Lecture 11D (Optional).

MA 751 Part 7

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

$$L(\mathbf{a}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) \equiv \frac{1}{n}\sum_{j=1}^{n}\xi_j + \lambda \mathbf{a}^T K \mathbf{a}$$

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

$$- \sum_{j=1}^{n}\mu_j \xi_j \qquad \text{(4b)}$$

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:

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

$$\frac{\partial L}{\partial \xi_i} = 0 \Rightarrow \frac{1}{n} - \alpha_j - \mu_j = 0. \qquad \text{(6b)}$$

Plugging in get reduced Lagrangian

$$L^*(\mathbf{a}, \boldsymbol{\alpha})$$

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

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

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 (6b) this implies

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

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 $\overline{\boldsymbol{\alpha}}$)

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

(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 (4b).

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

$$\langle \mathbf{w}_1 \cdot \mathbf{x}, \mathbf{w}_2 \cdot \mathbf{x} \rangle = \mathbf{w}_1 \cdot \mathbf{w}_2$$

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

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

Why use this norm? A priori information content.

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

Learning from training data:

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

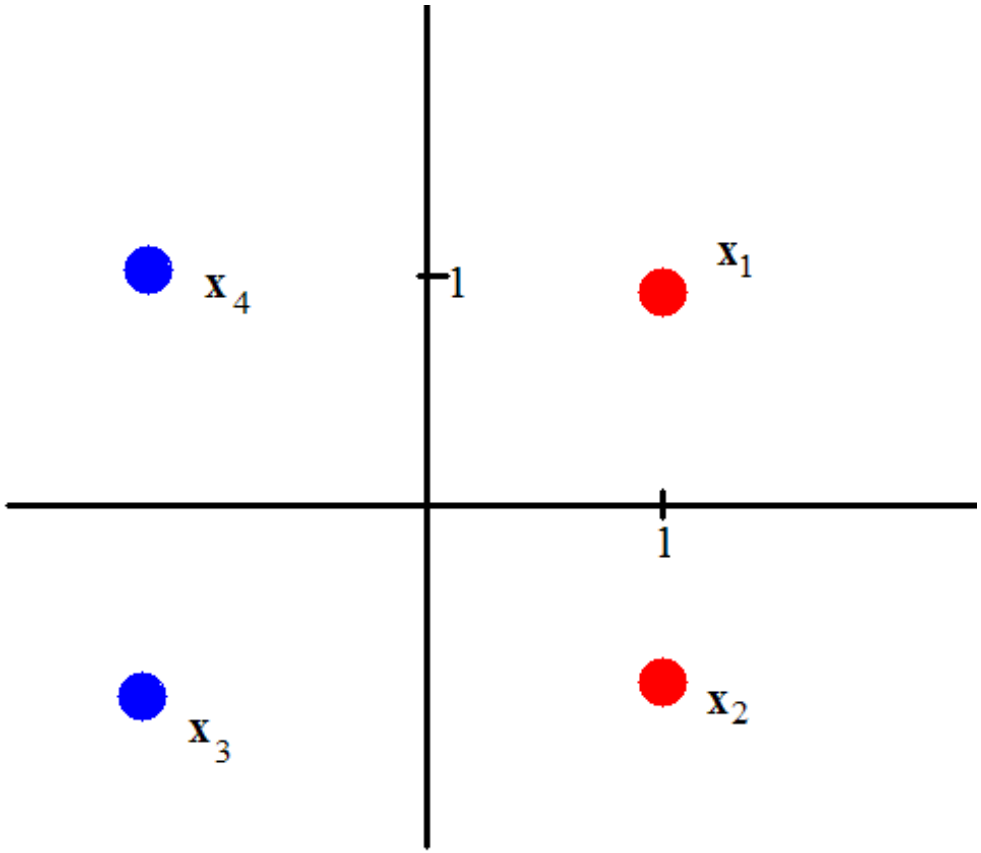
Thus

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

is set of linear separator functions (known as *perceptrons* in neural network theory).

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

3. Toy example:



Information

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

$$[(-1,1),-1],[(-1,-1),-1]\Big\}$$

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

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

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

$$K(\mathbf{x}_i, \mathbf{x})$$

so

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

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

(we let $\lambda = 1/2$; minimize wrt $\mathbf{w}$, 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 $\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_d \end{bmatrix}$.

Formulate

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

$$= \frac{1}{4}\sum_{j=1}^{4}\xi_i + \left(\sum_{j=1}^{4} a_i^2\right) - 2(a_1 a_3 + a_2 a_4)$$

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

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

$$= \frac{1}{4}\sum_{j=1}^{4}\xi_i + \frac{1}{2}\mathbf{a}^T K \mathbf{a}$$

$$-\sum_{j=1}^{4}\alpha_j\Big[\big((K\mathbf{a})_j + b\big)y_j - 1 + \xi_j\Big] - \sum_{j=1}^{4}\mu_j\xi_j$$

$$= \frac{1}{4}\sum_{j=1}^{4}\xi_j + \frac{1}{2}\mathbf{a}^T K \mathbf{a} - \boldsymbol{\alpha}^T Y K \mathbf{a} - b\boldsymbol{\alpha}^T \mathbf{y}$$

$$+ \sum_{j=1}^{4}\alpha_j - \boldsymbol{\alpha}\cdot\boldsymbol{\xi} - \boldsymbol{\mu}\cdot\boldsymbol{\xi} \quad (10)$$

with constraints

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

Solution has (see (7) above)

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

(recall

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

)

and (7a) above)

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

Finally optimize (8):

$$L_1 = \sum_{i=1}^{4} \overline{\alpha}_i - \frac{1}{2} \overline{\boldsymbol{\alpha}}^T P \overline{\boldsymbol{\alpha}},$$

where

$$P = Y K Y^T$$

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

$$\times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

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

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

constraint is

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

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

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

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

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

$$+ 1$$

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

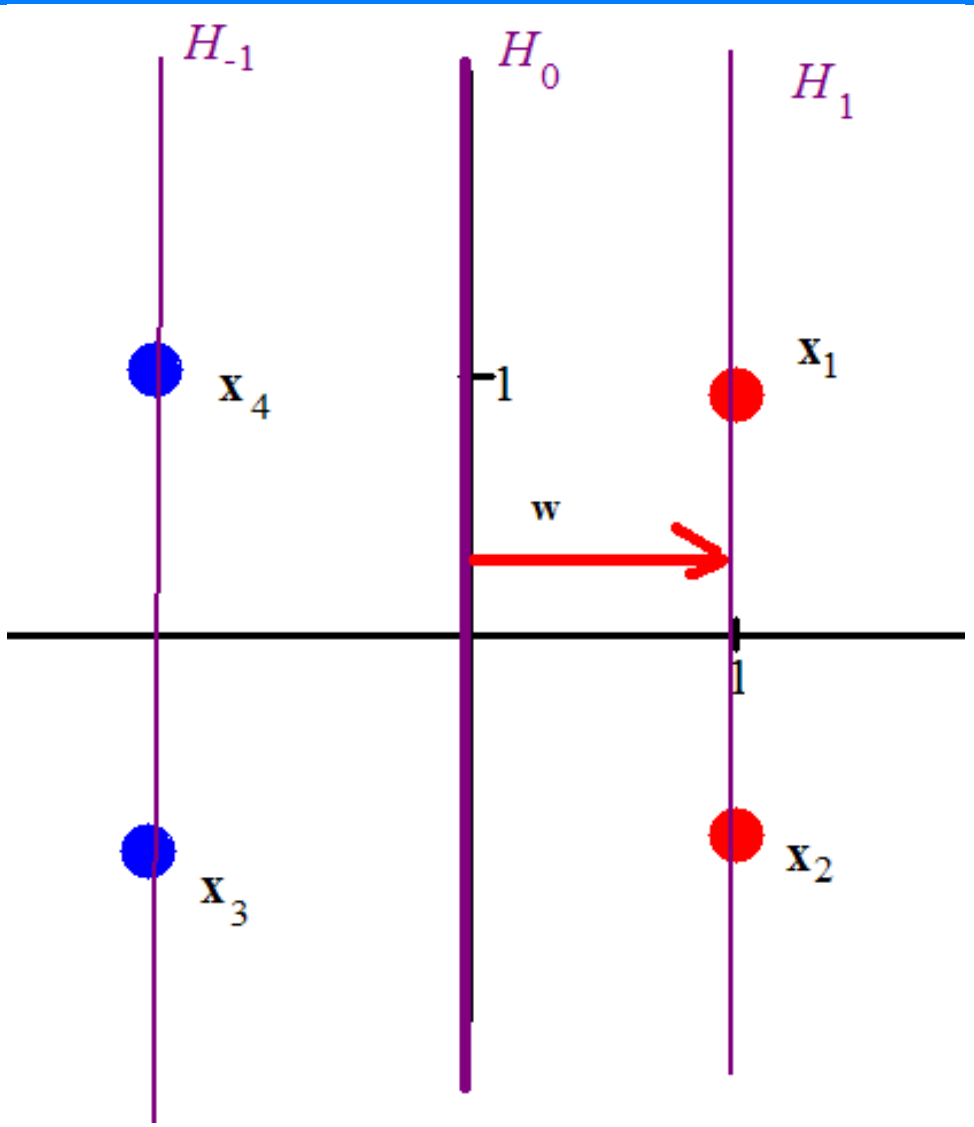
$$= \frac{1}{2}\big\{[-b]_+ + [b]_+\big\} + 1.$$

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

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

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

Example



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.,

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

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

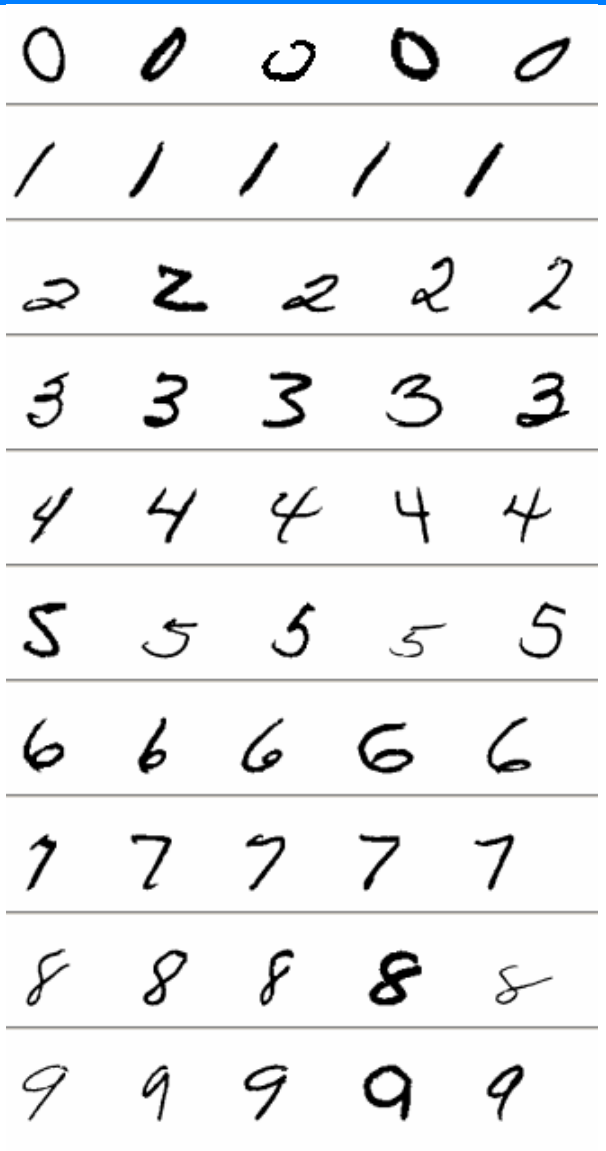
A Matlab package which implements most of these is Spider:

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

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

Handwritten digits:

Other kernels



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

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

Φ : digit $g \rightarrow$ 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.

7. Gene expression arrays for cancer classification

Goal: infer cancer genetics by looking at microarray.

Gene expression array 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

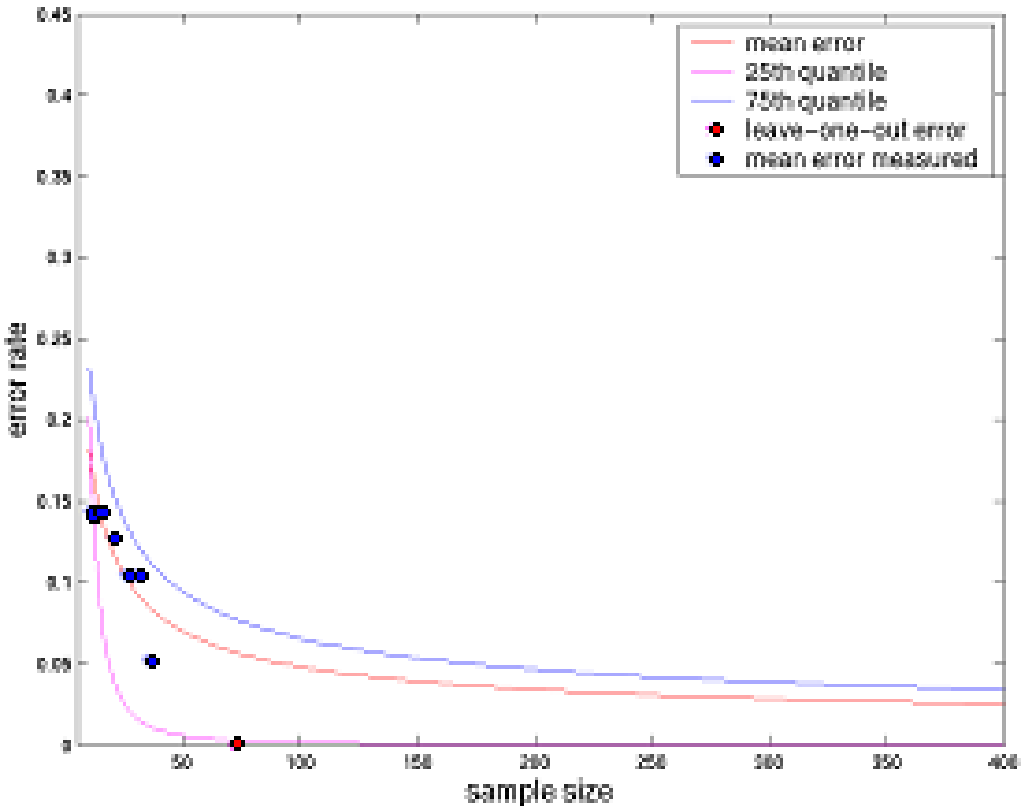
ALL: acute lymphoblastic leukemia
AML: acute myeloblastic leukemia

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



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.