Support vector machines
Support vector machines

- When the data is linearly separable, there may be more than one separator (hyperplane).

Which separator is best?
Support vector machines

- Find hyperplane that maximizes the *margin* between the positive and negative examples

\[
\begin{align*}
\text{x positive (} y = 1 \text{)}: & \quad x \cdot w + b \geq 1 \\
\text{x negative (} y = -1 \text{)}: & \quad x \cdot w + b \leq -1
\end{align*}
\]

For support vectors, \( x \cdot w + b = \pm 1 \)

Distance between point and hyperplane:

\[
\frac{|x \cdot w + b|}{||w||}
\]

Therefore, the margin is \( \frac{2}{||w||} \)

Finding the maximum margin hyperplane

1. Maximize margin $\frac{2}{\|w\|}$

2. Correctly classify all training data:
   - $x_i$ positive ($y_i = 1$): $x_i \cdot w + b \geq 1$
   - $x_i$ negative ($y_i = -1$): $x_i \cdot w + b \leq -1$

**Quadratic optimization problem:**

$$\min_{w,b} \frac{1}{2} \|w\|^2 \quad \text{subject to} \quad y_i (w \cdot x_i + b) \geq 1$$

SVM training in general

- Separable data: \[ \min_{w,b} \frac{1}{2} \|w\|^2 \text{ subject to } y_i(w \cdot x_i + b) \geq 1 \]

Maximize margin
Classify training data correctly

- Non-separable data:

\[ \min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \max(0, 1 - y_i(w \cdot x_i + b)) \]

Maximize margin
Minimize classification mistakes
SVM training in general

\[
\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \max \left(0, 1 - y_i (w \cdot x_i + b)\right)
\]
Nonlinear SVMs

• General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable.
Nonlinear SVMs

• Linearly separable dataset in 1D:

• Non-separable dataset in 1D:

• We can map the data to a higher-dimensional space:

Slide credit: Andrew Moore
The kernel trick

• Linear SVM decision function:

$$w \cdot x + b = \sum_i \alpha_i y_i x_i \cdot x + b$$

The kernel trick

• Linear SVM decision function:

\[ \mathbf{w} \cdot \mathbf{x} + b = \sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b \]

• Kernel SVM decision function:

\[ \sum_i \alpha_i y_i \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}) + b = \sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \]

• This gives a nonlinear decision boundary in the original feature space

The kernel trick

- Instead of explicitly computing the lifting transformation \( \varphi(x) \), define a kernel function \( K \) such that

\[
K(x, y) = \varphi(x) \cdot \varphi(y)
\]

(to be valid, the kernel function must satisfy Mercer’s condition)
Polynomial kernel: $K(x, y) = (c + x \cdot y)^d$
Gaussian kernel

- Also known as the radial basis function (RBF) kernel:

\[
K(x, y) = \exp\left(-\frac{1}{\sigma^2} \|x - y\|^2\right)
\]
Gaussian kernel

Demo: http://cs.stanford.edu/people/karpathy/svmjs/demo
SVMs: Pros and cons

• Pros
  • Kernel-based framework is very powerful, flexible
  • Training is convex optimization, globally optimal solution can be found
  • Amenable to theoretical analysis
  • SVMs work very well in practice, even with very small training sample sizes

• Cons
  • No “direct” multi-class SVM, must combine two-class SVMs (e.g., with one-vs-others)
  • Computation, memory (esp. for nonlinear SVMs)