SUPPORT VECTOR MACHINES

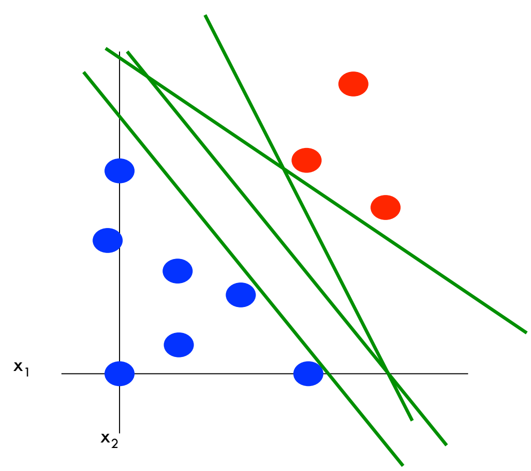
Today

- Support vector machines
- Training and Testing
- Modifications and improvements

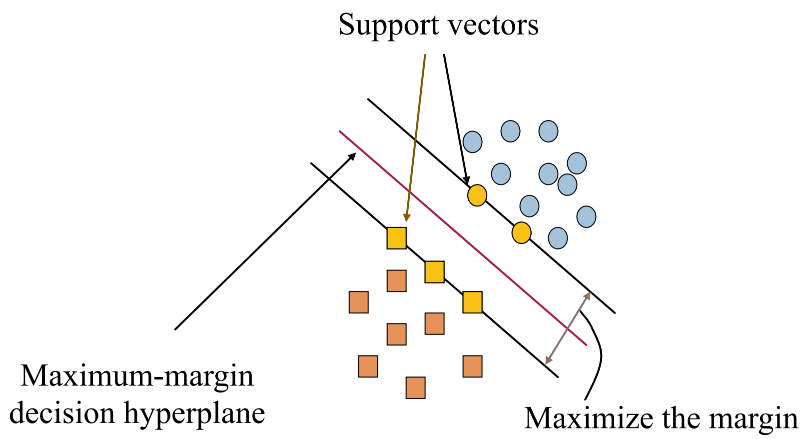
Support Vector Machines (SVMs)

- SVMs are a popular classifiers
- SVMs are linear classifiers
  - Kernels allow for non-linear classification
- Software Packages
  - LIBSVM (LIBLINEAR) – on the Resources page
  - SVM-Light

Which is the best decision boundary?



Support Vector Machines



What defines a hyperplane?

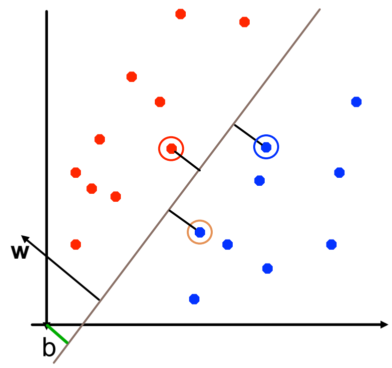
Classify a new instance (prediction)

$$D = \{(x_i, y_i) | i = 1 \ldots N\}$$

$$y_i \in \{-1, 1\}$$

$w^\intercal x + b = 0$ x on the decision boundary
$w^\intercal x + b < 0$ x "below" the decision boundary
$w^\intercal x + b > 0$ x "above" the decision boundary

$$h(x) = \text{sign}(w \cdot x + b)$$



Learning (Training)

Learning (Training)

Solving the Optimization Problem

$$\min_{w,b} \quad \frac{1}{2}||w||^2 \quad \text{s.t. } y_j\big(x_j \cdot w + b\big) \geq 1 \ \forall j$$

- Need to optimize a *quadratic* function subject to *linear* constraints
- The solution involves constructing a *dual problem* where a *Lagrange multiplier* (a scalar) is associated with every constraint in the primary problem

Solving the Optimization Problem

$$\min_{w,b} \frac{1}{2}||w||^2 \text{ such that } y^{(i)}\big(w^\intercal x^{(i)} + b\big) \geq 1 \quad \forall i$$

$$\left.\max_{\alpha} \min_{w,b} \frac{1}{2}||w||^2 - \sum_{i=1}^{N} \alpha_i \big[y_i(w \cdot x_i + b) - 1\big]\right\} \text{Dual}$$

Lagrange multipliers

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j \, (x_i \cdot x_j)$$

$$\text{subject to } \alpha_i \geq 0 \text{ and } \sum_i \alpha_i y_i = 0$$

Solving the Optimization Problem

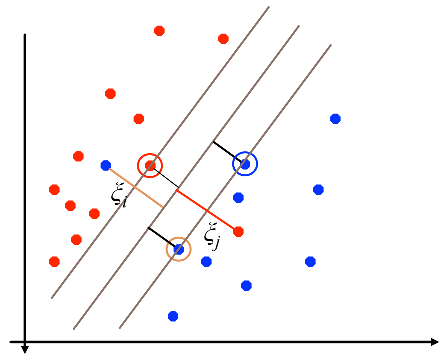
- The solution has the form:

$$w = \sum_{i=1}^{N} \alpha_i y_i \, x_i \text{ and } b = y_i - w \cdot x_i \text{ for any } x_i \text{ s.t. } \alpha_i \neq 0$$

- Each non-zero alpha indicates corresponding x_i is a *support vector*
- The classifying function has the form: $h(x) = \text{sign}\left(\sum_i \alpha_i y_i (x_i \cdot x) + b\right)$
- Relies on an dot product between the test point x and the support vectors x_i

Soft-margin Classification

- *slack variables* ξ_i can be added to allow misclassification of difficult or noisy examples.
- Still, try to minimize training set errors, and to place hyperplane "far" from each class (large margin)



How many support vectors?

- Determined by alphas in optimization
- Typically only a small proportion of the training data
- The number of support vectors determines the run time for prediction

How fast are SVMs?

Training

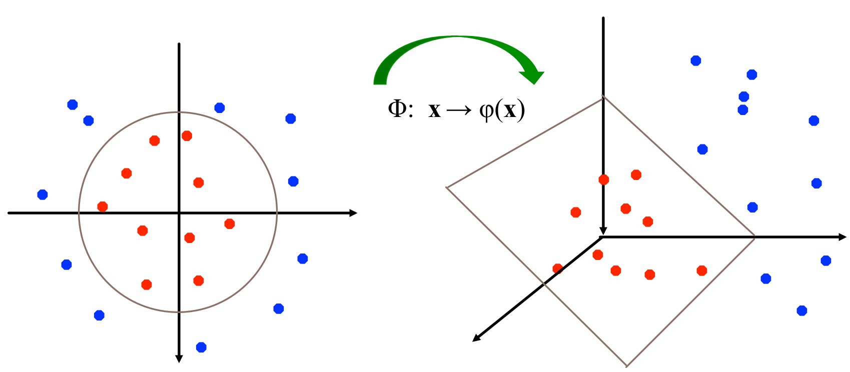
- Time for training is dominated by the time for solving the underlying quadratic programming problem
- Slower than Naïve Bayes
- Non-linear SVMs are worse

Testing (Prediction)

- Fast - as long as we don't have too many support vectors

Non-linear SVMs

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



The “Kernel” trick

The “Kernel” trick

The “Kernel” trick

Kernels

Why use kernels?

- Make non-separable problem separable.
- Map data into better representational space

Common kernels

- Linear
- Polynomial $\mathbf{K(x,z) = (1+x^Tz)^d}$
- Radial basis function (infinite dimensional space)

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2}$$

Summary

- Support Vector Machines (SVMs)
  - Find the maximum margin hyperplane
  - Only the support vectors needed to determine hyperplane
  - Use slack variables to allow some error
  - Use a kernel function to make non-separable data separable
  - Often among the best performing classifiers