Joseph Bell
ECE 271A

Homework 1 Problem 5

a) Reasonable estimates for the prior probabilities are 0.1919 and 0.8081 for the cheetah and background respectively. These values were calculated by dividing the number of elements in the training data for each type by the total number of elements of both sets of training data.

b) The values of PX|Y (x|cheetah) and PX|Y (x|grass) can be seen below in Figures 1 and 2 respectively.

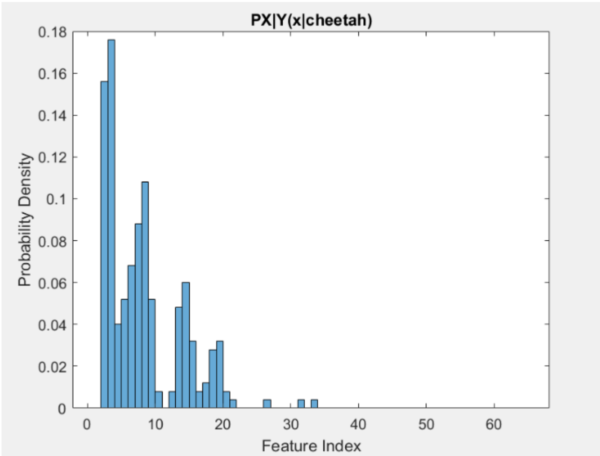

Figure 1: Histogram of PX|Y (x|cheetah)

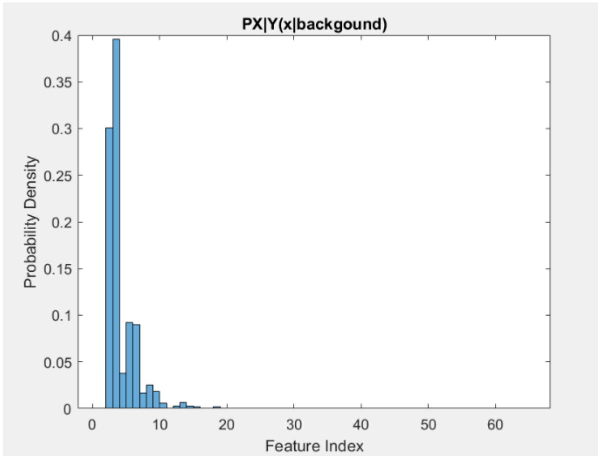

Figure 2: Histogram of PX|Y (x|grass)

c) Figure 3 was created by combining the data from part a and b and using a 0-1 loss Bayesian Decision Rule

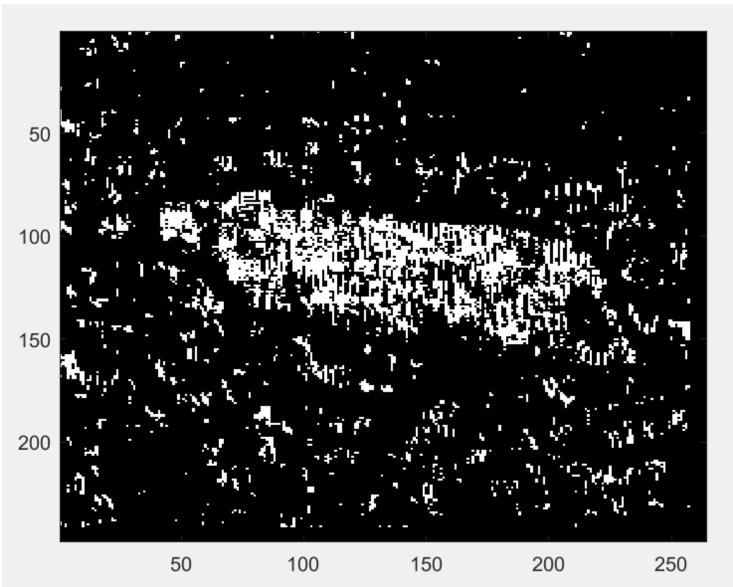

Figure 3: Picture of state array for each pixel

d) The error probability of the algorithm was computed by plotting the joint probability distributions for the background and foreground together (as seen in Figure 4) and extracting the lesser value for each index value that overlapped (as seen in Figure 5). The sum of the extracted values was computed and resulted in 0.1527, or a 15.27% probability of error. The absolute accuracy of the algorithm was also calculated by tallying the number of pixels in the image in Figure 3 that were in the same state as the pixels in the provided solution mask image and dividing by the total number of pixels in the solution mask image. This resulted in an accuracy of 82.22%, or an error rate of 17.78%.

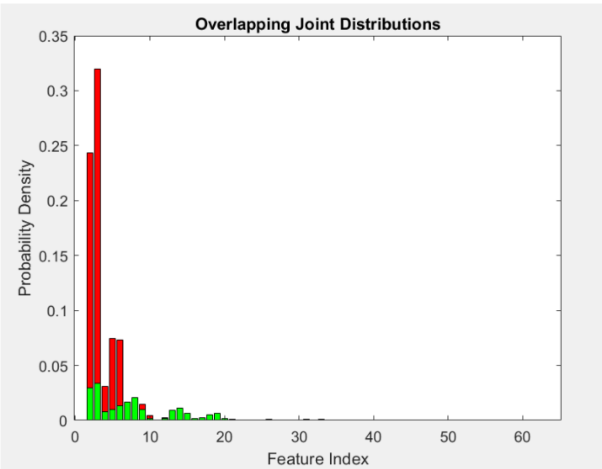

Figure 4: Overlapping joint distributions

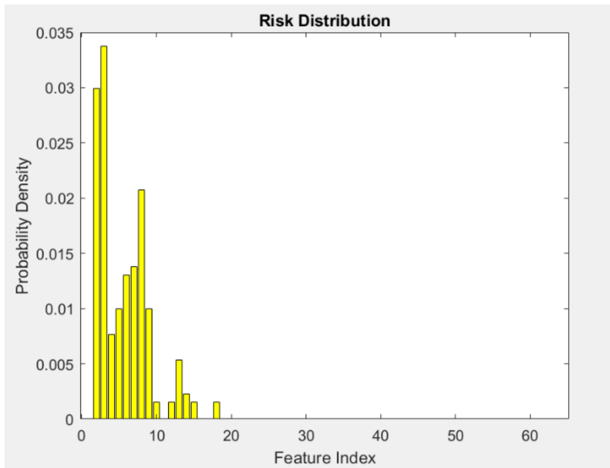

Figure 5: Histogram of algorithm risk

Appendix

```
%Joseph Bell
%ECE271A HW1

clc;
clear;
load('TrainingSamplesDCT_8.mat');

%%%%% CALCULATING PRIORS %%%%%
[rows_FG, cols_FG] = size(TrainsampleDCT_FG);
[rows_BG, cols_BG] = size(TrainsampleDCT_BG);

FG_training_elements = rows_FG*cols_FG;
BG_training_elements = rows_BG*cols_BG;
total_training_elements = FG_training_elements + BG_training_elements;

% priors are calculated from size of the matrices of the training data
prior_cheetah = FG_training_elements/total_training_elements; %0.1919
prior_background = BG_training_elements/total_training_elements; %0.8081

%%%%% Calculating indices of 2nd greatest dct coefficient %%%%%
BG_indices = [];
FG_indices =[];

for i=1:rows_BG
    [sorted_BG,index_BG] = sort(abs(TrainsampleDCT_BG(i,:)),'descend'); %sorting
absolute value of each row
    BG_indices =[BG_indices, index_BG(2)]; %appending the index of each 2nd largest
coefficient
end

for i=1:rows_FG
    [sorted_FG,index_FG] = sort(abs(TrainsampleDCT_FG(i,:)),'descend'); %sorting
abslute value of each row
    FG_indices =[FG_indices, index_FG(2)]; %appending the index of each 2nd largest
coefficient
end


%%%%% creating histograms for cheetah and background %%%%%
BG_hist=figure;
h_bg=histogram(BG_indices,1:65,'Normalization','pdf');
xlabel('Feature Index');
ylabel('Probability Density');
title('PX|Y(x|backgound)');
savefig(BG_hist,'BG_PDF');
```

```
FG_hist=figure;
h_fg=histogram(FG_indices,1:65,'Normalization','pdf');
xlabel('Feature Index');
ylabel('Probability Density');
title('PX|Y(x|cheetah)');
savefig(FG_hist, 'FG_PDF');



cheetah_img = imread('cheetah.bmp');
cheetah_img = im2double(cheetah_img); %converting to double values since training data
is of type double
[cheetah_rows, cheetah_cols] = size(cheetah_img);
cheetah_img = cheetah_img(1:8*floor(cheetah_rows/8),1:8*floor(cheetah_cols/8));
%modifying image so it can be split into 8x8 blocks
[cheetah_rows, cheetah_cols] = size(cheetah_img); %overwriting for modified dimensions
cheetah_row_blocks = cheetah_rows/8; %31
cheetah_col_blocks = cheetah_cols/8; %33
zz = load('Zig-Zag Pattern.txt');
zz = zz+1;
zz = zigzag(zz); %Credit to Alexey Sokolov from
https://www.mathworks.com/matlabcentral/fileexchange/15317-zigzag-scan
                 %for the zig zag code

%%%%% Block Window Sliding %%%%%
new_image = zeros(cheetah_rows, cheetah_cols);
prob_error = 0.0;
for i=1:cheetah_cols-7 %shift scan pointer over a column
    for j=1:cheetah_rows-7
%          disp(j);
%          disp(i);
        block = cheetah_img(j:7+j,i:7+i); %grab 8x8 block
        block_dct = dct2(block);
        zzblock_dct = zigzag(block_dct);
        %%%%% GET SECOND HIGHEST INDEX OF THAT BLOCK %%%%%
        [sorted_zzblock_dct,feature_indices] = sort(abs(zzblock_dct),'descend');
%sorting absolute value of array
        feature=feature_indices(2); %getting second index

        %%%%% DO BAYESIAN DECISION RULE %%%%%
        T_star = prior_cheetah/prior_background;
        choose_background = h_bg.Values(feature)/h_fg.Values(feature);
        if choose_background < T_star
            new_image(j:j,i:i) = 1;
        end
    end
end

figure
imagesc(new_image);
```

```
colormap(gray(255));

%%%%% Calculating Error %%%%%
joint_hist_overlap = figure;
joint_histb=bar(prior_background*h_bg.Values,'r');
hold on
joint_histc=bar(prior_cheetah*h_fg.Values,'g');
xlabel('Feature Index');
ylabel('Probability Density');
title('Overlapping Joint Distributions');
savefig(joint_hist_overlap,'BG_PDF');
hold off
error_vals =[];

for i=1:length(joint_histb.YData)
    if joint_histb.YData(i) ~= 0 && joint_histc.YData(i) ~= 0
        if joint_histb.YData(i) < joint_histc.YData(i)
            error_vals = [error_vals joint_histb.YData(i)];
        elseif joint_histb.YData(i) > joint_histc.YData(i)
            error_vals = [error_vals joint_histc.YData(i)];
        end
    else
        error_vals = [error_vals 0];
    end
end

figure
risk_plot=bar(error_vals,'y');
xlabel('Feature Index');
ylabel('Probability Density');
title('Risk Distribution');

%probability of error of algorithm%
error_probability = sum(error_vals); %0.1527

%actual error
cheetah_mask = double(imread('cheetah_mask.bmp')/255);

counter_correct = 0;
total_pixels = cheetah_rows*cheetah_cols;
for i=1:cheetah_rows
    for j=1:cheetah_cols
        if cheetah_mask(i,j) ==  new_image(i,j)
            counter_correct = counter_correct + 1;
        end
    end
end

percent_correct = counter_correct/total_pixels*100; %82.22 - Error pct=100-82.22=17.78
```