4F10: Deep Learning and Structured Data

Support Vector Machines

José Miguel Hernández–Lobato
Department of Engineering
University of Cambridge

Michaelmas Term

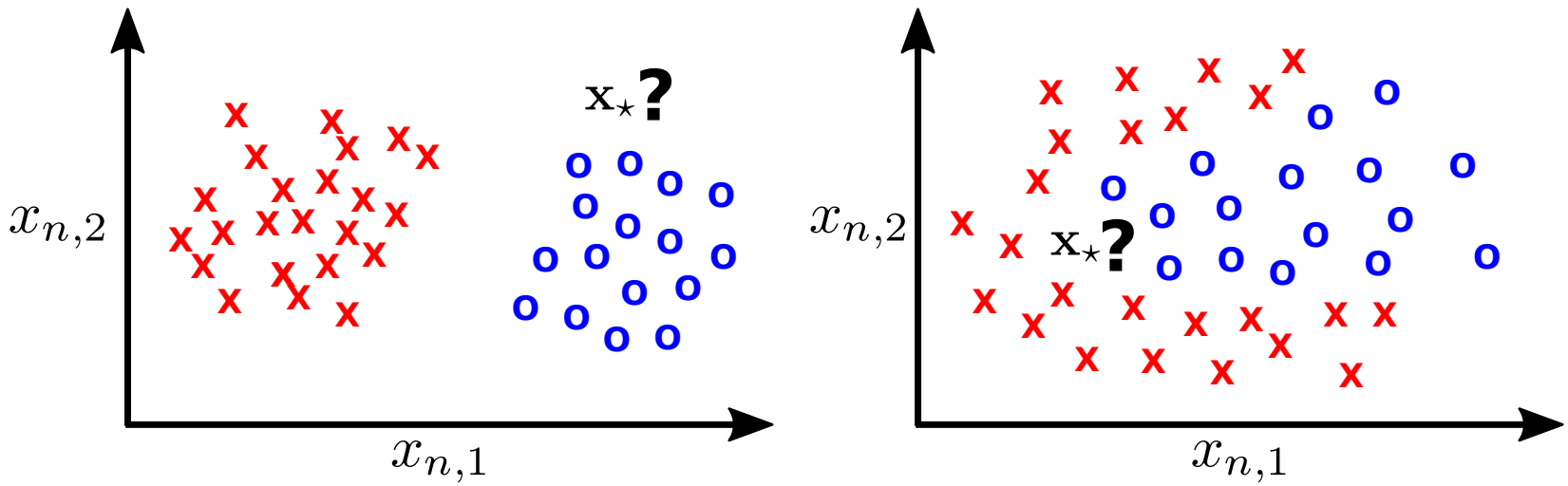
The Problem of binary classification

Given a dataset $\mathcal{D} = \{(\mathrm{x}_n, t_n)\}_{n=1}^N$, formed by pairs of input features $\mathrm{x}_n \in \mathbb{R}^d$ and target variables $t_n \in \{-1, 1\}$, we want to learn a classification function or **classifier** $y(\mathrm{x})$ such that

$$y(\mathrm{x}) \geq 0 \quad \text{if} \quad t_n = +1\,,$$
$$y(\mathrm{x}) < 0 \quad \text{if} \quad t_n = -1\,.$$

The classifier makes the correct prediction on a new input $\mathrm{x}_\star$ when $y(\mathrm{x}_\star)t_\star > 0$.

o $\rightarrow t_n = 1$ x $\rightarrow t_n = -1$



The **decision border** is the set of inputs for which $y(\mathrm{x}) = 0$.

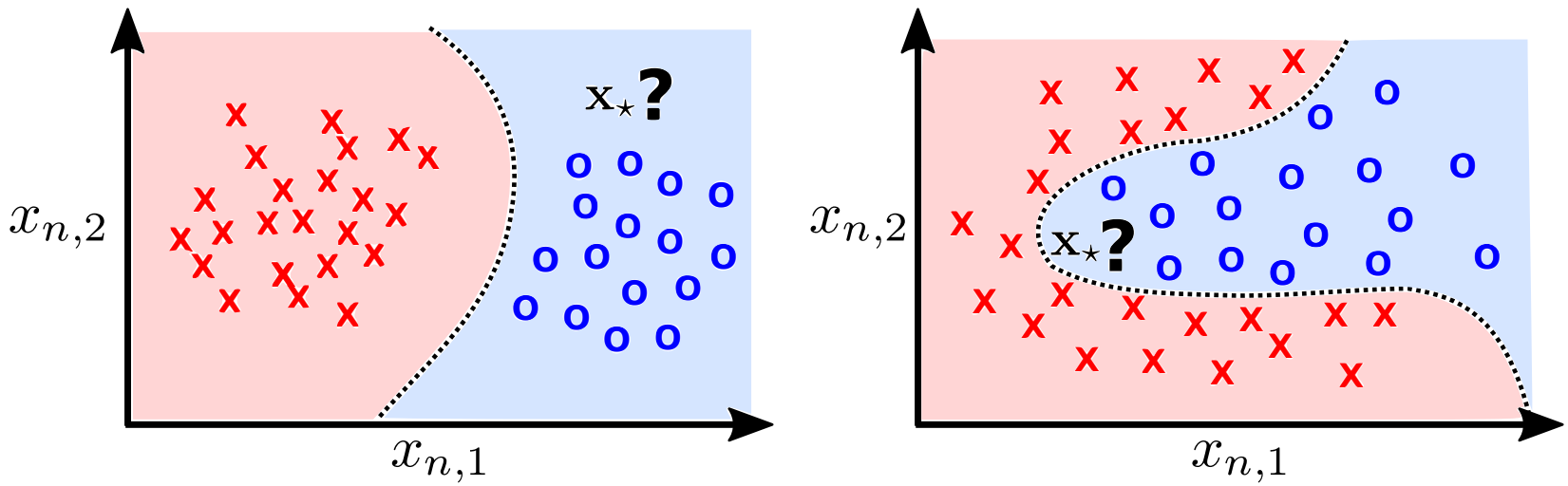
The Problem of binary classification

Given a dataset $\mathcal{D} = \{(\mathrm{x}_n, t_n)\}_{n=1}^N$, formed by pairs of input features $\mathrm{x}_n \in \mathbb{R}^d$ and target variables $t_n \in \{-1, 1\}$, we want to learn a classification function or **classifier** $y(\mathrm{x})$ such that

$$y(\mathrm{x}) \geq 0 \quad \text{if} \quad t_n = +1\,,$$
$$y(\mathrm{x}) < 0 \quad \text{if} \quad t_n = -1\,.$$

The classifier makes the correct prediction on a new input $\mathrm{x}_\star$ when $y(\mathrm{x}_\star)t_\star > 0$.

$$\mathbf{o} \rightarrow t_n = 1 \quad \mathbf{x} \rightarrow t_n = -1$$



The **decision border** is the set of inputs for which $y(\mathrm{x}) = 0$.

Linear separability

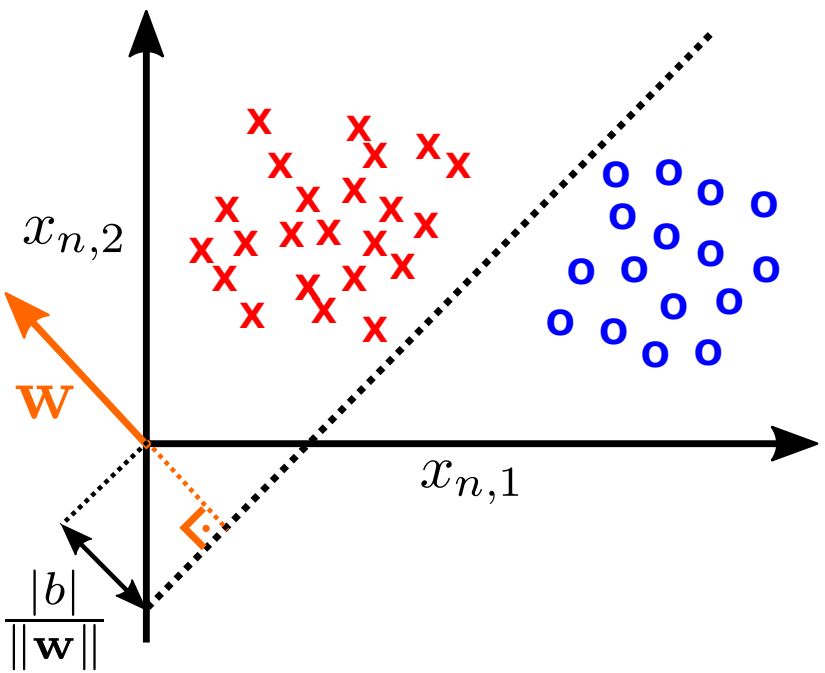
A classification problem is linearly separable when a classifier with a **linear decision border** makes no mistakes on the training data.

Linear classifier:

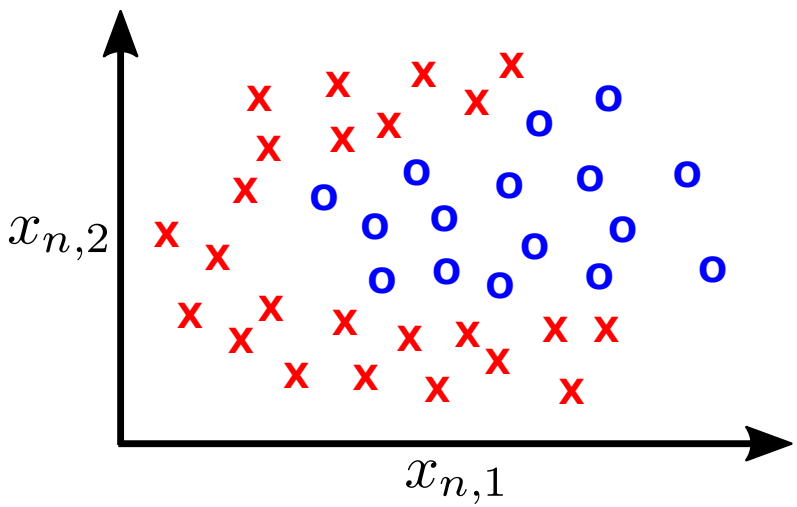
$$y(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x} + b\,,$$

$\mathbf{w}$ is a vector **orthogonal** to the decision border and b is the **bias** or intercept.

Linearly separable

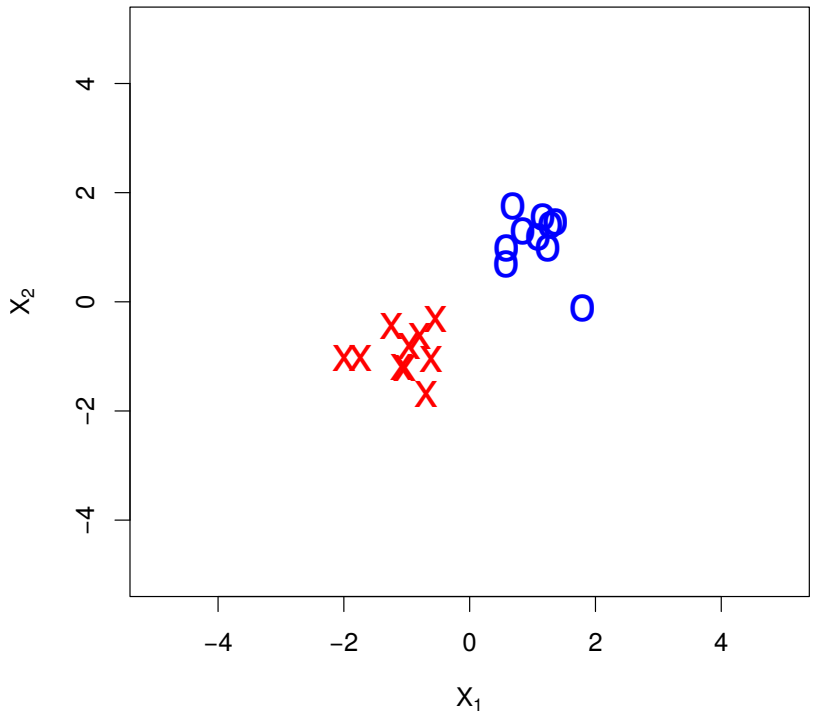


Not linearly separable



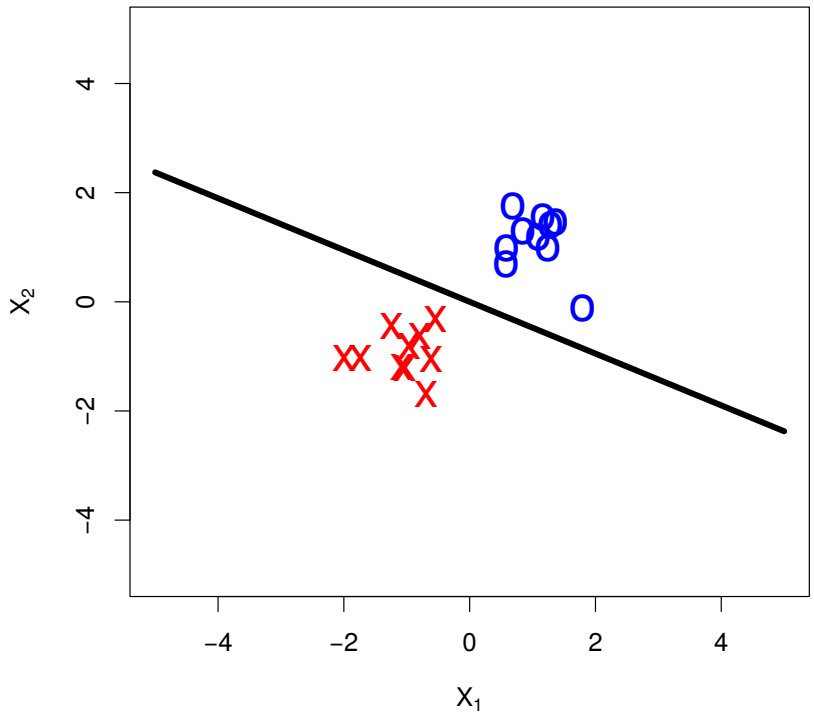
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



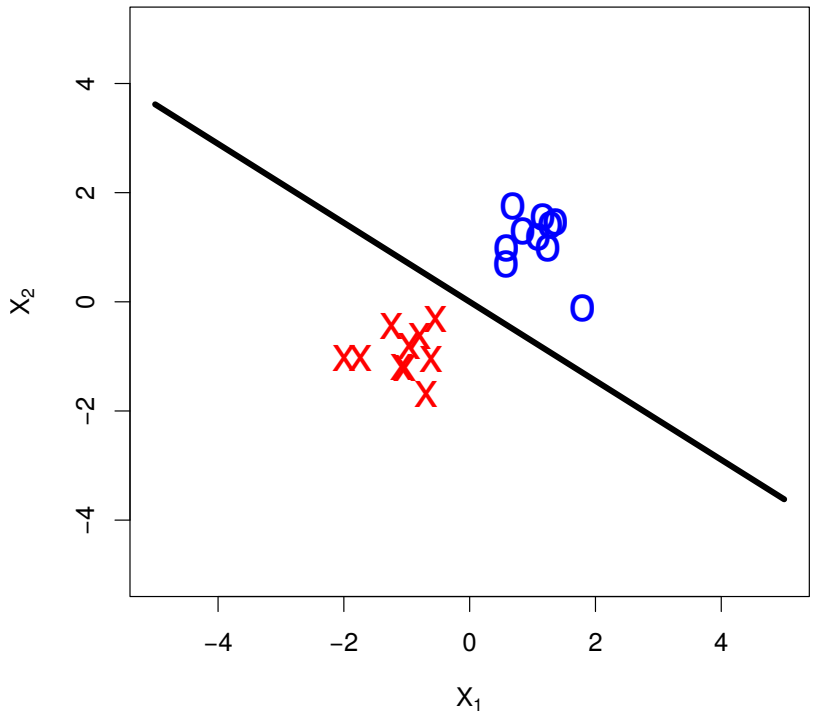
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



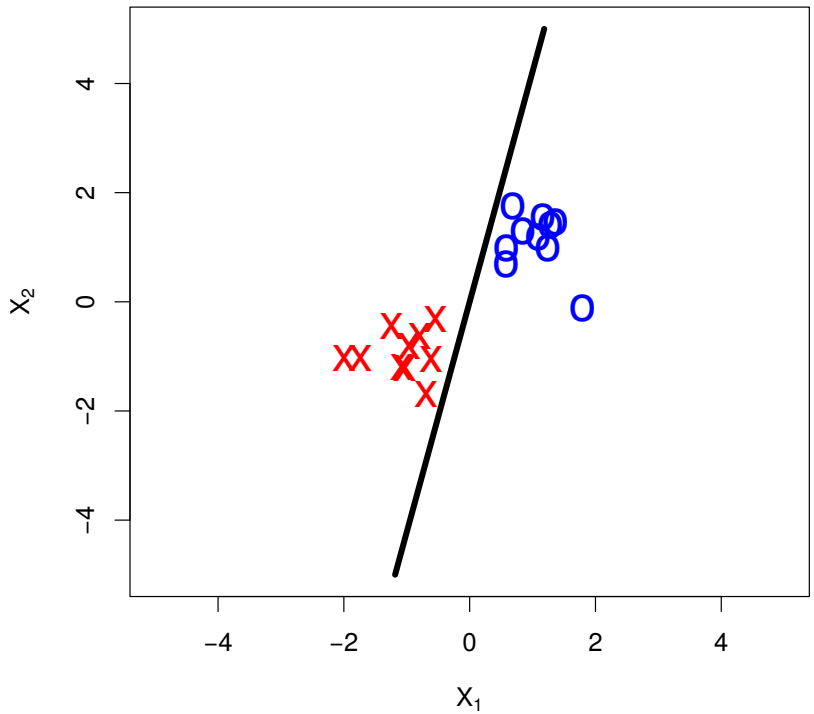
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



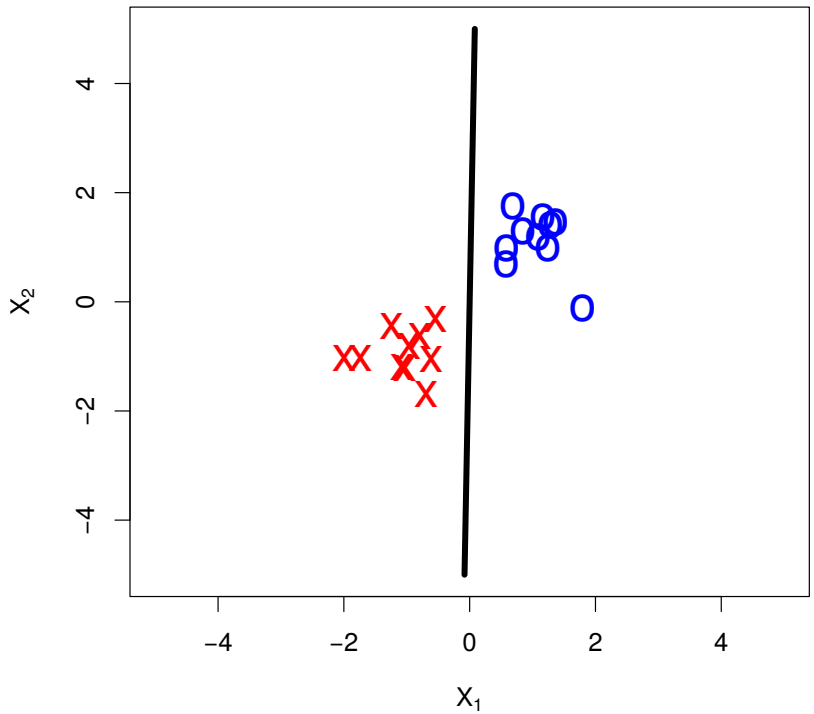
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



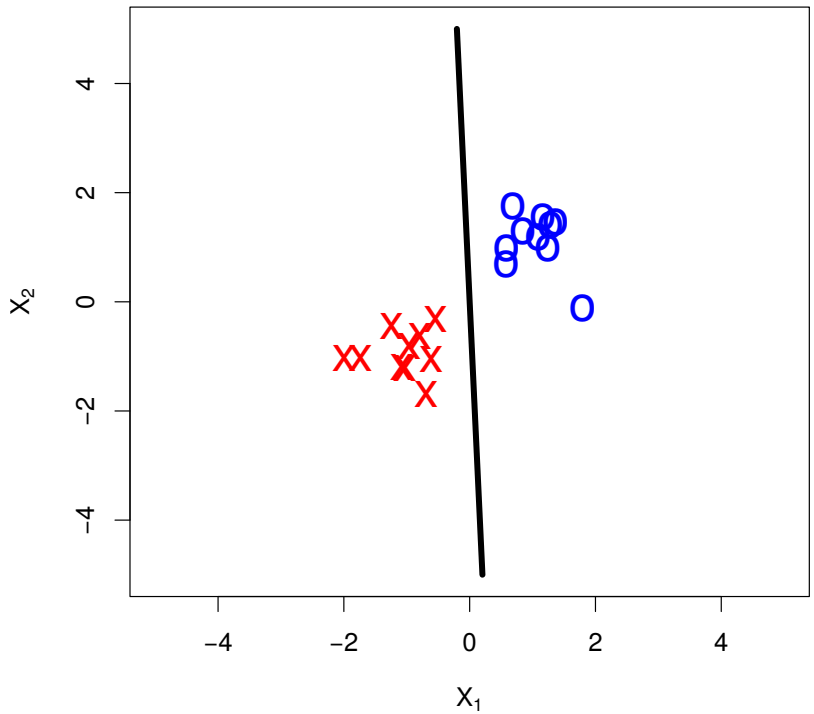
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



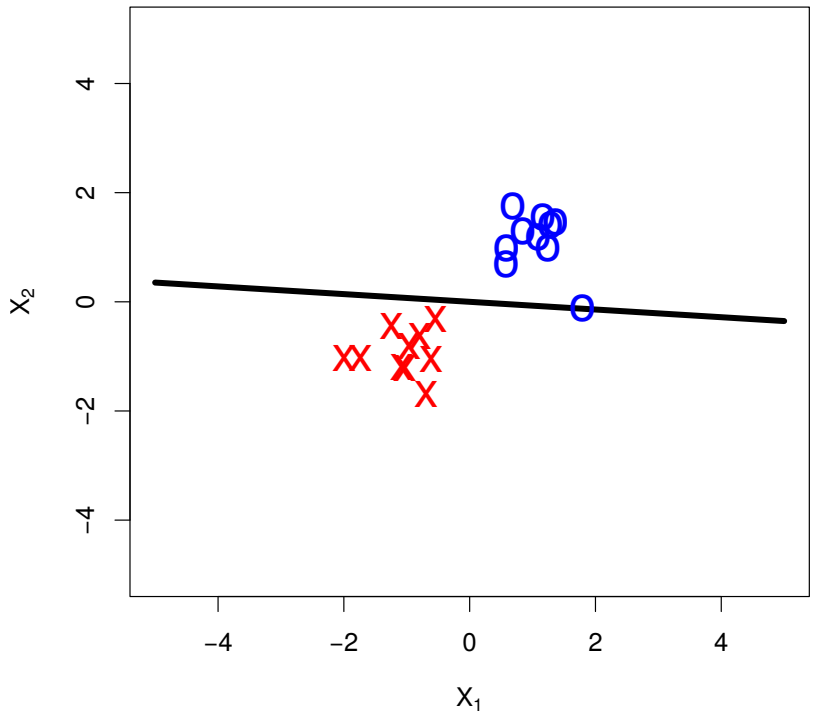
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



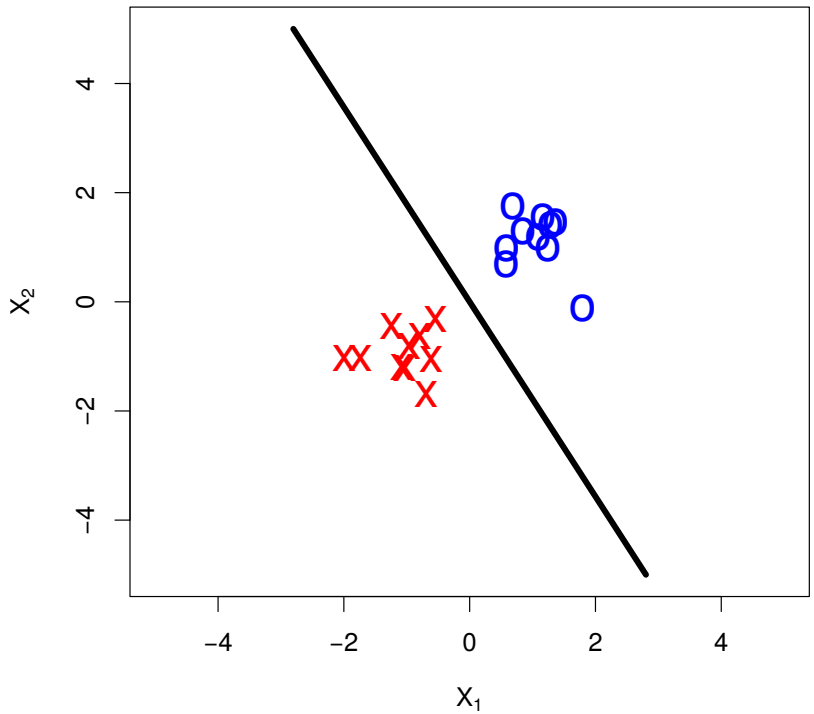
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



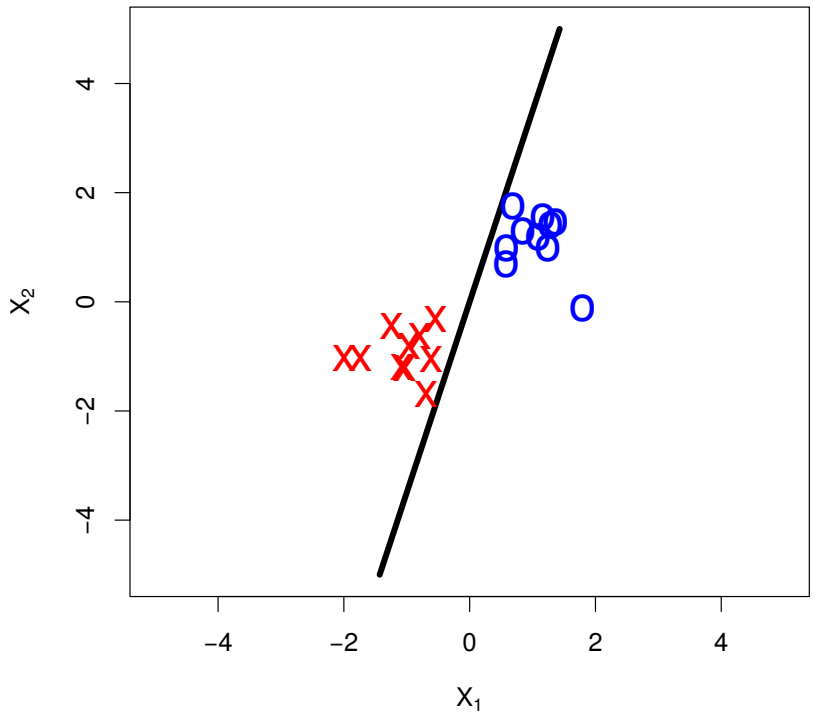
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



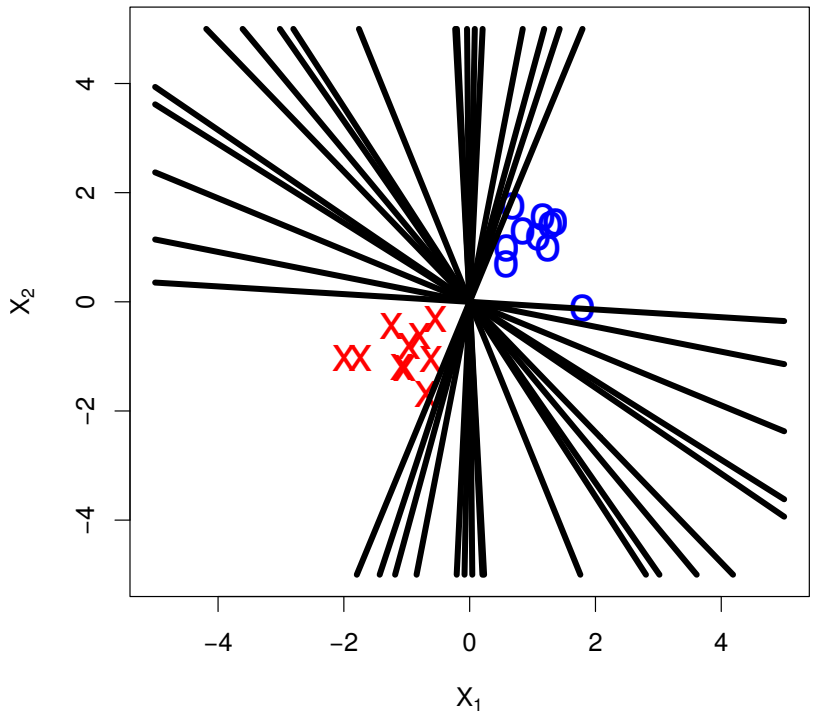
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



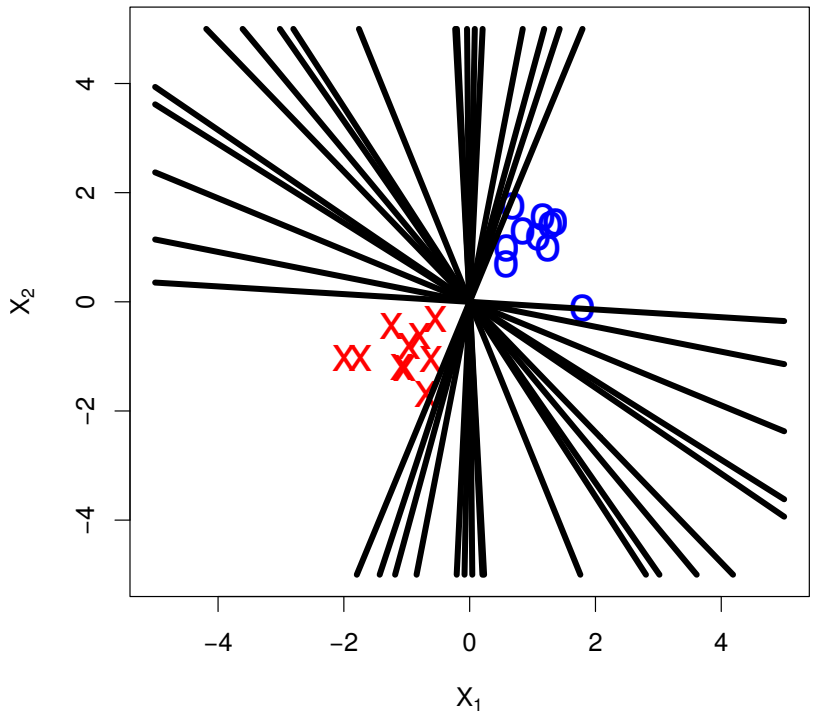
Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.



Maximum margin classifiers

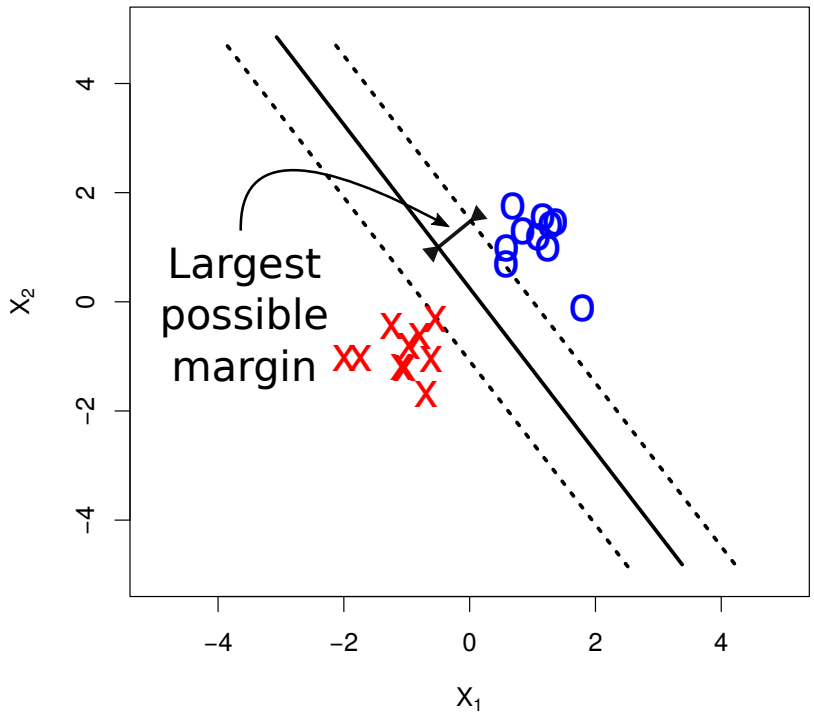
When the data is linearly separable, many possible w have zero training error.



Maximum margin idea: Choose plane whose distance to closest point in each class is maximal. This distance is called the **margin**.

Maximum margin classifiers

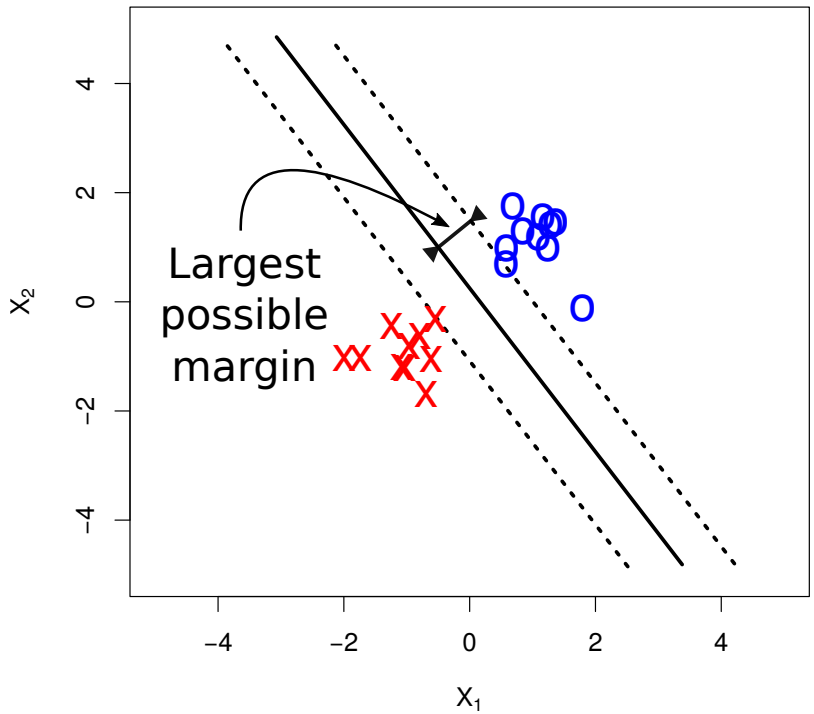
When the data is linearly separable, many possible w have zero training error.

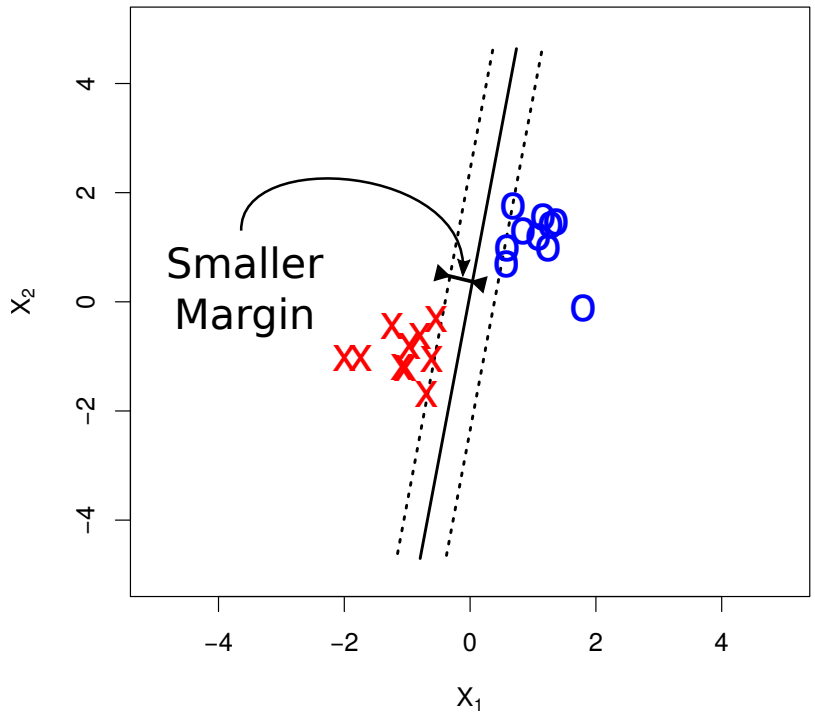


Maximum margin idea: Choose plane whose distance to closest point in each class is maximal. This distance is called the **margin**.

Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.

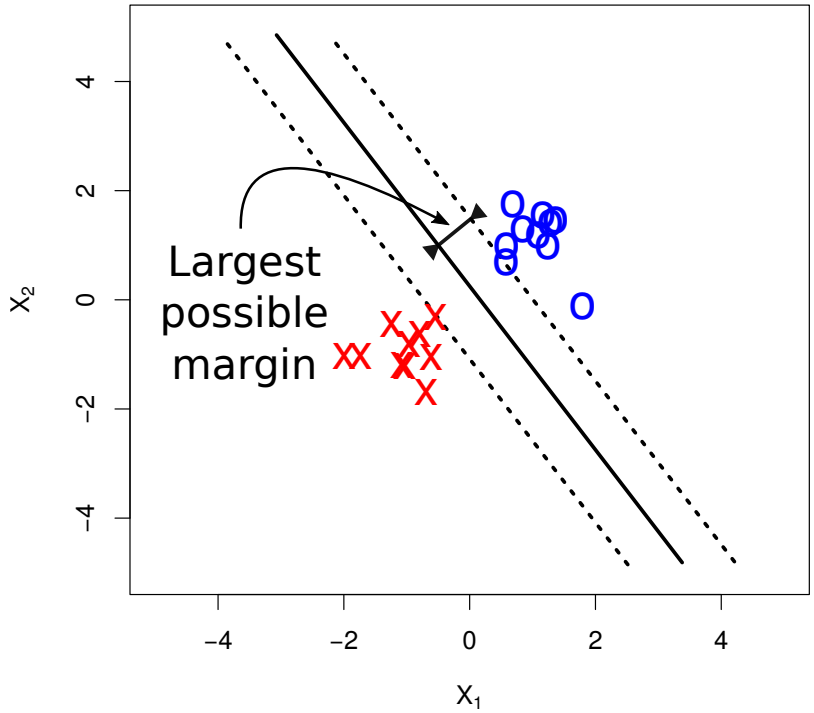


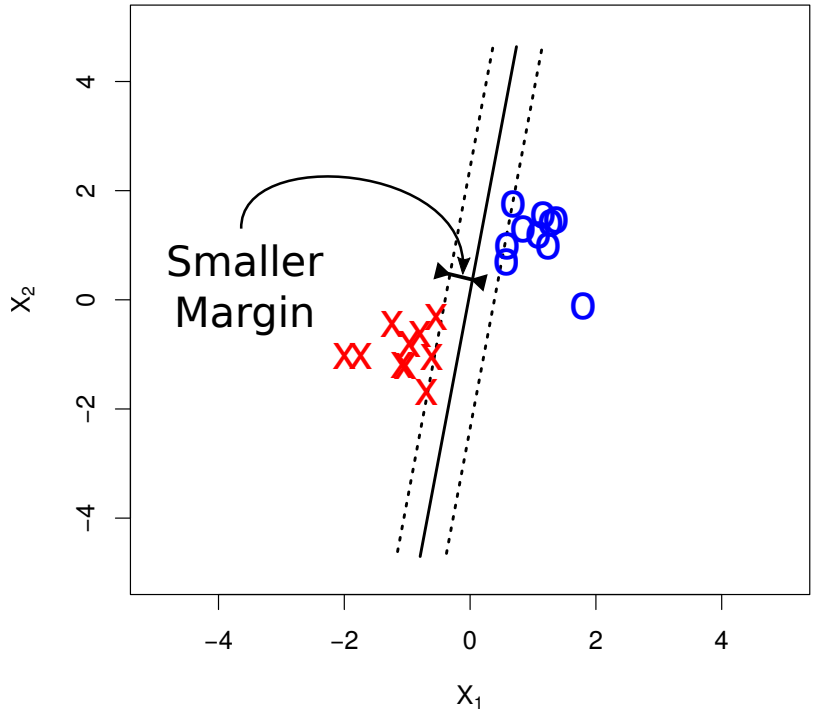


Maximum margin idea: Choose plane whose distance to closest point in each class is maximal. This distance is called the **margin**.

Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.





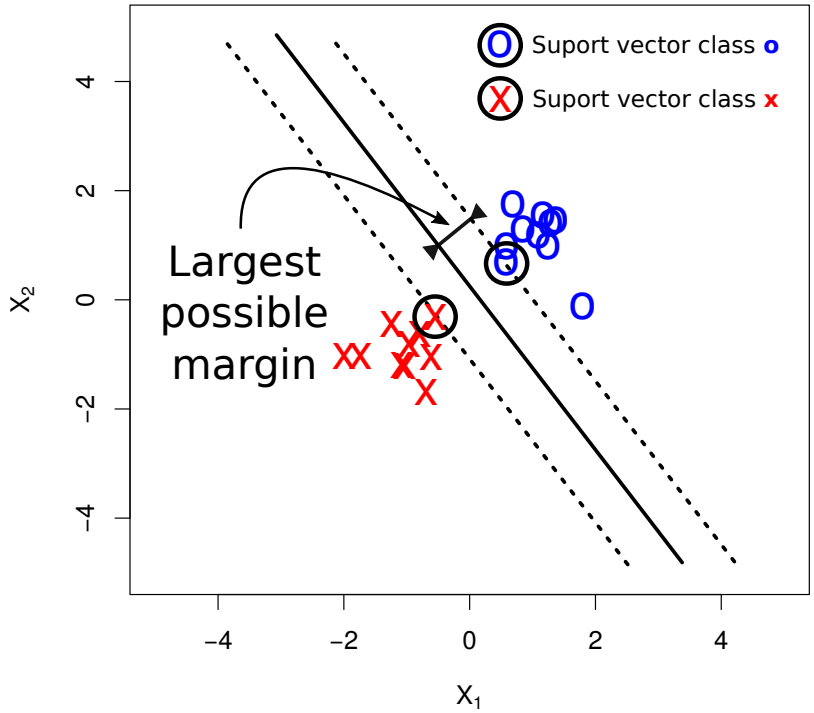
Maximum margin idea: Choose plane whose distance to closest point in each class is maximal. This distance is called the **margin**.

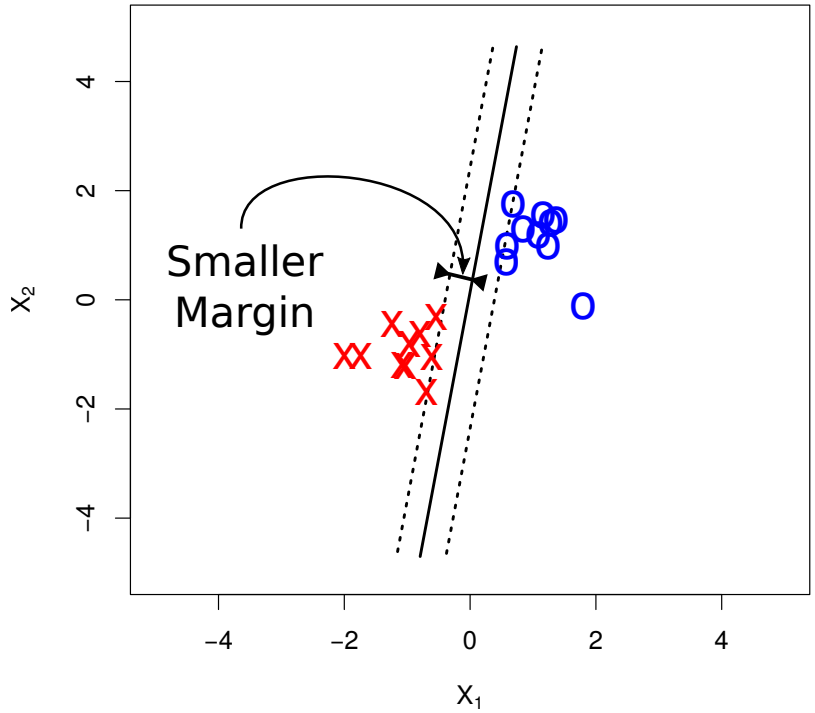
In a max-margin classifier, the data points on the margin are called **support vectors**. The max-margin classifier is fully determined by its support vectors.

Max margin-classifiers have some attractive theoretical justification.

Maximum margin classifiers

When the data is linearly separable, many possible w have zero training error.





Maximum margin idea: Choose plane whose distance to closest point in each class is maximal. This distance is called the **margin**.

In a max-margin classifier, the data points on the margin are called **support vectors**. The max-margin classifier is fully determined by its support vectors.

Max margin-classifiers have some attractive theoretical justification.

Computation of max-margin classifiers

The max-margin classifier or **support vector machine** (SVM) satisfies:

For any $c > 0$, any classifiers $y(\mathrm{x}) = \mathrm{w}^T\mathrm{x} + b$ and $y'(\mathrm{x}) = cy(\mathrm{x})$ have identical decision boundaries: the scale of the max-margin classifier is not important.

We can then choose the scale so that $y(\mathrm{x}_+) = +1$ for any positive support vector x_+. By symmetry, $y(\mathrm{x}_-) = -1$ for any negative support vector x_-. Therefore,

$$y(\mathrm{x}_+) = \mathrm{w}^\mathsf{T}\mathrm{x}_+ + b = +1\,,$$
$$y(\mathrm{x}_-) = \mathrm{w}^\mathsf{T}\mathrm{x}_- + b = -1\,,$$

The magnitude of the margin is then

$$\frac{\mathrm{w}^\mathsf{T}(\mathrm{x}_+ - \mathrm{x}_-)}{2||\mathrm{w}||} = \frac{(\mathrm{w}^\mathsf{T}\mathrm{x}_+ + b - \mathrm{w}^\mathsf{T}\mathrm{x}_- - b)}{2||\mathrm{w}||} = \frac{1}{||\mathrm{w}||}\,. \tag{1}$$

At the solution, **(1) is maximal** and the data is correctly classified with the minimum value of $|y(\mathrm{x})|$ achieved by support vectors. Therefore, $t_n y(\mathrm{x}_n) \geq 1\ \forall n$.

Example solution

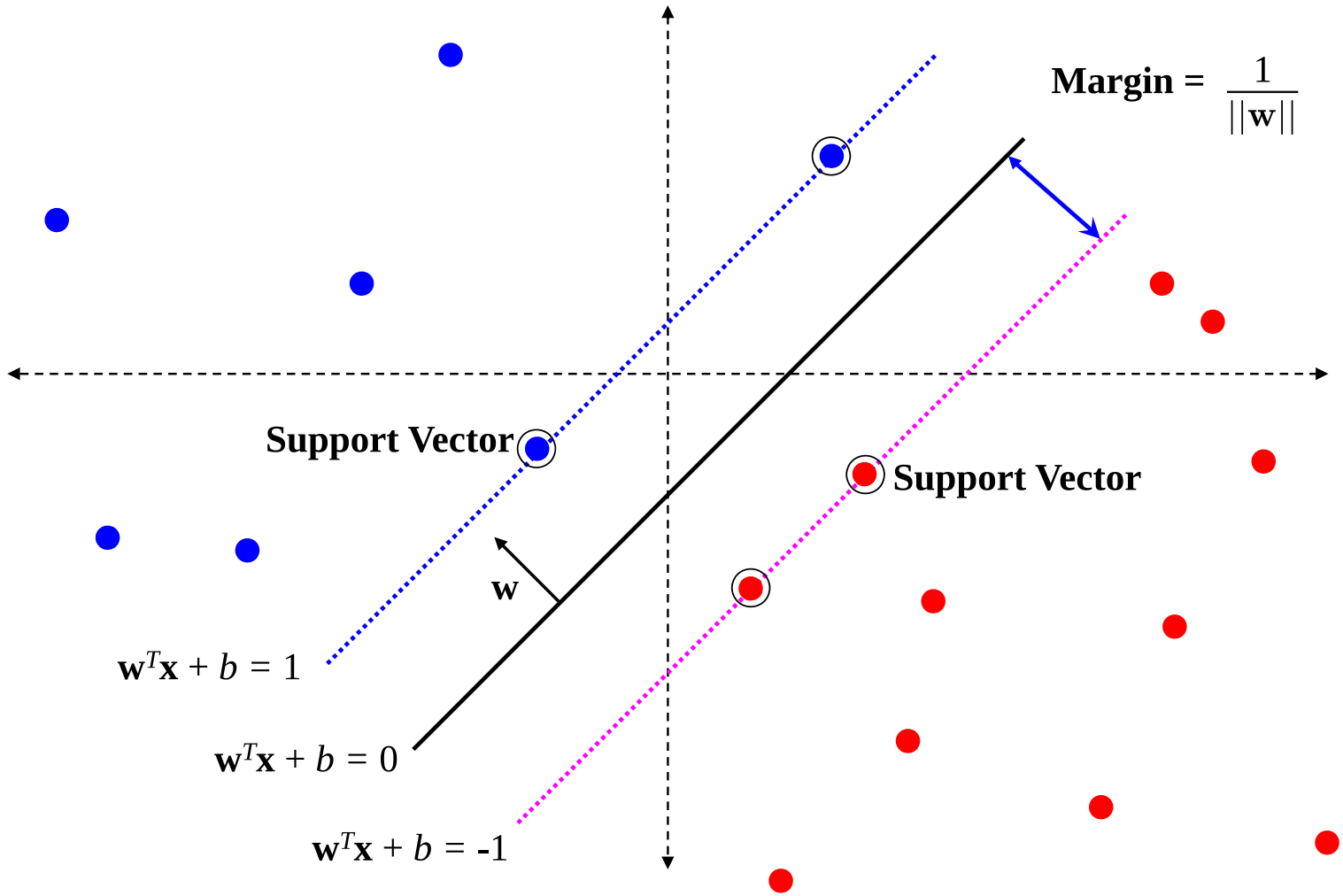


Figure source: A. Zisserman

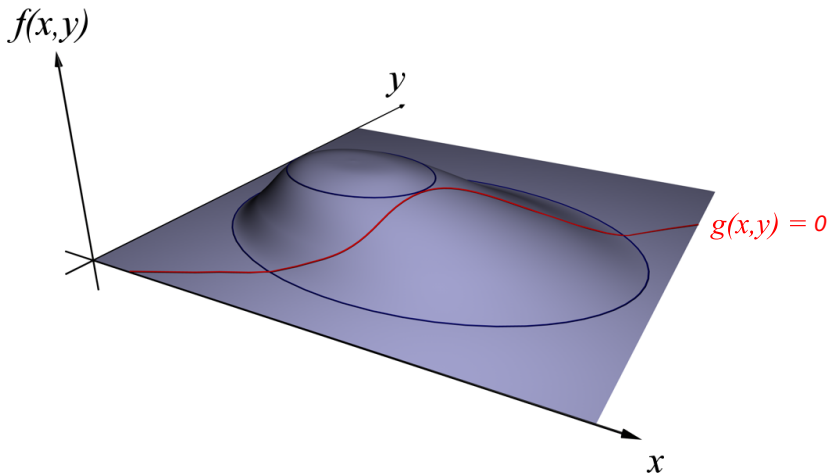
Optimization with inequality constraints

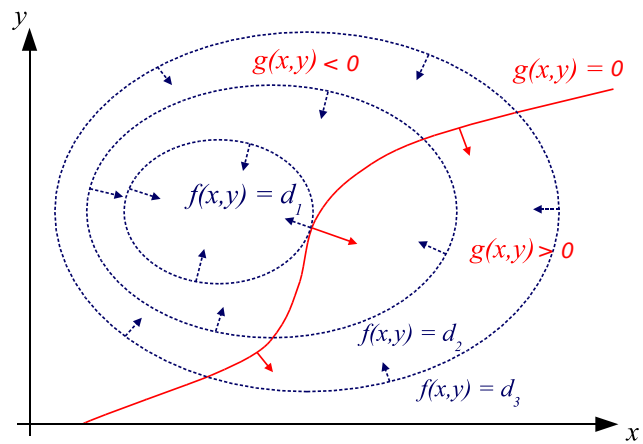
Consider the constrained optimization problem

$$\text{maximize} \quad f(x,y) \quad \text{subject to} \quad g(x,y) \geq 0\,. \tag{2}$$

Let $x_\star$ and $y_\star$ be the solution to this problem.

Consider the constraint is instead $g(x,y) = 0$. We can then use Lagrange multipliers.





The solution must satisfy conditions $\nabla_{x,y} f(x,y) = -\lambda \nabla_{x,y} g(x,y)$ and $g(x,y) = 0$.

Achieved by optimizing with respect to x, y and λ the **Lagrangian** function:

$$\mathcal{L}(x,y,\lambda) = f(x,y) + \lambda g(x,y)\,, \tag{3}$$

where $\lambda \neq 0$ is called the **Lagrange multiplier**, which could be **positive or negative**.

Karush-Kuhn-Tuker (KKT) conditions

When the constraint is $g(x, y) \geq 0$, we have two possibilities:

1. **Constraint is not active at solution**: $g(x_\star, y_\star) > 0$.

   Solution satisfies $\nabla_{x,y} f(x, y) = 0$. At the solution, the Lagrangian will also satisfy $\nabla_{x,y} \mathcal{L}(x, y, \lambda) = 0$, but only if $\lambda = 0$.

2. **Constraint is active at solution**: $g(x_\star, y_\star) = 0$.

   Stationary point of Lagrangian with $\lambda \neq 0$, that is, $\nabla_{x,y} f(x, y) = -\lambda \nabla_{x,y} g(x, y)$.

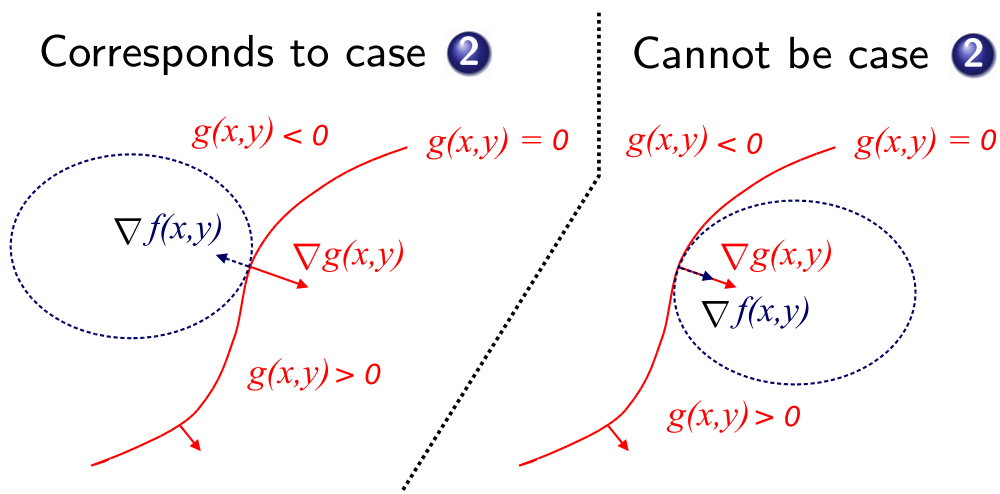
   λ **must be positive**: $\lambda > 0$, otherwise, solution located in region $g(x, y) > 0$ and constraint would not be tight.

Solution to (2) obtained by solving $\max_{x,y} \min_{\lambda} \mathcal{L}(x, y, \lambda)$ s.t. $\lambda \geq 0$.

In both cases above, $\lambda g(x, y) = 0$ holds.

Solution to (2) satisfies **KKT conditions**:

$$g(x, y) \geq 0\,, \quad \lambda \geq 0\,, \quad \lambda g(x, y) = 0\,,$$
$$\nabla_{x,y} f(x, y) = -\lambda \nabla_{x,y} g(x, y)\,.$$



Optimization problem

Since maximizing $1/||\mathrm{w}||$ is the same as minimizing $1/2||\mathrm{w}||^2$, the optimization problem can be written as

$$\min \quad \frac{1}{2}||\mathrm{w}||^2 \quad \text{subject to} \quad t_n(\mathrm{w}^{\mathsf{T}}\mathrm{x}_n + b) \geq 1\,, \quad n = 1, \ldots, N.$$

This is a quadratic problem with linear constraints: **there is a unique minimum**.

By introducing **Lagrange multipliers** $a_1 \geq 0, \ldots, a_N \geq 0$ we obtain the objective

$$L(\mathrm{w}, b, \mathrm{a}) = \frac{1}{2}||\mathrm{w}||^2 - \sum_{n=1}^{N} a_n \left\{ t_n(\mathrm{w}^{\mathsf{T}}\mathrm{x}_n + b) - 1 \right\} . \tag{4}$$

The negative sign in a_n appears because we will be maximizing w.r.t. each a_n.

Setting gradients with respect to w and b to zero results in the following expressions:

$$\mathrm{w} = \sum_{n=1}^{N} a_n t_n \mathrm{x}_n\,, \quad 0 = \sum_{n=1}^{N} a_n t_n\,.$$

Using these expressions to eliminate w and b from (4) leads to the **dual problem**:

$$\max_{\mathrm{a}} \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m \mathrm{x}_n^{\mathsf{T}}\mathrm{x}_m \quad \text{s.t.} \quad \sum_{n=1}^{N} a_n t_n = 0\,, \quad \{a_n \geq 0\}_{n=1}^{N}\,.$$

Solution and prediction formula

Need to optimize a **quadratic function** subject to **inequality constraints**.

No analytic solution, numerical methods have cost between $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$. SVMs are state-of-the-art methods for $N \leq 50,000$, otherwise too expensive!

Software for training SVMs: `http://www.support-vector-machines.org`.

From KKT conditions, at convergence, it holds that $a_n(t_n y(\mathbf{x}_n) - 1) = 0$, $\forall n$.

Hence, data points with $a_n > 0$ satisfy $t_n y(\mathbf{x}_n) = 1$ and are **support vectors**.

After training, **predictions** can be made using

$$y(\mathbf{x}) = \sum_{n \in \mathcal{S}} a_n t_n \mathbf{x}_n^{\mathsf{T}} \mathbf{x} + b\,, \tag{5}$$

with $\mathcal{S}$ being the set of indexes of support vectors: $\mathcal{S} = \{n : a_n > 0\}$.

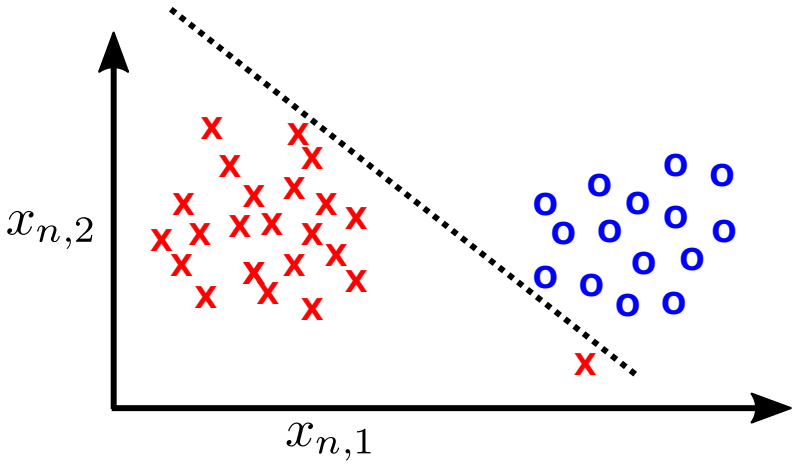
The bias b can be obtained from the set of constraints $\{t_m y(\mathbf{x}_m) = 1 : m \in \mathcal{S}\}$:

$$b = \frac{1}{|\mathcal{S}|} \sum_{m \in \mathcal{S}} \left\{ t_m - \sum_{n \in \mathcal{S}} a_n t_n \mathbf{x}_n^{\mathsf{T}} \mathbf{x}_m \right\}.$$

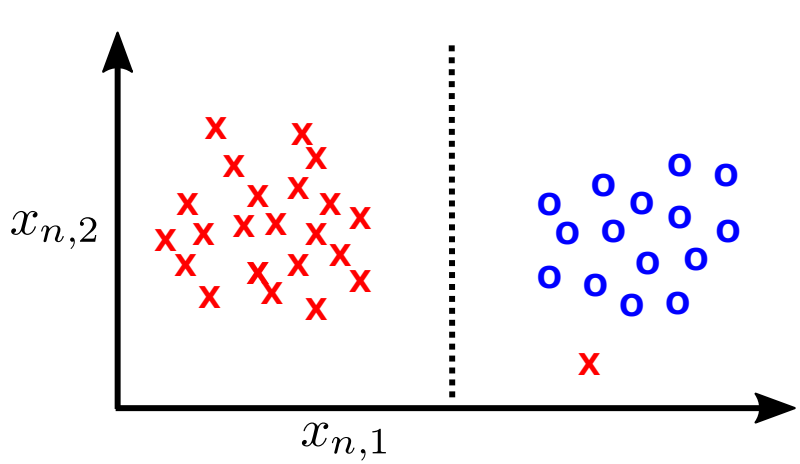
Numerical stability is improved by averaging the solution for b given by each constraint.

Constraint violation and soft margin

Sometimes allowing for mistakes in the training data produces better classifiers.



The data points are linear separable, but the margin is very small.



We obtain a larger margin by ignoring the misclassified data point.

There is a **trade-off** between the margin size and the number of training errors.

Errors are allowed by introducing **slack variables** $\xi_1, \ldots, \xi_N$ in the constraints:

$$t_n(\mathbf{w}^\mathsf{T}\mathbf{x}_n + b) \geq 1 - \xi_n\,, \quad \xi_n \geq 0\,, \quad n = 1, \ldots, N\,,$$

Points with $\xi_n = 0$ are correctly classified and are either on the margin or beyond.
Points with $0 \leq \xi_n \leq 1$ lie inside the margin, but on the correct side.
Points with $\xi_n > 1$ are misclassified.

Example

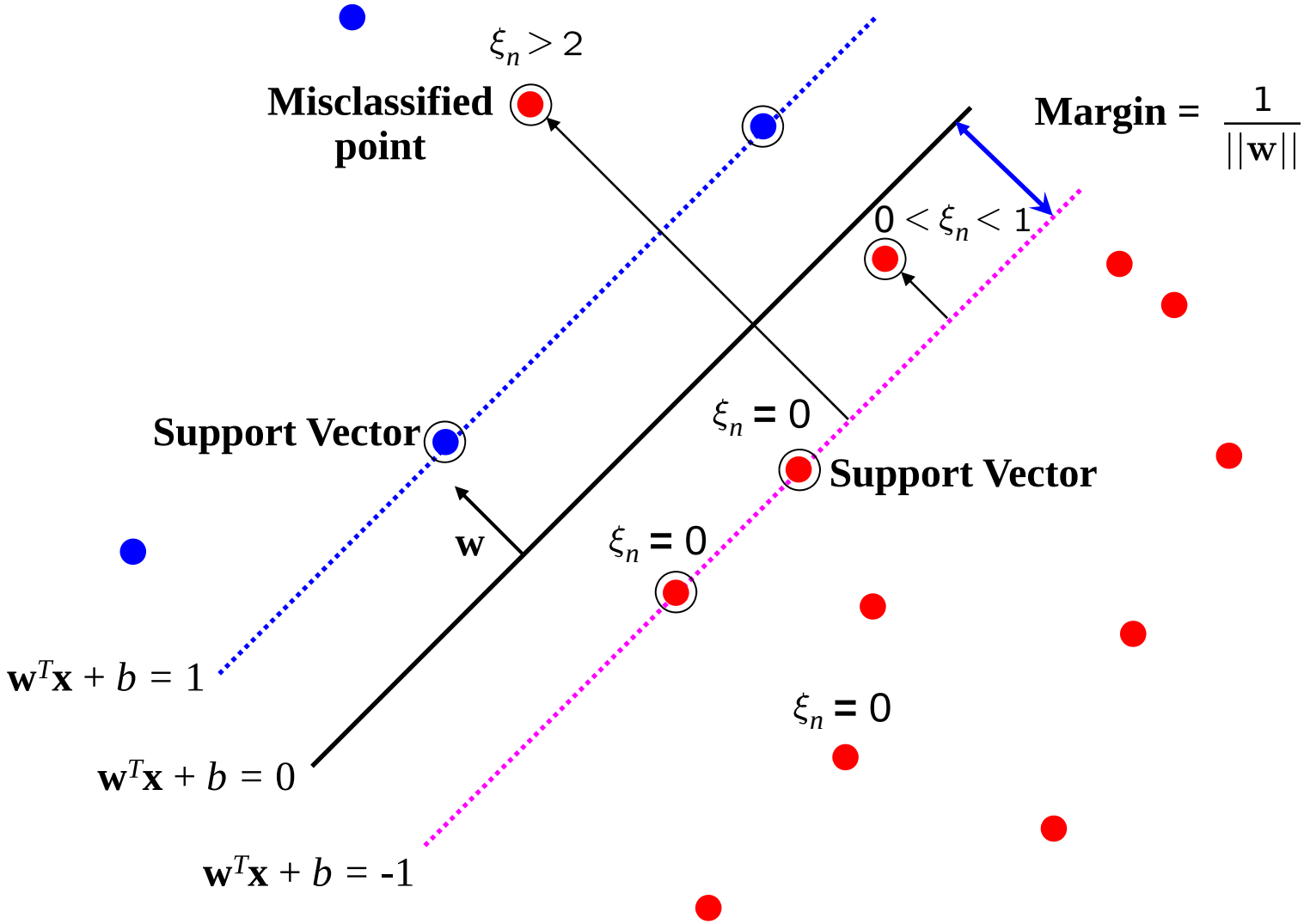


Figure source: A. Zisserman

Optimization problem for soft margin

We now solve

$$\min \quad \frac{1}{2}||\mathrm{w}||^2 + C\sum_{n=1}^{N}\xi_n \quad \text{s.t.} \quad t_n(\mathrm{w}^{\mathsf{T}}\mathrm{x}_n + b) \geq 1 - \xi_n\,, \quad \xi_n \geq 0\,, \quad n = 1, \ldots, N.$$

$C > 0$ controls the **trade-off** between the slack variable penalty and the margin.
Small C: soft constraints, large margin. Large C: hard constraints, narrow margin.

By introducing **Lagrange multipliers** $\{a_n \geq 0\}_{n=1}^{N}$, $\{\mu_n \geq 0\}_{n=1}^{N}$ we obtain

$$L(\mathrm{w}, b, \mathrm{a}) = \frac{1}{2}||\mathrm{w}||^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}a_n\left\{t_n(\mathrm{w}^{\mathsf{T}}\mathrm{x}_n + b) - 1 + \xi_n\right\} - \sum_{n=1}^{N}\mu_n\xi_n\,.$$

Setting gradients with respect to w, b and $\{\xi_n\}$ to zero:

$$\mathrm{w} = \sum_{n=1}^{N}a_n t_n \mathrm{x}_n\,, \quad 0 = \sum_{n=1}^{N}a_n t_n \quad \{a_n = C - \mu_n\}_{n=1}^{N}\,.$$

Using this to eliminate w, b and $\{\xi_n\}$ from $L(\mathrm{w}, b, \mathrm{a})$ leads to the **dual problem**:

$$\max_{\mathrm{a}} \sum_{n=1}^{N}a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}a_n a_m t_n t_m \mathrm{x}_n^{\mathsf{T}}\mathrm{x}_m \quad \text{s.t.} \quad \sum_{n=1}^{N}a_n t_n = 0\,, \quad \{0 \leq a_n \leq C\}_{n=1}^{N}\,.$$

Prediction formula, support vectors and bias

Again, we solve a **quadratic function** subject to **inequality constraints**.

After training, **predictions** can be made using (5), as before.

At convergence, KKT conditions indicate that $a_n(t_n y(\mathrm{x}_n) - 1 + \xi_n) = 0$, $\forall n$.

Hence, data points with $a_n > 0$ satisfy $t_n y(\mathrm{x}_n) = 1 - \xi_n$ and are **support vectors**.

If $0 < a_n < C$, then $\mu_n > 0$ and $\xi_n = 0$ (from KKT conditions) and hence, such points lie on the margin.

Points with $a_n = C$ can lie inside the margin boundaries and can either be

- correctly classified if $\xi_n \leq 1$
- misclassified if $\xi_n > 1$.

b obtained from constraints $\{t_n y(\mathrm{x}_n) = 1 - \xi_n : n \in \mathcal{S}\}$ with $\xi_n = 0$ $(0 < a_n < C)$:

$$b = \frac{1}{|\mathcal{M}|} \sum_{n \in \mathcal{M}} \left\{ t_n - \sum_{m \in \mathcal{S}} a_m t_m \mathrm{x}_m^{\mathsf{T}} \mathrm{x}_n \right\}.$$

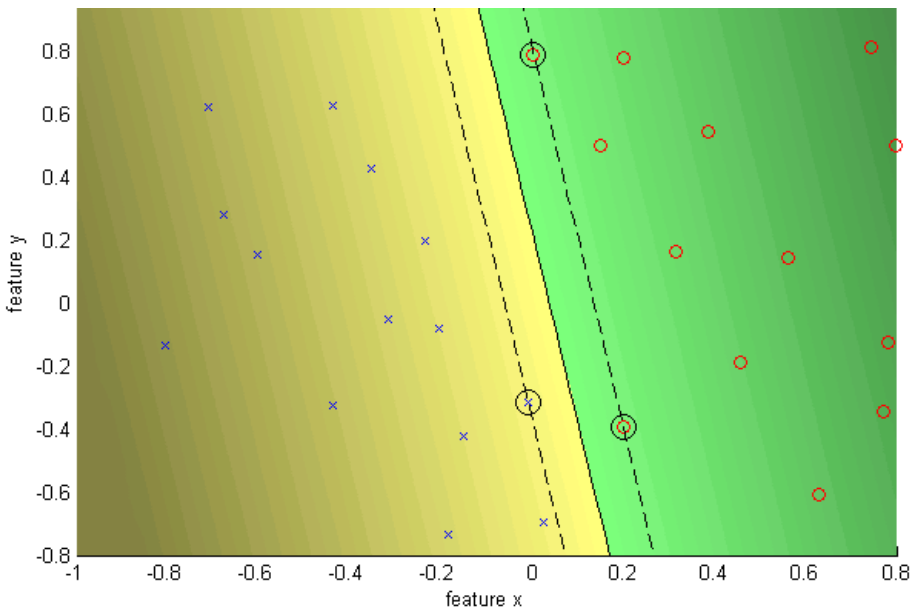
We improve numerical stability by averaging the solution for b given by each constraint.

where $\mathcal{M}$ contains the indexes of data points with $0 < a_n < C$.

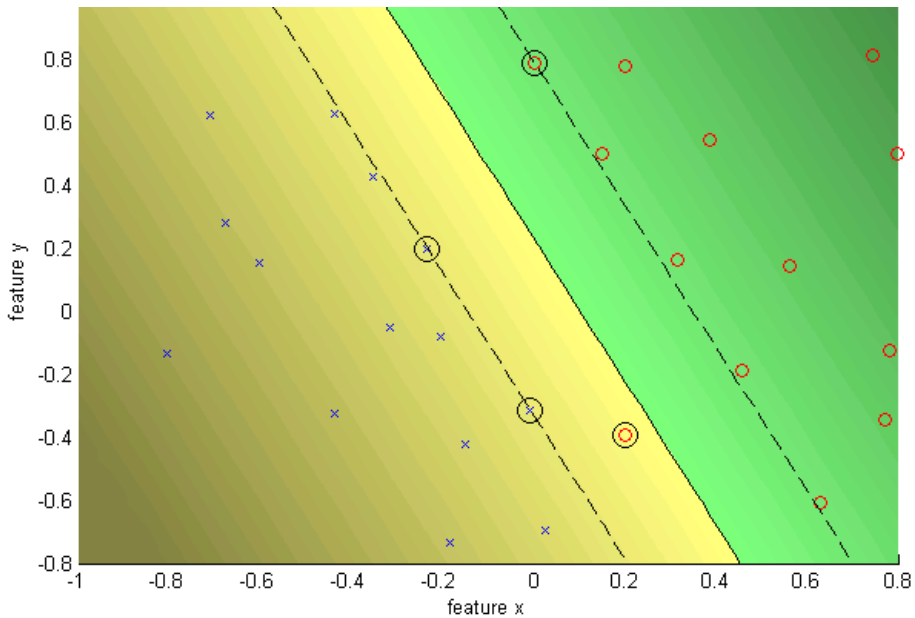
Example

Data linearly separable but only with a narrow margin.

$C = \infty$, hard margin.
No training errors.



$C = 10$, soft margin.
One training error.



How to choose the optimal value for C in practice?

Solution: try many and choose one with optimal predictions on a validation set.

Figure source: A. Zisserman

Summary

When the data is linearly separable, many possible w have zero training error.

Max-margin classifier (MMC) has maximal distance to closest point in each class.

MMCs are theoretically justified and are expected to have low prediction error.

The MMC is found by solving a quadratic problem with inequality constraints.

Lagrange multipliers and the KKT conditions can be used to analyze the solution.

Slack variables can be used to obtain soft MMCs that allow for training errors.

Typically, there is a trade-off between # of training errors and margin size.