Principles of Statistical Machine Learning
Fundamentals of Classification Learning
Basics of Discriminant Analysis for Classification

Fokoué Ernest, PhD
Professor of Statistics

@ErnestFokoue

To understand God's thoughts, one must study statistics, the measure of His purpose
Florence Nightingale

Classification of Diabetes in the Pima Indians Community

pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age	diabet
6	148	72	35	0	33.60	0.63	50	pos
1	85	66	29	0	26.60	0.35	31	neg
8	183	64	0	0	23.30	0.67	32	pos
1	89	66	23	94	28.10	0.17	21	neg
0	137	40	35	168	43.10	2.29	33	pos
5	116	74	0	0	25.60	0.20	30	neg
3	78	50	32	88	31.00	0.25	26	pos
10	115	0	0	0	35.30	0.13	29	neg
2	197	70	45	543	30.50	0.16	53	pos
8	125	96	0	0	0.00	0.23	54	pos
4	110	92	0	0	37.60	0.19	30	neg
10	168	74	0	0	38.00	0.54	34	pos
10	139	80	0	0	27.10	1.44	57	neg
1	189	60	23	846	30.10	0.40	59	pos

Build the best classifier for optimal recognition of diabetes!

```
library(mlbench);  data(PimaIndiansDiabetes)
```

Introduction to the basic idea of pattern recognition

1. *Data:* Given a data set $\Big\{(\mathbf{x}_1, \mathrm{y}_1), (\mathbf{x}_2, \mathrm{y}_2), \cdots, (\mathbf{x}_n, \mathrm{y}_n)\Big\}$ where $\mathbf{x}_i = (\mathrm{x}_{i1}, \cdots, \mathrm{x}_{iq})^\top \in \mathbb{R}^q$ and $\mathrm{y}_i \in \{1, 2, \cdots, L\}$. Here, y_i is simply the label of the class.
2. *Goal:* The immediate aspect of the pattern recognition (classification) problem deals with building a classification rule that assigns vectors (representing a collection of characteristics of entities under consideration) to L different categories
3. *Generalization:* Given the data, the goal in PR is to find the *best* classifier among all classifiers f, not just on the present data set, but also for all future entities generated by the same population.

Question: When one says best classifier, it is foundational important to specify the criterion by which the best is determined.

Introduction to the basic idea of pattern recognition

- *Cost function (Risk functional):* *The objective function is therefore the expected loss also known as risk*

$$R(f) = \mathbb{E}[\ell(Y, f(X))] = \int_{\mathcal{X}\times\mathcal{Y}} \ell(\mathrm{y}, f(\mathrm{x}))dP(\mathrm{x}, \mathrm{y})$$

- *Cost function as misclassification rate:* *It is interesting to see that the objective function (risk functional) is simply the probability of misclassification rate*

$$R(f) = \mathbb{E}[\ell(Y, f(X))] = \Pr[Y \neq f(X)]$$

Note: It turns out that in practice, the above risk functional cannot be obtained in closed-form, because clearly the joint cdf $P(\mathrm{x}, \mathrm{y})$ is not known. If it were, all would be easy.

Introduction to the basic idea of pattern recognition

- *Cost function (Risk functional):* *The risk functional* $R(f) = \mathbb{E}[\ell(Y, f(X))] = \Pr[Y \neq f(X)]$ *confirms our intuition because it is estimated in practice by simply computing the proportion of misclassified entities. We are basically saying that*
- *The universally best classifier:*

  the best classifier f^* *is the one that minimizes the rate of misclassifications.*

$$f^* = \arg\min_f \mathbb{E}[\ell(Y, f(X))] = \arg\min_f \Pr[Y \neq f(X)]$$

- *Note of generalization:*

  The minimization must be achieved on the whole population of entities to be classified, not just the ones found in some random sample from that population.

Introduction to the basic idea of pattern recognition

- *Approximation:* *Since searching all possible functions in the universe in order to find the one that best explains our data is clearly a daunting task, it is usually the case in ML and Data mining to approximate, i.e. choose a particular class of function.*
- *Linear classifiers:* *Search for the best among all linear classifiers.*

$$f^+ = \arg\min_{f \in \mathcal{L}} \mathbb{E}[\ell(Y, f(X))] = \arg\min_{f \in \mathcal{L}} \Pr[Y \neq f(X)]$$

- *Arbitrary set $\mathcal{H}$ of classifiers:* *Search for the best among all the classifiers from set $\mathcal{H}$.*

$$f^+ = \arg\min_{f \in \mathcal{H}} \mathbb{E}[\ell(Y, f(X))] = \arg\min_{f \in \mathcal{H}} \Pr[Y \neq f(X)]$$

The set $\mathcal{H}$ above could be finite or infinite. All the algorithms and methods of classification studied in this course search such a class as most start by assuming a certain form for the classifier.

Intuitive Motivation Discriminant Analysis

A reasonable classification rule: If $\boldsymbol{x}$ is the vector to be classified and f is the classifier, then a reasonable definition of f can be formulated as follows

$$f(\boldsymbol{x}) = \underset{l \in \{1,\cdots,L\}}{\texttt{argmax}} \{\delta_l(\boldsymbol{x})\}$$

where $\delta_l(\boldsymbol{x})$ is a function heretofore knows as the discriminant function for class l. The decision boundary between classes l and class j is the set

$$\{\boldsymbol{x} \in \mathbb{R}^q \ : \ \delta_l(\boldsymbol{x}) = \delta_j(\boldsymbol{x})\}, \qquad \forall\, j \neq l.$$

Given a training set, the classifier f can be estimated using $\hat{f}$, the class of $\boldsymbol{x}$ is estimated by

$$\hat{f}(\boldsymbol{x}) = \underset{l \in \{1,\cdots,L\}}{\texttt{argmax}} \{\hat{\delta}_l(\boldsymbol{x})\}$$

Note: This definition of a classifier is indeed reasonable. However, it does not say anything about the loss function that triggered it. In other words, we still have to find out what the estimated expected loss (risk) is for this classifier.

Bayes Classifier

- *Prior membership probability:* *The probability that a randomly selected entity comes from class l is*

$$\pi_l = \Pr[Y = l]$$

- *Class conditional density:* *The density of entities from group l is*

$$p(\boldsymbol{x}|y = l)$$

- *Posterior membership probability:* *Thanks to Bayes' rule,*

$$\Pr[Y = l|\boldsymbol{x}] = \frac{\pi_l p(\boldsymbol{x}|y = l)}{p(\boldsymbol{x})}$$

If one can compute the above posterior probability, then an excellent discriminant function is

$$\delta_l(\boldsymbol{x}) = \Pr[Y = l|\boldsymbol{x}]$$

i.e assign $\boldsymbol{x}$ to the class with the highest posterior probability.

Bayes Classifier is the Best Classifier

- *Let the misclassification rate of a classifier f be defined as*

$$R(f) = \Pr[Y \neq f(X)]$$

- *Consider binary classification, and let f^* be such that*

$$f^*(\boldsymbol{x}) = \begin{cases} 1 & \Pr[Y = 1|\boldsymbol{x}] > \frac{1}{2} \\ 0 & \texttt{Otherwise} \end{cases}$$

- *Then $f^\star$ is the Bayes classifier, and for any classifier f,*

$$R^* = R(f^*) < R(f)$$

No classifier exists that can do better that the Bayes' classifier

- *For a general multi-class setting, f^* is simply written*

$$f^*(\boldsymbol{x}) = \underset{y \in \mathcal{Y}}{\texttt{argmax}} \{\Pr[Y = y|\boldsymbol{x}]\}$$

Elements of Classifier Construction

Gaussian Bayes Classifier

- *The greatest challenge with Bayes classification lies in the fact that one usually never knows the class densities and therefore the posterior cannot be exactly computed*
- *Given a data set* $\{(\boldsymbol{x}_i, y_i)\}$*, estimating the class conditional densities* $p(\boldsymbol{x}|y=l)$ *is arguably one of the hardest tasks in statistical science, especially when* $\boldsymbol{x} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_q)^\top$ *with a large* q *very large.*
- *Density estimation in high dimensional spaces suffers from the curse of dimensionality.*
- *If* $p(\boldsymbol{x}|y)$ *is normal, ie the data in each group follow a multivariate Gaussian distribution with* $\boldsymbol{\mu}_l$ *and* $\boldsymbol{\Sigma}_l$*, i.e.*

$$p(\boldsymbol{x}|y=l) = \frac{1}{(2\pi)^{q/2}|\boldsymbol{\Sigma}_l|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_l)^\top \boldsymbol{\Sigma}_l^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_l)\right\} \quad (1)$$

then an estimate of the Bayes' classifier can be obtained

Gaussian Bayes Classifier - Linear Discriminant

Linear Discriminant Analysis (LDA)

Under the Gaussian Class Conditional

Gaussian Bayes Classifier - Linear Discriminant

If the two class-conditional densities are Gaussian with equal covariance matrices, then the Bayes rule simplifies to

$$\begin{aligned} f^*(\mathbf{x}) &= \underset{l\in\{0,1\}}{\mathtt{argmax}}\Big\{\delta_l(\mathbf{x})\Big\} = I\left(\Pr[Y=1|\boldsymbol{x}] > \Pr[Y=0|\boldsymbol{x}]\right) \\ &= I\left(\log\frac{\Pr[Y=1|\boldsymbol{x}]}{\Pr[Y=0|\boldsymbol{x}]} > 0\right) \\ &= I\left(\beta_0 + \boldsymbol{\beta}^\top\boldsymbol{x} > 0\right) \end{aligned}$$

as a result of the fact one can write

$$\log\left[\frac{\Pr(Y=1|\boldsymbol{x})}{\Pr(Y=0|\boldsymbol{x})}\right] = \beta_0 + \boldsymbol{\beta}^\top\boldsymbol{x}$$

where

$$\boldsymbol{\beta} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

and

$$\beta_0 = -\frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)^\top\Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) + \log\frac{\pi_1}{\pi_0}$$

Gaussian Bayes Classifier - Linear Discriminant

Linear Discriminant Analysis

It is often the case in practice that the covariance matrices are reasonably close. Hence the common assumption $\boldsymbol{\Sigma}_0 = \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}$. *In such a case, the discriminant function becomes*

$$\delta_l(\boldsymbol{x}) = \boldsymbol{b}_l^{\mathsf{T}} \boldsymbol{x} + c_l \tag{2}$$

where

$$\boldsymbol{b}_l = \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_l \tag{3}$$

and

$$c_l = -\frac{1}{2} \boldsymbol{\mu}_l^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_l + \log \pi_l \tag{4}$$

It is clear that the discriminant function $\delta_l(\boldsymbol{x})$ *(and hence the decision boundary* $\{\boldsymbol{x} : \delta_0(\boldsymbol{x}) = \delta_1(\boldsymbol{x})\}$*) is inherently linear. As a result, this procedure is called* `linear discriminant analysis (LDA)`.

Gaussian Bayes Classifier - Linear Discriminant

Theoretical Fact on Linear Discriminant Analysis

Bayes Risk in Binary Classification under Gaussian Class Conditional Densities with common covariance matrix: Let $\mathbf{x} = (\mathrm{x}_1, \mathrm{x}_2, \cdots, \mathrm{x}_p)^\top$ *be a* p*-dimensional vector coming from either class* 1 *or class* 2*. Let* f *be a function (classifier) that seeks to map* $\mathbf{x}$ *to* $y \in \{1, 2\}$ *as accurately as possible. Let* $R* = \min_f \{\Pr[f(X) \neq Y]\}$ *be the Bayes Risk, i.e. the smallest error rate among all possible* f*. If* $p(\mathbf{x}|y = 1) = \texttt{MVN}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma)$ *and* $p(\mathbf{x}|y = 2) = \texttt{MVN}(\mathbf{x}, \boldsymbol{\mu}_2, \Sigma)$*, then*

$$R^* = \Phi(-\sqrt{\Delta}/2) = \int_{-\infty}^{-\sqrt{\Delta}/2} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2} dz,$$

with

$$\Delta = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2).$$

Theoretical Fact on Linear Discriminant Analysis

If $p(\mathbf{x}|y=1) = \texttt{MVN}(\mathbf{x}, \boldsymbol{\mu}_1, \Sigma)$ *and* $p(\mathbf{x}|y=2) = \texttt{MVN}(\mathbf{x}, \boldsymbol{\mu}_2, \Sigma)$*, the Bayes classifier* f^**, the classifier that achieves the Bayes risk, coincides with the population Linear Discriminant Analysis (LDA),* $f_{\texttt{LDA}}$*, which, for any new point* $\mathbf{x}$*, yields the predicted class*

$$f^*(\mathbf{x}) = f_{\texttt{LDA}}(\mathbf{x}) = 2 - I_{\left\{\beta_0 + \boldsymbol{\beta}^\top \mathbf{x} > 0\right\}},$$

where

$$\boldsymbol{\beta} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2),$$

and

$$\beta_0 = -\frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)^\top \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) + \log\frac{\pi_1}{\pi_2},$$

with $\pi_1 = \Pr[Y=1]$ *and* $\pi_2 = 1 - \pi_1$ *representing the prior probabilities of class membership.*

Let $\mathcal{L}$ represent the function class of binary classifiers in q-dimension, ie

$$\mathcal{L} = \Big\{ f : \mathcal{X} \to \mathcal{Y} s.t \exists \boldsymbol{w} \in \mathbb{R}^q,\ \mathrm{w}_0 \in \mathbb{R},\ f(\mathbf{x}) = \mathtt{sign}(\boldsymbol{w}^\top \mathbf{x} + \mathrm{w}_0), \forall \mathbf{x} \in \mathcal{X} \Big\},$$

then $\mathtt{VCDim}(\mathcal{L}) = h = q + 1$.

With labels taken from $\{-1, +1\}$, and using the $0/1$ loss function, we have the fundamental theorem from Vapnik and Chervonenkis, namely,

For every $f \in \mathcal{L}$, and $n > h$, with probability at least $1 - \eta$, we have

$$R(f) \leq \widehat{R}_{emp}(f) + \sqrt{\frac{h\left(\log \frac{2n}{h} + 1\right) + \log\left(\frac{4}{\eta}\right)}{n}}$$

The above result holds true for LDA

Fisher Discriminant Analysis

Brilliant Binary Classification Idea

Introductory Idea of Fisher Discriminant Analysis

Basic Idea of Fisher Discriminant Analysis: *Given a vector* $\boldsymbol{x} = (\mathrm{x}_1, \mathrm{x}_2, \cdots, \mathrm{x}_q)^\top$ *whose label* $\texttt{class}(\boldsymbol{x}) \in \{0, 1\}$ *is to be determined, Fisher's approach to classification consists of two steps:*

- *Project the* p*-dimensional input vector* $\boldsymbol{x} = (\mathrm{x}_1, \mathrm{x}_2, \cdots, \mathrm{x}_q)^\top$ *onto one dimension, which corresponds to creating*

$$\mathrm{z} = \boldsymbol{w}^\top \boldsymbol{x} \tag{5}$$

  for some $\boldsymbol{w}^\top = (\mathrm{w}_1, \mathrm{w}_2, \cdots, \mathrm{w}_q)$.

- *Perform the classification using* z *instead of* $\boldsymbol{x}$.

Questions:

1. *How does one define the projection vector* $\boldsymbol{w}$*?*
2. *What does achieves through such a projection?*

Introductory Idea of Fisher Discriminant Analysis

- *Goal: Achieving good classification means choosing the vector* $\boldsymbol{w}^\top = (\mathrm{w}_1, \mathrm{w}_2, \cdots, \mathrm{w}_q)$ *that best separates the data, in the sense of forming groups characterized by means that are far apart relative to their spread.*
- *Note : Like the LDA described earlier, FLD will assume that* $\boldsymbol{\Sigma}_0 = \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}$. *As a result, the mean of the projection will be conditioned on the class, whereas the its variance will be the same across the classes.*
- *Fact: With the projection defined in* (5)*, it is easy to see that*

$$\mathbb{E}(Z|Y = l) = \mathbb{E}(\boldsymbol{w}^\mathsf{T} X|Y = l) = \boldsymbol{w}^\mathsf{T}\boldsymbol{\mu}_l$$

and

$$\mathbb{V}(Z) = \mathbb{V}(\boldsymbol{w}^\top X) = \boldsymbol{w}^\mathsf{T}\boldsymbol{\Sigma}\boldsymbol{w}$$

Introductory Idea of Fisher Discriminant Analysis

Definition: *Let the following quantity*

$$J(\boldsymbol{w}) = \frac{(\boldsymbol{w}^\top \boldsymbol{\mu}_0 - \boldsymbol{w}^\top \boldsymbol{\mu}_1)^2}{\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}} = \frac{\boldsymbol{w}^\top (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^\top \boldsymbol{w}}{\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}}$$

be the measure of separation, and define

$$\hat{J}(\boldsymbol{w}) = \frac{\boldsymbol{w}^\top S_{\text{B}} \boldsymbol{w}}{\boldsymbol{w}^\top S_{\text{W}} \boldsymbol{w}}$$

where $S_{\text{B}} = (\bar{\boldsymbol{x}}_0 - \bar{\boldsymbol{x}}_1)(\bar{\boldsymbol{x}}_0 - \bar{\boldsymbol{x}}_1)^\top$ *and* $S_{\text{W}} = \frac{(n_0 - 1)S_0 + (n_1 - 1)S_1}{(n_0 - 1) + (n_1 - 1)}$

- S_{B} *is called the* between-class *scatter matrix.* S_{B} *is symmetric and positive semi-definite, but because it is the outer product of two vectors, its rank is at most one.*
- S_{W} *is called the* within-class *scatter matrix. It is proportional to the pooled sample covariance matrix of the feature vectors* $\mathbf{x}$*.* S_{W} *is symmetric and positive semi-definite, and is nonsingular if* $n > p$*.*

Introductory Idea of Fisher Discriminant Analysis

- *Theorem:* *The vector*

$$\boldsymbol{w} = S_{\mathrm{W}}^{-1}(\bar{\boldsymbol{x}}_0 - \bar{\boldsymbol{x}}_1) \tag{6}$$

is a minimizer of $\hat{J}(\boldsymbol{w})$.

- *Definition:* *The function defined by*

$$g(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} = (\bar{\boldsymbol{x}}_0 - \bar{\boldsymbol{x}}_1)^\top S_{\mathrm{W}}^{-1} \boldsymbol{x} \tag{7}$$

is the Fisher linear discriminant function.

- *Fisher's classification rule is given by*

$$f(\boldsymbol{x}) = \begin{cases} 0 & \text{if } g(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} \geq w_0 \\ 1 & \text{otherwise} \end{cases} \tag{8}$$

where

$$w_0 = \frac{1}{2}(\bar{\boldsymbol{x}}_0 + \bar{\boldsymbol{x}}_1)^\top S_{\mathrm{B}}^{-1}(\bar{\boldsymbol{x}}_0 - \bar{\boldsymbol{x}}_1) \tag{9}$$

Fact: *FLD corresponds to LDA with* $\pi_1 = \pi_2 = \pi = \frac{1}{2}$.

Kernel Fisher Discriminant

- *The linearity of the above Fisher Linear Discriminant Analysis approach is indeed a limitation, in the sense that the approach would fail if the structure of the data is not linear, i.e separable on the basis of the means of the groups alone.*
- *The so-called Kernel Fisher Discriminant provides an extension of FLD that helps circumvent the above limitation.*
- *Essentially, Kernel Fisher Discriminant Analysis uses kernels to achieve Discriminant Analysis in some feature space $\mathcal{F}$ into which the projected data is linearly separable.*

Kernel Fisher Discriminant

Think of it this way: There is an - implicit - mapping $\Phi : \mathcal{X} \to \mathcal{F}$ *such that each vector* $\boldsymbol{x} \in \mathcal{X}$ *is replaced by its corresponding feature* $\Phi(\boldsymbol{x}) \in \mathcal{F}$*, and the needed operations are carried out in the new space* $\mathcal{F}$*. The good news is that this projection is transparent to you, thanks to a kernel* $\mathcal{K}(\cdot, \cdot)$ *such that*

$$\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i)^\top \Phi(\boldsymbol{x}_j) = \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j) \rangle$$

with $\boldsymbol{x} \mapsto \Phi(\boldsymbol{x})$ *and there exists* $\boldsymbol{w} \in \mathcal{F}$ *such that*

$$\boldsymbol{w}^\top \Phi(\boldsymbol{x}) = \sum_{j=1}^{n} \alpha_j \mathcal{K}(\boldsymbol{x}_j, \boldsymbol{x}).$$

`Step 1:` *Perform Kernel PCA on the data and extract the scores.*

`Step 2:` *Perform linear discriminant analysis on the scores.*

Linear Discriminant Analysis (LDA)
Practical Computational Implementation

Linear Discriminant Analysis

Parameter Estimation for Linear Discriminant Analysis

1. *Sample mean vectors*

$$\hat{\boldsymbol{\mu}}_j = \frac{1}{n_j} \sum_{i:y_i=j}^{n} \boldsymbol{x}_i$$

2. *The sample covariance matrices*

$$S_j = \frac{1}{n_j - 1} \sum_{i:Y_i=j}^{n} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)^\top$$

3. *The pooled covariance*

$$\hat{\Sigma} = \frac{(n_0 - 1)S_0 + (n_1 - 1)S_1}{n - 2} = \frac{1}{n - 1} \sum_{j=0}^{1} \sum_{i:Y_i=j}^{n} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)^\top$$

4. *Prior probabilities of class membership*

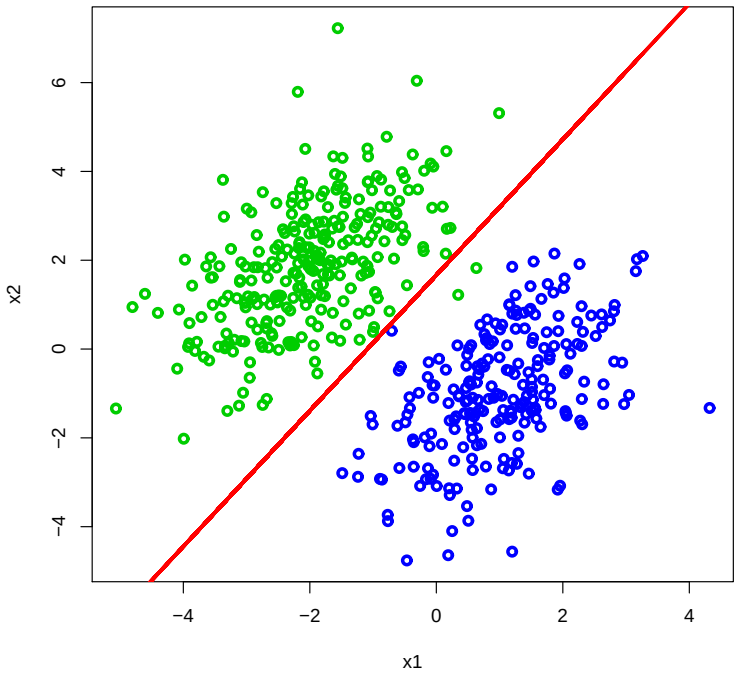
$$\hat{\pi}_j = \frac{n_j}{n} = \frac{1}{n} \sum_{i=1}^{n} \mathtt{I}(Y_i = j)$$

Computed Example of Linear Discriminant Analysis in R

- *Consider a 2d artificial problem:* $\boldsymbol{\mu}_1 = (1, -1)^\top$, $\boldsymbol{\mu}_2 = (-2, 2)^\top$ *and*

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1.00 & 0.75 \\ 0.75 & 2.00 \end{bmatrix}$$

- *Generate* $n = 500$ *pairs* $(\mathrm{x}_1, \mathrm{x}_2)^\top$ *using* $\pi_1 = 45/100$.

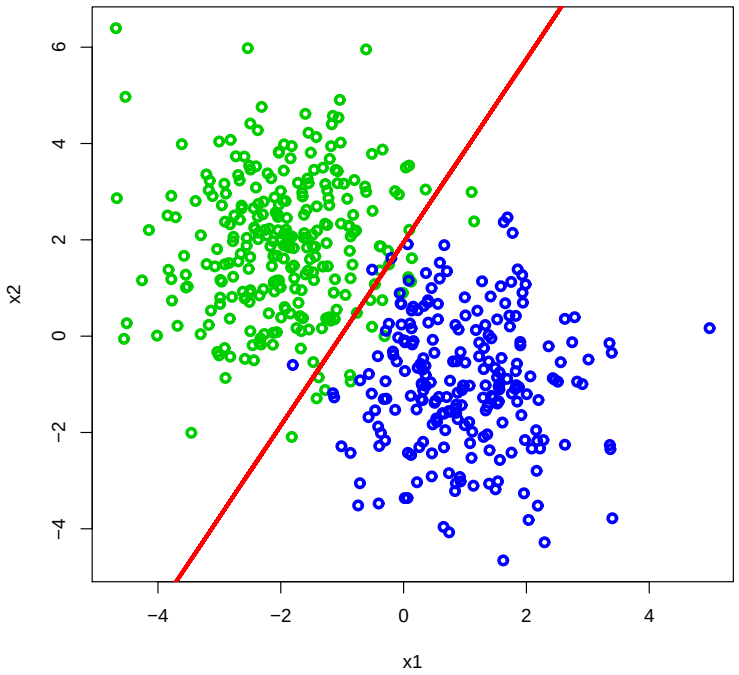


Computed Example of Linear Discriminant Analysis in R

- *Consider a 2d artificial problem:* $\boldsymbol{\mu}_1 = (1, -1)^\top$, $\boldsymbol{\mu}_2 = (-2, 2)^\top$ *and*

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 2.00 \end{bmatrix}$$

- *Generate* $n = 500$ *pairs* $(\mathrm{x}_1, \mathrm{x}_2)^\top$ *using* $\pi_1 = 45/100$.

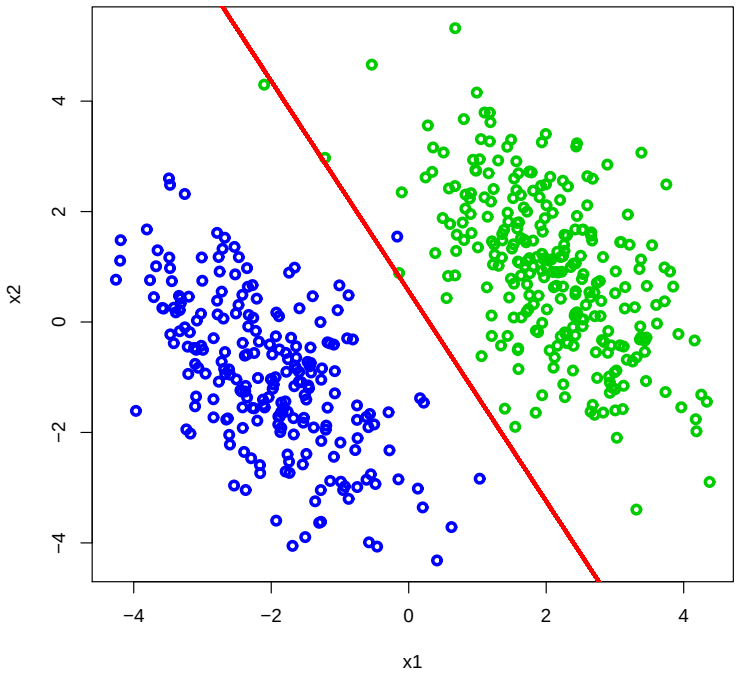


Computed Example of Linear Discriminant Analysis in R

- *Consider a 2d artificial problem:* $\boldsymbol{\mu}_1 = (-2, -1)^\top$, $\boldsymbol{\mu}_2 = (2, 1)^\top$ *and*

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1.00 & -0.85 \\ -0.85 & 2.00 \end{bmatrix}$$

- *Generate* $n = 500$ *pairs* $(\mathrm{x}_1, \mathrm{x}_2)^\top$ *using* $\pi_1 = 45/100$.



Computed Example of Linear Discriminant Analysis in R

```
n   <- 500;      psi <- 45/100
mu1 <- c(-2,-1);  mu2 <- c(2,1); rho <- -0.85
Sigma1 <- matrix(c(1,rho,rho, 2), nrow=2, ncol=2);
Sigma2 <- Sigma1

n1 <- round(psi*n);  n2 <- n - n1

set.seed(10121967)
xy <- cbind(mvrnorm(n2, mu2, Sigma2), rep(0, n2))
xy <- rbind(xy, cbind(mvrnorm(n1, mu1, Sigma1), rep(1, n1))

xy <- data.frame(xy)
colnames(xy) <- c('x1','x2','y')
y <- xy$y
```

Computed Example of Linear Discriminant Analysis in R

Perform lda in R and plot the decision boundary

```
lda.xy <- lda(y~., data=xy)

mu1.hat <- lda.xy$means[1,]
mu2.hat <- lda.xy$means[2,]
invSig <- solve(cov(xy[,-3]))
mup <- t(mu2.hat+mu1.hat)
mum <- mu2.hat-mu1.hat
beta0  <-  -0.5*mup%*%invSig%*%mum + log(n2/n1)
beta   <-  invSig%*%mum
beta1  <-  beta[1]
beta2  <-  beta[2]
x1.db  <-  xy[,1]
x2.db  <- -beta0/beta2 - (beta1/beta2)*x1.db
plot(xy[,-3], col=xy$y+3, lwd=3)
lines(x1.db, x2.db, col='red', lwd=3)
```

Computed Example of for Linear Discriminant in R

- *We can now predict the class of all the observations in the sample*

```
yhat <- predict(lda.xy, xy[,-3])$class
```

- *We can then generate the confusion matrix using*

```
table(y, yhat)
```

The corresponding confusion matrix is given by

		Prediction	
		0	1
Actual	0	275	0
	1	1	224

- *The Percentage of Correct Classification here is*

$$\texttt{pcc}(lda) = 499/500$$

Estimated Decision Boundary for LD Classifier

Given any new point $\boldsymbol{x}$, the LD classifier will estimate its class as

$$\hat{f}_{LDA}(\boldsymbol{x}) = I\left(\hat{\beta}_0 + \hat{\boldsymbol{\beta}}^\top \boldsymbol{x} > 0\right)$$

where

$$\hat{\boldsymbol{\beta}} = \hat{\Sigma}^{-1}(\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_0)$$

and

$$\hat{\beta}_0 = -\frac{1}{2}(\hat{\boldsymbol{\mu}}_1 + \hat{\boldsymbol{\mu}}_0)^\top \hat{\Sigma}^{-1}(\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_0) + \log\frac{\hat{\pi}_1}{\hat{\pi}_0}$$

The decision boundary in this case is simply the set

$$\{\boldsymbol{x} : \quad \hat{\beta}_0 + \hat{\boldsymbol{\beta}}^\top \boldsymbol{x} = 0\}$$

Performing Linear Discriminant Analysis in R

The function lda of the R package MASS does Linear Discriminant Analysis

```
lda(x, grouping, prior = proportions, CV = FALSE)
```

1. *x: (required if no formula is given as the principal argument.) a matrix or data frame or Matrix containing the explanatory variables.*
2. *grouping: (required if no formula principal argument is given.) a factor specifying the class for each observation.*
3. *prior: the prior probabilities of class membership. If unspecified, the class proportions for the training set are used. If present, the probabilities should be specified in the order of the factor levels.*
4. *CV: If true, returns results (classes and posterior probabilities) for leave-one-out cross-validation. Note that if the prior is estimated, the proportions in the whole dataset are used.*

Performing Linear Discriminant Analysis in R

The function lda of the R package MASS does Linear Discriminant Analysis

```
lda(formula, data, subset)
```

1. *formula: A formula of the form groups $\sim$ x1 + x2 + ... That is, the response is the grouping factor and the right hand side specifies the (non-factor) discriminators.*
2. *data: Data frame from which variables specified in formula are preferentially to be taken.*
3. *subset: An index vector specifying the cases to be used in the training sample. (NOTE: If given, this argument must be named.)*

Output

1. *object of class lda with several components*

Performing Linear Discriminant Analysis in R

To perform predictions using the estimated lda model

```
predict(object, newdata,...)
```

1. *object: object of class lda*
2. *newdata: data frame of cases to be classified or, if object has a formula, a data frame with columns of the same names as the variables used. A vector will be interpreted as a row vector. If newdata is missing, an attempt will be made to retrieve the data used to fit the lda object.*

List of components in output object

1. *class: The MAP classification (a factor)*
2. *posterior: posterior probabilities for the classes*
3. *x: the scores of test cases on up to dimen discriminant variables*

Quadratic Discriminant Analysis (QDA)

Under the Gaussian Class Conditional

Quadratic Discriminant Analysis

When it so happens that the covariance matrices are different, i.e. $\boldsymbol{\Sigma}_0 \neq \boldsymbol{\Sigma}_1$, *The discriminant function can be written as*

$$\delta_l(\boldsymbol{x}) = \boldsymbol{x}^\mathsf{T}\mathbf{A}_l\boldsymbol{x} + \boldsymbol{b}_l^\mathsf{T}\boldsymbol{x} + c_l \tag{10}$$

where

$$\mathbf{A}_l = -\frac{1}{2}\boldsymbol{\Sigma}_l^{-1} \qquad \boldsymbol{b}_l = \boldsymbol{\Sigma}_l^{-1}\boldsymbol{\mu}_l \tag{11}$$

and

$$c_l = -\frac{1}{2}\boldsymbol{\mu}_l^\mathsf{T}\boldsymbol{\Sigma}_l^{-1}\boldsymbol{\mu}_l - \frac{1}{2}\log|\boldsymbol{\Sigma}_l| + \log\pi_l \tag{12}$$

It is clear that the discriminant function $\delta_l(\boldsymbol{x})$ *(and hence the decision boundary* $\{\boldsymbol{x} : \delta_0(\boldsymbol{x}) = \delta_1(\boldsymbol{x})\}$*) is inherently quadratic. As a result this procedure is called* `quadratic discriminant analysis (QDA)`.

Challenges and Drawbacks with Bayes Gaussian DA

- *Pervading need to estimate the scatter matrix $\mathbf{\Sigma}$, but even more challenging to estimate the precision matrix $\mathbf{\Sigma}^{-1}$*
  - *In q-dimensional space, the computational complexity of inverting a matrix is*

$$\texttt{complexity}(\mathbf{\Sigma}^{-1}) = O(q^3)$$

    which becomes computationally burdensome when q is large, as it tends to be in many real life problems.
- *The need to estimate the precision matrix $\mathbf{\Sigma}^{-1}$ makes Gaussian discriminant analysis unusable when $q > n$, ie, when the number of variables is larger than the number of observations as in Microarray Gene Expression, Image Classification, just to name a few*
- *In a good numbmer of situations, QDA provides a tremendous natural improvement to LDA. Unfortunately, QDA is statistically far more demanding than LDA, because of the need to estimate L different covariance matrices and their inverses.*

Naive Bayes Classifier

Dreaming of Input Space Independence

The Naive Bayes Classifier - Motivation

When the class conditional densities are not Gaussian, the high dimensionality of the input space makes density estimation a difficult task.

- *Naive Bayes assumes that the q attributes $\mathbf{x}_j$ of $\boldsymbol{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_q)^\top$ are independent, so that the high dimensional density $p(\boldsymbol{x}|y = j)$ can be written as a product of one dimensionality densities*

$$p(\boldsymbol{x}|y = j) = \prod_{\ell=1}^{q} p_\ell(\mathrm{x}_\ell|y = j) = p_1(\mathrm{x}_1|y = j) \cdots p_q(\mathrm{x}_q|y = j)$$

  With the naive Bayes assumption,
  - *The density estimation task is now much easier*
  - *Vectors containing attributes of different can be handled*
- *The estimated discriminant function under naive Bayes is*

$$\hat{\delta}_j(\boldsymbol{x}) = \widehat{\Pr(Y = j)} \prod_{\ell=1}^{q} \hat{p}_\ell(\mathrm{x}_\ell|\mathrm{y} = j)$$

Naive Bayes Classifier

Despite its naive assumption, the Naive Bayes Classifier does enjoy some interesting advantages (strengths):

- *It is still a Bayes Classifier (remember Bayes classifier is the best of all), despite its naivete, remains an approximate solution to the true problem (minimizing) the misclassification rate, it is a least suboptimal one could say.*
- *Because each variable in the observed vector is addressed separately, naive bayes offers the potential to build classifiers for tasks involving variables of different types*
- *Empirical evidence seems to suggest that naive Bayes works very well in very important applications like text mining.*

Of course, exercise caution when relying too heavily on naive Bayes. If predictive optimality is your goal, use the test error to find out if naive Bayes is that bad after all. Question: Where else in Statistics can one get away with treating something as independent when it is not?

Naive Bayes Classifier in R

- *The R package e1071 offers the function naiveBayes that computes the conditional a-posterior probabilities of a categorical class variable given independent predictor variables using the Bayes rule.*
- *Training: It can be called using the formula*

```
naiveBayes(formula, data, subset)
```

or

```
naiveBayes(x, y)
```

- *Prediction/test: An object of class naiveBayes is returned, and prediction can be performed as usual*

```
predict(object, newdata)
```

Multiclass in seamless with DA

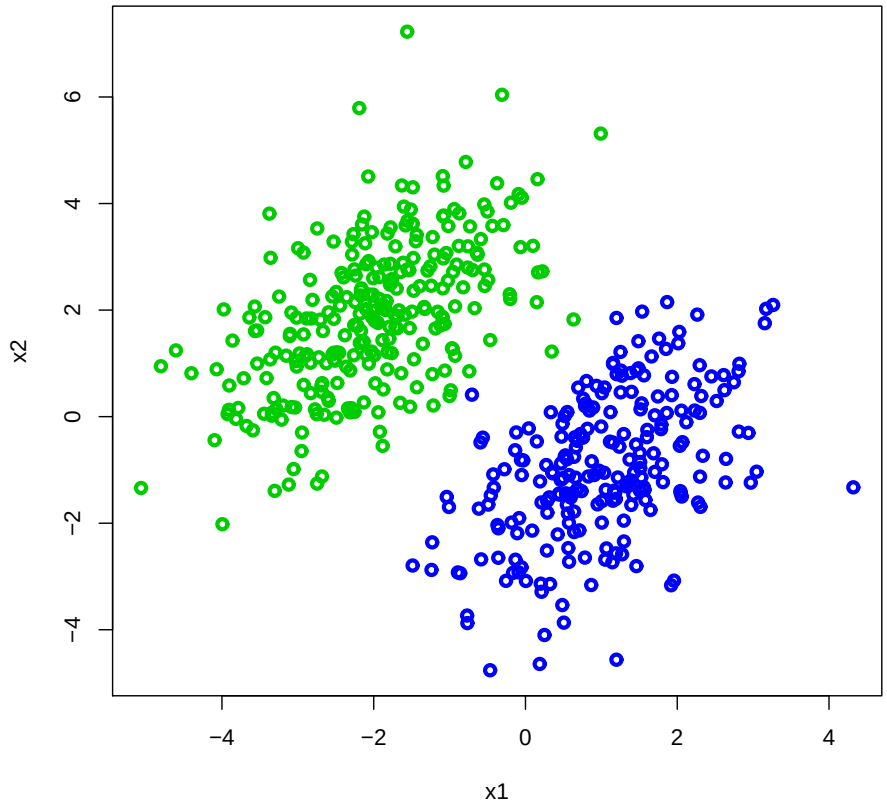
Naive Bayes Classifier in R

The R package e1071 offers the function naiveBayes that computes the conditional a-posterior probabilities of a categorical class variable given independent predictor variables using the Bayes rule. It can be called using the formula naiveBayes(formula, data, subset) or naiveBayes(x, y) An object of class naiveBayes is returned, and prediction can be performed as usual using predict(object, newdata)

1. *x: A numeric matrix, or a data frame of variables.*
2. *y: Class vector.*
3. *formula: A formula of the form class $\sim$ x1 + x2 + ...*
4. *data: Either a data frame of predictors or a contingency table.*
5. *object : An object of class naiveBayes.*
6. *newdata: A dataframe with new predictors (with possibly fewer columns than the training data).*

Motivating Examples of Pattern Recognition

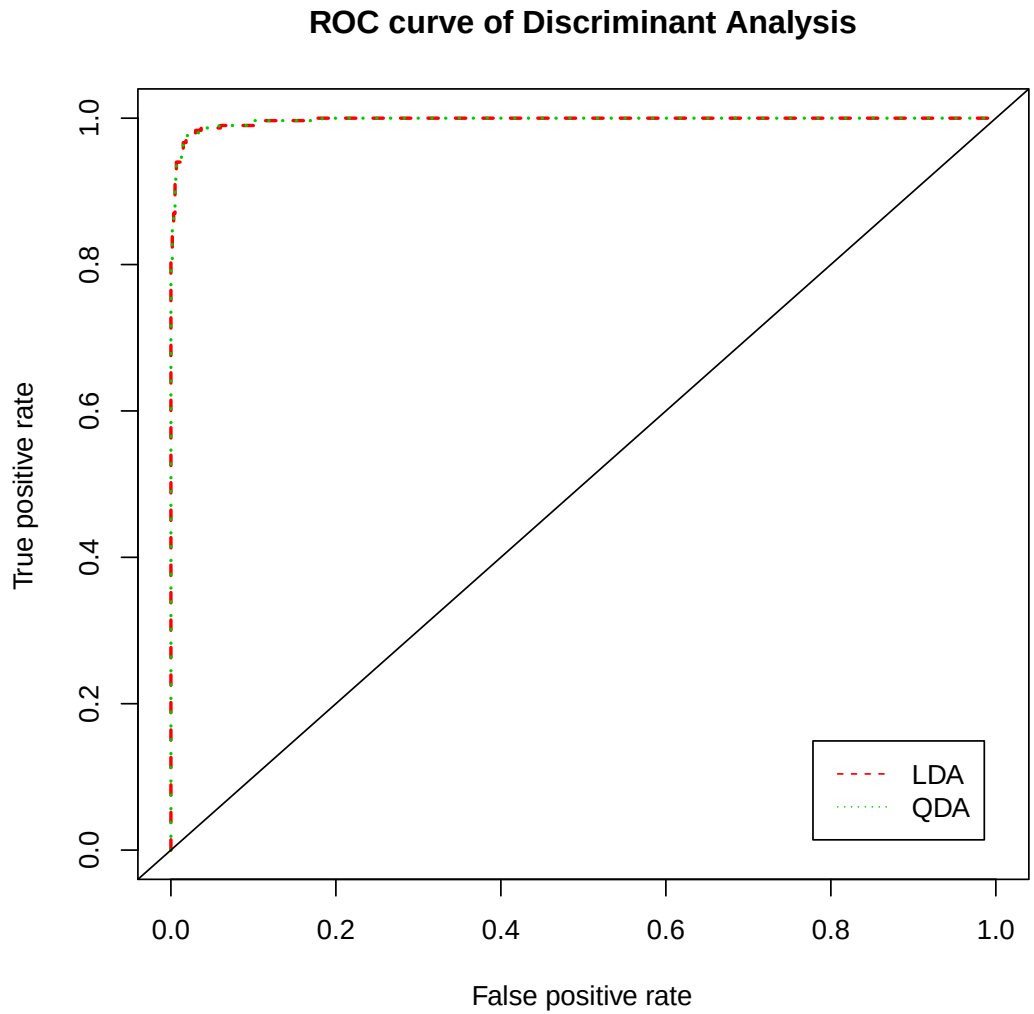
Task 1: *Consider the following two dimensional binary classification task.*



Question: Can the two classes be separated by a line?

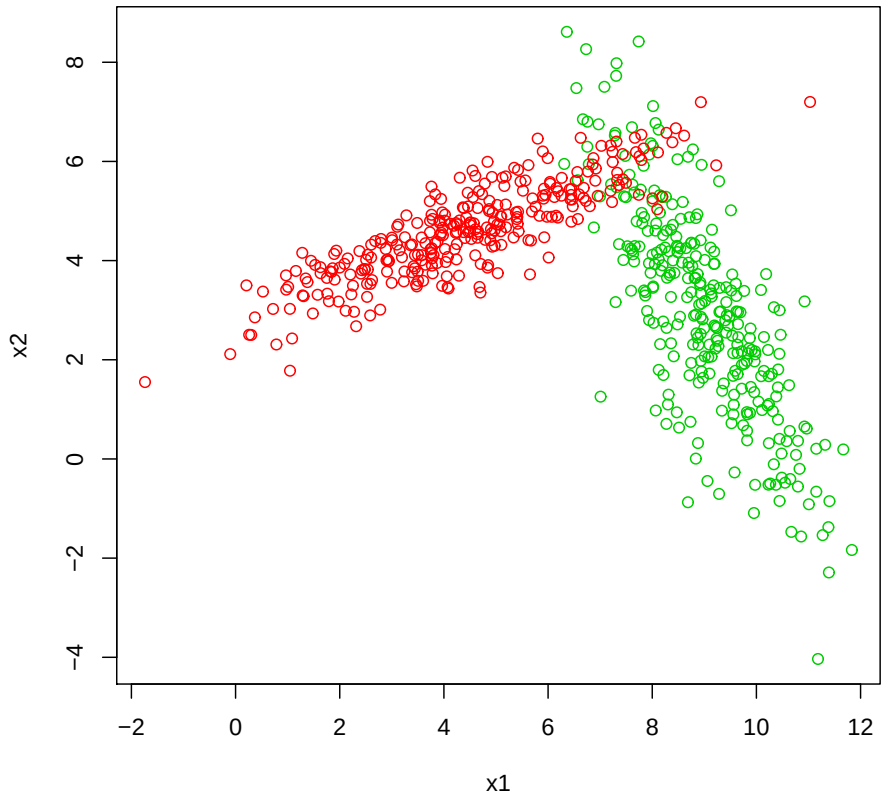
ROC Curve for Task 1

Task 1: Easy task for both LDA and QDA.



Motivating Examples of Pattern Recognition

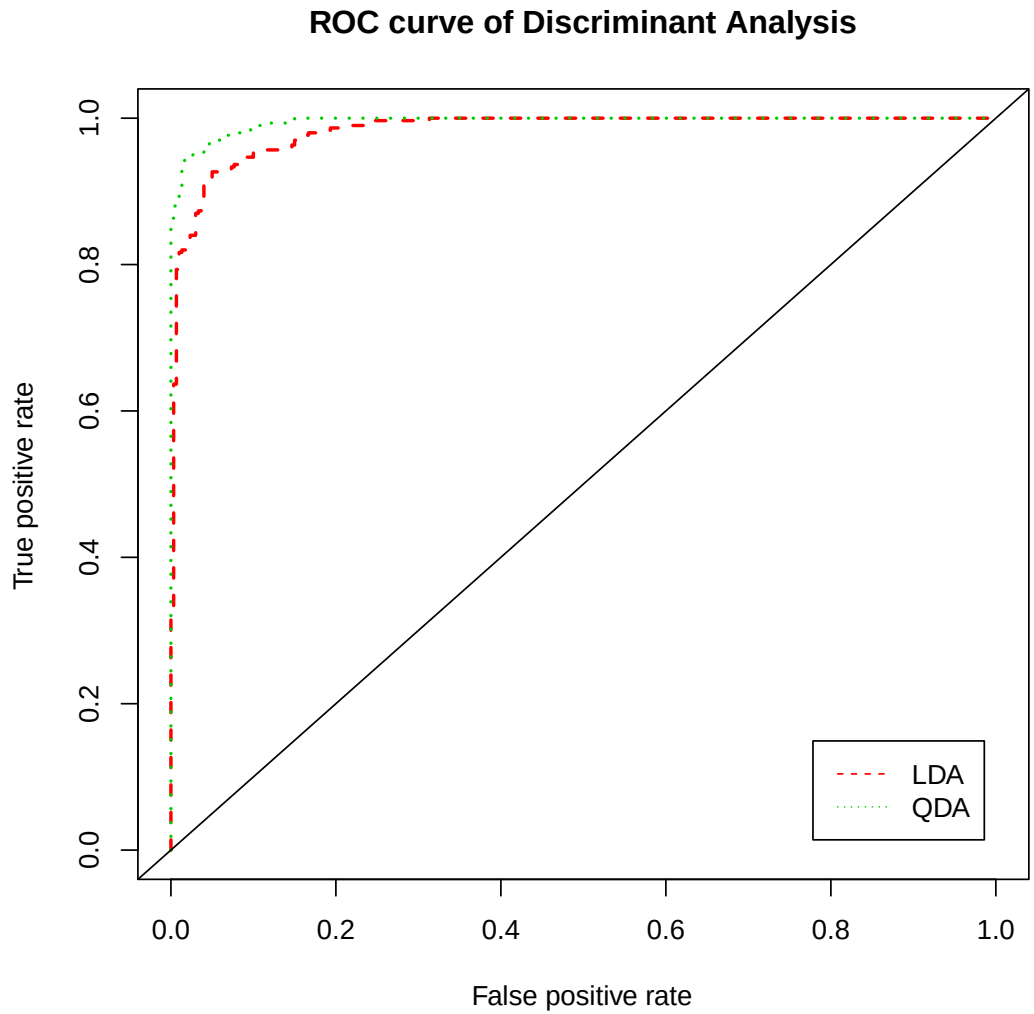
Task 2: Consider the following two dimensional binary classification task.



Question: How well can the two classes be separated by a line?

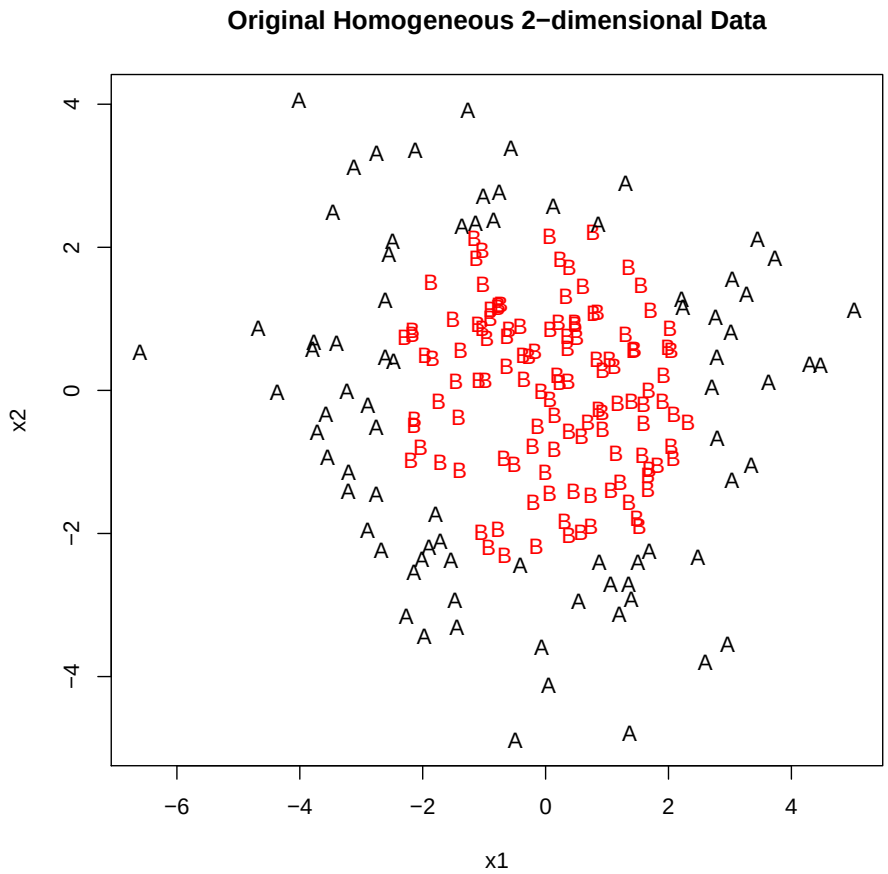
ROC Curve for Task 2

Task 2: Easy task for QDA, but a little hard for LDA.



Motivating Examples of Pattern Recognition

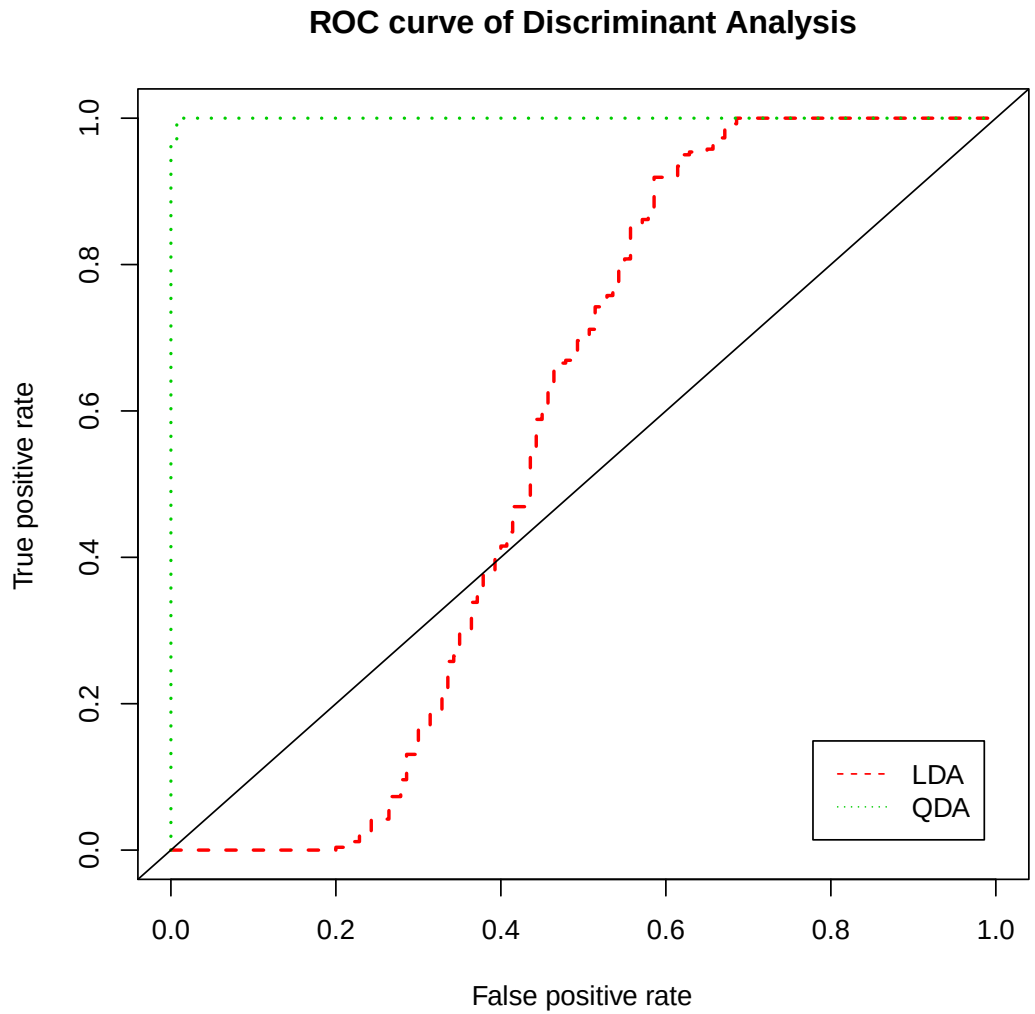
Task 3: Consider the following two dimensional binary classification task.



Question: How well can the two classes be separated by a line?

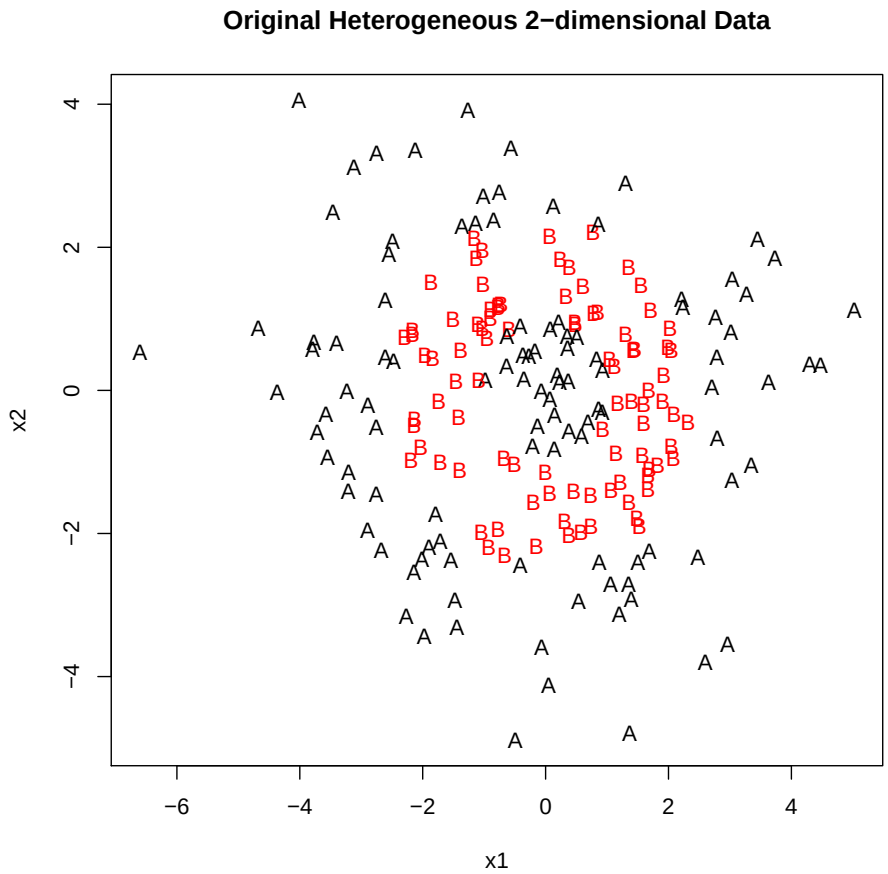
ROC Curve for Task 3

Task 3: Relatively easy for QDA, but really hard for LDA.



Motivating Examples of Pattern Recognition

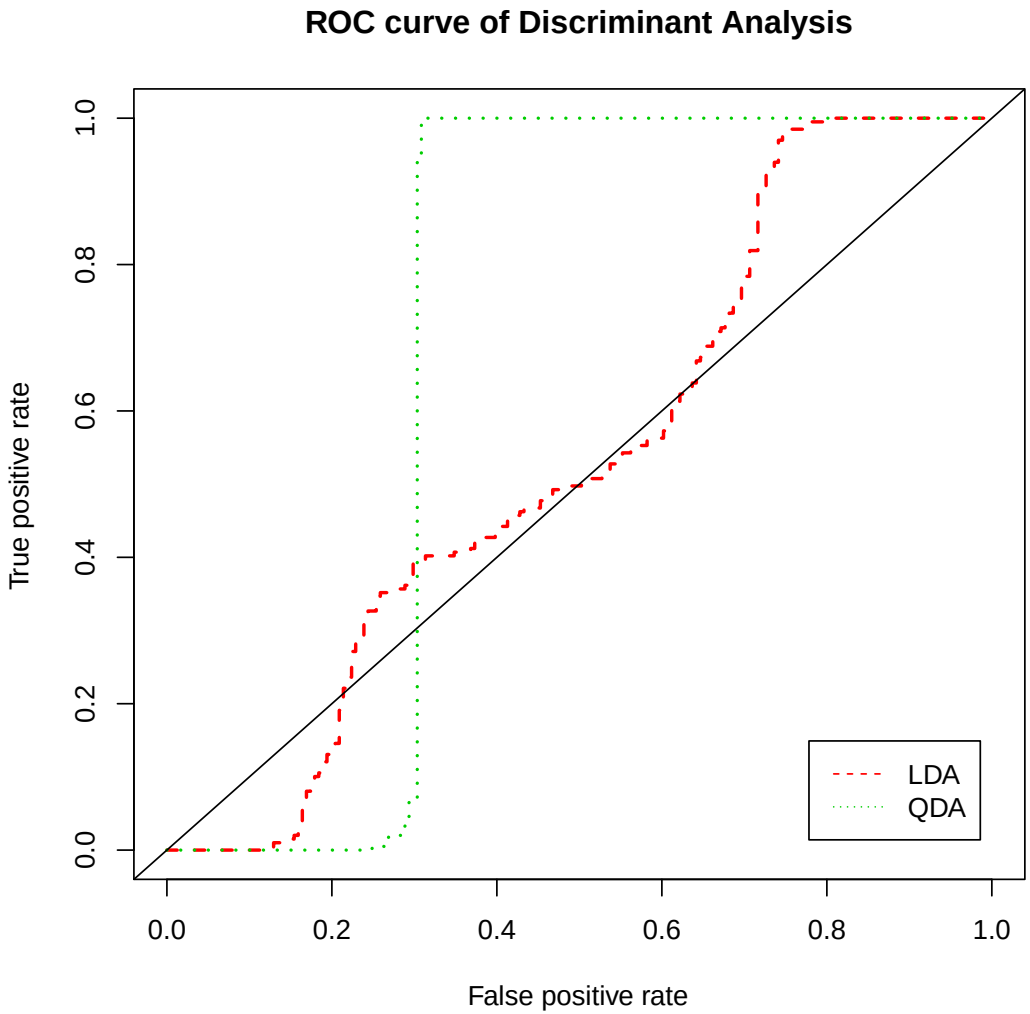
Task 4: Consider the following two dimensional binary classification task.



Question: Can the two classes ever be separated by a line?

ROC Curve for Task 4

Task 4: Very Hard for QDA, also really Hard for LDA.



Exercises for Exploration

Explore the following classification task using the software of your choice

1. *Perform a complete discriminant analysis on the Breast Cancer data set. You should assess the plausibility of normality prior to doing the PR tasks*
2. *Generate the confusion matrix of both LDA and QDA for Pima Indian dataset*
3. *Find a dataset on multiclass classification at UC Irvine data mining and machine learning repository and compare the difference in predictive accuracy between LDA and QDA*

Computational Comparisons

- *Ideally, we would like to compare the true theoretical performances measured by the risk functional*

$$\mathcal{R}(f) = \mathbb{E}[\ell(Y, f(X))] = \int_{\mathcal{X}\times\mathcal{Y}} \ell(\mathbf{x}, \mathrm{y}) dP(\mathbf{x}, \mathrm{y}), \tag{13}$$

- *Instead, we build the estimators using other optimality criteria, and then compare their predictive performances using the average test error* $\texttt{AVTE}(\cdot)$*, namely*

$$\texttt{AVTE}(\widehat{f}) = \frac{1}{R}\sum_{r=1}^{R}\left\{\frac{1}{m}\sum_{t=1}^{m}\ell(\mathrm{y}_{i_t}^{(r)}, \widehat{f}_r(\mathbf{x}_{i_t}^{(r)}))\right\}, \tag{14}$$

where $\widehat{f}_r(\cdot)$ *is the* r*-th realization of the estimator* $\widehat{f}(\cdot)$ *built using the training portion of the split of* $\mathcal{D}$ *into training set and test set, and* $\left(\mathbf{x}_{i_t}^{(r)}, \mathrm{y}_{i_t}^{(r)}\right)$ *is the* t*-th observation from the test set at the* r*-th random replication of the split of* $\mathcal{D}$*.*

Empirical Framework for Comparison of Various Learners

- For $r = 1$ *to* R
  - Draw l items without replacement from $\mathcal{D}$ to form $\mathcal{T}_r$
  - Train $\hat{f}^{(r)}(\cdot)$ based on the l items in $\mathcal{T}_r$
  - Predict $\hat{f}^{(r)}(\mathbf{x}_i)$ for the m items in $\mathcal{V}_r = \mathcal{D} \backslash \mathcal{T}_r$
  - Calculate $\widehat{\texttt{EPMSE}}(\hat{f}^{(r)}) = \frac{1}{m} \sum_{\mathbf{z}_i \in \mathcal{V}_r} \ell(\mathrm{y}_i, \hat{f}^{(r)}(\mathbf{x}_i))$
- End
- *Compute the average EPMSE for* $\hat{f}$*, namely*

$$\texttt{average}\{\texttt{EPMSE}(\hat{f})\} = \frac{1}{R} \sum_{r=1}^{R} \widehat{\texttt{EPMSE}}(\hat{f}^{(r)}).$$

With labels taken from $\{-1, +1\}$, *we have*

$$\hat{R}_{emp}(f) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}|\mathrm{y}_i - f(\mathbf{x}_i)| = \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}_{\{\mathrm{y}_i \neq f(\mathbf{x}_i)\}}$$

For every $f \in \mathcal{F}$, *and* $n > h$, *with probability at least* $1 - \eta$, *we have*

$$R(f) \leq \hat{R}_{emp}(f) + \sqrt{\frac{h\log\left(\frac{2n}{h}+1\right) - \log\left(\frac{4}{\eta}\right)}{n}}$$

In the above formula, h *is the VC (Vapnik-Chervonenkis) dimension of the space* $\mathcal{F}$ *of functions from which* f *is taken.*

Comparison of Learning Methods

	LDA	SVM	CART	rForest	GaussPR
`Musk`	*0.2227*	*0.1184*	*0.2450*	*0.1152*	*0.1511*
`Pima`	*0.2193*	*0.2362*	*0.2507*	*0.2304*	*0.2304*
`Crabs`	*0.0452*	*0.0677*	*0.1970*	*0.1097*	*0.0702*

	kNN	adaBoost	NeuralNet	Logistic
`Musk`	*0.1922*	*0.1375*	*0.1479*	*0.2408*
`Pima`	*0.3094*	*0.2243*	*0.2570*	*0.2186*
`Crabs`	*0.0938*	*0.1208*	*0.0350*	*0.0363*

Table: Computations made with *The Mighty R*

Comparison of Learning Methods

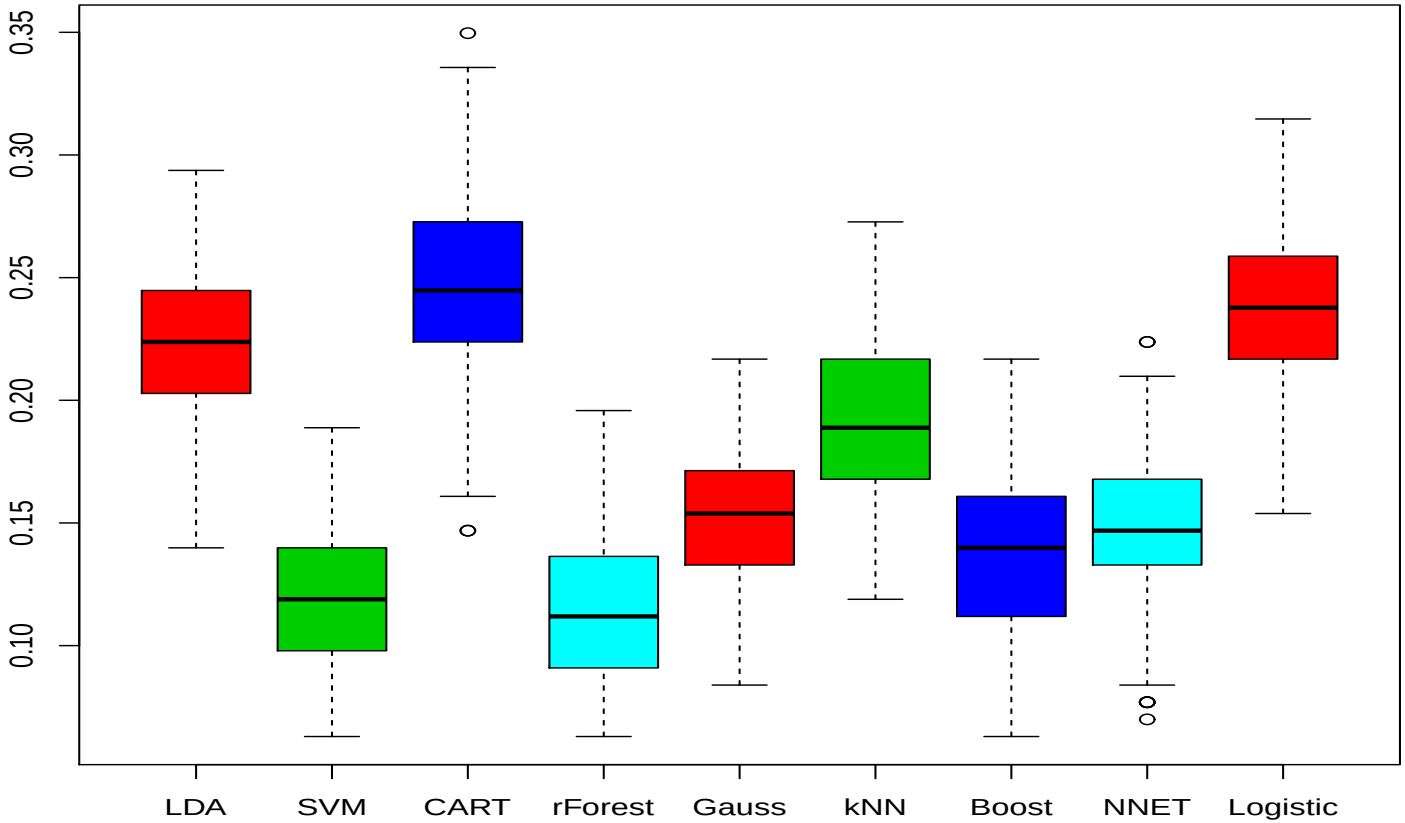


Figure: *Comparison of the average prediction error over* $R = 100$ *replications on the Musk data*

Comparison of Learning Methods

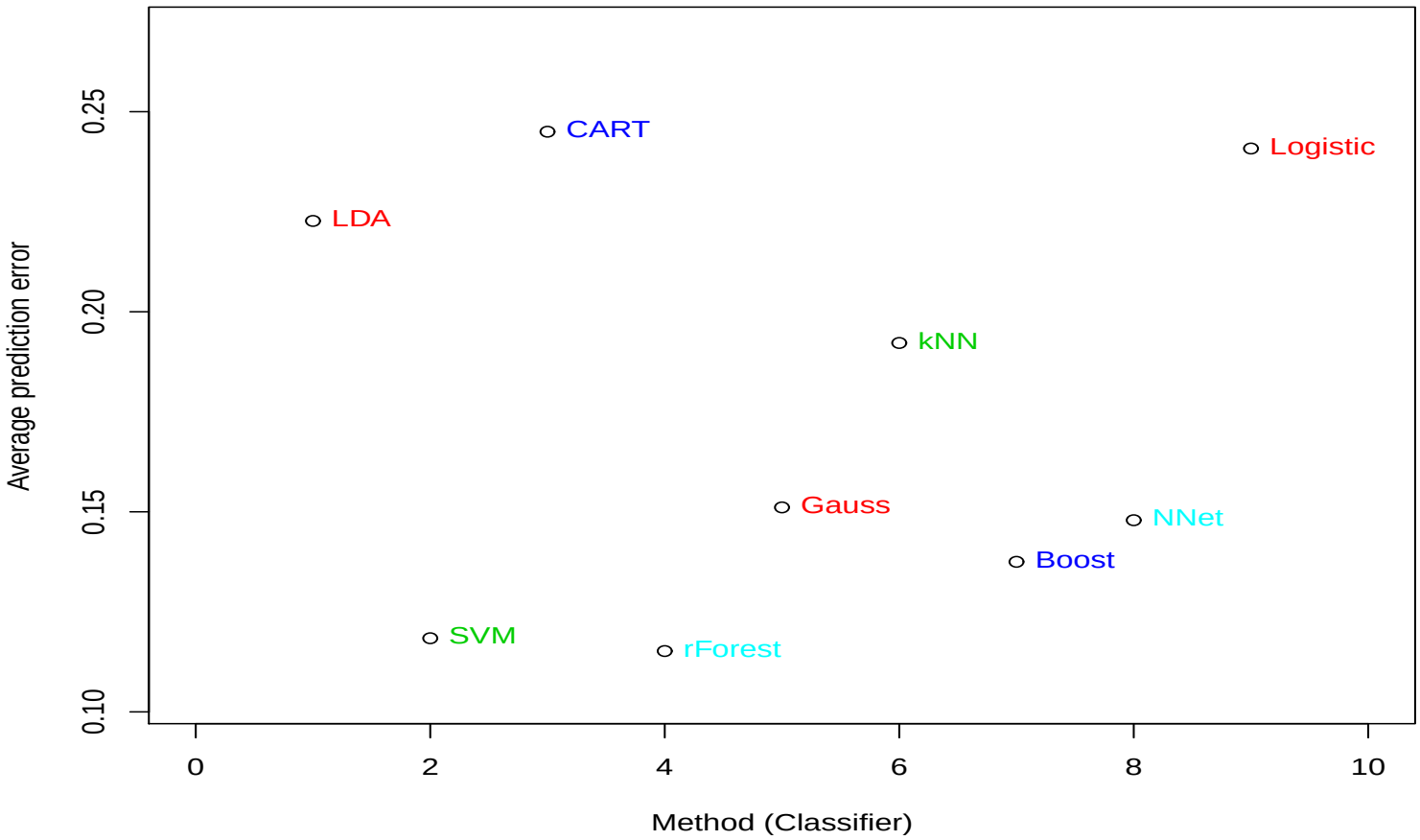


Figure: *Comparison of the average prediction error over* $R = 100$ *replications on the Musk data*

Comparison of Learning Methods

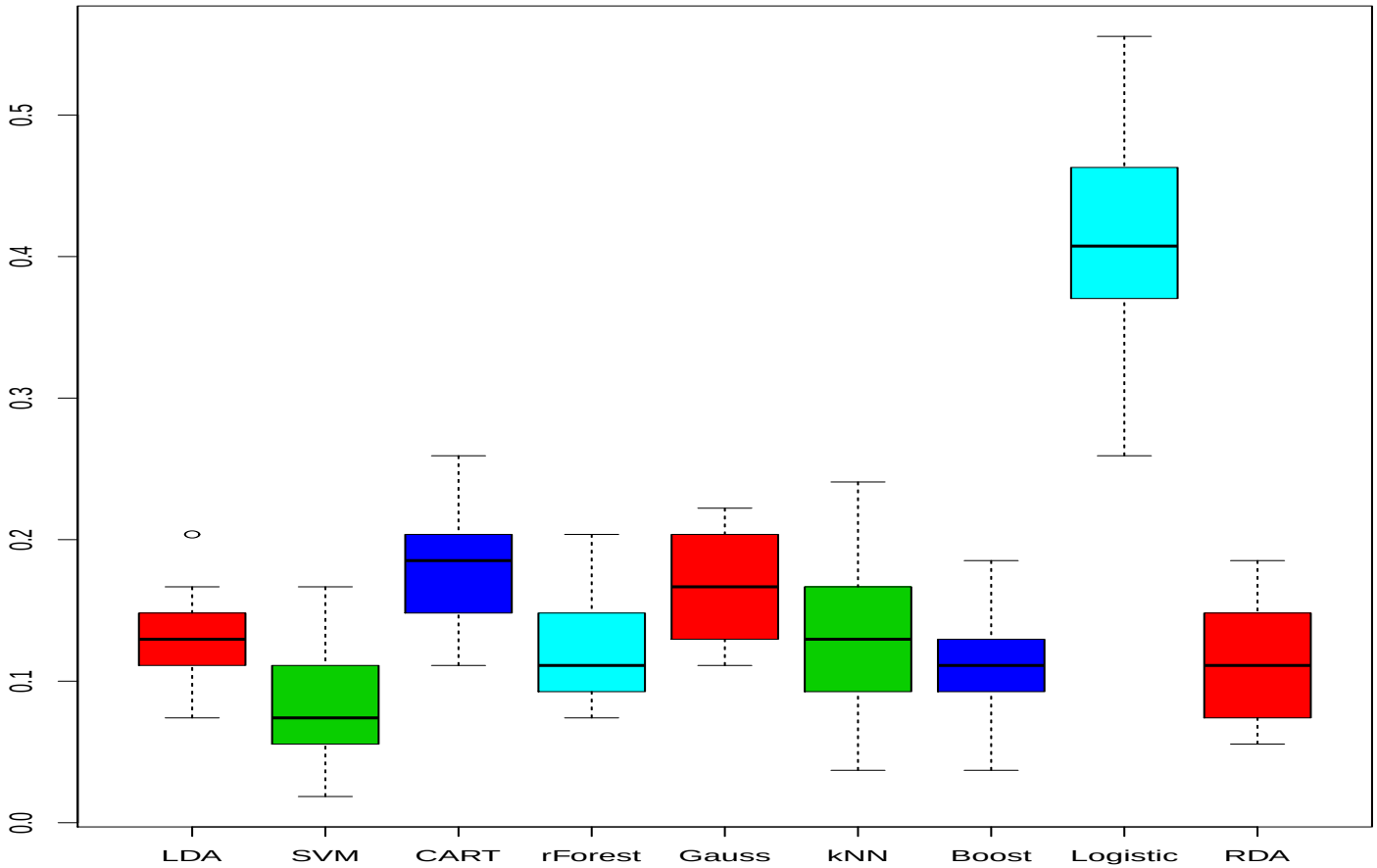


Figure: *Comparison of the average prediction error over* $R = 100$ *replications on the lymphoma data*

Comparison of Learning Methods

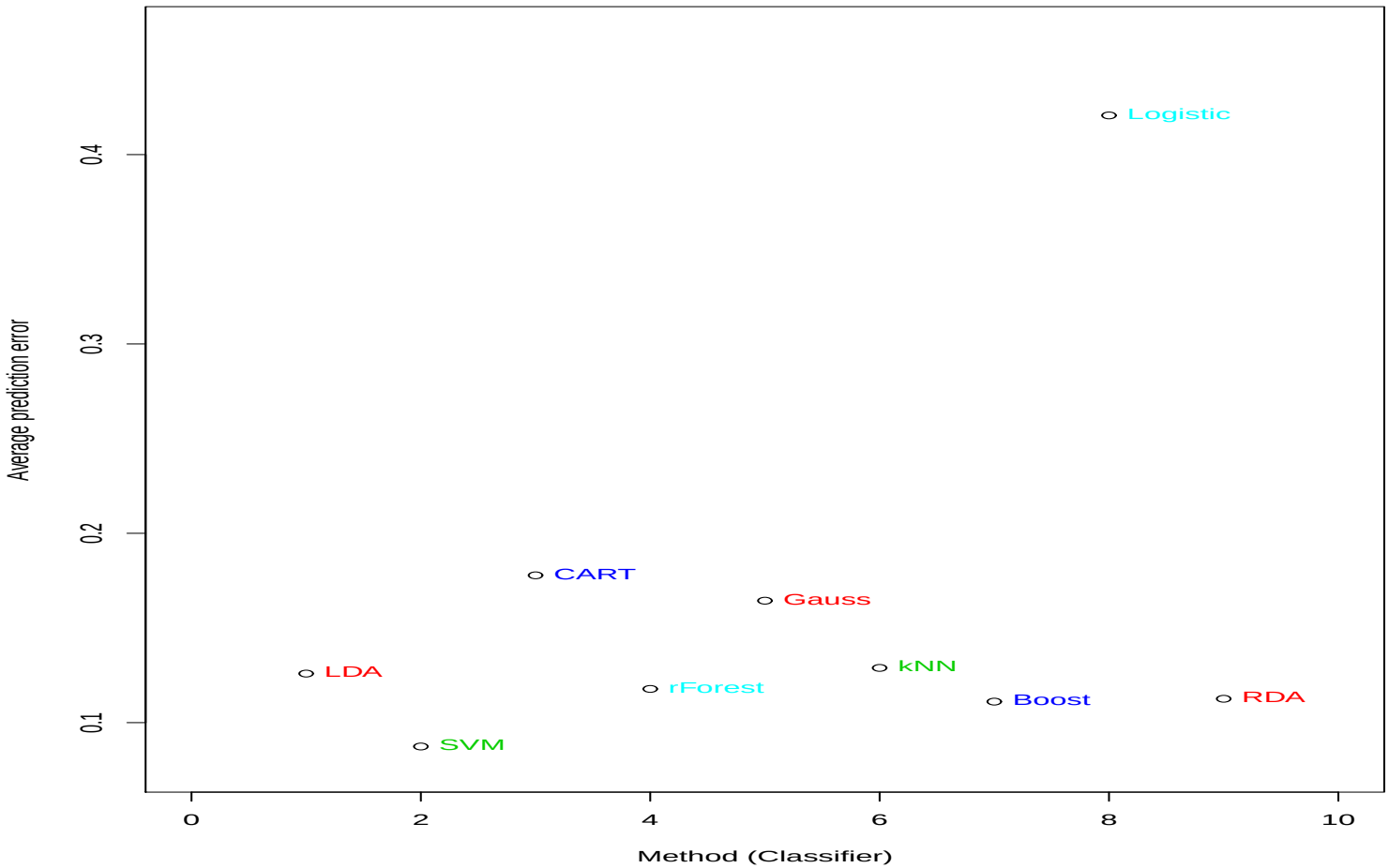


Figure: *Comparison of the average prediction error over* $R = 100$ *replications on the lymphoma data*

References

Clarke, B, Fokoué, E and Zhang, H (2009). Principles and Theory for Data Mining and Machine Learning. Springer Verlag, New York, (ISBN: 978-0-387-98134-5), (2009)