

10-301/601: Introduction to Machine Learning

Lecture 6 – Perceptron

Henry Chai & Matt Gormley

1/31/25

Front Matter

- Announcements:
 - HW2 released 1/22, due 2/3 at 11:59 PM
 - HW3 released 2/3, due 2/10 at 11:59 PM
 - HW3 is a written-only homework
 - **You may only use at most 2 late days on HW3**

Q & A:

Can we use k NNs with categorical features?

- Yes! We can either:

1. Convert categorical features into binary ones:

Cholesterol		Normal Cholesterol?	Abnormal Cholesterol?
Normal	→	1	0
Normal		1	0
Abnormal		0	1

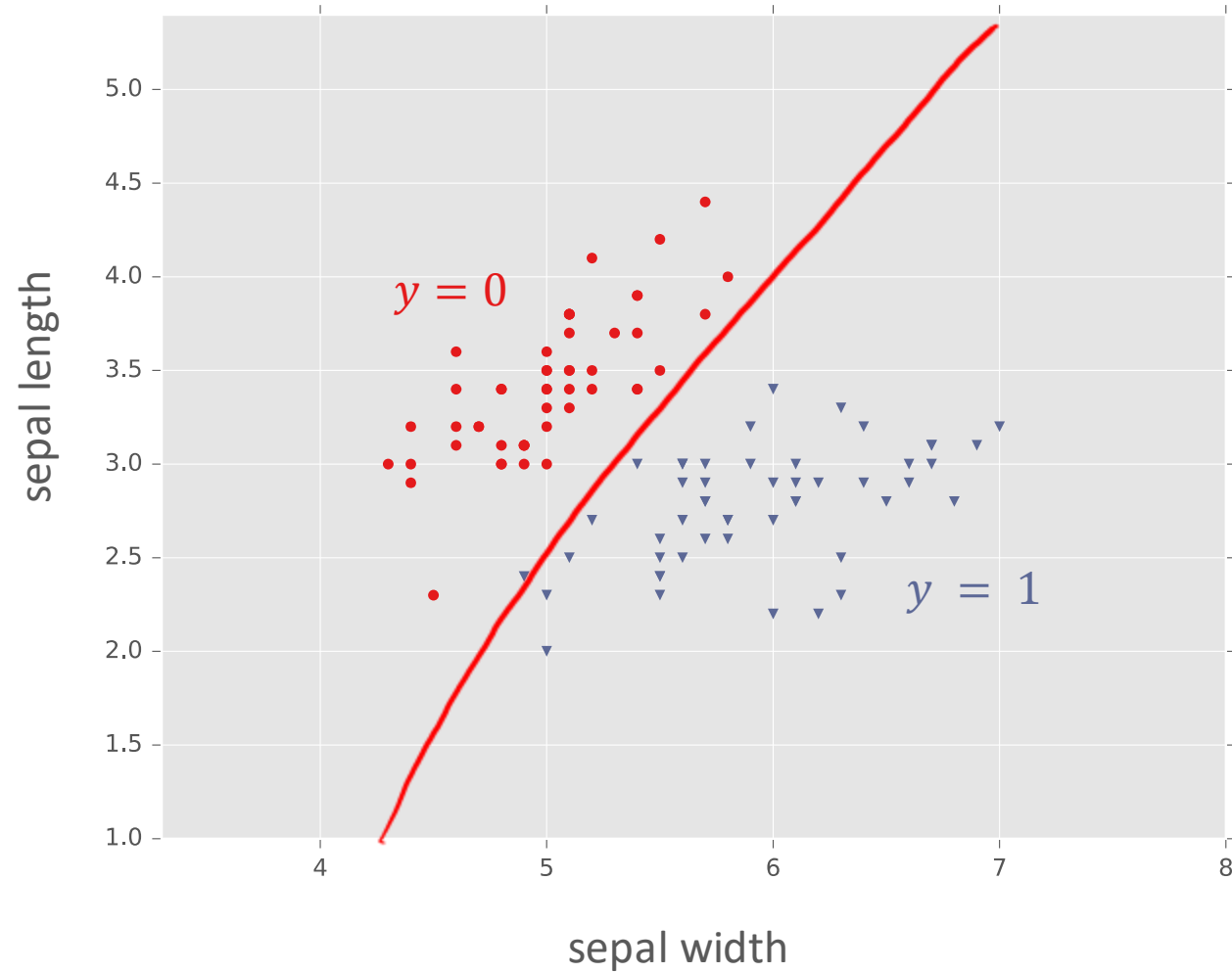
2. Use a distance metric that works over categorical features e.g., the Hamming distance:

$$d(\mathbf{x}, \mathbf{x}') = \sum_{d=1}^D \mathbb{1}(x_d \neq x'_d)$$

indicator

- See HW3 for an example of this

Recall: Fisher Iris Dataset



Linear Algebra Review

- Notation: in this class vectors will be assumed to be column vectors by default, i.e.,

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_D \end{bmatrix} \text{ and } \mathbf{a}^T = [a_1 \quad a_2 \quad \cdots \quad a_D]$$

- The dot product between two D -dimensional vectors is

$$\mathbf{a}^T \mathbf{b} = [a_1 \quad a_2 \quad \cdots \quad a_D] \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_D \end{bmatrix} = \sum_{d=1}^D a_d b_d$$

- The L_2 -norm of $\mathbf{a} = \|\mathbf{a}\|_2 = \sqrt{\mathbf{a}^T \mathbf{a}}$
- Two vectors are *orthogonal* iff

$$\mathbf{a}^T \mathbf{b} = 0$$

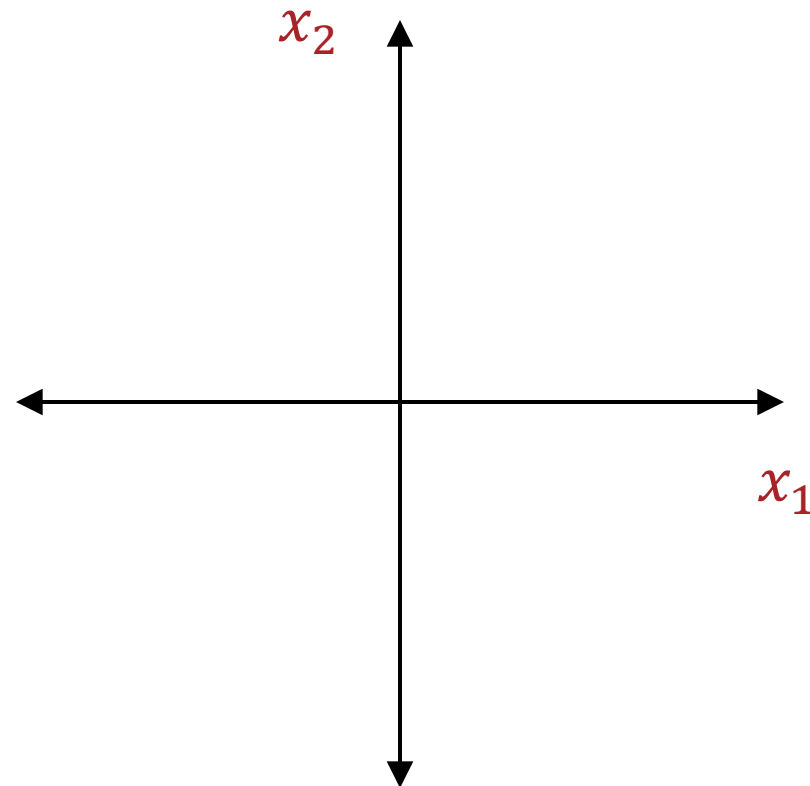
Geometry Warm-up

1. On the axes below, draw the region corresponding to

$$w_1x_1 + w_2x_2 + b > 0$$

where $w_1 = 1$, $w_2 = 2$ and $b = -4$.

2. Then draw the vector $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$



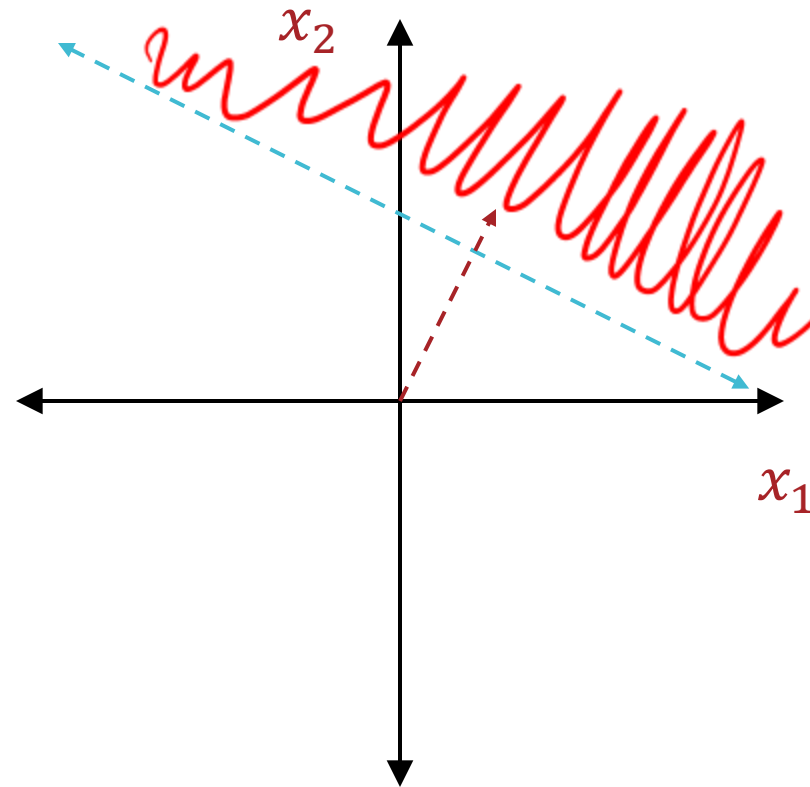
Geometry Warm-up

1. On the axes below, draw the region corresponding to

$$w_1x_1 + w_2x_2 + b > 0$$

where $w_1 = 1$, $w_2 = 2$ and $b = -4$.

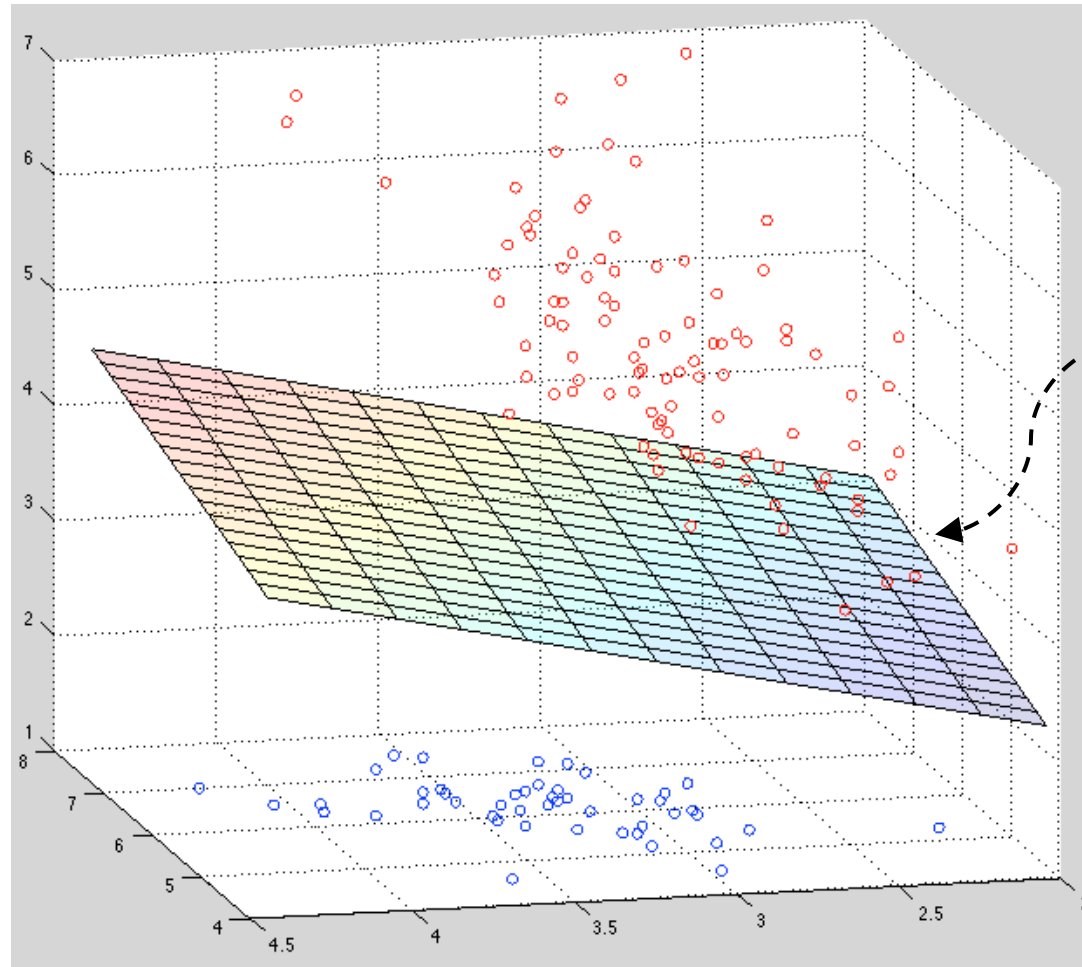
2. Then draw the vector $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$



Linear Decision Boundaries

- In 2 dimensions, $w_1x_1 + w_2x_2 + b = 0$ defines a *line*
- In 3 dimensions, $w_1x_1 + w_2x_2 + w_3x_3 + b = 0$ defines a *plane*
- In 4+ dimensions, $\mathbf{w}^T \mathbf{x} + b = 0$ defines a *hyperplane*
 - The vector $\mathbf{w}$ is always orthogonal to this hyperplane and always points in the direction where $\mathbf{w}^T \mathbf{x} + b > 0$!
- A hyperplane creates two *halfspaces*:
 - $\mathcal{S}_+ = \{\mathbf{x}: \mathbf{w}^T \mathbf{x} + b > 0\}$ or all $\mathbf{x}$ s.t. $\mathbf{w}^T \mathbf{x} + b$ is positive
 - $\mathcal{S}_- = \{\mathbf{x}: \mathbf{w}^T \mathbf{x} + b < 0\}$ or all $\mathbf{x}$ s.t. $\mathbf{w}^T \mathbf{x} + b$ is negative

Linear Decision Boundaries: Example



Goal: learn classifiers of the form $h(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + b)$ (assuming $y \in \{-1, +1\}$)

Key question: how do we learn the parameters, $\mathbf{w}$?
and b

Online Learning

- So far, we've been learning in the *batch* setting, where we have access to the entire training dataset at once
- A common alternative is the *online* setting, where examples arrive gradually and we learn continuously
- Examples of online learning:
 - predicting stock prices
 - real-time translation
 - recommender systems
 - infectious disease modeling

Online Learning: Setup

- For $t = 1, 2, 3, \dots$
 - Receive an unlabeled example, $\mathbf{x}^{(t)}$
 - Predict its label, $\hat{y} = h_{\mathbf{w}, b}(\mathbf{x}^{(t)})$
 - Observe its true label, $y^{(t)}$
 - Pay a penalty if we made a mistake, $\hat{y} \neq y^{(t)}$
 - Update the parameters, $\mathbf{w}$ and b
- Goal: minimize the number of mistakes made

(Online) Perceptron Learning Algorithm

- Initialize the weight vector and intercept to all zeros:

$$\mathbf{w} = [0 \quad 0 \quad \dots \quad 0] \text{ and } b = 0$$

- For $t = 1, 2, 3, \dots$
 - Receive an unlabeled example, $\mathbf{x}^{(t)}$
 - Predict its label, $\hat{y} = \text{sign}(\mathbf{w}^T \mathbf{x} + b) = \begin{cases} +1 & \text{if } \mathbf{w}^T \mathbf{x} + b \geq 0 \\ -1 & \text{otherwise} \end{cases}$
 - Observe its true label, $y^{(t)}$
 - If we misclassified a positive example ($y^{(t)} = +1, \hat{y} = -1$):
 - $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{x}^{(t)}$
 - $b \leftarrow b + 1$
 - If we misclassified a negative example ($y^{(t)} = -1, \hat{y} = +1$):
 - $\mathbf{w} \leftarrow \mathbf{w} - \mathbf{x}^{(t)}$
 - $b \leftarrow b - 1$

(Online) Perceptron Learning Algorithm

- Initialize the weight vector and intercept to all zeros:

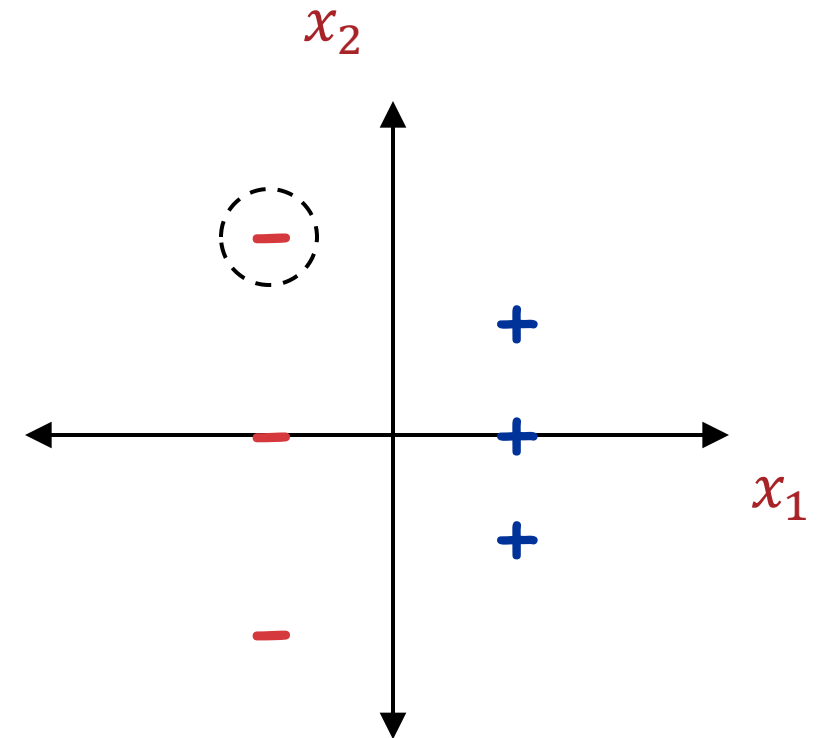
$$\mathbf{w} = [0 \quad 0 \quad \dots \quad 0] \text{ and } b = 0$$

- For $t = 1, 2, 3, \dots$
 - Receive an unlabeled example, $\mathbf{x}^{(t)}$
 - Predict its label, $\hat{y} = \text{sign}(\mathbf{w}^T \mathbf{x} + b) = \begin{cases} +1 & \text{if } \mathbf{w}^T \mathbf{x} + b \geq 0 \\ -1 & \text{otherwise} \end{cases}$
 - Observe its true label, $y^{(t)}$
 - If we misclassified an example ($y^{(t)} \neq \hat{y}$):
 - $\mathbf{w} \leftarrow \mathbf{w} + y^{(t)} \mathbf{x}^{(t)}$
 - $b \leftarrow b + y^{(t)}$

(Online) Perceptron Learning Algorithm: Example (no Intercept)

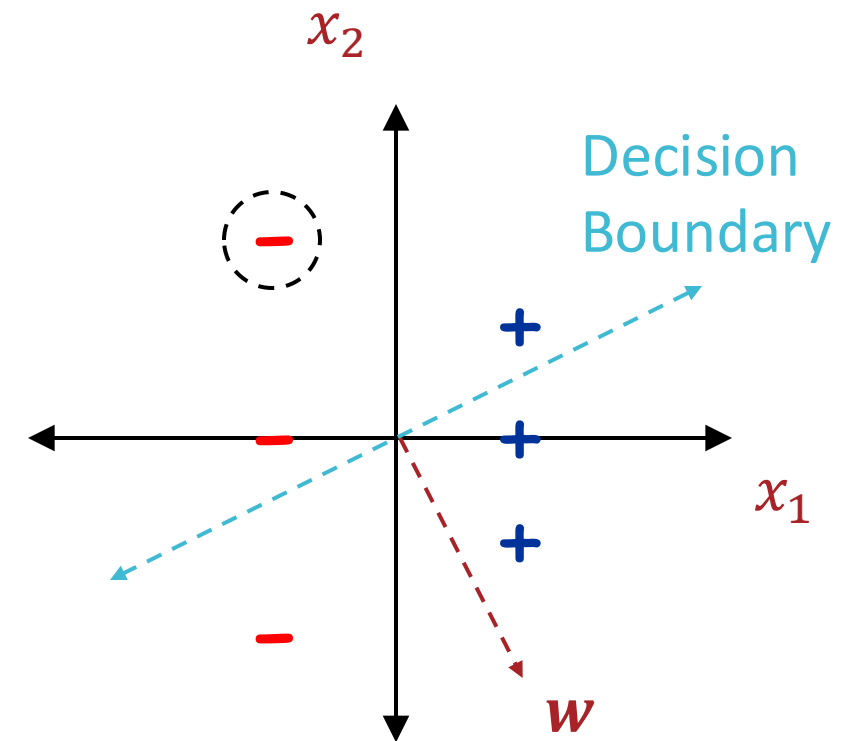
x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes

$$w = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$



(Online) Perceptron Learning Algorithm: Example (no Intercept)

x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes



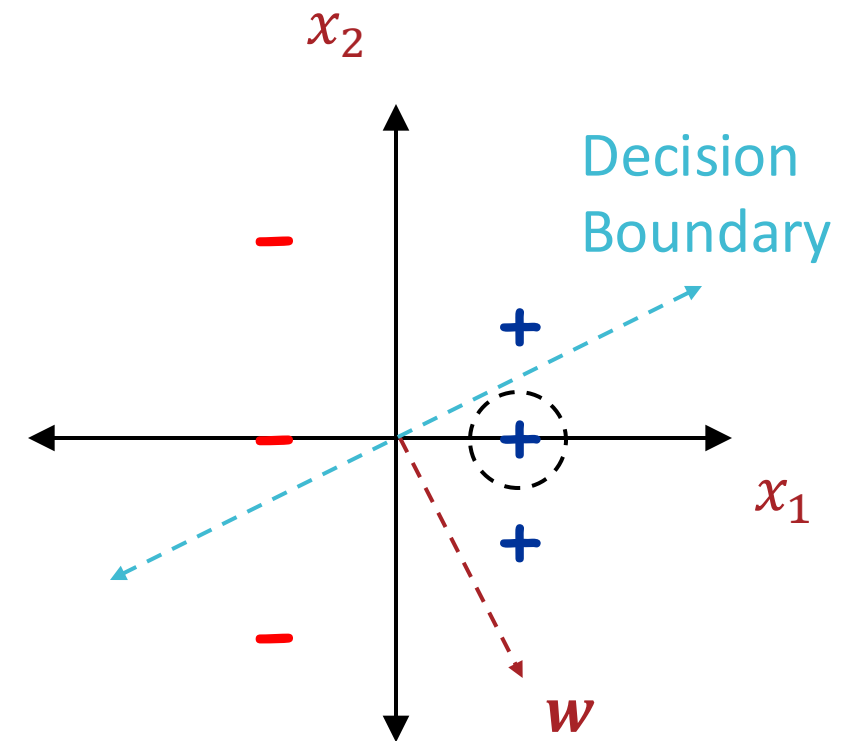
$$\mathbf{w} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathbf{w} \leftarrow \mathbf{w} + y^{(1)} \mathbf{x}^{(1)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

(Online) Perceptron Learning Algorithm: Example (no Intercept)

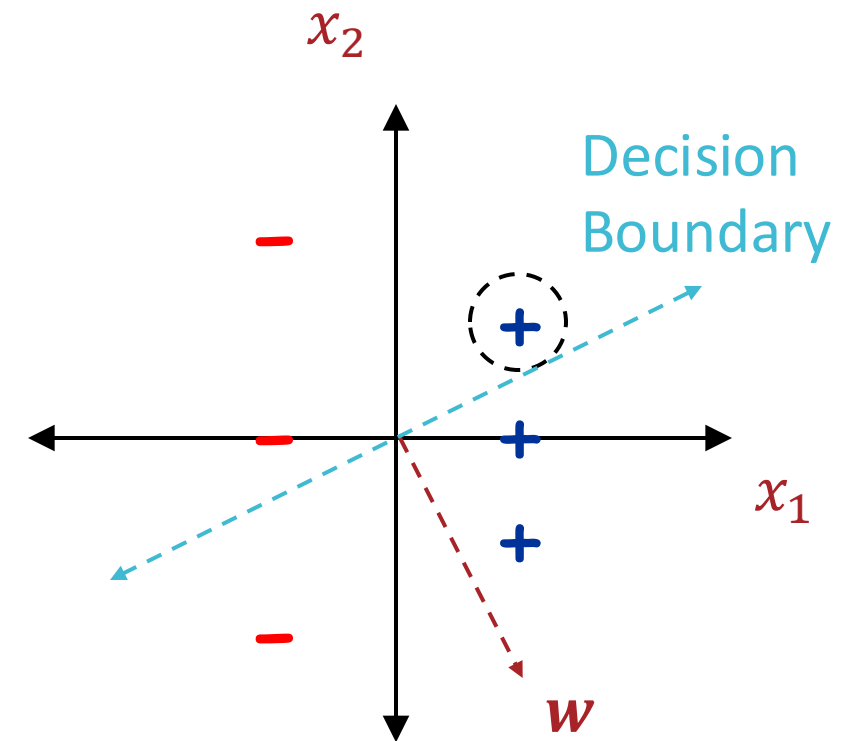
x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes
1	0	+	+	No

$$w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$



(Online) Perceptron Learning Algorithm: Example (no Intercept)

x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes
1	0	+	+	No
1	1	-	+	Yes

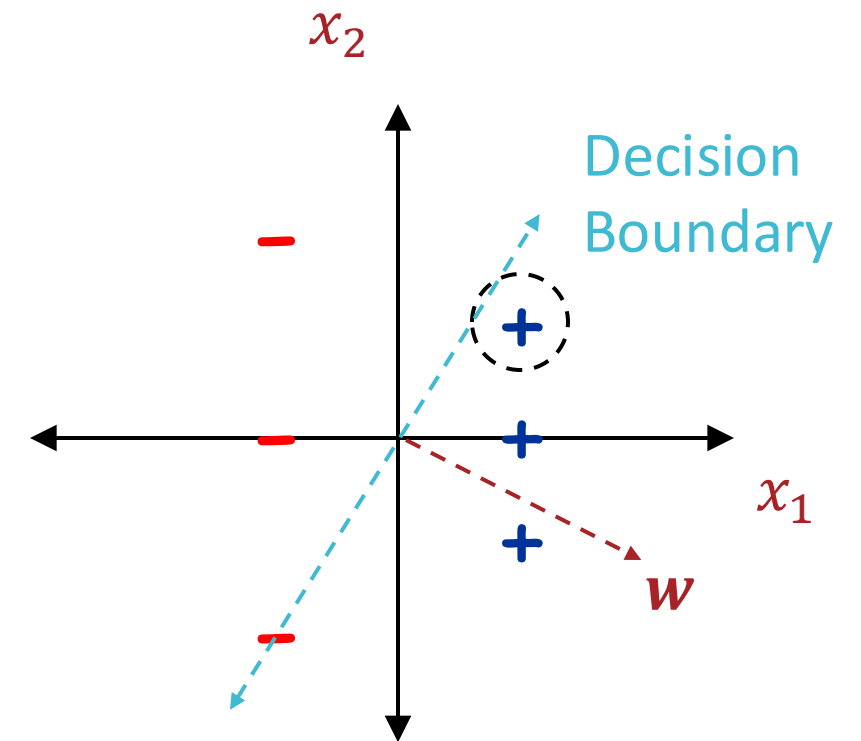


$$\mathbf{w} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

$$\mathbf{w} \leftarrow \mathbf{w} + y^{(3)} \mathbf{x}^{(3)} = \begin{bmatrix} 1 \\ -2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$

(Online) Perceptron Learning Algorithm: Example (no Intercept)

x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes
1	0	+	+	No
1	1	-	+	Yes



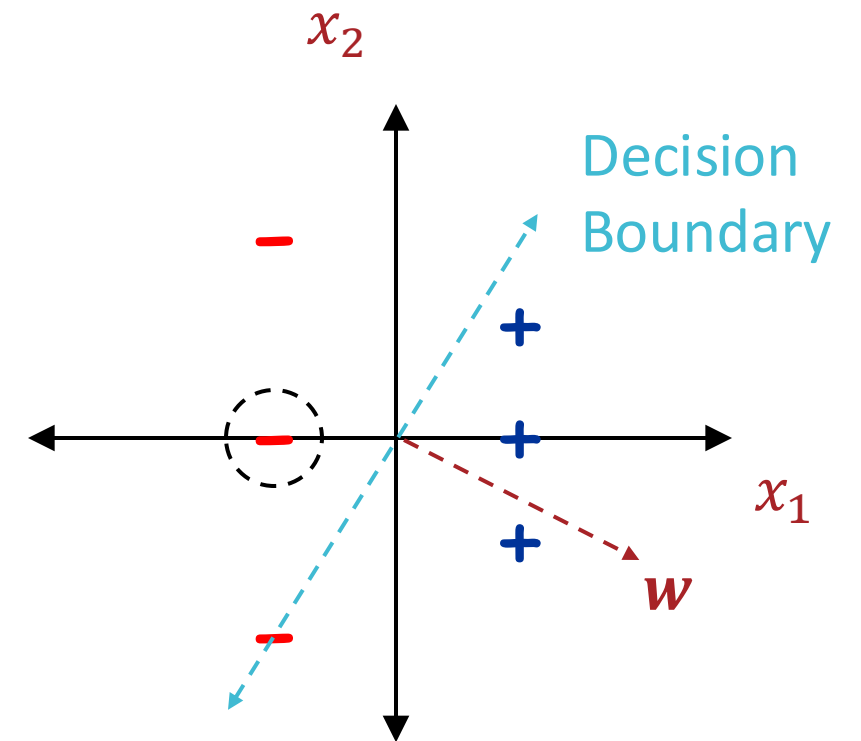
$$\mathbf{w} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

$$\mathbf{w} \leftarrow \mathbf{w} + y^{(3)} \mathbf{x}^{(3)} = \begin{bmatrix} 1 \\ -2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$

(Online) Perceptron Learning Algorithm: Example (no Intercept)

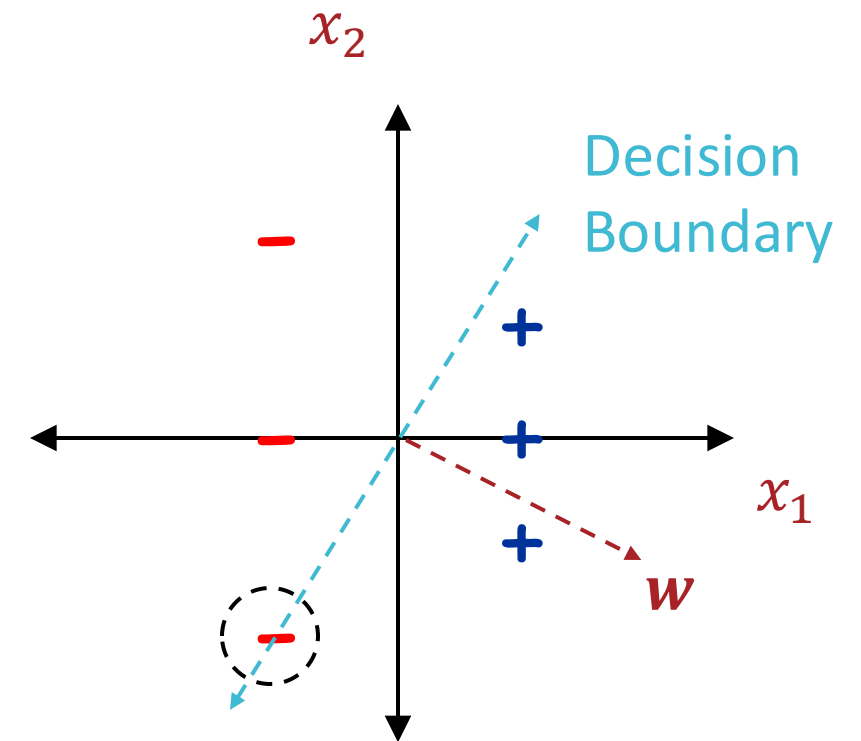
x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes
1	0	+	+	No
1	1	-	+	Yes
-1	0	-	-	No

$$w = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$



(Online) Perceptron Learning Algorithm: Example (no Intercept)

x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes
1	0	+	+	No
1	1	-	+	Yes
-1	0	-	-	No
-1	-2	+	-	Yes

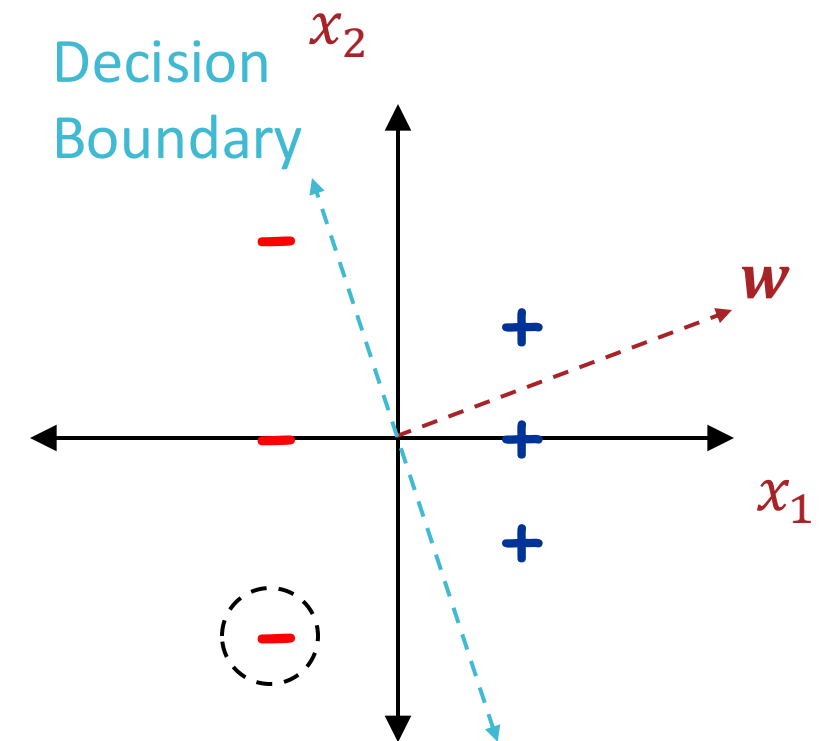


$$w = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$

$$w \leftarrow w + y^{(5)} x^{(5)} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} - \begin{bmatrix} -1 \\ -2 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

(Online) Perceptron Learning Algorithm: Example (no Intercept)

x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes
1	0	+	+	No
1	1	-	+	Yes
-1	0	-	-	No
-1	-2	+	-	Yes



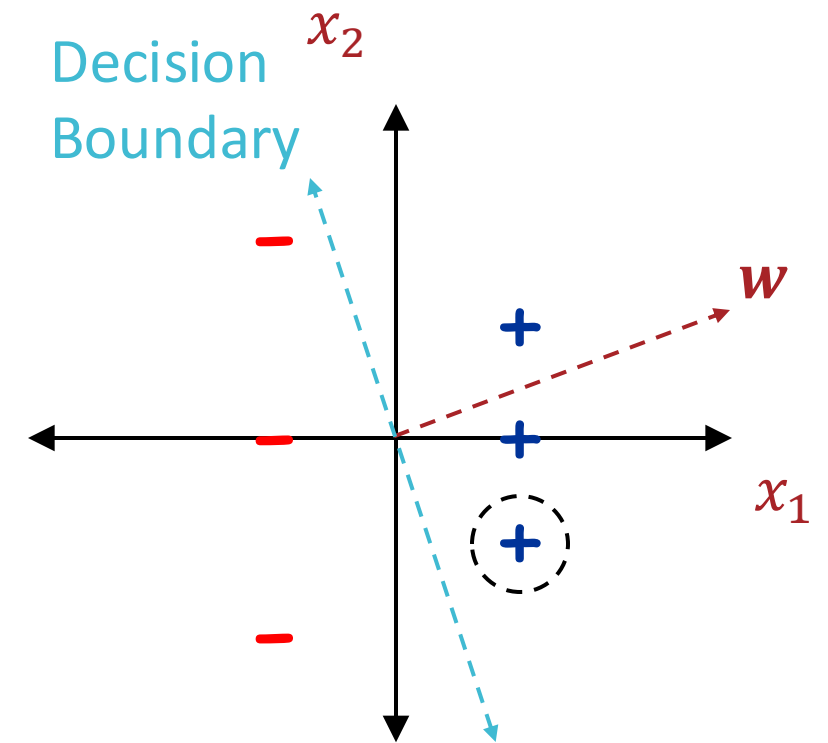
$$\mathbf{w} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$

$$\mathbf{w} \leftarrow \mathbf{w} + y^{(5)} \mathbf{x}^{(5)} = \begin{bmatrix} 2 \\ -1 \end{bmatrix} - \begin{bmatrix} -1 \\ -2 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

(Online) Perceptron Learning Algorithm: Example (no Intercept)

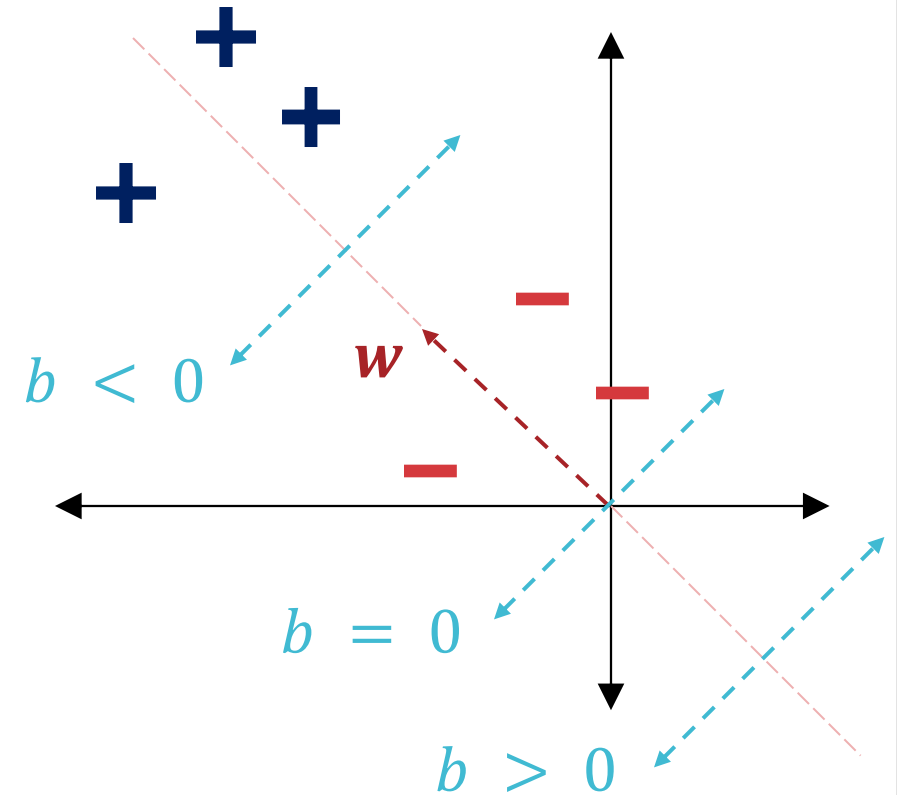
x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	+	-	Yes
1	0	+	+	No
1	1	-	+	Yes
-1	0	-	-	No
-1	-2	+	-	Yes
1	-1	+	+	No

$$w = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$



Updating the Intercept

- The intercept shifts the decision boundary off the origin
 - Increasing b shifts the decision boundary towards the negative side
 - Decreasing b shifts the decision boundary towards the positive side



Poll Question 1:

- **True or False:** Unlike Decision Trees and k -Nearest Neighbors, the Perceptron learning algorithm does not suffer from overfitting because it does not have any hyperparameters that could be over-tuned on a validation dataset.

Notational Hack

- If we add a 1 to the beginning of every example e.g.,

$$\mathbf{x}' = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix} \dots$$

- ... we can just fold the intercept into the weight vector!

$$\boldsymbol{\theta} = \begin{bmatrix} b \\ w_1 \\ w_2 \\ \vdots \\ w_D \end{bmatrix} \rightarrow \boldsymbol{\theta}^T \mathbf{x}' = \mathbf{w}^T \mathbf{x} + b$$

(Online) Perceptron Learning Algorithm

- Initialize the weight vector and intercept to all zeros:

$$\mathbf{w} = [0 \quad 0 \quad \dots \quad 0] \text{ and } b = 0$$

- For $t = 1, 2, 3, \dots$
 - Receive an unlabeled example, $\mathbf{x}^{(t)}$
 - Predict its label, $\hat{y} = \text{sign}(\mathbf{w}^T \mathbf{x} + b) = \begin{cases} +1 & \text{if } \mathbf{w}^T \mathbf{x} + b \geq 0 \\ -1 & \text{otherwise} \end{cases}$
 - Observe its true label, $y^{(t)}$
 - If we misclassified an example ($y^{(t)} \neq \hat{y}$):
 - $\mathbf{w} \leftarrow \mathbf{w} + y^{(t)} \mathbf{x}^{(t)}$
 - $b \leftarrow b + y^{(t)}$

(Online) Perceptron Learning Algorithm

- Initialize the parameters to all zeros:

$$\boldsymbol{\theta} = [0 \quad 0 \quad \dots \quad 0]$$

1 prepended
to $\mathbf{x}^{(t)}$

- For $t = 1, 2, 3, \dots$

- Receive an unlabeled example, $\mathbf{x}^{(t)}$

- Predict its label, $\hat{y} = \text{sign}(\boldsymbol{\theta}^T \mathbf{x}'^{(t)}) = \begin{cases} +1 & \text{if } \boldsymbol{\theta}^T \mathbf{x}'^{(t)} \geq 0 \\ -1 & \text{otherwise} \end{cases}$

- Observe its true label, $y^{(t)}$

- If we misclassified an example ($y^{(t)} \neq \hat{y}$):

- $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + y^{(t)} \mathbf{x}'^{(t)}$

Automatically handles
updating the intercept

(Online) Perceptron Learning Algorithm: Intuition

- Suppose $(x, y) \in \mathcal{D}$ is a misclassified training example and $y = +1$

- $\Theta^T x$ is negative

- Update: $\Theta_{\text{new}} = \Theta + \gamma x$

- After the update

$$\Theta_{\text{new}}^T x = (\Theta + \gamma x)^T x$$

$$= \Theta^T x + \gamma x^T x$$

$$\gamma > 0 \\ x^T x \geq 0$$

new prediction is "less negative" than the old prediction

(Online) Perceptron Learning Algorithm: Inductive Bias

- there are no outliers
- the data can be perfectly classified by a line
- more recent mistakes need to be corrected, potentially at the expense of older ^{correctly classified} data points
- there are only two ^{classes}
- the magnitude ^{non-overlapping} of the features matters

(Online) Perceptron Learning Algorithm

- Initialize the parameters to all zeros:

$$\boldsymbol{\theta} = [0 \quad 0 \quad \dots \quad 0]$$

- For $t = 1, 2, 3, \dots$
 - Receive an unlabeled example, $\mathbf{x}^{(t)}$
 - Predict its label, $\hat{y} = \text{sign}(\boldsymbol{\theta}^T \mathbf{x}'^{(t)})$
 - Observe its true label, $y^{(t)}$
 - If we misclassified an example ($y^{(t)} \neq \hat{y}$):
 - $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + y^{(t)} \mathbf{x}'^{(t)}$

(Batch) Perceptron Learning Algorithm

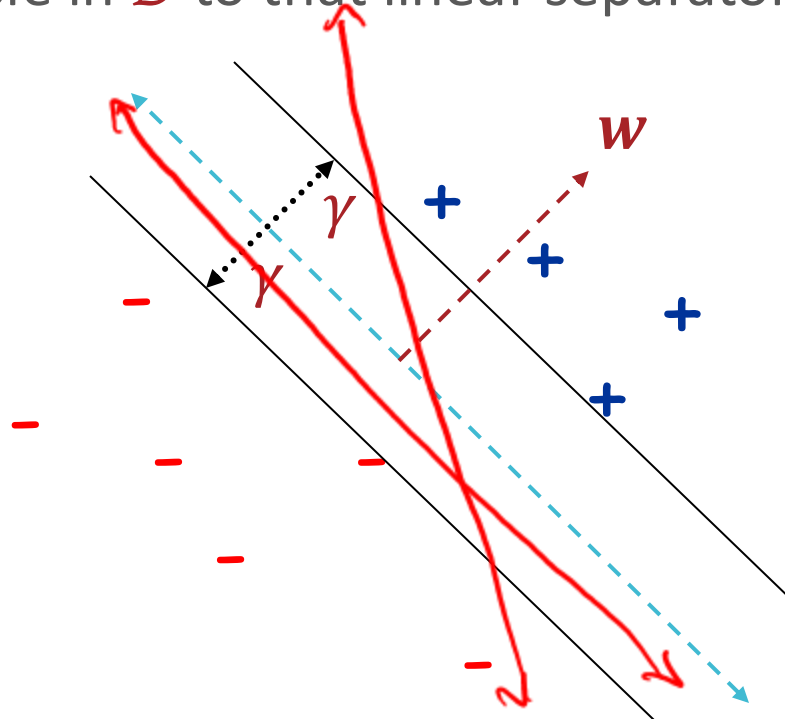
- Input: $\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})\}$
- Initialize the parameters to all zeros:

$$\boldsymbol{\theta} = [0 \quad 0 \quad \dots \quad 0]$$

- While **NOT CONVERGED**
 - For $t \in \{1, \dots, N\}$
 - Predict the label of $\mathbf{x}'^{(t)}$, $\hat{y} = \text{sign}(\boldsymbol{\theta}^T \mathbf{x}'^{(t)})$
 - Observe its true label, $y^{(t)}$
 - If we misclassified $\mathbf{x}'^{(t)}$ ($y^{(t)} \neq \hat{y}$):
 - $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + y^{(t)} \mathbf{x}'^{(t)}$

Perceptron Mistake Bound

- Definitions:
 - A dataset $\mathcal{D}$ is *linearly separable* if $\exists$ a linear decision boundary that perfectly classifies the examples in $\mathcal{D}$
 - The margin, γ , of a dataset $\mathcal{D}$ is the greatest possible distance between a linear separator and the closest example in $\mathcal{D}$ to that linear separator



Perceptron Mistake Bound

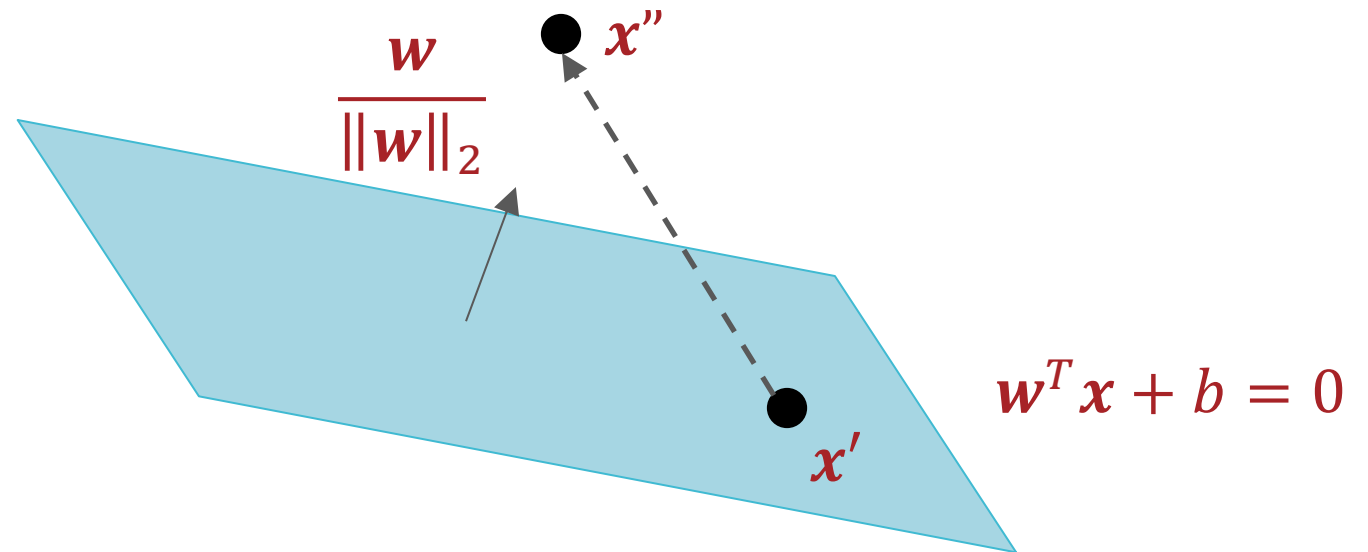
- Theorem: if the examples seen by the Perceptron Learning Algorithm (online and batch)
 1. lie in a ball of radius R (centered around the origin)
 2. have a margin of γ

then the algorithm makes at most $(R/\gamma)^2$ mistakes!

- Key Takeaway: if the training dataset is linearly separable, the batch Perceptron Learning Algorithm will converge (i.e., stop making mistakes on the training dataset or achieve 0 training error) in a finite number of steps!

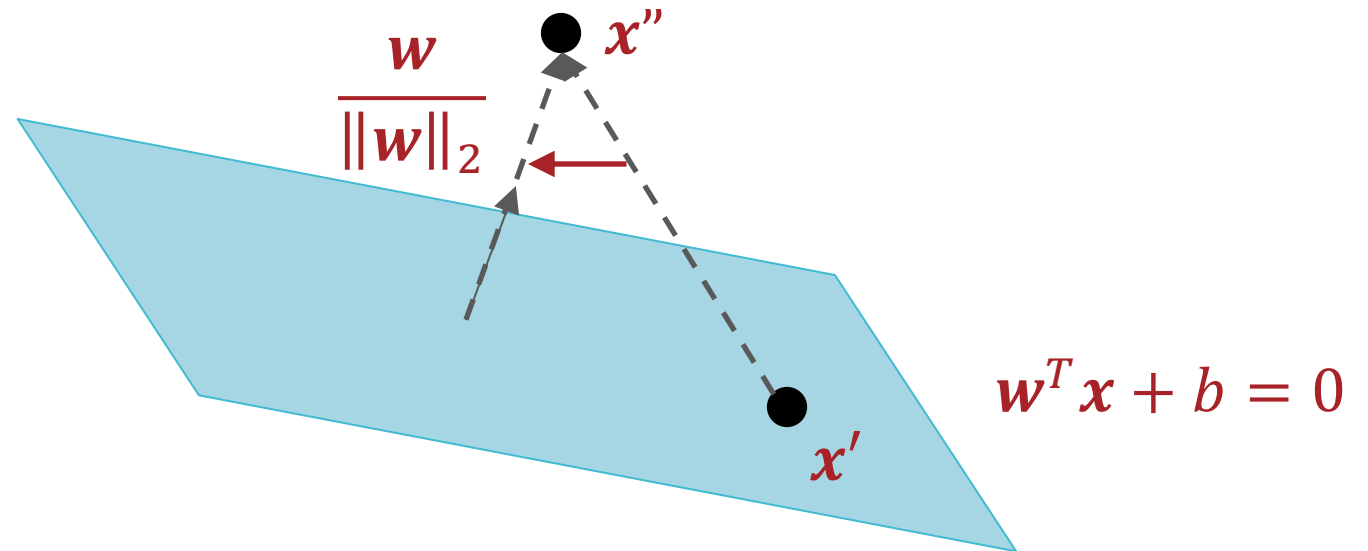
Computing the Margin

- Let $\mathbf{x}'$ be an arbitrary point on the hyperplane $\mathbf{w}^T \mathbf{x} + b = 0$ and let $\mathbf{x}''$ be an arbitrary point
- The distance between $\mathbf{x}''$ and $\mathbf{w}^T \mathbf{x} + b = 0$ is equal to the magnitude of the projection of $\mathbf{x}'' - \mathbf{x}'$ onto $\frac{\mathbf{w}}{\|\mathbf{w}\|_2}$, the unit vector orthogonal to the hyperplane



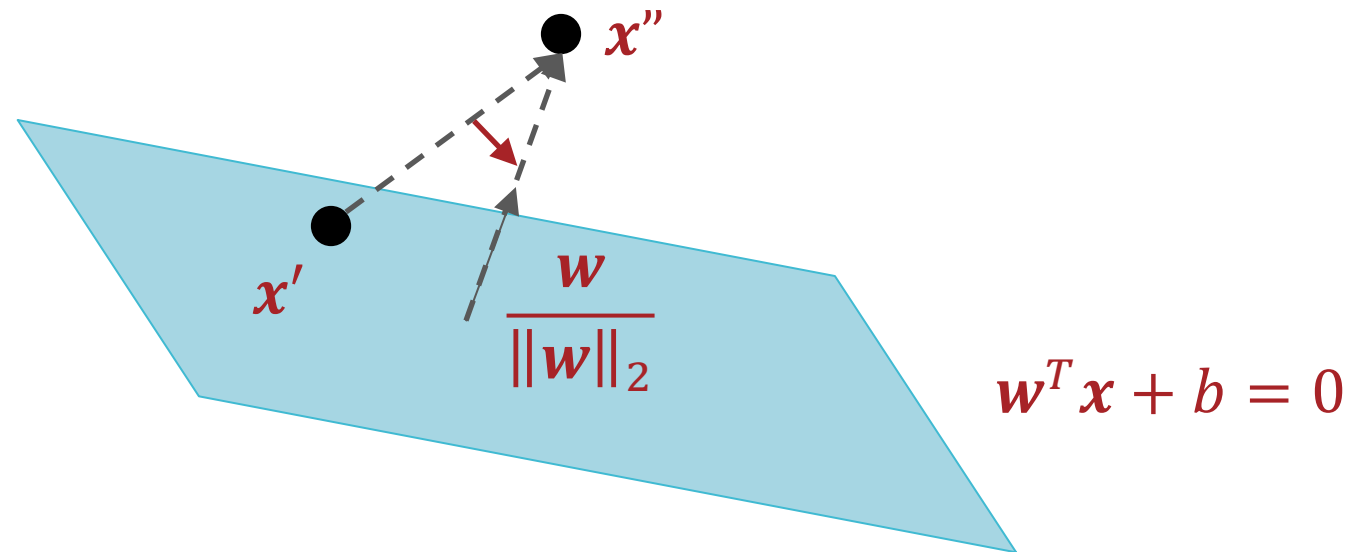
Computing the Margin

- Let $\mathbf{x}'$ be an arbitrary point on the hyperplane $\mathbf{w}^T \mathbf{x} + b = 0$ and let $\mathbf{x}''$ be an arbitrary point
- The distance between $\mathbf{x}''$ and $\mathbf{w}^T \mathbf{x} + b = 0$ is equal to the magnitude of the projection of $\mathbf{x}'' - \mathbf{x}'$ onto $\frac{\mathbf{w}}{\|\mathbf{w}\|_2}$, the unit vector orthogonal to the hyperplane



Computing the Margin

- Let $\mathbf{x}'$ be an arbitrary point on the hyperplane $\mathbf{w}^T \mathbf{x} + b = 0$ and let $\mathbf{x}''$ be an arbitrary point
- The distance between $\mathbf{x}''$ and $\mathbf{w}^T \mathbf{x} + b = 0$ is equal to the magnitude of the projection of $\mathbf{x}'' - \mathbf{x}'$ onto $\frac{\mathbf{w}}{\|\mathbf{w}\|_2}$, the unit vector orthogonal to the hyperplane



Computing the Margin

- Let $\mathbf{x}'$ be an arbitrary point on the hyperplane and let $\mathbf{x}''$ be an arbitrary point
- The distance between $\mathbf{x}''$ and $\mathbf{w}^T \mathbf{x} + b = 0$ is equal to the magnitude of the projection of $\mathbf{x}'' - \mathbf{x}'$ onto $\frac{\mathbf{w}}{\|\mathbf{w}\|_2}$, the unit vector orthogonal to the hyperplane

$$\left| \frac{\mathbf{w}^T (\mathbf{x}'' - \mathbf{x}')}{\|\mathbf{w}\|_2} \right| = \frac{|\mathbf{w}^T \mathbf{x}'' - \mathbf{w}^T \mathbf{x}'|}{\|\mathbf{w}\|_2} = \frac{|\mathbf{w}^T \mathbf{x}'' + b|}{\|\mathbf{w}\|_2}$$

Perceptron Learning Objectives

You should be able to...

- Explain the difference between online learning and batch learning
- Implement the perceptron algorithm for binary classification [CIML]
- Determine whether the perceptron algorithm will converge based on properties of the dataset, and the limitations of the convergence guarantees
- Describe the inductive bias of perceptron and the limitations of linear models
- Draw the decision boundary of a linear model
- Identify whether a dataset is linearly separable or not
- Defend the use of a bias term in perceptron (shifting points after projection onto weight vector)