ML Quarter 2 Project

Depth-Adaptive RF

Edward Nguyen, Aneeshwar Peri, Ritwik Sinha

January 20, 2026

How do we add more to RF?

Random Forest is an ensemble learning algorithm that works with Decision Trees to aggregate a decision, usually using a user-given value to prune all trees at a specific point.

→ **Our project aims to combat this constant value by pruning decision trees in Random Forest by an information-based threshold that varies with node depth instead of this value.**

This will generalize to data better because it will not use as much computation for simpler areas of a dataset while focusing more on complex areas of that set.

Edward Nguyen, Aneeshwar Peri, Ritwik Sinha

The problem

Many datasets can have complex data that can make Random Forests not generalize well in the context of the entire dataset.

Random forests have set pruning values that may gather too much info for simple data and too little for complex data.

Edward Nguyen, Aneeshwar Peri, Ritwik Sinha

Our solution

We will use an information-based threshold rather than a fixed maximum depth. Therefore, a given Decision Tree in a Random Forest will be pre-pruned when the new information it gains is too low to be considered valuable.

Related Works

- Decision Trees & Random Forests achieve strong performance but overfit when trees grow too deep
- Pruning techniques, often based on the Minimum Description Length (MDL) principle, reduce model complexity (Grünwald & Roos, 2019).
- Prior MDL-based methods:
  - MEPSI prunes ensembles after training (Bi et al., 2020)
  - Optimal pruning is computationally expensive, requiring heuristics (Harviainen et al., 2022)
- Other RF adaptations target concept drift or class imbalance, not depth control (Gomes et al., 2017; Mostafa & Khan, 2023; Agusta & Adiwijaya, 2018)
- Our work: depth-adaptive pre-pruning in Random Forests, inspired by MDL, that controls tree growth during training

Our dataset

From the UCI Machine Learning Repository, we chose the Cleveland Heart disease dataset. There were multiple locations, but we chose Cleveland, OH to limit the amount of data to test. Random Forest does not require discretization, so we did not discretize it.

→ 70-30 training-testing split using Stratified Random Sampling

Instances **297**

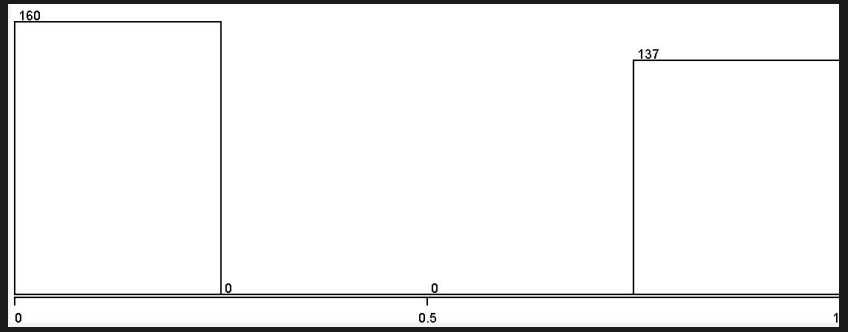
Attributes **14**

No.	1: age Numeric	2: sex Numeric	3: cp Numeric	4: trestbps Numeric	5: chol Numeric	6: fbs Numeric	7: restecg Numeric	8: thalach Numeric	9: exang Numeric	10: oldpeak Numeric	11: slope Numeric	12: ca Numeric	13: thal Numeric	14: **target** Numeric
1	63.0	1.0	1.0	145.0	233.0	1.0	2.0	150.0	0.0	2.3	3.0	0.0	6.0	0.0
2	67.0	1.0	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	2.0	3.0	3.0	1.0
3	67.0	1.0	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	2.0	2.0	7.0	1.0
4	37.0	1.0	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	3.0	0.0	3.0	0.0
5	41.0	0.0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	1.0	0.0	3.0	0.0
6	56.0	1.0	2.0	120.0	236.0	0.0	0.0	178.0	0.0	0.8	1.0	0.0	3.0	0.0
7	62.0	0.0	4.0	140.0	268.0	0.0	2.0	160.0	0.0	3.6	3.0	2.0	3.0	1.0
8	57.0	0.0	4.0	120.0	354.0	0.0	0.0	163.0	1.0	0.6	1.0	0.0	3.0	0.0
9	63.0	1.0	4.0	130.0	254.0	0.0	2.0	147.0	0.0	1.4	2.0	1.0	7.0	1.0
10	53.0	1.0	4.0	140.0	203.0	1.0	2.0	155.0	1.0	3.1	3.0	0.0	7.0	1.0

Edward Nguyen, Aneeshwar Peri, Ritwik Sinha

Our dataset: Class Structure

The class distribution was relatively balanced, with 160 instances labeled as no heart disease and 137 instances labeled as having signs of heart disease. Due to this mild imbalance, no class reweighting or resampling techniques were required



Our dataset: Attributes & Preprocessing

Note: no other attributes had missing values

“thal”

- Measures the outcome of a stress test that measures how blood flows to the heart
- Range: 3, 6 or 7
- Had missing values

“ca”

- Measures the number of major blood vessels being blocked via a heart imaging test.
- Range: 0 to 3
- Had missing values

Class attribute

- Measured severity of heart disease
- Range: 0 to 4
- We decided to discretize it into a binary class:
  0 (healthy)
  1 (had signs of heart disease) [1-4]

How is ours different?

Entropy is defined as $H(D) = -\sum_k p_k \log_2 p_k$

A candidate split using InfoGain is given by

$$\text{InfoGain(s)} = H_{\text{parent}}(D) - \sum_c \frac{|D_c|}{|D|} H(D_c)$$

In fixed pre-pruning, a split is only done when InfoGain exceeds some threshold ϵ_0 .

Using the *Minimum Description Length* principle, we instead balance model complexity against misclassification error. Our goal is to minimize the Total Description Length: $TDL = L(M) + L(D|M)$ where L(M) represents the cost of encoding the model and L(D|M) represents the cost of encoding data.

As the tree gets deeper, L(M) increases while L(D|M) decreases, as there is less information to gain from splitting further.

→ A split is beneficial if and only if the decrease in L(D|M) outweighs the increase in L(M).

How is ours different?

$$TDL = L(M) + L(D|M)$$

L(D|M) can be approximated by using InfoGain, but we will now modify it to be depth adaptive:

$$\text{InfoGain} \geq \epsilon(d); \text{where } \epsilon(d) = \epsilon_0 \cdot e^{-\alpha d}$$

where ϵ_0 represents the initial threshold at the root node, and thus α is able to control the splitting threshold, and d is the depth of the split.

This model of InfoGain prioritizes splitting closer to the root, where splits are able to greatly change model structure.

By using this adaptive method for all Decision Trees in the Random Forest, we aim to improve bias-variance trade offs, preventing unnecessary early branching while preserving expressive power in complex regions of the feature space.

Edward Nguyen, Aneeshwar Peri, Ritwik Sinha

Hyperparameter Selection

Number of Trees

- 75 trees
- Increasing the number of trees stabilizes predictions and will help compensate for the stronger pre-pruning applied to individual trees

Decay Rate

- Our decay rate is 0.15
  - This decreases the information gain threshold with increasing depth, allowing for more growth in more complex areas

Feature Subsampling

- At each split, a subset of sqrt(d) randomly selected features is considered, where d is the total number of features
- This subsampling allows for diversity among trees and reduces correlation between them.

Model Training and Prediction

For our baseline random forest and depth-adaptive random forest models:

- Training: The model was trained on the 208 training instances using bootstrap sampling for each tree.
- Prediction: The ensemble prediction was obtained by majority voting across all decision trees in the forest.
- Random Seed: A fixed random state of 42 was used to ensure reproducibility.

Baseline Random Forest

```
|--- ca <= 0.50
|   |--- thal <= 4.50
|   |   |--- cp <= 3.50
|   |   |   |--- cp <= 1.50
|   |   |   |   |--- class: 0.0
|   |   |   |--- cp >  1.50
|   |   |   |   |--- class: 0.0
|   |   |--- cp >  3.50
|   |   |   |--- sex <= 0.50
|   |   |   |   |--- chol <= 315.00
|   |   |   |   |   |--- class: 0.0
|   |   |   |   |--- chol >  315.00
|   |   |   |   |   |--- class: 0.0
|   |   |   |--- sex >  0.50
|   |   |   |   |--- class: 0.0
|   |--- thal >  4.50
|   |   |--- exang <= 0.50
|   |   |   |--- chol <= 293.00
|   |   |   |   |--- chol <= 204.50
|   |   |   |   |   |--- class: 0.0
|   |   |   |   |--- chol >  204.50
|   |   |   |   |   |--- slope <= 2.50
|   |   |   |   |   |   |--- age <= 51.00
|   |   |   |   |   |   |   |--- class: 1.0
|   |   |   |   |   |   |--- age >  51.00
|   |   |   |   |   |   |   |--- age <= 57.50
|   |   |   |   |   |   |   |   |--- class: 0.0
|   |   |   |   |   |   |   |--- age >  57.50
|   |   |   |   |   |   |   |   |--- class: 1.0
|   |   |   |   |   |--- slope >  2.50
|   |   |   |   |   |   |--- class: 0.0
|   |   |   |--- chol >  293.00
|   |   |   |   |--- class: 0.0
```

Depth-Adaptive Random Forest

```
if cp <= 3.0000:
  if oldpeak <= 1.1000:
    if thalach <= 152.0000:
      Predict -> 0
    else:  # thalach > 152.0000
      Predict -> 0
  else:  # oldpeak > 1.1000
    if ca <= 0.0000:
      if chol <= 245.0000:
        Predict -> 0
      else:  # chol > 245.0000
        if restecg <= 0.0000:
          if chol <= 309.0000:
            Predict -> 1
          else:  # chol > 309.0000
            Predict -> 0
        else:  # restecg > 0.0000
          Predict -> 0
    else:  # ca > 0.0000
      if chol <= 282.0000:
        if oldpeak <= 1.8000:
          if oldpeak <= 1.4000:
            Predict -> 1
          else:  # oldpeak > 1.4000
            Predict -> 0
        else:  # oldpeak > 1.8000
          Predict -> 1
      else:  # chol > 282.0000
        Predict -> 0
```

Accuracy

Traditional Random Forest

Train Accuracy	96.15%
Test Accuracy	83.15%
Precision	0.853
Recall	0.744
F1-Score	0.795
Average Tree Depth	7.51
Average Node Count	42.07

Edward Nguyen, Aneeshwar Peri, Ritwik Sinha

Accuracy

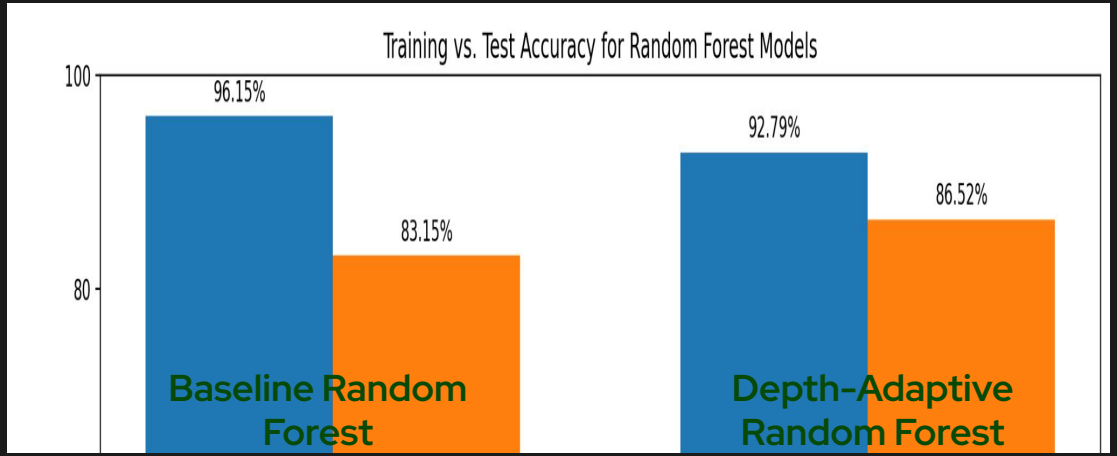
Depth-Adaptive Random Forest

Train Accuracy	92.79%
Test Accuracy	86.52%
Precision	0.935
Recall	0.744
F1-Score	0.829
Average Tree Depth	7.40
Average Node Count	37.83

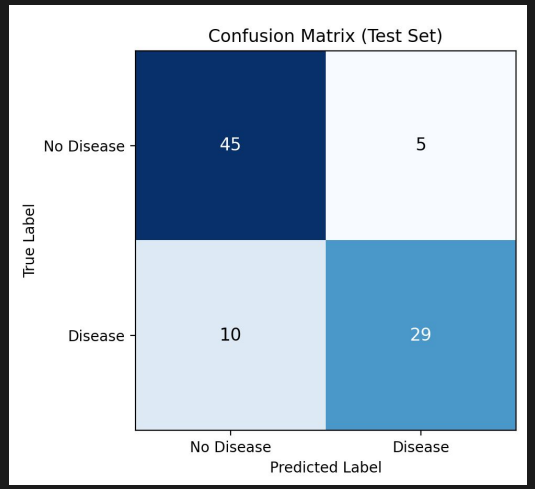
Edward Nguyen, Aneeshwar Peri, Ritwik Sinha

Comparison

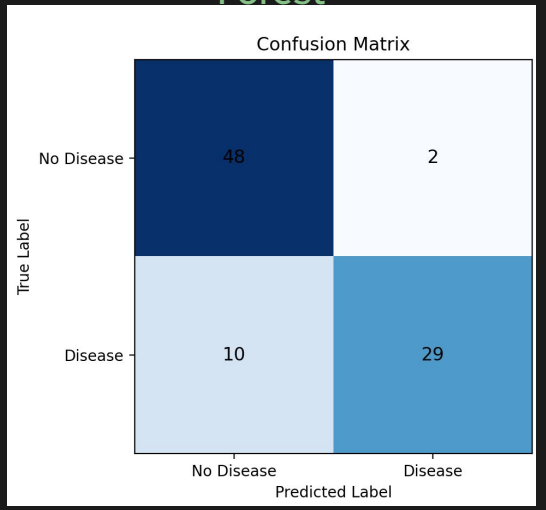
Baseline Random Forest vs.
Depth-Adaptive Random Forest



Baseline Random Forest



Depth-Adaptive Random Forest



Performance Analysis

Accuracy

Training accuracy **decreased** by 3.36%, but testing accuracy **increased** by 3.37%, which signified a better generalization across datasets, which is what we wanted to improve.

Other test metrics

Precision and F1-score **increased** by .082 and .034 respectively, while recall stayed the same.

This is because our model is classifying the same # of true positives, while decreasing # of false positives

Tree statistics

The average depth and average node count for Decision Trees in the Random Forest **decreased**, with a depth decrease of 0.11 and a node decrease of 4.24.

This helped us in achieving our goal of decreasing overfitting in our Random Forest

Thank you!