Machine Learning and Kernel Methods

Machine Learning

Primary references:
John Shawe-Taylor and Nello Cristianini, *Kernel Methods for Pattern Analysis*

Christopher Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2, 121–167 (1998).

Other references:
Aronszajn, Theory of reproducing kernels. Transactions of the American Mathematical Society, 686, 337-404, 1950.

Felipe Cucker and Steve Smale, On the mathematical foundations of learning. Bulletin of the American Mathematical Society, 2002.

Teo Evgeniou, Massimo Pontil and Tomaso Poggio, Regularization Networks and Support Vector Machines Advances in Computational Mathematics, 2000.

1. The problem: Learning theory

Given an unknown function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$, learn $f(\mathbf{x})$.

Example 1: $\mathbf{x}$ is retinal activation pattern (i.e., $x_i =$ activation level of retinal neuron i, and $y = f(\mathbf{x}) > 0$ if the retinal pattern is a chair; $y = f(\mathbf{x}) < 0$ otherwise.

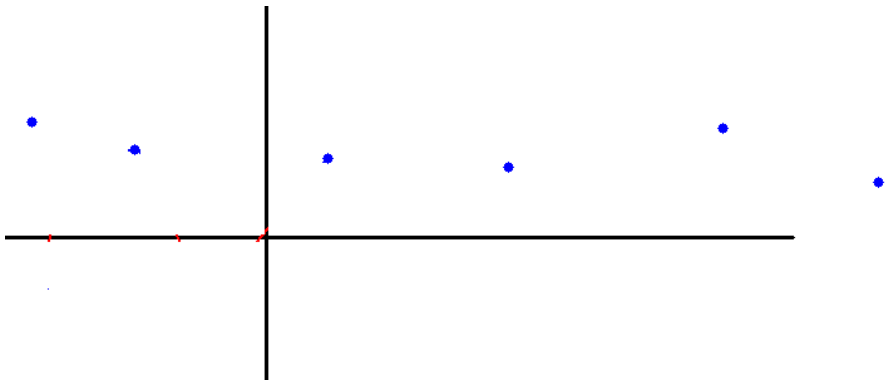
[Thus: want concept of a chair]

Given: examples of chairs (and non-chairs): $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, together with proper outputs $y_1, \ldots, y_n$. This is the information:

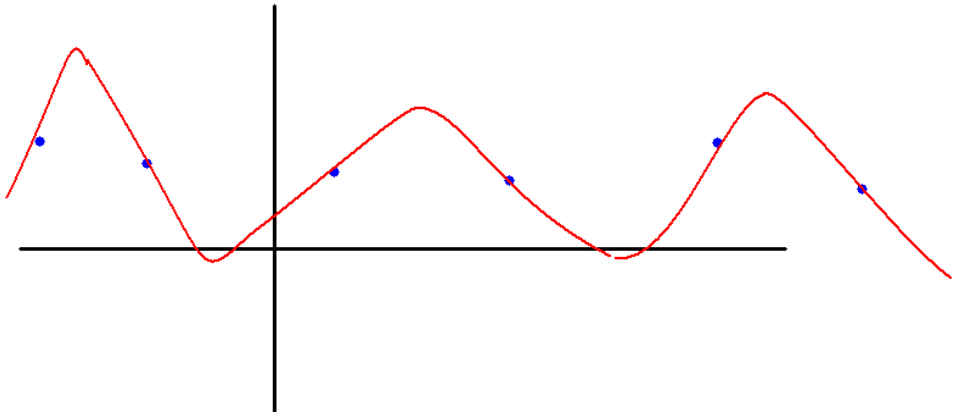
$$Nf = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))$$

Goal: Give best possible estimate of the unknown function f, i.e., try to learn the concept f from the examples Nf.

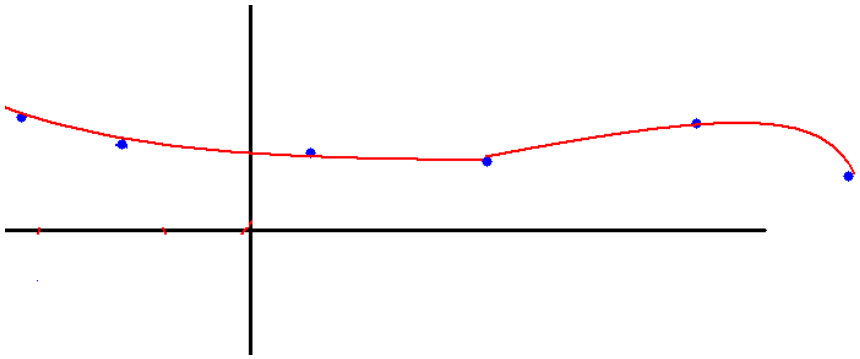
But: given pointwise information about f not sufficient: which is the "right" $f(x)$ given the data Nf below?



(a)



(b)



[How to decide?]

2. Regularization methods for choosing f

Finding f from Nf is an *ill-posed roblem*: the operator N^{-1} does not exist because N is not one to one.

Need to combine both:

(a) Data Nf
(b) A priori information, e.g., "f is smooth", e.g. expressing a preference for (b) over (a) above.

How to incorporate? Using *Tikhonov regularization methods.*

We introduce a *regularization loss functional* $L(f)$ representing penalty for choice of an "unrealistic" f such as that in (a) above.

Assume we want to find the correct function $f_0(\mathbf{x})$, from data

$$Nf_0(\mathbf{x}) = (f_0(\mathbf{x}_1), \ldots, f_0(\mathbf{x}_n)) = (y_1, \ldots, y_n)$$

Suppose we are given $f(x)$ as a candidate for approximating $f_0(\mathbf{x})$. We score f as a good or bad approximation based on a combination of its error on the known points $\{\mathbf{x}_i\}_{i=1}^n$, together with its "plausibility", i.e., how low

$$e = \frac{1}{n}\sum_{i=1}^{n} V(f(\mathbf{x}_i), y_i) + L(f)$$

is. Here $V(f(\mathbf{x}_i), y_i)$ is a measure of the loss whenever $f(\mathbf{x}_i) \neq y_i$, e.g.

$$V(f(\mathbf{x}_i), y_i) = |f(\mathbf{x}_i) - y_i|^2.$$

And $L(f)$ measures the *a priori loss*, i.e., a measure of discrepancy between the prospective choice f and our prior expectation about f.

Example:

$$L(f) = \|Af\|^2_{L^2},$$

where $Af = -\Delta f + f$ and

$$\|Af\|^2_{L^2} = \int_{\mathbb{R}^n} |Af(\mathbf{x})|^2 d\mathbf{x},$$

which measures the degree of non-smoothness that f has (i.e., we prefer smoother functions a priori).

Example 2: Consider the case $L(f) = \|Af\|^2$ above. The norm

$$\|f\|_{\mathcal{H}} = \|Af\|_{L^2}$$

= *reproducing kernel Hilbert space norm* (at least if dimension d is small).

If this is the case, in general things become easier.

In fact this norm comes from an inner product,

$$\langle f, g\rangle_{\mathcal{H}} = \langle Af, Ag\rangle_{L^2} = \int d\mathbf{x}\, Af(\mathbf{x}) Ag(\mathbf{x}).$$

Example 3: In the case $f = f(x),\ x \in \mathbb{R}^1$.

Suppose we choose:

$$Af = -\frac{d^2}{dx^2}f + f = \left(-\frac{d^2}{dx^2} + 1\right)f,$$

we have

$$L(f) = \|Af\|^2 = \int \left[\left(-\frac{d^2}{dx^2} + 1\right)f\right]^2 dx,$$

and $\|Af\|$ is a measure of "lack of smoothness" of f.

3. Reproducing Kernel Hilbert spaces:

Recall:

Definition 1. A $n \times n$ matrix M is symmetric if $M_{ij} = M_{ji}$ for all i, j. A symmetric M is *positive* if all of its eigenvalues are non-negative. Equivalently

$$\langle \mathbf{a}, M\mathbf{a} \rangle \equiv \mathbf{a}^T M \mathbf{a} \geq 0$$

for all vectors $\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$, with $\langle \cdot, \cdot \rangle$ the standard inner product on $\mathbb{R}^n$. Above $\mathbf{a}^T = (a_1, \ldots, a_n)$ is the transpose of $\mathbf{a}$.

Definition 2: A (real) *reproducing kernel Hilbert space (RKHS)* $\mathcal{H}$ is a Hilbert space of real-valued functions on a compact Hausdorff space X with the property that given $\mathbf{x} \in X$, the evaluation functional $\mathbf{x}^* : \mathcal{H} \to \mathbb{R}$ defined by

$$\mathbf{x}^*(f) = f(\mathbf{x})$$

is a bounded linear functional on $\mathcal{H}$, i.e., $\mathbf{x}^* \in X^*$.

Definition 3: We define a *kernel* to be a function $K : X \times X \to \mathbb{R}$ which is symmetric, i.e.,

$$K(\mathbf{x}, \mathbf{y}) = K(\mathbf{y}, \mathbf{x}).$$

We say that K is *positive* if for any collection

$$\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset X,$$

the $n \times n$ matrix

$$K = (K_{ij}) \equiv K(\mathbf{x}_i, \mathbf{x}_j)$$

is positive as an operator on the Hilbert space $\mathbb{R}^n$, i.e.,

$$\langle K\mathbf{a}, \mathbf{a} \rangle \geq 0$$

for $\mathbf{a} \in \mathbb{R}^n$, $\mathbf{a} \neq \mathbf{0}$. Equivalently the matrix K is positive as a matrix.

We now have the *representer theorem:*

Theorem 1: *Given an RKHS $\mathcal{H}$ of functions on X, there exists a unique symmetric positive definite kernel $K(\mathbf{x}, \mathbf{y})$ such that for $f \in \mathcal{H}$,*

$$f(\mathbf{x}) = \langle f(\,\cdot\,), K(\,\cdot\,, \mathbf{x}) \rangle_{\mathcal{H}}.$$

Proof: ($\Rightarrow$) for any $\mathbf{x} \in X$, since $\mathbf{x}^*$ is a continuous linear functional on $\mathcal{H}$, by the Riesz representation theorem there exists a fixed function $K_{\mathbf{x}}(\,\cdot\,)$ such that for $f \in \mathcal{H}$

$$f(\mathbf{x}) = \mathbf{x}^*(f) = \langle f(\,\cdot\,), K_{\mathbf{x}}(\,\cdot\,) \rangle. \tag{1}$$

(all inner products are in $\mathcal{H}$).

That is, evaluation of f at $\mathbf{x}$ is equivalent to an inner product with the function $K_{\mathbf{x}}$.

We define $K(\mathbf{x}, \mathbf{y}) = K_{\mathbf{x}}(\mathbf{y})$. Note by (1) we have:

$$\langle K_{\mathbf{x}}(\,\cdot\,), K_{\mathbf{y}}(\,\cdot\,) \rangle = K_{\mathbf{y}}(\mathbf{x}) = K_{\mathbf{x}}(\mathbf{y}),$$

so $K(\mathbf{x}, \mathbf{y})$ is symmetric.

To prove $K(\mathbf{x}, \mathbf{y})$ is positive, note: if $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$, then $K = (K_{ij})$ is a matrix with

$$\langle K\mathbf{c}, \mathbf{c}\rangle = \sum_{i,j=1}^{n} c_i c_j K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{i,j=1}^{n} c_i c_j \langle K_{\mathbf{x}_i}(\,\cdot\,), K_{\mathbf{x}_j}(\,\cdot\,)\rangle$$

$$= \left\langle \sum_{i=1}^{n} c_i K_{\mathbf{x}_i}(\,\cdot\,), \sum_{j=1}^{n} c_j K_{\mathbf{x}_j}(\,\cdot\,) \right\rangle = \left\| \sum_{i,j=1}^{n} c_j K_{\mathbf{x}_j}(\,\cdot\,) \right\|_{\mathcal{H}}^2 \geq 0. \quad \square$$

Definition 4: We call the above kernel $K(\mathbf{x}, \mathbf{y})$ the *reproducing kernel* of $\mathcal{H}$.

Definition 5: We define a *Mercer kernel* to be positive definite kernel $K(\mathbf{x}, \mathbf{y})$ which is also continuous.

Theorem 2:
(i) For every Mercer kernel $K : X \times X \to \mathbb{R}$, there exists a unique Hilbert space $\mathcal{H}$ of functions on X such that K is its reproducing kernel.
(ii) Moreover, $\mathcal{H}$ consists of continuous functions, and for $f \in \mathcal{H}$

$$\|f\|_\infty \leq M_K \|f\|_{\mathcal{H}},$$

where $M_K = \sup_{\mathbf{x},\mathbf{y} \in X} K(\mathbf{x}, \mathbf{y})$.

Proof: Let $K(\mathbf{x}, \mathbf{y}) : X \times X \to \mathbb{R}$ be a Mercer kernel. We will construct a reproducing kernel Hilbert space $\mathcal{H}$ with reproducing kernel K as follows.

Define

$$\mathcal{H}_0 = \text{span}\{K_{\mathbf{x}}(\,\cdot\,)\}_{\mathbf{x} \in X}.$$

Define inner product $\langle f, g\rangle_{\mathcal{H}_0}$ on $\mathcal{H}_0$ for

$$f(\,\cdot\,) = \sum_{i=1}^{l} a_i K_{\mathbf{x}_i}(\,\cdot\,), \quad g(\,\cdot\,) = \sum_{i=1}^{l} b_i K_{\mathbf{x}_i}(\,\cdot\,).$$

(Note we may assume f, g both use same set $\{\mathbf{x}_i\}$ since if not we may take a union without loss).

Then define

$$\langle f(\,\cdot\,), g(\,\cdot\,)\rangle = \left\langle \sum_{i=1}^{l} a_i K(\mathbf{x}_i,\,\cdot\,), \sum_{j=1}^{l} b_j K(\mathbf{x}_j,\,\cdot\,) \right\rangle$$

$$= \sum_{i,j=1}^{l} a_i a_j \langle K(\mathbf{x}_i,\,\cdot\,), K(\mathbf{x}_j,\,\cdot\,)\rangle = \sum_{i,j=1}^{l} a_i b_j K(\mathbf{x}_i, \mathbf{x}_j).$$

Easy to check that with this $\mathcal{H}_0$ is an inner product space. Now form the completion of this space into the Hilbert space $\mathcal{H}$. Note that for

$$f = \sum_i a_i K_{\mathbf{x}_i}(\,\cdot\,)$$

as above the norm

$$|f(\mathbf{x})| = \langle f(\,\cdot\,), K(\mathbf{x},\,\cdot\,)\rangle_{\mathcal{H}} \le \|f(\,\cdot\,)\|_{\mathcal{H}} \|K(\mathbf{x},\,\cdot\,)\|_{\mathcal{H}}$$

$$= \|f\|_{\mathcal{H}} \sqrt{\langle K(\mathbf{x},\,\cdot\,), K(\mathbf{x},\,\cdot\,)\rangle}$$

$$= \|f\|_{\mathcal{H}} \sqrt{K(\mathbf{x}, \mathbf{x})}$$

$$\le M_K \|f\|_{\mathcal{H}}.$$

This shows that the imbedding $I : \mathcal{H}_0 \to C(X)$ (the continuous functions) is continuous. Thus any Cauchy sequence in $\mathcal{H}_0$ is also Cauchy in $C(X)$, and so it follows easily that the completion $\mathcal{H}$ of $\mathcal{H}_0$ exists as a subset of $C(X)$.

That K is a reproducing kernel for $\mathcal{H}$ follows by approximation from $\mathcal{H}_0$. □

4. An example: Sobolev smoothing

Recall we used

$$L(f) = \|Af\|^2 = \int \left[\left(-\frac{d^2}{dx^2} + 1 \right) f \right]^2 dx.$$

Generalize this:

Basic definitions: Recall the Laplacian operator Δ on a function

$$f(\mathbf{x}) = f(x_1, \ldots, x_d)$$

is defined by

$$\Delta f = \frac{\partial^2}{\partial x_1^2} f + \ldots + \frac{\partial^2}{\partial x_d^2} f.$$

For $s > 0$ an even integer, we can define the Sobolev space H^s by:

$$H^s = \{ f \in L^2(\mathbb{R}^d) : (1 - \Delta)^{s/2} f \in L^2(\mathbb{R}^d) \}$$

to be functions in $L^2(\mathbb{R}^d)$ which are still in L^2 after taking the derivative $(1 - \Delta)^{s/2}$.

More specifically: we define the operator (when it exists)

$$(-\Delta + 1)^{s/2} f \equiv \mathcal{F}^{-1} \left[(|\boldsymbol{\omega}|^2 + 1)^{s/2} \widehat{f}(\omega) \right] \qquad (2)$$

Note that when $s =$ even integer and f sufficiently differentiable than the RHS of (2) coincides with the standard definition of the left hand side

For $f, g \in H^s$ define the new inner product

$$\langle f, g \rangle_{H^s} = \langle (-\Delta + 1)^{s/2} f, (-\Delta + 1)^{s/2} g \rangle_{L^2}.$$

$$= \langle (|\boldsymbol{\omega}|^2 + 1)^{s/2} \widehat{f}(\omega), (|\omega|^2 + 1) \widehat{g}(\omega) \rangle_{L^2}.$$

Can show that H^s is an RKHS with reproducing kernel

$$K(\mathbf{z}) = \mathcal{F}^{-1}\left(\frac{1}{(|\boldsymbol{\omega}|^2+1)^s}\right) \tag{3}$$

where $\mathcal{F}^{-1}$ denotes the inverse Fourier transform. The function $\frac{1}{(|\boldsymbol{\omega}|^2+1)^s}$ is a function on $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_d) \in \mathbb{R}^d$, where $|\boldsymbol{\omega}|^2 = \omega_1^2 + \ldots + \omega_d^2$.

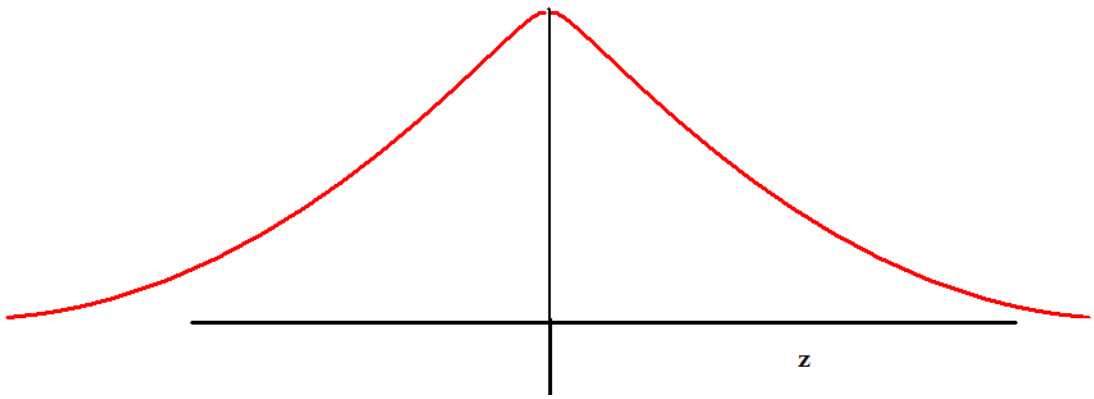


Fig 1: $K(\mathbf{z})$ in one dimension - a smooth kernel

$K(\mathbf{z})$ is called a *radial basis function.*

Note: the kernel $K(\mathbf{x}, \mathbf{y})$ (as function of 2 variables) is defined in terms of above K by

$$K(\mathbf{x}, \mathbf{y}) = K(\mathbf{x} - \mathbf{y}).$$

We thus have following Sobolev norm, measuring smoothness (with s an even integer):

$$\|f\|_{H^s} = \|(-\Delta + 1)^{s/2} f\|_{L^2} = \|(|\boldsymbol{\omega}|^2 + 1)^{s/2} \widehat{f}(\omega)\|_{L^2}. \tag{1}$$

We claim that the space H^s of functions is a reproducing kernel Hilbert space. Indeed, if we choose $K(\mathbf{x}, \mathbf{y}) = K(\mathbf{x} - \mathbf{y})$ by the property

$$K(\mathbf{z}) = \mathcal{F}^{-1}\left(\frac{1}{(2\pi)^{d/2}(|\omega|^2+1)^s}\right) \tag{3}$$

(notice we have s and not $s/2$ in the exponent). We claim that $K(\mathbf{x},\mathbf{y}) = K(\mathbf{x}-\mathbf{y})$ is a reproducing kernel for this space.

To see this note that if $f \in H^s$, then letting $A^s = (-\Delta+1)^{s/2}$:

$$\langle f(\,\cdot\,), K(\mathbf{x}-\,\cdot\,)\rangle_{H^s} = \langle A^s f(\,\cdot\,), A^s K(\mathbf{x}-\,\cdot\,)\rangle_{L^2}$$

$$= \langle A^s f(\,\cdot\,), (A^s K)(\mathbf{x}-\,\cdot\,)\rangle_{L^2}$$

$$= A^s f * A^s K(\mathbf{x}) = \mathcal{F}^{-1}\left[(2\pi)^{d/2}\mathcal{F}(A^s f)\mathcal{F}(A^s K)\right]$$

$$= \mathcal{F}^{-1}\left[(2\pi)^{d/2}\,(|\boldsymbol{\omega}|^2+1)^{s/2}\widehat{f}(\boldsymbol{\omega})(|\boldsymbol{\omega}|^2+1)^{s/2}\widehat{K}(\boldsymbol{\omega})\right]$$

$$= \mathcal{F}^{-1}\left[\widehat{f}(\boldsymbol{\omega})\right] = f(\mathbf{x}).$$

Remark: We call $K(\mathbf{z})$ a *radial basis function*, for reasons we will see. We remark that formally, this is the Green's function for the operator $(-\Delta+1)^s$. That is, it solves the formal differential equation

$$(-\Delta+1)^s K(\mathbf{z}) = \delta(\mathbf{z}).$$

Conclusion: $H^s =$ Sobolev space of s times differentiable functions in L^2 is a reproducing kernel Hilbert space, and its reproducing kernel is $K(\mathbf{z})$ given in (3) above. This kernel is also (informally) the Green's function for the operator $(-\Delta+1)^s$.

The Representation Theorem for RKHS

1. An application: using Sobolev spaces for regularization

Assume again we have an unknown function f on X, with only data

$$Nf = \mathbf{y} \equiv (y_1, \ldots, y_n) = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)).$$

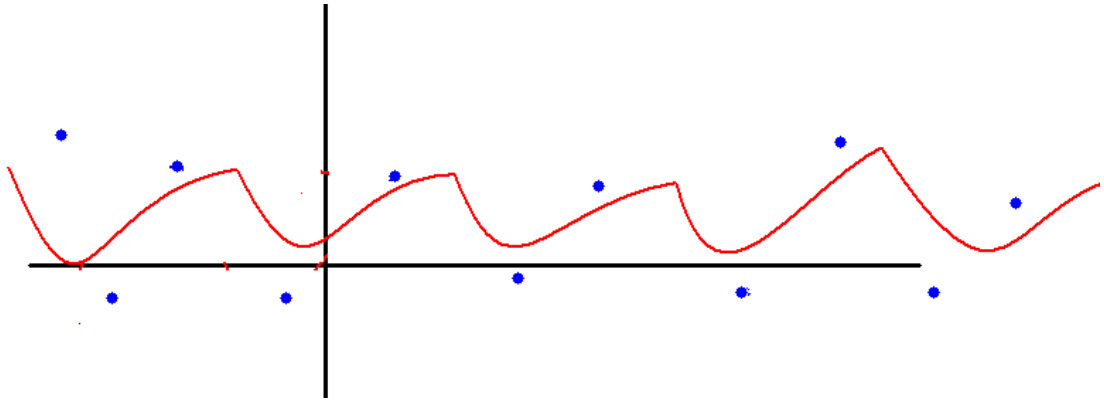
To find f_0, approximate it by the minimizer

$$\widehat{f} = \arg\min_{f \in H^s} \{ \|Nf - \mathbf{y}\|^2 + \lambda \|f\|_{H^s}^2 \}. \tag{4}$$

where λ can be some constant. Note we are finding an f which balances minimizing

$$\|Nf - \mathbf{y}\|^2 = \frac{1}{n} \sum_{i=1}^{n} (f(\mathbf{x}_i) - y_i)^2,$$

i.e., the data error, with minimizing $\|f\|_{H^s}^2$, i.e., maximizing the smoothness. The solution to such a problem will look like this:



It will compromise between fitting the data (which may have error) and trying to be smooth.

The amazing thing: $\widehat{f}$ can be found explicitly using radial basis functions.

2. Solving the minimization

Now consider the optimization problem (4). We claim that we can solve it explicitly. To see this works in general for RKHS, return to the general problem.

Given an unknown $f_0 \in \mathcal{H}$ = RKHS. Try to find the "best" approximation f to f_0 fitting the data $Nf_0 \equiv (f_0(\mathbf{x}_1), \ldots, f_0(\mathbf{x}_n)) = \mathbf{y}$, but ALSO satisfying a priori knowledge that $\|f_0\|_{\mathcal{H}}$ is small.

Specifically, we want to find

$$\arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} V(f(\mathbf{x}_i), y_i) + \lambda\|f\|_{\mathcal{H}}^2. \tag{1}$$

Note we can have, e.g., $V(f(\mathbf{x}_i), y_i) = (f(\mathbf{x}_i) - y_i)^2$. In that case

$$\sum_{i=1}^{n} V(f(\mathbf{x}_i), y_i) = \sum_{i=1}^{n} (f(\mathbf{x}_i) - y_i)^2 = \|Nf - \mathbf{y}\|^2.$$

Consider the general case (1), with arbitrary error measure V. We have the

Representation Theorem: *A solution of the Tikhonov optimization problem* (1) *can be written*

$$f(x) = \sum_{i=1}^{n} a_i K(\mathbf{x}, \mathbf{x}_i), \tag{2}$$

where K *is the reproducing kernel of the RKHS* $\mathcal{H}$.

Important theorem: says we only need to find a set of n numbers a_i to optimize the infinite dimensional problem (1) above.

Proof: Use calculus of variations. If a minimizer f_1 exists, then for all $g \in \mathcal{H}$, assuming that the derivatives with respect to ϵ exist:

$$0 = \frac{d}{d\epsilon}\frac{1}{n}\sum_{i=1}^{n} V((f_1 + \epsilon g)(\mathbf{x}_i), y_i) + \lambda\|f_1 + \epsilon g\|_{\mathcal{H}}^2$$

$$= \frac{1}{n}\sum_{i=1}^{n} \frac{\partial V}{\partial f_1(\mathbf{x}_i)}(f_1(\mathbf{x}_i), y_i) \cdot g(\mathbf{x}_i)$$

$$+ \lambda\frac{d}{d\epsilon}\{\langle f_1, f_1\rangle + 2\epsilon\langle f_1, g\rangle + \epsilon^2\langle g, g\rangle\}\Big|_{\epsilon=0}$$

$$= \frac{1}{n}\sum_{i=1}^{n} V_1(f_1(\mathbf{x}_i), y_i) \cdot g(\mathbf{x}_i) + 2\lambda\langle f_1, g\rangle,$$

where $V_1(a,b) = \frac{\partial}{\partial a}V(a,b)$ and all inner products are in $\mathcal{H}$.

Since the above is true for all $g \in \mathcal{H}$, it follows that if we let $g = K_{\mathbf{x}}$ we get:

$$0 = \frac{1}{n}\sum_{i=1}^{n} V_1(f_1(\mathbf{x}_i), y_i) K_{\mathbf{x}}(\mathbf{x}_i) + 2\lambda\langle f_1, K_{\mathbf{x}}\rangle$$

$$= \frac{1}{n}\sum_{i=1}^{n} V_1(f_1(\mathbf{x}_i), y_i) K_{\mathbf{x}}(\mathbf{x}_i) + 2\lambda f_1(\mathbf{x}),$$

or

$$f_1(\mathbf{x}) = \frac{1}{2\lambda n}\sum_{i=1}^{n} V_1(f_1(\mathbf{x}_i), y_i) K(\mathbf{x}, \mathbf{x}_i).$$

Thus if a minimizer f_1 exists for (1), it can be written in the form (2) as claimed, with

$$a_i = \frac{1}{2\lambda n} V_1(f_1(\mathbf{x}_i), y_i).$$

Note that this does not solve the problem, since the a_i are expressed in terms of the solution itself. But it does reduce the possibilities for what a solution looks like.

3. An example: square loss

Considering again the case where we have information

$$Nf = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)) = \mathbf{y},$$

to find the solution of the regularization problem we wish to find

$$f_0 = \arg\inf_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} V(f(\mathbf{x}_i), y_i) + \lambda\|f\|_{\mathcal{H}}^2 \qquad (1)$$

Plugging in the universal solution

$$f(\mathbf{x}) = \sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_j)$$

into (1) we get:

$$f_0 = \inf_{a_1,\ldots,a_n} \frac{1}{n} \sum_{i=1}^{n} V\left(\sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_j), y_j \right) + \lambda \left\| \sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_i) \right\|_{\mathcal{H}}^2 \qquad (1)$$

Notice that

$$\left\| \sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_j) \right\|_{\mathcal{H}}^2 = \sum_{i=1}^{n} a_i a_j K_{ij} = \mathbf{a}^T K \mathbf{a}.$$

where $K = (K_{ij}) = (K(\mathbf{x}_i, \mathbf{x}_j))$, and $\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$.

Thus

$$f_0 = \min_{\mathbf{a} \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^{n} V\left(\sum_{j=1}^{n} a_j K(\mathbf{x}_i, \mathbf{x}_j), y_i \right) + \lambda \mathbf{a}^T K \mathbf{a}.$$

This minimizes over $a_1, \ldots, a_n$ and is finite dimensional minimization problem. Can take derivatives wrt a_i and set equal to 0.

Special case: $V(f(\mathbf{x}), y) = (f(\mathbf{x}) - y)^2$.

Here

$$\mathbf{a} = \min_{\mathbf{a} \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{n} a_j K(\mathbf{x}_i, \mathbf{x}_j) - y_i \right)^2 + \lambda \mathbf{a}^T K \mathbf{a}$$

$$= \min_{\mathbf{a} \in \mathbb{R}^n} \frac{1}{n} (K\mathbf{a} - \mathbf{y})^2 + \lambda \mathbf{a}^T K \mathbf{a}.$$

Take the gradient with respect to $\mathbf{a}$ and setting to 0 we get:

$$0 = \frac{2}{n}K(K\mathbf{a} - \mathbf{y}) + 2\lambda K\mathbf{a} = \left(\frac{2K^2}{n} + 2\lambda K\right)\mathbf{a} - \frac{1}{n}K\mathbf{y}.$$

$$= K\left(\frac{K}{n} + \lambda\right)\mathbf{a} - K\frac{\mathbf{y}}{n}.$$

Thus if K is nonsingular:

$$\mathbf{a} = \left(\frac{K}{n} + \lambda\right)^{-1}\left(\frac{\mathbf{y}}{n}\right) = (K + \lambda n)^{-1}\mathbf{y}.$$

Explicit solution. Thus

$$f_0(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} a_i K(\mathbf{x}, \mathbf{x}_i)$$

= superposition of radial basis functions.

Support Vector Machines

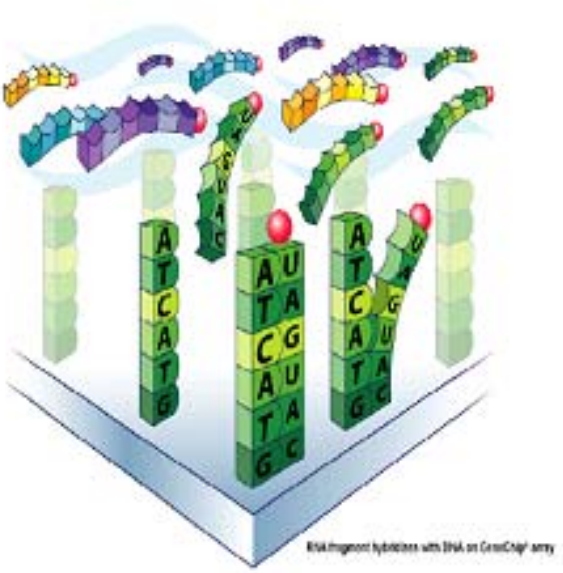
Microarray experiment:

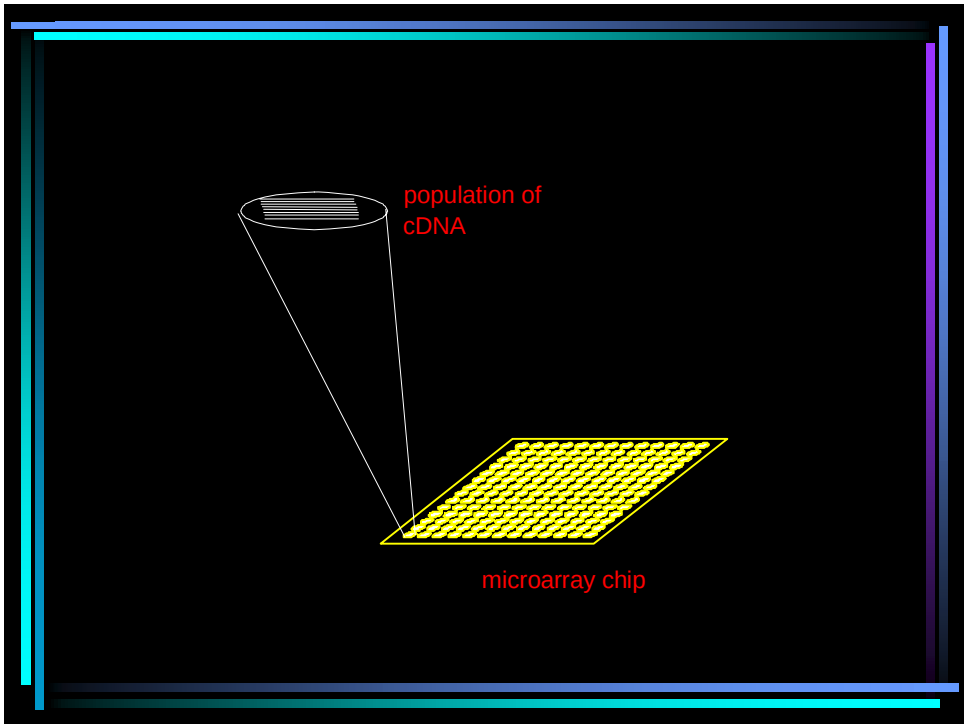
Question: Gene expression - when is the DNA in a gene g transcribed and thus expressed (as RNA) in a cell?

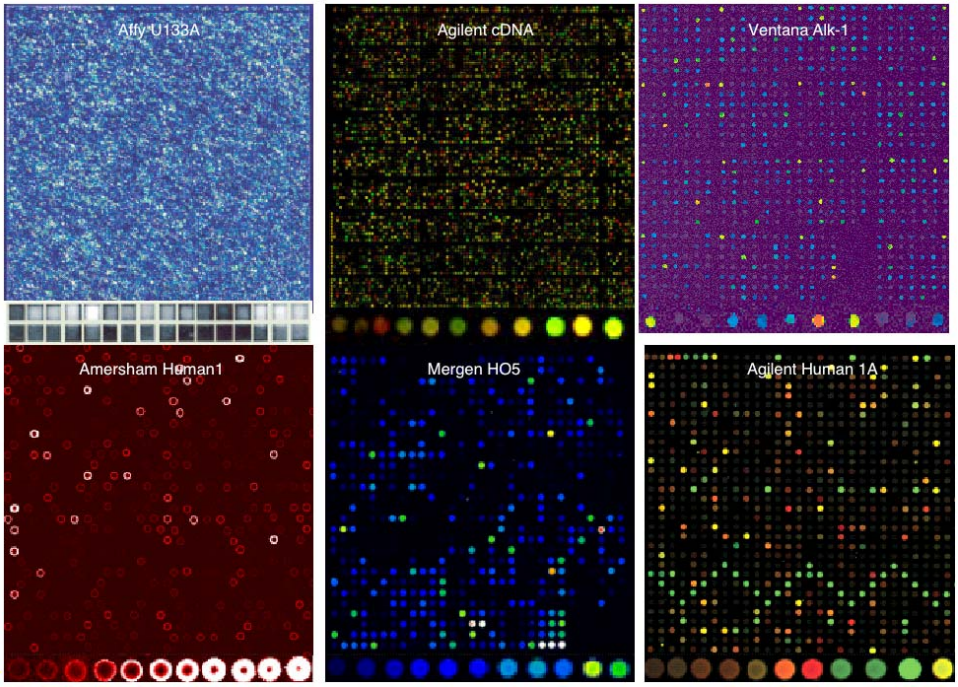
One solution: Measure RNA levels (result of transcription)

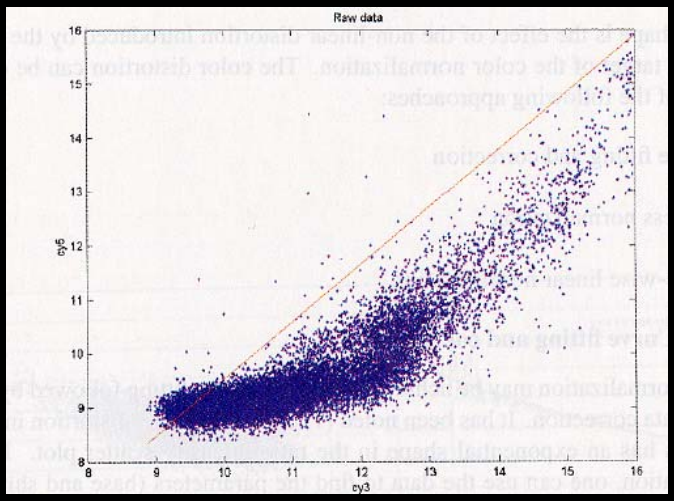
Method: Microarray

cDNA arrays: Spotted onto surface
oligonucleotide arrays: created on surface

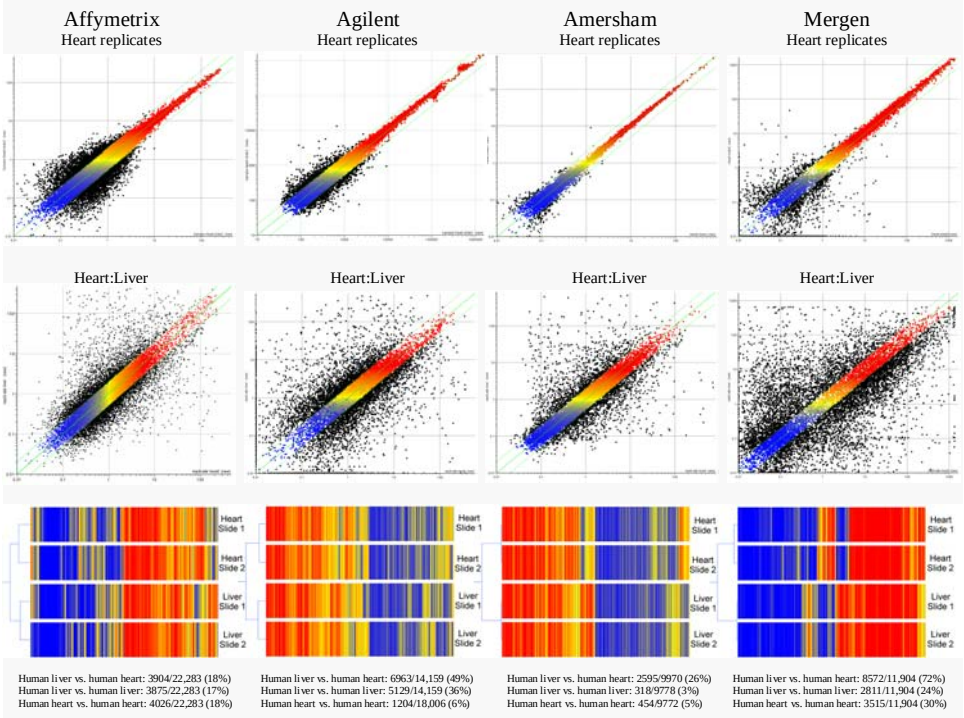



population of
cDNA
microarray chip





From: "Data Analysis Tools for DNA microarrays" by S. Draghici, published by Chapman and Hall/CRC Press.



Result: for each subject tissue sample s, obtain a *feature vector*

$$\Phi(s) = \mathbf{x} = (x_1, \ldots, x_{20,000})$$

consisting of expression levels of 20,000 genes.

Can we classify tissues this way? Can we differentiate cancers between ALL (lymphoblastic leukemia) and AML (myeloblastic leukemia)

Goals:

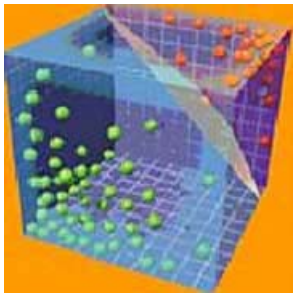
1. differentiate two different but similar cancers.
2. Understand genetic pathways of cancer

Basic difficulties: few samples (e.g., 30-200); high dimension (e.g., 5,000 - 100,000).

Curse of dimensionality - too few samples and too many parameters (dimensions) to fit them.

Tool: Support vector machine (SVM)

Procedure: look at feature space F in which $\Phi(s)$ lives, and differentiate examples of one and the other cancer with a hyperplane:



Methods needed: Machine learning; reproducing kernel Hilbert spaces (RKHS)

Consider a training data set

$$\mathcal{D}_0 = \{(\mathbf{x}_i, y_i)\}_{i=1}^n,$$

where $\mathbf{x}_i$ is a sampe microarray and $y_i \in \mathcal{B} = \{-1, 1\}$

Assume that

$$y_i = \begin{cases} 1 & \text{if tumor is ALL} \\ -1 & \text{if tumor is AML} \end{cases}.$$

How to learn the function $f_1 : F \to \mathcal{B}$ from examples?

Remark: The map f_1 is difficult to guess from examples $\mathcal{D}$.

With data set $\mathcal{D}$, can we find the right function $f_1 : V \to \mathcal{B}$ which generalizes the above examples, so that $f_1(\mathbf{v}) = y$ for all feature vectors?

Easier: find a $f : V \to \mathbb{R}$, where

$$f(\mathbf{x}) > 0 \text{ if } f_1(\mathbf{x}) = 1; \;\; f(\mathbf{x}) < 0 \;\; \text{if} \;\; f_1(\mathbf{x}) = -1.$$

4. Support vector machine framework

Recall the *regularization setting*: we have n examples

$$D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\},$$

with $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \mathbb{B} = \{\pm 1\}$.

As mentioned above, we want to find a function $f_1 : \mathbb{R}^d \to \mathbb{B}$ which *generalizes* the above data so that $f(\mathbf{x}) = y$ generalizes the data D.

As mentioned there, we will actually want here something more general: a function $f(\mathbf{x})$ which will best help us decide the true value of y. It may not need to be that we want $f(\mathbf{x}) = y$, but rather we want

$$\begin{cases} f(\mathbf{x}) >> 1 & \text{if } y = 1 \\ f(\mathbf{x}) << 1 & \text{if } y = -1\text{'} \end{cases} \tag{2}$$

i.e., $f(\mathbf{x})$ is large and positive if the correct answer is $y = 1$ (e.g. a chair) and $f(\mathbf{x})$ is large and negative if the correct answer is $y = -1$ (not a chair).

Then the decision rule will be to conclude the value of y based on the rule (2). This is made precise as follows. We have the following

optimization criterion for the 'right' f:

$$f = \arg\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} V(y_i, f(\mathbf{x}_i)) + \lambda \|f\|_K^2,$$

where $\|f\|_K =$ norm in an RKHS $\mathcal{H}$, e.g.,

$$\|f\|_K = \|Af\|_{L^2} = \int_{\mathbb{R}^n} (Af)^2 dx.$$

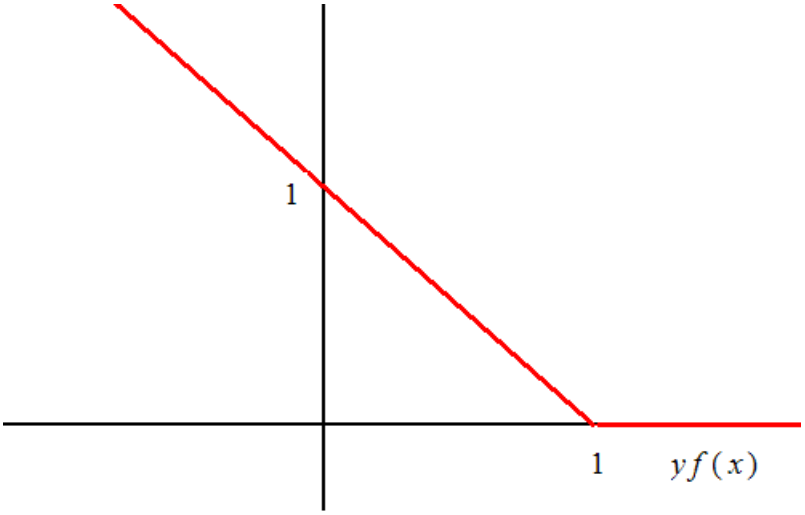
Above 'arg min' denotes the f which minimizes the above expression.

5. Loss function: hinge loss

Consider the error function

$$V(f(\mathbf{x}), y) = (1 - yf(\mathbf{x}))_+ \equiv \max(1 - yf(\mathbf{x}), 0).$$

$$= \begin{cases} \text{small} & \text{if } y, f(\mathbf{x}) \text{ have same sign} \\ \text{large} & \text{otherwise} \end{cases}.$$



This is called the *hinge loss function.*

[Notice a *margin* is built in the error is 0 only if $yf(\mathbf{x}) \geq 1$ (a more stringent requirement than just $yf(\mathbf{x}) \geq 0$)]

Thus data error is

$$e_d = \frac{1}{n}\sum_{i=1}^{n} V(f(\mathbf{x}_i), y_i)$$

What is a priori information? Note surface $H : f = 0$ will separate "positive" **x** with $f(\mathbf{x}) > 0$, and "negative" **x** with $f(\mathbf{x}) < 0$:

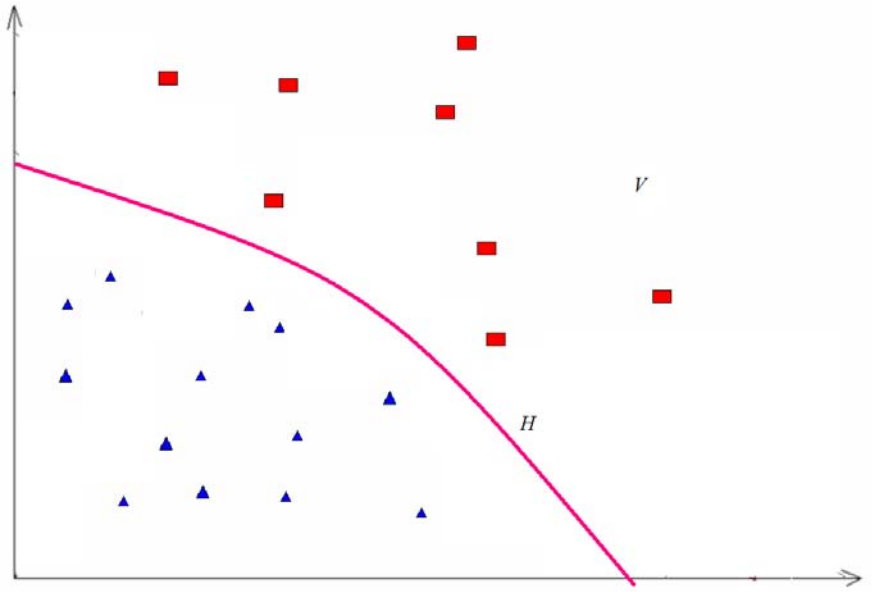


Fig. 1. Red points have $y = +1$ and blue have $y = -1$ in the space V. $H : f(\mathbf{x}) = 0$ is the separating surface.

Assume some a priori information defined in terms of an RKHS norm $\|\cdot\|_K$ so $\|f\|_K$ is small if a priori assumption is satisfied. Let $\mathcal{H}$ be corresponding RHKS.

Will specify desirable norm $\|\cdot\|_K$ later...

Now solve regularization problem for the above norm and loss V:

$$f_0 = \arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{j=1}^{n}(1 - y_j f(\mathbf{x}_j))_+ + \lambda\|f\|_K^2. \tag{1}$$

6. Introduction of slack variables

Define new variables ξ_j, and note if we find the min over $f \in \mathcal{H}$ and ξ_j of

$$\arg\min_{f\in\mathcal{H}} \frac{1}{n}\sum_{j=1}^{n}\xi_j + \lambda\|f\|_K^2$$

with the constraint

$$y_j f(\mathbf{x}_j) \geq 1 - \xi_j$$

$$\xi_j \geq 0,$$

we get the same solution f.

To see this, note the constraints are

$$\xi_j \geq \max\,(0, 1 - y_j f(\mathbf{x}_j)) = (1 - y_j f(\mathbf{x}_j))_+,$$

which yields the claim.

From form (1) above by representer theorem:

$$f(\mathbf{x}) = \sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_j).$$

To find $\mathbf{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}$ (see previous lecture): let $K = (K_{ij}) = K(\mathbf{x}_i, \mathbf{x}_j)$

$$\mathbf{a} = \arg\min_{\mathbf{a} \in \mathbb{R}^n} \frac{1}{n} \sum_{j=1}^{n} \xi_j + \lambda \mathbf{a}^T K \mathbf{a} \tag{2a}$$

with constraint:

$$\mathbf{y}_j \sum_{i=1}^{n} a_i K(\mathbf{x}_i, \mathbf{x}_j) \geq 1 - \xi_j \tag{2b}$$

$$\xi_j \geq 0. \tag{2c}$$

5. Bias

Given choice of $\mathcal{H}$, K we have concluded

$$f(\mathbf{x}) = \sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_j) \tag{3}$$

which optimizes (1), equivalently (2).

Now can *expand* class (2) of allowable f *ad hoc*. We may feel larger class than $\mathcal{H}$ is appropriate. Often adding a constant b is useful.

Thus change $f(\mathbf{x})$ by adding a bias term b:

$$f(\mathbf{x}) = \sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_j) + b. \tag{4}$$

Then plug into (1).

The effect: regularization term unchanged (i.e., we ignore b in the norm $\|f\|_{\mathcal{H}}$; remember any a priori assumption is valid if it is useful).

Note this is still a norm on the expanded space of functions of the form (4), but may not be positive definite, i.e., $\|f\| = 0$ for some f of the form (4).

For example if $\|b\|_{\mathcal{H}} = 0$, this may be the case.

But: minimization of (1) using (4) still makes sense and allows possibly richer set of functions than $\mathcal{H}$, as long as the regularization term $\|f\|_{\mathcal{H}}$ still makes sense for such a richer set.

In terms of slack variables ξ_i, new optimization problem is to find $\mathbf{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}$ which minimizes:

$$\frac{1}{n}\sum_{j=1}^{n} \xi_j + \lambda \mathbf{a}^T K \mathbf{a}$$

with constraints:

$$y_j \left(\sum_{i=1}^{n} a_i K(\mathbf{x}_i, \mathbf{x}_j) + b \right) \geq 1 - \xi_j$$

$$\xi_i \geq 0$$

(*quadratic programming* problem).

Solving SVM: Quadratic Programming

1. Quadratic programming (QP): Introducing Lagrange multipliers α_j and μ_j (can be justified in QP for inequality as well as equality constraints) we define the Lagrangian

$$L(\mathbf{a}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu})$$

$$\equiv \frac{1}{n}\sum_{i=1}^{n}\xi_i + \lambda \mathbf{a}^T K \mathbf{a} - \sum_{j=1}^{n}\alpha_j\left[\left(\sum_{j=1}^{n} a_j K(\mathbf{x}_i, \mathbf{x}_j) + b\right) y_j - 1 + \xi_j\right]$$

$$- \sum_{j=1}^{n}\mu_j\xi_j.$$

By Lagrange multiplier theory for constraints with inequalities, the minimum of this in

$$\mathbf{a}, b, \xi, \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n),\ \boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)$$

is a stationary point of this Lagrangian (derivatives vanish) is maximized wrt $\mathbf{a}, b, \boldsymbol{\xi}$, and minimized wrt the Lagrange multipliers, $\boldsymbol{\alpha}$, $\boldsymbol{\mu}$ subject to the constraints

$$\alpha_i, \mu_i \geq 0. \tag{5}$$

Derivatives:

$$\frac{\partial L}{\partial b} = 0 \ \Rightarrow\ \sum_{j=1}^{n}\alpha_j y_j = 0; \tag{6a}$$

$$\frac{\partial L}{\partial \xi_i} = 0 \ \Rightarrow \frac{1}{n} - \alpha_j - \mu_j = 0. \tag{6b}$$

Plugging in get reduced Lagrangian

$$L^*(\mathbf{a}, \boldsymbol{\alpha}) = \lambda \mathbf{a}^T K \mathbf{a} - \sum_{j=1}^{n}\alpha_j\left(y_j\sum_{j=1}^{n} a_j K(\mathbf{x}_i, \mathbf{x}_j) - 1\right)$$

$$= \sum_{j=1}^{n} \alpha_j + \lambda \mathbf{a}^T K \mathbf{a} - \boldsymbol{\alpha}^T Y K \mathbf{a}$$

where

$$Y = \begin{bmatrix} y_1 & 0 & 0 & \dots & 0 \\ 0 & y_2 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & y_{n-1} & 0 \\ 0 & 0 & 0 & \dots & y_n \end{bmatrix}$$

(note (6) eliminates the ξ_j terms) with same constraints (5).

Now:

$$\frac{\partial L^*}{\partial a_j} = 0 \;\Rightarrow 2\lambda K \mathbf{a} - KY\boldsymbol{\alpha} = 0 \qquad (7)$$

$$\Rightarrow a_i = \frac{\alpha_i y_i}{2\lambda}.$$

Plug in for a_i using (7), replacing $K\mathbf{a}$ by $\frac{1}{2\lambda} KY\boldsymbol{\alpha}$ everywhere:

$$L^*(\mathbf{a}, \boldsymbol{\alpha}) = \sum_{j=1}^{n} \alpha_j - \frac{1}{4\lambda} \boldsymbol{\alpha}^T Y K Y^T \boldsymbol{\alpha}$$

$$= \sum_{j=1}^{n} \alpha_j - \frac{1}{4\lambda} \boldsymbol{\alpha}^T P \boldsymbol{\alpha},$$

where $P = YKY^T$.

Constraints: $\alpha_j, \mu_j \geq 0$; by (6) this reduces to

$$0 \leq \alpha_j \leq \frac{1}{n}.$$

Define $C = \frac{1}{2\lambda n}$, $\overline{\boldsymbol{\alpha}} = \frac{1}{2\lambda} \boldsymbol{\alpha}$.

[note this does not mean complex conjugate!]

Then want to minimize (division by constant 2λ OK - does not change minimizing $\overline{\boldsymbol{\alpha}}$)

$$\frac{1}{2\lambda}\sum_{j=1}^{n} 2\lambda\overline{\alpha}_j - \frac{(2\lambda)^2}{2\lambda}\frac{1}{4\lambda}\overline{\boldsymbol{\alpha}}^T P\overline{\boldsymbol{\alpha}} = \sum_{j=1}^{n}\overline{\alpha}_j - \frac{1}{2}\overline{\boldsymbol{\alpha}}^T P\overline{\boldsymbol{\alpha}}, \tag{8}$$

subject to constraint $0 \leq \overline{\alpha}_i \leq C$; also convenient to include (6a) as constraint: $\overline{\boldsymbol{\alpha}} \cdot \mathbf{y} = 0$. Thus constraints are:

$$0 \leq \overline{\boldsymbol{\alpha}} \leq C; \;\; \overline{\boldsymbol{\alpha}} \cdot \mathbf{y} = 0.$$

Summarizing above relationships:

$$f(\mathbf{x}) = \sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_j) + b,$$

where

$$a_j = \frac{\alpha_j y_j}{2\lambda},$$

$$\alpha_j = 2\lambda\overline{\alpha}_j,$$

and $\overline{\alpha}_j$ are the (unconstrained) minimizers of (8), with

$$P = YKY^T.$$

After a_j are determined, b must be computed directly by plugging into

More briefly,

$$f(\mathbf{x}) = \sum_{j=1}^{n} \overline{\alpha}_j y_j K(\mathbf{x}, \mathbf{x}_j) + b,$$

where $\overline{\alpha}_j$ minimize (8).

Finally, to find b, must plug into original optimization problem: that is, we minimize

$$\frac{1}{n}\sum_{j=1}^{n}(1-y_j f(\mathbf{x}_j))_+ + \lambda\|f\|_K^2$$

$$= \frac{1}{n}\sum_{j=1}^{n}\left(1-y_j\left[\sum_{i=1}^{n}a_i K(\mathbf{x}_j,\mathbf{x}_i)+b\right]\right)_+ + \lambda\mathbf{a}^T K\mathbf{a}.$$

2. The RKHS for SVM

General SVM: solution function is (see (4) above)

$$f(\mathbf{x}) = \sum_j a_j K(\mathbf{x},\mathbf{x}_j) + b,$$

with sol'n for a_j given by quadratic programming as above.

Consider a simple case (linear kernel):

$$K(\mathbf{x},\mathbf{x}_j) = \mathbf{x}\cdot\mathbf{x}_j.$$

Then we have

$$f(\mathbf{x}) = \sum_j (a_j\mathbf{x}_j)\cdot\mathbf{x} + b \equiv \mathbf{w}\cdot\mathbf{x} + b,$$

where

$$\mathbf{w} \equiv \sum_j a_j\mathbf{x}_j.$$

This gives the kernel. What class of functions is the corresponding space $\mathcal{H}$?

Claim it is the set of linear functions of **x:**

$$\mathcal{H} = \{\mathbf{w}\cdot\mathbf{x} | \mathbf{w}\in\mathbb{R}^d\}$$

with inner product

$$\langle \mathbf{w}_1\cdot\mathbf{x}, \mathbf{w}_2\cdot\mathbf{x}\rangle_{\mathcal{H}} = \mathbf{w}_1\cdot\mathbf{w}_2$$

is the RKHS of $K(\mathbf{x},\mathbf{y})$ above.

Indeed to show that $K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}$ is the reproducing kernel for $\mathcal{H}$, note that if $f(\mathbf{x}) = \mathbf{x} \cdot \mathbf{w} \in \mathcal{H},$ then recall $K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}$, so

$$\langle f(\,\cdot\,), K(\,\cdot\,, \mathbf{y})\rangle_{\mathcal{H}} = \mathbf{w} \cdot \mathbf{y} = f(\mathbf{y}),$$

as desired.

Thus the matrix $K_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j$, and we find the optimal separator

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x}$$

by solving for $\mathbf{w}$ as before.

Note when we add b to $f(\mathbf{x})$ (as done earlier), have all affine functions $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$.

Note above inner product gives the norm

$$\|\mathbf{w} \cdot \mathbf{x}\|^2_{\mathcal{H}} = \|\mathbf{w}\|^2_{\mathbb{R}^n} = \sum_{j=1}^{n} w_j^2.$$

Why use this norm? A priori information content.

Thus we consider just linear functions as a start; will presumably lead to linear separators $f(\mathbf{x})$, with the hyperplane $H : f(\mathbf{x}) = 0$ separating the classes in the feature space V.

Final classification rule: $f(\mathbf{x}) > 0 \Rightarrow y = 1; \quad f(\mathbf{x}) < 0 \Rightarrow y = -1.$

Learning from training data:

$$Nf = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)) = (y_1, \ldots, y_n).$$

Thus

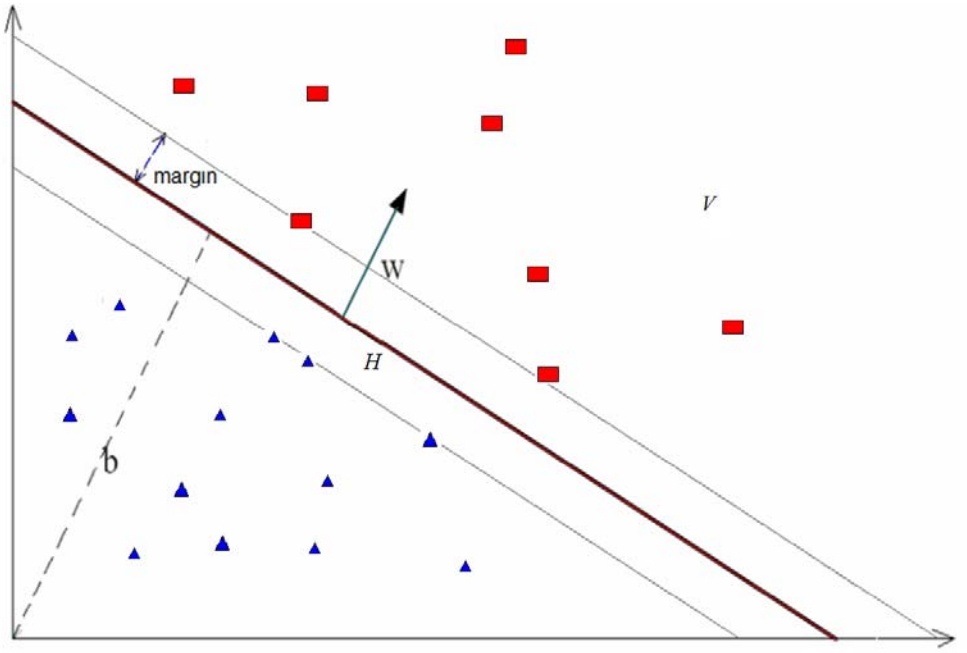
$$\mathcal{H} = \{f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} : \mathbf{w} \in \mathbb{R}^n\}$$

is the set of linear separator functions (known as *perceptrons* in neural network theory). Can easily check $\mathcal{H}$ is an n dimensional Hilbert space of functions, with inner product

$$\langle \mathbf{w}_1 \cdot \mathbf{x}, \mathbf{w}_2 \cdot \mathbf{x} \rangle_{\mathcal{H}} = \mathbf{w}_1 \cdot \mathbf{w}_2.$$

Consider separating hyperplane $H : f(\mathbf{x}) = 0$:

What's the best separator? One with greatest *margin*.



Thus: assuming data are separable by a plane, goal is to find hyperplane H with widest *margin*, i.e., perpendicular distance from plane to closest points on either side.

This is our *a priori* assumption - that the plane will likely have this property.

Can we incorporate this assumption into a RKHS norm, so $\|f\|_{\mathcal{H}}$ small if margin large?

We are using the following norm: If affine function $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$ for some $\mathbf{w} \in V$, define norm by:

$$\|f\|_{\mathcal{H}} = \|\mathbf{w}\|.$$

Then our functional to be minimized is:

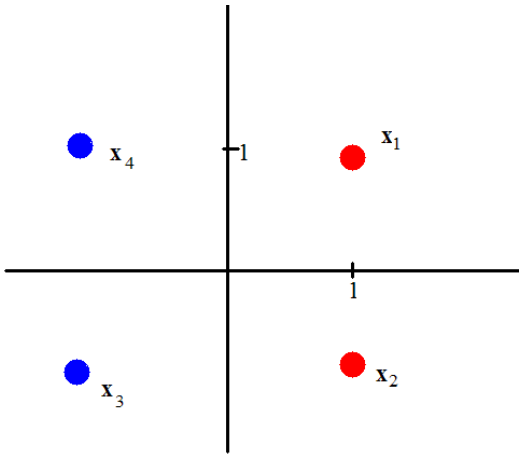
$$e = \frac{1}{n}\sum_{j=1}^{n}(1 - y_j f(\mathbf{x}_j)) + \lambda\|\mathbf{w}\| \equiv e_d + e_p \qquad (1)$$

(the minimization is over $(\mathbf{w}, b)$).

Why is this a good choice? And what should λ be?

Clearly we will consider the optimal choice of $(\mathbf{w}, b)$ to be the one with the largest margin.

3. Toy example:



Information

$$Nf = \{[(1,1),1],\ [(1,-1),1],\ [(-1,1),-1],[(-1,-1),-1]\}$$

(red = +1; blue = -1);

$$f = \mathbf{w}\cdot\mathbf{x} + b$$

$$= \sum_i a_i \underbrace{(\mathbf{x}_i \cdot \mathbf{x})}_{K(\mathbf{x}_i, \mathbf{x})} + b$$

so

$$\mathbf{w} = \sum_i a_i \mathbf{x}_i.$$

$$L(f) = \frac{1}{4}\sum_j (1 - f(\mathbf{x}_j)y_j)_+ + \frac{1}{2}|\mathbf{w}|^2 \tag{9}$$

(we let $\lambda = 1/2$; minimize wrt **w**, b).

Equivalent:

$$L(f) = \frac{1}{4}\sum_{j=1}^{4} \xi_j + \frac{1}{2}|\mathbf{w}|^2$$

$$y_j f(\mathbf{x}_j) \geq 1 - \xi_j; \qquad \xi_j \geq 0.$$

[Note effectively $\xi_i = (1 - (\mathbf{w} \cdot \mathbf{x}_i + b)y_i)_+$]

Define kernel matrix

$$K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j = \begin{bmatrix} 2 & 0 & -2 & 0 \\ 0 & 2 & 0 & -2 \\ -2 & 0 & 2 & 0 \\ 0 & -2 & 0 & 2 \end{bmatrix}$$

$$\|f\|_{\mathcal{H}} = |\mathbf{w}|^2 = \mathbf{a}^T K \mathbf{a} = 2\left(\sum_{i=1}^{4} a_i^2\right) - 4(a_1 a_3 + a_2 a_4).$$

where $\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_4 \end{bmatrix}$.

Formulate

$$L(f) = L(\mathbf{a}, b, \boldsymbol{\xi}) = \frac{1}{4}\sum_{j=1}^{4}\xi_j + \frac{1}{2}\mathbf{a}^T K\mathbf{a}$$

$$= \frac{1}{4}\sum_{j=1}^{4}\xi_i + \left(\sum_{j=1}^{4}a_i^2\right) - 2(a_1a_3 + a_2a_4)$$

subject to (Eq. 4a):

$$\xi_j \geq (1 - y_j[(K\mathbf{a})_j + b]), \qquad \xi_j \geq 0.$$

Lagrange multipliers $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^T$, $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)^T$ (see (4b)):

optimize

$$L(\mathbf{a}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = \frac{1}{4}\sum_{j=1}^{4}\xi_i + \frac{1}{2}\mathbf{a}^T K\mathbf{a} - \sum_{j=1}^{4}\alpha_j\big[\big((K\mathbf{a})_j + b\big)y_j - 1 + \xi_j\big]$$

$$- \sum_{j=1}^{4}\mu_j\xi_j$$

$$= \frac{1}{4}\sum_{j=1}^{4}\xi_j + \frac{1}{2}\mathbf{a}^T K\mathbf{a} - \boldsymbol{\alpha}^T YK\mathbf{a} - b\boldsymbol{\alpha}^T\mathbf{y} + \sum_{j=1}^{4}\alpha_j - \boldsymbol{\alpha}\cdot\boldsymbol{\xi} - \boldsymbol{\mu}\cdot\boldsymbol{\xi} \quad (10)$$

with constraints

$$\alpha_i, \mu_i \geq 0.$$

Solution has (see (7) above)

$$\boldsymbol{\alpha} = 2\lambda Y^{-1}\mathbf{a}$$

(recall $Y = \begin{bmatrix} y_1 & 0 & \ldots & 0 \\ 0 & y_2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots & y_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$)

and (7a above)

$$\overline{\boldsymbol{\alpha}} = \frac{1}{2\lambda}\boldsymbol{\alpha} = \boldsymbol{\alpha}.$$

Finally optimize (7b)

$$\sum_{i=1}^{4}\overline{\alpha}_i - \frac{1}{2}\overline{\boldsymbol{\alpha}}^T P\overline{\boldsymbol{\alpha}},$$

where

$$P = YKY^T$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 2 & 0 & -2 & 0 \\ 0 & 2 & 0 & -2 \\ -2 & 0 & 2 & 0 \\ 0 & -2 & 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \\ -2 & 0 & -2 & 0 \\ 0 & -2 & 0 & -2 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \\ 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \end{bmatrix}.$$

constraint is

$$0 \leq \overline{\boldsymbol{\alpha}} \leq C \equiv \frac{1}{2\lambda n} = \frac{1}{4}. \tag{10a}$$

Thus optimize

$$L_1 = \sum_{i=1}^{4}\overline{\alpha}_i - \left(\sum_{i=1}^{4}\overline{\alpha}_i^2 + 2\overline{\alpha}_1\overline{\alpha}_3 + 2\overline{\alpha}_2\overline{\alpha}_4\right)$$

$$= \sum_{i=1}^{4} \overline{\alpha}_i - (\overline{\alpha}_1 + \overline{\alpha}_3)^2 - (\overline{\alpha}_2 + \overline{\alpha}_4)^2.$$

$$= u + v - u^2 - v^2,$$

where

$$u = \overline{\alpha}_1 + \overline{\alpha}_3; \quad v = \overline{\alpha}_2 + \overline{\alpha}_4.$$

Minimizing:

$$1 - 2u = 0; \quad 1 - 2v = 0$$

$\Rightarrow$

$$u = v = \frac{1}{2}.$$

Thus we have

$$\overline{\alpha}_i = \frac{1}{4}$$

for all i (recall the constraint (10a)). Then

$$\boldsymbol{\alpha} = 2\lambda\overline{\boldsymbol{\alpha}} = \overline{\boldsymbol{\alpha}} = \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}.$$

Thus

$$\mathbf{a} = \frac{Y\boldsymbol{\alpha}}{2\lambda} = \begin{bmatrix} 1/4 \\ 1/4 \\ -1/4 \\ -1/4 \end{bmatrix}.$$

Thus

$$\mathbf{w} = \sum a_i \mathbf{x}_i = \frac{1}{4}(\mathbf{x}_1 + \mathbf{x}_2 - \mathbf{x}_3 - \mathbf{x}_4) = \frac{1}{2}((4,0)) = (1,0)\ .$$

Margin = $\frac{1}{|\mathbf{w}|} = 1$.

Now we find b separately from original equation (9); we will minimize with respect to b the original functional

$$L(f) = \frac{1}{4}\sum_j (1 - (\mathbf{w}\cdot\mathbf{x}_j + b)y_j)_+ + |\mathbf{w}|^2 \tag{11}$$

$$= \frac{1}{4}\Big\{[1-(1+b)(1)]_+ + [1-(1+b)(1)]_+$$

$$+ [1-(-1+b)(-1)]_+ + [(1-(-1+b)(-1)]_+\Big\} + 1$$

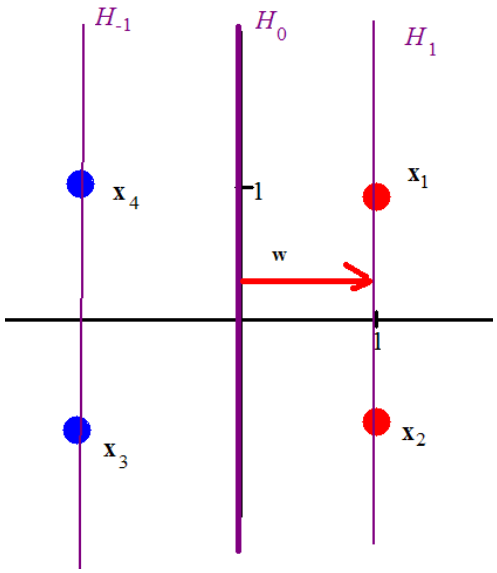
$$= \frac{1}{4}\Big\{[-b]_+ + [-b]_+ + [b]_+ + [b]_+\Big\} + 1$$

$$= \frac{1}{2}\{[-b]_+ + [b]_+\} + 1.$$

Clearly the above is minimized when $b = 0$.

Thus $\mathbf{w} = (1,0);\ b = 0 \ \Rightarrow$

$$f(\mathbf{x}) = \mathbf{w}\cdot\mathbf{x} + b = x_1$$



4. Other choices of kernel

Recall in SVM we have used the kernel

$$K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}.$$

There are many other choices of kernel, e.g.,

$$K(\mathbf{x}, \mathbf{y}) = e^{-|\mathbf{x}-\mathbf{y}|^2} \text{ or } K(\mathbf{x}, \mathbf{y}) = (1 + |\mathbf{x} \cdot \mathbf{y}|)^n$$

note - we must choose a kernel function which is positive definite. How do these choices change the discrimination function $f(\mathbf{x})$ in SVM?

Ex 1: Gaussian kernel

$$K_\sigma(\mathbf{x}, \mathbf{y}) = e^{-\frac{|\mathbf{x}-\mathbf{y}|^2}{2\sigma^2}}$$

[can show pos. def. Mercer kernel]

SVM: from (4) above have

$$f(\mathbf{x}) = \sum_j a_j K(\mathbf{x}, \mathbf{x}_j) + b = \sum_j a_j e^{-\frac{|\mathbf{x}-\mathbf{x}_j|^2}{2\sigma^2}} + b,$$

where examples $\mathbf{x}_j$ in F have known classifications y_j, and a_j, b are obtained by quadratic programming.

What kind of classifier is this? It depends on σ (see Vert movie).

Note Movie1 varies σ in the Gaussian ($\sigma = \infty$ corresponds to a linear SVM); then movie2 varies the margin $\frac{1}{|\mathbf{w}|}$ (in linear feature space F_2) as determined by changing λ or equivalently $C = \frac{1}{2\lambda n}$.

5. Software available

Software which implements the quadratic programming algorithm above includes:

- SVMLight: http://svmlight.joachims.org
- SVMTorch: http://www.idiap.ch/learning/SVMTorch.html
- LIBSVM: http://wws.csie.ntu.edu.tw/~cjlin/libsvm

A Matlab package which implements most of these is Spider:

http://www.kyb.mpg.de/bs/people/spider/whatisit.html

SVM: Interpretations and applications

1. Geometric interpretation

Recall: if

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$$

for some $\mathbf{w} \in F$, we have defined:

$$\|f\|_{\mathcal{H}} = |\mathbf{w}|$$

(independent of b).

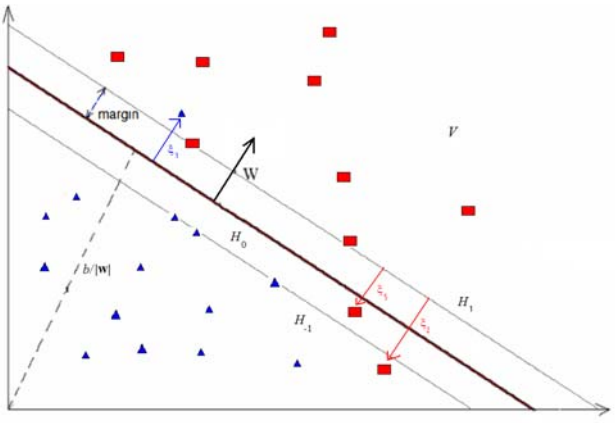


Fig 2: SVM geometry (2 dimensions)

Functional to be minimized:

$$L = \frac{1}{n}\sum_{j=1}^{n}(1 - y_j f(\mathbf{x}_j))_+ + \lambda|\mathbf{w}|^2 \equiv L_d + L_p \qquad \text{(8a)}$$

(minimization over $(\mathbf{w}, b)$).

Why is this a good choice for L? What should λ be?

Consider variables (see (1b) earlier)

$$\xi_j = (1 - y_j f(\mathbf{x}_j))_+.$$

Then

$$L = \frac{1}{n}\sum_{j=1}^{n}\xi_j + \lambda|\mathbf{w}|^2 \tag{8b}$$

In feature space F, define *positive* direction be parallel to **w**, *negative* direction antiparallel to **w**.

For $\mathbf{x} \in F$, value of $f(\mathbf{x}) = \mathbf{w}\cdot\mathbf{x} + b$ determined by $d(\mathbf{x}) =$ distance of **x** from the separating hyperplane

$$H_0 : f(\mathbf{x}) = 0.$$

We assume $d(\mathbf{x})$ positive in *positive* direction (parallel to **w**), negative in negative direction (antiparallel to **w**).

Specifically

$$f(\mathbf{x}) = |\mathbf{w}|d(\mathbf{x})$$

since gradient $\nabla f(\mathbf{x}) = \mathbf{w}$, so f increases along **w** rate $|\mathbf{w}|$ per unit change of **x** in **w** direction.

Note if $y_j = 1$ (i.e., $\mathbf{x}_j$ is in positive class),

$$\xi_j = (1 - |\mathbf{w}|d(\mathbf{x}))_+ = \begin{cases} 0 & \text{if } d(\mathbf{x}) \geq \frac{1}{|\mathbf{w}|} \\ 1 - |\mathbf{w}|d(\mathbf{x}) & \text{if } d(\mathbf{x}) < \frac{1}{|\mathbf{w}|} \end{cases}.$$

Define margin hyperplane (see diagram)

$$H_1 \colon f(\mathbf{x}) = 1.$$

If **x** on *positive* side of H_1 ($d(\mathbf{x}) \geq \frac{1}{|\mathbf{w}|}$):

$$\xi_j = 0,$$

if **x** on *negative* side of H_1:

$$\xi_j = 1 - |\mathbf{w}|d(\mathbf{x}) = +|\mathbf{w}|(\text{distance from } H_1).$$

Thus if $y_j = 1$

$$\xi_j = \begin{cases} 0 & \text{if } \mathbf{x}_j \text{ is on the "correct" side of margin } H_1 \\ |\mathbf{w}| \cdot (\text{distance from } H_1) & \text{if } \mathbf{x}_j \text{ on "wrong" side of } H_1 \end{cases}.$$

Similarly, defining the "negative margin" hyperplane

$$H_{-1} : f(\mathbf{x}) = -1,$$

we have if $y_j = -1$ ($\mathbf{x}_j$ in negative class)

$$\xi_j = \begin{cases} 0 & \text{if } \mathbf{x}_j \text{ is on the "correct" side of margin } H_{-1} \\ |\mathbf{w}| \cdot \text{distance from } H_{-1} & \text{if } \mathbf{x}_j \text{ on "wrong" side of } H_{-1} \end{cases}.$$

Therefore (see above figure)

$$\sum_j \xi_j = |\mathbf{w}| \cdot D$$

with D the total distance of points on the "wrong" sides of their respective margin hyperplanes $H_{\pm 1}$, i.e., $D =$ "total error".

Also:

distance from separating hyperplane H_0 to margin hyperplane $H_1 = \frac{1}{|\mathbf{w}|}$.

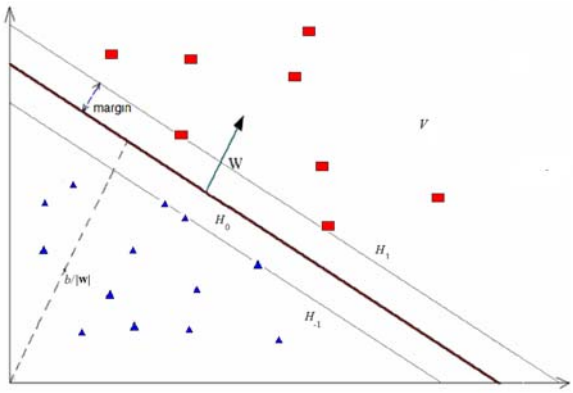
[note: vectors on top of or on wrong side of margins are only ones needed for quadratic programming calculation; these are the *support vectors*]

[fewer support vectors $\Rightarrow$ easier calculation $\Rightarrow$ *sparse* machine]

Conclusion: Minimization of (1) involves a balance between minimizing total error $\sum_j \xi_j$ and the margin width $\frac{1}{|\mathbf{w}|}$, the balance determined by the regularization parameter λ.

2. Special case: Perfect separability

If classes perfectly separable:



Minimizing

$$L = \underbrace{\frac{1}{n}\sum_{j=1}^{n}\xi_j}_{L_d} + \underbrace{\lambda|\mathbf{w}|^2}_{L_p} = L_d + L_p$$

involves maximizing margin $\frac{1}{|\mathbf{w}|}$ and minimizing the total error $\sum_j \xi_j$ with the balance determined by λ.

Choose **w** and b so H_0 bisects the two groups with the maximum "margin" (see diagram above), and the hyperplanes $H_{\pm 1}$ touch closest $\mathbf{x}_j$ to H_0 (such $\mathbf{x}_j$ are *support vectors*).

Then still have

$$\sum_j \xi_j = \text{total error} = 0,$$

while margin $\frac{1}{|\mathbf{w}|}$ is as large as possible.

Recall

$$\xi_j = |\mathbf{w}|(\text{dist. } \mathbf{x}_j \text{ to } H_1)$$

Thus If we *decrease* |**w**| further by increment $d|\mathbf{w}|$ (increase the margin $\frac{1}{|\mathbf{w}|}$),
we will change L by an amount

$$dL = \frac{1}{n} d\left(\sum_{j=1}^{n} \xi_j\right) + \lambda d\big(|\mathbf{w}|^2\big) \equiv dL_d + dL_p.$$

Thus if λ sufficiently small the second term (negative) is outweighed by the first term (positive), and $dL > 0$.

We thus have in perfectly separable case:

Theorem: The **w**, b which minimize (1) yield $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$ whose separating hyperplane $H : f(\mathbf{x}) = 0$ gives the widest margin, if λ is sufficiently small.

Summary: In the general case we choose $\|f\|_{\mathcal{H}} = |\mathbf{w}|,$ and we minimize

$$\sum_{j=1}^{n} \xi_j + \lambda |\mathbf{w}|^2$$

subject to

$$y_j(\mathbf{w} \cdot \mathbf{x} + b) \geq 1 - \xi_j$$

$$\xi_j \geq 0.$$

This is the basic SVM algorithm for finding $f(\mathbf{x})$; see earlier for the QP algorithm which this leads to.

3. The reproducing kernel

As shown earlier the reproducing kernel $K(\mathbf{x}, \mathbf{y})$ for $\mathcal{H}$ above is ordinary dot product of vectors:

$$K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}.$$

Indeed, recall

$$\langle f(\,\cdot\,), K(\mathbf{x}, \,\cdot\,)\rangle = \langle \mathbf{w} \cdot (\,\cdot\,), K(\mathbf{x}, \,\cdot\,)\rangle = \langle \mathbf{w} \cdot (\,\cdot\,), \mathbf{x} \cdot (\,\cdot\,)\rangle = \mathbf{w} \cdot \mathbf{x} = f(\mathbf{x}),$$

as desired. Thus in this case the initial optimization problem yields

$$f(\mathbf{x}) = \sum_{j=1}^{n} a_j K(\mathbf{x}, \mathbf{x}_j) + b = \sum_{j=1}^{n} a_j \mathbf{x} \cdot \mathbf{x}_j + b = \sum_{j=1}^{n} (a_j \mathbf{x}_j) \cdot \mathbf{x} + b,$$

so that the desired **w** must have the form

$$\mathbf{w} = \sum_{j=1}^{n} a_j \mathbf{x}_j,$$

where $\mathbf{x}_j$ are the examples (positive and negative) in the feature space V.

The a_j are computed by the above quadratic programming algorithm.

4. Support vectors:

Note from above that

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b = \sum_{j} (a_j \mathbf{x}_j) \cdot \mathbf{x} + b,$$

or more generally

$$f(\mathbf{x}) = \sum_{j} a_j K(\mathbf{x}_j, \mathbf{x}) + b,$$

with a_j obtained from the above optimization. It can be shown that

$$a_j \neq 0 \text{ iff } \xi_j \neq 0.$$

That is, in the end the only data $\mathbf{x}_j$ which contribute to the separator $f(\mathbf{x})$ are those which are on the "wrong side" of their margins (these are the *support vectors*).

Thus SVM is a *sparse machine*. That is, there are very few terms in the sum defining f, with contributions only from "critical" data points (for which $\xi_j \neq 0$) which are near the hyperplane boundary.

Sparseness leads to very good computational properties - the algorithms are much easier to solve when so few vectors are actually involved in the calculations.

5. Example application: handwritten digit recognition - USPS (Scholkopf, Burges, Vapnik)

Handwritten digits:

Training set: 7300; Test set: 2000

10 class classifier; i^{th} class has a separating SVM function

$$f_i(\mathbf{x}) = \mathbf{w}_i \cdot \mathbf{x} + b_i$$

Chosen class is

$$\text{Class} = \underset{i \in \{0,\ldots,9\}}{\text{argmax}}\, f_i(\mathbf{x}).$$

$$\Phi : \text{digit } g \rightarrow \text{feature vector } \Phi(g) = \mathbf{x} \in F$$

Kernels in feature space F:

RBF: $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{2\sigma^2}}$
Polynomial: $K = (\mathbf{x}_i \cdot \mathbf{x}_j + \theta)^d$
Sigmoidal: $K = \tanh(\kappa(\mathbf{x}_i \cdot \mathbf{x}_j) + \theta)$

Results:

polynomial: $K(\mathbf{x}, \mathbf{y}) = ((\mathbf{x} \cdot \mathbf{y})/256)^{\text{degree}}$

degree	1	2	3	4	5	6
raw error/%	8.9	4.7	4.0	4.2	4.5	4.5
av. # of SVs	282	237	274	321	374	422

RBF: $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\|\mathbf{x} - \mathbf{y}\|^2/(256\ \sigma^2)\right)$

σ^2		1.0	0.8	0.5	0.2	0.1
raw error/%		4.7	4.3	4.4	4.4	4.5
av. # of SVs		234	235	251	366	722

sigmoid: $K(\mathbf{x}, \mathbf{y}) = 1.04 \tanh(2(\mathbf{x} \cdot \mathbf{y})/256 - \Theta)$

Θ		0.9	1.0	1.2	1.3	1.4
raw error/%		4.8	4.1	4.3	4.4	4.8
av. # of SVs		242	254	278	289	296

Computational Biology Applications

References:

T. Golub et al Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression. Science 1999.

S. Ramaswamy et al Multiclass Cancer Diagnosis Using Tumor Gene Expression Signatures. PNAS 2001.

6. Microarrays for cancer classification

Goal: infer cancer genetics by looking at microarray.

Microarray reveals expression patterns and can hopefully be used to discriminate similar cancers, and thus lead to better treatments.

Usual problem: small sample size (e.g., 50 cancer tissue samples), high dimensionality (e.g., 20-30,000). *Curse of dimensionality.*

Example 1: Myeloid vs. Lymphoblastic leukemias

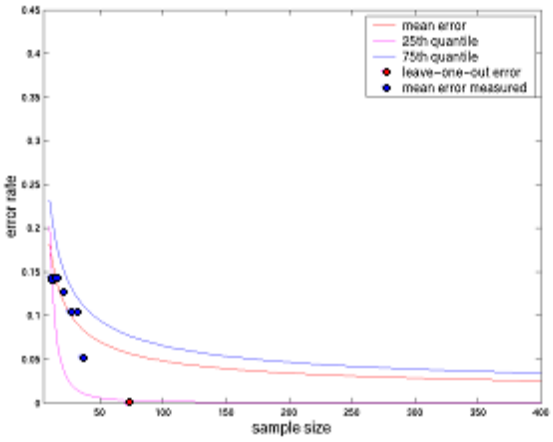
ALL: acute lymphoblastic leukemia
AML: acute myeloblastic leukemia

SVM training: leave one out cross-validation

Dataset	Algorithm	Total Samples	Total errors	Class 1 errors	Class 0 errors	Number Genes
Leukemia Morphology (trest) AML vs ALL	SVM	35	0/35	0/21	0/14	40
	WV	35	2/35	1/21	1/14	50
	k-NN	35	3/35	1/21	2/14	10
Leukemia Lineage (ALL) B vs T	SVM	23	0/23	0/15	0/8	10
	WV	23	0/23	0/15	0/8	9
	k-NN	23	0/23	0/15	0/8	10
Lymphoma FS vs DLCL	SVM	77	4/77	2/32	2/35	200
	WV	77	6/77	1/32	5/35	30
	k-NN	77	3/77	1/32	2/35	250
Brain MD vs Glioma	SVM	41	1/41	1/27	0/14	100
	WV	41	1/41	1/27	0/14	3
	k-NN	41	0/41	0/27	0/14	5

S. Mukherjee

fig. 1: Myeloid and Lymphoblastic Leukemia classification by SVM (other methods are k-nearest neighbors and weighted voting)



S. Mukherjee

fig 2: AML vs. ALL error rates with increasing sample size

In above figure the curves represent error rates with split between training and test sets. Red dot represents leave one out cross-validation.