SANDIA REPORT
SAND2024-13686R
Printed September 2024

Sandia National Laboratories

FORESTR: Finding, Organizing, Representing, Explaining, Summarizing, and Thinning Random forests

Katherine Goode & J. Derek Tucker

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185
Livermore, California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods

ABSTRACT

Random forests have become popular models used for data driven predictions. As a result, random forests are currently used or being considered for high-consequence mission applications in national security, such as the prediction of yield from optical signals and malware detection. While random forests may provide accurate predictions, the complexity of the algorithm causes a lack of interpretability. Random forests are an ensemble of regression or decision trees. Individual regression and decision trees are interpretable, but ensembles are inherently difficult to interpret due to the compilation of many models. We aim to increase the interpretability of random forests by finding patterns in the ensemble of trees that can be used to "thin" (or remove) trees. As a starting point, in this report, we develop a new distance metric for quantifying the similarity between trees based on their topologies (i.e., shapes). We base the metric on a novel distance metric for graphs that is a proper mathematical distance, is invariant to transformations, has registration between graphs, and computes topological evolutions between graphs. We use the tree distance metric to compute tree statistics such as a "mean tree" and to identify clusters of trees. We apply the developed methodology to a toy dataset and a mission relevant product inspection dataset to demonstrate how the metric can provide insight into random forests. Furthermore, we discuss the limitations of the approach and ideas for future research into how the metric could be used as a thinning tool to develop less complex models.

This page intentionally left blank.

CONTENTS

This page intentionally left blank.

LIST OF FIGURES

1. INTRODUCTION

Some machine learning (ML) models are referred to as "black-box" models due to their algorithmic complexity, which obscures the interpretation of their parameters in the application context. Despite this lack of transparency, black-box models exhibit desirable qualities, such as high predictive accuracy. These models are inherently data-driven, which enables them to identify patterns without requiring prior knowledge of the underlying data-generating mechanism. This characteristic makes these models particularly valuable in applications where the scientific principles are not well understood. The advantages of black-box ML models have led to their adoption (and consideration for adoption) in high-stakes decision-making environments, including mission-critical areas at Sandia National Laboratories. Examples include cybersecurity (e.g., the TAMIZAR network intrusion detection system) and nuclear forensics (e.g., the NA-22 device assessment multi-lab effort).

As the interest in using black-box models continues, there is an increasing need for techniques that elucidate how these models utilize input variables to generate predictions. The absence of transparency increases the risks associated with the use of a model in practice. As such, decision-makers may hesitate to rely on models with opaque decision-making processes, especially in high-consequence scenarios. Similarly, model developers may be reluctant to recommend the application of a model when it is not possible to interpret parameters to diagnose the model's credibility. These concerns have led to the research field of explainable ML [16].

In this report, we differentiate between explainability and interpretability. Interpretability focuses on developing models whose parameters have clear meanings within the application context (e.g., understanding a feature split in a tree model). In contrast, explainability allows the model to remain a black-box while employing post-hoc techniques (i.e., techniques applied after model training) to provide insights into how the model makes decisions. For instance, permutation feature importance is an explainability technique that quantifies the change in model predictive performance when an input variable is randomly permuted [6]. These metrics provide evidence of the influence an input variable has on predictions.

A variety of explainability techniques have been proposed, ranging from model-agnostic methods applicable to any model (e.g., permutation feature importance) to model-specific approaches designed specifically for an algorithm (e.g., Gini random forest feature importance [7]). While we do not provide a comprehensive overview of existing explainability methods in this report, we encourage interested readers to consult the extensive literature, including reviews, books, opinion pieces, and cautionary tales. For example, see [1, 2, 15, 17, 18]. Instead, our focus is on a specific area of explainability related to ensembles of tree models.

Ensembles of decision or regression tree models (such as random forest models [4]) are an example of a black-box ML model. Individual tree models are typically considered interpretable models,

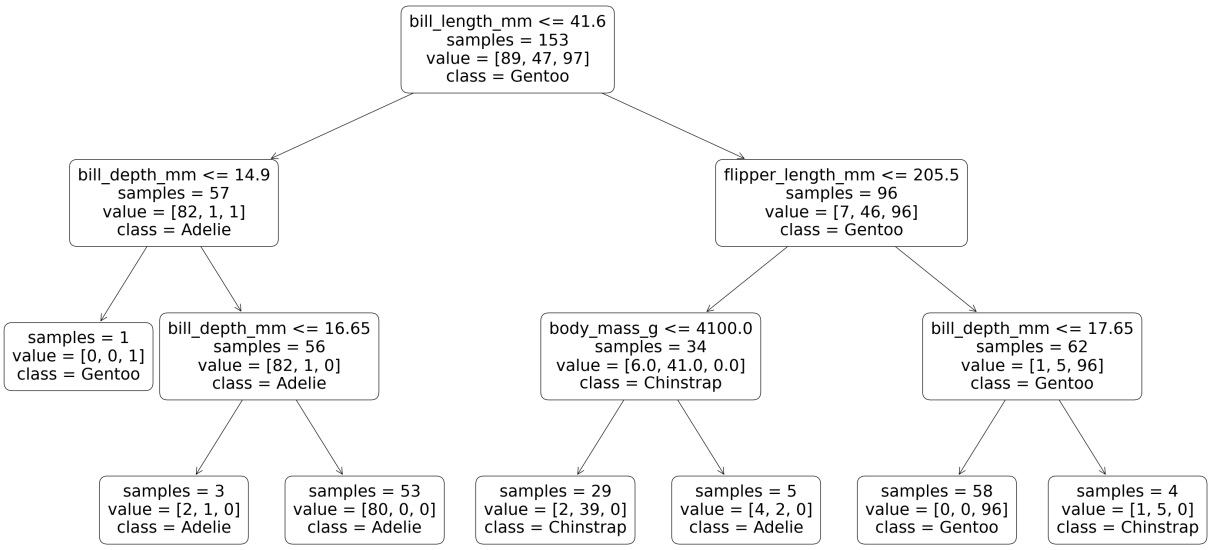

Figure 1-1.: Visualization of a decision tree model.

because it is possible to extract a set of rules from the tree that describe how a prediction is made. For example, Figure 1-1 depicts a single decision tree for predicting the species of a penguin based on body measurements. We could follow the path an observation takes to understand the role the variables play in a classification. That is, we can "interpret" the model "parameters". However, as the number of trees in an ensemble increases, it becomes more challenging to use the model parameters (split variable thresholds) to make sense of how a prediction is made. Figure 1-2 depicts the 50 trees in a random forest. We can interpret each tree separately, but it quickly becomes overwhelming if we try to aggregate the knowledge gained from each tree.

One proposed approach to "explaining" an ensemble of trees is to identify patterns in the trees. Different strategies have been suggested including identifying a representative tree such as a mean tree that captures the "average" decision process used by the forest [3, 19, 25]. Other methods aim to find clusters of trees to capture tree variability [5, 20]. Furthermore, representative trees may be found within each cluster to understand the nature of a cluster. Methods have also been developed for reducing the number of trees in a forest for computational reasons (e.g., [25]).

The majority of these methods use a quantitative metric to measure the distance between trees. There are two main types: (1) prediction and (2) topology distance metrics. Prediction distance metrics may consider

1. whether two trees return similar predictions,
2. whether observations fall into the same partitions for two trees, or
3. a combination of the above.

Tree topology distance metrics consider the structure of trees such as the variables used for splitting, the split threshold, the order of variables used for splitting, and so forth. For a further discussion

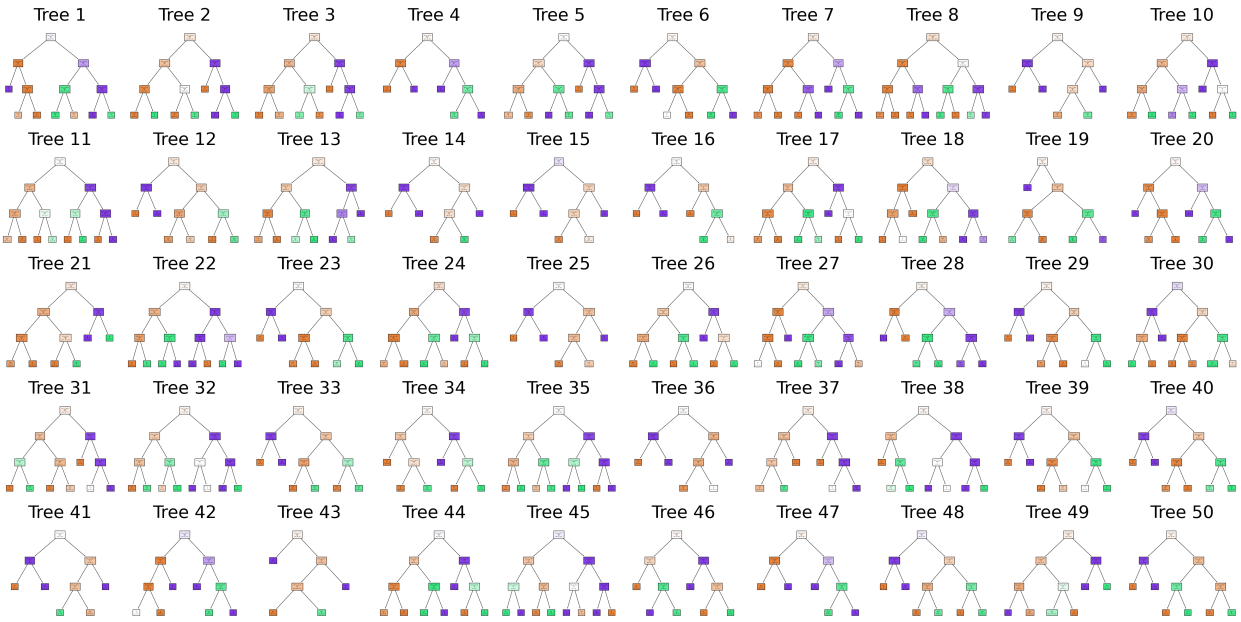

Figure 1-2.: Visualization of 50 decision trees in a random forest model.

of tree distance metrics, see [20].

An example of a prediction distance metric is the *fit metric* [5], which quantifies the average distance between predictions from two trees for a given set of data. That is, let T_t represent tree t with $t \in \{1, 2\}$, and $\hat{y}_{i,t}$ represent the prediction from tree t for observation i with $i \in \{1, ..., n\}$. The fit metric is computed as

$$d_{FM}\left(T_1, T_2\right) = \frac{1}{n}\sum_{i=1}^{n} m\left(\hat{y}_{i,1}, \hat{y}_{i,2}\right),$$

where m may be set to any metric that compares two predictions such that larger values indicate less similarity. For example, in the case of a regression tree, m could be the squared distance between two predictions:

$$m\left(\hat{y}_{i1}, \hat{y}_{i2}\right) = \left(\hat{y}_{i1} - \hat{y}_{i2}\right)^2.$$

In the case of a classification tree, m could be an indicator of prediction disagreement:

$$m\left(\hat{y}_{i1}, \hat{y}_{i2}\right) = \begin{cases} 1 & \text{if } \hat{y}_{i1} \neq \hat{y}_{i2} \\ 0 & \text{o.w.} \end{cases}.$$

The *partition metric* [5] is an example of a prediction distance metric that considers tree partitions by comparing how observations are divided between tree leaves. Specifically, the partition metric is computed as

$$d_{PM}\left(T_1, T_2\right) = \frac{\sum_{i>j} \left|I_1(i,j) - I_2(i,j)\right|}{\binom{n}{2}},$$

where

$$I_t(i,j) = \begin{cases} 1 & \text{if } T_t \text{ places observations } i \text{ and } j \text{ in the same terminal node} \\ 0 & \text{o.w.} \end{cases}.$$

An example of a topology metric is the *covariate metric* [3], which is computed as the proportion of covariate mismatches:

$$d_{CM}(T_1, T_2) = \frac{\sum_{k=1}^{K} I[c_{k,1} \neq c_{k,2}]}{K},$$

where

$$c_{k,t} = \begin{cases} 1 & \text{if variable } k \text{ is used by tree } t \\ 0 & \text{o.w.} \end{cases}.$$

In this report, we propose a new metric for computing distances between trees based on their topologies. We base our metric on a novel distance metric for comparing graph topologies [10], which we adapt for computing distances between trees based on their topologies. An advantage to basing our metric on the graph metric is that the graph metric is a proper mathematical distance. Many previously proposed tree topology distance metrics do not satisfy the mathematical definition of a distance. For example, a proper mathematical distance requires that for two distinct trees, $T_1 \neq T_2$, the distance, d, between the trees must be positive: $d(T_1, T_2) > 0$. In the case of the covariate metric, it is possible for $d(T_1, T_2) = 0$. Therefore, if there is interest in computing summary statistics using the distance metric (e.g., a mean tree), there may be fundamental issues.

Our objective is to provide transparency to random forest models by identifying tree topology patterns using the newly developed distance metric. In this report, we present our work on adapting the distance metric, and we demonstrate the use of the adapted metric to identify topological patterns via methods such as the identification of representative trees and clusters of trees. We use visualizations of the representative trees and clusters to gain insights into common decision paths and tree variability.

The information gained from this work also lays the groundwork for the development of a potential forest thinning method (i.e., removal of trees). The tree metric allows for the identification of outlier trees (e.g., trees with large distances from a representative tree) and trees containing potentially redundant information (e.g., trees separated by a small topological distance). This information could be used to develop algorithms for the removal of trees that would decrease the complexity of the model and lead to a more interpretable model. In contrast, a forest could also be built starting with representative trees from the original forest. In application spaces where model transparency is necessary, it would be valuable to possess an algorithm able to identify a simpler and more interpretable forest while preserving the predictive ability.

This report is outlined as follows. In Section 2 , we provide background details on the graph topology distance metric. Section 3 describes our work on adapting the graph metric to trees and a methodology for using the metric to compute a central tree. Section 4 presents applications of the methodology to two datasets: a toy dataset classifying penguin species and a mission relevant product inspection dataset. Finally, Section 5 contains a summary of our work, key findings, and directions for future research.

2. BACKGROUND

In this section, we provide an exposition of the graph metric employed to assess the similarity among trees within the random forest. For an exhaustive overview, the reader is referred to [9] and [10]. This framework constitutes an extension of prior contributions on the geometric characterization of graphs as documented in [12].

We consider a weighted graph G as an ordered pair (V, e), where V denotes the set of nodes (or vertices) and e represents a weight function: $e : V \times V \to \mathbb{R}^+$. Specifically, $e(v_i, v_j)$ characterizes the edge between nodes $v_i, v_j \in V$, where the elements of set $E = \{(v_i, v_j) \in V \times V : i \neq j\}$ constitute the edges of graph G. Given the cardinality $n = |V|$, the number of nodes in G, the graph can be represented by its adjacency matrix $A = \{a_{ij}\} \in \mathbb{R}^{n\times n}$, with elements $a_{ij} = e(v_i, v_j)$. In the case of an undirected graph G, the adjacency matrix A will exhibit symmetry. However, in this report, our focus on trees in random forests implies that G will be a directed graph, resulting in a non-symmetric A.

The set comprising all such matrices is denoted by $\mathcal{A} = \{A \in S^{n\times n} \mid \text{diag}(A) = \mathbf{0}\}$. Let d represent the Euclidean distance $\mathbb{R}^+$, which we shall employ to establish a metric on $\mathcal{A}$. Consequently, for any two matrices $A_1, A_2 \in \mathcal{A}$ with respective entries a^1_{ij} and a^2_{ij}, the metric is delineated as

$$d_a(A_1, A_2) = \sqrt{\sum_{i,j} d(a^1_{ij}, a^2_{ij})^2}. \tag{2.1}$$

This metric aptly measures the disparity between the two graphs G_1 and G_2 represented by the adjacency matrices A_1 and A_2, respectively.

The arrangement of nodes within graphs is typically arbitrary; thus, it is imperative that our analysis remains invariant under such permutations. Let $\mathcal{P}$ denote the set of all permutation matrices of dimension $n \times n$. It is noteworthy that $\mathcal{P}$ constitutes a subgroup of $O(n)$, the collection of all $n \times n$ orthogonal matrices, and that for any $P \in \mathcal{P}$, the inverse of P is given by P^T. We define the action of $\mathcal{P}$ on $\mathcal{A}$ as follows:

$$\mathcal{P} \times \mathcal{A} \mapsto \mathcal{A}, \ (P, A) \to PAP^T. \tag{2.2}$$

The verification of this scenario as a legitimate group action is straightforward, given that $\mathcal{P}$ constitutes a group. Furthermore, the orbit under this action can be articulated as $[A] = \{PAP^T | P \in \mathcal{P}\}$, encapsulating all potential permutations of the nodes in A. Two graphs, denoted A_1 and A_2, are defined as equivalent if there exists a permutation matrix P such that $PA_1P^T = A_2$. Consequently, the operation of $\mathcal{P}$ on $\mathcal{A}$ adheres to isometries, and with respect to the metric d_a, it holds that

$$d_a(PA_1P^T, PA_2P^T) = d_a(A_1, A_2). \tag{2.3}$$

We can subsequently define a metric within the graph space $\mathcal{G} = \mathcal{A}/\mathcal{P}$ as

$$d_g([A_1], [A_2]) = \min_{P \in \mathcal{P}} d_a(A_1, PA_2P^T). \tag{2.4}$$

The minimizing transformation P^* yields the optimal alignment between the graphs, A_1 and A_2. To address this optimization problem, one may employ the Umeyama Algorithm [22] or Fast Approximate Quadratic Programming [24]. An analogous metric representation can be formulated using the Laplacian matrix, as detailed in [10].

While the aforementioned metric predominantly considers the structural properties of the graph and edge weights, a more comprehensive extension can incorporate both edge weights and node attributes. Let D represent the $n \times n$ matrix of pairwise squared distances between nodes across the two graphs G_1 and G_2,

$$D = [d_{ij} = d(\alpha_1(v_i^1), \alpha_2(v_j^2))^2] \in \mathbb{R}^{n \times n}, \tag{2.5}$$

where α denotes the node attribute. Consequently, the metric transforms into

$$d_g([A_1], [A_2]) = \min_{P \in \mathcal{P}} \|PA_1P^T - A_2\|^2 + \lambda \mathrm{Tr}(PD) \tag{2.6}$$

where Tr indicates the trace operation and $\lambda > 0$ is a tuning parameter. For comprehensive algorithms addressing these optimization problems, refer to [10].

Finally, equipped with a metric on the space of graphs, $\mathcal{G}$, we can compute both the mean and the median of a set of graphs. Given a set of m graphs, denoted as $G_i \in \mathcal{G}, i = 1, \ldots, m$, and their respective adjacency matrices, $A_i \in \mathbb{R}^{n \times n}$, the mean graph is obtained by minimizing the sum of squared distances. This minimization problem is formalized as:

$$[A_\mu] = \arg\min_{A \in \mathbb{R}^{n \times n}} \sum_{i=1}^{m} d_g([A], [A_i])^2. \tag{2.7}$$

The mean graph is guaranteed to be unique, and the optimization is performed using a greedy optimization algorithm as detailed in [10].

3. METHODOLOGY

Our main objective is to provide transparency to random forest models by identifying topology patterns associated with the trees in an ensemble. In order to identify these patterns, we require a metric that computes the distances between the topologies of two trees. In this section, we describe the steps we take to adapt the graph metric described in Section 2 for trees.

3.1. Tree Topology Distances

We begin by letting G_t represent the weighted graph associated with tree $t = 1, ..., T$ in an ensemble of T trees. G_t is composed of ordered pairs, (V_t, e_t), of nodes and edges as described in Section 2. We let A_t represent the adjacency matrix associated with tree t. Recall that A_t will be non-symmetric since trees are directed acyclic graphs (DAG). That is, there is a direction associated with each edge between nodes and no cycles (i.e., no closed loops). Further, it is possible (and likely) that number of trees nodes will vary across the trees in the ensemble. When computing the distances between two trees (T_t and T_u), this is handled by letting $n = \max(\{n_t, n_u\})$, where $n_t = |V_t|$ and $n_u = |V_u|$ are the number of nodes in trees T_t and T_u, respectively. When A_t and A_u are created, rows and columns containing 0's are added to the smaller matrix in order for both to be $n \times n$ matrices. (Note that when computing a mean graph, $n = \max(\{n_1, ..., n_T\})$, and all T adjacency matrices as adjusted similarly.)

We are interested in computing all pairwise distances between the T trees in the ensemble as

$$d_g([A_t], [A_u]) \tag{3.1}$$

for trees $t \neq u$. Note that $d_g([A_t], [A_t]) = 0$. We could directly apply the metric described in Equation 2.4 to compute d_g, but this would not account of the hierarchical structure that trees possess. That is, each node in a tree is associated with a *depth*. Further, there are three types of nodes in a tree model with specific properties: root, internal, and leaf nodes.

- A *root node* falls at the top of the hierarchy. Each tree has one *root node* at a depth of 0. In our set up, we require that root nodes have exactly two edges connected to nodes at a depth of 1.
- *Leaf nodes* have exactly one edge connected to a node at the previous depth and no additional edges. Leaf nodes are the lowest in the hierarchy.
- *Internal nodes* are all nodes that are not the root node or leaf nodes. An internal node with a depth of d must have exactly one edge connected to a node at a depth of $d - 1$ and exactly two edges connected to nodes at a depth of $d + 1$.

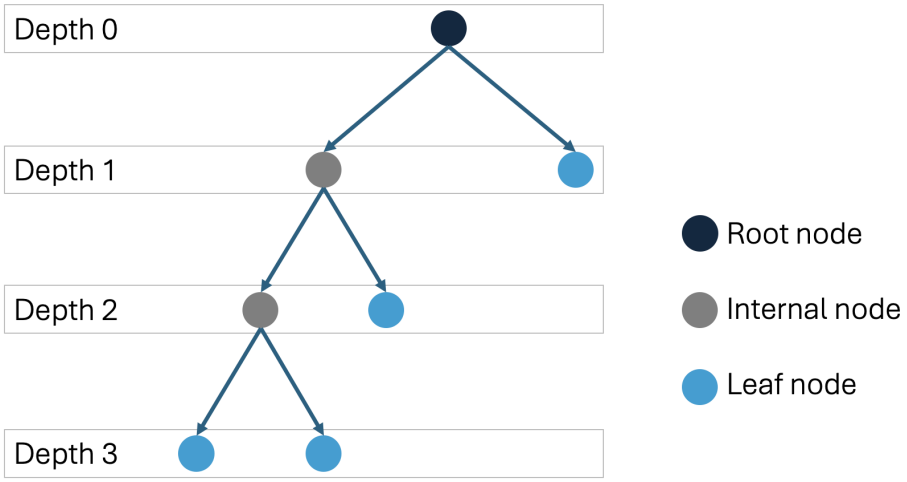

Figure 3-1.: Example depicting the hierarchical structure of a tree model. Arrows represent the edge directions, and colors represent the node types.

The edge direction always goes from the node with the smaller depth to the node with the larger depth. See Figure 3-1 for an example with a tree of maximum depth of 3.

In order to account for the hierarchical structure of trees when computing distances, we draw on Equation 2.6 to allow for incorporation of node attributes. In particular, we set the node depth as the node attribute in the distance computation. Thus, when computing the matrix of pairwise squared distances between nodes across two trees (as described in Equation 2.5), we let $\alpha_t(v_i^t)$ represent the node depth associated with node i in tree t. We compute the distance between node depths as the squared difference. That is, for trees G_t and G_u,

$$D_{t,u} = [d_{ij} = (\alpha_t(v_i^t) - \alpha_u(v_i^u))^2] \in \mathbb{R}^{n \times n}. \tag{3.2}$$

In this report, we only account for node depth in our computation, but in the future, multiple attributes could be included such as the feature used for splitting. This would result in $\alpha_t(v_i^t)$ being a vector of values. Note that if categorical values, such as split feature, were added to $\alpha_t(v_i^t)$, the distance function, d, in Equation 2.5 would need to be adjusted to handle both numeric and categorical values.

Finally, we compute the distance between trees G_t and G_u using Equation 2.6 as

$$d_g([G_t], [G_u]) = \min_{P \in \mathcal{P}} \|PA_tP^T - A_u\|^2 + \lambda \mathrm{Tr}(PD_{t,u}), \tag{3.3}$$

where λ is a tuning parameter that specifies how much weight is placed on the node attributes during the computation. If $\lambda = 0$, the node attributes are excluded from the calculation. For the process of finding the optimal P in Equation 3.3, we use the Umeyama Algorithm throughout our example analyses in this report. However, other optimization algorithms may be used as described in Section 2.

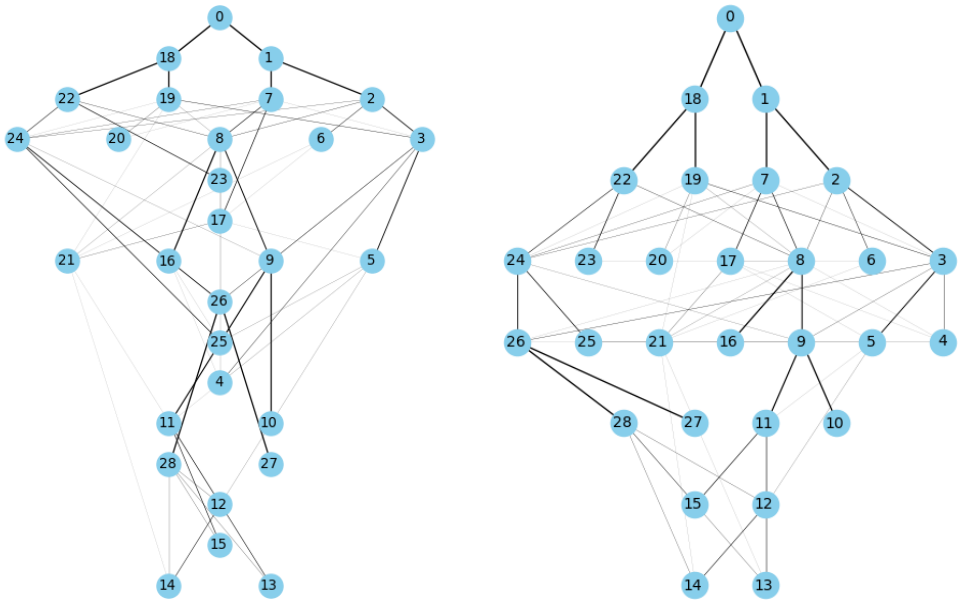

Figure 3-2.: Example mean graphs computed from a set of trees that violate tree properties. (Left) Mean graph before node depth attributes are rounded down to the nearest integer value. (Right) Mean graph after node depths are rounded with resulting edges between nodes with the same node depth.

3.2. Central Tree

After obtaining all pairwise distances between trees, we are able to perform additional computations to identify topology patterns. For example, we can apply clustering algorithms to the distance matrix and compute an average graph. The application of clustering algorithms is a straightforward process. See Section 4 for examples. However, a direct application of the minimization process in Equation 2.7 for computing a mean graph results in violations of the properties of trees. In particular, the process of finding the adjacency matrix, A, that minimizes the sum of squared tree distances may result in node depth attributes that are not integer valued.

Figure 3-2 (left) shows an example of a mean graph computed from a set of trees with non-integer valued node depths, which violates tree properties. Figure 3-2 (right) shows the same graph after rounding the node depths down to the nearest integer. The structure of the graph is more tree like, but now, there are edges between nodes with the same node depth (e.g., nodes 25 and 26). Note that in both graphs, some of the edges are darker than others. The darkness of an edge indicates the weight assigned to the edge during the computation of the mean. This weighting leads to additional tree violations such as nodes with more than one edge connected to a node at a smaller depth (e.g., node 8) and edges between nodes with a difference in node depths that is greater than 1 (e.g., nodes 5 and 12).

Future work could explore how to adjust the mathematical approach for computing the graph mean such that the resulting mean adheres to tree properties, but for now, this is an open question. The solution we use in this report for obtaining a representative tree from a set of trees is to identify

the tree (or trees) with the smallest distance to the mean graph as computed by Equation 2.7. We will refer to a representative tree identified in this manner as a *central tree*. That is, if we let G_μ represent the mean graph, the central tree of a set of trees, $\{G_1, ..., G_T\}$, is computed as

$$G_{central} = \underset{G \in \{G_1,...,G_T\}}{\arg\min} \; d_g(G, G_\mu). \tag{3.4}$$

Another option would be to compute a *medoid tree* as the tree in the set of trees, $\{G_1, ..., G_T\}$, that minimizes the sum of distances:

$$G_{medoid} = \underset{G \in \{G_1,...,G_T\}}{\arg\min} \sum_{t=1}^{T} d_g(G, G_t). \tag{3.5}$$

4. APPLICATIONS

In this section, we present two applications of the methodology described in Section 3. The first application considers the *Palmer penguin data*, which is a simple toy data set (Section 4.1). The second application demonstrates the methodology on a mission-relevant product inspection dataset referred to as *s500* (Section 4.2).

All analyses are implemented using Python code, which is available in the FORESTR GitLab repository on the Sandia CEE GitLab[1]. Our implementation of the tree metric methodology is heavily based on existing code written by the authors of the graph metric [10]. We also developed code for converting tree information from scikit-learn [**?**] random forest models into a pandas DataFrame [14] and visualizing scikit-learn random forest trees using trace plots. While the code is currently implemented only for random forest models, the methodology is applicable to any ensemble of trees.

4.1. Palmer Penguin Data

4.1.1. Data and Models

The Palmer penguin data [11] contain the species (adelie, chinstrap, and gentoo) and four body measurements (bill length, bill depth, flipper length, and body mass) from 342 penguins from the Palmer Archipelago in Antarctica. Figure 4-1 shows a parallel coordinate plot of the penguin data. There are clear relationships in the data between the body measurements and species. For example, gentoo penguins tend to have smaller bill depths and larger flipper lengths and body masses than the other species. We build a random forest model to predict the species of a penguin given their body measurements. We are interested in using the proposed tree metric to identify tree topology patterns in the model.

We randomly split the observations into training and testing data with 100 penguins in the testing data. In order to gain a better understanding of the tree metric, we train two random forests: (1) the *shallow forest* and (2) the *full depth forest*. Both models are trained using the scikit-learn random forest classifier [**?**] with 50 trees per model. For the shallow forest, we set the 'max_depth' parameter to 3 with all other input parameters left as default. For the full depth forest, all other input parameters are left as the defaults, which allows the trees to grow to depths that result in purity (i.e., all observations within a leaf node are assigned to the same class or the number of observations within a leaf node is less than a specified value). The test data accuracy metrics for the shallow and full depth forests are 0.99 and 1.0, respectively. Figure 4-2 depicts the confusion

[1] cee-gitlab.sandia.gov/kjgoode/forestr

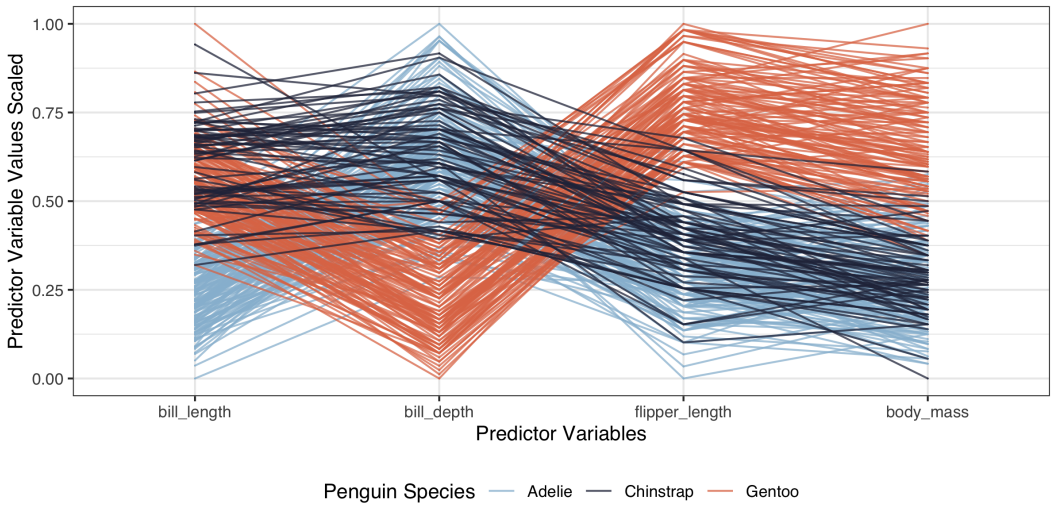

Figure 4-1.: Parallel coordinate plot depicting the Palmer penguin data. Each line represents one penguin, and the color of the line indicates the species of the penguin. The predictor variables are on the x-axis, and the y-axis depicts values of the predictor variables scaled between 0 and 1.

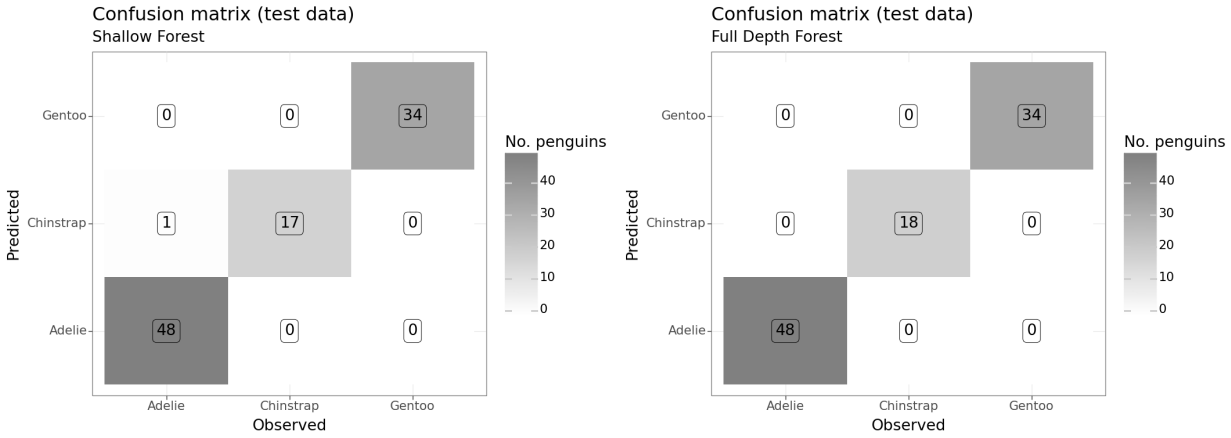

Figure 4-2.: Confusion matrices associated with the penguin random forest models.

matrices computed on the test data for both forest models. Both models perform well as expected given the clear variable relationships in the parallel coordinate plot.

In order to gain initial insight into how the random forest models use the input features for prediction, we consider the Gini index global feature importance values associated with each of the input features (Figure 4-3). Feature importance is one tool for explaining black-box models, and the Gini feature importance is a tool specific to random forests that measures reduction of the Gini index by a feature. A large value indicates that the feature is generally "important" to the model. Unsurprisingly, the importance associated with the features varies between the two penguin models. In the shallow forest, the variables of bill length, flipper length, and bill depth have approximately the same feature importance value. In the full depth model, bill length has the highest importance, and flipper length and bill depth are approximately tied for the second most important feature. In both models, body mass has the lowest importance. Feature importance

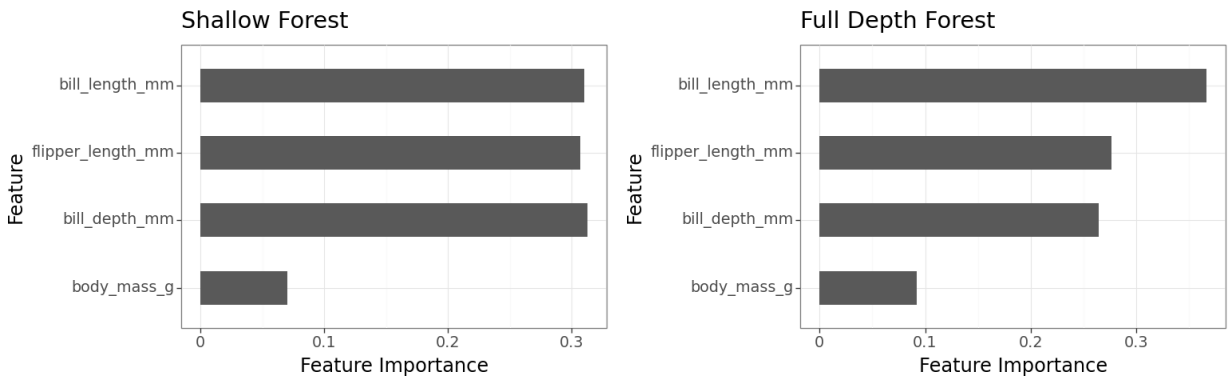

Figure 4-3.: Gini impurity-based feature importance values associated with the penguin random forest models.

provides one perspective into how the model uses the data for prediction, but feature importance does not provide a complete view. Additionally, research has shown that feature importance values may misrepresent the importance of a feature (e.g., [21]). As such, we aim to gain more insight into the models by exploring the tree topologies.

Figures 1-2 and 4-4 show all trees in the shallow and full depth forests, respectively, depicted using the traditional node and edge tree visualization. This type of visualization is useful for visualizing a small set of trees to interpret each tree individually but makes it challenging to compare trees. Figures 4-5 and 4-6 provide trace plots [8, 23] of the trees in the two models. Trace plots are useful visualizations for directly comparing the topology of the tree models. Figure 4-5 (left) shows the trace plot of one tree in the shallow forest. The node depth is plotted on the y-axis, and the features are plotted on the x-axis. The features are ordered from left to right by the most important to least important variable based on feature importance. The lines in the "trace" show which variable is used for splitting at a node. The horizontal lines associated with a node represent the range of the corresponding feature scaled to fall between 0 and 1, and the location where a trace passes through the horizontal line indicates the split threshold. Unlike the traditional tree visualizations, as in Figure 4-4, trace plots do not depict leaf nodes. The traces "end" at the last variable used for splitting before the leaf nodes.

The grid structure of trace plots allows for direct visual comparisons of multiple trees. Figures 4-5 and 4-6 (right) show trace plots of all 50 trees in the shallow and full depth forests, respectively. The traces are plotted with an opacity, so darker lines indicate multiple trees with the same path. These plots show variability in the tree topologies (e.g., different variables used for the first split) but also similar patterns. For example, body mass is rarely used for the first split but commonly used for splits at lower depths. While trace plots provide a visualization for starting to identify patterns in the tree topologies, we would like to use the tree metric to find clusters and representative trees to better identify the patterns.

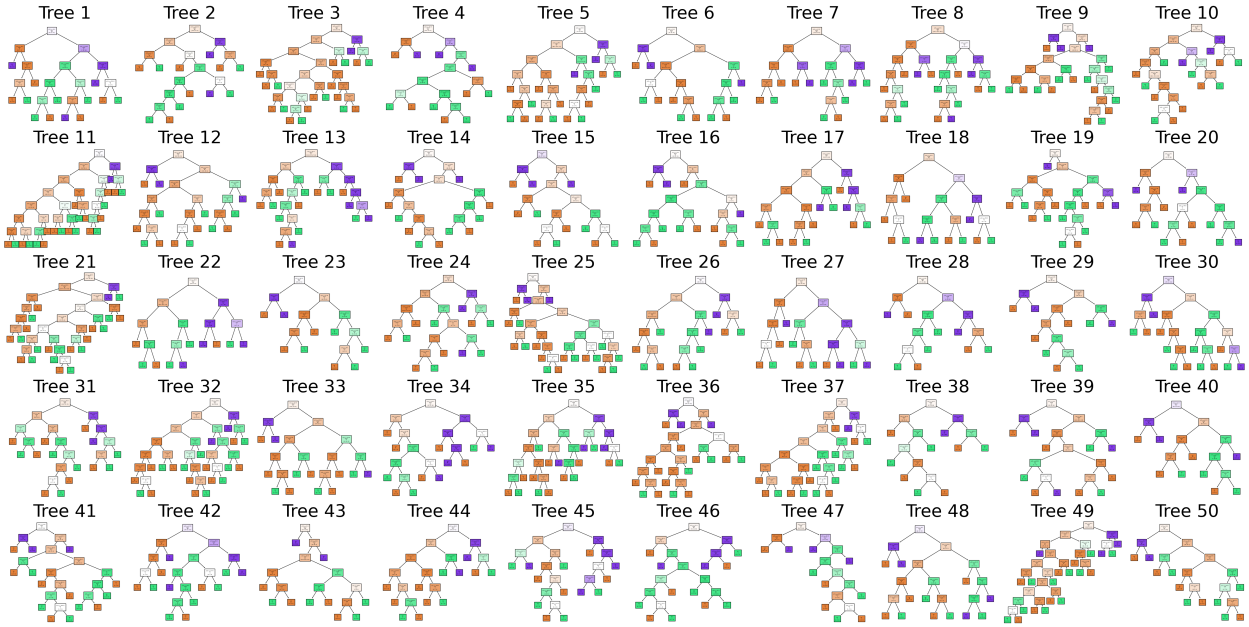

Figure 4-4.: The ensemble of trees in the penguin full depth random forest.

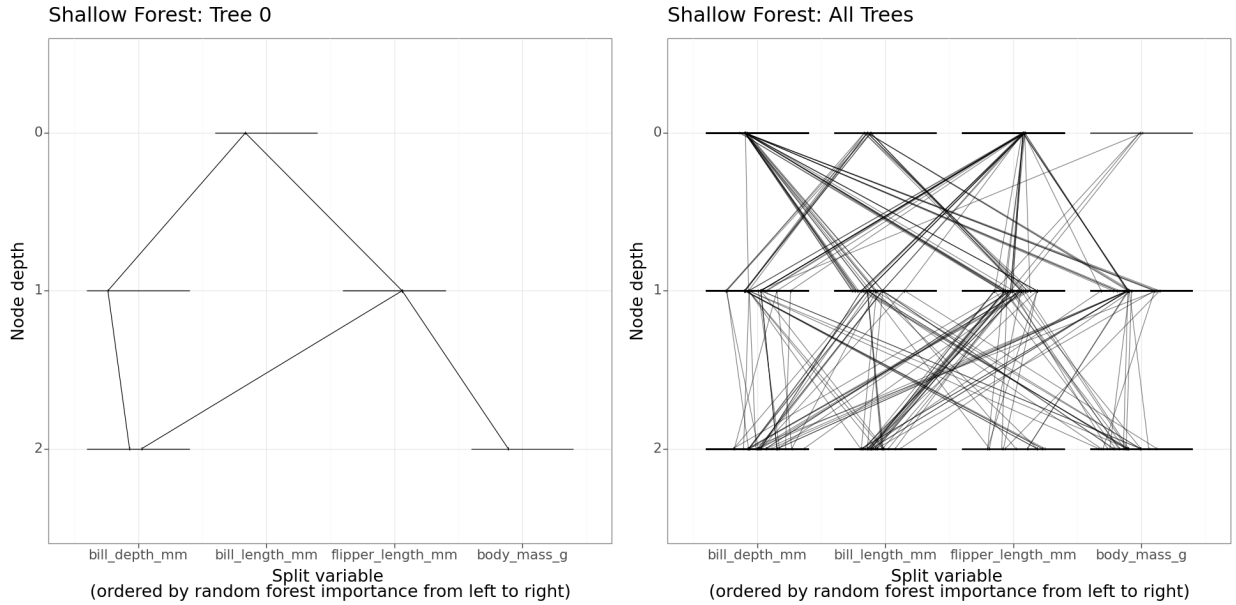

Figure 4-5.: (Left) Trace plot of one tree from the shallow penguin random forest. (Right) Trace plot of all trees in the shallow penguin random forest.

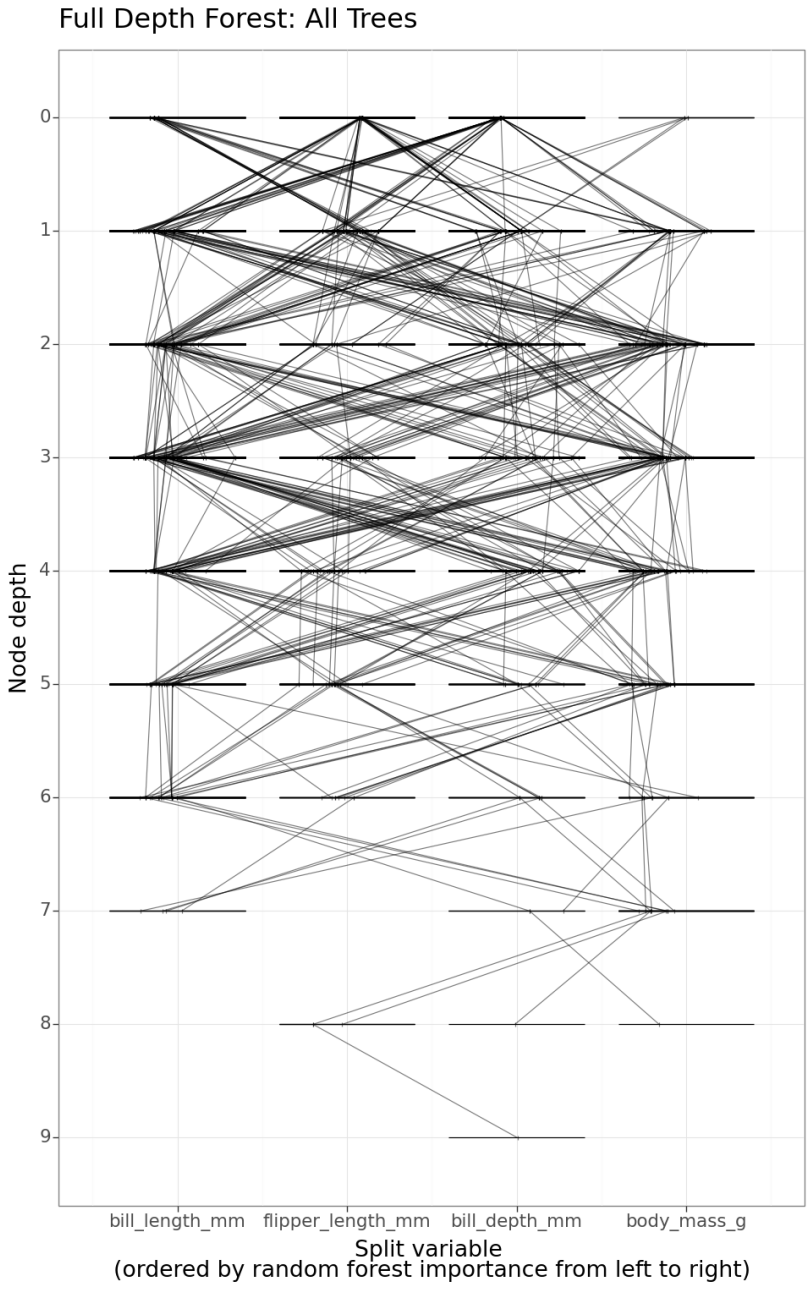

Figure 4-6.: Trace plot of all trees in the full depth penguin random forest.

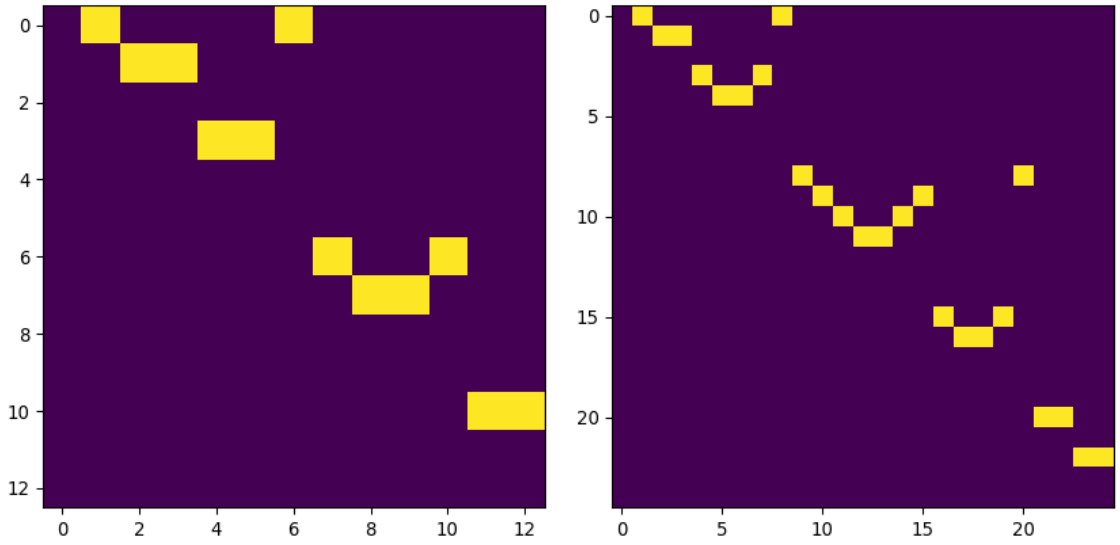

Figure 4-7.: Visual representations of the adjacency matrices for tree 0 in the shallow (left) and full depth (right) forests.

4.1.2. Tree Distances and Clusters

We use the tree metric described in Section 3 to compute all pairwise distances between the trees in both random forest models. To aid with understanding, Figure 4-7 depicts visual representations of the adjacency matrices associated with tree 0 in both shallow and full depth models. The row and column numbers are associated with the node numbers (including leaves). Thus, tree 0 in the shallow forest has 13 nodes, and tree 0 in the full depth forest has 25 nodes. The yellow cells indicate that there is an edge between two nodes. For example, in tree 0 of the shallow forest, there is an edge between the nodes numbered 0 and 1. In the numeric matrix form, the yellow cells would contain 1's, and the purple cells would contain 0's.

As described in Section 3, we compute the distances between trees using the node depths as node attributes, and we set $\lambda = 0.5$ to ensure weight is placed on these node attributes. Figure 4-8 depicts all pairwise distances between the trees in both random forests. The trees have been ordered based on a complete linkage clustering to help identify similarities between trees. In the shallow forest, there are clear clusters of trees with a distance of 0 (i.e., the trees possess the same topologies) and other larger clusters with small distances suggesting similar topologies. There are several clusters of trees in the full depth random forest as well, but the clusters are less clear than the shallow forest, which seems reasonable due to the larger trees. Figure 4-9 shows dendrograms from the complete linkage clusters. In both cases, five main clusters stand out.

We create trace plots of the trees within each of the identified clusters (Figure 4-10). The trace plots of the full depth forest clusters (Figure 4-10 bottom) show distinct differences between clusters. For example, the maximum depths of trees vary by cluster; the trees in cluster 2 are much shorter than those in clusters 4 and 5. Additionally, the feature used for the first several splits appears to vary. For example, all trees in clusters 1 and 5 split on flipper length first, but the second splits differ. That is, the trees in cluster 1 do not use body mass for the second split while the trees in cluster

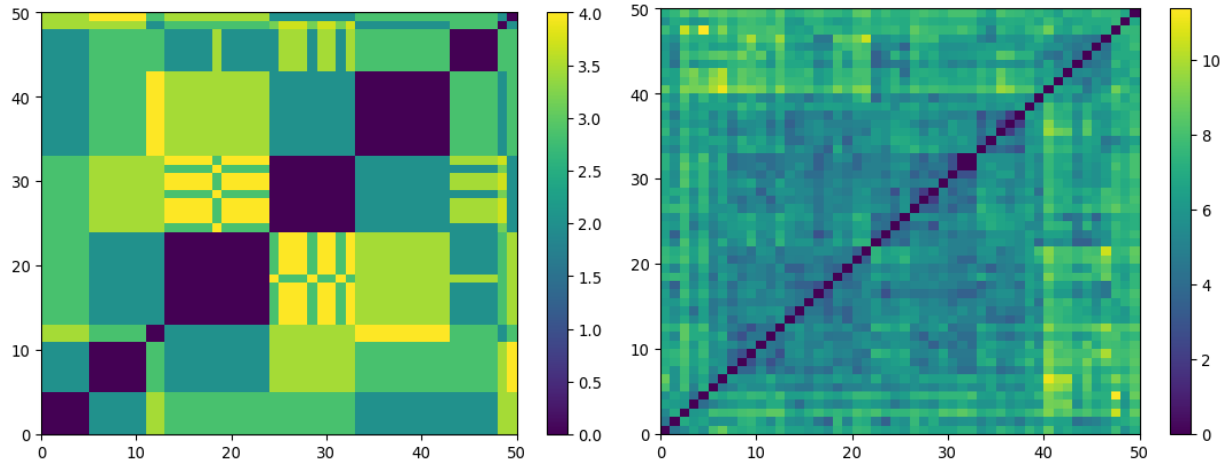

Figure 4-8.: Heatmaps of distances between trees in (left) shallow and (right) full depth penguin random forests.

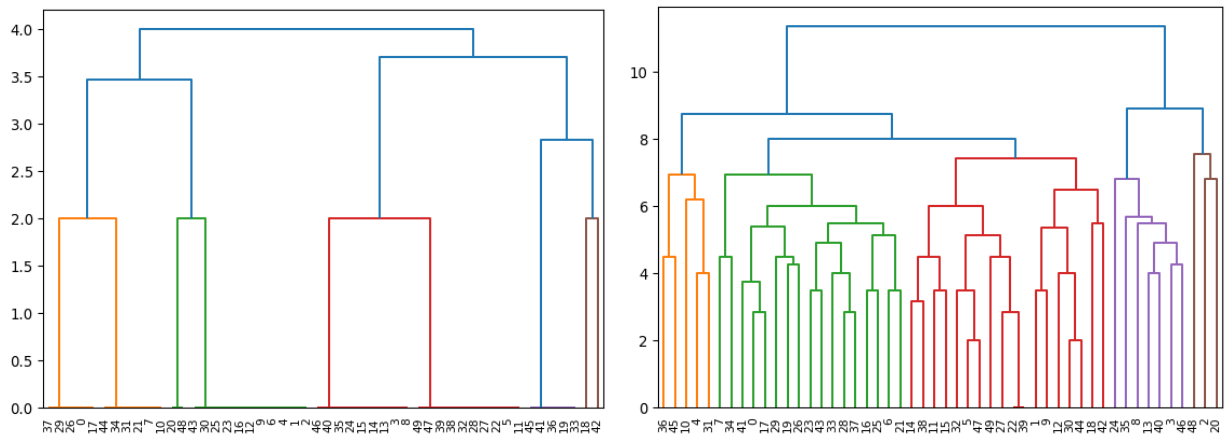

Figure 4-9.: Dendrograms of complete linkage clusters based on the distances between trees from the shallow (left) and full depth (right) penguin random forests.

5 do not use bill length for the second split. The trees in cluster 2 mostly use either bill length or flipper length for the first split, while the trees in clusters 3 and 4 mostly use bill depth for the first split. Regardless of the split pattern in low valued node depths, common split patterns are seen across the clusters at different node depths (e.g., a split on bill length followed by a split on body mass). These results seem to suggest that the overall topology of a tree is related to split decisions made early in the splitting process, but additional investigations into other factors that may affect the tree topologies (e.g., the bootstrap sample used to build a tree) should be considered.

The trace plots of the shallow forest clusters (Figure 4-10 top) also show differences in the most commonly used early splitting features. For example, a common pattern in cluster 2 is a first split on flipper length followed by splits on bill depth, bill length or flipper length. In cluster 3, the most common pattern is a split on bill depth first, followed by splits on bill length, flipper length, or body mass. As with the full depth forest, there is a relationship between the features used for early splitting and the tree topology, which is particularly interesting given that the feature used for splitting is not accounted for in the distance computation.

Since all trees in the shallow forest have a maximum depth of 3 (including leaves), it is a bit more challenging to pick out different in topological patterns in the trace plots in Figure 4-10. Figures 4-11 through 4-15 show individual trace plots of all trees within a cluster. These allow for a better understanding of the topology similarities within a cluster. For example, in cluster 1, most trees have all possible splits (i.e., split on 1 variable at depth 0, 2 splits at a depths of 1, and 4 four splits at a depth of 2). In contrast, cluster 4 has all trees with 1 split at depth 0, 2 splits at depth 1, but only 2 splits at depth 2. With these shallow trees, the tree topology similarities are especially apparent. However, there are some trees with the same tree topology in different clusters. For example, tree 0 in cluster 1 and tree 2 in cluster 2 (1 split at depth 0, 2 splits at depth 1, and 3 split at depth 2). Future work could explore why this is the case.

The trace plots of the tree clusters suggest that another distinguishing factor that is the number of splits in a tree, and as a result, the nodes within a tree. Figure 4-16 shows violin plots of the number of nodes in a tree by cluster and shallow or full depth random forest. There is an exceptionally clear trend in the shallow forest as we would expect based on the trace plots. The full depth forest has a less strong, but apparent, relationship between cluster and number of nodes. Clusters 1 and 5 stand out with the least amount of trees per cluster but containing many of the trees with the largest number of nodes. It would be interesting to explore how the random forest predictive performance is affected if the trees in these two clusters were removed.

4.1.3. Central Trees

To better understand the overall tree topology trends in the forest, we compute a mean graph from all trees of both random forests. As described in Section 3, we also compute a central tree as the tree in the forest with the smallest distance to the mean graph. Figure 4-17 shows histograms of the distances computed between each tree in a forest and the mean graph. In the shallow forest, there is a clear separation in distances that would be worth exploring further. In the full depth forest, there is right skew in the distribution of distances.

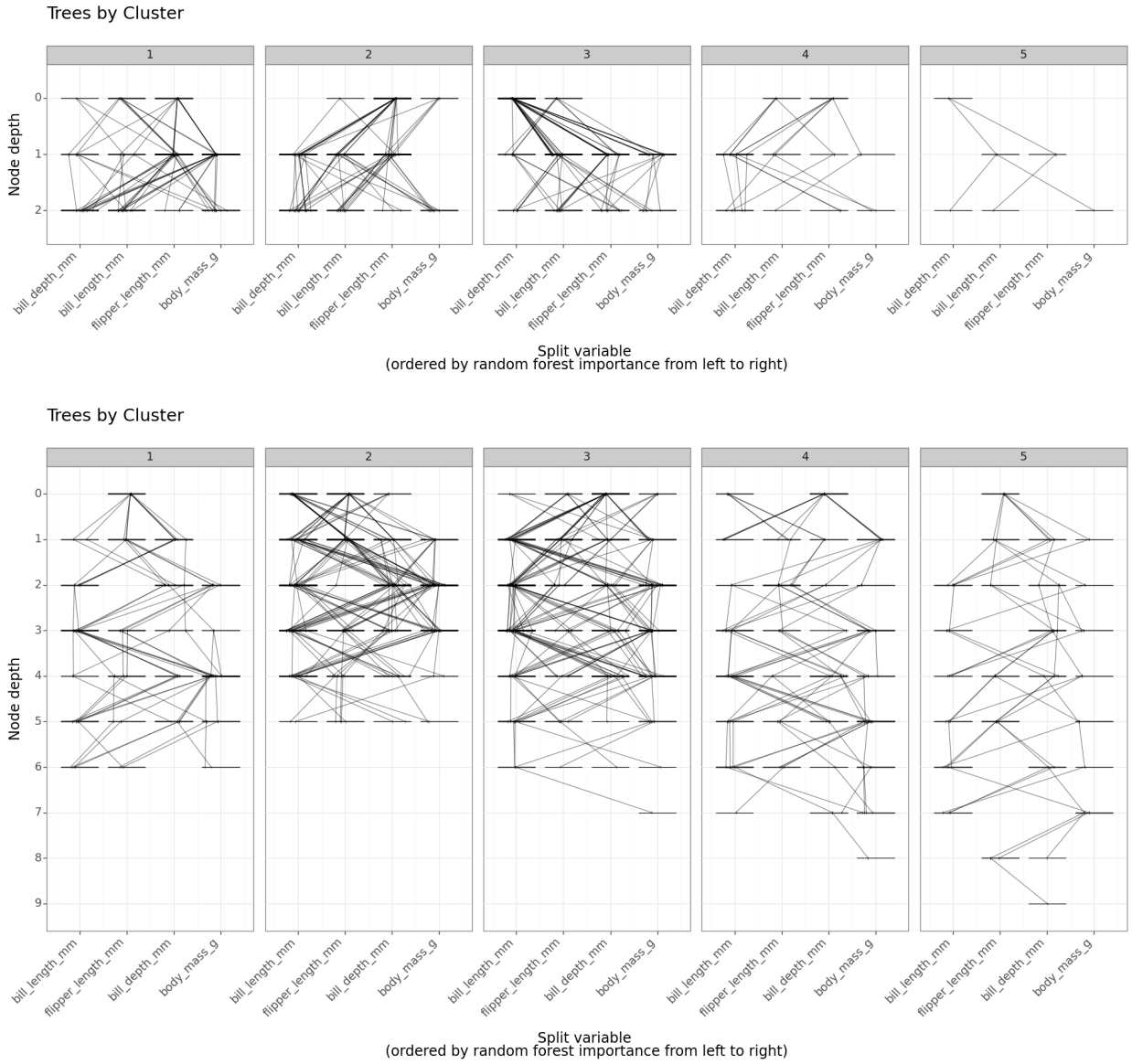

Figure 4-10.: Trace plots of trees within clusters identified based on tree topology metric from (top) shallow and (bottom) full depth penguin random forests.

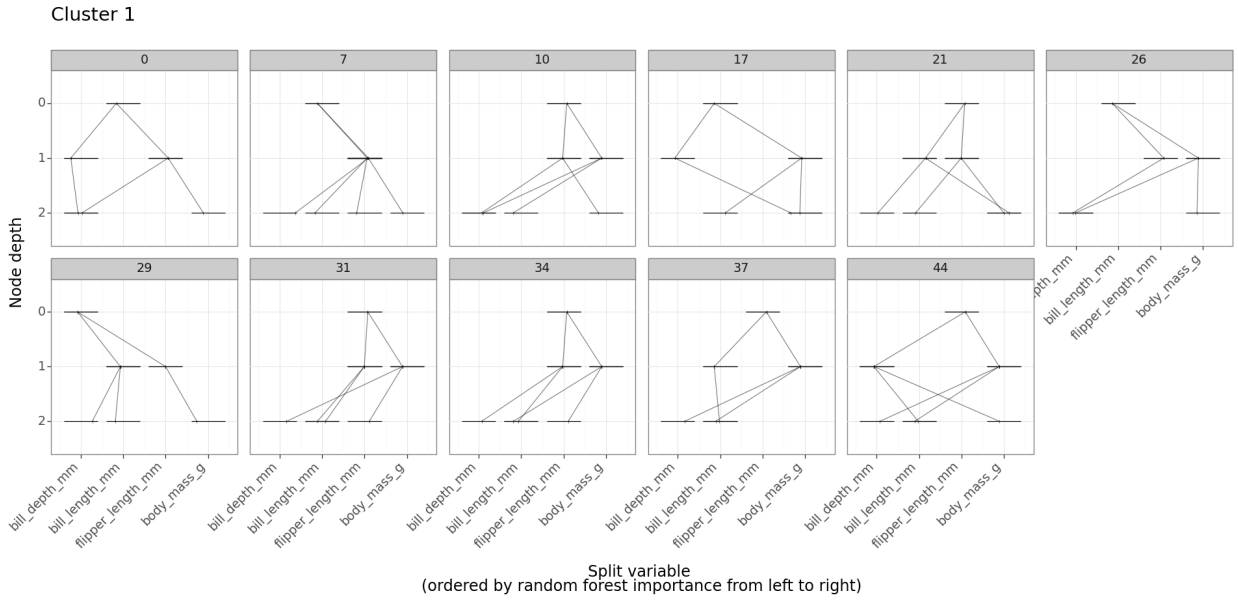

Figure 4-11.: Individual trace plots of all trees in cluster 1 from the shallow penguin random forest.

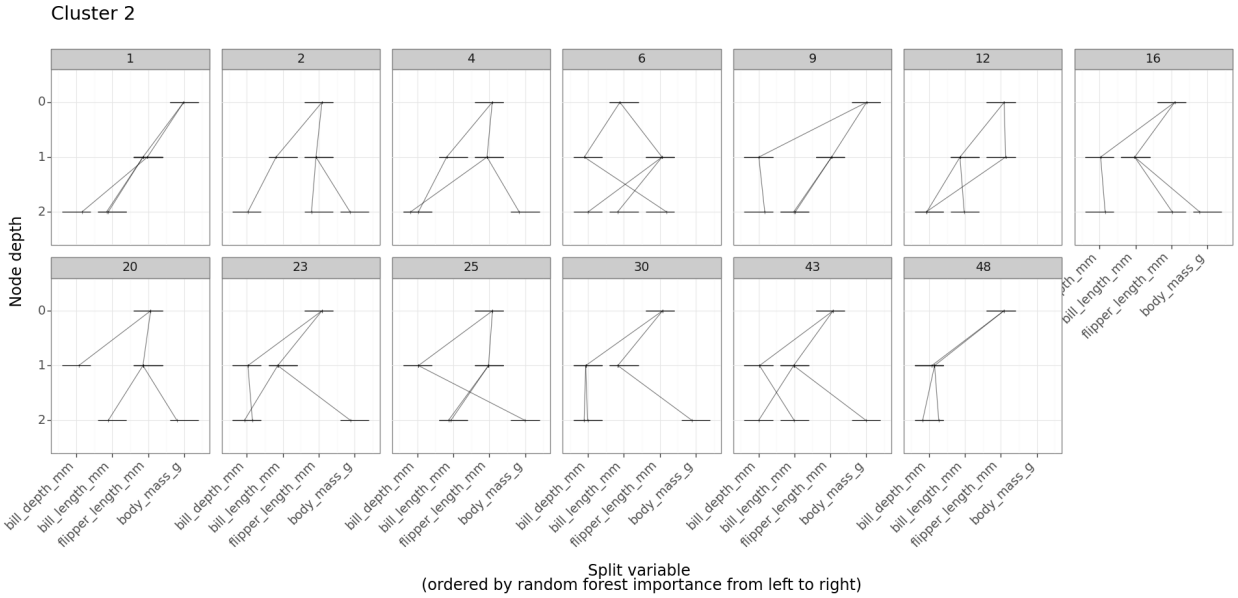

Figure 4-12.: Individual trace plots of all trees in cluster 3 from the shallow penguin random forest.

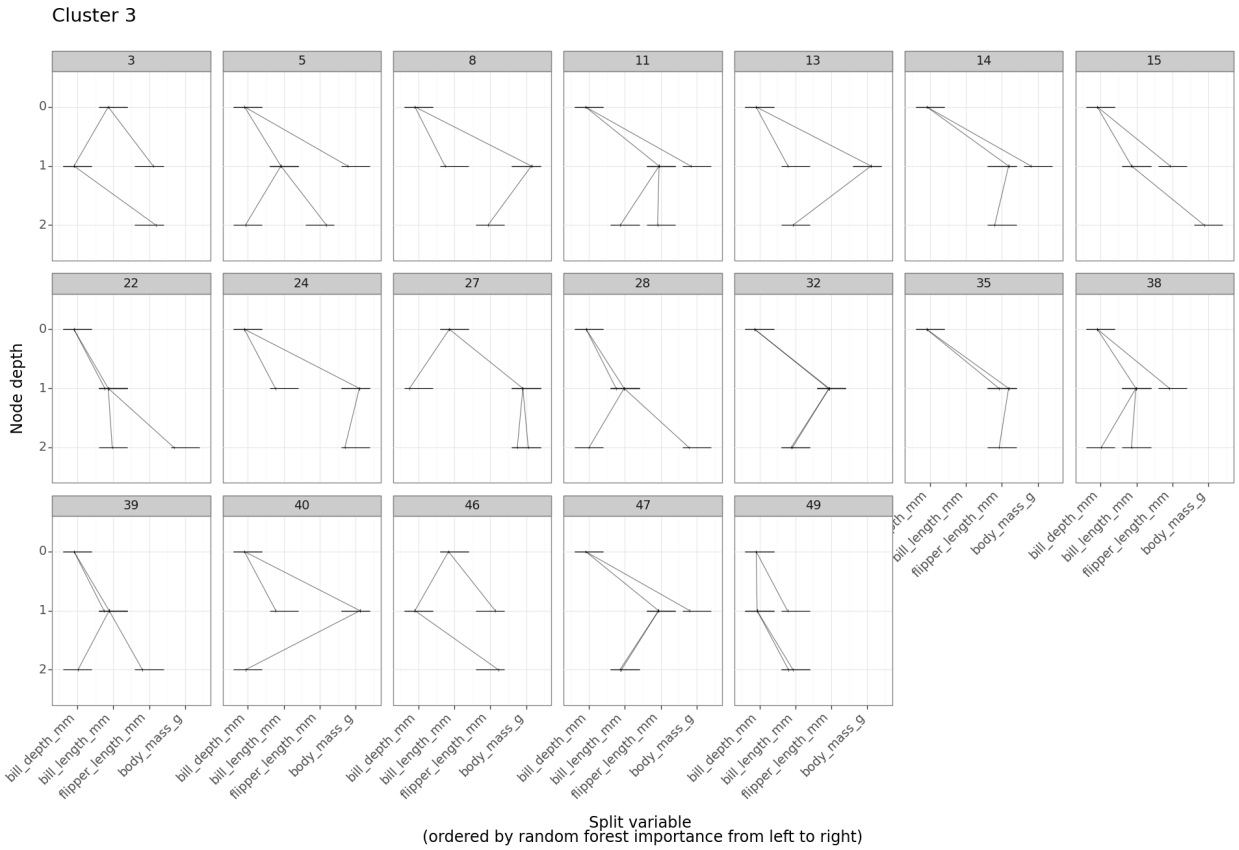

Figure 4-13.: Individual trace plots of all trees in cluster 3 from the shallow penguin random forest.

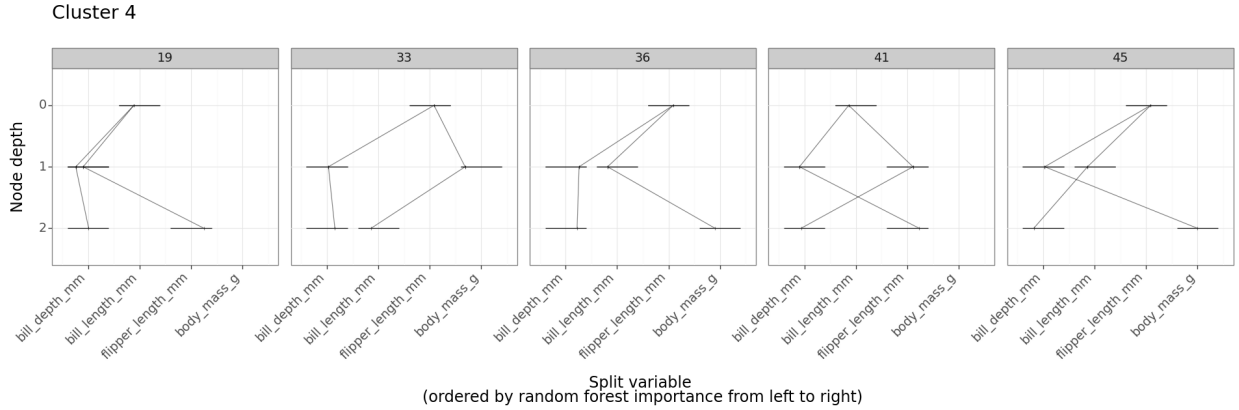

Figure 4-14.: Individual trace plots of all trees in cluster 4 from the shallow penguin random forest.

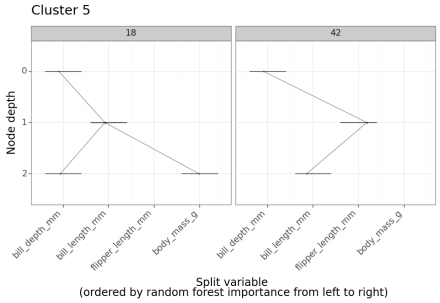

Figure 4-15.: Individual trace plots of all trees in cluster 5 from the shallow penguin random forest.

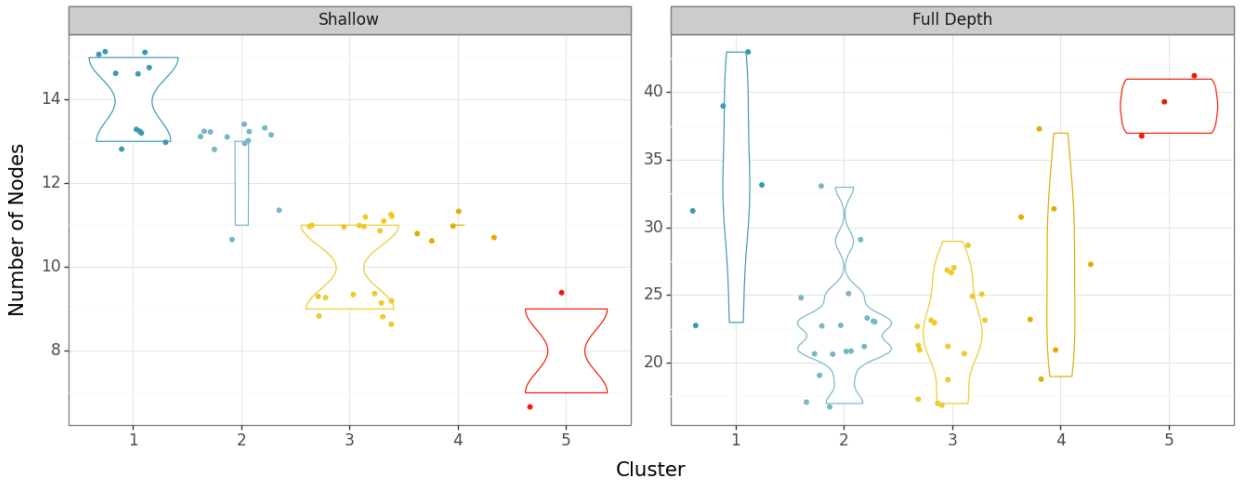

Figure 4-16.: Violin plots of the number of nodes in a tree by cluster (i.e., each point represents one tree) for the shallow (left) and full depth (right) penguin random forests.

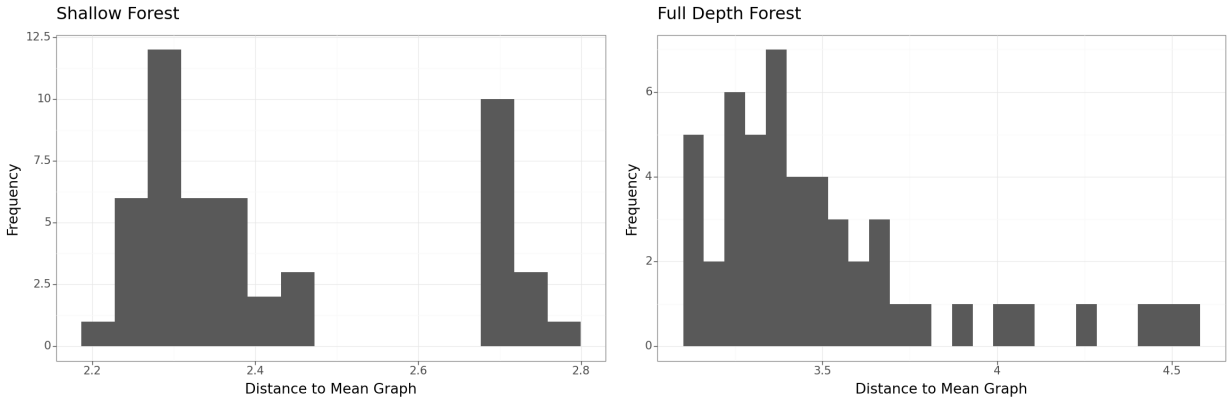

Figure 4-17.: Histograms of distances between tree topologies and the mean graph in the shallow (left) and full depth (right) forests.

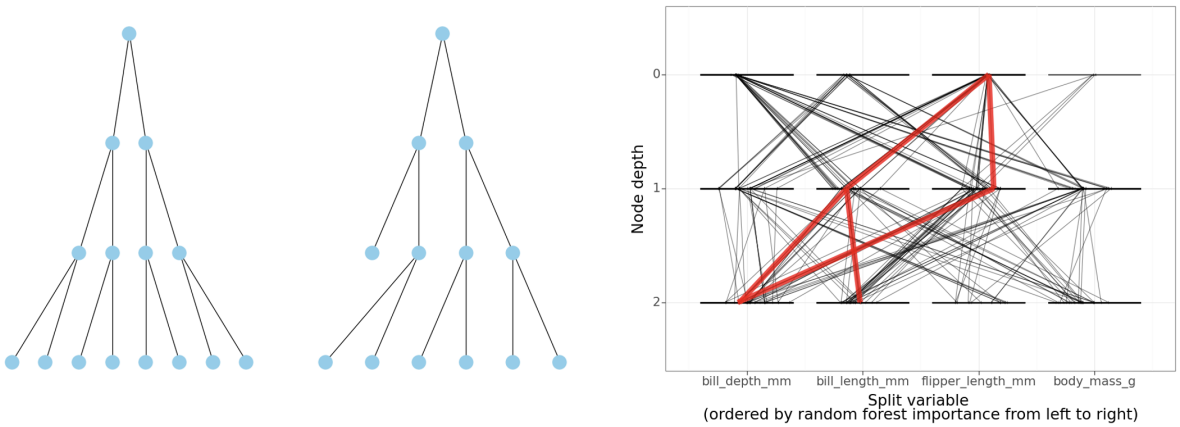

Figure 4-18.: (Left) Mean graph computed from all trees in shallow penguin random forest. (Center) Central tree from the shallow random forest. (Right) Trace plots of all trees in shallow random forest with central tree in red.

Figure 4-18 shows the mean graph (left) and the central tree (center) from the shallow forest. The central tree reflects the mean graph but does not contain all possible nodes like the mean graph. This is surprising since Figure 4-11 shows trees in the shallow forest that have all possible nodes and should be explored in future work. Figure 4-18 also includes a trace plot of the central tree in relationship to all other trees in the forest (right).

Figure 4-19 shows the mean graph, the central tree, and a trace plot of the central tree in relationship to all trees in the full depth forest. For this model, the mean graph has lots of edges with low weights (lighter lines). The topology of the central tree mainly follows the edges in the mean graph with higher weights. The trace plot highlights that the central tree has a maximum depth of 4, which is much lower than the maximum depth of the deepest tree in the forest (depth of 9).

We additionally compute the mean graph and central trees for each of the identified clusters in the shallow and full depth random forests. Figure 4-20 shows the distances between the trees within a cluster and the mean graph of the corresponding cluster. Figures 4-21 and 4-22 show the mean graphs (top rows) and central trees (bottom rows) for each of the clusters. Similar to the mean graphs and central trees for the entire ensembles of trees, the cluster central trees closely reflect the cluster mean graphs. In fact, the mean graphs and central trees for the shallow forest have the exact same topologies.

While only one tree is depicted as the central tree for the clusters in the shallow forest in Figure 4-21, there are actually multiple trees within a cluster that have the smallest distance to the mean graph of the cluster. For example, Figure 4-23 shows all trees in cluster 1 with the smallest distance to the mean graph. While the trees use different variables for splitting, the topologies of the trees are equivalent. In the case of the full depth forest, the central trees in each cluster are unique.

As desired, the central trees help to highlight the topology patterns identified by the clusters. For example, in cluster 4 from the full depth forest, the central tree stands out by having three leaf nodes at an early depth of 2. Figure 4-24 shows individuals trace plots of all trees in the cluster 4 of the full depth forest. All the trees have the distinct topology of leaf nodes at early depths. In fact, trees 8, 13, 24, and 40 even use the same features for splits at depths 0 and 1. This pattern

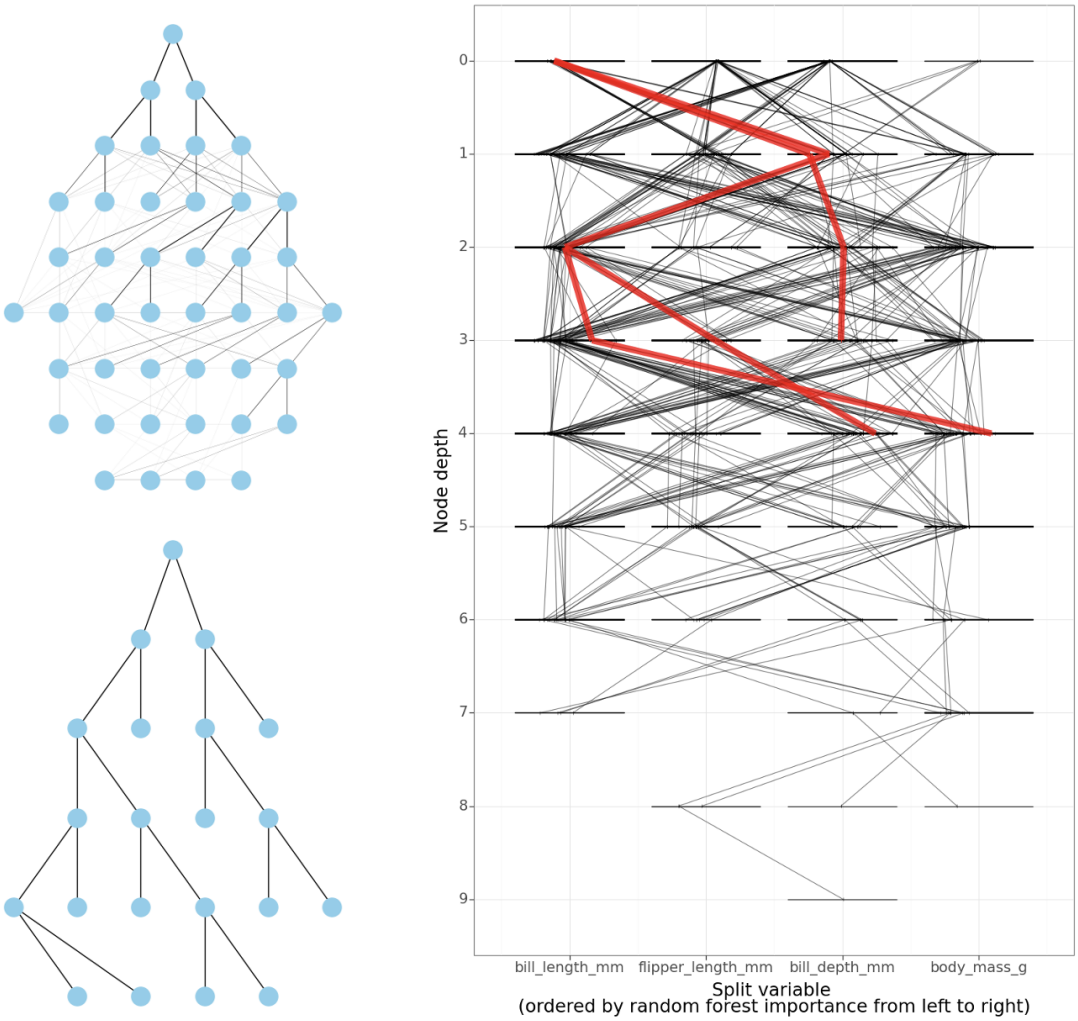

Figure 4-19.: (Left; Top) Mean graph computed from all trees in full depth penguin random forest. (Left; Bottom) Central tree from the full depth random forest. (Right) Trace plots of all trees in full depth random forest with central tree in red.

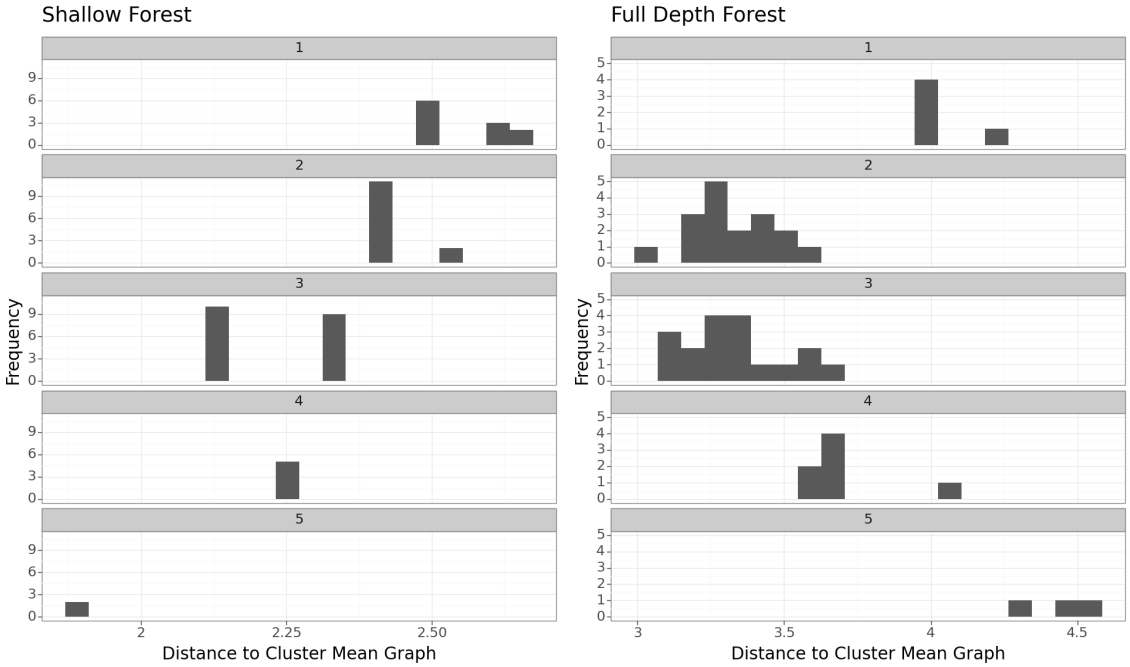

Figure 4-20.: Histograms of distances between a tree and the corresponding cluster mean graph from the shallow (left) and full depth (right) penguin random forests. The distances are faceted by cluster (rows).

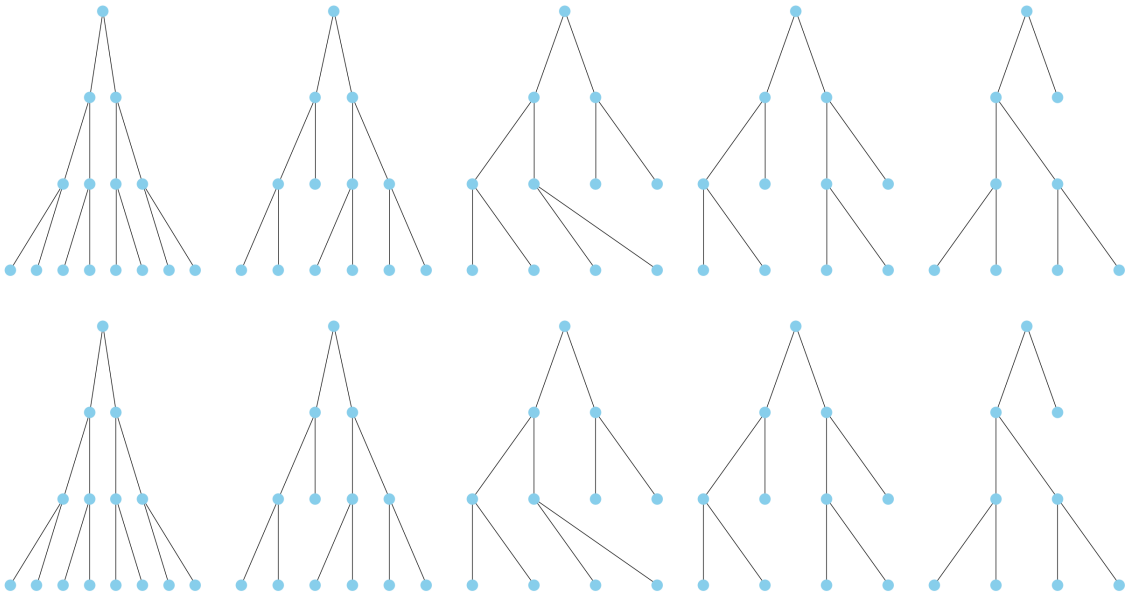

Figure 4-21.: Mean graphs (top row) and central trees (bottom) row for each of the tree clusters in the shallow penguin random forest (clusters ordered left to right from 1 to 5).

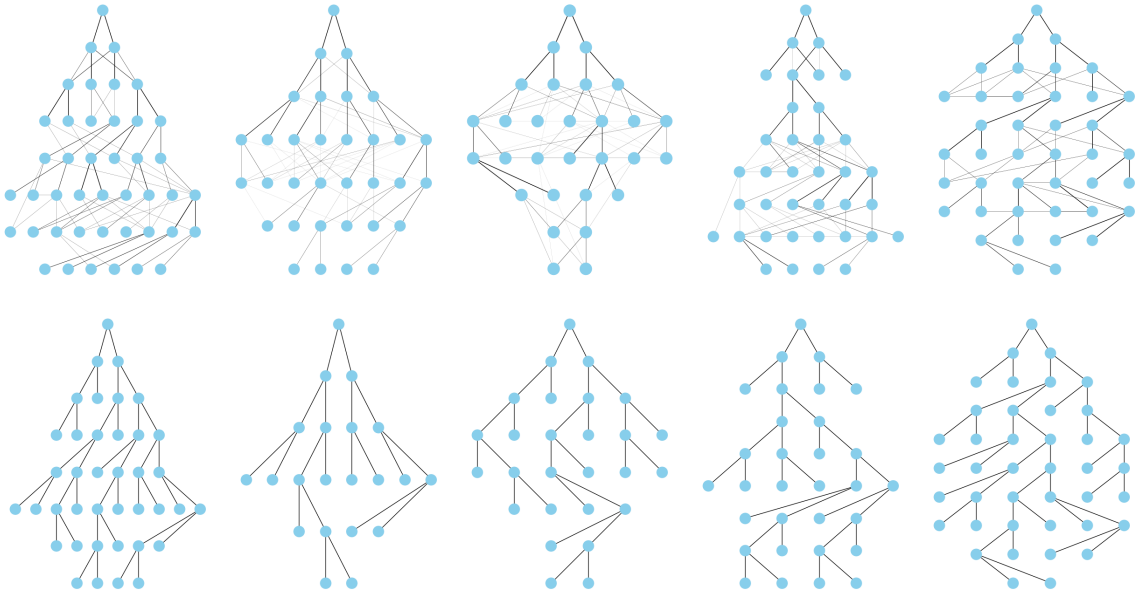

Figure 4-22.: Mean graphs (top row) and central trees (bottom) row for each of the tree clusters in the full depth penguin random forest (clusters ordered left to right from 1 to 5).

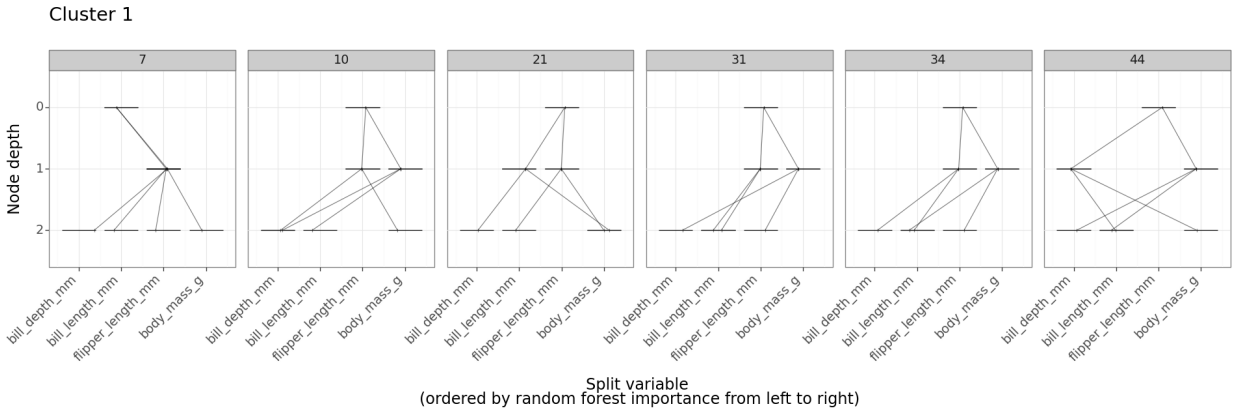

Figure 4-23.: All trees cluster 1 from the shallow penguin random forest with the equally smallest distances to the mean graph of cluster 1.

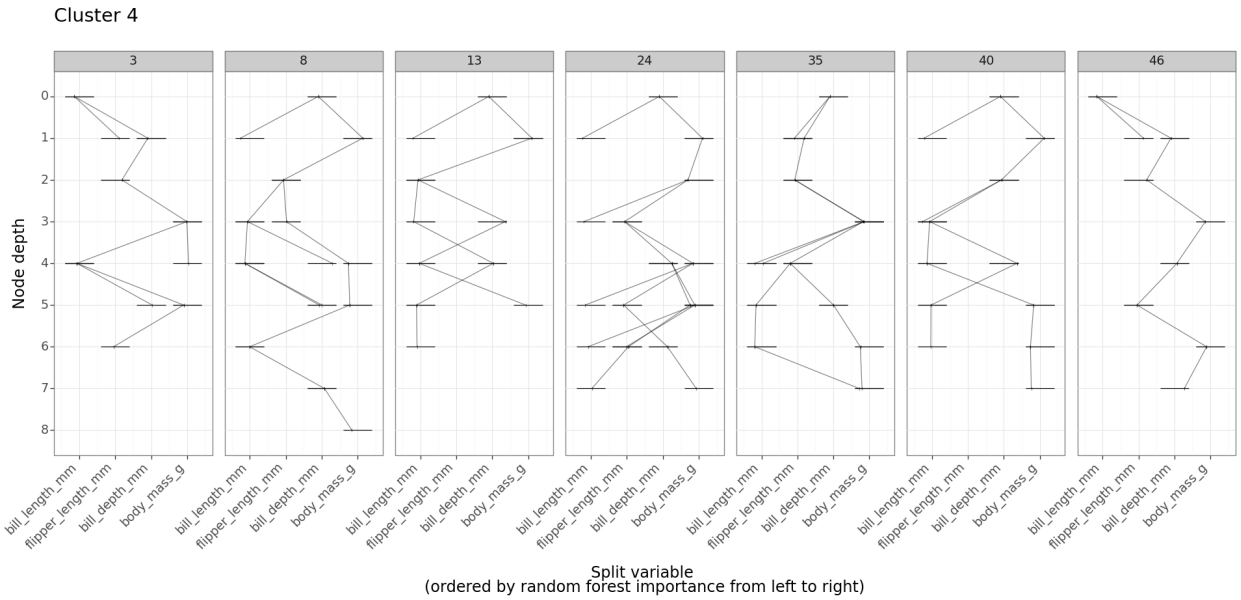

Figure 4-24.: Trace plots of all trees in cluster 4 of the full depth penguin random forest.

seems to suggest that the trees identify a clear decision boundary at early depths and then working to find more nuanced decision boundaries at larger depths. This contrasts with other trees that require more depths to find these decision boundaries. However, as previously mentioned, future work could explore if this pattern is due to the bootstrap samples used to train this tree or some other factor.

4.2. s500 Data

The s500 data contains information from a product inspection application [13]. There are 20 numeric features associated with 1000 products identified as good or bad. The data are separated into training and testing sets with 500 observations in each set. We train a random forest to predict whether the product is good or bad. The model consists of 25 trees with a specified maximum depth of 5. We choose to use a maximum depth of 5 for this analysis to simplify the tree topologies and increase computational efficiency. All other tuning parameters are set to the defaults in scikit-learn. However, future work could explore an analysis on the s500 data allowing the random forest trees to grow to purity.

The random forest accuracy on the s500 test data is 0.91. Figure 4-25 shows parallel coordinate plots of the train and test sets. The observations are colored by the true label of whether the object is good (1) or bad (-1). The features in the parallel coordinates plot are ordered from left to right based on highest to lowest Gini feature importance. The feature importance values are included in Figure 4-26. Figure 4-27 shows a trace plot of all trees in the s500 random forest.

We apply the tree metric to compute all pairwise distances between the trees in the s500 random forest with $\lambda = 0.5$. Again, we apply complete linkage clustering to the distances. Figure 4-28

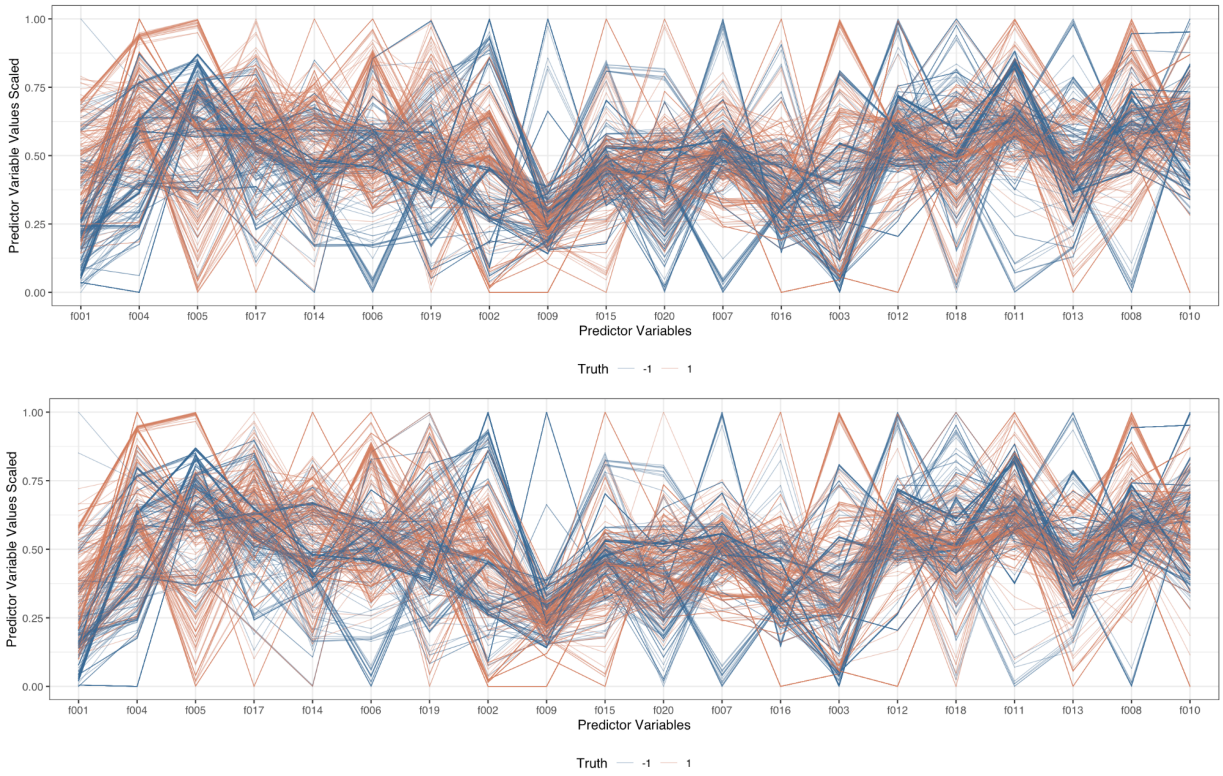

Figure 4-25.: Parallel coordinate plots of s500 training (top) and testing (bottom) data. Lines are colored by the inspection labels, and the features are ordered by Gini importance.

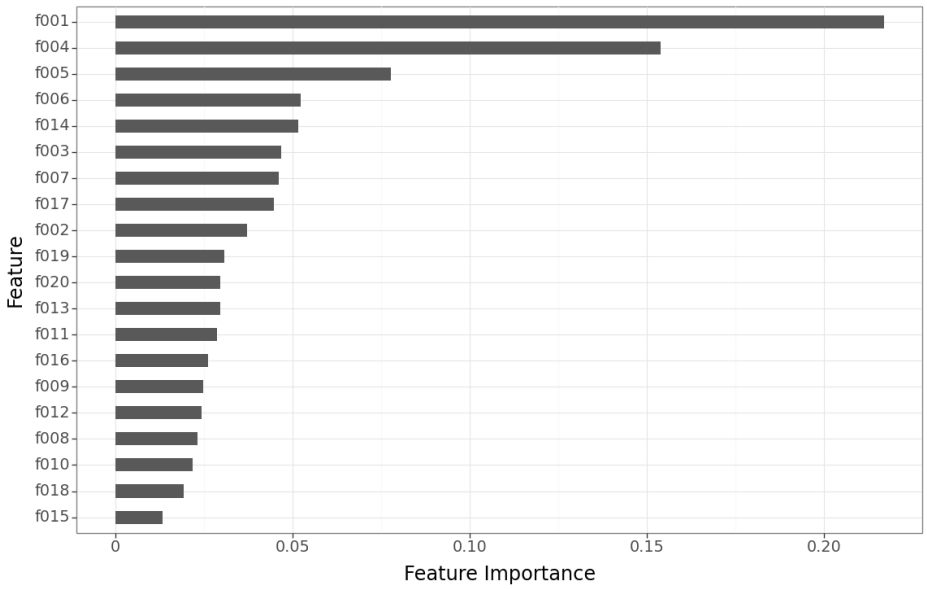

Figure 4-26.: Gini random forest feature importance from the s500 model.

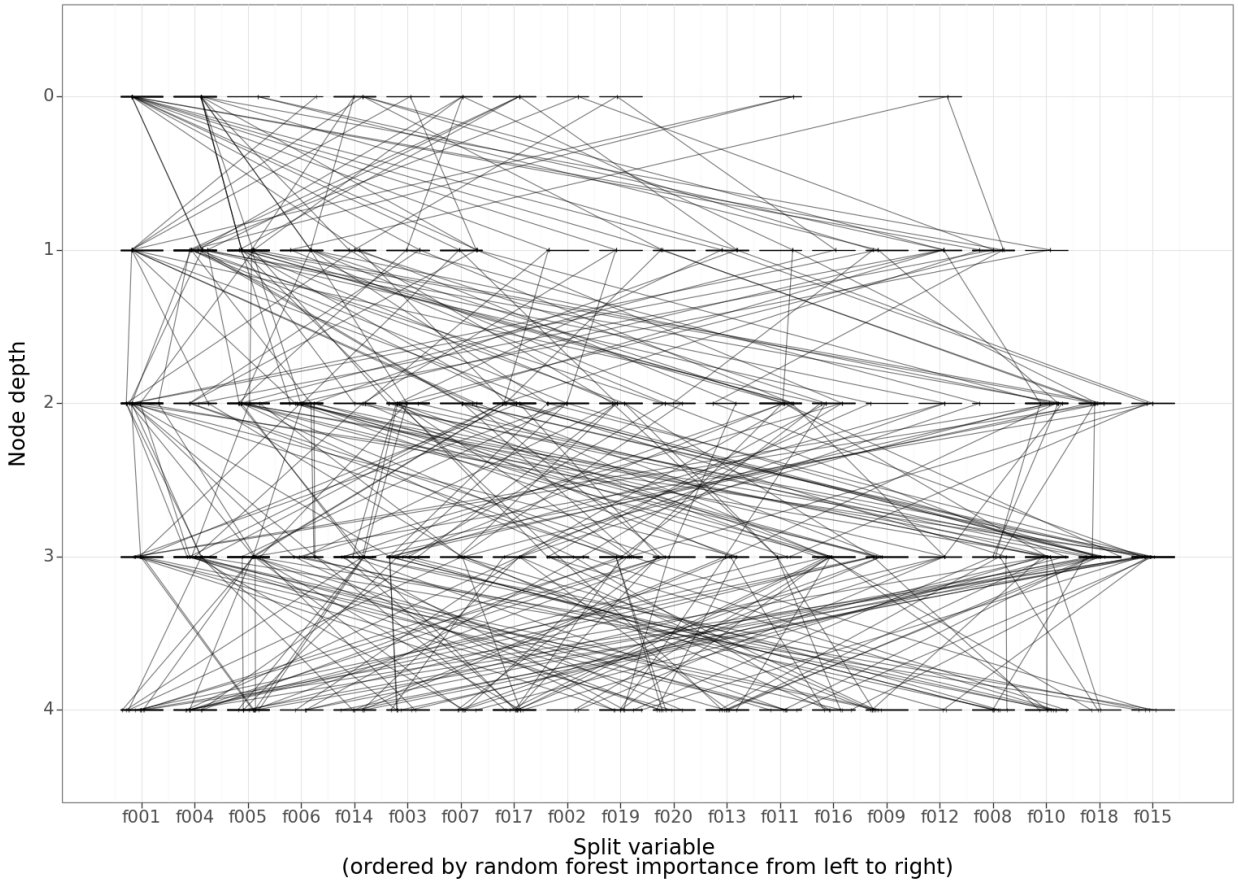

Figure 4-27.: Trace plot of the trees in the s500 random forest.

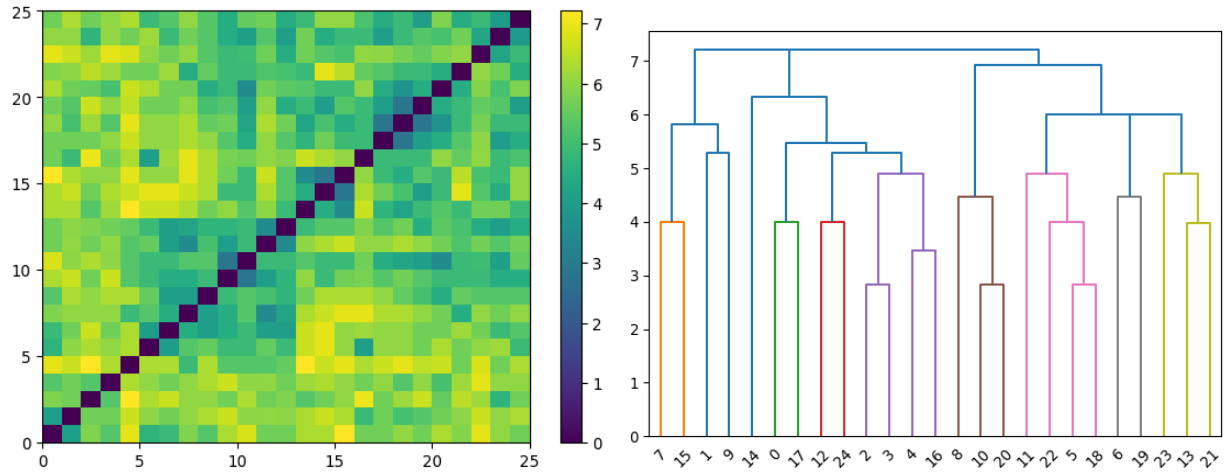

Figure 4-28.: (Left) Distances computed between the s500 random forest tree topologies. (Right) Dendrogram from complete linkage clustering of the tree topologies.

shows a heat map of all pairwise distances (left) and the dendrogram from the complete linkage clustering. We elect to cut the dendrogram at 6, which creates 5 large clusters. However, it would be interesting to return and consider the 10 smaller clusters identified by the colors in the dendrogram.

Figure 4-29 shows trace plots of the trees in the five clusters. The number of trees in clusters 3 and 4 is much less than the other clusters (e.g., only 1 tree in cluster 3). This suggests that the trees in clusters 3 and 4 might be outliers. Unlike the penguin data, there does not appear to be as clear of a relationship between the features used for splitting and the tree topologies based on the trace plot visualizations.

Again, we compute mean graphs and central trees for each cluster (Figure 4-30). The central trees help identify differences between the clusters: the depths at which leaf nodes begin to occur and the number of leaf nodes at a depth. For example, the central tree of cluster 1 has more leaf nodes at a depth of 4 than the other clusters, and cluster 4 is the only cluster whose central tree does not have any leaf nodes at a depth of 6.

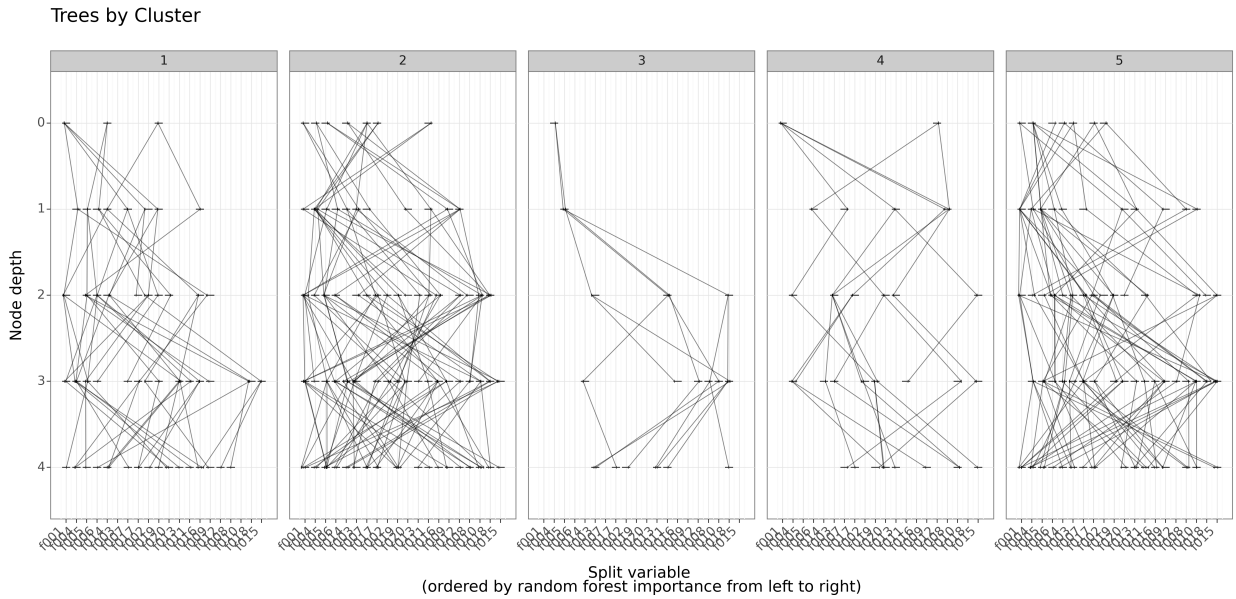

Figure 4-29.: Trace plots of the tree clusters from the s500 random forest.

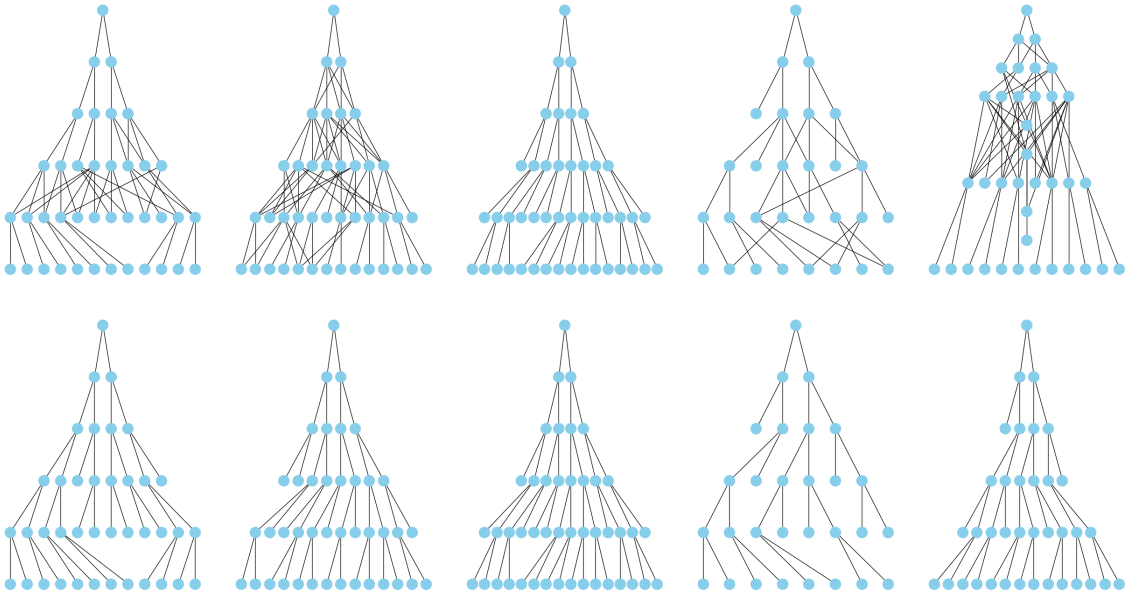

Figure 4-30.: Mean graphs (top row) and central trees (bottom row) from the tree clusters in the s500 random forest (clusters ordered left to right from 1 to 5).

This page intentionally left blank.

5. CONCLUSIONS

In this report, we propose a new metric for computing distances between trees based on their topologies. We introduce this metric to gain insight into the decision process of ensembles of trees. Specifically, we use the metric to identify clusters in an ensemble of trees and compute central trees. By visualizing tree clusters and central trees, we found tree topology similarities within clusters, differences between clusters, and central trees that meaningfully reflect the topologies within a cluster. We additionally found evidence of relationships between tree topologies and the features used for splits.

These results suggest that the proposed tree metric is able to identify topology patterns as hypothesized. This lays the groundwork for future research into using the developed metric to either thin random forests or build models using representative trees to create less complex models. While the main objective in creating a smaller ensemble is to have a simpler, more interpretable model, it is desirable to preserve the original model's predictive performance. Depending on the application, it may be permissible to sacrifice some predictive performance in favor of a more interpretable model, but it will be important to compare model predictive performance between the original and smaller ensembles. Figure 5-1 depicts a hypothetical comparison between model predictive performance and the number of trees in the model. Before removing trees, developers could decide on an allowable performance decrease for the given application.

Potential routes for using the tree metric to create more interpretable models include:

- A model with only the central tree computed from all trees in the original ensemble.
- An ensemble composed of the central trees from identified clusters.
- Gradually add trees to an ensemble of central trees based on their distance to the central tree (e.g., trees that are close to a central tree or a distribution of distances from a central tree that add diversity to the ensemble).
- Gradually remove trees from the original ensemble based on their distance to the overall central tree or cluster central trees.

In addition to the development of an algorithm for constructing a more interpretable ensemble, there are opportunities for improvement on the methodology proposed in this report. For example, it could be beneficial to compare the use of different clustering algorithms for identifying tree topology clusters to determine if certain methods are more suited for this endeavor. Another direction is further development of the tree distance calculation such as the inclusion of other tree characteristics or weights applied to the edges to place higher weight on certain parts of trees.

In general, the development of interpretable or explainable models is a challenging but important area for improved methodologies. While this report considers one potential route, it is important to

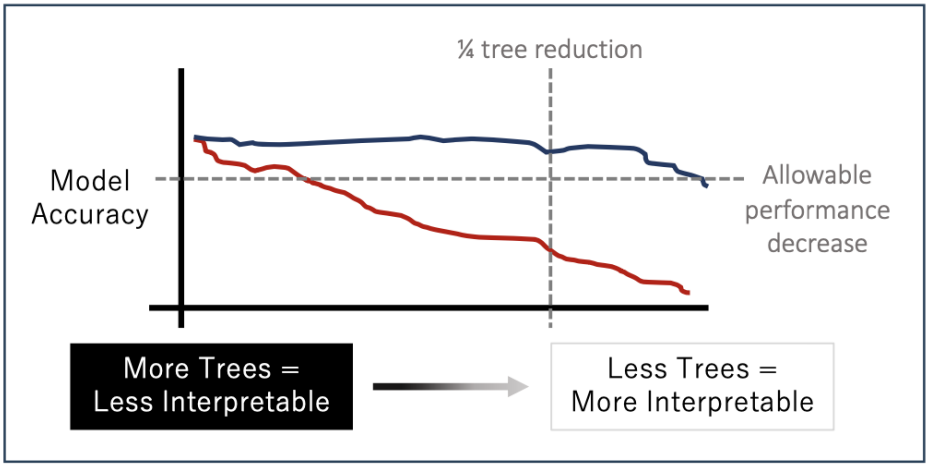

Figure 5-1.: Hypothetical comparison of model performance and the number of trees in a model.

recognize that one method may not be suitable for all data analyses. Furthermore, we advocate for a toolbox of methods for peering into black-box models as opposed to one method that sheds light on all aspects of a black-box. For example, even if an ensemble of trees is reduced by a quarter or half, the model may still possess enough complexity that additional explainability methods may be needed to help parse the decision paths used by the ensemble. Thus, we encourage the development of more explainability (and interpretability) methods and strategies for evaluating them.

BIBLIOGRAPHY

[1] A. Adadi and M. Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.

[2] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

[3] M. Banerjee, Y. Ding, and A. Noone. Identifying representative trees from ensembles. *Statistics in Medicine*, 31(15):1601–1616, 2012.

[4] L. Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[5] H. A. Chipman, E. I. George, and R. E. McCulloch. Making sense of a forest of trees. In *Proceedings of the 30th Symposium on the Interface*, pages 84—92, 1998.

[6] A. Fisher, C. Rudin, and F. Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research*, 20(177):1–81, 2019.

[7] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[8] Katherine Goode. *Visual diagnostics for explaining machine learning models*. Phd thesis, Iowa State University, Ames, IA, 2021.

[9] X. Guo and A. Srivastava. Representations, Metrics and Statistics For Shape Analysis of Elastic Graphs. *arXiv:2003.00287*, 2020.

[10] X. Guo, A. Srivastava, and S. Sarkar. A Quotient Space Formulation for Generative Statistical Analysis of Graphical Data. *Journal of Mathematical Imaging and Vision*, 63(6):735–752, 2021.

[11] A. M. Horst, A. P. Hill, and K. B. Gorman. *palmerpenguins: Palmer Archipelago (Antarctica) penguin data*, 2020. R package version 0.1.0.

[12] B. J. Jain. On the geometry of graph spaces. *Discrete Applied Mathematics*, 214:126–144, 2016.

[13] P. Kegelmeyer, T. M. Shead, J. Crussell, K. Rodhouse, D. Robinson, C. Johnson, D. Zage, W. Davis, J. Wendt, T. Cayton J. Doak, R. Colbaugh, K. Glass, B. Jones, and J. Shelburg. Counter adversarial data analytics. Technical report, Sandia National Labs (SAND2015-3711), 2015.

[14] W. McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[15] C. Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[16] C. Molnar, G. Casalicchio, and B. Bischl. Interpretable machine learning–a brief history, state-of-the-art and challenges. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 417–431. Springer, 2020.

[17] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

[18] C. Rudin, C. Chen, Z. Chen, H. Huang, L. Semenova, and C. Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys*, 16:1–85, 2022.

[19] W. D. Shannon and D. Banks. Combining classification trees using MLE. *Statistics in Medicine*, 18(6):727–740, 1999.

[20] A. Sies and I. V. Mechelen. C443: a Methodology to See a Forest for the Trees. *Journal of Classification*, 37(3):730–753, 1 2020.

[21] C. Strobl, A. Boulesteix, A. Zeileis, and T. Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC bioinformatics*, 8:1–21, 2007.

[22] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988.

[23] S. Urbanek. *Visualizing Trees and Forests*, pages 243–264. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[24] J. T. Vogelstein, J. M. Conroy, V. Lyzinski, L. J. Podrazik, S. G. Kratzer, E. T. Harley, D. E. Fishkind, R. J. Vogelstein, and C. E. Priebe. Fast Approximate Quadratic Programming for Graph Matching. *PLoS ONE*, 10(4):e0121002, 2015.

[25] A. I. Weinberg and M. Last. Selecting a representative decision tree from an ensemble of decision-tree models for fast big data classification. *Journal of Big Data*, 6(1):23, 2019.

DISTRIBUTION

Email—Internal

Name	Org.	Sandia Email Address
Technical Library	1911	sanddocs@sandia.gov

Hardcopy—Internal

Number of Copies	Name	Org.	Mailstop
1	Philip Kegelmeyer	87030	9159
1	Jeremy D. Wendt	5525	1138
1	L. Martin, LDRD Office	1910	0359

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.