Homework 2: Decision Trees, Logistic Regression, and Support Vector Machines (100 points)

Prof. Amine Aboussalah

Due Date: Monday 3rd November 2025.
Submission will be done on Brightspace.

Datasets

- **Decision Trees:** UCI Car Evaluation Dataset
- **Logistic Regression**: UCI Breast Cancer Wisconsin Dataset
- **Support Vector Machines**: UCI Breast Cancer Wisconsin Dataset

Overview

In this homework, you'll explore support vector machines, decision trees, and logistic regression and apply them to real-world classification tasks.

You must **implement all algorithms from scratch** without using Scikit-learn or similar libraries. If your implementation runs slowly, you are permitted to reduce the number of datapoints or/and features used from the provided datasets to ensure reasonable execution time, **but you have to mention it**.

Task 1: Decision Tree Classifier (30 points)

Objective

Implement a decision tree classifier from scratch using information gain and Gini impurity criteria.

Instructions

Implementation (20 points)

- Implement the decision tree algorithm with support for:
    - Gini impurity and information gain criteria (5 points)
    - Handling both categorical and numerical features (5 points)
    - Recursive tree construction with depth and minimum split size constraints (5 points)
    - Prediction on test samples (5 points)

Application & Evaluation (10 points)

- Load and preprocess the **Car Evaluation dataset** (e.g., convert categorical features to numerical form) (2 points)
- Split the dataset into 80% training and 20% test (2 points)
- Train your decision tree with different criteria (gini vs information gain) (2 points)
- Evaluate performance using:
    - Accuracy, precision, recall, F1-score (1 point)
    - Confusion matrix (1 point)
- Compare your implementation to scikit-learn's built-in tree classifier. Analyze and interpret the obtained results. (2 points)

Task 2: Logistic Regression (30 points)

Objective

Implement logistic regression from scratch using gradient descent, and apply it to a binary classification problem.

Instructions

Implementation (20 points)

- Implement logistic regression for binary classification using sigmoid activation. (8 points)

- Add support for:
    - Mini-batch gradient descent (4points)
    - L2 regularization (4 points)
    - Early stopping based on validation loss (4 points)

Application & Evaluation (10 points)

- Load and preprocess the UCI Breast Cancer Wisconsin Dataset (e.g., handle missing values, normalize) (2 points)
- Use 70/15/15 train/validation/test split (2 points)
- Evaluate with accuracy, precision, recall, F1-score, and ROC-AUC (3 points)
- Compare your model to `scikit-learn`'s `LogisticRegression` (3 points)

Task 3: Support Vector Machine Implementation (40 points)

Objective

Implement an SVM classifier from scratch and apply it to a binary classification problem.

Instructions

1. **Implementation** (20 points)
    - Code the SVM algorithm from scratch without using existing SVM libraries
    - Your implementation must include:
        - Core SVM algorithm design (8 points)
        - Linear kernel implementation (4 points)
        - Support for at least one non-linear kernel (RBF preferred) (4 points)
        - Optimization using Scipy library and prediction functionality (4 points)
2. **Application & Evaluation** (12 points)
    - Apply your SVM to the provided binary classification dataset (3 points)
    - Split the data into training (70%) and test (30%) sets (2 points)

    - Train your SVM with different hyperparameters: kernel type (e.g. linear, polynomial, RBF …) , kernel parameters (e.g. variance in the RBF kernel) (3 points)
    - Evaluate performance using: (4 points)
        - Accuracy, precision, recall, F1-score
        - ROC curve and AUC
        - Confusion matrix

3. **Analysis** (8 points)

    - Analyze how different hyperparameters affect decision boundaries (3 points)
    - Visualize support vectors and decision boundaries for a 2D projection of your data (3 points)
    - Compare your implementation with scikit-learn's SVM (performance and efficiency), and previous logistic regression results (2 points)

General Requirements (up to -10 points penalty)

- Submit as a **Jupyter notebook**

- All code must be original

- Document each step and algorithm clearly

- Code must include comments and basic error handling

- Clearly report and explain all hyperparameter choices