- assume two classes only; change notation so that $y = \pm 1$
- use *linear* combinations of inputs to predict y

$$y = \begin{cases} -1 & as \quad \beta_0 + x^T\beta < 0 \\ +1 & \quad \beta_0 + x^T\beta > 0 \end{cases}$$

- misclassification error $D(\beta, \beta_0) = -\Sigma_{i \in \mathcal{M}} y_i(\beta_0 + x_i^T\beta)$ where
- $\mathcal{M} = \{j; y_j(\beta_0 + x_j^T\beta) < 0\}$
- note that $D(\beta) > 0$ and proportional to the 'size' of $\beta_0 + x_i^T\beta$
- Can show that an algorithm to minimize $D(\beta, \beta_0)$ exists and converges to the plane that separates $y = +1$ from $y = -1$ if such a plane exists
- But it will cycle if no such plane exists and be very slow if the 'gap' is small

- Also if one plane exists there is likely many (Figure 4.13)
- The plane that defines the ”largest” gap is defined to be ”best”
- can show that this needs to

$$\begin{aligned} &\min_{\beta_0,\beta} \frac{1}{2}||\beta||^2 \\ \text{s.t.} \quad & y_i(\beta_0 + x_i^T\beta) \geq 1, \quad i = 1, \dots N \end{aligned} \qquad (4.44)$$

- See Figure 4.15
- the points on the edges (margin) of the gap called support points or support vectors; there are typically many fewer of these than original points
- this is the basis for the development of Support Vector Machines (SVM), more later
- sometimes add features by using basis expansions; to be discussed first in the context of regression (Chapter 5)

- plane in R^p defined by $f(\underline{x}) = \beta_0 + \underline{\beta}^T \underline{x} = 0$
- call this plane $\mathcal{L}$
  1. For any $x_0 \in \mathcal{L}, \beta_0 = -\beta^T x_0$
  2. if $x_1, x_2 \in \mathcal{L}$ then $\beta^T(x_1 - x_2) = 0$, i.e. $\beta \perp (x_1 - x_2)$, i.e. $\frac{\beta}{||\beta||} \perp \mathcal{L}$
  3. if $x \in R^p$ the distance to the closest point in $\mathcal{L}$, x_0, say, is

$$\frac{\beta^T}{||\beta||}(x - x_0) = \frac{\beta^T x + \beta_0}{||\beta||} = \frac{f(x)}{||\beta||}$$

- a perceptron returns $\mathrm{sign}(\beta_0 + \beta^T x)$
- The perceptron learning algorithm tries to find $\mathcal{L}$ by minimizing the number of misclassifications
-

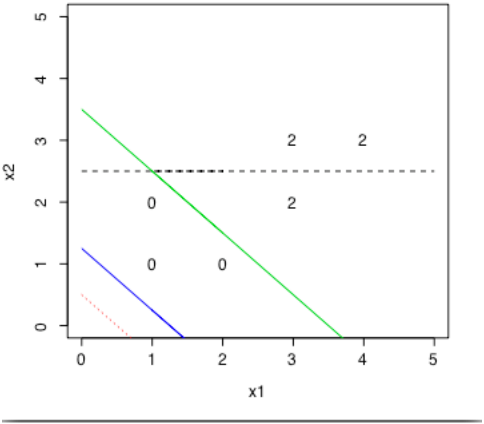
$$\min D(\beta, \beta_0) = -\sum_{i \in \mathcal{M}} y_i (x_i^T + \beta_0)$$

- Algorithm based on derivatives of D: visit points in M in some order
- update using

$$\begin{pmatrix} \beta^{(t)} \\ \beta_0^{(t)} \end{pmatrix} = \begin{pmatrix} \beta^{(t-1)} \\ \beta_0^{(t-1)} \end{pmatrix} + \rho \begin{pmatrix} y_i x_i \\ y_i \end{pmatrix}$$

- ρ called the learning rate
- if classes are linearly separable, this converges in a finite number of steps
- the separating hyperplane with the largest margin has

$$\max_{\beta, \beta_0, ||\beta||=1} C \qquad s.t. \quad y_i(x_i^T \beta + \beta_0) > C \quad \forall i$$


x2
x1
0
1
2
3
4
5
0
1
2
3
4
5
2
2
0
2
0
0

Finding the optimal separating plane: a quadratic programming problem:

$$\max_{\beta,\beta_0} C \qquad s.t \quad . \quad y_i(x_i^T\beta + \beta_0) \geq C||\beta||$$

$$\max \frac{C}{||\beta||} \qquad s.t. \qquad y_i(x_i^T\beta + \beta_0) \geq C$$

$$\max \frac{1}{||\beta||} \qquad s.t. \qquad y_i(x_i^T\beta + \beta_0) \geq 1$$

$$\min ||\beta|| \qquad s.t. \qquad y_i(x_i^T\beta + \beta_0) \geq 1$$

$$\min \frac{1}{2}||\beta||^2 \qquad s.t. \qquad y_i(x_i^T\beta + \beta_0) \geq 1$$

- Allowing overlap:

$$\min ||\beta|| \quad s.t. \quad y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \quad \forall i \quad (12.7)$$

  with

$$\xi_i \geq 0, \quad \sum \xi_i \leq constant$$

- ξ_i called slack variables
- note equivalent to $y_i(x_i^T\beta + \beta_0) \geq C(1 - \xi_i)$
- some points allowed to cross into the margin
- some points allowed to cross to the wrong side of the margin (see Figure 12.2)
- the number of $\xi_i > 1$ is the number of misclassified points
- $\sum \xi_i$ is the total proportional amount by which predictions are on the wrong side
- new optimization problem

$$\min_{\beta_0,\beta} \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{N} \xi_i \quad s.t. \quad \xi_i \geq 0, \quad y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i$$

- $$L_P = \frac{1}{2}||\beta||^2 + \gamma\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\{y_i(x_i^T\beta+\beta_0)-(1-\xi_i)\} - \sum_{i=1}^{N}\mu_i\xi_i$$
- to be minimized over β, β_0, ξ_i: α_i, μ_i are Lagrange multipliers

$$\begin{aligned}\beta &= \sum \alpha_i y_i x_i \\ 0 &= \sum \alpha_i y_i \\ \alpha_i &= \gamma - \mu_i, \quad \forall i\end{aligned}$$

as well as positivity constraints on α_i, μ_i, ξ_i. Substitute into L_P:

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

to be maximized subject to $0 \le \alpha_i \le \gamma$ and $\sum_{i=1}^{N}\alpha_i y_i = 0$. plus conditions (12.14) – (12.16)

- the solution for β has the form

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

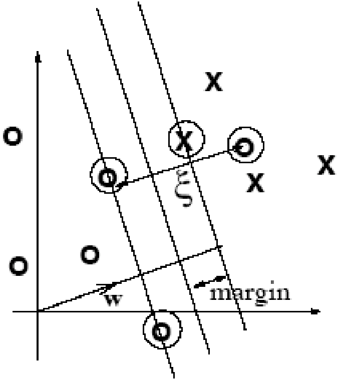
- i.e. linear combinations of x_i ($y_i = \pm 1$)
- only some of the $\hat{\alpha}_i$ are nonzero: those where the lower bound is exact (12.14)
- these observations are called the support vectors
- Figure 12.2
- the "Bayes optimal" classifier is based on the exact posterior probabilities (unknown in general; this example is simulated data)

- use basis function expansions to create more flexible boundaries
- $f(x) = h(x)^T\beta + \beta_0$
- new $L_D = \sum \alpha_i - \frac{1}{2}\sum\sum \alpha_i\alpha_{i'}y_iy_{i'}h(x_i)^T h(x_{i'})$
- solution looks like (12.20)
  $f(x) = \sum \alpha_i y_i < h(x), h(x_i) > +\beta_0$
- i.e. depends only on inner products; alternatively depends on $h(\cdot)$ only through its Kernel function
  $K(x, x') =< h(x), h(x') >$
  - polynomial: $(1+ < x, x' >)^d$
  - radial basis: $\exp(-||x - x'||^2/c)$
  - neural network $tanh(\kappa_1 < x, x' > +\kappa_2)$
- Figure 12.3

- another view of SVM: a penalization problem
- $$\min_{\beta_0,\beta} \sum \{1 - y_i f(x_i)\}_+ + \lambda ||\beta||^2$$
- λ corresponds to $1/(2\gamma)$
- Figure 12.4, Table 12.1
- The “kernelization” of the SVM algorithm means computations are easier; you specify a kernel function instead of (many many *h*’s)
- But we want a sparse solution (many $\alpha_i = 0$) to avoid overfitting
- kernelization can be applied to many algorithms (even ridge regression)
- see Lecture 13 for Sam Roweis’ course on machine learning
- the SVM algorithm induces kernelization *and* sparsity simultaneously

SUPPORT VECTOR MACHINES

- A *support vector machine* (SVM) is nothing more than a kernelized maximum-margin hyperplane classifier.
- You train it by solving the dual quadratic programming problem.
- You run it by finding dot products of the test point with all the training cases.
- Easy!



x
o
x
o
x
ξ
x
o
o
w
margin
o

Support Vector Machines

Description

`svm` is used to train a support vector machine. It can be used to carry out general regression and classification (of nu and epsilon-type), as well as density-estimation. A formula interface is provided.

Usage

```
## S3 method for class 'formula':
svm(formula, data = NULL, ..., subset, na.action =
na.omit, scale = TRUE)
## Default S3 method:
svm(x, y = NULL, scale = TRUE, type = NULL, kernel =
"radial", degree = 3, gamma = 1 / ncol(as.matrix(x)), coef0 = 0, cost = 1, nu = 0.5
class.weights = NULL, cachesize = 40, tolerance = 0.001, epsilon = 0.1,
shrinking = TRUE, cross = 0, probability = FALSE, fitted = TRUE,
..., subset, na.action = na.omit)
```

Arguments

`formula`	a symbolic description of the model to be fit.
`data`	an optional data frame containing the variables in the model. By default the variables are taken from the environment which 'svm' is called from.
`x`	a data matrix, a vector, or a sparse matrix (object of class `matrix.csr` as provided by the package **SparseM**).
`y`	a response vector with one label for each row/component of `x`. Can be either a factor (fo classification tasks) or a numeric vector (for regression).
`scale`	A logical vector indicating the variables to be scaled. If `scale` is of length 1, the value is recycled as many times as needed. Per default, data are scaled internally (both `x` and `y`

Example using R

```
R : Copyright 2004, The R Foundation for Statistical Computing
Version 2.0.1  (2004-11-15), ISBN 3-900051-07-0

...

> library(MASS)
> library(e1071)
Loading required package: class
> ?svm
> data(cats)
> dim(cats)
[1] 144   3
> cats[1:3,]
  Sex Bwt Hwt
1   F   2 7.0
2   F   2 7.4
3   F   2 9.5
> ## this is a simple example taken from the help file for plot.svm
> ## Bwt is body weight in kg; Hwt is heart weight in g
> m <- svm(Sex~., data=cats)
> plot(m,cats)
> quartz()
> plot(cats$Hwt,cats$Bwt,pch=21, bg=c("red","green3")[unclass(cats$Sex)])
> ## we can now look at various components of m, including coefs, SV, index
```

- For more than 2 classes, do $K(K-1)/2$ pairwise comparisons and average (somehow)
- There is a regression version (§12.3.5,6); see Figure 12.6
- Remainder of Chapter 12 is more flexible versions of discriminant analysis; generalizing LDA (I'll skip this)
- New page 385 available from book web site