Pattern Recognition practical 1

Maikel Withagen (s1867733) Steven Bosch (s1861948)

September 2015

1 Assignment 1

1.1

To compute the pair-wise correlation coefficients we used the following command:

Input

```
1 load('lab1_1.mat')
2 corrcoef(lab1_1)
```

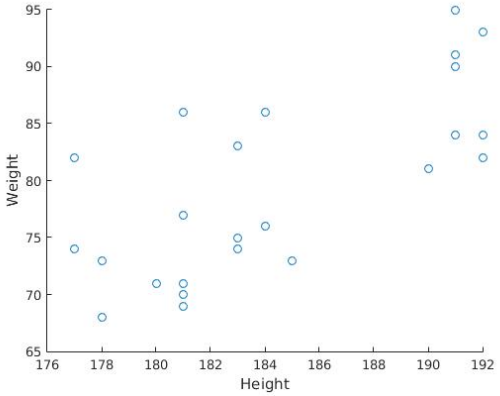
This yields us the following table of correlation coefficients:

Table 1: *Pair-wise correlation coefficients*

	Length	Age	Weight
Length	1	-0.0615	0.7156
Age	-0.615	1	0.5142
Weight	0.7156	0.5142	1

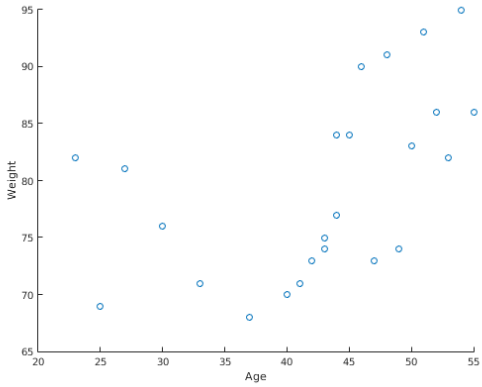
1.2

The two features for which the correlation is the largest are the first and third column, respectively the height and the weight.



(a) Scatterplot of weight to length

The two features for which the correlation is the second largest are the second and third column, respectively the height and the weight.



(b) Scatterplot of weight to age.

Figure 1

From a scatterplot alone it is hard to draw conclusions about any possible relationships between the different features. We do get indications though; figure 1a shows that there is likely to be a correlation between the weight and the height. An increase in weight seems to correspond to a (somewhat linear) increase in height. A similar kind of relationship can be seen in figure 1b, between the factors weight and age.

2 Assignment 2

2.1

The following subsections show the code we used to acquire the 1000 Hamming distances for set S and D.

a

To compute the set S we first create the set and fill it with zeros (line 1). Then for every element in the set we randomly pick a person and two random rows within that person and consequently load the actual content of these rows (lines 2-9). Then we compute the hamming distance between those two rows and store it within the appropriate place in the set (line 11). Finally the whole set is scaled so that every element in the set reflects the hamming distance between two elements instead of two rows.

Code for set S

```
1  hd_s = zeros(1,1000);
2  for i = 1:1000
3    person = randi([1,20]);
4    row1 = randi([1,20]);
5    row2 = row1;
6    while(row1 == row2)
7      row2 = randi([1,20]);
8    end
9    load(sprintf('person%02d.mat',person));
10
11   hd_s(i) = sum(abs(iriscode(row1,:)-iriscode(row2,:)));
12 end
13 hd_s = hd_s/30;
```

b

To compute the set D we do the same thing except for the fact that we compute the hamming distances between rows of two different persons:

Code for set D

```
1  hd_d = zeros(1,1000);
2  for i = 1:1000
3    person1 = randi([1,20]);
4    row1 = randi([1,20]);
5    row2 = randi([1,20]);
6    person2 = person1;
7    while(person1 == person2)
8      person2 = randi([1,20]);
9    end
10   load(sprintf('person%02d.mat',person1));
11   x = iriscode(row1,:);
```

```
    load(sprintf('person%02d.mat',person2));
    y = iriscode(row2,:);
    hd_d(i) = sum(abs(x-y));
end
hd_d = hd_d/30;
```

2.2

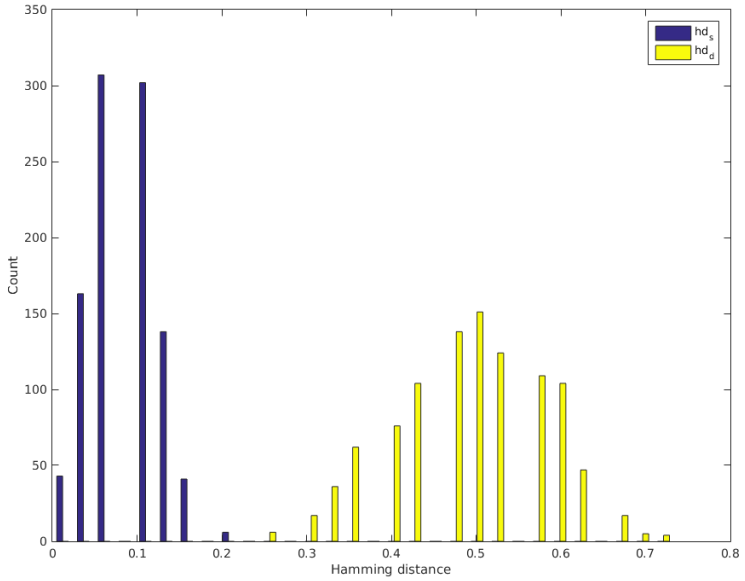


Figure 2: Histogram of sets S and D.

Figure 2 shows the histogram of sets S and D. The figure shows that the two distributions overlap very little, most of it around $hd = 5,6$.

2.3

The means and variances of both of the sets are the following:

Set	Mean	Variance	Standard deviation
S	0.0825	0.0016	0.0398
D	0.4946	0.0079	0.0886

Table 2: Means and variances of both of the sets

The prior propability of two bits being the same between persons the following is: $1-0.4946 = 0.5054$. Using the formula $n = p(1-p)/\sigma^2$ we get $n = (0.4946 * (1-0.4946))/(0.0886^2) \approx 31.84$ statistically independent bits that are needed to encode an iris pattern. This means that the current number of bits in the iris vectors is insufficient.

2.4

We acquired the graph in figure 3 (see the appendix for the matlab code we used).

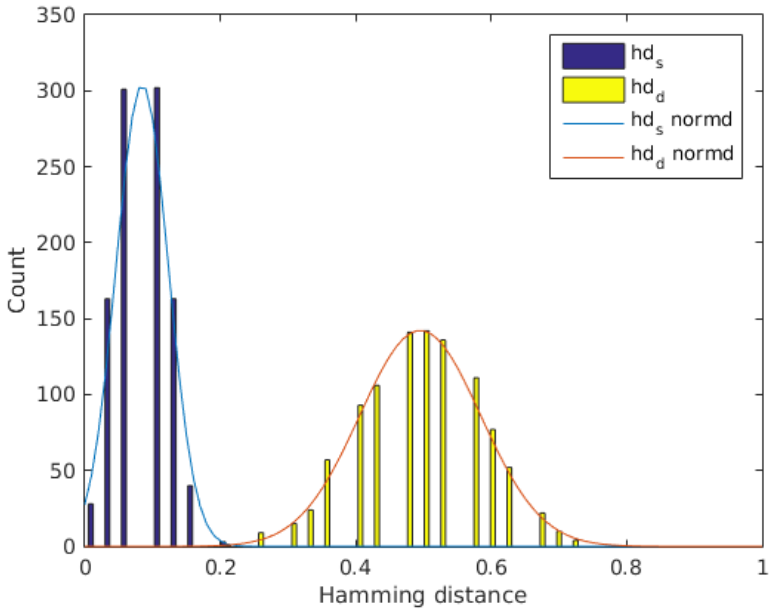


Figure 3: The histograms of sets S and D plotted with two Gaussian functions using the mean and variances.

Figure 3 shows the plot of the combined histograms and gaussian functions (note that the plot is slightly different from the previous subsections, because we reran the script in Matlab). The scaling is done by dividing the maximum value of the D and S sets by the maximum value of the initial Gaussian functions. This way the maximum value of the new Gaussian functions is the same as the maximum value of the D and S sets, so that the the Gaussians are nicely scaled to the histograms.

2.5

a

To estimate the value of the decision criterion we used the following code:

```
% Calculate the cumulative distribution function for the
% given x-value (range 0.000 - 1.000, with stepsize 0.001)
normc = normcdf([0:0.001:1],mean(hd_d),std(hd_d));
% Count the amount of x-values below the threshold value, minus 1 to correct
% from starting from 0.000, and divide by 1000 to get the decision criterion
d = (sum(normc(:) <= 0.0005)-1)/1000;
```

This yielded the value 0.185 as criterion. This means that below a hamming distance of 0.185, we will erroneously classify two irisses as being from the same person.

2.6

Using the criterion of 0.185, we get a false rejection rate of 0.00072, using the following code:

```
% Same as before, now for set S
normc = normcdf([0:0.001:1],mean(hd_s),std(hd_s));
% False rejection is integral of HD > d is 1 - normc(d)
1 - normc((d*1000)+1)
```

This means that up to 0.072% is falsely classified as being from two different persons.

2.7

Using the first part of the code in the appendix for assignment 2.6 we get a minimum hamming distance of 0, which we find in person 5, rows 1, 2, 3, 5, 6, 8, 10, 11, 12, 15, 17 and 18. Indeed, every known bit in testperson is the same as the bit in the same place in for example person 5, row 1:

It is clear that the the iris code of testperson most likely belongs to person 5. Now we use the second part of the code to calculate the average normalized hamming distance of the testperson to every row of person 5, which gives us a hamming distance of 0.0275.

Next we compute a set for person 5 just as we did for set S and set D. Finally we use this set and the HDmin value we got from the previous steps (0.0275) to calculate the integral below it, which gives us a significance value of $7.0872e-07$. This means that we can most likely reject the null hypothesis that the persons are different.

3 Assignment 3

3.1 Background

The topic of biometric identification has been receiving a lot of attention lately. The use of fingerprints, facial features, and iris recognition are considered appropriate for identification. Of these three, fingerprints and iris recognition are considered to be more accurate than facial features, while facial featuresare more flexible, e.g. for the use in surveillance scenarios. A possible improvement for the performance of facial feature identification is the combination of information from multiple sources, for instance the ear. Ear images can be obtained in a similar maner to face images, and a recent study suggests that they are comparable in recognition power[1], and a combination of the sources gives a significant improvement over their individual performance.

3.2 Method

3.3 Achieved Results

3.4 Conclusion

References

[1] Kyong Chang, Kevin W Bowyer, Sudeep Sarkar, and Barnabas Victor. Comparison and combination of ear and face images in appearance-based biometrics. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(9):1160–1165, 2003.

Appendix

Code for assignment 2.4:

```
hold off;
```

```
% Making the histogram
hd = [hd_s;hd_d].'
hist(hd,30); xlabel('Hamming distance'), ylabel('Count');
hold on;

% Create scaled Gaussian plots
x2 = sum(hd_s(:) == mode(hd_s));
normd = normpdf([0:0.01:1],mean(hd_s),std(hd_s));
plot([0:0.01:1],normd*(x2/max(normd)));

x1 = sum(hd_d(:) == mode(hd_d));
normd = normpdf([0:0.01:1],mean(hd_d),std(hd_d));
plot([0:0.01:1],normd*(x1/max(normd)));

legend('hd_s', 'hd_d', 'hd_s_normd', 'hd_d_normd');
```

Code for assignment 2.6:

```
% Load the testperson's iriscode
load(sprintf('testperson.mat'));
x = iriscode(1,:);

% Initialize variables
HDmin = 30; % HDmin starts at the maximum HD
person = 0; % The person with the minimum HD
row = 0; % The row within that person with HDmin

% Loop over all persons
for i = 1:20
    load(sprintf('person%02d.mat',i));
    % Loop over all rows within the persons
    for j = 1:20
        y = iriscode(j,:);
        HDtemp = 0; % HD for this specific row
        % Loop over every element, only add the HD to HDtemp if the element in
        % testperson is not 2 (since that bit is unknown)
        for k = 1:30
            if x(1,k) ~= 2
                HDtemp = HDtemp + abs(x(1,k)-y(1,k));
            end
        end
        % If the HD of the current row is the lowest, replace HDmin with
        % that and update the person and row
        if HDtemp <= HDmin
            HDmin = HDtemp;
            person = i;
            row = j;
            if HDmin == 0; % print the persons and rows with the lowest HDmin possible
                person
                row
            end
        end
```

```
35      end
36  end
37  % Normalize HDmin to the amount of known bits in testperson (which is 20,
38  % because there are 10 unknown bits)
39  HDmin = HDmin/20;
40
41
42  % Now calculate the normalized HDmin for the comparison of testperson with
43  % every row in person 5.
44
45  HDtemp = 0;
46  load(sprintf('person05.mat'));
47
48  for j = 1:20
49      y = iriscode(j,:);
50      % Loop over every element, only add the HD to HDtemp if the element in
51      % testperson is not 2 (since that bit is unknown)
52      for k = 1:30
53          if x(1,k) ~= 2
54              HDtemp = HDtemp + abs(x(1,k)-y(1,k));
55          end
56      end
57  end
58  HDmin = HDtemp/400 % Normalize HDmin
59
60  % Calculate the set hd_5 just as with setS and setD
61
62  hd_5 = zeros(1,1000);
63  for i = 1:1000
64      row1 = randi([1,20]);
65      row2 = row1;
66      while(row1 == row2)
67          row2 = randi([1,20]);
68      end
69      load(sprintf('person05.mat',person));
70
71      hd_5(i) = sum(abs(iriscode(row1,:)-iriscode(row2,:)));
72  end
73  hd_5 = hd_5/30;
74
75  % Calculate the integral of the tail of the distribution from the computed
76  % HDmin
77
78  normc = normcdf([0:0.001:1],mean(hd_5),std(hd_5));
79  1 - normc((HDmin*10000)+1)
```