Assignment 3:

Sudarshan Devkota

Problem 1:

1. Classification With Harris Features:

For the given image ‘input_hcd1’, we first implement harris corner detection to get corners, edges and flat regions in the image. The steps applied for corner detection with Harris Corner Detection technique are as follows:

I. Apply smoothing to the images with a gaussian filter of certain size

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution.

II. Compute the derivative of the given image $I(x, y)$ in x-direction and y-direction as Ix and Iy. Again, we will use Sobel operators here.

$$Ix = hx * I, \ \ and \ \ Iy = hy * I$$

III. Compute the pixel by pixel products of the gradient images.

$$Ixx = Ix \,.\, Ix$$

$$Iyy = Iy \,.\, Iy$$

$$Ixy = Ix \,.\, Iy$$

IV. Convolve the images obtained in the previous step with the window function w(x,y) to get the following three images:

$$Sxx = w * Ixx$$

$$Syy = w * Iyy$$

$$Sxy = w * Ixy$$

Here, the window function *w* is a 3x3, 5x5 or 7x7 filter of all 1s.

V. For every pixel (x,y), compute the matrix

$$\boldsymbol{H}_{xy} = \begin{bmatrix} S_{xx}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{yy}(x,y) \end{bmatrix}$$

VI. Compute the response of the detector at each pixel

$$R = det(M) - k(trace(M))^2$$

$$where,\ det(M) = \lambda_1 . \lambda_2$$

$$trace = \lambda_1 + \lambda_2$$

λ_1 *and* λ_2 *are the eigen values of* M, *and k is the sensitivity factor to separate corners from edges*

VII. Threshold *R(x,y)* with a threshold value of *‘th’* to get potential locations of corners, edges and flat regions.

$Corners: \ R(x,y) < th$
$Edges: \ R(x,y) < -th$
$Flats: \ -th < R(x,y) < th$

VIII. For each point of interest, we define a 2 x 1 vector using the eigenvalues: λ_1 and λ_2

IX. Now we model $P(x|C)$ as Gaussian distribution function and estimate the parameters that best fit i) edges, ii) corners, and iii) flat regions. The multivariate Gaussian distribution has the form:

$$p(\boldsymbol{x}|C_i) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} exp\left[-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right]$$

where μ_i is class mean and Σ_i is class covariance

$$\boldsymbol{\mu}_i = \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{x}_k^i$$

$$\boldsymbol{\Sigma}_i = \frac{1}{N}\sum_{k=1}^{N}(\boldsymbol{x}_k^i - \boldsymbol{\mu}_i)(\boldsymbol{x}_k^i - \boldsymbol{\mu}_i)^T$$

X. To obtain a prediction for class, we can either perform the Naive Bayes classification where we assume conditional independence between the features and make a decision based on the bayes rule:

$$P(C_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|C_i)P(C_i)}{p(\boldsymbol{x})}$$

Decision Strategy: Choose class *i,* if $p(\boldsymbol{x}|Ci) > p(\boldsymbol{x}|Cj)$ for all $j \neq i$

Or,

We can perform the Mahalanobis distance classification as:
To choose class *i*, we want $p(\boldsymbol{x}|Ci) > p(\boldsymbol{x}|Cj)$ for all $j \neq i$. So, we choose the class for which
Mahalanobis distance is the greatest. For the distance formulation, we have,

$$g_i(\boldsymbol{x}) = -\frac{1}{2}log(|\boldsymbol{\Sigma}_i|) - \frac{1}{2}\left[(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right]$$

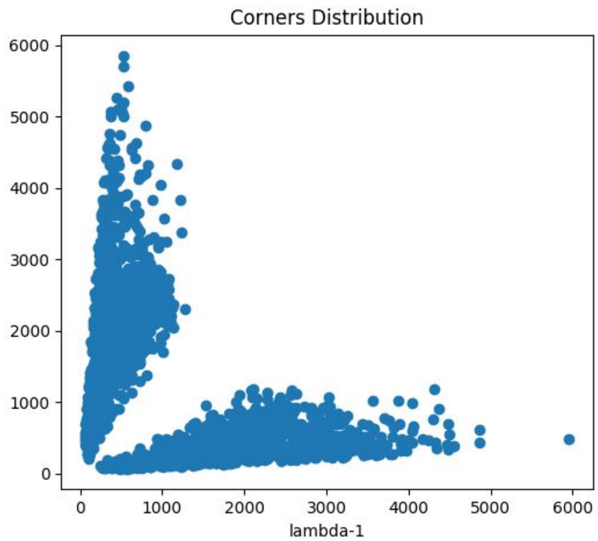
XI. Classify each point using the estimated distributions and find the error rates for each category.

2. Results:

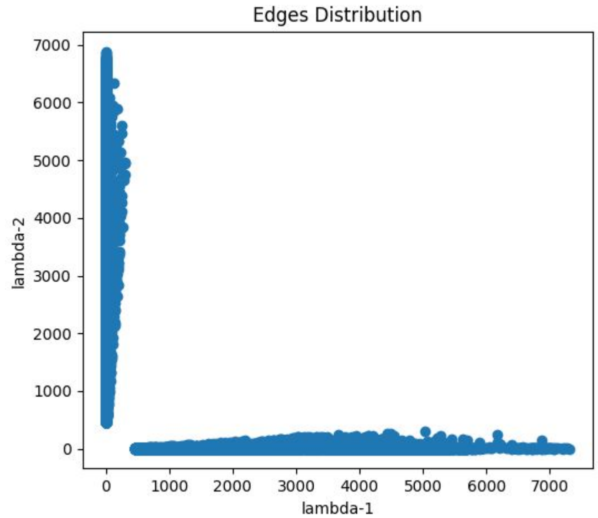
We obtain the following results for the given image. Table 1 shows the accuracy and error rates for each category, which are corners, edges, and flats. The overall accuracy is 94.7%

Table 1: Accuracy and error rates for each category:

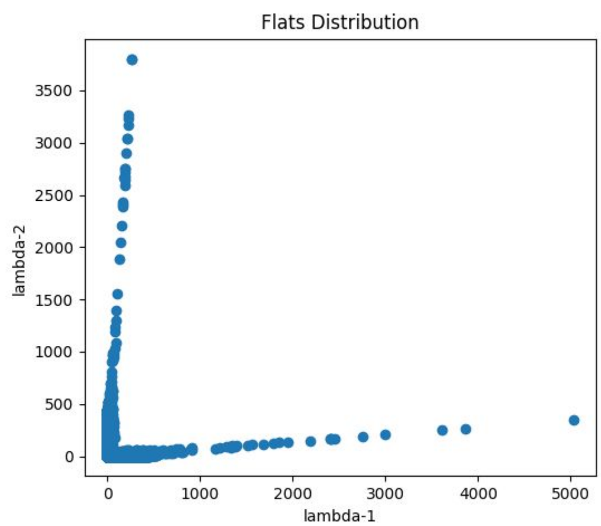
	Corners	Edges	Flats	Overall
Accuracy	0.0723	0.9801	0.99	0.947
Error rates	0.92	0.0198	0.01	0.053



a) Corners



b) edges



c) Flats

Fig 1: The figure above shows the distribution of a) Corners, b) Edges and c)Flats in the given image

The distribution of corners, edges, and flats are shown in the figure above. We can observe that corners are distributed in the region where both λ_1 and λ_2 are large. Edges are distributed in the region where either $\lambda_1 >> \lambda_2$ or $\lambda_1 << \lambda_2$. Similarly, flats are distributed where both λ_1 and λ_2 are small.

Problem 2

Bayes Linear Discriminant Classifier:

For the given digits dataset, we perform the following steps to implement the Bayes Linear Discriminant Classifier:

I. Consider ten classes for the dataset: 0, 1, .. 10. For each class we select first 100 images for training and testing the final classifier. We will use the first 50 images for training and the remaining 50 images for testing purposes.

II. Design a feature vector using the HCD algorithm. We use the HCD algorithm as mentioned in section 1 of problem 1 to compute the R scores for each training images. For each image, we choose the 10 largest positive scores, and 10 largest negative scores to form a 20 x 1 feature vector

III. Then we compute the 20 x 20 covariance matrix of the entire data set, and the 20 x 1 mean vectors for each class

For every class *i,* we have the feature vector: $x_k = [x_{k,1} \;\; x_{k,2} \;\; \dots \;\; x_{k,20}]$, where *k* is the number of training examples of that class. Then we can define the class mean vector as:

$$\boldsymbol{\mu}_i = \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{x}_k^i$$

And the overall covariance can be obtained as:

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{k=1}^{N}\left(\boldsymbol{x}_k^i - \boldsymbol{\mu}\right)\left(\boldsymbol{x}_k^i - \boldsymbol{\mu}\right)^T$$

Here, μ is the overall mean vector for all the 10 classes. The individual class mean vectors are different and show that the center of the three classes are distinct.

IV. Now, the feature vectors extracted from the images are processed by a bank of linear discriminant functions. The discriminant function for the i-th class is given by:

$$g_i(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i - \frac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i = \boldsymbol{x}^T\boldsymbol{h}_i + b_i$$

Where $\boldsymbol{h}_i = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i$ and $b_i = -\frac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i$

V. Therefore, for each class, we compute $g_i(x) = x^T h_i + b_i$ for $i = 1..k$, and assign *x* the label for the class for which $g_i(x)$ is the largest.

VI. For both the training and test data, we evaluate the probability of correct recognition and the probability of error for each case.

2. Results:

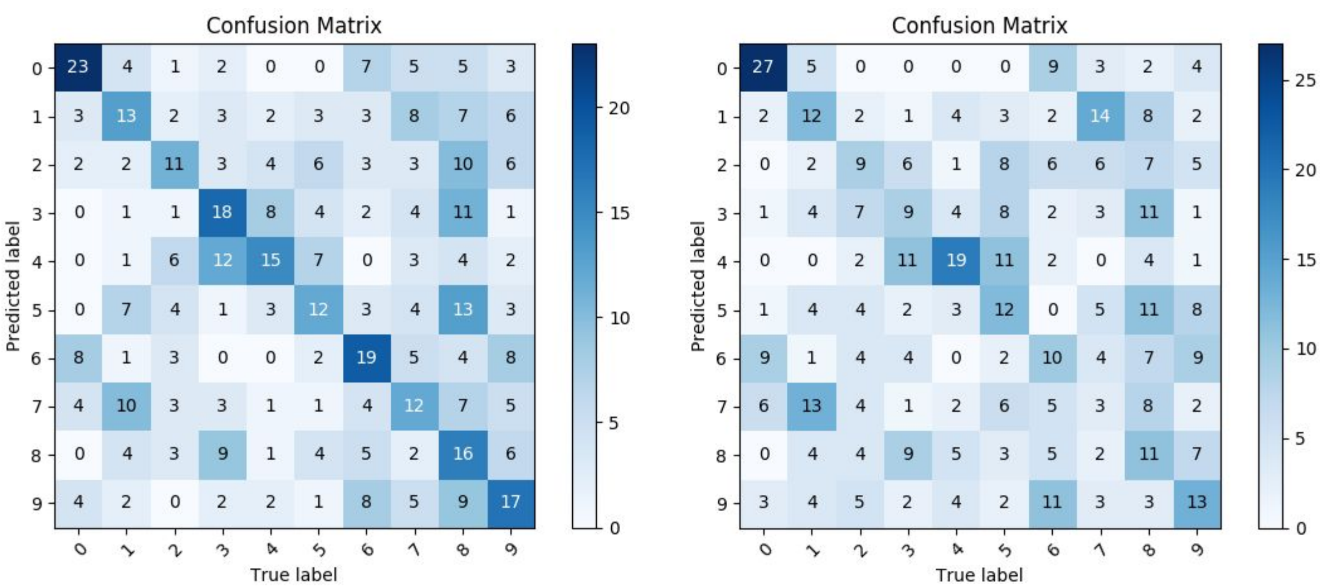
We obtain the following results for the classification of digits for the given dataset. Table 2 and 3 show the final statistics of the classifier in terms of probability of correct classification and probability of error for each class in training set and test set.

Table 2: Probability of correct classification and error for each class in training set:

Class labels	0	1	2	3	4	5	6	7	8	9
Prob. of correct classification (%)	55.27	28.88	32.35	33.96	41.66	30.0	35.18	23.52	18.60	29.82
Prob. of error (%)	47.72	71.11	67.64	66.03	58.33	70.0	64.81	76.47	81.39	70.17
Overall probability of correct recognition: 31.2 % Overall probability of error: 68.8 %										

Table 3: Probability of correct classification and error for each class in test set

Class Labels	0	1	2	3	4	5	6	7	8	9
Prob. of correct classification (%)	55.10	24.48	21.95	20.0	45.23	21.81	19.23	6.97	15.27	25.0
Prob. of error (%)	44.89	75.51	78.04	80.0	54.76	78.18	80.76	93.02	84.72	75.0
Overall probability of correct recognition: 25 % Overall probability of error: 75 %										



a) Training set b) Test set

Fig 2: Confusion matrix for a) training set and b) test set.

Instructions on how to run program

First, the following dependencies need to be installed to be able to run the program:

```
$ sudo apt-get install python3-pip
$ pip3 install numpy
$ pip3 install matplotlib
```

Unzip the 4808133.zip file and from inside the code/ directory run the following commands

```
$ python3 classification_with_hcd.py
$ python3 bayes_ldf.py
```

References:

[1] Class Notes - Abhijeet Mahalanobis