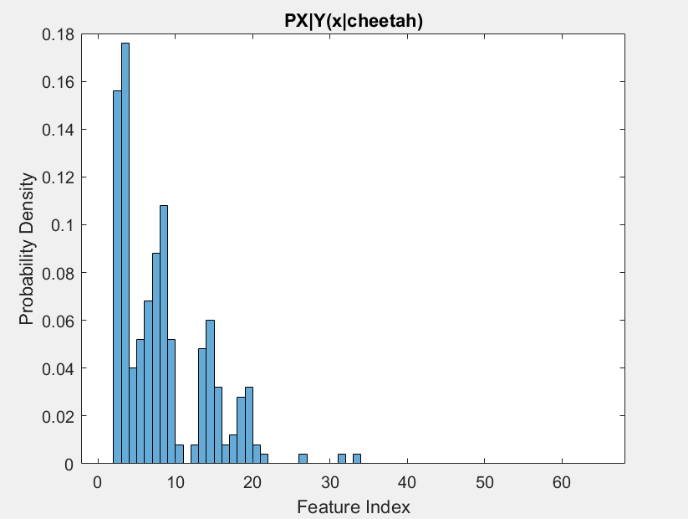
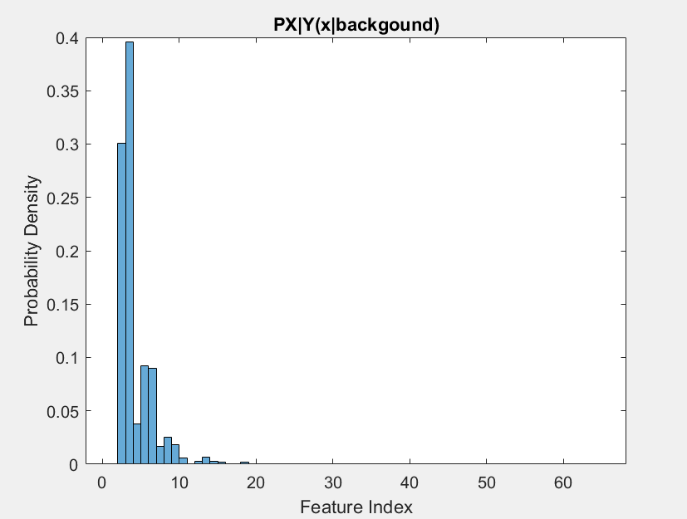
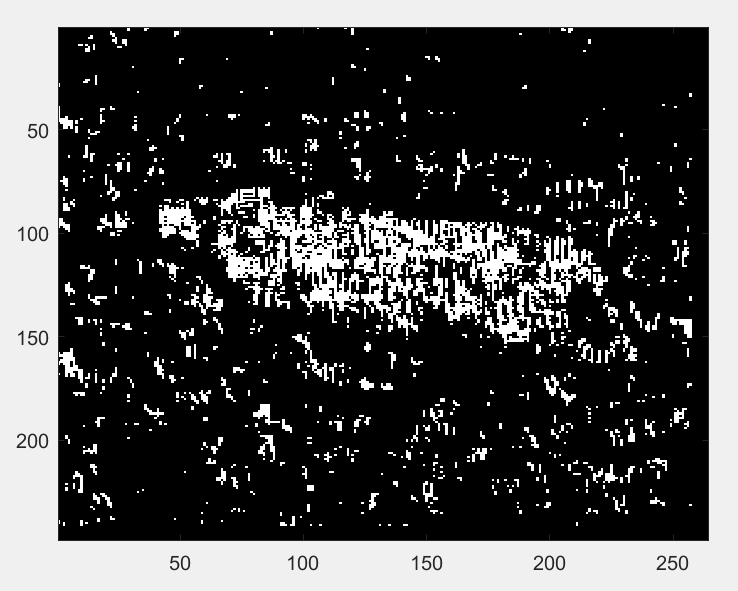
**a)** Reasonable estimates for the prior probabilities are 0.1919 and 0.8081 for the cheetah and background respectively. These values were calculated by dividing the number of elements in the training data for each type by the total number of elements of both sets of training data.

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



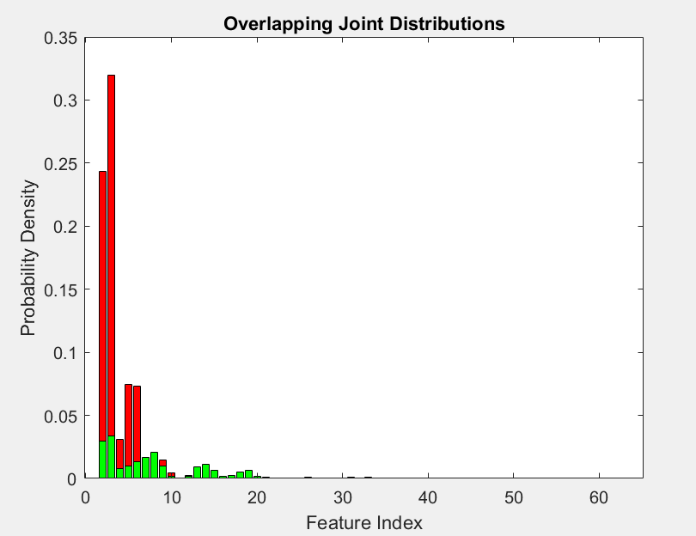
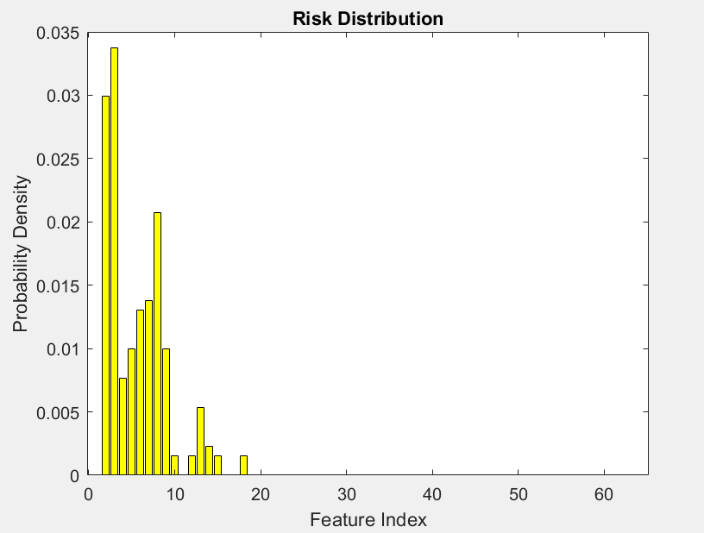
*Figure 1: Histogram of PX|Y (x|cheetah) Figure 2: Histogram of PX|Y (x|grass)*

**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