X càng cao thì càng khó dự đoán.

$$\mathrm{H}(p) = -\sum_{i=1}^{n} p_i \log_b p_i$$

Entropy:

Cross Entropy là độ đo giữa hai phân bố p (phân bố đúng - true distribution) và q (phân bố hiện tại) để đo lượng trung bình thông tin khi dùng mã hóa thông tin của phân bố q thay cho mã hóa thông tin phân bố p

$$H(p, q) = -\sum_{i=1}^{n} p_i \log_b q_i$$

Cross entropy

Kullback–Leibler divergence:
là một độ đo đi đo mức độ lệch của một phân bố đối với phân bố được chỉ định

$$D_{\mathrm{KL}}(p||q) = H(p, q) - H(p)$$

$$D_{\mathrm{KL}}(p||q) = -\sum_{i=1}^{n} p_i \log_b \frac{q_i}{p_i} = \sum_{i=1}^{n} p_i \log_b \frac{p_i}{q_i}$$

KL divergence:

2. Softmax function - định nghĩa

Softmax regression là một mô hình ứng với đầu vào x với đầu ra xác suất a_i để input đó rơi vào class i

- Điều đặc biệt là a_i đầu ra phải dương và tổng của chúng bằng 1
- Với công thức - softmax function bên dưới ta sẽ đảm bảo được đầu ra tổng của các a_i = 1, phân bố của từng z_i trên tổng số class C, với $z_i = \mathbf{w}_i^T \mathbf{x}$

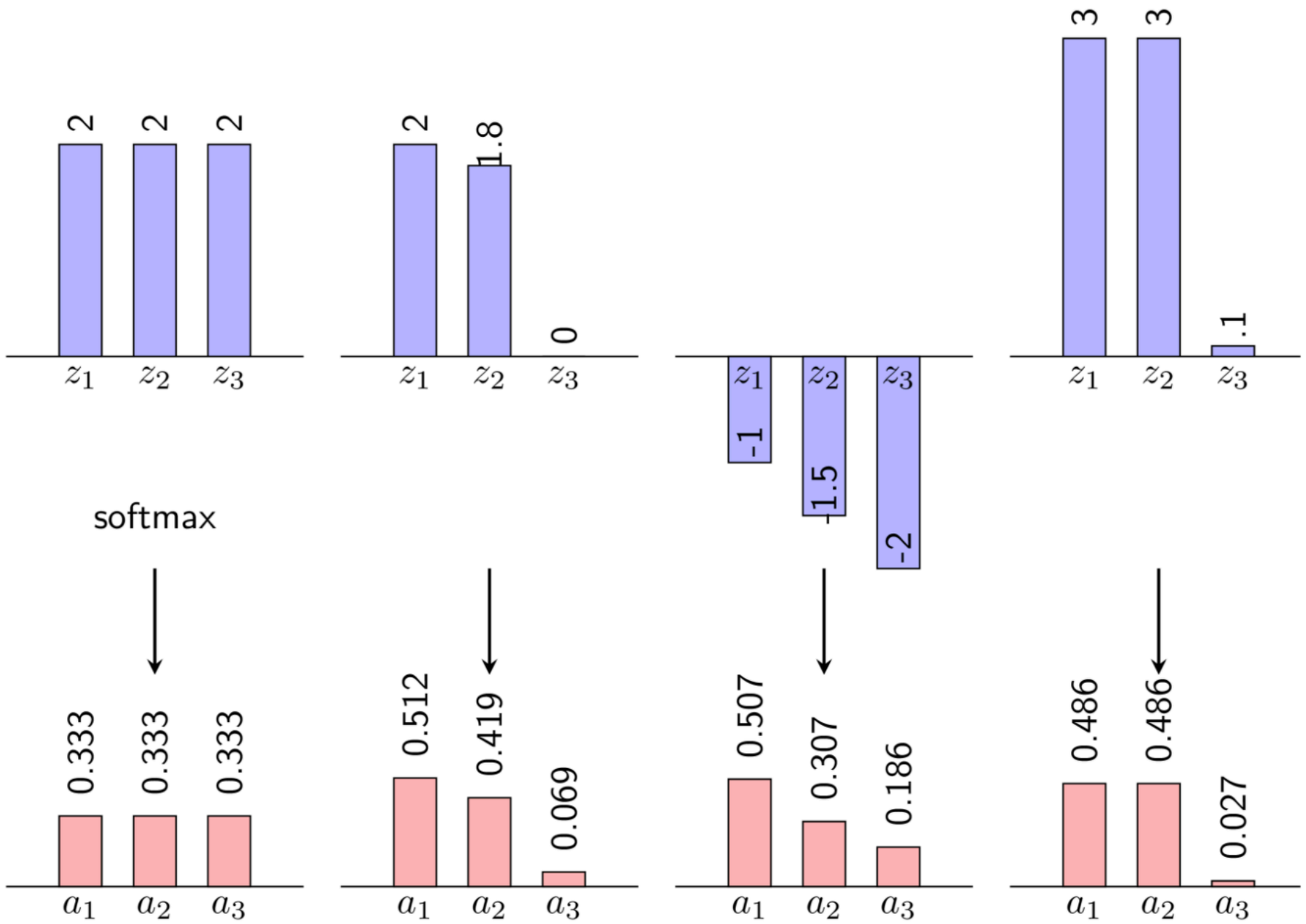
$$a_i = \frac{\exp(z_i)}{\sum_{j=1}^{C} \exp(z_j)}, \quad \forall i = 1,2,\ldots,C$$

- Tại sao exp() mà không phải là vì có thể nhận giá trị âm hoặc dương nên ta dùng exp() sẽ chuyển nó về dương => hàm đồng biến, dễ tính toán
  - **Chú ý rằng với cách định nghĩa này, không có xác suất a_i nào tuyệt đối bằng 0 hoặc tuyệt đối bằng 1, mặc dù chúng có thể rất gần 0 hoặc 1 khi z_i rất nhỏ hoặc rất lớn khi so sánh với các $z_j, j \neq i$**

$$P(y_k = i \,|\, \mathbf{x}_k; \mathbf{W}) = a_i$$

- Trong đó, $P(y = i \,|\, \mathbf{x}; \mathbf{W})$ được hiểu là xác suất để một điểm dữ liệu x rơi vào class thứ i nếu biết tham số mô hình (ma trận trọng số) là $\mathbf{W}$

2. Softmax function



Hình 3: Một số ví dụ về đầu vào và đầu ra của hàm softmax.

2. Softmax function - softmax tối ưu

- Vì đầu ra zi có thể là một số cực lớn nên ta có thể dùng một hằng số c để giảm giá trị nó xuống, làm cho hàm ra ổn định hơn
- Trong thực nghiệm, giá trị đủ lớn này thường được chọn là $c = \max_i z_i$

$$a_i = \frac{\exp(z_i)}{\sum_{j=1}^{C} \exp(z_j)} = \frac{\exp(-c)\exp(z_i)}{\exp(-c)\sum_{j=1}^{C} \exp(z_j)}$$

$$= \frac{\exp(z_i - c)}{\sum_{j=1}^{C} \exp(z_j - c)}$$

2. Softmax function - Hàm loss (TH đặc biệt)

- Hàm softmax với số class là 2 thì đầu ra nó tương tự như hàm sigmoid trong logistic regresstion

$$
\begin{aligned}
a_1 &= \frac{\exp(\mathbf{w}_1^T \mathbf{x})}{\exp(\mathbf{w}_1^T \mathbf{x}) + \exp(\mathbf{w}_2^T \mathbf{x})} \\
&= \frac{1}{1 + \exp((\mathbf{w}_2 - \mathbf{w}_1)^T \mathbf{x})}
\end{aligned}
$$

- Đầu ra dự đoán của điểm dữ liệu đó là $a_i = \text{sigmoid}(\mathbf{w}^T \mathbf{x})$ là xác suất để điểm đó rơi vào class thứ nhất. Xác suất để điểm đó rơi vào class thứ hai có thể được dễ dàng suy ra là $1 - a_i$. Vì vậy, hàm mất mát trong Logistic Regression chính là một trường hợp đặc biệt của Cross Entropy. (N được dùng để thể hiện số điểm dữ liệu trong tập training).

$$
J(\mathbf{w}) = -\sum_{i=1}^{N} (y_i \log a_i + (1 - y_i) \log(1 - a_i))
$$

2. Softmax function - Hàm loss

- Với Softmax Regression, trong trường hợp có C classes, loss giữa đầu ra dự đoán và đầu ra thực sự của một điểm dữ liệu x_i được tính bằng: (Cross entropy)

$$J(\mathbf{W}; \mathbf{x}_i, \mathbf{y}_i) = -\sum_{j=1}^{C} y_{ji} \log(a_{ji})$$

- Với y_{ji}và a_{ji}lần lượt là là phần tử thứ j của vector (xác suất) y_i và a_i . Và đầu ra a_i phụ thuộc vào đầu vào x_i và ma trận trọng số $\mathbf{W}$.

2. Softmax function - Hàm loss

- Ta có tất cả các cặp dữ liệu $\mathbf{x}_i, \mathbf{y}_i, i = 1,2,\ldots,N$, chúng ta sẽ có hàm mất mát cho Softmax Regression như sau:

$$J(\mathbf{W}; \mathbf{X}, \mathbf{Y}) = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ji}\log(a_{ji})$$

$$= -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ji}\log\left(\frac{\exp(\mathbf{w}_j^T\mathbf{x}_i)}{\sum_{k=1}^{C}\exp(\mathbf{w}_k^T\mathbf{x}_i)}\right)$$

- Sử dụng Gradient Descent (GD) để cập nhật trọng số w
- Trong đó $x_i(\mathbf{y}_i - \mathbf{a}_i)^T$ là đạo hàm của Loss theo w

$$\mathbf{W} = \mathbf{W} + \eta\mathbf{x}_i(\mathbf{y}_i - \mathbf{a}_i)^T$$

2. Softmax function - Hàm loss

- Vì không thể đạo hàm trực tiếp Loss trên w được, nên ta phải thông qua chain rule và tính từng đạo hàm tương ứng

$$\frac{\partial J}{\partial W} = \frac{\partial J}{\partial A} * \frac{\partial A}{\partial Z} * \frac{\partial Z}{\partial W}$$

- Tương ứng hàm đạo hàm loss trên hàm softmax $\frac{\partial J}{\partial A}$

- Tương ứng hàm đạo hàm y trên hàm z $\frac{\partial A}{\partial Z}$

- Tương ứng đạo hàm của phương trình tổng trên w $\frac{\partial Z}{\partial W}$

2. Softmax function - Hàm loss

- Phân tích từng đạo hàm của hàm softmax

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

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

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

$$= y_i - t_i$$

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

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

2. Softmax function - Hàm loss

- Sau khi kết hợp ta sẽ có hàm loss

$$J_i(\mathbf{W}) \triangleq J(\mathbf{W}; \mathbf{x}_i, \mathbf{y}_i) =$$

$$= -\sum_{j=1}^{C} y_{ji} \log\left(\frac{\exp(\mathbf{w}_j^T \mathbf{x}_i)}{\sum_{k=1}^{C} \exp(\mathbf{w}_k^T \mathbf{x}_i)}\right)$$

$$= -\sum_{j=1}^{C} \left(y_{ji} \mathbf{w}_j^T \mathbf{x}_i - y_{ji} \log\left(\sum_{k=1}^{C} \exp(\mathbf{w}_k^T \mathbf{x}_i)\right)\right)$$

$$= -\sum_{j=1}^{C} y_{ji} \mathbf{w}_j^T \mathbf{x}_i + \log\left(\sum_{k=1}^{C} \exp(\mathbf{w}_k^T \mathbf{x}_i)\right) \quad (3)$$

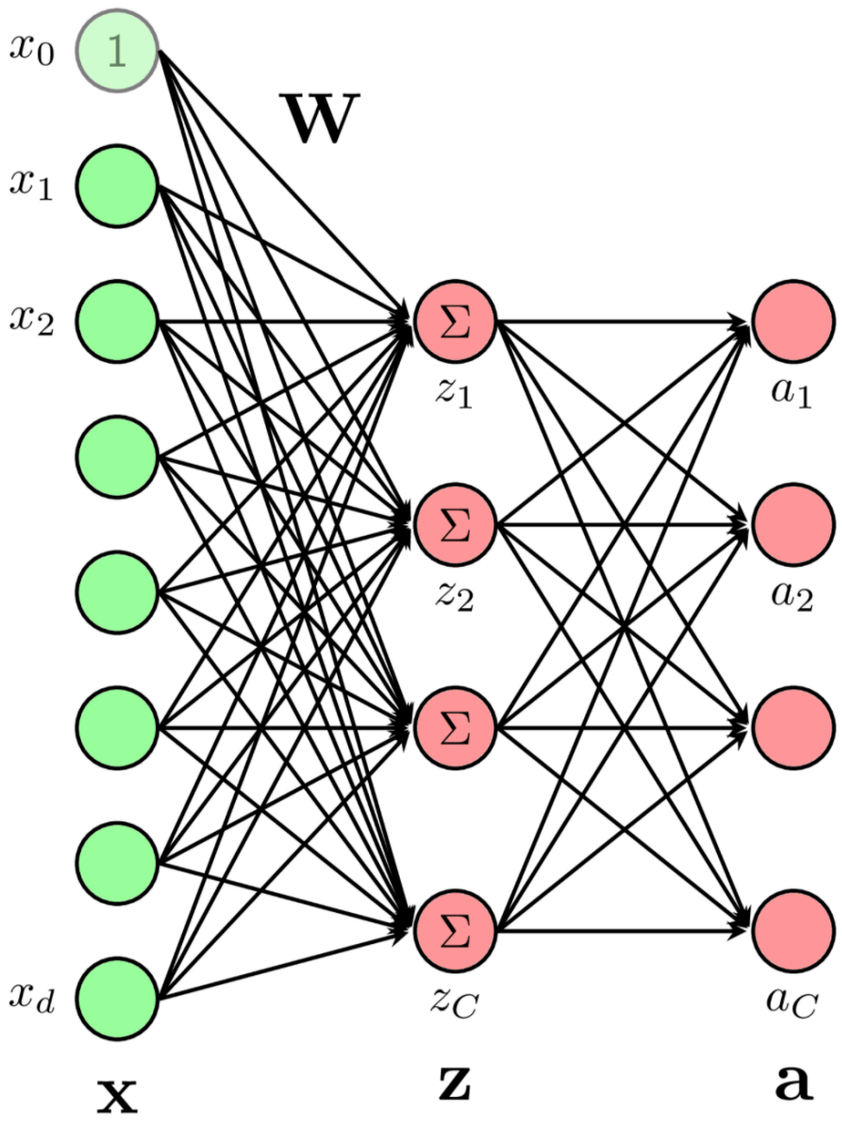
- Sau khi kết hợp ta sẽ có đạo hàm loss trên bộ dữ liệu W (gradient descend)

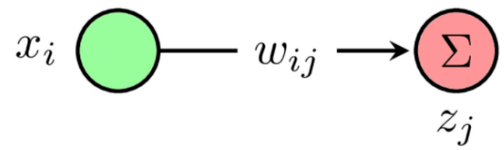
$$\frac{\partial J_i(\mathbf{W})}{\partial \mathbf{w}_j} = -y_{ji}\mathbf{x}_i + \frac{\exp(\mathbf{w}_j^T \mathbf{x}_i)}{\sum_{k=1}^{C} \exp(\mathbf{w}_k^T \mathbf{x}_i)} \mathbf{x}_i$$

$$= -y_{ji}\mathbf{x}_i + a_{ji}\mathbf{x}_i = \mathbf{x}_i(a_{ji} - y_{ji})$$

$$= e_{ji}\mathbf{x}_i \ (\textbf{where } e_{ji} = a_{ji} - y_{ji}) \quad (5)$$

2. Softmax function- trong Neural Network





w_{0j}: biases, don't forget!

d: data dimension

C: number of classes

$\mathbf{x} \in \mathbb{R}^{d+1}$

$\mathbf{W} \in \mathbb{R}^{(d+1)\times C}$

$z_i = \mathbf{w}_i^T \mathbf{x}$

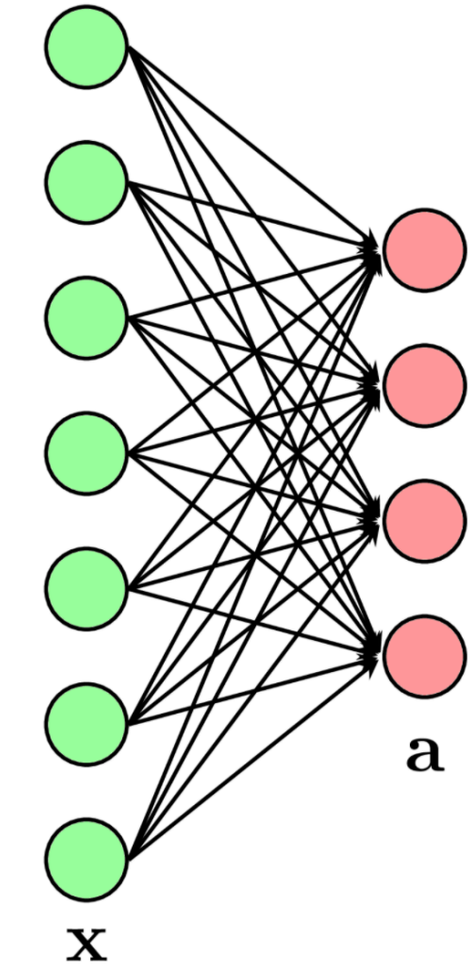
$\mathbf{z} = \mathbf{W}^T \mathbf{x} \in \mathbb{R}^C$

$\mathbf{a} = \text{softmax}(\mathbf{z}) \in \mathbb{R}^C$

$$a_i > 0, \quad \sum_{i=1}^{C} a_i = 1$$

$\mathbf{z} = \text{softmax}(\mathbf{W}^T \mathbf{x})$

short form



Hình 2: Mô hình Softmax Regression dưới dạng Neural network.

Reference

Sách Deep Learning cơ bản của thầy Nguyễn Thanh Tuấn

Khoá học Machine Learning của thầy Vũ Hữu Tiệp

https://machinelearningcoban.com/2017/02/24/mlp/

Và các tài liệu thầy đã reference cho em