Arda Cankat Bati

ECE 271A Homework #2, Computer Assignment

Std Id = A53284500

All output from the code (printed values, plots etc.) are included at the end of the report. They will be referenced in the report by name.

ANSWERS

PART A

In problem 2, we have a multinomial distribution. We have random variable X from which we draw n independent observations. X can take values from 1 up to n. We also have a random vector C, which contains the number of times each X_i was observed.

$$P_{C_1,\ldots,C_N}(c_1,\ldots,c_N) = \frac{n!}{\prod_{k=1}^{N} c_k!} \prod_{j=1}^{N} \pi_j^{c_j}.$$

To estimate the probability that X takes value i (probability: π_i), an estimator for π_i is derived through ML estimation. The result found in problem 2 is:

$$\pi_j^* = \frac{c_j}{n}.$$

In which we divide the number of times we observed each value (c_j) by the total number of observations. For the cheetah case, we have a binomial distribution, which is a special case of the multinomial distribution. This time our random variable X takes two values 0 and 1. If we set the N value as N=2 in problem 2, we would be solving for the binomial case. Therefore, the result obtained for the prior probability estimator in problem 2, is also the same for our case.

With this information we can calculate the priors as:

P(X = Cheetah) = (#Of Cheetah Data Points / # Total Data Points)

= 250/1303 = 0.1918649 = 0.1919 (roughly)

And similarly, for No Cheetah points:

P(X = No Cheetah) = (#Of No Cheetah Data Points / # Total Data Points)

= 1053/1303 = 0.8081351 = 0.8081 (roughly)

We also get the same result from the normalized histogram estimation, as seen in the Histogram Estimation plot (at the end of the report). This is the same calculation I did by common sense in the 1st Homework. Although I did not do the calculations of problem 2 then, I arrived at this result by common intuition and experience. Just as in problem 2, I assumed the distribution to be sampled from n different random variables, independent and with identical distribution (although I did not state this explicitly, it was more heuristic). I reasoned that for this case the estimator for the priors should be the number of observations divided by the total number of points, as all observations have equal value. (again I did not argue this explicitly in the report.)

PART B

The required plots were drawn in separate 2 by 2 subplots to take less space but at the same time conserve visibility. With 64 original plots, 8 best features and 8 worst features, a total of 80 plots were drawn. All subplots have an individual title revealing which feature they are representing, and an overall title is given for the 2 by 2 plot groups.

Each plot represents marginal distributions of feature X_k (k = 1,…,64), from the zigzag patterned DCT coefficients from the train data. There are two graphs in each subplot representing the foreground (cheetah) and background (no cheetah) cases. These marginal plots are color coded as:

$P_{X_k|Y}(x_k|\text{cheetah})$ --> Red & Straight lines

$P_{X_k|Y}(x_k|\text{no cheetah})$ --> Black & Dashed lines

The best and worst features were found by visual inspection of the 64 plots. The first observation I made was in general, the marginal distributions for (Y = cheetah) case had higher variance compared to the (Y = no cheetah) case. I also made the same observation in HW#1, reasoning that the camouflage pattern of the cheetah has broader frequency content than plain grass background. Therefore, this may be causing the DCT coefficients to be more varied for the Y = cheetah data points in general.

While in general the two class cases have similar means, the variances make the marginal distributions discernable from each other. (Apart from feature 1, which was clearly separate means for each class). I chose the best features by visual inspection, trying to find the most

discernable distributions. Also, I tried not to select too many consecutive features, but to choose from a broader set, as each feature represents a certain frequency. For this decision my reasoning was that the cheetah's camouflage pattern has different frequencies in it, and to discern them I need the corresponding features. For the worst features I just selected the ones in which the marginal densities look mostly similar (both mean and variance wise). The resulting features I chose were:

Best Features = [1 18 19 25 27 32 33 40]

Worst Features = [3 4 5 59 60 62 63 64]

For each feature above, the corresponding marginals were plotted with the same color coding stated before. The resulting plots are given at the end of the report.

For the estimations of the mean and variance the following formulas from the lecture notes were used:

$$\mu_i = \frac{1}{n}\sum_j x_j^{(i)} \qquad \Sigma_i = \frac{1}{n}\sum_j (x_j^{(i)} - \mu_i)(x_j^{(i)} - \mu_i)^T$$

For the scalar Gaussian case, the ML estimators were found in the lectures as:

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i = \bar{x} \qquad \hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \bar{x})^2$$

We also should determine that the Hessian matrix is negative definite for the Gaussian case Therefore we will know that we have a maximum point. The hessian matrix will be:

$$\nabla^2 f(X) = \begin{bmatrix} \frac{\partial^2 f}{\partial X_0^2}(X) & \cdots & \frac{\partial^2 f}{\partial X_0 \partial X_{n-1}}(X) \\ & \vdots & \\ \frac{\partial^2 f}{\partial X_{n-1} \partial X_0}(X) & \cdots & \frac{\partial^2 f}{\partial X_{n-1}^2}(X) \end{bmatrix}$$

Where f(x) is the log likelihood, x_0 will be the mean, x_1 will be the variance. The log likelihood function derived in the lectures was:

$$\Lambda = \ln L = -\frac{N}{2}\ln(2\pi) - N\ln\sigma_T - \frac{1}{2}\sum_{i=1}^{N}\left(\frac{T_i - \bar{T}}{\sigma_T}\right)^2$$

After proving this for the scalar case, we can generalize to the vector case.

PART C

For this part we our model of the distribution will be:

1) a 64-dimensional Gaussian composed of 64 total features

2) an 8-dimensional Gaussian (8 dimensions corresponding to the best 8 features)

For both cases the following Bayesian decision can be used to determine the class as i_1=cheetah or i_2=no cheetah:

The Gaussian classifier

- BDR can be written as

$$i^*(x) = \arg\min_i \left[d_i(x, \mu_i) + \alpha_i\right]$$

with

$$d_i(x, y) = (x - y)^T \Sigma_i^{-1} (x - y)$$

$$\alpha_i = \log(2\pi)^d \left|\Sigma_i\right| - 2\log P_Y(i)$$

The decisions are made separately for each pixel, using a sliding DCT window over the image. The estimated mean for the 64 and 8 dimensional cases were computed for the previous part, using the ML estimators. The covariance matrices are also produced with the corresponding formula from the ML estimator (given before in the report.) For the 64 and 8 best features cases, predicted images are formed, with the same procedure as in HW#1. These images are named decisionImage64 and decisionImage8 respectively.

The probability of error is calculated the same way as in HW#1 but for two separate decision images. First the true image mask is extracted from cheetah mask.bmp. The probability of error is calculated in each case as:

P(Error) = P(Y = No Cheetah) * P(False Positive) + P(Y = Cheetah) * P(False Negative)

Where

- P(Y = No Cheetah) -> calculated from the test image.
- P(Y = Cheetah) -> calculated from the test image
- P(False Positive) = Alpha -> calculated from the difference of test and decision images
- P(False Negative) = Beta -> calculated from the difference of test and decision images

The details of the calculation process were explained in the report for HW#1. Briefly, an error mask is used to determine the counts for True, False Negative and False Positive values. The results from the 64 dimensional and 8 dimensional (best features) cases are given below:

- True Prior for Cheetah is: 1.918519e-01
- True Prior for No Cheetah is: 8.081481e-01

- False Positive alpha for 64 Features is: 4.214518e-02
- False Negative beta for 64 Features is: 1.713983e-01
- Total Probability of Error for 64 Features is: 6.694263e-02

- False Positive alpha for best 8 features is: 1.958987e-03
- False Negative beta for best 8 features is: 4.954198e-01
- Total Probability of Error for best 8 features is: 9.663036e-02

Here the first observation to make is that both probabilities of error are better than a naïve classifier. A naïve classifier would go only with the priors and label everything as background, thus making errors with probability: 1.918519e-01.

Also, both probabilities of error are better than the one in HW#1 which was 1.718228e-01. In HW#1 we were only using 1 feature, as we have more features now, a better result is expected. This is because we have a better model representing the real distribution. The real distribution may still not be a Gaussian, but a multivariate Gaussian model seems to be a better approximation to it.

Comparing the two cases (64 and 8 features) we see that the model with the 64 features gave better results. (6.69% error compared to 9.66%) In the 64-dimensional case we have more features from the data and we give them all the same priority. There may be some irrelevant features that still contribute to the decision. In the 8-dimensional case, we ignore most features and only use the 1/8 of them. However, we still get a comparable error rate, even if we

discarded features. The main problem with the 8 feature classifier seems to be the lack of frequency content compared to 64 features.

Here there is a trade-off between discarding features and using relevant features. If we discard too many features to get the relevant ones, we end up with a higher error, as we are losing content from the data. If we use too many features, then the irrelevant features could become a majority and dominate the result, therefore increasing the error. In our case, maybe by choosing more features (for example 16 best features), and choosing them with a mathematical method would give better results. An even better method would be to do a feature transformation on the data to reduce the number of dimensions.

Finally, there is a big difference between the false positive (alpha) and false negative (beta) values of the 64 and 8 feature classifiers. The 64-feature classifier has higher # false positives and lower # false negatives than the 8-feature classifier. Therefore, it is more likely to label grass pixels as cheetah. The 8-feature case, on the other hand, is more likely to label cheetah pixels as grass. This may be because we are ignoring features in the 8-feature case, therefore ignoring some frequencies. In the end, we get a slightly less noisy image as features are more relevant. However, as we have less frequency content, the outline of the cheetah is not that sharp, and we have holes in the cheetah decision image. For the 64-feature case, the outline of the cheetah is clearer, and the overall cheetah image doesn't have many holes. However, as we are including many irrelevant features, there is some more noise, patches of the grass are mislabeled as cheetah.

```
%**********************
% Arda Cankat Bati
% ECE 271A - Homework#2
%**********************

clear
clc
load('TrainingSamplesDCT_8_new.mat');

%*********** GETTING THE DATA AND ESTIMATING PRIORS **********

FG = TrainsampleDCT_FG;
BG = TrainsampleDCT_BG;
FG_size = size(FG,1);
BG_size = size(BG,1);
sampleSize = FG_size + BG_size;

sampleCount = zeros(sampleSize,1);
sampleCount(1:FG_size) = 1;
hist = histogram(sampleCount);
arg = (hist.Values)/(sampleSize);
bar([0 1], arg,0.2)
xlim([-0.5 1.5]);
title('Histogram estimation, x = 0 is BG, x = 1 is FG')
xlabel('X'); ylabel('Histogram count of the class FG or BG')

CPrior = FG_size/(sampleSize);
fprintf('"Cheetah prior" estimated from the training data is: %d
\n',CPrior);
NCPrior = BG_size/(sampleSize);
fprintf('"No Cheetah prior" estimated from the training data is: %d
\n',NCPrior);

%*********************************************************


%******** SAMPLE MEAN & VARIANCE CALCULATION**********

% Sample Mean calculation
sampleMeanFG = zeros(64,1);
sampleMeanBG = zeros(64,1);
for i = 1:64
   sampleMeanFG(i,1) = sum(FG(:,i),1)/(FG_size);
   sampleMeanBG(i,1) = sum(BG(:,i),1)/(BG_size);
end

% Sample Variance calculation
sampleVarFG = zeros(64,1);
sampleVarBG = zeros(64,1);

%Covariance matrix calculation for 64 dimensional prob. distribution
CovMtxFG64 = zeros(64,64);
```

```
CovMtxBG64 = zeros(64,64);

for i = 1:FG_size
    CovMtxFG64 = CovMtxFG64 + ((FG(i,:)' - sampleMeanFG))*(((FG(i,:) -
 sampleMeanFG')));
end

for i = 1:BG_size
    CovMtxBG64 = CovMtxBG64 + ((BG(i,:)' - sampleMeanBG))*(((BG(i,:) -
 sampleMeanBG')));
end

CovMtxFG64 = CovMtxFG64/FG_size;
CovMtxBG64 = CovMtxBG64/BG_size;

%************************************************


%************ DRAWING THE REQUIRED PLOTS **************

%For all 64 Features plotting the estimated marginal densities
count = 1;
for i = 1:16
    figure()
    sgtitle('All 64 Features Marginals')
    for j = 1:4
        meanFG = sampleMeanFG(count);
        varFG = CovMtxFG64(count,count); stdFG = sqrt(varFG);
        meanBG = sampleMeanBG(count);
        varBG = CovMtxBG64(count,count); stdBG = sqrt(varBG);
        x1 = min(meanFG - 3*stdFG, meanBG - 3*stdBG);
        x2 = max(meanFG + 3*stdFG, meanBG + 3*stdBG);
        x = linspace(x1,x2,100);
        GaussianFG = (1/sqrt(2*pi*varFG))*exp(-((x-meanFG).^2)/
(2*varFG));
        GaussianBG = (1/sqrt(2*pi*varBG))*exp(-((x-meanBG).^2)/
(2*varBG));
        subplot(2,2,j)
        plot(x,GaussianFG, 'r-');
        hold on
        title(sprintf('Feature: %d, red-FG, black-BG',count))
        xlabel('X');
        ylabel('P(X|Y = i)');
        plot(x,GaussianBG, 'k-.');
        hold off
        count = count + 1;
    end
end

% By visual inspection from the graphs generated above,
% best and worst features are selected
BestFeatures = [1 18 19 25 27 32 33 40];
WorstFeatures = [3 4 5 59 60 62 63 64];
```

```
%For the best 8 Features plotting the estimated marginal densities
count = 1;
for i = 1:2
    figure()
    sgtitle('Best 8 Features Marginals')
    for j = 1:4
            idx = BestFeatures(count);
            meanFG = sampleMeanFG(idx);
            stdFG =  sqrt(CovMtxFG64(idx,idx));
            meanBG = sampleMeanBG(idx);
            stdBG =  sqrt(CovMtxBG64(idx,idx));
            x1 = min(meanFG - 3*stdFG, meanBG - 3*stdBG);
            x2 = max(meanFG + 3*stdFG, meanBG + 3*stdBG);
            x = linspace(x1,x2,100);
            GaussianFG = (1/sqrt(2*pi*stdFG))*exp(-((x-meanFG).^2)/
(2*(stdFG^2)));
            GaussianBG = (1/sqrt(2*pi*stdBG))*exp(-((x-meanBG).^2)/
(2*(stdBG^2)));
            subplot(2,2,j)
            plot(x,GaussianFG, 'r-')
            hold on
            plot(x,GaussianBG, 'k-.')
            xlabel('X');
            ylabel('P(X|Y = i)');
            title(sprintf('Feature: %d, red-FG, black-BG',idx))
            hold off
            count = count+1;
    end
end

%For the worst 8 Features plotting the estimated marginal densities
count = 1;
for i = 1:2
    figure()
    sgtitle('Worst 8 Features Marginals')
    for j = 1:4
        idx = WorstFeatures(count);
        meanFG = sampleMeanFG(idx);
        stdFG =  sqrt(CovMtxFG64(idx,idx));
        meanBG = sampleMeanBG(idx);
        stdBG =  sqrt(CovMtxBG64(idx,idx));
        x1 = min(meanFG - 3*stdFG, meanBG - 3*stdBG);
        x2 = max(meanFG + 3*stdFG, meanBG + 3*stdBG);
        x = linspace(x1,x2,100);
        GaussianFG = (1/sqrt(2*pi*stdFG))*exp(-((x-meanFG).^2)/
(2*(stdFG^2)));
        GaussianBG = (1/sqrt(2*pi*stdBG))*exp(-((x-meanBG).^2)/
(2*(stdBG^2)));
        subplot(2,2,j)
        plot(x,GaussianFG, 'r-')
        hold on
        plot(x,GaussianBG, 'k-.')
        xlabel('X');
        ylabel('P(X|Y = i)');
```

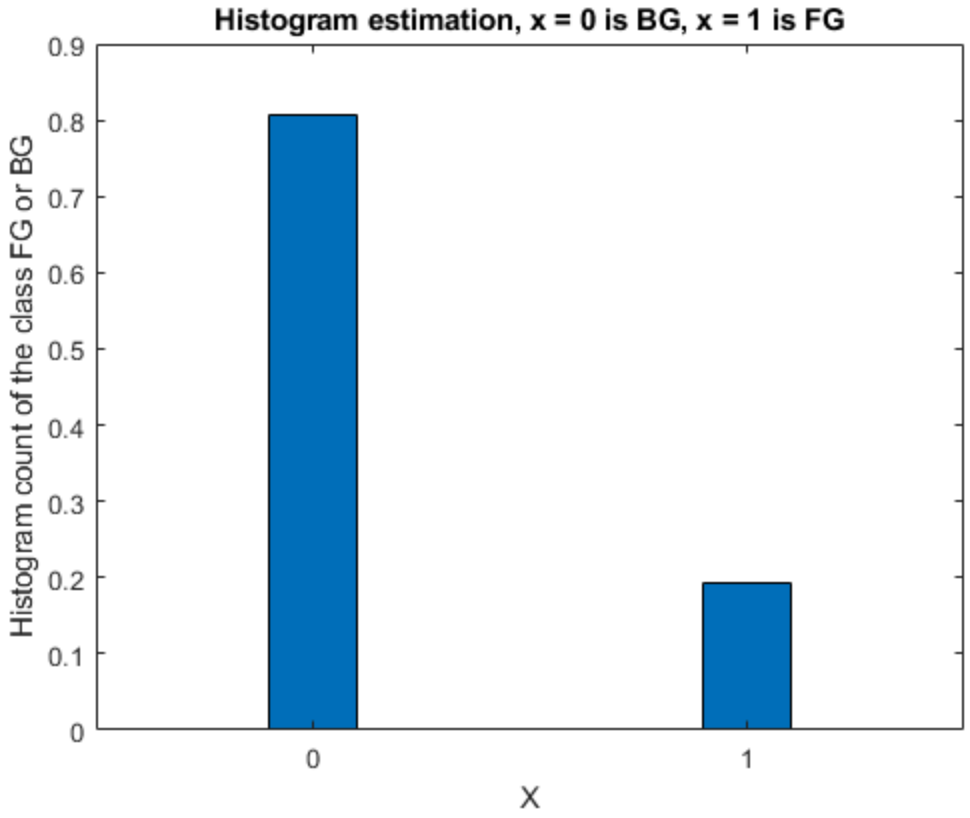
```
        title(sprintf('Feature: %d, red-FG, black-BG',idx))
        hold off
        count = count + 1;
    end
end
figure()


% For the best features calculating the 8x8 covariance matrix
%Covariance Matrices FG and BG for 8 best dimensions
%BestFeatures
CovMtxFG8 = zeros(8,8);
CovMtxBG8 = zeros(8,8);
sampleMeanFG8 = sampleMeanFG(BestFeatures);
sampleMeanBG8 = sampleMeanBG(BestFeatures);
FG8 = FG(:,BestFeatures);
BG8 = BG(:,BestFeatures);

for i = 1:FG_size
    CovMtxFG8 = CovMtxFG8 + ((FG8(i,:)' - sampleMeanFG8))*(((FG8(i,:)
 - sampleMeanFG8')));
end

for i = 1:BG_size
    CovMtxBG8 = CovMtxBG8 + ((BG8(i,:)' - sampleMeanBG8))*(((BG8(i,:)
 - sampleMeanBG8')));
end

CovMtxFG8 = CovMtxFG8 / FG_size;
CovMtxBG8 = CovMtxBG8 / BG_size;


%******* GETTING THE TEST IMAGE AND PADDING *******

[cImageOld colormap] = imread('cheetah.bmp');
RGB = ind2rgb(cImageOld,colormap);
cImage = RGB(:,:,1);
paddingType = 'symmetric';
cImage = padarray(cImage,[4 4],paddingType,'pre');
cImage = padarray(cImage,[3 3],paddingType,'post');
imshow(cImage); title('Original image with symmetric padding.');
figure();

%************************************************

%******* CLASSIFYING EACH PIXEL IN THE TEST IMAGE ********

cImageOldX = size(cImageOld,1); cImageOldY = size(cImageOld,2);
cImageX = size(cImage,1); cImageY = size(cImage,2);
A = [0  1  5  6  14  15  27  28 2  4  7  13  16  26  29  42 3  8  12
 17  25  30  41  43 9  11  18  24  31  40  44  53 10  19  23  32  39
 45  52  54 20  22  33  38  46  51  55  60 21  34  37  47  50  56  59
 61 35  36  48  49  57  58  62  63];
A = A + 1;
```

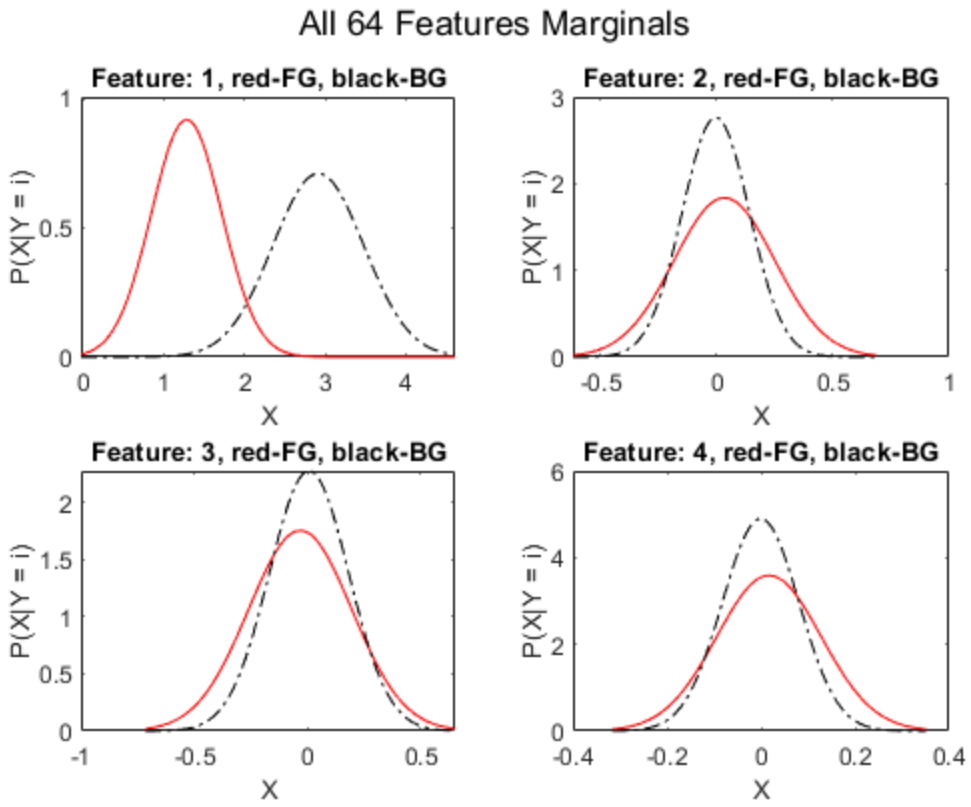
```
decisionImage64 = zeros(size(cImageOld,1),size(cImageOld,2));

%All 64 Features
point = zeros(64,1);
for i = 1:cImageOldX
    for j = 1:cImageOldY
        temp = (dct2(cImage(i:i+7, j:j+7)))';
        vectorDct= temp(:);
        point(A) = vectorDct;
        mhbDistFG = ((point - sampleMeanFG)')*(inv(CovMtxFG64))*(point
 - sampleMeanFG);
        alphaFG = log(((2*pi)^64)*det(CovMtxFG64)) - 2*log(CPrior);
        mhbDistBG = ((point - sampleMeanBG)')*(inv(CovMtxBG64))*(point
 - sampleMeanBG);
        alphaBG = log(((2*pi)^64)*det(CovMtxBG64)) - 2*log(NCPrior);
        [M,decision] = min([(mhbDistBG + alphaBG) (mhbDistFG +
 alphaFG)]);
        decision = decision - 1;
        decisionImage64(i,j) = decision;
    end
end

% 8 Best Features
decisionImage8 = zeros(size(cImageOld,1),size(cImageOld,2));
point = zeros(8,1);
for i = 1:cImageOldX
    for j = 1:cImageOldY
        temp = (dct2(cImage(i:i+7, j:j+7)))';
        vectorDct= temp(:);
        point(A) = vectorDct;
        point = point(BestFeatures);
        mhbDistFG = ((point - sampleMeanFG8)')*(inv(CovMtxFG8))*(point
 - sampleMeanFG8);
        alphaFG = log(((2*pi)^8)*det(CovMtxFG8)) - 2*log(CPrior);
        mhbDistBG = ((point - sampleMeanBG8)')*(inv(CovMtxBG8))*(point
 - sampleMeanBG8);
        alphaBG = log(((2*pi)^8)*det(CovMtxBG8)) - 2*log(NCPrior);
        [M,decision] = min([(mhbDistBG + alphaBG) (mhbDistFG +
 alphaFG)]);
        decision = decision - 1;
        decisionImage8(i,j) = decision;
    end
end

%**************************************************

%******* PRINTING THE FINAL BLACK & WHITE IMAGES   *******

I = mat2gray(decisionImage64,[0 1]);
imshow(I); title('64 Features prediction with W=Cheetah,
 B=NoCheetah');
figure()

I = mat2gray(decisionImage8,[0 1]);
```

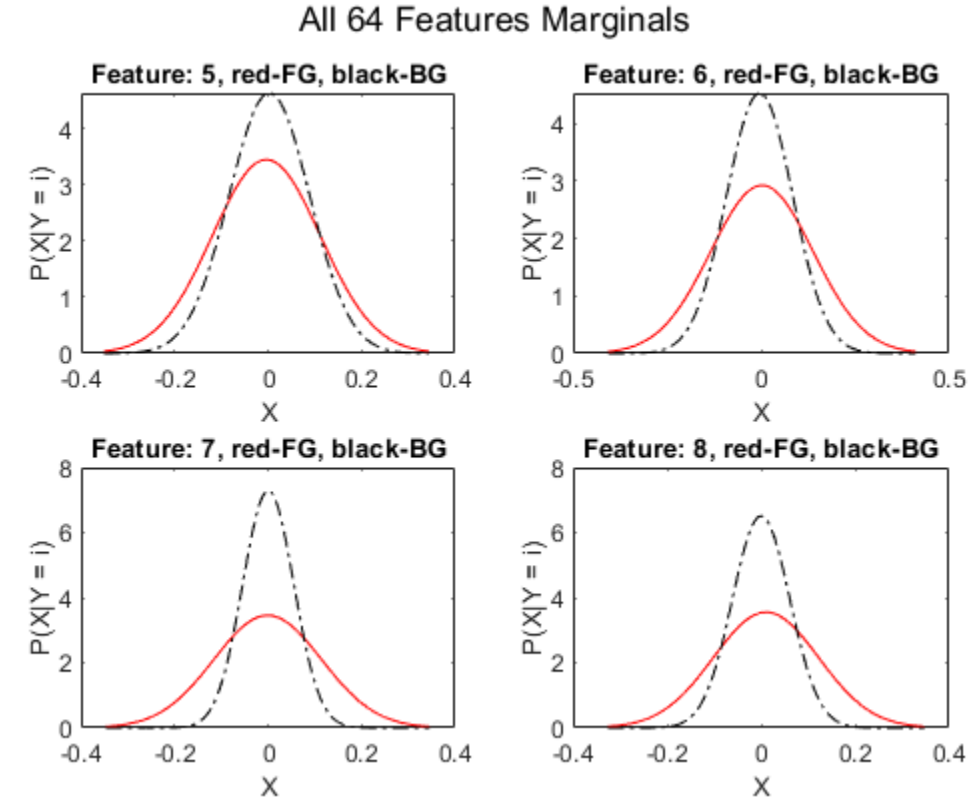
```
imshow(I); title('8 Best Features prediction with W=Cheetah,
 B=NoCheetah');

%*************************************************


%******* TOTAL ERROR CALCULATION FOR THE TWO CASES *********

[cImageReal colormap] = imread('cheetah_mask.bmp');
cImageReal = double(cImageReal)/255;
Image_Size = size(cImageReal,2)*size(cImageReal,1);
FG_Sum = sum(sum(cImageReal));
BG_Sum = Image_Size - FG_Sum;
truePriorCheetah = FG_Sum/Image_Size;
truePriorNoCheetah = 1 - truePriorCheetah;
fprintf('True Prior for Cheetah is: %d\n',truePriorCheetah);
fprintf('True Prior for No Cheetah is: %d\n\n',truePriorNoCheetah);

% 64 Gaussian Features Error
errorMask = decisionImage64 - cImageReal;
%FG misclassified as BG / total true FG
beta64 = sum(sum(errorMask == -1)) / FG_Sum; %False Negative --> beta
%BG misclassified as FG / total true BG
alpha64 = sum(sum(errorMask == 1)) / BG_Sum;  %False Positive -->
 alpha
ProbOfError64 = truePriorCheetah * beta64 + truePriorNoCheetah *
 alpha64;
fprintf('False Positive alpha for 64 Features is: %d\n',alpha64);
fprintf('False Negative beta for 64 Features is: %d\n',beta64);
fprintf('Total Probability of Error for 64 Features is: %d\n
\n',ProbOfError64);

% 8 Gaussian Features Error
errorMask = decisionImage8 - cImageReal;
%FG misclassified as BG / total true FG
beta8 = sum(sum(errorMask == -1)) / FG_Sum; %False Negative --> beta
%BG misclassified as FG / total true BG
alpha8 = sum(sum(errorMask == 1)) / BG_Sum;  %False Positive --> alpha
ProbOfError8 = truePriorCheetah * beta8 + truePriorNoCheetah * alpha8;
fprintf('False Positive alpha for best 8 features is: %d\n',alpha8);
fprintf('False Negative beta for best 8 features is: %d\n',beta8);
fprintf('Total Probability of Error for best 8 features is: %d
\n',ProbOfError8);
```

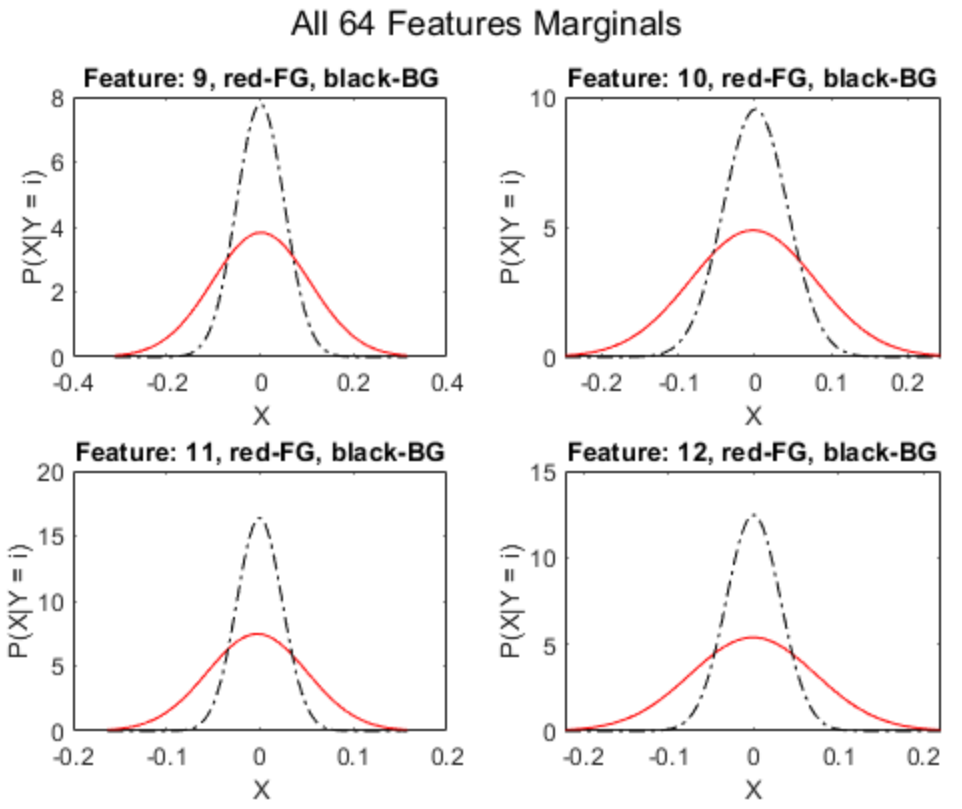
```
% ***** PRINT OUTPUT FROM THE CODE IS BELOW ****************
```

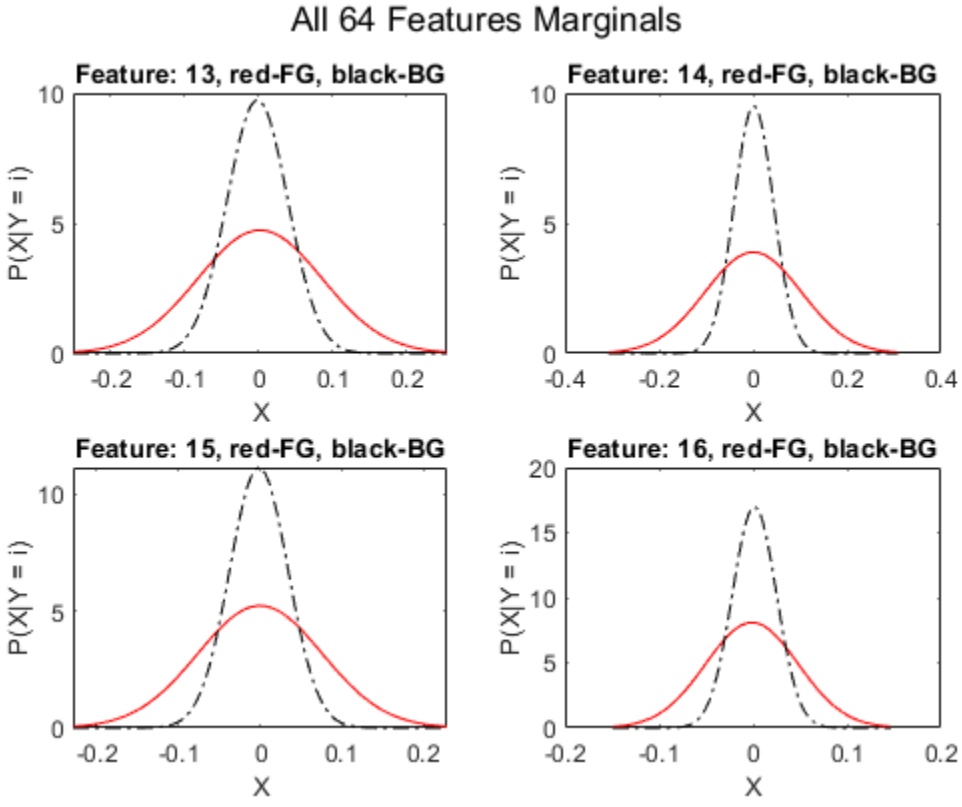
```
"Cheetah prior" estimated from the training data is: 1.918649e-01
"No Cheetah prior" estimated from the training data is: 8.081351e-01
True Prior for Cheetah is: 1.918519e-01
True Prior for No Cheetah is: 8.081481e-01

False Positive alpha for 64 Features is: 4.214518e-02
False Negative beta for 64 Features is: 1.713983e-01
Total Probability of Error for 64 Features is: 6.694263e-02

False Positive alpha for best 8 features is: 1.958987e-03
False Negative beta for best 8 features is: 4.954198e-01
Total Probability of Error for best 8 features is: 9.663036e-02
```

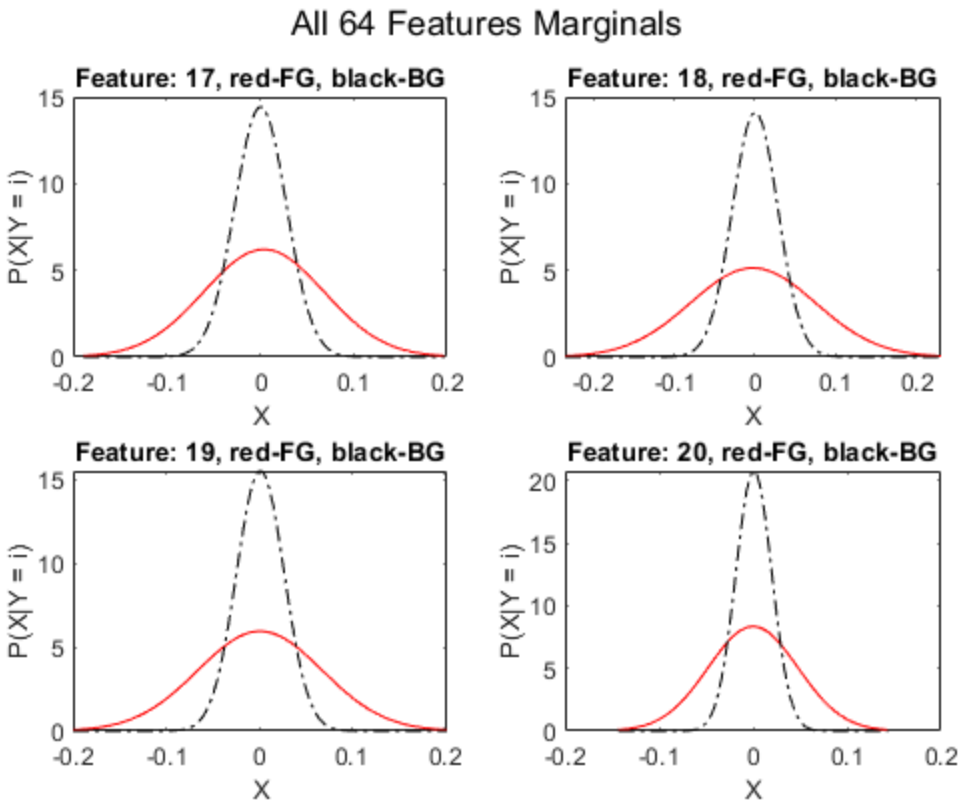


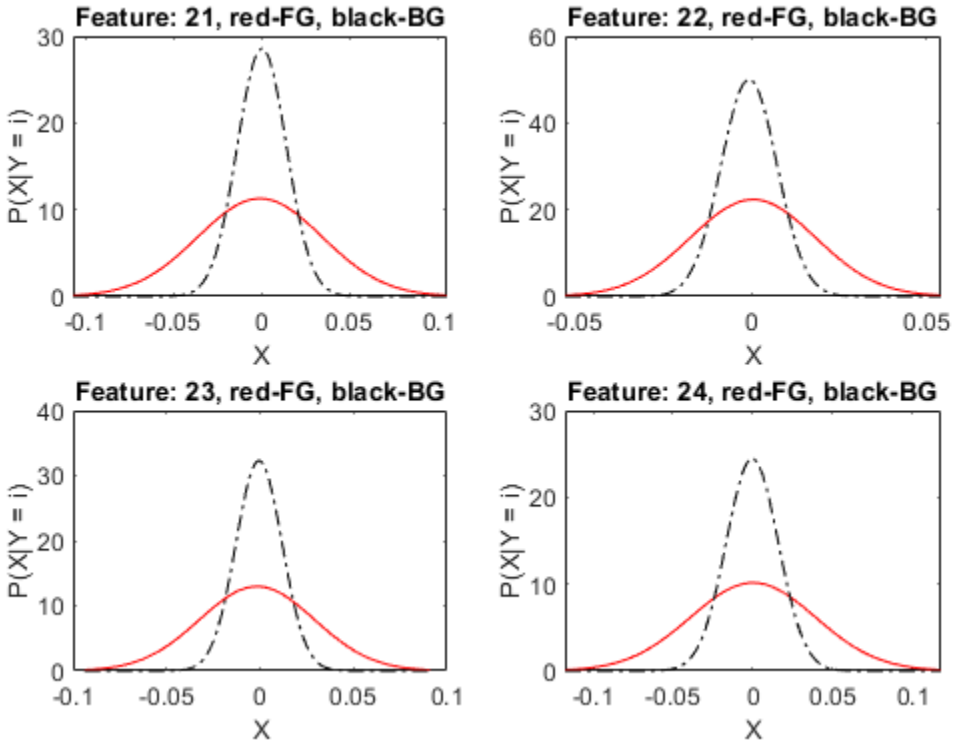

All 64 Features Marginals
Feature: 1, red-FG, black-BG
Feature: 2, red-FG, black-BG
Feature: 3, red-FG, black-BG
Feature: 4, red-FG, black-BG
P(X|Y = i)
X

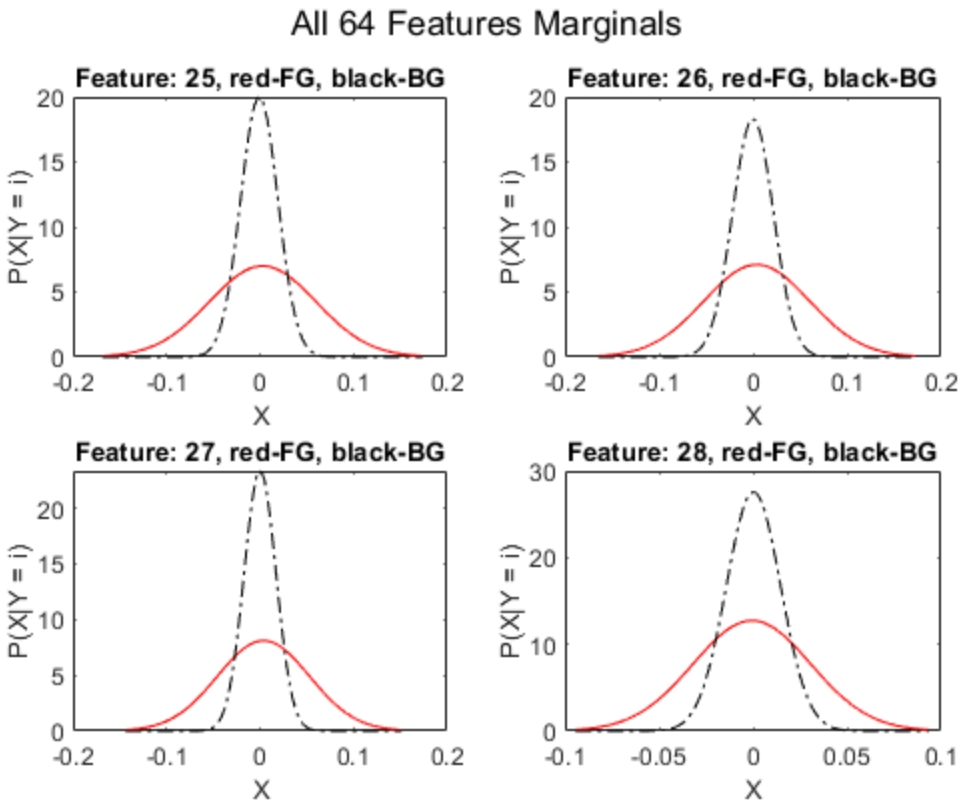

All 64 Features Marginals
Feature: 5, red-FG, black-BG
Feature: 6, red-FG, black-BG
Feature: 7, red-FG, black-BG
Feature: 8, red-FG, black-BG
P(X|Y = i)
X

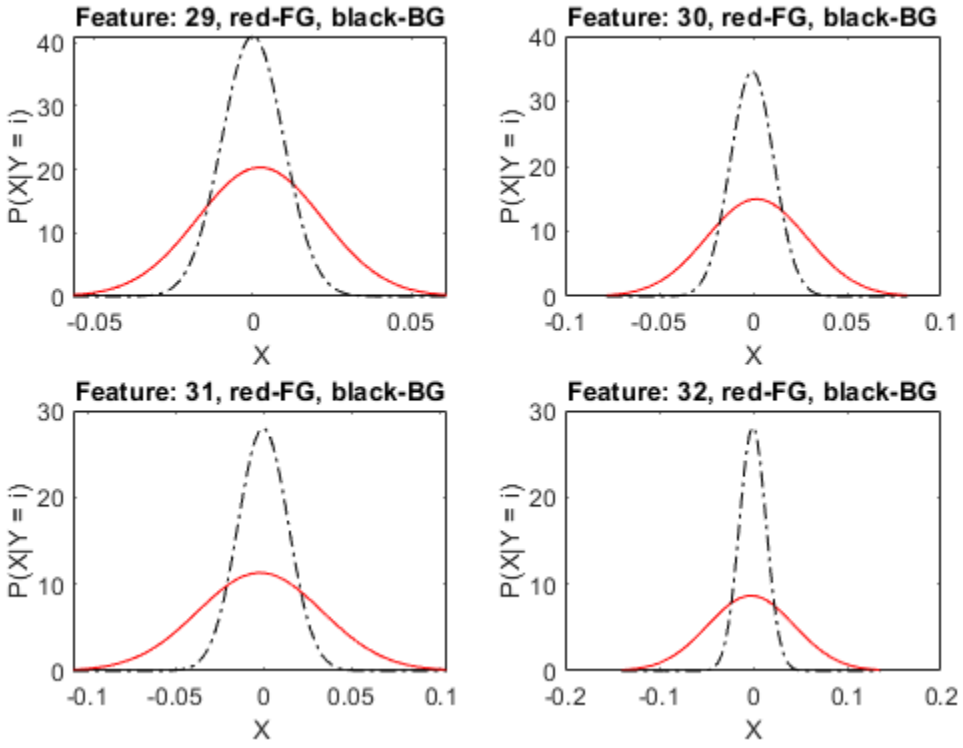

All 64 Features Marginals
Feature: 9, red-FG, black-BG
Feature: 10, red-FG, black-BG
Feature: 11, red-FG, black-BG
Feature: 12, red-FG, black-BG
P(X|Y = i)
X

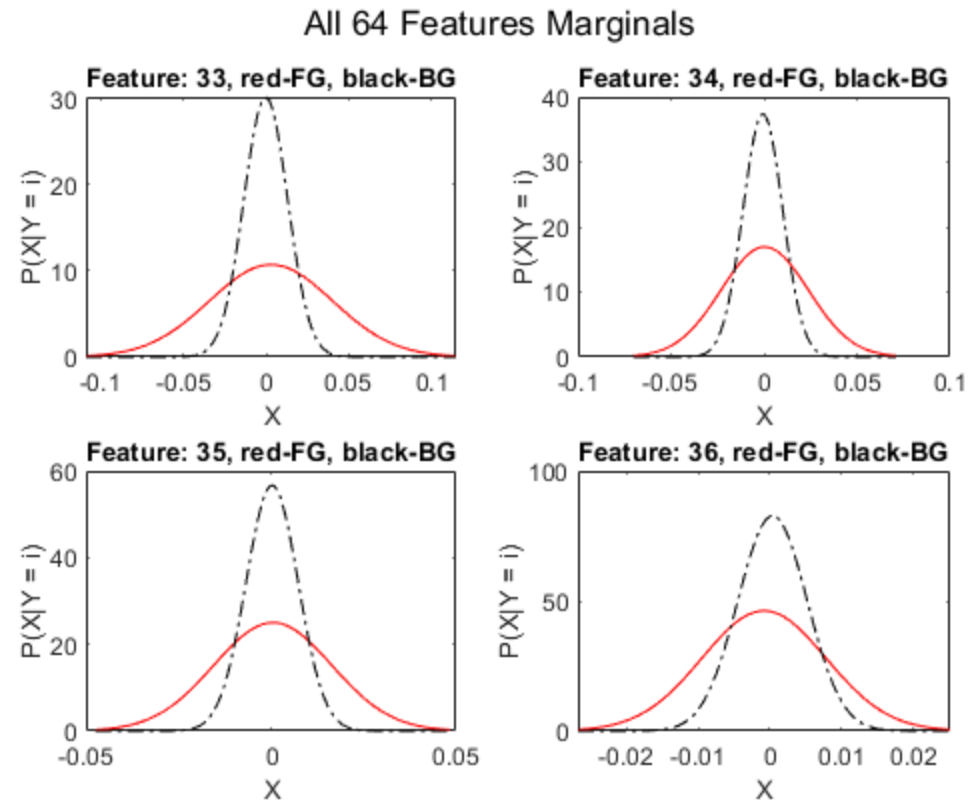

All 64 Features Marginals
Feature: 13, red-FG, black-BG
Feature: 14, red-FG, black-BG
Feature: 15, red-FG, black-BG
Feature: 16, red-FG, black-BG
P(X|Y = i)
X

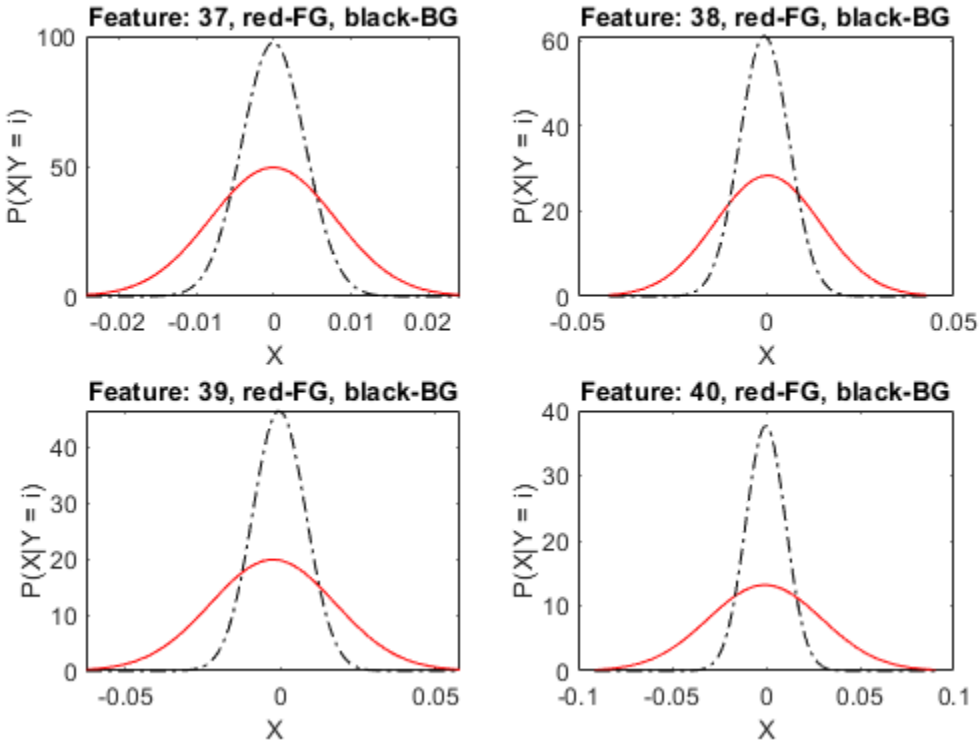

All 64 Features Marginals
Feature: 17, red-FG, black-BG
Feature: 18, red-FG, black-BG
Feature: 19, red-FG, black-BG
Feature: 20, red-FG, black-BG
P(X|Y = i)
X

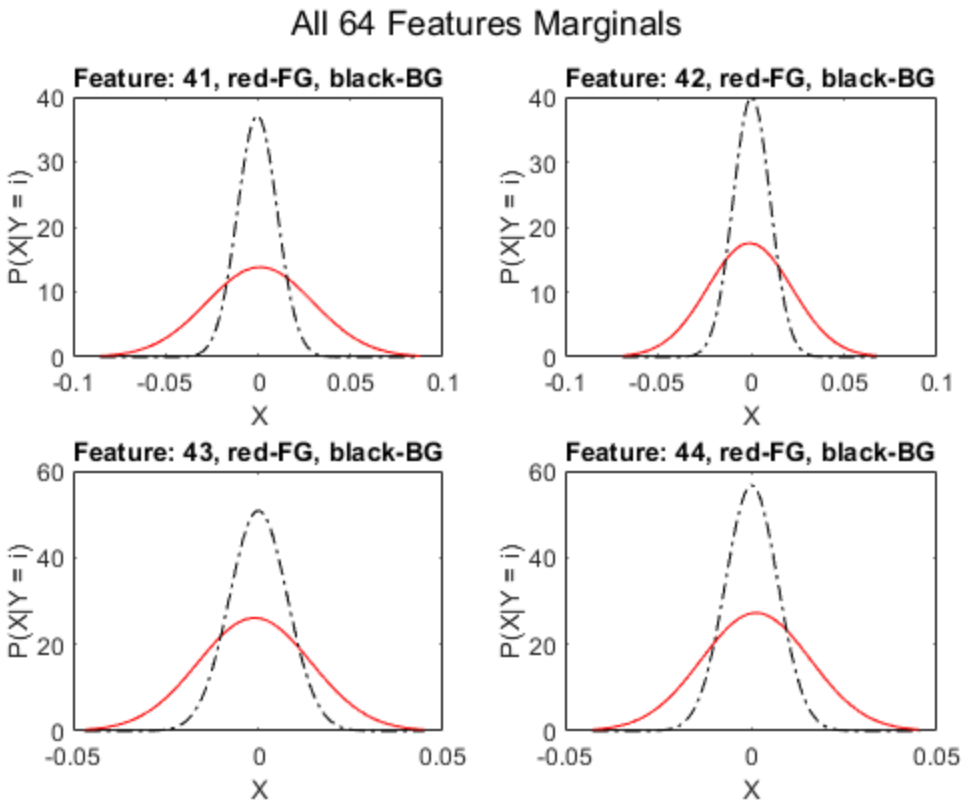

All 64 Features Marginals
Feature: 21, red-FG, black-BG
Feature: 22, red-FG, black-BG
Feature: 23, red-FG, black-BG
Feature: 24, red-FG, black-BG
P(X|Y = i)
X

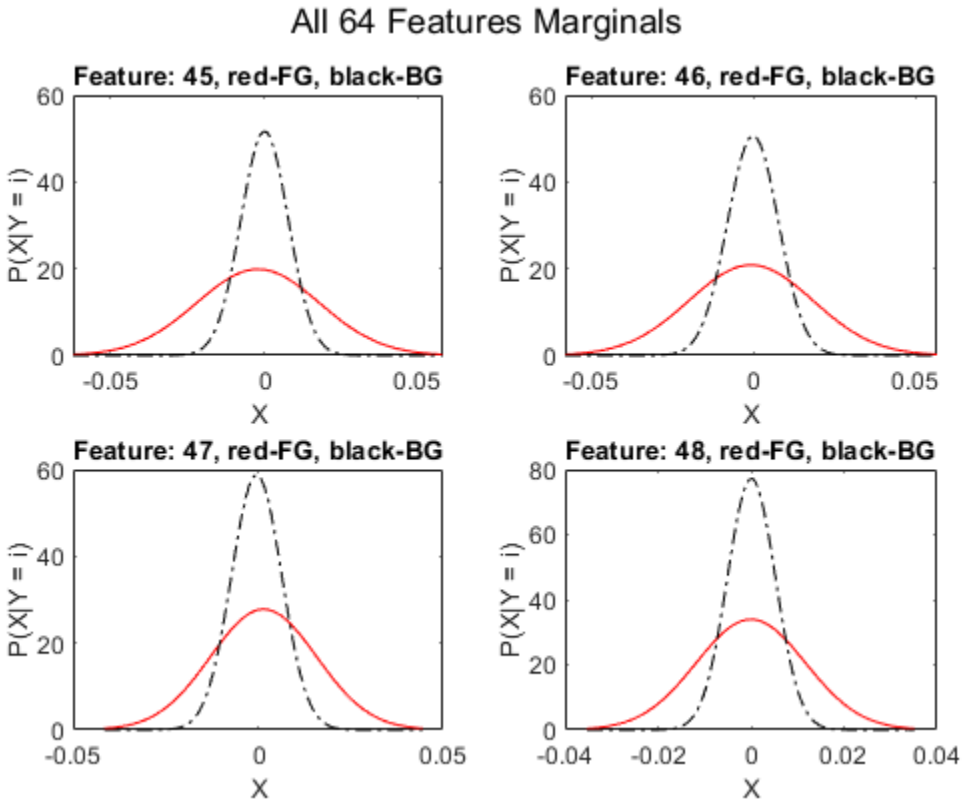

All 64 Features Marginals
Feature: 25, red-FG, black-BG
Feature: 26, red-FG, black-BG
Feature: 27, red-FG, black-BG
Feature: 28, red-FG, black-BG
P(X|Y = i)
X

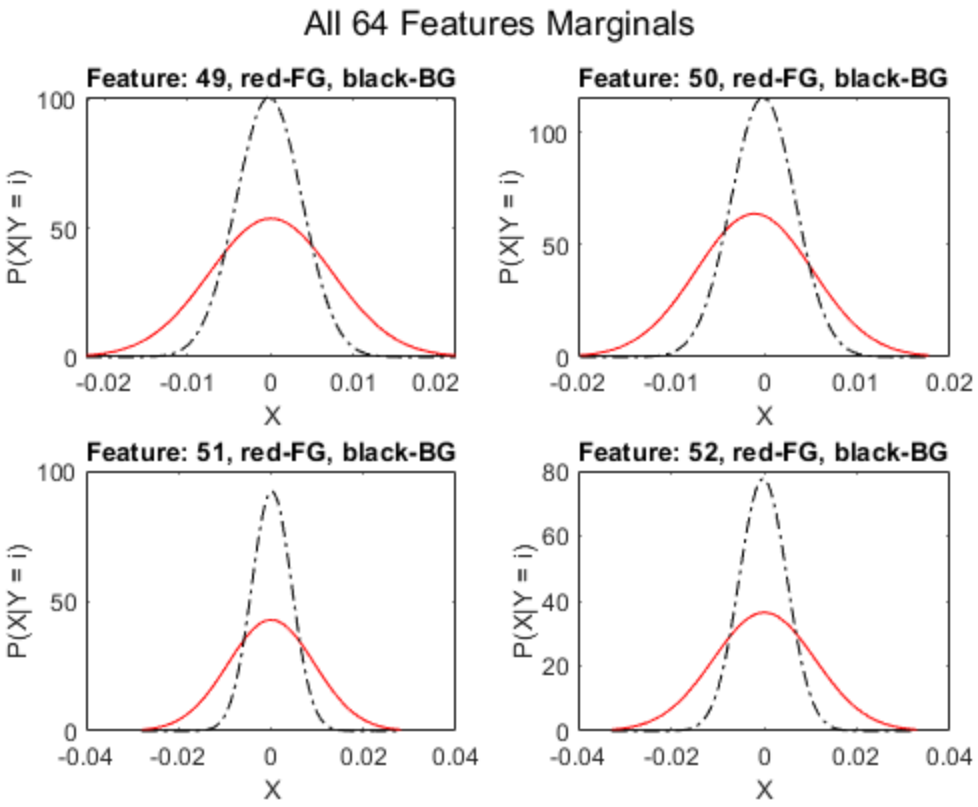

All 64 Features Marginals
Feature: 29, red-FG, black-BG
Feature: 30, red-FG, black-BG
Feature: 31, red-FG, black-BG
Feature: 32, red-FG, black-BG
P(X|Y = i)
X

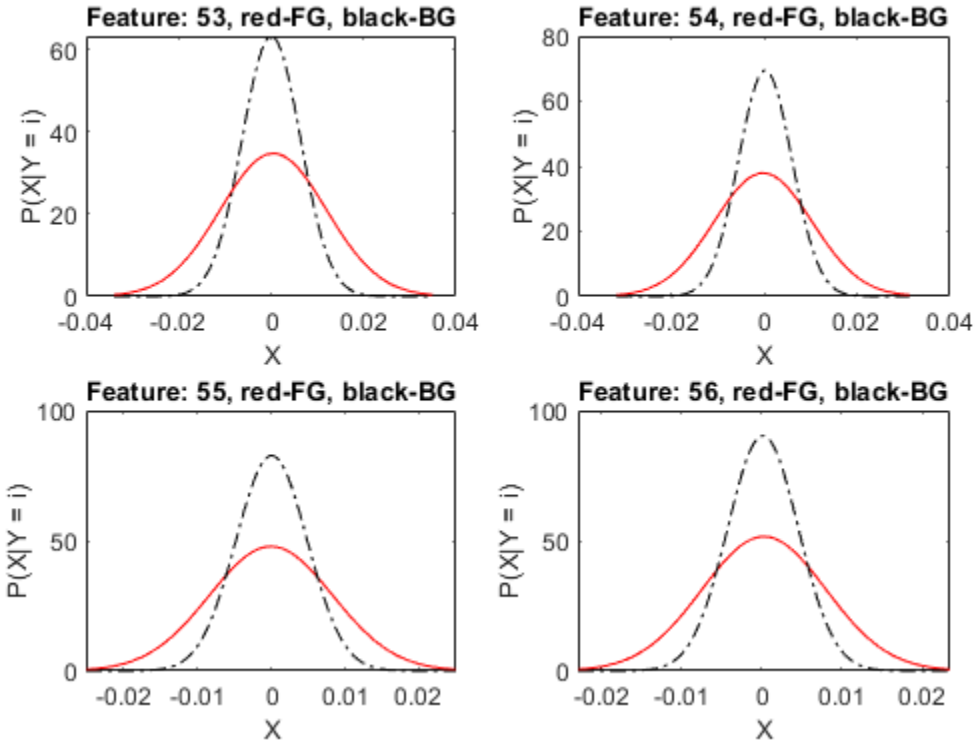

All 64 Features Marginals
Feature: 33, red-FG, black-BG
Feature: 34, red-FG, black-BG
Feature: 35, red-FG, black-BG
Feature: 36, red-FG, black-BG
P(X|Y = i)
X

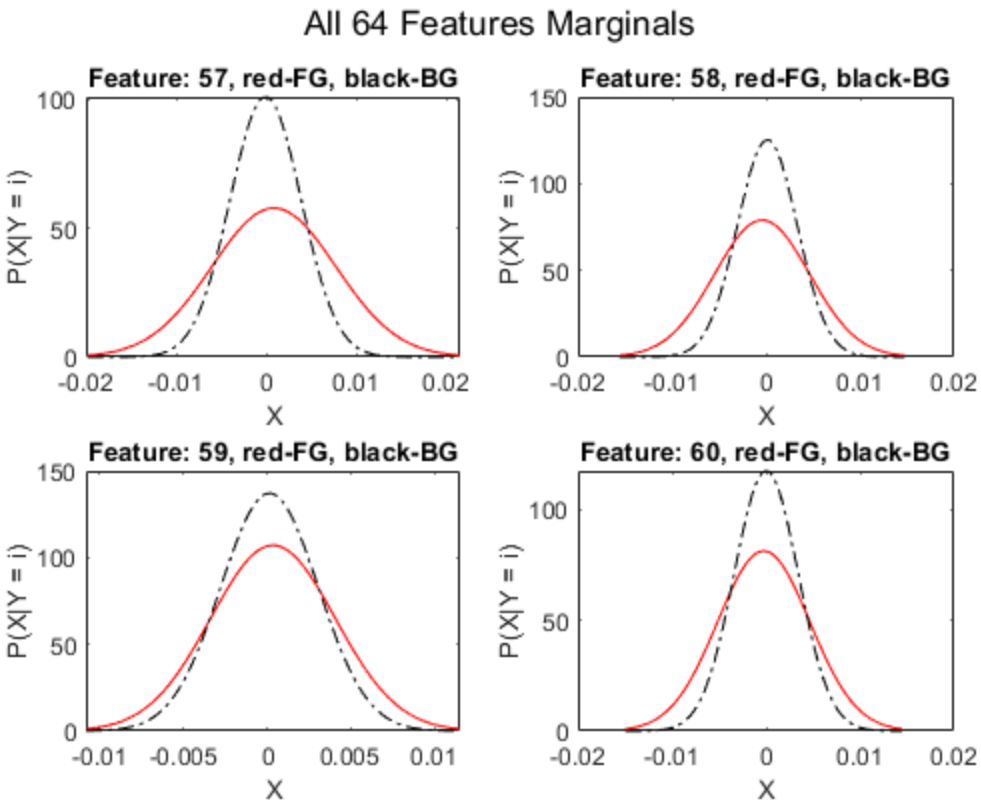

All 64 Features Marginals
Feature: 37, red-FG, black-BG
Feature: 38, red-FG, black-BG
Feature: 39, red-FG, black-BG
Feature: 40, red-FG, black-BG
P(X|Y = i)
X

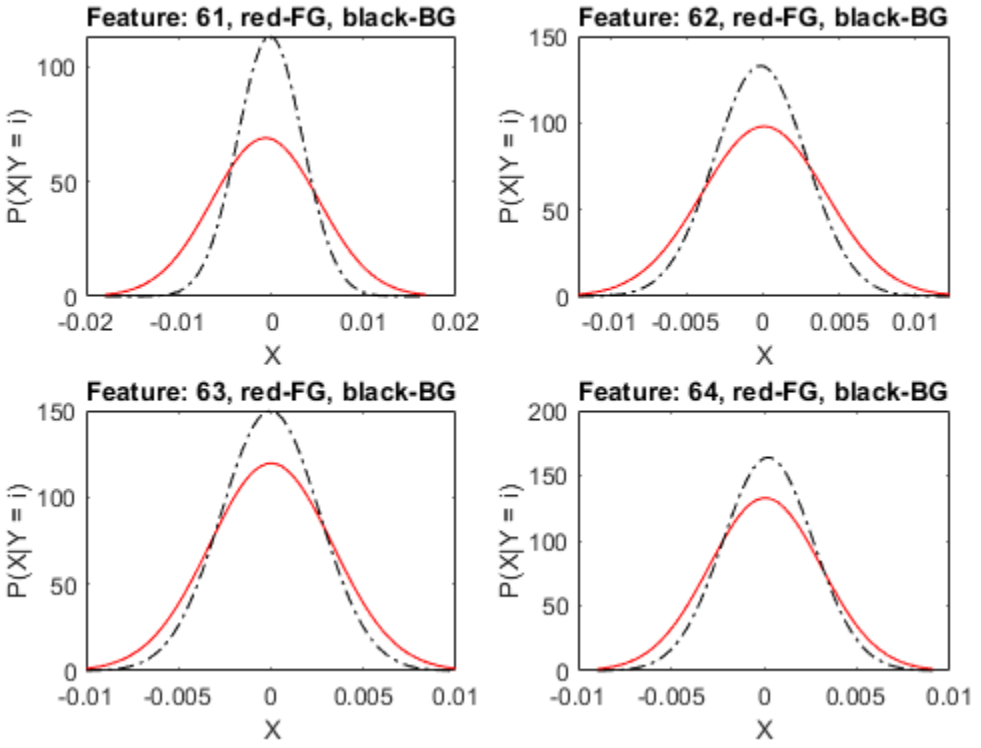

All 64 Features Marginals
Feature: 41, red-FG, black-BG
Feature: 42, red-FG, black-BG
Feature: 43, red-FG, black-BG
Feature: 44, red-FG, black-BG
P(X|Y = i)
X

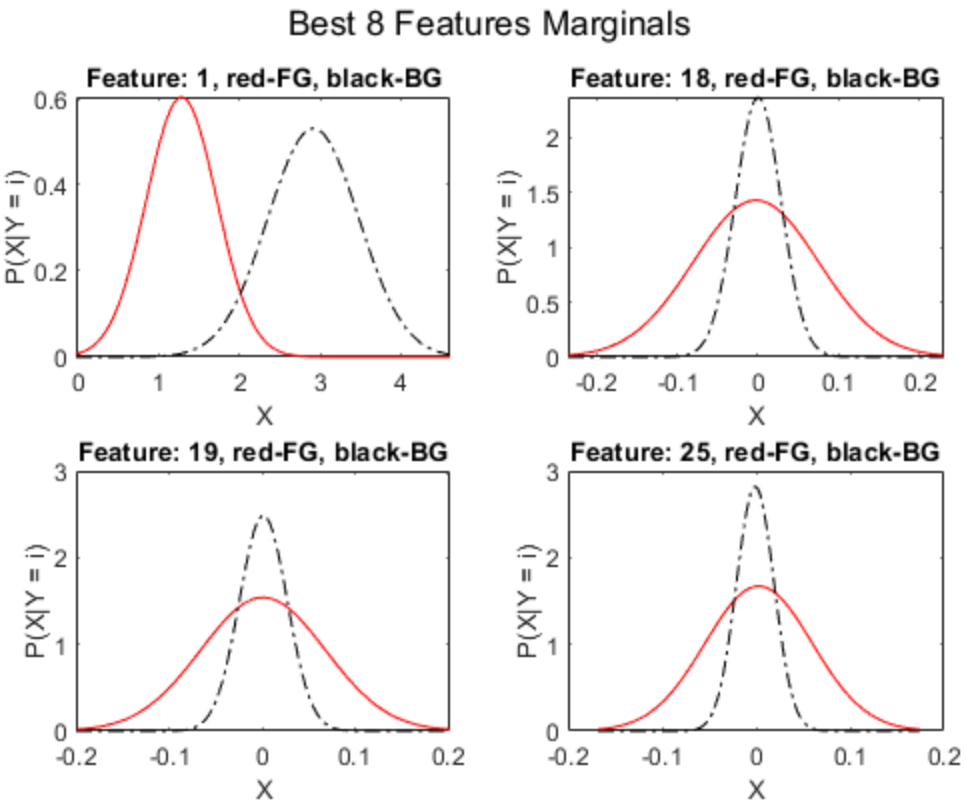

All 64 Features Marginals
Feature: 45, red-FG, black-BG
Feature: 46, red-FG, black-BG
Feature: 47, red-FG, black-BG
Feature: 48, red-FG, black-BG
P(X|Y = i)
X

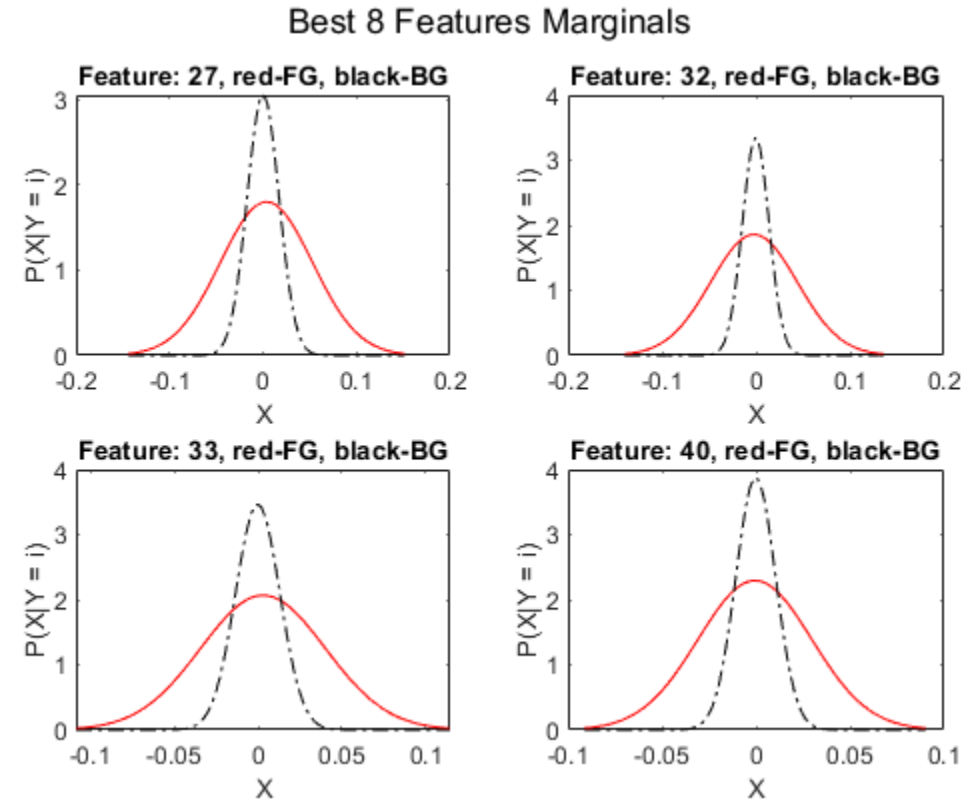

All 64 Features Marginals
Feature: 49, red-FG, black-BG
Feature: 50, red-FG, black-BG
Feature: 51, red-FG, black-BG
Feature: 52, red-FG, black-BG
P(X|Y = i)
X

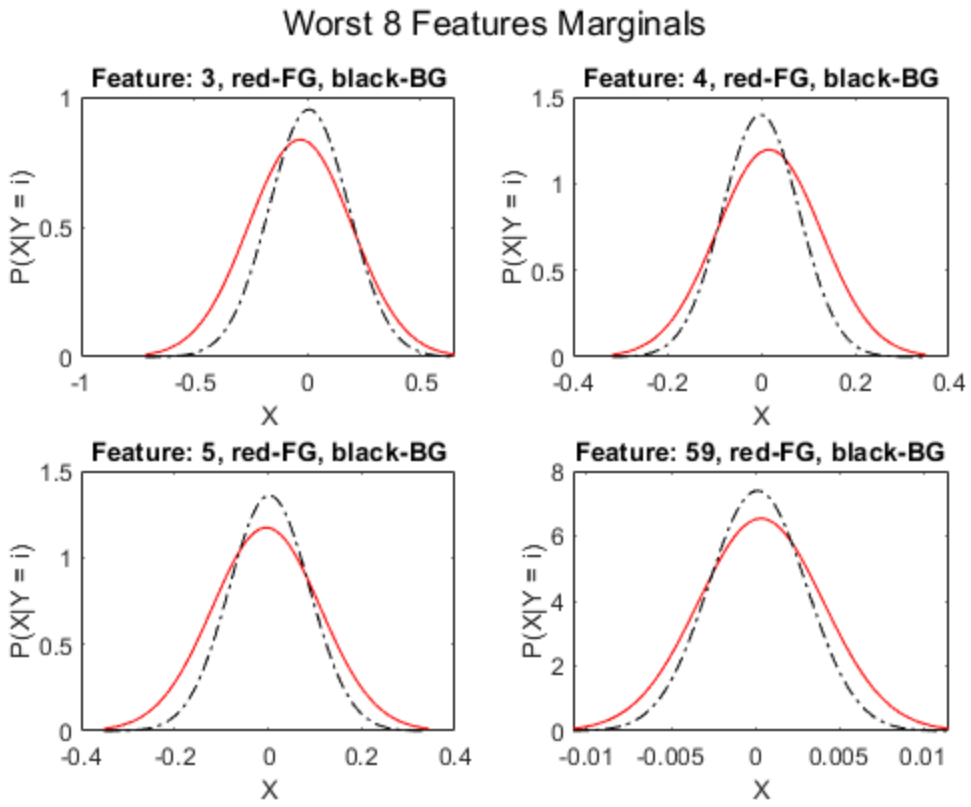

All 64 Features Marginals
Feature: 53, red-FG, black-BG
Feature: 54, red-FG, black-BG
Feature: 55, red-FG, black-BG
Feature: 56, red-FG, black-BG
P(X|Y = i)
X

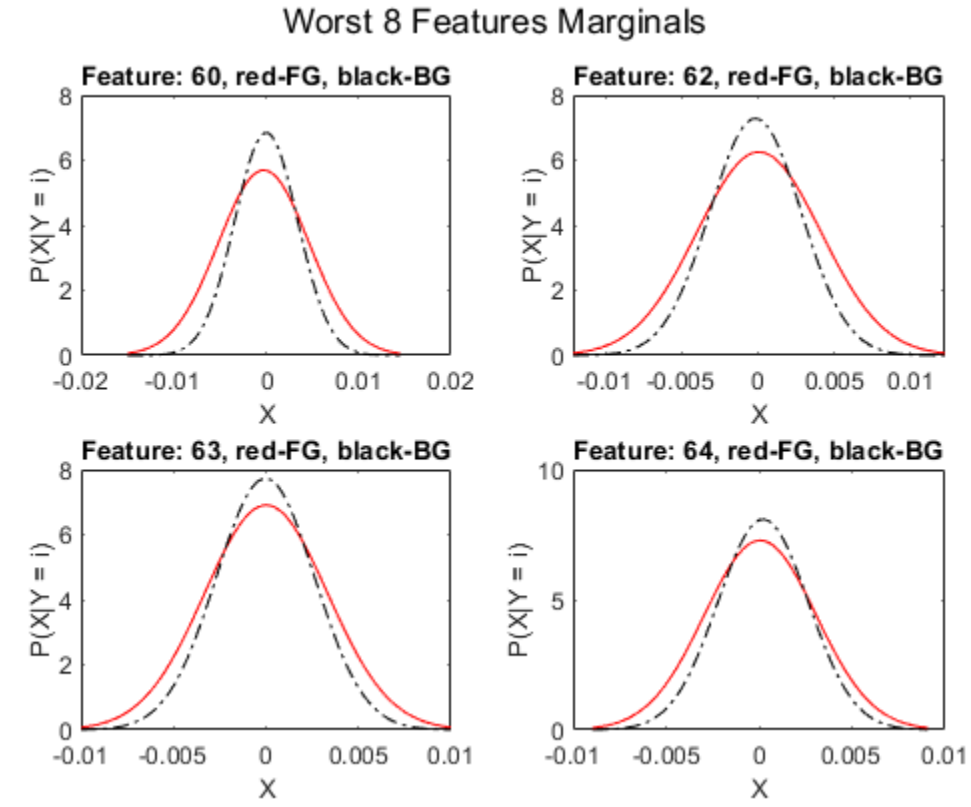

All 64 Features Marginals
Feature: 57, red-FG, black-BG
Feature: 58, red-FG, black-BG
Feature: 59, red-FG, black-BG
Feature: 60, red-FG, black-BG
P(X|Y = i)
X


All 64 Features Marginals
Feature: 61, red-FG, black-BG
Feature: 62, red-FG, black-BG
Feature: 63, red-FG, black-BG
Feature: 64, red-FG, black-BG
P(X|Y = i)
X


Best 8 Features Marginals
Feature: 1, red-FG, black-BG
Feature: 18, red-FG, black-BG
Feature: 19, red-FG, black-BG
Feature: 25, red-FG, black-BG
P(X|Y = i)
X

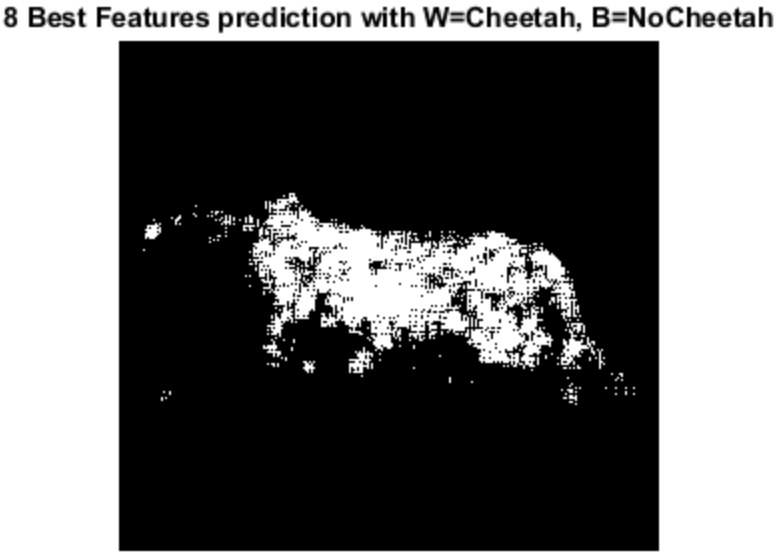

Best 8 Features Marginals
Feature: 27, red-FG, black-BG
Feature: 32, red-FG, black-BG
Feature: 33, red-FG, black-BG
Feature: 40, red-FG, black-BG
P(X|Y = i)
X


Worst 8 Features Marginals
Feature: 3, red-FG, black-BG
Feature: 4, red-FG, black-BG
Feature: 5, red-FG, black-BG
Feature: 59, red-FG, black-BG
P(X|Y = i)
X


Worst 8 Features Marginals
Feature: 60, red-FG, black-BG
Feature: 62, red-FG, black-BG
Feature: 63, red-FG, black-BG
Feature: 64, red-FG, black-BG
P(X|Y = i)
X

Original image with symmetric padding.

64 Features prediction with W=Cheetah, B=NoCheetah



Published with MATLAB® R2018b