# Perceptron and SVM

Sewoong Oh

CSE446
University of Washington

## Perceptron algorithm

# The perceptron algorithm

- One of the oldest algorithm in machine learning introduced by Rosenblatt in 1958
- the **perceptron algorithm** is an **online algorithm** for learning a linear classifier
$$\hat{y} \;=\; f_w(x) \;=\; w_0 + w_1 x[1] + \cdots$$
- an online algorithm is an iterative algorithm that takes a single paired example $(x_t, y_t)$ at $t$-iteration, and computes the updated iterate $w^{(t+1)}$ according to some rule
- for example, stochastic gradient descent algorithm with a mini-batch size of $m = 1$, that runs for $n$ iterations and stops, can be considered an online algorithm (where $n$ is the number of training samples)

Interactive demo: https://codepen.io/bagrounds/full/wdqypY

# The perceptron algorithm

- given a stream of samples $\{(x_1, y_1), (x_2, y_2), (x_3, y_3)\dots\}$
- initialize: $w^{(0)} = 0$,
  (one could normalize all $x_t = [x_t[0] = 1, x_t[1], x_t[2], \cdots]^T$'s to be norm 1)
- notice that we indexed samples by subscript $t$ to match the iterations, as it is an online algorithm
- for $t = 0, 1, \cdots$
  - if no mistake on current sample $(x_t, y_t)$, i.e. $y_t = \text{sign}((w^{(t)})^T x_t)$
    then do nothing
    $$w^{(t+1)} \leftarrow w^{(t)}$$
  - If mistake, then
    $$w^{(t+1)} \leftarrow w^{(t)} + y_t x_t$$

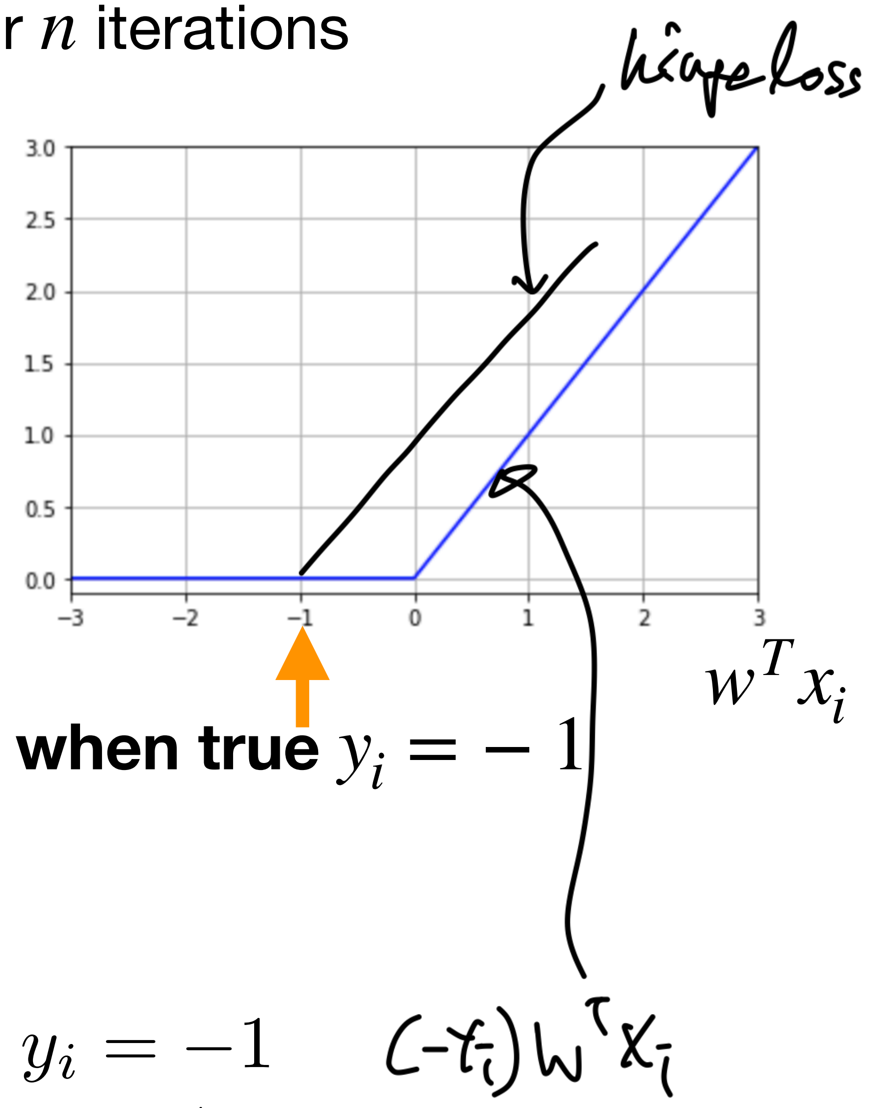
- this makes sense, as if $y_t = -1$ and $(w^{(t)})^T x_t > 0$ (for example), then
$$(w^{(t+1)})^T x_t = (w^{(t)})^T x_t + \underbrace{y_t \|x_t\|^2}_{<0}$$
- every time we make a mistake, the parameter move in a direction that is less likely to make the same mistake

# Can we derive perceptron algorithm?

- to get an online algorithm from gradient descent, suppose we apply stochastic gradient descent with mini-batch size $m = 1$, and run the algorithm for $n$ iterations

- Consider a **ReLU loss** is
  $\ell(w^T x_i, y_i) \ = \ \max\{-y_i\,(w^T x_i)\,,0\}$

- $y_i(w^T x_i)$ is also known as margin, and minimizing the ReLU loss is trying to maximize the margin of the current point
- the sub-gradient is

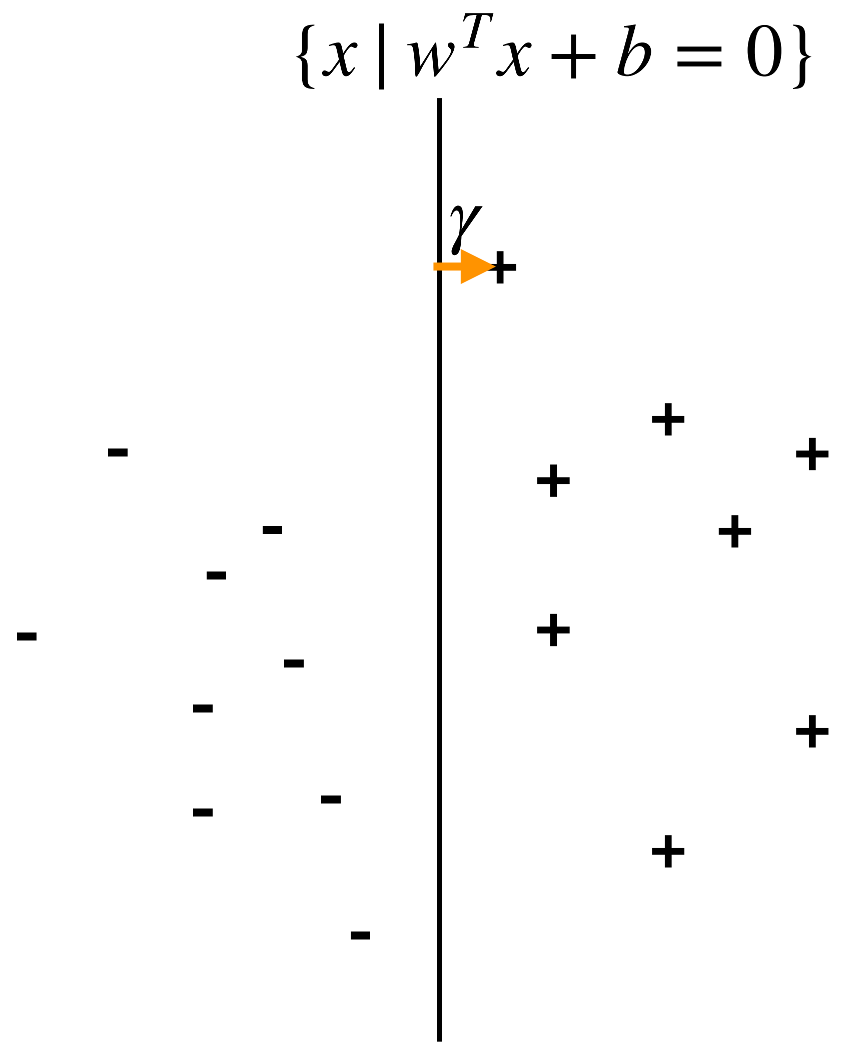


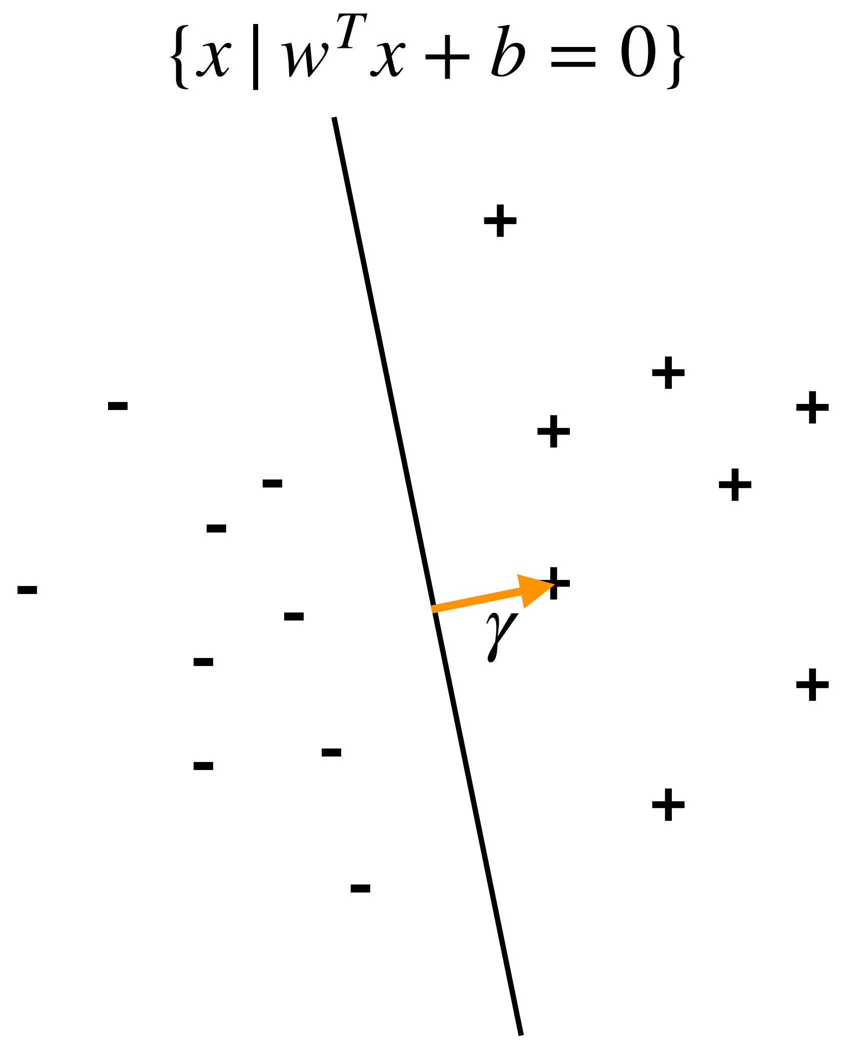
$$\partial\ell(w^T x_i, y_i) \ = \begin{cases} 0 & \text{if } y_i(w^T x_i) > 0 \\ -y_i x_i & \text{if } y_i(w^T x_i) < 0 \\ [0\,,\,+1]\,x_i & \text{if } y_i(w^T x_i) = 0 \text{ and } y_i = -1 \\ [-1\,,\,0]\,x_i & \text{if } y_i(w^T x_i) = 0 \text{ and } y_i = 1 \end{cases}$$

- the sub-gradient descent update with step size one is
  $w^{(t+1)} \leftarrow w^{(t)} + y_t x_t$
  only when there is a mistake, with a specific choice of updating with sub-gradient $\partial\ell(w^T x_i, y_i) = 0$ at the non-differentiable points

# Linear separability and margin

- a set of data is **linearly separable with a margin** $\gamma > 0$ if there exists a hyperplane $\{x \mid w^T x + b = 0\}$ such that the closest point to the hyperplane is at least distance $\gamma$ away, and positive and negative examples are all correctly separated





- **maximum margin** of a given dataset is the largest $\gamma$ achievable that linearly separates the dataset with margin $\gamma$

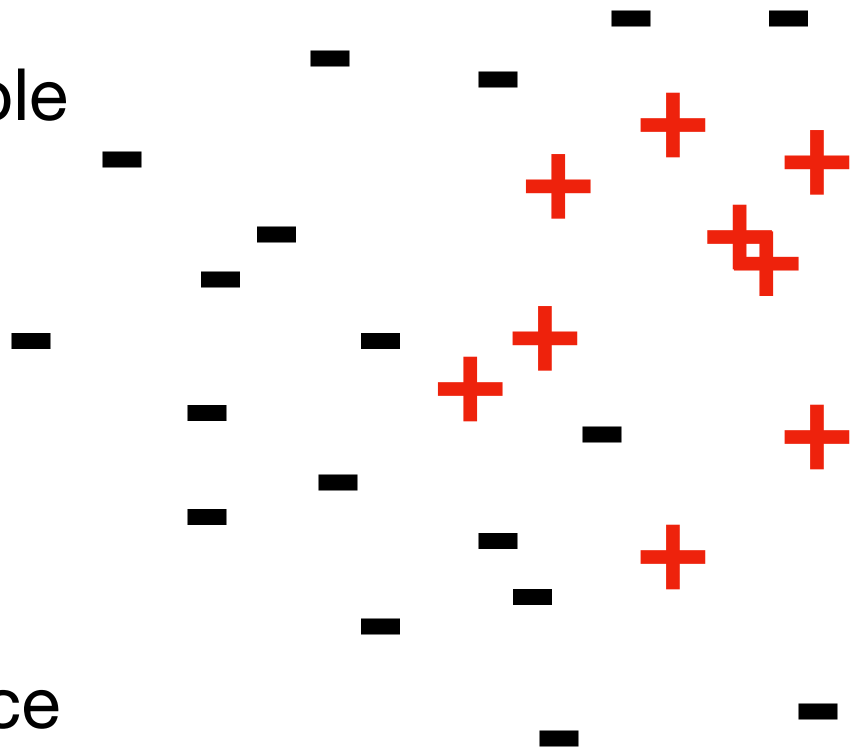
## Perceptron analysis

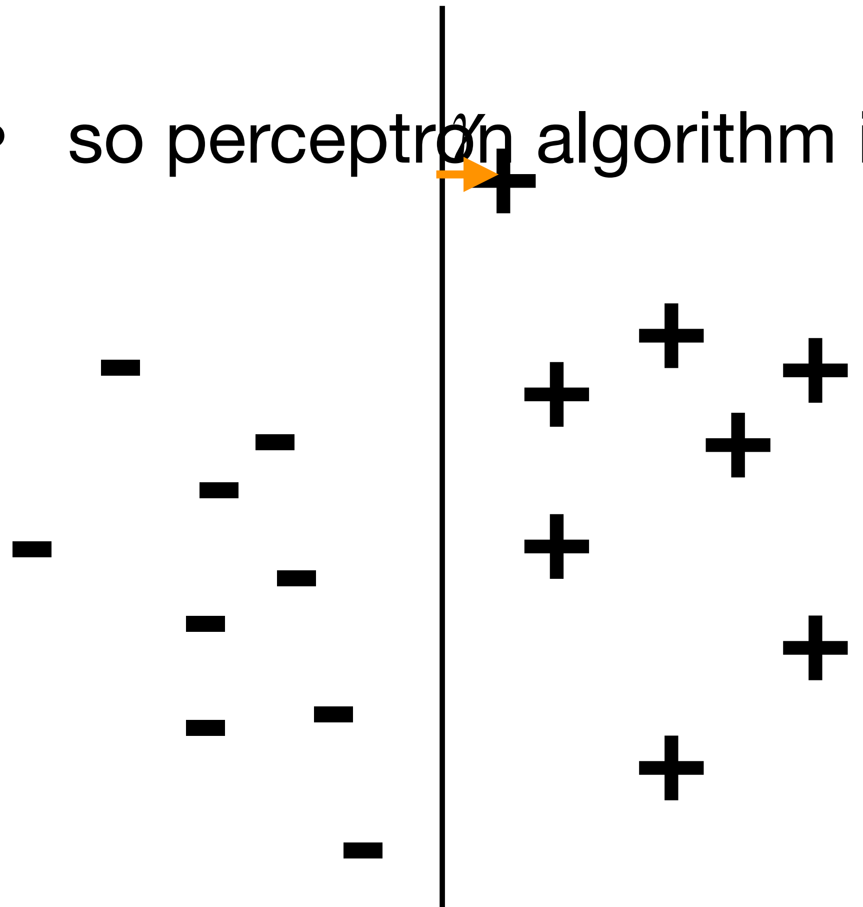
- consider an epoch based perceptron algorithm, where we run repeat the perceptron algorithm for many epochs, where an epoch is one run of perceptron algorithm that sees all training data exactly once
- Theorem [Block,Novicoff, 1962]
    - given a dataset which is linearly separable with margin $\gamma$
    - suppose each input vector has norm bounded by
      $$\|x_i\|_2^2 \leq R$$
      for all $i$
    - then, the number of mistakes made by the final output of a perceptron algorithm on the training data is upper bounded by
      $$\text{the number of mistakes made during the training process} \leq \frac{R^2}{\gamma^2}$$
    - this does not depend on the number of samples or the number of epochs
    - or the order the data was shown
    - Even if we run for many epochs, the algorithm converges and stops changing after a certain number of mistakes have been made

# Proof of the “mistake lemma”

- let $M_t$ be the number of mistakes made at time $t$
- if we make a mistake with $w^{(t)}$ on data $(x_t, y_t)$,
  then observe that $y_t((w^{(t-1)})^T x_t) \leq 0$
- let $w_*$ be the optimal linear separator with $\|w_*\|_2 = 1$,
  achieving margin $\gamma$ on the given dataset, then since we made a mistake
  $$w_*^T w^{(t)} = w_*^T(w^{(t-1)} + y_t x_t) = w_* w^{(t-1)} + y_t(w_*^T x_t)$$
  using a definition of margin (coming up later) that
  $y_t(w_*^T x_t) \geq \gamma$ for all $t$ in the training data, we get
  $$w_*^T w^{(t)} \geq w_*^T w^{(t-1)} + \gamma$$
  hence, every time you make a mistake, perceptron algorithm makes a progress towards the optimal separator by $\gamma$
- this implies that $w_*^T w^{(t)} \geq \gamma M_t$
- if we make a mistake at $t$, then (using $y_t((w^{(t-1)})^T x_t) \leq 0$)
  $\|w^{(t)}\|_2^2 = \|w^{(t-1)}\|_2^2 + 2y_t((w^{(t-1)})^T x_t)) + \|x\|_2^2 \leq \|w^{(t-1)}\|_2^2 + 0 + \|x\|_2^2 \leq \|w^{(t-1)}\|_2^2 + R^2$
- Since $\|w^{(t)}\|_2^2$ grows by at most $R^2$ at every mistake, $\|w^{(t)}\|_2^2 \leq R^2 M_t$
- together, they imply
  $$\gamma M_t \;\leq\; w_*^T w^{(t)} \;\leq\; \|w_*\|_2 \|w^{(t)}\|_2 \;\leq\; R\sqrt{M_t}$$
- this completes the proof, after solving for $M_t$

- (good) for linearly separable datasets,
    - even if infinite examples are shown, the resulting number of mistakes is fixed
- (bad) however, real world is not linearly separable
    - it is not clear what the algorithm does
- (bad) even if the data is linearly separable, margin can be very small
- so perceptron algorithm is never used in practice

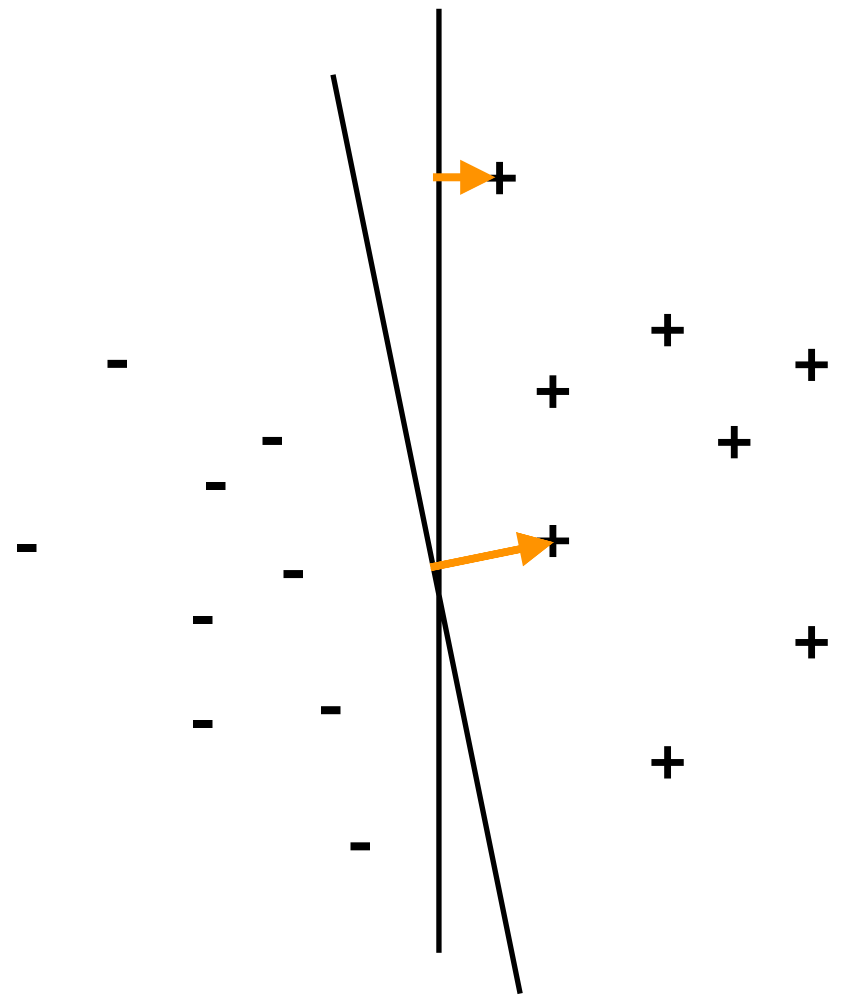




## Support Vector Machines (SVM)

## How do we choose the best linear classifier?

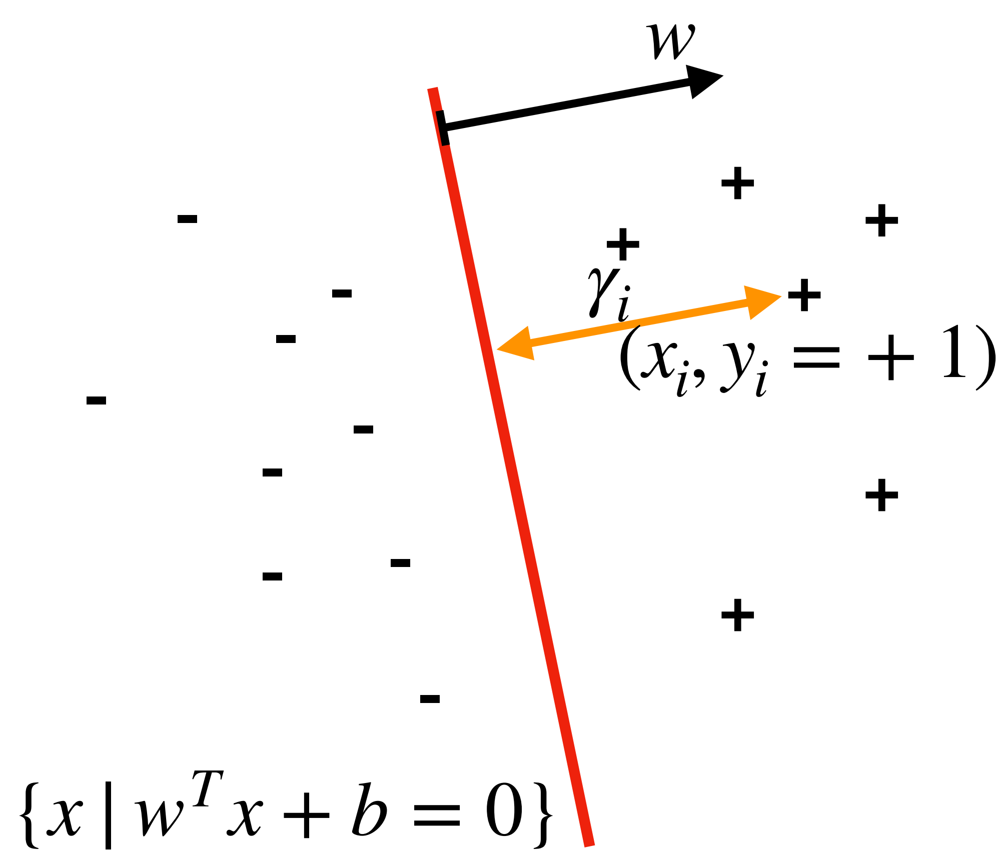
- for linearly separable datasets, **maximum margin** classifier is a natural choice
- large margin implies that the decision boundary can change without losing accuracy, so the learned model is more robust against new data points
- informally, **margin** of a set of examples to a decision boundary is the distance to the closest point to the decision boundary



# Geometric margin

- given a set of training examples $\{(x_i, y_i)\}_{i=1}^n$
- and a linear classifier $(w, b) \in \mathbb{R}^d \times \mathbb{R}$
- such that the decision boundary is
  a separating hyperplane $\{x \mid \underbrace{b + w_1 x[1] + w_2 x[2] + \cdots + w_d x[d]}_{w^T x + b} = 0\}$,
  which is the set of points that are orthogonal to $w$ with a shift of $b$
- note that we used to write the offset as $w[0]$, but for the purpose of discussing the margin, it is easier to explicitly have a separate notation for the offset $b$
- we define **functional margin** of $w$ with respect to a training example $(x_i, y_i)$ as the distance from the point $(x_i, y_i)$ to the decision boundary, which is

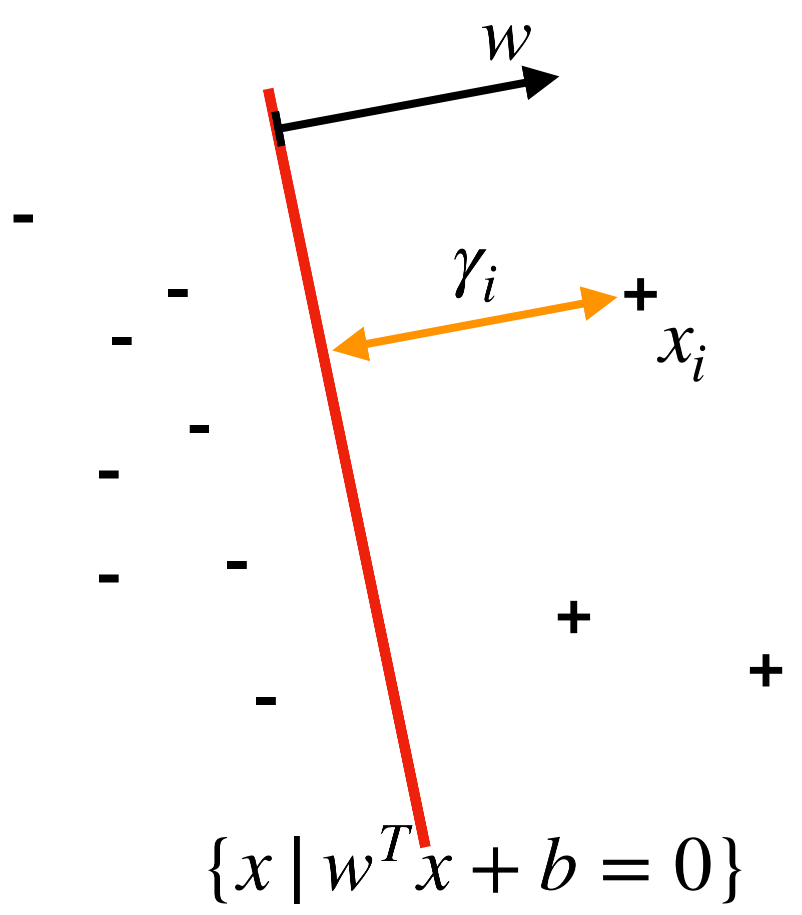
$$\gamma_i = y_i \frac{(w^T x_i + b)}{\|w\|_2}$$



# Geometric margin

- the distance $\gamma_i$ from a hyperplane $\{x \mid w^T x + b = 0\}$ to a point $x_i$ can be computed geometrically as follows
- We know that if you move from $x_i$ in the negative direction of $w$ by length $\gamma_i$, you arrive at the line, which can be written as

$$\left( x_i - \frac{w}{\|w\|_2}\gamma_i \right) \text{ is in } \{x \mid w^T x + b = 0\}$$



- so we can plug the point in the formula:

$$w^T\left( x_i - \frac{w}{\|w\|_2}\gamma_i \right) + b \;=\; 0$$

which is

$$w^T x_i - \frac{\|w\|_2^2}{\|w\|_2}\gamma_i + b \;=\; 0$$

and hence

$$\gamma_i \;=\; \frac{w^T x_i + b}{\|w\|_2},$$

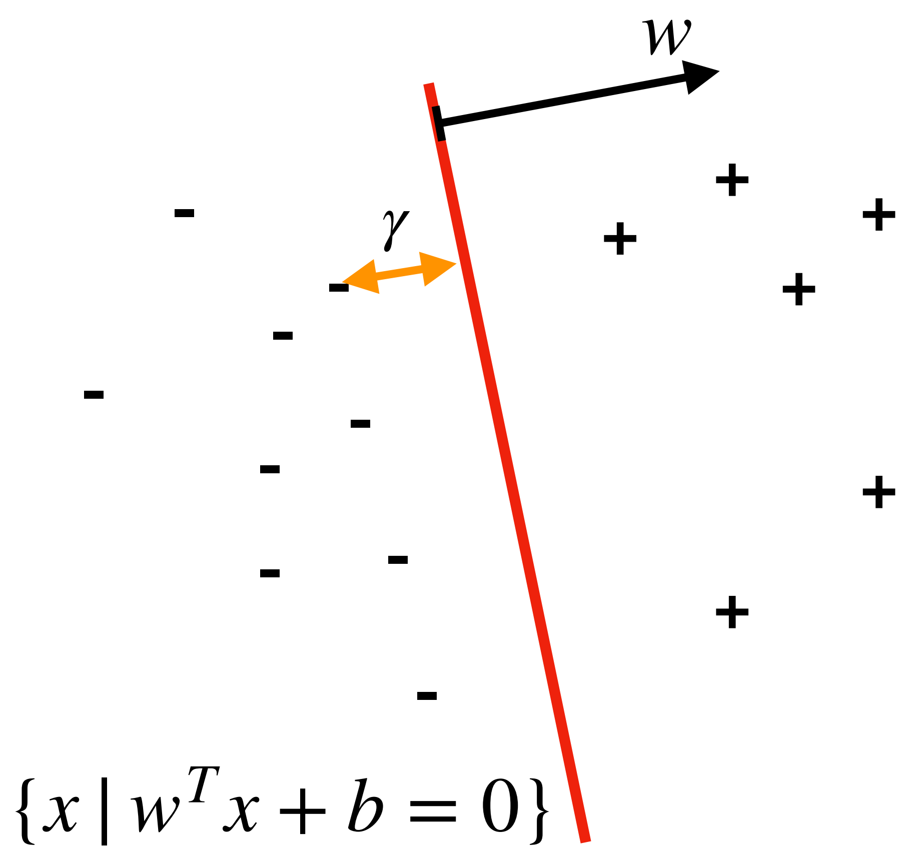
and we multiply it by $y_i$ s=o that for negative samples we use $-w$ instead of $w$

# Geometric margin

- the margin with respect to a set is defined as

$$\gamma = \min_{i=1}^{n} \gamma_i$$

- among all linear classifiers,
  we would like to find one that has the maximum margin



# Maximum margin classifier

- we propose the following optimization problem:

$$\text{maximize}_{w\in\mathbb{R}^d, b\in\mathbb{R}, \gamma\in\mathbb{R}} \quad \gamma$$

**(maximize the margin)**

$$\text{subject to} \quad \frac{y_i(w^T x_i + b)}{\|w\|_2} \geq \gamma \quad \text{for all } i \in \{1,\ldots,n\}$$

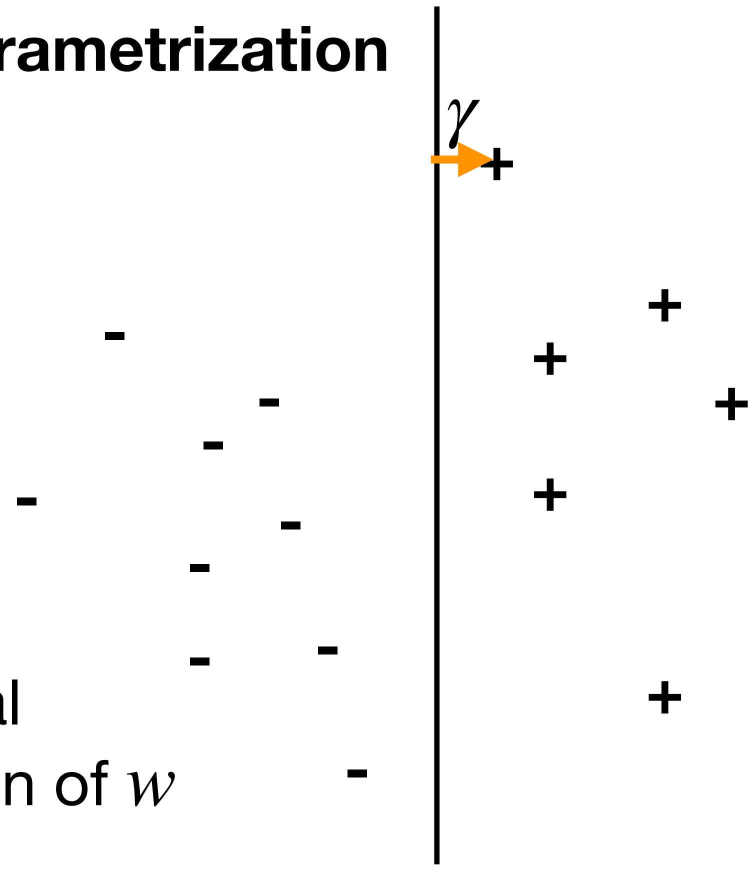
**(s.t. $\gamma$ is a lower bound on the margin)**

- if we fix $(w, b)$, above optimization gives the margin
- together with $(w, b)$, this finds the classifier with the maximum margin
- note that this problem is **scale invariant** in $(w, b)$, i.e. changing a $(w, b)$ to $(2w, 2b)$ does not change either the feasibility or the objective value
- the above optimization looks difficult, so we transform it using **reparametrization**

$$\text{maximize}_{w\in\mathbb{R}^d, b\in\mathbb{R}, \gamma\in\mathbb{R}} \quad \gamma$$

$$\text{subject to} \quad \frac{y_i(w^T x_i + b)}{\|w\|_2} \geq \gamma \quad \text{for all } i \in \{1,\ldots,n\}$$

$$\|w\|_2 = \frac{1}{\gamma}$$



- the optimal solution does not change, as the solutions to the original problem did not depend on $\|w\|_2$, and only depends on the direction of $w$

- $$\text{maximize}_{w\in\mathbb{R}^d, b\in\mathbb{R}, \gamma\in\mathbb{R}} \quad \gamma$$

  $$\text{subject to} \quad \frac{y_i(w^T x_i + b)}{\|w\|_2} \geq \gamma \ \text{ for all } i \in \{1,\ldots,n\}$$

  $$\|w\|_2 = \frac{1}{\gamma}$$

- the above optimization still looks difficult, but can be transformed into

  $$\text{maximize}_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \frac{1}{\|w\|_2}$$

  $$\text{subject to} \quad \frac{y_i(w^T x_i + b)}{\|w\|_2} \geq \frac{1}{\|w\|_2} \quad \text{for all } i \in \{1,\ldots,n\}$$

  **(maximize the margin)**

  **(now $\frac{1}{\|w\|_2}$ is a lower bound on the margin)**

  which simplifies to

  $$\text{minimize}_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \|w\|_2^2$$

  $$\text{subject to} \quad y_i(w^T x_i + b) \geq 1 \ \text{ for all } i \in \{1,\ldots,n\}$$

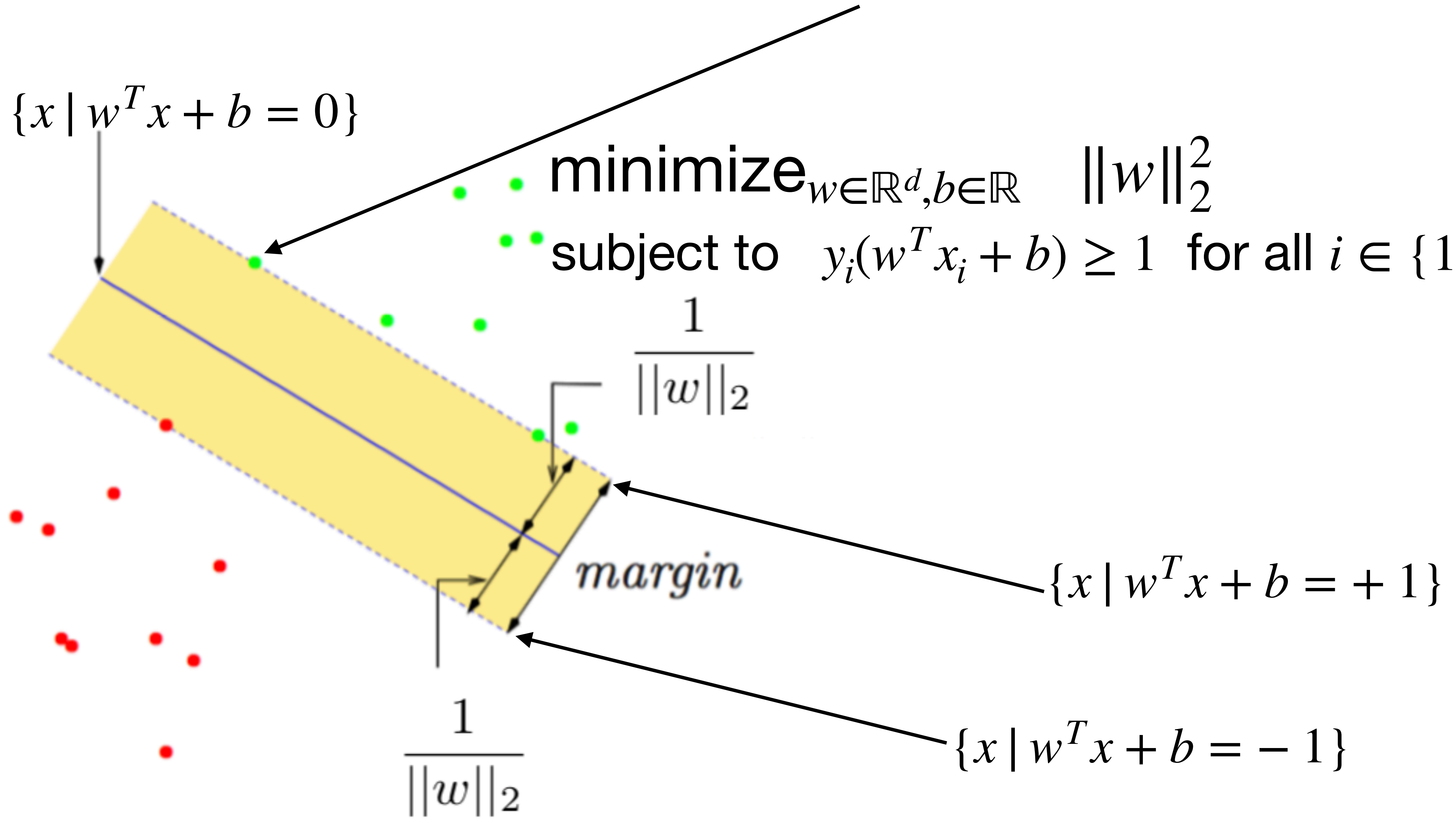
- this is a **quadratic program with linear constraints**, which can be easily solved
- once the optimal solution is found, the margin of that classifier $(w, b)$ is $\frac{1}{\|w\|_2}$

# What if the data is not separable?

- we cheated a little in the sense that the reparametrization of $\|w\|_2 = \frac{1}{\gamma}$ is possible only if the the margins are positive, i.e. the data is linearly separable with a positive margin
- otherwise, there is no feasible solution
- the examples at the margin are called **support vectors**

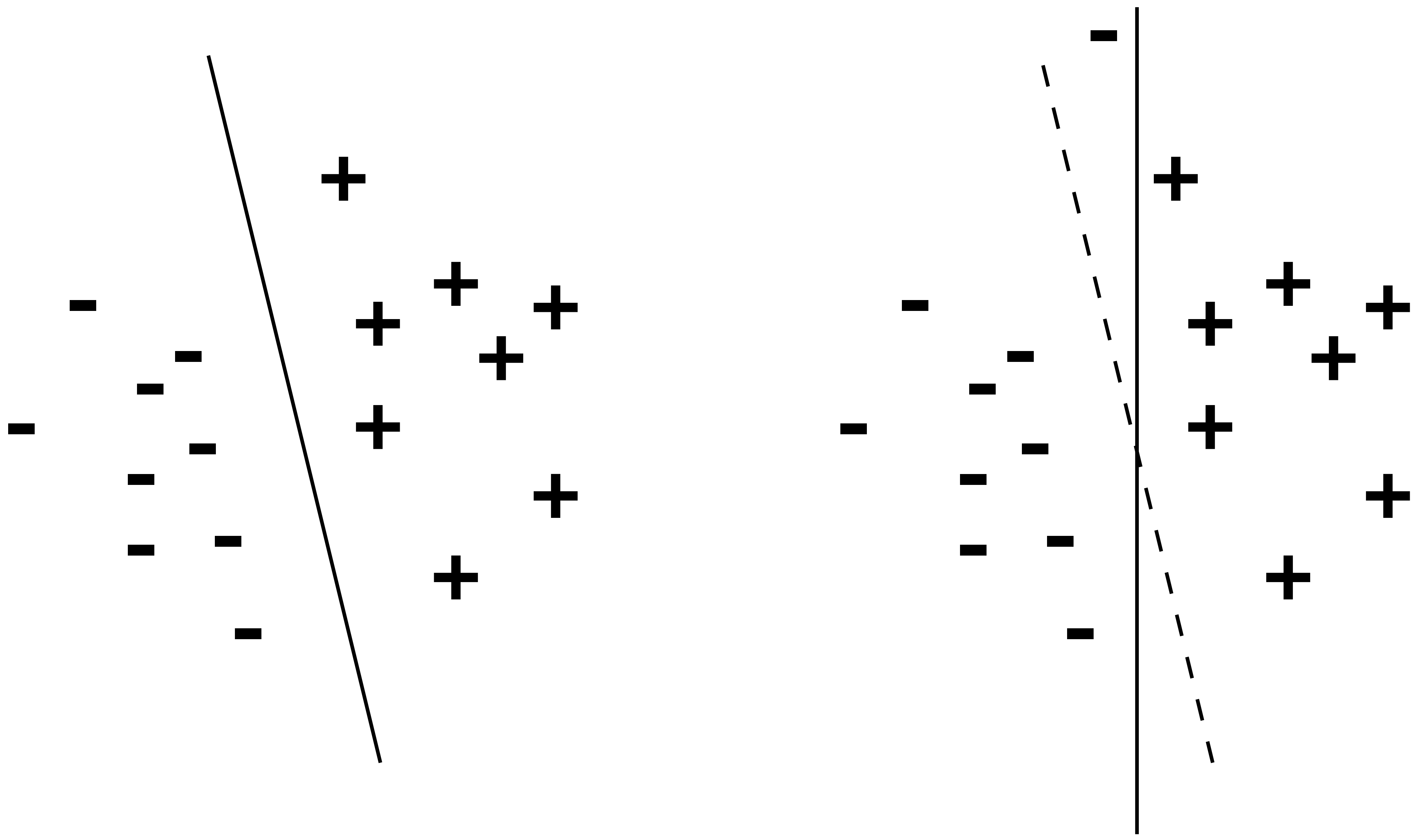
$$\text{minimize}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \quad \|w\|_2^2$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq 1 \ \text{ for all } i \in \{1, \ldots, n\}$$


$\{x \mid w^T x + b = 0\}$
$\frac{1}{||w||_2}$
margin
$\{x \mid w^T x + b = +1\}$
$\frac{1}{||w||_2}$
$\{x \mid w^T x + b = -1\}$

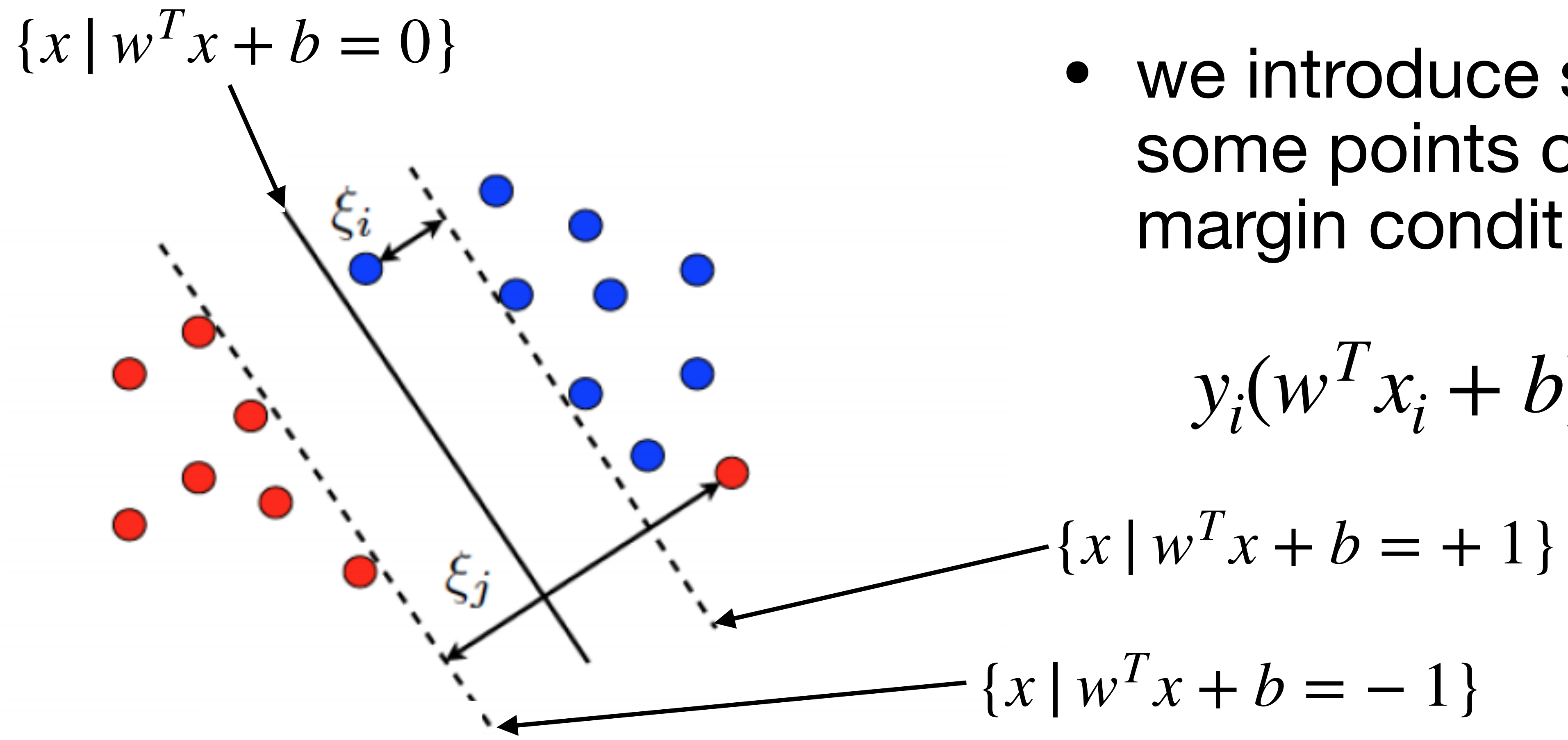
# Two issues

- max-margin formulation we proposed is sensitive to outliers



- it does not generalize to non-separable datasets

# What if the data is not separable?



$\{x \mid w^T x + b = 0\}$
$\xi_i$
$\xi_j$
$\{x \mid w^T x + b = +1\}$
$\{x \mid w^T x + b = -1\}$

- we introduce slack so that some points can violate the margin condition

$$y_i(w^T x_i + b) \;\geq\; 1 - \xi_i$$

- this gives a new optimization problem with some positive constant $c \in \mathbb{R}$

$$\text{minimize}_{w\in\mathbb{R}^d, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \|w\|_2^2 + c\sum_{i=1}^{n} \xi_i$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq 1 - \xi_i \quad \text{for all } i \in \{1,\ldots,n\}$$

$$\xi_i \;\geq 0 \quad \text{for all } i \in \{1,\ldots,n\}$$

the (re-scaled) margin (for each sample) is allowed to be less than one,
but you pay $c\xi_i$ in the cost, and $c$ balances the two goals:
maximizing the margin for most examples vs. having small number of violations

## Support Vector Machine

- for the optimization problem

$$\text{minimize}_{w\in\mathbb{R}^d, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \quad \|w\|_2^2 + c\sum_{i=1}^{n}\xi_i$$

$$\text{subject to} \quad y_i(w^Tx_i+b) \geq 1-\xi_i \quad \text{for all } i\in\{1,\ldots,n\}$$

$$\xi_i \geq 0 \quad \text{for all } i\in\{1,\ldots,n\}$$

notice that at optimal solution, $\xi_i$'s satisfy

- $\xi_i = 0$ if margin is big enough $y_i(w^Tx_i+b)\geq 1$, or
- $\xi_i = 1-y_i(w^Tx_i+b)$, if the example is within the margin $y_i(w^Tx_i+b)<1$

- so one can write
  - $\xi_i = \max\{0, 1-y_i(w^Tx_i+b)\}$, which gives

$$\text{minimize}_{w\in\mathbb{R}^d, b\in\mathbb{R}} \quad \frac{1}{c}\|w\|_2^2 + \sum_{i=1}^{n}\max\{0,1-y_i(w^Tx_i+b)\}$$

# Hinge loss with L2 regularizer

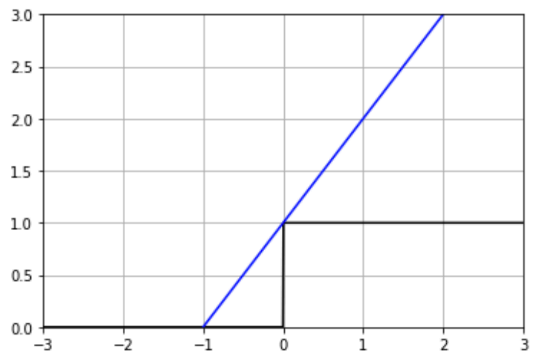
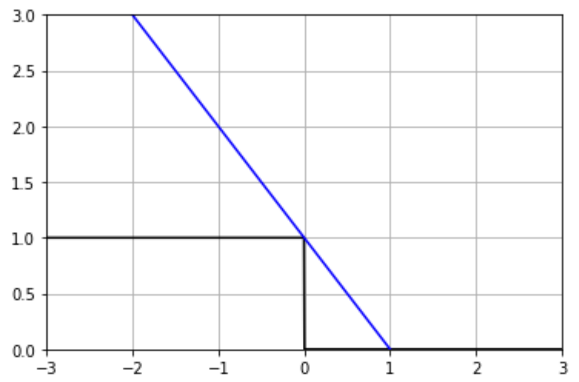
- Support vector machine:

$$\text{minimize}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \quad \frac{1}{c}\|w\|_2^2 + \sum_{i=1}^{n} \max\{0, 1 - y_i(w^T x_i + b)\}$$

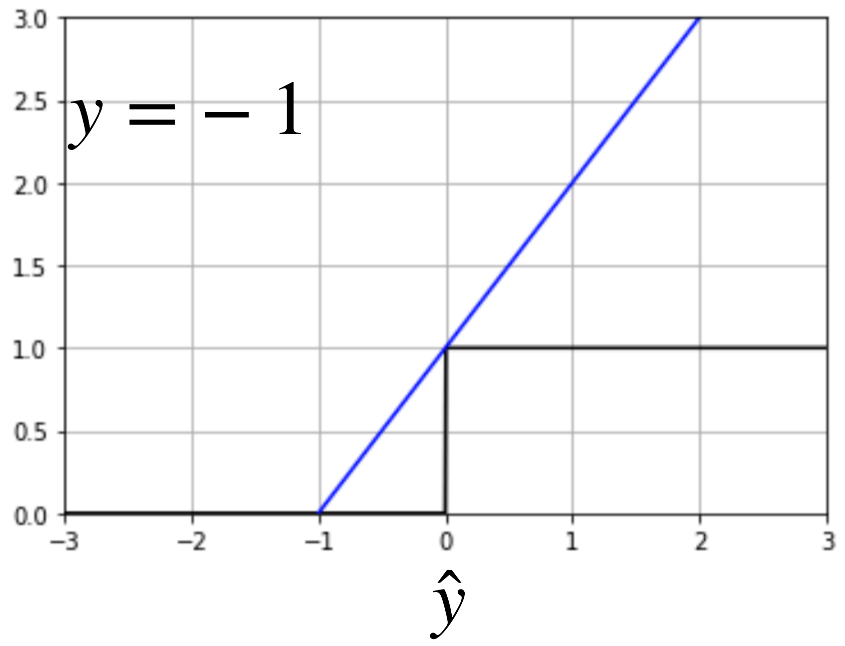
- instead, if we decided to train a linear classifier $(w, b)$ with empirical risk on **hinge loss** and **L2 regularizer**, we get the same

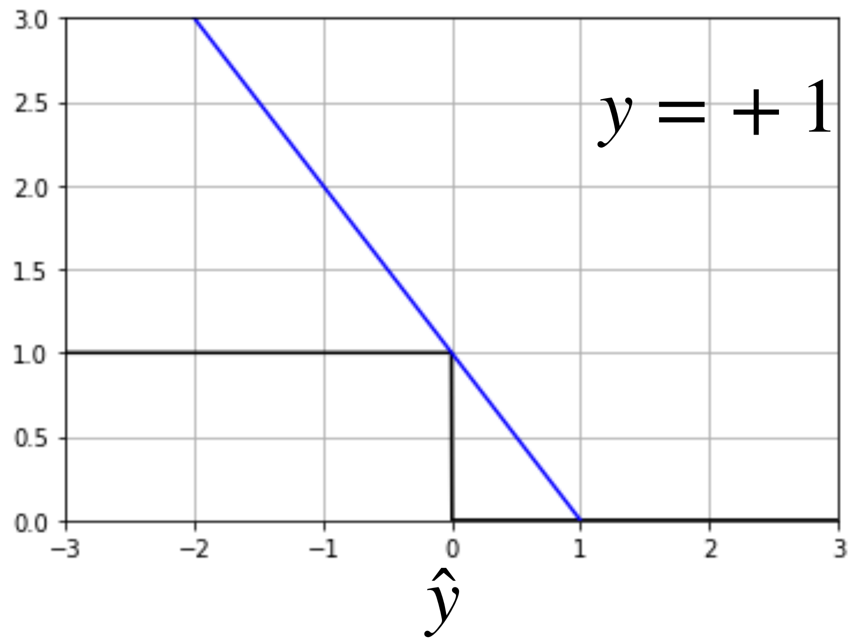
$$\text{minimize}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \quad \lambda\|w\|_2^2 + \sum_{i=1}^{n} \ell(w^T x_i + b, y_i)$$

where $\ell(\hat{y}, y) = \max\{0, 1 - y\,\hat{y}\}$

# Hinge loss





- for $\ell(\hat{y}, y) = \max\{0, 1 - y\,\hat{y}\}$, the sub-gradient is

$$\partial_{\hat{y}} \ell(\hat{y}, -1) \;=\; \begin{cases} 0 & \hat{y} < -1 \\ [0,1] & \hat{y} = -1 \\ 1 & \hat{y} > -1 \end{cases}$$

- for $\ell(w^T x + b, y)$, the sub-gradient is

$$\partial_w \ell(w^T x + b, -1) \;=\; \partial_{\hat{y}}(w^T x + b, -1)\, x$$

## Sub-gradient descent for SVM

- SVM is the solution of

$$\text{minimize}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \quad \frac{1}{c}\|w\|_2^2 + \sum_{i=1}^{n} \max\{0, 1 - y_i(w^T x_i + b)\}$$

- as it is non-differentiable, we solve it using sub-gradient descent
- which is exactly the same as gradient descent, except when we are at a non-differentiable point, we take one of the sub-gradients instead of the gradient (recall sub-gradient is a set)
- this means that we can take (a generic form derived from previous page)

$$\partial_w \ell(w^T x_i + b, y_i) \;=\; \mathbf{I}\{y_i(w^T x_i + b) \le 1\}(-y_i x_i)$$

and apply

$$w^{(t+1)} \;\leftarrow\; w^{(t)} - \eta \left( \sum_{i=1}^{n} \mathbf{I}\{y_i((w^{(t)})^T x_i + b^{(t)}) \le 1\}(-y_i x_i) \;+\; \frac{2}{c} w^{(t)} \right)$$

$$b^{(t+1)} \;\leftarrow\; b^{(t)} - \eta \sum_{i=1}^{n} \mathbf{I}\{y_i((w^{(t)})^T x_i + b^{(t)}) \le 1)\}(-y_i)$$