Binary Classification with Trees and Random Forests

Teodora Taleska

Introduction

This project aims to build and test classification trees and random forests from scratch using the TKI resistance FTIR spectral dataset. The dataset includes spectral measurements at various wavelengths for different cell lines, with a binary target for classification.

Part 1: Misclassification Rates and Uncertainty

Methodology

Classification Trees

In this project, **classification trees** were built by splitting data to minimize **Gini impurity**, which measures the chance of misclassifying a random element if labeled based on the subset's class distribution[1].

Random Forests

We also implemented **random forests**, an ensemble method that reduces prediction variance by averaging the outputs of multiple decision trees. Each tree is trained on a random subset of the data (using **bagging**)[1].

Quantifying Uncertainty

To quantify uncertainty, we compute the **misclassification rate (MR)** and its **standard error**. The MR measures the proportion of incorrectly classified samples[1], while the standard error estimates the uncertainty of this rate. The MR is calculated as:

$$\text{MR} = 1 - \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(y_i = \hat{y}_i),$$

where y_i is the true label, $\hat{y}_i$ is the predicted label, N is the total number of samples, and $\mathbb{I}(\cdot)$ is the indicator function. The standard error is given by Standard Error $= \frac{\sigma}{\sqrt{N}}$, where σ is the standard deviation of misclassification errors. Together, these metrics assess both model performance and the reliability of the estimate. A low MR with a small standard error indicates high accuracy and confidence, while a low MR with a large standard error suggests variability and less reliability.

Results

Decision Tree Misclassification Rates

The decision tree model achieved a perfect **training misclassification rate of** 0.0 ± 0.0. However, on the test set, the misclassification rate increased to $\mathbf{0.367 \pm 0.062}$, suggesting overfitting. The relatively high standard error on the test set indicates some variability in the model's performance on unseen data.

Random Forest Misclassification Rates

The random forest model, built with **100 trees**, also achieved a perfect **training misclassification rate of** 0.0 ± 0.0. However, on the test set, the random forest performed significantly better, with a misclassification rate of $\mathbf{0.217 \pm 0.053}$. This demonstrates that the random forest generalizes better to unseen data compared to the decision tree, reducing overfitting and improving test performance. The lower standard error indicates more consistent performance across test samples.

Misclassification vs. Number of Trees

Figure 1 illustrates how the misclassification rate on the test set evolves as the number of trees in the random forest grows. The shaded area represents the standard error, indicating uncertainty.

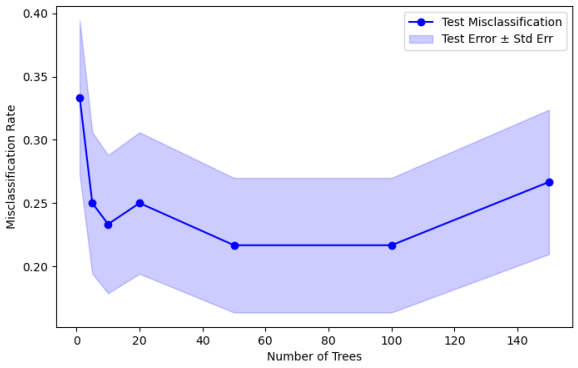

Figure 1. Misclassification Rate vs. Number of Trees in Random Forest

Key observations include: the misclassification rate decreases initially, reaching its lowest point at around **50 trees**.

Beyond 50 trees, the rate stabilizes, showing no notable improvement. The standard error remains steady, suggesting consistent performance once the forest is sufficiently large.

Part 2: Variable Importance Analysis

Methodology

Permutation-Based Feature Importance

We measured variable importance in random forests using out-of-bag (OOB) samples. For each tree, OOB samples (data not used in training) are passed through the tree to record prediction accuracy. To assess a variable's importance, its values in the OOB samples are permuted, and the accuracy is recalculated. The average decrease in accuracy across all trees, caused by this permutation, serves as the variable's importance measure. This approach simulates removing the variable's contribution, offering a reliable estimate of its predictive strength[2].

Results

Variable Importance and Root Frequency

To analyze the importance of features in the dataset, we computed the variable importance for a random forest with $n = 100$ trees. For comparison, we also calculated the frequency of features appearing as the root split in 100 non-random trees trained on randomized data.

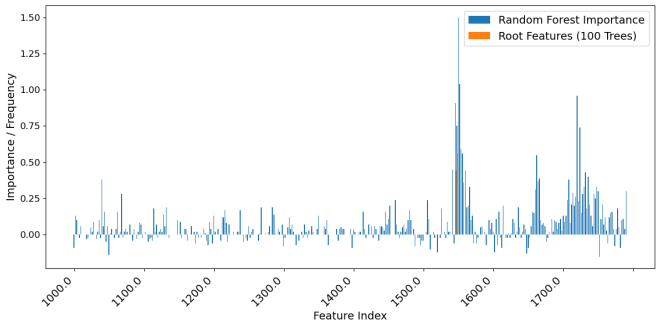

Figure 2. Variable Importance (Random Forest) vs. Root Frequency (Non-Random Trees)

Figure 2 reveals that only two features appeared as the root split in non-random trees:

- **Feature 1548.0**: Ranked 5th in importance by the random forest, it appeared as the root split in 56% of non-random trees.
- **Feature 1546.0**: Ranked 4th in importance by the random forest, it appeared as the root split in 44% of non-random trees.

Part 3: Extended Variable Importance

Extended Variable Importance to Combinations of Three Variables

To extend variable importance to combinations of three variables, we implemented the **importance3()** method. This approach permutes three features simultaneously and measures the drop in prediction accuracy, capturing critical feature interactions. We applied this method to a random forest with n=1000 trees and compared it to the single-feature importance method by comparing the performance of a tree on the best three features identified by each method.

Tree Structure-Based Importance

In scenarios where the original data is unavailable but a pre-built forest exists, we implemented the **importance3_structure()** method. This method identifies the best combination of three variables by analyzing the structure of the trees in the forest:

- For each tree, we extract the features used in its splits and count how often each combination of three features appears together.
- The combination with the highest frequency across all trees is selected as the most important.

Comparison of Classification Trees

Table 1. Classification Tree Performance Using Top 3 Features Identified by Different Methods

Importance Method	Top 3 Features	Accuracy
Single-Feature	['1548', '1552', '1550']	58.33%
3-Feature Combination	['1660', '1550', '1558']	63.33%
Tree Structure-Based	['1784', '1546', '1554']	63.33%

Why the Tree Structure-Based Method Works Well

The tree structure-based method works effectively because it analyzes all splits in the random forest to extract feature usage, even without access to the original data. By tracking the frequency of 3-feature combinations across the forest, it identifies the most frequently used combinations, which are likely to capture important patterns and partition the data effectively. Features frequently used in splits are those that provide the most information gain or Gini impurity reduction, indicating their importance. Counting frequencies at all levels, not just high-level splits, ensures a comprehensive understanding of the data. High-level splits capture global patterns, while lower-level splits refine these with more detailed, localized interactions. Ignoring lower levels risks missing critical feature combinations that only become apparent deeper in the trees. Additionally, evaluating all levels ensures robustness to variability across trees, avoids bias, and balances global and local importance, leading to more accurate feature selection and data partitioning.

References

[1] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[2] Trevor Hastie, Robert Tibshirani, Jerome Friedma. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.*