CS440/ECE448 Lecture 8: Perceptron

Mark Hasegawa-Johnson, 2/2021

Outline

- A little history: perceptron as a model of a biological neuron
- The perceptron learning algorithm
- Linear separability, yx, and convergence
    - It converges to the right answer, even with $\eta = 1$, if data are linearly separable
    - If data are not linearly separable, it's necessary to use $\eta = 1/n$.
- Multi-class perceptron

The Giant Squid Axon

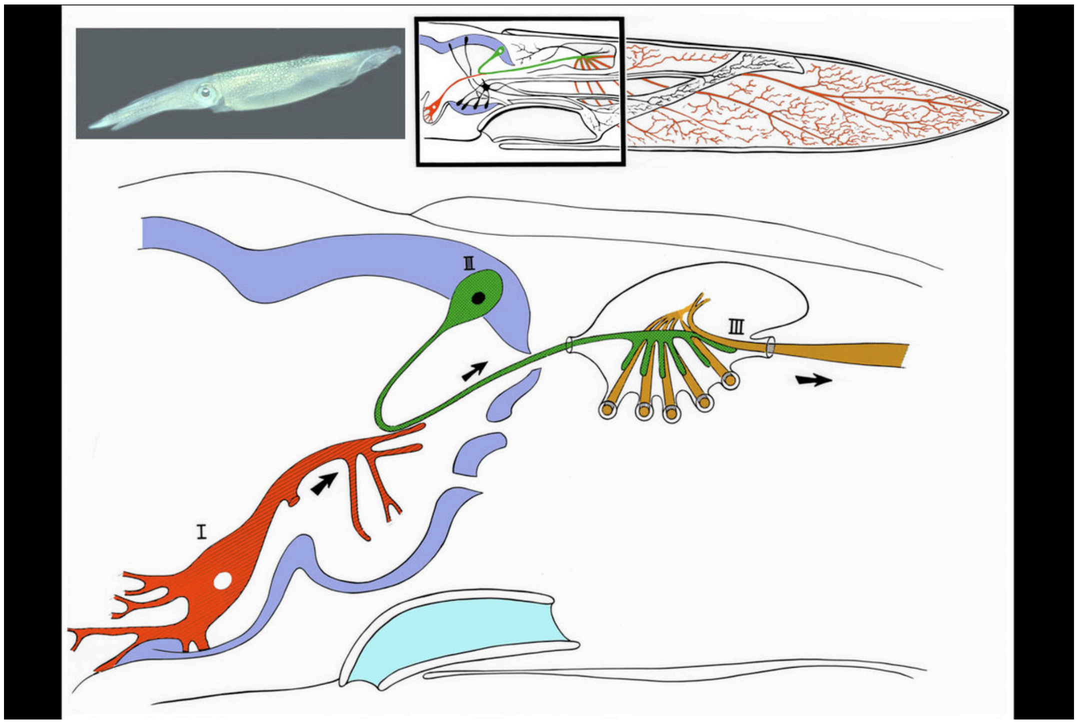
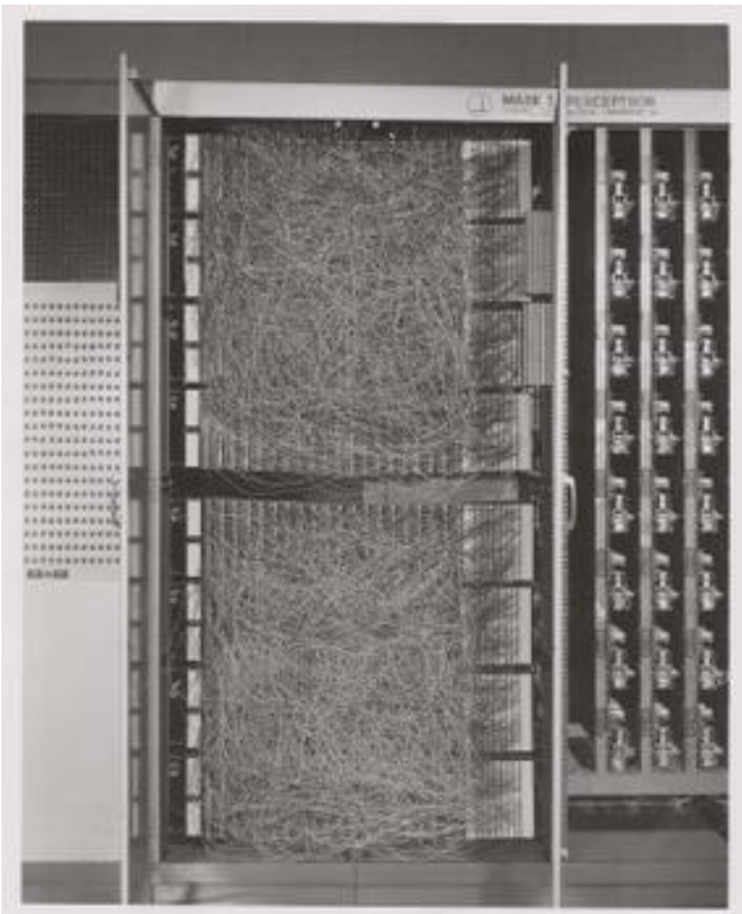


Image released to the public domain by lkkisan, 2007.
Modified from Llinás, Rodolfo R. (1999). The squid Giant Synapse.

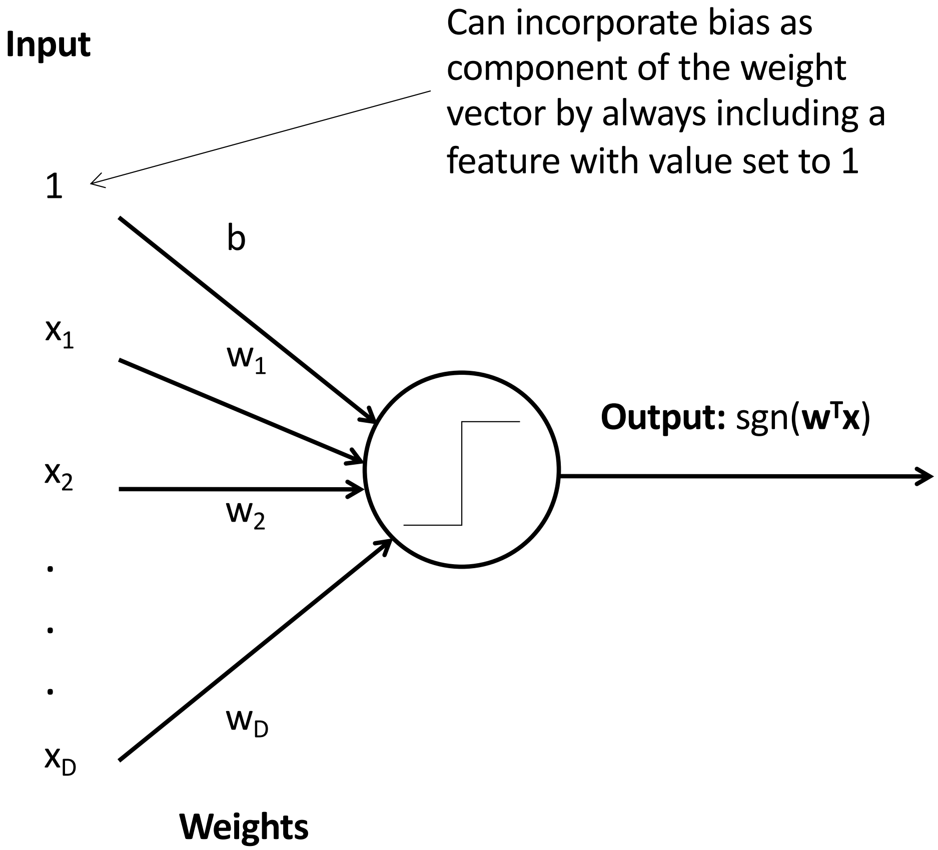
- 1909: Williams describes the giant squid axon (III: 1mm thick)
- 1939: Young describes the synapse.
- 1952: Hodgkin & Huxley publish an electrical current model for the generation of binary action potentials from real-valued inputs.

Perceptron



- 1959: Rosenblatt is granted a patent for the “perceptron,” an electrical circuit model of a neuron.

Perceptron



Perceptron model: action potential = signum(affine function of the features)

$$\hat{y} = \text{sgn}(w_1 x_1 + \ \dots + w_D x_D \ + b) \\ = \ \text{sgn}(w^T x)$$

Where $w = [w_1, \dots, w_D, b]^T$,

$x = [x_1, \dots, x_D, 1]^T$, and

$$\text{sgn}(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$

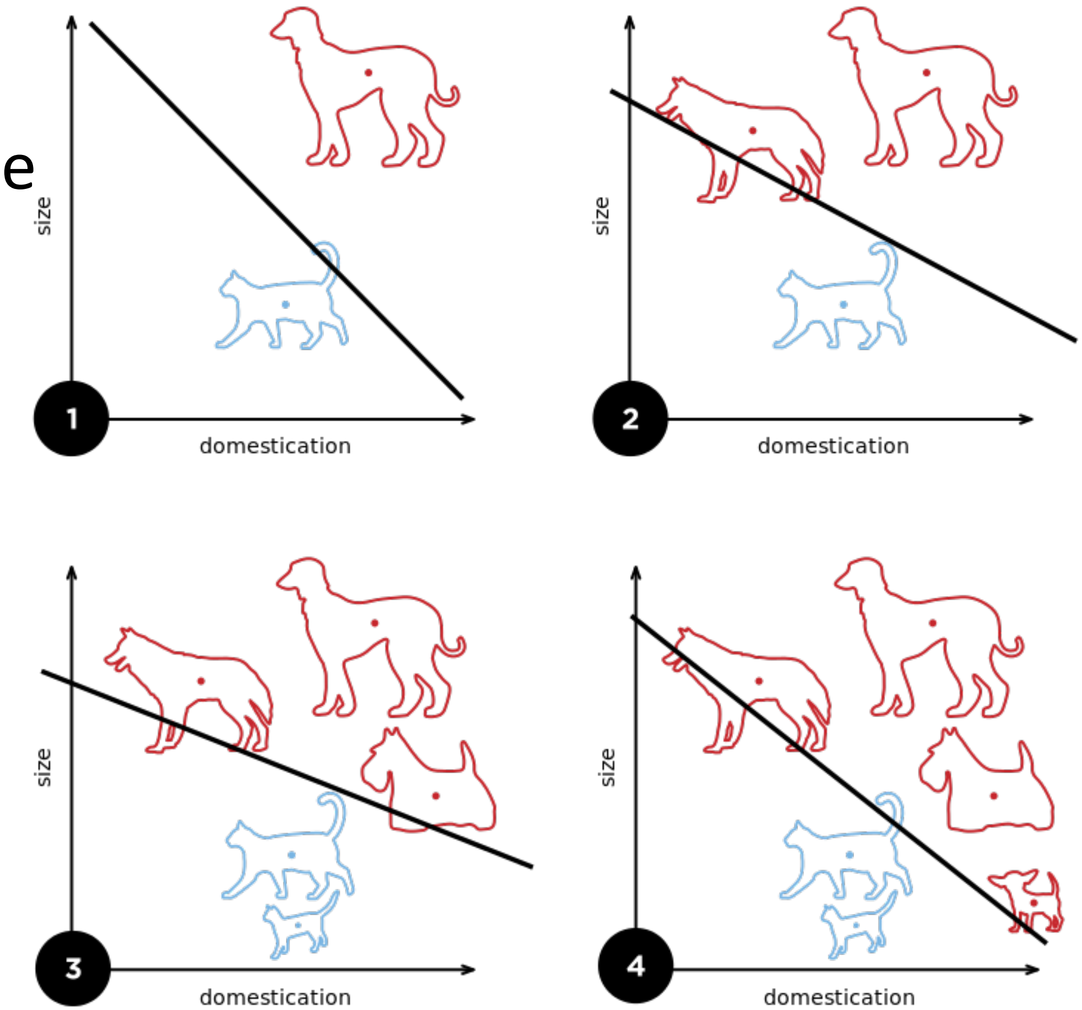
Outline

- A little history: perceptron as a model of a biological neuron
- The perceptron learning algorithm
- Linear separability, yx, and convergence
  - It converges to the right answer, even with $\eta = 1$, if data are linearly separable
  - If data are not linearly separable, it's necessary to use $\eta = 1/n$.
- Multi-class perceptron

Perceptron

Rosenblatt’s big innovation: the perceptron learns from examples.

- Initialize weights randomly
- Cycle through training examples in multiple passes (*epochs*)
- For each training example:
  - If classified correctly, do nothing
  - If classified incorrectly, update weights



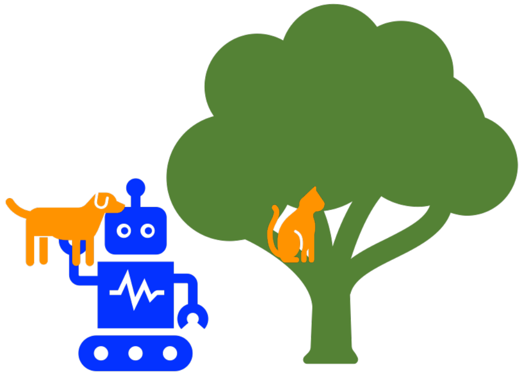
Perceptron

For each training instance $\boldsymbol{x}$ with ground truth label $y \in \{-1,1\}$:

- Classify with current weights: $\hat{y} = \text{sgn}(w^T x)$
- Update weights:
  - if $y = \hat{y}$ then do nothing
  - If $y \neq \hat{y}$ then $w = w + \eta y x$
  - η (eta) is a “learning rate.” For now, let’s assume η=1.

Perceptron example: dogs versus cats

Can you write a program that can tell which ones are dogs, and which ones are cats?



x_1 = # times the animal comes when called (out of 40).

x_2 = weight of the animal, in pounds.

$x = [x_1, x_2, 1]^T$.

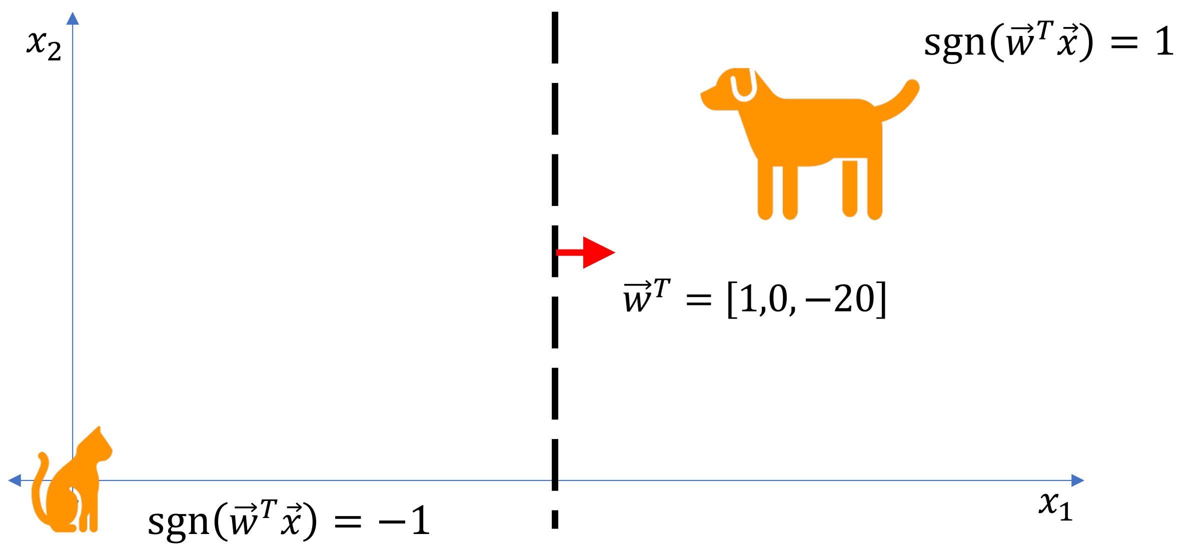
$y = 1$ means "dog"

$y = -1$ means "cat"

$$\hat{y} = \text{sgn}(w^T x)$$

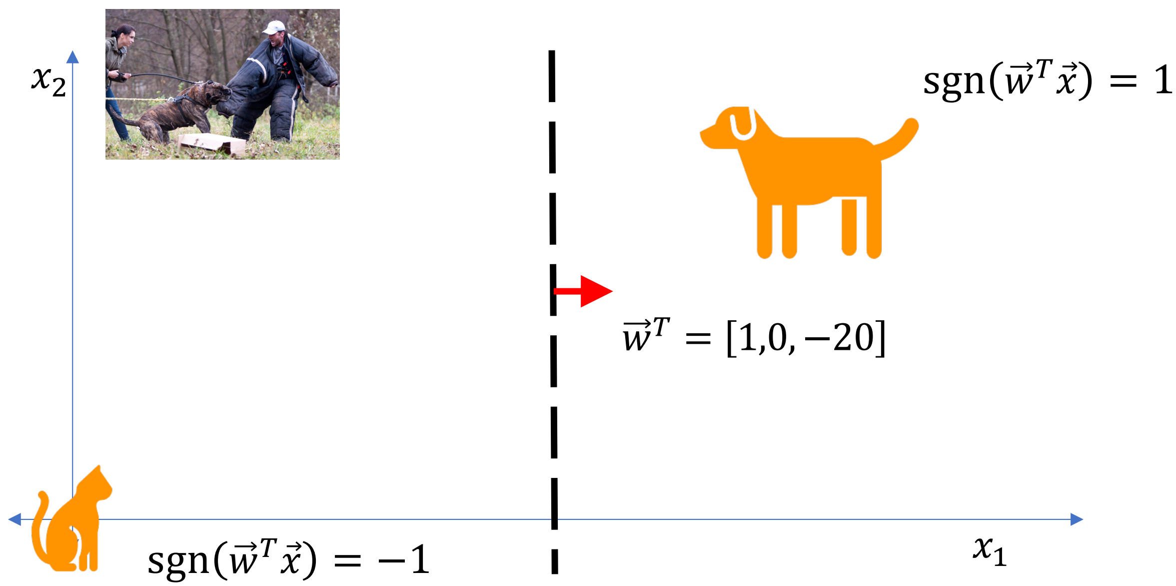
Perceptron training example: dogs vs. cats

- Let's start with the rule "if it comes when called (by at least 20 different people out of 40), it's a dog."
- Write that as an equation: $\hat{y} = \text{sgn}(x_1 - 20)$
- Write that as a vector equation: $\hat{y} = \text{sgn}(w^T x)$, where $w^T = [1,0,-20]$



Perceptron training example: dogs vs. cats

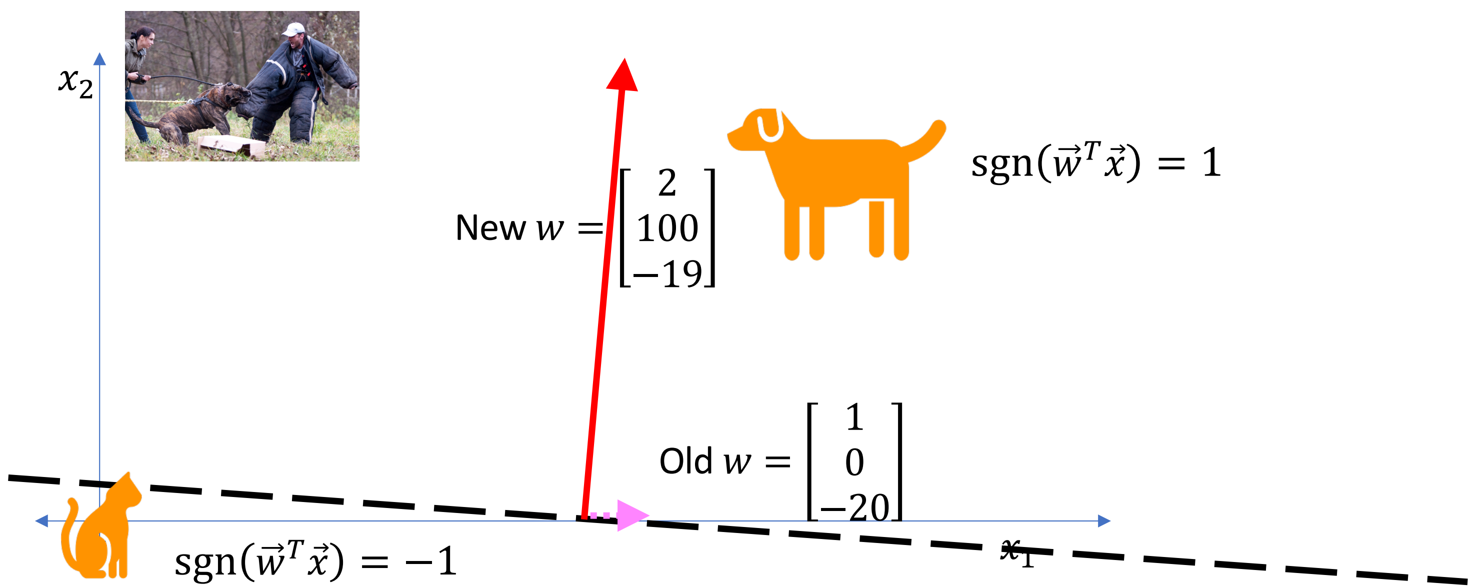
- The **Presa Canario** gets misclassified as a cat ($y = 1$, but $\hat{y} = -1$) because it only obeys its trainer, and nobody else ($x_1 = 1$).
- Though it rarely comes when called, is very large ($x_2 = 100$ pounds).
- $\vec{x}^T = [x_1, x_2, 1] = [1,100,1]$.



Perceptron training example: dogs vs. cats

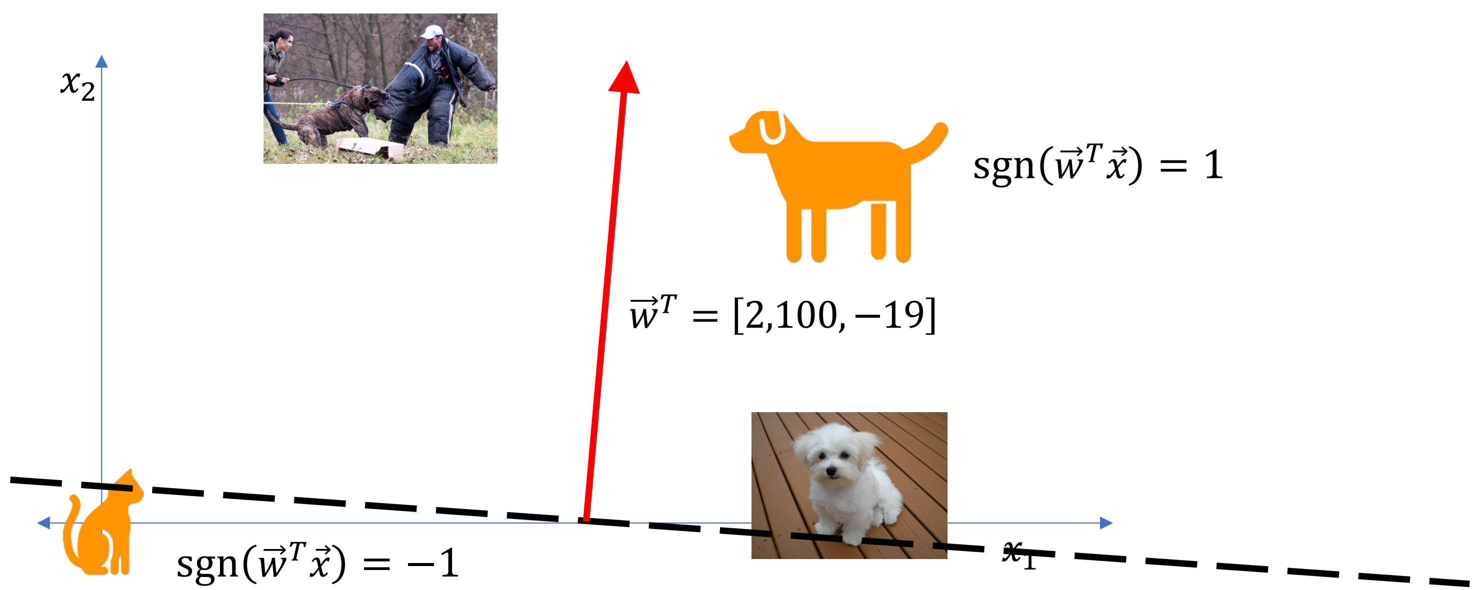
- The **Presa Canario** gets misclassified. $x^T = [x_1, x_2, 1] = [1,100,1]$.
- Perceptron learning rule: update the weights as:

$$w = w + yx = \begin{bmatrix} 1 \\ 0 \\ -20 \end{bmatrix} + 1\times \begin{bmatrix} 1 \\ 100 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 100 \\ -19 \end{bmatrix}$$



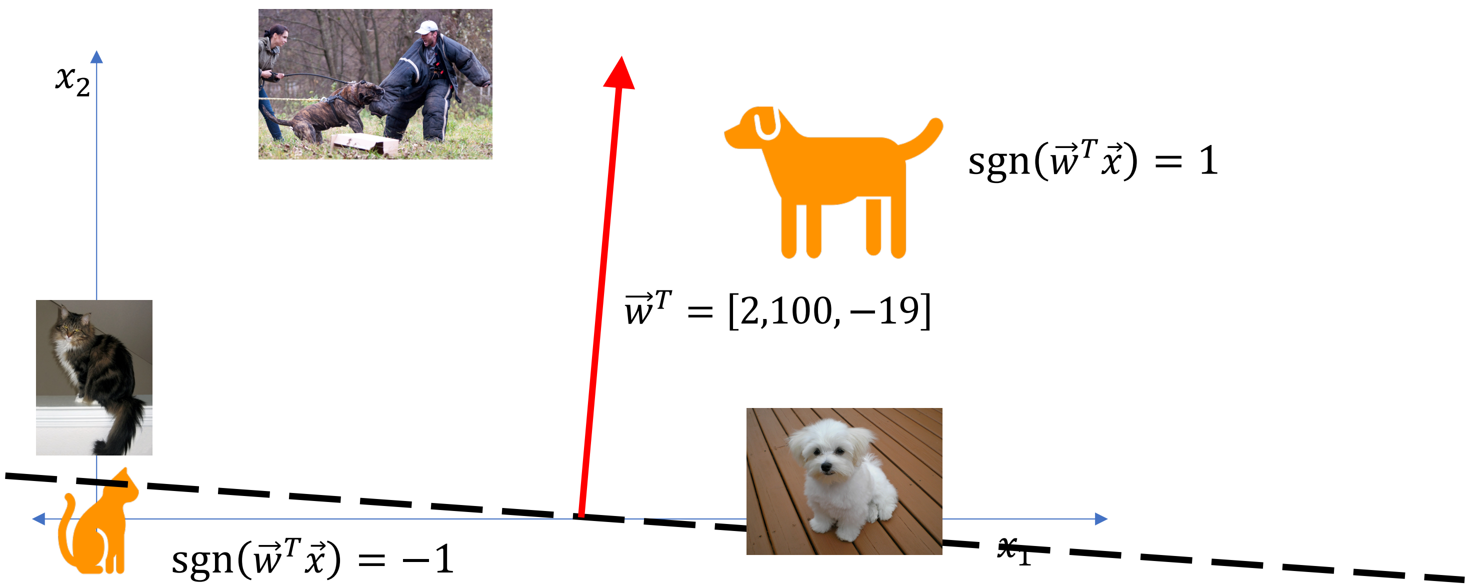
Perceptron training example: dogs vs. cats

- The **Maltese** is small ($x_2 = 10$ pounds) and very tame ($x_1 = 40$): $x = \begin{bmatrix} 40 \\ 10 \\ 1 \end{bmatrix}$.
- It's correctly classified! $\hat{y} = \text{sgn}(w^T x) = \text{sgn}(2\times 40 + 100\times 10 - 19) = +1$,
- so w is unchanged.



Perceptron training example: dogs vs. cats

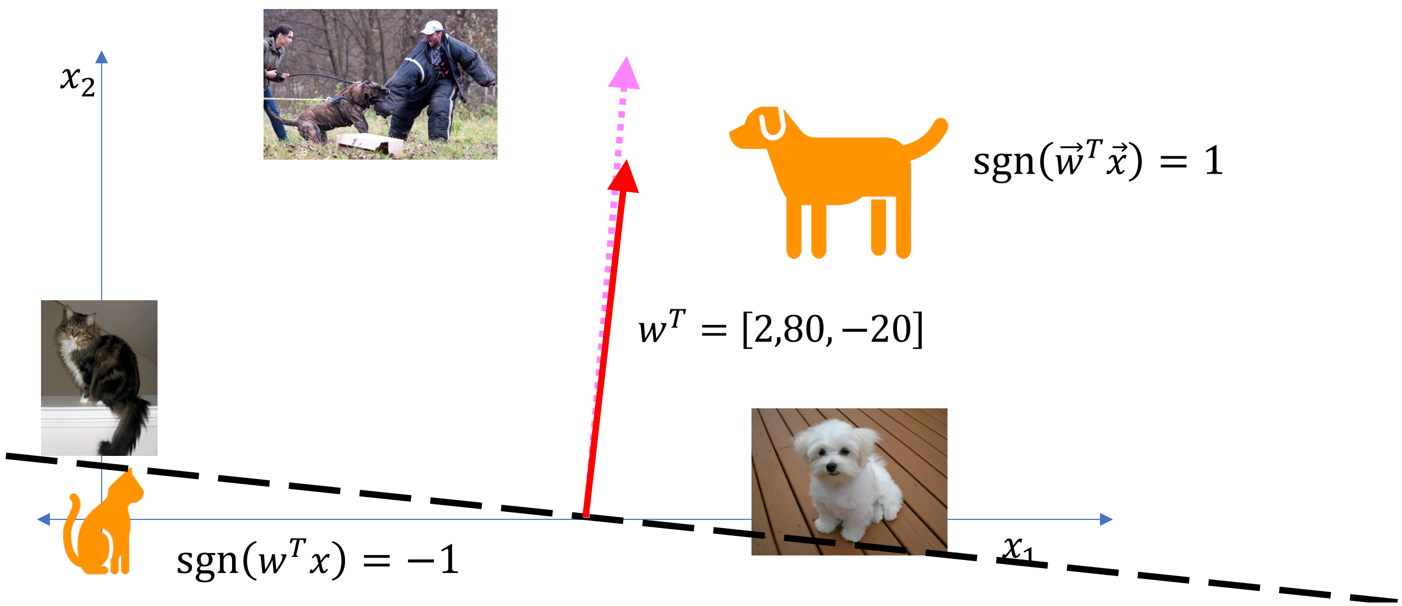
- The **Maine Coon** cat is big ($x_2 = 20$ pounds: $\vec{x} = [0,20,1]$), so it gets misclassified as a dog (true label is $y = -1$="cat," but the classifier thinks $\hat{y} = 1$="dog").



Perceptron training example: dogs vs. cats

- The **Maine Coon** cat is big ($x_2 = 20$ pounds: $\vec{x} = [0,20,1]$), so it gets misclassified, so we update w:

$$w = w + yx = \begin{bmatrix} 2 \\ 100 \\ -19 \end{bmatrix} + (-1)\times \begin{bmatrix} 0 \\ 20 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 80 \\ -20 \end{bmatrix}$$



Outline

- A little history: perceptron as a model of a biological neuron
- The perceptron learning algorithm
- Linear separability, yx, and convergence
    - It converges to the right answer, even with $\eta = 1$, if data are linearly separable
    - If data are not linearly separable, it's necessary to use $\eta = 1/n$.
- Multi-class perceptron

Does the perceptron find an answer?
Does it find the correct answer?

Suppose we run the perceptron algorithm for a very long time. Will it converge to an answer? Will it converge to the correct answer?

Does the perceptron find an answer? Does it find the correct answer?

Suppose we run the perceptron algorithm for a very long time. Will it converge to an answer? Will it converge to the correct answer?

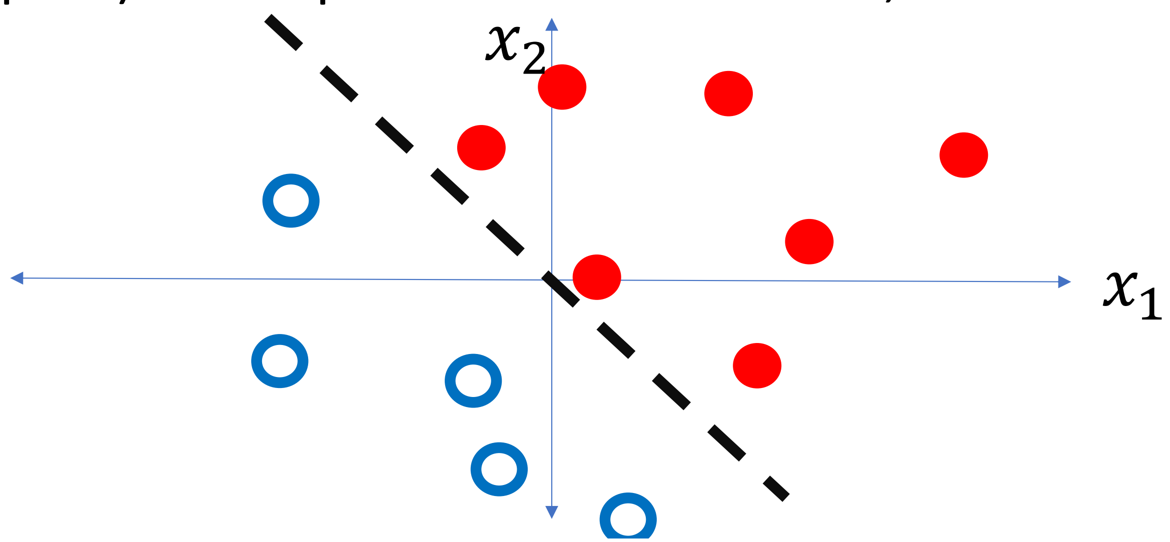
That depends on whether or not the data are linearly separable.

Does the perceptron find an answer? Does it find the correct answer?

Suppose we run the perceptron algorithm for a very long time. Will it converge to an answer? Will it converge to the correct answer?

That depends on whether or not the data are linearly separable.

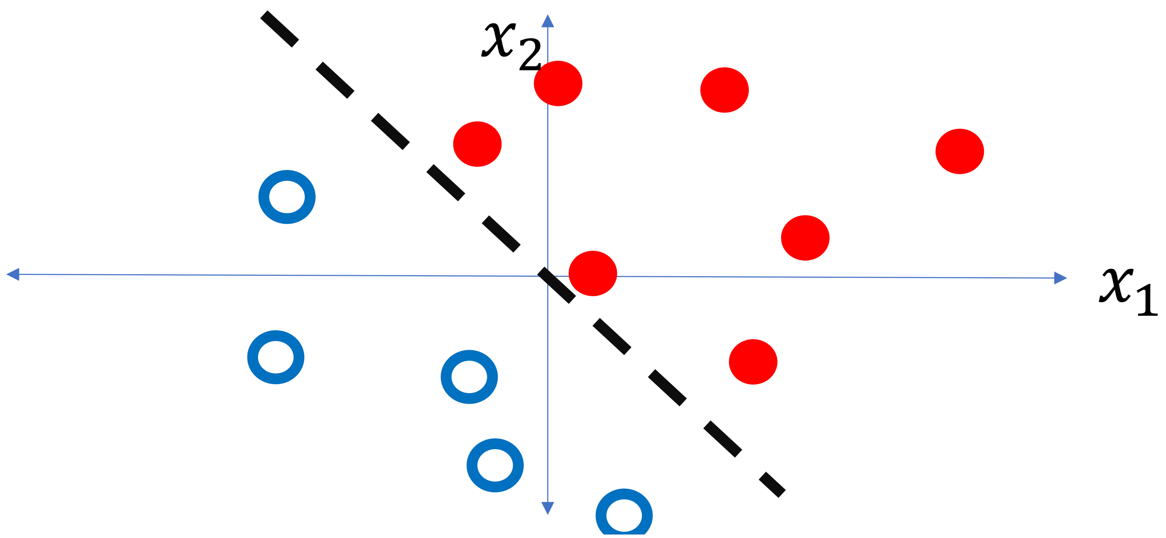
"Linearly separable" means it's possible to find a line (a hyperplane, in D-dimensional space) that separates the two classes, like this:



Does the perceptron find an answer? Does it find the correct answer?

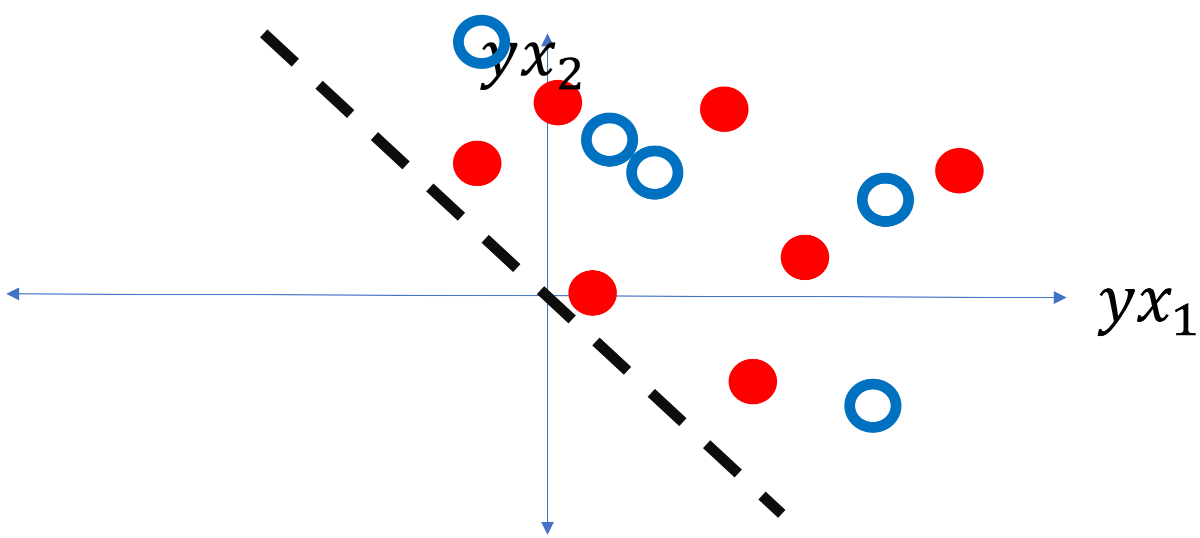
Suppose that, instead of plotting x, we plot yx.

- If $y = 1$, plot x
- If $y = -1$, plot $-x$



Does the perceptron find an answer? Does it find the correct answer?

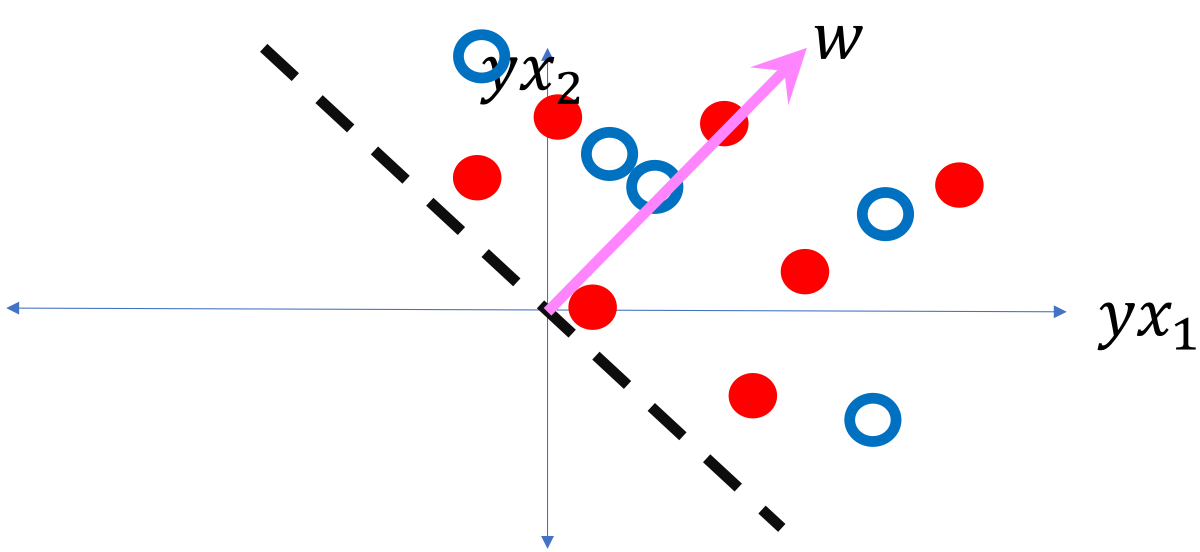
Notice that the original data (x) are linearly separable if and only if the signed data (yx) are all in the same half-plane.



Does the perceptron find an answer? Does it find the correct answer?

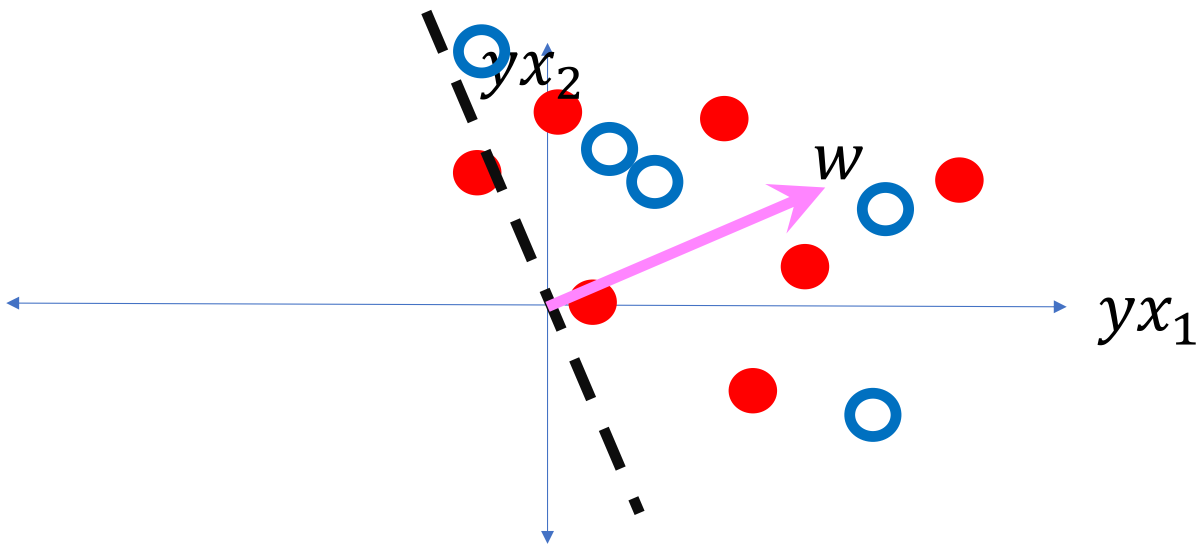
Notice that the original data (x) are linearly separable if and only if the signed data (yx) are all in the same half-plane.

That means that there is some vector w such that $\mathrm{sgn}(w^T(yx)) = 1$ for all of the data.



Does the perceptron find an answer?
Does it find the correct answer?

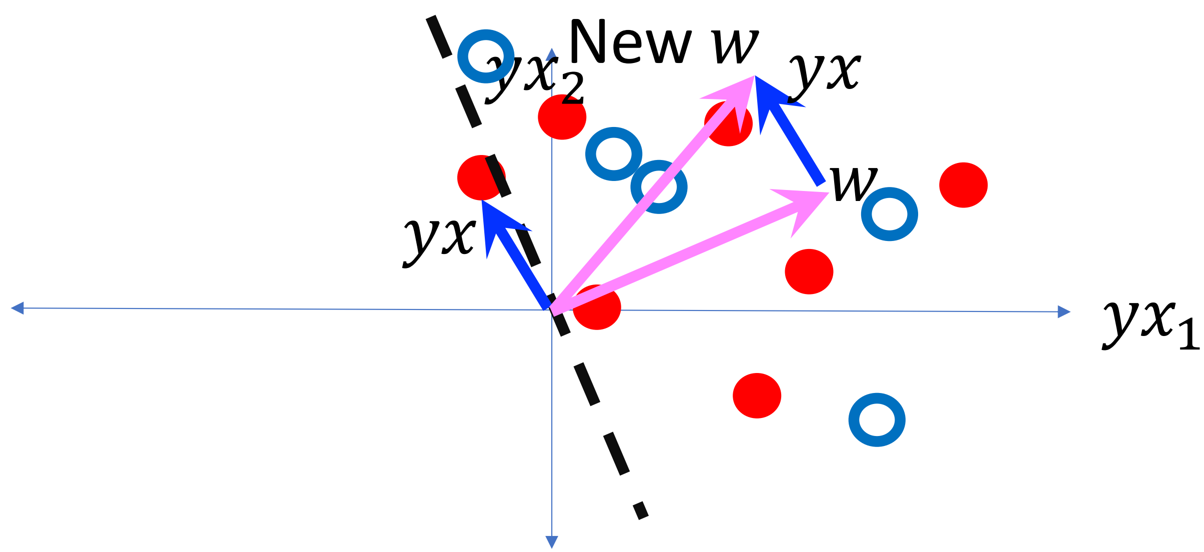
Suppose we start out with the wrong w, so that one token is misclassified.



Does the perceptron find an answer? Does it find the correct answer?

Suppose we start out with the wrong w, so that one token is misclassified. Then we update w as

$$w = w + yx$$

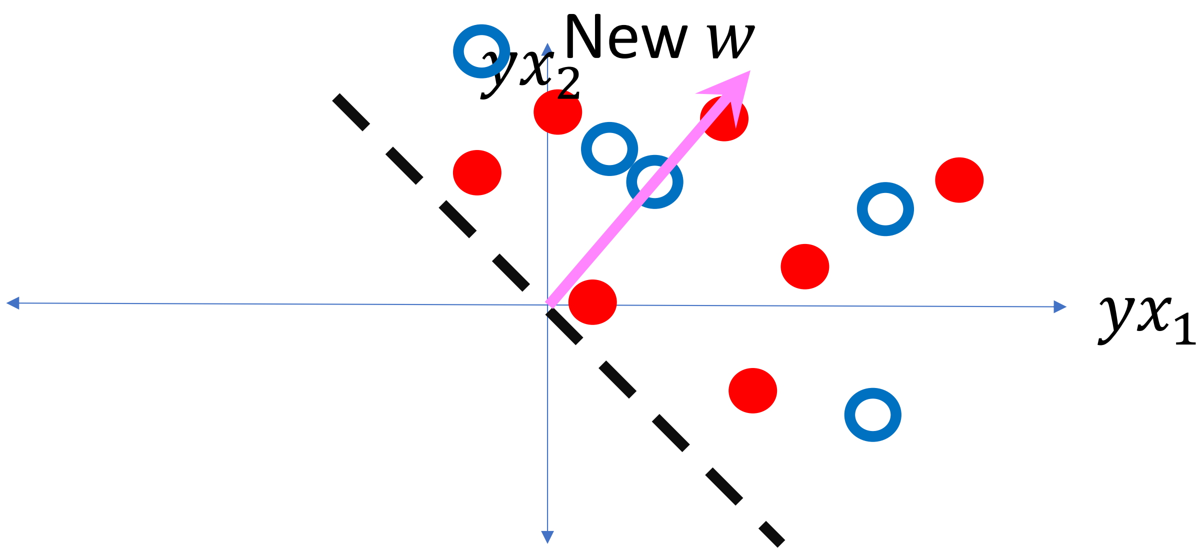


Does the perceptron find an answer? Does it find the correct answer?

Suppose we start out with the wrong w, so that one token is misclassified. Then we update w as

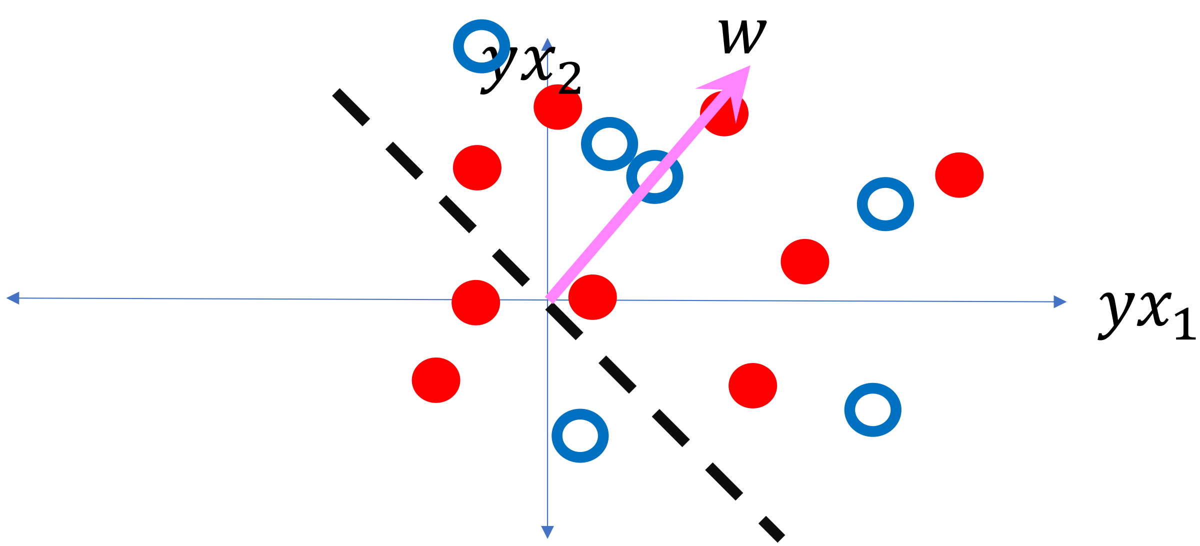
$$w = w + yx$$

…and the boundary moves so that the misclassified token is on the right side.



What if the data are not linearly separable?

… well, then in that case, the perceptron algorithm with $\eta = 1$ never converges. The only solution is to use a learning rate, η, that gradually decays over time, so that the update $\eta y x$ also gradually decays toward zero.



What about non-separable data?

- If the data are NOT linearly separable, then the perceptron with η=1 doesn't converge.
- In fact, that's what η is for.
- Remember that $w = w + \eta y x$.
- We can force the perceptron to stop wiggling around by forcing η (and therefore $\eta y\vec{x}$) to get gradually smaller and smaller.
- This works: for the n^{th} training token, set $\eta=\frac{1}{n}$.
- Notice: $\sum_{n=1}^{\infty}\frac{1}{n}$ is infinite. Nevertheless, $\eta=\frac{1}{n}$ works, because the yx tokens are not all in the same direction.

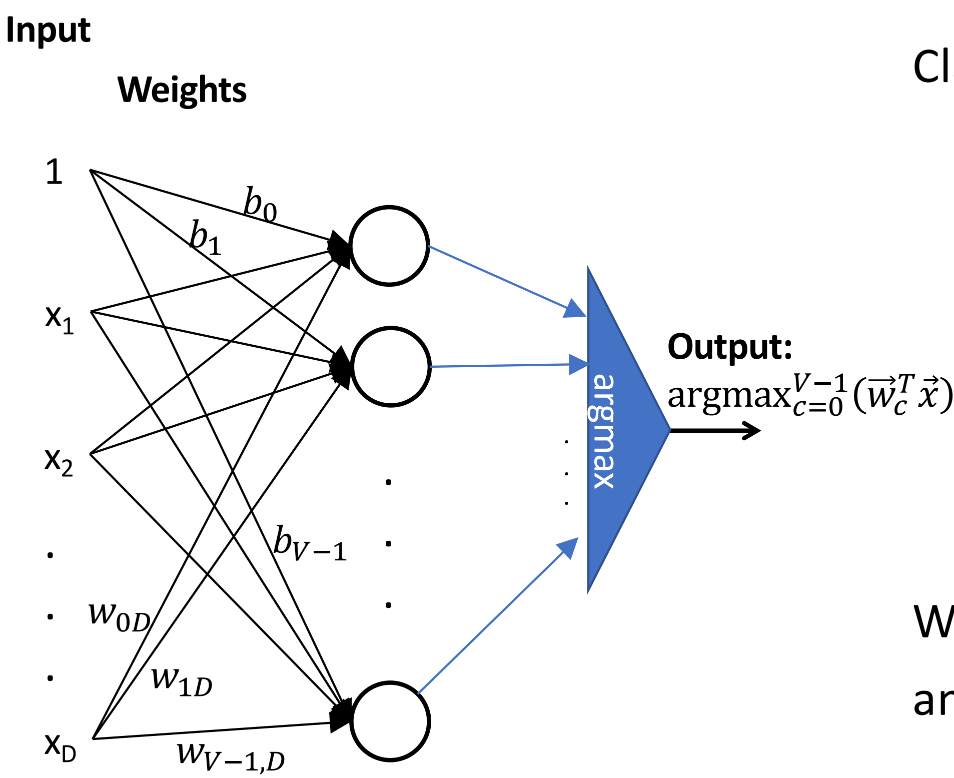
Outline

- A little history: perceptron as a model of a biological neuron
- The perceptron learning algorithm
- Linear separability, yx, and convergence
    - It converges to the right answer, even with $\eta = 1$, if data are linearly separable
    - If data are not linearly separable, it's necessary to use $\eta = 1/n$.
- Multi-class perceptron

Multi-Class Perceptron

True class is $y \in \{0,1,2,\ldots,V-1\}$

(i.e., V=vocabulary size = # of distinct classes).



Classifier output is

$$\hat{y} = \text{argmax}_{c=0}^{V-1}(w_{c1}x_1 + \cdots + w_{cD}x_D + b_c)$$

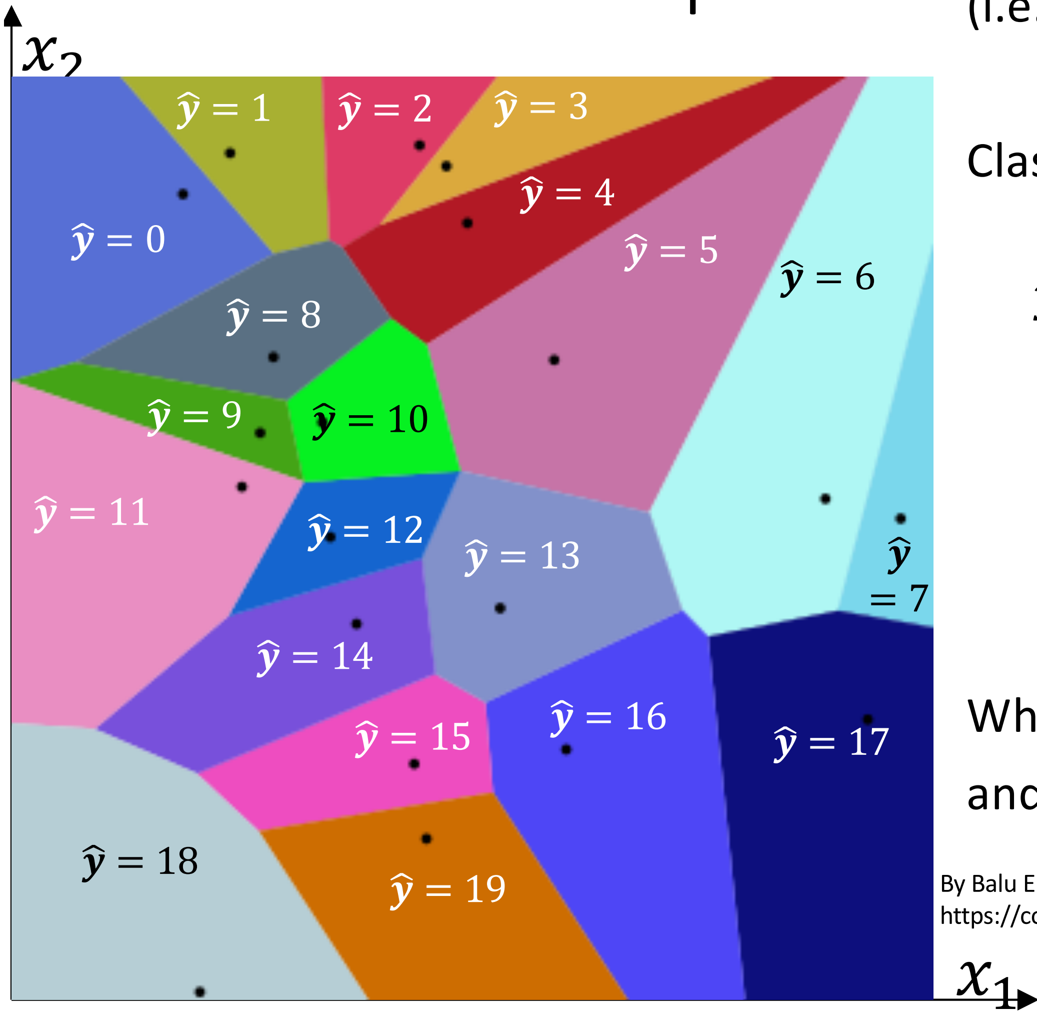
$$= \text{argmax}_{c=0}^{V-1}(w_c^T x)$$

$$\in \{0,1,\ldots,V-1\}$$

Where $w_c = [w_{c1},\ldots,w_{cD},b_c]^T$

and $x = [x_1,\ldots,x_D,1]^T$

Multi-Class Perceptron



True class is $y \in \{0,1,2,\ldots,V-1\}$
(i.e., V=vocabulary size = # of distinct classes).

Classifier output is

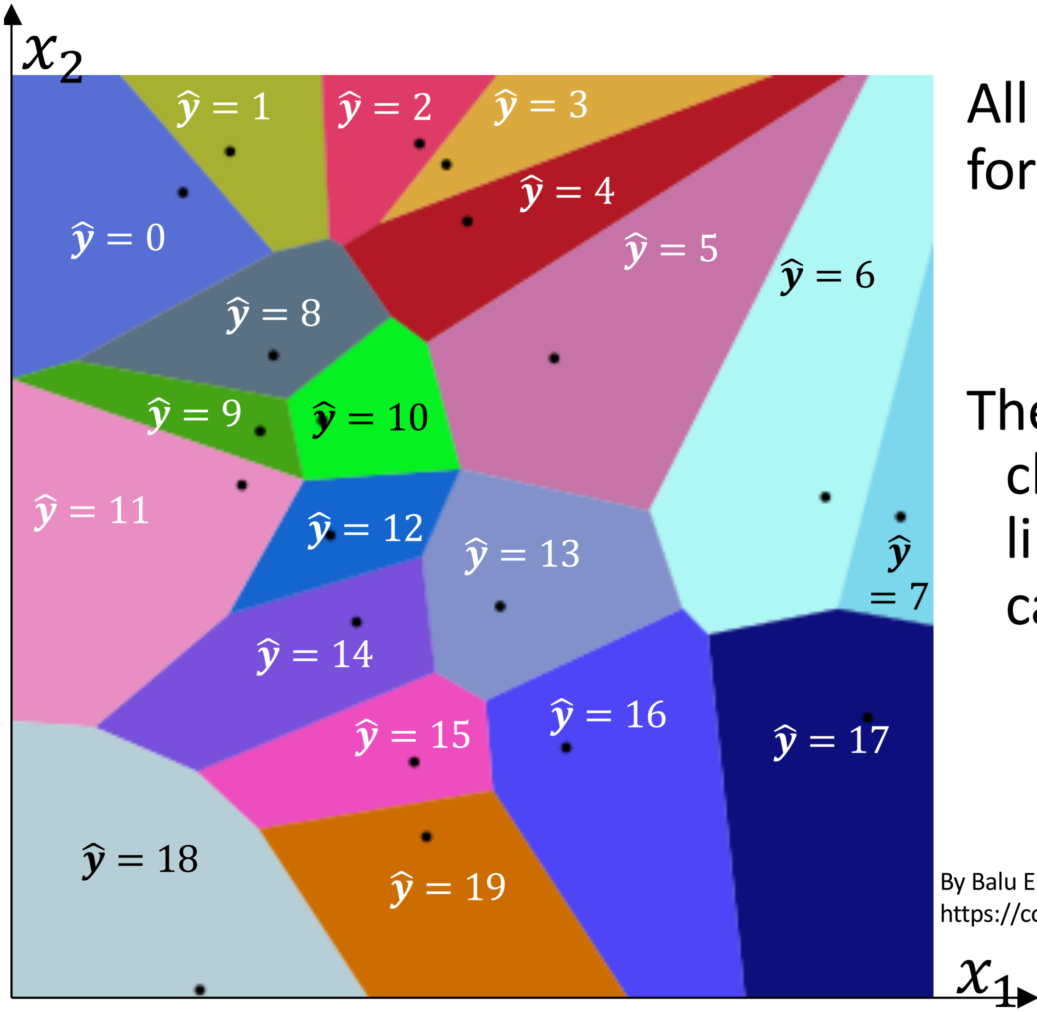
$$\hat{y} = \text{argmax}_{c=0}^{V-1}(w_{c1}x_1 + \cdots + w_{cD}x_D + b_c)$$

$$= \text{argmax}_{c=0}^{V-1}(w_c^T x)$$

$$\in \{0,1,\ldots,V-1\}$$

Where $w_c = [w_{c1},\ldots,w_{cD},b_c]^T$
and $x = [x_1,\ldots,x_D,1]^T$

Multi-Class Linear Classifiers



All multi-class linear classifiers have the form

$$\hat{y} = \text{argmax}_{c=0}^{V-1}(w_c^T x)$$

The region of x-space associated with each class label is convex with piece-wise linear boundaries. Such regions are called "Voronoi regions."

Training a Multi-Class Perceptron

For each training instance $\boldsymbol{x}$ w/ground truth label $y \in \{0,1,\ldots,V-1\}$:

- Classify with current weights: $\hat{y} = \mathrm{argmax}_{c=0}^{V-1}\left(w_c^T x\right)$
- Update weights:
  - if $\hat{y}$ is correct ($y = \hat{y}$) then do nothing
  - If $\hat{y}$ is incorrect ($y \neq \hat{y}$) then:
    - Update the correct-class vector as $w_y = w_y + \eta x$
    - Update the wrong-class vector as $w_{\hat{y}} = w_{\hat{y}} - \eta x$
    - **Don't change the vectors of any other class**

Conclusions

- Perceptron as a model of a biological neuron: $\hat{y} = \mathrm{sgn}(w^T x)$
- The perceptron learning algorithm: if $y = \hat{y}$ then do nothing, else $w = w + \eta \mathrm{y} x$.
- Linear separability, yx, and convergence
  - It converges to the right answer, even with $\eta = 1$, if data are linearly separable
  - If data are not linearly separable, it's necessary to use $\eta = 1/n$.
- Multi-class perceptron: if $y = \hat{y}$ then do nothing, else $w_y = w_y + \eta x$, and $w_{\hat{y}} = w_{\hat{y}} - \eta x$.