CSCI 145 Problem Set 8

October 23, 2025

Submission Instructions

Please upload *your* work by **11:59pm Monday October 27, 2025.**

- You are encouraged to discuss ideas and work with your classmates. However, you **must acknowledge** your collaborators at the top of each solution on which you collaborated with others and you **must write** your solutions independently.
- Your solutions to theory questions must be written legibly, or typeset in LaTeX or markdown. If you would like to use LaTeX, you can import the source of this document (available from the course webpage) to Overleaf.
- I recommend that you write your solutions to coding questions in a Jupyter notebook using Google Colab.
- You should submit your solutions as a **single PDF** via the assignment on Gradescope.

Grading: The point of the problem set is for *you* to learn. To this end, I hope to disincentivize the use of LLMs by **not** grading your work for correctness. Instead, you will grade your own work by comparing it to my solutions. This self-grade is due the Friday *after* the problem set is due, also on Gradescope.

Problem 1: Building Decision Trees

Consider labeled binary classification data $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$ with d real features i.e., $\mathbf{x}^{(i)} \in \mathbb{R}^d$, and binary labels $y^{(i)} \in \{-1, 1\}$. On a given subset of data $S \subseteq \{1, \ldots, n\}$, the loss of a prediction $\hat{y} \in \{-1, 1\}$ is given by

$$\mathcal{L}(S, \hat{y}) = \sum_{i \in S} \mathbb{1}[y^{(i)} \neq \hat{y}].$$

Part A: Choosing the Best Split

Suppose we are at a node in the decision tree, with data points S. Fix a feature $j \in \{1, \ldots, d\}$. We are attempting to find the best threshold τ to *split* the node into two leaves. The left leaf will consist of points

$$S_L(\tau) = \{i \in S : x_j^{(i)} < \tau\}$$

and the right leaf will consist of all points

$$S_R(\tau) = \{i \in S : x_j^{(i)} \geq \tau\}.$$

Describe an *efficient* algorithm to find the optimal split τ, and predictions $\hat{y}_L$ and $\hat{y}_R$, that minimizes

$$\sum_{i \in S_L} \mathcal{L}(S_L, \hat{y}_L) + \sum_{i \in S_R} \mathcal{L}(S_R, \hat{y}_R).$$

Hint: The complexity of your algorithm should be $O(m \log m)$, where $m = |S|$.

Part B: Recursion

Incorporating your solution to Part A, write pseudocode to build a depth h (for height) decision tree. Your algorithm should *recursively* search for the best split, until reaching depth d.

Problem 2: Gradient Boosting

Part A: Optimal Step Size

Consider the gradient boosting setup from class, with the mean-squared error loss function. Suppose we have already found a weak learner f_t. Derive the optimal step size α_t.

Part B: Implementation

Load a regression dataset of your choice. Using the `sklearn` function `DecisionTreeRegressor`, write your own class to build an ensemble of T gradient boosted trees, each with max depth h. Your class should support a `fit` method and a `predict` method.

Part C: Comparison to `xgboost`

Load the `xgboost` regressor, and compare it to yours in terms of a) train MSE, b) test MSE, and c) time to fit.