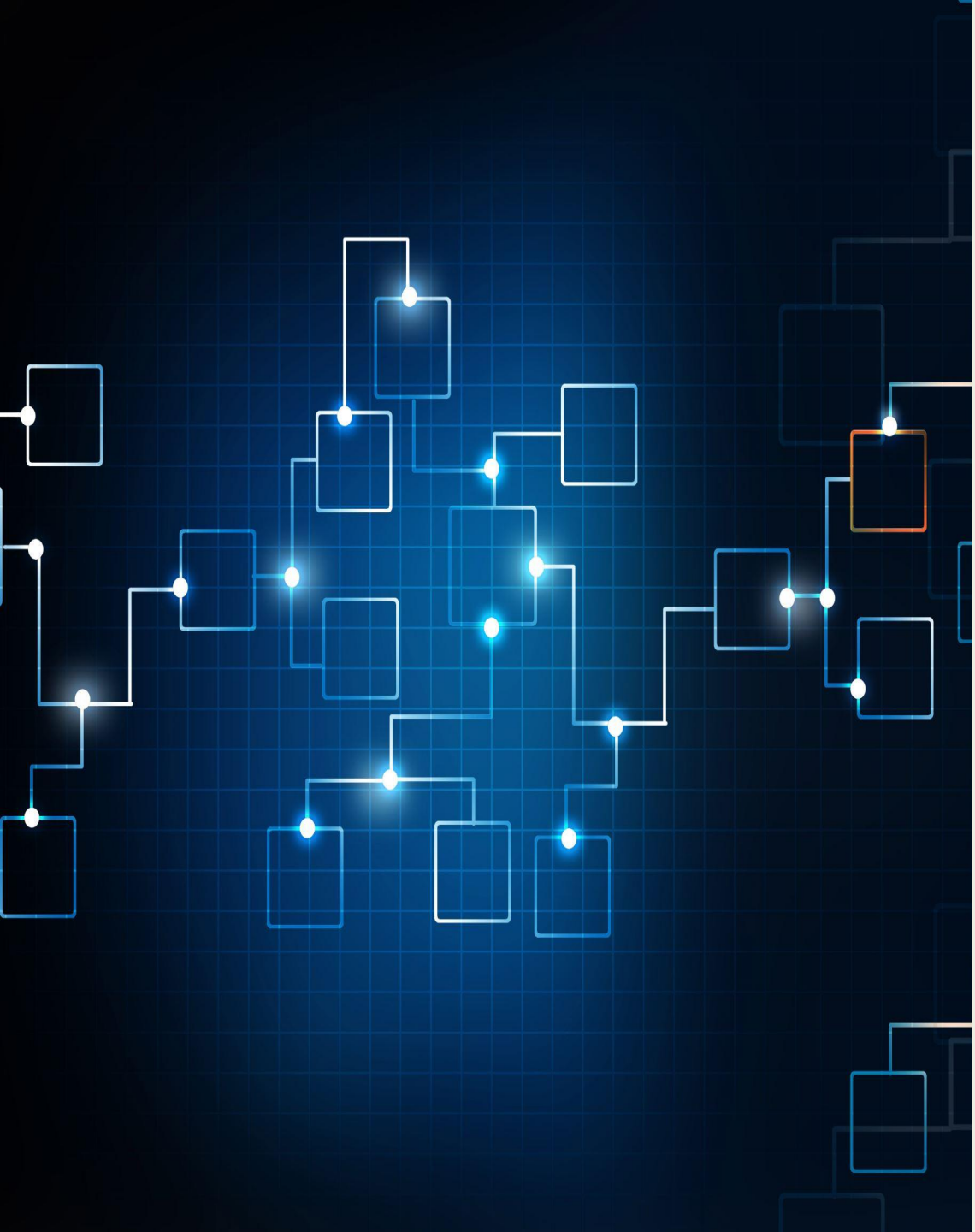


HIT391

MACHINE LEARNING: ADVANCEMENTS AND APPLICATIONS

CHARLES DARWIN UNIVERSITY
AUSTRALIA

Week 7:
Support Vector Machines

- **Learning Outcomes**
  - Hyperplane
  - Maximal Margin Classifier
  - Support Vector Classifier
  - Feature Expansion

Support Vector Machines

- Here we approach the two-class classification problem in a direct way:
  - We try and find a plane that separates the classes in feature space.
- **What if we can't linearly separate the data?**

  When a simple line (or hyperplane) can't separate the data, we get creative in two main ways:

  1. Support Vector Classifiers (Soft Margin)
  - Instead of requiring a perfect separation, we allow some misclassifications.
  - This gives us a "soft margin"—a more flexible boundary that can handle overlap or noise in the data.

  2. Data Transformations (Feature Expansion)
  - We transform the data into a new space where separation is possible.
  - This is like lifting the data into a higher-dimensional space where complex patterns become linear.

Hyperplane

- A hyperplane in p dimensions is a flat affine subspace of dimension p − 1.
- In general the equation for a hyperplane in p-dimensinoal space has the form:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

  - when the dimension $p = 2$, the hyperplane is a line (1D).
  - when $\beta_0 = 0$, the hyperplane goes through the origin, otherwise not.
  - The vector $\beta = (\beta_1, \beta_2, \cdots, \beta_p)$ is called the normal vector — it points in a direction orthogonal to the surface of a hyperplane.

An example (p=2)

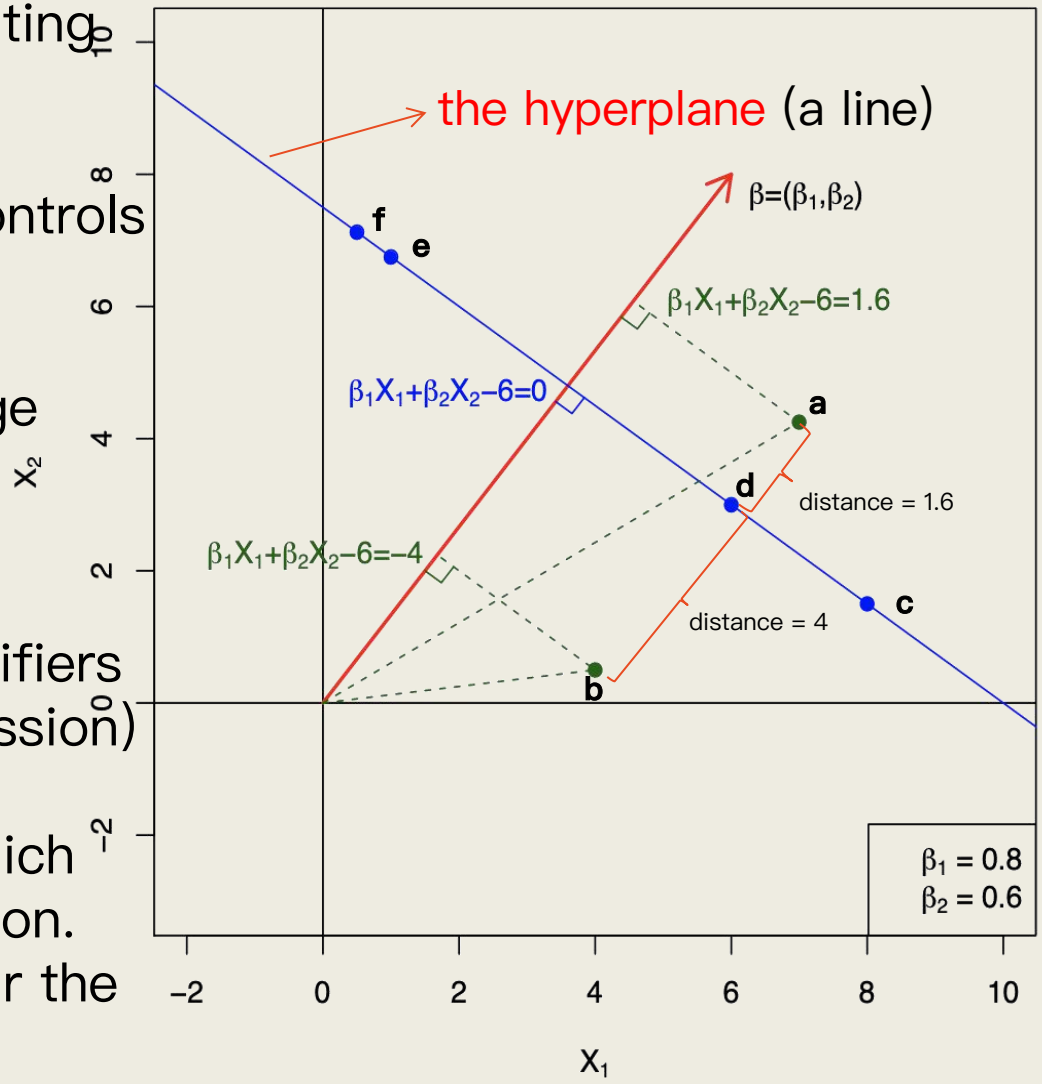
1. The hyperplane(blue) – separating the space.

2. **The normal vector β (red)** –controls orientation.

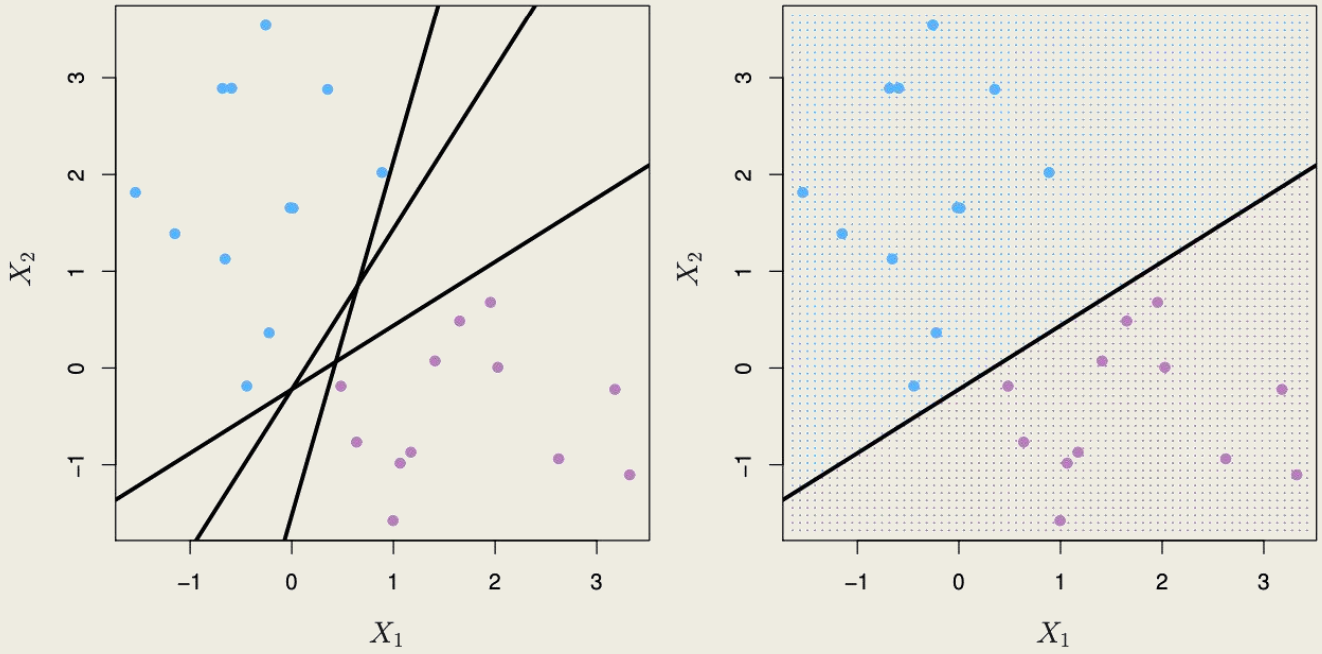
3. Projections – green and orange dashed lines.

4. **Connect to classification:**
– This is exactly how linear classifiers work (like SVMs or logistic regression)

–The sign of $\beta^T X - 6$ tells you which side of the hyperplane a point is on.
–The magnitude tells you how far the point is from the boundary.



Separating Hyperplanes



If $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$, then $f(X) > 0$ for points on one side of the hyperplane, and $f(X) < 0$ for points on the other.

If we code the colored points as $Y_i = +1$ for blue, say, and $Y_i = -1$ for mauve, then if $Y_i \cdot f(X_i) > 0$ for all i, $f(X) = 0$ defines a *separating hyperplane.*

Maximal Margin Classifier

- Among all separating hyperplanes, find the one that makes the biggest gap or margin between the two classes.
- Model formulation:

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

objective function

Constraints

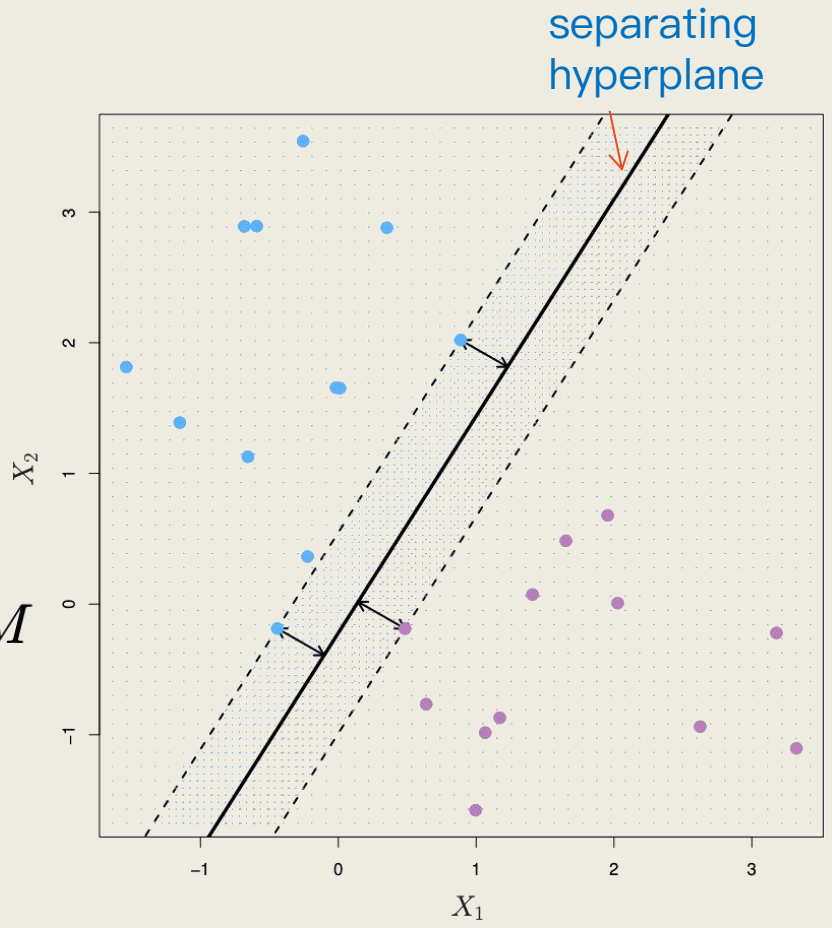
$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq M$$

$$\text{for all } i = 1, \ldots, N.$$

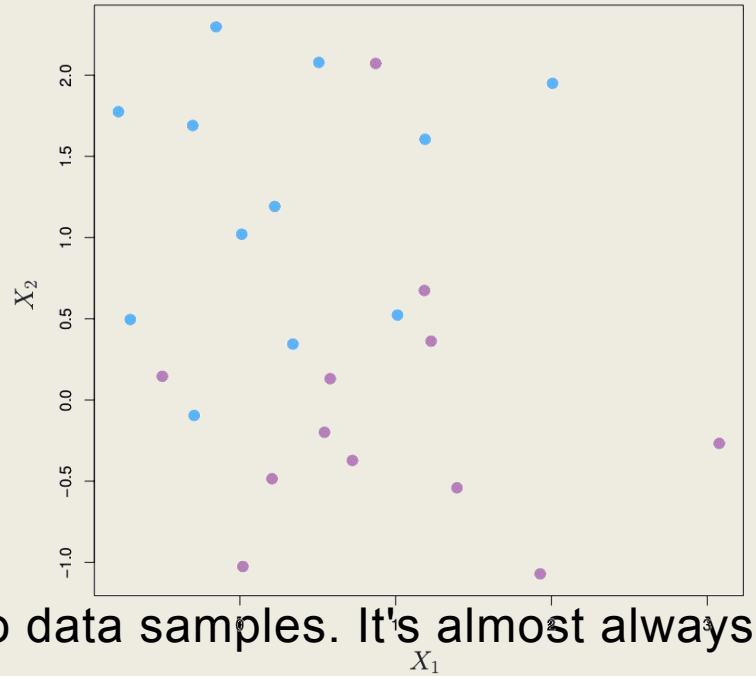
Inputs $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$

Outputs $\beta_0, \cdots, \beta_p$



Non–separable Data

- The data on the left are not separable by a linear boundary.
- This is often the case,
  - unless the number of data samples N is less than the dimension of inputs p, N < p



For example:

- In 3-dimension space, only have two data samples. It's almost always possible to find a plane (hyperplane) that separates the two.

- In high-dimensional spaces with few data points, the points are more "spread out." So, it's easier to find a hyperplane that slices between them without hitting any.

Non–separable Data

But in practice:

- We usually have $N \gg p$ (lots of data, few features), so:

- Perfect separation with a linear boundary is rare.

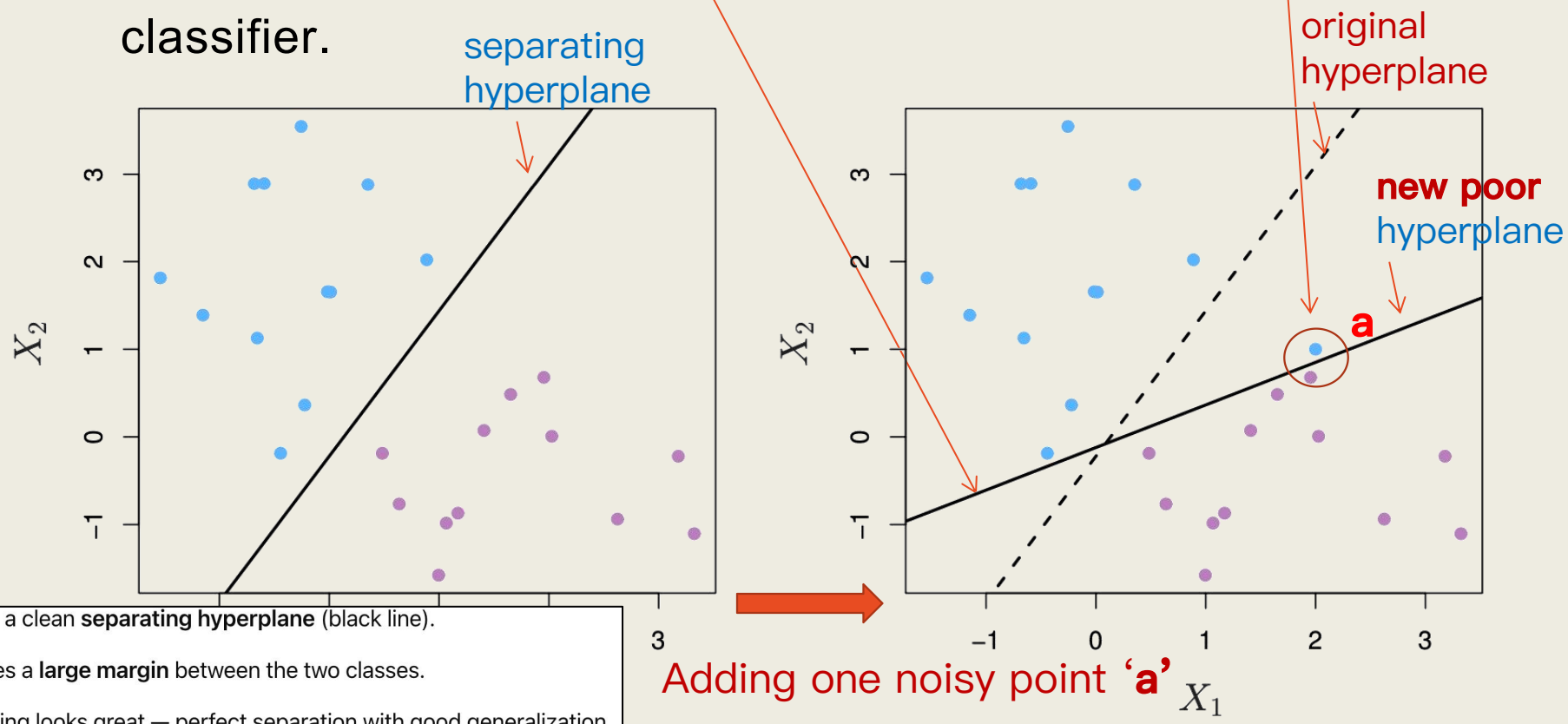
- That's why we use:

- Soft margins (allowing some misclassification)

- Nonlinear transformations (expanding to higher-dimensional spaces)

Noisy Data

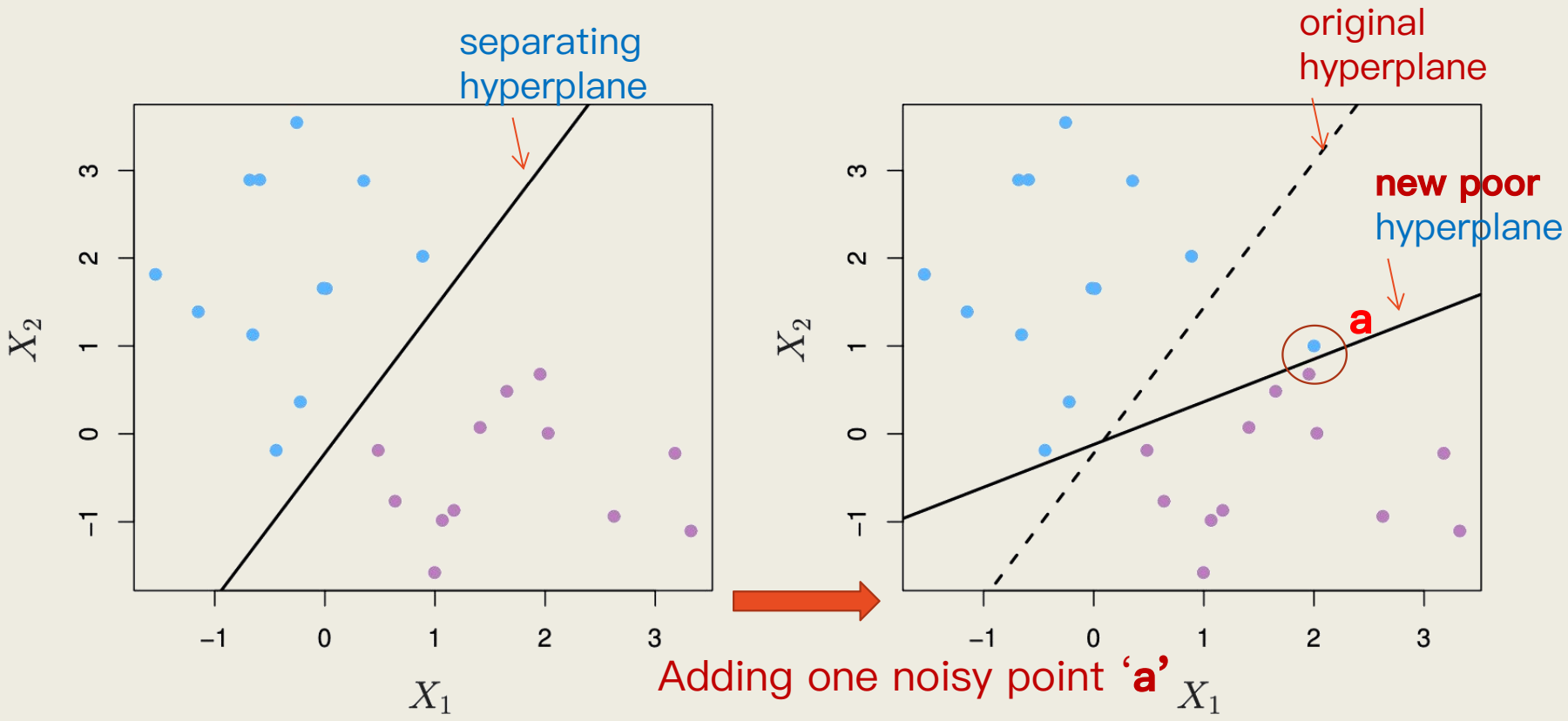
- Sometimes the data are separable, but have noisy samples.
  - This can lead to a **poor solution** for the maximal-margin classifier.



There's a clean **separating hyperplane** (black line).

It creates a **large margin** between the two classes.

Everything looks great — perfect separation with good generalization.



- The new hyperplane (solid line in the right figure) is pulled down and tilted — it now has **a smaller margin**.
- The original hyperplane (dashed line) would've been better if we could just ignore the noisy point.
- The model is now **overfitting to the noise** — giving a poorer solution overall.

Soft Margin

- Just because data are separable, doesn't mean a **hard-margin classifier** will find a good boundary.

- **Noisy samples** can lead to overfitting and poor generalization.

- This motivates the use of **soft margins** — where we allow a few misclassifications in exchange for a better overall boundary.

Support Vector Classifier

- A Support Vector Classifier allows some misclassifications or margin violations (ϵ) by maxmizing a **soft margin**, so it can find a better overall boundary in noisy or non-perfectly-separable data

- Model formulation:

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \; M \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

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

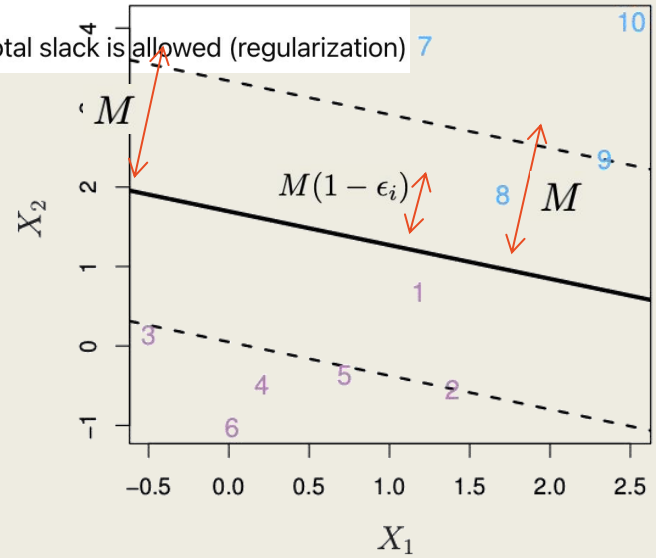
$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

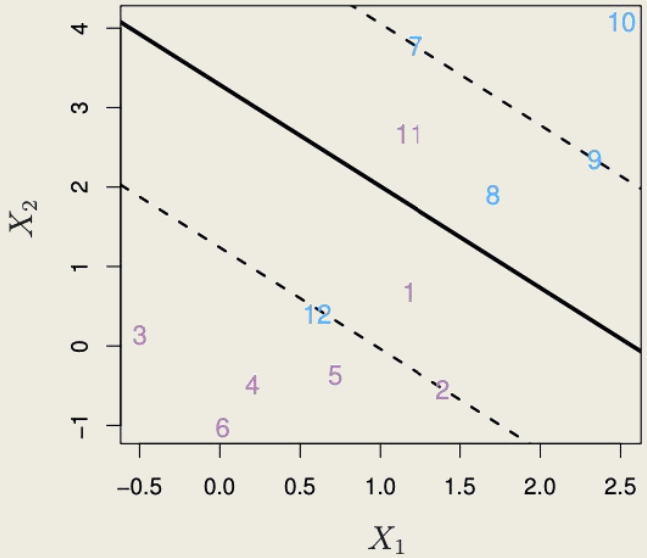
M: the **margin** (distance between support vectors)

ϵ_i: how much the i-th point **violates the margin**

- $\epsilon_i = 0$: perfectly classified, outside the margin
- $0 < \epsilon_i < 1$: inside the margin, but on the correct side
- $\epsilon_i > 1$: misclassified

C: controls how much total slack is allowed (regularization)





Hard Margin vs. Soft Margin

- Hard margins require perfect separation — very sensitive to noise.
- Soft margins (Support Vector Classifier) balance:
  - Maximizing margin
  - Minimizing total violations
- Controlled by slack variables ϵ_i and regularization parameter C

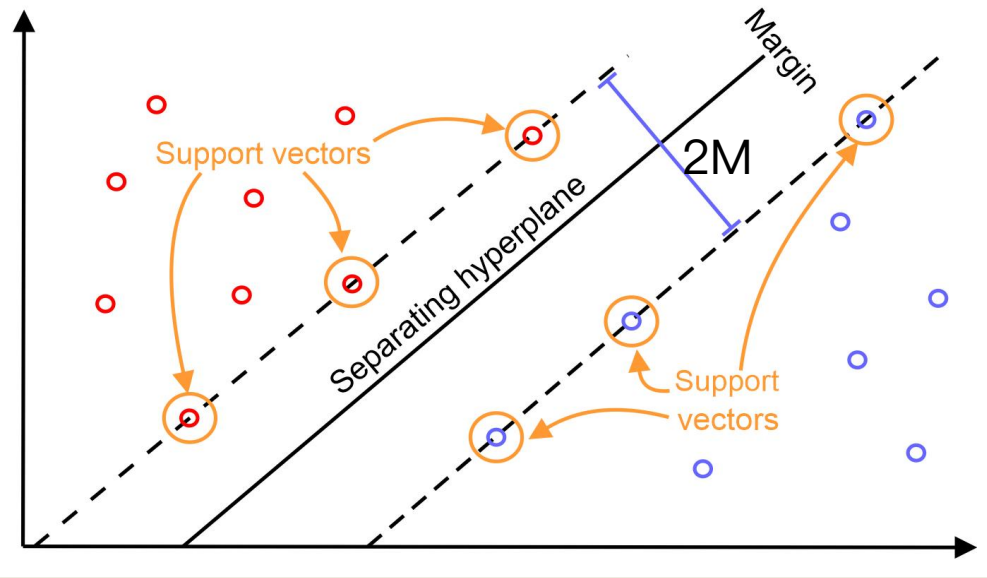
Support Vector Classifier

- Support vectors: vectors on two dashed lines

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

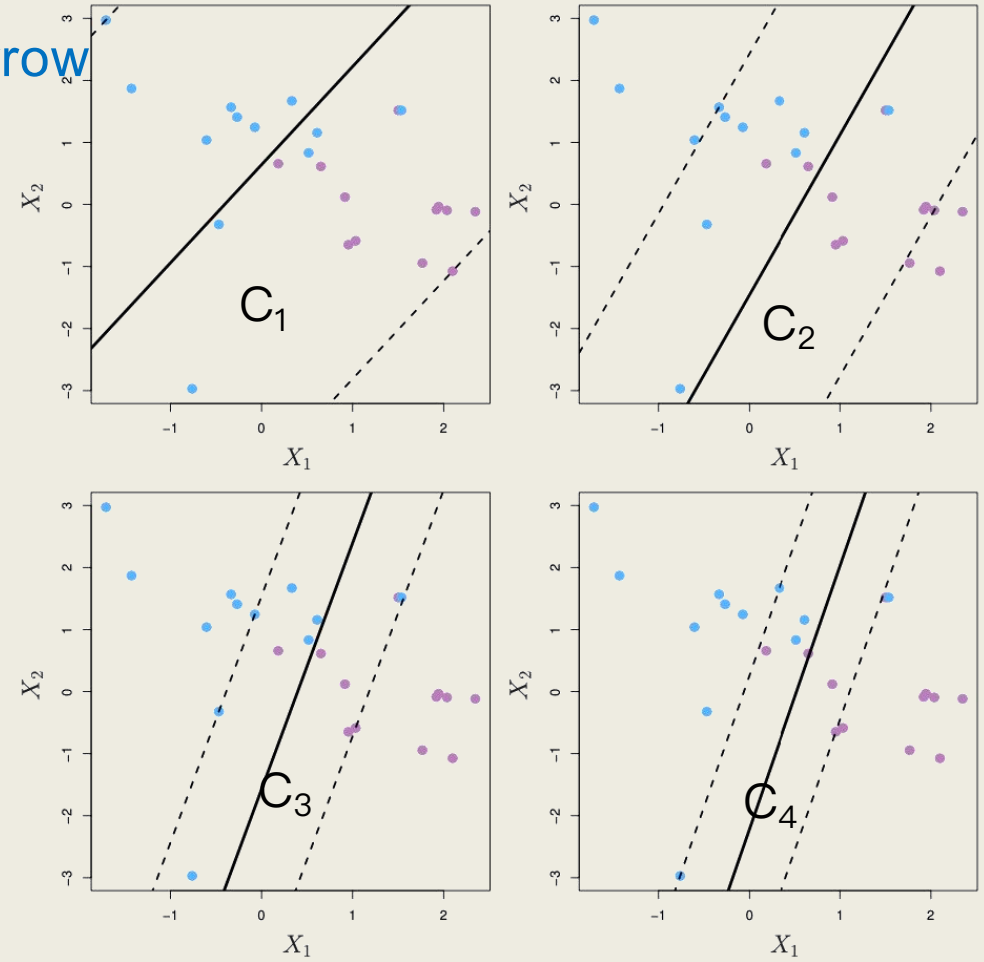
- Separating hyperplane (linear boundary):

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



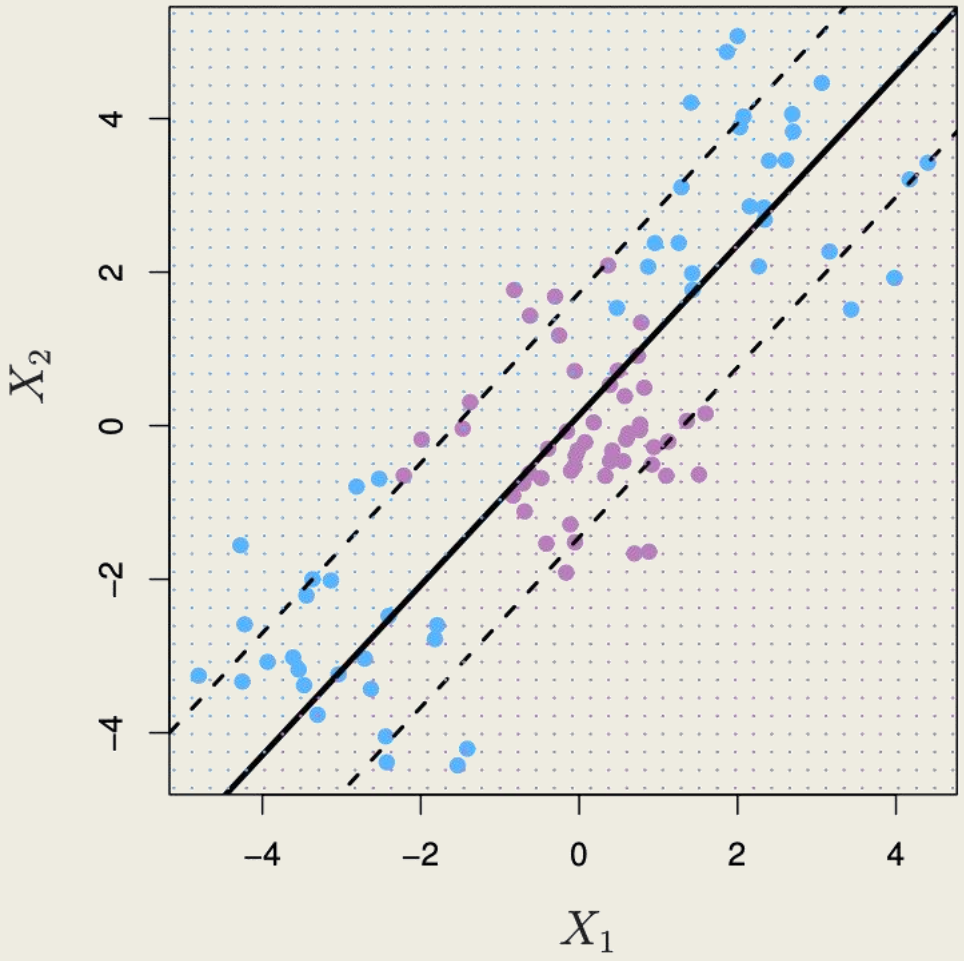
Hyper–parameter C

- where C is a regularization hyper-parameter
  - tune(choose) the optimal C by grid search
    - a small C results in narrow margins
    - as C increases the margins widen allowing more violations



Limitation

- Sometime a linear boundary simply won't work, no matter what value of C.
  - What shoud we do?
    - Feature Mapping



Data Transformation – Feature Expansion

- Enlarge the space of features:
  - feature expansion: $X_1^2, X_1^3, X_1X_2, X_1X_2^2, \ldots$
  - **p**-dimensional space is enlarged to a **q**-dimensional space, where **q > p**.
- Fit a support-vector classifier in the enlarged **q**-dimensional space.
- This results in non-linear decision boundaries in the original space.

Example: Suppose we use $(X_1,\ X_2,\ X_1^2,\ X_2^2,\ X_1X_2)$ instead of just (X_1, X_2). Then the decision boundary would be of the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 = 0$$

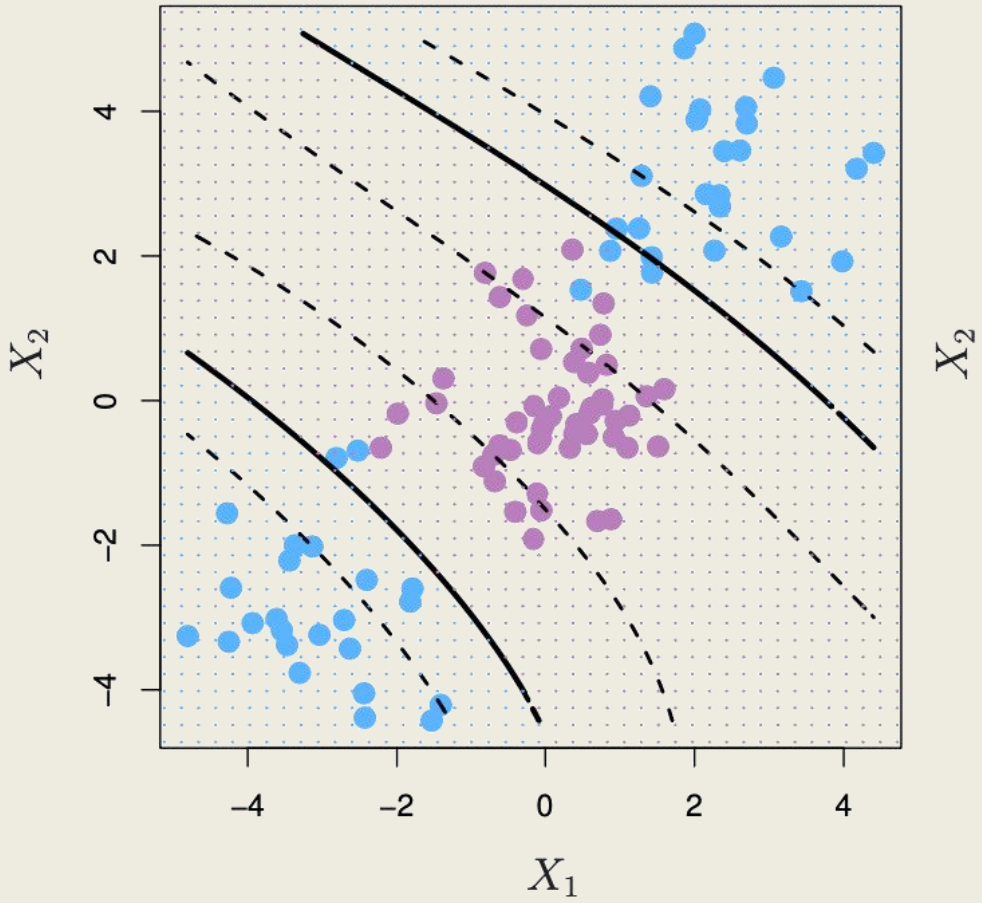
This leads to nonlinear decision boundaries in the original space

Cubic Polynomials

Here we use a basis expansion of cubic polynomials

From 2 variables to 9

The support-vector classifier in the enlarged space solves the problem in the lower-dimensional space



$$\beta_0+\beta_1X_1+\beta_2X_2+\beta_3X_1^2+\beta_4X_2^2+\beta_5X_1X_2+\beta_6X_1^3+\beta_7X_2^3+\beta_8X_1X_2^2+\beta_9X_1^2X_2=0$$

Extend SVMs handling more than 2 classes?

- Standard SVMs are binary classifiers, they only distinguish between two classes. What do we do when we have K > 2 classes?

OVA One versus All. Fit K different 2-class SVM classifiers $\hat{f}_k(x)$, $k = 1, \ldots, K$; each class versus the rest. Classify x_* to the class for which $\hat{f}_k(x_*)$ is largest.

OVO One versus One. Fit all $\frac{K(K-1)}{2}$ pairwise classifiers $f\ell(x)$. Classify x to the class that wins the most pairwise competitions.

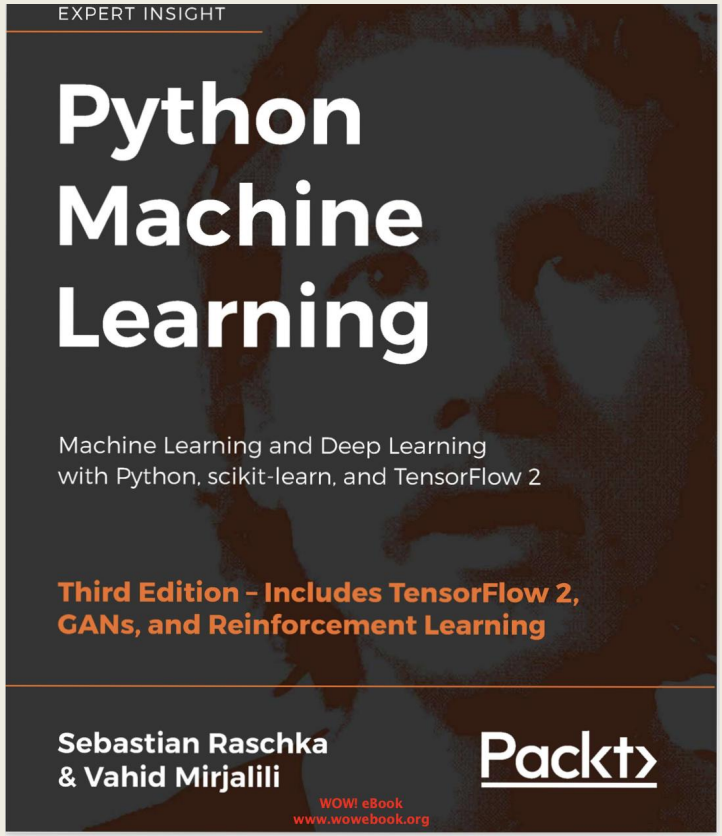
Which to choose? If K is not too large, use OVO.

- OVA is simpler and faster, especially when K is large.
- OVO is more precise, but requires more classifiers and computation.

Summary

- **Support Vector Machines**
  - Hyperplane
  - Maximal Margin Classifier
  - Support Vector Classifier
  - Feature Expansion

References



- Understanding Support Vector Machines In Depth Tutorial
- https://cambiotraining.github.io/intro-machine-learning/svm.html
- Sebastian Raschka, etl., 'Python machine learning', Third Edition