# Extra

Wednesday 22 May 2024 8:43 AM

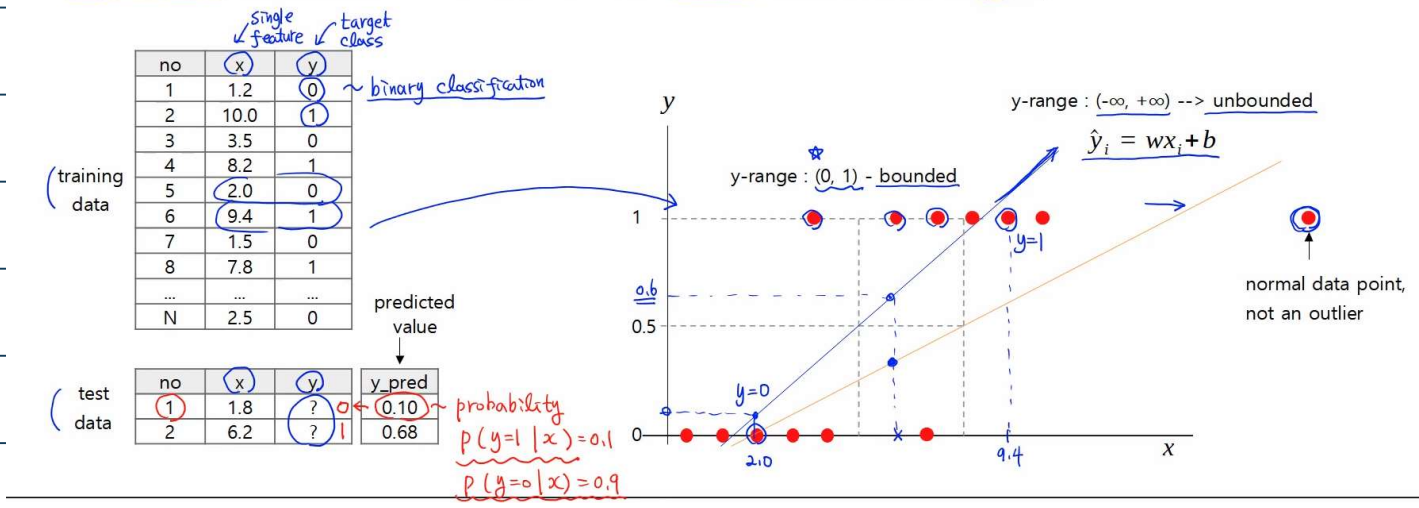
- Can we use a linear regression model to predict the y values for following data? The answer is no.

| no | x | y |
|---|---|---|
| 1 | 1.2 | 0 |
| 2 | 10.0 | 1 |
| 3 | 3.5 | 0 |
| 4 | 8.2 | 1 |
| 5 | 2.0 | 0 |
| 6 | 9.4 | 1 |
| 7 | 1.5 | 0 |
| 8 | 7.8 | 1 |
| ... | ... | ... |
| N | 2.5 | 0 |

(training data)

| no | x | y | y_pred |
|---|---|---|---|
| 1 | 1.8 | ? | 0.10 |
| 2 | 6.2 | ? | 0.68 |

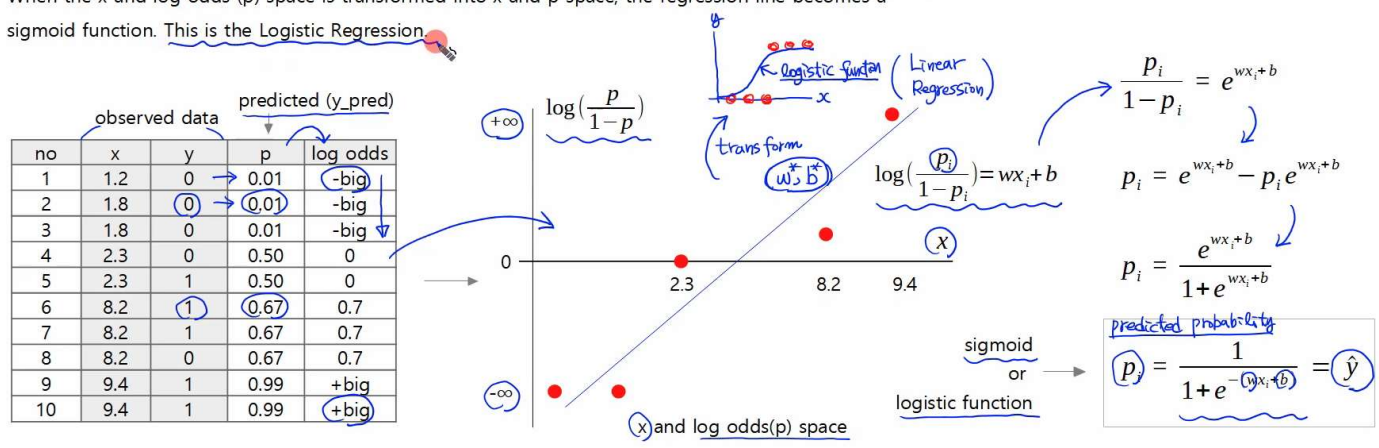
(test data)



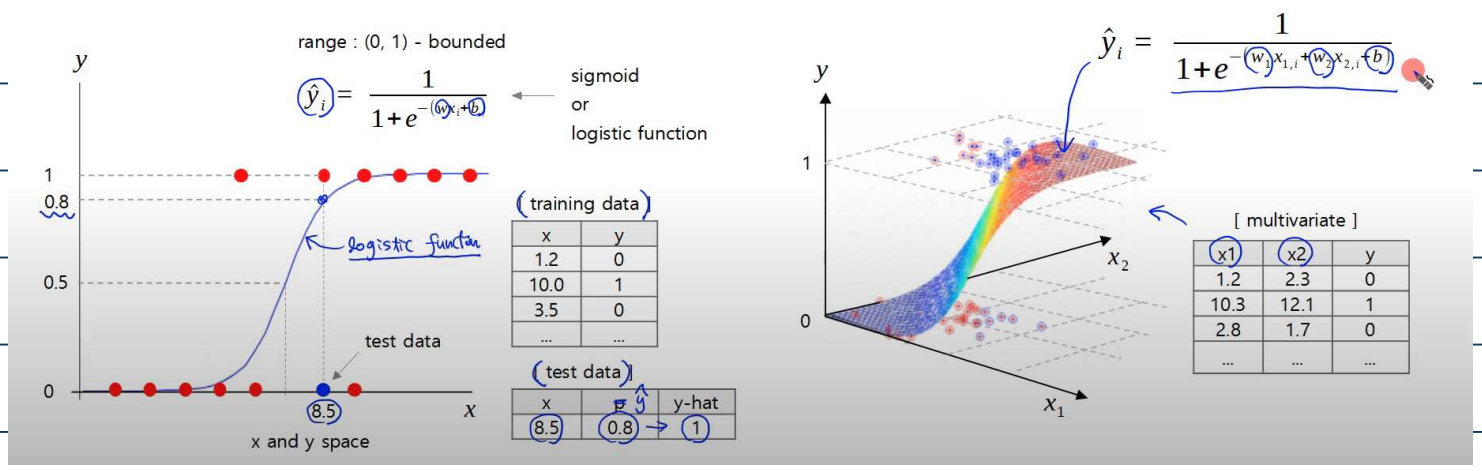

- The y value is bounded to be 0 or 1, and the probability p is bounded to be between 0 and 1.
- Linear regression can be applied to the values unbounded from -∞ to +∞.
- Therefore, it is not appropriate to apply linear regression to the data below.
- However, linear regression can be applied by converting the range of the predicted value to -∞ ~ +∞
- If the predicted value is expressed in log odds (logit), the range can be converted from -∞ to +∞.
- Linear regression can be applied in x and log odds (p) space instead of x and p space.
- When the x and log odds (p) space is transformed into x and p space, the regression line becomes a sigmoid function. This is the Logistic Regression.

| probability | odds | log odds (logit) |
|---|---|---|
| $p$ | $\frac{p}{1-p}$ | $\log(\frac{p}{1-p}) = logit(p)$ |
| (0 ~ 1) | (0 ~ +∞) | (-∞ ~ +∞) |

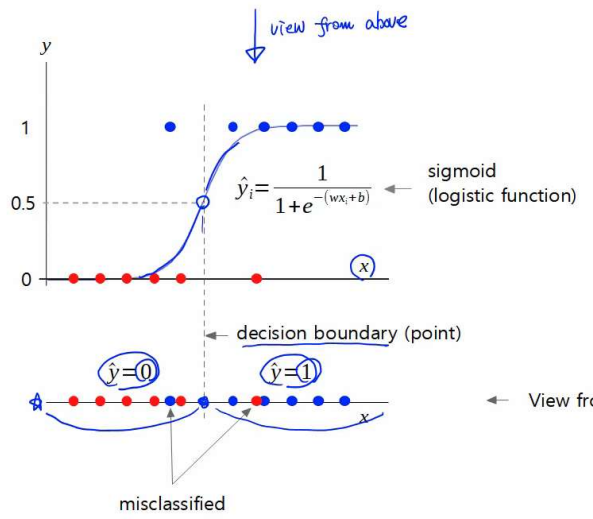
| no | x | y | p | log odds |
|---|---|---|---|---|
| 1 | 1.2 | 0 | 0.01 | -big |
| 2 | 1.8 | 0 | 0.01 | -big |
| 3 | 1.8 | 0 | 0.01 | -big |
| 4 | 2.3 | 0 | 0.50 | 0 |
| 5 | 2.3 | 1 | 0.50 | 0 |
| 6 | 8.2 | 1 | 0.67 | 0.7 |
| 7 | 8.2 | 1 | 0.67 | 0.7 |
| 8 | 8.2 | 0 | 0.67 | 0.7 |
| 9 | 9.4 | 1 | 0.99 | +big |
| 10 | 9.4 | 1 | 0.99 | +big |



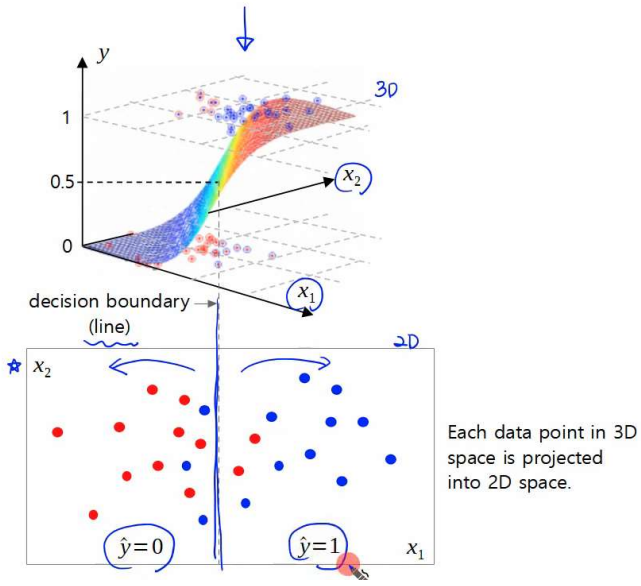

- Binary classification can be performed by fitting the logistic (or sgmoid) function to the training data in x and y space. w and b can be estimated.
- Use the estimated logistic function to predict the class y of the test data.



- If the sigmoid value is greater than 0.5, the y class is predicted to be 1, otherwise it is predicted to be 0.
- The point where the sigmoid value is 0.5 is called the decision boundary.




← View from above →



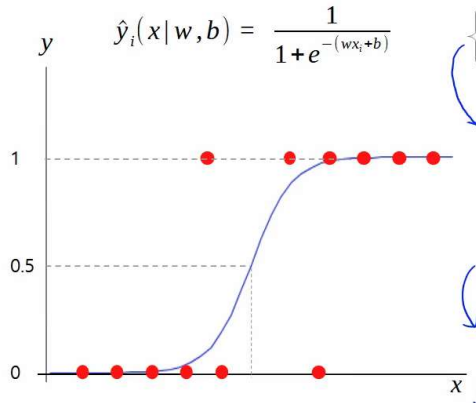

Each data point in 3D space is projected into 2D space.

- In linear regression, maximum likelihood estimation (MLE) was used to generate the objective function. In the same way, Logistic Regression can also use MLE to create an objective function that minimizes binary cross entropy.
- You can add a regularization term to the objective function in the same way as linear regression.

$$P(y|x) = \hat{y}^y (1-\hat{y})^{1-y}$$

| y | $\hat{y}$ | $P(y\|x)$ |
|---|---|---|
| True (1) | (0.8) | → $0.8^1 \times 0.2^0 = 0.8$ |
| False (1) | (0.2) | → $0.2^1 \times 0.8^0 = 0.2$ |
| True (0) | (0.2) | → $0.2^0 \times 0.8^1 = 0.8$ |

The larger this value, the better the prediction.




$$\hat{y}_i(x|w,b) = \frac{1}{1+e^{-(wx_i+b)}}$$

$$P(y=1|x;w,b) = \hat{y}(x;w,b) \quad \leftarrow y=1$$

$$P(y=0|x;w,b) = 1-\hat{y}(x;w,b) \quad \leftarrow y=0$$

(max.) $y=0, y=1$

$$P(y|x) = \hat{y}^{(y)}(1-\hat{y})^{(1-y)}$$ (notations w, b are omitted)

The large the P(y|x), the better the prediction performance.

$$L(w,b) = \prod_{i=1}^{N} P(y_i \mid x_i;w,b)$$

$$\log(L(w,b)) = \sum_{i=1}^{N} \log\left(\hat{y}_i^{\,y_i}(1-\hat{y}_i)^{1-y_i}\right)$$

$$\log(L(w,b)) = \sum_{i=1}^{N} [\, y_i \cdot \log \hat{y}_i + (1-y_i)\cdot\log(1-\hat{y}_i)] \quad (-BCE)$$

• Objective function (binary cross entropy)

$$\max_{w,b} \sum_i \log(L(w,b))$$

$$= \min_{w,b} \sum_i [-y_i \cdot \log \hat{y}_i - (1-y_i)\cdot\log(1-\hat{y}_i)]$$

$$= \min_{w,b} \sum_i BCE$$

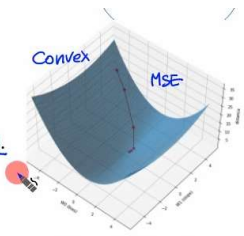
• Regularization

$$\min_{w,b} \sum_i BCE + \lambda \sum_{j=0}^{k} |w_j| \quad \text{(Lasso)}$$

$$\min_{w,b} \sum_i BCE + \lambda \sum_{j=0}^{k} w_j^2 \quad \text{(Ridge)}$$

## Convexity of the objective function

- Binary cross entropy, the objective function of logistic regression, is a convex function and has a unique solution.
- Since the second derivative of binary cross entropy is always greater than 0, we can see that it is a convex function.




Objective function for Linear Regression

$$J(w,b) = \sum_{i=1}^{N} [-y_i \cdot \log \hat{y}_i - (1-y_i)\cdot\log(1-\hat{y}_i)] \quad (BCE)$$

$$\hat{y}_i(x|w,b) = \frac{1}{1+e^{-(wx_i+b)}}$$

$$\hat{y}_i = \frac{1}{1+e^{-t}}, \quad (t = wx_i+b)$$

$$J_i = -y\cdot\log\hat{y}_t - (1-y)\cdot\log(1-\hat{y}_t)$$

→ binary cross entropy

$$\frac{d}{dt}\hat{y}_i = \frac{e^{-t}}{(1+e^{-t})^2} = \frac{1+e^{-t}}{(1+e^{-t})^2} - \frac{1}{(1+e^{-t})^2}$$

$$= \frac{1}{1+e^{-t}}\left(1-\frac{1}{1+e^{-t}}\right) = \hat{y}_t(1-\hat{y}_t)$$

$$\frac{d}{dt}(J_i) = -\frac{y}{\hat{y}_t}\hat{y}_t(1-\hat{y}_t) + (1-y)\frac{\hat{y}_t(1-\hat{y}_t)}{1-\hat{y}_t} = -y+\hat{y}_t$$

$$\frac{d^2}{dt^2}(J_i) = \hat{y}_t(1-\hat{y}_t) = \frac{1}{1+e^{-t}} - \frac{1}{(1+e^{-t})^2} = \frac{e^{-t}}{(1+e^{-t})^2} > 0$$

## ■ Multiclass classification – OvR (One-vs-Rest) = OvA (one-vs-All)

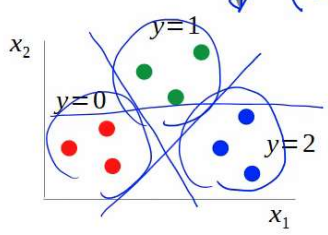
(Multinomial or Softmax Logistic Regression)

- Data with three classes (y = [0, 1, 2]) is one-hot encoded and then trained through three binary classification models (model-1, 2, 3).
- After training is complete, the test data is input into the three models to predict the probability that y is 0, 1, and 2, respectively, and the class with the highest probability is selected as the final class.

multiclass data → one-hot encoding

| no | x1 | x2 | y | p(y=0) | p(y=1) | p(y=2) |
|---|---|---|---|---|---|---|
| 1 | 0.8 | 1.2 | 0 | 1 | 0 | 0 |
| 2 | 9.8 | 2.0 | 2 | 0 | 0 | 1 |
| 3 | 3.2 | 3.5 | 0 | 1 | 0 | 0 |
| 4 | 4.8 | 5.2 | 1 | 0 | 1 | 0 |
| 5 | 2.7 | 2.0 | 0 | 1 | 0 | 0 |
| 6 | 8.3 | 3.4 | 2 | 0 | 0 | 1 |
| 7 | 5.7 | 4.5 | 1 | 0 | 1 | 0 |
| 8 | 8.3 | 7.8 | 1 | 0 | 1 | 0 |
| ... | ... | ... | ... | ... | ... | ... |
| N | 9.2 | 2.5 | 2 | 0 | 0 | 1 |

(0, 1)




Perform three binary classifications

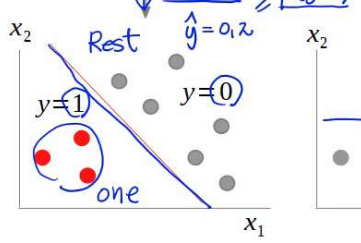
binary classification

| x1 | x2 | p(y=0) |
|---|---|---|
| 0.8 | 1.2 | 1 |
| 9.8 | 2.0 | 0 |
| 3.2 | 3.5 | 1 |
| 4.8 | 5.2 | 0 |
| 2.7 | 2.0 | 1 |
| 8.3 | 3.4 | 0 |
| 5.7 | 4.5 | 0 |
| 8.3 | 7.8 | 0 |
| ... | ... | ... |
| 9.2 | 2.5 | 0 |

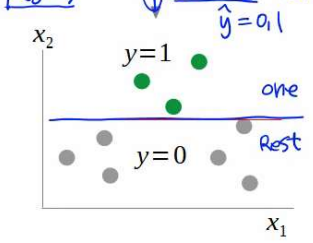
| x1 | x2 | p(y=1) |
|---|---|---|
| 0.8 | 1.2 | 0 |
| 9.8 | 2.0 | 0 |
| 3.2 | 3.5 | 0 |
| 4.8 | 5.2 | 1 |
| 2.7 | 2.0 | 0 |
| 8.3 | 3.4 | 0 |
| 5.7 | 4.5 | 1 |
| 8.3 | 7.8 | 1 |
| ... | ... | ... |
| 9.2 | 2.5 | 0 |

| x1 | x2 | p(y=2) |
|---|---|---|
| 0.8 | 1.2 | 0 |
| 9.8 | 2.0 | 1 |
| 3.2 | 3.5 | 0 |
| 4.8 | 5.2 | 0 |
| 2.7 | 2.0 | 0 |
| 8.3 | 3.4 | 1 |
| 5.7 | 4.5 | 0 |
| 8.3 | 7.8 | 0 |
| ... | ... | ... |
| 9.2 | 2.5 | 1 |

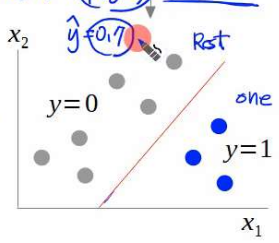
model-1 = P(y=0) ŷ=0.2




model-2 = P(y=1) ŷ=0.1




P(y=2) model-3 ŷ=0.7




- Let's assume that the data with three classes (y=0,1,2) is trained with two binary classification models (model-1, 2) as shown below.
- Set a baseline class and perform binary classification on the remaining classes. Any class can be a baseline.
- While OvR actually requires training multiple binary models this method estimates all parameters simultaneously with single model.

multiclass data → one-hot encoding

| no | x1 | x2 | y | p(y=0) | p(y=1) | p(y=2) |
|---|---|---|---|---|---|---|
| 1 | 0.8 | 1.2 | 0 | 1 | 0 | 0 |
| 2 | 9.8 | 2.0 | 2 | 0 | 0 | 1 |
| 3 | 3.2 | 3.5 | 0 | 1 | 0 | 0 |
| 4 | 4.8 | 5.2 | 1 | 0 | 1 | 0 |
| ... | ... | ... | ... | ... | ... | ... |
| N | 9.2 | 2.5 | 2 | 0 | 0 | 1 |

binary classification

| x1 | x2 | p(y=0) |
|---|---|---|
| 0.8 | 1.2 | 1 |
| 9.8 | 2.0 | 0 |
| 3.2 | 3.5 | 1 |
| 4.8 | 5.2 | 0 |
| ... | ... | ... |
| 9.2 | 2.5 | 0 |

| x1 | x2 | p(y=1) |
|---|---|---|
| 0.8 | 1.2 | 0 |
| 9.8 | 2.0 | 0 |
| 3.2 | 3.5 | 0 |
| 4.8 | 5.2 | 1 |
| ... | ... | ... |
| 9.2 | 2.5 | 0 |

| x1 | x2 | p(y=2) |
|---|---|---|
| 0.8 | 1.2 | 0 |
| 9.8 | 2.0 | 1 |
| 3.2 | 3.5 | 0 |
| 4.8 | 5.2 | 0 |
| ... | ... | ... |
| 9.2 | 2.5 | 1 |

baseline class

model-1

model-2

$$p(y=1)=\frac{\exp(w_1\cdot x+b_1)}{1+\exp(w_1\cdot x+b_1)+\exp(w_2\cdot x+b_2)}$$

$$p(y=2)=\frac{\exp(w_2\cdot x+b_2)}{1+\exp(w_1\cdot x+b_1)+\exp(w_2\cdot x+b_2)}$$

$$p(y=0)=1-p(y=1)-p(y=2)=\frac{1}{1+\exp(w_1\cdot x+b_1)+\exp(w_2\cdot x+b_2)}$$

simultaneous equations

$$\log\frac{P(y=0)}{P(y=0)}=0$$

$$\log\frac{p(y=1)}{p(y=0)}=w_1\cdot x+b_1$$

$$\log\frac{p(y=2)}{p(y=0)}=w_2\cdot x+b_2$$

$$\frac{p(y=1)}{1-p(y=1)-p(y=2)}=\exp(w_1\cdot x+b_1)$$

$$\frac{p(y=2)}{1-p(y=1)-p(y=2)}=\exp(w_2\cdot x+b_2)$$

$p(y=0)$

## ■ Multinomial Logistic Regression (or Softmax Regression)

Binary classification

$$\begin{cases} P(y=1\mid x)=\dfrac{1}{1+\exp(wx+b)} : \text{Logistic function} \\ P(y=0\mid x)=1-P(y=1\mid x) \end{cases}$$

- Softmax function

$$p(y=1)=\frac{\exp(w_1\cdot x+b_1)}{1+\exp(w_1\cdot x+b_1)+\exp(w_2\cdot x+b_2)} \Rightarrow \frac{\exp(w_1\cdot x+b_1)}{\exp(w_0\cdot x+b_0)+\exp(w_1\cdot x+b_1)+\exp(w_2\cdot x+b_2)} = \frac{\exp(w_1\cdot x+b_1)}{\sum_{k=0}^{C-1}\exp(w_k\cdot x+b_k)}$$

$w_0=0,\ b_0=0$

$$p(y=2)=\frac{\exp(w_2\cdot x+b_2)}{1+\exp(w_1\cdot x+b_1)+\exp(w_2\cdot x+b_2)}=\frac{\exp(w_2\cdot x+b_2)}{\sum_{k=0}^{C-1}\exp(w_k\cdot x+b_k)}$$

$$p(y=0)=\frac{1}{1+\exp(w_1\cdot x+b_1)+\exp(w_2\cdot x+b_2)}=\frac{\exp(w_0\cdot x+b_0)}{\sum_{k=0}^{C-1}\exp(w_k\cdot x+b_k)}$$

softmax

(C : the number of classes)

## ■ Loss function for Softmax Regression

Binary classification → Logistic function → MLE → binary cross entropy

## Loss function for Softmax Regression

Binary classification → Logistic function → MLE → binary cross entropy

- Just like we did for binary classification, we can use MLE to obtain the loss function for softmax regression. The loss function is cross entropy, which is a general expression for binary cross entropy.
- Regularization can also be applied to the cross entropy loss function.

| | y | $\hat{y}$ | probability $P(y\|x)=\prod_{k=0}^{C-1}\hat{y}_{i,k}^{\,1(y_i=k)}$ |
|---|---|---|---|
| | | k= 0 | 1 2 | |
| True | 1 | 0.1 | 0.7 | 0.2 | → $0.1^0 \times 0.7^1 \times 0.2^0 = 0.7$ |
| False | 1 | 0.8 | 0.1 | 0.1 | → $0.8^0 \times 0.1^1 \times 0.1^0 = 0.1$ |
| True | 0 | 0.8 | 0.1 | 0.1 | → $0.8^1 \times 0.1^0 \times 0.1^0 = 0.8$ |

The larger this value, the better the prediction.

(softmax)

$$\hat{y}_{i,k} = \frac{\exp(w_k \cdot x_i + b_k)}{\sum_{k=0}^{C-1}\exp(w_k \cdot x_i + b_k)}$$

i : data index, k : class index
C: the number of classes

$P(y|x)$

$$L_i(w,b) = \prod_{k=0}^{C-1}\hat{y}_{i,k}^{\,1(y_i=k)}$$ (max)

$1(y_i=k)$ : indicator function
- 1(true) = 1
- 1(false) = 0

one-hot encoding

| y | k=0 | 1 | 2 | |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | $y \neq k$ |
| 1 | 0 | 1 | 0 | |
| 2 | 0 | 0 | 1 | $y = k$ |
| | $y_{i,0}$ | $y_{i,1}$ | $y_{i,2}$ | |

$$\log(L_i(w,b)) = \sum_{k=0}^{C-1}\log\left(\hat{y}_{i,k}^{\,1(y_i=k)}\right)$$ (max)

$$= \sum_{k=0}^{C-1} 1(y_i=k)\cdot\log(\hat{y}_{i,k}) : \; 1(y_i=k) \rightarrow y_{i,k}$$

$$J(w,b) = \sum_{i=0}^{N-1}\left[\sum_{k=0}^{C-1} y_{i,k}\cdot\log(\hat{y}_{i,k})\right]$$ (max) (−CE)

N : the number of data points
A general expression for binary cross entropy.
C = 2 --> binary cross entropy.

- Objective function

$$\max_{w,b} J(w,b)$$

$$= \min_{w,b}\sum_{i=0}^{N-1}\sum_{k=0}^{C-1}\left[-y_{i,k}\cdot\log(\hat{y}_{i,k})\right]$$ (CE)

$$= \min_{w,b}\sum_i CE_i$$

↑ softmax

- Regularization

$$\min_{w,b}\sum_i CE_i + \lambda\sum_{k=0}^{C-1}|w_k|$$ (Lasso)

$$\min_{w,b}\sum_i CE_i + \lambda\sum_{k=0}^{C-1}w_k^2$$ (Ridge)

- comparison: binary classification

$$\hat{y}_i = \frac{1}{1+\exp(-(wx_i+b))}$$

$$\min_{w,b}\sum_i BCE_i$$

[MXML-4-01]

$$P(y|x) = \hat{y}^y(1-\hat{y})^{1-y}$$

$$= \prod_{k=0}^{1}\hat{y}_k^{\,1(y_i=k)}$$ (indicator function)

Let $\hat{y}_{k=0} = 1-\hat{y}$, $\hat{y}_{k=1} = \hat{y}$