

EE435: Biometric Signal Processing

Project 2: Pattern Recognition 2 (System Design)

Assigned: Tues 1/25/11

Due: Mon 2/7/11

I. Introduction

In this project, you will be using the information in the *people.dat* file can be downloaded from the course website under “Projects”. This file is *similar* to the file you used in project 1 in that it is information collected from 50 men and 50 women. For each, their height, shoe size, hair length and ring size were measured and recorded as a feature vector. The overall goal of this project is for you to develop a minimum distance pattern recognition system that is able to take in a person’s measurements and automatically classify that person correctly as a male or female.

Download the *newpeople.dat* file from the course website. Use the load command to bring this data into MATLAB:

```
>> load newpeople.dat
```

This will bring a variable into your MATLAB workspace called “newpeople”, which is a 400 row x 6 column matrix (the *peopledata.dat* file only had 5 columns). Each row contains a pattern vector from a different person. The six columns are laid out as follows:

1st column: the true gender of the individual (class 1 = male, 2 = female)—this is not really a feature, it is *ground truth*

2nd column: this column is all 0s to begin, but your code will insert your algorithm’s class number (1 = male, 2 = female)

3rd column: feature #1-height (inches) (Feature)

4rd column: feature #2-shoe size (Feature)

5th column: feature #3-hair length (inches) (Feature)

6th column: feature #4-ring size (Feature)

II. Pattern Recognition System Design

Create a model (or template) for what a man looks like (at least as far as these measurements go), and a model (template) for women. The simplest model is to create a pattern vector that is a mean pattern vector. For example: if you average the pattern vectors from 10 men, you could call that the model for class “men”, and you would do the same for women. The template would then be a 1x4 vector of

```
[feature#1 avg, feature#2 avg, feature#3 avg, feature#4 avg]  
= [avg_height avg_shoe_size avg_hair_length avg_ring_size].
```

Once you have a template for the male class and one for the female class, use the *nearest neighbor rule* as a decision rule to decide which class a new pattern vector belongs to. That is, after you compute the model (template) for class 1 (men) and class 2 (women), your system would take an unknown pattern vector and figure out which model (template) is closer...that will be your decision as to which class that unknown pattern vector belongs to. Then, by having the ground truth available for each pattern vector (which is contained in column 1 of *people.dat*, you can measure your system’s accuracy.

Please DO NOT share your algorithm or your results with the other groups. You can always ask me questions.

For this project:

1. Determine a model (template) for each class. Choose 15-25 random males and females to come up with templates for the male class, and for the female class. You have four features for each individual. It is possible that one or more of the features actually may degrade the performance of the system, so that fewer features may give better results. This may or may not be the case, but it should be a consideration; the scatter plots may help you decide if one or more of the features do not help the problem). If you choose fewer features, this basically means you’re ignoring columns in the *people* matrix.

- Write four well-commented functions called (*dabs*, *dwabs*, *dEuclidean* and *dwEuclidean* to compute the four different types of distances we discussed. The *dabs* and *dEuclidean* functions have two inputs (the two pattern vectors) and one output (the distance). The *dwabs* and *dwEuclidean* functions have three inputs (the two pattern vectors, and a vector of standard deviations of the features) and one output (the distance). The only error checking for each function is that both of the input pattern vectors must have the same length or else a null value [] is returned and an error message appears.
- For your system, determine the distance measure that gives you the best performance on the *newpeople.dat* file...you have the ground truth available. Performance will depend on the features you used, how you determined your prototype for each class, the distance measure you chose, and the data itself.
- You can measure performance of your system with the *newpeople.dat* data. Your code should cycle through each row of the *newpeople* matrix (in a *for* loop) and for each row (which represents a pattern vector), insert either a 1 (if your system decides class 1) or a 2 in the 2nd column. You can then find out how many errors you have by checking if the 1st column (ground truth) is equal to the 2nd column (your system's decision) for each row. Since there are 400 rows, your accuracy in % is equal to the number of rows where the 1st and 2nd column values are equal divided by 400, multiplied by 100%.
- When you're done testing and are satisfied with your system design (adequate performance), email me your *people.dat* file after you've filled the 2nd column with 1s or 2s. Save your *people* matrix as a *.dat* file using your last name(s) in a command like:

```
>> save LadoTurner_people.dat people -ascii
```

and email the *.dat* file to me so I can check the accuracy you obtained. Note: fill in your names in the file name with no spaces. Depending on which features you use, and how you measure distance, your resultant accuracy will probably be different from your classmates.

- Now you're ready to test your algorithm with unknown data: download the *testpeople.dat* file from the course website. Use the `load` command to bring this data into MATLAB:

```
>> load testpeople.dat
```

This will bring into your MATLAB workspace a variable called "testpeople", which is a 2000 row x 5 column matrix (note: **NOT** 6 columns). Consider each row a pattern vector. The five columns are laid out as follows:

1st column: this column is all 0s to begin, but your code will insert your algorithm's class number (1 = male, 2 = female)

2nd column: feature #1-height (inches)

3rd column: feature #2-shoe size

4rd column: feature #3-hair length (inches)

5th column: feature #4-ring size

Run your pattern recognition algorithm on this data, but note there is NO GROUND TRUTH. That is, you don't know whether each pattern vector represents a male or female. Fill in the first column in the *testpeople* matrix with a 1 or 2, depending on if your system called that row male (1) or female (2). After filling in the 1st column of the *testpeople* matrix, write it into a *.dat* file using your last name(s), for example:

```
>> save LadoTurner_testpeople.dat testpeople -ascii
```

Email me your *.dat* file, and I will determine your system's accuracy with the unknown data.

Use the formal project report described in the course policy statement. Include in your report information like: how you chose your features (should include useful scatterplots), how you determined the best distance metric, how you determined your prototype for each class; what your template pattern values were; what your accuracy was on the *people.dat* file, etc.

Distance formulas

Distances between two vectors X and Y: $X = [x_1 \ x_2 \ x_3 \ x_4]$ and $Y = [y_1 \ y_2 \ y_3 \ y_4]$

Absolute Distance:

$$d = |x_1 - y_1| + |x_2 - y_2| + |x_3 - y_3| + |x_4 - y_4|$$

Euclidean Distance:

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2 + (x_4 - y_4)^2}$$

Weighted Distance

Assume Y is the class template (model), and the standard deviations of each of the individual four features of Y is given by $[\sigma_1 \ \sigma_2 \ \sigma_3 \ \sigma_4]$

Weighted Absolute Distance:

$$d = \frac{|x_1 - y_1|}{\sigma_1} + \frac{|x_2 - y_2|}{\sigma_2} + \frac{|x_3 - y_3|}{\sigma_3} + \frac{|x_4 - y_4|}{\sigma_4}$$

Weighted Euclidean Distance:

$$d = \sqrt{\left(\frac{x_1 - y_1}{\sigma_1}\right)^2 + \left(\frac{x_2 - y_2}{\sigma_2}\right)^2 + \left(\frac{x_3 - y_3}{\sigma_3}\right)^2 + \left(\frac{x_4 - y_4}{\sigma_4}\right)^2}$$

Weighted Distance Example:

If the Y template is given by

$$y = [69.7700 \ 10.7700 \ 1.5952 \ 8.9300],$$

and the standard deviations of each of the features in Y is given by

$$[3.5243 \ 1.5459 \ 1.6275 \ 1.8585],$$

then the weighted Euclidean distance between an unknown vector

$$X = [64 \ 12 \ 5 \ 6]$$

and the template Y is computed by:

$$d = \sqrt{\left(\frac{64 - 69.7700}{3.5243}\right)^2 + \left(\frac{12 - 10.7700}{1.5459}\right)^2 + \left(\frac{5 - 1.5952}{1.6275}\right)^2 + \left(\frac{6 - 8.9300}{1.8585}\right)^2} = 3.19$$