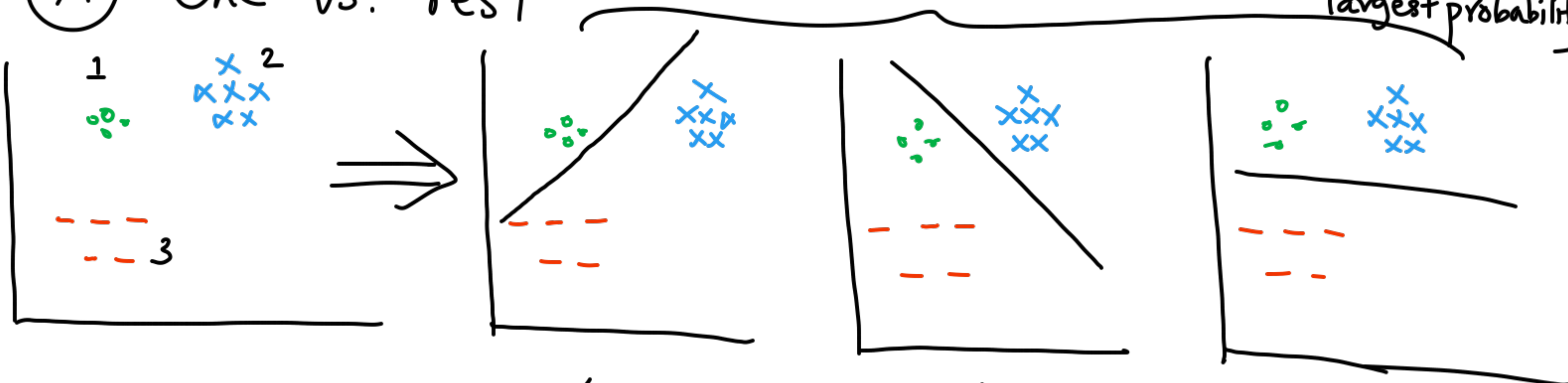
CS725

Logistic Regression with multiple classes

Ⓐ One vs. rest

Pick the label with the largest probability



$P(y=1|x)$ $\quad$ $P(y=2|x)$ $\quad$ $P(y=3|x)$

(B) SOFTMAX REGRESSION (MULTINOMIAL LOGISTIC REGRESSION)

Given $y_i \in \{1, 2, \ldots, K\}$, for a point x, we want a predictor function

$$h_w(x) = \begin{bmatrix} P(y=1 \mid x; w) \\ P(y=2 \mid x; w) \\ \vdots \\ P(y=K \mid x; w) \end{bmatrix} \longrightarrow (1)$$

Softmax regression assumes weight vectors, one per class, to yield scores for K classes;

$$\begin{bmatrix} w_1^T x \\ w_2^T x \\ \vdots \\ w_K^T x \end{bmatrix} \longrightarrow (2)$$

$w_1, w_2, \ldots, w_K$, where $w_j \in \mathbb{R}^d$

if [illegible] m

How do we go from ② to ① ? SOFTMAX TRANSFORMATION

What is softmax? If $V = [v_1, v_2, \ldots, v_k]$, $v_j \in \mathbb{R}$, $\text{Softmax}(V) = \dfrac{1}{\sum_k e^{v_k}} \begin{bmatrix} e^{v_1} \\ e^{v_2} \\ \vdots \\ e^{v_k} \end{bmatrix}$

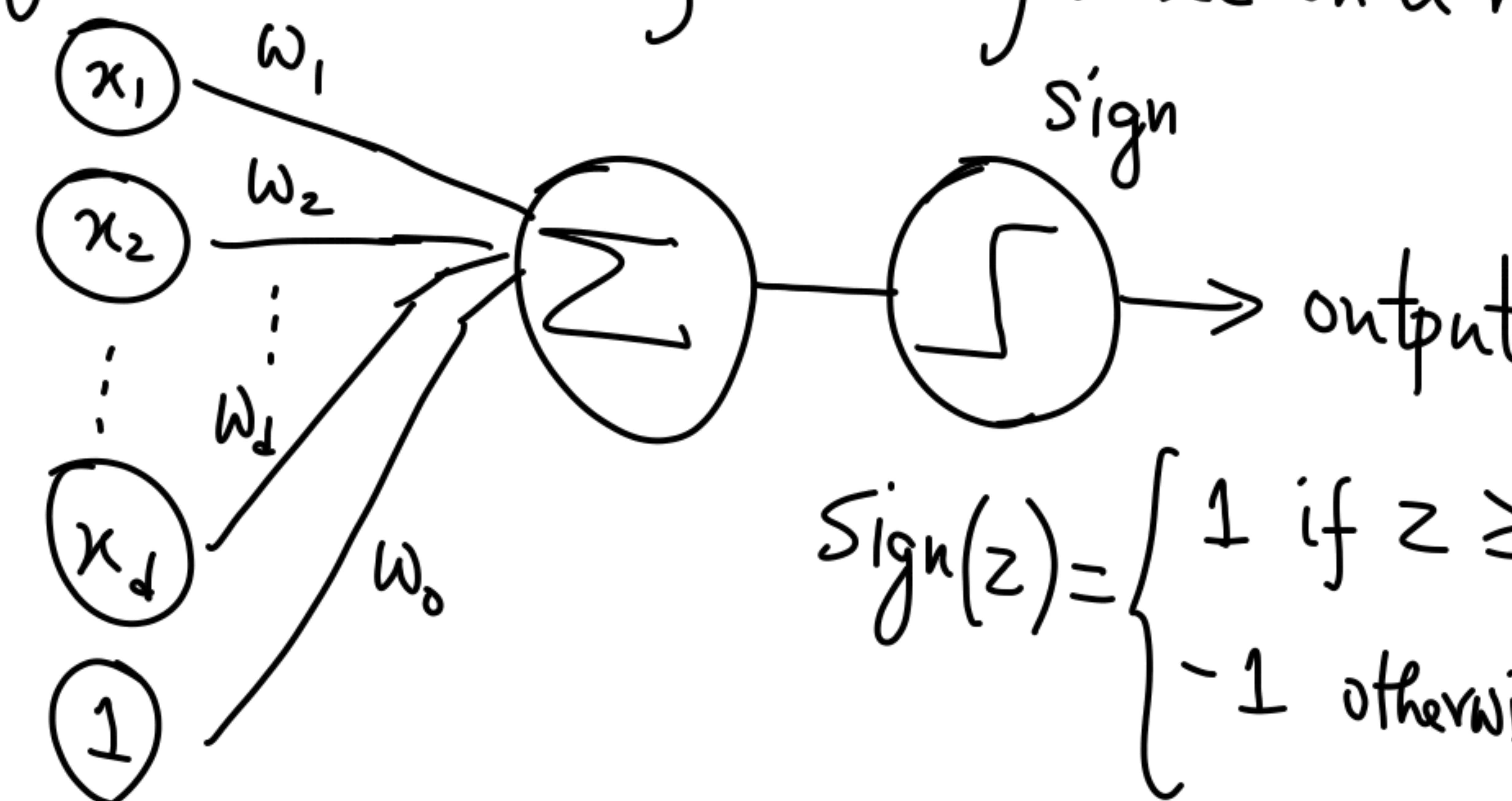
NORMALIZATION FACTOR

$$h_w(x) = \begin{bmatrix} P(y=1 \mid x; w) \\ P(y=2 \mid x; w) \\ \vdots \\ P(y=K \mid x; w) \end{bmatrix} = \frac{1}{\sum_j e^{w_j^T x}} \begin{bmatrix} e^{w_1^T x} \\ e^{w_2^T x} \\ \vdots \\ e^{w_k^T x} \end{bmatrix} \longrightarrow ③$$

Loss function remains cross-entropy loss with the predicted distribution derived as in 3 and the reference distribution being a one-hot vector over K classes.

PERCEPTRON MODEL OF CLASSIFICATION

Linear model of classification. The perceptron model is Ⓐ the basic unit of neural learning. Loosely based on a neuron.



ROSENBLATT'S PERCEPTRON MODEL

$$\text{sign}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

(B) Online Learner ⇒ that is, it acts on examples one after the other

(C) Error-driven or mistake-driven algorithm ⇒ make a weight update only when the learner makes a mistake

PERCEPTRON ALGORITHM

Inputs: Set of training points $\mathcal{D}$, max # of iterations T, initialize the wt vector w [Let $w \leftarrow \vec{0}$] (all zero's vector)

for $t = 1, \ldots, T$

for all $(x, y) \in \mathcal{D}$, $x \in \mathbb{R}^d$, $y \in \{-1, 1\}$

Can also write $\text{if}(y w^T x \leq 0)$

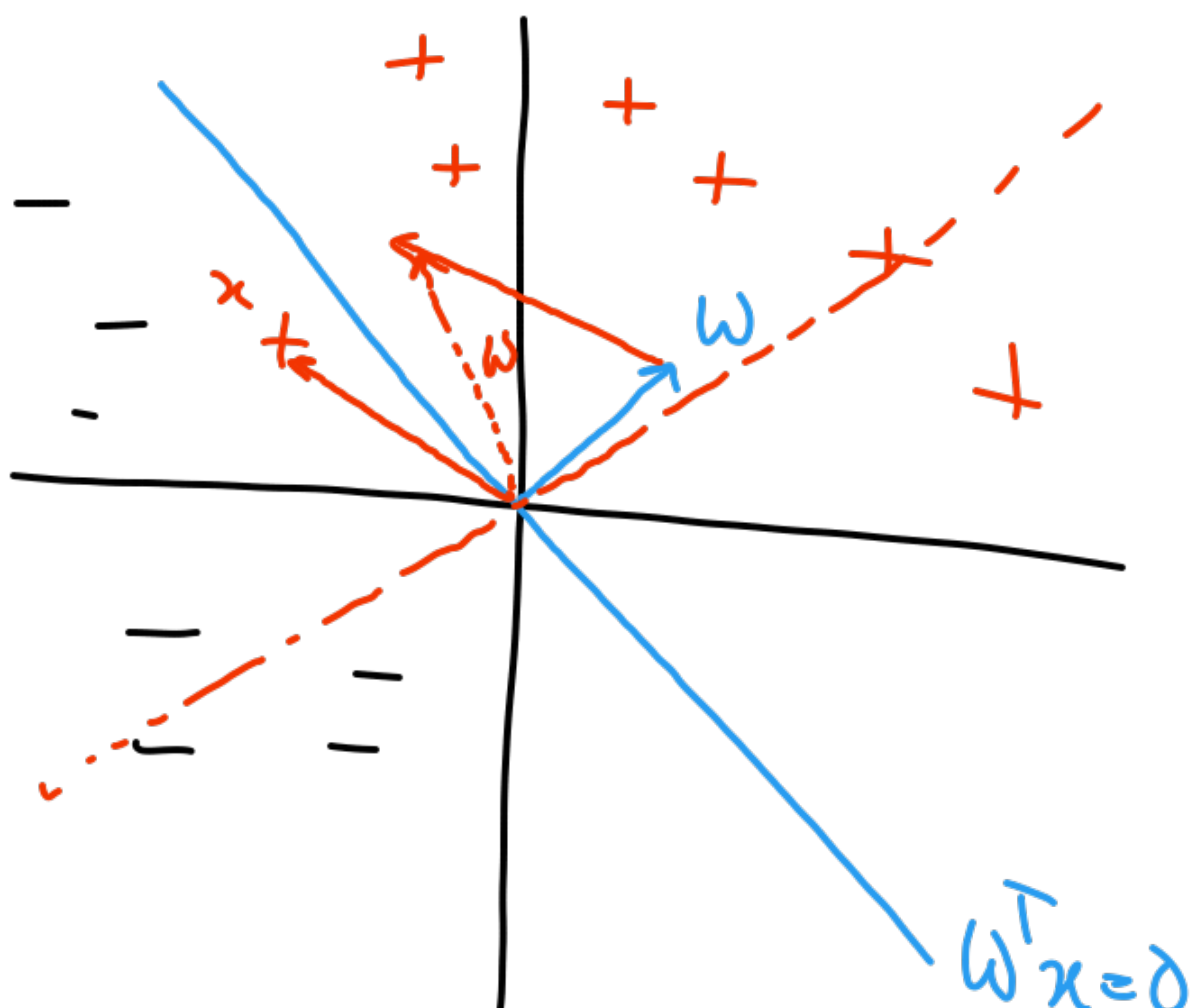
$$\left\{\begin{array}{l} \text{Predict } \hat{y} = \text{sign}(w^T x) \\ \text{if}(\hat{y} \neq y) \end{array}\right.$$

$$w \leftarrow w + \eta y x \longrightarrow w = \begin{cases} w + x & \text{if } y = 1 \\ w - x & \text{if } y = -1 \end{cases}$$

return w

Geometric Intuition for the Perceptron

Moving w towards or away from a misclassified example depending on whether its true label is 1 or -1, respectively



Does the perceptron guarantee to be more correct on a current misclassified example?

(A) Yes, more correct but not guaranteed to give the correct label with a single update

Consider a misclassified example $(x, y) \in \mathcal{D}$

$$w_{new} \leftarrow w_{old} + yx$$

$$y w_{new}^T x = y(w_{old} + yx)^T x = y w_{old}^T x + y^2 \|x\|_2^2 > y w_{old}^T x$$

We know that $y w_{old}^T x \leq 0$, and $y w_{new}^T x > y w_{old}^T x$

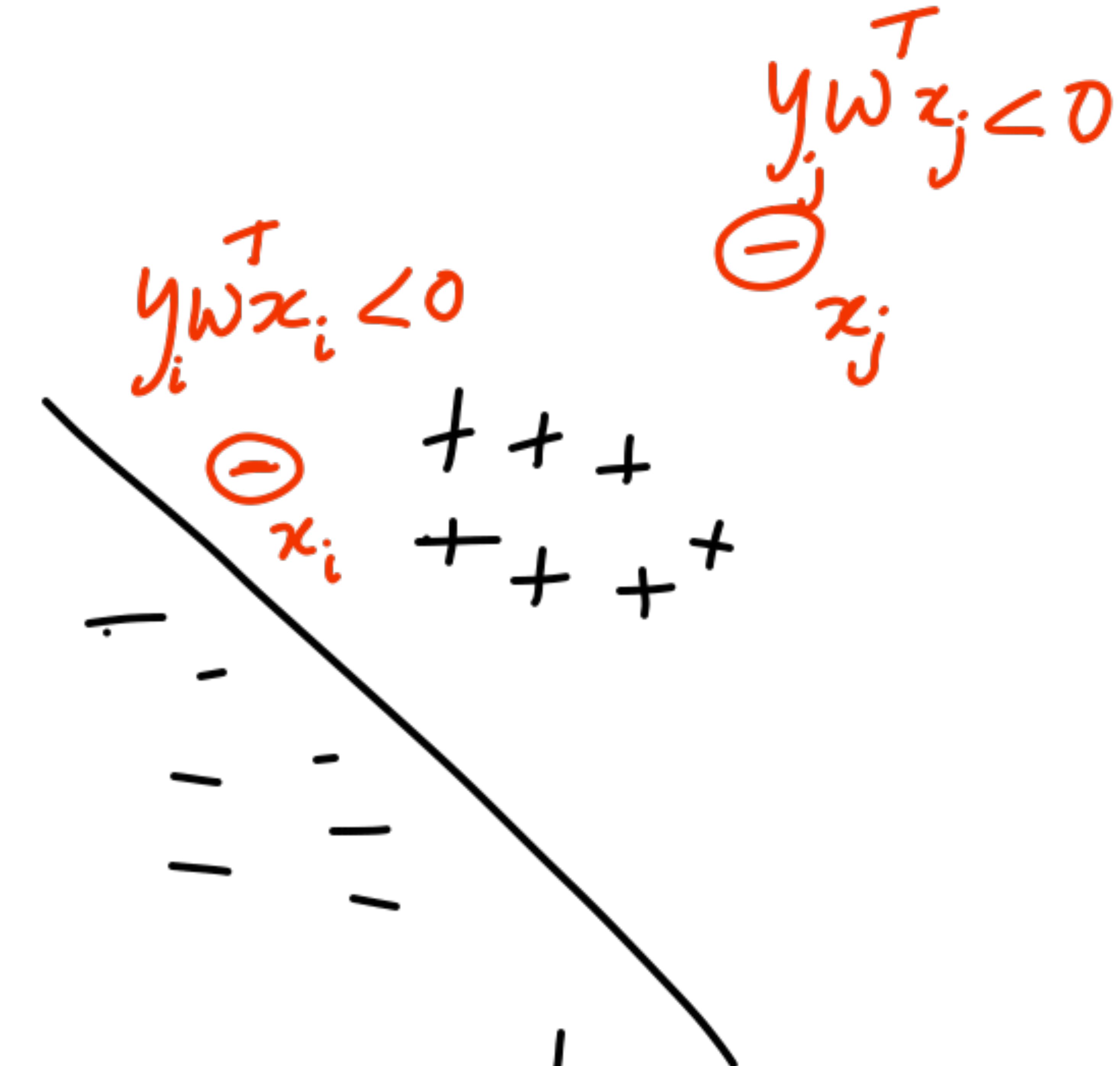
$\Rightarrow$ thus guaranteeing that we are more correct on (x, y)

What is the perceptron optimizing?

Minimize the misclassified examples

Let $\mathcal{D}'$ be the set of misclassified examples, then we want

$$\min \sum_{(x,y)\in\mathcal{D}'} -y w^T x$$



Perceptron loss for one example $(x,y) \Rightarrow \max(0, -y w^T x)$

Perceptron loss for $\mathcal{D} \Rightarrow L_{perc}(w;\mathcal{D}) = \sum_i \max(0, -y_i w^T x_i)$

(Grad descent on L_{perp} yields $w \leftarrow w + yx$ if $yw^Tx \leq 0$)

The perceptron is <u>guaranteed to converge</u> (i.e., make no more mistakes) on <u>linearly separable data</u>