Pattern Recognition: Project Assignment 1

Due date: 2025.4.30

Introduction

In our class, we have learned boosting algorithms. Boosting is an ensemble learning technique that combines multiple weak learners (models slightly better than random guessing) to form a strong learner with improved accuracy. The core idea is to sequentially train models, where each new model focuses on correcting the errors of its predecessors by adjusting sample weights.

This assignment focuses on exploring boosting algorithms, specifically **Gradient Boost**, using the breast cancer dataset. Its designed to help you understand how these methods combine simple models to make better predictions, particularly for medical diagnosis tasks like classifying tumors as benign or malignant. We will use the Wisconsin Diagnostic Breast Cancer (WDBC) dataset, which has 569 samples and 30 features, making it ideal for binary classification. You can access it from the UCI Repository or Kaggle, both of which are trusted sources for machine learning datasets. Also, in our homework 2 we have already used this dataset, you can directly reuse the dataset loading code.

Note that you **cannot** use any AI, (GPT, deepseek, etc), to help you with coding. Otherwise you will be recorded zero score for this assignment.

Implementation of Weak Learners (20 pts)

You are required to first implement and evaluate the following weak learners:

- **Linear Classifier**: A simple model with a linear decision boundary.
- **Two-Layer Decision Tree (Stump)**: A decision tree that has exactly two sequential splits (depth = 2).

Evaluate each weak learner individually on the dataset. Report and analyze their performance accuracy. They will serve as the weak learners in boosting algorithm.

1 Boosting Algorithms

In this section, you will implement Gradient Boosting. Gradient Boosting builds an additive model in a forward stage-wise fashion. It generalizes boosting to arbitrary differentiable loss functions. The idea is to fit a new learner to the negative gradient (i.e., the pseudo-residual) of the loss function of the whole ensemble. The classifier function is denoted as $F_m(x)$, where m denotes the number of iterations.

The loss function $L(y_i, F(x_i))$ can be cross entropy or average square loss.

Algorithm 1 Gradient Boosting Algorithm

1: **Input:** Training data $\{(x_i, y_i)\}_{i=1}^N$, loss function L, number of iterations M
2: Initialize $F_0(x)$ (e.g., as a constant)
3: **for** $m = 1$ to M **do**
4: **for** $i = 1$ to N **do**
5: Compute pseudo-residual: $r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$
6: **end for**
7: Fit weak learner $h_m(x)$ to $\{(x_i, r_{im})\}_{i=1}^N$
8: Compute $\gamma_m = \arg\min_\gamma \sum_{i=1}^N L\left(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)\right)$
9: Update $F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$
10: **end for**
11: **Output:** $F_M(x)$

Implement Boosting Algorithms (40 pt)

In this section, you need complete following tasks

1. Implement the Gradient Boosting algorithms using the weak classifiers developed in last question. Specify the M you use.
2. Experiment by varying the number of learners (iterations) and report:
   - The classification accuracy on a held-out test set.
   - The evolution of the decision boundary as the number of learners increases (plot the decision boundary for different iterations in several most related feature dimensions). How many weak classifier does it take to saturated?

You may use the `main.py` for implementing. But you can also write your own code.

Analysis and Ablation (40 pt)

In this section, you will investigate the details of the algorithm and understand it. Finish the following tasks or answer the questions.

- Plot the γ_m of learning procedure for linear and decision tree. What is the trend?
- How do you find the γ_m at each step?
- What is the intuition of step 7 in algorithm 1? Is $r_{i,m}$ monotonically changing for each i over step m? Why?

Bonus (20 pt)

Choose one to finish

- Implement any more advanced algorithm, i.e. XGBoost algorithm and compare the final result.
- Read paper: `https://arxiv.org/pdf/1206.6451` (greedy-miser). Use boosting algorithm as good feature selection and compare with LASSO.