INFSCI 2595

Fall 2019

Information Sciences Building: Room 403

Week 10: Support Vector Machines

With logistic regression we learned how to classify an event by modeling the <u>event probability</u>

$$y_n \mid \text{size} = 1, \mu_n \sim \text{Bernoulli}(y_n \mid \mu_n)$$

$$\mu_n = \text{logit}^{-1}(\eta_n)$$

$$\eta_n = \mathbf{x}_{n,:}\boldsymbol{\beta} \text{ or } \eta_n = \boldsymbol{\phi}_{n,:}\boldsymbol{\beta}$$

We've discussed several different prior formulations on the unknown linear predictor parameters $\boldsymbol{\beta}$

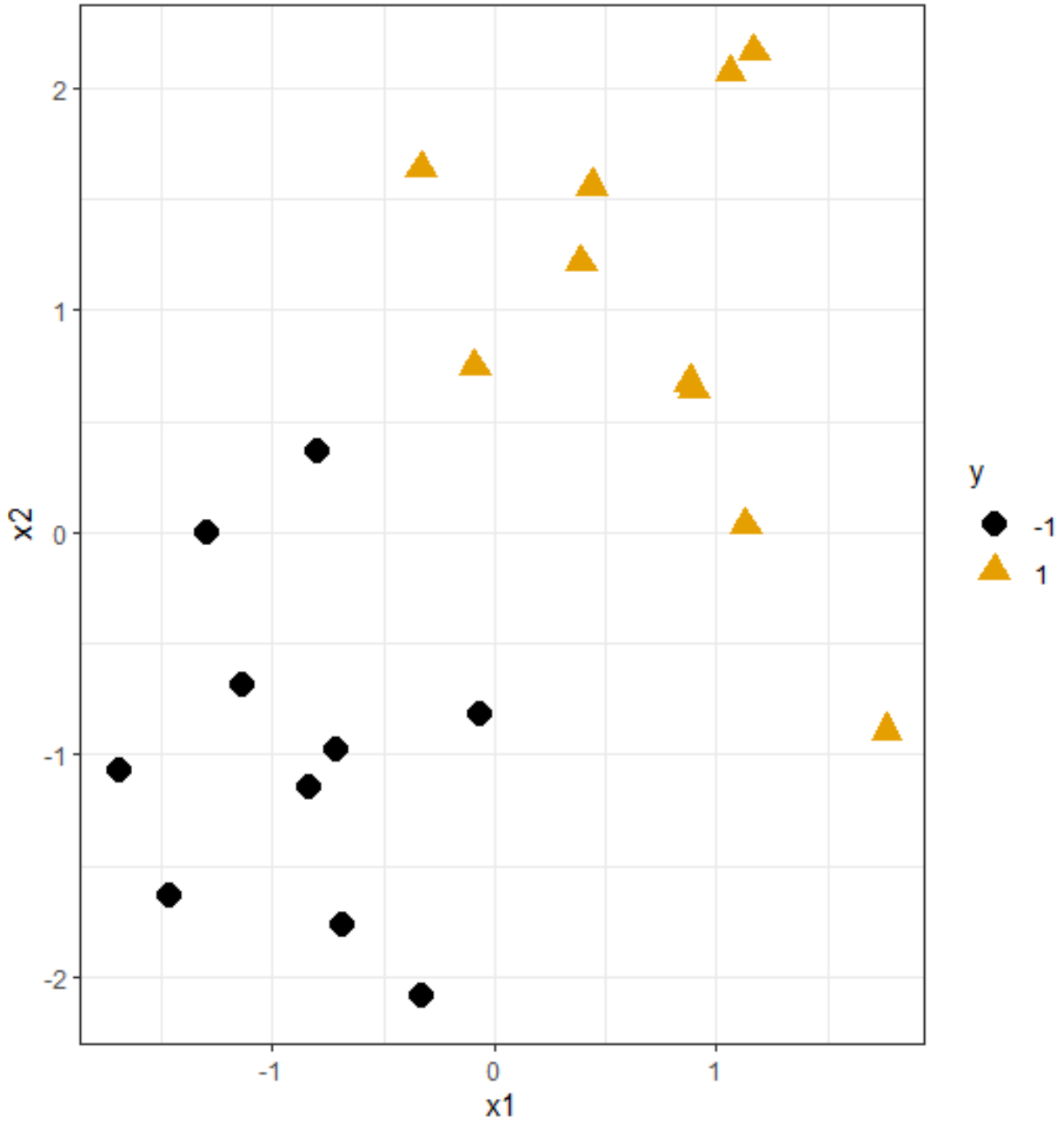
Let’s revisit the classification problem…

- We still have a binary outcome, but this time we will denote the two classes as $y = \{-1, +1\}$ instead of $y = \{0,1\}$.

- Rather than considering the probability of the event, let’s step through the “geometry” of the problem…

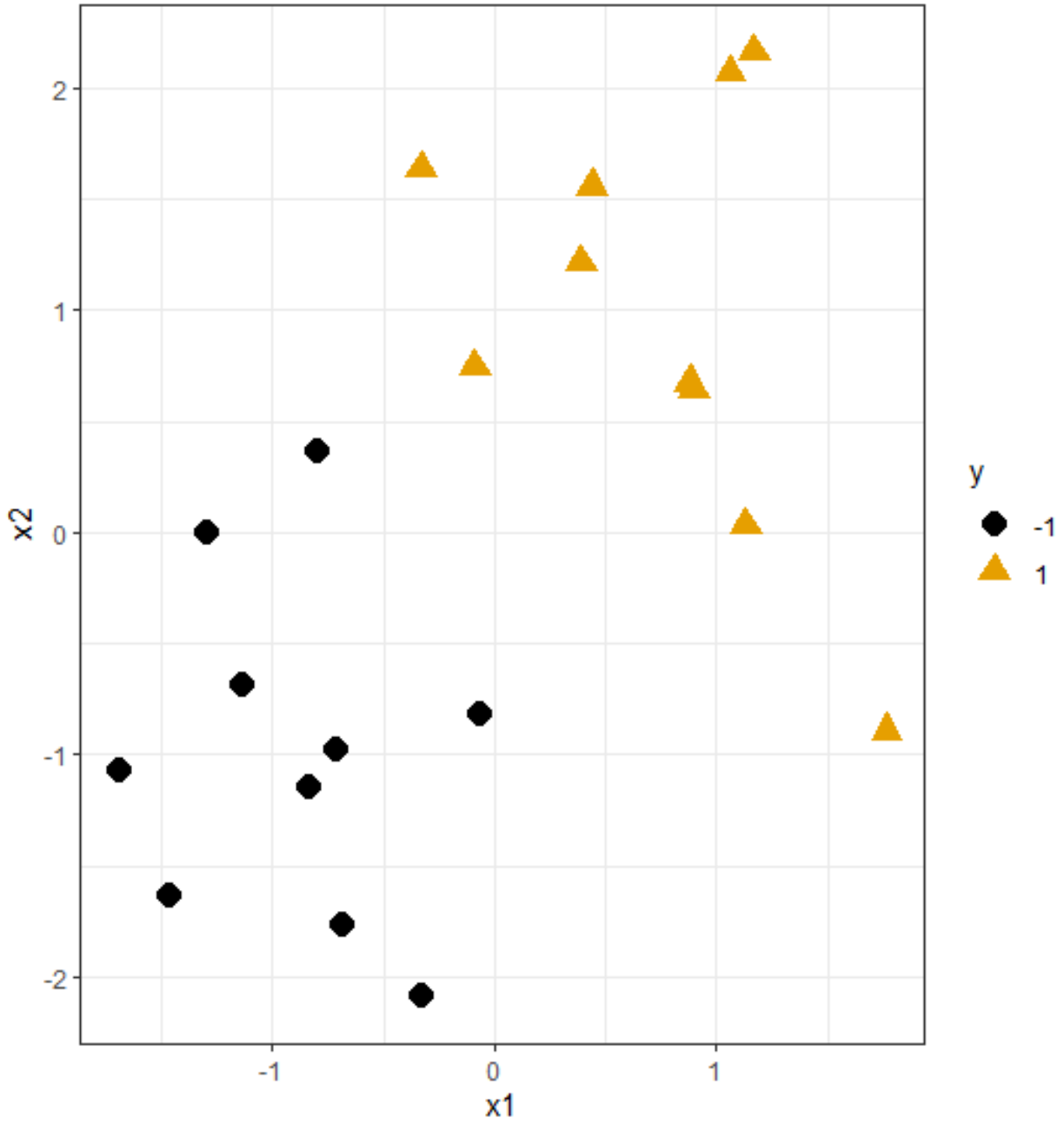
We have two inputs, x_1 and x_2. The response can take one of two classes (binary outcome).

Consider the case where the two classes, $y = -1$ and $y = 1$ are visibly separated.

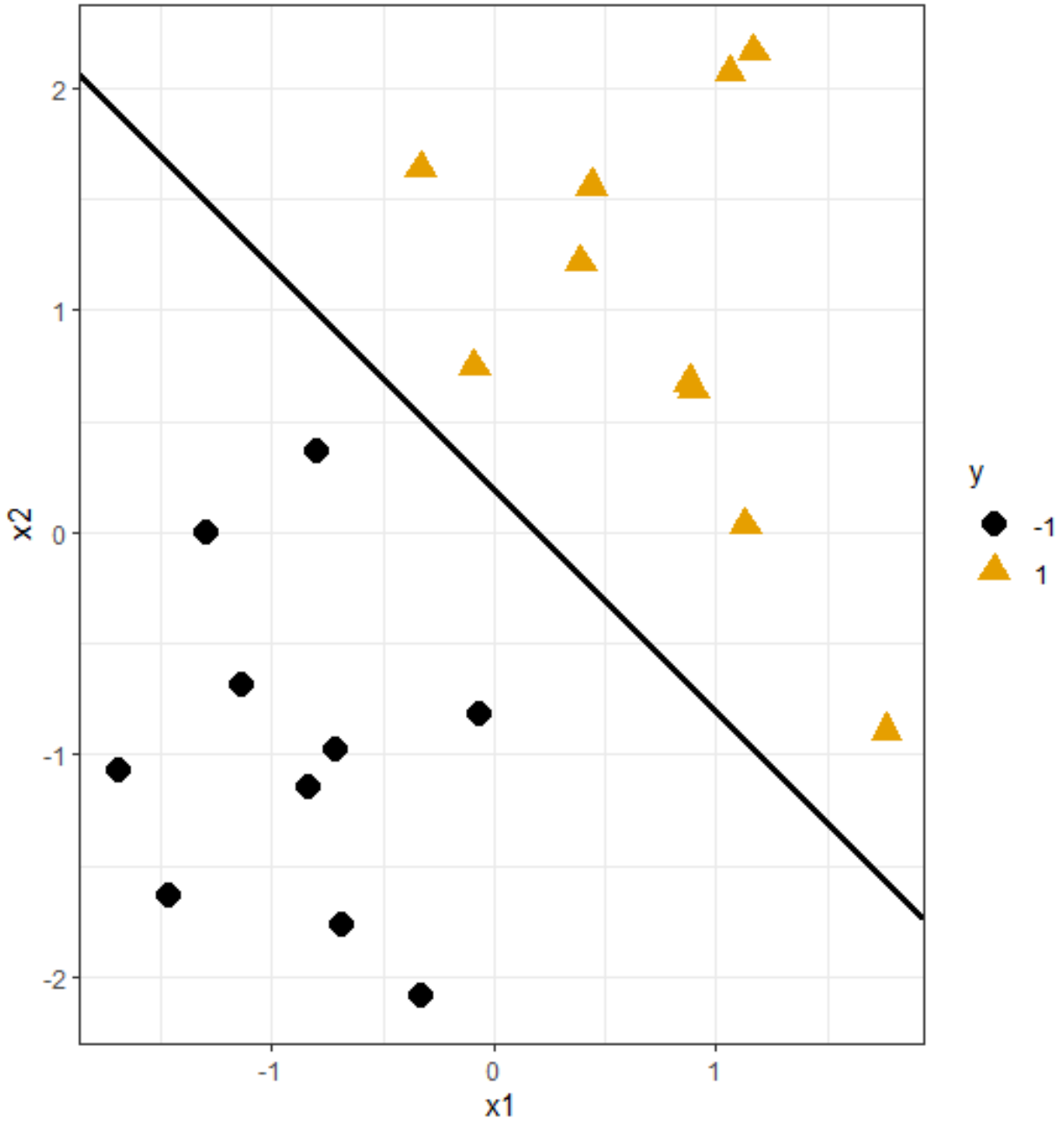


Can you draw a line that **separates** the two classes, $y = -1$ and $y = 1$, shown in the figure to the right?

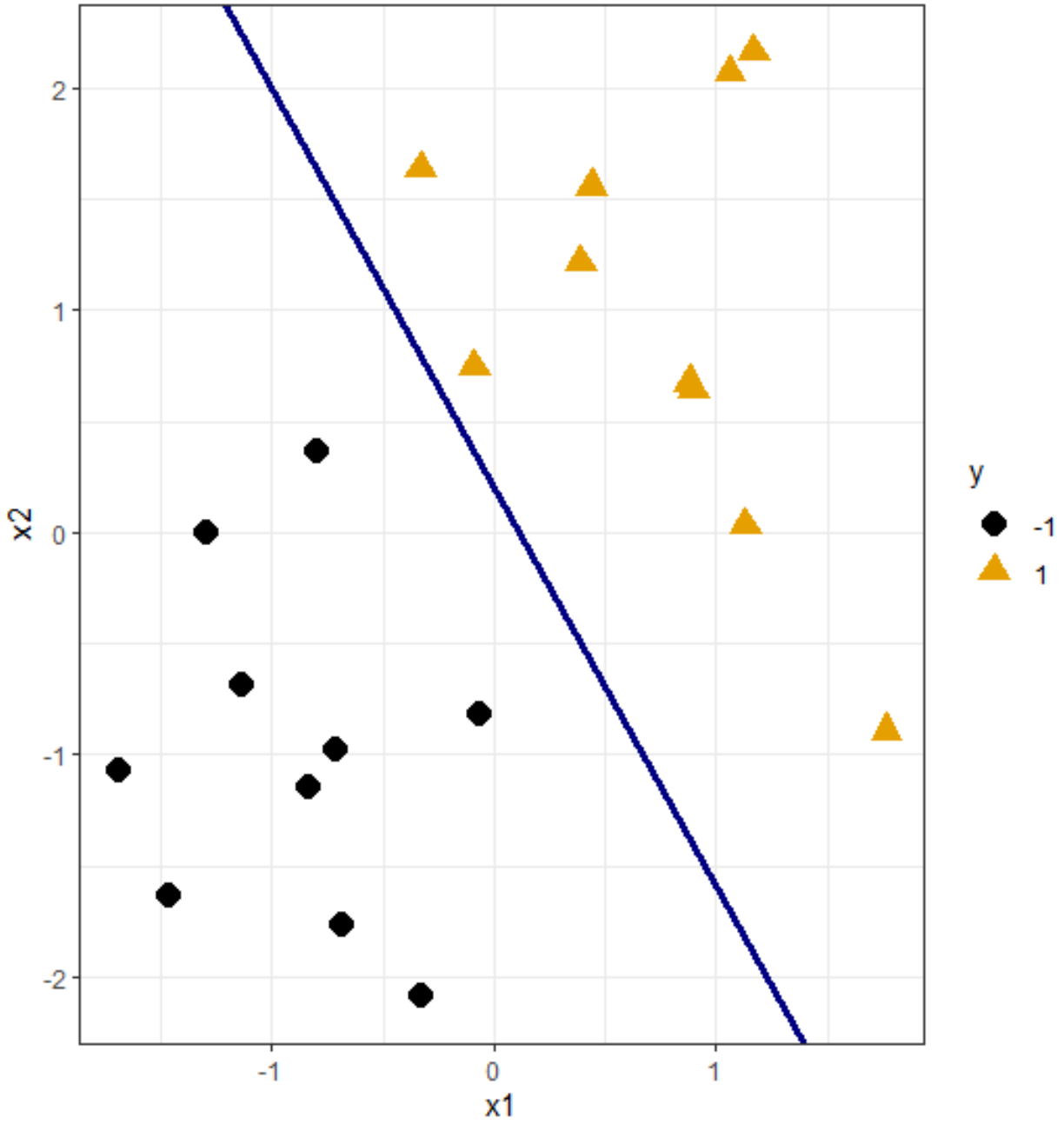
Where would you place the **decision boundary**?



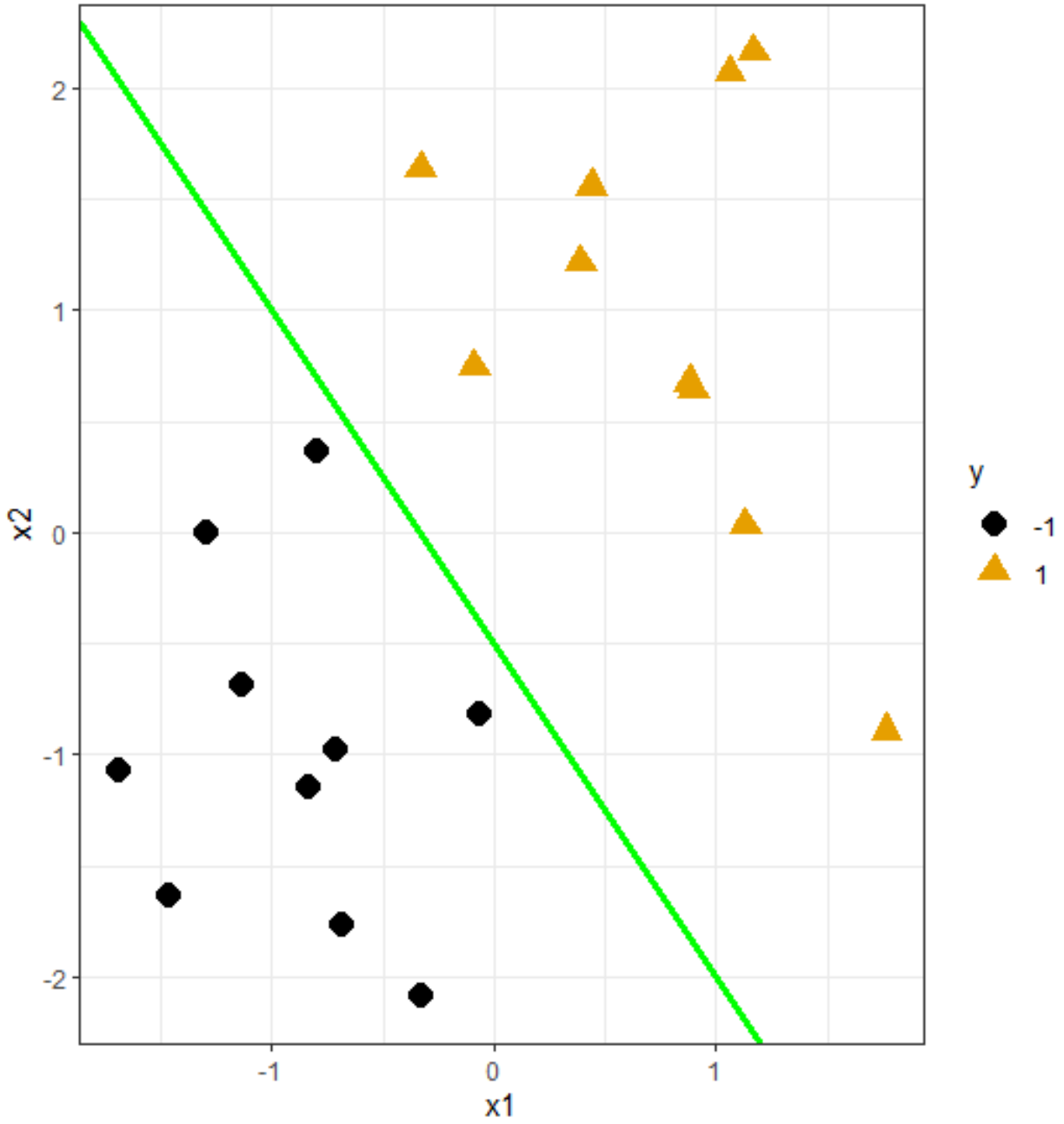
Is the black line a valid decision boundary?



What about the blue line?

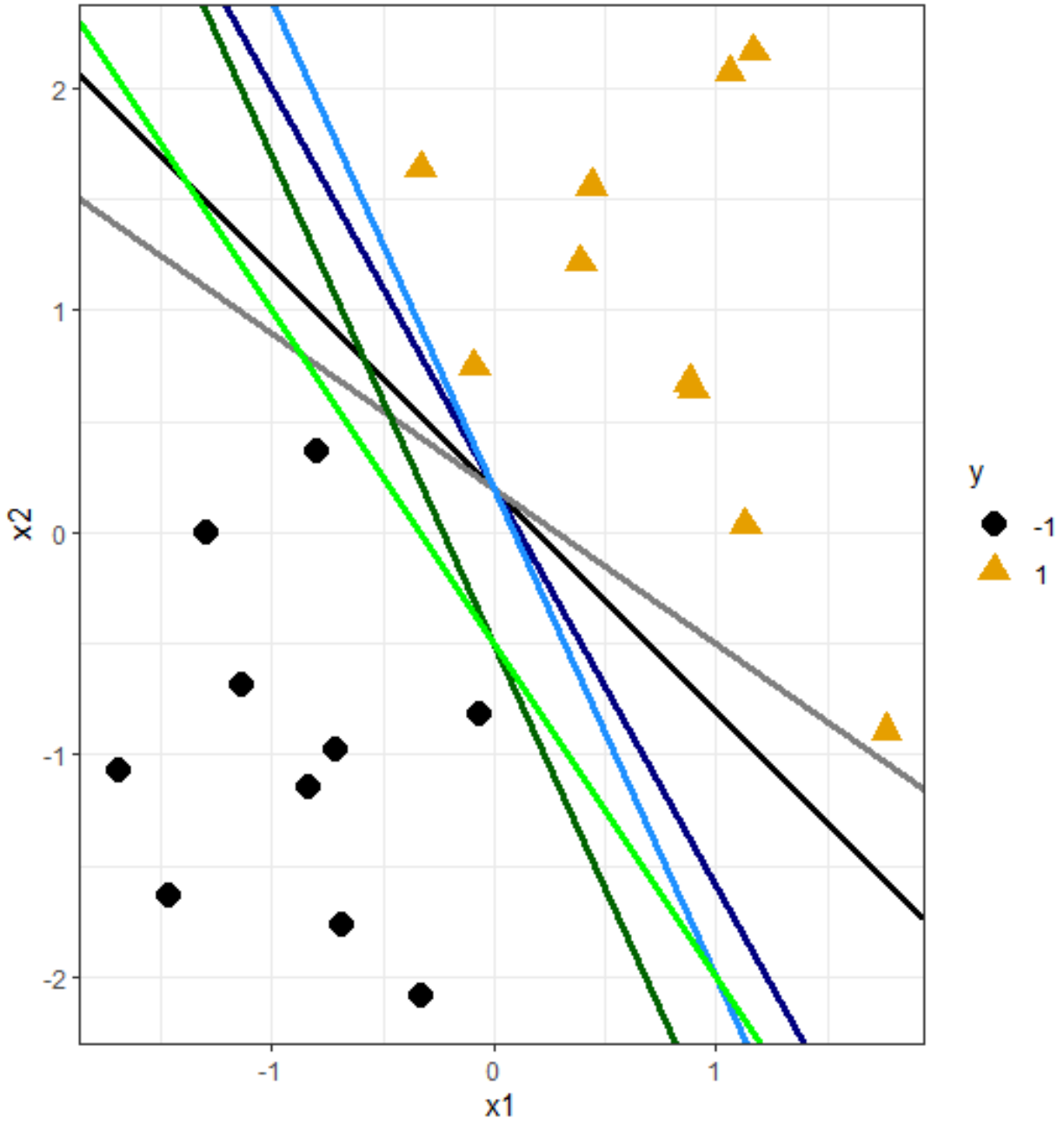


Or the green line?

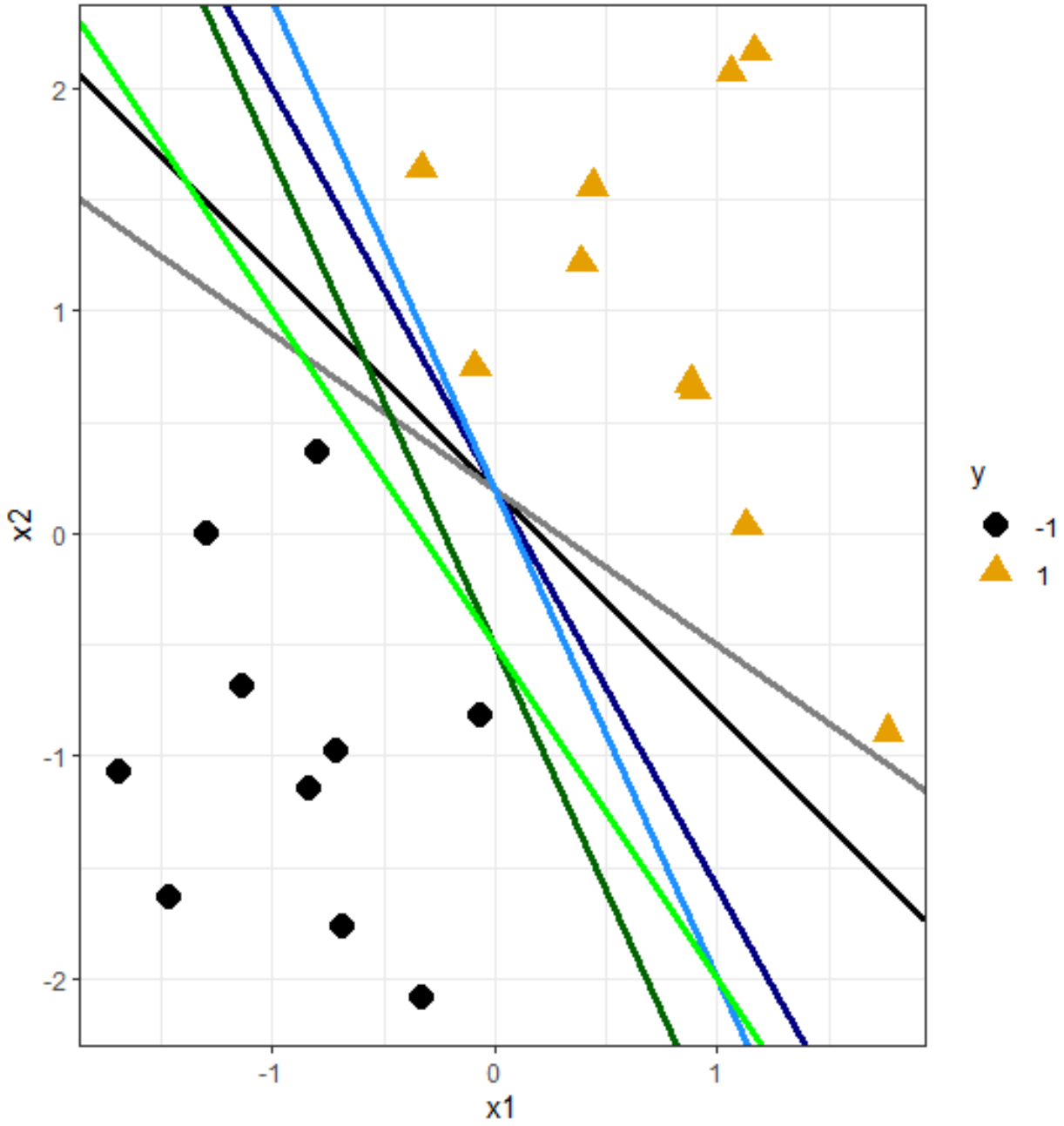


All lines displayed to the right are valid decision boundaries!

When the classes are **linearly** separable there are **infinitely** many ways to separate the classes!

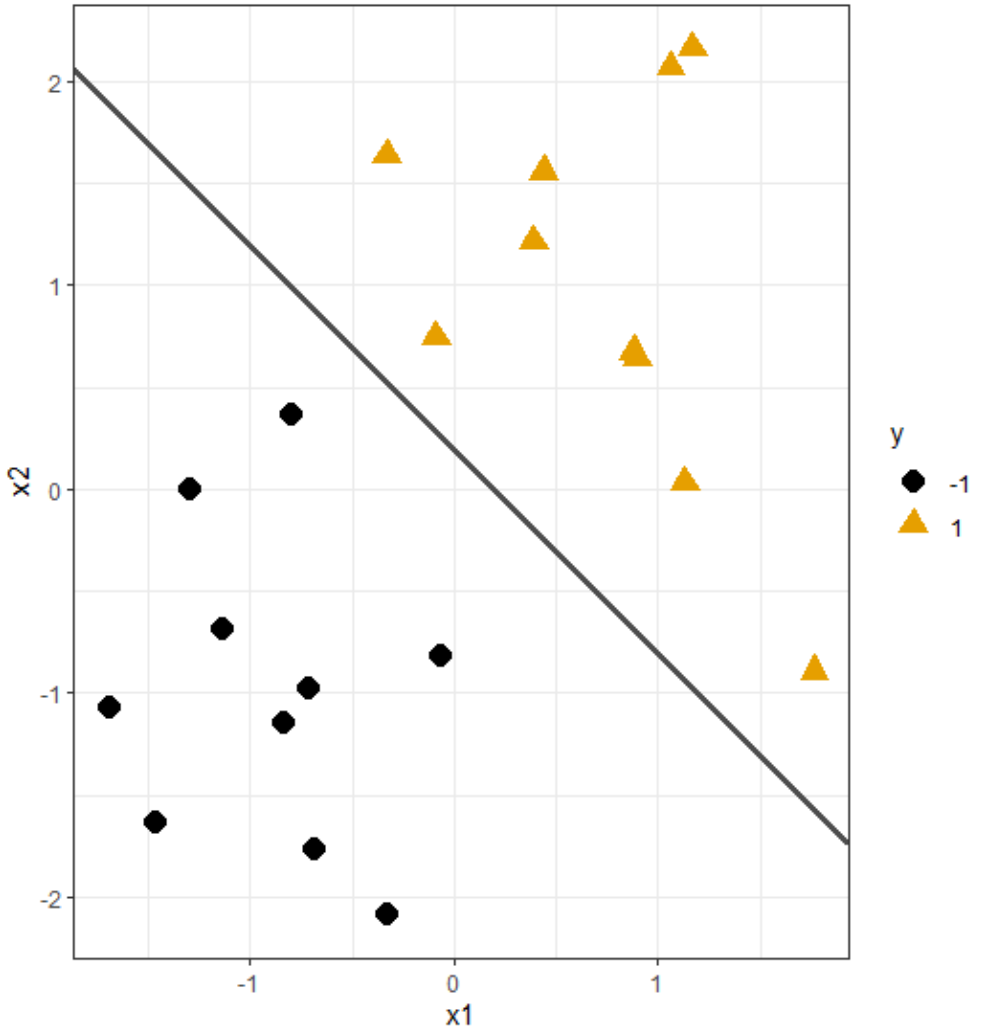


Is there a way to select the best or **optimal** separation line?



Before answering that, let's discuss some nomenclature. Our example is in two dimensions. In higher dimensions the decision boundary is referred to as a **hyperplane**.

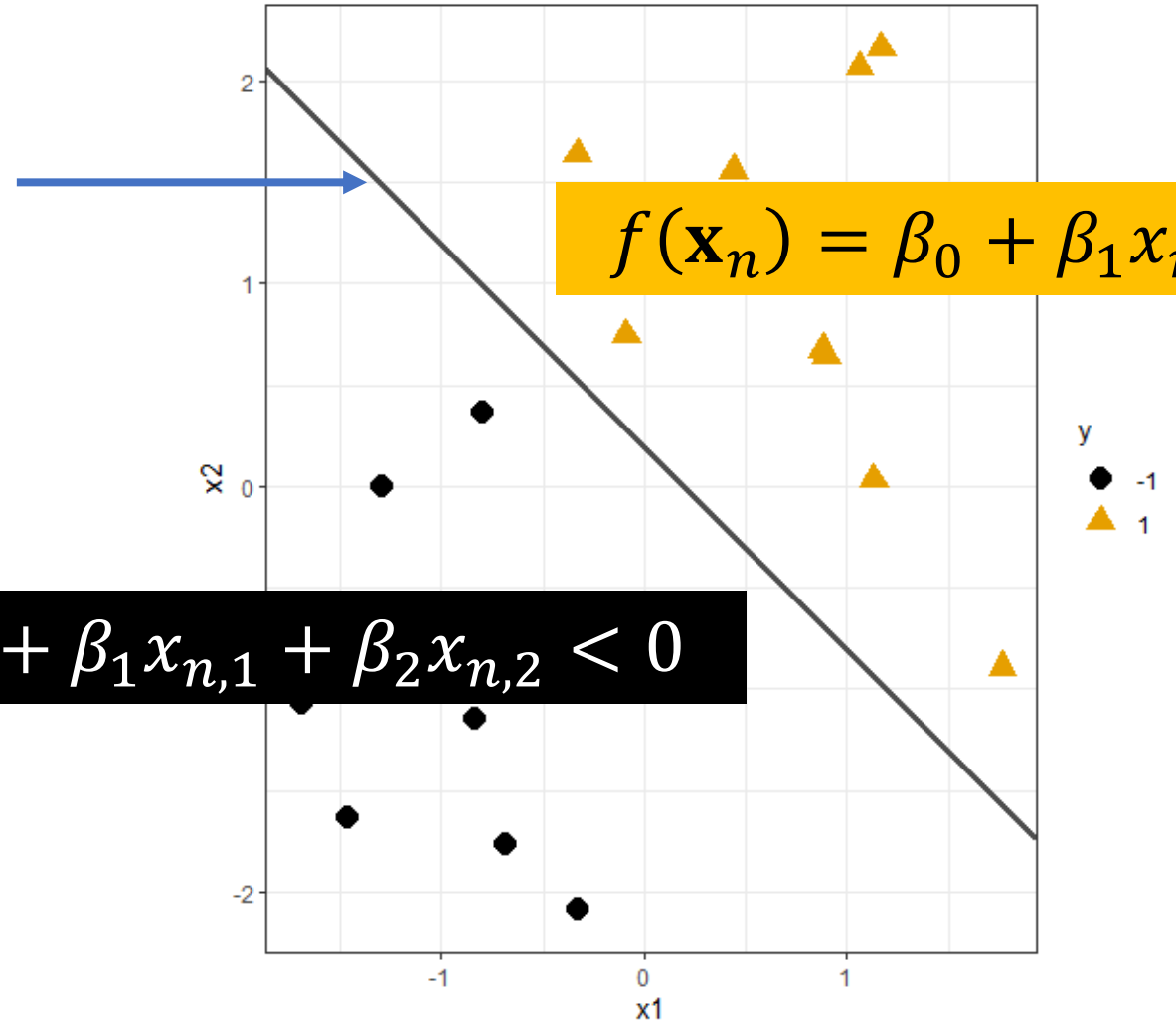
The general goal is therefore to find the optimal separating **hyperplane**.



Separating Hyperplane definition:

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

All points along the separating hyperplane have $f(\mathbf{x}) = 0$



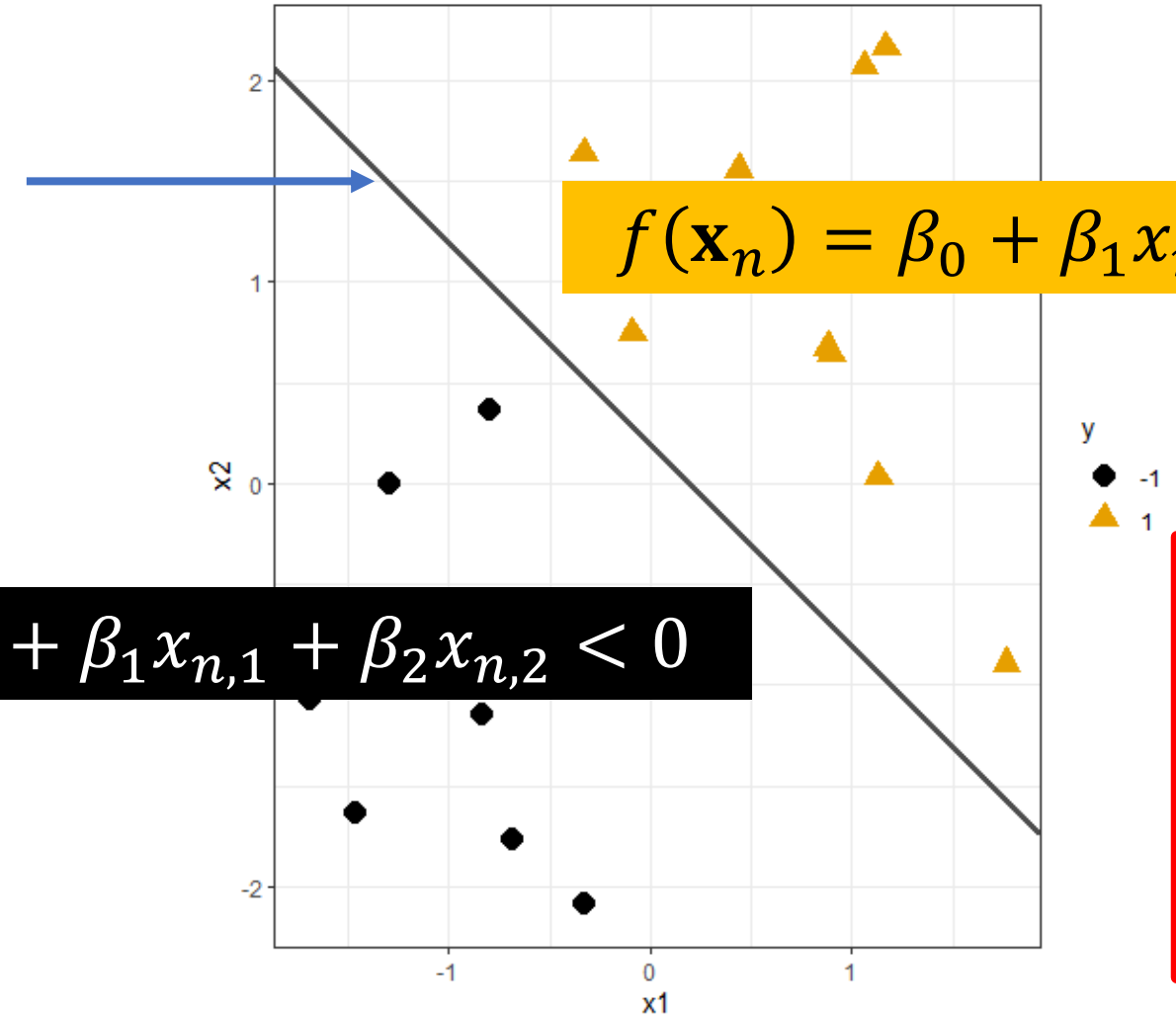
$$f(\mathbf{x}_n) = \beta_0 + \beta_1 x_{n,1} + \beta_2 x_{n,2} > 0$$

$$f(\mathbf{x}_n) = \beta_0 + \beta_1 x_{n,1} + \beta_2 x_{n,2} < 0$$

Separating Hyperplane definition:

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

All points along the separating hyperplane have $f(\mathbf{x}) = 0$



$$f(\mathbf{x}_n) = \beta_0 + \beta_1 x_{n,1} + \beta_2 x_{n,2} > 0$$

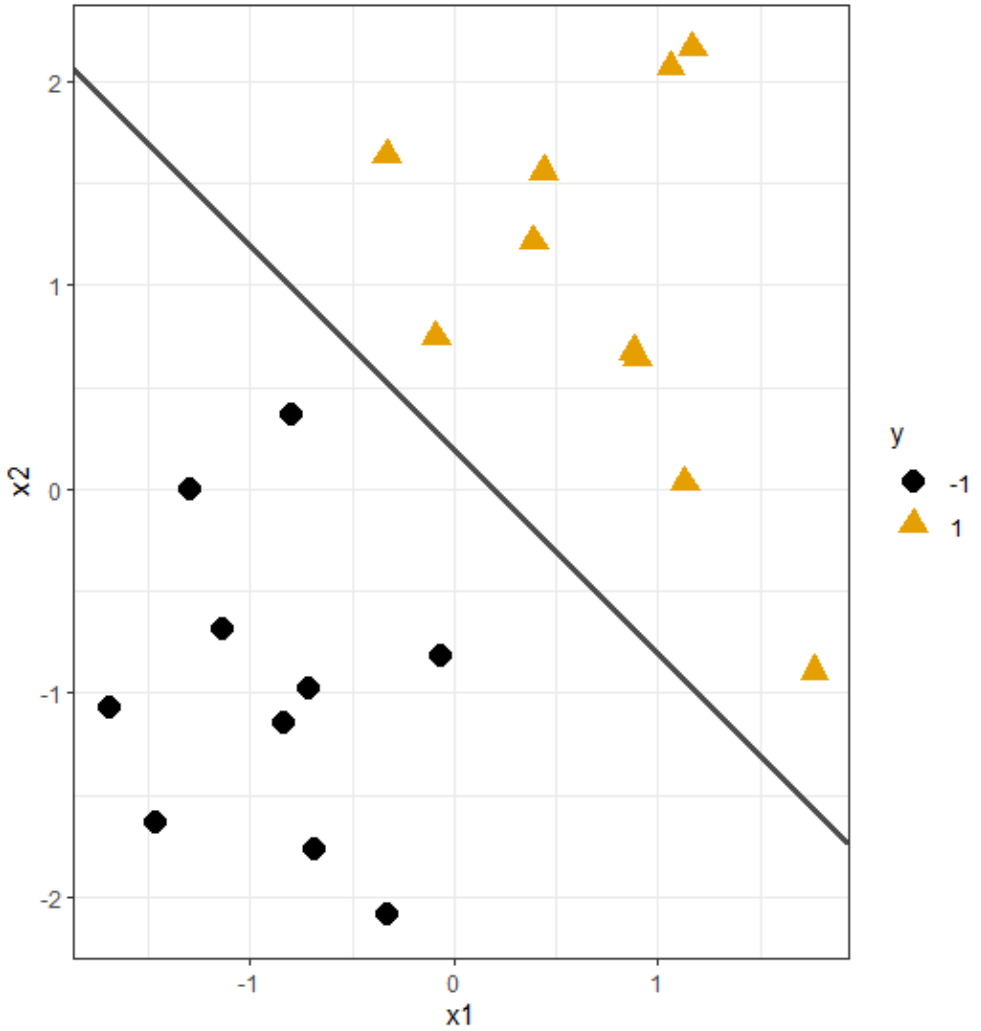
$$f(\mathbf{x}_n) = \beta_0 + \beta_1 x_{n,1} + \beta_2 x_{n,2} < 0$$

Classification rule:

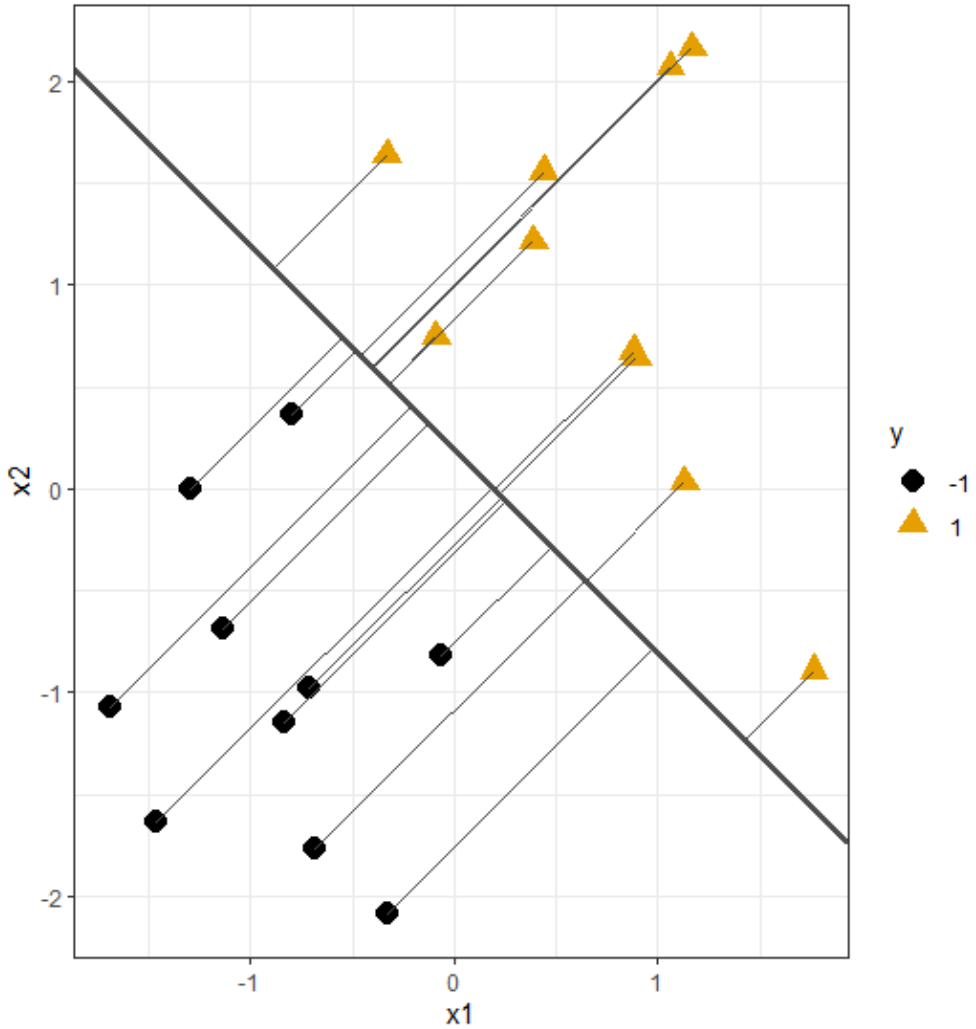
$$\text{sign}[f(\mathbf{x})]$$

Now back to the task at hand, how do we find the optimal separating hyperplane?

Consider the candidate hyperplane (the line) in the figure to the right.

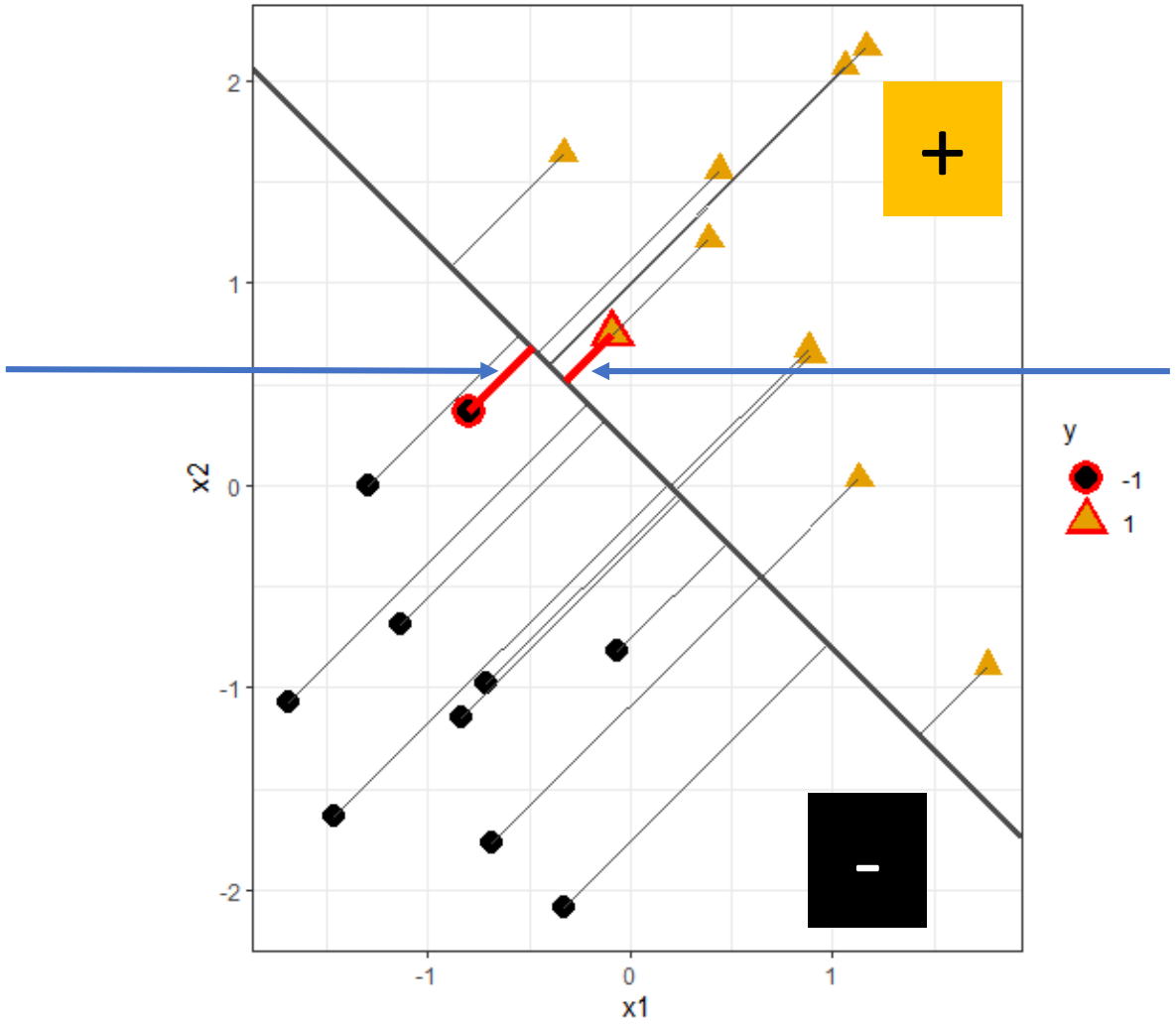


Calculate the distance between each (training) point and the candidate separating hyperplane.



Find the minimum distance on both sides of the candidate separating hyperplane

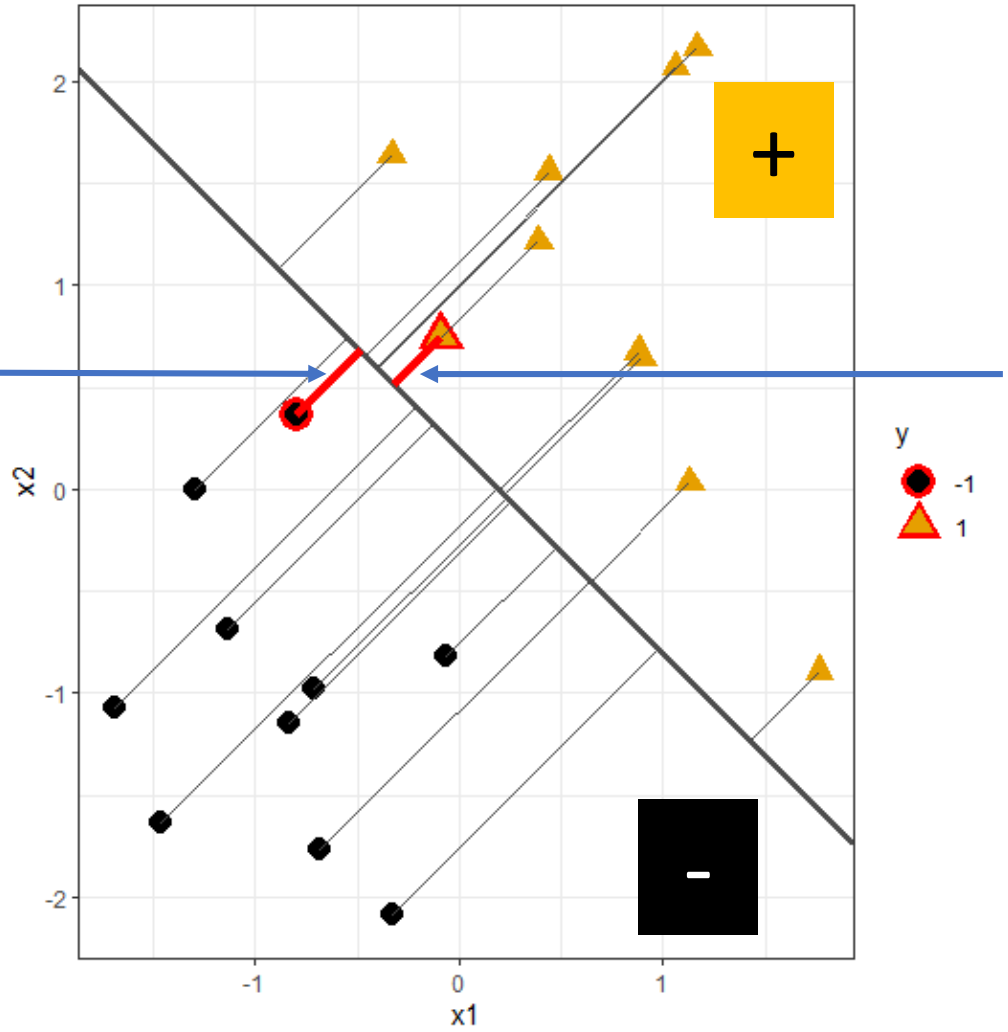
Minimum distance on the negative side: d_-



Minimum distance on the positive side: d_+

$d_- + d_+ = \text{MARGIN}$

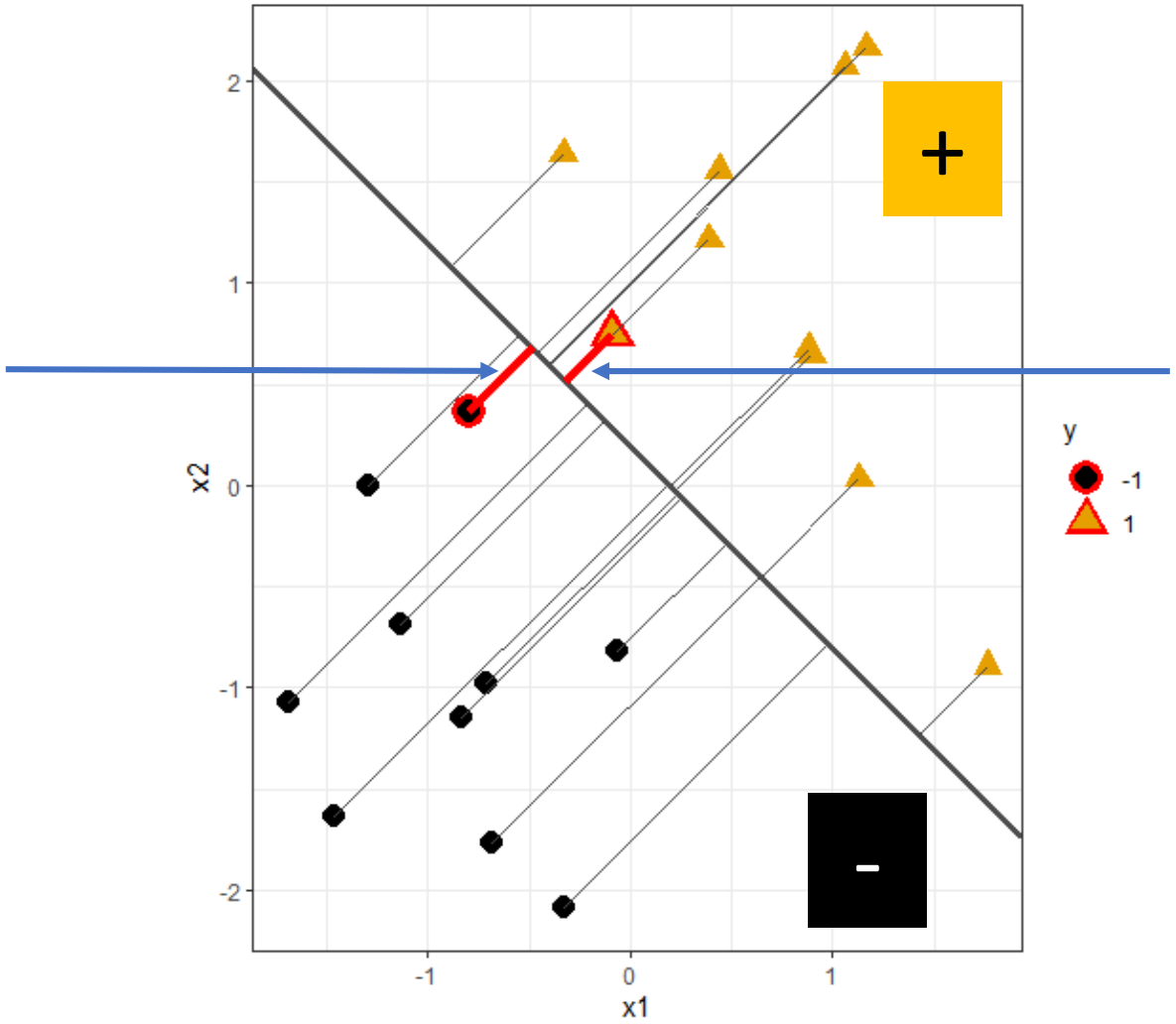
Minimum distance on the negative side: d_-



Minimum distance on the positive side: d_+

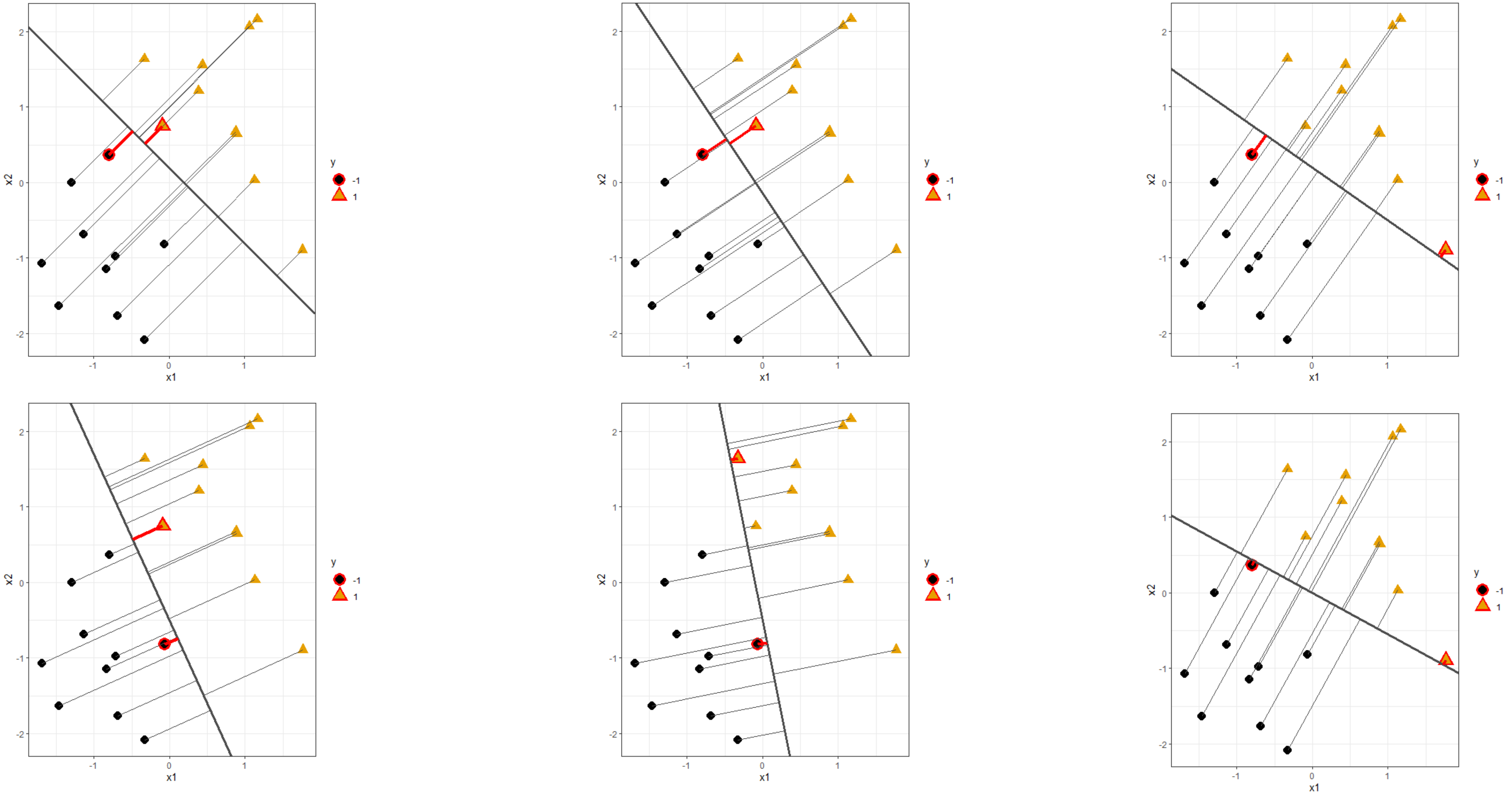
The optimal separating hyperplane is that which **MAXIMIZES** the **MARGIN**.

Minimum distance on the negative side: d_-

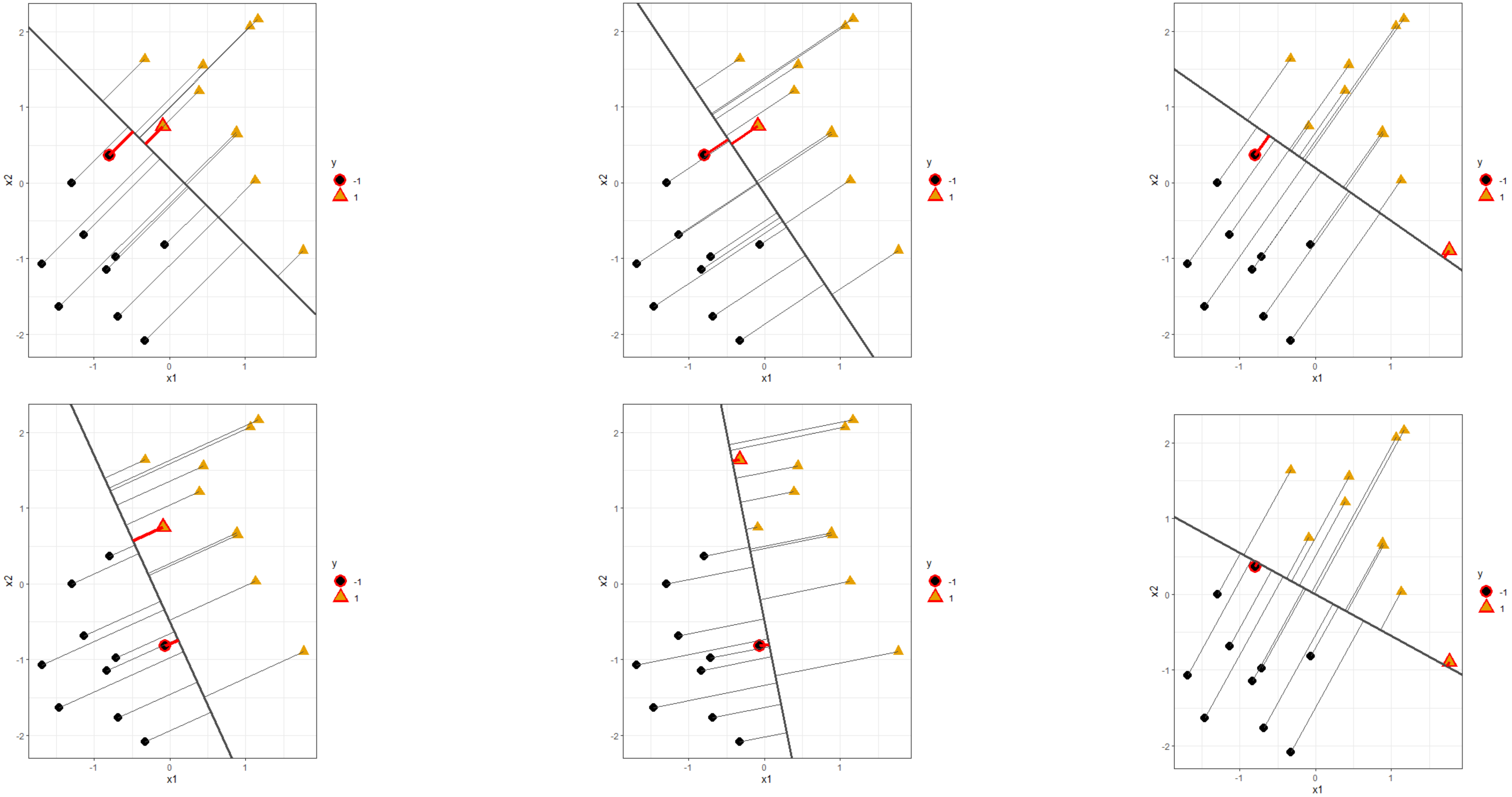


Minimum distance on the positive side: d_+

The margin might be associated with different points based on the candidate separating hyperplane.



Points associated with or that *support* the margin are known as **support points** or **support vectors**.



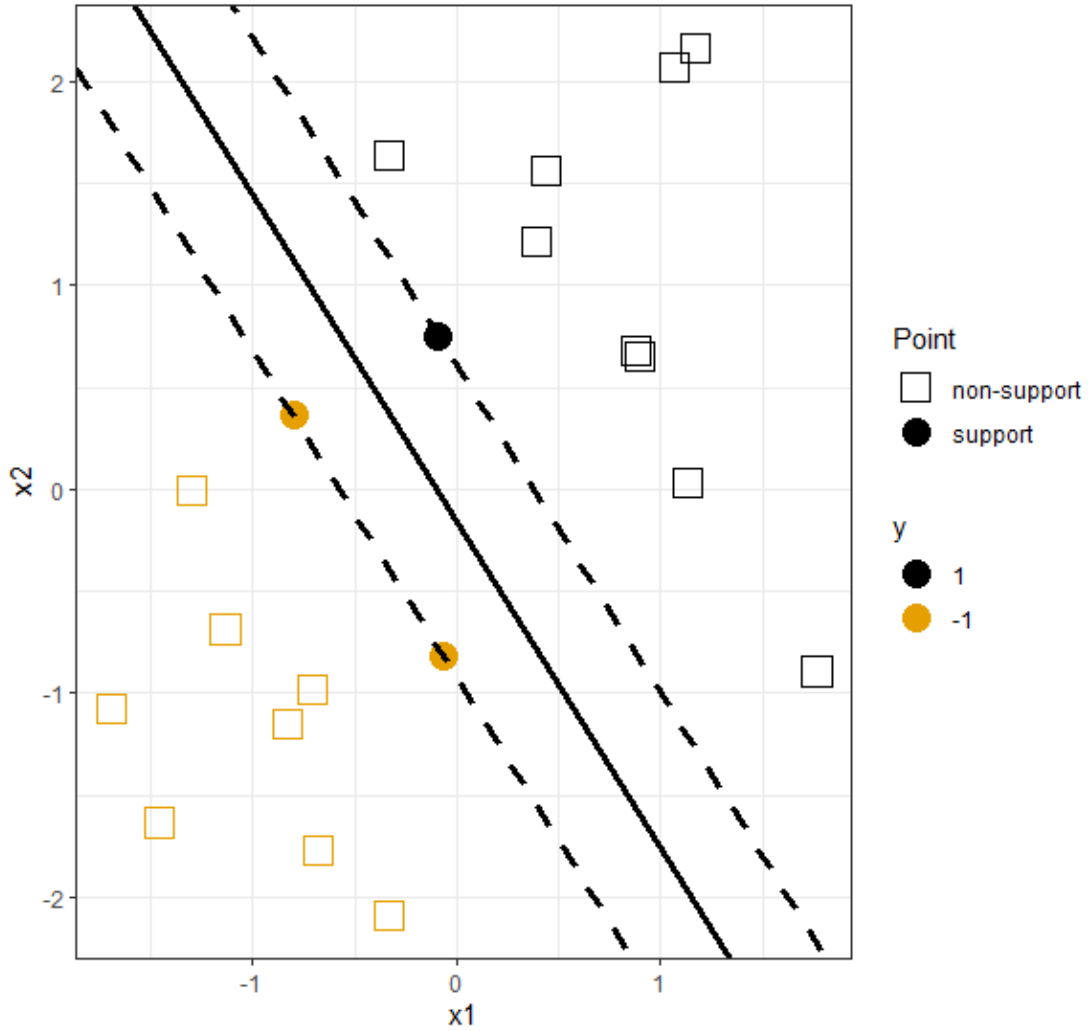
An optimization algorithm “tries out” many candidate separating hyperplanes

- The optimal separating hyperplane leads to the **maximum margin classifier**.

- Before going through how the optimization works, let’s first look at the optimal separating hyperplane for our simple example.

The optimal separating hyperplane shown in black with the margin boundary given by the dashed lines.

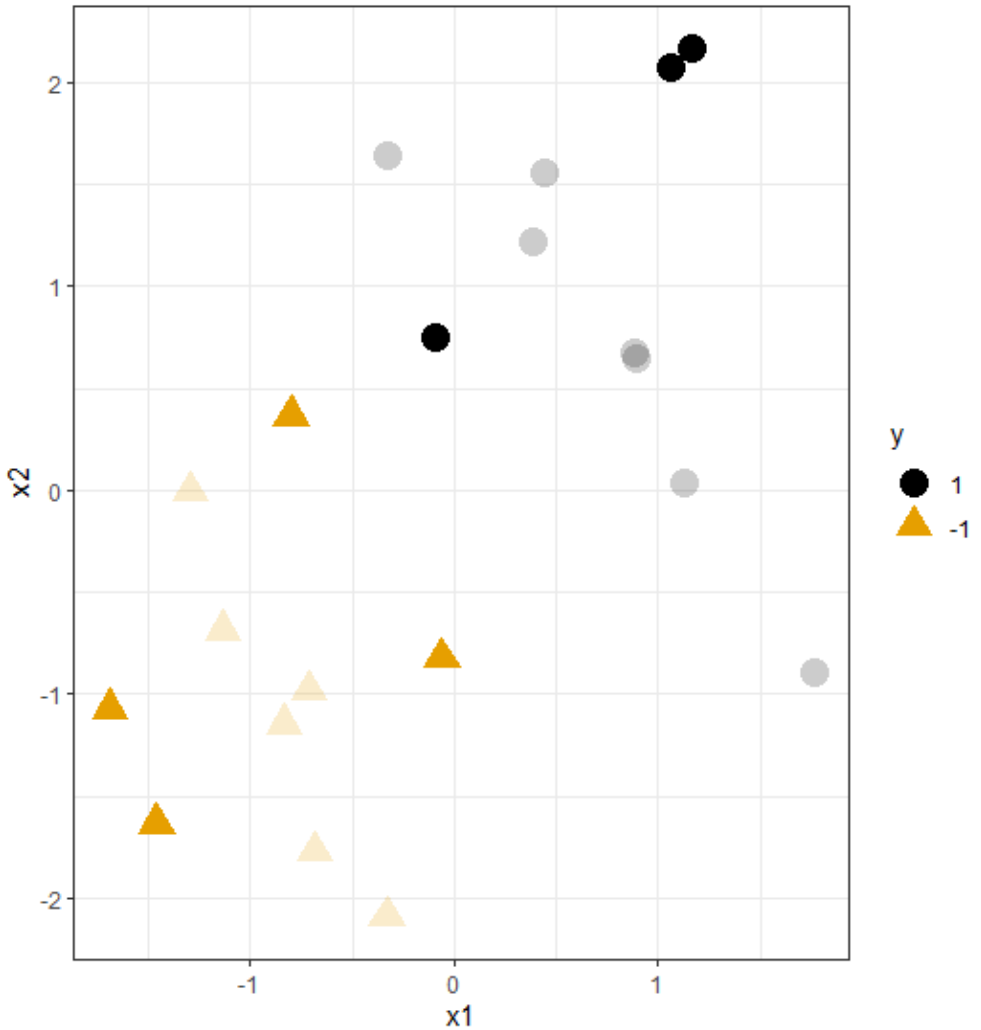
The support points are located on the boundary of the margin.



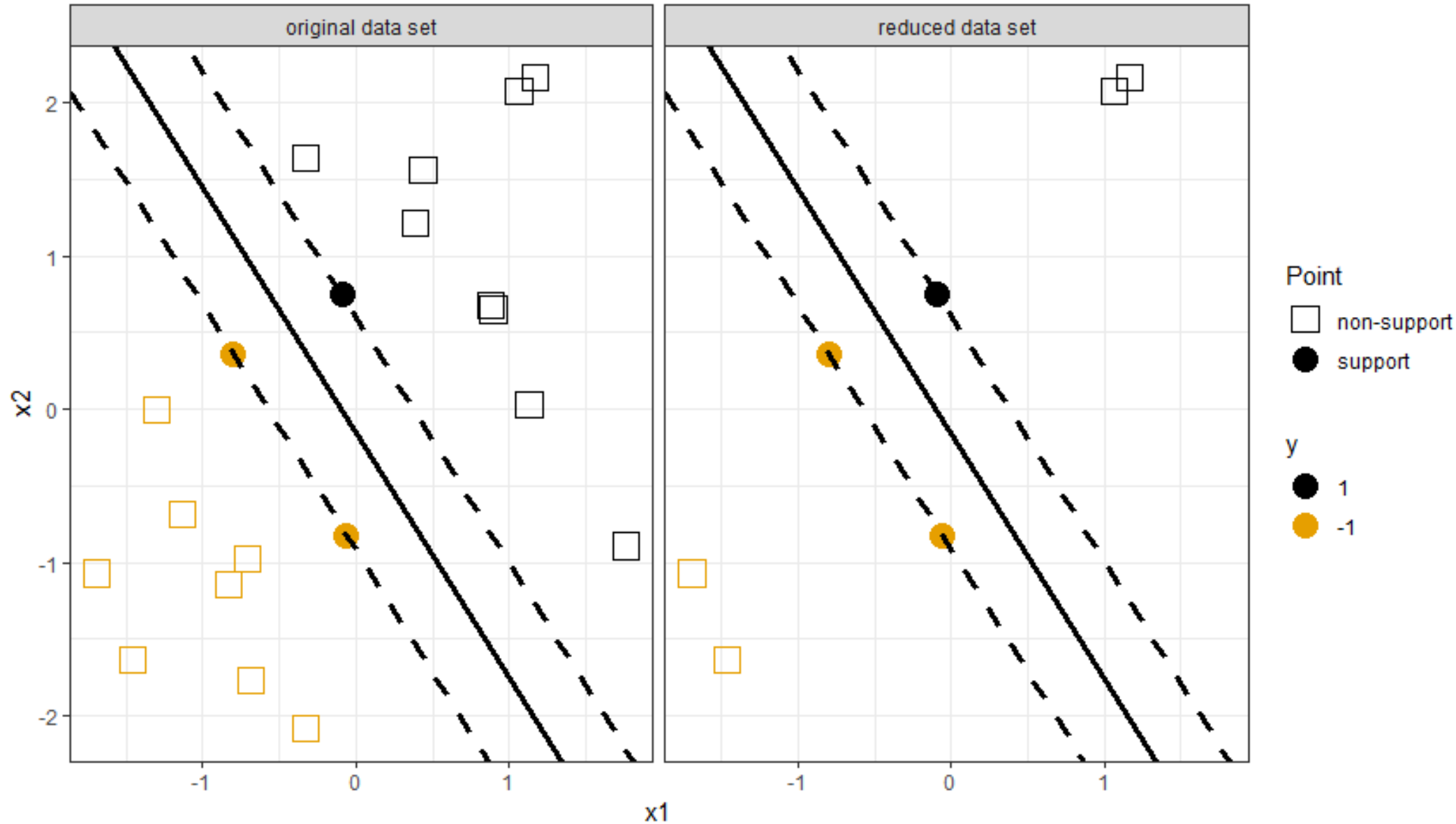
No observations are located within the margin.

What do you think will happen if we remove points? How will the solution change?

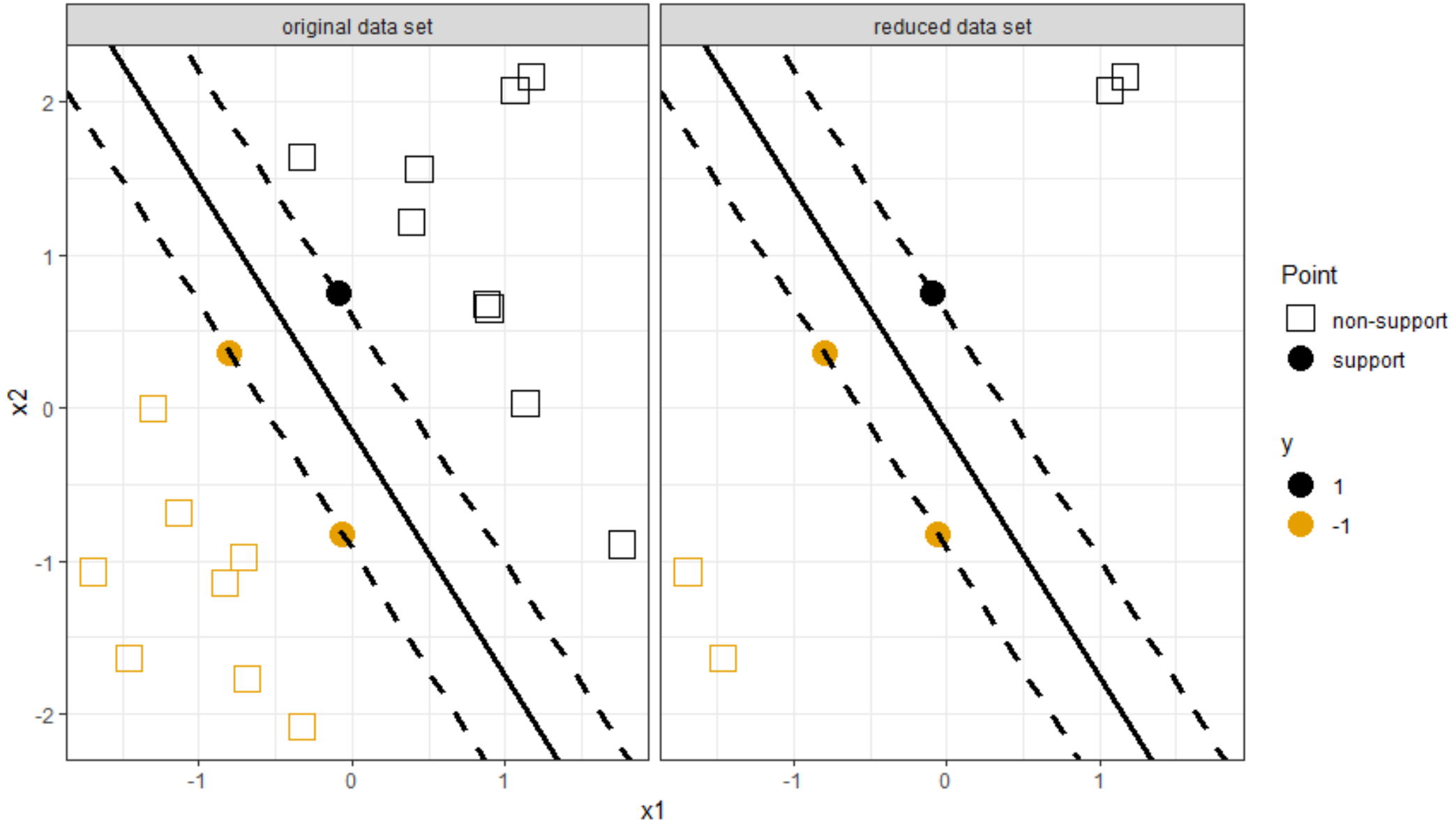
Transparent points have been removed.



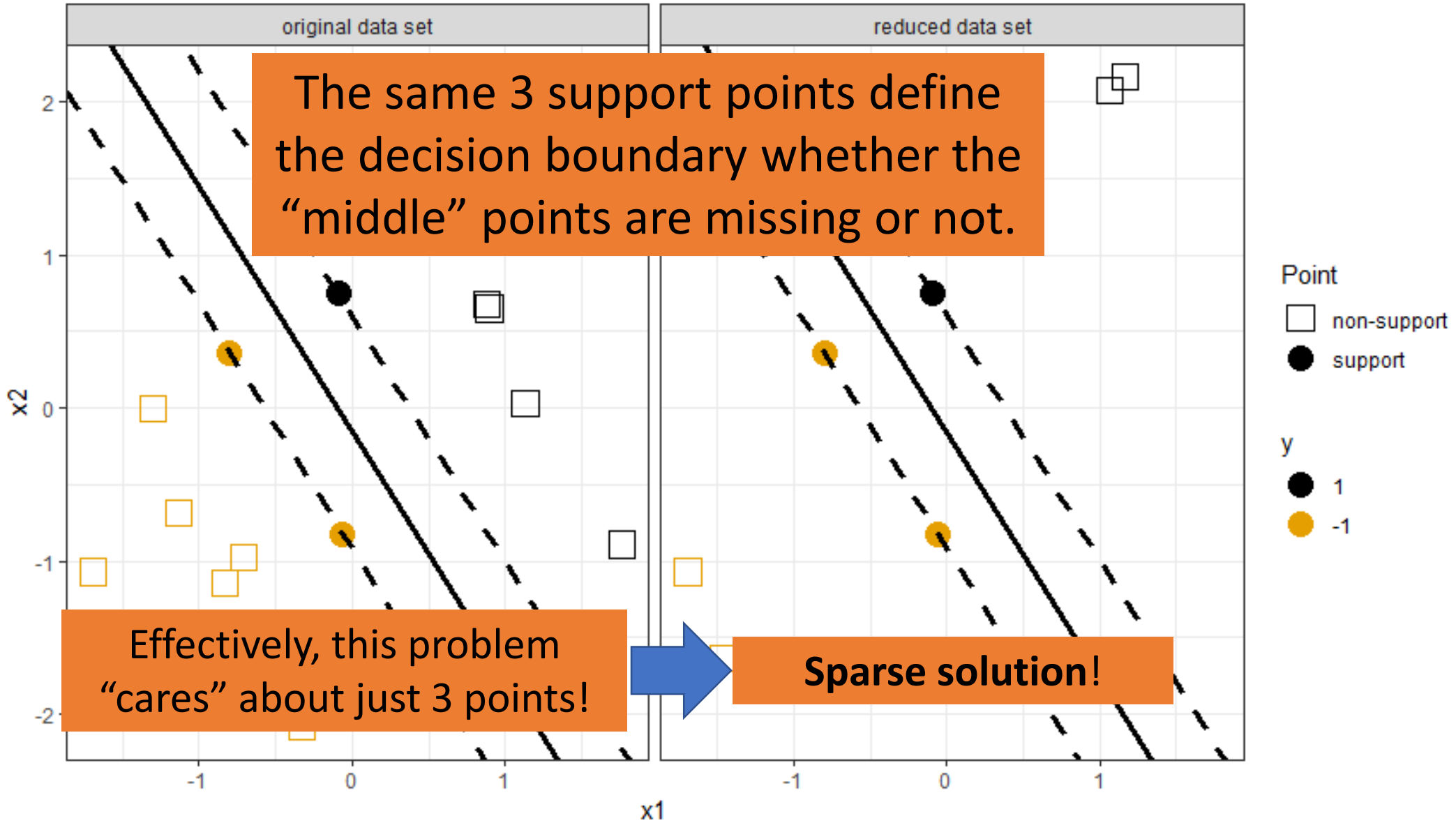
Will the decision boundary “move”?


original data set
reduced data set
x2
x1
2
1
0
-1
-2
-1
0
1
Point
non-support
support
y
1
-1

The optimal separating hyperplane ONLY depends on the **support points**!



The optimal separating hyperplane ONLY depends on the **support points**!



Let's consider a more general situation…

- We have N observations of D inputs, $\mathbf{x} = \{x_1, \ldots, x_D\}$, and the binary response, $y = \{-1, +1\}$.

- We typically structure the inputs into a $N \times (D + 1)$ design matrix $\mathbf{X}$ where the n-th row is the $1 \times (D + 1)$ row vector, $\mathbf{x}_{n,:}$.

However to be consistent with Support Vector literature we will use a different notation

- We will denote the n-th observation of the inputs as the $D \times 1$ column vector $\mathbf{x}_n$.

- The input vector associated with the first observation is then $\mathbf{x}_1$.

Observations associated with the two classes are defined as

$$\beta_0 + \sum_{d=1}^{D} \{x_{n,d}\beta_d\} > 0 \text{ if } y_n = 1$$

$$\beta_0 + \sum_{d=1}^{D} \{x_{n,d}\beta_d\} < 0 \text{ if } y_n = -1$$

With $\boldsymbol{\beta}$ defined as a $D \times 1$ column vector can rewrite in vector/matrix notation:

$$\beta_0 + \mathbf{x}_n^T \boldsymbol{\beta} > 0 \text{ if } y_n = 1$$

$$\beta_0 + \mathbf{x}_n^T \boldsymbol{\beta} < 0 \text{ if } y_n = -1$$

- Both definitions can be equivalently written as:

$$y_n(\beta_0 + \mathbf{x}_n^T \boldsymbol{\beta}) > 0$$

To find the optimal separating hyperplane we need to solve the following **constrained** optimization problem:

$$\arg\max_{\boldsymbol{\beta},\beta_0} M$$

$$\text{subject to } \|\boldsymbol{\beta}\|_2 = 1$$

$$y_n(\beta_0 + x_n^T\boldsymbol{\beta}) \geq M, n = 1, \ldots, N$$

To find the optimal separating hyperplane we need to

The constraints ensure the points are assigned to the correct side of the hyperplane at least a distance M away.

M represents the **MARGIN**: $2M = d_- + d_+$

$$\text{subject to} \|\boldsymbol{\beta}\|_2 = 1$$

$$y_n(\beta_0 + x_n^T\boldsymbol{\beta}) \geq M, n = 1, \ldots, N$$

Redefine the constraints the following way:

$$y_n(\beta_0 + x_n^T \boldsymbol{\beta}) \geq M \|\beta\|_2$$

If we set the MARGIN equal to:

$$M = 1/\|\boldsymbol{\beta}\|_2$$

Maximizing the MARGIN is equivalent to **minimizing**:

$$\|\boldsymbol{\beta}\|_2 \text{ or } \frac{1}{2}\|\boldsymbol{\beta}\|_2^2$$

The optimization problem is therefore

$$\arg\min_{\boldsymbol{\beta},\beta_0} \frac{1}{2}\|\boldsymbol{\beta}\|_2^2$$

$$\text{subject to } y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) \geq 1, n = 1, \ldots, N$$

The optimization problem is therefore

$$\arg\min_{\boldsymbol{\beta},\beta_0} \frac{1}{2}\|\boldsymbol{\beta}\|_2^2$$

$$\text{subject to } y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) \geq 1, n = 1, \ldots, N$$

CONVEX optimization problem.

Quadratic optimization criterion with linear inequality constraints.

Lagrange multipliers are used to find the optimum while satisfying the constraints

- The Lagrange (primal) function with Lagrange multipliers, α_n, is:

$$L_P(\boldsymbol{\beta}, \beta_0, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{\beta}\|_2^2 - \sum_{n=1}^{N}\{\alpha_n(y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) - 1)\}$$

Calculate the derivatives of L_P with respect to $\boldsymbol{\beta}$ and β_0, and set equal to zero:

$$\boldsymbol{\beta} = \sum_{n=1}^{N} \{\alpha_n y_n \mathbf{x}_n\}$$

$$0 = \sum_{n=1}^{N} \{\alpha_n y_n\}$$

Calculate the derivatives of L_P with respect to $\boldsymbol{\beta}$ and β_0, and set equal to zero:

$$\boldsymbol{\beta} = \sum_{n=1}^{N} \{\alpha_n y_n \mathbf{x}_n\}$$

The original $\boldsymbol{\beta}$ parameters depend on the Lagrange multipliers!!!

$$0 = \sum_{n=1}^{N} \{\alpha_n y_n\}$$

Using these conditions we obtain the dual representation which depends ONLY on the Lagrange multipliers:

$$L_D(\boldsymbol{\alpha}) = \sum_{n=1}^{N}\{\alpha_n\} - \frac{1}{2}\sum_{n=1}^{N}\left\{\sum_{m=1}^{N}\{\alpha_n\alpha_m y_n y_m \mathbf{x}_n^T\mathbf{x}_m\}\right\}$$

Find $\boldsymbol{\alpha}$ which maximizes L_D subject to the Karush-Kuhn-Tucker conditions:

$$\alpha_n \geq 0, n = 1, \ldots, N$$

$$0 = \sum_{n=1}^{N}\{\alpha_n y_n\}$$

$$\alpha_n(y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) - 1) = 0, n = 1, \ldots, N$$

Solve via quadratic programming.

The number of unknown parameters equals the number of observations!

- However, examine the last constraint in more detail:

$$\alpha_n(y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) - 1) = 0, n = 1, \ldots, N$$

- The above expression equals zero if:

$$\alpha_n = 0 \text{ or } y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) = 1$$

If $y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) = 1$:

- The n-th point, $\mathbf{x}_n$, is on the boundary of the margin!

- $\mathbf{x}_n$ is therefore a support point!

If $y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) > 1$:

- The constraint is only satisfied if $\alpha_n = 0$.

- Since $y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) > 1$, the n-th point is outside of the margin, and thus $\mathbf{x}_n$ is **NOT** a support point!

- Since $\alpha_n = 0$ the n-th point does **NOT** contribute to the original $\boldsymbol{\beta}$ parameters!

The only points that impact the optimal separating hyperplane are those with $\alpha_n > 0$

- If the support points are contained within the set $\mathcal{S}$ the original parameters that define the hyperplane are:

$$\widehat{\boldsymbol{\beta}} = \sum_{n \in \mathcal{S}} \{\hat{\alpha}_n y_n \mathbf{x}_n\}$$

- Which gives the classification rule at a new point $\mathbf{x}_*$ to be:

$$\text{sign}\left[\hat{\beta}_0 + \sum_{n \in \mathcal{S}} \{\hat{\alpha}_n y_n \mathbf{x}_*^T \mathbf{x}_n\}\right]$$

The only points that impact the optimal separating hyperplane are those with $\alpha_n > 0$

- If the support points are contained within the set $\mathcal{S}$ the original parameters that define the hyperplane are:

$$\hat{\boldsymbol{\beta}} = \sum_{n \in \mathcal{S}} \{\hat{\alpha}_n y_n \mathbf{x}_n\}$$

- Which gives the classification rule to be:

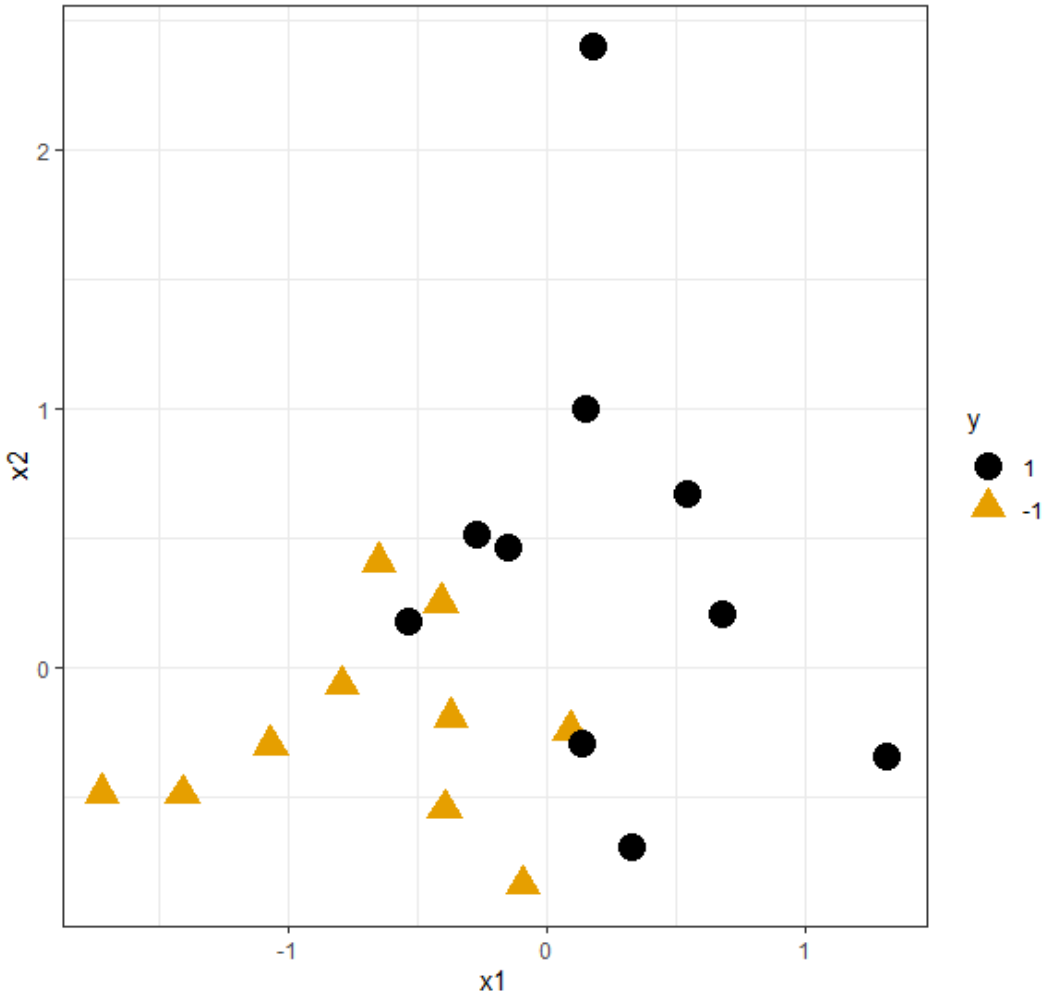
$$\text{sign}\left[\hat{\beta}_0 + \sum_{n \in \mathcal{S}} \{\hat{\alpha}_n y_n \mathbf{x}_*^T \mathbf{x}_n\}\right]$$

Do not need to calculate the original $\boldsymbol{\beta}$ parameters!

Now consider a non-separable or class overlap situation

We cannot easily draw a single decision boundary to separate the two classes.

How can we proceed?



We will still try to maximize the MARGIN…

- However, we cannot use a **hard** margin.
- We will allow some points to be on the wrong side of the margin, and thus the hyperplane.
- The margin is therefore a ***soft*** margin.

Introduce "slack" variables, ξ_n, into the optimization problem

$$\arg\max_{\boldsymbol{\beta},\beta_0} M$$

$$\text{subject to } \|\boldsymbol{\beta}\|_2 = 1$$

$$y_n(\beta_0 + x_n^T\boldsymbol{\beta}) \geq M(1-\xi_n), n = 1, \ldots, N$$

$$\xi_n \geq 0, \sum_{n=1}^{N} \xi_n \leq \text{constant}$$

Introduce "slack" variables, ξ_n, into the optimization problem

The slack variable, ξ_n, is proportional to the amount the n-th observation is on the wrong side of the margin.

The n-th observation ***violates*** the margin if $\xi_n > 0$.

Misclassifications occur when $\xi_n > 1$.

$$\xi_n \geq 0, \sum_{n=1}^{N} \xi_n \leq \text{constant}$$

Introduce "slack" variables, ξ_n, into the optimization problem

Bounding the summation over the slack variables bounds the total number of training misclassifications.

The constant is a tuning parameter we must learn through cross-validation.

$$\xi_n \geq 0, \sum_{n=1}^{N} \xi_n \leq \text{constant}$$

The soft margin optimization problem can be re-written as:

$$\arg\min_{\boldsymbol{\beta},\beta_0}\left\{\frac{1}{2}\|\boldsymbol{\beta}\|_2^2 + C\sum_{n=1}^{N}\xi_n\right\}$$

$$\text{subject to } \xi_n \geq 0,$$

$$y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) \geq 1 - \xi_n,$$

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

The soft margin optimization problem can be re-written as:

$$\arg\min_{\boldsymbol{\beta},\beta_0}\left\{\frac{1}{2}\|\boldsymbol{\beta}\|_2^2 + C\sum_{n=1}^{N}\xi_n\right\}$$

What does this remind you of?

$$\text{subject to } \xi_n \geq 0,$$

$$y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) \geq 1 - \xi_n,$$

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

Rewrite the slack variables

$$\xi_n = \max[0, 1 - y_n(\beta_0 + \mathbf{x}_n^T \boldsymbol{\beta})]$$

Substitute into the optimization problem

$$\arg\min_{\boldsymbol{\beta},\beta_0}\left\{\frac{1}{2}\|\boldsymbol{\beta}\|_2^2 + C\sum_{n=1}^{N}\max[0, 1 - y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta})]\right\}$$

Rearrange

$$\arg\min_{\boldsymbol{\beta},\beta_0}\left\{\frac{1}{2C}\|\boldsymbol{\beta}\|_2^2+\sum_{n=1}^{N}\max[0,1-y_n(\beta_0+\mathbf{x}_n^T\boldsymbol{\beta})]\right\}$$

$$\arg\min_{\boldsymbol{\beta},\beta_0}\left\{\lambda\|\boldsymbol{\beta}\|_2^2+\sum_{n=1}^{N}\max[0,1-y_n(\beta_0+\mathbf{x}_n^T\boldsymbol{\beta})]\right\}$$

The soft margin classifier optimization problem is therefore a penalized loss function in disguise!

$$\underbrace{\lambda\|\boldsymbol{\beta}\|_2^2}_{\text{Ridge-like regularization}} + \underbrace{\sum_{n=1}^{N} \max[0, 1 - y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta})]}_{\text{“Hinge” loss function}}$$

To find the soft margin classifier, we need to use Lagrange multipliers

- The Lagrange (primal) function includes two sets of Lagrange multipliers, α_n and μ_n:

$$L_P = \frac{1}{2}\|\boldsymbol{\beta}\|_2^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}\left\{\alpha_n\left(y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) - (1-\xi_n)\right)\right\} - \sum_{n=1}^{N}\mu_n\xi_n$$

Dual form is like the hard margin case:

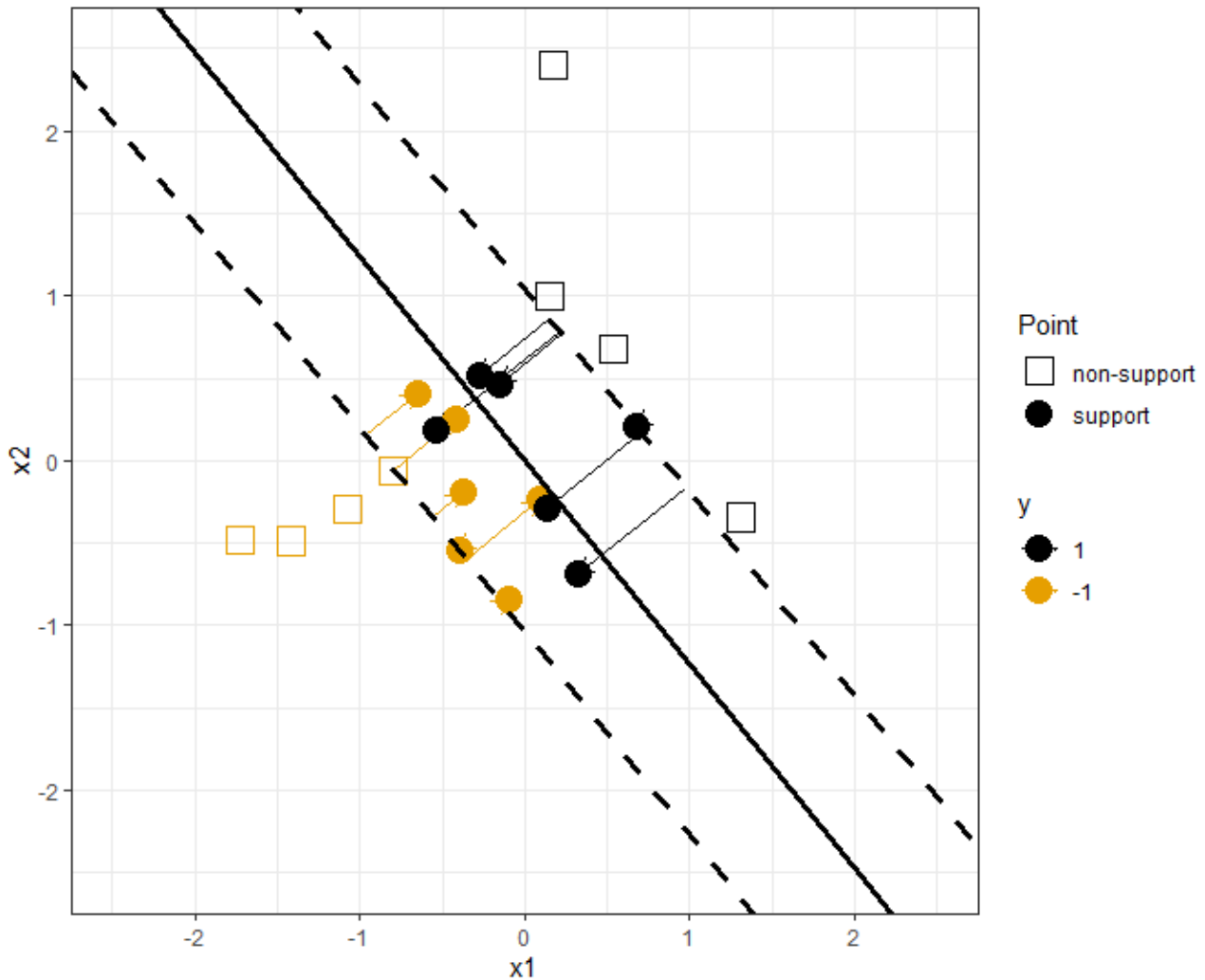
- Find $\boldsymbol{\alpha}$ which maximizes:

$$L_D(\boldsymbol{\alpha}) = \sum_{n=1}^{N}\{\alpha_n\} - \frac{1}{2}\sum_{n=1}^{N}\left\{\sum_{m=1}^{N}\{\alpha_n\alpha_m y_n y_m \mathbf{x}_n^T\mathbf{x}_m\}\right\}$$

- Subject to:

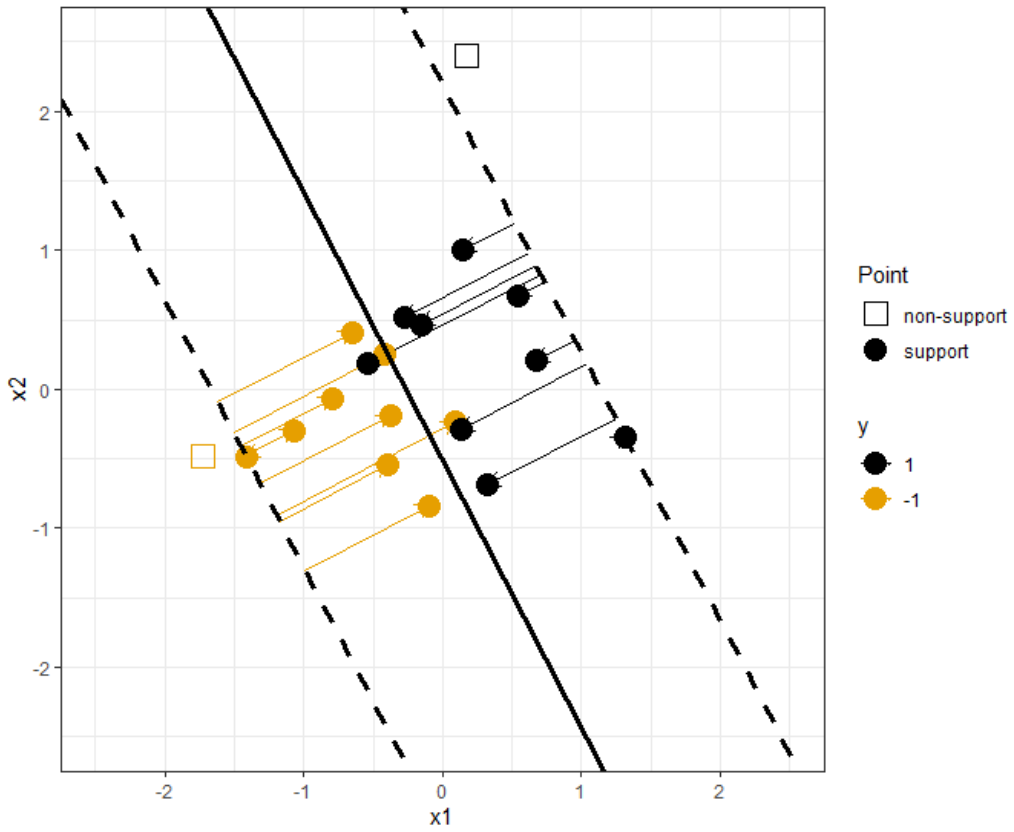
$$0 \le \alpha_n \le C, \textstyle\sum_{n=1}^{N}\alpha_n y_n = 0$$
$$\alpha_n\big(y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) - (1-\xi_n)\big) = 0$$
$$\mu_n\xi_n = 0$$
$$y_n(\beta_0 + \mathbf{x}_n^T\boldsymbol{\beta}) - (1-\xi_n) \ge 0$$

Soft margin classifier at $C = 1$

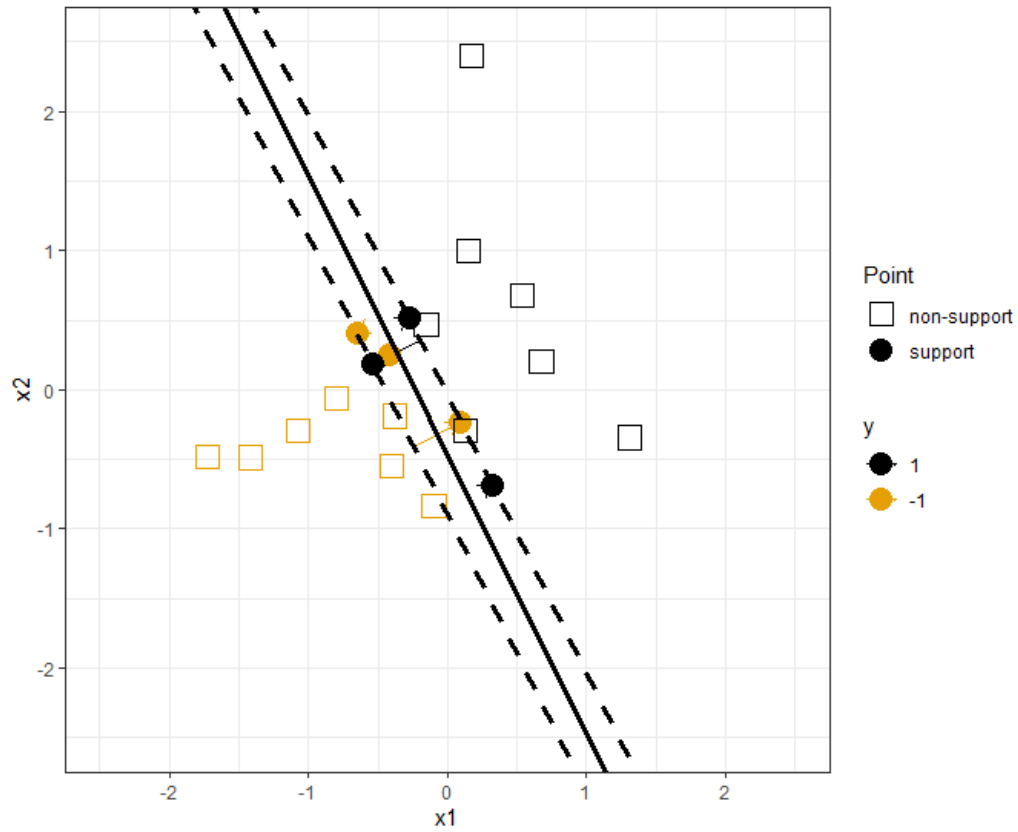


Soft margin classifier at two values of C

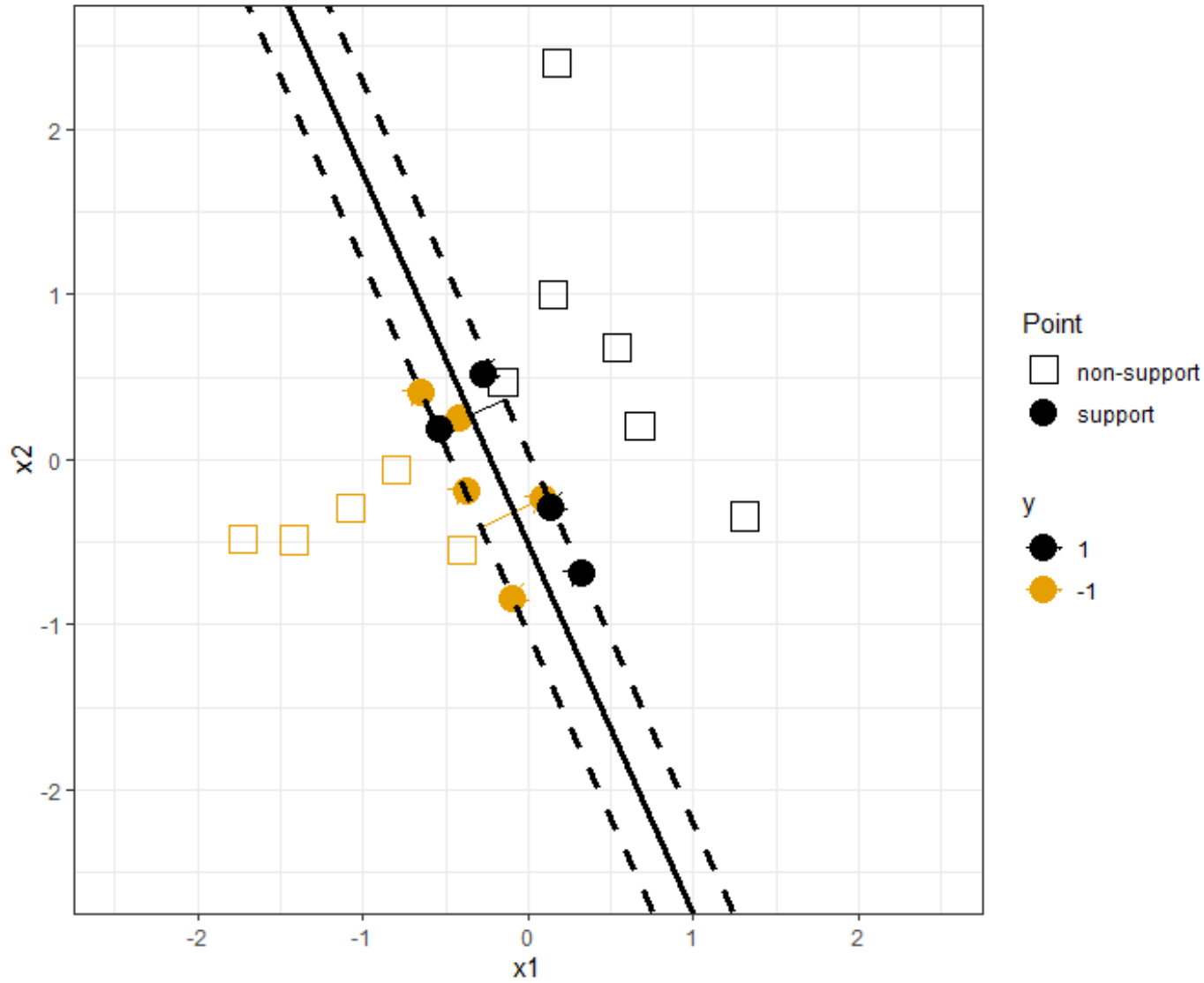
$C = 0.1$



$C = 100$



Tune C using cross-validation $\rightarrow C \approx 7.6$



What happens if the classes are separated by a non-linear decision boundary?

- In linear and generalized linear models, we modeled non-linear relationships via basis functions.

- We can do the same thing here.

- If we have J basis functions $\boldsymbol{\phi} = \{\phi_1(\mathbf{x}), \ldots, \phi_j(\mathbf{x}), \ldots, \phi_J(\mathbf{x})\}$

- The separating hyperplane is then: $\beta_0 + \boldsymbol{\phi}^T \boldsymbol{\beta} = 0$

The dual form of the optimization problem in terms of the basis functions:

$$L_D(\boldsymbol{\alpha}) = \sum_{n=1}^{N}\{\alpha_n\} - \frac{1}{2}\sum_{n=1}^{N}\left\{\sum_{m=1}^{N}\{\alpha_n\alpha_m y_n y_m \boldsymbol{\phi}_n^T\boldsymbol{\phi}_m\}\right\}$$

The dual form of the optimization problem in terms of the basis functions:

$$L_D(\boldsymbol{\alpha}) = \sum_{n=1}^{N}\{\alpha_n\} - \frac{1}{2}\sum_{n=1}^{N}\left\{\sum_{m=1}^{N}\{\alpha_n\alpha_m y_n y_m \boldsymbol{\phi}_n^T\boldsymbol{\phi}_m\}\right\}$$

The actual basis functions do NOT matter!!!

All that matters is their INNER PRODUCT !!!

So instead of explicitly defining the basis functions we define the **kernel**

- The kernel generalizes the inner product.

- The inner product gives us the similarity between two vectors.

- The kernel will therefore be a functional relationship of the similarity!

$$k(\mathbf{x}_n, \mathbf{x}_m) = \boldsymbol{\phi}_n^T \boldsymbol{\phi}_m$$

Rewrite the dual representation for the optimization in terms of the kernel

$$L_D(\boldsymbol{\alpha}) = \sum_{n=1}^{N}\{\alpha_n\} - \frac{1}{2}\sum_{n=1}^{N}\left\{\sum_{m=1}^{N}\{\alpha_n\alpha_m y_n y_m k(\mathbf{x}_n, \mathbf{x}_m)\}\right\}$$

Once the α-parameters have been learned

- The classification of a new point $\mathbf{x}_*$ is then:

$$\text{sign}\left[\hat{\beta}_0 + \sum_{n=1}^{N}\{\hat{\alpha}_n y_n k(\mathbf{x}_*, \mathbf{x}_n)\}\right]$$

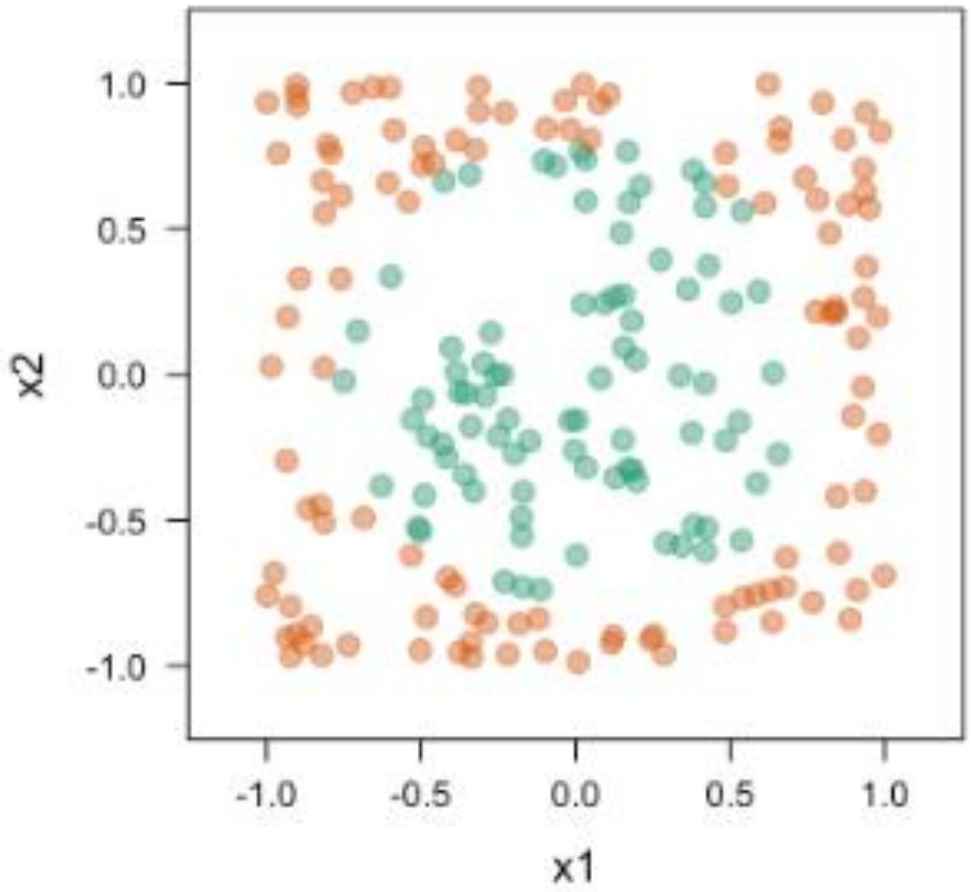
Support Vector Machines (SVMs)

- The soft-margin classifier uses a linear kernel: $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$.

- The soft-margin classifier combined with the kernel to handle non-linear hyperplanes is known as the **SVM**

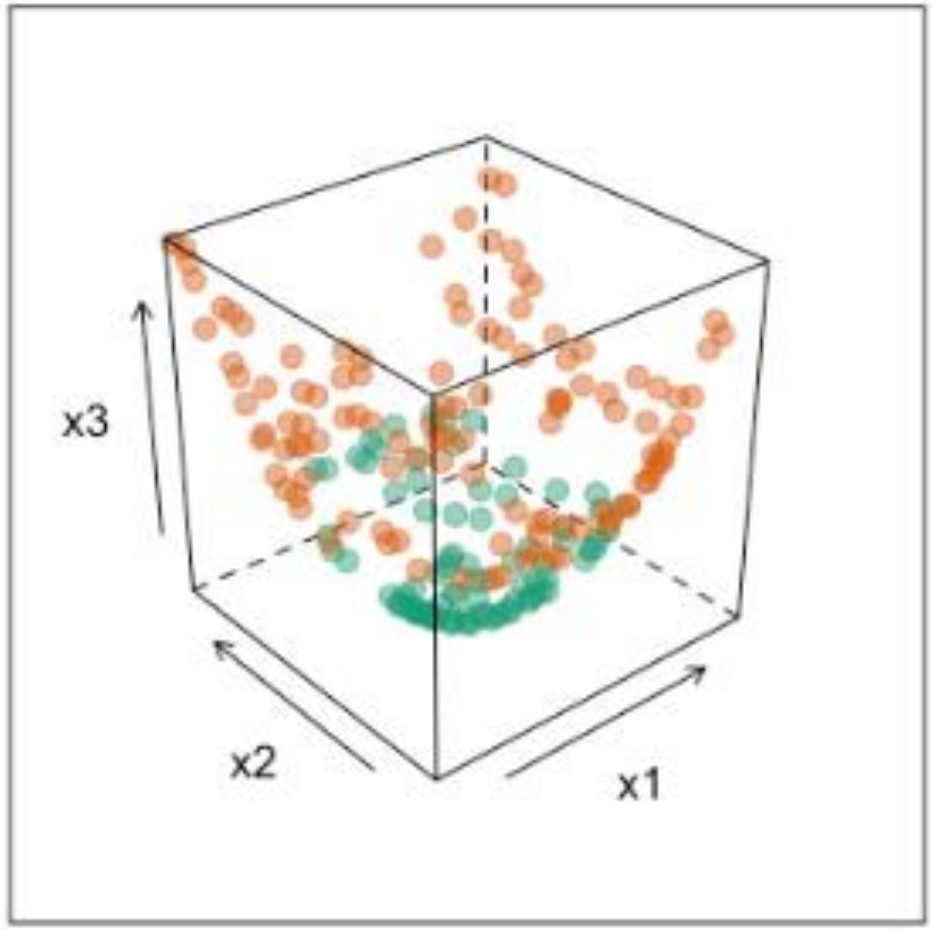
- Common kernels:
- d-th degree polynomial: $k(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^T \mathbf{x}')^d$
- Radial basis: $k(x, x') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|_2^2)$
- Neural network: $k(x, x') = \tanh(\kappa_1(\mathbf{x}^T \mathbf{x}') + \kappa_2)$

SVM example with circular decision boundary



Example from: https://bradleyboehmke.github.io/HOML/svm.html#fig:svm-hmc

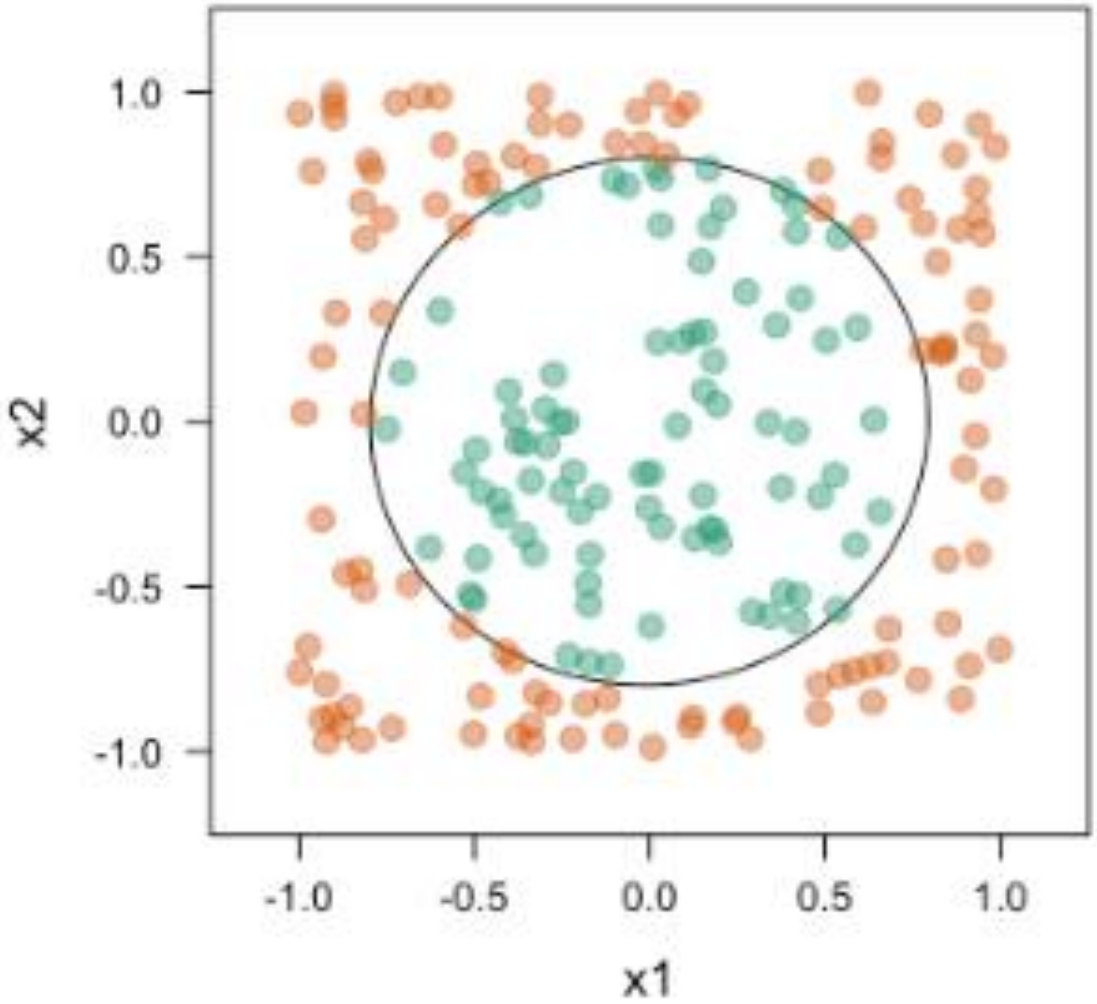
Use the d-th degree polynomial kernel with $d = 2$



- Equivalent to using a basis expansion of $x_3 = x_1^2 + x_2^2$
- In the x_3-space a linear separating hyperplane can be defined between the two classes.

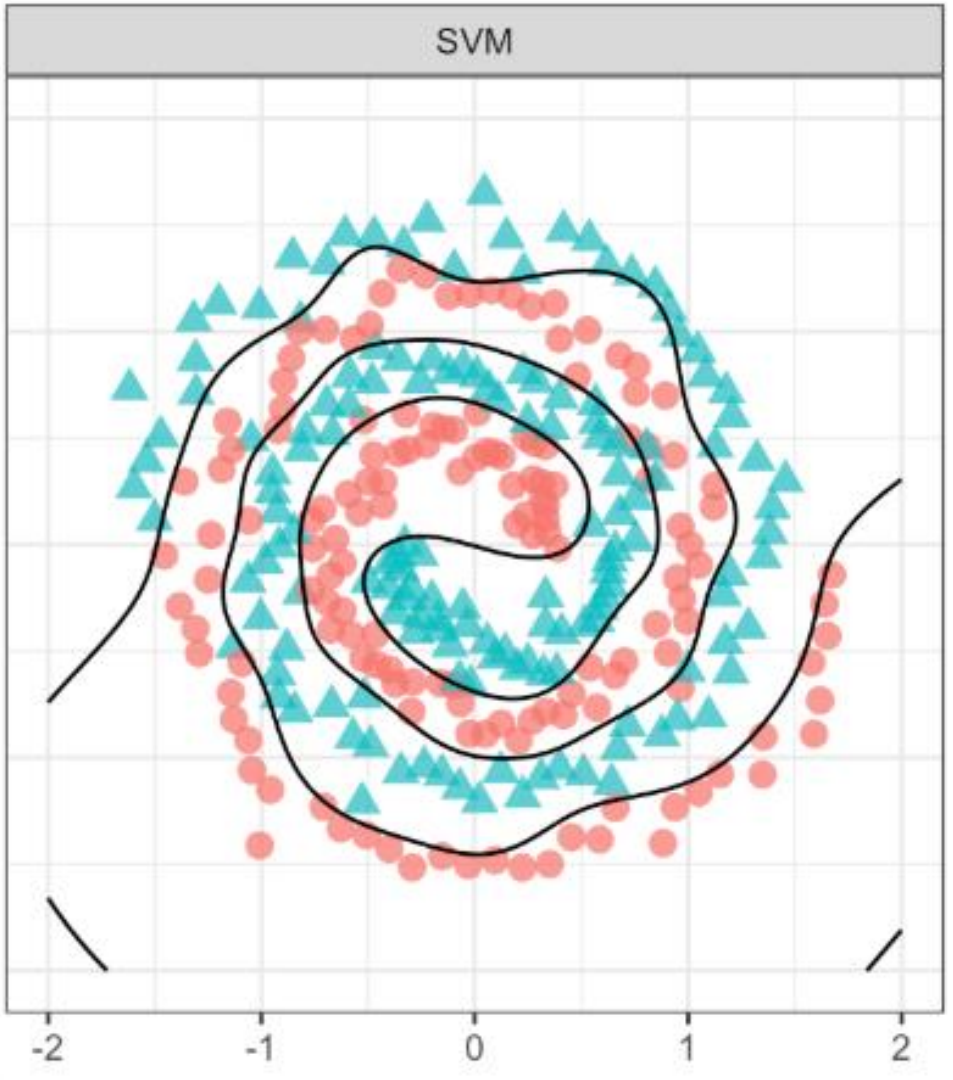
Example from: https://bradleyboehmke.github.io/HOML/svm.html#fig:svm-hmc

The decision boundary in the original input space is non-linear!



Example from: https://bradleyboehmke.github.io/HOML/svm.html#fig:svm-hmc

SVM with radial basis function applied to highly non-linear class separation



Example from:
https://bradleyboehmke.github.io/HOML/svm.html#fig:svm-hmc