STA 414/2104

Statistical Methods for Machine Learning and Data Mining

Radford M. Neal, University of Toronto, 2012

Week 9

Support Vector Machines

Another Way to Find the Hyperplane with Largest Margin

Recall that we if define a hyperplane by the equation $w^T x + b = 0$, we can find the maximum margin hyperplane by solving the following optimization problem:

$$\text{minimize } ||w||^2, \quad \text{subject to } y_i(w^T x_i + b) \geq 1 \text{ for } i = 1, \ldots, n$$

We can always write

$$w \;=\; \sum_{i=1}^{n} a_i x_i \;+\; \delta$$

where $\delta^T x_i = 0$ for all $i = 1, \ldots, n$, for some (not necessarily unique) set of a_i. With this representation of w,

$$||w||^2 \;=\; \Big(\sum_{i=1}^{n} a_i x_i + \delta\Big)^T \Big(\sum_{i'=1}^{n} a_{i'} x_{i'} + \delta\Big) \;=\; \sum_{i=1}^{n}\sum_{i'=1}^{n} a_i a_{i'} (x_i^T x_{i'}) \;+\; ||\delta||^2$$

and

$$y_i(w^T x_i + b) \;=\; y_i\Big(\sum_{i'=1}^{n} a_{i'} (x_i^T x_{i'}) \;+\; b\Big)$$

Since the constraints don't depend on δ, the minimization will set $\delta = 0$, so we can assume that $w = \sum_{i=1}^{n} a_i x_i$.

Another Way to Find the Hyperplane... (Continued)

So we see that we can find $w = \sum_{i=1}^{n} a_i x_i$ and b as follows:

$$\text{minimize} \sum_{i=1}^{n} \sum_{i'=1}^{n} a_i a_{i'} (x_i^T x_{i'}),$$

$$\text{subject to } y_i \Big(\sum_{i'=1}^{n} a_{i'} (x_i^T x_{i'}) + b \Big) \geq 1 \text{ for } i = 1, \ldots, n$$

This is also a quadratic programming problem — minimize a quadratic function of the a_i subject to linear constraints on the a_i and b — which could be solved by standard (and fairly efficient) methods.

However, the solution may not be unique (though the resulting w is). If the problem is formulated a bit differently, the result can be made unique, and often many of the a_i will be zero (with non-zero a_i only for the support vectors).

The formulation above does show one crucial property — the minimization depends only on inner products of input vectors (ie, on $x_i^T x_{i'}$). Predictions for test cases also depend only on such inner products, since we will classify x_* according to the sign of $w^T x_* + b = \sum_{i=1}^{n} a_i (x_*^T x_i) + b$.

Large Margin Classifiers Using Basis Functions

Rather than find a large margin classifier based on the original input vector, x, we can use a vector of basis function values, $\phi(x) = [\phi_1(x)\ \phi_2(x)\ \cdots\ \phi_m(x)]^T$.

The classes may be separable by a hyperplane in this space even if they aren't in the original space.

Finding $a_1, \ldots, a_n$ and b can be done as before, using inner products, $\phi(x_i)^T \phi(x_{i'})$.

A test case with input vector x_* is classified by the sign of $\sum_{i=1}^{n} a_i \left(\phi(x_*)^T \phi(x_i)\right) + b$.

Since all that matters are these inner products, we can define

$$K(x, x') \quad = \quad \phi(x)^T \phi(x') \quad = \quad \sum_{j=1}^{m} \phi_j(x) \phi_j(x')$$

and then look at $K(x_i, x_{i'})$ for training cases i and i', and $K(x_*, x_i)$ for a test case.

So once we have a formula for $K(x, x')$, we can forget about the ϕ functions.

Classification (and regression) methods based on this "kernel trick" are known as *Support Vector Machines* (abbreviated to "SVM").

Letting the Number of Basis Functions Go to Infinity

Since all we need is a formula for the "kernel function",

$$K(x, x') = \sum_{j=1}^{m} \phi_j(x)\phi_j(x')$$

we can consider letting the number of basis functions, m, go to infinity, as long the resulting infinite sum has a finite limit, and can be computed efficiently.

This is essentially identical to what we did earlier for Gaussian process models. The noise-free covariance function corresponding to a Bayesian linear basis function model with independent zero-mean normal priors for coefficients, with the variance of the coefficient for ϕ_j being ω_j^2, was found to be

$$K(x, x') = \sum_{j=0}^{m-1} \omega_j^2 \phi_j(x)\phi_j(x')$$

This becomes the same as above if we absorb a factor ω_j into the definition of ϕ_j (and replace 0 to $m-1$ with 1 to m).

Possible Kernel Functions

The possible kernel functions for a support vector machine are the same as the possible covariance functions for a Gaussian process model — all those that produce positive semi-definite matrices at any set of points.

Mercer's Theorem says that all such positive definite kernels can be represented in the form $K(x, x') = \sum \phi_j(x)\phi_j(x')$, though sometimes all but a finite number of the ϕ_j will be identically zero.

So the class of models defined using linear basis functions is the same as the class of models defined using a kernel/covariance function.

Commonly used kernel functions include $K(x, x') = (1 + x^T x')^d$, corresponding to polynomial basis functions to degree d, and $K(x, x') = \exp(-\rho^2 ||x - x'||^2)$.

Note that for an SVM (unlike for a Gaussian process), multiplying the kernel function by a positive constant does not change things.

More Elaborations on Support Vector Machines

- Which kernel function is best is usually not clear. Cross validation can be used to choose one.
- Finding a separating hyperplane (even if always possible in an infinite dimensional space) may not be a good idea, when class labels are actually "noisy". Introducing "slack variables" allows for some mis-classified points.
- Classification problems with more than two classes can be handled in various ways — eg, combining results from pairwise binary classifiers.
- Regression problems can be handled by using a "loss" function that is "ϵ-insensitive" — where small errors cost zero.

Support Vector Machines vs. Gaussian Process Models

SVM and GP models have a strong common element — the positive semi-definite kernel/covariance function. How do they compare otherwise?

Advantages of support vector machines:

- The number of support vectors is often much less than the total size of the training set, reducing computation time for training and prediction.
- Binary classification can be done directly, with a relatively fast optimization procedure, whereas Gaussian process classification requires handling a distribution over "latent variables".

Advantages of Gaussian process models:

- The covariance function has a probabilistic interpretation — one can sample from the prior over functions that it defines — which can guide the choice of a suitable covariance function.
- Finding good parameters of the covariance function can be done reasonably efficiently by maximum likelihood (or by Bayesian methods), without the need for cross validation.
- Classification problems with more than two classes can be handled naturally.