CS 360: Machine Learning

Sara Mathieson, Sorelle Friedler

Spring 2024

HAVERFORD
COLLEGE

Admin

- **Lab 6** posted, due **Monday March 25**
  - Note: deadline is wrong on github classroom
- In lab today: get everything set up on the lab machines, import and view data
- **Thursday**: Going over Midterm 1

Languages Left Behind:

Why Language Models Fail at Non-English Content Moderation

Aliya Bhatia
Policy Analyst
Free Expression Project

Gabriel Nicholas
Research Fellow

Center for Democracy & Technology

Tuesday
March 26, 2024
4:30pm in Lutnick 200
and on Zoom

HAVERFORD
LIBRARIES

Outline for March 19

- Perceptron Algorithm
- Informal check-in
- Handout 15 example
- Introduction Support Vector Machines

Outline for March 19

- Perceptron Algorithm
- Informal check-in
- Handout 15 example
- Introduction to Support Vector Machines

Perceptron as a neural network

Biological model of a neuron

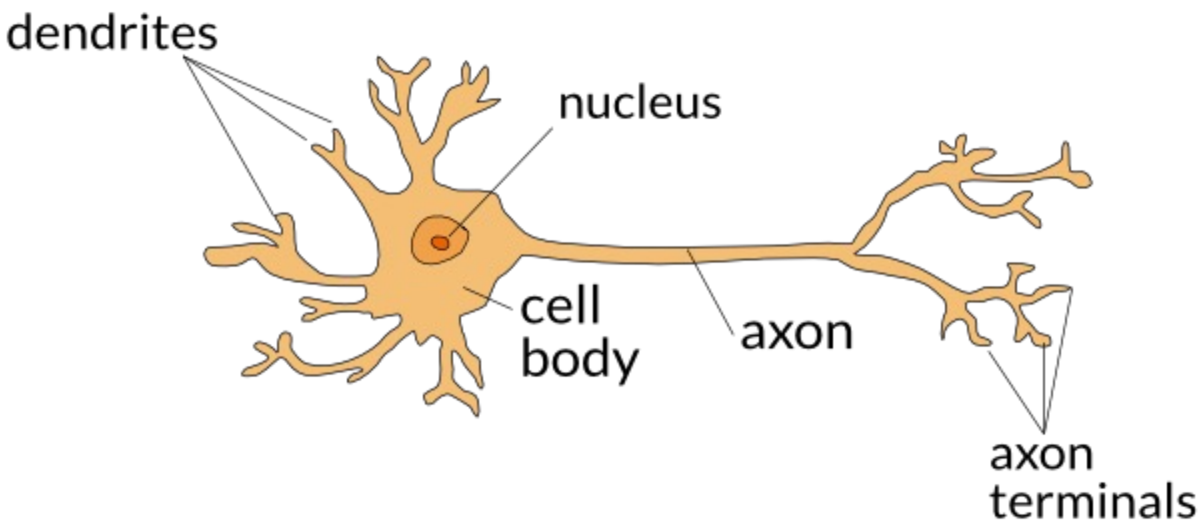


Image: modified from “Applied Go”

Perceptron as a neural network

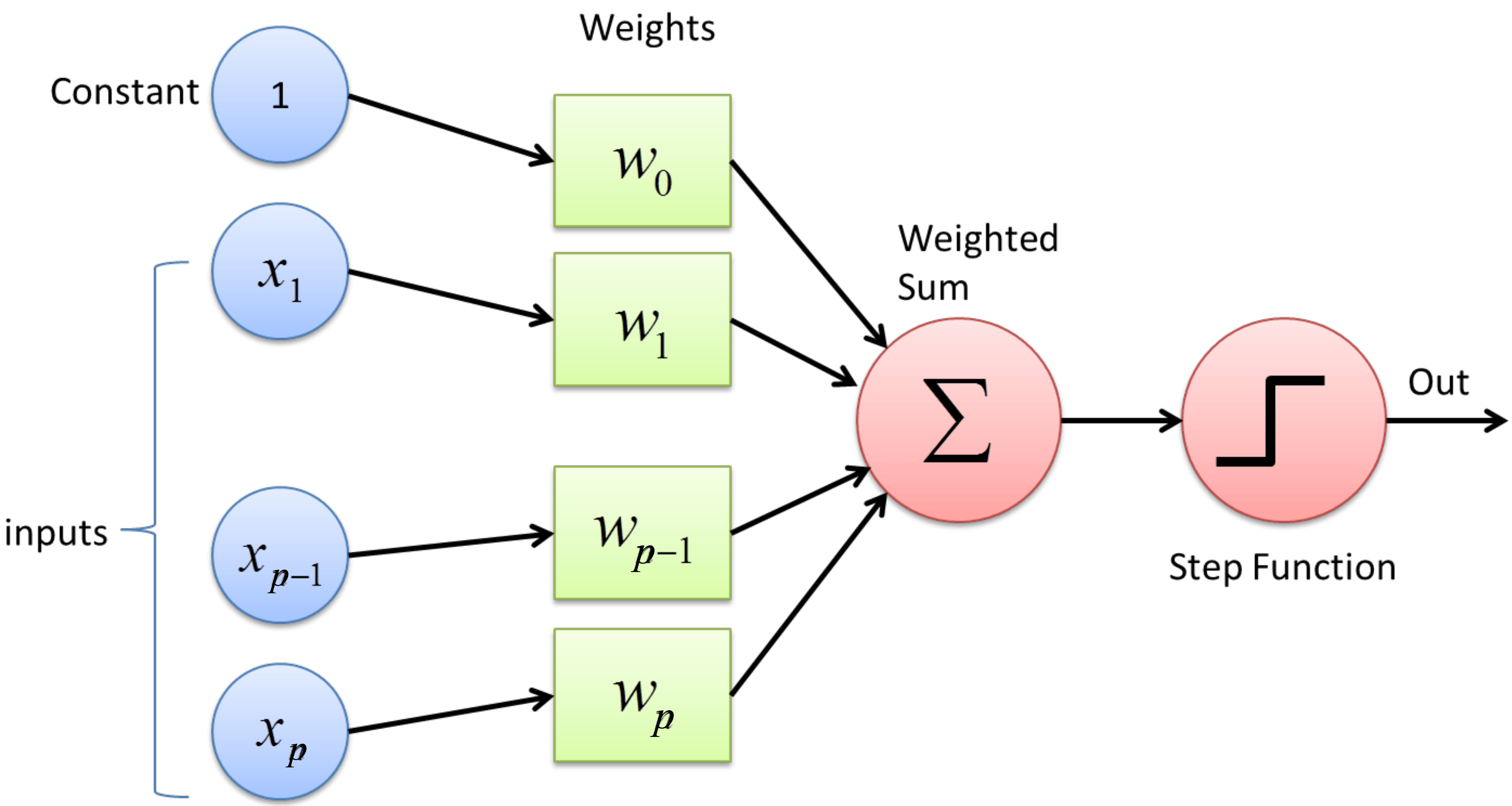


Image: modified from "Towards Data Science"

History of the Perceptron

- Invented in 1957 by Frank Rosenblatt

- Initially thought to be the "solution to AI"

  NYT said the perceptron was "*the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence*"

- Famous book "Perceptrons" by Marvin Minsky and Seymour Papert (1969)

- Confusion about the text contributed to first "AI winter"

Hyperplane divides space into positive (+1) and negative (-1)

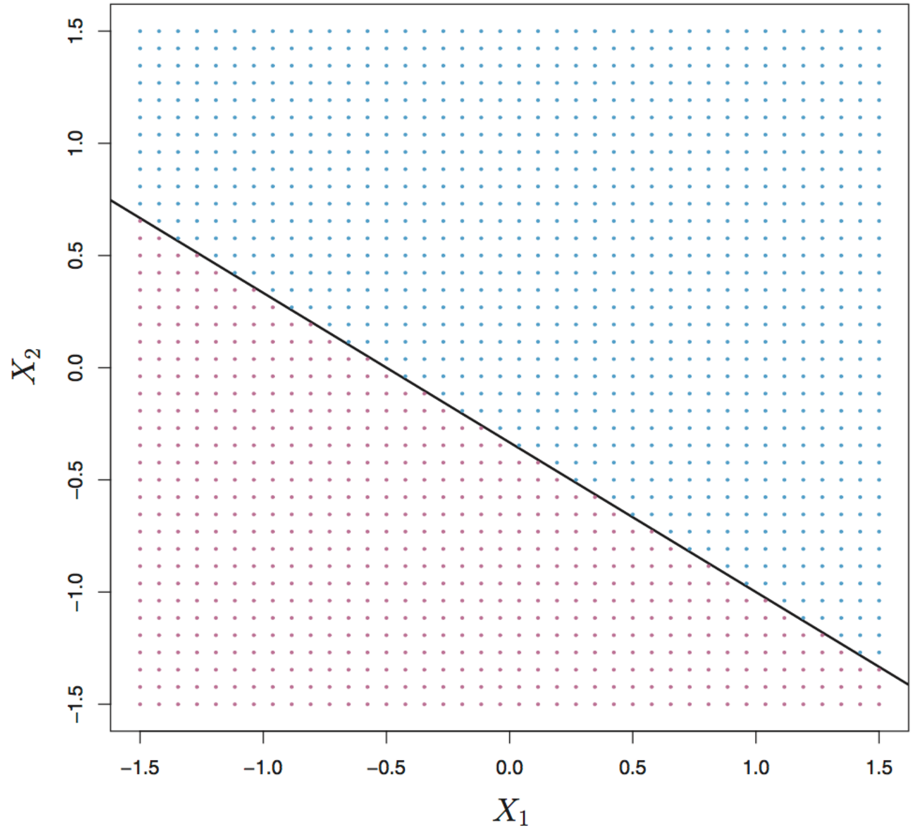


FIGURE 9.1. *The hyperplane* $1 + 2X_1 + 3X_2 = 0$ *is shown. The blue region is the set of points for which* $1 + 2X_1 + 3X_2 > 0$, *and the purple region is the set of points for which* $1 + 2X_1 + 3X_2 < 0$.

ISL: Figure 9.1

Goal: use training data to create a *separating* hyperplane

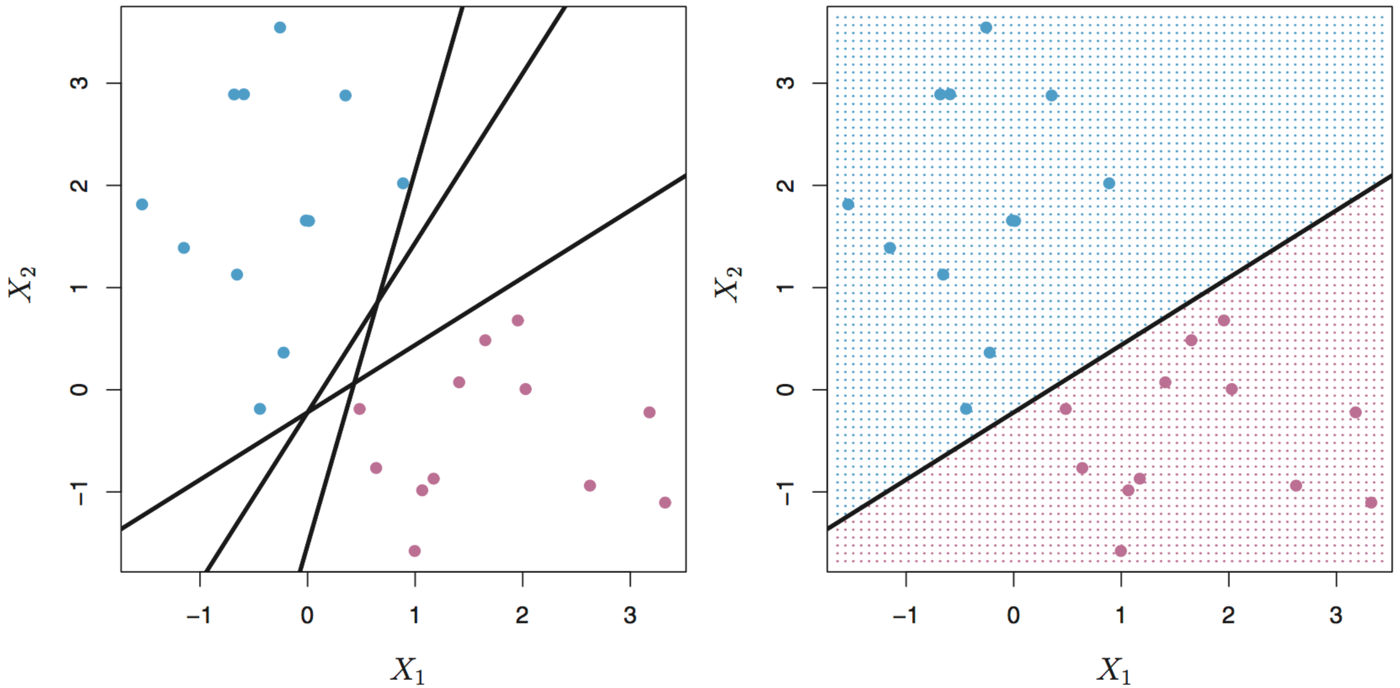


FIGURE 9.2. Left: *There are two classes of observations, shown in blue and in purple, each of which has measurements on two variables. Three separating hyperplanes, out of many possible, are shown in black.* Right: *A separating hyperplane is shown in black. The blue and purple grid indicates the decision rule made by a classifier based on this separating hyperplane: a test observation that falls in the blue portion of the grid will be assigned to the blue class, and a test observation that falls into the purple portion of the grid will be assigned to the purple class.*

ISL: Figure 9.2

These two hyperplanes would likely perform very differently on test data, but they both separate the training data

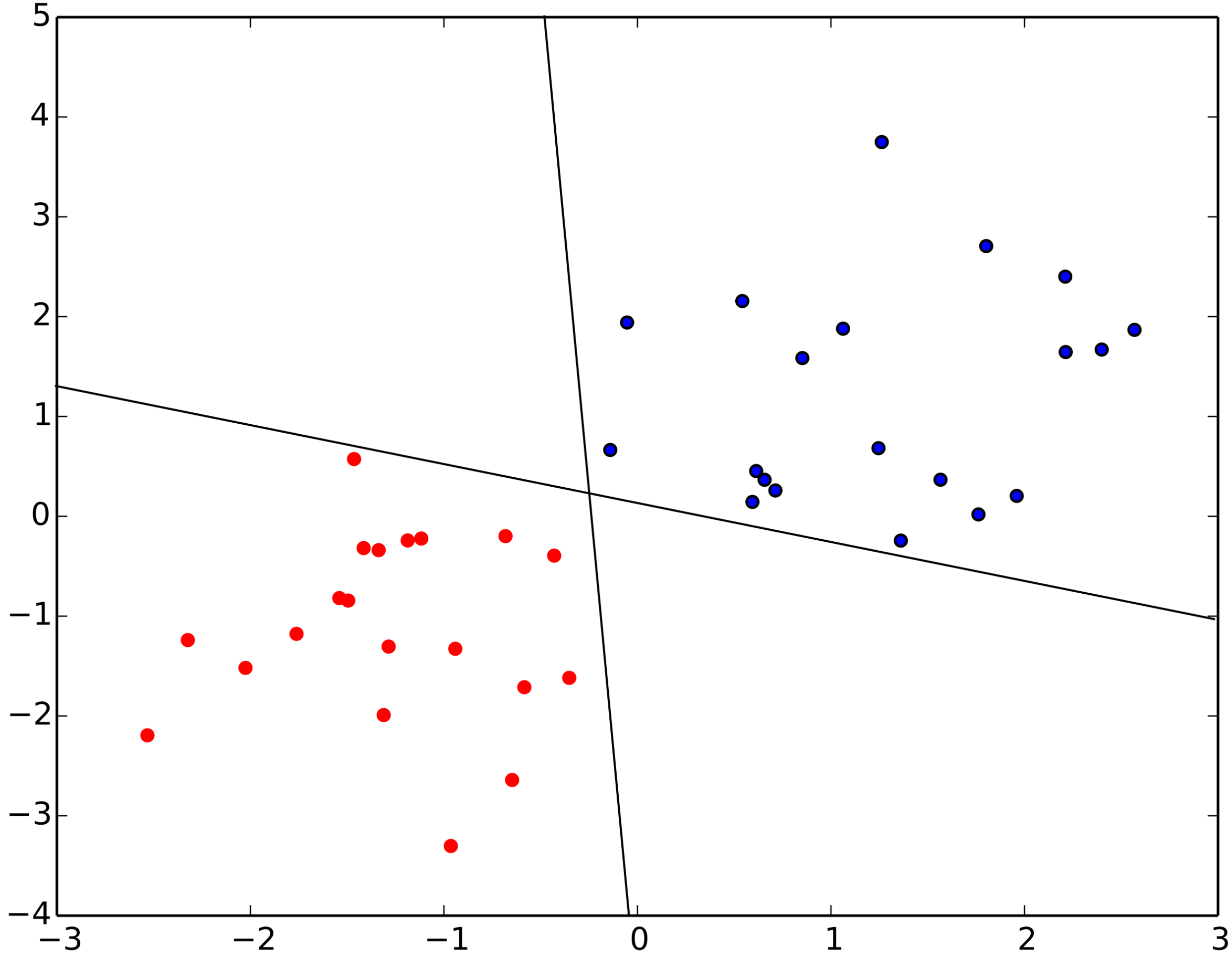
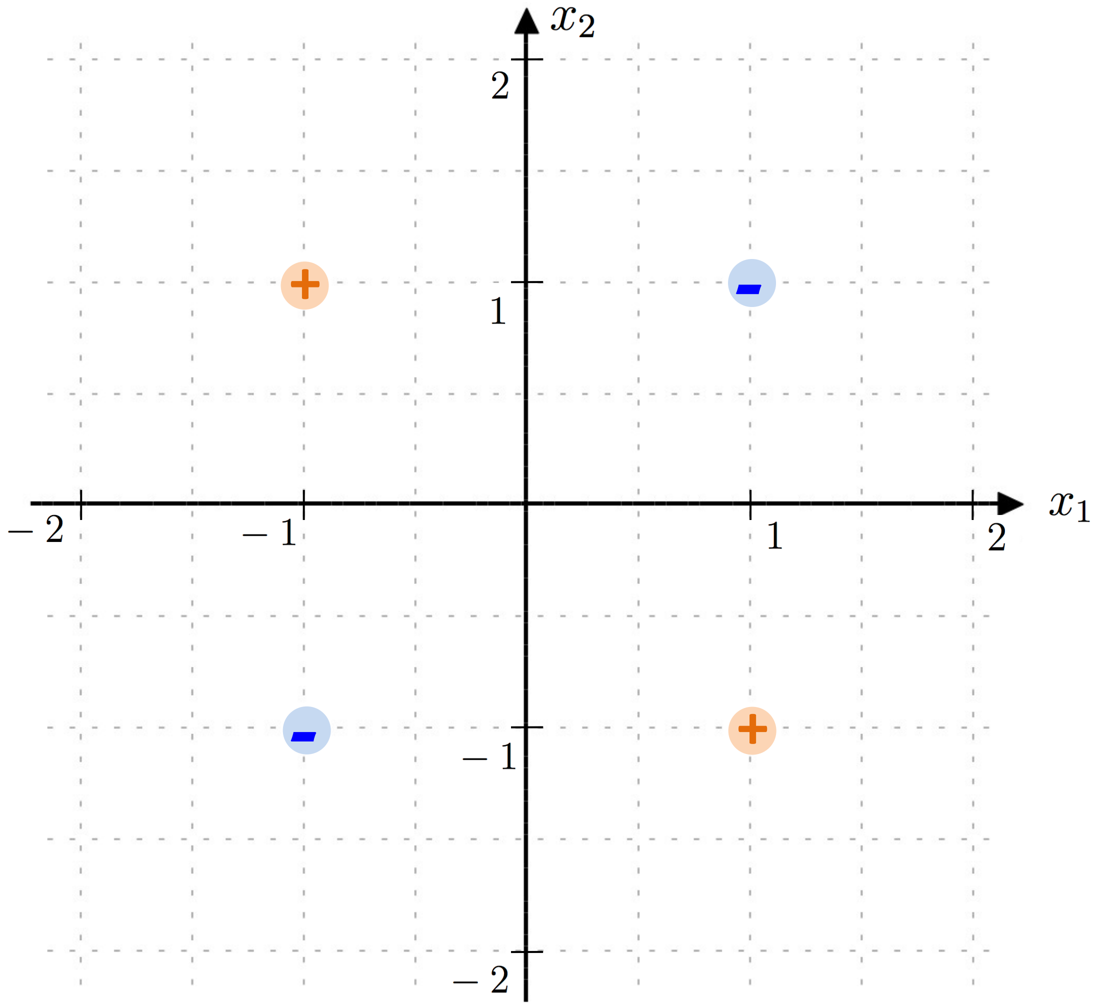


Image: Qwertyus, Wikipedia

Perceptron cannot learn XOR

(x_1 = 1 or x_2 = 1, but not both)

Why?

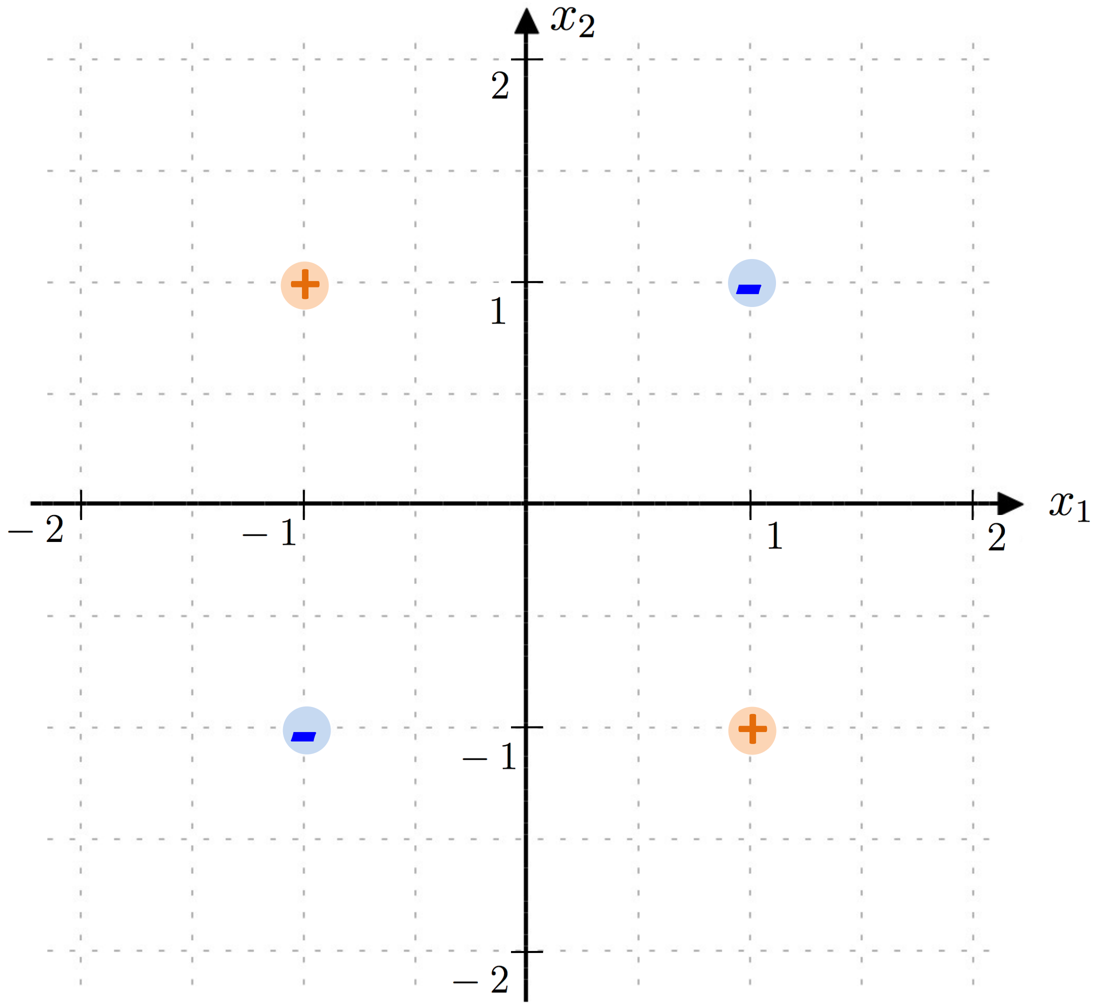


Perceptron cannot learn XOR

(x_1 = 1 or x_2 = 1, but not both)

Why?

Not linearly separable!



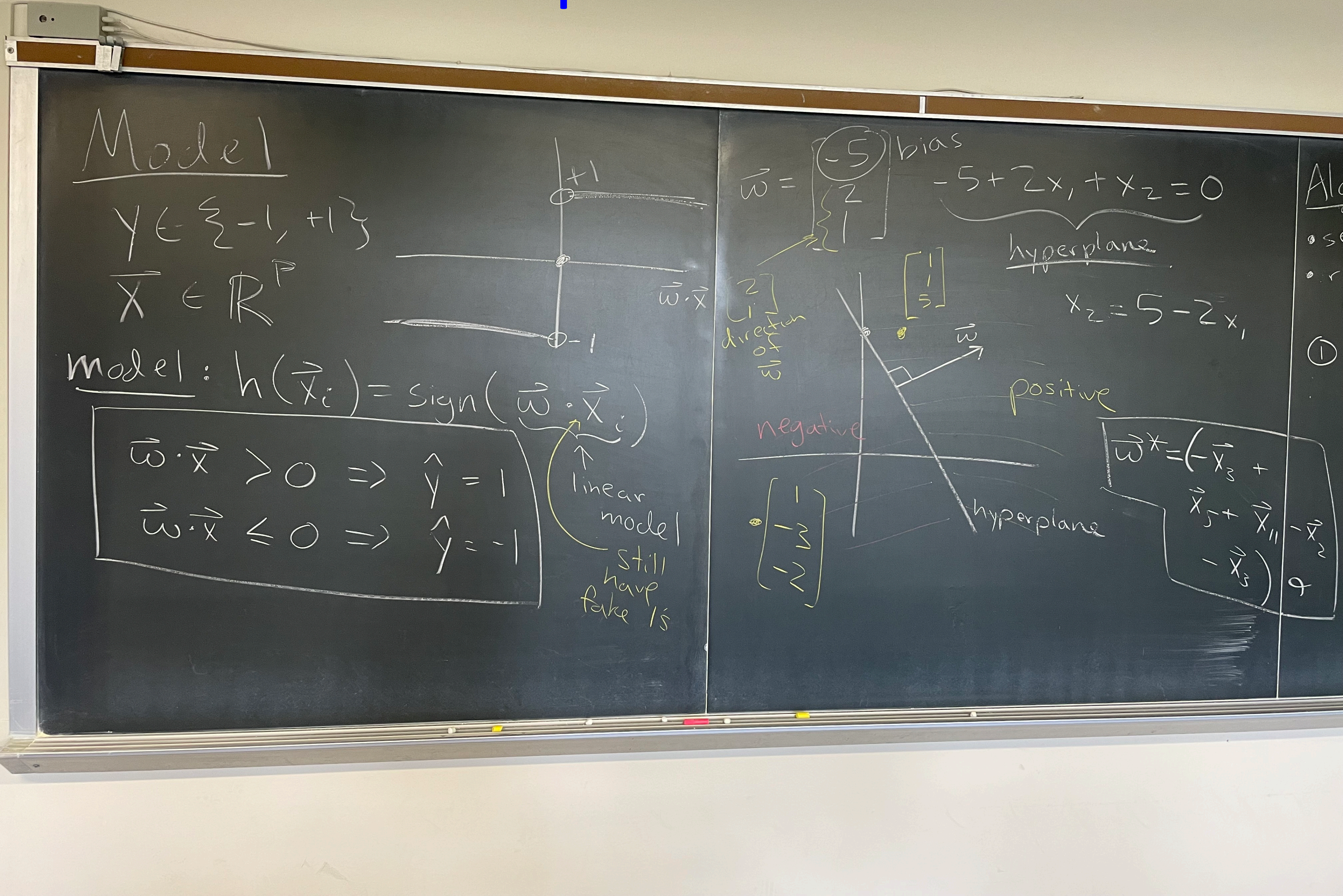
Convergence Guarantee

- Perceptron is guaranteed to converge to a solution if a separating hyperplane exists

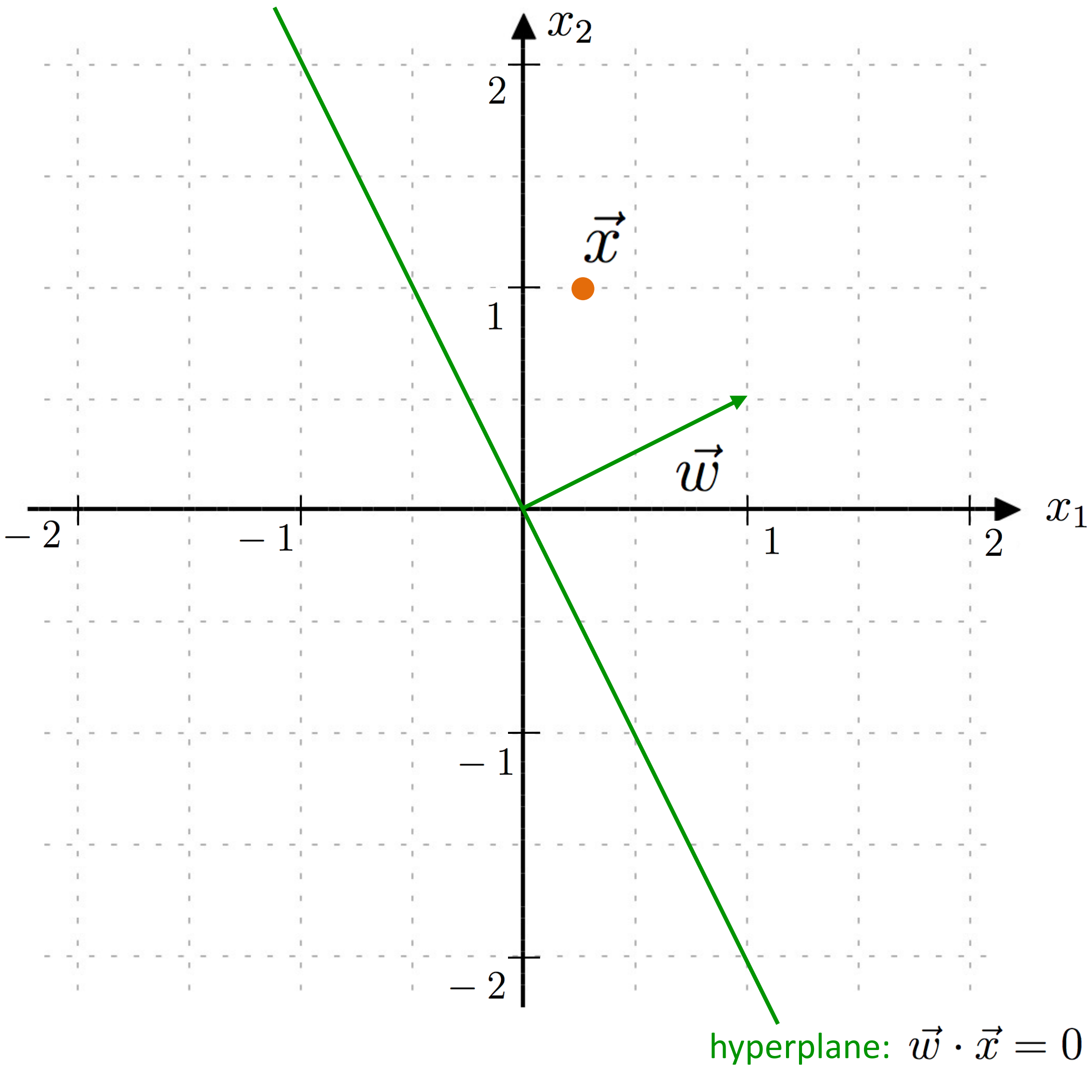
- Not guaranteed to converge to a "good" solution

- No guarantees about behavior if a separating hyperplane does not exist!

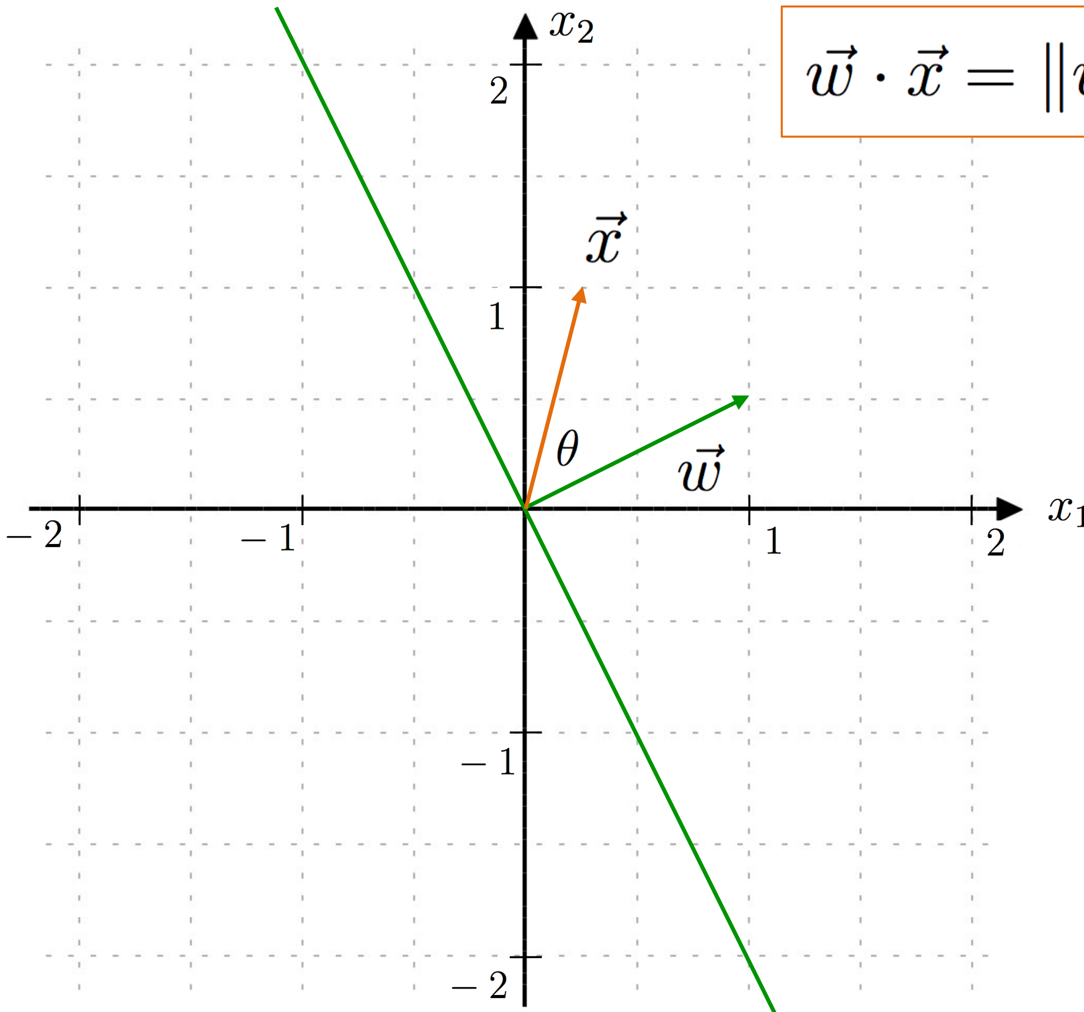
Perceptron Model



Intuition behind the dot product



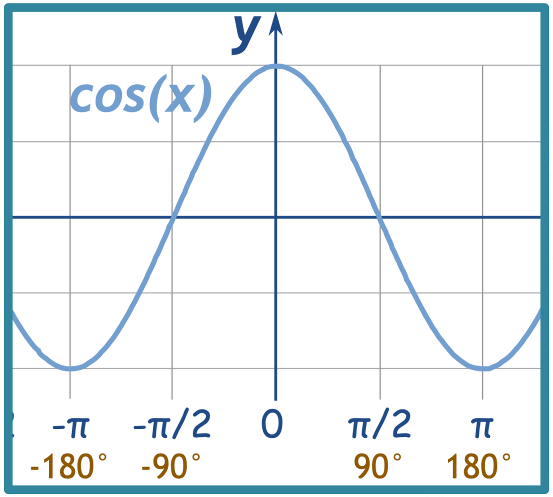
Intuition behind the dot product



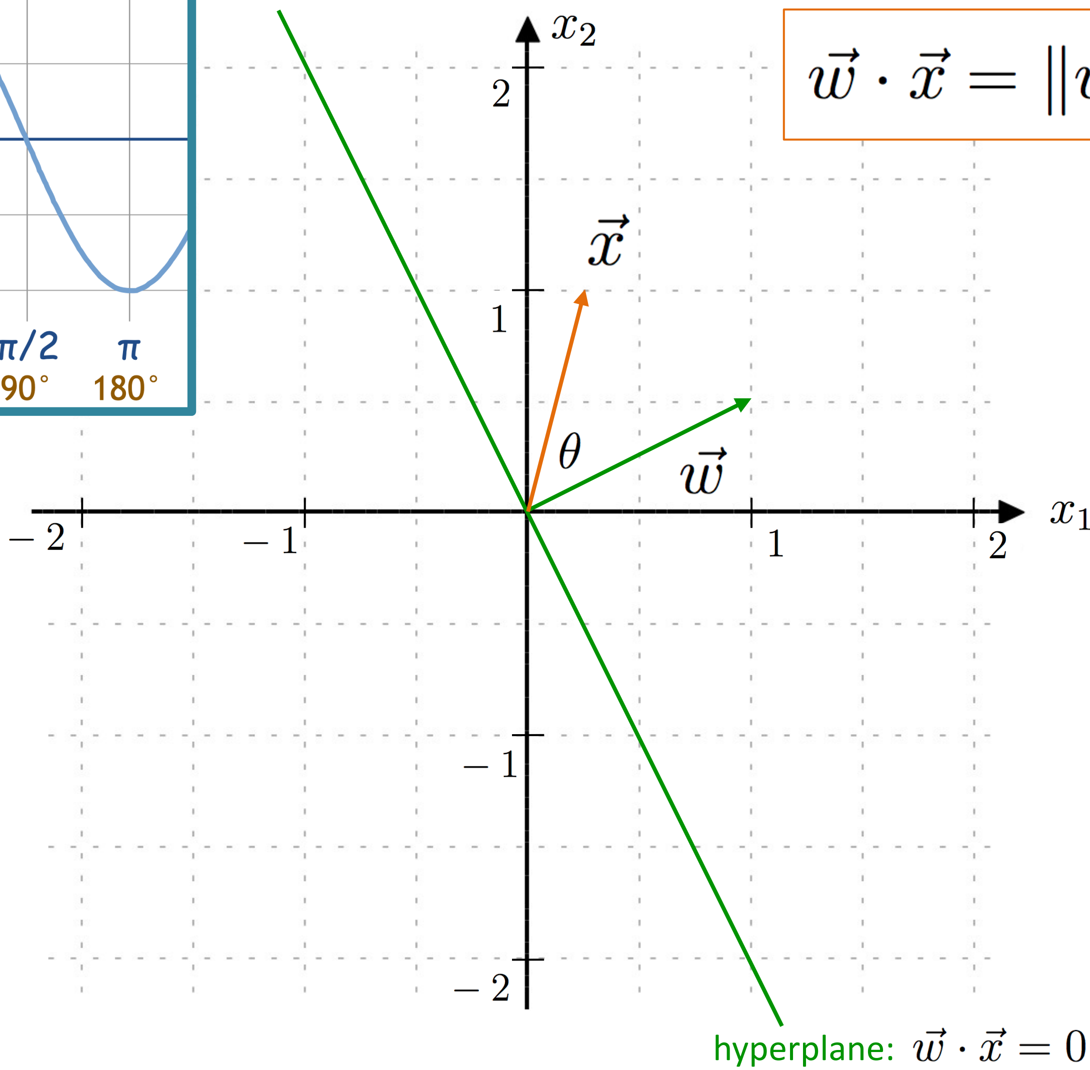
$$\vec{w} \cdot \vec{x} = \|\vec{w}\| \|\vec{x}\| \cos\theta$$

hyperplane: $\vec{w} \cdot \vec{x} = 0$

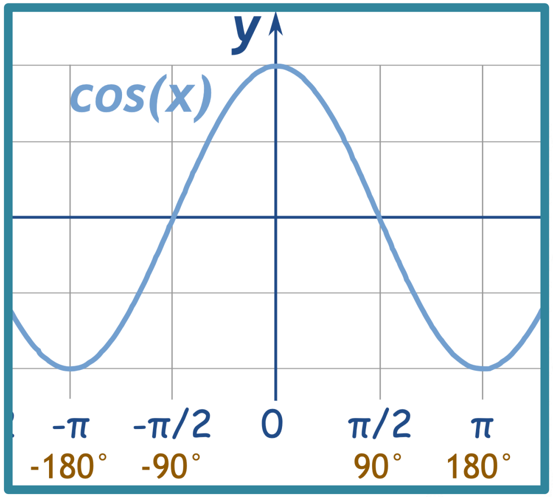
Intuition behind the dot product



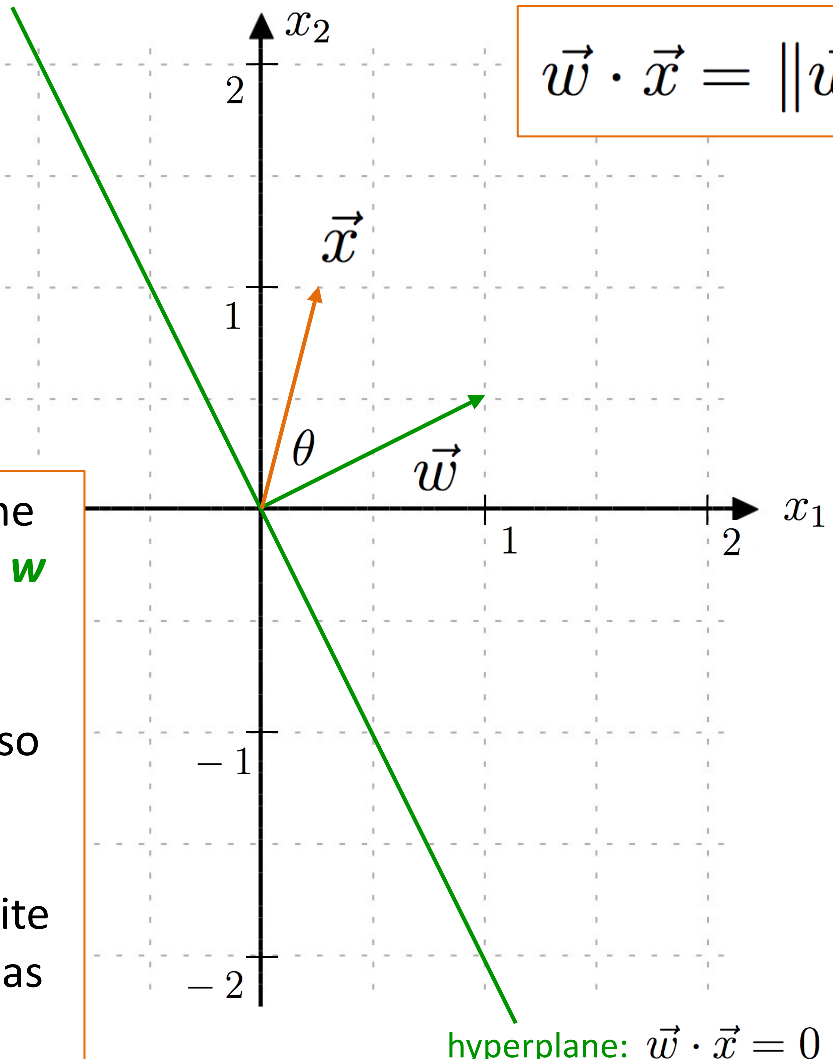
$$\vec{w} \cdot \vec{x} = \|\vec{w}\| \|\vec{x}\| \cos \theta$$



Intuition behind the dot product



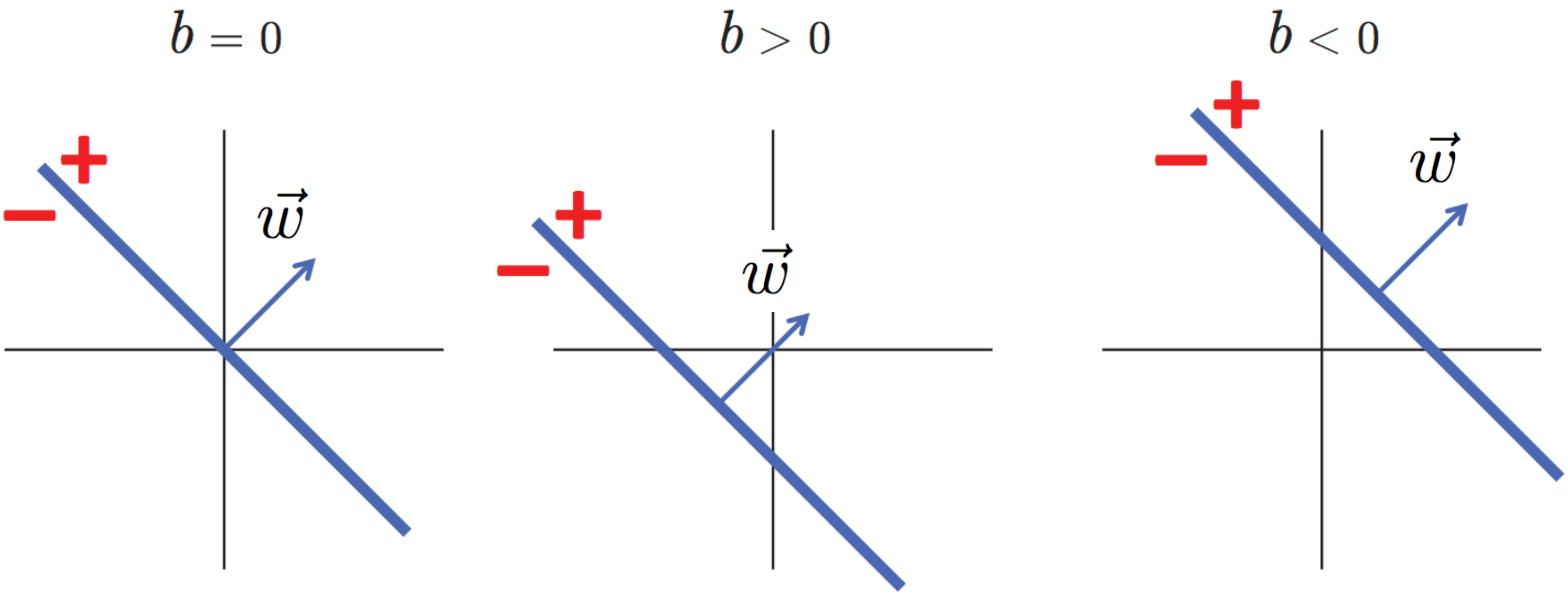
$$\vec{w} \cdot \vec{x} = \|\vec{w}\| \|\vec{x}\| \cos \theta$$



Takeaway: we only care about the sign of the angle between ***x*** and ***w***

- If cos θ > 0, ***x*** is on the same side of the hyperplane as ***w***, so we classify it as positive
- If cos θ < 0, ***x*** is on the opposite side from ***w***, so we classify it as negative

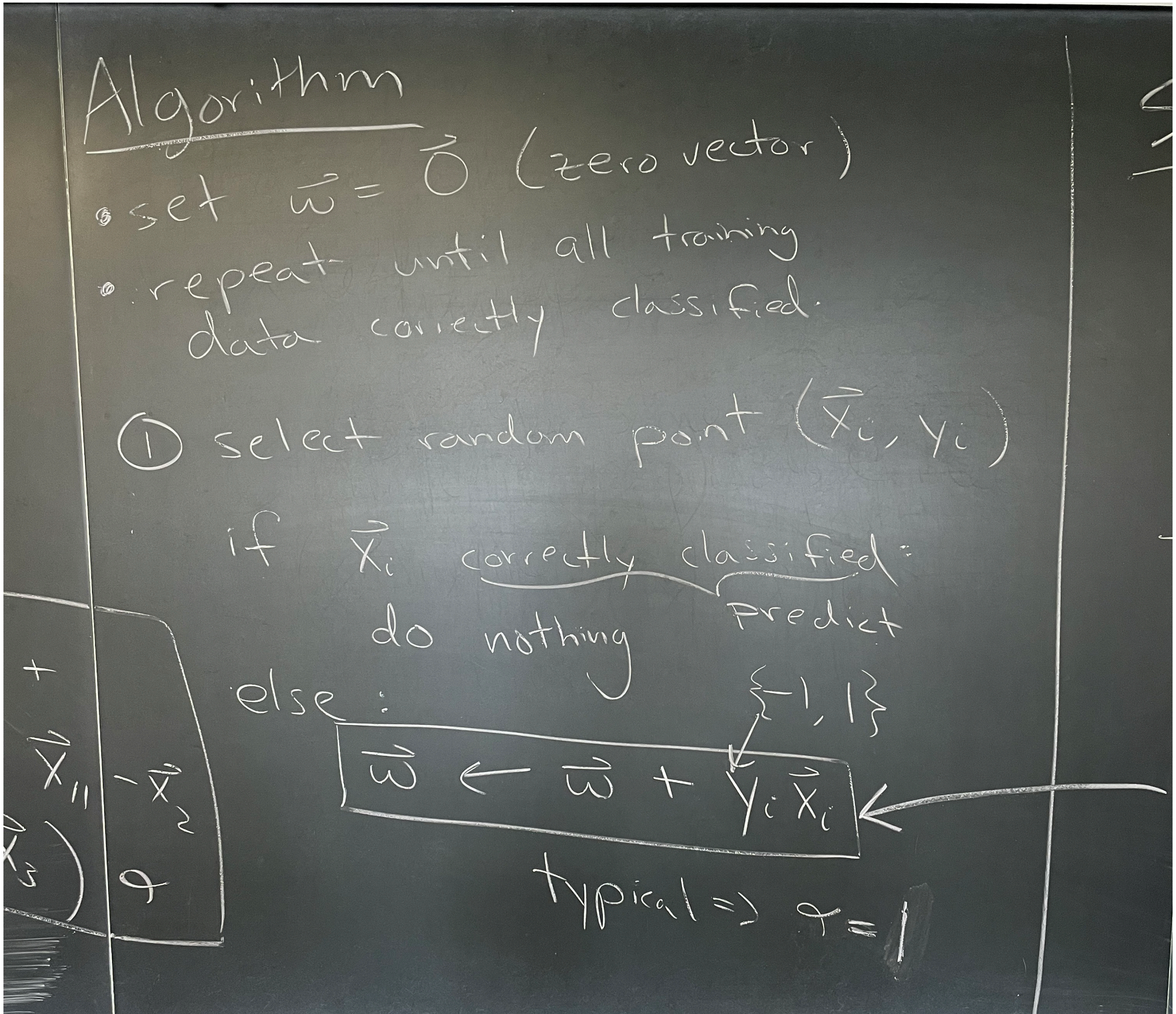
The ***bias*** (*b*) and the *y*-intercept are different, but they both capture a "shift" away from the origin.



With p=2, if w_2 is positive, then the above example holds

Example modified from Jessica Wu

Perceptron Algorithm

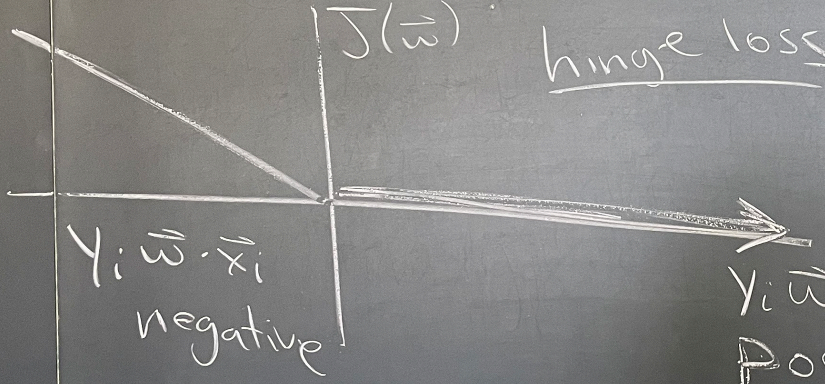


Surrogate Loss

$2w \Rightarrow 2$

$$J(\vec{w}) = \sum_{i=1}^{n} \max(0, -\underbrace{y_i \vec{w} \cdot \vec{x}_i}_{\text{if same sign}})$$

hinge loss



$\boxed{\text{if incorrect}}$

$$\boxed{\nabla J_{\vec{x}_i}(\vec{w}) = -y_i \vec{x}_i}$$

$$\vec{w} \leftarrow \vec{w} - \alpha \nabla J_{x_i}$$

$$\boxed{\vec{w} + \alpha y_i \vec{x}_i}$$

point 1

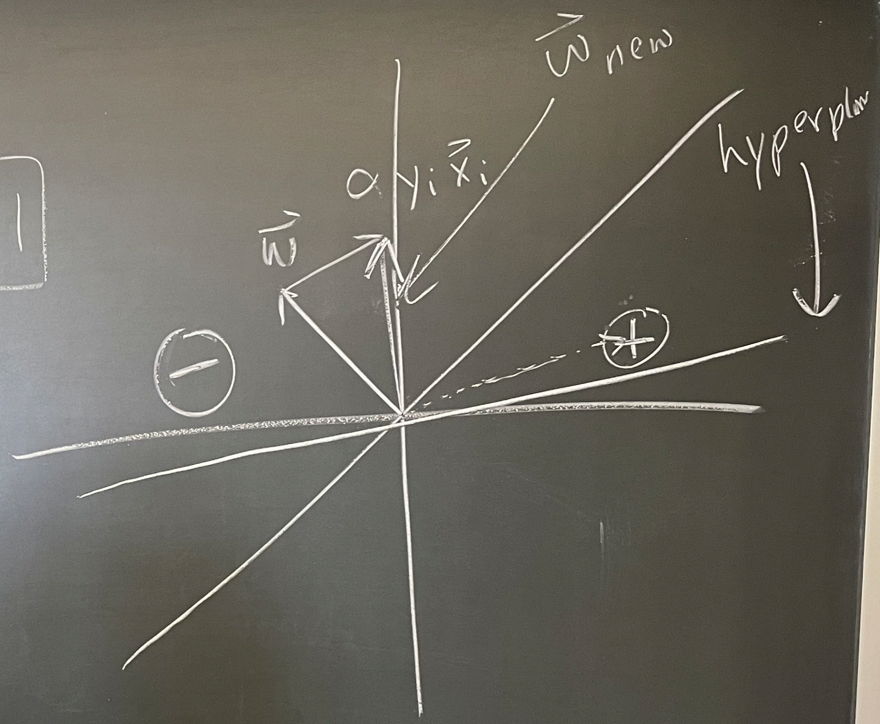
$$\begin{bmatrix} -5 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 5 \end{bmatrix} = 2 \Rightarrow \boxed{\hat{y} = 1}$$

$$\vec{w} \cdot \vec{x}$$

point 2

$$\begin{bmatrix} -5 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ -3 \\ -2 \end{bmatrix} = -13$$

$$\boxed{\hat{y} = -1}$$



Outline for March 19

- Perceptron Algorithm
- Informal check-in
- Handout 15 example
- Introduction to Support Vector Machines

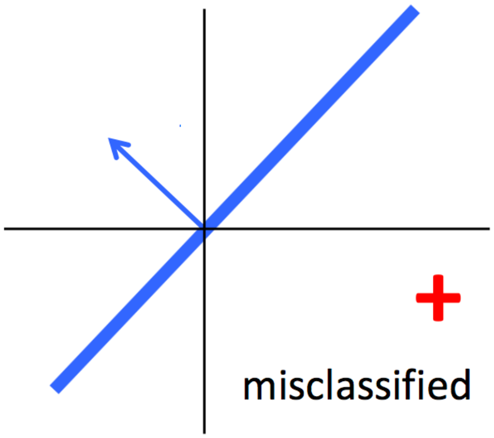
Informal discussion with a partner

1) What is the relationship between the weight vector ***w*** and the hyperplane?

2) Why is the perceptron cost function intuitive?

$$J(\vec{w}) = \sum_{i=1}^{n} \max\left(0, -y_i(\vec{w}^T \vec{x}_i)\right)$$

3) In the example to the right, how will the slope of the hyperplane change?



4) What are the weaknesses of the perceptron? Create a binary classifier "wishlist".

Informal discussion with a partner

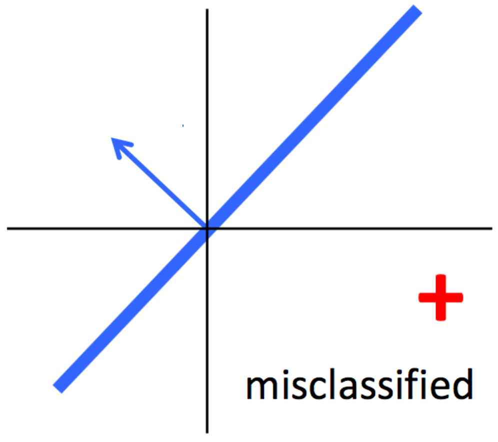
1) What is the relationship between the weight vector ***w*** and the hyperplane?

They are perpendicular

2) Why is the perceptron cost function intuitive?

$$J(\vec{w}) = \sum_{i=1}^{n} \max\left(0, -y_i(\vec{w}^T\vec{x}_i)\right)$$

3) In the example to the right, how will the slope of the hyperplane change?



4) What are the weaknesses of the perceptron? Create a binary classifier "wishlist".

Informal discussion with a partner

1) What is the relationship between the weight vector ***w*** and the hyperplane?

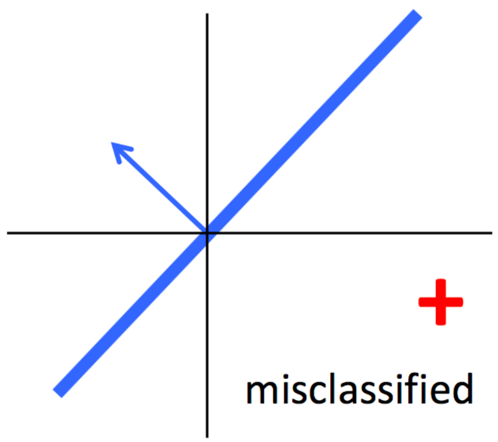
   They are perpendicular

2) Why is the perceptron cost function intuitive?

$$J(\vec{w}) = \sum_{i=1}^{n} \max\left(0, -y_i(\vec{w}^T \vec{x}_i)\right)$$

   Cost function is 0 when classification is correct, and positive when incorrect

3) In the example to the right, how will the slope of the hyperplane change?



4) What are the weaknesses of the perceptron? Create a binary classifier "wishlist".

Perceptron algorithm and intuition

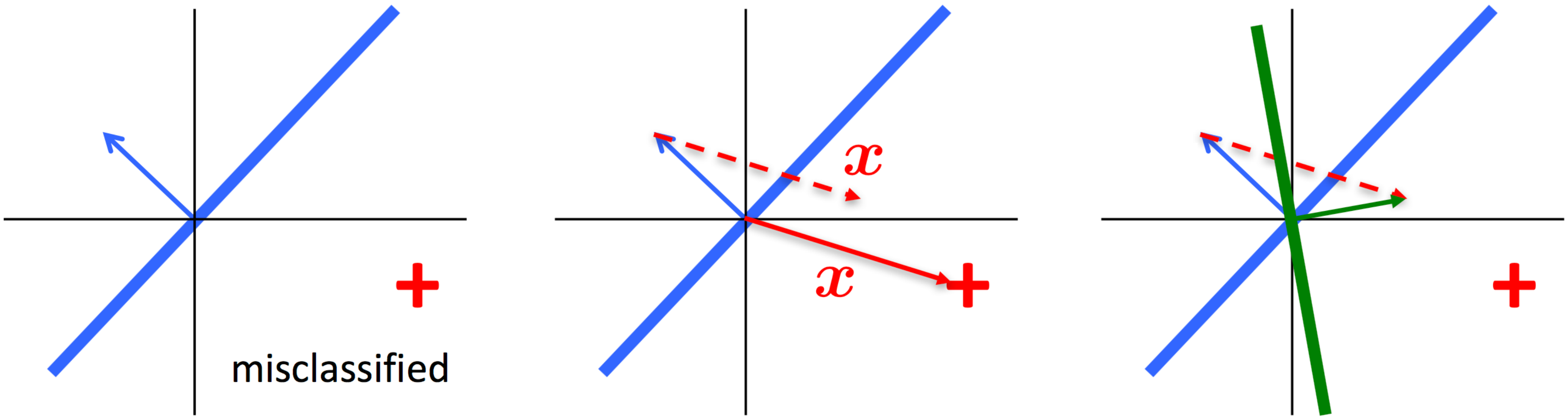
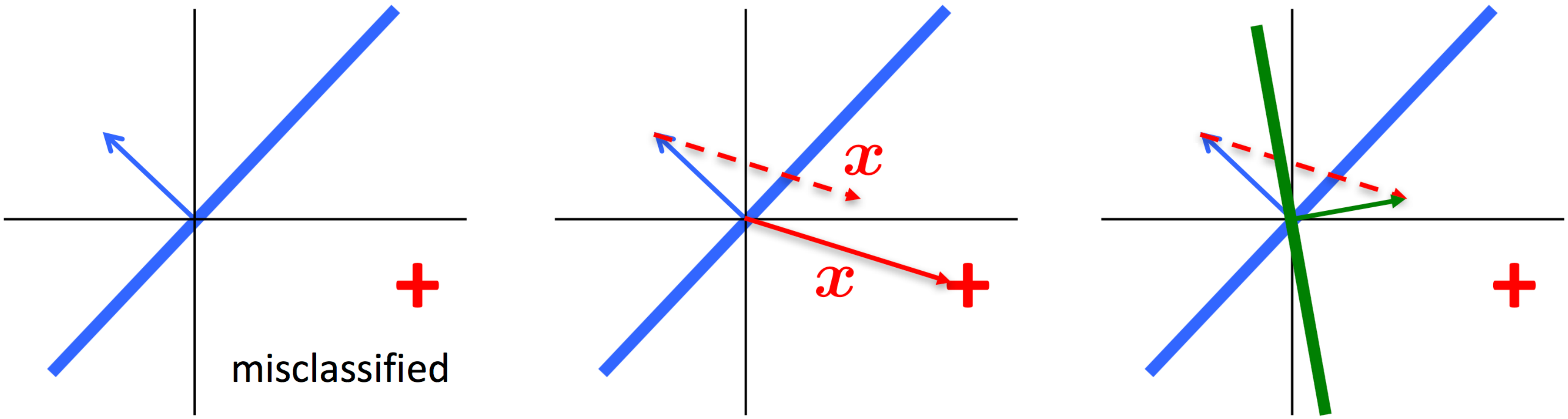


Image and Algorithm: modified from Eric Eaton

Perceptron algorithm and intuition



Let $\vec{w} = [0, 0, \cdots, 0]^T$

Repeat until convergence:

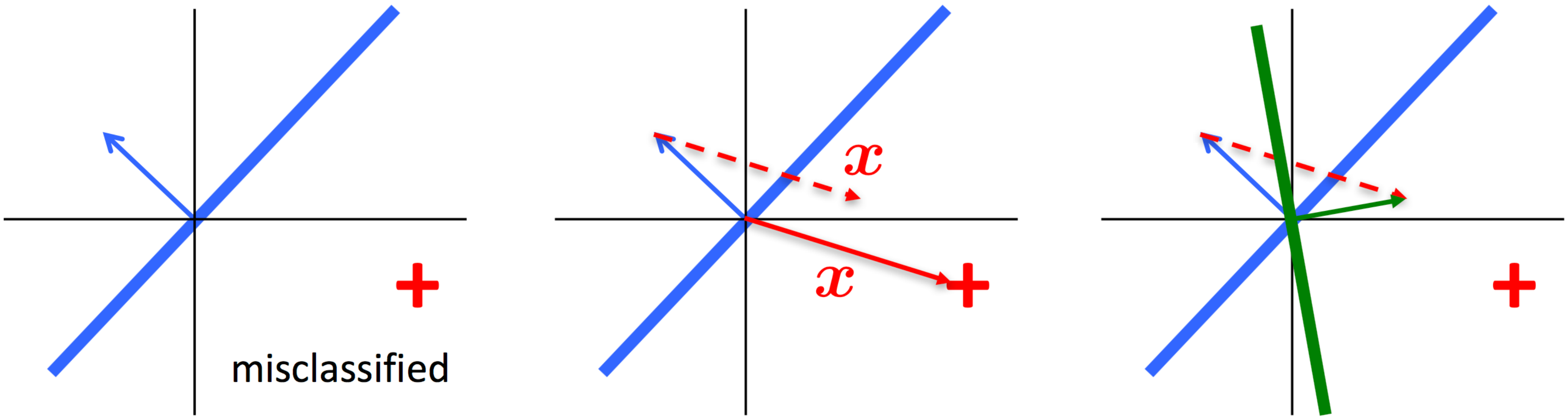
Receive training example $(\vec{x}_i, y_i)$

If $y_i(\vec{w}^T \vec{x}_i) \leq 0$ (incorrectly classified)

$$\vec{w} \leftarrow \vec{w} + \alpha y_i \vec{x}_i$$

Image and Algorithm: modified from Eric Eaton

Perceptron algorithm and intuition



Let $\vec{w} = [0, 0, \cdots, 0]^T$

Repeat until convergence:

Receive training example $(\vec{x}_i, y_i)$

If $y_i(\vec{w}^T \vec{x}_i) \leq 0$ (incorrectly classified)

$$\vec{w} \leftarrow \vec{w} + \alpha y_i \vec{x}_i$$

Convergence:

- All data points correctly classified
- Fixed number of iterations passed

Often: alpha = 1 (only changes magnitude of weight vector)

Image and Algorithm: modified from Eric Eaton

Binary classifier wishlist

- If data is linearly separable, want a “good” hyperplane (idea: far from points close to the boundary)

- If data is not linearly separable, want something reasonable (not just give up or fail to converge)

- Might not want to constrain ourselves to linear separators

Outline for March 19

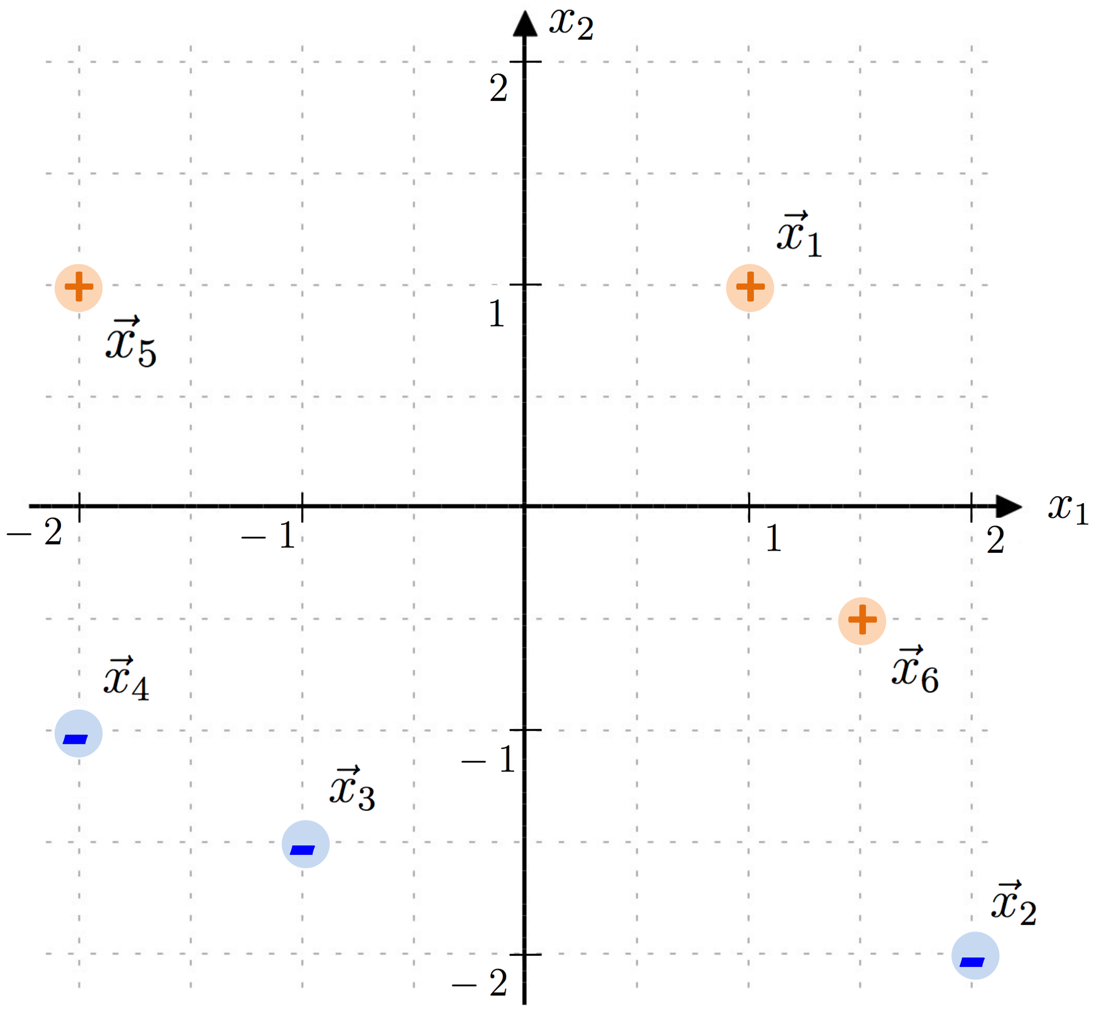
- Perceptron Algorithm
- Informal check-in
- Handout 15 example
- Introduction to Support Vector Machines

Handout 15 example

Initial values:

$$\alpha = 0.2$$

$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$



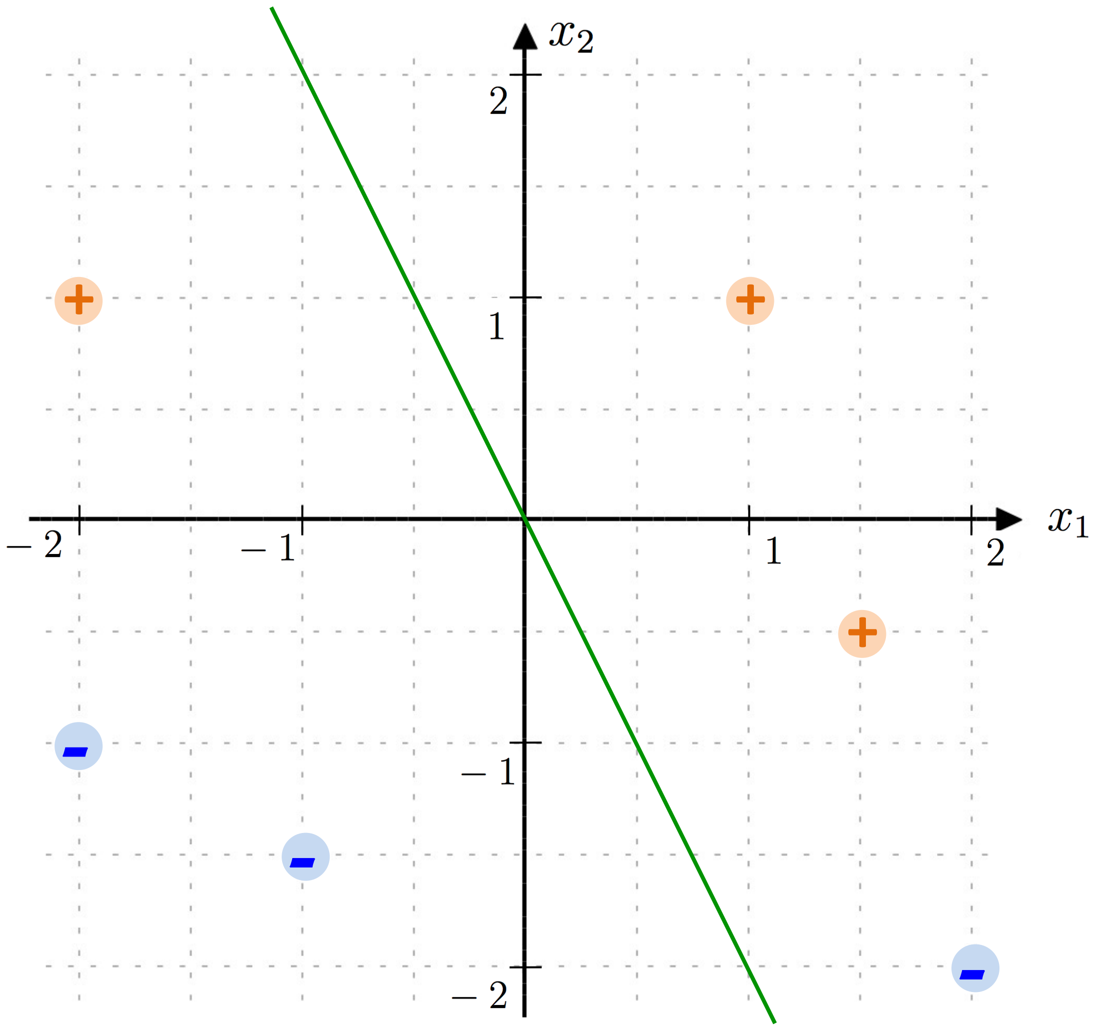
Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 15 example

Initial values:

$$\alpha = 0.2$$

$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 15 example

$$\alpha = 0.2$$

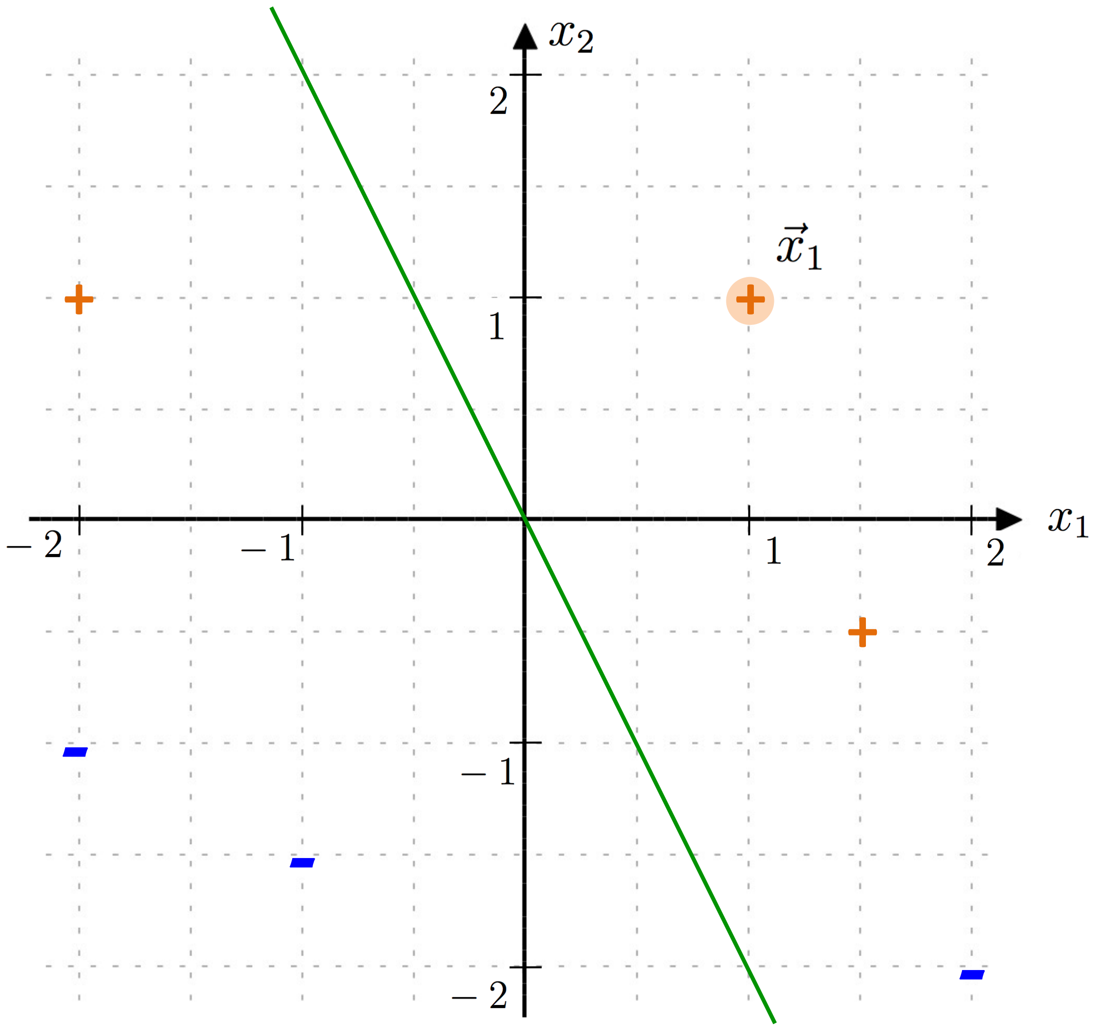
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 1:

$$\vec{x}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_1 > 0$$

Correct classification, no action



Handout 15 example

$$\alpha = 0.2$$

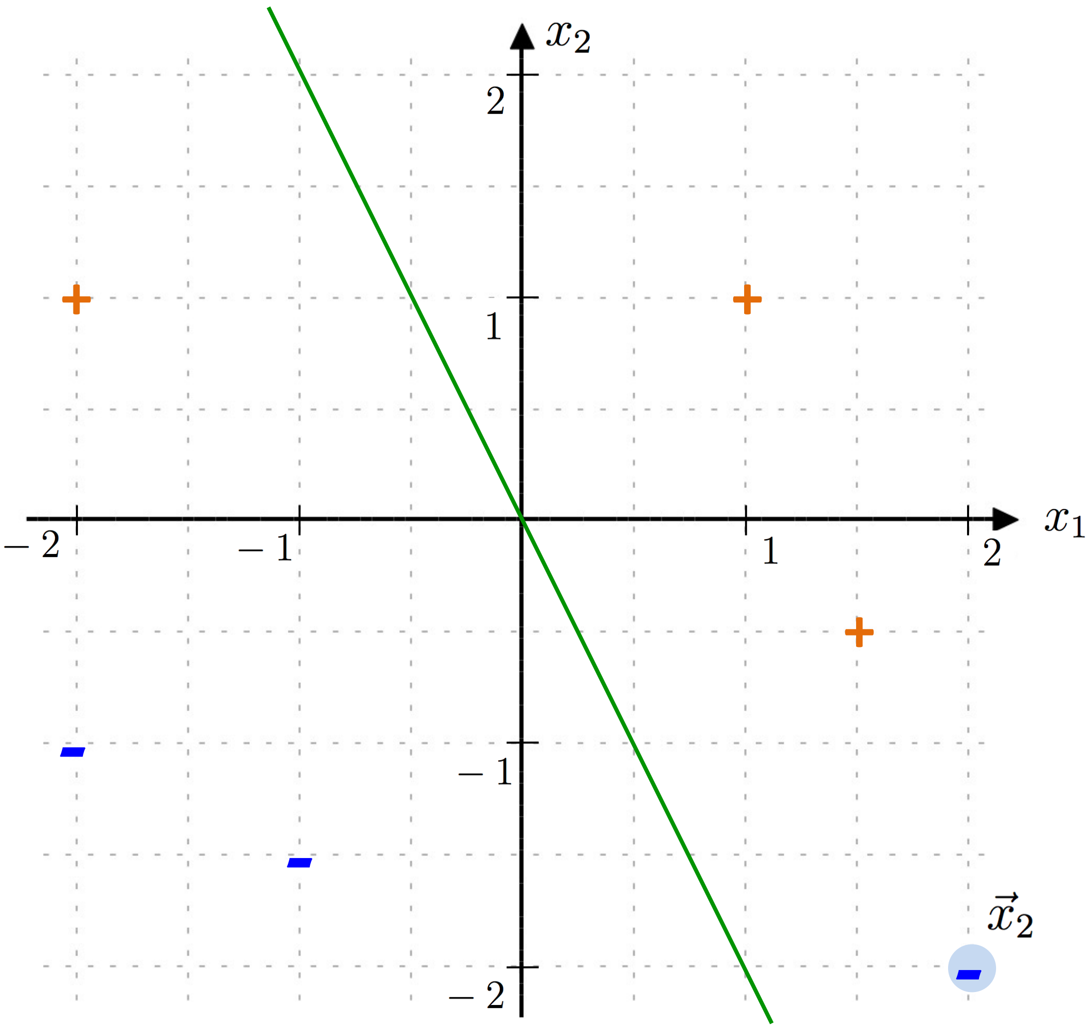
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 15 example

$$\alpha = 0.2$$

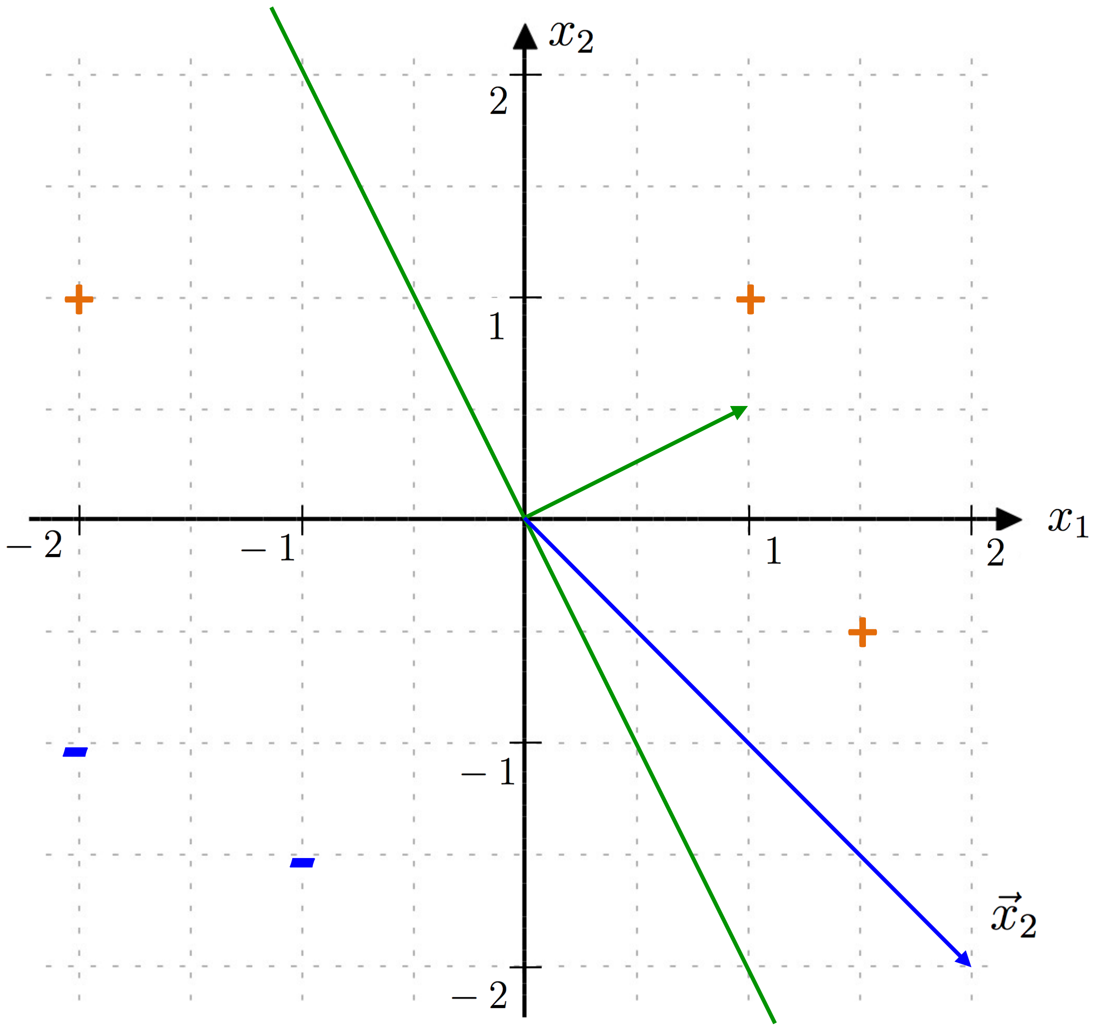
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 15 example

$$\alpha = 0.2$$

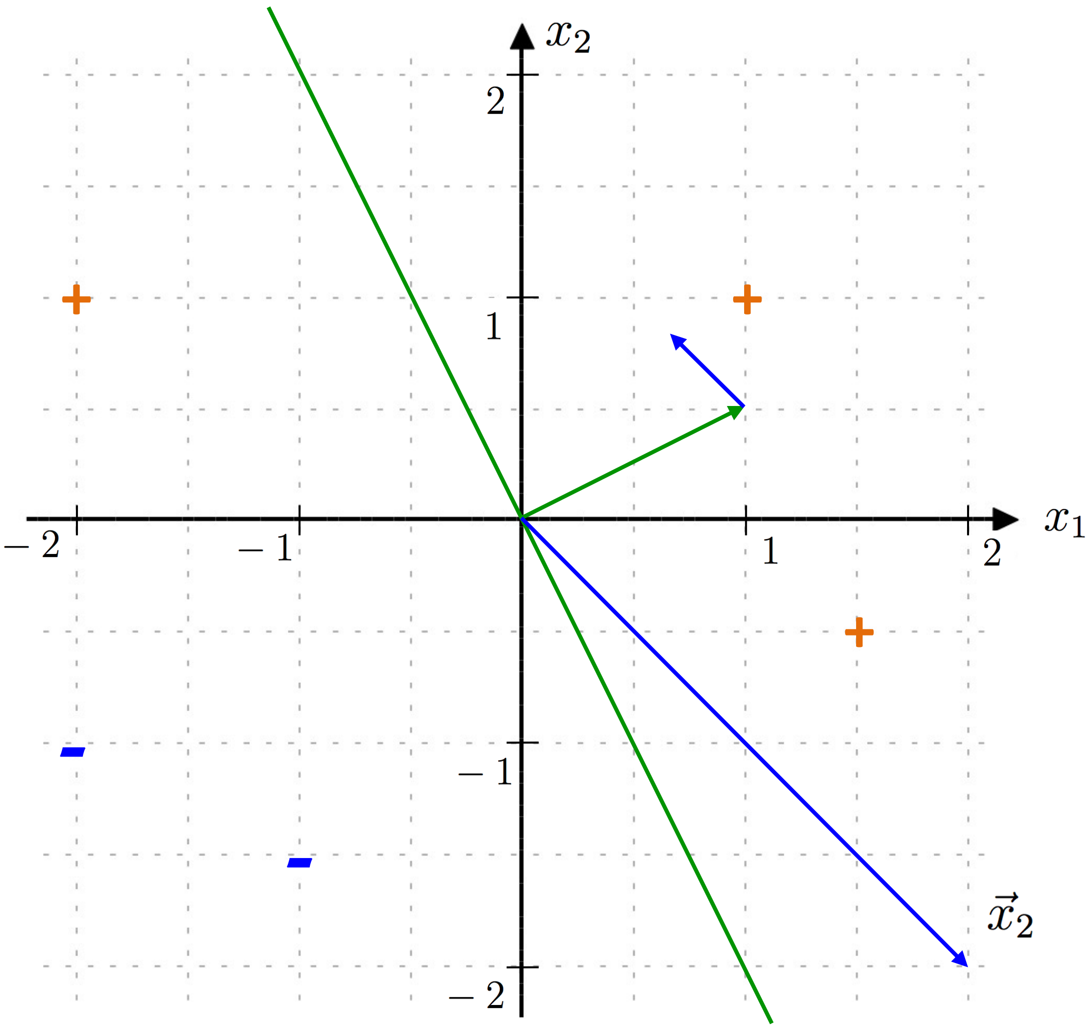
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification
"Push" ***w*** away from negative point



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 15 example

$$\alpha = 0.2$$

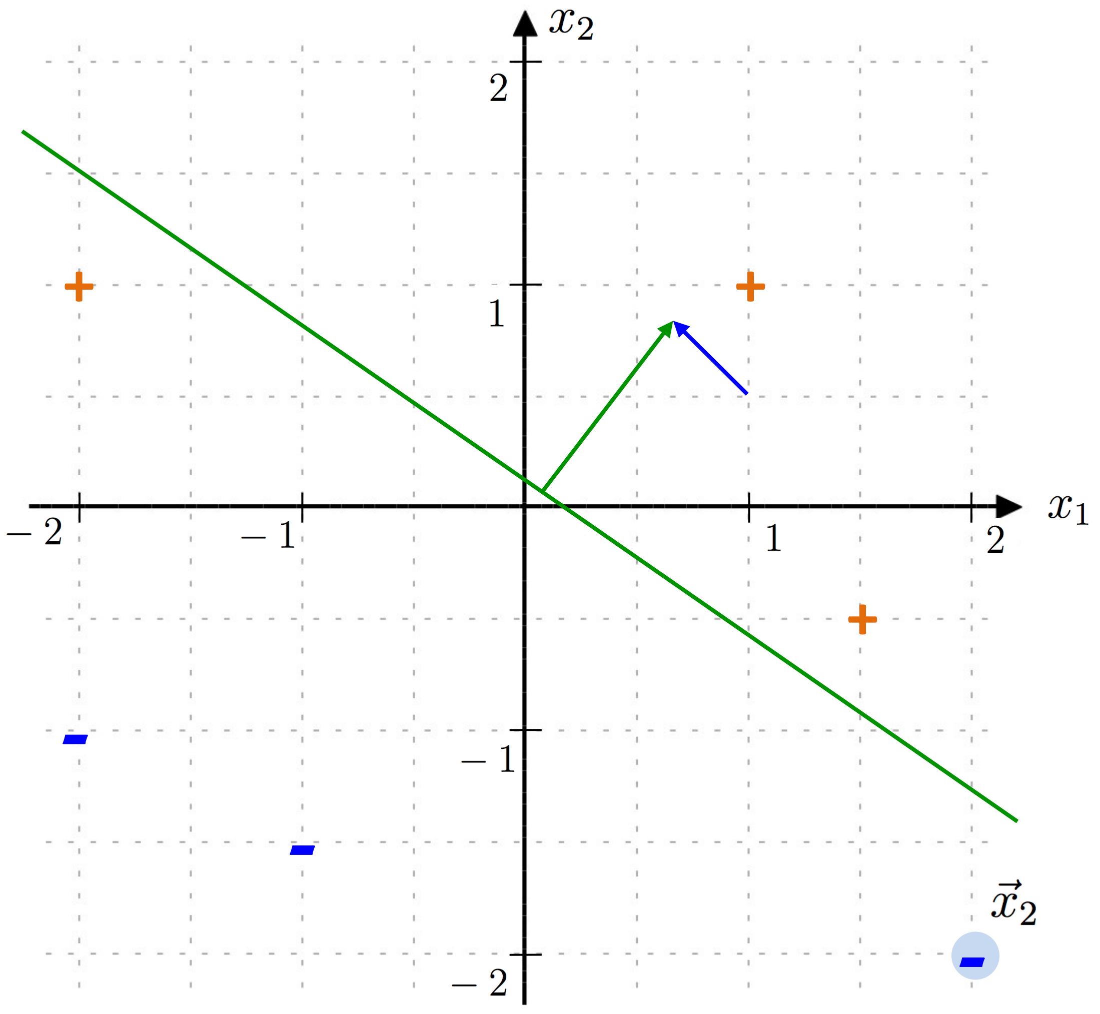
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification
"Push" ***w*** away from negative point



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 15 example

$$\alpha = 0.2$$

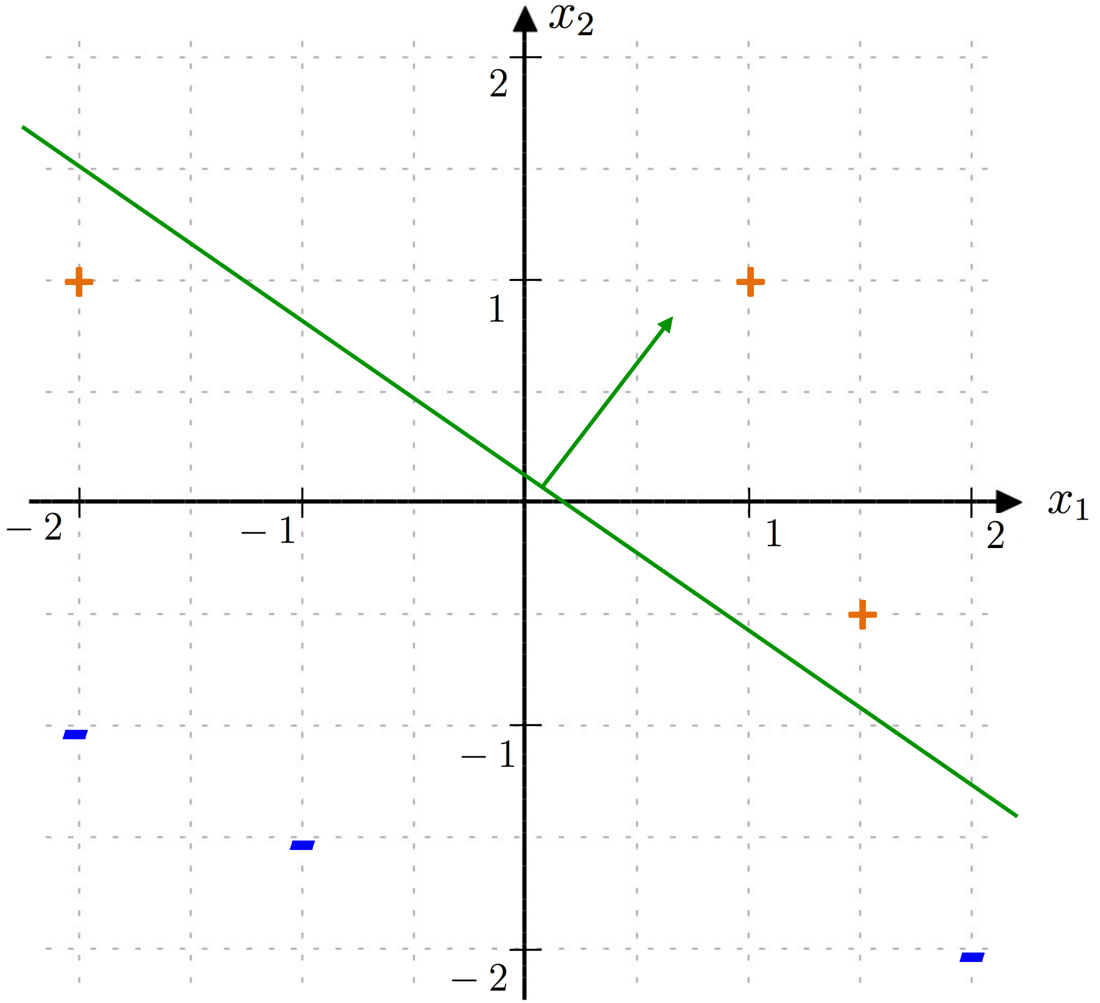
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

What is the new weight vector?



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 15 example

$$\vec{w} = \begin{bmatrix} -0.2 \\ 0.6 \\ 0.9 \end{bmatrix}$$

$$\overbrace{-0.2 + 0.6x_1 + 0.9x_2 = 0}^{\underline{\text{hyperplane}}}$$

$$\frac{0.9x_2}{0.9} = \frac{0.2 - 0.6x_1}{0.9}$$

x_2
y-intercept

$$x_2 = \boxed{\frac{0.2}{0.9}} - \frac{0.6x_1}{0.9}$$

Handout 15 example

Final solution (so you can check your work):

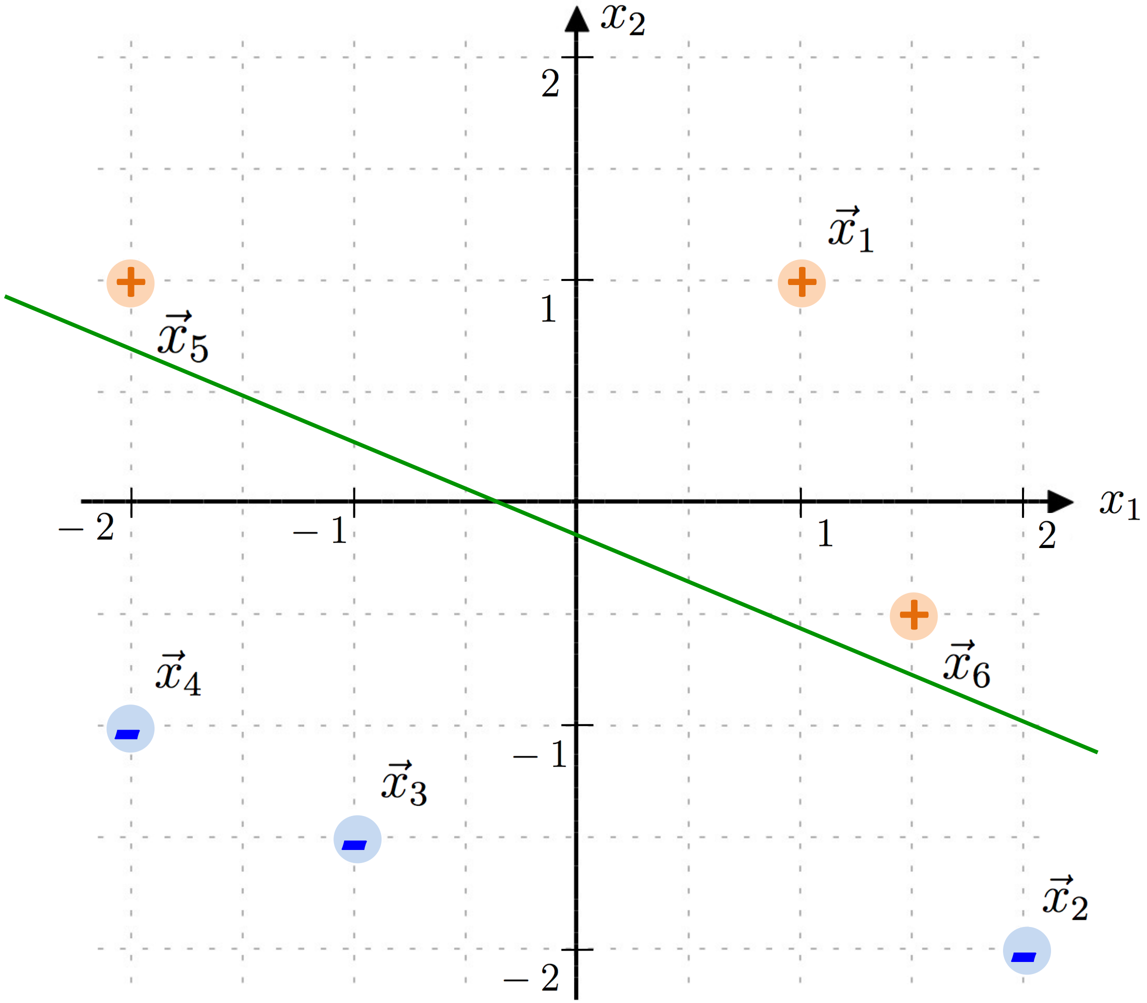
$$\vec{w}^* = \begin{bmatrix} 0.2 \\ 0.5 \\ 1 \end{bmatrix}$$

Final hyperplane:

$$0.2 + 0.5x_1 + x_2 = 0$$

$$\Rightarrow$$

$$x_2 = -0.2 - 0.5x_1$$



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Outline for March 19

- Perceptron Algorithm
- Informal check-in
- Handout 15 example
- Introduction to Support Vector Machines

Support Vector Machines (SVMs)

- Will give us everything on our wishlist!
- Often considered the best "off the shelf" binary classifier
- Widely used in many fields

Brief history

- 1963: Initial idea by Vladimir Vapnik and Alexey Chervonenkis
- 1992: nonlinear SVMs by Bernhard Boser, Isabelle Guyon and Vladimir Vapnik
- 1993: "soft-margin" by Corinna Cortes and Vladimir Vapnik

Idea: “best” hyperplane has a large margin

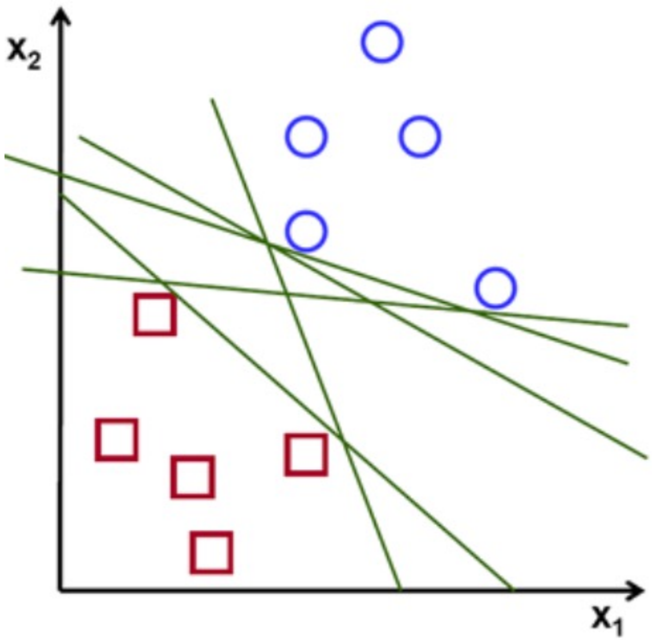


Image: “Towards Data Science”

Idea: “best” hyperplane has a large margin

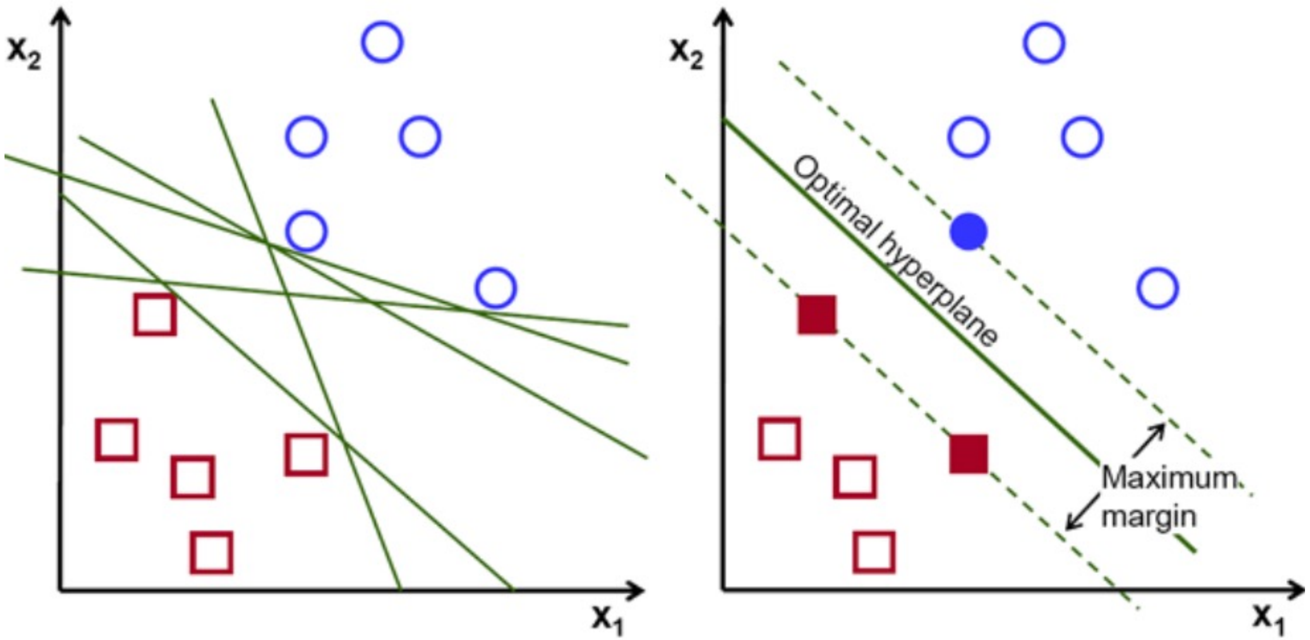
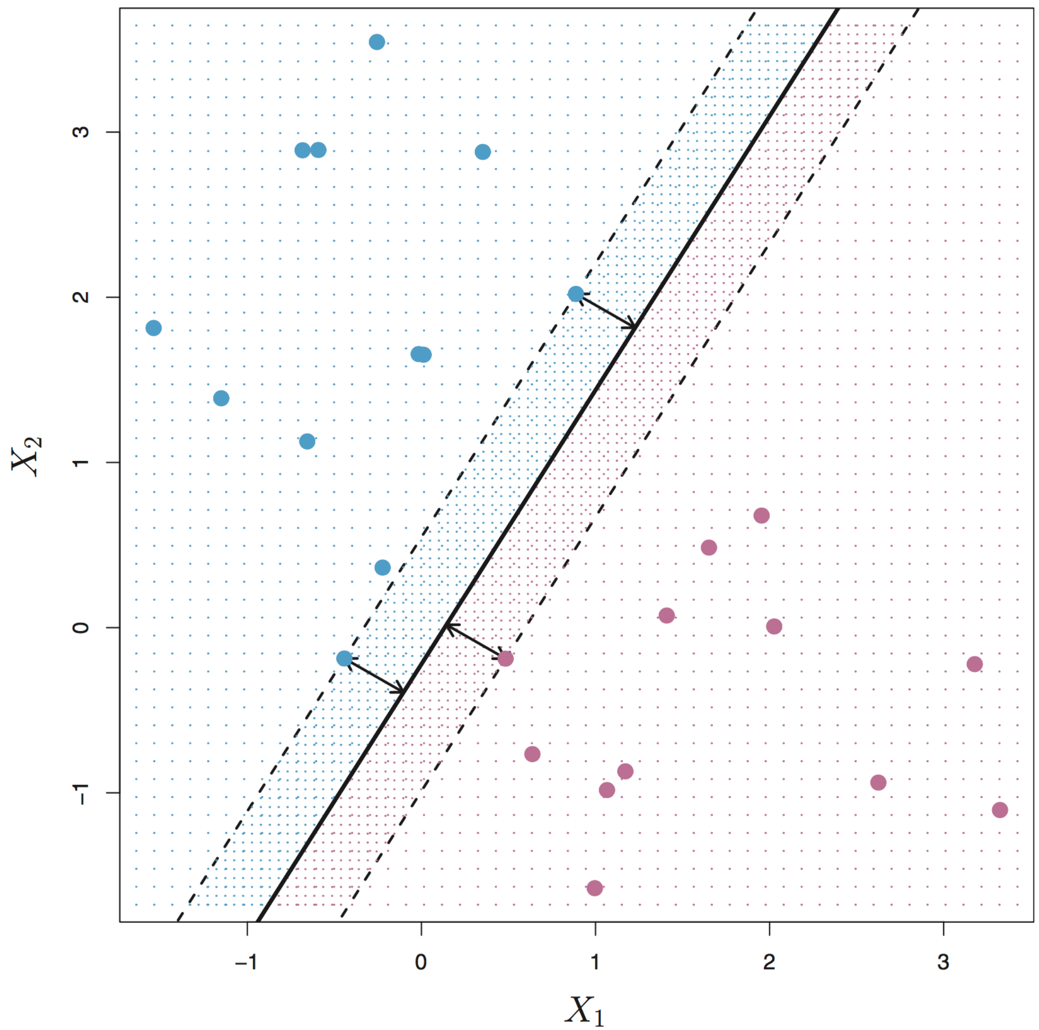


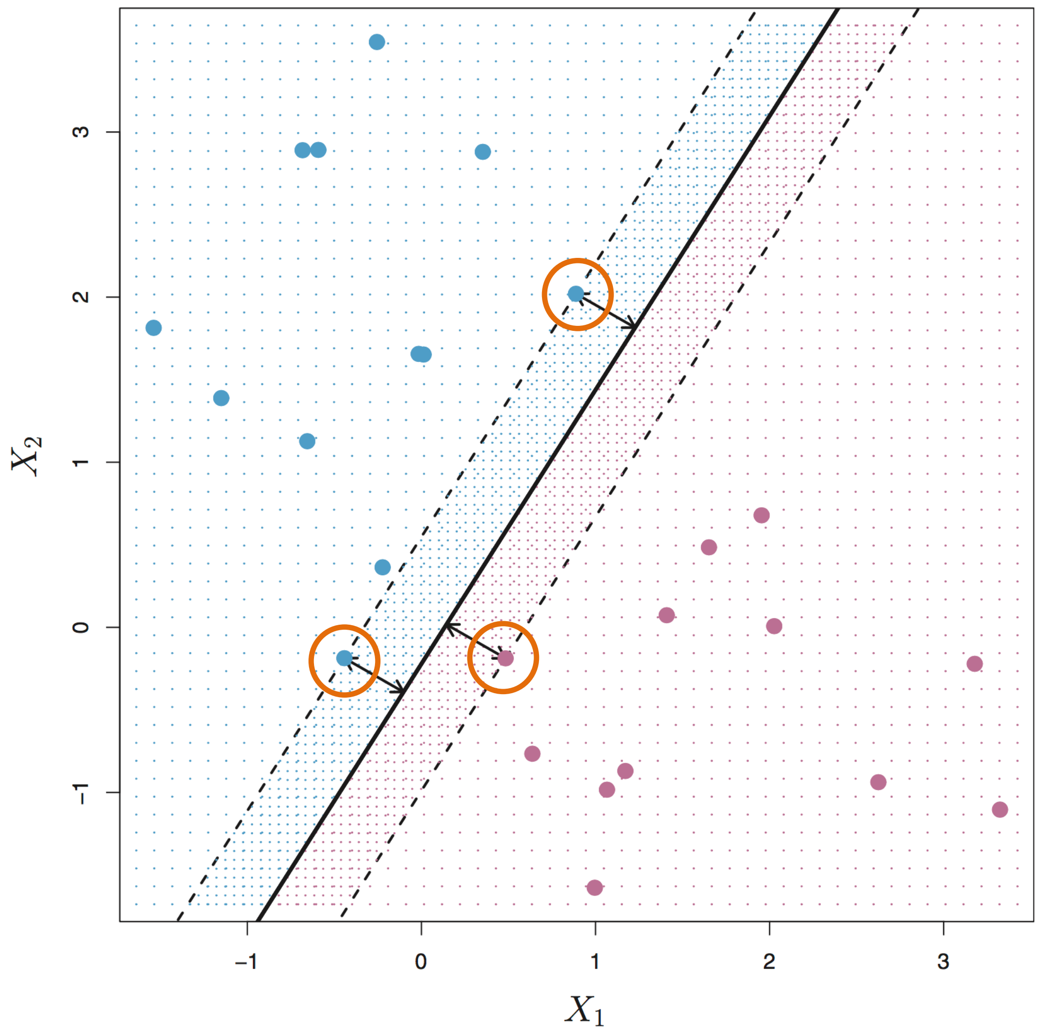
Image: “Towards Data Science”

Datapoints that lie on the margin are called "support vectors"



Modified from ISL: Figure 9.3

Datapoints that lie on the margin are called "support vectors"



Support vectors

Modified from ISL: Figure 9.3

SVMs

$$h(\vec{x}) = \text{sign}(\overset{p}{\vec{w}} \cdot \overset{p}{\vec{x}} + b)$$

no fake 1's

functional margin

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

geometric margin

dist between pt and hyperplane

γ_i

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\big(\vec{w} \cdot \vec{x} + b\big)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\big(\vec{w} \cdot \vec{x} + b\big)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\big(\vec{w} \cdot \vec{x} + b\big)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Geometric Margin:
(distance between
example and hyperplane)

$$\gamma_i = y_i \left(\frac{\vec{w}}{||\vec{w}||} \cdot \vec{x}_i + \frac{b}{||\vec{w}||} \right)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\left(\vec{w} \cdot \vec{x} + b\right)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Geometric Margin:
(distance between
example and hyperplane)

$$\gamma_i = y_i \left(\frac{\vec{w}}{||\vec{w}||} \cdot \vec{x}_i + \frac{b}{||\vec{w}||} \right)$$

Note:

$$\boxed{\gamma_i = \frac{\hat{\gamma}_i}{||\vec{w}||}}$$