Linear classifiers
Lecture 3

David Sontag
New York University

Slides adapted from Luke Zettlemoyer, Vibhav Gogate, and Carlos Guestrin

Example: Spam

- Imagine 3 features (spam is "positive" class):
  1. free (number of occurrences of "free")
  2. money (occurrences of "money")
  3. BIAS (intercept, always has value 1)

$$w \cdot f(x)$$

$$\sum_i w_i \cdot f_i(x)$$

x	$f(x)$	w
"free money"	BIAS : 1 free : 1 money : 1 ...	BIAS : -3 free : 4 money : 2 ...

$$\begin{aligned} &(1)(-3) \quad + \\ &(1)(4) \quad + \\ &(1)(2) \quad + \\ &\ldots \\ &= 3 \end{aligned}$$

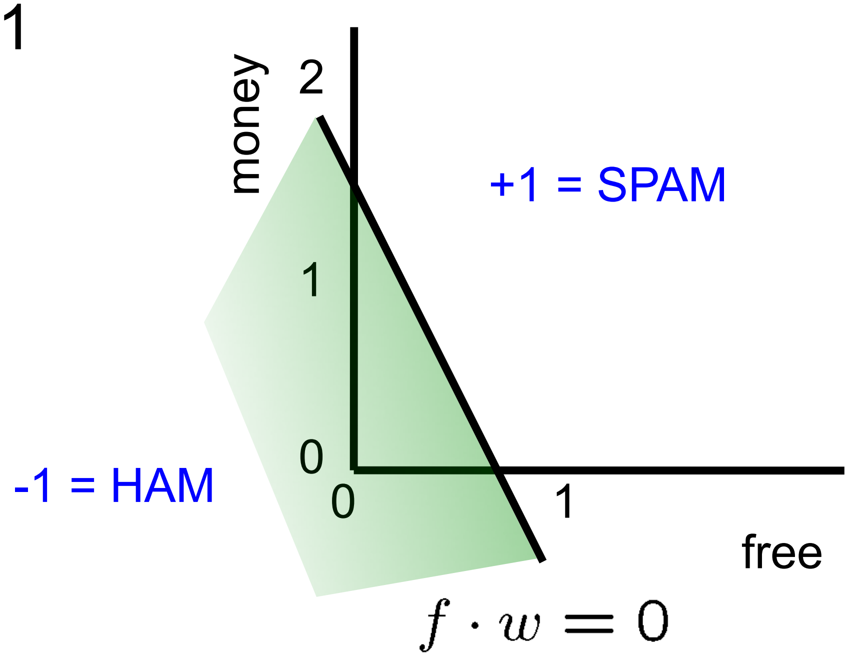
w.f(x) > 0 ➔ SPAM!!!

Binary Decision Rule

- In the space of feature vectors
  - Examples are points
  - Any weight vector is a hyperplane
  - One side corresponds to Y=+1
  - Other corresponds to Y=-1

$$w$$

```
BIAS  : -3
free  :  4
money :  2
...
```



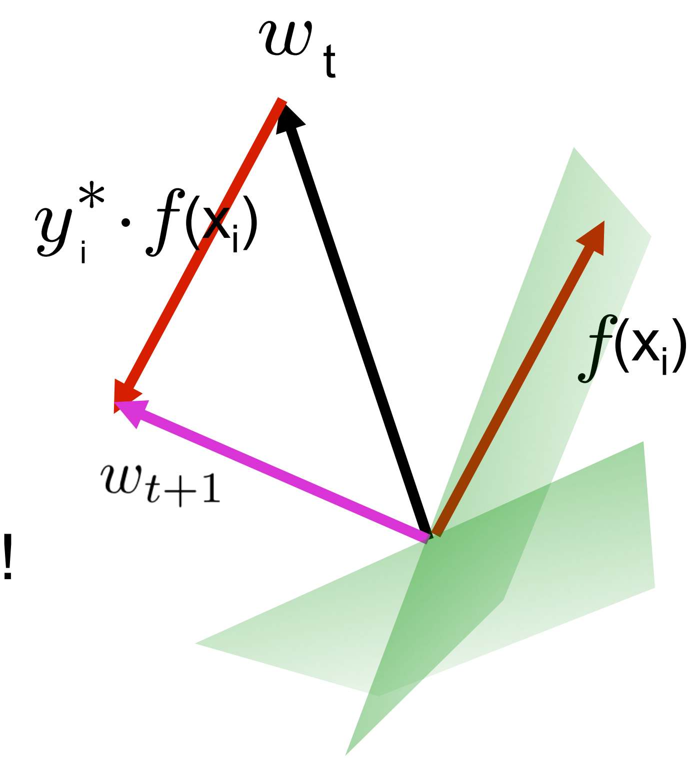
The perceptron algorithm

- Start with weight vector = $\vec{0}$
- For each training instance (x_i,y_i*):
  - Classify with current weights

$$y_i = \begin{cases} +1 & \text{if } \; w \cdot f(x_i) \geq 0 \\ -1 & \text{if } \; w \cdot f(x_i) < 0 \end{cases}$$

  - If correct (i.e., y=y_i*), no change!
  - If wrong: update

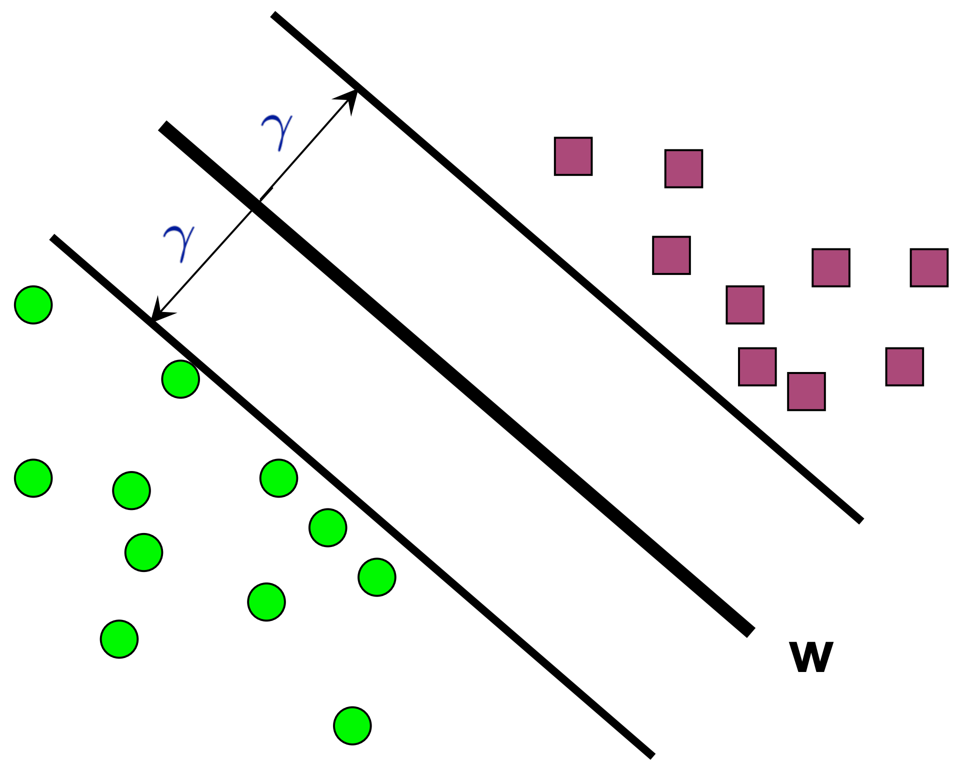
$$w = w + y_i^* f(x_i)$$



Def: Linearly separable data

$\exists \mathbf{w}$ such that $\forall t$ $\quad y_t(\mathbf{w} \cdot \mathbf{x}_t) \geq \gamma > 0$

Called the *margin*



Equivalently, for $y_t = +1$,

$$w \cdot x_t \geq \gamma$$

and for $y_t = -1$,

$$w \cdot x_t \leq -\gamma$$

Mistake Bound: Separable Case

- Assume the data set *D* is linearly separable with margin γ, i.e.,

$$\exists \mathbf{w}^*, |\mathbf{w}^*|_2 = 1, \ \forall t, y_t \mathbf{x}_t^\top \mathbf{w}^* \geq \gamma$$

- Assume $|\mathbf{x}_t|_2 \leq R, \forall t$
- <u>Theorem</u>: The maximum number of mistakes made by the perceptron algorithm is bounded by R^2/γ^2

[Rong Jin]

Proof by induction

Assume we make a mistake for $(\mathbf{x}_t, y_t)$

$$|\mathbf{w}_{t+1}|_2^2 = |\mathbf{w}_t + \; y_t \mathbf{x}_t| \leq |\mathbf{w}_t|_2^2 + \quad R^2$$

$$\mathbf{w}_{t+1}^\top \mathbf{w}^* = \mathbf{w}_t^\top \mathbf{w}^* + \; y_t \mathbf{x}_t^\top \mathbf{w}^* \geq \mathbf{w}_t^\top \mathbf{w}^* + \; \gamma$$

$$|\mathbf{w}_t|_2^2 \leq M_t \; \cdot R^2 \qquad\qquad \mathbf{w}_t^\top \mathbf{w}^* \geq M_t \cdot \gamma$$

$$M_t \leq \frac{R^2}{\gamma^2}$$

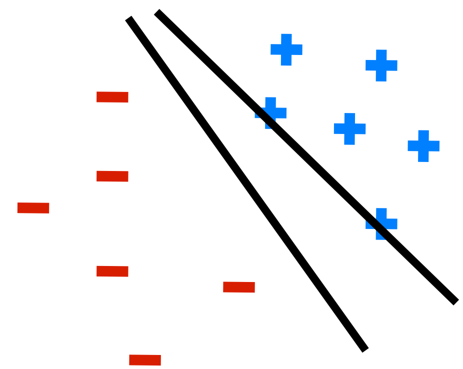
(full proof given on board)

[Rong Jin]

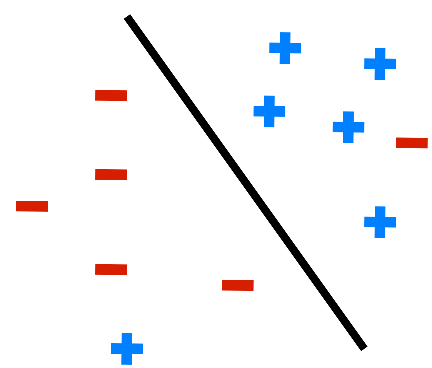
Properties of the perceptron algortihm

- Separability: some parameters get the training set perfectly correct
- Convergence: if the training is **linearly separable**, perceptron will eventually converge

Separable

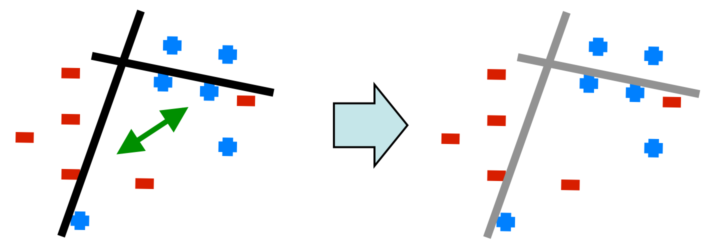


Non-Separable

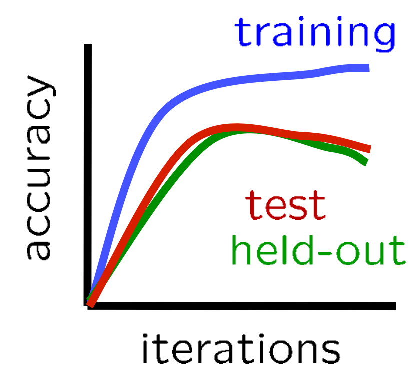


Problems with the perceptron algorithm

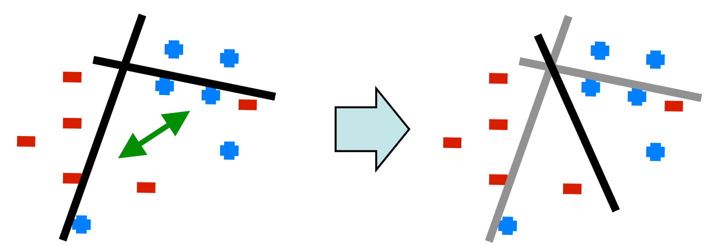
- **Noise**: if the data isn't linearly separable, no guarantees of convergence or accuracy



- Frequently the training data ***is*** linearly separable! Why?
  - When the number of features is much larger than the number of data points, there is lots of flexibility
  - As a result, Perceptron can significantly **overfit** the data

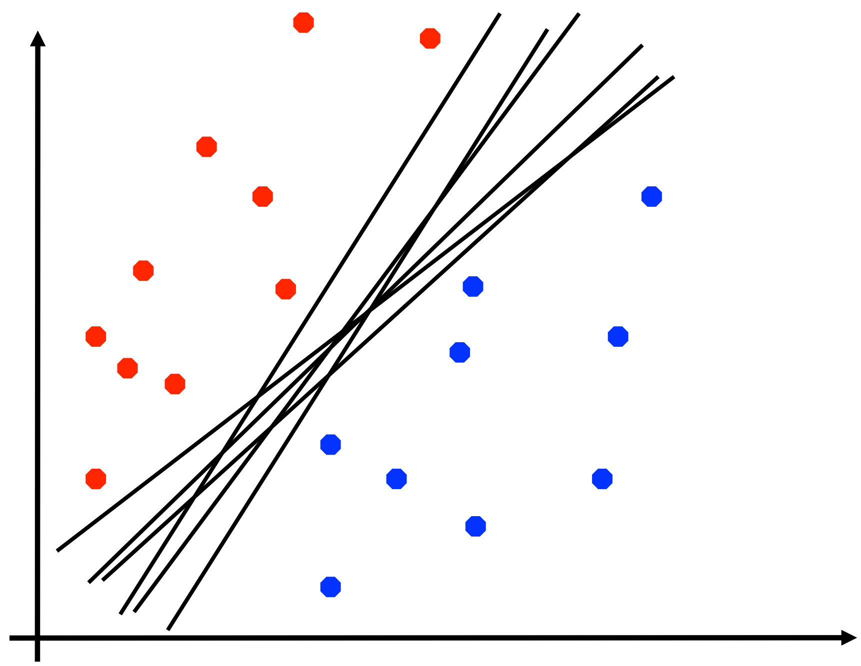


- **Averaged** perceptron is an algorithmic modification that helps with both issues
  - Averages the weight vectors across all iterations



Linear Separators

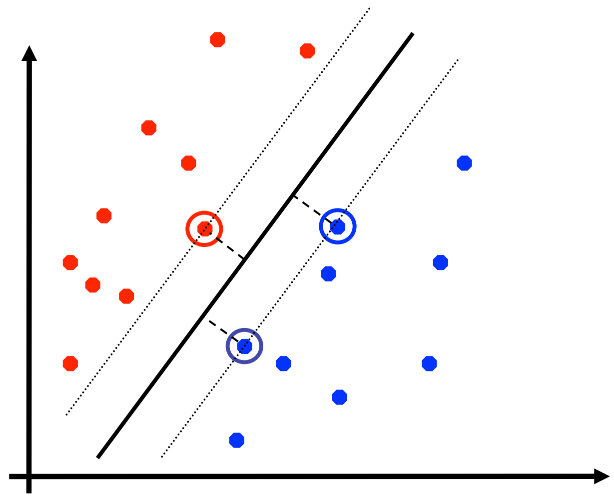
- Which of these linear separators is optimal?



Support Vector Machine (SVM)

- SVMs (Vapnik, 1990's) choose the linear separator with the **largest margin**

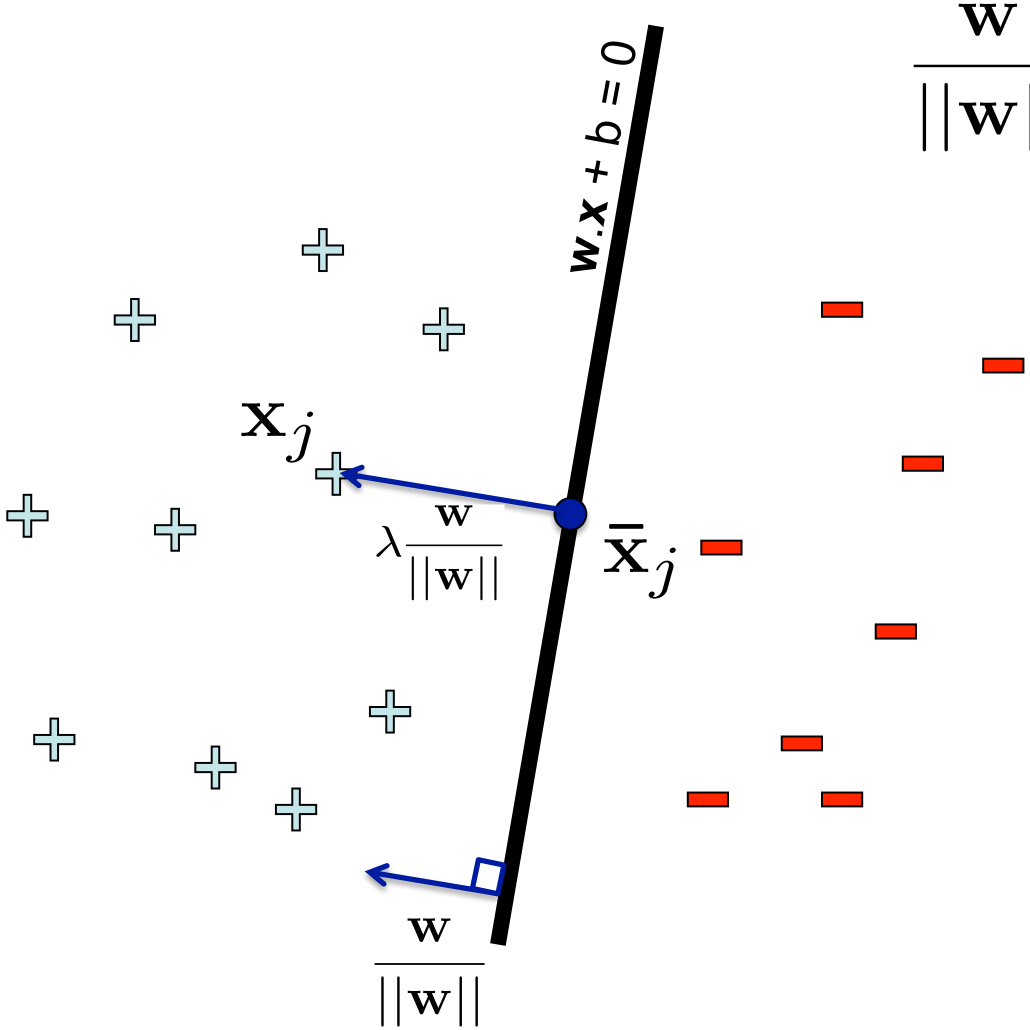
Robust to outliers!



V. Vapnik

- Good according to intuition, theory, practice
- SVM became famous when, using images as input, it gave accuracy comparable to neural-network with hand-designed features in a handwriting recognition task

Review: Normal to a plane



$\frac{\mathbf{w}}{||\mathbf{w}||}$ -- unit vector parallel to w

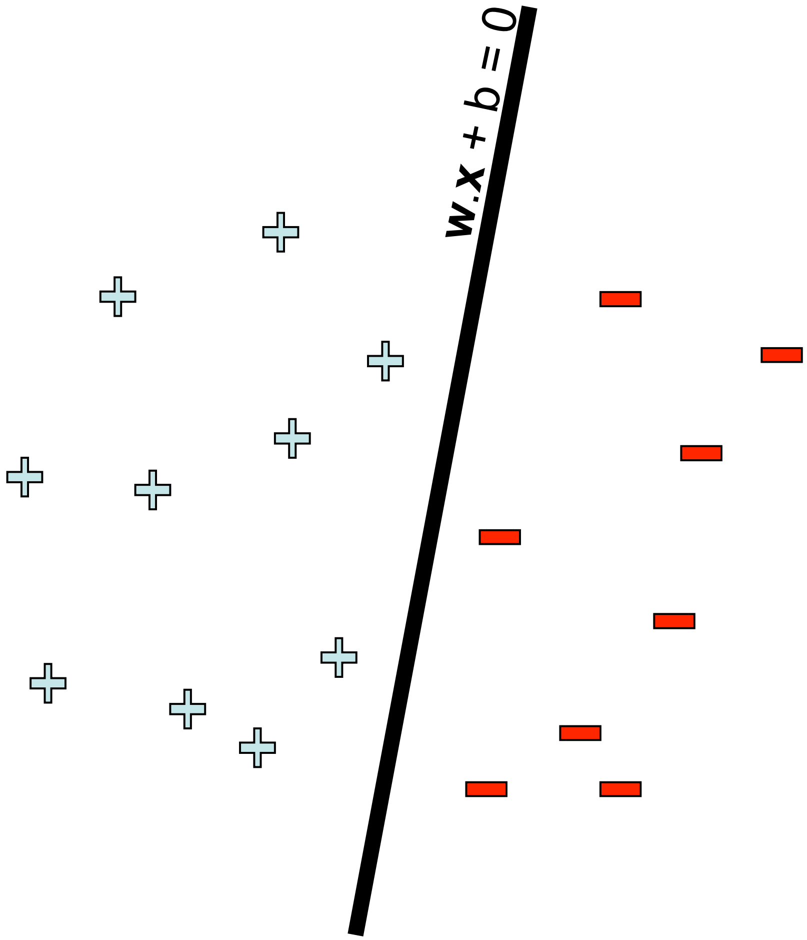
$\bar{\mathbf{x}}_j$ -- projection of x_j onto the plane

$$x_j - \bar{x}_j = \lambda \frac{w}{||w||}$$

λ is the length of the vector, i.e.

$$||x_j - \bar{x}_j|| = \frac{\lambda}{||w||}||w|| = \lambda$$

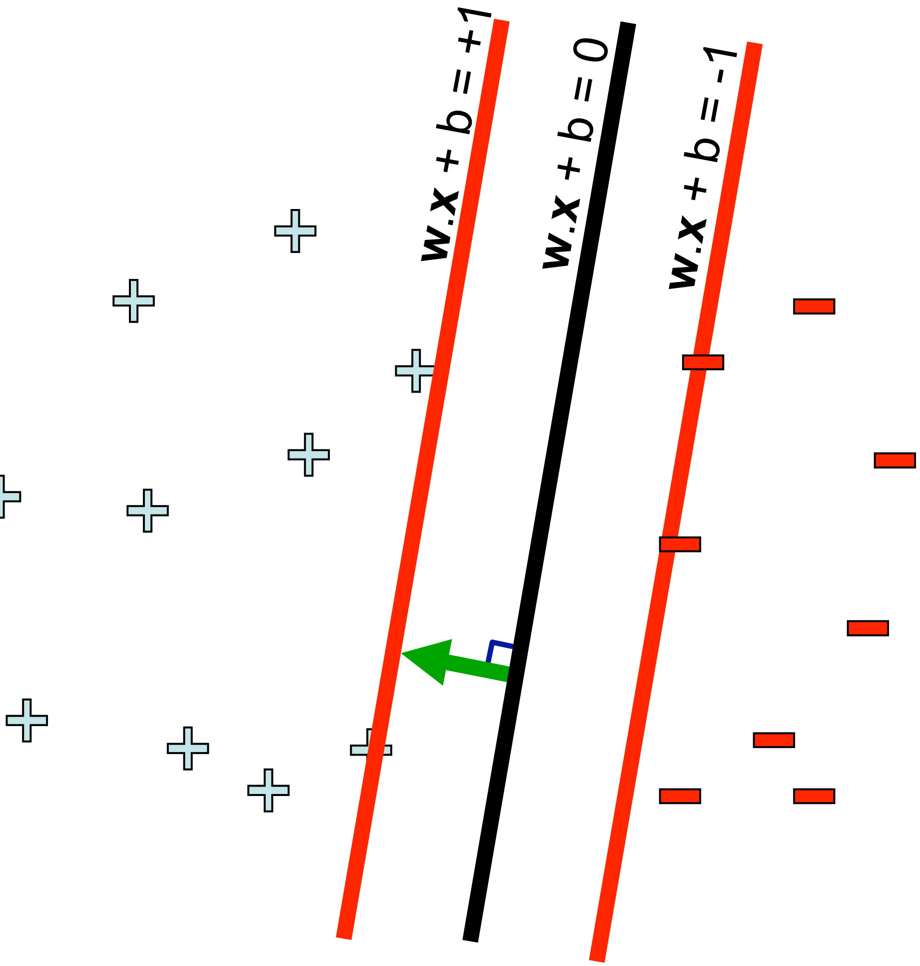
Scale invariance



Any other ways of writing the same dividing line?

- **w.x** + b = 0
- 2**w.x** + 2b = 0
- 1000**w.x** + 1000b = 0
- ….

Scale invariance



During learning, we set the scale by asking that, for all *t*,

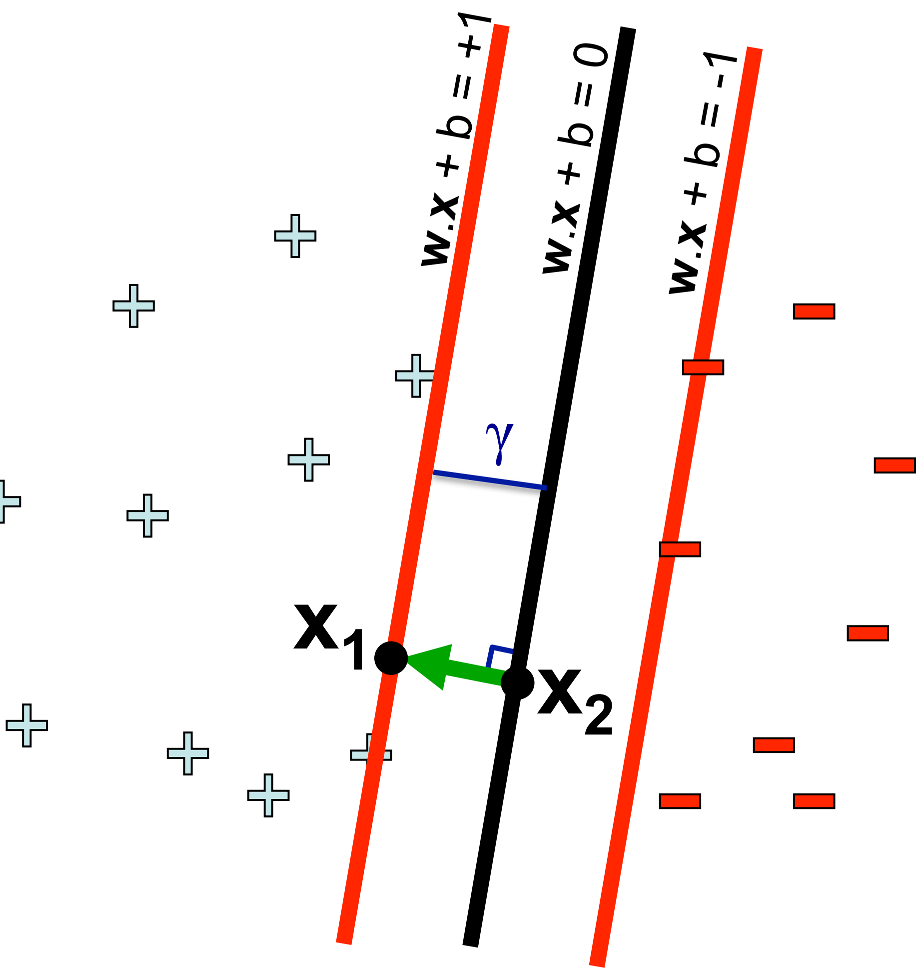
for y_t = +1, $w \cdot x_t + b \geq 1$

and for y_t = -1, $w \cdot x_t + b \leq -1$

That is, we want to satisfy all of the **linear** constraints

$$y_t \left(w \cdot x_t + b \right) \geq 1 \quad \forall t$$

What is γ as a function of **w**?



$$w \cdot x_1 + b = 1$$

-

$$w \cdot x_2 + b = 0$$

$$w \cdot (x_1 - x_2) = 1$$

Plug in

We also know that:

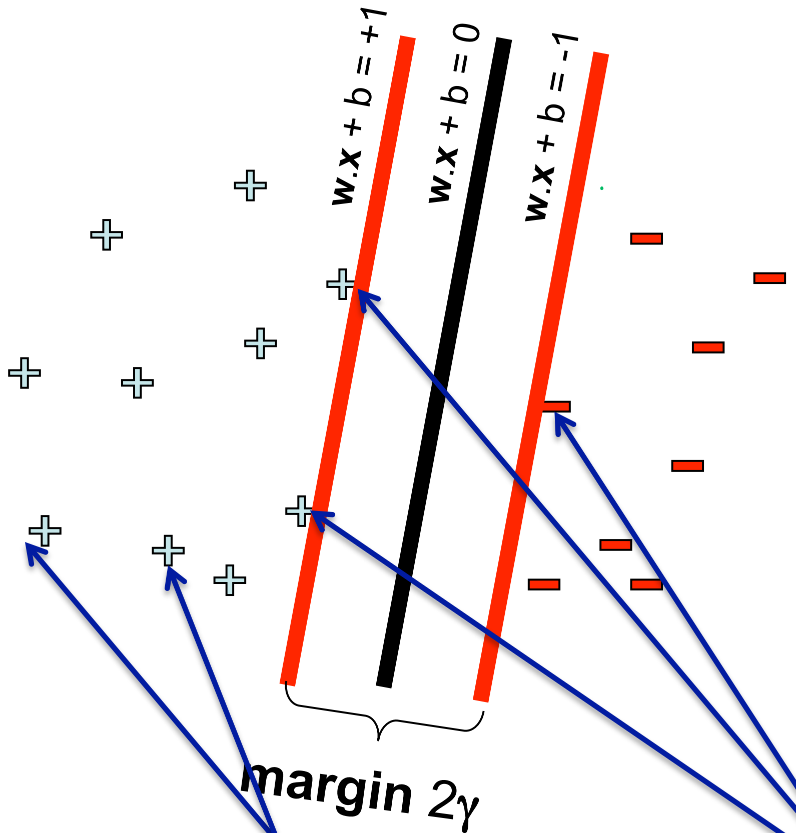
$$x_1 - x_2 = \gamma \frac{w}{||w||}$$

$$1 = w \cdot \left(\gamma \frac{w}{||w||} \right) = \frac{\gamma}{||w||} w \cdot w = \gamma ||w||$$

So, $\gamma = \dfrac{1}{||w||}$

Final result: can maximize margin by minimizing $||w||_2$!!!

Support vector machines (SVMs)



$$\text{minimize}_{\mathbf{w},b} \quad \mathbf{w}.\mathbf{w}$$
$$\left(\mathbf{w}.\mathbf{x}_j + b\right) y_j \geq 1, \;\; \forall j$$

- Example of a **convex optimization** problem
  - A quadratic program
  - Polynomial-time algorithms to solve!
- Hyperplane defined by **support vectors**
  - Could use them as a lower-dimension basis to write down line, although we haven't seen how yet
- More on these later

Non-support Vectors:
- everything else
- moving them will not change **w**

Support Vectors:
- data points on the canonical lines