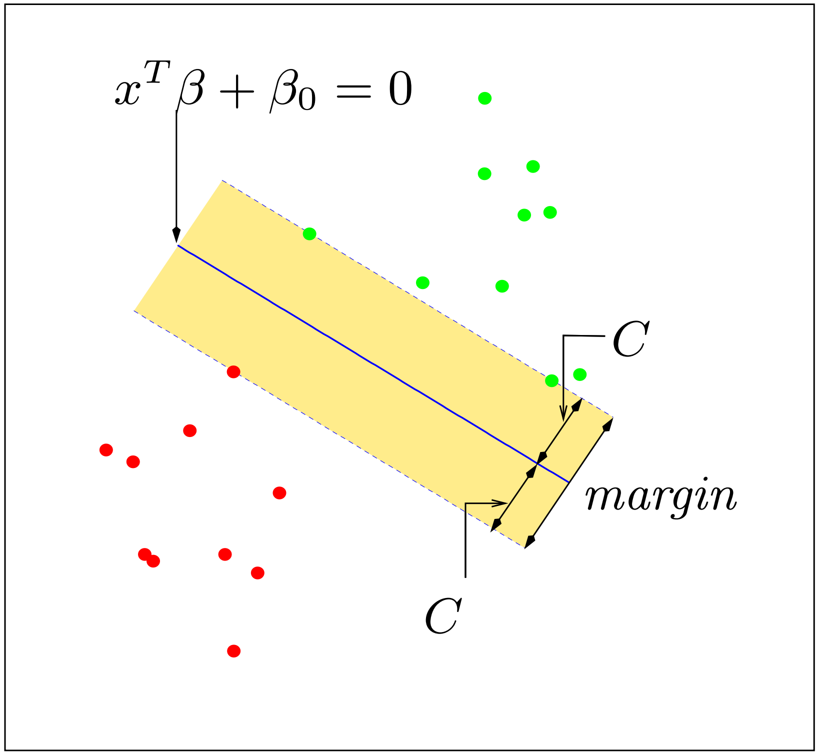
Support Vector Machines

Based on ESL and papers by Vladimir Vapnik, Trevor Hastie, Saharon Rosset, Rob Tibshirani, Ji Zhu

Outline

- Optimal separating hyperplanes and relaxations
- SVMs and kernel inner-products
- SVM as a function estimation problem
- LARS- style algorithm for SVMs

Maximum Margin Classifier



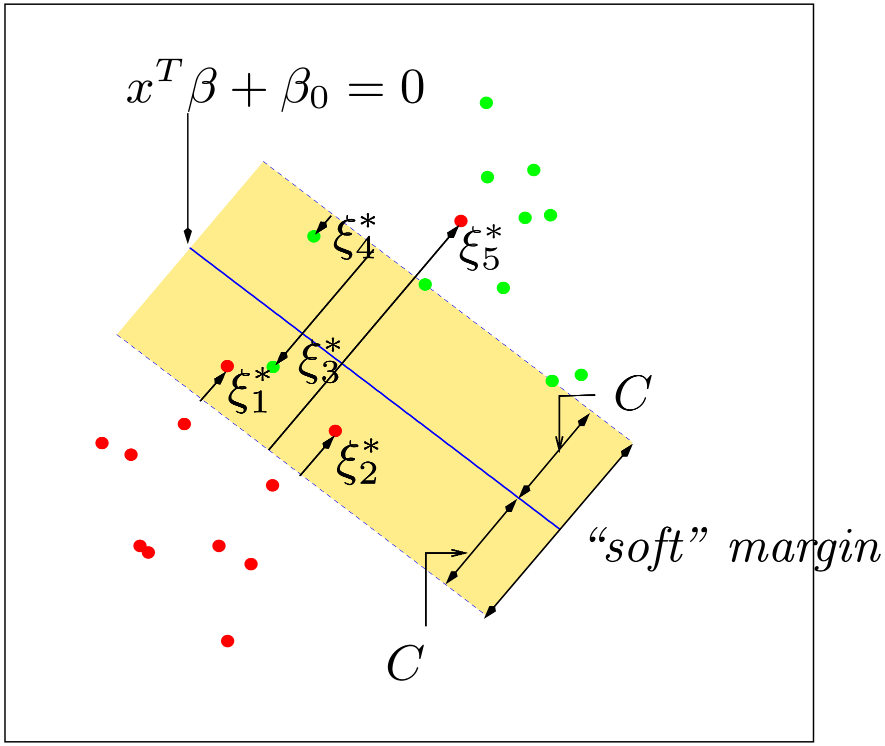
Vapnik(1995)

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

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

subject to $\quad y_i(x_i^T\beta + \beta_0) \geq C, \; i = 1, \ldots, N.$

Overlapping Classes



$\xi_i^* = C\xi_i$

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

subject to $y_i(x_i^T\beta + \beta_0) \geq C(1 - \xi_i),\ \ \xi_i \geq 0,\ \ \sum_i \xi_i \leq B$

Equivalent form of problem

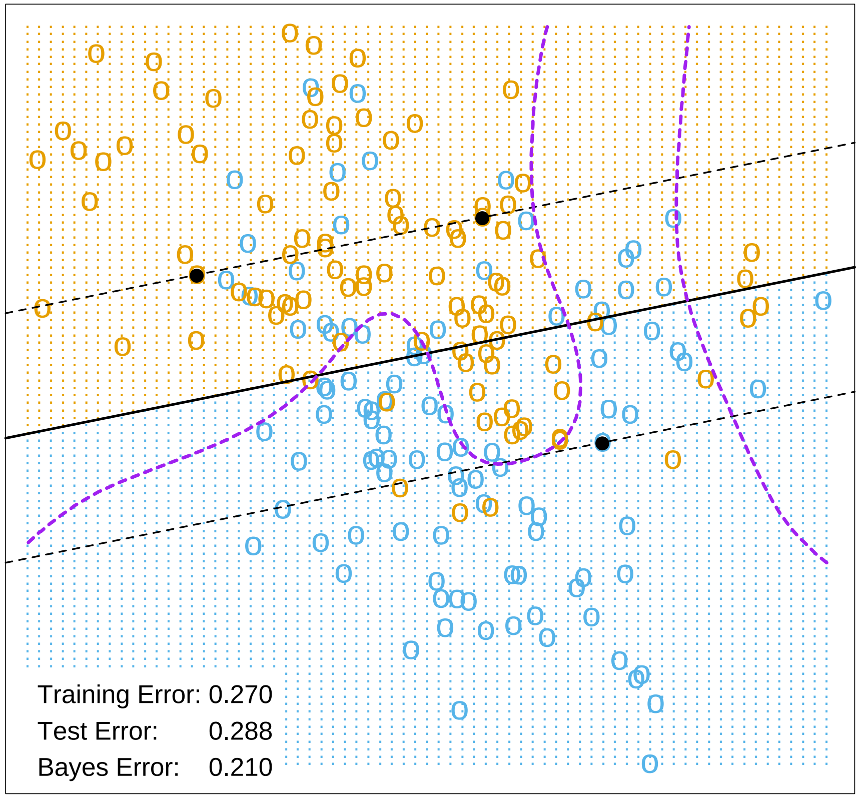
Define $C = 1/||\beta||$ and drop norm constraint on β, as in ESL sec 4.2:

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

subject to $y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i,\;\; \xi_i \geq 0,\;\; \sum_i \xi_i \leq B$

This is the original form given by Vapnik; we find it confusing due to the fixed scale"1" in the constraint.

Example



Fitted function is $\hat{f}(x) = x^T\hat{\beta} + \hat{\beta}_0$

Resulting classifier is $\hat{G}(x) = \text{sign}[\hat{f}(x)]$

Quadratic Programming Solution

After a lot of *stuff* we arrive at a Lagrange dual

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'}$$

which we maximize subject to constraints (involving B as well).

The solution is expressed in terms of fitted Lagrange multipliers $\hat{\alpha}_i$:

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i$$

Some fraction of $\hat{\alpha}_i$ are exactly zero (from KKT conditions); the x_i for which $\hat{\alpha}_i > 0$ are called support points $\mathcal{S}$.

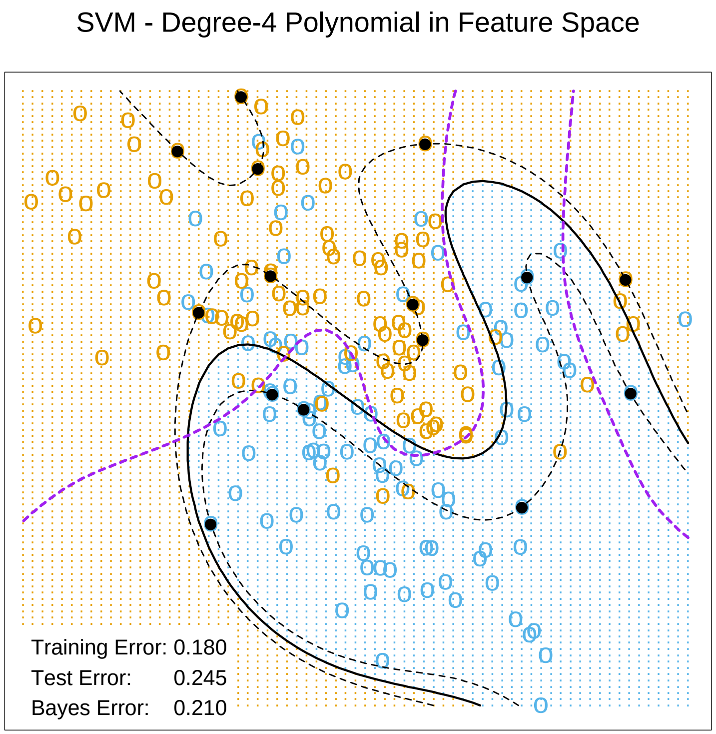
$$\hat{f}(x) = x^T \hat{\beta} + \hat{\beta}^0 = \sum_{i \in \mathcal{S}} \hat{\alpha}_i y_i x^T x_i + \hat{\beta}^0$$

Microarray example

$16,063$ genes; 144 training, 54 test samples, 14 classes

Methods	CV errors (SE) Out of 144	Test errors Out of 54	Number of Genes Used
1. Nearest shrunken centroids	35 (5.0)	17	6,520
2. L_2-penalized discriminant analysis	25 (4.1)	12	16,063
3. Support vector classifier	26 (4.2)	14	16,063
4. Lasso regression (one vs all)	30.7 (1.8)	12.5	1,429
5. k-nearest neighbors	41 (4.6)	26	16,063
6. L_2-penalized multinomial	26 (4.2)	15	16,063
7. L_1-penalized multinomial	17 (2.8)	13	269
8. Elastic-net penalized multinomial	22 (3.7)	11.8	384

Flexible Classifiers



Enlarge the feature space via basis expansions, e.g. polynomials of total degree 4. $h(x) = (h_1(x), h_2(x), \ldots, h_M(x))$

$$\hat{f}(x) = h(x)^T \hat{\beta} + \hat{\beta}_0$$

The kernel trick

- Consider the ridge regression prediction

$$\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_p)^{-1}\mathbf{X}^T\mathbf{y}$$

  If p is large, computations with $\mathbf{X}^T\mathbf{X} + \lambda I$ ($p \times p$ matrix) may be daunting. Instead write this as

$$\hat{\mathbf{y}} = (\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_N)^{-1}\mathbf{X}\mathbf{X}^T\mathbf{y}$$

- matrix is now only $N \times N$, and if N is small, this is much simpler computationally.
- note that we need only to compute inner products of the observations, to compute $\mathbf{X}\mathbf{X}^T$ This argument also applies if X represents not the original features but transformations of them. Hence we only need to define inner products in the transformed space; we don't need to actually transform the features!

SVM and Kernels

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \langle h(x_i), h(x_{i'})\rangle$$

$$\begin{aligned} f(x) &= h(x)^T\beta + \beta_0 \\ &= \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i)\rangle + \beta_0. \end{aligned}$$

L_D and solution $f(x)$ involve $h(x)$ only through inner-products

$$K(x, x') = \langle h(x), h(x')\rangle$$

Given a suitable positive kernel $K(x, x')$, don't need $h(x)$ at all!

$$\hat{f}(x) = \sum_{i \in \mathcal{S}} \hat{\alpha}_i y_i K(x, x_i) + \hat{\beta}_0$$

Popular Kernels

$K(x, x')$ is a symmetric, positive (semi-)definite function.

$$d\text{th deg. poly.: } K(x, x') = (1 + \langle x, x' \rangle)^d$$

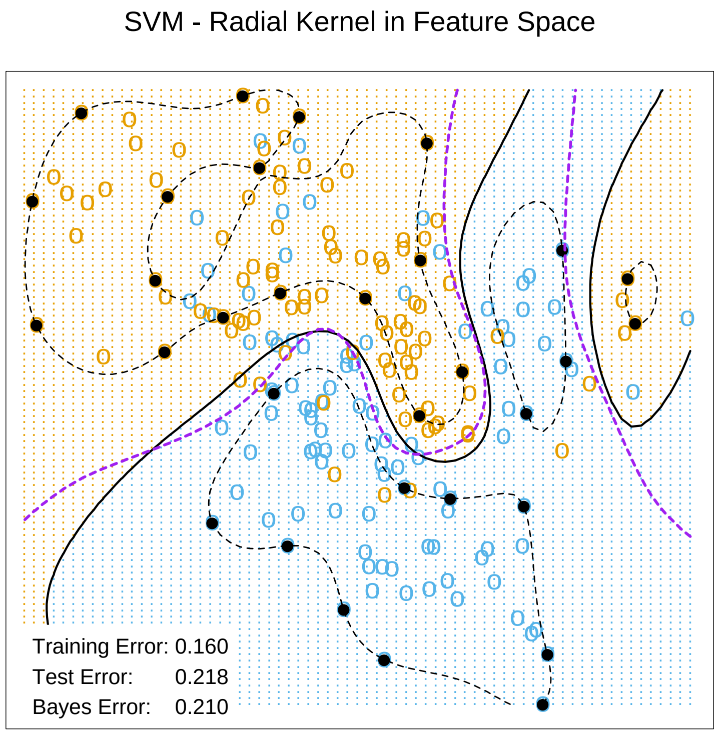
$$\text{radial basis: } K(x, x') = \exp(-\|x - x'\|^2/c)$$

Example: 2nd degree polynomial in $\mathbb{R}^2$.

$$\begin{aligned} K(x, x') &= (1 + \langle x, x' \rangle)^2 \\ &= (1 + x_1 x_1' + x_2 x_2')^2 \\ &= 1 + 2x_1 x_1' + 2x_2 x_2' + (x_1 x_1')^2 + (x_2 x_2')^2 + 2x_1 x_1' x_2 x_2' \end{aligned}$$

Then $M = 6$, and if we choose
$h_1(x) = 1$, $h_2(x) = \sqrt{2}x_1$, $h_3(x) = \sqrt{2}x_2$, $h_4(x) = x_1^2$, $h_5(x) = x_2^2$,
and $h_6(x) = \sqrt{2}x_1 x_2$,
then $K(x, x') = \langle h(x), h(x') \rangle$.

Dim $h(x)$ infinite



- Fraction of support points depends on overlap; here 45%.
- The smaller B, the smaller the overlap, and more wiggly the function.
- B controls generalization error.

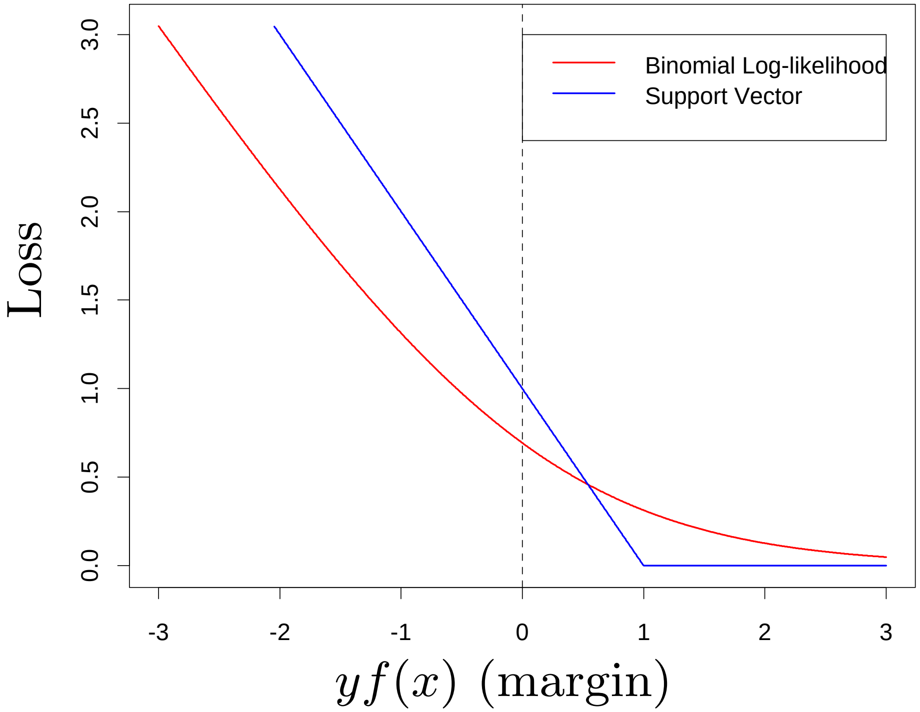
Curse of Dimensionality

Support Vector Machines can suffer in high dimensions.

		Test Error (SE)	
	Method	No Noise Features	Six Noise Features
1	SV Classifier	0.450 (0.003)	0.472 (0.003)
2	SVM/poly 2	0.078 (0.003)	0.152 (0.004)
3	SVM/poly 5	0.180 (0.004)	0.370 (0.004)
4	SVM/poly 10	0.230 (0.003)	0.434 (0.002)
5	BRUTO	0.084 (0.003)	0.090 (0.003)
6	MARS	0.156 (0.004)	0.173 (0.005)
	Bayes	0.029	0.029

The addition of 6 noise features to the 4-dimensional feature space causes the performance of the SVM to degrade. The true decision boundary is the surface of a sphere, hence a quadratic monomial (additive) function is sufficient.

SVM via Loss + Penalty



With $f(x) = h(x)^T\beta + \beta_0$ and $y_i \in \{-1, 1\}$, consider

$$\min_{\beta_0,\,\beta} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \lambda \|\beta\|^2$$

Solution identical to SVM solution, with $\lambda = \lambda(B)$.

In general $\displaystyle\min_{\beta_0,\,\beta} \sum_{i=1}^{N} L[y_i, f(x_i)] + \lambda \|\beta\|^2$

Loss Functions

For $Y \in \{-1, 1\}$

Log-likelihood: $L[Y, f(X)] = \log\left(1 + e^{-Yf(X)}\right)$

- (negative) binomial log-likelihood or deviance.
- estimates the logit

$$f(X) = \log \frac{\Pr(Y = 1|X)}{\Pr(Y = -1|X)}$$

SVM: $L[Y, f(X)] = (1 - Yf(X))_{+}$.

- Called "hinge loss"
- Estimates the classifier (threshold)

$$C(x) = \text{sign}\left(\Pr(Y = 1|X) - \frac{1}{2}\right)$$

SVM and Function Estimation

SVM with general kernel K minimizes:

$$\sum_{i=1}^{N}(1 - y_i f(x_i))_+ + \lambda \|f\|^2_{\mathcal{H}_K}$$

with $f = b + h$, $h \in \mathcal{H}_K$, $b \in \mathcal{R}$. $\mathcal{H}_K$ is the reproducing kernel Hilbert space (RKHS) of functions generated by the kernel K. The norm $\|f\|_{\mathcal{H}_K}$ is generally interpreted as a roughness penalty.

More generally we can optimize

$$\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda \|f\|^2_{\mathcal{H}_K}$$

Quadratic Programming (path algorithm)

$$L_P: \sum_{i=1}^{N} \xi_i + \frac{\lambda}{2}\beta^T\beta + \sum_{i=1}^{N} \alpha_i(1 - y_i f(x_i) - \xi_i) - \sum_{i=1}^{N} \gamma_i \xi_i$$

$$\frac{\partial}{\partial \beta}: \qquad \beta = \frac{1}{\lambda}\sum_{i=1}^{N} \alpha_i y_i x_i$$

$$\frac{\partial}{\partial \beta_0}: \qquad \sum_{i=1}^{N} y_i \alpha_i = 0,$$

along with the KKT conditions

$$\begin{aligned} \alpha_i(1 - y_i f(x_i) - \xi_i) &= 0 \\ \gamma_i \xi_i &= 0 \\ 1 - \alpha_i - \gamma_i &= 0 \end{aligned}$$

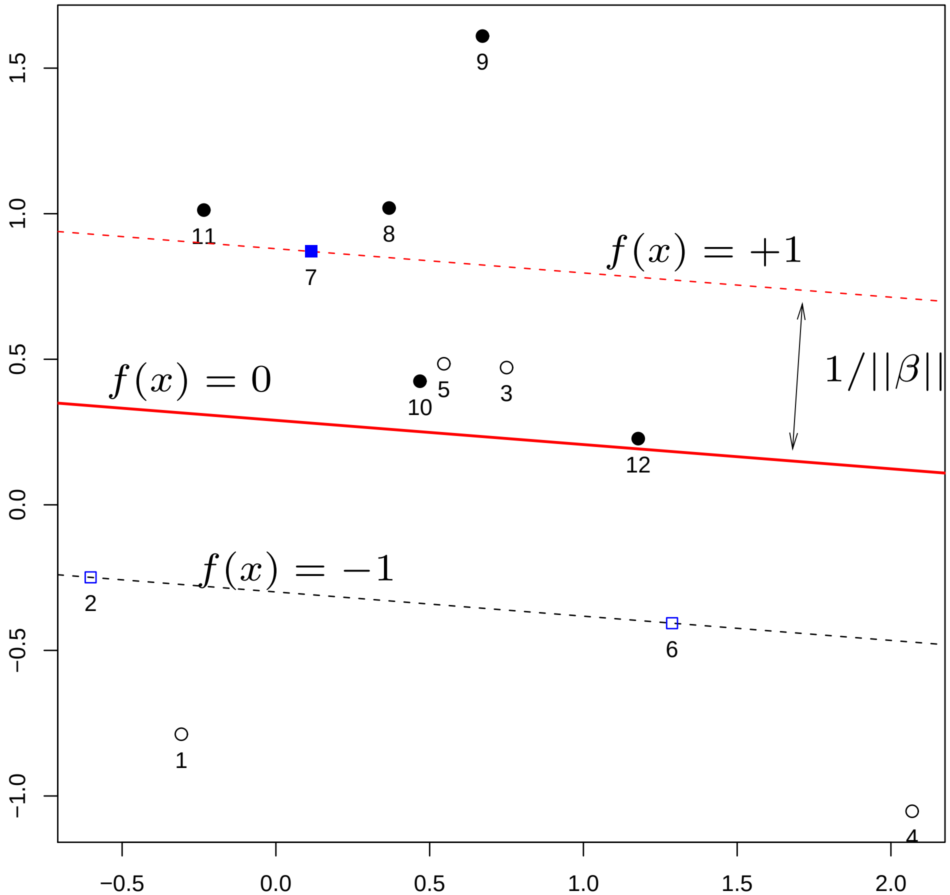
Implications of the KKT conditions

Observations are in one of three states:

- $\mathcal{L} = \{i : y_i f(x_i) < 1,\ \alpha_i = 1\}$, $\mathcal{L}$ for Left of the elbow
- $\mathcal{E} = \{i : y_i f(x_i) = 1,\ 0 \le \alpha_i \le 1\}$, $\mathcal{E}$ for Elbow
- $\mathcal{R} = \{i : y_i f(x_i) > 1,\ \alpha_i = 0\}$, $\mathcal{R}$ for Right of the elbow

- Start with λ large, and the margin very wide. All $\alpha_i = 1$ (if $N_+ = N_-$). As $\lambda \downarrow 0$, the margin gets narrower.
- For the narrowing margin to pass through a point, it's α has to change from 1 to 0 (or from 0 to 1). While this is happening, the point has to linger on the margin. Hence the point moves from $\mathcal{L}$ to $\mathcal{R}$ via $\mathcal{E}$.
- The condition $\sum_i y_i \alpha_i = 0$ demands a certain balance on opposite margins.

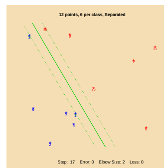
Example

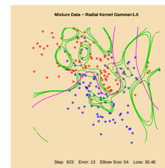


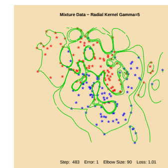
- $\lambda = 0.5$, and the width of the soft margin is $2/||\beta|| = 2 \times 0.587$.
- Two hollow points $\{3, 5\}$ are misclassified, while the two solid points $\{10, 12\}$ are correctly classified, but on the wrong side of their margin $f(x) = +1$; each of these has $\xi_i > 0$.
- The three square shaped points $\{2, 6, 7\}$ are exactly on the margin.

The Path

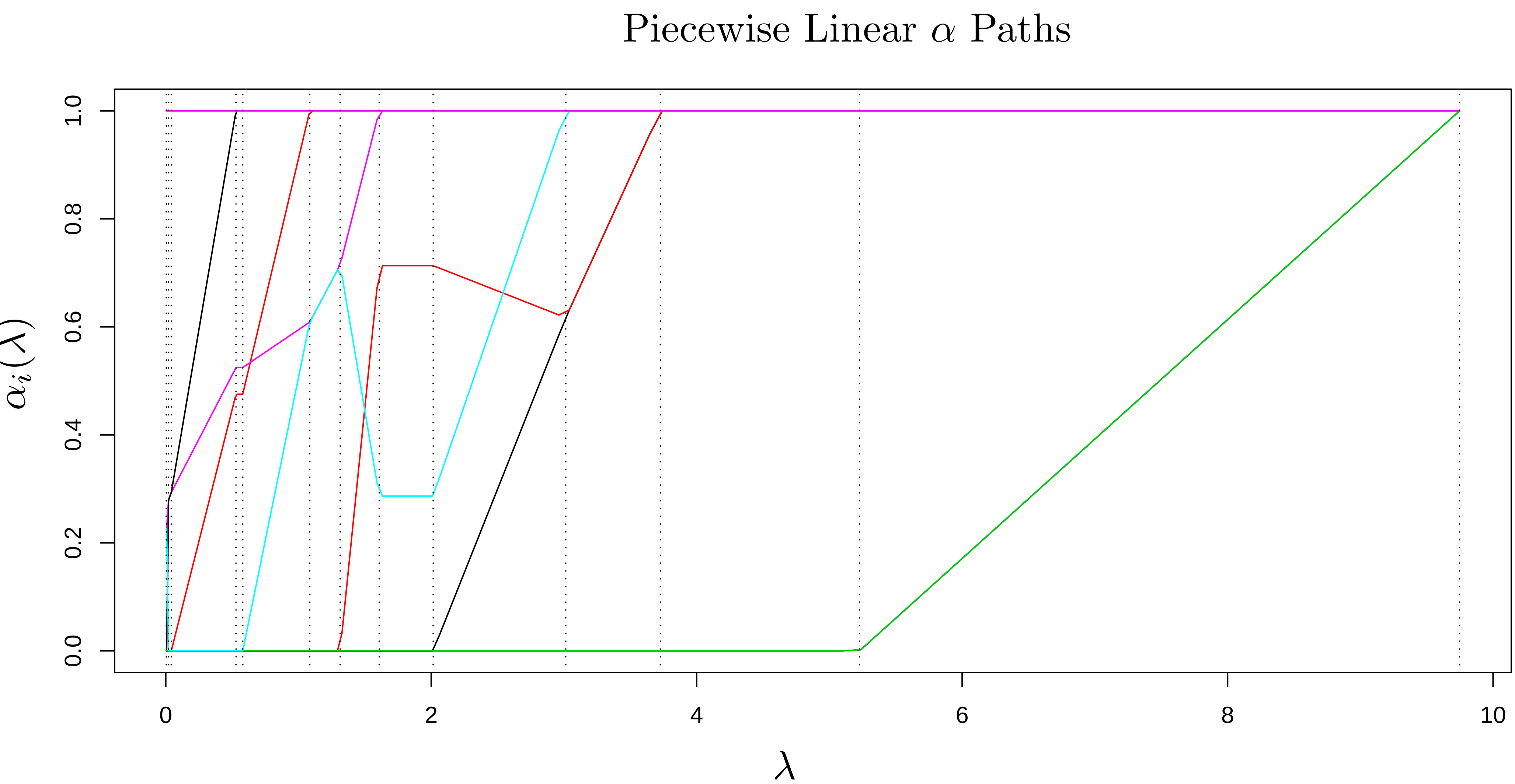
- The α_i are piecewise-linear in λ (or $1/C$)



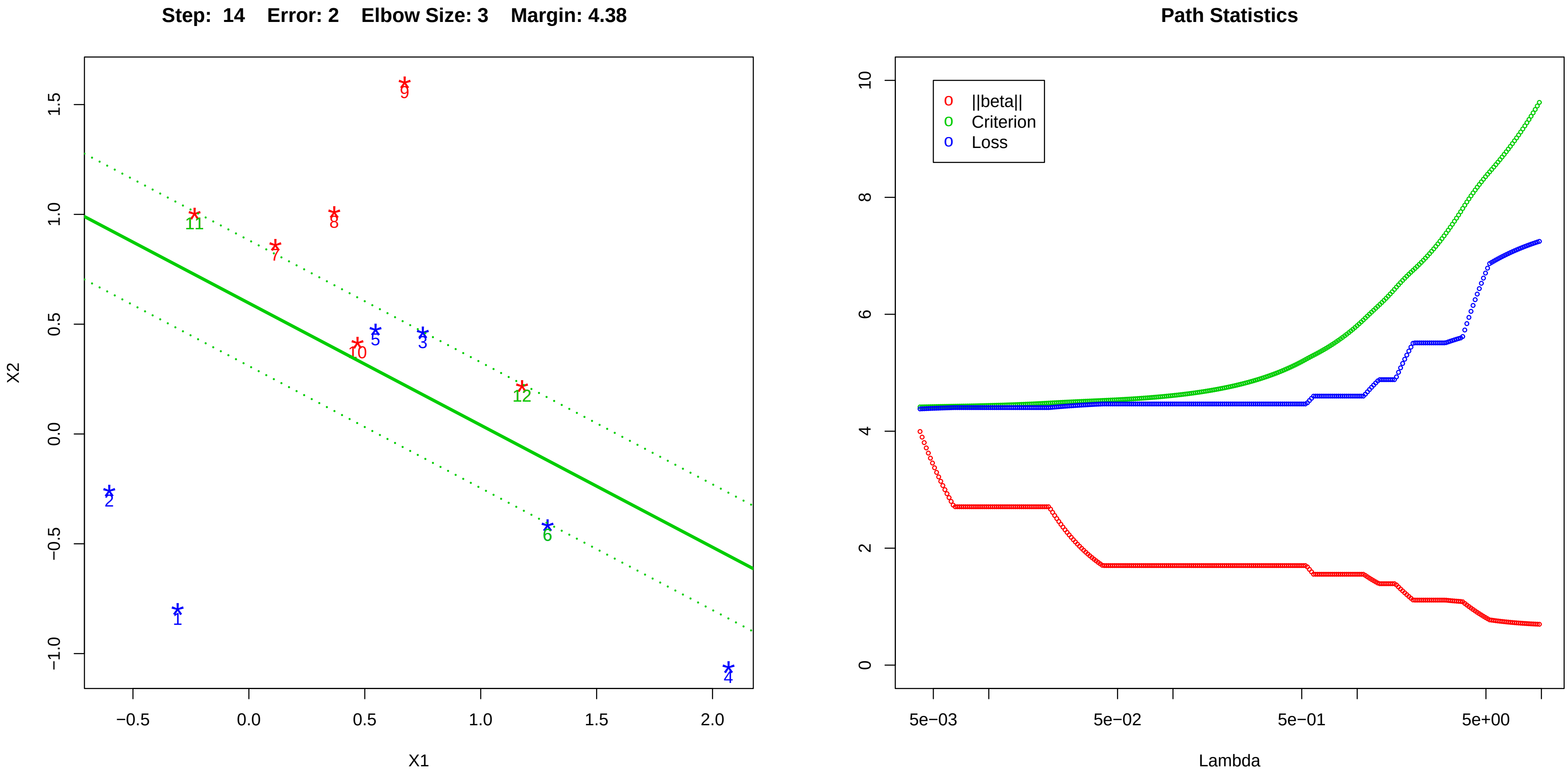




- The points in $\mathcal{E}$ characterize these paths, since points must stay on the margin $(y_i f(x_i) = 1)$ while their α_i lie in $(0, 1)$.
- Points can revisit the margin more than once.
- The coefficients β_0 and β are piecewise-linear in $C = 1/\lambda$. (LARS, Efron et. al., 2002): quadratic criterion, L_1 constraint.
- The margins can stay wedged while their α_i change, if they are "loaded to capacity".
- For non-separable data, the loss $\sum_i \xi_i$ achieves a minimum value, with a positive margin.

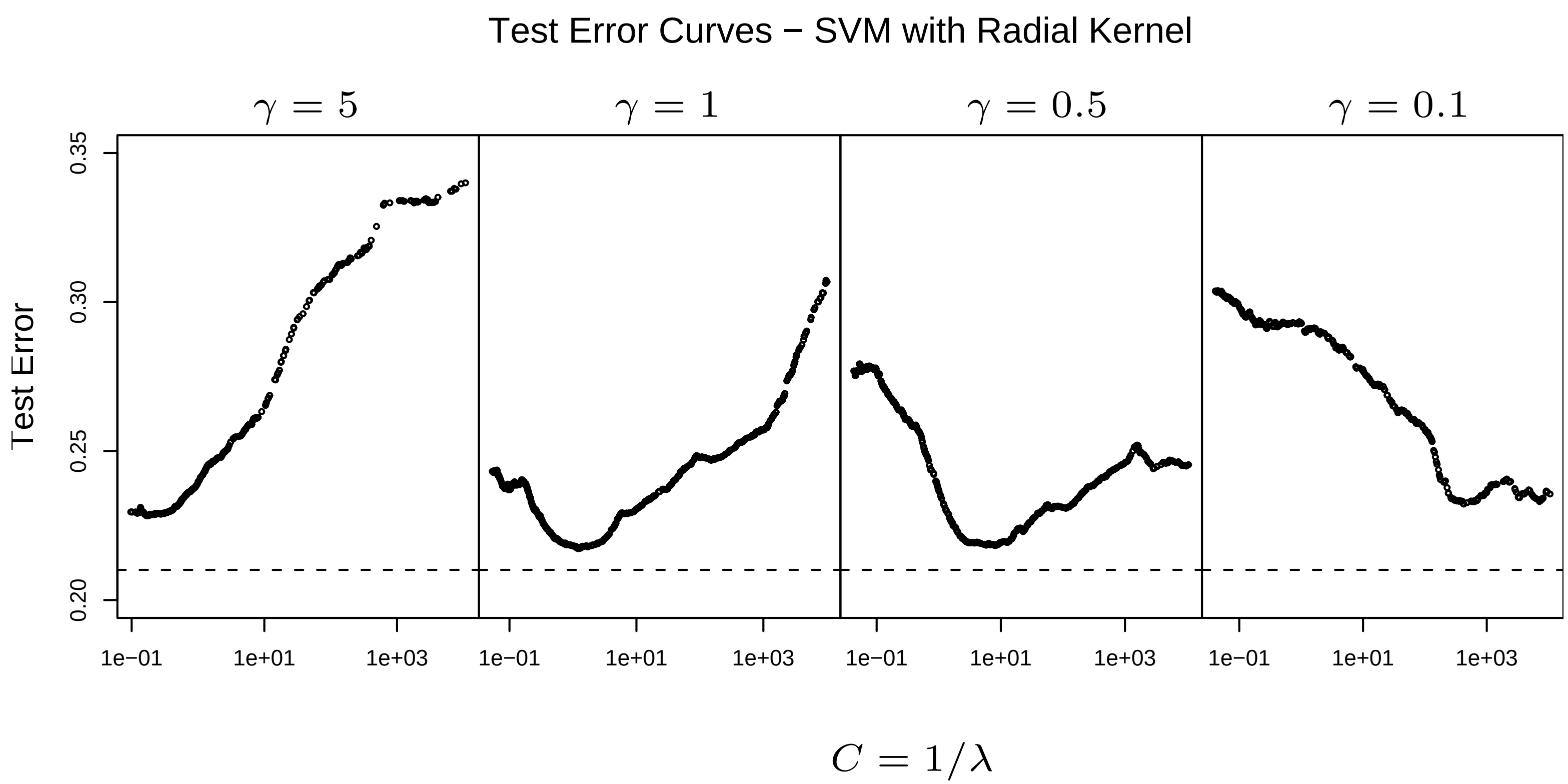


Piecewise Linear α Paths
$\alpha_i(\lambda)$
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10
λ



Step: 14 Error: 2 Elbow Size: 3 Margin: 4.38
X1
X2
1
2
3
4
5
6
7
8
9
10
11
12
Path Statistics
||beta||
Criterion
Loss
Lambda
5e−03
5e−02
5e−01
5e+00

The Need for Regularization



- γ is a kernel parameter: $K(x, z) = \exp(-\gamma||x - z||^2)$.
- λ (or C) are regularization parameters, which have to be determined using some means like cross-validation.

SVMs for regression

- Linear regression model:

$$f(x) = x^T\beta + \beta_0, \tag{1}$$

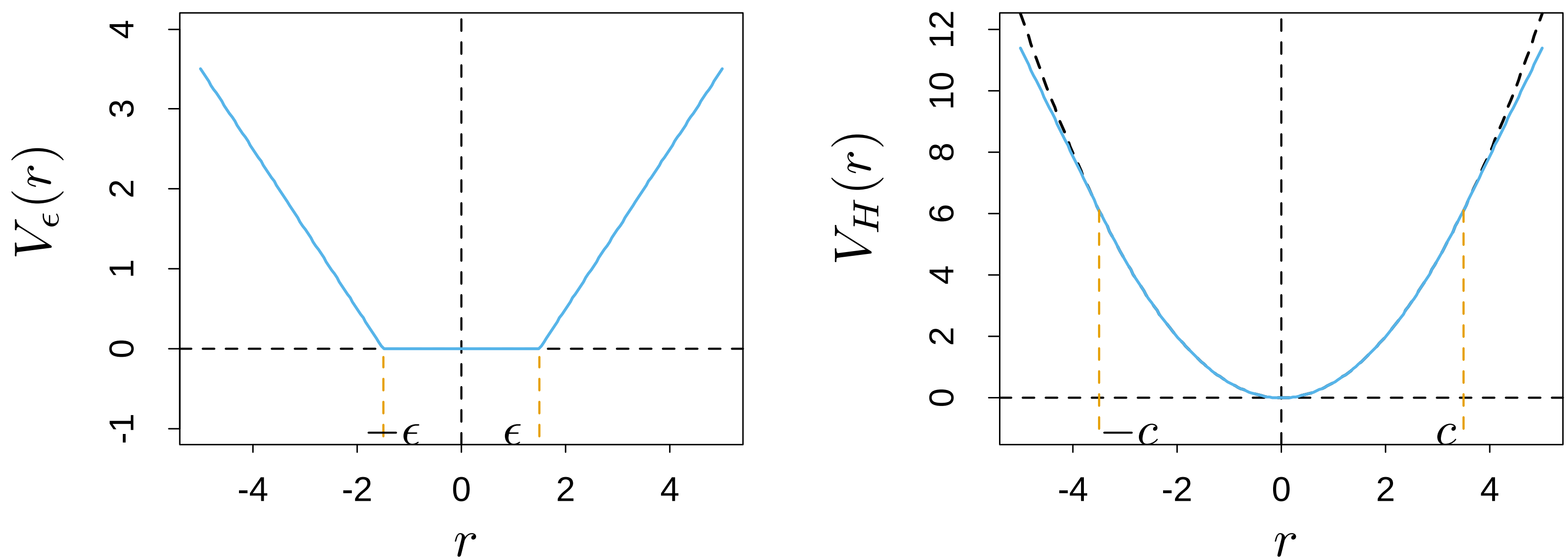
- To estimate β, we consider minimization of

$$H(\beta, \beta_0) = \sum_{i=1}^{N} V(y_i - f(x_i)) + \frac{\lambda}{2}\|\beta\|^2, \tag{2}$$

where

$$V_\epsilon(t) = \begin{cases} 0 & \text{if } |t| < \epsilon, \\ |t| - \epsilon, & \text{otherwise.} \end{cases} \tag{3}$$

This is called "ϵ-insensitive" error measure.



The left panel shows the ϵ-insensitive error function used by the support vector regression machine. The right panel shows the error function used in Huber's robust regression (green curve). Beyond $|c|$, the function changes from quadratic to linear.

Software

- In R, `e1071` package and `library(svmpath)` available from `CRAN`.
- Many other packages fit SVMs