THE UNIVERSITY OF SYDNEY

Lecture 7: More on classification, class imbalance and semi-supervised learning

STAT5003

Pengyi Yang

To cover: the intuition and increasing sophistication from

Maximal Margin Classifier

Support Vector Classifier

Support Vector Machine

Support Vector Machines (SVMs)

Basically idea behind SVM:

We try and find a plane that separates the classes in feature space.

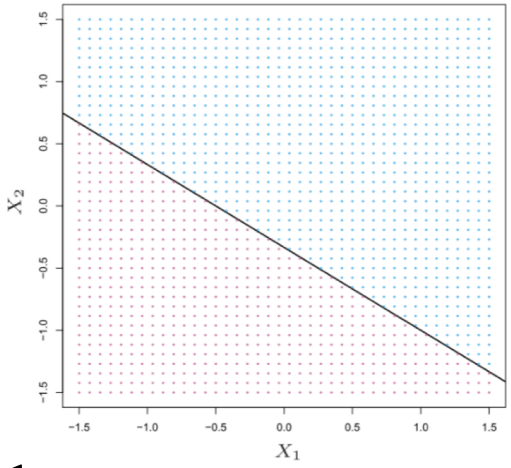
If we cannot, we get creative in two ways:

- We soften what we mean by "separates", and
- We enrich and enlarge the feature space so that separation is possible.

What is a hyperplane?

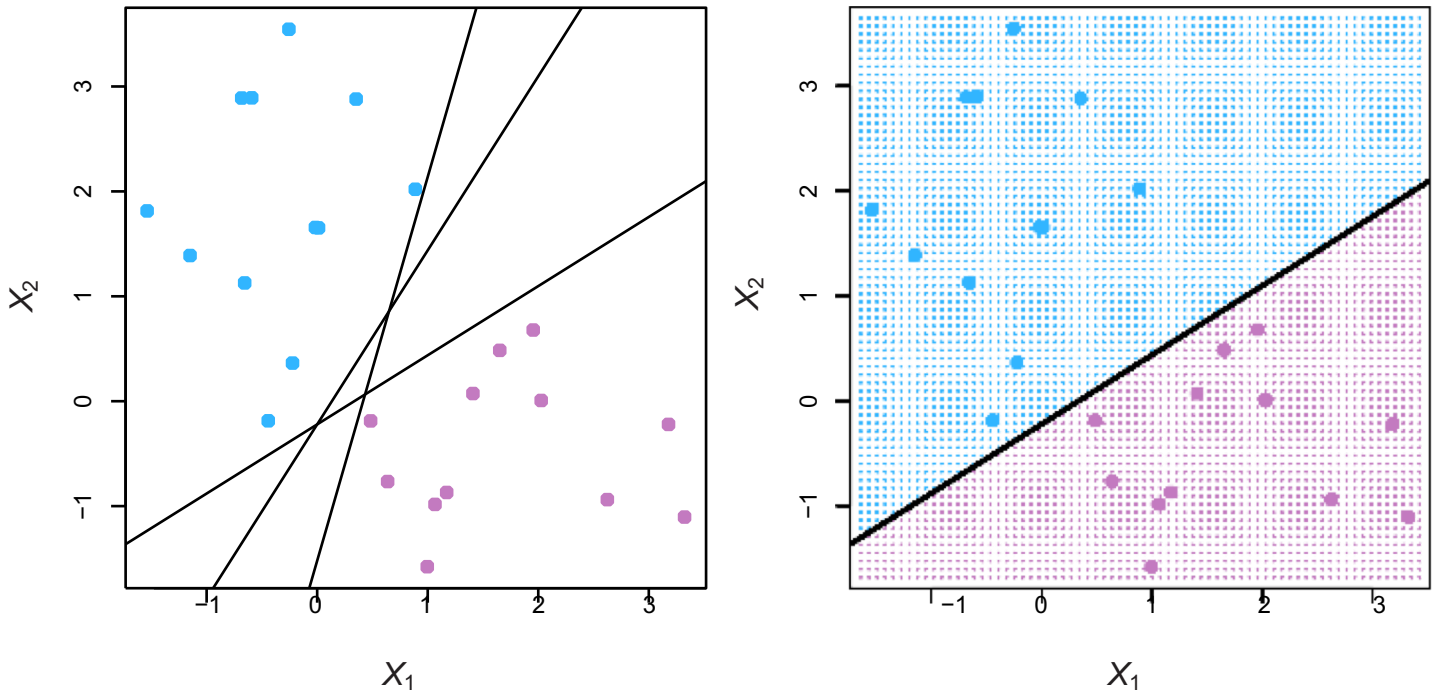
- A hyperplane in p dimensions is a flat affine subspace of dimension $p - 1$.
- In general the equation for a hyperplane has the form

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



- In $p = 2$ dimensions a hyperplane is a line.
- If $\beta_0 = 0$, 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.

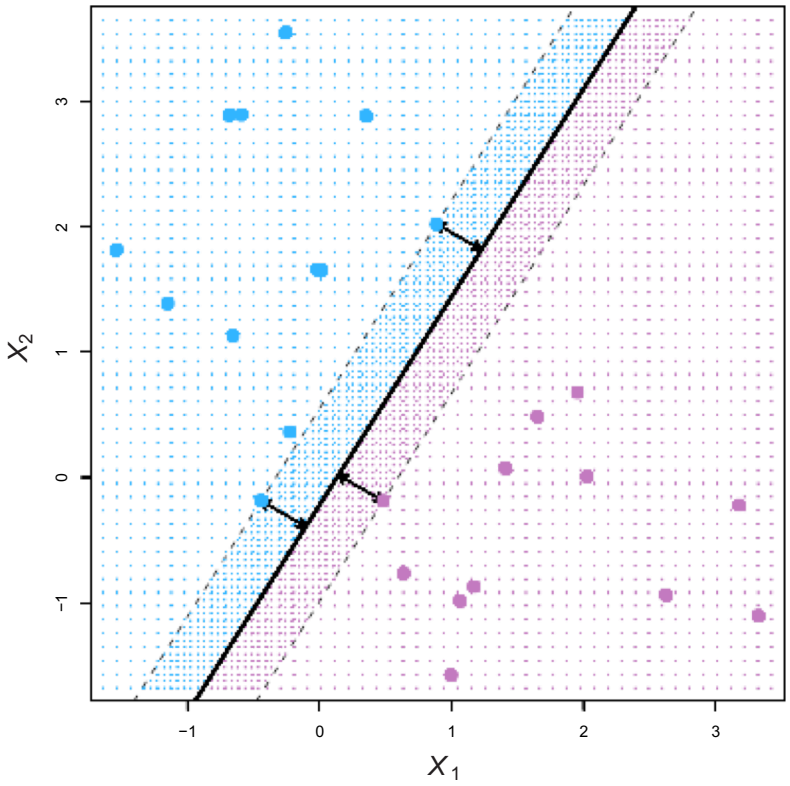
Separating Hyperplanes



- If we code the colored points as $Y_i = +1$ for blue, and $Y_i = -1$ for purple, then if $Y_i \cdot f(X_i)$ is greater than 0 for all i, $f(X) = 0$ defines a *separating hyperplane*.
- If $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$ defines a hyperplane, then $f(X) > 0$ defines for points on one side of the hyperplane, and $f(X) < 0$ defines for points on the other.

Maximal Margin Classifier

Among all separating hyperplanes, find the one that makes the biggest gap or ***margin*** between the two classes.



Maximal Margin Classifier:

Constrained optimisation problem

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

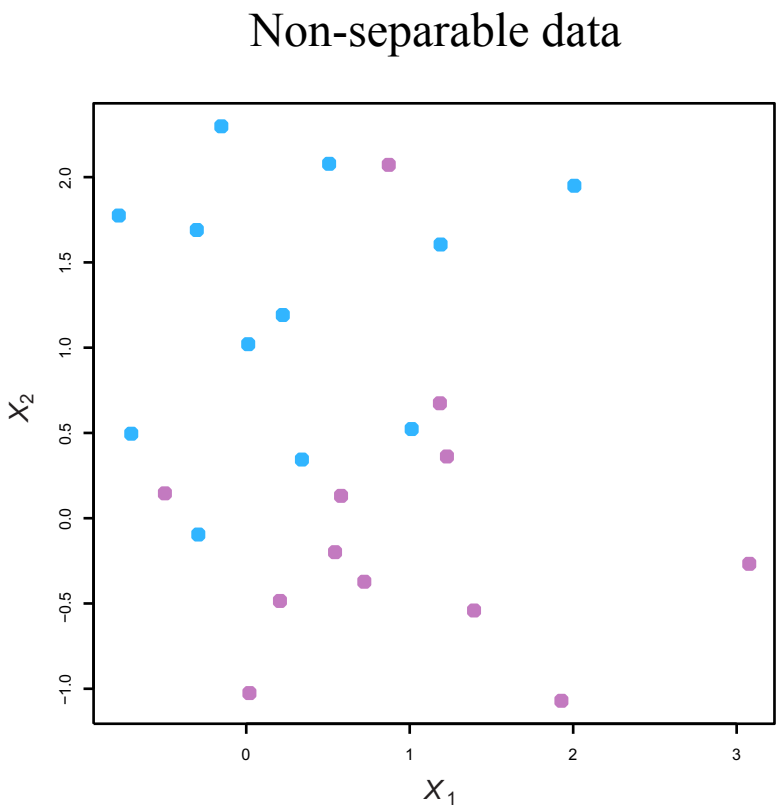
$$\text{subject to} \sum_{j=1}^{p} \beta_j^2 = 1,$$

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

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

Demonstration

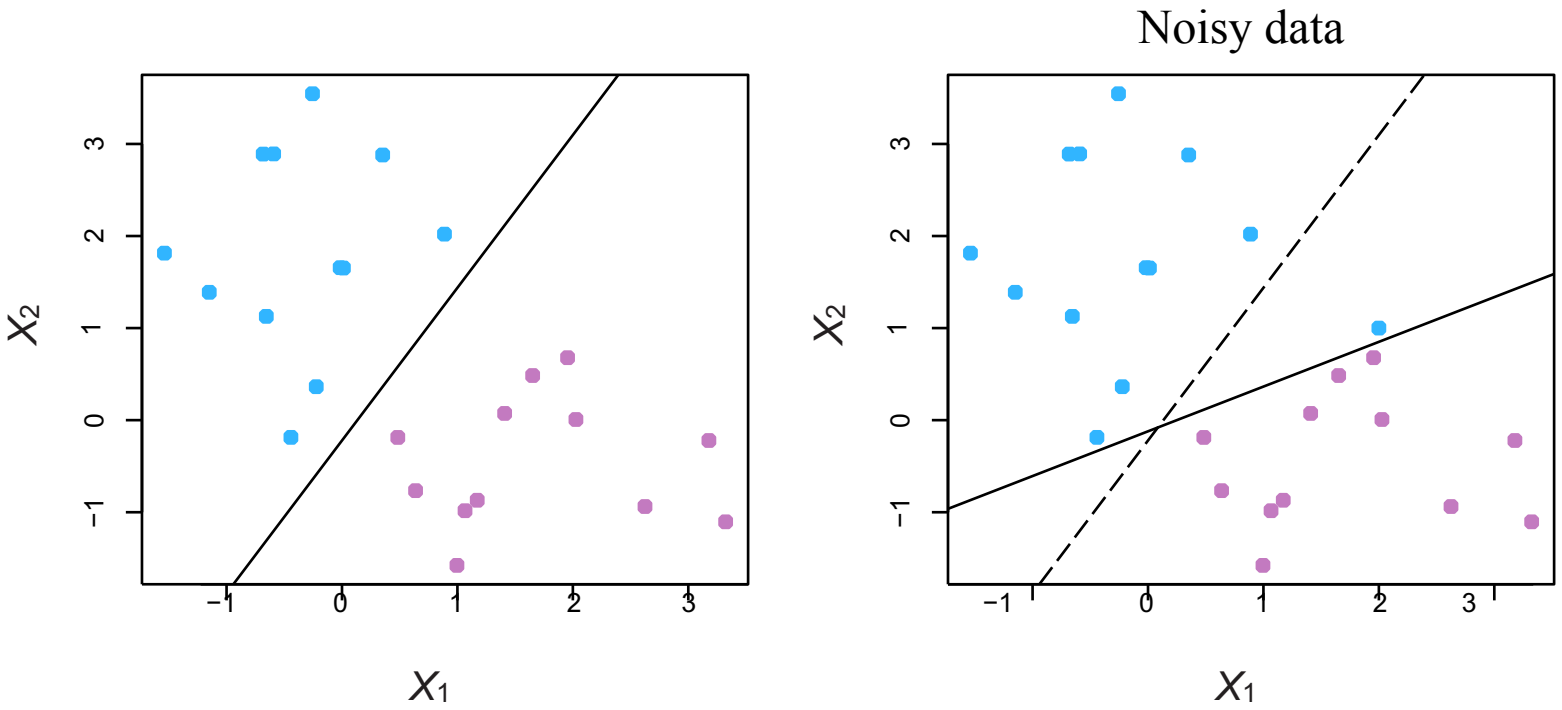
Problem associated with maximal margin classifier



The data on the left are not separable by a linear boundary.

This is often the case, unless $N < P$ (i.e. number of feature is greater than number of samples)

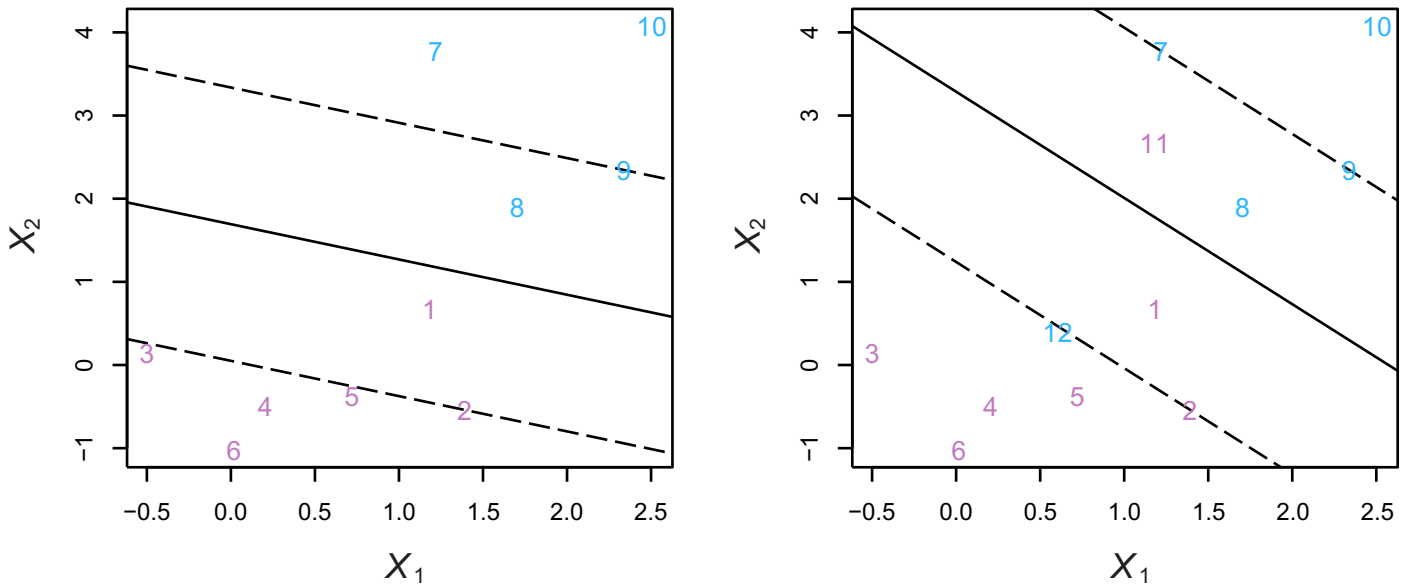
Problem associated with maximal margin classifier



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

The *support vector classifier* maximizes a *soft* margin.

Support Vector Classifier



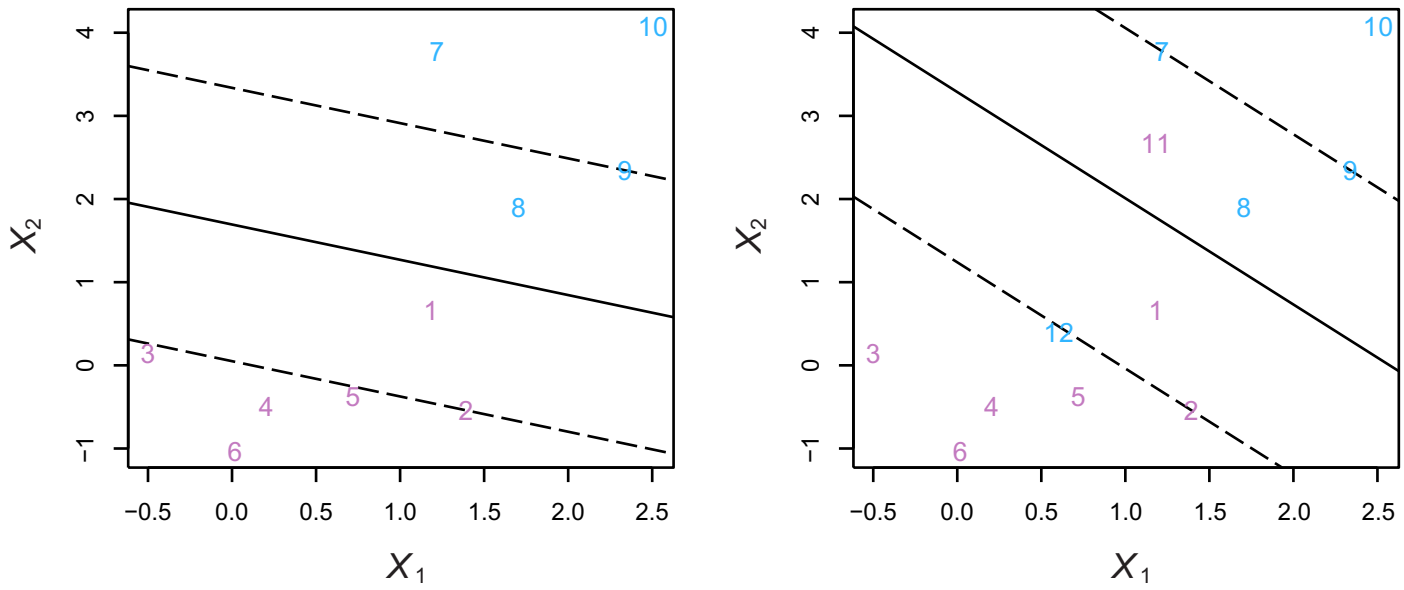
Support Vector Classifier:

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

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

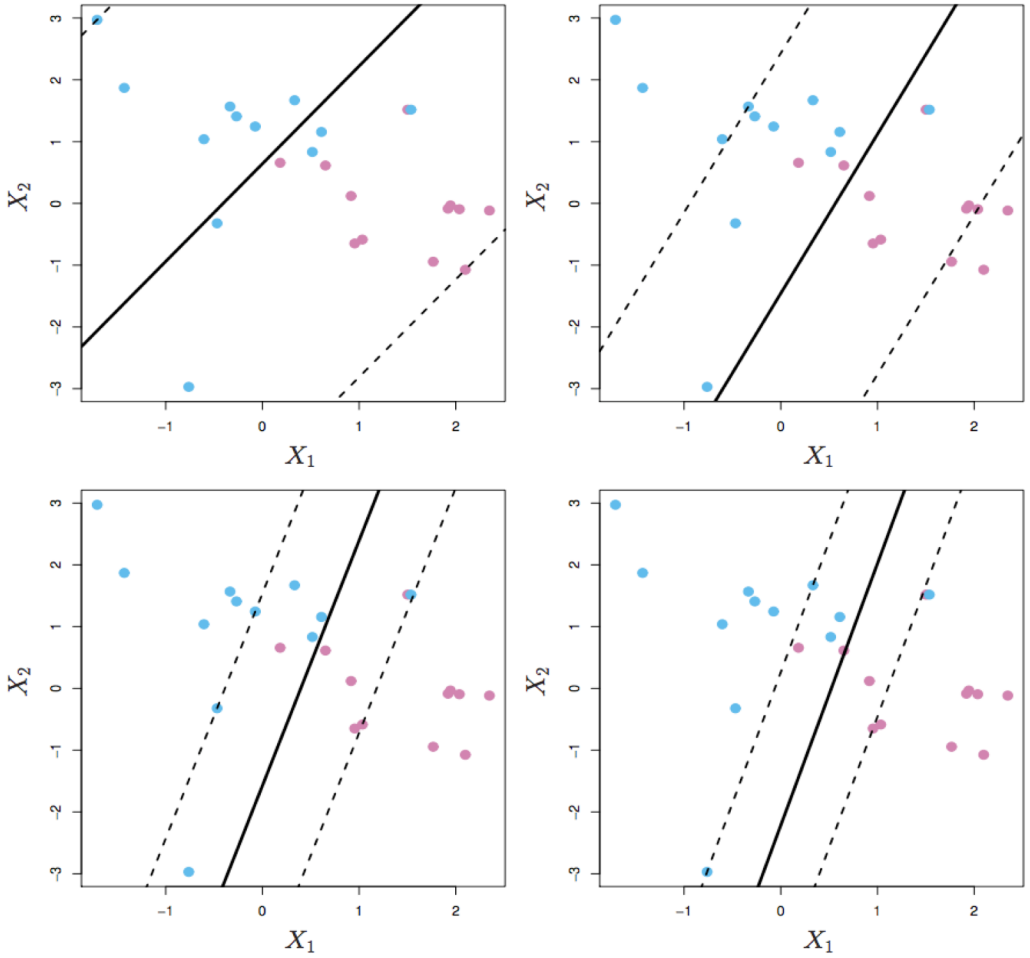
$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C, \quad \textbf{tuning parameter}$$

Support Vector Classifier: margin and hyperplane



A support vector classifier was fit to a small data set. The hyperplane is shown as a solid line and the margins are shown as dashed lines. Observations 3, 4, 5, and 6 are on the correct side of the margin, observation 2 is on the margin, and observation 1 is on the wrong side of the margin. Blue observations: Observations 7 and 10 are on the correct side of the margin, observation 9 is on the margin, and observation 8 is on the wrong side of the margin. No observations are on the wrong side of the hyperplane. Right: Same as left panel with two additional points, 11 and 12. These two observations are on the wrong side of the hyperplane and margin.

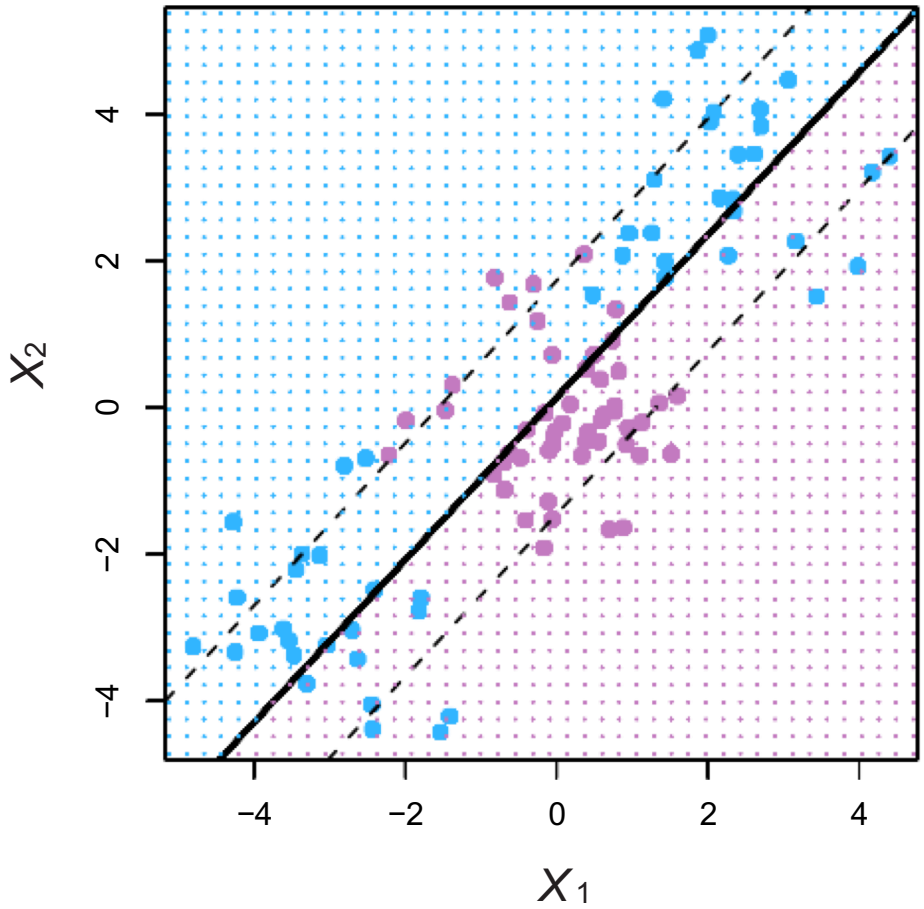
C is a regularization parameter



Demonstration

A support vector classifier was fit using four different values of the tuning parameter *C*. The largest value of *C* was used in the top left panel, and smaller values were used in the top right, bottom left, and bottom right panels. When *C* is large, then there is a high tolerance for observations being on the wrong side of the margin, and so the margin will be large.

Problem associated with support vector classifier: linear boundary can fail



Sometime a linear boundary just won't work if the data is nonlinearly separable. The example on the left is such a case. No matter what value of C is to be used, the support vector classifier will always fail. What should we do?

Feature expansion

- Enlarge the space of features by including transformations; e.g. X_1^2, X_1^3, X_1X_2, $X_1X_2^2$,.... Hence go from a p-dimensional space to a $M > p$ dimensional space.
- Fit a support-vector classifier in the enlarged 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

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

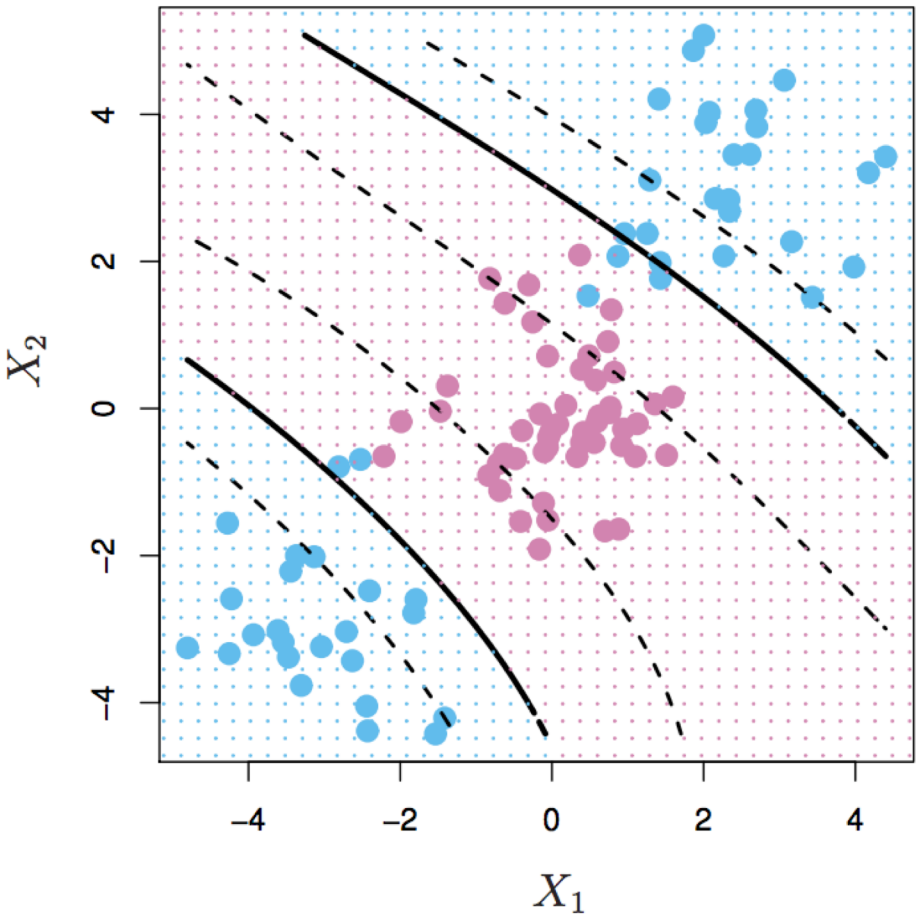
This leads to nonlinear decision boundaries in the original space (quadratic conic sections).

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



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

Demonstration

Nonlinearities and kernels

- Polynomials (especially high-dimensional ones) get very complicated rather fast.
- There is a more elegant and controlled way to introduce nonlinearities in support vector classifiers — through the use of *kernels*.
- Before we discuss these, we must understand the role of *inner products* in support vector classifiers.

Inner products and support vectors

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j} \quad \text{— inner product between vectors}$$

$$\begin{bmatrix} A_x & A_y & A_z \end{bmatrix} \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} = A_x B_x + A_y B_y + A_z B_z = \vec{A} \cdot \vec{B}$$

- The linear support vector classifier can be represented as

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

- To estimate the parameters $\alpha_1, \ldots, \alpha_n$ and β_0, all we need are the $\binom{n}{2}$ inner products $\langle x_i, x_{i'} \rangle$ between all pairs of training observations.

It turns out that most of the $\hat{\alpha}_i$ can be zero:

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

$\mathcal{S}$ is the *support set* of indices i such that $\hat{\alpha}_i > 0$.

Kernels and support vector machines

- If we can compute inner-products between observations, we can fit a SV classifier.
- Some special *kernel functions* can do this for us. E.g.

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

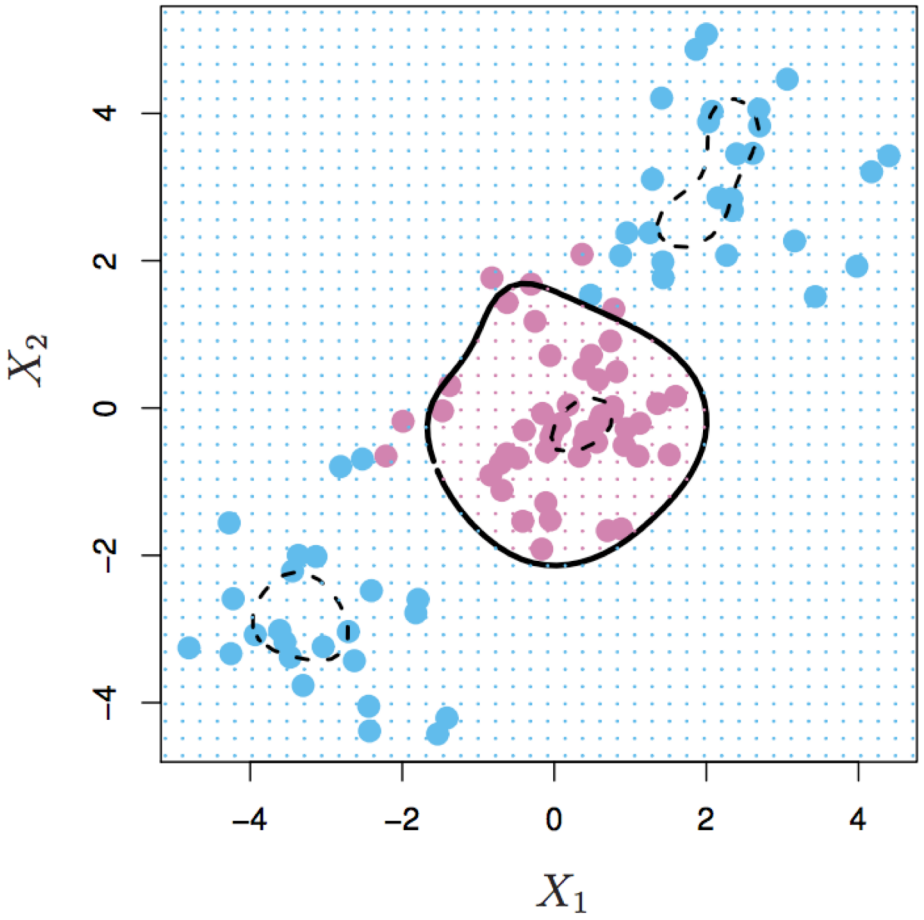
computes the inner-products needed for d dimensional polynomials — $\binom{p+d}{d}$ basis functions!

- The solution has the form

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i K(x, x_i).$$

Support Vector Machine with radial kernel

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



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

Implicit feature space; very high dimensional.

Demonstration

SVMs: more than 2 classes?

The SVM as defined works for $K = 2$ classes. What do we do if 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.

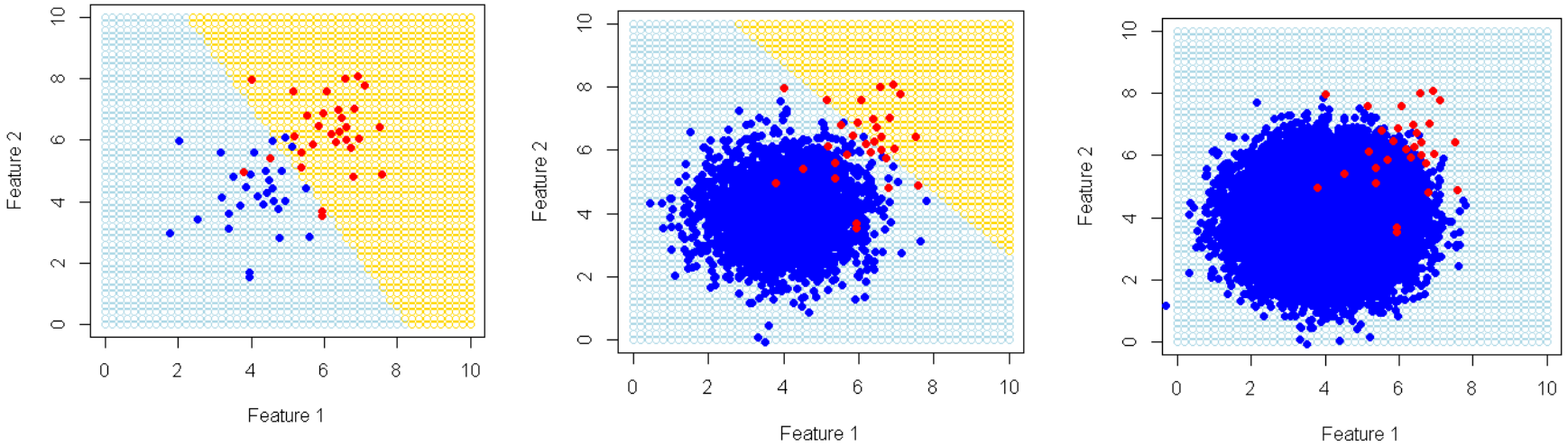
OVO One versus One. Fit all $\binom{K}{2}$ pairwise classifiers $\hat{f}_{k\ell}(x)$. Classify x^* to the class that wins the most pairwise competitions.

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

Characteristics of dataset

I: Class imbalance

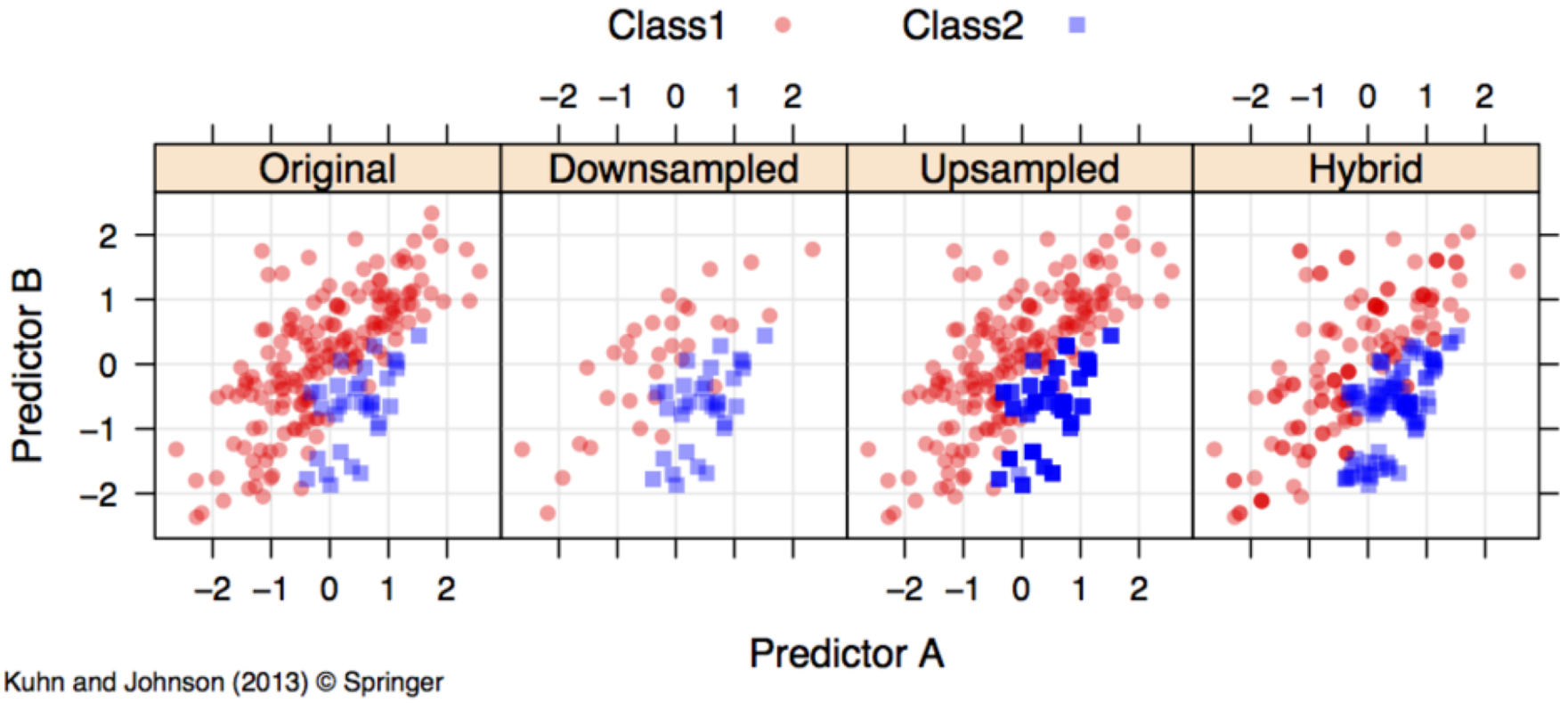
Degree of class imbalance



Decision boundary of a linear SVM

Assume ● are positive instances and ● are negative instances.

Random sampling to balance the data

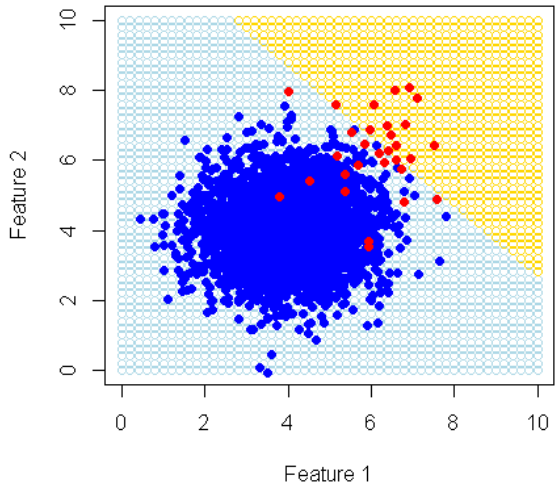


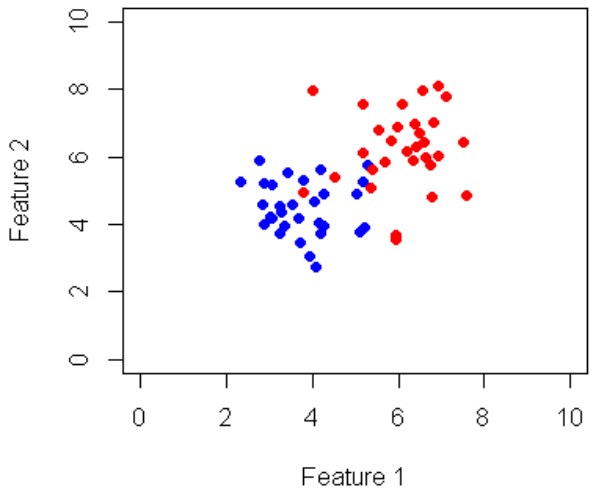
Kuhn and Johnson (2013) © Springer

Random up-sampling

Disadvantage: create duplicated and/or artificial instances which may introduce bias and /or noise to the original data

Random down-sampling to balance the data

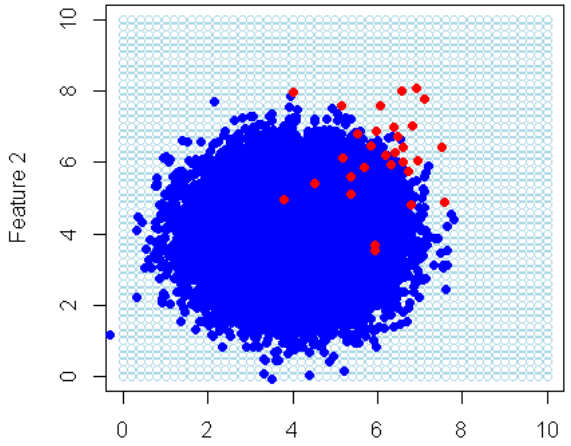




Advantage: do not introduce duplicates and/or artificial instances

Disadvantage: not all data points are used. Potentially removing useful information.

Better choice for data with very high class imbalance.



Repeated sampling

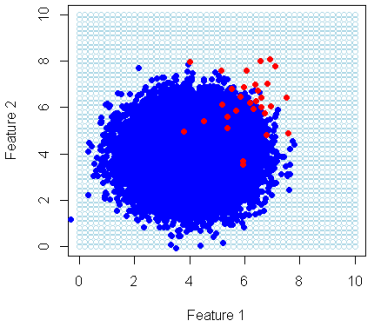
Phosphorylation sites: $\boldsymbol{x}_i \ (i = 1, \ldots, n)$

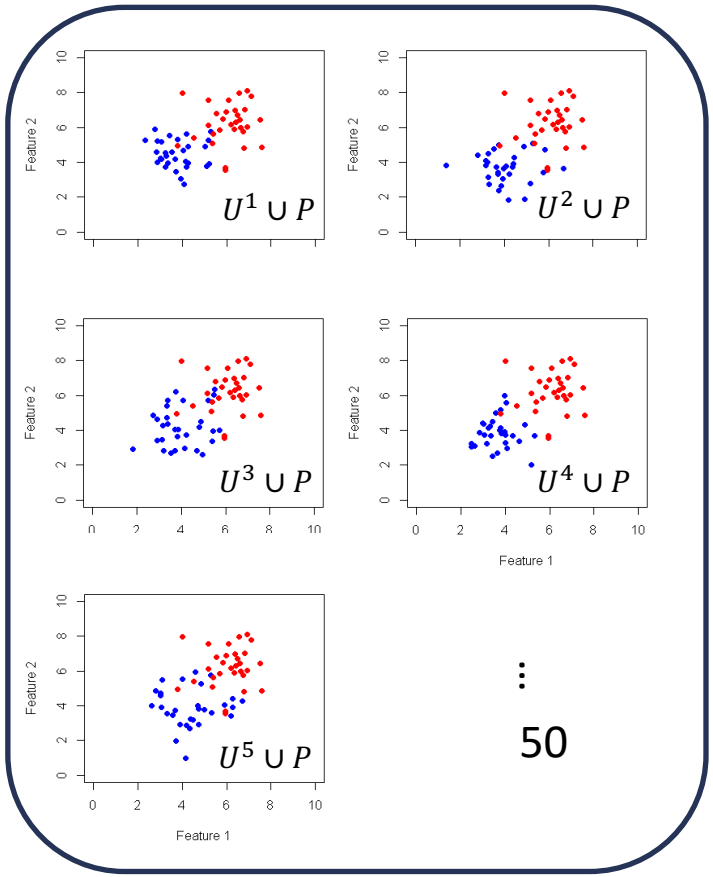
Whether is a substrate: $y_i \in \{-1, 1\}$

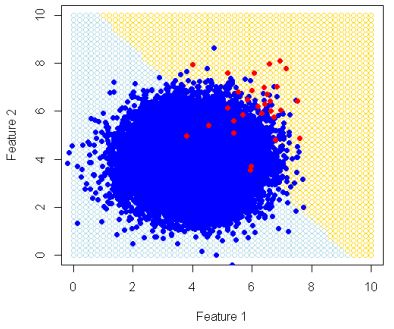
Classification: $p(y = 1|\boldsymbol{x}, D)$

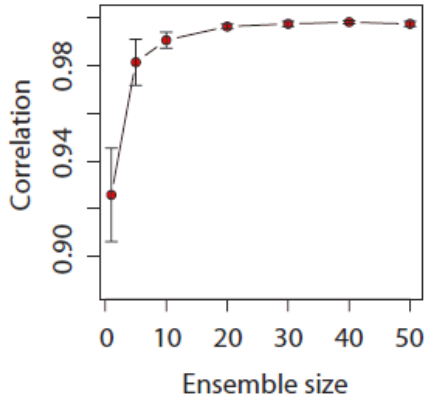
Random sampling from the majority class

$$p^E(y = 1|\boldsymbol{x}) = \frac{1}{r}\sum_{i=1}^{r} p(y = 1|\boldsymbol{x}, U^i \cup P)$$





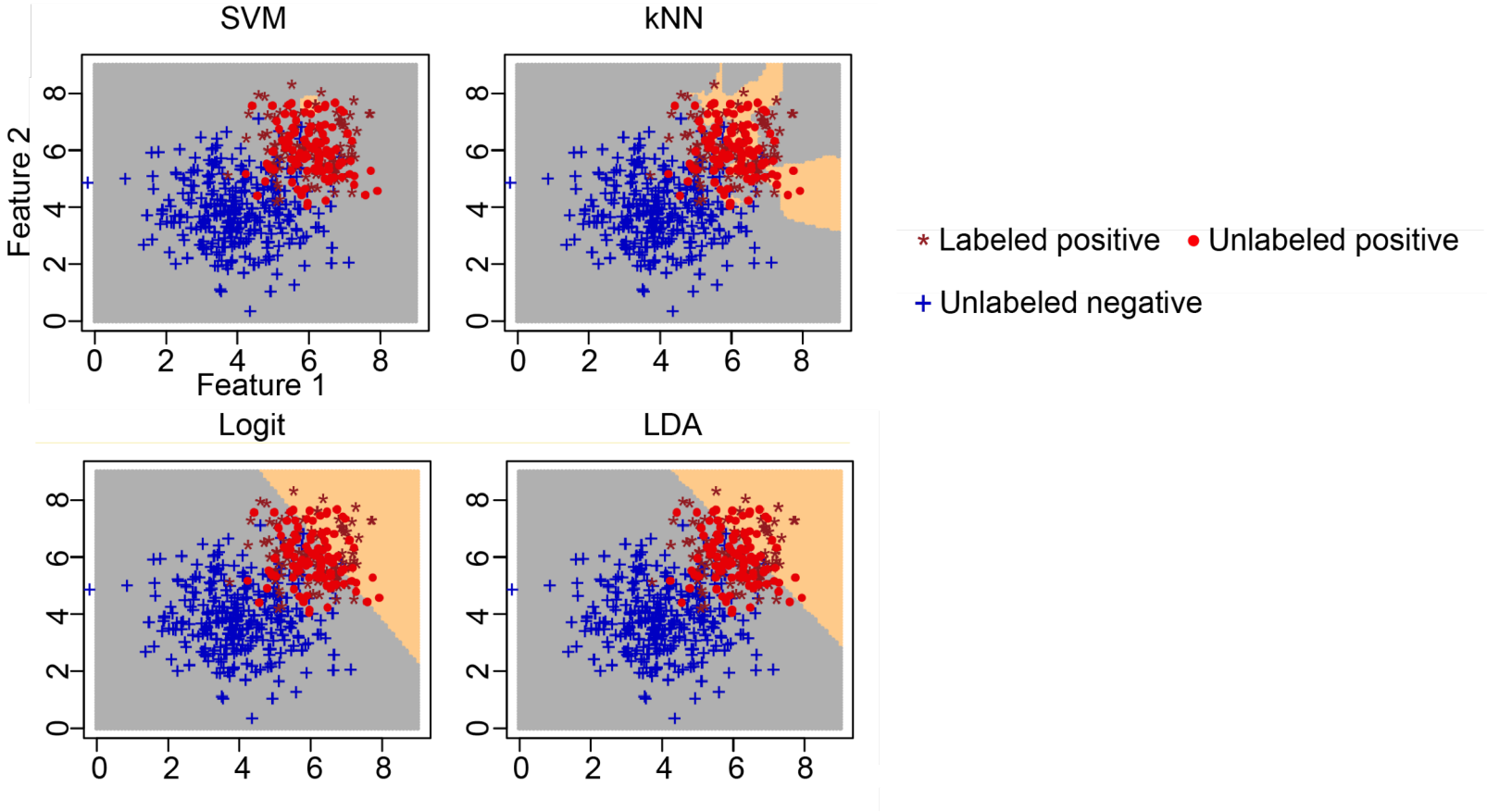




II: Partially labeled dataset

For example, when only a subset of positive instances are known whereas the rest of the instances are unlabeled.

This is known as **positive-unlabeled learning** – a type learning problem from partially labeled dataset.



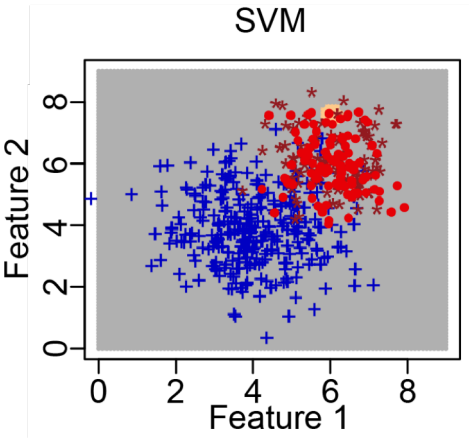
Semi-supervised learning

Semi-supervised learning is a class of **supervised learning** tasks and techniques that also make use of unlabeled data for training – typically a small amount of labeled data with a large amount of unlabeled data. (Wikipedia)

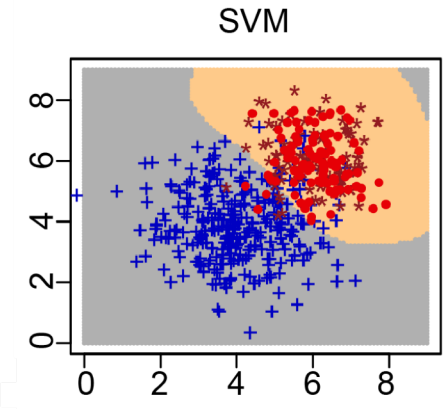
Positive-unlabeled learning is a type of a semi-supervised learning problem where only part of positive instances are labeled.

Typical approach for positive-unlabeled learning

1. Identify a robust set of negative instances from the unlabeled data.
2. Train a classifier with labeled positive and negative instance identified in step 1.
3. Classify all instances to recover unlabeled positive instances in the data.

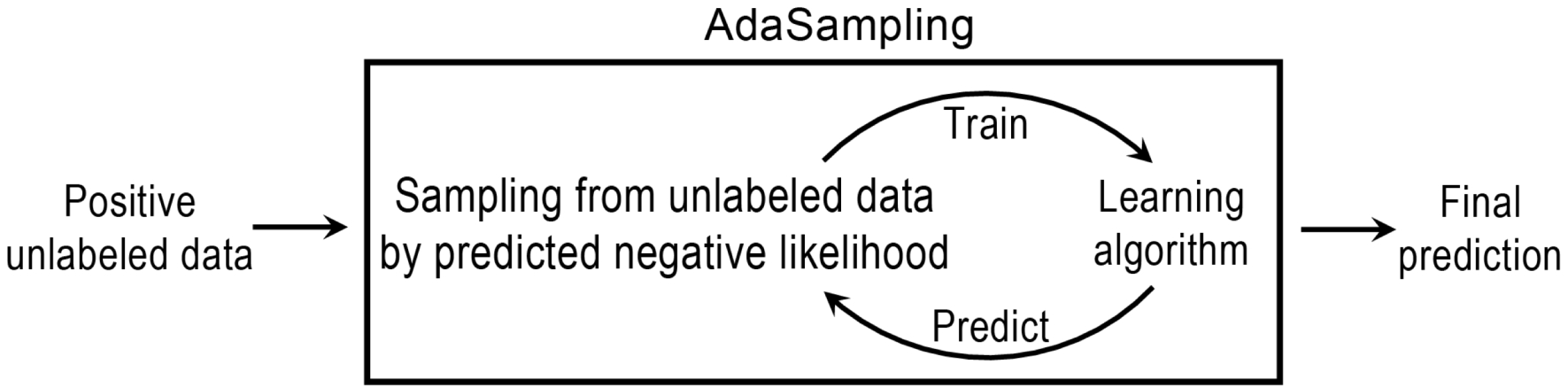


Before handling unlabeled data



After handling unlabeled data

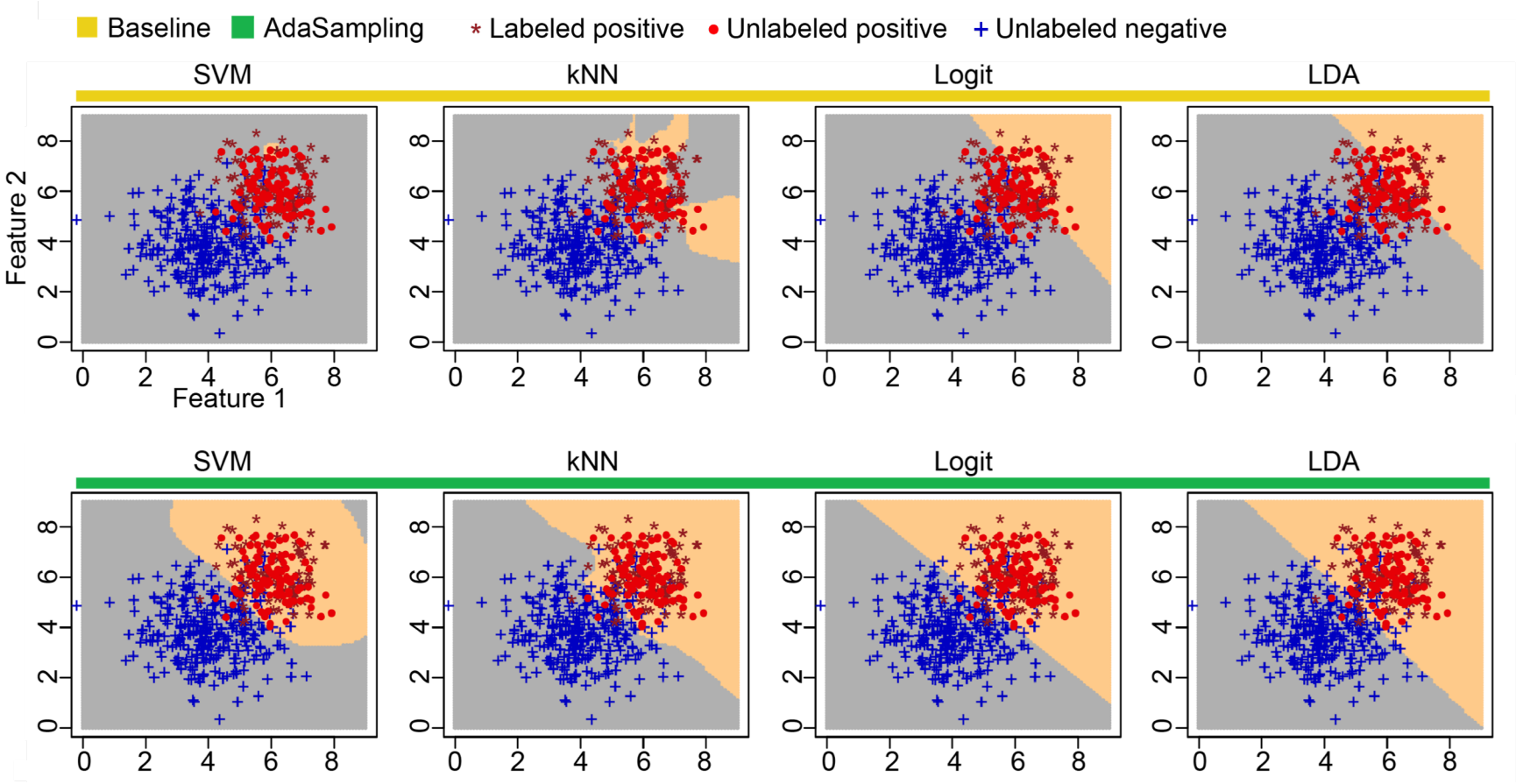
Adaptive Sampling for positive-unlabeled learning



P Yang et al. International Joint Conference on Artificial Intelligence (IJCAI), 2017.

AdaSampling for positive-unlabeled data

Results below are from classification with or without AdaSampling.



Evaluation (some tips)

- Evaluation (using cross-validation) on whether labeled positive instances are predicted correctly can be used as a segregate for model sensitivity.

- For specificity, we can make an assumption that all unlabelled instances are negative instances. This can give a lower bound estimation of model specificity.