AI VIETNAM
All-in-One Course

Softmax Regression

Quang-Vinh Dinh
Ph.D. in Computer Science

Year 2023

Outline

- Motivation
- Model Construction
- Loss Function
- Generalization (Further Reading)
- Another Approach (Further Reading)

Linear Regression

❖ Prediction

Area-based House Price Data

Feature	Label
area	price
6.7	9.1
4.6	5.9
3.5	4.6
5.5	6.7

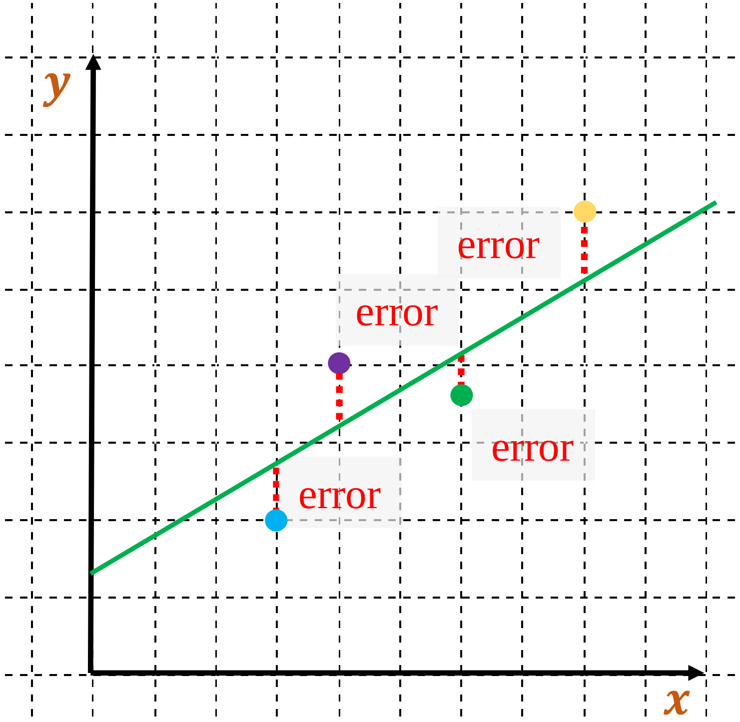
Training data

construct →

$$\hat{y} = \boldsymbol{\theta}^T \boldsymbol{x} = wx + b$$

$$\hat{y} \in (-\infty \quad +\infty)$$

Model



Find the line $\hat{y} = \boldsymbol{\theta}^T \boldsymbol{x}$ that is best fitting to given data, then use $\hat{y}$ to predict for new data

Logistic Regression

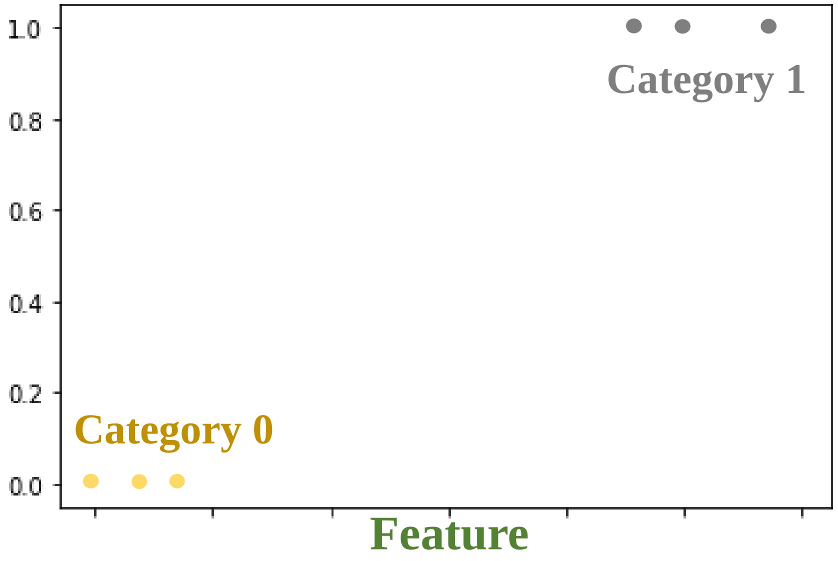
❖ Binary Classification

Feature	Label	
Petal_Length	Category	
1.4	Flower A	Category 0
1	Flower A	
1.5	Flower A	
3	Flower B	Category 1
3.8	Flower B	
4.1	Flower B	

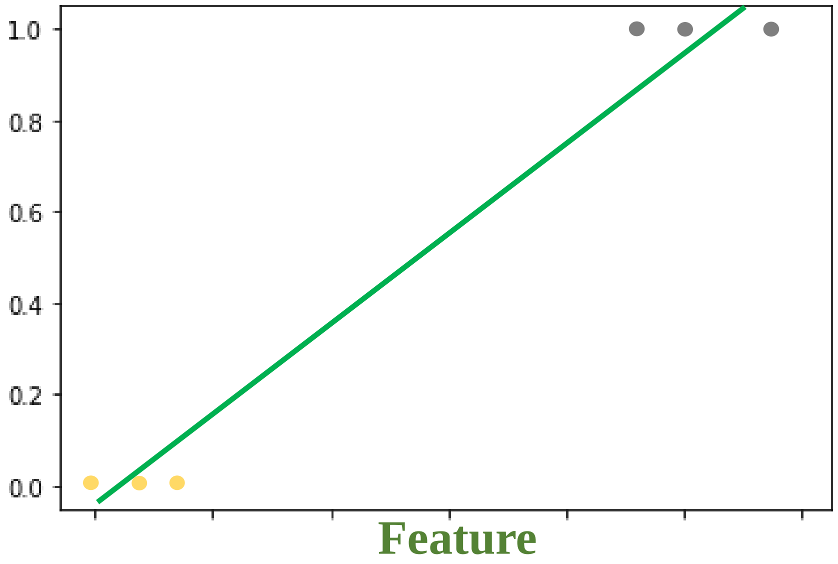
Assign numbers to categories

Feature	Label	
Petal_Length	Category	
1.4	0	Category 0
1	0	
1.5	0	
3	1	Category 1
3.8	1	
4.1	1	

Plot data



A line is not suitable for this data



Idea of Logistic Regression

❖ Binary Classification

Feature	Label
Petal_Length	Category
1.4	Flower A
1	Flower A
1.5	Flower A
3	Flower B
3.8	Flower B
4.1	Flower B

Category 0

Category 1

Assign numbers to categories

Feature	Label
Petal_Length	Category
1.4	0
1	0
1.5	0
3	1
3.8	1
4.1	1

Category 0

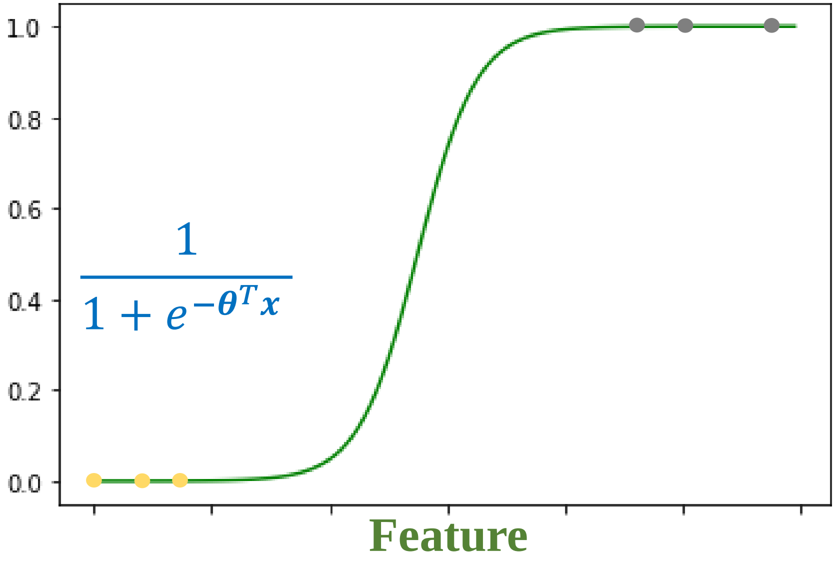
Category 1

Sigmoid function could fit the data

$$z = \boldsymbol{\theta}^T \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{\theta}$$

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

$$\hat{y} \in (0 \quad 1)$$

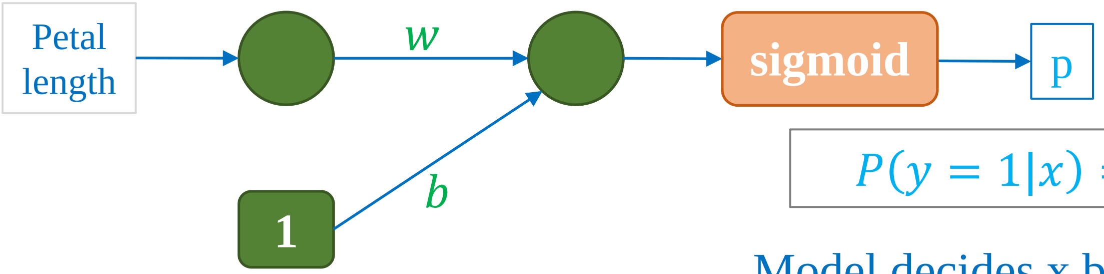


Binary cross-entropy

$$\mathrm{L} = -y\log\hat{y} - (1 - y)\log(1 - \hat{y})$$

Motivation

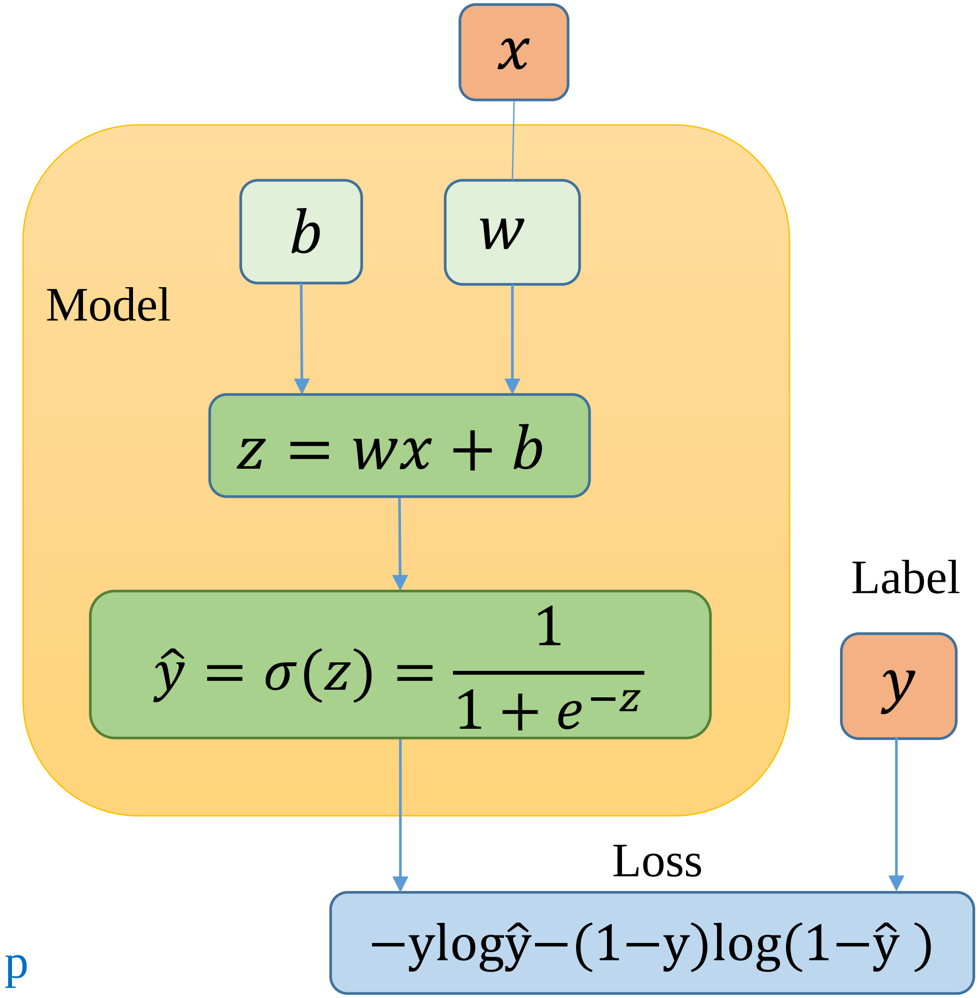
Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1



$P(y = 1|x) = p$

Model decides x belongs to the category (label=1) with the belief of p

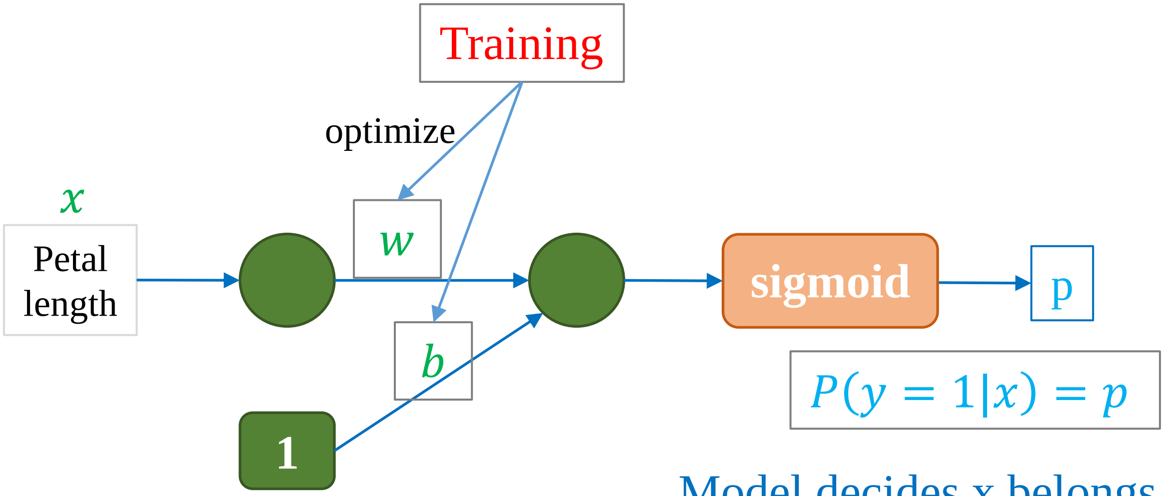
Implicitly conclude that $P(y = 0|x) = 1 - p$



Motivation

Problem!

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1



Model decides x belongs to category (y=1) with the belief of p

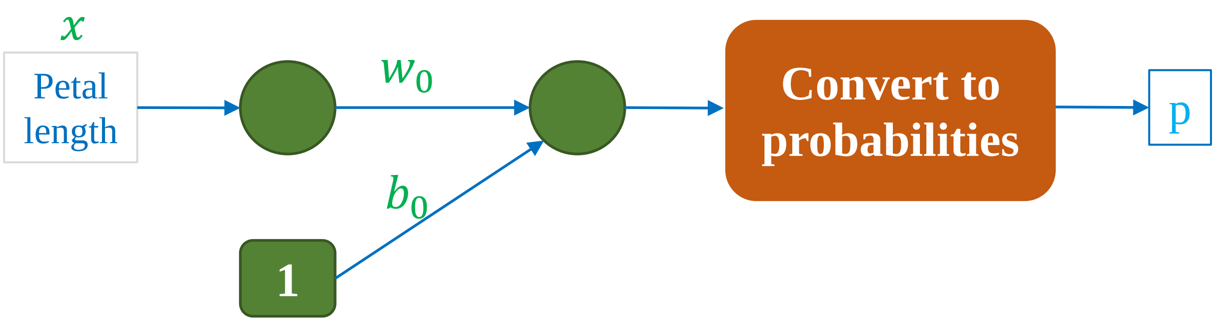
Implicitly extract that $P(label = 0|x) = 1 - p$

Optimize w and b for $P(label = 1|x)$ affects $P(label = 0|x)$ and vice versa

How to have explicitly $P(y = 0|x)$?

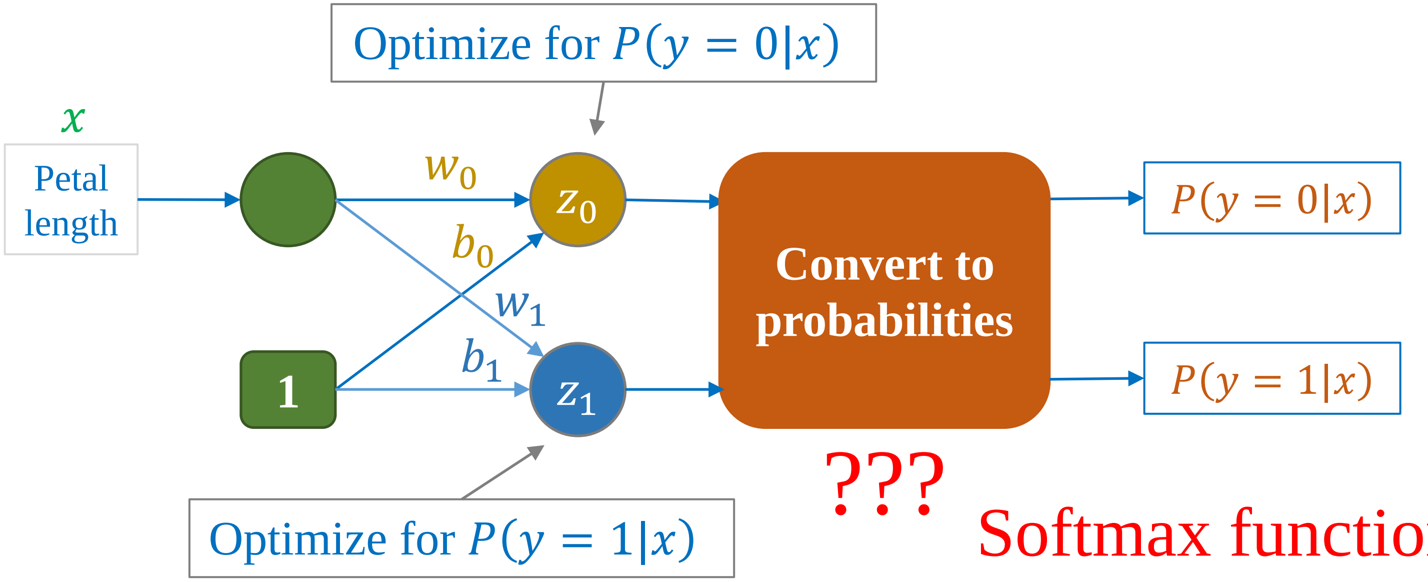
Motivation

Problem!



Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

Explicitly output $P(y=0|x)$ and $P(y=1|x)$



Softmax function

Motivation

Softmax function

$$P_i = f(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$0 \leq f(z_i) \leq 1$$

$$\sum_i f(z_i) = 1$$

Input		Probability
$z_1 = 1.0$	→ **Softmax** →	$f(z_1) = 0.12$
$z_2 = 3.0$		$f(z_2) = 0.88$

Input		Probability
$z_1 = 1.0$		$f(z_1) = 0.09$
$z_2 = 2.0$	→ **Softmax** →	$f(z_2) = 0.24$
$z_3 = 3.0$		$f(z_3) = 0.67$

Implementation (straightforward)

Softmax function

Chuyển các giá trị của một vector thành các giá trị xác suất

Formula

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

$$0 \le f(x_i) \le 1$$

$$\sum_i f(x_i) = 1$$

Input				Probability
$x_1 = 1.0$				$f(x_1) = 0.09$
$x_2 = 2.0$	➡	**Softmax**	➡	$f(x_2) = 0.24$
$x_3 = 3.0$				$f(x_3) = 0.67$

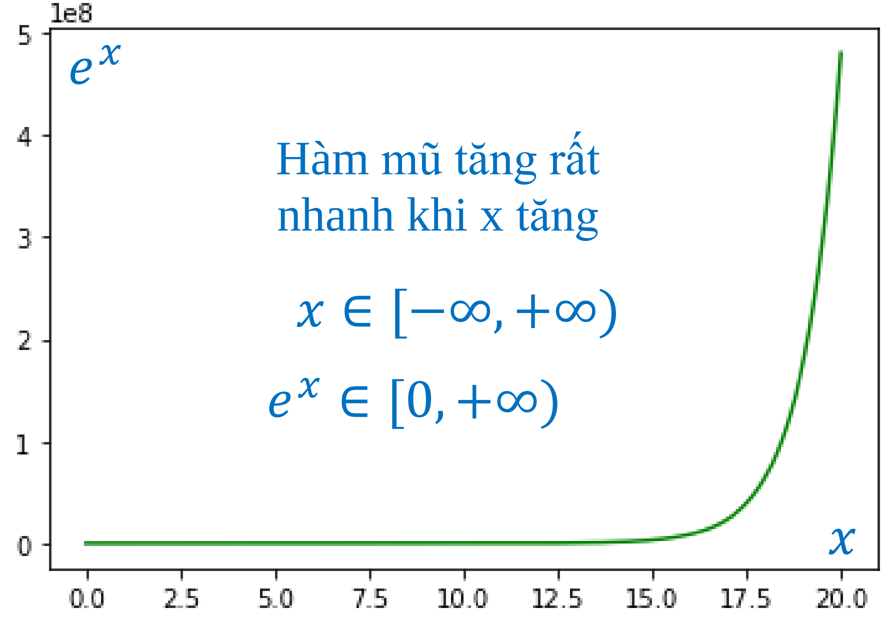
```python
import numpy as np

def softmax(X):
    exps = np.exp(X)
    return exps / np.sum(exps)
```

```python
X = np.array([1.0, 2.0, 3.0])
f = softmax(X)
print(f)
```

```
[0.09003057 0.24472847 0.66524096]
```

```python
X = np.array([1000.0, 1001.0, 1002.0])
f = softmax(X)
print(f)
```

```
[nan nan nan]
```



Giá trị nan vì e^x vượt giới hạn lưu trữ của biến

Softmax function (stable)

(Stable) Formula

$$m = \max(\boldsymbol{x})$$

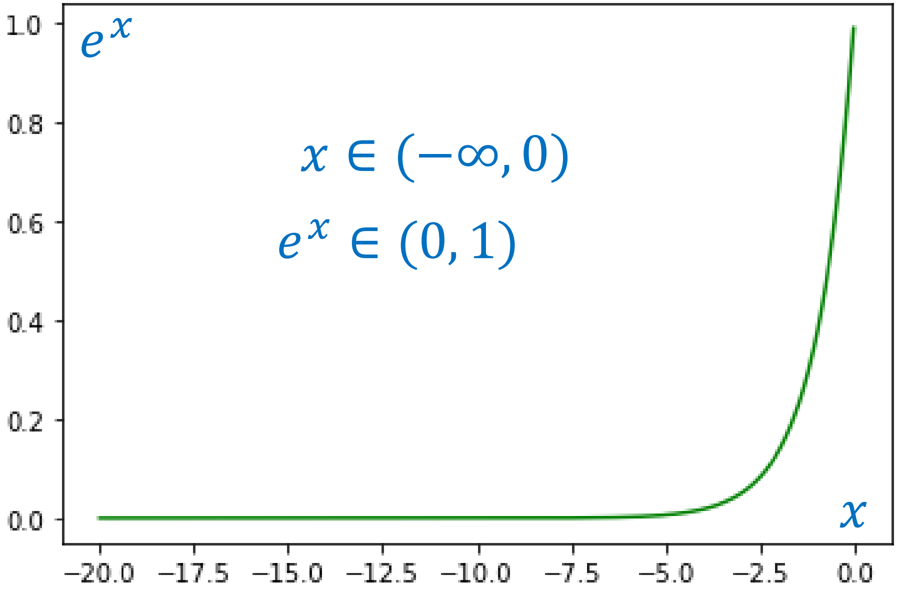
$$f(x_i) = \frac{e^{(x_i - m)}}{\sum_j e^{(x_j - m)}}$$

X	X-m		Probability
$x_1 = 1.0$	$x_1 = -2.0$		$f(x_1) = 0.09$
$x_2 = 2.0$ →	$x_2 = -1.0$ →	**Softmax** →	$f(x_2) = 0.24$
$x_3 = 3.0$	$x_3 = 0$		$f(x_3) = 0.67$

```python
import numpy as np

def stable_softmax(X):
    exps = np.exp(X-np.max(X))
    return exps / np.sum(exps)
```

```python
X = np.array([1.0, 2.0, 3.0])
f = stable_softmax(X)
print(f)
```

```
[0.09003057 0.24472847 0.66524096]
```




```python
X = np.array([1000.0, 1001.0, 1002.0])
f = stable_softmax(X)
print(f)
```

```
[0.09003057 0.24472847 0.66524096]
```

```python
X = np.array([1.0, 1001.0, 1002.0])
f = stable_softmax(X)
print(f)
```

```
[0.         0.26894142 0.73105858]
```

Motivation

Feature **Label**

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

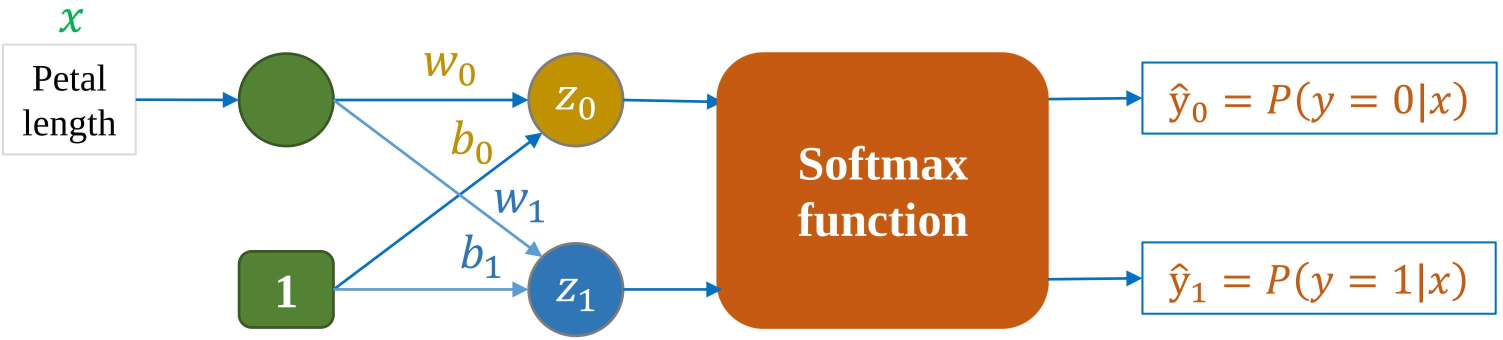
Softmax function

$$P_i = f(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$0 \leq f(z_i) \leq 1$$

$$\sum_i f(z_i) = 1$$

Explicitly output $P(y = 1|x)$ and $P(y = 0|x)$



How about loss function?

$$L(\boldsymbol{\theta}) = -\mathrm{y}\log\hat{\mathrm{y}}_1 - (1-\mathrm{y})\log(\hat{\mathrm{y}}_0)$$

Outline

- Motivation
- Model Construction
- Loss Function
- Generalization (Further Reading)
- Another Approach (Further Reading)

Model Construction

❖ 1-D Feature and two classes

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

#class=2

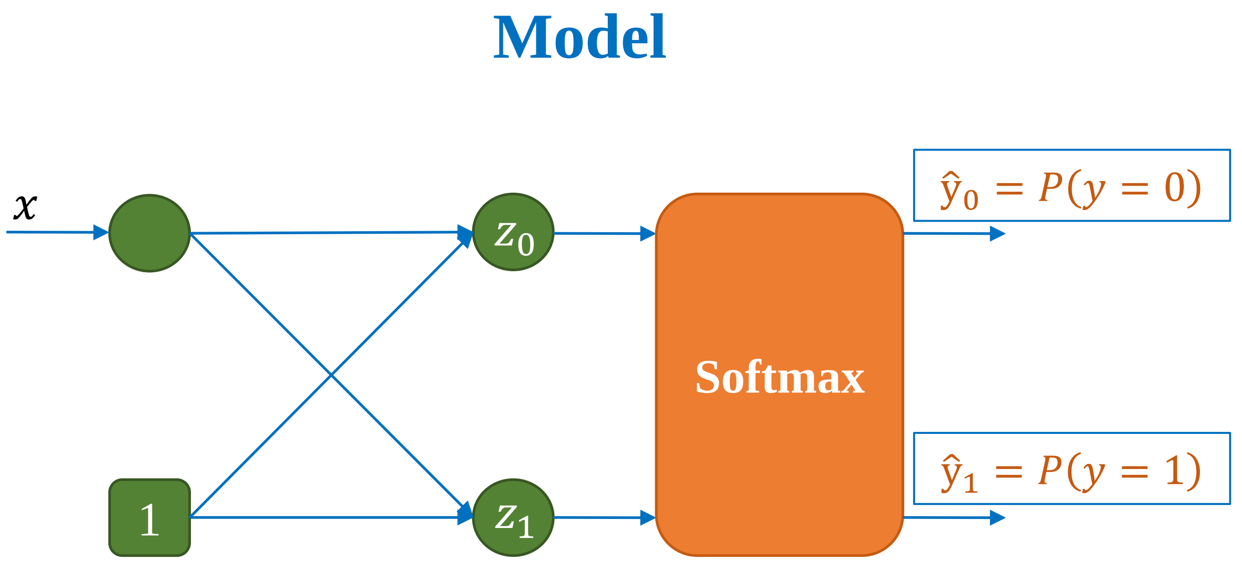
#feature=1

Feature is with one dimension

→ Need one node for input

Two categories

→ Need two node for output



Model Construction

❖ 1-D Feature and three classes

Feature **Label**

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1
5.2	2
5.6	2
5.9	2

#class=3

#feature=1

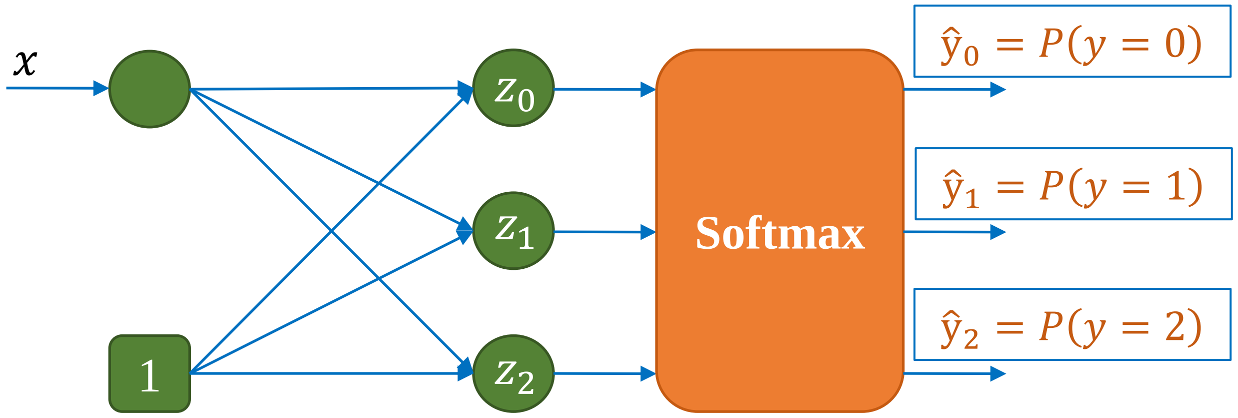
Feature is with one dimension

→ Need one node for input

Three categories

→ Need three nodes for output

Model



Model Construction

❖ 4-D Feature and three classes

Feature | Label

Petal_Length	Petal_Width	Label
1.4	0.2	0
1.4	0.2	0
1.3	0.2	0
4.5	1.5	1
4.9	1.5	1
4	1.3	1
4.5	1.7	2
6.3	1.8	2
5.8	1.8	2

#class=3

#feature=2

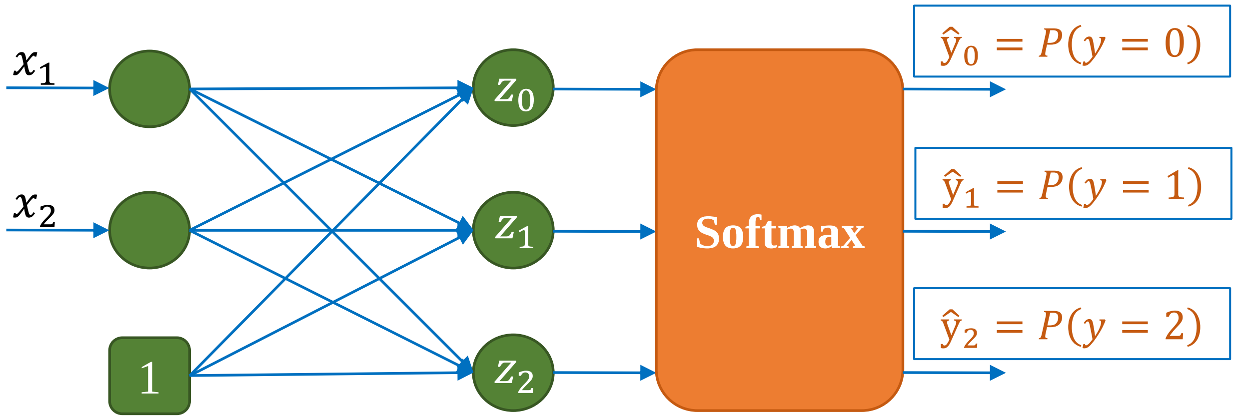
Feature is with two dimensions

→ Need two nodes for input

Three categories

→ Need three nodes for output

Model



Model Construction

❖ 4-D Feature and three classes

Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Label
5.1	3.5	1.4	0.2	0
4.9	3	1.4	0.2	0
4.7	3.2	1.3	0.2	0
6.4	3.2	4.5	1.5	1
6.9	3.1	4.9	1.5	1
5.5	2.3	4	1.3	1
4.9	2.5	4.5	1.7	2
7.3	2.9	6.3	1.8	2
6.7	2.5	5.8	1.8	2

Feature is with four dimensions

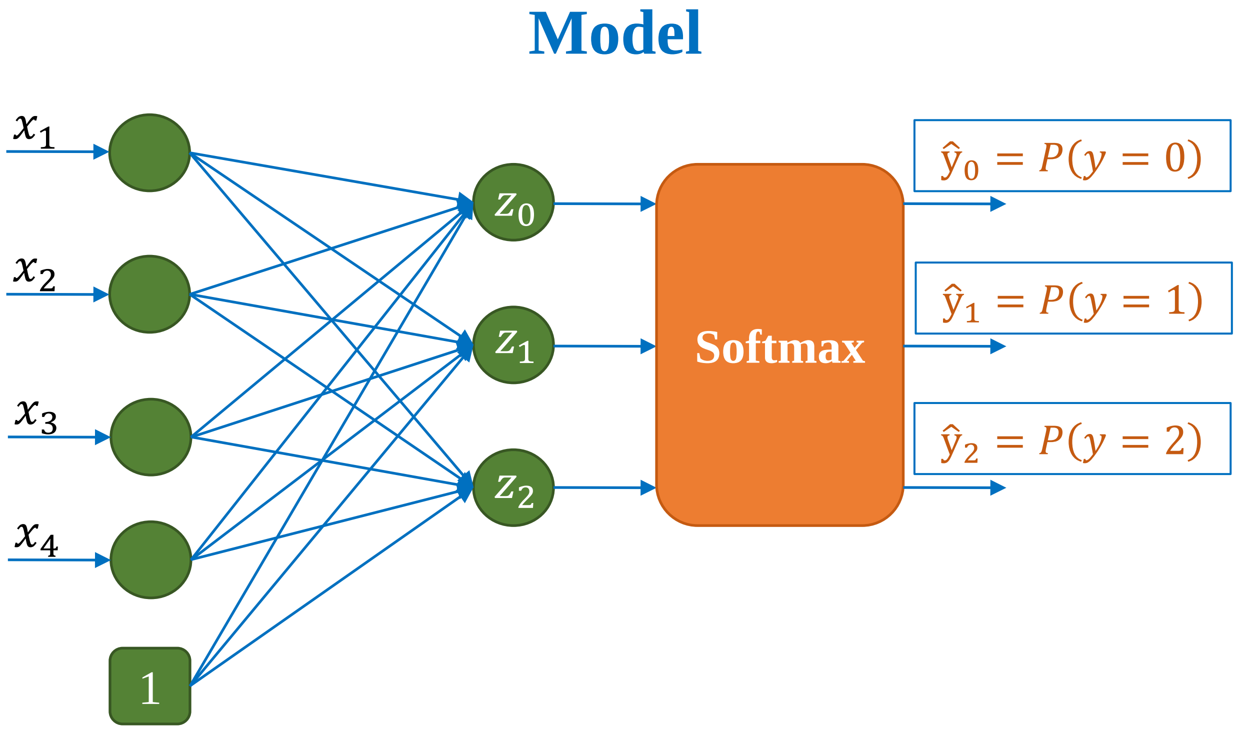
→ Need four nodes for input

Three categories

→ Need three nodes for output

#class=3

#feature=4



Outline

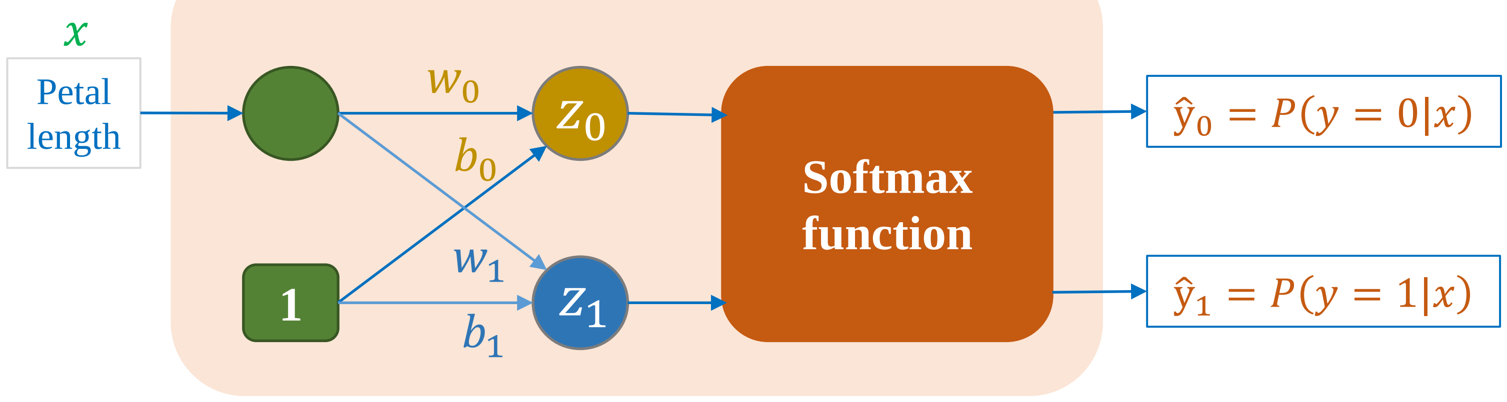
- Motivation
- Model Construction
- Loss Function
- Generalization (Further Reading)
- Another Approach (Further Reading)

Loss function

❖ Simple illustration

Feature	Label
Petal_Length	Category
1.4	0
1	0
1.5	0
3	1
3.8	1
4.1	1

Model



One-hot encoding for label

$$\begin{array}{l} \phantom{y = 0 \rightarrow \boldsymbol{y} = [}\; y_0 \quad y_1 \\ y = 0 \rightarrow \boldsymbol{y} = [1 \quad 0] \\ y = 1 \rightarrow \boldsymbol{y} = [0 \quad 1] \end{array}$$

scalar   vector

$$z_0 = xw_0 + b_0$$

$$z_1 = xw_1 + b_1$$

$$\hat{y}_0 = \frac{e^{z_0}}{\sum_{j=0}^{1} e^{z_j}}$$

$$\hat{y}_1 = \frac{e^{z_1}}{\sum_{j=0}^{1} e^{z_j}}$$

$$\boldsymbol{z} = \begin{bmatrix} z_0 \\ z_1 \end{bmatrix} = \begin{bmatrix} b_0 & w_0 \\ b_1 & w_1 \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_0^T \\ \boldsymbol{\theta}_1^T \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$\hat{\boldsymbol{y}} = \begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \end{bmatrix} = \frac{1}{\sum_{j=0}^{1} e^{z_j}} \begin{bmatrix} e^{z_0} \\ e^{z_1} \end{bmatrix} = \frac{e^{\boldsymbol{z}}}{\sum_{j=0}^{1} e^{z_j}}$$

A vector is by default a column vector $\boldsymbol{\theta}_0 = \begin{bmatrix} b_0 \\ w_0 \end{bmatrix}$

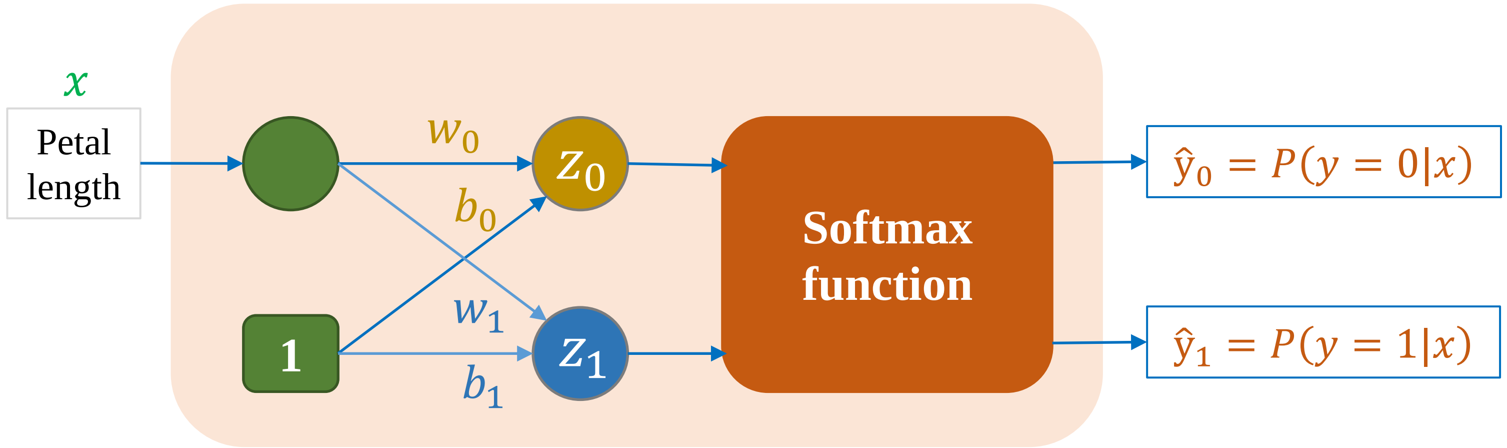
vector transpose $\boldsymbol{\theta}_0^T = [b_0 \quad w_0]$

Loss function

❖ Simple illustration

Feature	Label
Petal_Length	Category
1.4	0
1	0
1.5	0
3	1
3.8	1
4.1	1

Model



One-hot encoding for label

$$\begin{array}{l} \phantom{y = 0 \rightarrow \boldsymbol{y} = [}\; y_0 \quad y_1 \\ y = 0 \rightarrow \boldsymbol{y} = [1 \quad 0] \\ y = 1 \rightarrow \boldsymbol{y} = [0 \quad 1] \end{array}$$

scalar vector

$$z_0 = xw_0 + b_0$$

$$z_1 = xw_1 + b_1$$

$$\hat{y}_0 = \frac{e^{z_0}}{\sum_{j=0}^{1} e^{z_j}}$$

$$\hat{y}_1 = \frac{e^{z_1}}{\sum_{j=0}^{1} e^{z_j}}$$

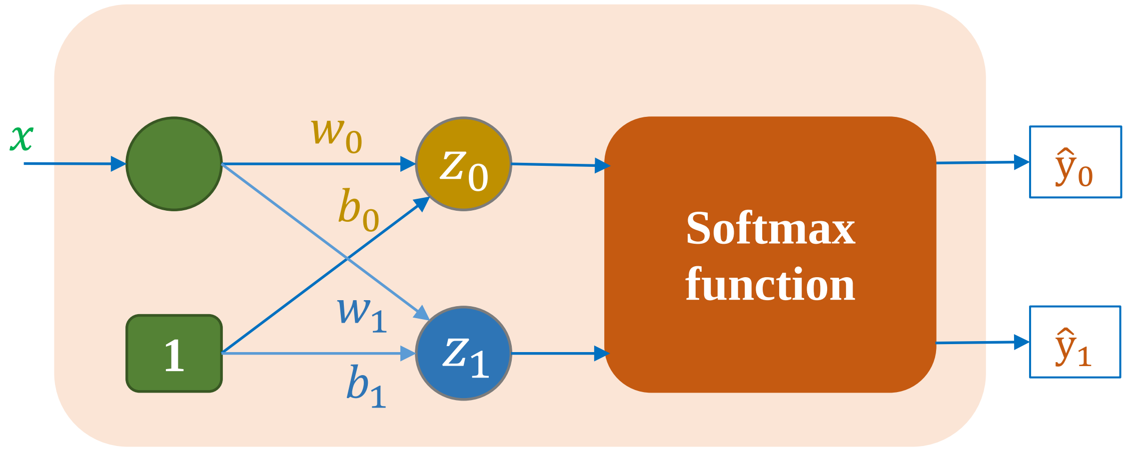
$$\boldsymbol{z} = \begin{bmatrix} z_0 \\ z_1 \end{bmatrix} = \begin{bmatrix} b_0 & w_0 \\ b_1 & w_1 \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_0^T \\ \boldsymbol{\theta}_1^T \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$\hat{\boldsymbol{y}} = \begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \end{bmatrix} = \frac{1}{\sum_{j=0}^{1} e^{z_j}} \begin{bmatrix} e^{z_0} \\ e^{z_1} \end{bmatrix} = \frac{e^{\boldsymbol{z}}}{\sum_{j=0}^{1} e^{z_j}}$$

$$L(\boldsymbol{\theta}) = -y_0 \log \hat{y}_0 - y_1 \log \hat{y}_1 = -\sum_{i=0}^{1} y_i \log \hat{y}_i = -\boldsymbol{y}^T log \hat{\boldsymbol{y}}$$

Loss function

Model



$$L(\boldsymbol{\theta}) = -y_0\log\hat{y}_0 - y_1\log\hat{y}_1 = -\sum_{i=0}^{1} y_i\log\hat{y}_i = -\boldsymbol{y}^T log\hat{\boldsymbol{y}}$$

$$\hat{y}_0 = \frac{e^{z_0}}{\sum_{j=0}^{1} e^{z_j}}$$

$$\hat{y}_1 = \frac{e^{z_1}}{\sum_{j=0}^{1} e^{z_j}}$$

Derivative

$$\frac{\partial \hat{y}_i}{\partial z_j} = \begin{cases} \hat{y}_i(1-\hat{y}_i) & if\ i = j \\ -\hat{y}_i\hat{y}_j & if\ i \neq j \end{cases}$$

$$\begin{aligned} \frac{\partial L}{\partial z_i} &= -\sum_k y_k \frac{\partial \log(\hat{y}_k)}{\partial z_i} \\ &= -\sum_k y_k \frac{\partial \log(\hat{y}_k)}{\partial \hat{y}_k}\frac{\partial \hat{y}_k}{\partial z_i} \\ &= -\sum_k y_k \frac{1}{\hat{y}_k}\frac{\partial \hat{y}_k}{\partial z_i} \end{aligned}$$

$$\begin{aligned} \frac{\partial L}{\partial z_i} &= -y_i(1-\hat{y}_i) - \sum_{k\neq i} y_k \frac{1}{\hat{y}_k}(-\hat{y}_k\hat{y}_i) \\ &= -y_i(1-\hat{y}_i) + \sum_{k\neq i} y_k\hat{y}_i \\ &= -y_i + y_i\hat{y}_i + \sum_{k\neq i} y_k\hat{y}_i \\ &= \hat{y}_i\left(y_i + \sum_{k\neq i} y_k\right) - y_i \\ &= \hat{y}_i - y_i \end{aligned}$$

Loss function

One-hot encoding for label

$$y = 0 \rightarrow \boldsymbol{y}^T = \begin{array}{c} \quad y_0 \;\; y_1 \\ [1 \;\; 0] \end{array}$$

$$y = 1 \rightarrow \boldsymbol{y}^T = [0 \;\; 1]$$

scalar — vector

$$z_0 = xw_0 + b_0$$

$$z_1 = xw_1 + b_1$$

$$\hat{y}_0 = \frac{e^{z_0}}{\sum_{j=0}^{1} e^{z_j}}$$

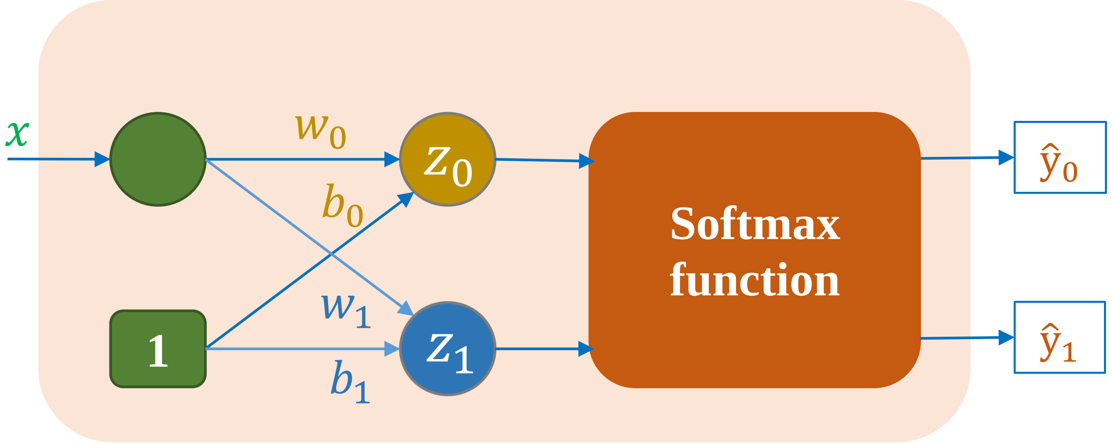
$$\hat{y}_1 = \frac{e^{z_1}}{\sum_{j=0}^{1} e^{z_j}}$$

$$\boldsymbol{z} = \begin{bmatrix} z_0 \\ z_1 \end{bmatrix} = \begin{bmatrix} b_0 & w_0 \\ b_1 & w_1 \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_0^T \\ \boldsymbol{\theta}_1^T \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \end{bmatrix} = \frac{1}{\sum_{j=0}^{1} e^{z_j}} \begin{bmatrix} e^{z_0} \\ e^{z_1} \end{bmatrix} = \frac{e^{\boldsymbol{z}}}{\sum_{j=0}^{1} e^{z_j}}$$

$$L(\boldsymbol{\theta}) = -\sum_{i=0}^{1} y_i \log \hat{y}_i = -\boldsymbol{y}^T log \hat{\boldsymbol{y}}$$

Model



Derivative

$$\frac{\partial L}{\partial \hat{y}_i} = -\frac{y_i}{\hat{y}_i}$$

$$\frac{\partial \hat{y}_i}{\partial z_j} = \begin{cases} \hat{y}_i(1 - \hat{y}_i) & if \; i = j \\ -\hat{y}_i \hat{y}_j & if \; i \neq j \end{cases}$$

$$\frac{\partial L}{\partial z_i} = \hat{y}_i - y_i$$

$$\frac{\partial L}{\partial w_i} = x(\hat{y}_i - y_i)$$

$$\frac{\partial L}{\partial b_i} = \hat{y}_i - y_i$$

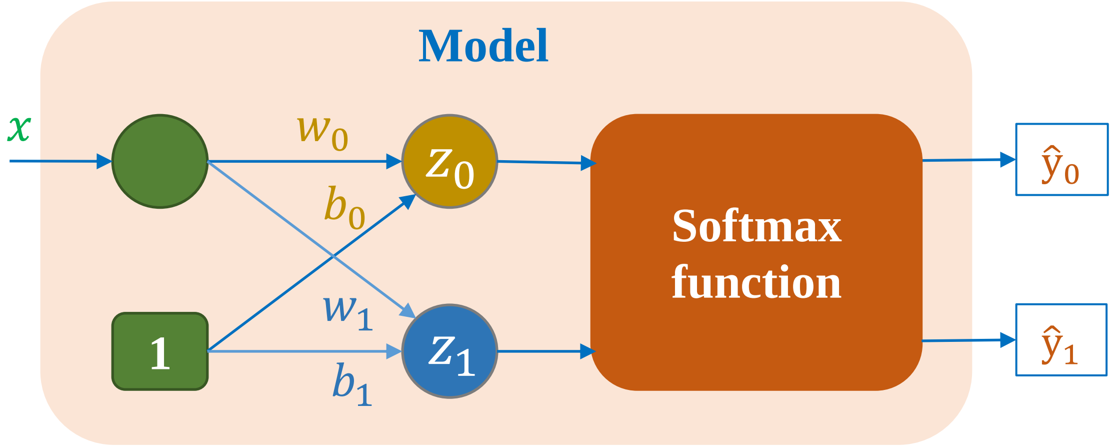
Simple Illustration - Summary

Feature	Label
Petal_Length	Category
1.4	0
1	0
1.5	0
3	1
3.8	1
4.1	1

One-hot encoding for label

$$\begin{array}{l} \qquad\qquad\qquad\qquad\quad y_0 \quad y_1 \\ y = 0 \rightarrow \boldsymbol{y}^T = [1 \quad 0] \\ y = 1 \rightarrow \boldsymbol{y}^T = [0 \quad 1] \end{array}$$

scalar vector



$$\boldsymbol{\theta} = \begin{bmatrix} b_0 & b_1 \\ w_0 & w_1 \end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix} 1 \\ x \end{bmatrix}$$

1. Forward computation

$$\boldsymbol{z} = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$\hat{\boldsymbol{y}} = \frac{e^{\boldsymbol{z}}}{\sum_{j=0}^{1} e^{z_j}}$$

2. Loss function

$$L(\boldsymbol{\theta}) = -\boldsymbol{y}^T log\hat{\boldsymbol{y}}$$

3. Derivative

$$\frac{\partial L}{\partial w_i} = x(\hat{y}_i - y_i) \qquad \frac{\partial L}{\partial b_i} = \hat{y}_i - y_i$$

$$\nabla_{\boldsymbol{\theta}} L = \boldsymbol{x}(\hat{\boldsymbol{y}} - \boldsymbol{y})^T$$

4. Update

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

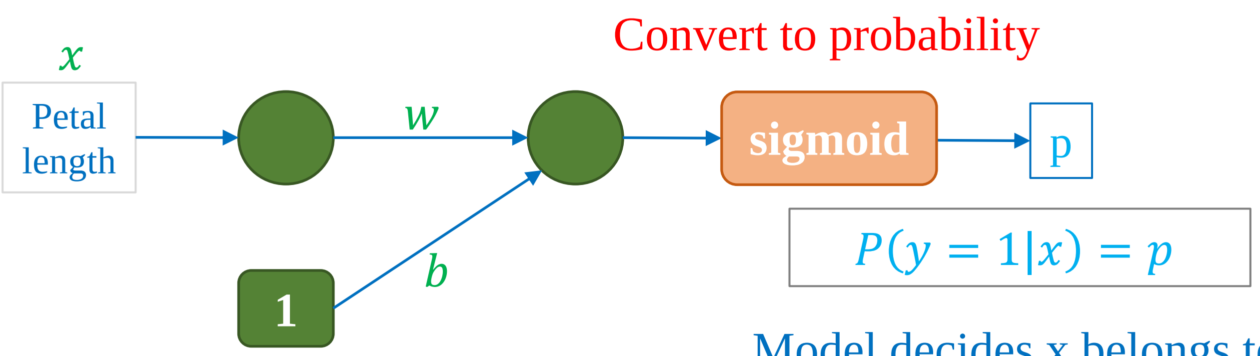
η is learning rate

Explaining Cross-entropy in another way

Motivation

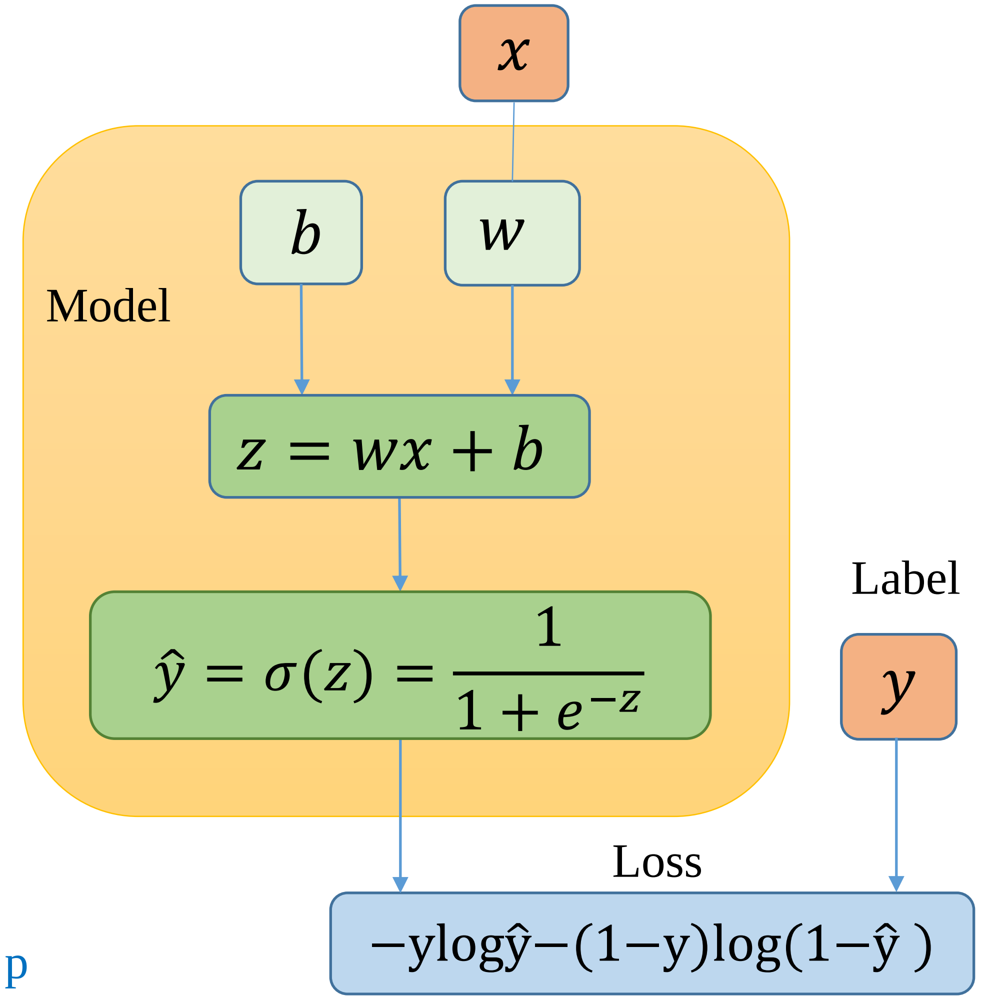
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

(Feature: Petal_Length; Label: Label)



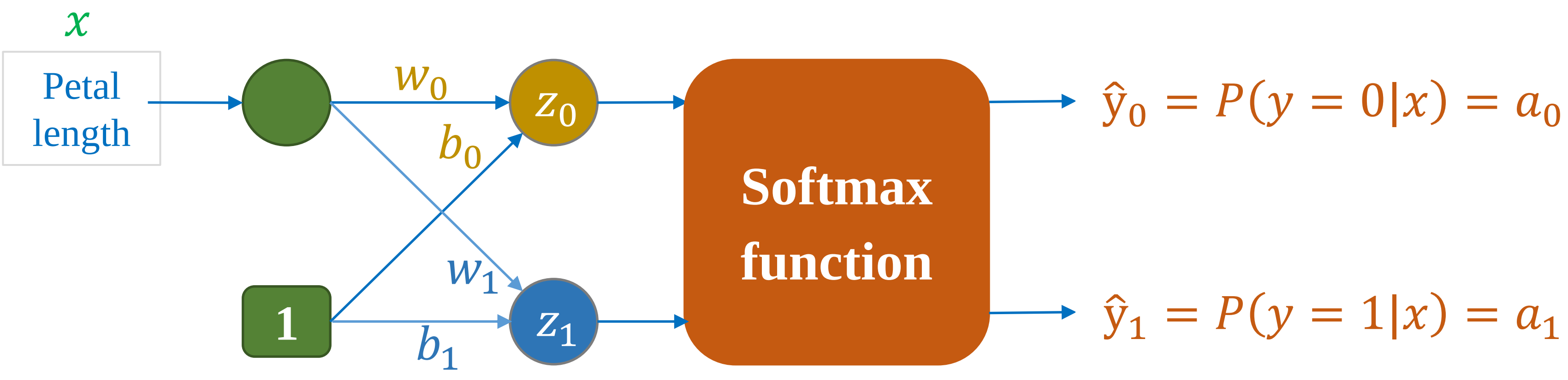
Model decides x belongs to the category (label=1) with the belief of p

Implicitly conclude that $P(y = 0|x) = 1 - p$



Outputs of Model

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1



Softmax function

$$\hat{y}_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$0 \leq f(z_i) \leq 1$$

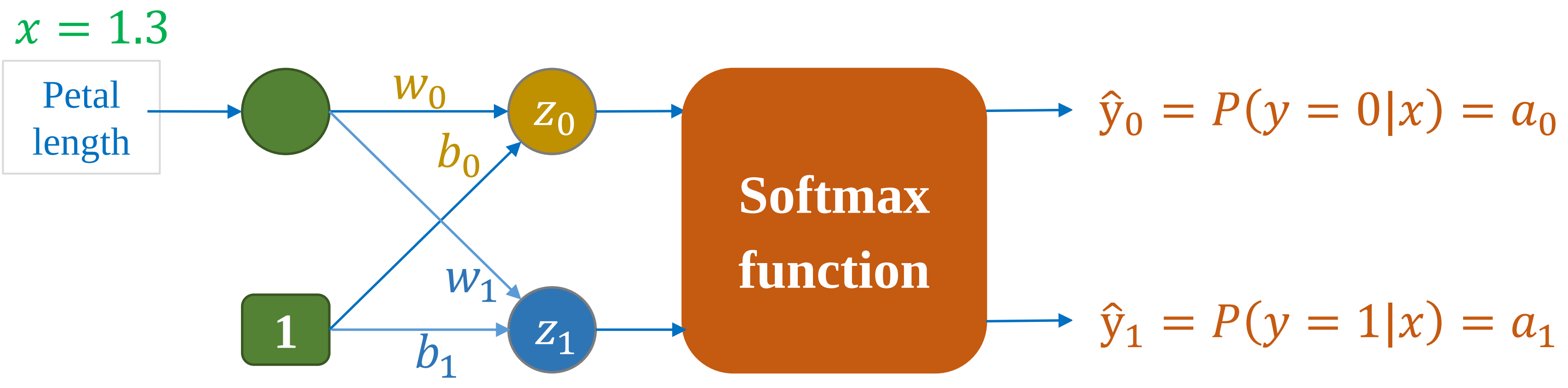
$$\sum_i f(z_i) = 1$$

Explicitly output $P(y = 1|x)$ and $P(y = 0|x)$

For a Given Sample

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

Given a sample $(x = 1.3, y = 0)$



Softmax function

$$\hat{y}_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$0 \leq f(z_i) \leq 1$$

$$\sum_i f(z_i) = 1$$

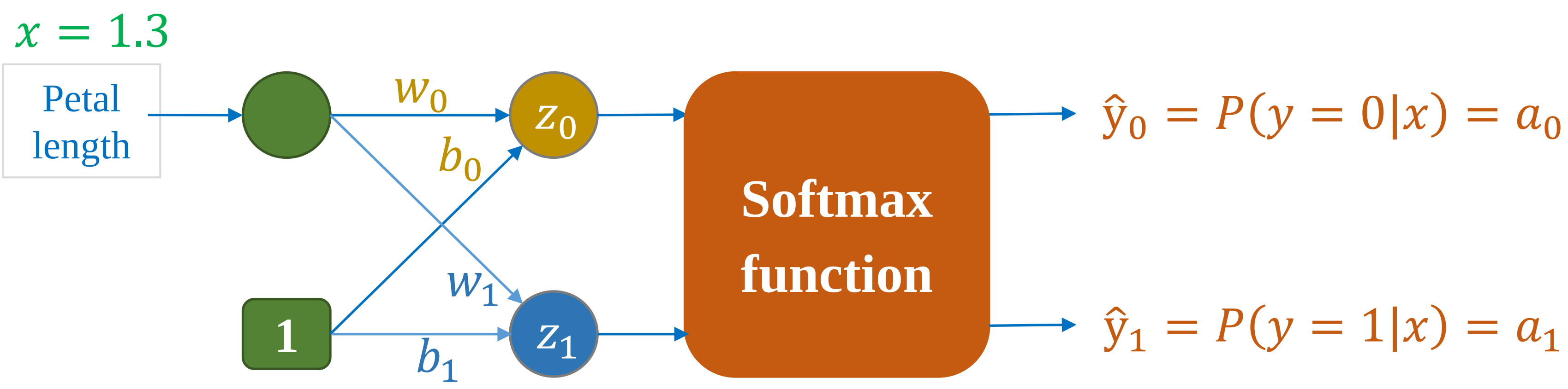
With $(x = 1.3, y = 0)$, model becomes better when a_0 increases and a_1 decreases

Differences between increasing a_0 and decreasing a_1?

For a Given Sample

Given a sample ($x = 1.3, y = 0$)

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1



With ($x = 1.3, y = 0$), model becomes better when a_0 increases and a_1 decreases

Softmax function

$$\hat{y}_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

$$0 \leq f(z_i) \leq 1$$

$$\sum_i f(z_i) = 1$$

Increasing a_0: $\hat{y}_0 = \frac{e^{z_0}}{e^{z_0} + e^{z_1}}$ ➡ increasing z_0 decreasing z_1

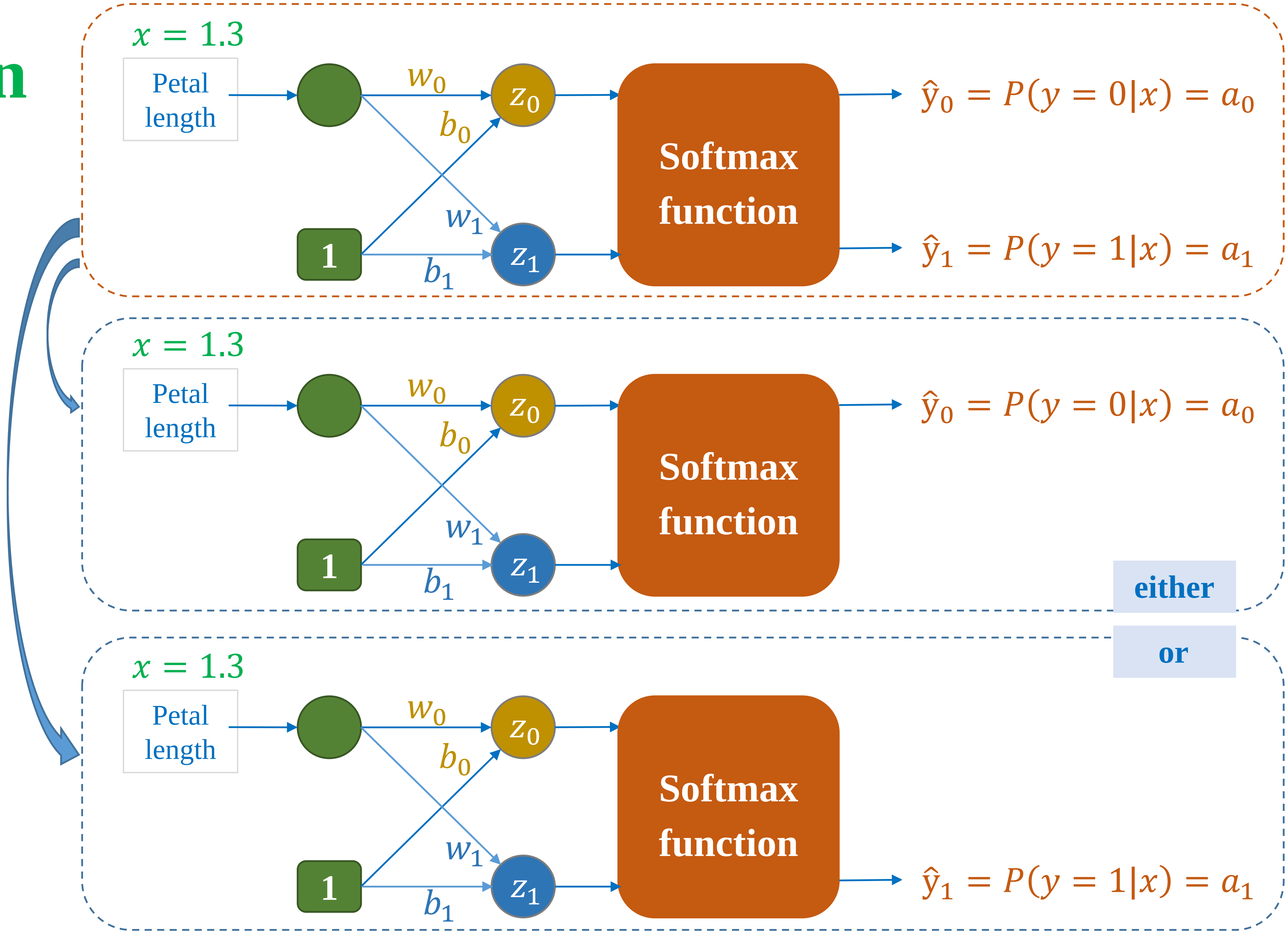
Decreasing a_1: $\hat{y}_1 = \frac{e^{z_1}}{e^{z_0} + e^{z_1}}$ ➡ increasing z_0 decreasing z_1

Observation

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

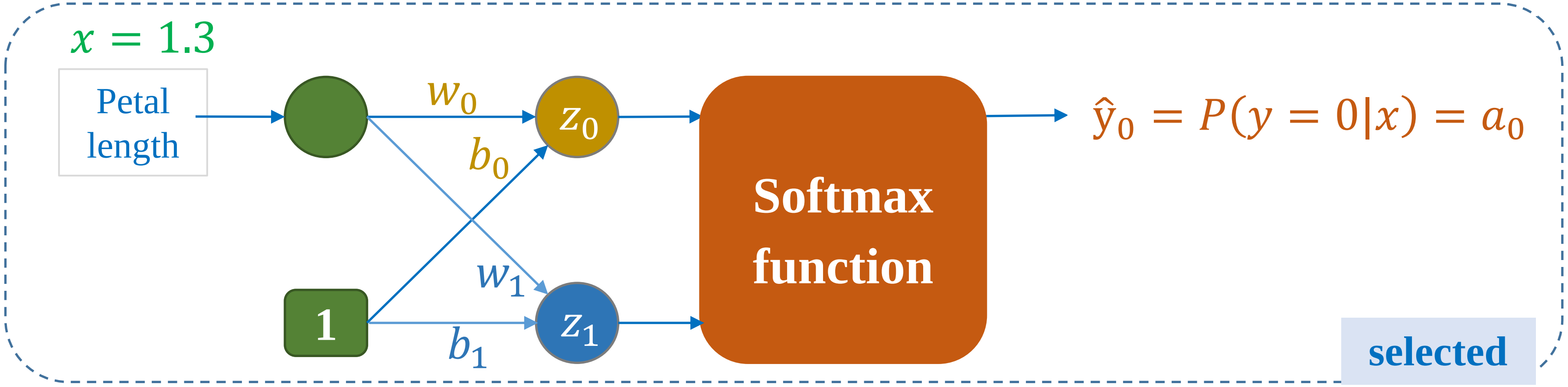
With $(x = 1.3, y = 0)$, model becomes better when a_0 increases and a_1 decreases

→ increasing z_0
decreasing z_1



Loss Computation

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1



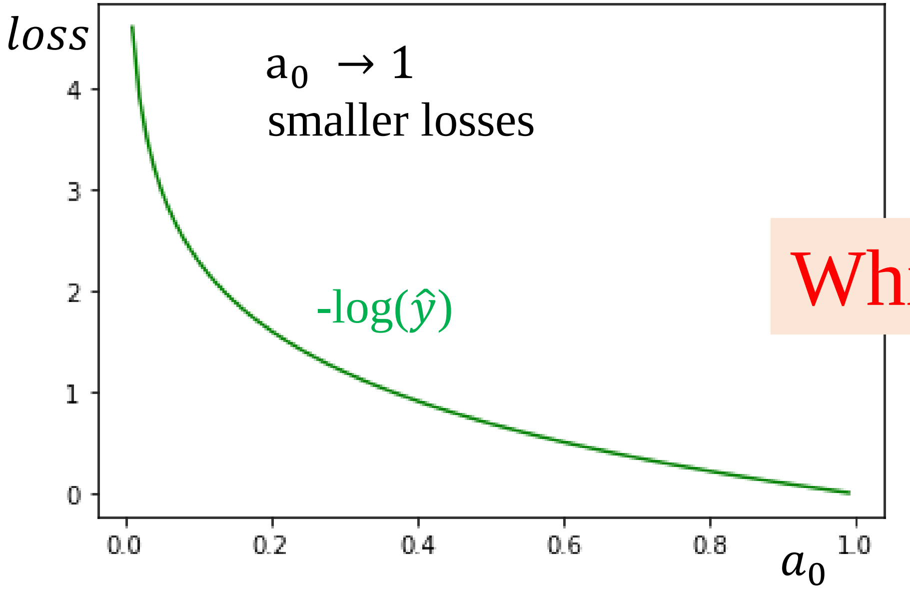
$a_0 \in [0,1]$

When $a_0 = 0$, the model (or θ) is worst

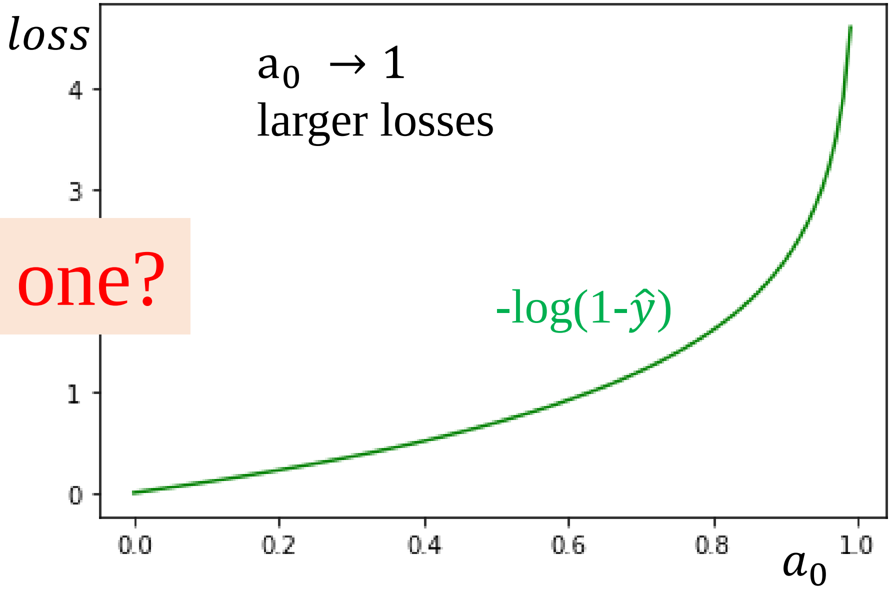
When $a_0 = 1$, the model (or θ) is perfect

With $(x = 1.3, y = 0)$, model becomes better when a_0 increases and a_1 decreases

➡ increasing z_0 decreasing z_1

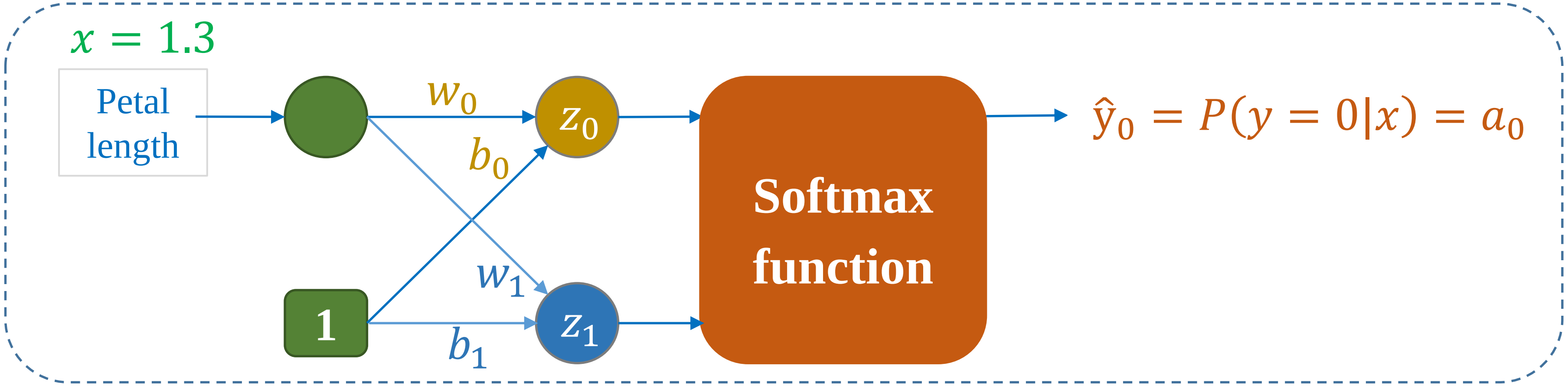


Which one?



Loss Computation

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1



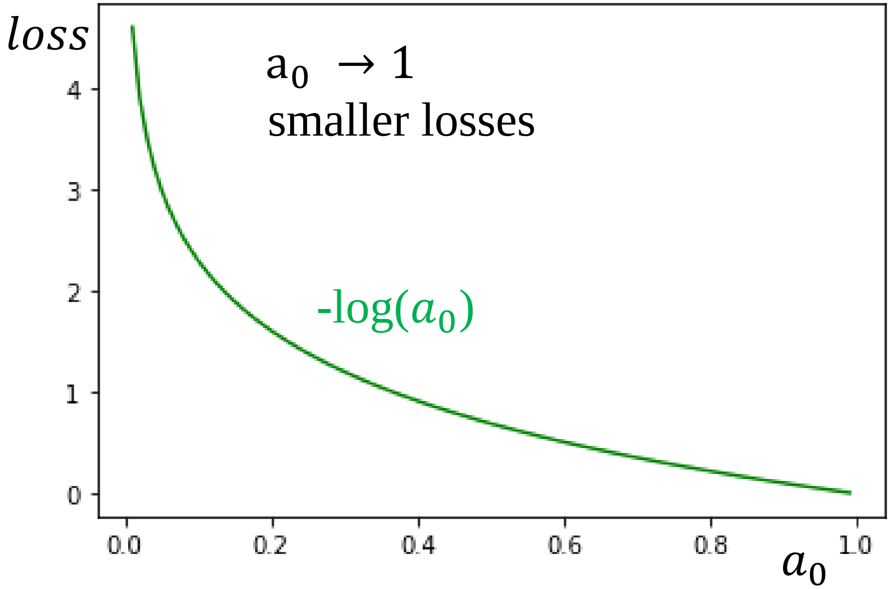
$a_0 \in [0,1]$

When $a_0 = 0$, the model (or θ) is worst

When $a_0 = 1$, the model (or θ) is perfect

With $(x = 1.3, y = 0)$, model becomes better when a_0 increases and a_1 decreases

→ increasing z_0 decreasing z_1



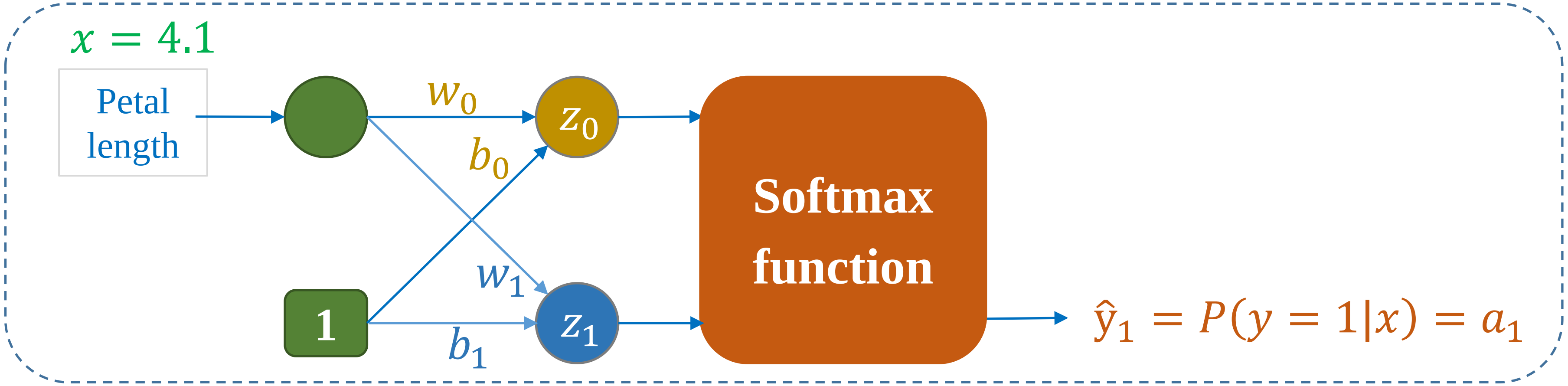
Loss function

$$L(\boldsymbol{\theta}) = -\log(\hat{y}_0)$$

Another Sample

Feature **Label**

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1



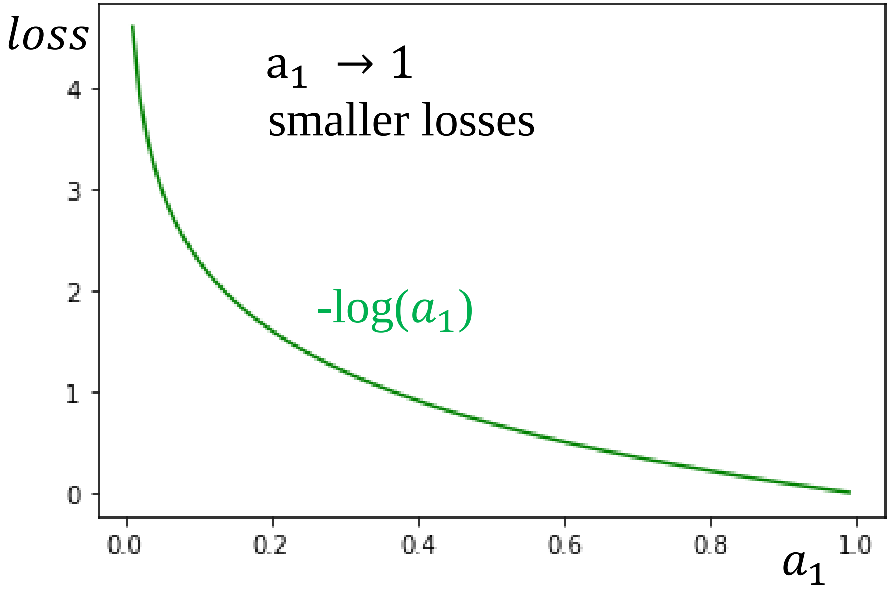
$a_1 \in [0,1]$

When $a_1 = 0$, the model (or θ) is worst

When $a_1 = 1$, the model (or θ) is perfect

With $(x = 4.1, y = 1)$, model becomes better when a_1 increases and a_0 decreases

➡ increasing z_1
decreasing z_0

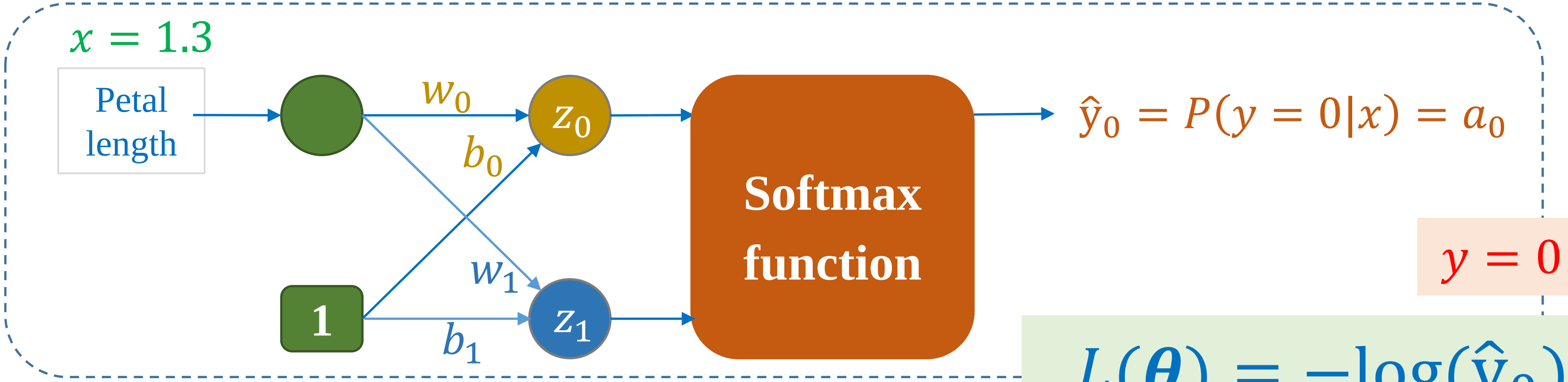


Loss function

$$L(\boldsymbol{\theta}) = -\log(\hat{y}_1)$$

Observation

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

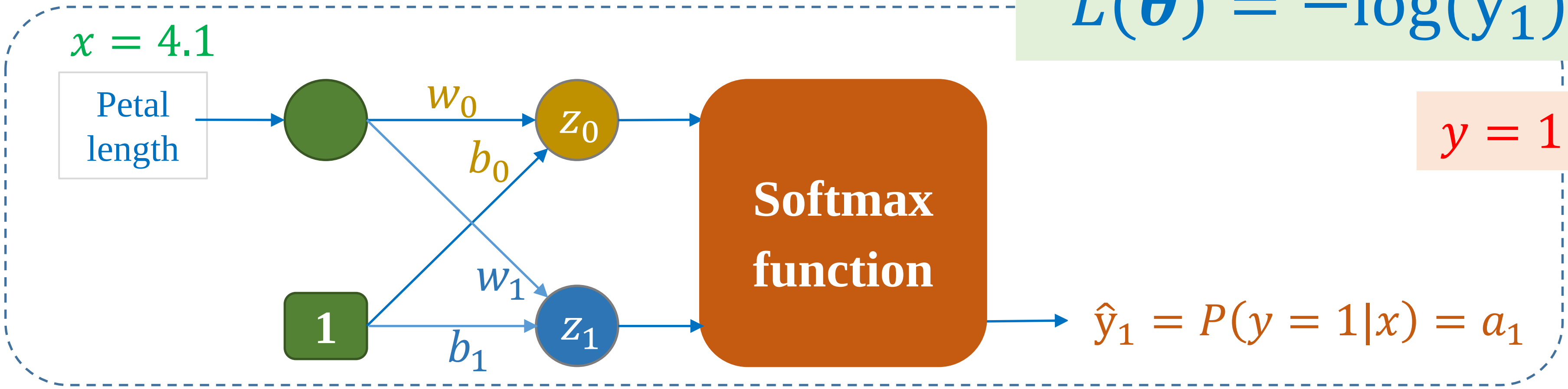


$y = 0$

$$L(\boldsymbol{\theta}) = -\log(\hat{y}_0)$$

$$L(\boldsymbol{\theta}) = -\log(\hat{y}_1)$$

$y = 1$



With $(x = \cdots, y = ?)$, model becomes better when $a_?$ increases

Loss function

$$L(\boldsymbol{\theta}) = -\mathrm{y}\log\hat{y}_1 - (1-\mathrm{y})\log(\hat{y}_0)$$

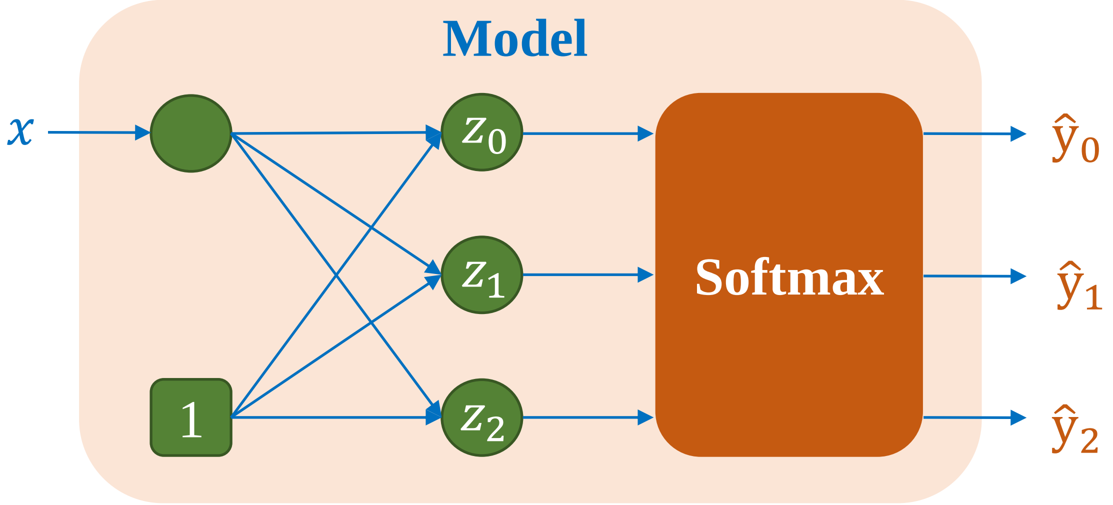
What about 3+ classes?

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1
5.2	2
5.6	2
5.9	2

#features = 1

#classes = 3

$y \in \{0,1,2\}$



$$\left.\begin{array}{l} y = 0 \rightarrow L(\boldsymbol{\theta}) = -\log(\hat{y}_0) \\ y = 1 \rightarrow L(\boldsymbol{\theta}) = -\log(\hat{y}_1) \\ y = 2 \rightarrow L(\boldsymbol{\theta}) = -\log(\hat{y}_2) \end{array}\right\}$$

How to convert into a single function?

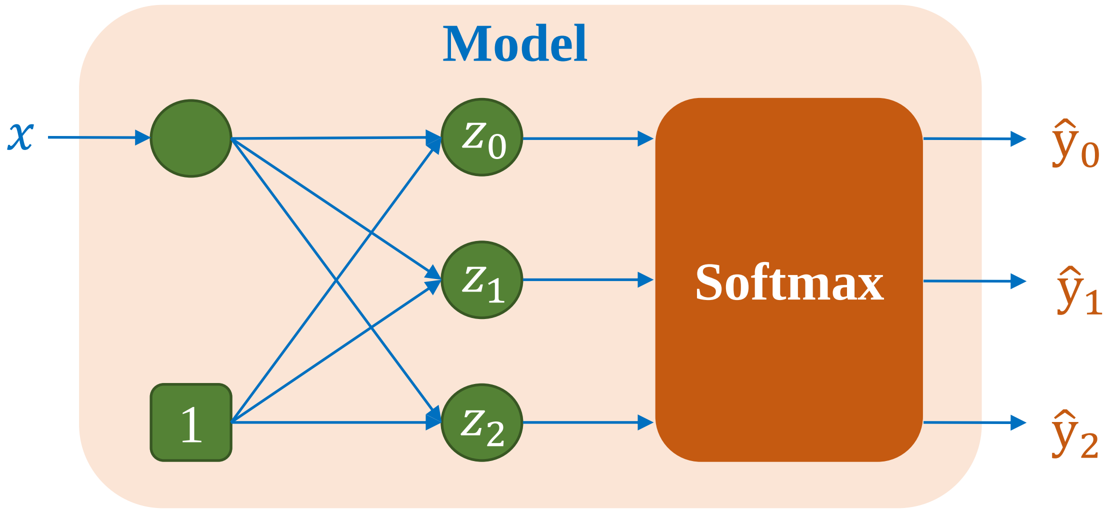
A Suggested Function

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1
5.2	2
5.6	2
5.9	2

#features = 1

#classes = 3

$y \in \{0,1,2\}$



$$L(\theta) = \underbrace{-\frac{y(1-y)}{-2}\log(\hat{y}_2)}_{y=2}\underbrace{-y(2-y)\log(\hat{y}_1)}_{y=1}\underbrace{-(1-y)\left(\frac{2-y}{2}\right)\log(\hat{y}_0)}_{y=0}$$

Ok! but awkward!!! … and how to improve?

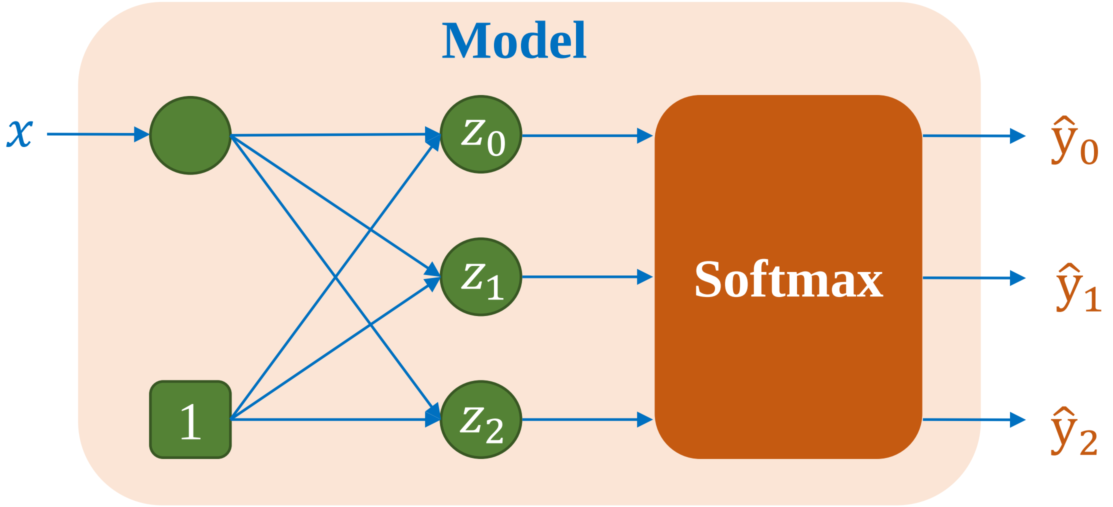
Using One-Hot Encoding

Feature	Label
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1
5.2	2
5.6	2
5.9	2

#features = 1

#classes = 3

$y \in \{0,1,2\}$



One-hot encoding for label

$$\boldsymbol{y} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} \qquad y_i \in \{0,1\} \qquad \sum_i y_i = 1$$

$$y = 0 \rightarrow \boldsymbol{y} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad y = 2 \rightarrow \boldsymbol{y} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad y = 1 \rightarrow \boldsymbol{y} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Loss function

$$\mathrm{L}(\boldsymbol{\theta}) = -y_2\log(\hat{y}_2) - y_1\log(\hat{y}_1) - y_0\log(\hat{y}_0)$$

$$= -\sum_i y_i \log(\hat{y}_i)$$

Summary

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

$$\boldsymbol{x} = \begin{bmatrix} 1 \\ x \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} b_0 & b_1 \\ w_0 & w_1 \end{bmatrix}$$

One-hot encoding for label

$$\begin{array}{l} \qquad\qquad\qquad\qquad\quad y_0 \;\; y_1 \\ y = 0 \rightarrow \boldsymbol{y}^T = [1 \quad 0] \\ y = 1 \rightarrow \boldsymbol{y}^T = [0 \quad 1] \end{array}$$

scalar

vector

$$z_0 = xw_0 + b_0$$

$$z_1 = xw_1 + b_1$$

$$\hat{y}_0 = \frac{e^{z_0}}{\sum_{j=0}^{1} e^{z_j}}$$

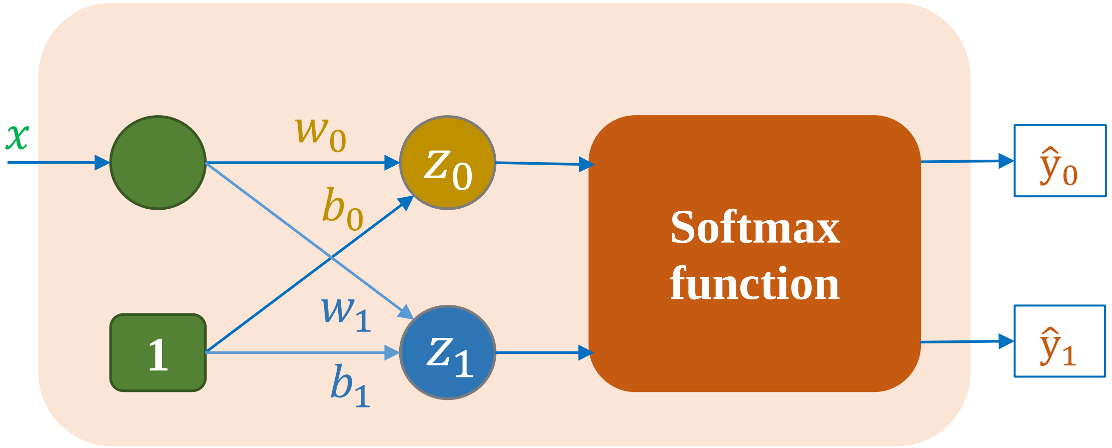
$$\hat{y}_1 = \frac{e^{z_1}}{\sum_{j=0}^{1} e^{z_j}}$$

$$\boldsymbol{z} = \begin{bmatrix} z_0 \\ z_1 \end{bmatrix} = \begin{bmatrix} b_0 & w_0 \\ b_1 & w_1 \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}_0^T \\ \boldsymbol{\theta}_1^T \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix} = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$\hat{\boldsymbol{y}} = \begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \end{bmatrix} = \frac{1}{\sum_{j=0}^{1} e^{z_j}} \begin{bmatrix} e^{z_0} \\ e^{z_1} \end{bmatrix} = \frac{e^{\boldsymbol{z}}}{\sum_{j=0}^{1} e^{z_j}}$$

$$L(\boldsymbol{\theta}) = -\sum_{i=0}^{1} y_i \log \hat{y}_i = -\boldsymbol{y}^T log \hat{\boldsymbol{y}}$$

Model



Derivative

$$\frac{\partial L}{\partial \hat{y}_i} = -\frac{y_i}{\hat{y}_i}$$

$$\frac{\partial \hat{y}_i}{\partial z_j} = \begin{cases} \hat{y}_i(1 - \hat{y}_i) & if\ i = j \\ -\hat{y}_i \hat{y}_j & if\ i \neq j \end{cases}$$

$$\frac{\partial L}{\partial z_i} = \hat{y}_i - y_i$$

$$\frac{\partial L}{\partial w_i} = x(\hat{y}_i - y_i)$$

$$\frac{\partial L}{\partial b_i} = \hat{y}_i - y_i$$

Outline

- Motivation
- Model Construction
- Loss Function
- Generalization (Further Reading)
- Another Approach (Further Reading)

Softmax Regression – Naïve

Training data

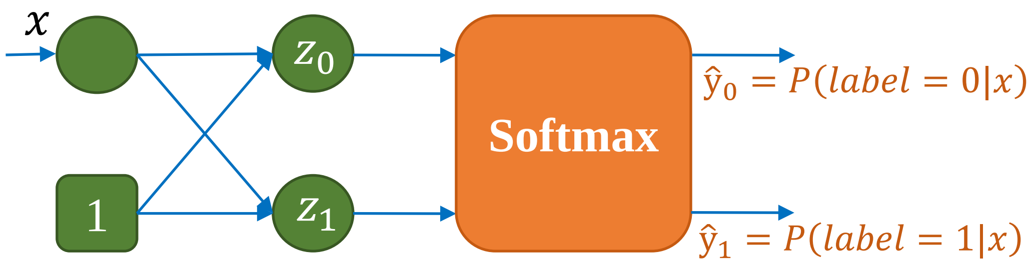
Feature	**Label**
Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

Category A (rows 1.4, 1.3, 1.5)

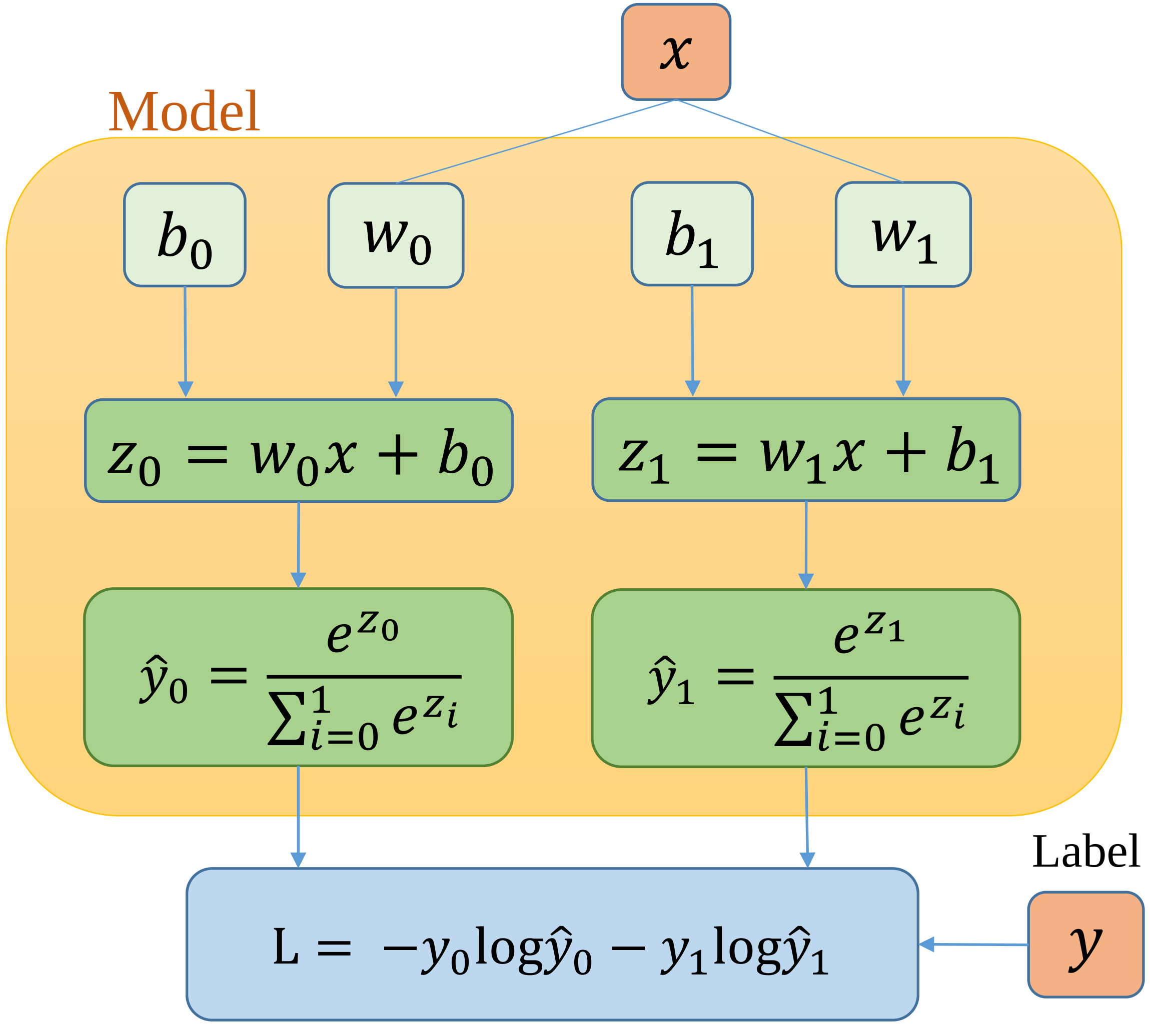
Category B (rows 4.5, 4.1, 4.6)

One-hot encoding for labels

$$\begin{array}{l} \quad\quad\quad\quad\quad\quad \text{index } 0 \;\; 1 \\ y = 0 \rightarrow \boldsymbol{y}^T = [1, 0] \\ y = 1 \rightarrow \boldsymbol{y}^T = [0, 1] \end{array}$$



Model



Softmax Regression - Naïve

Training data

Feature **Label**

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

#class=2

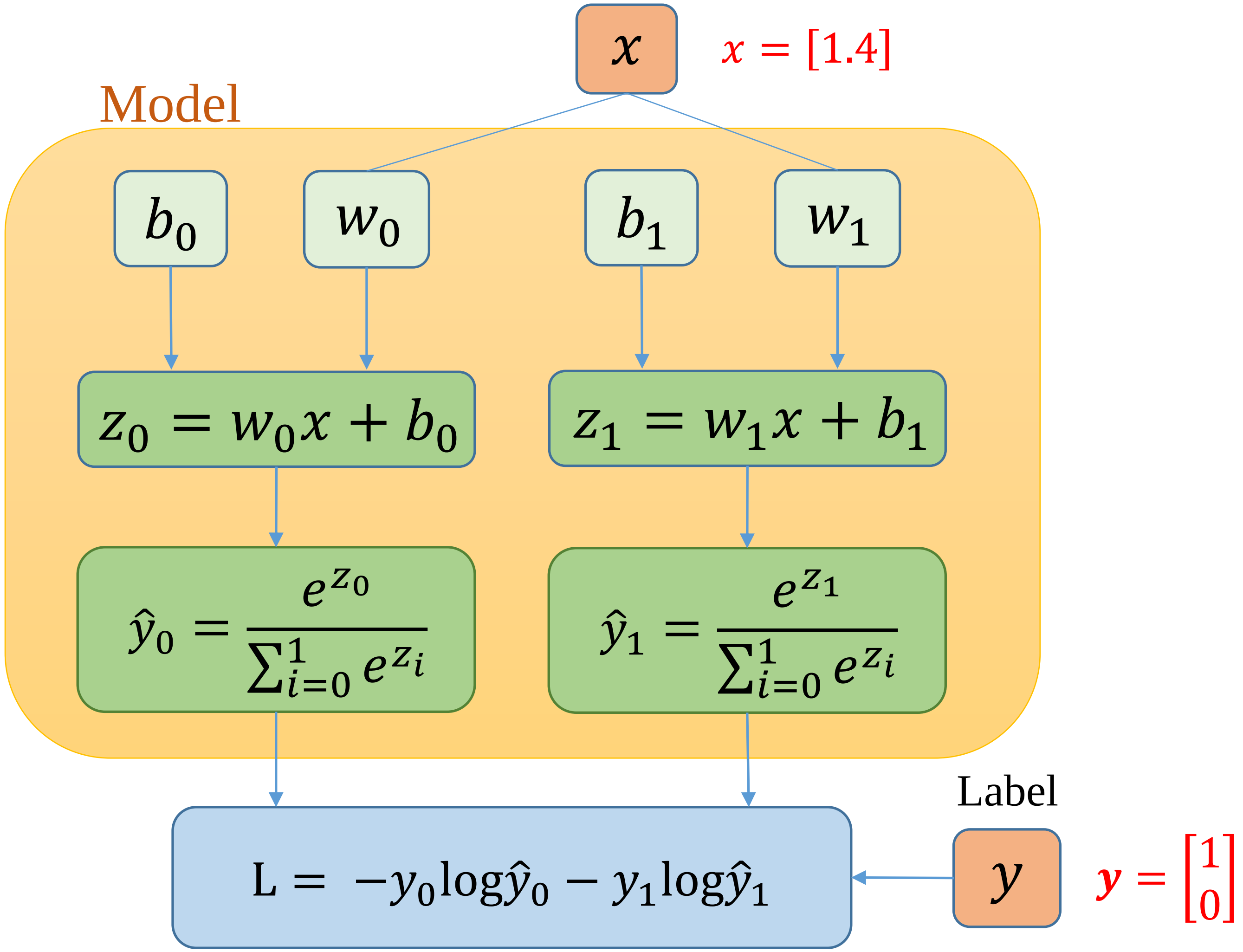
#feature=1

One-hot encoding for label

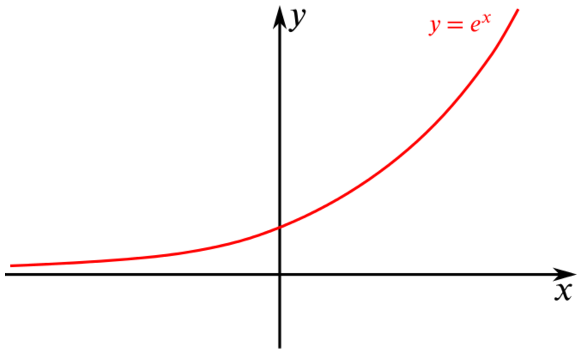
$$\begin{array}{l} \\ y = 0 \rightarrow \boldsymbol{y}^T = \\ y = 1 \rightarrow \boldsymbol{y}^T = \end{array} \begin{array}{cc} y_0 & y_1 \\ [1 & 0] \\ [0 & 1] \end{array}$$

Training example

$$(x, y) = (1.4, 0)$$



Softmax Regression Naïve



Training data

Feature **Label**

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

#class=2

#feature=1

One-hot encoding for label

$$
\begin{array}{l}
\qquad\qquad\qquad\quad\;\; y_0 \;\; y_1 \\
y = 0 \rightarrow \boldsymbol{y}^T = [1 \;\; 0] \\
y = 1 \rightarrow \boldsymbol{y}^T = [0 \;\; 1]
\end{array}
$$

Training example

$$(x, y) = (1.4, 0)$$

$\boldsymbol{x} = [1.4]$ x

b_0 0.1 w_0 0.2 b_1 0.05 w_1 -0.1

$\boldsymbol{z}_0 = [0.38]$ $z_0 = w_0 x + b_0$

$z_1 = w_1 x + b_1$ $\boldsymbol{z}_1 = [-0.09]$

$\hat{\boldsymbol{y}}_0 = [0.615]$ $\hat{y}_0 = \dfrac{e^{z_0}}{\sum_{i=0}^{1} e^{z_i}}$

$\hat{y}_1 = \dfrac{e^{z_1}}{\sum_{i=0}^{1} e^{z_i}}$ $\hat{\boldsymbol{y}}_1 = [0.385]$

$\mathrm{L} = -y_0 \log\hat{y}_0 - y_1 \log\hat{y}_1$

y $\boldsymbol{y} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

$L = [-\log 0.615] = [0.486]$

Softmax Regression - Naïve

Derivative

$$\frac{\partial L}{\partial z_i} = \hat{y}_i - y_i$$

$$\frac{\partial L}{\partial w_i} = x(\hat{y}_i - y_i)$$

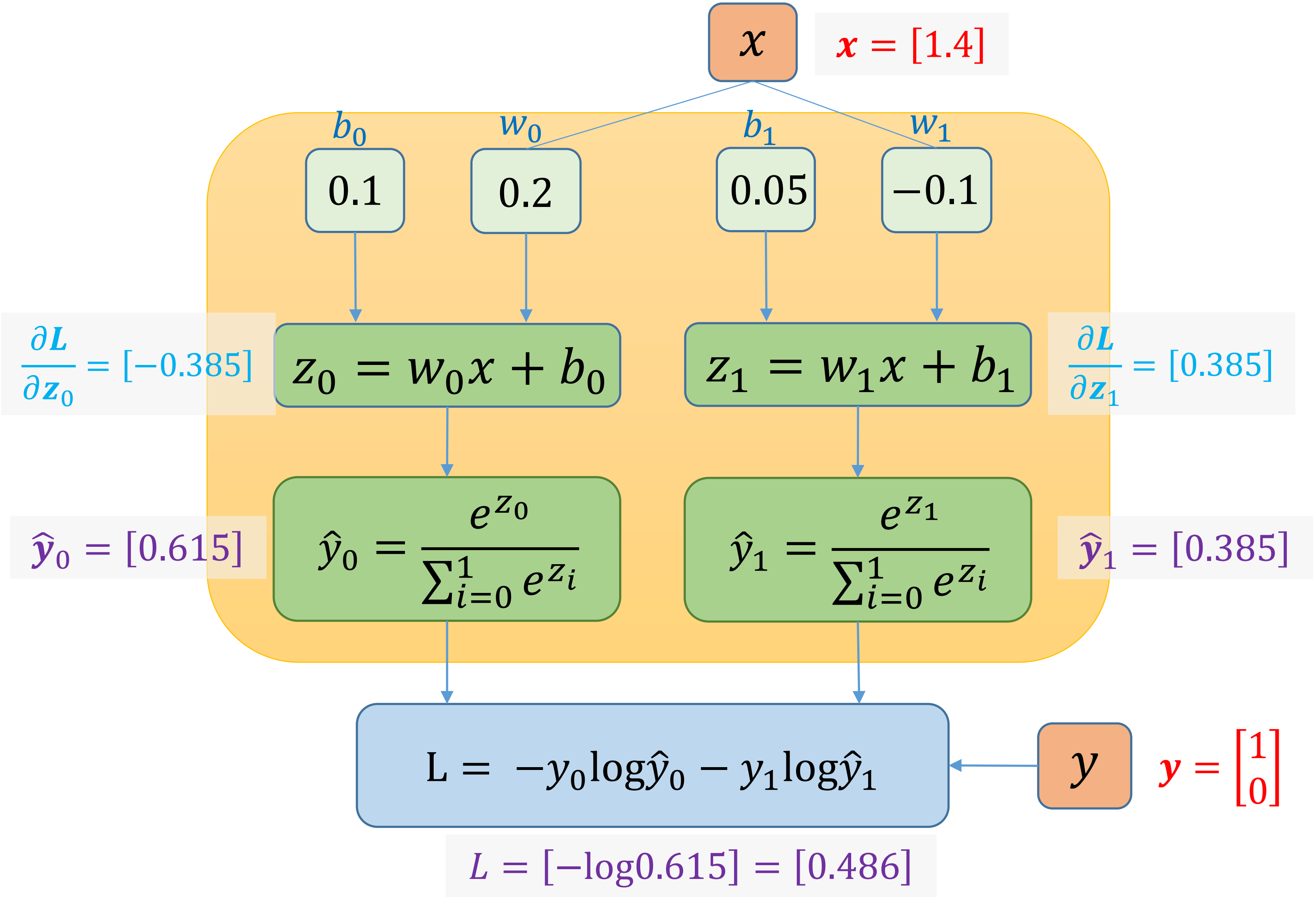
$$\frac{\partial L}{\partial b_i} = \hat{y}_i - y_i$$

$$\begin{array}{l} \\ y = 0 \rightarrow \boldsymbol{y}^T = \\ y = 1 \rightarrow \boldsymbol{y}^T = \end{array} \begin{array}{cc} y_0 & y_1 \\ [1 & 0] \\ [0 & 1] \end{array}$$

$$\frac{\partial L}{\partial \boldsymbol{z}_0} = \hat{y}_0 - 1$$

$$= 0.615 - 1 = -0.385$$

$$\frac{\partial L}{\partial \boldsymbol{z}_1} = \hat{y}_1 - 0 = 0.385$$



AI VIETNAM
All-in-One Course

Softmax Regression - Naïve

Derivative

$$\frac{\partial L}{\partial z_i} = \hat{y}_i - y_i$$

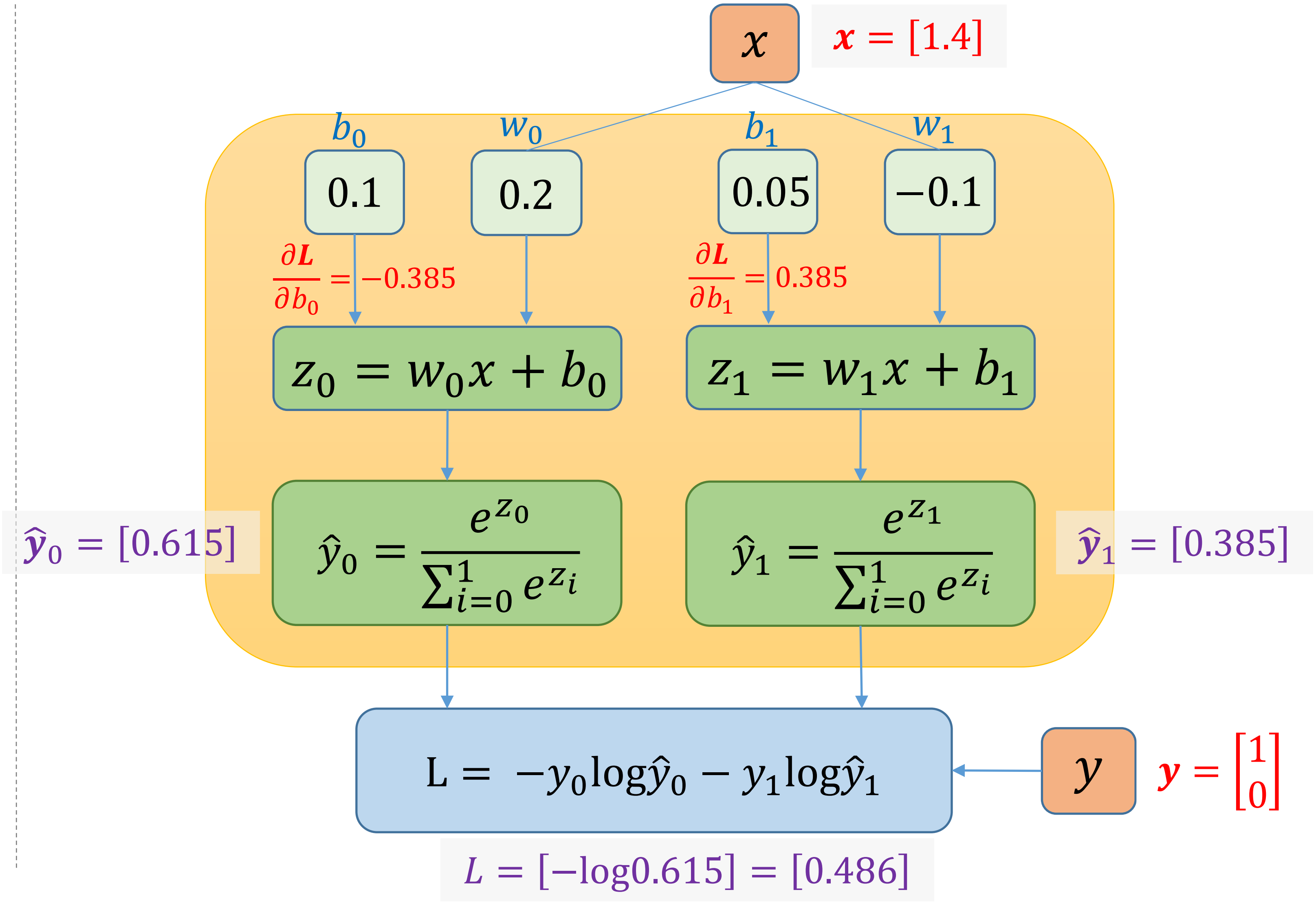
$$\frac{\partial L}{\partial w_i} = x(\hat{y}_i - y_i)$$

$$\frac{\partial L}{\partial b_i} = \hat{y}_i - y_i$$

$$\begin{array}{l} \\ y = 0 \rightarrow \boldsymbol{y}^T = \\ y = 1 \rightarrow \boldsymbol{y}^T = \end{array} \begin{array}{cc} y_0 & y_1 \\ [1 & 0] \\ [0 & 1] \end{array}$$

$$\frac{\partial \boldsymbol{L}}{\partial b_0} = (\hat{y}_0 - 1\ ) = -0.385$$

$$\frac{\partial \boldsymbol{L}}{\partial b_1} = (\hat{y}_1 - 0\ ) = 0.385$$



Softmax Regression - Naïve

Derivative

$$\frac{\partial L}{\partial z_i} = \hat{y}_i - y_i$$

$$\frac{\partial L}{\partial w_i} = x(\hat{y}_i - y_i)$$

$$\frac{\partial L}{\partial b_i} = \hat{y}_i - y_i$$

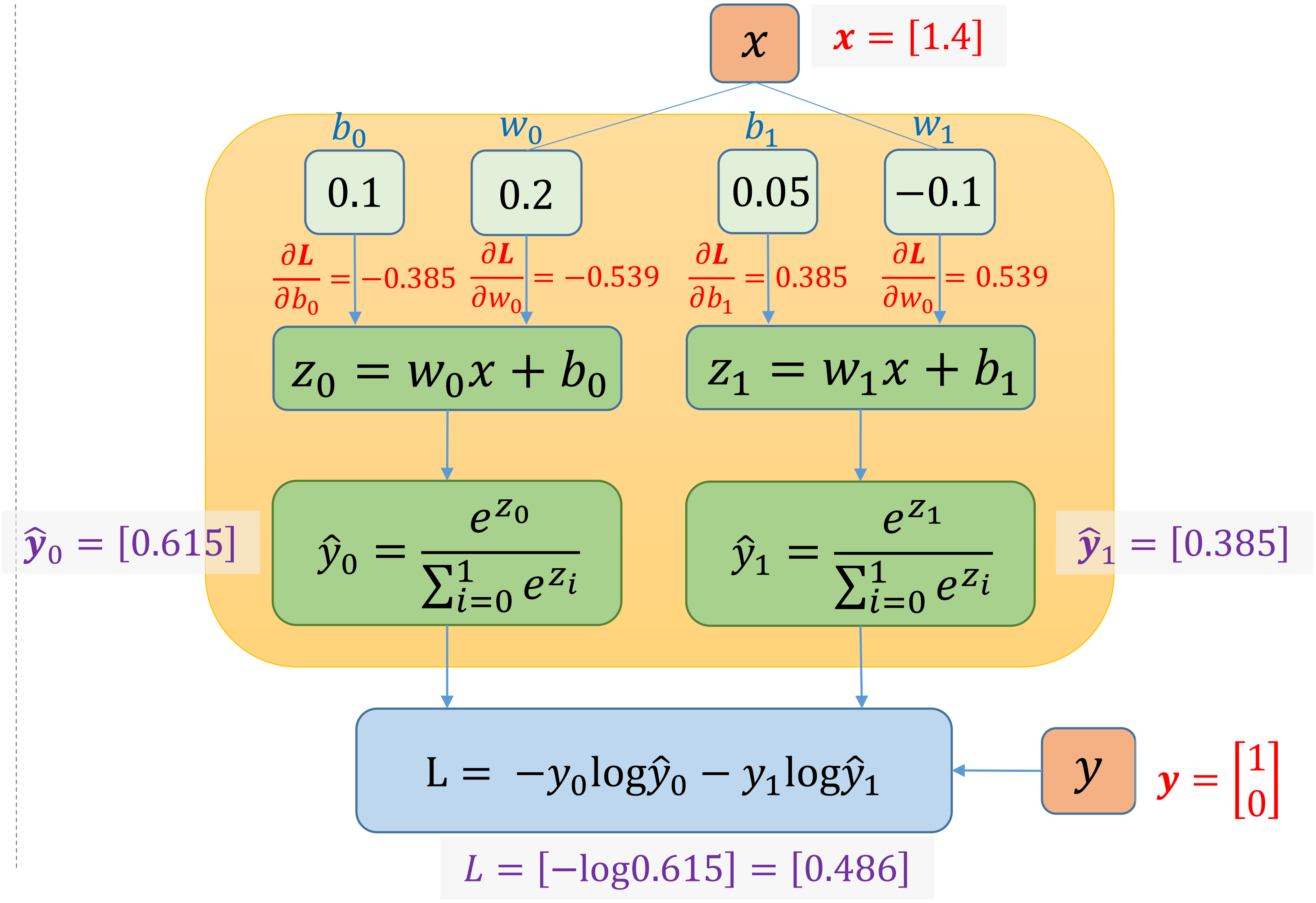
$$\begin{array}{ccc} & y_0 & y_1 \\ y = 0 \rightarrow \boldsymbol{y}^T = & [1 & 0] \\ y = 1 \rightarrow \boldsymbol{y}^T = & [0 & 1] \end{array}$$

$$\frac{\partial \boldsymbol{L}}{\partial w_0} = x(\hat{y}_0 - 1\ )$$

$$= -0.385*1.4 = -0.539$$

$$\frac{\partial \boldsymbol{L}}{\partial w_1} = x(\hat{y}_1 - 0\ )$$

$$= 0.385*1.4 = 0.539$$



Softmax Regression - Naïve

Update parameters

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

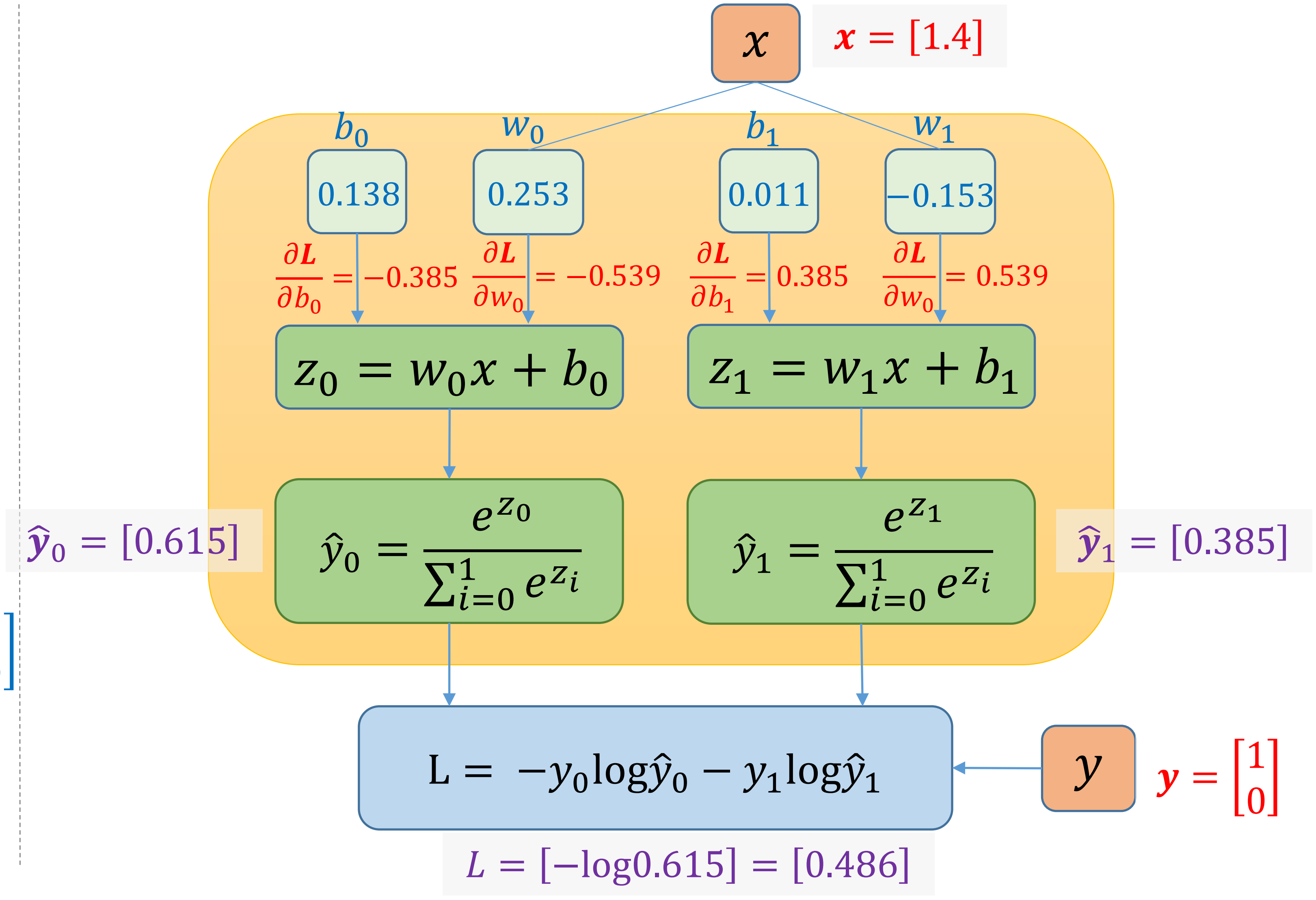
η is learning rate

$$\boldsymbol{\theta} = \begin{bmatrix} b_0 & b_1 \\ w_0 & w_1 \end{bmatrix} \qquad L'_{\boldsymbol{\theta}} = \begin{bmatrix} \frac{\partial L}{\partial b_0} & \frac{\partial L}{\partial b_1} \\ \frac{\partial L}{\partial w_0} & \frac{\partial L}{\partial w_1} \end{bmatrix}$$

$$\eta = 0.1$$

$$\boldsymbol{\theta} = \begin{bmatrix} 0.1 & 0.05 \\ 0.2 & -0.1 \end{bmatrix} - 0.1 \begin{bmatrix} -0.385 & 0.385 \\ -0.539 & 0.539 \end{bmatrix}$$

$$= \begin{bmatrix} 0.138 & 0.011 \\ 0.253 & -0.153 \end{bmatrix}$$



Softmax Regression - Naïve

Training data

Feature **Label**

Petal_Length	Label
1.4	0
1.3	0
1.5	0
4.5	1
4.1	1
4.6	1

One-hot encoding for label

$$\begin{array}{ccc} & y_0 & y_1 \end{array}$$

$$y = 0 \rightarrow \boldsymbol{y}^T = [1 \quad 0]$$

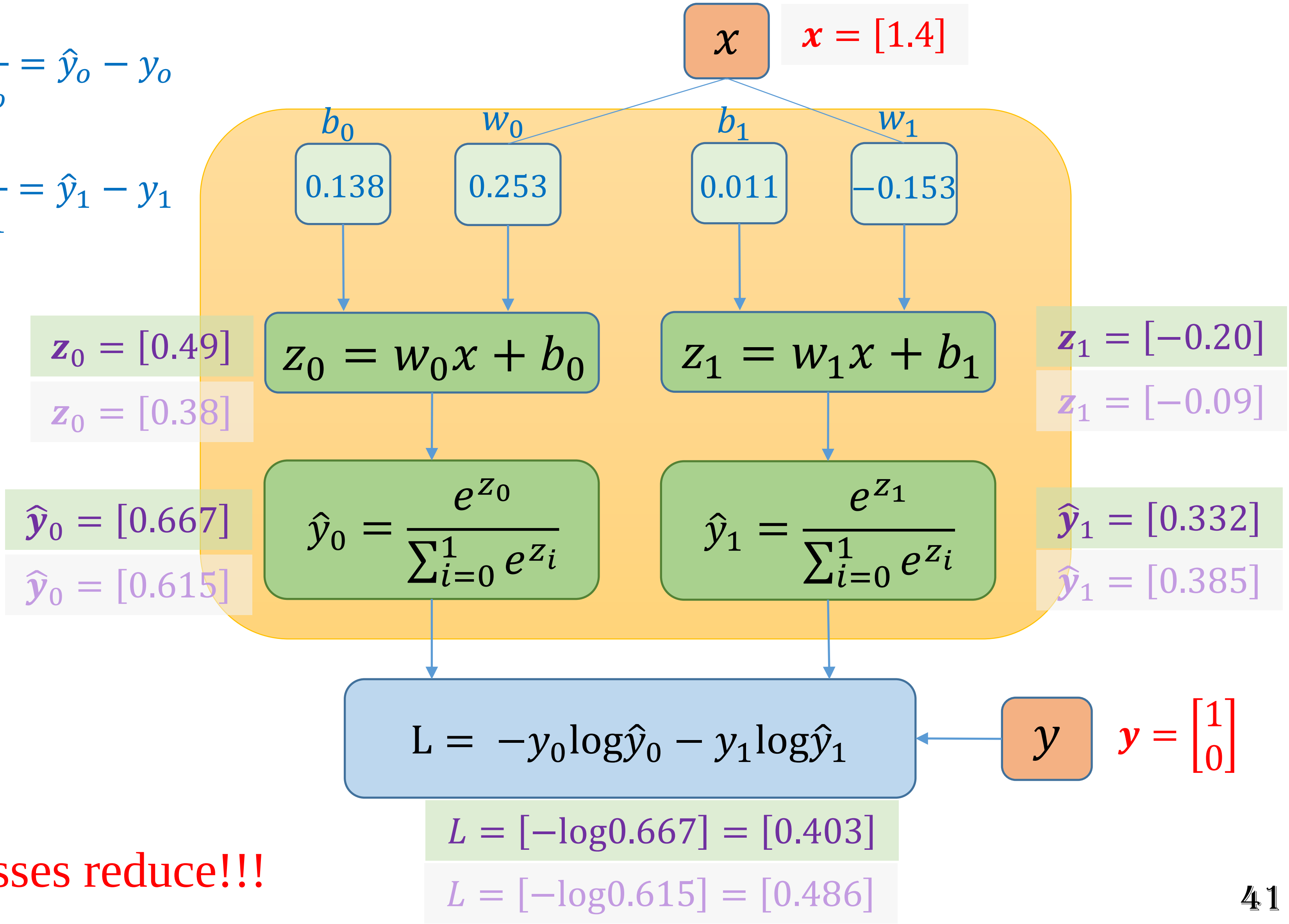
$$y = 1 \rightarrow \boldsymbol{y}^T = [0 \quad 1]$$

Training example

$$(x, y) = (1.4, 0)$$

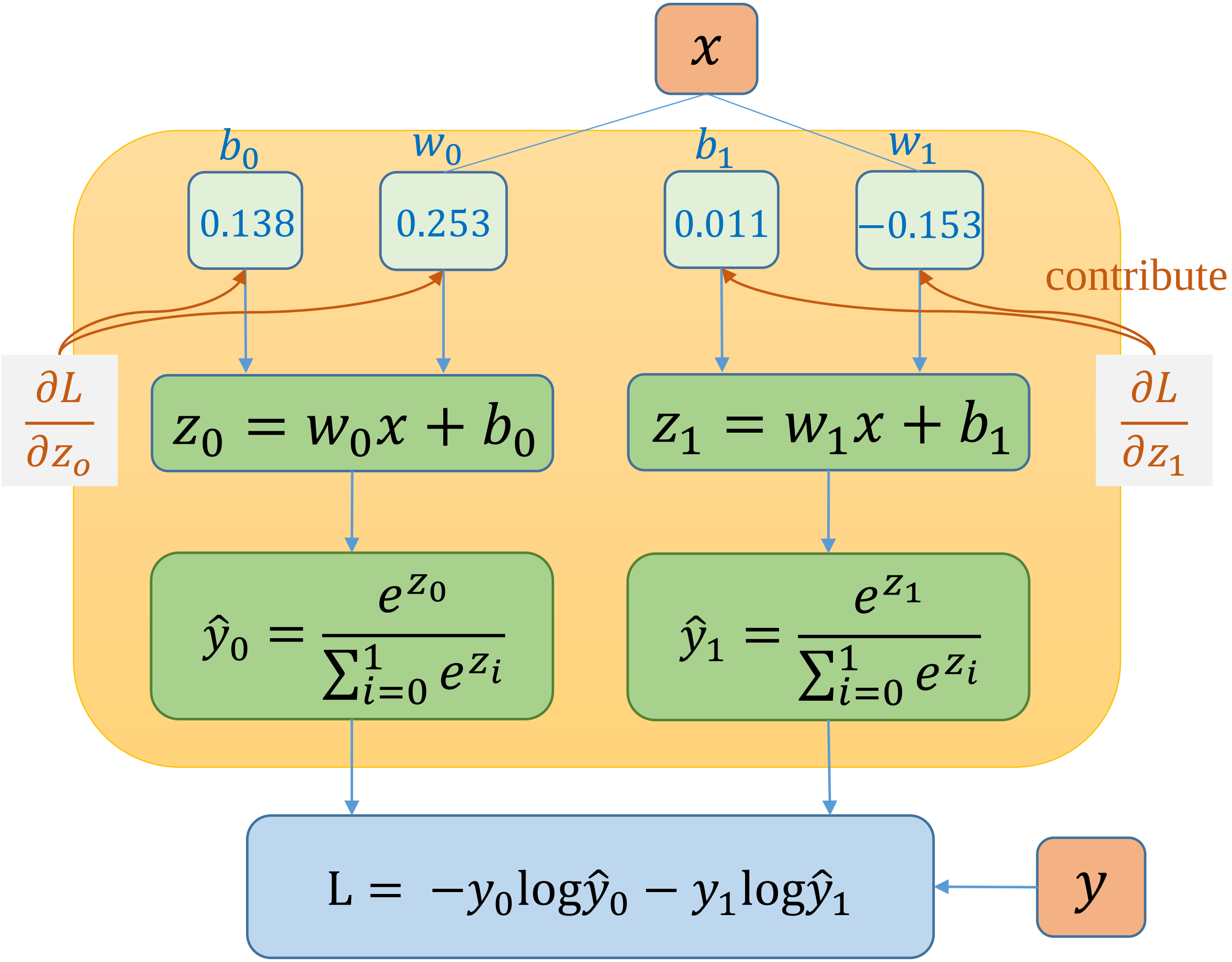
$$\frac{\partial L}{\partial z_o} = \hat{y}_o - y_o$$

$$\frac{\partial L}{\partial z_1} = \hat{y}_1 - y_1$$



losses reduce!!!

Softmax Regression - Naïve



$$\frac{\partial L}{\partial z_o} = \hat{y}_o - y_o \qquad \frac{\partial L}{\partial z_1} = \hat{y}_1 - y_1$$

$$\frac{\partial L}{\partial w_0} = x(\hat{y}_0 - y_0) \qquad \frac{\partial L}{\partial w_1} = x(\hat{y}_1 - y_1)$$

$$\frac{\partial L}{\partial b_0} = \hat{y}_0 - y_0 \qquad \frac{\partial L}{\partial b_1} = \hat{y}_1 - y_1$$

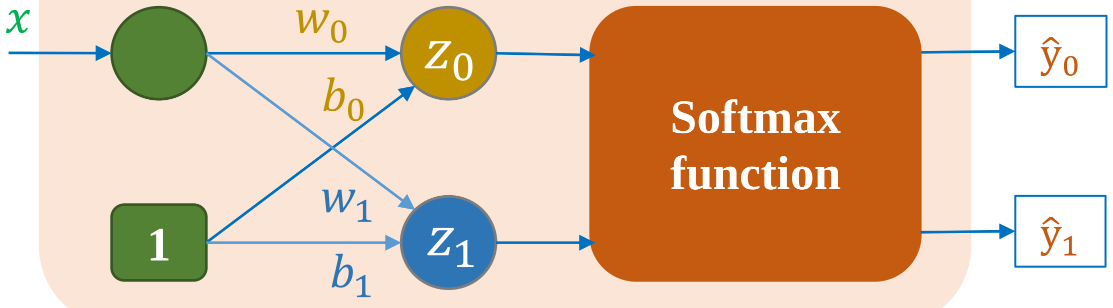
Softmax Regression - Naïve

$$\frac{\partial L}{\partial z_o} = \hat{y}_o - y_o \qquad \frac{\partial L}{\partial z_1} = \hat{y}_1 - y_1$$

$$\frac{\partial L}{\partial w_0} = x(\hat{y}_0 - y_0) \qquad \frac{\partial L}{\partial w_1} = x(\hat{y}_1 - y_1)$$

$$\frac{\partial L}{\partial b_0} = \hat{y}_0 - y_0 \qquad \frac{\partial L}{\partial b_1} = \hat{y}_1 - y_1$$

$$\boldsymbol{\theta} = [\boldsymbol{\theta}_o \quad \boldsymbol{\theta}_1] = \begin{bmatrix} b_0 & b_1 \\ w_0 & w_1 \end{bmatrix}$$



$$\nabla_{\boldsymbol{\theta}} L = \boldsymbol{x}(\hat{\mathbf{y}} - \boldsymbol{y})^T$$

$$\boldsymbol{x} = \begin{bmatrix} 1 \\ x \end{bmatrix}$$

$$L'_{\boldsymbol{\theta}} = \begin{bmatrix} \frac{\partial L}{\partial b_0} & \frac{\partial L}{\partial b_1} \\ \frac{\partial L}{\partial w_0} & \frac{\partial L}{w_1} \end{bmatrix} = \begin{bmatrix} 1\frac{\partial L}{\partial z_0} & 1\frac{\partial L}{\partial z_1} \\ x\frac{\partial L}{\partial z_o} & x\frac{\partial L}{\partial z_1} \end{bmatrix}$$

$$L'_{\boldsymbol{z}} = \begin{bmatrix} \frac{\partial L}{\partial z_o} \\ \frac{\partial L}{\partial z_1} \end{bmatrix}$$

$$L'_{\boldsymbol{\theta}} = \begin{bmatrix} \frac{\partial L}{\partial b_0} & \frac{\partial L}{\partial b_1} \\ \frac{\partial L}{\partial w_0} & \frac{\partial L}{w_1} \end{bmatrix} = \begin{bmatrix} 1\frac{\partial L}{\partial z_0} & 1\frac{\partial L}{\partial z_1} \\ x\frac{\partial L}{\partial z_o} & x\frac{\partial L}{\partial z_1} \end{bmatrix}$$

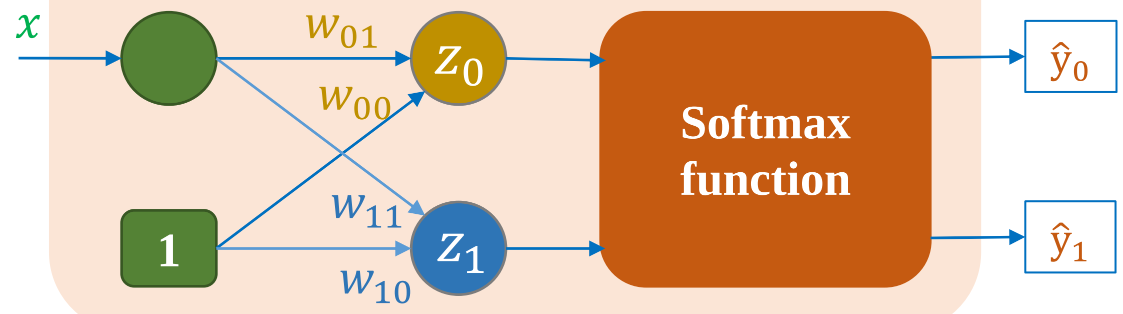
Softmax Regression - Vectorization

$$\frac{\partial L}{\partial z_o} = \hat{y}_o - y_o \qquad \frac{\partial L}{\partial z_1} = \hat{y}_1 - y_1$$

$$\frac{\partial L}{\partial w_0} = x\frac{\partial L}{\partial z_o} \qquad \frac{\partial L}{\partial w_1} = x\frac{\partial L}{\partial z_1}$$

$$\frac{\partial L}{\partial b_0} = \frac{\partial L}{\partial z_o} \qquad \frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial z_1}$$

$$\boldsymbol{\theta} = [\boldsymbol{\theta}_o \quad \boldsymbol{\theta}_1] = \begin{bmatrix} w_{00} & w_{10} \\ w_{01} & w_{11} \end{bmatrix}$$



$$\nabla_{\boldsymbol{\theta}} L = \boldsymbol{x}(\hat{\mathbf{y}} - \boldsymbol{y})^T$$

$$\boldsymbol{x} = \begin{bmatrix} \boxed{x_0} \\ \boxed{x_1} \end{bmatrix}$$

$$L'_{\boldsymbol{\theta}} = \begin{bmatrix} \dfrac{\partial L}{\partial w_{00}} & \dfrac{\partial L}{\partial w_{10}} \\ \dfrac{\partial L}{\partial w_{01}} & \dfrac{\partial L}{\partial w_{11}} \end{bmatrix} = \begin{bmatrix} \boxed{x_0}\dfrac{\partial L}{\partial z_o} & \boxed{x_0}\dfrac{\partial L}{\partial z_1} \\ \boxed{x_1}\dfrac{\partial L}{\partial z_o} & \boxed{x_1}\dfrac{\partial L}{\partial z_1} \end{bmatrix}$$

$$L'_{\boldsymbol{z}} = \begin{bmatrix} \boxed{\dfrac{\partial L}{\partial z_o}} \\ \boxed{\dfrac{\partial L}{\partial z_1}} \end{bmatrix}$$

$$L'_{\boldsymbol{\theta}} = \begin{bmatrix} \dfrac{\partial L}{\partial w_{00}} & \dfrac{\partial L}{\partial w_{10}} \\ \dfrac{\partial L}{\partial w_{01}} & \dfrac{\partial L}{\partial w_{11}} \end{bmatrix} = \begin{bmatrix} x_0\boxed{\dfrac{\partial L}{\partial z_o}} & x_0\boxed{\dfrac{\partial L}{\partial z_1}} \\ x_1\boxed{\dfrac{\partial L}{\partial z_o}} & x_1\boxed{\dfrac{\partial L}{\partial z_1}} \end{bmatrix}$$

Softmax Regression - Vectorization

1) Pick a sample from training data

2) Tính output $\hat{y}$

$$\boldsymbol{z} = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$\boldsymbol{d} = [1 \ \dots 1] e^{\boldsymbol{z}}$$

$$\hat{\boldsymbol{y}} = e^{\boldsymbol{z}} \oslash \boldsymbol{d}$$

$\oslash$ is Hadamard division

$$\hat{\boldsymbol{y}} = \frac{e^{\boldsymbol{z}}}{\sum_j e^{x_j}}$$

3) Tính loss (cross-entropy)

$$L(\boldsymbol{\theta}) = -\boldsymbol{y}^T log \hat{\boldsymbol{y}}$$

4) Tính đạo hàm

$$\nabla_{\boldsymbol{\theta}} L = \boldsymbol{x}(\hat{\boldsymbol{y}} - \boldsymbol{y})^T$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta L'_{\boldsymbol{\theta}}$$

η is learning rate

$$\boldsymbol{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} \qquad \boldsymbol{\theta} = [\boldsymbol{\theta}_o \quad \boldsymbol{\theta}_1] = \begin{bmatrix} w_{00} & w_{10} \\ w_{01} & w_{11} \end{bmatrix}$$

$$\hat{\boldsymbol{y}} = \begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \end{bmatrix}$$

$$y = 0 \rightarrow \boldsymbol{y}^T = [1 \quad 0] \qquad L(\boldsymbol{\theta}) = -\boldsymbol{y}^T log \hat{\boldsymbol{y}} = -log \hat{\boldsymbol{y}}_0$$

$$y = 1 \rightarrow \boldsymbol{y}^T = [0 \quad 1] \qquad L(\boldsymbol{\theta}) = -\boldsymbol{y}^T log \hat{\boldsymbol{y}} = -log \hat{\boldsymbol{y}}_1$$

$$L'_{\boldsymbol{\theta}} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} \begin{bmatrix} \frac{\partial L}{\partial z_0} & \frac{\partial L}{\partial z_1} \end{bmatrix} = \begin{bmatrix} x_0 \frac{\partial L}{\partial z_0} & x_0 \frac{\partial L}{\partial z_1} \\ x_1 \frac{\partial L}{\partial z_o} & x_1 \frac{\partial L}{\partial z_1} \end{bmatrix}$$

$$\boldsymbol{\theta} = \begin{bmatrix} w_{00} & w_{10} \\ w_{01} & w_{11} \end{bmatrix} - \eta \begin{bmatrix} x_0 \frac{\partial L}{\partial z_0} & x_0 \frac{\partial L}{\partial z_1} \\ x_1 \frac{\partial L}{\partial z_o} & x_1 \frac{\partial L}{\partial z_1} \end{bmatrix}$$

Outline

- Motivation
- Model Construction
- Loss Function
- Generalization (Further Reading)
- Another Approach (Further Reading)

Softmax Regression - Batch

1) Pick N samples from training data

2) Tính output $\hat{y}$

$$\boldsymbol{z} = \boldsymbol{x\theta}$$

$$\boldsymbol{d} = e^{\boldsymbol{z}}\mathbf{1}$$

$$\hat{\boldsymbol{y}} = (\mathbf{1} \oslash \boldsymbol{d})e^{\boldsymbol{z}}$$

$\oslash$ is Hadamard division

3) Tính loss (cross-entropy)

$$L(\boldsymbol{\theta}) = \mathbf{1}(-(\boldsymbol{y} \odot log\hat{\boldsymbol{y}})\mathbf{1})$$

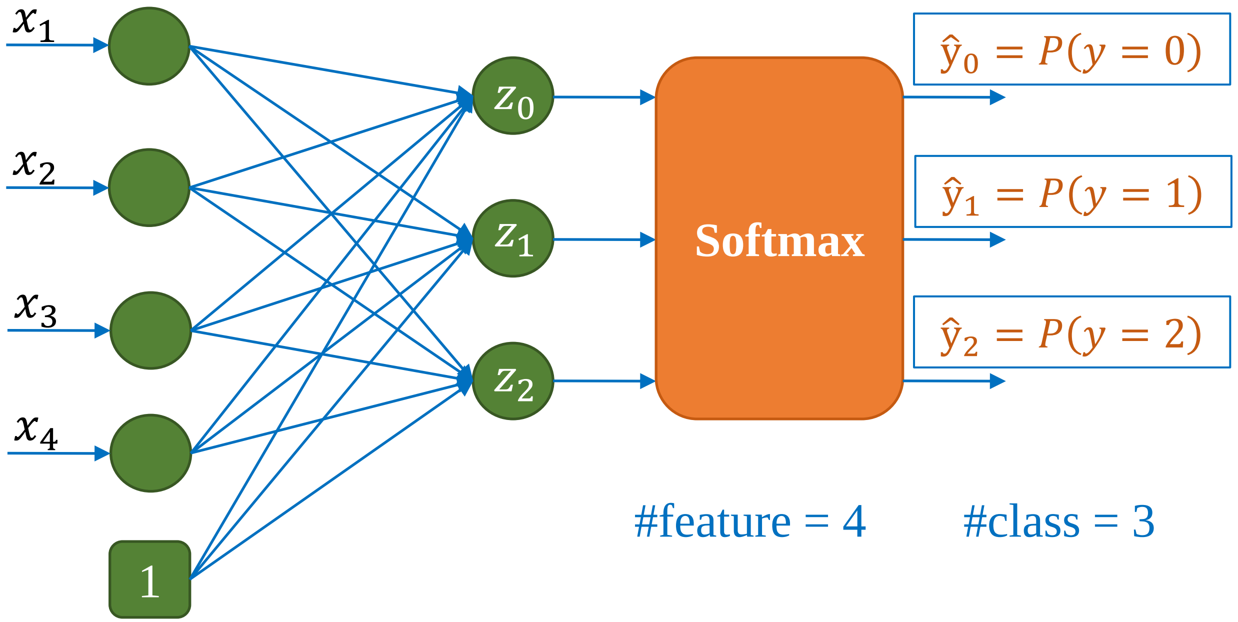
4) Tính đạo hàm

$$L'_{\boldsymbol{\theta}} = \boldsymbol{x}^T(\hat{\boldsymbol{y}} - \boldsymbol{y})$$

5) Cập nhật tham số

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{L'_{\boldsymbol{\theta}}}{N}$$

η is learning rate



#feature = 4 #class = 3

$$\boldsymbol{y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\boldsymbol{x} = \begin{bmatrix} x_0^{(1)} & x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & x_4^{(1)} \\ x_0^{(2)} & x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & x_4^{(2)} \end{bmatrix}$$

$$\boldsymbol{\theta} = [\boldsymbol{\theta}_0 \quad \boldsymbol{\theta}_1 \quad \boldsymbol{\theta}_2] = \begin{bmatrix} w_{00} & w_{10} & w_{20} \\ w_{01} & w_{11} & w_{21} \\ w_{02} & w_{12} & w_{22} \\ w_{03} & w_{13} & w_{23} \\ w_{04} & w_{14} & w_{24} \end{bmatrix}$$

Friday - Pytorch

PyTorch

```python
import numpy as np
import torch
import torch.nn as nn
import torch.optim as optim

# Load data
iris = np.genfromtxt('iris_1D_2c.csv', dtype=None,
                     delimiter=',', skip_header=1)
X = torch.tensor(iris[:, 0:1], dtype=torch.float32)
y = torch.tensor(iris[:, 1], dtype=torch.int64)
```

```python
# create model
input_dim = X.shape[1]
output_dim = len(torch.unique(y))
model = nn.Linear(input_dim, output_dim)

# Loss and optimizer
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(model.parameters(), lr=0.1)

# Training Loop
max_epoch = 100
for epoch in range(max_epoch):
    # Zero the gradients
    optimizer.zero_grad()

    # Forward pass
    outputs = model(X)
    loss = criterion(outputs, y)

    # Backward pass
    loss.backward()
    optimizer.step()
```