Patient-Specific Prediction with Bayesian Personalized Decision Paths

by

Adriana Louise Jurchak Johnson

Bachelor of Arts, Bard College, 2012

Master of Science, University of Pittsburgh, 2020

Submitted to the Graduate Faculty of the

School of Medicine in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2021

UNIVERSITY OF PITTSBURGH

SCHOOL OF MEDICINE

This dissertation was presented

by

Adriana Louise Jurchak Johnson

It was defended on

July 23, 2021

and approved by

Gregory F. Cooper, MD, PhD, Professor, Biomedical Informatics

Harry Hochheiser, PhD, Associate Professor, Biomedical Informatics

Gwendolyn A. Sowa, MD, PhD, Professor, Physical Medicine and Rehabilitation

Dissertation Director: Shyam Visweswaran, MD, PhD, Associate Professor, Biomedical Informatics

Patient-Specific Prediction with Bayesian Personalized Decision Paths

Adriana Louise Jurchak Johnson, PhD

University of Pittsburgh, 2021

Machine learning algorithms can be useful in predicting patient outcomes under uncertainty. Many algorithms employ "population" methods to optimize a single, static model to predict well on average for an entire population, but such models may perform poorly for patients who differ greatly from the average patient or majority of the population. Personalized methods seek to optimize predictive performance for every patient by tailoring a patient-specific model to each individual. Prior work on personalized methods includes clustering methods like k-nearest neighbor and tree-derived methods like decision paths.

It has been shown in multiple domains that ensembles of decision trees often outperform single decision tree models by reducing variance, capturing a range of significant features, and mitigating the uncertainty involved in model selection. However, ensemble methods have been used only sparingly in the context of personalized models. The use of Bayesian scoring in model construction has also been shown to improve predictive performance of decision tree and decision path algorithms.

In this dissertation, we developed and evaluated several novel personalized decision path methods – including methods that construct single personalized decision paths as well as methods that construct ensembles of paths that use Bayesian scoring in the form of personalized random forest and personalized boosted trees. We found that the use of a random forest ensemble approach was associated with improvements to the predictive performance of personalized decision paths in terms of discrimination and calibration, and the use of Bayesian scoring was associated with improvements to the predictive performance of personalized decision paths, decision trees, and

random forest ensembles of decision trees. However, we did not observe a global performance benefit from using personalization, ensemble approaches, and Bayesian scoring together compared to corresponding population and non-Bayesian algorithmic methods.

Table of Contents

List of Tables

List of Figures

Glossary

Variable: An attribute that can describe an entity (for example, hair color). Variables are denoted with upper case letters, such as X.

Value: A specific measurement for a variable (for example, two possible values for hair color are blonde and black). Values are denoted with lower case letters, such as x.

Feature: A variable with an assigned value (for example, hair color = black). Features are denoted with the variable and assigned value, such as $X = x$.

Target: The variable whose value supervised machine learning algorithms seek to predict by modeling its relationships with other variables of the training data.

Predictor: A variable of the training data which, along with its values, is used in models by supervised machine learning algorithms to predict the target variable.

Training case: A sample, patient, or individual from a training dataset whose features and target value are known.

Test case: A sample, patient, or individual whose features are known but whose target is unknown. Supervised machine learning methods aim to produce models that can predict a test case's target value.

Training data: A dataset of training cases, described by the n variables $\boldsymbol{V} = (X_1, X_2, \ldots, X_i, \ldots, X_n)$ and the target $T = (t_1, t_2, \ldots t_k, \ldots, t_r)$.

Eager method: An algorithmic method that constructs a predictive model from training data prior to encountering a test case.

Lazy method: An algorithmic method that waits to encounter a test case prior to constructing a predictive model from training data.

Population method: An algorithmic method that constructs a predictive model by optimizing predictive performance on average for all future cases.

Personalized method: An algorithmic method that constructs a predictive model optimized for the test case, often by using information in the test case (other than the target value).

Population model: A predictive model produced using a population method.

Personalized model: A predictive model produced for a test case using a personalized method.

Patient-specific model: A personalized model where the training and test cases are patients.

AUROC: Area under receiver operating characteristic. A measure of discrimination, or the ability of a model to differentiate between positive and negative cases.

ECE: Expected calibration error. A measure of calibration, or the ability of a model to produce class probability predictions that correctly represent uncertainty.

MPL: Mean path length (of path and tree models). A measure of model complexity, or the number of predictor variables used by a model to make a prediction.

Preface

I would like to acknowledge and thank the following people without whose support I could not have completed this dissertation:

- My advisor, Dr. Shyam Visweswaran, for his patient instruction, steady calmness, and for countless thoughtful conversations which opened my eyes to previously unknown possibilities within the world of biomedical informatics;
- My dissertation committee members, Dr. Gregory Cooper, Dr. Harry Hochheiser, and Dr. Gwen Sowa, for their mentorship, guidance, and support;
- Toni Porterfield for her endless encouragement and assistance;
- My teachers and mentors, Carl Gersten, Craig Friedland, Dr. John Cullinan, Dr. Samuel Hsiao, Dr. Eric Moulton, and Dr. David Borsook, for encouraging me to keep pursuing new academic challenges;
- The National Library of Medicine and the University of Pittsburgh Medical Scientist Training Program for funding and facilitating my training;
- My family and friends, with special thanks to:
    - My friend and fellow informatician Lauren Rost, for holding me accountable and helping me be kind to myself;
    - My MSTP and DBMI friends, Gaelen Dwyer, Laura Molina, Maryanna Owoc, and Michael Ding, for accompanying me on the marathon of MD-PhD training;
    - My DBMI colleagues Amin Tajgardoon and Chandra Rathnam for their help with my code;
    - My sister, Dr. Arielle Johnson, for being my cool scientist older sibling role model;

- My parents for raising me to love learning new things, grapple with difficult questions, and to not shy away from challenges;
- My mother-in-law, Lisa, for taking such good care of me;
- My family and friends for cheering me on every step of the way;
- My daughter, Margaret, for her presence and joy;
- And most especially thanks to my husband, Sam, for helping me endure many hard days and celebrate the good ones. When I told him eight years ago that I thought I might want to do MD-PhD, he asked, "Why would *anyone* want to do that?" After I explained what I hoped to learn, he supported me completely – moving to Pittsburgh with me, taking care of me while I toiled away at my studies, listening to my complaints and fears and frustrations, and never once doubting that I could do it (even though I often doubted myself). I dedicate this work to him.

1.0 Introduction

Prediction is critical to many activities in clinical medicine. This can include determining the presence or absence of disease, assessing risk of disease development in the future, and forecasting the likely course of a disease. This is important on an individual level for clinical medicine and on a population level for public health and biomedical research. Predictive models that are derived from biomedical data can be used to improve predictions by mitigating uncertainty and guide decision-making in clinical and public health contexts. Many predictive models are probabilistic and derive models from data using statistical and machine learning methods. Improving predictive models can help improve decision making and ensuing outcomes, potentially benefitting both individual patients and the population at large. Even small improvements in predictive performance can have meaningful impact on individual and public health outcomes and costs.

In this chapter, we introduce two different approaches for learning predictive models. The first is the use of a population method, which seeks to produce models that perform well for the population on average. The second is the use of personalized method, which seeks to a produce a patient-specific model that performs well for a patient of interest, customizing a model to each individual. We then describe our goals in developing novel patient-specific approaches and discuss their significance and innovation.

1.1 Overview of Prediction in Medicine

Clinical medicine requires accurate prediction of a patient's risk of different outcomes, and this task can be supported by predictive models. Machine learning methods can be used to produce predictive models which may improve patient care through high-quality predictive performance. Most machine learning methods produce a *population model* which is optimized to perform well on average over an entire population of future patients. This approach does not ensure optimal performance for every member of the population, however. Population methods can result in suboptimal predictions for certain patients, such as those who do not resemble the average member of a population. Use of population models may disadvantage certain minority subgroups and could negatively impact care and outcomes for such patients.

An alternate approach when seeking to make a prediction for a patient of interest is to produce a personalized predictive model tailored to that individual. Such methods result in a *patient-specific model* which is optimized to predict well for the patient of interest. Personalized methods may result in improved predictive performance and could ameliorate disparities in predictive performance between subgroups of a population, potentially improving outcomes for a broader range of patients.

Our group has previously developed several personalized methods based on decision trees called personalized decision paths. To construct a personalized decision path, the algorithm performs a search guided by features present in the patient of interest to generate a patient-optimized predictive model. Personalized decision path methods have demonstrated superior predictive performance compared to population methods like decision trees. However, there

remains room for improvement in regard to the overall predictive performance of these patient-specific models.

Predictive performance can sometimes be improved through the use of ensemble methods where multiple models are generated and their outputs are aggregated. There is always uncertainty when selecting a single "best" model based on limited data, and using ensembles can help mitigate that uncertainty. Multiple approaches have been developed to construct ensembles, including bagging, random forest, boosting, and Bayesian model averaging. In particular, boosting and random forest often outperform single decision trees and are popular in biomedicine. Predictive performance can also be improved with the use of Bayesian scoring methods, which incorporate smoothing and regularization to model uncertainty. In order to try to improve predictive performance of personalized decision paths, we have developed a variety of ensembles consisting of Bayesian personalized decision path models for patient-specific prediction. These methods were evaluated on real clinical datasets to predict presence of disease as well as morbidity and mortality.

We hypothesize that by combining Bayesian scoring, personalized methods, and ensemble approaches, performance is improved compared to entropy-scored single personalized models and population ensemble approaches as measured by area under receiver operating characteristic (AUROC) and expected calibration error (ECE), and that these new methods result in simpler models as modeled by mean path length (MPL). To evaluate this hypothesis, single Bayesian personalized decision paths and ensembles of Bayesian personalized decision paths constructed via random forest and boosting approaches were compared to ensembles of entropy-scored personalized decision paths, population decision trees, and single personalized decision paths, and predictive performance was evaluated on a variety of synthetic and real clinical data.

1.2 Aims of the Dissertation

The main goal in this dissertation is to introduce new Bayesian personalized decision path algorithmic methods and evaluate whether they yield better performance than commonly used population-wide methods.

Aim 1: Development of personalized Bayesian methods. Development was performed using synthetic data and employed single model, random forest, and boosting approaches with personalized decision paths that use Bayesian scores as the base model.

Aim 2: Assessment of predictive performance of personalized Bayesian methods using clinical datasets. The primary metric of predictive performance was discrimination as measured by AUROC. Additional measures included calibration (measured by expected calibration error) and model complexity (measured by mean path length). Patient-specific methods were compared to population methods based on decision trees.

1.3 Significance of Aims

Clinicians have long used diagnostic scores, risk scores, and predictive models to assist in patient diagnosis, prognosis, and risk assessment.[1–4] Clinical prediction can be further aided by leveraging patient data using computational methods. Automated learning algorithms can identify complex patterns in large datasets, producing sophisticated predictive models for use in patient care.[5] Many such methods have been developed, including applications to diagnosing retinopathy,[6] predicting severe sepsis,[7] diagnosing breast cancer,[8] and predicting response to chemotherapy.[9]

Some of these methods have been able to match clinician performance on classification tasks, and one study demonstrated that clinical implementation of a machine learning derived model for predicting sepsis risk was associated with improved patient outcomes.[7]

Most current machine learning methods are "population" methods, which fit a model to predict well on average for future cases in the population. For some medical problems, especially those associated with highly heterogenous patient populations, population methods may not be able to achieve high predictive performance.[10] Yet even when average performance is adequate, population methods may not model optimally for every individual in the population. Population methods may fail to capture significant patterns that only occur rarely or only in a minority of patients.[11] This can result in less informative predictions for individuals with uncommon features, such as rare genomic variants or members of minority subgroups, which could result in poorer health outcomes for these patients.[12]

These issues can possibly be ameliorated by the use of personalized methods, which produce a model tailored to the individual for whom a prediction will be made. Using personalized methods may help avoid some of the biases that occur in population methods. Personalization can come in several forms, including using a patient's own data to train a model or using a similarity metric to curate a training dataset of cases similar to the patient of interest which can then be used to train a model.[13] These approaches are limited, however; patient-sourced data is not available for many clinical prediction tasks (like mortality risk prediction), and models produced from personalized training datasets using patient similarity metrics may fail to capture uncommon features that are significant to the patient of interest.

Personalized methods have thus been developed that use patient information to guide feature selection and model fitting itself. The personalized decision path is an example of this

approach and is based on decision trees.[14] Our group has developed novel personalized decision path methods for creating patient-specific models, which have demonstrated superior predictive performance over population methods like decision trees for a variety of clinical prediction tasks.[15–18] However, we anticipated that predictive performance of our personalized methods could be further improved through ensemble approaches and the use of Bayesian scoring.

1.4 Innovation of Aims

We anticipated that the use of Bayesian scoring as well as and random forest and boosting ensemble approaches could improve predictive performance of personalized decision paths. Ensemble approaches have rarely been used in combination with personalized methods, but the few studies that exist have produced promising results.[19,20] This investigation into ensembles of patient-specific decision paths is the first of its kind and introduces multiple novel methods for performing personalized modeling.

1.5 Overview of Dissertation

The remainder of the dissertation is structured as follows. Chapter 2 provides relevant background on machine learning for clinical prediction, personalized machine learning methods, decision trees and decision paths, and ensemble approaches. Chapter 3 introduces the algorithmic methods for both the novel and comparison algorithms. Chapter 4 describes in detail the experimental methods used to evaluate predictive performance, including the datasets, performance

metrics, and statistical analysis used. Chapter 5 presents the results of our experiments. Chapter 6 contains the discussion of our findings, including limitations and possibilities for future areas of inquiry. Chapter 7 presents our main conclusions.

2.0 Background

In this chapter, we provide background on prediction in the clinical context, machine learning in medicine including patient specific predictive models, population-wide decision trees and personalized decision trees, and different types of ensemble approaches.

2.1 Prediction in Clinical Context

Clinicians are frequently required to make judgements in situations that involve uncertainty. It may not be clear what disease a patient is experiencing given their symptoms, what treatment might be most successful for a patient given their diagnosis, or what a patient's future risk of morbidity and mortality might be given their current presentation. Clinicians must make decisions regarding patient care despite this uncertainty. By improving the accuracy of their assessments, clinicians can help optimize outcomes for their patients.[21]

The process of identifying which disease most likely corresponds to a patient's presenting symptoms is called diagnosis. Depending on the diagnosis, different tests, treatments, and recommendations will be made to the patient to manage the disease. Incorrect diagnosis can result in the use of unnecessary and potentially invasive tests as well as ineffective or hazardous treatments.[22] It can also delay the introduction of effective treatment, potentially resulting in worsening of disease and prolonged suffering and stress for the patient.[23]

The characterization of a patient's likely disease course and possible outcomes given different interventions is called prognosis. Depending on the patient's prognosis, different therapeutic options can be offered. Prognosis is not dependent on diagnosis alone: two patients with the same diagnosis may have different probable disease courses given their underlying health conditions or may react differently to the same medication due to genetic differences.[24] Accurate prognosis allows for targeted interventions which can improve patient outcomes, but inaccurate prognosis can have a negative impact on treatment planning, patient quality of life, and result in misallocation of resources.

The identification and stratification of a patient's future possibility of morbidity and mortality given their current state and exposures is called risk assessment. Although a patient may not have a specific disease at the present time, identifying a patient's risk for developing disease in the future can help ensure appropriate monitoring, timely diagnosis, and efficient use of resources.[25] Incorrect risk assessment can lead to stress for the patient, inappropriate resource allocation, and missed opportunities for intervention to prevent severe morbidity and mortality.

Oftentimes clinicians rely on their education and experience in making these predictions and determinations, but they may also use tools which have been developed to assist in reducing uncertainty when making clinical decisions. Medical scoring systems and scales have long been used to aid in the evaluation of a patient's current state and prediction of future status. One of the first such measures to achieve widespread use was the Apgar Score, introduced in 1953 by Dr. Virginia Apgar to assess the health of newborns and their likelihood of requiring resuscitation.[2] Other longstanding examples include APACHE, a scoring system for assessing severity of illness and risk of death in intensive care unit patients,[1] the Beck Depression Inventory, a scale for diagnosing and assessing severity of depression,[3] and the Framingham coronary heart disease risk

score.[4] Such tools assign numerical values to objective findings which can then be combined to produce an overall risk score. Together, the selected clinical findings and corresponding numerical weights can be considered a clinical prediction model.[26]

Clinical prediction models can be constructed manually by expert clinicians. Predictor variables from a dataset are selected based on a combination of previously reported findings and the clinician's own expertise,[27] and these predictors are then statistically evaluated to determine their relationship with the outcome of interest. These results can then be used to determine predictor weights for the model. This process requires expensive resources (e.g. the time of expert clinicians) and limits the complexity of the models, as human cognitive capacity allows one to consider the combined effects of only a limited number of predictors.[5]

Technological advances and work in artificial intelligence have allowed for the automated construction and evaluation of more complex models that consider the influence and interactions of many more predictors at much lower cost than expert-derived systems.[28] These machine learning approaches have provided new ways of developing clinical prediction models.

2.2 Machine Learning in Medicine

Advances in machine learning methodologies have allowed for the development of sophisticated models with the ability to match the performance of trained physicians.[8,29,30] These predictive models are learned from patient data, of which there is an ever-increasing volume thanks to electronic health records,[31] next generation sequencing,[32] wearable biosensors,[33] and research databases.[34] The algorithms which produce these models are able to identify clinically useful

patterns that would be unidentifiable by humans alone. With the emphasis on evidence-based practice in contemporary medicine, such models can be useful in providing data-sourced clinical guidance to practitioners.

Many different types of machine learning methods have been applied to clinical problems. Unsupervised machine learning is used on unlabeled data for tasks like clustering to characterize the bacterial composition of the gut microbiome, and such methods do not produce a predictive model.[35] By contrast, supervised machine learning methods are used with "labeled" data for which a target outcome is known, such as clinical data that includes the presence or absence of a disease in each patient. When performing supervised learning on data with a categorical outcome, the task is called classification.[36] Machine learning classifiers take a dataset of training cases with labeled target outcomes, learn patterns in the data associated with the target outcome by identifying meaningful predictors and fitting parameters, and produce a model which can then be used to predict the target outcome for unlabeled test cases.[37]

A variety of methods have been successfully applied to clinical classification problems. Esteva et al.[29] presented the results of using a convolutional neural network to classify skin cancer from patient images. The model's performance in classifying malignant versus benign skin cancer was found to be similar to that of 21 board-certified dermatologists. Gulshan et al.[6] demonstrated that a convolutional neural network could be trained to classify diabetic retinopathy from retinal fundus photographs, and Ting et al.[30] showed that such a classification system could have comparable sensitivity to trained human graders, including ophthalmologists. Shimabukuro et al.[7] showed in a randomized controlled trial that a proprietary machine learning algorithm used for prediction of severe sepsis[38] resulted in a significant decrease in hospital length of stay as well as lower rates of in-hospital mortality. Ardekani et al.[39] showed that a random forest could use MRI

data from patients with mild cognitive impairment to identify those at high risk for developing Alzheimer's disease. Huang et al.[40] showed that a support vector machine classifier could predict a patient's response to chemotherapy based on tumor gene expression profiles. Ding et al.[9] demonstrated that a deep learning system could be used to accurately predict the efficacy of anti-cancer drugs. McKinney et al.[8] demonstrated that an ensemble of neural network models could classify breast cancer from mammography screenings with greater sensitivity and specificity than radiologists. Artzi et al.[41] demonstrated that using gradient boosting with decision trees on electronic health record data from early pregnancy allowed for accurate prediction of the development of gestational diabetes. These are a small selection of the numerous studies describing effective clinical prediction models produced by machine learning methods.

One aspect shared by the machine learning methods discussed above is that all are *population methods*. Population methods produce a fixed model (or collection of models), optimized to predict well on average for all future cases from the population. Population methods are often "eager learners," producing a population model from training data prior to encountering a "test case," which is an individual for whom a prediction will be made. The average performance of robust population models can be quite good, as seen in the previously described applications to clinical problems. However, there may always be patients for whom a population model does not predict well. To address the limitations of population methods, personalized methods have been developed as an alternative approach.

2.3 Patient-Specific Models

Personalized machine learning methods use known information about the test case to influence the generation of the predictive model. Personalized methods are often "lazy learners," in that these methods wait to produce a predictive model until a test case is encountered.[14] When used for clinical prediction tasks, the resulting predictive model is called a patient-specific model.

Patient-specific models can be generated by personalizing different aspects of the machine learning process. For machine learning classifiers, this process includes using a labeled training dataset, identifying meaningful predictors, and fitting a model to predict the target outcome. Thus, one approach for producing patient-specific models is to personalize the training dataset.

A simple option for personalizing the training dataset is to exclusively use the test patient's own data for training the model. Such an approach has been used to predict seizures based on the patient's own EEG recordings[42] as well as for predicting heart failure by estimating "remaining useful life" of a patient's cardiovascular system using the patient's own physiologic data.[10] One limitation of using exclusively patient-sourced data is that it can be difficult to obtain sufficient data to train a model. The seizure prediction study required patients to have EEG recordings of multiple seizures, which is costly and time intensive. An additional consequence of using only patient-sourced data rather than including data from similar patients is loss of statistical power.[43] Lastly, patient-sourced training data is simply impossible to obtain for many clinical tasks, such as the problem of trying to predict a patient's risk of mortality. Some problems require learning from data that is sourced from other patients.

An alternative approach is to identify a subgroup of training cases in the training dataset who are similar to the test patient, and then train a predictive model using this subgroup as a

personalized training dataset. One way to identify such a subgroup is by using a patient similarity metric (PSM).[13] A PSM allows for quantification of each training case's degree of similarity to the test patient. These similarity scores can then be used in a variety of ways to personalize the training dataset. The k-Nearest Neighbor (kNN) method selects a cohort of the *k* most similar training cases to the test patient, and the most common target outcome among these training cases is provided as the predicted class for the test patient.[44] The kNN approach does not produce a model, however. Alternatively, the cohort of the *k* most similar patients can also be used as a personalized training dataset to fit a predictive model using a population method like a decision tree[45] or logistic regression.[46] Several forms of PSM have been explored for patient-specific prediction, including Euclidean distance,[47] Malahanobis distance,[46] cosine similarity,[45] and random forest proximity.[48]

Many PSMs weigh each predictor in the training dataset equally when assessing similarity. It is not necessarily the case, however, that all predictors are equally useful in predicting the target outcome.[49] Feature selection can be performed to eliminate less meaningful predictors or weigh features by relative importance,[50] but these weights are calculated using population approaches. Unfortunately, the same predictors may not be equally meaningful for different patients.[51] Furthermore, the algorithms used to fit models on personalized training datasets are still population methods. Thus, the use of PSMs to curate training datasets may fail to capture patterns that are significant for the patient of interest.

Therefore, another option for producing patient-specific models is to personalize the selection of predictors and/or fitting of the model. One example of such an approach is the personalized regression method developed by Lengerich et al.,[52] which constructs personalized logistic regression models for each training case in the training dataset using a distance-matching regularizer and performs patient-specific prediction for a test case by averaging the model

parameters of the *k* nearest training cases to the test case. This patient-specific method was applied to successfully predict cancer status across a wide variety of cancer types.[43,52,53] Some machine learning methods, such as decision trees, perform predictor selection and model fitting simultaneously, and personalization of these concurrent processes allows for both patient-specific predictor selection and patient-specific model fitting. Decision trees have been adapted to produce personalized models called decision paths.

2.4 Decision Trees and Decision Paths

In order to describe personalized decision paths, we first must describe the population decision tree approach from which they are derived. In the following section, we describe the model structure and search strategy of decision trees as well as shortcomings due to the population approach that decision trees use. We then describe personalized decision path models and different methods that have been developed to construct them.

2.4.1 Decision Trees

Decision trees have been used since the 1960s for predicting clinical outcomes, including psychiatric diagnosis, cancer survival, mortality of intensive care unit patients, and hospital readmission risk.[54–56] The decision tree model consists of branching interior nodes that represent predictor variables and (terminal) leaf nodes that represent parameters of the predictive probability distribution of the target variable, such as a clinical outcome.[57] Each interior node partitions the

dataset into distinct subgroups according to the values the predictor variable can take, which are represented as branches of the tree. We differentiate between a variable and a feature; a feature is a variable – value pair. For example, if variable *X* denotes a history of angina and takes values absent and present, then *X* = absent and *X* = present are features. Note that there are two distinct features that correspond to a single binary variable *X*, and a patient will have only one of the two features for *X*. A path in the tree from the root node to a leaf node represents a conjunction of features, and the parameters in the leaf node are estimated from the known outcomes of cases in the training set whose features match the features in the corresponding path. To perform inference for a test case using a decision tree, a path in the tree is identified such that all of the features in the path match the features of the test case, and the parameters in the leaf node of the path specify the probability distribution of the target variable. Each test patient will have only one applicable path in the tree, and each path represents a distinct partition of the dataset.

Decision trees are often considered well-suited for clinical decision support in part due to their interpretability, which is an important aspect of clinician trust in a machine learning method.[58] The explanation for a prediction by a decision tree is given by the collection of features in the path, which can be presented as an if-then statement.[59] Decision trees are often cited as the canonical interpretable machine learning method, as these decision rules can be much easier to understand than the weights of a neural network.[60] However, decision tree interpretability is dependent on the number of features and the complexity of the model, as well as the context of interpretation.[61] Global interpretability of a model refers to whether a user can comprehend the population-wide conditional distribution of predictions given all possible input values.[62] Decision trees with large numbers of paths and features can be difficult to interpret on a global level. Local interpretability is the degree to which a single prediction can be explained,[60] which for decision trees depends on

the number of features in a path, with shorter paths being easier to interpret. However, with the development of post-hoc, model agnostic explainability measures that can produce explanations for predictions from any type of model (including black-box methods), model-specific interpretability may become less important.[63,64] Physician evaluations of model explanations indicate agreement with post-hoc explanations and a preference for model-agnostic explanations over model-specific explanations.[58,65,66] Increasing regulatory requirements[67] for machine learning methods to be able to explain their predictions, however, might make the model-specific local explainability of decision trees desirable. However, complex decision tree models comprised of lengthy paths will not serve the needs of clinician users in terms of interpretability.

As it is computationally intractable to exhaustively search the model space of all possible entire decision trees, decision tree methods rely on heuristic search to derive a locally optimal model from a dataset. Many decision tree methods use a greedy search called recursive splitting to select predictors that optimize a specified scoring criterion. At each step, every candidate variable is scored using the criterion, and the best-scoring variable is added to the tree, splitting the variable space into distinct branches based on the values the variable can take. For each branch, the next best-scoring variable is then identified; this process is repeated until a stopping condition is met.

The score of each candidate predictor is calculated using its leaf nodes. A candidate predictor defines a new partition of the dataset, resulting in a new set of candidate leaf nodes defined by the possible values the candidate predictor can take. Each candidate leaf node is scored using its probability distribution over the target, and the candidate predictor score is calculated as a weighted average of the scores of its leaf nodes. The weight is based on the proportion of training cases that correspond to each candidate leaf node. The score is commonly related to impurity,

where the best score is achieved when all cases corresponding to a leaf node share the same target value. Decision tree methods differ in the scoring criterion used: the Classification And Regression Tree (CART) uses the Gini index,[68] while the Interactive Dichotomizer 3 (ID3) and the C4.5 use entropy.[57]

Due to the use of the weighted average, the candidate predictor score can be heavily influenced by leaf nodes that correspond to values present in a larger proportion of the training dataset, and this influence can overwhelm the contributions to the score from smaller leaf nodes. Table 1 lays out the possible implications of this influence. Take a binary candidate variable X that has two candidate leaf nodes representing the two values (x_1 and x_2) that X can take. If a larger proportion of cases have feature $X = x_1$, the score of that leaf node may dominate the score of X.

Table 1. Possible tradeoffs in population decision tree methods.

Binary Candidate Predictor X		If leaf node $X = x_1$ has a large proportion of corresponding cases and a:	
		High Score ($X = x_1$ is informative)	Low Score ($X = x_1$ is uninformative)
If leaf node $X = x_2$ has a small proportion of corresponding cases and a:	High Score ($X = x_2$ is informative)	• X will have a high average score • Both features are informative (no tradeoff)	• X will have a low-medium average score • Model may exclude informative feature $X = x_2$
	Low Score ($X = x_2$ is uninformative)	• X will have a high-medium average score • Model may include uninformative feature $X = x_2$	• X will have a low average score • Both features are uninformative (no tradeoff)

This can lead to tradeoffs in selecting predictors for the model, resulting in the inclusion of uninformative features or exclusion of meaningful features that correspond to smaller proportions of the training dataset. With each addition of a variable, the training data is split into smaller subgroups, and this fragmentation of the data reduces the statistical support for predictions.[69] If a given variable, added to a path in the tree, is not informative on one of its values, then the addition of the uninformative feature adds to the complexity of the model while simultaneously reducing the statistical support for that path's prediction. Personalized decision path methods eliminate these tradeoffs by optimizing only the score of the feature that corresponds to the test case.

2.4.2 Personalized Decision Paths

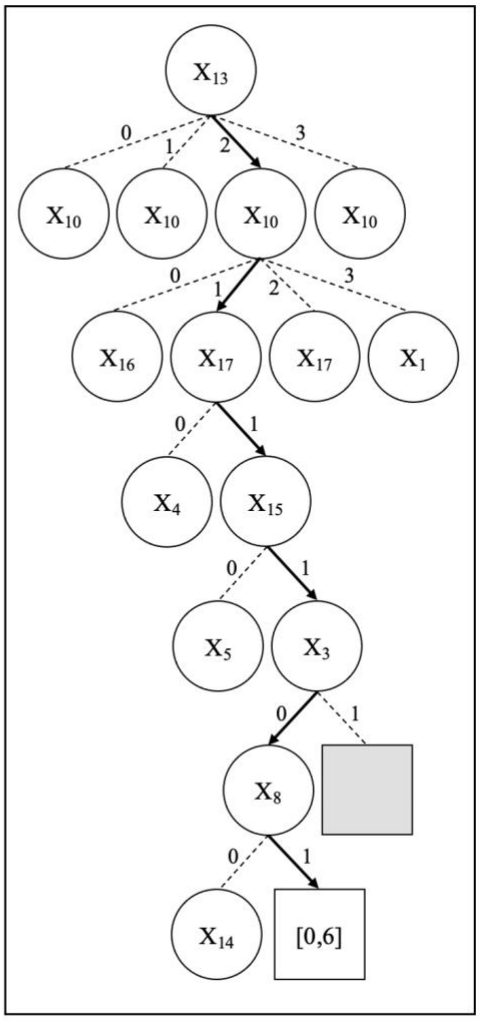
As mentioned previously, a decision tree is a collection of paths, and a decision tree algorithm seeks to identify the collection of paths that result in the highest average score. Since a test case only corresponds to one path in the decision tree, a decision path algorithm attempts to identify the highest scoring individual path for that test case. This personalized path, which consists of a collection of features, corresponds to a subgroup of the training dataset of training cases which share all features with the path. The decision path method attempts to identify an appropriate subgroup for the test case by identifying high-scoring features from the test case. In the process of searching those features, the algorithm constructs a predictive model defined by and tailored to the subgroup that shares the features in the path.

Like a decision tree, a decision path model is also derived from a dataset using greedy search and optimizes a scoring criterion. Unlike a decision tree, the decision path model consists

a.

Variables		Test Case Values
X_1	Sex	0
X_2	History of angina	0
X_3	History of coronary artery disease	0
X_4	History of lung disease	1
X_5	History of diabetes	0
X_6	History of PTCA	0
X_7	Pleural effusion	1
X_8	Pulmonary congestion	1
X_9	Positive finding on EKG	0
X_{10}	Systolic blood pressure	1
X_{11}	Serum potassium	0
X_{12}	Serum sodium	1
X_{13}	Serum BUN	2
X_{14}	Serum creatinine	1
X_{15}	WBC count	1
X_{16}	Arterial pH	1
X_{17}	Respiratory rate	1

b.



c.

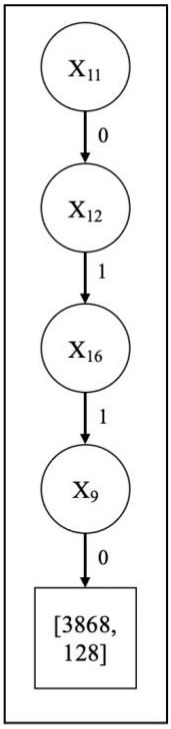


Figure 1. An example of a decision tree and a personalized decision path for predicting in-hospital mortality for a patient admitted with heart failure.

Panel (a) lists the training variables and corresponding values for a patient (test case) whose outcome we want to predict. Panel (b) shows a decision tree and the path (arrows in bold) used for inference for the patient, and panel (c) shows a personalized decision path (derived by the PDP-Bay method) for the patient. Terminal leaf nodes contain counts of corresponding training samples who survived or died during hospitalization. The patient in this case survived.

of a single path (rather than a collection of paths) whose features are shared by the test case. The search proceeds by extending the path by appending one feature (rather than a variable as in the tree) at a time from the test case that optimizes the scoring criterion.

Figure 1 illustrates the difference between a decision tree model and a decision path model for predicting in-hospital mortality for a patient admitted with heart failure. Panel (a) lists the variables and corresponding values for a patient (test case) whose outcome we want to predict. Panel (b) shows a decision tree and the path (arrows in bold) with features that are present in that patient, and panel (c) shows a personalized decision path (derived by the PDP-Bay method that is described in the next chapter) with features from the patient. For the current patient, the decision tree model contains many paths, including a path with features that are present in that patient, and the decision path model contains just one path. Terminal leaf nodes contain counts of corresponding training cases who survived or died during hospitalization. The probabilities of in-hospital mortality estimated by the decision tree and the decision path are 1.0 and 0.03 respectively. The patient in this case did not die in the hospital. With a probability threshold of 0.5, the decision tree would misclassify the patient, but the decision path would classify them correctly. Furthermore, the features in the two paths differ, and in this example the path in the tree has more features than the personalized path.

The first personalized decision path method was the Lazy Decision Tree.[14] The approach was similar to a standard decision tree in that it recursively partitioned the training dataset by evaluating and selecting combinations of predictors. The score used by the algorithm was a class-normalized information gain score. However, instead of calculating average entropy across all of the leaf nodes that correspond to the possible values of a given predictor variable, the LazyDT

only calculated entropy for the leaf to which the test case corresponds, and only considered that feature for further splitting. By only scoring the features that were present in the test case for inclusion in the model, the algorithm avoided the tradeoff that occurs when a predictor has a combination of high and low scoring leaf nodes that result in a high average score. This approach allows for construction of a model that is optimized for the features present in the test case.

Freidman et al. compared the predictive performance of the LazyDT method to a population decision tree method called C4.5 in terms of accuracy. To evaluate the algorithms, 28 UC Irvine datasets were used, which included medical datasets on heart disease, diabetes, breast cancer, and audiology. LazyDT had an average accuracy of 84.00% compared to the decision tree's average accuracy of 82.09%, and LazyDT outperformed the decision tree in terms of accuracy on 16 of the 28 datasets.

The next investigation into decision paths came in the form of the Patient-Specific Decision Path with Information Gain (PSDP-IG) and Balanced Accuracy (PSDP-BA) by Ferreira et al.[17] in 2013. These methods were very similar to LazyDT in that predictors were evaluated only using data that shared features with the test case, but both PSDP methods made modifications to the scoring metrics. Additionally, both PSDP methods attempted to perform pruning and used a Bayesian estimator called BDeu (as opposed to the more commonly used maximum likelihood estimate) to calculate probability estimates to mitigate possible overfitting. The PSDP-IG selected predictors whose leaves resulted in the highest information gain in terms of entropy, making it quite similar to LazyDT. PSDP-BA selected predictors whose leaves resulted in the highest balanced accuracy (calculated as the arithmetic mean of sensitivity and specificity) using leave-one-out cross validation. The methods were compared to a population decision tree method, CART, in terms of balanced accuracy and area under receiver operating characteristic curve

(AUROC) on five clinical datasets. They found that the population and personalized methods performed equally well in terms of balanced accuracy, but the PSDP methods outperformed the decision tree in terms of AUROC, with the mean AUROCs of the PSDP-IG ranging from 0.70 to 0.75 and those of the decision tree ranging from 0.65 to 0.68.

In 2015, Visweswaran et al.[15] introduced three new personalized decision path methods, each with a unique scoring criterion: DP-BAY, DP-IG, and DP-AUC. These methods all differed from LazyDT in that they used a novel approach to evaluate predictors for inclusion in the decision path, applied pruning to the final model, and used the BDeu estimator for calculating probability parameters in the models. Unlike previous decision path methods (which evaluated candidate predictors using only those cases in the training data that shared all features with the candidate path), the DP algorithm's scoring procedure also incorporated cases from the training data that shared all features with the path *except* for the value of the candidate predictor. Each candidate model therefore contained two leaf nodes, and the scores of these leaf nodes were combined as a weighted average to score the candidate predictor. This approach more closely resembled a population decision tree, in that the score incorporated data from training cases that did not share all features in the decision path with the test case.

The DP-BAY method used a Bayesian score that sought to derive the posterior probability of the candidate model given the data and select the candidate predictor that resulted in the model with the highest posterior probability. The DP-IG method used an Information Gain score that sought to identify the predictor that resulted in the greatest reduction in entropy. The DP-AUC method sought to select the predictor with the highest AUROC when calculated on the training data using leave-one-out cross validation. The DP methods were evaluated on five clinical and genomic datasets and compared to a population decision tree method, CART, in terms of AUROC

and Brier-skill-score (BSS). The DP-BAY had the highest mean AUROC (0.748), and both the DP-BAY and DP-IG had statistically significantly better performance than the decision tree in terms of AUROC and BSS.

Another AUROC-based personalized decision path method called PSDP-AUC was introduced by Ribeiro et al. in 2015.[16] Like LazyDT, PSDP-IG, and PSDP-Bay, the scoring only incorporated training data cases that shared all path features with the test case. When evaluated on 13 UC Irvine datasets, it had comparable performance to the PSDP-IG in terms of AUROC.

There have been a variety of personalized decision path methods developed over the past few decades. All of these methods construct a personalized decision path for a test case and can only handle discrete data. The differences between these methods are listed in Table 2. While all have demonstrated equivalent or superior performance to population methods, none resulted in notably high performance as measured by metrics such as accuracy or AUROC. Work therefore remains in improving the predictive performance of these methods.

Table 2. Differences across various decision path methods.

Method	**Scoring Metric**	**Score incorporates training data that does not share all path features with test case (non-path cases)**	**Pruning**	**Parameter Estimate Smoothing**
LazyDT	Entropy	No non-path cases	No Pruning	No Smoothing
PSDP-IG	Entropy	No non-path cases	Pruning	Smoothing
PSDP-BA	Balanced Accuracy	No non-path cases	Pruning	Smoothing
DP-Bay	Bayesian	Some non-path cases	Pruning	Smoothing
DP-IG	Entropy	Some non-path cases	Pruning	Smoothing
DP-AUC	AUROC	Some non-path cases	Pruning	Smoothing
PSDP-AUC	AUROC	No non-path cases	No Pruning	Smoothing

2.4.3 Scoring Criteria

As exhaustive search over all possible decision trees (referred to as the model space) for a globally optimal model is often intractable, heuristic search of the model space is often employed. The heuristic search is guided by a scoring criterion that assesses the degree of fitness between different models and the training data. Here we describe several approaches used in scoring decision tree and decision path models.

2.4.3.1 Information Theoretic Scoring

Decision tree methods, like CART and ID3, commonly use information theoretic scores.[57] Information theoretic scores seek to minimize the impurity of the partitions of a decision tree. The primary measure of the impurity is entropy, defined as $H(X) = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i)$ for a discrete random variable X with possible outcomes x_1, ..., x_n. Entropy varies based on the probability distribution of the outcomes: when the outcomes of a variable have fairly uniform probabilities, the variable is uninformative and the entropy is high, while if the probabilities of outcomes are unevenly distributed, the variable is informative and entropy is low. Purity is achieved when all samples with a given value of the variable share the same outcome. Information gain is a commonly used information theoretic score that measures the change in entropy following the transformation of an entity, such as the addition of a variable to a decision tree. The greatest information gain score results from the addition of a variable that results in the greatest reduction of entropy. It has been shown there is little difference in predictive performance for decision trees trained using different variations of information theoretic scoring criteria.[70]

2.4.3.2 Bayesian Scoring

The application of Bayesian methods to learning decision trees was first investigated by Buntine and Chipman.[71,72] When using Bayesian scores for model construction, we seek to identify the tree structure that maximizes the posterior probability of the tree conditioned on the data, thereby quantifying the degree of fitness between the structure and the data.[73] Bayesian scores tend to favor simpler structures, but can flexibly accommodate more complex structures if such structures are supported by sufficient data, and in this balance Bayesian trees may avoid overfitting.[74]

Bayesian scores take their name from Bayes' Theorem, which states: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$. Given a model *M* training dataset *D,* we aim to evaluate $P(M|D)$, the posterior probability of the tree conditioned on the data. By Bayes' theorem, we see that $P(M|D) = \frac{P(D|M)P(M)}{P(D)}$, where $P(D|M)$ is the marginal likelihood of the data given the model, $P(M)$ is the prior probability of the model structure, and $P(D)$ is the probability of the data and is constant. Thus, $P(M|D) \propto P(D|M)P(M)$. When all models are considered equally likely a priori, the prior $P(M)$ is a constant, and $P(M|D) \propto P(D|M)$. Therefore, by calculating the conditional probability distribution $P(D|M)$ (or a quantity proportional to it), Bayesian scores can be used to identify the model structure *M* with the highest posterior probability $P(M|D)$.

Bayesian scores incorporate prior probability distributions for the model structure and model parameters. It is often assumed that all models are equally likely a priori, resulting in the use of a uniform structure prior probability distribution.[74] However, prior belief regarding the expected number of predictor variables for a model can be used to calculate a nonuniform structure prior probability distribution, which can exert a regularization effect on model construction.

Similarly, the parameter prior probability distribution can use an uninformed prior (as is seen in the K2 score)[75] or it can incorporate a smoothing parameter called the prior equivalent sample size, which represents belief regarding the number of cases which would need to be seen to reach our current confidence in the parameter distribution (as seen in the BDe score).[76] The choice of this parameter is dependent on the user, and the effects of different values on model structures are not always predictable.[77]

2.5 Ensembles and Forests

As described previously, a decision tree algorithm constructs a model by selecting the highest scoring predictors to produce the highest scoring model. However, as exhaustive search over the model space is intractable and training data is finite, there is always uncertainty regarding the selection of the final model and its optimality. Rather than select a single model for prediction, one approach to address this uncertainty is to construct a variety of diverse models called an ensemble and average their outputs for prediction. This can reduce variance, resulting in lower error rates and higher predictive performance. When the base model used is a decision tree, the ensemble is referred to as a "forest."

2.5.1 Bagging

Bootstrap Aggregation, or Bagging, is a method for producing ensembles of models from a single training dataset.[78] The approach involves constructing multiple bootstrap datasets from

the original training dataset, training a given type of base model on each bootstrap, and averaging their individual predictions. Given a training dataset of *m* cases, a bootstrap dataset is produced by uniformly sampling *m* training cases with replacement from the original training dataset. On average, each bootstrap dataset contains approximately 62% of the cases from the training dataset.[79] Since the data distribution varies slightly between bootstrap datasets, a different model results from each bootstrap dataset despite using the same algorithm for training the models. Decision trees that are constructed with greedy search strategies can demonstrate instability and are sensitive to small changes in the data, and so bagging can be effective for improving predictive performance of decision trees.[80] Producing a variety of models may help capture the true relationships between the features and the target more effectively that a single model. To perform inference, the individual outputs of the models in the ensemble are averaged to produce an aggregate output.

2.5.2 Boosting

To further enhance the performance of ensembles of learners, an iterative approach for training models on weighted training datasets called boosting has been developed. Whereas bagging generates bootstrap datasets independently of one another in parallel, boosting modifies the training dataset in a sequential manner, incorporating performance information from the previous model trained on the weighted training dataset to influence the generation of a new weighted dataset via relevance weights.

Starting with equal weights initially, a boosting method trains a model on the training dataset and applies it to classify the cases in the uniformly weighted training dataset. The training

cases from the training dataset that are misclassified by the model are identified, and those training cases are weighted more heavily, while correctly classified training cases receive lower weights. The process is then repeated. This allows subsequent models in the ensemble to learn from difficult-to-classify cases and improve on the performance of previous models. One canonical boosting method is AdaBoost,[81] which uses decision trees as the base model and was found to reduce the classification error rate when compared to other ensemble methods like bagging.

2.5.3 Bayesian Model Averaging

Another ensemble approach is Bayesian model averaging (BMA), where multiple models are combined using their posterior probabilities. The theoretical "Optimal Bayes Classifier" is an ensemble of every possible model in the model space, and each model is weighted by its posterior probability when averaging across the ensemble to produce a prediction.[82] No other classifier can outperform this theoretical ensemble on average given the same model space and prior probabilities, but the construction of this optimal ensemble is not usually computationally feasible. To approximate this ensemble, heuristic search is used and models with posterior probabilities above a certain threshold are combined to form an ensemble. An even simpler approach involves performing bagging and calculating the target prediction as a weighted average according to each model's posterior probability; however, this approach may have issues with overfitting.[83]

2.5.4 Random Forest

One highly successful ensemble approach that uses decision trees is the random forest method.[84] Random forests use bagging to create an ensemble of randomized decision trees in parallel. The randomized decision tree algorithm only allows a random subset of predictors to be considered for inclusion in the decision tree at each potential split, which helps to increase the diversity of models in the ensemble. The number of variables sampled can vary, but a commonly used number is the square root of the total number of variables in the training dataset. By training a set of decorrelated trees, the random forest is better able to capture significant patterns in the data without increasing bias. Averaging the outputs of these trees then allows for a reduction of variance, lowering the overall error. Inference using a random forest is performed in the same manner as a bagged ensemble of standard decision trees: the corresponding path and leaf node for the test case is identified in each tree in the forest, and the parameters from the leaf nodes are then averaged to produce an aggregate probability distribution of the target variable. Random forest has been successfully used for clinical prediction, such as predicting future development of Alzheimer's disease in patients with mild cognitive impairment using MRI data.[39]

2.5.5 Personalized Ensembles and Forests

Some work has been done in developing personalized versions of the ensemble methods described above. Lee et al.[85] introduced a patient-specific bagging method where a patient similarity metric was used to quantify each training data case's similarity to the test case, and these similarity values were used as resampling weights to generate personalized bootstrap datasets.

Training cases that were more similar to the test case would appear with greater frequency in the bootstrap datasets. Standard decision tree and logistic regression models were trained on the bootstrap datasets, resulting in patient-specific bagged ensembles, but the authors failed to find an improvement in predictive performance using their method.

Margineantu et al.[19] presented the Bagged Lazy Option Tree in 2002, which used a variant of the LazyDT method in an ensemble approach. Option trees are similar to decision trees, except rather than select a single predictor and value to form an interior node, the option tree algorithm selects several possible predictors (or options) at an interior node and averages their predictions to produce the model prediction. Margineantu et al. developed a lazy option tree method based on LazyDT and used a standard bagging approach to create a personalized ensemble, which demonstrated better calibration than a population method when applied to 16 UC Irvine datasets.

Fern et al.[20] developed a novel personalized boosting method called BO-LazyDT that used LazyDT as the base model. Traditional boosting methods require classification of every case in the training dataset by the model to generate training case weights, but this is impossible with decision paths like LazyDT because a decision path model can only classify those cases that share all features contained in the path. The novel personalized boosting method proposed by Fern instead weights training cases using a path-derived relevance weight. The relevance weight is calculated by identifying the number of features in the path shared by each case in the training dataset. This allows each case in the training dataset to be weighted according to its relevance to the test case, and this weighted training dataset is used to train the subsequent model. This method was shown to outperform the base model (LazyDT) as well as a bagged ensemble of the base model (BA-LazyDT) in terms of accuracy and demonstrated comparable performance to AdaBoost, a population boosting method. Furthermore, the paths produced by the BO-LazyDT

were significantly shorter than those produced by the population method, indicating that personalized methods may allow for greater comprehensibility without sacrificing predictive performance.

Visweswaran et al.[86] described a novel machine learning approach that performed BMA with an ensemble of personalized Bayesian network models. This method was then employed to predict patient outcomes in four clinical datasets.[87] The authors found that, in terms of logarithmic loss and squared error, the patient-specific BMA approach outperformed both logistic regression and a patient-specific approach that selected a single model for prediction. These models were computationally expensive, however.

There has been little work in personalization of random forests, and the existing work involves personalization of the bootstrap datasets using similarity metrics. The Case-Specific Random Forest (CSRF) introduced by Xu et al.[88] in 2016 used a distance metric to identify cases in the training dataset which were similar to the test case. Similarity weights were then applied when bootstrap datasets were generated, so that similar cases appeared more frequently in the bootstrap datasets. Randomized decision trees are then trained on these personalized bootstrap datasets. Lee et al.[48] introduced a patient-specific random forest method that trained a random forest using a subset of the *M* most similar training data cases in the training dataset, determined using a patient similarity metric. They compared this method to CSRF to predict 30-day mortality in ICU patients, finding that both methods had an average AUROC of approximately 0.83.

Thus, limited work has been done on bagged and boosted ensembles of personalized decision paths, but these investigations did not focus on their use in patient-specific prediction. Furthermore, no investigation has previously been done into using personalized decision paths in

a random forest approach. The field of patient-specific prediction could thus benefit from further evaluation of novel personalized decision paths in ensemble approaches.

2.6 Existing and Novel Methods

Prior investigations in tree-based methods have explored the effects of different scoring criteria, search strategies, and approaches for constructing ensembles. In this dissertation, we present three novel algorithms.

First is the Personalized Decision Path with Bayesian score (PDP-Bay), which produces a single personalized model using Bayesian scoring. The second novel algorithmic method is the Lazy Random Forest (LazyRF), which produces an ensemble of randomized personalized decision paths. The third method is the Boosted PDP-Bay (BO-PDP-Bay), which produces an ensemble of boosted personalized decision paths with Bayesian scoring.

In Table 3 we provide an overview of prior work and highlight areas of new inquiry which are explored in this dissertation (designated in italics).

Table 3. Overview of tree-based models and ensemble methods.

Model	**Decision Tree**	**Personalized Path**
Entropy-Scored Single Model	CART (Breiman, 1984)	LazyDT (Friedman, 1996)
Bayesian-Scored Single Model	Bayes Trees (Buntine, 1992)	*PDP-Bay (Johnson, 2020)*
Bagged Ensemble	Bagged CART (Breiman, 1996)	BA-LazyDT (Fern, 2003)
Random Forest Ensemble	Random Forest (Breiman, 2001)	*LazyRF (Johnson, in submission)*
Boosted Ensemble	AdaBoost (Freund, 1997)	BO-LazyDT (Fern, 2003)
Bayesian Ensemble	Tree Averaging (Buntine, 1992) BART (Chipman, 2010)	*Boosted PDP-Bay*

3.0 Algorithmic Methods

This chapter describes the personalized and the corresponding population algorithmic methods used in this dissertation. Section 3.1 describes algorithms that learn personalized single decision paths; Section 3.2 describes algorithms that derive population decision trees; Section 3.3 describes algorithms that learn random forest ensembles of personalized decision paths; Section 3.4 describes algorithms that derive population random forests; Section 3.5 describes algorithms that derive boosted ensembles of personalized decision paths, and Section 3.6 describes AdaBoost which is a population boosting method. Figure 2 contains a Venn diagram which lists the algorithmic methods described in this chapter and displays the paradigms (personalized/ensemble/Bayesian) incorporated by each method.

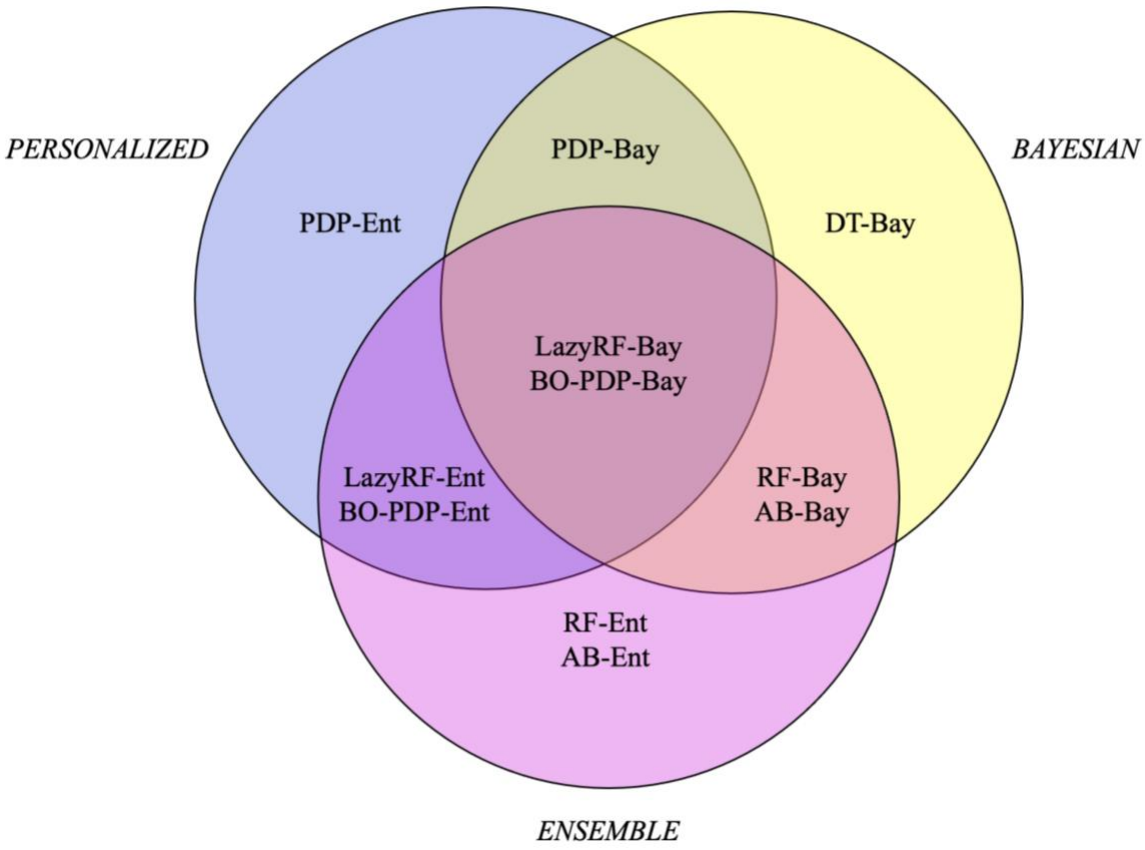


Figure 2. Venn diagram of algorithmic methods and correponding modeling paradigms.

3.1 Personalized Decision Paths

We developed a novel algorithm that learns a single decision path that is personalized to a test case (or a test individual); we call this method the Personalized Decision Path that uses a Bayesian score (PDP-Bay). We compare PDP-Bay with a previously developed personalized decision path method called the Personalized Decision Path that uses an Entropy score (PDP-Ent).

3.1.1 Background

The first Bayesian-scored personalized decision path was introduced by Visweswaran et al. in the form of the DP-Bay algorithm.[15] Algorithms that use Bayesian scoring seek to identify the model with the highest posterior probability given the corresponding data. This novel method, the PDP-Bay, provides a simpler procedure for identifying the most probable model than DP-Bay. We compared its performance to an entropy-scored personalized decision path algorithm (PDP-Ent, which is analogous to LazyDT) as well as population decision trees with entropy and Bayesian scores.

3.1.2 The PDP-Bay Method

This section provides details of the novel personalized decision path method, namely PDP-Bay. We first describe the model structure, then the search strategy, and finally the scoring criterion of the PDP-Bay method.

3.1.2.1 Model Structure

A decision-path model M is represented as $M = (S,\boldsymbol{\theta})$, where S is a path and $\boldsymbol{\theta}$ are the parameters of the probability distributions over the target T, which can take r possible values denoted by $(t_1, t_2, \ldots t_k, \ldots, t_r)$. Let $\boldsymbol{V} = (X_1, X_2, \ldots, X_i, \ldots, X_n)$ be a list of the n variables in the training dataset D. A path S consists of a conjunction of q features, such that $S = (X_1 = x_1 \wedge X_2 = x_2 \wedge \ldots X_j = x_j \wedge \ldots \wedge X_q = x_q)$. The variable list $\boldsymbol{V_S} = (X_1, X_2, \ldots, X_j, \ldots, X_q)$ is a subset of $\boldsymbol{V}$. The value list $\boldsymbol{v_S} = (x_1, x_2, \ldots, x_j, \ldots, x_q)$ consists of values for the variables in $\boldsymbol{V_S}$ in the test case. Finally, the parameter

list $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_k, .., \theta_r)$ denotes the r probabilities for the distribution $P(T \mid \boldsymbol{V_S} = \boldsymbol{v_S})$ over the target variable T.

The values of those probabilities are estimated from the cases in the training set for which $\boldsymbol{V_S} = \boldsymbol{v_S}$. To control for overfitting, we use a Bayesian estimator called the K2 score for calculating these probabilities. The estimate for probability θ_k is given as follows:

$$\theta_k \equiv P(T = t_k | \boldsymbol{V}_S = \boldsymbol{v}_S) = \frac{1+N_k}{r+N}, \tag{3.1}$$

where N is the number of cases in the training set that satisfy $\boldsymbol{V_S} = \boldsymbol{v_S}$ and N_k is the number of those cases that satisfy $\boldsymbol{V_S} = \boldsymbol{v_S}$ and for which $T = t_k$.

3.1.2.2 Search Strategy

The pseudocode for the PDP-Bay method is shown in Figure 2. The method uses a forward-stepping, greedy hill-climbing search and a sample-normalized Bayesian score (explained in the next section) as the criterion for evaluating features for inclusion in the path. Beginning with an empty path S, the algorithm successively adds features to S that locally maximize the score, until the score can no longer be improved. Currently, the method is designed to work with discrete data only.

PDP-Bay uses a training dataset D and a test case ***Test***. For each possible feature $X = x$, the algorithm temporarily appends $X = x$ to S to produce candidate path S'. This path is scored using training data $D_{S'}$, which is the data for all cases that share the features in S'. If the score of S' is greater than that of S, the score and candidate feature are stored. When all the features in $\boldsymbol{V}$ have been scored, the highest scoring feature is appended to S to create a new version of S, and the training dataset is reduced to cases that share values with ***Test*** for the variables in that new S. The growth of the path is terminated when no candidate feature can improve the score of path S, no

remaining training cases share values with ***Test*** for the variables in *S'*, or all remaining cases have the same value for *T*.

```
PDP-Bay (D, Test, α0, e)
    INPUT:   D is a training dataset consisting of m training cases and n variables and target variable T,
             Test is data for a test case that is not in D and whose T is to be predicted
             α0 is a parameter prior hyperparameter
             e is a structure prior hyperparameter
    OUTPUT: S, where S is a personalized path for the test case, and an estimate of P(T | Test)

1   S ← empty path with no predictor variables with a single leaf node
2   DS ← D
3   BayScore ← Bayesian score of S computed from DS, α0, and e using Equation 3.10
4   BestFeature ← ∅
5   LOOP until all cases in DS have the same value for T:
6       BayScoreBest ←BayScore
7       FOR each candidate feature X = x in Test such that X = x  is not in S:
8           S' ← add X = x to S
9           DS' ← training cases in DS with X = x
10          IF  DS' ≠ ∅ THEN:
11              BayScoreTemp ← Bayesian score of S' computed from DS', α0, and e using Equation 3.10
12              IF BayScoreTemp > BayScoreBest THEN:
13                  BayScoreBest ← BayScoreTemp
14                  BestFeature ← X
15      IF a BestFeature is found THEN:
16          S ← add BestFeature to S
17          DS ← cases in DS with X = x
19      ELSE:
20          EXIT from LOOP

21  RETURN S with P(T | Test) that is estimated using Equation 3.1 from DS (training data that satisfy S)
```

Figure 3. Pseudocode for PDP-Bay method.

3.1.2.3 Bayesian Scoring Criterion

The PDP-Bay method uses a Bayesian score which seeks to identify the model structure with the highest posterior probability. Given a candidate path *S'* that is derived from path *S* by temporarily appending a candidate feature $X = x$, we define the posterior probability of the path *S'* (i.e., the model structure) given the data, $D_{S'}$, that contains the cases that satisfy the path *S'* as

$$P(S'|D_{S'}) = \frac{P(D_{S'}|S')P(S')}{P(D_{S'})}, \tag{3.2}$$

where $P(D_{S'}|S')$ is the marginal likelihood of the data given the model structure, $P(S')$ is the prior probability of the model structure, and $P(D_{S'})$ is a normalizing constant. Thus, we see that

$$P(S'|D_{S'}) \propto P(D_{S'}|S')P(S'). \tag{3.3}$$

The marginal likelihood $P(D_{S'}|S')$ is measure of how well the path structure fits the data.

Marginal likelihood. Using the Bayesian approach, we compute $P(D_{S'}|S')$ by integrating over the parameter values:

$$P(D_{S'}|S') = \int_{\boldsymbol{\theta}} P(D_{S'}|S', \boldsymbol{\theta})P(\boldsymbol{\theta}|S')d\boldsymbol{\theta}, \tag{3.4}$$

where $P(D_{S'}|S', \boldsymbol{\theta})$ is the likelihood of the data given the path-model $(S', \boldsymbol{\theta})$ and $P(\boldsymbol{\theta}|S')$ is the prior distribution over different parameter values. Assuming a multinomial sample, parameter modularity and independence, no data is missing, cases occur independently, and parameter priors follow a Dirichlet distribution, we can compute the integral in Equation 3.4 in closed form using the Bayesian-Dirichlet likelihood equivalent (BDe) metric,[76] which is given by

$$P(D_{S'}|S') = \frac{\Gamma(\alpha_0)}{\Gamma(N+\alpha_0)} \prod_{k=1}^{r} \frac{\Gamma(N_k+\alpha_k)}{\Gamma(\alpha_k)}, \tag{3.5}$$

where r is the number of values of the target T, N_k is the number of cases in $D_{S'}$ for which $T = t_k$, $N = \sum_{k=1}^{r} N_k$ which is equal to $|D_{S'}|$, α_0 is a user-specified parameter prior, and $\alpha_k = \frac{\alpha_0}{r}$. In this equation, $\Gamma(\bullet)$ represents the gamma function. In this special case, when the parameter prior α_k is uniform, we get a specialized BDe score called the Bayesian-Dirichlet likelihood equivalent and uniform (BDeu) score.

If all values are also discrete, the integral in Equation 3.4 can alternatively be computed in closed form using the K2 metric,[75] which is given by

$$P(D_{S'}|S') = \frac{(r-1)!}{(N+r-1)!} \prod_{k=1}^{r} N_k!, \tag{3.6}$$

where r is the number of values of the target T, N_k is the number of cases in $D_{S'}$ for which $T = t_k$, and $N = \sum_{k=1}^{r} N_k$, which is equal to $|D_{S'}|$. It is useful to note that when the target variable is binary and $\alpha_0 = 2$, Equations 3.5 and 3.6 provide equivalent results (details are provided in Appendix B).

Structure prior. The structure prior probability distribution is often assumed to be uniform, in which case we see that $P(S'|D_{S'}) \propto P(D_{S'}|S')$, as $P(S')$ is constant. If we do not assume that all model structures are equally likely a priori, then we provide an informative structure prior $P(S')$ in Equation 3.3 using a binomial prior distribution. For a given path S', let q be the number of features in S'. Let e be the number of predictor variables in D that a domain expert would expect to be predictive of the target variable, and let n be the total number of predictors in D. The structure prior is then given by

$$P(S') = \left(\frac{e}{n}\right)^{q} \left(1 - \frac{e}{n}\right)^{(n-q)}. \tag{3.7}$$

Note that if e = 0 or n, the structure prior reduces to the uniform structure prior distribution.

Sample normalization. We sample-normalize the marginal likelihood by taking the geometric mean to enable score comparison between paths with varying sample sizes. Since we assume that cases occur independently, we see that $P(D_{S'}|S') = \prod_{l=1}^{N} P(Case_l|S')$, and so by taking the geometric mean we calculate the expected value of $P(Case|S')$. The sample-normalized posterior probability represents the expected marginal likelihood of a single future case, which serves a measure of how well on average $P(T \mid \textbf{\textit{Test}})$ predicts the cases in $D_{S'}$. For computational efficiency and precision, the final score is calculated in logarithmic form.

BayScore. The score of a path S' that includes the addition of variable X is therefore defined as

$$BayScore(S') = \log\left[P(D_{S'}|S')^{1/N} \cdot P(S')\right]. \quad (3.8)$$

This equation provides us with a value that is proportional to the log of the expected posterior probability of S'. Substituting the expression for $P(D_{S'}|S')$ from Equation 3.5 and the expression for $P(S')$ from Equation 3.7 into Equation 3.8, we obtain the BayScore using the BDeu score as

$$BayScore_{BDeu}(S') = \log\left[\left[\left(\frac{\Gamma(\alpha_0)}{\Gamma(N+\alpha_0)}\prod_{k=1}^{r}\frac{\Gamma(N_k+\alpha_k)}{\Gamma(\alpha_k)}\right)\right]^{1/N} \cdot \left(\left(\frac{e}{n}\right)^{q}\left(1-\frac{e}{n}\right)^{(n-q)}\right)\right], \quad (3.9)$$

which can be simplified to

$$BayScore_{BDeu}(S') = \frac{1}{N}\left[\log\left(\frac{\Gamma(\alpha_0)}{\Gamma(N+\alpha_0)}\right) + \sum_{k=1}^{r}\log\left(\frac{\Gamma(N_k+\alpha_k)}{\Gamma(\alpha_k)}\right)\right] + q\log\left(\frac{e}{n}\right) + (n-q)\log\left(1-\frac{e}{n}\right). \quad (3.10)$$

If we instead substitute the expression for $P(D_{S'}|S')$ from Equation 3.6 into Equation 3.8, we obtain the BayScore using the K2 score as

$$BayScore_{K2}(S') = \log\left[\left[\left(\frac{(r-1)!}{(N+r-1)!}\prod_{k=1}^{r}N_k!\right)\right]^{1/N} \cdot \left(\left(\frac{e}{n}\right)^{q}\left(1-\frac{e}{n}\right)^{(n-q)}\right)\right], \quad (3.11)$$

which can be simplified to

$$BayScore_{K2}(S') = \frac{1}{N}\left[\log\left(\frac{(r-1)!}{(N+r-1)!}\right) + \sum_{k=1}^{r}\log N_k!\right] + q\log\left(\frac{e}{n}\right) + (n-q)\log\left(1-\frac{e}{n}\right). \quad (3.12)$$

3.1.3 PDP-Bay-BDeu

When using the $BayScore$ to train a personalized decision path with PDP-Bay, the user may specify the expected number of predictors e as a hyperparameter of the structure prior, and when using $BayScore_{BDeu}$, the user may also specify the equivalent sample size α_0 as a hyperparameter of the parameter prior. Here we present the PDP-Bay-BDeu, which performs a grid search over combinations of α_0 and e to empirically identify the prior hyperparameters that correspond to the

model with the highest AUROC. For each combination of α_0 and e, we performed stratified 5-fold cross validation (5FCV) to estimate performance of decision paths trained using PDP-Bay in terms of AUROC. We then use the combination of α_0 and e that resulted in the highest AUROC to train a decision path with the PDP-Bay method. Pseudocode for PDP-Bay-BDeu appears in Figure 3.

```
PDP-Bay-BDeu(D, Test, Alphas, ENPs)
    INPUT: D is a training dataset consisting of m training cases and n variables and target variable T,
           Test is data for a test case that is not in D and whose T is to be predicted
           Alphas ← list of nonnegative real numbers to test for α0
           ENPs ← list of nonnegative integers to test for e
    OUTPUT: S, where S is a personalized path for the test case, and an estimate of P(T | Test)

1   BestAlpha ← ∅; BestENP ← ∅; BestAUROC ← 0
2   D ← 5 disjoint subsets of D with equal prevalence of values of T, each of size D/5
3   FOR each α0 in Alphas:
4       FOR each e in ENPs:
5           FOR each Di in D:
6               Dtest ← Di
7               Dtrain ← D - Di
8               FOR each case Testj in Dtest
9                   Pathj, Predictionj ← PDP-Bay(Dtrain, Testj, α0, e)
10          AUROC ← AUROC from predicted values of T and true values of T for each case in D
11          IF AUROC > BestAUROC
12                  BestAlpha ← α0; BestENP ← e
13  S, P(T | Test) ← PDP-Bay(D, Test, BestAlpha, BestENP)

RETURN S, P(T | Test)
```

Figure 4. Pseudocode for PDP-Bay-BDeu method.

3.1.4 The PDP-Ent Method

The PDP-Ent method is our implementation of a previously described decision path method that uses an entropy score (like LazyDT[14]). We describe the model structure, the search strategy, and the scoring criterion of the PDP-Ent method.

3.1.4.1 Model Structure

The PDP-Ent has the same model structure at PDP-Bay (described in 3.1.2.1). A decision-path model M is represented as $M = (S,\boldsymbol{\theta})$, where S is a collection of features and $\boldsymbol{\theta}$ are the parameters of the probability distributions over the target T.

3.1.4.2 Search Strategy

PDP-Ent utilizes a search strategy that is similar to that used in the PDP-Bay method. It performs a forward-stepping, greedy hill-climbing search and evaluates features for inclusion in the path using an information gain score based on the entropy of the path. Beginning with an empty path S, the algorithm successively adds features to S that locally maximize the score, until the score can no longer be improved.

3.1.4.3 Entropy Scoring Criterion

For a candidate path S and data D_S that contains the training cases that satisfy the path S, the entropy of S is given by

$$H(S) = -\sum_{k=1}^{r} P(T = t_k | \boldsymbol{V}_S = \boldsymbol{v}_S) \log_2 P(T = t_k | \boldsymbol{V}_S = \boldsymbol{v}_S), \quad (3.13)$$

where $P(T = t_k|\boldsymbol{V}_S = \boldsymbol{v}_S)$ is the proportion of cases in the dataset $D_{S'}$ that have the value t_k for T. In the PDP-Ent method, we use the maximum likelihood estimator to calculate the probabilities $\boldsymbol{\theta}$ associated with a path S. The estimate for probability θ_k is given by

$$\theta_k \equiv P(T = t_k|\boldsymbol{V}_S = \boldsymbol{v}_S) = \frac{N_k}{N}, \tag{3.14}$$

where N is the number of cases in the training set that satisfy $\boldsymbol{V_S} = \boldsymbol{v_S}$ and N_k is the number of those cases that satisfy $\boldsymbol{V_S} = \boldsymbol{v_S}$ and for which $T = t_k$. This is a common estimator that is used in many decision tree methods. Thus, the entropy of a variable X under consideration for inclusion in the candidate path S' is defined as

$$EntScore(S') = -\sum_{k=1}^{r} \frac{N_k}{N} \log_2 \left(\frac{N_k}{N}\right). \tag{3.15}$$

After some algebraic manipulation, Equation 15 can be rewritten as

$$EntScore(S') = -\frac{1}{N}\left(\sum_{k=1}^{r} N_k \log_2 \frac{N_k}{N}\right), \tag{3.16}$$

where N is the number of cases in the training set that satisfy $\boldsymbol{V_S} = \boldsymbol{v_S}$, N_k is the number of those cases that satisfy $\boldsymbol{V_S} = \boldsymbol{v_S}$ and for which $T = t_k$, and the term inside the parentheses is the log-likelihood. The search selects variables that maximize the information gain, which is calculated as the difference in entropy between the current path S and candidate path S'. The pseudocode for the PDP-Ent algorithm can be found in Appendix A, Figure 1.

3.2 The Decision Tree Method

We compare the predictive performance of personalized decision paths with population decision trees. We describe the model structure, the search strategy, and the scoring criteria of the decision tree methods.

3.2.1 Model Structure and Search Strategy

The decision-tree model M is represented as $M = (T, \boldsymbol{\theta})$, where $T = (S_1, S_2, \ldots, S_h, \ldots, S_m)$ is a collection of m paths. Given a training dataset D, the method performs a forward stepping, greedy hill-climbing search to add one variable at a time that locally maximizes a scoring criterion. The decision tree method does not perform pruning and terminates when the addition of a variable does not increase the tree's score.

3.2.2 Decision Tree with Entropy Score

The decision tree with entropy score (DT-Ent) uses a scoring criterion that maximizes information gain, which is the difference in entropy between the existing tree and the candidate tree. The entropy score of a tree T is given by

$$TreeEntScore(T) = \sum_{h=1}^{m} \left[\frac{|DS_h|}{|D|} \sum_h H(S_h) \right], \tag{3.17}$$

where DS_h is the set of cases in D that satisfy the path S_h and $H(S_h)$ is given by Equation 3.13. Like in PDT-Ent, we use the maximum likelihood estimator as given by Equation 3.14 for

calculating the probabilities $\boldsymbol{\theta}$ associated with a path S_h. This is a standard implementation of information gain in decision tree algorithms.[57]

3.2.3 Decision Tree Method with Bayesian Score

The decision tree with Bayesian score (DT-Bay) uses a scoring criterion that maximizes the posterior probability distribution of the tree structure given the data, $P(T|D)$. By Bayes' theorem,

$$P(T|D) \propto P(D|T)P(T), \tag{3.18}$$

where $P(D|T)$ is the marginal likelihood of the data given the tree and $P(T)$ is the prior probability of the tree structure.

Marginal likelihood. Using the Bayesian approach, we compute the marginal likelihood of the data given the tree, $P(D|T)$, by integrating over the parameter values:

$$P(D|T) = \int_{\boldsymbol{\theta}} P(D|T,\boldsymbol{\theta})P(\boldsymbol{\theta}|T)d\boldsymbol{\theta}, \tag{3.19}$$

where $P(D|T,\boldsymbol{\theta})$ is the likelihood of the data given the tree-model $(T,\boldsymbol{\theta})$ and $P(\boldsymbol{\theta}|T)$ is the prior distribution over different parameter values. Assuming a multinomial sample, parameter modularity and independence, no data is missing, cases occur independently, and parameter priors follow a Dirichlet distribution, we can compute the integral in Equation 3.19 in closed form using the BDeu score,[76] which is given by

$$P(D|T) = \prod_{l=1}^{m} \frac{\Gamma(\alpha_l)}{\Gamma(N_l+\alpha_l)} \left[\prod_{k=1}^{r} \frac{\Gamma(N_{lk}+\alpha_{lk})}{\Gamma(\alpha_{lk})}\right], \tag{3.20}$$

where m is the number of paths in T, r is the number of values of the target X_T, N_{lk} is the number of cases in D which belong to path l for which $X_T = t_k$, $N_l = \sum_{k=1}^{r} N_{lk}$, which is equal to the number

of cases that belong to path l, $\alpha_l = \frac{\alpha_0}{m}$ where α_0 is a user-specified parameter prior, and $\alpha_{lk} = \frac{\alpha_l}{r}$. In this equation, $\Gamma(\bullet)$ represents the gamma function.

Alternately, assuming parameter modularity and independence and that the variables are discrete, no data is missing, cases occur independently, and parameter priors follow a Dirichlet distribution, we can compute the integral in Equation 3.19 in closed form using the K2 score,[75] resulting in

$$P(D|T) = \prod_{l=1}^{m} \frac{(r-1)!}{(N_l+r-1)!} \left[\prod_{k=1}^{r} N_{lk}!\right], \tag{3.21}$$

where m is the number of paths in T, r is the number of values of the target X_T, N_{lk} is the number of cases in D which belong to path l for which $X_T = t_k$, and $N_l = \sum_{k=1}^{r} N_{lk}$, which is equal to the number of cases that belong to path l.

Structure Prior. The structure prior probability distribution is often assumed to be uniform, in which case we see that $P(T|D) \propto P(D|T)$, as $P(T)$ is constant. If we do not assume that all model structures are equally likely a priori, then we can also provide an informative structure prior $P(T)$ in Equation 3.18 using a binomial prior distribution.

For a given tree T, let q be the number of features in T. Let e be the number of predictor variables in D that a domain expert would expect to be predictive of the target variable, and let n be the total number of predictors in D. The structure prior is then given by

$$P(T) = \left(\frac{e}{n}\right)^q \left(1 - \frac{e}{n}\right)^{(n-q)}. \tag{3.22}$$

Note that if e = 0 or n, the structure prior assumes a uniform distribution.

TreeBayScore. For computational efficiency and precision, the final score is calculated in logarithmic form. Thus, substituting

$$TreeBayScore(T) = \log[P(D|T)P(T)]. \tag{3.23}$$

Substituting the equation for $P(D|T)$ from Equation 3.20 and the equation for $P(T)$ from Equation 3.22 into Equation 3.23, we obtain

$$TreeBayScore_{BDeu}(T) = \log\left[\left(\prod_{l=1}^{m}\frac{\Gamma(\alpha_l)}{\Gamma(N_l+\alpha_l)}\left[\prod_{k=1}^{r}\frac{\Gamma(N_{lk}+\alpha_{lk})}{\Gamma(\alpha_{lk})}\right]\right)\left(\frac{e}{n}\right)^{q}\left(1-\frac{e}{n}\right)^{(n-q)}\right]. \quad (3.24)$$

Rewritten, this takes the form

$$TreeBayScore_{BDeu}(T) = \sum_{l=1}^{m}\left[\log\left(\frac{\Gamma(\alpha_l)}{\Gamma(N_l+\alpha_l)}\right)+\sum_{k=1}^{r}\log\left(\frac{\Gamma(N_k+\alpha_k)}{\Gamma(\alpha_k)}\right)\right]+q\log\left(\frac{e}{n}\right)+(n-q)\log\left(1-\frac{e}{n}\right). \quad (3.25)$$

If we instead substitute the equation for $P(D|T)$ from Equation 3.21 into Equation 3.23, we obtain

$$TreeBayScore_{K2}(T) = \log\left[\left(\prod_{l=1}^{m}\frac{(r-1)!}{(N_l+r-1)!}\left[\prod_{k=1}^{r}N_{lk}!\right]\right)\left(\frac{e}{n}\right)^{q}\left(1-\frac{e}{n}\right)^{(n-q)}\right]. \quad (3.26)$$

Rewritten, this takes the form

$$TreeBayScore_{K2}(T) = \sum_{l=1}^{m}\left[\log\left(\frac{(r-1)!}{(N_l+r-1)!}\right)+\sum_{k=1}^{r}\log\left(N_{lk}!\right)\right]+q\log\left(\frac{e}{n}\right)+(n-q)\log\left(1-\frac{e}{n}\right). \quad (3.27)$$

DT-Bay-BDeu. The DT-Bay-BDeu is a variation of the DT-Bay algorithmic method which scores models using Equation 3.24 and performs prior hyperparameter tuning using 5FCV on the training data to identify the combination of α_0 and e that results in the highest AUROC.

3.3 Lazy Random Forest

In our first personalized ensemble method, we introduce a new method called the Lazy Random Forest (LazyRF). This method uses the previously described PDP-Bay as the base model of a Random Forest ensemble. We hypothesized that the new method would perform better than 1) the PDP-Bay alone and 2) a population Random Forest approach, and we evaluated this hypothesis on a range of datasets.

3.3.1 Background

Random forests are bagged ensembles of randomized decision trees, which are described in 2.5.4. Bagging, which is discussed in greater detail in 2.5.1, involves generating bootstrap datasets by sampling with replacement from a training dataset. Models are trained on these bootstrap datasets, and the individual predictions are averaged to produce an aggregate prediction. The standard random forest uses a population method, the randomized decision tree, as its base model, though some efforts have been made to produce patient-specific random forests. Lee et al.[48] used the Case Specific Random Forest (CSRF)[88] for patient-specific prediction. The CSRF constructs a personalized random forest by creating personalized bootstrap datasets using a patient similarity metric (PSM). However, the base model of the CSRF remains the randomized decision tree, which is a population method. To our knowledge, no work has been published using personalized modeling methods like decision paths to produce a personalized random forest.

Some studies have explored personalized forests consisting of ensembles of personalized decision paths, which were discussed in more detail in section 2.5.5. Fern et al.[20] introduced two new ensembles of personalized decision paths, the bagged LazyDT (BA-LazyDT) and the boosted LazyDT (BO-LazyDT), and demonstrated that using these ensemble approaches with personalized decision paths improved predictive accuracy over the base method and matched performance of an analogous population method.

Our novel method, the LazyRF-Bay, uses a randomized version of a personalized decision path method (the PDP-Bay described in 3.1.2) as the base model to produce a new type of personalized ensemble. This method, described in greater detail in the next section, is to our knowledge the first random forest approach using personalized decision paths.

3.3.2 The LazyRF-Bay Method

We describe the base model structure, the search strategy, and the scoring criterion of the LazyRF-Bay method.

3.3.2.1 Base Model Structure

The ensemble produced by the LazyRF-Bay method consists of a forest of decision-path models. Detailed terminology regarding the structure of decision-paths is outlined in 3.1.2.1. The base model is a randomized Personalized Decision Path that uses a Bayesian score (PDP-Bay), which is a modified version of the algorithm described in 3.1.2. In the LazyRF-Bay, the PDP-Bay search is modified to randomize variable selection, which is described in the next section.

3.3.2.2 Search Strategy

The pseudocode for the LazyRF-Bay method is shown in Figure 4. Given a training dataset D and a test case ***Test***, the method constructs multiple bootstrap datasets and trains a randomized decision-path on each one to produce forest $\boldsymbol{F} = (S_1, S_2, \ldots, S_h, \ldots, S_b)$. A bootstrap dataset B_h is a set of m cases uniformly sampled with replacement from D. Let $\boldsymbol{B} = (B_1, B_2, \ldots, B_h, \ldots, B_b)$ be the set of bootstrap datasets.

The LazyRF-Bay method constructs a randomized decision-path model for each bootstrap dataset B_h in $\boldsymbol{B}$. To construct each path, the method uses a forward-stepping, greedy hill-climbing search and a Bayesian score as the criterion for evaluating features for inclusion in the path. Beginning with an empty path S, the algorithm successively adds features from ***Test*** to S that

locally maximize the score, until the score can no longer be improved. Currently, the method is designed to work with discrete data only.

The algorithm starts by uniformly sampling u variables without replacement from the set of variables $\boldsymbol{V}$ to form subset $\boldsymbol{V_U} = (X_1, X_2, \ldots, X_p, \ldots, X_u)$, where $u = \sqrt{n}$ (rounded to the nearest integer). The value list $\boldsymbol{v_U} = (x_1, x_2, \ldots, x_p, \ldots, x_u)$ consists of values in the test case for the variables in $\boldsymbol{V_U}$. For each possible feature $X = x$, the algorithm temporarily appends $X = x$ to S to produce candidate path S'. This path is scored using training data $D_{S'}$, which is the data for all cases that share values with ***Test*** for the features in S'. If the score of S' is greater than that of S, the score and candidate feature are stored. When all the features in $\boldsymbol{V_U}$ have been scored, the highest scoring feature is appended to S to create a new version of S, and the training dataset is reduced to cases that share values with ***Test*** for the variables in that new S. The growth of the path is terminated when no candidate feature can improve the score of path S, no remaining training cases share values with ***Test*** for the variables in S', or all remaining cases have the same value for T.

3.3.2.3 Bayesian Scoring Criterion

The score of a candidate path S' is calculated using the $BayScore_{K2}$ described in 3.1.2.3 with a uniform structure prior. This score selects variables that maximize the posterior probability of the model structure given the corresponding data.

3.3.2.4 Inference

To perform inference for ***Test***, parameter estimates are calculated using Equation 3.1 for each path S in forest $\boldsymbol{F}$. These parameter estimates are then averaged. The estimate for probability θ_k is therefore given by:

$$\theta_k \equiv \frac{1}{b}\sum_{h=1}^{b} P_h\left(T = t_k \middle| \boldsymbol{V}_{S_h} = \boldsymbol{v}_{S_h}\right) = \frac{1}{b}\sum_{h=1}^{b} \frac{1+N_{kh}}{r+N_h}, \tag{3.28}$$

Where b is the number of paths in the forest, r is the number of values taken by the target variables, N_h is the number of cases in the bootstrap dataset B_h that satisfy $\boldsymbol{V}_{S_h} = \boldsymbol{v}_{S_h}$, and N_{kh} is the number of those cases that satisfy $\boldsymbol{V}_{S_h} = \boldsymbol{v}_{S_h}$ and for which $T = t_k$.

```
LazyRF-Bay (D, b, Test)
    INPUT:    D is a training dataset of m cases and n variables and target variable T,
              b is the number of bootstrap datasets to create (size of ensemble),
              Test is data for a test case that is not in D and whose T is to be predicted
    OUTPUT:   F is a forest of personalized paths S for the test case
              P_F is the average of the estimates of P(T | Test) from each path in the forest

1   B ← b empty bootstrap datasets
2   F, P ← empty forest and associated probability estimates

3   FOR each bootstrap dataset B_h in B
4       B_h ← select m random cases with replacement from D

5       S ← empty path with no predictor variables with a single leaf node
6       D_S ← B_h
7       BayScore ← Bayesian score of S computed from D_S and e = 0 using Equation 3.12

8       LOOP until all cases in D_S have the same value for T
9           BayScoreBest ←BayScore
10          V_U = randomly selected subset of √n features in Test such that X = x is not in S
11          FOR each feature X = x in V_U DO:
12              S' ← add X = x to S
13              D_S' ← training cases in D_S with X = x
14              IF D_S' ≠ ∅:

15                  BayScoreTemp ← Bayesian score of S' computed from D_S and e = 0 using Equation 3.12
16                  IF BayScoreTemp > BayScoreBest THEN
17                      BayScoreBest ← BayScoreTemp
18                      BestFeature ← X
19          IF BestFeature ≠ ∅ THEN:
20              S ← add BestFeature to S
21              D_S ← cases in D_S with X = x
22          ELSE
23              EXIT from LOOP

24      F, P ← add S, P(T | Test) that is estimated using Equation 3.1 from training data that satisfy S
25  P_F = (1/b) Σ_h P_h(T | Test)
26  RETURN F with P_F
```

Figure 5. Pseudocode for LazyRF-Bay method.

3.3.3 LazyRF-Ent

The LazyRF-Ent is a personalized random forest method that uses a randomized personalized decision path with entropy score (based on PDP-Ent, described in 3.1.4) rather than the randomized PDP-Bay as the base model. Inference is performed using the average of the individual path predictions in the ensemble, each of which is calculated using Equation 3.14. Although the scores for model construction and inference differ, the LazyRF-Ent has the same model structure and search strategies as LazyRF-Bay.

3.4 The Random Forest Method

We compare the predictive performance of the LazyRF-Bay method with a population random forest method. We describe the model structure, the search strategy, and the scoring criteria of the decision tree methods.

3.4.1 Model Structure and Search Strategy

The random forest method trains an ensemble of randomized decision trees on b bootstrap datasets. The structure of a decision tree is described in 3.2.1. The randomized decision tree is a population method that does not incorporate features from ***Test*** in the model search. The randomized decision tree is a variant of the decision tree described in 3.2 where the search is modified to add one variable at a time from a random selection of variables (rather than the full set of variables). Given a bootstrapped dataset D_b from training dataset D, the method performs a

forward stepping, greedy hill-climbing search to add one variable at a time from a random selection of variables that locally maximizes a scoring criterion. The randomized decision tree method does not perform pruning and terminates when the addition of a variable does not increase the tree's score.

3.4.2 Random Forest with Entropy Score

The random forest with entropy score (RF-Ent) is comprised of randomized decision trees that maximize an entropy information gain scoring criterion given by *TreeEntScore,* described in 3.2.2, to evaluate variables. Each randomized decision tree calculates parameter probabilities using the maximum likelihood estimator as given by Equation 3.14. Like the LazyRF-Ent method, forest parameter estimates are calculated as an average of the parameter estimates from the trees in the forest. This is our implementation of Breiman's random forest.[84]

3.4.3 Random Forest with Bayesian Score

The random forest with Bayesian score (RF-Bay) is comprised of randomized decision trees that maximize a decision trees that maximize a Bayesian scoring criterion, the $TreeBayScore_{K2}$ as described in 3.2.3 using a uniform structure prior. Parameter probabilities are calculated using the K2 estimator as given by Equation 3.1. Like the LazyRF-Bay method, forest parameter estimates are calculated as an average of the parameter estimates from the trees in the forest.

3.5 Boosted Personalized Decision Paths

In this section, we introduce the Boosted PDP-Bay, a boosted ensemble of Bayesian decision paths. It constructs an ensemble of personalized decision paths in sequence, varying the models in the ensemble by altering the training data case weights. Each training data case is weighted according to a relevance metric based on the features present in the decision path generated in the previous iteration.

3.5.1 Background

As mentioned previously, the small amount of work in ensembles of personalized decision paths includes the work of Fern et al.[20] in developing a boosted ensemble of LazyDTs. Traditionally, boosting involves weighting the training cases in a training dataset, training a model, and then using that model to re-weight the training cases, thereby producing an ensemble of diverse models. The re-weighting is typically done by using the model to classify the training cases and then increasing the weights of training cases that are incorrectly classified by the model and decreasing the weights of training cases that are correctly classified by the model. A decision path *S* cannot use this boosting approach, however, as only those training cases that correspond to all the features in *S* can be classified by *S*.

Fern et al.[20] thus developed a novel boosting method for personalized decision paths that re-weights a training case based on the number of features shared with a decision path. They compared the results of single LazyDTs, bagged LazyDTs (BA-LazyDT), and boosted LazyDTs (BO-LazyDT), and demonstrated that using these ensemble approaches with personalized decision

paths improved predictive accuracy over the base method and matched performance of the analogous population method, AdaBoost.

We used the personalized boosting methodology developed by Fern with our novel PDP-Bay method (described in 3.1.2) as the base model to produce a new type of Bayesian personalized ensemble.

3.5.2 The Boosted PDP-Bay Method

We first describe the base model structure, search strategy, and scoring criterion of the Boosted PDP-Bay (BO-PDP-Bay).

3.5.2.1 Base Model Structure

The ensemble produced by the BO-PDP-Bay method consists of a forest of decision-path models. Detailed terminology regarding the structure of decision-paths is outlined in section 3.1.2.1. The base model of the BO-PDP-Bay is the Personalized Decision Path that uses a Bayesian score (PDP-Bay), which is described in section 3.1.2.

3.5.2.2 Search Strategy

The pseudocode for the BO-PDP-Bay method is shown in Figure 5. Given a training dataset D of m training cases and a test case ***Test***, the method starts with uniform weights $W = (w_1, \ldots, w_m)$ where $w = 1$ for all of the training cases in D. A decision path S is trained in the manner described in section 3.1.2.2 with the $BayScore_{BDeu}$ scoring criterion. To construct each path, the method uses a forward-stepping, greedy hill-climbing search and a Bayesian score as the criterion

for evaluating features for inclusion in the path. Beginning with an empty path *S*, the algorithm successively adds features from ***Test*** to *S* that locally maximize the score, until the score can no longer be improved. Currently, the method is designed to work with discrete data only.

Once *S* is complete, the features in *S* and the corresponding predicted value for the target class are used to update *W* (the weights of the training cases). The predicted value for the target class t_S is determined by identifying the target value with the highest probability according to *S*. Weights are increased for training cases with target values that differ from t_S, while weights are decreased for training cases that share t_S as a target value.

Training cases that share more features with *S* have greater "relevance" to ***Test*** and their weights are scaled accordingly. Recall that a decision path $S = (X_1 = x_1 \wedge X_2 = x_2 \wedge \ldots X_j = x_j \wedge \ldots \wedge X_q = x_q)$ contains *q* features and can be organized in a hierarchical manner as nodes in a graph, starting with the initial "root node" $X_1 = x_1$, and terminating with a leaf node representing the probability distribution of the target variable. Each node represents a different "level" of the path. To calculate the relevance *R* of a training case ***Train***, the number of levels shared by ***Train*** and *S* are counted, starting with the root node $X_1 = x_1$. If ***Train*** does not take the same value as the root node, $R = 0$, and if ***Train*** shares all features with all levels of *S*, $R = q$ (the total number of levels in *S*). As the order of features in *S* is necessary to calculate relevance, it is possible that relevance is not necessarily equal to the total number of features shared by ***Train*** and *S*.

If ***Train*** has the same target value as the predicted target value from *S*, its weight is multiplied by α^R, otherwise its weight is multiplied by β^R, where $\alpha < 1$ and $\beta > 1$. When the weights for all training cases in *D* have been updated, the weights are normalized. A new decision path can then be trained on D_W, the training data weighted using the updated weights in *W*. This process is

repeated *b* times, producing an ensemble ***F*** of *b* decision paths. In this research, the number of paths *b* was set to 10, $\alpha = 0.98$, and $\beta = 1.15$, to match the parameters used by Fern et al.[20]

BO-PDP-Bay (D, b, ***Test***, α, β)
INPUT: D is a training dataset of m cases and n variables and target variable T,
b is the number of models to train (size of ensemble),
Test is data for a test case that is not in D and whose T is to be predicted
α and β are scaling hyperparameters where $\alpha < 1$ and $\beta > 1$
OUTPUT: F, where F is a forest of personalized paths ***S*** for the test case and P_F is the average of the estimates of $P(T \mid \textbf{\textit{Test}})$ from each path

$W \leftarrow m$ initial uniform weights for cases in training data
$F, P \leftarrow$ empty forest and associated probabilities

LOOP b times
  $D_W \leftarrow$ training dataset D with cases weighted by corresponding values in W
  $S \leftarrow$ path trained using *Test*, D_W, $\alpha_0 = 2$, and $e = 0$ using PDP-Bay algorithm and $BayScore_{BDeu}$
  $F, P \leftarrow$ add S, $P(T \mid \textbf{\textit{Test}})$ that is estimated using Equation 3.1 from training data that satisfy S
  FOR each training case $\textbf{\textit{Train}}_l$ in D:
    $R \leftarrow$ number of levels shared by $\textbf{\textit{Train}}_l$ with S
    $C \leftarrow$ target class with highest probability $P(T \mid \textbf{\textit{Test}})$
    IF C is the same target class as $\textbf{\textit{Train}}_l$:
      $C_l \leftarrow 0$
    ELSE:
      $C_l \leftarrow 1$
    $W_l \leftarrow W_l [(1 - C_l)\, \alpha^R + C_l\, \beta^R]$
  $W_{sum} \leftarrow$ sum of all weights in W
  FOR W_l in W:
    $W_l = W_l [m / W_{sum}]$

$P_F = \frac{1}{b} \sum_{h=1}^{b} P_h(T \mid \textbf{\textit{Test}})$
RETURN F with P_F

Figure 6. Pseudocode for BO-PDP-Bay method.

3.5.2.3 Bayesian Scoring Criterion

The score of a candidate path *S'* is calculated using the $BayScore_{BDeu}$ given by Equation 3.10 in 3.1.2.3 using a uniform structure prior and parameter prior $\alpha_0 = 2$. $BayScore_{BDeu}$

incorporates the number of training cases N that correspond to a given path, as well as the number of those cases which take each possible value of the target, denoted as N_k for the k^{th} value of the target. When calculating N and the values of N_k for a given path, the counts of the training cases are scaled according to their corresponding weights in W. This score selects features that maximize the posterior probability of the model structure given the corresponding data. When this score is used with these settings, it produces equivalent results to $BayScore_{K2}$ for discrete values, but it can also handle continuous values which occur when non-integer weights are applied to the training data, resulting in non-integer values for N.

3.5.2.4 Inference

To perform inference for ***Test***, parameter estimates are calculated using Equation 3.1 for each path S in forest $\boldsymbol{F}$. These parameter estimates are then averaged, and the estimate for probability θ_k is therefore calculated in the same manner as LazyRF-Bay, which is given in Equation 3.28 in 3.3.2.4.

3.5.3 Boosted PDP-Ent

The Boosted PDP-Ent (BO-PDP-Ent) is a personalized boosted ensemble method that uses a personalized decision path with entropy score (PDP-Ent) rather than the PDP-Bay as the base model. Inference is performed using the average of the individual path predictions in the ensemble, each of which is calculated using Equation 3.14. Although the scores for model construction and inference differ, the BO-PDP-Ent has the same model structure and search strategies as BO-PDP -Bay. This is our implementation of Fern's BO-LazyDT.[20]

3.6 AdaBoost

AdaBoost is a canonical boosting method that trains an ensemble of decision "stumps" (decision trees that are limited to a single variable) in sequence, using classification information from the previous model to weight training cases to increase the prevalence of misclassified (or "challenging") training cases.

3.6.1 Model Structure

The base model of AdaBoost is a decision stump, which is a decision tree that is limited to a single variable X. The decision-tree model M is represented as $M = (\boldsymbol{T}, \boldsymbol{\theta})$, where $\boldsymbol{T} = (S_1, S_2, \ldots, S_h, \ldots, S_m)$ is a collection of m paths, where each path $S_h = (X = x_h)$ consists of a single feature (value for variable X) and m is the number of values taken by the variable X. Given a training dataset D, the method selects a single variable that locally maximizes a scoring criterion.

3.6.2 Search Strategy

Starting with uniform weights $\boldsymbol{W} = (W_1, \ldots, W_m)$ where $W_l = 1$ for each training data case, a decision tree that is limited to a single variable is trained with the weighted training data. The tree is used to classify the training cases, and the predicted class for each training case $\boldsymbol{Train_l}$ is compared to the true class. $\boldsymbol{C} = (C_1, \ldots, C_m)$ contains information regarding classifications for the m training cases, where $C_l = 0$ if $\boldsymbol{Train_l}$ was classified correctly and $C_l = 1$ if $\boldsymbol{Train_l}$ was misclassified by the tree. The weighted misclassification error is then calculated as

$$Error = \sum_{l=1}^{m} W_l C_l \,/\, \sum_{l=1}^{m} W_l \,, \tag{3.29}$$

where W_l is the weight for ***Train**$_l$*.

This misclassification error is used to calculate a tree weight, defined as

$$\omega_t = \frac{1}{2}\log\left(\frac{1-Error_t}{Error_t}\right), \tag{3.30}$$

which is used to weight tree *t* in the ensemble prediction as well as update training case weights. Weight W_l for ***Train**$_l$* is updated as

$$W_l = (1 - C_l)W_l e^{-\omega_t} + C_l W_l e^{\omega_t}, \tag{3.31}$$

which results in misclassified training cases receiving higher weights and correctly classified cases receiving lower weights. The weights are normalized

$$W_l = W_l \frac{m}{W_{sum}}, \tag{3.32}$$

where *m* is the number of training cases and W_{sum} is the sum of all the weights in *W*. A new decision tree is trained on the re-weighted training data, and this process is repeated *b* times. A prediction for θ_{forest} is made by taking a weighted average of the individual outputs of the trees in the ensemble,

$$\theta_{forest} = \sum_{t=1}^{b} \omega_t \theta_t \,, \tag{3.33}$$

where ω_t is the weight of tree *t* and θ_t is the prediction for parameter θ by tree *t*.

3.6.3 AdaBoost with Entropy Score

AdaBoost with entropy score (AB-Ent) is comprised of boosted decision trees that maximize an entropy information gain scoring criterion given by *TreeEntScore* in 3.2.2 to evaluate variables and calculates parameter probabilities using the maximum likelihood estimator as given

by Equation 3.14. Each decision tree is limited to a single variable. This is our implementation of Freund and Schapire's AdaBoost algorithm.[84]

3.6.4 AdaBoost with Bayesian Score

AdaBoost with Bayesian score (AB-Bay) is comprised of boosted decision trees that maximize a Bayesian scoring criterion given by $TreeBayScore_{BDeu}$ as described in 3.2.3. The structure prior is uniform and $\alpha_0 = 2$ to match the priors used by the Boosted PDP-Bay. Parameter probabilities are calculated using the K2 estimator as given by Equation 3.1. Each decision tree is limited to a single variable.

3.7 Overview of Algorithms

In Table 4, we provide an overview of the algorithmic methods described in this chapter. The primary experimental algorithms are the PDP-Bay, LazyRF-Bay, and BO-PDP-Bay. All other algorithmic methods were used as control algorithms. The novel algorithmic methods include the three primary experimental algorithms (PDP-Bay, LazyRF-Bay, and BO-PDP-Bay) as well as PDP-Bay-BDeu, LazyRF-Ent, and DT-Bay-BDeu.

Table 4. Overview of algorithmic methods with descriptions.

Algorithmic methods in bold were developed for this dissertation. Red font denotes the primary experimental algorithmic methods, and algorithmic methods in italics were used as control algorithms.

Name	Full name	Brief description
PDP-Bay	Personalized Decision Path that uses a K2 Bayesian score	Single model, personalized, Bayesian score with uniform structure prior
PDP-Bay-BDeu	Personalized Decision Path that uses a BDeu Bayesian score	Single model, personalized, Bayesian score with prior hyperparameter tuning
PDP-Ent	Personalized Decision Path that uses an Entropy score	Single model, personalized, entropy score
LazyRF-Bay	Lazy Random Forest that uses a Bayesian score	Ensemble model, personalized, Bayesian score with uniform structure prior
LazyRF-Ent	Lazy Random Forest that uses an Entropy score	Ensemble model, personalized, entropy score
BO-PDP-Bay	Boosted Personalized Decision Path that uses a Bayesian score	Ensemble model, personalized, Bayesian score with uniform structure prior
BO-PDP-Ent	Boosted Personalized Decision Path that uses an Entropy score	Ensemble model, personalized, entropy score
DT-Bay	Decision Tree that uses a K2 Bayesian score	Single model, population, Bayesian score with uniform structure prior
DT-Bay-BDeu	Decision Tree that uses a BDeu Bayesian score	Single model, population, Bayesian score with prior hyperparameter tuning
DT-Ent	Decision Tree that uses an Entropy score	Single model, population, entropy score
RF-Bay	Random Forest that uses a Bayesian score	Ensemble model, population, Bayesian score with uniform structure prior
RF-Ent	Random Forest that uses an Entropy score	Ensemble model, population, entropy score
AB-Bay	Adaptive Boosting of Decision Trees that uses a Bayesian score	Ensemble model, population, Bayesian score with uniform structure prior
AB-Ent	Adaptive Boosting of Decision Trees that uses an Entropy score	Ensemble model, population, entropy score

3.8 Hypotheses

Here we state the three hypotheses of this dissertation.

1. We hypothesize that the novel PDP-Bay method outperforms the entropy-scored personalized decision path and decision tree methods.
2. We hypothesize that the novel LazyRF-Bay method outperforms the single PDP-Bay and random forest methods.
3. We hypothesize that the novel BO-PDP-Bay method outperforms the single PDP-Bay and AdaBoost methods.

The method of personalization used in this dissertation is a simple one that has been shown to improve predictive performance over population methods in non-Bayesian approaches, although the method is prone to overfitting.[14] The use of Bayesian scoring incorporates smoothing, which may mitigate overfitting and improve predictive performance over non-Bayesian personalized methods as well as population methods. Furthermore, ensemble approaches often result in better predictive performance than single model methods, such as in the case of boosting non-Bayesian decision paths.[20] Ensemble approaches like boosting and random forest combined with Bayesian scoring may result in further performance improvements for personalized decision paths. Overall, we hypothesize that personalized methods outperform population methods, that Bayesian methods outperform non-Bayesian methods, and that ensemble approaches outperform single model methods.

4.0 Experimental Methods

This chapter provides details of the experimental methods used for evaluating the novel algorithms used in this dissertation. Section 4.1 briefly describes the datasets, and Sections 4.2-4.4 describe the evaluation metrics, the statistical tests, and the algorithmic comparisons.

4.1 Datasets

We used the following datasets for evaluation of our methods. There are thirteen main datasets that include synthetic, genomic, and clinical datasets. We provide brief descriptions of the datasets in Table 5.

4.1.1 Chronic Pancreatitis Dataset

The chronic pancreatitis dataset was collected as part of the multicenter North American Pancreatitis Study 2.[89] It consists of the predictor variables, which are 142 SNVs, and a binary target variable (developed chronic pancreatitis or not). The data were previously de-identified and consist of 2,201 patients, 980 (44.5%) of whom were diagnosed with chronic pancreatitis, and 1,221 of whom were not.

Table 5. Datasets used in experimental evaluation.

Dataset	# Variables	# Values per variable	# Target values	# of Cases	# of Positive Cases (% of dataset)
chronic-pancreatitis	142	3	2	2201	980 (44.5%)
pneumonia	156	2-8	2	2287	261 (11.4%)
sepsis-d	19	2-5	2	1673	189 (11.3%)
sepsis-s	18	2-5	2	1673	478 (28.6%)
heart-failure-d	17	2-7	2	11,178	500 (4.47%)
heart-failure-c	20	2-7	2	11,178	1255 (11.2%)
synthetic-large	1000	3	2	10,000	1270 (12.7%)
synthetic-small	35	3	2	10,000	1270 (12.7%)
cleveland	13	1-4	2	296	136 (45.9%)
breast	30	1-4	2	569	212 (37.3%)
hepatitis	19	1-2	2	80	13 (16.3%)
heart	13	1-2	2	270	150 (55.6%)
diabetes	8	1-4	2	768	268 (34.1%)

4.1.2 Pneumonia Dataset

The pneumonia dataset was collected by the Pneumonia Patient Outcomes Research Team in a multisite study.[90] The dataset consists of 2,287 adult patients admitted with community acquired pneumonia. From data collected at the time of presentation, we have 156 predictor variables, which include clinical, laboratory, and radiographic findings. The target variable is a binary variable called dire outcome. A patient was considered to have experienced a dire outcome if any of the following occurred: 1) death within 30 days of presentation, 2) an initial intensive care unit admission for respiratory failure, respiratory or cardiac arrest, or shock, or 3) the presence of one or more specific, severe complications. According to these criteria, 261 patients (11.4%) experienced a dire outcome and 2,026 did not.

4.1.3 Sepsis Dataset

The sepsis dataset was collected in the multisite Genetic and Inflammatory Markers of Sepsis (GenIMS) project.[91] Data were collected on 1,673 patients who were admitted from an emergency department with a diagnosis of community acquired pneumonia. From the data collected at the time of presentation, we have 19 predictor variables that consist of demographic, clinical, and genetic findings as well as inflammatory markers. Two binary outcome variables were used: 1) death within 90 days of enrollment in the study, which was true for 189 patients (11.3%, sepsis-d dataset) and 2) development of severe sepsis during hospitalization, which was true for 478 patients (28.6%, sepsis-s dataset).

4.1.4 Heart Failure Dataset

The heart failure dataset was collected by 192 hospitals in Pennsylvania and consists of 11,178 patients who presented in emergency departments and were admitted with a diagnosis of heart failure.[92] There are 20 predictor variables that consist of demographic, clinical, laboratory, electrocardiographic, and radiographic findings. Two binary outcome variables were used: 1) death from any cause during hospitalization, which was true for 500 patients (4.47%, heart failure-d dataset) and 2) development of one or more severe complications (including death) during hospitalization, which was true for 1,255 patients (11.2%, heart failure-c dataset).

4.1.5 Synthetic Dataset

The synthetic datasets were generated using R code to generate synthetic genomic datasets. Each predictor variable represents a single nucleotide variant (SNV), which is a variation in a single nucleotide in a specific position in the genome. For a given SNV, alleles can be designated as "major" (the most commonly appearing allele in members of a population) and "minor" (the second most commonly appearing allele in members of a population), and the minor allele frequency (MAF) quantifies how frequently the minor allele appears in a population. It is thought that SNVs with rare MAFs can have a large impact on susceptibility to certain diseases.[93]

For the SNVs in the synthetic datasets, the binary disease variable was modeled as a function of 35 "signal" SNVs. To generate the MAFs and odds ratios (ORs) for these 35 SNVs, random values were sampled from a uniform distribution between a minimum and maximum value using the "runif" function. Of the 35 signal SNVs, 25 were "rare" with MAFs sampled from (0.0001, 0.01), and 10 were "common" with MAFs sampled from (0.05, 0.50). For each signal SNV, ORs were also randomly sampled, with values between (2, 10) for the 25 rare SNVs and between (1.05, 1.50) for the 10 common SNVs.

For each case, two random values for alleles were sampled from a uniform distribution from (0, 1) for each SNV, and if the value for an allele was less than the MAF for that SNV, the case was assigned a "1" for the presence of a rare allele. The values for the two alleles were then summed to determine a value for the SNV predictor variable. Since each case has two possible alleles for each SNV, the case could be assigned a total value of 0 (no rare alleles), 1 (one rare allele), or 2 (two rare alleles).

To determine the target value for each case, the value for each signal SNV was multiplied by the OR of that SNV to produce a risk value, and these products were summed across the 35 signal SNVs to calculate a total risk for the case. If a case's total risk surpassed a given threshold (set to 10), the case was designated as "disease."

The small dataset consists only of the 35 "signal" SNVs and the binary disease variable. The large synthetic dataset consists of 1,000 single nucleotide variants (SNVs) as predictor variables and the binary disease variable. The large synthetic dataset contains 965 "noise" SNVs in addition to the 35 signal SNVs, and these were randomly assigned MAFs from common to rare and had no effect on the disease status. Both datasets consist of 10,000 "patients," of which 1270 (12.7%) have a disease target value.

4.1.6 UCI Datasets

There are five additional health-related datasets from University of California at Irvine's (UCI) public dataset repository. These have been used as benchmark datasets for machine learning methods in a number of publications. The cleveland dataset[94] comes from Dr. Robert Detrano at the Cleveland Clinic Foundation and contains clinical and demographic data from 303 patients, 139 of whom (45.9%) were found to have clinically significant heart disease. The breast dataset[95] comes from the University of Wisconsin and contains pathology data from breast tissue samples from 699 women, 241 of whom (34.5%) were found to have malignant breast masses. The hepatitis dataset[94] comes from the Jozef Stefan Institute in Yugoslavia and contains clinical and demographic data from 155 patients with hepatitis, 32 of whom (20.6%) died. The heart dataset[96] comes from Faisalabad Institute of Cardiology in Pakistan and contains clinical and demographic

data from 299 patients with heart failure, of whom 96 (32.1%) died. The diabetes dataset[97] comes from the National Institute of Diabetes and Digestive and Kidney Diseases and contains clinical and demographic data from 768 female patients of Pima Indian heritage, of whom 268 (34.9%) had diabetes.

Each dataset that contained more than 1000 cases was divided into approximately 80%/20% train/test splits with approximately equal prevalence of the positive target in train and test datasets. We performed stratified 10-fold cross-validation twice (2x10FCV) on the UCI datasets, resulting in 20 train and test datasets. The predictions from individual folds were then aggregated to calculate the evaluation metrics.

4.2 Evaluation Metrics

In this section, we describe the metrics used to evaluate the methods as well as the statistical methods used to compare results.

4.2.1 Discrimination

Discrimination characterizes the ability of a classifier to differentiate between target classes. For binary targets, this can be measured by the area under the receiver operating characteristic curve (AUROC), which represents the probability that a randomly selected case from cases with the positive target class will be assigned a higher probability of a positive outcome by

the classifier than a randomly selected case from the negative class. The best possible value for AUROC is 1 and the worst is 0.5.

4.2.2 Calibration

Calibration characterizes the ability of a classifier to assign an accurate probability estimate of a positive outcome to a case. For example, we would expect that, of the cases assigned a 90% probability of a positive outcome by a well-calibrated model, 90% of these cases would actually have a positive target value. This can be measured by expected calibration error (ECE), which approximates the true calibration error by discretizing probability estimates and assigning cases into their corresponding bins.[98] The ECE is then given by the following equation:

$$ECE = \sum_{i=1}^{K} P(i)|\, o_i - \, e_i|,$$

where K is the number of bins, $P(i)$ is the fraction of cases corresponding to bin i, o_i is the fraction of positive cases in bin i, and e_i is mean of the predicted probabilities that the target value is positive for each of the cases in bin i. The best possible value for ECE is 0. For our experiments, K was set to 10.

4.2.3 Model Complexity

To characterize model complexity, we measure the mean path length (MPL) for each case in the test dataset. Each test case corresponds to a single path in a given model, which is either a standalone personalized path or a path from a decision tree, and path length is defined as the number of features in the path. For ensembles, the mean path length for each test case is calculated

as the average of the path lengths in the ensemble used for prediction, and these average path lengths are averaged across all test cases to produce the overall mean path length. Simpler models require fewer features to make a prediction and are likely to be easier to comprehend. Thus, shorter mean path lengths are considered better for the purposes of this work.

4.3 Statistical Tests

Measures were compared between methods across all datasets using Wilcoxon Signed Rank Test. Additionally, AUROCs were compared pairwise on individual datasets using DeLong's test. These are nonparametric tests that do not require the distribution of the data to be normal. Each dataset was randomly split approximately 80%/20% into a training and a test set such that the proportion of positive cases was the same across each pair of training and test sets. We trained models using the training sets and performed the evaluations on the test sets. We evaluated the methods on discrimination, calibration, and model complexity. For model complexity, we measured the average predictive path length and defined path length as the number of variables in the path that was used for inference. For ensembles, we calculated the average path length of the predictive paths in the ensemble.

4.4 Algorithmic Comparisons

An overview of the main comparisons can be found in Table 6.

Table 6. Overview of comparisons.

	Primary Method	**Comparison Methods**
Single Model Methods	PDP-Bay	DT-Bay, PDP-Ent, DT-Ent
	PDP-Bay	PDP-Bay-BDeu
	DT-Bay	DT-Bay-BDeu
Ensemble Methods	LazyRF-Bay	LazyRF-Ent, RF-Bay, RF-Ent
	BO-PDP-Bay	AB-Bay, BO-PDP-Ent, AB-Ent

We compared the predictive performance of the novel PDP-Bay to a personalized decision path with entropy score (PDP-Ent) and population decision trees, one with an entropy score (DT-Ent) and the other with a Bayesian score (DT-Bay), on the datasets listed in Table 5. By comparing PDP-Bay to PDP-Ent, we evaluate potential differences in predictive performance associated with Bayesian scoring. By comparing PDP-Bay to DT-Bay, we evaluate potential differences in predictive performance associated with personalization. By comparing PDP-Bay to DT-Ent, we evaluate potential differences in predictive performance associated with the combination of both personalization and Bayesian scoring, and compare our novel method (PDP-Bay) with an implementation of a commonly used machine learning algorithm (DT-Ent).

We compared the predictive performance of PDP-Bay with PDP-Bay-BDeu and the performance of DT-Bay with DT-Bay-BDeu on the datasets listed in Table 5. By comparing PDP-Bay to PDP-Bay-BDeu, we evaluate potential differences in predictive performance associated with the use of hyperparameter optimization in the context of personalized methods, and by comparing DT-Bay to DT-Bay-BDeu, we evaluate potential differences in predictive performance associated with the use of hyperparameter optimization in the context of population methods.

We compared the predictive performance of the novel LazyRF-Bay to the base model, PDP-Bay, and population random forest using both entropy (RF-Ent) and Bayesian (RF-Bay)

scores on the datasets listed in Table 5. By comparing LazyRF-Bay to PDP-Bay, we evaluate potential differences in predictive performance associated with the use of the random forest ensemble approach in the context of personalized Bayesian algorithmic methods. By comparing LazyRF-Bay to LazyRF-Ent, we evaluate potential differences in predictive performance associated with Bayesian scoring. By comparing LazyRF-Bay to RF-Bay, we evaluate potential differences in predictive performance associated with personalization. By comparing LazyRF-Bay to RF-Ent, we evaluate potential differences in predictive performance associated with the combination of both personalization and Bayesian scoring, and compare our novel method (LazyRF-Bay) with an implementation of a commonly used machine learning algorithm (RF-Ent).

We compared the predictive performance of the novel BO-PDP-Bay to the base model PDP-Bay, boosted decision paths with an entropy score (BO-PDP-Ent), and boosted ensembles of decision trees using both entropy (AB-Ent) and Bayesian (AB-Bay) scores on the datasets listed in Table 5. By comparing BO-PDP-Bay to PDP-Bay, we evaluate potential differences in predictive performance associated with the use of the personalized boosting approach in the context of personalized Bayesian algorithmic methods. By comparing BO-PDP-Bay to BO-PDP -Ent, we evaluate potential differences in predictive performance associated with Bayesian scoring. By comparing BO-PDP -Bay to AB-Bay, we evaluate potential differences in predictive performance associated with personalization. By comparing BO-PDP -Bay to AB-Ent, we evaluate potential differences in predictive performance associated with the combination of both personalization and Bayesian scoring, and compare our novel method (BO-PDP -Bay) with an implementation of a commonly used machine learning algorithm (AB-Ent).

We compared the predictive performance of all personalized methods described above (PDP-Bay, PDP-Bay-BDeu, PDP-Ent, LazyRF-Bay, LazyRF-Ent, BO-PDP-Bay, and BO-PDP-

Ent) with that of all population methods described above (DT-Bay, DT-Bay-BDeu, DT-Ent, RF-Bay, RF-Ent, AB-Bay, and AB-Ent). By comparing all personalized methods with corresponding population methods, we evaluate potential differences in predictive performance associated with our method of personalization.

We compared the predictive performance of all Bayesian methods described above (PDP-Bay, DT-Bay, LazyRF-Bay, RF-Bay, BO-PDP-Bay, and AB-Bay) with that of all non-Bayesian methods described above (PDP-Ent, DT-Ent, LazyRF-Ent, RF-Ent, BO-PDP-Ent, and AB-Ent). By comparing all Bayesian methods with corresponding non-Bayesian methods, we evaluate potential differences in predictive performance associated with our method of model scoring.

4.5 Bayesian Prior Hyperparameter Settings

For the Bayesian algorithmic methods, there were several possibilities for Bayesian scores and prior hyperparameter settings. Table 7 lists the specific score used and the prior hyperparameters used for the experiments.

Table 7. Prior hyperparameter values for Bayesian algorithmic methods.

Algorithmic Method	**Score**	**Parameter Prior** (α_0)	**Structure Prior** (e)
PDP-Bay	$BayScore_{K2}$	n/a	0
DT-Bay	$TreeBayScore_{K2}$	n/a	0
PDP-BDeu	$BayScore_{BDeu}$	[0.1, 0.5, 1.0, 2.0, 5.0, 10.0]	[0, 1, 3, 6]
DT-Bay-BDeu	$TreeBayScore_{BDeu}$	[0.1, 0.5, 1.0, 2.0, 5.0, 10.0]	[0, 1, 3, 6]
LazyRF-Bay	$BayScore_{K2}$	n/a	0
RF-Bay	*Tree* $BayScore_{K2}$	n/a	0
BO-PDP-Bay	$BayScore_{BDeu}$	2	0
AB-Bay	$TreeBayScore_{BDeu}$	2	0

A value of zero for the structure prior indicates that a uniform distribution was used for the structure prior probability distribution.

4.6 Implementation

We statistically compared the methods for each of the evaluation measures across the datasets with the Wilcoxon signed-rank test statistic using the R function “wilcox.test”. For discrimination, we computed the AUROCs and performed pairwise comparison of AUROCs given specific datasets with DeLong’s test using the R package “pROC”. We implemented the three methods in Python (version 3.7). We performed all experiments on a MacBook Pro with a 3.3 GHz Dual-Core Intel Core i5 processor and 16GB of RAM, running the 64-bit macOS Catalina operating system.

5.0 Results

This chapter presents the results of evaluation of the novel personalized algorithmic methods. Section 5.1 focuses on the PDP-Bay and PDP-BDeu methods, Section 5.2 focuses on the LazyRF-Bay method, and Section 5.3 focuses on the Boosted PDP-Bay method. Each of the personalized methods is compared to several comparison methods including population algorithmic methods. The evaluation results are presented in terms of discrimination (measured with AUROC), calibration (measured with ECE), and model complexity (measured with MPL). In most cases, the personalized methods were statistically compared to the comparison methods with the Wilcoxon signed-rank test.

5.1 PDP-Bay and Single Model Methods

This section presents the results of the personalized PDP-Bay method and compares its performance to that of population methods, DT-Bay and DT-Ent, and the entropy-scored personalized method, PDP-Ent (see Table 4 for descriptions of algorithmic methods). All of these methods derive a single model: PDP-Bay and PDP-Ent derive a single decision path model and DT-Bay and DT-Ent derive a single decision tree model.

5.1.1 Discrimination

The mean AUROC for the PDP-Bay method was 0.78. Mean AUROCs for comparison methods were 0.80 for DT-Bay, 0.70 for PDP-Ent, and 0.71 for DT-Ent. AUROCs for each method and dataset are listed in Table 8.

PDP-Bay did not have statistically significantly different mean AUROCs than DT-Bay ($p = 0.273$), and had statistically significantly higher mean AUROCs than PDP-Ent ($p = 0.002$) and DT-Ent ($p = 0.006$) at the 0.05 level on the Wilcoxon signed-rank test. In summary, in terms of the AUROC, the PDP-Bay performed better than the entropy-based personalized and population methods and on par with the Bayesian population method.

Table 8. AUROCs for PDP-Bay, DT-Bay, PDP-Ent, and DT-Ent methods.

Results in red font indicate statistically significantly higher AUROCs than that of PDP-Bay, and results in blue font indicate statistically significantly lower AUROCs than that of PDP-Bay on DeLong's test (values marked with * indicate Wilcoxon signed-rank test).

Dataset	PDP-Bay	DT-Bay	PDP-Ent	DT-Ent
chronic-pancreatitis	0.83	0.81 ($p = 0.254$)	0.74 ($p < 0.001$)	0.72 ($p < 0.001$)
pneumonia	0.66	0.82 ($p < 0.001$)	0.51 ($p < 0.001$)	0.67 ($p = 0.875$)
sepsis-d	0.81	0.84 ($p = 0.175$)	0.55 ($p < 0.001$)	0.66 ($p < 0.001$)
sepsis-s	0.74	0.75 ($p = 0.455$)	0.64 ($p < 0.001$)	0.63 ($p < 0.001$)
heart-failure-d	0.69	0.73 ($p = 0.084$)	0.54 ($p < 0.001$)	0.54 ($p < 0.001$)
heart-failure-c	0.64	0.73 ($p < 0.001$)	0.52 ($p < 0.001$)	0.58 ($p < 0.001$)
synth-large	0.77	0.63 ($p < 0.001$)	0.81 ($p = 0.080$)	0.59 ($p < 0.001$)
synth-small	0.82	0.70 ($p < 0.001$)	0.83 ($p = 0.739$)	0.66 ($p < 0.001$)
cleveland	0.83	0.88 ($p < 0.001$)	0.76 ($p < 0.001$)	0.79 ($p = 0.025$)
breast	0.97	0.98 ($p = 0.172$)	0.95 ($p < 0.001$)	0.95 ($p = 0.010$)
hepatitis	0.77	0.79 ($p = 0.453$)	0.63 ($p = 0.049$)	0.84 ($p = 0.069$)
heart	0.85	0.86 ($p = 0.252$)	0.81 ($p = 0.002$)	0.82 ($p = 0.170$)
diabetes	0.83	0.83 ($p = 0.894$)	0.82 ($p = 0.019$)	0.80 ($p = 0.003$)
Mean	0.78	0.80 ($p = 0.273$)	0.70* ($p = 0.002$)	0.71* ($p = 0.006$)

5.1.2 Calibration

The mean ECE for PDP-Bay was 0.13, for DT-Bay mean ECE was 0.05, for PDP-Ent mean ECE was 0.14, and for DT-Ent mean ECE was 0.15. ECEs for each method and dataset are listed in Table 9.

Table 9. ECEs for PDP-Bay, DT-Bay, PDP-Ent, and DT-Ent methods.

Results in red font indicate statistically significantly lower mean ECEs than that of PDP-Bay, and results in blue font indicate statistically significantly higher mean ECEs than that of PDP-Bay on the Wilcoxon signed-rank test.

Dataset	PDP-Bay	DT-Bay	PDP-Ent	DT-Ent
chronic-pancreatitis	0.25	0.10	0.26	0.25
pneumonia	0.11	0.09	0.11	0.14
sepsis-d	0.09	0.03	0.10	0.15
sepsis-s	0.14	0.04	0.21	0.26
heart-failure-d	0.04	0.01	0.04	0.09
heart-failure-c	0.09	0.04	0.11	0.18
synth-large	0.05	0.13	0.07	0.19
synth-small	0.07	0.02	0.07	0.10
cleveland	0.21	0.06	0.25	0.20
breast	0.04	0.02	0.05	0.05
hepatitis	0.13	0.07	0.12	0.12
heart	0.24	0.06	0.26	0.17
diabetes	0.17	0.03	0.18	0.09
Mean	0.13	0.05 (p = 0.006)	0.14 (p = 0.002)	0.15 (p = 0.191)

PDP-Bay had statistically significantly higher mean ECEs than DT-Bay ($p = 0.006$), statistically significantly lower mean ECEs than PDP-Ent ($p = 0.002$), and did not have statistically significantly different mean ECEs than DT-Ent ($p = 0.191$) at the 0.05 level on the Wilcoxon

signed-rank test. In summary, the Bayesian population method had the best calibration and the entropy-based population method had the worst calibration with the personalized methods performing in between.

5.1.3 Model Complexity

The mean MPL for PDP-Bay was 4.38, for DT-Bay mean MPL was 4.35, for PDP-Ent mean MPL was 4.00, and for DT-Ent mean MPL was 6.02. MPLs for each method and dataset are listed in Table 10.

Table 10. MPLs for PDP-Bay, DT-Bay, PDP-Ent, and DT-Ent methods.

Results in blue font indicate statistically significantly longer mean MPLs than that of PDP-Bay, and results in red font indicate statistically significantly shorter mean MPLs than that of PDP-Bay on the Wilcoxon signed-rank test.

Dataset	PDP-Bay	DT-Bay	PDP-Ent	DT-Ent
chronic-pancreatitis	2.76	5.74	2.44	5.25
pneumonia	2.97	8.44	2.36	4.79
sepsis-d	3.53	2.08	3.15	5.79
sepsis-s	4.27	2.16	3.95	6.26
heart-failure-d	6.37	3.18	5.96	7.80
heart-failure-c	8.20	5.10	7.42	8.87
synth-large	2.41	9.44	2.09	8.72
synth-small	13.53	4.69	11.79	9.79
cleveland	3.43	3.93	3.35	4.72
breast	1.53	2.23	1.49	2.89
hepatitis	1.94	2.52	1.64	2.79
heart	3.39	3.82	3.51	5.55
diabetes	2.65	3.20	2.84	5.05
Mean	4.38	4.35 (p = 1.00)	4.00 (p = 0.005)	6.02 (p = 0.017)

PDP-Bay did not have statistically significantly different mean MPLs than DT-Bay ($p = 1.00$), had statistically significantly longer mean MPLs than PDP-Ent ($p = 0.005$) and had statistically significantly shorter mean MPLs than DT-Ent ($p = 0.017$) at the 0.05 level on the Wilcoxon signed-rank test. In summary, the entropy-based personalized method had the simplest models, while the entropy-based population method had the most complex models. The Bayesian personalized and population methods fell in between.

5.1.4 PDP-Bay-BDeu

This section presents the results of the personalized PDP-Bay-BDeu method and population DT-Bay-BDeu method and compares their performance to that of the personalized PDP-Bay and population DT-Bay methods (see Table 4 for descriptions of algorithmic methods). All of these methods derive a single model: PDP-Bay and PDP-Bay-BDeu derive a single decision path model and DT-Bay and DT-Bay-BDeu derive a single decision tree model. The PDP-Bay-BDeu and DT-Bay-BDeu methods perform hyperparameter tuning to select optimized parameter and structure priors, while the PDP-Bay and DT-Bay use simple uniform priors.

5.1.4.1 Discrimination

The mean AUROC of the PDP-Bay was 0.78, and for the PDP-Bay-BDeu mean AUROC was 0.80. The mean AUROC of the DT-Bay was 0.80, and for the PDP-Bay-BDeu mean AUROC was 0.79. AUROCs for each method and dataset are listed in Table 11.

Table 11. AUROCs for PDP-Bay, PDP-BDeu, DT-Bay, and DT-BDeu methods. Results in red font indicate statistically significantly higher AUROCs than that of corresponding non-optimized method (PDP-Bay for PDP-Bay BDeu and DT-Bay for DT-Bay-BDeu), and results in blue font indicate statistically significantly lower AUROCs than that of corresponding non-optimized method on DeLong's test.

Dataset	PDP-Bay	PDP-Bay-BDeu	DT-Bay	DT-Bay-BDeu
chronic-pancreatitis	0.83	0.84 (p = 0.548)	0.81	0.83 (p = 0.209)
pneumonia	0.66	0.76 (p = 0.003)	0.82	0.77 (p = 0.050)
sepsis-d	0.81	0.84 (p = 0.216)	0.84	0.83 (p = 0.331)
sepsis-s	0.74	0.75 (p = 0.498)	0.75	0.76 (p = 0.673)
heart-failure-d	0.69	0.72 (p = 0.075)	0.73	0.71 (p = 0.160)
heart-failure-c	0.64	0.69 (p < 0.001)	0.73	0.74 (p = 0.677)
synth-large	0.77	0.75 (p = 0.395)	0.63	0.69 (p = 0.034)
synth-small	0.82	0.86 (p = 0.015)	0.70	0.66 (p = 0.078)
cleveland	0.83	0.81 (p = 0.033)	0.88	0.82 (p < 0.001)
breast	0.97	0.98 (p = 0.003)	0.98	0.98 (p = 0.092)
hepatitis	0.77	0.79 (p = 0.816)	0.79	0.84 (p = 0.212)
heart	0.85	0.83 (p = 0.003)	0.86	0.81 (p < 0.001)
diabetes	0.83	0.82 (p = 0.798)	0.83	0.79 (p < 0.001)
Mean	0.78	0.80 (p = 0.110)	0.80	0.79 (p = 0.497)

PDP-Bay did not have statistically significantly different mean AUROCs than PDP-Bay-BDeu ($p = 0.110$), and DT-Bay did not have statistically significantly different mean AUROCs than DT-Bay-BDeu ($p = 0.497$) at the 0.05 level on the Wilcoxon signed-rank test. In summary, in terms of AUROC, the BDeu-scored Bayesian methods performed on par with the K2-scored Bayesian methods.

5.1.4.2 Calibration

The mean ECE for the PDP-Bay was 0.13, and for the PDP-Bay-BDeu mean ECE was 0.11. The mean ECE for the DT-Bay was 0.05, and for the DT-Bay-BDeu mean ECE was 0.04. ECEs for each method and dataset are listed in Table 12.

PDP-Bay-BDeu had statistically significantly lower mean ECEs than PDP-Bay ($p = 0.033$), and DT-Bay did not have statistically significantly different mean ECEs than DT-Bay-BDeu ($p = 0.376$) at the 0.05 level on the Wilcoxon signed-rank test. In summary, in terms of ECE, the personalized BDeu-scored Bayesian methods performed better than the personalized K2-scored Bayesian methods.

Table 12. ECEs for PDP-Bay, PDP-BDeu, DT-Bay, and DT-BDeu methods.

Results in red font indicate statistically significantly lower ECEs than that of PDP-Bay on the Wilcoxon signed-rank test.

Dataset	PDP-Bay	PDP-Bay-BDeu	DT-Bay	DT-Bay-BDeu
chronic-pancreatitis	0.25	0.26	0.10	0.01
pneumonia	0.11	0.07	0.09	0.04
sepsis-d	0.09	0.06	0.03	0.03
sepsis-s	0.14	0.04	0.04	0.04
heart-failure-d	0.04	0.03	0.01	0.01
heart-failure-c	0.09	0.08	0.04	0.02
synth-large	0.05	0.01	0.13	0.02
synth-small	0.07	0.08	0.02	0.02
cleveland	0.21	0.21	0.06	0.10
breast	0.04	0.03	0.02	0.03
hepatitis	0.13	0.12	0.07	0.10
heart	0.24	0.25	0.06	0.09
diabetes	0.17	0.17	0.03	0.02
Mean	0.13	0.11 (p = 0.033)	0.05	0.04 (p = 0.376)

5.1.4.3 Model Complexity

The mean MPL for the PDP-Bay was 4.38, and for the PDP-Bay-BDeu mean MPL was 3.53. The mean MPL for the DT-Bay was 4.35, and for the DT-Bay-BDeu mean MPL was 2.58. MPLs for each method and dataset are listed in Table 13.

PDP-Bay-BDeu had statistically significantly shorter mean MPLS than PDP-Bay-BDeu (p = 0.017), and DT-Bay had statistically significantly shorter mean MPLS than DT-Bay-BDeu (p = 0.003) at the 0.05 level on the Wilcoxon signed-rank test. In summary, in terms of MPL, the BDeu-scored Bayesian methods produced simpler models than the K2-scored Bayesian methods.

Table 13. MPLs for PDP-Bay, PDP-BDeu, DT-Bay, and DT-BDeu methods.

Results in red font indicate statistically significantly shorter MPLs than that of PDP-Bay or DT-Bay on the Wilcoxon signed-rank test.

Dataset	PDP-Bay	PDP-Bay-BDeu	DT-Bay	DT-Bay-BDeu
chronic-pancreatitis	2.76	2.69	5.74	1.99
pneumonia	2.97	1.00	8.44	3.39
sepsis-d	3.53	1.00	2.08	2.12
sepsis-s	4.27	1.00	2.16	1.68
heart-failure-d	6.37	6.68	3.18	2.16
heart-failure-c	8.20	8.58	5.10	2.53
synth-large	2.41	1.00	9.44	3.63
synth-small	13.53	12.80	4.69	3.35
cleveland	3.43	3.15	3.93	2.66
breast	1.53	1.18	2.23	2.62
hepatitis	1.94	1.09	2.52	2.64
heart	3.39	3.18	3.82	2.89
diabetes	2.65	2.61	3.20	1.86
Mean	4.38	3.53 ($p = 0.017$)	4.35	2.58 ($p = 0.003$)

5.2 LazyRF-Bay and Random Forest Methods

This section presents the results of the personalized LazyRF-Bay method and compares its performance to that of population methods, RF-Bay and RF-Ent, and the personalized methods, LazyRF-Ent and PDP-Bay (see Table 4 for descriptions of algorithmic methods). All of these methods, except for PDP-Bay, derive ensemble models: LazyRF-Bay and LazyRF-Ent derive ensembles of decision paths and RF-Bay and RF-Ent derive ensembles of trees.

5.2.1 Discrimination

The mean AUROC for LazyRF-Bay was 0.82. Mean AUROCs for comparison methods were 0.78 for PDP-Bay, 0.81 for RF-Bay, 0.80 for LazyRF-Ent, and 0.77 for RF-Ent. AUROCs for each method and dataset are listed in Table 14.

LazyRF-Bay had statistically significantly higher mean AUROCs than PDP-Bay ($p = 0.040$), and did not have statistically significantly different mean AUROCs than RF-Bay ($p = 0.273$), LazyRF-Ent ($p = 0.305$), and RF-Ent ($p = 0.080$) at the 0.05 level on the Wilcoxon signed-rank test. In summary, in terms of AUROC, the LazyRF-Bay performed better than the single-model method and on par with the other ensemble methods.

Table 14. AUROCs for LazyRF- Bay, PDP-Bay, RF-Bay, LazyRF-Ent, and RF-Ent methods. Results in red font indicate statistically significantly higher AUROC than LazyRF-Bay, and results in blue indicate statistically significantly lower AUROC than LazyRF-Bay on DeLong's test (values marked with * indicate Wilcoxon signed-rank test).

Dataset	LazyRF-Bay	PDP-Bay	RF-Bay	LazyRF-Ent	RF-Ent
chronic-pancreatitis	0.83	0.83 (p = 0.786)	0.84 (p = 0.313)	0.82 (p = 0.769)	0.79 (p = 0.44)
pneumonia	0.72	0.66 (p = 0.174)	0.82 (p < 0.001)	0.54 (p < 0.001)	0.74 (p = 0.520)
sepsis-d	0.83	0.81 (p = 0.571)	0.87 (p = 0.035)	0.84 (p = 0.305)	0.85 (p = 0.303)
sepsis-s	0.78	0.74 (p = 0.177)	0.77 (p = 0.170)	0.77 (p = 0.141)	0.75 (p = 0.066)
heart-failure-d	0.75	0.69 (p = 0.051)	0.75 (p = 1.00)	0.72 (p = 0.007)	0.64 (p < 0.001)
heart-failure-c	0.74	0.64 (p < 0.001)	0.77 (p = 0.003)	0.71 (p < 0.001)	0.70 (p = 0.003)
synth-large	0.62	0.77 (p < 0.001)	0.59 (p = 0.546)	0.65 (p = 0.235)	0.49 (p < 0.001)
synth-small	0.93	0.82 (p < 0.001)	0.67 (p < 0.001)	0.91 (p = 0.328)	0.61 (p < 0.001)
cleveland	0.88	0.83 (p = 0.026)	0.90 (p = 0.065)	0.88 (p = 0.090)	0.89 (p = 0.491)
breast	0.99	0.97 (p = 0.028)	0.99 (p = 0.016)	0.99 (p = 0.853)	0.99 (p = 0.025)
hepatitis	0.86	0.77 (p = 0.275)	0.89 (p = 0.372)	0.87 (p = 0.769)	0.86 (p= 0.866)
heart	0.87	0.85 (p = 0.425)	0.88 (p = 0.269)	0.87 (p = 0.288)	0.86 (p = 0.986)
diabetes	0.82	0.83 (p = 0.787)	0.83 (p = 0.037)	0.82 (p = 0.346)	0.82 (p = 0.486)
Mean	0.82	0.78* (p = 0.040)	0.81 (p = 0.273)	0.80 (p = 0.305)	0.77 (p = 0.080)

5.2.2 Calibration

The mean ECE for LazyRF-Bay was 0.08, for PDP-Bay mean ECE was 0.13, for RF-Bay mean ECE was 0.04, for LazyRF-Ent mean ECE was 0.08, and for RF-Ent mean ECE was 0.05. ECEs for each method and dataset are listed in Table 15.

Table 15. ECEs for LazyRF-Bay, PDP-Bay, RF-Bay, LazyRF-Ent, and RF-Ent methods. Results in blue font indicate statistically significantly higher mean ECEs than that of LazyRF-Bay, and results in red font indicate statistically significantly lower mean ECEs than that of LazyRF-Bay on the Wilcoxon signed-rank test.

Dataset	LazyRF-Bay	PDP-Bay	RF-Bay	LazyRF-Ent	RF-Ent
chronic-pancreatitis	0.08	0.25	0.03	0.10	0.08
pneumonia	0.11	0.11	0.01	0.11	0.04
sepsis-d	0.07	0.09	0.05	0.08	0.03
sepsis-s	0.07	0.14	0.06	0.10	0.07
heart-failure-d	0.03	0.04	0.01	0.04	0.04
heart-failure-c	0.07	0.09	0.03	0.08	0.06
synth-large	0.12	0.05	0.04	0.14	0.09
synth-small	0.09	0.07	0.02	0.08	0.04
cleveland	0.06	0.21	0.05	0.04	0.05
breast	0.04	0.04	0.04	0.04	0.03
hepatitis	0.09	0.13	0.08	0.13	0.09
heart	0.07	0.24	0.05	0.06	0.07
diabetes	0.13	0.17	0.04	0.13	0.02
Mean	0.08	0.13 ($p = 0.033$)	0.04 ($p < 0.001$)	0.08 ($p = 0.340$)	0.05 ($p = 0.002$)

LazyRF-Bay had statistically significantly lower mean ECEs than PDP-Bay ($p = 0.033$), had statistically significantly higher mean ECEs than RF-Bay ($p < 0.001$) and RF-Ent ($p = 0.002$),

and did not have statistically significantly different mean ECEs than LazyRF-Ent ($p = 0.340$) at the 0.05 level on the Wilcoxon signed rank test. In summary, in terms of ECE, the population ensemble methods had the best calibration, the single-model method had the worst calibration, and the personalized ensemble methods performed in between.

5.2.3 Model Complexity

The mean MPL for LazyRF-Bay was 3.95, for PDP-Bay mean MPL was 4.38, for RF-Bay mean MPL was 3.87, for LazyRF-Ent mean MPL was 3.94, and for RF-Ent mean MPL was 5.76. MPLs for each method and dataset are listed in Table 16.

LazyRF-Bay did not have statistically significantly different mean MPLs than PDP-Bay ($p = 0.376$), RF-Bay ($p = 0.080$), and LazyRF-Ent ($p = 0.839$), and had statistically significantly shorter mean MPLs than RF-Ent ($p = 0.021$) at the 0.05 level on the Wilcoxon signed rank test. In summary, in terms of MPL, LazyRF-Bay produced simpler models than the entropy-based population ensemble, and was on par with the other Bayesian personalized and population methods as well as the entropy-based personalized ensemble method.

Table 16. MPLs for LazyRF-Bay, PDP-Bay, RF-Bay, LazyRF-Ent, and RF-Ent methods. Results in blue font indicate statistically significantly longer mean MPLs than that of LazyRF-Bay on the Wilcoxon signed-rank test.

Dataset	LazyRF-Bay	PDP-Bay	RF-Bay	LazyRF-Ent	RF-Ent
chronic-pancreatitis	4.00	2.76	4.68	3.73	5.09
pneumonia	4.84	2.97	6.28	4.18	5.66
sepsis-d	2.74	3.53	3.07	3.00	5.82
sepsis-s	3.06	4.27	3.51	3.41	5.95
heart-failure-d	4.66	6.37	3.83	4.89	8.34
heart-failure-c	5.84	8.20	5.88	6.08	9.86
synth-large	4.15	2.41	6.61	3.78	7.63
synth-small	12.41	13.53	2.09	12.17	6.71
cleveland	2.59	3.43	3.73	2.74	5.01
breast	1.92	1.53	2.13	1.98	2.59
hepatitis	1.47	1.94	1.86	1.56	2.74
heart	2.33	3.39	3.67	2.41	5.30
diabetes	1.29	2.65	3.00	1.33	4.21
Mean	3.95	4.38 (p = 0.376)	3.87 (p = 0.080)	3.94 (p = 0.839)	5.76 (p = 0.021)

5.3 BO-PDP-Bay and Boosted Methods

This section presents the results of the personalized BO-PDP-Bay method and compares its performance to that of population methods, AB-Bay and AB-Ent, and the personalized methods, BO-PDP-Ent and PDP-Bay (see Table 4 for descriptions of algorithmic methods). All of these methods, except for PDP-Bay, derive ensemble models using boosting: BO-PDP-Bay and BO-PDP-Ent derive ensembles of decision paths and AB-Bay and AB-Ent derive ensembles of trees.

5.3.1 Discrimination

The mean AUROC for BO-PDP-Bay was 0.82. Mean AUROCs for comparison methods were 0.78 for PDP-Bay, 0.80 for AB-Bay, 0.80 for BO-PDP-Ent, and 0.81 for AB-Ent. AUROCs for each method and dataset are listed in Table 17.

BO-PDP-Bay did not have statistically significantly different mean AUROCs than PDP-Bay ($p = 0.168$), BO-PDP-Ent ($p = 0.147$), AB-Bay ($p = 0.191$), or AB-Ent ($p = 0.168$) at the 0.05 level on the Wilcoxon signed-rank test.

Table 17. AUROCs for BO-PDP-Bay, PDP-Bay, AB-Bay, BO-PDP-Ent, and AB-Ent methods. Results in red font indicate statistically significantly higher AUROC than BO-PDP-Bay, and results in blue font indicate statistically significantly lower AUROC than BO-PDP-Bay on Delong's test.

Dataset	BO-PDP-Bay	PDP-Bay	AB-Bay	BO-PDP-Ent	AB-Ent
chronic-pancreatitis	0.83	0.83 (p = 0.667)	0.83 (p = 0.993)	0.81 (p = 0.097)	0.83 (p = 0.989)
pneumonia	0.78	0.66 (p = 0.005)	0.82 (p = 0.066)	0.64 (p < 0.001)	0.81 (p = 0.170)
sepsis-d	0.85	0.81 (p = 0.066)	0.85 (p = 0.953)	0.81 (p = 0.061)	0.85 (p = 0.798)
sepsis-s	0.77	0.74 (p = 0.169)	0.78 (p = 0.561)	0.75 (p = 0.206)	0.78 (p = 0.539)
heart-failure-d	0.66	0.69 (p = 0.229)	0.73 (p < 0.001)	0.61 (p = 0.003)	0.73 (p < 0.001)
heart-failure-c	0.74	0.64 (p < 0.001)	0.77 (p < 0.001)	0.68 (p < 0.001)	0.77 (p < 0.001)
synth-large	0.92	0.77 (p < 0.001)	0.62 (p < 0.001)	0.90 (p = 0.043)	0.62 (p < 0.001)
synth-small	0.98	0.82 (p < 0.001)	0.62 (p < 0.001)	0.97 (p = 0.134)	0.62 (p < 0.001)
cleveland	0.82	0.83 (p = 0.653)	0.86 (p = 0.004)	0.82 (p = 0.758)	0.89 (p < 0.001)
breast	0.98	0.97 (p = 0.390)	0.99 (p < 0.001)	0.97 (p = 0.435)	0.99 (p = 0.001)
hepatitis	0.79	0.77 (p = 0.901)	0.89 (p = 0.063)	0.86 (p = 0.088)	0.89 (p = 0.069)
heart	0.80	0.85 (p = 0.096)	0.87 (p < 0.001)	0.83 (p = 0.011)	0.88 (p < 0.001)
diabetes	0.79	0.83 (p = 0.012)	0.81 (p = 0.006)	0.80 (p = 0.047)	0.83 (p < 0.001)
Mean	0.82	0.78 (p = 0.168)	0.80 (p = 0.191)	0.80 (p = 0.147)	0.81 (p = 0.168)

5.3.2 Calibration

The mean ECE for BO-PDP-Bay was 0.14. Mean ECEs for comparison methods were 0.13 for PDP-Bay, 0.20 for AB-Bay, 0.13 for BO-PDP-Ent, and 0.20 for AB-Ent. ECEs for each method and dataset are listed in Table 18.

Table 18. ECEs for BO-PDP-Bay, PDP-Bay, AB-Bay, BO-PDP-Ent, and AB-Ent methods.

Dataset	BO-PDP-Bay	PDP-Bay	AB-Bay	BO-PDP-Ent	AB-Ent
chronic-pancreatitis	0.12	0.25	0.35	0.14	0.35
pneumonia	0.05	0.11	0.08	0.10	0.08
sepsis-d	0.02	0.09	0.04	0.04	0.04
sepsis-s	0.11	0.14	0.22	0.08	0.20
heart-failure-d	0.18	0.04	0.03	0.08	0.03
heart-failure-c	0.28	0.09	0.04	0.18	0.04
synth-large	0.05	0.05	0.09	0.05	0.09
synth-small	0.48	0.07	0.09	0.42	0.09
cleveland	0.17	0.21	0.32	0.17	0.32
breast	0.02	0.04	0.27	0.03	0.27
hepatitis	0.08	0.13	0.13	0.11	0.12
heart	0.17	0.24	0.44	0.18	0.42
diabetes	0.10	0.17	0.54	0.13	0.52
Mean	0.14	0.13 (p = 0.542)	0.20 (p = 0.216)	0.13 (p = 1.00)	0.20 (p = 0.244)

BO-PDP-Bay did not have statistically significantly different mean ECEs than PDP-Bay ($p = 0.542$), AB-Bay ($p = 0.216$), BO-PDP-Ent ($p = 1.00$), or AB-Ent ($p = 0.244$) at the 0.05 level on the Wilcoxon signed-rank test.

5.3.3 Model Complexity

The mean MPL for BO-PDP-Bay was 3.65. Mean MPLs for comparison methods were 4.38 for PDP-Bay, 0.89 for AB-Bay, 3.57 for BO-PDP-Ent, and 1.00 for AB-Ent. MPLs for each method and dataset are listed in Table 19.

Table 19. MPLs for BO-PDP-Bay, PDP-Bay, AB-Bay, BO-PDP-Ent, and AB-Ent methods. Results in red font indicate statistically significantly shorter mean MPLs than that of BO-PDP-Bay on the Wilcoxon signed-rank test.

Dataset	BO-PDP-Bay	PDP-Bay	AB-Bay	BO-PDP-Ent	AB-Ent
chronic-pancreatitis	2.89	2.76	1.00	2.56	1.00
pneumonia	2.79	2.97	1.00	2.50	1.00
sepsis-d	3.32	3.53	0.90	3.26	1.00
sepsis-s	4.33	4.27	0.40	4.10	1.00
heart-failure-d	6.69	6.37	1.00	6.45	1.00
heart-failure-c	7.82	8.20	1.00	7.35	1.00
synth-large	2.64	2.41	1.00	2.38	1.00
synth-small	5.37	13.53	1.00	5.96	1.00
cleveland	3.07	3.43	0.82	3.18	1.00
breast	1.52	1.53	1.00	1.49	1.00
hepatitis	1.80	1.94	1.00	1.65	1.00
heart	2.82	3.39	0.73	2.96	1.00
diabetes	2.34	2.65	0.79	2.59	1.00
Mean	3.65	4.38 ($p = 0.094$)	0.89 ($p < 0.001$)	3.57 ($p = 0.244$)	1.00 ($p < 0.001$)

BO-PDP-Bay did not have statistically significantly different mean MPLs than PDP-Bay ($p = 0.094$) and BO-PDP-Ent ($p = 0.244$); it had statistically significantly longer mean MPLs than AB-Bay ($p < 0.001$) and AB-Ent ($p < 0.001$) at the 0.05 level on the Wilcoxon signed-rank test. In

summary, in terms of MPL, BO-PDP-Bay produced more complex models than the population ensemble methods and produced similarly complex models as the other personalized methods.

5.4 Personalized Methods

We aggregated results of personalized and population methods (see Table 20) and compared the two groups using a paired Wilcoxon signed-rank test.

Table 20. Personalized methods and population methods.

Personalized Methods	Population Methods
PDP-Bay (13 datasets)	DT-Bay (13 datasets)
PDP-Bay-BDeu (13 datasets)	DT-Bay-BDeu (13 datasets)
PDP-Ent (13 datasets)	DT-Ent (13 datasets)
LazyRF-Bay (13 datasets)	RF-Bay (13 datasets)
LazyRF-Ent (13 datasets)	RF-Ent (13 datasets)
Boosted PDP-Bay (13 datasets)	AB-Bay (13 datasets)
Boosted PDP-Ent (13 datasets)	AB-Ent (13 datasets)

The mean AUROC across all personalized methods was 0.79 and the mean AUROC across all population methods was 0.78. Personalized and population methods did not have statistically significantly different AUROCs ($p = 0.058$) at the 0.05 level. The mean ECE across all personalized methods was 0.12 and the mean ECE across all population methods was 0.11. Population methods had statistically significantly lower ECEs than personalized methods ($p = 0.039$) at the 0.05 level. The mean MPL across all personalized methods was 3.86 and the mean MPL across all population methods was 3.50. Personalized and population methods did not have

statistically significantly different MPLs (p = 0.710) at the 0.05 level. These results are listed in Table 21.

Table 21. Aggregate results of personalized and population methods.

Results in red font indicate statistically significantly lower ECE than personalized methods on Wilcoxon signed-rank test.

	Personalized Methods	Population Methods
Mean AUROC	0.79	0.78 (p = 0.058)
Mean ECE	0.12	0.11 (p = 0.039)
Mean MPL	3.86	3.50 (p = 0.710)

In summary, in terms of calibration, personalized methods had slightly worse performance than population methods, and in terms of discrimination and model complexity, personalized methods had performance on par with population methods.

5.5 Bayesian Methods

We aggregated results of Bayesian and non-Bayesian scored methods (see Table 22) and compared them using a paired Wilcoxon signed-rank test.

Table 22. Bayesian and non-Bayesian methods.

Bayesian Methods	Non-Bayesian Methods
PDP-Bay (13 datasets)	PDP-Ent (13 datasets)
LazyRF-Bay (13 datasets)	LazyRF-Ent (13 datasets)
BO-PDP-Bay (13 datasets)	BO-PDP-Ent (13 datasets)
DT-Bay (13 datasets)	DT-Ent (13 datasets)
RF-Bay (13 datasets)	RF-Ent (13 datasets)
AB-Bay (13 datasets)	AB-Ent (13 datasets)

The mean AUROC across all Bayesian methods was 0.81 and the mean AUROC across all population methods was 0.76. Bayesian methods had statistically significantly higher AUROCs than non-Bayesian methods ($p < 0.001$) at the 0.05 level. The mean ECE across all Bayesian methods was 0.11 and the mean ECE across all population methods was 0.13. Bayesian methods had a statistically significantly lower ECEs than non-Bayesian methods ($p < 0.001$) at the 0.05 level. The mean MPL across all Bayesian methods was 3.52 and the mean AUROC across all population methods was 4.05. Bayesian methods had a statistically significantly shorter MPLs than non-Bayesian methods ($p = 0.035$) at the 0.05 level. These results are listed in Table 23.

Table 23. Aggregate results of Bayesian and non-Bayesian methods.

Results in blue font indicate statistically significantly worse value than Bayesian methods on Wilcoxon signed-rank test (lower for AUROC, higher for ECE and MPL).

	Bayesian Methods	**Non-Bayesian Methods**
Mean AUROC	0.81	0.76 ($p < 0.001$)
Mean ECE	0.11	0.13 ($p < 0.001$)
Mean MPL	3.52	4.05 ($p = 0.035$)

In summary, in terms of discrimination, calibration, and model complexity, Bayesian methods had better performance than non-Bayesian methods.

We performed pairwise comparison between Bayesian and non-Bayesian personalized methods for each approach (single personalized decision path, lazy random forest, and boosted ensemble of personalized decision paths).

Table 24. Paired results of Bayesian and non-Bayesian personalized methods. Results in red font indicate statistically significantly better value than corresponding Bayesian method, and results in blue font indicate statistically significantly worse value than corresponding Bayesian method on Wilcoxon signed-rank test (lower for AUROC, higher for ECE and MPL).

	PDP-Bay	**PDP-Ent**	**LazyRF-Bay**	**LazyRF-Ent**	**BO-PDP-Bay**	**BO-PDP-Ent**
Mean AUROC	0.78	0.70 (p = 0.002)	0.82	0.80 (p = 0.305)	0.82	0.80 (p = 0.147)
Mean ECE	0.13	0.14 (p = 0.002)	0.08	0.08 (p = 0.340)	0.14	0.13 (p = 1.00)
Mean MPL	4.38	4.00 (p = 0.005)	3.95	3.94 (p = 0.839)	3.65	3.57 (p = 0.244)

In summary, in terms of discrimination and calibration, the Bayesian personalized decision path had superior performance to the non-Bayesian personalized decision path. In terms of model complexity, the Bayesian personalized decision path had more complex models than the non-Bayesian personalized decision path. In terms of discrimination, calibration, and model complexity, there was no difference between the Bayesian and non-Bayesian ensembles of personalized decision paths. These results are listed in Table 24.

We also performed pairwise comparison between Bayesian and non-Bayesian population methods for each approach (single decision tree, random forest, and boosted ensemble of decision trees), the results of which are listed in Table 25.

Table 25. Paired results of Bayesian and non-Bayesian population methods.

Results in blue font indicate statistically significantly worse value than corresponding Bayesian method on Wilcoxon signed-rank test (lower for AUROC, higher for ECE and MPL).

	DT-Bay	DT-Ent	RF-Bay	RF-Ent	AB-Bay	AB-Ent
Mean AUROC	0.80	0.71 (p = 0.003)	0.81	0.77 (p < 0.001)	0.80	0.81 (p = 0.307)
Mean ECE	0.05	0.15 (p < 0.001)	0.04	0.05 (p = 0.040)	0.20	0.20 (p = 0.162)
Mean MPL	4.35	6.02 (p = 0.027)	3.87	5.76 (p = 0.001)	0.89	1.00 (p = 0.059)

In summary, in terms of discrimination, calibration, and model complexity, the Bayesian decision tree had statistically significantly better performance than the non-Bayesian decision tree, the Bayesian random forest had statistically significantly better performance than the non-Bayesian random forest, and the Bayesian boosted ensemble had performance on par with the non-Bayesian boosted ensemble.

For our final comparison, we performed comparison of the aggregated results of the three novel personalized Bayesian algorithmic methods (PDP-Bay, LazyRF-Bay, and BO-PDP-Bay) with the population non-Bayesian algorithmic methods. These results are listed in Table 26

Table 26. Aggregate results of Bayesian personalized and non-Bayesian population methods.

	Bayesian Personalized Methods	Non-Bayesian Population Methods
Mean AUROC	0.81	0.76 (p = 0.064)
Mean ECE	0.12	0.13 (p = 0.403)
Mean MPL	3.99	4.26 (p = 0.380)

In summary, in terms of discrimination, calibration, and model complexity, the Bayesian personalized methods were on par with the non-Bayesian population methods.

6.0 Discussion

This chapter discusses the results and insights gained from this dissertation, describes potential shortcomings, and explores directions for future work.

6.1 Summary of Results and Insights

This section reviews the main results across the algorithmic methods and discusses the findings in the context of the hypotheses. An overview of the best performing and worst performing algorithmic methods for a given experiment on a specific performance metric is given in Table 27.

Table 27. Summary of results.

Methods in red font had best performance of group for given metric, methods in blue font had worst performance for given metric, and methods in black font had performance in between.

Experimental Method vs Control Methods		*Support for corresponding hypothesis?*
PDP-Bay vs DT-Bay, PDP-Ent, DT-Ent		
Discrimination	PDP-Bay, DT-Bay, PDP-Ent, DT-Ent	1: Partial
Calibration	PDP-Bay, DT-Bay, PDP-Ent, DT-Ent	1: Partial
Model Complexity	PDP-Bay, DT-Bay, PDP-Ent, DT-Ent	1: Partial
LazyRF-Bay vs PDP-Bay, RF-Bay, LazyRF-Ent, RF-Ent		
Discrimination	LazyRF-Bay, PDP-Bay, RF-Bay, LazyRF-Ent, RF-Ent	2: Partial
Calibration	LazyRF-Bay, PDP-Bay, RF-Bay, LazyRF-Ent, RF-Ent	2: Partial
Model Complexity	LazyRF-Bay, PDP-Bay, RF-Bay, LazyRF-Ent, RF-Ent	2: Partial
BO-PDP-Bay vs PDP-Bay, AB-Bay, BO-PDP -Ent, AB-Ent		
Discrimination	BO-PDP-Bay, PDP-Bay, AB-Bay, BO-PDP-Ent, AB-Ent	3: None
Calibration	BO-PDP-Bay, PDP-Bay, AB-Bay, BO-PDP-Ent, AB-Ent	3: None
Model Complexity	BO-PDP-Bay, PDP-Bay, AB-Bay, BO-PDP-Ent, AB-Ent	3: None

6.1.1 PDP-Bay

Our hypothesis is that the Bayesian-scored personalized decision path demonstrates superior predictive performance to population decision trees (with both Bayesian and non-Bayesian scores) and the entropy-scored personalized decision path. We compared results across 13 datasets in terms of discrimination, calibration, and model complexity.

We found that in terms of discrimination, PDP-Bay is better than both population and personalized entropy-scored methods and equivalent to the Bayesian population method. In terms

of calibration, PDP-Bay is better than the entropy-scored personalized method, comparable to the entropy-scored population method, and inferior to the Bayesian population method. In terms of model complexity, PDP-Bay is better than the entropy-scored population method, comparable to the Bayesian-scored population method, and inferior to the entropy-scored personalized method.

Thus, the PDP-Bay has equivalent or better performance on all three measures when compared to the entropy-scored population method, mixed performance when compared to the entropy-scored personalized method, and equivalent or worse performance when compared to the Bayesian population method. No single method dominated across all three measures of performance.

As the PDP-Bay has equivalent or inferior performance to the Bayesian population method (DT-Bay), these results indicate that our approach to personalization may not greatly improve predictive performance. The method of personalization used in this dissertation has previously been observed to be prone to overfitting, although we hypothesized that the used of Bayesian scoring would help mitigate this. In comparing the calibration of the PDP-Bay and DT-Bay, we see that the Bayesian population method has significantly better calibration than the personalized method. This may indicate that our method of personalization is still overfitting, resulting in lower calibration than the population method. At the same time, the Bayesian personalized method had better calibration than the non-Bayesian personalized method, indicating that Bayesian scoring does mitigate some degree of overfitting resulting from the chosen method of personalization. One noteworthy strength of the PDP-Bay method is that it has better performance than the entropy-scored population method (DT-Ent), which is our implementation of an approach that is commonly used to train classification trees.

The PDP-Bay demonstrates worse performance in terms of calibration and model complexity and comparable performance in terms of discrimination when compared with the PDP-Bay-BDeu. We also compared the DT-Bay with DT-Bay-BDeu, and the DT-Bay demonstrates comparable performance with DT-Bay-BDeu in terms of discrimination and calibration and inferior performance in terms of model complexity. The PDP-Bay and DT-Bay use uniform parameter and structure priors, whereas PDP-Bay-BDeu and DT-Bay-BDeu perform a grid search to select optimized prior hyperparameters (which may or may not be uniform). These results demonstrate that some performance improvements can be derived from hyperparameter tuning with Bayesian methods.

6.1.2 LazyRF-Bay

Our hypothesis is that LazyRF-Bay, the Bayesian-scored random forest of personalized decision paths, demonstrates superior predictive performance over a single Bayesian personalized decision path, population random forests with decision trees scored using both Bayesian and non-Bayesian methods, and the entropy-scored random forest of personalized decision paths. We compared results across 13 datasets in terms of discrimination, calibration, and model complexity.

We found that in terms of discrimination, LazyRF-Bay is superior to the single Bayesian-scored personalized decision path and on par with the entropy-scored random forest of personalized decision paths and population random forests. In terms of calibration, LazyRF-Bay is superior to the single Bayesian-scored personalized decision path, on par with the entropy-scored random forest of personalized decision paths, and inferior to the population random forests. In terms of model complexity, LazyRF-Bay's performance is on par with the single Bayesian-

scored personalized decision path, the entropy-scored random forest of personalized decision paths, and the Bayesian population random forest and superior to the entropy-scored population random forest.

Thus, the LazyRF-Bay has equivalent performance on all three measures to the entropy-scored random forest of personalized decision paths. LazyRF-Bay has equivalent or better performance on all three measures when compared to the single Bayesian personalized decision path. LazyRF-Bay has mixed performance when compared to the entropy-scored population random forest method, and equivalent or worse performance when compared to the Bayesian population random forest method. No single method dominated across all three measures of performance.

As the LazyRF-Bay has equivalent or inferior performance to the Bayesian population method (RF-Bay), these results (like the PDP-Bay results) indicate that our approach to personalization may not greatly improve predictive performance. The LazyRF-Bay method has better performance on the whole than the single Bayesian personalized decision path, indicating that the use of ensemble methods and feature randomization may improve predictive performance in personalized decision paths as is seen with population decision trees. This is likely due to the reduction in variance while maintaining bias across models that results from training and averaging multiple decorrelated models.[80] However, the bias and variance of the LazyRF-Bay was not explicitly evaluated in these experiments.

6.1.3 BO-PDP-Bay

Our hypothesis is that BO-PDP-Bay, the ensemble of boosted Bayesian-scored personalized decision paths, demonstrates superior predictive performance over a single Bayesian-scored personalized decision path, population ensembles of boosted Bayesian-scored and entropy-scored decision trees, and the ensemble of boosted entropy-scored personalized decision paths. We compared results across 13 datasets in terms of discrimination, calibration, and model complexity.

We found that in terms of discrimination and calibration, BO-PDP-Bay performs on par with all comparison methods. In terms of model complexity, BO-PDP-Bay is on par with the single Bayesian-scored personalized decision path and the ensemble of boosted entropy-scored personalized decision paths and is inferior to the population ensembles of boosted decision trees.

It is important to note, however, that AdaBoost, the method for training population ensembles of boosted decision trees, specifies that each tree in the ensemble is limited to a single variable. This makes the maximum possible mean path length for AB-DT and AB-Ent equal to one. The method for training ensembles of boosted personalized decision paths developed by Fern et al.,[20] which was used for the BO-PDP-Bay and BO-PDP-Ent, does not limit the length of the decision path. Thus, difference in mean path lengths between the personalized and population ensembles of boosted models cannot be solely attributed to personalization.

In the original results of Fern et al., which compared ensembles of boosted entropy-scored personalized decision paths to single entropy-scored personalized decision paths as well as population ensembles of boosted entropy-scored decision trees, they found that ensembles of boosted personalized decision paths have better performance in terms of accuracy than single

decision paths and have comparable performance to population ensembles of boosted decision trees. The BO-PDP-Ent is our implementation of the BO-LazyDT in the Fern paper, and to replicate their results, we compared the mean AUROCs of BO-PDP-Ent with PDP-Ent and AB-Ent on Wilcoxon signed-rank test (detailed in Appendix Table 1). Like Fern et al., we found that the ensembles of boosted entropy-scored personalized decision paths have better performance in terms of discrimination than single entropy-scored decision paths and have comparable performance to population ensembles of boosted entropy-scored decision trees.

Fern et al. also found that ensembles of boosted personalized decision paths have shorter mean path lengths than population ensembles of boosted decision trees. Their results indicate that they used a different implementation of AdaBoost which did not restrict the number of variables allowed in the decision trees, as the mean path length of the models produced by their approach was greater than one. We therefore cannot compare our results on model complexity for the boosted methods with their findings.

6.1.4 Personalized Methods

When we aggregated the results of all personalized methods and compared these to the aggregated results of all population methods, we found that there was no statistically significant difference between personalized methods and population methods in terms of discrimination and model complexity (although, as noted in the previous section, the restriction of tree size for the population ensembles of boosted decision trees may have affected these results). The personalized methods had statistically significantly worse calibration than the population methods, which may be due to possible overfitting by this type of personalization. These findings indicate that our

approach to personalization alone does not guarantee improved predictive performance in terms of discrimination, calibration, and model complexity. It is important to note, however, that the approach to personalization tested in this dissertation is only one of many possible approaches, and that other methods of personalization could produce very different findings. These conclusions therefore do not apply to all types of patient-specific modelling.

6.1.5 Bayesian Methods

When we aggregated the results of all Bayesian methods and compared these to the aggregated results of all non-Bayesian methods, we found that the Bayesian methods had statistically significantly superior performance in terms of discrimination, calibration, and model complexity. The results indicate that, on average, the use of Bayesian scoring results in better predictive performance for tree models.

One common issue with tree methods (including some personalized decision paths) is overfitting. This occurs when models incorporate non-informative features that correlate with specific target outcomes in the training data, but do not represent real relationships in the overall population. These observations of false relationships become more likely when sample sizes are small. The stopping criteria for tree methods often aim to achieve purity, which is when all training cases corresponding to a given partition share the same target value. This can result in partitions that produce very small and skewed subgroups, including subgroups consisting of a single training case. The subgroups defined by the tree are used for parameterization and prediction. Thus, predictions from these small, pure subgroups are often poorly calibrated, incorporate many non-informative features, and are based on a very small sample size. Furthermore, many tree methods

estimate parameters using maximum likelihood. When a subgroup corresponds to a single training case, the predicted probability of the target outcome will be 100% for the value of the target for that training case when the subgroup is used for inference.

The use of Bayesian scoring prevents overfitting in several ways, which may account for the observed improvements in predictive performance. The Bayesian scoring methods incorporate a parameter prior probability distribution that is not present in the entropy-based scoring methods, as they use an entropy score that is equivalent to the sample-normalized log likelihood. The use of the prior exerts regularization effects on model training. Additionally, the parameter estimates of Bayesian methods involve Laplace smoothing, so that even when a pure subgroup is used for inference, it is not possible to predict with 0% or 100% probability. This helps account for uncertainty in calculating predictions using limited data and may limit overfitting.

However, the performance improvements associated with Bayesian methods are not necessarily uniform. We observed that for single population decision trees and population random forests, Bayesian scoring was associated with better performance in terms of discrimination, calibration, and model complexity, but for ensembles of boosted decision trees, there was no difference in terms of discrimination, calibration, or model complexity. We observed that for single personalized decision paths, Bayesian scoring was associated with better discrimination and calibration, but inferior model complexity. For personalized random forests and ensembles of boosted personalized decision paths, there was no difference in performance between Bayesian and non-Bayesian methods. Prior empiric work has shown that information theoretic scores may perform better than Bayesian scores for small datasets.[99] Though our results support the use of Bayesian scoring for tree methods, further work is needed to characterize the relationships between datasets and optimal machine learning approaches.

Additionally, performance gains from the use of ensemble approaches may be limited when combined with Bayesian methods. Of note, we found that ensembles of boosted Bayesian personalized decision paths did not have better performance than single Bayesian personalized decision paths. This may be a result of the higher baseline performance of the single Bayesian personalized decision paths when compared to the single entropy-scored personalized decision path. The mean AUROCs of the PDP-Ent and BO-PDP-Ent were 0.70 and 0.80, respectively; meanwhile, the mean AUROCs of the PDP-Bay and BO-PDP-Bay were 0.78 and 0.82, respectively.

Additionally, the method used for boosting may not work well with Bayesian approaches. The personalized boosting method aims to improve predictive performance by altering training case weights used to train the personalized decision paths. Changes to the training weights are dependent on the classifications made by previous iterations of decision paths in the ensemble. Boosting methods like AdaBoost usually use weak learners (such as shallow decision trees).[81] It is therefore possible that the PDP-Bay is not sufficiently weak to benefit from boosting. Tree methods are notoriously sensitive to changes in data, which is why boosting methods result in a diverse set of models via perturbations of the training data weights. However, Bayesian scoring methods are potentially less sensitive to these small changes in the training data due to previously discussed features like Laplace smoothing in parameter estimation and a parameter prior. These elements of the *BayScore*, which may be responsible for the higher mean AUROC of the PDP-Bay compared to the PDP-Ent, may also be the reason why boosting does not appear to improve predictive performance for Bayesian personalized decision paths. If the Bayesian scoring metric used by PDP-Bay makes it less sensitive to perturbations of the data, the changes to the distribution of the data from altering training case weights might not affect the predictor scores.

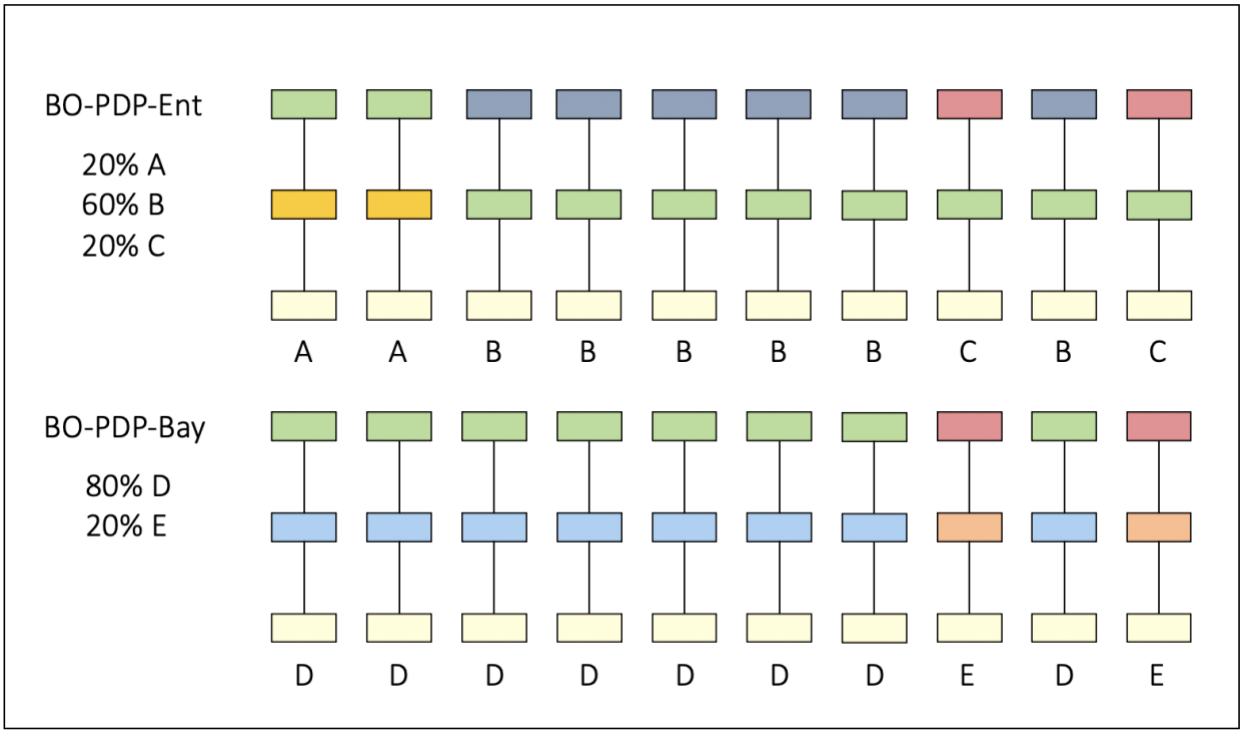


Figure 7. Two ensembles of boosted personalized decision paths.

This figure shows two ensembles of decision paths (the top one produced by the BO-PDP-Ent method and the bottom one produced by the BO-PDP-Bay method) for the same test case from the chronic pancreatitis dataset. Each color represents a specific feature, and each distinct combination of features is labeled with a letter (A-E). In this example, the BO-PDP-Ent ensemble is more heterogenous than the BO-PDP-Bay ensemble.

This could result in the same predictors being selected by the BO-PDP-Bay algorithm repeatedly, resulting in less diverse ensembles of models than when non-Bayesian scoring metrics are used. Figure 7 shows two ensembles of boosted decision paths for a single test case, one produced by the BO-PDP-Ent method and the other by the BO-PDP-Bay. The Bayesian-scored ensemble is less diverse than the non-Bayesian scored ensemble in this example. When ensembles consist of the same model repeatedly, the average prediction will not be very different from the prediction of a single model, and no effect on predictive performance will be observed. This is

consistent with our results from the BO-PDP-Bay experiments. Rather than see synergistic benefits of Bayesian and ensemble approaches in combination, there may be different benefits to each method that may not be additive when used together.

6.2 Limitations

This section discusses some of the limitations of the algorithmic methods and findings of this dissertation.

6.2.1 Generalizability

The generalizability of the results may be limited by several factors. The clinical datasets used for evaluation were all collected for research purposes, were comprised of discretized data, and had binary targets. For most of the datasets, experts selected the features for inclusion in the dataset, and it is not clear how well the datasets used in this dissertation represent all possible clinical datasets. Therefore, our results may not generalize to other clinical datasets, such as those with more heterogenous or noisy data like electronic health record data. Additionally, we used only a moderate number of datasets. Further development and evaluation would be needed before these new algorithms could be used with a wider range of data sources.

We only evaluated one method of personalization in this dissertation, and our findings may not generalize to other personalized decision path methods or other approaches to producing personalized models. Other methods for producing personalized decision paths like DP-Bay

incorporate more data from the training dataset, whereas our method only uses training data cases which share specific features with the test case. Comparing our methods to a wider range of personalized decision path algorithms and creating ensembles using different types of personalized decision paths could provide more insight into the potential differences in predictive performance associated with various approaches to personalization.

We aimed to assess the effects of personalization, Bayesian scoring, and ensemble approaches on predictive performance of algorithms, and in focusing on those effects, we did not include other strategies that are commonly used with tree algorithms. For example, we did not implement pruning for the population decision trees (DT-Bay and DT-Ent) because our base personalized decision path methods did not perform pruning. However, pruning is often employed with tree algorithms to improve model parsimony and avoid overfitting.[80] There are methods for pruning personalized decision paths,[20] and the effects of pruning on personalized decision paths could be an additional area of inquiry for improving the performance of our methods.

We did not perform hyperparameter tuning for ensemble size or for the boosting hyperparameters, and instead chose values that were used by other studies to allow for replication of other published work, comparison across methods, and efficiency. Early results from initial experiments did not show a significant difference in AUROCs for ensembles trained using 10 versus 25 models, and we did not evaluate performance on larger ensembles. We do not know if our findings change with larger ensemble sizes. Hyperparameter tuning can increase the time needed to train a model (especially in the case of ensembles of models). For population methods, this may not be significant, as the model is usually trained in a distinct process from performing inference on patients of interest. In the case of personalized methods, the model is trained at the time of the encounter with the patient of interest. Thus, the costs of using time-consuming

approaches like hyperparameter tuning or larger ensemble sizes that greatly increase model training time must be weighed against the possible corresponding gains in predictive performance. However, time complexity was not explicitly evaluated in this dissertation. Further work comparing different hyperparameter settings, ensemble sizes, and pruning applied to both personalized methods and population methods might be necessary to increase the generalizability of results by incorporating these commonly used strategies for learning tree models.

6.2.2 Mean Path Length

One of the evaluation metrics used in this dissertation was mean path length (MPL), which measured the average number of predictors in a decision path used for prediction. In the case of ensembles, this was calculated as the mean across all models in the ensemble, and then averaged across all test cases. For single model methods, it was calculated as the mean across all test cases. While most methods did not limit the number of predictors that could be included in a model, the AdaBoost methods (AB-Ent and AB-Bay) limited the number of predictors in each model in the ensemble to one. This metric was used as a measure of model complexity with the assumption that less complex models are easier for users to comprehend. It is not clear, however, whether a longer single decision path is less comprehensible than a collection of many shorter decision paths in the form of an ensemble. We could alternately have measured the average total number of predictors across all paths in each ensemble, but since the same path might appear multiple times in an ensemble, this may not be a useful measure of complexity either. Thus, MPL may not adequately capture model comprehensibility or consistently assess whether a given method produced simpler or more comprehensible models.

6.2.3 Clinical Significance

It is difficult to determine the significance of our findings without knowledge of the clinical decision being supported by a given algorithmic method. While statistically significant improvements in performance were observed for several methods, the clinical significance of a given performance improvement is difficult to evaluate without knowledge of the context of the clinical decision that these methods would be employed to support. For example, AUROC characterizes the probability that a randomly selected case with a positive target value will have a higher prediction for the positive class than a randomly selected case with a negative target value. This measures discrimination, or the ability to differentiate between positive and negative cases on average. A classifier with a high AUROC can help avoid excessive false positive or false negative predictions. Without knowing the cost of these false predictions, however, it is difficult to assess whether a small performance gain in terms of discrimination is clinically meaningful. A difference of 0.04 between two AUROCs may or may not correspond to a clinically meaningful difference in a clinical decision support system, which can also depend on the patient population, clinical decision, utilities of outcomes, clinical workflow, and users.

Additionally, the selected performance metrics may not measure the most important aspects of predictive performance for a given clinical problem. AUROC does not incorporate any assessment of the calibration of the predicted probabilities themselves; it only represents the relationship between predictions for positive and negative cases on average. However, depending on the use case, another metric (such as the calibration of a model) may be more important than discrimination. For example, if an algorithmic method is used to calculate patient-specific probabilities of outcomes for use in combination with utilities as part of formal decision analysis,

low ECE may be more important than high AUROC. No single algorithmic method was decidedly superior to every other method in terms of all three performance measures, and future use in clinical decision support may require application of other metrics not used in this dissertation for evaluating predictive performance depending on the goals and needs of the clinical problem.

6.2.4 Patient-Specific Performance and Algorithmic Fairness

All of the performance measures used in this dissertation distill average performance into a single number, which allows for succinct and clear statistical comparison between methods. However, these measures do not tell us how the algorithms differ on a case-by-case basis. As mentioned previously, the goal of precision medicine is to provide the right care to the right patient at the right time. Our analysis of these algorithmic methods does not tell us whether specific individuals receive better predictions from the use of personalized algorithmic methods over population algorithmic methods. We do not know whether observed performance improvements stem from performance gains for only certain subgroups of patients – gains that may outweigh and mask possible performance losses for other subgroups in the test data.

One of the reasons machine learning is applied to decision support is the perception that it is objective. Unfortunately, models produced via machine learning are susceptible to a range of biases caused by both the data that is used to create them and the methods used for model construction.[100] Research has shown that an algorithm used in the criminal justice system to make sentencing recommendations discriminated against people of color, and an algorithm used for hiring at Amazon was abandoned after it was found to discriminate against women.[101] These examples demonstrate how machine learning methods can learn patterns that reflect biased

practices. Other work has shown that automated facial analysis algorithms trained on datasets comprised of predominantly lighter-skinned faces demonstrate significantly higher misclassification rates for darker-skinned female faces. This example demonstrates how, in the process of fitting a model, many machine learning algorithms disregard rare patterns present only in minority subgroups of the training data, leading to disparities in predictive performance in such groups.[102]

Machine learning in the medical domain is not immune: Obermeyer et al.[103] found that an algorithm used to allocate healthcare resources disadvantaged Black patients. Machine learning algorithms faithfully learn and reproduce patterns in data, including patterns that are caused by and reflect social inequities.[104] The use of such machine learning algorithms can then contribute to a vicious cycle that perpetuates those inequities. Even when the training data does not reflect discriminatory practices, underrepresentation of minority populations can also lead algorithms to learn unfair models. Most machine learning algorithms fit models that represent patterns present in the majority of the samples in the data, and this can lead to minority subgroups being modeled poorly.[11]

Concerns regarding such findings have led to a rise in research into fair machine learning with the goal of defining and measuring fairness in the context of machine learning as well as developing fairness-aware algorithms.[105] This field aims to identify unfair models and develop algorithms that produce models that do not discriminate against or disadvantage subgroups, especially legally protected classes.[106]

Fairness in medical machine learning is especially important because underrepresentation of minority groups in research data is a longstanding and rampant problem in biomedical research and trials used to evaluate treatments and establish care guidelines.[107–109] Although the National

Institutes of Health mandated representative inclusion of women and members of racial minority groups in clinical research in 1993,[110] numerous studies demonstrate that women, racial minorities, elderly patients, rural patients, and patients of low socioeconomic status remain underrepresented in research studies, including investigations into cancer treatment,[111] AIDS/HIV treatment,[112] migraine treatment,[113] and cardiovascular health guidelines.[108] Use of these imbalanced research datasets for training algorithms for clinical decision support may result in unfair models that do not generalize to the entire population, resulting in disparate impact and possibly poorer outcomes for members of these underrepresented subgroups.[114]

This type of disparate impact may be mitigated by algorithms that can better fit models to subgroups in the population. Chen et al.[115] hypothesized that clustering methods may mitigate this type of algorithmic unfairness. It is possible that personalized methods may result in fairer models, as the predictive model is tailored to the individual patient rather than the majority or population on average. A number of papers have been published in recent years discussing the importance of considering fairness in the context of clinical machine learning.[12,116–120] Many measurements have also been proposed to aid in the evaluation of algorithmic fairness (some of which are, unfortunately, mathematically incompatible).[121] However, only a few studies have actually evaluated the fairness of clinical machine learning algorithms.[103,115,122–124] This dissertation did not examine the demographics of training datasets or evaluate the predictive performance on subgroups of the training data to evaluate fairness, but this would be a crucial area of investigation prior to using such methods for clinical decision support.

6.3 Future Work

An open question following this dissertation is why Bayesian methods and ensemble methods are not consistently associated with improved predictive performance when used in combination. Bayesian scoring and ensemble approaches separately have each been associated with improved predictive performance for tree models. Ensemble methods have been found to reduce variance through averaging the outputs of multiple decorrelated models without increasing bias, which results in an overall reduction of error. Performing bias-variance analysis on the algorithmic methods presented in this dissertation could provide insight into how these approaches work.

Since Bayesian methods were shown to often be associated with improved predictive performance, further exploration into hyperparameter tuning for the PDP-Bay-BDeu could be beneficial. Hyperparameter tuning was performed using simple grid search to identify priors that resulted in models that maximized AUROC. More sophisticated tuning procedures could also improve performance, and different metrics could be used to evaluate the performance of models using different priors, such as the BayScore of the models themselves (which is proportional to the expected posterior probability of model given the data).

Although we conjectured that personalized methods could increase predictive performance over population methods by selecting informative combinations of features that do not appear in population models, we did not explicitly examine which features from each dataset were used for prediction by personalized and population methods. It could therefore be useful to identify differences in the specific features used for prediction by personalized and population tree models and evaluate whether personalized methods are able to identify a wider range of meaningful

features than population methods. For ensemble methods, feature importance rankings for each prediction can be calculated, either by ordering features by frequency of occurrence in the ensemble or by producing post-hoc explanations using a method like LIME.[64] Once significant features are identified, they could also be evaluated by clinicians to characterize the explainability of different methods. We assume that shorter mean path lengths indicate more comprehensible models, but it is not clear how to compare the interpretability of the output of a single complex decision tree with a collection of 10 parsimonious personalized decision paths. It is likely that summary methods for identifying significant features could be necessary for ensembles of simple models, and limited work has been done comparing clinicians' evaluations of interpretability of different types of models and explanation methods.

We also conjectured that personalized methods could reduce performance disparities amongst subgroups of the population, but we did not calculate performance measures on subgroups of the test datasets. Automated methods for identifying subgroups with performance disparities and comparing rates of disparities between algorithmic methods have been developed and could be useful in evaluating the fairness of personalized algorithms.[11]

We only evaluated one type of personalization in this dissertation. It is important to determine how our PDP-Bay method compares to other types of Bayesian personalized decision path algorithms like DP-Bay.[15] Additionally, using alternative methods of personalization with the boosting and random forest ensemble approaches may produce different results, and so further exploration into personalized ensembles using methods like DP-Bay is necessary.

Evaluating the performance of these algorithmic methods a wider variety of datasets would also be beneficial. Prior to using these methods for any type of clinical decision support, the algorithms would need to be evaluated on less curated datasets (such as electronic health record

data). This would likely require the development of a machine learning pipeline, which could process the data for use with the algorithms, as well as adjustments to the algorithms to handle a wider variety of data types. Incorporating methods for handling both continuous and discrete variables would allow for application of these algorithms to a wider variety of datasets.

7.0 Conclusions

This dissertation presents several new personalized machine learning algorithms that use Bayesian scores, including the personalized decision path with Bayesian score (PDP-Bay), the personalized decision path with BDeu score (PDP-Bay-BDeu), the lazy random forest with Bayesian score (LazyRF-Bay), and the boosted personalized decision path with Bayesian score (BO-PDP-Bay). This dissertation also presents the first investigation into random forest ensembles of personalized decision paths as well as the first evaluation of ensembles of personalized decision paths on biomedical data for predicting patient-specific outcomes.

Our main findings are as follows:

- Our method of personalization is not associated with performance improvements when applied to single Bayesian trees, random forest ensembles of Bayesian decision trees, or ensembles of boosted Bayesian decision trees.
- The use of Bayesian scoring methods is associated with performance improvements over corresponding non-Bayesian tree methods in terms of discrimination, calibration, and model complexity when applied to single personalized decision paths, single decision trees, and random forest ensembles of decision trees. This association does not hold for random forest ensembles of personalized decision paths, ensembles of boosted personalized decision paths, and ensembles of boosted decision trees.
- The use of a random forest ensemble approach is associated with performance improvements over corresponding single, non-randomized tree models in terms of

discrimination and calibration when applied to Bayesian and non-Bayesian personalized decision paths and entropy-scored decision trees. This association does not hold for Bayesian decision trees.

- The use of a boosting ensemble approach is associated with performance improvements over corresponding single tree models in terms of discrimination when applied to non-Bayesian personalized decision paths and decision trees. This associated does not hold for Bayesian personalized decision paths and decision trees.

The findings of this dissertation indicate that while ensemble approaches and Bayesian scoring can separately improve predictive performance of tree models, the benefits are not necessarily additive when combined together. Furthermore, we found that our method of personalization did not improve predictive performance over population Bayesian methods.

Further work is needed to deeply understand the mechanisms of each approach and the effects of their use in combination. However, the novel combinations of personalization, Bayesian scoring, and ensemble approaches for tree algorithms presented in this dissertation demonstrate equivalent or superior performance on clinical datasets to commonly used entropy-scored algorithms like decision trees and random forests, and could be considered when choosing algorithmic methods for clinical decision support.

Appendix A – Pseudocode of Comparison Methods

The pseudocode for the PDP-Ent and LazyRF-Ent methods is provided in this section.

PDP-Ent (*D*, ***Test***)
Inputs:
D is a training dataset consisting of ***m*** training cases and ***n*** variables and a target variable *T*
Test is the test case (patient of interest) for which a prediction will be made

```
1   S ← empty path
2   D_S ← D
3     LOOP:
4       BestScore ← 0; BestFeature ← ∅
5       FOR each candidate feature X = x in Test:
6         S'← add X = x to S
7         D_S' ← training cases in D_S that satisfy S'
8         IF D_S' ≠ ∅:
9           Score ← EntScore (S) - EntScore(S') calculated using Equation 3
10        IF Score > BestScore: BestFeature ← X = x
11        IF BestFeature ≠ ∅:
12          Add BestFeature to S
13        ELSE: Exit from LOOP
```

Output: Path *S* and $P(T \mid \textbf{\textit{Test}})$ estimated using Equation 1 from training data that satisfy *S*

Appendix Figure 1. Pseudocode for PDP-Ent method.

LazyRF-Ent (D, ***Test***, B)

Inputs:

D is a training dataset consisting of m training cases and n variables and a target variable T

Test is the test case (patient of interest) for which a prediction will be made

B is the number of models in the ensemble

```
1  F ← empty forest of paths
2  P ← empty list of predictions
3  Generate B bootstrap datasets by randomly drawing m training cases from D with replacement
4  FOR each bootstrap dataset:
5     S ← empty path
6     LOOP:
7        BestScore ← 0; BestFeature ← ∅
8  Randomly select √n candidate features from Test
9  FOR each randomly selected candidate feature X = x:
10    S'← add X = x to S
11    D_S' ← training cases in bootstrap dataset that satisfy S'
12    IF D_S' ≠ ∅:
13       Score ← EntScore (S) - EntScore(S') calculated using Equation 3
14       IF Score > BestScore: BestFeature ← X = x
15    IF BestFeature ≠ ∅:
16       Add BestFeature to S
17    ELSE: Exit from LOOP
18    F ← add S to F
19    P ← add P_S(T | Test) which is estimated using Equation 1 from training data that satisfy S
```

Output: Forest of paths ***F*** and $P_F(T \mid \textbf{\textit{Test}})$ calculated as the mean of the predictions in ***P***

Appendix Figure 2. Pseudocode for LazyRF-Ent method.

Appendix B – Comparison of $BayScore_{K2}$ and $BayScore_{BDeu}$ for Binary Targets

Recall the equations for the sample normalized Bayesian scores from 3.1.2.3:

$$BayScore_{K2}(S') = \frac{1}{N}\left[\log\left(\frac{(r-1)!}{(N+r-1)!}\right) + \sum_{k=1}^{r} \log N_k!\right] + q\log\left(\frac{e}{n}\right) + (n-q)\log\left(1-\frac{e}{n}\right)$$

$$BayScore_{BDeu}(S') = \frac{1}{N}\left[\log\left(\frac{\Gamma(\alpha_0)}{\Gamma(N+\alpha_0)}\right) + \sum_{k=1}^{r} \log\left(\frac{\Gamma(N_k+\alpha_k)}{\Gamma(\alpha_k)}\right)\right] + q\log\left(\frac{e}{n}\right) + (n-q)\log\left(1-\frac{e}{n}\right)$$

When a uniform structure prior is used, these scores simplify to:

$$BayScore_{K2}(S') = \frac{1}{N}\left[\log\left(\frac{(r-1)!}{(N+r-1)!}\right) + \sum_{k=1}^{r} \log N_k!\right]$$

$$BayScore_{BDeu}(S') = \frac{1}{N}\left[\log\left(\frac{\Gamma(\alpha_0)}{\Gamma(N+\alpha_0)}\right) + \sum_{k=1}^{r} \log\left(\frac{\Gamma(N_k+\alpha_k)}{\Gamma(\alpha_k)}\right)\right]$$

For a dataset with a binary target, $r = 2$. If $\alpha_0 = 2$ and $r = 2$, $\alpha_k = 1$. Note that $\Gamma(n) = (n-1)!$, and so if N and N_k are integers,

$$BayScore_{BDeu}(S') = \frac{1}{N}\left(\log\left[\frac{\Gamma(2)}{\Gamma(N+2)}\right] + \sum_{k=1}^{r} \log\frac{\Gamma(N_k+1)}{\Gamma(1)}\right)$$

$$= \frac{1}{N}\left(\log\left[\frac{(2-1)!}{(N+2-1)!}\right] + \sum_{k=1}^{r} \log\frac{(N_k+1-1)!}{(1-1)!}\right)$$

$$= \frac{1}{N}\left(\log\left[\frac{(2-1)!}{(N+2-1)!}\right] + \sum_{k=1}^{r} \log\frac{(N_k)!}{1}!\right)$$

$$= \frac{1}{N}\left(\log\left[\frac{(2-1)!}{(N+2-1)!}\right] + \sum_{k=1}^{r} \log N_k!\right)$$

which is equivalent to $BayScore_{K2}$ when $r = 2$.

Appendix C – Comparison of Entropy-Scored Single Path and Boosted Methods

In Appendix Table 1, the mean results from the ensemble of boosted entropy-scored personalized decision paths (BO-PDP-Ent) are compared to those of the single entropy-scored personalized decision path (PDP-Ent) and the population ensemble of boosted entropy-scored decision trees (AB-Ent).

Appendix Table 1. Comparison of results for entropy-scored single path and boosted ensemble methods. Results in blue font indicate statistically significantly worse values than BO-PDP-Ent, and results in red font indicate statistically significantly better values than BO-PDP-Ent on Wilcoxon signed-rank test,.

Dataset	BO-PDP-Ent	PDP-Ent	AB-Ent
Mean AUROC	0.80	0.70 ($p < 0.001$)	0.81 ($p = 0.168$)
Mean ECE	0.13	0.14 ($p = 0.168$)	0.20 ($p = 0.191$)
Mean MPL	3.57	4.00 ($p = 1.00$)	1.00 ($p < 0.001$)

In summary, in terms of discrimination, the ensemble of boosted entropy-scored personalized decision paths has superior performance over the single entropy-scored personalized decision path and equivalent performance to the population ensemble of boosted entropy-scored decision trees. In terms of model complexity, the ensemble of boosted entropy-scored personalized decision paths has more complex models than the population ensemble of boosted entropy-scored decision trees.

Appendix D – Comparison of Base Models and Ensembles

Appendix Table 2 presents the results of base models with corresponding ensemble results.

Appendix Table 2. Comparisons of base models with ensemble models.

Results in red font indicate statistically significantly better value than base model, and results in blue font indicate statistically significantly worse value than base model on Wilcoxon signed-rank test.

Non-Bayesian Decision Tree	**DT-Ent**	**RF-Ent**	**AB-Ent**
Mean AUROC	0.71	0.77 (p = 0.033)	0.81 (p = 0.002)
Mean ECE	0.15	0.05 (p < 0.001)	0.20 (p = 0.684)
Mean MPL	6.02	5.76 (p = 0.497)	1.00 (p < 0.001)
Bayesian Decision Tree	**DT-Bay**	**RF-Bay**	**AB-Bay**
Mean AUROC	0.80	0.81 (p = 0.080)	0.80 (p = 0.414)
Mean ECE	0.05	0.04 (p = 0.685)	0.20 (p = 0.006)
Mean MPL	4.35	3.87 (p = 0.305)	0.89 (p < 0.001)
Non-Bayesian Personalized Decision Path	**PDP-Ent**	**LazyRF-Ent**	**BO-PDP-Ent**
Mean AUROC	0.70	0.80 (p = 0.008)	0.80 (p < 0.001)
Mean ECE	0.14	0.08 (p = 0.033)	0.13 (p = 0.168)
Mean MPL	4.00	3.94 (p = 0.787)	3.57 (p = 1.00)
Bayesian Personalized Decision Path	**PDP-Bay**	**LazyRF-Bay**	**BO-PDP-Bay**
Mean AUROC	0.78	0.82 (p = 0.040)	0.82 (p = 0.168)
Mean ECE	0.13	0.08 (p = 0.033)	0.14 (p = 0.542)
Mean MPL	4.38	3.95 (p = 0.376)	3.65 (p = 0.094)

Appendix E – Training and Prediction Times

In this section, we present the average times for model training and prediction for each algorithmic method. Training times for personalized methods and prediction times for both personalized and population methods were measured using 10 test cases and full training datasets.

Appendix Table 3. Mean training and prediction times for single model algorithmic methods (in seconds). Results in blue font indicate statistically significantly longer mean times than PDP-Bay, and results in red font indicate statistically significantly shorter mean times than PDP-Bay on the Wilcoxon signed-rank test.

Dataset	PDP-Bay	DT-Bay	PDP-Ent	DT-Ent
Training				
chronic-pancreatitis	7.27E-02	91.3	7.27E-02	156
pneumonia	0.135	131	0.135	67.5
sepsis-d	1.35E-02	1.86	1.35E-002	15.8
sepsis-s	9.06E-03	1.28	9.04E-03	29.7
heart-failure-d	3.66E-02	14.9	3.66E-02	158
heart-failure-c	4.70E-02	106	4.70E-02	515
synth-large	2.77	34200	2.77	7830
synth-small	8.20E-02	18.7	8.21E-02	356
Mean	0.396	4320 (p = 0.008)	0.396 (p = 1)	1140 (p = 0.008)
Prediction				
chronic-pancreatitis	0.238	1.98E-05	0.260	3.66E-05
pneumonia	0.379	4.01E-05	0.361	1.85E-05
sepsis-d	4.81E-02	3.02E-05	2.83E-02	1.94E-05
sepsis-s	4.51E-02	9.56E-06	1.35E-02	2.47E-05
heart-failure-d	0.165	1.18E-05	0.104	5.61E-05
heart-failure-c	0.243	1.6E-05	0.132	4.93E-05
synth-large	5.92	3.64E-04	4.51	4.12E-05
synth-small	0.991	1.67E-05	0.514	3.11E-05
Mean	1.00	6.35E-05 (p = 0.008)	0.740 (p = 0.039)	3.46E-05 (p = 0.008)

Appendix Table 4. Mean training and prediction times for random forest methods (in seconds). Results in blue font indicate statistically significantly longer mean times than LazyRF-Bay, and results in red font indicate statistically significantly shorter mean times than LazyRF-Bay on the Wilcoxon signed-rank test.

Dataset	LazyRF-Bay	PDP-Bay	RF-Bay	LazyRF-Ent	RF-Ent
	Training				
chronic-pancreatitis	0.342	7.27E-02	44.4	0.267	59.2
pneumonia	0.457	0.135	38.3	0.321	57.9
sepsis-d	0.121	1.35E-02	1.86	0.103	14.6
sepsis-s	0.115	9.06E-03	5.08	0.107	31.5
heart-failure-d	0.937	3.66E-02	52.6	0.578	285
heart-failure-c	0.823	4.70E-02	292	0.654	829
synth-large	4.24	2.77	6390	3.75	3630
synth-small	3.20	8.20E-02	10.2	2.27	402
Mean	1.28	0.396 (p = 0.008)	854 (p = 0.008)	1.01 (p = 0.008)	664 (p = 0.008)
	Prediction				
chronic-pancreatitis	2.43E-04	0.238	2.05E-04	2.35E-04	2.10E-04
pneumonia	2.79E-04	0.379	2.09E-04	2.30E-04	2.30E-04
sepsis-d	1.92E-04	4.81E-02	1.61E-04	2.00E-04	2.49E-04
sepsis-s	2.18E-04	4.51E-02	1.50E-04	2.38E-04	2.51E-04
heart-failure-d	3.29E-04	0.165	1.91E-04	2.88E-04	3.83E-04
heart-failure-c	3.23E-04	0.243	2.11E-04	3.44E-04	3.94E-04
synth-large	2.62E-04	5.92	2.50E-04	2.45E-04	2.86E-04
synth-small	6.65E-04	0.991	1.29E-04	6.97E-04	2.76E-04
Mean	3.14E-04	1.00 (p = 0.008)	1.88E-04 (p = 0.008)	3.10E-04 (p = 0.844)	2.85E-04 (p = 0.674)

Appendix Table 5. Mean training and prediction times for boosted algorithmic methods (in seconds). Results in blue font indicate statistically significantly longer mean times than BO-PDP-Bay, and results in red font indicate statistically significantly shorter mean times than BO-PDP-Bay on the Wilcoxon signed-rank test.

Dataset	BO-PDP-Bay	PDP-Bay	AB-Bay	BO-PDP-Ent	AB-Ent
	Training				
chronic-pancreatitis	2.61	7.27E-02	3.35	2.49	3.33
pneumonia	3.02	0.135	3.25	2.89	3.04
sepsis-d	0.518	1.35E-02	0.547	0.503	0.579
sepsis-s	0.715	9.06E-03	0.474	0.641	0.729
heart-failure-d	5.15	3.66E-02	3.37	4.76	3.14
heart-failure-c	7.13	4.70E-02	4.16	5.88	3.91
synth-large	66.0	2.77	109	66.5	106
synth-small	10.8	8.20E-02	6.51	11.5	7.21
Mean	12.0	0.396 (p = 0.008)	16.3 (p = 0.742)	11.9 (p = 0.547)	16.0 (p = 1)
	Prediction				
chronic-pancreatitis	2.25E-04	0.238	1.03E-04	2.11E-04	9.86E-05
pneumonia	2.20E-04	0.379	9.88E-05	2.10E-04	1.03E-04
sepsis-d	2.50E-04	4.81E-02	1.01E-04	2.45E-04	1.07E-04
sepsis-s	3.33E-04	4.51E-02	7.87E-05	2.81E-04	1.09E-04
heart-failure-d	4.10E-04	0.165	1.05E-04	4.54E-04	1.15E-04
heart-failure-c	5.16E-04	0.243	1.81E-04	4.58E-04	9.96E-05
synth-large	2.12E-04	5.92	1.04E-04	2.04E-04	9.78E-05
synth-small	3.61E-04	0.991	9.59E-05	4.23E-04	2.02E-04
Mean	3.16E-04	1.00 (p = 0.008)	1.08E-04 (p = 0.008)	3.11E-04 (p = 0.547)	1.17E-04 (p = 0.008)

Overall, mean training times for personalized methods were statistically significantly shorter than population methods for single model and random forest methods, and were not statistically significantly different for boosted methods. Mean training times for personalized methods ranged from 9 milliseconds to 1 minute and 6 seconds, whereas training times for

population methods ranged from 0.5 seconds to 9.5 hours. Training times were largely dependent on dataset size, with datasets that had more predictor variables and more training samples taking more time. Methods that evaluated a greater number of predictors took more time to train models on average.

Prediction times for personalized methods were statistically significantly longer than population methods for all but one population method. Mean prediction times for personalized methods ranged from 0.192 milliseconds to 5.92 seconds, and mean prediction times for population methods ranged from 0.00956 milliseconds to 0.394 milliseconds. These results reflect an opportunity for optimization of our code, as the methods for prediction using decision trees and decision paths should be similar, but our implementation of predicting for personalized methods is clearly less efficient.

Bibliography

1. Knaus, W. A., Zimmermann, J. E., Wagner, D. P., Draper, E. A. & Lawrence, D. E. APACHE - acute physiology and chronic health evaluation: a physiologically based classification system. *Crit. Care Med.* **9**, 591–597 (1981).

2. Apgar, V. A proposal for a new method of evaluation of the newborn infant. *Currrent Res. Anesth. Analg.* 260–267 (1953). doi:10.1213/ANE.0b013e31829bdc5c

3. Beck, A. T., Ward, C. H., Mendelson, M., Mock, J. & Erbaugh, J. An inventory for measuring depression. *Arch. Gen. Psychiatry* **4**, 561–571 (1961).

4. Wilson, P. W. F. *et al.* Prediction of coronary heart disease using risk factor categories. *Circulation* **97**, 1837–1847 (1998).

5. Obermeyer, Z. & Emanuel, E. Predicting the future - big data, machine learning, and clinical medicine. *N. Engl. J. Med.* **375**, 1212–1216 (2016).

6. Gulshan, V. *et al.* Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA* **316**, 2402–2410 (2016).

7. Shimabukuro, D. W., Barton, C. W., Feldman, M. D., Mataraso, S. J. & Das, R. Effect of a machine learning-based severe sepsis prediction algorithm on patient survival and hospital length of stay: a randomised clinical trial. *BMJ Open Respir. Res.* **4**, e000234 (2017).

8. McKinney, S. M. *et al.* International evaluation of an AI system for breast cancer screening. *Nature* **577**, 89–94 (2020).

9. Ding, M. Q., Chen, L., Cooper, G. F., Young, J. D. & Lu, X. Precision oncology beyond targeted therapy: combining omics data with machine learning matches the majority of cancer cells to effective therapeutics. *Mol. Cancer Res.* **16**, 269–278 (2018).

10. Roy, A., Bruce, C., Schulte, P., Olson, L. & Pola, M. Failure prediction using personalized models and an application to heart failure prediction. *Big Data Anal.* **5**, 1–19 (2020).

11. Chouldechova, A. & G'Sell, M. *Fairer and more accurate, but for whom?* (2017).

12. Gianfrancesco, M. A., Tamang, S., Yazdany, J. & Schmajuk, G. Potential biases in machine

learning algorithms using electronic health record data. *JAMA Intern. Med.* **178**, 1544–1547 (2018).

13. Sharafoddini, A., Dubin, J. A. & Lee, J. Patient similarity in prediction models based on health data: a scoping review. *JMIR Med Inf.* **5**, e7 (2017).

14. Friedman, J. H., Kohavi, R. & Yun, Y. Lazy decision trees. *AAAI-96 Proc.* **1**, (1996).

15. Visweswaran, S., Ferreira, A., Ribeiro, G. A., Oliveira, A. C. & Cooper, G. F. Personalized modeling for prediction with decision-path models. *PLoS One* **10**, e0131022 (2015).

16. Ribeiro, G. A. S., de Oliveira, A. C. M., Ferreira, A. L. S., Visweswaran, S. & Cooper, G. F. Patient-specific modeling of medical data. in *Machine Learning and Data Mining in Pattern Recognition* (ed. Perner, P.) **9166**, 415–424 (Springer International Publishing, 2015).

17. Ferreira, A., Cooper, G. F. & Visweswaran, S. Decision path models for patient-specific modeling of patient outcomes. in *AMIA Annual Symposium Proceedings* 413–421 (2013).

18. Johnson, A., Cooper, G. F. & Visweswaran, S. Patient specific modeling with personalized decision paths. in *AMIA Annual Symposium Proceedings* (2020).

19. Margineantu, D. D. & Dietterich, T. G. Improved class probability estimates from decision tree models. in *Nonlinear Estimation and Classification* pp 173-188 (2002).

20. Fern, X. Z. & Brodley, C. E. Boosting lazy decision trees. in *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)* (2003).

21. Chowdhury, M. Z. I. & Turin, T. C. Precision health through prediction modelling: factors to consider before implementing a prediction model in clinical practice. *J. Prim. Health Care* **12**, 3 (2020).

22. Poungvarin, N., Viriyavejakul, A. & Komontri, C. Siriraj stroke score and validation study to distinguish supratentorial intracerebral haemorrhage from infarction. *BMJ* **302**, 1565–1567 (1991).

23. Bleicher, R. J. Timing and Delays in Breast Cancer Evaluation and Treatment. *Ann. Surg. Oncol.* **25**, 2829–2838 (2018).

24. Moons, K. G. M., Royston, P., Vergouwe, Y., Grobbee, D. E. & Altman, D. G. Prognosis and prognostic research: what, why, and how. *BMJ* **338**, b375 (2009).

25. Howell, A. *et al.* Risk determination and prevention of breast cancer. *Breast Cancer Res.* **16**, 1–19 (2014).

26. Beattie, P. & Nelson, R. Clinical prediction rules: What are they and what do they tell us? *Aust. J. Physiother.* **52**, 157–163 (2006).

27. Lee, Y.-H., Bang, H. & Kim, D. J. How to establish clinical prediction models. *Endocrinol Metab* **31**, 38–44 (2016).

28. Waljee, A. K., Higgins, P. D. R. & Singal, A. G. A primer on predictive models. *Clin. Transl. Gastroenterol.* **5**, e44 (2014).

29. Esteva, A. *et al.* Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **542**, 115–118 (2017).

30. Ting, D. S. W. *et al.* Development and validation of a deep learning system for diabetic retinopathy and related eye diseases using retinal images from multiethnic populations with diabetes. *JAMA* **318**, 2211–2223 (2017).

31. Wu, P. Y. *et al.* -Omic and electronic health record big data analytics for precision medicine. *IEEE Trans. Biomed. Eng.* **64**, 263–273 (2017).

32. He, K. Y., Ge, D. & He, M. M. Big data analytics for genomic medicine. *Int. J. Mol. Sci.* **18**, (2017).

33. Kim, J., Campbell, A. S., de Ávila, B. E. F. & Wang, J. Wearable biosensors for healthcare monitoring. *Nature Biotechnology* **37**, 389–406 (2019).

34. Tomczak, K., Czerwińska, P. & Wiznerowicz, M. The Cancer Genome Atlas (TCGA): An immeasurable source of knowledge. *Contemp Oncol* **19**, A68–A77 (2015).

35. Camacho, D. M., Collins, K. M., Powers, R. K., Costello, J. C. & Collins, J. J. Next-generation machine learning for biological networks. *Cell* **173**, 1581–1592 (2018).

36. Goecks, J., Jalili, V., Heiser, L. M. & Gray, J. W. How machine learning will transform biomedicine. *Cell* **181**, 92–101 (2020).

37. Rajkomar, A., Dean, J. & Kohane, I. Machine learning in medicine. *N. Engl. J. Med.* **380**, 1347–58 (2019).

38. Calvert, J. S. *et al.* A computational approach to early sepsis detection. *Comput. Biol. Med.*

74, 69–73 (2016).

39. Ardekani, B. A., Bermudez, E., Mubeen, A. M. & Bachman, A. H. Prediction of incipient Alzheimer's disease dementia in patients with mild cognitive impairment. *J. Alzheimer's Dis.* **55**, 269–281 (2016).

40. Huang, C. *et al.* Machine learning predicts individual cancer patient responses to therapeutic drugs with high accuracy. *Sci. Rep.* **8**, 16444 (2018).

41. Artzi, N. S. *et al.* Prediction of gestational diabetes based on nationwide electronic health records. *Nature Medicine* **26**, 71–76 (2020).

42. Minasyan, G. R., Chatten, J. B., Chatten, M. J. & Harner, R. N. Patient-specific early seizure detection from scalp electroencephalogram. *J. Clin. Neurophysiol.* **27**, 163–178 (2010).

43. Lengerich, B., Aragam, B. & Xing, E. P. Learning sample-specific models with low-rank personalized regression. in *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)* (2019).

44. Cover, T. M. & Hart, P. E. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theory* **13**, 21–27 (1967).

45. Lee, J., Maslove, D. M. & Dubin, J. A. Personalized mortality prediction driven by electronic medical data and a patient similarity metric. *PLoS One* **10**, 1–13 (2015).

46. Ng, K., Sun, J., Hu, J. & Wang, F. Personalized Predictive Modeling and Risk Factor Identification using Patient Similarity. in *AMIA Jt Summits Transl Sci Proc.* 132–6 (2015).

47. Park, Y.-J., Kim, B.-C. & Chun, S.-H. New knowledge extraction technique using probability for case-based reasoning: application to medical diagnosis. *Expert Syst.* **23**, 2–20 (2006).

48. Lee, J. Patient-specific predictive modeling using random forests: an observational study for the critically ill. *JMIR Med. Informatics* **5**, e3 (2017).

49. Remeseiro, B. & Bolon-Canedo, V. A review of feature selection methods in medical applications. *Comput. Biol. Med.* **112**, 103375 (2019).

50. Campillo-Gimenez, B., Jouini, W., Bayat, S. & Cuggia, M. Improving case-based reasoning systems by combining K-nearest neighbour algorithm with logistic regression in the prediction of patients' registration on the renal transplant waiting list. *PLoS One* **8**, e71991

(2013).

51. Li, J., Wu, L., Dani, H. & Liu, H. Unsupervised personalized feature selection. in *32nd AAAI Conference on Artificial Intelligence* 3514–3521 (2018).

52. Lengerich, B., Aragam, B. & Xing, E. P. Every sample a task: pushing the limits of heterogeneous models with personalized regression. in *ATML* (2019).

53. Lengerich, B. J., Aragam, B. & Xing, E. P. Personalized regression enables sample-specific pan-cancer analysis. *Bioinformatics* **34**, i178–i186 (2018).

54. Garbe, C. *et al.* Primary cutaneous melanoma. Identification of prognostic groups and estimation of individual prognosis for 5093 patients. *Cancer* **75**, 2484–2491 (1995).

55. Hilbert, J. P., Zasadil, S., Keyser, D. J. & Peele, P. B. Using decision trees to manage hospital readmission risk for acute myocardial infarction, heart failure, and pneumonia. *Appl. Health Econ. Health Policy* **12**, 573–585 (2014).

56. El-Solh, A. A., Stubeusz, D. L., Grant, G. B. & Grant, B. J. B. Outcome of AIDS patients requiring mechanical ventilation predicted by recursive partitioning. *Chest* **109**, 1584–1590 (1996).

57. Kotsiantis, S. B. Decision trees: a recent overview. *Artif. Intell. Rev.* **39**, 261–283 (2013).

58. Diprose, W. K. *et al.* Physician understanding, explainability, and trust in a hypothetical machine learning risk calculator. *J. Am. Med. Informatics Assoc.* **27**, 592–600 (2020).

59. Laber, E. B. & Zhao, Y. Q. Tree-based methods for individualized treatment regimes. *Biometrika* **102**, 501–514 (2015).

60. Stiglic, G. *et al.* Interpretability of machine learning-based prediction models in healthcare. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **10**, 1–13 (2020).

61. Narayanan, M. *et al.* How do humans understand explanations from machine learning systems? An evaluation of the human-interpretability of explanation. 1–21 (2018).

62. Elshawi, R., Al-Mallah, M. H. & Sakr, S. On the interpretability of machine learning-based model for predicting hypertension. *BMC Med. Inform. Decis. Mak.* **19**, (2019).

63. Lundberg, S. M. & Lee, S. I. A unified approach to interpreting model predictions. in *Advances in Neural Information Processing Systems* 4766–4775 (2017).

64. Ribeiro, M. T., Singh, S. & Guestrin, C. 'Why should I trust you?' Explaining the predictions of any classifier. in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 1135–1144 (2016). doi:10.1145/2939672.2939778

65. Tajgardoon, M., Samayamuthu, M. J., Calzoni, L. & Visweswaran, S. Patient-specific explanations for predictions of clinical outcomes. *ACI Open* **03**, e88–e97 (2019).

66. Barda, A. J., Horvat, C. M. & Hochheiser, H. A qualitative research framework for the design of user-centered displays of explanations for machine learning model predictions in healthcare. *BMC Med. Inform. Decis. Mak.* **20**, 1–16 (2020).

67. Forcier, M. B., Gallois, H., Mullan, S. & Joly, Y. Integrating artificial intelligence into health care through data access: Can the GDPR act as a beacon for policymakers? *J. Law Biosci.* **6**, 317–335 (2019).

68. Murthy, S. K. Automatic construction of decision trees from data: a multi-disciplinary survey. *Data Min. Knowl. Discov.* **2**, 345–389 (1998).

69. Vilalta, R., Valerio, R., Ocegueda-Hernandez, F. & Watts, G. The effect of the fragmentation problem in decision tree learning applied to the search for single top quark production. *J. Phys. Conf. Ser.* **219**, (2010).

70. Berzal, F., Cubero, J. C., Cuenca, F. & Martín-Bautista, M. J. On the quest for easy-to-understand splitting rules. *Data Knowl. Eng.* **44**, 31–48 (2003).

71. Buntine, W. Learning Classification Trees. (1991).

72. Chipman, H. A., George, E. I. & McCulloch, R. E. Bayesian CART model search. *Journal of the American Statistical Association* **93**, 935–948 (1998).

73. Linero, A. R. A review of tree-based Bayesian methods. *Commun. Stat. Appl. Methods* **24**, 543–559 (2017).

74. Koller, D. & Friedman, N. *Probabilistic Graphical Models: Principles*. *Italica* **51**, (1974).

75. Cooper, G. F. & Herskovits, E. A Bayesian method for the induction of probabilistic networks from data. *Mach. Learn.* **9**, 309–347 (1992).

76. Heckerman, D., Geiger, D. & Chickering, D. M. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Mach. Learn.* **20**, 197–243 (1995).

77. Steck, Harald and Jaakkola, T. S. On the Dirichlet prior and Bayesian regularization. *Adv. Neural Inf. Process. Syst.* 713--720 (2003). doi:10.1.1.19.5502

78. Breiman, L. Bagging predictions. *Mach. Learn.* **24**, 123–140 (1996).

79. Domingos, P. Why does bagging work? A Bayesian account and its implications. *Proc. Third Int. Conf. Knowl. Discov. Data Min.* 2–5 (1997).

80. Hastie, T., Tibshirani, R. & Friedman, J. *The elements of statistical learning*. **1**, (Springer, 2009).

81. Freund, Y. & Schapire, R. E. Experiments with a new boosting algorithm. *Proc. 13th Int. Conf. Mach. Learn.* 148–156 (1996). doi:10.1.1.133.1040

82. Mitchell, T. M. *Machine Learning*. (McGraw-Hill, 1997).

83. Domingos, P. Bayesian averaging of classifiers and the overfitting problem. *Icml* 223–230 (2000).

84. Breiman, L. Random forests. *Mach. Learn.* **45**, 5–32 (2001).

85. Lee, J. Personalized mortality prediction for the critically ill using a patient similarity metric and bagging. in *2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)* 332–335 (Institute of Electrical and Electronics Engineers Inc., 2016). doi:10.1109/BHI.2016.7455902

86. Visweswaran, S. & Cooper, G. F. Learning instance-specific predictive models. *J. Mach. Learn. Res.* **11**, 3333–3369 (2010).

87. Visweswaran, S. *et al.* Learning patient-specific predictive models from clinical data. *J. Biomed. Inform.* **43**, 669–685 (2010).

88. Xu, R., Nettleton, D. & Nordman, D. J. Case-specific random forests. *J. Comput. Graph. Stat.* **25**, 49–65 (2016).

89. Whitcomb, D. C. *et al.* Multicenter approach to recurrent acute and chronic pancreatitis in the United States: The North American Pancreatitis Study 2 (NAPS2). *Pancreatology* **8**, 520–531 (2008).

90. Fine, M. J. *et al.* Processes and outcomes of care for patients with community-acquired pneumonia: results from the Pneumonia Patient Outcomes Research Team (PORT) cohort

study. *Arch. Intern. Med.* **159**, 970–980 (1999).

91. Kellum, J. A. *et al.* Understanding the inflammatory cytokine response in pneumonia and sepsis: results of the Genetic and Inflammatory Markers of Sepsis (GenIMS) study. *Arch. Intern. Med.* **167**, 1655–1663 (2007).

92. Auble, T. E. *et al.* A prediction rule to identify low-risk patients with heart failure. *Acad. Emerg. Med.* **12**, 514–521 (2005).

93. Hernandez, R. D., Uricchio, L. H., Hartman, K., Ye, C. & Zaitlen, N. Ultra-rare variants drive substantial cis-heritability of human gene expression. *Nat Genet* **51**, 1349–1355 (2019).

94. Dua, D. & Graff, C. UCI Machine Learning Repository. *Irvine, CA: University of California, School of Information and Computer Science.* (2019).

95. Wolberg, W. H. & Mangasarian, O. L. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proc. Natl. Acad. Sci. U. S. A.* **87**, 9193–9196 (1990).

96. Chicco, D. & Jurman, G. Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. *BMC Med. Inform. Decis. Mak.* **20**, 1–16 (2020).

97. Naz, H. & Ahuja, S. Deep learning approach for diabetes prediction using PIMA Indian dataset. *J. Diabetes Metab. Disord.* **19**, 391–403 (2020).

98. Naeini, P., Cooper, G. F. & Hauskrecht, M. Obtaining well calibrated probabilities using Bayesian binning. *Twenty-Ninth AAAI Conf. Artif. Intell.* 2901–2907 (2015).

99. Carvalho, A. Scoring functions for learning bayesian networks. *Inesc-id Tec. Rep* 1–48 (2009).

100. Suresh, H. & Guttag, J. V. *A Framework for Understanding Unintended Consequences of Machine Learning*. (2019).

101. Köchling, A. & Wehner, M. C. Discriminated by an algorithm: a systematic review of discrimination and fairness by algorithmic decision-making in the context of HR recruitment and HR development. *Bus. Res.* 1–54 (2020). doi:10.1007/s40685-020-00134-w

102. Buolamwini, J. & Gebru, T. Gender shades: intersectional accuracy disparities in commercial gender classification. in *Conference on Fairness, Accountability, and Transparency* **81**, 1–15 (2018).

103. Obermeyer, Z., Powers, B., Vogeli, C. & Mullainathan, S. Dissecting racial bias in an algorithm used to manage the health of populations. *Science (80-.).* **366**, 447–453 (2019).

104. Parikh, R. B., Teeple, S. & Navathe, A. S. Addressing bias in artificial intelligence in health care. *JAMA* **322**, 2377–2378 (2019).

105. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K. & Galstyan, A. *A Survey on Bias and Fairness in Machine Learning*. (2019).

106. Chouldechova, A. & Roth, A. *The Frontiers of Fairness in Machine Learning*. (2018).

107. Sateren, W. B. *et al.* How sociodemographics, presence of oncology specialists, and hospital cancer programs affect accrual to cancer treatment trials. *J. Clin. Oncol.* **20**, 2109–2117 (2002).

108. Sardar, M. R., Badri, M., Prince, C. T., Seltzer, J. & Kowey, P. R. Underrepresentation of women, elderly patients, and racial minorities in the randomized trials used for cardiovascular guidelines. *JAMA Intern. Med.* **174**, 1868–1870 (2014).

109. Nazha, B., Mishra, M., Pentz, R. & Owonikoko, T. K. Enrollment of racial minorities in clinical trials: old problem assumes new urgency in the age of immunotherapy. *Am. Soc. Clin. Oncol. Educ. B.* **39**, 3–10 (2019).

110. Freedman, L. S. *et al.* Inclusion of women and minorities in clinical trials and the NIH Revitalization Act of 1993 - The perspective of NIH clinical trialists. *Control. Clin. Trials* **16**, 277–285 (1995).

111. Hutchins, L. F., Unger, J. M., Crowley, J. J., Coltman, C. A. & Albain, K. S. Underrepresentation of patients 65 years of age or older in cancer-treatment trials. *N. Engl. J. Med.* **341**, 2061–2067 (1999).

112. Gifford, A. L. *et al.* Participation in research and access to experimental treatments by HIV-infected patients. *N. Engl. J. Med.* **346**, 1373–1382 (2002).

113. Robbins, N. M. & Bernat, J. L. Minority representation in migraine treatment trials. *Headache* **57**, 525–533 (2017).

114. Ayanian, J. Z., Landon, B. E., Newhouse, J. P. & Zaslavsky, A. M. Racial and ethnic disparities among enrollees in medicare advantage plans. *N. Engl. J. Med.* **371**, 2288–2297 (2014).

115. Chen, I. Y., Johansson, F. D. & Sontag, D. Why is my classifier discriminatory? in *32nd Conference on Neural Information Processing Systems (NeurIPS 2018)* 3539–3550 (2018).

116. Wiens, J., Saria, S. & Goldenberg, A. Do no harm: a roadmap for responsible machine learning for health care. *Nat. Med.* **25**, 1337–1340 (2019).

117. Hague, D. C. Benefits, Pitfalls, and Potential Bias in Health Care AI. *N. C. Med. J.* **80**, 219–223 (2019).

118. Rajkomar, A., Hardt, M., Howell, M. D., Corrado, G. & Chin, M. H. Ensuring fairness in machine learning to advance health equity. *Ann. Intern. Med.* **169**, 866–872 (2018).

119. McCradden, M. D., Joshi, S., Mazwi, M. & Anderson, J. A. *Ethical limitations of algorithmic fairness solutions in health care machine learning*. (2020). doi:10.1016/S2589-7500(20)30065-0

120. Paulus, J. K. & Kent, D. M. Predictably unequal: understanding and addressing concerns that algorithmic clinical prediction may increase health disparities. *npj Digit. Med.* **3**, 1–8 (2020).

121. Verma, S. & Rubin, J. Fairness definitions explained. in *Proceedings - International Conference on Software Engineering* 1–7 (2018). doi:10.1145/3194770.3194776

122. Chen, I. Y., Szolovits, P. & Ghassemi, M. *Can AI Help Reduce Disparities in General Medical and Mental Health Care? AMA Journal of Ethics* **21**, (2019).

123. Pfohl, S. *et al.* Creating fair models of atherosclerotic cardiovascular disease. in *AIES 2019 - Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society* 271–278 (2019). doi:10.1145/3306618.3314278

124. Seyyed-Kalantari, L., Liu, G., McDermott, M. & Ghassemi, M. *CheXclusion: Fairness gaps in deep chest X-ray classifiers*. (2020).