10-301/601: Introduction to Machine Learning Lecture 6 – Perceptron

Henry Chai & Matt Gormley

9/19/22

Q & A:

Suppose we do model selection using a validation dataset. For our final model, should we train using *both* training and validation datasets?

- Yes, absolutely! So really the sketch from last lecture should look something like:
    1. Split $\mathcal{D}$ into $\mathcal{D}_{train} \cup \mathcal{D}_{val} \cup \mathcal{D}_{test}$
    2. Learn classifiers using $\mathcal{D}_{train}$
    3. Evaluate models using $\mathcal{D}_{val}$ and choose the one with lowest *validation* error:
    4. **Learn a new classifier from the best model using $\mathcal{D}_{train} \cup \mathcal{D}_{val}$**
    5. Optionally, use $\mathcal{D}_{test}$ to estimate the true error

Q & A:

Can we use KNNs with categorical features?

- Again, yes! We can either convert categorical features into binary ones or use a distance metric that works over categorical features e.g., the Hamming distance:

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

Front Matter

- Announcements:
  - HW2 released 9/7, due 9/19 (today!) at 11:59 PM
  - HW3 released 9/21, due 9/28 at 11:59 PM
    - **Only two grace days allowed on HW3**

Recall: Fisher Iris Dataset

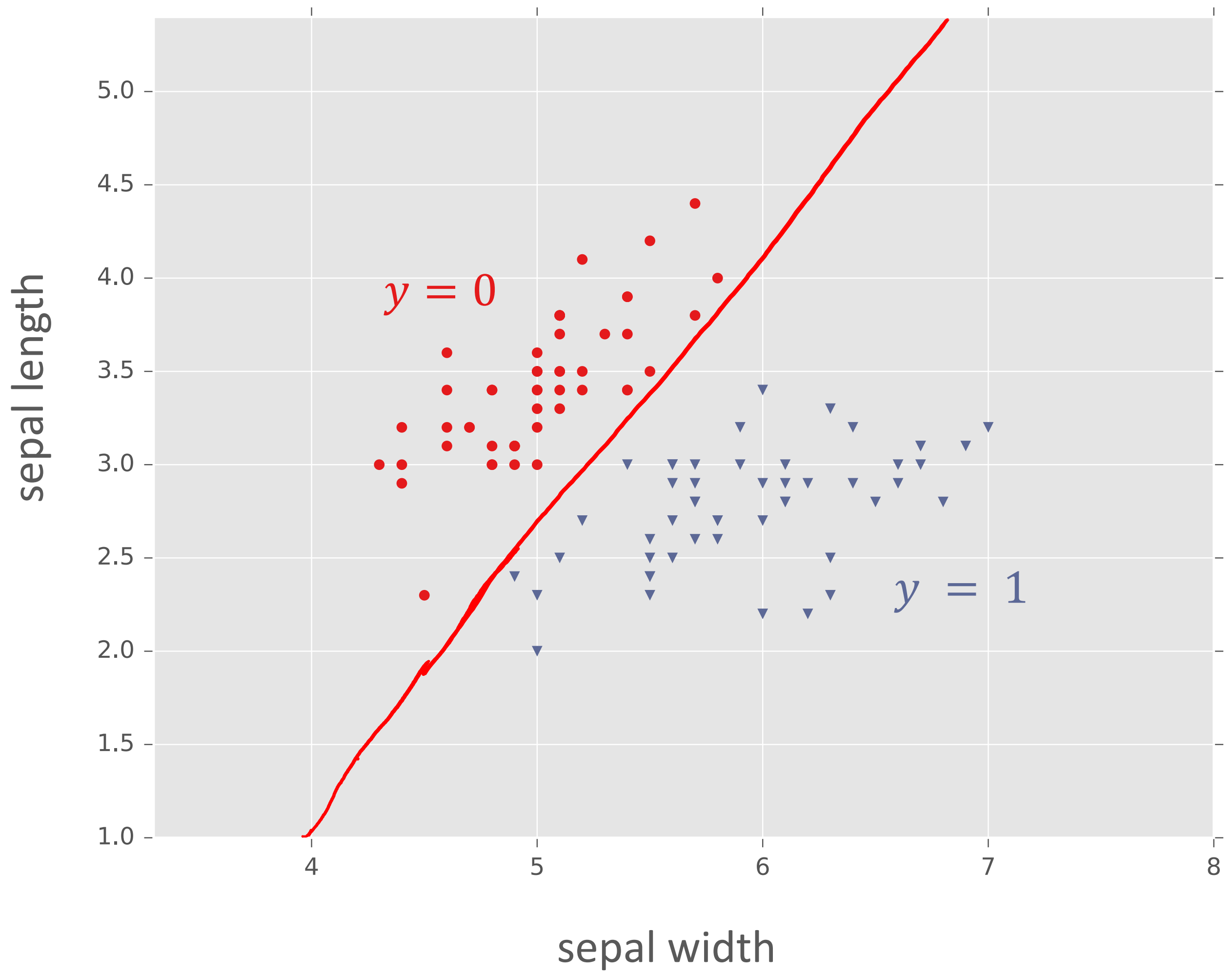


Figure courtesy of Matt Gormley

Linear Algebra Review

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

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_D \end{bmatrix} \text{ and } \boldsymbol{a}^T = \begin{bmatrix} a_1 & a_2 & \cdots & a_D \end{bmatrix}$$

- The dot product between two D-dimensional vectors is

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

- The $L2$-norm of $\boldsymbol{a} = \|\boldsymbol{a}\|_2 = \sqrt{\boldsymbol{a}^T\boldsymbol{a}}$
- Two vectors are *orthogonal* iff

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

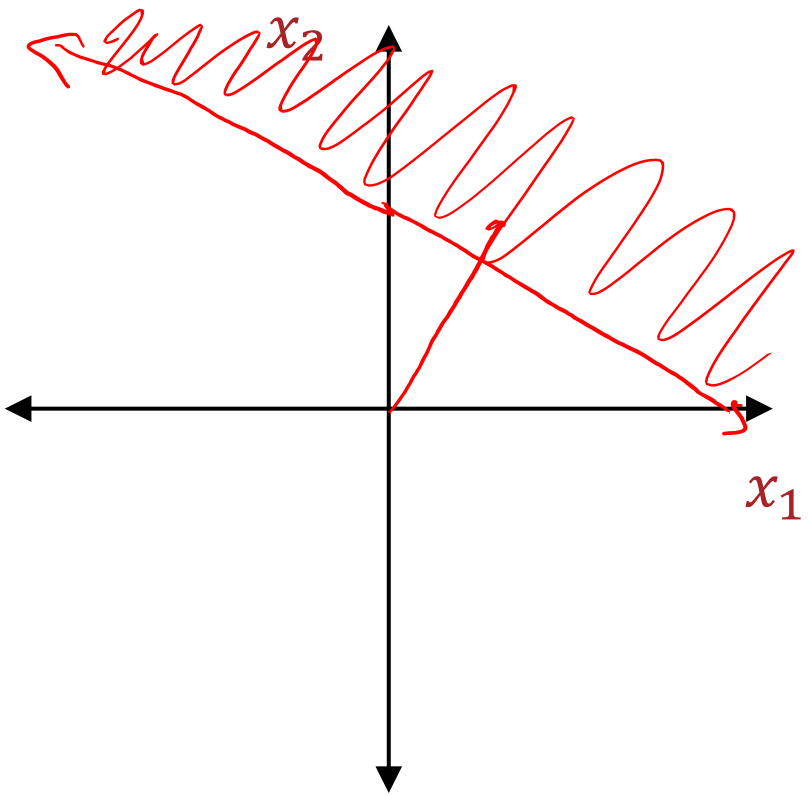
Geometry Warm-up

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

$$w_1 x_1 + w_2 x_2 + b > 0$$

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

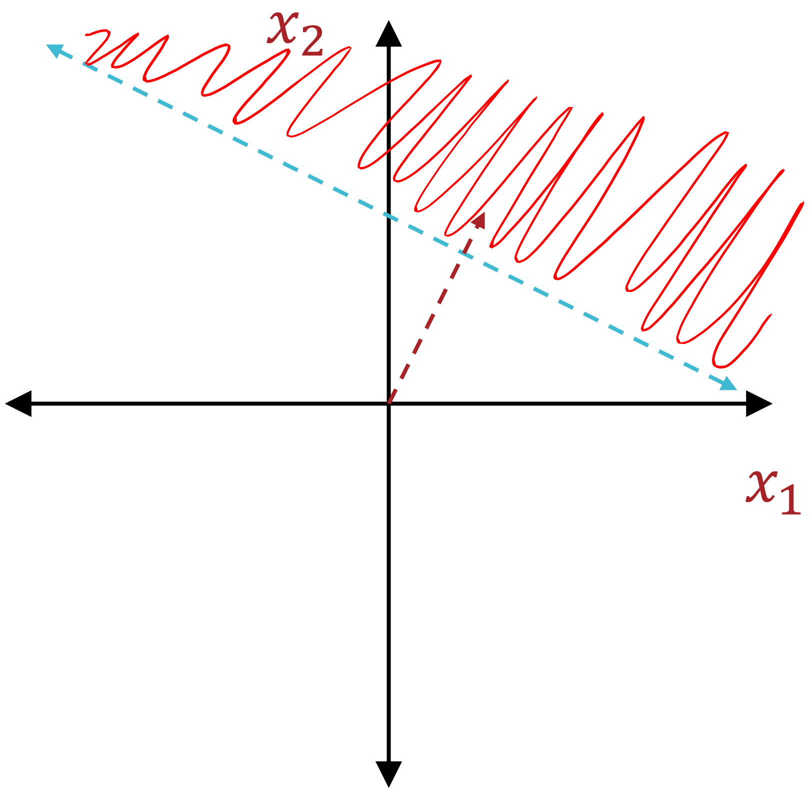
2. Then draw the vector $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$



Geometry Warm-up

1. On the axes below, draw the region corresponding to
$$w_1 x_1 + w_2 x_2 + b > 0$$
where $w_1 = 1$, $w_2 = 2$ and $b = -4$.

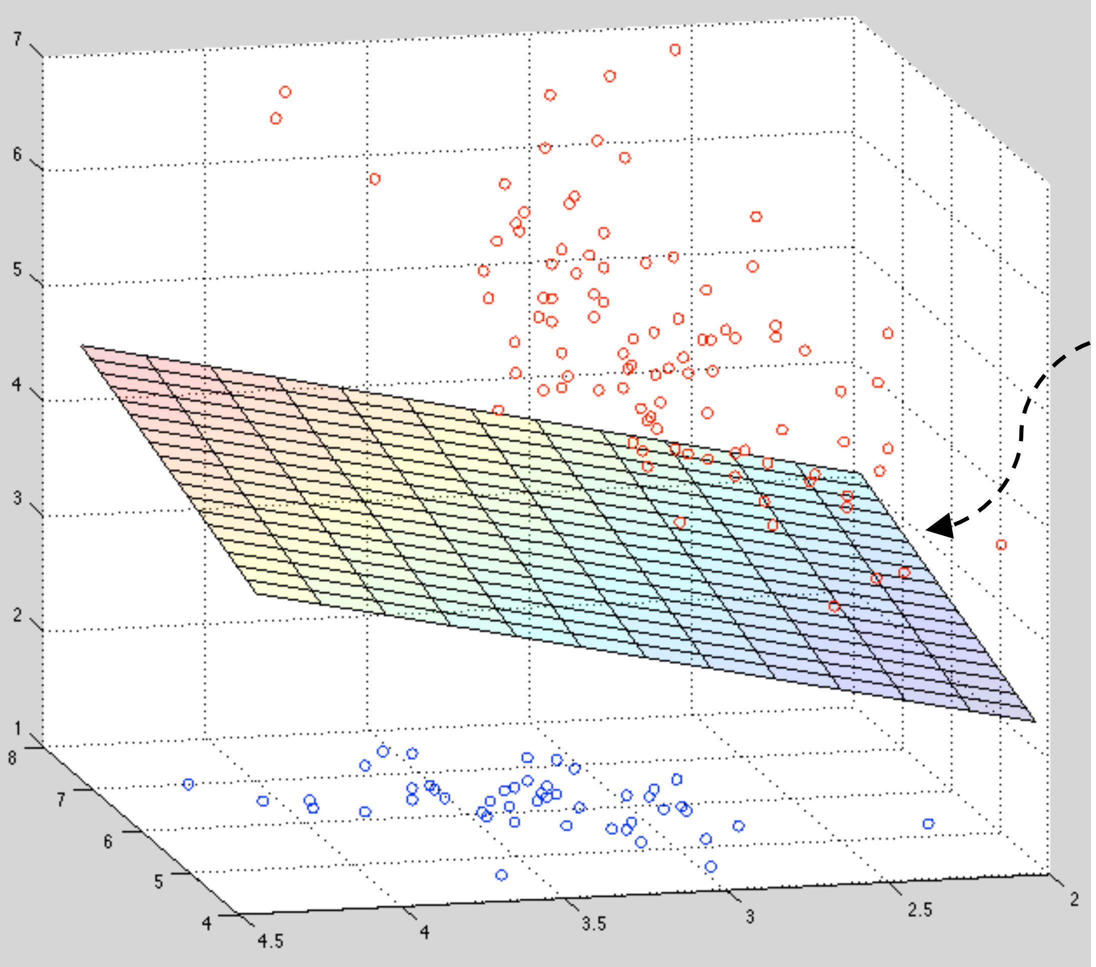
2. Then draw the vector $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$



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

Linear Decision Boundaries: Example



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

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

Figure courtesy of Matt Gormley

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:

– online shopping: customer purchases as new examples

– predicting success in classes

– recommender systems

– siri (any voice agent)

Online Learning: Setup

- For $t = 1, 2, 3, \ldots$
  - Receive an unlabeled example, $\boldsymbol{x}^{(t)}$
  - Predict its label, $\hat{y} = h_{\boldsymbol{w},b}\left(\boldsymbol{x}^{(t)}\right)$
  - Observe its true label, $y^{(t)}$
  - Pay a penalty if we made a mistake, $\hat{y} \neq y^{(t)}$
  - Update the parameters, $\boldsymbol{w}$ and b

- Goal: minimize the number of mistakes made

(Online) Perceptron Learning Algorithm

- Initialize the weight vector and intercept to all zeros:

$$\boldsymbol{w} = [0 \quad 0 \quad \cdots \quad 0] \text{ and } b = 0$$

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

(Online) Perceptron Learning Algorithm

- Initialize the weight vector and intercept to all zeros:

$$\boldsymbol{w} = [0 \quad 0 \quad \cdots \quad 0] \text{ and } b = 0$$

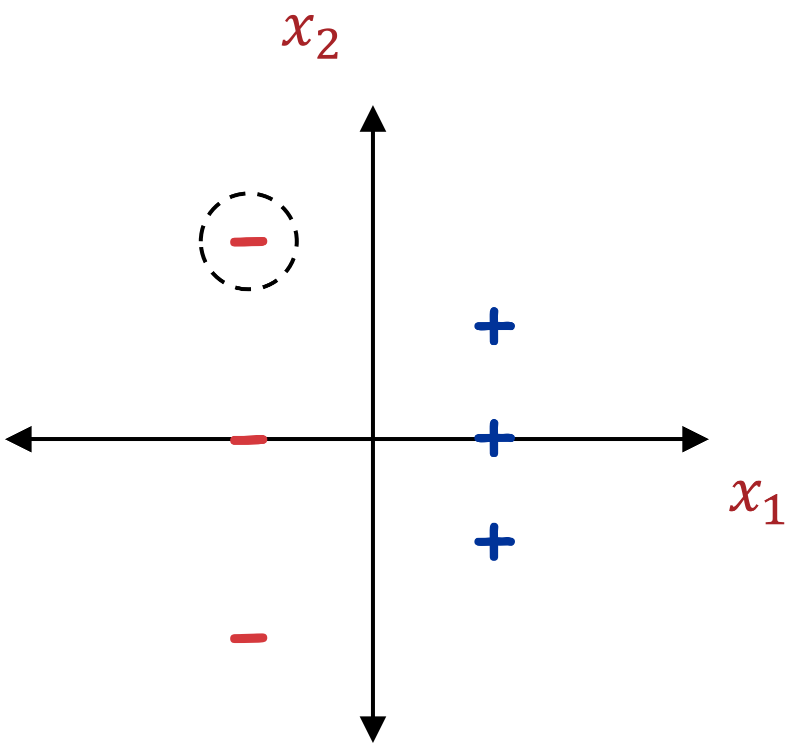
- For $t = 1, 2, 3, \ldots$
  - Receive an unlabeled example, $\boldsymbol{x}^{(t)}$
  - Predict its label, $\hat{y} = \text{sign}(\boldsymbol{w}^T\boldsymbol{x} + b) = \begin{cases} +1 \text{ if } \boldsymbol{w}^T\boldsymbol{x} + b \geq 0 \\ -1 \text{ otherwise} \end{cases}$
  - Observe its true label,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

(1)

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

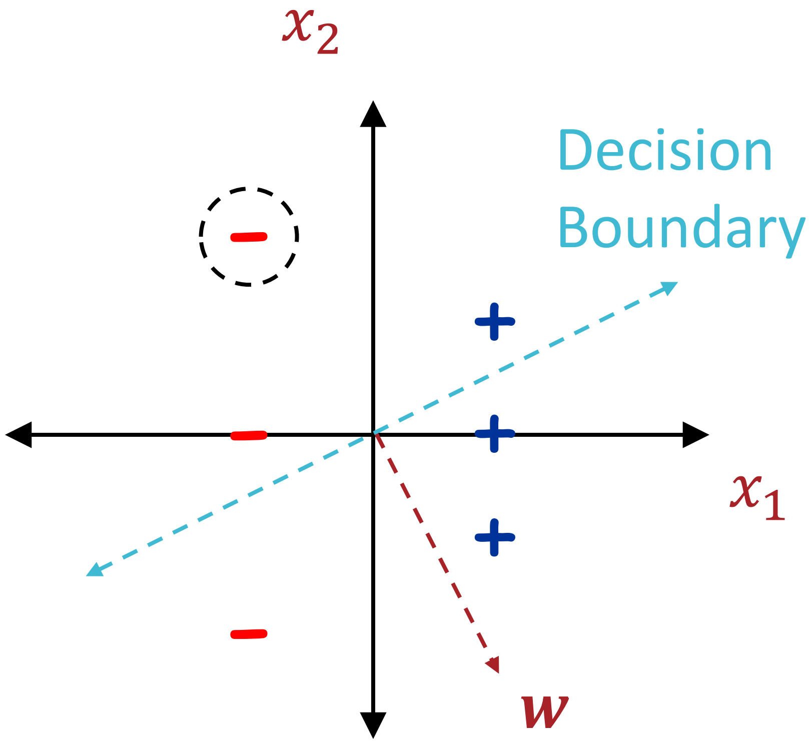
x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	$+$	$-$	Yes

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



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

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



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

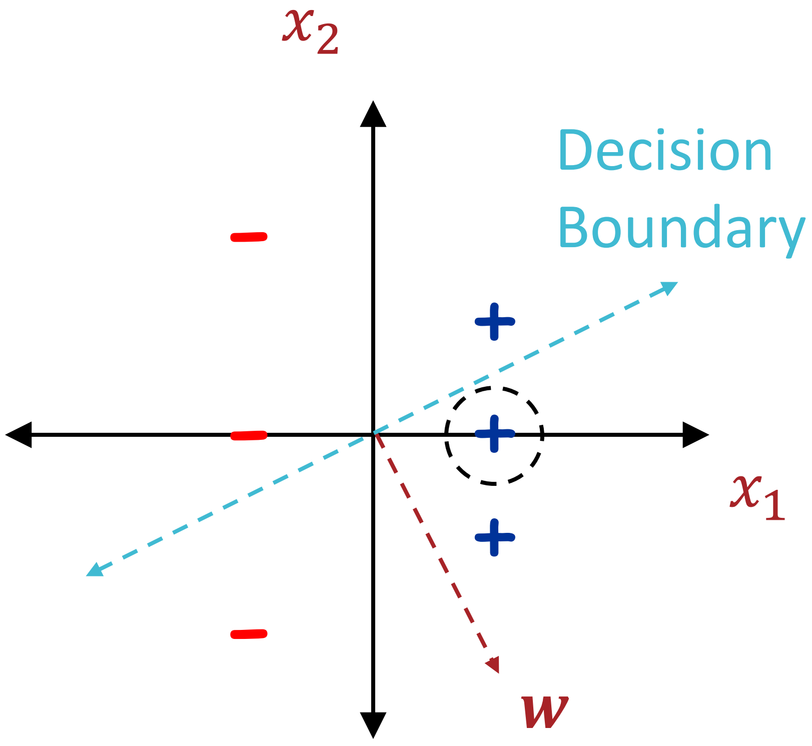
$$\boldsymbol{w} \leftarrow \boldsymbol{w} + y^{(1)}\boldsymbol{x}^{(1)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

Example courtesy of Nina Balcan

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

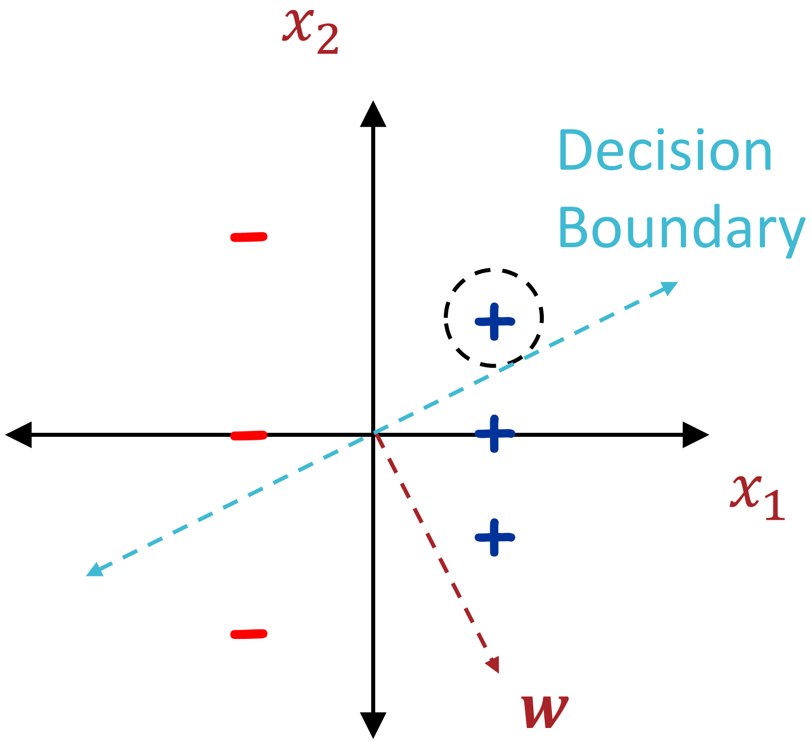
x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	$+$	$-$	Yes
1	0	$+$	$+$	No

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



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

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

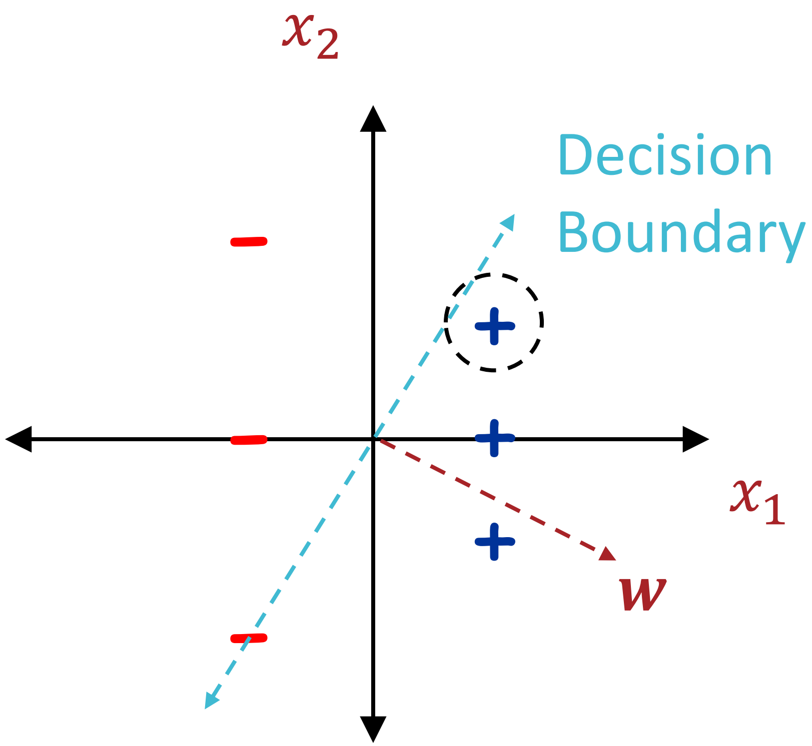


$$\boldsymbol{w} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

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

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

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



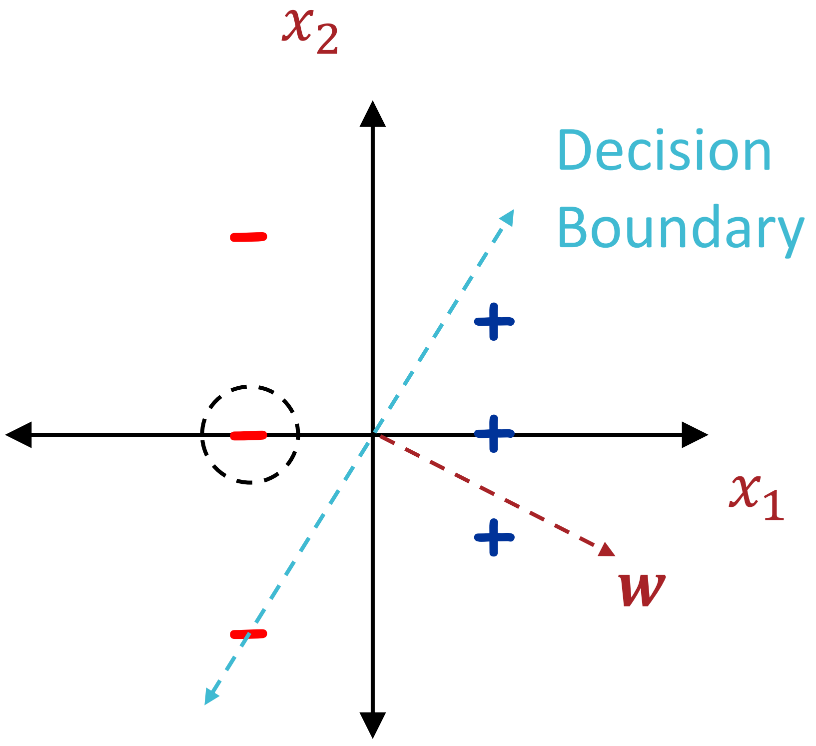
$$\boldsymbol{w} = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

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

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

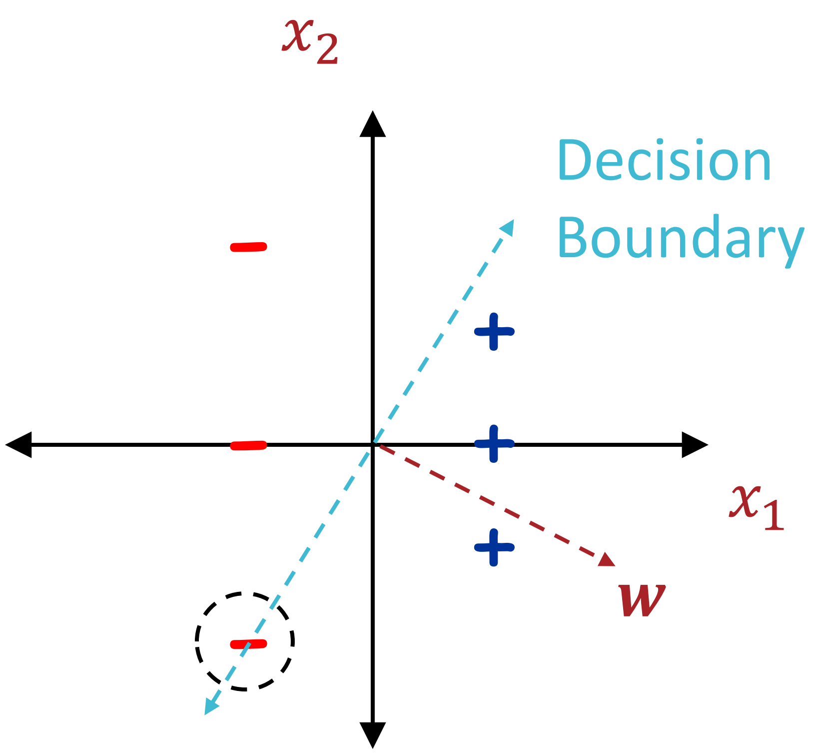
x_1	x_2	$\hat{y}$	y	Mistake?
-1	2	$+$	$-$	Yes
1	0	$+$	$+$	No
1	1	$-$	$+$	Yes
-1	0	$-$	$-$	No

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



(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



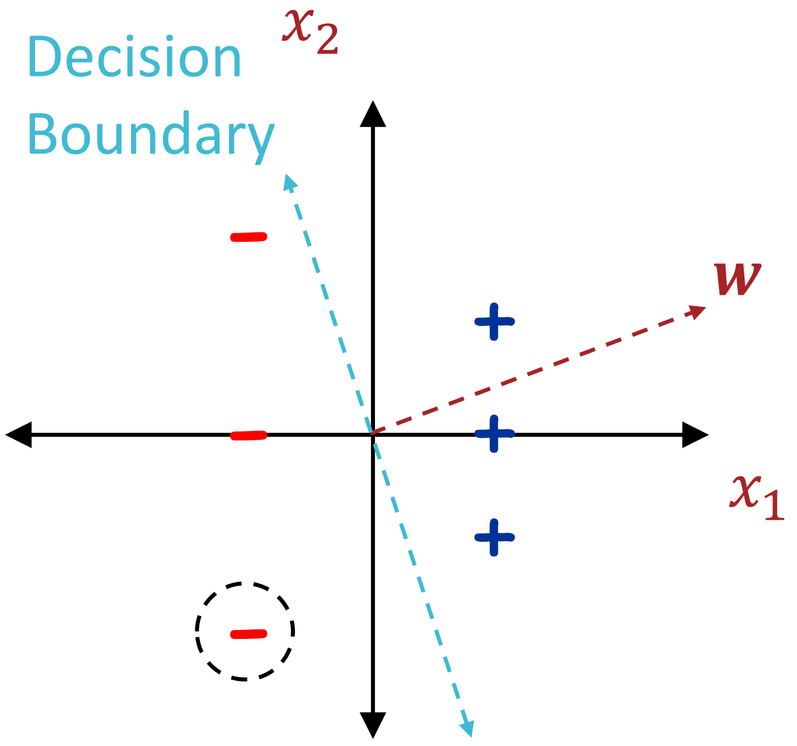
$$\boldsymbol{w} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$

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

Example courtesy of Nina Balcan

(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



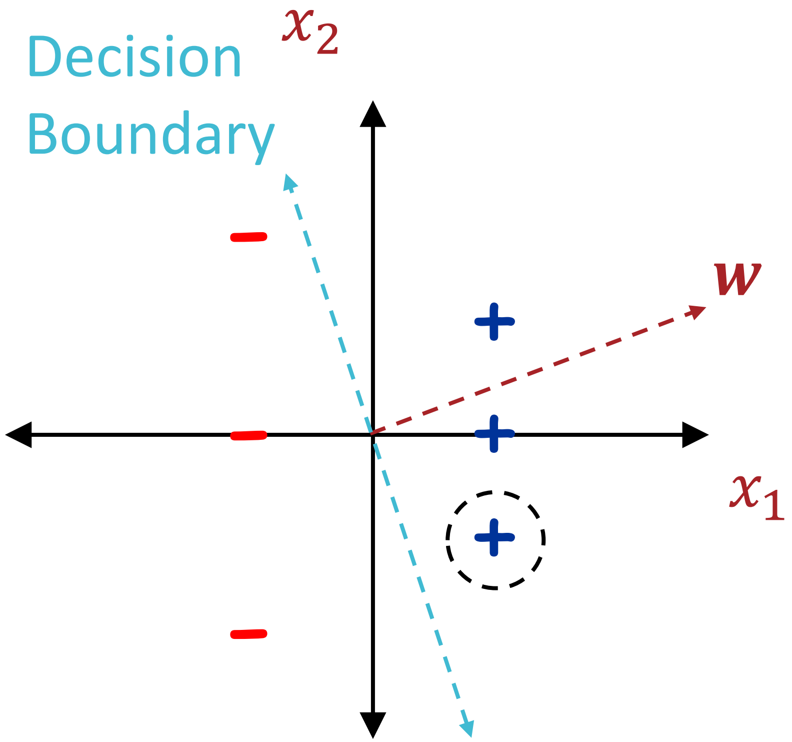
$$\boldsymbol{w} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$

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

(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

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



Example courtesy of Nina Balcan

Poll Question 1:

- **True or False**: Unlike Decision Trees and k-Nearest Neighbors, the Perceptron algorithm does not suffer from overfitting because it does not have any hyperparameters that could be over-tuned on the training data.
  A. True
  B. True and False **(TOXIC)**
  C. False

Notational Hack

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

$$\boldsymbol{x}' = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix} \ldots$$

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

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

(Online) Perceptron Learning Algorithm

- Initialize the weight vector and intercept to all zeros:

$$\boldsymbol{w} = [0 \quad 0 \quad \cdots \quad 0] \text{ and } b = 0$$

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

(Online) Perceptron Learning Algorithm

- Initialize the parameters to all zeros:

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

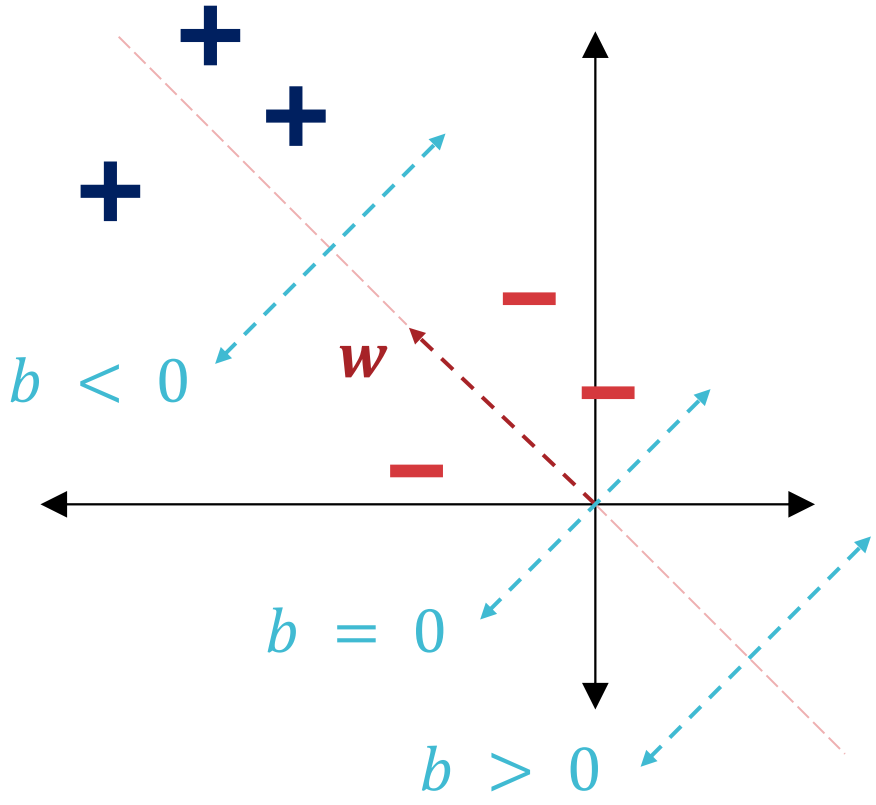
1 prepended to $\boldsymbol{x}^{(t)}$

- For $t = 1, 2, 3, \ldots$
  - Receive an unlabeled example, $\boldsymbol{x}^{(t)}$
  - Predict its label, $\hat{y} = \text{sign}\left(\boldsymbol{\theta}^T \boldsymbol{x}'^{(t)}\right) = \begin{cases} +1 \text{ if } \boldsymbol{\theta}^T \boldsymbol{x}'^{(t)} \geq 0 \\ -1 \text{ otherwise} \end{cases}$
  - Observe its true label, $y^{(t)}$
  - If we misclassified an example ($y^{(t)} \neq \hat{y}$):
    - $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + y^{(t)} \boldsymbol{x}'^{(t)}$

Automatically handles updating the intercept

Updating the Intercept

- The intercept shifts the decision boundary off the origin
  - Increasing b shifts the decision boundary towards the negative side (in the opposite direction of $\boldsymbol{w}$)
  - Decreasing b shifts the decision boundary towards the positive side (in the direction of $\boldsymbol{w}$)



(Online) Perceptron Learning Algorithm: Inductive Bias

1. More recent mistakes are more important than older ones (and should be corrected immediately)
2. The decision boundary should be linear
3. The features should behave similarly

(Online) Perceptron Learning Algorithm

- Initialize the parameters to all zeros:

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

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

(Batch) Perceptron Learning Algorithm

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

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

- While NOT CONVERGED ← Convergence — θ does not change in one full pass through $\mathcal{D}$ ⇒ achieve 0 training error
  - For $t \in \{1, \dots, N\}$
    - Predict the label of $\boldsymbol{x}'^{(t)}$, $\hat{y} = \text{sign}\left(\boldsymbol{\theta}^T \boldsymbol{x}'^{(t)}\right)$
    - Observe its true label, $y^{(t)}$
    - If we misclassified $\boldsymbol{x}'^{(t)}$ $(y^{(t)} \neq \hat{y})$:
      - $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + y^{(t)} \boldsymbol{x}'^{(t)}$

Poll Question 2:

- **True or False**: The parameter vector $\boldsymbol{w}$ learned by the batch Perceptron Learning Algorithm can be **written as a linear combination** of the examples, i.e.,

$$\boldsymbol{w} = c_1 \boldsymbol{x}^{(1)} + c_2 \boldsymbol{x}^{(2)} + \cdots + c_N \boldsymbol{x}^{(N)}$$

A. True if you replace “linear” with “polynomial”

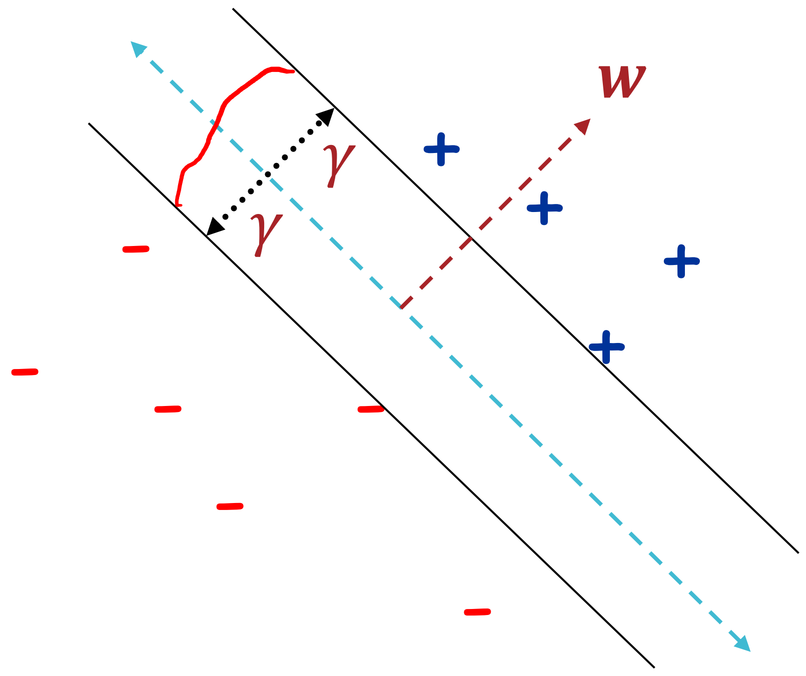
B. True and False **(TOXIC)**

C. True

D. False

Perceptron Mistake Bound

- Definitions:
  - A dataset $\mathcal{D}$ is *linearly separable* if ∃ 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

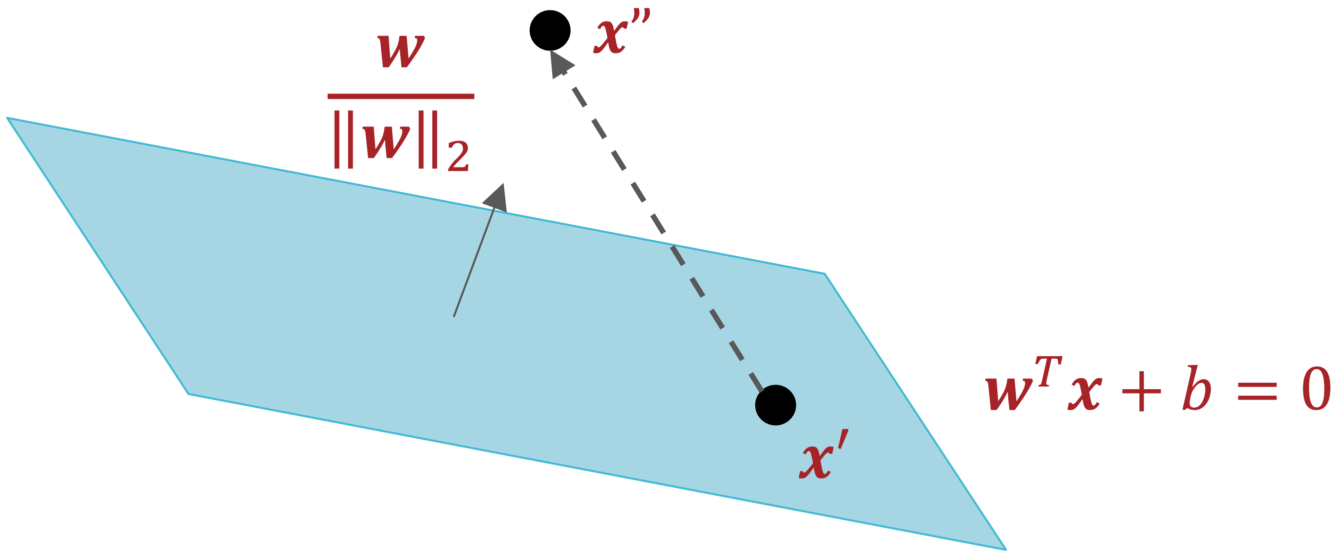
- Theorem: if the examples seen by the Perceptron Learning Algorithm (online and batch)
  1. lie in a ball of radius R (centered around the origin)
  2. have a margin of γ

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!

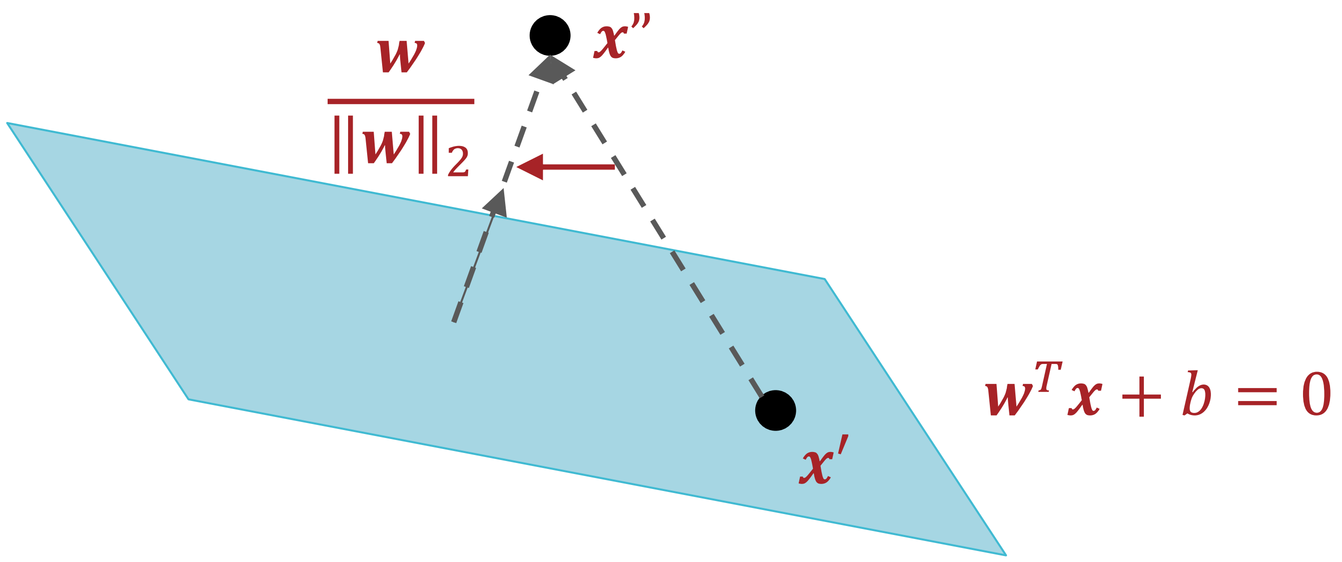
Computing the Margin

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



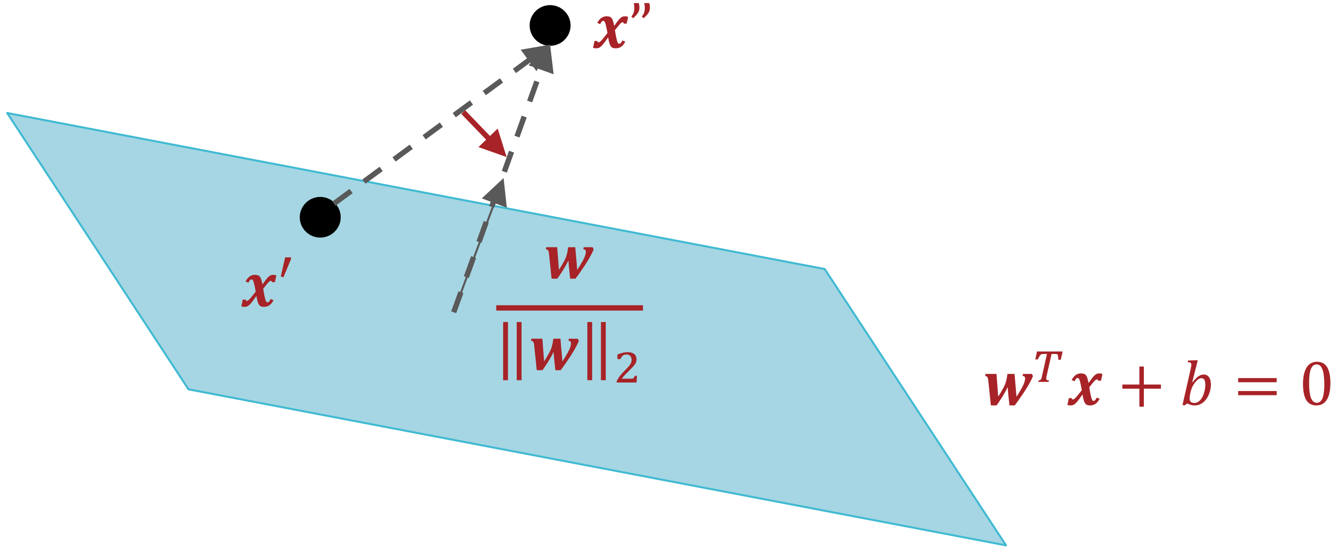
Computing the Margin

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



Computing the Margin

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



Computing the Margin

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

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

Model Selection 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)