

BBM406

Fundamentals of Machine Learning

Lecture 15:
Support Vector Machines

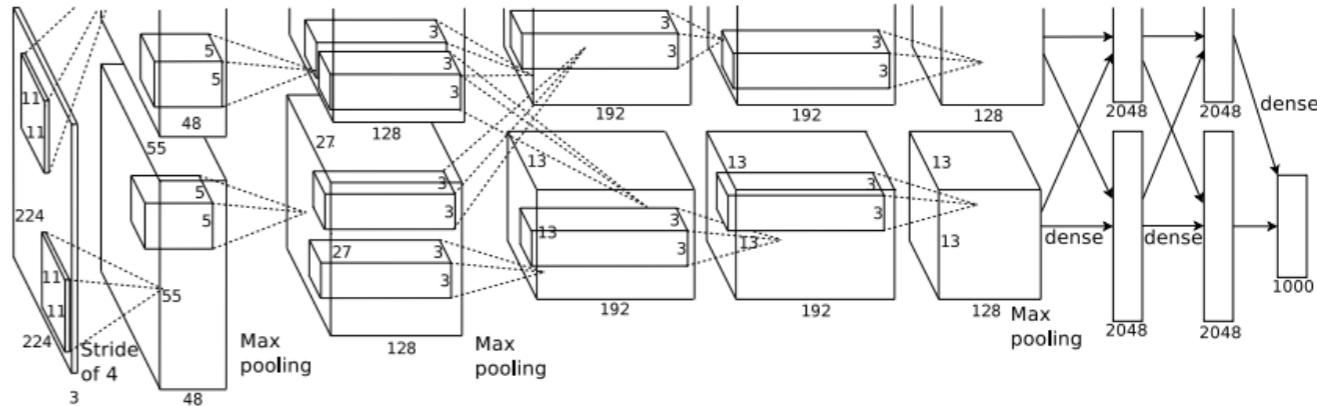


Announcement

- Midterm exam on ~~Nov 29~~ Dec 6, 2019 at 09.00 in rooms D3 & D4
- No class next Wednesday! Extra office hour.
- No class class on Friday! Make-up class on Dec 2 (Monday), 15:00-17:00
- No change in the due date of your Assg 3!

Last time...

AlexNet [Krizhevsky et al. 2012]



Full (simplified) AlexNet architecture:

[227x227x3] INPUT

[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0

[27x27x96] MAX POOL1: 3x3 filters at stride 2

[27x27x96] NORM1: Normalization layer

[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2

[13x13x256] MAX POOL2: 3x3 filters at stride 2

[13x13x256] NORM2: Normalization layer

[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1

[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1

[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1

[6x6x256] MAX POOL3: 3x3 filters at stride 2

[4096] FC6: 4096 neurons

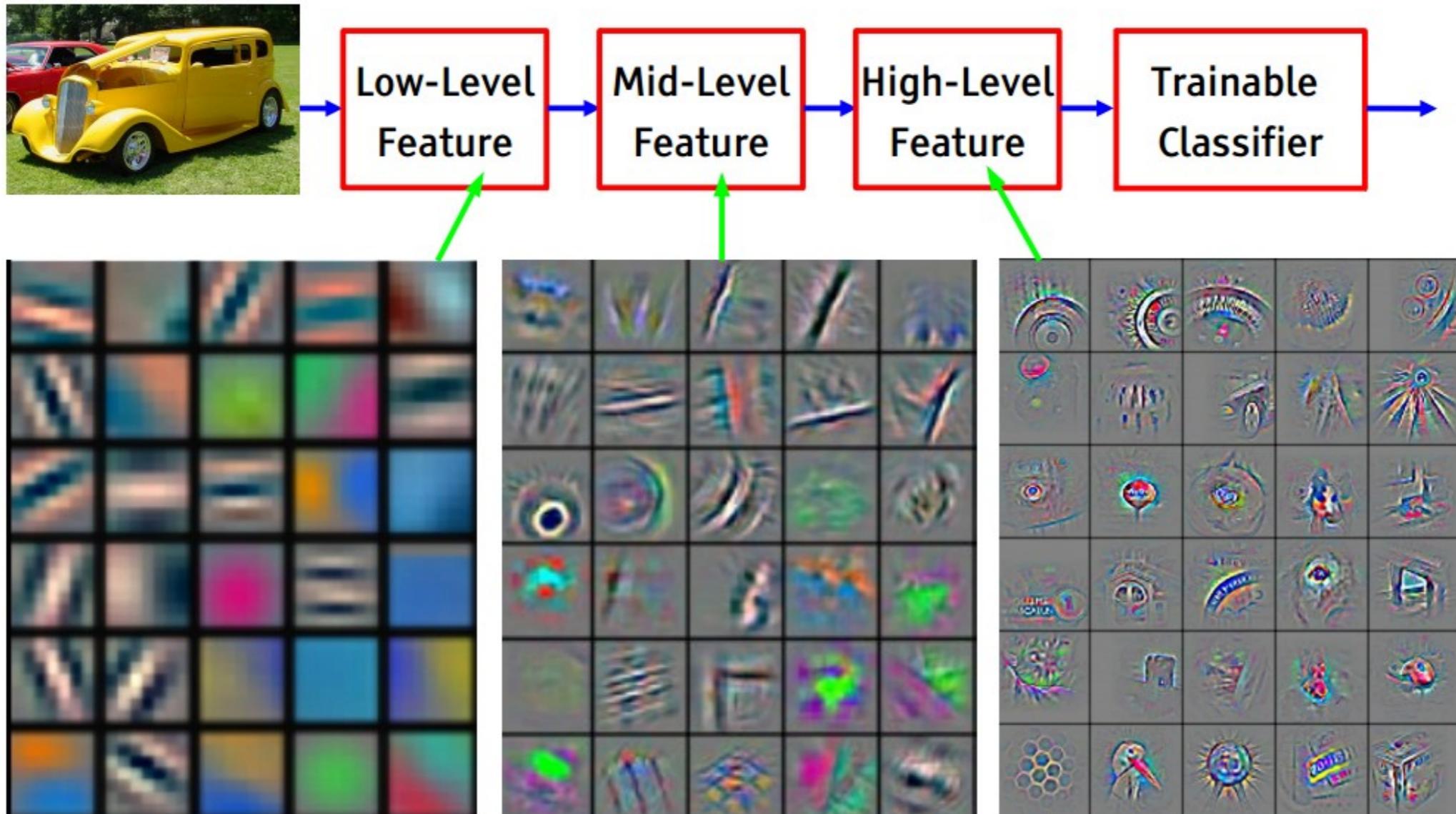
[4096] FC7: 4096 neurons

[1000] FC8: 1000 neurons (class scores)

Details/Retrospectives:

- first use of ReLU
- used Norm layers (not common anymore)
- heavy data augmentation
- dropout 0.5
- batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2, reduced by 10 manually when val accuracy plateaus
- L2 weight decay 5e-4
- 7 CNN ensemble: 18.2% -> 15.4%

Last time.. Understanding ConvNets

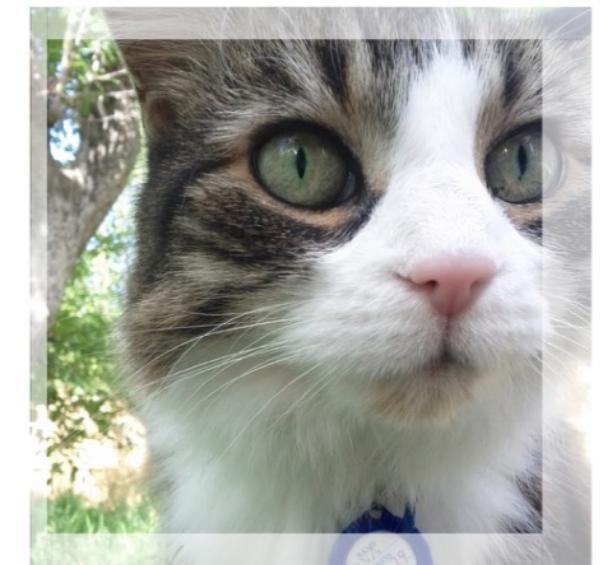


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

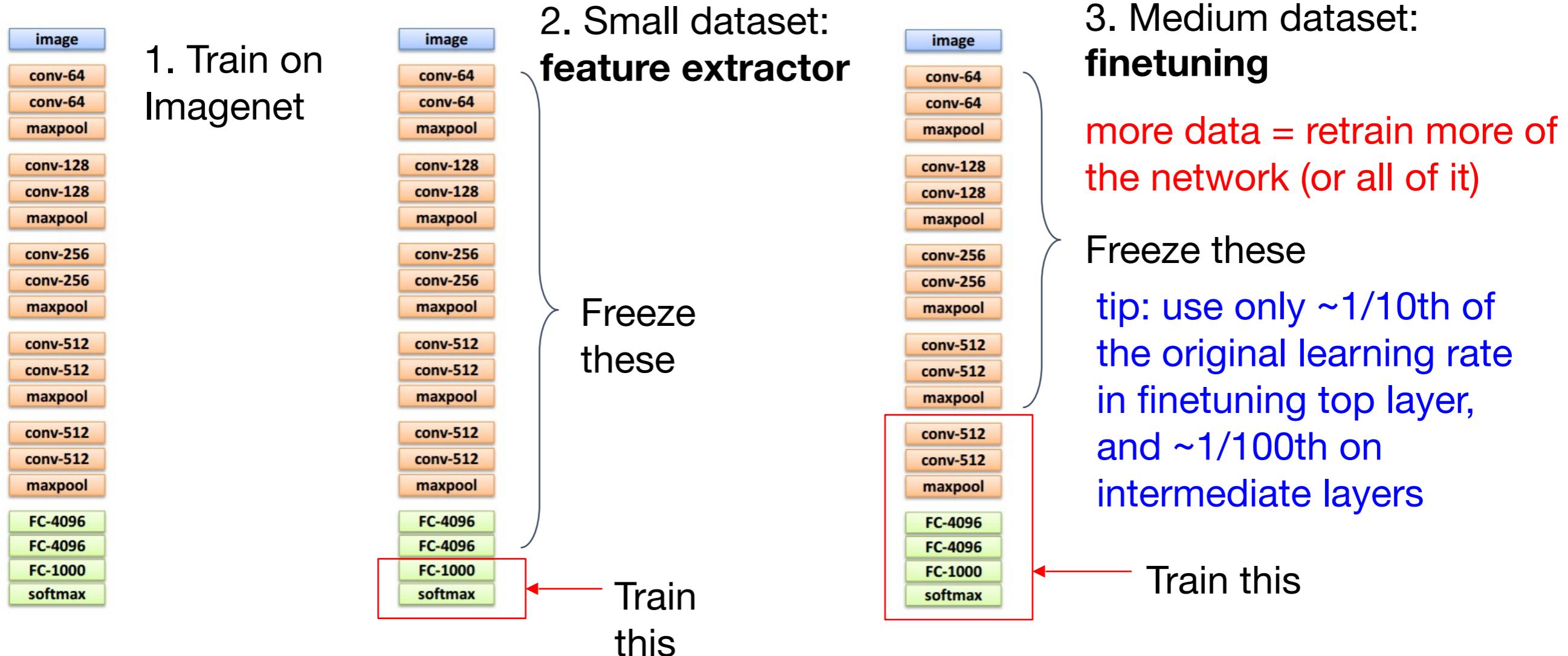
Last time... Data Augmentation

Random mix/combinations of:

- translation
- rotation
- stretching
- shearing,
- lens distortions, ...



Last time... Transfer Learning with Convolutional Networks



Today

- Support Vector Machines
 - Large Margin Separation
 - Optimization Problem
 - Support Vectors

Recap: Binary Classification Problem

- **Training data:** sample drawn i.i.d. from set $X \subseteq \mathbb{R}^N$ according to some distribution D ,
$$S = ((x_1, y_1), \dots, (x_m, y_m)) \in X \times \{-1, +1\}.$$
- **Problem:** find hypothesis $h: X \mapsto \{-1, +1\}$ in H (classifier) with small generalization error $R_D(h)$.
- **Linear classification:**
 - Hypotheses based on hyperplanes.
 - Linear separation in high-dimensional space.

Example: Spam

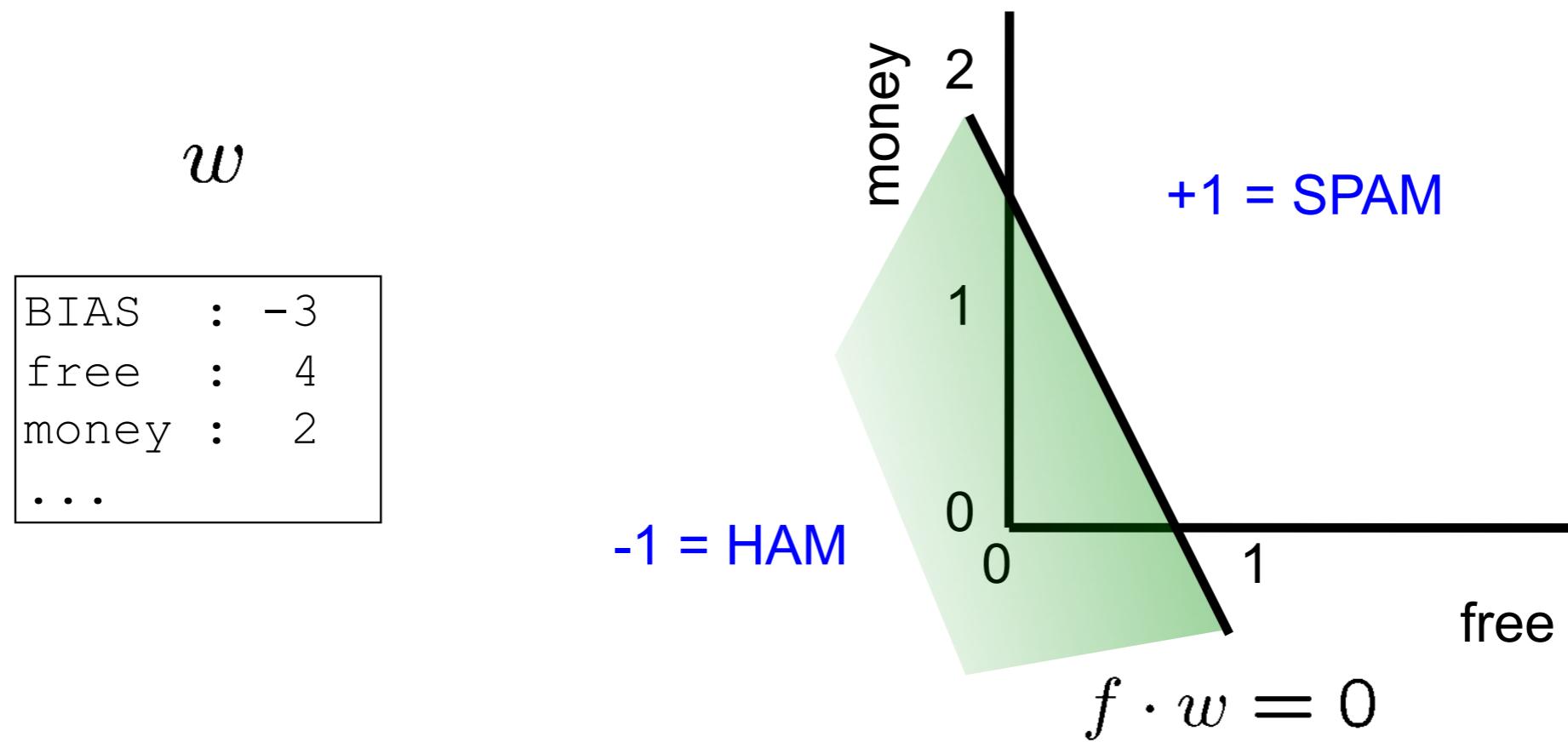
- Imagine 3 features (spam is “positive” class):
 - free (number of occurrences of “free”)
 - money (occurrences of “money”)
 - BIAS (intercept, always has value 1)

	x	$f(x)$	w									
“free money”	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>BIAS : 1</td></tr><tr><td>free : 1</td></tr><tr><td>money : 1</td></tr><tr><td>... : ...</td></tr></table>	BIAS : 1	free : 1	money : 1	... : ...	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>BIAS : -3</td></tr><tr><td>free : 4</td></tr><tr><td>money : 2</td></tr><tr><td>... : ...</td></tr></table>	BIAS : -3	free : 4	money : 2	... : ...	$w \cdot f(x) = \sum_i w_i \cdot f_i(x)$ $(1)(-3) + (1)(4) + (1)(2) + \dots = 3$	
BIAS : 1												
free : 1												
money : 1												
... : ...												
BIAS : -3												
free : 4												
money : 2												
... : ...												

$w \cdot f(x) > 0 \rightarrow \text{SPAM!!!}$

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 $Y = -1$



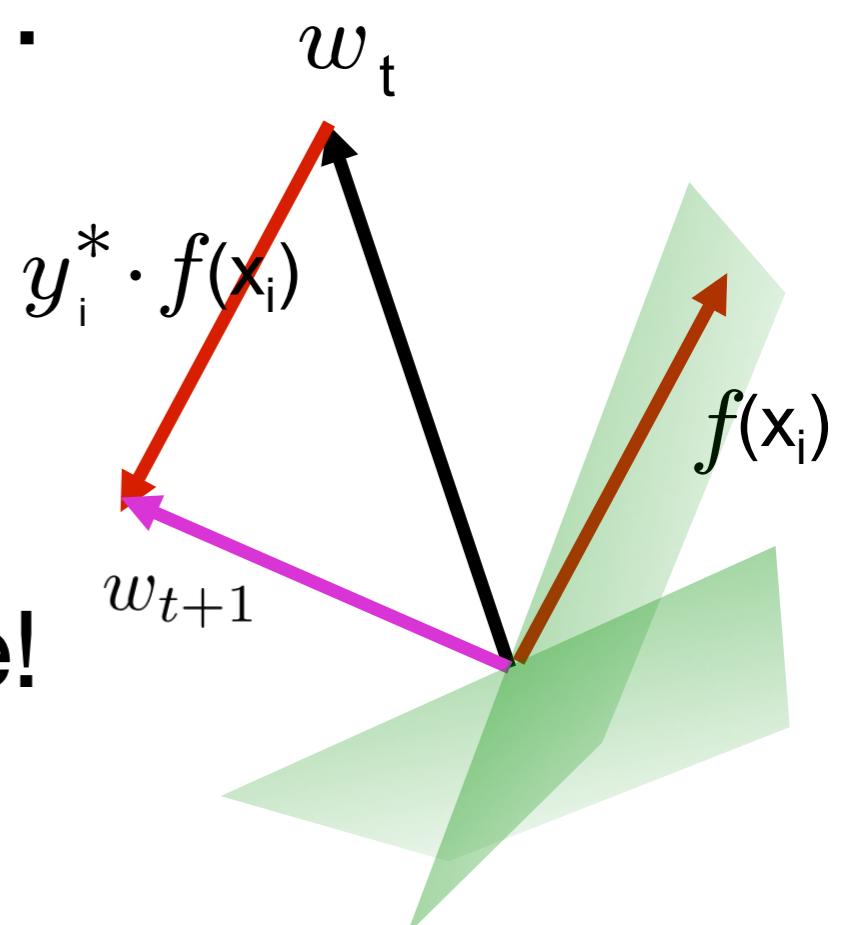
The perceptron algorithm

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

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

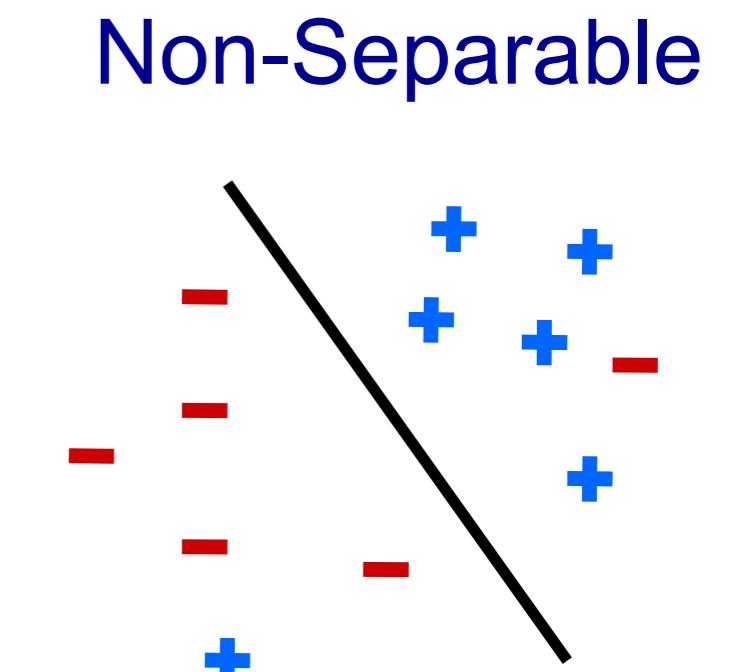
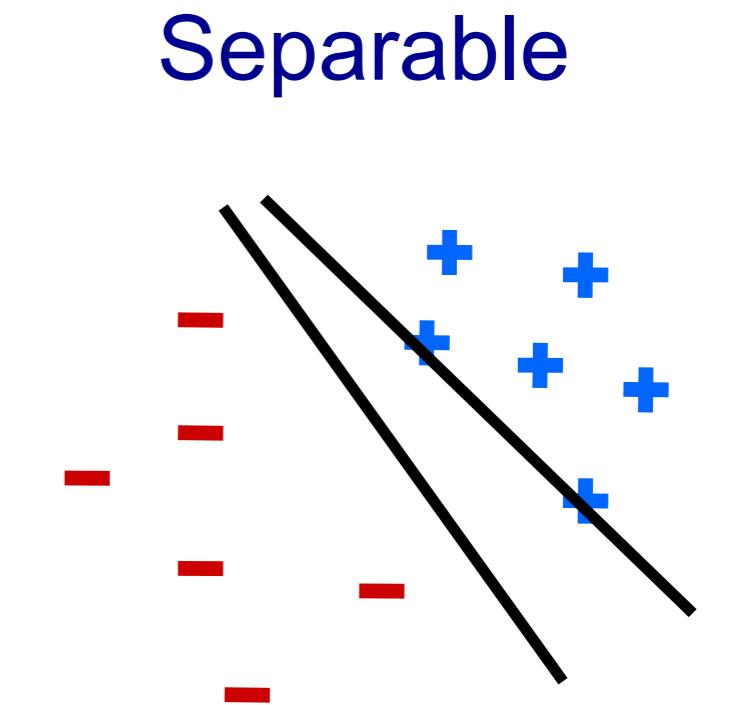
- If correct (i.e. $y = y_i^*$), no change!
- If wrong: update

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



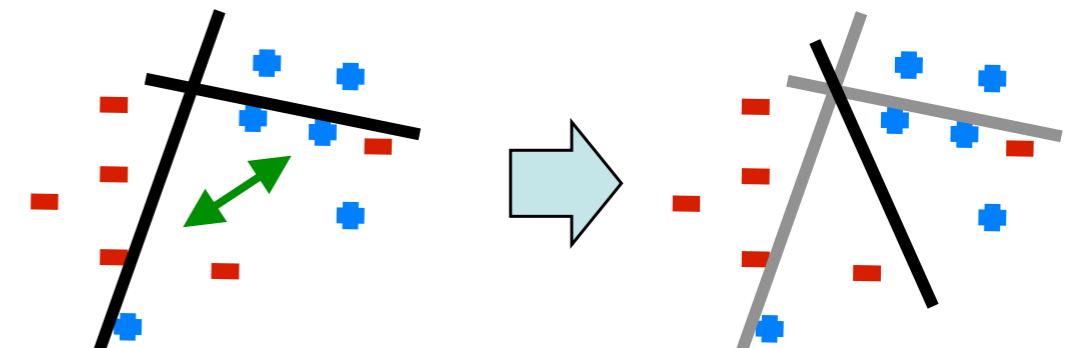
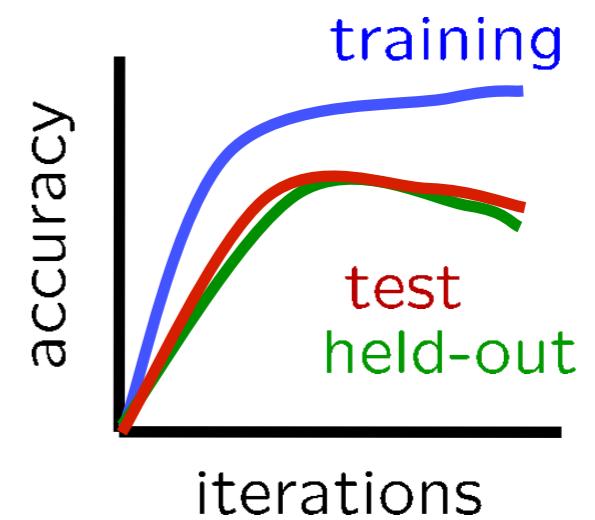
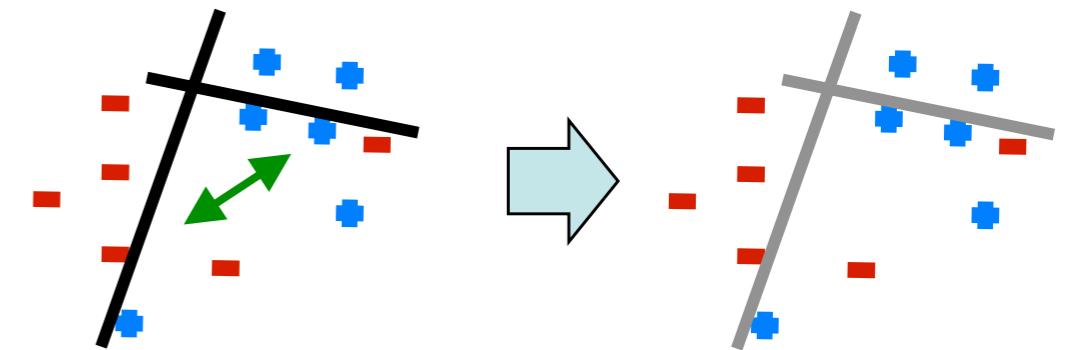
Properties of the perceptron algorithm

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



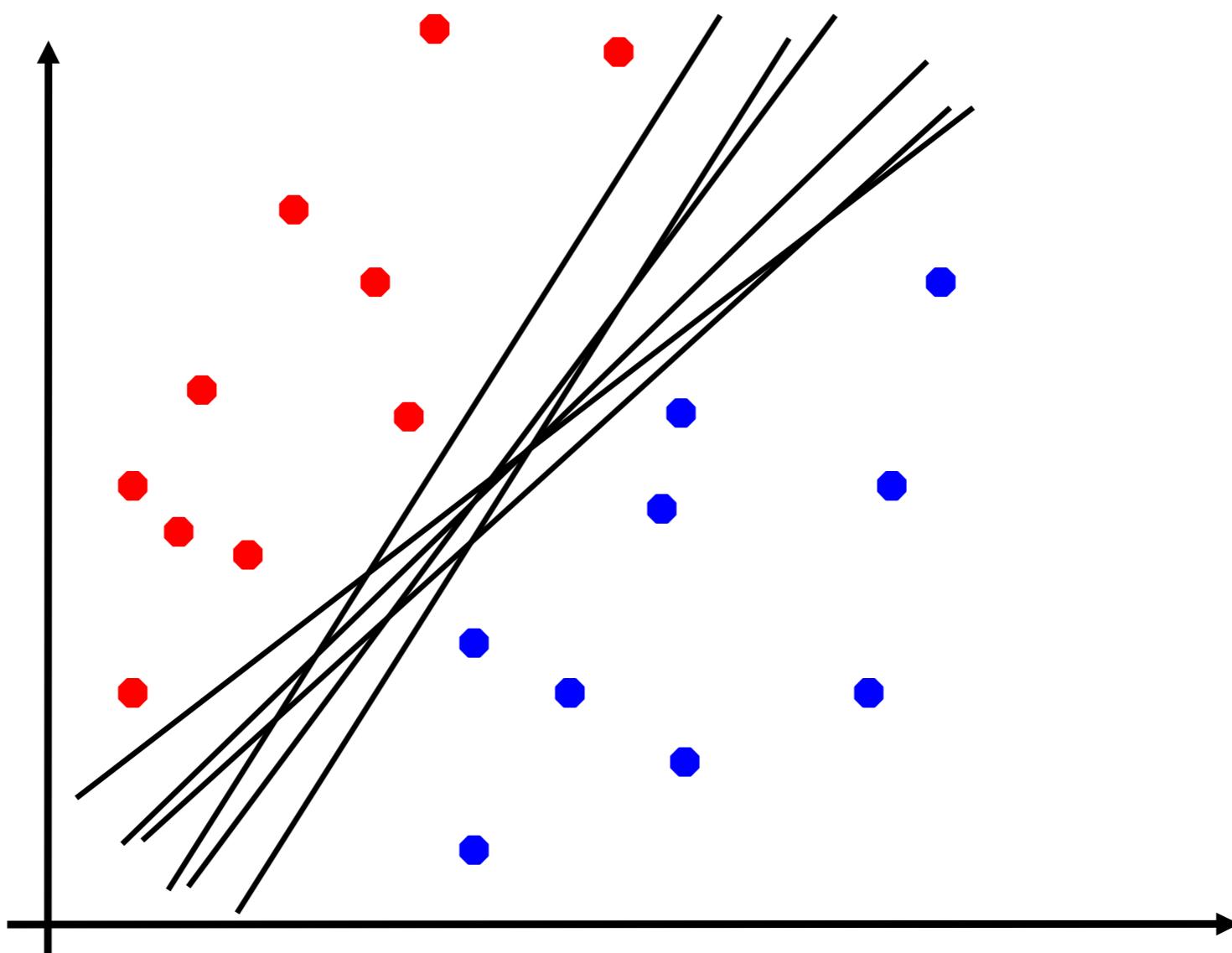
Problems with the perceptron algorithm

- **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



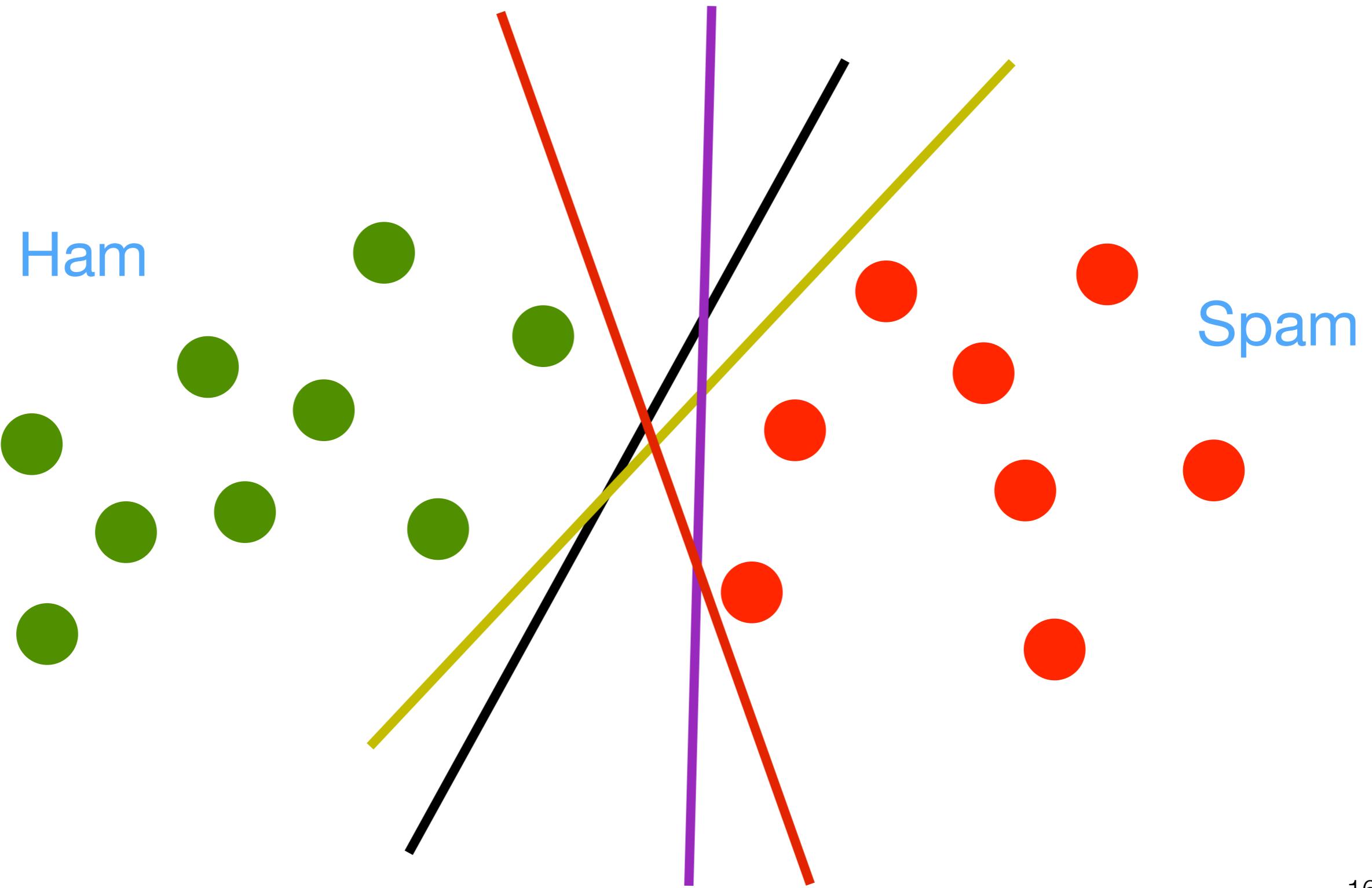
Linear Separators

- Which of these linear separators is optimal?

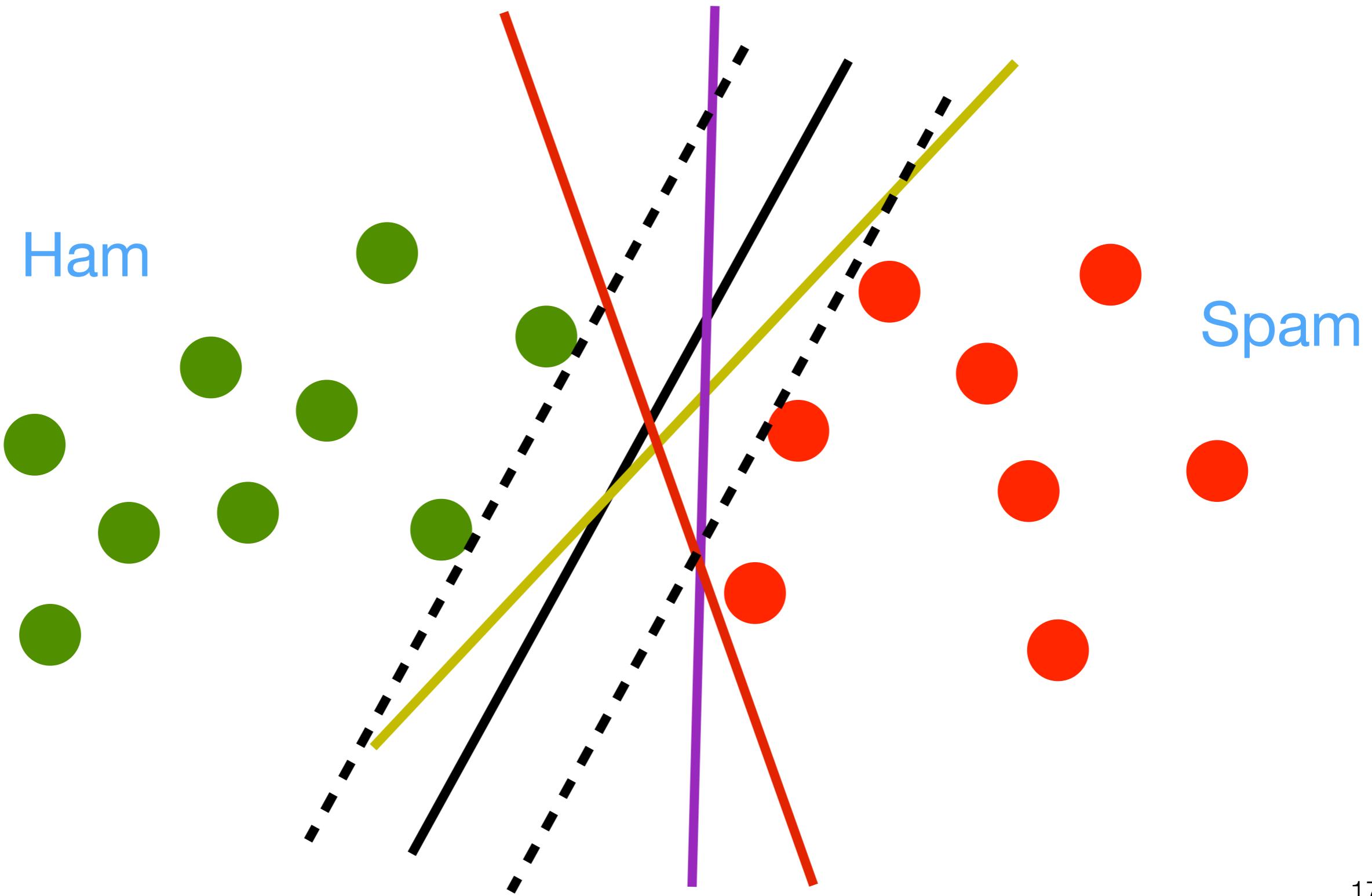


Support Vector Machines

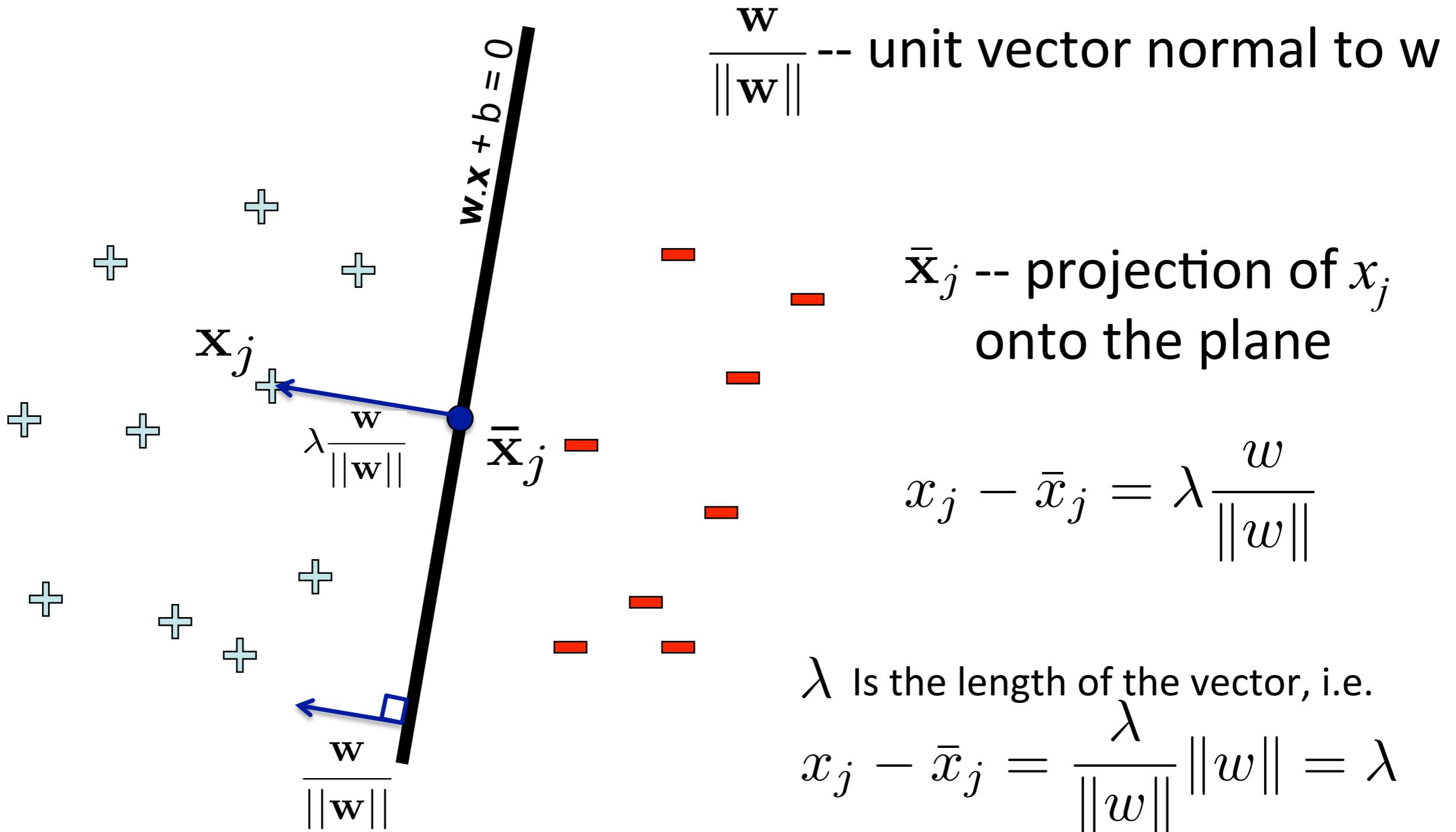
Linear Separator



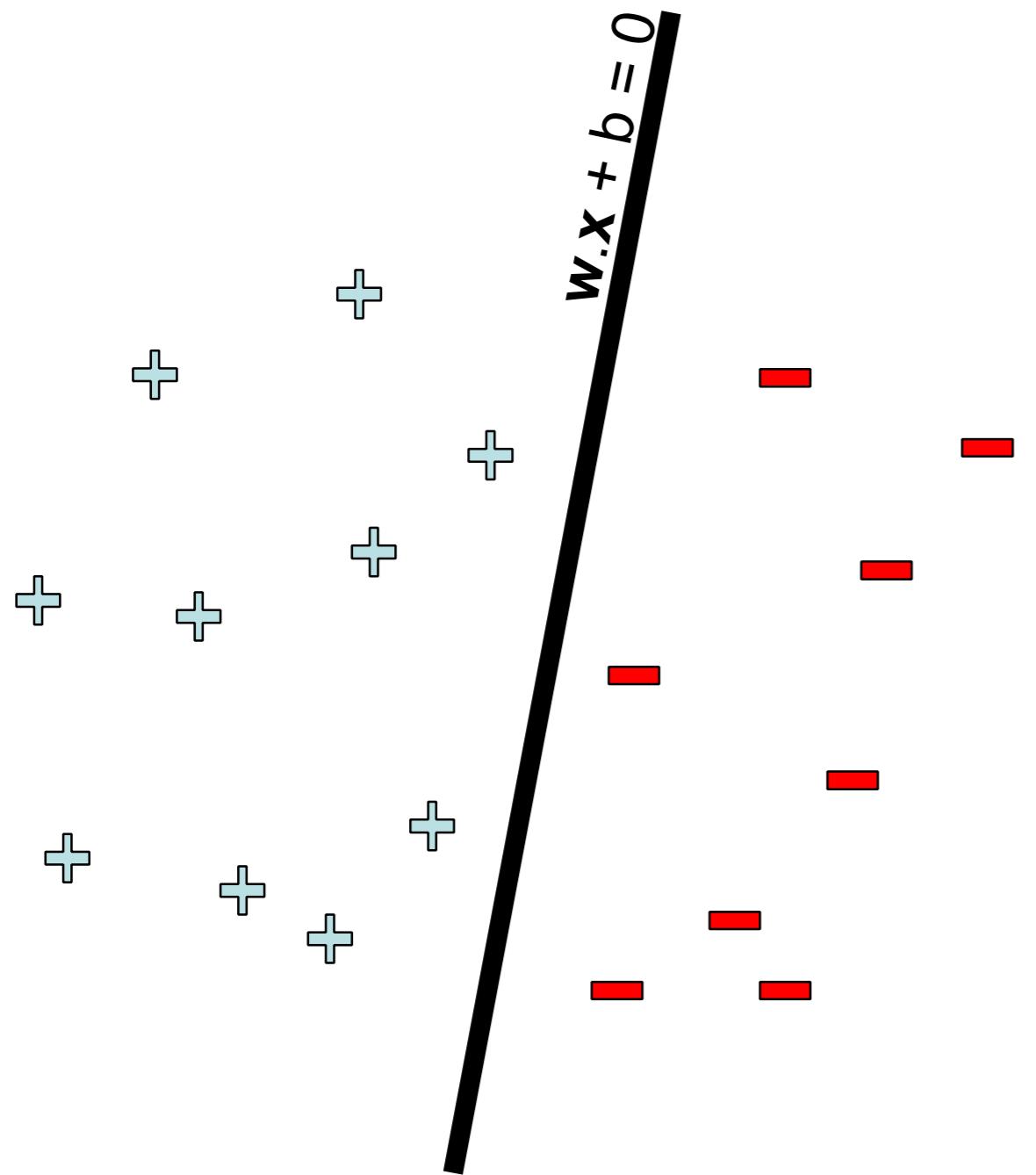
Large Margin Classifier



Review: Normal to a plane



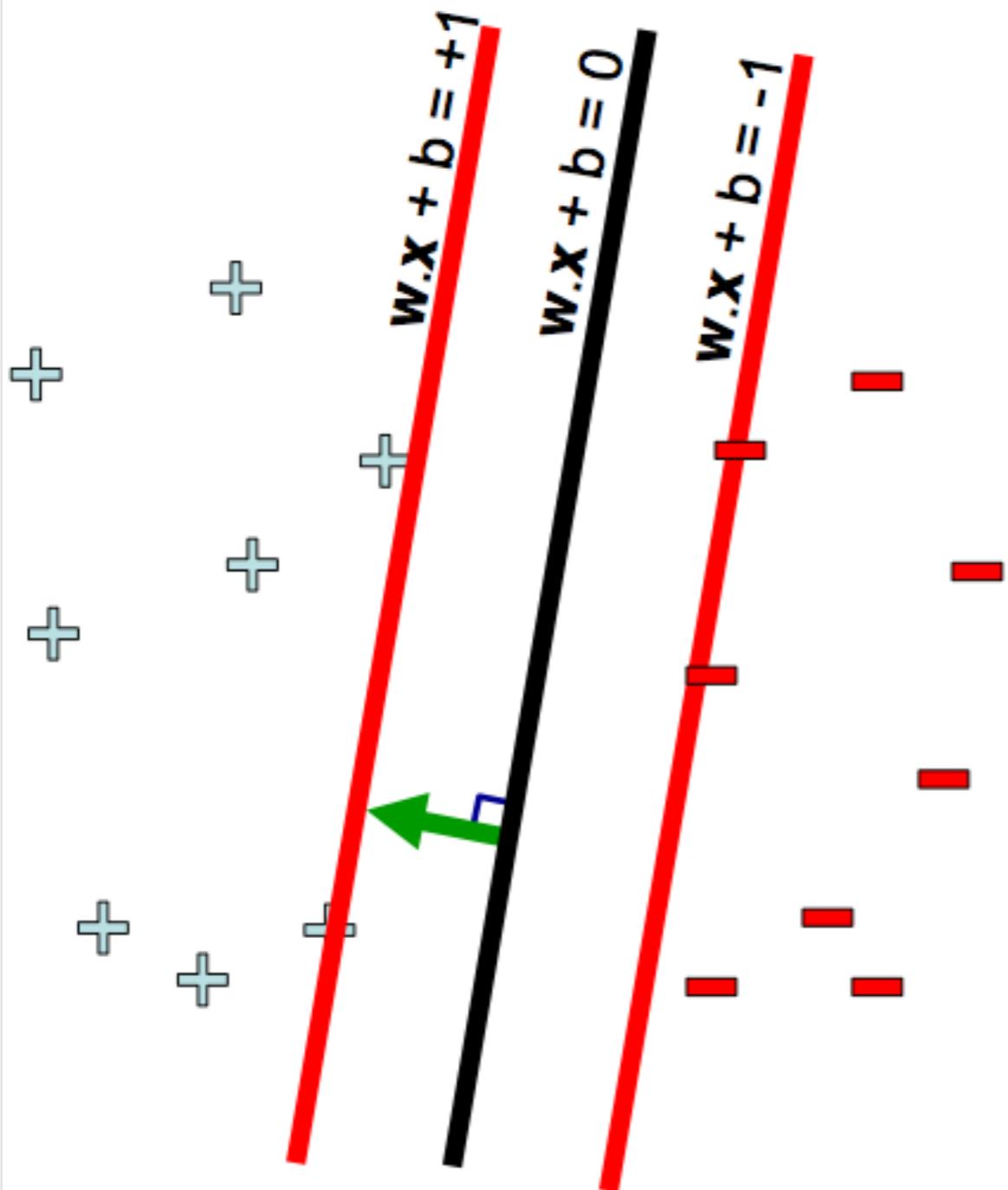
Scale invariance



Any other ways of writing the same dividing line?

- $w \cdot x + b = 0$
- $2w \cdot x + 2b = 0$
- $1000w \cdot x + 1000b = 0$
-

Scale invariance

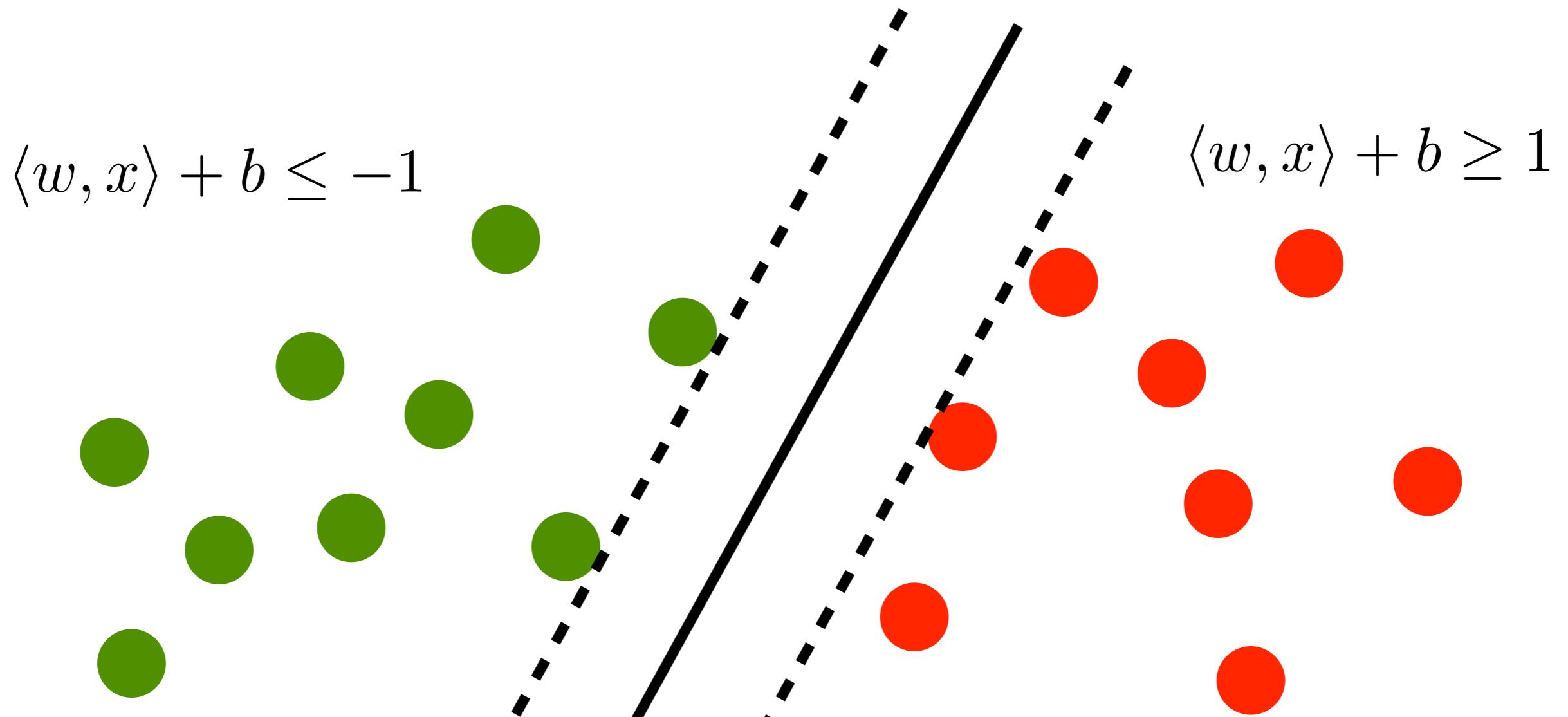


During learning, we set the scale by asking that, for all t ,
for $y_t = +1, w \cdot x_t + b \geq 1$
and for $y_t = -1, w \cdot x_t + b \leq -1$

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

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

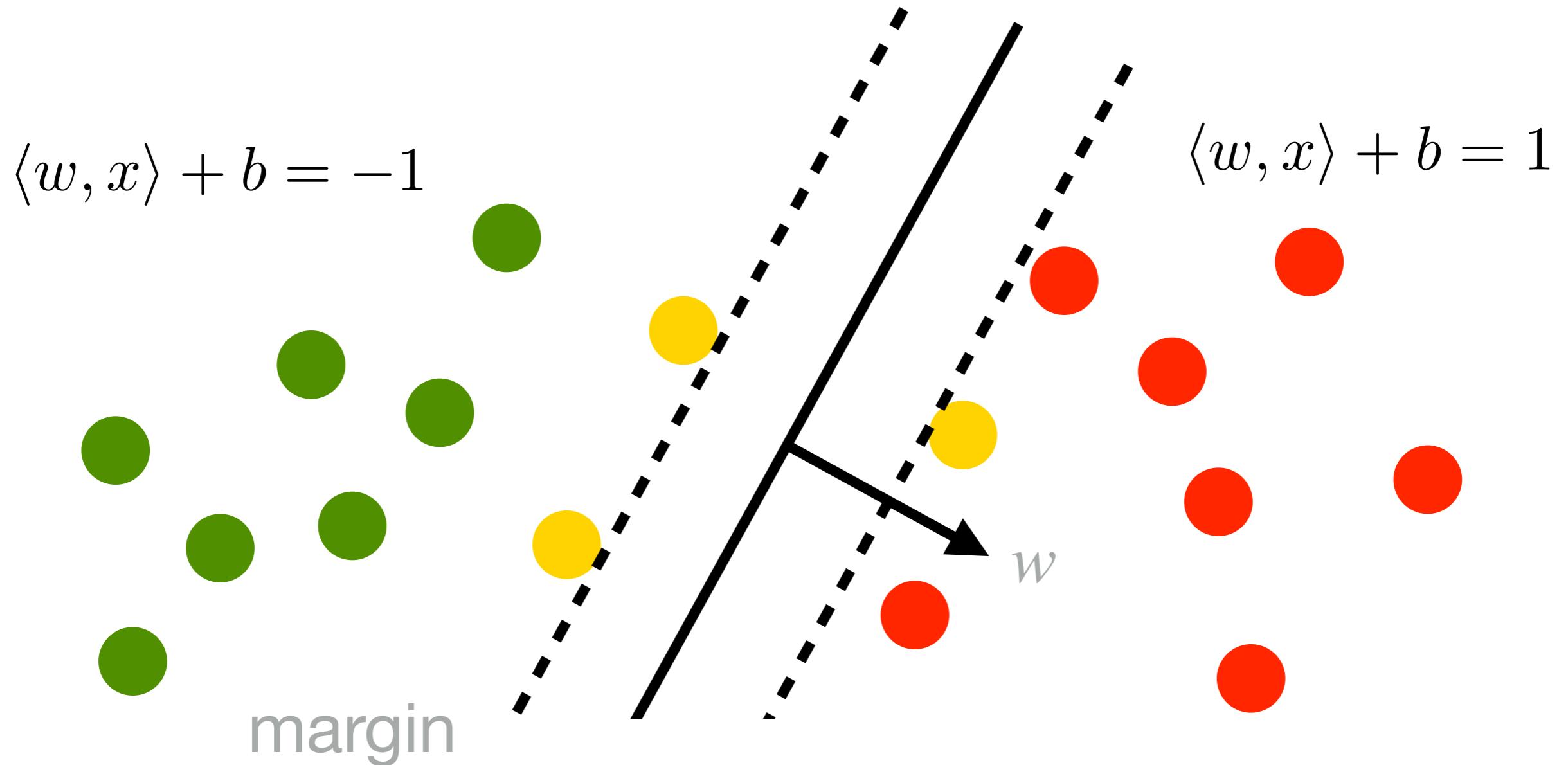
Large Margin Classifier



linear function

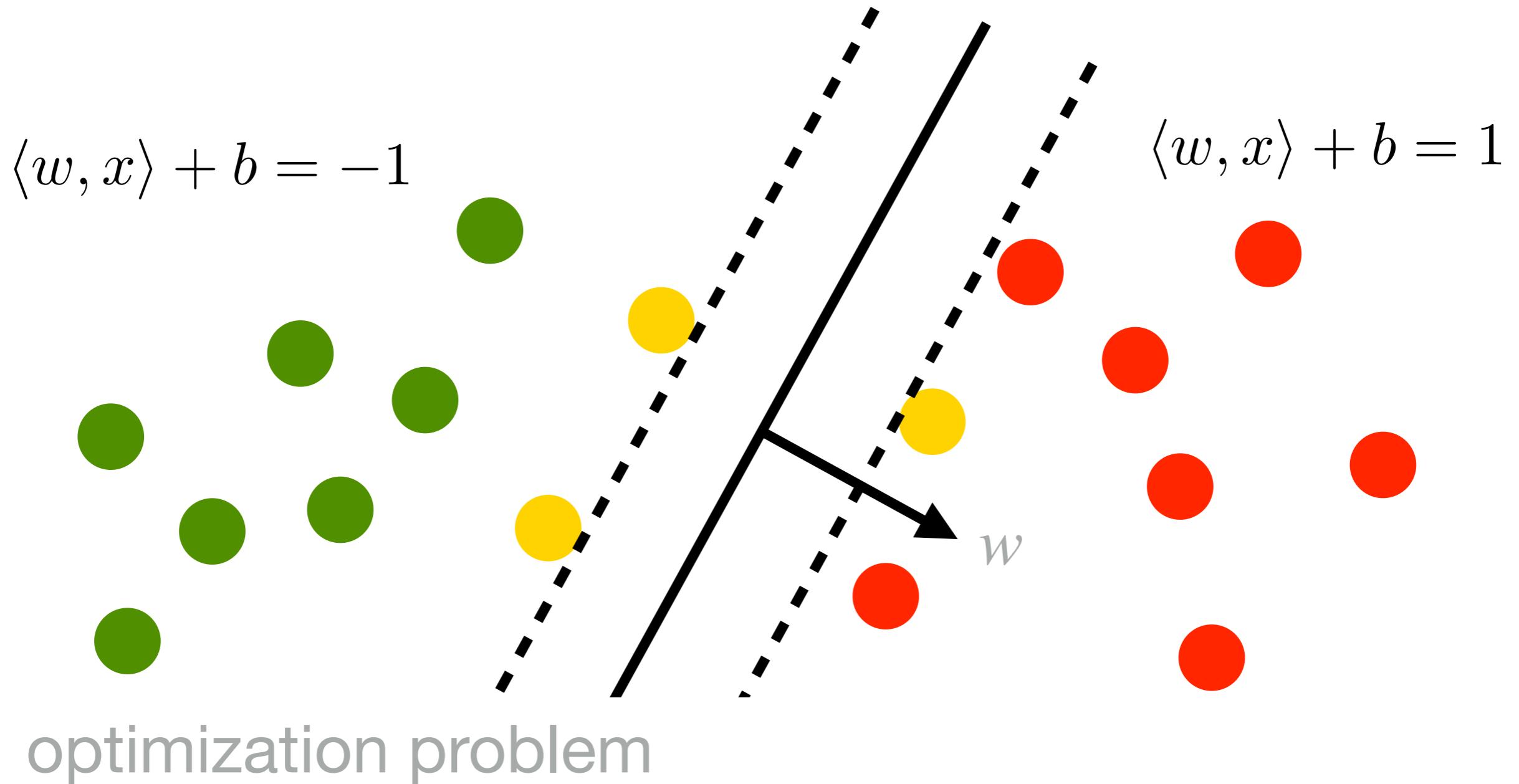
$$f(x) = \langle w, x \rangle + b$$

Large Margin Classifier



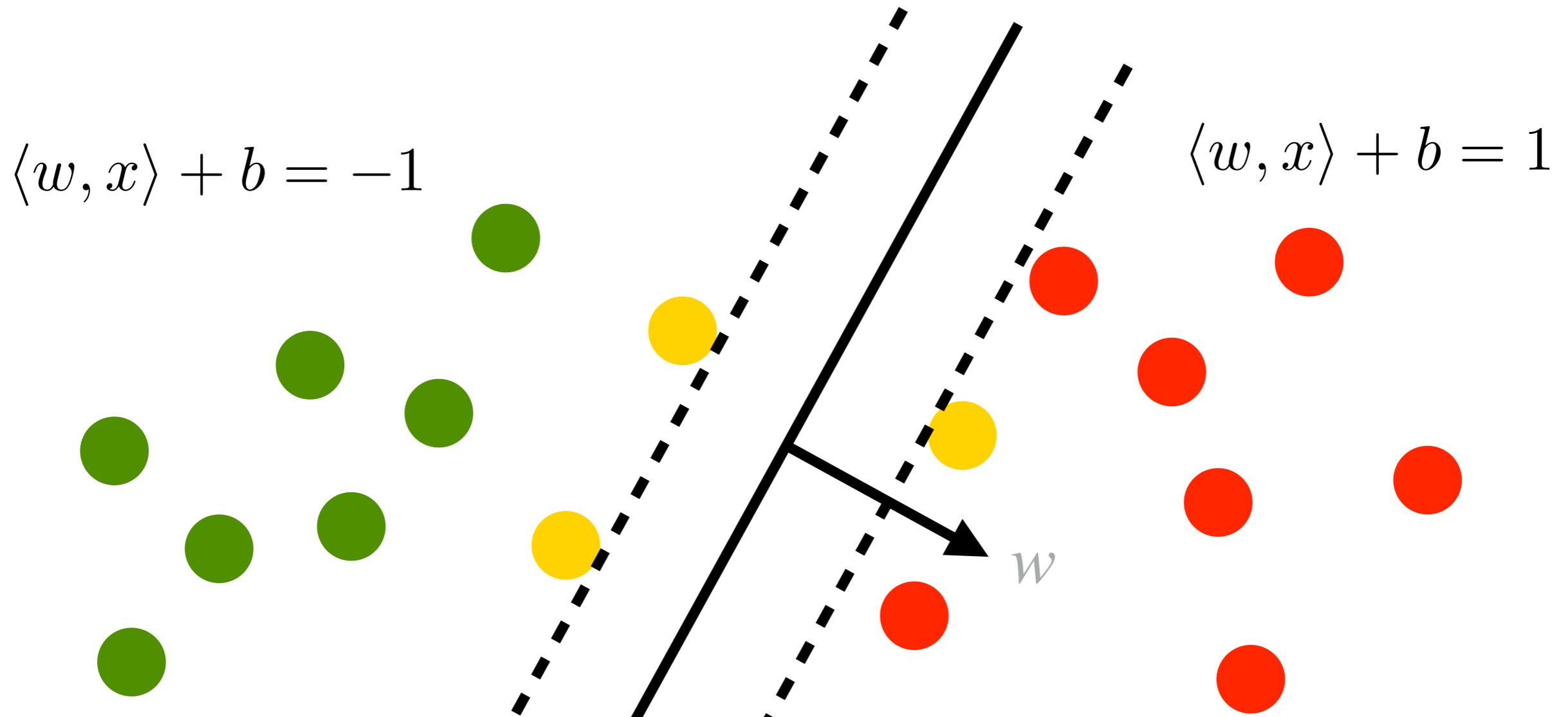
$$\frac{\langle x_+ - x_-, w \rangle}{2\|w\|} = \frac{1}{2\|w\|} [\langle x_+, w \rangle + b - \langle x_-, w \rangle - b] = \frac{1}{\|w\|}$$

Large Margin Classifier



$$\underset{w,b}{\text{maximize}} \frac{1}{\|w\|} \text{ subject to } y_i [\langle x_i, w \rangle + b] \geq 1$$

Large Margin Classifier



optimization problem

$$\underset{w,b}{\text{minimize}} \frac{1}{2} \|w\|^2 \text{ subject to } y_i [\langle x_i, w \rangle + b] \geq 1$$

Convex Programs for Dummies

- Primal optimization problem

$$\underset{x}{\text{minimize}} \, f(x) \text{ subject to } c_i(x) \leq 0$$

- Lagrange function

$$L(x, \alpha) = f(x) + \sum \alpha_i c_i(x)$$

- First order optimality conditions in x

$$\partial_x L(x, \alpha) = \partial_x f(x) + \sum_i \alpha_i \partial_x c_i(x) = 0$$

- Solve for x and plug it back into L

$$\underset{\alpha}{\text{maximize}} \, L(x(\alpha), \alpha)$$

(keep explicit constraints)

Dual Problem

- Primal optimization problem

$$\underset{w,b}{\text{minimize}} \frac{1}{2} \|w\|^2 \text{ subject to } y_i [\langle x_i, w \rangle + b] \geq 1$$

- Lagrange function

$$L(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_i \alpha_i [y_i [\langle x_i, w \rangle + b] - 1]$$

constraint

Optimality in w, b is at saddle point with α

- Derivatives in w, b need to vanish

Dual Problem

- Lagrange function

$$L(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_i \alpha_i [y_i [\langle x_i, w \rangle + b] - 1]$$

- Derivatives in w, b need to vanish

$$\partial_w L(w, b, \alpha) = w - \sum_i \alpha_i y_i x_i = 0$$

$$\partial_b L(w, b, \alpha) = \sum_i \alpha_i y_i = 0$$

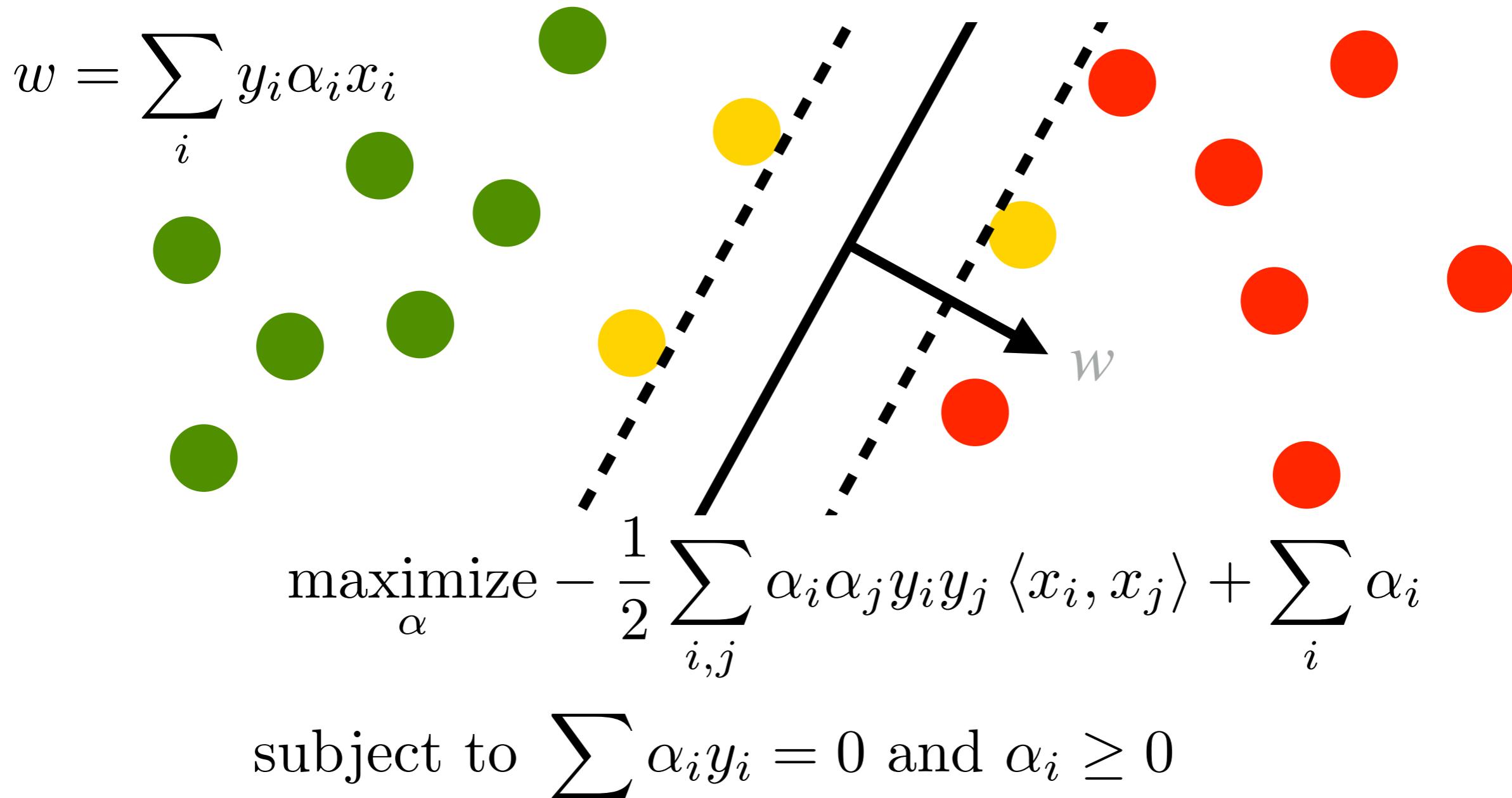
- Plugging terms back into L yields

$$\underset{\alpha}{\text{maximize}} -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + \sum_i \alpha_i$$

subject to $\sum_i \alpha_i y_i = 0$ and $\alpha_i \geq 0$

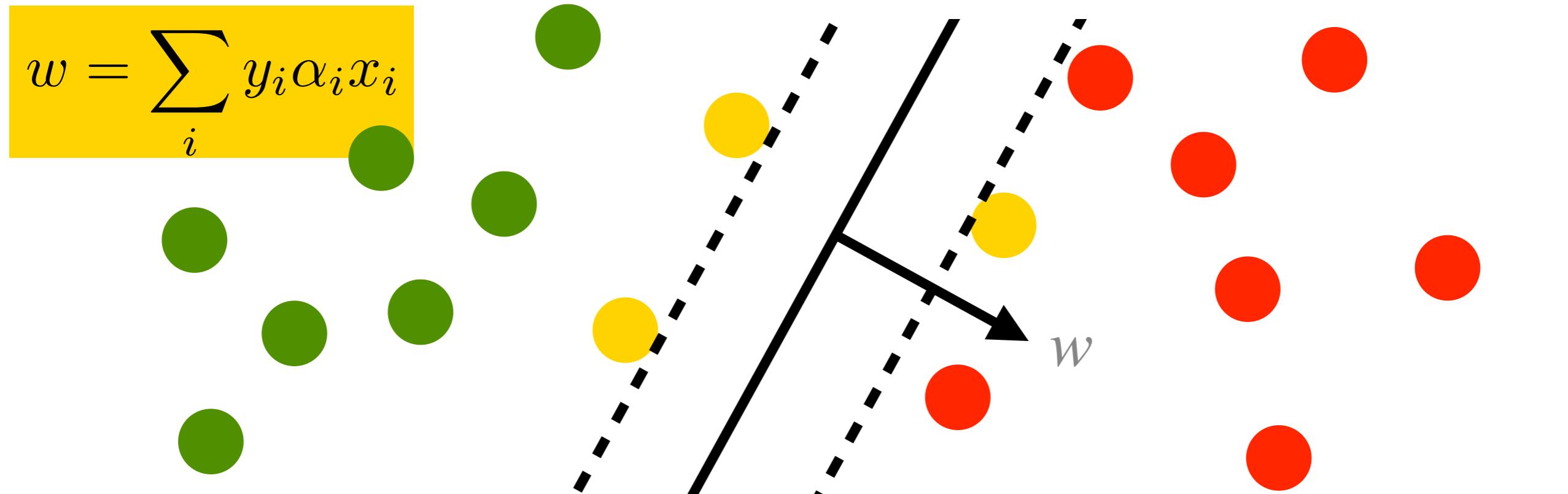
Support Vector Machines

$$\underset{w,b}{\text{minimize}} \frac{1}{2} \|w\|^2 \text{ subject to } y_i [\langle x_i, w \rangle + b] \geq 1$$

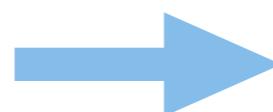


Support Vectors

$$\underset{w,b}{\text{minimize}} \frac{1}{2} \|w\|^2 \text{ subject to } y_i [\langle x_i, w \rangle + b] \geq 1$$



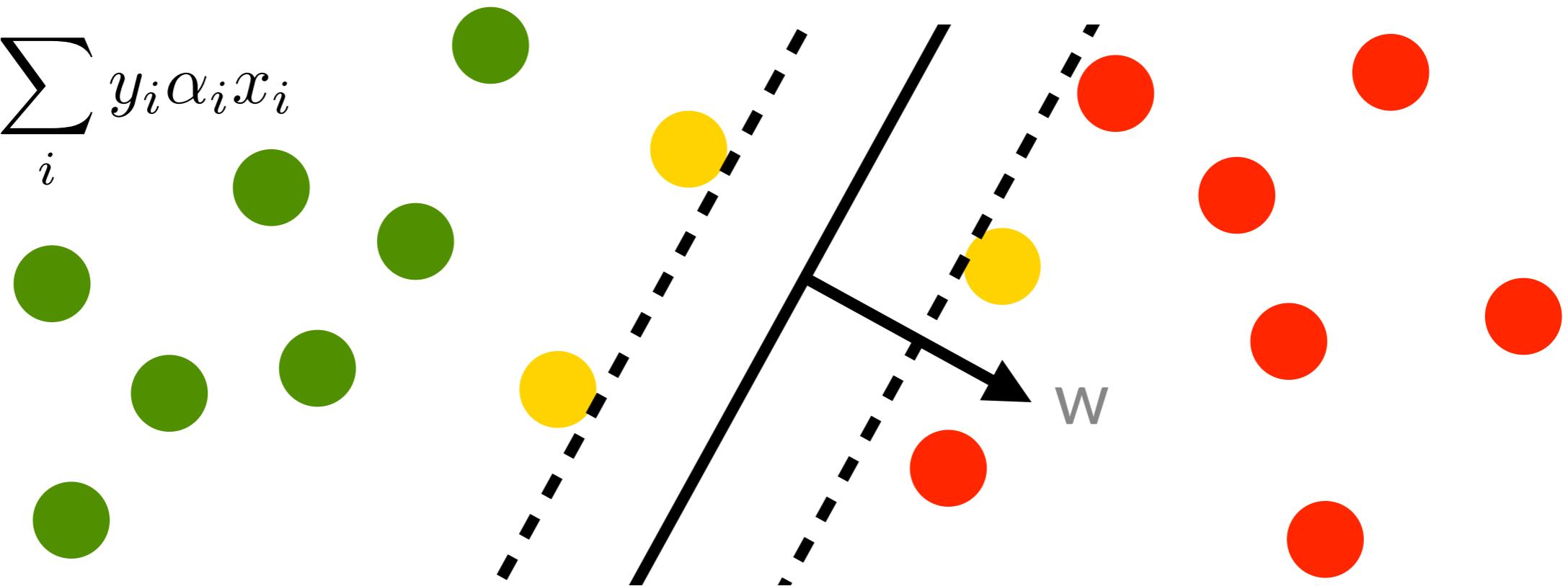
Karush Kuhn Tucker
Optimality condition



$$\alpha_i = 0$$
$$\alpha_i > 0 \implies y_i [\langle w, x_i \rangle + b] = 1$$

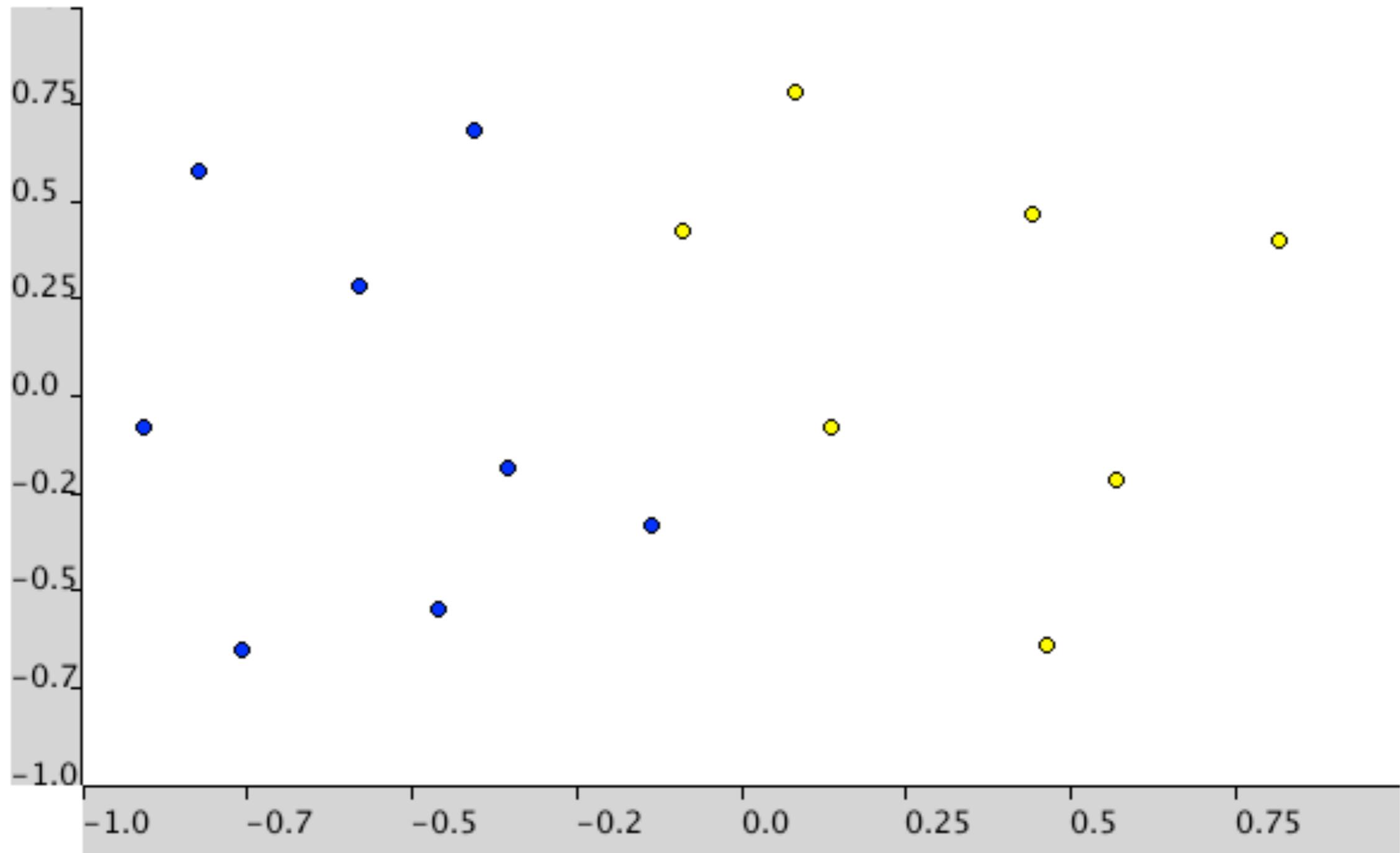
Properties

$$w = \sum_i y_i \alpha_i x_i$$



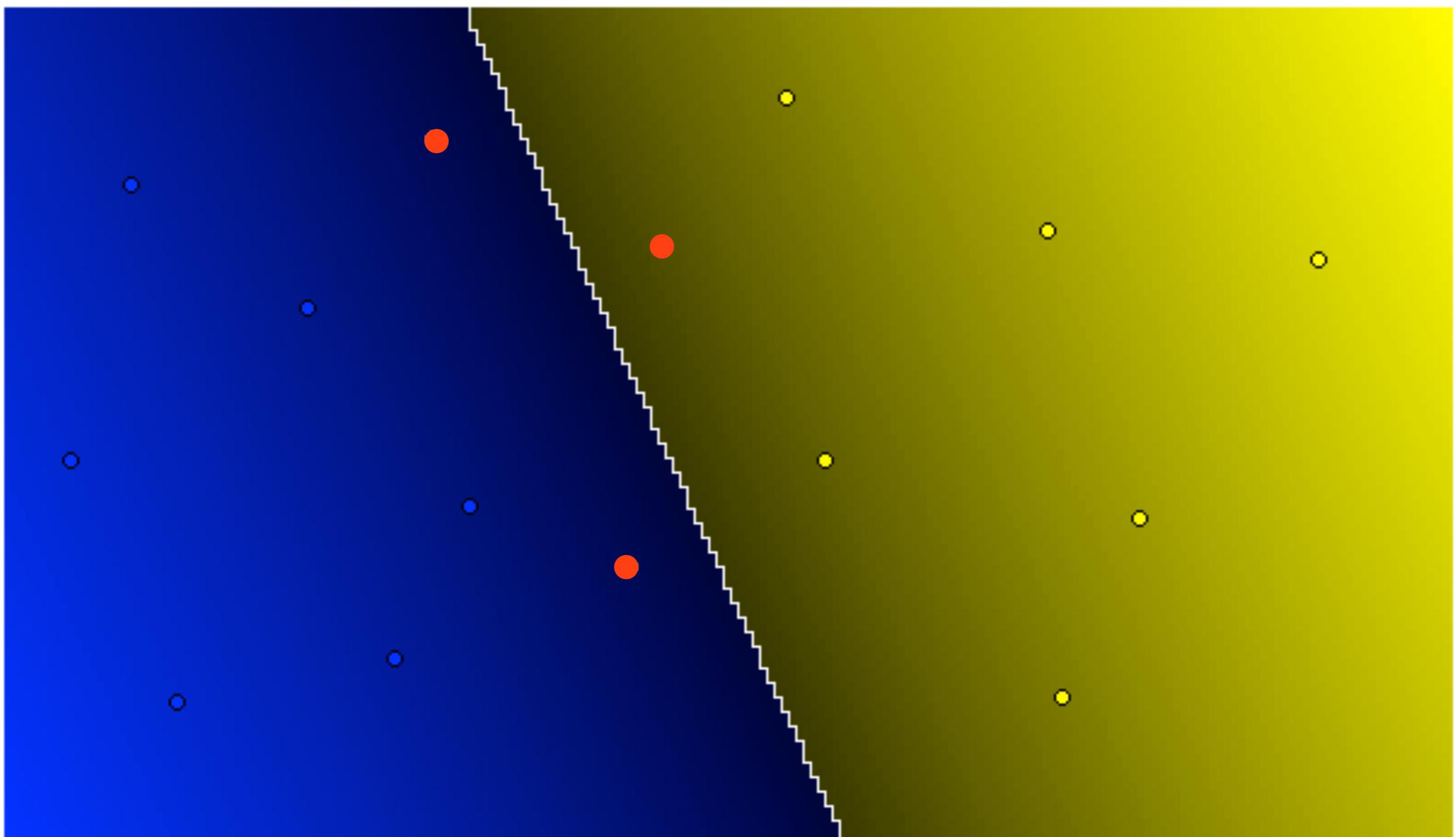
- Weight vector w as weighted linear combination of instances
- Only points on margin matter (ignore the rest and get same solution)
- Only inner products matter
 - Quadratic program
 - We can replace the inner product by a kernel
- Keeps instances away from the margin

Example

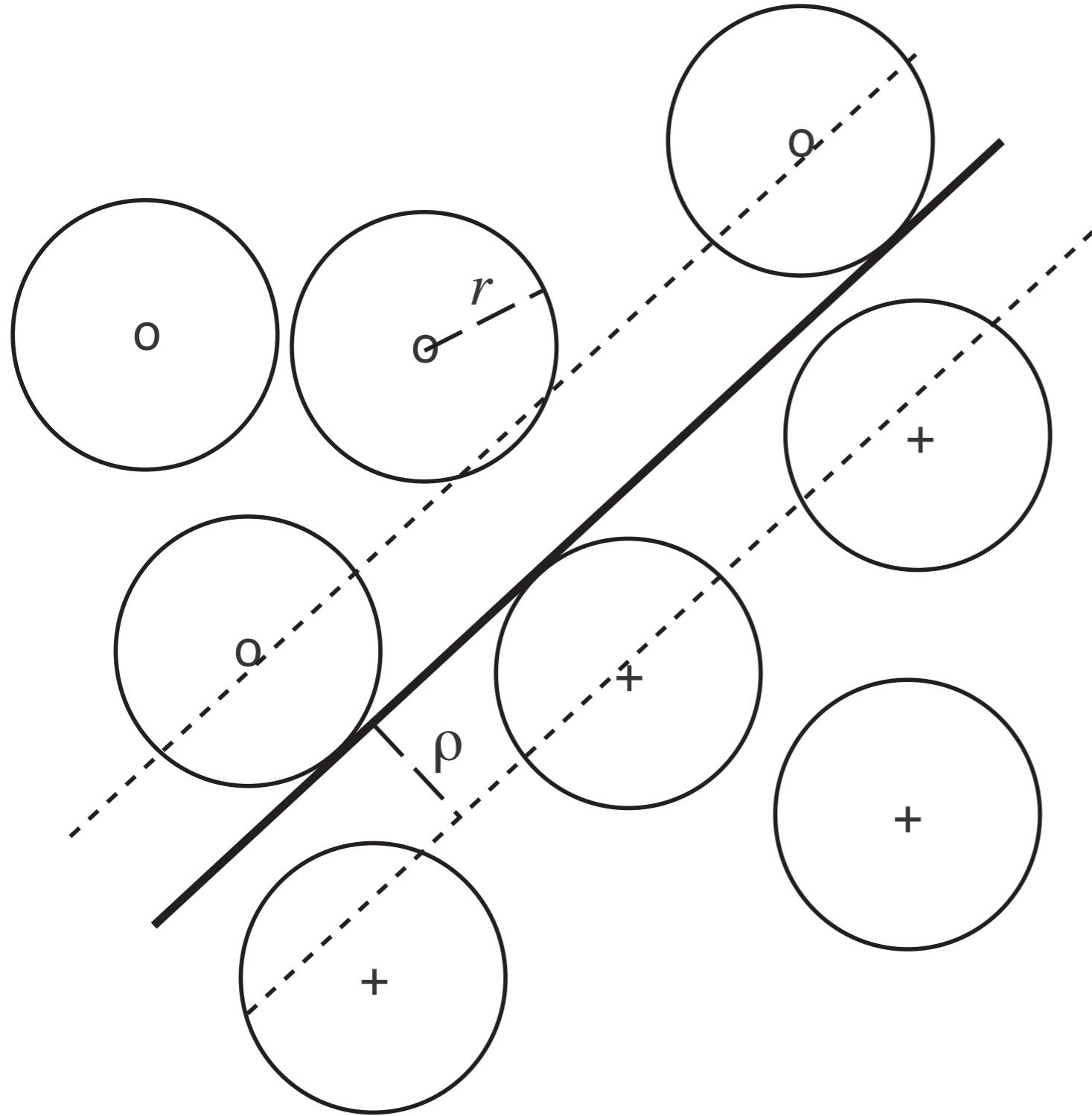


Example

Number of Support Vectors: 3 (-ve: 2, +ve: 1) Total number of points: 15

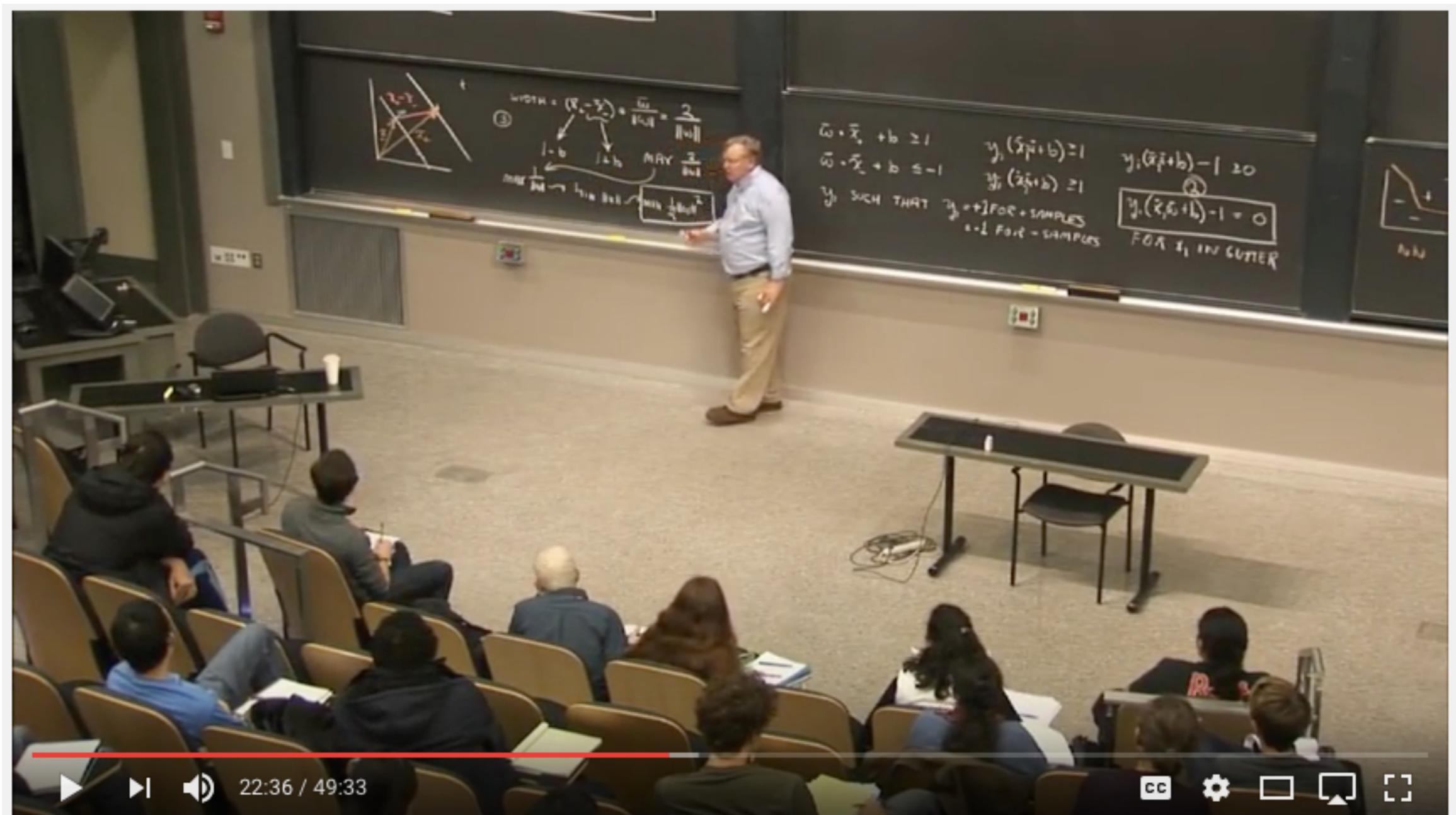


Why Large Margins?



- Maximum robustness relative to uncertainty
- Symmetry breaking
- Independent of correctly classified instances
- Easy to find for easy problems

Watch: Patrick Winston, Support Vector Machines



<https://www.youtube.com/watch?v=PwhiWxHK8o>

Next Lecture:
Soft Margin Classification,
Multi-class SVMs