Announcement

- Kaggle submission by Wed 11:59 pm
- Kaggle report due Thursday 11:59 pm
- Midterm 1 Next Friday (2/21) in the class
  ; covers week 1- this week (SVM)

CSCI4622 Machine Learning 2020 Spring

Support Vector Machine

Geena Kim

*Slide contents adopted from ISLR material

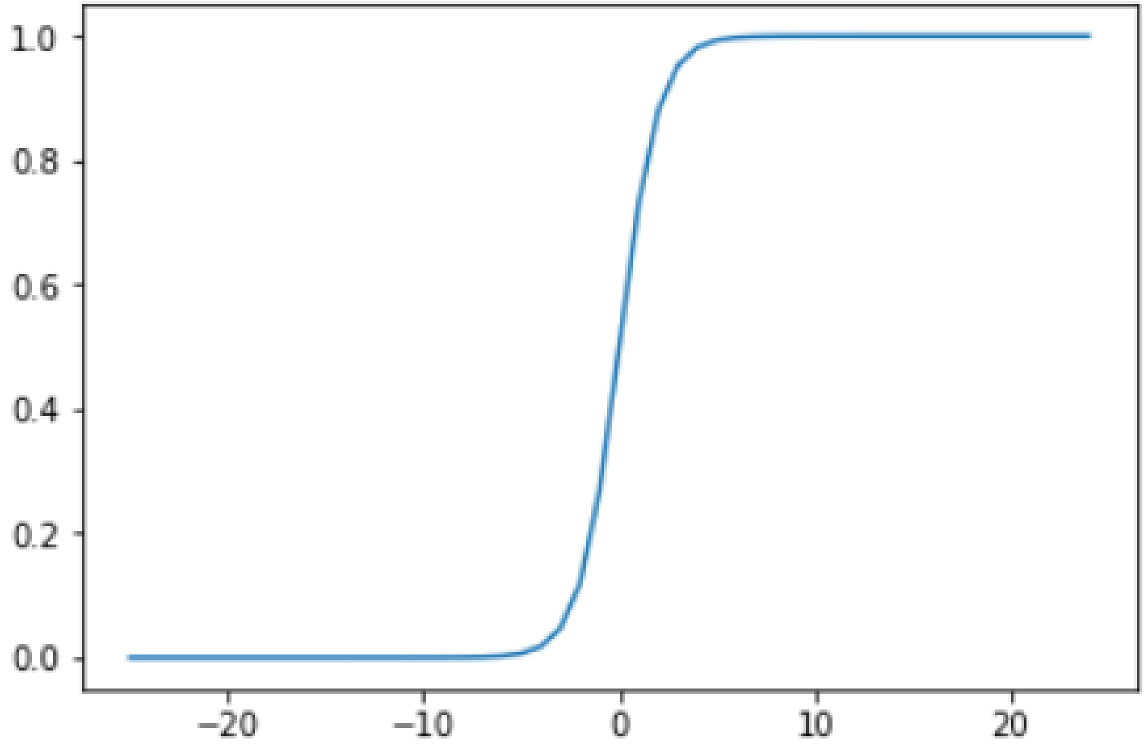
Review: Logistic Function

$$P^{(i)} \in \mathbb{R}[0,1]$$

$$P^{(i)} = \sigma(z^{(i)})$$

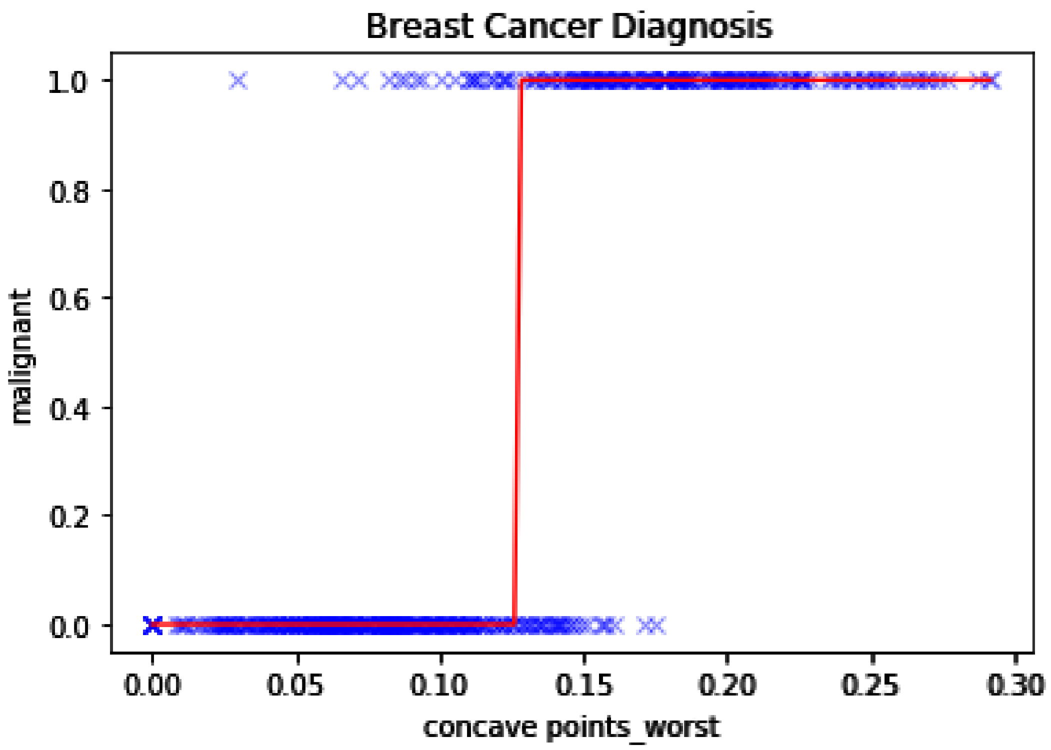
$$\sigma(z) = \frac{1}{1+e^{-z}}$$

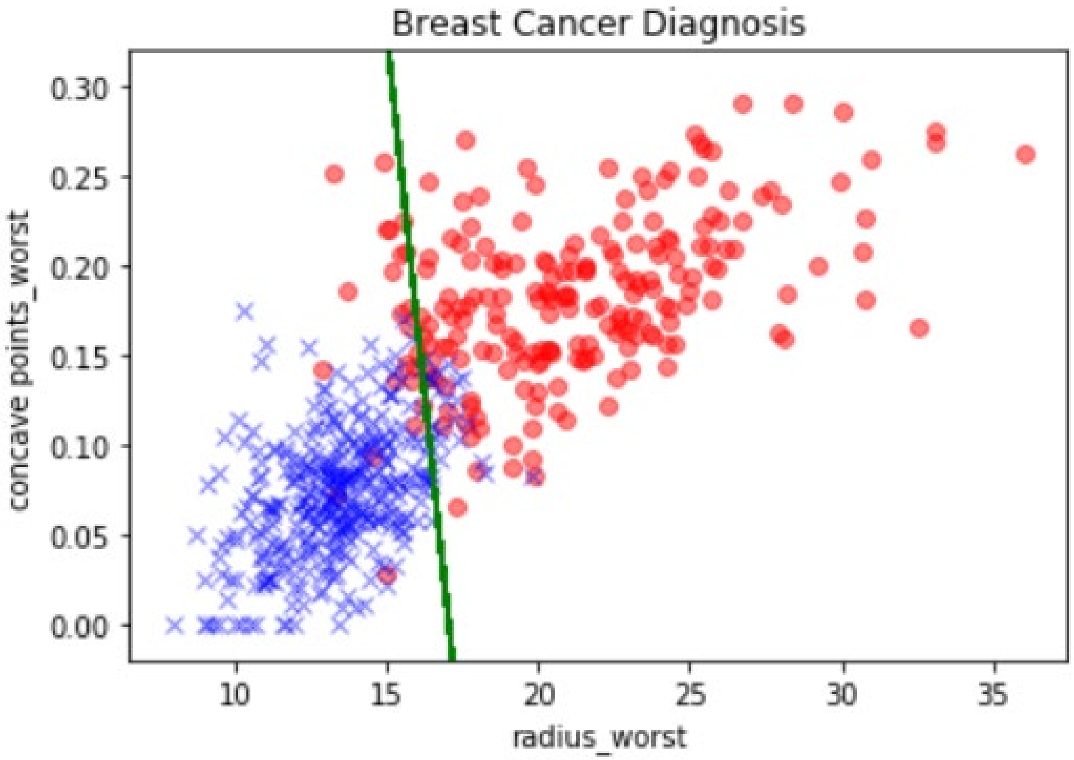
$$z^{(i)} = \boldsymbol{W} \cdot \boldsymbol{X} + b$$



Called “logit” and is related to the decision boundary

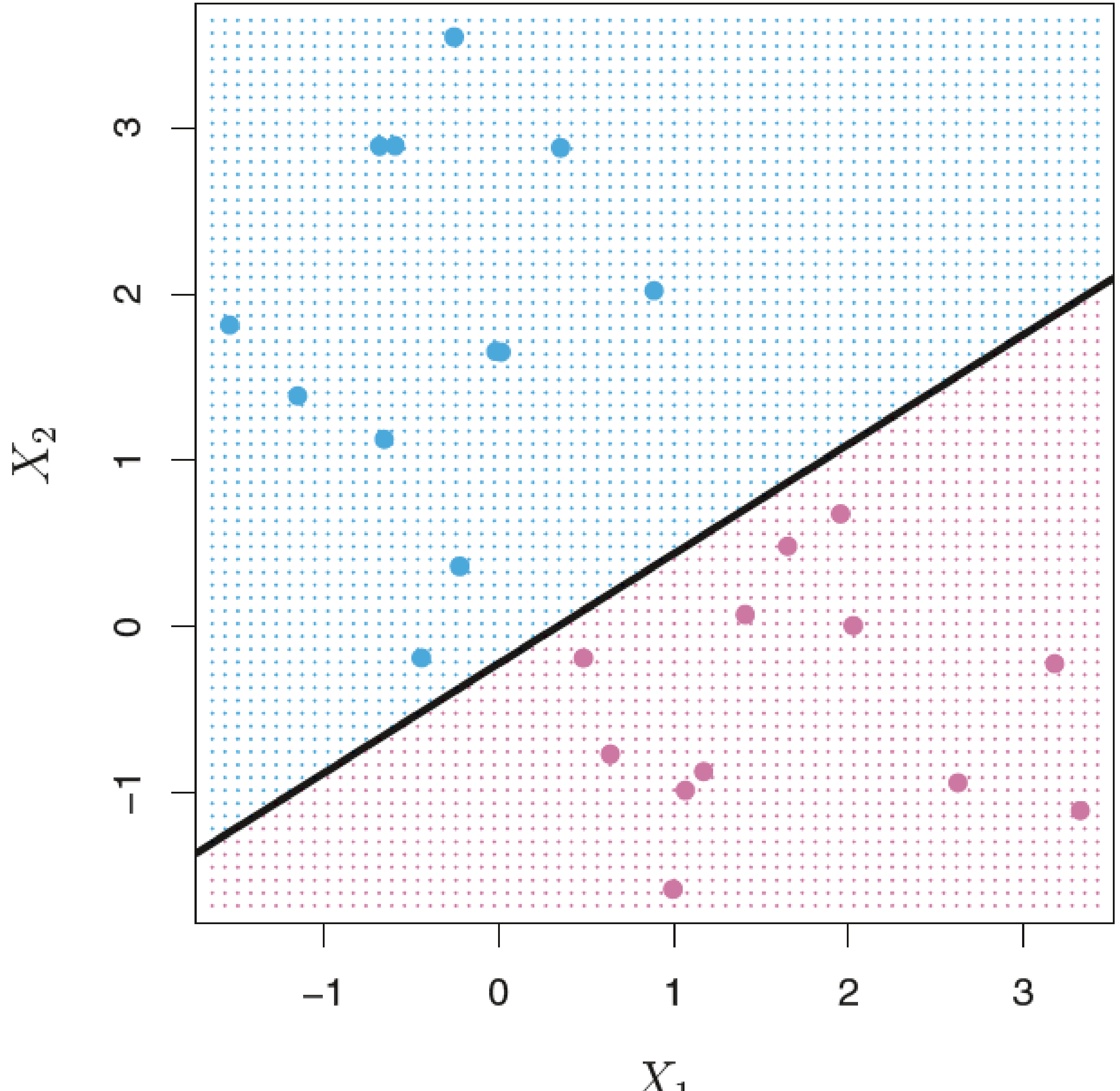
Review: Logistic Regression Decision Boundary





$z = 0.443\ x1 + 2.76\ x2 - 7.57 = 0$

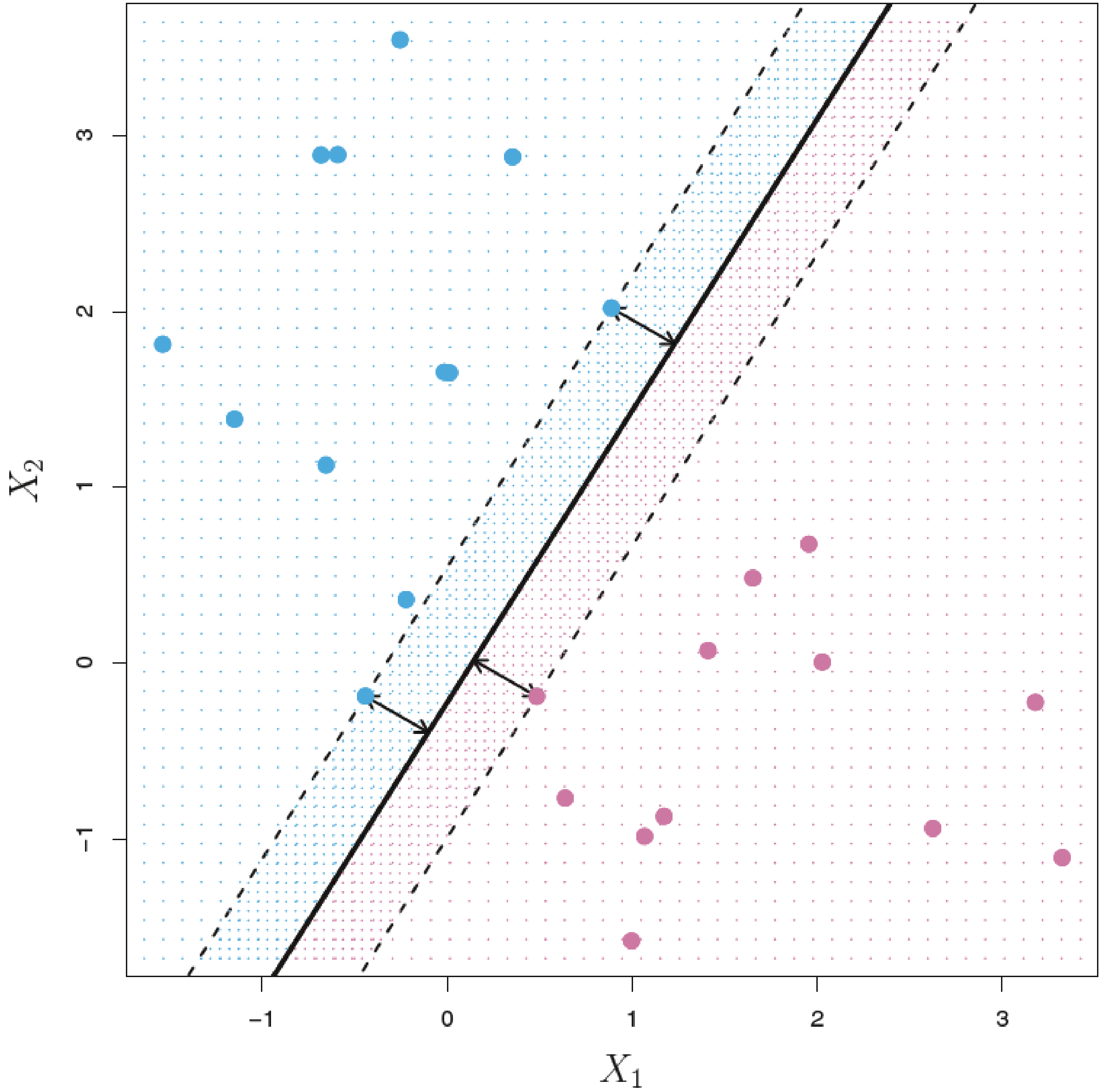
Hyperplane as a Decision Boundary



We can separate the two classes using a hyper plane!

This hyperplane is called “separating hyperplane”

Maximum margin classifier

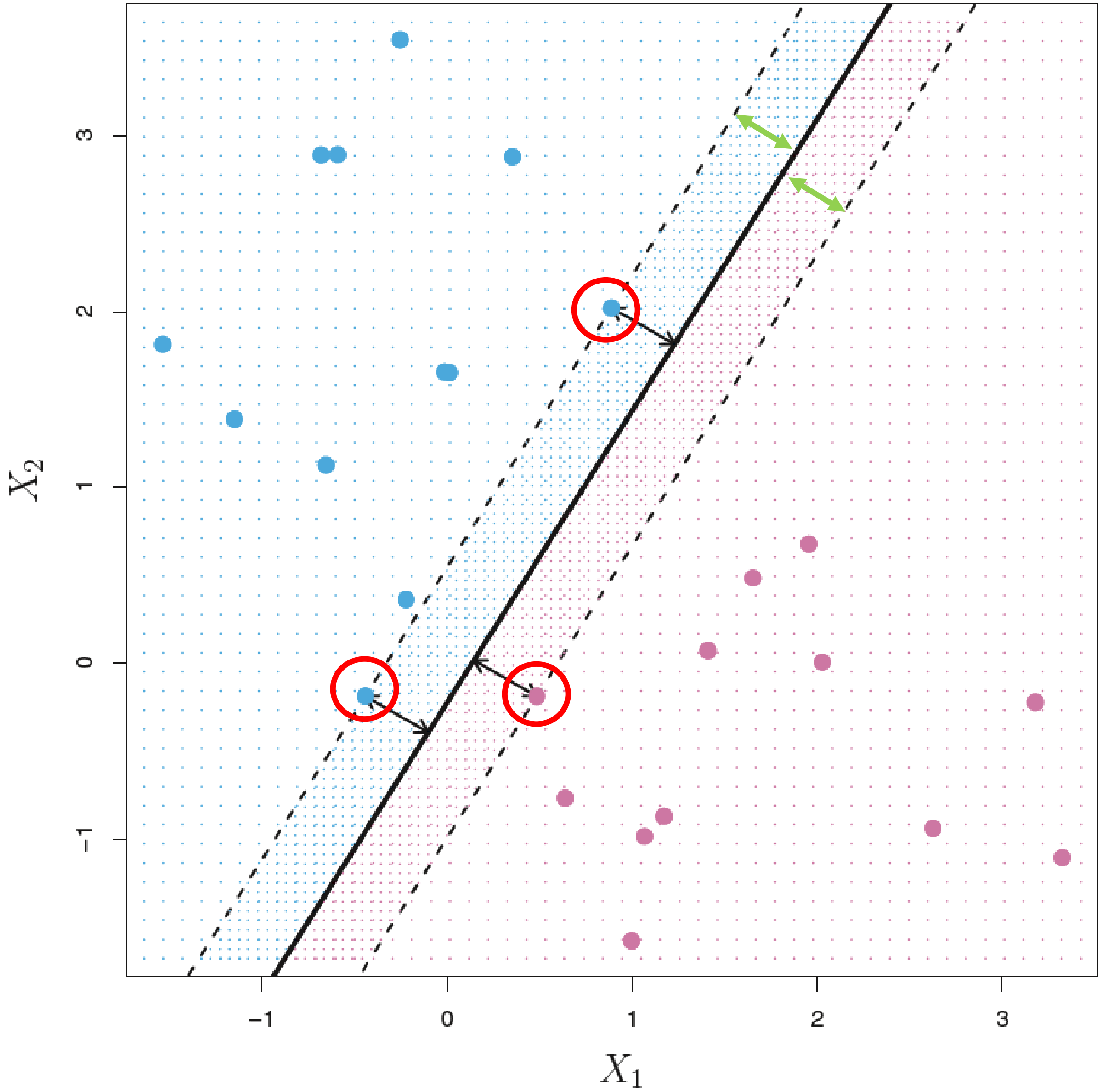


Which hyperplane should we choose?

The one with the least likely to misclassify the test data

= The one with the biggest margin

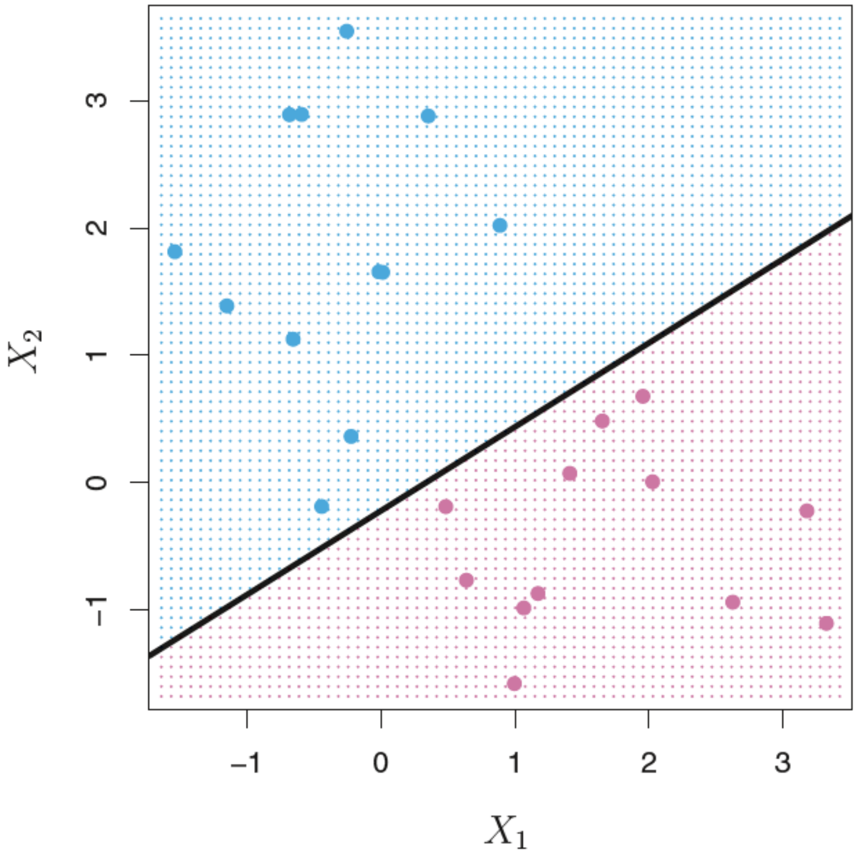
Maximum margin classifier



Support

Margin

How to find the maximal margin hyperplane?



$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > 0$$

for all $i = 1, \ldots, n$

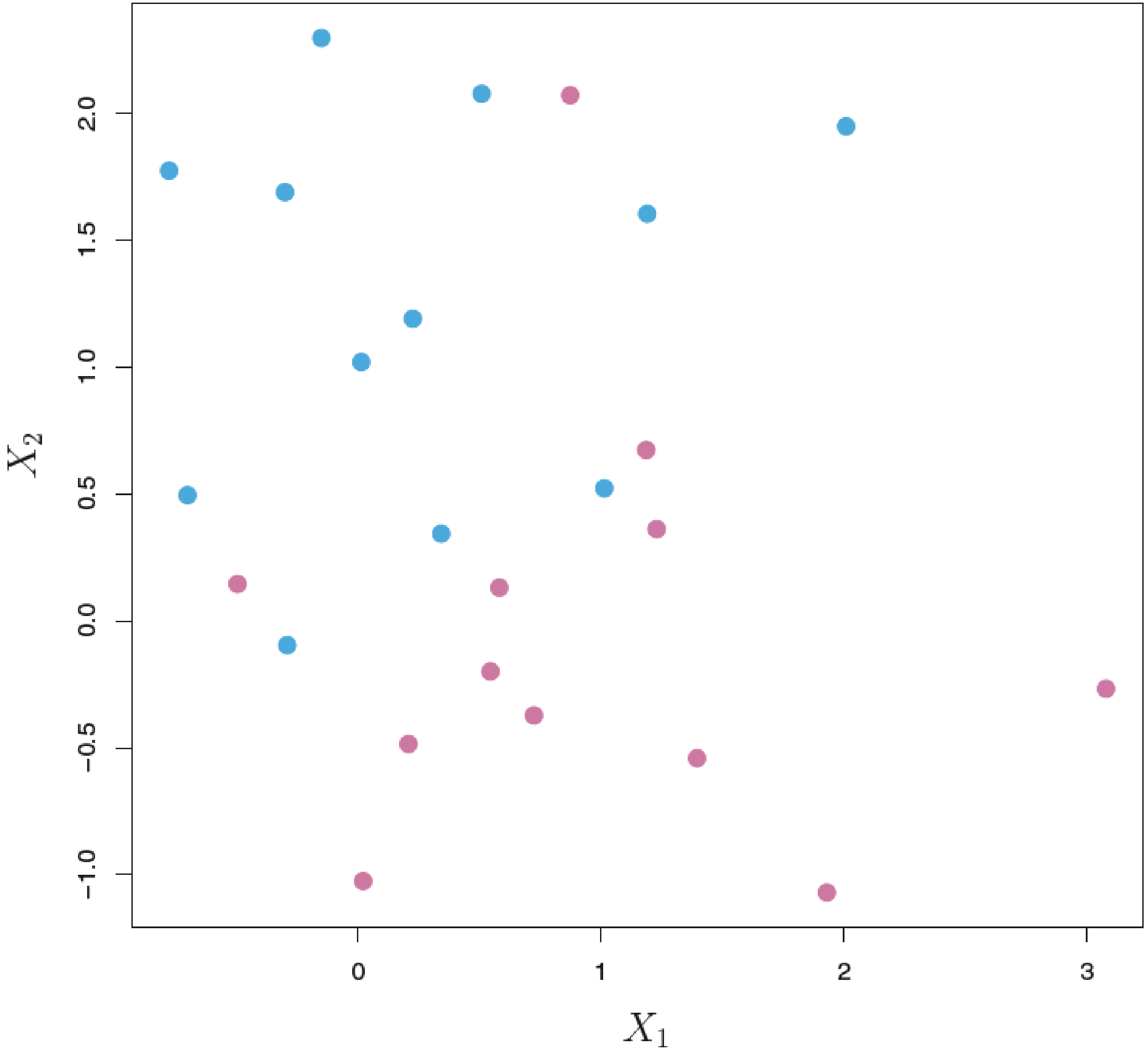
$$y_1, \ldots, y_n \in \{-1, 1\}$$

$$\underset{\beta_0, \beta_1, \ldots, \beta_p, M}{\text{maximize}} \; M$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall \, i = 1, \ldots, n$$

How to deal with an inseparable case

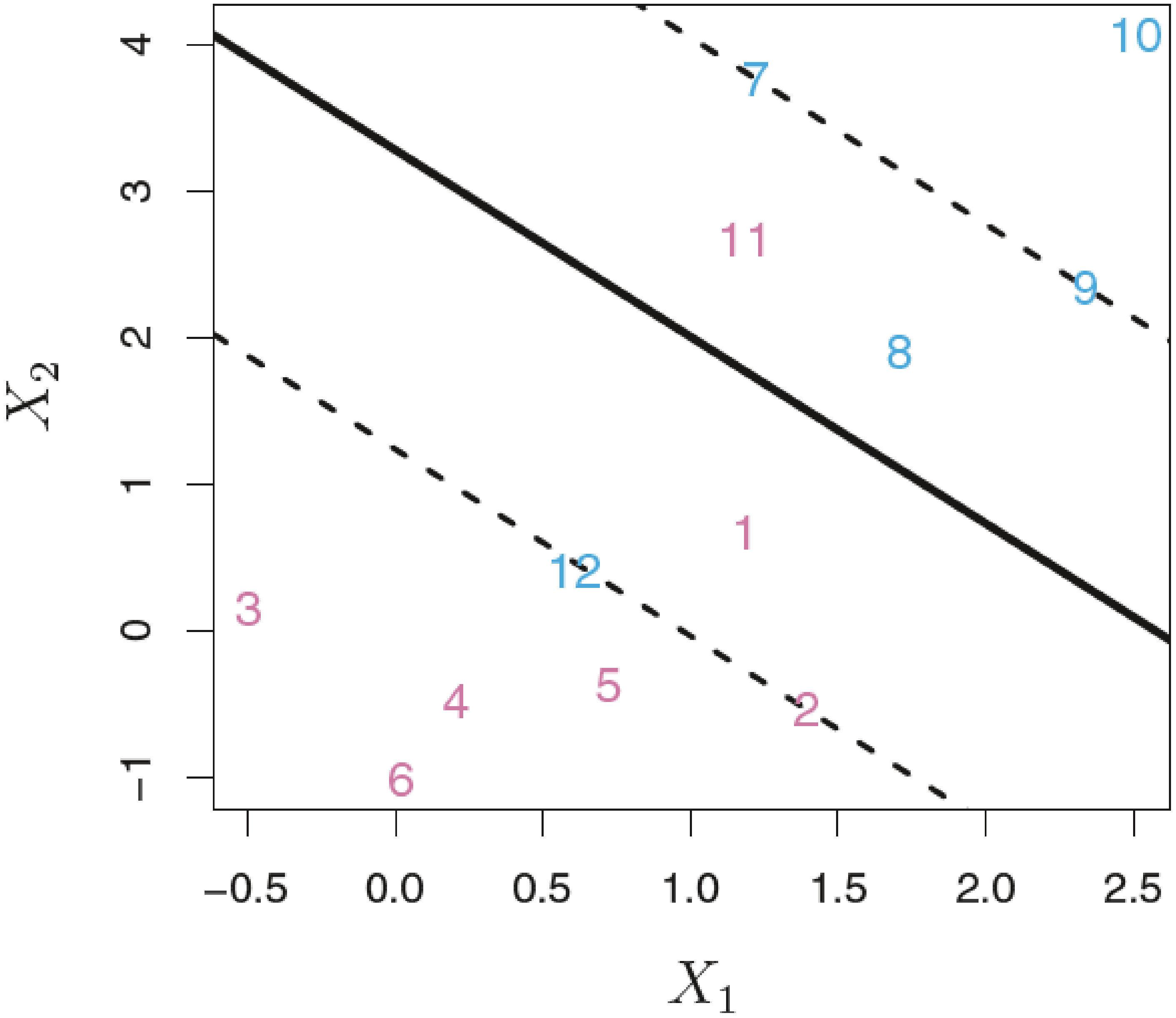


We'll have to accept some errors by softening the margin

"soft margin classifier"

or called
"support vector classifier"

Soft margin classifier



Soft margin classifier

Formulating support vector classifier

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1$$

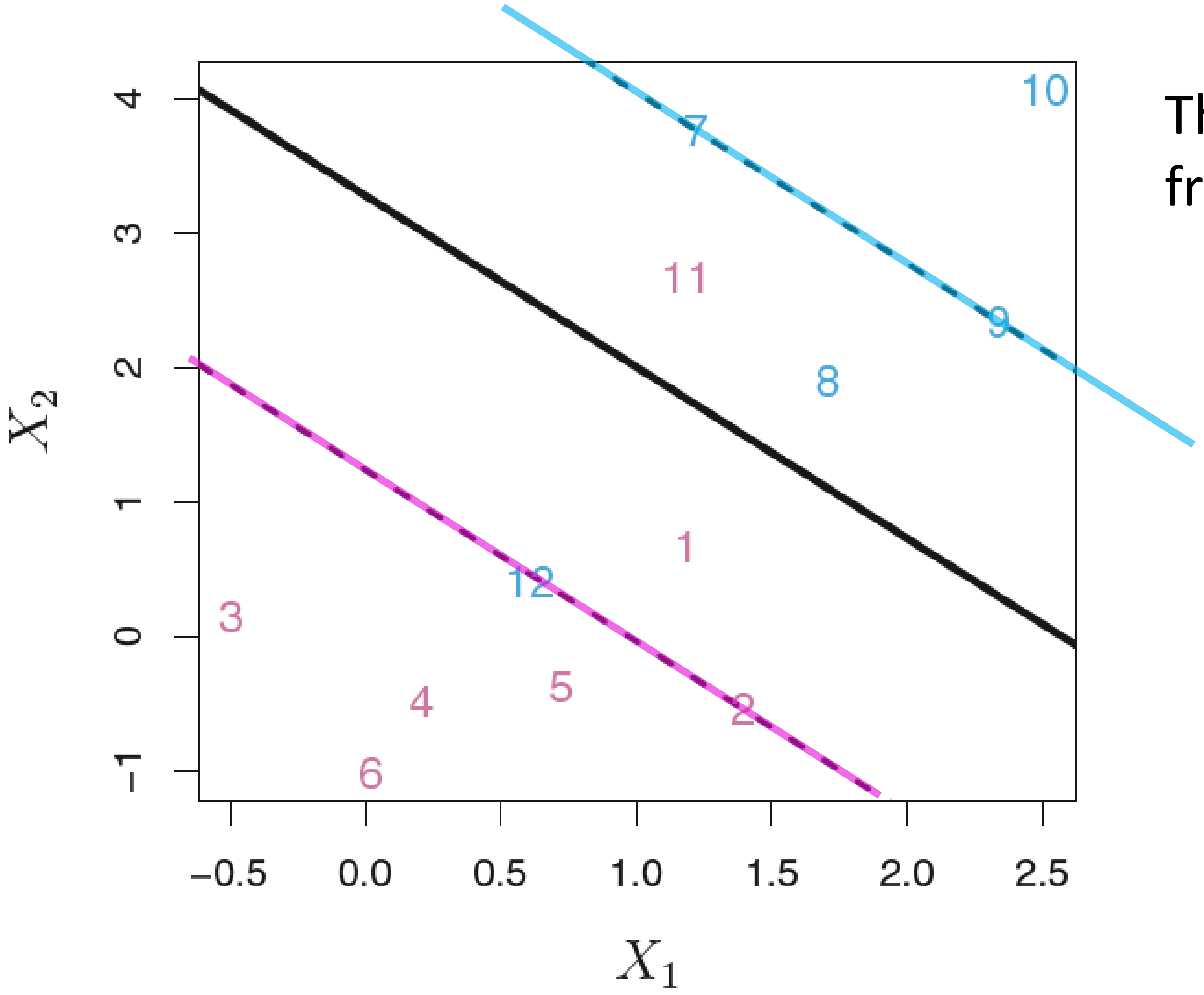
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

$$\epsilon_i \geq 0$$

$$\sum_{i=1}^{n} \epsilon_i \leq C$$

Soft margin classifier

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \qquad \epsilon_i \geq 0$$



The slack is measured conservatively: from the correct side margin

The role of C parameter

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

$$\epsilon_i \geq 0$$

$$\sum_{i=1}^{n} \epsilon_i \leq C$$

C bounds both number and severity of violations

C is an error budget

C is a hyperparameter

How SVM actually finds a solution?

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

$$\epsilon_i \geq 0$$

$$\sum_{i=1}^{n} \epsilon_i \leq C$$

Why SVM called non-parameteric when there are coefficients?

In fact, it doesn't find the coefficients directly

It computes inner product between observations

How SVM actually finds a solution?

It computes inner product between observations

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

The original function $f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$ can be rewritten to

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$$

(Caution: SVM needs inputs normalized)

We need $n(n-1)/2$ inner products to calculate

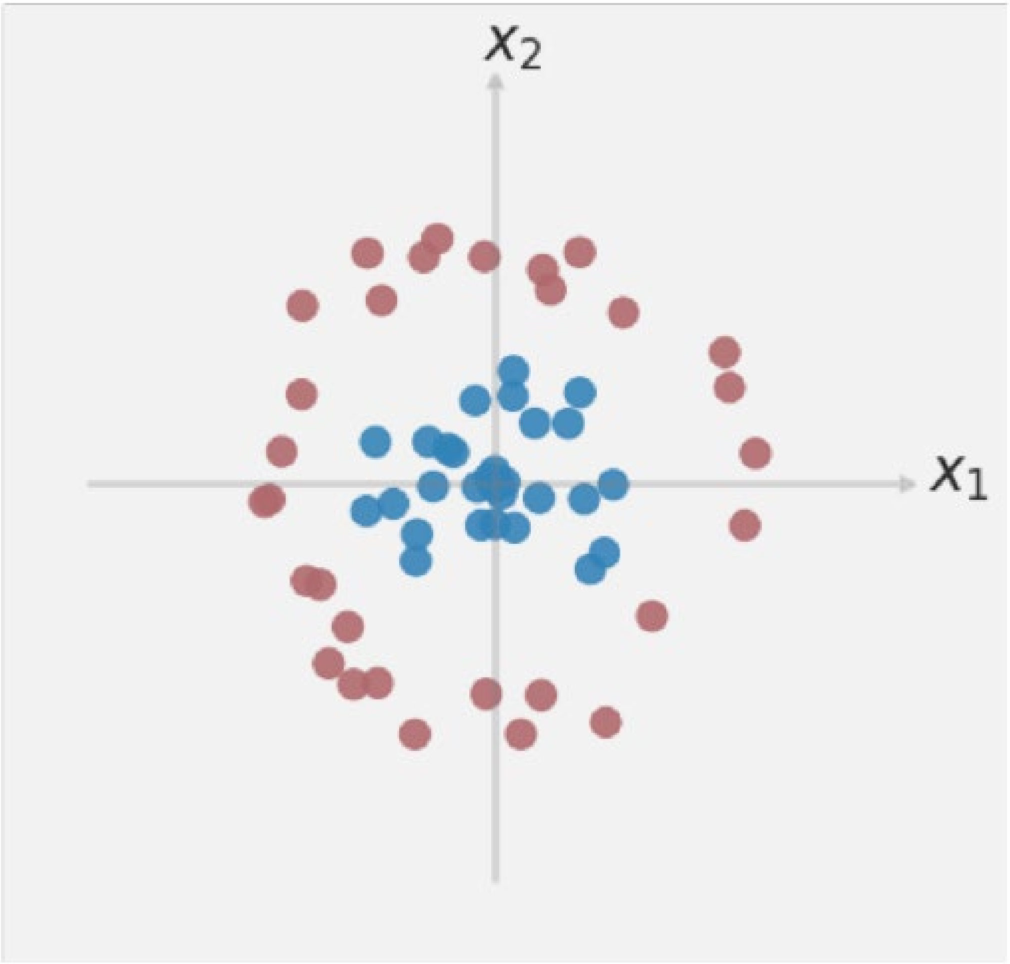
How SVM actually finds a solution?

Actually, we don't need them all.

Only the support vectors have non-zero coefficients

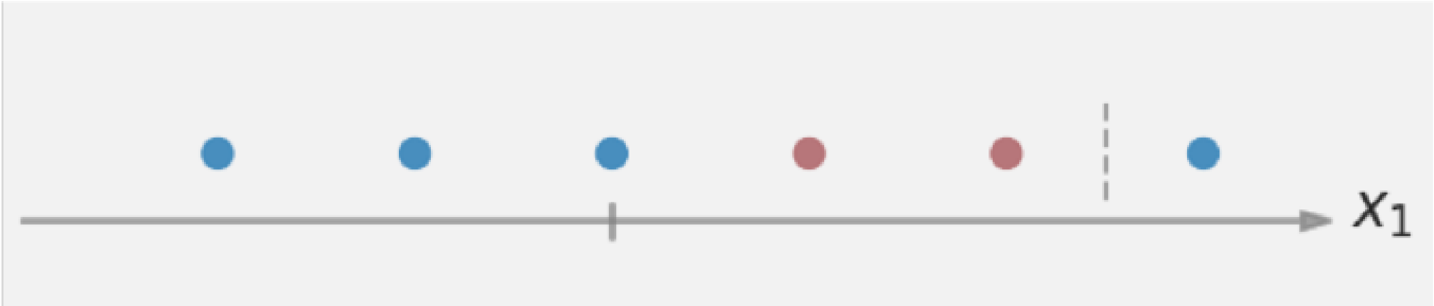
$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_i \rangle$$

What about this data?



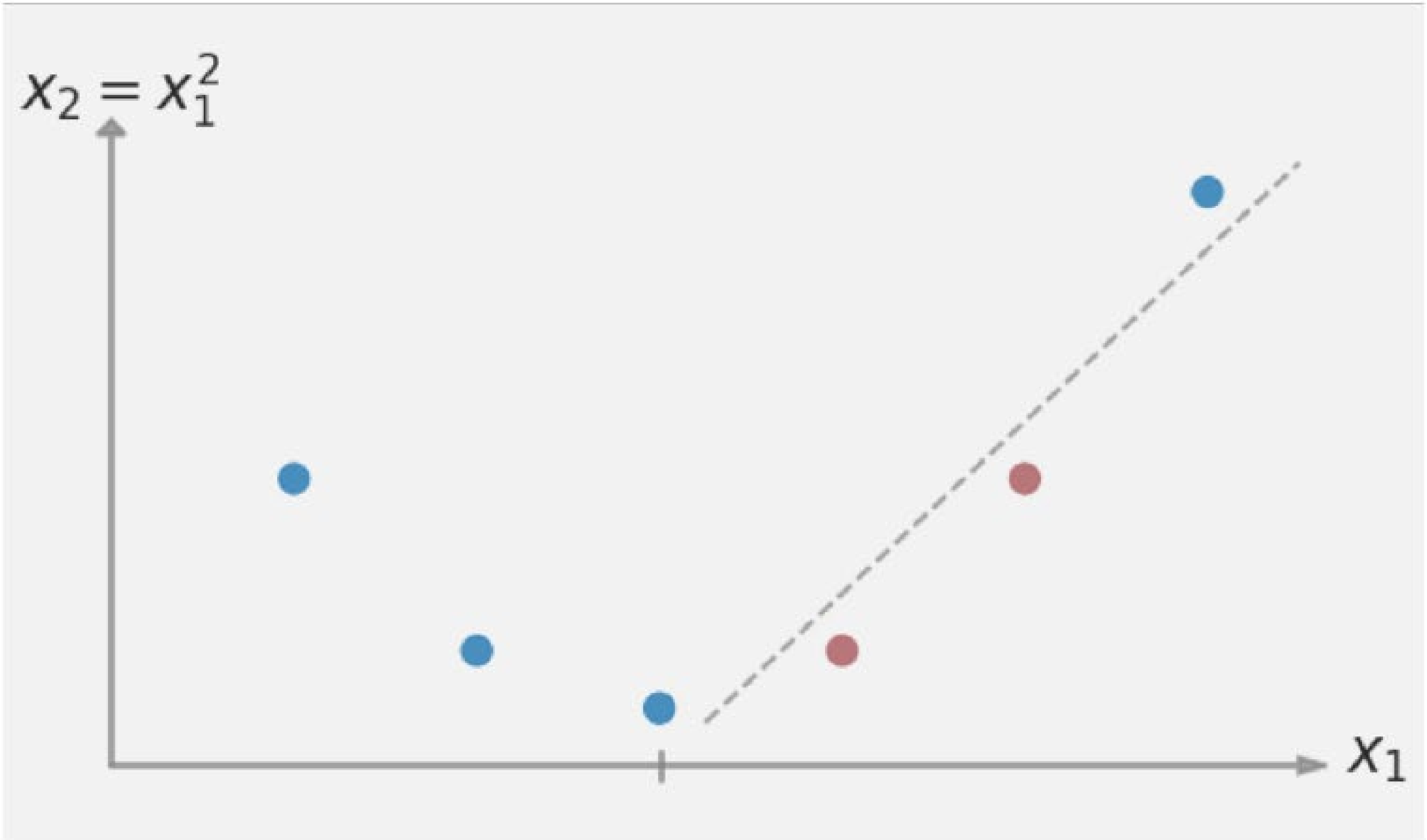
What about this data?

What can we do if the data is clearly not linearly separable?



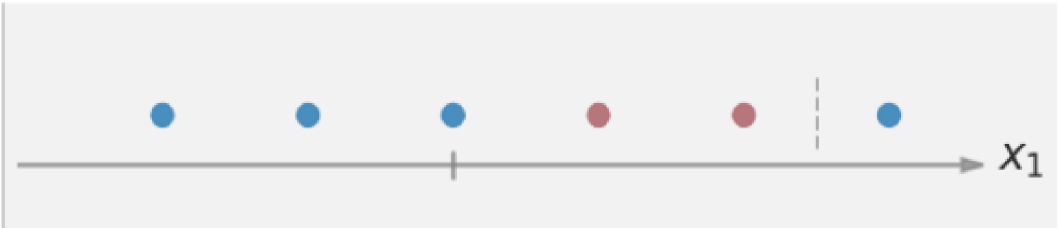
What about this data?

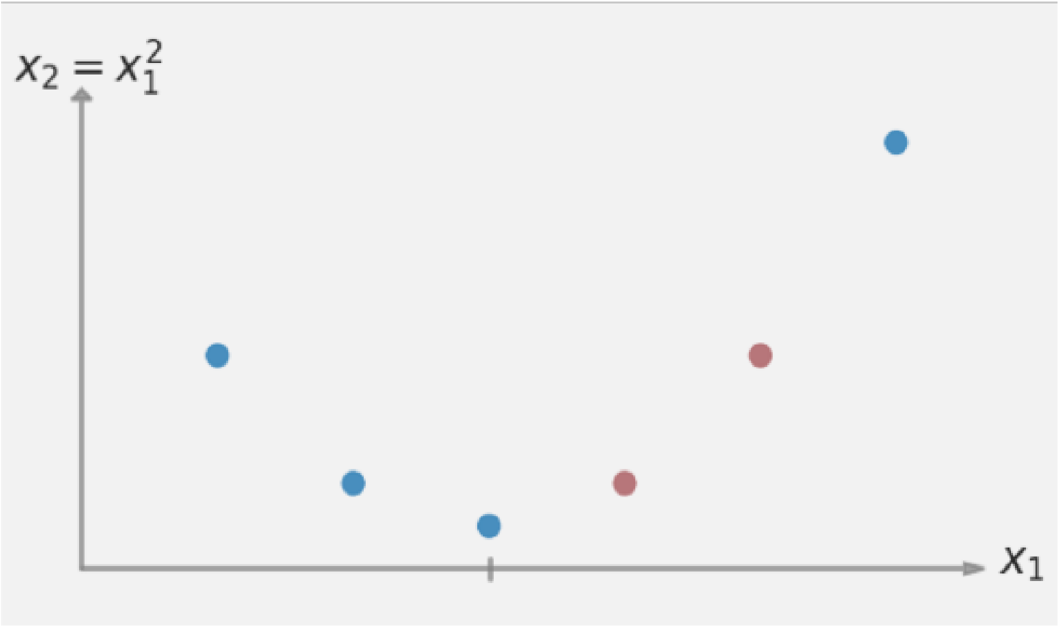
Add a dimension.



What about this data?

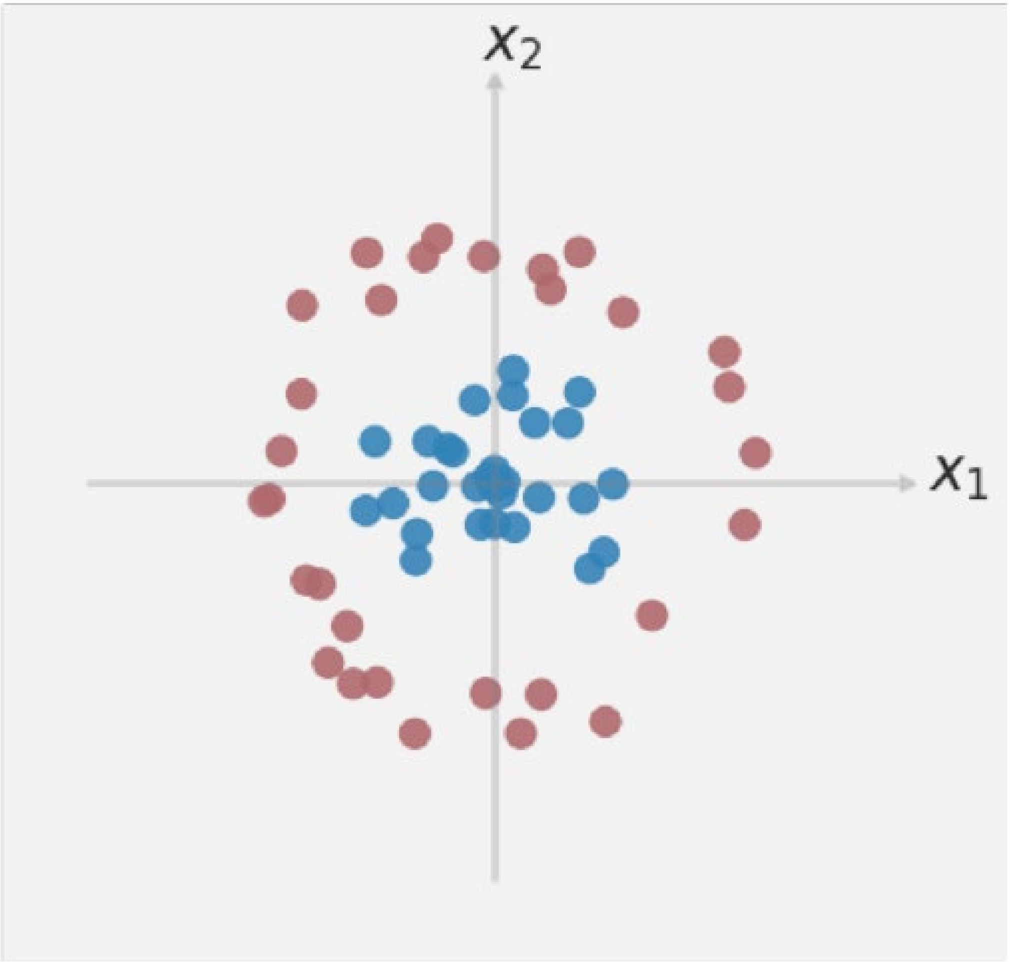
We started with the original feature vector, $\boldsymbol{x} = (x_1)$,
and we created a new derived feature vector, $\phi(\boldsymbol{x}) = (x_1, x_1^2)$.



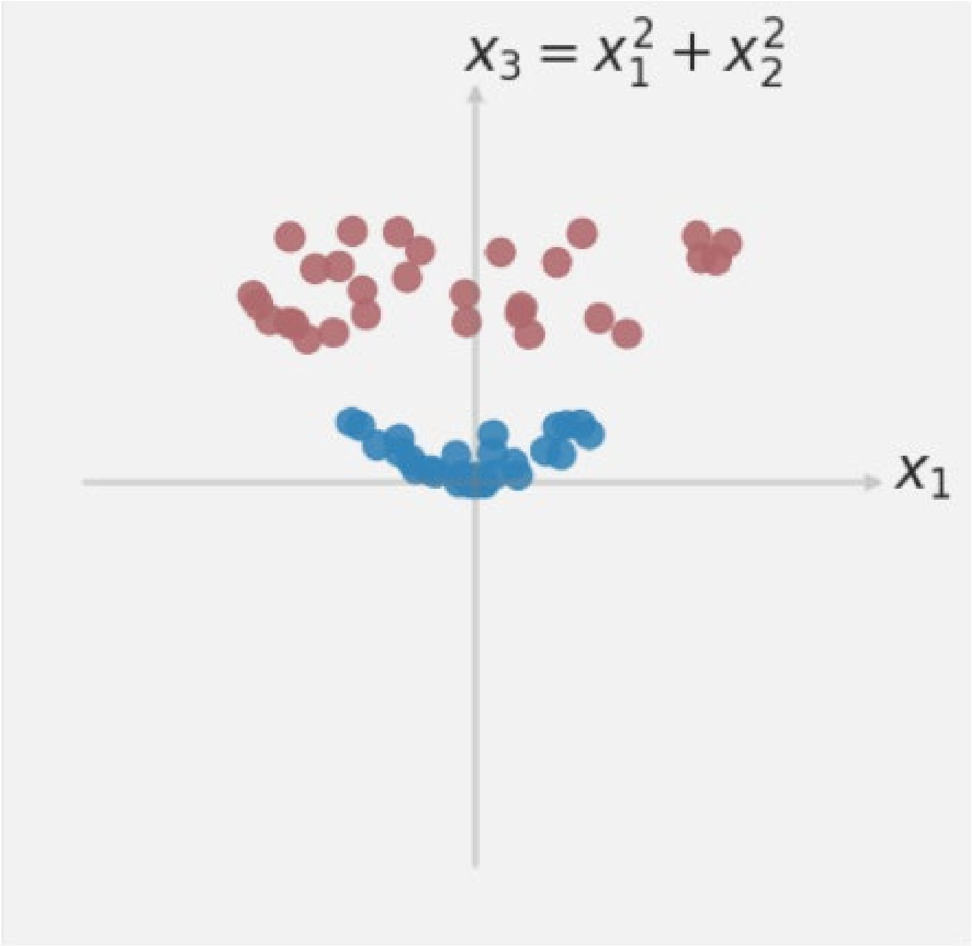


What about this data?

Not linearly separable in 2D



We can separate in 3D



What about this data?

$X_1, X_2, \ldots, X_p$ Great! I can add higher order terms…

$X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2$ But….

$$\underset{\beta_0,\beta_{11},\beta_{12}\ldots,\beta_{p1},\beta_{p2},\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} \quad M$$

$$\text{subject to } y_i\left(\beta_0+\sum_{j=1}^{p}\beta_{j1}x_{ij}+\sum_{j=1}^{p}\beta_{j2}x_{ij}^2\right)\geq M(1-\epsilon_i)$$

$$\sum_{i=1}^{n}\epsilon_i\leq C,\quad \epsilon_i\geq 0,\quad \sum_{j=1}^{p}\sum_{k=1}^{2}\beta_{jk}^2=1.$$

The Kernel trick

Let's generalize this function (the inner product)

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

to a kernel $K(x_i, x_{i'})$

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

Then, we get

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i)$$

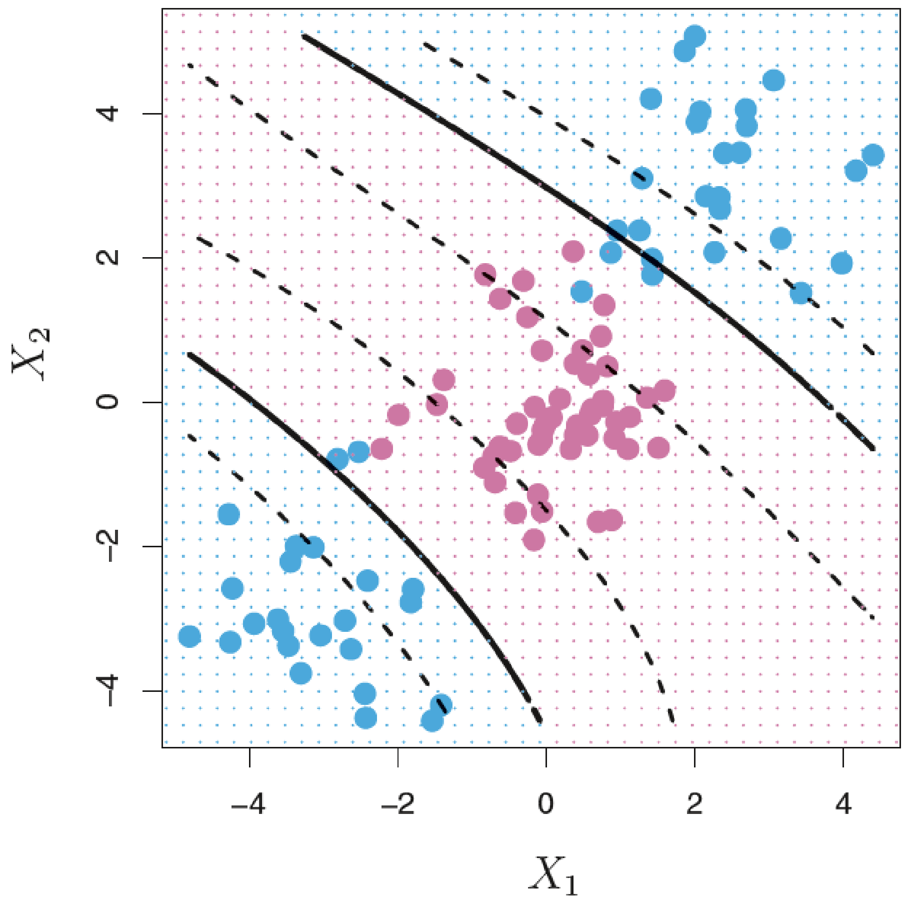
The Kernel trick

Non-linear kernels can take care of non-linear decision boundary

Polynomial kernel

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

d=3

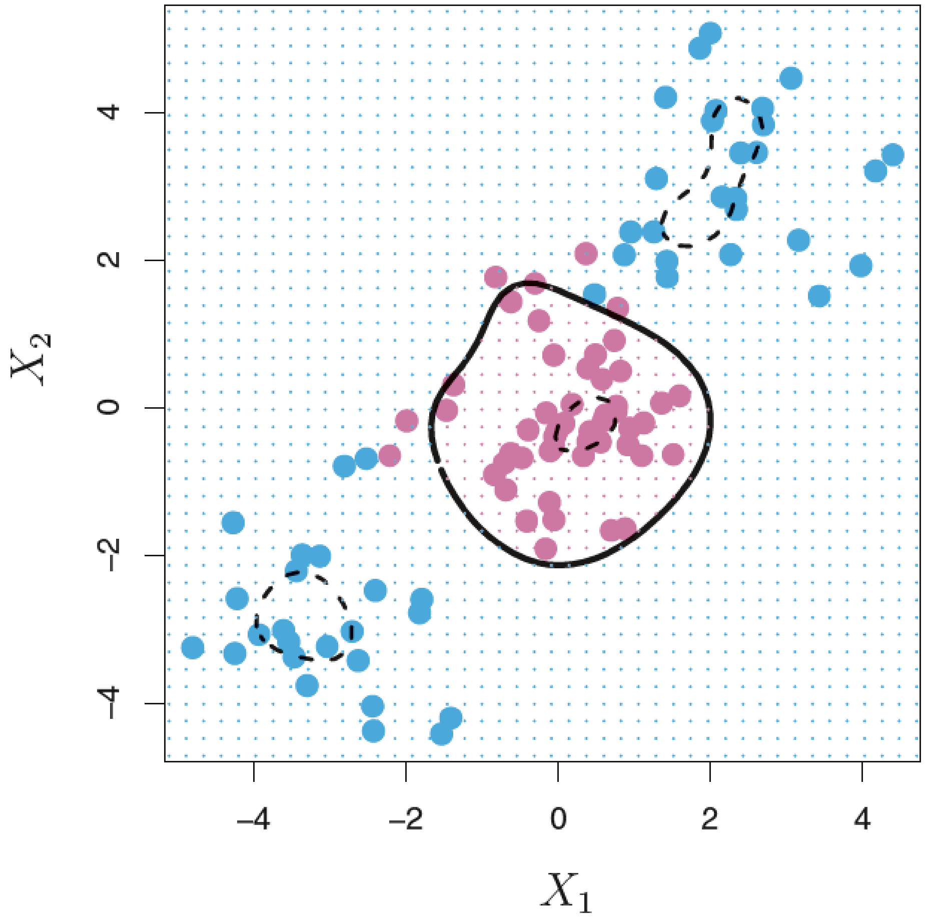


The Kernel trick

Non-linear kernels can take care of non-linear decision boundary

Radial kernel

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$



Hinge Loss

Another way to formulate the same problem:

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{minimize}} \left\{ \sum_{i=1}^{n} \max\left[0, 1 - y_i f(x_i)\right] + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

When to use which model?

For Binary classification

Logistic regression

SVM