Assignment 7
Name: Yash Maske
Roll No: 282007
Batch: B1

Statement

In this assignment, the goal is to: a) Apply a **Decision Tree Classifier** to predict student admission based on GRE scores and academic achievements.
b) Preprocess the data by managing missing values, encoding categorical variables, and normalizing if needed.
c) Split the dataset into training and test sets for training and evaluation.
d) Assess the classifier's performance using **accuracy**, **confusion matrix**, **precision**, **recall**, and **F1-score**.
e) Visualize the dataset and model results using histograms, scatter plots, and a decision tree diagram.

Objective

1. Learn preprocessing techniques for classification problems.
2. Build and evaluate a Decision Tree Classifier using Scikit-Learn.
3. Interpret model results using standard classification metrics.
4. Visualize data features and relationships to support analysis.

Resources Used

- **Software**: Visual Studio Code
- **Libraries**: Pandas, Seaborn, Matplotlib, Scikit-Learn

Introduction to Classification

Classification is a supervised machine learning method used to assign labels to data based on input features. The **Decision Tree Classifier** models decisions in a tree-like structure by splitting features based on specific thresholds. It is easy to interpret and works well on structured datasets like student admission records.

Functions and Methods Utilized

1. pd.read_csv() – Load data into a Pandas DataFrame.
2. .dropna() / .fillna() – Handle missing data using suitable imputation methods.
3. train_test_split() – Divide data for training and testing.

4. DecisionTreeClassifier() – Create the decision tree model.
5. accuracy_score(), confusion_matrix(), classification_report() – Evaluate classifier performance.

Methodology

1. Data Collection and Exploration

- Loaded the **Graduate Admissions** dataset using Pandas.
- Explored data types, checked for missing values, and examined basic structure.

2. Data Preprocessing

- Imputed missing values using median or mean strategies.
- Converted the target column *"Chance of Admit"* into a binary variable:
    - 1 if chance ≥ 0.5
    - 0 if chance < 0.5
- Selected relevant features such as **GRE Score** and **CGPA**.
- Normalized features if necessary for consistency.

3. Summary Statistics

- Used .describe() and NumPy functions to obtain:
    - Mean, median, max, min
    - Standard deviation, variance, and percentiles

4. Data Visualization

- Plotted histograms using sns.histplot() to visualize individual feature distributions.
- Used scatter plots to observe correlations between GRE scores and CGPA.

5. Model Training and Evaluation

- Split dataset using train_test_split() with 80% for training and 20% for testing.
- Trained a **Decision Tree Classifier**.
- Predicted test results and evaluated performance with:
    - **Accuracy Score**
    - **Confusion Matrix**
    - **Precision, Recall, and F1-score** using classification_report()
- Plotted the decision tree for interpretation.

Advantages

- Pandas allows easy manipulation of datasets.
- Decision Trees are intuitive and interpretable.
- Visualization helps in understanding data trends.
- Classification metrics provide detailed performance insights.

Disadvantages

- Larger datasets may consume significant memory.
- Decision Trees can overfit on training data without pruning.
- Class imbalance can negatively affect model performance.

Conclusion

This assignment illustrated how a **Decision Tree Classifier** can be used to predict student admission chances using academic performance data. The process involved data cleaning, transformation, model training, and detailed performance evaluation. Libraries like Pandas, Seaborn, and Scikit-Learn streamlined both analysis and visualization, enhancing understanding of the classification process.