ECE 271A Statistical Learning HW1

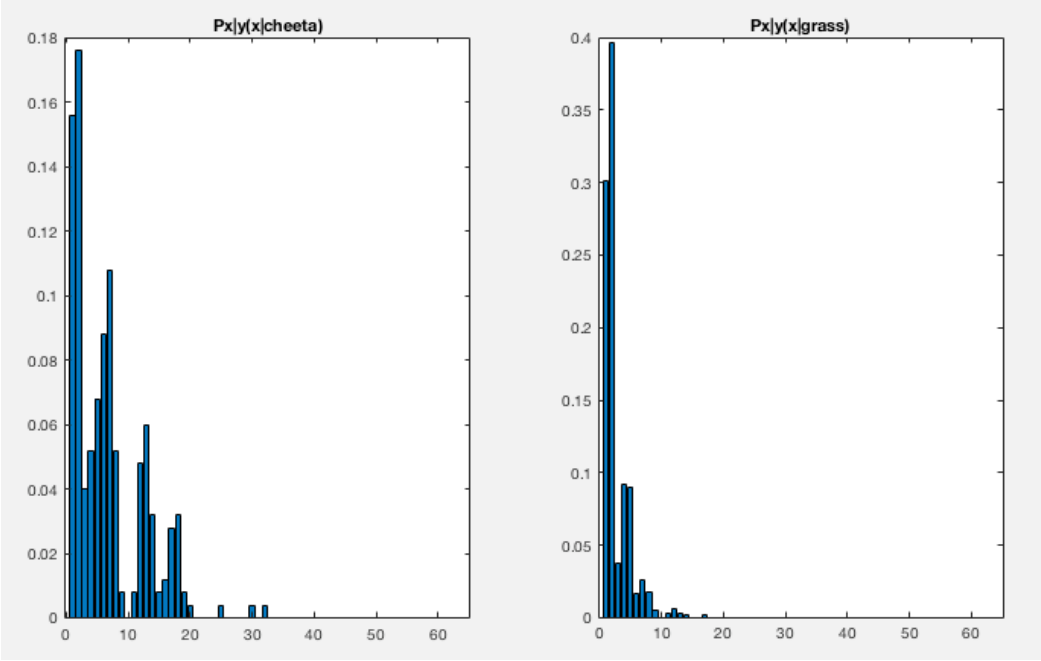
a) In this problem we need to compute the prior probabilities of cheetah and grass from the training dataset, the results are showed below:

$P_Y(\text{cheetah}) = \frac{\#\ of\ Cheetah\ samples}{\#\ of\ Cheetah\ samples + \#\ of\ grass\ samples} = 0.1919$

$P_Y(\text{grass}) = \frac{\#\ of\ Cheetah\ samples}{\#\ of\ Cheetah\ samples + \#\ of\ grass\ samples} = 0.8081$

b) For this problem, we need to use function dct2 to obtain the DCT coefficient of every 8*8 pixels from the training dataset. After that, take out the second largest coefficient and get it's index. At last plot the probabilities in two histogram to represent $P_{X|Y}(x|\text{cheetah})$ and $P_{X|Y}(x|\text{grass})$ respectively.

The histogram of $P_{X|Y}(x|\text{cheetah})$ and $P_{X|Y}(x|\text{grass})$



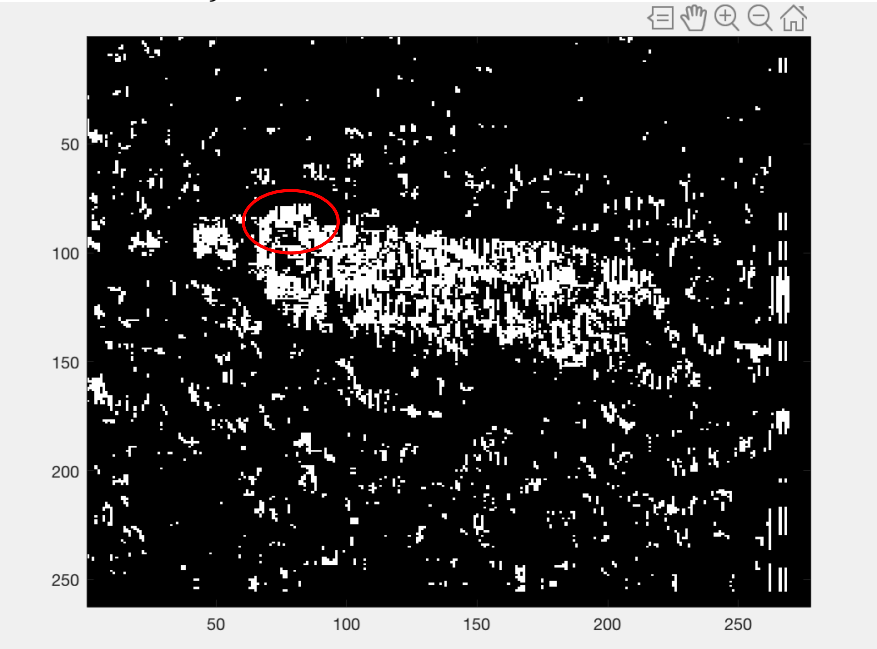
c) In this problem we need to get the picture of the result that the computer learned to classify the background and the foreground.
As we know if $P_{Y|X}(\text{cheetah}|x) > P_{Y|X}(\text{grass}|x)$, the computer will classify the pixel as the "cheetah". However, we cannot compute $P_{Y|X}(\text{cheetah}|x)$ and $P_{Y|X}(\text{grass}|x)$ easily, so we use the Bayes rule to turn the problem into other variables that can easily be computed.
By Bayes rule:

$$P_{Y|X}(\text{cheetah}|x) = \frac{P_{X|Y}(x|\text{cheetah}) * P_Y(\text{cheetah})}{P_X(x)}$$

So we if we want to compute $P_{Y|X}(\text{cheetah}|x) > P_{Y|X}(\text{grass}|x)$
We can rewrite it as below:

$$\frac{P_{X|Y}(x|\text{cheetah}) * P_Y(\text{cheetah})}{P_X(x)} > \frac{P_{X|Y}(x|\text{grass}) * P_Y(\text{grass})}{P_X(x)}$$

Base on this we can use the results from a) and b) to get the result as below. (error rate 17.41%)



Follow up:
I tried to improve the performance, in the histogram there are a lot of features that their probabilities are zero. This will make

$$\frac{\mathrm{PX|Y(x|cheetah)} * \mathrm{PY(cheetah)}}{PX(x)} = \frac{\mathrm{PX|Y(x|grass)} * \mathrm{PY(grass)}}{PX(x)} = 0$$

(because both PX|Y(x|cheetah) and PX|Y(x|grass) are zero)
In this case I take the result of pixels around this pixel into consideration, if the pixels around are mostly white, then it the pixel should be white. Same as the black condition.
Due to this operation, I improve the error rate from 17.68 % to 17.41%.
The performance with worse result: (error rate 17.68%)

d)

Probability of Error:

$$\text{error(FG)} = \frac{\text{\#FG pixels misclassified as BG}}{\#FG\ pixels\ in\ ground\ truth\ of\ test\ set} \times prior\ probability\ of\ FG$$

Error of ground truth Foreground misclassified as Background,
its rate is 4.79%.

$$\text{error(BG)} = \frac{\text{\#BG pixels misclassified as FG}}{\#BG\ pixels\ in\ ground\ truth\ of\ test\ set} \times prior\ probability\ of\ BG$$

Error of ground truth Background misclassified as Foreground,
its rate is 12.62%

The total error rate is 4.79 + 12.62 % = 17.41%.

Code:

```
clear all;
%% TrainingSamplesDCT_8.mat
%compute the priors Py(cheeta) & Py(grass)
load('TrainingSamplesDCT_8.mat');
p_background = size(TrainsampleDCT_BG,1) / (size(TrainsampleDCT_BG,1)
+ size(TrainsampleDCT_FG,1)); % PY(grass)
p_foreground = 1 - p_background; %PY(cheeta)

%compute class-conditioned Px|y(x|cheeta)
index_cheeta = zeros(1,64);
TrainsampleDCT_FG(:,1) = [];
for ci = 1: size(TrainsampleDCT_FG,1)
    [VC,C] = max(abs(TrainsampleDCT_FG(ci,:)));
    index_cheeta(C) = index_cheeta(C) + 1;
end
index_cheeta = index_cheeta / size(TrainsampleDCT_FG,1);

%compute class-conditioned Px|y(x|grass)
index_grass = zeros(1,64);
TrainsampleDCT_BG(:,1) = [];
for gi = 1: size(TrainsampleDCT_BG,1)
    [VG,G] = max(abs(TrainsampleDCT_BG(gi,:)));
    index_grass(G) = index_grass(G) + 1;
end
index_grass = index_grass / size(TrainsampleDCT_BG,1);
 %plot on histogram
figure(1);
subplot(1,2,1);
bar(index_cheeta);
title('Px|y(x|cheeta)');

subplot(1,2,2);
bar(index_grass);
title('Px|y(x|grass)');

%If Py|x(cheeta|x) > Py|x(grass|) ----> the pixel is cheeta
%same as Px|y(x|cheeta)Py(cheeta) > Px|y(x|grass)Py(grass)
```

```
%input bmp file & ZigZag pattern
A = imread('cheetah.bmp');
Z = load('Zig-Zag Pattern.txt');
%% original size 255*270 ---> need to use padarray() to fill 255+7,
270+7
B = padarray(A,[7,7],'symmetric','post');
[q,l] = size(B);
matrix_zigzag=[];
mask = zeros(q,l);
%% make the picture into 8*8 pixels and use mask determine which
pixel is cheeta/grass
for i=1:q-7
    for j=1:l-7
        matrix_dct2 = dct2(B(i:i+7,j:j+7));
        matrix_zigzag(Z+ones(8,8)) = matrix_dct2; %store matrix_dct
into zigzag which is from 0~63 so plus a ones(8,8)
        [V,I] = max(abs(matrix_zigzag(2:64)));
        if index_cheeta(I)*p_foreground > index_grass(I)*p_background
            mask(i,j) = 1;
        elseif index_cheeta(I)*p_foreground <
index_grass(I)*p_background
            mask(i,j) = 0;
        else     %if index_cheeta(I) == index_grass(I) == 0 ---> use
outer 2*2 to classify
            white = 0;
            black = 0;
            for z=i-2:i+2
                for w=j-2:j+2
                    if mask(z,w)== 1
                        white = white + 1;
                    elseif mask(z,w) == 0
                        black = black + 1;
                    end
                end
            end
            if black > white
              mask(i,j) = 1;
            else
```

```
            mask(i,j) = 0;
        end
    end
end
figure(2);
imagesc(mask);
colormap(gray(255));

%% compute error rate
diff_grass = 0;
diff_cheeta = 0;
count_grass = 0;
count_cheeta = 0;

groundtruth_mask = im2double(imread('cheetah_mask.bmp'));
for i=1:(q-7)
    for j=1:(l-7)
        if mask(i,j) == 1
            if groundtruth_mask(i,j) == 0
                diff_cheeta = diff_cheeta + 1;
            else
                diff_cheeta = diff_cheeta;
            end
        elseif mask(i,j) == 0
            if groundtruth_mask(i,j) == 1
                diff_grass = diff_grass + 1;
            else
                diff_grass - diff_grass;
            end
        end
     end
end

for i=1:(q-7)
    for j=1:(l-7)
        if groundtruth_mask(i,j) == 1
            count_cheeta = count_cheeta + 1;
```

```
        else
            count_grass = count_grass + 1;
        end
    end
end
error_cheeta = (diff_cheeta / count_cheeta)*p_foreground ;
error_grass = (diff_grass / count_grass)*p_background;
error_rate = error_cheeta + error_grass;
```