Tutorial 7: Linear Discriminant Function Analysis

Worked Example 1: Egyptian Skulls

Work through the Egyptian Skulls example in the lecture notes. The data is in the file `akulls.dat` in the course data directory.

1. Perform a Linear Discriminant Analysis. What are the discriminant functions?
2. Construct a table indicating the classification of the skulls from each period.
3. Repeat using a jacknife 'leave-one-out' method for constructing the classifier. Are the results substantially different?
4. Perform a quadratic discriminant analysis. Are the results better or worse?

Worked Example 2: Diabetes

Consider the diabetes data in `diabetes.txt`. Ignore the first 4 columns; they do not contain useful information. The three primary variables are `glucose area` (measure of glucose intolerance), `insulin area` (measure of insulin response to oral glucose), `SSPG` (steady state plasma glucose - a measure of insulin resistance). In addition, `relative weight` and `fasting plasma glucose` were measured. The three clinical classifications (the target variable) are: overt diabetic (Class 1, 33 individuals), chemical diabetic (Class 2, 36) and normal (Class 3, 76).

Draw a scatterplot matrix of all 5 variables with different colours or symbols representing the three classes of diabetes. Do these pairwise plots suggest multivariate Gaussian distributions for each class with equal covariance matrices? Carry out an LDA and draw a 2D scatterplot of the first two discriminating functions. Using the leave-one-out cross validation procedure, find the confusion table (how many from each class have been classified according to each class) and identify those observations which have been wrongly classified according to the LDA rule. Do the same for QDA.

Hint You'll find the package **GGAlly** useful for the scatterplot matrix. For adding colours, the package seems to like factors, so apply `as.factor()` to the column that represents the different colours (different classes of diabetes).

You'll see clearly how using two discriminant functions helps with the classification in this example.

Solution Firstly, we need to get the data.

```
www2 <-
"https://www.mimuw.edu.pl/~noble/courses/MultivariateStatistics/data/
diabetes.txt"
diabetes <-read.table(www2,header=F)
```

This data doesn't have column headers, so let's add them:

```
>
colnames(diabetes)<-c("one","two","three","four","glucArea","InsArea",
"SSPG","relweight","FPG","type")
```

Later, we'll discover that **ggplot2** and **GGally** need the class variable to be a *factor* and not a *numerical* variable. We can see the type of each variable as follows:

```
> sapply(diabetes,class)
      one       two     three      four  glucArea   InsArea      SSPG
relweight       FPG
"integer" "integer" "integer" "integer" "numeric" "integer" "integer"
"integer" "integer"
     type
"integer"
```

To use **GGally**, we need the *class* variable to be a *factor* and we can do this as follows:

```
> diabetes$type <-as.factor(diabetes$type)
```

Now activate the libraries with the appropriate graph packages;

```
> library(GGally)
Loading required package: ggplot2
Registered S3 method overwritten by 'GGally':
  method from
  +.gg   ggplot2
> library(ggplot2)
```

Now we would like to visually examine pairs of variables to see if there are good pairs that will help with classification. Scatterplots are useful here. For example, `Glucose area` versus `Insulin area` where the points are coloured according to `type` of diabetes is done as follows:

```
> ggplot(diabetes,aes(x=glucArea,y=InsArea,color=type))+geom_point()
```

Look at the plot. We can see that even with these two variables, we can make a reasonable job of classification; type 3 diabetes can be determined by the `InsArea` variable, although the difference between types 1 and 2 is not so obvious from this pair of variables.

From `GGally`, we can get scatter plots of all pairs of variables, coloured by `type` as follows:

```
> ggpairs(data=diabetes,columns=5:9,aes(colour=type))
```

To do LDA, we need the package **MASS**

```
> library(MASS)
> discrim <-
lda(type~glucArea+InsArea+SSPG+relweight+FPG,data=diabetes,na.action=
"na.omit",CV=FALSE)
> discrim
Call:
lda(type ~ glucArea + InsArea + SSPG + relweight + FPG, data =
diabetes,
    CV = FALSE, na.action = "na.omit")

Prior probabilities of groups:
        1         2         3
0.2275862 0.2482759 0.5241379

Group means:
   glucArea   InsArea      SSPG relweight      FPG
1 0.9839394 217.66667 1043.7576  106.0000 318.8788
2 1.0558333  99.30556  493.9444  288.0000 208.9722
3 0.9372368  91.18421  349.9737  172.6447 114.0000

Coefficients of linear discriminants:
                    LD1          LD2
glucArea  -1.3624356881 -3.784142444
InsArea    0.0336487883  0.036633317
SSPG      -0.0125763942 -0.007092017
relweight  0.0001022245 -0.006173424
FPG       -0.0042431866  0.001134070

Proportion of trace:
   LD1    LD2
0.8812 0.1188
```

Note that if we do not assign prior probabilities, it simply takes the proportion of each class in the training set.

Here we see that there are *two* discriminant functions and we make a scatterplot as follows:

```
> discrim.lda <-predict(discrim,diabetes[,5:9])
```

This applies the discriminant classifier to the data. This is (of course) unsound, because we've used the same data to construct the classifier as we are using to do the classification!

The loadings for the discriminant functions for this data set are found under $\$x$. We add the two discriminant function loadings to the data frame:

```
> diabetes$lda1 <- discrim.lda$x[,1]
> diabetes$lda2<-discrim.lda$x[,2]
> ggplot(diabetes,aes(x=lda1,y=lda2,color=type))+geom_point()
```

This puts the scores of the discriminant functions as additional columns of the data frame. We can clearly see that, using *both* discriminant functions, we can determine, with good precision, the type of diabetes from these variables.

To see how well the classifier is doing (hoping that using the same data to learn and to test doesn't make too much difference):

```
> pred<-predict(discrim,diabetes[,5:9])
> tab <- table(diabetes$type,pred$class)
> tab

     1  2  3
  1 27  5  1
  2  0 31  5
  3  0  3 73
```

Of course, to get a more accurate idea of the performance, use `CV = TRUE` to ensure that, for each observation, the observation is not used in the construction of the classifier.

Exercises

1. Perform a linear discriminant analysis on the Swiss bank note data. This is found in the file `swisslab.dat` in the data directory of the course page. It coutains various measurements for 200 Swiss bank notes, 100 of which are genuine and the other 100 forged. How many are correctly classified from real and forged respectively?

2. The `wine` data set, found in `winetrain.txt` for easy loading into R and `winetrain.xls`, with additional information about the variables, are the results of chemical analysis of 178 wines grown over the decade 1970-79 in the same region of Italy, but derived from different cultivars (Barolo, Grignolino, Barbera). Compute a LDA, draw a 2d scatterplot of the first two LDF (linear discriminant function) co-ordinates and colour-code the points by wine type. Are there any interesting features?

3. Consider `The Insurance Company Benchmark` data. There are 86 variables on product-usage data and socio-demographic data for customers of an insurance company. The file `ticdata2000.txt` is a learning set of 5822 customers; the file `ticeval2000.txt` is a test set of 4000 customers. A description of the variables may be found here:

   `http://kdd.ics.uci.edu/databases/tic/dictionary.txt`

   More information may be found here:

   `http://kdd.ics.uci.edu/databases/tic/tic.data.html`

   The target variables is variable 86: 'CARAVAN:Number of mobile home policies'.

   The problem is to predict who in the test set would be interested in buying a caravan insurance policy. Use the classification methods you have encountered in the course so far. Apply them to the learning set to construct classification method and then apply them to the test set. Compare your predictions for the test set with those given in the file `tictgts2000.txt` and estimate the test set error rate. Which variables (or linear combinations of variables) are most useful in predicting the purchase o a caravan insurance policy?

4. The data in the file `covertype.data` were obtained from the U.S. forest Service and are concerned with seven different types of forest cover. There are 581 012 observations (each a 30 x 30 metre cell) on 54 input variables (10 quantatative variables, 4 binary wilderness areas and 4o binary soil type variables). The data can be downloaded in compressed format from

   `kdd.ics.uci.edu/databases/covertype`

   where you can find a description of the variables and a description of the task.

   Divide the data randomly into learning and test sets and use any method of your choice to predict the forest cover type for the test set. Estimate the test error rate.

5. Consider again the `The Insurance Company Benchmark` data. There are 86 variables on product-usage data and socio-demographic data for customers of an insurance company. The file `ticdata2000.txt` is a learning set of 5822 customers; the file `ticeval2000.txt` is a test set of 4000 customers. A description of the variables may be found here:

   `http://kdd.ics.uci.edu/databases/tic/dictionary.txt`

   More information may be found here:

   `http://kdd.ics.uci.edu/databases/tic/tic.data.html`

The target variable is variable 86: ‘CARAVAN:Number of mobile home policies’. The data set gives information on customers of an insurance company and contains 86 variables on product-usage data and socio-demographic data derived from zip area codes. There are 5822 customers in the learning set and another 4000 in the test set. The data were collected to answer the following question: can you predict who would be interested in buying a caravan insurance policy and an explanation why?

Does the LASSO variable selection technique help to give a good classifier?