

Assignment 4

Aim: Implementation of Decision Tree Classifier on Gender Classification Dataset

Objective: To implement and evaluate a Decision Tree Classifier using Python to predict gender based on physical and behavioral features in the gender classification dataset.

Introduction:

Importance of Decision Tree:

Decision Trees are a powerful supervised learning algorithm for classification and regression tasks. They split the dataset into branches based on feature values to make predictions. In this assignment, we preprocess the **Gender classification dataset**, train a **Decision Tree Classifier**, analyze its performance, and visualize the results.

Advantages of Decision Trees:

- **Easy to Interpret and Understand:** The tree structure is intuitive.
- **Handles Both Categorical and Numerical Data:** Unlike many other algorithms, Decision Trees can process mixed data types.
- **Minimal Data Preprocessing Required:** No need for extensive feature scaling or transformation.
- **Captures Non-Linear Relationships:** Can efficiently model complex decision boundaries.
- **Useful for Feature Selection:** Identifies important variables based on how frequently they are used to split nodes.

Dataset:

The dataset used in this assignment is the **Gender Classification Dataset**, containing various features extracted from physical traits and characteristics. It includes both numerical and categorical attributes. The key features include:

- `long_hair`: Presence of long hair (0/1)
- `forehead_width_cm`: Width of forehead in centimeters
- `forehead_height_cm`: Height of forehead in centimeters
- `nose_wide`: Wide nose or not
- `nose_long`: Long nose or not
- `lips_thin`: Thin lips or not
- `distance_nose_to_lip_long`: Long distance from nose to lip
- `gender`: Target variable representing the gender (Male/Female)

Steps of Implementation:

1. Importing Libraries:

- Python libraries such as Pandas, NumPy, Matplotlib, Seaborn, and Scikit-Learn are used for data handling, visualization, and model training.

2. Loading the Dataset:

- The **Gender classification dataset** is imported using Pandas, and an initial exploration is performed using `.shape()`, `.head()`, and `.info()`.
3. **Data Preprocessing:**
 - **Encoding categorical variables**
 - Handling **missing values** by filling categorical columns with **mode** and numerical columns with **median**.
 - **Feature selection**
 - **Splitting the dataset** into 67% training and 33% testing.
 4. **Training the Decision Tree Model:**
 - A **Decision Tree Classifier** with **Gini Index** as the splitting criterion and a maximum depth of **3** is used.
 5. **Making Predictions:**
 - The trained model is used to **predict Gender** on the test dataset.
 6. **Model Evaluation:**
 - **Accuracy Score:** Measures the overall correctness of the model.
 - **Confusion Matrix:** Provides insights into classification errors, showing false positives and false negatives.
 - **Classification Report:** Displays precision, recall, and F1-score for evaluating model performance.
 7. **Visualization of Results:**
 - **Decision Tree Visualization:** The structure of the Decision Tree is plotted to help interpret the decision-making process based on input features.

Conclusion:

- The Decision Tree Classifier was successfully implemented to classify gender based on facial features and traits.
- **Accuracy Score** indicates how well the model performs on unseen data.
- **Confusion Matrix** helps identify classification errors.
- **Classification Report** offers insights into precision, recall, and F1-score.
- **Tree Visualization** provides an intuitive understanding of the model's logic and key features used for decision-making.

References:

- [GeeksforGeeks - Decision Tree](#)
- Github Repository: <https://github.com/Utkarsh-Rane43/ML-LAB---122B1F110>