Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Faculty of Sciences and Engineering (FSE)
Semester: (Spring, Year: 2025), B.Sc. in CSE (Day)

ARFF Tree Explorer

Course Title: Artificial Intelligence Lab
Course Code: CSE 316
Section: 222-D2

Students Details

Name	ID
Md. Zehadul Islam	222902069
Md Abdullah Al Moin	222902070

Submission Date: 14.04.2025
Course Teacher's Name: Md. Abu Rumman Refat

[For teachers use only: Don't write anything inside this box]

Lab Project Status

Marks:	**Signature:**
Comments:	**Date:**

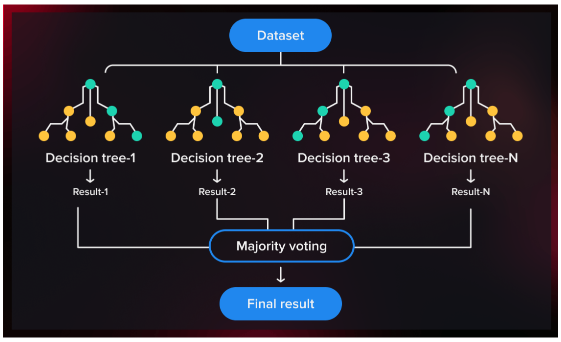
Contents

Chapter 1

Introduction

1.1 Overview

This project **"ARFF Tree Explorer"**focuses on building an interactive desktop-based application for visualizing machine learning models using ARFF (Attribute-Relation File Format) data. This application is built entirely in Python, using technologies such as Tkinter for the GUI, Pandas for data handling, and Scikit-learn for model training and visualization. It allows users to upload ARFF files and visualize various decision-making models like J48 (Decision Tree), Random Forest, REPTree, Decision Stump, and Logistic Model Tree through clear and interactive tree diagrams. The tool supports data preprocessing (including encoding categorical features), model training, and tree visualization using Matplotlib. The goal is to provide a user-friendly interface for educational, analytical, or research purposes where users can explore how decision trees are formed from ARFF datasets. [**?**] [1] [2]



1.2 Motivation

With the increasing use of machine learning and data mining techniques in education and research, there is a need for tools that make it easier to understand complex models. While platforms like Weka provide GUI-based access to ARFF files, many users prefer open-source, code-based environments. The ARFF Tree Explorer was developed to bridge this gap — offering a lightweight, Python-only tool that lets users quickly load

ARFF files, train decision models, and visualize the results in an intuitive manner. This project is especially useful for students, educators, and data science enthusiasts who want to explore classification models visually without switching between platforms.

1.3 Problem Definition

There are many tools for machine learning, but very few of them provide visual interaction with decision tree models using ARFF files in a pure Python environment. Most either require advanced coding or lack graphical representation. The challenge is to create a solution that is both technically robust and easy to use, especially for those working with ARFF files, which are common in Weka and academic datasets.

1.3.1 Problem Statement

The core problem addressed in this project is the lack of simple, open-source tools for visually exploring decision tree models trained on ARFF files. By offering model training, visualization, and support for multiple tree types in a single Python interface, the project aims to provide a practical and accessible solution for both learning and analysis.

1.3.2 Complex Engineering Problem

To further elaborate, Table 1.1 summarizes the attributes of the complex engineering problem addressed by this project:

Name of the P	Attributes	Explanation of how to address
P1: Depth of knowledge required	Machine learning, Python GUI, ARFF format	Understanding how to preprocess data, train ML models, and visualize trees in Python
P2: Range of conflicting requirements	Simplicity vs. functionality	Designing an interface that is simple yet supports multiple ML models and visualizations
P3: Depth of analysis required	Tree structure understanding	Ensuring that users can explore tree depth, splitting criteria, and node decisions interactively.
P4: Familiarity of issues	GUI design, data handling	Handling ARFF parsing errors, encoding, and GUI responsiveness
P5: Extent of applicable codes	Use of open-source libraries	Ensuring that all components use reliable Python libraries (like sklearn, pandas, matplotlib)
P6: Extent of stakeholder involvement and conflicting requirements	Educators, learners, analysts	Gathering feedback from students and researchers for usability improvements
P7: Interdependence	Integration of multiple components	Linking GUI, data loading, model training, and visual output into a cohesive app

Table 1.1: Attributes of the complex engineering problem

1.4 Design Goals/Objectives

The main goal of the ARFF Tree Explorer is to provide an effective and interactive way to explore decision-making models from ARFF files in Python. Below are the key design objectives:

- **Objectives Of My Project Are :**

User-Friendly Interface:

- Provide a clean and intuitive interface using Tkinter
- Allow easy file uploading and model selection

Model Training and Visualization:

- Train models like J48, REPTree, Random Forest, Stump, and LMT
- Visualize decision trees using Matplotlib.

Data Processing:

- Handle categorical data using Label Encoding.
- Ensure robustness in ARFF data parsing.

Modularity and Scalability:

- Structure the code for easy addition of more models.
- Support multiple trees (e.g., from Random Forest) with separate visualizations.

Educational Utility:

- Serve as a learning tool for students to understand how decision trees work.
- Make ML more accessible without coding in depth

Platform Independence:

- Ensure the application runs on any OS with Python installed.
- Use only Python-native libraries.

1.5 Application

The ARFF Tree Explorer has a variety of practical applications:

- **Educational Use:**
  - Teach students how decision trees are formed based on real datasets.
- **Model Debugging:**
  - Understand how models make decisions by visually inspecting splits and nodes.
- **Research and Prototyping:**
  - Quickly test ARFF datasets with various models before deeper experiments.
- **Weka Alternative:**
  - Use as a lightweight alternative to Weka for basic decision tree analysis in Python.
- **Data Visualization:**
  - Present decision logic visually for reports, presentations, or academic writing.
- **Learning Tool:**
  - Help new data science learners get comfortable with classification models and tree structures.

References

[1] Tkinter gui programming. https://docs.python.org/3/library/tk.html.

[2] Scikit-learn documentation. https://scikit-learn.org/.