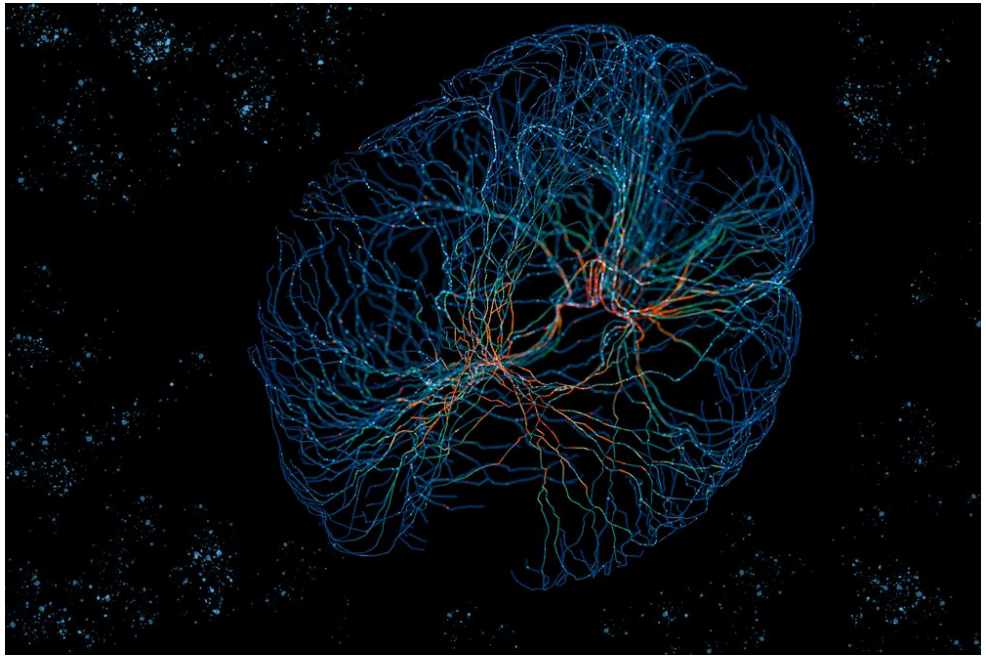
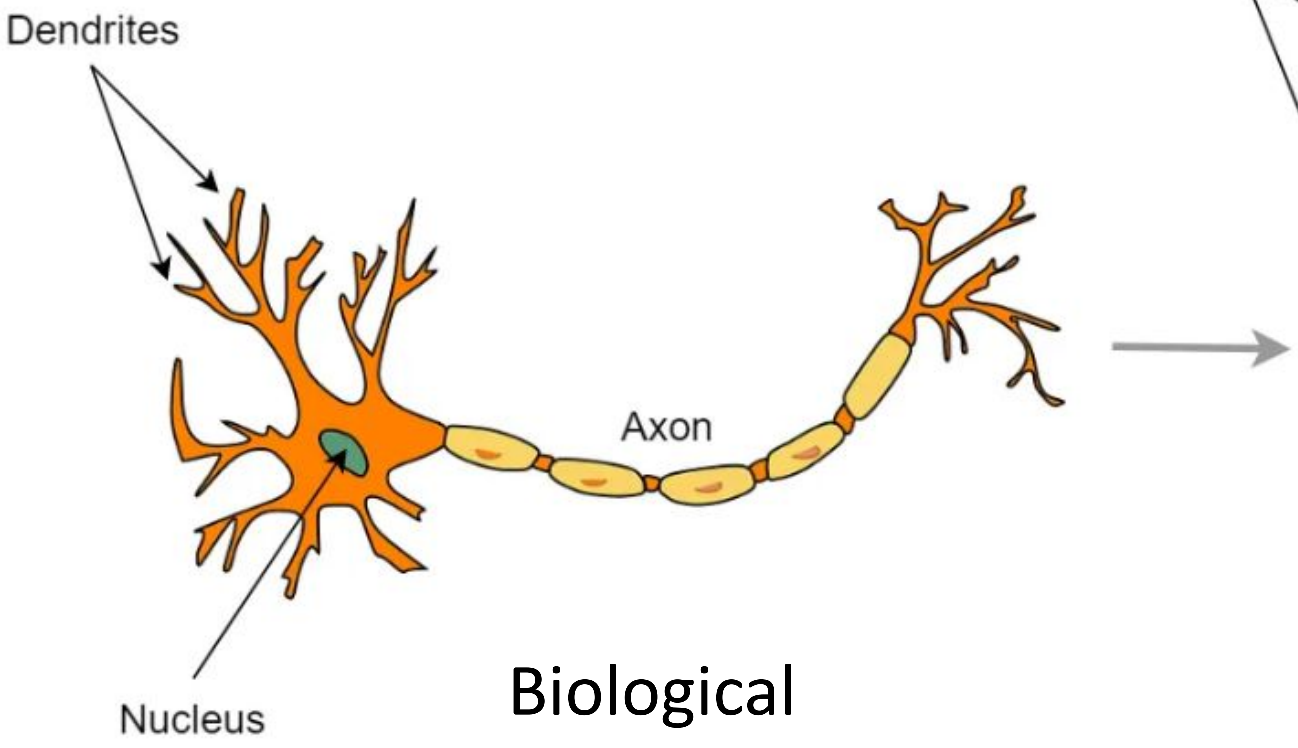
Basics of Neural Networks

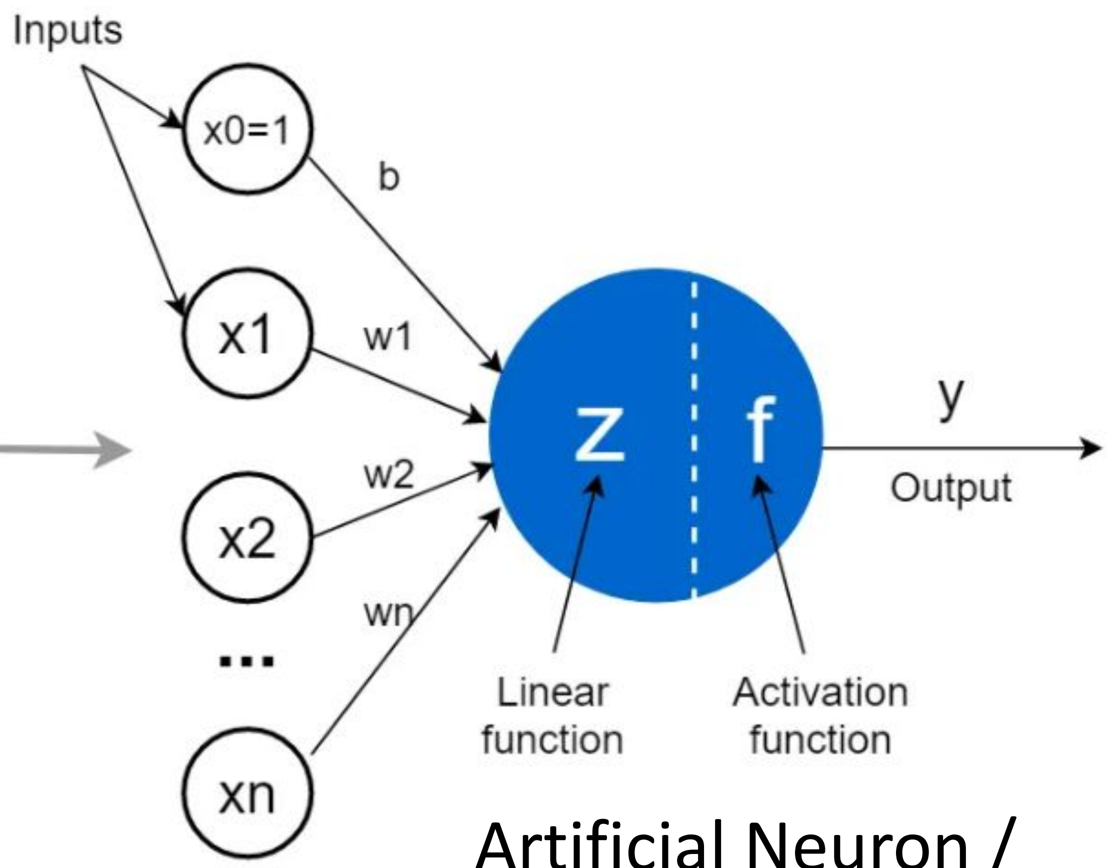


Basics of Neural Networks

Deep Learning



Biological Neuron



Artificial Neuron / Perceptron

Recap:

Logistic Regression!

Will it Rain?

x_i = features for day i

features

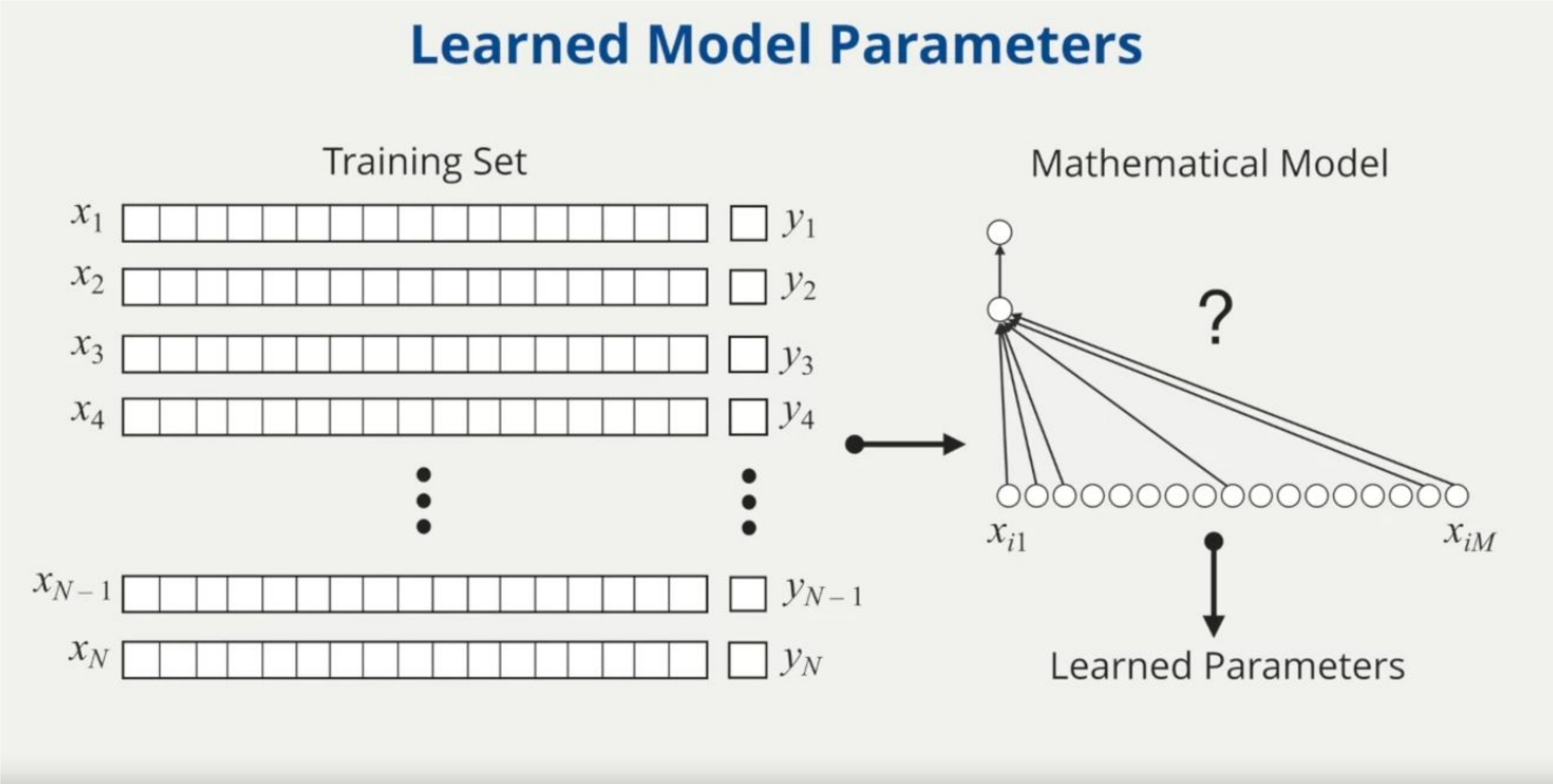
Cloud Cover	Humidity	Temperature	Air Pressure
0.5	80%	75	1.2

outcome

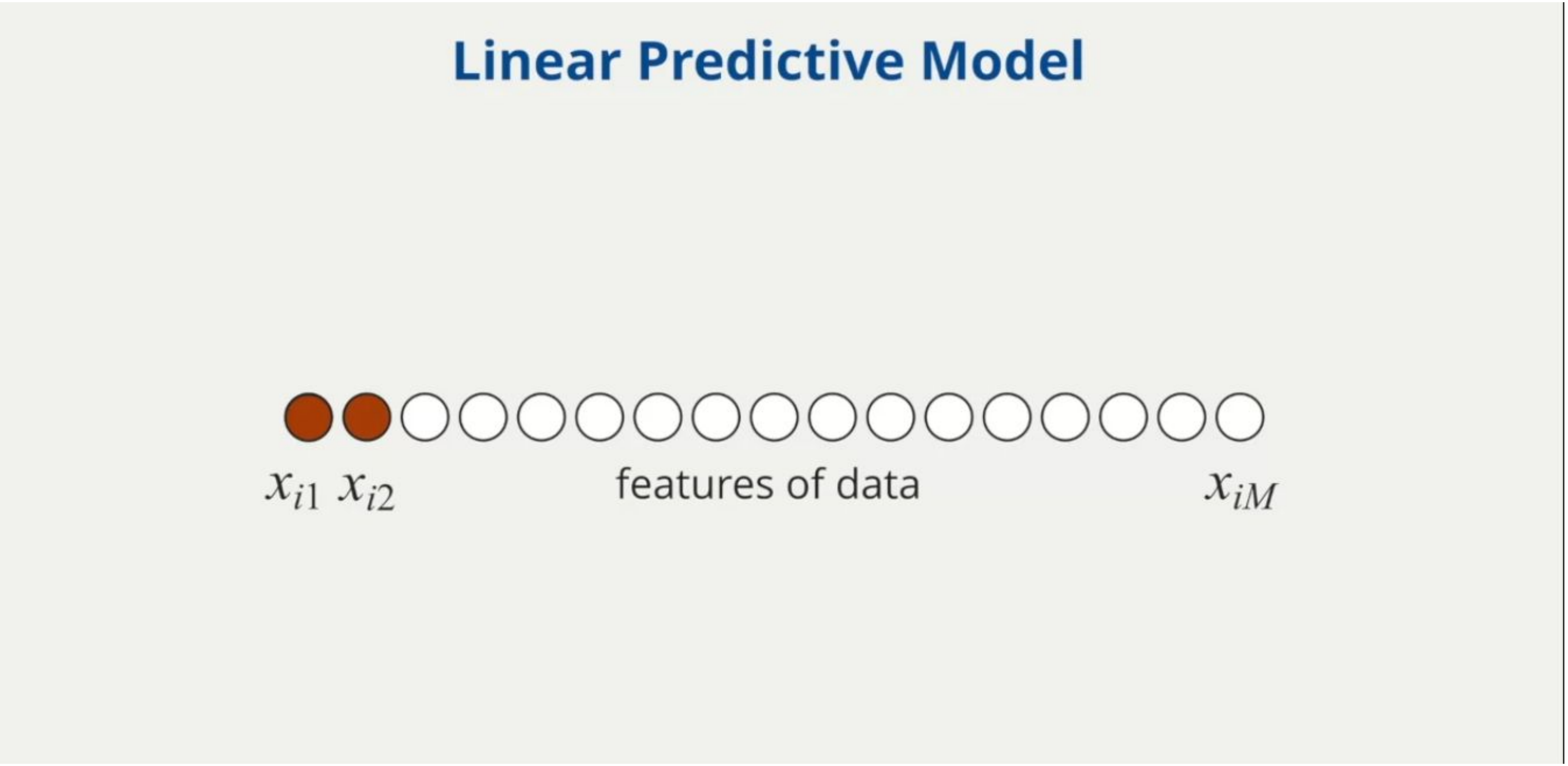
Did it Rain
1

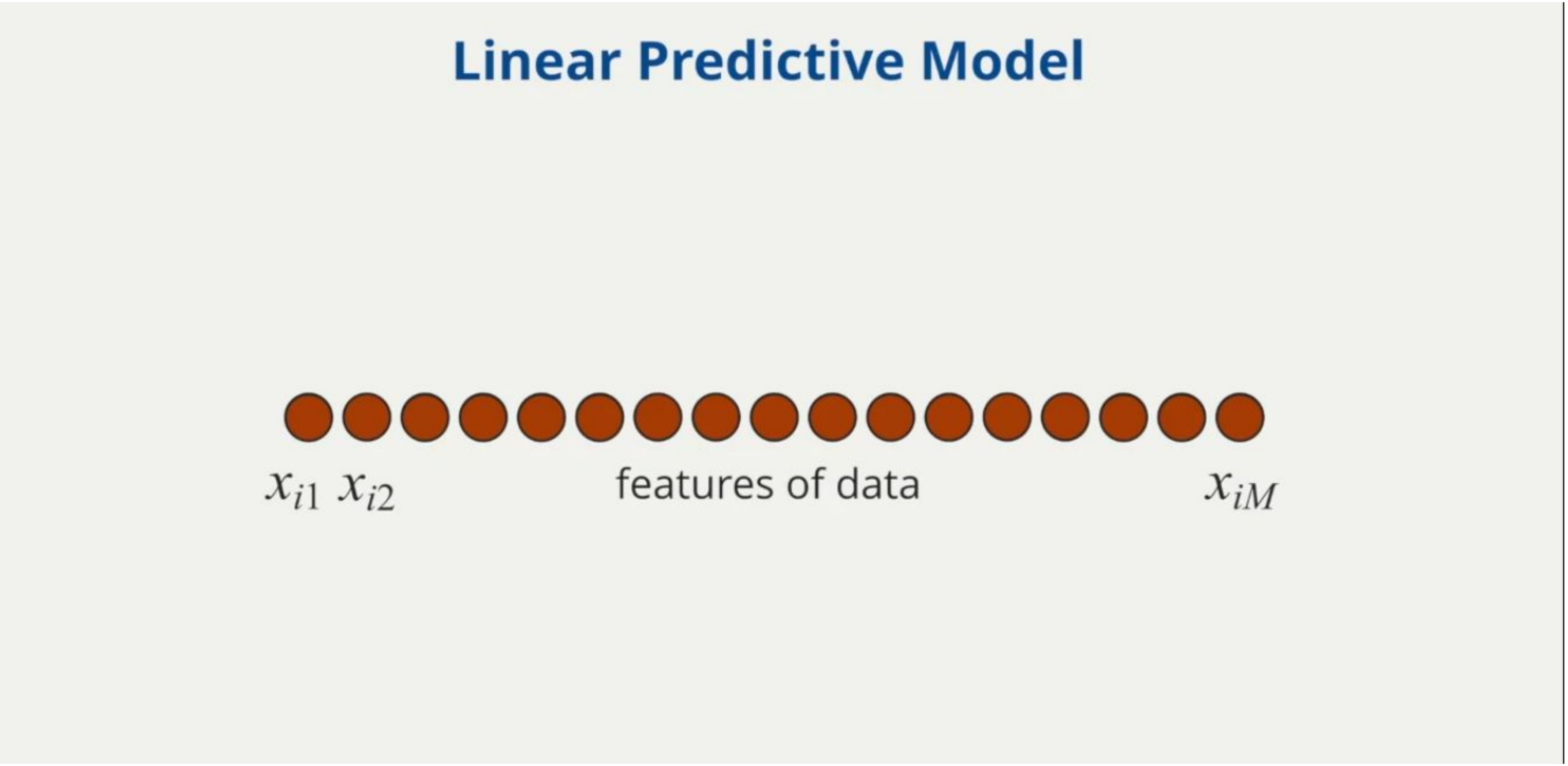
y_i = 1, rains

y_i = 0, no rain


Learned Model Parameters
Training Set
Mathematical Model
x_1
x_2
x_3
x_4
x_{N-1}
x_N
y_1
y_2
y_3
y_4
y_{N-1}
y_N
?
x_{i1}
x_{iM}
Learned Parameters

Linear Predictive Model


Linear Predictive Model
x_{i1} x_{i2}
features of data
x_{iM}


Linear Predictive Model
x_{i1} x_{i2}
features of data
x_{iM}

Linear Predictive Model

$$(b_1 \times x_{i1})$$

Linear Predictive Model

$$(b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM})$$

Linear Predictive Model

$$(b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + \underbrace{b_0}_{\text{bias}}$$

x_{i1} x_{i2} features of data x_{iM}

Basics of Neural Networks

Will it Rain?

x_i = features for day i

features

Cloud Cover	Humidity	Temperature	Air Pressure
0.5	80%	75	1.2
0.2	95%	83	1.3

outcome

Did it Rain
1
0

y_i = 1, yes

y_i = 0, no

$z_1 = (b_1 \times 0.5) + (b_2 \times 0.8) + (b_3 \times 75) + (b_4 \times 1.2) + b_0 \quad y_1 = 1$

$z_2 = (b_1 \times 0.2) + (b_2 \times 0.95) + (b_3 \times 83) + (b_4 \times 1.3) + b_0 \quad y_2 = 0$

Will it Rain?

x_i = features for day i

y_i = 1, yes

y_i = 0, no

features				outcome
Cloud Cover	Humidity	Temperature	Air Pressure	Did it Rain
0.5	80%	75	1.2	1
0.2	95%	83	1.3	0

$z_1 = (b_1 \times 0.5) + (b_2 \times 0.8) + (b_3 \times 75) + (b_4 \times 1.2) + b_0 \quad y_1 = 1$

$z_2 = (b_1 \times 0.2) + (b_2 \times 0.95) + (b_3 \times 83) + (b_4 \times 1.3) + b_0 \quad y_2 = 0$

sigma

$$p(y_i = 1 | x_i) = \sigma(z_i)$$

Will it Rain?

x_i = features for day i

features

Cloud Cover	Humidity	Temperature	Air Pressure
0.5	80%	75	1.2
0.2	95%	83	1.3

outcome

Did it Rain
1
0

y_i = 1, yes

y_i = 0, no

$z_1 = (b_1 \text{ X } 0.5) + (b_2 \text{ X } 0.8) + (b_3 \text{ X } 75) + (b_4 \text{ X } 1.2) + b_0$ $\quad y_1 = 1$

$z_2 = (b_1 \text{ X } 0.2) + (b_2 \text{ X } 0.95) + (b_3 \text{ X } 83) + (b_4 \text{ X } 1.3) + b_0$ $\quad y_2 = 0$

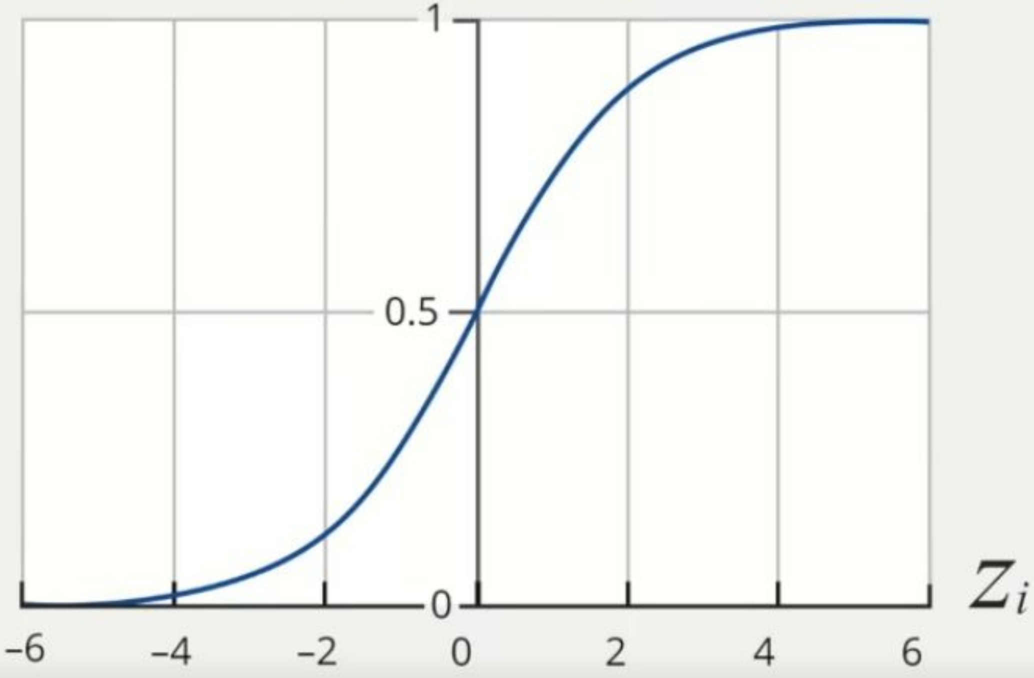
sigma

$$p(y_i = 1 | x_i) = \sigma(z_i)$$

Convert to a Probability

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

$$p(y_i = 1|x_i) = \sigma(z_i)$$

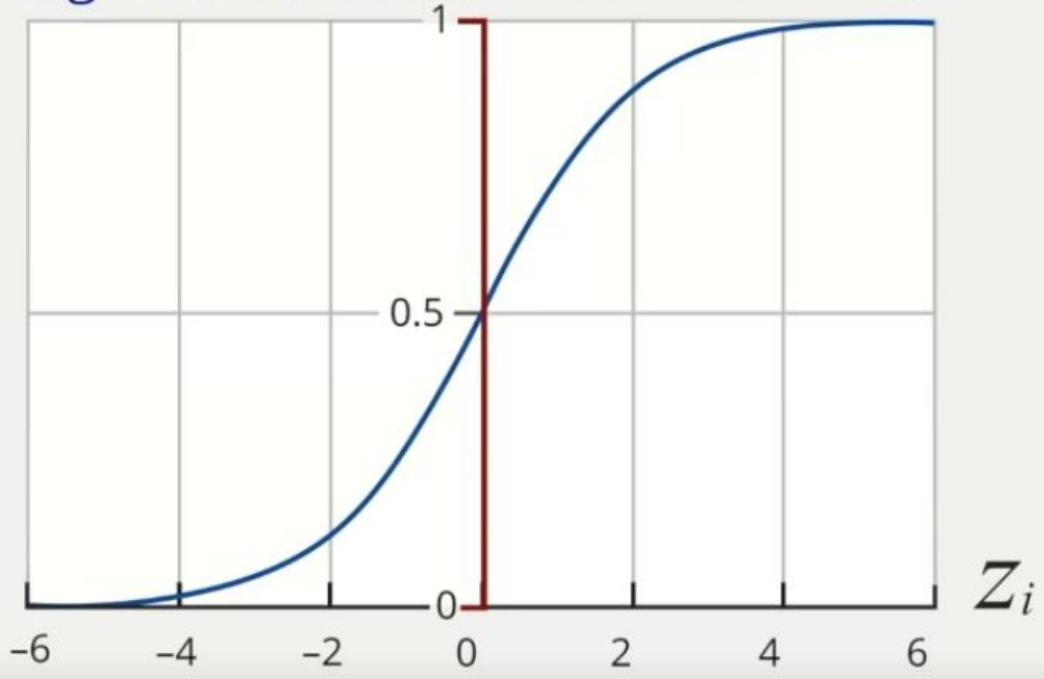


Basics of Neural Networks

Convert to a Probability

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

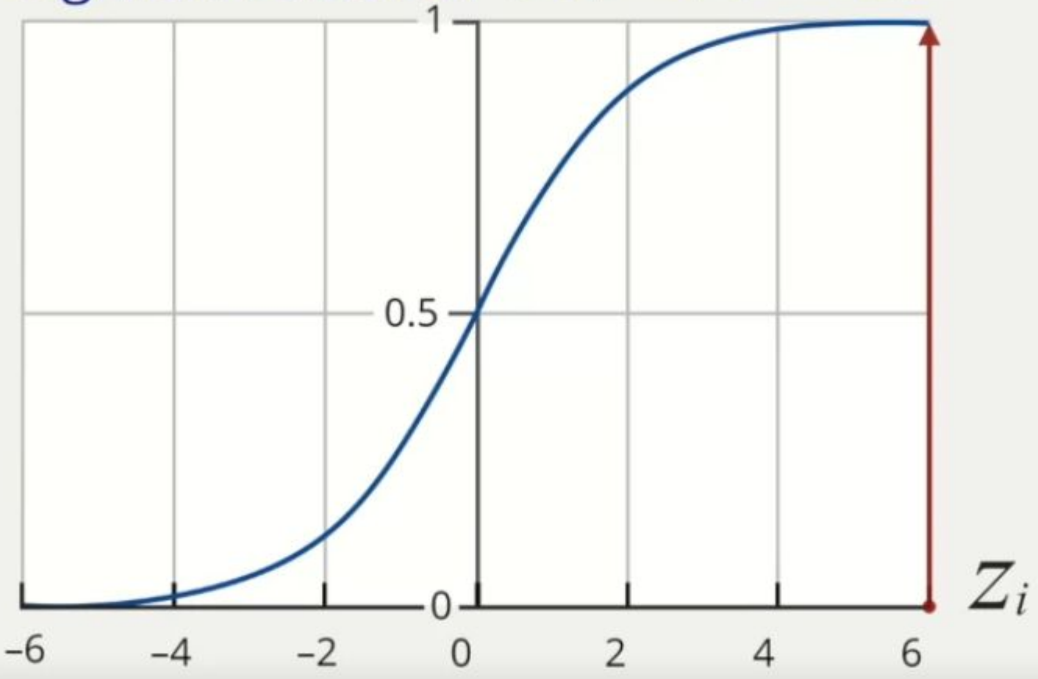
Sigmoid Function $p(y_i = 1|x_i) = \sigma(z_i)$



Convert to a Probability

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

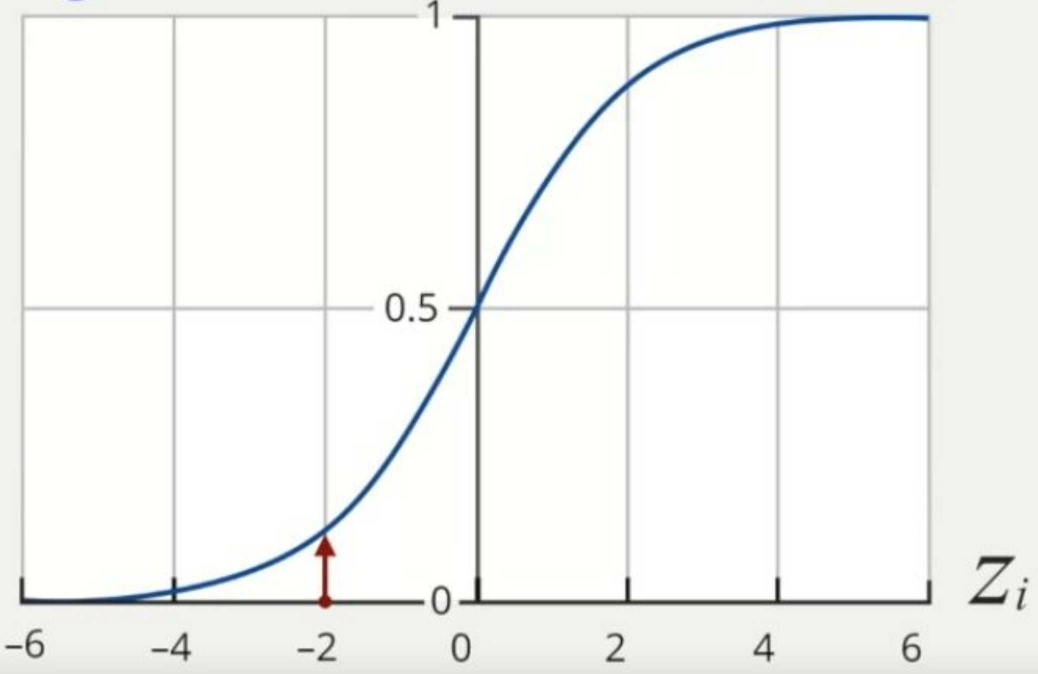
Sigmoid Function $p(y_i = 1|x_i) = \sigma(z_i)$



Convert to a Probability

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

Sigmoid Function $p(y_i = 1|x_i) = \sigma(z_i)$



Basics of Neural Networks

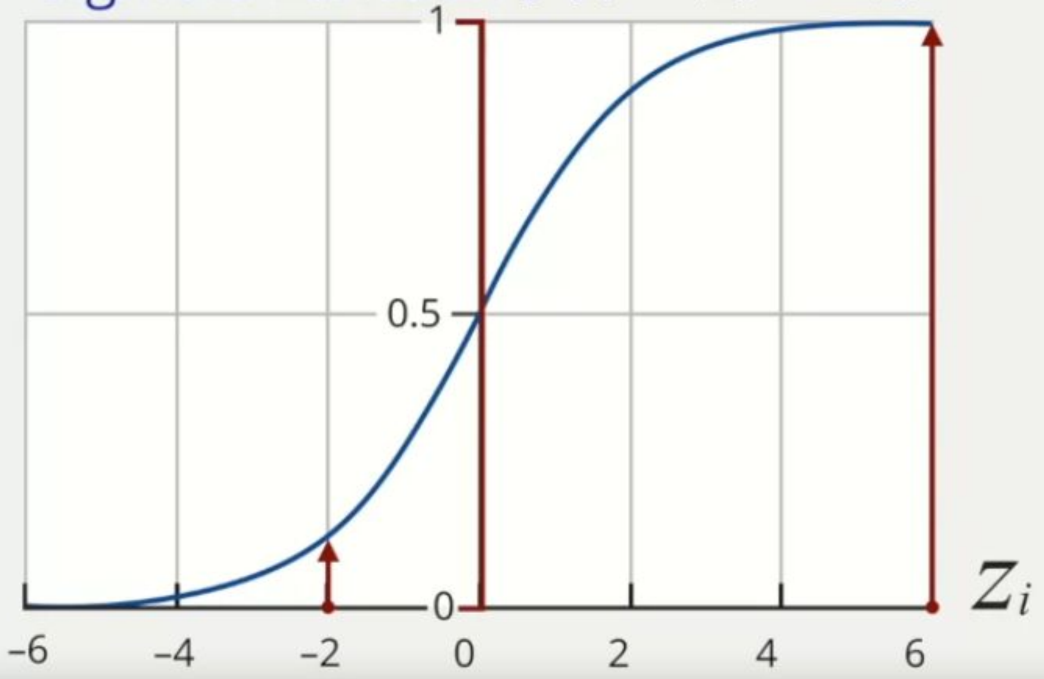
Sigmoid Function is a way to convert predictions to a probabilistic perspective

$$f(x) = \frac{1}{1 + e^{-(x)}}$$

Convert to a Probability

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

Sigmoid Function $p(y_i = 1|x_i) = \sigma(z_i)$



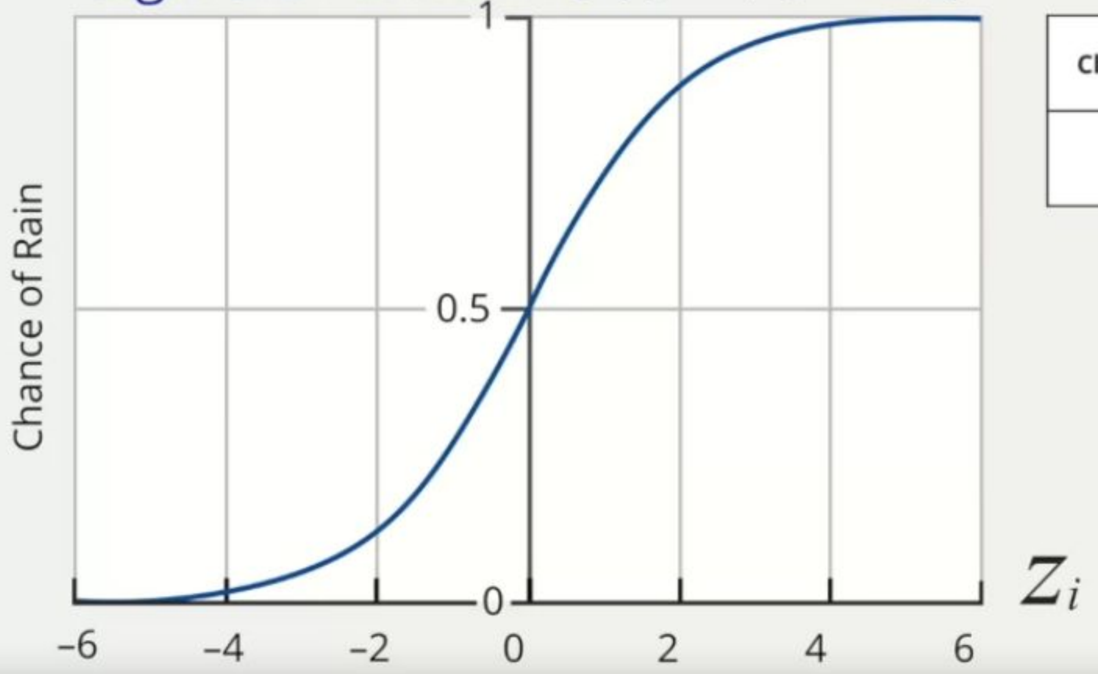
Outcome of Z

- Z_i = Large and positive indicates $y_i = 1$ is likely
- Z_i = Large and negative indicates $y_i = 0$ is likely

Convert to a Probability

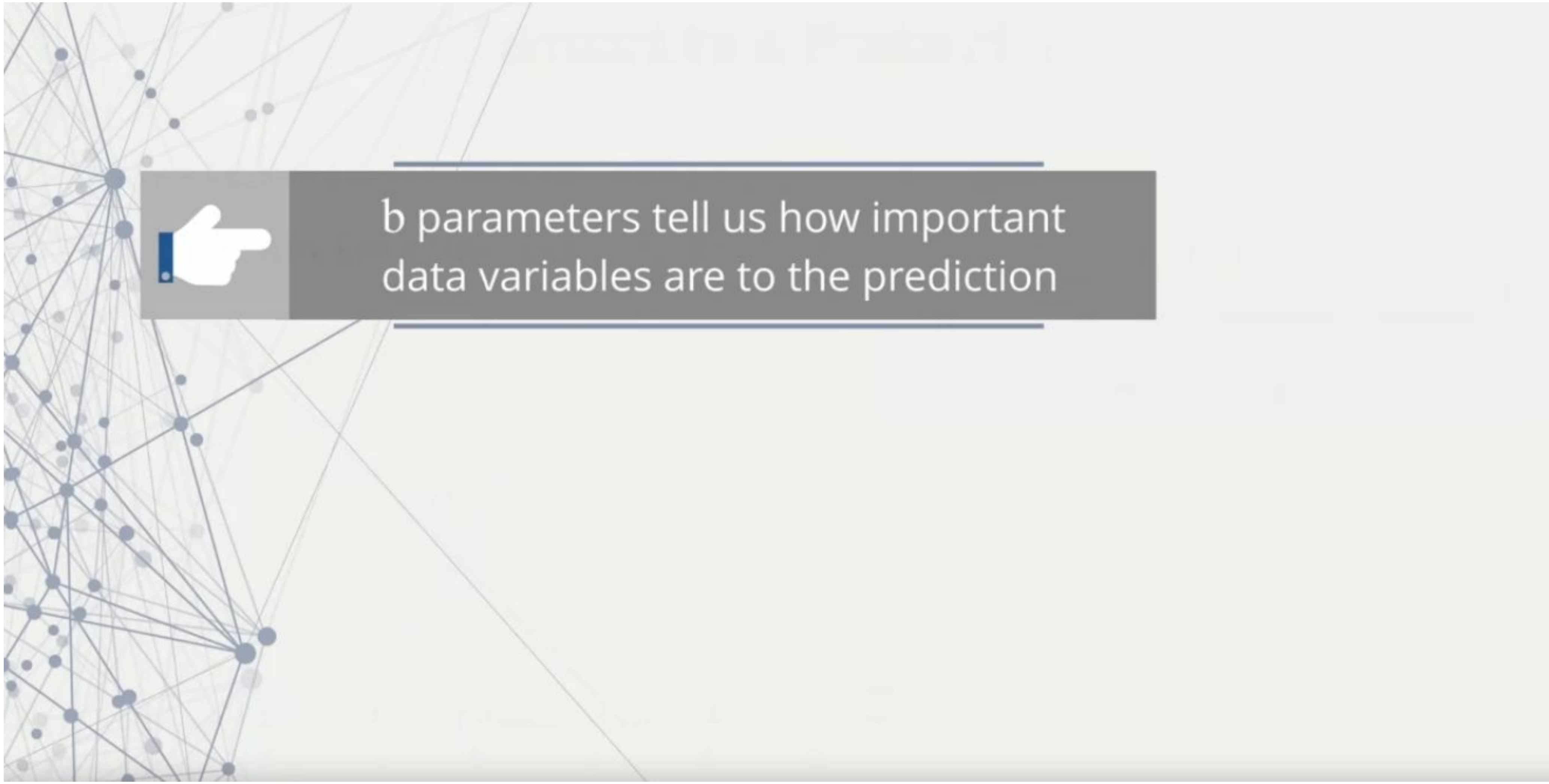
$$z_i = (b_1 \times 0.5) + (b_2 \times 0.8) + (b_3 \times 75) + (b_4 \times 1.2) + b_0$$

Sigmoid Function $p(y_i = 1 | x_i) = \sigma(z_i)$

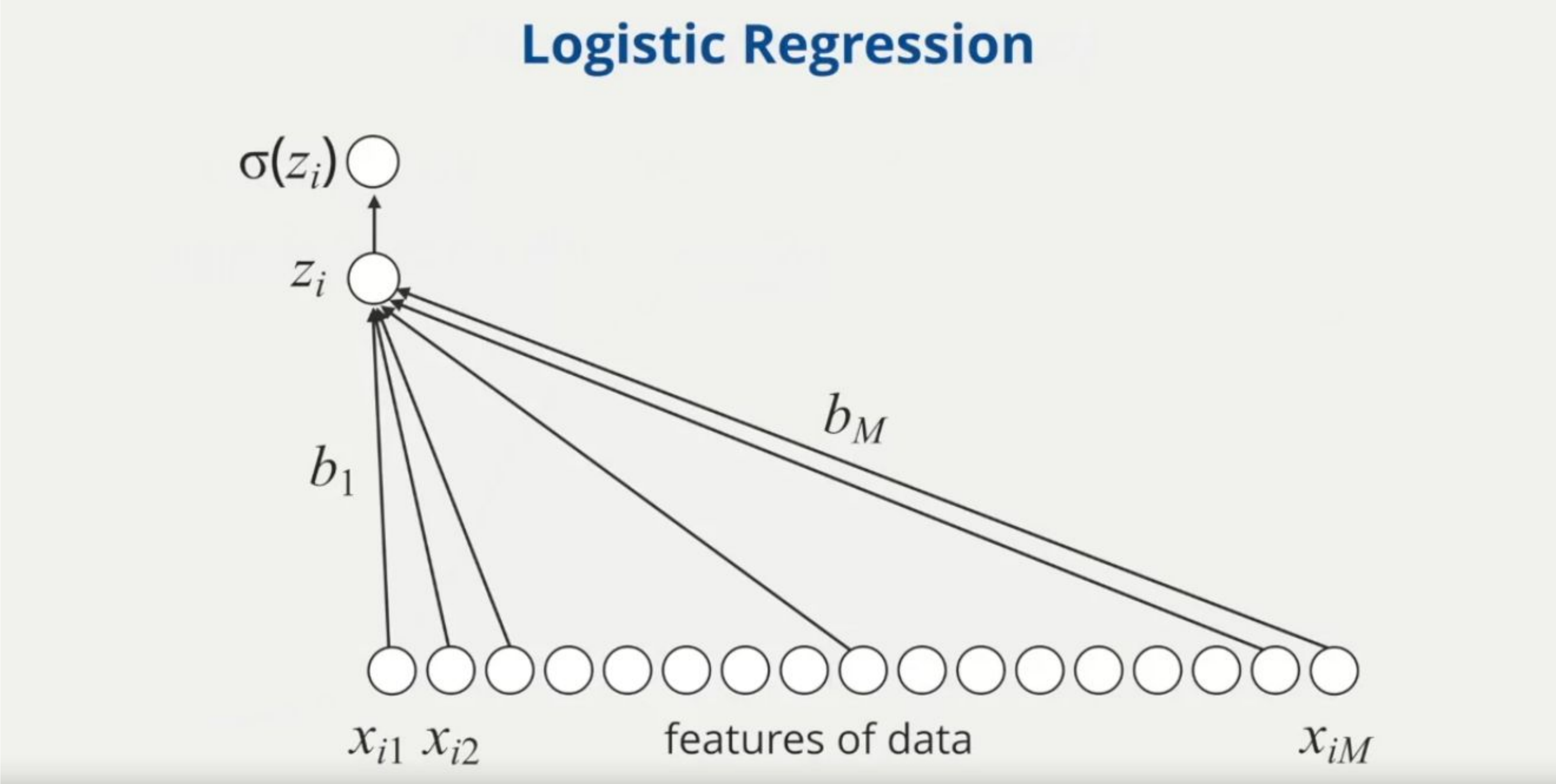


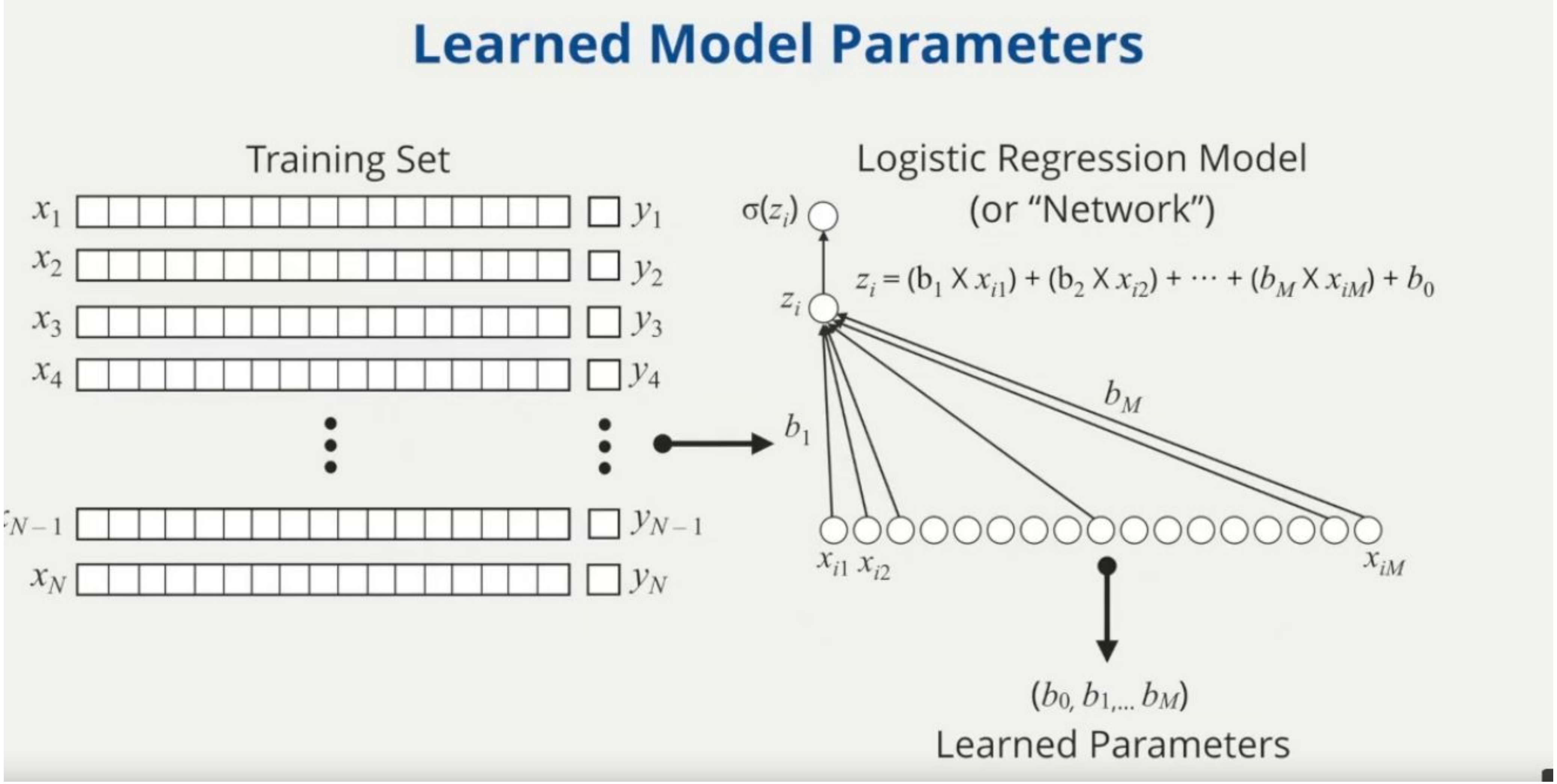
features

Cloud Cover	Humidity	Temperature	Air Pressure
0.5	80%	75	1.2


b parameters tell us how important
data variables are to the prediction

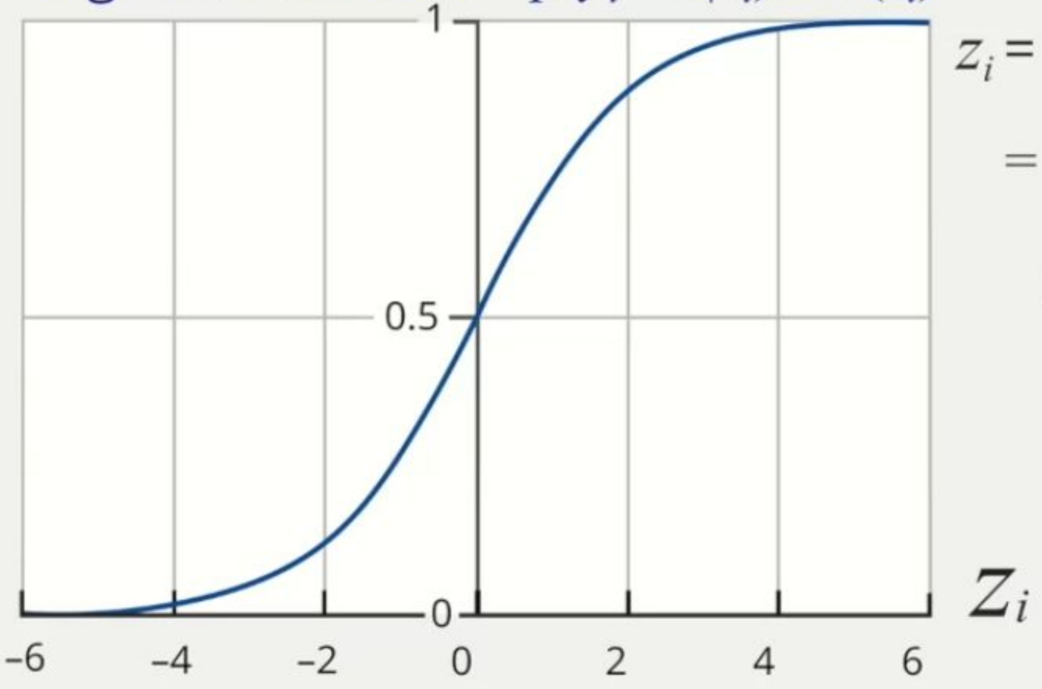
Basics of Neural Networks




Learned Model Parameters
Training Set
x_1 y_1
x_2 y_2
x_3 y_3
x_4 y_4
x_{N-1} y_{N-1}
x_N y_N
Logistic Regression Model
(or “Network”)
$\sigma(z_i)$
z_i
$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$
b_M
b_1
x_{i1} x_{i2}
x_{iM}
$(b_0, b_1, \ldots b_M)$
Learned Parameters

Logistic Regression

Sigmoid Function $p(y_i = 1 | x_i) = \sigma(z_i)$



$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

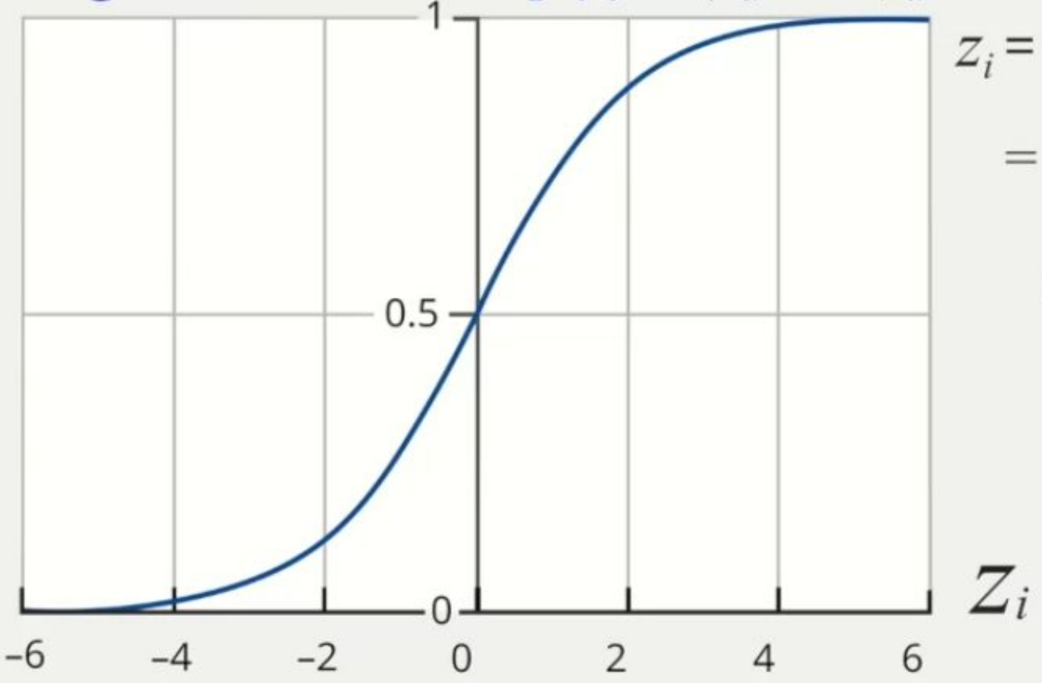
$$= b_0 + x_i \odot b$$

bias

inner product

Logistic Regression

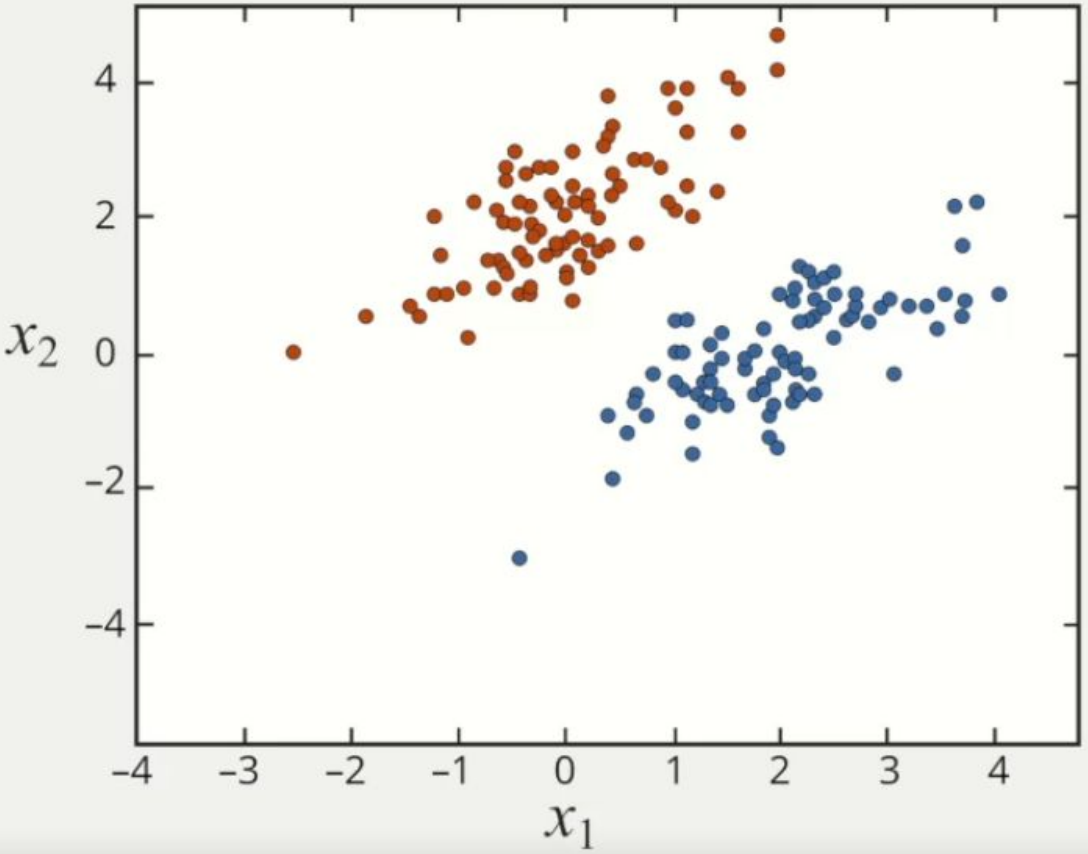
Sigmoid Function $p(y_i = 1 | x_i) = \sigma(z_i)$



$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

$$= \underbrace{b_0}_{\text{bias}} + \underbrace{x_i \odot b}_{\text{inner product}}$$

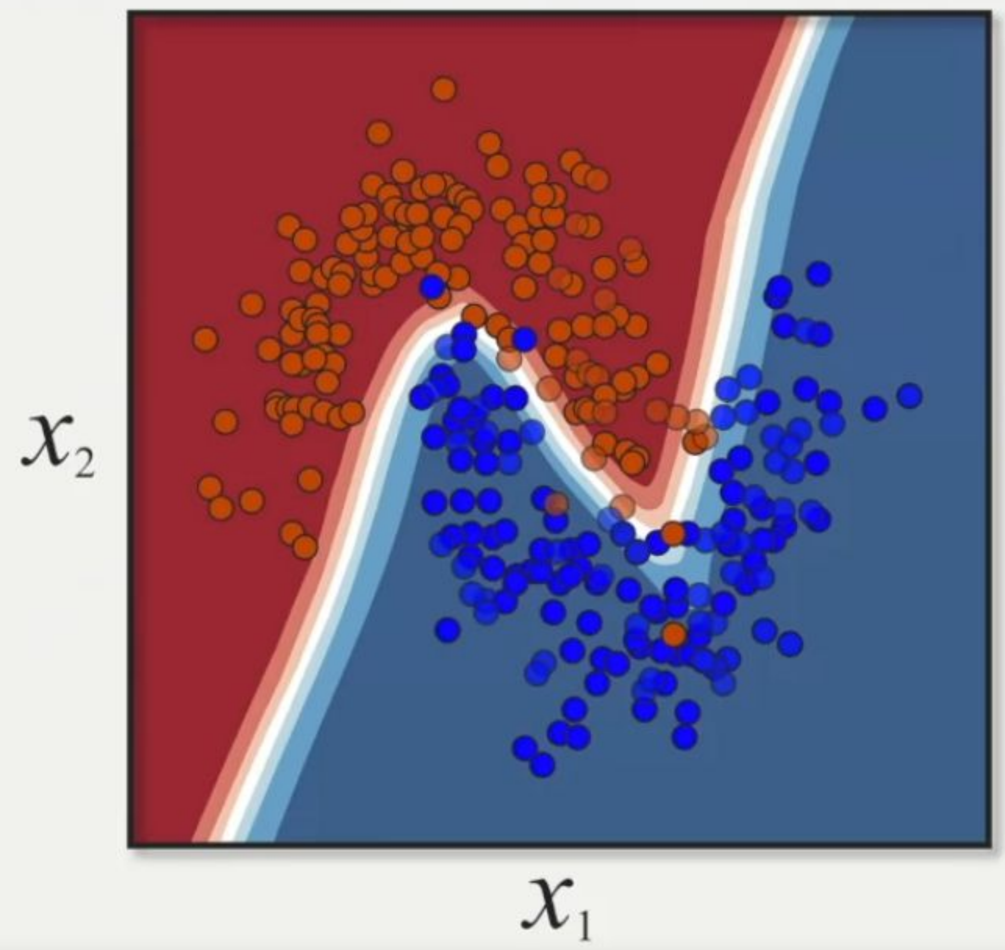
Limitations of Logistic Regression



Linear

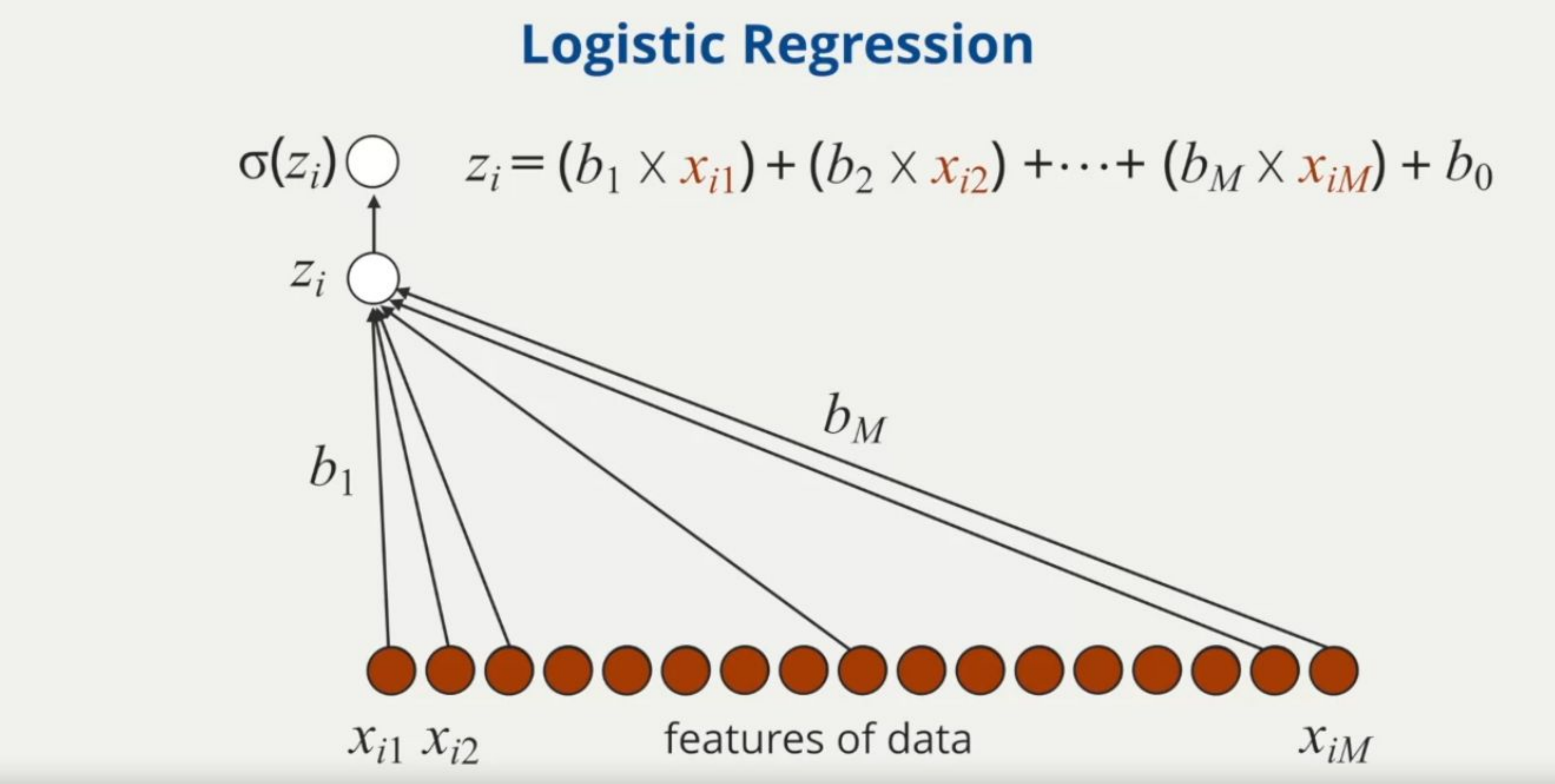
- Linear classifiers can only represent limited relationships
- Often want to use a classifier that can handle non-linearities

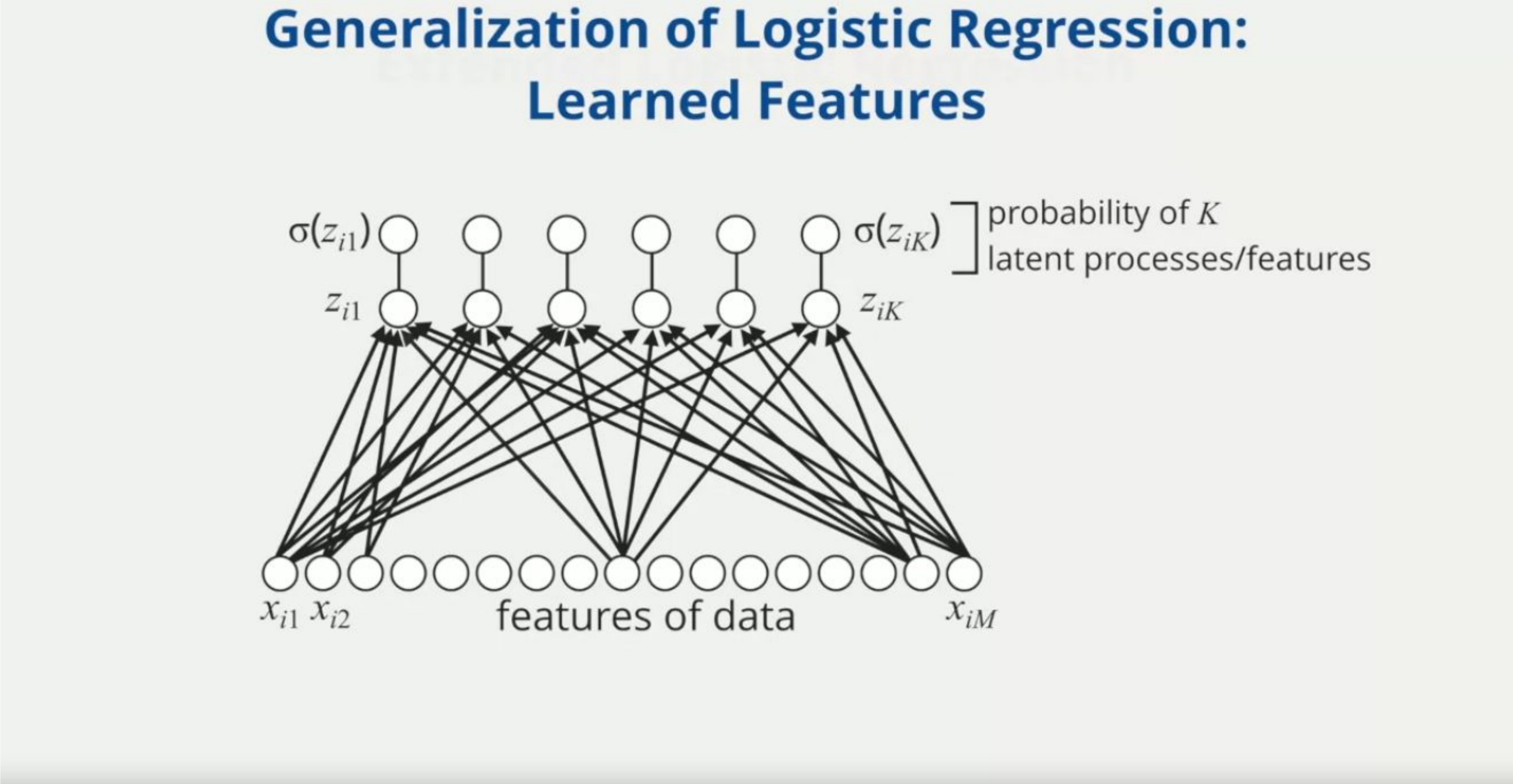
Basics of Neural Networks



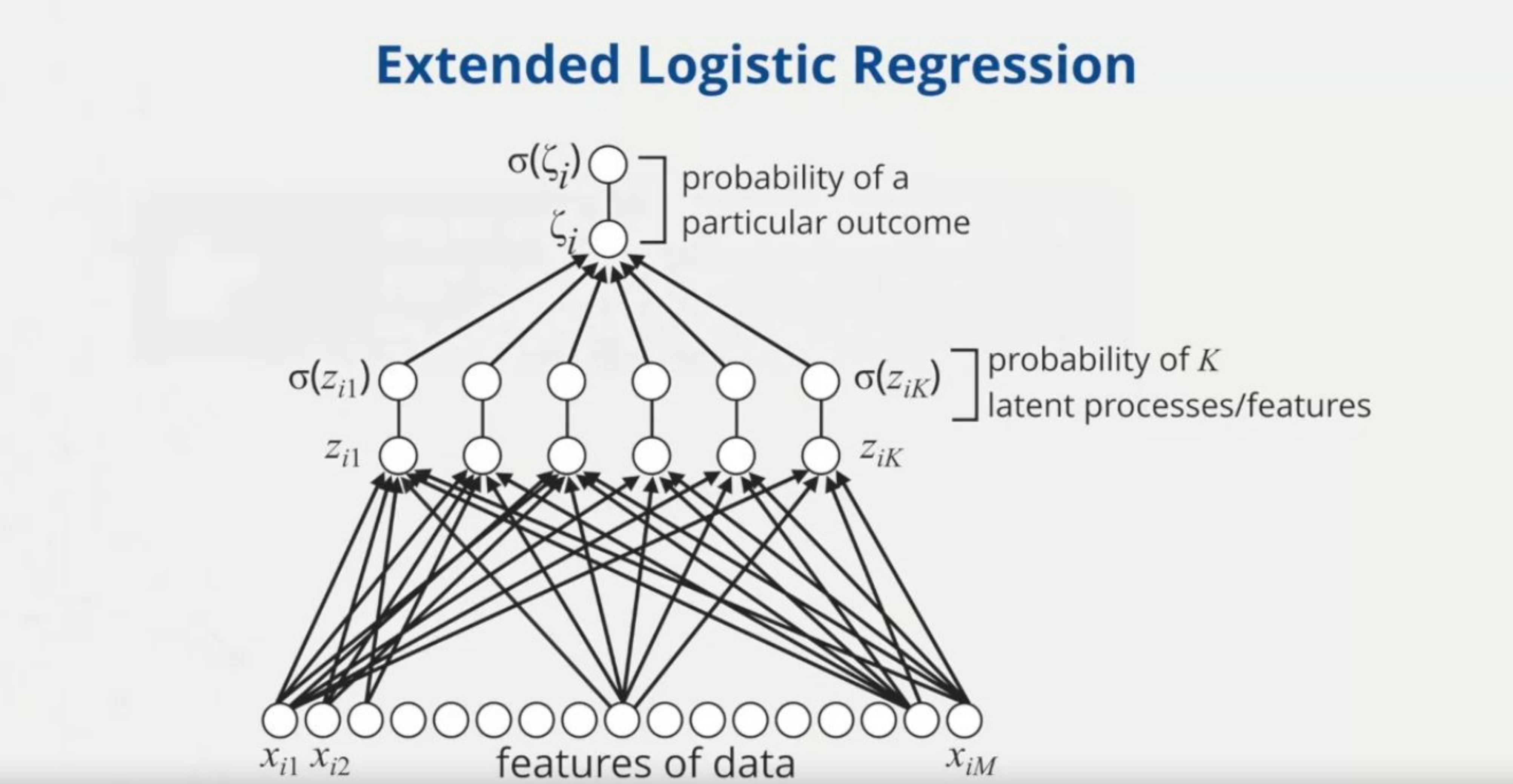
Generalization of Logistic Regression: Learned Features

Basics of Neural Networks

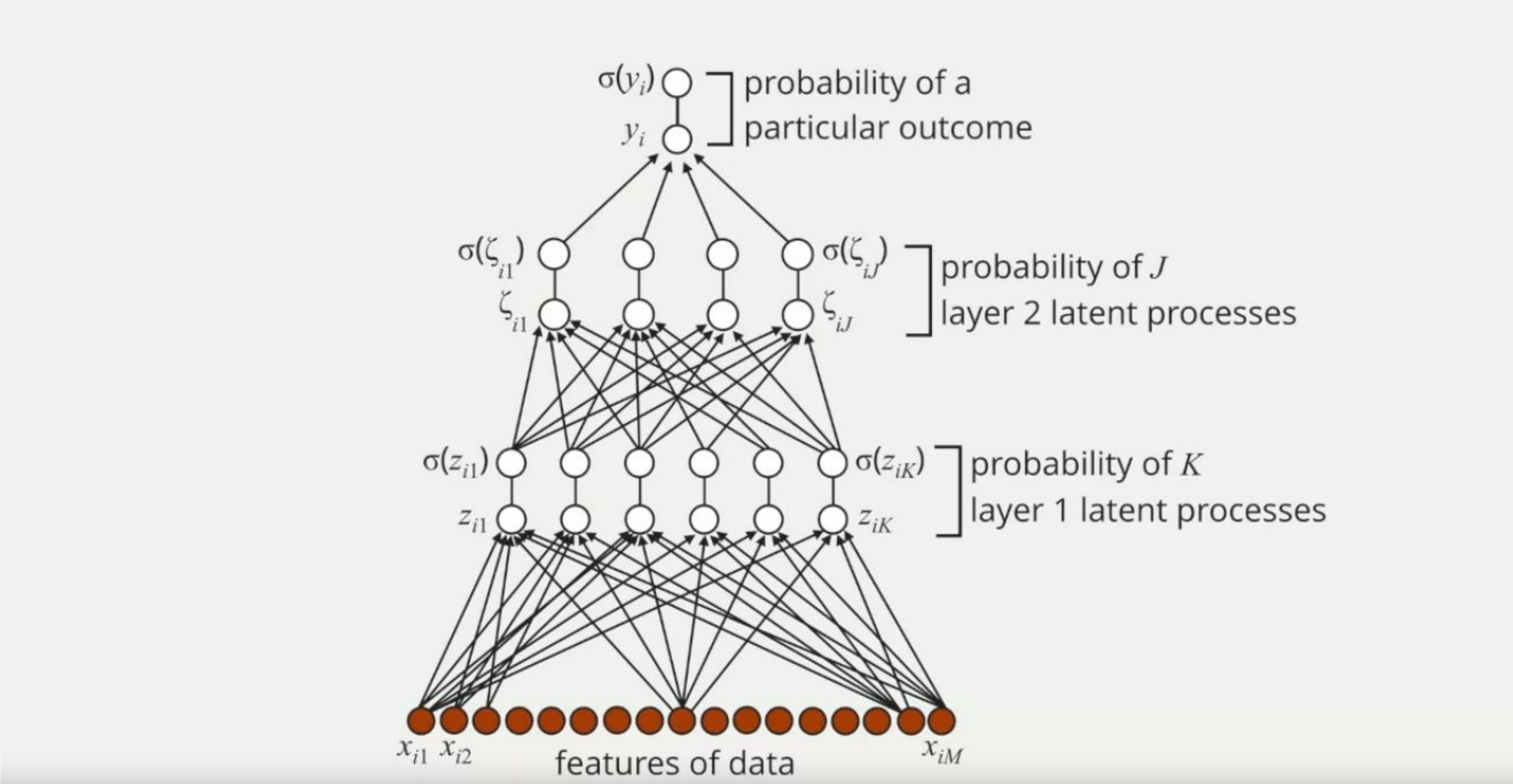


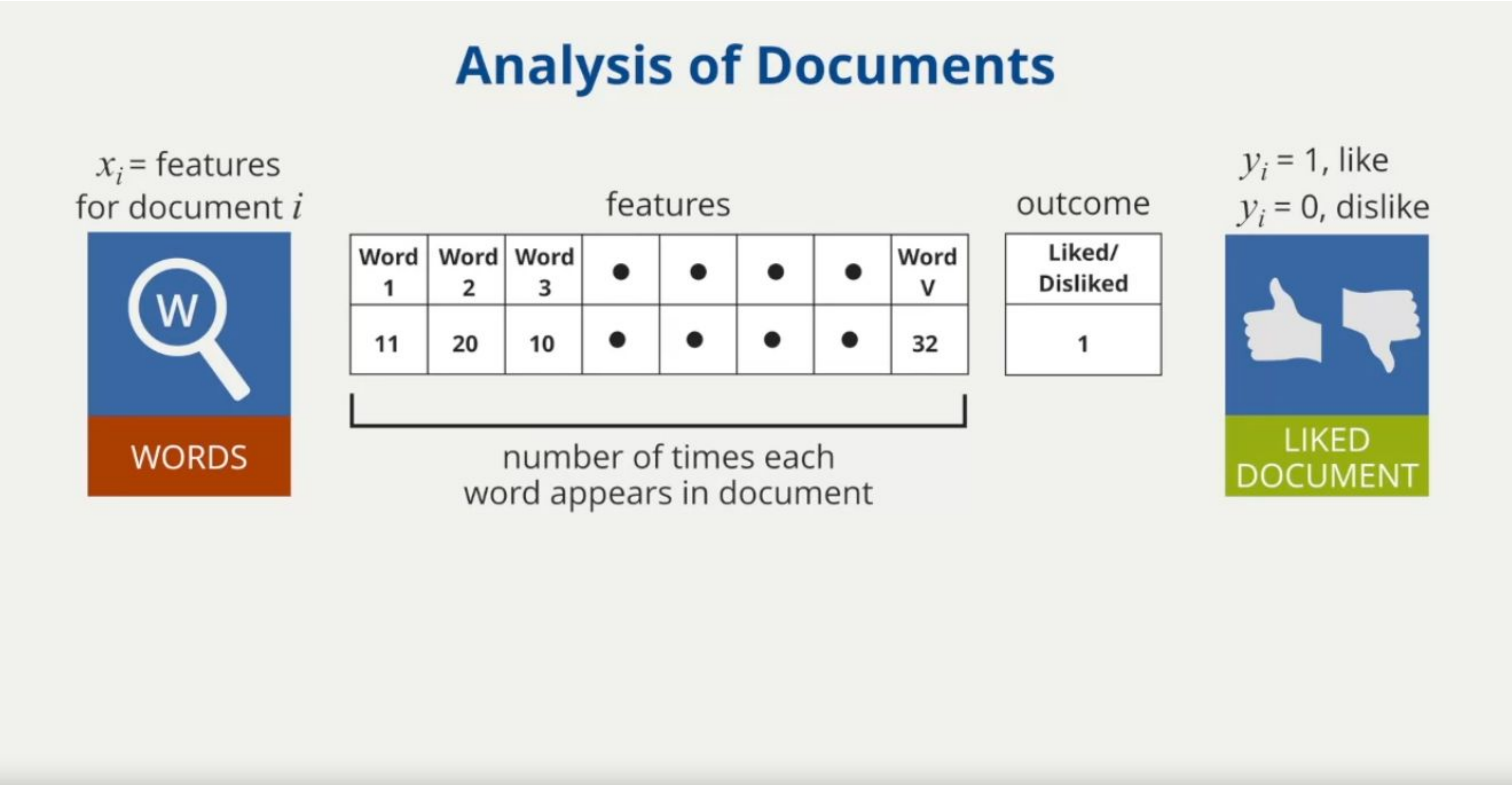

Generalization of Logistic Regression:
Learned Features
σ(z_i1)
σ(z_iK)
probability of K
latent processes/features
z_i1
z_iK
x_i1 x_i2
features of data
x_iM

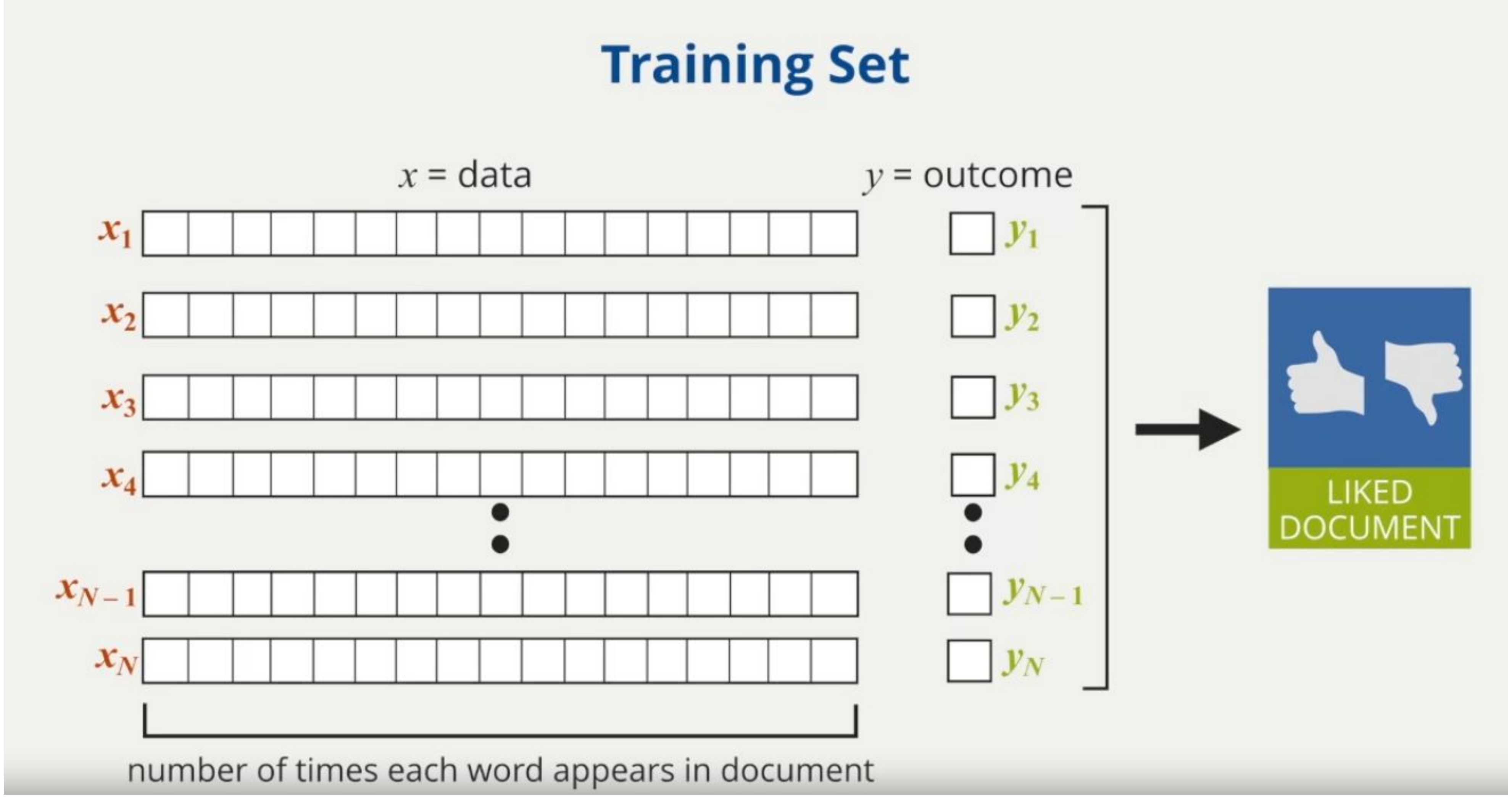
Basics of Neural Networks



Basics of Neural Networks




Analysis of Documents
x_i = features for document i
W
WORDS
features
Word 1 | Word 2 | Word 3 | ● | ● | ● | ● | Word V
11 | 20 | 10 | ● | ● | ● | ● | 32
number of times each word appears in document
outcome
Liked/ Disliked
1
y_i = 1, like
y_i = 0, dislike
LIKED DOCUMENT


Training Set
x = data
y = outcome
x1
x2
x3
x4
xN – 1
xN
y1
y2
y3
y4
yN – 1
yN
LIKED
DOCUMENT
number of times each word appears in document

topics

b_1	b_2			
sports	history			

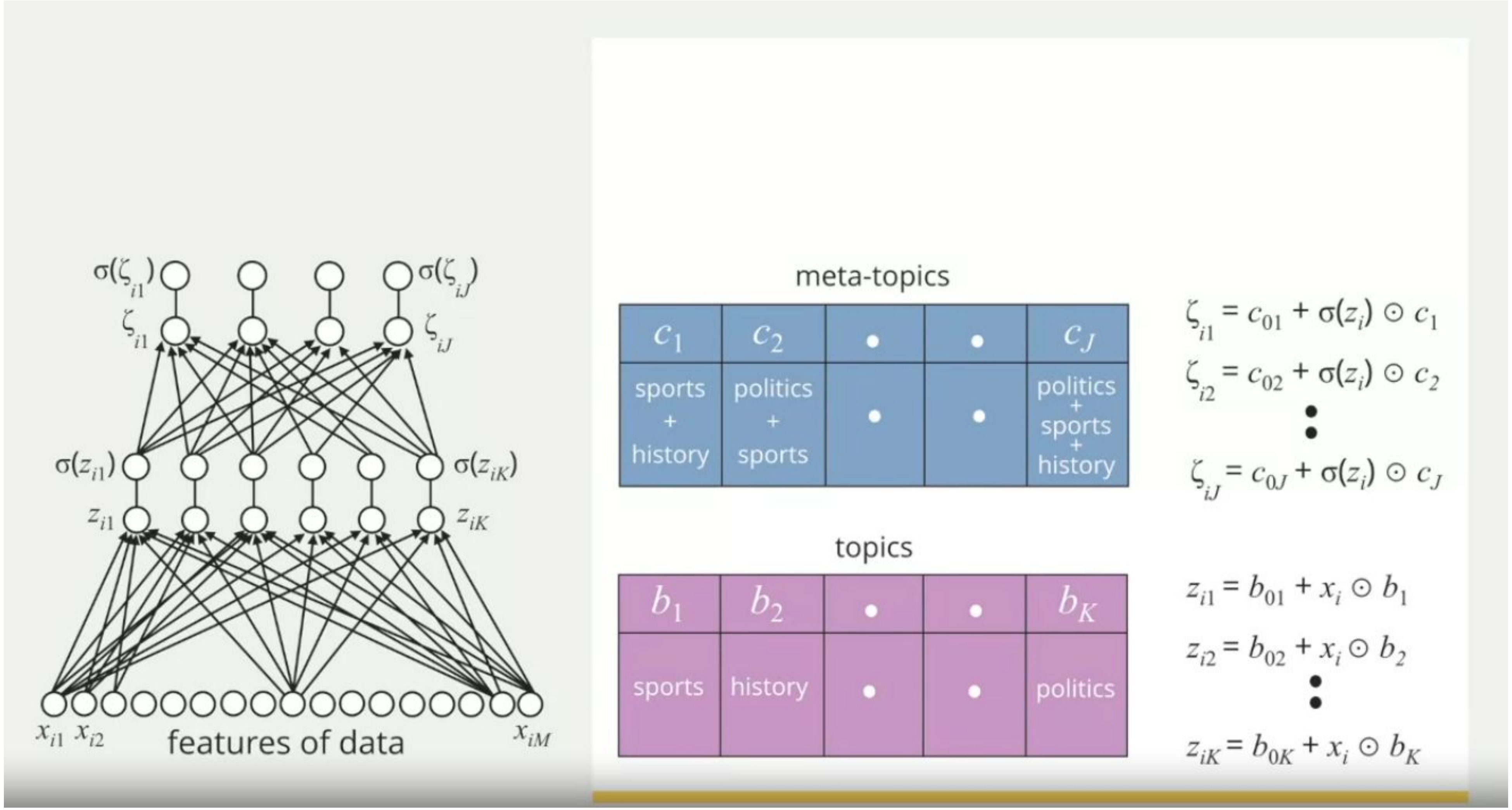
$$z_{i1} = b_{01} + x_i \odot b_1$$

$$z_{i2} = b_{02} + x_i \odot b_2$$

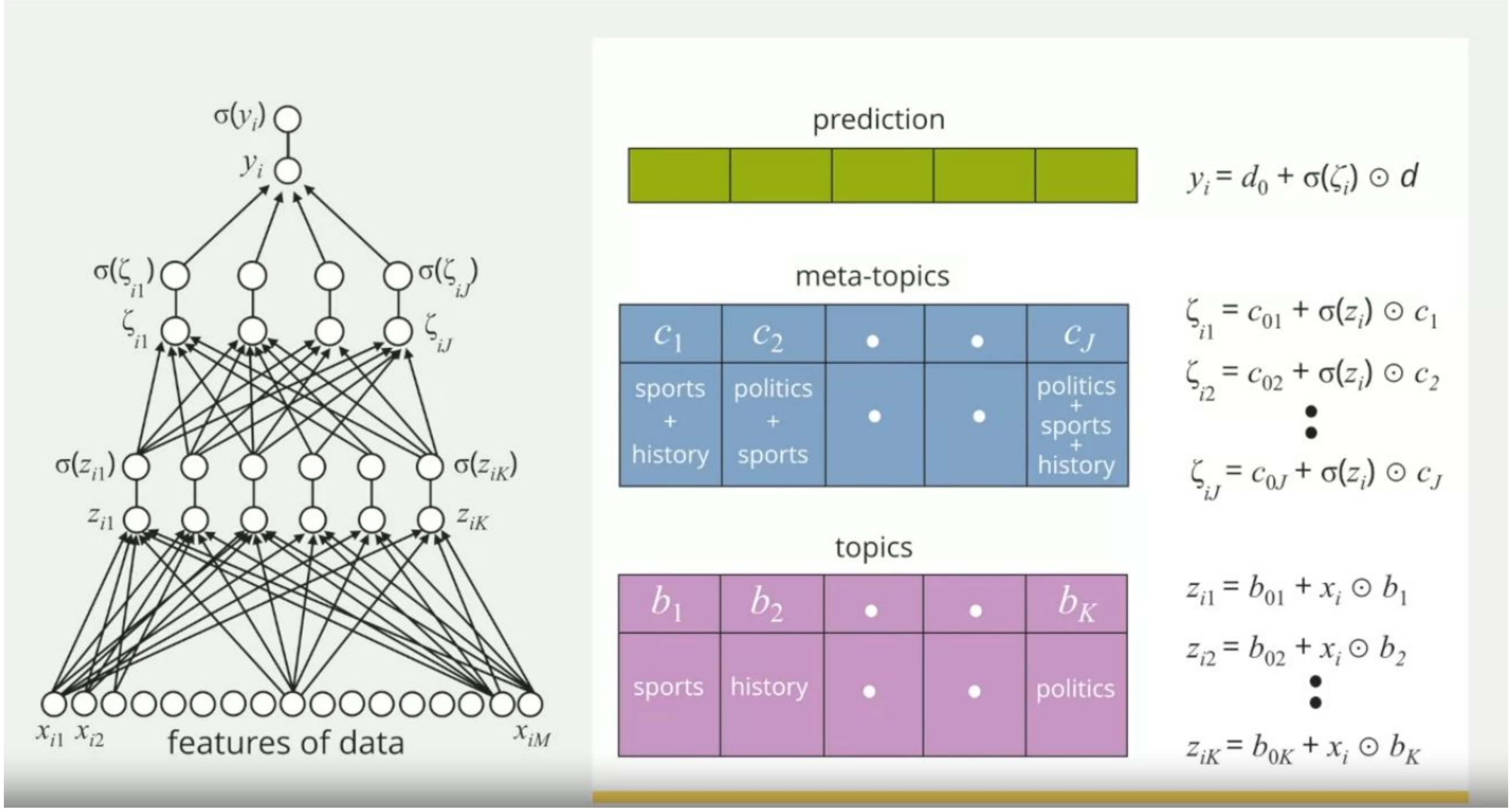
$$\vdots$$

$$z_{iK} = b_{0K} + x_i \odot b_K$$

Basics of Neural Networks

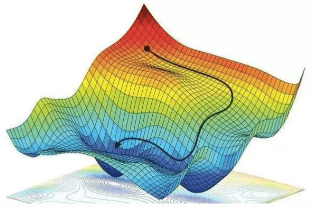


Basics of Neural Networks



Basics of Neural Networks

Gradient Descent



Network parameters $\theta = \{w_1, w_2, \cdots, b_1, b_2, \cdots\}$

Starting Parameters $\theta^0 \longrightarrow \theta^1 \longrightarrow \theta^2 \longrightarrow \ldots\ldots$

$$\nabla L(\theta) = \begin{bmatrix} \partial L(\theta)/\partial w_1 \\ \partial L(\theta)/\partial w_2 \\ \vdots \\ \partial L(\theta)/\partial b_1 \\ \partial L(\theta)/\partial b_2 \\ \vdots \end{bmatrix}$$

$Compute\ \nabla L(\theta^0)$ $\qquad \theta^1 = \theta^0 - \eta \nabla L(\theta^0)$

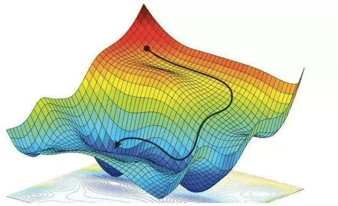
$Compute\ \nabla L(\theta^1)$ $\qquad \theta^2 = \theta^1 - \eta \nabla L(\theta^1)$

Millions of parameters ……

To compute the gradients efficiently, we use ***backpropagation***.

Basics of Neural Networks

Gradient Descent



Network parameters $\theta = \{w_1, w_2, \cdots, b_1, b_2, \cdots\}$

Starting Parameters $\theta^0 \longrightarrow \theta^1 \longrightarrow \theta^2 \longrightarrow \ldots\ldots$

$$\nabla L(\theta) = \begin{bmatrix} \partial L(\theta)/\partial w_1 \\ \partial L(\theta)/\partial w_2 \\ \vdots \\ \partial L(\theta)/\partial b_1 \\ \partial L(\theta)/\partial b_2 \\ \vdots \end{bmatrix}$$

Compute $\nabla L(\theta^0)$ $\qquad \theta^1 = \theta^0 - \eta \nabla L(\theta^0)$

Compute $\nabla L(\theta^1)$ $\qquad \theta^2 = \theta^1 - \eta \nabla L(\theta^1)$

Millions of parameters ……

To compute the gradients efficiently, we use ***backpropagation***.

Chain Rule

Case 1

$$y = g(x) \quad z = h(y)$$

$$\Delta x \to \Delta y \to \Delta z \qquad \frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}$$

Case 2

$$x = g(s) \qquad y = h(s) \qquad z = k(x, y)$$

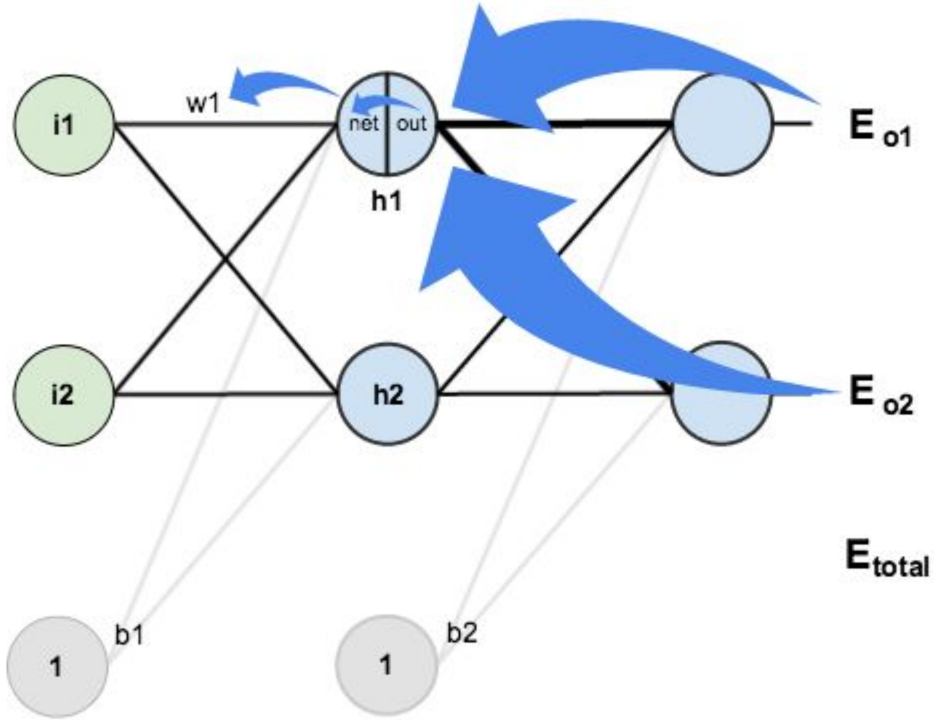
$$\begin{tikzcd} & \Delta x \arrow[dr] & \\ \Delta s \arrow[ur] \arrow[dr] & & \Delta z \\ & \Delta y \arrow[ur] & \end{tikzcd}$$

$$\frac{dz}{ds} = \frac{\partial z}{\partial x}\frac{dx}{ds} + \frac{\partial z}{\partial y}\frac{dy}{ds}$$

Basics of Neural Networks

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$



$E_{total} = E_{o1} + E_{o2}$

Basics of Neural Networks