CS 360: Machine Learning

Prof. Sara Mathieson

Fall 2020

HAVERFORD COLLEGE

Outline: optional material on SVMs

- Recap Perceptron (+ Handout 13)
- Support Vector Machines (SVMs) overview
- Extensions of SVMs
- SVMs meta-optimization process (+ Handout 14)

Outline: optional material on SVMs

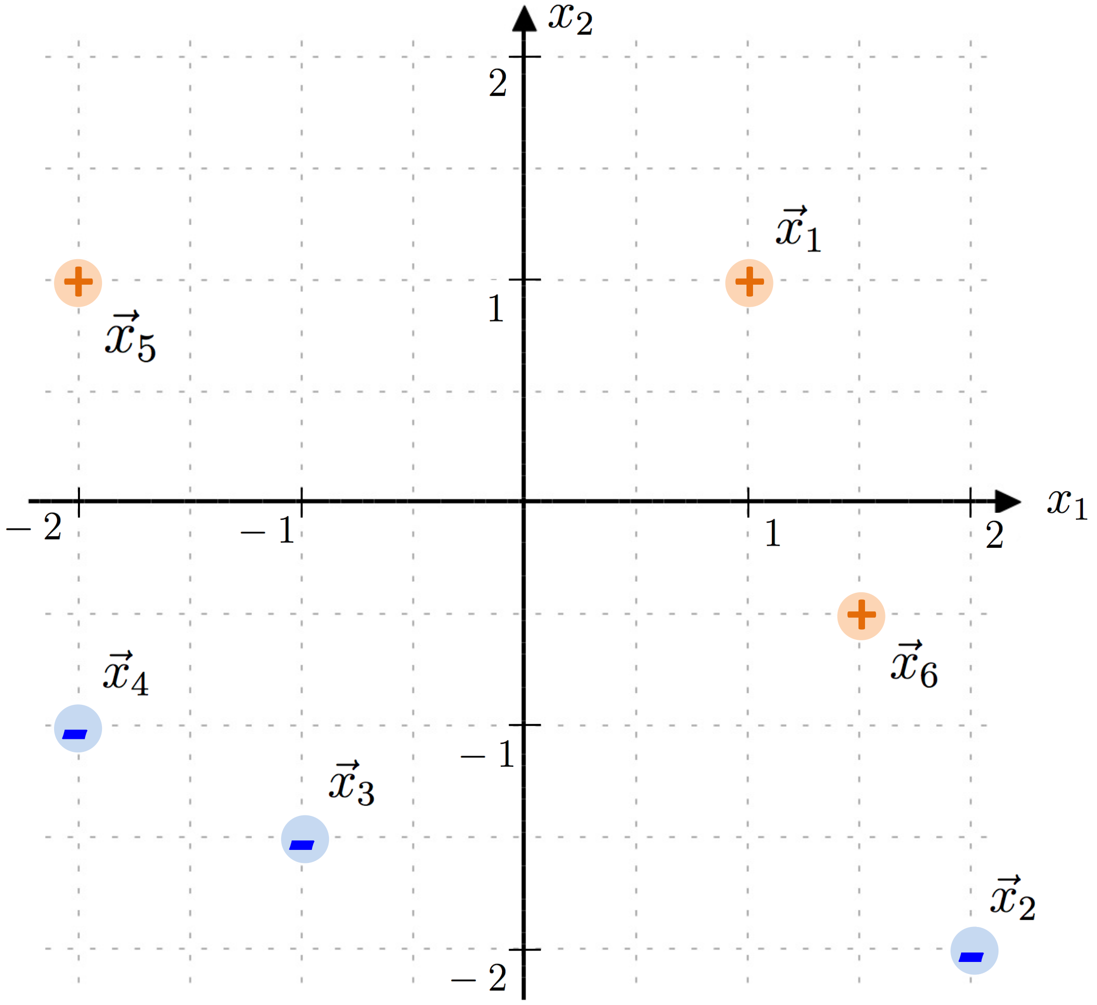
- Recap Perceptron (+ Handout 13)
- Support Vector Machines (SVMs) overview
- Extensions of SVMs
- SVMs meta-optimization process (+ Handout 14)

Handout 13 example

Initial values:

$$\alpha = 0.2$$

$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$



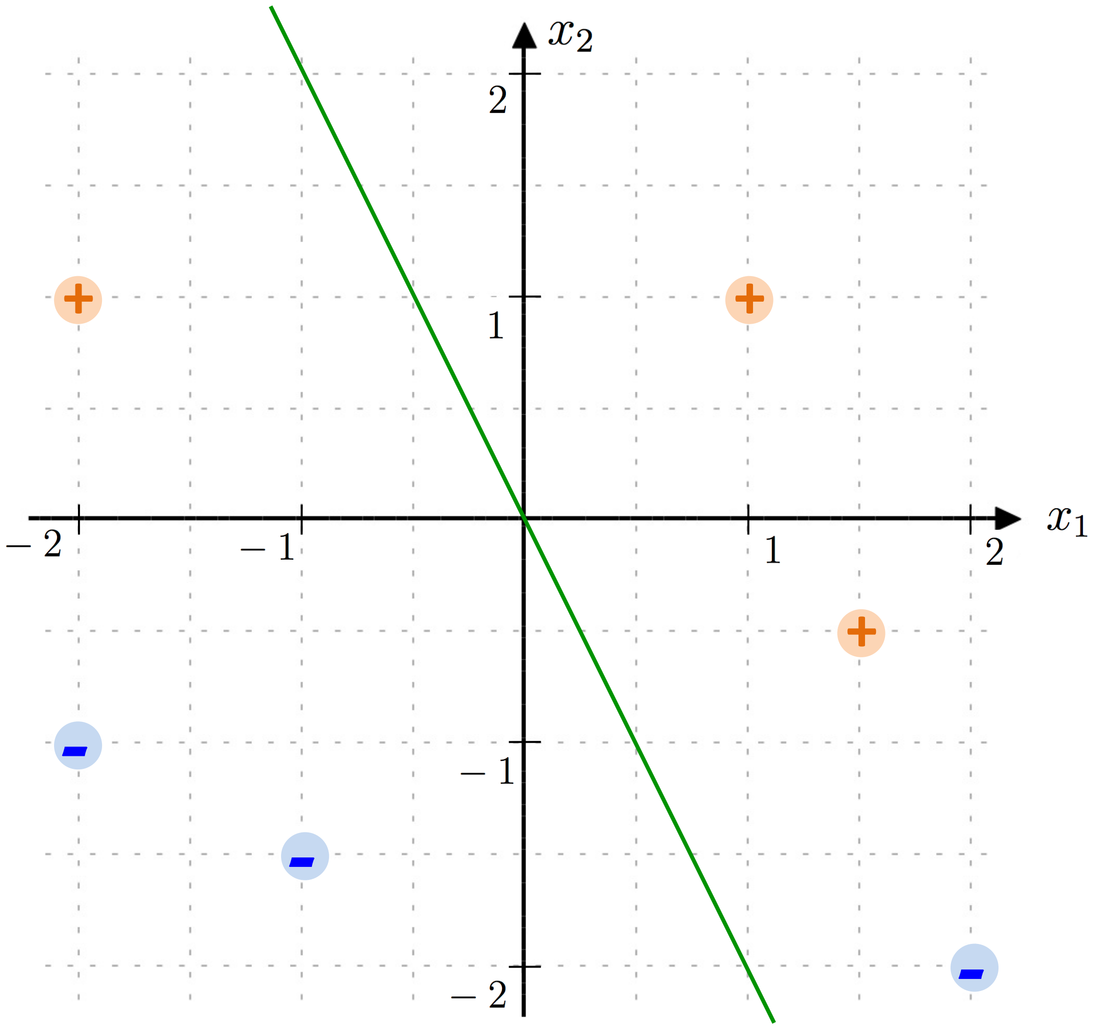
Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 13 example

Initial values:

$$\alpha = 0.2$$

$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 13 example

$$\alpha = 0.2$$

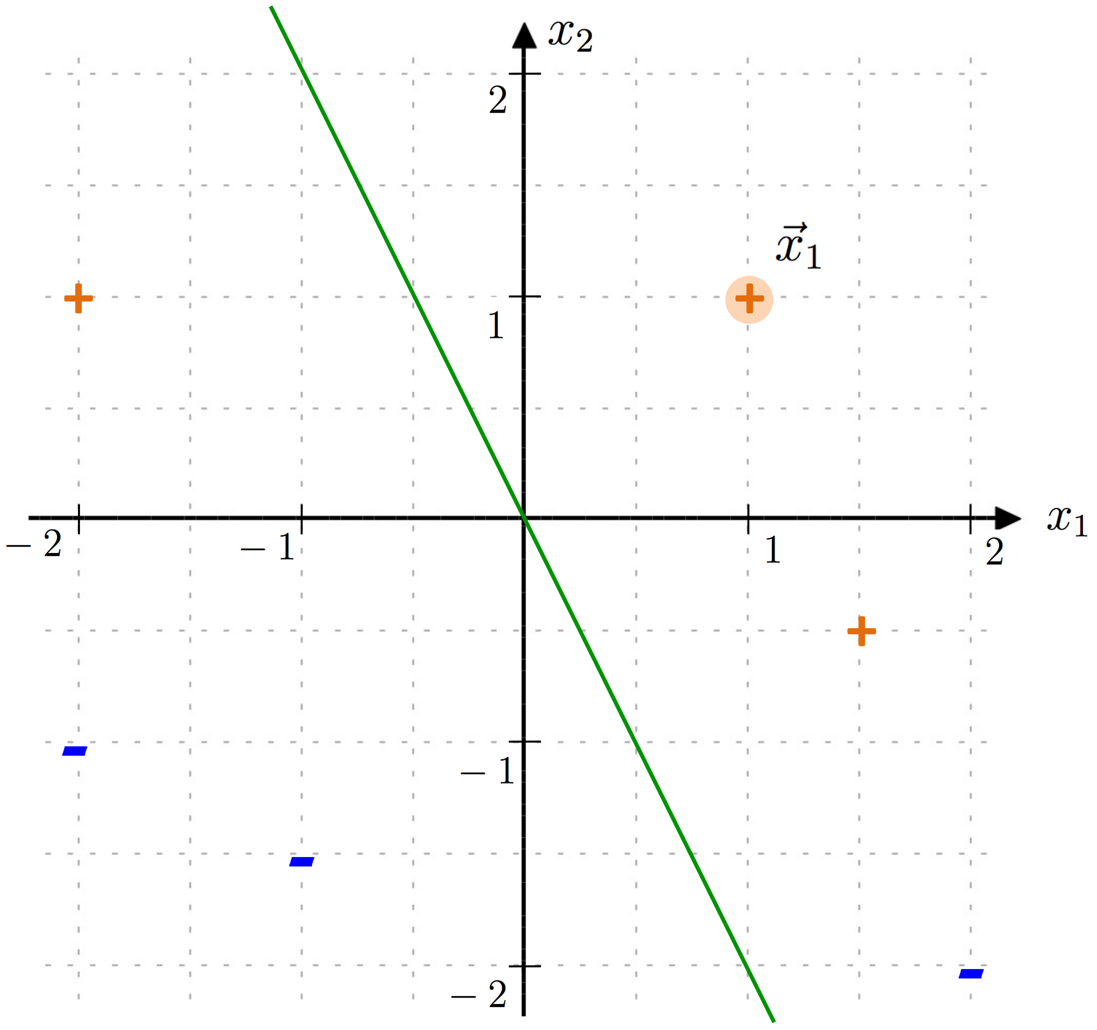
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 1:

$$\vec{x}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_1 > 0$$

Correct classification, no action



Handout 13 example

$$\alpha = 0.2$$

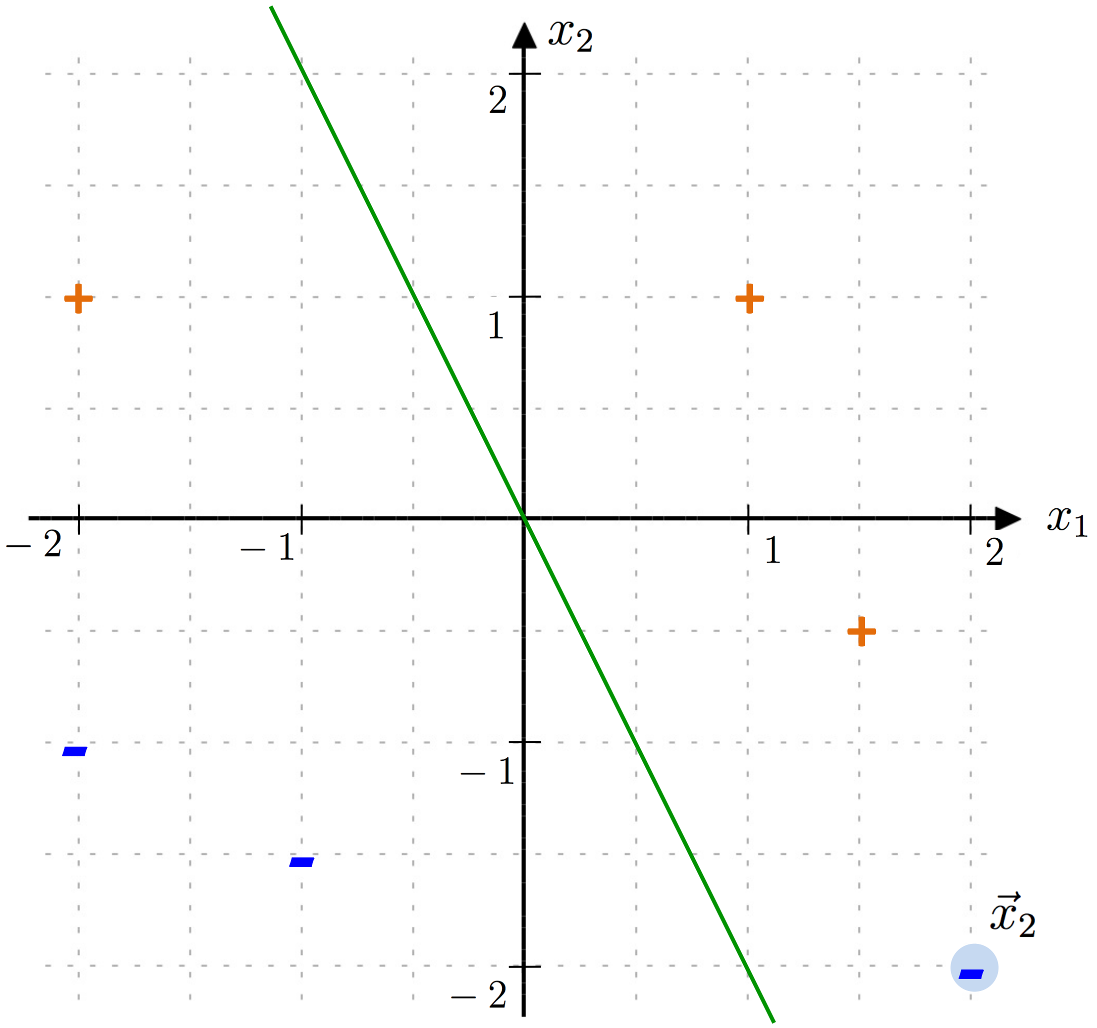
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 13 example

$$\alpha = 0.2$$

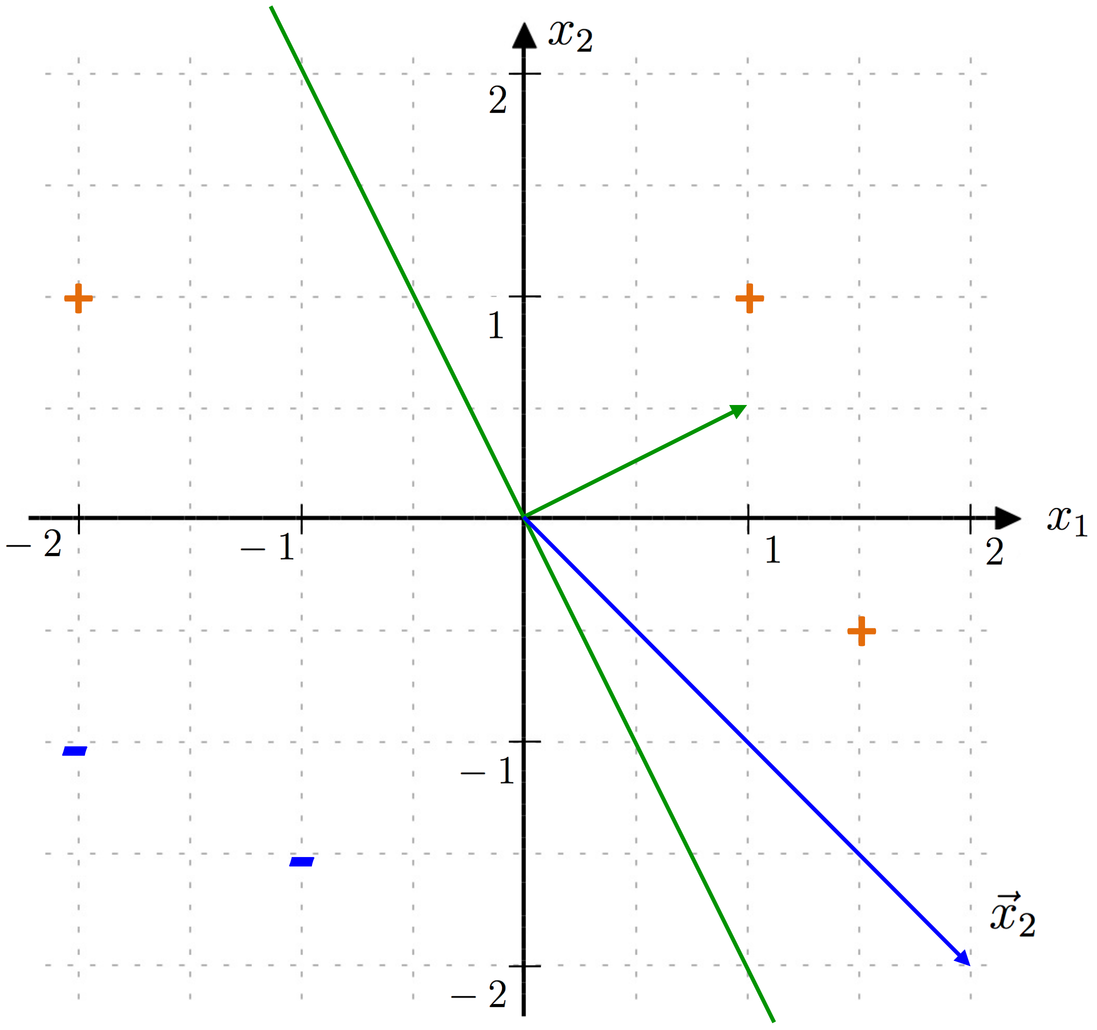
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Handout 13 example

$$\alpha = 0.2$$

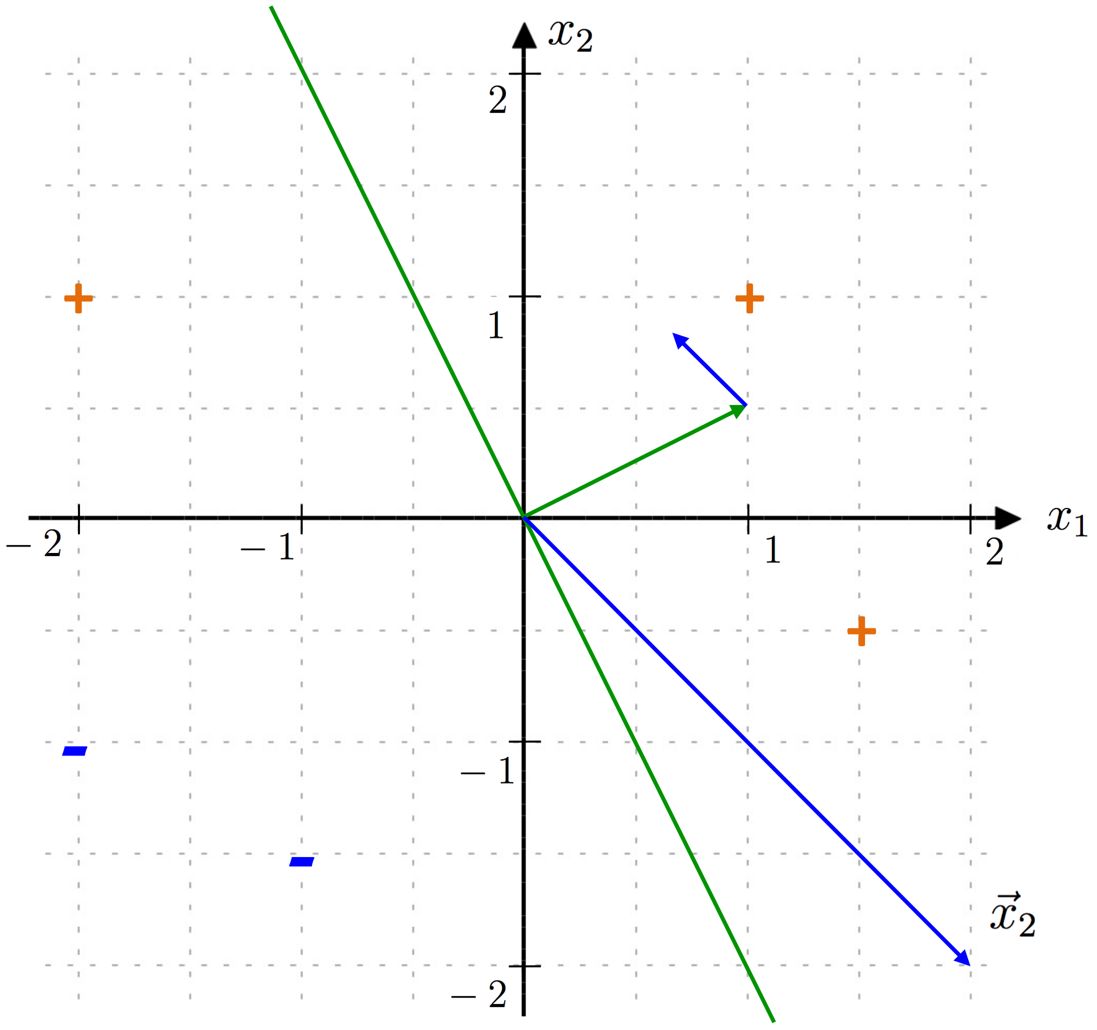
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification
"Push" ***w*** away from negative point



Handout 13 example

$$\alpha = 0.2$$

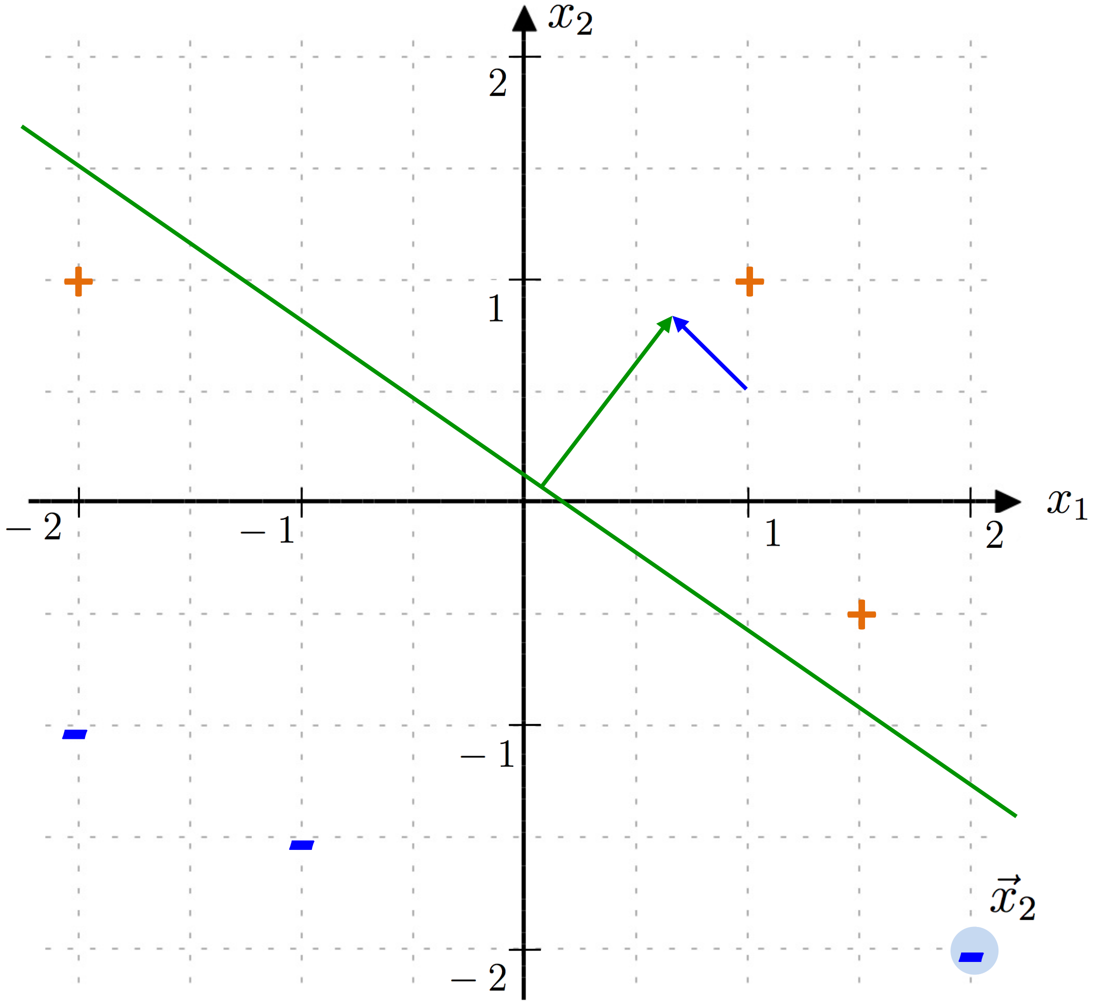
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

Incorrect classification
"Push" ***w*** away from negative point



Handout 13 example

$$\alpha = 0.2$$

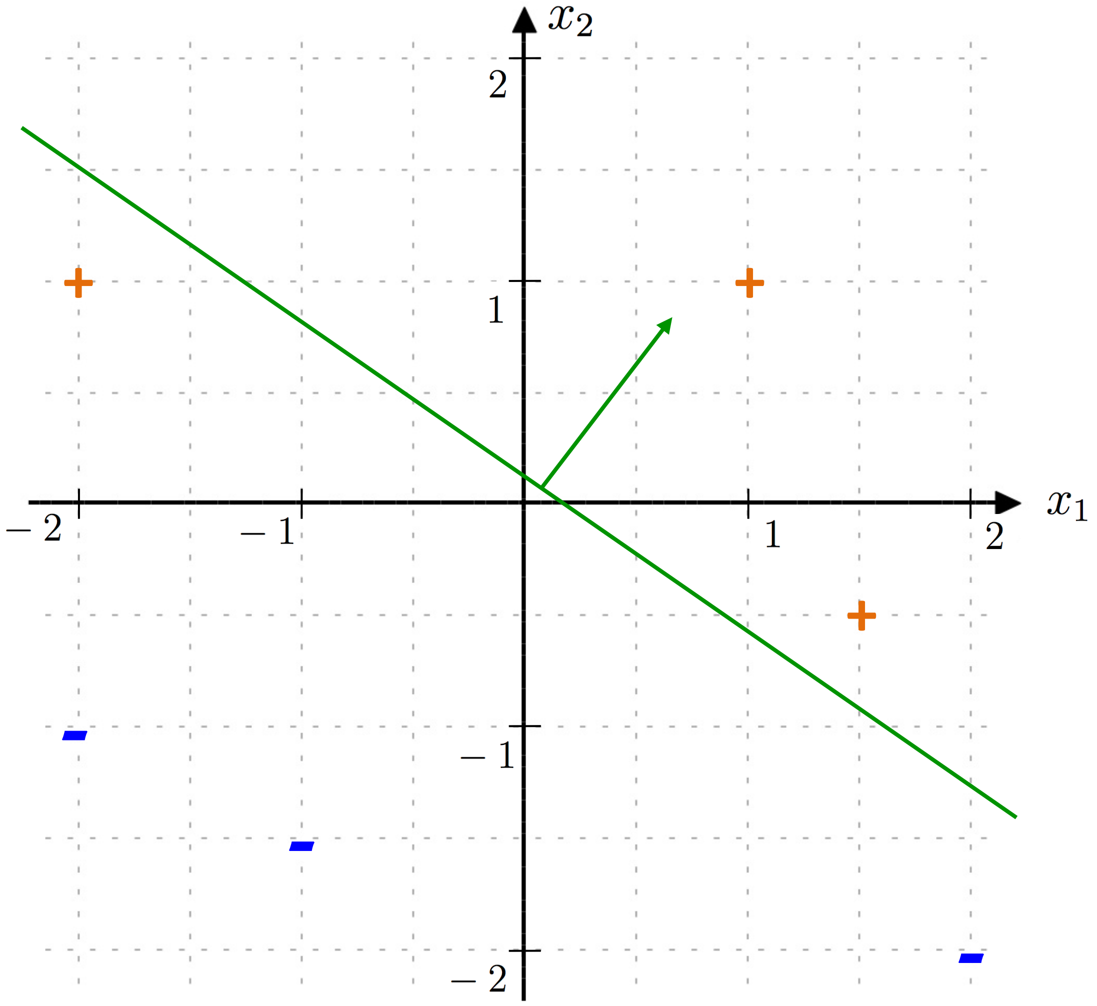
$$\vec{w} = \begin{bmatrix} 0 \\ 1 \\ 0.5 \end{bmatrix}$$

Round 2:

$$\vec{x}_2 = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$$

$$\vec{w} \cdot \vec{x}_2 > 0$$

What is the new weight vector?



Handout 13 example

Final solution (so you can check your work):

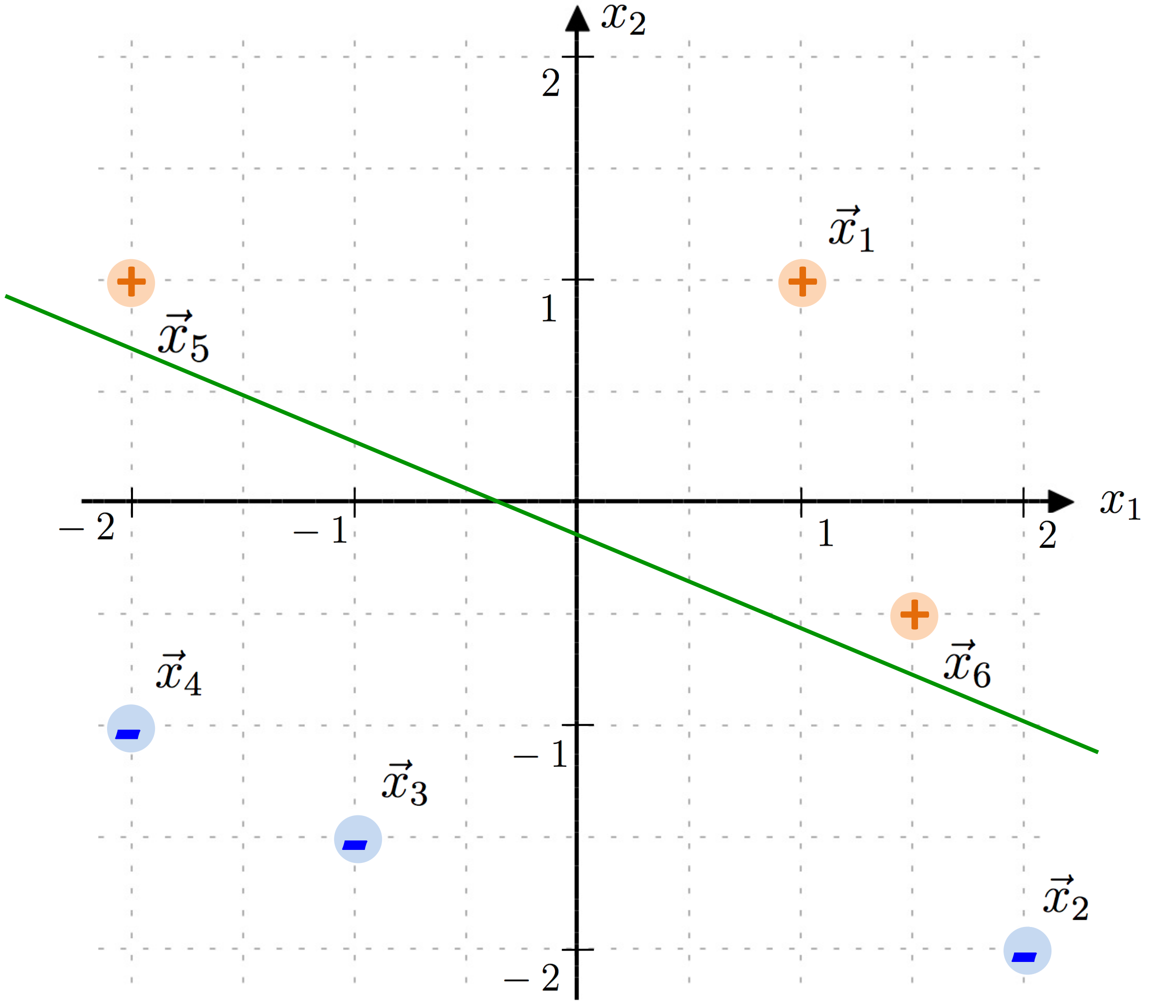
$$\vec{w}^* = \begin{bmatrix} 0.2 \\ 0.5 \\ 1 \end{bmatrix}$$

Final hyperplane:

$$0.2 + 0.5x_1 + x_2 = 0$$

$$\Rightarrow$$

$$x_2 = -0.2 - 0.5x_1$$



Example modified from Achim J. Lilienthal & Thorsteinn Rögnvaldsson

Reading Quiz

1. What is the goal of the perceptron algorithm? Circle all that apply:

   (a) predict a continuous outcome

   (b) quantify how important each feature is for predicting the outcome

   (c) create a linear decision boundary between positives and negatives

   (d) obtain the probability of a positive label for each test example

Reading Quiz

1. What is the goal of the perceptron algorithm? Circle all that apply:

   (a) predict a continuous outcome

   [✓ (b) quantify how important each feature is for predicting the outcome]

   [✓ (c) create a linear decision boundary between positives and negatives]

   (d) obtain the probability of a positive label for each test example

2. *True or False*: The perceptron algorithm was inspired by how neurons are activated in our brains.

Reading Quiz

1. What is the goal of the perceptron algorithm? Circle all that apply:

   (a) predict a continuous outcome

   [✓ (b) quantify how important each feature is for predicting the outcome]

   [✓ (c) create a linear decision boundary between positives and negatives]

   (d) obtain the probability of a positive label for each test example

2. *True or False*: The perceptron algorithm was inspired by how neurons are activated in our brains.

   True

3. Say at some point in the perceptron algorithm I have $\vec{w} = [3, -1, 2]^T$ and $\vec{x} = [1, 2, -2]^T$. What label would we predict for $\vec{x}$?

Reading Quiz

1. What is the goal of the perceptron algorithm? Circle all that apply:

   (a) predict a continuous outcome

   [✓ (b) quantify how important each feature is for predicting the outcome]

   [✓ (c) create a linear decision boundary between positives and negatives]

   (d) obtain the probability of a positive label for each test example

2. *True or False*: The perceptron algorithm was inspired by how neurons are activated in our brains.

   True

3. Say at some point in the perceptron algorithm I have $\vec{w} = [3, -1, 2]^T$ and $\vec{x} = [1, 2, -2]^T$. What label would we predict for $\vec{x}$?

   Dot product = -3 => predict label -1

4. In the example above, say the true label is -1. How would the weights be updated when using this point?

Reading Quiz

1. What is the goal of the perceptron algorithm? Circle all that apply:

   (a) predict a continuous outcome

   [✓ (b) quantify how important each feature is for predicting the outcome]

   [✓ (c) create a linear decision boundary between positives and negatives]

   (d) obtain the probability of a positive label for each test example

2. *True or False*: The perceptron algorithm was inspired by how neurons are activated in our brains.

   True

3. Say at some point in the perceptron algorithm I have $\vec{w} = [3, -1, 2]^T$ and $\vec{x} = [1, 2, -2]^T$. What label would we predict for $\vec{x}$?

   Dot product = -3 => predict label -1

4. In the example above, say the true label is -1. How would the weights be updated when using this point?

   No weight update!

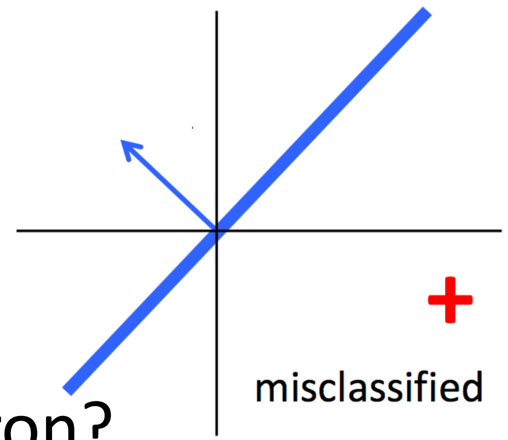
Informal discussion with a partner

1) What is the relationship between the weight vector ***w*** and the hyperplane?

2) Why is the perceptron cost function intuitive?

$$J(\vec{w}) = \sum_{i=1}^{n} \max\left(0, -y_i(\vec{w}^T\vec{x}_i)\right)$$

3) In the example to the right, how will the slope of the hyperplane change?



4) What are the weaknesses of the perceptron? Create a binary classifier "wishlist".

Informal discussion with a partner

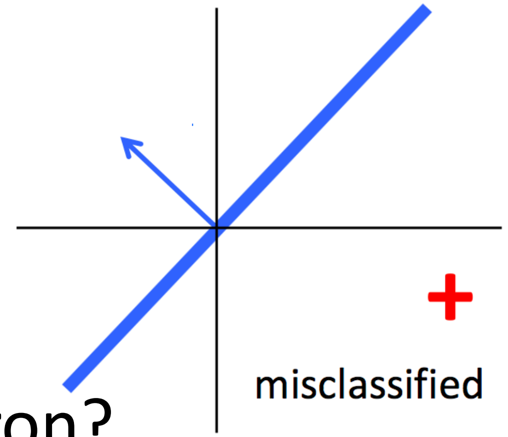
1) What is the relationship between the weight vector ***w*** and the hyperplane?

They are perpendicular

2) Why is the perceptron cost function intuitive?

$$J(\vec{w}) = \sum_{i=1}^{n} \max\left(0, -y_i(\vec{w}^T \vec{x}_i)\right)$$

3) In the example to the right, how will the slope of the hyperplane change?



4) What are the weaknesses of the perceptron? Create a binary classifier "wishlist".

Informal discussion with a partner

1) What is the relationship between the weight vector ***w*** and the hyperplane?

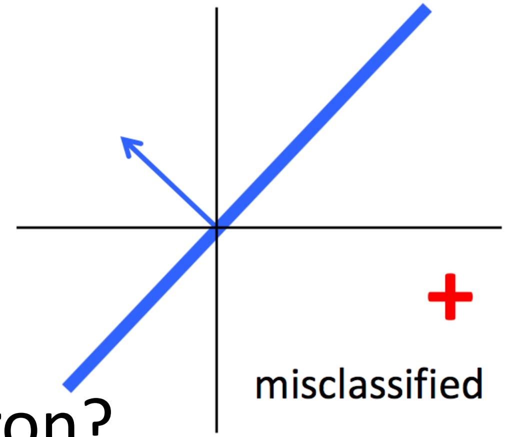
They are perpendicular

2) Why is the perceptron cost function intuitive?

$$J(\vec{w}) = \sum_{i=1}^{n} \max\left(0, -y_i(\vec{w}^T \vec{x}_i)\right)$$

Cost function is 0 when classification is correct, and positive when incorrect

3) In the example to the right, how will the slope of the hyperplane change?



4) What are the weaknesses of the perceptron? Create a binary classifier "wishlist".

Perceptron algorithm and intuition

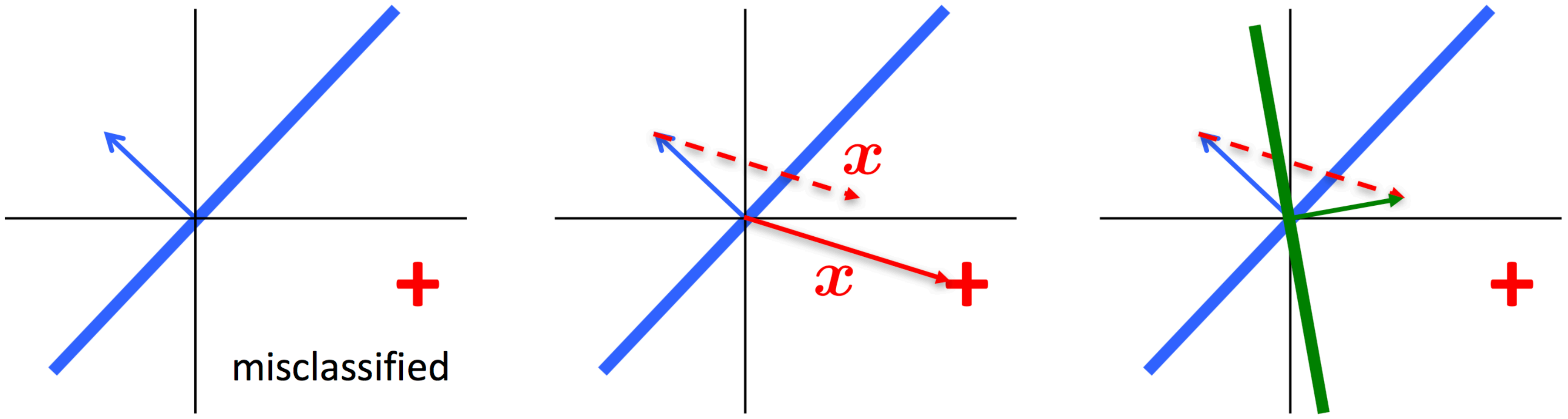
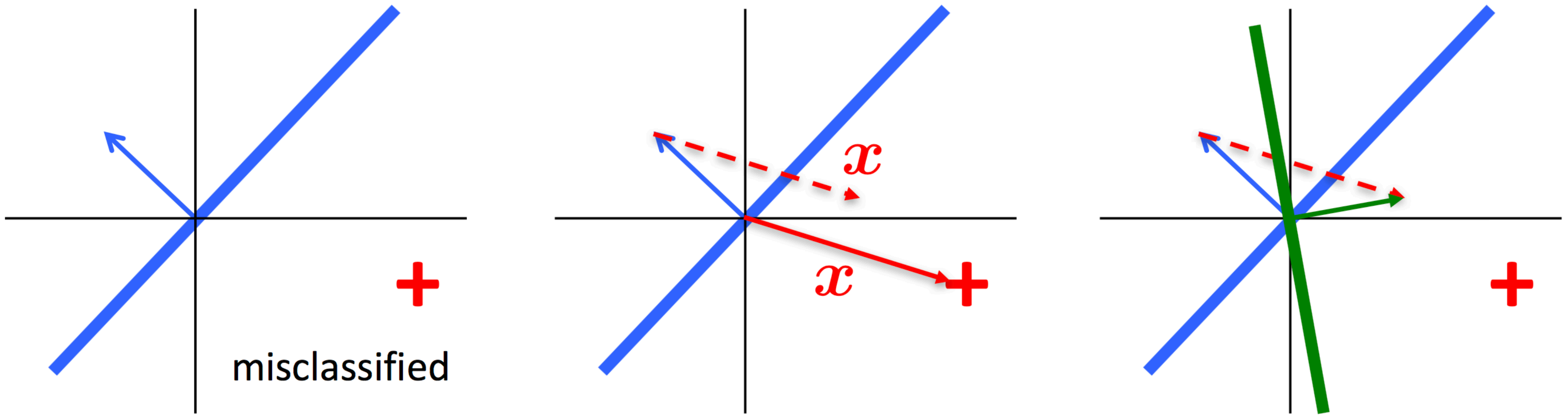


Image and Algorithm: modified from Eric Eaton

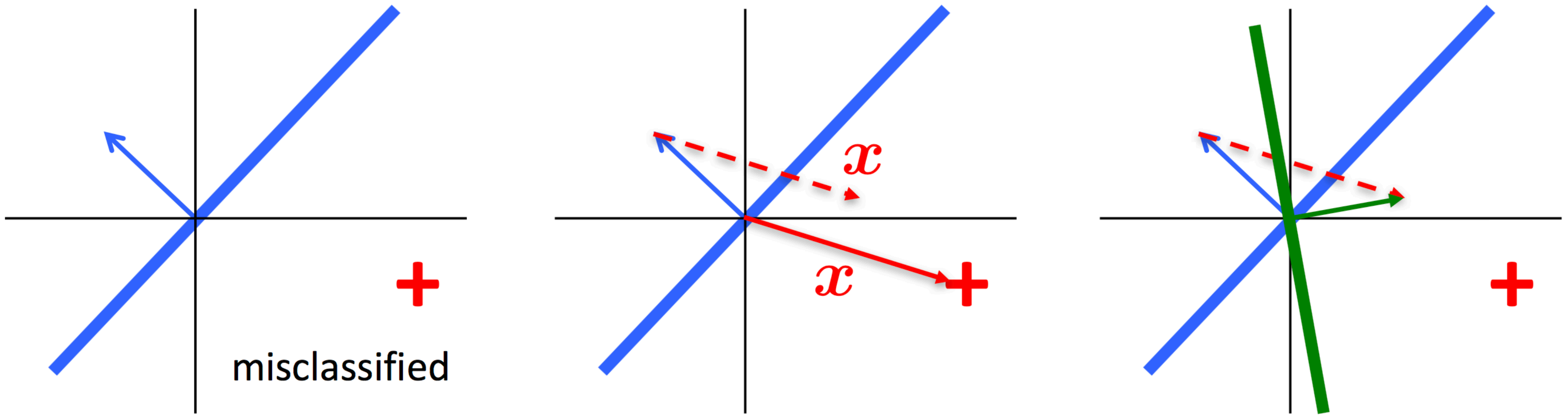
Perceptron algorithm and intuition



```
Let  $\vec{w} = [0, 0, \cdots, 0]^T$
Repeat until convergence:
    Receive training example  $(\vec{x}_i, y_i)$
    If  $y_i(\vec{w}^T \vec{x}_i) \leq 0$    (incorrectly classified)
        $\vec{w} \leftarrow \vec{w} + \alpha y_i \vec{x}_i$
```

Image and Algorithm: modified from Eric Eaton

Perceptron algorithm and intuition



```
Let $\vec{w} = [0, 0, \cdots, 0]^T$
Repeat until convergence:
    Receive training example $(\vec{x}_i, y_i)$
    If $y_i(\vec{w}^T\vec{x}_i) \leq 0$    (incorrectly classified)
        $\vec{w} \leftarrow \vec{w} + \alpha y_i \vec{x}_i$
```

Convergence:

- All data points correctly classified
- Fixed number of iterations passed

Often: alpha = 1 (only changes magnitude of weight vector)

Image and Algorithm: modified from Eric Eaton

Binary classifier wishlist

- If data is linearly separable, want a “good” hyperplane (idea: far from points close to the boundary)

- If data is not linearly separable, want something reasonable (not just give up or fail to converge)

- Might not want to constrain ourselves to linear separators

Outline: optional material on SVMs

- Recap Perceptron (+ Handout 13)
- Support Vector Machines (SVMs) overview
- Extensions of SVMs
- SVMs meta-optimization process (+ Handout 14)

Support Vector Machines (SVMs)

- Will give us everything on our wishlist!
- Often considered the best “off the shelf” binary classifier
- Widely used in many fields

Brief history

- 1963: Initial idea by Vladimir Vapnik and Alexey Chervonenkis
- 1992: nonlinear SVMs by Bernhard Boser, Isabelle Guyon and Vladimir Vapnik
- 1993: “soft-margin” by Corinna Cortes and Vladimir Vapnik

Idea: “best” hyperplane has a large margin

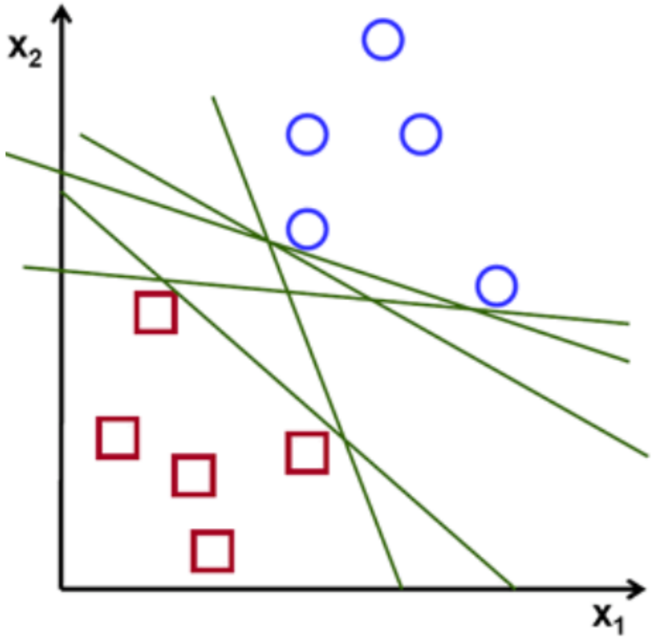


Image: “Towards Data Science”

Idea: "best" hyperplane has a large margin

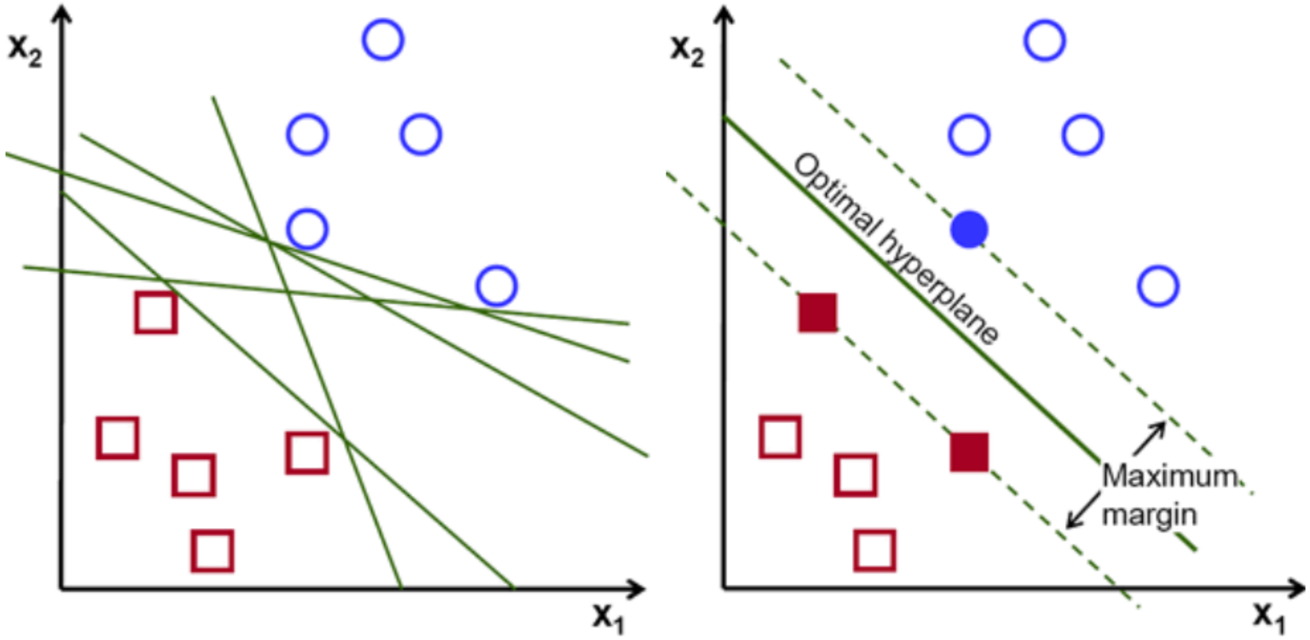
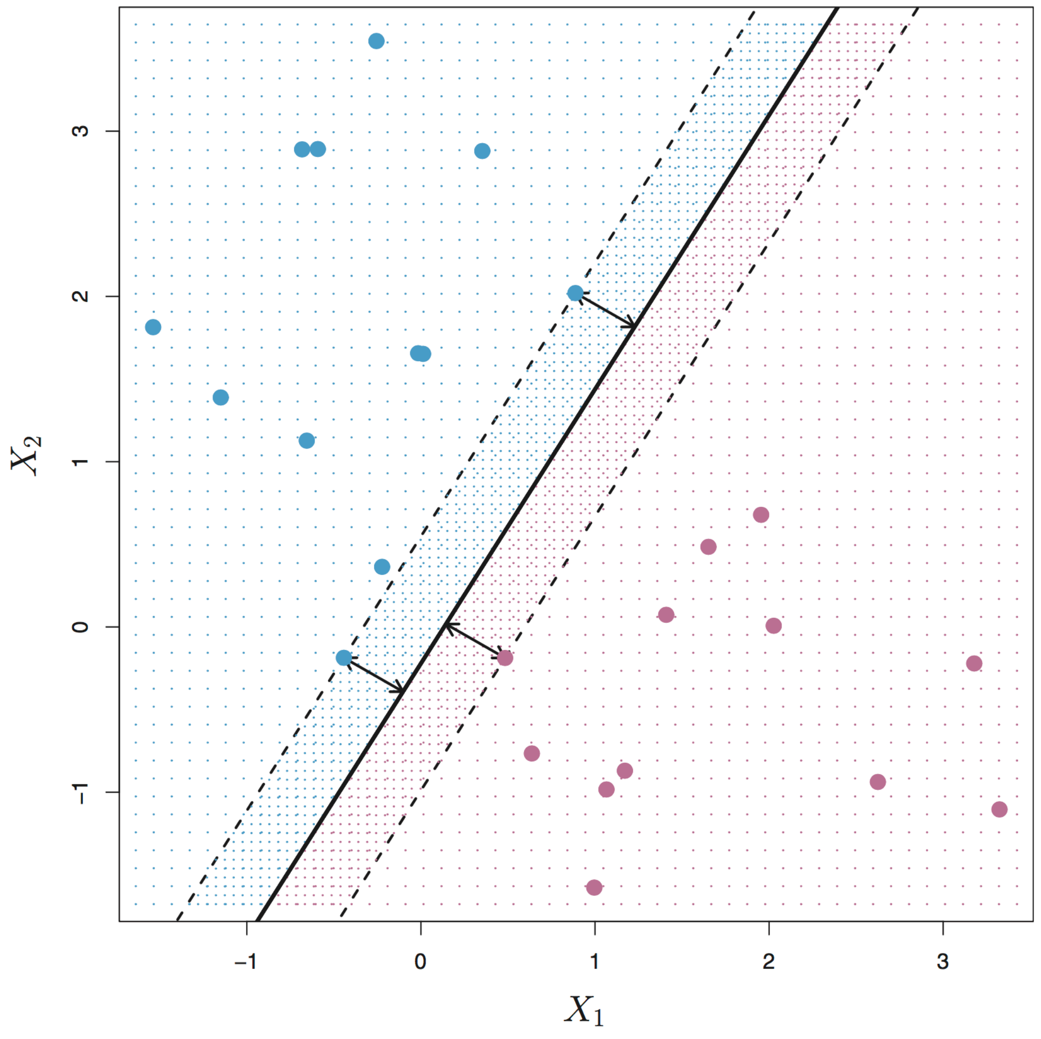


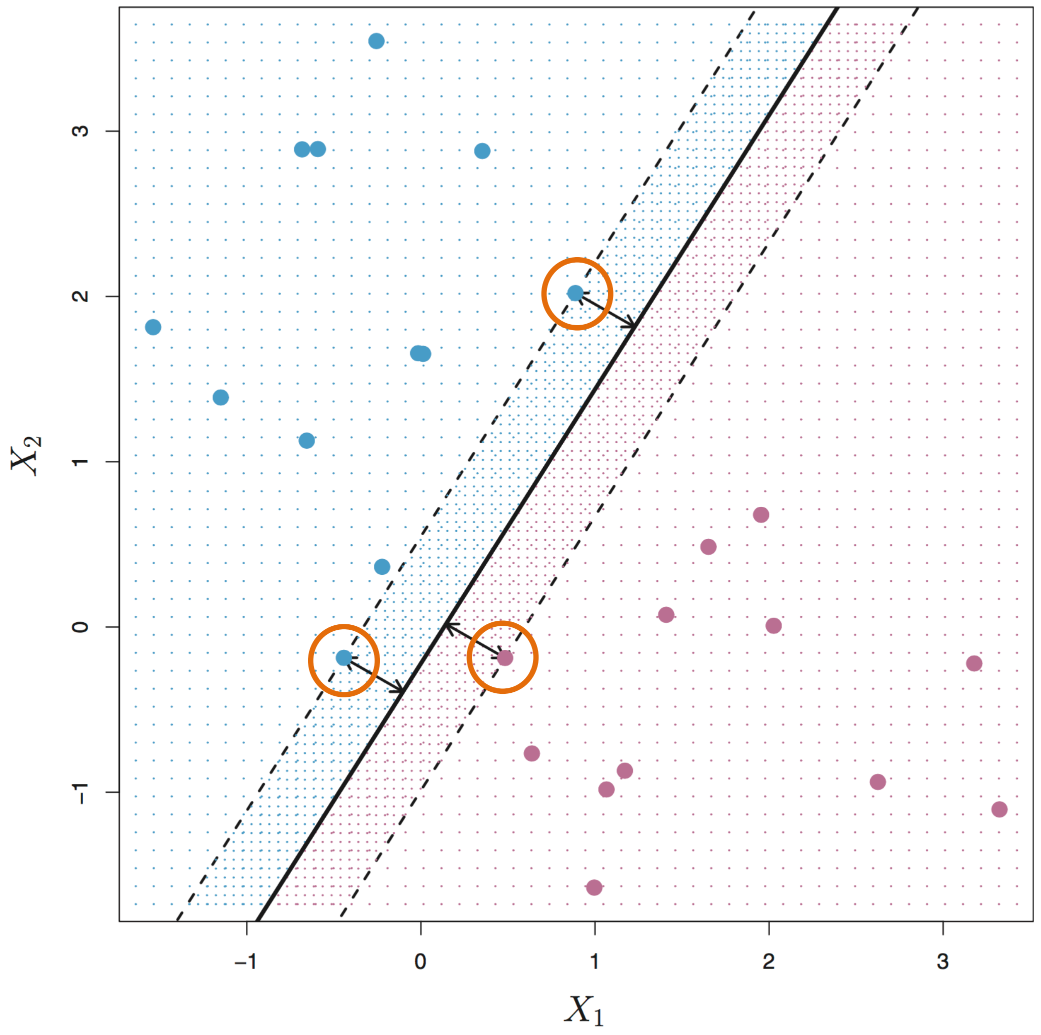
Image: "Towards Data Science"

Datapoints that lie on the margin are called "support vectors"



Modified from ISL: Figure 9.3

Datapoints that lie on the margin are called "support vectors"



Support vectors

Modified from ISL: Figure 9.3

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\left(\vec{w} \cdot \vec{x} + b\right)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \operatorname{sign}\big(\vec{w} \cdot \vec{x} + b\big)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\left(\vec{w} \cdot \vec{x} + b\right)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Geometric Margin:
(distance between
example and hyperplane)

$$\gamma_i = y_i \left(\frac{\vec{w}}{||\vec{w}||} \cdot \vec{x}_i + \frac{b}{||\vec{w}||} \right)$$

Functional and Geometric Margins

SVM classifier:
(same as perceptron)

$$h(\vec{x}) = \text{sign}\left(\vec{w} \cdot \vec{x} + b\right)$$

Functional Margin:

$$\hat{\gamma}_i = y_i(\vec{w} \cdot \vec{x}_i + b)$$

Geometric Margin:
(distance between example and hyperplane)

$$\gamma_i = y_i\left(\frac{\vec{w}}{||\vec{w}||} \cdot \vec{x}_i + \frac{b}{||\vec{w}||}\right)$$

Note:

$$\boxed{\gamma_i = \frac{\hat{\gamma}_i}{||\vec{w}||}}$$

Optimization Problem: try 1

Goal: maximize the minimum distance between example and hyperplane

$$\gamma = \min_{i=1,\cdots,n} \gamma_i$$

Optimization Problem: try 1

Goal: maximize the minimum distance between example and hyperplane

$$\gamma = \min_{i=1,\cdots,n} \gamma_i$$

Formulation: optimize a function with respect to a constraint

$$\begin{aligned} \max_{\gamma,\vec{w},b} \quad & \gamma \\ \text{s.t.} \quad & y_i(\vec{w}\cdot\vec{x}_i + b) \geq \gamma, \quad i = 1,\cdots,n \\ \text{and} \quad & \|\vec{w}\| = 1 \end{aligned}$$

(force functional and geometric margin to be equal)

Optimization Problem: try 2

Idea: substitute functional margin divided by magnitude of weight vector

$$\max_{\hat{\gamma},\vec{w},b} \quad \frac{\hat{\gamma}}{\|\vec{w}\|}$$

$$\text{s.t.} \quad y_i(\vec{w}\cdot\vec{x}_i + b) \geq \hat{\gamma}, \quad i = 1,\cdots,n$$

(gets rid of non-convex constraint)

Optimization Problem: try 3

Idea: put arbitrary constraint on functional margin

$$\boxed{\hat{\gamma} = 1}$$

$$\min_{\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2$$

$$\text{s.t.} \quad y_i(\vec{w}\cdot\vec{x}_i + b) \geq 1, \quad i = 1, \cdots, n$$

Optimization Problem: try 3

Idea: put arbitrary constraint on functional margin

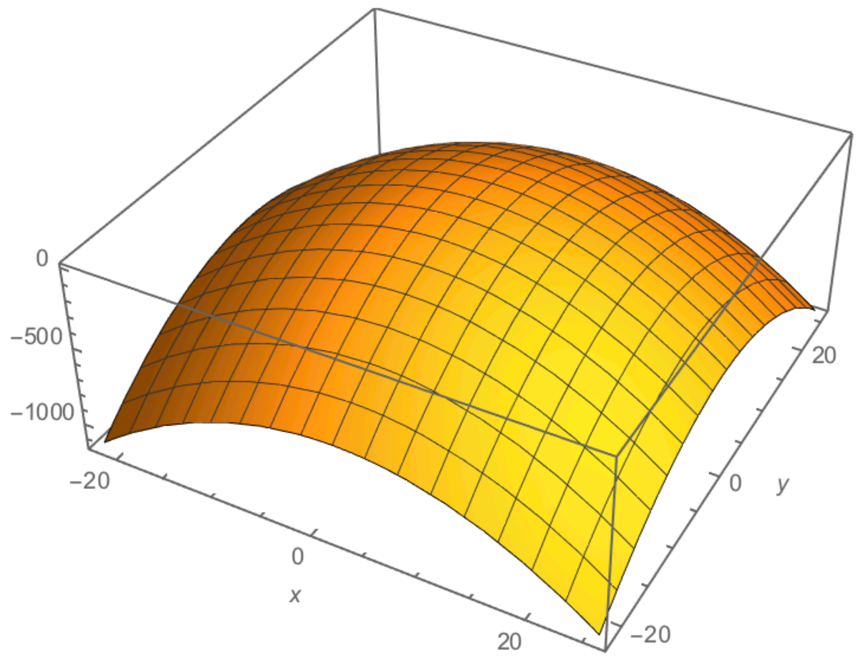
$$\boxed{\hat{\gamma} = 1}$$

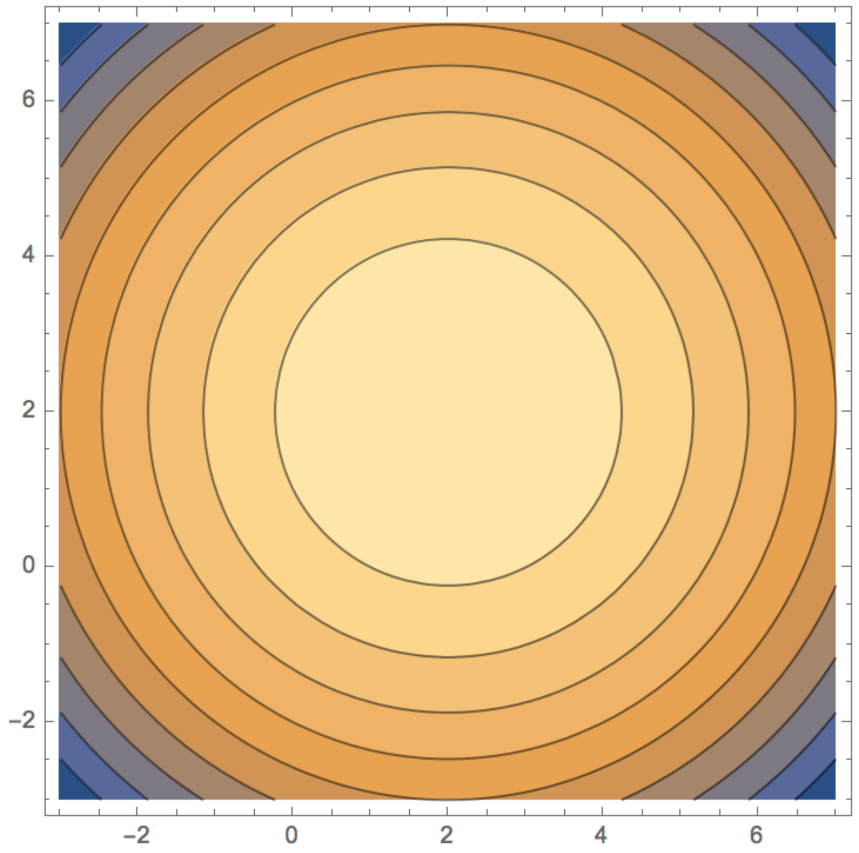
$$\begin{aligned} \min_{\vec{w},b} \quad & \frac{1}{2}\|\vec{w}\|^2 \\ \text{s.t.} \quad & y_i(\vec{w}\cdot\vec{x}_i + b) \geq 1, \quad i = 1, \cdots, n \end{aligned}$$

$$\boxed{\begin{aligned} \min_{\vec{w},b} \quad & \frac{1}{2}\|\vec{w}\|^2 \\ \text{s.t.} \quad & y_i(\vec{w}\cdot\vec{x}_i + b) - 1 \geq 0, \quad i = 1, \cdots, n \end{aligned}}$$

Lagrange multipliers example

$$f(x,y) = 5 - (x-2)^2 - (y-2)^2$$



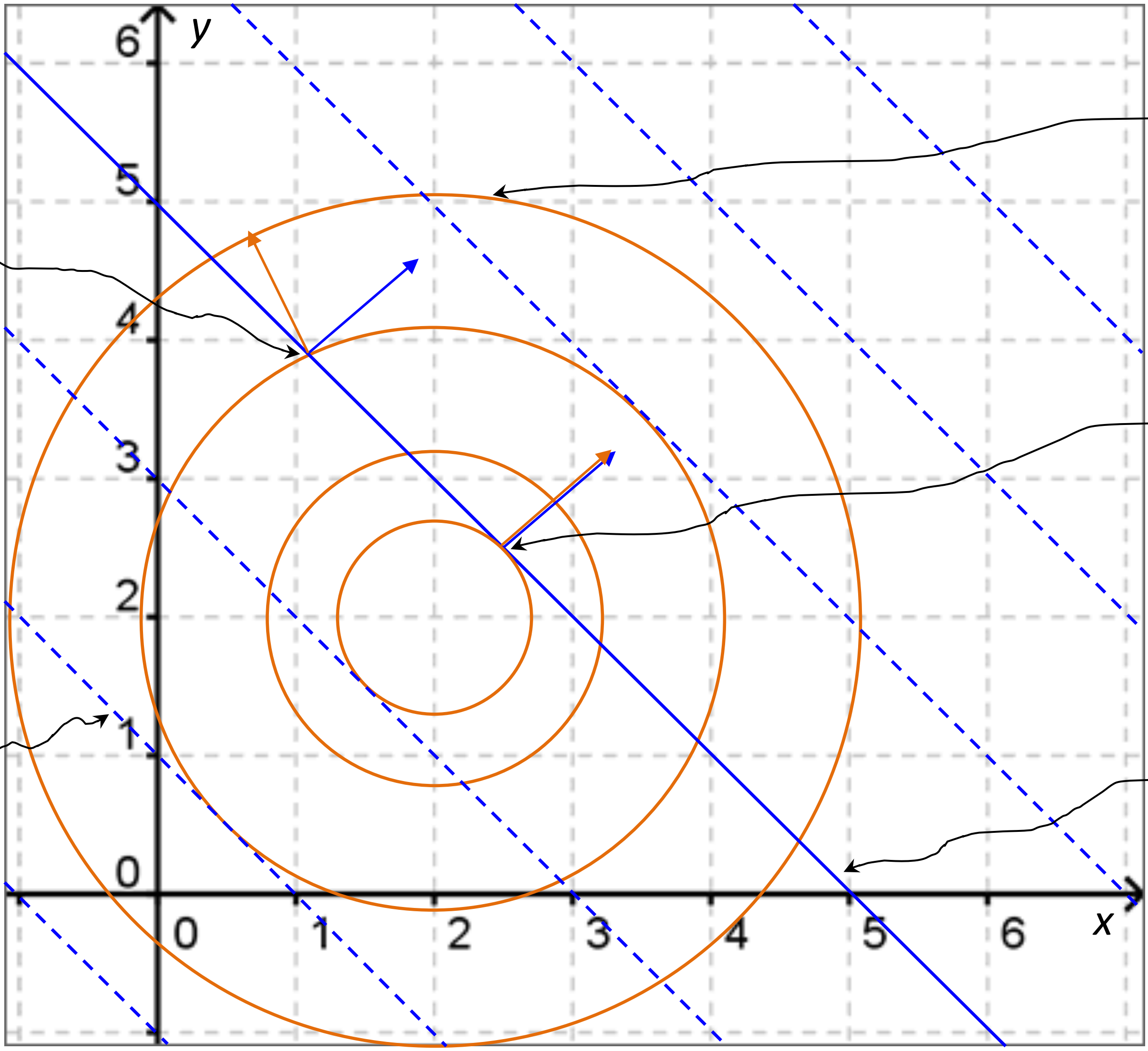


Contour plot of *f*(*x*,*y*)

$$\begin{aligned} &\text{maximize}_{x,y} \quad f(x,y) \\ &s.t. \quad g(x,y) = 0 \end{aligned}$$

$$g(x,y) = -5 + x + y$$

Lagrange multipliers example



Normals (and derivatives) **not** parallel

level curves of $f(x,y)$

<u>SOLUTION</u>: Normals (and derivatives) parallel

x = 2.5

y = 2.5

level curves of $g(x,y)$

$g(x,y) = 0$

Outline: optional material on SVMs

- Recap Perceptron (+ Handout 13)
- Support Vector Machines (SVMs) overview
- Extensions of SVMs
- SVMs meta-optimization process (+ Handout 14)

Kernel Idea

- By solving the dual form of the problem, we have seen how all computations can be done in terms of inner products between examples

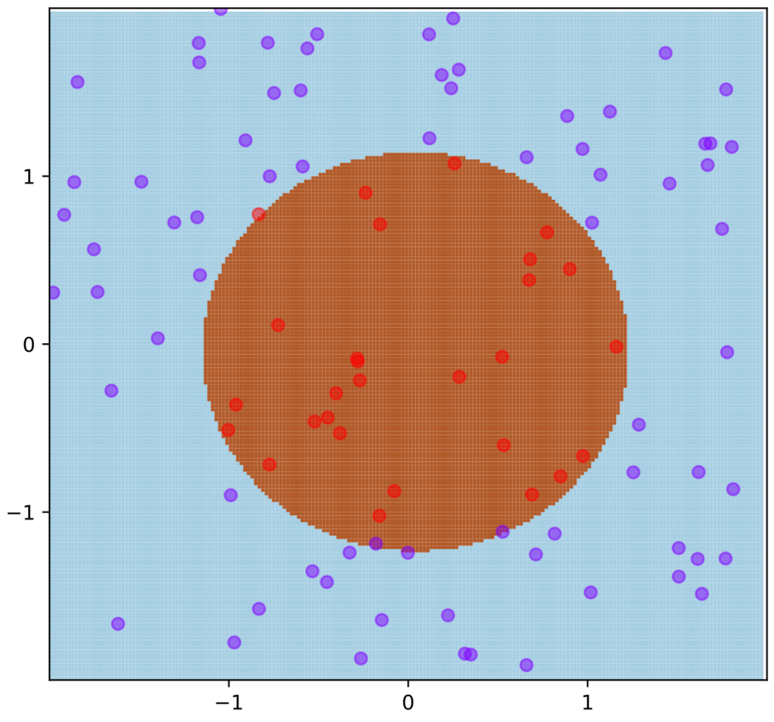
- One example of an inner product is the dot product, which is the linear version of SVMs

- But there are many others!

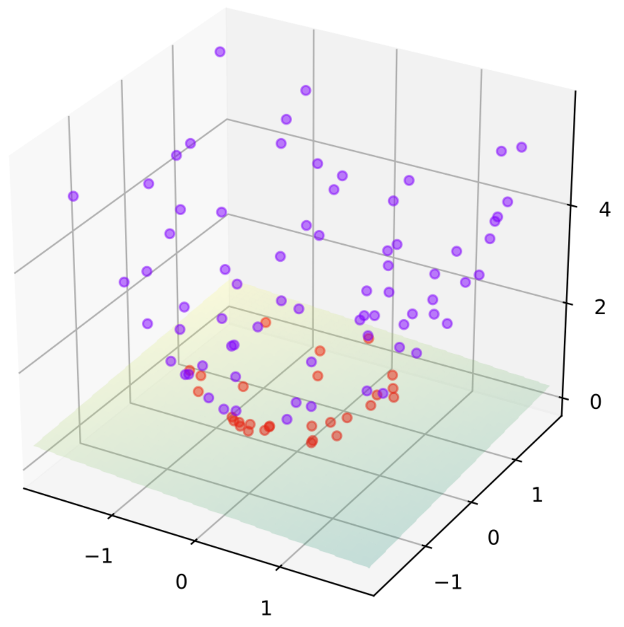
- Intuition: if points are close together, their kernel function will have a large value (measure of similarity)

Kernel Trick example

Feature mapping: $\varphi(\boldsymbol{x}) = (x_1,\ x_2,\ x_1^2 + x_2^2)$



Original feature space



Mapping after applying kernel
(can now find a hyperplane)

Kernel function: $K(\boldsymbol{x},\ \boldsymbol{z}) = \boldsymbol{x}\bullet\boldsymbol{z} + \|\boldsymbol{x}\|^2\,\|\boldsymbol{z}\|^2$

Image: Shiyu Ji (wikipedia)

Gaussian Kernel

- Gaussian kernel is near 0 when points are far apart and near 1 when they are similar
- Also called Radial Basis Function (RBF) kernel

$$K(\vec{x}, \vec{z}) = \exp\left(-\frac{\|\vec{x} - \vec{z}\|^2}{2\sigma^2}\right)$$

Gaussian Kernel

- Gaussian kernel is near 0 when points are far apart and near 1 when they are similar
- Also called Radial Basis Function (RBF) kernel

$$K(\vec{x}, \vec{z}) = \exp\left(-\frac{\|\vec{x} - \vec{z}\|^2}{2\sigma^2}\right)$$

Often re-parametrized by gamma (different gamma!)

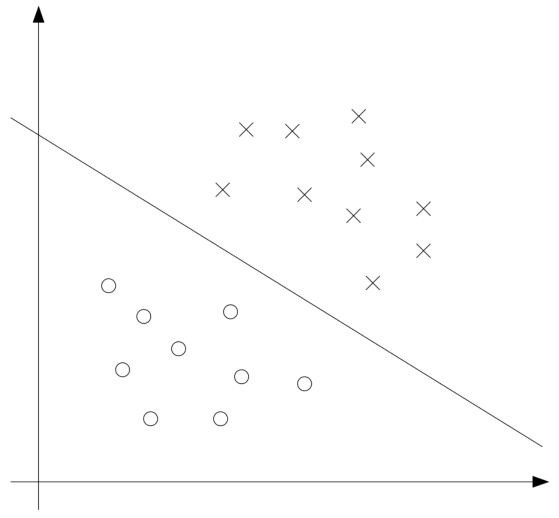
$$\boxed{\gamma = \frac{1}{2\sigma^2}}$$

$$K(\vec{x}, \vec{z}) = \exp\left(-\gamma\|\vec{x} - \vec{z}\|^2\right)$$

Soft-margin SVMs (non-separable case)

- Idea: we will use regularization to add a cost for each point being incorrectly classified by the hyperplane

- Hopefully many costs will be 0, but we can accommodate a few outliers



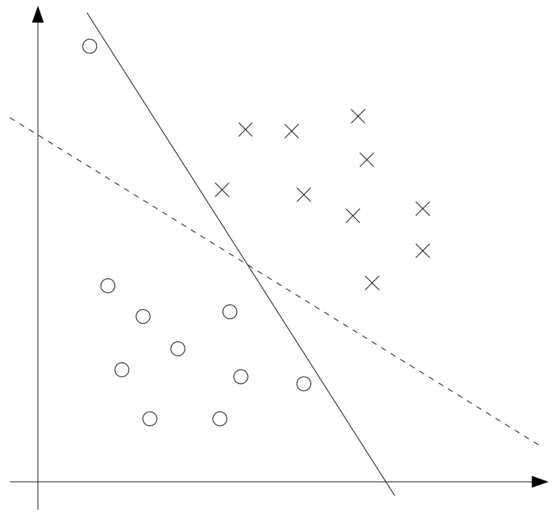


Figure: Andrew Ng

Soft-margin SVMs (non-separable case)

- New optimization problem with regularization

$$\min_{\xi,\vec{w},b} \quad \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

"flexible margin"

$$\text{s.t.} \quad y_i(\vec{w}\cdot\vec{x}_i + b) \geq 1 - \xi_i, \quad i = 1,\cdots,n$$

$$\text{and} \quad \xi_i \geq 0, \quad i = 1,\cdots,n$$

Outline: optional material on SVMs

- Recap Perceptron (+ Handout 13)
- Support Vector Machines (SVMs) overview
- Extensions of SVMs
- SVMs meta-optimization process (+ Handout 14)

Meta-optimization process

- Incremental SVM optimization algorithm

Meta-optimization process

- Incremental SVM optimization algorithm
- Choose a subset S of examples and run optimization to get alpha values

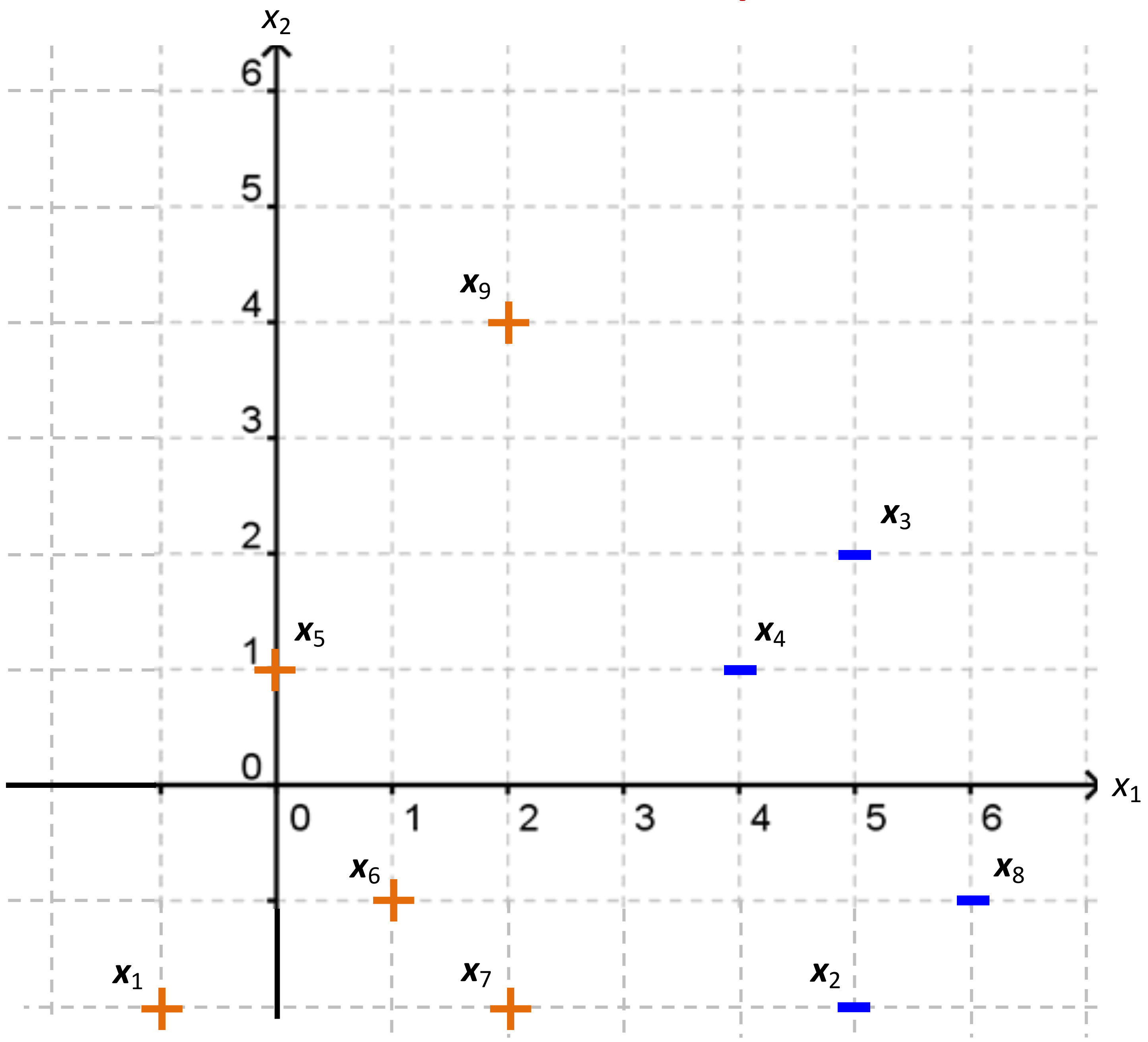
Meta-optimization process

- Incremental SVM optimization algorithm
- Choose a subset S of examples and run optimization to get alpha values
- Identify which alpha values are 0 => these cannot be support vectors in final solution!

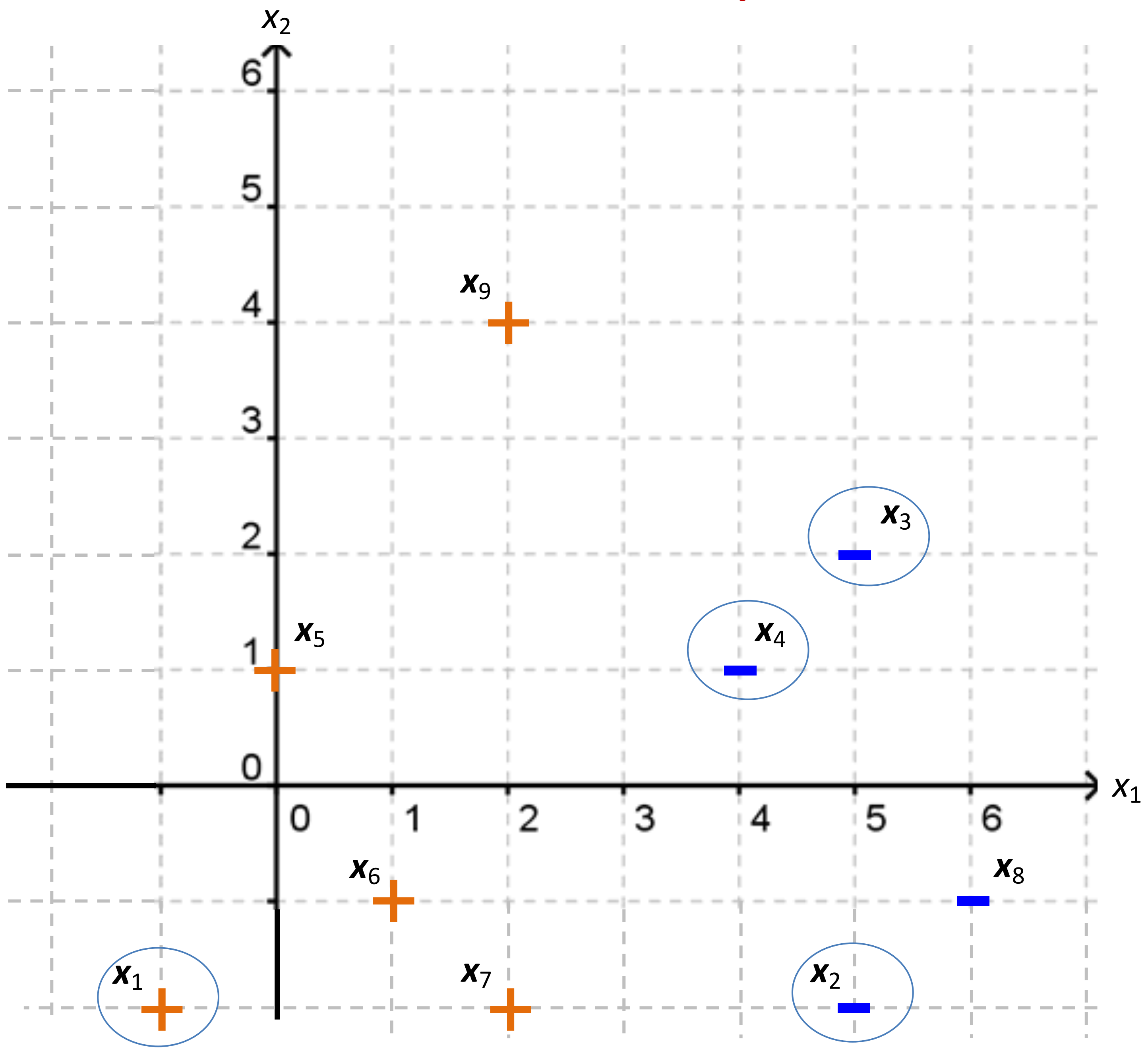
Meta-optimization process

- Incremental SVM optimization algorithm
- Choose a subset S of examples and run optimization to get alpha values
- Identify which alpha values are 0 => these cannot be support vectors in final solution!
- Discard these points and add new ones; repeat

Meta-optimization: example



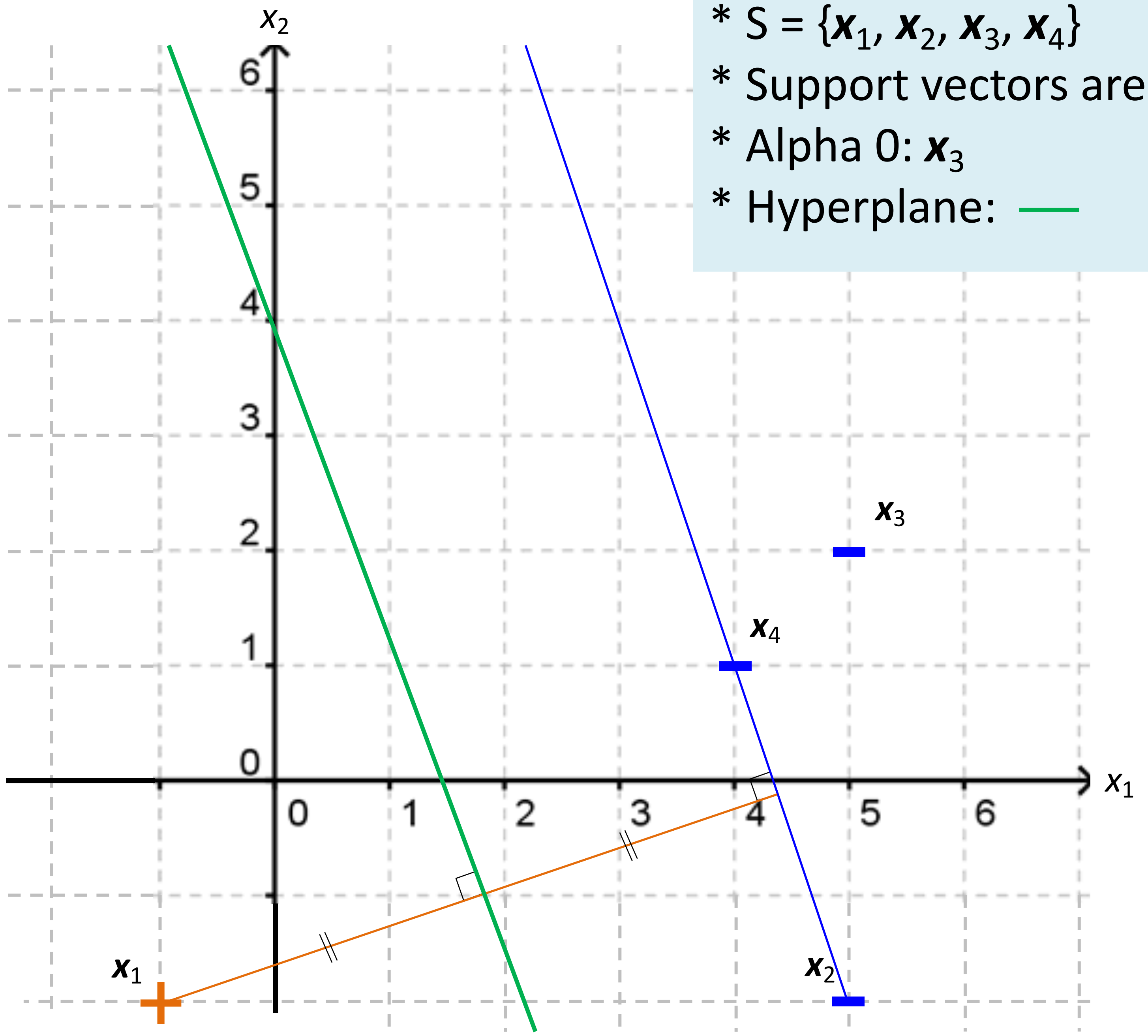
Meta-optimization: example

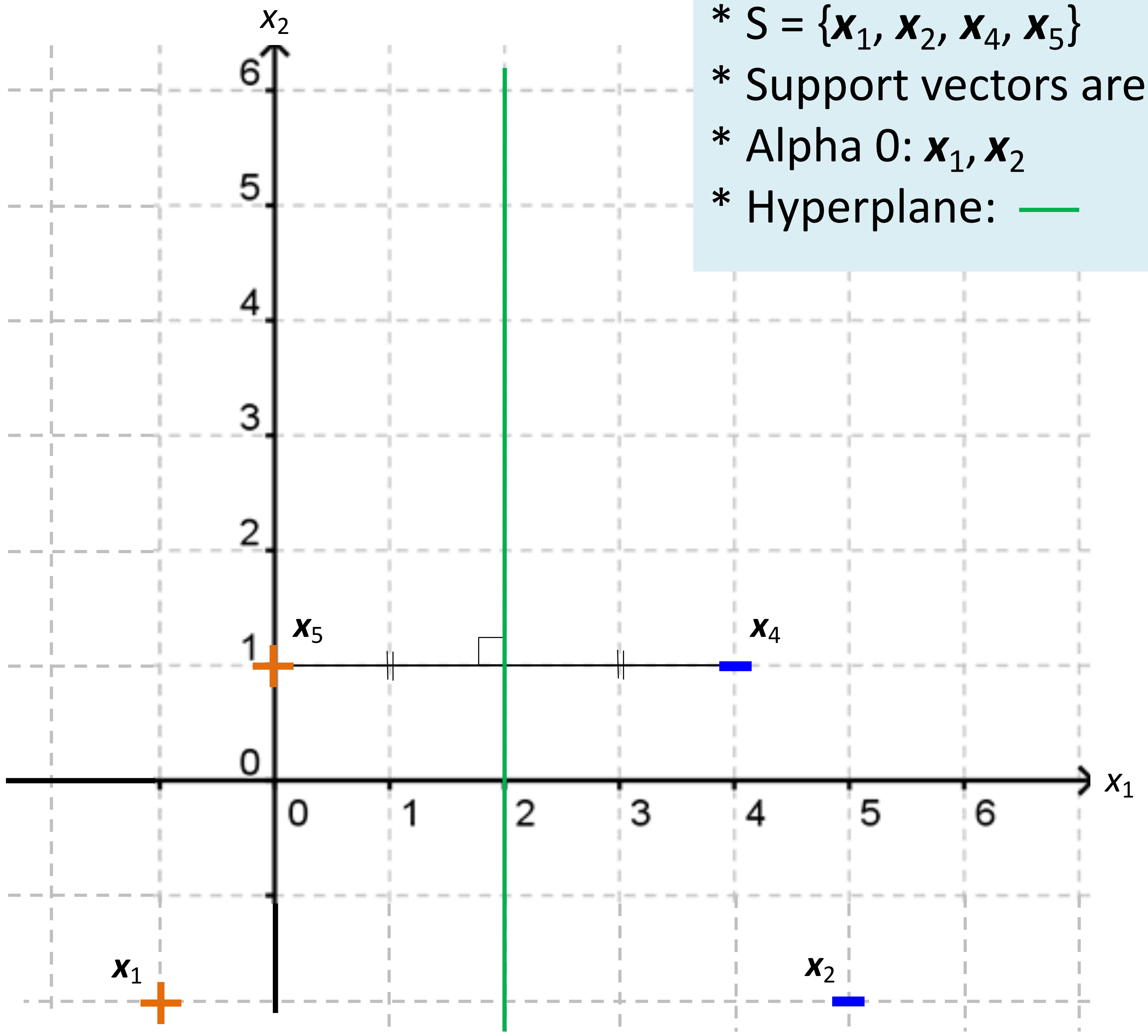


$K = 4$

Round 1:

* S = {$\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_3$, $\boldsymbol{x}_4$}
* Support vectors are: $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_4$
* Alpha 0: $\boldsymbol{x}_3$
* Hyperplane: 🟩



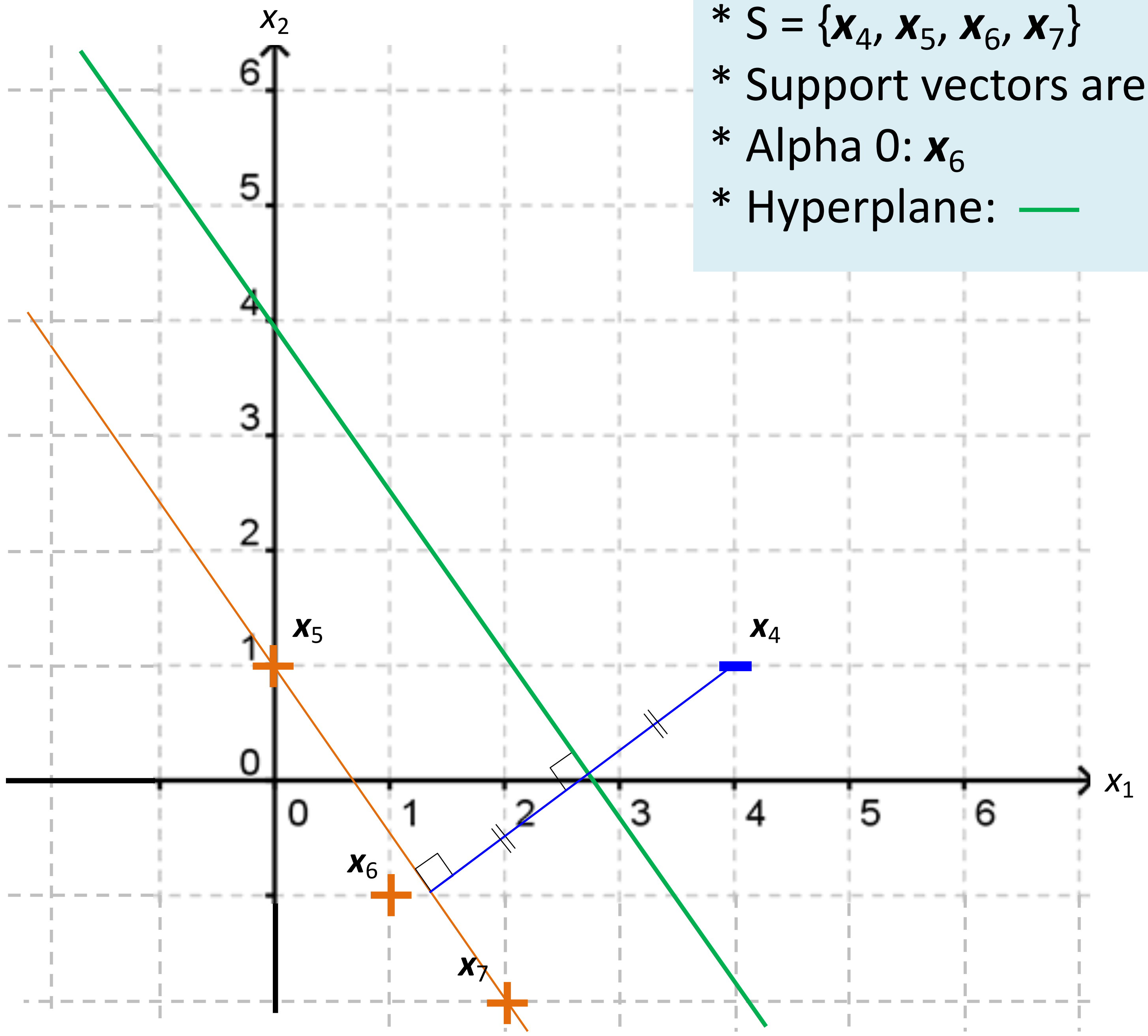


Round 1:

* S = $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_4, \boldsymbol{x}_5\}$
* Support vectors are: $\boldsymbol{x}_4$, $\boldsymbol{x}_5$
* Alpha 0: $\boldsymbol{x}_1$, $\boldsymbol{x}_2$
* Hyperplane: 🟩

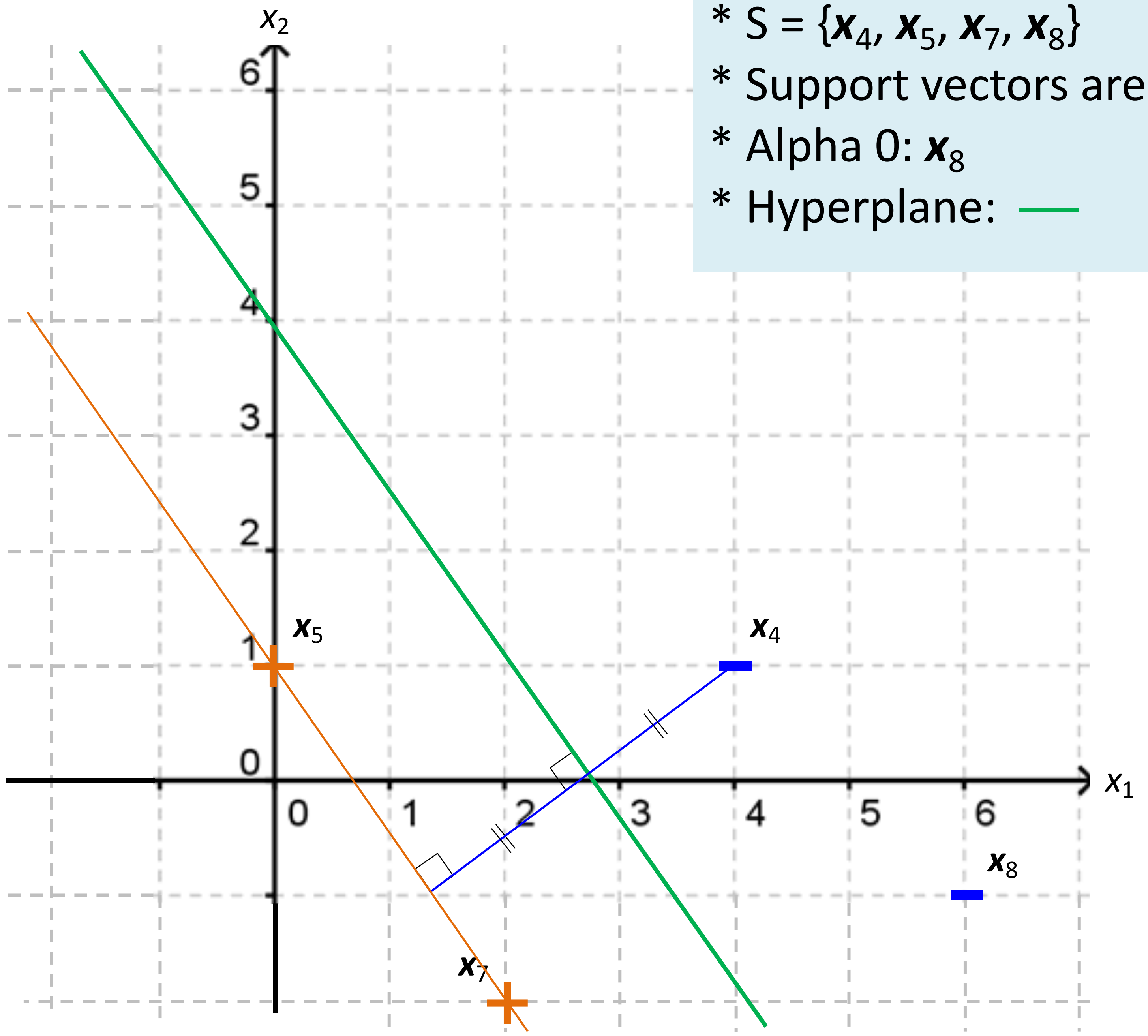
Round 3:

* S = {$\boldsymbol{x}_4$, $\boldsymbol{x}_5$, $\boldsymbol{x}_6$, $\boldsymbol{x}_7$}
* Support vectors are: $\boldsymbol{x}_4$, $\boldsymbol{x}_5$, $\boldsymbol{x}_7$
* Alpha 0: $\boldsymbol{x}_6$
* Hyperplane: —



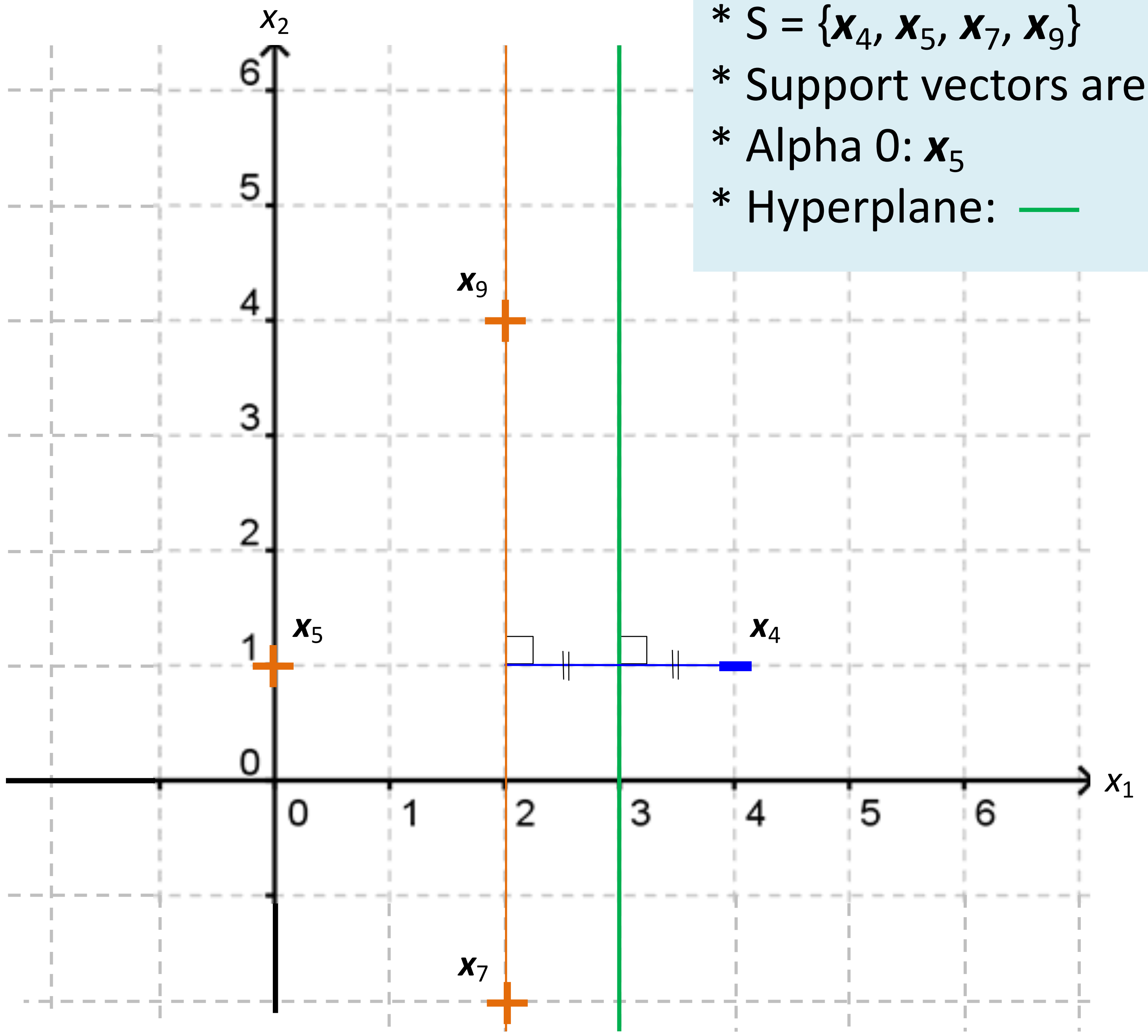
Round 4:

* S = {$\boldsymbol{x}_4$, $\boldsymbol{x}_5$, $\boldsymbol{x}_7$, $\boldsymbol{x}_8$}
* Support vectors are: $\boldsymbol{x}_4$, $\boldsymbol{x}_5$, $\boldsymbol{x}_7$
* Alpha 0: $\boldsymbol{x}_8$
* Hyperplane: 🟩

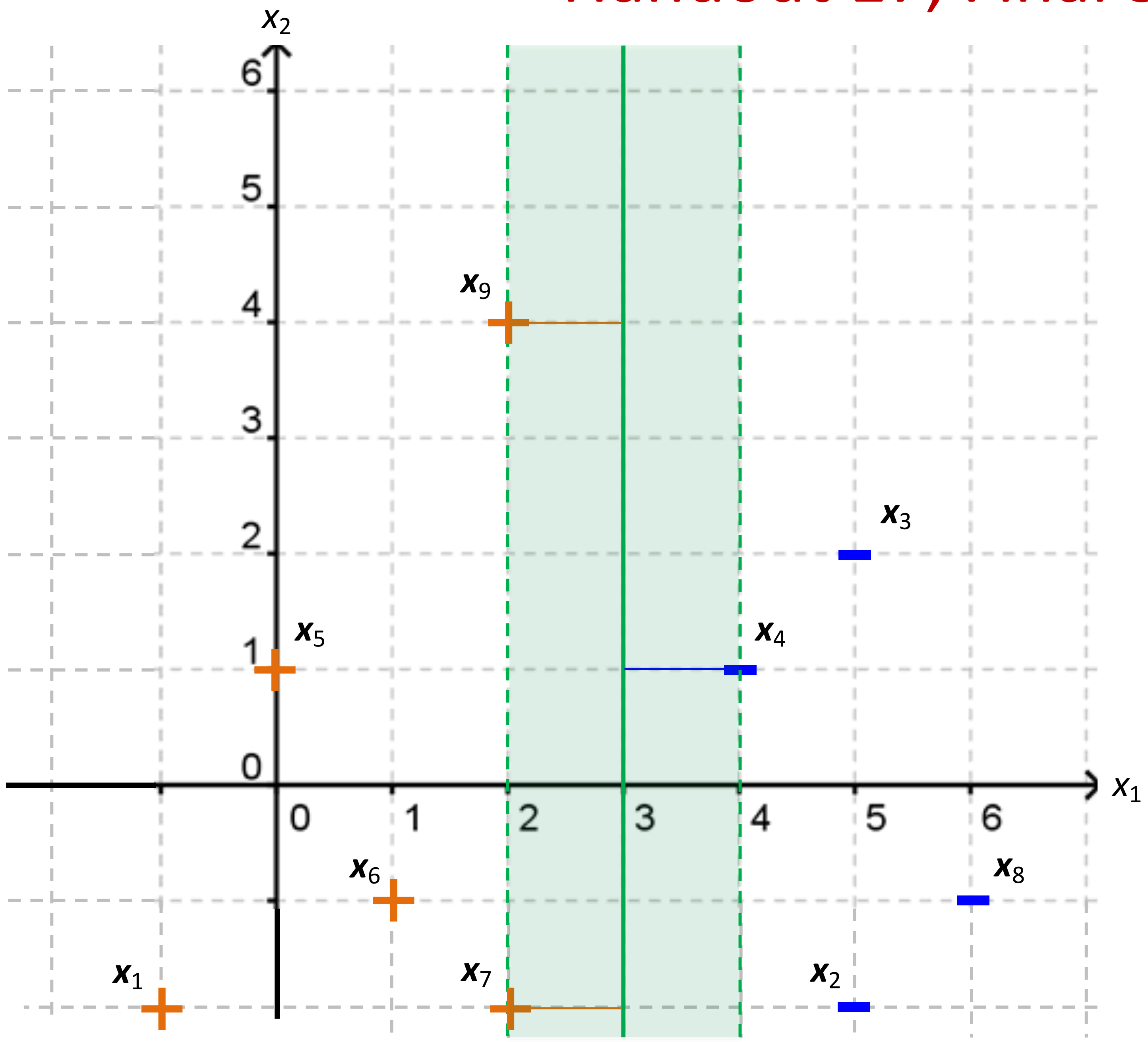


Round 5:

* S = {$\boldsymbol{x}_4$, $\boldsymbol{x}_5$, $\boldsymbol{x}_7$, $\boldsymbol{x}_9$}
* Support vectors are: $\boldsymbol{x}_4$, $\boldsymbol{x}_7$, $\boldsymbol{x}_9$
* Alpha 0: $\boldsymbol{x}_5$
* Hyperplane: 🟩



Handout 17, Final Solution



Reading Quiz #8

1. If $\vec{x}_i$ is a support vector, what can we say about it? Circle all that apply:

   (a) its Lagrange multiplier $\alpha_i > 0$

   (b) its Lagrange multiplier $\alpha_i = 0$

   (c) $y_i(\vec{w} \cdot \vec{x}_i + b) = 0$

   (d) $y_i(\vec{w} \cdot \vec{x}_i + b) = 1$

   (e) $\vec{x}_i$ lies on the margin

Reading Quiz #8

1. If $\vec{x}_i$ is a support vector, what can we say about it? Circle all that apply:

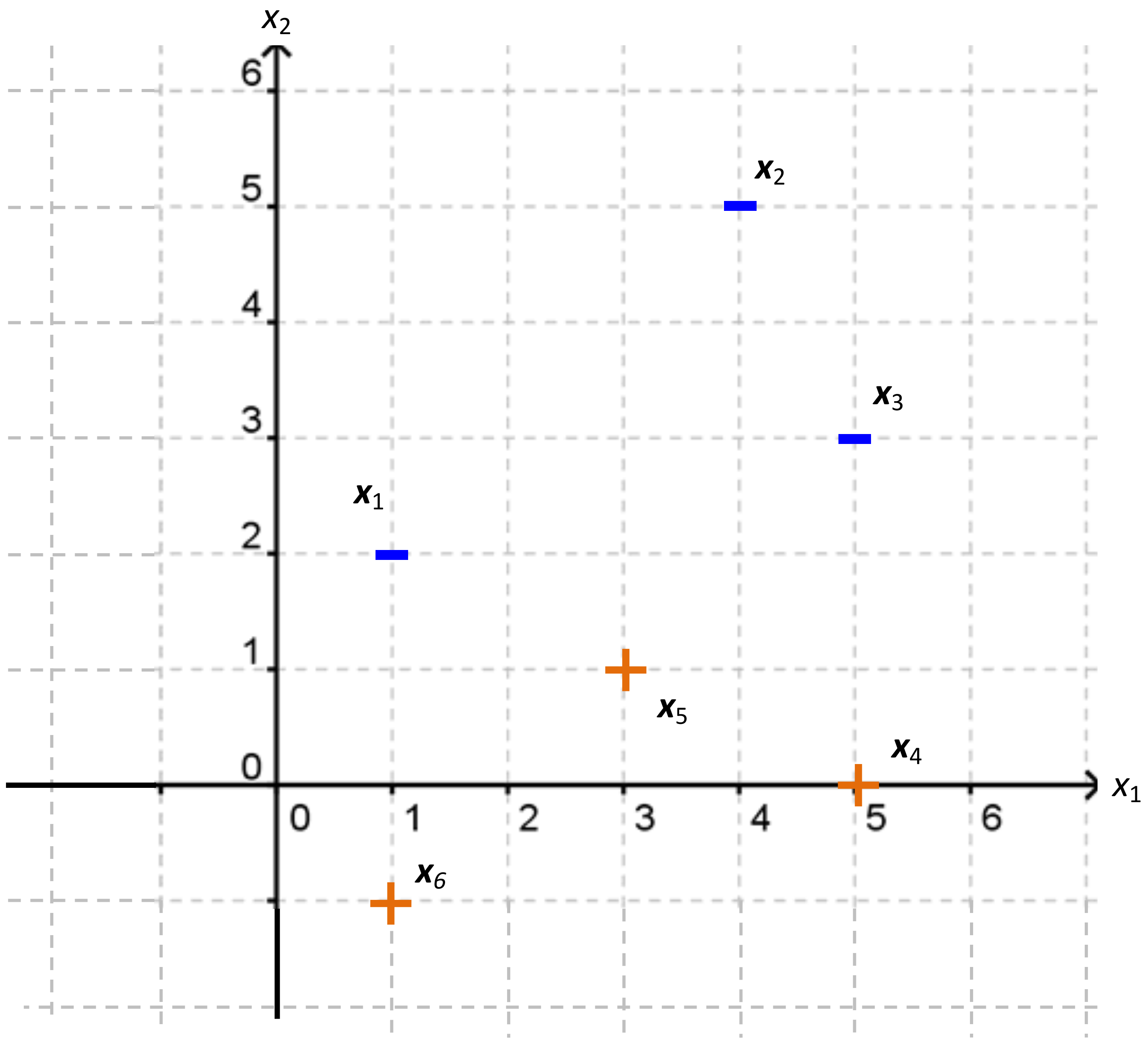
   [✓ (a) its Lagrange multiplier $\alpha_i > 0$]

   (b) its Lagrange multiplier $\alpha_i = 0$

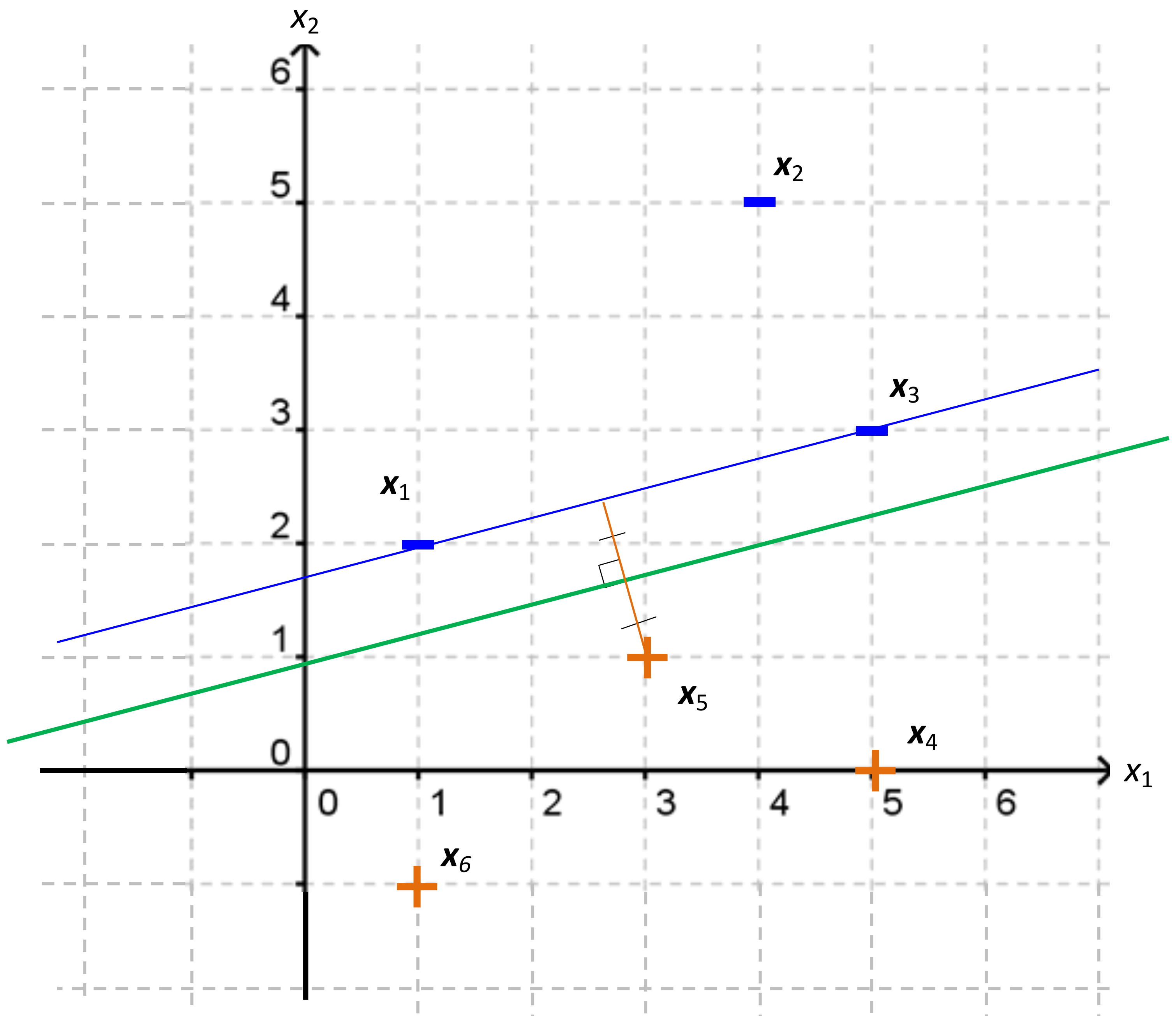
   (c) $y_i(\vec{w} \cdot \vec{x}_i + b) = 0$

   [✓ (d) $y_i(\vec{w} \cdot \vec{x}_i + b) = 1$]

   [✓ (e) $\vec{x}_i$ lies on the margin]



x_2
x_1
$\boldsymbol{x}_1$
$\boldsymbol{x}_2$
$\boldsymbol{x}_3$
$\boldsymbol{x}_4$
$\boldsymbol{x}_5$
$\boldsymbol{x}_6$
0 1 2 3 4 5 6



x_2
x_1
6
5
4
3
2
1
0
0
1
2
3
4
5
6
$\boldsymbol{x}_1$
$\boldsymbol{x}_2$
$\boldsymbol{x}_3$
$\boldsymbol{x}_4$
$\boldsymbol{x}_5$
$\boldsymbol{x}_6$

Reading Quiz #8

3. After training an SVM and obtaining the α values for each training example, I can use this formula to find the optimal weight vector:

$$\vec{w}^* = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$$

Then when I predict a label for a test example $\vec{x}$, I can use:

$$\hat{y} = h(\vec{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i (\vec{x}_i \cdot \vec{x}) + b\right)$$

Explain why it does not take $O(n)$ work to predict a label for each test point.

Reading Quiz #8

3. After training an SVM and obtaining the α values for each training example, I can use this formula to find the optimal weight vector:

$$\vec{w}^* = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i$$

Then when I predict a label for a test example $\vec{x}$, I can use:

$$\hat{y} = h(\vec{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i (\vec{x}_i \cdot \vec{x}) + b \right)$$

Explain why it does not take $O(n)$ work to predict a label for each test point.

Most of the alpha values are 0, so we only need to consider the support vectors!