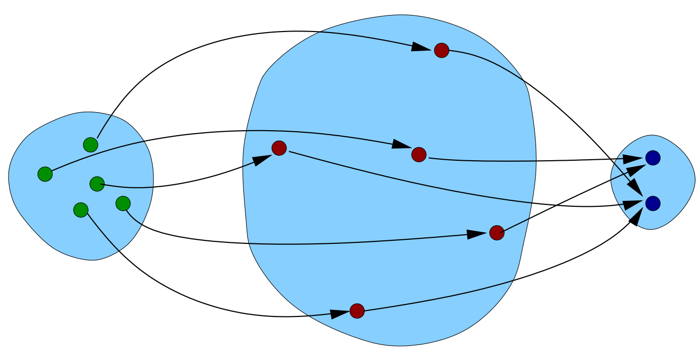
Kernel Methods

1. Kernels
   - Bypassing High-Dimensional Computations
   - Re-Formulation of the Minimization Task
2. Support Vector Machines
   - Classification with Minimal Risk
   - Slack Variables
   - Function Approximation

1. Kernels
   - Bypassing High-Dimensional Computations
   - Re-Formulation of the Minimization Task
2. Support Vector Machines
   - Classification with Minimal Risk
   - Slack Variables
   - Function Approximation

Transform input data non-linearly into a high-dimensional feature space



Idea behind Kernels

Utilize the advantages of a high-dimensional space without actually representing anything high-dimensional

- Condition: The only operation done in the high-dimensional space is to compute *scalar products* between pairs of items
- Common in ANN
- Trick: The scalar product is computed using the original (low-dimensional) representation

Example

Points in 2D

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Transformation to 4D

$$\phi(\vec{x}) = \begin{bmatrix} x_1^3 \\ \sqrt{3}x_1^2x_2 \\ \sqrt{3}x_1x_2^2 \\ x_2^3 \end{bmatrix}$$

$$\begin{aligned} \phi(\vec{x})^T \cdot \phi(\vec{y}) &= x_1^3y_1^3 + 3x_1^2y_1^2x_2y_2 + 3x_1y_1x_2^2y_2^2 + x_2^3y_2^3 \\ &= (x_1y_1 + x_2y_2)^3 \\ &= (\vec{x}^T \cdot \vec{y})^3 \\ &= \mathcal{K}(\vec{x}, \vec{y}) \end{aligned}$$

Common Kernels

Polynomials

$$\mathcal{K}(\vec{x}, \vec{y}) = (\vec{x}^T\vec{y} + 1)^p$$

Radial Bases

$$\mathcal{K}(\vec{x}, \vec{y}) = e^{\frac{1}{2\rho^2}||\vec{x}-\vec{y}||^2}$$

Structural Risk Minimization

Minimize

$$\vec{w}^T\vec{w}$$

Constraints

$$t_i(\vec{w}^T\vec{x}_i + b) \geq 1 \qquad \forall i$$

- Include b in the weight vector
- Non-linear transformation ϕ of input $\vec{x}$

New formulation

Minimize

$$\frac{1}{2}\vec{w}^T\vec{w}$$

Constraints

$$t_i\vec{w}^T\phi(\vec{x}_i) \geq 1 \qquad \forall i$$

Structural Risk Minimization

Minimize

$$\frac{1}{2}\vec{w}^T\vec{w}$$

Constraints

$$t_i\vec{w}^T\phi(\vec{x}_i) \geq 1 \qquad \forall i$$

Lagranges Multiplier Method

$$L = \frac{1}{2}\vec{w}^T\vec{w} - \sum_i \alpha_i \left[t_i\vec{w}^T\phi(\vec{x}_i) - 1\right]$$

Minimized w.r.t. $\vec{w}$, maximize w.r.t. $\alpha_i \geq 0$

$$\frac{\partial L}{\partial \vec{w}} = 0$$

$$L = \frac{1}{2}\vec{w}^T\vec{w} - \sum_i \alpha_i \left[t_i\vec{w}^T\phi(\vec{x}_i) - 1\right]$$

$$\frac{\partial L}{\partial \vec{w}} = 0 \implies \vec{w} - \sum_i \alpha_i t_i \phi(\vec{x}_i) = 0$$

$$\vec{w} = \sum_i \alpha_i t_i \phi(\vec{x}_i)$$

Use

$$\vec{w} = \sum_i \alpha_i t_i \phi(\vec{x}_i)$$

to eliminate $\vec{w}$

$$L = \frac{1}{2}\vec{w}^T\vec{w} - \sum_i \alpha_i \left[t_i\vec{w}^T\phi(\vec{x}_i) - 1\right]$$

$$L = \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j t_i t_j \phi(\vec{x}_i)^T\phi(\vec{x}_j) - \sum_{i,j} \alpha_i\alpha_j t_i t_j \phi(\vec{x}_i)^T\phi(\vec{x}_j) + \sum_i \alpha_i$$

$$L = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j t_i t_j \phi(\vec{x}_i)^T\phi(\vec{x}_j)$$

The Dual Problem

Maximize

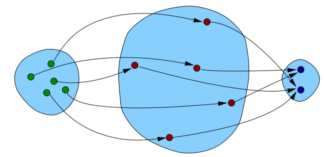
$$\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j t_i t_j \phi(\vec{x}_i)^T\phi(\vec{x}_j)$$

Under the constraints

$$\alpha_i \geq 0 \quad \forall i$$

- $\vec{w}$ has disappeared
- $\phi(\vec{x})$ only appear in scalar product pairs

1. Kernels
   - Bypassing High-Dimensional Computations
   - Re-Formulation of the Minimization Task
2. Support Vector Machines
   - Classification with Minimal Risk
   - Slack Variables
   - Function Approximation

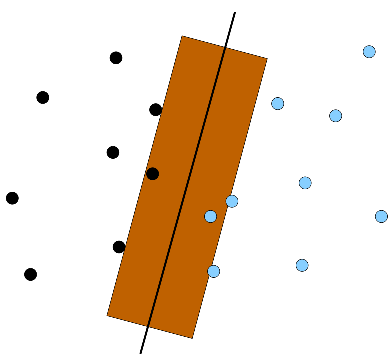


1. Choose a suitable kernel function
2. Compute α_i (solve the maximization problem)
3. $\vec{x}_i$ corresponding to $\alpha_i \neq 0$ are called support vectors
4. Classify new data points via

$$\sum_i \alpha_i t_i \mathcal{K}(\vec{x}, \vec{x}_i) > 0$$

None-Separable Training Samples

Allow for Slack



Re-formulation of the minimization problem

Minimize

$$\frac{1}{2}\vec{w}^T\vec{w} + C\sum_i \xi_i$$

Constraints

$$t_i \vec{w}^T \phi(\vec{x}_i) \geq 1 - \xi_i$$

ξ_i are called *slack variables*

Dual Formulation with Slack

Maximize

$$\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i\alpha_j t_i t_j \phi(\vec{x}_i)^T \phi(\vec{x}_j)$$

With constraints

$$0 \leq \alpha_i \leq C \quad \forall i$$

Otherwise, everything remains as before

Support vector methods can also be used for function approximation