## Fitting Linear Discriminant Analysis and Quadratic Discriminant Analysis

To fit an LDA model, we use the lda() function from the MASS package.

**library**(MASS)

data(iris)

iris\_lda = **lda**(Species ~ ., data = iris\_trn)

iris\_lda

## Call:

## lda(Species ~ ., data = iris\_trn)

##

## Prior probabilities of groups:

## setosa versicolor virginica

## 0.3733333 0.3200000 0.3066667

##

## Group means:

## Sepal.Length Sepal.Width Petal.Length Petal.Width

## setosa 4.978571 3.378571 1.432143 0.2607143

## versicolor 5.995833 2.808333 4.254167 1.3333333

## virginica 6.669565 3.065217 5.717391 2.0956522

##

## Coefficients of linear discriminants:

## LD1 LD2

## Sepal.Length 0.7100013 -0.8446128

## Sepal.Width 1.2435532 2.4773120

## Petal.Length -2.3419418 -0.4065865

## Petal.Width -1.8502355 2.3234441

##

## Proportion of trace:

## LD1 LD2

## 0.9908 0.0092

Here we see the estimated ^πkπ^k and ^μkμ^k for each class.

**is.list**(**predict**(iris\_lda, iris\_trn))

## [1] TRUE

**names**(**predict**(iris\_lda, iris\_trn))

## [1] "class" "posterior" "x"

**head**(**predict**(iris\_lda, iris\_trn)$class, n = 10)

## [1] setosa virginica setosa setosa virginica setosa

## [7] virginica setosa versicolor setosa

## Levels: setosa versicolor virginica

**head**(**predict**(iris\_lda, iris\_trn)$posterior, n = 10)

## setosa versicolor virginica

## 23 1.000000e+00 1.517145e-21 1.717663e-41

## 106 2.894733e-43 1.643603e-06 9.999984e-01

## 37 1.000000e+00 2.169066e-20 1.287216e-40

## 40 1.000000e+00 3.979954e-17 8.243133e-36

## 145 1.303566e-37 4.335258e-06 9.999957e-01

## 36 1.000000e+00 1.947567e-18 5.996917e-38

## 119 2.220147e-51 9.587514e-09 1.000000e+00

## 16 1.000000e+00 5.981936e-23 1.344538e-42

## 94 1.599359e-11 9.999999e-01 1.035129e-07

## 27 1.000000e+00 8.154612e-15 4.862249e-32

As we should come to expect, the predict() function operates in a new way when called on an lda object. By default, it returns an entire list. Within that list class stores the classifications and posterior contains the estimated probability for each class.

iris\_lda\_trn\_pred = **predict**(iris\_lda, iris\_trn)$class

iris\_lda\_tst\_pred = **predict**(iris\_lda, iris\_tst)$class

We store the predictions made on the train and test sets.

calc\_class\_err = **function**(actual, predicted) {

**mean**(actual != predicted)

}

**calc\_class\_err**(predicted = iris\_lda\_trn\_pred, actual = iris\_trn$Species)

## [1] 0.04

**calc\_class\_err**(predicted = iris\_lda\_tst\_pred, actual = iris\_tst$Species)

## [1] 0.01333333

As expected, LDA performs well on both the train and test data.

**table**(predicted = iris\_lda\_tst\_pred, actual = iris\_tst$Species)

## actual

## predicted setosa versicolor virginica

## setosa 22 0 0

## versicolor 0 26 1

## virginica 0 0 26

Looking at the test set, we see that we are perfectly predicting both setosa and versicolor. The only error is labeling a virginica as a versicolor.

iris\_lda\_flat = **lda**(Species ~ ., data = iris\_trn, prior = **c**(1, 1, 1) / 3)

iris\_lda\_flat

## Call:

## lda(Species ~ ., data = iris\_trn, prior = c(1, 1, 1)/3)

##

## Prior probabilities of groups:

## setosa versicolor virginica

## 0.3333333 0.3333333 0.3333333

##

## Group means:

## Sepal.Length Sepal.Width Petal.Length Petal.Width

## setosa 4.978571 3.378571 1.432143 0.2607143

## versicolor 5.995833 2.808333 4.254167 1.3333333

## virginica 6.669565 3.065217 5.717391 2.0956522

##

## Coefficients of linear discriminants:

## LD1 LD2

## Sepal.Length 0.7136357 -0.8415442

## Sepal.Width 1.2328623 2.4826497

## Petal.Length -2.3401674 -0.4166784

## Petal.Width -1.8602343 2.3154465

##

## Proportion of trace:

## LD1 LD2

## 0.9901 0.0099

Instead of learning (estimating) the proportion of the three species from the data, we could instead specify them ourselves. Here we choose a uniform distributions over the possible species. We would call this a “flat” prior.

iris\_lda\_flat\_trn\_pred = **predict**(iris\_lda\_flat, iris\_trn)$class

iris\_lda\_flat\_tst\_pred = **predict**(iris\_lda\_flat, iris\_tst)$class

**calc\_class\_err**(predicted = iris\_lda\_flat\_trn\_pred, actual = iris\_trn$Species)

## [1] 0.04

**calc\_class\_err**(predicted = iris\_lda\_flat\_tst\_pred, actual = iris\_tst$Species)

## [1] 0

This actually gives a better test accuracy!

**11.2 Quadratic Discriminant Analysis**

QDA also assumes that the predictors are multivariate normal conditioned on the classes.

X∣Y=k∼N(μk,Σk)X∣Y=k∼N(μk,Σk)

fk(x)=1(2π)p/2|Σk|1/2exp[−12(x−μk)′Σ−1k(x−μk)]fk(x)=1(2π)p/2|Σk|1/2exp⁡[−12(x−μk)′Σk−1(x−μk)]

Notice that now ΣkΣk **does** depend on kk, that is, we are allowing a different ΣkΣk for each class. We only use information from class kk to estimate ΣkΣk.

iris\_qda = **qda**(Species ~ ., data = iris\_trn)

iris\_qda

## Call:

## qda(Species ~ ., data = iris\_trn)

##

## Prior probabilities of groups:

## setosa versicolor virginica

## 0.3733333 0.3200000 0.3066667

##

## Group means:

## Sepal.Length Sepal.Width Petal.Length Petal.Width

## setosa 4.978571 3.378571 1.432143 0.2607143

## versicolor 5.995833 2.808333 4.254167 1.3333333

## virginica 6.669565 3.065217 5.717391 2.0956522

Here the output is similar to LDA, again giving the estimated ^πkπ^k and ^μkμ^k for each class. Like lda(), the qda() function is found in the MASS package.

Consider trying to fit QDA again, but this time with a smaller training set. (Use the commented line above to obtain a smaller test set.) This will cause an error because there are not enough observations within each class to estimate the large number of parameters in the ΣkΣk matrices. This is less of a problem with LDA, since all observations, no matter the class, are being use to estimate the shared ΣΣ matrix.

iris\_qda\_trn\_pred = **predict**(iris\_qda, iris\_trn)$class

iris\_qda\_tst\_pred = **predict**(iris\_qda, iris\_tst)$class

The predict() function operates the same as the predict() function for LDA.

**calc\_class\_err**(predicted = iris\_qda\_trn\_pred, actual = iris\_trn$Species)

## [1] 0.01333333

**calc\_class\_err**(predicted = iris\_qda\_tst\_pred, actual = iris\_tst$Species)

## [1] 0.04

**table**(predicted = iris\_qda\_tst\_pred, actual = iris\_tst$Species)

## actual

## predicted setosa versicolor virginica

## setosa 22 0 0

## versicolor 0 23 0

## virginica 0 3 27

Here we find that QDA is not performing as well as LDA. It is misclassifying versicolors. Since QDA is a more complex model than LDA (many more parameters) we would say that QDA is overfitting here.

Also note that, QDA creates quadratic decision boundaries, while LDA creates linear decision boundaries. We could also add quadratic terms to LDA to allow it to create quadratic decision boundaries.