**Batch T4**

**Practical No. 1**

**Title of Assignment :**

**An univariate classifier from first principles**

**Student Name: Parshwa Herwade**

**Student PRN: 22510064**

1. An univariate classifier from first principles

a. Generate distributions (gaussian to start with) for male and female heights (1000 samples each). Fix the mean of female heights to 152 cm and male mean height to 166 cm. label the appropriate gender for samples in each of the distribution (M or F)

b. Fix the sd of both the distributions to 5

c. Try classification of gender using following approaches with aim to minimise

misclassification

i. Assign gender based on likelihood calculated from distributions (empirically estimated mean and sd and calculate probability assuming gaussian distributions)

ii. Derive a threshold hight to separate male female

iii. quantize the data at scale of 0.5 cm and empirically estimate the likelihood of male female in each segment based on majority

iv. In each of the above cases output a confusion matrix for classification

d. Try following values of sd (eg 2.5, 7.5 and 10) repeat 3.a, 3.b, 3.c, 3,d observe impact of change in sd on classification accuracy

e. Change the quantization interval length (say 0.001, 0.05, 0.1, 0.3, 1, 2, 5,10 cm etc) repeat 3.a, 3.b, 3.c, 3,d observe impact of change in sd on classification accuracy

**CODE**:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from scipy.stats import norm

from sklearn.metrics import confusion\_matrix, accuracy\_score

**( Generating height data for males and females )**

def gen\_data(mean, sd, size, label):

    heights = np.random.normal(mean, sd, size)

    return pd.DataFrame({'height': heights, 'label': [label] \* size})

# Plotting the height dist. for both males and females

def plot\_hist(female, male, bins=50, label=""):

    plt.hist([female, male], bins=bins, label=['Female', 'Male'], alpha=0.7, color=['purple', 'green'])

    plt.title(f'Height Distribution {label}')

    plt.xlabel('Height (cm)')

    plt.ylabel('Frequency')

    plt.legend(loc='upper right')

    plt.show()

**(Threshold-based classification)**

def threshold\_class(female, male, threshold):

    data = np.concatenate([female, male])

    preds = np.where(data < threshold, 'F', 'M')

    true\_labels = np.concatenate([['F'] \* len(female), ['M'] \* len(male)])

    return true\_labels, preds

**(Prob-based classification)**

def prob\_class(female, male, f\_mean, f\_sd, m\_mean, m\_sd):

    def classify(h):

        f\_prob = norm.pdf(h, f\_mean, f\_sd)

        m\_prob = norm.pdf(h, m\_mean, m\_sd)

        return 'F' if f\_prob > m\_prob else 'M'

    data = np.concatenate([female, male])

    preds = np.array([classify(h) for h in data])

    true\_labels = np.concatenate([['F'] \* len(female), ['M'] \* len(male)])

    return true\_labels, preds

(**Quantized classification)**

def quant\_class(female, male, interval):

    def quantize(data):

        return pd.Series(np.floor(data / interval)).value\_counts()

    f\_quant = quantize(female)

    m\_quant = quantize(male)

    intervals = set(f\_quant.index).union(set(m\_quant.index))

    preds = []

    true\_labels = []

    for interval in intervals:

        f\_count = f\_quant.get(interval, 0)

        m\_count = m\_quant.get(interval, 0)

        label = 'F' if f\_count >= m\_count else 'M'

        preds.extend([label] \* (f\_count + m\_count))

        true\_labels.extend(['F'] \* f\_count + ['M'] \* m\_count)

    return np.array(true\_labels), np.array(preds)

**(Evaluation if the classifier performance)**

def evaluate(true\_labels, preds, label=""):

    cm = confusion\_matrix(true\_labels, preds, labels=['F', 'M'])

    accuracy = accuracy\_score(true\_labels, preds)

    print(f"\n{label}")

    print("Confusion Matrix:")

    print(f"\n\t\tPredicted F\tPredicted M\nActual F\t{cm[0, 0]:<5} (TP)\t{cm[0, 1]:<5} (FN)\nActual M\t{cm[1, 0]:<5} (FP)\t{cm[1, 1]:<5} (TN)")

    print(f"\nAccuracy: {accuracy \* 100:.2f}%")

    return cm, accuracy

**(Main function to run other values for experimentation)**

def run():

    f\_mean = 152

    m\_mean = 166

    std\_devs = [2.5, 5, 7.5, 10]

    intervals = [0.001, 0.05, 0.1, 0.3, 1, 2, 5, 10]

    size = 1000

    results = []

    for sd in std\_devs:

        print(f"\n--- SD = {sd} ---")

**(Generation of synthetic data)**

        female = gen\_data(f\_mean, sd, size, 'F')['height'].values

        male = gen\_data(m\_mean, sd, size, 'M')['height'].values

**(Plotting histograms)**

        plot\_hist(female, male, label=f'(SD={sd})')

**(Threshold classification)**

        threshold = (f\_mean + m\_mean) / 2

        true\_labels, preds = threshold\_class(female, male, threshold)

        cm, acc = evaluate(true\_labels, preds, label=f"Threshold (Threshold={threshold})")

        results.append({"Method": "Threshold", "SD": sd, "Interval": None, "Accuracy": acc})

**(Probability classification)**

        true\_labels, preds = prob\_class(female, male, f\_mean, sd, m\_mean, sd)

        cm, acc = evaluate(true\_labels, preds, label="Probability Classifier")

        results.append({"Method": "Probability", "SD": sd, "Interval": None, "Accuracy": acc})

**(Quantized classification)**

        for interval in intervals:

            true\_labels, preds = quant\_class(female, male, interval)

            cm, acc = evaluate(true\_labels, preds, label=f"Quantized (Interval={interval})")

            results.append({"Method": "Quantized", "SD": sd, "Interval": interval, "Accuracy": acc})

**(Savin results to .csv file)**

    df = pd.DataFrame(results)

    df.to\_csv('results.csv', index=False)

    print("\nResults saved to 'results.csv'.")

run()

**OBSERVATIONS:**

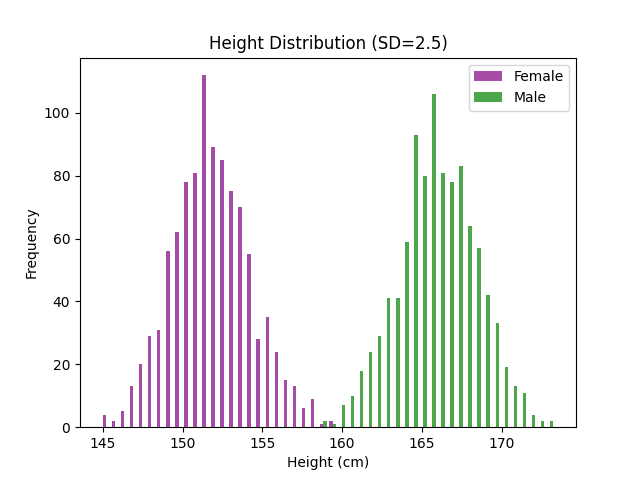
As the standard deviation (SD) increases, the accuracy of the classification methods tends to decrease. This happens because the distributions of male and female heights start overlapping more. With a smaller SD (like 2.5), the difference between the two groups is clearer, making classifiers like the Threshold-based and Probability-based methods more effective. However, as the SD increases (such as to 10), the distributions spread out, making it harder to distinguish between the genders. The Threshold-based method struggles more with this overlap, while the Probability-based method fares slightly better, as it considers the entire distribution, though its accuracy still declines as SD grows.

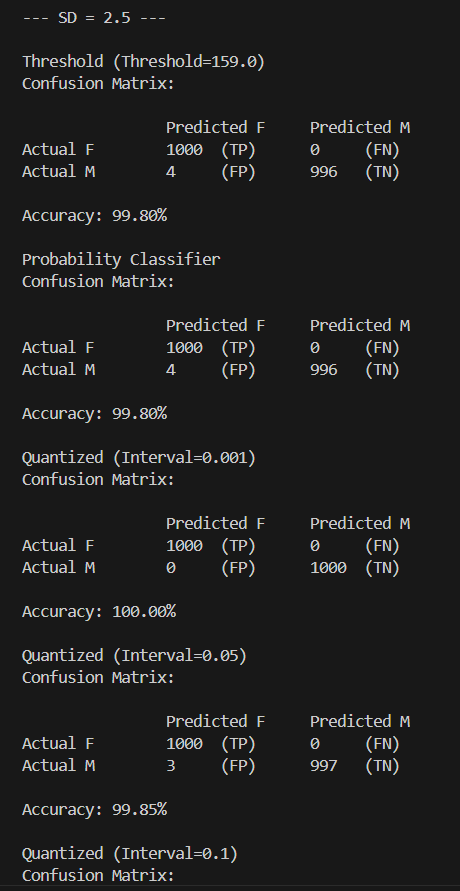
The Quantization method, which divides the data into intervals, is less sensitive to SD. It performs reasonably well even with larger SD values. This method works best when the intervals are not too small (like 0.001 cm) or too large (like 5 cm). If the intervals are too small, the method may overfit the data, while larger intervals result in rougher outcomes.

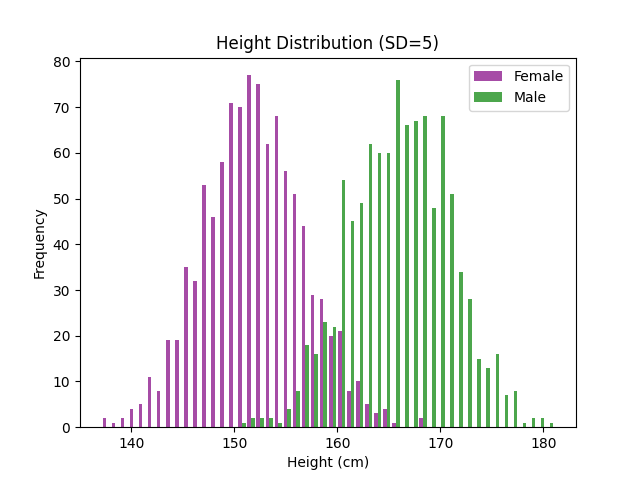
The confusion matrices reveal that as SD increases, misclassifications become more frequent. The Threshold method tends to have more False Negatives, while the Probability-based method results in more False Positives. The Quantization method remains more stable, though its performance still depends on the SD and interval size.

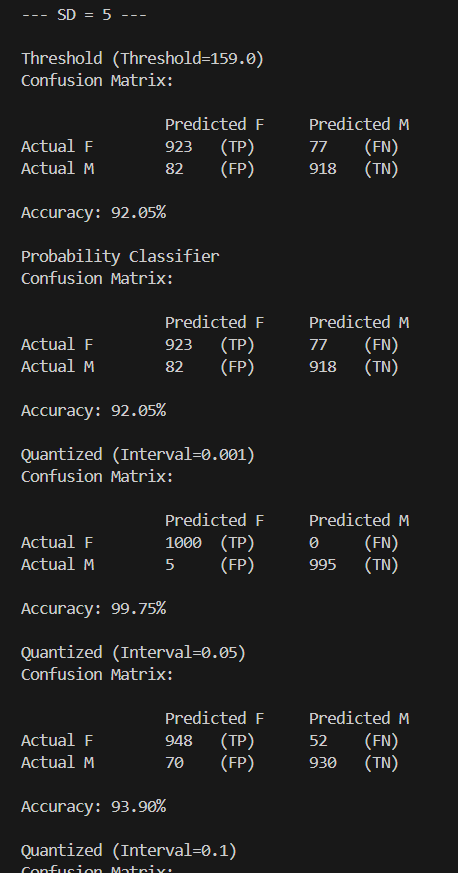
Overall, accuracy improves with lower SD values, and the Probability-based method handles SD changes most effectively. In the future, tweaking quantization intervals or utilizing machine learning models could help with cases where the distributions overlap more.

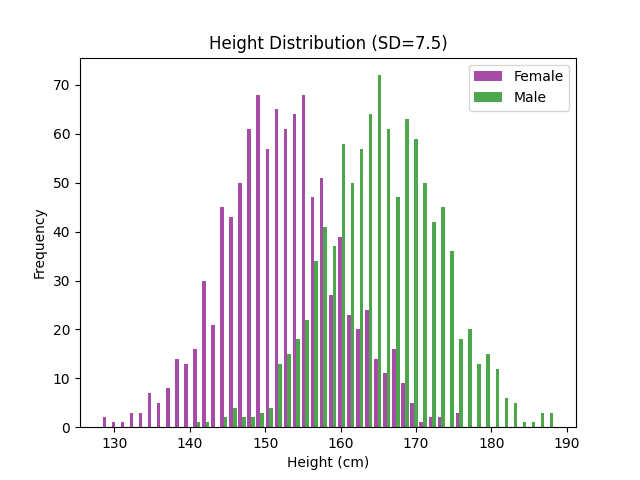
HISTOGRAMS:

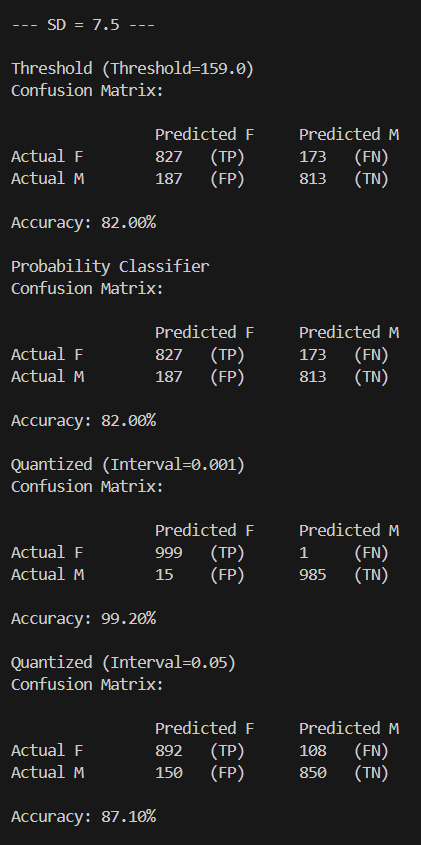


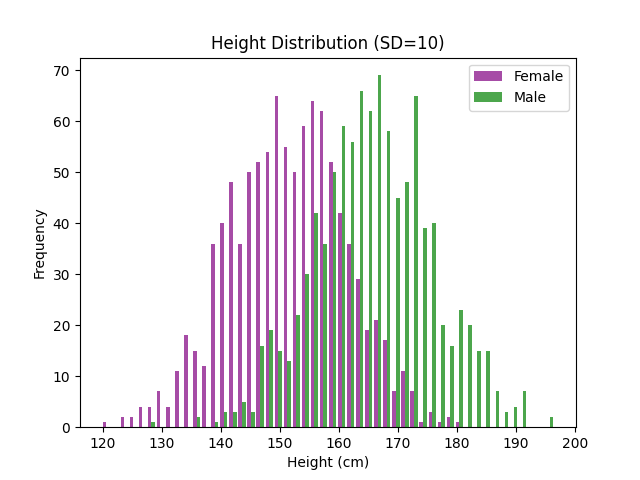


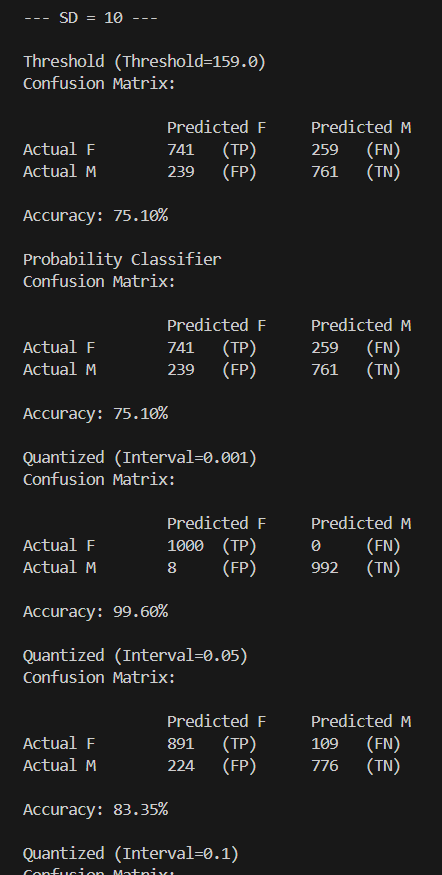












The accuracy for diff S.D’s is observed as we can seee int the above screenshots.

With that the confusion matrix having True and false positives and negatives for each are observed for both male and female therby giving us the accuracy and also the accuracy goes on decreasing gradually for the S.D. and for the increment in intervals as asked for us to do in the question 0.001 , 0.1 ,1 and so on…..

As you can see as the S.D. inceases the and the scale increase too the last image for S.D.=10 tells us most of the points lie in the middle and some points are far away thereby giving us the bell shaped kinda shape

To the left and right there is a thing observed that is there is scarcity of points as we move form the middle portion.

And at last the Matrix Classifier results are stored in .csv file :

