Linear methods for classification and their extensions

- Logistic regression
- linear and quadratic discriminant analysis
- optimal separating hyperplane
- Support vector machines with kernels, that are nonlinear (later)

Probability of the data given the model

- Assume we have 15 red balls and 5 blue balls in a bag.
- Repeat 5x:
  - select a ball
  - put it back.
- The probability of the sequence red, blue, blue,red, red is $\frac{3}{4} \cdot \frac{1}{4} \cdot \frac{1}{4} \cdot \frac{3}{4} \cdot \frac{3}{4}$.
- The logarithm log_2 of the probability is $\approx -0.4 - 2 - 2 - 0.4 - 0.4 = -5.2$

Likelihood of the model given the data

- Assume we do not know the probabilities, let θ be the probability of *red*. We have following probabilities of data for different θ.

θ	red	blue	blue	red	red	
$\frac{3}{4}$	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{3}{4}$	$\frac{3}{4}$	$\frac{3^3}{4^5}$

- Take the log_2 of the probabilities:

θ	red	blue	blue	red	red	
$\frac{1}{2}$	-1	-1	-1	-1	-1	-5
$\frac{3}{5}$	-0.74	-1.32	-1.32	-0.74	-0.74	-4.86
$\frac{3}{4}$	-0.4	-2	-2	-0.4	-0.4	-5.2

- Probability of the data given model is called **likelihood** of the model θ given the data.
- Maximum likelihood θ estimate is in our case $\frac{3}{5}$.
- Predicting probabilities, maximum likelihood estimate is the same as maximum log-likelihood estimate.

(Log)likelihood

train data		prediction			likelihood	loglik
x_i	g_i	$P(green\|x_i)$	$P(blue\|x_i)$	$P(yellow\|x_i)$		
1	green	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	-1
1	yellow	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	-1
2	green	$\frac{2}{3}$	$\frac{1}{3}$	0	$\frac{2}{3}$	$log_2\frac{2}{3}$
2	green	$\frac{2}{3}$	$\frac{1}{3}$	0	$\frac{2}{3}$	$log_2\frac{2}{3}$
2	blue	$\frac{2}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$	$-log_2 3$
3	blue	0	1	0	1	0
						$-2 - log_2 3$ $+2log_2\frac{2}{3}$

- **loglik** logarithm of likelihood function is defined as:

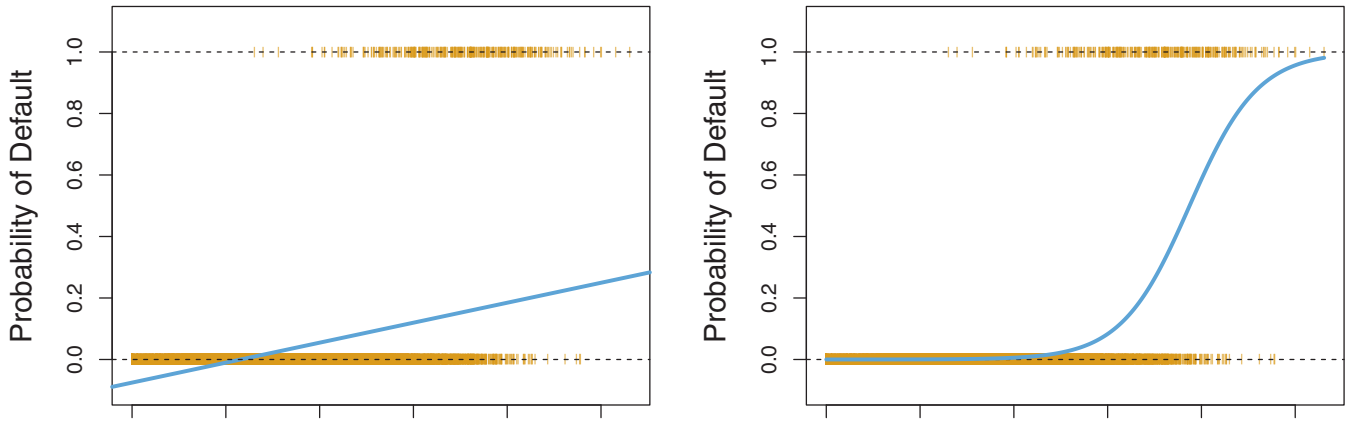
$$l(\theta) = \sum_{i=1}^{N} log(P(G = g_i|x_i, \theta))$$

- Logistic regression uses:

$$P(G = G_k|X = x) = \frac{e^{\beta_{k0}+\beta_k^T x}}{1+\sum_{l=1,\ldots,K-1} e^{\beta_{l0}+\beta_l^T x}}.$$

Logistic Function

- Probability should be from the interval $\langle 0, 1 \rangle$.
- Linear prediction is transformed by logistic function (sigmoid) with the maximum L.
- **logistic** $\frac{L}{1+e^{-k(x-x_0)}}$.
- Inverse function is called logit.
- **logit** $\log \frac{p}{1-p}$,



Logistic Regression

- For $K-$ class classification we estimate $2 \times (K-1)$ parameters $\theta = \{\beta_{10}, \beta_1^T, \ldots, \beta_{(K-1)0}, \beta_{K-1}^T\}$.

$$\begin{aligned}
\log \frac{P(G = G_1|X = x)}{P(G = G_K|X = x)} &= \beta_{10} + \beta_1^T x \\
\log \frac{P(G = G_2|X = x)}{P(G = G_K|X = x)} &= \beta_{20} + \beta_2^T x \\
&\vdots \\
\log \frac{P(G = G_{K-1}|X = x)}{P(G = G_K|X = x)} &= \beta_{(K-1)0} + \beta_{K-1}^T x
\end{aligned}$$

that is

$$p_k(x;\theta) \leftarrow P(G = G_k|X = x) = \frac{e^{\beta_{k0}+\beta_k^T x}}{1 + \sum_{l=1,\ldots,K-1} e^{\beta_{l0}+\beta_l^T x}}$$

$$p_K(x;\theta) \leftarrow P(G = G_K|X = x) = \frac{1}{1 + \sum_{l=1,\ldots,K-1} e^{\beta_{l0}+\beta_l^T x}}.$$

Fitting Logistic Regression Two class

- This model is estimated iteratively maximizing conditional likelihood of G given X.

$$\ell(\theta) = \sum_{i=1}^{N} \log p_{g_i}(x_i; \theta)$$

- Two class model: g_i encoded via a $0/1$ response y_i; $y_i = 1$ iff $g_k = g_1$. Let $p(x;\theta) = p_1(x;\theta)$, $p_2(x;\theta) = 1 - p(x;\theta)$. Then:

$$\begin{aligned} \ell(\theta) &= \sum_{i=1}^{N} (y_i \log p(x_i;\beta) + (1-y_i)\log(1-p(x_i;\beta)) \\ &= \sum_{i=1}^{N} (y_i \beta^T x_i - \log(1+e^{\beta^T x_i})) \end{aligned}$$

- Set derivatives to zero:

$$\frac{\partial \ell(\beta)}{\partial \beta} = \sum_{i=1}^{N} x_i (y_i - p(x_i;\beta)) = 0,$$

- which is $p+1$ nonlinear equations in β.
- First component: $x_i \equiv 1$ specifies $\sum_{i=1}^{N} y_i = \sum_{i=1}^{N} p(x_i;\beta)$ the expected

Newton–Raphson Algorithm

- We use Newton–Raphson Algorithm to solve the system of equations

$$\frac{\partial \ell(\beta)}{\partial \beta} = \sum_{i=1}^{N} x_i (y_i - p(x_i; \beta)) = 0,$$

- we need the second–derivative or Hessian matrix

$$\frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} x_i x_i^T p(x_i; \beta)(1 - p(x_i; \beta)).$$

- Starting with β^{old} a single Newton–Raphson update is

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^T} \right)^{-1} \frac{\partial \ell(\beta)}{\partial \beta},$$

- where the derivatives are evaluated at β^{old}.

Newton–Raphson Algorithm in Matrix Notation

Let us denote:

$\mathbf{y}$ the vector of y_i

$\mathbf{X}$ $N \times (p+1)$ data matrix x_i

$\mathbf{p}$ the vector of fitted probabilities with ith element $p(x_i; \beta^{old})$

$\mathbf{W}$ diagonal matrix with weights $p(x_i; \beta^{old})(1 - p(x_i; \beta^{old}))$

$$\frac{\partial \ell(\beta)}{\partial \beta} = \mathbf{X}^T(\mathbf{y} - \mathbf{p})$$

$$\frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^T} = -\mathbf{X}^T\mathbf{W}\mathbf{X}$$

The Newton–Raphson step is ($\beta^0 \leftarrow 0$)

$$\begin{aligned}
\beta^{new} &= \beta^{old} + (\mathbf{X}^\mathbf{T}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y} - \mathbf{p}) \\
&= (\mathbf{X}^\mathbf{T}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}(\mathbf{X}\beta^{old} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p})) \\
&= (\mathbf{X}^\mathbf{T}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z} \\
\mathbf{z} &= \mathbf{X}\beta^{old} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p}) \quad \textbf{adjusted response}
\end{aligned}$$

- p, W, z change each step

This algorithm is reffered to as **iteratively reweighted least squares IRLS**

$$\beta^{new} \leftarrow \arg\min_{\beta} (\mathbf{z} - \mathbf{X}\beta)^T\mathbf{W}(\mathbf{z} - \mathbf{X}\beta)$$

South African Heart Disease

- Analyzing the risk factors of myocardian infarction MI
- prevalence 5.1%, in the data 160 positive 302 controls

TABLE 4.2. *Results from a logistic regression fit to the South African heart disease data.*

	Coefficient	Std. Error	Z Score
(Intercept)	−4.130	0.964	−4.285
sbp	0.006	0.006	1.023
tobacco	0.080	0.026	3.034
ldl	0.185	0.057	3.219
famhist	0.939	0.225	4.178
obesity	-0.035	0.029	−1.187
alcohol	0.001	0.004	0.136
age	0.043	0.010	4.184

- Wald test: Z score $|Z| > 2$ is significant at at the 5% level.

TABLE 4.3. *Results from stepwise logistic regression fit to South African heart disease data.*

	Coefficient	Std. Error	Z score
(Intercept)	−4.204	0.498	−8.45
tobacco	0.081	0.026	3.16
ldl	0.168	0.054	3.09
famhist	0.924	0.223	4.14
age	0.044	0.010	4.52

South African Heart Disease

- Wald test: Z score $|Z| > 2$ is significant at at the 5% level.

TABLE 4.3. *Results from stepwise logistic regression fit to South African heart disease data.*

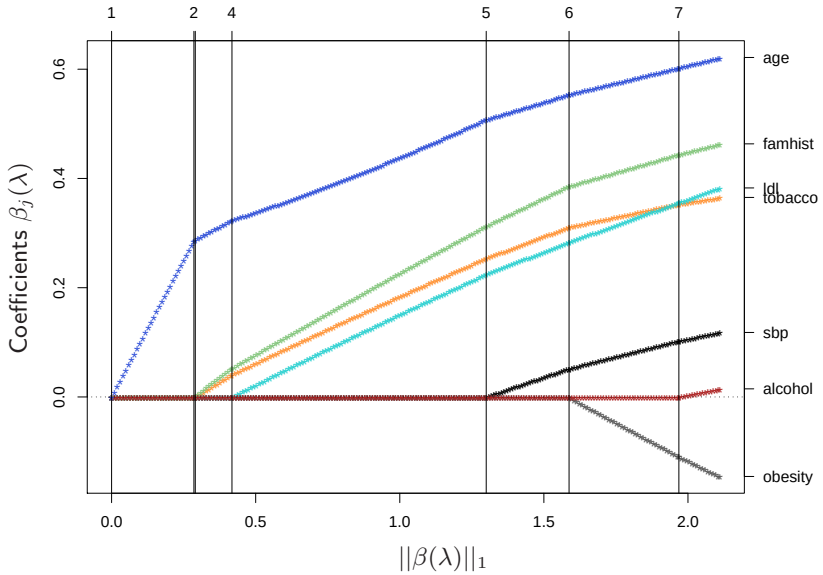
	Coefficient	Std. Error	Z score
`(Intercept)`	−4.204	0.498	−8.45
`tobacco`	0.081	0.026	3.16
`ldl`	0.168	0.054	3.09
`famhist`	0.924	0.223	4.14
`age`	0.044	0.010	4.52

- $P(MI|x_i, \theta) = \frac{e^{-4.204+0.081x_{tobacco}+0.168x_{ldl}+0.924x_{famhist}+0.044x_{age}}}{1+(e^{-4.204+0.081x_{tobacco}+0.168x_{ldl}+0.924x_{famhist}+0.044x_{age}})}$
- Interval estimate $odds_{tobacco} = e^{0.081 \pm 2 \times 0.026} = (1.03, 1.14)$ increase of odds of MI based of the increase of $x_{tobacco}$.

L_1 regularization 'Lasso'-like

$$argmax_{\beta_0,\beta}\left(\sum_{i=1}^{N}\left(y_i(\beta_0+\beta^T x_i)-\log(1+e^{(\beta_0+\beta^T x_i)})\right)-\lambda\sum_{j=1}^{p}|\beta_j|\right)$$

- Newton–Raphson Algorithm or nonlinear programming.
- $\lambda = 0$ standard logistic regression.
- $\lambda \to \infty$ moves coefficients towards 0.
- β_0 is not included into the penalty.

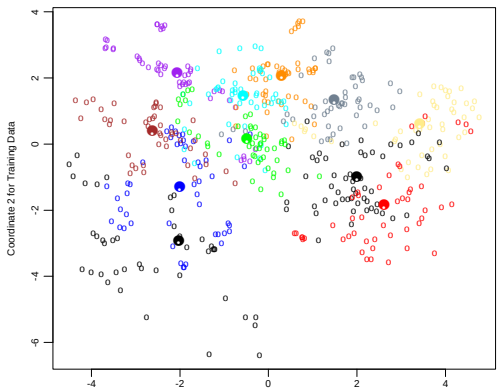


Linear Discriminant Analysis

- LDA gives similar results as LR for two classes. It avoids masking in k–dimensional classification.
- LDA assumes multivariate gaussian distribution of each class with a common covariance matrix.

$$\phi(k) = \frac{1}{\sqrt{|2\pi\Sigma|}} e^{-\frac{1}{2}(x-\mu_k)\Sigma^{-1}(x-\mu_k)}$$

  - Under this assumptions it provides bayes optimal estimate.
- Different covariance matrix for each class leads to Quadratic Discriminant Analysis.
- Let us denote N_k number of training data in the class G_k.



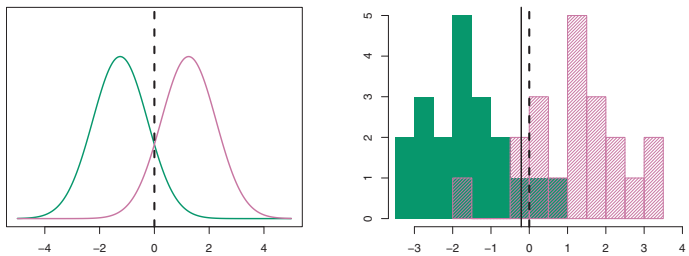


FIGURE 4.4. Left: *Two one-dimensional normal density functions are shown.*

Linear Discriminant Analysis

$$
\begin{aligned}
P(G = G_k | X = x) &= \frac{\phi_k(x)\pi_k}{\sum_{\ell=1}^{K} \phi_\ell(x)\pi_\ell} \\
\hat{\pi}_k &= \frac{N_k}{N} \\
\phi_k(x) &= N(\mu_k, \Sigma) \\
\hat{\mu}_k &= \frac{\sum_{\{x_i : G(x_i) = G_k\}} x_i}{N_k} \\
\hat{\Sigma} &= \sum_{k=1}^{K} \sum_{\{x_i : G(x_i) = G_k\}} \frac{(x_i - \mu_k)^T (x_i - \mu_k)}{(N - K)}
\end{aligned}
$$

To classify new instance x we predict the G_k with maximal δ_k:

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + log \pi_k$$

Linear Discriminant Analysis

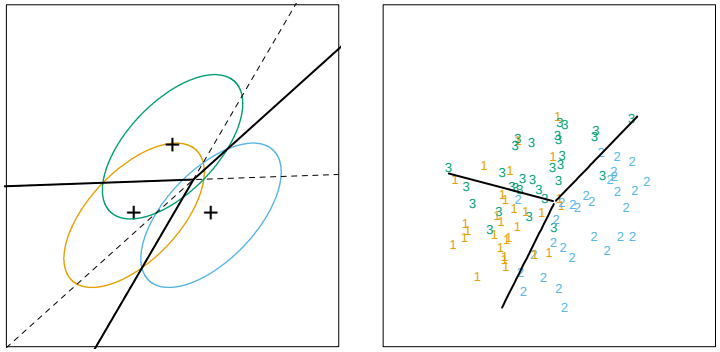


FIGURE 4.5. *The left panel shows three Gaussian distributions, with the same covariance and different means. Included are the contours of constant density*

Example Vovel data ESL $X \in \mathbb{R}^{10}$:

	train	test
Linear regression	0.48	0.67
Linear discriminant analysis	0.32	0.56
Quadratic discriminant analysis	0.01	0.53
Logistic regression	0.22	0.51

Quadratic Discriminant Analysis

$$
\begin{aligned}
P(G = G_k | X = x) &= \frac{f_k(x)\pi_k}{\sum_{\ell=1}^{K} f_\ell(x)\pi_\ell} \\
\hat{\pi}_k &= \frac{N_k}{N} \\
f_k(x) &= N(\mu_k, \Sigma_k) \\
\hat{\mu}_k &= \frac{\sum_{\{x_i : G(x_i) = G_k\}} x_i}{N_k} \\
\hat{\Sigma}_k &= \sum_{\{x_i : G(x_i) = G_k\}} \frac{(x_i - \mu_k)^T (x_i - \mu_k)}{(|G_k| - 1)}
\end{aligned}
$$

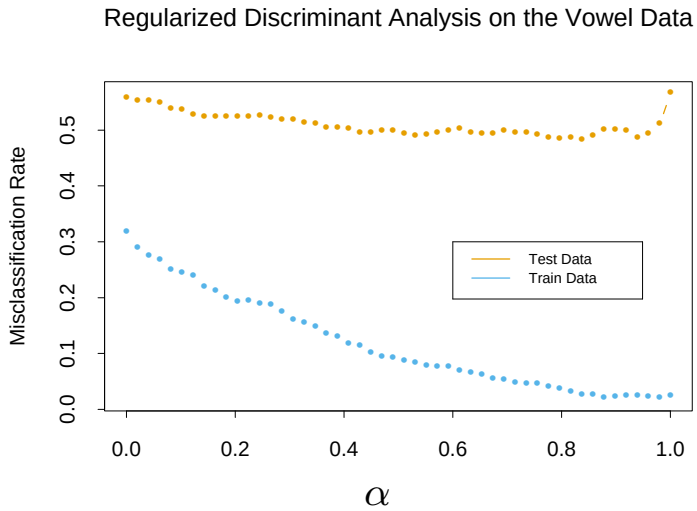
To classify new instance x we predict the G_k with maximal δ_k:

$$
\delta_k(x) = -\frac{1}{2} \log |\Sigma_k| - \frac{1}{2}(x^T - \mu_k)^T \Sigma_k^{-1} (x^T - \mu_k) + \log \pi_k
$$

Quadratic and Regularized Discriminant Analysis

Regularized:

$$\hat{\Sigma}_k(\alpha) = \alpha\hat{\Sigma}_k + (1-\alpha)\hat{\Sigma}$$



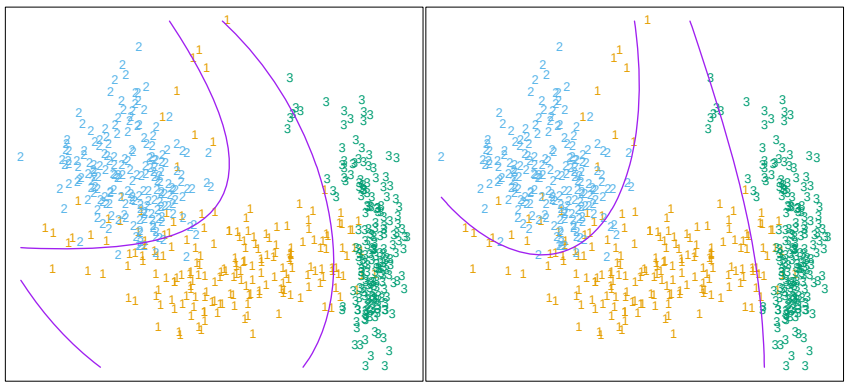


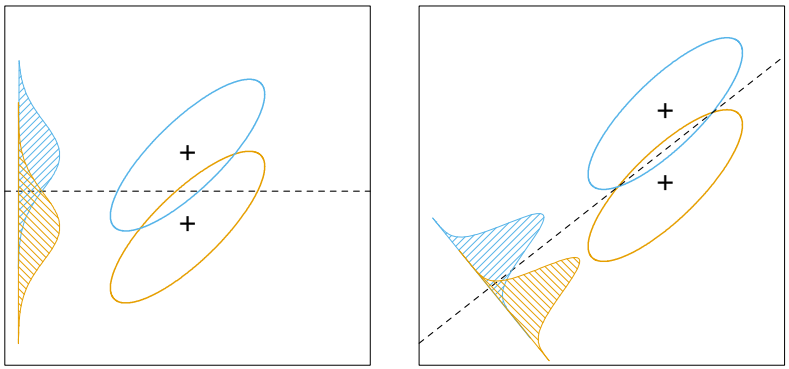
FIGURE 4.6. *Two methods for fitting quadratic*

- LDA parameters: $(K-1)\times(p+1)$
- QDA parameters: $(K-1)\times(\frac{p(p+3)}{2}+1)$

Computations for LDA

- $O(N^3)$, often $O(N^{2.376})$
- QDA and LDA may be computed using matrix decomposition:
  - Compute the eigendecomposition for each $(x - \hat{\mu}_k)^T \hat{\Sigma}_k (x - \hat{\mu}_k) = [\mathbf{U}_k^T (x - \hat{\mu}_k)]^T \mathbf{D}_k^{-1} [\mathbf{U}_k^T (x - \hat{\mu}_k)]$
  - $\log |\hat{\Sigma}_k| = \sum_\ell \log d_{k\ell}$.
- Using this decomposition, LDA classifier can be implemented by the following pair of steps:
  - *Sphere* the data with respect to the common covariance estimate $\hat{\Sigma}$: $X^* \leftarrow \mathbf{D}^{-\frac{1}{2}} \mathbf{U}^T X$, where $\hat{\Sigma} = \mathbf{U}\mathbf{D}\mathbf{U}^T$.
    The common covariance estimate of X^* will now be the identity.
  - Classify to the closest class centroid in the transformed space, modulo the effect of the class prior probabilities π_k.

Reduced–Rank Linear Discriminant Analysis

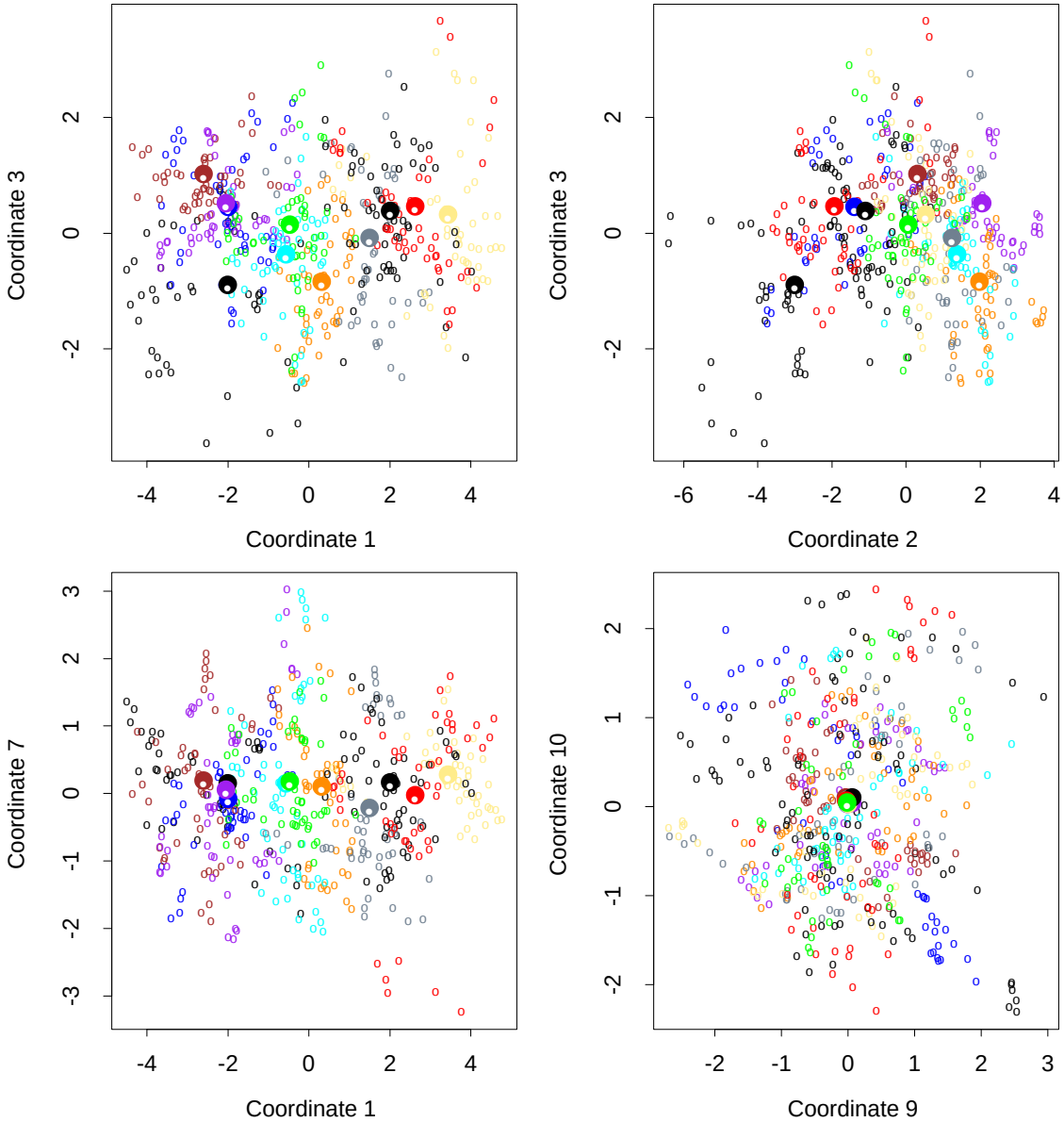


$$max_a \frac{a^T B a}{a^T W a}$$

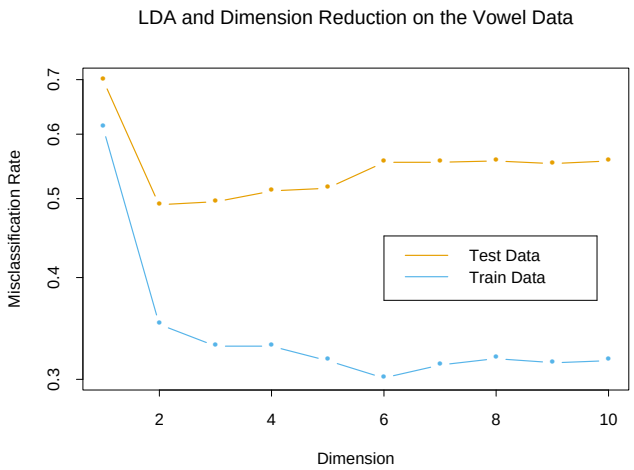
Finding the sequence of optimal subspaces for LDA:

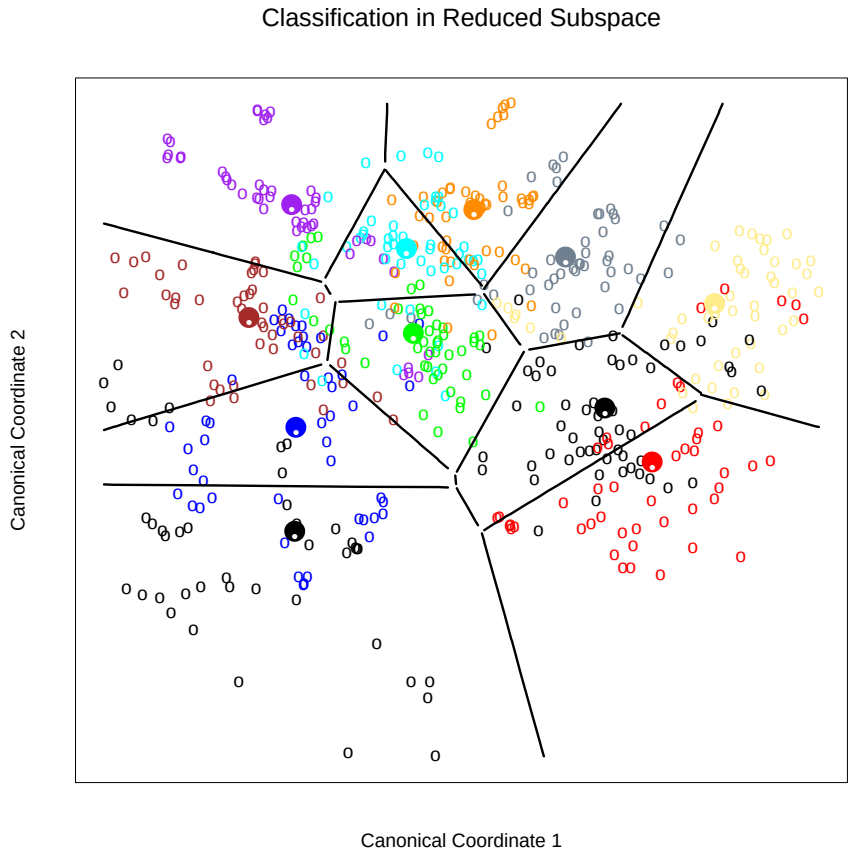
- compute the $K \times p$ matrix of class centroids M and the common covariance matrix W (**within–class** covariance);
- compute $M^* = MW^{-\frac{1}{2}}$ using the eigen–decomposition of W;
- compute B^* **between–class** covariance, the covariance matrix of M^* and its eigen–decomposition $B^* = V^* D_B V^{*T}$.
  - order D_B in the decreasing order
  - v_ℓ^* of V^* in sequence define the coordinates of the optimal subspaces
  - $Z_\ell = v_\ell^T X$ with $v_\ell = W^{-\frac{1}{2}} v_\ell^*$.

Reduced–Rank Linear Discriminant Analysis



Vovel Example



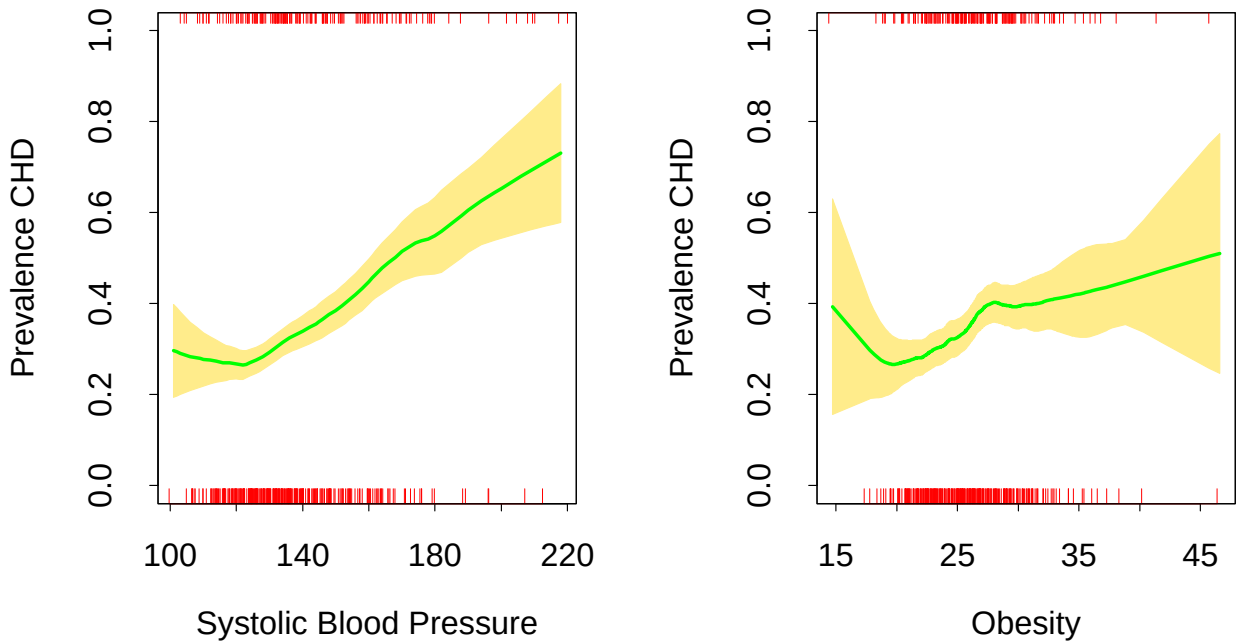


Local Likelihood and other methods

- logistic and log-linear models involve the covariates in a linear fashion

$$l(\beta(x_0)) = \sum_{i=1}^{N} K_\lambda(x_0, x_i) l(y_i, x^T \beta(x_0))$$

$$\sum_{i=1}^{N} K_\lambda(x_0, x_i) \left\{ y_i \beta^T (x_i - x_0) - \log(1 + e^{\beta^T (x_i - x_0)}) \right\}$$



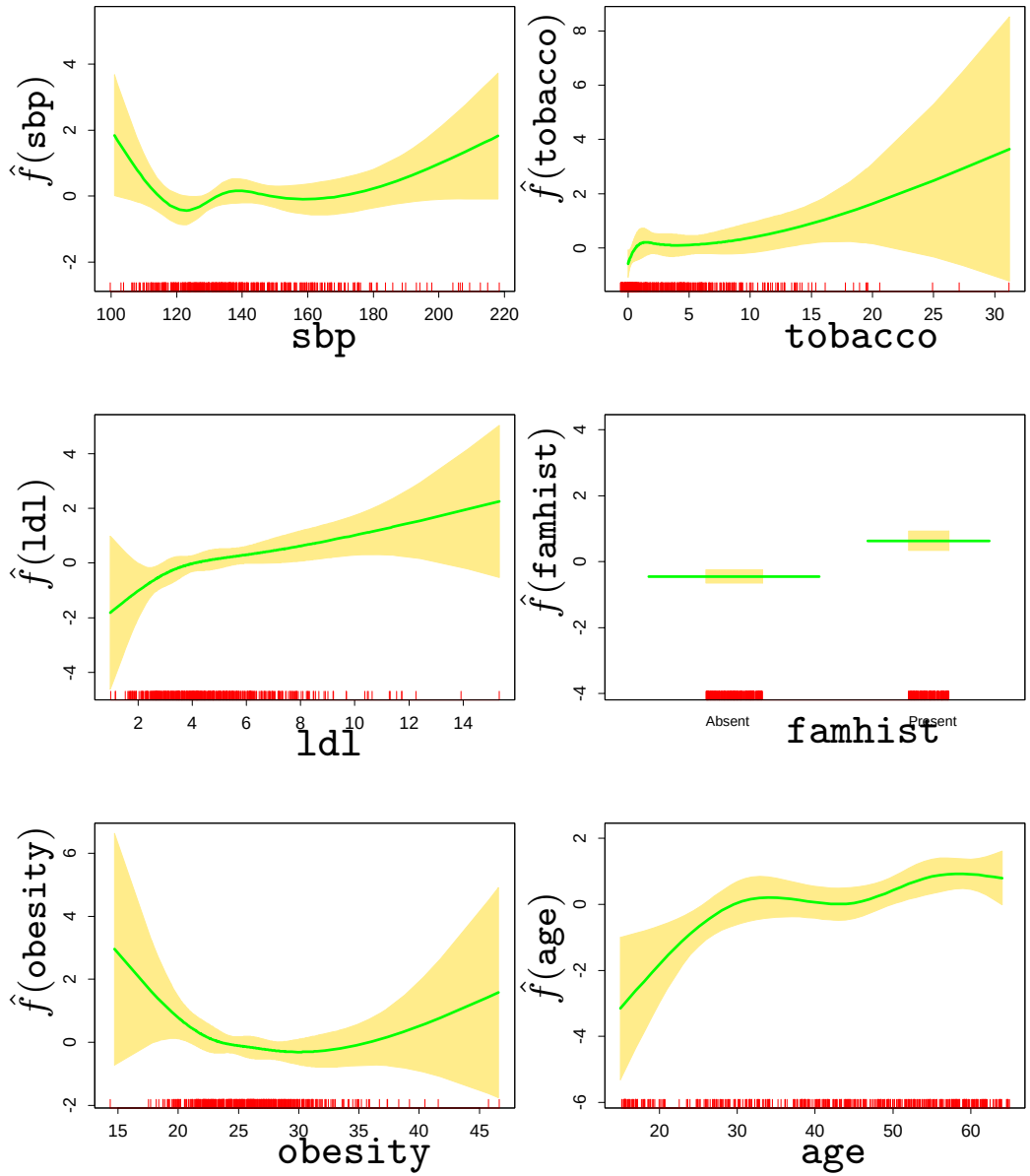
South African Heart Disease continued

- Each feature X_j is approximated by a natural spline.
- The overall model is:

$$logit[P(ch|X)] = \theta_0 + h_1(X_1)^T\theta_1 + h_2(X_2)^T\theta_2 + \ldots + h_p(X_p)^T\theta_p$$

- θ_j are vectors of coefficients multiplying their associated vector of natural spline basis functions h_j
- four basis functions (three inner knots) per spline in this example.
- binary *familyhist* with a single coefficient.
- Combine all p vectors of basic functions into one big vector $h(X)$, $df = 1 + \sum_{j=1}^{p} df_j$
- each basis function is evaluated at each of the N samples
- resulting in a $N \times df$ basis matrix $\mathbf{H}$.
- and use 'standard' logistic regression.

- Alcohol not significant by AIC test
- covariance $Cov(\hat{\theta})$ is estimated by $\hat{\Sigma} = (\mathbf{H}^{\mathsf{T}}\mathbf{W}\mathbf{H})^{-1}$
  - W the diagonal weight matrix
- variance of a single variable j is:
  - $v_j(X_j) = Var[f_j(X_j)] = h_j(X_j)^T \hat{\Sigma}_{jj} h_j(X_j)$
- error bounds $\hat{f}_j(X_j) \pm 2\sqrt{v_j(X_j)}$.



Separating hyperplane, Optimal separating hyperplane

- Classification, we encode the goal class by -1 and 1, respectively.
- separate the space x by a hyperplane
- not necessary optimal for LDA;
- logistic regression finds one if it exists
- **Perceptron** finds separating hyperplane if it exists
  exact position depends on initial parameters

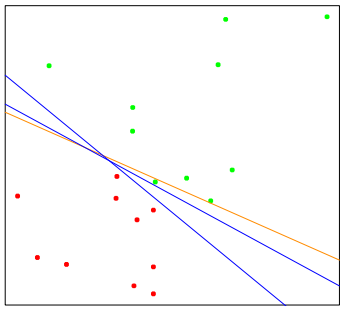


FIGURE 4.14. *A toy example with two classes separable by a hyperplane. The orange line is the least squares solution, which misclassifies one of the training points. Also shown are two blue separating hyperplanes found by the* perceptron learning algorithm *with different random starts.*

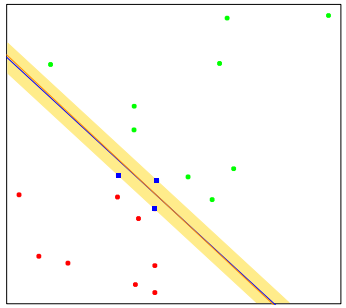


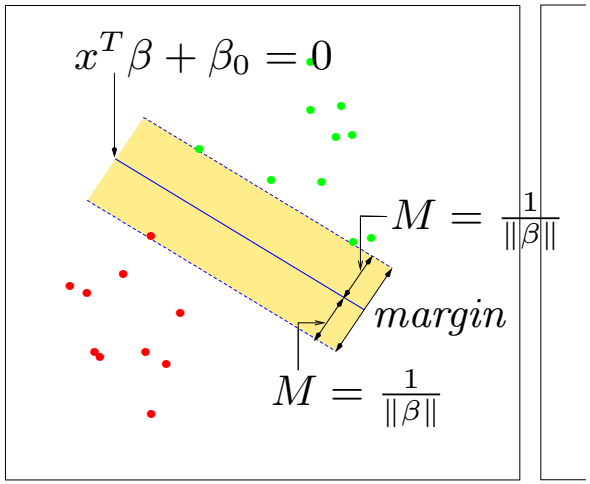
FIGURE 4.16. *The same data as in Figure 4.14. The shaded region delineates the maximum margin separating the two classes. There are three support points indicated, which lie on the boundary of the margin, and the optimal separating hyperplane (blue line) bisects the slab. Included in the figure is the boundary found using logistic regression (red line), which is very close to the optimal separating hyperplane (see Section 12.3.3).*

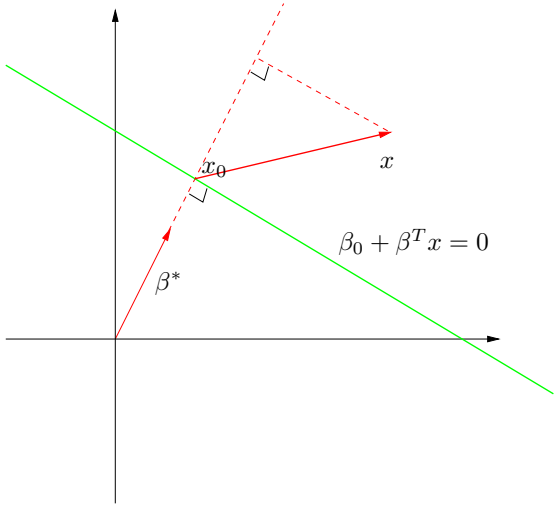
Optimal Separating Hyperplane (separble case)

We define **Optimal Separating Hyperplane** as a separating hyperplane with maximal free space C without any data point around the hyperplane.
Formally:

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

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





Formally:

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

subject to $y_i(x_i^T\beta + \beta_0) \geq C$ for all $i = 1, \ldots, N$.
We re-define: $\|\beta\| = 1$ can be moved to the condition (and redefine β_0):

$$\frac{1}{\|\beta\|} y_i(x_i^T\beta + \beta_0) \geq C$$

Since for any β and β_0 satisfying these inequalities, any positively scaled multiple satisfies them too, we can set $\|\beta\| = \frac{1}{C}$ and we get:

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$$

subject to $y_i(x_i^T\beta + \beta_0) \geq 1$ pro $i = 1, \ldots, N$.
This is a convex optimization problem. The Lagrange function, to be minimized w.r.t. β and β_0, is:

$$L_P = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i[y_i(x_i^T\beta + \beta_0) - 1].$$

$$L_P = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(x_i^T \beta + \beta_0) - 1].$$

Setting the derivatives to zero, we obtain:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i$$

Substituing these in L_P we obtain the so–called Wolfe dual:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

subject to $\alpha_i \geq 0$
The solution is obtained by maximizing L_D in the positive orthant, for which standard software can be used.

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

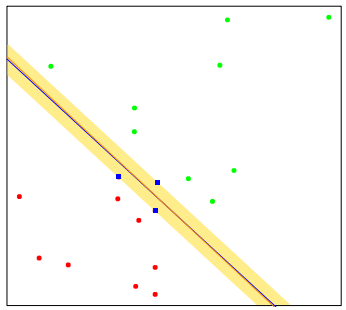
subject to $\alpha_i \geq 0$
In addition the solution must satisfy the Karush–Kuhn–Tucker conditions:

$$\alpha_i [y_i (x_i^T \beta + \beta_0) - 1] = 0$$

for any i, therefore for any $\alpha_i > 0$ must $[y_i (x_i^T \beta + \beta_0) - 1] = 0$, that means x_i is on the boundary of the slab and for all x_i outside the boundary is $\alpha_i = 0$.
The boundary is defined by x_i with $\alpha_i > 0$ – so called **support vectors**.

We classify new observations
$\hat{G}(x) = sign(x^T \beta + \beta_0)$

- where $\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$,
- $\beta_0 = y_s - x_s^T \beta$ for any support vector $\alpha_s > 0$.



Optimal Separating Hyperplane (nonseparble case)

- We have to accept incorrectly classified instances in a non–separable case.
- We limit the number of incorrectly classified examples.

We define **slack** ξ for each data point $(\xi_1, \ldots, \xi_N) = \xi$ as follows:

- ξ_i is the distance of x_i from the boundary for x_i at the wrong side of the margin
- and $\xi_i = 0$, for x_i at the correct side.

We require $\sum_{i=1}^{N} \xi_i \leq K$.

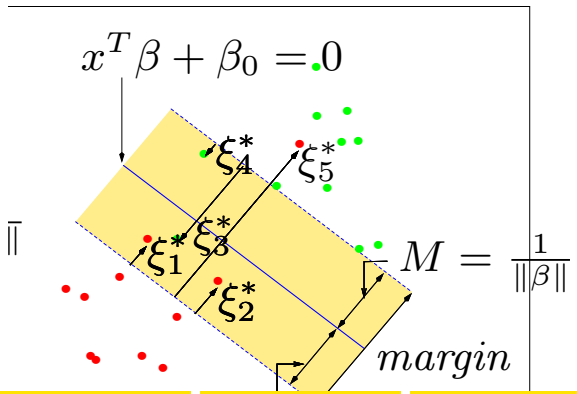
We solve the optimization problem

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

subject to:

$$y_i(x^T\beta + \beta_0) \geq C(1 - \xi_i)$$

where $\forall i$ is $\xi_i \geq 0$ a $\sum_{i=1}^{N} \xi_i \leq K$.



Optimal Separating Hyperplane (nonseparble case)

Again, we omit replace the condition $\|\beta\|$ by defining $C = \frac{1}{\|\beta\|}$ and optimize

$$\min \|\beta\| \text{subject to} \{ \begin{array}{c} y_i(x^T\beta + \beta_0) \geq (1 - \xi_i) \forall i \\ \xi_i \geq 0, \sum \xi_i \leq \textit{constant} \end{array} \tag{13}$$

We solve

$$\min_{\beta, \beta_0} \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \xi_i$$

subject to $\xi_i \geq 0$ and $y_i(x^T\beta + \beta_0) \geq (1 - \xi_i)$ where γ has replaced the constant$K$, we cans set $\gamma = \infty$ for the separable case.

We solve

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \xi_i$$

subject to $\xi_i \geq 0$ and $y_i(x^T\beta + \beta_0) \geq (1 - \xi_i)$ where γ has replaced the constant$K$, we cans set $\gamma = \infty$ for the separable case.
Lagrange multiplyers again for α_i, μ_i:

$$L_P = \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i[y_i(x_i^T\beta + \beta_0) - (1 - \xi_i)] - \sum_{i=1}^{N} \mu_i\xi_i$$

Setting the derivative $= 0$ we get:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i$$

$$\alpha_i = \gamma - \mu_i$$

Substitute to get Wolfe dual:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

and maximize L_D subject to $0 \leq \alpha_i \leq \gamma$ a $\sum_{i=1}^{N} \alpha_i y_i = 0$.
Solution satisfyes:

$$\begin{aligned} \alpha_i [y_i (x_i^T \beta + \beta_0) - (1 - \xi_i)] &= 0 \\ \mu_i \xi_i &= 0 \\ \left[y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) \right] &\geq 0 \end{aligned}$$

Solution: $\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i$
with nonzero coefficients $\widehat{\alpha_i}$ – these are boundary points where $\widehat{\xi_i} = 0$ (therefore $0 < \widehat{\alpha_i} < \gamma$), and points on the wrong side of the margin $\widehat{\xi_i} > 0$ (and $\widehat{\alpha_i} = \gamma$. Any point with $\widehat{\xi_i} = 0$ can be used to calculate $\widehat{\beta_0}$, typically an average is taken.
$\widehat{\beta_0}$ for a boundary point $\xi_i = 0$:

$$\alpha_i \left[y_i (x^T \widehat{\beta} + \widehat{\beta_0}) - (1 - 0) \right] = 0$$

Parameter γ settled by tuning (crossvalidation).

Support Vector Machines

- Genaralization

Let us have the training data $(x_i, y_i)_{i=1}^N$, $x_i \in \mathbb{R}^p$, y_i in $\{-1, 1\}$. We define a hyperplane

$$\{x : f(x) = x^T \beta + \beta_0 = 0\} \tag{14}$$

where $\|\beta\| = 1$.
We classify according to

$$G(x) = sign\left[x^T \beta + \beta_0\right]$$

where $f(x)$ is a signed distance of x from the hyperplane.

Kernels

- We searched for a linear boundary.
- We enhance the feature space to nonlinear one.
- Imagine M functions $h_m(x)$, $m = 1, \ldots, M$.
- $h(x_i) = (h_1(x), \ldots, h_M(x))$ and a decision boundary $\widehat{f}(x) = h(t)^T \widehat{\beta} + \widehat{\beta_0}$.

The classification function $\hat{f}(x) = h(x)^T \beta + \beta_0 = \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i) \rangle + \beta_0$ does not need evalutation of $h(i)$, only the scalar product $\langle h(x), h(x_i) \rangle$.

- **kernal functions** are function to replace scalar product with a scalar product in a transformed space.

dth Degree polynomial:	$K(x, x^{\mid}) = (1 + \langle x, x^{\mid} \rangle)^d$
Radial basis	$K(x, x^{\mid}) = exp(\frac{-\|x - x^{\mid}\|^2}{c})$
Neural network	$K(x, x^{\mid}) = tanh(\kappa_1 \langle x, x^{\mid} \rangle + \kappa_2)$

Example: Degree 2 with two dimensional input:
$K(x, x') = (1 + \langle x, x' \rangle)^2 = (1 + 2x_1 x_1' + 2x_2 x_2' + (x_1 x_1')^2 + (x_2 x_2')^2 + 2x_1 x_1' x_2 x_2')$
that is. $M = 6$, $h_1(x) = 1$, $h_2(x) = \sqrt{2} x_1$, $h_3(x) = \sqrt{2} x_2$,
$h_4(x) = x_1^2$, $h_5(x) = x_2^2$, $h_6(x) = \sqrt{2} x_1 x_2$.

Učení struktury, diskrétní proměnné

Definition (KL–divergence)

KL–divergence dvou pravděpodobnostních rozložení P, Q na stejné doméně $sp(P) = sp(Q)$ je definovaná jako: $D_{KL}(P||Q) = \sum_{i \in sp(P)} P(i) log \frac{P(i)}{Q(i)}$.

Definition (Entropie)

Entropie pravděpodobnostního rozložení P na doméně $sp(P)$ je definovaná jako: $H(P) = -\sum_{i \in sp(P)} P(i) log P(i)$.

Definition (Vzájemná informace (Mutual Information))

Mutual Information dvou veličin X, Y na doménách $sp(X), sp(Y)$ je definovaná jako: $I(X;Y) = \sum_{i \in sp(X)} \sum_{j \in sp(Y)} P(X = i, Y = j) log \frac{P(X=i, Y=j)}{P(X=i)P(Y=j)}$.

Chow–Liu Tree

Lemma

Pro stromy s interakcemi maximálně druhého řádu platí pro D_{KL} aproximace P' a vzoru P

$$D_{KL}(P||P') = -\sum I(X_i, X_{j(i)}) + \sum H(X_i) - H(X_1, \ldots, X_n)$$

kde $X_{j(i)}$ je rodič vrcholu X_i.

1: **procedure** CHOW–LIU:(dataset)
2: Calculate $I(A, B)$ for each pair of nodes
3: Find maximal spanning tree (kostru)
4: Orient edges (no head–to–head connection)
5: Learn parameters
6: # Remove orientation.
7: **end procedure**

Neorientované modely s diskrétními proměnnými

- Boltzmann machine (=Ising models; special case of Markov random field)
  - visible and hidden nodes
  - only pairwise interactions
  - binary valued nodes
  - constant node $X_0 \equiv 1$.

$$p(X, \Theta) = exp\left[\sum_{(j,k)\in E} \theta_{jk} X_j X_k - \Phi(\Theta)\right]$$

$$\Phi(\Theta) = log \sum_{x\in\mathcal{X}} \left[exp(\sum_{(j,k)\in E} \theta_{jk} X_j X_k)\right]$$

- Issing model implies a logistic form for each node conditional on the others

$$P(X_j = 1 | X_{-j} = x_{-j}) = \frac{1}{1 + exp(-\theta_{j0} - \sum_{(j,k)\in E} \theta_{jk} x_k)}$$

- Restricted Boltzmann machines
  - dvě vrstvy, viditelná a skrytá, v rámci vrstvy žádné hrany - snáze se učí.

Chow–Liu Tree Undirected Variant

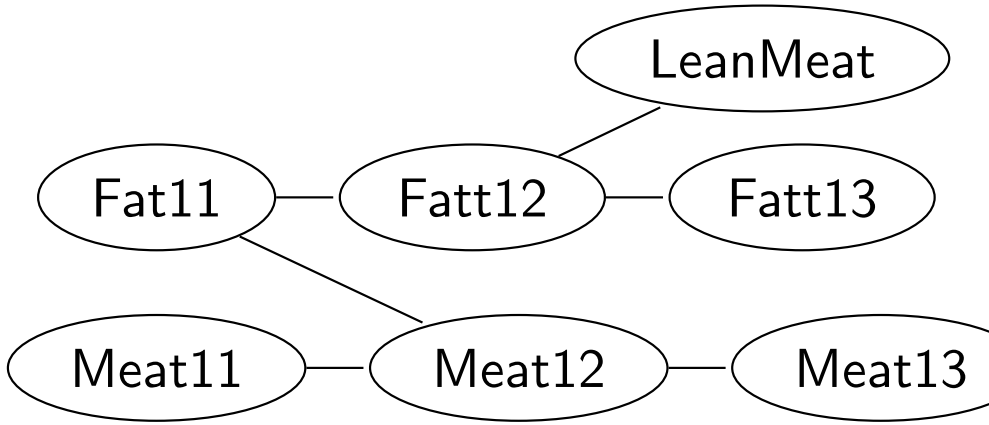
Lemma

For models with interactions of degree maximally 2 (trees) it holds D_{KL} the approximation P' of P

$$D_{KL}(P||P') = -\sum_{(i,j)\in E} I(X_i, X_j) + \sum_{i=1\ldots p} H(X_i) - H(X_1, \ldots, X_p)$$

Algorithm: Chow–Liu (variant)

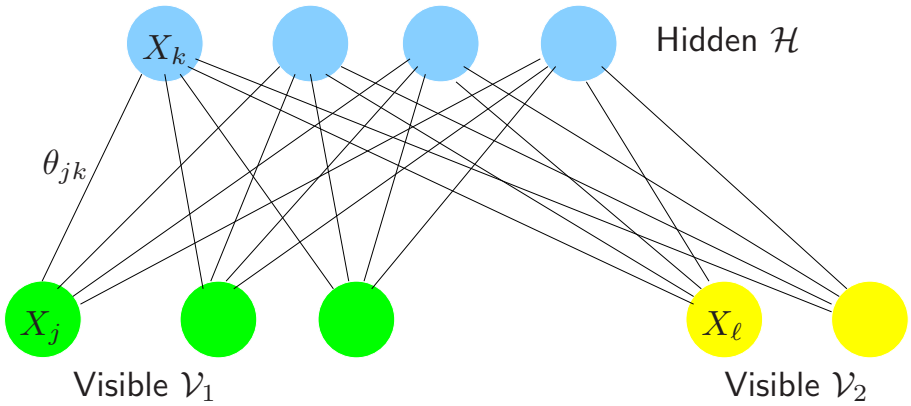
- Calculate $I(X_i, X_j)$ for each pair of nodes
- Find maximal spanning tree (kostru)
- Learn parameters.

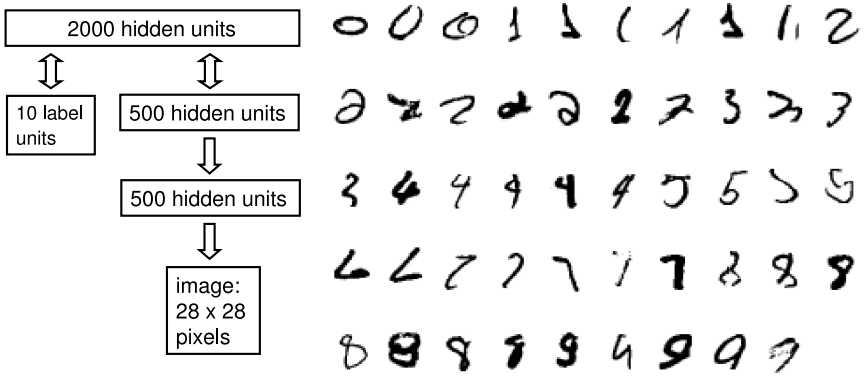


Učení Bolznamnn machine

- Učení parametrů
  - iterativně
  - např. Iterative proportional fitting IPF Jiroušek and Přeučil.
- Učení struktury
  - např. Hoefling and Tibshirany: glasso extension to discrete Markov Networks.
  - pořád dost pomalé a trochu nepřesné.
- Restricted Boltzmann machine
  - učení daleko rychlejší díky podmíněné nezávislosti

Restricted Boltzmann Machine





Markov Properties (Zeros are dangerous)

Definition (Markov properties: Global, Local, Pairwise)

Let G be an undirected graph over V.

(GM) A probability measure P over V is **(globally) Markov** with respect to G iff

$$\forall(\mathcal{A}, \mathcal{B} \in V, \mathcal{C} \subseteq V)\ \mathcal{A} \perp\!\!\!\perp_G \mathcal{B}|\mathcal{C} \Rightarrow \mathcal{A} \perp\!\!\!\perp \mathcal{B}|\mathcal{C} \text{ v P.}$$

(LM) A probability measure has **local Markov property** iff
$(\forall A \in V):\ A \perp\!\!\!\perp V \setminus Fa_A | N_A\ [P]$

(PM) A probability measure has **pairwise Markov property** iff $\forall A, B \in V, A \neq B$ not connected by an edge holds $A \perp\!\!\!\perp B | V \setminus \{A, B\}\ [P]$

Theorem

These properties are equivalent for stricty positive measures.

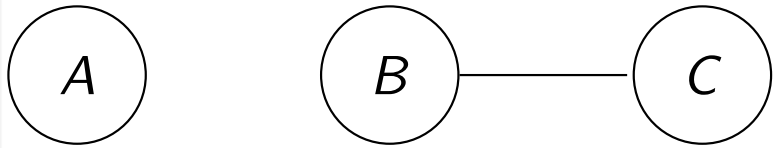
Counterexamples for measures with zero probability everywhere except $(0,0,0)$ and $(1,1,1)$.
See [Milan Studený:*Struktury podmíněné nezávislosti, Matfyzpress 2014*]..

Examples

Example

$V = \{A, B, C\}, E = \{(b, c)\}$. Let us have a binary probability measure V nonzero at points $(0, 0, 0)$ and $(1, 1, 1)$ [Studený p.101].

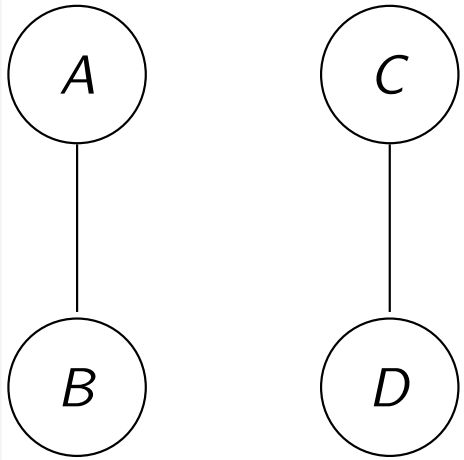
$A \perp\!\!\!\perp B|\{C\}$
$A \perp\!\!\!\perp C|\{B\}$ & does not imply $A \perp\!\!\!\perp BC|\{\}$.



Example

$V = \{A, B, C\}, E = \{(b, c)\}$. Let us have a binary probability measure V nonzero at points $(0, 0, 0, 0)$ and $(1, 1, 1, 1)$ [Studený p.101].

$A \perp\!\!\!\perp CD|\{B\}$
$B \perp\!\!\!\perp CD|\{A\}$
$C \perp\!\!\!\perp AB|\{D\}$ & does not imply $A \perp\!\!\!\perp C|\{\}$.
$D \perp\!\!\!\perp AB|\{C\}$



Separating hyperplane, Optimal separating hyperplane

- Classification, we encode the goal class by -1 and 1, respectively.
- separate the space x by a hyperplane
- not necessary optimal for LDA;
- logistic regression finds one if it exists
- **Perceptron** finds separating hyperplane if it exists
  exact position depends on initial parameters

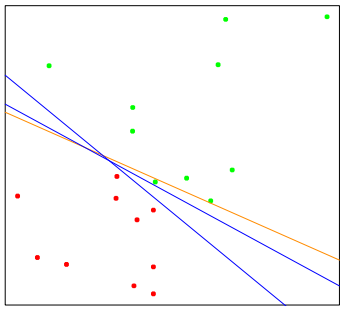


FIGURE 4.14. *A toy example with two classes separable by a hyperplane. The orange line is the least squares solution, which misclassifies one of the training points. Also shown are two blue separating hyperplanes found by the* perceptron learning algorithm *with different random starts.*

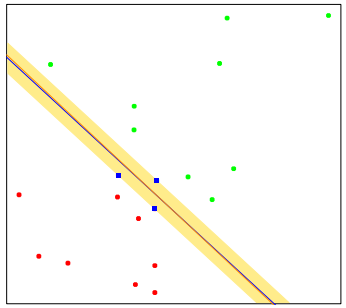


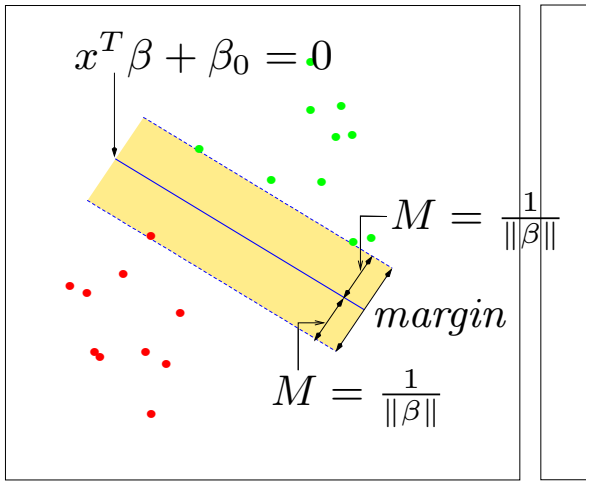
FIGURE 4.16. *The same data as in Figure 4.14. The shaded region delineates the maximum margin separating the two classes. There are three support points indicated, which lie on the boundary of the margin, and the optimal separating hyperplane (blue line) bisects the slab. Included in the figure is the boundary found using logistic regression (red line), which is very close to the optimal separating hyperplane (see Section 12.3.3).*

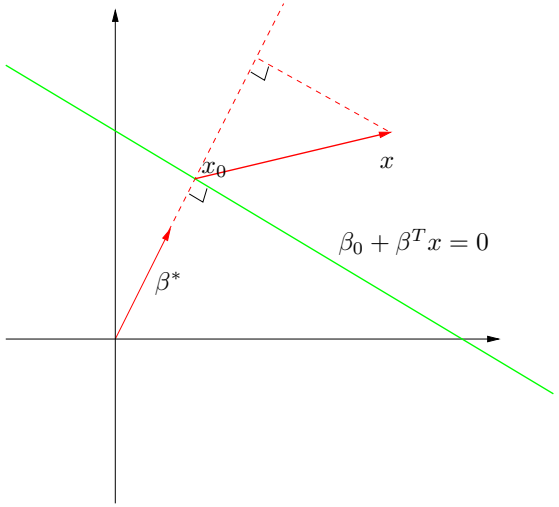
Optimal Separating Hyperplane (separble case)

We define **Optimal Separating Hyperplane** as a separating hyperplane with maximal free space C without any data point around the hyperplane.
Formally:

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

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





Formally:

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

subject to $y_i(x_i^T\beta + \beta_0) \geq C$ for all $i = 1, \ldots, N$.
We re-define: $\|\beta\| = 1$ can be moved to the condition (and redefine β_0):

$$\frac{1}{\|\beta\|} y_i(x_i^T\beta + \beta_0) \geq C$$

Since for any β and β_0 satisfying these inequalities, any positively scaled multiple satisfies them too, we can set $\|\beta\| = \frac{1}{C}$ and we get:

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$$

subject to $y_i(x_i^T\beta + \beta_0) \geq 1$ pro $i = 1, \ldots, N$.
This is a convex optimization problem. The Lagrange function, to be minimized w.r.t. β and β_0, is:

$$L_P = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i[y_i(x_i^T\beta + \beta_0) - 1].$$

$$L_P = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(x_i^T \beta + \beta_0) - 1].$$

Setting the derivatives to zero, we obtain:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i$$

Substituing these in L_P we obtain the so–called Wolfe dual:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

subject to $\alpha_i \geq 0$
The solution is obtained by maximizing L_D in the positive orthant, for which standard software can be used.

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

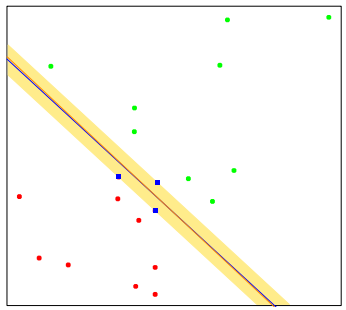
subject to $\alpha_i \geq 0$
In addition the solution must satisfy the Karush–Kuhn–Tucker conditions:

$$\alpha_i [y_i (x_i^T \beta + \beta_0) - 1] = 0$$

for any i, therefore for any $\alpha_i > 0$ must $[y_i (x_i^T \beta + \beta_0) - 1] = 0$, that means x_i is on the boundary of the slab and for all x_i outside the boundary is $\alpha_i = 0$.
The boundary is defined by x_i with $\alpha_i > 0$ – so called **support vectors**.

We classify new observations
$\hat{G}(x) = sign(x^T \beta + \beta_0)$

- where $\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$,
- $\beta_0 = y_s - x_s^T \beta$ for any support vector $\alpha_s > 0$.



Optimal Separating Hyperplane (nonseparble case)

- We have to accept incorrectly classified instances in a non–separable case.
- We limit the number of incorrectly classified examples.

We define **slack** ξ for each data point $(\xi_1, \ldots, \xi_N) = \xi$ as follows:

- ξ_i is the distance of x_i from the boundary for x_i at the wrong side of the margin
- and $\xi_i = 0$, for x_i at the correct side.

We require $\sum_{i=1}^{N} \xi_i \leq K$.

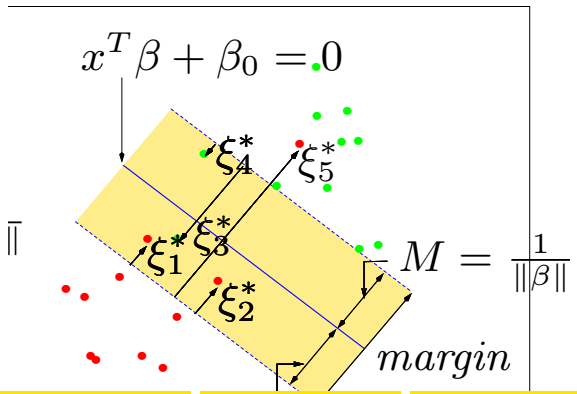
We solve the optimization problem

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

subject to:

$$y_i(x^T\beta + \beta_0) \geq C(1 - \xi_i)$$

where $\forall i$ is $\xi_i \geq 0$ a $\sum_{i=1}^{N} \xi_i \leq K$.



Optimal Separating Hyperplane (nonseparble case)

Again, we omit replace the condition $\|\beta\|$ by defining $C = \frac{1}{\|\beta\|}$ and optimize

$$\min \|\beta\| \text{subject to} \{ \begin{array}{c} y_i(x^T\beta + \beta_0) \geq (1 - \xi_i) \forall i \\ \xi_i \geq 0, \sum \xi_i \leq \textit{constant} \end{array} \tag{10}$$

We solve

$$\min_{\beta, \beta_0} \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \xi_i$$

subject to $\xi_i \geq 0$ and $y_i(x^T\beta + \beta_0) \geq (1 - \xi_i)$ where γ has replaced the constant$K$, we cans set $\gamma = \infty$ for the separable case.

We solve

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \xi_i$$

subject to $\xi_i \geq 0$ and $y_i(x^T\beta + \beta_0) \geq (1 - \xi_i)$ where γ has replaced the constant$K$, we cans set $\gamma = \infty$ for the separable case.
Lagrange multiplyers again for α_i, μ_i:

$$L_P = \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i[y_i(x_i^T\beta + \beta_0) - (1 - \xi_i)] - \sum_{i=1}^{N} \mu_i\xi_i$$

Setting the derivative $= 0$ we get:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i$$

$$\alpha_i = \gamma - \mu_i$$

Substitute to get Wolfe dual:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

and maximize L_D subject to $0 \leq \alpha_i \leq \gamma$ a $\sum_{i=1}^{N} \alpha_i y_i = 0$.
Solution satisfyes:

$$\begin{aligned} \alpha_i [y_i (x_i^T \beta + \beta_0) - (1 - \xi_i)] &= 0 \\ \mu_i \xi_i &= 0 \\ \left[y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) \right] &\geq 0 \end{aligned}$$

Solution: $\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i$
with nonzero coefficients $\widehat{\alpha_i}$ – these are boundary points where $\widehat{\xi_i} = 0$ (therefore $0 < \widehat{\alpha_i} < \gamma$), and points on the wrong side of the margin $\widehat{\xi_i} > 0$ (and $\widehat{\alpha_i} = \gamma$. Any point with $\widehat{\xi_i} = 0$ can be used to calculate $\widehat{\beta_0}$, typically an average is taken.
$\widehat{\beta_0}$ for a boundary point $\xi_i = 0$:

$$\alpha_i \left[y_i (x^T \widehat{\beta} + \widehat{\beta_0}) - (1 - 0) \right] = 0$$

Parameter γ settled by tuning (crossvalidation).

Support Vector Machines

- Genaralization

Let us have the training data $(x_i, y_i)_{i=1}^N$, $x_i \in \mathbb{R}^p$, y_i in $\{-1, 1\}$. We define a hyperplane

$$\{x : f(x) = x^T \beta + \beta_0 = 0\} \tag{11}$$

where $\|\beta\| = 1$.
We classify according to

$$G(x) = sign\left[x^T \beta + \beta_0\right]$$

where $f(x)$ is a signed distance of x from the hyperplane.

Kernels

- We searched for a linear boundary.
- We enhance the feature space to nonlinear one.
- Imagine M functions $h_m(x)$, $m = 1, \ldots, M$.
- $h(x_i) = (h_1(x), \ldots, h_M(x))$ and a decision boundary $\widehat{f}(x) = h(t)^T \widehat{\beta} + \widehat{\beta_0}$.

The classification function $\hat{f}(x) = h(x)^T \beta + \beta_0 = \sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i) \rangle + \beta_0$ does not need evalutation of $h(i)$, only the scalar product $\langle h(x), h(x_i) \rangle$.

- **kernal functions** are function to replace scalar product with a scalar product in a transformed space.

dth Degree polynomial:	$K(x, x^{\mid}) = (1 + \langle x, x^{\mid} \rangle)^d$
Radial basis	$K(x, x^{\mid}) = exp(\frac{-\|x - x^{\mid}\|^2}{c})$
Neural network	$K(x, x^{\mid}) = tanh(\kappa_1 \langle x, x^{\mid} \rangle + \kappa_2)$

Example: Degree 2 with two dimensional input:
$K(x, x') = (1 + \langle x, x' \rangle)^2 = (1 + 2x_1 x_1' + 2x_2 x_2' + (x_1 x_1')^2 + (x_2 x_2')^2 + 2x_1 x_1' x_2 x_2')$
that is. $M = 6$, $h_1(x) = 1$, $h_2(x) = \sqrt{2} x_1$, $h_3(x) = \sqrt{2} x_2$,
$h_4(x) = x_1^2$, $h_5(x) = x_2^2$, $h_6(x) = \sqrt{2} x_1 x_2$.

Table of Contens