CS479 - Programming Assignment 2: Estimating Parameters Using Maximum Likelihood Estimation

Tim Kwist and Shane Melton

Due March 14, 2016 - Submitted March 14, 2016

Contents

1 Technical Discussion

Maximum Likelihood Estimation's goal is to estimate the parameters of a given distribution based on a sample of data from said distribution. In this assignment, we estimate the parameters of two pairs of distributions - first using 10,000 samples from each distribution, then by only using 1,000 samples from each distribution. Once the parameters for each distribution are estimated, we run through the samples again to classify them with a Bayesian Classifier using the new parameters. For the full theory on the Bayesian Classifier used, see the Project Report from Programming Assignment 1. The distributions we use here to generate the samples are the same as the distributions used in Programming Assignment 1. Below are the mean and covariance matrix for each pair of distributions.

$$\mu_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Sigma_1 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \mu_2 = \begin{bmatrix} 6 \\ 6 \end{bmatrix} \Sigma_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \tag{1}$$

$$\mu_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Sigma_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \mu_2 = \begin{bmatrix} 6 \\ 6 \end{bmatrix} \Sigma_2 = \begin{bmatrix} 4 & 0 \\ 0 & 8 \end{bmatrix} \tag{2}$$

To generate the parameters for each distribution, we use the most generic case of Maximum Likelihood Estimation, in which both the mean and the covariance matrix are unknown:

$$\hat{\mu} = \frac{1}{n} \sum_{k=1}^{n} \mathbf{x_k} => (SampleMean) \tag{3}$$

$$\hat{\mathbf{\Sigma}} = \frac{1}{n} \sum_{k=1}^{n} (\mathbf{x_k} - \hat{\mu})(\mathbf{x_k} - \hat{\mu})^t => (SampleCovariance) \tag{4}$$

Additionally, we extend the use of Maximum Likelihood Estimation to the detection of faces in PPM images. As outlined in Section 3.1 of [Yang96 "A Real-time Face Tracker"], a face color distribution can be represented by a Gaussian Model $N(\mu, \Sigma^2)$ by collecting samples of RGB values for pixels in an image in which there is human skin and estimating the mean and covariance of those samples. However, in order to reduce our model from 3 dimensions to 2 dimensions, we normalize the RGB values as follows:

$$r = \frac{R}{R + B + G} \tag{5}$$

$$g = \frac{G}{R + B + G} \tag{6}$$

$$b = \frac{B}{R + B + G} \tag{7}$$

From this, we can eliminate b, as r + g + b = 1 and b = 1 - r - g. Thus, each of our samples consists of a 2-dimensional vector consisting of our normalized red and green values of each pixel. We can apply Maximum Likelihood Estimation here to find a sample mean and sample covariance that can be used to find the Bayesian classifier discriminant needed to classify skin distributions in other images.

Because the RGB color system does not factor out luminance, and image processing cannot easily distinguish luminance differences compared to chromatic differences, we also apply Maximum Likelihood Estimation techniques to our RGB values transformed into the YCbCr color space with the following transformation:

$$Y = 0.299R + 0.587G + 0.114B \tag{8}$$

$$Cb = -0.169R - 0.332G + 0.500B \tag{9}$$

$$Cr = 0.500R - 0.419G - 0.081B \tag{10}$$

Since the Y component contains luminance, we ignore it and use 2-dimensional vectors consisting of Cb and Cr for our samples.

Finally, once we have built the color distribution models for each color space, we attempt to "find" skin pixels in new images by calculating a Gaussian discriminant, using our estimated parameters, for each pixel and using a

threshold to determine whether to accept the given pixel as skin. In the results section, ROC curves are displayed to compare the False Acceptance Rate and False Rejection Rate of our classifier and parameter estimator at various threshold points.

Extra Credit: In addition to the threshold classification for skin color distribution, we also tested a method of classification in which we estimate the parameters of the non-skin color distribution. With parameters for both distributions, we now introduce a second class and can turn this threshold problem into a two-class Bayesian Classification problem in which we calculate the discriminant for each class and classify based on which discriminant returns a larger number (which represents a larger probability).

2 Results

After studying the technical details and math regarding Maximum Likelihood Estimation we implemented the theories using C++. We re-generated the samples generated in Programming Assignment 1, as we no longer had the old data available, using the same parameters listed in (1) and (2).

2.1 Part 1

For part 1 of the assignment we generated new samples using the provided mean and covariance matrix (1) and then classified the data using those known parameters. Afterwards, we then used our maximum likelihood estimator (3) and (4) functions to estimate the parameters listed in (1). Using these estimated parameters we then reattempted classification and compared the results with the results obtained using the known parameters. The estimated parameters and a table of comparisons can be seen below.

$$\hat{\mu_1} = \begin{bmatrix} 1.0332 \\ 0.9637 \end{bmatrix} \hat{\mathbf{\Sigma_1}} = \begin{bmatrix} 4.0279 & 0.0386 \\ 0.0386 & 4.0307 \end{bmatrix} \hat{\mu_2} = \begin{bmatrix} 6.0174 \\ 6.0192 \end{bmatrix} \hat{\mathbf{\Sigma_1}} = \begin{bmatrix} 4.0937 & 0.0713 \\ 0.0713 & 4.0307 \end{bmatrix} \tag{11}$$

Next we repeated the same process, however, we only used $\frac{1}{10}$th the sample size when estimating the parameters and the results can be seen below.

$$\hat{\mu_1} = \begin{bmatrix} 0.9914 \\ 1.0103 \end{bmatrix} \hat{\mathbf{\Sigma_1}} = \begin{bmatrix} 4.0470 & -0.0512 \\ -0.0512 & 3.694 \end{bmatrix} \hat{\mu_2} = \begin{bmatrix} 5.9985 \\ 5.9859 \end{bmatrix} \hat{\mathbf{\Sigma_1}} = \begin{bmatrix} 4.1392 & -0.0169 \\ -0.0169 & 3.8184 \end{bmatrix} \tag{12}$$

	Known Parameters	Unknown Parameters	Unknown Parameters (Small Sample)
Misclassified Sample One	370	363	366
Misclassified Sample Two	376	387	381
Total Misclassified	746	750	747

Table 1: Comparing misclassification errors between when parameters are known (1) or unknown (11)-(12).

2.2 Part 2

For part 2 we repeated the same steps in part 1, however, we used the parameters listed in (2) for sample generation. The data for these tests can be seen below.

$$\hat{\mu_1} = \begin{bmatrix} 1.0186 \\ 0.9885 \end{bmatrix} \hat{\mathbf{\Sigma_1}} = \begin{bmatrix} 4.0939 & 0.0092 \\ 0.0092 & 4.0107 \end{bmatrix} \hat{\mu_2} = \begin{bmatrix} 5.9408 \\ 6.0417 \end{bmatrix} \hat{\mathbf{\Sigma_1}} = \begin{bmatrix} 15.9547 & -.68068 \\ -0.6808 & 64.5353 \end{bmatrix} \tag{13}$$

Again, we used estimate the parameters using only $\frac{1}{10}$th of the original sample. The estimated parameters can be seen below.

$$\hat{\mu_1} = \begin{bmatrix} 1.0311 \\ 0.8955 \end{bmatrix} \hat{\mathbf{\Sigma_1}} = \begin{bmatrix} 4.1129 & 0.0217 \\ 0.0217 & 4.0229 \end{bmatrix} \hat{\mu_2} = \begin{bmatrix} 5.8784 \\ 6.0788 \end{bmatrix} \hat{\mathbf{\Sigma_1}} = \begin{bmatrix} 15.4465 & -0.1301 \\ -0.1301 & 62.846 \end{bmatrix} \tag{14}$$

	Known Parameters	Unknown Parameters	Unknown Parameters (Small Sample)
Misclassified Sample One	948	479	1
Misclassified Sample Two	1190	1348	5686
Total Misclassified	2138	1827	5687

Table 2: Comparing misclassification errors between when parameters are known (2) or unknown (13)-(14).

2.3 Part 3

For part 3 we extended the use of Maximum Likelihood Estimation to detect skin colored pixels in PPM images. After using Training1.ppm and ref1.ppm to generate a normal distribution model for skin color, we applied the model to Training3.ppm and Training6.ppm to detect skin which pixels were likely skin colored based upon a changing threshold. Graphs plotting the ROC curve for Training3.ppm and Training6.ppm can be seen below in Figures 1 and 2.

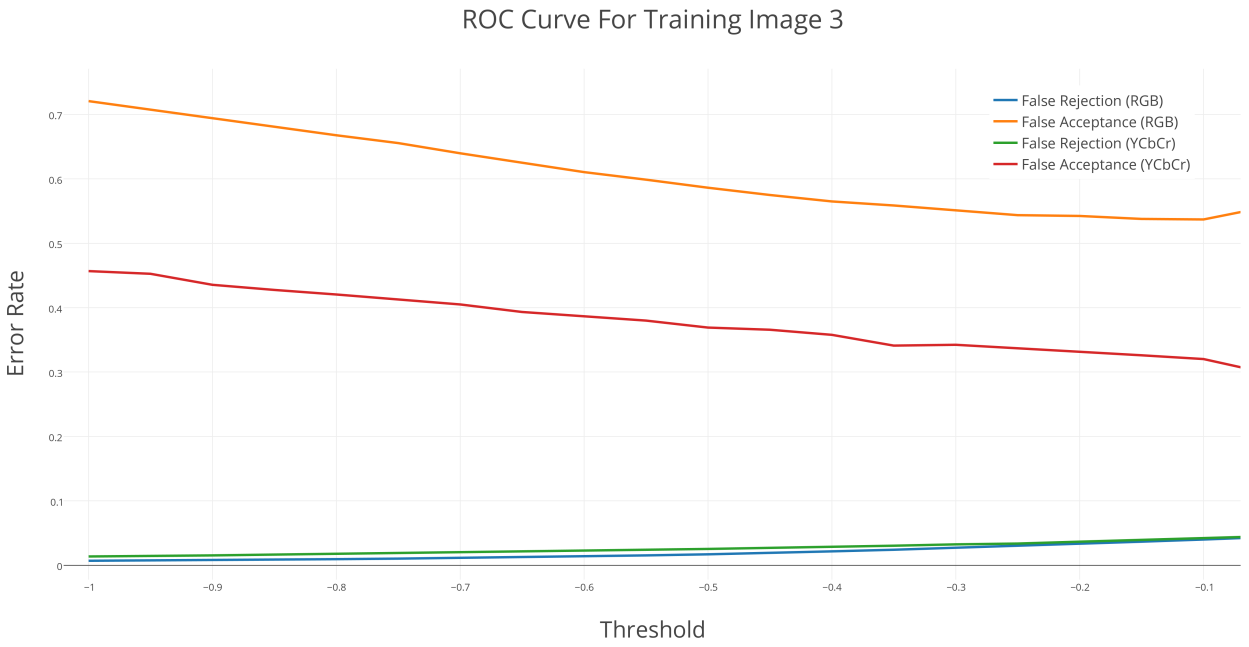

Figure 1: Plotting the ROC curve for Training_3.ppm (both RGB and YCbCr) when compared to ref3.ppm

As you can see, by varying the threshold of our classification we are able to minimize our false acceptance and false rejection rates significantly for both images.

2.4 Part 3 (Extra Credit)

In addition to the previous classifier, we implemented a two class Bayesian classifier. We modeled a class for what skin pixels looked like and another class for what non-skin pixels looked like. Using these estimated parameters for both models, we applied the third case of a Multivariate Gaussian Bayesian classification to compare its performance with the single class classification. As you can see in the Table 3 their were considerably more false acceptances and rejections than when using the single class classification with the optimum threshold.

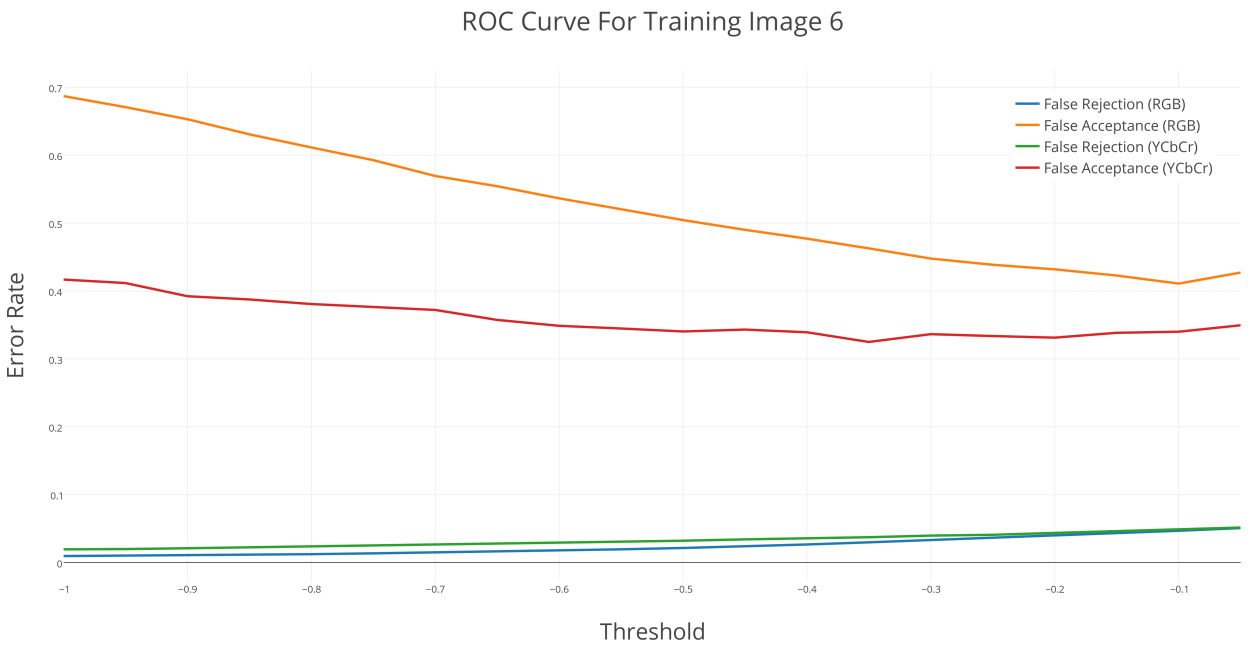

Figure 2: Plotting the ROC curve for Training_6.ppm (both RGB and YCbCr) when compared to ref6.ppm

	False Rejection Rate	False Acceptance Rate
Training Image 3 (RGB)	.00074	.85024
Training Image 3 (YCbCr)	.00131	.63569
Training Image 6 (RGB)	.00236	.81686
Training Image 6 (YCbCr)	.00418	.59163

Table 3: Comparing false rejection and acceptance rates for each image and color scheme using two class Bayesian classification.

3 Division of Work

Shane:

- Results write-up
- Organize main files
- Organize data output and results

Tim:

- Technical write-up
- Working proto-type of image read/write data
- Program MLE functions
- Program documentation + Readme

Both (Essentially pair programming):

- Debugging of image threshold and classification
- Developing overall main files
- Choosing and testing thresholds for skin color distribution

4 Program Listings

Source will be sent by email and can be found on Github at the following URL:
`https://github.com/timkwist/CS479/tree/master/PA2-Parameter_Estimation`

Eigen library can be found at the following URL:
`http://eigen.tuxfamily.org/index.php?title=Main_Page`

Box-Muller Transformation C++ Code:
`ftp://ftp.taygeta.com/pub/c/boxmuller.c`

Image Mainpulation Classes:
`http://www.cse.unr.edu/~bebis/CS302/`