Multivariate Analysis Lecture 12: Linear Discrminant Analysis

Zhaoxia Yu
Professor, Department of Statistics

2025-05-08

Section 1

Summary of PCA

The Spectral Decomposition of A Covariance Matrix

- Let $\mathbf{\Sigma}_{p \times p}$ be the covariance matrix of $\mathbf{X}$.
- A Covariance matrix is a positive definite or positive semi-definite, which means

$$\mathbf{\Sigma} = \Gamma \Lambda \Gamma^T$$

where Λ is the diagonal elements with the diagonal elements being

$$\lambda_1 \geq \lambda_2 \geq \cdots \lambda_p \geq 0$$

- $\Gamma = (\gamma_1, \cdots, \gamma_p)$, with the ith column being the eigenvector corresponding to λ_i.

The Maximum Variance of $a^T\mathbf{X}$ S.B.T $||a|| = 1$

- First Principal Component is the linear combination with the maximum variance. The first PC is $Y_1 = \gamma_1^T\mathbf{X}$. We have shown that

$$\gamma_1 = \arg\max_{a^Ta=1} a^T\mathbf{\Sigma}a$$

- The second PC is $Y_2 = \gamma_2^T\mathbf{X}$. We have shown that

$$\gamma_2 = \arg\max_{a^Ta=1, a^T\gamma_1=0} a^T\mathbf{\Sigma}a$$

- The ith principal component is

$$Y_i = \gamma_i^T\mathbf{X}$$

  and

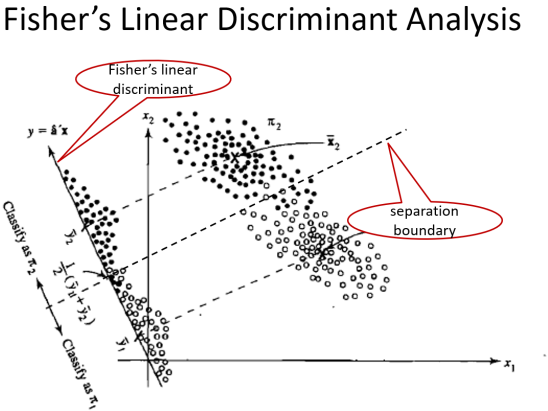
$$\gamma_i = \arg\max_{a^Ta=1, a^T\gamma_1=0, \cdots, a^T\gamma_{i-1}=0} a^T\mathbf{\Sigma}a$$

Section 2

Introduction

Linear Discrminant Analysis

```r
knitr::include_graphics("img/FLDA.png")
```



LDA looks for linear boundaries

- Linear Discriminant Analysis (LDA) is a classification technique
- A linear discriminant is a linear function of the variables / features
- The goal of LDA is to find linear combinations of features that best separate the classes in the data.
- LDA sounds similar to PCA

PCA vs LDA

- Both PCA and LDA try to find linear functions. Among all linear combinations,
  - the first PC explains the most variance of the data
  - the first LD leads to the maximum separation of the data
- Use of data labels
  - PCA analyzes the pooled data with using the label information, although the leading PCs often show separation of the data
  - LDA uses the labels
- Type of learning
  - PCA is an unsupervised learning technique
  - LDA is a supervised learning technique

PCA vs LDA

- PCA reduces the dimensionality of a dataset by identifying the most important features or variables that capture the most variance in the data.
- PCA tries to find a lower-dimensional representation of the data that retains as much of the original information as possible.
- PCA is useful for data visualization, noise reduction, and feature extraction.
- LDA tries to find a lower-dimensional representation of the data that maximizes the separation between classes.
- LDA is useful for predicting new observations and identifying the most important features for classification.

Outline

- FLDA: LDA for two classes
  - Fisher's LDA (FLDA)
  - maximum likelihood
  - minimum distance
- LDA for multiple classes
- Decision rules
- LDA vs logistic regression
- QDA

Section 3

FLDA

LDA for two classes (Fisher's LDA)

- Fisher 1936 proposed a dichotomous discriminant analysis
- Fisher's linear discriminant function is a linear function
- The linear function has the maximum ability to discriminant between samples
- Once we find the linear function, we
  - project the data on to it
  - find the boundary of different classes
  - allocate new observations

FLDA: Assumptions

- Let's consider a two-class classification problem with n_1 and n_2 observations in classes 1 and 2, respectively.
- Suppose we have two independent random samples
  - Sample 1: $X_{1j} \overset{iid}{\sim} (\mu_1, \mathbf{\Sigma})$, where $j = 1, \cdots, n_1$
  - Sample 2: $X_{2j} \overset{iid}{\sim} (\mu_2, \mathbf{\Sigma})$, where $j = 1, \cdots, n_2$
- Sample mean vectors:

$$\bar{\mathbf{X}}_1 = \frac{1}{n_1}\sum_{j=1}^{n_1} X_{1j}, \bar{\mathbf{X}}_2 = \frac{1}{n_2}\sum_{j=1}^{n_2} X_{2j}$$

FLDA: The Goal

- FLDA aims to find a linear combination of features that maximally separates two samples.
- How to define separability of a linear function?
- Consider a linear function with coefficients being denoted by a vector a.
    - $a^T\bar{\mathbf{X}}_1 \sim (a^T\mu_1, \frac{1}{n_1}a^T\boldsymbol{\Sigma}a)$
    - $a^T\bar{\mathbf{X}}_2 \sim (a^T\mu_2, \frac{1}{n_2}a^T\boldsymbol{\Sigma}a)$
- $a^T\bar{\mathbf{X}}_1 - a^T\bar{\mathbf{X}}_2$ measures the difference but the variation of this difference depends on the scale of a and also the covariance structure
- We need to "standardize" it by its standard error

FLDA: Maximum Separability

- Recall that we have two independent random samples. Therefore,
  - $\bar{\mathbf{X}}_1$ and $\bar{\mathbf{X}}_2$ are independent
  - As a result,

$$(a^T\bar{\mathbf{X}}_1 - a^T\bar{\mathbf{X}}_2) \sim \left(a^T\mu_1 - a^T\mu_2, (\frac{1}{n_1} + \frac{1}{n_2})\boldsymbol{a}^T\Sigma a\right)$$

- The standardized version is

$$\frac{a^T\bar{\mathbf{X}}_1 - a^T\bar{\mathbf{X}}_2}{\sqrt{(\frac{1}{n_1} + \frac{1}{n_2})a^T\boldsymbol{\Sigma}a}}$$

FLDA: Maximum Separability

- The sign does not matter. So we consider the squared statistic

$$\frac{(a^T \bar{\mathbf{X}}_1 - a^T \bar{\mathbf{X}}_2)^2}{(\frac{1}{n_1} + \frac{1}{n_2}) a^T \mathbf{\Sigma} a}$$

- Note that this is the squared t-statistic for testing $a^T \mu_1 = a^T \mu_2$
- The Fisher LDA aims to find a linear combination of features $Y = a^T X$ that maximally separates the classes while minimizing the within-class variance. This can be expressed as:

$$\frac{(a^T \bar{\mathbf{X}}_1 - a^T \bar{\mathbf{X}}_2)^2}{a^T \mathbf{\Sigma} a}$$

The Maximization Problem

- First, we write the ratio

$$
\begin{aligned}
& \max \frac{(a^T\bar{\mathbf{X}}_1 - a^T\bar{\mathbf{X}}_2)^2}{a^T\boldsymbol{\Sigma}a} \\
= & \max \frac{a^T(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T a}{a^T\boldsymbol{\Sigma}a} \\
\text{Let } \overset{b=\Sigma^{1/2}a}{=} & \max \frac{b^T\Sigma^{-1/2}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T\Sigma^{-1/2}b}{b^Tb}
\end{aligned}
$$

The Maximization Problem (Continued)

- Let
$$A = \mathbf{\Sigma}^{-1/2}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T \Sigma^{-1/2}$$
- The maximization problem becomes to maximize
$$\frac{b^T A b}{b^T b}$$
- From the derivation of *PCA*, we understand that the maximum equals the largest eigenvalue of A.
- The rank of A is 1, which means there is only one non-zero eigenvalue, which is
$$(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T \mathbf{\Sigma}^{-1}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)$$

The Maximization Problem (Continued)

- The maximum of $\frac{b^T A b}{b^T b}$ is attained when b is the first eigenvector of A. It can be verfied that

$$b = \mathbf{\Sigma}^{-1/2}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2),$$

which implies that

$$a = \mathbf{\Sigma}^{-1}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)$$

- Therefore, the linear discriminant is a projection to the vector $a = \mathbf{\Sigma}^{-1}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)$

Practical Issues

- We need to replace $\boldsymbol{\Sigma}$ by the pooled sample covariance matrix

$$\mathbf{S}_p = \frac{(n_1 - 1)\mathbf{S}_1 + (n_2 - 1)\mathbf{S}_2}{n_1 + n_2 - 2}$$

where $\mathbf{S}_1$ and $\mathbf{S}_2$ are the sample covariance matrices:

$$\mathbf{S}_1 = \frac{1}{n_1 - 1} \sum_{j=1}^{n_1} (X_{1j} - \bar{\mathbf{X}}_1)(X_{1j} - \bar{\mathbf{X}}_1)^T$$

$$\mathbf{S}_2 = \frac{1}{n_2 - 1} \sum_{j=1}^{n_2} (X_{2j} - \bar{\mathbf{X}}_2)(X_{2j} - \bar{\mathbf{X}}_2)^T$$

Allocate New Observations

- The linear function

$$f(x) = a^T x \text{ where } a = \mathbf{S}_p^{-1}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)$$

is called Fisher's linear discriminant function.

- Let

$$m = a^T \frac{\bar{\mathbf{X}}_1 + \bar{\mathbf{X}}_2}{2} = (\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T \mathbf{S}_p^{-1} \frac{\bar{\mathbf{X}}_1 + \bar{\mathbf{X}}_2}{2}$$

- Consider an observation X_0. We compute $f(X_0)$ and allocate it to
  - class 1 if $f(X_0) > m$
  - class 2 if $f(x_0) < m$

The Linear Boundary

- The boundary $f(x) = m$ is linear
- Consider a two-class classification problem in $\mathbb{R}^2$, i.e., there are two features.
- The line that separates the two classes is $f(x) = m$, i.e.,

$$a_1 x_1 + a_2 x_2 = m$$

- Rewrite it into the standard intercept and slope format, we have

$$x_2 = \frac{m}{a_2} - \frac{a_1}{a_2} x_1$$

Subsection 1

As A Distance Approach

The FLDA as A Distance Approach

- The rule in FLDA is equivalent to obtain the sign of $f(X_0) - m$

$$
\begin{aligned}
f(X_0) - m &= a^T(X_0 - \frac{\bar{\mathbf{X}}_1 + \bar{\mathbf{X}}_2}{2}) \\
&= (\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T \mathbf{S}_p^{-1}(X_0 - \frac{\bar{\mathbf{X}}_1 + \bar{\mathbf{X}}_2}{2}) \\
&= (\bar{\mathbf{X}}_1 - X_0 + X_0 - \bar{\mathbf{X}}_2)^T \mathbf{S}_p^{-1}(\frac{X_0 - \bar{\mathbf{X}}_1 + X_0 - \bar{\mathbf{X}}_2}{2}) \\
&= \frac{1}{2}[(X_0 - \bar{\mathbf{X}}_2)^T \mathbf{S}_p^{-1}(X_0 - \bar{\mathbf{X}}_2) - (X_0 - \bar{\mathbf{X}}_1)^T \mathbf{S}_p^{-1}(X_0 - \bar{\mathbf{X}}_1)] \\
&= \frac{1}{2}[D^2_{S_p}(X_0, \bar{\mathbf{X}}_2) - D^2_{S_p}(X_0, \bar{\mathbf{X}}_1)]
\end{aligned}
$$

The FLDA as A Distance Approach

- In the previous slide, we showed that

$$f(X_0) - m = \frac{1}{2}[D^2_{S_p}(X_0, \bar{\mathbf{X}}_2) - D^2_{S_p}(X_0, \bar{\mathbf{X}}_1)]$$

  where $D^2_{S_p}(X_0, \bar{\mathbf{X}}_g)$ denotes the Mahalanobis distance between X_0 and $\bar{\mathbf{X}}_g$ for $g = 1, 2$.
- Therefore, $f(X_0) - m > 0 \Leftrightarrow D_{S_p}(X_0, \bar{\mathbf{X}}_2) > D_{S_p}(X_0, \bar{\mathbf{X}}_1)$, we class X_0 to class 1.
- Similarly, we allocate X_0 to class 2 if $D_{S_p}(X_0, \bar{\mathbf{X}}_2) < D_{S_p}(X_0, \bar{\mathbf{X}}_1)$
- Thus, the FLDA is also a minimum distance method for classsification.

Subsection 2

As A Maximum Likelihood Approach

The FLDA as A Maximum Likelihood Approach

- The likelihood function if X_0 is from $N(\mu_1, \mathbf{\Sigma})$

$$L_1 \propto |\mathbf{\Sigma}|^{-1/2} exp\{-\frac{1}{2}(X_0 - \mu_1)^T \mathbf{\Sigma}^{-1}(X_0 - \mu_1)\}$$

- The likelihood function if X_0 is from $N(\mu_2, \mathbf{\Sigma})$

$$L_2 \propto |\mathbf{\Sigma}|^{-1/2} exp\{-\frac{1}{2}(X_0 - \mu_2)^T \mathbf{\Sigma}^{-1}(X_0 - \mu_2)\}$$

The FLDA as A Maximum Likelihood Approach

- The ratio is

$$\frac{L_1}{L_2} \overset{Data}{=} exp\{\frac{1}{2}[(X_0-\bar{\mathbf{X}}_2)^T\mathbf{\Sigma}^{-1}(X_0-\bar{\mathbf{X}}_2)-(X_0-\bar{\mathbf{X}}_1)^T\mathbf{\Sigma}^{-1}(X_0-\bar{\mathbf{X}}$$
$$= exp\{\frac{1}{2}[D^2_{S_p}(X_0,\bar{\mathbf{X}}_2)-D^2_{S_p}(X_0,\bar{\mathbf{X}}_1)]\}$$

- Allocate X_0 to class 1 if

$$\frac{L_1}{L_2} > 1 \Leftrightarrow D^2_{S_p}(X_0,\bar{\mathbf{X}}_2) > D^2_{S_p}(X_0,\bar{\mathbf{X}}_1) \Leftrightarrow f(X_0) > m$$

- Therefore, FLDA is also a maximum likelihood approach
- We will discuss decision theory in classification next week

Section 4

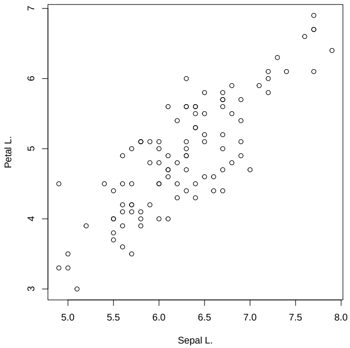
Examples

Subsection 1

Iris Data: Two Species, Two Features

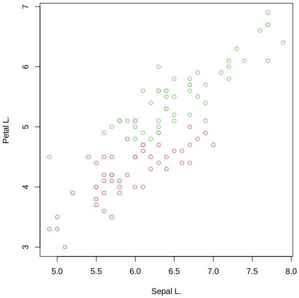
The Data

```
sample1=iris3[,c(1,3),2]#Versicolor
sample2=iris3[,c(1,3),3]#Virginica
sample12=rbind(sample1, sample2)
par(pty="s")
plot(sample12)
```



This is a Supervised Learning

```
pch=c("e","i"); col=c(2,3); xlab="SepalL"; ylab="PetalL"
par(pty="s")
plot(sample12,type="n")
points(sample1, col=col[1])
points(sample2, col=col[2])
```



The Data: Sample Mean Vectors

```r
colMeans(sample1)
```

```
## Sepal L. Petal L.
##    5.936    4.260
```

```r
colMeans(sample2)
```

```
## Sepal L. Petal L.
##    6.588    5.552
```

```r
mean.diff=c( colMeans(sample1)-colMeans(sample2) )
data.center=c( (colMeans(sample1)+colMeans(sample2))/2 )
```

The Data: Pooled Sample Covariance Matrix

```
n1=dim(sample1)[1]
n2=dim(sample2)[1]
S.pooled=((n1-1)*cov(sample1)+(n2-1)*cov(sample2))/(n1+n2-2)
S.pooled
```

```
##           Sepal L.  Petal L.
## Sepal L. 0.3353878 0.2430939
## Petal L. 0.2430939 0.2627020
```

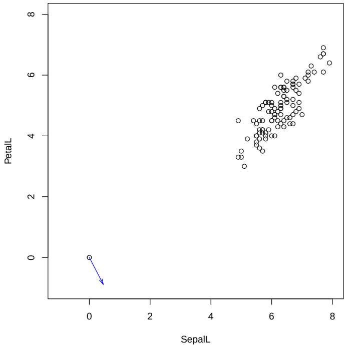
Compute The Linear Discrminant

```
lda.coeff=solve(S.pooled)%*% mean.diff
#rescale it so that is has norm 1
lda.coeff=lda.coeff/sqrt(sum(lda.coeff^2))
lda.coeff
```

```
##                [,1]
## Sepal L.  0.4610660
## Petal L. -0.8873658
```

Visualize the LD Coefficients

```
par(pty="s")
plot(sample12, xlim=c(-1,8), ylim=c(-1,8), xlab=xlab, ylab=ylab)
points(0, 0)
arrows(0, 0, lda.coeff[1], lda.coeff[2], length = 0.1, angle=15, col="blue")
```



The Projection

- We project the data matrix to the linear discriminant vector

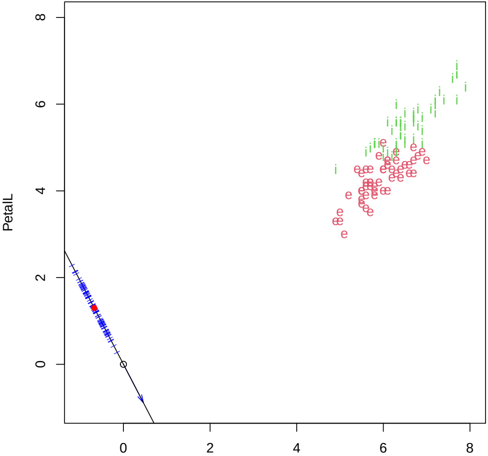
$$Porj_a(\mathbf{X}) = (\mathbf{X}a)a^T$$

```
m=c(t(lda.coeff)%*%data.center)
proj=(sample12%*%lda.coeff)%*%matrix(lda.coeff, 1,2)
#note: proj includes the direction of the projected values
#note: (sample12%*%lda.coeff) gives the scalar values
```

Visualize the Projection

```
par(pty="s")
plot(sample12, xlim=c(-1,8), ylim=c(-1,8), xlab=xlab, ylab=ylab, type="n")
points(sample1, pch=pch[1], col=col[1])
points(sample2, pch=pch[2], col=col[2])
points(0,0)
arrows(0, 0, lda.coeff[1], lda.coeff[2], length = 0.1, angle=15, col="blue")
abline(a=0, b=lda.coeff[2]/lda.coeff[1])
for(i in 1: (n1+n2))
  text(x=proj[i,1],y=proj[i,2], labels="|", col="blue", srt=atan(lda.coeff[2]/lda.coeff[1])*180/pi, cex=0.
#the center the projected data
points(m*lda.coeff[1], m*lda.coeff[2], pch=16, col="red")
```

Visualize the Projection

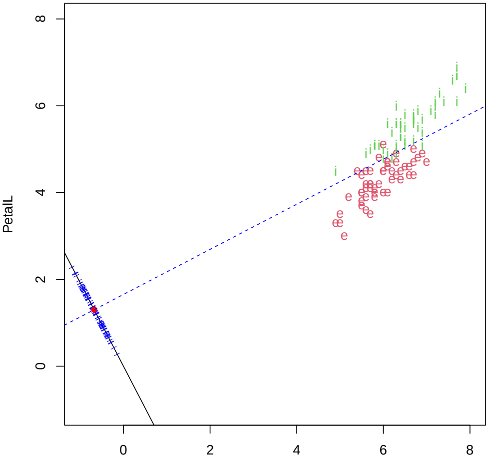


Find the Boundary

```
par(pty="s")
plot(sample12, xlim=c(-1,8), ylim=c(-1,8), xlab=xlab, ylab=ylab, type="n")
points(sample1, pch=pch[1], col=col[1])
points(sample2, pch=pch[2], col=col[2])

abline(a=0, b=lda.coeff[2]/lda.coeff[1])
for(i in 1: (n1+n2))
  text(x=proj[i,1],y=proj[i,2], labels="|", col="blue", srt=atan(lda.coeff[2]/lda.coeff[1])*180/pi, cex=0.
points(m*lda.coeff[1], m*lda.coeff[2], pch=16, col="red")
abline(a=m/lda.coeff[2], b=-lda.coeff[1]/lda.coeff[2], col="blue", lty=2)
```

Find the Boundary

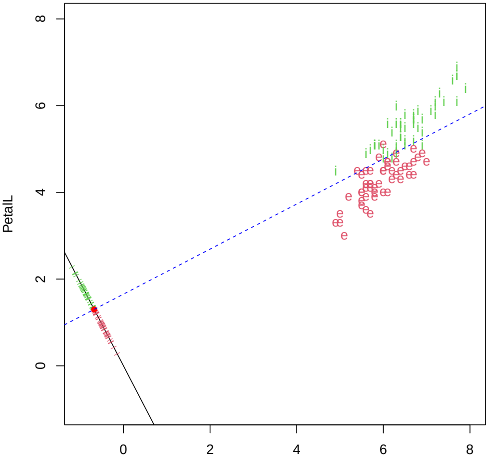


Visualize the Allocations

```r
proj.scalar=(sample12%*%lda.coeff)
par(pty="s")
plot(sample12, xlim=c(-1,8), ylim=c(-1,8), xlab=xlab, ylab=ylab, type="n")
points(sample1, pch=pch[1], col=col[1])
points(sample2, pch=pch[2], col=col[2])

abline(a=0, b=lda.coeff[2]/lda.coeff[1])
for(i in 1: (n1+n2)){
  if(proj.scalar[i]>m)
    text(x=proj[i,1],y=proj[i,2], labels="|", col=col[1],
         srt=atan(lda.coeff[2]/lda.coeff[1])*180/pi, cex=0.5)
  if(proj.scalar[i]<m)
    text(x=proj[i,1],y=proj[i,2], labels="|", col=col[2],
         srt=atan(lda.coeff[2]/lda.coeff[1])*180/pi, cex=0.5)
}
points(m*lda.coeff[1], m*lda.coeff[2], pch=16, col="red")
abline(a=m/lda.coeff[2], b=-lda.coeff[1]/lda.coeff[2], col="blue", lty=2)
```

Visualize the Allocations



Misclassification (Training Error)

```
sum(proj.scalar[1:n1]>m)
```

```
## [1] 47
```

```
sum(proj.scalar[1:n1]>m)/n1
```

```
## [1] 0.94
```

```
sum(proj.scalar[-c(1:n2)]<m)
```

```
## [1] 47
```

```
sum(proj.scalar[-c(1:n2)]<m)/n2
```

```
## [1] 0.94
```

```
(sum(proj.scalar[1:n1]>m) + sum(proj.scalar[-c(1:n1)]<m)) /(n1+n2)
```

```
## [1] 0.94
```

Subsection 2

PCA vs LDA

PCA and LDA: Linear Combinations

- Although both are linear combinations, the goals are quite different
- LDA vs the 1st PC:
  - LDA: $a = \mathbf{S}_p^{-1}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)$
  - First PC: $a = \gamma_1$, the first eigenvector of $\mathbf{\Sigma}$.
- In the Iris two-class two-feature example, the coefficients are

```
lda.coeff
```

```
##                 [,1]
## Sepal L.  0.4610660
## Petal L. -0.8873658
```

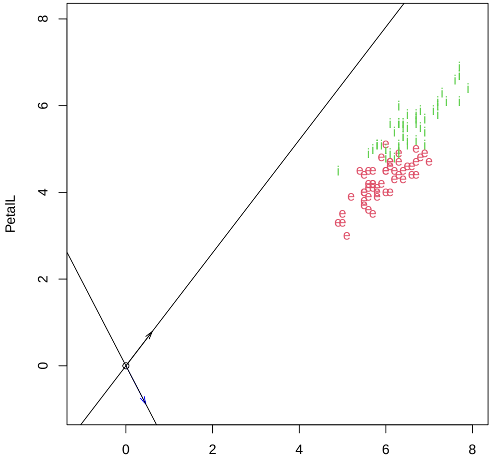
```
pca.coeff=eigen(cov(sample12))$vector[,1]
pca.coeff
```

```
## [1] 0.6090576 0.7931260
```

PCA and LDA: Linear Combinations

```
proj.pca=(sample12%*%pca.coeff)%*%matrix(pca.coeff, 1,2)
par(pty="s")
plot(sample12, xlim=c(-1,8), ylim=c(-1,8), xlab=xlab, ylab=ylab, type="n")
points(sample1, pch=pch[1], col=col[1])
points(sample2, pch=pch[2], col=col[2])
points(0, 0)
arrows(0, 0, lda.coeff[1], lda.coeff[2], length = 0.1, angle=15, col="blue")
abline(a=0, b=lda.coeff[2]/lda.coeff[1])
arrows(0, 0, pca.coeff[1], pca.coeff[2], length = 0.1, angle=15, col="black")
abline(a=0, b=pca.coeff[2]/pca.coeff[1])
```

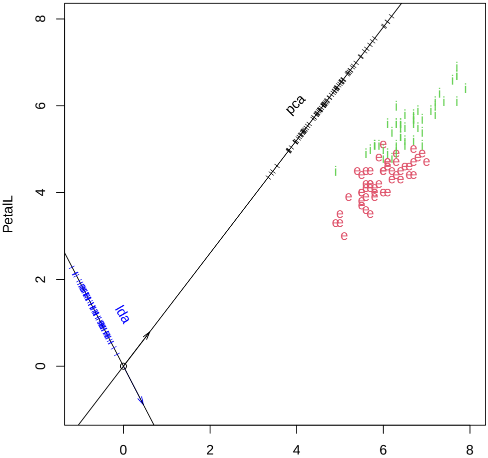
PCA and LDA: Linear Combinations



PCA and LDA: Projections

```
proj.pca=(sample12%*%pca.coeff)%*%matrix(pca.coeff, 1,2)
par(pty="s")
plot(sample12, xlim=c(-1,8), ylim=c(-1,8), xlab=xlab, ylab=ylab, type="n")
points(sample1, pch=pch[1], col=col[1])
points(sample2, pch=pch[2], col=col[2])
points(0, 0)
arrows(0, 0, lda.coeff[1], lda.coeff[2], length = 0.1, angle=15, col="blue")
abline(a=0, b=lda.coeff[2]/lda.coeff[1])
arrows(0, 0, pca.coeff[1], pca.coeff[2], length = 0.1, angle=15, col="black")
abline(a=0, b=pca.coeff[2]/pca.coeff[1])
for(i in 1: (n1+n2)){
  text(x=proj[i,1],y=proj[i,2], labels="|", col="blue",
       srt=atan(lda.coeff[2]/lda.coeff[1])*180/pi, cex=0.5)
  text(x=proj.pca[i,1],y=proj.pca[i,2], labels="|", col="black",
       srt=atan(pca.coeff[2]/pca.coeff[1])*180/pi, cex=0.5)}
text(x=0, y=1.2, "lda", srt=-60, col="blue")
text(x=4, y=6, "pca", srt=45, col="black")
```

PCA and LDA: Projections



Subsection 3

LDA in R

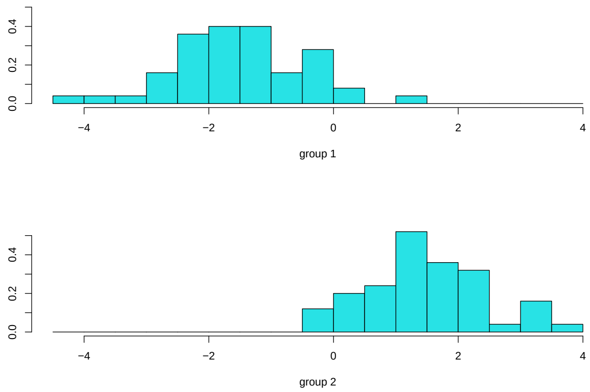
Use the “lda” function in R

```r
library(MASS)
mydata12=data.frame(G=rep(1:2, each=50),
                    x1=c(sample1[,1], sample2[,1]),
                    x2=c(sample1[,2], sample2[,2]))
obj <- lda(G ~ x1 + x2, data=mydata12)
obj
```

```
## Call:
## lda(G ~ x1 + x2, data = mydata12)
##
## Prior probabilities of groups:
##   1   2
## 0.5 0.5
##
## Group means:
##      x1    x2
## 1 5.936 4.260
## 2 6.588 5.552
##
## Coefficients of linear discriminants:
##          LD1
## x1 -1.637937
## x2  3.152368
```

Plot an “lda” Object

```r
plot(obj)
```



Subsection 4

Iris Data: Two Species, Four Features

Iris Data: Two Species, Four Features

- We will use versicolor and virginica samples
- All the four features will be used in LDA

Compute LDA

```r
sample1=iris3[,,2]#Versicolor
sample2=iris3[,,3]#Virginica
sample12=rbind(sample1, sample2)
n1=dim(sample1)[1]
n2=dim(sample2)[1]
mean.diff=c( colMeans(sample1)-colMeans(sample2) )
data.center=c( (colMeans(sample1)+colMeans(sample2))/2 )
S.pooled=((n1-1)*cov(sample1)+(n2-1)*cov(sample2))/(n1+n2-2)
lda.coeff=solve(S.pooled)%*% mean.diff
#rescale it so that is has norm 1
lda.coeff=lda.coeff/sqrt(sum(lda.coeff^2))
lda.coeff
```

```
##                [,1]
## Sepal L.  0.2268500
## Sepal W.  0.3558499
## Petal L. -0.4446115
## Petal W. -0.7900826
```

Compute Classificatino Error (Training Error)

```
m=c(t(lda.coeff)%*%data.center)
proj.scalar=(sample12%*%lda.coeff)
sum(proj.scalar[1:n1]>m)
```

```
## [1] 48
```

```
sum(proj.scalar[1:n1]>m)/n1
```

```
## [1] 0.96
```

```
sum(proj.scalar[-c(1:n2)]<m)
```

```
## [1] 49
```

```
sum(proj.scalar[-c(1:n2)]<m)/n2
```

```
## [1] 0.98
```

```
(sum(proj.scalar[1:n1]>m) + sum(proj.scalar[-c(1:n1)]<m)) /(n1+n2)
```

```
## [1] 0.97
```

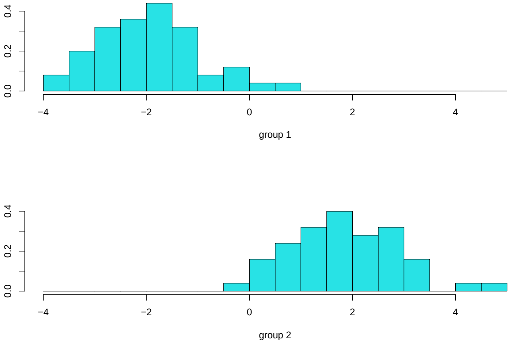
Use R to conduct LDA

```
mydata12=data.frame(G=rep(1:2, each=50),
                    x1=sample12[,1], x2=sample12[,2],
                    x3=sample12[,3], x4=sample12[,4])
obj <- lda(G ~ x1 + x2 + x3 + x4, data=mydata12)
obj
```

```
## Call:
## lda(G ~ x1 + x2 + x3 + x4, data = mydata12)
##
## Prior probabilities of groups:
##   1   2
## 0.5 0.5
##
## Group means:
##      x1    x2    x3    x4
## 1 5.936 2.770 4.260 1.326
## 2 6.588 2.974 5.552 2.026
##
## Coefficients of linear discriminants:
##           LD1
## x1 -0.9431178
## x2 -1.4794287
## x3  1.8484510
## x4  3.2847304
```

Visualize the lda object from R

```r
plot(obj)
```



Section 5

Multi-Class LDA

Three-Class Classification

- Using the same strategy, we can construct linear discriminants for a three-class problem
- How many LDs do we need for a three-class problem?
  - We can need one for classes 1 and 2, one for classes 1 and 3.
  - It can be shown that the LD for classes 2 and 3 are not necessary
- Suppose there are 3 independent random samples
  - sample sizes n_1, n_2, n_3
  - mean vectors $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\mu}_3$
  - a common covariance matrix $\boldsymbol{\Sigma}$

Subsection 1

The Linear Discriminants

The Linear Discriminants

- Sample mean vectors

$$\bar{\mathbf{X}}_1, \bar{\mathbf{X}}_2, \bar{\mathbf{X}}_3$$

- Pooled sample covariance

$$\mathbf{S}_p = \frac{(n_1 - 1)S_1 + (n_2 - 1)S_2 + (n_3 - 1)S_3}{n_1 + n_2 + n_3 - 3}$$

- Let a_{12}, a_{13}, and a_{23} denote the linear discriminants for the three pairs, respectively
- Let m_{12}, m_{13}, and m_{23} denote the projected centers

The Linear Disriminants

- Following from FLDA, we have

$$a_{ij} = \mathbf{S}_p^{-1}(\bar{\mathbf{X}}_i - \bar{\mathbf{X}}_j), m_{ij} = a_{ij}^T \frac{\bar{\mathbf{X}}_i + \bar{\mathbf{X}}_j}{2}$$

- The three linear boundaries are given by the three equations

$$f_{ij}(x) = a_{ij}^T x = m_{ij}$$

Allocate New Observations

- Let X_0 be a new observation
- We allocate X_0 to
  - class 1 if $f_{12}(X_0) > m_{12}$ and $f_{13}(X_0) > m_{13}$
  - class 2 if $f_{23}(X_0) > m_{23}$ and $f_{12}(X_0) < m_{12}$
  - class 3 if $f_{13}(X_0) < m_{13}$ and $f_{23}(X_0) < m_{23}$

Subsection 2

Minimum Distance Approach

Minimum Distance Approach

- Following the argument we used for minimum distance in the two-class problem, the allocation rule in the previous slide is equivalent to allocate X_0 to
  - class 1 if $D_{S_p}(X_0, \bar{\mathbf{X}}_1) < D_{S_p}(X_0, \bar{\mathbf{X}}_2)$ and $D_{S_p}(X_0, \bar{\mathbf{X}}_1) < D_{S_p}(X_0, \bar{\mathbf{X}}_3)$
  - class 2 if $D_{S_p}(X_0, \bar{\mathbf{X}}_2) < D_{S_p}(X_0, \bar{\mathbf{X}}_1)$ and $D_{S_p}(X_0, \bar{\mathbf{X}}_2) < D_{S_p}(X_0, \bar{\mathbf{X}}_3)$
  - class 3 if $D_{S_p}(X_0, \bar{\mathbf{X}}_3) < D_{S_p}(X_0, \bar{\mathbf{X}}_1)$ and $D_{S_p}(X_0, \bar{\mathbf{X}}_3) < D_{S_p}(X_0, \bar{\mathbf{X}}_2)$
- In summary, we allocate X_0 to the group with the minimum Mahalanobis distance.

Subsection 3

Maximum Likelihood Approach

Maximum Likelihood Approach

- Again, following the argument used in the two-class problem, we allocate X_0 to
  - class 1 if $\frac{L_1}{L_2} > 1$ and $\frac{L_1}{L_3} > 1$
  - class 2 if $\frac{L_2}{L_3} > 1$ and $\frac{L_2}{L_3} > 1$
  - class 3 if $\frac{L_3}{L_1} > 1$ and $\frac{L_3}{L_2} > 1$
- Therefore, the LDA is equivalent to the maximum likelihood approach.

Subsection 4

Measurement of Separation

The Number of Linear Disriminants

- The linear discriminants for groups (1,2) and (1,3) are:

$$(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T \mathbf{\Sigma}^{-1} x = \frac{1}{2}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T \mathbf{\Sigma}^{-1} (\bar{\mathbf{X}}_1 + \bar{\mathbf{X}}_2)$$
$$(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_3)^T \mathbf{\Sigma}^{-1} x = \frac{1}{2}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_3)^T \mathbf{\Sigma}^{-1} (\bar{\mathbf{X}}_1 + \bar{\mathbf{X}}_3)$$

- Subtracting the first equation from the second equation

$$\begin{aligned}(\bar{\mathbf{X}}_2 - \bar{\mathbf{X}}_3)^T \mathbf{\Sigma}^{-1} x &= \frac{1}{2}(\bar{\mathbf{X}}_2^T \mathbf{\Sigma}^{-1} \bar{\mathbf{X}}_2 - \bar{\mathbf{X}}_3^T \mathbf{\Sigma}^{-1} \bar{\mathbf{X}}_3) \\ &= \frac{1}{2}(\bar{\mathbf{X}}_2 - \bar{\mathbf{X}}_3)^T \mathbf{\Sigma}^{-1} (\bar{\mathbf{X}}_2 - \bar{\mathbf{X}}_3)\end{aligned}$$

- This means that the first two linear boundaries jointly imply the third one; in other words, we only need two linear discriminants