**Batch -** T7
**Assignment No. -** 4
**Title –** Multivariate Classifiers from First Principles
**Student Name -** Sharaneshwar Bharat Punjal
**Student PRN -** 23520011

Objective of this assignment is to design multivariate classifiers from first principles. You may choose to either extend the univariate methods from assignment 1, for the multivariate scenarios or design new approaches.
However, do not use any off the shelf classification algorithms.
Please note that this assignment does not require you to submit the code.
However, you may choose to implement your methods to explore concepts.

The task of your classifier is gender identification, based on measured parameters. We have a toy data set of 1000 male and 1000 female (labelled) samples. We attempt a height based univariate classifier and realize that due to overlap in heights, there are limitation on improving accuracy meaningfully without exploring other features. So, we decide to identify and add new features to our learning algorithms to reduce the prediction error further.
Consider following scenarios of increasing complexity.

## 1. Uncorrelated input features (5 marks)

a. We have two input features say Height (in cm) and Heamoglobin levels measured for all 2000 samples. Let's assume that both these features are normally distributed within each gender. These features are pretty much uncorrelated within each gender.
b. Can you design approaches to train a classification algorithms to predict gender?

**Answer -**
**Approach 1:**
Consider 2 dataset
1. Height
2. Haemoglobin
Now we will generate the normal distribution for both the attribute with std = 5 for height and std = 2 for haemoglobin.

1. Likelihood Classification:
We will take the independent probabilities and then multiply them if we are considering the male probability and if we get the female probability higher then we will misclassify that point same for females.

P (Height, Haemoglobin / Gender) = P(Height/Gender) * P(Haemoglobin/Gender)

2. Threshold Classification:
For this we will make assign the threshold value for male and female for both height and haemoglobin.
If (male_ht > threshold_ht && male_haem > threshold_haem) classify as “Male”.
If (female_ht < threshold_ht && female_haem < threshold_haem) classify as “Female”.
Else in any case we will treat that point as misclassified.

3. Quantization:
In quantization we will take the cross product of the bins and then for each value we will take the count of the male and the female points and then on the basis of that we will calculate the misclassification.
Eg:
Height_bins = [ (158, 160), (161, 163), (164, 165) ...]
Haem_bins = [ (11.4, 11.7), (11.8, 12), (12, 12.7) ...]
Now we will take the cross product of this.
Bins = [ ((158, 160), (11.4, 11.7)), ((158, 160), (11.8, 12)) ...]
Like this we will make all the bins and then we will calculate the no. of male and females on each point and on the basis of that we will calculate the misclassification rate.

**Approach 2:**
Linear regression can also be used as a classification algorithm. Here’s the conceptual implementation:
1. Consider the two features: Height and Haemoglobin.
2. Treat gender as a binary variable by assigning 1 for male and 0 for female.
3. Perform linear regression to fit a line to predict gender based on the input features:
   $Y=\beta 0 + \beta 1\times$Height $+ \beta 2\times$Haemoglobin, where Y is the predicted probability for gender.
4. Apply a threshold of 0.5:
   If $Y>0.5$, classify the sample as male.
   If $Y\leq 0.5$, classify the sample as female.

## 2. Input features with non-zero correlation (3 marks)

a. In this scenario, we have two input features say, Height (in cm) and Weight (in kg) measured for all 2000 samples. Both are normally distributed within each gender. The correlation between these features is 0.6 within each gender.
b. Which of the algorithms you designed for uncorrelated features would work as is? If they don't, what changes can you make to your algorithms to accommodate correlations?

**Answer –**

The algorithms we used for uncorrelated features don’t work well here because they assume that height and weight are independent, meaning one doesn't affect the other. But in reality, with a correlation of 0.6, taller individuals are more likely to weigh more, and this relationship makes the features connected. For example, in **Likelihood Classification**, we multiply probabilities of height and weight separately, but this won't capture their combined effect. Similarly, in **Threshold Classification**, the height and weight thresholds can't work independently anymore because they’re linked—like someone tall but lighter might still be male. **Quantization** also struggles because overlapping bins from correlated features make it harder to separate males and females clearly. That’s why we need methods like **Linear Regression**, which can handle such relationships better.

**Approach 1 - Use Mahalanobis Distance**

Mahalanobis distance looks at both Height and Weight together and adjusts them, so the relationship between them is taken into account before applying the regression model.
This means that the model won’t get confused by the correlation and will make more accurate predictions.

**Approach 2 - Rotate the Features:**

Another way to fix the issue is by rotating the data (kind of like tilting the graph), so that Height and Weight are no longer related.
Once the features are rotated, they are no longer correlated, and the model can focus on each feature separately without confusion.

To solve the problem of correlation between Height and Weight, we can use a combination of both methods:
Step 1 - Use Mahalanobis distance to adjust the data so that the correlation doesn’t mess up the predictions.
Step 2 - Rotate the features to make them uncorrelated. This step makes sure the model treats Height and Weight as independent features.
Step 3 - Now that the features are adjusted and uncorrelated, we can safely apply linear regression to make predictions, without the model getting confused by the correlation.

By first adjusting the features using Mahalanobis distance and then rotating them, we can make sure the regression model works properly, even with correlated data. This way, the model can make more accurate predictions without being thrown off by the relationship between Height and Weight.

## 3. How far can we go? (2 marks)

a. We observe that accuracy improves with addition of one new feature in both of the above scenario. Can we reach a conclusion that accuracy can be improved further by adding multiple such features to the input? How many such features would you add in your quest to improve accuracy? Would addition of new features require any changes to the experimental set up?

**Answer –**

Adding more features can help improve accuracy at first, but it doesn’t always work. If we keep adding features, some might not be useful or could repeat the same information, which can confuse the model and make it worse at handling new data. So, it’s better to add features step by step and stop when the accuracy stops improving. If we add new features, we might also need to change the setup, like checking for relationships between features or normalizing the data, to keep things working properly.