INT104 ARTIFICIAL INTELLIGENCE

Support Vector Machine Review

Fang Kang
Fang.kang@xjtlu.edu.cn

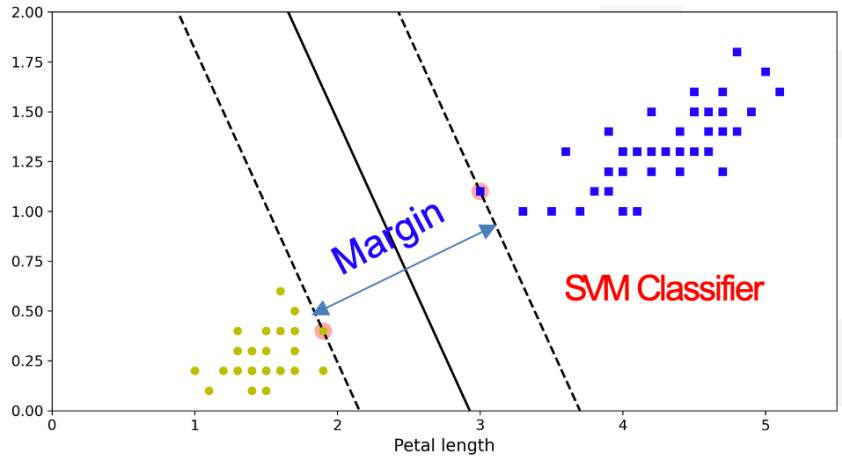
Xi'an Jiaotong-Liverpool University
西交利物浦大学

CONTENT

- Linear SVM
  - Hard-margin Classification
  - Soft-margin Classification
- Non-Linear SVM
  - Nonlinear SVM Classification
  - Kernel method
  - Polynomial Features
- SVM Regression

Hard Margin Classification

- All instances being off the street and on the right side is named "*hard margin classification*"



Minimize: $\frac{1}{2}||\mathbf{w}||^2$

Subject to: $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \qquad (i = 1 \sim N)$

- The main limitation of hard margin classification is
  - The data must be linearly separable
  - Sensitive to outliers

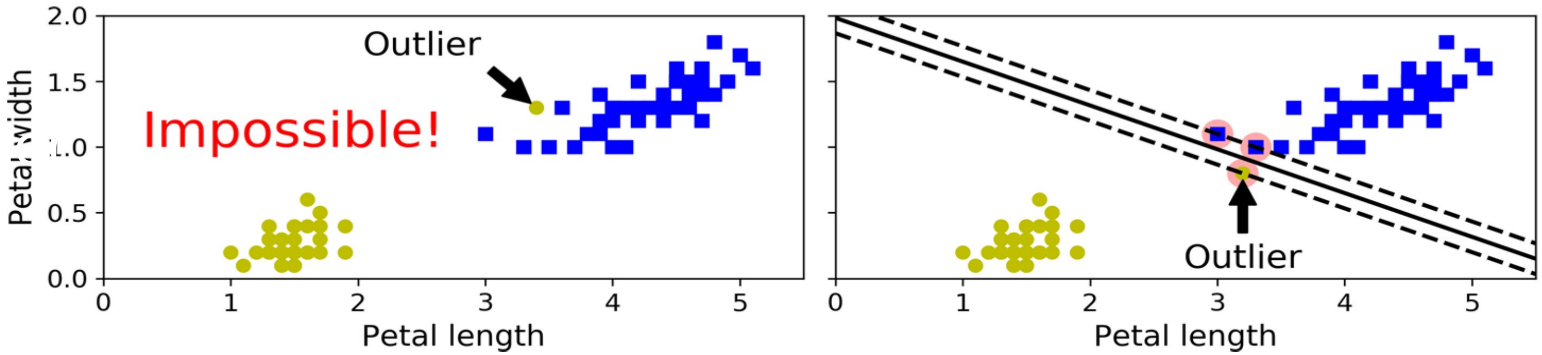


Figure 5-3. Hard margin sensitivity to outliers

Soft Margin Classification

- Allow margin violations
- The algorithm balances
  - The width of street
  - The amount of margin violations
- A hyper-parameter C is defined
  - A low value of C leads to more margin violations
  - A high value of C limits the flexibility

Minimize: $\frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i$ (Slack variable)

Subject to: $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \quad (i = 1 \sim N)$

$\xi_i \geq 0$

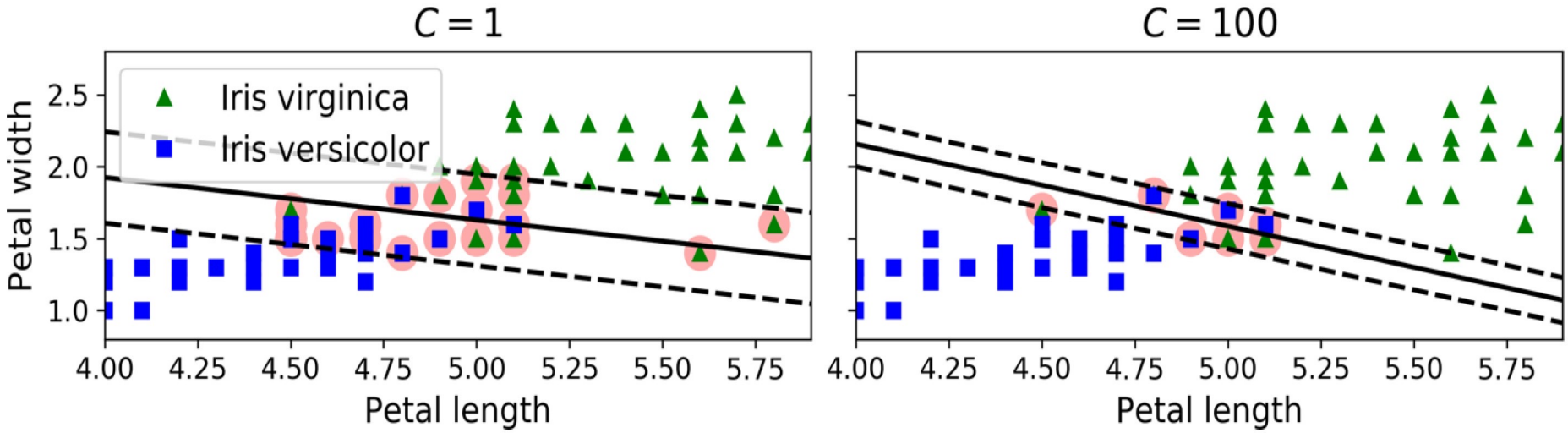


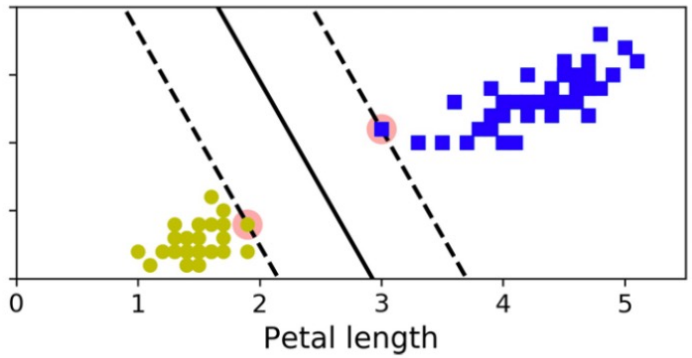
Figure 5-4. Large margin (left) versus fewer margin violations (right)

Tip:

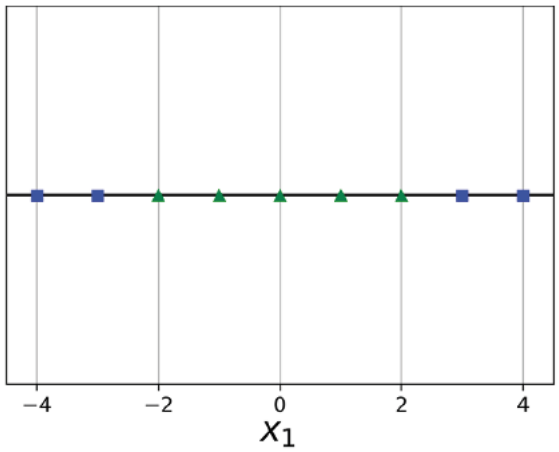
- To regularize SVM, reduce C
- C is inverse of regularizing hyperparameter alpha $J(\boldsymbol{\theta}) = \text{MSE}(\boldsymbol{\theta}) + \alpha\frac{1}{2}\sum_{i=1}^{n}{\theta_i}^2$

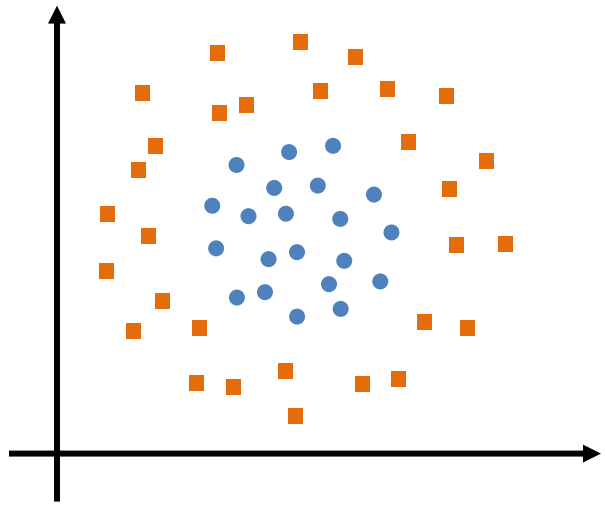
Nonlinear SVM Classification

Linearly separable

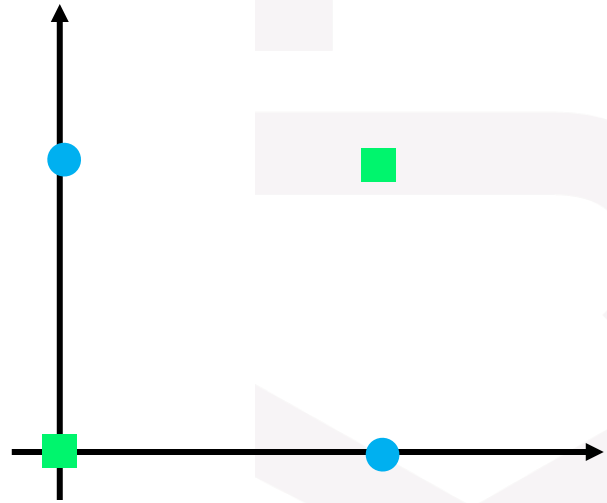


How do we deal with these cases?





Not linearly separable



Nonlinear SVM Classification

How do we deal with nonlinearity?

Solution

1. Directly tackle nonlinear problems

Multilayer Perceptron， Deep Learning, etc.

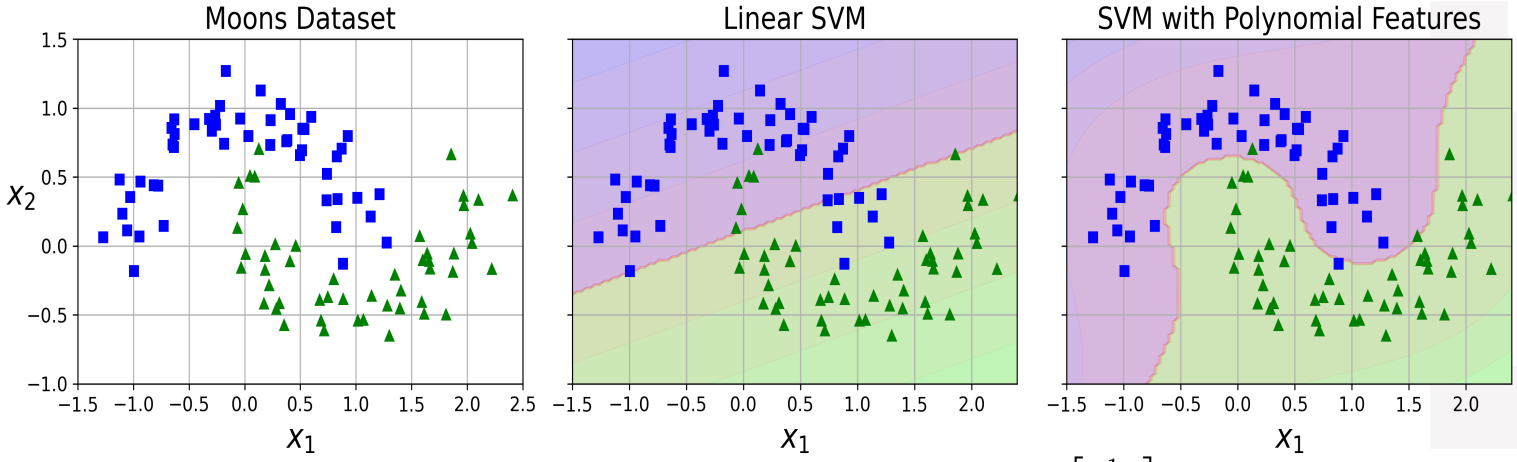
2. Transform nonlinear problems into linear issues

Kernel SVM

Nonlinear SVM Classification

- Polynomial Features

  Polynomial features involve taking an existing feature and raising it to a power. This is useful for capturing non-linear relationships between the feature and the target variable. For example, if you have a feature X, polynomial features could include X^2, X^3, etc.



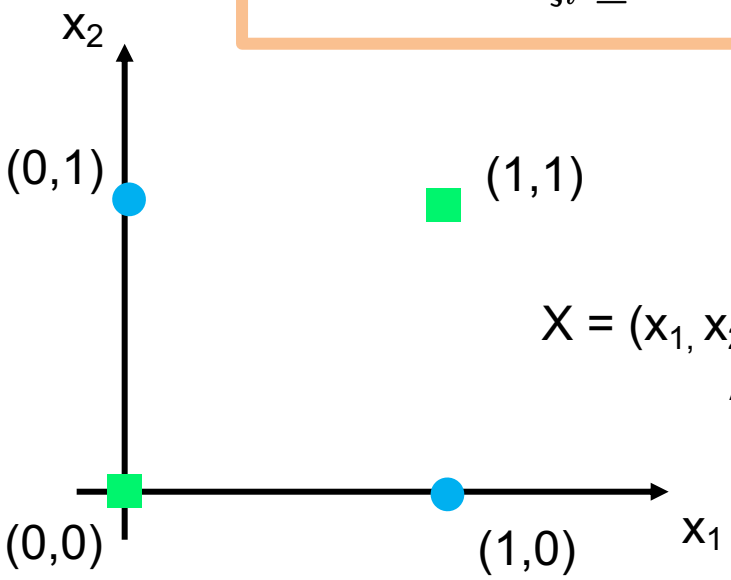
$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 \\ X_1 \\ X_2 \\ X_1X_2 \\ X_1^2 \\ X_2^2 \\ X_1^2X_2 \\ X_1X_2^2 \\ X_1^3 \\ X_2^3 \end{bmatrix}$$

Polynomial features of degree 3
2D to 10D

Nonlinear SVM Classification

Minimize: $\frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi_i$ (ξ_i Slack variable)

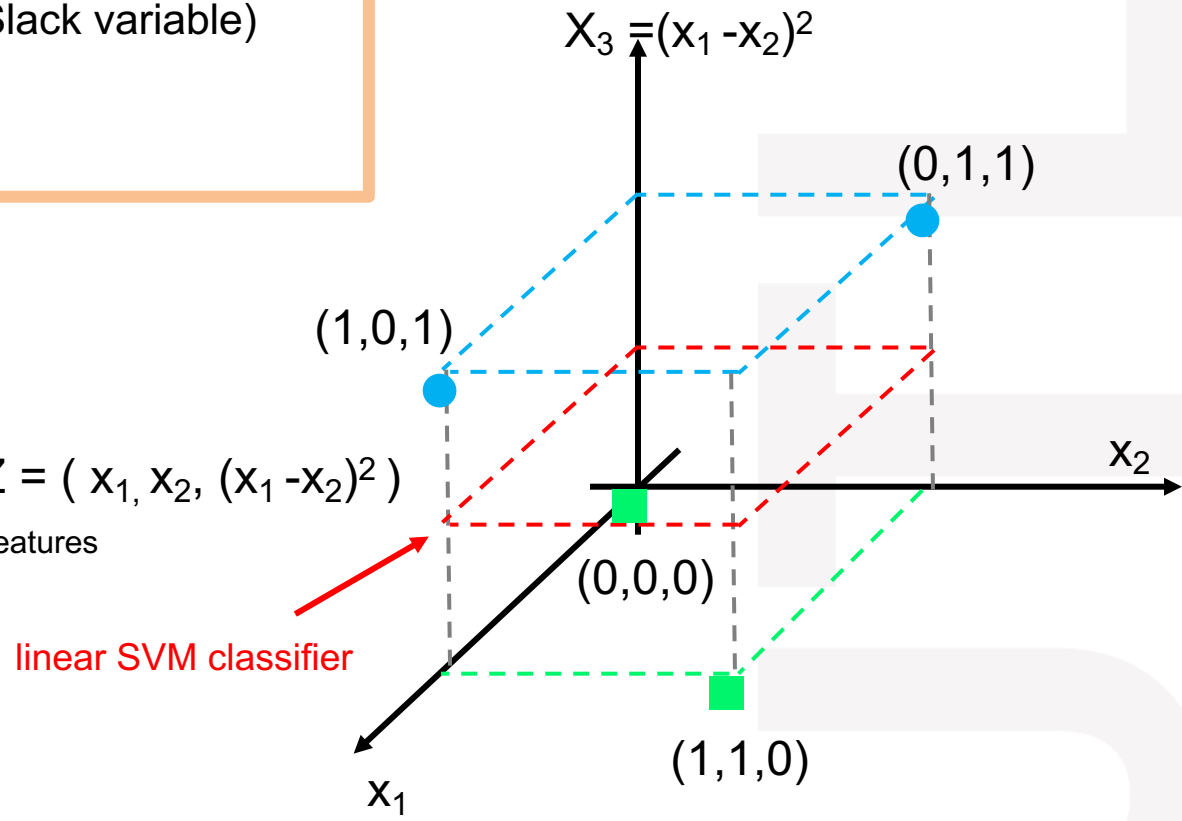
Subject to: $\xi_i \geq 0$



$\emptyset(x)$

$X = (x_1, x_2) \Longrightarrow Z = (x_1, x_2, (x_1 - x_2)^2)$

Add polynomial features



Nonlinear transformation $\emptyset(x)$: not only one form

Cover's theorem: High-dimensional space is more likely to be linearly separable than in a low-dimensional space.

https://en.wikipedia.org/wiki/Cover's_theorem

Nonlinear SVM Classification

Minimize: $\frac{1}{2}||\mathbf{w}||^2$

Subject to: $y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \qquad (i = 1 \sim N)$

$$\max_{\lambda} -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j x_i^T x_j + \sum_{i=1}^{N}\lambda_i,\ s.t.\ \lambda_i \geq 0$$

$\emptyset(x_i)^T\emptyset(x_j)$

$\phi(\mathbf{x})_i$ High dimension (infinite)

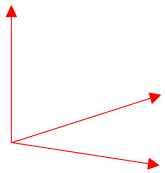
Difficult to calculate!

How to solve $\emptyset(x_i)^T\emptyset(x_j)$?

Nonlinear SVM: Kernel Trick

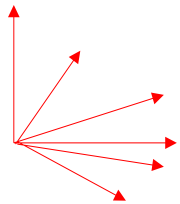
Input Space: dimension n

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$



High-dimensional Feature Space: dimension N >> n

$$\phi(x) = \begin{bmatrix} \phi_1(x) \\ \phi_2(x) \\ \phi_3(x) \\ \vdots \\ \phi_N(x) \end{bmatrix} \quad N \gg n$$



Expensive operation and requires large memory

$$K(\boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{a}^T \boldsymbol{b} = [a_1 \quad a_2 \quad a_3 \quad \cdots \quad a_n] \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}$$

$$K(\phi(\boldsymbol{a}), \phi(\boldsymbol{b})) = \phi(\boldsymbol{a})^T \phi(\boldsymbol{b}) = [\phi_1(a) \quad \phi_2(a) \quad \phi_3(a) \quad \cdots \quad \phi_N(a)] \begin{bmatrix} \phi_1(b) \\ \phi_2(b) \\ \phi_3(b) \\ \vdots \\ \phi_n(b) \end{bmatrix}$$

$$\phi(\boldsymbol{a})^T \phi(\boldsymbol{b}) = \text{function}\left(\boldsymbol{a}^T \boldsymbol{b}\right)$$

Kernel Trick

Common kernels:

$$\begin{aligned} \text{Linear: } K(\mathbf{a}, \mathbf{b}) &= \mathbf{a}^\top \mathbf{b} \\ \text{Polynomial: } K(\mathbf{a}, \mathbf{b}) &= \left(\gamma \mathbf{a}^\top \mathbf{b} + r\right)^d \\ \text{Gaussian Radial Basis Function: } K(\mathbf{a}, \mathbf{b}) &= \exp\left(-\gamma \|\mathbf{a} - \mathbf{b}\|^2\right) \\ \text{Sigmoid: } K(\mathbf{a}, \mathbf{b}) &= \tanh\left(\gamma \mathbf{a}^\top \mathbf{b} + r\right) \end{aligned}$$

Universal approximator.
Corresponding feature space
$\phi(x)$ is infinite dimensional space

non-linearly separable data

infinite-dimensional space

Nonlinear SVM: Kernel Method

Linearly separable	A little violations	Strict nonlinearity
Hard-margin SVM	Soft-margin SVM	$\emptyset(x)$+hard-margin -> kernel SVM

Kernel Method: Ideologically, transform low-dimensional non-linear space to high-dimensional linear space, using kernel function.

Kernel Function: Kernel Function $= < \emptyset(x), \emptyset(x') >= \emptyset(\boldsymbol{x}_i)^{\mathrm{T}}\emptyset(\boldsymbol{x}_j)$, <·> means dot-product
It covers non-linear transformations and an inner product operation on nonlinear transformations.

Kernel Trick: Computationally, avoiding explicitly computing the transformation to another feature space.

- Project input space into a very high-dimensional feature space, may be even infinity
- Problem:
  - Projecting training data in to a high-dimensional space is expensive
  - large number of parameters
- Trick:
  - Compute dot-product between training samples in the projected high-dimensional space without ever projecting.