Linear discriminant analysis (LDA) and quadratic discriminant analysis (PCR)

Søren Højsgaard

Thu Sep 26 13:24:02 2024

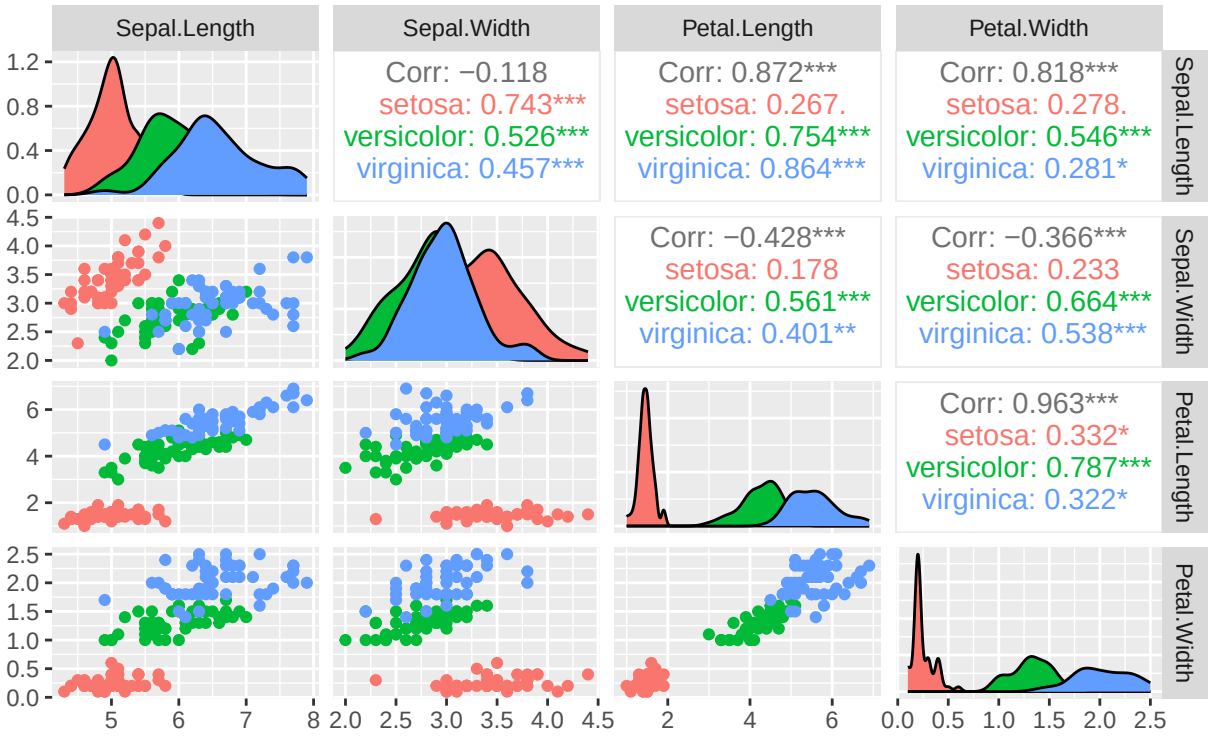
Contents

1 Introduction

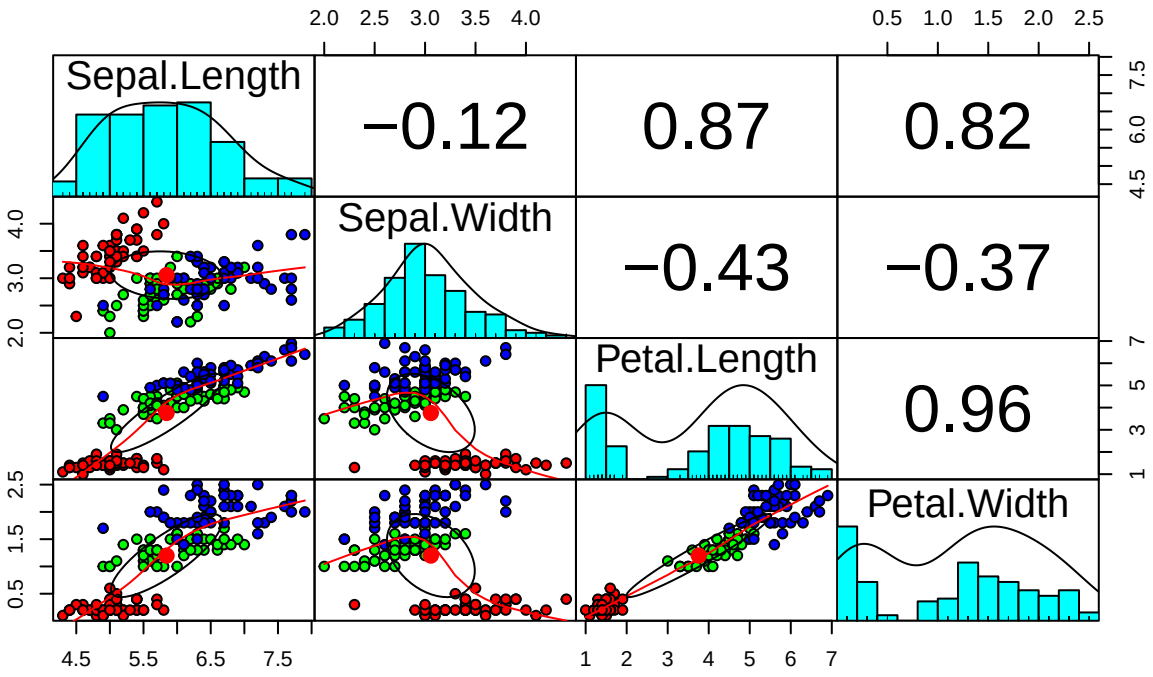
1. Linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA) are two classical methods for classification.
2. They fall into the category of supervised learning methods. Requires a training set with known class labels. Based on this training set, the methods learn to classify new observations.
3. The methods are based on the assumption that the data are normally distributed.

1.1 Example: Iris data

```
library(GGally)
ggpairs(iris, columns = 1:4, mapping = aes(color = Species), progress=FALSE)
```




```
library(psych)
pairs.panels(iris[1:4],
             gap = 0,
             bg = c("red", "green", "blue")[iris$Species],
             pch = 21)
```



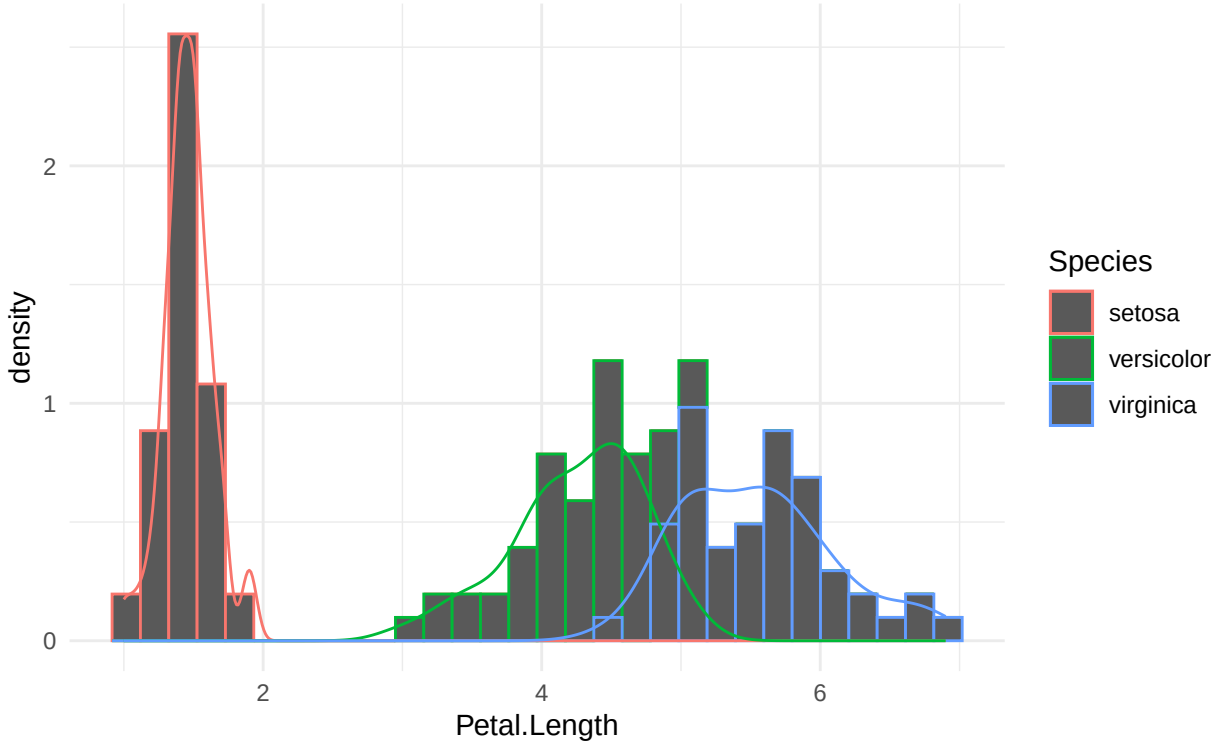
1.2 The very idea of LDA / QDA

1. The idea of LDA is to find a linear combination (weighed sum) of the predictors that best separates the classes.
2. The linear combination is chosen such that the between-class variance is maximized and the within-class variance is minimized.

3. The linear combination is called the discriminant function.
4. Allocate a new observation to the most likely class.
5. QDA is a generalization of LDA that allows for different covariance matrices for the classes.

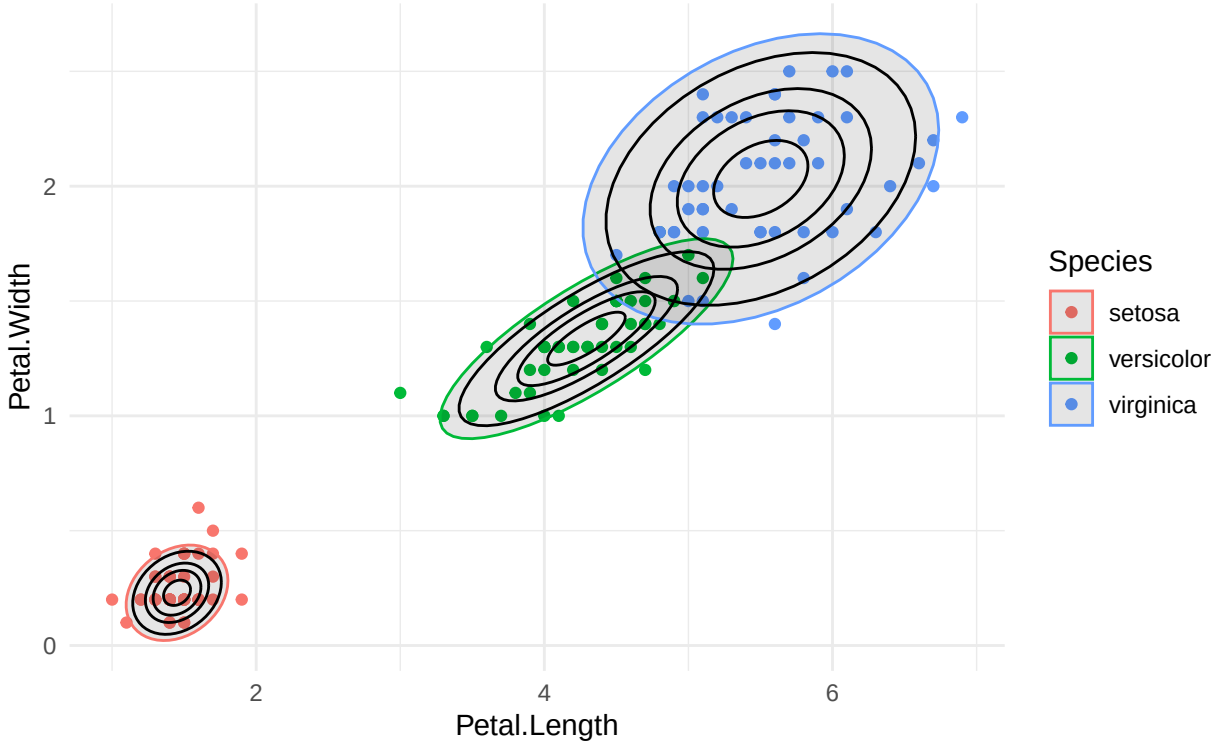
In one dimension:

```
iris |>
  ggplot(aes(x=Petal.Length, y = after_stat(density), color=Species)) +
  geom_histogram() + geom_density() +
  theme_minimal()
```



In two dimensions

```
iris |>
  ggplot(aes(x=Petal.Length, y=Petal.Width, color=Species, group=Species)) +
  geom_point() +
    stat_ellipse(geom = "polygon", fill=1, alpha=0.1) +
    stat_ellipse(level = 0.2, color=1) +
    stat_ellipse(level = 0.5, color=1) +
    stat_ellipse(level = 0.7, color=1) +
    stat_ellipse(level = 0.9, color=1) +
    theme_minimal()
```



---

2 LDA in R

```
set.seed(2024)
idx <- sample(nrow(iris), 0.5*nrow(iris))
train <- iris[idx,]
test <- iris[-idx,]
```

```
library(MASS)
lda.fit <- lda(Species ~ ., data=train)
```

```
pr <- predict(lda.fit, newdata=train)
pr$class |> head(4)
```

```
## [1] versicolor setosa     setosa     virginica
## Levels: setosa versicolor virginica
```

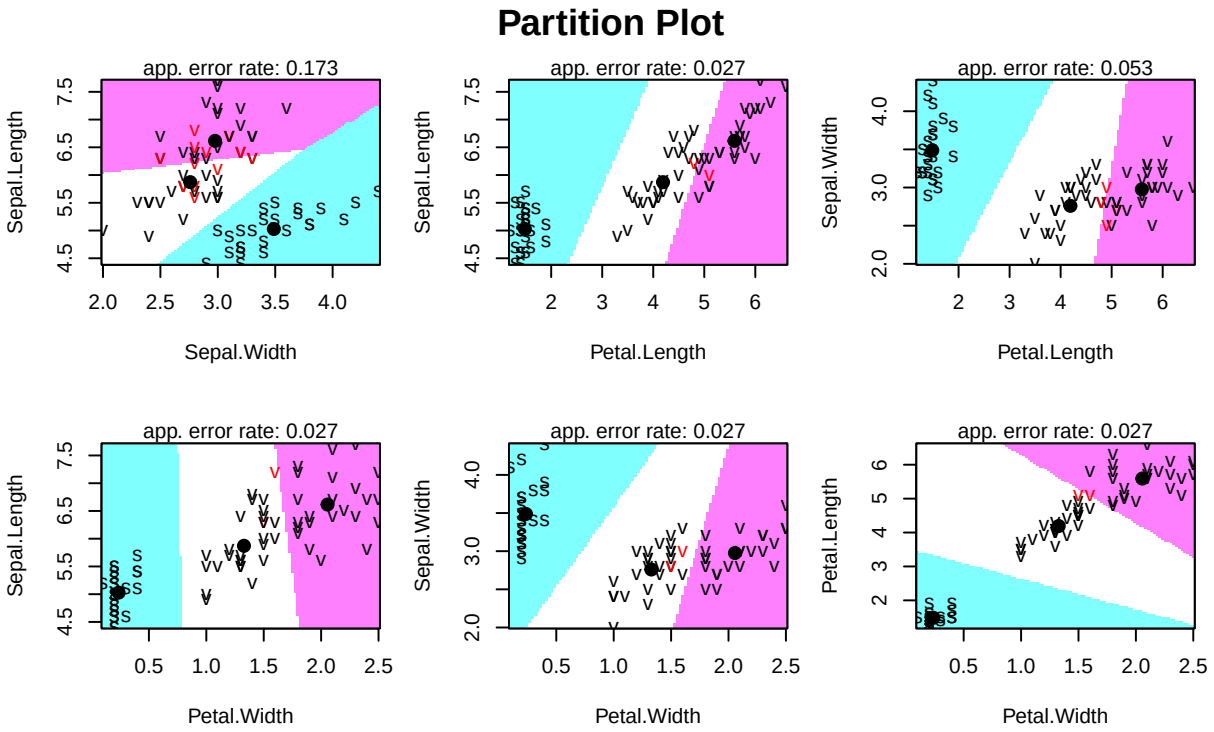
```
pr$posterior |> zapsmall() |> head(4)
```

```
##     setosa versicolor virginica
## 66       0          1         0
## 37       1          0         0
## 45       1          0         0
## 145      0          0         1
```

2.1 Visualization

In LDA, the feature space is separated into planes (regions separated by hyperplanes) that are perpendicular to the discriminant function.

```
klaR::partimat(Species ~ ., data=train, method="lda")
```



2.2 Prediction

Internal:

```
p1 <- predict(lda.fit, newdata=train)$class
tab <- table(Actual = train$Species, Predicted = p1)
tab
```

```
##             Predicted
## Actual       setosa versicolor virginica
##   setosa         25          0         0
##   versicolor      0         24         1
##   virginica       0          1        24
```

```
sum(diag(tab))/sum(tab)
```

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

External:

```
p2 <- predict(lda.fit, newdata=test)$class
tab1 <- table(Actual = test$Species, Predicted = p2)
tab1
```

```
##             Predicted
## Actual       setosa versicolor virginica
##   setosa         25          0         0
##   versicolor      0         25         0
##   virginica       0          0        25
```

```
sum(diag(tab1))/sum(tab1)
```

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

---

3 QDA in R

If variances differ between classes, LDA may not be the best choice. Instead, we can use QDA. While LDA assumes that the covariance matrix is the same for all classes, QDA allows for different covariance matrices.

```
qda.fit <- qda(Species ~ ., data=train)
```

```
pr <- predict(qda.fit, newdata=train)
pr$class |> head(4)
```

```
## [1] versicolor setosa     setosa     virginica
## Levels: setosa versicolor virginica
```

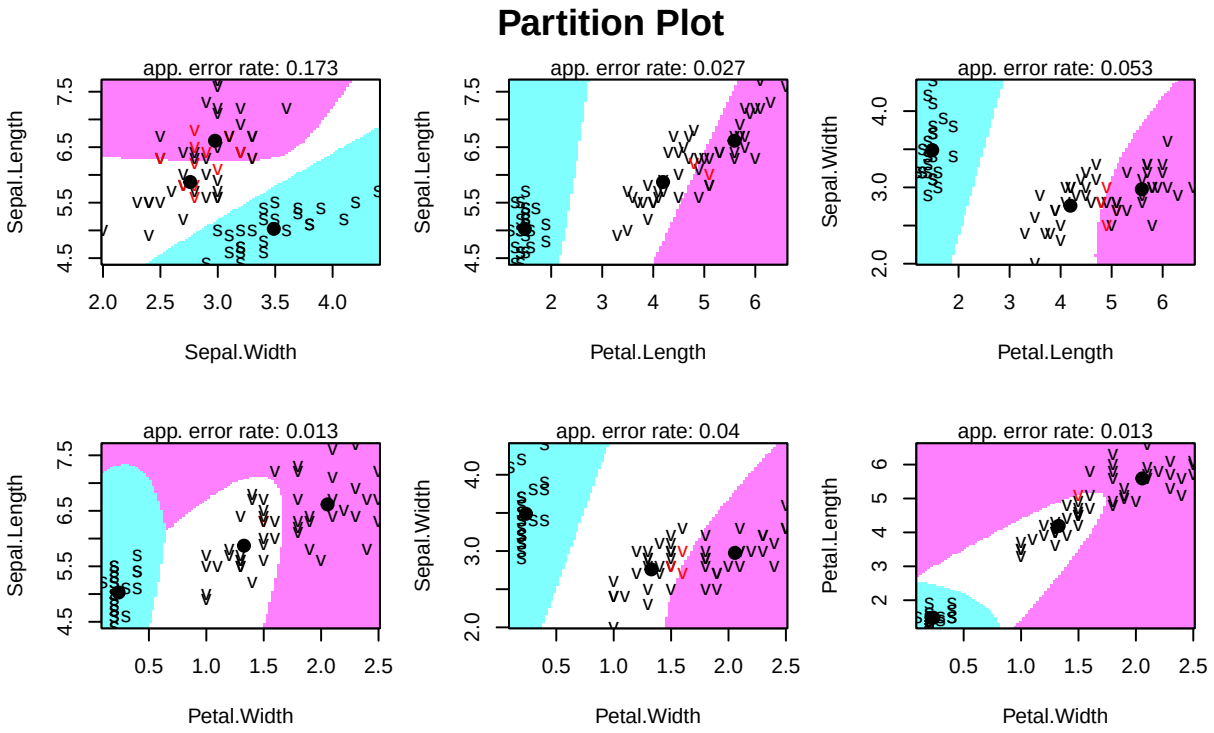
```
pr$posterior |> zapsmall() |> head(4)
```

```
##     setosa versicolor virginica
## 66       0          1         0
## 37       1          0         0
## 45       1          0         0
## 145      0          0         1
```

3.1 Visualization

In QDA, the feature space is separated into more complicated regions

```
klaR::partimat(Species ~ ., data=train, method="qda")
```



3.2 Prediction

Internal:

```
p1 <- predict(qda.fit, newdata=train)$class
tab <- table(Actual = train$Species, Predicted = p1)
tab
```

```
##             Predicted
## Actual       setosa versicolor virginica
##   setosa         25          0         0
##   versicolor      0         24         1
##   virginica       0          0        25
```

```
sum(diag(tab))/sum(tab)
```

```
## [1] 0.99
```

External:

```
p2 <- predict(qda.fit, newdata=test)$class
tab1 <- table(Actual = test$Species, Predicted = p2)
tab1
```

```
##             Predicted
## Actual       setosa versicolor virginica
```

```
##   setosa          25          0         0
##   versicolor       0         24         1
##   virginica        0          0        25
```

```
sum(diag(tab1))/sum(tab1)
```

```
## [1] 0.99
```

3.3 Example: The cancer data

```
BC <- read_delim("https://asta.math.aau.dk/datasets?file=BC0.dat",
col_types = cols(Class = col_factor()))

BC |> head(4)
```

```
## # A tibble: 4 x 6
##   nuclei cromatin Size.low Size.medium Shape.low Class
##    <dbl>    <dbl> <lgl>    <lgl>       <lgl>     <fct>
## 1      1        3 TRUE     FALSE       TRUE      benign
## 2     10        3 FALSE    TRUE        FALSE     benign
## 3      2        3 TRUE     FALSE       TRUE      benign
## 4      4        3 FALSE    FALSE       FALSE     benign
```

```
set.seed(2024)
i <- sample(nrow(BC), 0.5*nrow(BC))
BCtrain <- BC[i,]
BCtest <- BC[-i,]
lda.fit <- lda(Class ~ ., data = BCtrain)
qda.fit <- qda(Class ~ ., data = BCtrain)
```

Notice how factors are handled: Recoded as dummy variables:

```
BCtrain |> head(4)
```

```
## # A tibble: 4 x 6
##   nuclei cromatin Size.low Size.medium Shape.low Class
##    <dbl>    <dbl> <lgl>    <lgl>       <lgl>     <fct>
## 1     10        4 FALSE    TRUE        FALSE     malignant
## 2      1        2 TRUE     FALSE       TRUE      benign
## 3     10        9 FALSE    FALSE       FALSE     malignant
## 4     10        7 FALSE    FALSE       FALSE     malignant
```

```
model.matrix(Class ~ ., data = BCtrain) |> head(4)
```

```
##   (Intercept) nuclei cromatin Size.lowTRUE Size.mediumTRUE Shape.lowTRUE
## 1           1     10        4            0               1             0
## 2           1      1        2            1               0             1
## 3           1     10        9            0               0             0
## 4           1     10        7            0               0             0
```

```
p1 <- predict(lda.fit, BCtrain)$class
p2 <- predict(lda.fit, BCtest)$class

tab1 <- table(Actual = BCtrain$Class, Predicted = p1)
tab1
```

```
##            Predicted
## Actual      benign malignant
##   benign       202         8
##   malignant      5       126
```

```
tab2 <- table(Actual = BCtest$Class, Predicted = p2)
tab2
```

```
##            Predicted
## Actual      benign malignant
##   benign       228         6
##   malignant      7       101
```

```r
p1 <- predict(qda.fit, BCtrain)$class
p2 <- predict(qda.fit, BCtest)$class

tab1 <- table(Actual = BCtrain$Class, Predicted = p1)
tab1
```

```
##            Predicted
## Actual      benign malignant
##   benign       200        10
##   malignant      4       127
```

```r
tab2 <- table(Actual = BCtest$Class, Predicted = p2)
tab2
```

```
##            Predicted
## Actual      benign malignant
##   benign       223        11
##   malignant      4       104
```

3.4 Exercise

Consider the breast cancer data set from the doBy package: There are measurents of gene expression for 1000 genes for 85 patients. The patients are divided into two groups: 1) patients with breast cancer and 2) patients without breast cancer. The last column of the data set contains the group membership.

Is it possible to classify the patients into the two groups based on the gene expression data using LDA or QDA? Compare the results of the two methods (how well is an external dataset classified) and discuss the results.

```r
dat <- doBy::breastcancer
dim(dat)
```

```
## [1]  250 1001
```

```r
dat$code |> table()
```

```
##
##    case control
##      58     192
```

```r
dat$code |> head()
```

```
## [1] control control control control control control
## Levels: case control
```

```r
dat$code |> tail()
```

```
## [1] case case case case case case
## Levels: case control
```

Hint: If we do as above we get a warning message. This is because the data set is too small relative to the number of predictors (there are 1000)

```r
set.seed(2024)
idx <- sample(nrow(dat), size=0.8*nrow(dat))
train <- dat[idx,]
test <- dat[-idx,]
lda.fit <- lda(code~., data=train)
```

```
## Warning in lda.default(x, grouping, ...): variables are collinear
```

How can we move ahead?