UNIVERSITY OF EDINBURGH
Business School

Predictive Analytics and Modelling of Data

CMSE11428 (2020-2021)

Dr Xuefei Lu
The University of Edinburgh Business School

Support vector machines

Basic setup

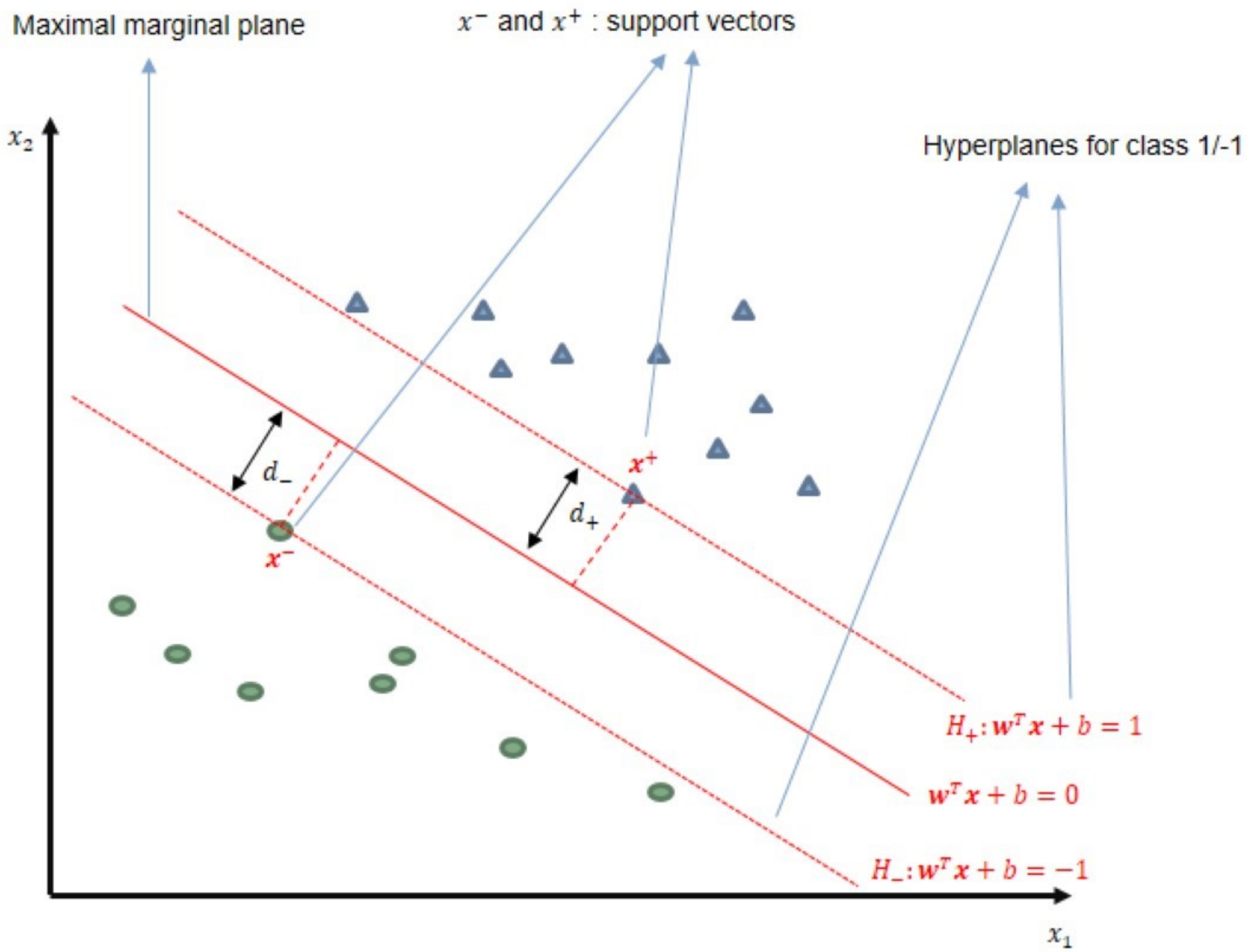
- Support vector machines construct hyperplanes (lines in 2D) that separate classes as much as possible
- They do so by maximising (using mathematical programming) the distance between the hyperplanes through the most extreme observations of each class (these observations are called support vectors)

Distance between maximal marginal plane and x^+:

$$d_+ = \frac{|\boldsymbol{w}^T\boldsymbol{x}^+ + b|}{||w||} = \frac{1}{||w||}$$

$||w||$ = square root of inner product $\boldsymbol{w}^T\boldsymbol{w}$

Total margin: $\frac{2}{||w||}$



Optimisation function

Maximising the distance between the margins between extreme points is easier expressed as a minimization exercise:

$$\min \frac{\boldsymbol{w}^T \boldsymbol{w}}{\mathbf{2}}$$

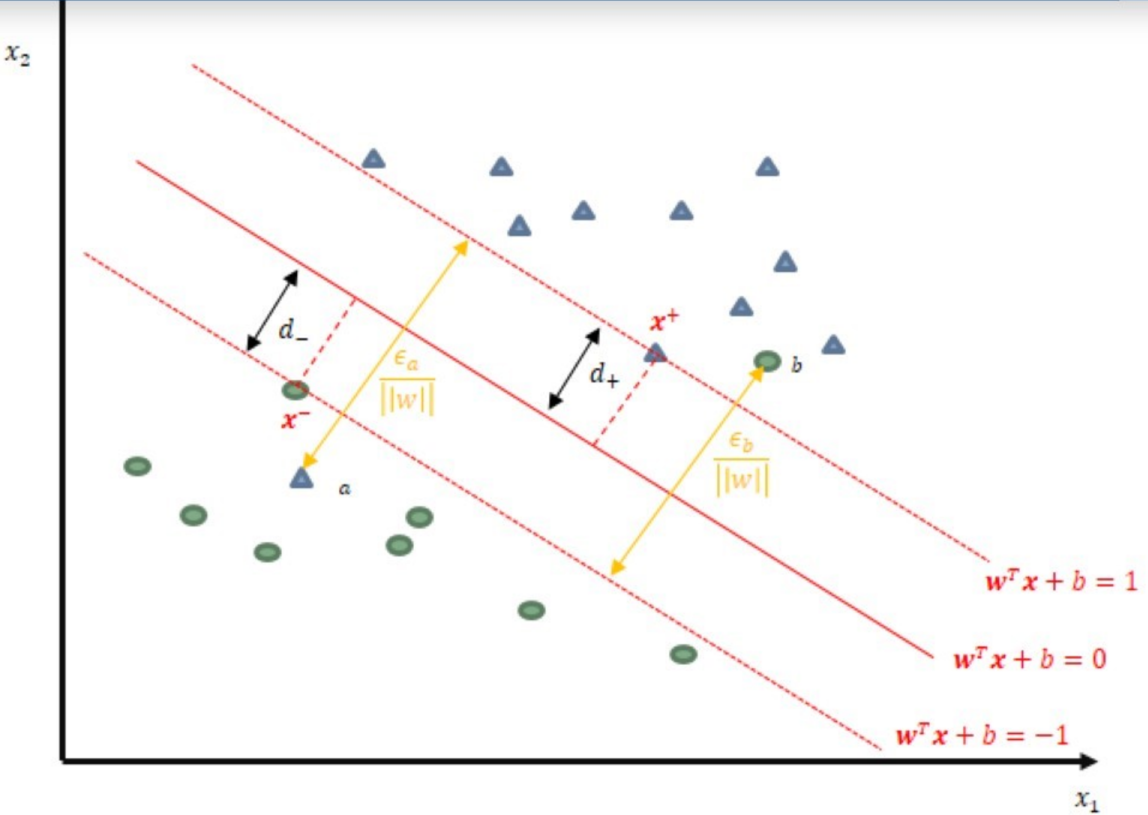
subject to $\quad y_i\underbrace{(\boldsymbol{w}^T \boldsymbol{x}_i + b)}_{\hat{y}_i} \geq 1, \quad i = 1, \ldots, n$ (for n datapoints)

Note: $y_i = 1, \hat{y}_i = 1, \quad y_j = -1, \hat{y}_j = -1, \quad y_i\hat{y}_i = 1 = y_j\hat{y}_j$

This is a quadratic programming problem solved using Lagrangian optimisation

What if we can't find perfect separation?

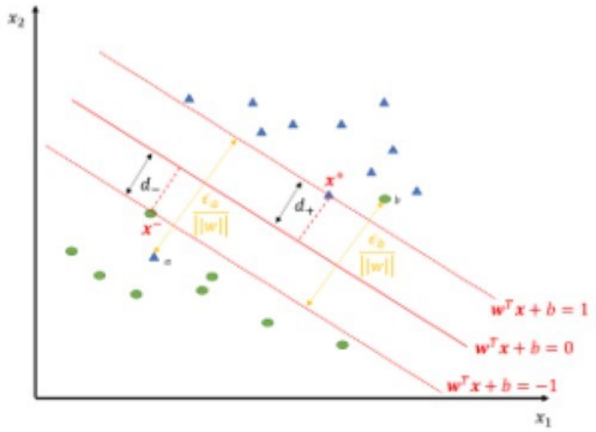
If we have outliers or points lying beyond the margins, we need to allow for slack



x_1

What if we can't find perfect separation?

- We introduce 'soft margins' by using 'slack variables' $\epsilon_i \geq 0$
  - $H_+ = \boldsymbol{w}^T\boldsymbol{x} + b \geq 1 - \epsilon_i \qquad (y = 1)$
  - $H_- = \boldsymbol{w}^T\boldsymbol{x} - b \leq -1 + \epsilon_i \qquad (y = -1)$



- Optimisation:

$$\min \frac{\boldsymbol{w}^T\boldsymbol{w}}{2} + C\Sigma_{i=1}^{n}\epsilon_i$$
$$\text{St. } y_i(\boldsymbol{w}^T\boldsymbol{x} + b) \geq 1 - \epsilon_i, i = 1, \dots, n$$

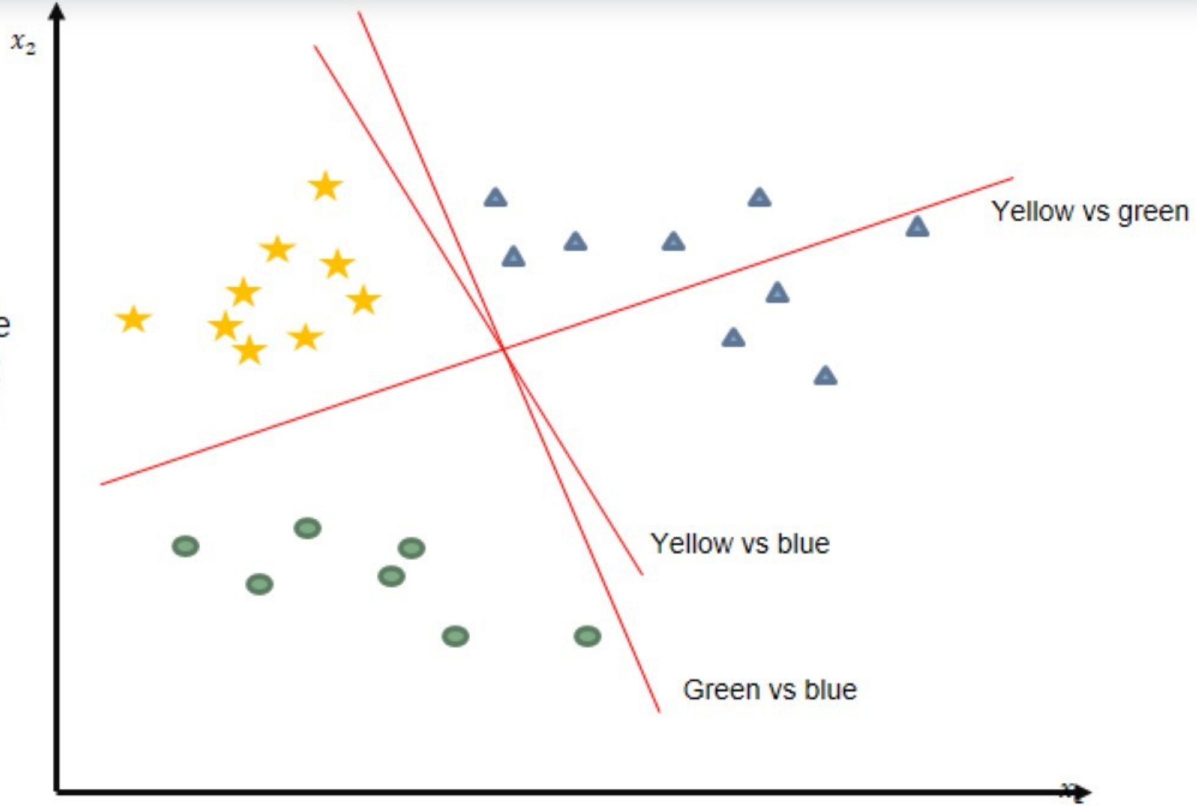
  - C is a cost parameter which controls how much we 'punish' observations that deviate from their classes

What with multi-class classification?

- Two approaches:
  - One-vs-one: build a model for every combination of classes
  - One-vs-rest (or One-vs-All): build a binary model for every predictor vs. all other observations being the other class

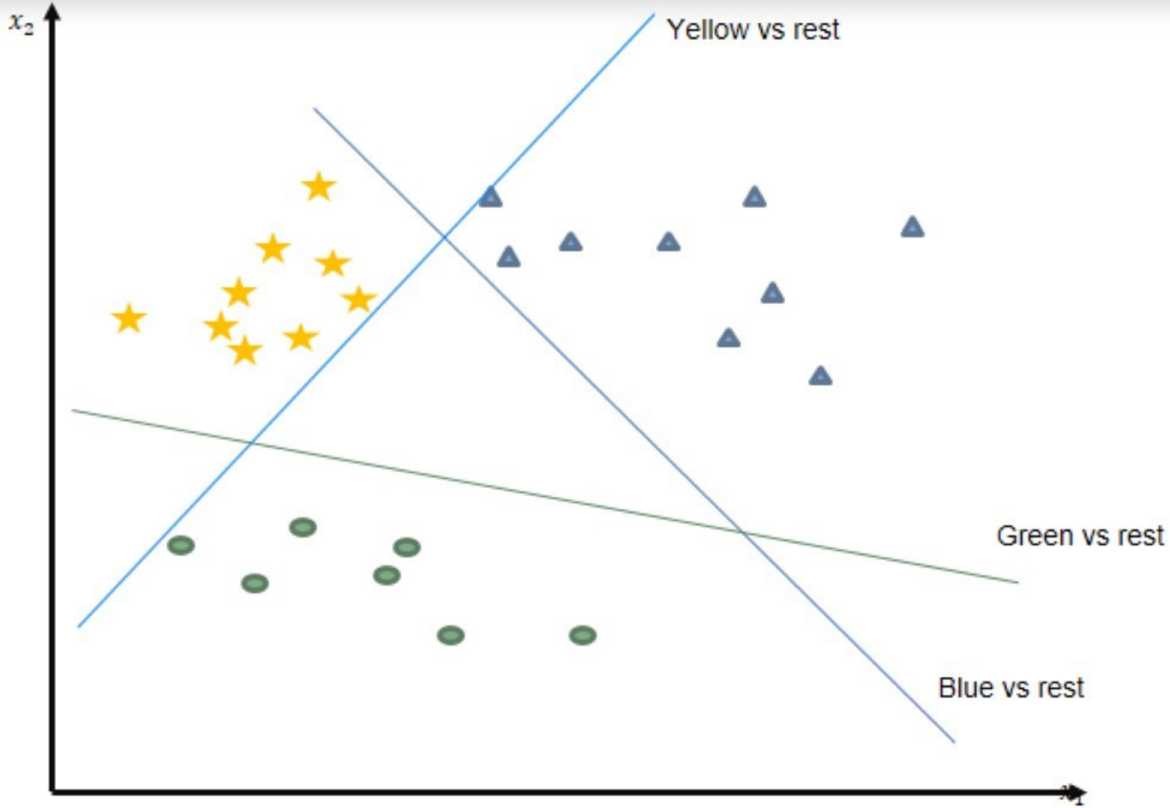
One vs. one

A prediction is made for a new data point based on a majority vote between the different models



x_1

One vs. rest



OvR & OvO

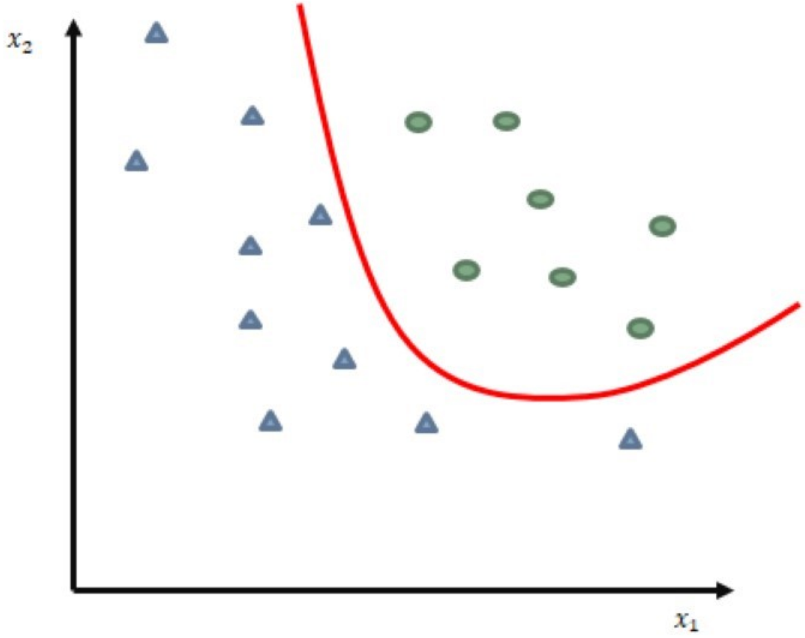
- For OvR, we only have to train as many SVMs as there are classes,
- For OvO, we have to train as many SVMs as there are class pairs $l(l-1)2$ with l the number of classes).
- OvO approach is much more robust against class imbalances, as classes are used multiple times to train a model.
- OvR also suffers from the fact that data points get different classes assigned in multiple models ➔ classification ambiguous.

- 5 - Building support vector machines.ipynb + churn_ibm.csv

Activity: Building support vector machines

The kernel trick

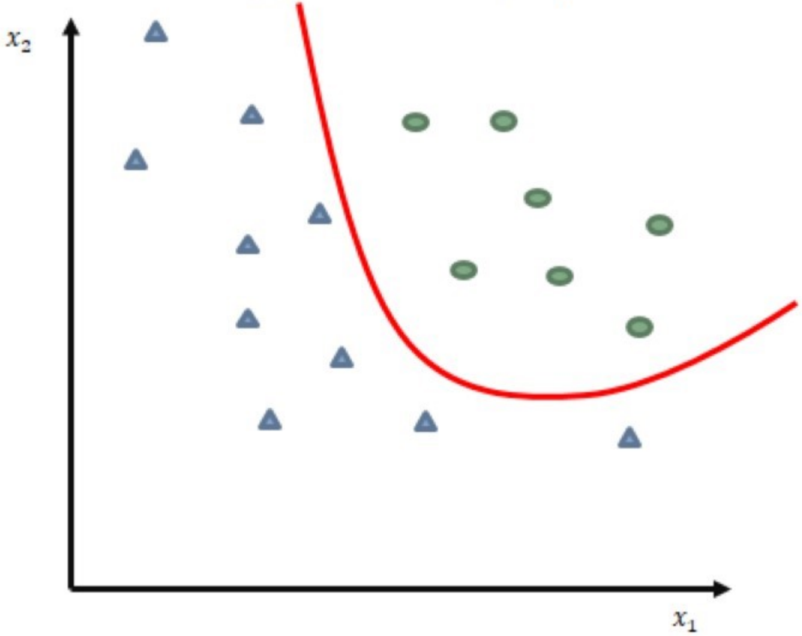
- Linear separation might not always work:

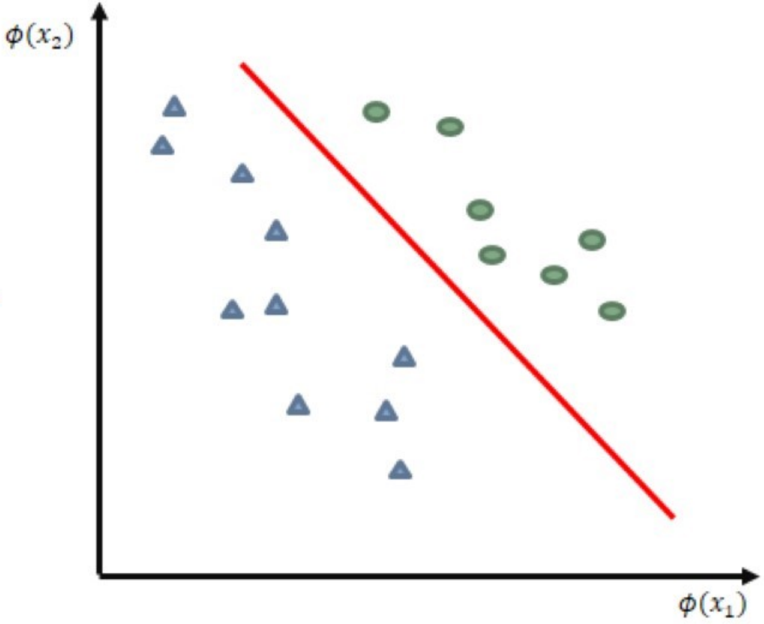


The kernel trick

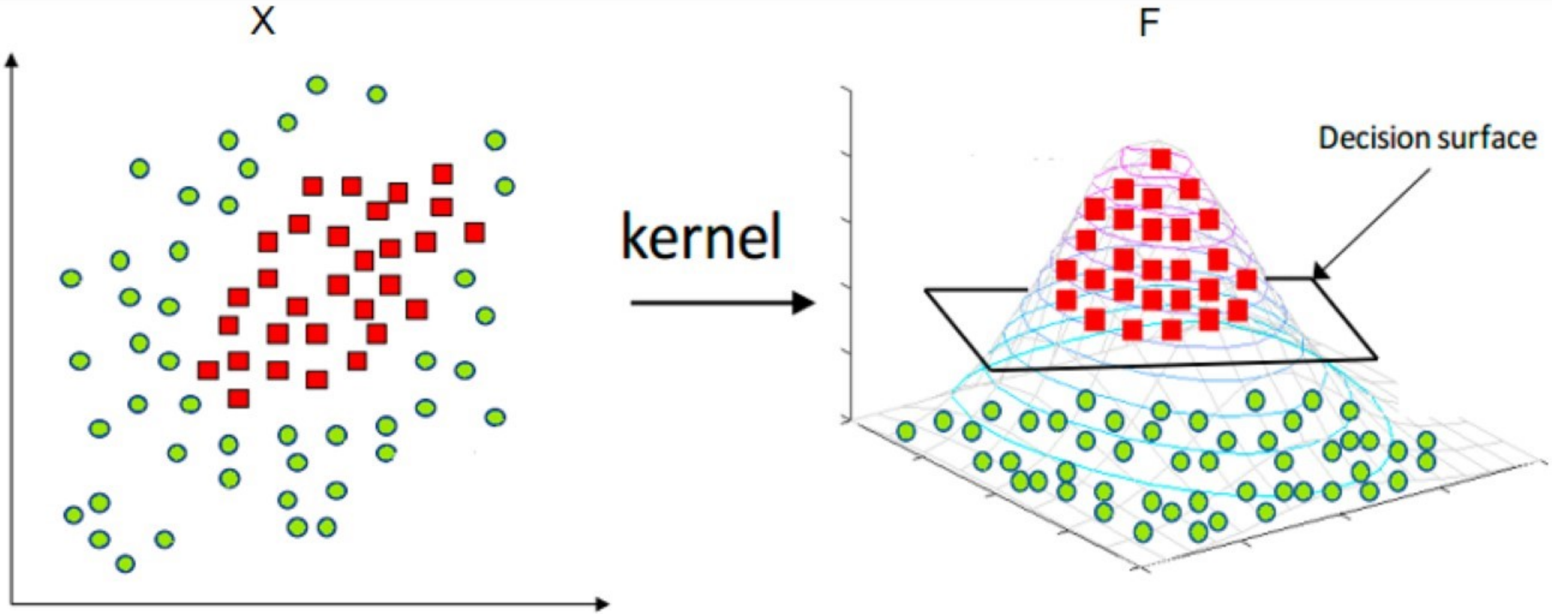
We need to transform our input data, e.g. with transformation function $\phi(\cdot): X \rightarrow F$

The goal is to find the best transformation function that is capable of turning the feature spaces into something that is linearly separable.





The kernel trick



https://medium.com/@zxr.nju/what-is-the-kernel-trick-why-is-it-important-98a98db0961d

The kernel trick

- We use a transformation to a higher dimension in which linear separation can be found
- We use kernel functions K for which the transformation ϕ is the dot product of two vectors.

- Example:
  - $\phi: \mathbb{R}^2 \to \mathbb{R}^3: \binom{x_1}{x_2} \to \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix}$
  - $\phi(\boldsymbol{x}) \cdot \phi(\mathbf{z}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)\begin{pmatrix} z_1^2 \\ z_2^2 \\ \sqrt{2}z_1z_2 \end{pmatrix} = x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2$

$$= (x_1z_1 + x_2z_2)^2 = (\boldsymbol{x} \cdot \mathbf{z})^2 = \mathrm{K}(\mathbf{x}, \mathbf{z})$$

  - Here, the kernel function is the squared dot product (polynomial kernel)
  - We can apply this kernel function, and don't need to know what the transformation looks like, but our data is mapped to a higher dimension!

UNIVERSITY OF EDINBURGH Business School

The kernel trick

- Some kernel functions that have this property:
  - Linear kernel: $K(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x} \cdot \boldsymbol{z})$
  - Polynomial kernel: $K(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x} \cdot \boldsymbol{z})^d, d \in \mathbb{N}$
  - Gaussian kernel: $K(\boldsymbol{x}, \boldsymbol{z}) = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{z}\|^2}{2\sigma^2}}, \sigma > 0$
  - Radial Basis Function (RBF): $K(\boldsymbol{x}, \boldsymbol{z}) = e^{-\gamma\|\boldsymbol{x}-\boldsymbol{z}\|^2}, \gamma > 0$

- Best approach: try different ones.

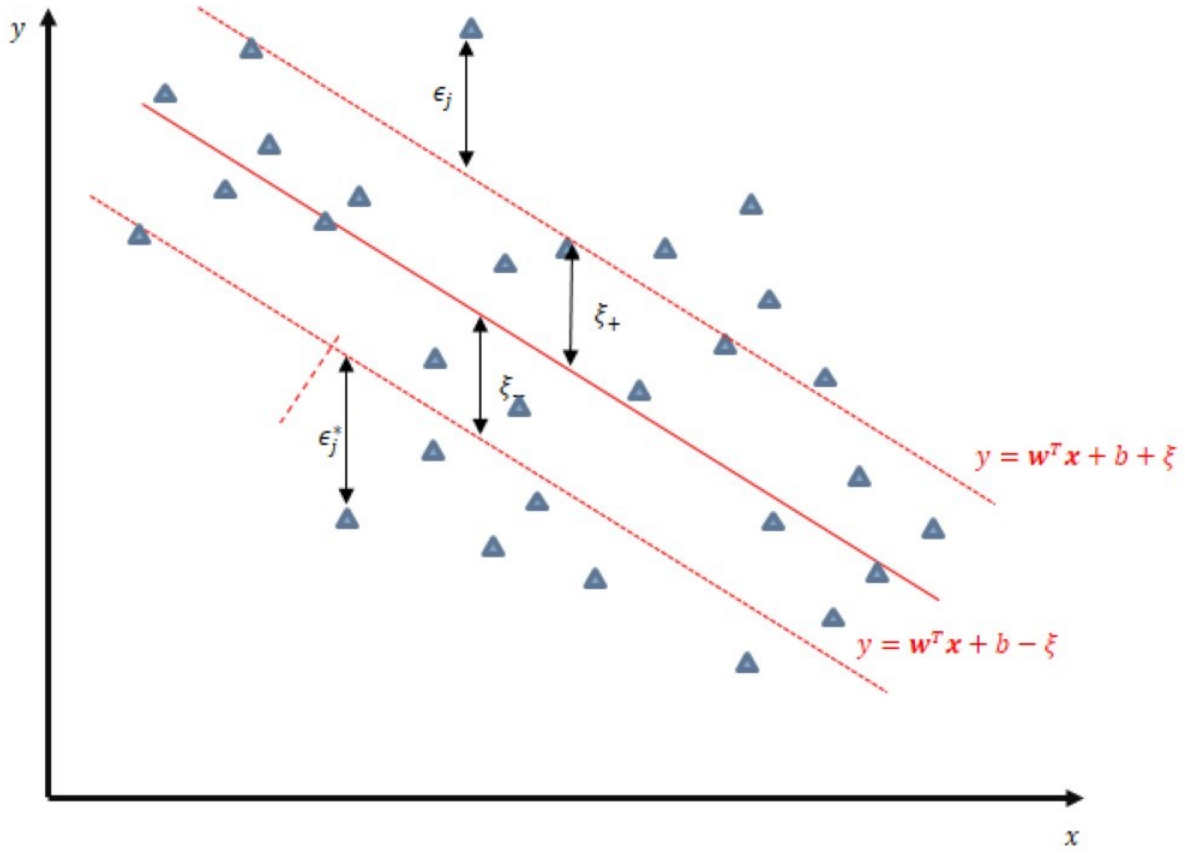
Activity: Build SVMs with different kernels

- 6 - Building support vector machines with different kernels.ipynb + churn_ibm.csv

Support vector regression

ξ allows for (residual) errors not being counted towards the minimisation

Again ϵ allows for further soft margins beyond ξ



Optimization

- We are looking for the best hyperplane(s) again that minimise(s) the errors:

$$\min \frac{\boldsymbol{w}^{\mathrm{T}}\mathbf{w}}{2} + C(\Sigma_{i=1}^{n} \epsilon_{\mathrm{i}} + \epsilon_{\mathrm{i}}^{*})$$

St. $y_i - \boldsymbol{w}^T \boldsymbol{x}_i - b \leq \xi + \epsilon_i$ — Distance to 'upper' hyperplane

$\boldsymbol{w}^T \boldsymbol{x}_i + b - y_i \leq \xi + \epsilon_i^*$ — Distance to 'lower' hyperplane

$\epsilon_i, \epsilon_i^* \geq 0$

- Again, C is a cost parameter, and ϵ_i and ϵ_i^* represent the errors beyond the upper/lower hyperplanes respectively
- Similarly to support vector machines, different kernels can be used as well.
- Large ξ leads to underfitting; small ξ can result in overfitting.

What about support vector regression?

- Mathematical programming approach to minimise the residual errors
- Margins are used again: to tolerate errors within a particular band around a hyperplane (which is our regression line)
- This is captured by the variable ξ, hence this approach is called ξ-insensitive regression
- The data points lying on the planes at distance ξ are called the support vectors
- Different kernels can be used again

SVMs wrap-up

- Benefits:
  - Are a non-parametric approach that can capture very complex relationships using different kernels
  - Work very well in a high-dimensional environment

- Downsides:
  - Linear separation might not always be obtainable straightforwardly (even after using transformation)
  - Can overfit quickly

Want more math? Check out https://towardsdatascience.com/support-vector-machines-for-classification-fc7c1565e3

- 7 - Building a support vector regression model.ipynb

Activity: Building an SVR model

Quiz

Which statement is true?

A) SVMs can divide the feature space using non-linear separation.

B) Decision trees can divide the feature space into any shape.

Quiz

Support vectors are...

- A...the hyperplanes separating the classes.
- B...lying on the hyperplanes separating the classes.
- C...lying on the maximal margin plane.
- D. None of the above.

Quiz

SVMs/SVRs are known for:

- A. Being able to handle a high dimensional feature space.
- B. Being able to classify with non-linear relations.
- C. Using optimisation.
- D. All of the above.

Quiz

Which statement is true?

- A. Non-linear kernels are better regardless the dataset.
- B. A higher cost parameter leads to overfitting.
- C. A lower cost parameter is better for large datasets.
- D. None of the above.

UNIVERSITY OF EDINBURGH
Business School