# SUPPORT VECTOR MACHINES

## What is a support vector?

In seperating clouds of points from two different classes, we use the surface

$$\mathbf{w}^T\mathbf{x} + b = 0$$

Lets take two nearest points from the two classes and use the construction

$$\mathbf{w}^T\mathbf{x_i} + b \geq 0 \text{ for } d_i = +1 \quad (1)$$
$$\mathbf{w}^T\mathbf{x_i} + b < 0 \text{ for } \ d_i = -1 \quad (2)$$

## We want to minimize the length of $\mathbf{w}$. Why?

Last week we found out that the perpendicular distance from the hyperplane to a vector $\mathbf{x}_s$ is

$$r = \frac{g(\mathbf{x}_s)}{||\mathbf{w}_o||}$$

Using Eqs. (1) and (2) we have

$$r = \frac{\pm 1}{||\mathbf{w}_o||}$$

Scaling

So that finally we see that we want to minimze the length of $\mathbf{w}$ since the seperating distance is

$$\frac{2}{||\mathbf{w}_o||}$$

**The problem statement**

Given a set of training data $\{(\mathbf{x}_i, d_i), i = 1, \ldots, N\}$, minimize

$$\Phi(\mathbf{w} = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to the constraint that

$$d_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1, i = 1, \ldots, N$$

**The problem statement**

Given a set of training data $\{(\mathbf{x}_i, d_i), i = 1, \ldots, N\}$, minimize

$$\Phi(\mathbf{w} = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to the constraint that

$$d_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1, i = 1, \ldots, N$$

**Looks like a job for LAGRANGE MULTIPLIERS!**

$$J(\mathbf{w}, b, \lambda) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \lambda_i(d_i(\mathbf{w}^T\mathbf{x_i} + b) - 1)$$

So that

$$J_{\mathbf{w}} = \mathbf{0} = \mathbf{w} - \sum_{i=1}^{N} \lambda_i d_i \mathbf{x_i} \tag{3}$$

and

$$J_b = 0 = \sum_{i=1}^{N} \lambda_i d_i \tag{4}$$

**Now for the DUAL PROBLEM**

$$J(\mathbf{w}, b, \lambda) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \lambda_i d_i \mathbf{w}^T\mathbf{x_i} + b\sum_{i=1}^{N} \lambda_i d_i + \sum_{i=1}^{N} \lambda_i$$

Note that from (4) third term is zero. Using Eq. (3):

$$Q(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j$$

Lets enjoy the moment!

**DUAL PROBLEM**

$$\max Q(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$$

Subject to constraints

$$\sum_{i=1}^{N} \lambda_i d_i = 0$$

$$\lambda_i \geq 0, \quad i = 1, \ldots, N$$

This is *easier to solve* than the original. Furthermore it only depends on the training samples $\{(\mathbf{x}_i, d_i), i = 1, \ldots, N\}$.

Once you have the $\lambda_i$s, get the $\mathbf{w}$ from

$$\mathbf{w} = \sum_{i=1}^{N} \lambda_i d_i \mathbf{x_i}$$

and the $b$ from a support vector that has $d_i = 1$,

$$b = 1 - \mathbf{w}^T \mathbf{x}_s$$

Almost done ...

## KERNEL FUNCTIONS

Now the big bonus occurs because all the machinery we have developed will work if we map the points $\mathbf{x}_i$ to a higher dimensional space, provided we observe certain conventions.

Let $\phi(\mathbf{x}_i)$ be a function that does the mapping. So the new hyperplane is

$$\sum_{i=1}^{N} w_i \phi_i(\mathbf{x}) + b = 0$$

For simplicity in notation define

$$\phi(\mathbf{x}) = (\phi_0(\mathbf{x}), \phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_{m_1}(\mathbf{x}))$$

where $m_1$ is the new dimension size and by convention $\phi_0(\mathbf{x}) = 1$.

Then all the work we did with $\mathbf{x}$ works with $\phi(\mathbf{x})$. The only issue is that instead of $\mathbf{x}_i^T \mathbf{x}_j$ we have a *Kernel function*, $K(\mathbf{x}_i, \mathbf{x}_j)$ where

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi_i(\mathbf{x})^T \phi_j(\mathbf{x})$$

and Kernel functions need to have certain nice properties. : )

### Examples

Polynomials

$$(\mathbf{x}_i^T \mathbf{x}_j + 1)^p$$

Radial Basis Functions

$$\exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_i||^2}{2\sigma^2}\right)$$