Name: Pathan Firdos

Roll No: 281073

Batch: A3

Practical 7

Practical Overview

This practical involves the application of data pre-processing techniques and machine learning to predict student admissions using the Admission_Predict.csv dataset. The focus is on cleaning and transforming the data, preparing it for analysis, and training a Decision Tree Classifier to forecast whether a student will be admitted based on their academic profile. The model's performance is then evaluated using various classification metrics.

Objectives

- **Data Preprocessing:**
  - Load and inspect the dataset using Pandas.
  - Handle missing values through imputation or removal.
  - Normalize or standardize features to improve model performance.
  - Apply Label Encoding to categorical variables when necessary.
- **Data Preparation:**
  - Split the dataset into training (80%) and testing (20%) sets to validate model performance effectively.
- **Model Development:**
  - Train a Decision Tree Classifier using Scikit-learn on the training data.
  - Tune hyperparameters such as max_depth and criterion for better results.
- **Model Evaluation:**
  - Generate a confusion matrix to visualize model predictions.
  - Calculate key metrics including Accuracy, Precision, Recall, and F1-score to assess the classifier's effectiveness.

Resources Utilized

- **Software:**

- Visual Studio Code

- **Libraries:**
  - Pandas
  - NumPy
  - Scikit-learn
  - Matplotlib
  - Seaborn

Theoretical Background

Classification:
Classification is a supervised learning process where a model learns to assign labels to data points. In this assignment, the classifier determines whether a student will be admitted based on several features such as GRE Score, TOEFL Score, University Rating, SOP Strength, LOR Strength, Undergraduate GPA, and Research Experience.

Decision Tree Classifier:

- A Decision Tree is structured like a flowchart:
  - **Internal Nodes:** Decision points based on feature values.
  - **Branches:** Outcomes of each decision.
  - **Leaf Nodes:** Final class labels (admitted or not admitted).
- The algorithm recursively splits the dataset to minimize classification errors, making it a straightforward yet powerful method for classification tasks.

Methodology

1. **Data Preprocessing:**
   - **Dataset Loading:**
     Use Pandas to load the Admission_Predict.csv file and perform an initial data inspection.
   - **Handling Missing Values:**
     Identify any missing data and address them through imputation or removal to maintain data integrity.

   - **Normalization and Transformation:**
     Normalize numerical features if required and apply Label Encoding to convert categorical data into a suitable numeric format.

2. **Train-Test Split:**

   - Divide the preprocessed dataset into training (80%) and testing (20%) sets using Scikit-learn's train_test_split function to ensure that the model is validated on unseen data.

3. **Model Training and Prediction:**

   - **Model Implementation:**
     Train a Decision Tree Classifier using Scikit-learn.

   - **Hyperparameter Tuning:**
     Adjust parameters like max_depth and criterion to optimize the classifier's performance.

   - **Prediction:**
     Use the trained model to predict student admissions on the test set.

4. **Model Evaluation:**

   - **Confusion Matrix:**
     Generate a confusion matrix to clearly visualize the True Positives, True Negatives, False Positives, and False Negatives.

   - **Performance Metrics:**
     Compute Accuracy, Precision, Recall, and F1-score to evaluate how well the model predicts student admissions.

Conclusion

The Decision Tree Classifier effectively predicted student admissions based on the provided academic metrics. The data pre-processing steps ensured that the dataset was clean and well-prepared, while the train-test split facilitated an unbiased evaluation of the model. Performance metrics, including Accuracy, Precision, Recall, and F1-score, demonstrated the model's capability in distinguishing between admitted and non-admitted students. Further improvements could be achieved through advanced feature engineering or by experimenting with alternative classification algorithms, such as Random Forest or Logistic Regression, to enhance prediction accuracy and reliability.