Support Vector Machines

Charlotte Wickham

May 10, 2007

Basic Set Up

We have some labeled data from two classes,

$$E = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\},$$

where $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$.

Our goal is to come up with a function that will classify new unlabeled data into one of the two classes.

Separating Hyperplanes

Imagine two linearly separable classes. There are many possible hyperplanes that will separate the groups.
Can define a hyperplane by:

$$\{x : f(x) = \beta_0 + \beta^T x = 0\}$$

- If we can find β_0 and β for such a separating hyperplane then we can classify a new point, x', by $\text{sign}(f(x'))$.
- For linearly separable classes there are many possible separating hyperplanes. How can we choose one?

Optimal Separating Hyperplane

- Choose an optimal criterion for a hyperplane
- Maximise the minimal distance from any point to the boundary
- Maximise the margin
- Can show: The distance from any point, x, to the hyperplane is proportional to $f(x)$ (equal if $||\beta|| = 1$).
- So we can write the problem as

$$\max_{\beta, \beta_0, ||\beta||=1} C$$

subject to $\mathrm{y}_i(x_i^T \beta + \beta_0) \geq C, \quad i = 1, \ldots, N$

Finding the Optimal Separating Hyperplane

Using tricks from convex optimization the problem can be rephrased as maximising

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

subject to constraints $\quad \alpha_i \geq 0$
$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$
$0 = \sum_{i=1}^{N} \alpha_i y_i$
$\alpha_i (y_i (x_i^T \beta + \beta_0) - 1) = 0 \quad \forall i$

So, if $\alpha_i > 0$ then $y_i (x_i^T \beta + \beta_0) = 1$, and x_i is called a support vector. Also β is calculated only using these support vectors.

Extending the Optimal Separating Hyperplane

- In real data classes aren't generally separable.
- How can we deal with classification error?
- How can we have non linear classification boundaries?
- The support vector machine deals with both of these problems

Support Vector Machines

Classification Error

- Relax the constraint to

$$y_i(x_i^T\beta + \beta_0) \geq C(1 - \epsilon_i),$$

  i = 1,. . . , N, $\quad \epsilon_i \geq 0 \quad \sum_{i=1}^{N} \epsilon_i \leq K$

- Allows points to be on the "wrong" side of the margin.
- Misclassified if $\epsilon_i > 1$.
- Number of training misclassifications bounded by K.

Now some of the "support vectors" on the wrong side of the margin.

Support Vector Machines

Nonlinear Boundaries

- Project the original data into a higher dimensional space
- Hyperplanes in the higher dimensional space will be non linear boundaries in the original space
- Seems like it makes it a much harder problem to solve, but it doesn't, why?

Non linear boundaries

Use a mapping ϕ to take values in the original space into a higher dimensional space.

Turns out to find our hyperplane we don't need to know ϕ but just

$$k(x, w) = \phi(x)^T \phi(w)$$

the dot product in the higher dimensional space.

k(x,w) is called the kernel function.

Can think of it as a similarity function.

Some examples,

- linear $k(x, w) = x^T w$
- dth degree polynomial $k(x, w) = (1 + x^T w)^d$
- Radial basis $k(x, w) = exp(-||x - w||^2 \gamma)$

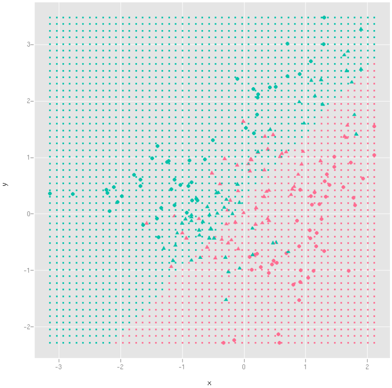
In Practice

- Need to choose parameters
  - Kernel Function
  - Parameters in kernel function
  - Tuning parameter (how much misclassification are we allowing?)
- Use cross validation to help choose.
- Number of support vectors can also be used to diagnose over fitting

Toy Example

Linear Kernels

Linear kernel C=1
Number of Support Vectors: 106
Total Accuracy: 78.5



Linear kernel C = 20
Number of Support Vectors: 102
Total Accuracy: 78.5

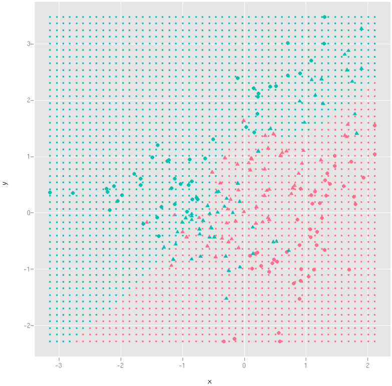
RBF Kernels

Radial kernel C=1, $\gamma = 0.5$
Number of Support Vectors: 94
Total Accuracy: 80.5

Radial kernel C=16, $\gamma = 4$
Number of Support Vectors: 94
Total Accuracy: 82.5

Polynomial Kernels

Radial kernel C=1, degree = 4
Number of Support Vectors: 116
Total Accuracy: 75



Radial kernel C=16, degree = 3
Number of Support Vectors: 19
Total Accuracy: 79

Chirstopher Bishop.
Machine Learning and Pattern Recognition.
Springer, 2006.

Doina Caragea, Dianne Cook, Hadley Wickham, and Vasant Honavar.
Visual methods for examining svm classifiers.
Technical report, 2006.

Pai-Hsuen Chen, Chih-Jen Lin, and Bernhard Schlkopf.
A tutorial on -support vector machines.
URL `citeseer.ist.psu.edu/605359.html`.

T. Hastie, R. Tibshirani, and J. Friedman.
The Elements of Statistical Learning.
Springer, 2001.

Marti A. Hearst.
Support vector machines.
IEEE Intelligent Systems, 13(4):18–28, 1998.
ISSN 1541-1672.
doi: http://dx.doi.org/10.1109/5254.708428.

Raquel A. Romano, Celcilia Aragorn, and Chris Ding.
Supernova recognition using support vector machines.
Technical report, 2006.

Computing the support vector classifier

Forget mapping into a higher dimension for a moment.
We need to solve the optimization problem

$$\min ||\beta||$$

subject to
$y_i(x_i^T\beta + \beta_0) \geq 1 - \epsilon_i, \quad \epsilon_i \geq 0$ and $\sum_{i=1}^N \epsilon_i \leq \textit{constant}$
(same as before letting $C = 1/||\beta||$).
Which is equivalent to solving the following

$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^N \epsilon_i$$

subject to $y_i(x_i^T\beta + \beta_0) \geq 1 - \epsilon_i, \quad \epsilon_i \geq 0$

Computing the support vector classifier...

Using tricks from convex optimization this is equivalent to

(1)