LAB ASSIGNMENT 04 — Implementing Naive Bayes Classifier

Course: CSC-462 – Artificial Intelligence
CLO: CLO-6 — Apply machine learning and probabilistic reasoning to solve classification problems
Total Marks: 25

PART A Implementing Naive Bayes From Scratch *(12 Marks)*

Problem A1 Build a Naive Bayes Text Classifier (Coding Required) *(6 Marks)*

Write a Python program that implements a **Multinomial Naive Bayes classifier from scratch** (NO libraries for ML allowed only Python, NumPy allowed but optional).

Dataset (students create manually or provided by instructor):

A small text dataset with two classes:

- **Positive**
- **Negative**

Example documents:

- Positive: ["I love this food", "This movie is great", "Amazing experience"]
- Negative: ["I hate this taste", "The movie was bad", "Worst experience ever"]

Your program must:

1. Preprocess text:
    - Convert to lowercase
    - Remove punctuation
    - Tokenize into words
2. Build vocabulary from training documents.
3. Compute:
    - Prior probabilities P(Class)
    - Likelihood probabilities P(word | Class) with **Laplace smoothing**
4. Implement Naive Bayes classification using:

$$P(Class \mid Document) \propto P(Class) \prod_{word \in doc} P(word \mid Class)$$

5. Test your classifier on **3 new documents** and print predicted class.

Output Requirements:

- Print vocabulary
- Print priors and likelihood tables
- Print class prediction for each test document
- Print intermediate probability calculations for at least one example

Problem A2 Analyze the Classifier's Behavior *(6 Marks)*

Answer the following questions based on your implementation:

a. How does Laplace smoothing change the likelihood values?
b. What happens when the dataset is extremely small?
c. Why does Naive Bayes assume independence between features, and how does this simplify computation?
d. Give one scenario where Naive Bayes performs surprisingly well, and one scenario where it usually performs poorly.

PART B Naive Bayes for Numerical Features *(13 Marks)*

Problem Statement

You are required to implement a Naive Bayes classifier (discrete version, non-Gaussian) for a dataset containing numerical features.
Since Naive Bayes (discrete) requires categorical features, your first task is to convert the numerical values into discrete bins, then compute probability tables manually.

You must not use any machine-learning libraries (e.g., sklearn).
Only Python, math, and NumPy are allowed.

Dataset

Use the following dataset containing two numerical features: Age and Income.
Your job is to convert these numeric features into discrete bins, then build Naive Bayes probability tables.

Age	Income (k USD)	Class
23	25	No
45	60	Yes
35	45	Yes
52	110	Yes
33	30	No
28	28	No
40	85	Yes
60	95	Yes
25	40	No

Task A Feature Discretization *(5 Marks)*

1. **Convert Age into 3 bins:**
    - **Young (≤30)**
    - **Mid (31–50)**
    - **Senior (>50)**
2. **Convert Income into 3 bins:**
    - **Low (≤40)**
    - **Medium (41–80)**
    - **High (>80)**
3. **Add the discretized columns to your dataset.**

Output:

- **Print the final transformed table.**

Task B Build Naive Bayes Model (Discrete) *(10 Marks)*

You must compute manually:

1. **Prior probabilities:**

$$P(\textbf{Class = Yes}), P(\textbf{Class = No})$$

2. **Likelihood tables:**
   Example:

$$P(\textbf{Age = Young} \mid \textbf{No})$$
$$P(\textbf{Income = High} \mid \textbf{Yes})$$

3. **Apply Laplace smoothing (add-one smoothing) to avoid zero probabilities.**
4. **For the following test sample:**

Age Income

34 72

Convert it into discrete bins and compute:

$$P(\textbf{Yes} \mid \textbf{X}), P(\textbf{No} \mid \textbf{X})$$

5. **Predict the class label by selecting the class with the higher posterior probability.**

Output required:

- **All prior probability calculations**
- **All likelihood tables**
- **Posterior probability computations**
- **Final predicted class**

Submission Requirements

- **Python code**
- **Printed tables for priors, likelihoods, and posteriors**
- **PDF containing answers to Task C**
- **All steps must be clearly explained**