Programming Assignment-3: Naive Bayes and Decision Trees (ID3)

Objective

This assignment aims to provide hands-on experience in implementing and understanding two fundamental machine learning algorithms: Naive Bayes Classifier and Decision Trees (ID3 algorithm). Implement these algorithms from scratch and evaluate their performance on provided datasets.

Problem 1: Naive Bayes Classifier (1 marks)

Use the datasets provided to solve the problems assigned below:

DATASET-1 : Binary Class data with two features - Naive-Bayes-Classification-Data.csv

Problem Description

In this section, you will implement the Naive Bayes algorithm using a dataset involving binary class having multiple features.

Instructions

Complete the following steps for implementing and evaluating the Naive Bayes Classifier:

1. **Dataset Generation/Loading:**
    - Use the **DATASET-1** for this problem.
2. **Data Splitting:**
    - Split both datasets into training and testing sets (e.g., 70% training, 30% testing) using `sklearn.model_selection.train_test_split`.
3. **Naive Bayes Implementation:**
    - Implement the **Naive Bayes algorithm from scratch**.
    - Your implementation should include:
        - A method to calculate the prior probabilities for each class.
        - A method to calculate the likelihood (probability density function) for each feature given each class, assuming a Gaussian distribution.

        - A `fit` method that learns these probabilities from the training data.
        - A `predict` method that uses Bayes' theorem to classify new data points.
4. **Model Training:**
    - Train your Naive Bayes model on the training data.
5. **Prediction:**
    - Make predictions on the test sets for both datasets.
6. **Evaluation:**
    - For test dataset, calculate and report the following metrics:
        - Accuracy score.
        - Precision, Recall, and F1-score (using `sklearn.metrics.classification_report`).
        - Confusion Matrix (using `sklearn.metrics.confusion_matrix`).
    - Discuss the performance of your Naive Bayes classifier binary class problem.

7. **Visualization:**
    - Visualize the decision boundaries learned by your Naive Bayes model using the two features of the dataset (test).

Problem 2: Decision Tree (ID3 Algorithm) (1.5 marks)

Goal

Implement the ID3 Decision Tree algorithm from scratch and evaluate its performance on a classification task

DATASET-2 : Multiple Class with Multiple features - car_evaluation.csv The Car Evaluation Database contains the structural information that directly relates CAR to the six input attributes: *buying, maint, doors, persons, lug_boot, safety.*

Information about Attribute Information:

Class Values*(Decision about Car purchase):* unacc, acc, good, vgood (*Unacceptable, Acceptable, Good, Very Good*)

Attributes (Features)*: Depending upon the following attribute decision has to be made.*

1. buying: vhigh, high, med, low. (*Buying Price*)
2. maint: vhigh, high, med, low. (*Maintenance Cost*)
3. doors: 2, 3, 4, 5more. (*Number of Doors*)
4. persons: 2, 4, more. (number of persons)

5. lug_boot: small, med, big (size of luggage boot)
6. safety: low, med, high (safety)

Problem Description

In this section, you will implement the ID3 algorithm, which uses information gain to construct a decision tree. You will work with DATASET-2.

Instructions

Complete the following steps for implementing and evaluating the ID3 Decision Tree algorithm:

1. **Dataset Preparation:**
    - Use **DATASET-2** for this problem.
    - Represent your data using Pandas DataFrames.
2. **Data Splitting:**
    - Split the dataset-2 into a training and testing set (e.g., 70% training, 30% testing) using `sklearn.model_selection.train_test_split`.
3. **ID3 Algorithm Implementation:**
    - Implement the ID3 algorithm from scratch. Your implementation should include:
        - A function to calculate the Entropy of a dataset.
        - A function to calculate the Information Gain for a given feature.
        - A recursive function to build the decision tree:
            - It should select the best feature to split on based on information gain.
            - It should handle pure nodes (all samples belong to the same class).
            - It should handle cases where no more features are available or no information gain can be achieved (majority class prediction).
            - The tree structure can be represented using nested dictionaries or custom tree node objects or json objects.
        - A `predict` method that traverses the built tree to classify a new data point.
4. **Model Training:**
    - Train your ID3 Decision Tree model on your train dataset.
5. **Prediction:**
    - Make predictions on test data points using your trained tree.
6. **Evaluation (Conceptual):**
    - For test dataset, calculate and report the following metrics:

        - Accuracy score.

7. **Visualization/Representation of the Tree:**
    - Represent the learned decision tree in a human-readable format (e.g., print the tree structure, or use a library like `graphviz` if time permits). If the tree is very large, represent up to a depth of 3.
    - Write function tree_depth for finding depth and breadth of decision tree.

Submission Guidelines:

- Submit a single Jupyter Notebook with all your code.
- Clearly separate Problem 1 (Naive Bayes) and Problem 2 (ID3 Decision Tree) with markdown headings.
- Ensure all code blocks are executed and produce the expected outputs.
- Add comments to explain your code, especially for custom implementations and complex logic.
- Ensure your code is well-structured and easy to read.
- For any plots, include appropriate titles, axis labels, and legends.

Evaluation Criteria

Your assignment will be evaluated based on the following:

- **Correctness:** Proper implementation of Naive Bayes and ID3 algorithms.
- **Functionality:** The code runs without errors and produces expected outputs.
- **Evaluation Metrics:** Accurate calculation and interpretation of classification metrics.
- **Code Quality:** Readability, comments, and adherence to Python best practices.
- **Understanding:** Demonstrated understanding of the underlying concepts of both algorithms, including their assumptions and limitations.

Due Date

Please submit your completed Jupyter Notebook notebook by Oct 15, 2025 12:00 AM GMT+5:30 for full marks (**by 11.59 pm of 15th October, 2025**).

Last submission deadline is (where points will be reduced by 50%) Oct 16, 2025 12:00 AM GMT+5:30 (**by 11.59 pm of 16th October, 2025**)