**Assignment 4**

**Machine Learning**

Name: Jasmine I. Sayyad

PRN: 22510033

Batch: T2

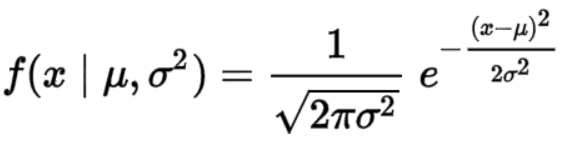
**1. Uncorrelated input features**

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?

**Approach 1:**

1. First let’s find the overlapping height region. Since height follows the normal distribution for both male and female we will first determine the overlapping region.
   1. Lower\_limit = Male\_ht.min()
   2. Upper\_limit = Female\_ht.max()
2. In next step, we will focus on the people who fall in this overlapping region, where classification based on height alone is unreliable.
3. Now use Haemoglobin to resolve this ambiguity. First find out the Probability Density Function (pdf) of haemoglobin for both male and female using their normal distributions to compute likelihood.
4. For each person in overlapping heights region, compare the pdf values i.e the likelihood. If female\_likelihood > male\_likelihood, classify as Female else male.



male\_likelihood = norm.pdf(hb\_level, np.mean(male\_hb), np.std(male\_hb)) female\_likelihood = norm.pdf(hb\_level, np.mean(female\_hb), np.std(female\_hb))

**2. Input features with non-zero correlation**

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 it is? If they don’t, what changes can you make to your algorithms to accommodate correlations.

**Answer:** No, the algorithm I designed for uncorrelated features that is height and haemoglobin won’t work as it is for the correlated features height and weight. Using PDFs separately for weight (like we did for haemoglobin) is incorrect, as height and weight are **jointly distributed**.

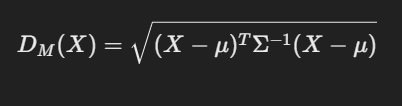
Instead of this we can use Mahalanobis distance because

* It will consider the relationship between height and weight
* Normalizes based on variance
* The smaller the Mahalanobis distance, the closer the sample is to that gender’s distribution.

Steps:

1. Calculate mean height & weight for males and females, compute the covariance matrix for each gender.



1. Then find out the inverse of covariance matrix.
2. Find Mahalanobis distance to Male mean and then to the female mean using below formula
3. classify based on the minimum distance i.e if distance to the female mean is less than the distance to male mean then classify as female else male.

Thus, this approach is simpler than computing probability densities and doesn’t require overlapping region handling.

**3. How far can we go?**

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:**

* Yes, adding more features to the input can help to improve the accuracy but up to certain limit only. Adding more features helps to improve accuracy because, it can provide some new/extra information to distinguish between data, this feature should add unique value instead of being correlated to already present feature (this won’t add much value).
* But if we start adding so many features to input, the number of dimensions will increase making learning more complex. We may face problems like overfitting means model will start memorizing training data but will fail while deriving output for the new data. More features may lead to the more computations.

How many such features would you add in your quest to improve accuracy?

* Instead of adding all the features at a time, start adding incrementally. Only adding the domain related features will be more useful. Avoid adding the features which are more correlated because they will not add much value.

Would addition of new features require any changes to the experimental set up?

* Yes, adding more features changes the entire experimental setup. As the number of features increases, the covariance matrix expands, moving from a simple **2×2 matrix** to an N×N matrix, where N represents the total number of features. This makes computations more complex.