Ensemble Tree-Based Machine Learning for Imaging Data

Reza Iranzad
University of Arkansas, Fayetteville

Ensemble Tree-Based Machine Learning for Imaging Data

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Engineering with a concentration in Industrial Engineering

by

Reza Iranzad
Sharif University of Technology
Bachelor of Science in Industrial Engineering, 2015
Sharif University of Technology
Master of Science, Industrial Engineering, 2017

August 2022
University of Arkansas

This dissertation is approved for recommendation to the Graduate Council.

Abstract

In particular medical imaging data, such as positron emission tomography (PET), computed tomography (CT), and fluorescence intravital microscopy (IVM), have become prevalent for use in a wide variety of applications, from diagnostic purposes, tracking diseases' progress, and monitoring the effectiveness of treatments to decision-making processes. The detailed information generated by medical imaging has enabled physicians to provide more comprehensive care. Although numerous machine learning algorithms, especially those used for imaging data, have been developed, dealing with unique structures in imaging data remained a big challenge. In this dissertation, we are proposing novel statistical tree-based methods with more efficient and more accurate responses for use in medical imaging applications.

In Chapter 2, we introduce a gradient Boosted Trees for Spatial Data (Boost-S) with covariate information. The main innovation of this chapter is to incorporate the spatial correlation structure into the boosting structure. Boosting trees are one of the most successful statistical learning approaches that involve sequentially growing an ensemble of simple regression trees. However, gradient boosted trees are not yet available for spatially correlated data. Boost-S integrates the spatial correlation structure into the classical framework of gradient boosted trees. Each tree is constructed by solving a regularized optimization problem, where the objective function takes into account the underlying spatial correlation and involves two penalty terms on tree complexity. A computationally-efficient greedy heuristic algorithm is proposed to obtain an ensemble of trees. The proposed Boost-S is applied to the spatially-correlated FDG-PET (fluorodeoxyglucose-positron emission tomography) imaging data collected from clinical trials of cancer chemoradiotherapy. Quantitatively assessing and monitoring tumor response to therapy is essential for an optimized treatment plan. Hence, accurately predicting the change of SUV (standardized uptake value) is critical for treatment optimization and control. Our numerical investigations successfully demonstrate the advantages of the proposed Boost-S over existing approaches for this particular application.

In Chapter 3, we propose a Structured Adaptive Boosting Trees algorithm (AdaBoost.S) for the edge detection problem associated with medical images. The main innovation of this chapter is to develop structural learning in an additive boosting model. The algorithm is motivated by the well-known observation that edges over an image mask often exhibit special structures and are highly interdependent. Such structures can be predicted using the features extracted from a bigger image patch that covers the image mask. We present the details of feature extraction and the technical details of constructing structured boosting trees leveraging the classical framework of adaptive boosting. The proposed AdaBoost.S is applied to detect the platelet-neutrophil aggregates from a large number of fluorescence IVM images of the pulmonary microcirculation. The platelet-neutrophil aggregates are important to assess lung injury from e-cigarette exposure. Therefore, a statistical learning algorithm is required to efficiently and accurately detect the edges of platelet-neutrophil aggregates. The predictive capabilities of the proposed approach are demonstrated by comparing the F-score, precision, and recall with those of other methods.

In Chapter 4, we review a variety of feature selection (FS) techniques that are built around random forests (wrappers based on RF) since high dimensional datasets recently have become overwhelmingly generated in different fields, especially in gene selection studies. The main goal of using such techniques is to identify and eliminate features with less or no predictive power in order to improve the predictive accuracy by removing unimportant or non-informative features, enhance the interpretability of a former complex data structure, and significantly reduce the computational complexity of the predictor. Our review includes Boruta, RRF, GRRF, GRF, r2VIM, PIMP (Altmann), NTA (vita), varSelRF, VSURF, RF-SRC, AUCRF, and RFE methods. Also, the mentioned methods are applied to three publicly available datasets.

Keywords: Statistical Learning; Ensemble Learning; Boosting Trees; Spatial Statistics; Medical Imaging; Feature Selection; Random Forests; Gradient Boosted Trees.

Acknowledgements

I would like to give a very special thanks to my advisor and mentor, Dr. Xiao Liu, whose endless guidance and support have taught me creativity, critical thinking, and being a researcher. I am extremely grateful to him for making this long journey pleasant and I'd always thought about how lucky I am to end up working with him. I would also like to thank my dissertation committee, Prof. Ed Pohl, Prof. W. Art Chaovalitwongse, and Dr. Margaret Bennewitz, whose significant instructions and feedback have given rise to the quality of my education and research work.

Dedication

To Maryam, Baba, and Armin

I would like to dedicate this dissertation to my love, my dear father, and my beloved brother, who have always been beside me, supported me and believed in me. There is no doubt that none of my success (if I can say so) would have been done without you.

Contents

List of Figures

List of Tables

1 Introduction

With the rapid development of technology in the past few decades, a wide variety of equipment has been emerged to recreate medical images. Medical imaging refers to a set of techniques and processes in which various images of tissues, internal organs, and limbs can be generated by clinicians. Evolving the medical imaging technologies has opened a new chapter in diagnostic purposes, monitoring diseases progress, and treatment planning. Medical imaging complies with non-invasive procedures, enabling physicians to observe living cells in the actual environment and their behavior over time while putting patients at the lowest risks. This efficient, fast, and reliable resource generates large volumes of two-dimensional (2D) and three-dimensional (3D) data. From a wide range of applications and techniques that medical imaging provides, we focus on 1) positron emission tomography (PET) to detect metabolically active malignant lesions and to quantitatively assess tumor response to treatment and 2) fluorescence intravital microscopy (IVM) images of the pulmonary microcirculation to locate and detect the edges of aggregates within blood vessels due to e-cigarette vapor exposure. PET imaging focuses on the organ level at lower resolution and is noninvasive; it provides functional information about the metabolism of the tumor. Intravital imaging focuses on the cellular level at much higher resolution ($\sim$ 250 nm) but is invasive and usually requires cutting into the tissue; intravital imaging can also provide functional information about cellular movement in blood vessels or tissues over time. Clinicians are starting to use optical imaging, but it is only for surface applications such as skin imaging or with cameras to image the gastrointestinal tract. In this dissertation, we propose statistical algorithms to analyze phenomena at different scales (organ level vs. cellular level).

Machine learning (ML) is a technique that by using data, enables systems to mimic human learning skills and predict outcomes more accurately. With the large amount of data generated in clinical trials, machine learning plays a major role in the healthcare sector's developments. Although research in computer analysis of healthcare applications brings more promises to improve patients' comprehensive care, a lot more research yet needs to be done to improve diagnosis and prediction accuracy. Among the machine learning methods, ensemble models have become one of the most successful statistical learning approaches. Ensemble models are supervised learning techniques that combine numerous base models (which are also referred to as "weak learners") to render better predictive performance than any individual weak learners could possibly perform. Even though several sophisticated ensemble models have been developed, we found some important incapability of them which we will address in this dissertation with our two novel boosting trees models. Specifically, the first major contribution within this thesis is proposing a gradient Boosted Trees algorithm for Spatial Data (Boost-S) with covariate information where Boost-S integrates the spatial correlation into the classical framework of eXtreme Gradient Boosting. Our second research contribution is the proposal of a Structured Adaptive Boosting Trees algorithm (AdaBoost.S) for the edge detection problem associated with medical images where the model predicts mask structures using the features extracted from a bigger image patch that covers the image mask.

In Chapter 2, we introduce a novel Gradient Boosting Trees algorithm for Spatial Data (Boost-S) with covariate information to be applied to the spatially-correlated FDG-PET (fluorodeoxyglucose-positron emission tomography) imaging data collected from FLARE-RT clinical trial of cancer chemoradiotherapy. FDG-PET has been widely used in cancer diagnosis to detect metabolically active malignant lesions, and plays a critical role in quantitatively assessing and monitoring tumor response to treatment. **18**F-FDG PET is a radiotracer that contains a glucose analog that is labeled with F18. Tumors perform glycolysis to generate ATP (energy), which is much less efficient than oxidative phosphorylation. As tumors grow, they need more ATP, which leads to elevated glucose metabolism. The **18**F-FDG is taken up by tumor cells and is metabolized; the localization of the radiolabeled glucose analog in living tumor cells is what gives the PET signal. Tumor cells that are successfully killed will not take up the imaging agent and will not cause a hot spot on the image. The Standardized Uptake Values (SUV) is a ratio that measures cells activity in PET imaging. By comparing the SUV PET images at different time spots, before and in the middle of the treatment, it is possible to observe the shrinkage of the tumor in case of the effective treatment. The data used in this work are obtained from patients diagnosed with locally advanced and surgically unresectable non-small cell lung cancer (NSCLC) who enrolled onto the FLARE-RT clinical trial. A tumor is a 3D object and typically presents spatially-correlated and spatially-heterogeneous responses to radiotherapy. Some areas of a tumor may respond well to treatment while some areas may not respond to treatment because the treatment did not reach all areas of the tumor or because certain tumor cells can be resistant to treatment. One that is important to mention is that the overall SUV level decreases three weeks after the radiotherapy. Also, the change in SUV, i.e., tumor's response to radiotherapy, varies over space. Therefore, the capability of modeling and predicting the change of SUV is critical for treatment optimization. We aim to model the differences between the Pre-RT SUV (before treatment) and the Mid-RT SUV (during the third week of treatment course), which helps clinicians further optimize or control the treatment plans. However, the complex relationship between the change of SUV and available features can hardly be directly specified. The non-linear effects, as well as the interactions among features, motivate us to investigate the non-parametric tree-based models.

Spatial data refer to an important type of data which arise in a spatial area and are often correlated in space. One challenge arising from practice is to capture such correlation between features and responses, which can rarely be adequately specified by linear models. Thus, non-parametric approaches, especially the additive-tree-based models, provide some major modeling advantages. Constructing a tree does not need parametric assumptions on the complex relationship between features and responses. An individual tree performs a partition of the feature space. For each sub feature space (represented by a tree leaf), a predicted value is found for the individuals over that sub feature space. A sum-of-trees model consists of multivariate components that effectively handle the complex interaction effects among features (Chipman et al., 2010). Figure 1.1 represents the architecture of boosting trees. It has been shown that as more trees are added to the ensemble at each iteration, more variability could be explained; hence, the error is gradually decreased.

Among the additive-tree-based methods, gradient boosted trees have become one of the most successful statistical learning approaches. The main idea behind the gradient boosting is fitting

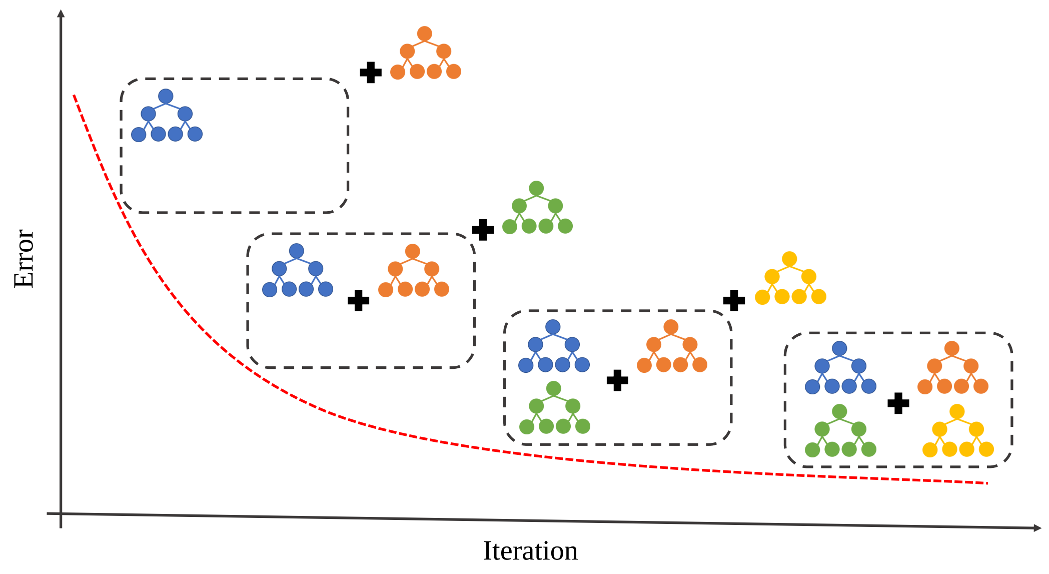


Figure 1.1: The architecture of Boosting Trees

a sequence of correlated "weaker learners" (such as simple trees) such that each tree attempts to explain only a small amount of variation not captured by previous trees.

Among the vast majority of existing tree-based methods, a key assumption is that observations are independent. Many existing boosting trees, such as XGBoost, do not consider the possible spatial correlation at all when they are applied to spatial data. This means that all residual correlation, i.e. correlation that is not accounted for by the regression function, is ignored. Modeling such correlation allows not only for better learning of the regression function, but it is also important for prediction, in particular for probabilistic prediction and for predicting averages of sums over space, time, and groups of clusters (Sigrist, 2020). Hence, the main contribution of this research is to propose a computationally-efficient gradient boosting method for growing the ensemble trees for spatially-correlated data.

In Chapter 3, we propose a Structured Adaptive Boosting Trees (AdaBoost.S) for the edge detection problem associated with medical imaging data. This research is part of a bigger investigation of the effects of e-cigarette vapor on vascular systems as well as finding possible invisible damages that may occur. Neutrophils are the most abundant type of white blood cells and can release a network of DNA fibers called neutrophil extracellular traps (NETs) as a host defense mechanism. NETs can cause downstream injury through enabling inflammation, vessel damage, and clot formation. Platelets, which are known to activate clotting, can adhere to neutrophils to form platelet-neutrophil aggregates that can block blood vessels and prevent gas exchange. Therefore, detection of platelet-neutrophil aggregates from the fluorescence lung IVM images is an essential step toward understanding the effects of e-cigarette vapor and other pathologies driven by platelet-neutrophil interactions. In IVM imaging, fluorescent dyes and fluorescent antibodies are injected into the blood vessels that will either illuminate the bloodstream or certain cells of interest in different colors to be visible on IVM. Our proposed method is applied to detect vascular lung injury on fluorescence IVM images of mice exposed to e-cigarette vapor. IVM allows one to track interactions between multiple cell types over time in several organs throughout the body including but not limited to the lungs, liver, and brain of living subjects.

Fluorescence IVM captures large datasets of dynamic multicellular interactions within various organs. In one imaging session, gigabytes of dynamic data are generated which presents a challenge to manual analysis of cell interactions. To meet this need, microscope manufacturers have designed analytical software to detect the individual labeled cells, but these platforms are met with limitations including high cost, requiring human input for each video, multicellular interactions artifacts, and lung movement artifacts. Herein, a statistical learning approach is developed to automate detection of multicellular aggregates of distinctly labeled blood cells within the microcirculation of live mice. By comparing fluorescence IVM images of the pulmonary microcirculation of healthy mice and mice exposed to e-cigarette vapor during the experimental trial, the formation of platelet-neutrophil aggregates (the total area of objects containing colocalized neutrophil and platelet staining) can be clearly observed from the mouse exposed to e-cigarette vapor, indicating the potential for decreased blood flow. The data used in this research are obtained from mice undergoing e-cigarette vapor exposure in the preclinical study conducted by the authors' team.

An edge in an image can be defined as an abrupt change of color intensity between two neighboring pixels (Ziou, Tabbone, et al., 1998). Edge detection is a process that detects the presence and location of edges formed by sharp changes in the image's color intensity (Ramana et al., 2017). A critical observation is that edges over an image mask often exhibit special structures and are highly interdependent and this notion pipes us into a family of novel methods called structured learning. The main point of structural learning is to learn the structures of a group of pixels, called masks, rather than ordinary computing if individual pixels contain edges. Because Random Forests and boosting trees are main competitors of each other, which are based on opposite ideas, this research fills a gap in the literature where the structured boosting trees are not yet available. Hence, the main contribution of this research is to propose a new Structured Adaptive Boosting Trees algorithm (AdaBoost.S) for edge detection problems.

In Chapter 4, we review and do the comparative study of random forest based feature selection for classification problems. Feature selection (FS) is a pre-processing step to achieve a more efficient dataset and to build a more accurate predictive model. The core idea behind FS methods is to identify and eliminate features with less or no predictive power. FS methods are generally classified into filters, wrappers, and embedded algorithms which we focus on wrappers built over random forest classification in the review. Random forests are commonly used machine learning models because of their high predictive accuracy and the calculating measures of variable importance. Hence, random forest is a strong candidate for building wrappers. Our review includes Boruta, RRF, GRRF, GRF, r2VIM, PIMP (Altmann), NTA (vita), varSelRF, VSURF, RF-SRC, AUCRF, and RFE. Also, we evaluate the mentioned FS methods on three publicly available datasets.

2 Gradient Boosted Trees for Spatial Data and its Application to Medical Imaging Data

2.1 Introduction

Fluorodeoxyglucose-Positron Emission Tomography (**18**F-FDG-PET) has been widely used in cancer diagnosis to detect metabolically active malignant lesions, and plays a critical role in quantitatively assessing and monitoring tumor response to treatment. For illustrative purposes, Figure 2.1 shows the Standardized Uptake Values (SUV) obtained from the FDG-PET imaging of a patient. The FDG-PET SUV distribution is overlaid as a hot metal false colorwash (range SUV 0-20) along with the grayscale CT image. The CT and PET images are used to help pin the exact location of the tumor as PET shows active malignant lesions and CT shows body structure. Radiation isodose lines are displayed on a rainbow false colorscale as percentages of the prescribed radiation therapy dose (98% - red, 90% - orange, 80% - yellow, 65% - green, 50% - cyan, 25% - blue). Different doses are needed since the density of malignant tumors varies spatially, also we want to minimize the side effects to surrounding normal tissue; hence, the highly concentrated areas in the center of the tumor need a higher dosage.

In Figure 2.1, the top row shows the baseline FDG-PET image taken before radiotherapy (i.e., Pre-RT), while the bottom row shows the image three weeks after initiation of radiotherapy (i.e., Mid-RT image). By comparing the top and bottom rows of this figure, it is possible to observe the shrinkage of the tumor, indicating the effectiveness of treatment. Hence, the difference between the Mid-RT and Pre-RT images can be effectively used to quantify the tumor's spatial response to treatment. Figure 2.1 also suggests that a tumor typically presents spatially-correlated and spatially-heterogeneous responses. Darker areas (dead tumor cells with no radiotracer uptake) may respond well to treatment, while brighter areas (metabolically active tumor cells that are still taking up radiotracer) appear to be less responsive. The importance of capturing such spatially-varying and spatially-correlated responses has been thoroughly discussed in Bowen et al., 2019.

A tumor is a 3D object. Figure 2.2 shows the SUV in three-dimensional spaces for another

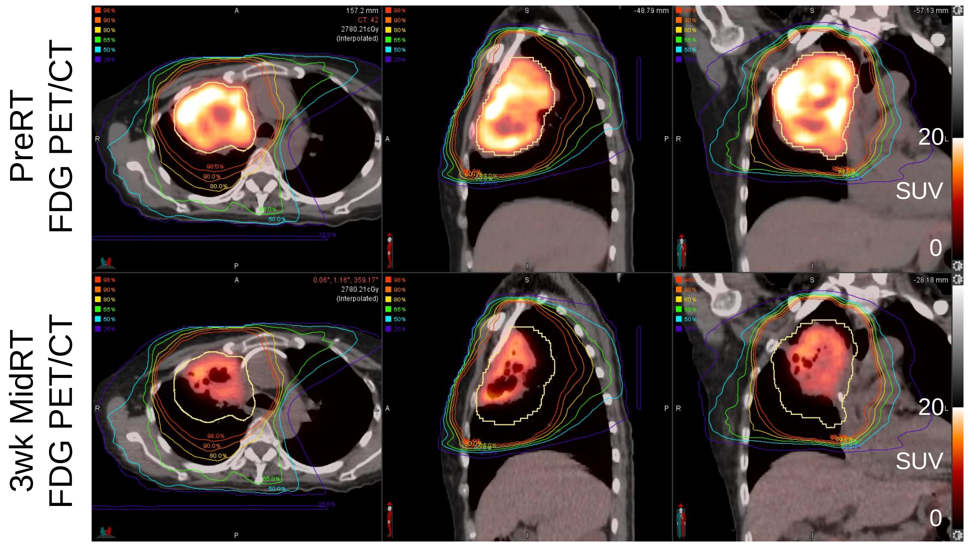
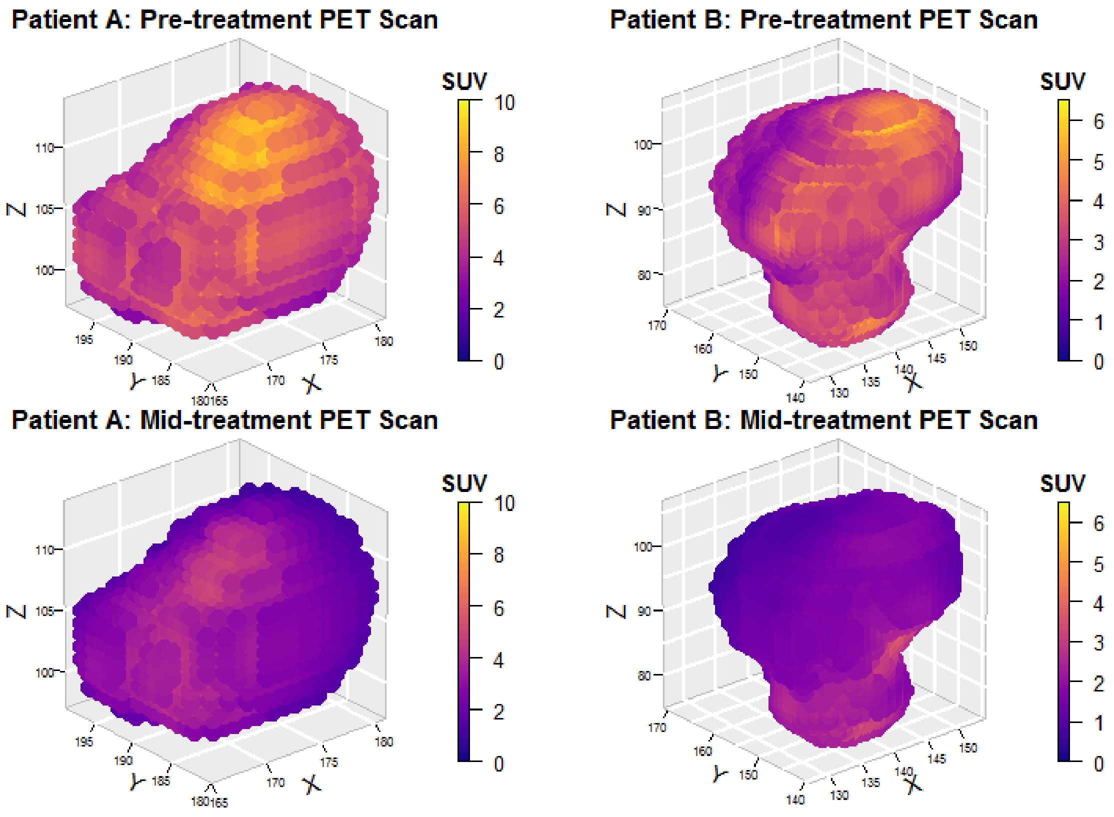


Figure 2.1: An example of spatially-correlated Pre-RT and Mid-RT FDG-PET images. The khaki-colored contour is the pre-treatment metabolic tumor volume. The rainbow contours are radiation isodose lines corresponding to varying levels of radiation dose, 98%, 90%, 80%, 65%, 50%, and 25% respectively.

two patients. The top row shows the Pre-RT data, while the bottom row shows the Mid-RT data collected three weeks after the radiotherapy. It is seen that,

- The overall SUV level decreases three weeks after the radiotherapy;
- The change in SUV, i.e., tumor's response to radiotherapy, varies over space. In general, the SUV level is higher in the center of the tumor and gradually decreases from the centroid to the surface of a tumor.

Hence, the capability of modeling and predicting the change of SUV is critical for treatment optimization and control. To model the change of SUV at location s within the spatial domain $\mathcal{S}$ occupied by a tumor, a statistical spatial model is needed that enables clinicians to understand how the change of SUV depends on a vector of features (including geometric features, therapy dosage, and so on). However, the complex relationship between the change of SUV and available features can hardly be directly specified. The non-linear effects, as well as the interactions among features, motivate us to investigate the non-parametric tree-based methods in this paper.



Patient A: Pre-treatment PET Scan
Patient B: Pre-treatment PET Scan
Patient A: Mid-treatment PET Scan
Patient B: Mid-treatment PET Scan
SUV
Z
Y
X

Figure 2.2: 3D Pre-RT and Mid-RT SUV images from two patients

2.1.1 Literature Review

Spatial data refer to an important type of data which arise in a spatial area and are often correlated in space. Capturing such correlation is the centerpiece of statistical analysis of spatial data (Cressie & Wikle, 2011b; Schabenberger & Gotway, 2005). Applications of statistical spatial data modeling can be found in a spectrum of scientific and engineering applications ranging from energy (Ezzat et al., 2019), reliability (Fang et al., 2019; Liu et al., 2018c), quality and manufacturing (Wang et al., 2016; Yue et al., 2020; Zang & Qiu, 2019), environmental and natural process (Guinness & Stein, 2013; Liu et al., 2018a; Liu et al., 2020), medical informatics (Yan et al., 2019; Yao & Yang, 2021; Yao et al., 2017), etc.

The pioneering work of statistical modeling of spatial data can be found in Banerjee et al., 2004, Schabenberger and Gotway, 2005 and Cressie and Wikle, 2011a. The mainstay approach, also known as the geostatistical paradigm, models spatial processes by random fields with fully specified covariance functions. A linear relationship between covariates and process mean is often assumed, and the model parameters can be obtained through Generalized Least Squares or Maximum Likelihood Estimation. The covariance structures are typically derived from moments of multivariate probability distributions and motivated by considerations of mathematical convenience (e.g., stationary, isotropic, space-time separable, etc.). Hence, there have been prolonged research interests to provide flexible and effective ways to construct non-stationary covariance functions (Cressie & Huang, 1999; Fuentes et al., 2005; Ghosh et al., 2010; Gneiting, 2002; Gneiting et al., 2006; Lenzi et al., 2019; Reich et al., 2011). The geostatistical modeling paradigm, which heavily relies on random fields, becomes less practical for large problems and approximations are commonly used, such as the Gaussian Markov Random Fields representation (Lindgren & Rue, 2011), Nearest-Neighbor Gaussian Process (Banerjee, 2017; Datta et al., 2016), kernel convolution (Higdon, 1998), low-rank representation (Banerjee et al., 2008; Cressie & Johannesson, 2002; Nychka & Wikle, 2002), the approximation of likelihood functions (Fuentes, 2007; Guinness & Fuentes, 2015; Stein et al., 2004), Bayesian inference for latent Gaussian models based on the integrated nested Laplace approximations (R-INLA, 2019; Rue et al., 2009), Lagrangian spatio-temporal covariance function (Gneiting et al., 2006), matrix-free state-space model (Mondal & Wang, 2019), Vecchia approximations of Gaussian processes (Katzfuss et al., 2020), as well as the multi-resolution approximation (M-RA) of Gaussian processes observed at irregular spatial locations (Katzfuss, 2017).

Other spatial models have also been proposed in the literature. Notably, the Markov Random Fields (MRF) model focuses on the (conditional) distribution of the quantity at a particular spatial location given the neighboring observations, such as the auto Poisson model (Besag, 1974), Conditional Autoregressive Model (Carlin & Banerjee, 2003; Liu et al., 2018b), etc. The Spatial Moving Average (SMA) approach models a spatial process through a process convolution with a convolution kernel (Brown et al., 2000; Higdon, 1998; Liu et al., 2016). There is also a large body of literature focusing on spatio-temporal data. For example, the Hierarchical Dynamical

Spatio-Temporal Models (DSTM) (Berliner, 2003; Cressie & Wikle, 2011a; Katzfuss et al., 2020; Stroud et al., 2010; Wikle & Cressie, 1999), and the SPDE-based modeling approach that aims to integrate governing physics into statistical spatio-temporal models (Brown et al., 2000; Hooten & Wikle, 2008; Liu et al., 2020; Sigrist et al., 2015; Stroud et al., 2010). A summary of the latest advances in the spatial modeling with SPDE can be found in Cressie and Wikle, 2011a and Krainski et al., 2019.

2.1.2 Challenges and Contributions

One challenge arising from practice is to specify the relationship between features and responses, which can rarely be adequately captured by linear models. For the FDG-PET imaging data, for example, both non-linear and interaction effects are expected between tumor's response and features (such as treatment, geometric features of the tumor, etc.). Hence, non-parametric approaches, especially the additive-tree-based approaches, provide some major modeling advantages. Constructing a tree does not require parametric assumptions on the complex relationship between features and responses. An individual tree performs a partition of the feature space. For each sub feature space (represented by a tree leaf), a predicted value is found for the individuals over that sub feature space. A sum-of-trees model consists of multivariate components that effectively handle the complex interaction effects among features (Chipman et al., 2010). Feature selection is also possible under the framework of additive-tree-based models (Hastie et al., 2009; Liu & Pan, 2020).

Among the additive-tree-based methods, gradient boosted trees have become one of the most successful statistical learning approaches, generating 17 winning solutions among 29 Kaggle challenges in 2015 (Chen & Guestrin, 2016). Schapire, 1999 introduced the first applicable Boosting method. The main idea of gradient boosting hinges on fitting a sequence of correlated "weaker learners" (such as simple trees). Each tree explains only a small amount of variation not captured by previous trees (Chipman et al., 2010; Hastie et al., 2009).

In fact, gradient boosted trees have been used for medical applications. Leveraging an open-source implementation, Oguz et al., 2017 illustrated the gradient boosted trees in a corrective learning for the segmentation of the caudate nucleus, putamen and hippocampus. de Melo et al., 2018 applied Light Gradient Boosting Machine to detect the posterior wall of the left ventricle from echocardiogram images. Fu et al., 2020 developed gradient boosted trees classifier to diagnose the lesion as malignant or benign for Breast Cancer Diagnosis. Nishio et al., 2018 performed the comparison between support vector machine and XGBoost for computer-aided diagnosis of lung nodule. Xie et al., 2019 applied extreme gradient boosting and gradient boosting machine to predict modified Rankin scale scores using biomarkers for 512 patients enrolled in a retrospective study.

Among the vast majority of existing tree-based methods, a key assumption is that observations are independent. Many existing boosting trees, such as XGBoost, do not consider the possible spatial correlation at all when they are applied to spatial data. To our best knowledge, Sigrist, 2020 recently proposed the only boosted-trees-based approach for Gaussian Process (which captures correlated errors). Such a method minimizes the negative log-likelihood at each tree node splitting, and is available in the R package, GPBoost. The package also provides a range of regularization and tuning parameter options. As also noted by Sigrist, 2020,"In many state-of-the-art supervised machine learning algorithms, in particular in boosting, it is assumed that a flexible and potentially complex function relates a set of predictor variables to a response variable. Conditional on the predictor variables, the response variable is assumed to be independent across observations. This means that all residual correlation, i.e. correlation that is not accounted for by the regression function, is ignored. Modeling such correlation allows not only for better learning of the regression function, but it is also important for prediction, in particular for probabilistic prediction and for predicting averages or sums over space (block kriging), time, and groups or clusters."

In our paper, on the other hand, regularization terms are directly added to the objective function in order to control the complexity of individual trees (i.e., number of leaves and leaf weights), leading to a regularized optimization problem at each tree node splitting. The regularization terms are motivated by the fundamental idea behind boosting trees which involves a sequence of correlated simple trees. This idea has been adopted by XGBoost (Chen & Guestrin, 2016), while an alternative approach adopted by the Bayesian Additive Regression Trees (BART) involves assigning prior distributions on parameters characterising the tree structure (Chipman et al., 2010). Hence, the **main contribution** of the paper is to propose a computationally-efficient gradient boosting method for growing the ensemble trees for spatially-correlated data. Each tree is grown by solving a regularized optimization problem, where the objective function involves regularizations on tree complexity and takes into account the underlying spatial correlation. The proposed algorithm is referred to as Boost-S, which stands for Gradient Boosted Trees for Spatial Data with covariate information (Boost-S). Boost-S integrates the spatial correlation structure into the classical framework of gradient boosted trees. In Statistics, Ordinary Least Squares is extended to Generalized Least Squares for correlated data. **An analogous notion can be formulated here when extending the classical framework of gradient boosted trees to spatially-correlated data, giving rise to the proposed Boost-S**.

The rest of this paper is structured as follows: Section 2.2 presents the technical details of the Boost-S algorithm. The application of Boost-S is presented in Section 2.3. Section 5 concludes the paper.

2.2 Boost-S: Technical Details

2.2.1 The Problem Statement

In this paper, we are concerned with the modeling problem:

$$Y(\boldsymbol{s}) = Z(\boldsymbol{x}_{\boldsymbol{s}}) + \varepsilon(\boldsymbol{s}), \qquad \boldsymbol{s} \in \mathcal{S} \tag{2.1}$$

where $Y(\boldsymbol{s})$ represents the observation at a spatial location $\boldsymbol{s} \in \mathcal{S}$, $\boldsymbol{x}_{\boldsymbol{s}}$ is a vector that collects the available features at $\boldsymbol{s}$, $Z(\boldsymbol{x}) \equiv \mathbb{E}(Y; \boldsymbol{x})$ is the mean-value function given $\boldsymbol{x}$, and $\varepsilon(\boldsymbol{s})$ is an isotropic and weakly stationary spatial noise process with zero-mean and a covariance $C(h)$, where $C(h) = \text{cov}(Y(\boldsymbol{s}), Y(\boldsymbol{s}'))$ with h being some distance measure between $\boldsymbol{s}$ and $\boldsymbol{s}'$; for example, the Euclidean distance.

The goal of this paper is to tackle the modeling problem (2.1) by devising a new *additive-tree-based* method that approximates $Z(\boldsymbol{x})$ as follows:

$$Z(\boldsymbol{x}) \approx \phi(\boldsymbol{x}) = \sum_{k=1}^{K} f_k(\boldsymbol{x}), \qquad f \in \mathcal{F} \tag{2.2}$$

where $\{f_k(\boldsymbol{x})\}_{k=1}^{K}$ represents an ensemble of binary decision trees, and $\mathcal{F}$ represents the tree space.

Apparently, the problem hinges on **how the ensemble of trees $\{f_k(\boldsymbol{x})\}_{k=1}^{K}$ can be grown, while taking into account the important spatial correlation of $Y(\boldsymbol{s})$**. In particular, we exploit the idea of gradient boosting which involves sequentially growing an ensemble of simple regression trees. Mathematically, each tree is added to the ensemble by solving an optimization problem with a carefully chosen objective function. This goal boils down to three fundamental tasks to be addressed in this paper: (***i***) formulate a (regularized) optimization problem that balances the complexity of individual trees and takes into account the spatial correlation associated with $Y(\boldsymbol{s})$; (***ii***) devise an algorithm that solves the regularized optimization problem in a computationally efficient manner (so that trees can be added sequentially to the ensemble); and (***iii***) validate the performance of the proposed method on real datasets.

Suppose that data are collected from a number of n spatial locations, $\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_n$. At each location, we observe a response y and a m-dimensional feature vector $\boldsymbol{x} = (x_1, x_2, ..., x_m)^T$. From (2.1) and (2.2), Boost-S aims to construct an ensemble of binary trees, $\{f_k(\boldsymbol{x})\}_{k=1}^{K}$, using gradient boosting such that

$$Y(\boldsymbol{s}) = \sum_{k=1}^{K} f_k(\boldsymbol{x}) + \varepsilon(\boldsymbol{s}), \qquad \boldsymbol{s} \in \{\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_n\}. \tag{2.3}$$

Let $\boldsymbol{Y} = (Y(\boldsymbol{s}_1), Y(\boldsymbol{s}_2), ..., Y(\boldsymbol{s}_n))^T$ be a multivariate random vector representing the responses from the n spatial locations, and let $\boldsymbol{f}^{(k)}$ be a vector of predicted values at the n spatial locations generated from the kth tree ($k \geq 0$), we re-write (2.3) as

$$\boldsymbol{Y} = \sum_{k=0}^{K} \boldsymbol{f}^{(k)} + \boldsymbol{\varepsilon}, \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}}) \tag{2.4}$$

where $\boldsymbol{f}^{(0)}$ is a vector of zeros corresponding to the initial condition when no tree has been grown.

2.2.2 A Regularized Problem

This subsection presents the detailed tree structures and formulates a regularized optimization problem that leads to a sequence of ensemble trees. In (2.3), each tree f_k is a Classification and Regression Tree (CART) that resides in a binary tree space

$$\mathcal{F} = \left\{ f(\boldsymbol{x}) = w_{q(\boldsymbol{x})} \right\} \tag{2.5}$$

where $q : \mathcal{R}^p \to T$ and $w \in \mathcal{R}^T$. Here, T represents the number of tree leaves (i.e., terminal nodes), w is the value on a tree leaf (i.e., leaf weight), and q determines the tree structure (i.e., a mapping that links a feature vector $\boldsymbol{x}$ to a tree leaf).

Suppose that a number of $k-1$ trees have been grown ($k \geq 1$). Then, the (ensemble) predicted values at the n spatial locations are given by $\hat{\boldsymbol{y}}^{(k-1)} = \sum_{j=0}^{k-1} \boldsymbol{f}^{(j)}$ from the $k-1$ trees. An immediate next step is to construct the kth tree and add the new tree to the ensemble such that $\hat{\boldsymbol{y}}^{(k)} = \hat{\boldsymbol{y}}^{(k-1)} + \boldsymbol{f}^{(k)} = \sum_{j=0}^{k} \boldsymbol{f}^{(j)}$. For binary trees, this involves finding the optimal split features as well as the split points for the kth tree. This task can be formulated as a regularized optimization problem:

$$\min_{\boldsymbol{f}^{(k)}} \left\{ \ell(\hat{\boldsymbol{y}}^{(k-1)} + \boldsymbol{f}^{(k)}) + \Omega(\boldsymbol{f}^{(k)}) \right\} \tag{2.6}$$

where ℓ is a loss function that depends on the output of the kth tree, and the regularization Ω is given by:

$$\Omega(\boldsymbol{f}) = \gamma T + \frac{1}{2}\lambda\|\boldsymbol{w}\|^2. \tag{2.7}$$

The first term, γT, regularizes the depth of the tree (by penalizing the total number of leaves), while the second term is used to regularize the contribution of tree k to the ensemble predictions (by penalizing the weights on tree leaves). Recall that, the fundamental idea behind boosting trees is to construct a sequence of correlated "weaker learners" (i.e., simple trees), where each "weaker learner" is added to explain the unexplained variation by existing trees in the ensemble (Chipman et al., 2010). Hence, the regularization (2.7) effectively controls the complexity of individual trees. In fact, it is worth noting that the regularization also helps to prevent the well-known overfitting issue of boosting trees. When a sufficient number of trees have been included in the ensemble, the penalty of adding one more tree may dominate the benefit (of adding more trees), which stops the algorithm from growing more trees.

Because the multivariate response $\boldsymbol{Y}$ is spatially correlated with the covariance matrix $\boldsymbol{\Sigma_\theta}$, a sensible choice for the loss function ℓ is the squared Mahalanobis length of the residual vector:

$$\begin{aligned} \ell(\hat{\boldsymbol{y}}^{(k-1)} + \boldsymbol{f}^{(k)}) &= \ell(\hat{\boldsymbol{y}}^{(k)}) \\ &\equiv (\boldsymbol{y} - \hat{\boldsymbol{y}}^{(k)})^T \boldsymbol{\Sigma}_\theta^{-1}(\boldsymbol{y} - \hat{\boldsymbol{y}}^{(k)}) \end{aligned} \tag{2.8}$$

and (2.6) can thus be written as

$$\min_{\boldsymbol{f}^{(k)}} \left\{ (\boldsymbol{y} - \hat{\boldsymbol{y}}^{(k)})^T \boldsymbol{\Sigma}_\theta^{-1}(\boldsymbol{y} - \hat{\boldsymbol{y}}^{(k)}) + \gamma T + \frac{1}{2}\lambda\|\boldsymbol{w}\|^2 \right\}. \tag{2.9}$$

Note that, (2.9) above extends the classical XGBoost which does not consider the correlation among the elements of $\boldsymbol{Y}$ (Chen & Guestrin, 2016). The extension made by this paper is in analogy to the extension from Ordinary Least Squares to Generalized Least Squares. However, such an extension requires new algorithms for the problem (2.9) to be efficiently solved.

The regularized optimization above is a formidable combinatorial optimization problem which can hardly be directly solved. Hence, we approximate the objective function $\ell(\hat{\boldsymbol{y}}^{(k)}) + \Omega(\boldsymbol{f}^{(k)})$ by a second-order multivariate Taylor expansion:

$$\begin{aligned}
&\ell(\hat{\boldsymbol{y}}^{(k)}) + \Omega(\boldsymbol{f}^{(k)}) \\
&\qquad \approx \ell(\hat{\boldsymbol{y}}^{(k-1)}) + \boldsymbol{g}^T \boldsymbol{f}^{(k)} + \frac{1}{2}(\boldsymbol{f}^{(k)})^T \boldsymbol{H} \boldsymbol{f}^{(k)} + \Omega(\boldsymbol{f}^{(k)})
\end{aligned} \tag{2.10}$$

where $\boldsymbol{g}$ is the column gradient vector of the loss function with its ith element given by $g_i = \partial \ell(\hat{\boldsymbol{y}}^{(k-1)})/\partial \hat{y}_i^{(k-1)}$, and $\boldsymbol{H}$ is the Hessian matrix with its (i,j)th entry being given by $h_{i,j} = \partial^2 \ell(\hat{\boldsymbol{y}}^{(k-1)})/\partial \hat{y}_i^{(k-1)} \partial \hat{y}_j^{(k-1)}$.

The Taylor expansion approximates the original combinatorial optimization problem (2.9), and enables us to develop the greedy algorithm that yields a heuristic solution. Note that, the truncation error of the Taylor expansion is $\sum_{|\alpha|\geq 3} \frac{d^\alpha \ell(\hat{\boldsymbol{y}}^{(k-1)})}{\alpha!}(\boldsymbol{f}^{(k)})^\alpha$ where d^α is α-th order partial derivatives of a multivariate function $\ell : \mathbb{R}^n \to \mathbb{R}$. Hence, for the truncation error to be small, $\boldsymbol{f}^{(k)}$ needs to be small (i.e., the contribution from the kth tree). Interestingly, $\boldsymbol{f}^{(k)}$ can be kept small through the penalty term, $\frac{1}{2}\lambda\|\boldsymbol{w}\|^2$, which controls the learning rate of the gradient boosted trees. This observation also helps to justify why a small learning rate is often preferred in training boosting trees.

Because the first term on the right-hand-side of (2.10) is a constant, it is sufficient to minimize the sum of the remaining three terms:

$$L^{(k)} = \boldsymbol{g}^T \boldsymbol{f}^{(k)} + \frac{1}{2}(\boldsymbol{f}^{(k)})^T \boldsymbol{H} \boldsymbol{f}^{(k)} + \Omega(\boldsymbol{f}^{(k)}). \tag{2.11}$$

Note that, for any given tree structure, it is possible to define a set $I_p = \{i \mid q(x_i) = p\}$ that consists of all samples that fall into leaf p. Then, we let $\boldsymbol{g}_p$ be a column vector that only retains the elements in $\boldsymbol{g}$ corresponding to samples in I_p, and similarly, let $\boldsymbol{H}_{p,q}$ be a matrix by only keeping the rows and columns of $\boldsymbol{H}$ corresponding to samples in I_p and I_q, respectively.

Figure 2.3 provides an illustration of how $\boldsymbol{g}_p$ and $\boldsymbol{H}_{p,q}$ are constructed. Consider a simple example where $n = 7$ (i.e., only 7 samples are available), then, the dimensions of the gradient vector $\boldsymbol{g}$ and the Hessian matrix $\boldsymbol{H}$ are 7×1 and 7×7, respectively. Suppose that $I_p = 1, 2, 6$ and $I_q = 3, 4$. Then, the vector $\boldsymbol{g}_p$ consists of the 1st, 2nd and the 6th element in $\boldsymbol{g}$, and the matrix $\boldsymbol{H}_{p,q}$ is a 3×2 matrix that retains the entries $\boldsymbol{H}$ located at the intersections of rows 1, 2 and 6, and columns 3 and 4.

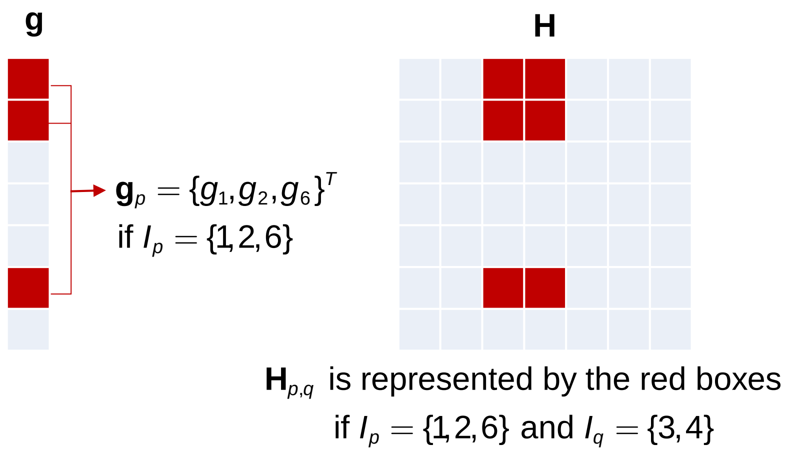

g
H
$\mathbf{g}_p = \{g_1, g_2, g_6\}^T$
if $I_p = \{1, 2, 6\}$
$\mathbf{H}_{p,q}$ is represented by the red boxes
if $I_p = \{1, 2, 6\}$ and $I_q = \{3, 4\}$

Figure 2.3: An illustration of the vector $\boldsymbol{g}_p$ and the matrix $\boldsymbol{H}_{p,q}$

Then, for any given tree structure $q(\boldsymbol{x})$, we re-write (2.11) as follows:

$$\begin{aligned} L^{(k)} = & \sum_{p=1}^{T} \boldsymbol{g}_p^T \boldsymbol{w}_p + \frac{1}{2} \left\{ \sum_{p \in T} \boldsymbol{w}_p^T \boldsymbol{H}_{p,p} \boldsymbol{w}_p \right\} \\ & + \frac{1}{2} \left\{ \sum_{(p,q) \in C_2^T} \boldsymbol{w}_p^T \boldsymbol{H}_{p,q} \boldsymbol{w}_q \right\} + \Omega(\boldsymbol{f}^{(k)}) \end{aligned} \tag{2.12}$$

where C_2^T denotes the combination (i.e., the number of 2-combinations from a given set of T elements), $\boldsymbol{w}_p = w_p \mathbf{1}_{|I_p|}$ with w_p and $\mathbf{1}_{|I_p|}$ respectively being the weight on tree leaf p and a column vector of ones of dimension $|I_p|$, and $|\cdot|$ represents the cardinality of a set.

Substituting (2.7) into (2.12) yields

$$\begin{aligned} L^{(k)} =&\gamma T + \sum_{p=1}^{T}\left\{ w_p \sum_{i\in I_p} g_i + \frac{1}{2}\left[\lambda + \sum_{(i,j)\in I_p} h_{i,j}\right] w_p^2 \right\} \\ &+ \frac{1}{2}\sum_{p=1}^{T}\left\{ \sum_{q=1;q\neq p}^{T} \left[\lambda + \sum_{i\in I_p;j\in I_q} h_{i,j}\right] w_p w_q \right\}. \end{aligned} \tag{2.13}$$

Taking the partial derivatives of (2.13) with respect to $\{w_p\}_{p=1}^{T}$ leads to a system of equations, which provides the key *computational advantag*e: given any tree structure, the optimal weights $\boldsymbol{w} = \{w_1, w_2, ..., w_T\}$ can be quickly found by solving the linear system:

$$\Xi \boldsymbol{w} = -\tilde{\boldsymbol{g}} \tag{2.14}$$

where Ξ is a $T \times T$ matrix with its pth row given by

$$\frac{1}{2}(\lambda + \sum_{i\in I_p;j\in I_1} h_{i,1}), \frac{1}{2}(\lambda + \sum_{i\in I_p;j\in I_2} h_{i,2}), ..., (\lambda + \sum_{(i,j)\in I_p} h_{i,j}), ..., \frac{1}{2}(\lambda + \sum_{i\in I_p;j\in I_T} h_{i,T}) \tag{2.15}$$

and $\tilde{\boldsymbol{g}}$ is a $T \times 1$ vector with its pth element given by $\sum_{i\in I_p} g_i$.

Obtaining the linear system (2.14) plays an extremely important role in searching for the optimal tree: given any candidate tree structure, it is possible to quickly and accurately find the optimal weights $\boldsymbol{w}$ on the leaf nodes by solving (2.14) using least squares, i.e., no numerical search is required. Substituting the optimal $\boldsymbol{w}$ into (2.11) immediately yields the value of the objective function $L^{(k)}$.

2.2.3 The Sequential Update of the Unknown Covariance Matrix

Constructing the linear system (2.14) and evaluating the objective function $L^{(k)}$ require a known covariance matrix of the errors, $\boldsymbol{\Sigma_\theta}$. However, $\boldsymbol{\Sigma_\theta}$ is not known and needs to be estimated before the first tree ($k = 1$), or any subsequent tree ($k > 1$), can be constructed. In this section, we describe how $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ can be consistently estimated before the kth tree is constructed, given the outputs from the first $k-1$ trees.

Suppose that $k-1$ trees have been constructed ($k \geq 1$), and let $\boldsymbol{r}^{(k-1)} = \boldsymbol{Y} - \sum \boldsymbol{f}^{(k-1)}$ be the residual vector. It follows from (2.4) that $\boldsymbol{r}^{(k-1)}$ is Gaussian with the covariance $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$. Note that, the mean of $\boldsymbol{r}^{(k-1)}$ may not even be close to zero when k is small, i.e., when the ensemble has not yet had sufficient trees to well capture the mean of $\boldsymbol{Y}$. In this case, we model $\boldsymbol{r}^{(k-1)}$ by a Locally Weighted Mixture of Linear Regressions (LWMLR) (Stroud et al., 2001). LWMLR is a spatial model constructed by Gaussian weighted kernels and linear regression surfaces that:

$$\boldsymbol{r}^{(k-1)} = \sum_{j=1}^{J} \pi_j^T(\boldsymbol{s})\boldsymbol{k}_j(\boldsymbol{s})\boldsymbol{\beta}_j + \varepsilon(\boldsymbol{s}), \qquad k \geq 1 \tag{2.16}$$

where J denotes the total number of kernels, $\boldsymbol{k}_j(\boldsymbol{s}) = \{k_{j1}(\boldsymbol{s}), ..., k_{jq}(\boldsymbol{s})\}$ is a set of spatial basis functions, $\boldsymbol{\beta}_j = (\beta_{j1}, ..., \beta_{jq})$ is a vector of unknown coefficients, $\pi_j(\boldsymbol{s})$ is a Gaussian kernel given as follows:

$$\pi_j(\boldsymbol{s}) \propto |\boldsymbol{V}_j|^{-1/2} \exp\left\{-\frac{1}{2}(\boldsymbol{s} - \boldsymbol{\mu}_j)^T \boldsymbol{V}_j^{-1}(\boldsymbol{s} - \boldsymbol{\mu}_j)\right\} \tag{2.17}$$

Note that, we may re-write (2.16) as a linear model, $\boldsymbol{r}^{(k-1)} = \boldsymbol{X}\boldsymbol{B} + \boldsymbol{\varepsilon}$, where $\boldsymbol{X} = (\text{diag}(\pi_1)\boldsymbol{X}_1, ..., \text{diag}(\pi_J)\boldsymbol{X}_J)$, $\boldsymbol{X}_j = (\boldsymbol{k}_j^T(\boldsymbol{s}_1), \boldsymbol{k}_j^T(\boldsymbol{s}_2), ..., \boldsymbol{k}_j^T(\boldsymbol{s}_n))^T$, and $\boldsymbol{B} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, ..., \boldsymbol{\beta}_J)^T$. To elaborate, we let $\boldsymbol{k}_j^T(\boldsymbol{s}_i) = (1, \boldsymbol{s}_{i1}, \boldsymbol{s}_{i2}, \boldsymbol{s}_{i3})^T$ represent the linear surfaces where $\boldsymbol{s}_{i1}$, $\boldsymbol{s}_{i2}$, and $\boldsymbol{s}_{i3}$ show the first, second, and third coordinates of location $\boldsymbol{s}$, respectively. Then, it is possible to obtain a consistent estimate of $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$, $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}}^{(k-1)}$, using the Feasible Generalized Least Square (FGLS), before the kth tree can be constructed.

2.2.4 The Boost-S Algorithm

Following the discussions above, Figure 2.4 provides a high-level illustration of the flow of the Boost-S algorithm. At the initialization stage, we obtain the initial estimate of the covariance matrix $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}}^{(0)}$. Then, by solving the regularized optimization problem (2.6), a new tree k is constructed and added to the ensemble. Next, the estimate of the covariance matrix is updated $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}}^{(k)}$ before tree $k+1$ can be constructed. The steps are repeated until K trees have been grown in the ensemble.

The Boost-S algorithm is summarized by Algorithm 1. It is possible to see that the computational complexity of Boost-S is $\mathcal{O}(Kdn^3 \log m)$, where K is the number of trees, d is the maximum depth of trees, n is the sample size, and m is the number of features. This computational complexity can be directly derived from the well-known complexity of XGBoost, which is $\mathcal{O}(Kdn \log m)$. Note that, Boost-S incorporates the covariance matrix into the loss function. In order to compute the second-order Taylor expansion, Boost-S needs to compute all elements in a dense Hessian matrix while XGBoost only calculates the second-order derivatives of the loss function on the diagonal line of the Hessian matrix. Hence, the time complexity is cubic in sample size n for Boost-S, while the time complexity is linear in sample size n for XGBoost. The results above are not surprising. Recall that, the time complexity is $\mathcal{O}(m^2 n)$ for Ordinary Least Squares, while the complexity of Generalized Least Squares becomes $\mathcal{O}(m^2 n^3)$ when correlated errors are taken into account.

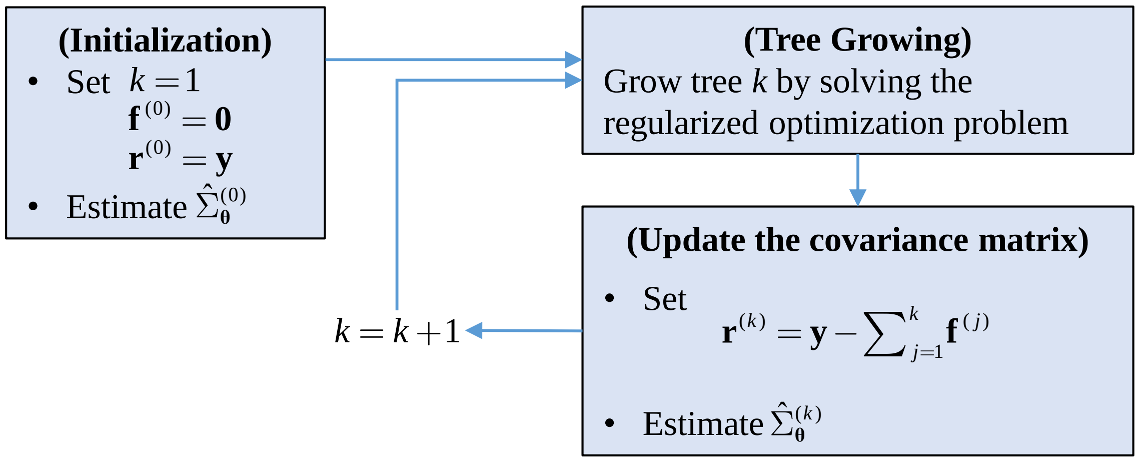

(Initialization)
• Set $k=1$
$\mathbf{f}^{(0)}=\mathbf{0}$
$\mathbf{r}^{(0)}=\mathbf{y}$
• Estimate $\hat{\Sigma}_{\boldsymbol{\theta}}^{(0)}$
(Tree Growing)
Grow tree k by solving the regularized optimization problem
(Update the covariance matrix)
• Set $\mathbf{r}^{(k)}=\mathbf{y}-\sum_{j=1}^{k}\mathbf{f}^{(j)}$
• Estimate $\hat{\Sigma}_{\boldsymbol{\theta}}^{(k)}$
$k=k+1$

Figure 2.4: An illustration of the flow of the Boost-S algorithm

Algorithm 1: Boost-S: Gradient Boosted Trees for Spatial Data

```
1  Set the values for $\lambda$, $\gamma$ and $K$
2  Let k = 0, $\boldsymbol{f}^{(0)} = \mathbf{0}$ and $\boldsymbol{r}^{(0)} = \boldsymbol{y}$
3  Obtain the initial estimate, $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}}^{(0)}$, from (2.16) using FGLS
4  for $k$=1,...,$K$ do
5      Grow tree $k$ by repeating the following steps:
6      (i) Given the current tree topology, generate a set of all possible new tree structures by
         splitting a tree node based on candidate split variables and candidate split values.
7      (ii) For each new tree structure, obtain the weights $\boldsymbol{\omega}$ on the leaf nodes by solving the
         linear system $\Xi \boldsymbol{w} = -\tilde{\boldsymbol{g}}$, and evaluate the objective function $L^{(k)}$ in (2.11) for the
         new topology.
8      (iii) If there exists at least one new tree structure that further reduces the objective
         function (over the existing tree structure), retain the new topology that generates the
         greatest reduction and go to (i); otherwise, terminate the tree growing process for tree
         $k$, and go to the next step.
9      (iv) Update the estimate $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}}^{(k)}$using FGLS.
10 end
```

2.3 Illustration of Boost-S on the FDG-PET Imaging Data

This section re-visits the motivating application presented in Section 4.1. We apply Boost-S to real datasets and compare the performance of Boost-S to that of existing approaches.

2.3.1 Data

The data used in this section are obtained from 25 patients diagnosed with locally advanced and surgically unresectable non-small cell lung cancer (NSCLC) who enrolled onto the FLARE-RT clinical trial (NCT02773238). All patients were consented under the approval of the University of Washington/Fred Hutchinson Cancer Consortium Institutional Review Board (CC IRB 9599). For each patient, this clinical trial data set contains geographic features, dosage, Pre-RT and Mid-RT SUV levels. As discussed in Section 4.1, the goal is to model the difference between the Pre-RT SUV (before treatment) and Mid-RT SUV (during the third week of treatment course), which helps clinicians further optimize or control the treatment plans. In fact, it allows oncologists to modify the dosage level or develop different follow-up plans by administering alternative therapies if response is minimal.

Let $Y(\boldsymbol{s})$ represent the ratio between Mid-RT SUV and Pre-RT SUV at location $\boldsymbol{s}$ (a voxel in the image) within the spatial domain $\mathcal{S}$ occupied by the tumor, i.e.,

$$Y(\boldsymbol{s}) = \frac{\text{Mid-RT SUV}}{\text{Pre-RT SUV}}. \tag{2.18}$$

If the treatment is effective, the Mid-RT SUV is expected to be lower than the Pre-RT SUV level. Hence, a lower ratio indicates more effective treatment.

2.3.2 Application of Boost-S

We first demonstrate the application of Boost-S on the data collected from one of the 25 patients. The PET scan for this patient has 3110 voxels (i.e., the number of spatial locations). We randomly split the data into two parts: 15% for training while 85% for testing. Such a low training-testing ratio is chosen to demonstrate the out-of-sample prediction capability of Boost-S constructed from a relatively small training dataset.

To obtain the initial estimate of the covariance matrix $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}}^{(0)}$, the FGLS is used to solve the linear model (2.16) in Algorithm 1. Before the FGLS is performed, one needs to first choose a parametric spatial covariance function $c(\cdot)$ of the process ε in (2.16). Figure 2.5 shows both the empirical semivariance and the fitted semivariance using FGLS assuming the Gaussian covariance function. The sill, range and nugget effect can be clearly seen from Figure 2.5, and the Gaussian covariance function appears to be an appropriate choice.

Algorithm 1 requires the tuning parameters λ and γ to be pre-specified. The choice of these two parameters affects both the depth and contribution of individual trees to the ensemble prediction, which in turn influences the total number of trees in the ensemble. A common strategy suggests that we explore the suitable values for λ and γ while leaving K as the primary parameter (Hastie et al., 2009). Although it is theoretically possible to perform a grid search for the best combinations of λ and γ on a two-dimensional space, such an approach may not be practical nor necessary in

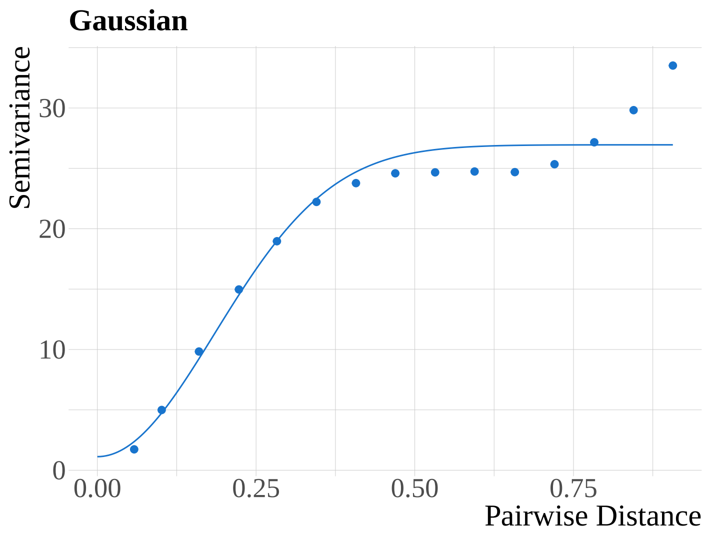


Figure 2.5: Exploratory analysis on the covariance structure: plot of the empirical and fitted semi-variance using FGLS

practice when it is computationally intensive to run Boost-S on big datasets. Hence, we resort to a powerful tool in computer experiments—the space-filling designs (Joseph, 2016). The idea of space-filling designs is to have points everywhere in the experimental region with as few gaps as possible, which serves our purpose very well. For example, the Maximum Projection Latin Hypercube Design (MaxPro) can be adopted. The main advantage of MaxPro is that such a design approach ensures good projections to all subspaces of the tuning parameters. Based on MaxPro, N pairs of tuning parameters (λ_i, γ_i), for $i = 1, 2, ..., N$, are found by minimizing the following criterion (Joseph et al., 2015)

$$\min \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \frac{1}{|\lambda_i - \lambda_j|^2 |\gamma_i - \gamma_j|^2} \tag{2.19}$$

Figure 2.6 shows the MaxPro design of 16 pairs of λ and γ, where the experimental ranges for these two parameters are respectively $[0, 0.1]$ and $[0, 10]$.

For each design, Figure 2.7 shows the box plot of the number of tree leaves per tree in an ensemble for each combination of λ and γ. Since the key idea behind boosting trees is that each individual tree needs to be kept simple (Hastie et al., 2009), we identify that Designs #7, #8 and

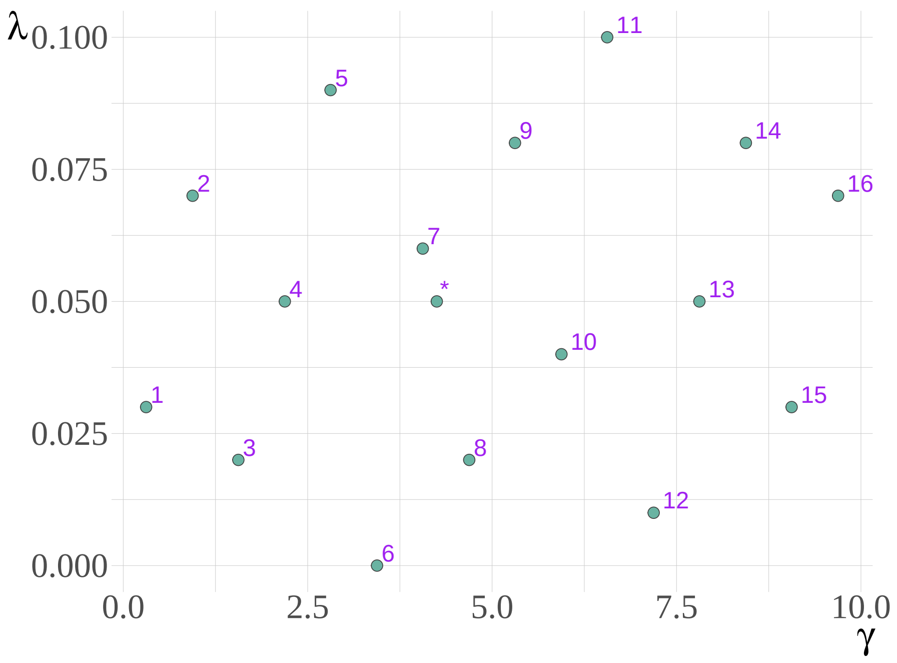


Figure 2.6: Maximum projection design of 16 runs with different combinations of λ and γ. Design "$*$" yields an appropriate combination such that $\lambda = 0.05$ and $\gamma = 4.25$.

#9 provide the most suitable combinations of λ and γ. Note that, the choice on the right-sized trees for boosting can be very empirical. For example, in Sec. 10.11 of Hastie et al., 2009, the authors suggest that "Experience so far indicates that $4 \leq T \leq 8$ (T is the number of leaves) works well in the context of boosting". As shown in Figure 2.7, trees based on Designs #1$\sim$#6 may be too complex (excessive number of leaves), while trees based on Designs #10$\sim$#16 may not have enough leaves. Note that, the Designs #7, #8 and #9 are adjacent to each other, indicating that the appropriate choices for λ and γ are approximately within $[0.025, 0.075]$ and $[4, 5.5]$. A refined search in a much smaller experimental region yields an appropriate combination of $\lambda = 0.05$ and $\gamma = 4.25$ (Design "$*$" in Figure 2.6). Design "$*$" is between Designs #7 and #8, and the 25th, 50th and 75th empirical quartiles of the number of tree leaves are 6, 8 and 10 as shown in Figure 2.7.

After λ and γ have been appropriately chosen, 49 trees are constructed using the training dataset to form the ensemble predictor using Algorithm 1. Figure 2.8 (top panel) shows that the Mahalanobis distance (i.e., the objective function) decreases as more trees have been included into the ensemble, indicating that the algorithm is working as expected. Figure 2.8 (bottom panel) shows

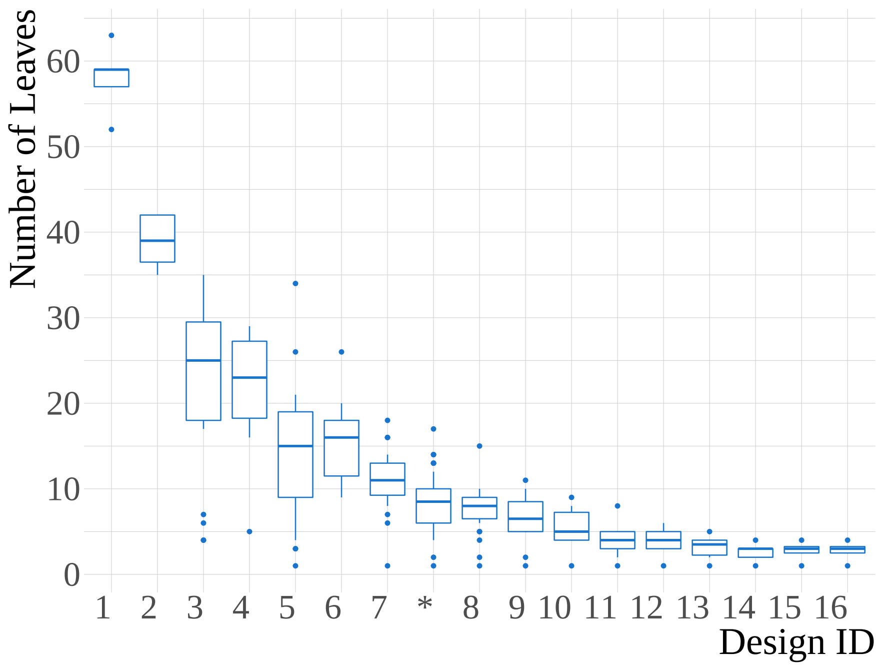


Figure 2.7: Boxplot of the number of tree leaves per tree in an ensemble for the 16 candidate designs. Design "$*$" indicates the chosen combination such that $\lambda = 0.05$ and $\gamma = 4.25$.

the number of tree leaves for individual trees in this ensemble. It is interesting to note that the algorithm no longer splits the (root) tree node after 37 trees have been grown. In addition, the outputs of these one-node trees are all zeros, indicating that all trees after tree 37 are completely redundant. This is precisely due to the regularization γT and $\frac{1}{2}\lambda\|\boldsymbol{w}\|^2$ in (2.6). The gain in ℓ is outweighted by the loss caused by Ω if one more tree is added.

Applying the constructed ensemble trees to the testing dataset, Figure 2.9 shows the out-of-sample Root-Mean-Square-Error (RMSE) and Mean-Gross-Error (MGE). We see that both performance metrics decrease as more trees are included and stabilize approximately after 30 to 40 trees have been grown. As discussed above, this is precisely due to the fact that all trees after #37 are redundant with zero output. Figure 2.10 shows the (out-of-sample) predictions against actual observations of the SUV level at different voxels. The figure shows that the proposed Boost-S accurately predicts the SUV levels given the covariate information.

Furthermore, we reconstruct the Mid-RT SUV (3 weeks after treatment) by the proposed Boost-S, and compare the actual and predicted Mid-RT SUV using slice plots (Figure 2.11). Because a

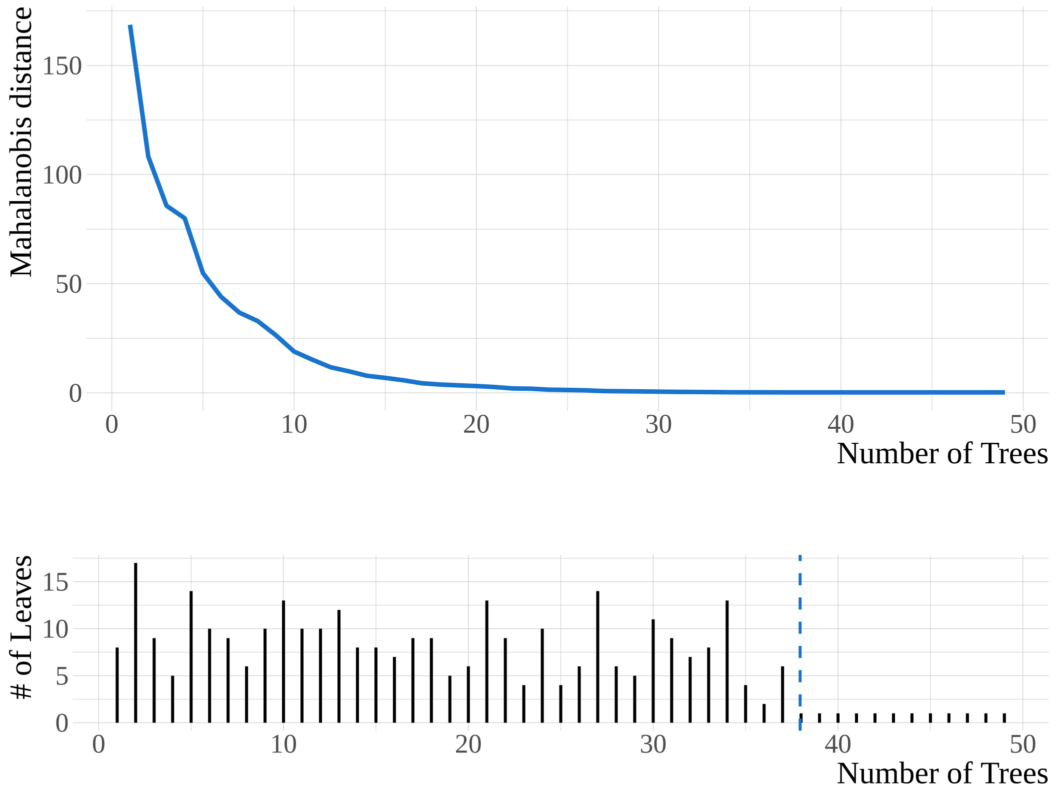


Figure 2.8: Top panel: the Mahalanobis distance decreases as the number of trees grows; Bottom panel: the number of tree leaves of individual trees.

tumor is a 3D object with three coordinates, x, y, and z, the MidPET images are shown on the $x - y$ plane for given values of z. We see that the ensemble trees are capable of predicting the Mid-RT SUV for the entire tumor body.

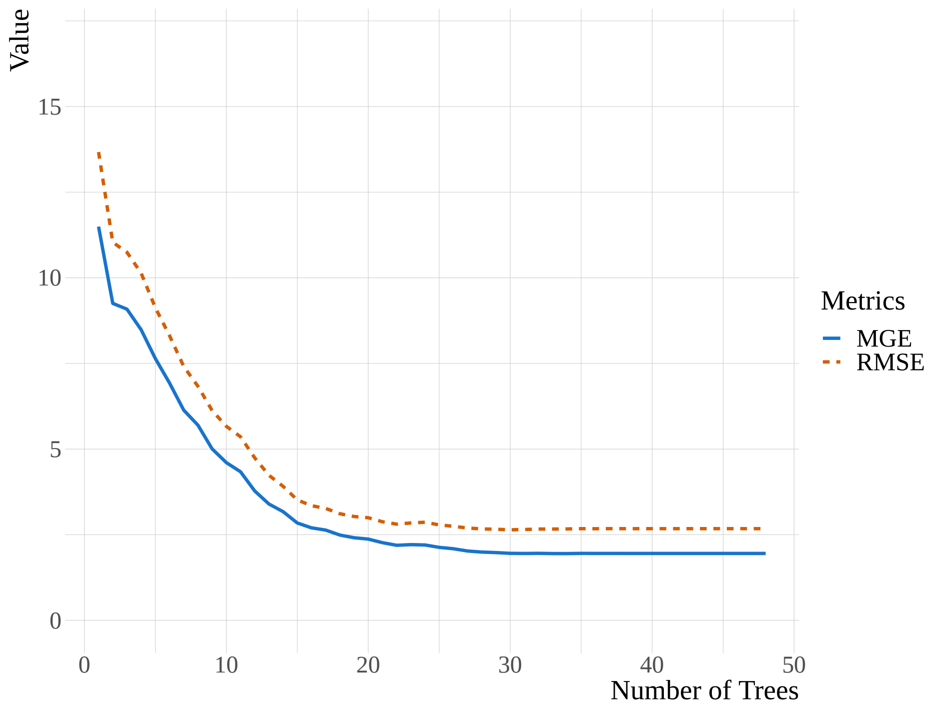


Figure 2.9: Out-of-sample model performance in terms of RMSE and MGE

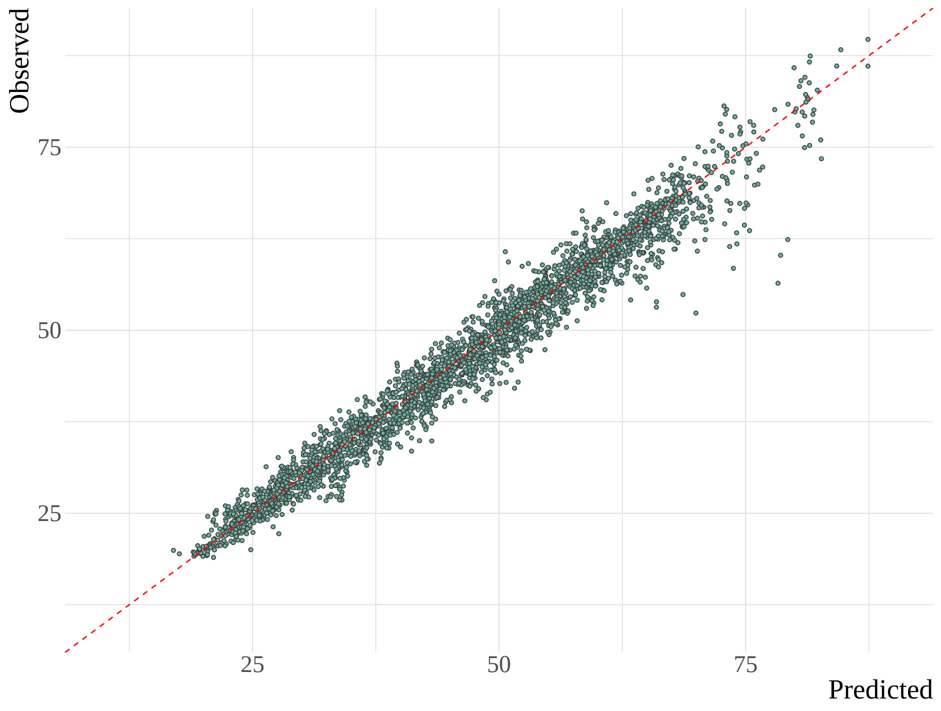


Figure 2.10: Out-of-sample predictions against actual observations

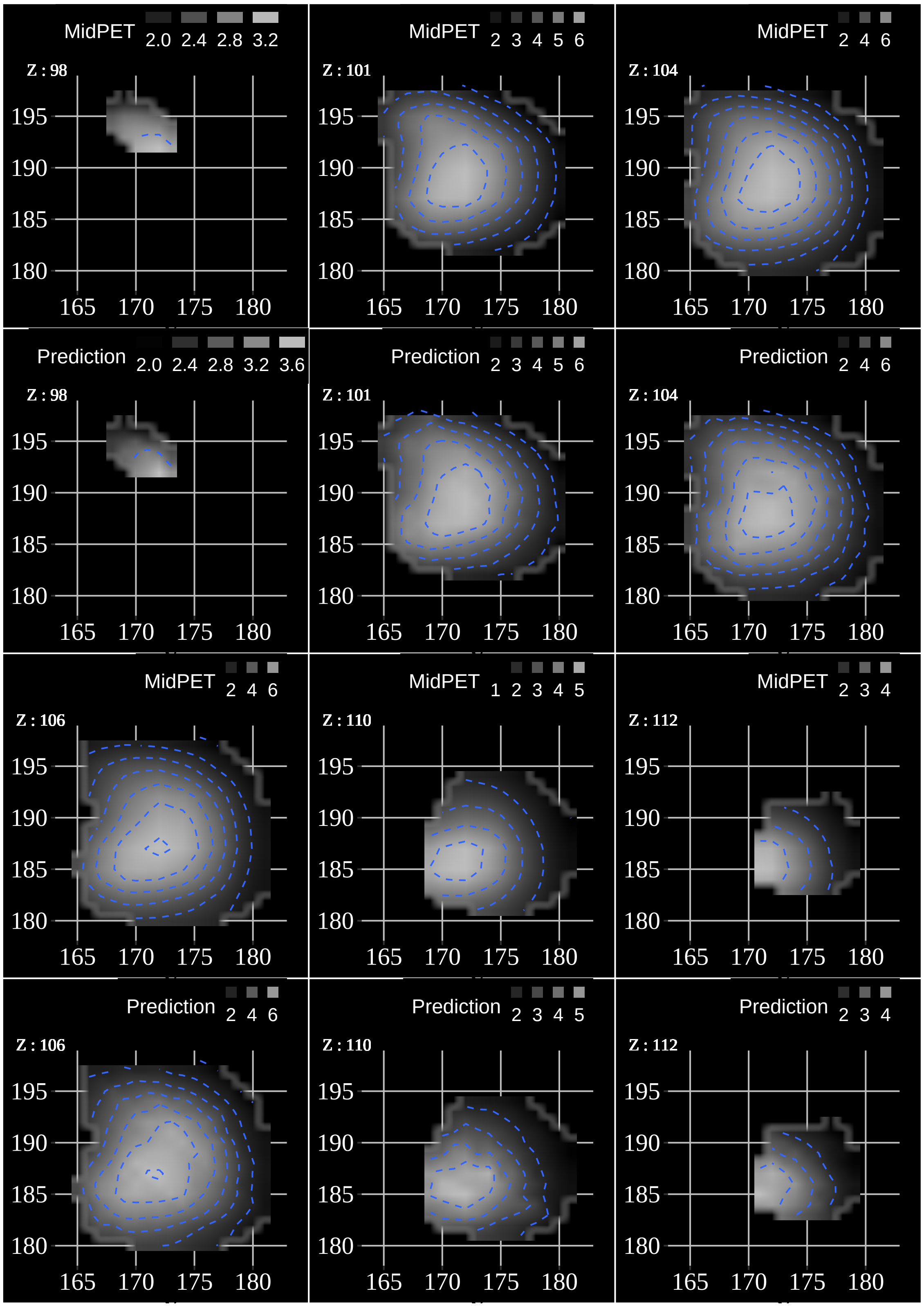


Figure 2.11: MidPET images on the $x - y$ plane for different values of z. Rows 1 and 3: observed images; Row 2 and 4: reconstructed images.

2.3.3 Comparison Study

We compare the predictive capability of Boost-S to six other methods using the data collected from 25 patients. The six methods included in the comparison study are: LightGBM (Ke et al., 2017), Random Forests (RF) (Breiman, 2001), Extreme Gradient Boosting Trees (XGBoost) (Chen & Guestrin, 2016), non-parametric cubic splines, universal kriging with a linear spatial trend, and multiple linear regression without considering the spatial correlation. For each patient, 15% of the observations are randomly chosen as the training dataset, and all 7 models are constructed to generate the out-of-sample predictions using the testing dataset. Such a low training-testing ratio is used to test the predictive capabilities of these methods.

Tables 1, 2 and 3 respectively show the Mean Gross Error (MGE), Relative Error (RE) and Rooted Mean Squared Error (RMSE) of the out-of-sample predictions for the 25 patients using different methods. MGE, RE, and RMSE are performance metrics for regression models. All of them involve calculating errors between predicted values and the actual values and are commonly used metrics to report and summarize the predictive advantages of a model. While the changes in MGE are linear and intuitive, RMSE punishes larger errors more than smaller errors; hence, it is more sensitive to outliers. This is due to the square of the error values where MGE does not give weights to errors. RE expresses the performance of a model in generic percentage terms and is more useful for whom that are not familiar with the unit of the data. It is interesting to observe that

- Boost-S yields the lowest MGE for 19 out of the 25 patients, while RF performs the best for the remaining 6 patients;
- Boost-S yields the lowest RE for 18 out of the 25 patients, while RF performs the best for the remaining 7 patients;
- Boost-S yields the lowest RMSE for 17 out of the 25 patients, while RF performs the best for the remaining 8 patients;

In addition, Tables 1, 2, and 3 also present the p-values of the Wilcoxon signed-rank test that compares Boost-S with other methods. The Wilcoxon signed-rank test is a non-parametric test to assess whether the population mean ranks differ (one may refer to Neuhaus, 1989 for the well-established procedures). It is possible to conclude that, **the proposed Boost-S provides the best performance for majority of the patients in terms of all three performance measures, although no method uniformly outperforms others (which is realistic and expected given the well-known modeling power of RF and XGBoost)**. The reason why Boost-S outperforms other additive-tree-based methods, i.e., RF and XGBoost, is likely because of its capability of accounting for the spatial correlation among observations (as this is the main differentiating feature of Boost-S relative to these other two methods). The reason why Boost-S outperforms universal Kriging and multiple linear regression may be due to the advantages of non-parametric tree-based method in capturing complex non-linear and interaction effects between features and responses. These results demonstrate that Boost-S is able to effectively combine the strengths of these different approaches in order to improve on their performance in this setting with spatial correlation among observations.

In the Appendix, Figures 2.12, 2.13 and 2.14 respectively show the MGE, RE and RMSE of the out-of-sample predictions for the 25 patients. Such visualizations provide a holistic view on the performance of the seven candidate methods.

Table 2.1: Mean Gross Error of the out-of-sample predictions for 25 patients using 7 different methods (note that: rows 1 to 25 respectively show the results corresponding to the 25 patients, while the last row shows the p-value of the one-side paired Wilcoxon test. The "NA" p-value is due to the fact that it is not necessary to compare Boost-S with itself.)

Patient	Boost-S	RF	LightGBM	XGBoost	Cubic Splines	Universal Kriging	Linear Regression
1	8.24	**6.78**	7.73	7.31	8.45	10.32	9.01
2	10.68	**9.61**	11.30	10.55	11.57	13.41	13.79
3	**5.84**	6.22	8.13	6.32	7.14	10.37	9.08
4	**4.30**	4.37	5.51	4.68	6.19	6.76	6.20
5	**3.15**	4.45	5.79	3.97	4.70	10.04	8.58
6	**1.95**	2.40	3.30	2.51	3.07	5.42	4.25
7	**5.20**	5.40	7.64	5.67	5.47	13.30	10.34
8	**5.04**	5.30	6.60	5.93	9.00	10.39	9.88
9	**8.59**	9.28	11.76	9.50	9.22	12.55	11.79
10	**2.37**	2.79	3.89	3.17	4.46	6.70	5.97
11	9.08	**8.63**	9.55	8.91	11.16	10.79	11.22
12	**1.44**	1.74	2.77	2.08	2.64	4.00	2.99
13	**1.65**	1.91	2.63	2.08	3.61	4.36	3.69
14	**0.74**	1.32	1.90	1.35	1.58	2.48	1.83
15	10.19	**9.77**	11.45	10.59	10.98	15.03	13.71
16	**4.17**	4.37	6.58	4.25	7.79	9.64	7.89
17	**2.71**	3.20	4.48	3.78	4.22	7.15	6.47
18	**6.53**	7.06	9.14	8.27	8.41	13.43	12.18
19	13.10	**10.62**	12.83	11.36	12.57	13.77	14.23
20	**3.51**	3.96	5.18	4.06	4.44	9.30	6.85
21	**1.22**	1.24	1.88	1.57	1.58	2.55	2.03
22	**4.17**	4.82	5.83	5.01	7.69	11.25	8.71
23	**5.47**	6.62	7.88	6.43	10.65	18.82	12.05
24	3.87	**3.63**	4.70	4.13	5.08	5.52	5.47
25	**1.86**	2.42	3.24	2.58	3.57	5.60	4.19
p-value	N.A.	0.04	$< 10^{-6}$	0.001	$< 10^{-6}$	$< 10^{-7}$	$< 10^{-7}$

Table 2.2: Relative Error (in %) of the out-of-sample predictions for 25 patients using 7 different methods (note that: rows 1 to 25 respectively show the results corresponding to the 25 patients, while the last row shows the p-value of the one-side paired Wilcoxon test. The "NA" p-value is due to the fact that it is not necessary to compare Boost-S with itself.)

Patient	Boost-S	RF	LightGBM	XGBoost	Cubic Splines	Universal Kriging	Linear Regression
1	9.67	**8.01**	9.10	8.58	9.91	12.29	10.68
2	56.62	**51.01**	61.28	55.37	60.03	60.47	67.27
3	**11.04**	12.52	15.79	11.93	13.72	18.40	17.01
4	**5.03**	5.34	6.65	5.53	7.60	8.36	7.61
5	**7.30**	9.41	12.28	8.26	10.77	20.25	19.07
6	**4.42**	5.73	7.53	5.65	7.29	12.52	9.72
7	**8.92**	9.41	13.52	9.52	9.72	20.53	18.06
8	23.25	**22.32**	27.34	25.52	43.34	43.51	43.78
9	**13.62**	15.45	19.11	15.18	14.45	20.04	18.91
10	**2.28**	2.74	3.80	3.07	4.26	6.48	5.77
11	11.44	**10.78**	12.31	10.94	14.42	13.41	14.39
12	**1.09**	1.32	2.11	1.55	1.99	3.06	2.25
13	**3.66**	4.25	5.82	4.55	8.00	9.70	8.20
14	**1.49**	2.75	3.89	2.72	3.17	4.70	3.59
15	21.40	**19.97**	23.97	21.21	23.26	32.02	29.69
16	**8.36**	8.67	13.14	8.13	15.27	20.18	15.49
17	**3.90**	4.68	6.49	5.34	6.05	9.93	9.51
18	**7.46**	8.79	11.28	9.97	9.99	17.54	15.31
19	21.70	**17.32**	21.40	18.19	20.08	21.71	23.53
20	**4.02**	4.58	6.01	4.60	5.05	10.15	7.95
21	**1.91**	1.94	2.91	2.43	2.53	4.08	3.24
22	**6.34**	7.18	8.67	7.61	11.55	17.54	13.37
23	**8.10**	9.55	11.61	8.73	16.69	25.45	18.96
24	5.24	**5.01**	6.55	5.63	7.10	7.85	7.61
25	**5.07**	6.59	8.69	6.86	9.72	14.92	11.26
p-value	N.A.	0.09	$< 10^{-6}$	0.004	$< 10^{-6}$	$< 10^{-7}$	$< 10^{-7}$

Table 2.3: Rooted Mean Squared Error of the out-of-sample predictions for 25 patients using 7 different methods (note that: rows 1 to 25 respectively show the results corresponding to the 25 patients, while the last row shows the p-value of the one-side paired Wilcoxon test. The "NA" p-value is due to the fact that it is not necessary to compare Boost-S with itself.)

Patient	Boost-S	RF	LightGBM	XGBoost	Cubic Splines	Universal Kriging	Linear Regression
1	11.05	**9.01**	10.05	9.66	10.89	13.00	11.66
2	16.46	**14.77**	15.88	16.22	16.14	19.66	19.42
3	**7.73**	7.83	10.42	8.10	9.17	12.89	11.61
4	5.74	**5.51**	6.96	5.89	7.73	8.28	7.74
5	**4.13**	5.97	7.75	5.41	6.00	12.83	10.64
6	**2.68**	3.11	4.27	3.24	3.89	6.68	5.36
7	7.03	**6.68**	10.16	7.13	6.98	17.10	12.64
8	**7.62**	7.88	9.86	8.58	11.50	14.09	13.07
9	**11.15**	11.69	14.64	12.02	11.80	15.57	15.04
10	**3.21**	3.80	5.09	4.32	5.97	8.28	7.66
11	12.52	**12.06**	12.76	12.41	14.47	14.68	14.87
12	**1.89**	2.20	3.48	2.64	3.64	4.75	4.02
13	**2.20**	2.46	3.38	2.66	4.56	5.10	4.68
14	**1.04**	1.69	2.38	1.72	1.95	3.21	2.37
15	14.17	**13.19**	15.05	14.17	14.29	18.66	17.32
16	**5.77**	5.94	9.01	5.71	10.18	12.27	10.31
17	**3.48**	4.15	5.84	4.73	5.30	8.99	8.28
18	**8.72**	8.98	11.69	10.50	10.83	16.69	15.26
19	19.26	**15.23**	17.88	16.30	17.69	20.01	19.80
20	**5.02**	5.21	6.50	5.29	5.94	11.18	8.33
21	**1.72**	1.82	2.75	2.22	2.09	3.29	2.72
22	**5.52**	6.61	8.09	6.59	9.75	13.63	11.06
23	**7.36**	8.73	10.18	8.68	13.29	23.28	15.15
24	5.10	**4.78**	6.05	5.41	6.63	7.07	7.07
25	**2.55**	3.16	4.16	3.34	4.67	7.14	5.39
p-value	N.A.	0.19	$< 10^{-5}$	0.003	$< 10^{-4}$	$< 10^{-7}$	$< 10^{-7}$

2.4 Conclusion

This paper proposed a new gradient Boosted Trees algorithm for Spatial Data with covariate information (Boost-S). It has been shown that Boost-S successfully integrates spatial correlation into the classical framework of gradient boosted trees. A computationally-efficient algorithm as well as the technical details have been presented. The Boost-S algorithm grows individual trees by solving a regularized optimization problem, where the objective function involves two penalty terms on tree complexity and takes into account the underlying spatial correlation. The advantages of the proposed Boost-S, over six other commonly used approaches, have been demonstrated using real datasets involving the spatially-correlated FDG-PET imaging data collected during concurrent cancer chemotherapy and radiotherapy. In the current application, Boost-S was applied to data from individual patients separately. One immediate future task is to extend the current algorithm such that it can simultaneously handle data from a group of patients involved in the FLARE-RT phase II clinical trial (NCT02772338). Random-effect model needs to be integrated into Boost-S so as to account for the between-patient variation. The long-run purpose of the project is to predict whether the treatment will be effective for a patient or not before performing the treatment. This task requires a more complicated model that involves random-effect that can account for patients' information such as sex, age, etc. To achieve such a model, we have to build a model on a patient-level basis that needs to be applied to multiple patients at once.

2.5 Acknowledgment

This investigation was supported in part by National Institutes of Health grant R01CA204301.

References

Banerjee, S. (2017). High-Dimensional Bayesian Geostatistics. *Bayesian Analysis*, *12*(2), 583–614.

Banerjee, S., Carlin, B. P., & Gelfand, A. E. (2004). *Hierarchical Modeling and Analysis for Spatial Data, 2nd ed.* CRC Press.

Banerjee, S., Gelfand, A. E., Finley, A. O., & Sang, H. (2008). Gaussian Predictive Proess Models for Large Spatial Data Sets. *Journal of the Royal Statistical Society: Series B*, *70*(4), 825–848.

Berliner, L. M. (2003). Physical-Statistical Modeling in Geophysics. *Journal of Geophysical Research-Atmospheres*, *108*(D24), 3–10.

Besag, J. E. (1974). Spatial Interaction and the Statistical Analysis of Lattice Systems. *Journal of the Royal Statistical Society, B*, *36*, 192–225.

Bowen, S. R., Hippe, D. S., Chaovalitwongse, W. A., Duan, C., Thammasorn, P., Liu, X., Miyaoka, R. S., Vesselle, H. J., Kinahan, P. E., Rengan, R., et al. (2019). Voxel Forecast for Precision Oncology: predicting spatially variant and multiscale cancer therapy response on longitudinal quantitative molecular imaging. *Clinical Cancer Research*, *25*(16), 5027–5037.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32.

Brown, P. E., Karesen, K. F., Roberts, G. O., & Tonellato, S. (2000). Blur-Generated Non-Separable Space-Time Models. *Journal of the Royal Statistical Society: Series B*, *62*(4), 847–860.

Carlin, B. P., & Banerjee, S. (2003). *Hierarchical Multivariate CAR Models for Spatio-Temporally Correlated Survival Data (with discussion)*. Oxford University Press.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Chipman, H. A., George, E. I., & McCulloch, R. E. (2010). Bart: Bayesian Additive Regression Trees. *The Annals of Applied Statistics*, *4*, 266–298.

Cressie, N., & Huang, H. C. (1999). Classes of Nonseparable, Spatio-Temporal Stationary Covariance Functions. *Journal of the American Statistical Association*, *94*(448), 1330–1340.

Cressie, N., & Johannesson, G. (2002). Fixed Rank Kriging for Very Large Spatial Data Sets. *Journal of the Royal Statistical Society: Series B*, *70*(1), 209–226.

Cressie, N., & Wikle, C. (2011a). *Statistics for Spatio-Temporal Data*. John Wiley & Sons.

Cressie, N., & Wikle, C. K. (2011b). *Statistics for Spatio-Temporal Data*. John Wiley & Sons.

Datta, A., Banerjee, S., Finley, A. O., & Gelfand, A. E. (2016). Hierarchical Nearest-Neighbor Gaussian Process Models for Large Geostatistical Datasets. *Journal of the American Statistical Association*, *111*(514), 800–812.

de Melo, V. V., Ushizima, D. M., Baracho, S. F., & Coelho, R. C. (2018). Gradient Boosting Decision Trees for Echocardiogram Images. *2018 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN.2018.8489523

Ezzat, A., Jun, M., & Ding, Y. (2019). Spatio-temporal short-term wind forecast: A calibrated regime-switching method. *The Annals of Applied Statistics*, 1484–1510.

Fang, X., Paynabar, K., & Gebraeel, N. (2019). Image-Based Prognostics Using Penalized Tensor Regression. *Technometrics*, *61*, 369–384.

Fu, Y., Patel, B. K., Wu, T., Li, J., & Gao, F. (2020). Advanced Medical Imaging Analytics in Breast Cancer Diagnosis. In A. Smith (Ed.), *Women in Industrial and Systems Engineering, Women in Engineering and Science*. Springer, Cham.

Fuentes, M. (2007). Approximate Likelihood for Large Irregularly Spaced Spatial Data. *Journal of the American Statistical Association*, *102*(477), 321–331.

Fuentes, M., Chen, L., Davis, J. M., & Lackmann, G. M. (2005). Modeling and Predicting Complex Space-Time Structures and Patterns of Coastal Wind Fields. *Environmetrics*, *16*(5), 449–464.

Ghosh, S. K., Bhave, P. E., Davis, J. M., & Lee, H. (2010). Spatio-Temporal Analysis of Total Nitrate Concentrations using Dynamic Statistical Models. *Journal of the American Statistical Association*, *105*(490), 538–551.

Gneiting, T. (2002). Nonseparable, Stationary Covariance Functions for Space-Time Data. *Journal of the American Statistical Association*, *97*(458), 590–600.

Gneiting, T., Genton, M. G., & Guttorp, P. (2006). Geostatistical space-time models, stationarity, separability, and full symmetry. In B. Finkenstadt, L. Held, & V. Isham (Eds.), *Statistical Methods for Spatio-Temporal Systems* (pp. 151–175). Chapman & Hall.

Guinness, J., & Fuentes, M. (2015). Likelihood Approximations for Big Nonstationary Spatial-Temporal Lattice Data. *Statistica Sinica*, *25*, 329–349.

Guinness, J., & Stein, M. (2013). Interpolation of Nonstationary High Frequency Spatial-Temporal Temperature Data. *Annals of Applied Statistics*, *7*, 1684–1708.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning, 2nd Edition*. Springer.

Higdon, D. (1998). A Process-Convolution Approach to Modeling Temperatures in the North Atlantic Ocean. *Environmental Ecology Statistics*, *5*(2), 173–190.

Hooten, M. B., & Wikle, C. K. (2008). A hierarchical Bayesian non-linear spatio-temporal model for the spread of invasive species with appliation to the Eurasian Collared-Dove. *Environmental and Ecological Statistics*, *15*, 59–70.

Joseph, V. R. (2016). Space-filling designs for computer experiments: A review. *Quality Engineering*, *28*, 28–35.

Joseph, V. R., Gul, E., & Ba, S. (2015). Maximum Projection Designs for Computer Experiments. *Biometrika*, *102*, 371–380.

Katzfuss, M. (2017). A multi-resolution approximation for massive spatial datasets. *Journal of the American Statistical Association*, *112*(517), 201–214.

Katzfuss, M., Stroud, J. R., & Wikle, C. K. (2020). Ensemble Kalman methods for high-dimensional hierarchical dynamic space-time models. *Journal of the American Statistical Association*, *115*, 866–885.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, *30*, 3146–3154.

Krainski, E. T., Gomez-Rubio, V., Bakka, H., Lenzi, A., Castro-Camilo, D., Simpson, D., Lindgren, F., & Rue, H. (2019). *Advanced Spatial Modeling with Stochastic Partial Differential Equations Using R and INLA*. Chapman; Hall/CRC.

Lenzi, A., Castruccio, S., Rue, H., & Genton, M. G. (2019). Improving Bayesian Local Spatial Models in Large Data Sets. *arXiv:1907.06932*.

Lindgren, F., & Rue, H. (2011). An Explicit Link between Gaussian Fields and Gaussian Markov Random Fields: the Stochastic Partial Differntial Equation Approach. *Journal of the Royal Statistical Society: Series B*, *73*(4), 423–498.

Liu, X., Gopal, V., & Kalagnanam, J. (2018a). A Spatio-Temporal Modeling Framework for Weather Radar Image Data in Tropical Southeast Asia. *The Annals of Applied Statistics*, *12*(1), 378–407.

Liu, X., Gopal, V., & Kalagnanam, J. (2018b). A Spatio-Temporal Modeling Framework for Weather Radar Image Data in Tropical Southeast Asia. *The Annals of Applied Statistics*, *12*, 378–407.

Liu, X., & Pan, R. (2020). Analysis of Large Heterogeneous Repairable System Reliability Data with Static System Attributes and Dynamic Sensor Measurement in Big Data Environment. *Technometrics*, *62*, 206–222.

Liu, X., Yeo, K. M., Hwang, Y. D., Singh, J., & Kalagnanam, J. (2016). A Statistical Modeling Approach for Air Quality Data Based on Physical Dispersion Processes and Its Application to Ozone Modeling. *The Annals of Applied Statistics*, *10*(2), 756–785.

Liu, X., Yeo, K. M., & Kalagnanam, J. (2018c). A Statistical Modeling Approach for Spatio-Temporal Degradation Data. *Journal of Quality Technology*, *50*, 166–182.

Liu, X., Yeo, K. M., & Lu, S. Y. (2020). Statistical Modeling for Spatio-Temporal Data from Stochastic Convection-Diffusion Processes. *Journal of the American Statistical Association*, *to appear*, arXiv:1910.10375.

Mondal, D., & Wang, C. (2019). A matrix-free method for spatial-temporal Gaussian state-space models. *Statstica Sinica (to appear)*, *29*, 2205–2227.

Neuhaus, G. (1989). *Rank Tests with Estimated Scores and Their Application*. Verlag.

Nishio, M., Nishizawa, M., Sugiyama, O., Kojima, R., Yakami, M., Kuroda, T., & Togashi, K. (2018). Computer-aided diagnosis of lung nodule using gradient tree boosting and Bayesian optimization. *PLOS ONE*, *https://doi.org/10.1371/journal.pone.0195875*.

Nychka, D., & Wikle, J. A., C. Royle. (2002). Multiresolution Models for Nonstationary Spatial Covariance Functions. *Statistical Modeling*, *2*(4), 315–331.

Oguz, B. U., Shinohara, R. T., Yushkevich, P. A., & Oguz, I. (2017). Gradient Boosted Trees for Corrective Learning. *Machine Learning for Medical Imaging*, *10541*, 203–211.

Reich, B. J., Eidsvik, M., J. Guindani, Nail, A. J., & Schmidt, A. M. (2011). A Class of Covariate-Dependent Spatiotemporal Covariance Functions for the Analysis of Daily Ozone Concentration. *The Annals of Applied Statistics*, *5*(4), 2425–2447.

R-INLA. (2019). *The R-INLA Project, http://www.r-inla.org/*.

Rue, H., Martino, S., & Chopin, N. (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society, B*, *71*, 319–392.

Schabenberger, O., & Gotway, C. A. (2005). *Statistical Methods for Spatial Data Analysis*. Chapman & Hall/CRC.

Schapire, R. E. (1999). A Brief Introduction to Boosting. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*.

Sigrist, F. (2020). Gaussian Process Boosting. *arXiv:2004.02653v2*.

Sigrist, F., Kunsch, H. R., & Stahel, W. A. (2015). Stochastic Partial Differential Equation based Modelling of Large Space-Time Data Sets. *Journal of the Royal Statistical Society: Series B*, *77*(1), 3–33.

Stein, M. L., Chi, Z., & Welty, L. J. (2004). Approximating Likelihoods for Large Spatial Data Sets. *Journal of the Royal Statistical Society: Series B*, *66*(2), 275–296.

Stroud, J. R., Muller, P., & Sanso, B. (2001). Dynamic Models for Spatiotemporal Data. *Journal of the Royal Statistical Society: Series B*, *63*(4), 673–689.

Stroud, J. R., Stein, M. L., Lesht, B. M., Schwab, D. J., & Beletsky, D. (2010). An Ensemble Kalman Filter and Smoother for Satellite Data Assimilation. *Journal of American Statistical Association*, *105*, 978–990.

Wang, K., Jiang, W., & Li, B. (2016). A Spatial Variable Selection Method for Monitoring Product Surface. *International Journal of Production Research*, *54*, 4161–4181.

Wikle, C. K., & Cressie, N. (1999). A Dimension-Reduced Approach to Space-Time Kalman Filtering. *Biometrika*, *86*(2), 815–829.

Xie, Y., Jiang, B., Gong, E., Li, Y., Zhu, G., Michel, P., Wintermark, M., & Zaharchuk, G. (2019). Use of Gradient Boosting Machine Learning to Predict Patient Outcome in Acute Ischemic Stroke on the Basis of Imaging, Demographic, and Clinical Information. *American Journal of Roentgenology*, *212*, 44–51.

Yan, H., Zhao, X., Hu, Z., & Du, D. (2019). Physics-based Deep Spatio-temporal Metamodeling for Cardiac Electrical Conduction Simulation.

Yao, B., & Yang, H. (2021). Spatiotemporal Regularization for Inverse ECG Modeling. *IISE Transactions on Healthcare Systems Engineering*, *11*, 11–23.

Yao, B., Zhu, R., & Yang, H. (2017). Characterizing the location and extent of myocardial infarctions with inverse ECG modeling and spatiotemporal regularization. *IEEE Journal of Biomedical and Health Informatics*, *22*, 1445–1455.

Yue, X., Wen, Y., Hunt, J. H., & Shi, J. (2020). Active Learning for Gaussian Process considering Uncertainties, with an Application to Automatic Shape Control of Composite Fuselage. *arxiv: 2004.10931*.

Zang, Y., & Qiu, P. (2019). Phase I Monitoring of Spatial Surface Data from 3D Printing. *Technometrics*, *60*, 169–180.

2.6 Appendix

The following figures respectively show the MGE, RE and RMSE of the out-of-sample predictions for the 25 patients. Such visualizations provide a holistic view on the performance of the seven candidate methods. Note that, the proposed Boost-S yields (i) the lowest MGE for 19 out of the 25 patients, while RF performs the best for the remaining 6 patients; (ii) the lowest RE for 18 out of the 25 patients, while RF performs the best for the remaining 7 patients; and (iii) the lowest RMSE for 17 out of the 25 patients, while RF performs the best for the remaining 8 patients.

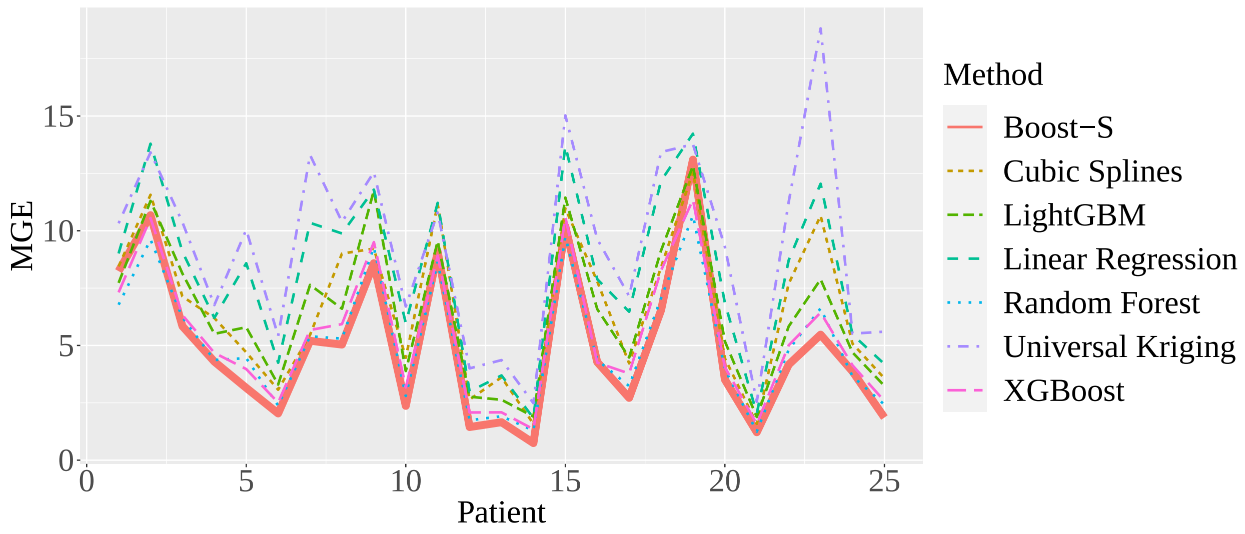


Figure 2.12: Mean Gross Error (MGE) of the out-of-sample predictions for 25 patients using 7 different methods

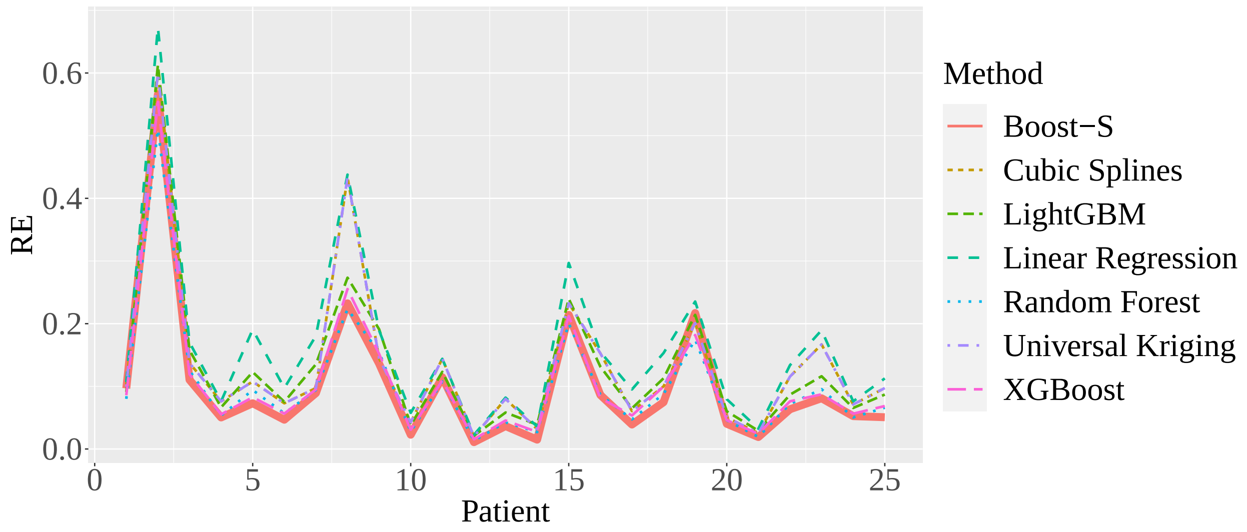


Figure 2.13: Relative Error (RE) of the out-of-sample predictions for 25 patients using 7 different methods

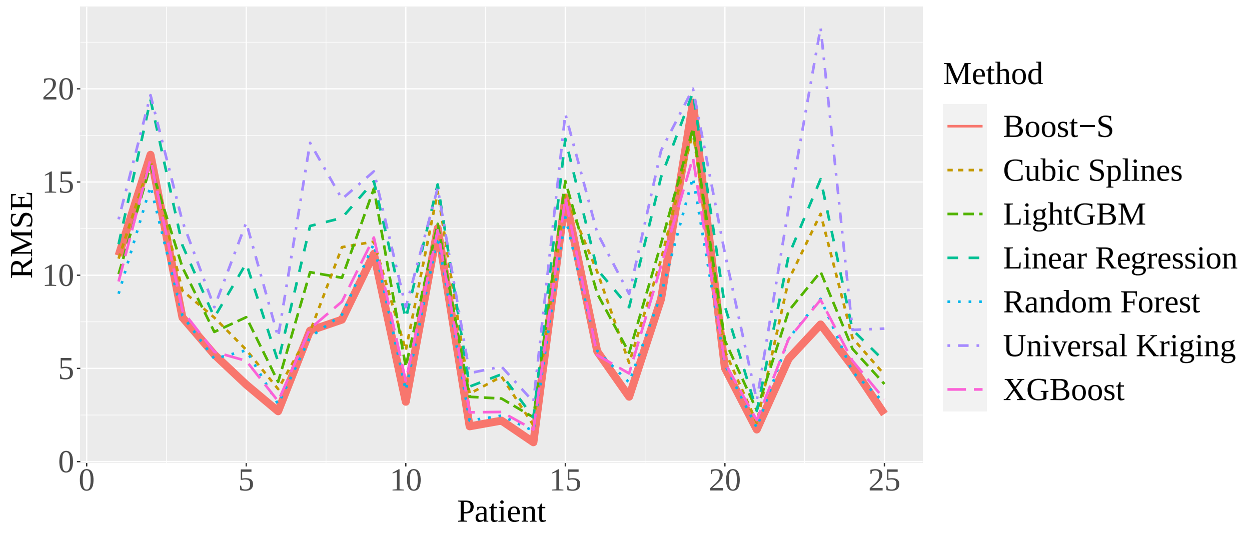


Figure 2.14: Rooted Mean Squared Error (RMSE) of the out-of-sample predictions for 25 patients using 7 different methods

3 Structured Adaptive Boosting Trees and Detection of Platelet-Neutrophil Aggregations due to E-Cigarette Exposure

3.1 Introduction

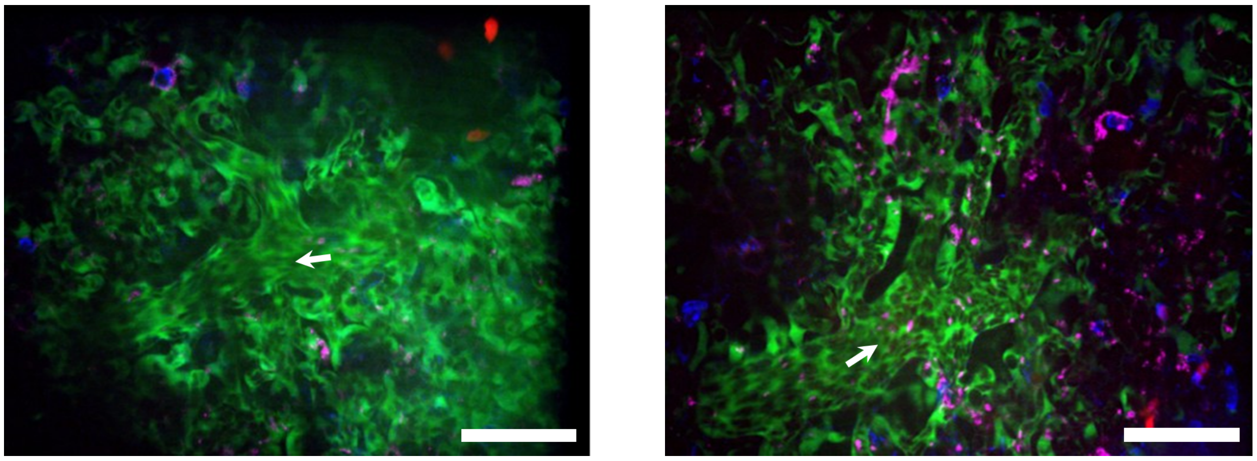
To surmount the edge detection problems associated with medical image analysis, this paper proposes a Structured Adaptive Boosting Trees algorithm (AdaBoost.S) for fast detection of dynamic multicellular aggregates present in fluorescence intravital microscopy (IVM) image data. The algorithm is primarily motivated by the well-known observation that edges over an image mask often exhibit special structures and are highly interdependent. Such structures can be exploited and predicted, by statistical learning methods, using features extracted from bigger image patches that cover the image masks. This paper presents (1) how features can be extracted from image patches, (2) how extracted features can be used to predict the edge structures using the proposed AdaBoost.S, and (3) how the proposed approach can be applied to detect vascular lung injury on fluorescence intravital microscopy (IVM) data sets of mice exposed to e-cigarette vapor.

3.1.1 Motivating Application: Detection of Multicellular Aggregates in Fluorescence Intravital Microscopy

IVM can be utilized to track interactions between multiple cell types over time in several organs throughout the body including but are not limited to the lungs, liver, and brain (Bennewitz et al., 2017; Kolaczkowska et al., 2015; Rapoport et al., 2015; Secklehner et al., 2017; Weigert et al., 2010). To enable visualization of dynamic cell phenomena, the desired tissue is first gently immobilized against a glass coverslip. The vasculature and cell populations of interest are highlighted by injection of fluorescent dyes or antibodies, or through genetic engineering of fluorescent proteins. In one imaging session, gigabytes of dynamic data are generated which presents a challenge to manual analysis of cell interactions. To meet this need, microscope manufacturers have designed analytical software to detect the individual labeled cells, but these platforms are met with limitations including high cost. Additionally, processing requires human input to create a recipe that must be manually tweaked for every video, as fluorescent intensity varies for every mouse. Multicellular interactions cannot be quantified longitudinally and fluorescent artifacts cannot be excluded. Furthermore, lung movement is not corrected, which causes stationary cells to appear mobile. Finally, there is no method to determine cell types within aggregates or the size of the blocked vessel. In contrast, machine learning has the capability for smart detection and longitudinal analysis of the multicellular interactions observed in IVM.

In this paper, we focus on blood cell interactions in the lungs of live mice exposed to e-cigarette vapor. Currently, e-cigarettes have received much media attention due to the recent surge in teen vaping of 135%, combined with more than 60 fatalities and more than 2700 hospitalizations associated with e–cigarette use (Spectrum Health, 2022). While many believe that e-cigarettes are less harmful than traditional cigarettes, the possible invisible damage of vaping has not yet been fully elucidated. One of our goals with machine learning is to characterize the subclinical pathologies resulting from e-cigarette vapor exposure to better predict the risks that accompany chronic use. Specifically, the role of blood cells including neutrophils and platelets in lung injury needs to be evaluated. Neutrophils are the most abundant type of white blood cells and can release a network of DNA fibers called neutrophil extracellular traps (NETs) as a host defense mechanism (Brinkmann et al., 2004). NETs can cause downstream injury through enabling inflammation, vessel damage, and clot formation (Demers et al., 2012; Saffarzadeh et al., 2012). Platelets, which are known to activate clotting, can adhere to neutrophils to form platelet-neutrophil aggregates that can block blood vessels and prevent gas exchange (Etulain et al., 2015; Ma & Kubes, 2008).

For illustrative purposes, Figure 3.1 shows two fluorescence IVM images of the pulmonary microcirculation of a healthy mouse (left), and a mouse exposed to e-cigarette vapor for three hours per day over three days (right). These images are obtained from a preclinical study conducted by the co-authors' team. Blood vessels (green), platelets (pink), neutrophils (blue), and NETs (red) were visualized in lung blood vessels after injection of FITC dextran, AlexaFluor 647 CD49b, Pacific Blue Ly6G, and SYTOX orange, respectively. By comparing the two IVM images in Figure 3.1, the formation of platelet-neutrophil aggregates (the total area of objects containing



(a) Healthy Mouse (b) Exposed Mouse

Figure 3.1: Fluorescence IVM images, obtained from a preclinical study conducted by the co-authors' team, of the pulmonary microcirculation of (left) a healthy mouse and (right) a mouse exposed to e-cigarette vapor. Blood vessels are shown in green, platelets in pink, neutrophils in blue and NETs in red. Arrows denote the direction of blood flow. Scale bar: 50μm.

colocalized neutrophil (blue) and platelet (purple) staining) are clearly observed from the mouse exposed to e-cigarettes vapor, indicating the potential for decreased blood flow. Hence, given a large number of fluorescence IVM images, a statistical learning algorithm is needed to efficiently and accurately detect the edges of platelet-neutrophil aggregates, which serves as an essential step toward understanding the effects of e-cigarettes and other pathologies driven by neutrophil-platelet interactions.

3.1.2 Literature Review and Contributions

An edge in an image can be defined as an abrupt change of color intensity between two neighboring pixels (Ziou, Tabbone, et al., 1998). Edge detection is a process that detects the presence and location of edges formed by sharp changes in the image's color intensity (Ramana et al., 2017). Over the past decades, edge detection has remained as a fundamental yet challenging problem in image analysis, computer vision and statistical learning (Goodfellow et al., 2016; Xie & Tu, 2015).

Early work has focused on detecting significant intensity changes or gradient magnitudes using convolutional masks on images such as Canny, Sobel, and Prewitt (Senthilkumaran & Rajesh, 2009). These methods are fast but do not have the desired outcomes in most cases because they are not capable of capturing texture edges and are too sensitive to noise. To better capture texture edges and improve the robustness of the algorithms against noise, the state-of-the-art edge detection approaches employ various feature engineering techniques to extract information for making meaningful differences between pixels. Then, machine learning techniques can be used to learn the mapping between edges and extracted features (Bansal et al., 2022; Dollar et al., 2006; Goodfellow et al., 2016; Lim et al., 2013; Singh & Sharma, 2021; Zheng et al., 2010). These approaches predict whether a pixel contains an edge by considering the features extracted from surrounding pixels, i.e., an image patch.

A critical observation is that edges over an image mask often exhibit special structures and are highly interdependent (see Figure 3.7 in Section 3.3 for some examples). Nowozin and Lampert, 2011 and Dollár and Zitnick, 2013 investigated the interdependency among local patches and introduced a family of novel methods called *Structured Learning* (SL). The key idea behind SL is to learn the structures of a group of pixels, called masks, rather than computing if individual pixels contain edges. Dollár and Zitnick, 2013 proposed an important ensemble tree-based algorithm for fast edge detection—the Structured Random Forest (Structured RF). Such an algorithm grows an ensemble of decision trees that robustly map the structured labels to a discrete space in which the standard framework of Random Forest can be utilized (Hastie et al., 2009). It has been shown that this approach can well leverage the inherent structure in edge patches and is computationally efficient for real-time image edge detection.

Following the idea of Structured RF, this paper further makes the following contributions: it proposes a new Structured Adaptive Boosting Trees algorithm (AdaBoost.S) for edge detection problems. This work is naturally driven by the quest that, if the framework of Random Forest can be adapted for structured learning, then, the framework of boosting trees should also have the potential to be adapted for structured learning. Because Random Forest and boosting trees are main competitors of each other, which are based on opposite ideas (fully-grown de-correlated trees in parallel v.s. simple correlated trees in series), this paper fills a gap in the literature where the structured boosting trees are not yet available. We describe in detail how features can be extracted from image patches (such as color channels, gradient magnitude features, and orientation-based features at multiple scales using Gabor wavelets), and how extracted features are used to predict the edge structure over an image mask inside the image patch (see Figure 3.2). The AdaBoost.S algorithm is applied to detect the platelet-neutrophil aggregates within lung blood vessels visualized on IVM images, which facilitates our understanding of the potential damaging effects of e-cigarette vapor exposure.

The rest of this paper is outlined as follows: Section 3.2 presents the feature extraction process and the technical details of the proposed AdaBoost.S. The application of AdaBoost.S is presented in Section 3.3, and Section 3.4 concludes the paper.

3.2 Structured Adaptive Boosting Trees

3.2.1 Basic Terminologies

We first define the key terminologies used in this paper:

An Image in Grayscale. An image is a matrix of pixels (picture elements) arranged in columns and rows. For an image in grayscale, the value of each pixel represents only the intensity information. In this paper, we let $I(s_1, s_2)$ represent the intensity of a pixel, where $s_1 \in \mathcal{S}_1$ and $s_2 \in \mathcal{S}_2$ are the horizontal and vertical coordinates of the pixel, respectively.

Segmentation Mask. Image segmentation partitions an image into two segments (objects of interest and the background). The segmented binary image is called the segmentation mask of the original image. Image segmentation can be performed using the clustering algorithms such as K-means (Dhanachandra et al., 2015), region-based Segmentation (Kohler, 1981), Mask R-CNN (Huang et al., 2019), etc. These algorithms create the so-called black/white image, known as binarization, where 0 and 1 indicate whether a pixel belongs to the background or an object. Image segmentation is an essential process that reduces noise and defines the objects of interest; for example, the platelets-neutrophils aggregates.

Image Patch. An image patch is a pixel array that has a predetermined dimension.

Edge Mask. An edge mask is a pixel array in which the value of each element of this pixel array is an edge label that indicates if this pixel belongs to an edge.

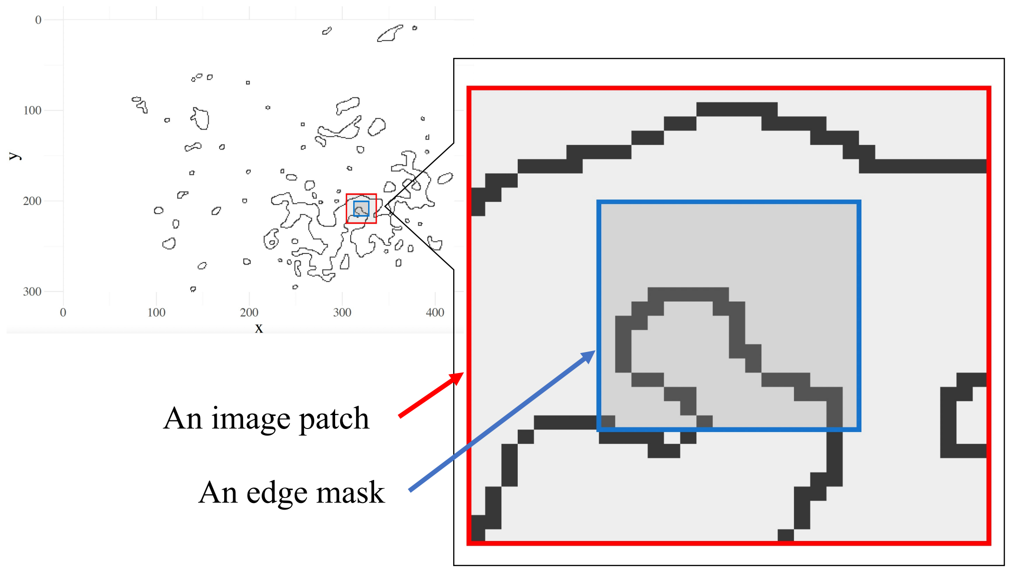


Figure 3.2: (left) an IVM image; (right) a zoomed-in view of an image patch (outer square) and an edge mask (inner square) taken from the IVM image.

Figure 3.2 provides an example of an image patch and an edge mask (inside the patch) taken from an IVM image. In edge detection problems, using the features extracted from the surrounding image patch, the values on an edge mask are predicted.

3.2.2 Feature Extraction

In this paper, the goal is to predict the values on an edge mask using features extracted from the surrounding image patch. Feature extraction starts with creating the segmentation mask, which distinguishes objects of interest from the background. In the motivating example, the objects of interest are the platelet-neutrophil aggregates. The Gaussian low-pass blur filters are firstly employed to enhance image structures at three different scales (including the original scale). Then, the gradient magnitude features are extracted, and each gradient magnitude feature is further split into four orientation-based features by the Gabor feature extraction technique. Each step of feature extraction is explained in further detail below.

Gaussian blur is a low-pass filter that reduces the level of noise in the image, which improves the result of the edge-detection algorithms (Deriche, 1992; Van Vliet et al., 1998). When applying the Gaussian blur filter, the image is convolved with an isotropic kernel Gaussian function:

$$\begin{aligned} I^{\text{GF}}(s_1, s_2; \sigma) &= H^{\text{Gaussian}} * I(s_1, s_2) \\ &\equiv \int_{\mathcal{S}_1} \int_{\mathcal{S}_2} H^{\text{Gaussian}}(u, v) I(s_1 - u, s_2 - v) du dv \end{aligned} \tag{3.1}$$

where $*$ represents the convolution operation, and H^{Gaussian} is a two-dimensional Gaussian kernel:

$$H^{\text{Gaussian}}(u, v; \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{u^2 + v^2}{2\sigma^2}\right) \tag{3.2}$$

where σ is the patch radius (standard deviation). The choice of σ controls the blurring scales: the higher the σ, the lower the image resolution. In this paper, three different scales are used, $\sigma_1 = 0$ (the original scale), $\sigma_2 = 1$, and $\sigma_3 = 2$, and we obtain three low-pass filtered images $I^{\text{GF}}(s_1, s_2; \sigma_1)$, $I^{\text{GF}}(s_1, s_2; \sigma_2)$, and $I^{\text{GF}}(s_1, s_2; \sigma_3)$.

Based on the filtered images, gradient magnitudes can be computed:

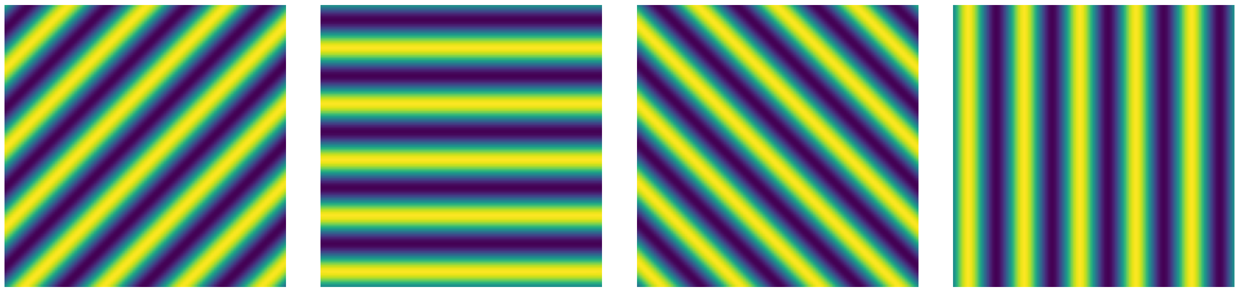
$$|\nabla I(s_1, s_2; \sigma)| = \sqrt{g_{s_1}^2 + g_{s_2}^2} \tag{3.3}$$

where ∇ is the gradient operator, and

$$g_{s_1} = \frac{\partial I(s_1, s_2; \sigma)}{\partial s_1}, \quad g_{s_2} = \frac{\partial I(s_1, s_2; \sigma)}{\partial s_2}. \tag{3.4}$$

Hence, the gradient magnitudes create three (feature) channels corresponding to the three scales of σ of the Gaussian kernel, and we denote these three channels by $C_1(s_1, s_2)$, $C_2(s_1, s_2)$, and $C_3(s_1, s_2)$.

Next, the Gabor filter is applied to each channel $C_i(s_1, s_2)$, $i = 1, 2, 3$, to create additional feature channels. The Gabor filter is a linear filter used for analyzing whether there is any specific frequency content in the image along specific directions within a localized region around the point



(a) $\theta = \frac{\pi}{4}$ (b) $\theta = \frac{\pi}{2}$ (c) $\theta = \frac{3\pi}{4}$ (d) $\theta = \pi$

Figure 3.3: Gabor filter bank at orientations $\frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}$, and π

or region of analysis. A two-dimensional Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave:

$$H^{\text{Gabor}}(u, v; \tau, \theta, \lambda, \gamma, \phi) = \exp\left(-\frac{u'^2 + \lambda^2 v'^2}{2\tau^2}\right) \exp\left(i\left(2\pi\frac{u'}{\lambda} + \phi\right)\right) \tag{3.5}$$

where $u' = u\cos\theta + v\sin\theta$ and $v' = -u\sin\theta + v\cos\theta$, τ represents the standard deviation, θ indicates the angle, λ represents the wavelength, γ is the aspect ratio, ϕ controls the offset, and i is an imaginary number. As an illustration, Figure 3.3 shows the Gabor filter at orientations $\pi/4$, $\pi/2$, $3\pi/4$, and π.

Applying the four Gabor filters (with orientations $\pi/4$, $\pi/2$, $3\pi/4$, and π) to each of the gradient magnitude channels, $C_1(s_1, s_2)$, $C_2(s_1, s_2)$, $C_3(s_1, s_2)$, creates another 12 (feature) channels. In particular, we let $C_{i,j}(s_1, s_2)$ represent the channel created by applying the j-th Gabor filter to the i-th gradient magnitude channel. Table 3.1 provides a summary of the 16 features extracted for the edge detection algorithm to be described in the next section. The 16 features channels include the 3 gradient magnitude channels (i.e., $C_1(s_1, s_2)$, $C_2(s_1, s_2)$, and $C_3(s_1, s_2)$), the 12 orientation-based channels created by the Gabor filter (i.e., $C_{i,j}$ for $i = 1, 2, 3$ and $j = 1, 2, 3, 4$), and the original grayscale image (i.e., $I(s_1, s_2)$).

Table 3.1: A summary of the 16 features extracted for the edge detection algorithm

Feature	Notation	Number of channels
Grayscale intensity	$I(s_1, s_2)$	1
Gradient magnitude	$C_1(s_1, s_2)$, $C_2(s_1, s_2)$, and $C_3(s_1, s_2)$	3
Orientation-based features	$C_{i,j}$, $i = 1, 2, 3$ and $j = 1, 2, 3, 4$	12

3.2.3 Structured Learning using AdaBoost.S

We first present the data structure used to train the proposed structured adaptive boosting trees. The extracted features are used to generate the feature matrix. From the images of the training dataset, suppose that N image patches can be obtained and each patch is an $n \times n$ pixel array. For patch i, we can obtain the n^2 pixel values from each of the 16 feature channels and arrange the pixel values into a $1 \times n^2$ row vector. The 16 row vectors are combined to form a long $1 \times 16n^2$ feature vector for patch i (see Figure 3.4). In total, there are N such rows in the feature matrix which has a dimension of $N \times 16n^2$.

Associated with each patch i (i.e., each row of the feature matrix), there exists an edge mask which is an $m \times m$ pixel array. Each pixel on the edge mask takes a value of either 1 or 0, where 1 indicates that the pixel belongs to an edge. Figure 3.5 provides an illustrative example of two distinct edge structures on two 4×4 edge masks. If we vectorize the $m \times m$ pixel arrays by columns, we obtain a binary response vector $y_i = (y_{i,1}, y_{i,2}, \cdots, y_{i,m^2})$ for each edge mask, $i = 1, 2, \cdots, N$.

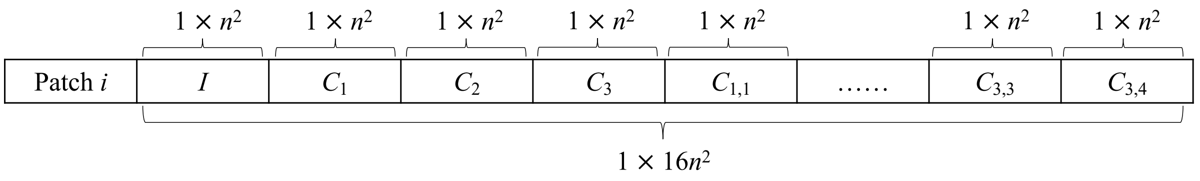


Figure 3.4: Illustration of the i-th row of the feature matrix

It is important to note that, in structured regression trees, each response vector needs to effectively represent a special edge structure. However, simple vectorization of the edge mask

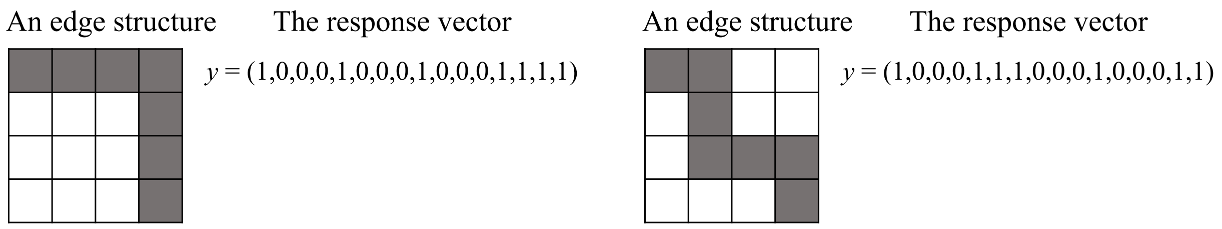

An edge structure
The response vector
y = (1,0,0,0,1,0,0,0,1,0,0,0,1,1,1,1)
An edge structure
The response vector
y = (1,0,0,0,1,1,1,0,0,0,1,0,0,0,1,1)

Figure 3.5: Each response vector represents a special edge structure.

(e.g., the y vectors shown in Figure 3.5) does not allow us to quantify the similarity between two edge structures by computing the Euclidean distance between two vectors. For example, consider three different edge structures on a 3×3 edge mask and their corresponding y vectors, $y_1 = (1,1,1,0,0,0,0,0,0)$, $y_2 = (1,1,0,0,0,0,0,0,0)$ and $y_3 = (1,0,1,0,0,0,0,0,0)$. Although the Euclidean distance between y_1 and y_2 is the same as the distance between y_1 and y_3, the first and second edge structures are clearly more similar to each other.

To tackle this challenge, Dollár and Zitnick, 2013 advocated to create a long one-dimensional vector $z_i \in \mathcal{Z}$ from $y_i \in \mathcal{Y}$ so as to represent the edge structure over an edge mask i through a mapping:

$$\Pi : \mathcal{Y} \rightarrow \mathcal{Z}. \tag{3.6}$$

The length of z_i is $\binom{m^2}{2}$ and each element of z_i corresponds to a pair of pixels within y_i. In particular, for any pair $y_{i,j}$ and $y_{i,j'}$, the corresponding element in z_i is set to 1 if $y_{i,j} \neq y_{i,j'}$; otherwise, the corresponding element in z_i is set to 0. Hence, a vector z_i, $i = 1, 2, \cdots, N$, is a binary vector that uniquely represents an edge structure over edge mask i. It is now possible to define the dissimilarity between two edge structures by computing the Euclidean distance of two z vectors. When m is large, Principal Component Analysis (PCA) can be used to reduce the dimension of the z vectors.

Figure 3.6 illustrates the structure of the data used to train the structured boosting trees. The feature matrix is extracted from the image patches. The response vectors (i.e., the z vectors) are obtained from the vectorized edge masks (i.e., the y vectors).

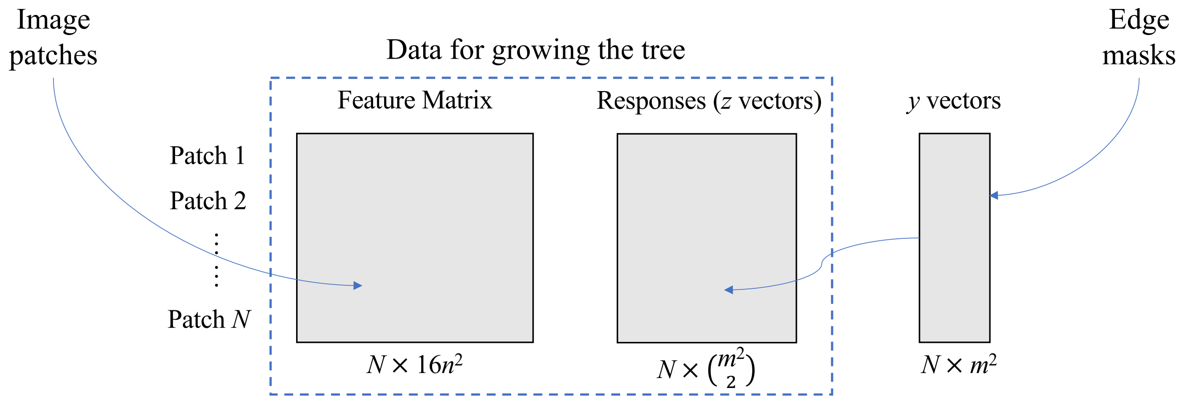


Figure 3.6: Illustration of the structure of the dataset used to train the structured boosting trees algorithm.

Based on the prepared training data, AdaBoost.S involves growing an ensemble of trees. It is important to note that, an individual tree here is *not* a regular multivariate classification tree. In this paper, because each z vector represents an edge structure in the training dataset, it is essential to predict the entire response vector (i.e., the edge structure) all at once.

To tackle this challenge, Dollár and Zitnick, 2013 proposed to first perform a clustering analysis of the response vectors on a tree node before this tree node can be split. The clustering analysis effectively finds similar edge structures by grouping the response vectors (the z vectors) into a small number of classes (i.e., each vector is assigned a discrete label). After this step, the tree node splitting can be carried out as if we were growing a classification tree. Once the two daughter nodes are found, the original response vectors are recovered, and the steps above are repeated on each daughter node to complete the tree.

To elaborate, like how a classification tree is typically grown, we utilize the framework of greedy algorithm (Hastie et al., 2009). At each tree node, we find the optimal splitting variable j and split point l that define the left and right daughter nodes:

$$R_L(j,l) = \left\{x | x^{(j)} \leq l\right\}, \qquad R_R(j,l) = \left\{x | x^{(j)} > l\right\} \tag{3.7}$$

where $x^{(j)}$ is the j-th feature of the feature vector x.

Before splitting a tree node , the feature vectors on the tree node are first clustered into K classes using existing clustering algorithms (e.g., K-means, hierarchical clustering, etc.). The clustering analysis defines a mapping $\mathcal{C}$ that assigns each z_i on the tree node with a label c_i

$$\mathcal{C} : z \mapsto c \in [1, 2, \cdots, K]. \tag{3.8}$$

After this step, the optimal splitting variable j and split point l are found by minimizing

$$\min_{j,l} \left\{ \sum_{x_i^{(j)} \in R_L(j,l)} \delta(c_i, \hat{c}_L) + \sum_{x_i^{(j)} \in R_R(j,l)} \delta(c_i, \hat{c}_R) \right\} \tag{3.9}$$

where $\hat{c}_L$ and $\hat{c}_R$ are the predicted labels on the left and right daughter nodes which can be determined by the majority class on a node, and $\delta(\cdot, \cdot)$ can be any commonly used measures such as the misclassification error, Gini index, entropy, and so on.

When a tree is fully grown, the predicted label $\hat{c}$ is converted back to the z vector through the inverse mapping $\mathcal{C}^{-1}$. In particular, suppose a tree divides the feature space into D regions, $R_1, R_2, \cdots, R_D$, and the predicted structure vector over each region is denoted by $\hat{z}^{(m)}$, then, the output of the tree is given by

$$f(x) = \sum_{m=1}^{D} \hat{z}^{(d)} I_{\{x \in R_d\}} \tag{3.10}$$

where $I_{\{x \in R_d\}} = 1$ if $x \in R_d$, otherwise $I_{\{x \in R_d\}} = 0$.

However, it is immediately noted that there often exist multiple distinct z vectors associated with the same label c on a terminal node. Hence, the medoid of the z vectors on a terminal node d is used as the predicted $\hat{z}^{(d)}$ on that node:

$$\hat{z} = \text{argmin}_k \sum_{i,j} (z_{k,j} - z_{i,j})^2 \tag{3.11}$$

which is the vector that has the minimum average distance to all other vectors.

Once it is possible to construct a single tree, we embed the structured trees into the classical framework of AdaBoost.M1 (Freund & Schapire, 1997), and obtain the following AdaBoost.S algorithm:

Initialization. All edge masks in the data set are given equal weights, $\omega_i^{(0)} = N^{-1}$, $i = 1, 2, \cdots, N$.

Grow Structured trees. Grow M structured trees sequentially, $\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_T$. For $t = 1, 2, \cdots, T$, this involves:

• Grow a structured tree $\mathcal{T}_t$ based on the training data with weights $\omega_i^{(t-1)}$. Note that, the weighted medoid of the z vectors on a terminal node m is used as the predicted $\hat{z}^{(m)}$ on that node. In particular, let $\mathcal{I}_d$ be a set that contains the image masks on terminal node d. Then, we have

$$\hat{z} = \text{argmin}_k \frac{1}{\omega_k^{(t-1)}} \sum_{i \in \mathcal{I}_d, j=1,\cdots,\binom{m^2}{2}} (z_{k,j} - z_{i,j})^2, \quad k \in \mathcal{I}_d \tag{3.12}$$

which is the vector that has the minimum weighted average distance to all other vectors.

• Compute the modeling error of $\mathcal{T}_t$ by

$$\text{err}_t = \frac{\sum_i^n \omega_i^{(t-1)} I(y_i \neq \mathcal{T}_t(x_i))}{\sum_i^n \omega_i^{(t-1)}} \tag{3.13}$$

where $I = 1$ if $y_i \neq \mathcal{T}_t(x_i)$, and $I = 0$ otherwise.

• Update the weights

$$\omega_i^{(t)} \leftarrow \omega_i^{(t-1)} \exp\{\alpha_t I(y_i \neq \mathcal{T}_t(x_i))\}, \quad i = 1, 2, \cdots, N \tag{3.14}$$

where $\alpha_t = \log((1 - \text{err}_t)/\text{err}_t)$. Note that, it is necessary to make sure that $\alpha_t > 0$, or equivalently $\text{err}_t < 0.5$. This can be achieved by controlling the complexity (i.e., tree depth and number of terminal nodes) of individual trees.

Ensemble output. Finally, for any image mask i, let $\hat{z}_i^{(t)}$ be its predicted value from tree t for

$t = 1, 2, \cdots, T$. Then, the ensemble predictor of mask i is given by the weighted medoid of $\hat{z}_i^{(t)}$

$$\hat{z}_i^{\text{ensemble}} = \mathcal{T}(x_i) = \text{argmin}_k \frac{1}{\alpha_k} \sum_t (\hat{z}_i^{(k)} - \hat{z}_i^{(t)})^2. \tag{3.15}$$

3.3 Application: AdaBoost.S for Platelet-Neutrophil Aggregates Detection

This section revisits the example of fluorescence IVM images showing lung injury in mice exposed to e-cigarette vapor in Section 3.1.1. Herein, we demonstrate the performance of the proposed AdaBoost.S to identify platelet-neutrophil aggregates within this real dataset.

The data used in this section are obtained from mice undergoing e-cigarette vapor exposure in the preclinical study conducted by the authors' team. In the experiment, 3 mice are put into one group and there are a total of 4 groups with different exposure paradigms. At least 15 videos are taken of the lung in each region for each group. In this numerical example, we randomly extract 47 images (i.e., snapshots) from the videos and each image has a dimension of 420×300 pixels. Image patches are extracted from the images. Each image patch has a dimension of 8×8 pixels, and the edge mask within an image patch has a dimension of 4×4 pixels. Figure 3.7 shows some examples of edge structures on selected edge masks. To demonstrate the proposed algorithm, we focus on the edges of the overlay of neutrophil and platelet. In total, there are 1809 unique structures in our dataset. Based on the image patches and the feature extraction procedure described in Section 3.2.2, $16 \times 8^2 = 1024$ features are extracted. We split the data according to a 55/45 ratio to create the training and testing datasets, although other ratios may be used. In this case, 26 images are used to grow the tree, while the remaining 21 images are used to validate the algorithm.

It is also important to note that, the images used in this example are extremely sparse, meaning that most of the image patches are empty (which do not contain any edges). In our dataset, 91.5% of masks are empty and contain no structural information. Such data imbalance is known to create a significant challenge for classification problems. To overcome this issue, we first use the conventional AdaBoost to classify the image patches into two categories: the patches that contain edges and the patches that do not contain any edges. Then, we apply the AdaBoost.S algorithm only to those image patches that contain edges. This hybrid procedure is illustrated in Figure 3.8. Hence, given a new testing image, the trained AdaBoost model first identifies those non-empty image patches. After that, the trained Adaboost.S model predicts the edge masks for those non-empty image patches.

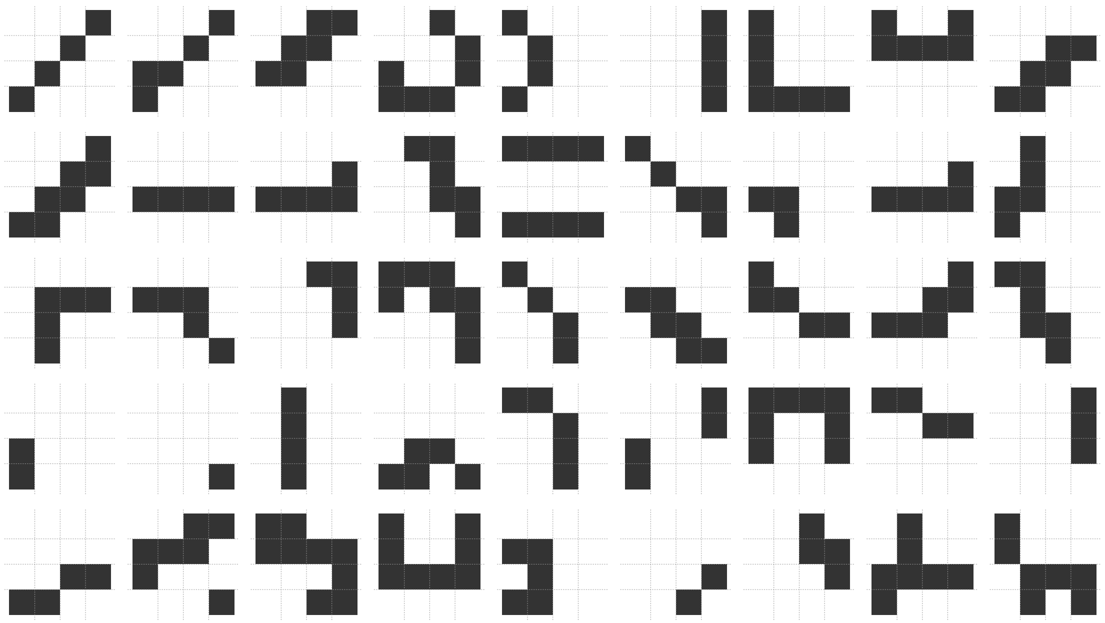


Figure 3.7: Examples of selected edge structures of neutrophil-platelet aggregates.

Following the procedure outlined in Figure 3.8, we train the AdaBoost.S ensemble trees using the training dataset and test the model using the testing dataset. To demonstrate the performance of AdaBoost.S, Figures 3.9, 3.10, and 3.11 present three examples that compare the predicted edges with the labelled edges. To label the edges, the original images are firstly denoised and the pixels that belong to the edges of the overlay of neutrophil and platelet are carefully identified. It is immediately seen that the proposed AdaBoost.S is capable of detecting different edge structures presented in these images.

Tables 3.2, 3.3 and 3.4 present the F-score, precision, and recall of the proposed AdaBoost.S based on the testing dataset. Precision indicates the ratio between the number of correctly predicted edge pixels and the total number of predicted edge pixels; Recall indicates the ratio between the number of correctly predicted edge pixels and the number of actual edge pixels; and the F-score is the harmonic mean of precision and recall. For all three metrics, their values lie between 0

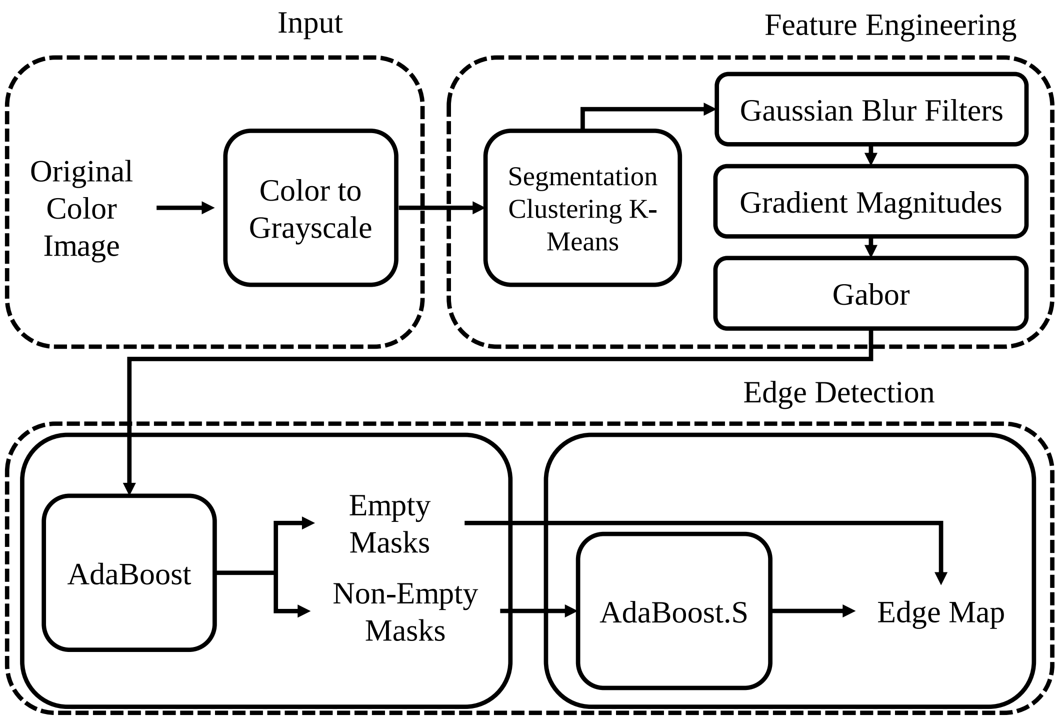
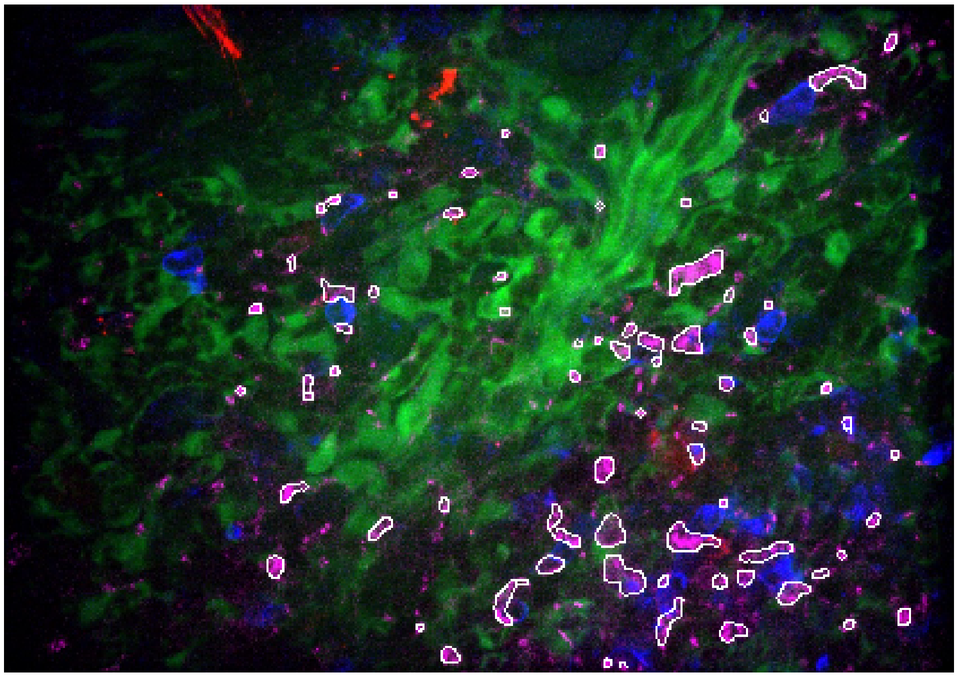


Figure 3.8: The hybrid procedure adopted for the edge detection problem in the numerical example: AdaBoost first identifies those non-empty image patches, and Adaboost.S then predicts the edge masks for those non-empty image patches.

and 1, and a higher value indicates better model performance. In addition, we also include the comparison between the proposed AdaBoost.S and other tree-based methods including Random Forest, XGBoost, and classification trees. It is seen that the proposed AdaBoost.S has the highest recall and F-score on all testing cases. Although Random Forest yields the highest precision, it yields a lower recall. Further investigation reveals that Random Forest generates fewer false edge points so that a higher precision is achieved, while AdaBoost.S predicts far less non-edge pixels, which distinguishes our algorithm from its competitors. When comparing the F-score, which is the harmonic mean of precision and recall, the proposed AdaBoost.S significantly outperforms the other methods. Note that, deep learning methods, such as the Convolutional Neural Networks (CNN) (Goodfellow et al., 2016), are not included in the comparison because of the relatively small training sample size in this numerical example (26 images are used to train the AdaBoost.S trees). Hence, this numerical example demonstrates the performance of the proposed method for problems with small and moderate data sizes, while adapting a deep learning method such as CNN for structured learning can be a very important topic once we have more data from our experiments.



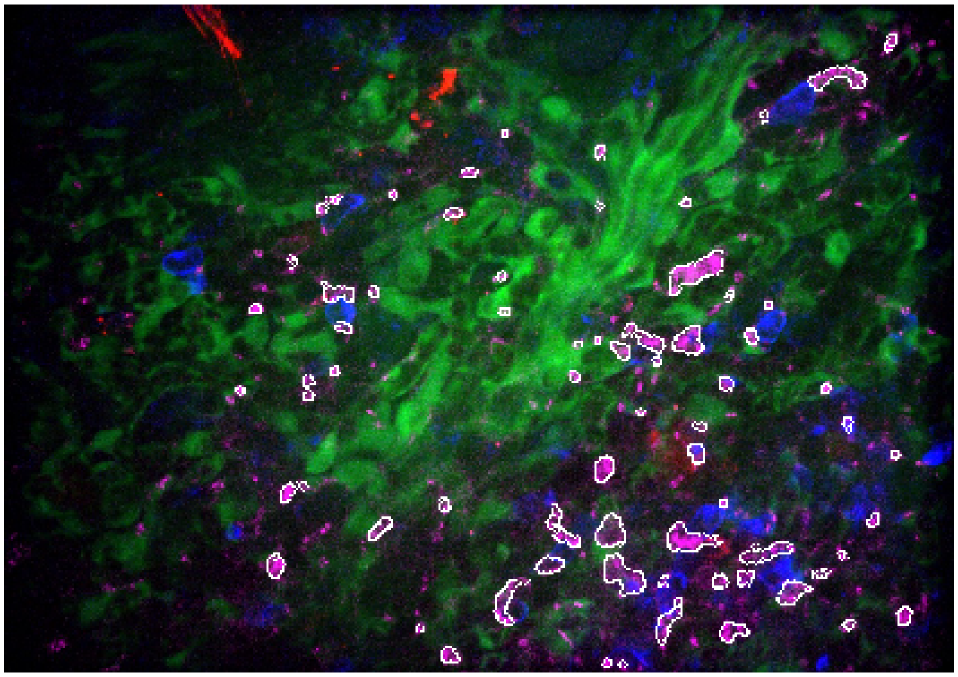
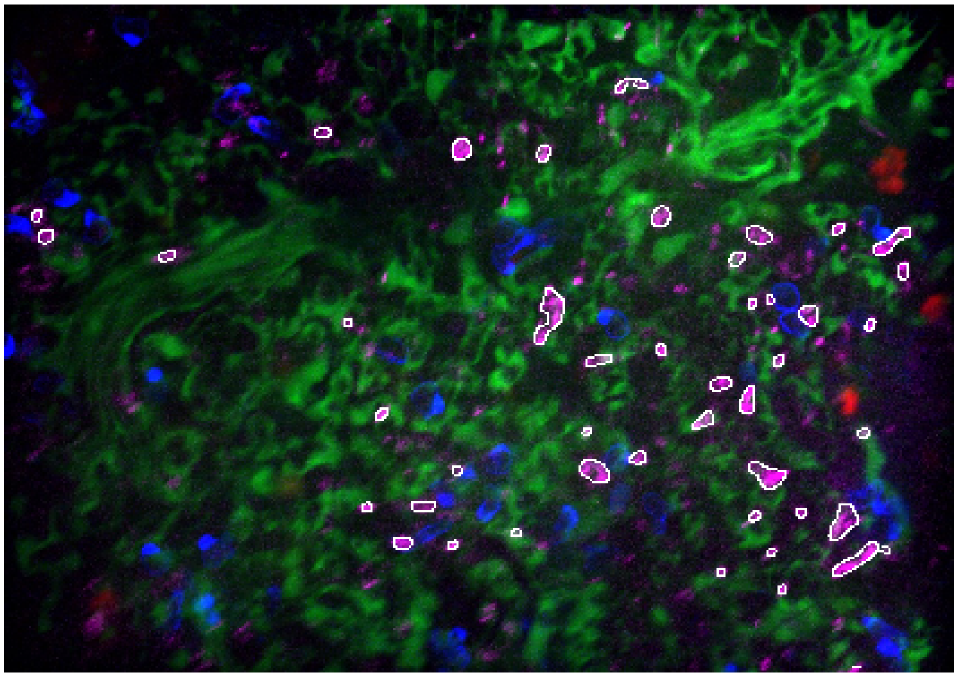


Figure 3.9: Illustration of the AdaBoost.S responses in the first selected field of view (FOV). The top figure shows the actual fluorescence IVM image with the labelled edge map of neutrophil-platelet aggregates, and the bottom figure shows the predicted edges of neutrophil-platelet aggregates.



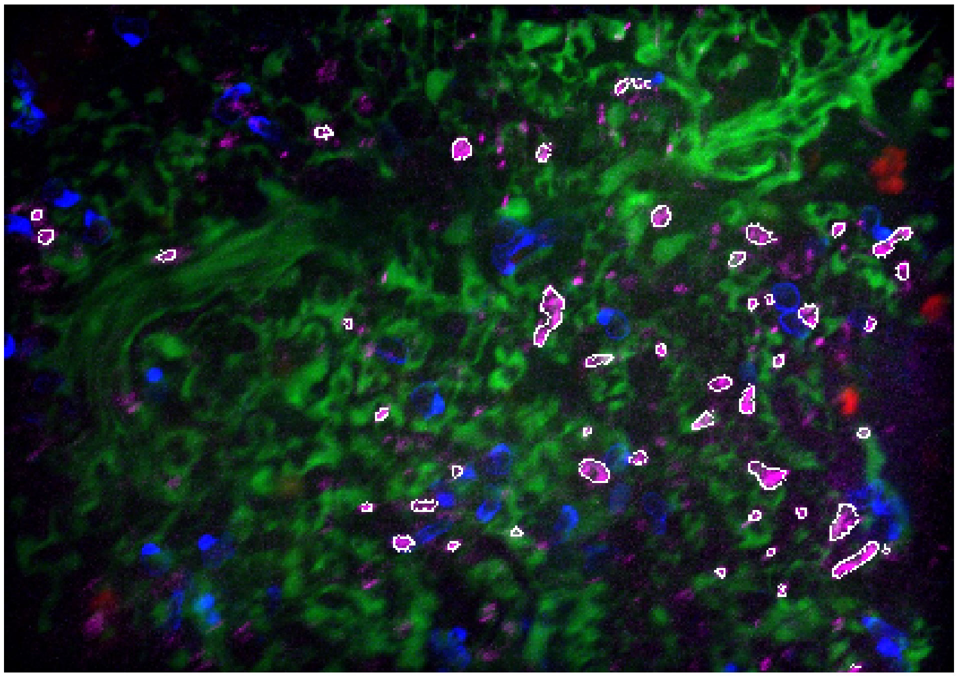
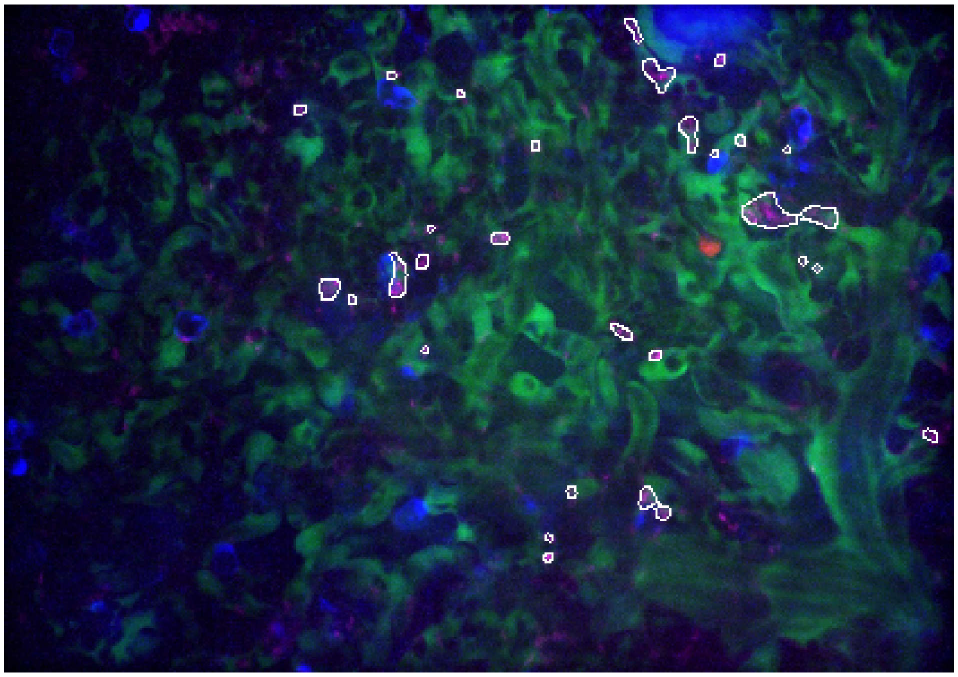


Figure 3.10: Illustration of the AdaBoost.S responses in the second selected FOV. The top figure shows the actual fluorescence IVM image with the labelled edge map of neutrophil-platelet aggregates, and the bottom figure shows the predicted edges of neutrophil-platelet aggregates.



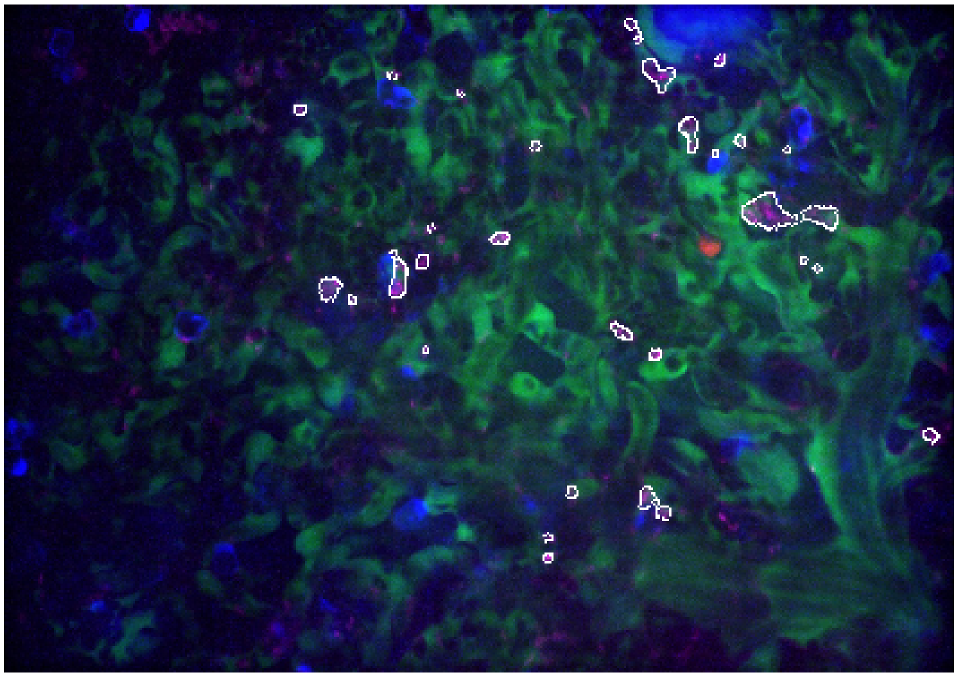


Figure 3.11: Illustration of the AdaBoost.S responses in the third selected FOV. The top figure shows the actual fluorescence IVM image with the labelled edge map of neutrophil-platelet aggregates, and the bottom figure shows the predicted edges of neutrophil-platelet aggregates.

Table 3.2: F-score of the out-of-sample predictions

Image	AdaBoost.S	Random Forest	XGBoost	Decision Tree
1	**0.82**	0.65	0.64	0.11
2	**0.84**	0.65	0.63	0.11
3	**0.80**	0.66	0.65	0.10
4	**0.85**	0.54	0.61	0.09
5	**0.81**	0.55	0.58	0.10
6	**0.84**	0.64	0.65	0.11
7	**0.78**	0.55	0.61	0.10
8	**0.73**	0.18	0.45	0.09
9	**0.72**	0.39	0.51	0.11
10	**0.81**	0.54	0.61	0.08
11	**0.77**	0.52	0.60	0.11
12	**0.72**	0.44	0.54	0.08
13	**0.74**	0.49	0.57	0.10
14	**0.77**	0.57	0.60	0.12
15	**0.81**	0.61	0.63	0.10
16	**0.83**	0.64	0.63	0.11
17	**0.85**	0.68	0.69	0.10
18	**0.75**	0.54	0.57	0.13
19	**0.80**	0.64	0.62	0.11
20	**0.81**	0.58	0.62	0.09
21	**0.84**	0.66	0.63	0.10
Average ± *Stdv*	**0.79 ± 0.04**	0.56 ± 0.12	0.60 ± 0.05	0.10 ± 0.01

Table 3.3: Precision of the out-of-sample predictions

Image	AdaBoost.S	Random Forest	XGBoost	Decision Tree
1	0.79	**0.81**	0.69	0.23
2	0.83	**0.85**	0.70	0.25
3	0.78	**0.85**	0.73	0.25
4	**0.83**	0.81	0.71	0.21
5	0.79	**0.86**	0.66	0.23
6	0.82	**0.85**	0.75	0.26
7	0.78	**0.87**	0.69	0.24
8	0.75	**0.78**	0.65	0.21
9	0.73	**0.80**	0.68	0.25
10	0.80	**0.82**	0.69	0.17
11	0.75	**0.85**	0.69	0.24
12	0.72	**0.77**	0.69	0.18
13	0.73	**0.81**	0.67	0.23
14	0.74	**0.78**	0.66	0.25
15	0.79	**0.82**	0.69	0.24
16	0.81	**0.85**	0.69	0.27
17	0.83	**0.84**	0.74	0.23
18	0.73	**0.78**	0.65	0.24
19	0.78	**0.82**	0.67	0.24
20	0.80	**0.81**	0.69	0.23
21	**0.82**	0.79	0.71	0.24
Average ± Stdv	0.78 ± 0.04	**0.82 ± 0.03**	0.69 ± 0.03	0.23 ± 0.02

Table 3.4: Recall of the out-of-sample predictions

Image	AdaBoost.S	Random Forest	XGBoost	Decision Tree
1	**0.86**	0.54	0.60	0.07
2	**0.85**	0.53	0.58	0.07
3	**0.84**	0.53	0.58	0.07
4	**0.87**	0.41	0.53	0.06
5	**0.83**	0.40	0.51	0.07
6	**0.86**	0.51	0.58	0.07
7	**0.79**	0.42	0.55	0.06
8	**0.70**	0.10	0.34	0.06
9	**0.71**	0.25	0.40	0.07
10	**0.81**	0.40	0.55	0.05
11	**0.79**	0.38	0.53	0.07
12	**0.73**	0.31	0.45	0.05
13	**0.75**	0.35	0.50	0.06
14	**0.80**	0.45	0.55	0.08
15	**0.84**	0.49	0.58	0.07
16	**0.85**	0.51	0.58	0.07
17	**0.87**	0.57	0.65	0.06
18	**0.78**	0.42	0.51	0.09
19	**0.83**	0.52	0.57	0.07
20	**0.83**	0.45	0.56	0.06
21	**0.87**	0.56	0.56	0.07
Average ± Stdv	**0.81 ± 0.05**	0.43 ± 0.11	0.54 ± 0.07	0.07 ± 0.01

3.4 Conclusions

We have proposed a Structured Adaptive Boosting Trees algorithm (AdaBoost.S) for edge detection problems associated with medical image analysis. The paper described in detail the feature extraction process from image patches, and presented the technical details of how the extracted features can be used to predict the edge structures using the proposed AdaBoost.S. The algorithm captures the special edge structures which are highly interdependent. The proposed approach was applied to a fluorescence IVM dataset for edge detection of platelet-neutrophil aggregates within the pulmonary microcirculation of live mice exposed to e-cigarette vapor. AdaBoost.S is widely applicable to analysis of other fluorescence IVM applications including cancer, infection, and cardiovascular disease, with edge detection capability beyond platelets and neutrophils. Our comparison studies have demonstrated the advantages of the proposed AdaBoost.S over existing methods. One important future research direction is to adapt deep learning approaches, especially CNN, for structure learning problems with large training datasets. For future work, CNN for structured learning can be a very important topic once we have more data from our experiments. Also, neutrophil-platelet aggregates can be labeled individually, and the direction and the range of movement can be identified as the shape of clots is not changing drastically in a video. Having such information, we are able to overcome the movement artifact. Furthermore, by labeling different neutrophil-platelet aggregates, we can draw information such as their sizes and perimeters.

References

Bansal, P., Garg, R., & Soni, P. (2022). Detection of melanoma in dermoscopic images by integrating features extracted using handcrafted and deep learning models. *Computers & Industrial Engineering*, *168*, 108060.

Bennewitz, M. F., Jimenez, M. A., Vats, R., Tutuncuoglu, E., Jonassaint, J., Kato, G. J., Gladwin, M. T., & Sundd, P. (2017). Lung vaso-occlusion in sickle cell disease mediated by arteriolar neutrophil-platelet microemboli. *JCI insight*, *2*(1).

Brinkmann, V., Reichard, U., Goosmann, C., Fauler, B., Uhlemann, Y., Weiss, D. S., Weinrauch, Y., & Zychlinsky, A. (2004). Neutrophil extracellular traps kill bacteria. *science*, *303*(5663), 1532–1535.

Demers, M., Krause, D. S., Schatzberg, D., Martinod, K., Voorhees, J. R., Fuchs, T. A., Scadden, D. T., & Wagner, D. D. (2012). Cancers predispose neutrophils to release extracellular DNA traps that contribute to cancer-associated thrombosis. *Proceedings of the National Academy of Sciences*, *109*(32), 13076–13081.

Deriche, R. (1992). Recursively implementing the Gaussian and its derivatives. *Proc. Secound Int. Conf. On Image Processing*, 263–267.

Dhanachandra, N., Manglem, K., & Chanu, Y. J. (2015). Image segmentation using K-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, *54*, 764–771.

Dollar, P., Tu, Z., & Belongie, S. (2006). Supervised learning of edges and object boundaries. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, *2*, 1964–1971.

Dollár, P., & Zitnick, C. L. (2013). Structured forests for fast edge detection. *Proceedings of the IEEE international conference on computer vision*, 1841–1848.

Etulain, J., Martinod, K., Wong, S. L., Cifuni, S. M., Schattner, M., & Wagner, D. D. (2015). P-selectin promotes neutrophil extracellular trap formation in mice. *Blood, The Journal of the American Society of Hematology*, *126*(2), 242–246.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, *55*(1), 119–139.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning, 2nd Edition*. Springer.

Huang, Z., Huang, L., Gong, Y., Huang, C., & Wang, X. (2019). Mask scoring r-cnn. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6409–6418.

Kohler, R. (1981). A segmentation system based on thresholding. *Computer Graphics and Image Processing*, *15*(4), 319–338.

Kolaczkowska, E., Jenne, C. N., Surewaard, B. G., Thanabalasuriar, A., Lee, W.-Y., Sanz, M.-J., Mowen, K., Opdenakker, G., & Kubes, P. (2015). Molecular mechanisms of NET formation and degradation revealed by intravital imaging in the liver vasculature. *Nature communications*, *6*(1), 1–13.

Lim, J. J., Zitnick, C. L., & Dollár, P. (2013). Sketch tokens: A learned mid-level representation for contour and object detection. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3158–3165.

Ma, A., & Kubes, P. (2008). Platelets, neutrophils, and neutrophil extracellular traps (NETs) in sepsis. *Journal of Thrombosis and Haemostasis*, *6*(3), 415–420.

Nowozin, S., & Lampert, C. H. (2011). *Structured learning and prediction in computer vision* (Vol. 6). Now publishers Inc.

Ramana, R., Rathnam, T., & Reddy, A. S. (2017). A review on edge detection algorithms in digital image processing applications. *International Journal on Recent Innovation Trends in Computing and Communication*, *5*(10), 69–75.

Rapoport, N., Gupta, R., Kim, Y.-S., & O'Neill, B. E. (2015). Polymeric micelles and nanoemulsions as tumor-targeted drug carriers: insight through intravital imaging. *Journal of Controlled Release*, *206*, 153–160.

Saffarzadeh, M., Juenemann, C., Queisser, M. A., Lochnit, G., Barreto, G., Galuska, S. P., Lohmeyer, J., & Preissner, K. T. (2012). Neutrophil extracellular traps directly induce epithelial and endothelial cell death: a predominant role of histones. *PloS one*, *7*(2), e32366.

Secklehner, J., Lo Celso, C., & Carlin, L. M. (2017). Intravital microscopy in historic and contemporary immunology. *Immunology and cell biology*, *95*(6), 506–513.

Senthilkumaran, N., & Rajesh, R. (2009). Image segmentation-a survey of soft computing approaches. *2009 International Conference on Advances in Recent Technologies in Communication and Computing*, 844–846.

Singh, B., & Sharma, D. K. (2021). SiteForge: Detecting and localizing forged images on microblogging platforms using deep convolutional neural network. *Computers & Industrial Engineering*, *162*, 107733.

Spectrum Health. (2022). Facts vs. Myths [[Online; accessed March 28, 2022]].

Van Vliet, L. J., Young, I. T., & Verbeek, P. W. (1998). Recursive Gaussian derivative filters. *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No. 98EX170)*, *1*, 509–514.

Weigert, R., Sramkova, M., Parente, L., Amornphimoltham, P., & Masedunskas, A. (2010). Intravital microscopy: a novel tool to study cell biology in living animals. *Histochemistry and cell biology*, *133*(5), 481–491.

Xie, S., & Tu, Z. (2015). Holistically-nested edge detection. *Proceedings of the IEEE international conference on computer vision*, 1395–1403.

Zheng, S., Yuille, A., & Tu, Z. (2010). Detecting object boundaries using low-, mid-, and high-level information. *Computer Vision and Image Understanding*, *114*(10), 1055–1067.

Ziou, D., Tabbone, S. et al. (1998). Edge detection techniques-an overview. *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, *8*, 537–559.

4 A Review of Random Forest Based Feature Selection Techniques

4.1 Introduction

High dimensional datasets have been provided in recent years, especially those collected in gene expression studies (Hua et al., 2005; Jirapech-Umpai & Aitken, 2005; Lee et al., 2005; Liu et al., 2010; Ruiz et al., 2006; Zhu et al., 2007). Hence, many feature selection techniques have been widely introduced in order to select a subset of features from a feature set such that the new set increases the predictive accuracy by removing unimportant or non-informative features, enhances the interpretability of a former complex data structure, and significantly reduces the computational complexity of the predictor. Although many feature selection techniques are available, finding the appropriate method for a specific application could be an overwhelming task. Therefore, we need to classify them based on their nature as well as their objectives.

In general, feature selection methods are classified into *filters*, *embedded* algorithms, and *wrappers* (Guyon & Elisseeff, 2003; Kohavi & John, 1997).

Filters employ some statistical measures to evaluate the relationship between the predictors and the response vector such as correlation prior to the model building process. They select a subset of features independent from the predictor, which in this paper all are classifiers. One major drawback of filters is that they ignore the model performance, although they usually are faster than wrappers.

Embedded algorithms are the ones that contain built-in feature selection. These techniques perform along the training process and select features that ensure accuracy improvements.

Finally, wrappers include predictive performance in their feature selection objective. Meaning they fit different models to different combinations of features and identify the optimal set of features that maximizes model performance. Also, wrappers can be built over any kind of model that calculates feature importance. It is true that wrappers have the ability to eliminate unimportant features, but the risk of overfitting is eminent.

Random Forests (RF) successful performance have been shown in a wide variety of applications. RF carries many advantages, such as its robustness against overfitting, the ability to handle noisy and missing data, and calculating measures of the so-called variable importance (VIM) (Breiman, 2001). Therefore, RF is a strong candidate for building wrapper feature selection techniques. RF based feature selection techniques or wrappers built over RF have been widely considered as the need for feature selection has been surging.

In this paper, We only focused on wrapper feature selection techniques built over RF. Before we move on, we further categorized the methods as either *performance-based* or *test-based*. Performance-based methods tend to work under a forward selection and/or a backward elimination strategy to select or remove features in the direction of prediction accuracy improvements. On the other hand, test-based approaches tend to incorporate permutation frameworks in their algorithms and exploit statistical tests to identify statistically significant features (Speiser et al., 2019).

One also needs to mention that different feature selection methods have various goals and address different problems. One objective is finding a *minimal optimal set* in which they identify a possible subset of features to perform the prediction. The other objective is known as identifying *all relevant* features (strongly relevant and weakly relevant) in which removing these features has a negative impact on the prediction accuracy (Kursa & Rudnicki, 2011).

The rest of this paper outlined as follows: Section 4.2 reviews background on RF classifiers. Wrapper feature selection methods built over RF are presented in details in Section 4.3. Numerical results are provided in Section 4.4, and Section 5 concludes the paper.

4.2 Random Forest

Random Forests (RF) are one of the most successful statistical learning approaches that build an ensemble by growing many uncorrelated classification or regression trees (CART) in parallel. The idea looks straightforward, combining many noisy but almost unbiased models in order to reduce the variance of an estimated prediction function. The RF method is a supervised learner that builds fully-grown trees and unpruned. For the ease of explanation, we provide Algorithm 2.

At the bootstrapping sampling, a portion of the dataset is picked with the replacement for building a tree and those that opt out, are called out-of-bag (*OOB*) and are used to provide an

Algorithm 2: Random Forests for Regression and Classification

```
1 Set the number of trees $ntree = B$
2 Set the value of $mtry = m$
3 for b=1,...,B do
4     Grow tree $t$ by repeating the following steps:
5     (i) Draw a bootstrap sample of size N from the training set.
6     (ii) Grow a binary decision tree $T_b$ to the bootstrapped data as follow, until the
        stoppage criterion is met:
          • Randomly select m variable from the feature set
          • Find the best split variable, and split value that minimize impurity
          • Split the node into two daughter nodes
      (iii) Output the ensemble of trees $\{T_b\}_1^B$
7 end
8 To predict a new data point: use majority voting for classification problems and average
    for regression problems
```

unbiased estimate of the model predictive performance. After growing each tree, an *OOB* error rate is estimated like the way errors are calculated by N-fold cross-validation. Randomness in RF helps to develop uncorrelated trees and as a result, has a low error rate. Also using a subset of variables (i.e. *mtry*) at splitting nodes further reduces the model complexity and makes the model robust to overfitting. Although, there is a rule of thumb for setting $m = \sqrt{p}$ yet choosing the right *mtry* is highly dependent on the problem and can be treated as a tuning parameter.

To elaborate, consider a set of *B* independent and identically distributed random variables. The average on these *B* random variables, each with variance σ^2, produces variance $\frac{1}{B}\sigma^2$. On the other hand, if we exclude the independence assumption among variables, the average of these *B* variables becomes:

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2 \tag{4.1}$$

where ρ is the positive pairwise correlation.

RF fades the second term away by creating many trees. Also, it seeks to further reduce the overall ensemble variance by adding more noise to disrupt the correlation among trees while being conservative about increasing the variables' variance. This strategy is achieved by bootstrap sampling from the training set at each tree and by using a random subset of variables at each tree node.

Finally, RF uses majority voting and averaging over all individual trees for classification and regression problems, respectively to determine the prediction.

RF carries many advantages, such as its robustness against overfitting, the ability to handle noisy and missing data, and calculating measures of the so-called variable importance measures (VIM).

4.2.1 Variable Importance Scores

The RF algorithm has the built-in property that calculates variable importance scores in two different ways. By Mean Decrease Impurity (also known as Gini importance) or Mean Decrease Accuracy (also known as Permutation importance). Either way, RF records the improvement in the split criterion separately for each variable and accumulates them in the end. Putting the scores in order gives a ranking in which the most important variable stands at the top of the list.

• **Mean Decrease Impurity**

The Gini index at node ν can be computed as follow:

$$Gini(\nu) = \sum_{c=1}^{C} p_c^{\nu}(1 - p_c^{\nu}) \tag{4.2}$$

where p_c^{ν} is a portion of class-c observations at node ν. The Gini impurity of the variable X_i for the two daughter nodes of ν is then derived from:

$$Gain(X_i, \nu) = Gini(X_i, \nu) - \omega_L Gini(X_i, \nu^L) - \omega_R Gini(X_i, \nu^R) \tag{4.3}$$

let ν^L and ν^R be two daughter nodes of ν, and ω^L and ω^R be proportions of observations dropped in each daughter node.

The Gini criterion enables the RF classification model to split nodes at their highest impurity. RF then calculates the variable importance (also known as Gini importance) by taking the average of all decreases in Gini impurity.

Variable importance ranks features based on their contribution to the training process. Although VIMs are an excellent source of variable assessments yet there is no benchmark to distinguish informative from non-informative features. The next section introduces 12 wrappers build around RF classification to cover the above issue.

• **Mean Decrease Accuracy**

The mean decrease accuracy is a method that evaluates the variable contributions in building the ensemble by permuting *OOB* samples of each feature one after another. For each variable, the predictive accuracy is once again evaluated and the decrease of performance induced by permutation is averaged over all trees. The obtained values are then used as a measure of the variable importance.

4.3 Feature Selection Methods

In this section, we provide a comprehensive review of the feature selection methods based on RF for classification problems. In particular, a total number of 12 methods are reviewed, including Boruta, RRF, GRRF, GRF, r2VIM, PIMP, NTA, varSelRF, VSURF, RF-SRC, AUC-RF and RFE. For each method, we describe its main idea as well as the R packages for implementation. A high-level overview of these methods is provided in Table 4.1.

Method	References	R Package	Approach	Strategy
Boruta	Kursa and Rudnicki, 2010	*Boruta*	Test-based	Permutation based algorithm
RRF	Deng and Runger, 2012	*RRF*	Performance-based	Forward feature selection using regularized RF
GRRF	Deng and Runger, 2013	*RRF*	Performance-based	Guide RRF using ordinary RF
GRF	Deng, 2013	*RRF*	Performance-based	A subset of GRRF that allows parallel computation
r2VIM	Szymczak et al., 2016	*Pomona*	Test-based	Permutation based algorithm
PIMP	Altmann et al., 2010	*vita*	Test-based	Permutation test
NTA	Janitza et al., 2018	*vita*	Test-based	Hold-out VIM test
varSelRF	Dıéaz-Uriarte and Alvarez de Andrés, 2006	*varSelRF*	Performance-based	Backward feature elimination using OOB-error rate
VSURF	Genuer et al., 2015	*VSURF*	Performance-based	Two-step procedure, first applies backward elimination then forward selection
RF-SRC	Ishwaran et al., 2010	*randomForestSRC*	Performance-based	Forward stepwise regularized feature selection
AUCRF	Calle et al., 2011	*AUCRF*	Performance-based	Backward feature elimination using OOB-AUC
RFE	Guyon et al., 2002	*caret*	Performance-based	Backward feature elimination using OOB-error rate

Table 4.1: A summary of wrapper methods based on RF for classification problems

4.3.1 Boruta

Boruta, proposed by Kursa and Rudnicki (2010), is a wrapper feature selection method built around the RF classification algorithm. Boruta is a test-based heuristic approximation algorithm that attempts to find a threshold for feature selection rather than ranking features with some VIMs. In other words, it solves the challenging all-relevant feature selection problem rather than finding a minimal set of features. Boruta has been implemented in the R package, `Boruta` (Kursa & Rudnicki, 2010).

The core idea behind the Boruta is that a feature is not important if its calculated importance is less than the importance of a randomly permuted feature. Because removing irrelevant features is expected to increase the accuracy in computing the variable importance, Boruta hinges upon sequentially removing features that are found significantly irrelevant.

At each iteration, the Boruta algorithm duplicates each feature that exists in the feature set with a random permutation of their observations. The duplicated features are referred to as the *shadow features*. The RF is performed upon all features, including the shadow features, and the feature importance is computed. Then, the features that have the higher importance compared to the maximum importance among all the shadow features are deemed relevant, while the remaining features are considered irrelevant and removed from the feature set. Since the random forest classifier produces different importance measures due to its randomness and the presence of shadow features, Boruta usually repeats the process above multiple times until no more irrelevant features can be removed. Finally, Boruta categorizes each feature as relevant (irrelevant) if it has an importance significantly higher (lower) than the maximum $Z-$score among all random shadow features (MZSF). For those features with close importance measure to the best shadow feature (known as tentative features), the Boruta algorithm does not make any decisions and let the user decide if those tentative features should be included given the context of the application.

4.3.2 RRF, GRRF, and GRF

Regularized Random Forest (RRF) is a wrapper feature selection technique built over the RF binary classification problems. Unlike Boruta, RRF attempts to find a minimal optimal set of relevant features and remove non-relevant features. RRF is available in the R package `RRF`. (Deng & Runger, 2012)

Note that, although it is often possible to select the first K features with the highest importance scores using the RF algorithm, selecting non-relevant features among the K features is likely in the presence of correlated features. Hence, the RRF is an ensemble and greedy feature selection technique that employs a regularized framework together with an upper bound of the Gini information gain value when computing the feature importance. The regularized feature selection by RRF helps on identify a compact feature subset possible to perform the prediction. Lets F be the empty set of indices, $\lambda \in (0, 1]$ be the penalty coefficient, and $Gain^* = 0$ be the initial upper bound of Gini information gain. At each tree node, for any feature X_i that are not in the set of indices $i \notin F$, RRF penalizes it by multiplying λ with the Gini information gain. The regularized gain for variable X_i at a non-leaf node ν is then calculated as follow:

$$\text{Gain}_R(X_i, \nu) = \begin{cases} \lambda \times \text{Gain}(X_i, \nu) & \text{if} \quad i \notin F \\ \text{Gain}(X_i, \nu) & \text{if} \quad i \in F \end{cases} \tag{4.4}$$

Hence, the split on X_i only occurs if the $\text{Gain}_R(X_i, \nu)$ exceeds the upper bound $Gain^*$ obtained from previous splits, and the $Gain^*$ is updated after the node splitting. A comprehensive description of the algorithm is provided in Deng and Runger (2012).

Note that, although RRF can be used as an ensemble classifier, RRF is often recommended to use only for feature selection purposes. It is also noted that, as the depth of a tree increases, fewer observations may drop in non-leaf nodes. The lack of enough observations may affect calculating the Gini information gain, known as the node sparsity issue. Because of the node sparsity issue, selecting a subset of features that includes weakly relevant features is probable.

To alleviate the node sparsity issue, Deng and Runger (2013) proposed an enhanced version of the RRF called the Guided Regularized Random Forest (GRRF). GRRF is also available in the R package `RRF`. GRRF aims to select a compact feature set by building multiple ensembles, and features are evaluated on the entire training set. GRRF incorporates the importance scores calculated by an ordinary RF to guide the feature selection procedure in RRF, while the penalty coefficient λ is no longer fixed for the entire feature set. In other words, GRRF dynamically penalizes features out of the set of indices as follows:

$$\text{Gain}_R(X_i, \nu) = \begin{cases} \lambda_i \times \text{Gain}(X_i, \nu) & \text{if} \quad i \notin F \\ \text{Gain}(X_i, \nu) & \text{if} \quad i \in F \end{cases} \tag{4.5}$$

where λ_i is given by

$$\lambda_i = 1 - \gamma(1 - \text{Imp}_i) \tag{4.6}$$

with γ and Imp_i respectively being the weighting control parameter and the normalized variable importance score of the variable X_i.

Based on experimental results on 10 gene datasets, it has been shown that GRRF may be more robust than RRF against parameter changes, while the overall accuracy of RRF is higher. Hence, RRF is recommended when accuracy is the major concern, while one may choose GRRF if shrinking the feature set is the priority.

Finally, Deng (2013) proposed the Guided Random Forest (GRF) as a special case of GRRF. GRF also shares the same idea of using a regular RF VIM to guide the feature selection process. However, despite the GRRF that uses sequentially grown trees, GRF intends to build independent trees in which parallel computation is applicable. Moreover, GRF removes the regularized part in GRRF, making each split in a tree node highly dependent on previous splits. In a numerical investigation based on 10 gene datasets, GRF in general selects more features than GRRF does. However, it ends up building a more accurate classification RF model than building the RF using the entire features. GRF is available in the R package `RRF` as well.

4.3.3 r2VIM

Recurrent Relative Variable Importance Measure (r2VIM) was proposed by Szymczak et al. (2016) and is implemented in the R package `Pomona` (Fouodo, 2022). r2VIM is a test-based feature selection technique built around the RF. The method is applicable for both regression or classification problems. Like Boruta, the main goal is to find all-relevant features. In contrast, r2VIM establishes a criterion to determine the number of features that need be selected.

The main idea of r2VIM is that relevant features have relatively high variable importance scores no matter how many times and with what seeds we run the RF. On the other hand, irrelevant features only occasionally have high importance scores.

The r2VIM algorithm involves three main components. First, it builds several RFs and calculates VIMs associated with the ensembles. Second, since observing negative VIM for a noise feature is probable, each variable importance score is divided by the observed absolute value of the minimum importance score, called the relative importance score. Finally, all variables with relative importance scores larger than a threshold in all runs are selected as all-relevant features. The main advantage of r2VIM is that it is able to limit the number of false-positives under the null hypothesis.

4.3.4 PIMP

Permutation Importance (PIMP) was proposed by Altmann et al. (2010) to distinguish relevant predictors from less important ones. The key idea of using permutation in PIMP is to destroy any sort of correlations between features and the response variable. PIMP is a heuristic approach that attempts to find unbiased importance scores by fitting RFs to different permutations of the response vector.

The PIMP algorithm starts by obtaining variable importance scores using the intact response vector. Then, RFs are fitted on N different permutations of the response vector and the importance scores for all variables are calculated. Finally, based on the N sets of important scores, the *p*-value is computed for each variable. Here, the *p*-values can be obtained by computing the fraction of N importance scores that exceed the original importance score. Very often, to reduce the number of permutations, a prior distribution such as Gaussian, log-normal, or gamma can be assumed. The PIMP algorithm fits the distribution by computing the maximum likelihood estimates. After that, the *p*-values for each variable are defined as the probability of observing an importance score greater than the original importance score. Once the *p*-values for all variables have been computed, variables with *p*-values less than a predetermined threshold (e.g. 0.05) are statistically significant and thus selected. The R code for this method is available in the R package `vita`. (Celik, 2015)

4.3.5 NTA

One limitation that almost all test-based feature selection techniques is that RF does not provide a threshold for selecting features. Hereby, Janitza et al. (2018) proposed a Naive and New Testing Approach (NTA), a computationally fast heuristic variable importance test, that aims to find a cut-off point in the VIMs generated by RF so as to find all-relevant features for classification problems. NTA is available in the R package `vita`. (Celik, 2015)

NTA uses the hold-out variable importance, also known as the two-fold cross-validation method. It first splits the entire dataset into two equal-sized subsets. Then, RF is applied to one of the subsets and the variable importance scores are calculated for the other set. This process is repeated for the other subset as well. The hold-out variable importance is defined as the mean variable importance scores. The main idea of NTA is that irrelevant features do not create positive variable importance. Based on the non-positive variable importance scores, NTA constructs an approximate null hypothesis distribution, and subsequently, computes the *p*-values corresponding to the features of interest. Finally, the features that are statistically significant are included as all-relevant features.

4.3.6 varSelRF and RFE

Dıéaz-Uriarte and Alvarez de Andrés (2006) proposed a performance-based Variable Selection method using Random Forests (varSelRF) and is available in the R package `varSelRF` (Diaz-Uriarte, 2007). Using a backward elimination strategy under the aggressive variable selection framework, the goal of this method is to find the smallest possible set of variables that has relatively good predictive accuracy for either two-class or multi-class classification problems. The varSelRf algorithm calculates variable importance only once by fitting RF on all features. Then, it fits several RFs successively, and for each RF model, a predefined fraction of features with the lowest importance scores (e.g., 20%) is removed. Finally, the algorithm identifies the smallest set of features that has the smallest OOB error rate with an error rate within a chosen standard errors of the minimum error rate among the fitted RFs.

It is noted that the varSelRF algorithm above ranks feature importance only once, which may not be suitable in the existence of highly correlated predictors. Hence, Guyon et al. (2002) proposed a modified version of varSelRf called Recursive Feature Elimination (RFE) to handle the issue above. RFE is a performance-based feature selection algorithm that works under a backward elimination strategy to find a minimal set of features with the minimum OOB-error rate. RFE is an iterative algorithm where at each step:

(1) A RF model is fitted.

(2) Variable importance is calculated.

(3) OOB-error rate is estimated using samples that were not used in the model.

(4) A predefined ratio of least important features is removed from the feature set.

One needs to stress that, unlike varSelRF, RFE repeatedly calculates variable importance. The algorithm stops when only a single feature remains. In the end, RFE identifies a set of features that has the minimum OOB-error rate or has an error within a small range of the minimum error. RFE is available in the R package `caret`. (Kuhn, 2020)

4.3.7 VSURF

Variable Selection Using Random Forests (VSURF) is another performance-based feature selection technique applicable to both regression and classification problems. VSURF outputs two subsets of features: one subset for interpreting purposes including all features that are highly correlated with the response variable, and the other subset (known as the interpretation subset) for predicting purposes that exclude redundant features so that the model involves only a smaller number of features (Genuer et al., 2015). The algorithm is available in the R package `VSURF`. (Genuer et al., 2019)

The VSURF algorithm consists of two steps: (*i*) preliminary elimination and ranking, and (*ii*) variable selection. In the first step, a number of RFs are constructed and the variable importance scores for all variables are obtained. Then, taking the average over all importance scores from all RF runs, variables are sorted in descending manner in terms of their importance and those with the least importance scores are removed from the feature set.

In the second step, using features retained from step one with the same order, VSURF fits a RF to the most important variable and iteratively adds another variable until the last RF is fitted to all variables. The interpretation subset thus becomes the smallest feature set that yields the lowest OOB error rate with an error rate within its standard deviation. In particular, VSURF uses a forward selection strategy to further shrink the feature set to achieve a better prediction performance. A threshold is derived as follows:

$$\frac{1}{m-m^{'}}\sum_{j=m^{'}}^{m-1}|\text{errOOB}(j+1)-\text{errOOB}(j)| \tag{4.7}$$

where m is the number of variables selected in the first step, $m^{'}$ is the number of variables that exist in the interpretation subset, and errOOB(j) is the OOB error of the j-th RF constructed. Hence, a variable is selected if the decrease of OOB error exceeds the threshold.

4.3.8 RF-SRC

Random Forests for Survival, Regression, and Classification problems (RF-SRC) was proposed and developed in the R package `randomForestSRC` (Ishwaran et al., 2022). In the confrontation with high-dimensional data, the number of features p may exceed the sample size n. As a result, the trees in a RF model cannot reach sufficient depth where the predictive variables can be identified. Motivated by this limitation, Ishwaran et al. (2010) proposed the Variable Hunting (RSF-VH) approach which performs as a forward stepwise regularized feature selection method.

The key idea behind RSF-VH is that features that are being split in nodes closer to the root node are likely to be more important. Hence, a new concept of order statistic, called the minimal depth of maximal subtrees, is employed to calculate variable importance scores. The algorithm of RSF-VH works as follows:

(1) Fit a RF and select features using minimal depth thresholding.

(2) The set of features obtained in step (1) is used as an initial model. After that, other features are added to the initial model based on minimal depth ranking until the variable importance becomes stabilized in the nested models.

(3) Repeat steps 1 and 2 several times. Finally, a feature that appears most is selected if its size is greater than the average.

It is reported that, for very high-dimensional microarray datasets, RSF-VH is able to select a small set of features, genes in these particular datasets, along with low predictive error compared to other state-of-the-art methods. (Wang & Li, 2017)

4.3.9 AUC-RF

AUC-RF was proposed by Calle et al. (2011) and is available in the R package `AUCRF` (Urrea & Calle, 2012). AUC-RF is a performance-based variable selection method using RFs for problems with a labeled response vector. Using a backward elimination strategy under the aggressive variable selection framework, the goal of this method is to find the set of features that has the highest area under the ROC curve (AUC).

Like varSelRf, AUC-RF fits several RF models successively. For each RF constructed, feature importance scores are obtained and features with the lowest importance scores are eliminated by a predetermined ratio between 0 and 1 (e.g., 0.2). Finally, the algorithm selects a set of predictive variables that has the highest OOB-AUC value among all RFs constructed.

4.4 Examples and Discussions

In this section, we apply the feature selection methods reviewed in Section 4.3 to three datasets, and compare the results generated by different approaches. The data used in this section is obtained from the UCI Machine Learning Database Repository (Frank, 2010).

The first dataset is the Sonar, Mines vs Rocks (SMR) dataset (Gorman & Sejnowski, 1988). This dataset contains the sonar (sound navigation and ranging) data that can be used to distinguish metal sea mines and rocks on seafloor. This dataset contains 60 features and 208 instances where 111 of them are metal samples and the rest are rock samples. The features are derived from the receiving sonar signals under various conditions and from different angles, spanning 90 degrees for the cylinder and 180 degrees for the rocks.

The second dataset is the Wisconsin Breast Cancer (WBC) dataset from patients diagnosed with cancer(Wolberg & Mangasarian, 1990; Zhang, 1992). This dataset can be used to predict whether a cancerous tumor is malignant or benign utilizing the information obtained from biopsy procedures. The dataset has 9 features and involves 699 patients. Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

The third dataset is the Spam Emails (SE) dataset (Hopkins et al., 1999). The data are collected from personal and work-related emails, and include 4601 sample emails categorized into two groups: spam and nonspam. A total number of 57 features are available, including the ratio of 48 different words that repeat in an email over the total number of words, the frequency of 6 different characters occurring in the email, and 3 other features relating to uninterrupted sequences of capital letters.

For all three datasets, we apply the ordinary Random Forest (using all features) as well as the 12 feature selection approaches (using selected features). Figures 4.1, 4.2, and 4.3 respectively show the number of features selected by each method for the three datasets. An initial examination immediately suggests that *the number of features selected by different feature selection methods vary dramatically*. One that needs to be explained here is that having more features or less features highly depends on the user preferences and the nature of the application. Too many features add much complexity to models that cause excessively slow computational processes. Therefore, feature selection techniques try to eliminate features with less predictive importance and sacrifice some predictive accuracy in favor of interpretability and computational complexity. There must be a trade-off between them, but we can safely say in a case that if predictive accuracy did not change much by shrinking its feature set, a smaller feature set is preferred for sure. For all three datasets, GRF always selects the smallest number of features based on all three datasets, while Boruta tends to include most of the features. The lack of consistency is the first indication that one should always carefully choose feature selection methods given the problem at hand.

Furthermore, Tables 4.8, 4.9, and 4.10 respectively show exactly what features are selected by different approaches for the three datasets. In these tables, “1” and “0” respectively denote that a feature is included or excluded, and the last column shows the number of times that a feature is selected by the 12 feature selection methods. It is not surprising to see that different feature selection methods select different sets features, as these methods are built upon different ideas. One interesting question to answer is to see which of these feature selection methods tends to select the same features. In other words, we want to detect associations or relationships between feature selection methods. For example, if a feature is picked by a method set, how likely would it be to be picked by another method set. To shed some light on this question, we perform an association analysis of the features.

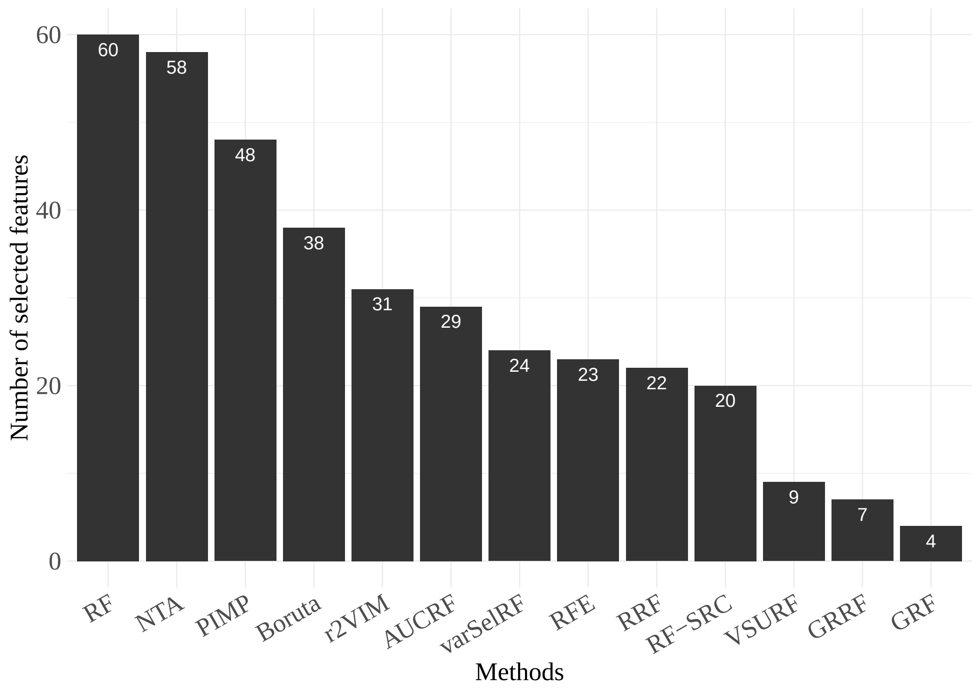


Figure 4.1: Number of features selected by each method for the SMR dataset.

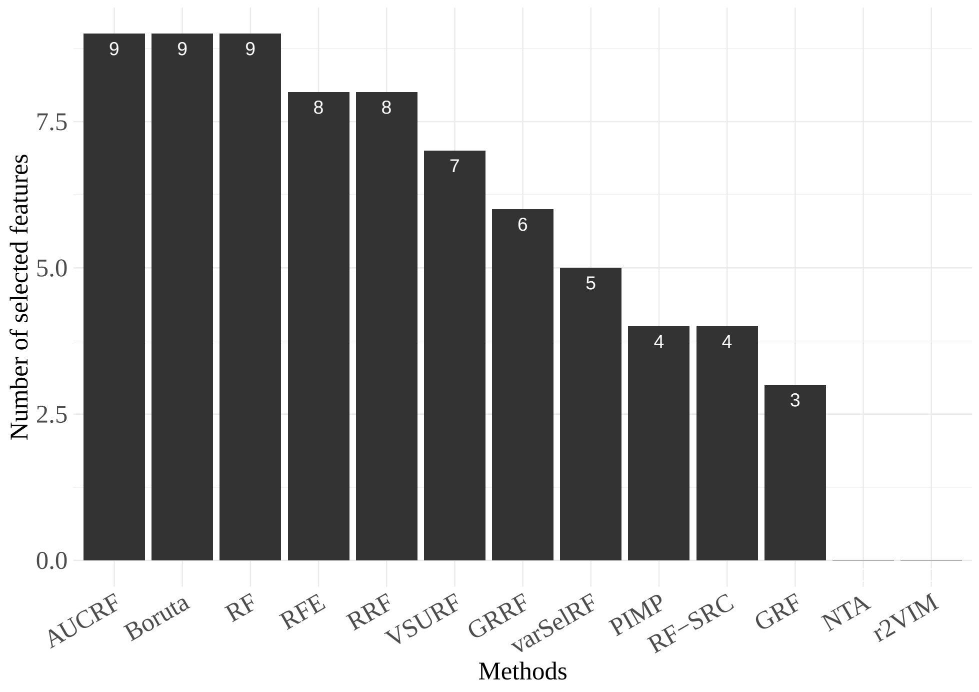


Figure 4.2: Number of features selected by each method for the WBC dataset.

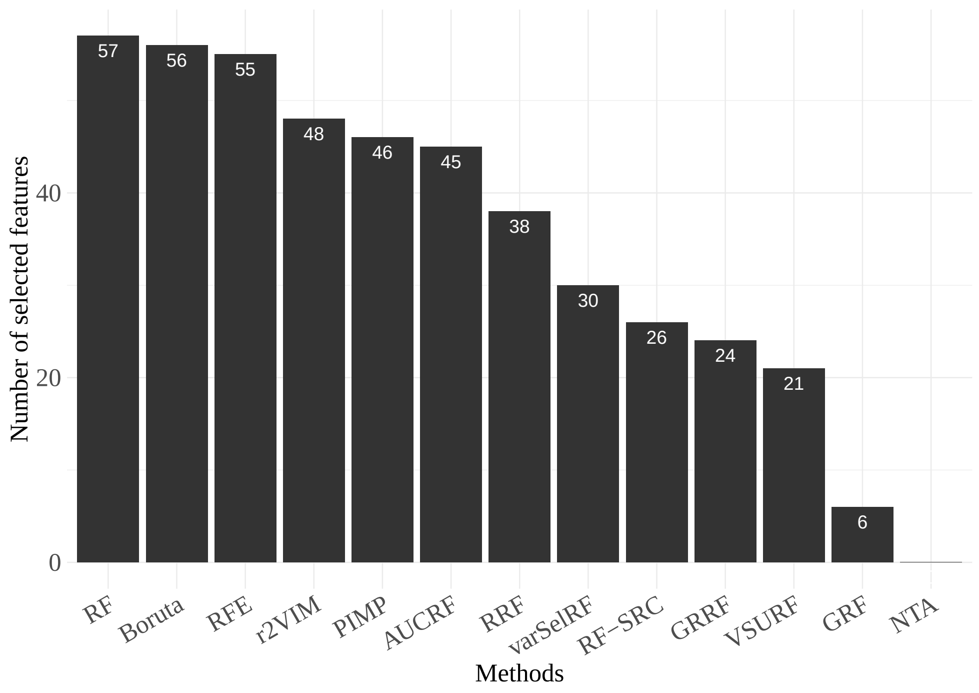


Figure 4.3: Number of features selected by each method for the SE dataset.

Table 4.2: Association analysis sorted by support on SMR dataset

Left method set	Right method set	Support	Confidence
PIMP	NTA	0.8	1
NTA	PIMP	0.8	0.82
Boruta	NTA	0.63	1
NTA	Boruta	0.63	0.65
Boruta	PIMP	0.58	0.92
PIMP	Boruta	0.58	0.72
Boruta, PIMP	NTA	0.58	1
Boruta, NTA	PIMP	0.58	0.92
NTA, PIMP	Boruta	0.58	0.72
r2VIM	NTA	0.51	1
NTA	r2VIM	0.51	0.53
r2VIM	Boruta	0.5	0.96
Boruta	r2VIM	0.5	0.78
Boruta, r2VIM	NTA	0.5	1
r2VIM, NTA	Boruta	0.5	0.96

Table 4.3: Association analysis sorted by confidence on SMR dataset

Left method set	Right method set	Support	Confidence
GRRF	AUCRF	0.11	1
GRRF	r2VIM	0.11	1
GRRF	Boruta	0.11	1
GRRF	NTA	0.11	1
VSURF	varSelRF	0.15	1
VSURF	AUCRF	0.15	1
VSURF	r2VIM	0.15	1
VSURF	Boruta	0.15	1
VSURF	NTA	0.15	1
RFE	NTA	0.38	1
RF.SRC	AUCRF	0.33	1
RF.SRC	r2VIM	0.33	1
RF.SRC	Boruta	0.33	1
RF.SRC	NTA	0.33	1
varSelRF	Boruta	0.40	1

Table 4.4: Association analysis sorted by support on WBC dataset

Left method set	Right method set	Support	Confidence
AUCRF	Boruta	1	1
Boruta	AUCRF	1	1
RFE	AUCRF	0.89	1
AUCRF	RFE	0.89	0.89
RFE	Boruta	0.89	1
Boruta	RFE	0.89	0.89
RRF	AUCRF	0.89	1
AUCRF	RRF	0.89	0.89
RRF	Boruta	0.89	1
Boruta	RRF	0.89	0.89
RFE, AUCRF	Boruta	0.89	1
Boruta, RFE	AUCRF	0.89	1
Boruta, AUCRF	RFE	0.89	0.89
RRF, AUCRF	Boruta	0.89	1
Boruta, RRF	AUCRF	0.89	1

Table 4.5: Association analysis sorted by confidence on WBC dataset

Left method set	Right method set	Support	Confidence
GRF	RF.SRC	0.33	1
GRF	varSelRF	0.33	1
GRF	GRRF	0.33	1
GRF	VSURF	0.33	1
GRF	RRF	0.33	1
GRF	AUCRF	0.33	1
GRF	Boruta	0.33	1
PIMP	varSelRF	0.44	1
PIMP	VSURF	0.44	1
PIMP	RRF	0.44	1
PIMP	AUCRF	0.44	1
PIMP	Boruta	0.44	1
RF.SRC	GRRF	0.44	1
RF.SRC	VSURF	0.44	1
RF.SRC	RRF	0.44	1

Table 4.6: Association analysis sorted by support on SE dataset

Left method set	Right method set	Support	Confidence
RFE	Boruta	0.95	0.98
Boruta	RFE	0.95	0.96
r2VIM	RFE	0.84	1
RFE	r2VIM	0.84	0.87
r2VIM	Boruta	0.84	1
Boruta	r2VIM	0.84	0.86
r2VIM, RFE	Boruta	0.84	1
Boruta, r2VIM	RFE	0.84	1
Boruta, RFE	r2VIM	0.84	0.89
PIMP	RFE	0.81	1
RFE	PIMP	0.81	0.84
PIMP	Boruta	0.81	1
Boruta	PIMP	0.81	0.82
RFE, PIMP	Boruta	0.81	1
Boruta, PIMP	RFE	0.81	1

Table 4.7: Association analysis sorted by confidence on SE dataset

Left method set	Right method set	Support	Confidence
GRF	VSURF	0.11	1
GRF	GRRF	0.11	1
GRF	RF.SRC	0.11	1
GRF	varSelRF	0.11	1
GRF	RRF	0.11	1
GRF	AUCRF	0.11	1
GRF	PIMP	0.11	1
GRF	r2VIM	0.11	1
GRF	RFE	0.11	1
GRF	Boruta	0.11	1
VSURF	RRF	0.37	1
VSURF	r2VIM	0.37	1
VSURF	RFE	0.37	1
VSURF	Boruta	0.37	1
GRRF	AUCRF	0.42	1

Table 4.8: Selected features of SMR dataset

Predictor	Boruta	varSelRF	r2VIM	RFE	RRF	GRRF	GRF	NTA	PIMP	VSURF	AUCRF	RF-SRC	Sum
V1	1	0	1	1	0	0	0	1	1	0	0	0	5
V2	1	0	0	0	1	0	0	1	0	0	0	0	3
V3	0	0	0	0	0	0	0	1	0	0	0	0	1
V4	1	1	1	1	0	0	0	1	1	0	1	1	8
V5	1	0	1	1	0	0	0	1	1	0	1	1	7
V6	0	0	0	0	0	0	0	1	0	0	1	0	2
V7	0	0	0	0	0	0	0	1	0	0	0	0	1
V8	1	0	0	1	1	0	0	1	1	0	0	0	5
V9	1	1	1	1	0	1	1	1	1	0	1	1	10
V10	1	1	1	0	0	1	1	1	1	0	1	1	9
V11	1	1	1	0	1	1	1	1	1	1	1	1	11
V12	1	1	1	1	1	1	1	1	1	1	1	1	12
V13	1	1	1	0	1	0	0	1	1	0	1	1	8
V14	0	0	0	1	0	0	0	1	0	0	0	0	2
V15	1	1	1	0	0	0	0	1	1	0	0	0	5
V16	1	1	1	0	1	0	0	1	1	1	1	1	9
V17	1	1	1	1	1	0	0	1	0	1	1	1	9
V18	1	1	1	0	0	1	0	1	0	0	1	0	6
V19	1	0	0	0	0	0	0	1	1	0	0	0	3
V20	1	1	1	0	1	0	0	1	1	0	1	0	7
V21	1	1	1	1	0	0	0	1	1	1	1	1	9

Continue on the next page

Predictor	Boruta	varSelRF	r2VIM	RFE	RRF	GRRF	GRF	NTA	PIMP	VSURF	AUCRF	RF-SRC	Sum
V22	1	0	1	0	1	0	0	1	1	0	0	0	5
V23	1	1	1	1	1	0	0	1	1	0	1	0	8
V24	0	0	0	0	0	0	0	1	1	0	0	0	2
V25	0	0	0	1	0	0	0	1	1	0	0	0	3
V26	1	0	0	0	0	0	0	1	1	0	0	0	3
V27	1	1	1	0	0	0	0	1	1	1	1	0	7
V28	1	1	1	1	0	0	0	1	1	0	1	1	8
V29	1	0	0	0	0	0	0	1	1	0	0	0	3
V30	0	0	0	1	0	0	0	1	1	0	0	0	3
V31	1	0	1	1	1	0	0	1	1	0	1	0	7
V32	0	0	0	0	1	0	0	1	1	0	0	0	3
V33	0	0	0	0	0	0	0	1	0	0	0	0	1
V34	1	0	0	1	0	0	0	1	1	0	0	0	4
V35	1	1	0	0	0	0	0	1	1	0	1	0	5
V36	1	1	1	1	1	1	0	1	1	1	1	1	11
V37	1	1	1	0	0	0	0	1	1	1	1	1	8
V38	0	0	0	0	0	0	0	1	1	0	0	0	2
V39	0	0	1	0	0	0	0	1	1	0	1	0	4
V40	0	0	0	0	0	0	0	1	0	0	0	0	1
V41	0	0	0	0	0	0	0	0	0	0	0	0	0
V42	0	0	0	1	0	0	0	1	1	0	0	0	3

Continue on the next page

Predictor	Boruta	varSelRF	r2VIM	RFE	RRF	GRRF	GRF	NTA	PIMP	VSURF	AUCRF	RF-SRC	Sum
V43	1	0	1	1	1	0	0	1	1	0	0	0	6
V44	1	1	1	1	1	0	0	1	1	0	1	0	8
V45	1	1	1	0	1	0	0	1	1	0	1	1	8
V46	1	1	1	0	1	0	0	1	1	0	1	1	8
V47	1	1	1	1	1	0	0	1	1	0	1	1	9
V48	1	1	1	0	1	0	0	1	1	1	1	1	9
V49	1	1	1	1	0	0	0	1	1	0	1	1	8
V50	0	0	0	0	0	0	0	1	1	0	0	0	2
V51	1	0	1	1	0	0	0	1	1	0	1	1	7
V52	1	0	1	1	1	1	0	1	1	0	1	1	9
V53	0	0	0	0	1	0	0	1	1	0	0	0	3
V54	0	0	0	0	0	0	0	1	1	0	0	0	2
V55	0	0	0	0	0	0	0	1	1	0	0	0	2
V56	0	0	0	0	1	0	0	0	0	0	0	0	1
V57	0	0	0	0	0	0	0	1	0	0	0	0	1
V58	0	0	0	0	0	0	0	1	1	0	0	0	2
V59	1	0	0	0	0	0	0	1	1	0	0	0	3
V60	0	0	0	0	0	0	0	1	1	0	0	0	2

Table 4.9: Selected features of WBC dataset

Predictor	Boruta	varSelRF	r2VIM	RFE	RRF	GRRF	GRF	NTA	PIMP	VSURF	AUCRF	RF-SRC	Sum
Cl.thickness	1	1	0	1	1	0	0	0	1	1	1	0	7
Cell.size	1	1	0	0	1	1	1	0	1	1	1	1	9
Cell.shape	1	1	0	1	1	1	1	0	0	1	1	1	9
Marg.adhesion	1	0	0	1	1	0	0	0	0	0	1	0	4
Epith.c.size	1	0	0	1	1	1	0	0	0	1	1	0	6
Bare.nuclei	1	1	0	1	1	1	1	0	1	1	1	1	10
Bl.cromatin	1	0	0	1	1	1	0	0	0	1	1	1	7
Normal.nucleoli	1	1	0	1	1	1	0	0	1	1	1	0	8
Mitoses	1	0	0	1	0	0	0	0	0	0	1	0	3

Table 4.10: Selected features of SE dataset

Predictor	Boruta	varSelRF	r2VIM	RFE	RRF	GRRF	GRF	NTA	PIMP	VSURF	AUCRF	RF-SRC	Sum
make	1	0	0	1	1	0	0	0	0	0	1	0	4
address	1	0	1	1	0	0	0	0	1	0	1	0	5
all	1	1	1	1	1	0	0	0	1	0	1	0	7
num3d	1	0	0	1	0	0	0	0	0	0	0	0	2
our	1	1	1	1	1	1	0	0	1	1	1	1	10

Continue on the next page

Predictor	Boruta	varSelRF	r2VIM	RFE	RRF	GRRF	GRF	NTA	PIMP	VSURF	AUCRF	RF-SRC	Sum
over	1	1	1	1	1	1	0	0	1	0	1	0	8
remove	1	1	1	1	1	1	1	0	1	1	1	1	11
internet	1	1	1	1	1	0	0	0	1	0	1	1	8
order	1	0	1	1	1	1	0	0	1	0	1	0	7
mail	1	0	1	1	1	1	0	0	1	0	1	1	8
receive	1	1	1	1	0	0	0	0	1	0	1	1	7
will	1	1	1	1	1	1	0	0	1	1	1	1	10
people	1	0	1	1	1	0	0	0	1	0	1	0	6
report	1	0	0	1	0	0	0	0	1	0	1	0	4
addresses	1	0	1	1	0	0	0	0	1	0	0	0	4
free	1	1	1	1	1	1	1	0	1	1	1	1	11
business	1	1	1	1	1	0	0	0	1	1	1	1	9
email	1	1	1	1	1	1	0	0	1	0	1	1	9
you	1	1	1	1	1	1	0	0	1	1	1	1	10
credit	1	1	1	1	1	1	0	0	1	0	1	0	8
your	1	1	1	1	1	1	1	0	1	1	1	1	11
font	1	0	1	1	0	0	0	0	1	0	1	0	5
num000	1	1	1	1	1	0	0	0	1	1	1	1	9
money	1	1	1	1	0	0	0	0	1	0	1	1	7
hp	1	1	1	1	1	1	0	0	1	1	1	1	10
hpl	1	1	1	1	0	0	0	0	1	0	1	1	7

Continue on the next page

Predictor	Boruta	varSelRF	r2VIM	RFE	RRF	GRRF	GRF	NTA	PIMP	VSURF	AUCRF	RF-SRC	Sum
george	1	1	1	1	1	1	0	0	1	1	1	1	10
num650	1	1	1	1	1	0	0	0	1	1	1	0	8
lab	1	0	1	1	1	0	0	0	0	1	0	0	5
labs	1	1	1	1	0	0	0	0	1	0	1	0	6
telnet	1	0	1	1	0	0	0	0	1	0	0	0	4
num857	1	0	0	0	0	0	0	0	0	0	0	0	1
data	1	0	1	1	1	1	0	0	1	0	1	0	7
num415	1	0	0	0	0	0	0	0	0	0	0	0	1
num85	1	0	1	1	1	0	0	0	1	0	1	0	6
technology	1	0	1	1	1	0	0	0	1	0	1	0	6
num1999	1	1	1	1	1	0	0	0	1	1	1	1	9
parts	1	0	0	1	0	0	0	0	0	0	0	0	2
pm	1	0	1	1	0	0	0	0	0	0	1	0	4
direct	1	0	0	1	0	0	0	0	1	0	0	0	3
cs	1	0	0	1	1	0	0	0	0	0	0	0	3
meeting	1	1	1	1	1	1	0	0	1	1	1	1	10
original	1	0	1	1	0	0	0	0	0	0	0	0	3
project	1	0	1	1	1	0	0	0	1	0	1	0	6
re	1	1	1	1	1	1	0	0	1	1	1	1	10
edu	1	1	1	1	1	1	0	0	1	1	1	1	10
table	0	0	0	1	0	0	0	0	0	0	0	0	1

Continue on the next page

Predictor	Boruta	varSelRF	r2VIM	RFE	RRF	GRRF	GRF	NTA	PIMP	VSURF	AUCRF	RF-SRC	Sum
conference	1	0	1	1	0	0	0	0	1	0	0	0	4
charSemicolon	1	0	1	1	1	1	0	0	1	0	1	0	7
charRoundbracket	1	1	1	1	1	0	0	0	1	0	1	1	8
charSquarebracket	1	0	1	1	0	1	0	0	0	0	1	0	5
charExclamation	1	1	1	1	1	1	1	0	1	1	1	1	11
charDollar	1	1	1	1	1	1	1	0	1	1	1	1	11
charHash	1	0	1	1	1	0	0	0	1	0	1	0	6
capitalAve	1	1	1	1	1	1	1	0	1	1	1	1	11
capitalLong	1	1	1	1	1	1	0	0	1	1	1	1	10
capitalTotal	1	1	1	1	1	1	0	0	1	1	1	1	10

Finally, a 10-fold cross-validation is performed for all approaches. The mean and standard deviation of the classification AUC (Area under the Curve), F1-score, and OOB-error rate are reported. Tables 4.11, 4.12, and 4.13 respectively present the results obtained from the three datasets. Each table shows the 10-fold cross-validation AUC, F1-score, OOB-error rate, and the number of features selected by each method.

For the SMR dataset, we see from Table 4.11 that the number of features selected by different feature selection methods vary dramatically. While the NTA method identifies 58 features, the GRF method only includes 4 features. Remarkably, it is noted that the VSURF approach achieves the *highest* F-1 score and the *lowest* OOB-error rate by including the *smallest* number of features (i.e., only 9 out of the 60 features are included to achieve the best performance). VSURF also produces a reasonably high AUC which is only slightly lower than that of PIMP, varSelRF, and

NTA, while the latter three approaches involve way more features.

For the WBC dataset, we obtain from Table 4.12 a similar observation that the number of features selected by different feature selection methods can vary dramatically. While the Boruta and AUCRF approaches retain all 9 features, GRF method only selects 3 features. It is also noted that the r2VIM and NTA approaches are not able to produce valid results (indicated by "NAs") in this table. As already mentioned in Section 4.3, this is because the r2VIM and NTA algorithms lean on negative variable importance created by non-relevant features, and error messages are returned.

For the SE dataset, all methods but NTA are able to produce valid results as shown in Table 4.13. The number of features selected by different feature selection methods again vary dramatically. GRF approach identifies a much smaller number of features. However, it also has the worst overall performance among the 13 approaches. Because all methods (except for GRF) yield similar predictive performance, one naturally prefer those methods that include a smaller feature set features for a better balance of model complexity and model performance, such as VSURF (21 features), GRRF (24 features), and RF-SRC (26 features).

Table 4.11: Summary of RF 10-fold cross-validation using Sonar, Mines vs Rocks dataset

Method	AUC	F1-score	OOB(%)	Selected Features
RF	0.946 ± 0.036	0.824 ± 0.108	15.97 ± 1.17	60
Boruta	0.936 ± 0.040	0.802 ± 0.148	15.01 ± 1.65	38
varSelRF	0.938 ± 0.045	0.809 ± 0.096	14.79 ± 1.44	24
r2VIM	0.936 ± 0.043	0.838 ± 0.096	14.63 ± 1.54	31
RFE	0.929 ± 0.048	0.781 ± 0.137	16.93 ± 1.15	23
RRF	0.923 ± 0.049	0.798 ± 0.106	16.56 ± 1.44	22
GRRF	0.872 ± 0.073	0.777 ± 0.112	20.62 ± 1.97	7
GRF	0.800 ± 0.087	0.703 ± 0.085	26.87 ± 1.47	4
NTA	0.939 ± 0.039	0.796 ± 0.170	15.49 ± 1.65	58
PIMP	0.949 ± 0.037	0.828 ± 0.132	15.11 ± 0.88	48
VSURF	0.937 ± 0.049	0.842 ± 0.063	14.37 ± 1.03	9
AUCRF	0.936 ± 0.045	0.812 ± 0.112	15.06 ± 0.91	29
RF-SRC	0.922 ± 0.043	0.802 ± 0.140	17.25 ± 1.26	20

Table 4.12: Summary of RF 10-fold cross-validation using Wisconsin Breast Cancer dataset

Method	AUC	F1-score	OOB(%)	Selected Features
RF	0.993 ± 0.009	0.963 ± 0.029	2.86 ± 0.33	9
Boruta	0.993 ± 0.009	0.963 ± 0.029	2.86 ± 0.33	9
varSelRF	0.989 ± 0.011	0.949 ± 0.043	3.28 ± 0.53	5
r2VIM	NA	NA	NA	NA
RFE	0.992 ± 0.009	0.961 ± 0.035	2.50 ± 0.34	8
RRF	0.993 ± 0.008	0.963 ± 0.032	2.74 ± 0.35	8
GRRF	0.990 ± 0.011	0.948 ± 0.047	3.28 ± 0.53	6
GRF	0.990 ± 0.010	0.935 ± 0.043	4.52 ± 0.57	3
NTA	NA	NA	NA	NA
PIMP	0.989 ± 0.010	0.942 ± 0.052	3.52 ± 0.43	4
VSURF	0.992 ± 0.009	0.961 ± 0.029	2.89 ± 0.44	7
AUCRF	0.993 ± 0.009	0.961 ± 0.033	2.63 ± 0.25	9
RF-SRC	0.990 ± 0.010	0.927 ± 0.034	4.63 ± 0.61	4

Table 4.13: Summary of RF 10-fold cross-validation using the Spam Emails dataset

Method	AUC	F1-score	OOB(%)	Selected Features
RF	0.986 ± 0.006	0.936 ± 0.010	4.93 ± 0.14	57
Boruta	0.986 ± 0.006	0.935 ± 0.012	4.91 ± 0.10	56
varSelRF	0.986 ± 0.005	0.937 ± 0.010	4.91 ± 0.10	30
r2VIM	0.986 ± 0.006	0.938 ± 0.011	4.91 ± 0.05	48
RFE	0.986 ± 0.006	0.937 ± 0.012	4.95 ± 0.13	55
RRF	0.986 ± 0.005	0.941 ± 0.009	4.71 ± 0.11	38
GRRF	0.985 ± 0.005	0.935 ± 0.010	5.04 ± 0.07	24
GRF	0.957 ± 0.006	0.895 ± 0.010	7.82 ± 0.23	6
NTA	NA	NA	NA	NA
PIMP	0.986 ± 0.005	0.932 ± 0.010	4.82 ± 0.17	46
VSURF	0.986 ± 0.005	0.936 ± 0.010	4.79 ± 0.16	21
AUCRF	0.986 ± 0.005	0.938 ± 0.012	4.76 ± 0.11	45
RF-SRC	0.986 ± 0.005	0.936 ± 0.007	5.02 ± 0.12	26

5 Conclusion

Feature selection is still a hot research topic as datasets with excessive-high dimensionality have been collected recently, especially in gene expression studies. FS advantages lie underneath of removing unimportant and highly correlated features, which reduces the complexity of models, enhances interpretability of the features, and improves the predictive accuracy in practice. The algorithms and strategies of variant wrappers built over RF were fully discussed. In our review, VSURF performed very well while selecting relatively smaller sets of features. However, we make no recommendation here since finding an appropriate FS method highly depends on the application and users' preferences.

To sum up, it has been shown that GRF always selects the smallest sets, but this does impact its predictive performance. NTA and r2VIM suffer from their algorithm foundation. As we mentioned, they are not applicable if they cannot find any negative variable importance. It has been observed that Boruta selects a large number of features and its performance is approximately close to RF itself. Also, it is interesting to see that VSURF offered outstanding predictive performances, although it selects relatively smaller sets of features compared to other approaches.

References

Altmann, A., Toloşi, L., Sander, O., & Lengauer, T. (2010). Permutation importance: a corrected feature importance measure. *Bioinformatics*, *26*(10), 1340–1347.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32.

Calle, M. L., Urrea, V., Boulesteix, A.-L., & Malats, N. (2011). AUC-RF: a new strategy for genomic profiling with random forest. *Human heredity*, *72*(2), 121–132.

Celik, E. (2015). *vita: Variable Importance Testing Approaches* [R package version 1.0.0]. https://CRAN.R-project.org/package=vita

Deng, H. (2013). Guided random forest in the RRF package. *arXiv preprint arXiv:1306.0237*.

Deng, H., & Runger, G. (2012). Feature selection via regularized trees. *The 2012 International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Deng, H., & Runger, G. (2013). Gene selection with guided regularized random forest. *Pattern Recognition*, *46*(12), 3483–3489.

Diaz-Uriarte, R. (2007). GeneSrF and varSelRF: a web-based tool and R package for gene selection and classification using random forest. *BMC bioinformatics*, *8*(1), 1–7.

Dıéaz-Uriarte, R., & Alvarez de Andrés, S. (2006). Gene selection and classification of microarray data using random forest. *BMC bioinformatics*, *7*(1), 1–13.

Fouodo, C. (2022). *Pomona: Identification of relevant variables in omics data sets using Random Forests* [R package version 1.0.2].

Frank, A. (2010). UCI machine learning repository. *http://archive. ics. uci. edu/ml*.

Genuer, R., Poggi, J.-M., & Tuleau-Malot, C. (2015). VSURF: an R package for variable selection using random forests. *The R Journal*, *7*(2), 19–33.

Genuer, R., Poggi, J.-M., & Tuleau-Malot, C. (2019). *VSURF: Variable Selection Using Random Forests* [R package version 1.1.0]. https://CRAN.R-project.org/package=VSURF

Gorman, R. P., & Sejnowski, T. J. (1988). Analysis of hidden units in a layered network trained to classify sonar targets. *Neural networks*, *1*(1), 75–89.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, *3*(Mar), 1157–1182.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, *46*(1), 389–422.

Hopkins, M., Reeber, E., Forman, G., & Suermondt, J. (1999). Spambase data set. *Hewlett-Packard Labs*, *1*(7).

Hua, J., Xiong, Z., Lowey, J., Suh, E., & Dougherty, E. R. (2005). Optimal number of features as a function of sample size for various classification rules. *Bioinformatics*, *21*(8), 1509–1515.

Ishwaran, H., Kogalur, U. B., Gorodeski, E. Z., Minn, A. J., & Lauer, M. S. (2010). High-dimensional variable selection for survival data. *Journal of the American Statistical Association*, *105*(489), 205–217.

Ishwaran, H., Kogalur, U. B., & Kogalur, M. U. B. (2022). Package 'randomForestSRC'. *breast*, *6*, 1.

Janitza, S., Celik, E., & Boulesteix, A.-L. (2018). A computationally fast variable importance test for random forests for high-dimensional data. *Advances in Data Analysis and Classification*, *12*(4), 885–915.

Jirapech-Umpai, T., & Aitken, S. (2005). Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes. *BMC bioinformatics*, *6*(1), 1–11.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial intelligence*, *97*(1-2), 273–324.

Kuhn, M. (2020). *caret: Classification and Regression Training* [R package version 6.0-86]. https://CRAN.R-project.org/package=caret

Kursa, M. B., & Rudnicki, W. R. (2010). Feature selection with the Boruta package. *Journal of statistical software*, *36*, 1–13.

Kursa, M. B., & Rudnicki, W. R. (2011). The all relevant feature selection using random forest. *arXiv preprint arXiv:1106.5112*.

Lee, J. W., Lee, J. B., Park, M., & Song, S. H. (2005). An extensive comparison of recent classification tools applied to microarray data. *Computational Statistics & Data Analysis*, *48*(4), 869–885.

Liu, H., Liu, L., & Zhang, H. (2010). Ensemble gene selection for cancer classification. *Pattern Recognition*, *43*(8), 2763–2772.

Ruiz, R., Riquelme, J. C., & Aguilar-Ruiz, J. S. (2006). Incremental wrapper-based gene selection from microarray data for cancer classification. *Pattern Recognition*, *39*(12), 2383–2392.

Speiser, J. L., Miller, M. E., Tooze, J., & Ip, E. (2019). A comparison of random forest variable selection methods for classification prediction modeling. *Expert systems with applications*, *134*, 93–101.

Szymczak, S., Holzinger, E., Dasgupta, A., Malley, J. D., Molloy, A. M., Mills, J. L., Brody, L. C., Stambolian, D., & Bailey-Wilson, J. E. (2016). r2VIM: A new variable selection method for random forests in genome-wide association studies. *BioData mining*, *9*(1), 1–15.

Urrea, V., & Calle, M. (2012). *AUCRF: Variable Selection with Random Forest and the Area Under the Curve* [R package version 1.1]. https://CRAN.R-project.org/package=AUCRF

Wang, H., & Li, G. (2017). A selective review on random survival forests for high dimensional data. *Quantitative bio-science*, *36*(2), 85.

Wolberg, W. H., & Mangasarian, O. L. (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the national academy of sciences*, *87*(23), 9193–9196.

Zhang, J. (1992). Selecting typical instances in instance-based learning. *Machine learning proceedings 1992* (pp. 470–479). Elsevier.

Zhu, Z., Ong, Y.-S., & Dash, M. (2007). Markov blanket-embedded genetic algorithm for gene selection. *Pattern Recognition*, *40*(11), 3236–3248.

6 Summary

In machine learning, numerous methods have been developed and are being used in a wide variety of applications. However, the vital role these models play in the healthcare sector's developments and the importance of improving diagnosis and prediction accuracy has prompted this dissertation. Among the machine learning methods, ensemble models have become one of the most successful statistical learning approaches. Despite existing sophisticated ensemble models, we could extend the depth of this branch of modeling with our two proposed boosting trees models. In this dissertation, we introduced Boost-S, a gradient boosted trees algorithm for spatial data with covariate information, and AdaBoost.S, a structured adaptive boosting trees algorithm for edge detection problems.

It has been shown that Boost-S successfully integrates spatial correlation into the classical framework of gradient boosted trees. A computational-efficient algorithm as well as the technical details have been presented. The Boost-S algorithm grows individual trees by solving a regularized optimization problem, where the objective function involves two penalty terms on tree complexity and takes into account the underlying spatial correlation. Boost-S performance and advantages were assessed by applying based on the real datasets involving the spatial-correlated FDG-PET imaging and comparing to six other commonly used approaches.

We also described in detail the feature extraction process from image patches and presented the technical details of how the extracted features can be used to predict the edge structures which are highly interdependent. AdaBoost.S capabilities and predictive accuracy was evaluated based on real datasets of fluorescence IVM images for edge detection of platelet-neutrophil aggregates within pulmonary microcirculation of live mice exposed to e-cigarette vapor. Our comparison studies have demonstrated the advantages of the proposed AdaBoost.S over existing methods.

In the end, as datasets with excessive-high dimensionality have been collected recently, feature selection has become a hot research topic. Therefore, we gave a comprehensive review and compared different random forest based feature selections for classification problems. We showed that

GRF always selects the smallest sets, but this does impact its predictive performance. NTA and r2VIM are not applicable to all datasets since their algorithms lean on producing negative variable importance. Also, it has been observed that Boruta selects a large number of features and its performance is approximately close to RF. It was interesting to see that VSURF offered outstanding predictive performances while selecting relatively smaller sets of features compared to other approaches. Finally, We found that a good FS method highly depends on the application and user's preferences