A brief introduction to Support Vector Machines

McElory Hoffmann

Department of Mathematical Sciences (Applied Mathematics)
University of Stellenbosch

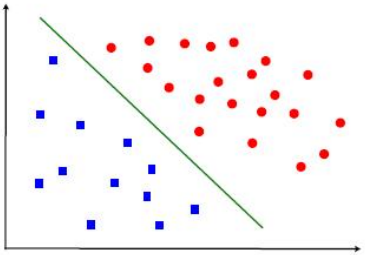
May 2006

Outline

Uses of SVM's

$$Y = \mathrm{sgn}[\mathbf{w}^\top \mathbf{x} + b > 0] = \begin{cases} +1 & \mathbf{w}^\top \mathbf{x} + b > 0 \\ -1 & \mathbf{w}^\top \mathbf{x} + b \leq 0 \end{cases}$$

- Supervised Learning
  - Pattern recognition
  - Regression and time series
- Unsupervised Learning
  - Dimensionality Reduction (Non-linear PCA)
  - Clustering
  - Novelty detection



[Graphics taken from [Meir R., 2002]]

An introductory example

[Example taken from [Schölkopf B., 2000].]

- Empirical data $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{X} \times \{\pm 1\}$ is given.
- Given some new pattern $x \in \mathcal{X}$, predict $y \in \{\pm 1\}$.
- Choose y such that (x, y) is in some sense similar to training examples.
- Similarity measure:
  - For outputs: Use loss function.
  - For inputs: kernel.

Similarity of inputs

- Symmetric function:

$$k : \mathcal{X} \times \mathcal{X} \to \Re$$
$$(x, x') \mapsto k(x, x').$$

- Important example of k is dot product. If, $\mathcal{X} \in \Re^N$, dot product is defined as

$$(\boldsymbol{x} \cdot \boldsymbol{x}') \triangleq \sum_i (\boldsymbol{x})_i (\boldsymbol{x}')_i.$$

$(\boldsymbol{x})_i$ is the i'th entry of $\boldsymbol{x}$.

Similarity of inputs (ctd)

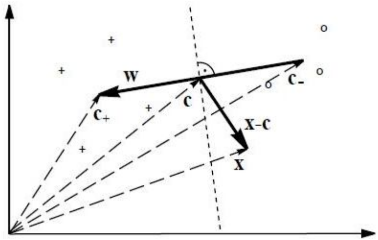
- If $\mathcal{X}$ is not dot product space, assume existence of map $\Phi : \mathcal{X} \rightarrow \mathcal{H}$ such that

$$k(x, x') \triangleq (\boldsymbol{x} \cdot \boldsymbol{x}') = \left(\Phi(x), \Phi(x')\right) .$$

- In this case, we can deal with patterns geometrically.
- Choice of mapping Φ enable large variety of learning algorithms.

Kernel algorithm

- Classify points $\boldsymbol{x} = \Phi(x)$ in feature space according to which of the two class means is closer.
- The means are: $\boldsymbol{c}_+ = \frac{1}{m_+}\sum_{y_i=+1} \boldsymbol{x}_i$ and $\boldsymbol{c}_- = \frac{1}{m_-}\sum_{y_i=-1} \boldsymbol{x}_i$.



- Compute the sign of the dot product between $\boldsymbol{w} \triangleq \boldsymbol{c}_+ - \boldsymbol{c}_-$ and $\boldsymbol{x} - \boldsymbol{c}$.

[Graphics taken from [Schölkopf, B. et al.]]

Kernel algorithm (ctd)

- We have:

$$
\begin{aligned}
y &= \operatorname{sgn}\left((\boldsymbol{x}-\boldsymbol{c})\cdot\boldsymbol{w}\right)\\
&= \operatorname{sgn}\left(\left(\boldsymbol{x}-\frac{\boldsymbol{c}_+ + \boldsymbol{c}_-}{2}\right)\cdot(\boldsymbol{c}_+ - \boldsymbol{c}_-)\right)\\
&= \operatorname{sgn}\left((\boldsymbol{x}\cdot\boldsymbol{c}_+) - (\boldsymbol{x}\cdot\boldsymbol{c}_-) + b\right)\\
&= \operatorname{sgn}\left(\frac{1}{m_+}\sum_{y_i=+1}(\boldsymbol{x}\cdot\boldsymbol{x}_i) - \frac{1}{m_-}\sum_{y_i=-1}(\boldsymbol{x}\cdot\boldsymbol{x}_i) + b\right)\\
&= \operatorname{sgn}\left(\frac{1}{m_+}\sum_{y_i=+1}k(x,x_i) - \frac{1}{m_-}\sum_{y_i=-1}k(x,x_i) + b\right)
\end{aligned}
$$

Kernel algorithm (ctd)

- Similarities with ”more advanced algorithms”:
  - Linear in feature space.
  - Example based: kernels centered on training examples.
- Differences with ”more advanced algorithms”:
  - Selection of the examples that the kernels are centered on.
  - Weights of the individual kernels in the decision function.

Learning from Examples

- Estimate function $f : \mathcal{X} \to \{\pm 1\}$ based on training data.
- Assume data was generated independently from unknown (but fixed) prob. dist. $P(x, y)$.
- Empirical risk or training error given by

$$R_{emp}[f] = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} |f(x_i) - y_i|.$$

- Test error or risk is

$$R[f] = \int \frac{1}{2} |f(x) - y| dP(x, y).$$

- Small training error does not imply small test error!
- Objective: Find 'good' function f which generalise.

Vapnik-Chervonenkis (VC) theory

- With a probability of at least $1-\eta$, the bound

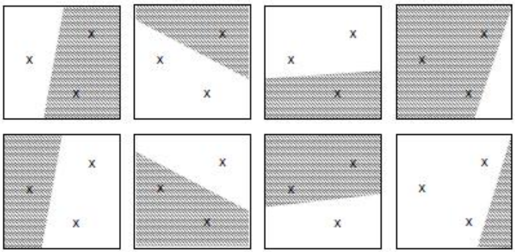
$$R(\alpha) \leq R_{emp}(\alpha) + \phi\left(\frac{h}{m}, \frac{\log(\eta)}{m}\right)$$

  holds.
- Confidence term ϕ is defined as

$$\phi\left(\frac{h}{m}, \frac{\log(\eta)}{m}\right) = \sqrt{\frac{h(\log\frac{2m}{h}+1) - \log(\frac{\eta}{4})}{m}}.$$

- h is the VC dimension, m the number of training samples and $h < m$.
- No dimension of data term!

VC Dimension



[Graphics taken from [Schölkopf, B. et al.]]

Restriction of capacity

- Consider the class of hyperplanes

$$(\boldsymbol{w} \cdot \boldsymbol{x}) + b = 0 \quad \boldsymbol{w} \in R^N, b \in R.$$

- Corresponding decision functions $f(\boldsymbol{x}) = \text{sgn}((\boldsymbol{w} \cdot \boldsymbol{x}) + b)$.
- Vapnik 95: If $||\boldsymbol{w}|| \leq A$ and $\boldsymbol{x}_i \in$ Ball of radius L, then $h \leq \min(A^2 L^2, d) + 1$.
  - Observe: If we restrict our function class, the capacity (e.g. VC dimension) is smaller.
  - Suggestion: Use hyperplane with minimal norm.

The separable problem

- Solve the optimisation problem:

$$\begin{aligned} \text{minimise} \quad & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} \\ \text{subject to} \quad & y_i\left(\boldsymbol{w}\cdot\boldsymbol{x}_i + b\right) \geq 1, i = 1, \ldots m. \end{aligned}$$

- Constrained optimisation problem. Is solved using Lagrange multipliers (α_i).
- Will skip detail of optimisation theory until another time!

Construction of Optimal Hyperplane in Dual Space

- By employing standard optimisation theory, we can rewrite the optimisation problem in dual space.
- This yields:

$$\text{maximise} \quad W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i \boldsymbol{x}_j^T$$

$$\text{subject to} \quad \sum_{i=1}^{m} \alpha_i y_i = 0; \alpha_i \geq 0.$$

- The hyperplane decision function can thus be written as

$$f(\boldsymbol{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} y_i \alpha_i \cdot (\boldsymbol{x} \cdot \boldsymbol{x}_i) + b \right).$$

- Note that we use only dot-products when working with $\boldsymbol{x}$.

Graphic illustration of separable problem

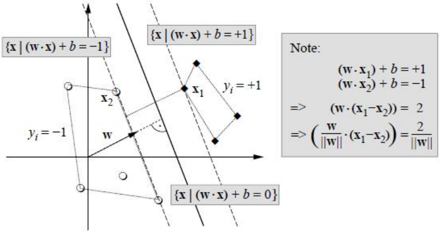


Figure 1: A binary classification toy problem: separate balls from diamonds. The *optimal hyperplane* is orthogonal to the shortest line connecting the convex hulls of the two classes (dotted), and intersects it half-way between the two classes. The problem being separable, there exists a weight vector $\mathbf{w}$ and a threshold b such that $y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) > 0$ $(i = 1, \ldots, m)$. Rescaling $\mathbf{w}$ and b such that the point(s) closest to the hyperplane satisfy $|(\mathbf{w} \cdot \mathbf{x}_i) + b| = 1$, we obtain a *canonical* form $(\mathbf{w}, b)$ of the hyperplane, satisfying $y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1$. Note that in this case, the *margin*, measured perpendicularly to the hyperplane, equals $2/\|\mathbf{w}\|$. This can be seen by considering two points $\mathbf{x}_1, \mathbf{x}_2$ on opposite sides of the margin, i.e. $(\mathbf{w} \cdot \mathbf{x}_1) + b = 1$, $(\mathbf{w} \cdot \mathbf{x}_2) + b = -1$, and projecting them onto the hyperplane normal vector $\mathbf{w}/\|\mathbf{w}\|$.

[Graphics taken from [Schölkopf B., 2000]]

Non-separable case

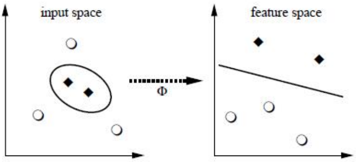
- Problem: Cannot satisfy $y_i\left(\boldsymbol{w} \cdot \boldsymbol{x}_i + b\right) \geq 1\ \forall i$.
- Solution: Introduce slack variables ξ_i.
- Optimisation problem is now:

$$
\begin{aligned}
\text{minimise} \quad & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{m}\xi_i^2 \\
\text{subject to} \quad & y_i\left(\boldsymbol{w} \cdot \boldsymbol{x}_i + b\right) \geq 1 - \xi_i, i = 1, \ldots m \\
& \xi_i \geq 0, i = 1, \ldots, m.
\end{aligned}
$$

- There exist bound for this problem.
- New parameter C is found using cross-validation.

Motivation

- Linear separability is more likely in high dimensions.
- Map input into high (possibly infinite) dimensional feature space Φ.
- Construct linear classifier in Φ.



[Graphics taken from [Schölkopf B., 2000]]

Kernel functions

- Using the ideas mentioned, classifier is

$$f(\boldsymbol{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} y_i\alpha_i \cdot (\Phi(x) \cdot \Phi(x_i)) + b\right)$$
$$= \operatorname{sgn}\left(\sum_{i=1}^{m} y_i\alpha_i \cdot k(x, x_i) + b\right).$$

- The dual quadratic problem is

$$\text{maximise} \quad W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i\alpha_j y_i y_j k(x_i, x_j)$$

$$\text{subject to} \quad \sum_{i=1}^{m} \alpha_i y_i = 0; \alpha_i \geq 0.$$

The kernel trick

- Any algorithm that only depends on dot products can benefit from the kernel trick.
- This way, we can apply linear methods to vectorial as well as non-vectorial data.
- Think of the kernel as a nonlinear similarity measure.
- Examples of kernels:
  - Polynomial: $k(x, x') = (\langle x, x' \rangle + c)^d$.
  - Sigmoid: $k(x, x') = \tanh(\kappa \langle x, x' \rangle + \Theta)$.
  - Gaussian: $k(x, x') = \exp(-\frac{||x - x'||^2}{2\sigma^2})$.

SHOW DEMO!

For Further Reading

- Schölkopf, B. (2000).
  Statistical Learning and Kernel Methods.
  Microsoft Research Technical Report, MSR-TR-2000-23
- Schölkopf, B., Rasmussen, C., Franz, M. (2005)
  Lecture notes of Learning in Computer Vision II (2005).
  Max-Planck-Institut für biologische Kybernetik,
  http://www2.tuebingen.mpg.de/agbs/lcvii/wiki/LearningInComputerVisionII

For Further Reading (ctd)

- Meir, R. (2002).
  Support Vector Machines - an Introduction.
  In *Department of Electrical Engineering, Technion, Isreal*,
  `www.ee.technion.ac.il/ȓmeir/SVMReview.pdf`
- Christianini N., Shawe-Taylor, J. (2000)
  An Introduction to Support Vector Machines.
  Cambridge University Press, ISBN 0 521 78019 5

For Further Reading (ctd)

- Kienzle, W., Bakir, G., Franz, M. Schölkopf, B. (2005)
  Face Detection Demo
  http://www.kyb.tuebingen.mpg.de/bs/people/kienzle/facedemo/facedemo.htm
- Schölkopf, B., Smola, A. (2002)
  Learning with Kernels
  http://www.learning-with-kernels.org/