Support Vector Machines (I): Overview and Linear SVM

LING 572
Advanced Statistical Techniques for NLP
February 13 2020

Why another learning method?

- Based on some “beautifully simple” ideas (Schölkopf, 1998)
  - Maximum margin decision hyperplane

- Member of class of kernel models (vs. attribute models)

- Empirically successful:
  - Performs well on many practical applications
  - Robust to noisy data, complex distributions
  - Natural extensions to semi-supervised learning

Kernel methods

- Family of “pattern analysis” algorithms
- Best known member is the Support Vector Machine (SVM)
- Maps instances into higher dimensional feature space efficiently
- Applicable to:
  - Classification
  - Regression
  - Clustering
  - ….

History of SVM

- Linear classifier: 1962
  - Use a hyperplane to separate examples
  - Choose the hyperplane that maximizes the minimal margin

- Non-linear SVMs:
  - Kernel trick: 1992

History of SVM (cont'd)

- Soft margin: 1995
  - To deal with non-separable data or noise

- Semi-supervised variants:
  - Transductive SVM: 1998
  - Laplacian SVMs: 2006

Main ideas

- Use a hyperplane to separate the examples.
- Among all the hyperplanes wx+b=0, choose the one with the maximum margin.
- Maximizing the margin is the same as minimizing ||w|| subject to some constraints.

Main ideas (cont'd)

- For data sets that are not linearly separable, map the data to a higher dimensional space and separate them there by a hyperplane.
- The Kernel trick allows the mapping to be "done" efficiently.
- Soft margin deals with noise and/or inseparable data sets.

Papers

- (Manning et al., 2008)
  - Chapter 15

- (Collins and Duffy, 2001): tree kernel

Outline

- Linear SVM
  - Maximizing the margin
  - Soft margin
- Nonlinear SVM
  - Kernel trick
- A case study
- Handling multi-class problems

Inner product vs. dot product

Dot product

The dot product of two vectors $x=(x_1, ..., x_n)$ and $z=(z_1, ..., z_n)$

is defined as $x \cdot z = \sum_i x_i z_i$

$$||x|| = \sqrt{\sum_i x_i^2} = \sqrt{x \cdot x}$$

Inner product

- An inner product is a generalization of the dot product.

$$||x|| = \sqrt{< x, x >}$$

- A function that satisfies the following properties:

$$< u + v, w > = < u, w > + < v, w >$$

$$< cu, v > = c < u, v >$$

$$< u, v > = < v, u >$$

$$< u, u > \geq 0 \text{ and } < u, u > = 0 \text{ iff } u = 0$$

Some examples

$$< x, z >= \sum_i c_i x_i z_i$$

$$< (a, b), (c, d) >= (a + b)(c + d) + (a - b)(c - d)$$

$$< f, g >= \int f(x)g(x)dx \text{ where } f, g\text{: } [a, b] \to R$$

Linear SVM

The setting

- Input:
  - x is a vector of real-valued feature values
- Output: y in Y , Y = {-1, +1}
- Training set: $S = \{(x_1, y_1), \ldots, (x_i, y_i)\}$
- Goal: Find a function y = f(x) that fits the data:

  f: X ➔ R

➔ Warning: x_i is used in two ways in this lecture.

Notation

x_i has two meanings

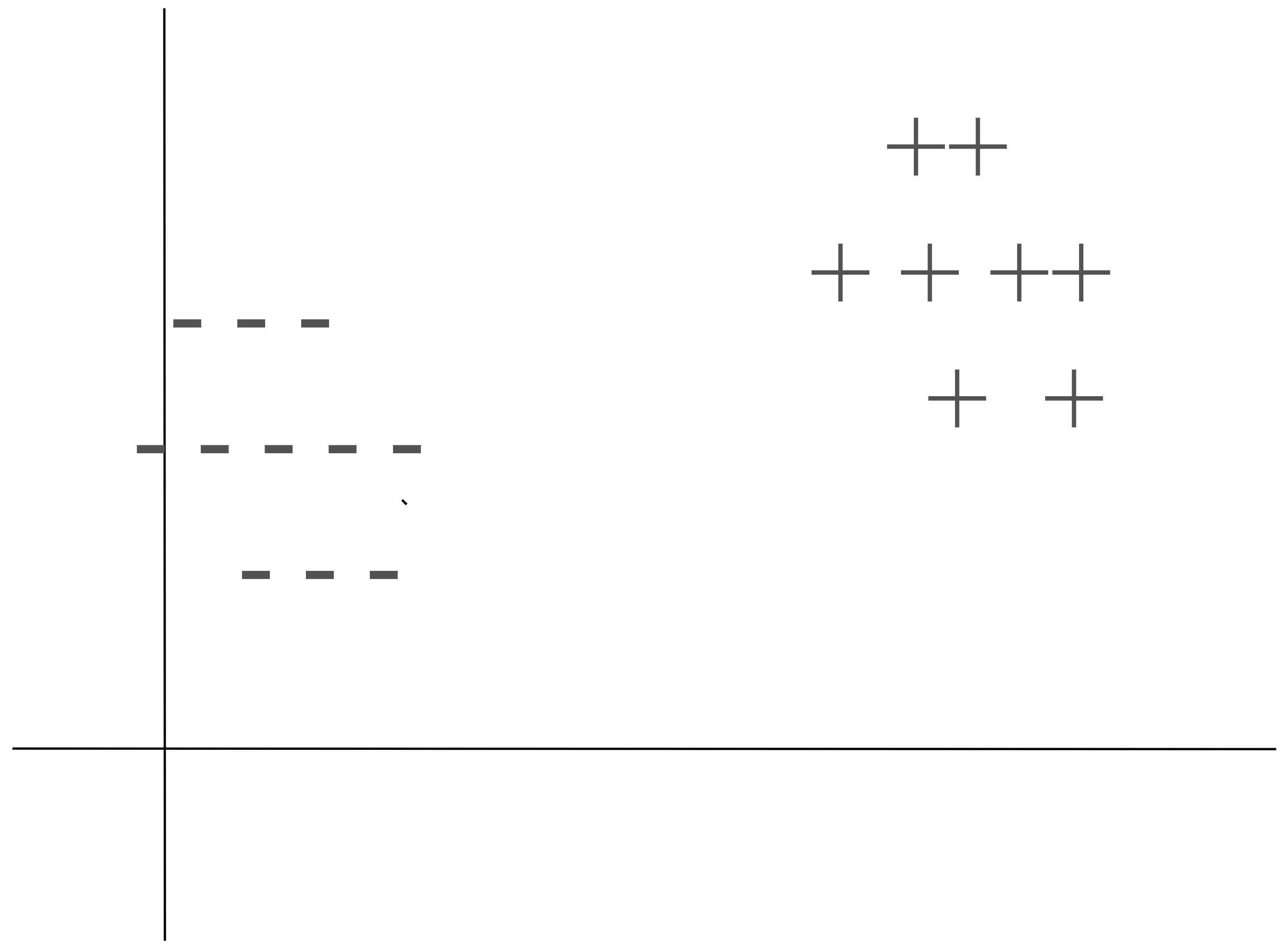
- $\vec{x_i}$: It is a vector, representing the i-th training instance.
- x_i: It is the i-th element of a vector $\vec{x}$

x, w, and z are vectors.

b is a real number

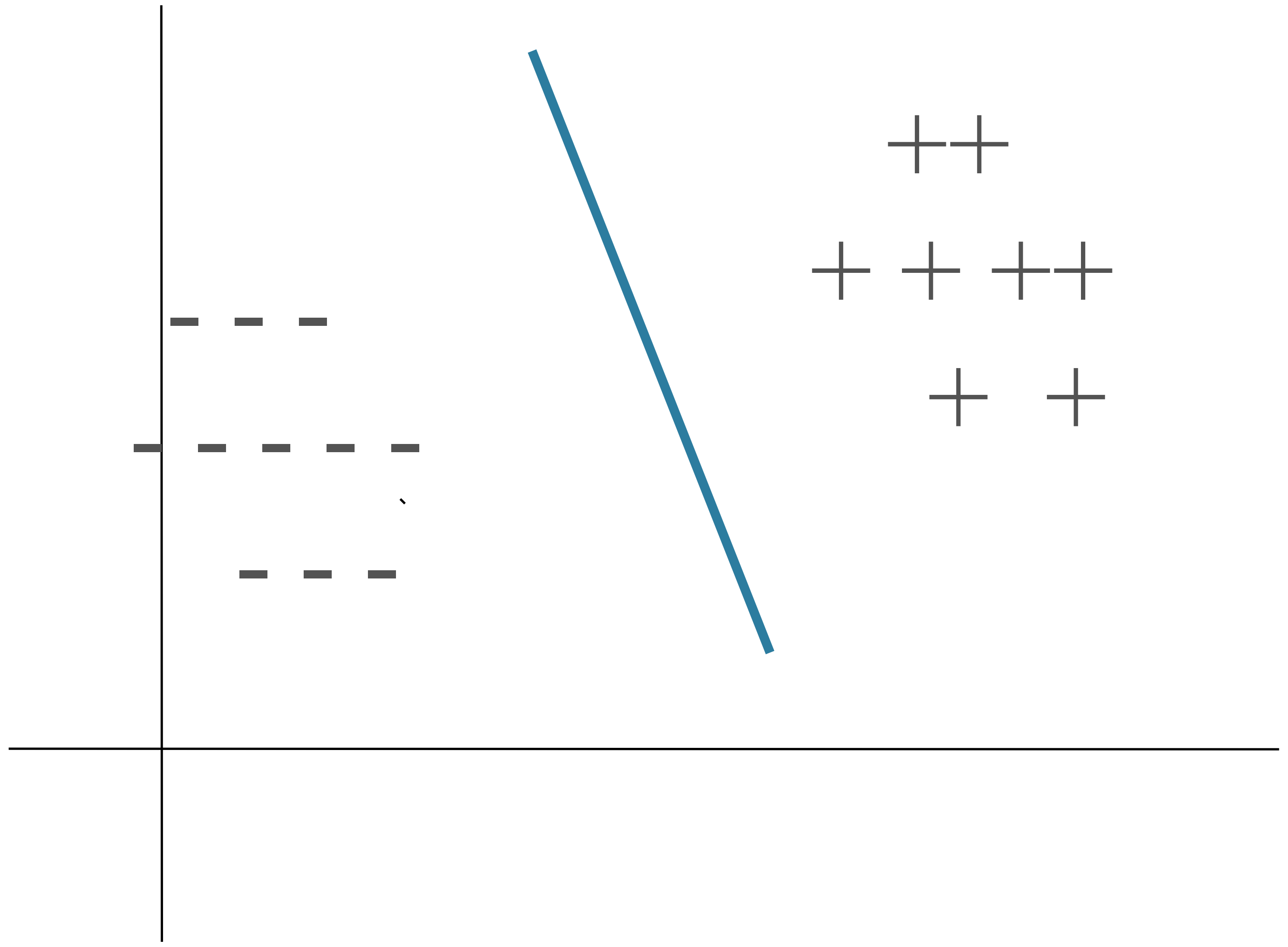
Linear classifier

- Consider the 2-D data
- +: Class +1
- -: Class -1
- Can we draw a line that separates the two classes?



Linear classifier

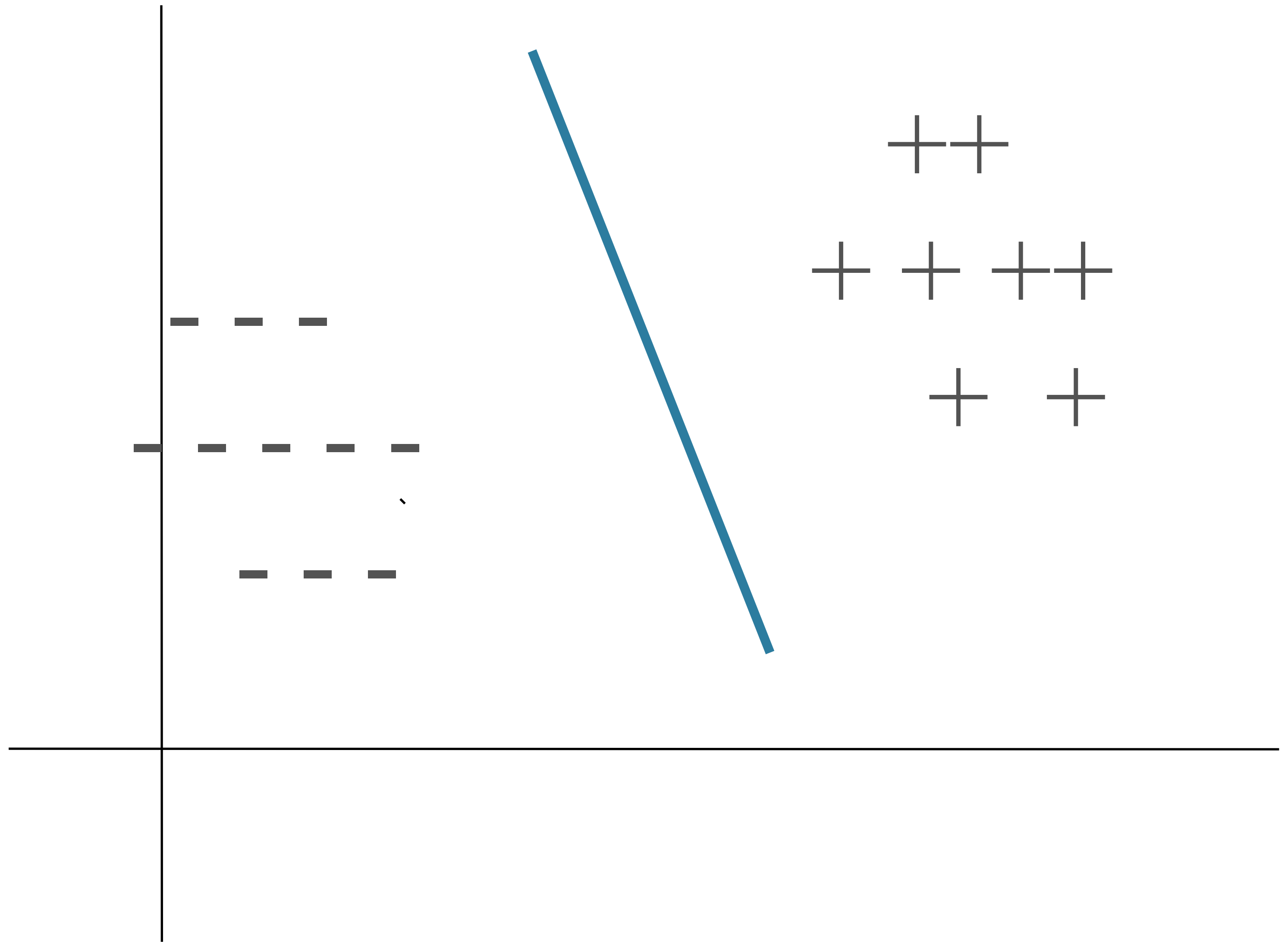
- Consider the 2-D data
- +: Class +1
- -: Class -1
- Can we draw a line that separates the two classes?
- Yes!
  - We have a linear classifier/separator; >2D→   hyperplane



Linear classifier

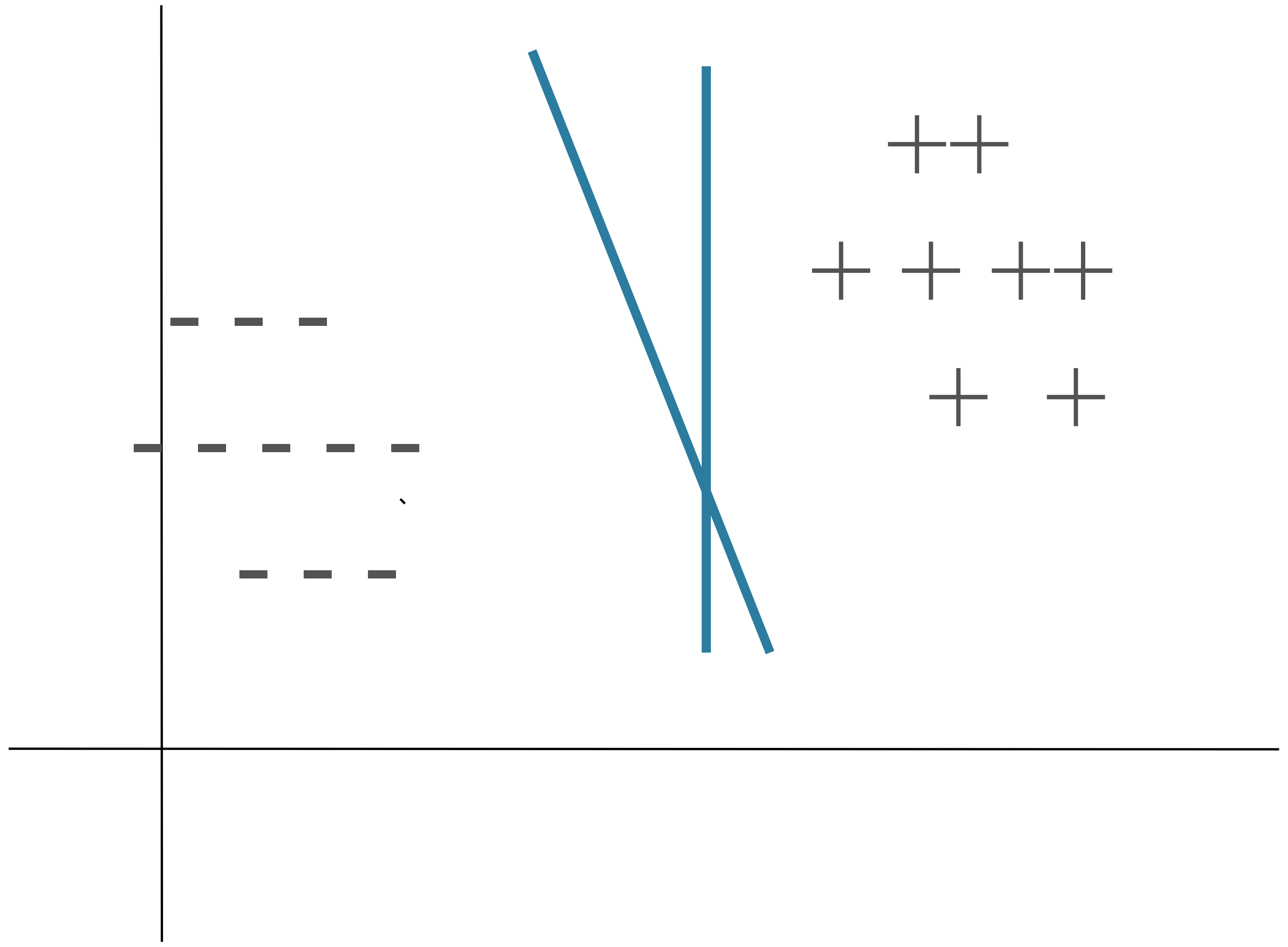
- Consider the 2-D data
- +: Class +1
- -: Class -1
- Can we draw a line that

  separates the two classes?
- Yes!
  - We have a linear classifier/separator; >2D→ hyperplane
- Is this the only such separator?



Linear classifier

- Consider the 2-D data below
- +: Class +1
- -: Class -1
- Can we draw a line that separates the two classes?

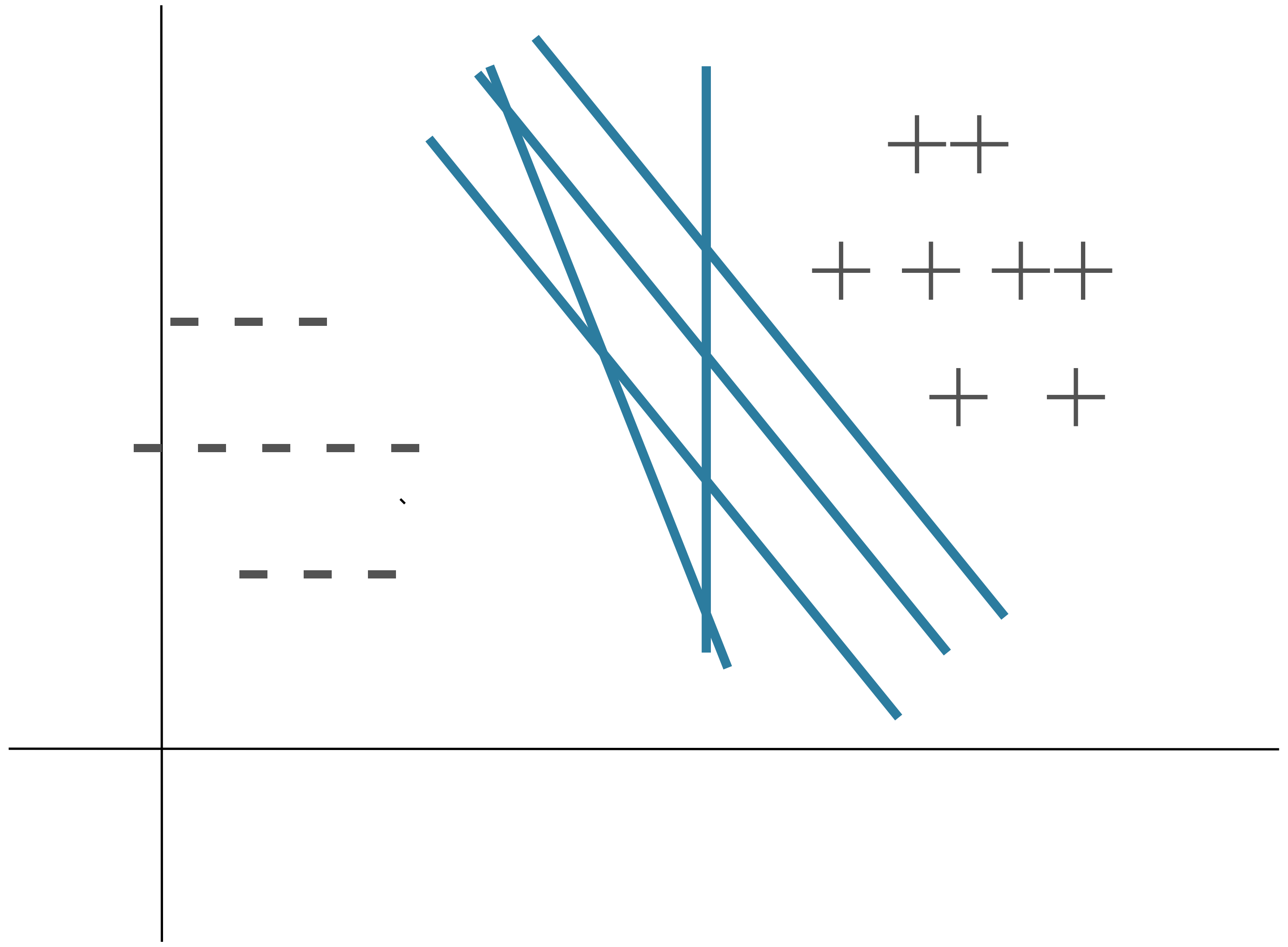


- Yes!
  - We have a linear classifier/separator; >2D→ hyperplane
- Is this the only such separator?
  - No

Linear classifier

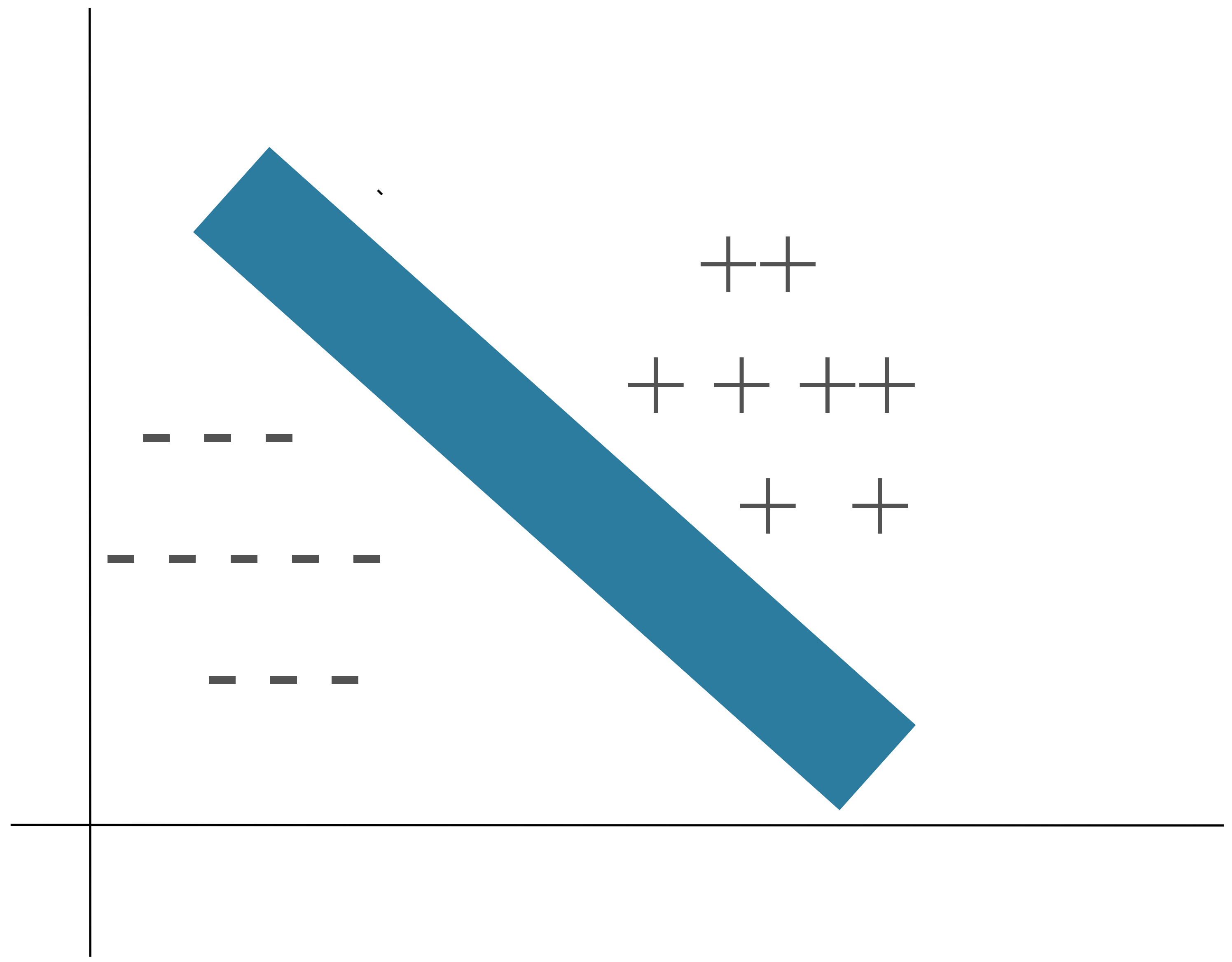
- Consider the 2-D data
- +: Class +1
- -: Class -1
- Can we draw a line that

  separates the two classes?
- Yes!
  - We have a linear classifier/separator; >2D→ hyperplane
- Is this the only such separator?
  - No
- Which is the best?



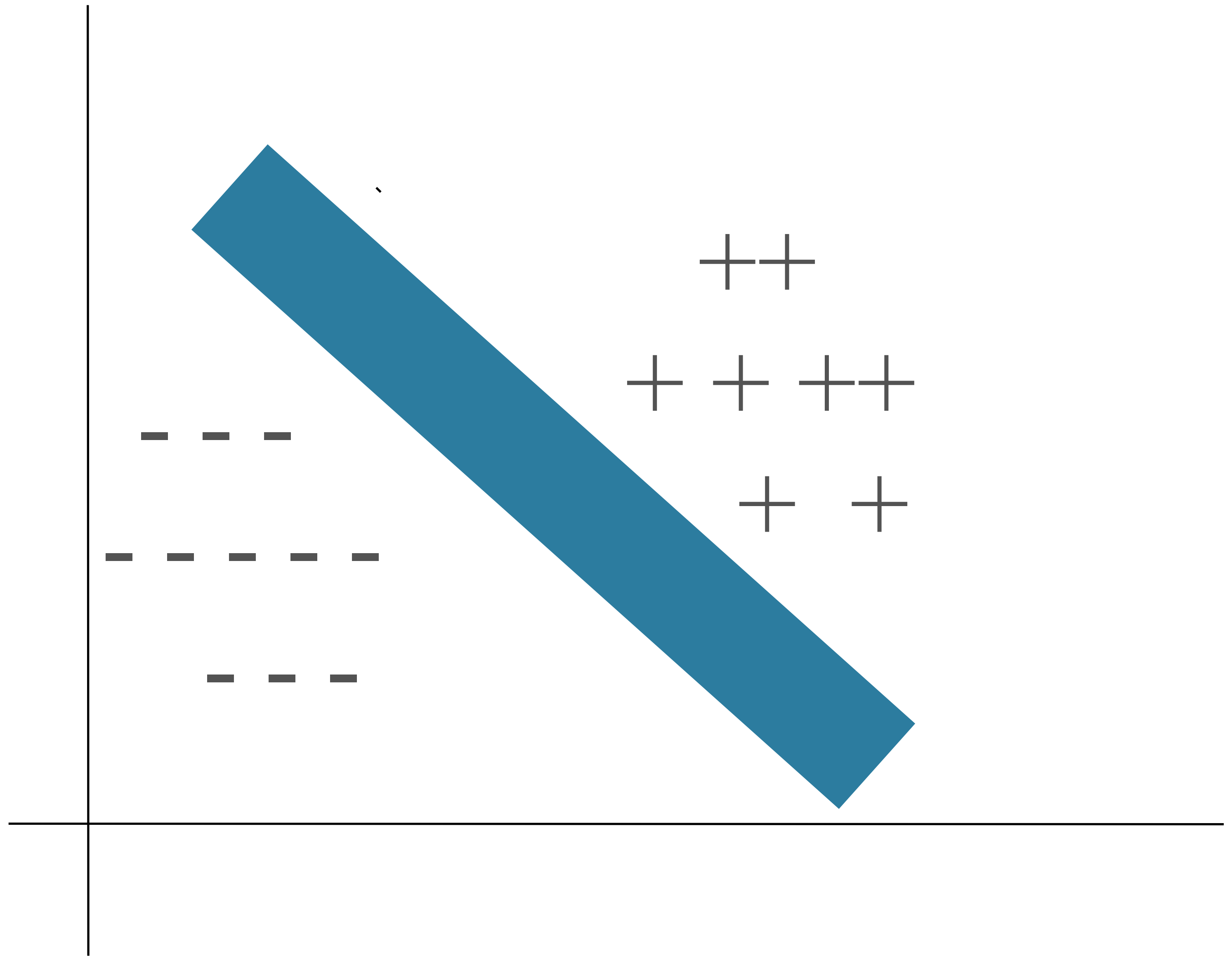
Maximum Margin Classifier

- What's best classifier?



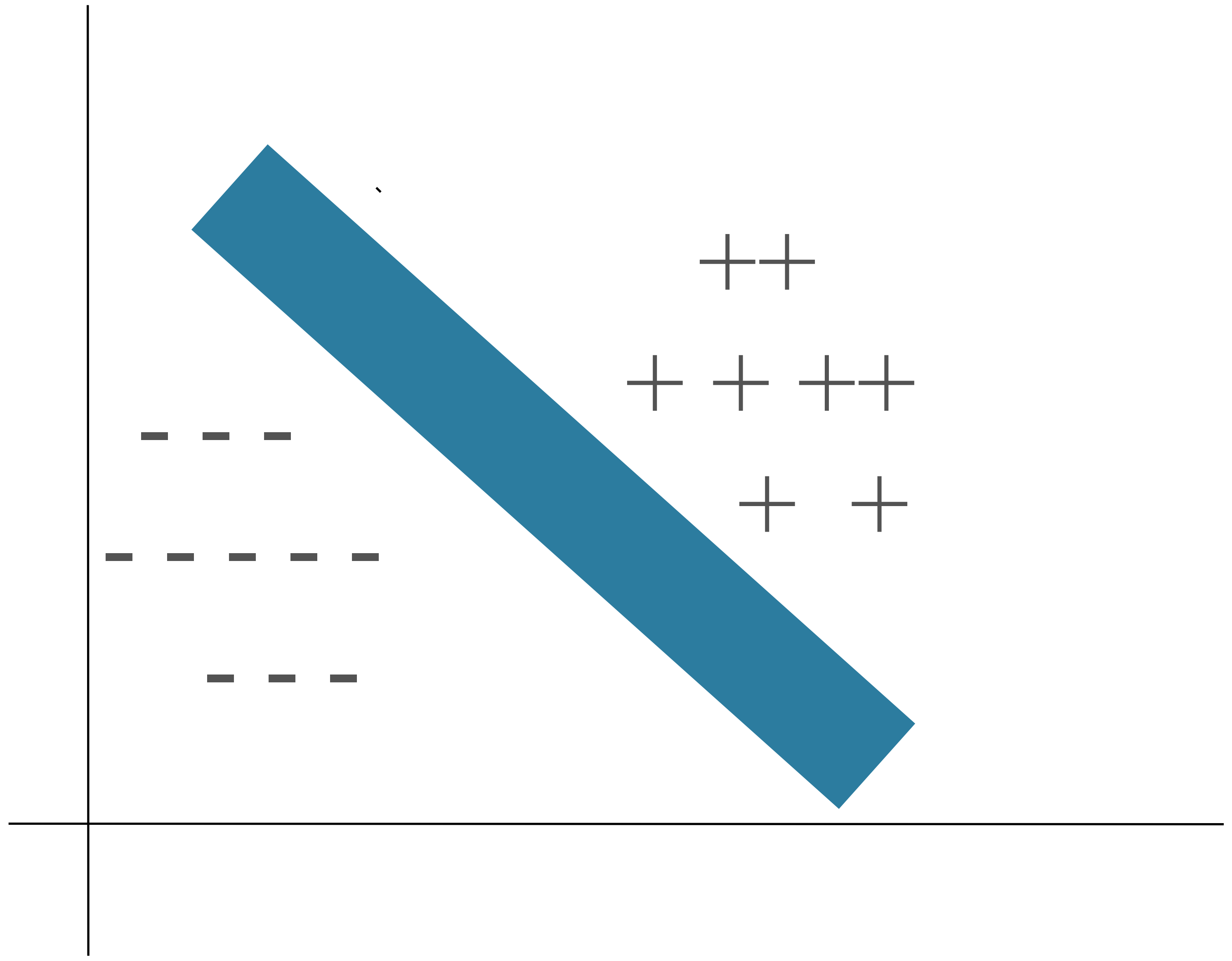
Maximum Margin Classifier

- What's best classifier?
  - Maximum margin
    - Biggest distance between decision boundary and closest examples
- Why is this better?
  - Intuition:



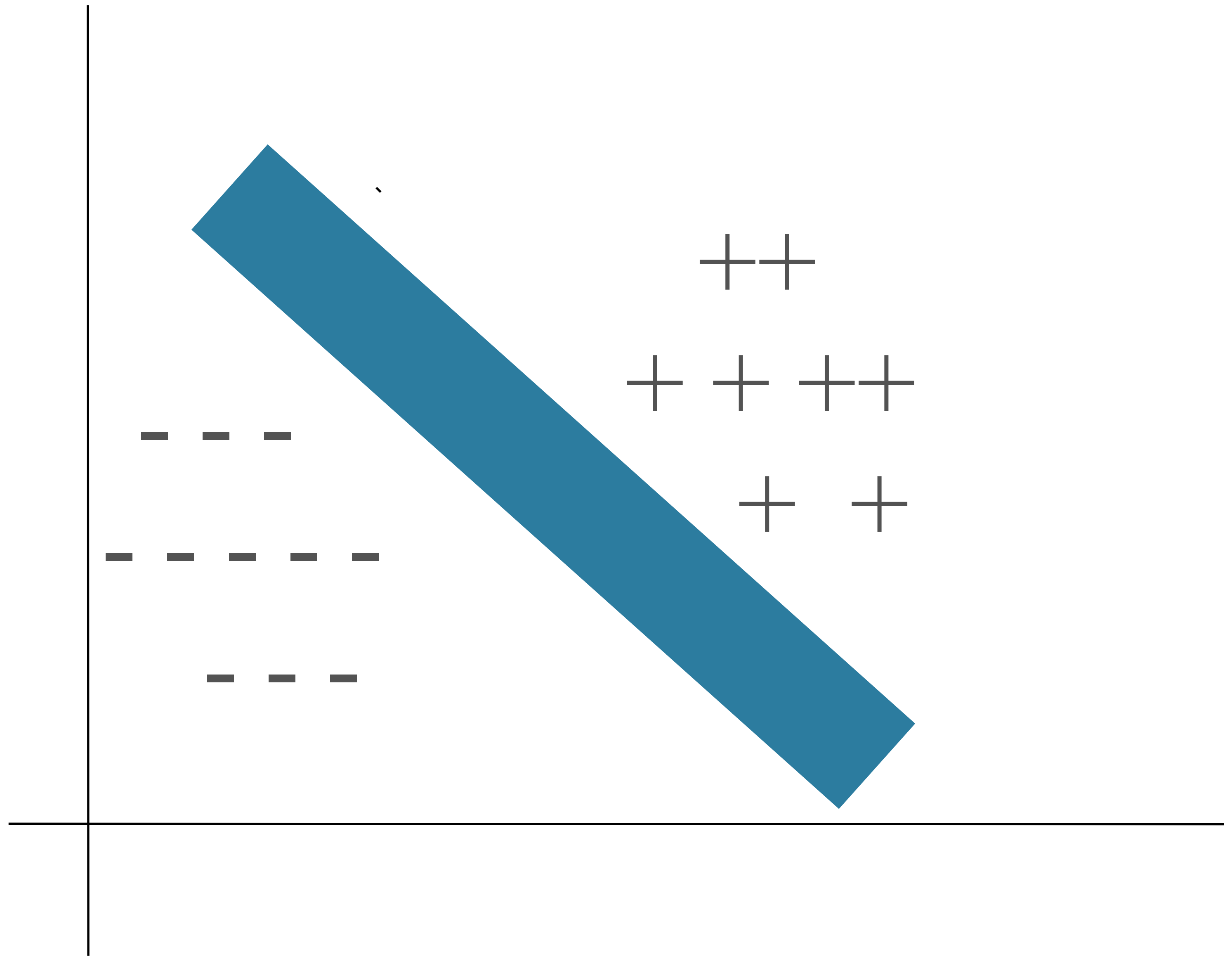
Maximum Margin Classifier

- What's best classifier?
  - Maximum margin
    - Biggest distance between decision boundary and closest examples
- Why is this better?
  - Intuition:
    - Which instances are we most sure of?
      - Furthest from boundary
    - Least sure of?
      - Closest
    - Create boundary with most 'room' for error in attributes



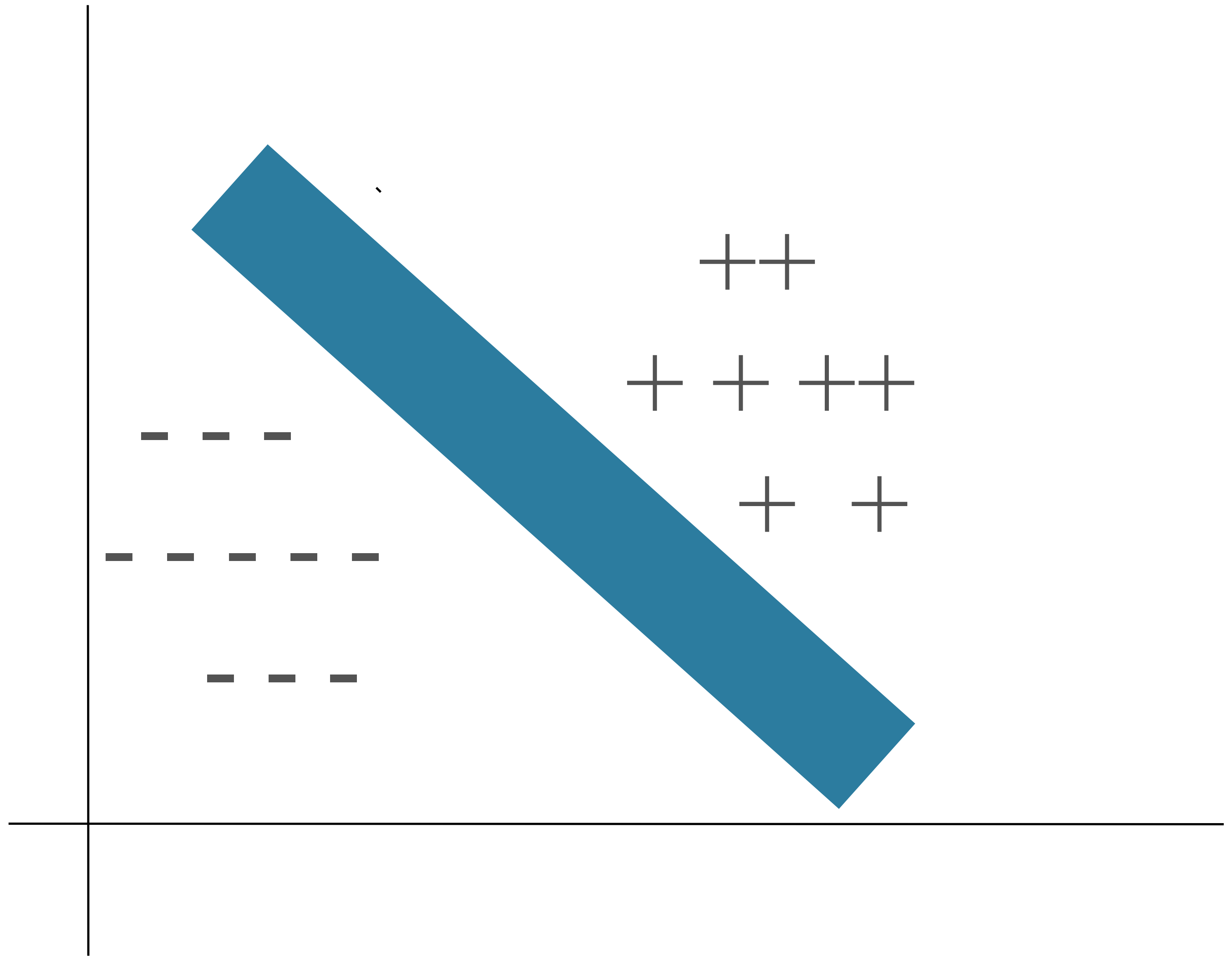
Maximum Margin Classifier

- What's best classifier?
  - Maximum margin
    - Biggest distance between decision boundary and closest examples
- Why is this better?
  - Intuition:
    - Which instances are we most sure of?



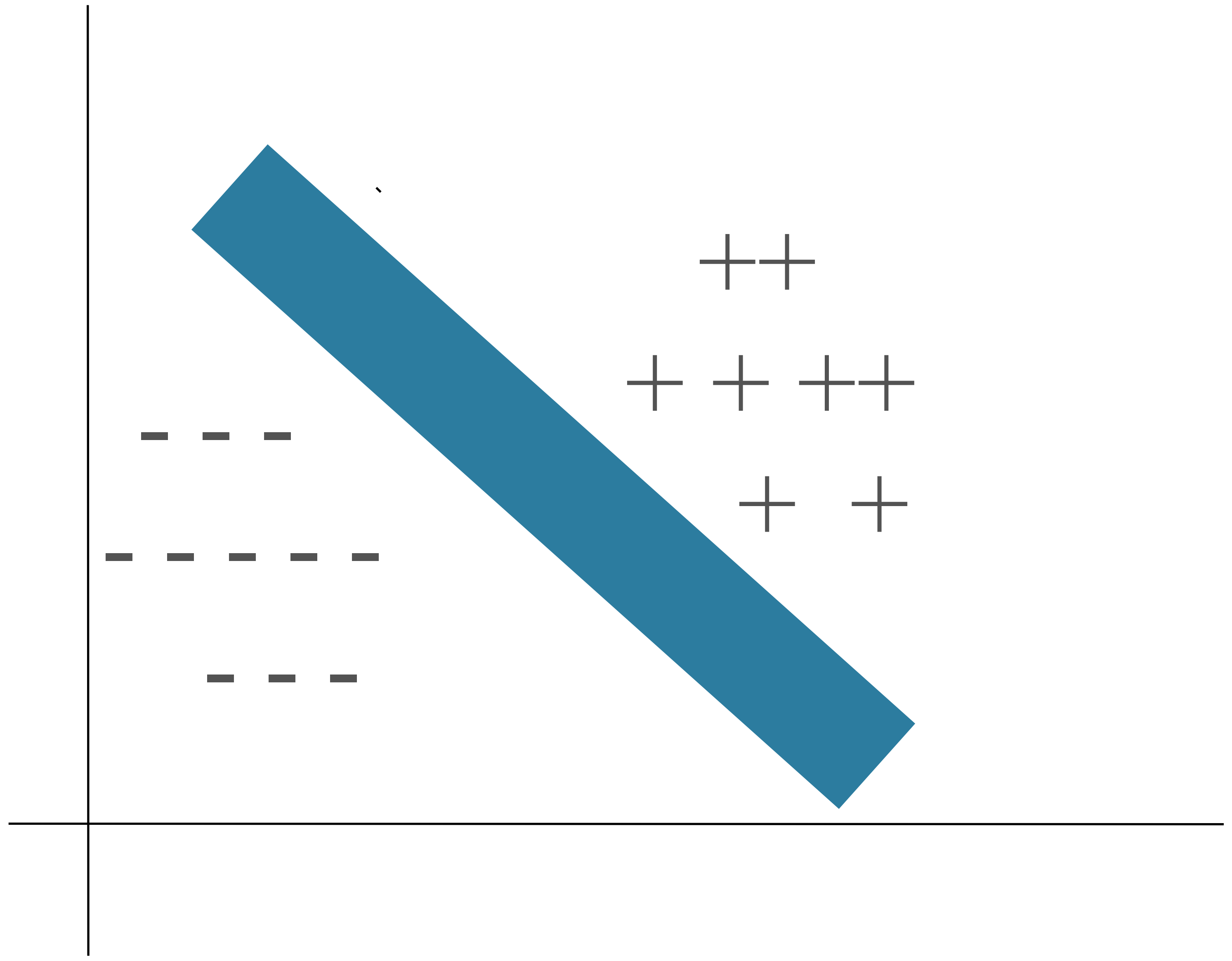
Maximum Margin Classifier

- What's best classifier?
  - Maximum margin
    - Biggest distance between decision boundary and closest examples
- Why is this better?
  - Intuition:
    - Which instances are we most sure of?
      - Furthest from boundary
    - Least sure of?



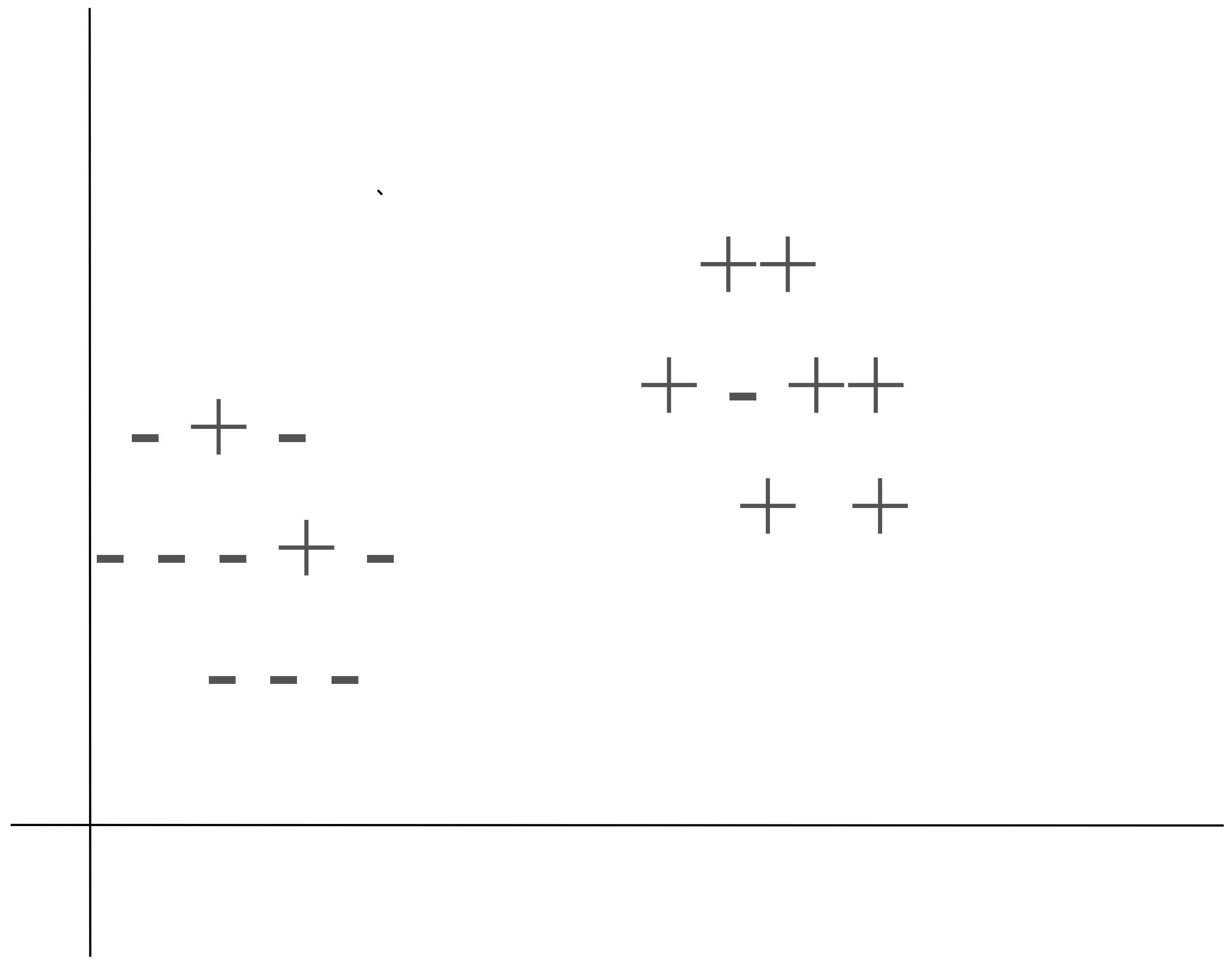
Maximum Margin Classifier

- What's best classifier?
  - Maximum margin
    - Biggest distance between decision boundary and closest examples
- Why is this better?
  - Intuition:
    - Which instances are we most sure of?
      - Furthest from boundary
    - Least sure of?
      - Closest
    - Create boundary with most 'room' for error in attributes



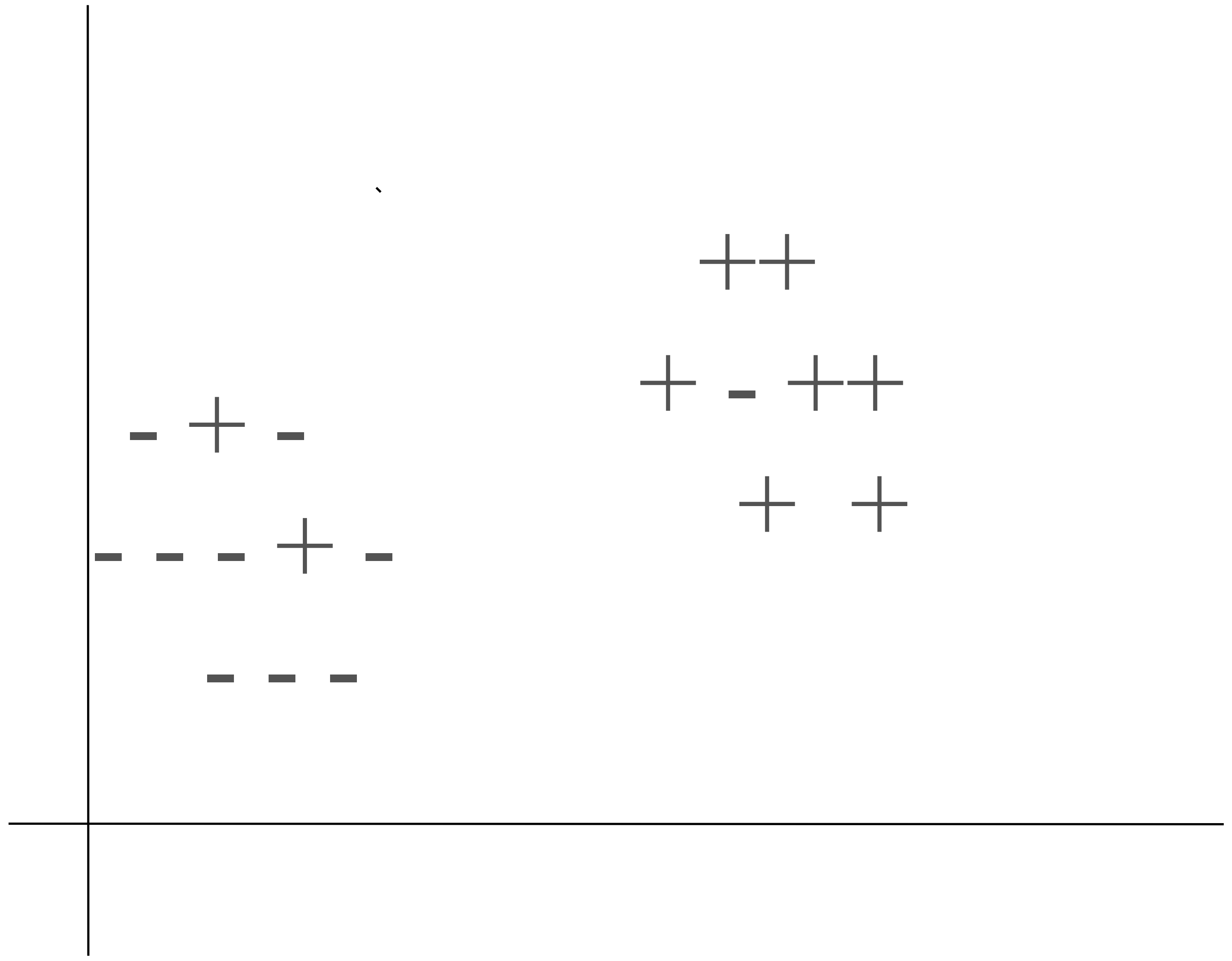
Complicating Classification

- Consider the new 2-D data:
- +: Class +1;   -: Class -1
- Can we draw a line that separates the two classes?



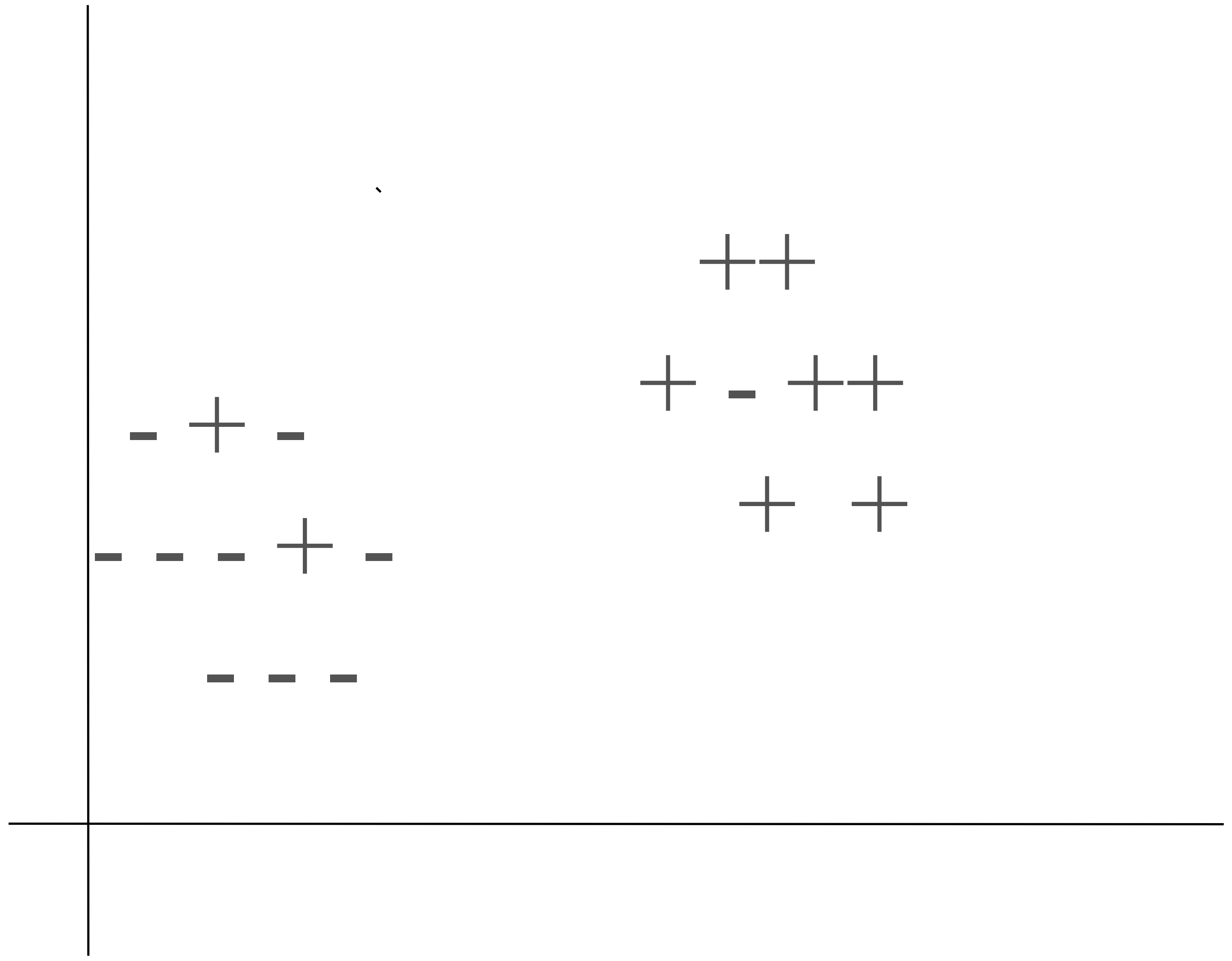
Complicating Classification

- Consider the new 2-D data
- +: Class +1;   -: Class -1
- Can we draw a line that separates the two classes?
  - No.
- What do we do?
  - Give up and try another classifier? No.



Noisy/Nonlinear Classification

- Consider the new 2-D data
- +: Class +1; -: Class -1
- Two basic approaches:
  - Use a linear classifier, but allow some (penalized) errors
    - soft margin, slack variables
  - Project data into higher dimensional space
    - Do linear classification there
    - Kernel functions



Multiclass Classification

- SVMs create linear decision boundaries
  - At basis binary classifiers

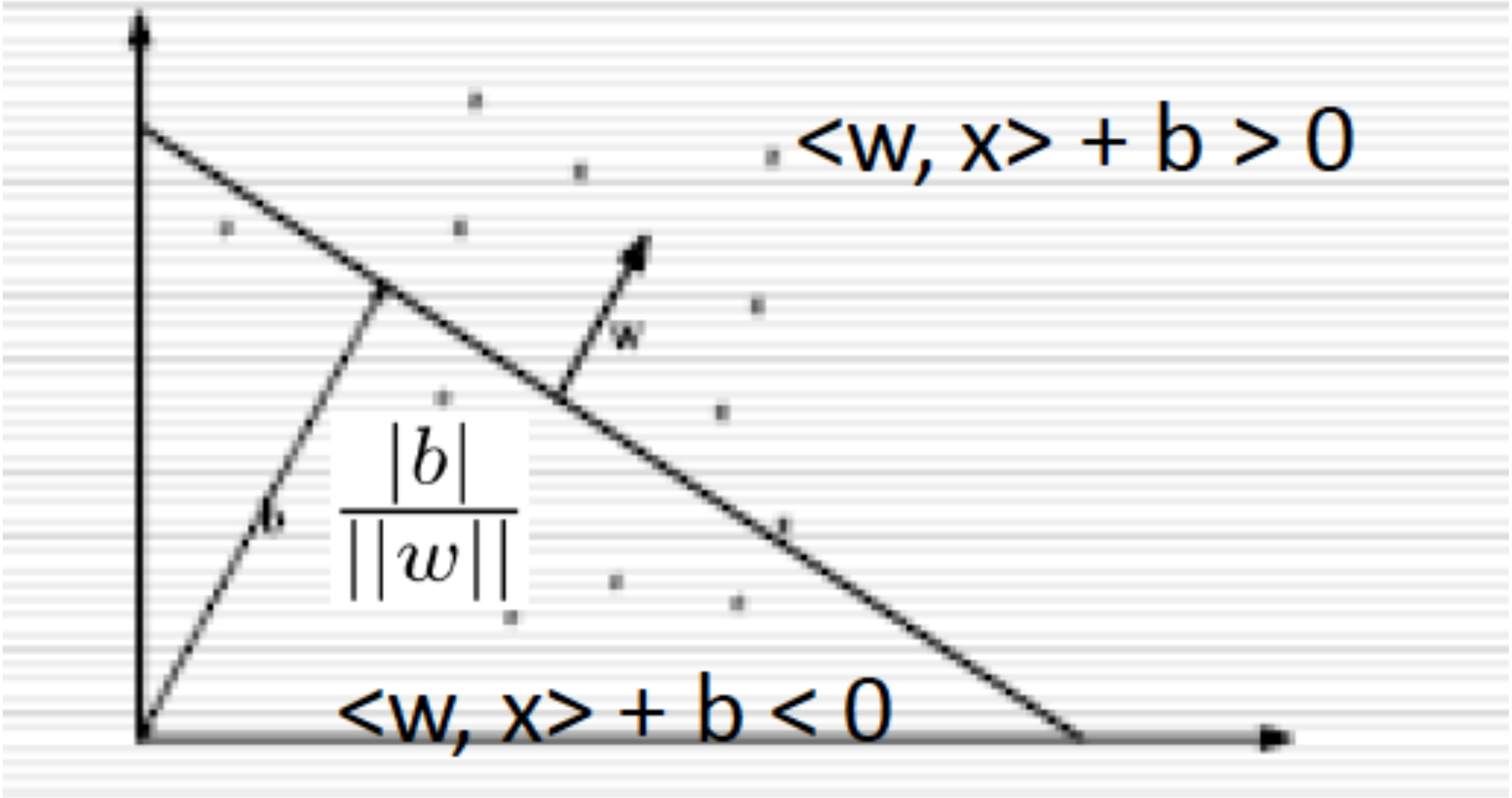
- How can we do multiclass classification?
  - One-vs-all
  - All-pairs
  - ECOC
  - ...

SVM Implementations

- Many implementations of SVMs:
  - SVM-Light: Thorsten Joachims
    - http://svmlight.joachims.org
  - LibSVM: C-C. Chang and C-J. Lin
    - http://www.csie.ntu.edu.tw/~cjlin/**libsvm**/
    - Scikit-learn wrapper: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC
  - Weka's SMO
  - …

SVMs: More Formally

- A hyperplane: $\langle w, x \rangle + b = 0$
- *w:* normal vector (aka weight vector), which is perpendicular to the hyperplane
- *b*: intercept term
- $||w||$:
  - Euclidean norm of *w*
- $\frac{|b|}{||w||}$ = offset from origin



Inner product example

- Inner product between two vectors

$$< \vec{x}, \vec{z} >= \sum_i x_i z_i$$

$$\vec{x} = (1,\ 2)$$

$$\vec{z} = (\text{-}2,\ 3)$$

$$< \vec{x}, \vec{z} > = 1*(\text{-}2) + 2*3$$
$$= \text{-}2 + 6 = 4$$

Inner product (cont'd)

$$< \vec{x}, \vec{z} >= \sum_i x_i z_i$$

$$cos(\vec{x}, \vec{z})$$

$$= \frac{\sum_i x_i z_i}{||x|| * ||z||} \qquad \text{where } ||x|| = \sqrt{\sum_i x_i^2}$$

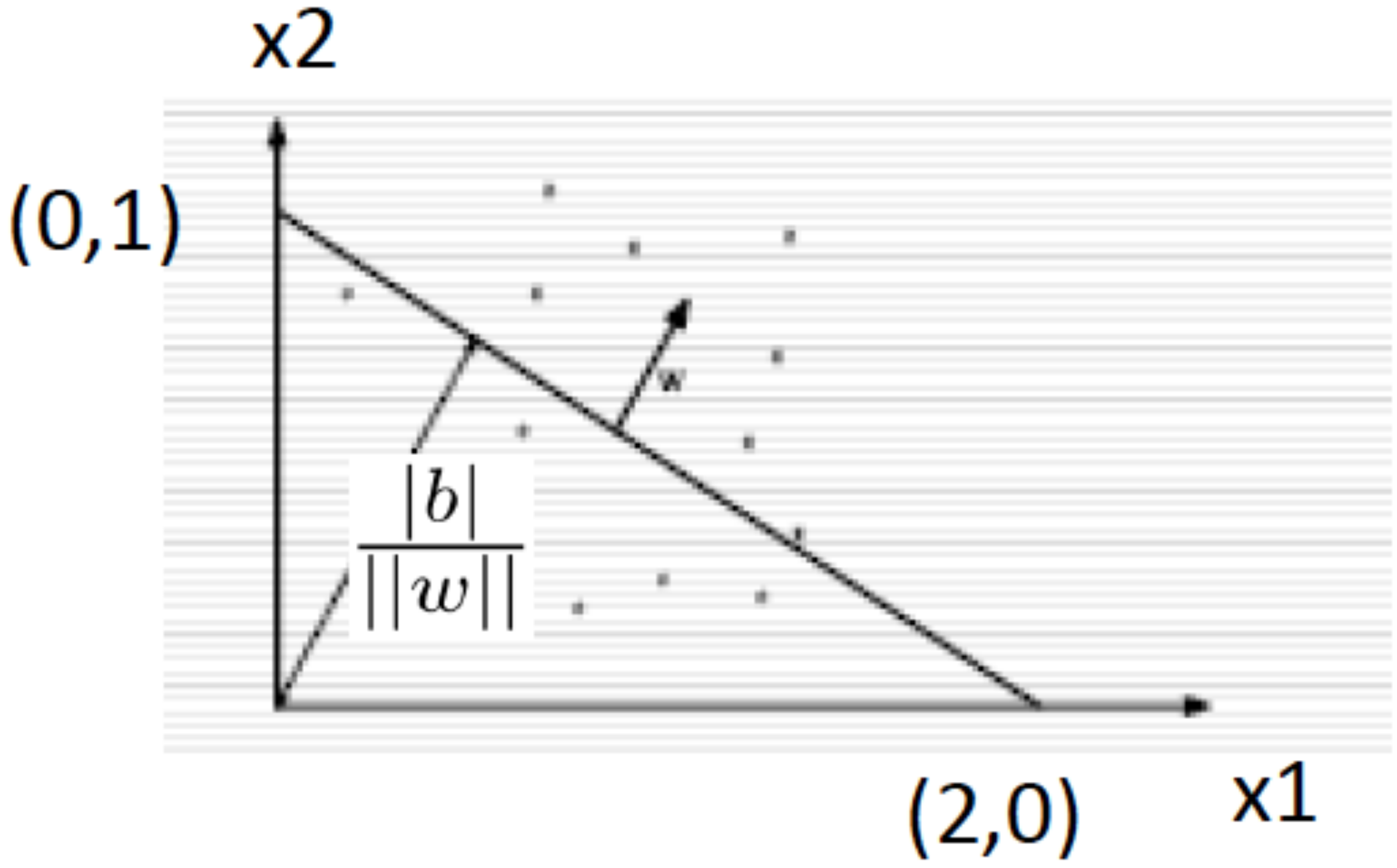
$$= \frac{<x, z>}{||x|| * ||z||}$$

cosine similarity = scaled inner product

Inner product is a similarity function.

Hyperplane Example

- <w,x>+b=0
- How many (w,b)s?
- Infinitely many!
  - Just scaling



$x_1+2x_2-2 = 0$ w=(1,2) b=-2

$10x_1+20x_2-20 = 0$ w=(10,20) b=-20

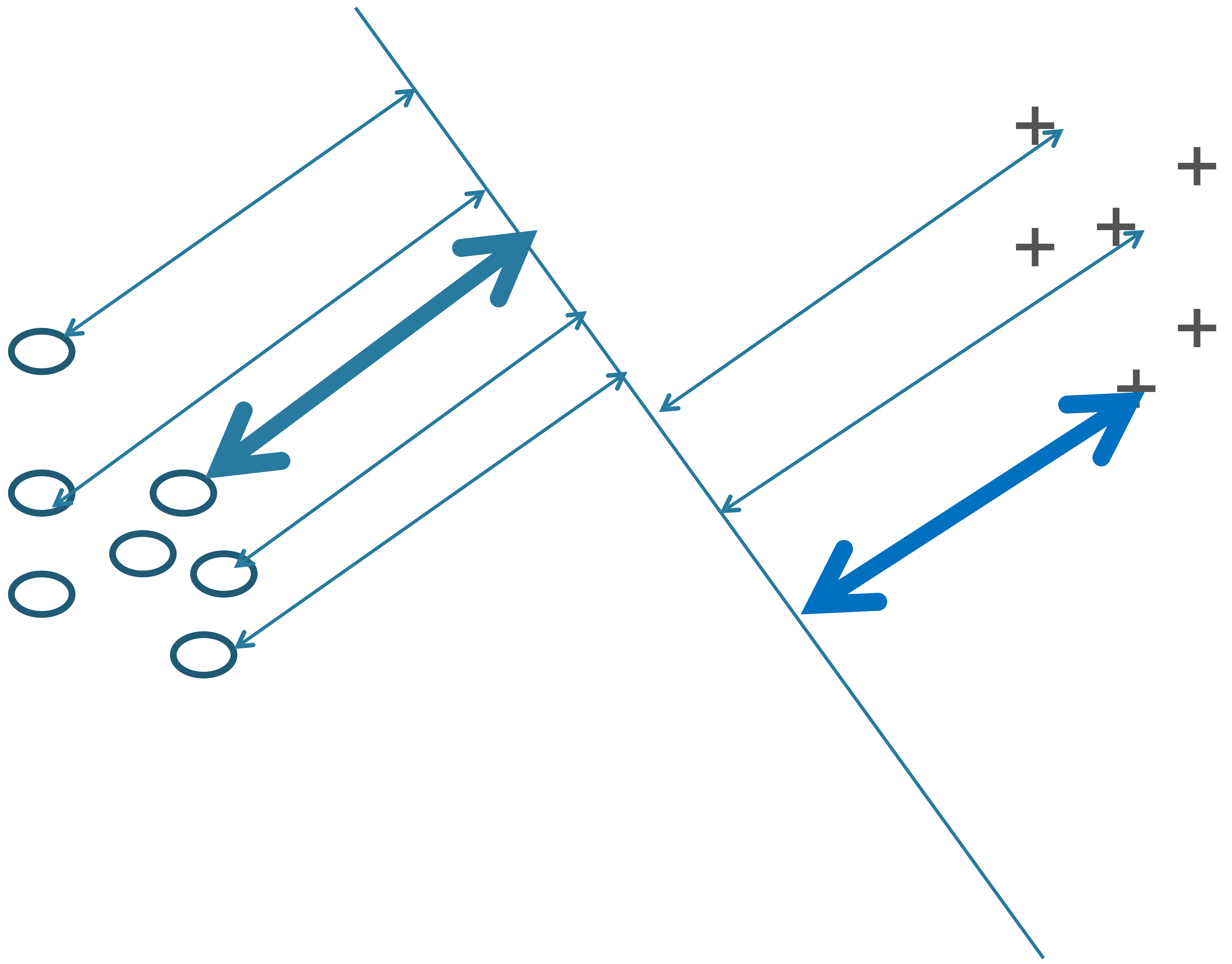
Finding a hyperplane

- Given the training instances, we want to find a hyperplane that separates them.

- If there is more than one hyperplane, SVM chooses the one with the maximum margin.

$$max_{\vec{w},b} \; min_{\vec{x_i} \in S} \{||\vec{x} - \vec{x_i}|| \;\mid\; \vec{x} \in R^N, < \vec{w}, \vec{x} > +b = 0\}$$

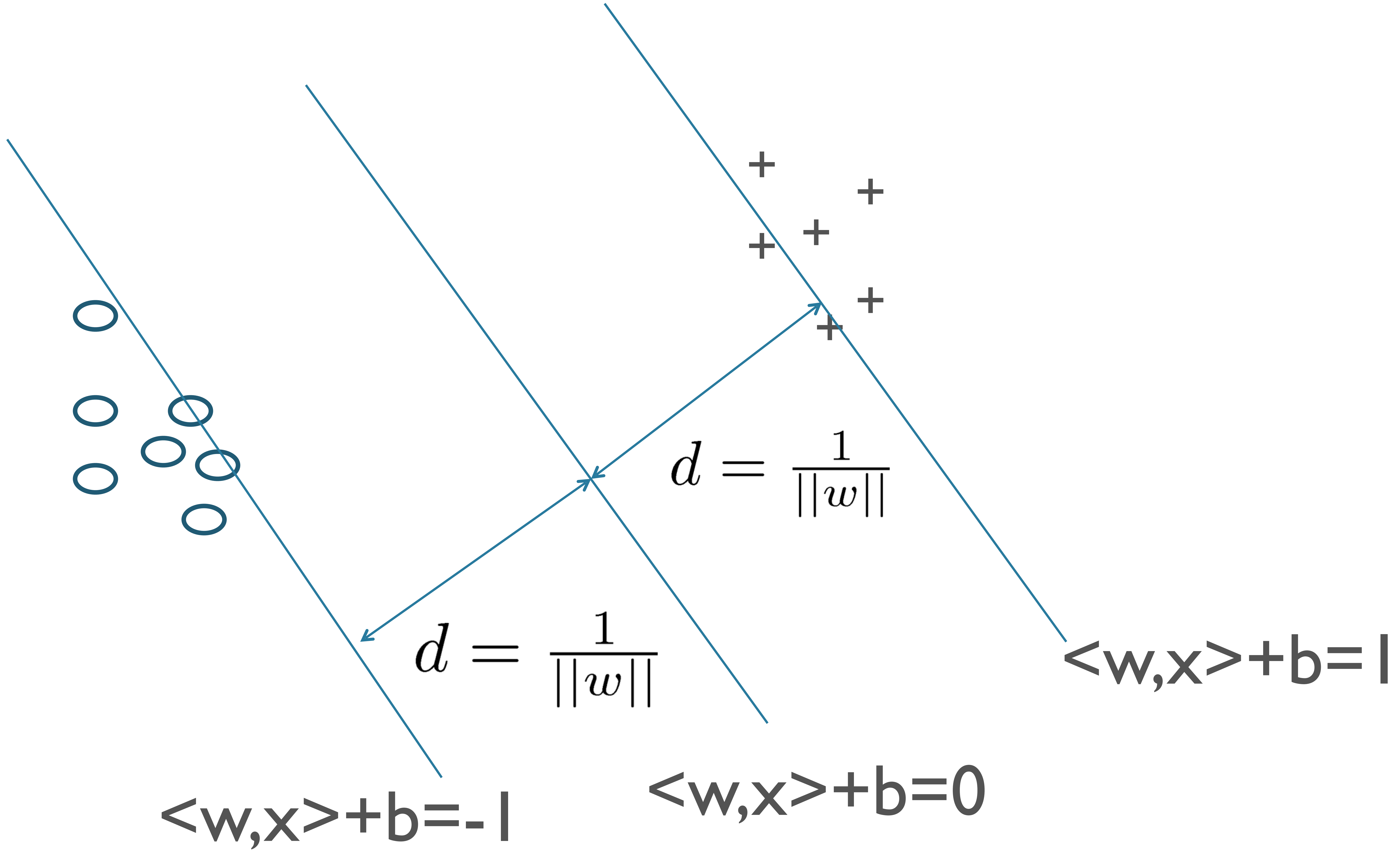
Maximizing the margin



Training: to find w and b.

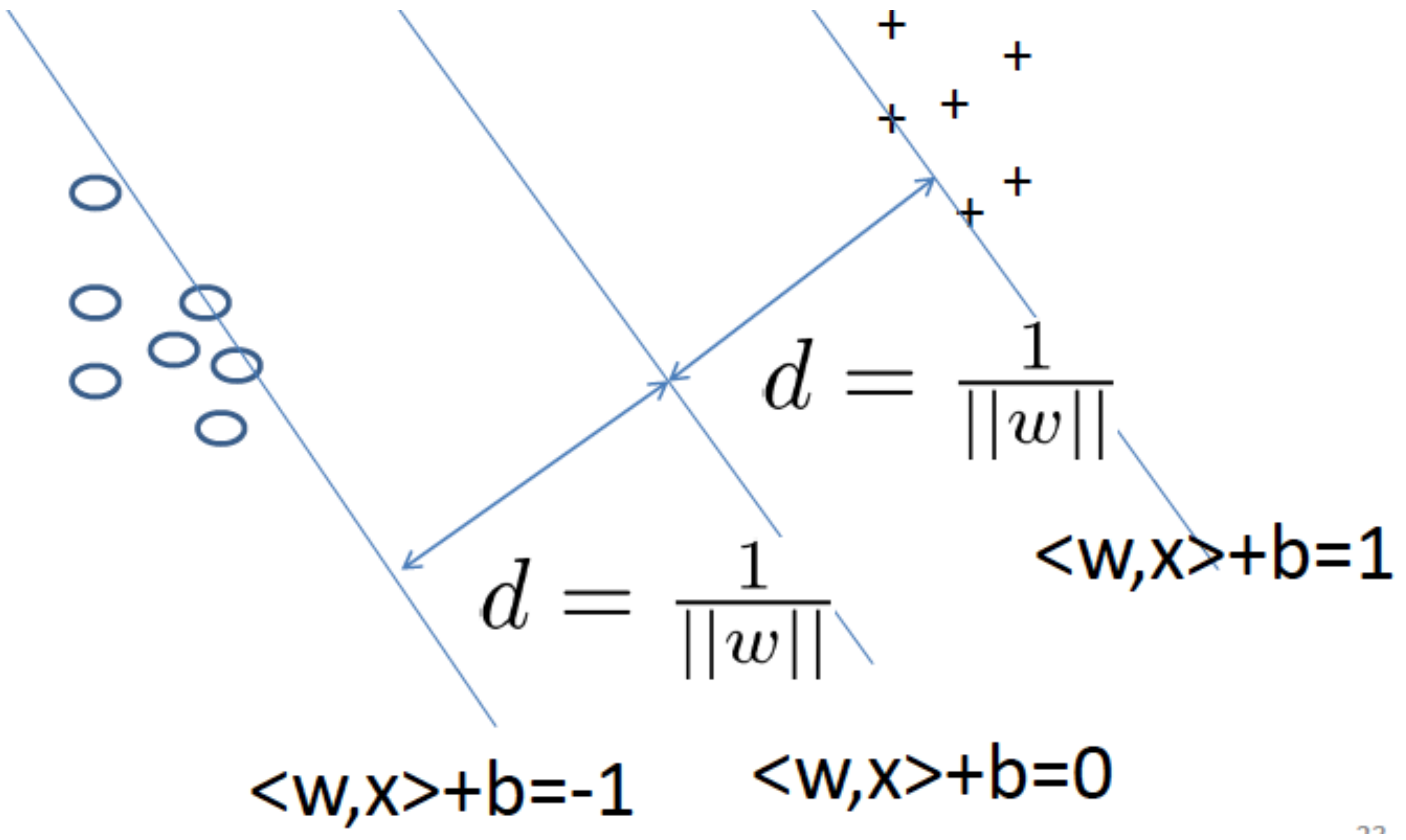
<w,x>+b=0

Support vectors



Margins & Support Vectors

- Closest instances to hyperplane:
  - “Support Vectors”
  - Both pos/neg examples



- Add Hyperplanes through
  - Support vectors

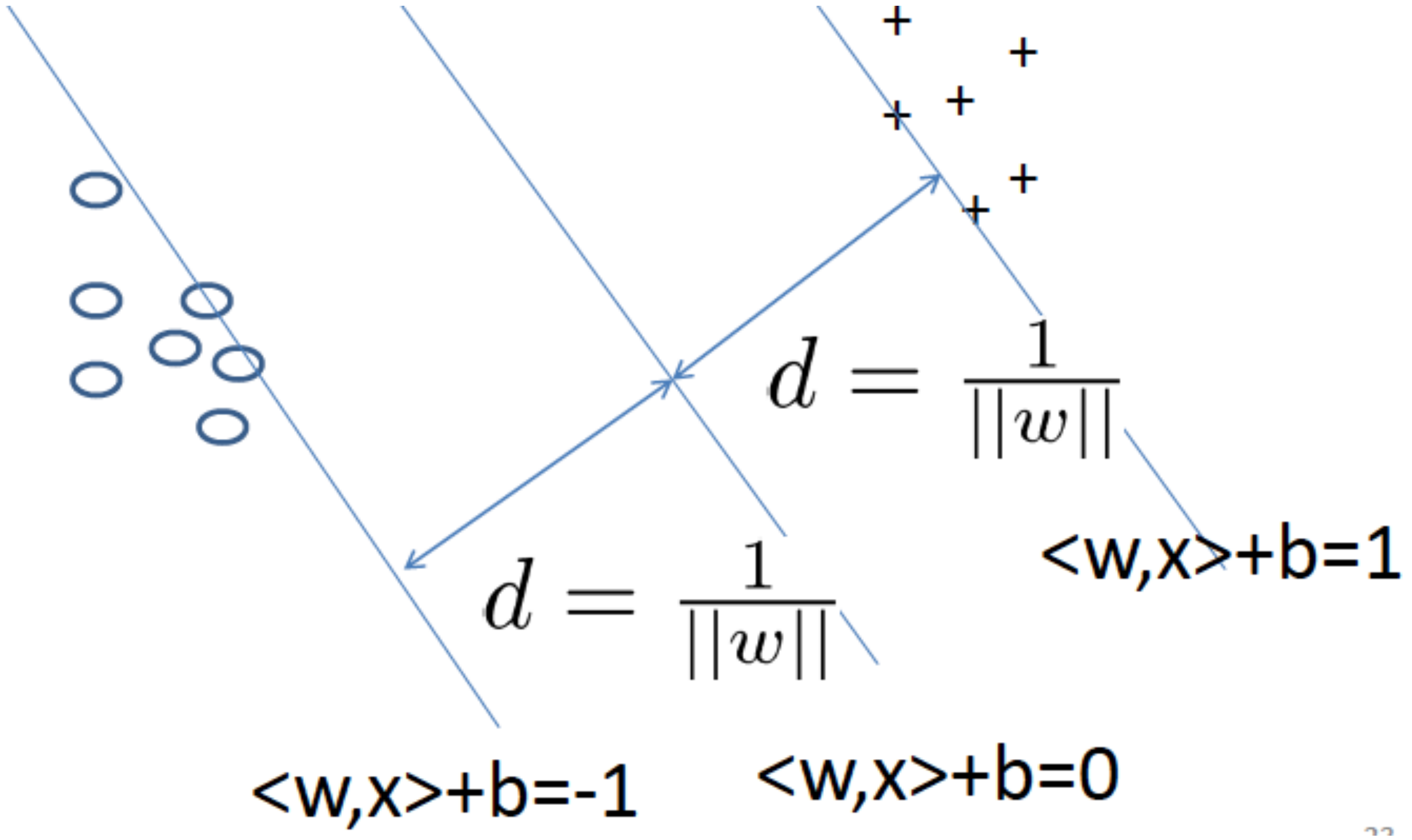
- d= 1/llwll

- How do we pick support vectors? Training

- How many are there? Depends on data set

SVM Training

- Goal: Maximum margin, consistent w/training data
  - Margin = 1 /llwll
- How can we maximize?
  - Max d ➔ Min llwll
- So we are:
  - Minimizing $||w||^2$
    subject to
    $y_i(<w,x_i>+b) >= 1$
- Quadratic Programming (QP) problem
  - Can use standard QP solvers



$$y_i(< \vec{w}, \vec{x_i} > +b) \geq 1$$

Let w=(w1, w2, w3, w4, w5)

X1 1 f1:2 f3:3.5 f4:-1
X2 -1 f2:-1 f3:2
X3 1 f1:5 f4:2 f5:3.1

We are trying to choose w and b for the hyperplane wx + b = 0

1*(2w1 + 3.5w3 - w4) >= 1
(-1)*(-w2 + 2w3) >= 1
1*(5w1 + 2w4 + 3.1w5) >= 1

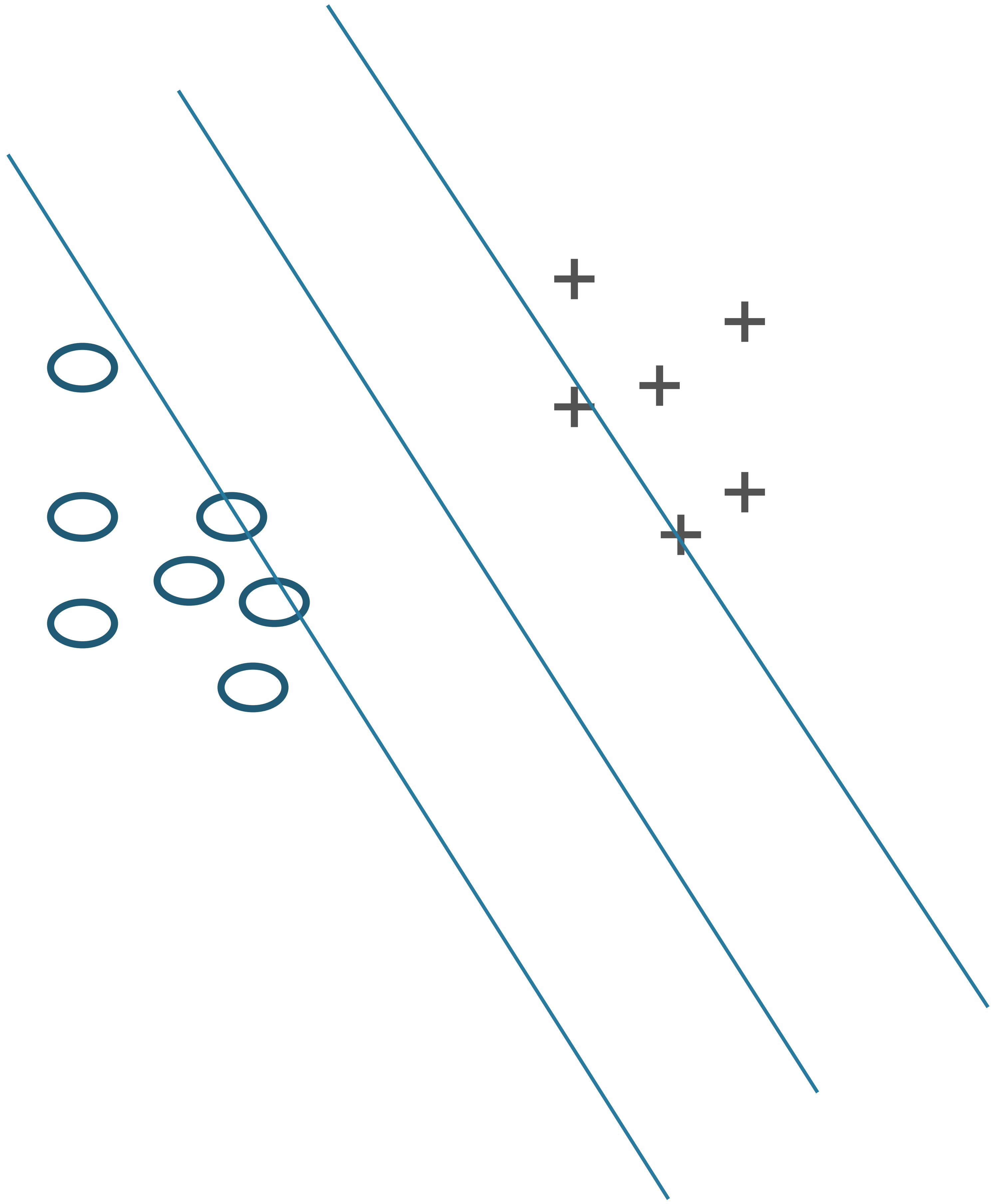
➔

2w1 + 3.5w3 – w4 >= 1
-w2 +2w3 <= 1
5w1 + 2w4 + 3.1w5 >= 1

With those constraints, we want to minimize

$w_1^2+w_2^2+w_3^2+w_4^2+w_5^2$

Training (cont'd)



$$\text{Minimize } ||w||^2$$

subject to the constraint

$$y_i(<\vec{w},\vec{x_i}> +b) \geq 1$$

$\Updownarrow$

$$y_i(<\vec{w},\vec{x_i}> +b) - 1 \geq 0$$

Lagrangian**

For each training instance $(\vec{x_i}, y_i)$, introduce $\alpha_i \geq 0$.

Let $\alpha = (\alpha_1, \alpha_2, ..., \alpha_N)$

$$L(\vec{w}, b, \alpha) = \frac{1}{2}||\vec{w}||^2 - \sum_i \alpha_i(y_i(< \vec{w}, \vec{x_i} > +b) - 1)$$

minimize L w.r.t. $\vec{w}$ and b

$$\vec{w} = \sum_{i=1}^N \alpha_i y_i \vec{x_i} \text{ and } \sum_{i=1}^N \alpha_i y_i = 0$$

The dual problem **

- Find $\alpha 1, \ldots, \alpha N$ such that the following is maximized

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j < \vec{x_i}, \vec{x_j} >$$

- Subject to

$$\alpha_i \geq 0 \text{ and } \sum_i \alpha_i y_i = 0$$

- The solution has the form

$$\vec{w} = \sum_i \alpha_i y_i \vec{x}_i$$

$$b = y_k - < \vec{w}, \vec{x}_k >$$ for any x_k whose weight is non-zero

An example

$$\vec{w} = \sum_i \alpha_i y_i \vec{x}_i$$

$x_1=(1,0,3)$, $y_1= 1$, $\alpha_1=2$

$x_2=(-1,2,0)$, $y_2=-1$, $\alpha_2=3$

$x_3=(0,-4,1)$, $y_3=1$, $\alpha_3=0$

An example

$$\vec{w} = \sum_i \alpha_i y_i \vec{x_i}$$

x_1=(1,0,3), y_1= 1, α_1=2

x_2=(-1,2,0), y_2=-1, α_2=3

x_3=(0,-4,1), y_3=1 , α_3=0

w= (1*1*2+ (-1)*(-1)*3+0*1*0,
0 + 2*(-1)*3+0,
3*1*2+0+0)
= (5,-6,6)

For support vectors, $\alpha_i > 0$

For other training examples, $\alpha_i = 0$

Removing them will not change the model.

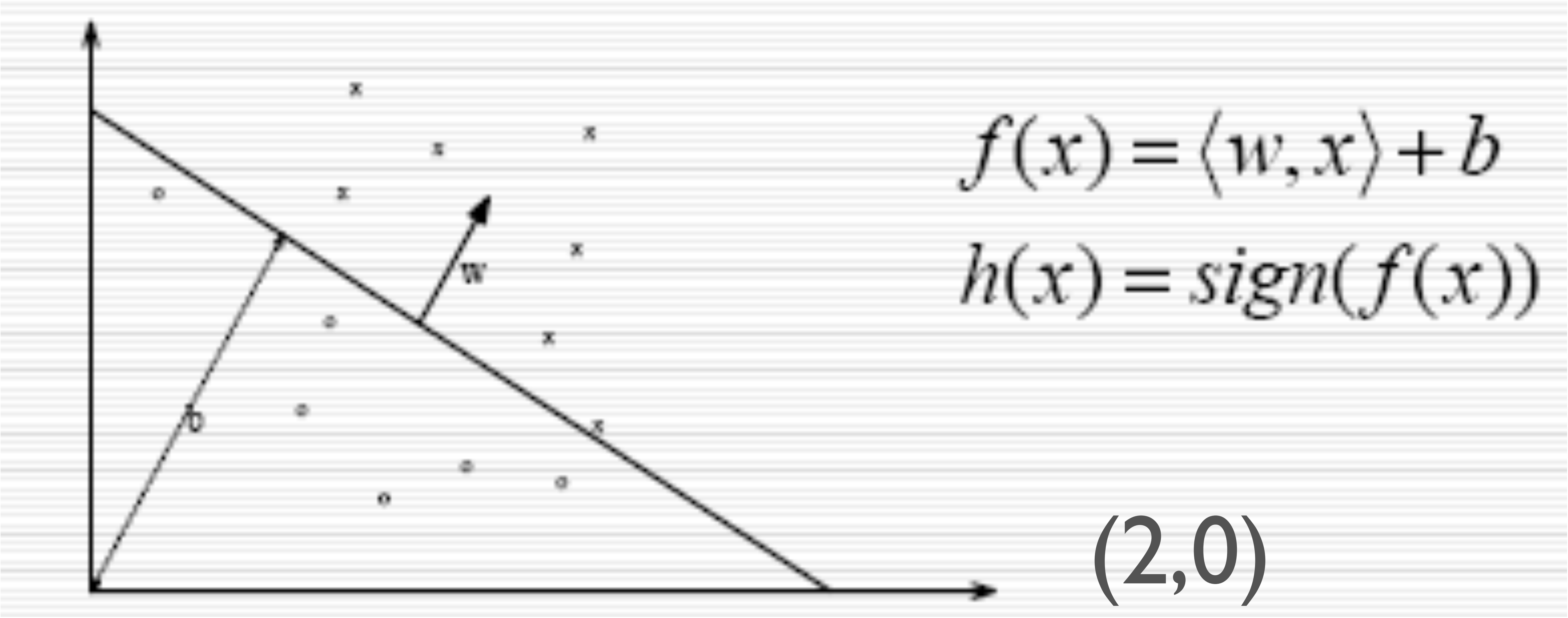
Finding w is equivalent to finding support vectors and their weights.

Finding the solution

- This is a Quadratic Programming (QP) problem.
- The function is convex and there are no local minima.
- Solvable in polynomial time.

Decoding with w and b

(0,1)



Hyperplane: w=(1,2), b=-2

$f(x) = x_1 + 2\ x_2 - 2$

x=(3,1) f(x) = 3+2-2 = 3 > 0

x=(0,0) f(x) = 0+0-2 = -2 < 0

Decoding with α_i

$$\vec{w} = \sum_i \alpha_i y_i \vec{x_i}$$

Decoding:

$$f(\vec{x}) = < \vec{w}, \vec{x} > +b$$

$$f(\vec{x}) = < \sum_i \alpha_i y_i \vec{x_i}, \vec{x} > +b$$

$=\Sigma_i<\alpha_i y_i x_i, x> +b$

$$= \sum_i \alpha_i y_i < \vec{x_i}, \vec{x} > +b$$

$$< u + v, w > = < u, w > + < v, w >$$

$$< cu, v > = c < u, v >$$

kNN vs. SVM

- Majority voting:

  $c^* = \arg\max_c g(c)$

- Weighted voting: weighting is on each neighbor

  $c^* = \arg\max_c \sum_i w_i \, \delta(c, f_i(x))$

- Weighted voting allows us to use more training examples:

  e.g., $w_i = 1/\text{dist}(x, x_i)$ $\quad f(\vec{x}) = \sum_i w_i y_i$ (weighted kNN, 2-class)

  ➔ We can use all the training examples.

$$f(\vec{x}) = \sum_i \alpha_i y_i < \vec{x_i}, \vec{x} > + b$$

$$= \sum_i \alpha_i < \vec{x_i}, \vec{x} > y_i + b \qquad \text{(SVM)}$$

Summary of linear SVM

- Main ideas:
  - Choose a hyperplane to separate instances:

    $\langle w,x \rangle + b = 0$
  - Among all the allowed hyperplanes, choose the one with the max margin
  - Maximizing margin is the same as minimizing $\|w\|$
  - Choosing w is the same as choosing α_i

The problem

Training: Choose $\vec{w}$ and b

Mimimizes $||w||^2$ subject to the constraints

$y_i(<\vec{w}, \vec{x_i}> +b) \geq 1$ for every $(\vec{x_i}, y_i)$

Decoding: Calculate $f(x) =< w, x > +b$

The dual problem **

Training: Calculate α_i for each $(\vec{x_i}, y_i)$

Maximize $L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j < \vec{x_i}, \vec{x_j} >$

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

Decoding: $f(\vec{x}) = \sum_i \alpha_i y_i < \vec{x_i}, \vec{x} > +b$

Remaining issues

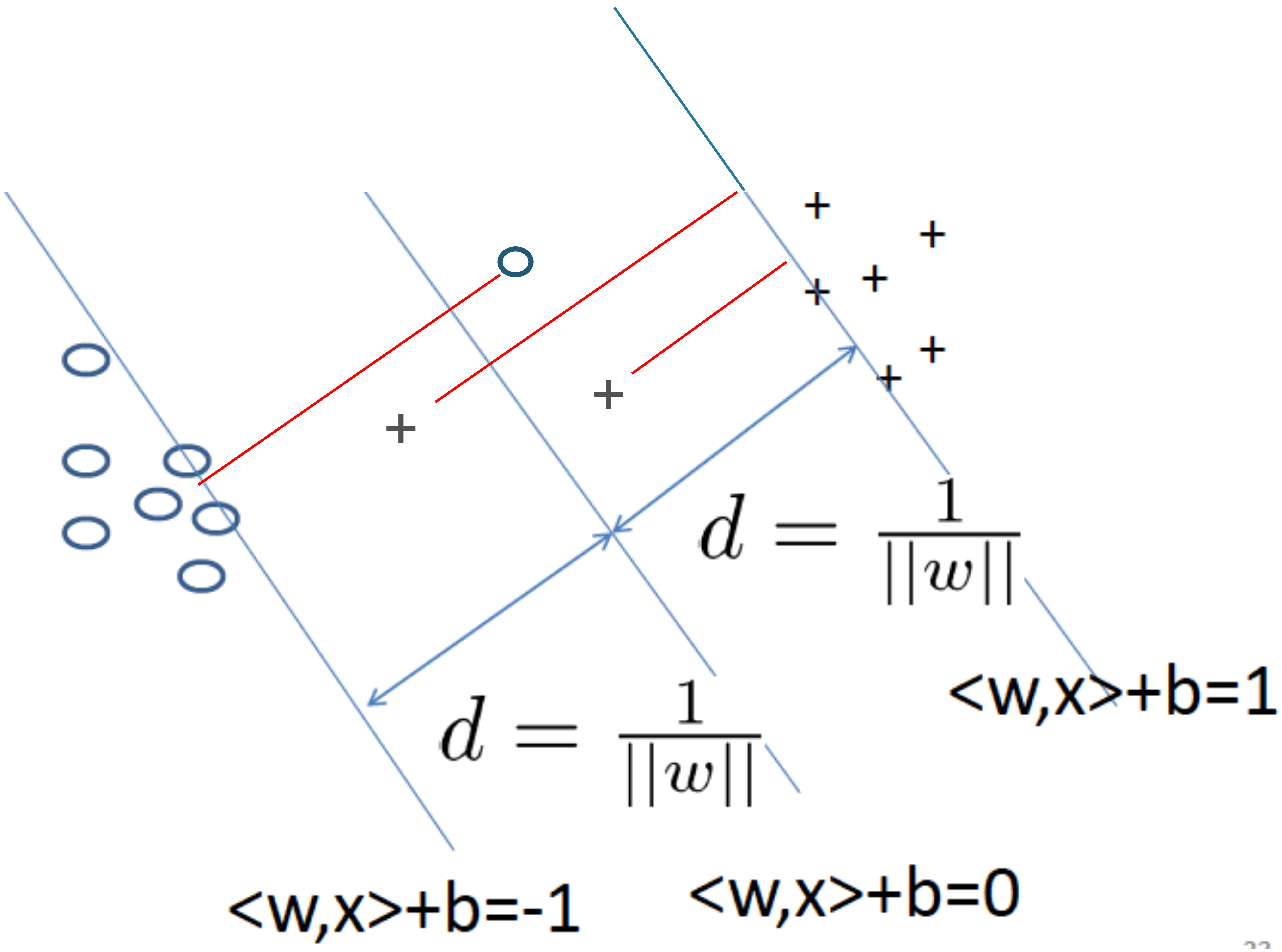
- Linear classifier: what if the data is not separable?
  - The data would be linearly separable without noise
    ➔ soft margin

  - The data is not linearly separable
    ➔ map the data to a higher-dimension space

Soft margin

Highlights

- Problem: Some data set is not separable or there are mislabeled examples.
- Idea: split the data as cleanly as possible, while maximizing the distance to the nearest cleanly split examples.
- Mathematically, introduce “slack variables”


$d = \frac{1}{||w||}$
<w,x>+b=1
$d = \frac{1}{||w||}$
<w,x>+b=-1
<w,x>+b=0

Objective Function

- For each training instance x_i, introduce a slack variable ξ_i
- Minimizing
- such that

$$\frac{1}{2}||w||^2 + C(\sum_i \xi_i)^k$$

$$y_i(< \vec{w}, \vec{x_i} > + b) \geq 1 - \xi_i$$

$$\text{where } \xi_i \geq 0$$

- C is a regularization term (for controlling overfitting),
- k = 1 or 2

Objective Function

- For each training instance x_i, introduce a slack variable ξ_i
- Minimizing
- such that

$$\frac{1}{2}||w||^2 + C(\sum_i \xi_i)^k$$

$$y_i(< \vec{w}, \vec{x_i} > +b) \geq 1 - \xi_i$$

$$\text{where } \xi_i \geq 0$$

- C is a regularization term (for controlling overfitting),
- k = 1 or 2

The dual problem**

- Maximize

$$L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j < \vec{x_i}, \vec{x_j} >$$

- Subject to

$$\boxed{C \geq} \alpha_i \geq 0 \text{ and } \sum_i \alpha_i y_i = 0$$

- The solution has the form

$$\vec{w} = \sum_i \alpha_i y_i \vec{x}_i$$

$$b = y_k(1-\xi_k) - \langle w, x_k \rangle \text{ for } \quad k = \operatorname{argmax}_k \alpha_k$$

X_i with non-zero α_i is called a support vector
Every data point which is misclassified or within the margin will have a non-zero α_i

$$\text{Decoding: Calculate } f(x) = < w, x > + b$$