Statistical Image Analysis
Computer Exercise 3: Pattern Recognition

Mats Kvarnström
Department of Mathematical Statistics
Chalmers University of Technology

January 2005

1 Introduction

The purpose of this computer exercise is to do the calculations presented in Chapter 2 in the Lecture Notes [1] on Pattern Recognition. Therefore you should have read this before you begin with this exercise.

Here, we are confronted with a data file consisting of 50 specimens from 3 different species (or classes). For each specimen, data consists of four variables. These four variables will serve as our feature vector. Under the assumption that the feature vector have a multinormal distribution but with different means according to class (but the same covariance matrix), we minimize the probability of misclassification if we choose a *linear discrimination* function when deciding which class an observed object belongs to.

Section 2 deals with the theory needed in this exercise. In Section 3 we begin with loading the data do some scatter plots. Then we implement the linear discriminator as a Matlab function, after which we select the features to use in order to minimize the estimated *error rate*, using a *cross-validation* method.

1.1 Data

The data we are going to use is contained in the file

- `iris_alt.txt`: Fisher's Iris data set. It consists of four variables measured for 50 plants of each of the three *Iris* (*svärdslilja* in Swedish) species; Iris setosa, Iris versicolor, and Iris virginica.

The file can be found on the course homepage under 'Data'.

Take a look at the data by left-clicking on the link. First there are two rows preceded with the comment sign '%', explaining the data. The actual data is listed in 150 rows, each row representing a plant. The first column in each row tells us which of the three species it belongs to, and the next four columns are the measured lengths and widths of that plant.

To download it (in Netscape), right-click on the filename and choose 'Save Link As...'.

2 Theory

Everything in this section is, more or less, covered in the Lecture Notes [1]. We highlight the parts needed for the actual calculations in this exercise, by repeating them.

All vectors are assumed to be column vectors.

2.1 Optimal discrimination

The optimality criterion used in this exercise will be to minimize the probability of misclassification. Let $f_i(x)$ be the probability density function and π_i be the prior probability of class ω_i, $i = 1, 2, 3$. Given a feature vector X, the optimal rule is to, prefer class ω_i to ω_j if

$$\pi_i f_i(X) > \pi_j f_j(X) \tag{1}$$

This boils down to choosing the class ω_i if

$$\pi_i f_i(X) = \max_{1 \leq j \leq 3} \pi_j f_j(X) \tag{2}$$

The problem is that we do not know the distributions or prior probabilities. Therefore, we have to make assumptions and/or simplifications.

2.2 Linear discrimination

An important special case in discrimination, is to assume that the feature vector in each class ω_i has a multivariate normal distribution with expectation μ_i (a vector if we use more than one feature) and covariance matrix C_i. Then if X is a d-dimensional feature vector of class ω_i, the probability density of X is

$$f_i(x) = \frac{1}{(2\pi)^{d/2}(\det C_i)^{1/2}} \exp\{-\frac{1}{2}(x-\mu_i)^T C_i^{-1}(x-\mu_i)\}. \tag{3}$$

If we furthermore assume that the covariance matrices are equal for the all classes

$$C_i = C, \quad i = 1, \ldots, k$$

and use this in equation (3) inserted in the optimal rule (1) and taking the natural logarithm, we get the linear discrimination rule: Given the feature vector x, we prefer class ω_i to ω_j if

$$-\frac{1}{2}(x-\mu_i)^T C^{-1}(x-\mu_i) + \frac{1}{2}(x-\mu_j)^T C^{-1}(x-\mu_j) > \log\frac{\pi_j}{\pi_i}. \tag{4}$$

Rearranging this gives us the rule as stated in the Lecture Notes: Given the feature vector x, we prefer class ω_i to ω_j if

$$(\mu_i - \mu_j)^T C^{-1}(x - \frac{1}{2}(\mu_i + \mu_j)) > \log \frac{\pi_j}{\pi_i}. \qquad (5)$$

Note that in equation (5), if C is an identity matrix, or a multiple of one (i.e. we have independent features with equal variances) and $\pi_i = 1/3$ for all i, then we choose x to belong to the class which has its expectation vector closest (in the normal, Euclidean sense) to x. Think about that for a while, and admit that it sounds like a reasonable decision rule since the assumptions on C and π_i means that we do not have much information, except for the expectation of the features of each class.

2.3 Estimation

All this looks great. There is only one small problem. We do not know the expectation vectors μ_1, μ_2, and μ_3 and the covariance matrix C. We have the training set though. We can estimate the expectations by taking the mean of the feature vectors from each class

$$\hat{\mu}_i = \frac{1}{n_i} \sum_{m=1}^{n_i} X_{i,m} \qquad (6)$$

where $X_{i,m}$ is the m:th observed feature vector of class ω_i. Estimation of the covariance matrix C is done by first estimating C_i for each class by

$$\hat{C}_i = \frac{1}{n_i - 1} \sum_{m=1}^{n_i} (X_{i,m} - \hat{\mu}_i)(X_{i,m} - \hat{\mu}_i)^T \qquad (7)$$

and then use the *pooled* covariance matrix

$$\hat{C} = \frac{1}{n_1 + n_2 + n_3 - 3} \sum_{i=1}^{3} (n_i - 1)\hat{C}_i \qquad (8)$$

3 Discrimination with Matlab

In this section we are going to explore the Iris data set and write functions in Matlab used for linear discrimination and feature selection. We assume that the prior probabilities are $\pi_i = 1/3$ for all i (a reasonable choice since we have 50 species of each class).

It is recommended that you use a *script*-file called for example `lab3_main.m` so that you do not have to re-write everything manually in the command window once you have made some changes. See the Appendix of CE2 for an explanation of what a script file is (one could say that it is a 'function without the header') and how to use it.

3.1 Loading the data to Matlab

Once you have downloaded `iris_alt.txt` in your working directory, you can load the data to a variable in Matlab by using `load`

```
>>load iris_alt.txt
```

Notice that you do not have to specify an output variable. Matlab automatically stores the data in a variable with the same name as the file (except for the suffix '.txt'). Type `iris_alt` to display the matrix and compare it with the source `iris_alt.txt` so that they seem to be in agreement with each other.

Our present data variable is quite cumbersome to deal with. What you should do is to extract the rows corresponding to the different species (classes) into three separate variables, called for example `X1`, `X2`, and `X3`. When doing this the `find` command is quite useful (remember how we used it in CE2). First find the indices of the rows beginning with '1' using `find` and then take the last four columns of these rows and call them `X1`. So, for `X1`, this is done in a single command line by writing

```
>>X1=iris_alt(find(iris_alt(:,1)==1),2:5);
```

And, of course, for `X2` and `X3`, changing the `1` to `2` and `3`, respectively, as

```
>>X2=iris_alt(find(iris_alt(:,1)==2),2:5);
>>X3=iris_alt(find(iris_alt(:,1)==3),2:5);
```

Now we have organized the data to a form where `X1`, `X2` and `X3` are the observed feature vectors for the three classes Iris Setosa, Iris Versicolor, and Iris Virginica, respectively. This form is suitable for our forthcoming work.

3.2 Scatter plots

Draw scatter plots for all the 150 observations and all six pairs of variables (features). This is useful in order to get a feeling for the data. You might see pairs of variables which seem better than others, to use in the discrimination.

To plot feature 3 (Petal length) against feature 4 (Petal width), write

```
>>plot(X1(:,3),X1(:,4),'r*',X2(:,3),X2(:,4),'g*',X3(:,3),X3(:,4),'b*')
```

We have written a color code and star after each class with red, green, and blue representing the three classes. (What happens if you forget the `*` ?)

You should also try a three dimensional scatter plot. For this, use the command `plot3`.

3.3 Expectation and covariance estimation

To estimate the expectation vector as in equation (6), the Matlab command `mean` might be useful. For the mean of the entire feature vector (i.e. all four variables) of class ω_1, type

```
>>mu1=mean(X1)
```

As you might expect, `mean` takes the mean value over each column, resulting in a row vector. Since all vectors in the formulas of Section 2 are column vectors, take the transpose right away, in order not to confuse things later:

```
>>mu1=mean(X1)'
```

Now, do the same for the mean of `X2` and `X3` and call them `mu2` and `mu3`, respectively.

Note that if you want the mean vector of a subset of the features, say, of the first and third feature of class ω_2, you just type

```
>>mean(X2(:,[1,3]))
```

The covariance estimation in equation (7), can also be done using a built-in Matlab function; it is called `cov`. Type

```
>>C1=cov(X1)
```

and `C1` will be the estimated covariance matrix of class ω_1. Check that this is a square matrix of size 4. (What happens if you type `cov(X1')` instead, and why?) If you want to estimate the covariance matrix of a subset of features, do as you did with the mean.

For the pooled covariance matrix in equation (8), just add `C1`, `C2`, and `C3` and divide by three, since we have that $n_i = 50$ for all i,

```
>>C=(C1 + C2 + C3)/3
```

3.4 Discrimination

Now have the tools needed to implement the linear discrimination rule (2) by using the estimated $\hat{\mu}_i$ and $\hat{C}$ from above, together with the assumption of equal prior probabilities $\pi_i = 1/3$.

Write a function which returns the class number `i`, an arbitrary feature vector `x` belongs to, using the linear discrimination rule. The features used should be any subset of the four available. The input variables should be, in addition to `x`, the pooled covariance `C` and the estimated expectation vectors `mu1`, `mu2`, and `mu3`.

To get started, here is how the header and the first lines could look like:

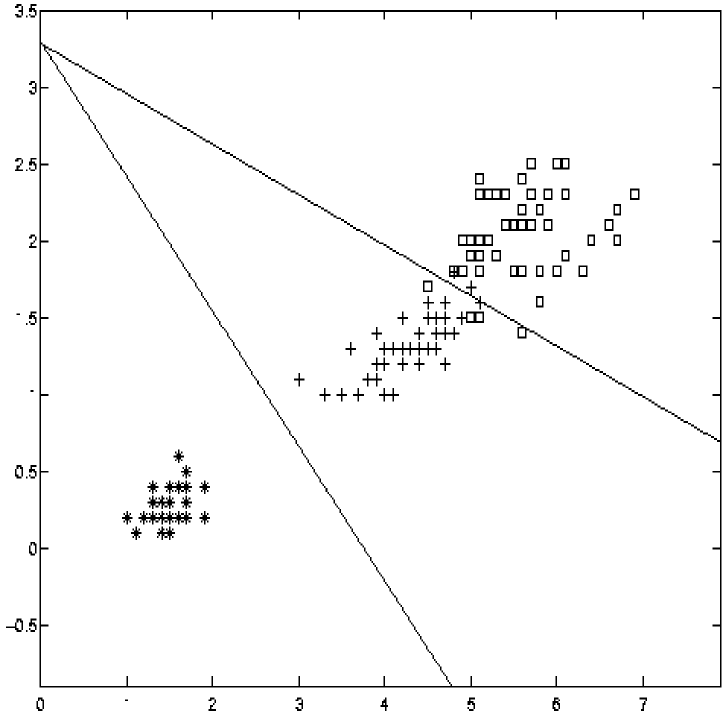

Figure 1: Scatter plot for feature 3 (Petal length) against feature 4 (Petal width) together with the linear discrimination boundaries. The species Iris Setosa, Iris Versicolor and Iris Virginica are represented by stars, crosses, and squares, respectively. Using these two features you can see that 6 of observed plants in training set would be misclassified.

```
function class=lin_disc(x,C,mu1,mu2,mu3)
%class=lin_disc(x,C,mu1,mu2,mu3)
%
%Assigns which class x should belong to, using
%a linear discrimination rule with equal
%prior probabilities.
%
%x, mu1, mu2, and mu3 are column vectors (of equal size)
%and C is the covariance matrix.

logf1=-0.5*(x-mu1)'*(C\(x-mu1));
logf2=%????
logf3=%????

[logfmax, class] = max([logf1 logf2 logf3]);
```

```
%The rest is up to you!!!!
```

3.5 Error rate estimates and feature selection

The `lin_disc.m` above is all you need to make the actual decision of discrimination. The questions are then: Which features should I use? Should I use all four or is it enough with for example two?

It is **not** true that more features automatically means better discrimination. Section 2.5 and 2.6 of the Lecture Notes [1] deals with the problem of selecting features and estimating error rates. If you have not read these two sections, you should read them now.

In order to select features, we need to estimate the error rate of a given subset of the features. The method we are going to use is the *cross-validation* method. What we need to do is to go through all 150 observations (species), each time letting one of them serve as a sample to classify, while using the other 149 observations as training set (i.e. we estimate the expectation vectors and the pooled covariance matrix with the remaining 149 observations).

It is recommended that you implement this is as a function. Use the data matrices `X1`, `X2`, and `X3` together with a variable specifying which features to use as input. The output should the estimated error rate, computed as the number of misclassifications divided by the total number of observations (i.e. 150). The beginning of this function can be found on the course homepage under 'Computer exercises'.

Since the number of features available are only four, it is feasible to calculate the error rate for all 2^4 combinations and use the subset of features which gives the smallest error rate. If there is a tie, always choose the combination of lowest dimension (i.e. the one with fewest number of features).

Notice, that if we would have had, say 20 features, the method of trying all of the 2^{20} combinations would not be very practical. Then you probably would have to use another method, one of which is the method of *forward selection*. For this; see Section 2.4 in the Lecture Notes.

3.6 Optional: Quadratic discrimination

When you have done all the parts above, you probably have realized that it is not harder to implement the *quadratic discriminator*, which is the same but without the assumption of equal covariance matrices. All you need is a function similar to `lin_disc.m` with three covariance matrices instead of the pooled as input, and the quadratic discrimination rule: Given the feature vector x, we prefer class ω_i to ω_j if

$$(x-\mu_j)^T C_j^{-1}(x-\mu_j) - (x-\mu_i)^T C_i^{-1}(x-\mu_i) - \log\left(\frac{\det C_i}{\det C_j}\right) > 0 \tag{9}$$

instead of equation (5). You are strongly encouraged to do this if you have the time.

Do we get a better error rate with a quadratic discriminator?

4 Matlab commands used in this exercise

- `cov`: Estimates the covariance matrix of an observation matrix where each row is an observation and each column is a variable.
- `find`: This function returns the indices of the non-zero elements of the input where the input usually is a binary expression.
- `load`: Loads data contained in a file (text or binary) to a variable in Matlab.
- `mean`: Takes the average of the input elements. If the input is a matrix, the mean is over each column.
- `plot3`: Plots lines and points in a 3-dimensional space.

References

[1] Mats Rudemo. *Image Analysis and Spatial Statistics*. Dept. of Mathematical Statistics, Chalmers University of Technology, 2003.