Summary:

This week introduced tree-based methods for regression and classification, focusing on how decision trees can segment the predictor space to make simple yet powerful predictions. We also explored how ensemble methods like bagging, random forests, and boosting build upon trees to greatly improve predictive accuracy. While basic decision trees offer interpretability, ensemble approaches sacrifice some transparency in favor of substantial performance gains.

Concepts:

- Decision Trees:
    - Partition the feature space using recursive binary splitting.
    - Can be used for both regression (minimize RSS) and classification (maximize purity).
    - Trees are prone to overfitting; hence, cost-complexity pruning is used to simplify them.
    - Cross-validation helps determine the optimal complexity (using tuning parameter α\alphaα).
- Ensemble Methods:
    - Bagging (Bootstrap Aggregating):
        - Builds multiple trees on bootstrapped samples.
        - Reduces variance and helps stabilize predictions.
        - Out-of-bag (OOB) error provides an internal estimate of prediction error.
    - Random Forests:
        - Extends bagging by randomly selecting a subset of predictors at each split.
        - Helps de-correlate trees, further improving performance.
        - Provides variable importance measures, which identify influential features based on impurity reduction.

- Boosting:
    - Builds trees sequentially, each focusing on the residuals of the prior model.
    - Controlled by tuning parameters: number of trees BBB, learning rate λ\lambdaλ, and tree depth ddd.
    - Performs well in practice but requires careful tuning to avoid overfitting.

Uncertainties:

I understand how boosting improves performance by correcting previous errors, but I'm still unclear on how to choose an optimal combination of learning rate λ\lambdaλ, number of trees BBB, and interaction depth ddd in practice. How do I balance these trade-offs efficiently—especially on large datasets where tuning is computationally intensive?