Kushagra Agrawal
A53264533

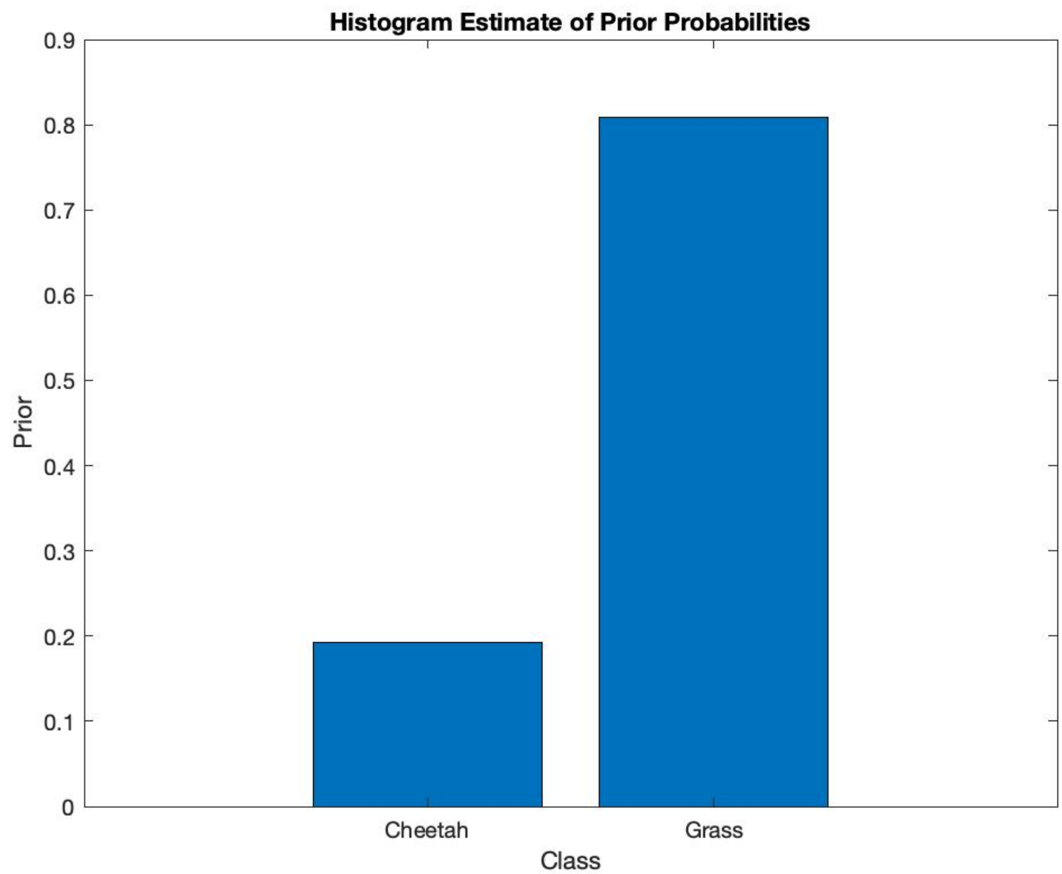
Homework 2

Problem 5

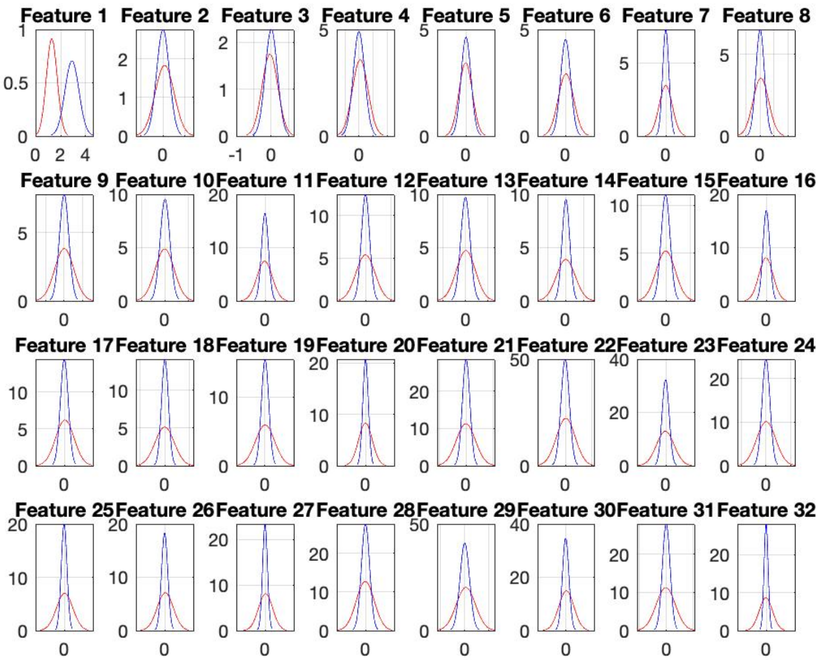
1. Using the results of Problem **2**, the n independent observations of different classes of prior probabilities form a random variable X such that $P_x(k) = \pi_k, k \in \{cheetah, grass\}$. $P_x(k)$ is calculated as $P_x(k) = c_k/n$, where c_k is the number of times the observed value is k. Using this, the prior probabilities can be calculated as follows:

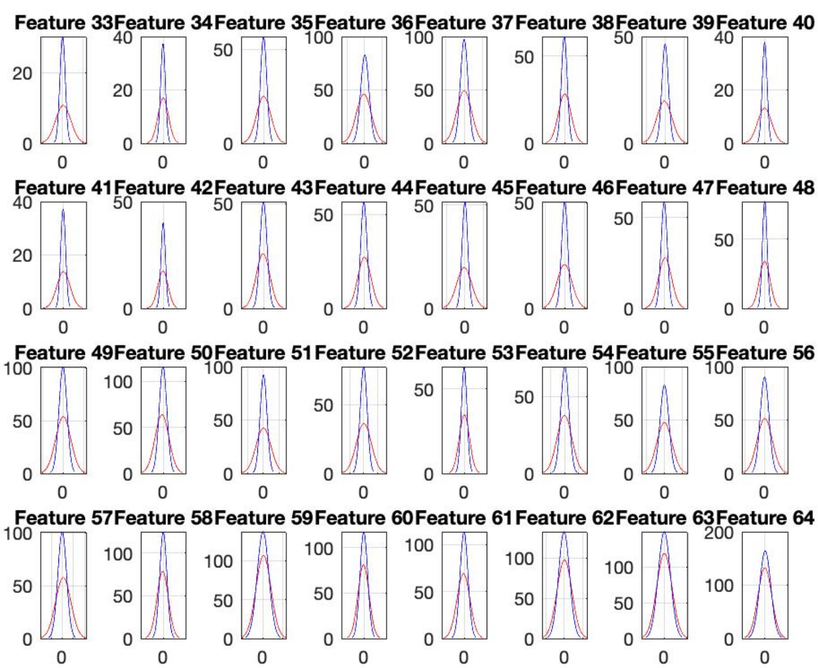
$$P(Y|cheetah) = \tfrac{250}{250+1053} = 0.1918 \quad P(Y|grass) = \tfrac{1053}{250+1053} = 0.8081$$

   The above calculation is exactly same as the one in Homework 1. In homework 1, I had used the frequencies of the classes in the training data to calculate prior probability.

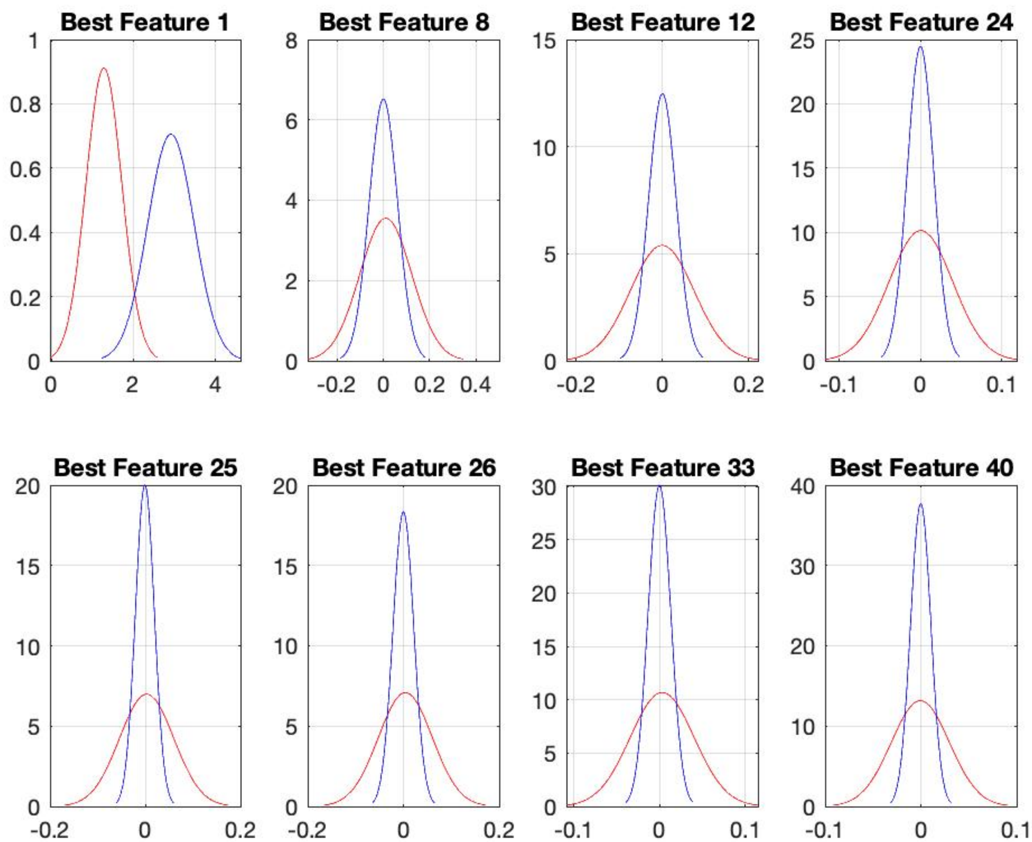


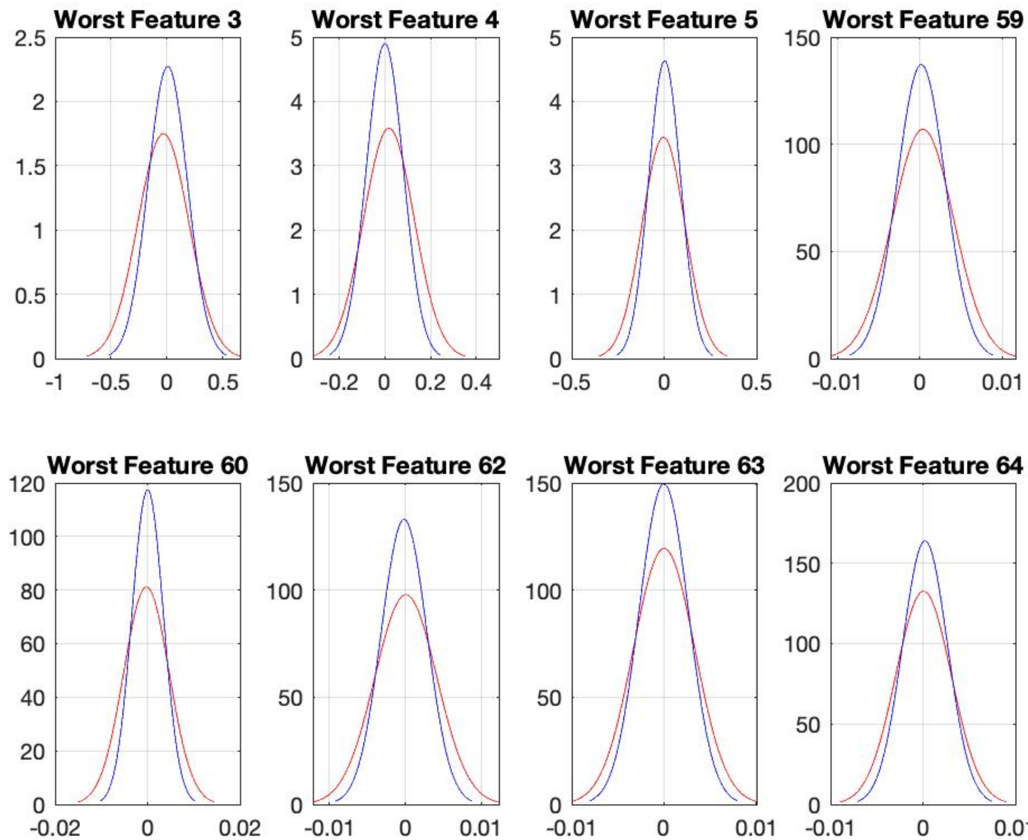
2. Using the training data provided and under the Gaussian assumption, I calculated the mean and standard deviation for each of the 64 features. The marginal densities for each of the 64 features are as below. The red line illustrates *cheetah* class and blue line illustrates *grass* class.


Feature 1
Feature 2
Feature 3
Feature 4
Feature 5
Feature 6
Feature 7
Feature 8
Feature 9
Feature 10
Feature 11
Feature 12
Feature 13
Feature 14
Feature 15
Feature 16
Feature 17
Feature 18
Feature 19
Feature 20
Feature 21
Feature 22
Feature 23
Feature 24
Feature 25
Feature 26
Feature 27
Feature 28
Feature 29
Feature 30
Feature 31
Feature 32


Feature 33
Feature 34
Feature 35
Feature 36
Feature 37
Feature 38
Feature 39
Feature 40
Feature 41
Feature 42
Feature 43
Feature 44
Feature 45
Feature 46
Feature 47
Feature 48
Feature 49
Feature 50
Feature 51
Feature 52
Feature 53
Feature 54
Feature 55
Feature 56
Feature 57
Feature 58
Feature 59
Feature 60
Feature 61
Feature 62
Feature 63
Feature 64

Analyzing the above marginal densities visually, I roughly picked the 8 best and 8 worst densities by checking the distribution spread for the densities. This led me to choosing [1,8,12,24,25,26,33,40] as the best 8 features and [3,4,5,59,60,62,63,64] as the worst 8 features. They are plotted below.



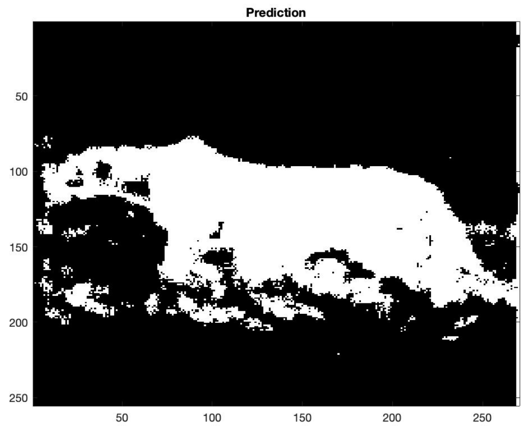


3. Using the multivariate gaussian equation taught in class, I classified each 8x8 block in the cheetah image into the two classes using 64 and the 8 best features calculated earlier. The image was padded by replicating the border pixels in the top left. The error probability and predicted maps are on the next page. The probability of error is calculated using the following equation:

$$P(Error) = P_{X|Y}(grass|cheetah) * P_Y(Cheetah) + P_{X|Y}(cheetah|grass) * P_Y(grass)$$

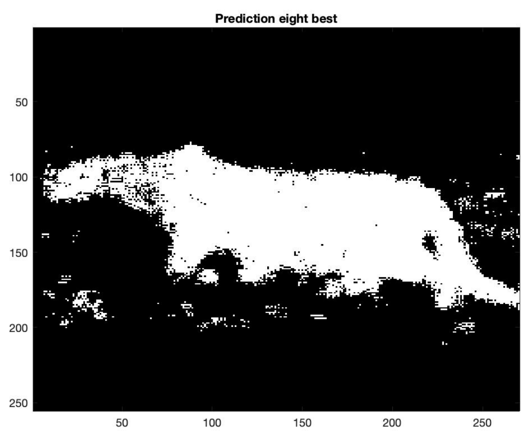
   (a) Using the 64 features Gaussian.

$$\begin{aligned} P(Error) &= 0.0753 * 0.1919 + 0.0957 * 0.8081 \\ &= 0.0918 \end{aligned}$$



(b) Using the 8 features Gaussian

$$\begin{aligned} P(Error) &= 0.0899 * 0.1919 + 0.0361 * 0.8081 \\ &= 0.0464 \end{aligned}$$



The above two figures indicate that using more features doesn't always result in a better classification. Here, I got a better mask and lower probability of error using the 8 best features as compared to using all the 64 features.

MATLAB code

```
clear
clc

trainSample = load('TrainingSamplesDCT_8_new.mat');
fgSamples = trainSample.TrainsampleDCT_FG;
bgSamples = trainSample.TrainsampleDCT_BG;

fgSamplesDim = size(fgSamples);
bgSamplesDim = size(bgSamples);

% fgSamples has 250 training examples of 64 features each
% bgSamples has 1053 training examples of 64 features each

priorYCheetah = fgSamplesDim(1) / (fgSamplesDim(1) +
    bgSamplesDim(1));
priorYGrass = bgSamplesDim(1) / (fgSamplesDim(1) +
    bgSamplesDim(1));

figure;
str = {'Cheetah','Grass'};
bar([priorYCheetah,priorYGrass]);
title('Histogram Estimate of Prior Probabilities');
xlabel('Class');
ylabel('Prior');
set(gca, 'XTickLabel',str);

disp('prior probability of Cheetah');
disp(priorYCheetah);
disp('prior probability of Grass');
disp(priorYGrass);

fgFeatureMean = sum(fgSamples) / fgSamplesDim(1);
bgFeatureMean = sum(bgSamples) / bgSamplesDim(1);

fgFeatureStd = std(fgSamples);
bgFeatureStd = std(bgSamples);

for i = 1:fgSamplesDim(2)
    marginFG(:, i) = [(fgFeatureMean(i) - 3*fgFeatureStd(i) :
        fgFeatureStd(i) / 50 : fgFeatureMean(i) + 3*
```

```
        fgFeatureStd(i))];
    fgGaussian(:, i) = calculateGaussian(marginFG(:, i),
        fgFeatureMean(i), fgFeatureStd(i));
end
for i = 1:bgSamplesDim(2)
    marginBG(:, i) = [(bgFeatureMean(i) - 3*bgFeatureStd(i) :
        bgFeatureStd(i) / 50 : bgFeatureMean(i) + 3*
        bgFeatureStd(i))];
    bgGaussian(:, i) = calculateGaussian(marginBG(:, i),
        bgFeatureMean(i), bgFeatureStd(i));
end

for i = 1:2
    figure;
    for j = 1:32
        subplot(4,8, j);
        plot(marginFG(:, (i-1)*32 + j), fgGaussian(:, (i-1)*32
            + j), '-r', marginBG(:, (i-1)*32 + j), bgGaussian
            (:, (i-1)*32 + j), '-b');
        grid on;
        title(['Feature ',num2str((i-1)*32+j)]);
    end
end

best_eight_features = [1,8,12,24,25,26,33,40];
worst_eight_features = [3,4,5,59,60,62,63,64];

figure;
for i = 1:8
    subplot(2, 4, i);
    plot(marginFG(:, best_eight_features(i)), fgGaussian(:,
        best_eight_features(i)), '-r', ...
        marginBG(:, best_eight_features(i)), bgGaussian(:,
            best_eight_features(i)), '-b');
    grid on;
    title(['Best Feature ', num2str(best_eight_features(i))]);
end

figure;
for i = 1:8
    subplot(2, 4, i);
    plot(marginFG(:, worst_eight_features(i)), fgGaussian(:,
```

```
        worst_eight_features(i)), '-r', ...
         marginBG(:, worst_eight_features(i)), bgGaussian(:, worst_eight_features(i)), '-b');
    grid on;
    title(['Worst Feature ', num2str(worst_eight_features(i))]);
end

original_Image = imread('cheetah.bmp');
pad_Image = padarray(original_Image, [7 7], 'replicate', 'pre');
imageModified = im2double(pad_Image);
[image_row, image_col] = size(imageModified);

fgFeatureCov_64 = cov(fgSamples);
determinant64FeaturesFG = det(fgFeatureCov_64);
bgFeatureCov_64 = cov(bgSamples);
determinant64FeaturesBG = det(bgFeatureCov_64);

fgSamples_eight_dim = fgSamples(:, best_eight_features);
bgSamples_eight_dim = bgSamples(:, best_eight_features);
fgFeatureCov_8 = cov(fgSamples_eight_dim);
determinant8FeaturesFG = det(fgFeatureCov_8);
bgFeatureCov_8 = cov(bgSamples_eight_dim);
determinant8FeaturesBG = det(bgFeatureCov_8);
fgFeatureMean_8 = sum(fgSamples_eight_dim) / fgSamplesDim(1);
bgFeatureMean_8 = sum(bgSamples_eight_dim) / bgSamplesDim(1);

alphaFG = log(((2*pi)^64) * determinant64FeaturesFG) - 2*log(priorYCheetah);
alphaBG = log(((2*pi)^64) * determinant64FeaturesBG) - 2*log(priorYGrass);

alphaFG_eight = log(((2*pi)^8) * determinant8FeaturesFG) - 2*log(priorYCheetah);
alphaBG_eight = log(((2*pi)^8) * determinant8FeaturesBG) - 2*log(priorYGrass);

% create feature vector
zigzagPattern = load('Zig-Zag Pattern.txt');
zigzagPattern = zigzagPattern + 1; % 1 indexing in MATLAB

```

```
calculatedMask = zeros(image_row - 7, image_col - 7);
calculatedMask_eight = zeros(image_row - 7, image_col - 7);
% index = 1;
for i = 1:image_row - 7
    for j = 1:image_col - 7
        block = imageModified(i:i+7, j: j+7);
        dctOutput = dct2(block);
        orderedDCTOutput(zigzagPattern(:)) = dctOutput(:);
        dctOutput_eight = orderedDCTOutput(:,
            best_eight_features);
        calculatedMask_eight(i, j) = calculateMask(
            dctOutput_eight, fgFeatureMean_8, bgFeatureMean_8,
            fgFeatureCov_8, bgFeatureCov_8, alphaFG_eight,
            alphaBG_eight);
        calculatedMask(i, j) = calculateMask(orderedDCTOutput,
             fgFeatureMean, bgFeatureMean, fgFeatureCov_64,
            bgFeatureCov_64, alphaFG, alphaBG);
    end
end

figure;
imagesc(calculatedMask);
title('Prediction');
colormap(gray(255));
figure;
imagesc(calculatedMask_eight);
title('Prediction eight best');
colormap(gray(255));


groundTruth = imread('cheetah_mask.bmp');
groundTruthModified = im2double(groundTruth);

groundTruthFGCount = 0;
groundTruthBGCount = 0;
for i = 1 : image_row - 7
    for j = 1 : image_col - 7
        if groundTruthModified(i, j) == 1
            groundTruthFGCount = groundTruthFGCount + 1;
        else
            groundTruthBGCount = groundTruthBGCount + 1;
        end
```

```matlab
    end
end

[error_FG_eight, error_BG_eight] = calculateErrorCount(groundTruthModified, calculatedMask_eight, image_row - 7, image_col - 7);
[error_FG, error_BG] = calculateErrorCount(groundTruthModified, calculatedMask, image_row - 7, image_col - 7);


fgError = error_FG / groundTruthFGCount;
bgError = error_BG / groundTruthBGCount;

fgError_eight = error_FG_eight / groundTruthFGCount;
bgError_eight = error_BG_eight / groundTruthBGCount;

probError = (fgError * priorYCheetah) + (bgError * priorYGrass);
probError_eight = (fgError_eight * priorYCheetah) + (bgError_eight * priorYGrass);

disp('Probability of Error');
disp(probError);

disp('Probability of Error for eight features');
disp(probError_eight);

function mask = calculateMask(dctOutput, meanFG, meanBG, fgCov, bgCov, alphaFG, alphaBG)
    mahalanobisFG = (dctOutput - meanFG) * inv(fgCov) * transpose(dctOutput - meanFG);
    mahalanobisBG = (dctOutput - meanBG) * inv(bgCov) * transpose(dctOutput - meanBG);
    if mahalanobisFG + alphaFG < mahalanobisBG + alphaBG
        mask = 1;
    else
        mask = 0;
    end
end

function [fgCount, bgCount] = calculateErrorCount(groundTruthModified, mask, image_row, image_col)
```

```
    errorFGCount = 0; % false negative
    errorBGCount = 0; % false positive
    for i = 1:image_row
        for j = 1:image_col
            if mask(i,j) ==0 && groundTruthModified(i, j) == 1
                errorFGCount = errorFGCount + 1;
            elseif mask(i,j) == 1 && groundTruthModified(i, j)
                == 0
                errorBGCount = errorBGCount + 1;
            end
        end
    end
    fgCount = errorFGCount;
    bgCount = errorBGCount;
end

function g = calculateGaussian(x, mu, sigma)
    g = (1./(sqrt(2*pi) * sigma)).*exp(-(x - mu).^2./(2*sigma
        .^2));
end
```