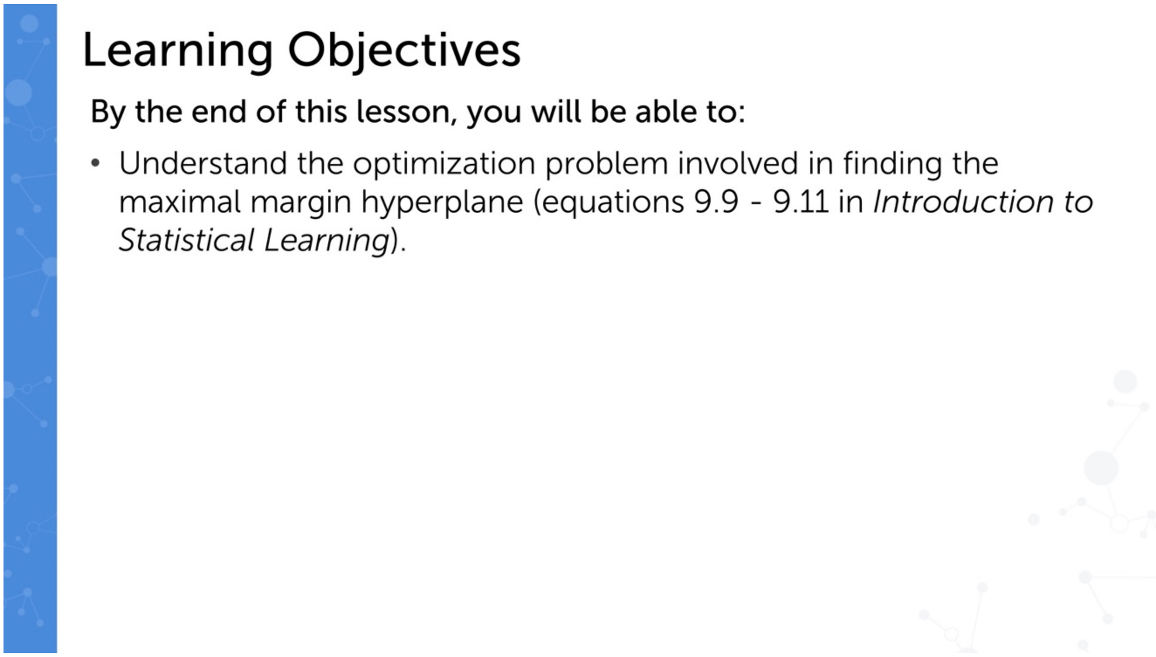
UNIVERSITY OF WISCONSIN
DATA SCIENCE

DS 740

Data Mining

Technical Details of the Maximal Margin Classifier

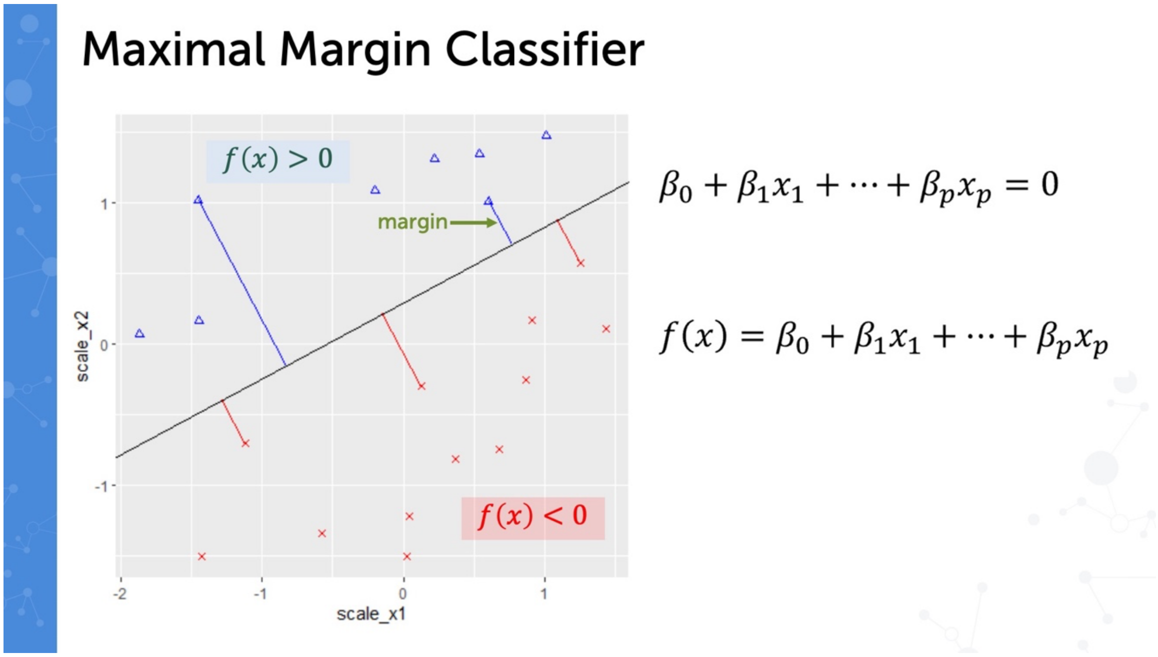
Important note: Transcripts are **not** substitutes for textbook assignments.



Learning Objectives

By the end of this lesson, you will be able to:

- Understand the optimization problem involved in finding the maximal margin hyperplane (equations 9.9 - 9.11 in *Introduction to Statistical Learning*).

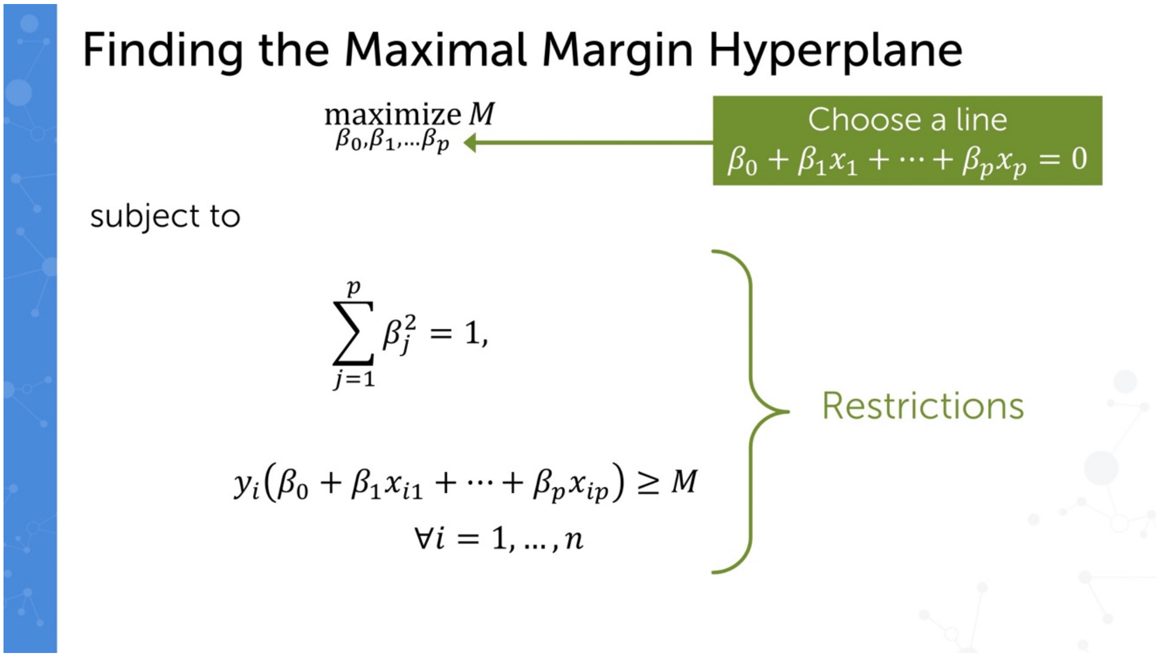


Maximal Margin Classifier

$$\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p = 0$$

$$f(x) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

We've seen that the maximum margin classifier works by finding a hyperplane of the form beta sub 0 plus beta sub 1 times the first predictor variable, plus dot, dot, dot, up through beta sub p times the p-th predictor variable equals 0, in such a way that maximizes the margin or the smallest perpendicular distance between the line and any of the data points. To use this as a classifier, we set f of x equal to the left hand side of the equation for the hyperplane, and categorize points based on whether f of x is greater than 0 or less than 0, meaning the points are above or below the hyperplane.



Finding the Maximal Margin Hyperplane

$$\underset{\beta_0,\beta_1,\ldots\beta_p}{\text{maximize}}\, M \longleftarrow \boxed{\begin{array}{c}\text{Choose a line}\\ \beta_0+\beta_1x_1+\cdots+\beta_px_p=0\end{array}}$$

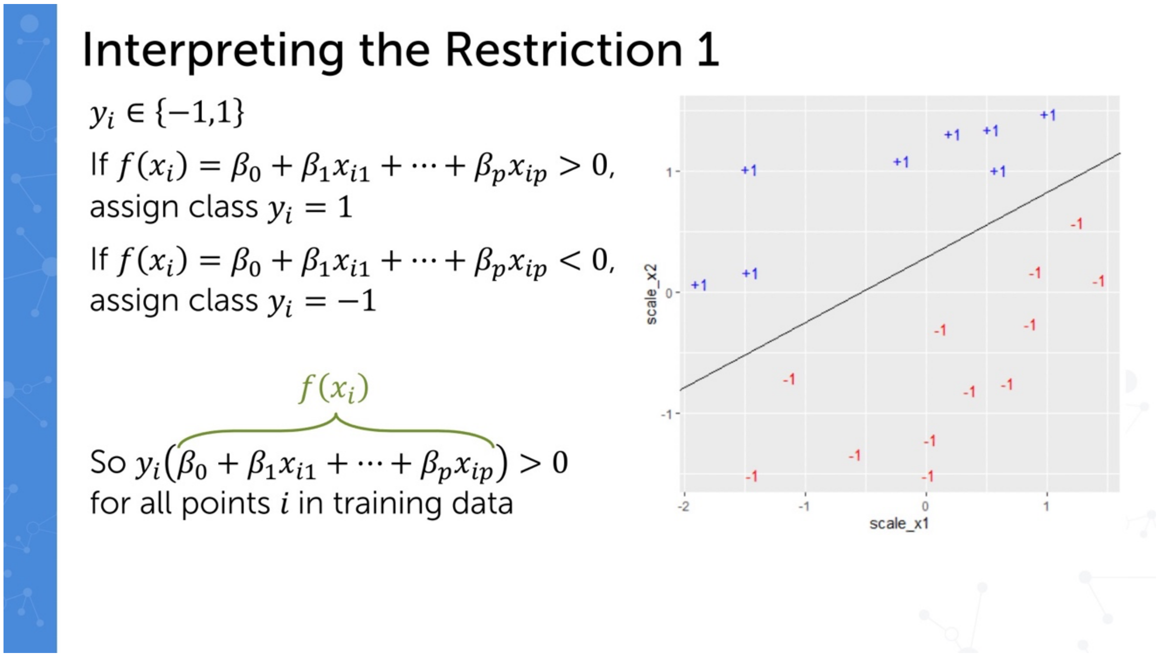
subject to

$$\left.\begin{array}{c}\displaystyle\sum_{j=1}^{p}\beta_j^2=1,\\[2em] y_i(\beta_0+\beta_1x_{i1}+\cdots+\beta_px_{ip})\geq M\\ \forall i=1,\ldots,n\end{array}\right\}\ \text{Restrictions}$$

The technical details of finding the maximum margin hyperplane can look pretty intimidating. So let's break these down. At its heart, this is an optimization problem that can be solved using optimization techniques that you may have learned about in previous courses, such as Lagrange multipliers.

We're trying to maximize m, the margin, by choosing parameters beta sub 0 through beta sub p. That means we're just trying to choose a line or hyperplane of the same form as we've already looked at. All of the rest of these details are the restrictions on what line we can choose.

Notes:
For more information about solving this optimization problem using Lagrange multipliers, see p. 420 of The Elements of Statistical Learning by Hastie, Tibshirani, and Friedman.



Interpreting the Restriction 1

$y_i \in \{-1,1\}$

If $f(x_i) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} > 0$, assign class $y_i = 1$

If $f(x_i) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} < 0$, assign class $y_i = -1$

$$\text{So } y_i \overbrace{\left(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}\right)}^{f(x_i)} > 0$$

for all points i in training data

To interpret these restrictions, we need to have the values of our response variable, y sub i, be in the set negative 1, 1. So in the figure shown here, we've relabeled all of our blue category points as plus 1s, and all of our red category points as minus 1s. Then, if we want to categorize a new data point, call it x sub i, we plug it into the function f of x And if f of x of i is greater than 0, we assign it to class y sub i equal to 1. If f of x sub i is less than 0, we assign it to class negative 1.

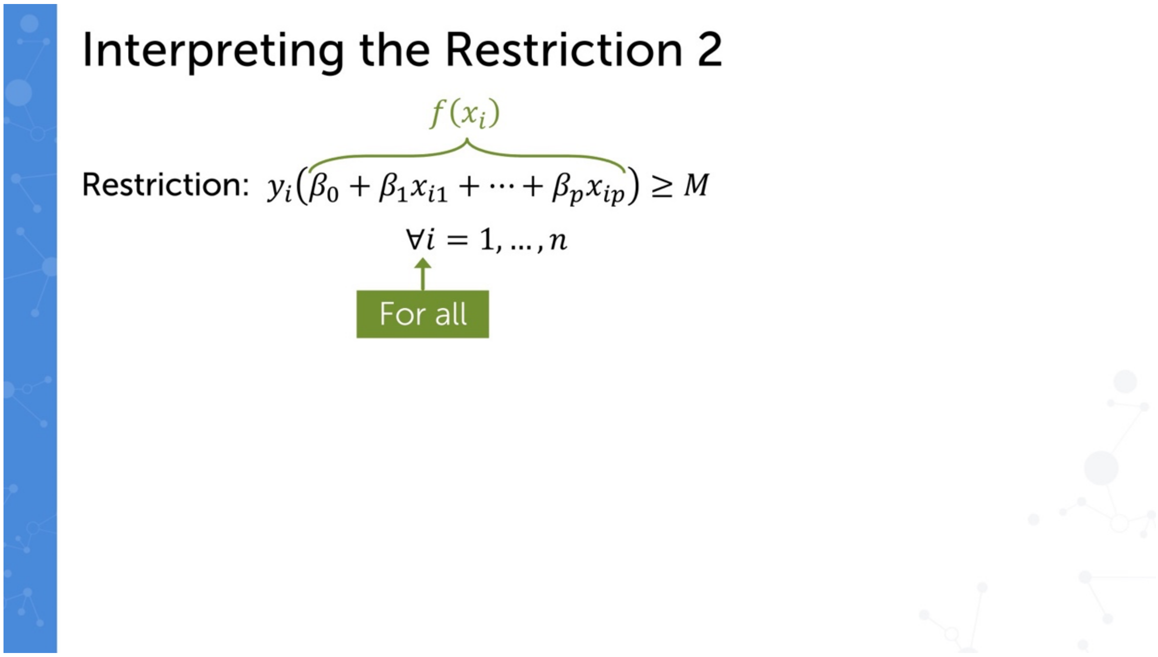
But this means that y sub i times the quantity f of x sub i has to be greater than 0 for all the points in the training dataset, assuming that our dataset could be separated into categories by a hyperplane in the first place. To understand this, let's take a point in the blue plus 1 category. For these points, y sub i is plus 1 because that's the class that they belong to.

And f of x sub i is greater than 0 in order for them to be categorized as part of class 1. So we get y sub i is plus 1, f of x sub i is positive. So we have 1 times a positive number. That's greater than 0.

On the other hand, take a point in the red negative 1 category. For these points, y sub i equals negative 1 and f of x sub i is less than 0 as well. So we have negative 1 times another negative number. A negative times a negative is a positive. So as long as a point is on the correct side of the separating hyperplane, y sub i times f of x sub i must be greater than 0 for that point.

Notes:

If the point is on the correct side of the line, either *f*(*x*) and *y* are both positive or both negative.



Interpreting the Restriction 2

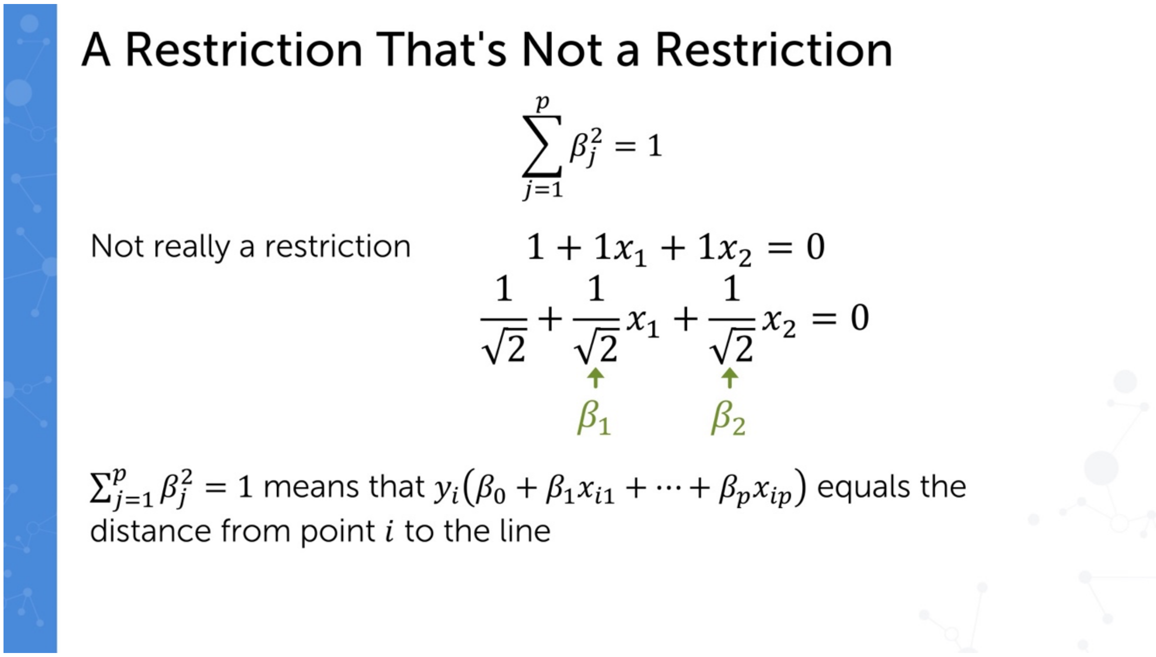
$$\text{Restriction: } y_i\big(\overbrace{\beta_0+\beta_1 x_{i1}+\cdots+\beta_p x_{ip}}^{f(x_i)}\big) \geq M$$

$$\forall i = 1, \ldots, n$$

For all

Now let's look back at our restriction, which was that y sub i times f of x sub i must be greater than or equal to m. The upside down a symbol here means for all. So this restriction is saying that y sub i times f of x sub i must be greater than or equal to m for all points i from 1 up to n, the number of points in the training dataset.

Now we know that if we set m equal to 0, we can make this restriction work. We can find the line that will satisfy it. Because we already said that if all of the points are on the correct side of the line, then y sub i times f of x sub i is greater than or equal to 0. The optimization problem is all about asking, is there a value of m that's bigger than 0 that will still work? And if so, what's the largest possible value?



A Restriction That's Not a Restriction

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

Not really a restriction

$$1 + 1x_1 + 1x_2 = 0$$

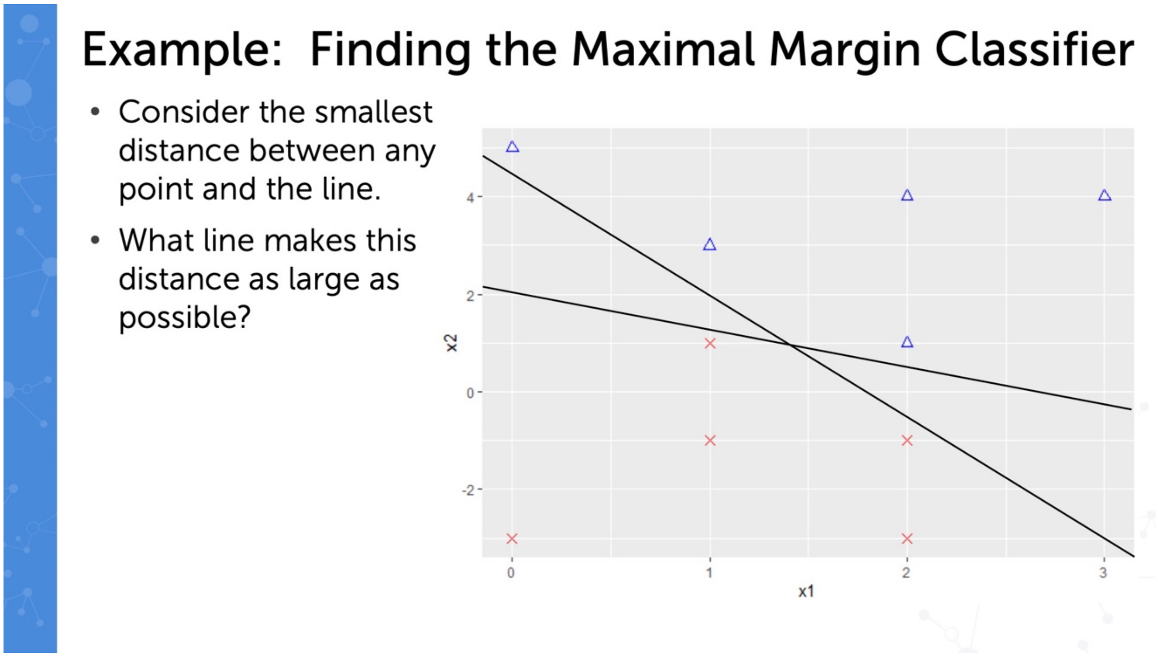
$$\frac{1}{\sqrt{2}} + \underset{\underset{\beta_1}{\uparrow}}{\frac{1}{\sqrt{2}}} x_1 + \underset{\underset{\beta_2}{\uparrow}}{\frac{1}{\sqrt{2}}} x_2 = 0$$

$\sum_{j=1}^{p} \beta_j^2 = 1$ means that $y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip})$ equals the distance from point i to the line

The other part of the restriction for this optimization problem is that the sum of the squares of the coefficients, beta sub j, as j goes from 1 up to p, the number of predictor variables, has to equal 1. This isn't really a restriction, because if you think about the equation for any hyperplane we might be interested in, we can take all of the coefficients and multiply them by the same constant to get a different equation for the same hyperplane.

But now, the sum of the squares of the coefficients of all of the terms with x's in them add up to 1. So this restriction isn't really restricting our choice of hyperplane. It's just restricting what coefficients we use to represent that hyperplane.

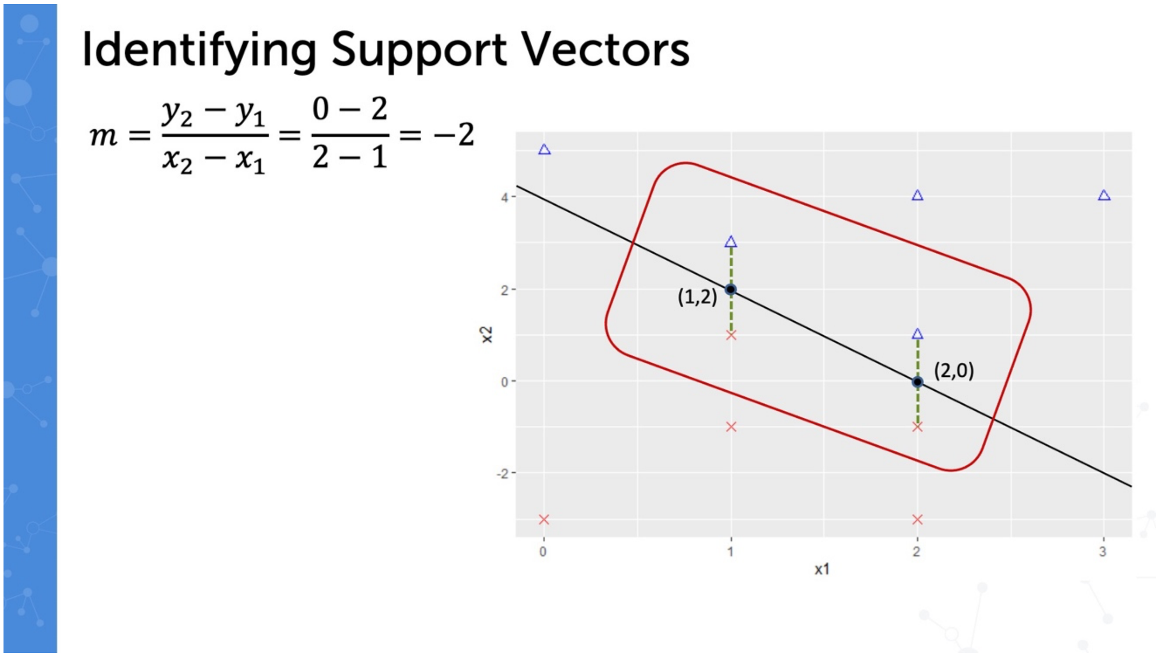
So, why do we want that restriction on our choice of coefficients? Well, it's because this restriction means that y sub i times f of x sub i equals the perpendicular distance from the point to the line. So this restriction is all about making sure that the margin m really represents the smallest distance between any point and the line.



Example: Finding the Maximal Margin Classifier

- Consider the smallest distance between any point and the line.
- What line makes this distance as large as possible?

For example, let's find the maximal margin classifier for these points by hand. There are many possible lines we could draw that would separate these two classes. But in order to find the best possible line, we want to consider the smallest distance between any point and the line. That's the margin. The best possible line that separates these two classes is the one that makes this smallest distance or margin as large as possible.

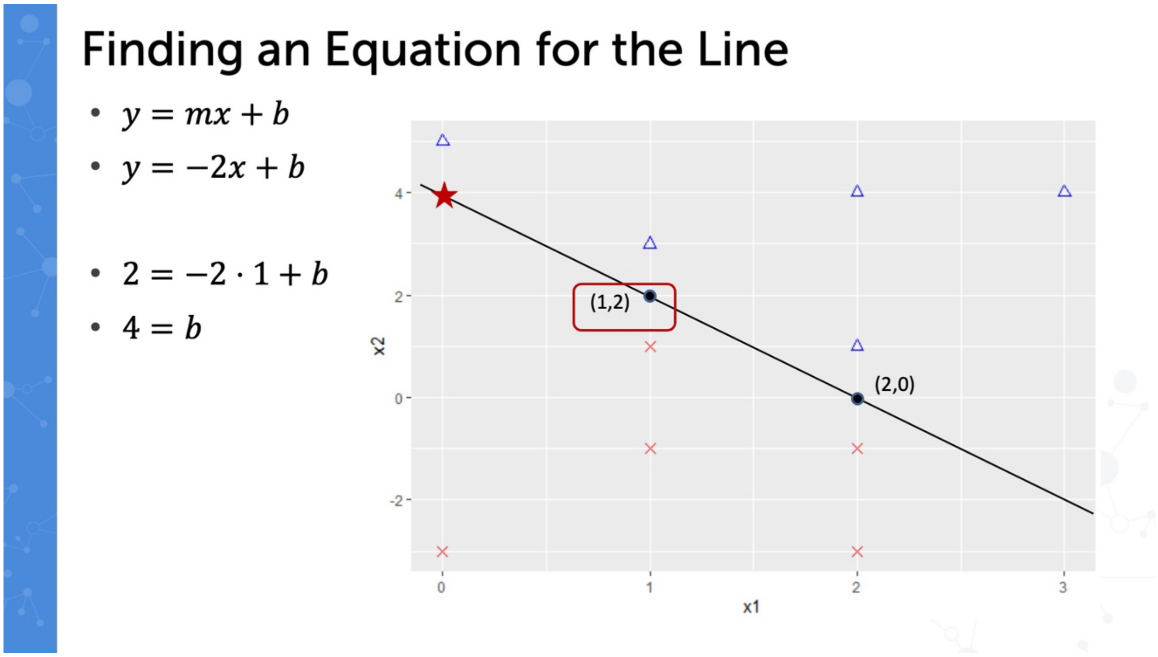


Identifying Support Vectors

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{0-2}{2-1} = -2$$

Let's focus on these four points in the middle. Because these points are the closest to the other class, these will be our support vectors. In this case, the vertical distance between each pair of points is the same. This means that we can get the best margin by having our maximal margin classifier go through the midpoint of each of these pairs of points.

We can then compute the slope of the maximal margin classifier by finding the slope of the line that goes through the points 1, comma, 2 and 2, comma, 0. Recall from algebra that the slope of a line is the change in y divided by the change in x. So this gives us a slope of negative 2



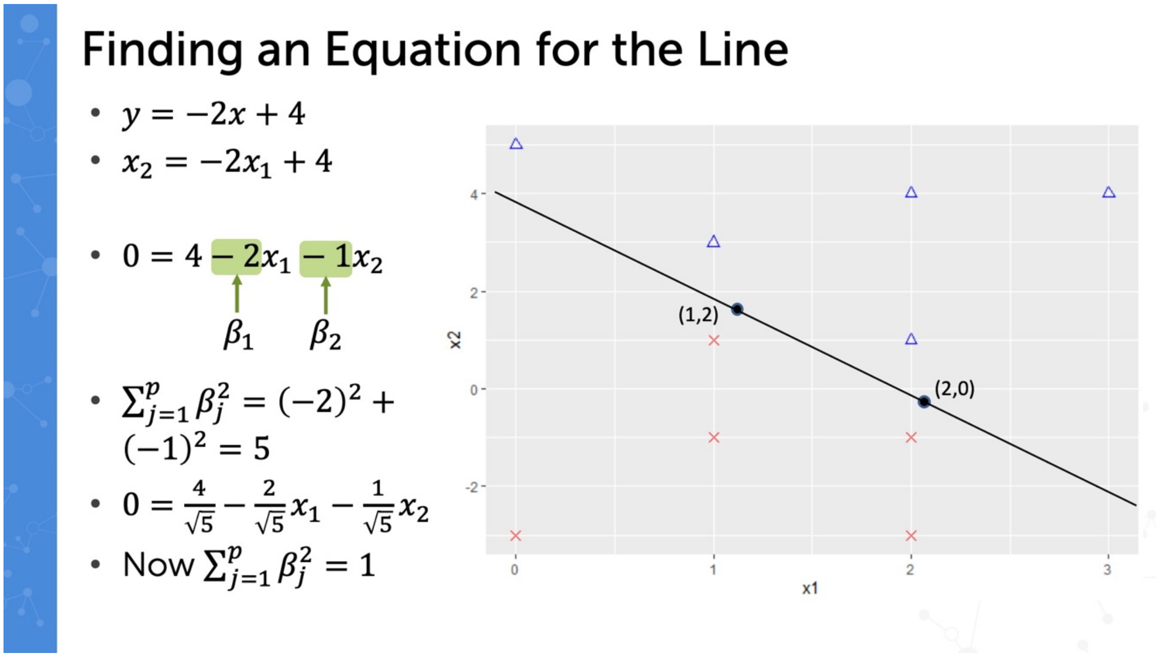
Finding an Equation for the Line

- $y = mx + b$
- $y = -2x + b$

- $2 = -2 \cdot 1 + b$
- $4 = b$

Then we'll use a formula for an equation of a line. I like slope intercept format, y equals mx plus b. We just found the slope m so our equation becomes y equals negative 2x plus b. Then we want to find the y-intercept b. And we can do that by plugging in the values of x and y for one of the two points that we know is on the line.

We could pick either one. Let's say we're using the point 1, comma, 2. Then plugging in to 2 for y and 1 for x, we get 2 equals negative 2 times 1 plus b. Solving this for b gives us b equals 4. And looking at our graph, this looks about right for the height of the y-intercept when x is equal to 0.

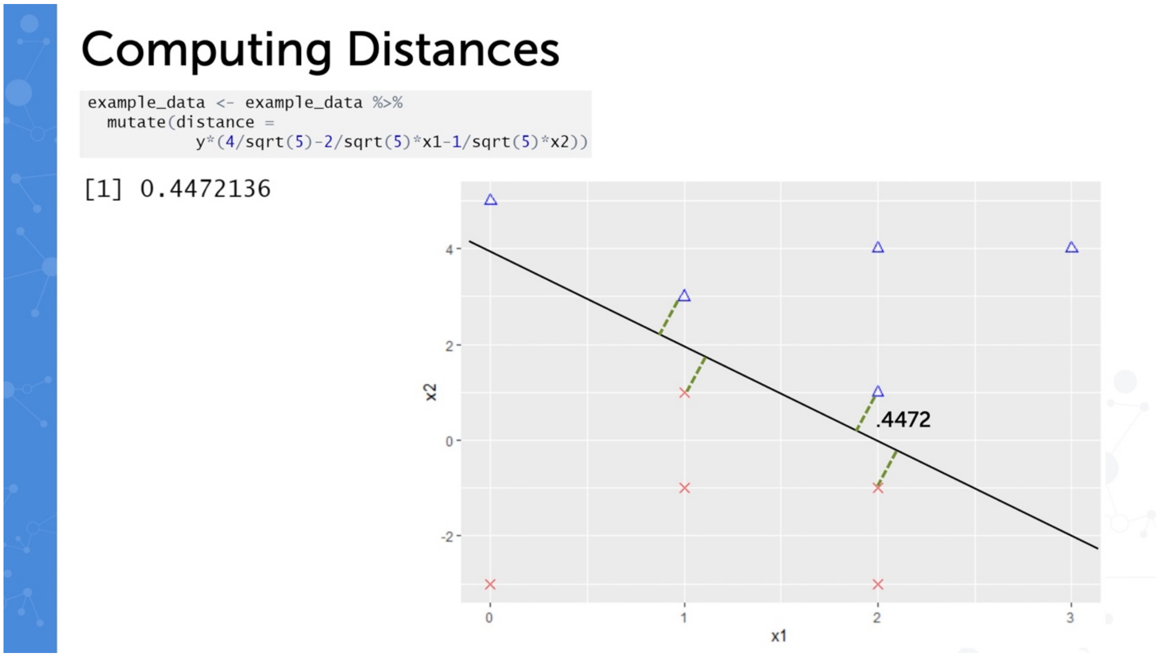


Finding an Equation for the Line

- $y = -2x + 4$
- $x_2 = -2x_1 + 4$
- $0 = 4 \underbrace{-2}_{\beta_1}x_1 \underbrace{-1}_{\beta_2}x_2$
- $\sum_{j=1}^{p} \beta_j^2 = (-2)^2 + (-1)^2 = 5$
- $0 = \frac{4}{\sqrt{5}} - \frac{2}{\sqrt{5}}x_1 - \frac{1}{\sqrt{5}}x_2$
- Now $\sum_{j=1}^{p} \beta_j^2 = 1$

Now we know that the equation for our best separating line is y equals negative 2x plus 4. In the language of support vector classifiers, our predictor variables are x sub 1 and x sub 2 so that y can be reserved to represent which class each point belongs to.

Next, we want to get this in the form of stuff equal to 0 so we can rearrange this equation by subtracting x sub 2 from both sides. Now we have beta 1 is negative 2, the coefficient of x sub 1. And beta 2 is negative 1, the coefficient of x sub 2. But this means that the sum of the beta sub j's squared, not counting the y-intercept of 4, is negative 2 squared plus negative 1 squared, which is 5. And we want this sum to equal 1 to meet the restriction for a maximum margin classifier. The way we accomplish this is by dividing each term in our equation by the square root of 5. Now we have the sum of the squares of the beta sub j's is equal to 1 and our equation still represents the same line.



Computing Distances

```
example_data <- example_data %>%
  mutate(distance =
           y*(4/sqrt(5)-2/sqrt(5)*x1-1/sqrt(5)*x2))
```

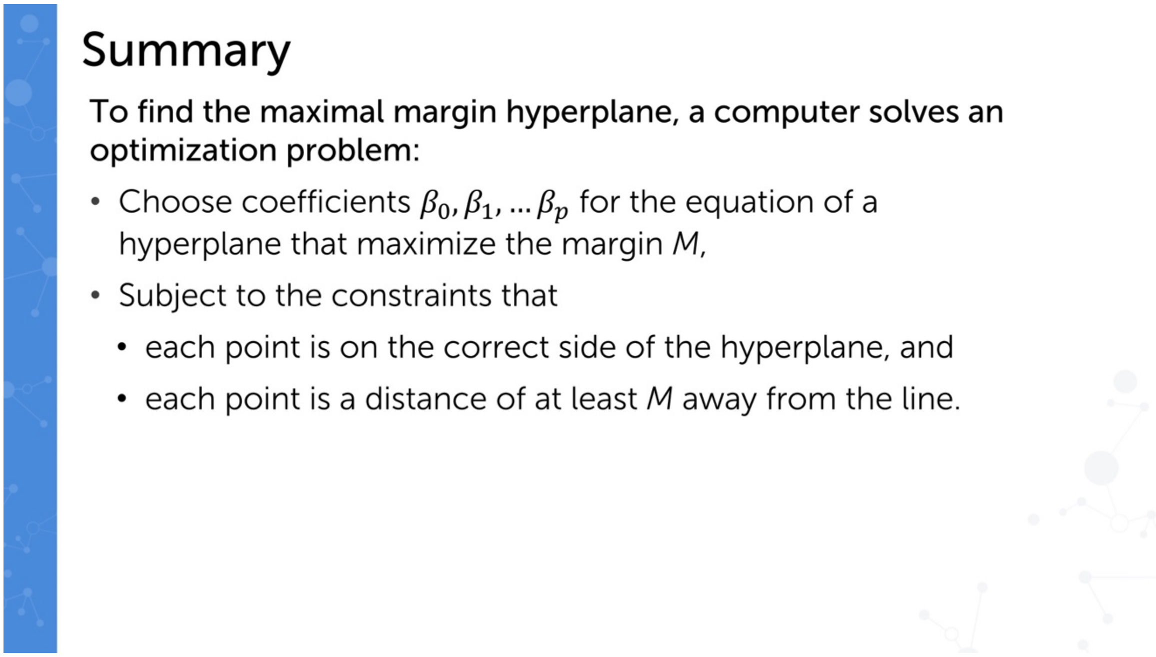
```
[1] 0.4472136
```

Now that our equation is in the correct format, we can compute the perpendicular distance between each point and the line by taking that point's value of x1 and x2 and plugging them in to the formula for our equation for the line and then multiplying by the value of y, which is either positive or negative 1, whichever one makes the distance positive. The margin then is the minimum distance between any point in the line. In this case, that's 0.4472, which is the distance between each of our four support vectors and the maximum margin classifier.

Notes:

```
example_data <- example_data %>%
  mutate(distance = y*(4/sqrt(5)-2/sqrt(5)*x1-1/sqrt(5)*x2))

margin = min(example_data$distance)
margin
```



Summary

To find the maximal margin hyperplane, a computer solves an optimization problem:

- Choose coefficients $\beta_0, \beta_1, \ldots \beta_p$ for the equation of a hyperplane that maximize the margin *M*,
- Subject to the constraints that
  - each point is on the correct side of the hyperplane, and
  - each point is a distance of at least *M* away from the line.