

# Support Vector Machine

Mahdi Roozbahani  
Georgia Tech

# Math moment

$$7 \times 0 = 0$$

$$3 \times 0 = 0$$


$$0 = 0$$

$$7 \times 0 = 3 \times 0$$

$$7 = 3$$



# Outline

- Precursor: Linear Classifier and Perceptron 
- Support Vector Machine
- Parameter Learning

# Binary Classification

$$f(x) = x\Theta$$

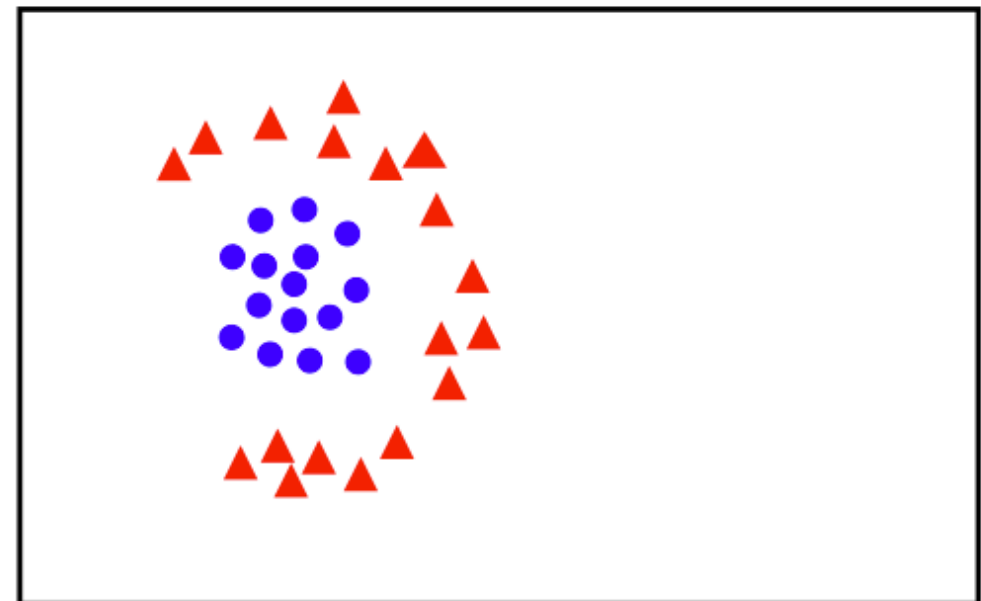
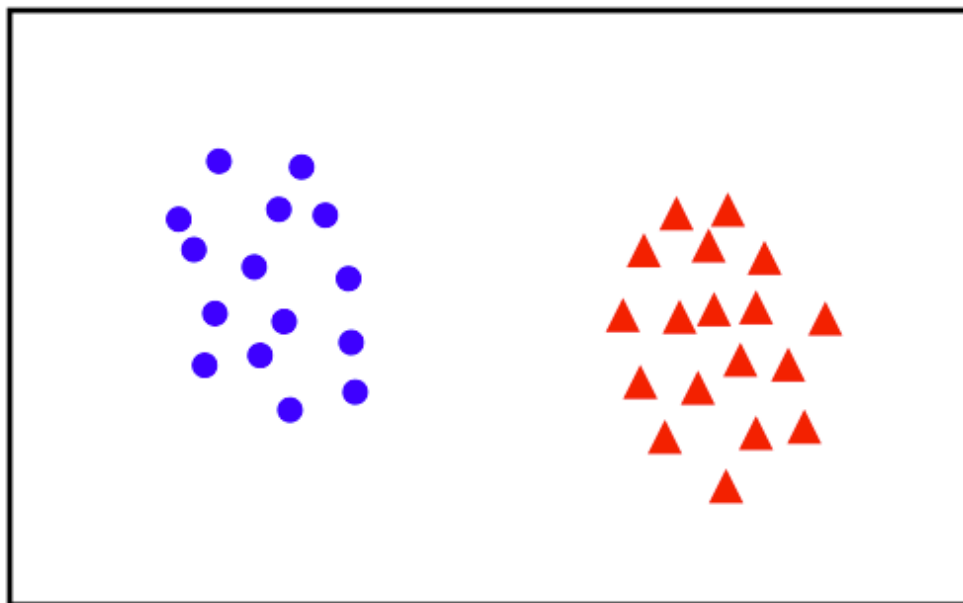
$$f(x_i) = x_i\Theta \leadsto d \times 1 \in \mathbb{R}$$

$\hookrightarrow 1 \times d$

Given training data  $(\mathbf{x}_i, y_i)$  for  $i = 1 \dots N$ , with  $\mathbf{x}_i \in \mathbb{R}^d$  and  $y_i \in \{-1, 1\}$ , learn a classifier  $f(\mathbf{x})$  such that

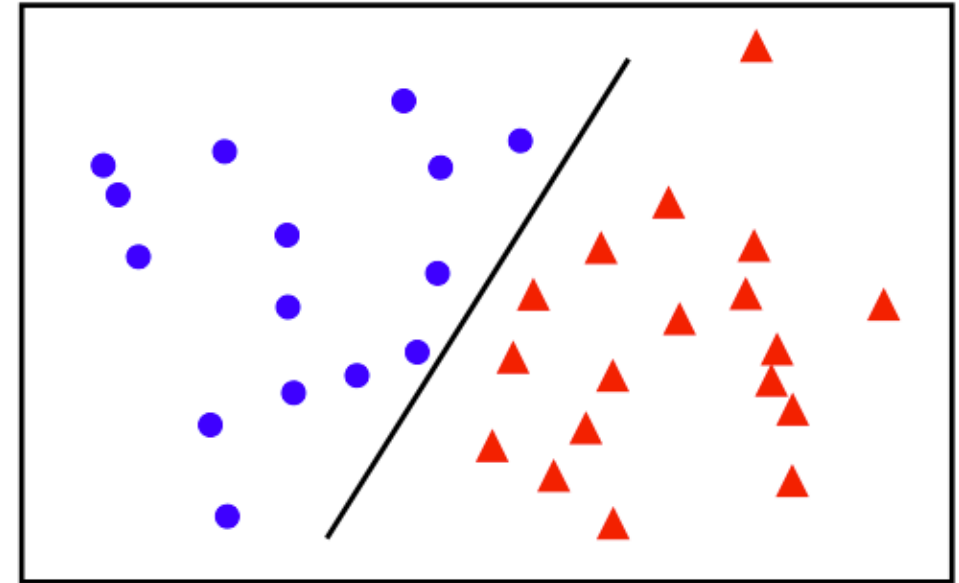
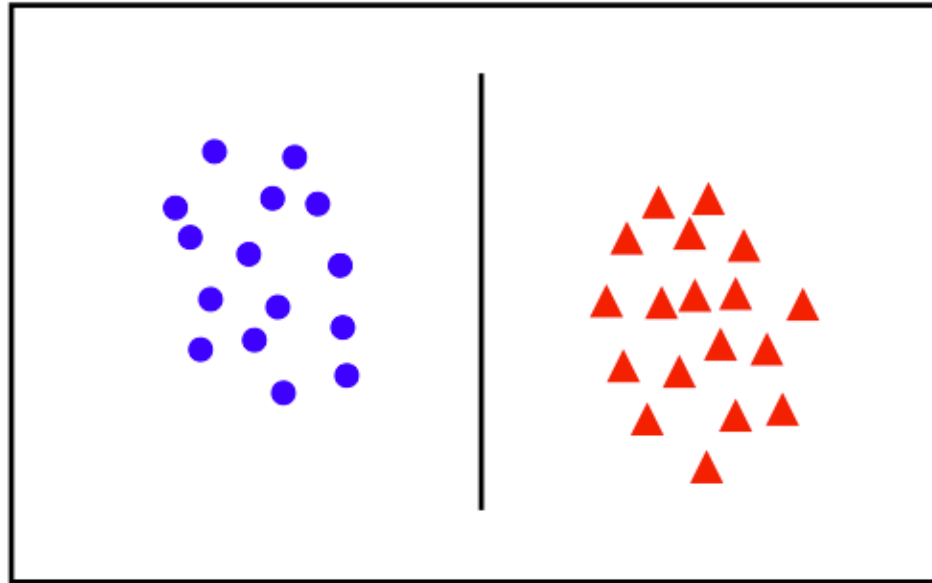
$$f(\mathbf{x}_i) \begin{cases} \geq 0 & +1 \\ < 0 & -1 \end{cases}$$

$\begin{matrix} + & + \\ - & - \end{matrix}$   
i.e.  $y_i f(\mathbf{x}_i) > 0$  for a correct classification.

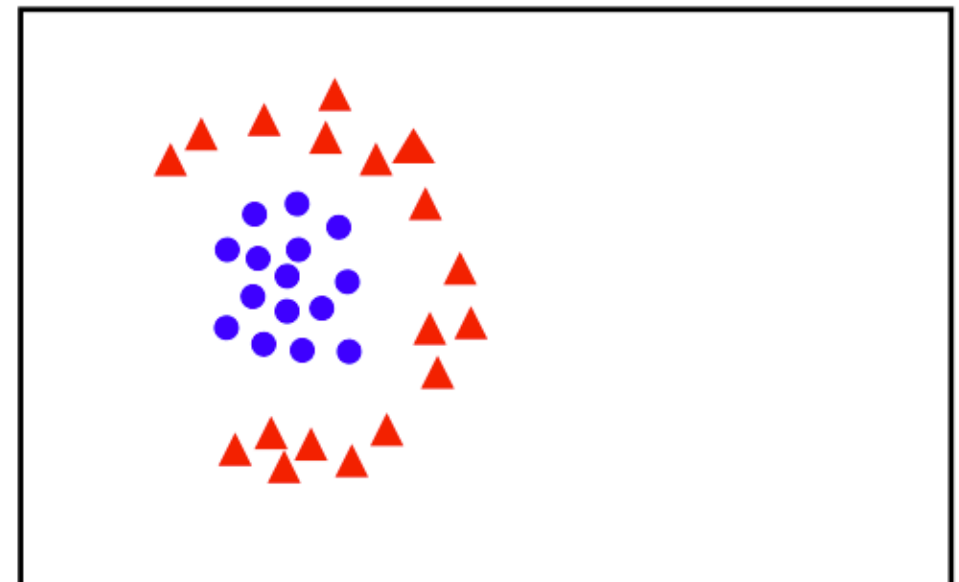
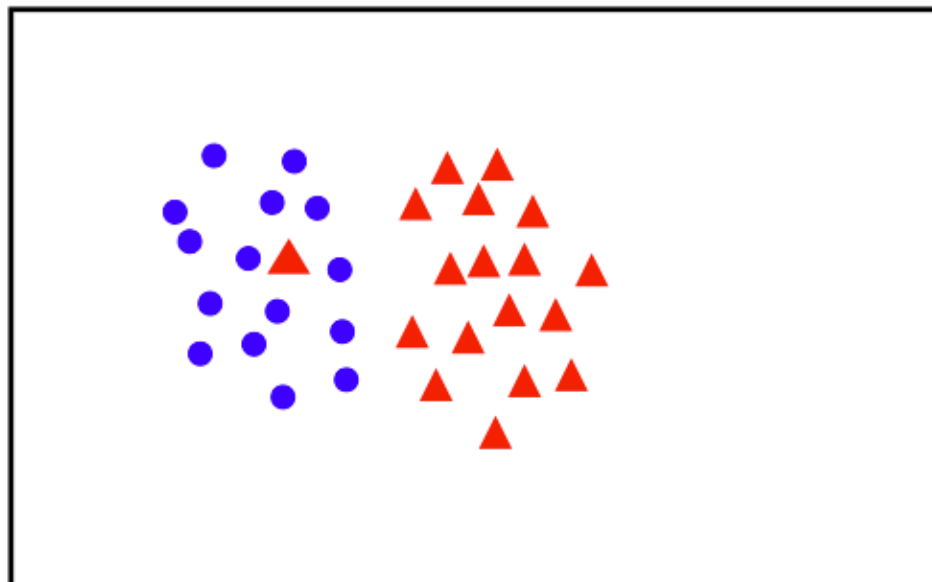


# Linear Separability

linearly  
separable



not  
linearly  
separable

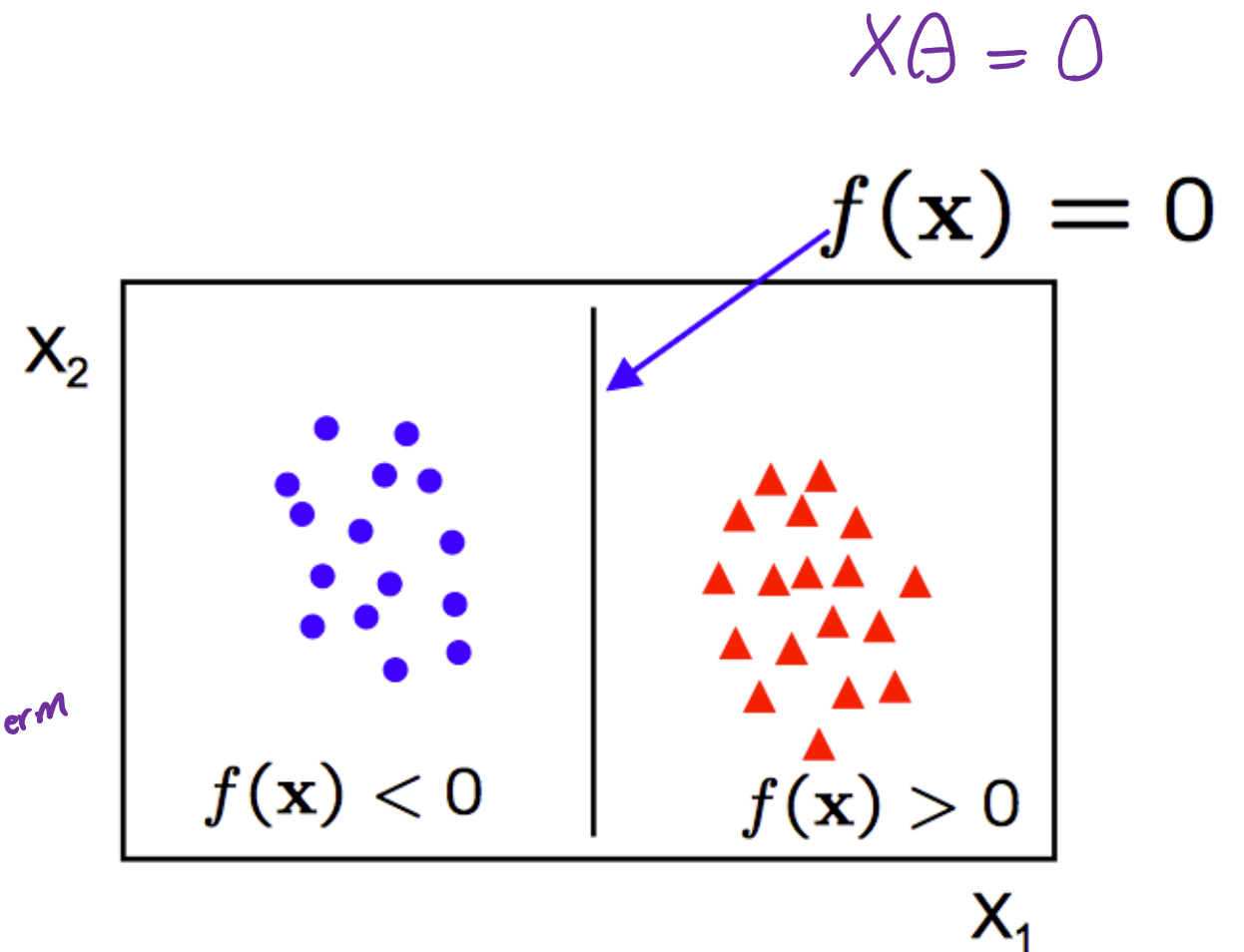


# Linear Classifier

A linear classifier has the form

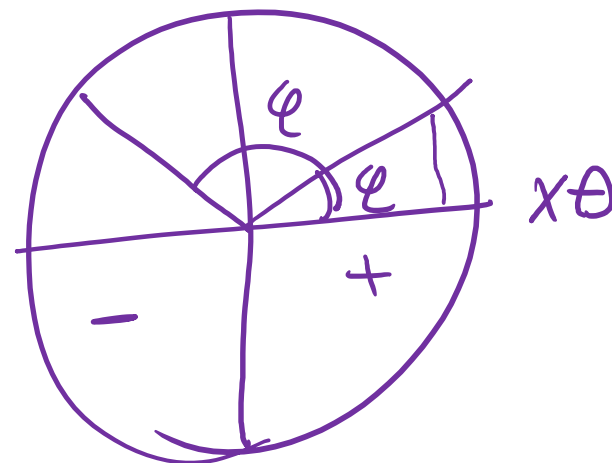
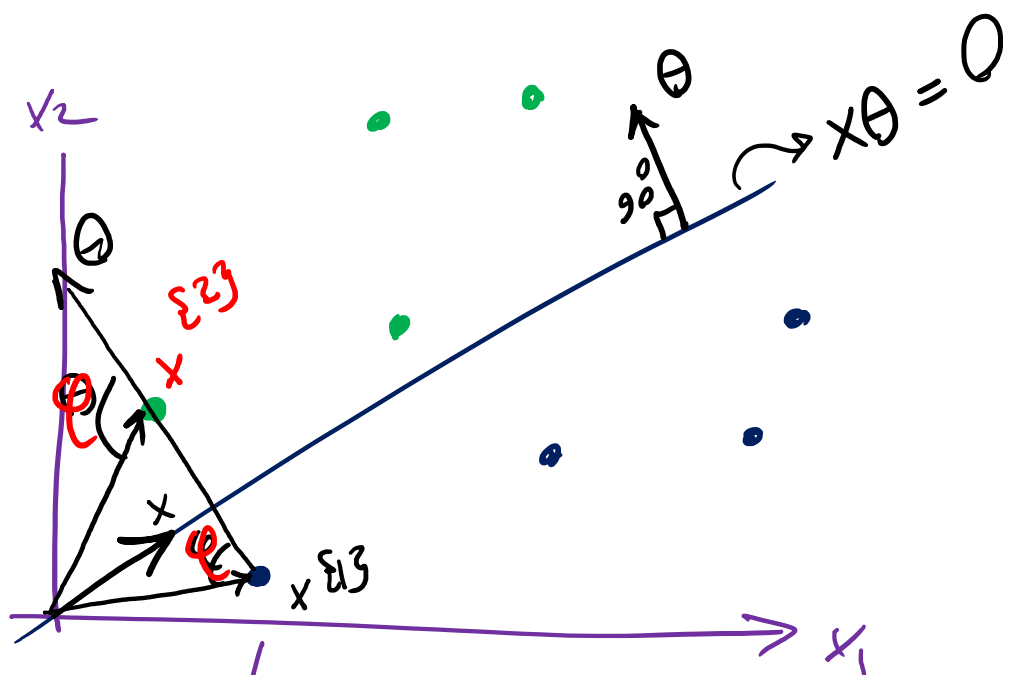
$$f(x) = x\theta + \theta_0$$

*( $\theta_0$  is circled with a purple line and labeled "bias term" with a purple arrow)*



- in 2D the discriminant is a line
- $\theta$  is the **normal** to the line, and  $\theta_0$  the **bias**
- $\theta$  is known as the **weight vector**

$$X^{\{1\}}_{1 \times d}$$



$$X^{\{1\}} \cdot \theta = |X^{\{1\}}| |\theta| \cos \phi > 0$$

$$\text{Sign}(>0) = +1$$

$$\downarrow$$

$$\phi < 90$$

$$X^{\{2\}} \cdot \theta = |X^{\{2\}}| |\theta| \cos \phi < 0$$

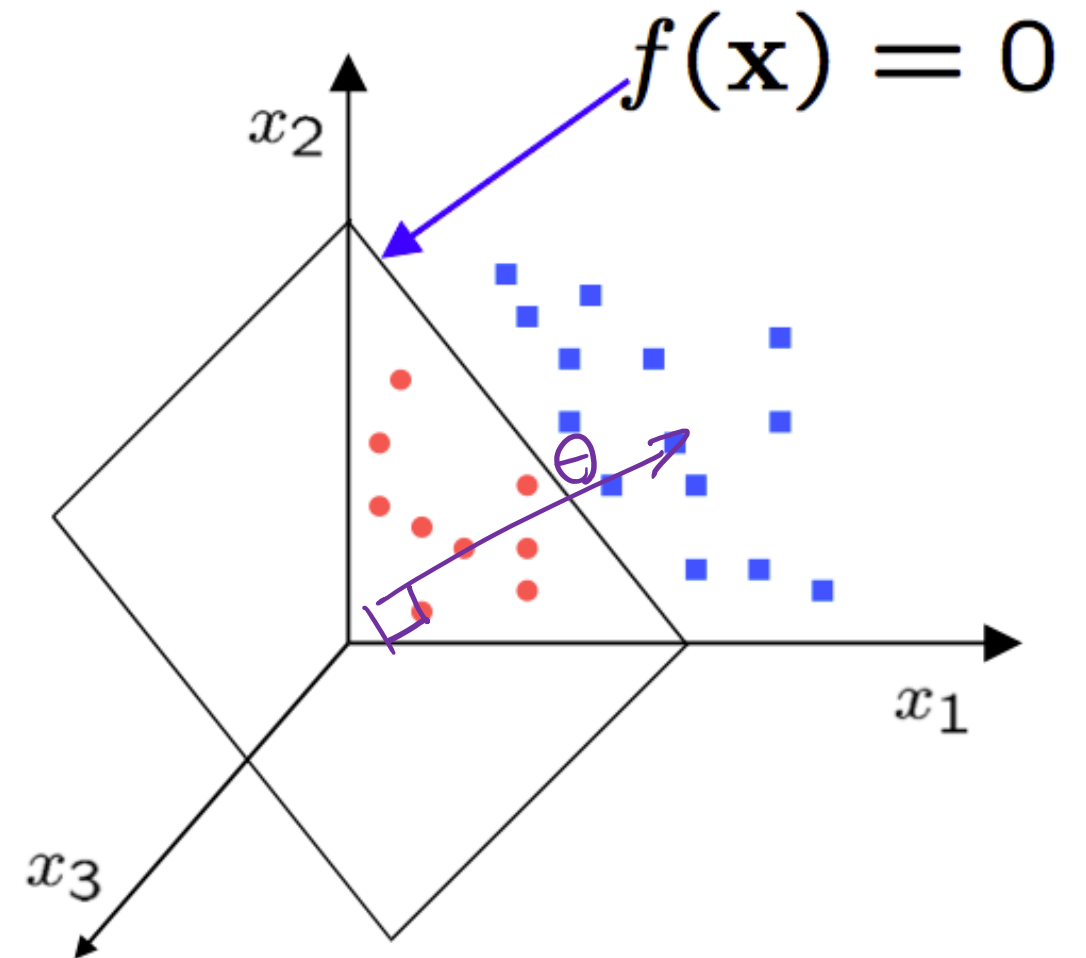
$$\text{Sign}(<0) = -1$$

$$\phi > 90$$

# Linear Classifier (higher dimension)

A linear classifier has the form

$$f(x) = x\theta + \theta_0$$



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



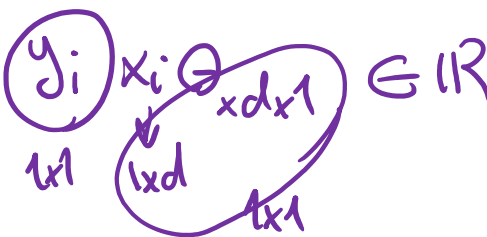
# The Perceptron Classifier

Considering  $\mathbf{x}$  is linearly separable and  $y$  has two labels of  $\{-1, 1\}$

$$f(x_i) = x_i \theta \quad \text{Bias is inside } \theta \text{ now}$$

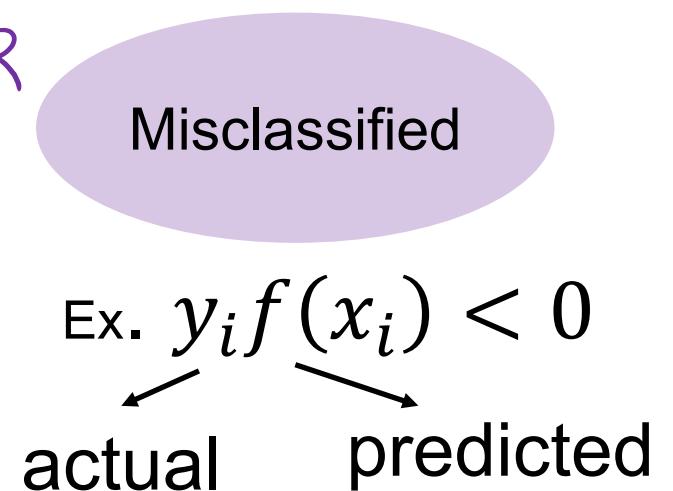
How can we separate datapoints with label 1 from datapoints with label -1 using a line?

$y_i f(x_i) < 0$



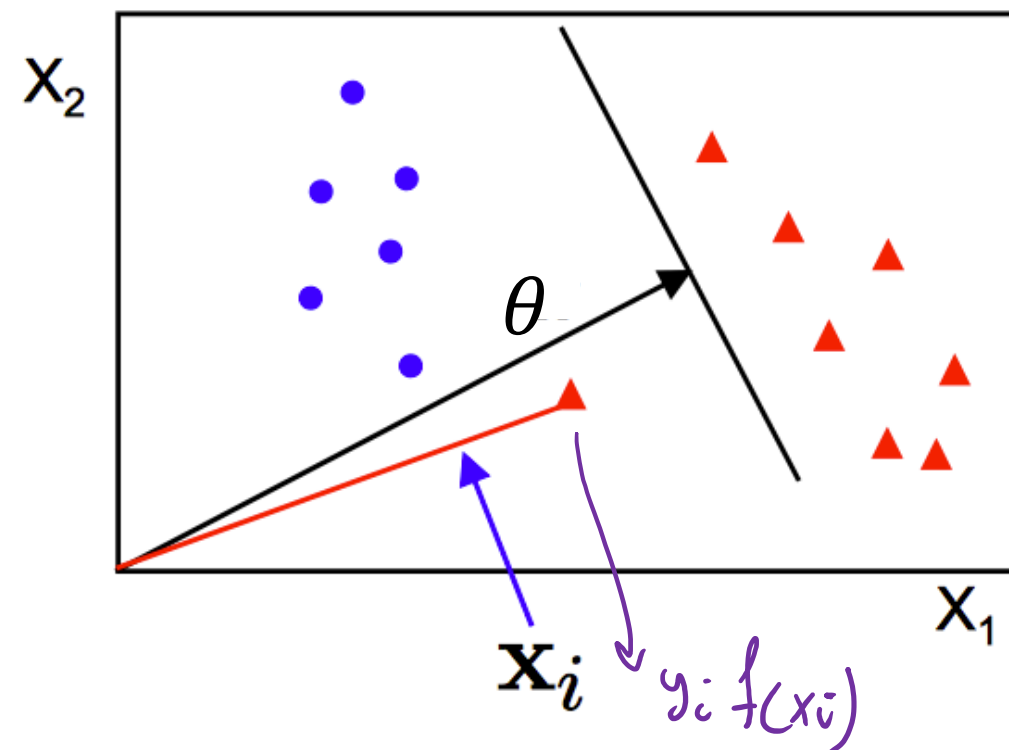
## Perceptron Algorithm:

- Initialize  $\theta = 0$
- Go through each datapoint  $\{x_i, y_i\}$ 
  - If  $x_i$  is misclassified then  $\theta^{t+1} \leftarrow \theta^t + \alpha y_i x_i$
- Until all datapoints are correctly classified

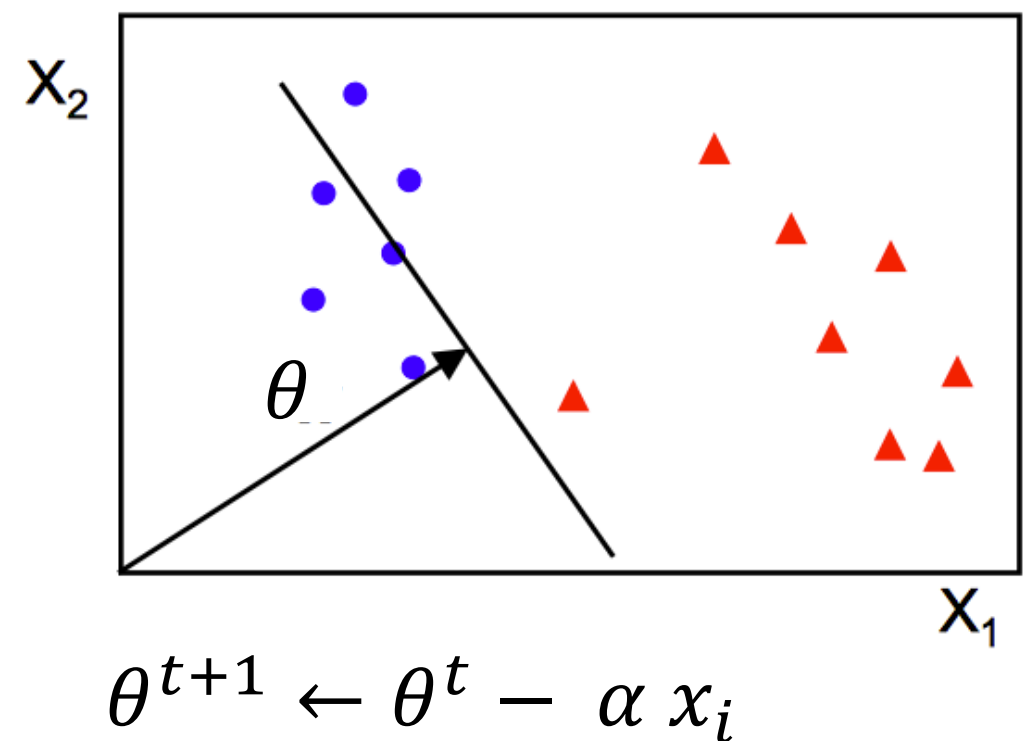


- Initialize  $\theta = 0$
- Go through each datapoint  $\{x_i, y_i\}$ 
  - If  $x_i$  is misclassified then  $\theta^{t+1} \leftarrow \theta^t + \alpha y_i x_i$
- Until all datapoints are correctly classified

before update



after update

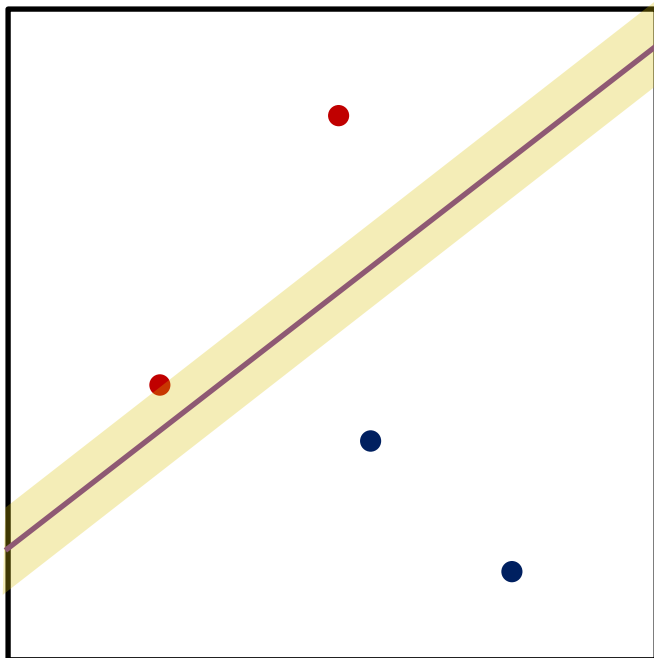


# Linear separation

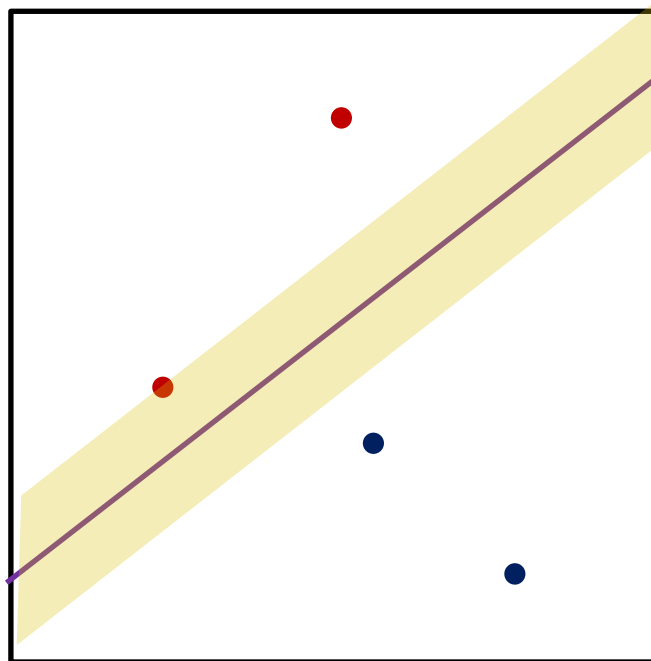
We can have different separating lines

$$\theta = [\theta_0 \ \theta_1 \dots \ \theta_d]$$

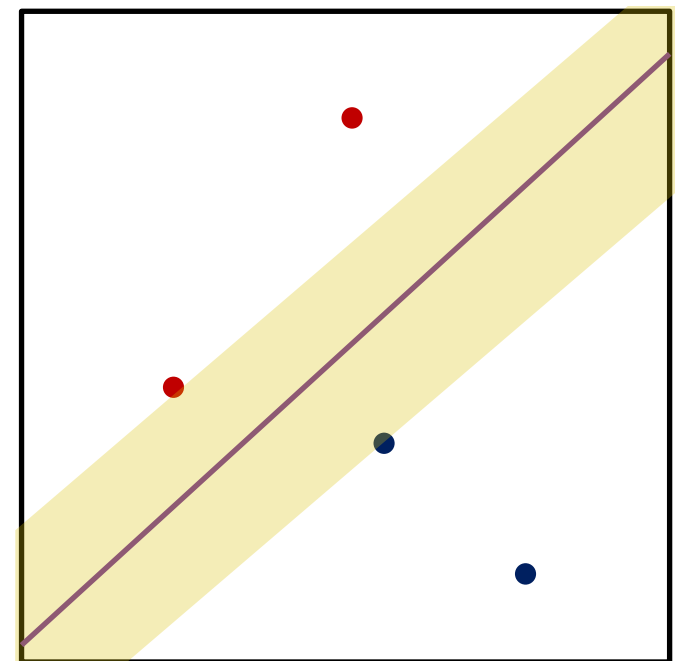
①



②



③



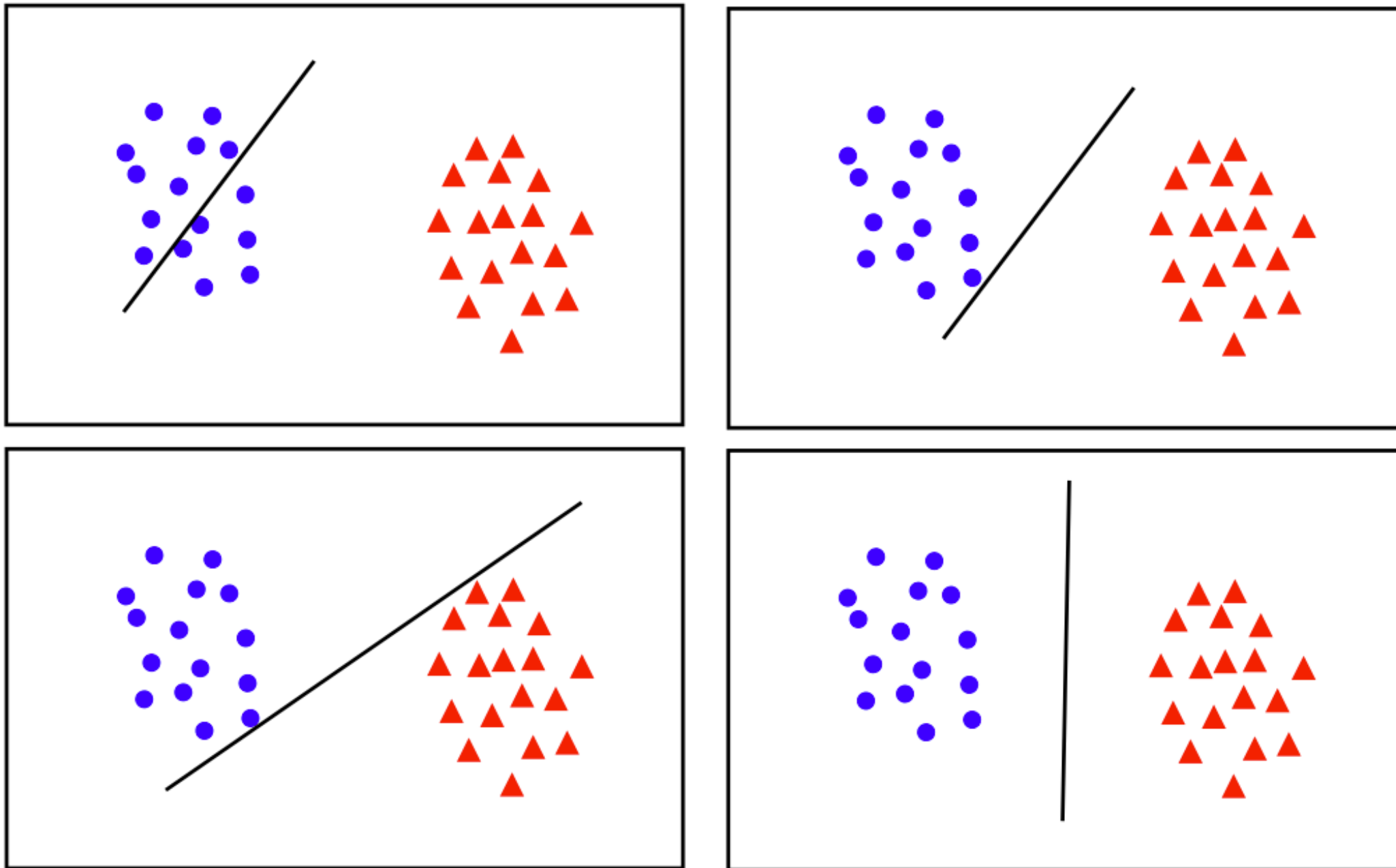
Which line is the best?

All cases, error is zero and they are linear, so they are all good for generalization.

Why is the bigger margin better?

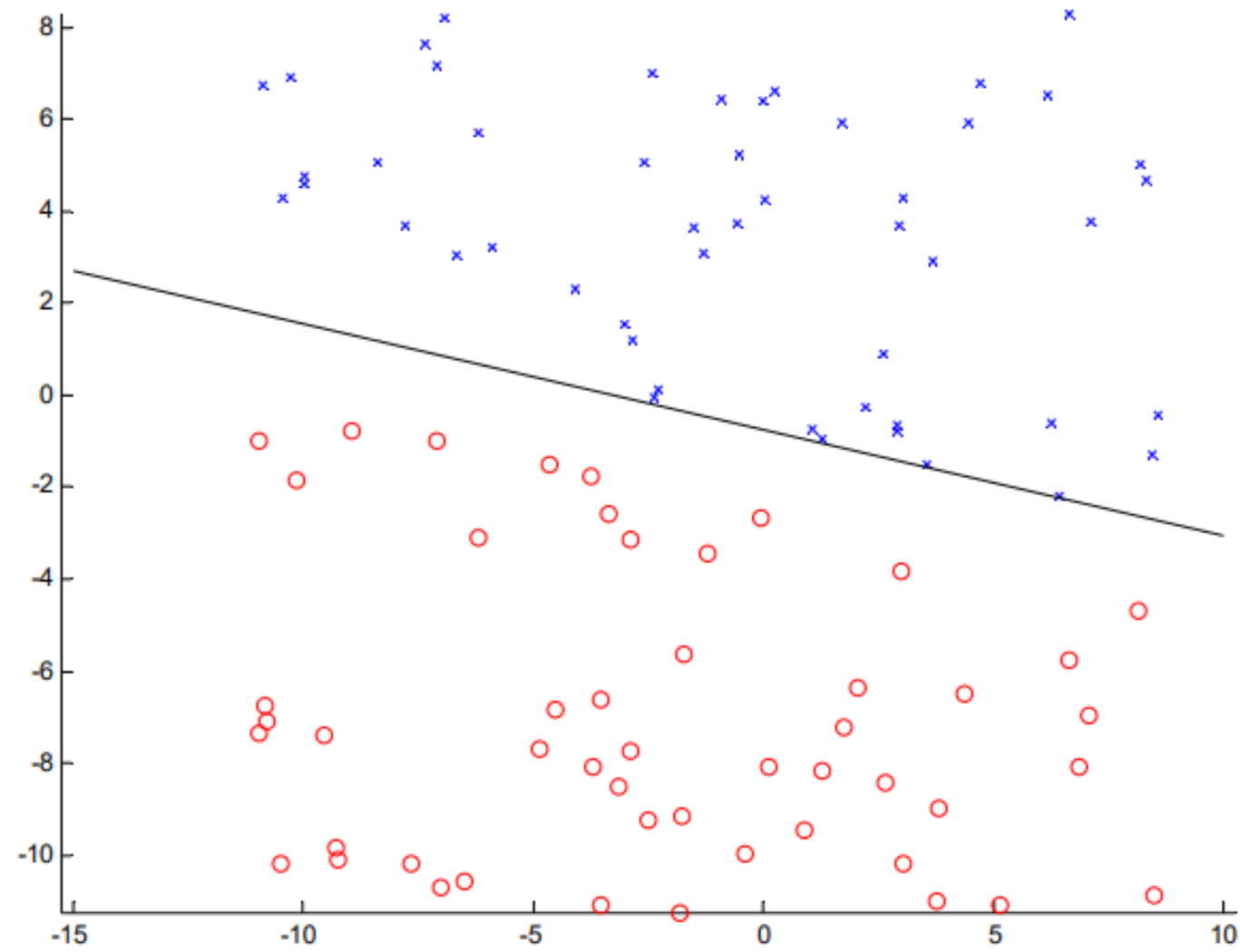
What  $\theta$  maximizes the margin?

# What is the Best $\theta$ ?





- **maximum margin** solution: most stable under perturbations of the inputs

## Perceptron example



- if the data is linearly separable, then the algorithm will converge
- convergence can be slow ...
- separating line close to training data
- we would prefer a larger margin for generalization (better generalization)

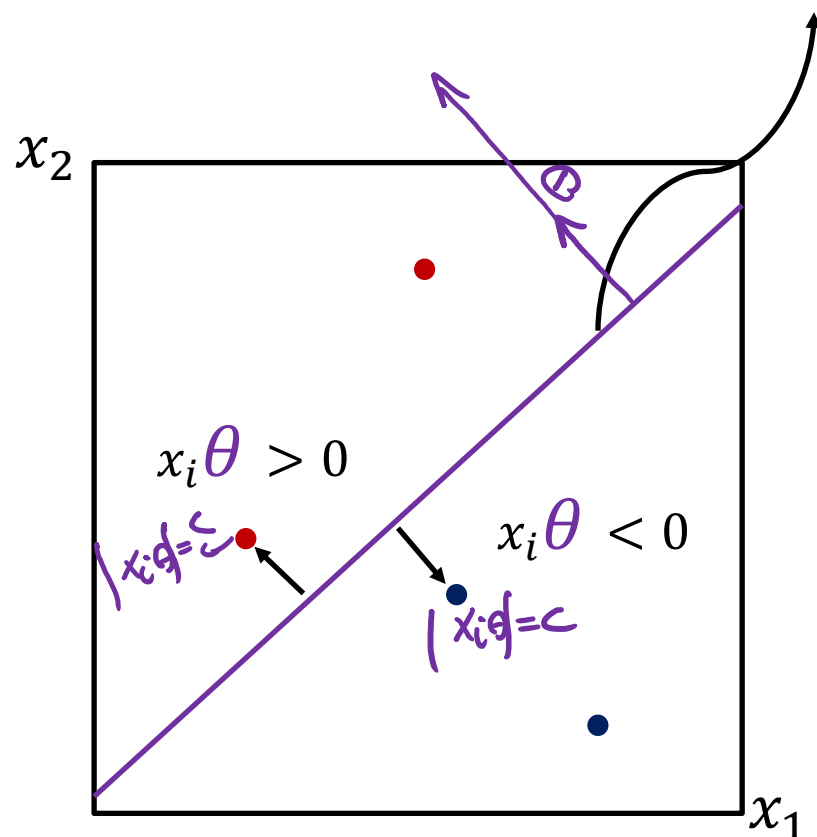
# Outline

- Precursor: Linear Classifier and Perceptron
- Support Vector Machine 
- Parameter Learning 

# Finding $\theta$ with a **fat** margin

$f(x) = x\theta = 0$  ↗ on decision line

Solution (decision boundary) of the line:  $x\theta = 0$



$x_i\theta = c$  ↕↗  $|x_i\theta| = 1$

Let  $x_i$  to be the nearest data point to the line (plane):

$$|x_i\theta| > 0$$

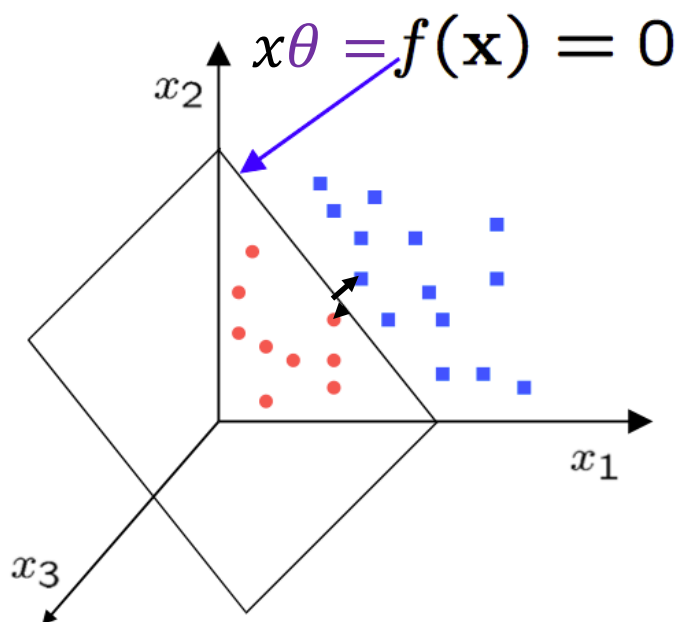
Our line solution is  $x\theta = 0$

Does it matter if I scale up or down  $\theta$  for the decision boundary?

$$|x_i\theta| = 1 \rightarrow \text{normalization}$$

Let's pull out  $\theta_0$  from  $\theta = (\theta_1, \dots, \theta_d)$  and call it be  $b$

Decision boundary would be:  $x\theta + b = 0$



# Computing the distance

The distance between  $x_i$  and the line  $x\theta + b = 0$  where  $|x_i\theta + b| = 1$

The vector  $\theta$  is perpendicular to the decision line.

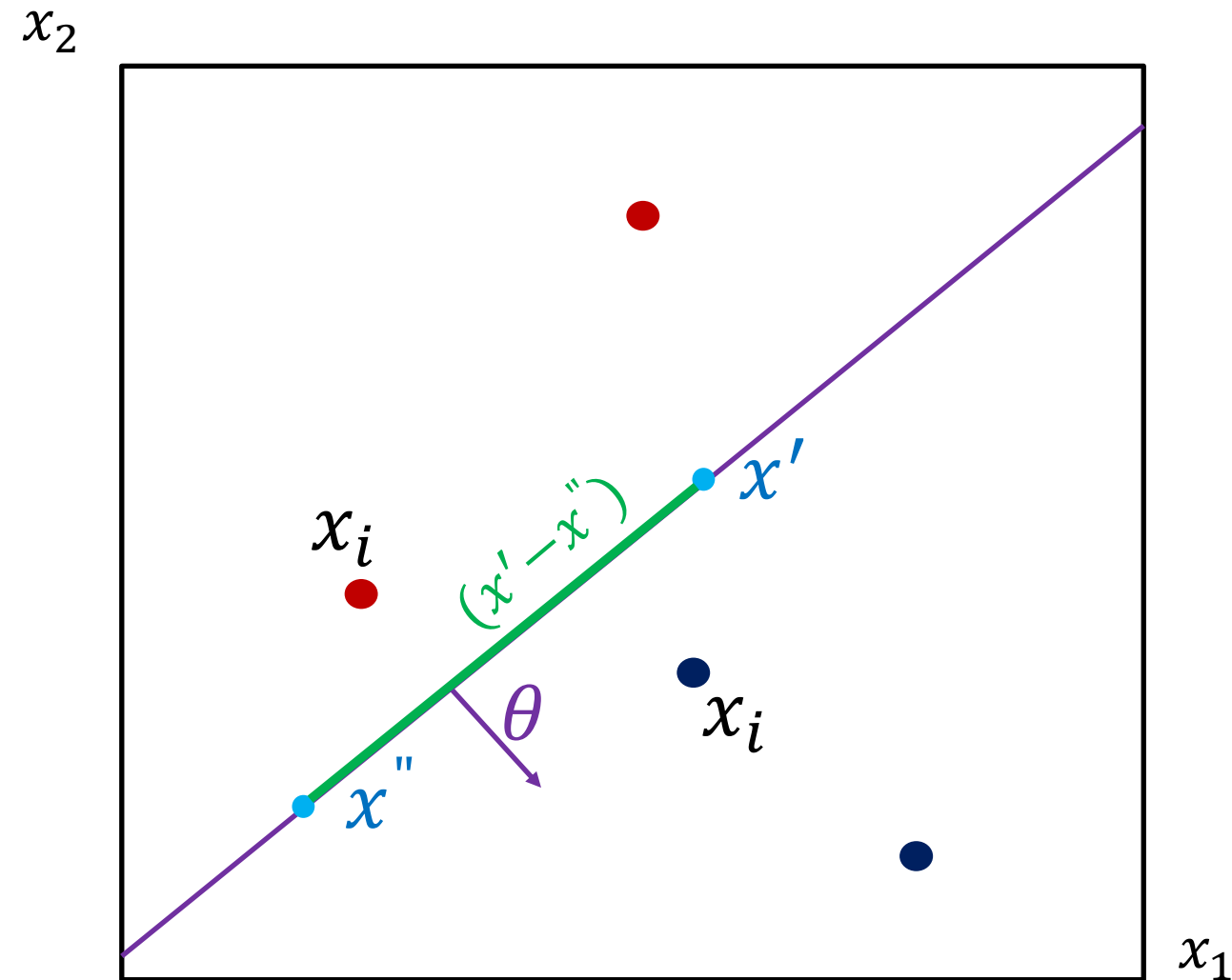
You should ask me why?

Consider  $x'$  and  $x''$  on the plane

$$x'\theta + b = 0 \quad \text{and} \quad x''\theta + b = 0$$

$$\Downarrow$$
$$x'\theta + b = x''\theta + b$$

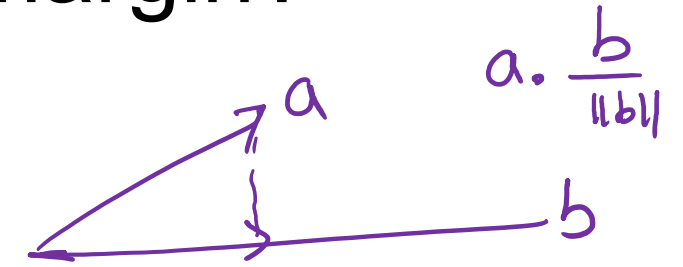
$$\Rightarrow (x' - x'')\theta = 0$$





# What is the distance of my fat margin?

What is the distance between  $x_i$  and the plane?



Let's take any point  $x$  on the line:

Distance would be projection of  $(x_i - x)$  vector on  $\theta$ .

To project the vector, we need to normalize  $\theta$  to get the unit vector.

$$\hat{\theta} = \frac{\theta}{\|\theta\|} \Rightarrow \text{distance} = |(x_i - x) \hat{\theta}| \text{ which is the dot product}$$

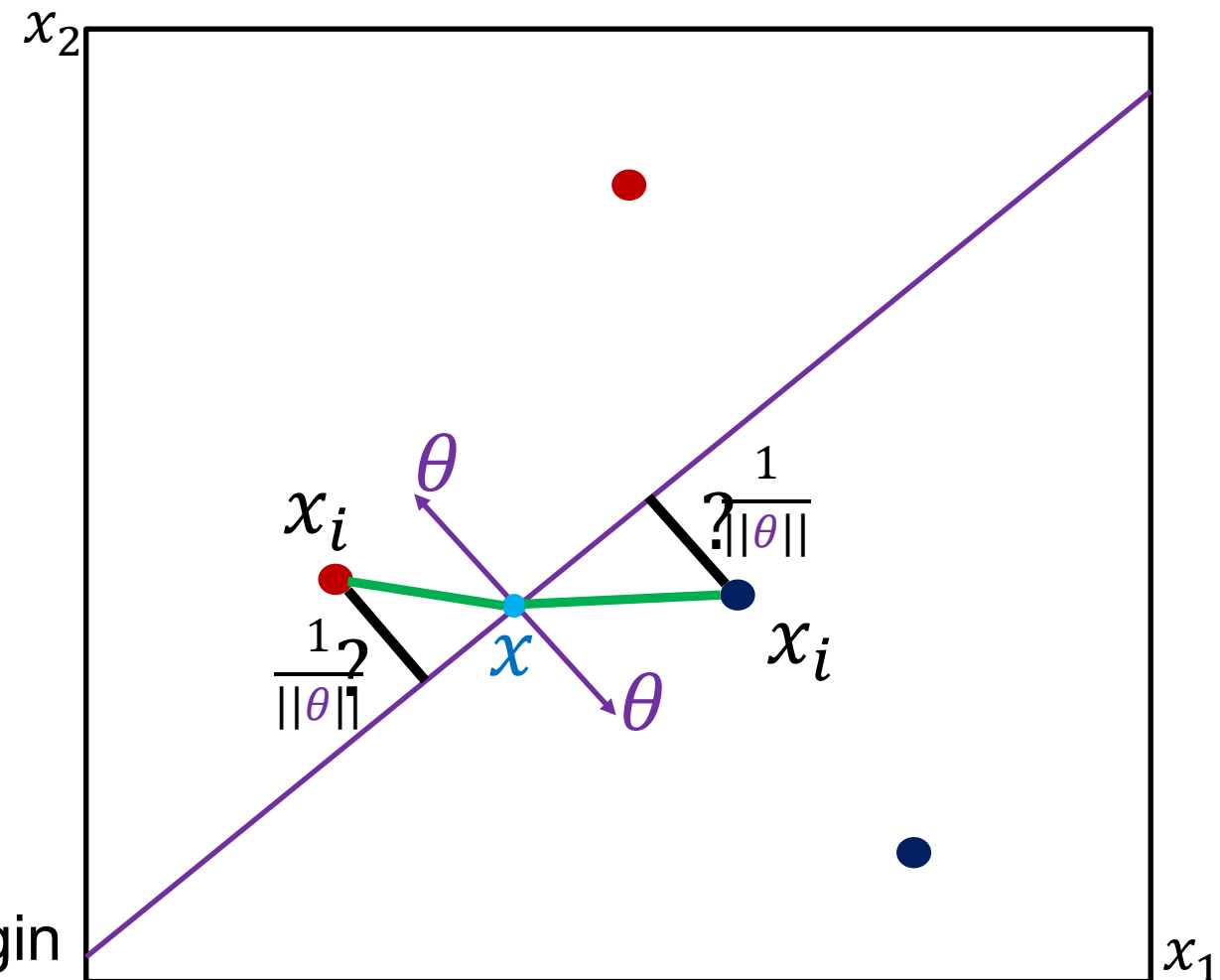
$$\text{distance} = \frac{1}{\|\theta\|} |(x_i \theta - x \theta)|$$

$$= \frac{1}{\|\theta\|} |(\underbrace{x_i \theta + b}_{\text{My constraint}} - \underbrace{x \theta + b}_{\text{A point on the decision line}})| = \frac{1}{\|\theta\|}$$

My constraint  
 $|x_i \theta + b| = 1$

A point on the  
decision line  
 $x \theta + b = 0$

The margin



Now we need to maximize the margin

Maximize  $\frac{1}{||\theta||}$

Subject to Min value of  $|x_i\theta + b| = 1 \Rightarrow \text{nearest neighbour}$   
 $i = 1, 2, \dots, N$

There is a “min” in our constraining; it can be hard to optimize this problem(non-convex form)

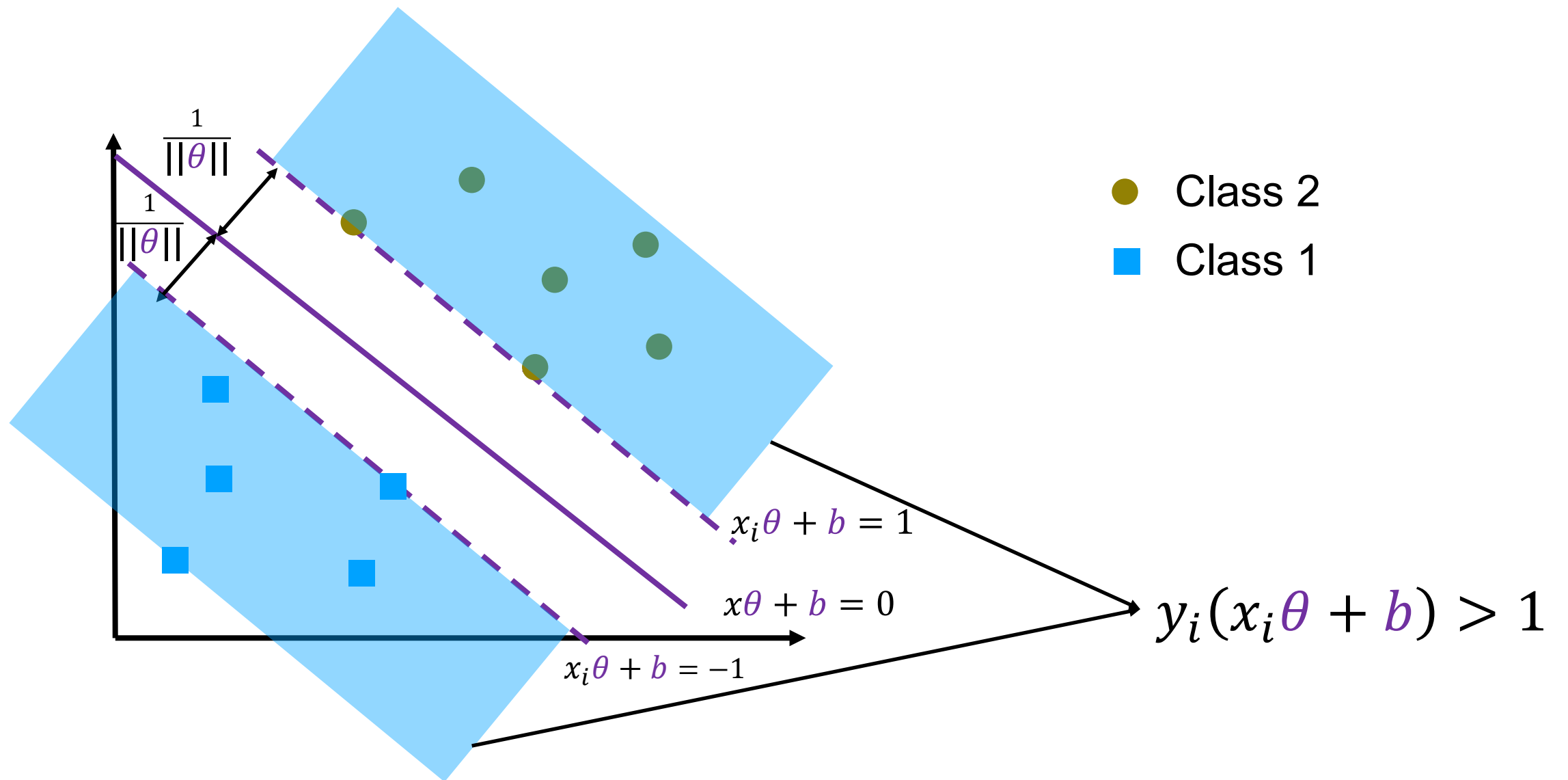
Can I write the following term to get rid of absolute value?

$$|x_i\theta + b| = y_i(x_i\theta + b) \Rightarrow \text{for a correct classification}$$

If min  $|x_i\theta + b| = 1 \Rightarrow \text{so it can be at least 1}$

Maximize  $\frac{1}{||\theta||}$

Subject to  $y_i(x_i\theta + b) \geq 1$  for  $i = 1, 2, \dots, N$



Maximize  $\frac{2}{\|\theta\|_2^2}$

If  $\theta \neq 0$ , there exists a max value

Subject to  $y_i(x_i\theta + b) \geq 1$  for  $i = 1, 2, \dots, N$

Minimize  $\frac{1}{2} \theta \theta^T$

$\|\theta\|_2^2 = \theta \theta^T$

Subject to  $y_i(x_i\theta + b) \geq 1$  for  $i = 1, 2, \dots, N$

# Constrained optimization

$$\text{Minimize } \frac{1}{2} \theta \theta^T$$

$$\text{Minimize } L(\theta, \alpha) = \frac{1}{2} \theta \theta^T - \alpha g(x)$$

$$\text{Subject to } y_i(x_i \theta + b) \geq 1 \quad \text{for } i = 1, 2, \dots, N$$

$g(x) \geq 0$

$$\theta \in \mathbb{R}^d, b \in \mathbb{R}$$

Using Lagrange method: But wait, there is an **inequality** in our constraints

We use Karush-Kuhn-Tucker (KKT) condition to deal with this problem

$$g(x) = y_i(x_i \theta + b) - 1 \quad \alpha = \text{lagrange multiplier}$$

We need to optimize

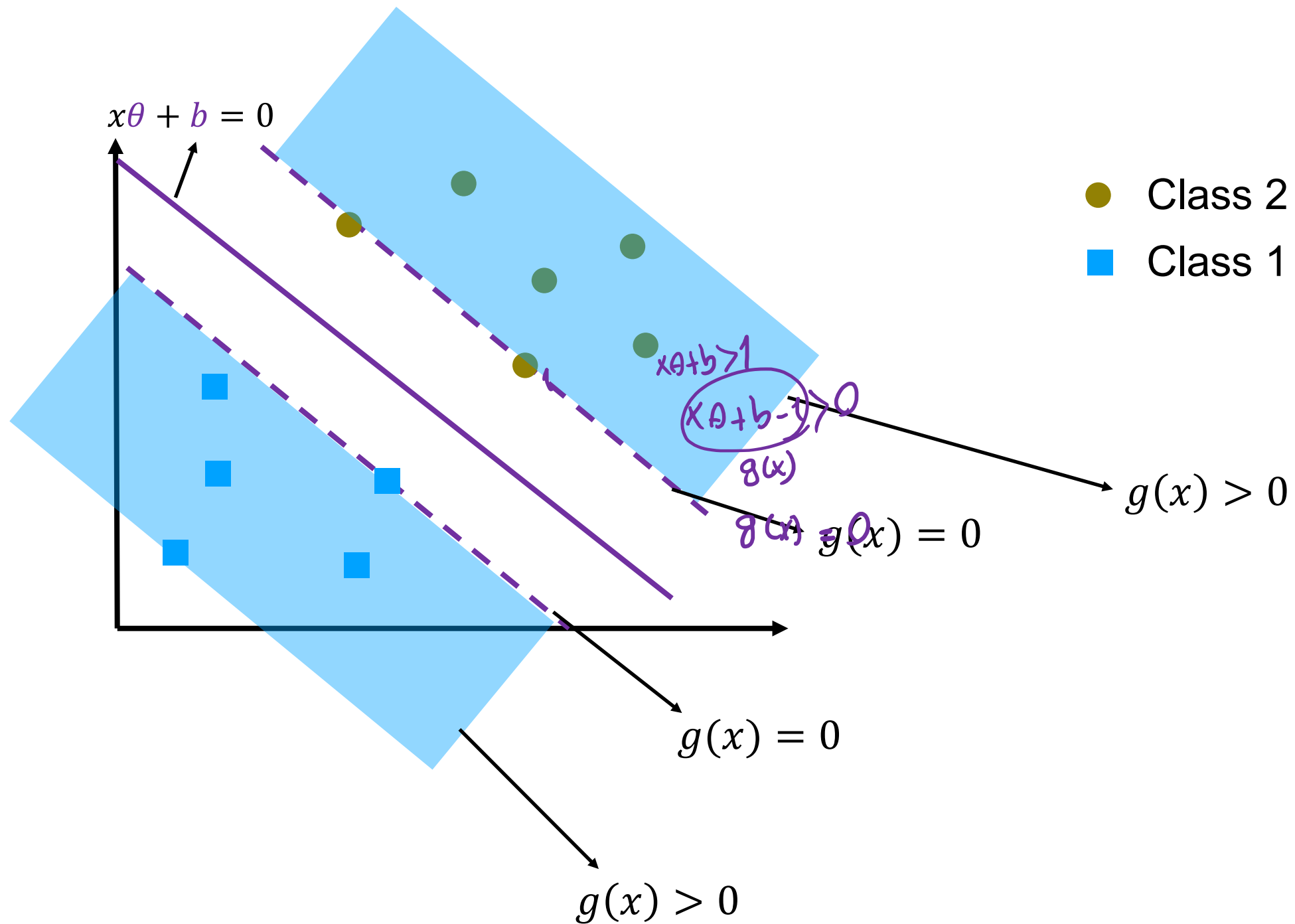
1) Stationary

$\theta, b, \text{ and } \alpha$

2)  $g(x) \geq 0$  Primal feasibility

3)  $\alpha \geq 0$  Dual feasibility

4)  $g(x) \alpha = 0$  Complementary slackness  $\Rightarrow \begin{cases} g(x) > 0, & \alpha = 0 \\ \alpha > 0, & g(x) = 0 \end{cases}$



$$\forall_i (x_i\theta + 1) = 1$$

$$g(x) = y_i(x_i\theta + b) - 1 = 0$$

$$3) \quad g(x)\alpha = 0 \quad \text{Complementary slackness} \Rightarrow \begin{cases} g(x) > 0, & \alpha = 0 \\ \alpha > 0, & g(x) = 0 \end{cases}$$

# Lagrange formulation

$$\text{Minimize } \frac{1}{2} \theta \theta^T \quad \text{s.t.} \quad y_i(x_i \theta + b) - 1 \geq 0$$

→ Primal form

$$\mathcal{L}(\theta, b, \alpha) = \frac{1}{2} \theta \theta^T - \sum_{i=1}^N \alpha_i (y_i(x_i \theta + b) - 1)$$

Minimize w.r.t  $\theta$  and  $b$  and maximize w.r.t each  $\alpha_i \geq 0$

$$\nabla_{\theta} \mathcal{L}(\theta, b, \alpha) = \theta - \sum_{i=1}^N \alpha_i y_i x_i = 0$$

$$\nabla_b \mathcal{L}(\theta, b, \alpha) = - \sum_{i=1}^N \alpha_i y_i = 0$$

$$\theta_{1 \times d} = \sum \alpha_i y_i x_i$$

$$\theta = \sum_{i=1}^N \alpha_i y_i x_i$$

$$\sum_{i=1}^N \alpha_i y_i = 0$$

Let's substitute these in the Lagrangian:

$$\mathcal{L}(\theta, b, \alpha) = \frac{1}{2} \theta \theta^T - \sum_{i=1}^N \alpha_i (y_i (x_i \theta + b) - 1)$$

$$\mathcal{L}(\theta, b, \alpha) = \sum_{i=1}^N \alpha_i + \frac{1}{2} \theta \theta^T - \sum_{i=1}^N \alpha_i (y_i (x_i \theta + b))$$

$\Theta^T$   
 $-\left[ \sum \alpha_i y_i x_i \theta + b \sum \alpha_i y_i \right]$

$$\begin{aligned} \mathcal{L}(\theta, b, \alpha) &= \sum_{i=1}^N \alpha_i + \frac{1}{2} \theta \theta^T - \sum_{i=1}^N \alpha_i (y_i (x_i \theta)) = \sum_{i=1}^N \alpha_i + \frac{1}{2} \theta \theta^T - \theta \theta^T = \\ &= \sum_{i=1}^N \alpha_i - \frac{1}{2} \theta \theta^T \end{aligned}$$

$$\theta = \sum_{i=1}^N \alpha_i y_i x_i$$

$$\sum_{i=1}^N \alpha_i y_i = 0$$

$$\mathcal{L}(\theta, b, \alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \theta \theta^T$$

↗ Dual form

$$\mathcal{L}(\theta, b, \alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j x_i x_j^T$$

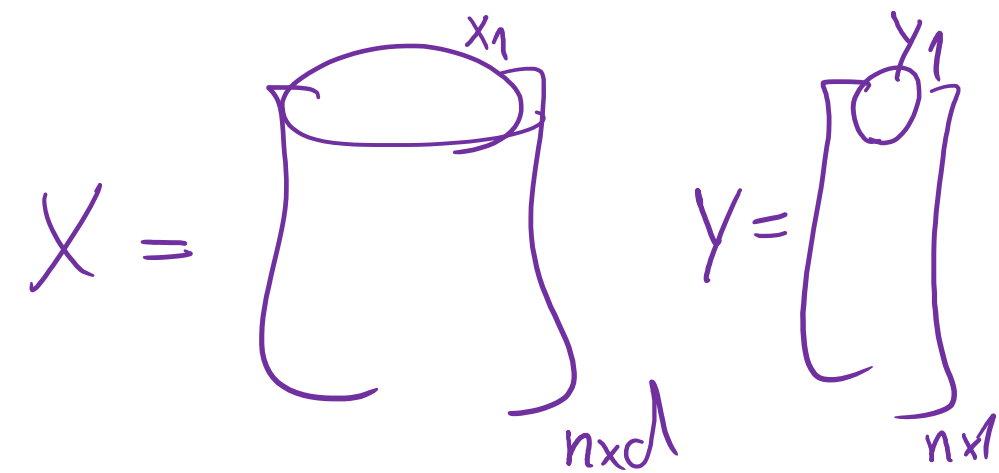
maximize w.r.t each  $\alpha_i \geq 0$  for  $i = 1, \dots, N$

and

$$\sum_{i=1}^N \alpha_i y_i = 0$$



# The solution – quadratic programming



$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j x_i x_j^T$$

Quadratic programming packages usually use “min”

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j x_i x_j^T - \sum_{i=1}^N \alpha_i$$

$$\min_{\alpha} \frac{1}{2} \alpha^T \begin{bmatrix} \underbrace{y_1 y_1 x_1 x_1^T}_{1 \times d} & y_1 y_2 x_1 x_2^T & \dots & y_1 y_N x_1 x_N^T \\ y_2 y_1 x_2 x_1^T & y_2 y_2 x_2 x_2^T & \dots & y_2 y_N x_2 x_N^T \\ \dots & \dots & \dots & \dots \\ y_N y_1 x_N x_1^T & y_N y_2 x_N x_2^T & \dots & y_N y_N x_N x_N^T \end{bmatrix}_{N \times N} \alpha + (-I^T) \alpha$$

$$\min_{\alpha} \frac{1}{2} \alpha^T \underbrace{\begin{bmatrix} y_1 y_1 x_1 x_1^T & y_1 y_1 x_1 x_2^T & \dots & y_1 y_N x_1 x_N^T \\ y_2 y_1 x_2 x_1^T & y_2 y_2 x_2 x_2^T & \dots & y_2 y_N x_2 x_N^T \\ \dots & \dots & \dots & \dots \\ y_N y_1 x_N x_1^T & y_N y_2 x_n x_2^T & \dots & y_N y_N x_N x_N^T \end{bmatrix}}_{\text{Quadratic coefficients}} \alpha + \underbrace{(-I^T) \alpha}_{\text{Linear term}}$$

**Subject to**  $\underbrace{\sum_{i=1}^N \alpha_i y_i = y^T \alpha = 0}_{\text{Linear equality constraint}}$

Pass these to a quadratic programming package

$$\text{lower bound}(0) \leq \alpha \leq \text{upper bound}(\infty)$$

$\alpha \geq 0$

$$\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - 1^T \alpha \quad \text{subject to} \quad y^T \alpha = 0; \alpha \geq 0$$

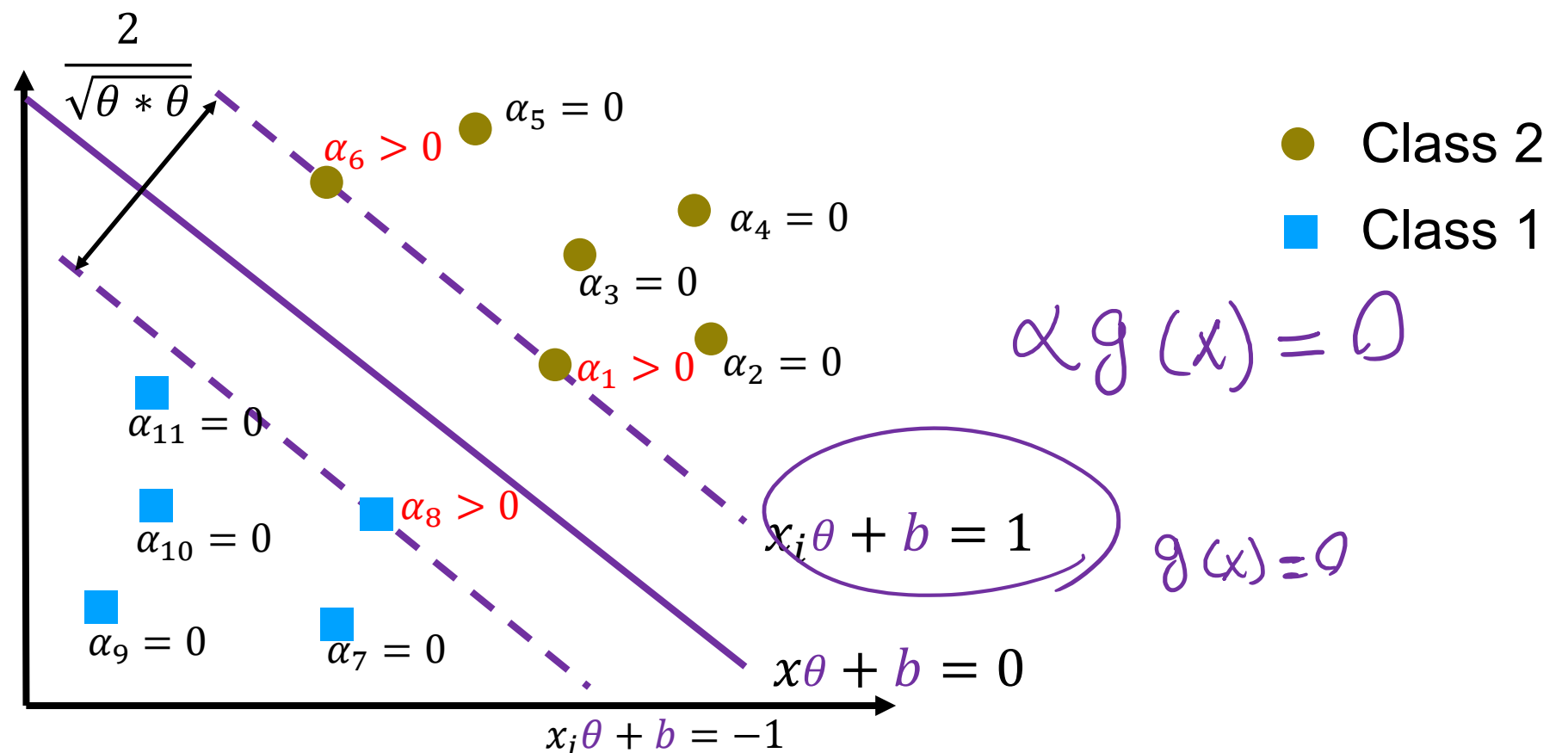
Quadratic programming will give us  $\alpha$

Solution:  $\alpha = \alpha_1, \dots, \alpha_N$

KKT condition ( $\alpha_i g_i(\theta) = 0$ ):  $\alpha_i (y_i (x_i \theta + b) - 1) = 0$

$$(y_i (x_i \theta + b) - 1) > 0 \Rightarrow \alpha_i = 0$$

$$(y_i (x_i \theta + b) - 1) = 0 \Rightarrow \alpha_i > 0 \Rightarrow x_i \text{ is a support vector}$$



# Training

$$\theta = \sum_{i=1}^N \alpha_i y_i x_i$$

No need to go over all datapoints

$$\rightarrow \theta = \sum_{x_i \text{ in } SV} \alpha_i y_i x_i$$

and for  $b$  pick any support vector and calculate:  $y_i(x_i \theta + b) = 1$

# Testing

For a new test point  $s$

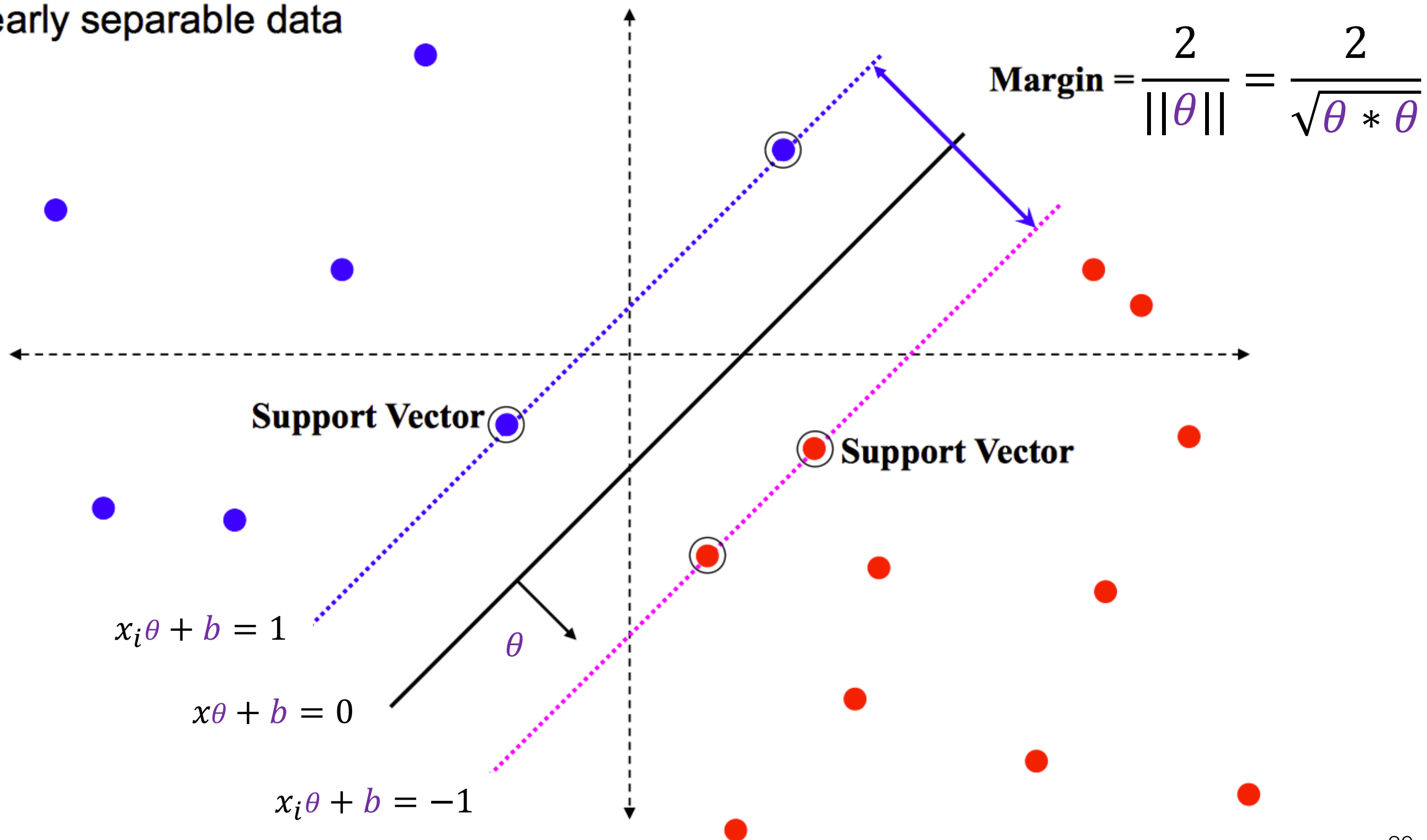
Compute:

$$\underbrace{s\theta + b}_{\text{Primal form}} = \underbrace{\sum_{x_i \text{ in } SV} \alpha_i y_i x_i s^T + b}_{\text{Dual form}}$$

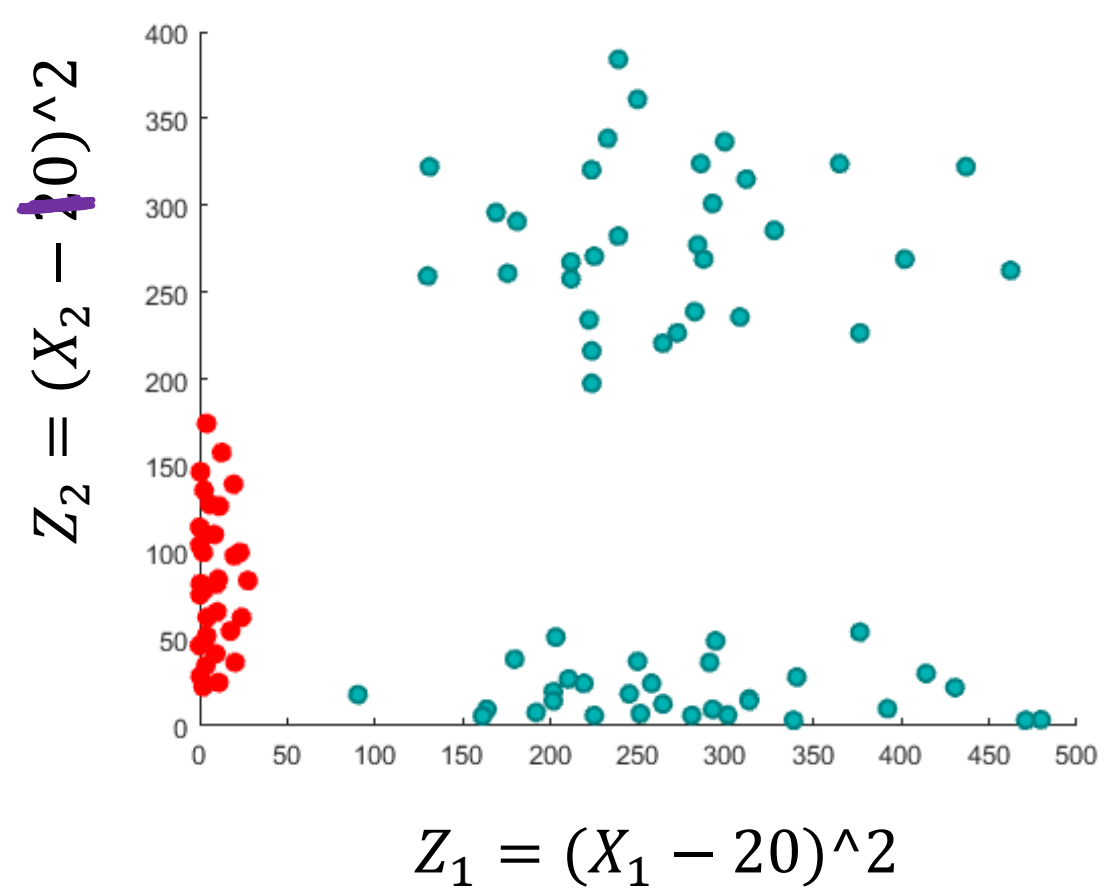
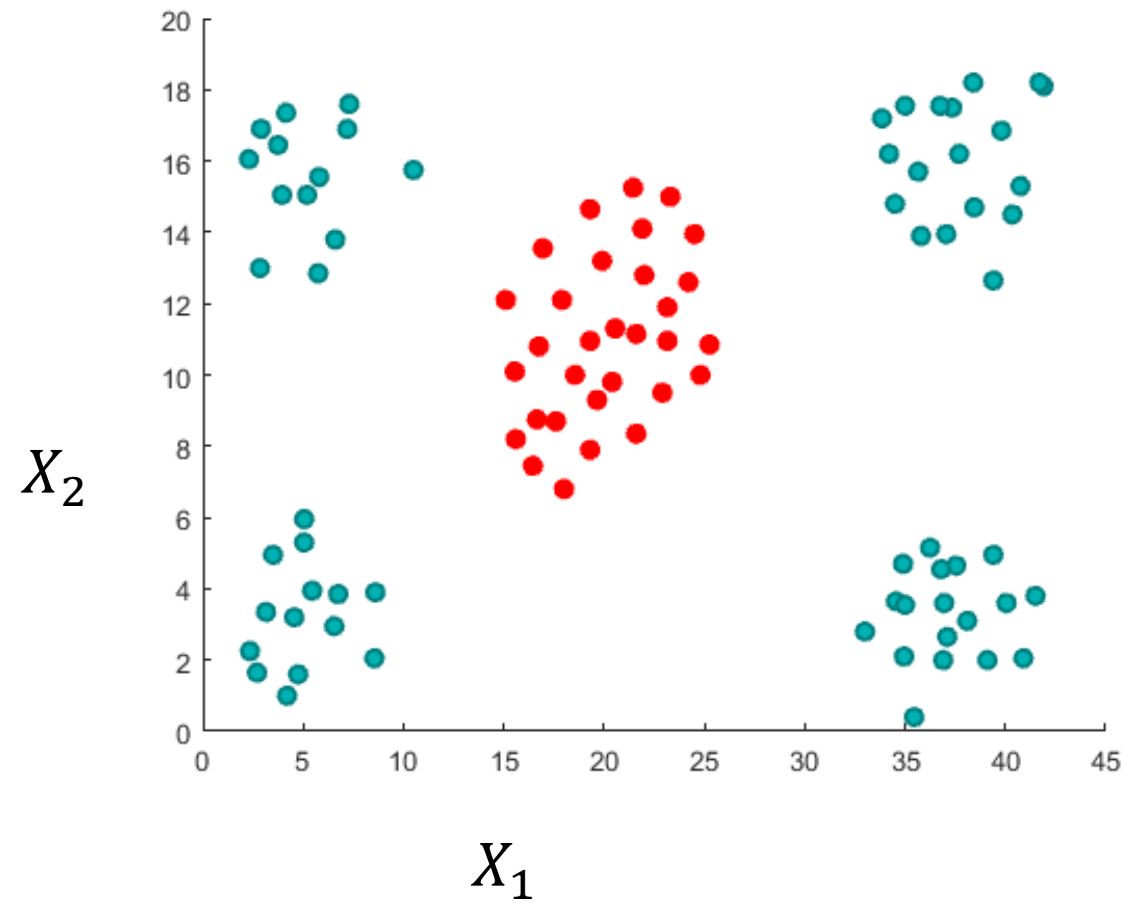
Classify  $s$  as class 1 if the result is positive, and class 2 otherwise

# Geometric Interpretation

linearly separable data



# From $x$ to $z$ space

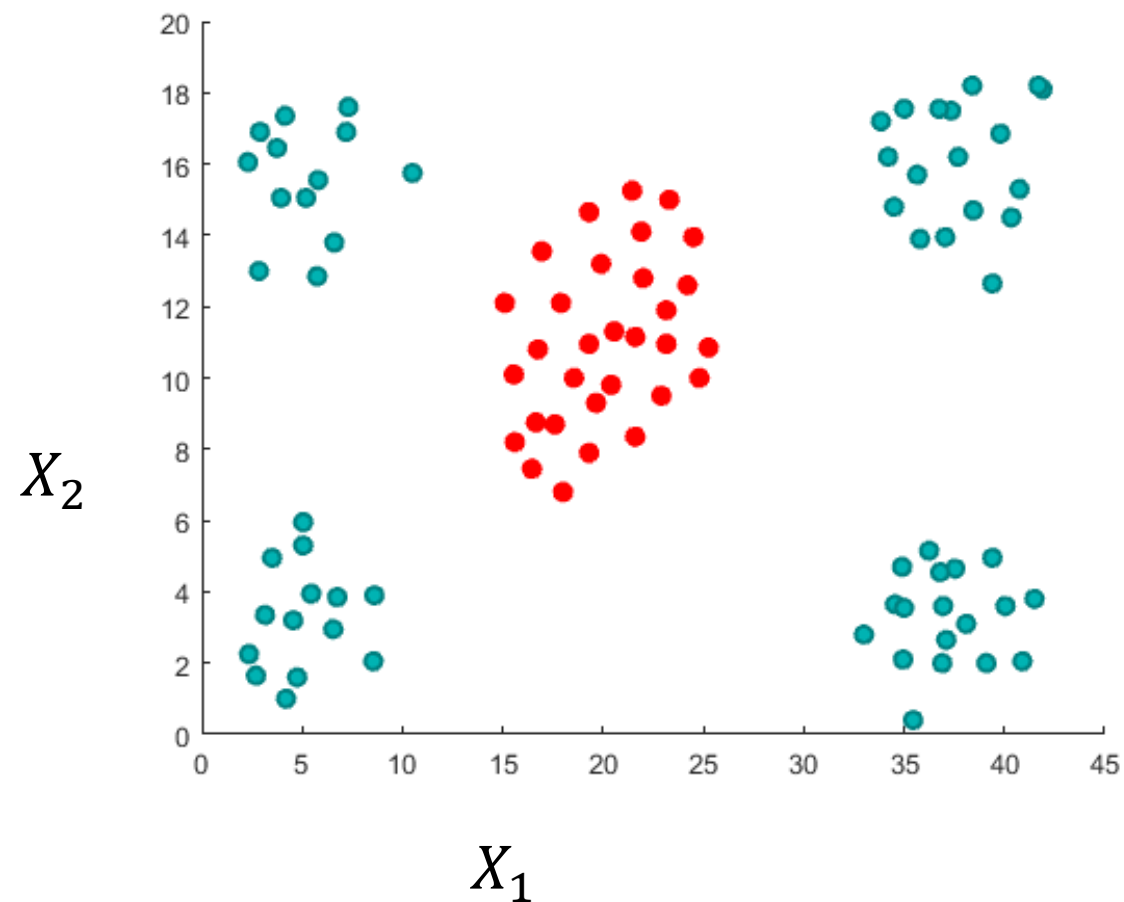


$$X \xrightarrow{(X_1^2, X_2^2)} Z$$

In  $x$  space

$$X_{n \times d} \quad (XX^T)_{n \times n}$$

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j \mathbf{x}_i \mathbf{x}_j^T$$

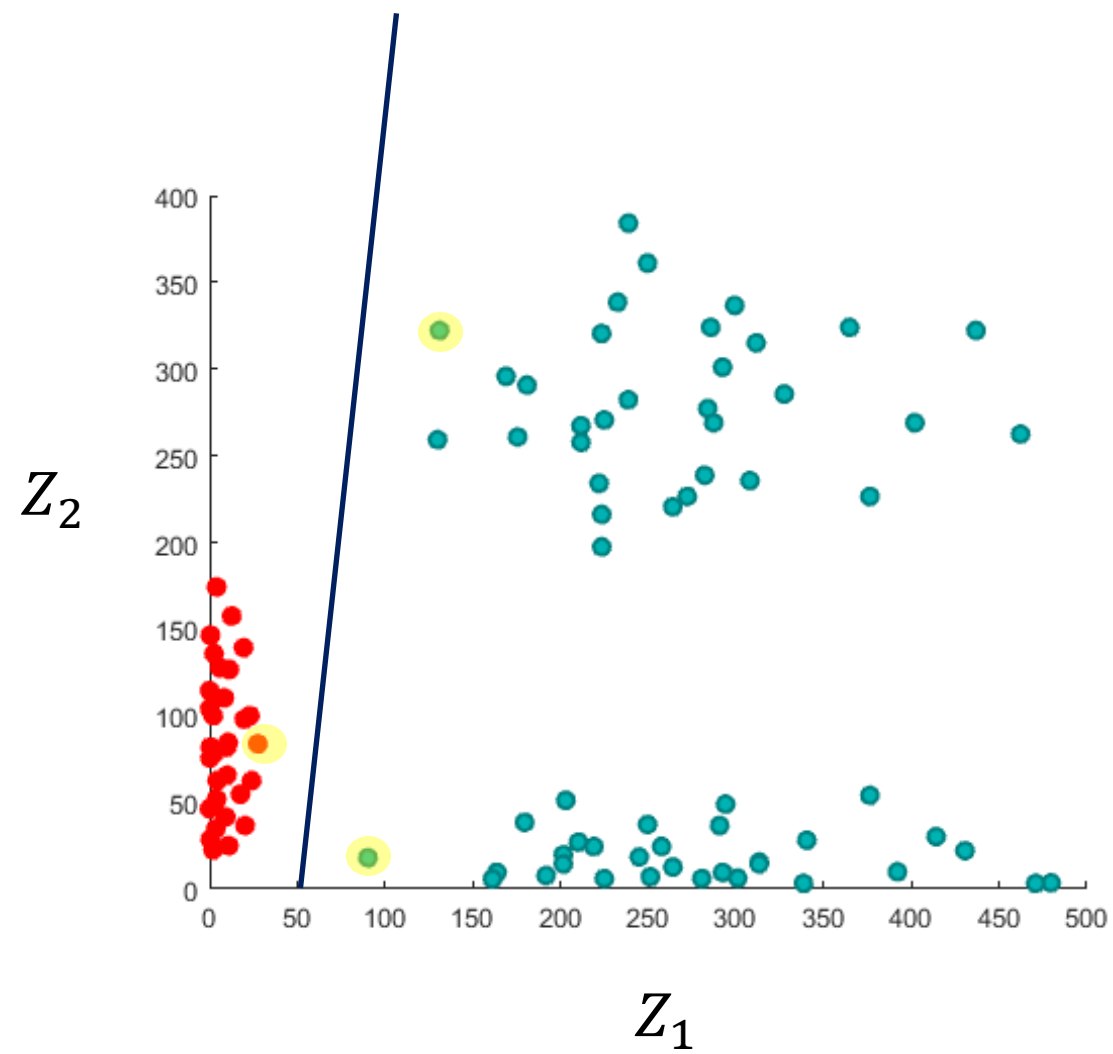


let's say  $\mathbf{x}$  is  $n \times d$   
 $\mathbf{x}\mathbf{x}^T$  will be  $n \times n$

If I add millions of  
dimensions to  $\mathbf{x}$ , would  
it affect the final size of  
 $\mathbf{x}\mathbf{x}^T$ ?

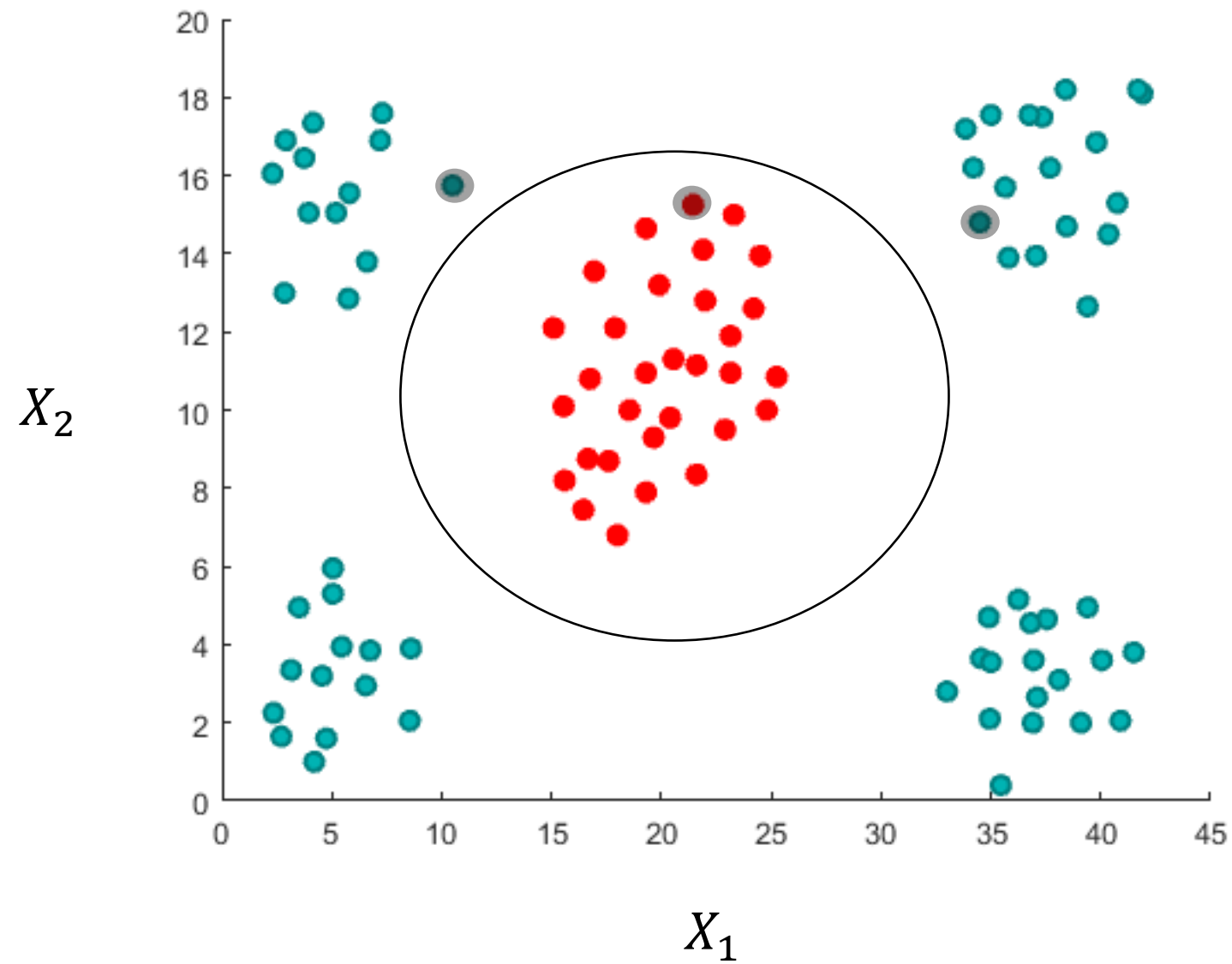
In  $z$  space

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N y_i y_j \alpha_i \alpha_j \mathbf{z}_i \mathbf{z}_j^T$$





In  $x$  space, they are called pre-images of support vectors



# Take-Home Messages

- Linear Separability
- Perceptron
- SVM: Geometric Intuition and Formulation