8-2022

Geometry- and Accuracy-Preserving Random Forest Proximities with Applications

Jake S. Rhodes
Utah State University

GEOMETRY- AND ACCURACY-PRESERVING RANDOM FOREST PROXIMITIES WITH APPLICATIONS

by

Jake S. Rhodes

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Statistics

Approved:

UTAH STATE UNIVERSITY
Logan, Utah

2022

ABSTRACT

Geometry- and Accuracy-Preserving Random Forest Proximities with Applications

by

Jake S. Rhodes, Doctor of Philosophy

Utah State University, 2022

Major Professor: Kevin R. Moon, Ph.D.
Department: Mathematics & Statistics

Many machine learning algorithms require pairwise distances or similarities to generate decision boundaries or perform more general pattern analysis. Most similarity measures are supervised and do not make any use of data labels or other auxiliary information. Supervised adaptations have been proposed that directly incorporate class labels to exaggerate inter-class separation or reduce distances between within-class observations. However, it can be shown that such incorporations disrupt the data structure. Random forests are classical machine learning predictors which may be used to generate supervised similarities known in the literature as random forest proximities. We show that traditionally-defined random forest proximities inherit certain weaknesses similar to those found in other supervised similarity measures. We introduce a novel random forest proximity definition that reflects the true learning of the random forest. We exploit these proximities in a variety of applications, including data visualization, missing value imputation, and outlier detection, and show improvements in each application. Additionally, we use random forest proximities in a new diffusion-based information geometry for supervised dimensionality reduction and quantify improvements over existing supervised dimensionality reduction techniques.

(116 pages)

PUBLIC ABSTRACT

Geometry- and Accuracy-Preserving Random Forest Proximities with Applications

Jake S. Rhodes

Many machine learning algorithms use calculated distances or similarities between data observations to make predictions, cluster similar data, visualize patterns, or generally explore the data. Most distances or similarity measures do not incorporate known data labels and are thus considered unsupervised. Supervised methods for measuring distance exist which incorporate data labels and thereby exaggerate separation between data points of different classes. This approach tends to distort the natural structure of the data. Instead of following similar approaches, we leverage a popular algorithm used for making data-driven predictions, known as random forests, to naturally incorporate data labels into similarity measures known as random forest proximities. In this dissertation, we explore previously defined random forest proximities and demonstrate their weaknesses in popular proximity-based applications. Additionally, we develop a new proximity definition that can be used to recreate the random forest's predictions. We call these random forest-geometry- and accuracy-Preserving proximities or RF-GAP. We show by proof and empirical demonstration can be used to perfectly reconstruct the random forest's predictions and, as a result, we argue that RF-GAP proximities provide a truer representation of the random forest's learning when used in proximity-based applications. We provide evidence to suggest that RF-GAP proximities improve applications including imputing missing data, detecting outliers, and visualizing the data. We also introduce a new random forest proximity-based technique that can be used to generate 2- or 3-dimensional data representations which can be used as a tool to visually explore the data. We show that this method does well at portraying the relationship between data variables and the data labels. We show quantitatively and qualitatively that this method surpasses other existing methods for this task.

To my wife, Jenifer, and my children, Sawyer, Oakley, Maddison, and Navi.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Kevin R. Moon for his continued support over the past four years. Regarding me, he was starting with a blank slate so far as research experience is concerned. Under his supervision, I have benefited from much hands-on exploration of many areas of machine learning. I admire the patients he has had with me through this learning process. I am also grateful for Dr. Adele Cutler upon whose prior work much of this dissertation is founded. I am grateful for her continued support post-retirement and for her enthusiasm toward my continued work with random forests. I would like to thank Dr. Juergen Symanzik for his devotion to excellent storytelling via statistical visualization. His keen eye and detailed feedback proved critical for revising this dissertation and other prior works.

I am especially grateful for my wife, Jenifer. I have had her full support throughout virtually all of my educational endeavors. She has been there for me through good times and through bad times. Without her, obtaining my doctoral degree would not have been possible. I'm grateful for her dedication to our family. She has worked exceptionally hard to take care of our children, our home, and our happiness, while I was working long and late hours to finish my degree and financially support our family. I am truly indebted to her for her continued dedication to me and our children. I'm thankful for the love and support of my four beautiful children, Sawyer, Oakley, Maddison, and Navi. Sawyer also comments on the beautiful pictures I draw on the computer (usually in reference to my scatterplots). My girls would frequently take turns on my lap while at work on my computer. They would often wake up in the morning to find me working on my laptop in the front room. Their "interruptions" for snuggles were the best part of my day.

I am grateful for my parents and my siblings for their support thought my graduate studies. My brother, Lance, though unbeknownst to him, has been a great support through his own example as he recently finished his dissertation.

My preparatory education and Weber State University have played an important role in the completion of my doctoral degree. I'm grateful for my then cohort, especially Marshall Johnson and Jarrod Mao, the latter of which also came to Utah State. I'm grateful for the excellent faculty at Weber State, especially Dr. Steele, who pushed my limits in real analysis (pun intended), and Dr. Cora Neal who stated in reference to her probability and statistics course, "If you like this course, you should pursue a doctoral degree in statistics." I'm glad I took that general advice. I'm grateful for Dr. Paul Talaga who took me on as an adjunct instructor at Weber State, and for Dr. Sandra Fital-Akelbek for her continued support in that position.

I'm grateful for my Father in Heaven and my Saviour, Jesus Christ. Through him all things are possible.

Jake S. Rhodes

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Many methods in machine learning can be described as falling into one of two broad categories, supervised learning and unsupervised learning. In supervised learning, data have associated labels. For example, we may have images of cats and dogs respectively labeled as such. A primary goal in a supervised setting is to learn a function that defines a decision boundary between classes (classification) or predict some continuous numerical response (e.g., price of a house) given a collection of predictor variables. In unsupervised learning, the data do not have labels. Some common goals in unsupervised learning include clustering or grouping "similar" data together, pattern recognition, and dimensionality reduction (selecting or extracting feature variables). Unsupervised machine learning can be used for initial exploration analysis (e.g., via visualizations).

Many supervised and unsupervised machine learning algorithms rely on a notion of pair-wise distance. Citing a few examples, the k-nearest neighbors (k-NN) algorithm predicts class labels based on observational distance; nearby observations (based on some distance metric, often Euclidean or Mahalanobis) determine a test example's class based on a majority vote for classification or distance-weighted average for regression. The support vector machine (SVM) uses observational similarities in its optimization problem to determine an optimal separating hyperplane to discriminate classes [1]. Multidimensional scaling is an unsupervised algorithm that optimizes a stress function to best preserve pair-wise distances in a lower-dimensional Euclidean space [2]. Isomap constructs a k-nearest neighbor graph and computes geodesic distances from the shortest path between each pair of the graph's nodes [3]. Uniform manifold approximation and projection (UMAP) uses distances to generate a weighted graph from which a lower-dimensional manifold is estimated [4]. Algorithms that require the "kernel trick" (e.g., SVM) apply a kernel function to pair-wise distances, generating an inner-product from a higher-dimensional space to augment non-linear decision-boundaries.

Distances are computed based on the data features which are often numeric, but may also be categorical, images, text, audio, graph-based, or otherwise. Some common distances include Minkowski (Euclidean and Manhattan distances being special cases of this), Mahalanobis, Wasserstein (or earth-movers distance, which is often associated with generative adversarial networks (GANs) [5]), Hamming, which is often used for text strings, and Jaccard (for set dissimilarities). Each of these metrics is unsupervised; they do not incorporate observation labels in their construction. Labels can provide additional information about the data distribution. A common assumption in statistics is that the data is independent and identically distributed. However, data pertaining to different classes (e.g., have different labels) often follow different distributions. The incorporation of label-based auxiliary information in the metric construction can aid in the recovery of class-conditional distributions so long as it is done in a way that does not disrupt the data structure. Labels can also be used to determine which features are relevant as discriminators and thus improve distance metrics under noisy conditions.

In many domains, expert knowledge is required to generate labeled data. In some cases, labeled examples may be scarce or expensive to generate; such is the case in the medical field when experts are required to make a diagnosis based on a medical image. However, when labels are available, they can provide additional insight into the collected data. Formulations of supervised distance metrics have been introduced in various contexts. Many of these use class labels to increase the value of a known distance metric (e.g., Euclidean) between observations of opposing classes [6,7]. Similarity measures between observations of a shared class may also be increased. For example, the authors of [8] use both a class-based similarity and dissimilarity measure to augment a Gaussian kernel used in a supervised variant of non-negative matrix factorization (NMF). Some algorithms attempt to learn a metric that may be used to increase class discrimination. For example, neighborhood components analysis (NCA) learns a Mahalanobis distance measure which maximizes the leave-one-out (LOO) classification accuracy based on a nearest-neighbor classifier [9].

Current methodologies for generating supervised distance metrics suffer in many of the above-mentioned applications (such as dimensionality reduction and visualization approaches) due to their excessively discriminating nature. In some cases, supervised metrics perfectly separate classes when such separation would not be possible without consideration of the labels. In a sense, direct and explicit use of labels to exaggerate distance overemphasizes the label's importance over that of the feature variables' in determining a metric. Thus, the direct use of labels in the metric construction distorts the data's inherent structure and artificially increases the discrimination between classes. We may analogize this with a classification task. In a classification problem, a function learns a decision boundary that minimizes some objective loss. The function assigns observations a class based on the observation's features variables; labels are not used as features but are objects of the prediction. Directly using labels in the construction of the class assignment function defeats the purpose of the learner. Similarly, a supervised distance metric should not directly incorporate the class labels, especially if the purpose of the learned metric is for downstream classification.

Observation labels can be influential in the construction of a similarity matrix by means of a random forest classifier or regressor [10]. These similarities, known as random forest proximities, were first introduced by Leo Breiman and have since been adapted and used in a variety of applications [11–15]. In this dissertation, we present a new random forest proximity measure that more accurately reflects the random forest's learning and demonstrates that these proximities improve a variety of proximity-based applications. We then derive a new random forest proximity-based visualization algorithm that outperforms existing supervised and unsupervised methods in regard to visualizing feature relevance.

To better understand the construction of pairwise similarities from a random forest, we visit the random forest algorithm in Chapter 2, discussing the implications of the derived similarity or proximity measure and providing background and discussion on the implications and alternative definitions of the random forest proximities. We introduce our novel definition of the random forest proximities in Chapter 3 which preserves the data-geometry learned by the random forest. These random forest geometry- and accuracy-preserving proximities (RF-GAP) perfectly preserve the random forest's predictions when serving as weights in a nearest-neighbor predictor model. Since the random forest's learning is preserved, applications that use these proximities will more accurately reflect the learned information. This work has been submitted for publication and is currently under review (see [16]). We compare RF-GAP proximities with existing common applications including data imputation, visualization, and outlier detection, and show that each of these methods is improved using RF-GAP proximities. More information about these applications can be found in Chapter 4. A demonstration of the `R` package, `rfgap`, used to run these applications can be found in Chapter 5. Our main contributions provided in [16] include:

- Defining a new RF proximity (RF-GAP) which preserves the RF-learned data geometry
- Proving that RF-GAP proximities perfectly reconstruct RF predictions when used as weights in a nearest-neighbor (NN) predictor
- Showing that RF-GAP proximities improve data imputation when compared to existing proximities
- Empirically demonstrate that RF-GAP improves common proximity-based applications, including visualization and outlier detection

In Chapter 6, we introduce a supervised variation of a manifold-learning approach to dimensionality reduction via random forest proximities. Existing supervised dimensionality reduction algorithms are not optimized for visualization. Most of these adapt commonly used distance metrics (such as Euclidean distance) or kernels (e.g., Gaussian) using labels to accentuate the distance between classes. In doing so, the natural data geometry is distorted.

Our approach indirectly uses data labels to generate a kernel matrix in the form of random forest proximities. We apply these proximities in a dimensionality reduction method optimized for visualization, rather than class separation. We show that the resulting embedding naturally captures variable importance in low dimensions, and discuss a means of quantifying the quality of supervised embeddings. Our method uses advances from the PHATE (Potential of Heat-diffusion for Affinity-based Transition Embedding) algorithm [17] which are applied in the Diffusion maps' process [18], resulting in the algorithm we call RF-PHATE. This work was published in the IEEE Statistical Signal Processing Workshop (2021) [19]. Our main contributions to the paper include:

- Constructing a novel approach to supervised dimensionality reduction based on diffusion and random forest proximities.
- Defining a new method for quantifying the fit of low-dimensional embeddings in a supervised context.
- Demonstrating the utility of this method in exploratory analysis.

Finally, we provide a conclusion and statement about future works in Chapter 7.

CHAPTER 2

Random Forests and Proximities

More than twenty years after the original publication, random forests are still used in many scientific fields with great success. Citing some recent examples, Benali et al. [20] demonstrated superior solar radiation prediction using random forests over neural networks. The work of [21] showed that random forests produced the best results with the lowest standard errors in classifying error types in rotating machinery when compared with more commonly used models in this application, such as the support vector machine (SVM) and neural networks. Random forests have been used to predict surface water salinity [22], assess shear strength of soft clays [23], analyze building structure impact on CO_2 emissions [24], model the heterogeneity of water quality [25], predict COVID-19 patient health [26], estimate spatio-temporal COVID-19 cases [27], forecast infectious diarrhea [28], map landslide susceptibility [29], predict the status of cardiovascular disease [30], predict deforestation rates [31], estimate nanofluid viscosity [32], asses credit in rural locations [33], classify activities via wearable sensors [34], predict heavy-metal distribution [35], predicting RNA pseudouridine sites [36], and classifying activities from wearable sensors [34]. In [37], random forests were used to discriminate between antioxidant treatments applied to Raman spectra collected from cells exposed to diesel exhaust particles (DEPs).

In addition to high predictive accuracy, random forests provide a number of benefits to the user [11]. Random forests:

1. Handle problems in both regression and classification
2. Train quickly (especially when compared to other state-of-the-art methods, such as neural networks)
3. Are relatively insensitive to tuning parameters
4. Provide an estimate of generalization error

5. Work for both low- and high-dimensional problems
6. Are trivially parallelizable
7. Are insensitive to monotonic transformations
8. Handle irrelevant variables (noise variables)
9. Handle mixed-variable types (e.g., continuous, categorical)
10. Are capable of handling missing values
11. Can impute missing values
12. Model non-linear interactions
13. Are relatively robust to outliers in the predictor space
14. Perform various data analysis types (survival analysis, unsupervised learning)
15. Provide a natural similarity measure

2.1 Random Forest Construction

Prior to Leo Breiman's work on random forests, he assisted in the development of classification and regression trees (CART) [38] which provide the framework for the random forest. A random forest is an ensemble method that uses decision trees as base learners for its construction. We first review the formulation of the decision trees (base-learners) to understand the random forest.

Consider the training data space $\mathcal{M} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots (\mathbf{x}_N, y_N)\}$ where each $\mathbf{x}_i \in \mathcal{X}$, is a d-dimensional vector of predictor variables with corresponding response, $y_i \in \mathcal{Y}$. A decision tree recursively partitions $\mathcal{X}$ by its predictor variables. The full dataset is considered at the root node. Each partitioning split forms two daughter nodes. Nodes which are not split are considered terminal. The collection of terminal nodes defines the final partition, resulting in the decision space.

For a given continuous predictor variable, split-points are considered between each pair of consecutive observed values. Usually, these split-points are the midpoints between values, though any point would do. Observations with the feature value less than the split-point are sent to the left daughter node while remaining observations are sent to the right daughter node. See a depiction in Figure 2.1. Considering a categorical predictor variable X_j with finite set of categories $G_j = \{g_{j,1}, g_{j,2}, \cdots, g_{j,c}\}$, splits are determined using a subset $S \in G_j$. Observations containing the included categories are sent to the left daughter node, excluded to the right. (Note that the determination of left- or right-daughter nodes is by a convention that is not shared by all software packages. Additionally, not all decision tree packages are capable of handling categorical variables, such as that from the commonly used Python library, scikit-learn [39]).

Here we discuss the criteria for the splits. The problem formulation depends on whether the response variable is categorical or continuous. If the response variable is categorical (i.e. a classification problem) splits are determined to maximize the class purity of the resulting daughter nodes. Purity is most frequently measured using the Gini index, $C = \sum_{k \neq k'}^{K} \hat{p}_k \hat{p}_{k'}$, where $\hat{p}_k$ is the proportion of observations in class k. This was used in the original CART paper [38]. Other criteria may be used, such as entropy or misclassification rate [39]. For a regression problem, a measure of goodness-of-fit is used. Here, the mean-squared error (MSE) is the typical criterion: $C = \frac{1}{n} \sum_{i=1}^{n} (y_i - \bar{y})^2$, where $\bar{y}$ is the mean response value. The respective criteria are assessed in the resulting daughter nodes, weighted by the number of observations to the left or right of each split-point.

The binary partition is performed and observations are sent to one of the two resulting daughter nodes. At each node, the partitioning algorithm is repeated until some stopping criteria are met, resulting in the set of terminal nodes. See Algorithm 1. Examples of stopping criteria include node purity, a predefined tree height, or a minimum number of node observations. The final partition (the terminal nodes) forms the basis for the decision space for the problem. For classification, a non-weighted majority vote determines the prediction of a new observation. For regression, an average response value is used. Note

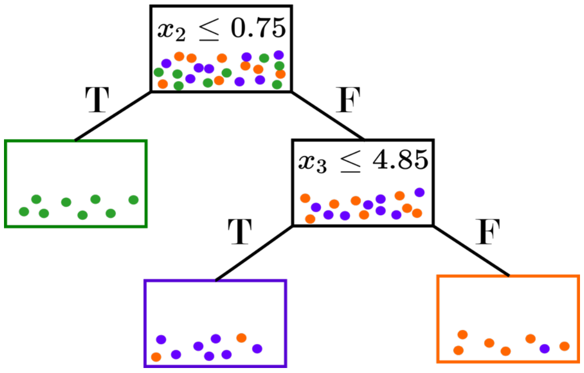


Fig. 2.1: A depiction of a simple decision tree. Here we have three classes denoted by the colors, orange, purple, and green. The first split is determined by the second variable of the dataset (x_2), at the split-point 0.75. The second split is determined by the third variable at the split-point 4.85. Node purity is increased with each split.

that for standalone decision trees, the trees are often pruned and not grown to purity, which often results in overfitting. In contrast, trees in a random forest are typically not pruned.

```
Algorithm 1: Binary Recursive Decision Tree
Input: A set $\mathcal{M} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, each $\mathbf{x}_i \in \mathbb{R}^d$ with label $y_i$.

    1. Begin with all points $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
    2. Find the best split across all possible binary splits on each dimension
    3. Split the data into two daughter nodes according to the previous step
    4. Repeat until a stopping criteria is met
    5. The predicted label for an observation $\mathbf{x}$ which resides in terminal node $\ell$ is given by:
        • $\hat{h}(x) = \bar{y}_\ell = \frac{1}{n}\sum_{i=1}^{n} y_{\ell_i}$ for regression
        • $\hat{h}(x) = \arg\max_y \sum_{i=1}^{n} I(y_{\ell_i} = y)$ for classification
        • where $y_{\ell_1}, \ldots, y_{\ell_n}$ span the observed labels in terminal node $\ell$ and $I(\cdot)$ is the
          0-1 indicator function
```

Random forests are comprised of an ensemble of randomized decision trees. Consider again the dataset $\mathcal{M} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots (\mathbf{x}_N, y_N)\}$. Let $\mathcal{T}$ be the set of decision trees in the forest. For each decision tree $t \in \mathcal{T}$, a bootstrap sample of size N is taken to train the tree. Observations that are part of the bootstrap sample are known as in-bag observations, while those which are left out are considered out-of-bag (OOB). Additional randomization is also used during the partitioning (splitting) process. In a decision tree, all possible split-points are considered across all of the feature variables and the splits are determined as described above. Decision trees within a random forest use only a subset of size $m < d$ feature variables to determine the split-points at each splitting node. This two-part randomization reduces the correlation between the trees which improves its general predictive ability [40]. The combined set of terminal nodes forms the decision space for the random forest. Predictions for a new observations class (or continuous outcome for regression) are determined using a majority vote across all trees (for classification) or an unweighted average for a continuous response (regression). See Algorithm 2.

Algorithm 2: Breiman's Random Forest

```
Input: A set $\mathcal{M} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$, with $\mathcal{X} = \{\mathbf{x_1}, \cdots, \mathbf{x_N}\}$ and each $\mathbf{x}_i$
       having label $y_i$.
For each $t$ with index $1, \cdots, |\mathcal{T}|$:
  1. Take a bootstrap sample $\mathcal{X}_t$ of size $N$ from $\mathcal{X}$.
  2. Use the data $\mathcal{X}_t$ to train a decision tree:
     (a) Start with all of $\mathcal{X}_t$
     (b) Find the best binary split across a random selection of $m < d$ dimensions
     (c) Split the data into two daughter nodes according to the previous step
     (d) Repeat until a stopping criterion is met
  To make a prediction on $\mathbf{x}$:
     • $\hat{f}(x) = \frac{1}{T} \sum_{t=1}^{\mathcal{T}} \hat{h}_t(x)$ for regression
     • $\hat{f}(x) = \arg\max_y \sum_{t=1}^{\mathcal{T}} I\left(\hat{h}_t(x) = y\right)$ for classification
```

2.2 Random Forest Proximities

Random forest predictions are determined by voting (or averaging values) across the trees. Observations which frequently reside in the same terminal nodes are similar to each other with respect to variables important in the decision process, while observations which never reside together are distant from each other in the decision space. Thus, the terminal nodes of the trees in the random forest can be used to derive a similarity measure with regards to splitting variables and the overall supervised task. In the following sections, we formally define these similarities or proximities, discussing applications and alternative proximity versions. Importantly, we introduce a novel random forest proximity measure which will be formally defined in Chapter 3, discussing its implications and advantage in proximity-based applications.

2.2.1 The Original Proximities

The space of terminal nodes can be used to define a similarity measure between observations. The recursive partitioning of decision trees based on feature variables organizes observations according to split-points across variables which are most useful at partitioning data as measured by class purity or goodness of fit. Observations which frequently share terminal nodes are similar to each other with respect to important variables relative to the supervised problem. Observations which never reside together are distant from each other in the same regard. Thus, the frequency in which observations reside in the same terminal nodes can thus be used as a measure of the closeness of the observations in the context of the supervised task.

The random forest proximities form a symmetric, $N \times N$ affinity matrix ($\mathcal{P}$) with ones across the main diagonal. The proximity values range from 0 to 1 as they are simply proportions of co-occurrences of observations; 0 means the two observations are dissimilar enough to never share a terminal node, 1 being the case where they always reside together. Thus, $1 - (\mathcal{P})$ may be viewed as a squared dissimilarity measure, where the dissimilarity takes into account the supervised task. It is important to note that this dissimilarity measure naturally captures variable importance. We see that, unlike unsupervised distances (e.g., Euclidean), random forest proximity-based dissimilarities use additional information from the class labels. While two observations may be considered somewhat similar in many dimensions of a Euclidean space, they may have a low proximity value if they differ in variables important to the supervised task. Random forests provide the added benefit that they can encode these dissimilarities using both continuous and categorical variables, whereas most distance metrics require numerical predictor variables. Additionally, proximity-based dissimilarity is relatively robust to noise variables. Of the m variables randomly selected at each split, noise variables are not likely to be used to determine splits whenever a meaningful variable is present in the subset. These proximities may thus encode meaningful relationships in high-dimensional data (such as RNA-sequencing data [41], for example).

The decision trees in random forests are usually grown until the terminal nodes are pure (of one class). As a natural consequence, in-bag observations (observations that were used in training a given decision tree) of opposing classes will necessarily reside in different terminal nodes. In a given decision tree, the probability of an observation being in-bag is $1 - \frac{1}{e} \approx \frac{2}{3}$. The original proximity definition thus tends to exaggerate class separation. An example displaying this behavior can be seen in Figure 2.2. In this figure, multidimensional scaling (MDS) was applied to the original proximities constructed from a random forest trained on the Sonar dataset [42] which achieved an OOB error rate of 15.85%. The two-dimensional representation portrays the dataset as nearly perfectly linearly separable, though this is clearly not reflected in the random forests error rate.

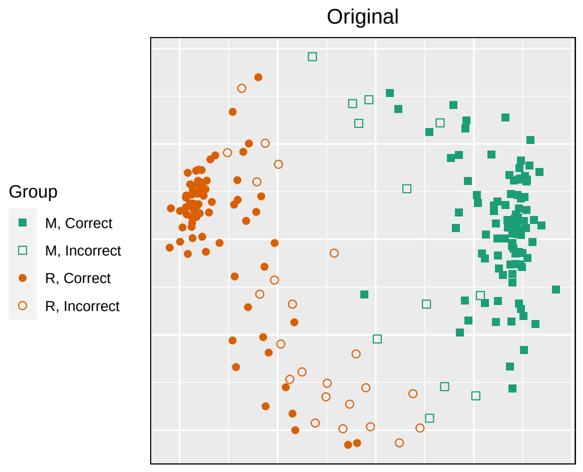


Fig. 2.2: MDS applied to the original random forest proximities. The OOB error rate was 15.85%, however, the lower-dimensional plot appears to be nearly linearly separable. Here we see the exaggerated separation captured by the original proximities.

The original definition also appears to capture a signal in the noise. Consider the example in Figure 2.3. Here we construct a bivariate normal distribution and randomly assign points to one of three classes. Multidimensional scaling applied to the proximities constructed on this dataset appears to show a partial separation of the classes which clearly should not exist (see Figure 2.3).

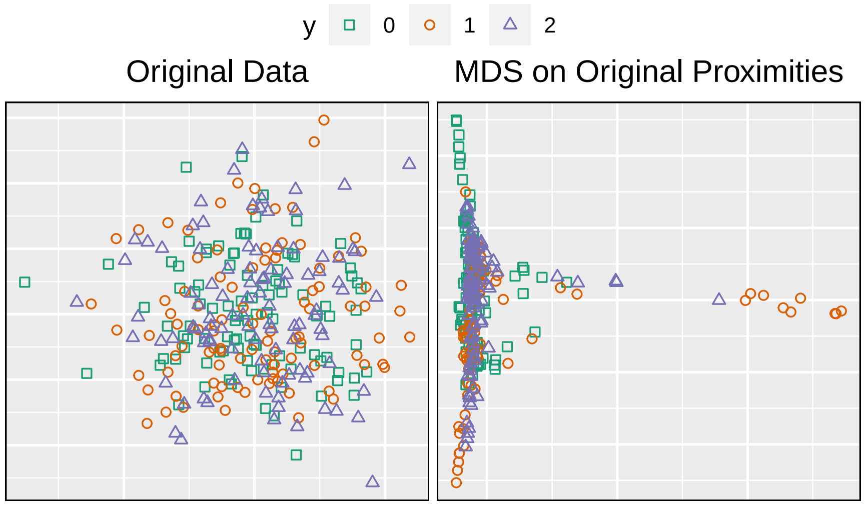


Fig. 2.3: A random sample of 300 points generated from a bivariate normal distribution was randomly assigned one of three classes. (a) shows the original bivariate normal data. In (b), multidimensional scaling was applied to the random forest proximities using the original definition. Here, we can see that the proximities have some tendency to naturally separate classes where no true separation should exist. This does not occur using other proximity definitions which will be described in Chapter 3.

Another proximity definition has been proposed to overcome this apparent weakness [12, 40]. Instead of using all training examples in each tree for the proximities' construction, the alternative definition only uses OOB samples. This definition overcomes the weakness of exaggerated separation and diminishes the class bias in the proximities.

We can see in Figure 2.4 that the exaggerated class separation is no longer an issue, but the relative positions of the scaled observations do appear to be influenced by additional noise.

Additional random forest proximities have also been defined and will be described and compared in the following chapter.

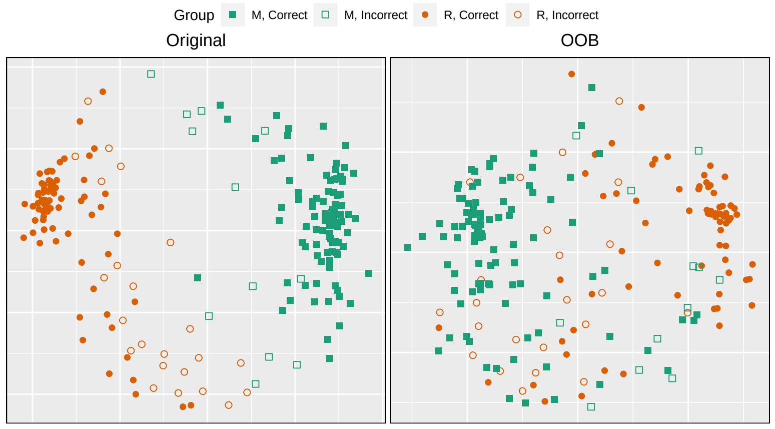


Fig. 2.4: (a) shows the MDS plot using the original definitions. In contrast, (b) uses the OOB definition (Definition 2). Here, we no longer see exaggerated class separation, but the plot appears to be much noisier.

2.3 Random Forest Geometry- and Accuracy-Preserving Proximities

The work of Lin and Jeon suggests that random forests may be viewed as a nearest-neighbor classifier with an adaptive bandwidth [43]. Here, they define voting points as in-bag observations in the shared terminal node of a test example and show that the test example's predicted label is defined by a weighted sum of the voting-point labels (they only explored the regression task here). To this end, a random forest's predictions may be found using a weighted sum (for continuous responses) or a weighted majority average (for categorical responses). However, Lin and Jeon's approach only works for new, test observations. If applied to training observations, the predicted labels do not necessarily match the true labels.

With their paper as inspiration, we define proximities as weights that, when used in a nearest-neighbor classifier, perfectly reconstruct the random forest's predictions. Empirical results show that the original proximities, when rescaled to serve as weights, do not preserve the forest's predictions. This is corroborated by the work of Feng and Baumgartner, who compared random forest proximities to the random forest predictive errors in regression, classification, and survival analysis settings [44]. The OOB definition also does not meet this criterion, nor do those defined in [45, 46].

2.4 Conclusion

In this chapter, we provided enough details of the random forest construction to enable the reader to sufficiently understand the varied random forest proximity definitions to follow. We provided high-level insights regarding the most commonly used proximities. In Chapter 3, we define a new random forest proximity measure that preserves the random forest predictions and thus encapsulates the data-geometry learned by the random forest. We name the proximity random forest-geometry- and accuracy-preserving proximities (RF-GAP) and show this preservation empirically and by proof. We compare these proximities with other random forest proximity constructions and show that RF-GAP proximities provide an advantage in many applications.

CHAPTER 3

RANDOM FOREST GEOMETRY- AND ACCURACY-PRESERVING PROXIMITIES

Random forests have a natural extension to produce pair-wise proximity (similarity) measures determined by the partitioning space of the decision trees which comprise them. Unlike unsupervised similarity measures, random forest proximities incorporate variable importance relative to the supervised task as these variables are more likely to be used in determining splits in the decision trees [11]. Ideally, random forest proximities should define a data geometry that is consistent with the learned random forest; that is, the random forest predictive ability should be recoverable from the proximities. In this case, applications involving random forest proximities, such as data visualization, can lead to improved interpretability of the random forests specifically, and more generally the data geometry relative to the supervised task.

One way to test for consistency is to compute a proximity-weighted predictor where a data point's predicted label consists of a proximity-weighted sum of the labels of all other points. This predictor should match the random forest prediction if the proximities are consistent with the random forest. However, under Breiman's original definition, the proximity-weighed predictions do not match those of the random forest, even when applied to the training data (see Section 3.3). Thus this definition does not capture the data geometry learned by the random forest, limiting its potential for improved interpretability of the random forest.

We define a new random forest proximity measure called random forest-geometry- and accuracy-preserving proximities (RF-GAP) that defines a data geometry such that the proximity-weighted predictions exactly match those of the random forest for both regression and classification. Under our definition, an out-of-bag observation's proximities are computed via in-bag (training) observations. That is, the sample used to generate a decision tree also generates the proximities of out-of-bag observations (observations not used to construct the tree). The proximities of an out-of-bag observation are the mean reciprocal of the number of in-bag observations in its shared terminal nodes. We prove the equivalence between the proximity-weighted predictions with those of the random forest and demonstrate this empirically.

3.1 Random Forest Proximities

Here we provide the details and definitions of several existing random forest proximities followed by RF-GAP that preserves the geometry learned by the random forest.

Let $\mathcal{M} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots (\mathbf{x}_N, y_N)\}$ be the training data where each $\mathbf{x}_i \in \mathcal{X}$, is a d-dimensional vector of predictor variables with corresponding response, $y_i \in \mathcal{Y}$.

We will use the following notation (see Fig. 3.1 for a visual example):

- $\mathcal{T}$ is the set of decision trees in a random forest with $|\mathcal{T}| = T$.
- $B(t)$ is the multiset of indices in the bootstrap sample of the training data that is randomly selected to train the tree $t \in \mathcal{T}$. Thus $B(t)$ contains the indices of the in-bag observations.
- $O(t) = \{i = 1, \ldots, n | i \notin B(t)\}$. Thus $O(t)$ is the set of indices of the training data that are not contained in $B(t)$. $O(t)$ is often referred to as the out-of-bag (OOB) sample.
- $S_i = \{t \in \mathcal{T} | i \in O(t)\}$. This is the set of trees in which the observation i is OOB.
- $v_i(t)$ contains the indices of all observations that end up in the same terminal node as $\mathbf{x}_i$ in tree t.
- $J_i(t) = v_i(t) \cap B(t)$. This is the set of indices in $v_i(t)$ that correspond with the in-bag observations of t. I.e. these are the observations that are in-bag and end up in the same terminal node as $\mathbf{x}_i$.

Each decision tree t in a random forest is grown by recursively partitioning or splitting the bootstrap sample into nodes, where splits are determined across a subset of feature

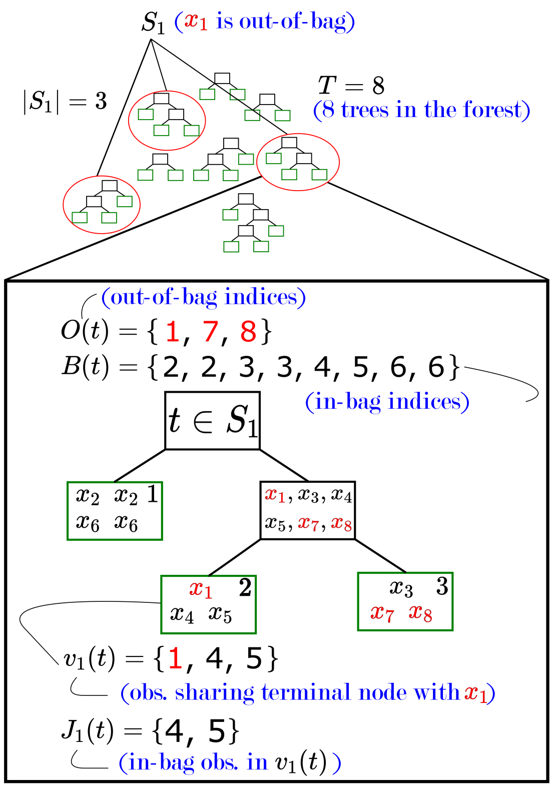


Fig. 3.1: An example of a random forest and notation with regards to a particular observation x_1. The red-encircled trees are those in which x_1 is out of bag, making up the set of trees S_1. A particular tree in S_1 is exhibited. The out-of-bag indices for the tree are given in red ($i \in O(t)$), while the in-bag indices ($i \in B(t)$) are shown in black. The indices of observations residing in the same terminal node as x_1 is given by the set $v_1(t)$. $J_1(t)$ gives the in-bag observation indices in the terminal node $v_1(t)$.

variables to maximize purity (classification) or minimize the mean squares of the residuals (regression) in the resulting nodes. This process repeats until a stopping criterion is met. For classification, splits are typically continued until nodes are pure or of a single one class. For regression, a common stopping criterion is a predetermined minimum node size (e.g., 5). The trees in random forests are typically not pruned. OOB samples are commonly used to provide an unbiased estimate of the forest's generalization error rate [10].

The strength of the random forest is highly dependent on the predictive power of the individual decision trees (base learners) and on the low correlation between the decision trees [40, 47]. In addition to bootstrap sampling, further correlational decrease between trees is ensured by selecting a random subset of predictor variables at each node for split optimization. The number of variables to be considered is designated by the parameter `mtry` in many random forest packages [12, 48]. The resulting terminal nodes partition the input space $\mathcal{X}$. This partition is often used in defining random forest proximities as in Breiman's original definition:

Definition 1 (Original Random Forest Proximity [47]). *The random forest proximity between observations i and j is given by:*

$$p_{Or}(i,j) = \frac{1}{T}\sum_{t=1}^{T} I(j \in v_i(t)),$$

where T is the number of trees in the forest, $v_i(t)$ contains the indices of observations that end up in the same terminal node as $\mathbf{x}_i$ in tree t, and $I(.)$ is the indicator function. That is, the proximity between observations i and j is the proportion of trees in which they reside in the same terminal node, regardless of bootstrap status.

Definition 1 (the original definition) does not capture the data geometry learned by the random forest as it does not take an observation's bootstrap status (whether or not the observation was used in the training of any particular tree) into account in the proximity calculation: both in-bag and out-of-bag samples are used. In-bag observations of different classes will necessarily terminate in different nodes, as trees are grown until pure. Thus this produces an over-exaggerated class separation in the proximity values.

Despite these weaknesses, this definition has been used for outlier detection, data imputation, and visualization. However, these applications may produce misleading results as this definition tends to overfit the training data, quantified by low error rates as proximity-weighted predictors. One attempt to overcome this issue redefines the proximity measure between observations i and j using only trees in which both observations are out-of-bag (OOB proximities):

Definition 2 (OOB Proximity [12, 40]). *The OOB proximity between observations with indices i and j is given by:*

$$p_{OOB}(i,j) = \frac{\sum_{t \in S_i} I(j \in O(t) \cap v_i(t))}{\sum_{t \in S_i} I(j \in O(t))},$$

where $O(t)$ denotes the set of indices of the observations that are out-of-bag in tree t, and S_i is the set of trees for which observation i is out-of-bag. In other words, this proximity measures the proportion of trees in which observations i and j reside in the same terminal node, both being out-of-bag.

The OOB definition does not directly use the in-bag training examples in the proximity construction, leading to proximities that are, in effect, built by test points (OOB observations have no part in the classifier's training), rather than training examples. In a given decision tree, the probability of an observation being OOB is $\frac{1}{e}$. Given the independence of observational selection in the bootstrap process (the sampling is done with replacement), a given pair of observations being OOB is $\frac{1}{e^2} \approx \frac{1}{9}$. Thus, we need a larger number of decision trees to achieve stability in the proximities following the OOB definition.

This definition is currently used in the `randomForest` [12] package by Liaw and Wiener in the `R` programming language [49]. It may have been inadvertently used in papers that used this package but none made explicit mention of the use of OOB observations in building proximities. The OOB proximities define a similarity measure that does not reflect the nuances of the training data, unlike the original proximities, but still do not match the random forest predictions when serving as weights. We find that this definition generally produces higher error rates as a proximity-weighted predictor (when compared to the random forest's OOB error rate; see Section 3.3).

A few alternative random forest proximity measures beyond Definitions 1 and 2 have been proposed previously. In [45], the authors define a proximity-based kernel (PBK) which accounts for the number of branches between observations in each decision tree, defining the proximity between i and j as $p_{PBK}(i,j) = \frac{1}{T}\sum_{t=1}^{T} \frac{1}{e^{w \cdot g_{ijt}}}$, where T is the number of trees in the forest, w is a user-defined parameter, and g_{ijt} is the number of branches between observations i and j in tree t. g is defined to be 0 if the observations reside in the same terminal node. The proximity quality was quantified using the classification accuracy when applied as a kernel in a support-vector machine. This definition showed some improvement over the original definition (Definition 1) only when considering very small numbers of trees (5 or 10). However, PBK is computationally expensive as all pair-wise branch distances must be computed within each tree. This is not an issue for small numbers of trees, but typically the number of trees in a random forest is measured in hundreds or thousands (`randomForest` [12] has a default of 500 trees). Additionally, this method adds a user-defined, tunable parameter which adds to its complexity.

The authors in [50] describe an approach for computing random forest proximities in the context of a larger class of Random Partitioning Kernels. While most random forest proximities are determined primarily through associations within terminal nodes, this approach selects a random tree height and partitions the data based on this higher-level splitting. The authors do not compare with other proximity definitions (nor do they frame their work in the context of random forest proximities) but they compare this random forest kernel to other typical kernels (linear, radial basis function (RBF), etc.) using a log-likelihood test. The random forest kernel outperformed the others in most cases and the authors visually demonstrated their kernel using 2D PCA plots. The code for this approach is not publicly available.

Cao et al. introduced two random forest dissimilarity measures which are used in the context of multi-view classification [46]. The first measure (denoted RFDisNC) weights the proximity values by the proportion of correctly-classified observations within each node, accounting for both in- and out-of-bag observations. The second (RFDisIH) is based on instance hardness. Euclidean distances between observations at each terminal node are calculated (using only feature variables which were used as splitting variables leading to the terminal node, to avoid the curse of dimensionality) and used as weights as a part of the dissimilarity measure. Given this distance, they use k-Disagreeing Neighbors (kDN) in the formulation of the dissimilarity measure:

$$d_t(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} kDN(\mathbf{x}_j), & \text{if } v_i(t) = v_j(t) \\ 1, & \text{otherwise} \end{cases},$$

where

$$kDN\left(\mathbf{x}_i\right) = \frac{\left|\left\{\mathbf{x}_j : \mathbf{x}_j \in kNN\left(\mathbf{x}_i\right), y_j \neq y_i\right\}\right|}{k},$$

where $k - NN(\mathbf{x}_i)$ is the set of the k-nearest neighbors of $\mathbf{x}_i$ in the training data.

The use of k-DN gives a notion of difficulty in classifying a particular observation. In the multiview problem, the dissimilarities from different views are averaged before classification. The authors showed that RFDisIH performed better overall on classification tasks compared with other multi-view methods. However, RFDisIH was not compared with other random forest proximities in their commonly-used applications (e.g., visualization or imputation). A similarity measure can be constructed from RFDisIH as RFProxIH $= 1-$ RFDisIH.

While these alternative definitions have shown promise in their respective applications, their connection to the data geometry learned by the random forest is not clear. In contrast, we present a new definition of random forest proximities that exactly characterizes the random forest performance on both in-bag and out-of-bag samples. The design of the new proximities purposefully models them to match the random forest predictions and follows the same schema for a classification learning problem. For a typical learner, a training set is used to build the model, which is subsequently tested on a validation set or previously unseen observations. Our proximities are similarly constructed. Training examples (in-bag observations) are used to construct the proximity values to unseen (OOB) observations. In a given decision tree, the quantity of in-bag observations in an OOB point's terminal node determines the weight of the node in the proximity construction. Using training points to act on test points follows typical classifier behavior, and this particular weighting leads to the random forest predictions for both the training and test sets.

Definition 3 (Random Forest-Geometry- and Accuracy-Preserving Proximities)**.** *Let $B(t)$ be the multiset of (potentially repeated) indices of bootstrap (in-bag) observations. We define $J_i(t)$ to be the set of in-bag observations which share the terminal node with observation i in tree t, or $J_i(t) = B(t) \cap v_i(t)$ with cardinality $|J_i(t)|$. Then, for given observations i and j, their proximity measure is defined as:*

$$p_{GAP}(i,j) = \frac{1}{|S_i|} \sum_{t \in S_i} \frac{I\left(j \in J_i(t)\right)}{|J_i(t)|}.$$

That is, considering only trees for which observation i is out-of-bag, the proximity between i and j is the average proportion of in-bag observations in the shared terminal node of i and j over all trees where i is out of bag and j is in bag.

We show in Section 3.2 that the random forest OOB prediction (and thus the generalization error rate) is exactly reproduced as a weighted sum (for regression) or a weighted-majority vote (for classification) using the proximities in Definition 3 as weights. Thus, this definition characterizes the random forest's predictions, keeping intact the learned data geometry. Subsequently, applications using this proximity definition will provide results that are truer to the random forest from which the proximities are derived. Importantly, the overfitting present in the original definition is overcome by the RF-GAP construction as in-bag to in-bag pairs are not used in their construction.

3.2 Random Forests as Proximity-Weighted Predictors

Here we show that the random forest prediction is exactly reproduced as a weighted sum (for regression) or a weighted-majority vote (for classification) using RF-GAP as weights. We first show that for a given observation, the proximities are non-negative and sum to one. In contrast, the proximities in Definitions 1 and 2 must be row-normalized to sum to one. Note that we require that $p_{GAP}(i,i) = 0$ for the proximities to sum to one, although the exact value for $p_{GAP}(i,i)$ does not matter in practice as it is not considered in the proximity-weighted prediction.

Proposition 1. *prop:weights Defining $p_{GAP}(i,i) = 0$, the random forest proximities (under Definition 3) are non-negative and $\sum_{j=1}^{N} p_{GAP}(i,j) = 1$.*

Proof. It is clear from the definition that $p_{GAP}(i,j) \geq 0$ for all i, j. The sum-to-one property falls directly from the definition:

$$\begin{aligned}
\sum_{j=1}^{N} p_{GAP}(i,j) &= \sum_{j=1}^{N} \frac{1}{|S_i|} \sum_{t \in S_i} \frac{I\left(j \in J_i(t)\right)}{|J_i(t)|} \\
&= \frac{1}{|S_i|} \sum_{t \in S_i} \frac{1}{|J_i(t)|} \sum_{j=1}^{N} I(j \in J_i(t)) \\
&= \frac{1}{|S_i|} \sum_{t \in S_i} \frac{1}{|J_i(t)|} |J_i(t)| \\
&= \frac{1}{|S_i|} \sum_{t \in S_i} 1 \\
&= 1.
\end{aligned}$$

□

This proposition allows us to directly use RF-GAP as weights for classification or regression. We show that the proximity-weighted prediction under Definition 3 matches the random forest OOB prediction, giving the same OOB error rate in both the classification and regression settings. The OOB error rate is typically used to estimate the forest's generalization error and quantify its goodness of fit, this indicates that RF-GAP accurately represents the geometry learned by the random forest.

Theorem 1 (Proximity-Weighted Regression)**.** *For a given training data set*

$\mathcal{S} = \{(\mathbf{x}_1, y_1) \dots (\mathbf{x}_N, y_N)\}$*, with* $y_i \in \mathbb{R}$*, the random forest OOB regression prediction is exactly determined by the proximity-weighted sum using RF-GAP (Definition 3).*

Proof. For a given tree, t, and $i \in O(t)$, the decision tree predictor of y_i is the mean response of the in-bag observations in the appropriate terminal node. That is,

$$\hat{y}_i(t) = \frac{1}{|J_i(t)|} \sum_{j \in J_i(t)} y_j.$$

The random forest prediction, $\hat{y}_i$, is the mean response over all trees for which i is out of bag. That is,

$$\begin{aligned}\hat{y}_i &= \frac{1}{|S_i|}\sum_{t\in S_i}\hat{y}_i(t)\\ &= \frac{1}{|S_i|}\sum_{t\in S_i}\frac{1}{|J_i(t)|}\sum_{j\in J_i(t)} y_j.\end{aligned}$$

The proximity-weighted predictor, $\hat{y}_i^p$, is simply the weighted sum of responses, $\{y_j\}_{j\neq i}$.

$$\begin{aligned}\hat{y}_i^p &= \sum_{j=1}^{N} p_{GAP}(i,j)y_j\\ &= \sum_{j=1}^{N}\left\{\frac{1}{|S_i|}\sum_{t\in S_i}\frac{I(j\in J_i(t))}{|J_i(t)|}\right\}y_j\\ &= \frac{1}{|S_i|}\sum_{t\in S_i}\frac{1}{|J_i(t)|}\sum_{j=1}^{N} I(j\in J_i(t))y_j\\ &= \frac{1}{|S_i|}\sum_{t\in S_i}\frac{1}{|J_i(t)|}\sum_{j\in J_i(t)} y_j\\ &= \hat{y}_i.\end{aligned}$$

□

Theorem 2 (Proximity-Weighted Classification)**.** *For a given training data set* $\mathcal{M} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$*, with* $y_i \in \{1, \cdots, K\}$ *for all* $i \in \{1, \cdots, N\}$*, the random forest OOB classification prediction is exactly determined by the weighted-majority vote using RF-GAP (Definition 3) as weights.*

Proof. Given the training set $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, with $y_i \in \{1 \ldots K\}$, for a given tree t and observation $i \in O(t)$, the decision tree prediction of the label y_i is determined by the majority vote among in-bag observations within the shared terminal node:

$$\hat{y}_i(t) = \underset{l=1,\ldots,K}{\arg\max}\sum_{j\in J_i(t)} I(y_j = l).$$

Thus, the random forest classification prediction, $\hat{y}_i$, is the most popular predicted class over all $t \in S_i$:

$$\begin{aligned}
\hat{y}_i &= \operatorname*{arg\,max}_{k=1,\ldots,K} \sum_{t \in S_i} I(\hat{y}_i(t) = k) \\
&= \operatorname*{arg\,max}_{k=1,\ldots,K} \sum_{t \in S_i} I\left(\left\{ \operatorname*{arg\,max}_{l=1,\ldots,K} \sum_{j \in J_i(t)} I(y_j = l) \right\} = k\right)
\end{aligned}$$

We show equivalence with the proximity-weighted predictor. The proximity-weighted predictor predicts the class with the largest proximity-weighted vote:

$$\begin{aligned}
\hat{y}_i^p &= \operatorname*{arg\,max}_{k=1,\ldots,K} \sum_{j=1}^{N} p_{GAP}(i,j) I(y_j = k) \\
&= \operatorname*{arg\,max}_{k=1,\ldots,K} \left\{ \sum_{j=1}^{N} \left(\frac{1}{|S_i|} \sum_{t \in S_i} \frac{I(j \in J_i(t))}{|J_i(t)|} \right) I(y_j = k) \right\} \\
&= \operatorname*{arg\,max}_{k=1,\ldots,K} \left\{ \frac{1}{|S_i|} \sum_{t \in S_i} \frac{1}{|J_i(t)|} \left(\sum_{j=1}^{N} I(j \in J_i(t), y_j = k) \right) \right\} \\
&= \operatorname*{arg\,max}_{k=1,\ldots,K} \left\{ \frac{1}{|S_i|} \sum_{t \in S_i} \frac{1}{|J_i(t)|} \sum_{j \in J_i(t)} I(y_j = k) \right\} \\
&= \operatorname*{arg\,max}_{k=1,\ldots,K} \left\{ \sum_{t \in S_i} \frac{1}{|J_i(t)|} \sum_{j \in J_i(t)} I(y_j = k) \right\}.
\end{aligned}$$

The last line holds as $|S_i|$ does not depend on k. As classification trees in a random forest are grown until terminal (leaf) nodes are pure, all in-bag observations belong to the same class. Denote the common class for any observation $j \in J_i(t)$ as $y_{i,t}$. Then the single tree predictor is given by

$$\begin{aligned}\hat{y}_i(t) &= \operatorname*{arg\,max}_{l=1,\ldots,K} \sum_{j \in J_i(t)} I(y_j = l) \\ &= y_{i,t}.\end{aligned}$$

and the random forest predictor is

$$\hat{y}_i = \operatorname*{arg\,max}_{k=1,\ldots,K} \sum_{t \in S_i} I(y_{i,t} = k).$$

The proximity-weighted predictor is thus

$$\begin{aligned}\hat{y}_i^p &= \operatorname*{arg\,max}_{k=1,\ldots,K} \left\{ \sum_{t \in S_i} \frac{1}{|J_i(t)|} \sum_{j \in J_i(t)} I(y_j = k) \right\} \\ &= \operatorname*{arg\,max}_{k=1,\ldots,K} \left\{ \sum_{t \in S_i} \frac{1}{|J_i(t)|} \sum_{j \in J_i(t)} I(y_{i,t} = k) \right\} \\ &= \operatorname*{arg\,max}_{k=1,\ldots,K} \left\{ \sum_{t \in S_i} \frac{1}{|J_i(t)|} |J_i(t)| \, I(y_{i,t} = k) \right\} \\ &= \operatorname*{arg\,max}_{k=1,\ldots,K} \sum_{t \in S_i} I(y_{i,t} = k) \\ &= \hat{y}_i.\end{aligned}$$

□

3.3 Experimental Validation of Proximity-Weighed Prediction

To demonstrate that the RF-GAP proximities preserve the random-forest learned data geometry, we empirically validate Theorems 1 and 2 from Section 3.2, demonstrating that the random forest predictions are preserved in the proximity construction. We also compare to the proximity-weighted predictor using the original random forest proximity definition (Definition 1), the OOB adaptation (Definition 2), PBK [45], and RFProxIH [46].

We compared proximity-prediction results on 19 datasets from the UCI repository [51]. Each dataset was randomly partitioned into training (80%) and test (20%) sets. For each dataset, the same trained random forest was used to produce all compared proximities. Table 3.1 gives the absolute difference between the proximity-weighted training errors and the random forest OOB error rate. The proximity-weighted predictor using RF-GAP almost exactly matches the random forest OOB error rates; discrepancies are due to random tie-breaking. In contrast, the original definition, PBK, and RFProxIH typically produce much lower training error rates, suggesting overfitting. The OOB proximities (Definition 2) produce training error rates that are sometimes lower and sometimes higher than the random forest OOB error rate. Table 3.1 also provides the difference between the test error rates for the same datasets. Here, we still see that the proximity predictions using RF-GAP nearly always match those of the random forest, while this is not the case for the other proximity constructions.

Table 3.1: Comparison of proximity-weighted predictions to the random forest errors. The random forest OOB error (on the training set) and test errors are given in the columns under RF. The absolute difference of the training and test errors with, respectively, the OOB error and RF test error are given for each proximity-weighed prediction. The results nearest the random forest predictions are bold. RF-GAP nearly perfectly matches the random forest results, with differences being accounted for by randomly broken ties in the forest. The other definitions tend to overfit the training data, as can be seen with the large differences between the OOB and test error rates. This is corroborated by Fig. 3.3, which plots the differences. Note that RFProxIH is not written for data with a continuous response. Additionally, since it is generated using Euclidean distance, it is not compatible with datasets with categorical variables or missing values.

Type	RF		RF-GAP		Original		OOB		PBK		RFProxIH	
Data	OOB	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
Arrhythmia	0.258	0.297	**0**	**0**	0.042	0.077	0.067	0.088	0.031	0.077	NA	NA
Balance Scale	0.156	0.144	**0.004**	**0**	0.07	0.056	0.05	0.056	0.068	0.056	0.068	0.056
Banknote	0.009	0.011	**0**	**0**	0.001	0.011	0.009	0.011	0.011	0.018	0.001	0.011
Breast Cancer	0.03	0.043	**0**	**0**	0.007	0.014	0.02	0.014	0.011	0.014	0.007	0.014
Car	0.038	0.052	0.007	0.023	**0**	**0.003**	0.017	0.006	0.048	0.043	NA	NA
Diabetes	0.243	0.182	**0.002**	**0**	0.148	0.006	0.028	0.013	0.124	**0**	0.147	**0**
Ecoli	0.153	0.088	**0**	**0**	0.052	**0**	0.007	0.015	0.041	0.029	0.049	**0**
Glass	0.211	0.186	**0**	**0**	0.135	0.023	0.029	0.023	0.111	0.047	0.123	0.047
Heart Disease	0.417	0.475	**0**	**0**	0.302	0.115	0.194	0.115	0.252	0.18	NA	NA
Hill Valley	0.436	0.41	**0.002**	**0**	0.347	0.082	0.11	0.131	0.304	0.098	0.331	0.082
Ionosphere	0.075	0.042	**0**	**0**	0.025	**0**	0.004	**0**	0.021	**0**	0.025	**0**
Iris	0.05	0.067	**0**	**0**	0.033	**0**	0.008	**0**	0.033	**0**	0.033	**0**
Liver	0.305	0.336	**0**	**0**	0.186	0.078	0.071	0.078	0.162	0.052	NA	NA
Lymphography	0.153	0.2	**0.008**	**0**	0.093	0.033	**0.008**	**0**	0.068	0.033	0.085	0.033
Parkinsons	0.09	0.103	**0**	**0**	0.013	**0**	0.051	**0**	0.006	**0**	0.013	**0**
Seeds	0.063	0.075	**0.006**	**0**	0.038	0.025	0.019	0.05	0.019	0.05	0.038	0.025
Sonar	0.169	0.167	**0**	**0**	0.145	0.024	0.066	0.024	0.12	0.024	0.145	0.024
Statlog	0.234	0.28	**0.001**	**0**	0.139	0.005	0.044	0.015	0.116	0.005	NA	NA
Tic-Tac-Toe	0.048	0.042	**0.003**	**0**	0.031	0.026	0.038	0.021	0.022	0.057	NA	NA
Train - Test	–	–	**0.003 ± 0.008**		0.092 ± 0.019		0.004 ± 0.012		0.076 ± 0.016		0.071 ± 0.019	

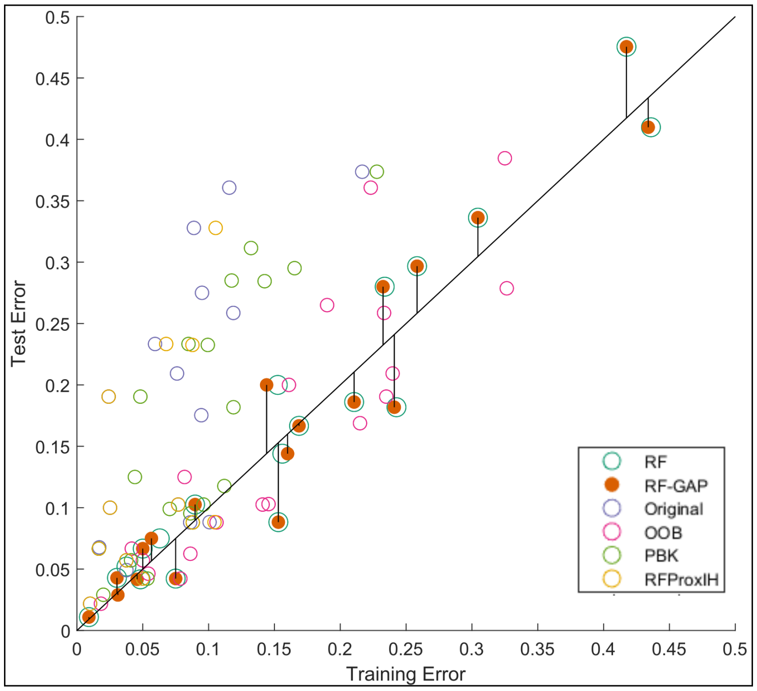


Fig. 3.2: Training vs. test error of the proximity-weighted predictions across multiple datasets. We see that the original proximities, PBK, and RFProxIH, tend to overfit the training data, as demonstrated by points above the line $y = x$. The random forest errors and RF-GAP nearly perfectly align in most cases and are each well-described by the line. OOB also follows the identity line well but does not match the RF predictions.

The RF-GAP proximities also generally produce the lowest test errors. This can be seen in Fig. 3.2, which plots the training versus test error rates using the different proximity measures. Table 3.2 gives the regression slope for each proximity definition. From here it is clear that the original proximities, PBK, and RFProxIH overfit the training data on average. This is corroborated in Fig. 3.3, which plots the difference between the random forest out-of-bag error rates and the proximity-weighted errors across the same datasets, demonstrating that the RF-GAP predictions nearly perfectly match those of the random forest for both training and tests sets. It is clear that the original definition, PBK, and RFProxIH overfit the training data in contrast.

3.4 Conclusion

We perform comparisons with applying MDS to the proximities for visualization in Section 4.1, data imputation in Section 4.4, and outlier detection in Section 4.2. Random

Table 3.2: The regression slopes of each proximity type corresponding to the points in Fig. 3.2. RF-GAP and OOB do not exhibit bias towards the training data as they have a slope close to one, while the larger slope of the other proximity definitions indicate they are overfitting the data.

Type	Slope
RF-GAP	**1.036**
Original	1.700
OOB	1.044
PBK	1.664
RFProxIH	1.3955

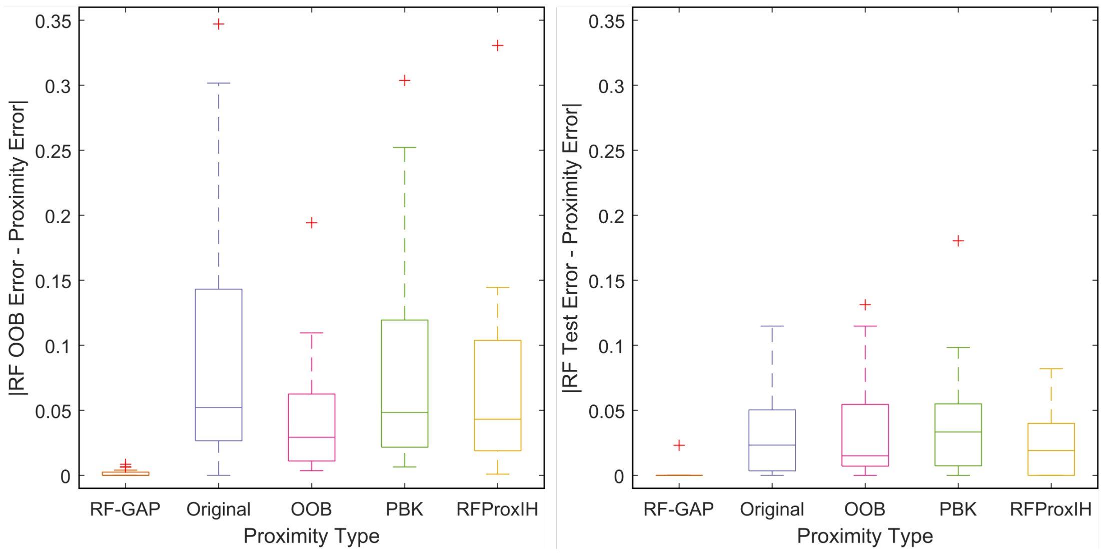


Fig. 3.3: These boxplots show the absolute difference between the proximity-weighted prediction training and test errors with the random forest OOB error rate and test error, respectively, across five proximity definitions. RF-GAP proximity predictions most nearly match the random forest predictions for both the training (left) and test (right) data, thus, best preserving the geometry learned by the random forest. Various UCI datasets were randomly split into training and test datasets for this (80% training, 20% test).

forest proximities have already been established as successful in these applications. Thus, we focus our comparisons in these applications on existing random forest proximity definitions to show that the improved representation of the random forest geometry in RF-GAP leads to improved performance.

In this chapter, we presented a new definition of random forest proximities called RF-GAP that characterizes the random forest out-of-bag prediction results using a weighted nearest neighbor predictor. We proved that the performance of the proximity-weighted predictor exactly matches the out-of-bag prediction results of the trained random forest and also demonstrated this relationship empirically. Thus, RF-GAP proximities capture the random forest-learned data geometry which can provide improvements over existing definitions in proximity-based applications. See the arXiv version of this work at `https://arxiv.org/abs/2201.12682`.

CHAPTER 4
APPLICATIONS OF RF PROXIMITIES

Under Leo Breiman's original random forest proximity definition and under the out-of-bag (OOB) adaptation, random forest proximities are kernel matrices (matrices indicating pair-wise similarities). They are symmetric, positive definite, and bounded above by one. The random forest- geometry-and accuracy-preserving proximities (RF-GAP) from [16], however, are not symmetric and the diagonals, at least for the purpose of weighted-neighbor predictions, are defined to be 0. Both of these may be simply addressed to form a proper kernel. Serving as a similarity matrix, random forest proximities have been applied to a variety of tasks in multiple scientific fields. Some applications include clustering, visualization, imputation, and visualization, among others. Random forest (RF) proximities may replace traditional affinities, such as a Gaussian kernel, or distances, such as Euclidean distance, in virtually any machine learning (ML) task which requires them. Here we discuss common applications in which RF proximities are traditionally used, though they may benefit several additional uses.

4.1 Visualization and Clustering

Leo Breiman [10, 47] first used multi-dimensional scaling (MDS) on random forest proximities to visualize the data. Since then, MDS with random forest proximities has been used in many applications including tumor profiling [15], visualizing biomarkers [52], pathway analysis [53], multi-view learning [13, 14], and unsupervised learning [54]. On a related note, proximities can be useful in data clustering [54].

In [54], Shi and Horvath introduced an unsupervised approach to random forests which uses generated synthetic data as a second class. Original data is given the label 1, while the synthetic data is given the label 0. The now binary-class dataset is used to train a random forest and used to produce a proximity matrix. In that paper, the authors demonstrate the ability of random forest proximities to cluster data into biologically meaningful groups more favorably when compared to Euclidean distance-based clustering. This method was also compared in [55] and showed that random-forest proximity-based clustering provided superior results when compared to Euclidean, Chebyshev, and Canberra distances in the context of high-throughput cellular sequencing data.

Similar observations were found when clustering tumor profiling from micro-array data [15]. Here, the authors also extended proximity-based clustering to t-SNE plots for enhanced visualization. [56] used random forest proximities for a k-Medoids clustering problem in the context of network traffic. They showed that this random-forest-based approach provided the highest accuracy (comparing ground-truth labels to cluster-generated labels) compared to a number of algorithms commonly used in the field. The authors of [57] came up with a proximity-based ensemble method for clustering. Here, they were able to improve clustering improvement on single nucleotide polymorphisms. The cluster fit was quantified using normalized mutual information. [14] used random forest proximities to visualize Alzheimer's patient outcomes via multi-dimensional scaling, showing improved separation between the groups of patients when compared with Euclidean-based visualizations. [58] used proximities to find cluster centers for normal and anomalous training examples in the context of detection systems. Random forest-based visualizations are improved using the RF-GAP proximities, though no experimentation has been done using these proximities for unsupervised clustering.

Here, we compare visualizations using the five different proximities. In each case, metric MDS was applied to $\sqrt{1 - \text{prox}}$ to produce two-dimensional visualizations.

Figure 4.1 gives an example of MDS applied to the classic Sonar dataset from UCI [51] with an OOB error rate was 15.85%. RF-GAP proximities (Figure 4.1 (a)) show two class groupings with misclassified observations between the groups or within the opposing class. The original proximities, PBK, and RFProxIH (Figure 4.1 (b, d, and e, respectively)) show a fairly clear separation between the two classes. For these proximities, they appear nearly linearly separable which does not accurately reflect the data nor the geometry learned

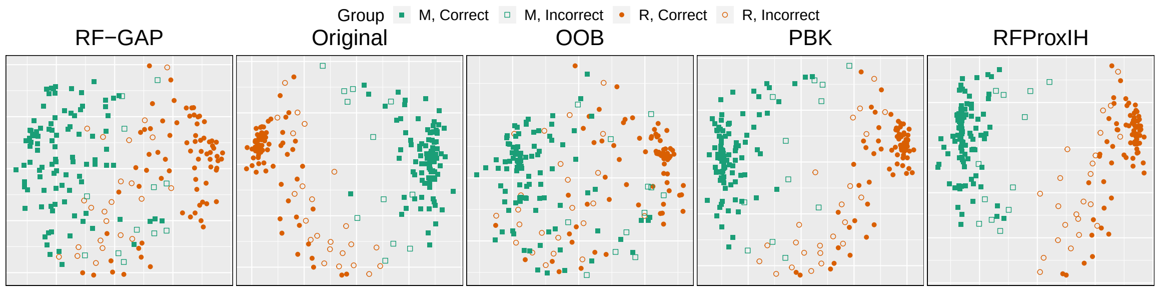


Fig. 4.1: Comparison of MDS embeddings using different RF proximity definitions. Proximities were constructed from a random forest trained on the two-class Sonar dataset (208 observations of 60 variables) from the UCI repository which gave an OOB error rate of 15.87%. Multi-dimensional scaling (MDS) was applied to $\sqrt{1 - \text{prox}}$ using (a) RF-GAP proximities, (b) the original proximities, (c) OOB proximities (Definition 2), (d) PBK, and RFProxIH. Using RF-GAP proximities, the visualization depicts a good representation of the forest's classification problem. For correctly-classified points (filled), there are two clear groupings, while misclassified points (unfilled) are generally located between the groupings or found within the opposite class's cluster, albeit closer to the decision boundary than not. The original definition, PBK, and RFProxIH over-exaggerate the separation between classes. This is apparent in examples (b), (d), and (e) as the two classes appear nearly linearly separable which does not accurately depict the random forest's performance on the dataset. Using only OOB samples to generate the proximities improves upon those three but seems to add some noise to the visualization. There are still two major class clusters, but some correctly classified points are found farther inside the opposite class' cluster compared to the RF-GAP visualization.

by the random forest. Definition 2 (Figure 4.1 (c) has a similar effect as the RF-GAP definition, but with a less clear boundary and seemingly misplaced observations that are deep within the wrong class. These results suggest that RF-GAP can lead to improved supervised visualization and dimensionality reduction techniques. See Section 4.7.1 for further experiments.

4.2 Outlier Detection

Random forest proximities may be used to calculate outliers in a supervised setting. An observation's outlier score is typically defined to be inversely proportional to its average within-class proximity measure. In [59], the authors compared proximity-based outlier detection in the context of the Internet of Things (IoT) device sensing and demonstrated its advantage over other multivariate methods. Similarly, the authors of [60] found that network anomalies can be detected using proximities. This same methodology was used to detect and then remove outliers in pathway analysis problems which lead to much better results in both classification and regression contexts [53]. In [61], it was shown that proximity-based outlier detection was able to accurately identify adulterated foods via infrared spectroscopy. The algorithm has also been shown to be useful in modeling species distribution [62] and predicting galaxy spectral measurements [63]. [64] uses proximities as a part of a non-linear model and demonstrates its effectiveness in fault detection.

In the classification setting, outliers may be described as observations with measurements that significantly differ from those of other observations within the same class. In some cases, these outliers may be similar to observations in a different class, or perhaps they may distinguish themselves from observations in all known classes. In either case, outlying observations may negatively impact the training of many classification and regression algorithms, although random forests themselves are rather robust to outliers in the feature variables [11].

Random forest proximity measures can be used to detect within-class outliers which are likely to have small proximity measures with other observations within the same class. Thus, small average proximity values within a class may be used as an outlier measure. We describe the algorithm as follows:

1. For each observation, i, compute the raw outlier measure score as $\sum_{j \in \text{class}(i)} \frac{n}{prox^2(i,j)}$.
2. Within each class, determine the median and mean absolute deviation of the raw scores.
3. Subtract the median from the raw score and divide by the mean absolute deviation.

The outlier detection measure may be used in conjunction with MDS for visualization. See Figure 4.2 for an example using the Gene Expression Cancer dataset from UCI which has 5 classes across 801 observations and 20,531 variables. Here, the point sizes of the scatter plot are proportionally scaled to the outlier measure. From the figure, it is clear that points outside of their respective class clusters have higher outlier measures. That the outlier measure is inversely proportional to the average proximity to within-class observations is clear in the case of RF-GAP proximities (Figure 4.2 (a)) and is not very clear in the cases

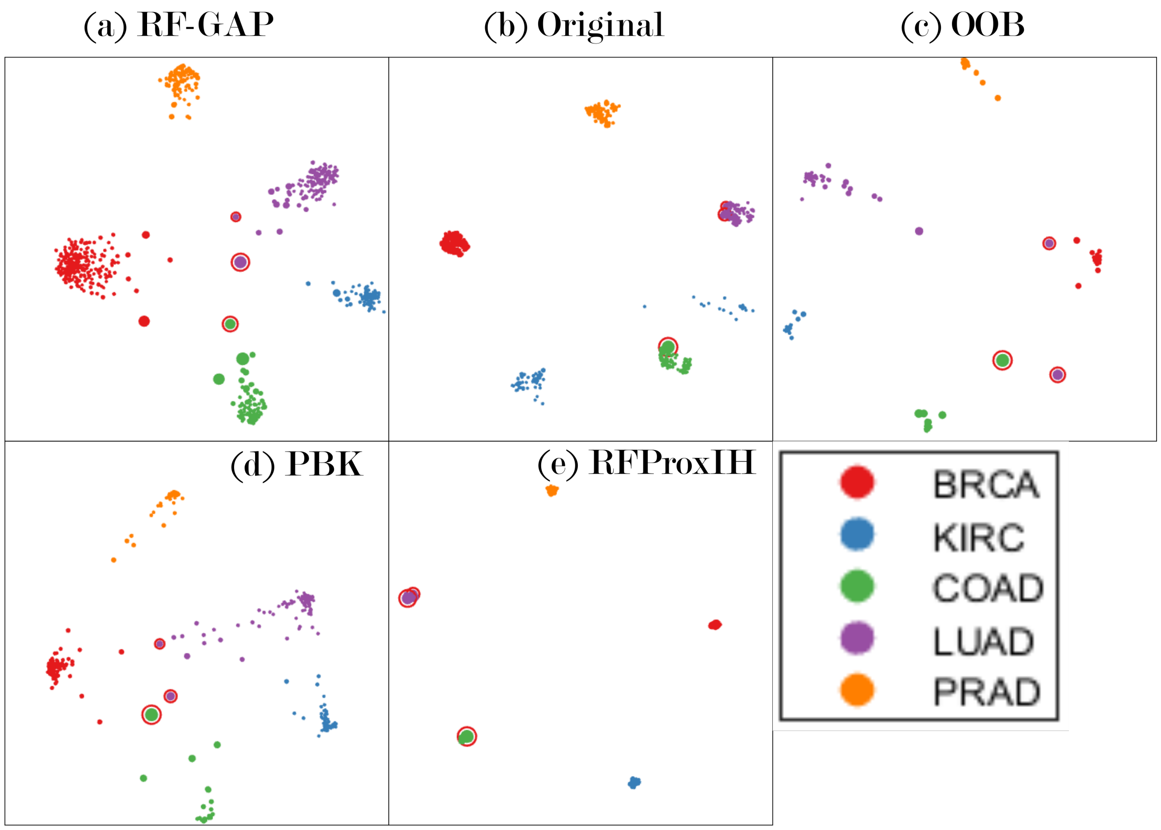


Fig. 4.2: MDS applied to the random forest proximities computed from the Gene Expression Cancer dataset from UCI. The point sizes are inversely proportional to the average proximity of a given observation to all other within-class observations. Misclassified observations are designated by an outline of the color of the misclassified label. The misclassifications in (a) (RF-GAP), (c) (OOB), and (d) (PBK) are clear based on the distance from the blue cluster. The original proximities, (b), and RFProxIH (e) do not clearly account for the misclassified points. The outlier measure scaling in (a) gives a clear reflection of the distance of points to their respective clusters.

of the original definition (b) and RFProxIH (e). This suggests that RF-GAP can be used to improve random forest outlier detection.

Figures 4.3 and 4.4 provide the outlier scores and some outlying examples in the MNIST hand-written digit dataset based on the RF-GAP score. The images are those which received the highest proximity-based outlier scores. In particular, the first digit is a 6 which may be easily mistaken for a 1 given the minuscule loop at the bottom of the digit. The second digit has a true label 0 but looks much more like a 6 with its protruding tail at the top of the digit. Some of the figures may be indistinguishable for a human.

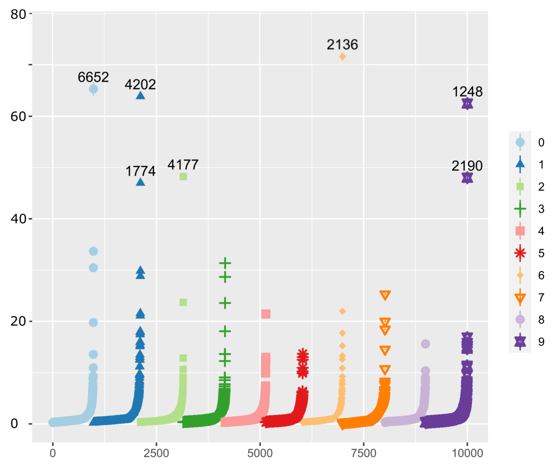


Fig. 4.3: A sorted plot of the outlier measures for the MNIST test dataset as provided by RF-GAP. The vertical axis is the outlier measure as described in Section 4.2. The top seven outlying images are labeled with an index and shown in Figure 4.4.

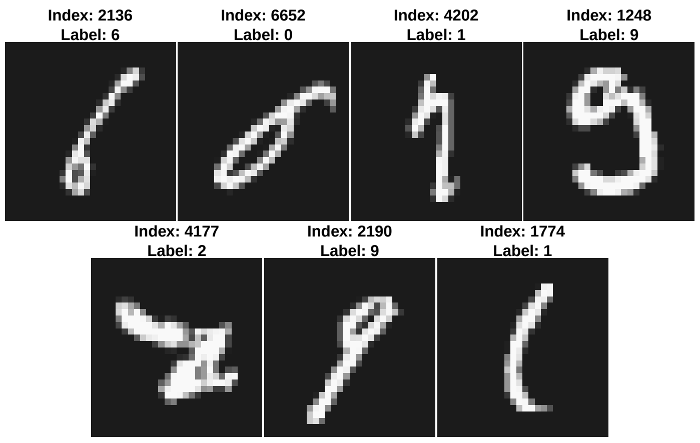


Fig. 4.4: The top seven outlying digits per RF-GAP (see Figure 4.3). Some of these digits may even be difficult for a human to classify, corroborating the RF-GAP outlier score.

4.3 Multi-Modal Learning

Multi-modal/multiview problems can also be approached using random forest proximities. Gray et al. additively combined random forest proximities from different modalities or views (FDG-PET and MR imaging) of persons with Alzheimer's disease or with mild cognitive impairment. They applied MDS to the combined proximities to create an embedding used for classification. Classification on the multi-modal embedding showed significantly better results than classification on both modes separately [14]. Cao, Bernard, Sabourin, and Heutte explored variations of proximity-based classification techniques in the context of multi-view radiomics problems [13]. They compared with the work from [14] and joined proximity matrices using linear combinations. The authors explored random forest parameters to determine the quality of the proximity matrices. They concluded that a large number of maximum-depth trees produced the best quality proximities, quantified using a one-nearest neighbor classifier. Using our proposed random forest proximity measure which accurately reflects the random forest predictions from each view may add to the success of this method, thus creating a truer forest ensemble for multi-view learning.

4.4 Imputation

Random forest proximities are used to impute missing data by replacing missing values of a given variable with a proximity-weighted sum of non-missing values. The authors of [65] compared nine methods (including simple methods, such as mean-value imputation, and more sophisticated models, such as Bayesian Principal Components Analysis and nearest-neighbor-based approaches) for data imputation across seven mechanisms (points missing at random, completely at random, not at random, etc.) for artificially removed data points. They showed that in most cases, random-forest proximity-based imputation provided the best imputation results.

These results were corroborated by Pantanowitz and Marwala, who used proximity-based methods to impute missing data in the context of HIV seroprevalence [66]. They compared their results with five additional imputation methods, including neural networks, and random forest-neural network hybrids, and concluded that random forest imputation produced the most accurate results with the lowest standard errors. Shah et al. compare random forest imputation with multivariate imputation by chained equations (MICE [67]) on cardiovascular health records data [68]. They showed that random forest imputation methods typically produced more accurate results and that in some circumstances MICE gave biased results under default parameterization. In [65] it was similarly shown that random forests produced the most accurate imputation in a comprehensive metabolomics imputation study. Here we describe the algorithm for random forest imputation and compare results using various proximity definitions.

To impute missing values of variable j:

1. If variable j is continuous, initialize the imputation with the median of the in-class values of variable j; otherwise, initialize it with the most frequent in-class value. In the regression context, the median or most frequent values are computed across all observations without missing values in variable j
2. Train a random forest using the imputed dataset
3. Construct the proximity matrix from the forest
4. If variable j is continuous, replace the missing values with the proximity-weighted sum of the non-missing values. If categorical, replace the missing values with the proximity-weighed majority vote
5. Repeat steps 2 - 4 as required. In many cases, a single iteration is sufficient

We show empirically that random forest imputation is generally improved using the RF-GAP proximity definition. For our experiment, we selected various datasets from the UCI repository [51] and for each dataset, we removed 5%, 10%, 25%, 50%, and 75% of values at random (that is, the values are missing completely at random, or MCAR), using the `missMethods` `R` package [69]. Two comparisons were made: 1) we computed the mean MSE across 100 repetitions using a single iteration, and 2) we computed the mean (across 10 repetitions) MSE at each of 15 iterations.

A summary of performance rankings is given in Table 4.1. Across all compared proximity definitions (RF-GAP, OOB, Original, and RFProxIH), RF-GAP achieved the best imputation scores at all percentages less than 75% and outperformed across 69% of the datasets when the percentage of missing values was 75%. For full results, see Table 4.2 in the supplementary materials which gives the mean MSE across 100 repetitions using a single iteration of the above-described algorithm. The number of observations and variables for each dataset are provided in the table.

Supplementary figures in Section 4.7 (Figure 4.11, 4.12, and 4.13) compare imputation results across 16 datasets, using 15 iterations in each experiment with the mean MSE and

Table 4.1: The average ranks of the imputation scores across the various UCI datasets and five percentages of missing values. Each imputation experiment was repeated 100 times with different random initialization. For each percentage under 75%, RF-GAP always produced the best results. See Table 4.2 for full imputation results.

	5%	10%	25%	50%	75%
RF-GAP	1.00	1.00	1.00	1.00	1.31
Original	2.69	2.88	2.62	2.69	2.44
OOB	2.62	2.62	2.62	2.56	2.38
RFProxIH	3.69	3.50	3.75	3.75	3.88

standard errors recorded for each of the repetitions. The value recorded at iteration 0 is the MSE given the median- or majority-imputed datasets. In many cases, the imputation appears to converge quickly with relatively few iterations. Generally, the RF-GAP proximities outperformed the other definitions at each number of iterations. For high percentages of missing values (at least 75%), or for small datasets, the random forest imputation does not always converge and performance may actually decrease as the number of iterations increases. These results suggest that RF-GAP can be used to improve random forest imputation.

4.5 Variable Importance Assessment

Random forest variable importance assessment is usually done via random variable permutation, but may also be done using the forest's proximity values. One approach is to measure the changes in proximities as variables are randomly permuted. This approach was used in the context of gene selection and was shown to be more sensitive to selecting meaningful genes when compared with the traditional random forest permutation importance [70]. In [71], feature contributions to the random forest decision space (defined by proximities) are explored. While many measures of variable importance are generally computed at a global level, the authors propose local, permutation-based feature importance which captures both the contribution (influence of the feature in the decision space) and closeness (position in the decision space relative to the in- and out-class), giving further insight to the contribution of each feature at the terminal node level.

4.6 Additional Uses

Seoane et al. proposed the use of random forest proximities to measure gene annotations with some improvement in precision over other existing methods [72]. In [73], the authors proposed the use of proximities as a matching method for observational studies. They iteratively matched subjects with the highest proximity values but of opposing groups in order to group similar subjects for comparisons across the study. They showed that this method was superior to propensity and other techniques. In [74], the authors provide a proximity-based surrogate model to estimate extreme tower loads on a wind turbine. Here, the motivation for a random-forest proximity-based model was the incapability of existing models to handle the sparse, high-dimensional data. The proximities were used to successfully impute turbine loads based on proximity-matching.

4.7 Additional Experimental Results

Here we present additional experimental results, demonstrating that using RF-GAP leads to improved imputation, visualization, and outlier detection over the other random forest proximity measures.

4.7.1 Multidimensional Scaling and Outlier Detection

Here we provide additional examples of MDS applied to the various random forest proximities. In Figure 4.5, we compare the plotted MDS embeddings on the Ionosphere data from the UCI [51]. It is clear from the images that the random forest's misclassified points are typically found on the border between the two class clusters in the RF-GAP embeddings while this is not always the case for the other proximity measures. Additional figures (4.6, 4.7, and 4.8) are given to display MDS applied to the proximities. In Figure 4.6 we see similar patterns which were displayed in Figure 4.1; exaggerated separation in the Original and RFProxIH and excess noise in OOB. RF-GAP seems to accurately portray why the misclassifications are made in the context of proximity-weighed predictions.

Figures 4.8, 4.9, and 4.10 give additional examples of proximity-based outlier scores. Points that are farther from their respective class clusters can be viewed as outliers and are

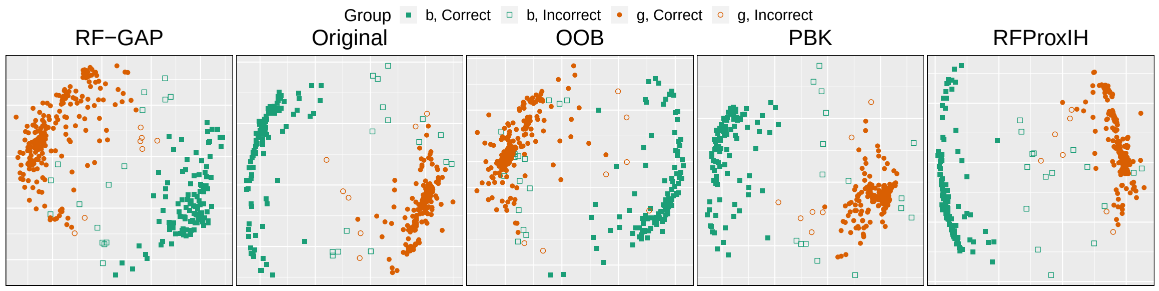


Fig. 4.5: MDS applied to various random forest proximities on the Ionosphere dataset. This binary classification problem predicts whether or not returned radar signals are representative of a structure (good) or not (bad). We see a similar pattern here regarding the MDS embeddings as in Figure 4.1. The class separations are somewhat exaggerated for the Original, PBK, and RFProxIH proximities, while points clearly susceptible to misclassification are identifiable in the RF-GAP and OOB plots.

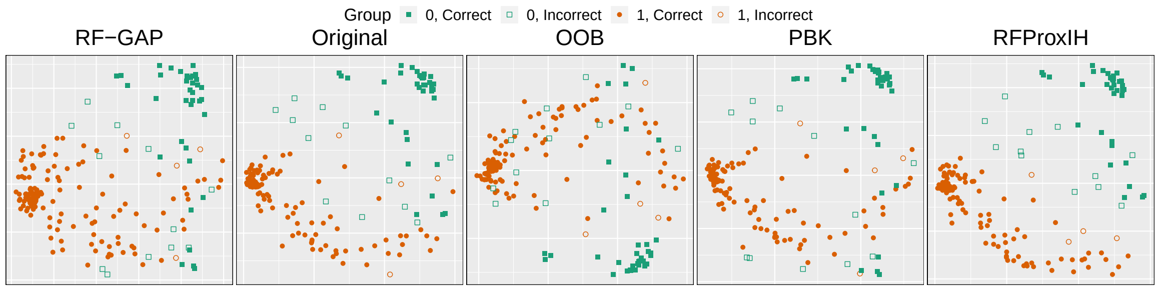


Fig. 4.6: MDS applied to various random forest proximities on the Parkinson's dataset (UCI), which tests whether machine learning algorithms can discriminate between healthy and unhealthy speech signals recorded from people with Parkinson's disease. From the RF-GAP embeddings, it is clear that misclassified points are on the borders or edges of the main clusters. This provides an example where random forest predictions correspond to proximity-weighted predictions. This is not always clear in the other embeddings. For example, the Original MDS embeddings show a misclassified 1 (in the bottom right of the figure) which is the nearest observation of the same class. Again, RFProxIH shows a nearly perfectly linear separation between classes, which is unreasonable with a random forest error rate of 8.2%.

often misclassified. The point size is proportional to the outlier measure in the figure. This, however, may not be as clear when two or three points are far from their respective cluster but near to each other.

4.7.2 Data Imputation

Here we show extended results on data imputation using random forest proximities. Table 4.2 shows the average imputation results for 100 trials across 16 datasets from the UCI repository [51] using four of the proximity measures with a single iteration. For each dataset, values were removed completely at random in amounts of 5%, 10%, 25%, 50%,

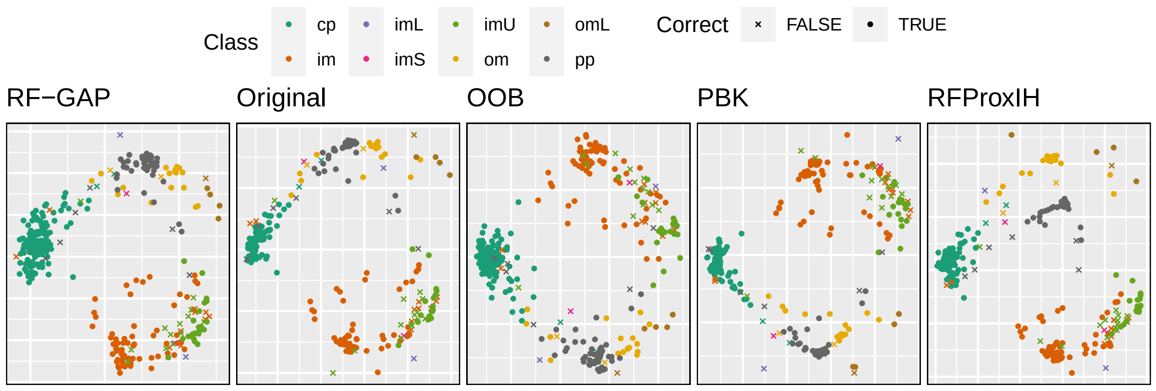


Fig. 4.7: The ecoli dataset with eight classes after applying MDS to the random forest proximities. RF-GAP and OOB show looser clusters compared with the others. This is suggestive of less overfitting of the training data.

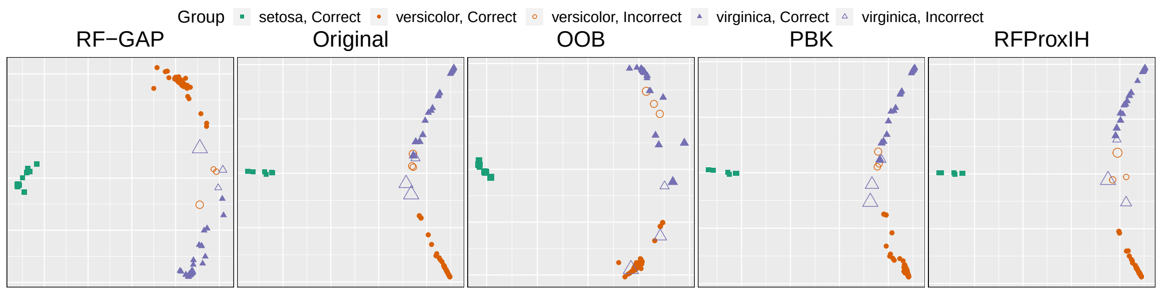


Fig. 4.8: Fisher's Iris dataset with MDS applied to the random forest proximities. Here, point size is proportional to the outlier score provided by each method. In each case, observations with high outlier scores corresponded to misclassified points.

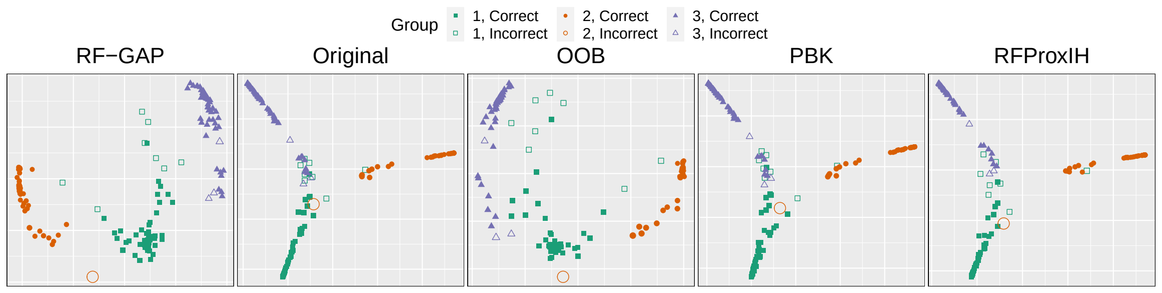


Fig. 4.9: The seeds dataset compares three varieties of wheat seeds using geometric properties (e.g., width, length) as features. The OOB and RF-GAP proximities produce more cluster-like structures, vs. the branching seen by the other definitions. RF-GAP clearly shows why the misclassifications are taking place.

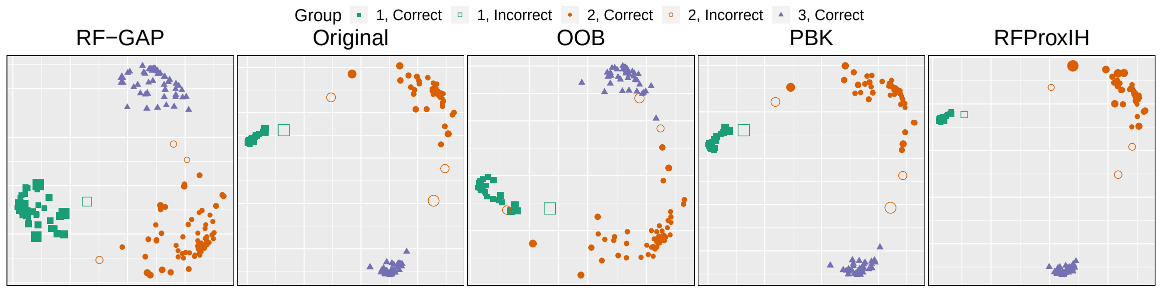


Fig. 4.10: The wine dataset consists of three classes (corresponding to locations of cultivation) and 13 features. The tight clusters of each class using the original, PBK, and RFProxIH proximities suggests overfitting to the training data. It seems RF-GAP may show more sensitivity to outliers.

and 75%. The PBK proximities were omitted from this study due to their slow computational complexity. Additionally, some datasets were not compatible with RFProxIH due to continuous responses or categorical features.

The mean squared error (MSE) is almost universally lower when using RF-GAP for imputation. RF-GAP is only outperformed sometimes when the amount of missing values reaches 75%. Even in these cases, RF-GAP is always in second place. The Banknote, Ionosphere, Optical Digits, Parkinson's, and Waveform datasets particularly show good examples of RF-GAP for imputation. Here, RF-GAP outperforms each of the other definitions and the error decreases monotonically.

Figures 4.11, 4.12, and 4.13 show imputation results across multiple iterations. Each experiment was repeated 100 times across 15 iterations. In general, RF-GAP outperforms the other proximity-weighted imputations although the imputation tends to be much noisier for smaller datasets (Balance Scale, Ecoli, Iris, and Seeds, for example [51]) and less reliable for large percentages of missing values. This is particularly prominent when 75% of the data is missing. In some of these cases, the error increases with the number of iterations, for example, in the Ecoli and Diabetes data sets.

4.8 Proximity Application Conclusion

All of the above-described applications have historically used definitions of random forest proximities that do not envelope the data geometry learned by the random forest. In contrast, RF-GAP accurately reflects this geometry which was demonstrated by our

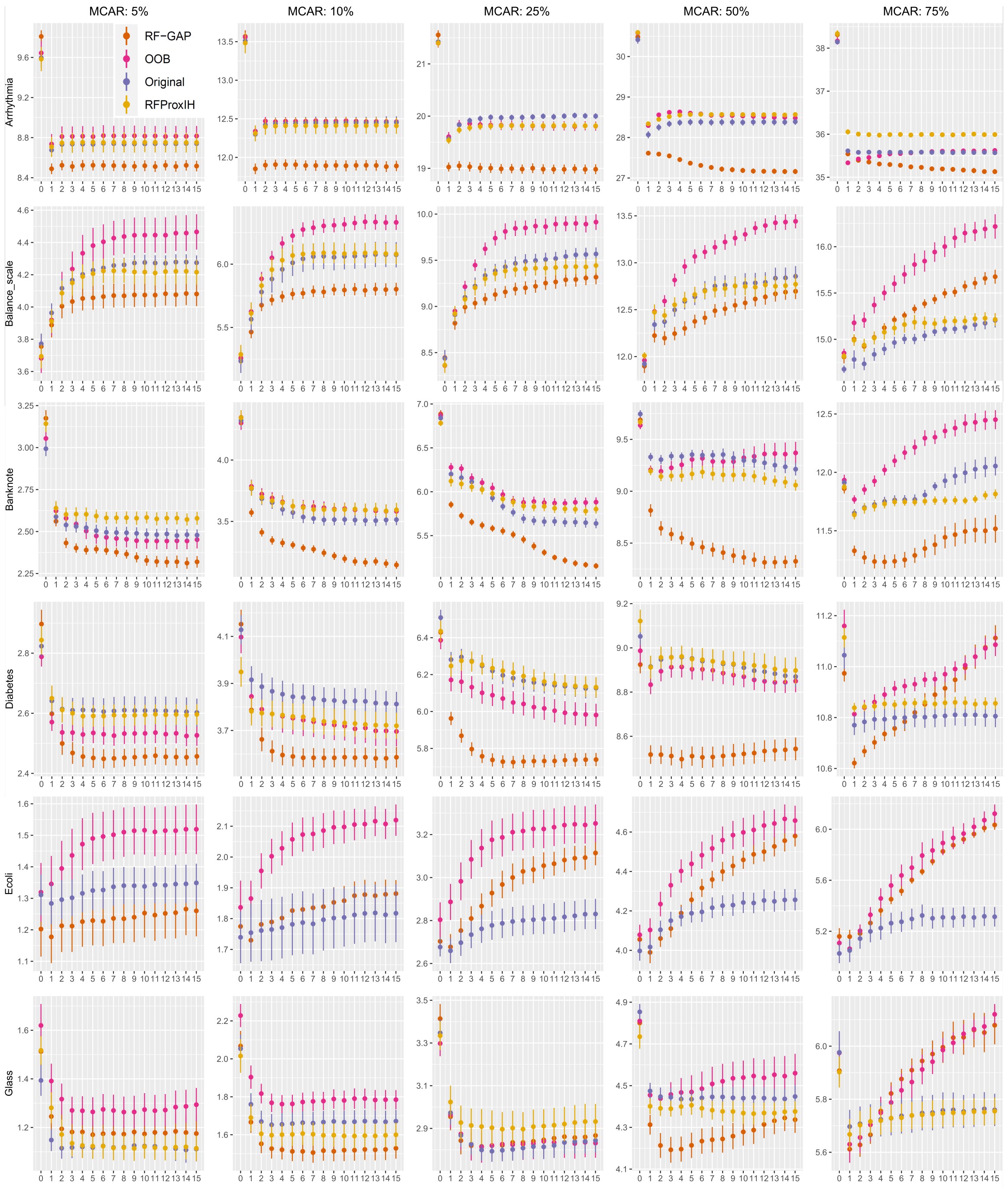


Fig. 4.11: Additional imputation results. See Figure 4.13 for more details.

application experiments presented in this chapter. Indeed, we showed empirically that using RF-GAP gives data visualizations that more accurately represent the geometry learned from the random forests, outlier scores that are more reflective of the random forest's learning,

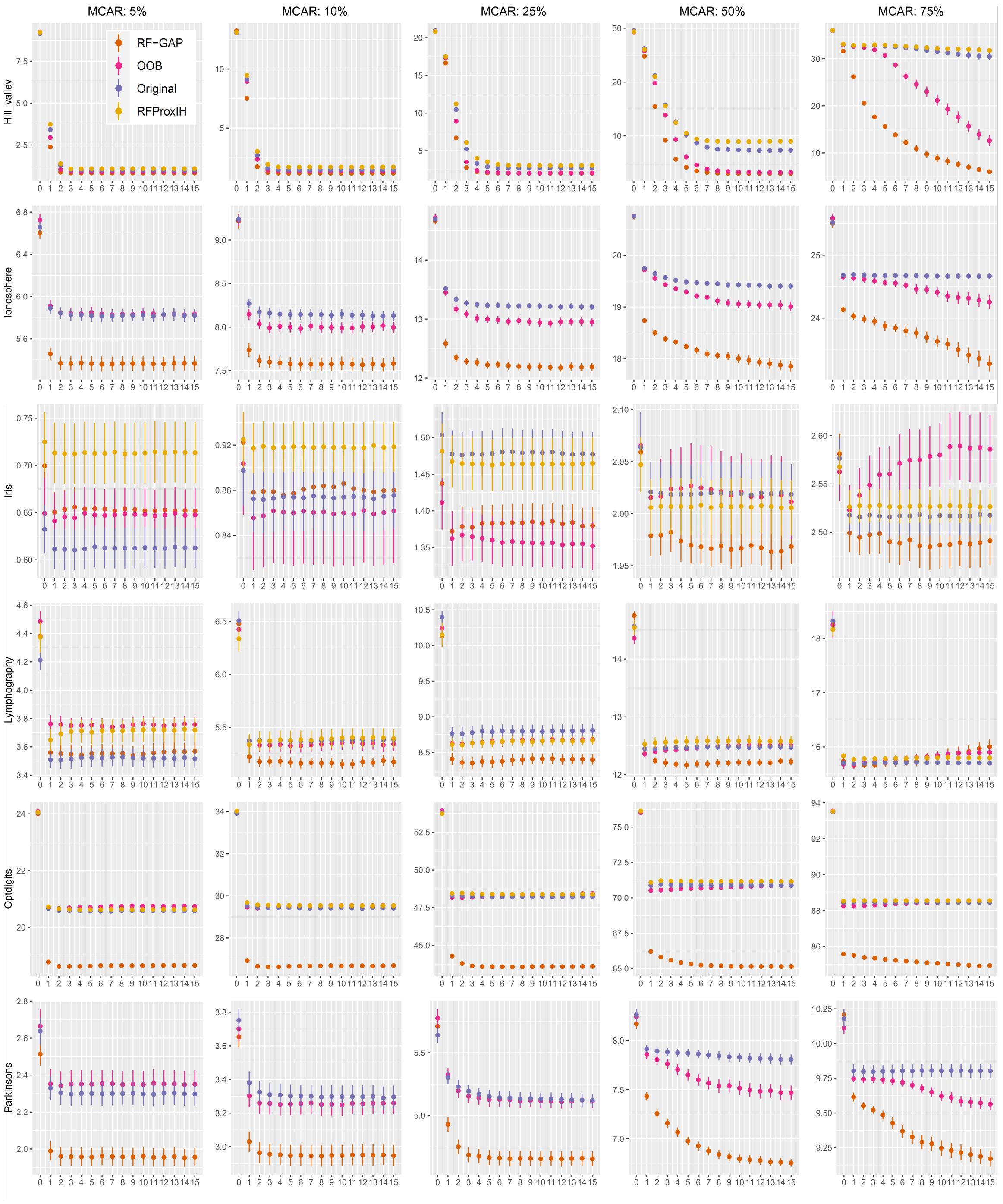


Fig. 4.12: Additional imputation results. See Figure 4.13 for more details.

and improved random forest imputations.

Additional random forest proximity applications will be explored in future works, including quantifying outlier detection performance, comparing outlier detection against non-

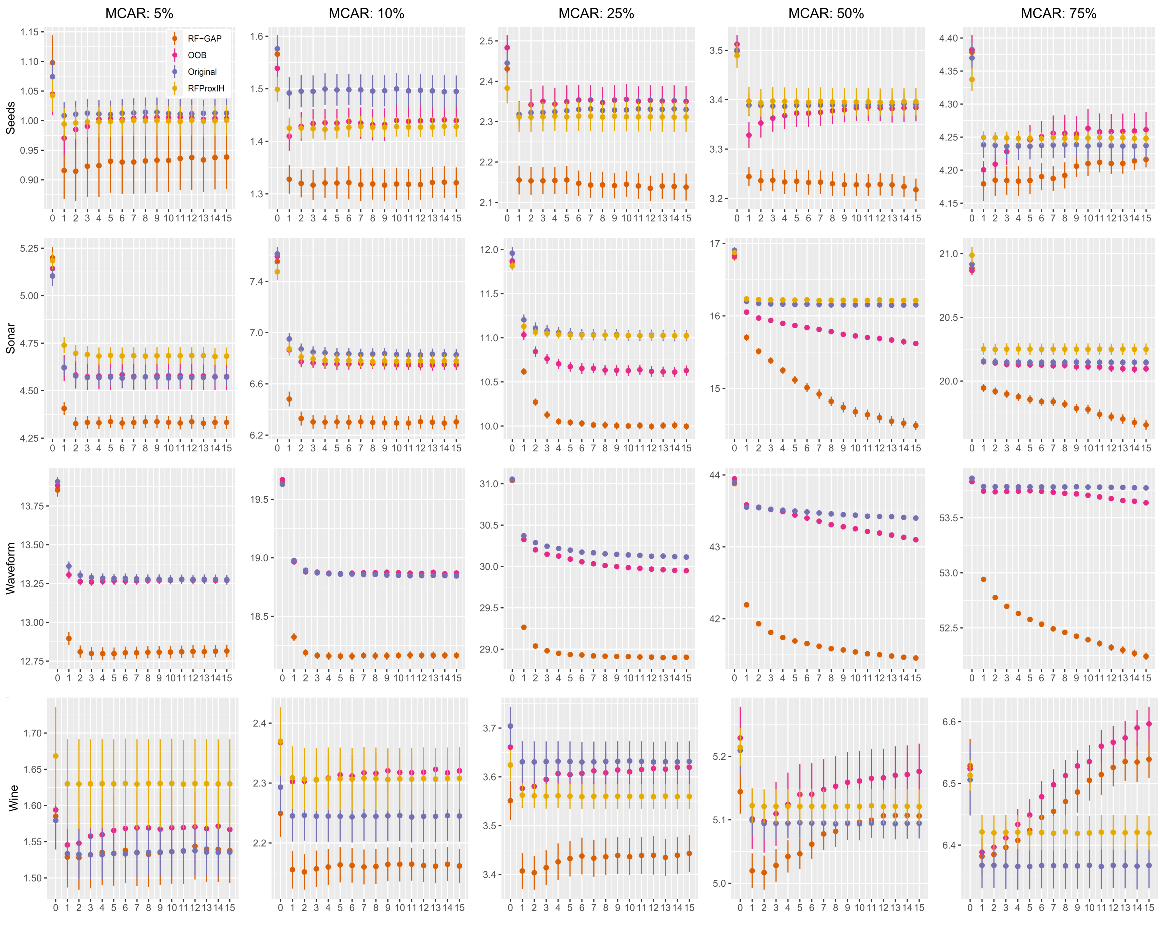


Fig. 4.13: This figure, Figure 4.11 and 4.12 give the mean-squared error (MSE) between the original and imputed values using four random forest proximity measures. All data variables were scaled from 0 to 1 for comparability. The scores were compared using 1 to 15 imputation iterations as described in Section 4.4 and each experiment was conducted over 10 repetitions using the original proximities, OOB proximities, RF-GAP proximities, and RFProxIH. Imputation 0 provides the MSE for the median-filled imputation. Four different percentages of values missing completely at random (MCAR) were used (5%, 10%, 25%, and 50%) across several datasets from the UCI repository. In general, RF-GAP outperforms the other proximity-weighted imputations. See additional results in Table 4.2 for a single iteration.

tree based methods, assessing variable importance, and applying RF-GAP to multi-view learning. This last application shows promise as classification accuracy was greatly increased after combining proximities in [13, 14] using other definitions. Multi-view learning may also be paired with this approach in some domains to visualize and assess contributions from the various modes or to perform manifold alignment. An additional area of improvement is scalability. Our approach is useful for datasets with a few thousand observations, but we can expand its capabilities by implementing a sparse version of RF-GAP. Further adaptations may make random forest proximity applications accessible for even larger datasets.

Table 4.2: Complete results for data imputation using random forest proximities using a single iteration. For each considered dataset, values were removed completely at random in the amounts of 5%, 10%, 25%, 50%, and 75%. Missing values were imputed using various proximity definitions (PBK was not used here for computational considerations). The experiment was repeated 100 times for all datasets. The two numbers directly below each dataset indicate the number of observations and number of variables, respectively. The average of the mean-squared errors (MSE) between the original and imputed values are recorded along with the standard errors. For missing value percentages up through 50%, RF-GAP proximities (Definition 3) provided the most accurate imputation across all datasets. At 75% missing values, RF-GAP proximities outperformed across 69% of datasets. Otherwise, the RF-GAP proximities are in second place. Note: some datasets were not compatible with RFProxIH due to continuous response or categorical feature vectors.

Data	Proximity	5%	10%	25%	50%	75%
Arrhythmia	RF-GAP	**8.35 ± 0.03**	**11.76 ± 0.02**	**18.99 ± 0.02**	**27.68 ± 0.02**	35.49 ± 0.02
452	Original	8.72 ± 0.02	12.24 ± 0.03	19.54 ± 0.02	28.20 ± 0.02	35.66 ± 0.02
279	OOB	8.77 ± 0.02	12.30 ± 0.02	19.59 ± 0.02	28.24 ± 0.02	**35.30 ± 0.01**
	RFProxIH	8.75 ± 0.03	12.27 ± 0.02	19.62 ± 0.02	28.33 ± 0.02	35.95 ± 0.02
Balance Scale	RF-GAP	**3.87 ± 0.02**	**5.47 ± 0.02**	**8.78 ± 0.02**	**12.35 ± 0.02**	15.01 ± 0.02
645	Original	3.94 ± 0.02	5.59 ± 0.02	8.92 ± 0.03	12.43 ± 0.02	**15.00 ± 0.03**
4	OOB	3.96 ± 0.02	5.63 ± 0.03	8.97 ± 0.02	12.49 ± 0.02	15.05 ± 0.02
	RFProxIH	3.91 ± 0.02	5.61 ± 0.03	8.93 ± 0.02	12.45 ± 0.02	15.02 ± 0.03
Banknote	RF-GAP	**2.50 ± 0.02**	**3.56 ± 0.01**	**5.86 ± 0.01**	**8.88 ± 0.02**	**11.40 ± 0.01**
1372	Original	2.64 ± 0.01	3.77 ± 0.01	6.21 ± 0.02	9.25 ± 0.02	11.66 ± 0.01
5	OOB	2.64 ± 0.02	3.81 ± 0.02	6.24 ± 0.01	9.28 ± 0.02	11.70 ± 0.01
	RFProxIH	2.65 ± 0.01	3.77 ± 0.02	6.20 ± 0.02	9.28 ± 0.01	11.70 ± 0.01
Diabetes	RF-GAP	**2.56 ± 0.02**	**3.69 ± 0.02**	**5.95 ± 0.02**	**8.58 ± 0.01**	**10.70 ± 0.01**
678	Original	2.67 ± 0.02	3.81 ± 0.02	6.19 ± 0.02	8.83 ± 0.02	10.80 ± 0.01
8	OOB	2.63 ± 0.02	3.78 ± 0.02	6.15 ± 0.01	8.81 ± 0.01	10.80 ± 0.01
	RFProxIH	2.65 ± 0.02	3.83 ± 0.02	6.23 ± 0.02	8.88 ± 0.01	10.81 ± 0.01
Ecoli	RF-GAP	**1.20 ± 0.02**	**1.74 ± 0.02**	**2.73 ± 0.02**	**4.01 ± 0.02**	**5.10 ± 0.02**
336	Original	1.20 ± 0.02	1.79 ± 0.02	2.82 ± 0.02	4.03 ± 0.02	5.13 ± 0.03
8	OOB	1.28 ± 0.02	1.76 ± 0.03	2.91 ± 0.02	4.09 ± 0.02	5.17 ± 0.04
	RFProxIH	NA	NA	NA	NA	NA
Glass	RF-GAP	**1.19 ± 0.02**	**1.73 ± 0.02**	**2.89 ± 0.02**	**4.29 ± 0.02**	**5.63 ± 0.02**
214	Original	1.25 ± 0.02	1.78 ± 0.02	3.01 ± 0.02	4.43 ± 0.02	5.67 ± 0.02
10	OOB	1.28 ± 0.02	1.81 ± 0.02	3.03 ± 0.02	4.40 ± 0.02	5.634 ± 0.01
	RFProxIH	1.29 ± 0.02	1.78 ± 0.02	3.02 ± 0.02	4.45 ± 0.02	5.69 ± 0.02
Hill Valley	RF-GAP	**2.38 ± 0.02**	**7.57 ± 0.05**	**16.60 ± 0.03**	**24.88 ± 0.02**	**31.50 ± 0.02**
606	Original	3.46 ± 0.03	9.27 ± 0.03	17.40 ± 0.03	26.04 ± 0.02	33.05 ± 0.02
101	OOB	2.90 ± 0.03	9.03 ± 0.03	17.30 ± 0.03	25.88 ± 0.02	32.95 ± 0.02
	RFProxIH	3.77 ± 0.03	9.54 ± 0.03	17.54 ± 0.03	26.06 ± 0.02	33.06 ± 0.02
Ionosphere	RF-GAP	**5.47 ± 0.02**	**7.71 ± 0.02**	**12.59 ± 0.02**	**18.77 ± 0.02**	**24.12 ± 0.01**
351	Original	5.90 ± 0.02	8.34 ± 0.02	13.56 ± 0.02	19.81 ± 0.02	24.68 ± 0.01
34	OOB	5.84 ± 0.02	8.25 ± 0.02	13.43 ± 0.02	19.68 ± 0.02	24.67 ± 0.01
	RFProxIH	NA	NA	NA	NA	NA
Iris	RF-GAP	**0.63 ± 0.01**	**0.85 ± 0.01**	**1.37 ± 0.01**	**1.98 ± 0.01**	2.53 ± 0.01
150	Original	0.63 ± 0.01	0.88 ± 0.01	1.40 ± 0.01	2.02 ± 0.01	2.53 ± 0.01
4	OOB	0.63 ± 0.01	0.88 ± 0.01	1.41 ± 0.01	2.00 ± 0.01	**2.52 ± 0.01**
	RFProxIH	0.61 ± 0.01	0.88 ± 0.01	1.43 ± 0.01	2.03 ± 0.01	2.54 ± 0.01
Lymphography	RF-GAP	**3.62 ± 0.02**	**5.29 ± 0.02**	**8.44 ± 0.02**	**12.34 ± 0.02**	15.69 ± 0.03
148	Original	3.64 ± 0.02	5.43 ± 0.03	8.65 ± 0.02	12.51 ± 0.02	**15.69 ± 0.02**
18	OOB	3.70 ± 0.02	5.43 ± 0.02	8.67 ± 0.02	12.56 ± 0.02	15.71 ± 0.02
	RFProxIH	3.71 ± 0.02	5.42 ± 0.02	8.75 ± 0.02	12.50 ± 0.02	15.74 ± 0.02
Optdigits	RF-GAP	**18.75 ± 0.02**	**26.86 ± 0.02**	**44.19 ± 0.02**	**66.30 ± 0.02**	**85.63 ± 0.02**
5620	Original	20.63 ± 0.02	29.58 ± 0.02	48.21 ± 0.02	70.84 ± 0.02	88.43 ± 0.01
64	OOB	20.60 ± 0.02	29.44 ± 0.02	48.03 ± 0.02	70.51 ± 0.02	88.25 ± 0.01
	RFProxIH	20.69 ± 0.02	29.63 ± 0.02	48.45 ± 0.02	71.02 ± 0.02	88.48 ± 0.01
Parkinsons	RF-GAP	**2.08 ± 0.02**	**3.04 ± 0.02**	**4.93 ± 0.02**	**7.46 ± 0.02**	**9.59 ± 0.01**
197	Original	2.27 ± 0.02	3.30 ± 0.02	5.39 ± 0.02	7.90 ± 0.01	9.80 ± 0.01
23	OOB	2.26 ± 0.02	3.28 ± 0.02	5.32 ± 0.01	7.80 ± 0.01	9.76 ± 0.01
	RFProxIH	NA	NA	NA	NA	NA
Seeds	RF-GAP	**0.92 ± 0.01**	**1.31 ± 0.01**	**2.16 ± 0.01**	**3.21 ± 0.01**	**4.14 ± 0.01**
210	Original	1.01 ± 0.01	1.44 ± 0.01	2.34 ± 0.01	3.38 ± 0.01	4.23 ± 0.01
7	OOB	1 ± 0.01	1.41 ± 0.01	2.28 ± 0.01	3.30 ± 0.01	4.18 ± 0.01
	RFProxIH	1.04 ± 0.01	1.46 ± 0.01	2.36 ± 0.01	3.42 ± 0.01	4.27 ± 0.01
Sonar	RF-GAP	**4.39 ± 0.01**	**6.49 ± 0.01**	**10.59 ± 0.02**	**15.72 ± 0.01**	**19.95 ± 0.01**
208	Original	4.67 ± 0.02	6.86 ± 0.02	11.08 ± 0.01	16.17 ± 0.01	20.17 ± 0.01
60	OOB	4.64 ± 0.02	6.78 ± 0.02	10.98 ± 0.01	16.06 ± 0.01	20.12 ± 0.01
	RFProxIH	4.72 ± 0.02	6.91 ± 0.02	11.20 ± 0.02	16.17 ± 0.01	20.17 ± 0.01
Waveform	RF-GAP	**12.92 ± 0.01**	**18.33 ± 0.01**	**29.27 ± 0.01**	**42.24 ± 0.01**	**52.91 ± 0.01**
5000	Original	13.32 ± 0.01	18.97 ± 0.01	30.37 ± 0.01	43.58 ± 0.01	53.77 ± 0.01
21	OOB	13.31 ± 0.01	18.93 ± 0.01	30.32 ± 0.01	43.54 ± 0.01	53.75 ± 0.01
	RFProxIH	NA	NA	NA	NA	NA
Wine	RF-GAP	**1.53 ± 0.01**	**2.22 ± 0.01**	**3.50 ± 0.01**	**5.06 ± 0.01**	6.38 ± 0.01
178	Original	1.58 ± 0.01	2.27 ± 0.01	3.55 ± 0.01	5.10 ± 0.01	**6.38 ± 0.01**
13	OOB	1.60 ± 0.01	2.25 ± 0.01	3.54 ± 0.01	5.10 ± 0.01	6.38 ± 0.01
	RFProxIH	1.59 ± 0.01	2.27 ± 0.01	3.5 ± 0.01	5.10 ± 0.01	6.38 ± 0.01

CHAPTER 5

RF-GAP R PACKAGE

5.1 Introduction

Though many papers describe and make use of applications of random forest proximities, very little has been done in examining the quality of the proximity measures regarding how well the proximities incorporate the random forest's learning. A few papers have compared variations of random forest proximities in the context of classification, but none have directly related these classification results to the random forest's learning. For example, Englund et al. compared the quality of random forest proximities, which they termed proximity-based kernels (PBK) by using them as a kernel for the support vector machine [45]. Davies and Ghahramani compared random forest proximities with other common kernel functions, including linear, polynomial, and radial basis function (RBF). They used these kernels in Gaussian Processes and support vector machines (SVM) [50]. Cao et al. [46] compared two novel random forest proximity definitions with the Leo Breiman's original formulation and that of Englund et al. to show minor improvements in classification tasks and in multi-modal learning. However, in each of these cases, it is unclear how these proximities relate to the random forest's learning.

Random forest predictions are calculated by voting within terminal nodes, which nodes form the decision space of the forest. In this sense, proximity constructions based on terminal nodes should be able to convey all of the information about the random forest's predictions. However, existing random forest proximities do not accurately convey the decision space of the forest. That is, the proximities cannot be directly used to reconstruct random forest predictions. We showed that under the RF-GAP proximity definition, the random forest's predictions may be perfectly reconstructed using proximities as weights in a nearest-neighbor prediction problem. Since these proximities can be used to directly reconstruct random forest predictions, we argue that the data geometry learned by the random forest is captured by these proximities, thus, applications using these proximities provide a truer representation of the learning.

For transparency and ease of use, we make the RF-GAP software freely available to users through an `R` [49] package named after the methodology, `rfgap`. See the Github repository here `https://github.com/KevinMoonLab/RF-GAP`. In the package, we provide a simple means of constructing and comparing three types of random forest proximities, including RF-GAP, Leo Breiman's original formulation in which the proximity between two observations is the proportion of shared terminal nodes across all trees, and a variation of this in which only out-of-bag (non-training) samples are used in the proximity matrix construction. See Definitions 1, 2, and 3. Two additional proximity constructions were compared in our paper [16], including Englund's PBK [45] and Cao's RFDisIH [46]; however, neither of these papers provided publicly available code, and our implementations were not optimized for computational efficiency.

Though there exist `R` packages that compute the original and OOB random forest proximities [12,75,76], we construct our own proximities to enable a direct comparison with RF-GAP proximities using the same random forest. For fast computation and ease of use, we use the `ranger` [48] package to build the forest from which we construct proximities.

The `rfgap` package makes it simple to compare the extent to which the proximities capture the random forest's learning. In this chapter, we demonstrate the use of our package in a variety of applications, including proximity construction, classification and regression using proximities as weights, visualization via multidimensional scaling, missing data imputation, and outlier detection using RF-GAP proximities.

5.2 Constructing Proximities

Random forests are capable of making predictions on both continuous and categorical response variables for regression and classification, respectively. Additionally, random forests handle mixed feature variables, that is, predictor variables may be either numeric or categorical and since the partitioning decisions are rank-based, numerical predictor variables do not need to be normalized or standardized as is typical in other machine learning contexts (e.g., neural networks). Thus, the construction of random forest-based proximities does not require the same preprocessing steps as may be needed for other ML processes. This simplifies the use of random forests and thus, for our purposes, the generation of random forest proximities.

Let `x` be a dataframe or matrix object with labels `y`. Here `y` must be numeric (for a regression forest) or a factor type (for a classification task). If `y` is a character vector it will be coerced to be a factor type. To generate the proximities, we use the `get_proximities` function. The user may use a pre-trained random forest to construct the proximities, which has the benefit of a direct comparison of proximity types, or to train when calling `get_proximities`. We demonstrate these two options below in Listing 5.1 using the `iris` dataset accessible in `R`.

```
library(rfgap)

# Defining the data and labels
x <- iris[, -5]
y <- iris[, 5]

# Constructing the RF-GAP proximities
proximities <- get_proximities(x, y, seed = 42)
```

Listing 5.1: Defining the RF-GAP proximities on the Iris dataset.

This is the simplest way to generate proximities. Here we simply call `get_proximities` using the dataframe `x` and labels `y` as inputs. By default, RF-GAP proximities are constructed. The argument `type` allows the user to select the type of proximities to be constructed, the package currently supports `"original"`, `"oob"`, and `"rfgap"`.

The user may train a random forest prior to calling `get_proximities`. In this case, the user must train the `ranger` forest with the options `keep.inbag` and `write.forest` set to `TRUE`. Using a pre-trained forest allows the user to fairly compare different proximity types without the need of retraining a forest each time. An example of constructing the proximities in this manner is shown below in Listing 5.2.

```
library(rfgap)

# Defining the data and labels
x <- iris[, -5]
y <- iris[, 5]

# Training the random forest
rf <- ranger(x = x, y = y, keep.inbag = TRUE, write.forest =
   TRUE, seed = 42)

# Constructing three sets of proximities
proximities_rfgap <- get_proximities(x, rf = rf,
                                     type = 'rfgap')
proximities_oob   <- get_proximities(x, rf = rf,
                                     type = 'oob')
proximities_orig  <- get_proximities(x, rf = rf,
                                     type = 'original')
```

Listing 5.2: Here we pretrain a random forest prior to the proximity construction. This allows the user to easily compare different proximities. In this example, the code generates all three available proximity types.

`get_proximities` has the additional option for the user to supply a test set. Including the test set will extend the proximities to the test observations. This is done by using the argument `x_test`, as demonstrated below in 5.3. The returned proximity matrix will have $n\_train + n\_test$ rows and columns.

```
set.seed(42)
train_idx <- sample(nrow(x), size = round(.7 * nrow(x)))
```

```
x_train <- x[train_idx, ]
y_train <- y[train_idx]

x_test  <- x[-train_idx, ]
y_test  <- y[-train_idx]

proximities <- get_proximities(x = x_train, y = y_train,
                               x_test = x_test)
```

Listing 5.3: Here we split the data into training and test sets. The training set is used to build the forest and construct the proximities. Pairwise proximities between all training and test examples are constructed.

The returned proximity matrix is an S3 object of type `rf_proximities`. This object type has additional methods associated with it for making predictions, producing visualizations, detecting outliers, and imputing missing data. We will discuss these methods in subsequent sections.

5.3 Proximity-Based Predictions

We have shown that RF-GAP proximities serve as weights which may be used to perfectly reconstruct the forests predictions. To make predictive comparisons simple, we extend the generic `predict` function to accept the `rf_proximities` class. The user simply needs to provide the proximities and corresponding labels `y`. The labels must be either numeric for regression, or factors for classification. Predictions are made using a proximity-weighted class vote or proximity-weighted sum. That is, for the regression problem,

$$\hat{y}_i^p = \sum_{j=1}^{N} prox(i,j) y_j \tag{5.1}$$

and for the classification problem with K classes,

$$\hat{y}_i^p = \underset{k=1,\ldots,K}{\arg\max} \sum_{j=1}^{N} prox(i,j) I\left(y_j = k\right) \tag{5.2}$$

where $\hat{y}_i^p$ is the proximity-weighted prediction, and $I(.)$ is the $0-1$ indicator function. Note that for proximity types other than `"rfgap"` the proximities are rescaled so that the rows sum to one to define the weights. In Listing 5.4, we compute the proximity-weighted predictions and compare the results to those of the random forest. Here we see that the RF-GAP proximity-weighted predictions match those of the random forest.

```
library(rfgap)


# Defining the data and labels
x <- iris[, -5]
y <- iris[, 5]


# Training the random forest
rf <- ranger(x = x, y = y, write.forest = TRUE,
             keep.inbag = TRUE, seed = 42)
predictions_rf <- rf$predictions


# Constructing the RF-GAP proximities
proximities_rfgap <- get_proximities(x, rf = rf)
proximities_original <- get_proximities(x, rf = rf,
                                        type = 'original')


# Proximity-weighted predictions
predictions_rfgap <- predict(proximities_rfgap, y)
predictions_original <- predict(proximities_original, y)
```

```
# Proportion of predictions matching the rf predictions
sum(predictions_rfgap$predictions == predictions_rf) / nrow(x)
# 1
sum(predictions_original$predictions == predictions_rf) / nrow
    (x)
# 0.9533333
```

Listing 5.4: Here we construct and compare the proximity-weighted predictions with those of the random forest. The generic `predict` function takes on the `rf_proximities` S3 object along with the class labels to make the prediction. We compare the proximity-weighted predictions with those of the random forest. RF-GAP proximity predictions perfectly match.

The result of this `predict` function is a list with two elements; `predictions` which give the classification or regression predictions, and `error` which is the error rate of the predictions. The behavior of this function is similar to that of a k-NN function. To extend the functionality to a test set, the user will need to include the proximity matrix which includes both training and test examples, the training labels `y`, and the test labels `y_test`. If the test labels are included, the `predictions` list will also contain the test predictions and test error rate. An example is given in Listing 5.5.

```
library(rfgap)

# Defining the data and labels
x <- iris[, -5]
y <- iris[, 5]

set.seed(42)
train_idx <- sample(nrow(x), size = round(.7 * nrow(x)))

x_train <- x[train_idx, ]
```

```
y_train <- y[train_idx]


x_test  <- x[-train_idx, ]
y_test  <- y[-train_idx]


proximities <- get_proximities(x = x_train, y = y_train, x_
   test = x_test)
predictions <- predict(proximities, y, y_test = y_test)
```

Listing 5.5: Here we demonstrate the use of the `predict` function with the RF-GAP proximities extended to a test set. The number of rows of the proximity matrix must match the number of training + test observations.

The primary purpose of the proximity-weighted predictions is not to circumvent the forest, but to compare the predicted results with those of the forest to assess the learning captured by the proximities. In Chapter 3, we showed that RF-GAP proximity predictions match those of the random forest, while the original and OOB proximities do not.

5.4 Random Forest Imputation

Empirical studies conducted in [66] showed that data imputation via random forests was, on average, 32% more accurate than imputation using auto-associative neural networks and genetic algorithms. We showed that in datasets with missing value percentages less than 75%, RF-GAP proximities always outperformed other random forest proximity-based imputations and usually outperformed when 75% of the data were missing. `rfgap` provides a simple means of imputing missing values using the `rf_impute` function.

Suppose `x` is an $n \times d$ dataframe with missing values. The imputation process for each variable in `x` has two parts. First, if variable j is categorical, missing values are replaced with the most common within-class value. If continuous, the median within-class value is used. After the initial imputation, a random forest is trained using the imputed dataframe and a proximity matrix is constructed. The missing values are imputed using a proximity-weighted sum or majority vote of non-missing variable values. The process is iterative. Repeating the latter step of training the forest and reconstructing the proximities tends to improve the imputation results, but often one to four iterations are sufficient.

The `rfgap` package provides a simple function to run random forest imputation. It is assumed that the missing data takes the form `NA`. The function `rf_impute` requires the dataset with missing values, `x`, vector of associated labels, `y`, the proximity `type` (default is `rfgap`), number of iterations to run the imputation (`n_iters`, default 1), and any additional `ranger` options (`...`). The function returns a dataframe with the imputed values. An additional argument, `x_true`, may be used to supply the true data without missing values. This is used for testing the quality of the imputation. If the user supplies `x_true`, then function returns a list with two elements, the imputed dataframe and the mean-squared error between the true and imputed values.

An example is given below in Listing 5.6 using the `airquality` dataset provided in `R`.

```
library(rfgap)


x <- airquality[, 1:4]
y <- airquality[, 5]
x_imp <- rf_impute(x, y, n_iters = 5, seed = 42)
```

Listing 5.6: The proximity-based random forest imputation applied to the `airquality` `R` dataset. Here we run the iterative imputation 5 times.

5.5 Visualization Via MDS

The original and OOB random forest proximities serve as proper kernel matrices. They are symmetric, positive definite with ones along the main diagonal. RF-GAP proximities may be symmetrized and the proximity value between an observation and itself redefined to serve as a proper kernel. As such, $1 - \mathit{proximities}$ may be viewed as a squared distance in a Euclidean space. We apply these random forest distances, $d(x_i, x_j) = \sqrt{1 - \mathit{prox}(x_i, x_j)}$

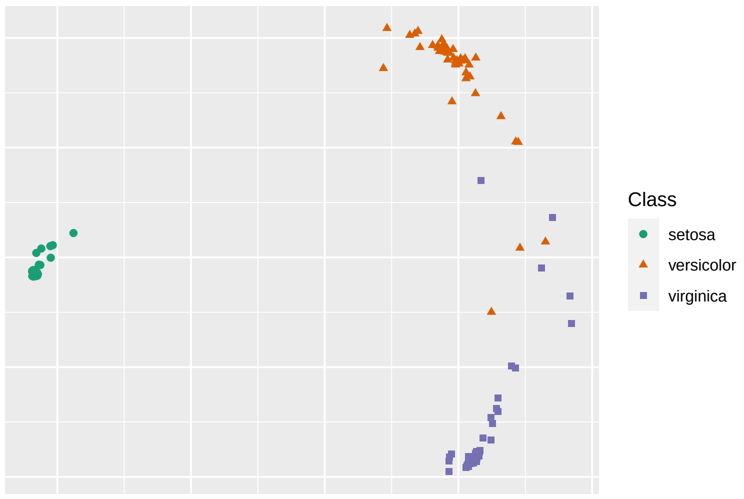


Fig. 5.1: Non-metric MDS applied to RF-GAP proximities generated from the Iris dataset.

to multidimensional scaling using the function `rf_mds`. To use this function, the user may choose to supply a precomputed proximity matrix, a trained `ranger` object, or just the dataframe `x` with labels `y` (`x` is required). If a proximity matrix is not supplied, the user may choose the proximity type (default is RF-GAP). Two types of MDS may be run; metric MDS using the `cmdscale` function from the `stats` package, and non-metric MDS using the `isoMDS` function from the `MASS` packages. The number of dimensions can be selected using the `n_dim` argument (default is 2). The generic `plot` function may be used to generate a scatterplot of the MDS embeddings based on the `ggplot2` package [77]. If the labels, `y`, are supplied, the points will be colored and shaped according to class if `y` is of factor type, or just colored according to scale if `y` is numeric. See an example in Listing 5.5.

```
# Defining the dataframe and labels
x <- iris[, -5]
y <- as.factor(iris[, 5])

# Get the non-metric MDS embeddings
mds <- rf_mds(x, y)
plot(mds, y)
```

5.6 Outlier Detection

Random forest proximities provide a means of assigning outlier scores in a classification setting. Most outlier/anomaly detection methods are unsupervised where class labels are not used or not available. We briefly describe the random forest proximity-based outlier scoring.

1. For a given observation x_i, compute the raw outlier score $\sum_{x_j \in \text{class}(x_i)} \frac{n}{prox^2(x_i, x_j)}$
2. For each class, calculate the median and mean absolute deviation (MAD) of the scores
3. Standardize the raw score, subtracting the mean and dividing by the MAD

The original description of this algorithm (see [47]) suggests carefully observing observations exceeding an outlier score threshold of 10. However, the outlier scoring appears to depend on the dataset, type of proximity, and scaling of the dataset and proximities.

To compute the outlier scores, we use the function `rf_outliers` which takes the a dataframe or `rf_proximities` object, `x`, labels `y`, and proximity type as arguments. The proximity type is ignored if an `rf_proximities` object is supplied. Additionally, the user may provide a pretrained `ranger` if `x` is the data matrix, rather than a proximity matrix. `rf_outliers` returns an object of S3 type `rf_outlier` which is an array of the length of the number of objects in the dataset `x`.

We extend the generic plot function to take the `rf_outlier` S3 class. `plot.rf_outlier` provides a scatterplot of an MDS embedding where point sizes are scaled by the outlier scores. The arguments `min_point_size` and `scale_factor` determine the minimum point size and the factor by which to scale the outlier scores to determine the point sizes of the plot. See example code in Listing 5.7 with resulting Figure 5.2.

```
cars <- mtcars[, -2]
cyl  <- as.factor(mtcars[, 2])


# Computing the outlier scores and plot the MDS embedding
out <- rf_outliers(x = cars, y = cyl)
```

```
plot(out, data = cars, y = cyl)
```

Listing 5.7: Here we determine the RF-GAP proximity outlier scores for the `mtcars` dataset using the number of cylinders as the data labels and subsequently generate a scatterplot using the outlier scores to scale the point sizes.

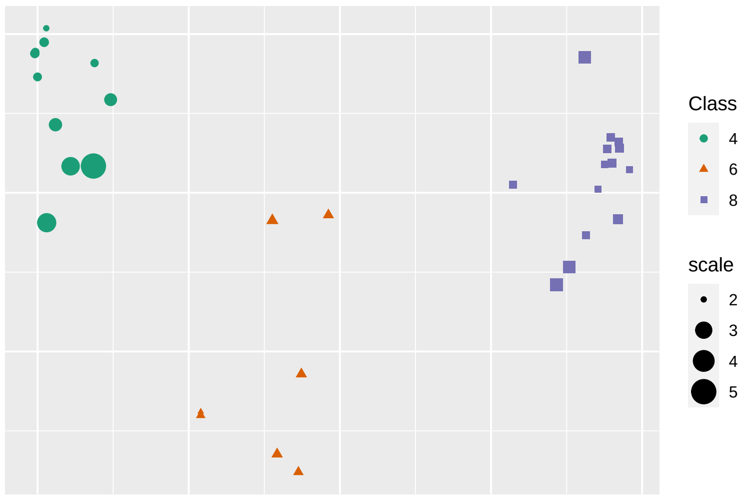


Fig. 5.2: MDS applied to the `mtcars` dataset with point sizes scaled according to the random forest proximity outlier scores.

5.7 Conclusion

The `rfgap` package allows the user to simply generate three types of random forest proximities, including RF-GAP proximities as defined in 3. We make it simple for the user to use and evaluate the most common proximity-based applications including imputation, visualization, and outlier detection. In the future, we will include additional applications, including variable prototyping, artificial data upsampling, and multi-modal learning.

CHAPTER 6
RF-PHATE[1]

6.1 Introduction

In the ever-growing presence of large, high-dimensional data, dimensionality reduction plays an important role in the data preprocessing pipeline. In many cases, high-dimensional data can be well-described in much lower dimensions. In some cases, collected feature variables may be unimportant or uninformative, while in others, redundant information is captured. Dimensionality reduction is typically done in one of two ways: feature selection and feature extraction. In the first case, relevant features pertaining to some task are typically chosen via a backward or forward selection process, as is often done in the context of linear regression. In the latter case, new features are generated from existing variables. Perhaps the most prevalent example of this is principal components analysis (PCA [78]) which performs a change of basis of the original data and the number of components is selected according to the amount of variance explained by the components. Additional feature extraction techniques include multidimensional scaling (MDS [2]), and non-negative matrix factorization (NMF [79]). However, such methods are generally used in preparation for additional downstream tasks and are not usually intended for or adequate to present visual representations of high-dimensional data. Additionally, PCA and NMF are linear methods and thus incapable of modeling non-linear interactions in the latent space.

Non-linear approaches have been designed to overcome this inherent weakness. For example, manifold learning approaches assume that the data are sampled from a low-dimensional manifold embedded in a high-dimensional space and are designed to find the underlying (usually non-linear) manifold structure. Some examples of well-known

[1]Much of the foundational work for this chapter was published in the IEEE Statistical Signal Processing Workshop in July 2021 [19]. The original article can be found at https://ieeexplore.ieee.org/document/9513749. The IEEE does not require individuals working on a thesis to obtain a formal reuse license.

manifold learning approaches include t-distributed stochastic neighborhood embedding (t-SNE [80]), uniform manifold approximation and projection (UMAP [4]), isometric mappings (e.g., Isomap [3]), the potential of heat diffusion for affinity-based transition embedding (PHATE [17]), diffusion maps (DM [18]), and autoencoders [81]).

However, each of these approaches is unsupervised; auxiliary information (e.g., class labels) is ignored or unavailable. Supervised dimensionality reduction approaches use this additional information to further distinguish observational distributions. Supervised adaptations for some manifold learning approaches have been formed. For example, supervised versions of t-SNE, UMAP, and Isomap have been used for both classification and visualization [6, 82–85]. In these examples, class labels are directly incorporated in the distance metric or kernel function used in the construction of the latent space, inducing exaggerated class separation and making the embedding impractical as a preprocessing step for a downstream classification task. Additionally, the proposed kernel constructions are only intended for classification tasks and are not defined for continuous labels.

In spite of the weaknesses of many existing supervised dimensionality reduction algorithms, making use of auxiliary information may be useful in differentiating class-based distributions or recovering information in noisy datasets. We seek to find a supervised distance measure that preserves observational relationships without disturbing the underlying manifold structure. To this end, we introduce a supervised visualization approach that incorporates random forest learning [10] and diffusion-based dimensionality reduction [18,86], following advances from PHATE [17] to extract information through a diffusion process optimized for visualization. Hence, we name our visualization technique RF-PHATE. We extend PHATE's algorithm leveraging a damping factor related to Google's PageRank algorithm [87] to overcome problems related to non-uniform data sampling. Our approach naturally incorporates variable importance from the trained random forest and provides a noise-resilient visualization of the feature space which may be used for data exploration. We show that RF-PHATE outperforms other dimensionality reduction methods in preserving the feature importance relative to the supervised problem.

6.2 Supervised Dimensionality Reduction

Dimensionality reduction algorithms can generally be categorized into three major groups: principal-component-based reduction, matrix factorization, and manifold learning approaches [88]. The first two categories are typically used as data preprocessing steps and are not usually intended for visualization. We discuss those here.

Principal components analysis (PCA) determines a projection onto a linear subspace where the explained variance is maximized [78]. This common approach is often used for preprocessing high-dimensional data. Its frequent usage is due to its quick implementation speed and simplicity. The earliest supervised rendition of principal components analysis (SPCA) performs a subset selection of features that are most highly correlated with the data labels onto which PCA is then applied to generate a new feature space to be used in the regression problem [89]. The primary motivation for this approach was to handle datasets where the number of features exceeds the number of observations ($p > n$), which can be crucial for some downstream tasks. Another variation of SPCA was introduced in [90] based on probabilistic PCA. Here the authors attempt to model the covariance between the data and its associated labels. In [91], a kernelizable variation of SPCA was introduced with the aim of improved classification and visualization. Principal-component-based methods are typically computationally efficient and thus suitable for large data; however, PCA is linear and does not therefore accurately capture non-linear relationships in the data structure in low dimensions. Additionally, the number of components selected is usually determined based on the global variance explained and does not adequately capture local relationships [90].

Another related and common approach is linear discriminant analysis, or LDA [92]. Similar to PCA, LDA seeks a linear combination of features to maximize explained variance but does so based on class labels. An extension of LDA was adapted for high-dimensional data in [93] and shown to be useful in some biological applications. Partial least squares regression (PLS) and partial least squares-discriminant analysis (PLS-DA) are bilinear factor models which transform both the feature and response spaces and seek to optimally fit a linear model to the response variables [94]. PLS is often used in chemometrics [37, 94] and is well-suited for problems with multicollinearity, but is not generally intended for visualization. Both LDA and PLS are not suitable for learning non-linear relationships.

Non-negative matrix factorization or NMF [79] seeks to decompose a data matrix $\mathbf{X}_{n\times p}$ into the product of two matrices $\mathbf{U}_{n\times d}$ and $\mathbf{V}_{d\times p}$, by minimizing the Frobenius norm of the difference between $\mathbf{X}$ and $\mathbf{UV}$ while restricting the entries of $\mathbf{U}$ and $\mathbf{V}$ to be nonnegative. The rows of $\mathbf{V}$ can be regarded as basis vectors while the columns of $\mathbf{U}$ form the axes of the lower-dimensional space [95]. A number of supervised and semi-supervised modifications to NMF (SNMF or SSNMF) have been proposed, such as constrained NMF [96], structured NMF [97], and NMF for constrained clustering [98]. Most of these approaches use the labels in a regularization term in the optimization problem. The authors of [8] proposed a semi-supervised NMF with both similarity and dissimilarity regularization terms. In [99–101] NMF was used for low-dimensional visualization. However, NMF is still a linear method and therefore does not capture the intrinsic geometric structure of nonlinear data [102]. In addition, supervised versions of NMF tend to accentuate class differences in clusters, providing inflated separation between groups, and tight clustering within groups. These supervised approaches are also unsuitable for continuous labels.

6.3 Supervision in Manifold Learning

Manifold-based dimensionality reduction methods attempt to discover the low-dimensional manifold from which the data is sampled. These approaches are capable of modeling non-linear relationships between data points and are suitable for finding visually-meaningful data representations in low dimensions. A few examples are briefly described below.

Isometric mapping or Isomap [3] forms a k-nearest neighbor (k-NN) graph using Euclidean distances and seeks to find the shortest path between nodes, thus approximating the true geodesic distances upon which MDS is applied for dimensionality reduction. T-SNE [80] constructs a graph of similarities between observations in the form of conditional probabilities. A low-dimensional probability distribution (a t-distribution) is found using gradient descent to optimize the KL divergence between the high- and low-dimensional distributions. T-SNE performs well at reconstructing local similarities but tends to lose the global structure. UMAP [4] works similarly to t-SNE but makes use of fuzzy logic and nearest neighbor neighbor-descent (NN-descent) to speed up computations. UMAP maintains local structure and does a better job than t-SNE at capturing global structure, although this is largely attributed to a better initialization strategy [103]. Locally-linear embedding (LLE [104]) uses weighted linear reconstructions of points based on the points' nearest neighbors. These weights are used in an optimization problem to construct the low-dimensional embedding. Diffusion maps (DM [18]) build a kernel (e.g., Gaussian) from a Euclidean k-NN graph to calculate local similarities. The kernel is row-normalized to form a diffusion operator to simulate the transition probabilities of a single step in a "random walk". Eigendecomposition is applied to the powered diffusion operator to map to lower dimensions. PHATE [17] applies a novel kernel function, the α-decaying kernel, to Euclidean distances to learn local similarities before applying diffusion steps to learn the global data structure. MDS is applied to log-transformed rows of the powered diffusion operator (this process is known as finding the potential distance) to form the embedding. Laplacian Eigenmaps form another example where Euclidean k-NN or ϵ-ball distances form a graph to be mapped to lower dimensions via some optimization function.

For most manifold learning algorithms, Euclidean distances are used to form a NN graph to represent local relationships. The supervised counterparts typically use a class-based dissimilarity measure instead of Euclidean distances. In each dissimilarity measure, distances or affinities are defined conditionally upon an instance's class.

For example, S-Isomap (a supervised variation of Isomap) uses a class-conditional variation of a Gaussian kernel function to accentuate the distance between classes and diminish within-class distance and apply the original Isomap algorithm to the NN graph-based constructed from this dissimilarity (see Equation 6.1). This same dissimilarity measure was used in an extension called ES-Isomap [6], in a supervised variation of locally linear embedding called enhanced-supervised LLE (ESLLE [7]), in supervised Laplacian Eigenmaps, S-LapEig [105], and a supervised version of t-SNE [83]:

$$D'\left(\mathbf{x}_i,\mathbf{x}_j\right) = \begin{cases} \sqrt{1-e^{\frac{-D^2\left(\mathbf{x}_i,\mathbf{x}_j\right)}{\beta}}} & y_i = y_j \\ \sqrt{e^{\frac{D^2\left(\mathbf{x}_i,\mathbf{x}_j\right)}{\beta}}} - \alpha & y_i \neq y_j \end{cases} \tag{6.1}$$

Here, $D(.,.)$ denotes a distance function (usually Euclidean, though other distance metrics were tried in [7]), β is typically set to the average distance between data points, and α attempts to diminish separation between similar points of opposing classes.

Other dissimilarity measures have been introduced to artificially exaggerate class separation or understate intra-class distances. For example, the authors of [106] introduced Equation 6.2 as a dissimilarity measure to perform WeightedIso, a supervised variation of Isomap. This dissimilarity measure shrinks the distance between within-class observations by a simple rescaling of the Euclidean distances.

$$D'(x_i,x_j) = \begin{cases} \frac{1}{\alpha}D\left(x_i,x_j\right) & y_i = y_j, \quad \alpha > 1 \\ D\left(x_i,x_j\right) & y_i \neq y_j \end{cases} \tag{6.2}$$

Here the level of α has the effect of shrinking within-class distance while distances between observations of opposing classes remain unchanged.

In [107], the authors introduced a variation of supervised locally linear embeddings (SLLE) using the dissimilarity measure given by Equation 6.3 which additively increases the distance between inter-class observations. The added distance is the maximum inter-class distance scaled by the parameter α. The embedding tends to be sensitive to the choice of α and tends to collapse class clusters if α is too large.

$$D'(x_i,x_j) = \begin{cases} D\left(x_i,x_j\right) & y_i = y_j \\ D\left(x_i,x_j\right) + \alpha \max \mathcal{D} & y_i \neq y_j, \quad 0 \leq \alpha \leq 1 \end{cases} \tag{6.3}$$

where $\mathcal{D}$ is the set of all pairwise distances of the training set.

In each of these approaches, class labels are incorporated to form supervised dissimilarity measures to accentuate class-based distance. The three similarities, given by Equations 6.1, 6.2, and 6.3, are commonly used in supervised manifold learning. In [108], Hajderanj et al. formally analyzed these dissimilarity measures following the theoretical framework of Balcan et al. [109], who developed a theory of kernel functions to assess their quality as similarity measures. They define the "goodness" of a kernel function with respect to a given learning problem, in this case, as a step in dimensionality reduction. Hajderanj et al. show that the dissimilarity measures are not order-isomorphic functions and thus do not provide optimal lower-dimensional representations when applied to manifold learning algorithms. The authors conclude that these measures may aid in downstream classification tasks, but should not be used in visualization as the manifold structure is destroyed.

Instead of directly incorporating class labels into a dissimilarity measure, we propose to use random forests to learn the local data structure. Random forests naturally provide a measure of similarity via the random forest proximities discussed previously. In Section 6.4, we discuss the advantages of using random forest proximities for dimensionality reduction, emphasizing the benefits of using RF-GAP proximity measures from [16].

6.4 Random Forest-Based Dimensionality Reduction

Random forests are highly-effective supervised learners which are easy to train given their flexibility with the variable type (they handle mixed categorical and continuous variables) and little or no parameter tuning [10]. In addition to high predictive accuracy, random forests provide a measure of similarity or proximity. The random forest proximities are supervised measures that are driven by but do not directly incorporate data labels; the decision space of random forests partitions data according to class but the similarity measures are not rescaled or exaggerated conditional upon the class labels as do most of the supervised manifold learning approaches described in Section 6.2 (See Equations 6.1, 6.2 and 6.3). Furthermore, the use of random forests provides a means of generating supervised similarities to unlabeled, out-of-sample points.

Random forests are formed by ensembling randomized binary-recursive decision trees. In each tree, the recursion process partitions or splits the data favorable to class purification (in a classification problem) or goodness of fit (in a regression problem).

In a classification forest, a non-weighted majority vote determines the prediction of a new observation. For regression, an average response value is used. Thus, the random forest can be seen as a nearest-neighbor decision algorithm, where "neighbors" are observations in a shared terminal node. These local neighborhoods form the bases for a supervised similarity measure which may be used as a weighted k-NN graph used in manifold learning.

The space of terminal nodes is used to define a similarity measure between observations. The recursive partitioning based on feature variables organizes observations according to split-points across variables which are most useful at partitioning data according to class purity or goodness of fit. Observations which frequently share terminal nodes are similar to each other with respect to important variables relative to the supervised problem. Observations which never reside together are distant from each other in the same regard. Thus, the frequency in which observations reside in the same terminal nodes is an indication of the closeness of the observations in the context of the supervised task. We emphasize that this distance measure naturally captures variable importance pertaining to the decision space. While two observations may be considered similar in certain dimensions of a Euclidean space, they may have a low proximity value if they differ in variables important to the supervised task. Proximity-based distance is also robust to noise variables, making them useful for high-dimensional data as may be the case in biological contexts.

Analysis of both the originally defined and OOB proximities was performed in Chapter 3, in which we concluded that neither definition accurately reflects the random forest's general learning. In the same chapter, we argue that RF-GAP proximities, which preserve the random forest's decision space, can be used to improve proximity-based applications. However, as the RF-GAP proximities do not form a proper kernel function (they are not symmetric nor positive-definite), we redefine the main diagonal entries as 1 and symmetrize them as a fix. With these adjustments, we can use RF-GAP proximities as a supervised kernel to be used in manifold learning.

Random forests have been shown to behave as a k-NN algorithm with a weighted, adaptive metric [43]. The proximities produced by the random forest can therefore be viewed as locally-adaptive affinities in the predictor space. Manifold learning approaches typically start with a notion of local distance or affinity, often in the form of a k-NN graph. Thus, the random forest proximities prove useful in manifold learning for reducing dimensionality to two or three dimensions for visualization. Indeed, random forest proximities have been widely used in the context of visualization [15,52–54], but almost exclusively in conjunction with MDS, although they have been used in t-SNE [55]. MDS with random forest proximities can provide meaningful visualizations when applied to small, simple datasets, but more advanced manifold learning approaches allow for useful, low-dimensional representations for noisy, large datasets. In Section 6.5, we discuss recent advances in diffusion-based processes optimized for visualization suitable for large datasets.

6.5 Diffusion-based Information Geometry

The diffusion process begins with the construction of a k-NN graph from pairwise Euclidean distances. A kernel function (typically Gaussian with a fixed bandwidth) is applied to the graph to represent observational similarities in an $N \times N$ matrix. The matrix is row-normalized ensuring each row's entries sum to one. This row-stochastic matrix, P, is known as the diffusion operator and represents the probabilities of all possible single-step transitions between observations. High-affinity values between observations suggest that observations are similar and their transition probabilities will also be high. Through a powering process, P^t represents a random walk across the manifold structure where global relationships are learned. Through the powering process, small probabilities are quickly reduced to zero, providing a means of filtering or denoising the learned manifold. Finally, eigendecomposition is applied to the powered diffusion operator to map to lower dimensions.

Although the process is intuitive and works well for dimensionality reduction, diffusion maps has some weaknesses that prevent it from creating good visualizations in many contexts. First, a fixed bandwidth for all points is often not appropriate as the data may not be sampled uniformly. Second, choosing a good time scale t for the diffusion process is difficult and largely overlooked in classical diffusion maps. A small value of t can lead to insufficient denoising and an overemphasis on the local structure while a large value of t can lead to oversmoothing and an overemphasis on the global structure. Third, the eigendecomposition in diffusion maps often places the information learned in P^t into different and higher dimensions [17,110], which is not amenable for visualization. We discuss counters to each of these weaknesses.

Instead of applying a kernel function to Euclidean distances, we encode local similarities as random forest proximities. These locally-adaptive similarities partially overcome the weakness of a fixed-bandwidth kernel function. However, if the data has many partially disjoint clusters, global relationships may not be properly reflected in the transition process. For example, the transition process may favor a single cluster with only transition inlets but no outlets, leading to over- and under-represented clusters after diffusing. We counter this scenario with the introduction of a damping factor, $\beta \in (0, 1]$, inspired by the PageRank algorithm in [87].

Created in the context of ranking internet pages, the damping factor was inspired to overcome the problems of "spider traps" and dead ends, that is, sets of links from which all out-links pointed amongst themselves or towards a single page, forming an inward-facing "cluster" of links. Analogously, we counter poor exit probabilities by redefining the diffusion operator P as $P_\beta = \frac{\beta}{N} P + \frac{(1-\beta)}{N} \mathbf{1}\mathbf{1}^T$, where $\mathbf{1}$ is a vector of ones of length N and β has a default of 0.9. Here, the transition probabilities allow for random jumps or "teleportation", creating small exit probabilities from isolated clusters or dead-ends.

To properly select t, we follow the inspiration of PHATE [17], which uses von Neumann Entropy (VNE) of the diffused operator to provide a good choice of t for visualization. The VNE of the diffused operator P_β^t is the Shannon entropy of the normalized eigenvalues of P_β^t. Since the entropy of a discrete random variable is maximized with a uniform distribution, the VNE is a soft proxy for the number of significant eigenvalues of P_β^t. The more significant eigenvalues there are the closer the normalized eigenvalues are to a uniform distribution, and the higher the VNE. Typically, t is chosen to be around the transition from rapid to slow decay in the VNE as this is considered to be a point in the diffusion process where noise has been eliminated and oversmoothing begins [17].

To overcome the weaknesses stemming from the eigendecomposition in standard diffusion maps, we apply an information distance to the powered diffusion operator P_β^t to create the potential distance [17]. The potential distance is calculated by applying a log-transformation on P_β^t and then calculating the Euclidean distance between rows, although other information distances can also be used [111]. The potential distance is sensitive to differences in both the tails and the more dense regions of the diffused probabilities, resulting in a distance that preserves both local and global relationships. These distances are then embedded into low-dimensions using metric MDS, as is done in Isomap [3]. This extracts the information in low dimensions for better visualization.

The RF-PHATE algorithm proceeds as follows. Given the training dataset $\mathcal{M} = \{(\mathcal{X}, \mathcal{Y})\} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots (\mathbf{x}_N, y_N)\}$, we generate the low-dimensional embedding G following the steps below.

1. Train the random forest (RF) on $\mathcal{M}$
2. Generate the RF-GAP [16] proximities ($\mathcal{P}$) from the RF
3. Row-normalize $\mathcal{P}$ to form the initial diffusion operator P
4. Apply damping to form $P_\beta = \frac{\beta}{N} P + \frac{(1-\beta)}{N} \mathbf{1}\mathbf{1}^T$
5. Perform t steps as selected using VNE to form P_β^t
6. Determine the potential distance D_t from P_β^t
7. Form the embedding G applying MDS to D_t

6.6 Experimental Results

Here we demonstrate the utility of RF-PHATE in data exploration. The use of random forests aids the method's ability to capture variable importance in the lower dimensional embedding. In Figure 6.1, we display the RF-PHATE embedding of the Optical Handwritten Digits training dataset [51] which consists of 3828 handwritten digits (0 - 9) scaled down to 8×8 pixels. Each pixel is assigned a value from 0 to 16 depending on the gray-scale intensity. The trained random forest identified the features V44, V22, and V43 as the most important features to classify the digits. In light of the RF-PHATE embedding, V44 and 43 appear to be important in distinguishing the digits 3, 5, and 9 from the remaining digits. The figure also shows that V22 can be used to distinguish digits 5 and 6 from the rest. These particular pixel locations make sense for these distinctions, given they represent values in the lower-left hand part of the digit, and upper right, respectively (see Figure 6.2).

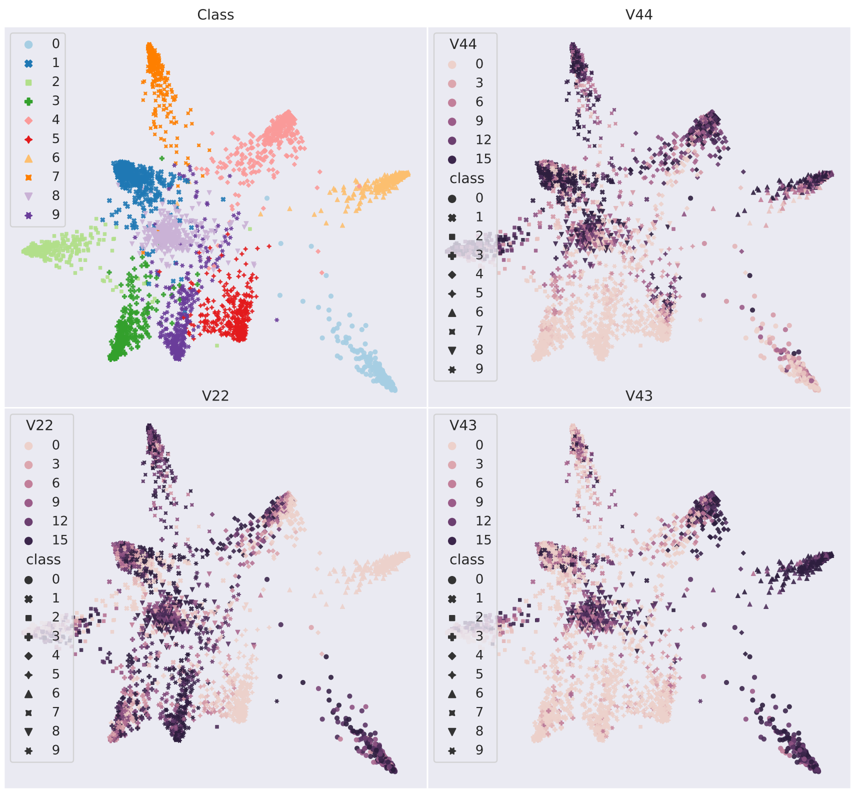


Fig. 6.1: The RF-PHATE embedding of the Optical Handwritten Digits dataset colored by digit label and the top three important variables for classification, respectively. Here we see that pixel 44 can be used to distinguish digits 3, 5, and 9 from the remaining digits. Pixel 22 appears to be useful at discriminating digits 5 and 6 from the rest. This makes sense in light of the positions of these pixels, as demonstrated in Figure 6.2

In Figure 6.3, we color the RF-PHATE embedding of the Car Evaluation Database [112] according to class and important variables. The dataset consists of 1728 instances of cars with six variables including buying price, maintenance costs, number of doors, person capacity, luggage boot size, and safety evaluation. The object of this dataset is to predict the

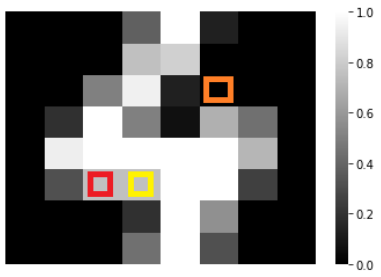


Fig. 6.2: An example image from the Optical Handwritten Digits dataset with pixels 43, 22, and 44 highlighted in red, orange, and yellow, respectively. Viewed in conjunction with the RF-PHATE embedding in Figure 6.1, it is clear to see why pixels 43, and 44 may be used to discern digits 3, 5, and 9 from the others and why pixel 22 helps differentiate between 5, 6, and the rest.

car acceptability level. According to the trained random forest, the most important variable for determining the car's acceptability level is the safety rating. The next two important variables are the car's carrying capacity and price. It is clear in the figure that the safety rating partitions the dataset in two, with high safety ratings in the right-most clusters and medium-low safety on the left. Cars that only hold two people are always considered unacceptable according to the dataset ratings. This can be seen in the small cluster on the bottom left of the graph. After accounting for both safety and capacity, the smaller clusters are then split into buying price levels. Cars that are labeled "very good" always have either a low or medium buying price, but never a high or very high price.

Unsupervised dimensionality reduction methods tend to fail in the presence of noisy variables. Supervised methods such as PLS and LDA determine class-based discriminating features and are thus capable of handling noise variables. However, such linear methods are not optimized for low-dimensional visualizations. In noisy settings, supervised manifold learning approaches that use the dissimilarity measures given by Equations 6.1, 6.3, and 6.2 tend to put same-class points into very tight clusters, while at the same time perfectly separating classes. This can be suitable for downstream classification but does not accurately exhibit the data's structure.

Due to random forest's noise resilience, RF-PHATE can produce meaningful embeddings even in the presence of noise variables. In Figure 6.4, we display the RF-PHATE embedding of the Iris [113] dataset with an additional 500 noise variables, each sampled

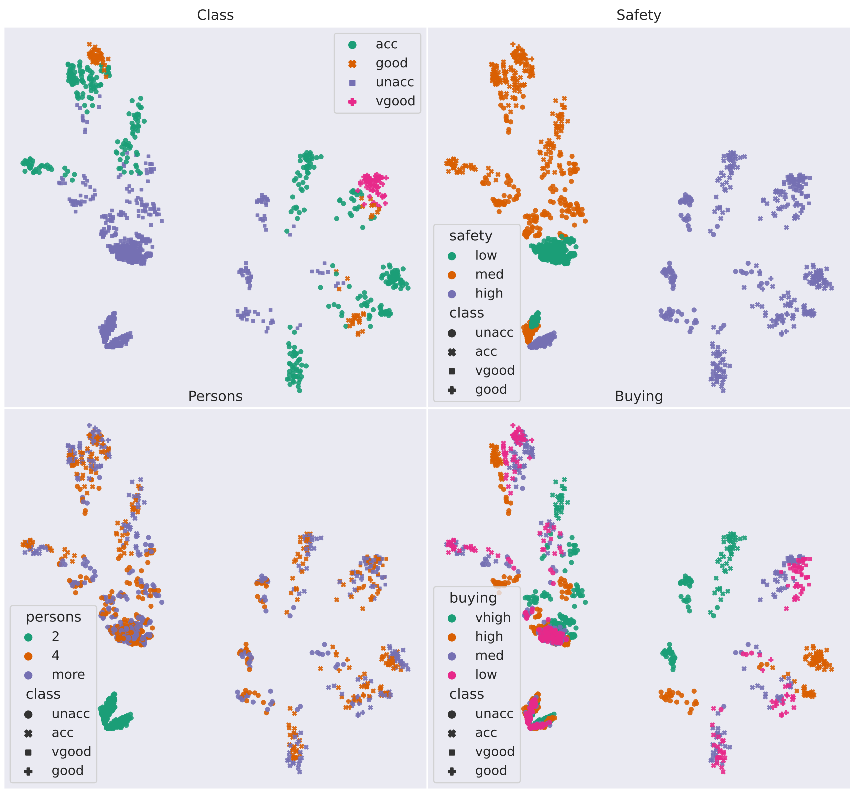


Fig. 6.3: An RF-PHATE embedding of the Car Evaluation Database. The top important variable, safety rating, partitions the embedding into two linearly-separable sections between the "high" class and the remaining. From here, the data is conditionally partitioned based on capacity and then buying price. Any car with a person capacity of two is automatically assigned the label "unacceptable". "Very good" labels are always associated with high safety ratings and low or medium buying prices.

from a uniform [0, 1] distribution. The original four variables of the Iris dataset are also normalized to the same scale. In the same figure, we compare 18 additional embeddings on the same dataset, 7 supervised and 11 unsupervised. The RF-PHATE embedding closely resembles the expectation of the familiar data's structure.

6.7 Quantifying an Embedding's Fit

The quality of an embedding can be difficult to quantify. In an unsupervised setting, order-preserving or rank-based distance correlations may be used to determine goodness of fit, such as Mantel's test [114] or DEMaP [17]. However, in supervised settings, the data distributions may be class-conditional and the goals of downstream tasks may not align

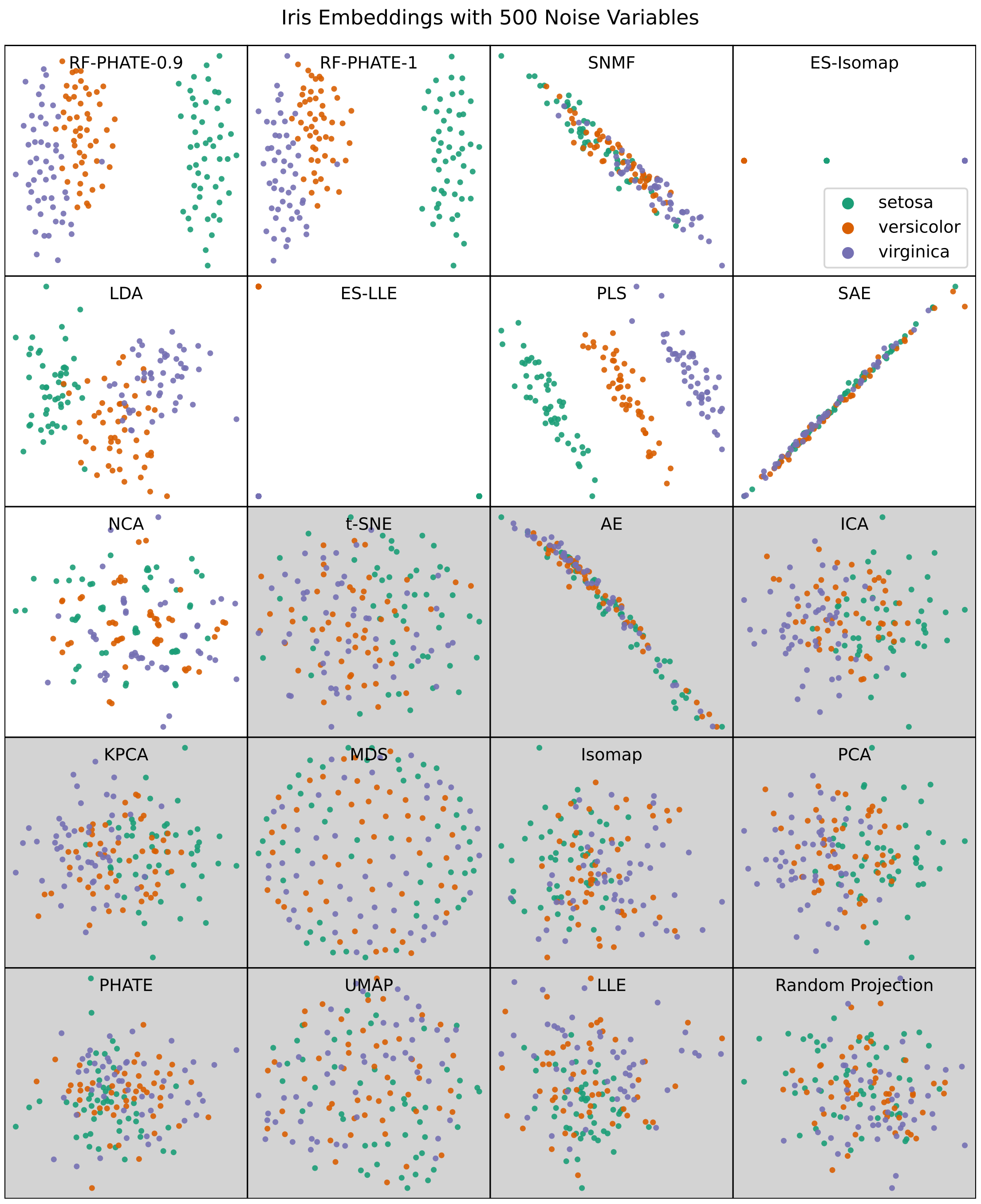


Fig. 6.4: 20 embeddings of the Iris dataset with an additional 500 uniformly distributed noise variables. Supervised embeddings are shown with white backgrounds, while unsupervised are shown with grey backgrounds. The RF-PHATE embeddings (shown with β values of 0.9 and 1) exhibit a structure true to the data. None of the unsupervised embeddings display any meaningful visualizations. Class-dissimilarity manifold learning approaches (ES-Isomap and ES-LLE) completely destroy the data structure.

with those of unsupervised settings. In some supervised cases, a k-NN classifier is used to assess the quality of the embedding [6, 7]. However, this only assesses the local data structure and is not a global indicator of fit. Here we present a new approach to quantify supervised embeddings, using a local means (k-NN) to describe a global measure of variable importance.

Practitioners often seek a selection of variables that are best suited for a prediction task. It is often the case that noisy or otherwise unusual variables are collected alongside useful data. To assess supervised embeddings, we determine to what extent important variables are used in the embedding structure. One means of ascribing variable importance is through a permutation process. Permutation feature importance performs a random permutation across all feature variables taken one at a time. The model's score (error or loss) is assessed after each permutation. If the error is significantly increased after a certain variable permutation, that variable is deemed important for the supervised task.

We assess the embedding quality by determining to what extent variables important to the supervised task are used in the embedding generation. That is, embeddings useful in classification should also be useful in constructing the embedding space. The process is described below.

1. Determine the permutation variable importance scores using a k-NN classifier
2. Use a k-NN regressor to predict the embeddings space using the original data features
3. Run the permutation variable importance scores using the k-NN regressor
4. Compute the correlation between the two sets of importance scores

To summarize our results, we computed the importance correlation scores using 19 datasets from the UCI repository [51]. The permutation scores were run 10 times per dataset. We normalized each correlation, ρ, across each dataset, by differencing with the maximum correlation within the dataset. That is, we compute $\bar{\rho} = \rho_{max} - \rho$ (lower is better). We present the summarized results in Figure 6.5. Here we see that RF-PHATE best captures variable importance in low dimensions. Additionally, and not unexpectedly, each of the supervised methods outperforms the unsupervised ones at this task.

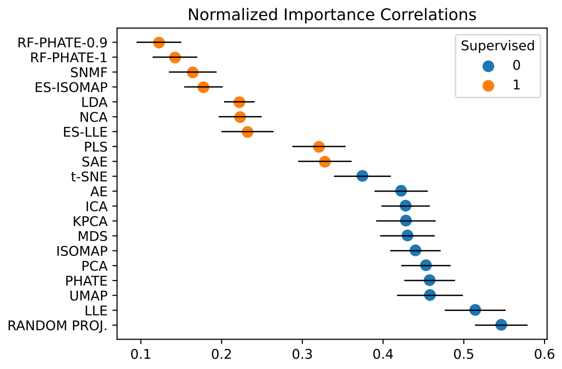


Fig. 6.5: Summarized results of the normalized importance correlations across 19 UCI datasets (lower is better) reduced to two dimensions. RR-PHATE with $\beta = 0.9$ most consistently captures variable importance in lower dimensions. As expected, each of the supervised embeddings outperform the unsupervised ones according to this measure.

6.8 Out-of-Sample Extension

Modern manifold learning algorithms produced fixed coordinates in the latent space, but do not typically provide a means to extend to new observations. That is, to incorporate previously unseen data in the embedding, the algorithm must be rerun in its entirety. Additionally, the use of a full pair-wise distance matrix can be impractical when working with large datasets. To overcome these issues, we provide an autoencoder extension of RF-PHATE inspired by geometry-regularized autoencoders (GRAE) [115]. In the paper, the authors use a learned manifold embedding as a regularization term added to the reconstruction loss of a standard autoencoder.

An autoencoder (AE) is a type of artificial neural network often used for dimensionality reduction. Autoencoders take an input, X, and learn an encoder function to compress the input into a lower dimensional space while simultaneously learning a decoder function to map the lower-dimensional embedding to the original input space. That is, defining an encoder function $f_e(X)$, a decoder function $f_d(Z)$, and a loss or objective function

$L(f_e, f_d) = L_{recon}(X, f_d(f_e(X))$, the vanilla autoencoder seeks to minimize $L(f_e, f_d)$ via stochastic gradient descent or a variant thereof, where L_{recon} is the reconstruction loss (e.g., MSE).

Given a learned geometric embedding, G, GRAE defines the autoencoder's loss function as $L_g(f_e, f_d) = L_{recon}(X, f_d(f_e(X)) + \lambda L_{geom}(f_e(X), G)$. Here, L_{geom} prevents f_e from learning a latent space that differs drastically from the provided manifold, and the parameter λ determines the level at which the manifold is used to encode X. The purpose of GRAE is to provide a manifold learning approach that is both extendable (i.e. able to generate embedding coordinates for previously unseen points) and invertible (the original points may be constructed via the decoder). We seek to use the regularized autoencoder as an out-of-sample extension, seeking a good approximation for the RF-PHATE embedding.

To incorporate the random forest's learning, we adapt the geometry-regularized autoencoder's structure to reconstruct random forest proximities instead of the original points. Using a trained random forest, we can easily extend proximities to previously unseen data. The proximity measures from the new points to the original training data provide a starting point to extend the embedding while incorporating the random forest's learning.

To extend RF-PHATE to a set $\mathcal{X}_E$, a random forest is first trained on $\mathcal{M} = \{(\mathcal{X}, \mathcal{Y})\} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots (\mathbf{x}_N, y_N)\}$ from which the proximities, $\mathcal{P}$, are extracted. An initial RF-PHATE embedding is generated from these proximities following the steps described in Section 6.5. An autoencoder is then trained to reconstruct the proximities with the RF-PHATE embedding as regularization in the geometry-regularized loss function L_g to use the intrinsic manifold structure to guide the autoencoder's learning. We attempted adaptations to GRAE in order to better preserve the random forest's learning in the AE.

Here we briefly describe our variations of the GRAE algorithm. The first variation adds a linear layer for regression to the bottleneck layer. In addition to the reconstruction, the model attempts to predict the random forest proximities directly from the bottleneck layer. The second approach takes the RF proximities as the initial input and attempts to reconstruct the original data. We call this architecture PROXLAYER. The final approach reconstructs the RF proximities and uses proximity-based landmarks from the training set; we call this architecture RFPROXCON (random forest proximity reconstruction). We find this architecture provides the truest RF-PHATE embeddings and describe the model in more detail below.

Let $\mathbf{x}_E \in \mathcal{X}_E$. For each $\mathbf{x} \in \mathcal{X}$, we construct a set of proximities $\{p(\mathbf{x}_E, \mathbf{x})\}_{\mathbf{x} \in \mathcal{X}}$ using the trained random forest. We find that the use of proximities in the reconstruction provides an embedding truer to the original RF-PHATE embedding, as measured by using Mantel's test [114]. However, for large training sets, it is impractical to reconstruct the full set of proximities $\{p(\mathbf{x}_E, \mathbf{x})\}_{\mathbf{x} \in \mathcal{X}}$. To combat this, we provide use a proximity-based landmark construction to select observations to which we construct the proximities.

We wish to select landmarks that are good representatives of each class. Since the proximities define a measure of similarity, we select the observations with the highest average within-class proximity values. We thus only need to reconstruct a proximity vector of length $N_{landmark}$ when training the autoencoder. Not only does this approach speed up training time, but the proximity-based landmarks provide more stereotypical class representations, further denoising the learned embedding. We provide Figure 6.6 to demonstrate RFPROXCON architecture.

Figure 6.7 shows an example comparing a 2D RF-PHATE embedding using 20% of the MNIST handwritten digit training dataset (12,000 observations) and extending to the remaining 80%. Here we used 1,200 landmarks (10%) for the proximity reconstruction. The landmarks show typical examples from each of the 10-digit classes. The extended embedding (on the out-of-sample 80%) takes the same basic form as the original embedding formed by the 20% sample. The neural network is able to interpolate areas where no existing points lay in the original embedding.

To assess the different architectures, we compare the neural network embeddings on a test set to those generated using the full RF-PHATE algorithm. Mantel's test allows us to compute the correlations between matrices, in this case, between the matrices of pair-wise distances of the constructed embeddings. We aggregated results using our four architectures

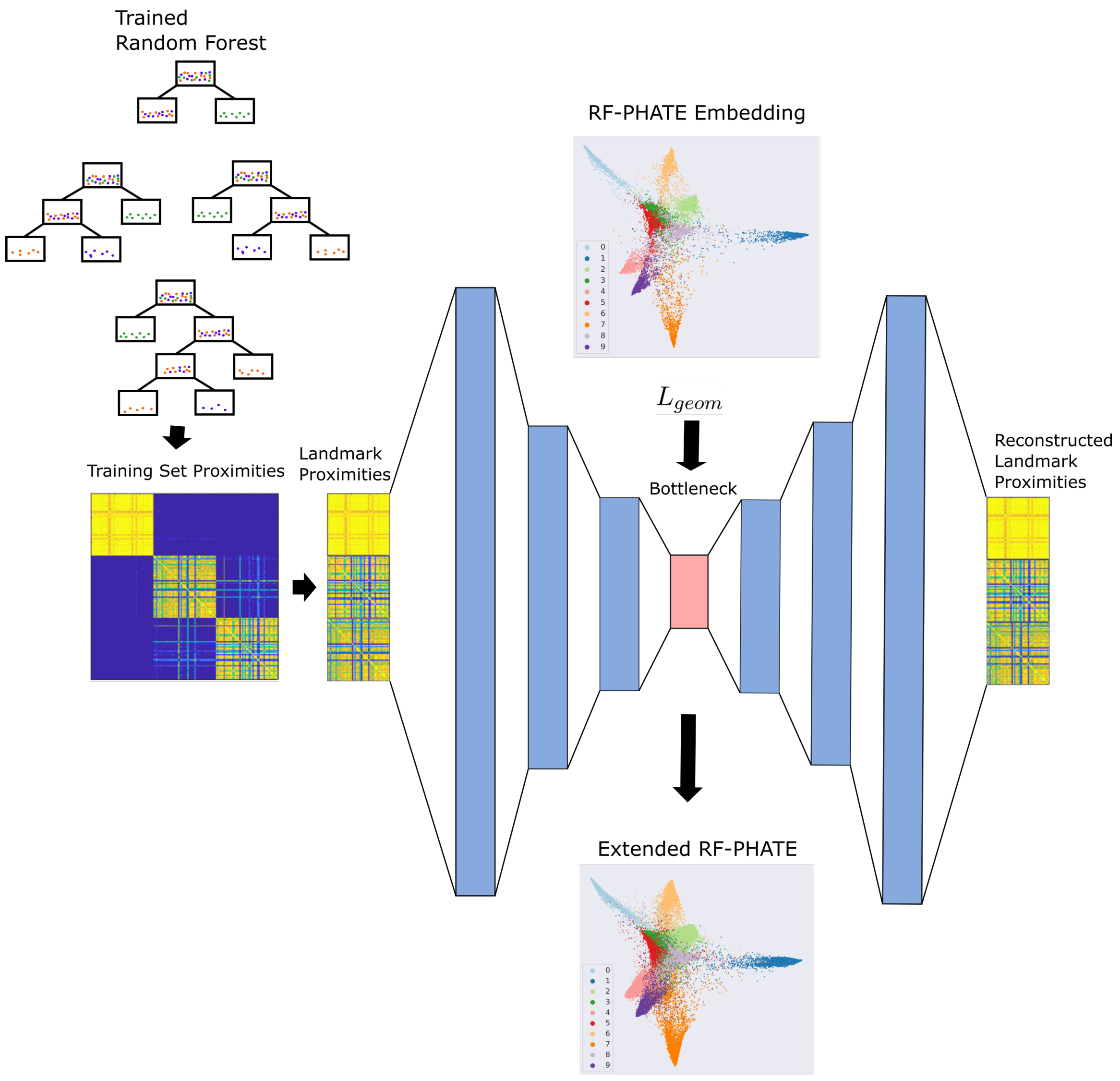


Fig. 6.6: The RF-PHATE autoencoder-based extension, RFPROXCON. The autoencoder reconstructs the proximity landmarks selected as those observations which have the highest within-class proximity mean. The bottleneck layer is regularized by the RF-PHATE embedding at the time of training. Once trained, the embedding may be extended to previously unseen observations.

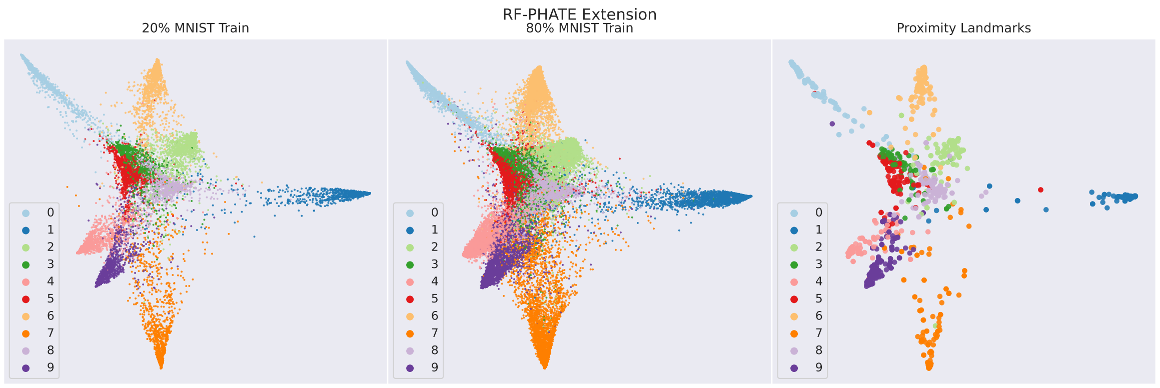


Fig. 6.7: A representation of the RF-PHATE geometry-regularized autoencoder embedding. Here we used a small subset (20%) of the MNIST training dataset to construct the RF-PHATE embeddings for the regularization of the autoencoder. Of these, 10% were selected as prototypical landmarks based on the highest average within-class proximity values. The center plot shows the embedding extended to the remaining 80% of the data, while the right plot shows the landmark points used for the proximity reconstruction.

across 21[2] datasets, five λ values (1, 10, 100, 1,000, and 10,000) and five repetitions each. We split each dataset into 70% training data and 30% test data. We trained each neural network using the training data and generated embeddings using each network and the full RF-PHATE algorithm. The Mantel test was used to compare the pairwise distances between the neural network test embeddings and the RF-PHATE test embeddings. We found that reconstructing the proximities using proximity-based landmarks in RFPROXCON provided the most consistent results. See the summarized results in Figure 6.8.

6.9 Conclusion

The majority of dimensionality reduction algorithms, including manifold learning approaches, are unsupervised, making no use of auxiliary class information. Existing supervised manifold learning algorithms form dissimilarity measures conditional upon class labels which distorts the intrinsic data manifold, inducing exaggerated class separation and producing asymmetric affinities. In this chapter, we formed a random-forest-based supervised manifold learning algorithm directly incorporating geometry- and accuracy-preserving

[2]Most of these datasets come from the UCI repository [51], including Banknote, Breast Cancer, Car, Diabetes, Ecoli, Glass, Heart Disease, Hill Valley, Ionosphere, Iris, Liver, Optdigits, Parkinson's, RNA-Seq, Seeds, Tic-Tac-Toe, Waveform, and Wine. Additional datasets include a Raman spectra cellular dataset used in [37], a version of the Titanic dataset from Kaggle, and an artificial tree dataset from [17].

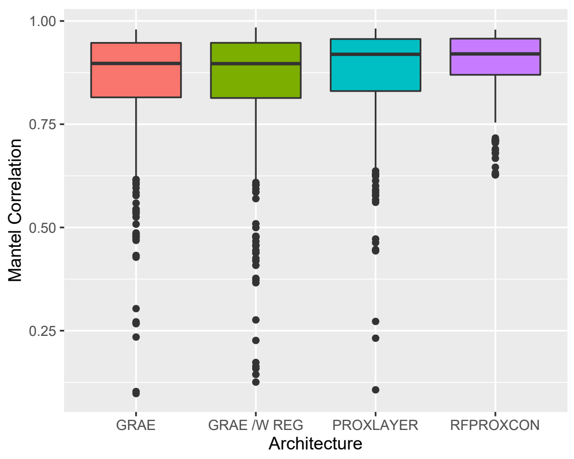


Fig. 6.8: The aggregated Mantel test results across 21 datasets, 5 λ values (1, 10, 100, 1,000, 10,000) and five repetitions each. The Mantel test was applied to the pairwise distances generated using both the neural network embedding and the full RF-PHATE algorithm applied to a test set. Higher correlations are associated with better distance preservation in the embedding. We compared the original GRAE network using the RF-PHATE embedding for regularization, GRAE with an additional regression head used to predict the proximity values (GRAE W/ REG), an adaptation that takes the proximities as inputs (PROXLAYER), and RFPROXCON, which reconstructs landmark proximities. Each was trained for 100 epochs in each experiment. Here, RFPROXCON provides the best and most consistent results.

random forest proximities (RF-GAP proximities [16]) to form an information geometry inspired by PHATE [17], calling our approach RF-PHATE. RF-PHATE outperforms existing supervised and unsupervised dimensionality reduction methods in preserving the variable importance structure relative to the supervised task. Additionally, RF-PHATE is relatively robust to noise and requires little to no parameter tuning to produce visually meaningful results. Via a neural network, we are able to extend this embedding to new points without rerunning the full algorithm, making RF-PHATE scaleable to larger datasets.

CHAPTER 7

CONCLUSION AND FUTURE WORK

This dissertation is comprised of two broad topics, random forest proximities and supervised dimensional reduction. In Chapter 2 we discussed the reasons why random forests remain relevant in modern machine learning where neural networks generally make headline appearances. We provided the background necessary to understand random forest proximities, their construction, applications, and how they encode information in supervised learning. Previously defined proximities (described in the writeup as the “original” and “oob” proximities) provide an intuitive way of transcribing a random forest’s learning as pairwise affinities, but cannot directly explain the random forest’s prediction. For this reason, we concluded that such proximities do not truly reflect the forest’s learning. As a consequence, applications using these proximities do not mirror the decision space of the forest, although they are constructed via this decision space.

In light of this, we developed a random forest proximity measure (RF-GAP proximities) which does encode the voting points of the random forest and may be used to reconstruct the forest’s predictions (see Chapter 3). The importance of this fact is not to show we may circumvent the forest to make predictions, the random forest provides many more benefits to the user in addition to prediction and pair-wise affinities are more costly to store than the forest, instead, this affirms that applications making use of RF-GAP proximities better represent the random forest.

We showed that RF-GAP proximities outperform other forest-based affinities in data imputation, demonstrated their utility in detecting outliers, and can be used to visualize relationships between observations and their variables which are important to their classification (via MDS) See Chapter 4. In regards to RF proximities, we:

- Defined a new RF proximity (RF-GAP) which preserves the RF-learned data geometry

- Proved that RF-GAP proximities perfectly reconstruct RF predictions when used as weights in a NN predictor. This was also verified empirically.
- Showed that RF-GAP proximities improve data imputation when compared to existing proximities
- Provided evidence to suggest RF-GAP improves common proximity-based applications, including imputation, visualization, and outlier detection
- Wrote an R package to develop and compare random forest proximities (see work at `https://github.com/KevinMoonLab/RF-GAP`)

Although visualization via MDS provides some insight to the random forest's learning, directly encoding low-dimensional embeddings this way is less beneficial for larger and more complex datasets. In these situations, manifold learning approaches have been proven useful. While most manifold learning approaches are unsupervised, we discussed variations that incorporate class labels in various similarity measures in Chapter 6. These supervised manifold learning adaptations disrupt the intrinsic data manifold, or at best encode different, class-based manifold structures.

Instead of artificially disrupting the manifold structure based on class, we proposed to learn a supervised manifold that naturally encompasses supervision variable importance via random forest proximities, using RF-GAP proximities in conjunction with diffusion-based manifold learning. This supervised dimensionality reduction approach is a diffusion process following visually-optimizing advances from PHATE [17] and is thus called RF-PHATE. We showed that RF-PHATE preserves variable importance in low dimensions better than any other compared supervised or unsupervised dimensionality reduction approach. Additionally, we demonstrated its utility as a visual aid in understanding a dataset's structure and spatially relevant features. In summary, the primary results in Chapter 6 consisted of:

- Constructing a novel approach to supervised dimensionality reduction based on diffusion and random forest proximities, called RF-PHATE.

- Defining a new method for quantifying the fit of low-dimensional embeddings in a supervised context.
- Demonstrating the utility of RF-PHATE in exploratory analysis.
- Providing an out-of-sample approach to embed new points using a proximity-reconstructing autoencoder.

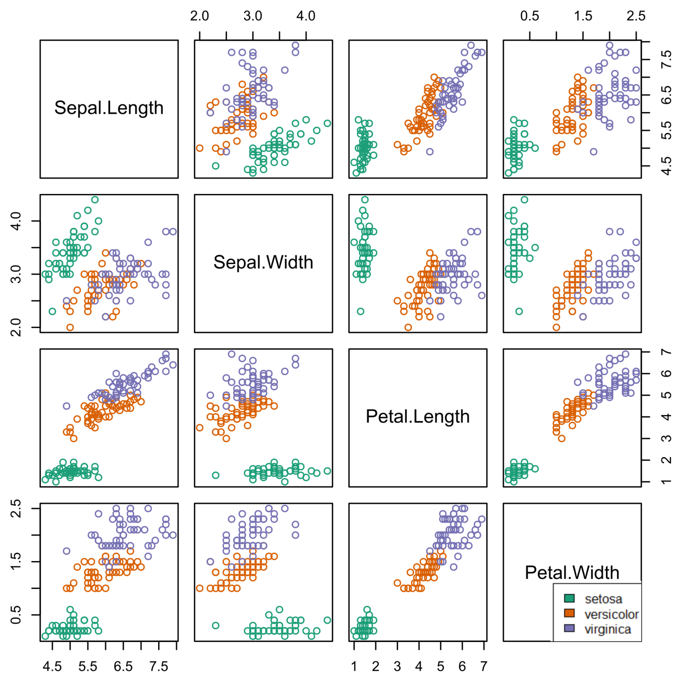
In addition to their use in RF-PHATE and the above-described applications, RF-GAP proximities may benefit other applications as well. We describe a few items for future work regarding these proximities in Section 7.1.

7.1 Future Works

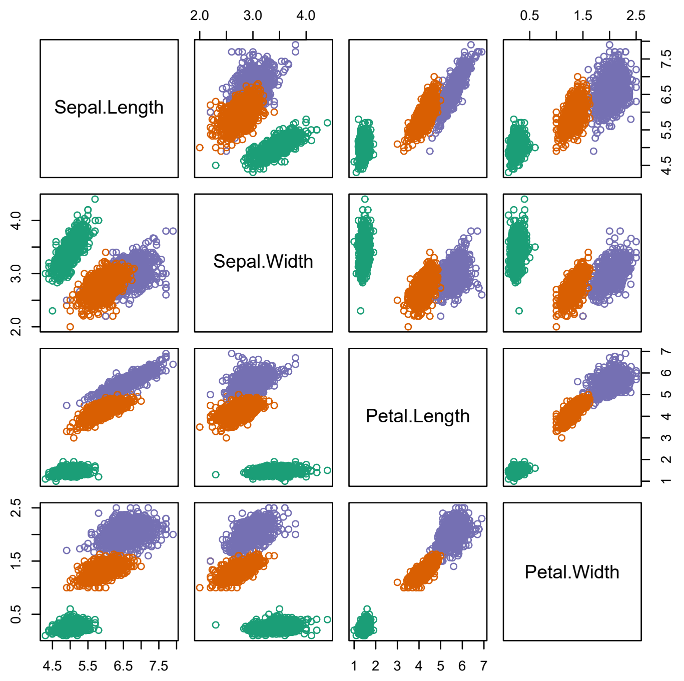
Many datasets suffer from unbalanced class labels. In extreme examples, where a particular class is very uncommon, a classification algorithm can attain near-perfect predictive accuracy by always predicting the dominant class. One method to overcome this issue is to downsample the majority class. However, this is often not feasible especially when data is limited. Another approach is to artificially upsample classes with fewer observations. Synthetic minority over-sampling technique (SMOTE) [116] is a randomized k-NN-approach method for upsampling.

RF proximities can be used for this problem by randomly selecting same-class observations and using proximities to weight the feature variables to simulate a new observation. Although we have not yet extensively explored this approach, Figure 7.1b provides a compelling example that RF upsampling may provide meaningful samples. Here we provide the pair-wise variable plot of the Iris dataset along with that of 1000 RF-GAP-generated artificial samples. The class-based, variable distributions appear to be similar to those of the original dataset.

Multi-modal learning involves collecting and analyzing related data from different spaces. For example, RNA-sequencing data may be collected alongside clinical data related to diseased persons. Both of these feature spaces are related to the same individual or class of individuals, but the features themselves are different and somehow incompatible to



(a) Pairwise variable plot of the iris dataset.



(b) 1000 artificial points created using proximity-weighted sampling.

Fig. 7.1: A comparison of the pairwise variable plot of the Iris dataset with that of the upsampled dataset. Here 1000 samples were generated using RF-GAP proximities as weights.

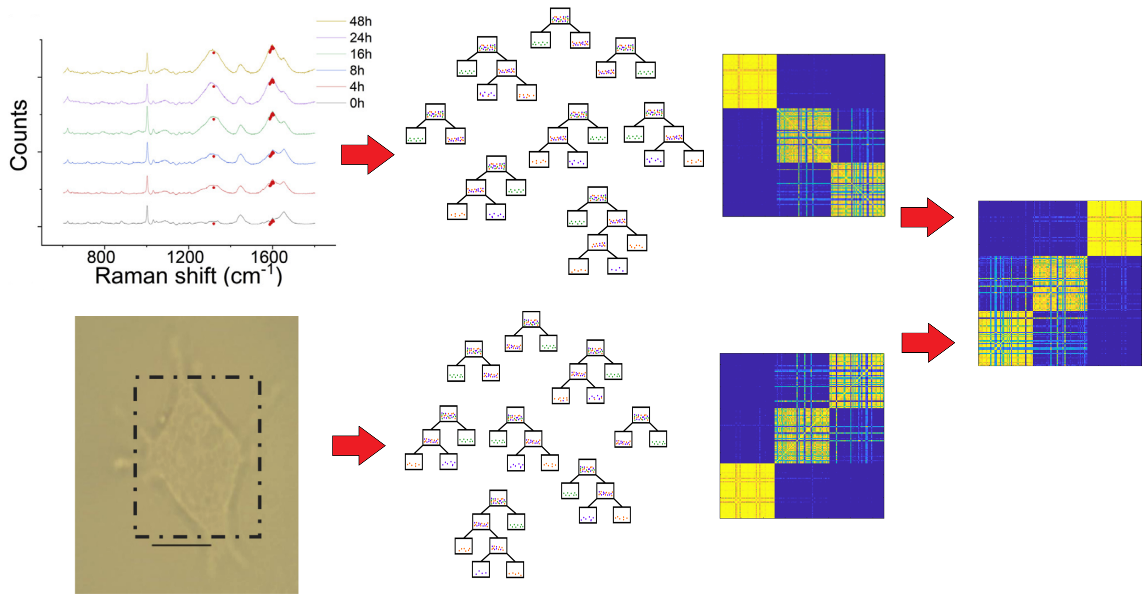


Fig. 7.2: A depiction of the combining of spectral features with those of images via random forest proximities.

directly join together in a tabular format. In such cases, information relative to the supervised task from each domain must be extracted separately and is often done so via neural networks (see [117] for a survey), but random forest proximities have also been used in some contexts [13, 14]. The proximities are used to form a space in which training features relevant to both views are combined, either by addition or linear combination. Since RF-GAP proximities preserve random forest learning, they may prove useful in concatenating learned features to improve classification or other tasks. Figure 7.2 provides a symbolic example of combining proximities to join spectral features with images.

Random forest proximities may be supplied in any machine learning method which makes use of pairwise distances or similarities. They naturally encode variable importance relevant for supervised models without artificially disrupting the underlying data structure. We will continue to seek applications fitting these supervised similarities and hope to find additional benefits of RF-GAP proximities.

REFERENCES

[1] M. Hearst, S. Dumais, E. Osuna *et al.*, "Support vector machines," *IEEE Intell. Syst.*, vol. 13, no. 4, pp. 18–28, 1998. [Online]. Available: http://doi.org/10.1109/5254.708428

[2] J. Kruskal and M. Wish, *Multidimensional Scaling*, ser. Multidimensional Scaling. Newbury Park, California: Sage Publications, 1978, no. 11.

[3] J. B. Tenenbaum, V. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000. [Online]. Available: https://doi.org/10.1126/science.290.5500.2319

[4] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," *arXiv*, vol. abs/1802.03426, 2018. [Online]. Available: https://arxiv.org/abs/1802.03426

[5] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th Int. Conf. on Mach. Learn. - Volume 70*, ser. ICML'17. JMLR.org, 2017, p. 214–223.

[6] B. Ribeiro, A. Vieira, and J. Carvalho das Neves, "Supervised isomap with dissimilarity measures in embedding learning," in *Progress in Pattern Recognition, Image Analysis and Applications*, J. Ruiz-Shulcloper and W. G. Kropatsch, Eds. Berlin, Heidelberg: Springer, 2008, pp. 389–396. [Online]. Available: https://doi.org/10.1007/978-3-540-85920-8_48

[7] S. Zhang, "Enhanced supervised locally linear embedding," *Pattern Recognit. Lett*, vol. 30, no. 13, pp. 1208 – 1218, 2009. [Online]. Available: https://doi.org/10.1016/j.patrec.2009.05.011

[8] Y. Jia, S. Kwong, J. Hou *et al.*, "Semi-supervised non-negative matrix factorization with dissimilarity and similarity regularization," *IEEE Trans. Neural. Netw. Learn. Syst.*, pp. 1–12, 2019. [Online]. Available: http://doi.org/10.1109/TNNLS.2019.2933223

[9] J. Goldberger, S. Roweis, G. Hinton *et al.*, "Neighbourhood components analysis," in *Adv. Neural. Inf. Process. Systs.*, ser. NIPS'04. Cambridge, MA, USA: MIT Press, 2004, p. 513–520.

[10] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001. [Online]. Available: http://doi.org/10.1023/A:1010933404324

[11] A. Cutler, D. R. Cutler, and J. R. Stevens, "Random forests," in *Ensemble Machine Learning: Methods and Applications*, C. Zhang and Y. Ma, Eds. Boston, MA: Springer US, 2012, pp. 157–175. [Online]. Available: https://doi.org/10.1007/978-1-4419-9326-7_5

[12] A. Liaw and M. Wiener, “Classification and regression by randomForest,” *R News*, vol. 2, no. 3, pp. 18–22, 2002.

[13] H. Cao, S. Bernard, R. Sabourin *et al.*, “Random forest dissimilarity based multi-view learning for radiomics application,” *Pattern Recognit.*, vol. 88, pp. 185 – 197, 2019. [Online]. Available: https://doi.org/10.1016/j.patcog.2018.11.011

[14] K. R. Gray, P. Aljabar, R. A. Heckemann *et al.*, “Random forest-based manifold learning for classification of imaging data in dementia,” in *Machine Learning in Medical Imaging*, K. Suzuki, F. Wang, D. Shen *et al.*, Eds. Berlin, Heidelberg: Springer, 2011, pp. 159–166. [Online]. Available: https://doi.org/10.1007/978-3-642-24319-6_20

[15] T. Shi, D. Seligson, A. S. Belldegrun *et al.*, “Tumor classification by tissue microarray profiling: random forest clustering applied to renal cell carcinoma,” *Mod. Pathol.*, vol. 18, no. 4, pp. 547–557, Apr 2005. [Online]. Available: https://doi.org/10.1038/modpathol.3800322

[16] J. S. Rhodes, A. Cutler, and K. R. Moon, “Geometry- and accuracy-preserving random forest proximities,” 2022. [Online]. Available: https://arxiv.org/abs/2201.12682

[17] K. R. Moon, D. van Dijk, Z. Wang *et al.*, “Visualizing structure and transitions in high-dimensional biological data,” *Nat. Biotechnol.*, vol. 37, no. 12, pp. 1482–1492, Dec 2019. [Online]. Available: https://doi.org/10.1038/s41587-019-0336-3

[18] R. R. Coifman and S. Lafon, “Diffusion maps,” *Appl. Comput. Harmon. Anal.*, vol. 21, no. 1, pp. 5–30, 2006, special Issue: Diffusion Maps and Wavelets. [Online]. Available: https://doi.org/10.1016/j.acha.2006.04.006

[19] J. S. Rhodes, A. Cutler, G. Wolf *et al.*, “Random forest-based diffusion information geometry for supervised visualization and data exploration,” *2021 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 331–335, 2021. [Online]. Available: https://doi.org/10.1109/SSP49050.2021.9513749

[20] L. Benali, G. Notton, A. Fouilloy *et al.*, “Solar radiation forecasting using artificial neural network and random forest methods: Application to normal beam, horizontal diffuse and global components,” *Renewable Energy*, vol. 132, pp. 871–884, 2019. [Online]. Available: doi={https://doi.org/10.1016/j.renene.2018.08.044}

[21] T. Han, D. Jiang, Q. Zhao *et al.*, “Comparison of random forest, artificial neural networks and support vector machine for intelligent diagnosis of rotating machinery,” *Trans. Inst. Meas. Control*, vol. 40, pp. 2681 – 2693, 2018. [Online]. Available: https://doi.org/10.1177/0142331217708242

[22] M. Ali Khan, M. Izhar Shah, M. Faisal Javed *et al.*, “Application of random forest for modelling of surface water salinity,” *Ain Shams Engineering Journal*, vol. 13, no. 4, p. 101635, 2022. [Online]. Available: https://doi.org/10.1016/j.asej.2021.11.004

[23] W. Zhang, C. Wu, H. Zhong *et al.*, "Prediction of undrained shear strength using extreme gradient boosting and random forest based on Bayesian optimization," *Geosci. Front.*, vol. 12, no. 1, pp. 469–477, 2021. [Online]. Available: https://doi.org/10.1016/j.gsf.2020.03.007

[24] J. Lin, S. Lu, X. He *et al.*, "Analyzing the impact of three-dimensional building structure on CO_2 emissions based on random forest regression," *Energy*, vol. 236, p. 121502, 2021. [Online]. Available: https://doi.org/10.1016/j.energy.2021.121502

[25] F. Wang, Y. Wang, K. Zhang *et al.*, "Spatial heterogeneity modeling of water quality based on random forest regression and model interpretation," *Environ. Res.*, vol. 202, p. 111660, 2021. [Online]. Available: https://doi.org/10.1016/j.envres.2021.111660

[26] C. Iwendi, A. K. Bashir, A. Peshkar *et al.*, "COVID-19 patient health prediction using boosted random forest algorithm," *Front. Public Health*, vol. 8, p. 357, 2020. [Online]. Available: https://doi.org/10.3389/fpubh.2020.00357

[27] C. M. Yeşilkanat, "Spatio-temporal estimation of the daily cases of COVID-19 in worldwide using random forest machine learning algorithm," *Chaos, Solitons Fractals*, vol. 140, p. 110210, 2020. [Online]. Available: https://doi.org/10.1016/j.chaos.2020.110210

[28] X. Fang, W. Liu, J. Ai *et al.*, "Forecasting incidence of infectious diarrhea using random forest in Jiangsu province, China," *BMC Infect. Dis.*, vol. 20, no. 1, p. 222, Mar 2020. [Online]. Available: https://doi.org/10.1186/s12879-020-4930-2

[29] V. Nhu, A. Shirzadi, H. Shahabi *et al.*, "Shallow landslide susceptibility mapping by random forest base classifier and its ensembles in a semi-arid region of Iran," *Forests*, vol. 11, no. 4, 2020. [Online]. Available: https://doi.org/10.3390/f11040421

[30] L. Yang, H. Wu, X. Jin *et al.*, "Study of cardiovascular disease prediction model based on random forest in eastern China," *Sci. Rep.*, vol. 10, no. 1, p. 5245, Mar 2020. [Online]. Available: https://doi.org/10.1038/s41598-020-62133-5

[31] S. Saha, M. Saha, K. Mukherjee *et al.*, "Predicting the deforestation probability using the binary logistic regression, random forest, ensemble rotational forest, REPTree: A case study at the Gumani River Basin, India," *Sci. Total Environ.*, vol. 730, p. 139197, 2020. [Online]. Available: https://doi.org/10.1016/j.scitotenv.2020.139197

[32] M. Gholizadeh, M. Jamei, I. Ahmadianfar *et al.*, "Prediction of nanofluids viscosity using random forest (RF) approach," *Chemom. Intell. Lab. Syst.*, vol. 201, p. 104010, 2020. [Online]. Available: https://doi.org/10.1016/j.chemolab.2020.104010

[33] C. Rao, M. Liu, M. Goh *et al.*, "2-stage modified random forest model for credit risk assessment of p2p network lending to "three rurals" borrowers," *Appl. Soft Comput.*, vol. 95, p. 106570, 2020. [Online]. Available: https://doi.org/10.1016/j.asoc.2020.106570

[34] S. B. ud din Tahir, A. Jalal, and M. Batool, "Wearable sensors for activity analysis using SMO-based random forest over smart home and sports datasets," *2020 3rd*

International Conference on Advancements in Computational Sciences (ICACS), pp. 1–6, 2020. [Online]. Available: http://doi.org/10.1109/ICACS47775.2020.9055944

[35] K. Tan, H. Wang, L. Chen *et al.*, "Estimation of the spatial distribution of heavy metal in agricultural soils using airborne hyperspectral imaging and random forest," *J. Hazard. Mater.*, vol. 382, p. 120987, 2020. [Online]. Available: https://doi.org/10.1016/j.jhazmat.2019.120987

[36] Z. Lv, J. Zhang, H. Ding *et al.*, "RF-PseU: A random forest predictor for rna pseudouridine sites," *Front. Bioeng. Biotechnol.*, vol. 8, pp. 134–134, Feb 2020. [Online]. Available: https://doi.org/10.3389/fbioe.2020.00134

[37] W. Zhang, J. S. Rhodes, A. Garg *et al.*, "Label-free discrimination and quantitative analysis of oxidative stress induced cytotoxicity and potential protection of antioxidants using raman micro-spectroscopy and machine learning," *Anal. Chim. Acta*, vol. 1128, pp. 221–230, 2020. [Online]. Available: https://doi.org/10.1016/j.aca.2020.06.074

[38] L. Breiman, J. H. Friedman, R. A. Olshen *et al.*, *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks, 1984. [Online]. Available: https://doi.org/10.1201/9781315139470

[39] L. Buitinck, G. Louppe, M. Blondel *et al.*, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[40] T. Hastie, R. Tibshirani, and J. Friedman, "Random forests," in *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY: Springer New York, 2009, pp. 587–604. [Online]. Available: https://doi.org/10.1007/978-0-387-84858-7_15

[41] K. Chang, C. J. Creighton, C. Davis *et al.*, "The cancer genome atlas pan-cancer analysis project," *Nat. Genet.*, vol. 45, no. 10, pp. 1113–1120, Oct 2013. [Online]. Available: https://doi.org/10.1038/ng.2764

[42] R. Gorman and T. J. Sejnowski, "Analysis of hidden units in a layered network trained to classify sonar targets," *Neural Netw*, vol. 1, no. 1, pp. 75–89, 1988. [Online]. Available: https://doi.org/10.1016/0893-6080(88)90023-8

[43] Y. Lin and Y. Jeon, "Random forests and adaptive nearest neighbors," *J. Am. Stat. Assoc.*, vol. 101, no. 474, pp. 578–590, 2006. [Online]. Available: https://doi.org/10.1198/016214505000001230

[44] D. Feng and R. Baumgartner, "Random forest (rf) kernel for regression, classification and survival," *ArXiv*, vol. abs/2009.00089, 2020. [Online]. Available: https://arxiv.org/abs/2009.00089

[45] C. Englund and A. Verikas, "A novel approach to estimate proximity in a random forest: An exploratory study," *Expert Syst. Appl.*, vol. 39, no. 17, p. 13046–13050, Dec. 2012. [Online]. Available: http://doi.org/10.1016/j.eswa.2012.05.094

[46] H. Cao, S. Bernard, R. Sabourin *et al.*, "A novel random forest dissimilarity measure for multi-view learning," *2020 25th Int. Conf. Pattern Recognit. (ICPR)*, pp. 1344–1351, 2021. [Online]. Available: http://doi.org/DOI:10.1109/ICPR48806.2021.9412961

[47] L. Breiman and A. Cutler, "Random forests," https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#prox, (Accessed on 04/15/2020).

[48] M. N. Wright and A. Ziegler, "ranger: A fast implementation of random forests for high dimensional data in C++ and R," *J. Stat. Softw.*, vol. 77, no. 1, pp. 1–17, 2017.

[49] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2019.

[50] A. Davies and Z. Ghahramani, "The random forest kernel and other kernels for big data from random partitions," 2014. [Online]. Available: https://arxiv.org/abs/1402.4293

[51] D. Dua and C. Graff, "UCI machine learning repository," 2017, (Accessed on 05/21/2020). [Online]. Available: http://archive.ics.uci.edu/ml

[52] E. J. Finehout, Z. Franck, L. H. Choe *et al.*, "Cerebrospinal fluid proteomic biomarkers for Alzheimer's disease," *Ann. Neurol.*, vol. 61, no. 2, pp. 120–129, Feb. 2007. [Online]. Available: https://doi.org/10.1002/ana.21038

[53] H. Pang, A. Lin, M. Holford *et al.*, "Pathway analysis using random forests classification and regression," *Bioinformatics*, vol. 22, no. 16, pp. 2028–2036, 06 2006. [Online]. Available: http://doi.org/10.1093/bioinformatics/btl344

[54] T. Shi and S. Horvath, "Unsupervised learning with random forest predictors," *J. Comput. Graphical Stat.*, vol. 15, no. 1, pp. 118–138, 2006. [Online]. Available: https://doi.org/10.1198/106186006X94072

[55] M. B. Pouyan, J. Birjandtalab, and M. Nourani, "Metric learning using random forest for cytometry data," *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2590–2590, 2016. [Online]. Available: http://doi.org/10.1109/EMBC.2016.7591260

[56] D. R. Vangumalli, K. Nikolopoulos, and K. Litsiou, "Clustering, forecasting and cluster forecasting: using k-medoids, k-NNs and random forests for cluster selection," Bangor Business School, Prifysgol Bangor University (Cymru / Wales), Working Papers 19016, 2019. [Online]. Available: https://doi.org/10.1080/01969722.2021.1902049

[57] L. Alhusain and A. M. Hafez, "Cluster ensemble based on random forests for genetic data," *BioData Min.*, vol. 10, no. 1, p. 37, Dec 2017. [Online]. Available: https://doi.org/10.1186/s13040-017-0156-2

[58] P. A. A. Resende and A. C. Drummond, "A survey of random forest based methods for intrusion detection systems," *ACM Comput. Surv.*, vol. 51, no. 3, may 2018. [Online]. Available: https://doi.org/10.1145/3178582

[59] N. Nesa, T. Ghosh, and I. Banerjee, "Outlier detection in sensed data using statistical learning models for IoT," *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2018. [Online]. Available: http://doi.org/10.1109/WCNC.2018.8376988

[60] V. Narayana, A. Gopi, S. Khadherbhi *et al.*, "Accurate identification and detection of outliers in networks using group random forest methodoly," *J. Crit. Rev.*, vol. 7, no. 6, pp. 381–384, 2020. [Online]. Available: http://doi.org/10.31838/jcr.07.06.67

[61] F. B. de Santana, W. Borges Neto, and R. J. Poppi, "Random forest as one-class classifier and infrared spectroscopy for food adulteration detection," *Food Chem.*, vol. 293, pp. 323–332, 2019. [Online]. Available: https://doi.org/10.1016/j.foodchem.2019.04.073

[62] C. Liu, M. White, and G. Newell, "Detecting outliers in species distribution data," *J. Biogeogr.*, vol. 45, no. 1, pp. 164–176, 2018. [Online]. Available: https://doi.org/10.1111/jbi.13122

[63] D. Baron and D. Poznanski, "The weirdest SDSS galaxies: results from an outlier detection algorithm," *Mon. Not. R. Astron. Soc.*, vol. 465, no. 4, pp. 4530–4555, 11 2016. [Online]. Available: http://doi.org/10.1093/mnras/stw3021

[64] C. Aldrich and L. Auret, "Fault detection and diagnosis with random forest feature extraction and variable importance methods," *IFAC Proceedings Volumes*, vol. 43, no. 9, pp. 79–86, 2010, 13th IFAC Symposium on Automation in Mining, Mineral and Metal Processing. [Online]. Available: https://doi.org/10.3182/20100802-3-ZA-2014.00020

[65] M. Kokla, J. Virtanen, M. Kolehmainen *et al.*, "Random forest-based imputation outperforms other methods for imputing LC-MS metabolomics data: a comparative study," *BMC Bioinf.*, vol. 20, no. 1, p. 492, Oct 2019. [Online]. Available: http://doi.org/10.1186/s12859-019-3110-0

[66] A. Pantanowitz and T. Marwala, "Missing data imputation through the use of the random forest algorithm," in *Advances in Computational Intelligence*, W. Yu and E. N. Sanchez, Eds. Berlin, Heidelberg: Springer, 2009, pp. 53–62. [Online]. Available: http://doi.org/10.1007/978-3-642-03156-4_6

[67] S. van Buuren and K. Groothuis-Oudshoorn, "mice: Multivariate imputation by chained equations in R," *J. Stat. Softw.*, vol. 45, no. 3, pp. 1–67, 2011. [Online]. Available: http://doi.org/10.18637/jss.v045.i03

[68] A. D. Shah, J. W. Bartlett, J. Carpenter *et al.*, "Comparison of random forest and parametric imputation models for imputing missing data using MICE: a caliber study," *Am. J. Epidemiol.*, vol. 179, no. 6, pp. 764–774, Mar 2014. [Online]. Available: http://doi.org/10.1093/aje/kwt312

[69] T. Rockel, *missMethods: Methods for Missing Data*, 2020, r package version 0.2.0. [Online]. Available: https://CRAN.R-project.org/package=missMethods

[70] Q. Zhou, W. Hong, L. Luo *et al.*, "Gene selection using random forest and proximity differences criterion on dna microarray data," *J. Convergence Inf. Technol.*, vol. 5, pp. 161–170, 2010. [Online]. Available: http://doi.org/10.4156/JCIT.VOL5.ISSUE6.17

[71] L. S. Whitmore, A. George, and C. M. Hudson, "Explicating feature contribution using random forest proximity distances," 2018. [Online]. Available: https://arxiv.org/abs/1807.06572

[72] J. A. Seoane, I. N. M. Day, J. P. Casas *et al.*, "A random forest proximity matrix as a new measure for gene annotation," in *22th European Symposium on Artificial Neural Networks, ESANN 2014, Bruges, Belgium, April 23-25, 2014*, 2014.

[73] P. Zhao, X. Su, T. Ge *et al.*, "Propensity score and proximity matching using random forest," *Contemp. Clin. Trials*, vol. 47, pp. 85–92, Mar 2016. [Online]. Available: http://doi.org/10.1016/j.cct.2015.12.012

[74] M. S. Nielsen and V. Rohde, "A surrogate model for estimating extreme tower loads on wind turbines based on random forest proximities," *J. Appl. Stat.*, vol. 49, no. 2, pp. 485–497, 2022. [Online]. Available: https://doi.org/10.1080/02664763.2020.1815675

[75] H. Ishwaran and U. Kogalur, "Random survival forests for r," *R News*, vol. 7, no. 2, pp. 25–31, October 2007. [Online]. Available: https://CRAN.R-project.org/doc/Rnews/

[76] T. Hothorn, P. Buehlmann, S. Dudoit *et al.*, "Survival ensembles," *Biostatistics*, vol. 7, no. 3, pp. 355–373, 2006. [Online]. Available: https://doi.org/10.1093/biostatistics/kxj011

[77] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*. New York, NY: Springer, 2016. [Online]. Available: https://doi.org/10.1007/978-0-387-98141-3

[78] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901. [Online]. Available: https://doi.org/10.1080/14786440109462720

[79] D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, Oct 1999. [Online]. Available: https://doi.org/10.1038/44565

[80] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008. [Online]. Available: http://jmlr.org/papers/v9/vandermaaten08a.html

[81] G. E. Hinton and R. Zemel, "Autoencoders, minimum description length and helmholtz free energy," in *Advances in Neural Information Processing Systems*, J. Cowan, G. Tesauro, and J. Alspector, Eds., vol. 6. Morgan-Kaufmann, 1993. [Online]. Available: https://proceedings.neurips.cc/paper/1993/file/9e3cfc48eccf81a0d57663e129aef3cb-Paper.pdf

[82] J. Cheng, H. Liu, F. Wang *et al.*, "Silhouette analysis for human action recognition based on supervised temporal t-SNE and incremental learning," *IEEE Transactions on Image Processing*, vol. 24, no. 10, pp. 3203–3217, 2015. [Online]. Available: http://doi.org/10.1109/TIP.2015.2441634

[83] L. Hajderanj, I. Weheliye, and D. Chen, "A new supervised T-SNE with dissimilarity measure for effective data visualization and classification," in *Proceedings of the 2019 8th International Conference on Software and Information Engineering*, ser. ICSIE '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 232–236. [Online]. Available: https://doi.org/10.1145/3328833.3328853

[84] T. Sainburg, L. McInnes, and T. Q. Gentner, "Parametric UMAP embeddings for representation and semisupervised learning," *Neural Computation*, vol. 33, no. 11, pp. 2881–2907, 10 2021. [Online]. Available: https://doi.org/10.1162/neco_a_01434

[85] C.-G. Li and J. Guo, "Supervised isomap with explicit mapping," *First International Conference on Innovative Computing, Information and Control - Volume I (ICICIC'06)*, vol. 3, pp. 345–348, 2006. [Online]. Available: http://doi.org/10.1109/ICICIC.2006.530

[86] B. Nadler, S. Lafon, R. R. Coifman *et al.*, "Diffusion maps, spectral clustering and reaction coordinates of dynamical systems," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 113–127, 2006, special Issue: Diffusion Maps and Wavelets. [Online]. Available: https://doi.org/10.1016/j.acha.2005.07.004

[87] L. Page, S. Brin, R. Motwani *et al.*, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Technical Report 1999-66, November 1999, previous number = SIDL-WP-1999-0120. [Online]. Available: http://ilpubs.stanford.edu:8090/422/

[88] G. Chao, Y. Luo, and W. Ding, "Recent advances in supervised dimension reduction: A survey," *Mach. Learn. Knowl. Extr.*, vol. 1, no. 1, p. 341–358, Jan 2019. [Online]. Available: http://dx.doi.org/10.3390/make1010020

[89] E. Bair, T. Hastie, D. Paul *et al.*, "Prediction by supervised principal components," *J. Am. Stat. Assoc.*, vol. 101, no. 473, pp. 119–137, 2006. [Online]. Available: https://doi.org/10.1198/016214505000000628

[90] S. Yu, K. Yu, V. Tresp *et al.*, "Supervised probabilistic principal component analysis," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, ser. KDD '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 464–473. [Online]. Available: https://doi.org/10.1145/1150402.1150454

[91] E. Barshan, A. Ghodsi, Z. Azimifar *et al.*, "Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds," *Pattern Recognit.*, vol. 44, no. 7, pp. 1357–1371, 2011. [Online]. Available: https://doi.org/10.1016/j.patcog.2010.12.015

[92] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936. [Online]. Available: https://doi.org/10.1111/j.1469-1809.1936.tb02137.x

[93] M. Amouzgar, D. R. Glass, R. Baskar *et al.*, “Supervised dimensionality reduction for exploration of single-cell data by hybrid subset selection - linear discriminant analysis,” *bioRxiv*, 2022. [Online]. Available: https://www.biorxiv.org/content/early/2022/01/06/2022.01.06.475279

[94] S. Wold, M. Sjöström, and L. Eriksson, “Pls-regression: a basic tool of chemometrics,” *Chemometrics and Intelligent Laboratory Systems*, vol. 58, no. 2, pp. 109–130, 2001, pLS Methods. [Online]. Available: https://doi.org/10.1016/S0169-7439(01)00155-1

[95] S. Tsuge, M. Shishibori, S. Kuroiwa *et al.*, “Dimensionality reduction using non-negative matrix factorization for information retrieval,” *2001 IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace (Cat.No.01CH37236)*, vol. 2, pp. 960–965 vol.2, 2001. [Online]. Available: http://doi.org/10.1109/ICSMC.2001.973042

[96] H. Liu, W. Zhaohui, L. Xuelong *et al.*, “Constrained nonnegative matrix factorization for image representation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1299–1311, 2012. [Online]. Available: https://doi.org/10.1007/s11277-018-5325-1

[97] Z. Li, J. Tang, and X. He, “Robust structured nonnegative matrix factorization for image representation,” *IEEE Trans. Neural Netw. Learn. Syst*, vol. 29, no. 5, pp. 1947–1960, 2018. [Online]. Available: http://doi.org/10.1109/TNNLS.2017.2691725

[98] X. Zhang, L. Zong, X. Liu *et al.*, “Constrained clustering with nonnegative matrix factorization,” *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 27, no. 7, pp. 1514–1526, 2016.

[99] W. Liu, K. Yuan, and D. Ye, “Reducing microarray data via nonnegative matrix factorization for visualization and clustering analysis,” *J. Biomed. Inform.*, vol. 41, no. 4, pp. 602 – 606, 2008. [Online]. Available: https://doi.org/10.1016/j.jbi.2007.12.003

[100] J. Tapson and J. Greene, “Plant data visualization using non-negative matrix factorization,” *IFAC*, vol. 38, no. 1, pp. 73 – 78, 2005. [Online]. Available: https://doi.org/10.3182/20050703-6-CZ-1902.01814

[101] M. Babaee, S. Tsoukalas, G. Rigoll *et al.*, “Immersive visualization of visual data using nonnegative matrix factorization,” *Neurocomputing*, vol. 173, pp. 245–255, 2016. [Online]. Available: https://doi.org/10.1016/j.neucom.2015.03.121

[102] D. Cai, X. He, X. Wu *et al.*, “Non-negative matrix factorization on manifold,” *2008 Eighth IEEE International Conference on Data Mining*, pp. 63–72, 2008. [Online]. Available: https://doi.org/10.1007/s11063-019-10111-y

[103] D. Kobak and G. C. Linderman, “Initialization is critical for preserving global data structure in both t-SNE and UMAP,” *Nature Biotechnology*, vol. 39, no. 2, pp. 156–157, Feb 2021. [Online]. Available: https://doi.org/10.1038/s41587-020-00809-z

[104] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.290.5500.2323

[105] Q. Jiang and M. Jia, "Supervised laplacian eigenmaps for machinery fault classification," *2009 WRI World Congress on Computer Science and Information Engineering*, vol. 7, pp. 116–120, 2009. [Online]. Available: http://doi.org/10.1109/CSIE.2009.765

[106] M. Vlachos, C. Domeniconi, D. Gunopulos *et al.*, "Non-linear dimensionality reduction techniques for classification and visualization," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '02. New York, NY, USA: Association for Computing Machinery, 2002, p. 645–651. [Online]. Available: https://doi.org/10.1145/775047.775143

[107] D. de Ridder, O. Kouropteva, O. Okun *et al.*, "Supervised locally linear embedding," in *Artificial Neural Networks and Neural Information Processing — ICANN/ICONIP 2003*, O. Kaynak, E. Alpaydin, E. Oja *et al.*, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 333–341.

[108] L. Hajderanj, D. Chen, and I. Weheliye, "The impact of supervised manifold learning on structure preserving and classification error: A theoretical study," *IEEE Access*, vol. 9, pp. 43 909–43 922, 2021. [Online]. Available: 10.1109/ACCESS.2021.3066259

[109] M.-F. Balcan, A. Blum, and N. Srebro, "A theory of learning with similarity functions," *Mach. Learn.*, vol. 72, no. 1, pp. 89–112, Aug 2008. [Online]. Available: https://doi.org/10.1007/s10994-008-5059-5

[110] L. Haghverdi, M. Buettner, F. A. Wolf *et al.*, "Diffusion pseudotime robustly reconstructs lineage branching," *Nature Methods*, vol. 13, no. 10, p. 845, 2016. [Online]. Available: http://doi.org/10.1038/nmeth.3971

[111] A. F. Duque, G. Wolf, and K. R. Moon, "Visualizing high dimensional dynamical processes," in *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing*, Pittsburgh, PA, 2019, pp. 1–6. [Online]. Available: http://doi.org/10.1109/MLSP.2019.8918875

[112] M. Bohanec and V. Rajkovič, "V.: Knowledge acquisition and explanation for multi-attribute decision," in *Making, 8 th International Workshop "Expert Systems and Their Applications*, 1988.

[113] E. Anderson, "The species problem in iris," *Annals of the Missouri Botanical Garden*, vol. 23, no. 3, pp. 457–509, 1936. [Online]. Available: https://doi.org/10.2307/2394164

[114] N. Mantel, "The detection of disease clustering and a generalized regression approach," *Cancer Research*, vol. 27, no. 2_Part_1, pp. 209–220, 02 1967.

[115] A. F. Duque, S. Morin, G. Wolf *et al.*, "Extendable and invertible manifold learning with geometry regularized autoencoders," *2020 IEEE International Conference on Big Data (Big Data)*, pp. 5027–5036, 2020. [Online]. Available: http://doi.org/10.1109/BigData50022.2020.9378049

[116] N. V. Chawla, K. W. Bowyer, L. O. Hall *et al.*, “SMOTE: Synthetic minority over-sampling technique,” *J. Artif. Int. Res.*, vol. 16, no. 1, p. 321–357, jun 2002.

[117] D. Ramachandram and G. W. Taylor, “Deep multimodal learning: A survey on recent advances and trends,” *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 96–108, 2017. [Online]. Available: http://doi.org/10.1109/MSP.2017.2738401

Jake Rhodes

801-645-0774 | jake.rhodes@usu.edu | https://www.linkedin.com/in/jake-rhodes-a7a9a685/

Education

Utah State University, GPA: 4.0 Logan, UT
PhD, Statistics *Aug. 2018 – Aug 2022 (Expected)*

Weber State University, GPA: 4.0 Ogden, UT
Bachelor's of Science, Applied Mathematics *Aug. 2015 – May 2017*

Utah State University, GPA: 3.94 Logan, UT
Bachelor's of Arts, Finance and Economics *Aug. 2012 – May 2015*

Experience

Assistant Professor Aug. 2022 – Present
Idaho State University *Pocatello, ID*

- Explored effects of image sizes and padding on convolutional neural network accuracy
- Examined behavior of BiT models on distribution shifts
- Determined that convolutional neural networks were learning based on color and image size of remote sensing images via meta-data learning
- Initialized study of the utility of capsule networks in the remote-sensing domain

National Security Intern (NSIP) Jun. 2021 – Jul. 2022
Pacific Northwest National Laboratory *Remote*

- Explored effects of image sizes and padding on convolutional neural network accuracy
- Examined behavior of BiT models on distribution shifts
- Determined that convolutional neural networks were learning based on color and image size of remote sensing images via meta-data learning
- Initialized study of the utility of capsule networks in the remote-sensing domain

Graduate Research Assistant Jun. 2019 – Jul. 2022
Utah State University *Logan, UT*

- Developed methods for supervised dimensionality reduction using random-forest kernels and diffusion
- Applied novel unsupervised machine learning methods on lung cell Raman spectroscopy
- Classified cell types and treatments using convolutional neural networks on Raman images
- Aided in the development and initial experimentation with novel random forests proximity measures
- Explored distributional divergence of seizure onset zones of epileptic patients

Graduate Teaching Assistant Aug. 2018 – Jul. 2022
Utah State University *Logan, UT*

- Taught sections of trigonometry; wrote and administered learning assessments and lessons
- Aided student learning for courses in real analysis, linear regression, experimental design, and business statistics
- Incorporated programming for visual aids to induce logical thinking

Adjunct Mathematics Instructor Aug. 2017 – Jul. 2022
Weber State University *Logan, UT*

- Incorporated higher-level learning to promote the development of mathematical maturity
- Designed lesson plans for courses in college algebra, trigonometry, statistics, and contemporary mathematics
- Introduced new mathematical concepts which extended existing student knowledge
- Fostered student engagement by building a classroom community and incorporating Just-in-Time Teaching

Financial Analyst Aug. 2015 – Aug. 2018
The Department of Technology Services, State of Utah *Salt Lake City, UT*

- Used data visualization to optimize decision making in accordance with company goals
- Created over 80 data-based reports for use of the team and management
- Prepared visuals and statistics for upper management and the entire division of finance group
- Trained staff on general statistical methods and specific reporting tools (ServiceNow)

Financial Audit Intern May 2015 - Aug. 2015
The Walt Disney Company *Glendale, California*

- Teamed with audit seniors and managers to improve annual planning, fieldwork, documentation, and reporting tasks
- Performed tests on support documentation to uncover discrepancies and determine outliers
- Validated document accuracy to minimize time spent correcting mistakes
- Created concise, standardized templates for addressing common audit issues for efficient documentation

Start-up Consulting Intern Jan. 2014 - April 2014
Wasatch Social Ventures *Trujillo, Peru*

- Taught classes in finance, accounting, marketing, and other business topics in the students' native language (Spanish)
- Assisted students in developing business plans; 20% of business plans were approved for loans
- Performed due diligence and follow-up on startups
- Recorded and monitored startups' financial statements

Publications

Refereed Journal Articles

- **J.S. Rhodes**, A. Cutler, K.R. Moon, ”Geometry- and Accuracy-Preserving Random Forest Proximities”, under preparation for submission to *IEEE Transactions on Pattern Analysis and Machine Learning.*
- W. Zhang, **J.S. Rhodes**, K.R. Moon , B.S. Knudsen, L. Nokolova, A. Zhou, “Imaging of PD-L1 in single cancer cells by SERS-based hyperspectral analysis,” accepted in *Biomedical Optics Express*, Sept. 2020.
- W. Zhang*, **J. Rhodes***, A. Garg, J. Takemoto, X. Qi, S. Harihar, C.T. Chang, K.R. Moon**, A. Zhou**, ”Label-free discrimination and quantitative analysis of oxidative stress induced cytotoxicity and potential protection of antioxidants using Raman micro-spectroscopy and machine learning,” accepted in *Analytica Chimica Acta, vol. 1128, pp. 221-230*, Sept. 2020.

Refereed Conference Papers

- **J.S. Rhodes**, A. Cutler, G. Wolf, K.R. Moon, ”Random Forest-based Diffusion Information Geometry for Supervised Visualization and Data Exploration,” IEEE Statistical Signal Processing Workshop (SSP), July 2021.

* = These authors contributed equally
** = These authors contributed equally

Presentations

- ”Random Forest-Based Diffusion Information Geometry for Supervised Visualization and Data Exploration, *JSM 2022* August 8, 2022, Washington D.C.
- ”Resizing Effects on Convolutional Neural Network Accuracy and Robustness”, *PNNL Student Research Symposium*, August 19, 2021, (Virtual)
- ”Random Forest-based Diffusion Information Geometry for Supervised Visualization and Data Exploration,” *IEEE Statistical Signal Processing Workshop*, July 13, 2021, Rio de Janerio (Virtual)
- ”Supervised Visualization for Data Exploration”, *Utah State Student Research Symposium*, April 15, 2021, Logan, UT

Teaching

Courses Taught [# of Semesters]

- **MATH 5210** Elementary Real Analysis (Assistant) [1]
- **STAT 5200** Analysis of Designed Experiments (Assistant) [3]
- **STAT 5100** Modern Regression (Assistant) [1]
- **MATH 1210** Calculus II (Assistant) [1]
- **MATH 1210** Calculus I [1]
- **MATH 1100** Calculus Techniques (Assistant) [1]
- **MATH 1060** Trigonometry [3]
- **MATH 1050** College Algebra [10]
- **MATH 1040** Intro to Statistics [2]
- **MATH 1030** Contemporary Mathematics [2]

Awards

- Excellence in Research Award, Utah State University (2021)
- Teaching in Excellence Award, Utah State University (2019)
- Valedictorian, College of Science, Weber State University (2017)
- Outstanding Math Graduate of the Year, Weber State University (2017)
- College of Science Private Foundation Scholarship, (2015 - 2016)
- Joshua Eisenstat Memorial Scholarship, (2015 - 2016)
- Hansen Scholarship Recipient, (2015)
- Jon M. Huntsman Scholar (2013 - 2015)
- Presidential Transfer Scholarship (Awarded to top 5% of USU transfer students (2012 - 2015))

Memberships

- American Mathematical Society
- American Statistical Association
- IEEE

Service / Leadership

- Assisted the reviewing of papers for *NeurIPS*, *iEEE SSP*, *ICML*, *AISTATS* (2019 - 2021)
- Jon M. Huntsman Scholar Mentor (2014 - 2015)
- President Society for International Business and Economic Development, Utah State University, (2014 - 2015)
- Volunteer at the Senior Missionary Training Center, teaching Croatian / Serbian (2011 - 2013)
- Eagle Scout Award