Machine Learning Lab Assignment 1

Shreeyash S. Dongarkar

22510025 ( T2 )

*import* numpy *as* np  
*from* scipy.stats *import* norm  
*import* matplotlib.pyplot *as* plt

*def* generate\_dataset(std\_deviation):  
 *"""  
 Generates a random dataset with given standard deviation and mean using a Gaussian Distribution ( Normal Distribution )  
 """* male\_heights = np.random.normal(mean\_male\_height, std\_deviation, size)  
 female\_heights = np.random.normal(mean\_female\_height, std\_deviation, size)  
   
 *return* male\_heights, female\_heights

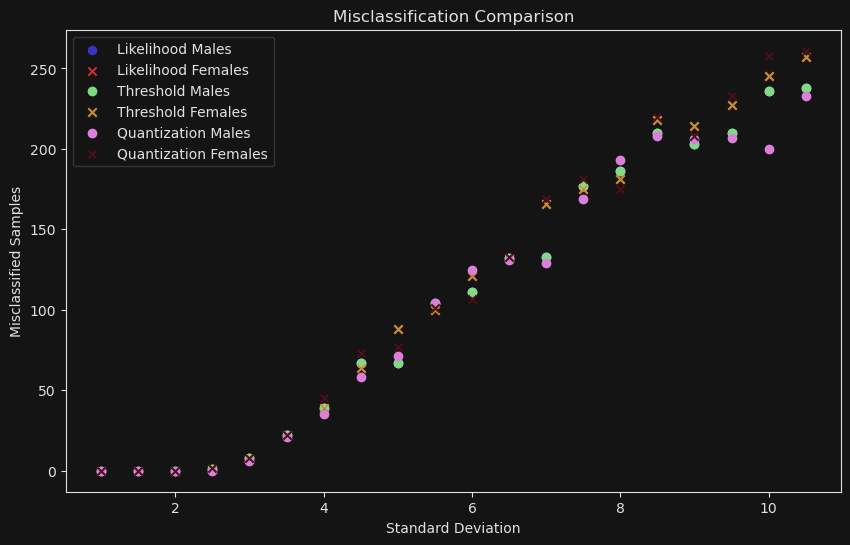
*def* likelihood(male\_heights, female\_heights, std\_deviation):  
 *"""  
 Calculates the likelihood of samples i.e. the probability density of each sample belonging to male and female distributions and compares to make prediction and counts the misclassified samples.  
 """* male\_likelihood\_m = norm.pdf(male\_heights, mean\_male\_height, std\_deviation)  
 female\_likelihood\_m = norm.pdf(male\_heights, mean\_female\_height, std\_deviation)  
 misclassified\_males = np.sum(female\_likelihood\_m > male\_likelihood\_m)  
  
 male\_likelihood\_f = norm.pdf(female\_heights, mean\_male\_height, std\_deviation)  
 female\_likelihood\_f = norm.pdf(female\_heights, mean\_female\_height, std\_deviation)  
 misclassified\_females = np.sum(male\_likelihood\_f > female\_likelihood\_f)  
  
 *return* misclassified\_males, misclassified\_females

*def* threshold(male\_heights, female\_heights, threshold\_val):  
 *"""  
 Male heights that are less than or equal to the given threshold will be misclassified as female and female heights that are greater than the given threshold will be misclassified as male.  
 """* misclassified\_males = np.sum(male\_heights <= threshold\_val)  
 misclassified\_females = np.sum(female\_heights > threshold\_val)  
  
 *return* misclassified\_males, misclassified\_females

*def* quantization(male\_heights, female\_heights, interval):  
 *"""  
 To predict which data point belongs to which bucket, we first need to determine the bin intervals (ranges). Once the intervals are defined, we identify the upper bound of each bin. To classify a data point, we compare its value with these intervals and assign it to the corresponding bin.  
 The prediction for a given bin is based on the mode (most frequent value) of the data points within that bin. The majority class in that bin will be the predicted class for any new data point falling into that bin. If a data point belongs to a bin but does not match the mode of that bin, it is considered misclassified.  
 The statement side = "right" is used to handle cases where a data point matches the exact end of an interval. Since the interval is not inclusive of the upper bound, the data point is assigned to the next interval. This ensures that any data point equal to the upper bound of an interval is classified into the subsequent bin.  
 """* min\_height = *min*(male\_heights.min(), female\_heights.min())  
 max\_height = *max*(male\_heights.max(), female\_heights.max())  
 bins = np.arange(min\_height, max\_height + interval, interval)  
  
 male\_counts = np.zeros(*len*(bins) - 1)  
 female\_counts = np.zeros(*len*(bins) - 1)  
  
 *for* height *in* male\_heights:  
 bin\_index = np.searchsorted(bins, height, side='right') - 1  
 *if* 0 <= bin\_index < *len*(male\_counts):  
 male\_counts[bin\_index] += 1  
  
 *for* height *in* female\_heights:  
 bin\_index = np.searchsorted(bins, height, side='right') - 1  
 *if* 0 <= bin\_index < *len*(female\_counts):  
 female\_counts[bin\_index] += 1  
  
 misclassified\_males = 0  
 misclassified\_females = 0  
  
 *for* i *in range*(*len*(male\_counts)):  
 *if* female\_counts[i] > male\_counts[i]:  
 misclassified\_males += male\_counts[i]  
 *else*:  
 misclassified\_females += female\_counts[i]  
  
 *return* misclassified\_males, misclassified\_females

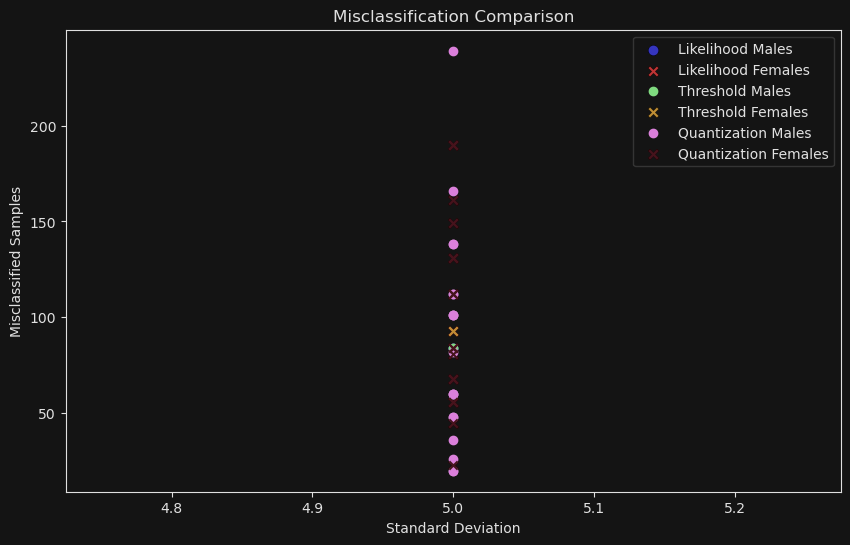
*def* plot\_misclassification(mean\_male\_height, mean\_female\_height, std\_deviation\_values, intervals, size = 1000):  
 likelihood\_males = []  
 likelihood\_females = []  
 threshold\_males = []  
 threshold\_females = []  
 quantization\_males = []  
 quantization\_females = []  
  
 *for* std\_deviation *in* std\_deviation\_values:  
 *print*(f"\nSD: {std\_deviation} =>")  
 male\_heights, female\_heights = generate\_dataset(std\_deviation)  
  
 *# Likelihood Misclassification* misclassified\_males, misclassified\_females = likelihood(male\_heights, female\_heights, std\_deviation)  
 *print*("\nLIKELIHOOD CLASSIFICATION:")  
 *print*(f"Misclassified Males: {misclassified\_males}")  
 *print*(f"Misclassified Females: {misclassified\_females}")  
 likelihood\_males.append(misclassified\_males)  
 likelihood\_females.append(misclassified\_females)  
  
 *# Threshold Misclassification* mean = (mean\_male\_height + mean\_female\_height) / 2  
 misclassified\_males, misclassified\_females = threshold(male\_heights, female\_heights, mean)  
 *print*("\nTHRESHOLD CLASSIFICATION:")  
 *print*(f"Misclassified Males: {misclassified\_males}")  
 *print*(f"Misclassified Females: {misclassified\_females}")  
 threshold\_males.append(misclassified\_males)  
 threshold\_females.append(misclassified\_females)  
  
 *# Quantization Misclassification  
 print*("\nQUANTIZATION CLASSIFICATION:")  
 *for* interval *in* intervals:  
 *print*(f"Quantization Interval: {interval}")  
 misclassified\_males, misclassified\_females = quantization(male\_heights, female\_heights, interval)  
 *print*(f"Misclassified Males: {misclassified\_males}")  
 *print*(f"Misclassified Females: {misclassified\_females}\n")  
 quantization\_males.append(misclassified\_males)  
 quantization\_females.append(misclassified\_females)  
  
 *print*("--------------------------------------------------------------------------------")  
  
 *# Plotting* plt.figure(figsize=(10, 6))  
  
 *# Likelihood* plt.scatter(std\_deviation\_values, likelihood\_males, color='blue', label='Likelihood Males', marker='o')  
 plt.scatter(std\_deviation\_values, likelihood\_females, color='red', label='Likelihood Females', marker='x')  
  
 *# Threshold* plt.scatter(std\_deviation\_values, threshold\_males, color='green', label='Threshold Males', marker='o')  
 plt.scatter(std\_deviation\_values, threshold\_females, color='orange', label='Threshold Females', marker='x')  
  
 *# Quantization* repeated\_std = np.repeat(std\_deviation\_values, *len*(intervals))  
 plt.scatter(repeated\_std, quantization\_males, color='purple', label='Quantization Males', marker='o')  
 plt.scatter(repeated\_std, quantization\_females, color='pink', label='Quantization Females', marker='x')  
  
 plt.xlabel('Standard Deviation')  
 plt.ylabel('Misclassified Samples')  
 plt.title('Misclassification Comparison')  
 plt.legend()  
 plt.show()

mean\_male\_height = 166  
mean\_female\_height = 152  
size = 1000  
  
*# For same interval mis-classification increases as standard deviation goes on increasing*std\_deviation\_values = np.arange(1, 11, 0.5)  
intervals = [1]  
  
*# May lead to overfitting as we may not be able to capture noises and complex patterns as data is split into many small segments  
# intervals = [0.00001]  
  
# For same standard deviation mis-classification increases as interval goes on increasing  
# std\_deviation\_values = [5]  
# intervals = np.arange(0.5, 10, 0.5)*plot\_misclassification(mean\_male\_height, mean\_female\_height, std\_deviation\_values, intervals, size)



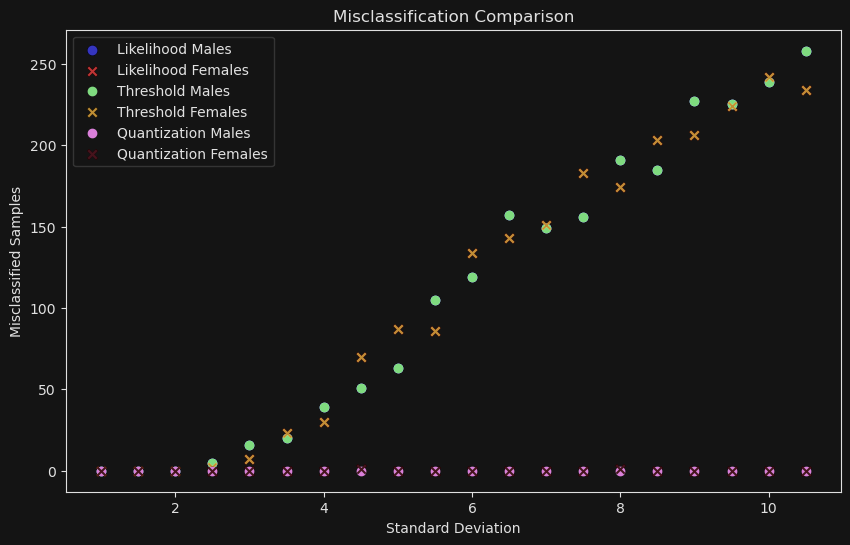
OBSERVATIONS

* As the value of standard deviation increases ( for the same size of interval ) misclassification goes on increasing as higher standard deviation implies data is more spread from mean resulting in greater overlap between two distributions.
* In case of likelihood for greater standard distribution more samples from one distribution may fall into region dominated by other distribution thus increasing misclassification.
* In case of binning more spread due to greater standard distribution leads to more samples falling into overlapping bins thus increasing the count of samples being misclassified



OBSERVATIONS

* As the size of the interval (or bin) increases, while keeping the standard deviation constant, the number of misclassifications typically increases. This occurs because when the bin size is large, the total number of bins decreases. With fewer bins, the range of values within each bin becomes larger. As a result, samples at the extreme ends of the distribution are more likely to be classified into an incorrect bin, leading to higher chances of misclassification.



OBSERVATIONS

* For very small size of interval misclassification remains the same irrespective of changing standard deviation as the data is split into fine bins but this can be case of overfitting thus leading to fail in identifying complex patterns.