Support Vector Machine and Discriminant Analysis

Yanrong Yang

RSFAS/CBE, Australian National University

4th October 2022

Contents of this week

Two classification methods

- Review Setting-up of Classification Problems
- Support Vector Machines (SVM)
- Discriminant Analysis (DA): Flexible DA, Penalized DA, Mixture DA

Review on Classification

Classification Problem

Setup of Classification Problem:

- Observations $x_1, x_2, \ldots, x_n$ are from K different classes $C_1, C_2, \ldots, C_K$.
- For each observation x_i, the class that x_i belongs to is observed, i.e. $x_i \in C_k$ for some k.
- The aim is to construct a classification rule (or classifier) based these observations.
- Classifier can be used to predict which class does any new observation (or test data) belongs to.

Format of Classification

Regression Format for Classification

$$y_i = f(x_i) + \varepsilon_i, \quad i = 1, 2, \ldots, n, \tag{1}$$

where

1. $x_1, x_2, \ldots, x_n$ are observations;
2. y_i is class C_k that x_i belongs to;
3. f is an unknown function and ε_i is error component.

Remark: the response y_i for classification is qualitative! f is the classifier.

An Ideal Classifier: Bayesian Classifier

Bayes Classifier is an ideal classifier which assigns a test observation with predictor x_0 to the class j for which

$$Pr\left(Y = j | X = x_0\right)$$

is largest.
In two-class problem, the Bayes Classifier predicts class one if $Pr\left(Y = 1 | X = x_0\right) > 0.5$ and class two otherwise.

Example: Default Data

Let's consider the Default data set, where the response default falls into one of two categories, Yes or No.

There are four predictors for this data set:

1. default: A factor with levels No and Yes indicating whether the customer defaulted on their debt
2. student: A factor with levels No and Yes indicating whether the customer is a student
3. balance: The average balance that the customer has remaining on their credit card after making their monthly payment
4. income: Income of customer

One goal is to predict which customers will default on their credit card debt.

Example: Default Data

Rather than modeling this response Y directly, logistic regression models the probability that Y belongs to a particular category.

For the Default data, logistic regression models the probability of default (defaulting on credit card debt).

For example, the probability of default given balance can be written as

$$Pr(default = Yes|balance).$$

We will denote the above probability by $p(balance)$ for convenience.

Estimation of Bayes Classifier: Linear Regression

How should we model the relationship between

$$p(X) = Pr(Y = 1 \mid X) \quad \text{and} \quad X?$$

Recall the linear regression model

$$p(X) = \beta_0 + \beta_1 X$$

where the goal is to use default = Yes to predict balance.

Estimation of Bayes Classifier: Logistic Regression

We model $p(X)$ using a function that gives outputs between 0 and 1 for all values of X, where

$$p(X) = \beta_0 + \beta_1 X$$

In logistic regression, we use the logistic function,

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}}. \quad (1)$$

Estimation of Bayes Classifier: LDA

1. Utilizing the Bayes classifier, which classifies an observation $X = x$ to the class for which $Pr(Y = k|X = x)$ is largest among all the K classes;
2. Estimating $Pr(Y = k|X = x)$ based on Bayes's theorem

$$Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{\ell=1}^{K} \pi_\ell f_\ell(x)}, \tag{1}$$

where π_k: is the prior probability, that is the probability that a given observation is associated with the kth category of the response variable Y; $f_k(x) = Pr(X = x|Y = k)$.

Assessment for Classifiers

Training Error Rate is

$$\frac{1}{n}\sum_{i=1}^{n} I\left(y_i \neq \widehat{y}_i\right), \tag{2}$$

where $\widehat{y}_i$ is the predicted class label for the observation y_i with $\widehat{f}$.
Test Error Rate is

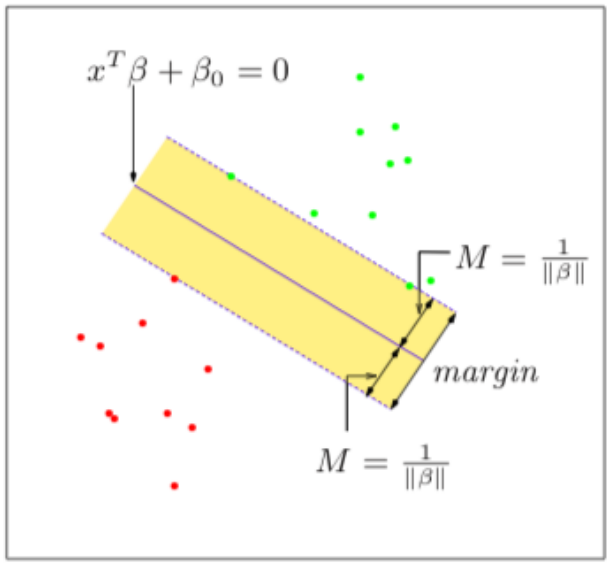
$$\frac{1}{m}\sum_{j=1}^{m} I\left(y_j^t \neq \widehat{y}_j^t\right), \tag{3}$$

where $\widehat{y}_j^t$ is the predicted class label for the test data $\left(x_j^t, y_j^t\right)$, $j = 1, 2, \ldots, m$.

Review on Support Vector Classifier

Separable Case

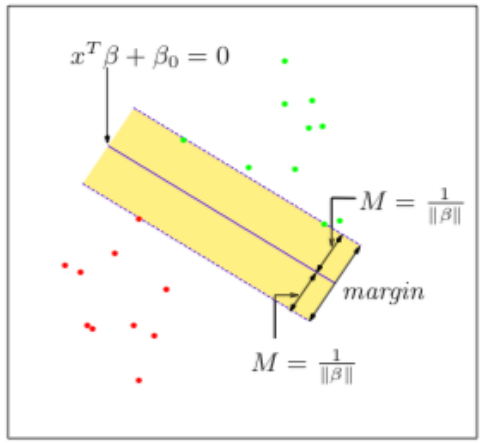
- **Training Data**: $(x_1, y_1), \ldots, (x_n, y_n)$ with $x_i \in \mathbb{R}^p; y_i \in \{-1, 1\}$.
- **Aim**: Find the separating hyperplane with biggest margin between training points for class $+1$ and -1.



- **Optimization problem** (if the margin is $2M$)

$$\max_{\beta, \beta_0, \|\beta\|=1} M \quad \text{subject to} \quad y_i(x_i^t\beta + \beta_0) \geq M, i = 1, \ldots, n$$

Separable Case



- **Rephrasing the optimization problem**
  - Distance of x_i to decision boundary is $|x_i^t\beta + \beta_0|/\|\beta\|$
  - Let the x_i's on margin have $|x_i^t\beta + \beta_0| = 1$
  - Then $M = 1/\|\beta\|$

  Then optimization problem can be restated as

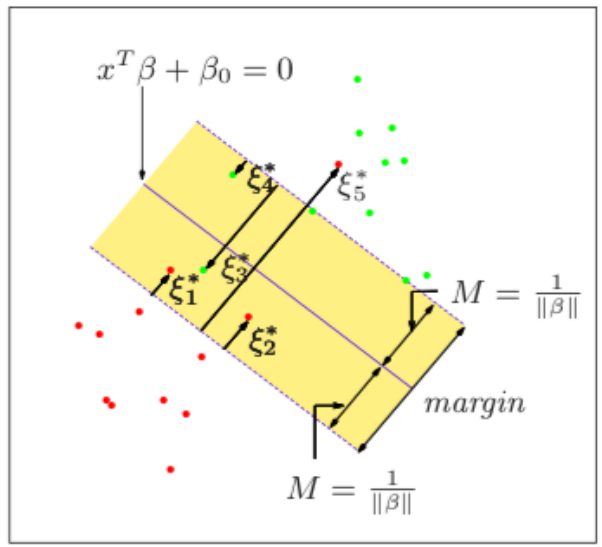
$$\min_{\beta,\beta_0} \|\beta\|^2 \quad \text{subject to} \quad y_i(x_i^t\beta + \beta_0) \geq 1,\ i = 1, \ldots, n$$

- This is a **QP** problem with linear inequality constraints.

Nonseparable Case

- **Training Data**: $(x_1, y_1), \ldots, (x_n, y_n)$ overlap

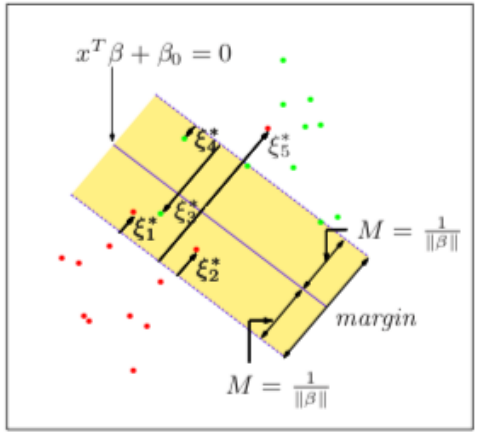
  $\Longrightarrow$ no feasible solution to previous optimization



- **Instead**: Find hyperplane with biggest margin but allow some points to be on the wrong side of the margin.

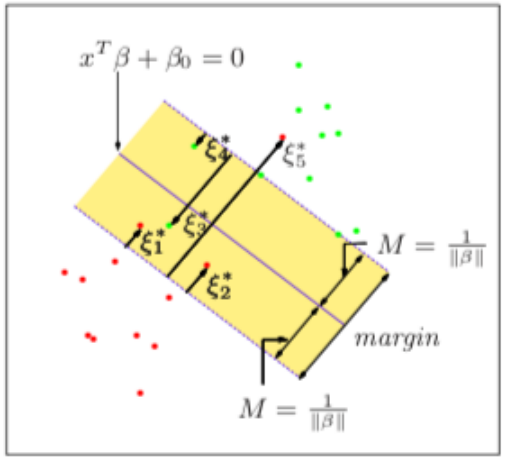
- Introduce slack variables $\xi = (\xi_1, \xi_2, \ldots, \xi_n)$ for each example with $\xi_i \geq 0$.

Nonseparable Case



- Modify constraints: $y_i(x_i^t\beta + \beta_0) \geq M$ to $y_i(x_i^t\beta + \beta_0) \geq M - \xi_i$ with $\xi_i \geq 0,\ \forall i$ and $\sum \xi_i \leq C$
- Intuitively appealing - measure amount of overlap in real distances to margin - but it does not lead to a convex problem.
- **Instead**: Modify $y_i(x_i^t\beta + \beta_0) \geq M$ to $y_i(x_i^t\beta + \beta_0) \geq M(1 - \xi_i)$ with $\xi_i \geq 0,\ \forall i$ and $\sum \xi_i \leq C$
- Overlap now measured in relative distances which changes with M but now have a convex problem !

Optimization Formulation



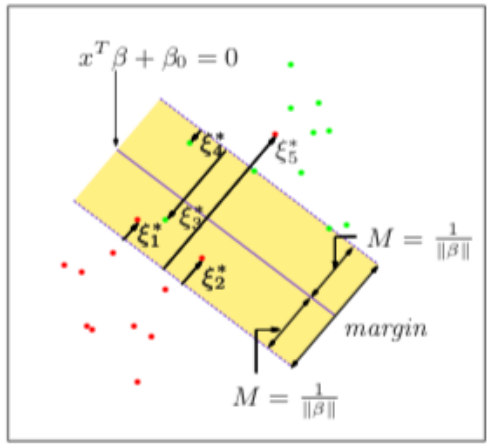
- $\xi_i \geq 0$ in

$$y_i(x_i^t\beta + \beta_0) \geq M(1 - \xi_i)$$

is the proportional amount by which the prediction $f(x_i) = x_i^t\beta + \beta_0$ is on the wrong side of the margin.

- $\xi > 1 \implies$ prediction on wrong side of margin
- $\sum \xi_i \leq K \implies$ at most K training pts on wrong side of margin.

Optimization Formulation



- **Optimization Statement I**

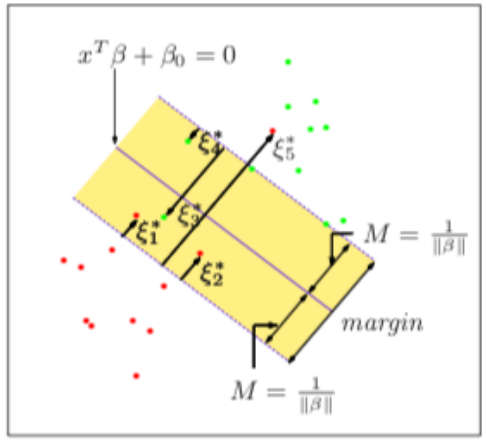
$$\max_{\beta,\beta_0,\|\beta\|=1} M \quad \text{subject to} \quad y_i(x_i^t\beta + \beta_0) \geq M(1-\xi_i),\ \forall i$$

$$\xi_i \geq 0,\ \forall i, \quad \sum \xi_i \leq \text{constant}$$

- Can drop constraint $\|\beta\| = 1$ by choosing $|x_i^t\beta + \beta_0| = 1$ for x_i on the margin.

- $\implies M = 1/\|\beta\|$ **and**
  constraint $y_i(x_i^t\beta + \beta_0) \geq M(1-\xi_i)$ becomes $y_i(x_i^t\beta + \beta_0) \geq 1 - \xi_i$.

Optimization Formulation



- **Optimization Statement II**

$$\min_{\beta,\beta_0} \|\beta\|^2 \quad \text{subject to} \quad y_i(x_i^t\beta + \beta_0) \geq 1 - \xi_i, \ \forall i$$

$$\xi_i \geq 0, \ \forall i, \quad \sum \xi_i \leq \text{constant}$$

- Note x_i's inside their class boundary do not play a role in shaping the decision boundary.
- $\implies$ $\beta^{(\text{SVM})}$ differs from $\beta^{(\text{LDA})}$ as $\beta^{(\text{LDA})}$ is influenced by all the training points.

Computation

- **Optimization Statement III**

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}\xi_i \quad \text{subject to} \quad y_i(x_i^t\beta+\beta_0) \geq 1-\xi_i,\ \forall i$$

$$\xi_i \geq 0,\ \forall i$$

where cost parameter C replaces the constraint $\sum \xi_i \leq$ constant (separable case corresponds to $C = \infty$.)

- To solve this constrained optimization construct its Lagrangian

$$\mathcal{L}_P(\beta,\beta_0,\xi,\alpha,\mu) =$$

$$\frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\left[y_i(x_i^t\beta+\beta_0) - (1-\xi_i)\right] - \sum_{i=1}^{n}\mu_i\,\xi_i$$

with $\alpha_i \geq 0$ and $\mu_i \geq 0\ \forall i$.

- If $(\beta^*,\beta_0^*,\xi^*)$ is minimum for the constraint problem then $\exists\alpha^*$ and μ^* s.t. $(\beta^*,\beta_0^*,\xi^*,\alpha^*,\mu^*)$ is a stationary point of $\mathcal{L}_p$.

Dual Optimization Problem

- Set the derivatives $\partial \mathcal{L}_p / \partial \beta, \ldots$ to 0 gives

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

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

$$\alpha_i = C - \mu_i, \ \forall i$$

- Remember $\alpha_i, \mu_i, \xi_i \geq 0, \ \forall i$
- By substituting these into the Lagrangian get the dual

$$\begin{aligned} \mathcal{L}_D(\alpha, \mu) &= \inf_{\beta, \beta_0, \xi} \mathcal{L}_P(\beta, \beta_0, \xi, \alpha, \mu) \\ &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \, \alpha_{i'} \, y_i \, y_{i'} \, x_i^t \, x_{i'} \end{aligned}$$

Dual Optimization Problem

- Maximize

$$\mathcal{L}_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \, \alpha_{i'} \, y_i \, y_{i'} \, x_i^t \, x_{i'}$$

  subject to the constraints

$$0 \leq \alpha_i \leq C \text{ for } i = 1, \ldots, n \quad \text{and} \quad \sum_i \alpha_i y_i = 0$$

- An easier QP than the original primal problem especially if $p > n$.
- Solving the dual problem for the SVM is equivalent to solving the primal problem as
  - The primal problem is convex **and**
  - constraints are affine

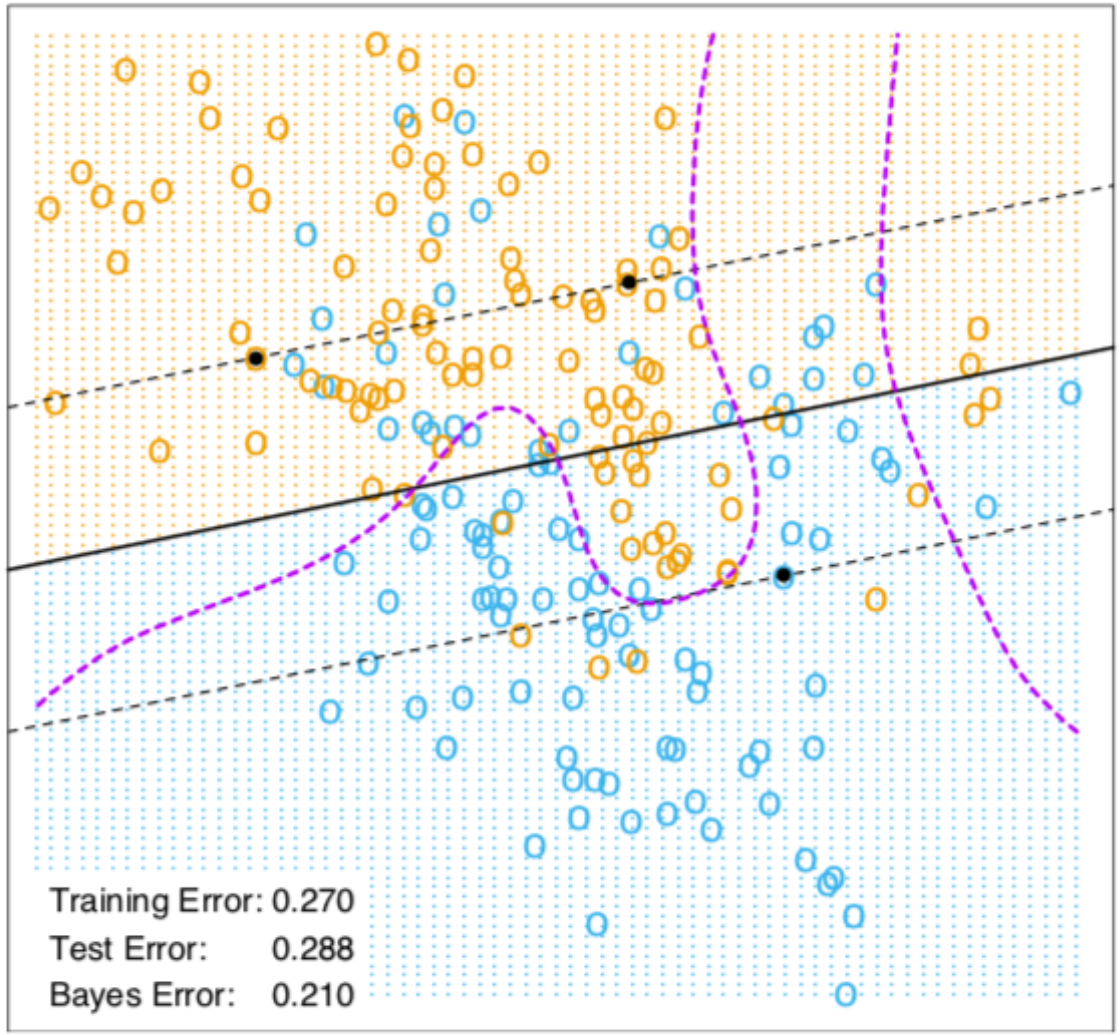
Solution to SVC

- Solution for β has the form

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

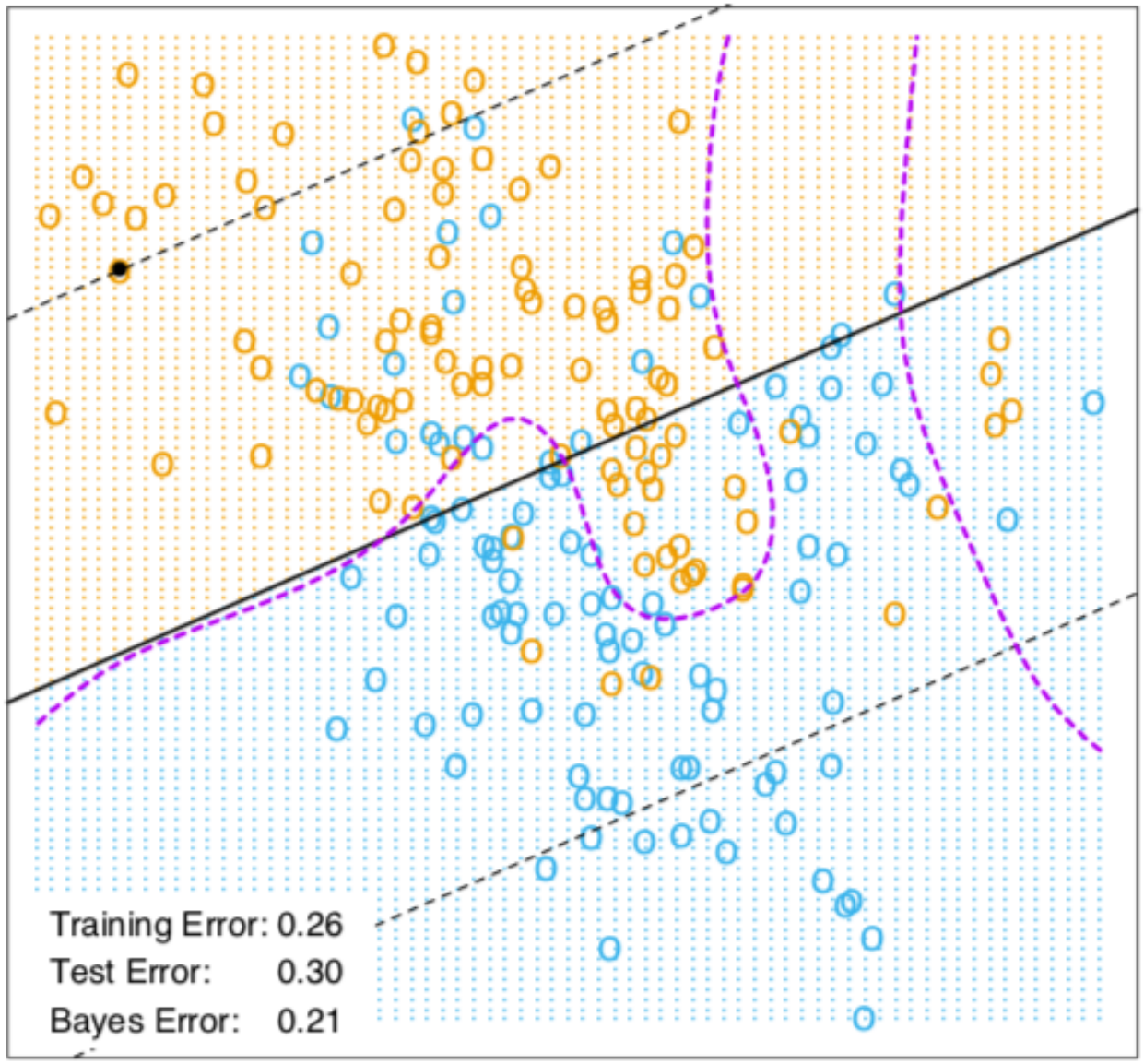
  - $\hat{\alpha}_i$ is non-zero only if $y_i(\hat{\beta}_i^t \, x_i + \hat{\beta}_0) = 1 - \xi_i$
  - These observations are the support vectors.
  - For support vectors with $\xi_i = 0$ (on the margin) then $0 < \hat{\alpha}_i < C$
  - The other support vectors have $\xi_i > 0$ with $\hat{\alpha}_i = C$

Example: SVC



$$C = 10000$$

Example: SVC



$C = 0.01$

Support Vector Machine (SVM)

SVM for Classification

- Can find non-linear decision boundaries by using basis expansion.
- Select basis functions $h_m(x), m = 1, \ldots, M$ and proceed as before.
- Fit SV classifier using inputs $h(x_i) = (h_1(x_i), \ldots, h_M(x_i))^t$
- This produces the non-linear function

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

- The classifier is $\hat{G}(x) = \text{sign}\{\hat{f}(x)\}$
- The Support Vector Machine is an extension of this idea where M can get very large or even infinite.

SVM for Classification

- Can represent the optimization problem

$$\max_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}\xi_i \quad \text{subject to} \quad y_i(h(x_i)^t\beta + \beta_0) \geq 1 - \xi_i,\ \forall i$$

$$\xi_i \geq 0,\ \forall i$$

  and its solution that only involves input features via inner products.

- $\Longrightarrow$ for particular choices of h these inner products can be computed very cheaply.

Kernel Representation for SVM

- The dual optimization problem is to maximize

$$\mathcal{L}_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle h(x_i), h(x_j) \rangle$$

  subject to $\sum_i \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$ for $i = 1, \ldots n$.

- The solution function $\hat{f}(x)$ can be written as

$$\begin{aligned} \hat{f}(x) &= h(x)^t \hat{\beta} + \hat{\beta}_0 \\ &= \sum_{i=1}^{n} \hat{\alpha}_i y_i \langle h(x), h(x_i) \rangle + \hat{\beta}_0 \end{aligned}$$

- Both these equations only involve $h(x)$ through inner-products.

- $\Longrightarrow$ only need knowledge of the **kernel function** that computes the inner-product in the transformed space

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

Kernel Representation for SVM

- The solution $\hat{f}$ can then be written as

$$\begin{aligned}\hat{f}(x) &= h(x)^t \hat{\beta} + \hat{\beta}_0 \\ &= \sum_{i=1}^{n} \hat{\alpha}_i y_i \langle h(x), h(x_i) \rangle + \hat{\beta}_0 \\ &= \sum_{i=1}^{n} \hat{\alpha}_i y_i K(x, x_i) + \hat{\beta}_0\end{aligned}$$

- K should be a symmetric positive semi-definite function.

- Popular choices for K in the SVM literature are

dth-Degree polynomial:	$K(x, x') = (1 + \langle x, x' \rangle)^d$
Radial basis:	$K(x, x') = \exp\{-\gamma \|x - x'\|^2\}$
Neural network:	$K(x, x') = \tanh\{\gamma_1 \langle x, x' \rangle + \gamma_2\}$

Example of Kernel Functions

- Consider a 2D feature vector $X = (X_1, X_2)$ and a polynomial kernel of degree 2:

$$\begin{aligned} K(X, X') &= (1 + \langle X, X' \rangle)^2 \\ &= (1 + X_1 X_1' + X_2 X_2')^2 \\ &= 1 + 2X_1 X_1' + 2X_2 X_2' + (X_1 X_1')^2 + (X_2 X_2')^2 + 2X_1 X_1' X_2 X_2' \end{aligned}$$

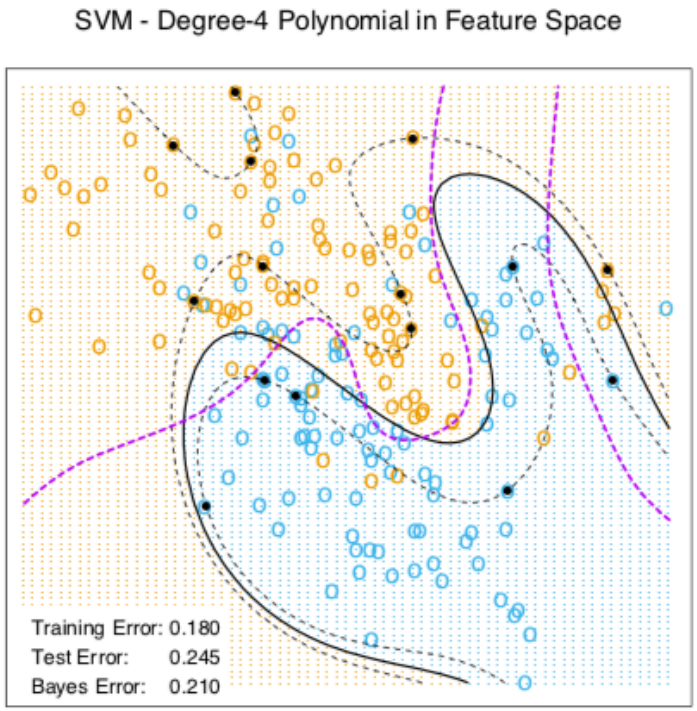
- In this case $M = 6$ with

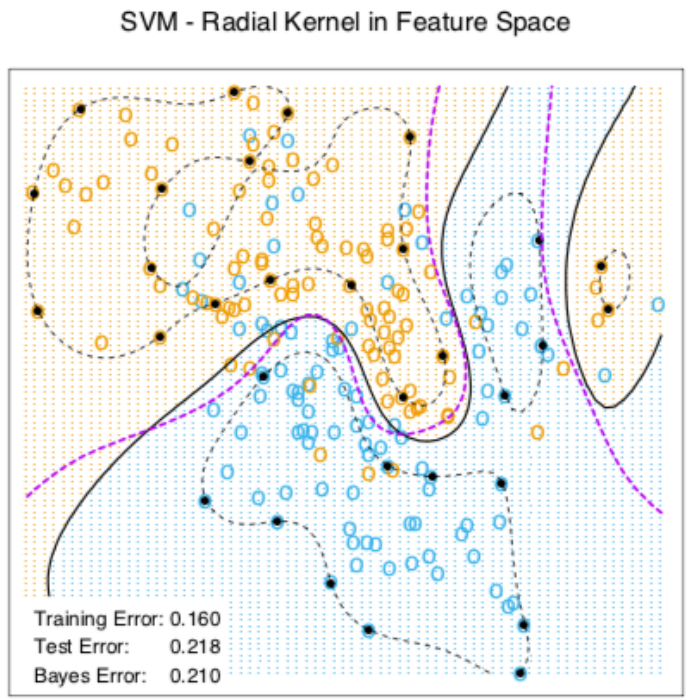
$$\begin{aligned} h_1(X) &= 1 & h_2(X) &= \sqrt{2}\, X_1 \\ h_3(X) &= \sqrt{2}\, X_2 & h_4(X) &= X_1^2 \\ h_5(X) &= X_2^2 & h_6(X) &= \sqrt{2}\, X_1 X_2 \end{aligned}$$

- Obviously then

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

Example: SVM





$C = 1$ in both cases

Dashed purple line is the Bayes decision boundary.

Dashed black lines are $yf(x) = \pm 1$.

The Role of C

- Large C discourages any positive ξ_i which may lead to overfit wiggly boundary.
- Small C encourages a small value of $\|\beta\| \implies$ a smoother decision boundary.

SVM: Penalization Method

With $f(x) = h(x)^t\beta + \beta_0$ consider this optimization problem:

$$\min_{\beta,\beta_0} \sum_{i=1}^{n} \max(0, 1 - y_i f(x_i)) + \frac{\lambda}{2}\|\beta\|^2$$

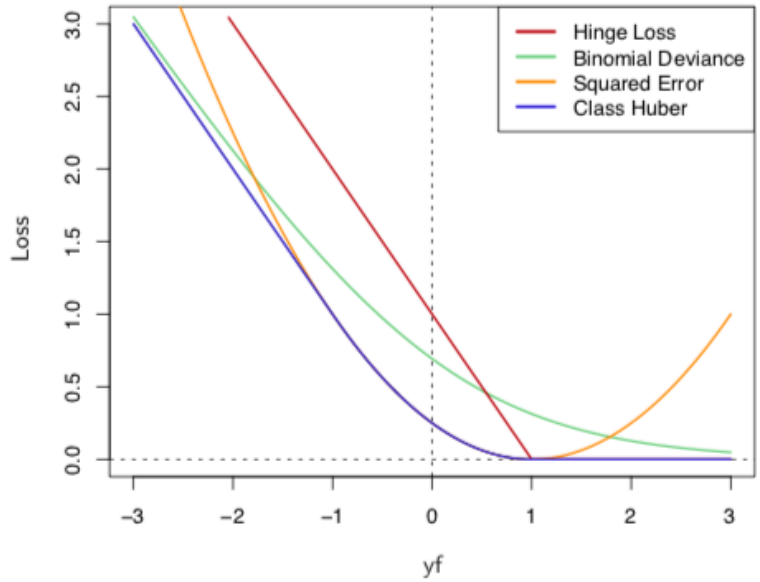
- This cost function has the form

  Loss $+$ **Penalty**

- soln this optimization problem $=$ soln to typical formulation of SVM if $\lambda = 1/C$.

- Loss function $\max(0, 1 - yf(x))$ is known as the **Hinge Loss**.

Common Loss Functions



Loss function	$L(y, f(x))$	Minimizing $f(x)$
Binomial deviance	$\log(1+\exp\{-yf(x)\})$	$\log \frac{P(Y=1\|x)}{P(Y=0\|x)}$
SVM Hinge Loss	$\max\{0,1-yf(x)\}$	$\text{sign}\{P(Y=1\|x)-.5\}$
Squared error	$(1-yf(x))^2$	$2P(Y=1\|x)-1$
Huberised squared err	$\begin{cases} -4yf(x) & \text{if } yf(x) < 1 \\ \max\{0,1-yf(x)\} & \text{otherwise} \end{cases}$	$2P(Y=1\|x)-1$

SVM: Curse of Dimensionality

- For the 2nd degree polynomial kernel of p dimensional features is

$$K(X, X') = 1 + 2\sum_{j=1}^{p} X_j X'_j + \sum_{j=1}^{p}(X_j X'_j)^2 + 2\sum_{i=1}^{p}\sum_{j=i+1}^{p} X_i X'_i X_j X'_j$$

- Kernel cannot adapt to concentrate on just a subset of the features.
- Disaster if p is large and only a small subset of features are relevant.
- If knew beforehand which features were relevant could adapt the kernel accordingly, but....
- $\Longrightarrow$ SVM kernel methods alone are not ideal for discovering structure.

SVM: Curse of Dimensionality

- Case I: No noise features
  - Class 1:

    $X = (X_1, X_2, X_3, X_4)^t$ where each $X_i \sim \mathcal{N}(0, 1)$ and are indept.
  - Class 2:

    $X = (X_1, X_2, X_3, X_4)^t$ with $X_i \sim \mathcal{N}(0, 1)$ and $9 \leq \sum_{j=1}^{4} X_j^2 \leq 16$
- Case II: case I augmented with noise features
  - Class 1:

    $X = (X_1, \ldots, X_{10})^t$ where each $X_i \sim \mathcal{N}(0, 1)$ and are indept.
  - Class 2:

    $X = (X_1, \ldots, X_{10})^t$ with $X_i \sim \mathcal{N}(0, 1)$ and $9 \leq \sum_{j=1}^{4} X_j^2 \leq 16$

SVM: Curse of Dimensionality

For the two cases

- Generated 100 training examples for each class and 1000 test examples.
- Ran 50 simulations and averaged the test error.

		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

BRUTO fits an additive spline model adaptively.
MARS fits a low-order interaction model adaptively.

SVM: Curse of Dimensionality

	Method	Test Error (SE)	
		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

- **Note**: For each SVM classifier an optimal C was used.
- A hyperplane cannot separate classes $\Longrightarrow$ linear SVM does poorly.
- Polynomial SVMs do better but are affected by the noise features.
- Second-degree polynomial kernel does best as true decision boundary is a 2nd-degree polynomial.
- Higher degree kernels do much worse.

Generalization of LDA

- **FlexibleDA**
  - Recast LDA problem as a linear regression problem.
  - Generalize linear regression to more flexible, nonparametric regression.
- **PenalizedDA**
  - Have a high dimensional and correlated set of predictors.
  - Fit an LDA model, but penalize its coefficients to be smooth.
- **MixtureDA**
  - Model each class by $K \geq 2$ Gaussians with different centroids.
  - Every component Gaussian has the same covariance matrix .

Optimal Scoring

- Have K classes each with a label $\{1, 2, \ldots, K\}$
- Let function

$$\theta : \{1, 2, \ldots, K\} \to \mathbb{R}$$

assign a score to each class.

- For data $\{(x_i, g_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}^p$ and $g_i \in \{1, 2, \ldots, K\}$ want to solve

$$\min_{\beta, \theta} \sum_{i=1}^{n} (\theta(g_i) - x_i^t \beta)^2$$

subject to

$$\sum_i \theta(g_i) = 0 \quad \text{and} \quad \frac{1}{n} \sum_i \theta^2(g_i) = 1$$

Optimal Scoring in Matrix Notation

- Create the $n \times K$ indicator response matrix Y, that is

$$Y_{ik} = \begin{cases} 1 & \text{if } g_i = k \\ 0 & \text{otherwise} \end{cases}$$

- Let $\Theta = (\theta(1), \theta(2), \ldots, \theta(k))^t$

- The optimization problem can be written as

$$\min_{\Theta,\beta} (Y\Theta - X\beta)^t (Y\Theta - X\beta)$$

subject to

$$\frac{1}{n}\Theta^t Y^t Y \Theta = 1$$

Solution to Optimal Scoring

- The optimization problem

$$\min_{\Theta,\beta} (Y\Theta - X\beta)^t (Y\Theta - X\beta) \quad \text{subject to } \frac{1}{n}\Theta^t Y^t Y \Theta = 1$$

is solved by

$$\hat{\beta} = (X^t X)^{-1} X^t Y \hat{\Theta}$$

where $\hat{\Theta}$ is a solution to the generalized e-value problem:

$$Y^t X (X^t X)^{-1} X^t Y \Theta = \mu Y^t Y \Theta$$

More General Optimal Scoring

- More generally can find up to $L \leq K-1$ indept scorings

$$\theta_l : \{1, 2, \ldots, K\} \to \mathbb{R}, \quad l = 1, \ldots, L$$

and L vectors of linear coefficients $\beta_1, \ldots, \beta_L$.

- Want to solve

$$\min_{\{\beta_l, \theta_l\}} \sum_{l=1}^{L} \sum_{i=1}^{n} (Y\Theta_l - x_i^t \beta_l)^2$$

subject to

$$\tfrac{1}{n} \Theta_l^t Y^t Y \Theta_l = 1 \quad l = 1, \ldots, L$$

and

$$\Theta_l^t \, \Theta_k = 0 \quad \text{for } l \neq k$$

Connection with LDA

- The sequence of discriminant vectors by LDA are identical to the sequence $\hat{\beta}_l$ up to a constant.

- Mahalanobis distance of point x to the kth class centroid $\hat{\mu}_k$

$$\delta(x, \hat{\mu}_k) = \sum_{l=1}^{K-1} w_l (\hat{\beta}_l^t x - \hat{\beta}_l^t \mu_k)^2 + D(x)$$

  where w_l is defined in terms of the mean squared residual r_l^2 of the lth optimally scored fit

$$w_l = \frac{1}{r_l^2 (1 - r_l^2)}$$

- LDA can be performed by a sequence of linear regressions, followed by classification to the closest class centroid in the space of fits....

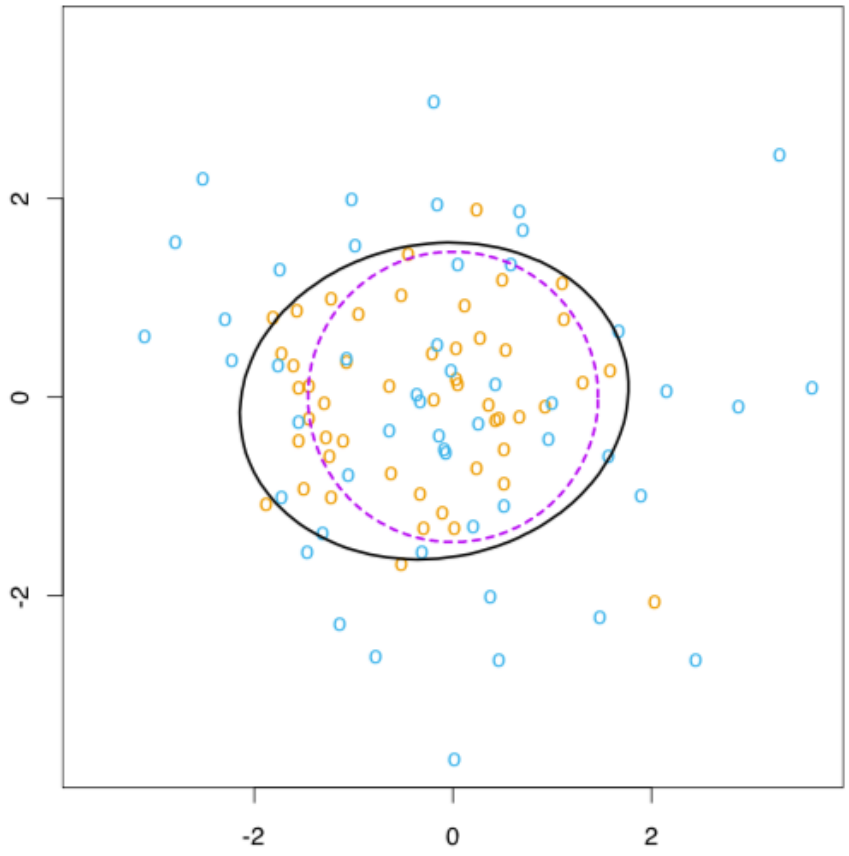
FDA

- Can replace the linear regression fits $x_i^t \beta_l$ with more flexible, nonparametric fits: $\eta_l(x)$
- $\Longrightarrow$ more flexible classifier than LDA
- General form of the regression problem

$$\text{ASR}(\{(\theta_l, \eta_l)\}_{l=1}^L) = \frac{1}{n} \sum_{l=1}^{L} \left[\sum_{i=1}^{n} (\theta_l(g_i) - \eta_l(x_i))^2 + \lambda J(\eta_l) \right]$$

where J is an appropriate regularizer.

Example



Dashed purple line is the Bayes decision boundary.

Black ellipse is the decision boundary found by FDA using 4 degree-two polynomial regression.

Example: Vowel Data

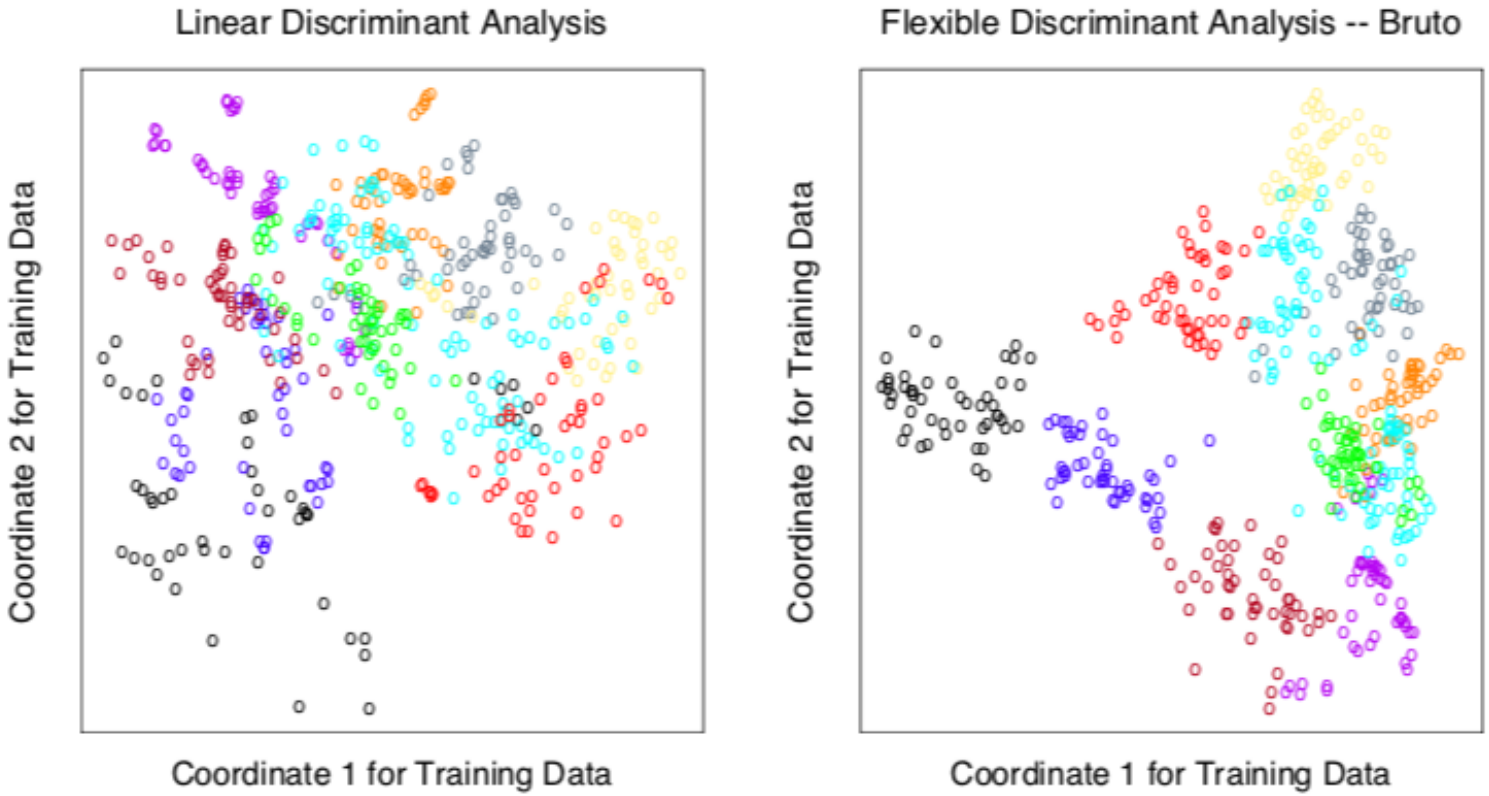


FIGURE 12.10. *The left plot shows the first two LDA canonical variates for the vowel training data. The right plot shows the corresponding projection when FDA/BRUTO is used to fit the model; plotted are the fitted regression functions* $\hat{\eta}_1(x_i)$ *and* $\hat{\eta}_2(x_i)$*. Notice the improved separation. The colors represent the eleven different vowel sounds.*

Example: Performance Results

TABLE 12.3. *Vowel recognition data performance results. The results for neural networks are the best among a much larger set, taken from a neural network archive. The notation FDA/BRUTO refers to the regression method used with FDA.*

	Technique	Error Rates	
		Training	Test
(1)	LDA	0.32	0.56
	Softmax	0.48	0.67
(2)	QDA	0.01	0.53
(3)	CART	0.05	0.56
(4)	CART (linear combination splits)	0.05	0.54
(5)	Single-layer perceptron		0.67
(6)	Multi-layer perceptron (88 hidden units)		0.49
(7)	Gaussian node network (528 hidden units)		0.45
(8)	Nearest neighbor		0.44
(9)	FDA/BRUTO	0.06	0.44
	Softmax	0.11	0.50
(10)	FDA/MARS (degree = 1)	0.09	0.45
	Best reduced dimension (=2)	0.18	0.42
	Softmax	0.14	0.48
(11)	FDA/MARS (degree = 2)	0.02	0.42
	Best reduced dimension (=6)	0.13	0.39
	Softmax	0.10	0.50

Computation of FDA

- FDA computations simpler when can write

  non-parametric regression as a linear operator

- $\Longrightarrow \hat{y} = S_\lambda \, y,$ e.g. *additive splines*

- Optimal scoring can be computed by a single eigen-decomposition.

- The algorithm is as follows...

Computation of FDA

- Create the $n \times K$ indicator response matrix Y, that is

$$Y_{ik} = \begin{cases} 1 & \text{if } g_i = k \\ 0 & \text{otherwise} \end{cases}$$

- **Multivariate nonparametric regression**:
  - Fit a nonparametric regression of Y on X.
  - Let $\hat{Y} = S_\lambda Y$.
  - Let $\eta^*(x)$ be the vector of fitted regression functions.
- **Optimal scores**:
  - Compute the eigen-decomposition of $Y^t\hat{Y} = Y^t S_\lambda Y$
  - Eigenvectors Θ are normalized: $\Theta^t Y^t Y \Theta / n = I$
- **Update** the model: $\eta(x) = \Theta^t \eta^*(x)$.

PDA

- Consider
  - linear regression onto a basis expansion with
  - a quadratic penalty on the coefficients
- Find $\{(\theta_l, \beta_l)\}_{l=1}^{L}$ which minimize

$$\frac{1}{n}\sum_{l=1}^{L}\left[\sum_{i=1}^{n}(\theta_l(g_i) - h^t(x_i)\beta_l)^2 + \lambda\beta_l^t\Omega\beta_l\right]$$

- Choice of Ω depends on the problem.
- Generalization of LDA called penalized discriminant analysis (PDA).

Computation of PDA

- Extract a basis expansion $h(x_i)$ of each x_i.
- The penalized Mahalanobis distance to a class centroid is given by

$$D(x, \mu) = (h(x) - h(\mu))^t (\Sigma_W + \lambda\Omega)^{-1} (h(x) - h(\mu))$$

  where Σ_W = within-class covariance of the $h(x_i)$'s.

- Decompose the classification subspace using a penalized metric

$$\max_u \; u^t \Sigma_B u \quad \text{subject to} \quad u^t (\Sigma_W + \lambda\Omega) u = 1$$

Note: Penalized Mahalanobis distance

- tends to give less weight to "*rough*" coordinates and more weight to "*smooth*" ones.
- the same applies to linear combinations that are rough or smooth if Ω not diagonal

MDA

- A method for classification (supervised) based on mixture models.
- Extension of LDA
- The mixture of normals is used to obtain a density estimation for each class.

MDA and Gaussian Mixture Models

- GMM for the k-th class has density

$$P(X \mid G = k) = \sum_{r=1}^{R_k} \pi_{kr}\, \phi(X; \mu_{kr}, \Sigma)$$

  where $\sum_{r=1}^{R_k} \pi_{kr} = 1$

- Have R_k prototypes for each class and the same Σ for each component.

- Class posterior probabilities are given by

$$P(G = k \mid X) = \frac{\sum_{r=1}^{R_k} \pi_{kr}\, \phi(X; \mu_{kr}, \Sigma)\, \Pi_k}{\sum_{l=1}^{K} \sum_{r=1}^{R_l} \pi_{lr}\, \phi(X; \mu_{lr}, \Sigma)\, \Pi_l}$$

  where Π_k represents the class prior probabilities.

Estimation of MDA

- Estimate the parameters by minimizing

$$\sum_{k=1}^{K} \sum_{i \in I_k} \log \left\{ \Pi_k \sum_{r=1}^{R_k} \pi_{kr} \, \phi(x_i; \mu_{kr}, \Sigma) \right\}, \quad I_k = \{i \mid g_i = k\}$$

$\Longrightarrow$ maximize the joint-likelihood $P(G, X)$ of training data $\{(x_i, g_i)\}_{i=1}^{n}$.

- Direct optimization hard $\Longrightarrow$ use EM.

- Steps of EM

  - **E-step**: Given current estimate of parameters $\mu_{kr}^{(j)}, \pi_{kr}^{(j)}$. Compute the *responsibility* for each point

$$w_{kir}^{(j)} = \frac{\pi_{kr}^{(j)} \, \phi(x_i; \mu_{kr}^{(j)}, \Sigma)}{\sum_{l=1}^{R_k} \pi_{lr}^{(j)} \, \phi(x_i; \mu_{lr}^{(j)}, \Sigma)} \quad \text{for } r=1,\ldots,R_k;\ i \in I_k;\ k=1,\ldots,K$$

  - **M-step**: Compute the weighted MLEs for all the parameters...

Estimation of MDA

- Steps of EM
  - **E-step**: Given current estimate of parameters $\Sigma^{(j)}$, $\mu_{kr}^{(j)}$. Compute the *responsibility* for each point

$$w_{kir}^{(j)} = \frac{\pi_{kr}^{(j)}\, \phi(x_i; \mu_{kr}^{(j)}, \Sigma)}{\sum_{l=1}^{R_k} \pi_{lr}^{(j)}\, \phi(x_i; \mu_{lr}^{(j)}, \Sigma)} \quad \text{for } r=1,\ldots,R_k;\ i \in I_k;\ k=1,\ldots,K$$

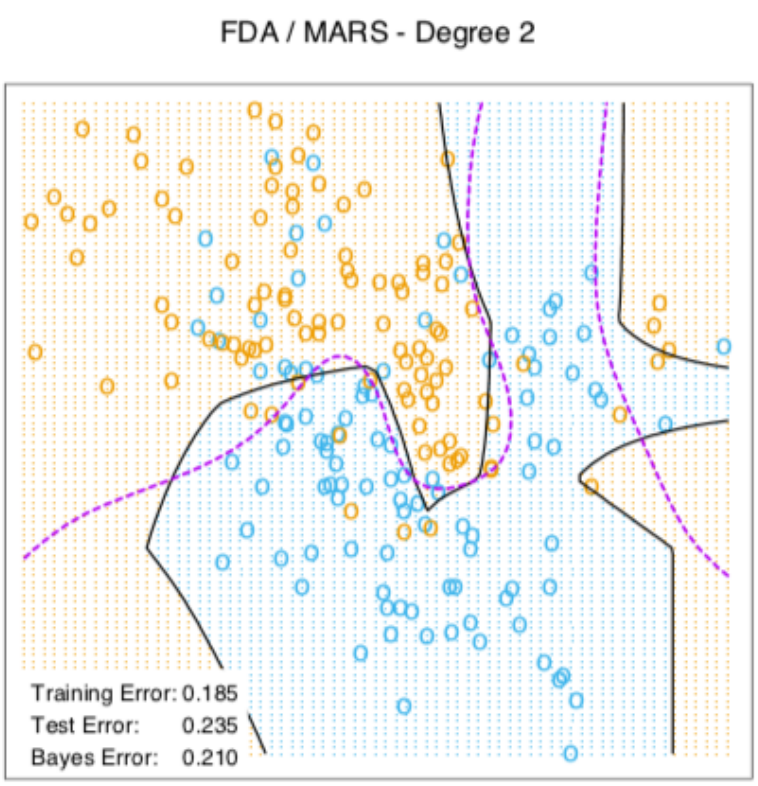
  - **M-step**: Compute the weighted MLEs for all the parameters

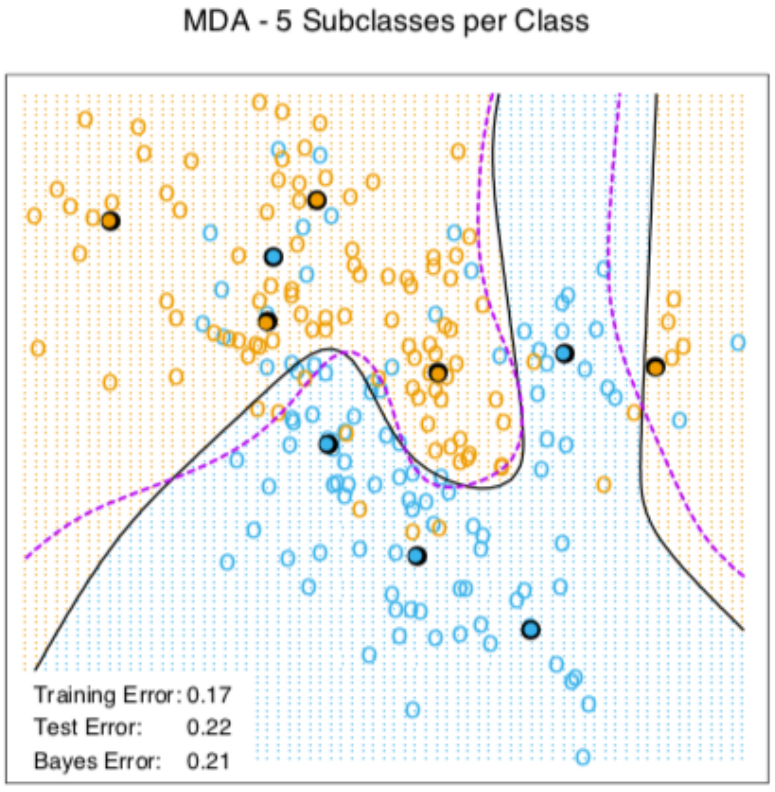
$$\mu_{kr}^{(j+1)} = \frac{\sum_{i \in I_k} w_{ikr}^{(j)}\, x_i}{\sum_{i \in I_k} w_{ikr}^{(j)}}, \qquad \pi_{kr}^{(j+1)} = \frac{\sum_{i \in I_k} w_{ikr}^{(j)}}{|I_k|}$$

$$\Sigma = \frac{1}{n} \sum_{k=1}^{K} \sum_{i \in I_k} \sum_{r=1}^{R_k} w_{ikr}^{(j)} (x_i - \mu_{kr}^{(j)})(x_i - \mu_{kr}^{(j)})^t$$

- EM requires
  - **initialization** of parameters **and**
  - setting the values R_k for $k = 1, \ldots, K$.

Example: Decision Boundaries





Dashed purple line is the Bayes decision boundary.