Lecture 6: The SVM classifier

Slide credit: many of the slides were transcribed and adapted from
Andrew Zisserman (“Machine Learning”, lecture 2) and
Yaser S. Abu-Mostafa (“Learning from Data”, lecture 14)

Outline

- Review of linear classifiers
  - Realizability == Linear separability
  - Revisiting perceptron
- Support Vector machine (SVM) classifier
  - Wide margin principle
  - Derivation
  - Dual form
  - Slack variables
  - Loss function
  - Multi class
  - Practical advice

Binary Classification

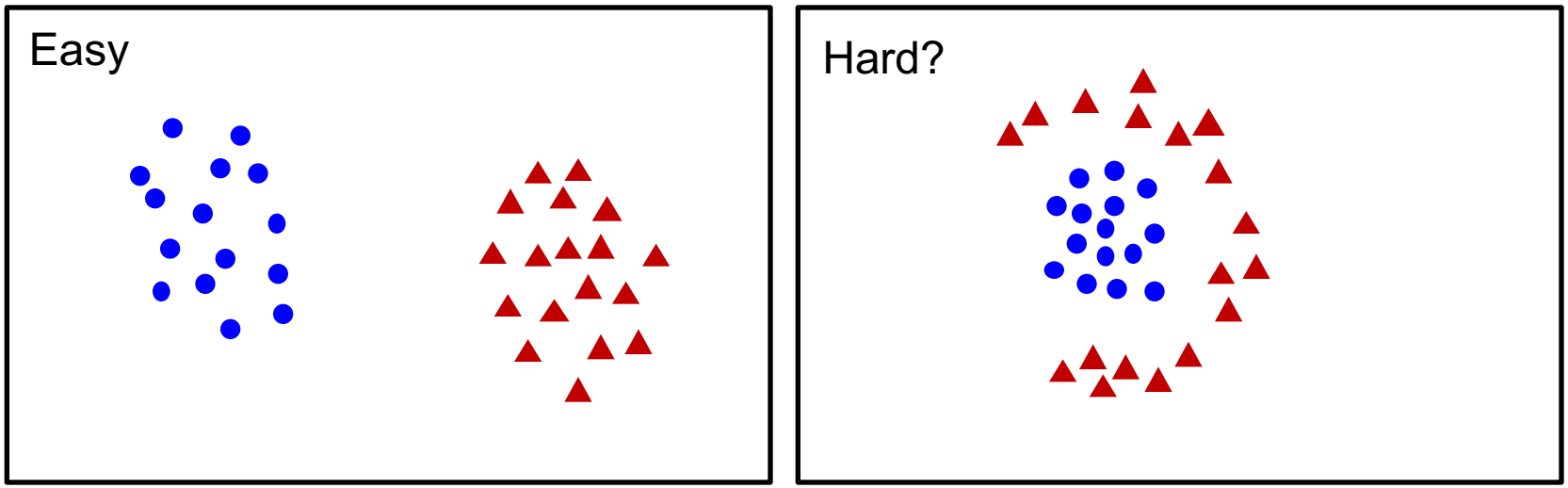
Given training data $\{(\mathrm{x}_i, y_i)\}_{i=1..N}$

with $\mathrm{x}_i \in \mathrm{R}^d$ and $y_i \in \{-1,1\}$

learn a classifier $f(\mathrm{x})$ such that

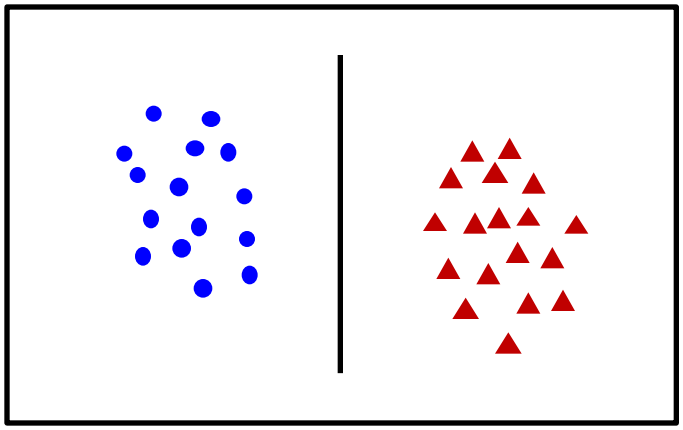
$$f(\mathrm{x}_i) \begin{cases} \geq 0 & y_i = +1 \\ < 0 & y_i = -1 \end{cases}$$

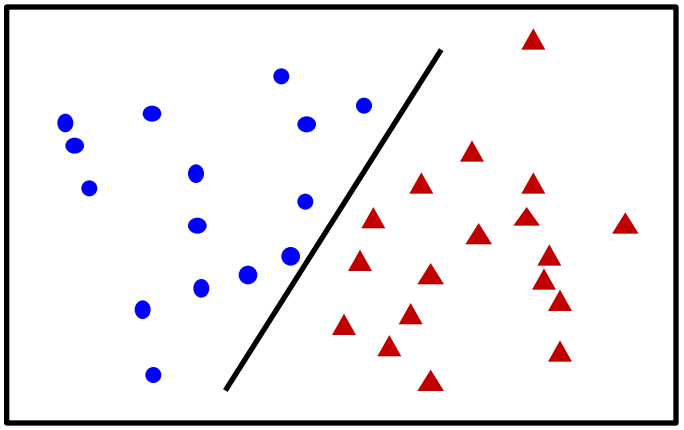
i.e., $y_i f(\mathrm{x}_i) > 0$ for a correct classification



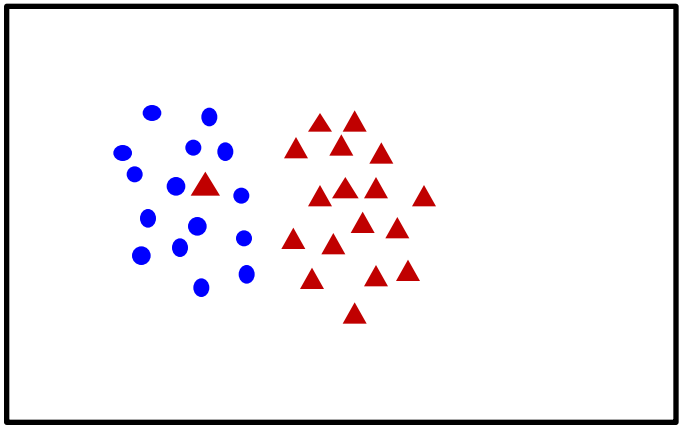
Linear separability

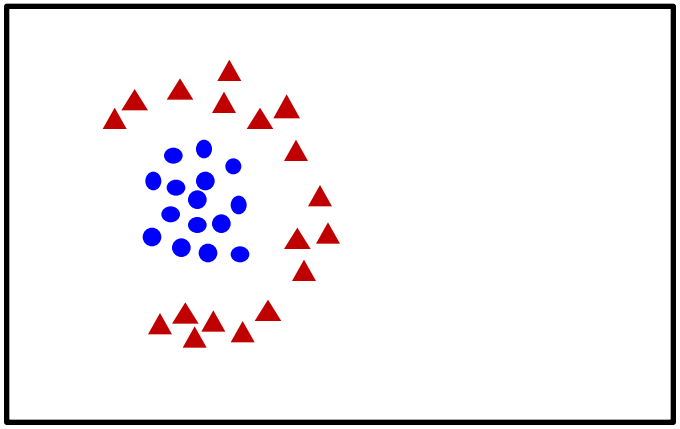
linearly separable





not linearly separable

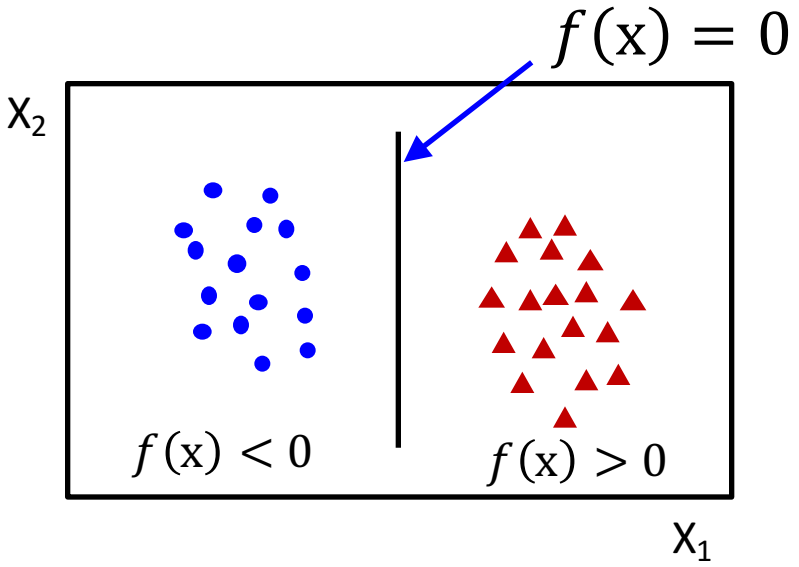




Linear classifiers

A linear classifier has the form

$$f(\mathrm{x}) = \mathrm{w}^{\mathrm{T}}\mathrm{x} + b$$

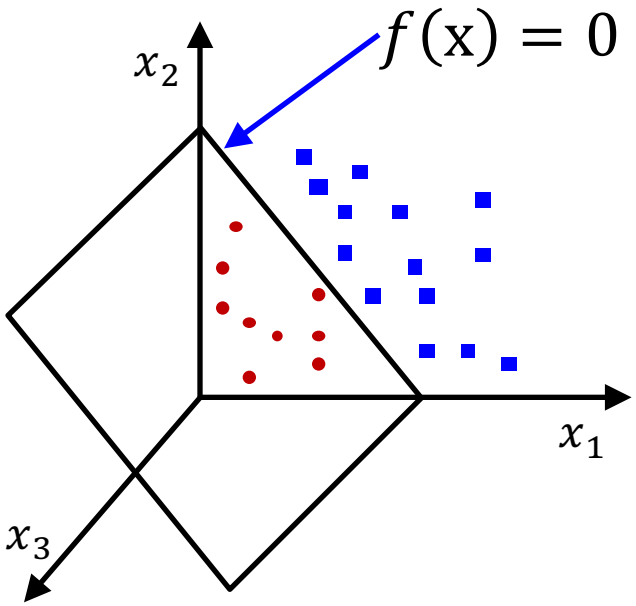


- in 2D the discriminant is a line
- w is the normal to the line, and b the bias
- w is known as the weight vector
- Before we added x_0=1 and used w_0 instead of b

Linear classifiers

A linear classifier has the form

$$f(\mathrm{x}) = \mathrm{w}^{\mathrm{T}}\mathrm{x} + b$$



- in 3D the discriminant is a plane, and in nD it is a hyperplane

Reminder: The Perceptron Classifier

Given linearly separable data $\mathrm{x_i}$ labelled into two categories $\mathrm{y_i} = \{-1,1\}$, find a way to vector w such that the discriminant function

$$f(\mathrm{x}_i) = \mathrm{w}^{\mathrm{T}}\mathrm{x}_i + b$$

Separates the categories for i=1,…,N

- How can we find this separating hyperplane?

The Perceptron Algorithm (class 4)

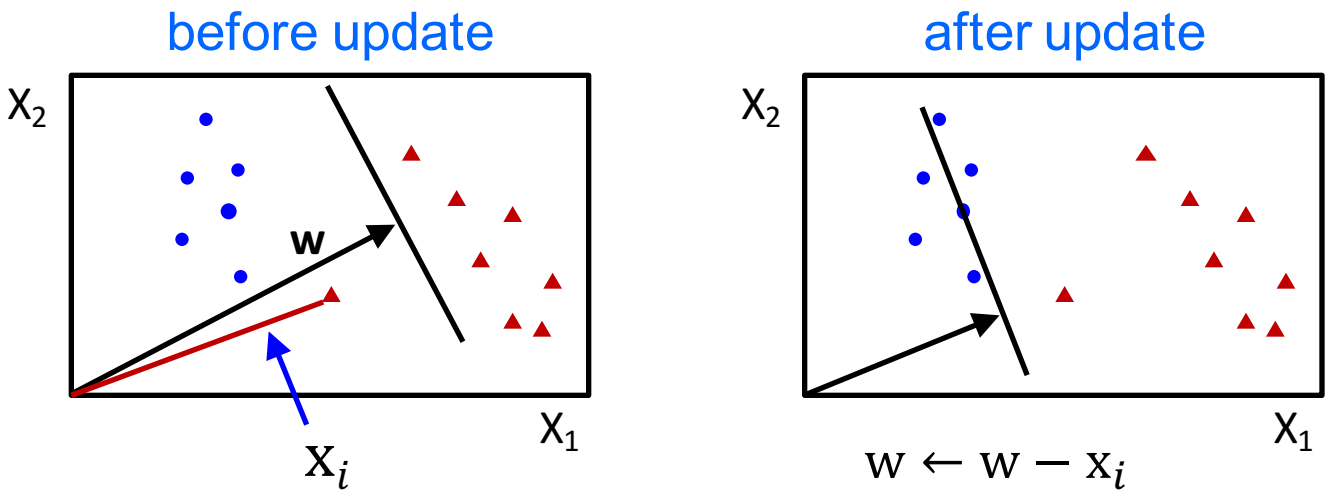
Write classifier as $f(\mathrm{x}_i) = \widetilde{\mathrm{w}}^{\mathrm{T}}\tilde{\mathrm{x}}_i + \omega_0 = \mathrm{w}^{\mathrm{T}}\mathrm{x}_i$

where $\mathrm{w} = (\widetilde{\mathrm{w}}, \omega_0), \mathrm{x}_i = (\tilde{\mathrm{x}}_i, 1)$

- Initialize w = 0
- Cycle though the data points $\{\mathrm{x_i}, \mathrm{y_i}\}$
  - if $\mathrm{x_i}$ is misclassified then $\mathrm{w} \leftarrow \mathrm{w} - \mathrm{sign}\big(f(\mathrm{x}_i)\big)x_i$
- Until all the data is correctly classified

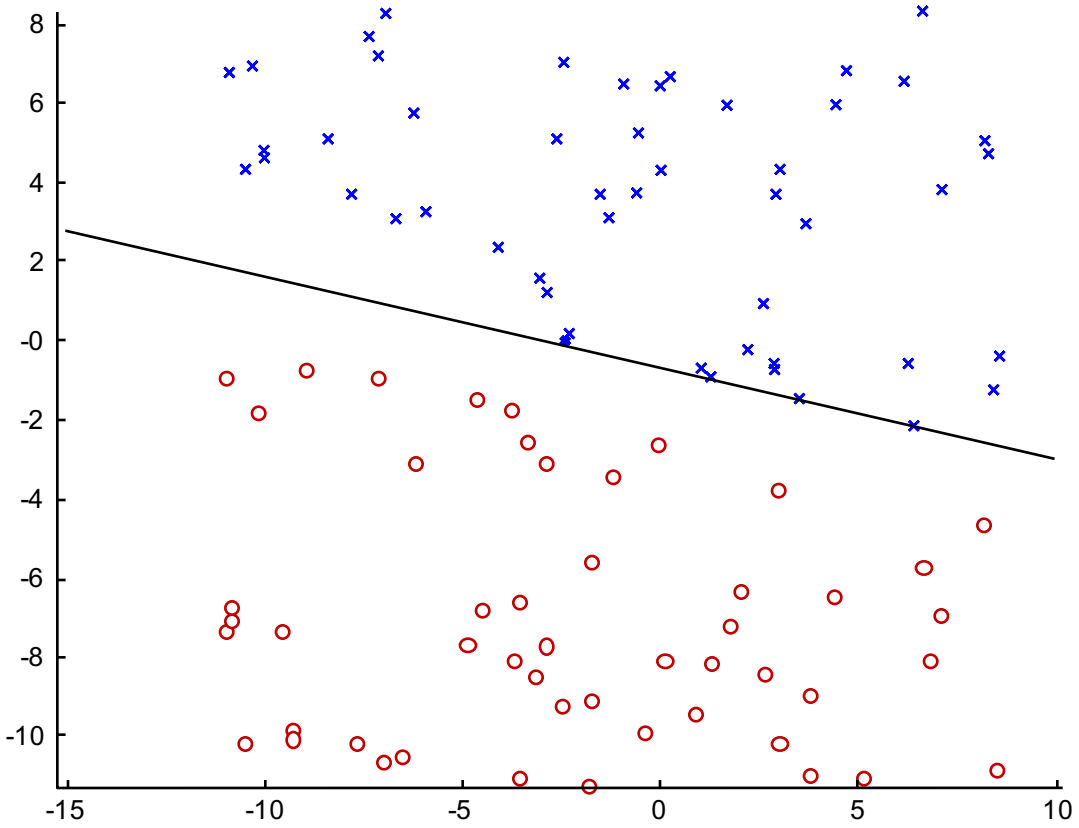
For example in 2D

- Initialize w = 0
- Cycle though the data points $\{x_i, y_i\}$
  - If x_i is misclassified then $w \leftarrow w + \text{sign}(f(x_i))x_i$
- Until all the data is correctly classified



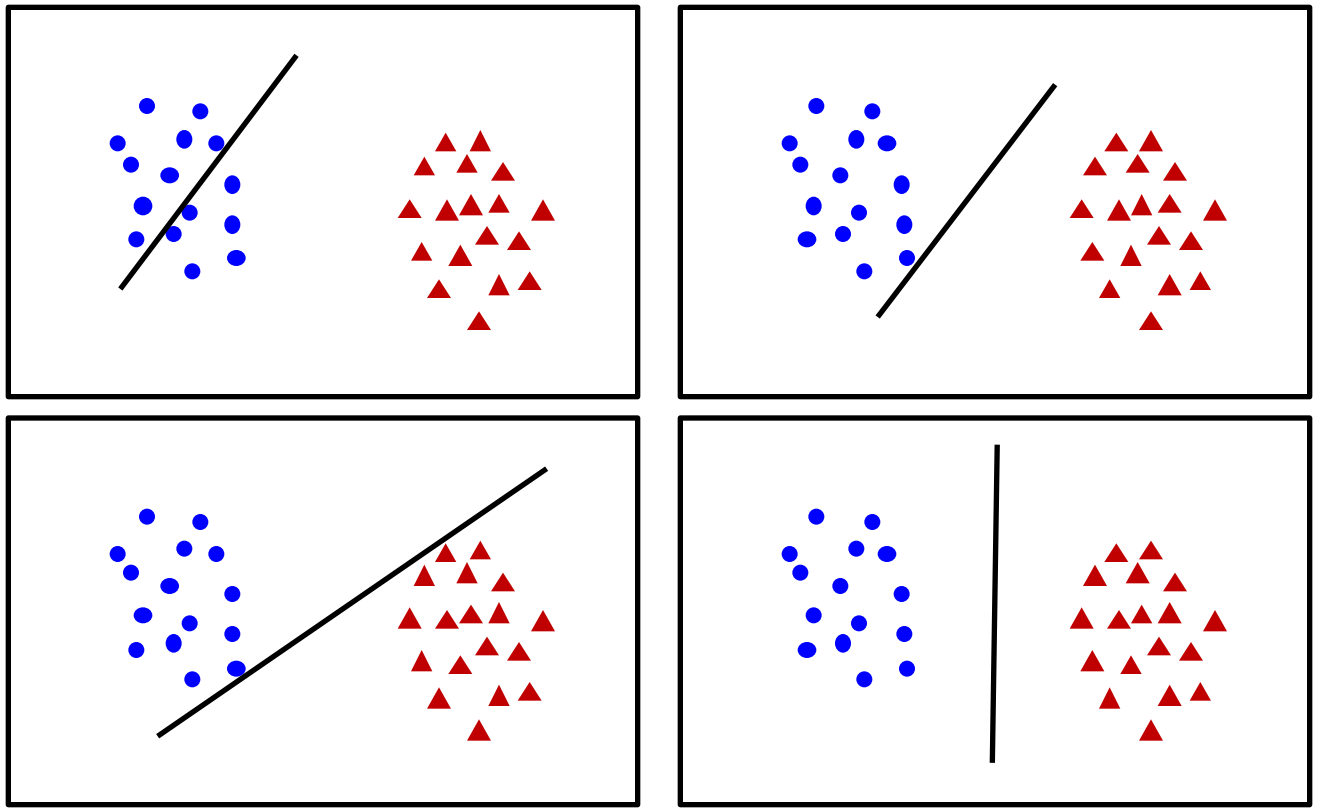
After convergence $w = \sum_i^N \alpha_i x_i$

Properties of the Perceptron Algorithm



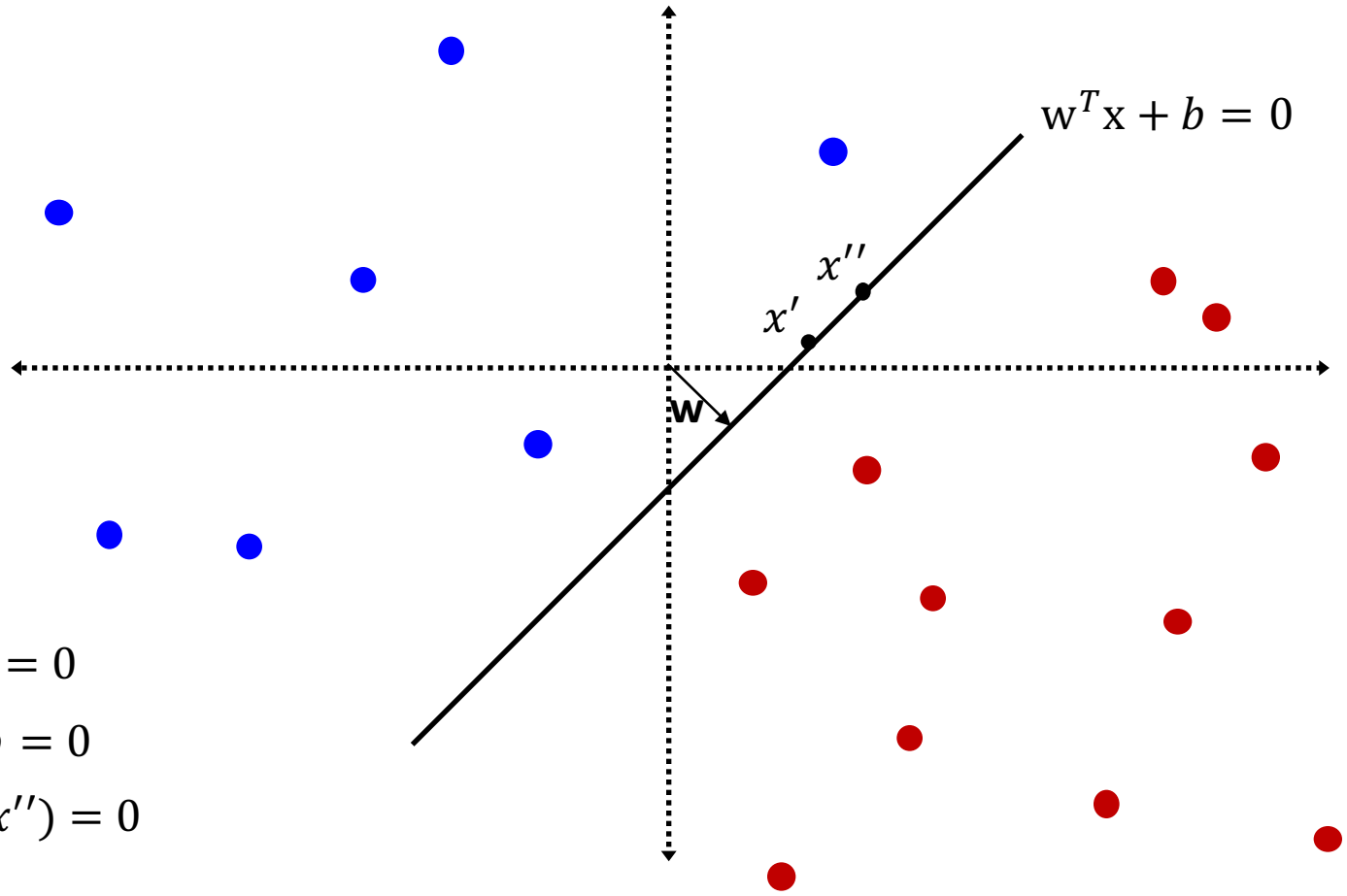
- Guaranteed convergence in the realizable case
- convergence can be very slow
- separating line close to training data

What is the best w?



- maximum margin solution
  - most stable under perturbations of the inputs
  - more confident classification for all examples

What is w?

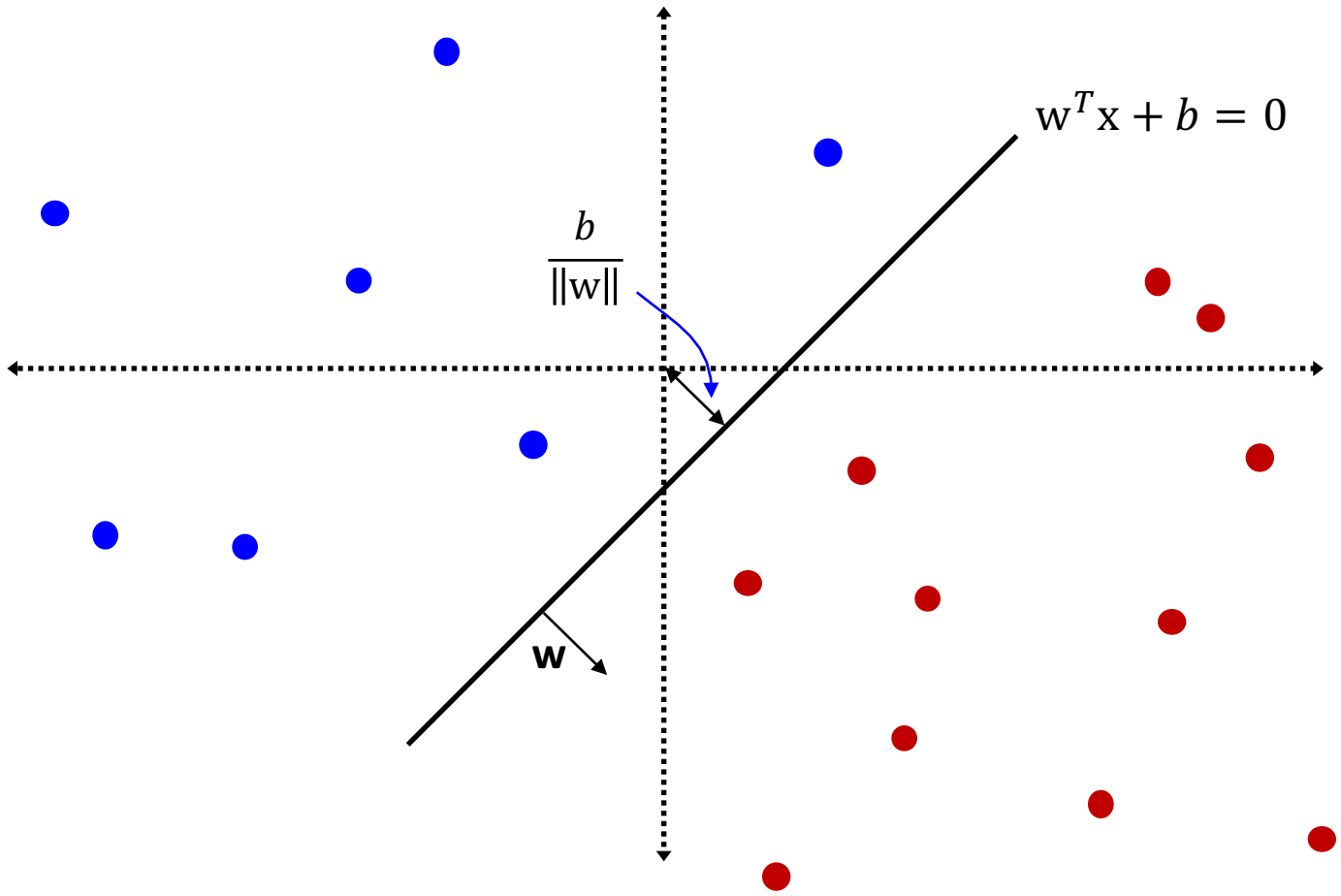


$\mathrm{w}^T x' + b = 0$

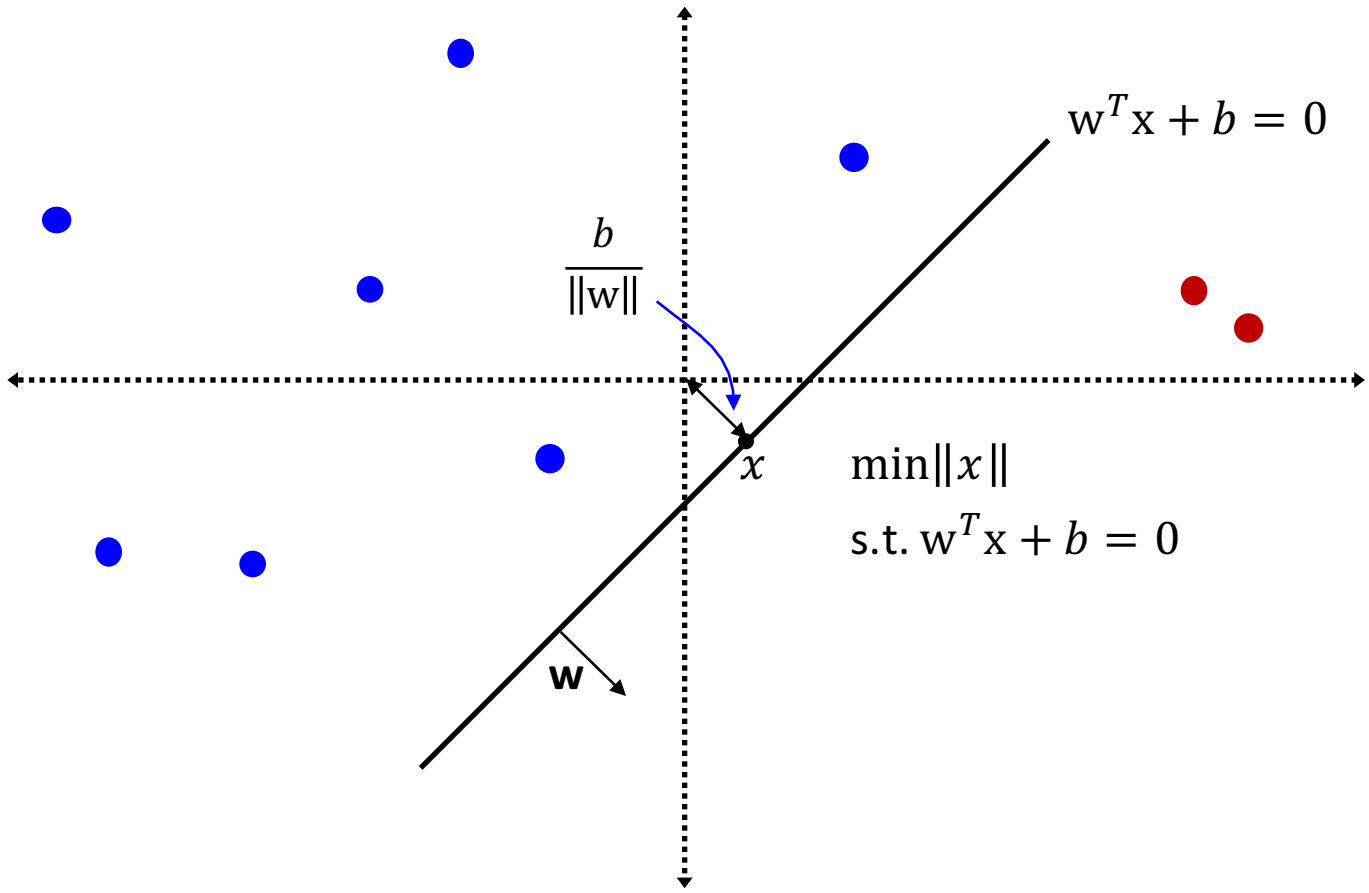
$\mathrm{w}^T x'' + b = 0$

$\mathrm{w}^T (x' - x'') = 0$

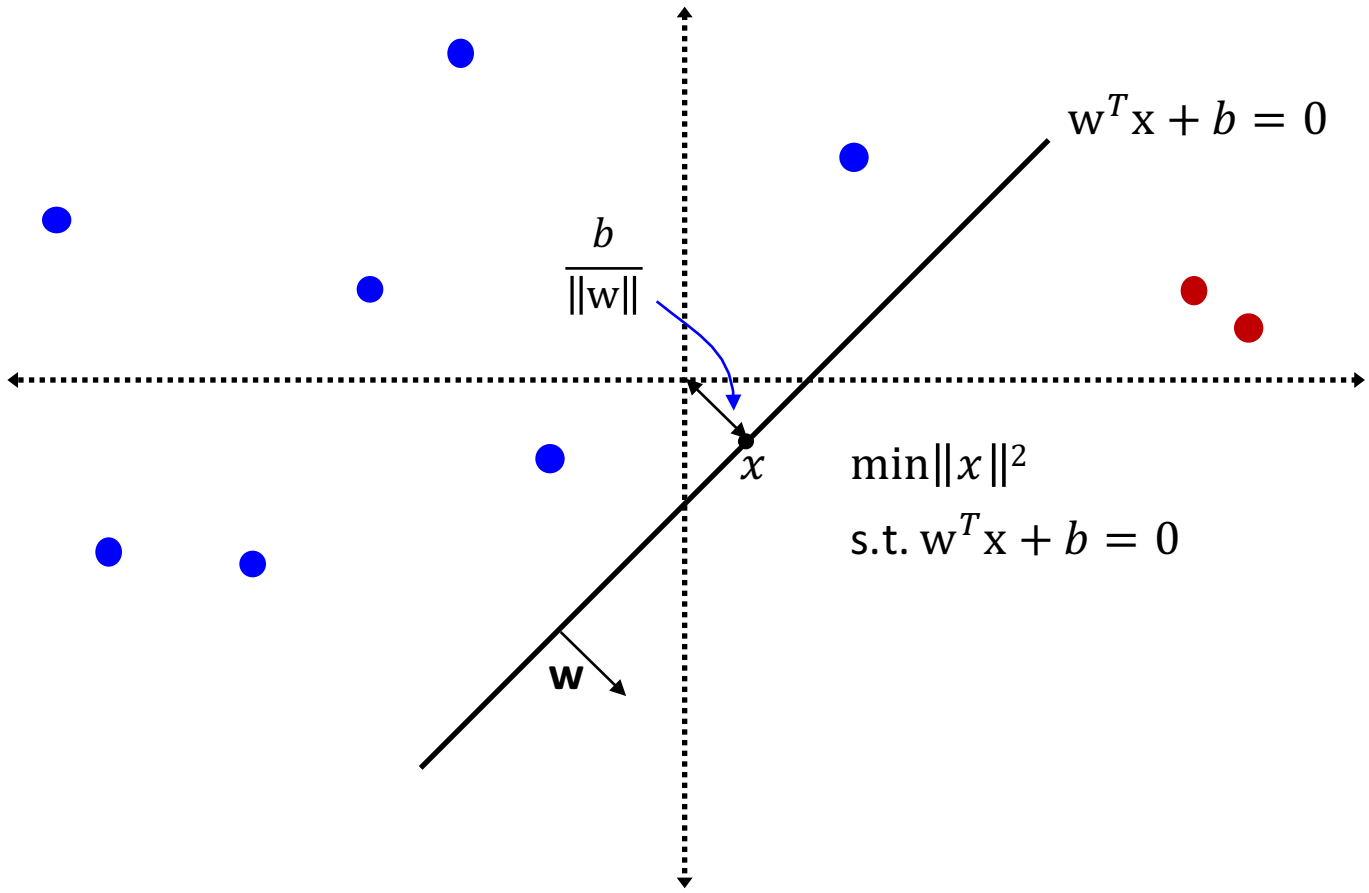
What is b?



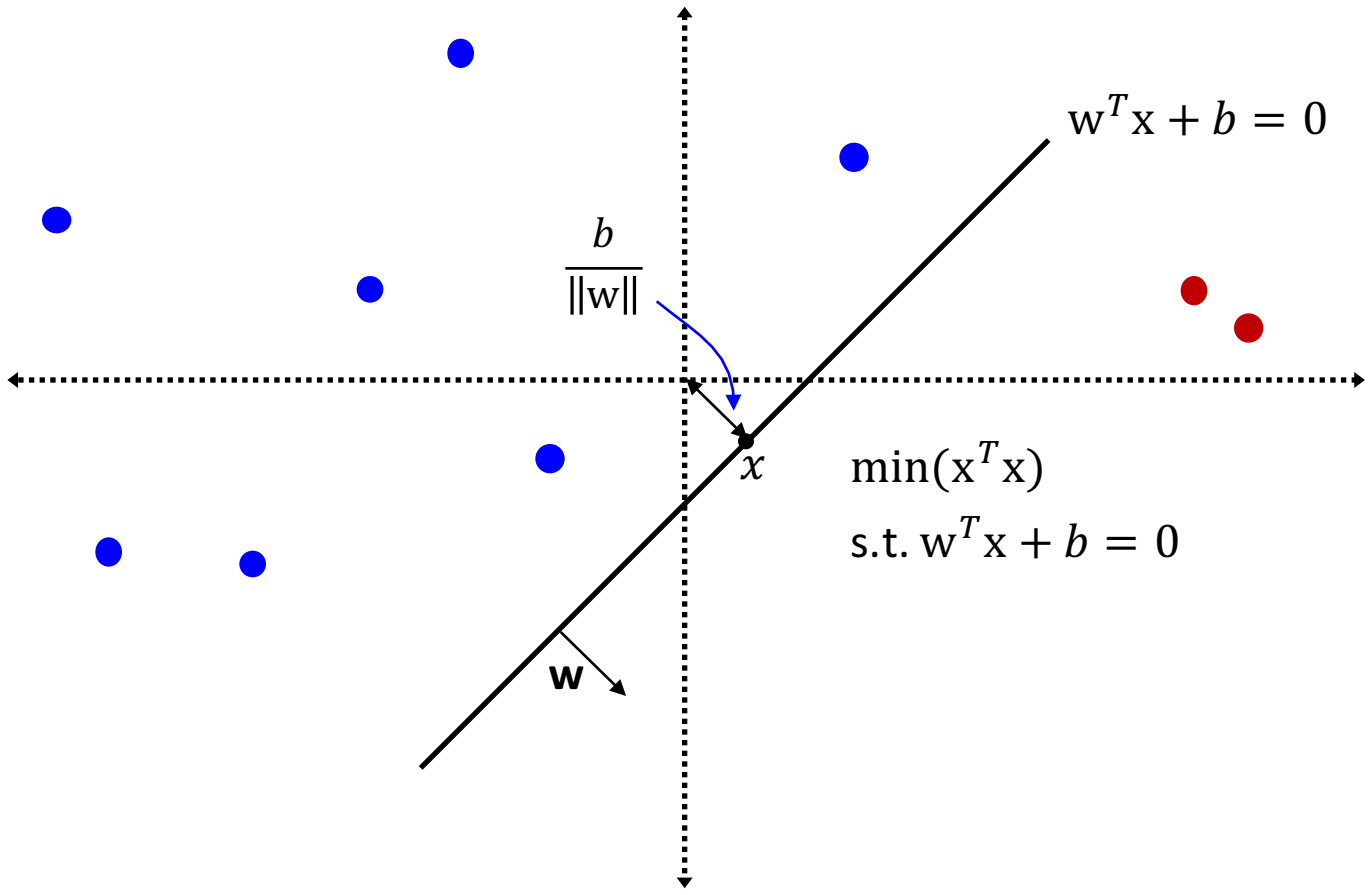
What is b?



What is b?



What is b?



What is b?

$$\min(\mathrm{x}^T\mathrm{x})$$

$$\text{s.t. } \mathrm{w}^T\mathrm{x} + b = 0$$

$$L(x,\lambda) = \mathrm{x}^T\mathrm{x} - \lambda\,(\mathrm{w}^T\mathrm{x} + b)$$

$$2\,\mathrm{x} - \lambda\, w = 0$$

Google: "The matrix cookbook"

$$\mathrm{x} = \frac{\lambda}{2} w$$

$$\mathrm{w}^T(\frac{\lambda}{2} w) + b = 0$$

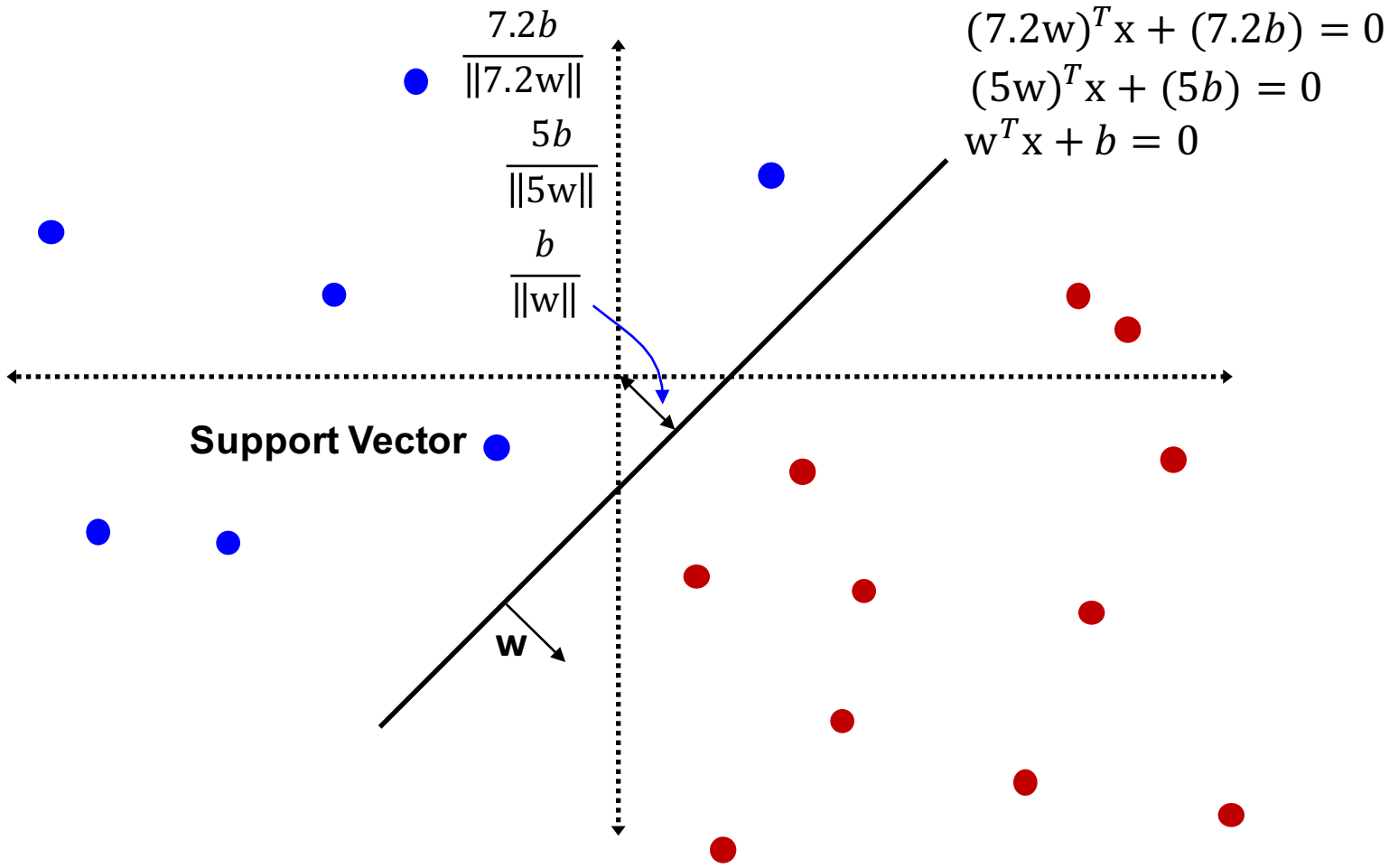
$$\frac{\lambda}{2}\mathrm{w}^T w + b = 0$$

$$\lambda = \frac{-2b}{\mathrm{w}^T w}$$

$$\mathrm{x} = \frac{\lambda}{2} w = \frac{-b}{\mathrm{w}^T w}\mathrm{w}$$

$$\|x\| = \sqrt{x^T x} = \sqrt{(\frac{-b}{\mathrm{w}^T w}\mathrm{w})^{\mathsf{T}}\,(\frac{-b}{\mathrm{w}^T w}\mathrm{w})} = \frac{b}{\mathrm{w}^T w}\sqrt{(\mathrm{w}^T w)} = \frac{b}{\sqrt{(\mathrm{w}^T w)}} = \frac{b}{\|w\|}$$

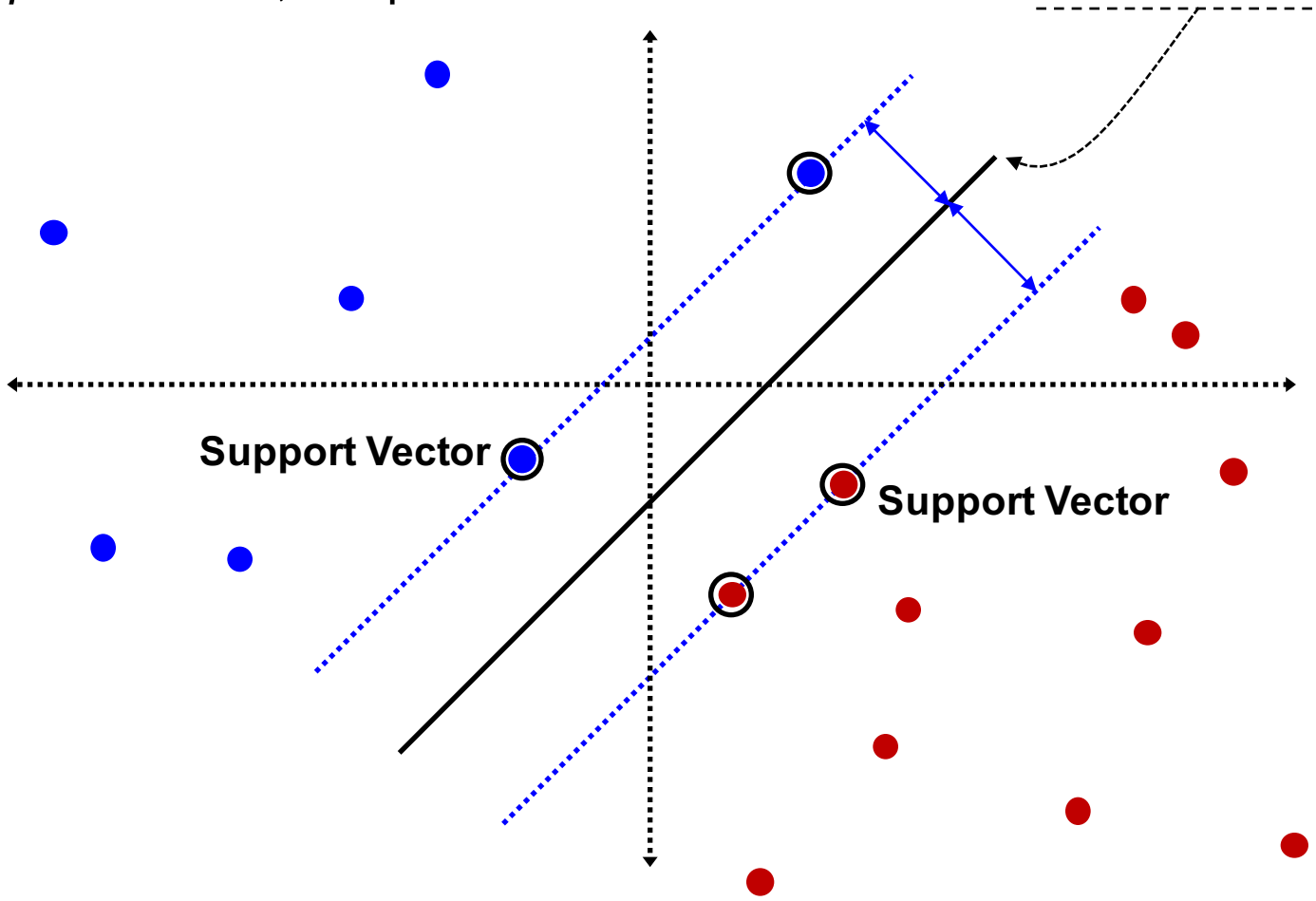
Scale is arbitrary



What about negative scale?

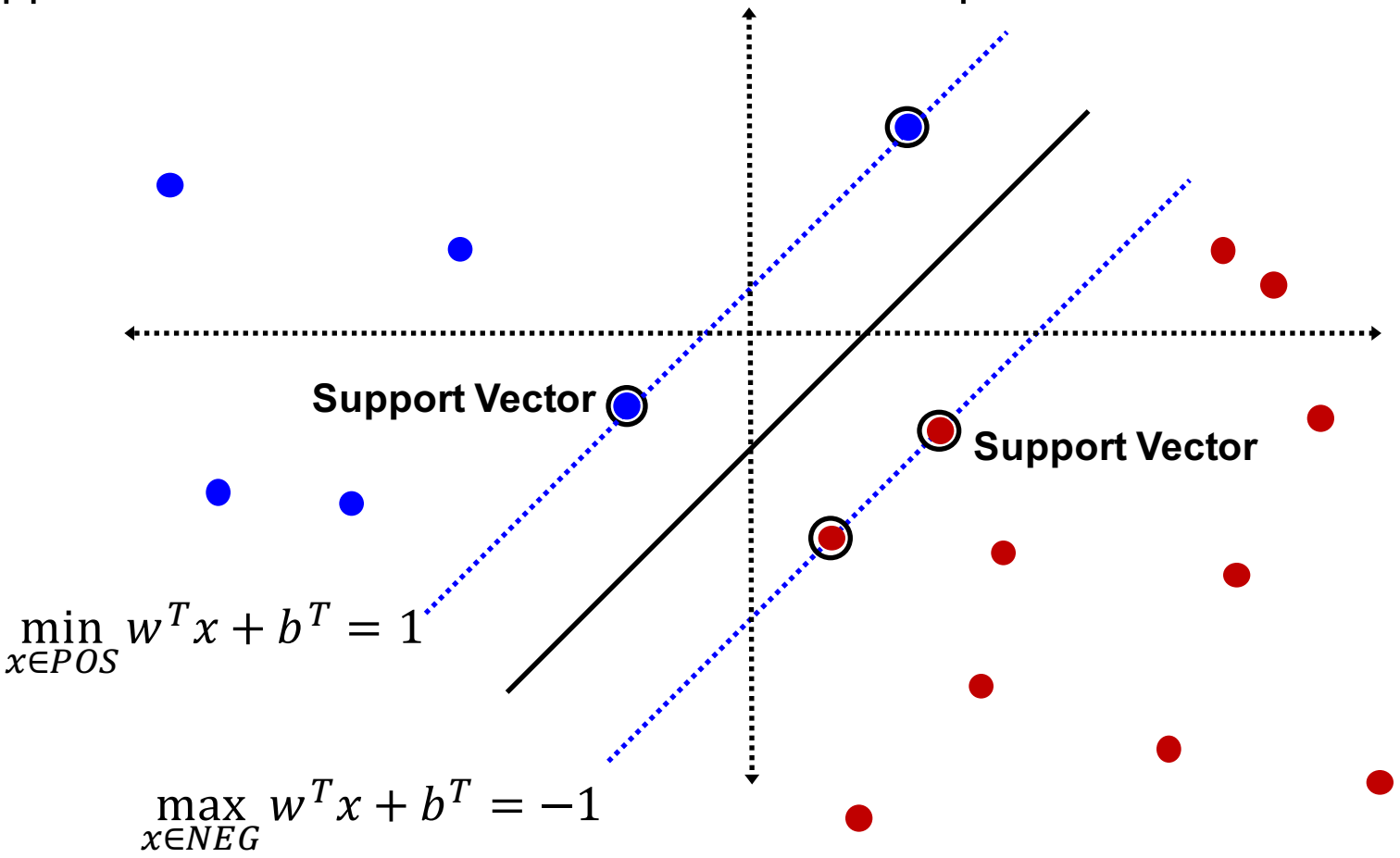
Which points influence the line?

The *support vectors*, the points that are closest to the *decision boundary*



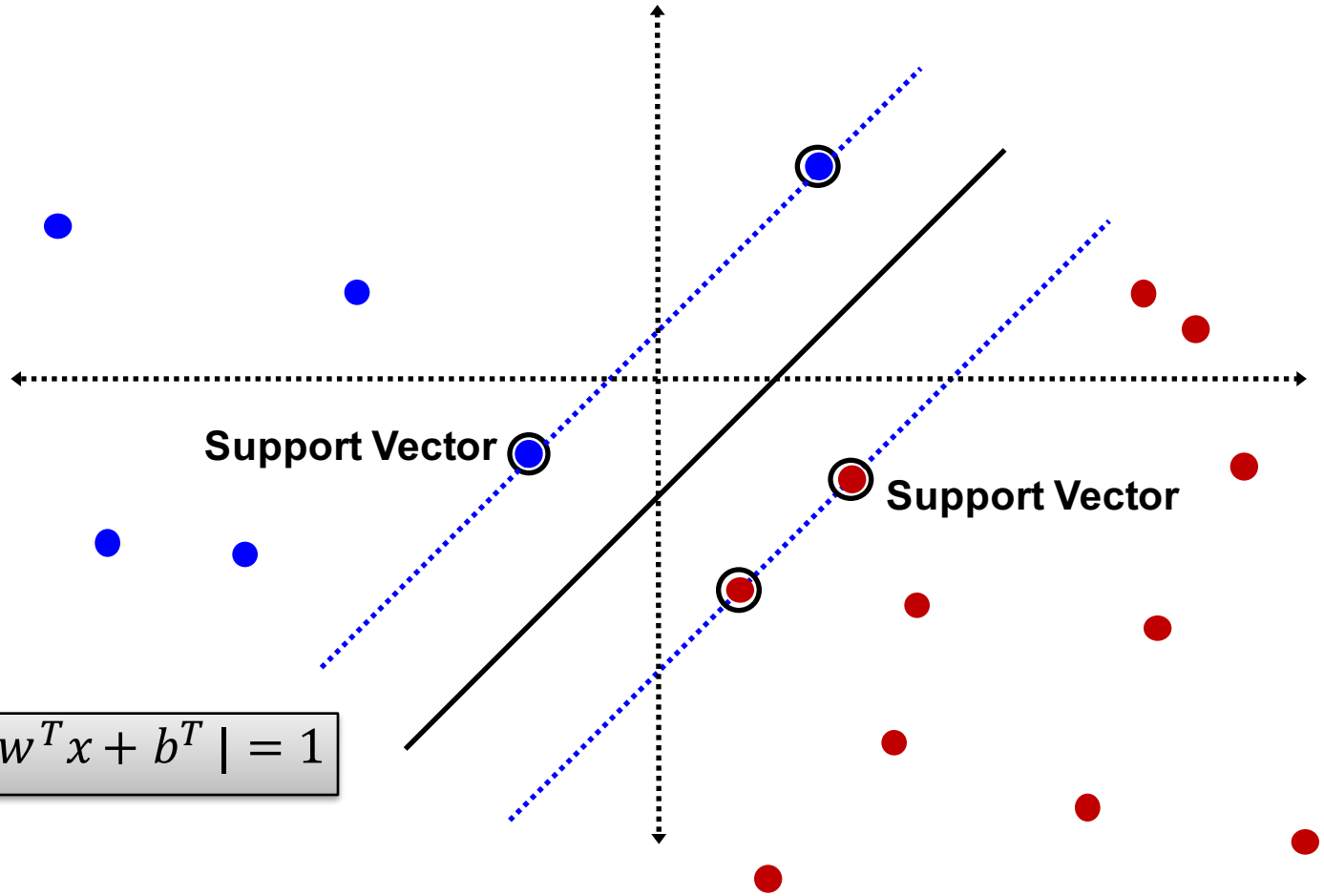
Fixing the scale

Since we have freedom to choose the scale, we choose it such that support vectors have a value of 1 in the line equation



Fixing the scale

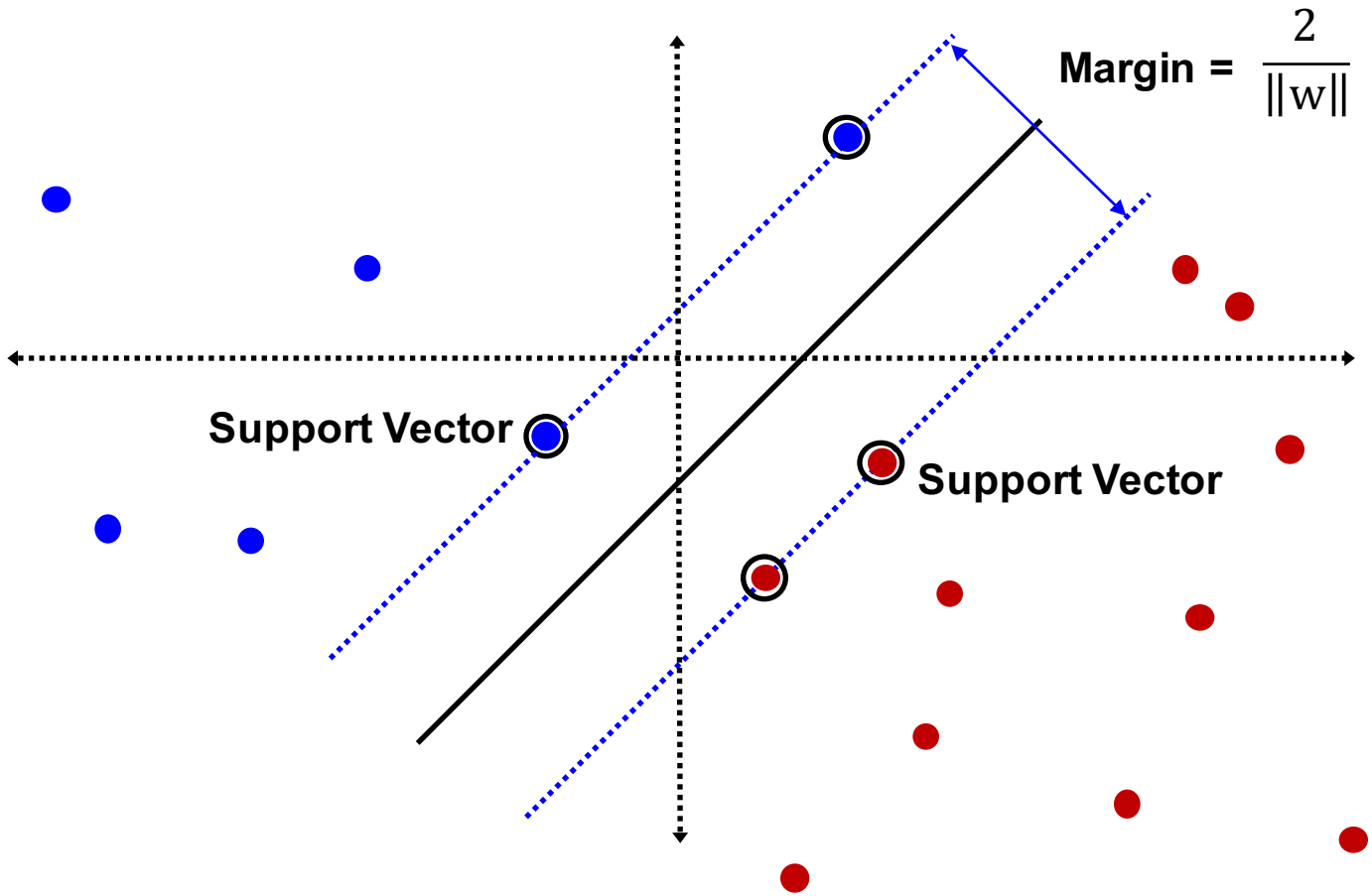
Equivilently – all points have values ≥ 1



$$\min_x |w^T x + b^T| = 1$$

Fixing the scale

In this case, we will show next, the margin is inversely proportional to $\|w\|$



Computing the distance

Let x_n be the nearest data point to the plane (w,b). How far is it?

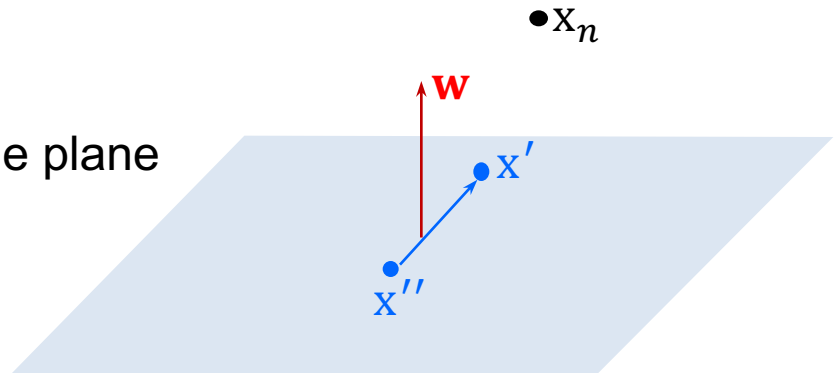
The distance between x_n and the plane $w^T x + b = 0$
where $|w^T x_n + b| = 1$

The vector w is $\perp$ to the plane:

As we've seen, take x' and x'' on the plane

$\mathrm{w}^{\mathrm{T}}\mathrm{x}' + b = 0$ and $\mathrm{w}^{\mathrm{T}}\mathrm{x}'' + b = 0$

$\Rightarrow \quad \mathrm{w}^{\mathrm{T}}(\mathrm{x}' - \mathrm{x}'') = 0$



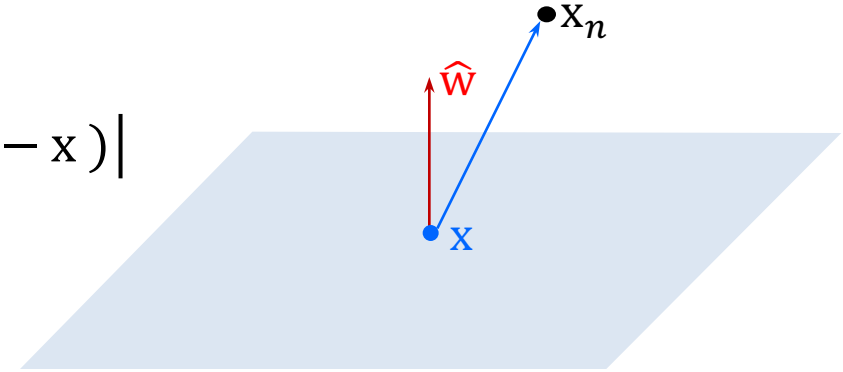
and the distance is …

Distance between x_n and the plane

Take any point x on the plane

Projection of $\mathrm{x}_n - \mathrm{x}$ on w

$$\widehat{\mathrm{w}} = \frac{\mathrm{w}}{\|\mathrm{w}\|} \quad \Rightarrow \quad \text{distance} = \left|\widehat{\mathrm{w}}^{\mathrm{T}}(\mathrm{x}_n - \mathrm{x})\right|$$



Distance =

$$\frac{1}{\|\mathrm{w}\|}\left|\mathrm{w}^{\mathrm{T}}\mathrm{x}_n - \mathrm{w}^{\mathrm{T}}\mathrm{x}\right| = \frac{1}{\|\mathrm{w}\|}\left|\mathrm{w}^{\mathrm{T}}x_n + b - \mathrm{w}^{\mathrm{T}}x - b\right|$$

The vanilla SVM optimization problem

Maximize $\frac{1}{\|\mathbf{w}\|}$

subject to $\min_{n=1,2,\ldots,N} \left|\mathbf{w}^{\mathrm{T}}\mathrm{x}_n + b\right| = 1$

The vanilla SVM optimization problem

Maximize $\frac{1}{\|\mathrm{w}\|}$

subject to $\min_{n=1,2,\ldots,N} \left|\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right| = 1$

Notice: $\left|\mathrm{w}^{\mathrm{T}}x_n + b\right| = y_n(\mathrm{w}^{\mathrm{T}}x_n + b)$

Maximize $\frac{1}{\|\mathrm{w}\|}$

subject to $y_n\left(\mathrm{w}^{\mathrm{T}}x_n + b\right) \geq 1$ for $n = 1,2,\ldots,N$

Minimize $\frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w}$

subject to $y_n\left(\mathrm{w}^{\mathrm{T}}x_n + b\right) \geq 1$ for $n = 1,2,\ldots,N$

Constrained optimization

Minimize $\frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w}$

subject to $y_n(w^T x_n + b) \geq 1$ for $n = 1,2,\ldots,N$

$\mathrm{w} \in \mathrm{R}^d, b \in \mathrm{R}$

This is a quadratic optimization problem subject to linear constraints a.k.a Quadratic Programming

x = quadprog(H,f,A,b) minimizes 1/2*x'*H*x + f'*x
subject to the restrictions A*x ≤ b.
A is a matrix of doubles, and b is a vector of doubles.

What is x? What is H? What if f?

H is positive semi-definite ➔ unique global solution

Constrained optimization

Minimize $\frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w}$

subject to $y_n(w^T x_n + b) \geq 1$ for $n = 1,2,\ldots,N$

$\mathrm{w} \in \mathrm{R}^d, b \in \mathrm{R}$

Lagrange? inequality constraints

Lagrange formulation

$$L(\mathrm{w}, b, \propto) = \frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w} - \sum_{n=1}^{N} \propto_n \left(y_n\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right) - 1\right)$$

Minimize w.r.t $\mathbf{w}$ and $\boldsymbol{b}$ and **Maximize** w.r.t each $\propto_{\boldsymbol{n}} \geq 0$

$\propto_{\boldsymbol{n}}$ plays the adversary:

If for all n, $y_n\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right) - 1 > 0$
then $\max_{\propto} L(\mathrm{w}, b, \propto)$ is obtained at $\propto = 0$ and is simply $\frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w}$

If for any n, $y_n\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right) - 1 < 0$
then $\propto_{\boldsymbol{n}} = \infty$ and $\max_{\alpha} L(\mathrm{w}, b, \propto)$ becomes ∞

Lagrange formulation

$$L(\mathrm{w}, b, \propto) = \frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w} - \sum_{n=1}^{N} \propto_n \left(y_n\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right) - 1\right)$$

Minimize w.r.t $\mathbf{w}$ and $\boldsymbol{b}$ and **Maximize** w.r.t each $\propto_{\boldsymbol{n}} \geq 0$

$$\nabla_{\mathrm{w}} L = \mathrm{w} - \sum_{n=1}^{N} \propto_n y_n \mathrm{x}_n = 0$$

$$\frac{\partial L}{\partial b} = -\sum_{n=1}^{N} \propto_n y_n = 0$$

Substituting…

$\mathrm{w} = \sum_{n=1}^{N} \propto_n y_n \mathrm{x}_n$ and employing $\sum_{n=1}^{N} \propto_n y_n = 0$

in the Lagrangian $L(\mathrm{w}, b, \propto) = \frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w} - \sum_{n=1}^{N} \propto_n \left(y_n\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right) - 1\right)$

we get $L(\propto) = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} y_n y_m \propto_n \propto_m \mathrm{x}_n^{\mathrm{T}} x_m + \sum_{n=1}^{N} \propto_n$

Maximize w.r.t to $\propto$ <u>subject to</u> $\propto_n \geq 0$ for $n = 1, \ldots, N$ and $\sum_{n=1}^{N} \propto_n y_n = 0$

The solution – quadratic programming

$$\min_{\propto} \frac{1}{2} \propto^{\mathrm{T}} \underbrace{\begin{bmatrix} y_1 y_1 \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_1 & y_1 y_2 \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_2 & \dots & y_1 y_N \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_N \\ y_2 y_1 \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_1 & y_2 y_2 \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_2 & \dots & y_2 y_N \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_N \\ \dots & \dots & \dots & \dots \\ y_N y_1 \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_1 & y_N y_2 \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_2 & \dots & y_N y_N \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_N \end{bmatrix}}_{\text{quadratic coefficients}} \propto + \underbrace{\left(-1^{\mathrm{T}}\right)}_{\text{linear}} \propto$$

subject to $\underbrace{y^{\mathrm{T}} \propto = 0}_{\text{linear constraint}}$

$$\underbrace{0}_{\text{lower bounds}} \leq \propto \leq \underbrace{\infty}_{\text{upper bounds}}$$

Minimize $\frac{1}{2} \mathrm{w}^{\mathrm{T}} \mathrm{w}$

subject to $y_n(w^T x_n + b) \geq 1$
$n = 1,2, \dots, N$

$\mathrm{w} \in \mathrm{R}^d, b \in \mathrm{R}$

QP hands us α

Solution: $\propto = \propto_1, \ldots, \propto_N$

$$\Rightarrow \quad \mathrm{w} = \sum_{n=1}^{N} \propto_n y_n \mathrm{x}_n$$

From the optimization:
for $n = 1, \ldots, N$

$$\propto_n \left(y_n\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right) - 1\right) = 0$$

Intuition – remember we maximize w.r.t $\propto$

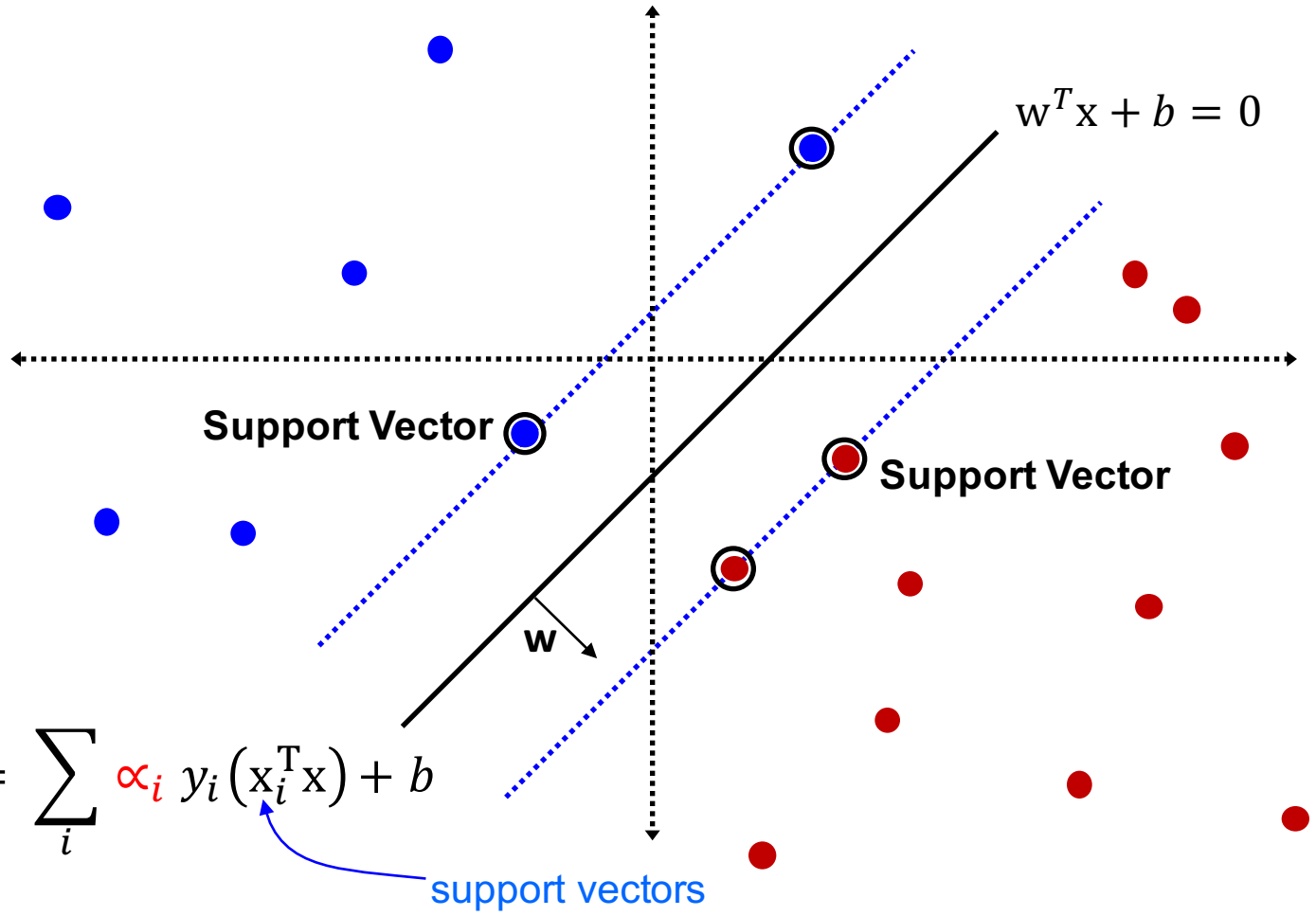
$$L(\mathrm{w}, b, \propto) = \frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w} - \sum_{n=1}^{N} \propto_n \left(y_n\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right) - 1\right)$$

$\propto_n > 0 \Rightarrow y_n\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_n + b\right) = 1 \Rightarrow \mathrm{x}_n$ is a **support vector**

How do we get b?

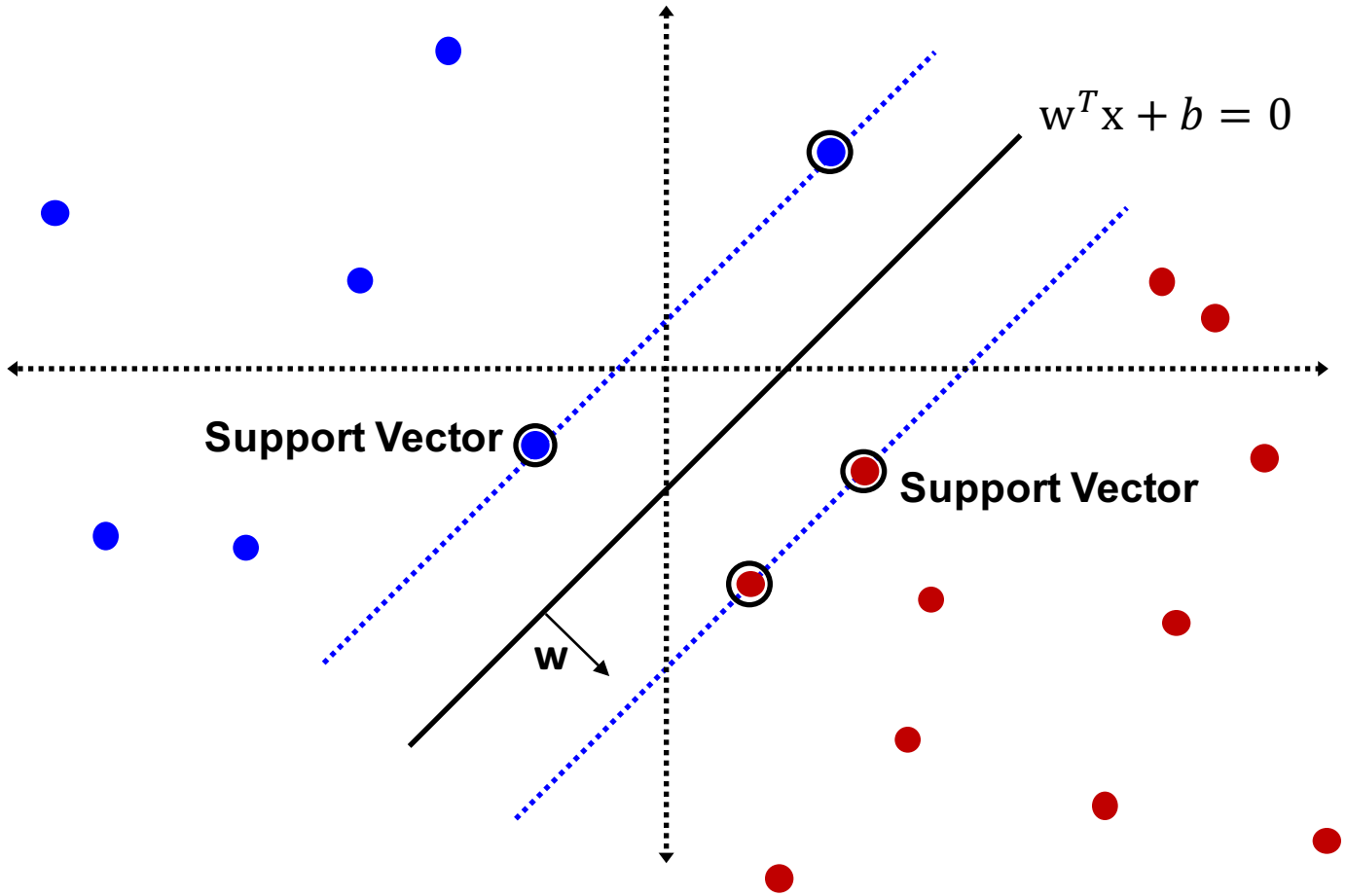
Support Vector Machine

linearly separable data



Leave One Out (LOO) error bound

How many LOO mistakes can we make in the worse case?

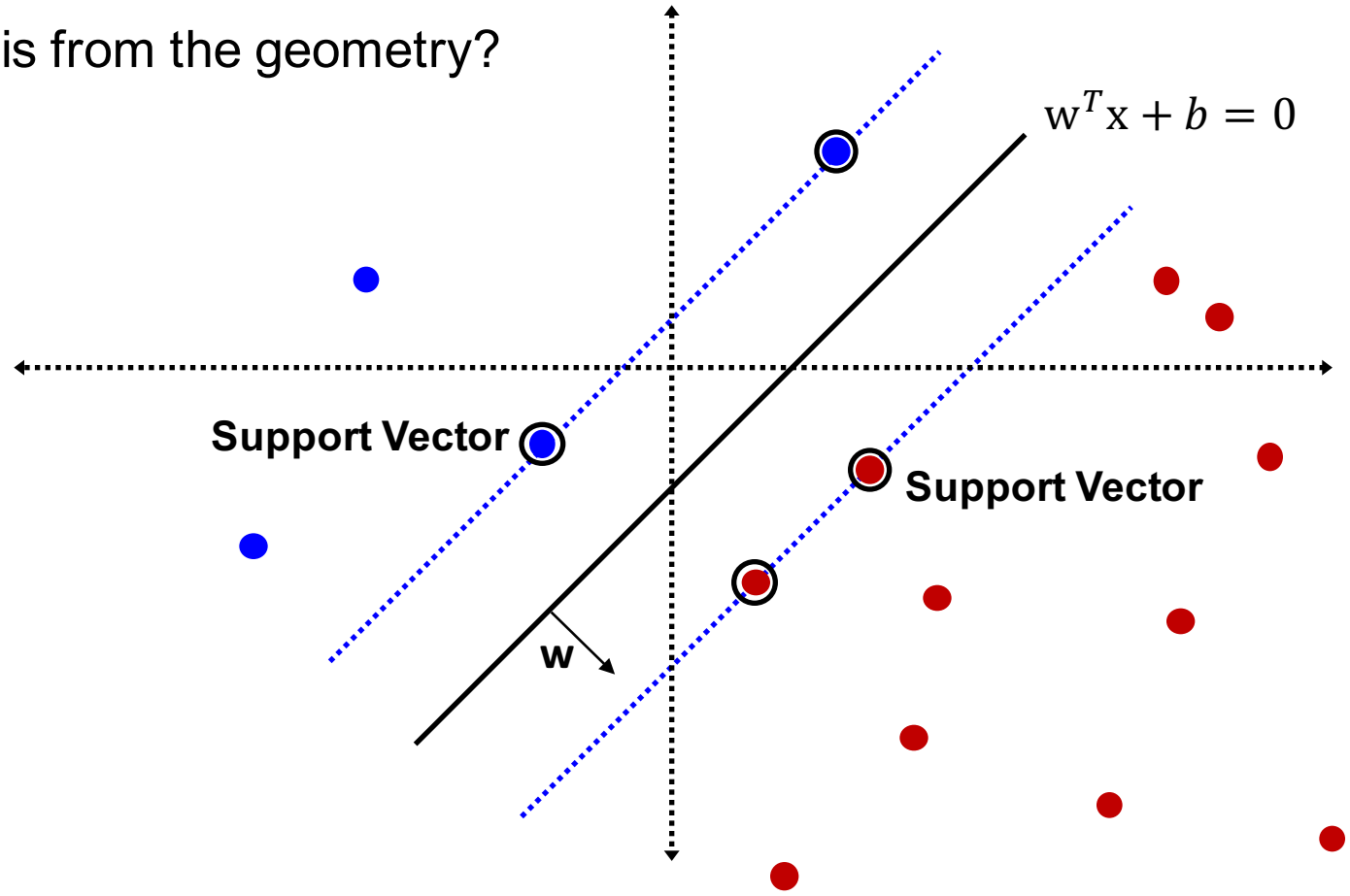


Leave One Out (LOO) error bound

How many LOO mistakes can we make in the worse case?

#SV, i.e., the LOO error is bounded by #SV/N

Can you see this from the geometry?



Leave One Out (LOO) error bound

How many LOO mistakes can we make in the worse case?

#SV, i.e., the LOO error is bounded by #SV/N

Can you see this from the primal QP?

Minimize $\frac{1}{2}\mathrm{w}^{\mathrm{T}}\mathrm{w}$

subject to $y_n(w^T x_n + b) \geq 1$

$n = 1,2,\ldots,N$

$\mathrm{w} \in \mathrm{R}^d, b \in \mathrm{R}$

Leave One Out (LOO) error bound

How many LOO mistakes can we make in the worse case?

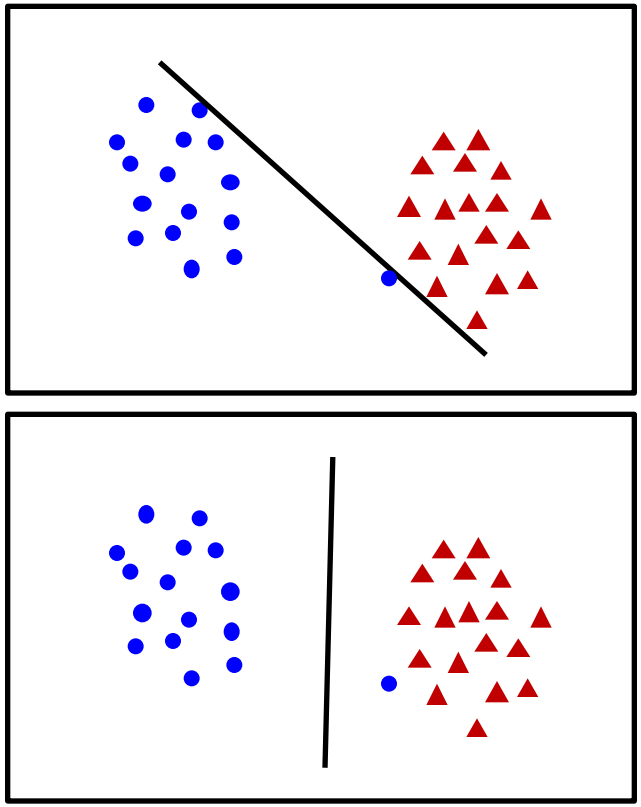
#SV, i.e., the LOO error is bounded by #SV/N

Can you see this from the dual QP?

$$\min_{\propto} \frac{1}{2} \propto^{\mathrm{T}} \begin{bmatrix} y_1 y_1 \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_1 & y_1 y_2 \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_2 & \cdots & y_1 y_N \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_N \\ y_2 y_1 \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_1 & y_2 y_2 \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_2 & \cdots & y_2 y_N \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_N \\ \cdots & \cdots & \cdots & \cdots \\ y_N y_1 \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_1 & y_N y_2 \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_2 & \cdots & y_N y_N \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_N \end{bmatrix} \propto + \left(-1^{\mathrm{T}}\right) \propto$$

subject to $y^{\mathrm{T}} \propto = 0 \qquad 0 \leq \propto \leq \infty$

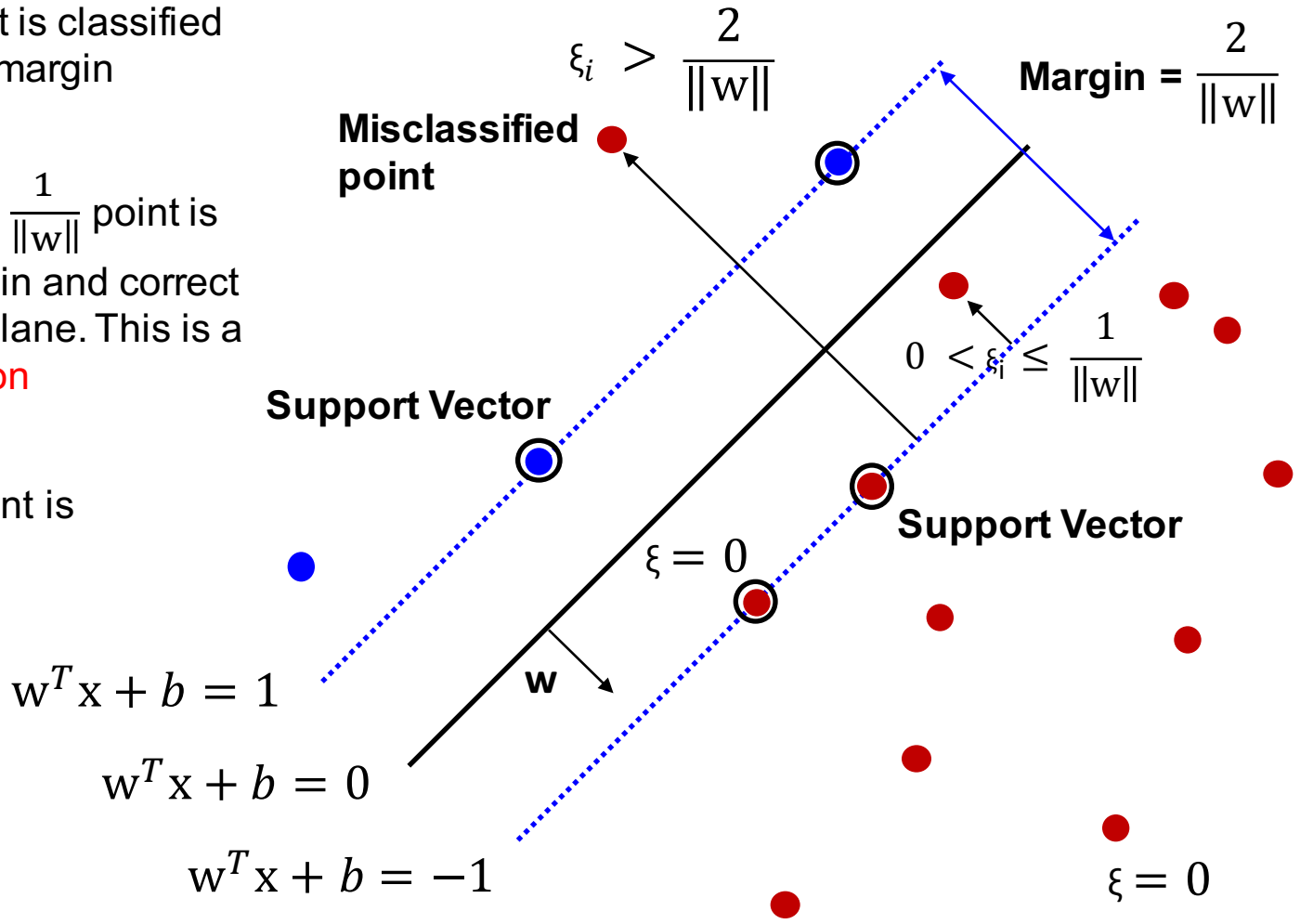
Linear separability again: What is the best w?



- linearly separated but very narrow margin
- possibly the large margin solution is better, even though one constraint is violated
- trade off between the margin and the number of mistakes on the training data

Introduce "slack" variables $\xi_i \geq 0$

- For $\xi = 0$ point is classified correctly with margin
- For $0 < \xi \leq \frac{1}{\|w\|}$ point is between margin and correct side of hyperplane. This is a margin violation
- for $\xi > \frac{1}{\|w\|}$ point is misclassified



“Soft” margin solution

The PRIMAL optimization problem becomes the following QP problem

$$\min_{\mathrm{w}\in\mathrm{R}^d,\xi_i>0} \|\mathrm{w}\|^2 + C\sum_{i}^{N}\xi_i$$

subject to

$$y_i\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_i + b\right) \geq 1 - \xi_i \text{ for } i = 1 \ldots N$$

- Every constraint can be satisfied if ξ_i is sufficiently large
- *C* is a regularization parameter:
  - small *C* allows constraints to be easily ignored → large margin
  - large *C* makes constraints hard to ignore → narrow margin
  - *C* = ∞ enforces all constraints: hard margin
- There is still a unique minimum
- Pain: we have now a parameter C
- Pain2: adding apples and oranges

The dual formulation

$$\min_{\propto} \frac{1}{2} \propto^{\mathrm{T}} \underbrace{\begin{bmatrix} y_1 y_1 \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_1 & y_1 y_2 \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_2 & \dots & y_1 y_N \mathrm{x}_1^{\mathrm{T}} \mathrm{x}_N \\ y_2 y_1 \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_1 & y_2 y_2 \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_2 & \dots & y_2 y_N \mathrm{x}_2^{\mathrm{T}} \mathrm{x}_N \\ \dots & \dots & \dots & \dots \\ y_N y_1 \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_1 & y_N y_2 \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_2 & \dots & y_N y_N \mathrm{x}_N^{\mathrm{T}} \mathrm{x}_N \end{bmatrix}}_{\text{quadratic coefficients}} \propto + \underbrace{\left(-1^{\mathrm{T}}\right)}_{\text{linear}} \propto$$

subject to $\underbrace{y^{\mathrm{T}} \propto = 0}_{\text{linear constraint}}$

$$\underbrace{0}_{\text{lower bounds}} \leq \propto \leq \underbrace{C}_{\text{upper bounds}}$$

Another point of view for SVM

SVM was formulated as a constrained optimization problem over w and positive ξ

$$\min_{\mathrm{w}\in\mathrm{R}^d,\xi_i>0} \|\mathrm{w}\|^2 + C\sum_i^N \xi_i \quad \text{subject to} \quad y_i\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_i + b\right) \geq 1 - \xi_i \text{ for } i = 1 \ldots N$$

The constraint $y_i\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_i + b\right) \geq 1 - \xi_i$ can be written as

$$y_i f(\mathrm{x}_i) \geq 1 - \xi_i$$

since $\xi_i \geq 0$ and since we want to "pay" as little as possible

$$\xi_i = \max(0, 1 - y_i f(\mathrm{x}_i))$$

Substituting, we get the unconstrained optimization problem over w alone

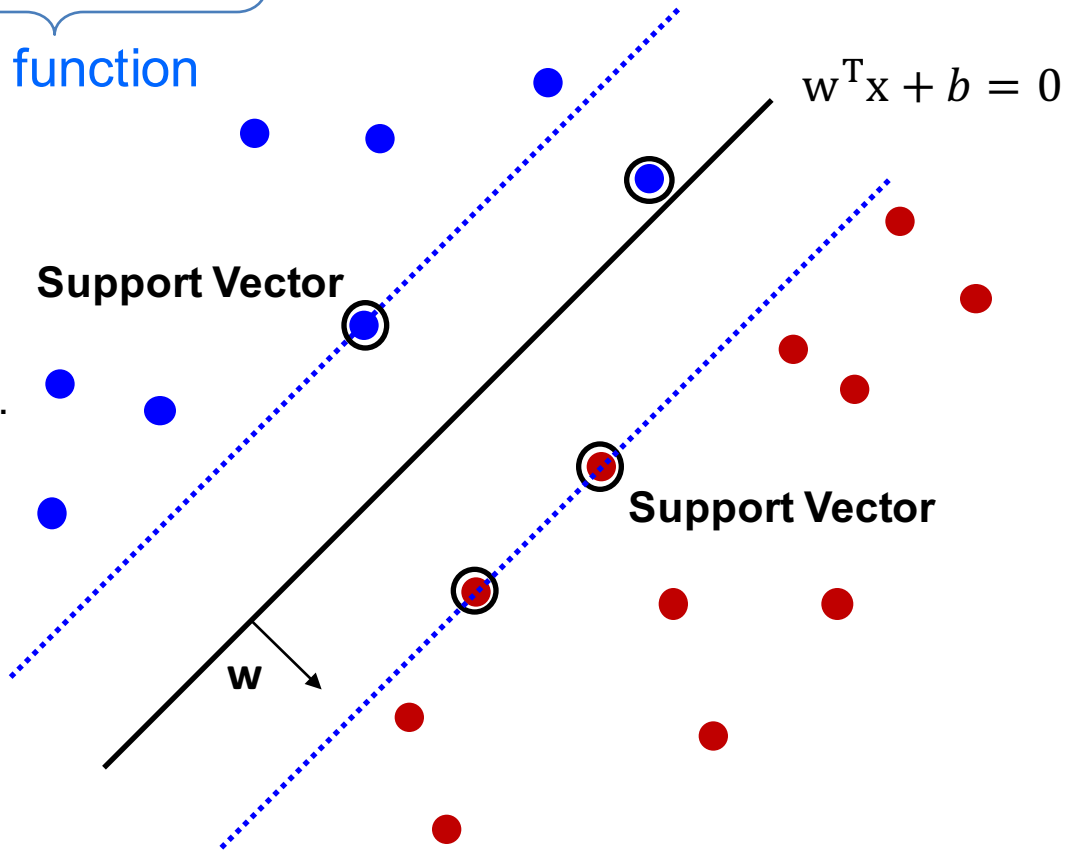
$$\min_{\mathrm{w}\in\mathrm{R}^d} \underbrace{\|\mathrm{w}\|^2}_{\text{regularization}} + C\sum_i^N \underbrace{\max(0, 1 - y_i f(\mathrm{x}_i))}_{\text{loss function}}$$

Loss function

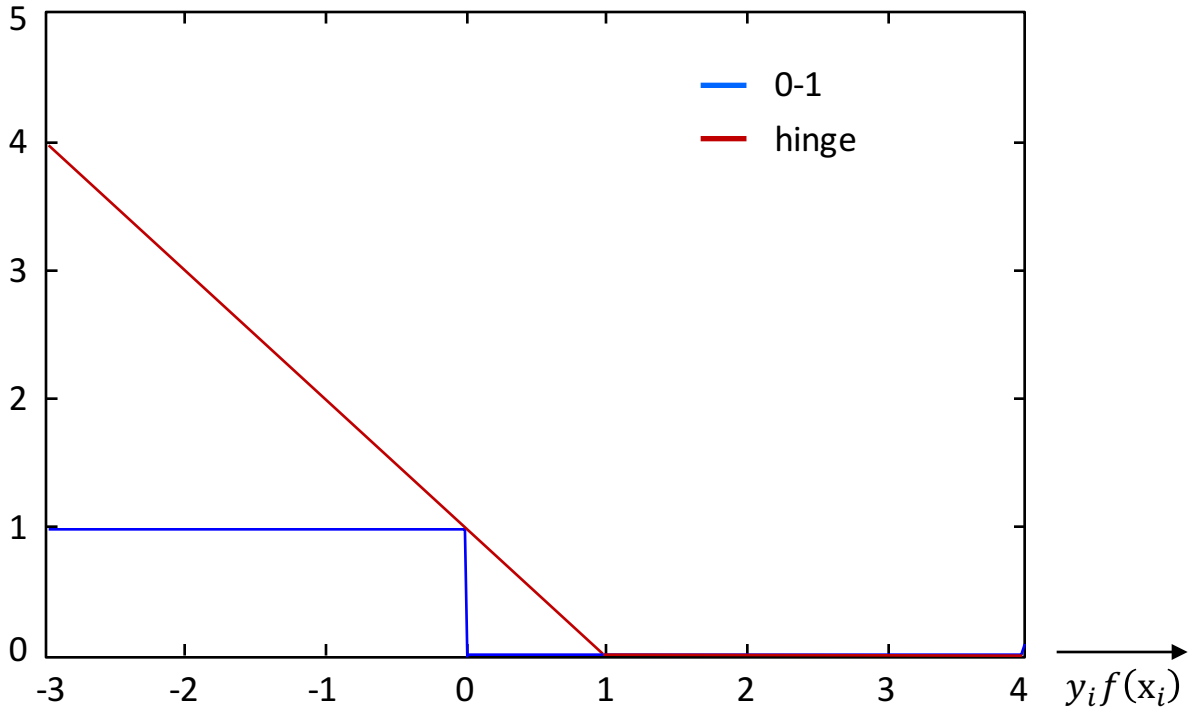
$$\min_{w \in R^d} \|w\|^2 + C \sum_{i}^{N} \underbrace{\max(0, 1 - y_i f(x_i))}_{\text{loss function}}$$

Points are in two categories:

1. $y_i f(x_i) \geq 1$
   Point is on or outside margin.
   No contribution to loss
2. $y_i f(x_i) < 1$
   Point violates margin constraint. Contributes to loss.



Hinge loss



- SVM uses the hinge loss $\max(0,1 - y_i f(\mathrm{x}_i))$
- an approximation to the 0-1 loss

What about multi-class SVMs?

- Unfortunately, there is no "definitive" multi-class SVM formulation
- In practice, we have to obtain a multi-class SVM by combining multiple two-class SVMs
- One vs. others
  - Traning: learn an SVM for each class vs. the others
  - Testing: apply each SVM to test example and assign to it the class of the SVM that returns the highest decision value
- One vs. one
  - Training: learn an SVM for each pair of classes
  - Testing: each learned SVM "votes" for a class to assign to the test example

SVM: Practical Advice

- Avoid MATLAB's implementation

- Use http://www.csie.ntu.edu.tw/~cjlin/**libsvm**/
- Or http://www.csie.ntu.edu.tw/~cjlin/**liblinear**/ when you know you want linear SVM

- Play with C (?)

- Avoid unbalanced classes (?)

SVM – sketch derivation

- Since $\mathrm{w}^{\mathrm{T}}\mathrm{x} + b = 0$ and $c\left(\mathrm{w}^{\mathrm{T}}\mathrm{x} + b\right) = 0$ define the same plane, we have the freedom to choose the normalization of w
- Choose normalization such that $\mathrm{w}^{\mathrm{T}}\mathrm{x}_{+} + b = +1$ and $\mathrm{w}^{\mathrm{T}}\mathrm{x}_{-} + b = -1$ for the positive and negative support vectors respectively
- Then the margin is given by

$$\frac{\mathrm{w}}{\|\mathrm{w}\|}.(\mathrm{x}_{+} - \mathrm{x}_{-}) = \frac{\mathrm{w}^{\mathrm{T}}(\mathrm{x}_{+} - \mathrm{x}_{-})}{\|\mathrm{w}\|} = \frac{2}{\|\mathrm{w}\|}$$

SVM – Optimization

- Learning the SVM can be formulated as an optimization:

$$\max_{\mathrm{w}} \frac{2}{\|\mathrm{w}\|} \quad \text{subject to } \mathrm{w}^{\mathrm{T}}\mathrm{x}_i + b \begin{matrix} \geq 1 \\ \leq -1 \end{matrix} \begin{matrix} \text{if } y_i = +1 \\ \text{if } y_i = -1 \end{matrix} \quad \text{for } i = 1 \ldots N$$

- Or equivalently

$$\min_{\mathrm{w}} \|\mathrm{w}\|^2 \quad \text{subject to } y_i\left(\mathrm{w}^{\mathrm{T}}\mathrm{x}_i + b\right) \geq 1 \text{ for } i = 1 \ldots N$$

- This is a quadratic optimization problem subject to linear constraints and there is a unique minimum

SVM – sketch derivation

- Since $\mathrm{w}^{\mathrm{T}}\mathrm{x} + b = 0$ and $c\left(\mathrm{w}^{\mathrm{T}}\mathrm{x} + b\right) = 0$ define the same plane, we have the freedom to choose the normalization of w
- Choose normalization such that $\mathrm{w}^{\mathrm{T}}\mathrm{x}_{+} + b = +1$ and $\mathrm{w}^{\mathrm{T}}\mathrm{x}_{-} + b = -1$ for the positive and negative support vectors respectively
- Then the margin is given by

$$\frac{\mathrm{w}}{\|\mathrm{w}\|}.(\mathrm{x}_{+} - \mathrm{x}_{-}) = \frac{\mathrm{w}^{\mathrm{T}}(\mathrm{x}_{+} - \mathrm{x}_{-})}{\|\mathrm{w}\|} = \frac{2}{\|\mathrm{w}\|}$$

Finding **w** with large margin

Let x_n be the nearest data point to the plane $\mathrm{w}^{\mathrm{T}}\mathrm{x} = 0$. How far is it?

2 preliminary technicalities:

1. **Normalize** w:
$$\left|\mathrm{w}^{\mathrm{T}}\mathrm{x}_n\right| = 1$$

2. **Pull out** ω_0:
$\mathrm{w} = (\omega_1, \ldots, \omega_d)$ apart from b

   The plane is now $\mathrm{w}^{\mathrm{T}}\mathrm{x} + b = 0$ (no x_0)