Multivariate Analysis Lecture 13: LDA for Multi-Class Problems

Zhaoxia Yu
Professor, Department of Statistics

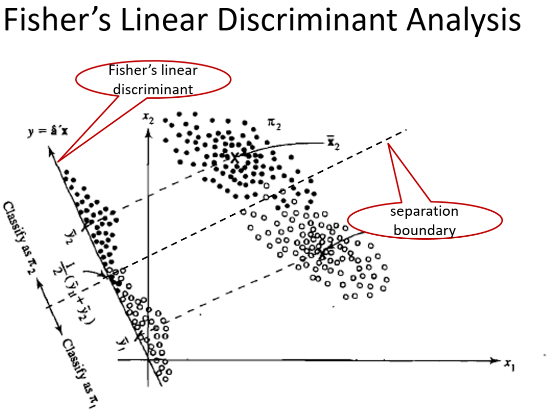
2025-05-13

Section 1

Summary of LDA

Linear Discrminant Analysis

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



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

Subsection 1

FLDA: Maximum Separability

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:

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

FLDA: Maximum Separability

- The maximization problem is

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

- Use an argument similar to PCA, such a is the first eigenvector of $\mathbf{\Sigma}^{-1}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)^T$.
- We can show that $a = \mathbf{S}_p^{-1}(\bar{\mathbf{X}}_1 - \bar{\mathbf{X}}_2)$.
- 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.

Subsection 2

Allocate New Observations

Allocate New Observations

- Consider an observation X_0. We compute

$$f(X_0) = a^T X_0$$

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

- Let

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

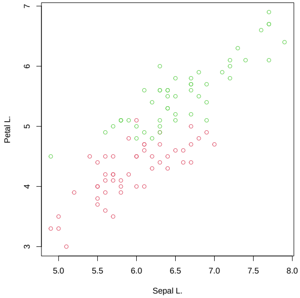
- Allocate X_0 to
    - class 1 if $f(X_0) > m$
    - class 2 if $f(x_0) < m$

Section 2

PCA vs LDA: An Example

PCA vs LDA: An Example

```
sample1=iris3[,c(1,3),2]#Versicolor
sample2=iris3[,c(1,3),3]#Virginica
sample12=rbind(sample1, sample2)
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])
```



PCA vs LDA: Compute LDA

```
n1=dim(sample1)[1]
n2=dim(sample2)[1]

#LDA
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))
m=c(t(lda.coeff)%*%data.center)
#PCA
pca.coeff=eigen(cov(sample12))$vector[,1]
#project data to LDA and PCA
proj.lda=(sample12%*%lda.coeff)%*%matrix(lda.coeff, 1,2)
proj.pca=(sample12%*%pca.coeff)%*%matrix(pca.coeff, 1,2)

lda.coeff
```

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

```
pca.coeff
```

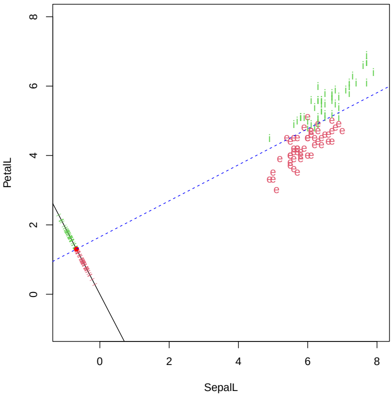
```
## [1] 0.6090576 0.7931260
```

Visualize the LDA

```
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.lda[i,1],y=proj.lda[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.lda[i,1],y=proj.lda[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)
```

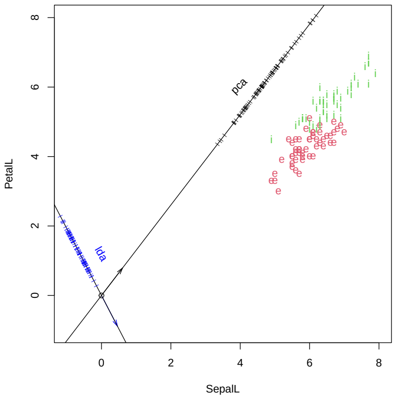
Visualize the LDA Allocations



Project Data to LDA and PCA

```
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.lda[i,1],y=proj.lda[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")
```

Project Data to LDA and PCA




```
lda.iris = lda(Species ~ ., data = iris[51:150, c(1,2,5)])
```

Section 3

Three-Class LDA

Three-Class Classification

- Using the same strategy, we can construct linear discriminants for a three-class problem
- 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.

Section 4

Example Iris

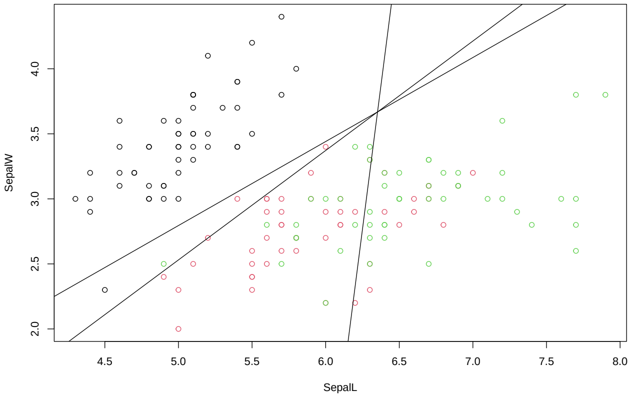
Iris Data: three species, two features: SepalL SepalW

```
i=1; j=2 #(SpealL and Sepal W)
s.pooled=((50-1)*cov(iris3[,c(i,j),1]) +
            (50-1)*cov(iris3[,c(i,j),2])+
            (50-1)*cov(iris3[,c(i,j),3]))/(150-3)
x1.bar=colMeans(iris3[,c(i,j),1])
x2.bar=colMeans(iris3[,c(i,j),2])
x3.bar=colMeans(iris3[,c(i,j),3])
a12=c(solve(s.pooled)%*%(x1.bar-x2.bar))
a13=c(solve(s.pooled)%*%(x1.bar-x3.bar))
a23=c(solve(s.pooled)%*%(x2.bar-x3.bar))
a12=a12/sqrt(sum(a12^2)); m12=c(a12%*%(x1.bar+x2.bar)/2)
a13=a13/sqrt(sum(a13^2)); m13=c(a13%*%(x1.bar+x3.bar)/2)
a23=a23/sqrt(sum(a23^2)); m23=c(a23%*%(x2.bar+x3.bar)/2)
```

Example Iris: SepalL SepalW

```
plot(iris3[,i,1], iris3[,j,1], xlim=c(min(iris3[,i,]),max(iris3[,i,])), xlab="SepalL", ylab="SepalW",
ylim=c(min(iris3[,j,]),max(iris3[,j,])))
points(iris3[,i,2], iris3[,j,2], col=2)
points(iris3[,i,3], iris3[,j,3], col=3)
abline(m12/a12[2], -a12[1]/a12[2])
abline(m13/a13[2], -a13[1]/a13[2])
abline(m23/a23[2], -a23[1]/a23[2])
```

Example Iris: SepalL SepalW



Example Iris: SepalL SepalW

```
# slopes
k12=a12[2]/a12[1]
k13=a13[2]/a13[1]
k23=a23[2]/a23[1]

# the joint point
joint.point=solve(rbind(a12,a13))%*%c(m12,m13) # point at which the three lines cross
```

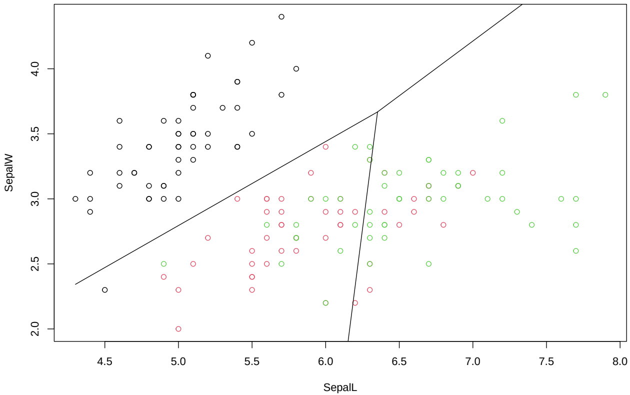
Example Iris: SepalL SepalW

```
### remove extra lines
plot(iris3[,i,1], iris3[,j,1], xlim=c(min(iris3[,i,]),max(iris3[,i,])), xlab="SepalL", ylab="SepalW",
ylim=c(min(iris3[,j,]),max(iris3[,j,])))
points(iris3[,i,2], iris3[,j,2], col=2)
points(iris3[,i,3], iris3[,j,3], col=3)
#classes 1 vs 2
lines(
c(min(iris3[,i,]), joint.point[1]),
c(1/k12*(x1.bar[1]+x2.bar[1])/2 + (x1.bar[2]+x2.bar[2])/2 -min(iris3[,i,])/k12,
1/k12*(x1.bar[1]+x2.bar[1])/2 + (x1.bar[2]+x2.bar[2])/2 -joint.point[1]/k12))

#classes 1 vs 3
lines(
c(joint.point[1],max(iris3[,i,])),
c(1/k13*(x1.bar[1]+x3.bar[1])/2 + (x1.bar[2]+x3.bar[2])/2 -joint.point[1]/k13,
1/k13*(x1.bar[1]+x3.bar[1])/2 + (x1.bar[2]+x3.bar[2])/2 -max(iris3[,i,])/k13))

#classes 2 vs 3
lines(
c(min(iris3[,i,]), joint.point[1]),
c(1/k23*(x2.bar[1]+x3.bar[1])/2 + (x2.bar[2]+x3.bar[2])/2 -min(iris3[,i,])/k23,
1/k23*(x2.bar[1]+x3.bar[1])/2 + (x2.bar[2]+x3.bar[2])/2 -joint.point[1]/k23))
```

Example Iris: SepalL SepalW



Subsection 1

The Three Boundaries Meet at Same Point

The Three Boundaries Meet at Same Point

- 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

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

- The result indicates that the three lines meet at the same point

Example Iris: SepalL SepalW: Performance

```r
lda.iris = lda(Species ~ ., data = iris[, c(1,2,5)])
con.mat=table(Pred = predict(lda.iris, iris[, c(1,2,5)])$class,
              True = iris$Species)
#Confusion Matrix
con.mat
```

```
##             True
## Pred         setosa versicolor virginica
##   setosa         49          0         0
##   versicolor      1         36        15
##   virginica       0         14        35
```

```r
#Training Error
1-sum(diag(con.mat))/sum(con.mat)
```

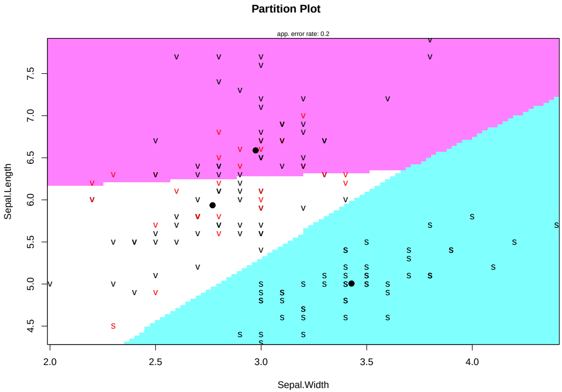
```
## [1] 0.2
```

Example Iris: Visualization

```r
library(klaR)
```

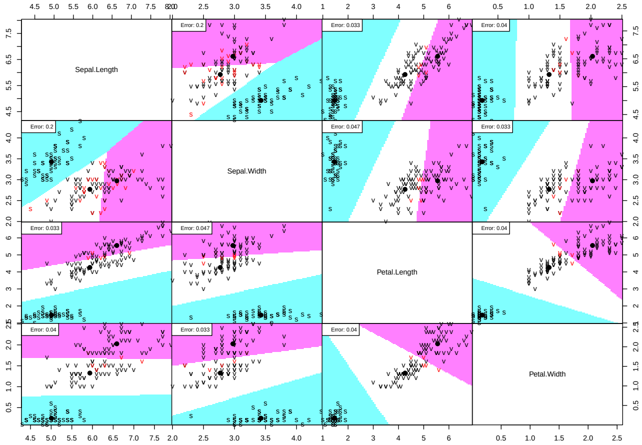
```
## Warning: package 'klaR' was built under R version 4.2.3
```

```r
partimat(Species ~ Sepal.Length + Sepal.Width, data = iris, method = "lda")
```



Example Iris: Pairwise Features

```
partimat(Species ~ ., data = iris, method = "lda", plot.matrix=TRUE)
```



Iris All Features (2 LDs): Performance

```
lda.iris = lda(Species ~ ., data = iris)
con.mat=table(Pred = predict(lda.iris, iris)$class,
              True = iris$Species)
#Confusion Matrix
con.mat
```

```
##             True
## Pred         setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48         1
##   virginica       0          2        49
```

```
#Training Error
1-sum(diag(con.mat))/sum(con.mat)
```

```
## [1] 0.02
```

Iris All Features (1st LD): Performance

```
lda.iris = lda(Species ~ ., data = iris)
con.mat=table(Pred = predict(lda.iris, iris, dimen=1)$class,
              True = iris$Species)
#Confusion Matrix
con.mat
```

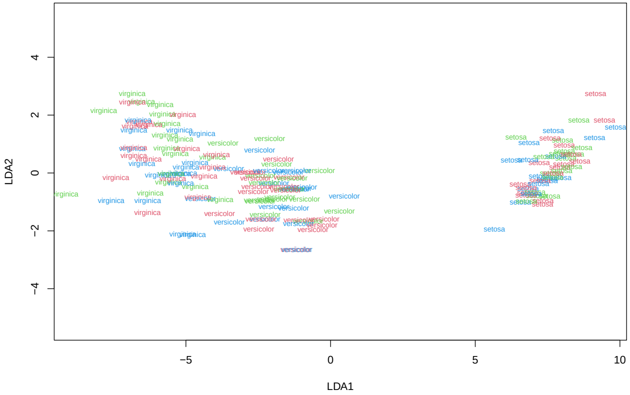
```
##             True
## Pred         setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48         0
##   virginica       0          2        50
```

```
#Training Error
1-sum(diag(con.mat))/sum(con.mat)
```

```
## [1] 0.01333333
```

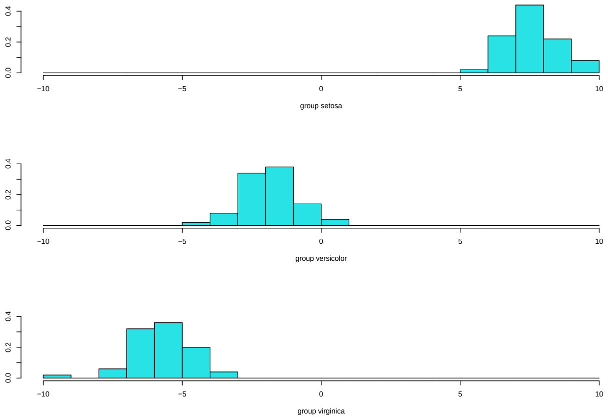
Visualize the LDA

```
lda.iris = lda(Species ~ ., data = iris)
plot(lda.iris, col=c(2,3,4), xlab="LDA1", ylab="LDA2")
```



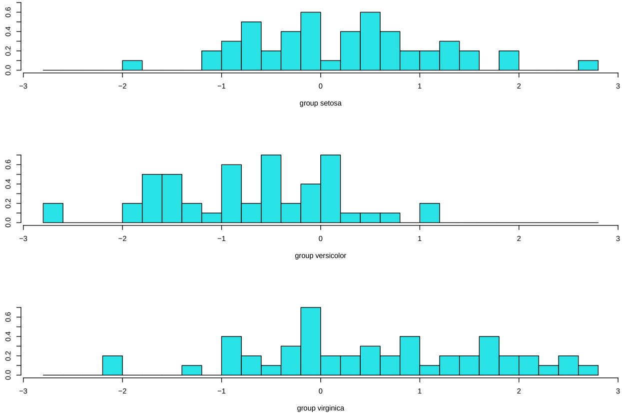
Visualize the LDA: LD1

```
lda.iris.scores=predict(lda.iris)
ldahist(lda.iris.scores$x[,1], g=iris$Species, xlab="LDA1")
```



Visualize the LDA: LD2

```
lda.iris.scores=predict(lda.iris)
ldahist(lda.iris.scores$x[,2], g=iris$Species, xlab="LDA1")
```



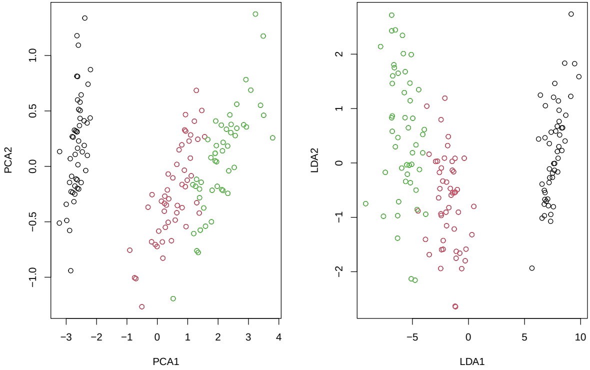
Subsection 2

Iris Data: PCA vs LDA

Iris Data: PCA vs LDA

```
pca.iris = princomp(iris[,1:4], cor = FALSE, scores = TRUE)
par(mfrow=c(1,2))
plot(pca.iris$scores[,1:2], xlab="PCA1", ylab="PCA2")
points(pca.iris$scores[51:100,1:2], col=2)
points(pca.iris$scores[101:150,1:2], col=3)
plot(predict(lda.iris, iris)$x, xlab="LDA1", ylab="LDA2")
points(predict(lda.iris, iris)$x[51:100,], col=2)
points(predict(lda.iris, iris)$x[101:150,], col=3)
```

Iris Data: PCA vs LDA



Section 5

Multi-Class LDA

Extend FLDA to g Classes

- Consider g classes. We have g independent random samples:

$$
\begin{aligned}
X_{1j} &\overset{iid}{\sim} (\mu_1, \Sigma), j = 1, \cdots, n_1 \\
X_{2j} &\overset{iid}{\sim} (\mu_2, \Sigma), j = 1, \cdots, n_2 \\
&\cdots\cdots \\
X_{gj} &\overset{iid}{\sim} (\mu_g, \Sigma), j = 1, \cdots, n_g
\end{aligned}
$$

- Want to find a linear function $Y_{ij}^{(1)} = a^T X_{ij}$ that leads to maximum separation

Quantify Separation in a g-Class Problem

- Measure separation using F statistic

$$
\begin{aligned}
F(a) &= \frac{MSB}{MSW} = \frac{SSB/(g-1)}{SSW/(n-g)} \\
&= \frac{\sum_{i=1}^{g} n_i(\bar{Y}_{i.}^{(1)} - \bar{Y}_{..}^{(1)})^2/(g-1)}{\sum_{i=1}^{g}(n_i-1)S^2_{Y_i^{(1)}}/(n-g)} \\
&= \frac{a^T \sum n_i(\bar{X}_{i.} - \bar{X}_{..})(\bar{X}_{i.} - \bar{X}_{..})^T a}{a^T \sum_{i=1}^{g}\sum_{j=1}^{n_i}(X_{ij} - \bar{X}_{i.})(X_{ij} - \bar{X}_{i.})^T a} \frac{n-g}{g-1} \\
&= \frac{a^T \mathbf{B} a}{a^T \mathbf{W} a} \frac{n-g}{g-1}
\end{aligned}
$$

where $n = \sum_{i=1}^{g} n_i$, $\mathbf{B}$ is the between-group sample covariance matrix, and $\mathbf{W}$ is the within-group sample covariance matrix.

Linear Discriminants

- The first linear discriminant is the linear function that maximizes $F(a)$. It can also be shown that the first linear discriminant is given by the first eigenvector of $\mathbf{W}^{-1}\mathbf{B}$, i.e.,

$$Y_{ij}^{(1)} = \gamma_1^T X_{ij}$$

  where γ_1 is the first eigenvector of $\mathbf{W}^{-1}\mathbf{B}$.
- Similarly, for $k = 1, \cdots, rank(\mathbf{B})$, the kth linear discriminant is given by the kth eigenvector of $\mathbf{W}^{-1}\mathbf{B}$

$$Y_{ij}^{(k)} = \gamma_k^T X_{ij}$$

Use the Linear Discriminants

- Allocate an observation to the group with the minimum distance defined by the Euclidean distance in space spanned by the linear discriminants.
- Let X_0 be a new observation.
- Calculate $Y_0^{(k)} = \gamma_k^T X_0$, the projection of X_0 to the kth linear discriminant for $k = 1, \cdots, rank(B)$.
- Calculate the distance between $(Y_0^{(1)}, \cdots, Y_0^{(rank(B))})$ and $(\bar{Y}_{i.}^{(1)}, \cdots, \bar{Y}_{i.}^{(rank(B))})$

$$D^2(X_0, i) = \sum_{k=1}^{rank(B)} [Y_0^{(k)} - \bar{Y}_{i.}^{(k)}]^2$$

- Allocate X_0 to

$$\underset{i}{\operatorname{argmin}} D^2(X_0, i)$$

The Number of Linear Discriminants

- Recall that $rank(B) = min(p, g-1)$ (Lecture 8). The number of total linear discriminants is $min(p, g-1)$.
- In the two-class, $rank(B) = 1$. Thus, there is only one linear discrminant.
- When $rank(B)$ is large, it is often helpful to use the top linear discriminants.
- A scree plot of the eigenvalues of $\mathbf{W}^{-1}\mathbf{B}$ can be used to find an elbow point, if there is one

Section 6

Example: Glasses

Example: Glasses

```
#install.packages("mlbench")
data(Glass, package = "mlbench")
table(Glass$Type)

glass_subset <- Glass[Glass$Type %in% c(1,2,6,7), ]
glass_subset$Type <- factor(glass_subset$Type)
# Fit LDA model
lda_model <- lda(Type ~ ., data = glass_subset)

# Print LDA results
lda_model

# Plot LDA results
plot(lda_model)
```

Section 7

Further Reading/Topics

Further Reading/Topics

- Cross validation in LDA
  - the lda function in R has an option to do leave one out cross validation
- Better visualization of classification results
  - R package: WeDiBaDis
  - https://uw.pressbooks.pub/appliedmultivariatestatistics/chapter/discriminant-analysis/
- LDA with unequal covariance matrices
- Quadratic Discriminant Analysis (QDA)

Section 8

Additional Example: Crude Oil

Additional Example: Crude Oil

```
urlfile='https://raw.githubusercontent.com/yu-zhaoxia/teaching-multivariate/d87ce8b30ecb15fc09aa543047b3b0
co=read.table(urlfile, header=F)
names(co)=c("V1","V2","V3","V4","V5","type")
```

- Exploratory analysis shows that transformation might be helpful

```
co[,2]=sqrt(co[,2])
co[,3]=sqrt(co[,3])
co[,4]=1/co[,4]
```

```
obj=lda(type~V1+V2+V3+V4+V5, data=co)
ld=predict(obj, co)$x
ld1=ld[,1]
ld2=ld[,2]

T=(56-1)*cov(co[,1:5])
W=(7-1)*cov(co[1:7,1:5]) +(11-1)*cov(co[8:18,1:5])+
  (38-1)*cov(co[19:56,1:5])
B=T-W
lambda=eigen(solve(W)%*%B)$values
lambda[abs(lambda)<1.0e-10]=0
lambda=Re(lambda)
```

Example Crude Oil

```r
par(mfrow=c(2,2))
par(yaxt="n")
plot(ld1, rep(0,56), xlab="LD1", ylab="", type="n")
points(ld1[1:7], rep(0, 7), pch="w", col=1)
points(ld1[8:18], rep(0, 11), pch="s", col=2)
points(ld1[19:56], rep(0, 38), pch="u", col=3)

plot(ld2, rep(0,56), xlab="LD2", ylab="", type="n")
points(ld2[1:7], rep(0, 7), pch="w", col=1)
points(ld2[8:18], rep(0, 11), pch="s", col=2)
points(ld2[19:56], rep(0, 38), pch="u", col=3)

par(yaxt="t")
plot(ld1, ld2, xlab="LD1", ylab="LD2", type="n")
points(ld1[1:7], ld2[1:7], pch="w", col=1)
points(ld1[8:18], ld2[8:18], pch="s", col=2)
points(ld1[19:56], ld2[19:56], pch="u", col=3)

#### scree plot
plot(lambda, type="b", ylab="", main="eigenvalues of BW^{-1}")
```

Example Crude Oil