# Data Mining with R

## Fisher Discrimination

Hugh Murrell

# references

These slides are based on a paper of mine that appeared in **The Mathematica Journal** in 2011:

*Fisher Discrimination with Kernels*
by: Hugh Murrell, Kazuo Hashimoto and Daichi Takator

the paper may be downloaded from:
`www.mathematica-journal.com/data/uploads/2011/07/Murrell.pdf`

# linear Fisher discriminantion

In 1936 *R.A. Fisher* published a proceedure for finding the *best* separating plane in data with non-separable classes.

These ideas are now known as *Linear Discriminant Analysis* or LDA and are now offered as a powerful classification tool by all statistical packages including R.

In these slides we will describe Fisher's original algorithm but we will use R's built in LDA function to learn a full classification proceedure from the `iris` training data.

But first we need to understand the concept of a *scatter* matrix for a distribution of points in feature space.

# the scatter matrix

To introduce the *scatter* matrix and its *action*, consider a collection of points in feature space $X = \{\vec{x}_i\}$ with mean $\vec{\mu}$. Let $\vec{w}$ be a unit vector. Now the *action* is the *variance* of the projections of the $\vec{x}_i - \vec{\mu}$ onto $\vec{w}$.

$$\begin{aligned} A(X, \vec{w}) &= \sum_i ((\vec{x}_i - \vec{\mu}) \cdot \vec{w})^2 \\ &= \sum_i (\vec{w}^T (\vec{x}_i - \vec{\mu})) \times ((\vec{x}_i - \vec{\mu})^T \vec{w}) \\ &= \vec{w}^T (\sum_i (\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T) \vec{w} \\ &= \vec{w}^T S \vec{w} \end{aligned}$$

This then defines the *scatter* matrix, $S$ and its *action* on a unit vector $\vec{w}$.

# The Fisher Discriminant

A Linear Fisher Discriminant is the vector $\vec{w}$ that maximizes:

$$J(\vec{w}) = \frac{\vec{w}^T S_B \vec{w}}{\vec{w}^T S_W \vec{w}}$$

where the between-class and within-class scatter matrices are defined by:

$$S_B = \sum_c N_c (\vec{\mu}_c - \vec{\mu})(\vec{\mu}_c - \vec{\mu})^T$$

and

$$S_W = \sum_c \sum_{i \in c} (\vec{x}_i - \vec{\mu}_c)(\vec{x}_i - \vec{\mu}_c)^T$$

where $\vec{\mu}$ is the mean of the $\vec{x}_i$ and $\vec{\mu}_c$ is the mean of the $\vec{x}_i$ in class $c$.

# 2D test data

For the purposes of demonstration we generate two sets of normally distributed points with an elliptical shape. The two elliptical sets are rotated and translated away from each other and then adjusted to have zero combined mean.

```
> library(MASS)
> rotmat <- function(t){
+   return( matrix(c(cos(t),sin(t),-sin(t),cos(t)),nrow=2) )
+ }
> translate <- function(d,v){
+   plus <- function(v1,v2){v1+v2}
+   return(t(apply(d,1,plus,v)))
+ }
> np <- 1000
> nn <- 500
> xp <- mvrnorm(n = np, mu=c(0,0),
+         Sigma=matrix(c(1,0.7,0.7,1),nrow=2))
> xn <- mvrnorm(n = nn, mu=c(0,0),
+         Sigma=matrix(c(1,0.7,0.7,1),nrow=2))
> x <- rbind( translate(xp %*% rotmat(-pi/8),c(3/2,3/2)),
+             translate( xn %*% rotmat(pi/8) , c(-3/2,-3/2)) )
> m <- apply(x,2,mean)
> x <- translate(x,-m)
> x.class <- c(rep(1,np),rep(-1,nn))
> plot(x,col=x.class+3,asp=1)
```

# plot of test data

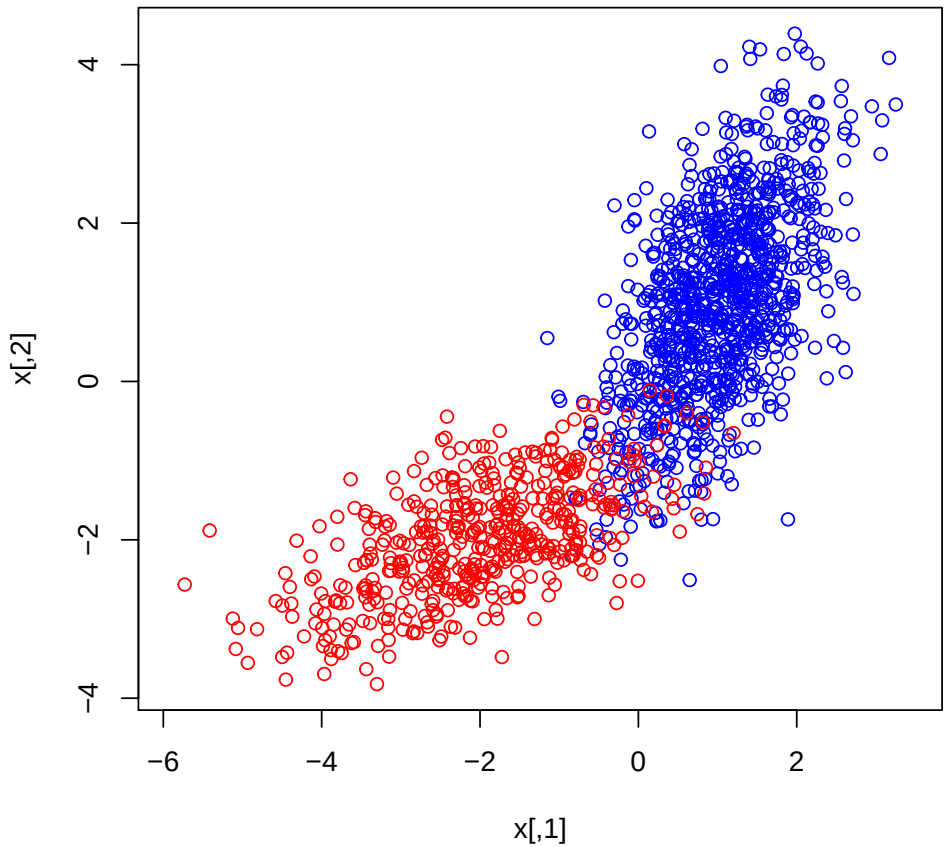


Figure: test data for Fisher discriminant: $+1$ in blue and -1 in red

# plot of test data

Now we can compute the $2 \times 2$ between-class and within-class scatter matrices for these two-dimensional datasets and plot the ratio of their action on a potential projection discriminant vector $\vec{w}$.

by *action* of matrix $M$ on unit row vector $\vec{u}$ we mean $\vec{u} \cdot M \cdot \vec{u}^T$. It is the *projection* of the transformed unit vector onto itself.

Scatter is another word for variance. $J(\vec{w})$ is the ratio of between-class and within-class variances after projection of the data onto a prospective discriminant, $\vec{w}$.

The Fisher linear discriminant is the vector that maximizes the scatter ratio and the Fisher separating plane is perpendicular to the Fisher discriminant.

# maximum scatter ratio

```
> u <- apply(x,2,mean)
> up <- apply(subset(x,x.class==+1),2,mean)
> un <- apply(subset(x,x.class==-1),2,mean)
> np <- sum(x.class==+1)
> nn <- sum(x.class==-1)
> SB <- nn * (un - u) %*% t(up - u)
> scatter <- function(v){
+   ( (v - un) %*% t(v - un) ) + ( (v - up) %*% t(v - up) )
+ }
> SW <- matrix( apply( apply(x,1,scatter), 1, sum ), nrow=2 )
> t <- seq(- pi , pi, 0.05)
> uv <- cbind(cos(t), sin(t))
> action <- function(uv, m) {
+   abs( uv %*% m %*% matrix(uv) )
+ }
> ratios <- apply(uv, 1, action, SB) / apply(uv, 1, action, SW)
> plot(x,col=x.class+3,asp=1)
> par(lwd=2)
> lines(20 * ratios * uv)
> mr <- which.max(ratios)
> muv <- uv[mr,]
> mv <- 40*ratios[mr]*muv
> arrows(0,0,mv[1],mv[2])
```

# Fisher projection vector for test dataset

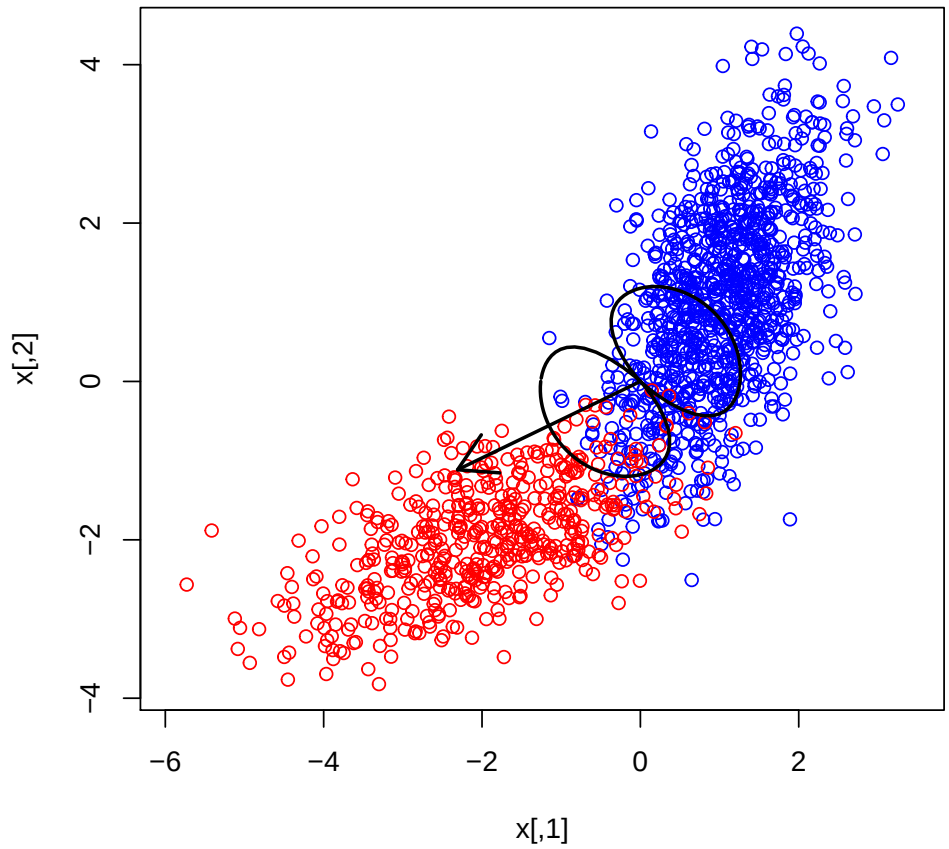


Figure: test data with best discriminating projection vector

# using the Fisher projection to predict class

The discriminating projection vector, $\vec{w}$, only gives the normal to the discriminating plane. We must still provide a bias, $b$, to translate the discriminating plane to a position that best separates the data.

We do this by projecting the data onto $\vec{w}$ and then using the so-called *Mahalonobis bias* to separate the two classes:

$$b = \frac{\mu_+\sigma_- + \mu_-\sigma_+}{\sigma_+ + \sigma_-}$$

where the various $\mu$ and $\sigma$ are the means and standard deviations of the two classes.

# using the Fisher projection to predict class

```
> xp <- as.vector(x %*% muv)
> rxp <- round(range(xp),0)+c(-1,1)
> xpn <- subset(xp,x.class==-1)
> xpp <- subset(xp,x.class==+1)
> break_points <- seq(rxp[1],rxp[2],length.out=30)
> hn <- hist(xpn,breaks=break_points,plot=FALSE)
> hp <- hist(xpp,breaks=break_points,plot=FALSE)
> b = (mean(xpp) * sd(xpn) + mean(xpn) * sd(xpp)) /
+     (sd(xpp) + sd(xpn))
> plot( hp, col=rgb(0,0,1,1/4),xlim=rxp,main="",xlab=
> plot( hn, col=rgb(1,0,0,1/4),xlim=rxp,add=TRUE)
> par(lwd=3)
> abline(v=b)
```

# Mahalonobis separation in projection space

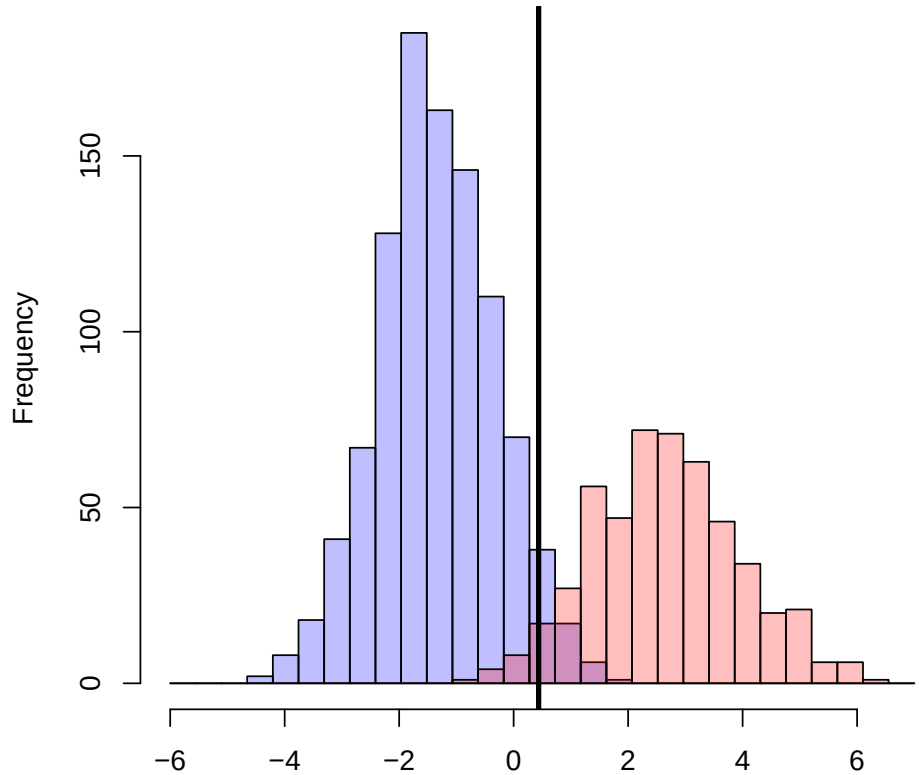


Figure: classes in projection space separated by $\frac{\mu_+\sigma_- + \mu_-\sigma_+}{\sigma_+ + \sigma_-}$

# drawing the discriminant

Now we have enough information to draw the Fisher discriminant.

```
> plot(x,col=x.class+3,asp=1)
> par(lwd=2)
> lines(10 * ratios * uv)
> arrows(0,0,mv[1],mv[2])
> abline(b/muv[2],-muv[1]/muv[2])
```

# Test dataset with Fisher discriminator

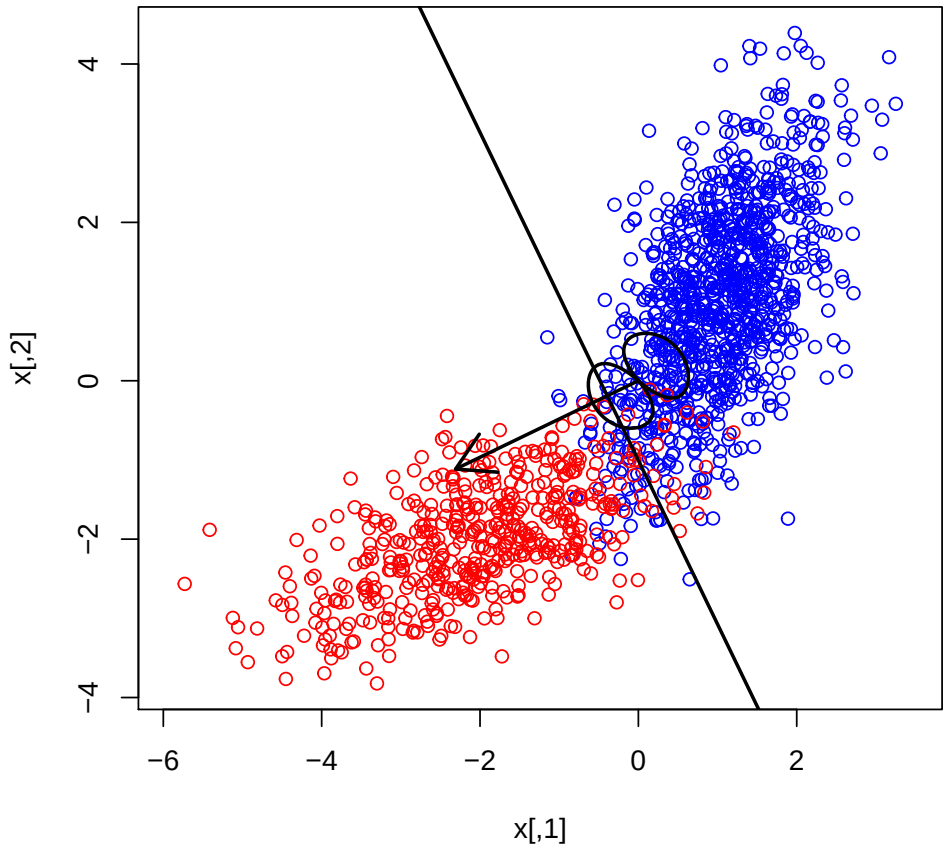


Figure: Test dataset with Fisher discriminator $(\vec{w}, b)$.

# predicting class from the Fisher discriminant

Given optimal Fisher projection vector $\vec{w}$ and Mahalonobis bias $b$ we can predict the class of an ovservation $\vec{x}$ via:

$$\hat{Y}(\vec{x}, b, \vec{w}) = \text{sign}(\vec{x} \cdot \vec{w} - b)$$

R code for predicting class for a collection of observations is as follows:

```
> distance.from.plane = function(x,w,b) {
+   b - sum(x*w)
+ }
> classify.fisher = function(x,w,b) {
+   distances =
+     apply(x, 1, distance.from.plane, w, b)
+   return(ifelse(distances < 0, -1, +1))
+ }
```

# classifying the non-separable iris data

```
> data(iris)
> x <- cbind(iris$Petal.Length,iris$Petal.Width)
> Y <- ifelse(iris$Species == "virginica", +1, -1)
> u <- apply(x,2,mean)
> up <- apply(subset(x,Y==+1),2,mean)
> un <- apply(subset(x,Y==-1),2,mean)
> np <- sum(Y==+1)
> nn <- sum(Y==-1)
> SB <- nn * (un - u) %*% t(up - u)
> SW <- matrix( apply( apply(x,1,scatter), 1, sum ), nrow=2 )
> ratios <- apply(uv, 1, action, SB) / apply(uv, 1, action, SW)
> mr <- which.max(ratios)
> muv <- uv[mr,]
> mv <- 40*ratios[mr]*muv
> xp <- as.vector(x %*% muv)
> rxp <- round(range(xp),0)+c(-1,1)
> xpn <- subset(xp,Y==-1)
> xpp <- subset(xp,Y==+1)
> b = (mean(xpp) * sd(xpn) + mean(xpn) * sd(xpp)) /
+     (sd(xpp) + sd(xpn))
> plot(x,col=Y+3,asp=1)
> par(lwd=2)
> abline(b/muv[2],-muv[1]/muv[2])
> sum(abs(Y - classify.fisher(x,muv,b) ))

[1] 18
```

# iris dataset with Fisher discriminator

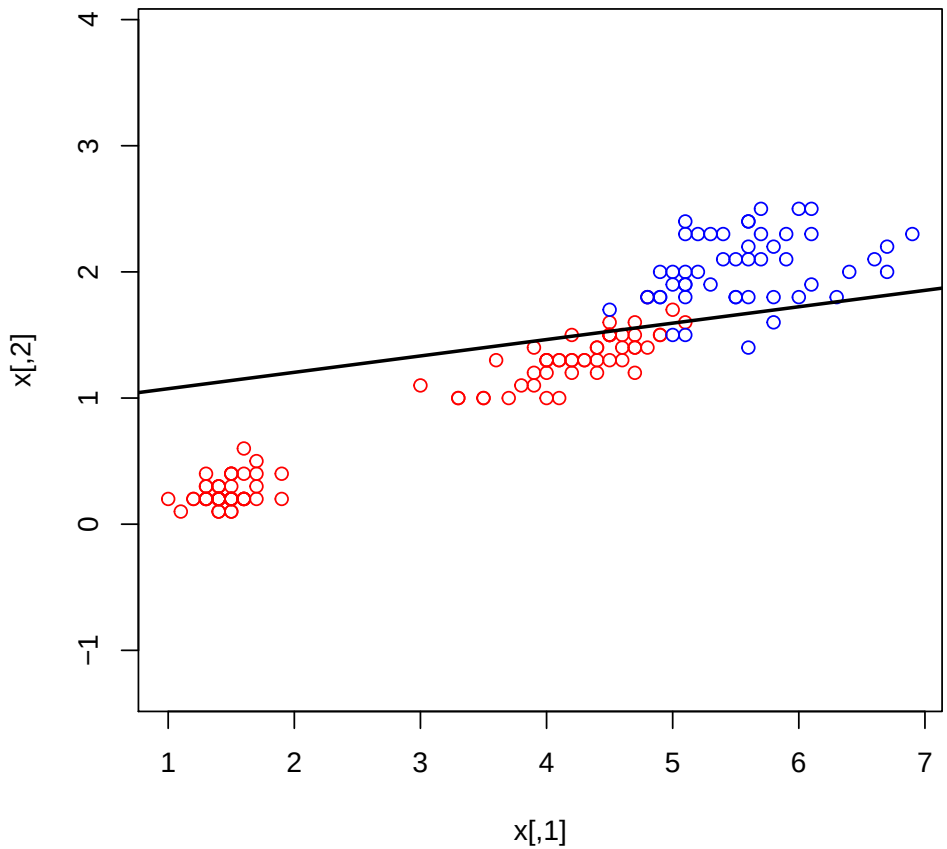


Figure: iris dataset with Fisher discriminator $(\vec{w}, b)$.

# lda: linear discriminant analysis

The MASS package provides a Fisher type linear discriminant analysis function:

```
lda(formula, data, ...)
```

where formula is of the form `Y  X1 + X2 + ...` specifying the response variable Y and the discriminators Xi in the data frame data.

After execution the lda function returns an lda object that can be used to predict classification of new data points with Xi coordinates.

In the next two slides we will use lda to separate virginica from the other two species in the iris dataset. Although lda does not explicidly return a Fisher separation line we will trick the function into revealing the line.

# lda for partial iris dataset

```
> library(MASS)
> X <- data.frame(length=iris$Petal.Length,width=iris$Petal.Width)
> X$class <- ifelse(iris$Species == "virginica", +1, -1)
> LDA <- lda(class ~ length + width, data = X)
> X$pred <- predict(LDA)$class
> X$col <- 2+ifelse(X$pred == X$class, as.numeric(X$class), 0)
> color <- c("red","green", "blue","red","blue")
> plot(X$length,X$width,col=color[X$col],pch=2,cex=1)
> pts <- 3000
> rpts <- function(v,pts){
+   runif(pts,min=v[1],max=v[2])
+ }
> grid <- data.frame(apply(apply(X[,1:2],2,range),2,rpts,pts))
> grid$pred <- 3+as.numeric(predict(LDA,grid)$class)
> points(grid$length,grid$width,col=color[grid$pred],cex=0.1)
> sum(X$class != X$pred)

[1] 8
```

# lda separation for partial iris dataset

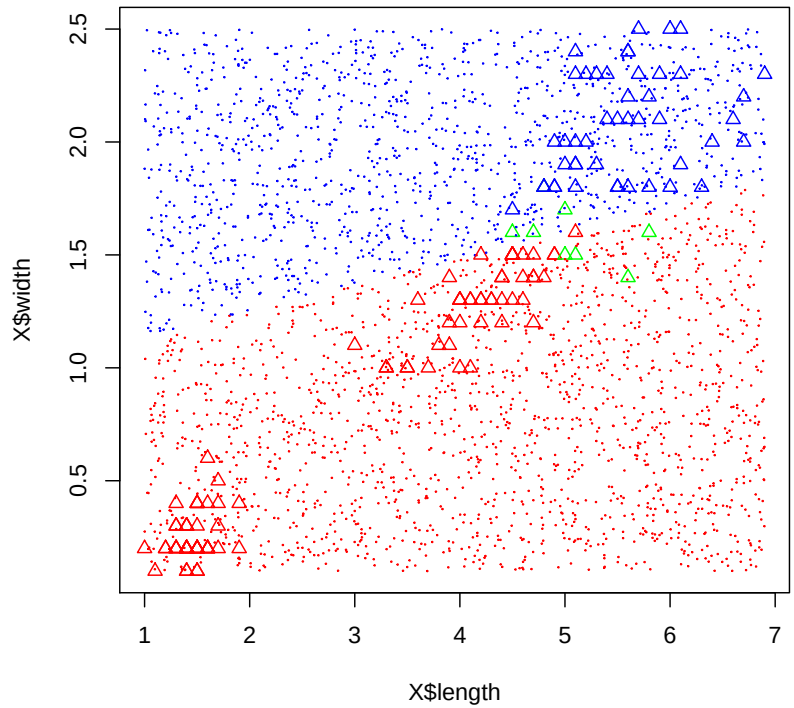


Figure: lda separation on partial iris dataset. The Species `virginica` is separated ffrom the rest using `petal.length` and `petal.width`. The Fisher separation line is revealed by asking `lda` to classify a large collection of randomly generated data.

# lda for full feature set

Like Fisher discrimination, `lda` is not limited to two discriminators. One can perform discrimination on a response variable using multiple discriminators.

The `lda` function returns projection vectors in order of importance. Projections onto the *best* discriminant are subtracted from the data and the process is repeated to obtain the `next best` discriminant vector.

The `predict` function wil project new data points onto these *best* discriminants thus allowing the user to view separation of multiple classes in the plane of the best two discriminators

In the next two slides we will make use of `lda` and `ggplot` to separte all three iris Species using the best two linear discriminants.

# lda for full iris dataset

```
> LDA <- lda(Species ~ ., data = iris)
> X <- data.frame(predict(LDA)$x)
> X$class <- factor(ifelse(predict(LDA)$class == iris$Species,
+                          as.numeric(iris$Species), 0))
> library(ggplot2)
> color <- c("red", "green", "blue","purple")
> pts <- 50000
> cube <- data.frame(
+   apply(apply(iris[,1:4],2,range),2,rpts,pts))
> cube.pred <- predict(LDA,cube)
> X.grid <- data.frame(cube.pred$x)
> X.grid$class <- factor(as.numeric(cube.pred$class))
> p <- ggplot(X, aes(x = LD1, y = LD2)) +
+   geom_point(aes(colour=class),size=3)
> p + geom_point(data=X.grid,aes(colour=class),size=0.5)
> sum(iris$Species != predict(LDA)$class)

[1] 3
```

# lda separation for full iris dataset

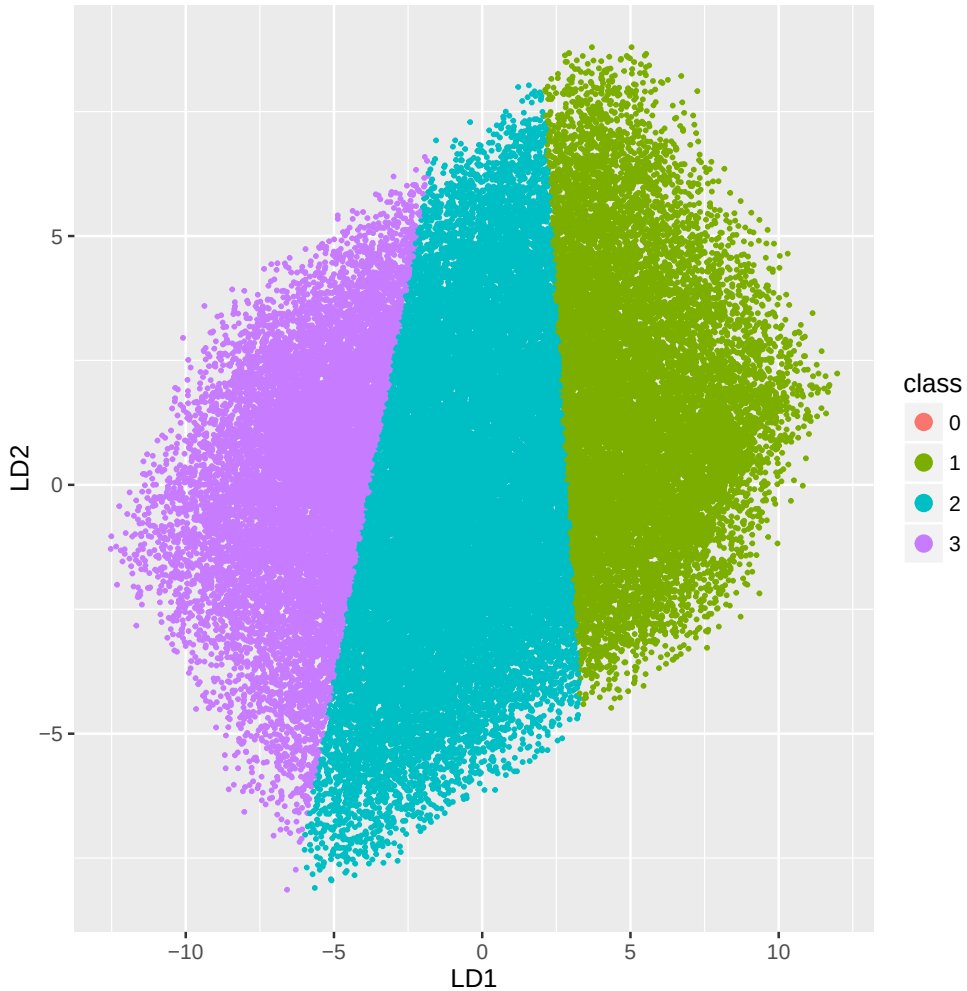


Figure: lda separation on full iris dataset.

# exercises

- **code:** Perform full linear discriminant analysis on the `wine` dataset from my website and show how Cultivar separates completely in the 2 dimensional linear discriminant feature space `(LD1, LD2)`.
- **submit:** email your lda script to me by Monday the $2^{nd}$ May, 06h00. When run, your script should generate the following separation diagram.

# exercises ...

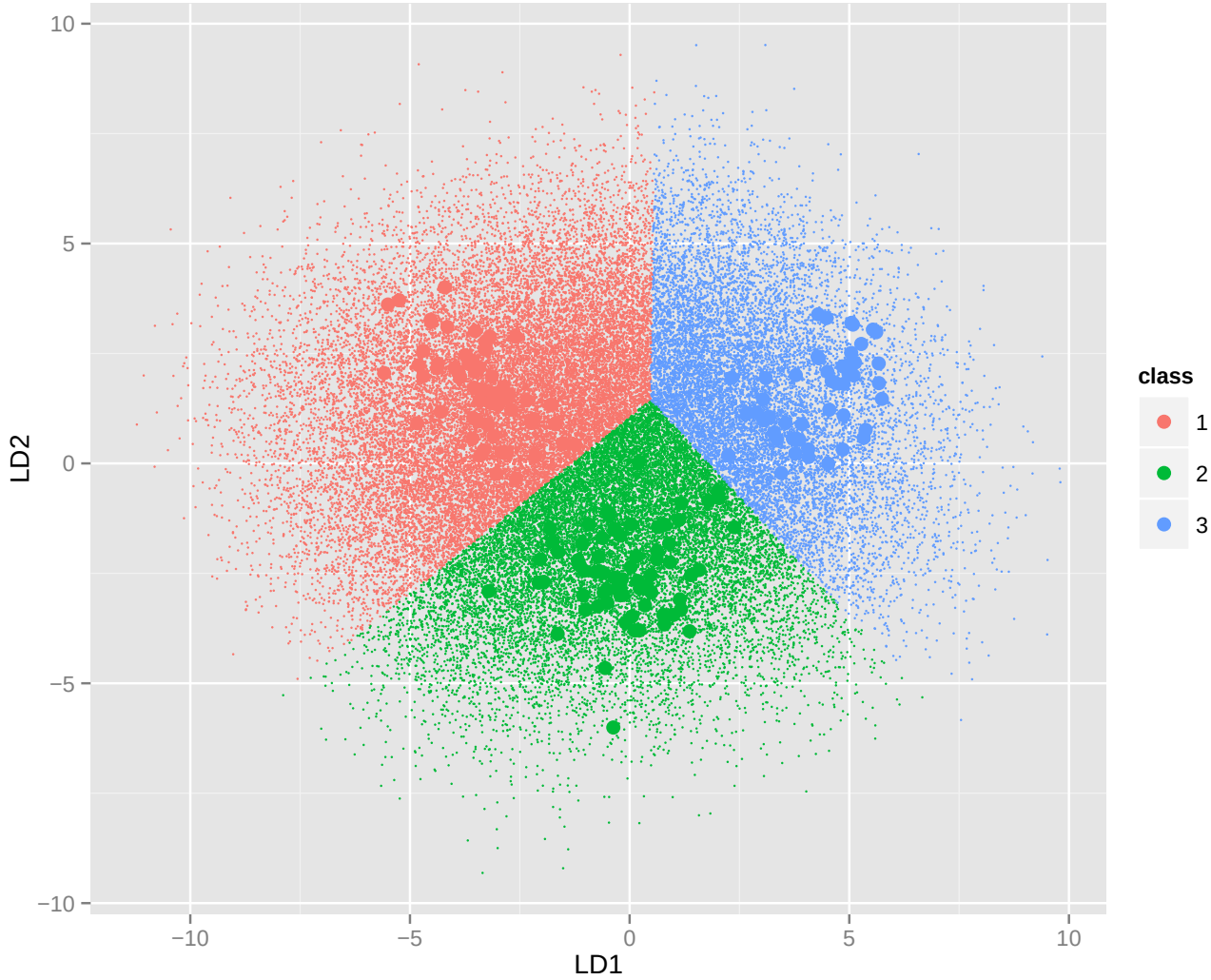


Figure: `lda` separation of `Cultivar` in the `wine` dataset.