UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

Master's thesis
(Magistrsko delo)

Automated pipeline for binary classification of high-dimensional biomedical data

(Avtomatiziran cevovod binarne klasifikacije visoko dimenzionalnih biomedicinskih podatkov)

Ime in priimek: *Đorđe Klisura*
Študijski program: *Podatkovna znanost, 2. stopnja*
Mentor: *izr. prof. dr. Leo Lahti*
Somentor: *izr. prof. dr. Marko Tkalčič*

Koper, avgust 2022

Ključna dokumentacijska informacija

Ime in PRIIMEK: Đorđe KLISURA

Naslov magistrskega dela: Avtomatiziran cevovod binarne klasifikacije visoko dimenzionalnih biomedicinskih podatkov

Kraj: Koper
Leto: 2022

Število listov: 60 Število slik: 15 Število tabel: 5
Število prilog: 6 Število strani prilog: 9
Število referenc: 34

Mentor: Izr. prof. dr. Leo Lahti
Somentor: Izr. prof. dr. Marko Tkalčič

UDK: 004.8(043.2)

Ključne besede: cevovod, binarna klasifikacija, biomedicinski podatki, borelioza

Izvleček:
Metabolomika se v kliničnem okolju vse pogosteje uporablja za diagnosticiranje bolezni, prognozo in napovedovanje tveganja. Algoritmi strojnega učenja so še posebej pomembni pri izgradnji modela, ki temelji na metabolitnih podatkih. Ta magistrski projekt je vključeval razvoj cevovoda strojnega učenja za binarno kategorizacijo borelioze. Prikazali smo nov potek dela, ki uporabnikom omogoča uvajanje in usposabljanje modelov z uporabo visokodimenzionalnih (biomedicinskih) podatkov. Naši prispevki gredo v smeri posodobitve in razširitve cevovoda, ki je prej vključeval predvsem usposabljanje in uvajanje modelov. Dodali smo faze urejanja, čiščenja in obdelave manjkajočih vrednosti podatkov, izbire lastnosti, vizualizacije podatkov, prilagajanja modela in ocene modela, ki jo sestavljajo grafične primerjave metrik uspešnosti modela. Razvili smo odprtokodno izvajanje tega cevovoda. Za našo študijo smo imeli na voljo dva nabora podatkov: enega, ki vsebuje značilnosti XCMS za učne vzorce seruma - 4 851-dimenzionalni vektor značilnosti intenzivnosti, in drugi nabor podatkov, ki vsebuje 118 bolnikov s stanjem bolezni, razvrščenih kot: (i) zgodnja razširjena borelioza (EDL), (ii) zgodnja lokalizirana borelioza (ELL), (iii) zdrava kontrola neendemična (HCN) in (iv) zdrava kontrola wormser (HCE1). Naši klasifikatorji so se pri diagnosticiranju borelioze izkazali precej dobro, pri čemer je bil najboljši model XGBoost s površino pod krivuljo značilnosti delovanja 0,98 (IQR 0,97 do 0,98).

Key document information

Name and SURNAME: Đorđe KLISURA
Title of the thesis: Automated pipeline for binary classification of high-dimensional biomedical data

Place: Koper
Year: 2022

Number of pages: 60 Number of figures: 15 Number of tables: 5
Number of appendices: 6 Number of appendix pages: 9
Number of references: 34

Mentor: Assoc. prof. Leo Lahti, PhD
Co-Mentor: Assoc. prof. Marko Tkalčič, PhD

UDC: 004.8(043.2)

Keywords: pipeline, binary classification, biomedical data, lyme disease

Abstract:
Metabolomics is increasingly being used in the clinical setting for disease diagnosis, prognosis and risk prediction. Machine learning algorithms are particularly important in the construction of a metabolite-data-based model. This thesis project involved the development of a machine learning pipeline for the binary categorization of Lyme disease. We demonstrated a novel workflow enabling users to deploy and train models using high-dimensional (biomedical) data. Our contributions go toward updating and expanding the pipeline that previously mainly involved model training and deployment. We added the phases of data organization, cleaning, and handling of missing values, feature selection, data visualization, model fitting, and a model assessment comprised of graphical comparisons of the model's performance metrics. We developed an open-source implementation of the pipeline. For our study we had the two datasets: the one that contains XCMS features for the training serum samples - 4,851 dimensional feature vector of intensities and the other dataset that contains 118 patients with disease state classified as: i) Early Disseminated Lyme (EDL), ii) Early Localized Lyme (ELL), iii) Healthy Control Non-Endemic (HCN) and iv) Healthy Control Wormser (HCE1). Our classifiers performed quite well in diagnosing Lyme disease, with XGBoost model being the best with an area under the operating characteristic curve of 0.98 (IQR 0.97 to 0.98).

List of Contents

List of Tables

List of Figures

List of Appendices

List of Abbreviations

i.e.	that is
e.g.	for example
ML	Machine learning
$Spec$	Specificity
$Sens$	Sensitivity
Acc	Accuracy
$AUROC$	Area under the ROC curve
OTU	Operational taxonomic unit
PCA	Principal component analysis
$UMAP$	Uniform Manifold Approximation and Projection

Acknowledgments

First and foremost, I would like to thank my mentor and professor Leo Lahti for his assistance and ideas during the preparation, development, and writing of my master's thesis.

Then, I want to express my gratitude to the University of Turku for giving me the chance and financial support to work on my thesis for four months at the Department of Computing.

I would also want to thank my co-mentor, prof. Marko Tkalčič, for his guidance and assistance during the thesis writing process.

I would want to show my appreciation to my friends Đorđe and Aleksandar, who have been there for me for the last five years and have supported me in every aspect of my professional, academic, and personal development.

I would want to express my gratitude to Bernie, Anita, and Florian, whom I met throughout this experience and who have always been there for me.

Na kraju, želeo bih da se zahvalim svojim roditeljima koji su me uvek podržavali u svemu i bez kojih ništa ne bi bilo moguće, kao i bratu i sestri koji su sve prolazili sa mnom.

1 INTRODUCTION

1.1 MOTIVATION

The allied fields of informatics within the space of biology and medicine have existed in various forms for several decades: medical informatics, clinical informatics, health informatics, bioinformatics, and biomedical informatics. These fields all refer to the development of methods to analyze biomedical data [1]. The application of big data tools in biomedical and clinical research is rapidly growing as an enormous volume of biological and clinical data has been produced and gathered in the past years [18]. In conducting clinical research, frequent data gathering methods include questionnaire surveys, patient self-reported data, proxy/informant information, hospital and ambulatory medical records, and the collection and evaluation of biological samples [33].

Biomarkers are one of the most popular biological samples for data collection in clinical research due to the simplicity of data measurement and their ability to discriminate between diseased and healthy patient samples. They are detected (small molecules) and quantified in areas of the body like the blood or tissue, and they give accurate biological or physical indications of disease states [17]. Gene profiling technologies, such as DNA microarrays and RNA sequencing, provide large gene data sets which enable data-driven biomarker and disease discovery [26].

With the expansion of these data sets, there is a rising need to apply machine learning (ML) to construct models for identifying metabolomic biomarkers (small-molecule metabolites that are useful for understanding disease phenotypes), which help in the diagnosis, prognosis, and risk evaluation of disease. Machine learning is frequently applied to biological data to produce user-friendly approaches (ML pipelines). In this research, we examine patient data consisting of metabolomic biomarkers that will be utilized to develop a machine learning pipeline for classifying Lyme disease. [31].

Lyme disease is a condition that is spreading quickly but is still not well understood by experts. It is the most common disease transmitted by ticks in Eastern Europe. Typically, a patient may have non-specific symptoms including weariness, malaise, and joint and muscular aches along with a skin lesion. Not all patients have or detect an EM skin lesion, despite the fact that it is the most typical sign of Lyme disease and is utilized for clinical diagnosis in endemic regions, making it more difficult to diagnose. Consequently, it makes logical to use ML pipeline to such a situation. [10].

When discussing machine learning pipelines, diverse approaches are centered on training and assessing the model, but not on visualizing the original data sets that can give us a sense of outliers and clusters, selecting the characteristics that have the most significant impact on the prediction that can allow users to reduce training time and to choose optimal biomarkers, and the data preparation process, that can be useful in the process of normalization or treating missing values in biomedical data as they are common.

Patrick et al. developed a machine learning (ML) pipeline for microbiome-based classification challenges for training, validating, and evaluating ML models [31]. The preparation phase of the pipeline does not involve the handling of missing values, which are frequent in such data sets. Also, the pipeline does not permit best feature selection, which might be a significant drawback given that these data sets are large (with many attributes) and it could take days to weeks to train models on the entire dataset.

Another study by Yazdani et al. illustrates a pipeline that selects significant features using the Kolomogorov-Smirnov test [20]. However, if the sample size is small, the two distributions must be entirely different for this test to demonstrate statistical significance, which is a limitation of this technique.

The absence of preprocessing techniques, as well as feature significance selection, in the existing ML pipelines for processing biological data, will be addressed in this thesis. We will update a current pipeline by designing a reproducible, automated process for data preparation, feature selection, data presentation, and model training/evaluation. Two linear and three nonlinear models will be trained on the basis of the open-source metabolomic profiling data, to categorize the state of Lyme disease, where the outcome is binary.

The research questions that we will be answering in this study include whether and how accurately the Lyme disease status can be properly classified based on open-source metabolomic profiling data, using updated and automated machine learning (ML) pipeline for biomedical data that addresses the current interpretability and data handling flaws. Which classifier, in terms of model performance based on training time and performance metrics including balanced accuracy, area under the receiving characteristic curve (AUROC), F1 statistics, and Kappa, performs the best at identifying patients with Lyme disease.

1.2 STRUCTURE OF THE THESIS

In chapter 2, we discuss related works and how our work extends to them.

In the chapter 3, we describe terminology that are necessary for comprehension of the work.

In chapter 4, we recreate the research to develop the work of the existing pipeline from *mikropml* package and we outline the additional stages and the means by which we adapt the existing pipeline to process and visualize and train five models based on metabolomic data.

In chapter 5, we present and examine the findings, while in chapter 6, we discuss the findings.

2 RELATED WORK

A current problem, that this thesis will be dealing with, is the lack of methods in the ML pipeline for data preparation, visualization and feature importance selection. Numerous research has addressed frameworks and automated processes for the application of biological data, and we give a selection of them in the following paragraphs.

Schloss et al. proposed steps toward standardization of machine learning approaches for microbiome research, which are often inadequately documented and implemented, in their paper [31]. In order to demonstrate a rigorous machine learning (ML) pipeline and shed light on how ML model selection can impact modeling results, an empirical analysis comparing the predictive performance, interpretability, data requirements, and training times of seven modeling approaches using the same data set and pipeline is conducted. They created a machine learning pipeline that can be used to a variety of model training and model evaluation situations. Other host-associated and environmental microbiome data sets may simply use this framework. This framework serves as the foundation for our study, allowing us to train our model using the techniques it offers. This pipeline also has a data preparation method that does not at all handle missing values. It simply eliminates them all. It can be problematic when, as in our situation, the outcome variable (disease state) of a dataset has missing values. Due to the fact that we have just 118 samples with over 4800 characteristics, this methodology would drastically reduce the number of samples, leaving us with insufficient samples for the model training procedure.

In the paper, *Biomarker selection and a prospective metabolite-based machine learning diagnostic for lyme disease* accessible at [10], a pipeline for data preprocessing, biomarker selection, and classification of serum samples from liquid chromatography -mass spectrometry (LCMS) is outlined in order to create a prospective diagnostic test for lyme disease. The number of biomarkers are selected in the study and created a discriminant model for Lyme disease with high accuracy using machine learning (ML) technologies such as sparse support vector machines (SSVM), iterative feature removal (IFR), and k-fold feature ranking. The selection of the classifier is a drawback of this pipeline. Pipeline only supports one approach for model training and is built for a specific purpose. It is not reusable for all biological data types and is not automated, necessitating that the user possess additional knowledge to operate it.

Last but not least, we draw attention to the study *Predicting Lyme Disease from Patients' Peripheral Blood Mononuclear Cells Profiled with RNA-Sequencing*, where

several classifiers are trained using machine learning algorithms in order to generate unique classifications of Lyme disease. Here, classifiers were applied to the entire raw data without following a rigorous machine learning procedure (pipeline). This raises the question of whether this omission resulted in a failure to identify people with early Lyme disease. The study is accessible at [6].

Our objective is to address the deficiencies of existing pipelines by designing a reproducible, automated process for data preparation, feature selection, data presentation, and model training/evaluation, with benefits such as feature selection using the sum rank Wilcoxon test, dimension-reduction visualizations, and k-nn imputation of missing values. We will apply the pipeline to the open-source metabolomic profiling data and train two linear models: logistic regression and SVM with radial basis kernel, as well as three nonlinear models: decision trees, random forest, and gradient boosted trees, in order to evaluate the models' ability to classify the binary state of Lyme disease, after passing through our stringent process.

3 PRELIMINARIES

In this chapter, we define and discuss the tools for microbial analysis, machine learning algorithms, visualization approaches, and statistical tests used in our research.

3.1 TOOLS FOR MICROBIAL ANALYSIS

In order to evaluate variations in microbial communities across body locations and people, researchers may now create millions of sequences per sample. Because of the improved sequencing capability, it has been necessary to build computational tools that are just as powerful in order to manage the growing volume of sequence data that emerging technologies are producing. The RDP (Ribosomal Database Project) pyrosequencing tools, mothur, WATERS (Workflow for the Alignment, Taxonomy, and Ecology of the Microbial Environment), and QIIME (Quantitative Insights Into Microbial Ecology, pronounced "chime") are a few pipelines for evaluating microbial community data [34].

In microbial ecology, OTUs or clusters with sequence similarity at the molecular level are generally acknowledged as an analytical unit. When examining gene sequence databases, OTUs have been the most often utilized units throughout the whole history of taxonomy. Today, OTU is often used in a variety of applications. In most instances, it refers to clusters of uncultivable microorganisms that are grouped based on the similarity of their DNA sequences. This resemblance is recognized by a particular gene marker for taxonomy. In other words, in the lack of standard biological categorization systems, OTUs are employed as pragmatic substitutes for microorganisms at various taxonomic levels. For 16S rRNA data, sequences must be grouped into OTUs [9].

Our study's dataset comprises of more than 4800 OTU abundances from 118 patients.

3.2 ML ALGORITHMS FOR CLASSIFICATION TASKS

Classification is a task that entails the application of machine learning algorithms to learn how to give a class label to problem domain instances. Binary classification refers to classification tasks with two class labels. The regular state is represented by one class in most binary classification issues, while the aberrant situation is represented by

the other class [13].

Many machine learning algorithms may be used to tackle a binary classification problem. Table 1 presents the methods that will be used in this study to tackle the problem. The following subsections provide additional in-depth explanations.

Table 1: Characteristics of the machine learning models in our study.

Model	**Brief description**	**Linearity**
Logistic regression	A predictive regression analysis when the dependent variable is binary	Linear
SVM with linear kernel	A classifier that distinguishes between labels and is specified by an ideal linear separation hyperplane	Linear
Decision tree	A classifier that analyzes an attribute's ability to distinguish between labels when sorting data from the root to the leaf node	Nonlinear
Random forest	A classifier that is an ensemble of decision trees that grows randomly with subsampled data	Nonlinear
Gradient boosted trees	A classifier that is an ensemble of greedily growing decision trees	Nonlinear

3.2.1 Logistic regression

An efficient supervised machine learning approach for binary classification issues is logistic regression (when target is categorical). In essence, logistic regression models a binary output variable using the logistic function given below. Despite its name, logistic regression is more of a classification model than a regression model.

$$P(y = 1) = \frac{1}{1 - e^{-x}}$$

Overfitting the data is a common problem with logistic regression models, especially in high-dimensional contexts (cases with lots of predictors). Because of this, regularization techniques are often used to prevent the model from fitting the training data too closely. In our study, we use L2 regularized logistics regression, also known as Ridge Regression.

For situations involving binary and linear classification, logistic regression is a straightforward and more effective approach. It's a classification model that's incredibly simple to implement and performs well with linearly separable classes. It is a widely used categorization method that is very efficient. Multiclass classification may be used to the logistic regression model in general [8].

3.2.2 Decision trees

Regression and classification issues may be resolved using the decision tree algorithm. By learning simple decision rules derived from previous data, a Decision Tree is used to build a training model that may be used to predict the class or value of the target variable (training data). By analyzing a tree of if-then-else true/false feature questions and calculating the bare minimum of questions required to evaluate the likelihood of choosing the right choice, decision trees provide a model that predicts the label [12].

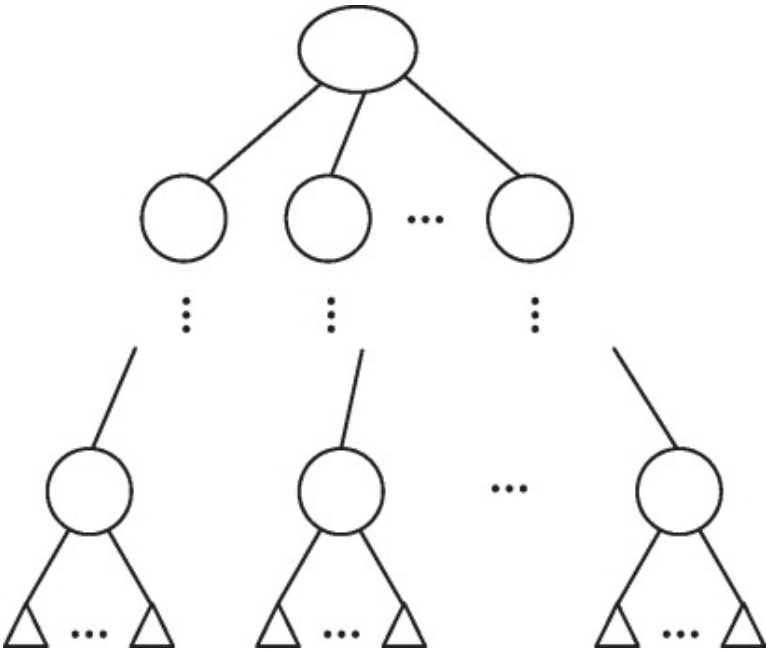

Figure 1: Decision tree classifier.

A basic decision tree, as illustrated in Figure 1, is made up of one root node, a number of internal and leaf nodes, and branches.

In decision trees, we begin at the tree's root when forecasting a record's class label. We contrast the root attribute's values with those of the attribute on the record. We follow the branch that corresponds to that value and go on to the next node based on the comparison. To determine whether to divide a node into two or more sub-nodes, decision trees use a variety of techniques. The homogeneity of newly formed sub-nodes is increased by sub-node formation. In other words, we may claim that the node's purity improves in relation to the desired variable. The decision tree divides the nodes based on all factors that are accessible before choosing the split that produces the most homogenous sub-nodes [22].

The decision tree classifier's key benefit is its flexibility to use multiple feature subsets and decision rules at different stages of categorization.

3.2.3 Random forest

The Random Forest (RF) classifiers vary significantly from the other ML techniques. They fall under the category of ensemble classifiers, which combine predictions from many learned base classifiers. Each basis classifier must be distinct and vary in either the method being used, the hyperparameter values, or the training data in order to prevent significant correlation across base classifiers, which in turn results in overfitting. A decision tree serves as the primary classifier for RFs. Consequently, we are working with a collection of several decision trees (a forest of random decision trees).

In order to address the overfitting issue, random forest classifiers combine many trees (usually 100+ trees). It specifically makes advantage of CART optimization (Classification and Regression Tree). By only permitting a random subset of characteristics to be used as the basis for the split at each node, the technique significantly lessens the correlation problem that was previously discussed (typically the number in this subsample is equal to the square root of the total number of available features) [21].

3.2.4 SVM with radial basis kernel

The purpose of the linear kernel support vector machine (SVM-Lin) technique is to locate, in an M-dimensional space (M = number of features), a hyperplane that classifies the N data points in the X matrix (N x M) in a separate manner. There are several hyperplanes from which to choose when separating two classes of data points. The objective of the SVM method is to identify the orientation (or rotation) of the hyperplane that maximizes the discrimination margin (i.e. the distance between the closest data points at the edge of each class is made as large as possible). Support vectors are the data points that define this margin most precisely. Importantly, and what makes SVM distinctive, is that the process of maximising the margin makes SVM resilient for accurately classifying fresh data that may reside inside that margin on either side of the classification hyperplane (acting like a classification buffer).

By implicitly translating input data into a high-dimensional feature space, SVMs may be designed to do nonlinear classification. This method is referred to as the kernel trick. The goal is to achieve linear separation by mapping the data to a space with more dimensions. There are other kernel functions available, but the radial basis function is the most used (RBF) [21].

3.2.5 Gradient boosted trees

Boosting is an ensemble modelling strategy that seeks to construct a powerful classifier from a collection of weak classifiers. It is accomplished by constructing a model from a sequence of weak models. Initially, a model is constructed using the training data. The

second model is then constructed in an attempt to address the faults in the previous model. This approach is repeated and models are added until the whole training data set is accurately predicted or the maximum number of models has been added.

Extreme Gradient Boosting (XGBoost) is a distributed gradient-boosted decision tree (GBDT) machine learning library that is scalable. It is the leading machine learning package for regression, classification, and ranking problems and offers parallel tree boosting.

In this approach, consecutive decision trees are generated. In XGBoost, weights play a crucial role. All independent variables are allocated weights, which are subsequently input into the decision tree used to predict outcomes. The weight of variables for which the tree made incorrect predictions is raised, and these variables are then given to a second decision tree. The ensemble of these independent classifiers/predictors yields a robust and more accurate model. It can solve issues including regression, classification, ranking, and user-defined prediction [4].

3.2.6 K-Nearest neighbors

The k-nearest neighbors algorithm, usually referred to as KNN or k-NN, is a supervised learning classifier that employs proximity to produce classifications or predictions about the grouping of a single data point.

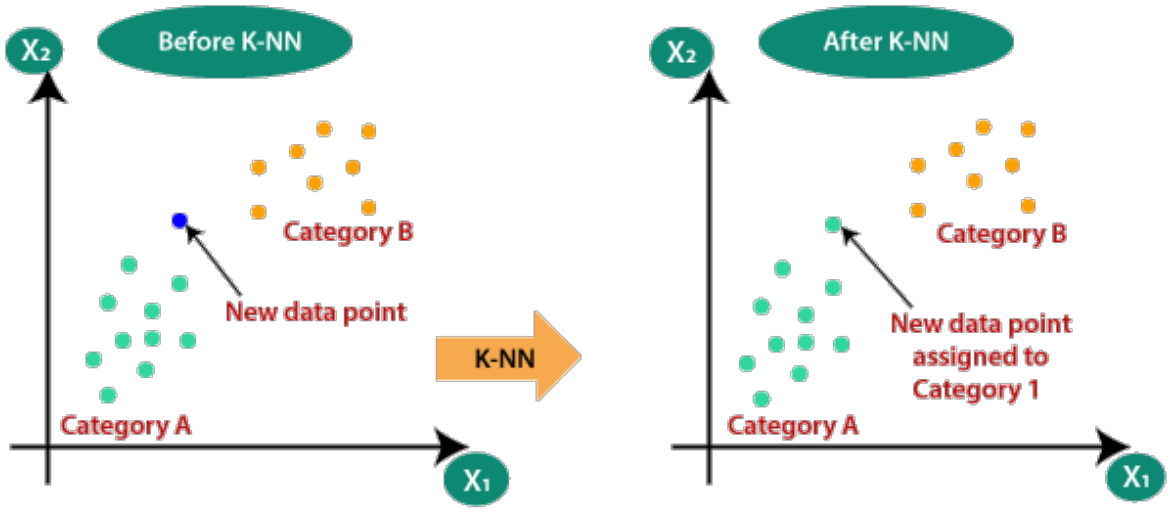

Figure 2: K-NN algorithm.

Although it may be used to classification or regression issues, it is commonly employed as a classification method since it relies on the idea that comparable points can be discovered close to one another. The algorithm's objective is to locate a query

point's closest neighbors so that we may categorize that location. The algorithm's input and output are shown in Figure 2. The k parameter in the k-NN method specifies how many neighbors will be examined to establish a particular query point's categorization.

The distance between the query point and the other data points must be determined in order to discover which data points are closest to a certain query point. These distance measurements aid in the creation of decision borders, which divide query points into several zones. You may choose from a variety of distance units, including Eucledian, Manhattan, Minkowski, Hamming, and others [30].

This approach has advantages such as nonparametric architecture, which is simple and powerful and requires no training time, but it also has disadvantages such as memory requirements and sluggish classification and estimation.

In this study, we impute missing data using the k-Nearest Neighbors method ($k - NN$).

3.2.7 Performance evaluation of the model

Here, we present the measures that will be used to assess the performance of trained models.

One of the many metrics used to evaluate the development of a classification problem is machine learning model accuracy. Accuracy is calculated as the number of accurate predictions divided by the total number of forecasts: $Accuracy = \frac{\#correct}{total}$. A model that consistently made correct predictions would have a score of 1.0. All other things being equal, accuracy is an acceptable statistic to use when the classes in the dataset occur with about the same frequency.

Sensitivity, often known as the "true positive" rate (TPR), assesses how frequently a test accurately yields a positive result for those who have the ailment being tested for. A highly sensitive test will detect virtually all cases of the illness and provide fewer false-negative findings (for instance, a test with a sensitivity of 90% would accurately yield a positive result for 90% of those with the condition, but would produce a false-negative result for 10% of those who should have tested positive but did not).

The specificity or a "true negative" rate (TNG), on the other hand, is a measure of a test's ability to accurately provide a negative result for individuals who do not have the ailment that is being tested for. A test with high specificity won't provide many false-positive findings and will successfully rule out practically everyone who doesn't have the illness (for instance, a test with 90% specificity will accurately yield a negative result for 90% of persons who do not have the illness, but will produce a positive result, a false positive, for 10% of the people who should have tested negative but did not because they do not have the condition) [24].

A measurement tool for binary classification issues is the Receiver Operator Charac-

teristic (ROC) curve. In essence, it separates the "signal" from the "noise" by plotting the true positive rates against the false positive rates at different threshold levels. The capacity of a classifier to differentiate between classes is measured by the Area Under the ROC curve (AUROC), which is used as a summary of the ROC curve.

The model performs better at differentiating between the positive and negative classes the higher the AUROC. AUROC equals 1.00 indicates that the classifier can accurately discriminate between all Positive and Negative class points. The classifier would be predicting all Negatives as Positives and all Positives as Negatives, however, if the AUROC had been 0 [25].

The kappa statistic, which controls for the performance of a random classifier as assessed by the predicted accuracy, is essentially a measurement of how well the instances categorized by the machine learning classifier matched the data designated as ground truth [7].

The last metric that we are going to mention here is a F1 score. In order to understand it, we need to define two terms - the precision and recall. Precision measures the proportion of predictions that come true within the positive predictions and recall determines the number of positive results the model actually found among all positive results. Finally, it brings us to the formula for F1 score, that is defined as the harmonic mean of precision and recall [15].

3.3 HYPERPARAMETER TUNING

In order to train a model for a particular modeling problem, a set of parameters known as hyperparameters must be given or adjusted by the user [2].

The cost hyperparameter, which determines the regularization strength, is tuned for L2-regularized logistic regression and L1- and L2-regularized SVM with linear and radial basis function kernels. Smaller values indicate higher regularization.

The sigma hyperparameter is tuned for SVM with radial basis function kernel and controls the range of a single training instance. For a large value of sigma, the SVM decision boundary will rely on the points that are closest to the decision border.

The decision tree model's tree depth is tuned such that the deeper the tree, the more splits it contains.

Finding the best tree split in random forest requires tuning the amount of features taken into account.

The learning rate is tuned and the fraction of samples used for fitting the individual base learners, for XGBoost (Gradient boosted trees model) [31].

3.4 FEATURE SELECTION

Nowadays, feature selection is used extensively in a variety of classification issues. Feature selection has been the focal point of machine learning and data mining in recent years [16]. Based on their relationship to learning algorithms, feature selection techniques may be categorized as embedding, filter, or wrapper. Embedded techniques include feature selection within the learning algorithm and may choose the appropriate features during training. Filter techniques concentrate on assessing the qualities of data to produce the subset of features, and do not use any learning algorithms. As assessment criterion, wrapper strategies use learning algorithms to discover these pertinent aspects. Filter techniques are the most used in empirical research [28].

There are several similarity metrics in machine learning algorithms, including Euclidean, Manhattan, and Hamming distances. Distance, information, dependence, and consistency measurements are the criteria for evaluating filter techniques.

Class prediction based on high-dimensional biomedical data has been the subject of several methodological and applied studies with applications to, for example, the molecular diagnosis of cancer or the prediction of therapeutic response. Even if the selected classification technique is capable of handling numerous predictors, it is customary to do univariate variable selection prior to building a classifier. In binary classification, it is typical to rank genes according to the P-value obtained in, for example, the t-test for two independent samples and related techniques or the Wilcoxon rank sum test, also known as the Mann-Whitney test [3].

The features with the lowest p-values are then chosen and utilized to build the classifier. The Wilcoxon rank sum test does not need normal distribution of the expression levels within both groups, unlike the t-test, and is resistant against outliers, which are common in microarray data. This is a significant benefit since gene expression data normality is sometimes in doubt even after normalization. One of the largest comparative studies on microarray-based classification has shown that Wilcoxon-based variable selection performs quite well [14].

3.5 VISUALIZATION TECHNIQUES

Numerous machine learning tasks include thousands of features. Having so many features leads to a variety of issues, the most significant ones being:

- slows down training to an absurd degree and
- makes it challenging to find a good solution.

This is referred to as regarded as the *curse of dimensionality* and, to put it simply, *dimensionality reduction* is the process of lowering the set of features to those that

are most important. Data visualization is one of the most crucial components of dimensionality reduction. By reducing the dimensionality of the data to two or three, it is feasible to depict the data on a two-dimensional or three-dimensional plot, allowing for the analysis of these patterns in terms of clusters and other concepts. In our study we will use the two such dimensionality reduction algorithms for the visualizations - PCA and UMAP.

PCA (Principal Component Analysis) is one of the most well-known "unsupervised" dimensionality reduction algorithms. This method involves finding the hyperplane that is closest to the data, projecting the data onto that hyperplane while preserving the majority of the data set's variation.

A nonlinear dimensionality reduction technique called UMAP (abbreviation for Uniform Manifold Approximation and Projection) is exceptionally good at showing clusters or groupings of data points together with their relative distances [19].

4 PIPELINE CONSTRUCTION

In this chapter, we recreate an existing research to evaluate the operation of the pipeline and then upgrade it to construct and test a binary classification pipeline for Lyme disease based on open source tools and publicly accessible scientific data found at [10].

4.1 MIKROPML PIPELINE

In this study, we update the well-established open-source machine learning pipeline described in the article *A Framework for Effective Application of Machine Learning to Microbiome-Based Classification Problems* [31], depicted in Figure 3.

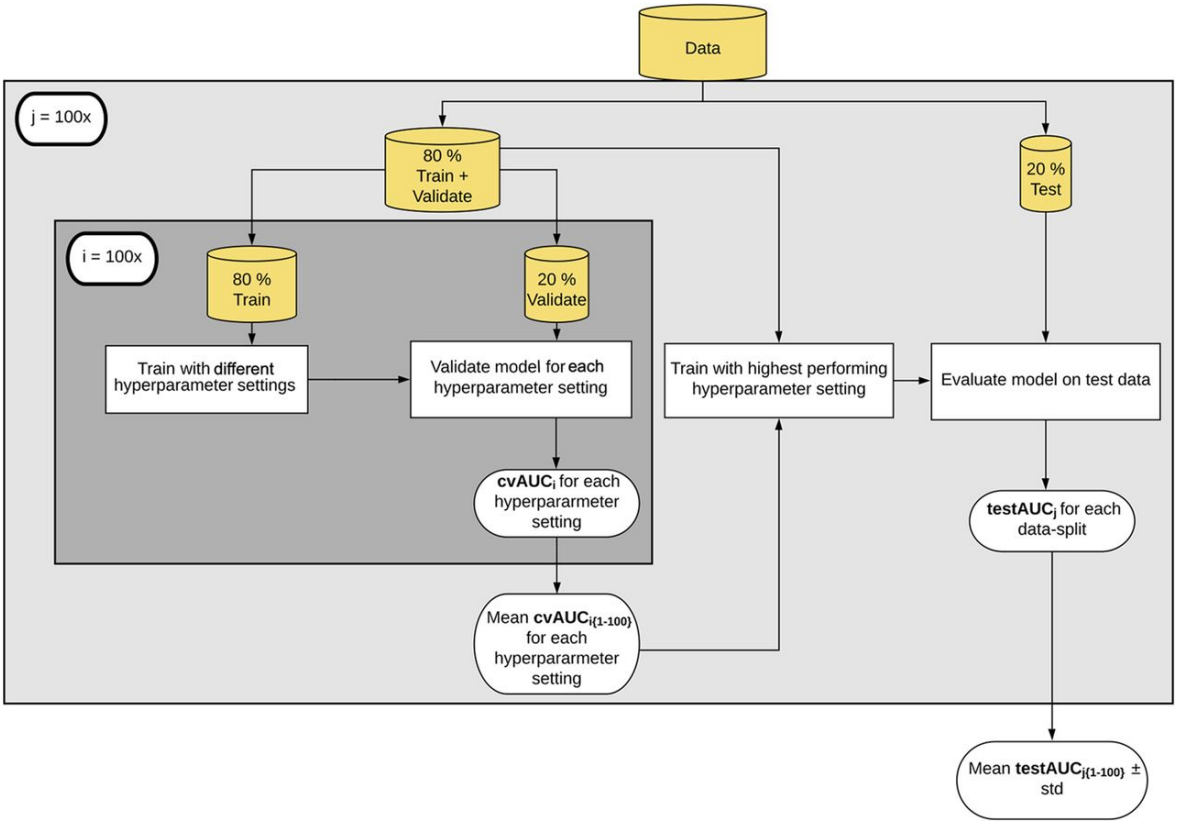

Figure 3: Machine learning pipeline for microbiome-based classification problems [31].

The pipeline works as follows: Initially, the data is divided into training and validation (80%) and held-out test sets (20%). To preserve the overall class distribution, the splits are stratified. We then again split the training and validation data into 80%

train and 20% validation. The optimum hyperparameter setting (parameters that the user must provide or adjust in order to train a model for a particular modeling task) is then chosen using a k-fold cross-validation on the training data (using the area under the receiver operating characteristic curve, the hyperparameter settings that provide the greatest average CV predictive performance are selected) and validated on the test data. For each hyperparameter we get average cv_AUC (cross validated area under the ROC curve - AUROC) and these hyperparameters are subsequently utilized to train the model on the whole 80% of the data. The held-out data set is ultimately used to test the model. Then we get the mean (repeating 100 times) AUC.

The pipeline is open-source and available under the mikropmpl [32] package in R.

A dataframe including a sample or observation for each row represents the input data. All of the other columns are features, whereas one column represents the desired outcome.

In Table 2, we list the methods offered by the mikropml package.

Table 2: Methods available in the mikropml package.

Method	**Classifier**
glmnet	Logistic regression
rpart2	Decision trees
rf	Random forest
svmRadial	SVM radial basis kernel
xgbTree	Gradient boosted trees

4.1.1 Study replication

We use the dataset from Schloss and Sze, accessible at [29], to evaluate the pipeline's functionality.

We examine the two datasets containing OTU abundances and colonoscopy (fecal 16S rRNA sequence data) diagnosis of 490 patients. Using the ML pipeline, we utilize these datasets to evaluate the findings from the study *A Framework for Effective Application of Machine Learning to Microbiome-Based Classification Problems* [31].

We perform a left join and select just the feature columns along with the output column "dx". The feature columns represent the number of each Operational Taxonomic Unit (OTU) in microbiome samples from cancer, adenoma and healthy patients.

After obtaining the dataset with feature columns (OUTs) and an outcome column, we need to assure the binary options of the outcome column, in order to have a binary classification. In our study, the disease outcome values for the patients are cancer, normal and adenoma (a benign tumour). Patients with cancer and adenoma diagnoses

are classified as having a screen relevant neoplasia (SRN), whilst those with a "normal" diagnostic are categorized as healthy.

After processing and preparing the data, we execute the preprocessing function from the mikrompl package, which normalizes the data and deletes features with near-zero variance, and then we apply the pipeline. We train the models using the following parameters: 80% train, validation and 20% test split with 5-fold cross validation repeated 100 times and we set the seed so that the results are reproducible.

In Figure 4 we visualize models performances.

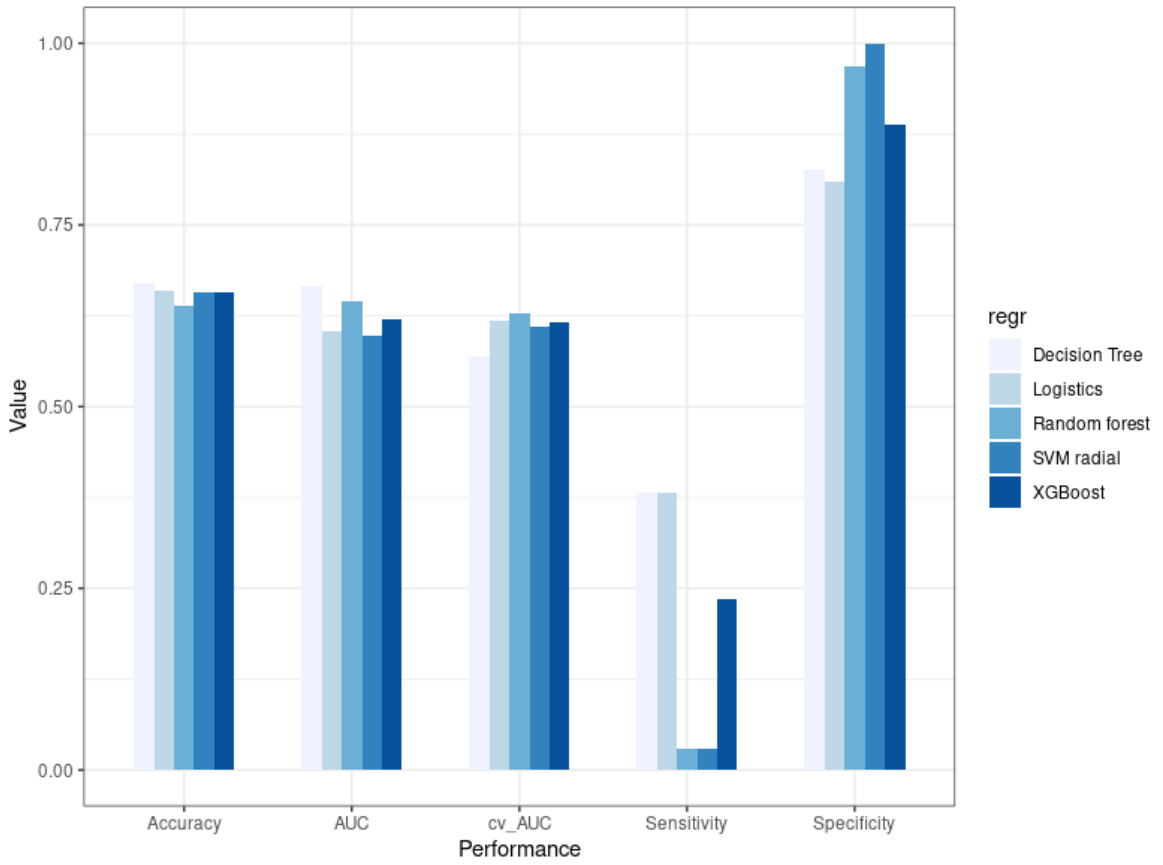

Figure 4: Comparing trained models based on chosen metrics.

In Table 3 we summarize trained model performances. The decision tree model detects SRNs with the highest area under the receiver operating characteristic curve (AUROC) (0.6657), followed by the random forest classifier (0.644). The AUROC scales from 0, where a model's predictions are completely erroneous, to 1.0, when a model can discriminate cases from controls with absolute accuracy. Results are somewhat better than random predictions (AUROC = 0.5), but not by much. Except for decision trees, we can observe that cross-validated AUROC is comparable to test AUROC for every classifier.

The logistic regression model was trained in the shortest amount of time (in sec-

Table 3: Performance metrics of the trained models. Abbreviations: AUROC - area under the receiver operating characteristic, cvAUROC - cross-validated AUROC, Spes - Specificity, Sens - Sensitivity.

Method	**cvAUROC**	**AUROC**	**Accuracy**	**Spec**	**Sens**	**F1**
glmnet	0.618	0.603	0.659	0.809	0.382	0.440
rpart2	0.567	0.665	0.670	0.825	0.382	0.448
rf	0.628	0.644	0.639	0.962	0.029	0.054
svmRadial	0.609	0.598	0.659	1.0	0.029	0.057
xgbTree	0.616	0.619	0.659	0.889	0.235	0.316

onds), whereas the XGBoost classifier required the most time (in several minutes). Our dataset has 490 samples and 268 feature vectors, which is much smaller than the data for our primary research.

We proceed with the description of the data upon which our pipeline is built.

4.2 DATA DESCRIPTION

Our study utilizes two datasets: PeakTable data consisting of 4,851-dimensional feature vectors of intensities (mass/charge, retention time) measured by an Agilent mass spectrometer and processed by the R-package XCMS for the training serum samples and clinical metabolomics dataset consisting of 118 samples with the Lyme disease diagnosis.

The following diagnosis of Lyme disease is indicated:

- Early Disseminated Lyme (EDL)
- Early Localized Lyme (ELL)
- Healthy Control Non-Endemic (HCN)
- Healthy Control Wormser (HCE1)

In the section 4.3 we deal with data refining and processing.

4.3 DATA PREPROCESSING

Following data reading, we use string matching to take the date out of sample names. We transpose the whole dataset since the ML pipeline expects features to be in columns. After transposing, we match the diagnosis from the clinical dataset for each sample and preserve it as the final column result.

4.3.1 Treating missing data

Next in the data refining process is the treatment of missing values. 19,327 missing values are present. In addition, we lack the diagnosis for 15 patients.

We impute missing values using the k-nearest neighbours approach with a k value of 5. This algorithm may be used for continuous, discrete, ordinal, and categorical data, making it especially effective for dealing with all types of missing data. Figure 5 illustrates the UMAP presentation of log transformed and KNN imputed data and Figure 6 depicts a segregated UMAP view for each outcome diagnosis, generated with the *facet_wrap* function from the *ggplot* package in R.

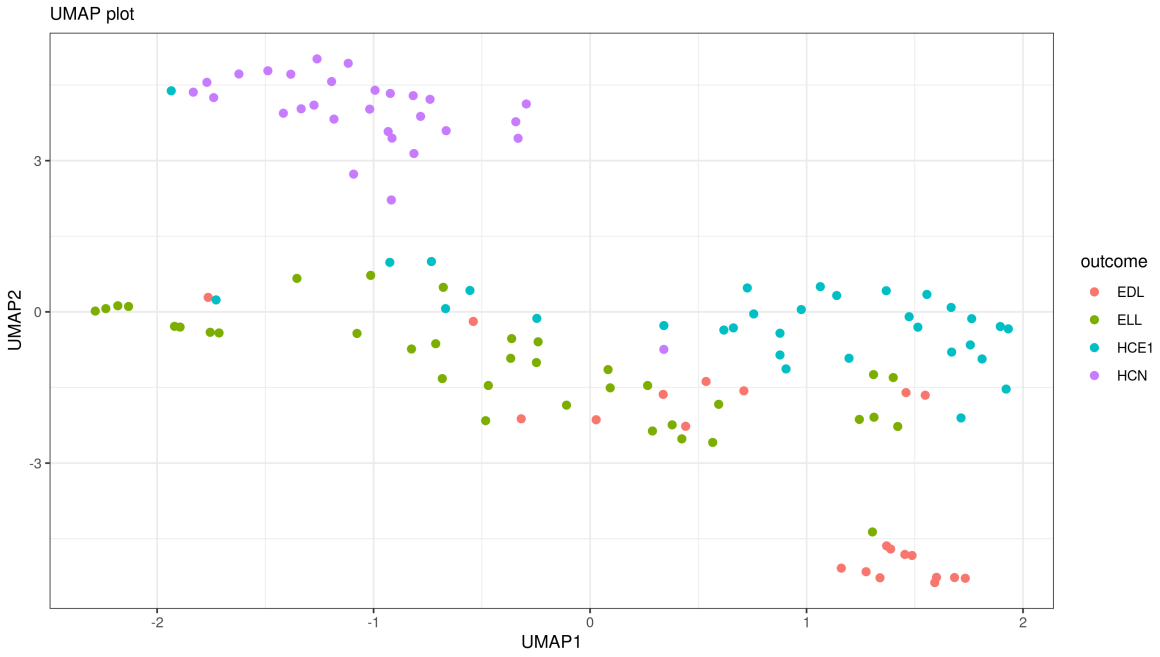

Figure 5: UMAP visualization of log transformed and KNN imputed data.

UMAP (Uniform Manifold Approximation and Projection for Dimension Reduction) can easily handle large datasets and high-dimensional data, and it can be used to see clustering patterns in high-dimensional data.

Finally, in order to construct a binary classification machine learning pipeline, we categorize patients with early disseminated lyme (EDL) and early localized lyme (ELL) as having lyme disease and those with healthy control non-endemic (HCN) and healthy control wormser (HCE1) as being healthy (not having Lyme disease).

We continue with the data visualizations in Section 4.4.

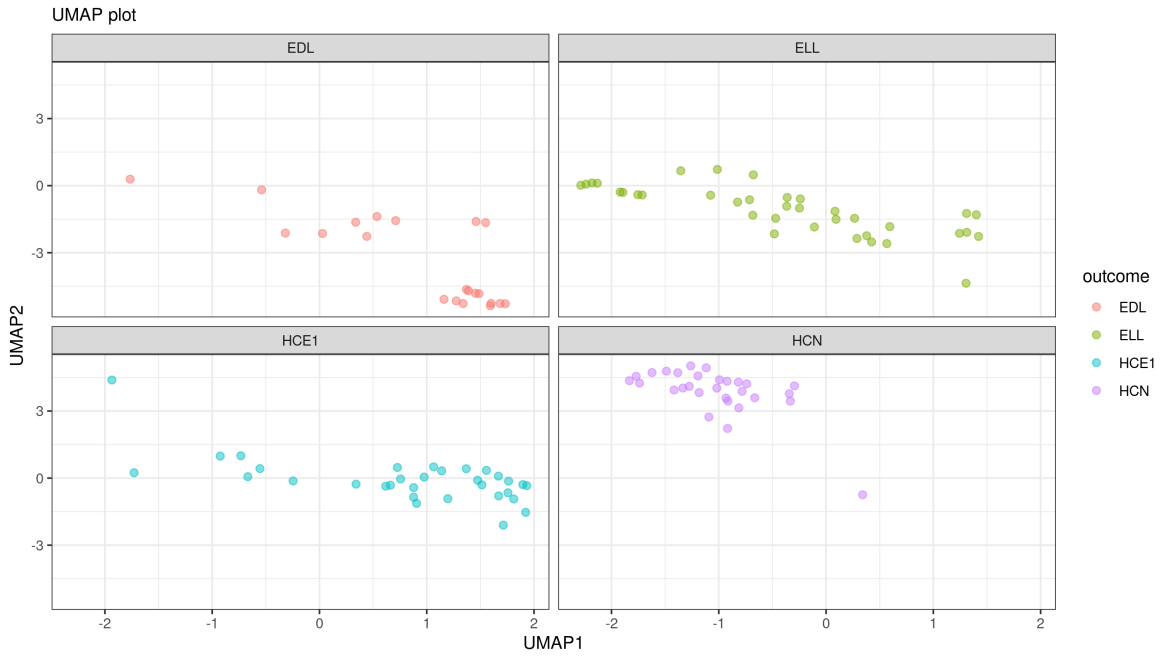

Figure 6: Distinct UMAP view for each outcome.

4.4 DATA VISUALIZATION

PCA (Principal Component Analysis) is an additional visualization approach that we shall use throughout pipeline construction (Figure 7). Several reasons for employing PCA include:

- outliers may be detected more easily,
- PCA helps overcome the problem of overfitting by lowering the number of features,
- it is exceedingly difficult to view and comprehend data with a large number of dimensions. PCA transforms high-dimensional data into low-dimensional data in order to facilitate visualization [23].

The following visualization, shown in Figure 8, enables us to create clusters for the two groups. Cluster 1 represents patients with the disease, while Cluster 2 represents healthy patients.

In Figure 9 we present the probability ellipse plot of the data. A probability ellipse is a contour of constant probability within which a certain proportion of the distribution resides. The correlation between two data points of interest may be determined based on the breadth and direction of probability ellipses. As can be seen, data points in cluster 1 have a positive correlation whereas those in cluster 2 have a weaker correlation.

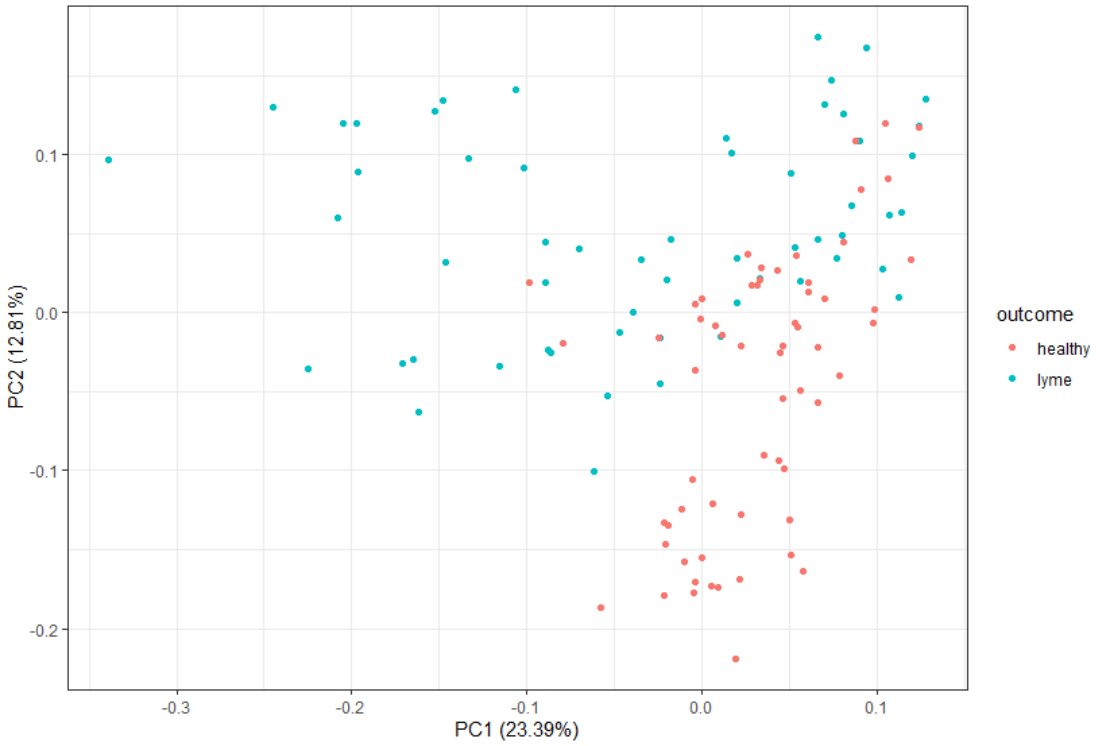

Figure 7: PCA visualization of log transformed and KNN imputed data..

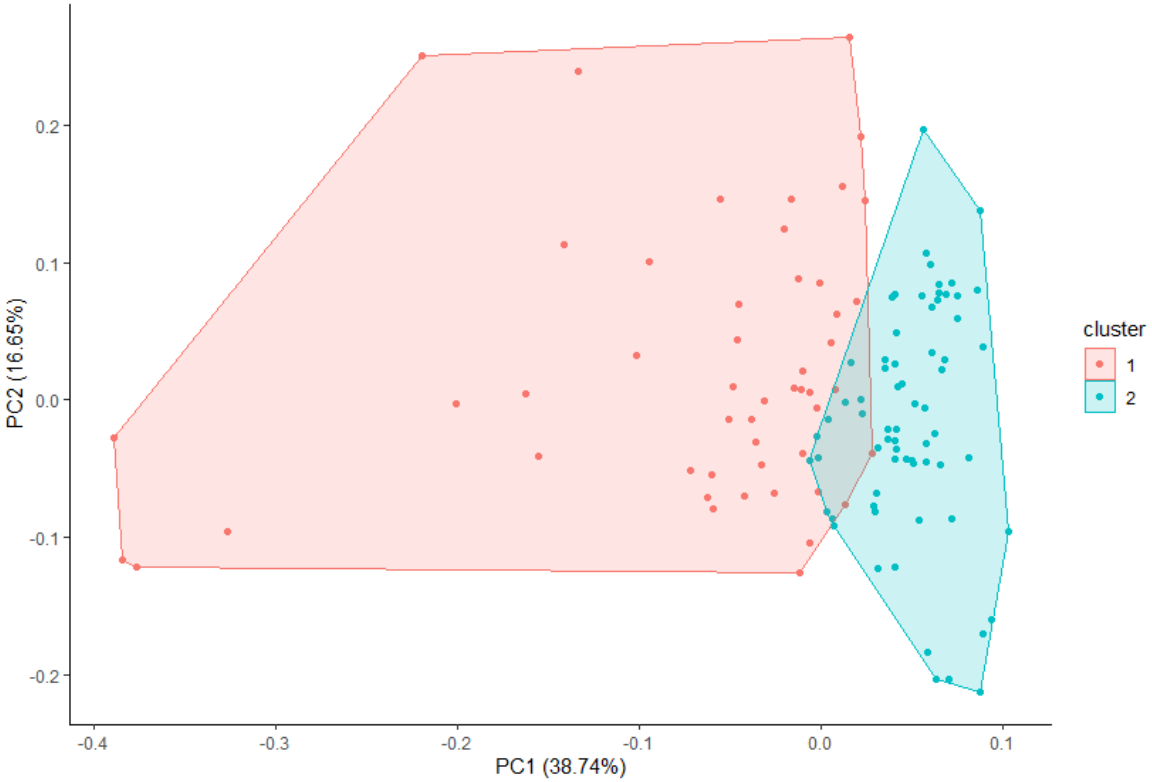

Figure 8: Constructing clusters.

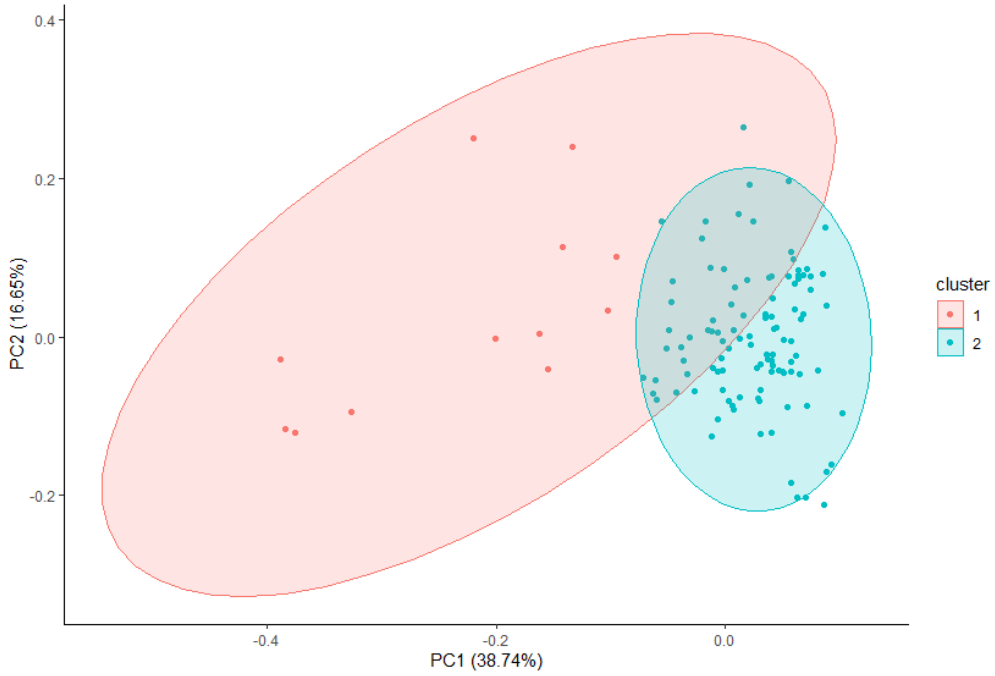

Figure 9: Probability ellipse.

Section 4.5 explains how we choose the best features from the dataset upon which the user may build the model.

4.5 FEATURE SELECTION

Using the metabolomics data analysis toolbox (mt) package, we rate the relevance of features using the Wilcoxon rank sum test. The output consists of a vector of feature ranking scores, a vector of feature ordering from best to worst, a vector of statistics, and a vector of p-values. It is handy if the user wants to select the best features (most important for the classifier), in our case biomarkers, before training the model.

4.6 MODEL SELECTION AND TRAINING

We train and asses the predictive performance of L2-regularized logistic regression, decision trees, random forest, SVM with radial basis kernel and a gradient boosted trees (XGBoost) classifier.

Setting the alpha hyperparameter to 0 indicates L2 regularization. glmnet allows us to do both L1 and L2 regularization. If alpha is changed to 1, L1-regularized logistic regression is performed. Additionally, you may adjust alpha by giving a range of values between 0 and 1. When using a number between 0 and 1, an elastic net is in operation.

The default range for the L2 regularization penalty adjusting hyperparameter lambda

is between 10^{-4} and 10. When examining the 100 repeated cross-validation performance indicators such as AUROC, Accuracy, and cvAUROC for each tested lambda value, we find that some are inapplicable to this data set, as many of them performed equally good (Figure 10).

lambda	logLoss	AUC	prAUC	Accuracy
1e-04	0.2241741	0.9751198	0.8705618	0.9100433
1e-03	0.2241741	0.9751198	0.8705618	0.9100433
1e-02	0.2241741	0.9751198	0.8705618	0.9100433
1e-01	0.2241741	0.9751198	0.8705618	0.9100433
1e+00	0.2241741	0.9751198	0.8705618	0.9100433
1e+01	0.2655810	0.9752040	0.8706280	0.9045140

Figure 10: Range by default of the lambda hyperparameter.

To choose the optimal lambda values, we must execute the machine learning pipeline on numerous data partitions and examine the mean cross-validation performance of each lambda over all modeling trials. Here, we train the model with the newly determined lambda range. We execute it three times, each time with a different seed, resulting in distinct training and testing set partitions. As shown in Figure 11, when we run three data splits, we get a mean maxima around 0.04, which is the optimal lambda value for this dataset. It is essential to give a sufficiently broad range to exhaust our lambda search as we construct the model.

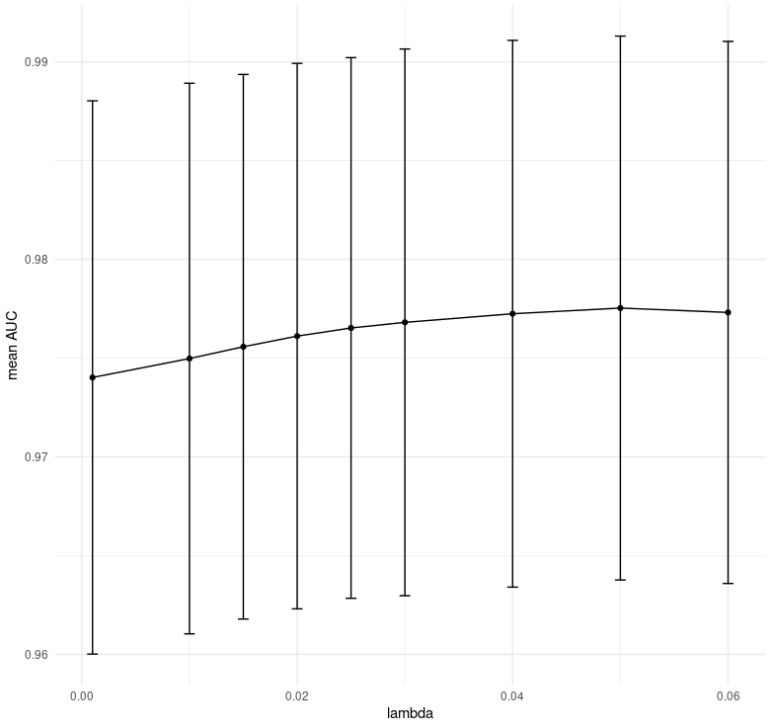

Figure 11: Tuning parameter lambda.

After acquiring the required lambda parameter, we train the model on three random data splits with 80 percent data used for training and evaluating the model, and 20 percent data used for testing, with the five fold cross validation repeated 100 times.

The random baseline is then tested by shuffling the outcome variable (diagnosis). Then, we retrain the model and compare results in Figure 12. We see that the performance of the random baseline classifier is considerably inferior (with AUROC value around 0.5).

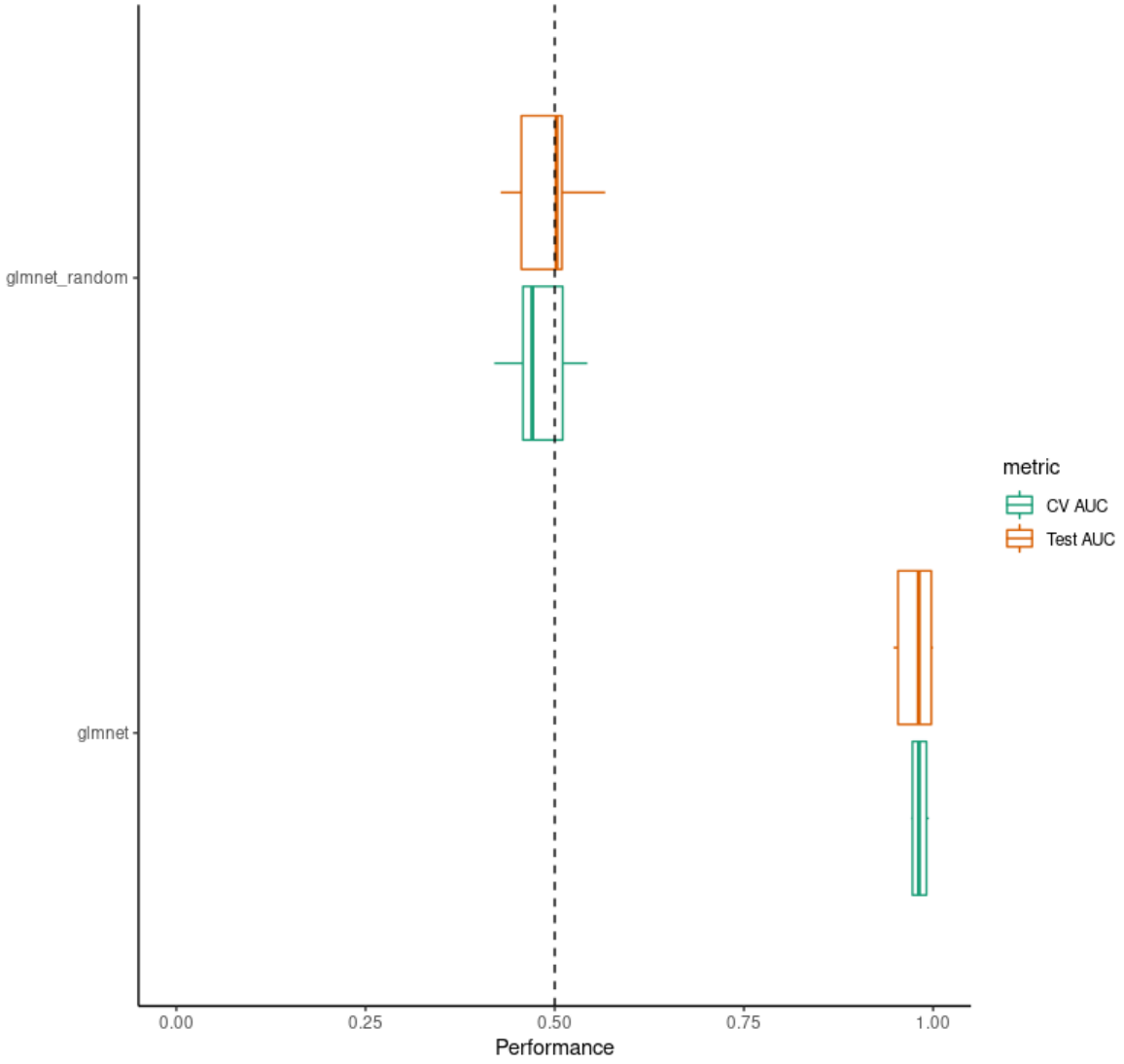

Figure 12: Comparing performance of logistics classifier with random.

Then, we train the decision trees, random forest, SVM with radial basis, and gradient boosted trees classifiers with five fold cross-validated hyperparameters (cross-validation repeated 100 times).

The results are presented in Chapter 5.

The study's code and data are accessible in the git repository available at [11].

4.7 AUTOMATED PIPELINE

In this section, we will walk you through the final process of the pipeline, which will make use of the user-friendly function that can be found in Appendix E. We integrate all preceding processes to produce a repeatable procedure that is simple for users to execute.

Table 4 provides a summary of the pipeline's functioning, including the inputs that are anticipated from the user, the stages that are conducted, and the output that the user receives.

Table 4: Automated ML pipeline for binary classification of Lyme disease status based on metabolite profiles.

AUTOMATED PIPELINE:

Inputs: A dataset with feature vectors being in the columns, outcome variable, training method, parameters.

1. Data organization
2. Missing value imputation using $k - NN$ algorithm
3. Feature selection and model fitting
4. Model evaluation: comparison, visualization.
5. Return dataset following kNN imputation, feature ranking from best to worst, UMAP charts, PCA plot, Cluster plot, Probability ellipse plot, and model performance.

5 RESULTS

In this chapter, we show model performance results based on multiple metrics.

5.1 PREDICTION RESULTS

Table 5 displays the results of five trained models (using *glmnet, rpart2, rf, xgbTree*, and *svmRadial* classifiers; see Table 2) on three random data splits.

Table 5: The performance metrics of the five trained models on three random data splits. cVAUROC stands for cross-validated area under the operating characteristics curve. Acc denotes Accuracy, Spec denotes Specificity, Sens denotes Sensitivity, BalAcc denotes Balanced Accuracy, and Kp denotes the Kappa value.

Method	**cvAUROC**	**AUROC**	**Acc**	**F1**	**Spec**	**Sens**	**BalAcc**	**Kp**
glmnet	0.964	0.976	0.931	0.929	0.933	0.929	0.931	0.862
glmnet	0.986	0.976	0.862	0.867	0.8	0.929	0.864	0.725
glmnet	0.968	0.995	0.966	0.963	1.00	0.929	0.964	0.931
rpart2	0.873	0.895	0.897	0.889	0.933	0.857	0.895	0.792
rpart2	0.918	0.855	0.793	0.75	0.933	0.643	0.788	0.582
rpart2	0.854	0.919	0.897	0.889	0.933	0.857	0.895	0.792
rf	0.976	0.971	0.862	0.875	0.733	1.00	0.867	0.726
rf	0.986	0.962	0.931	0.929	0.933	0.929	0.931	0.862
rf	0.971	1.00	0.966	0.966	0.933	1.00	0.967	0.931
xgbTree	0.979	0.986	0.931	0.929	0.933	0.929	0.931	0.862
xgbTree	0.985	0.962	0.931	0.933	0.867	1.00	0.933	0.863
xgbTree	0.977	0.990	0.897	0.889	0.933	0.857	0.895	0.792
svmRadial	0.991	0.957	0.862	0.846	0.933	0.786	0.860	0.722
svmRadial	0.988	0.938	0.931	0.933	0.867	1.00	0.933	0.863
svmRadial	0.972	1.00	0.966	0.966	0.933	1.00	0.967	0.931

The gradient boosted tree model (XGBoost) performed best at detecting the patients with lyme disease with an area under the receiving operating characteristic curve (AUROC) of 0.985 (interquartile range [IQR], 0.973 to 0.988), followed by L2-regularized logistics regression with an AUROC value of 0.976 and random forest model

with an AUROC value of 0.9714286. Decision trees model performed worst with AUROC of 0.895 (IQR 0.875 to 0.907). The approximate equal values of cross-validated AUROCS and test AUROC indicate that the models are not overfitted.

SVM with a radial basis model had the greatest balanced accuracy, with a score of 0.933 on the test data (IQR 0.896 to 0.95).

In terms of model training duration, XGBoost took the longest to train (about 5 hours), whereas L2-regularized logistics regression took the shortest amount of time (took only 10 minutes to train).

5.2 MODEL COMPARISONS

Figure 13 depicts the training time (in hours) for the five trained ML models. XGBoost had the longest median training time (took around 6 hours to train), whereas L2-regularized logistic regression had the lowest (took around 10 minutes to train).

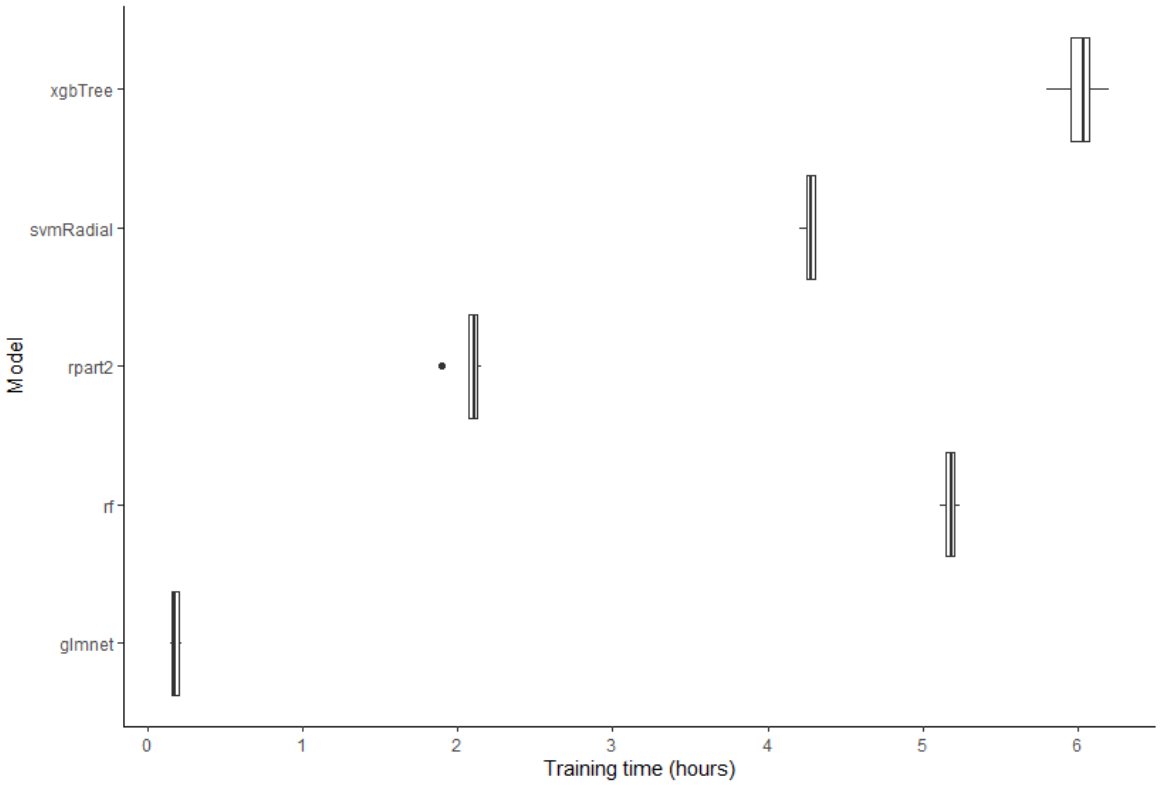

Figure 13: Training time of five ML models.

In Figure 14, the five training models' cross-validated AUROC and test AUROC are compared. Random forest classifier and gradient boosted tree classifier perform somewhat better than other models, but decision tree classifier performed the worst at detecting patients with Lyme disease.

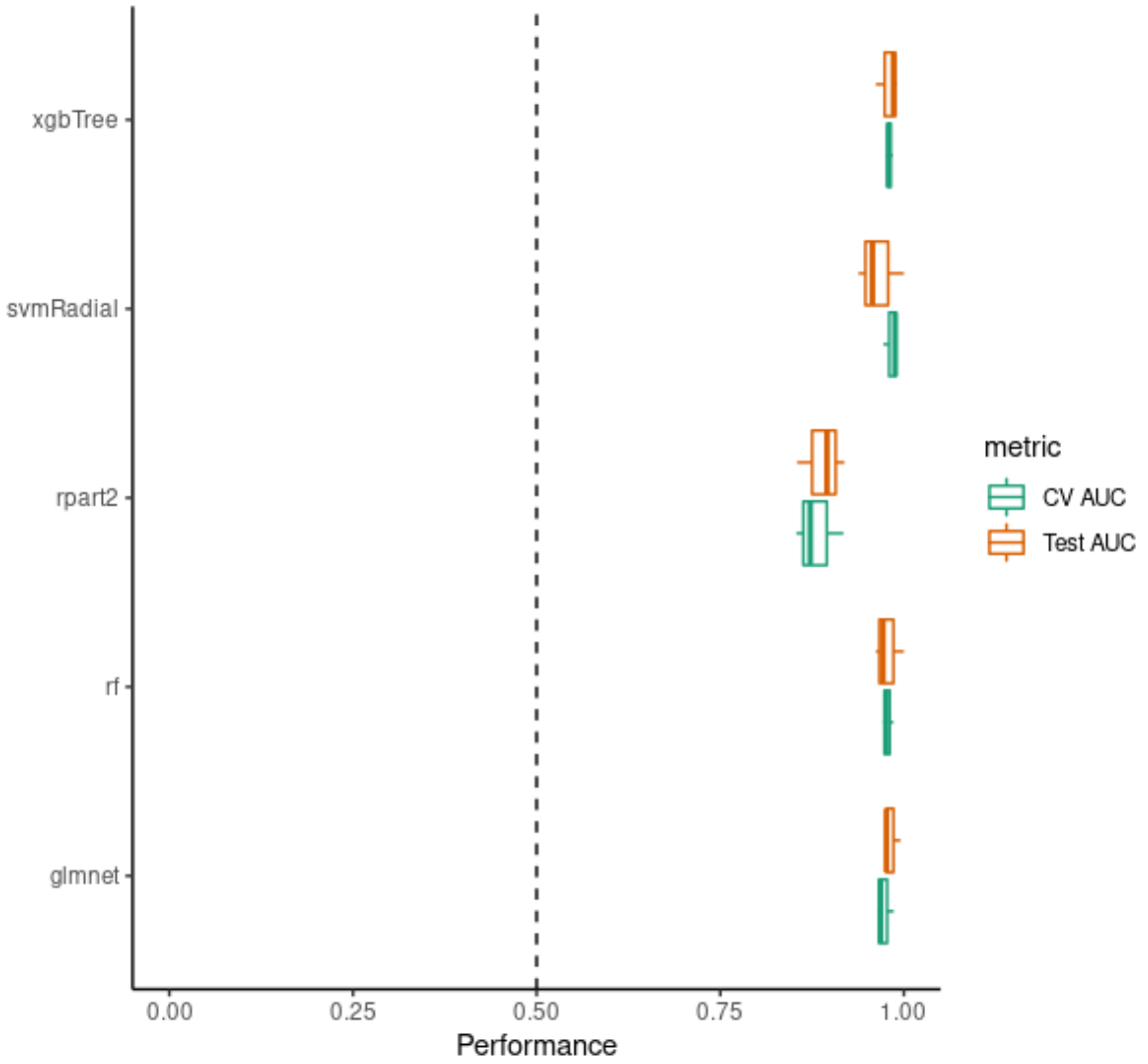

Figure 14: Comparing cross-validated AUC to test AUC of five models (Table 2).

Figure 15 compares the five trained models' accuracy, AUROC, balanced accuracy, cross-validated AUROC, F1, and Kappa value (trained on three random data splits).

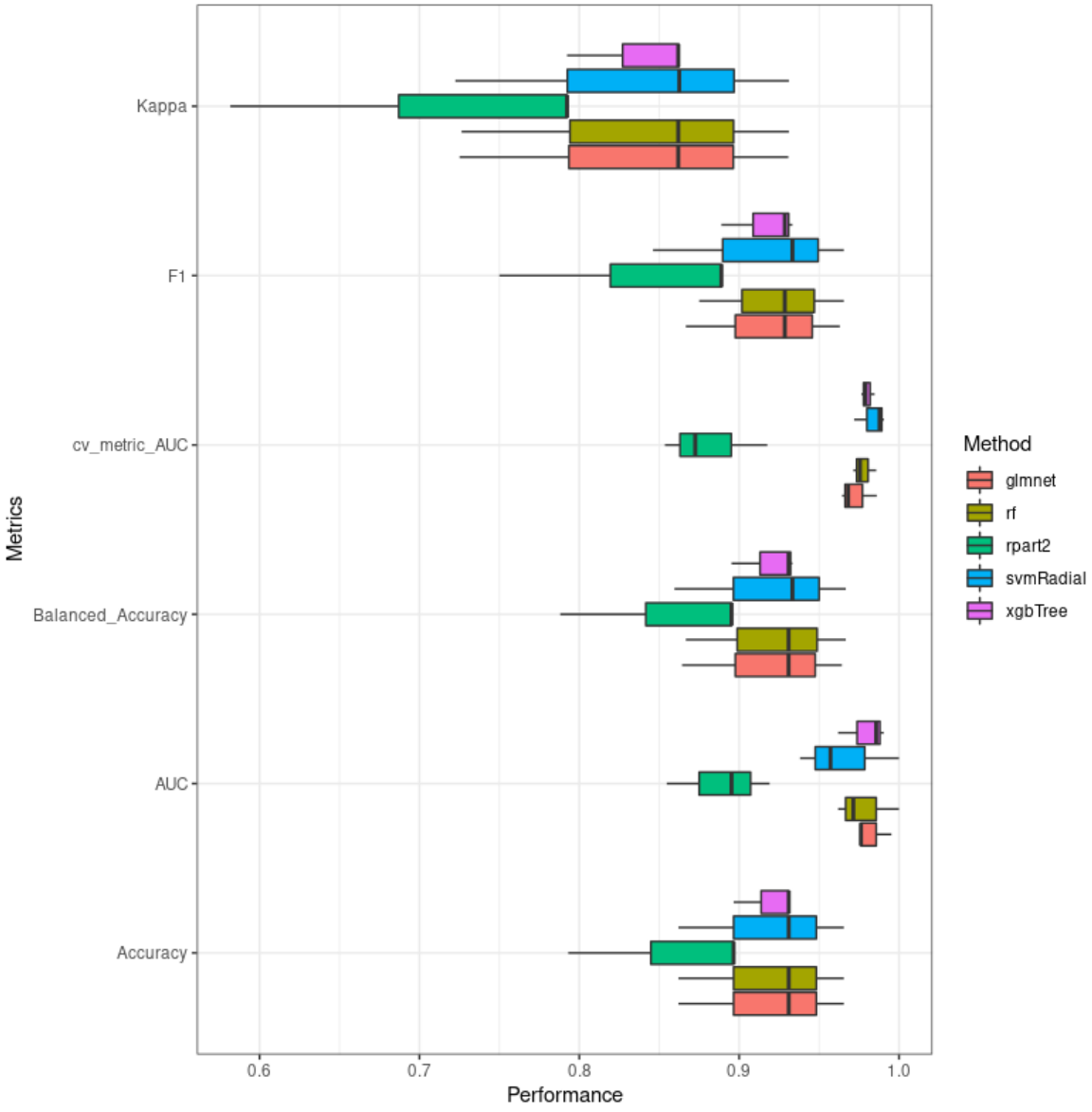

Figure 15: The efficiency of the trained models according to six performance metrics.

6 DISCUSSION

This thesis topic involved the development of a machine learning pipeline for the binary categorization of Lyme disease. We demonstrated a novel workflow enabling users to deploy and train models using high-dimensional (biomedical) data. Our contributions go toward updating and expanding the pipeline that previously mainly involved model training and deployment. We added the phases of data organization, cleaning, and handling of missing values, feature selection, data visualization, model fitting, and a model assessment comprised of graphical comparisons of the model's performance metrics. We developed an open-source implementation of the pipeline, which can be found at [11].

The replicated study from the paper found at [31], in which we examined the functionality of an existing pipeline, yielded the initial findings. We cannot directly compare the results since we used a smaller sample size than in the original study. In our case, the decision tree classifier performed best with an accuracy of 0.67 and an AUROC score of 0.66, whereas the original study indicated that the random forest classifier performed best at recognizing SRNs with an AUROC score of 0.695. (interquartile range [IQR], 0.651 to 0.739).

Our classifiers performed quite well in diagnosing Lyme disease in the main study, with XGBoost model being the best with AUROC of 0.98 (IQR 0.97 to 0.98), balanced accuracy 0.933, F1 statistics 0.933 and Kappa value of 0.863. Decision tree classifier performed worst (among five trained models) at detecting the disease with AUROC value of 0.89 (IQR 0.87 to 0.90), balanced accuracy 0.788, F1 statistics 0.75 and Kappa 0.582. L2-regularized logistic regression performed almost as well as nonlinear models and required considerably less training time (10 minutes on average compared to 4 hours on average for nonlinear models). When selecting a modeling approach, microbiome researchers often fail to provide sufficient justification because they implicitly believe that more complex models are preferable. This has resulted in the utilization of nonlinear models such as random forests and deep neural networks rather than simpler methods such as logistic regression or other linear models. Complex models may frequently capture substantial nonlinear relationships and yield improved predictions, but they can also generate impenetrable black boxes that are difficult to comprehend.

The dataset that lacked sufficient samples exemplified the limitations of our research. A dataset including 118 patient analyses was utilized. Although the data set had more than 4,800 features (attributes), making it high-dimensional, the number

of samples used to train the models was insufficient. With a bigger sample size, our models would be better trained and the study would be of a higher quality and greater precision.

Future research can utilize multi-class classification to determine the precise Lyme disease stage (we encounter four states). This would assist in the detection of Early Disseminated Lyme (EDL), which is difficult to diagnose at present [27].

7 CONCLUSION

In this Master's thesis project, we managed to build a machine learning (ML) pipeline for binary classification of Lyme disease and how machine learning classifiers may be very helpful and efficient in disease diagnosis. We began the thesis by outlining the motivation for our study, then moved on to relevant current work in the field and presented the preliminary theory required to comprehend the classification algorithms, performance measures, visualization methodologies, etc. In chapter 4, we built our pipeline by replicating and analyzing an existing pipeline to create a brand-new procedure with several phases. We showed the trained model performance findings in chapter 5 and discussed them in chapter 6.

We provide a pipeline, that is publicly accessible at [11], for data preprocessing and visualization, machine learning model deployment, and evaluation. We trained two linear and three nonlinear models and compared their performance metrics-based results. Gradient boosted trees (XGBoost) classifier performed best at detecting patients with Lyme disease based on their metabolome data with AUROC score of 0.98.

It is important to highlight that our methodology is reusable and potentially applicable to other high-dimensional (biomedical) data sets. In the future, we want to apply it to real-world case studies.

8 DALJŠI POVZETEK V SLOVENSKEM JEZIKU

Strojno učenje (ML) hitro postaja priljubljena metoda za diagnosticiranje bolezni v raziskavah mikrobioma. Modele ML lahko uporabimo za izboljšanje našega znanja o trenutnih razlikah v strukturi podatkov in za napovedovanje najnovejših podatkov. Raziskovalci so uporabili modele strojnega učenja za prepoznavanje in boljše razumevanje bolezni, vključno s cirozo jeter, rakom debelega črevesa in vnetno črevesno boleznijo [34].

Številne človeške bolezni in okoljski procesi so zdaj razumljeni kot posledica številnih bakterijskih populacij in ne enega samega organizma. Tradicionalne statistične metode so v pomoč pri odkrivanju okoliščin, ko je s procesom povezan en sam organizem. Pristopi strojnega učenja pa omogočajo vključitev celotne strukture mikrobnih skupnosti in ugotavljanje povezav med strukturo skupnosti in bolezenskim stanjem. Če je mogoče skupnosti dosledno razvrstiti, se lahko pristopi ML uporabijo za opredelitev mikrobnih populacij znotraj skupnosti, ki so bistvene za razvrstitev.

V naši študiji uporabljamo metode iz študij mikrobiomov, da bi jih uporabili za razvrščanje metabolomskih profilov. V nadaljevanju besedila bomo navedli nadaljnje podrobnosti o metabolomskem profiliranju in boreliozi. Borelioza je bolezen, ki se hitro širi, vendar je strokovnjaki še vedno ne razumejo dobro. Je najpogostejša bolezen, ki jo prenašajo klopi v vzhodni Evropi [5].

Odvisnost obstoječe diagnostike od serološkega odziva, ki se v zgodnjem obdobju okužbe morda ne oblikuje v celoti in ne more razlikovati med sedanjo in predhodno okužbo, dodatno otežuje diagnosticiranje zgodnje borelioze. Zaradi teh pomanjkljivosti v obstoječi diagnostiki borelioze je treba razmisliti o neimunskih diagnostičnih metodah.

Namen naše študije je razviti ponovljiv pristop za usposabljanje modelov strojnega učenja na visokodimenzionalnih bioloških podatkih ter potek strojnega učenja za boreliozo. V študiji imamo dva nabora podatkov - enega, ki vsebuje značilnosti XCMS za učne vzorce seruma (4 851-dimenzionalni vektor značilnosti intenzivnosti za (masa/naboj, retencijski čas), izmerjene z masnim spektrometrom Agilent in obdelane s paketom R XCMS; in drugi nabor podatkov, ki vsebuje 118 vzorcev, razvrščenih kot : i) zgodnja razširjena borelioza (EDL), ii) zgodnja lokalizirana borelioza (ELL), iii) zdrava kontrola neendemična (HCN) in iv) zdrava kontrola wormser (HCE1).

Cilji in koraki:

- Ponovite študijo iz članka "A framework for effective application of machine learning to microbiome-based classification problems" [31] ter poročajte o rezultatih in primerjavah.

- Uporabite podatke o mikrobiomih za boreliozo iz članka "Biomarker selection and a prospective metabolite-based machine learning diagnostic for lyme disease" [10] (Izbira biomarkerjev in perspektivna na metabolitih temelječa diagnostika s strojnim učenjem za boreliozo) za oblikovanje delovnega postopka ML, ki bo deloval na naslednji način: i) čiščenje in filtriranje podatkov, ii) imputiranje manjkajočih vrednosti z algoritmom KNN, iii) vizualizacija podatkov (UMAP, PCA, grozdi, elipsa verjetnosti) za več izidov in binarne izide nabora podatkov o boreliozi, iv) izbira značilnosti z uporabo Wilcoxonovega testa, v) usposabljanje modelov strojnega učenja (logistična regresija, drevesa odločanja, naključni gozd in xgboost) z navzkrižnim preverjanjem, ki je enako 5, in s 100 ponovitvami navzkrižnega preverjanja, deležem usposabljanja 80 % - 20 % (učni oziroma testni podatki), vi) primerjava učinkovitosti modelov in izbira najboljšega modela glede na izbrane metrike.

- Ustvarjanje metode, ki prevzame nabor podatkov o mikrobiomu, ime stolpca izida in ime metode usposabljanja ML ter pridobi vizualizacije, pripisane manjkajoče vrednosti in uspešnost modela kot izid.

Zbirke podatkov iz prejšnjih raziskav so prosto dostopne v kodni bazi github. Izvesti moramo vse potrebne korake za čiščenje podatkovne zbirke, ujemanje določenih stolpcev in na koncu dobiti eno podatkovno zbirko, ki vsebuje stolpce z značilnostmi in en binarni stolpec z rezultati (stanje bolezni - zdravo ali borelioza).

Ustvarimo 5 različne ML modele in primerjamo rezultate. Kot smo že navedli, bomo modele usposobili in potrdili z uporabo gnezdene navzkrižne validacije. To nam bo omogočilo, da preprečimo pretirano prilagajanje testni množici in da bodo modeli bolj prilagodljivi različnim vrstam podatkov.

Modeli so ovrednoteni z osnovno vrednostjo: napovedovanje binarnega izida stanja bolezni. Za vsak model bomo izračunali specifičnost, občutljivost, natančnost, AUC (kjer drži), F1 in natančnost ter primerjali uspešnost modelov.

Struktura magistrskega dela:

V poglavju 1 opisujemo motivacijo za študijo.

V 2. poglavju obravnavamo sorodna dela in kako jih naše delo dopolnjuje.

V poglavju 3 opisujemo terminologijo, ki je potrebna za razumevanje dela.

V poglavju 4 poustvarjamo raziskavo za razvoj dela obstoječega cevovoda iz paketa *mikropml* ter opisujemo dodatne stopnje in sredstva, s katerimi prilagodimo obstoječi

cevovod za obdelavo in vizualizacijo ter usposabljanje modela na podlagi metabolomskih podatkov.

V poglavju 5 predstavimo in ovrednotimo ugotovitve, v poglavju 6 pa razpravljamo o ugotovitvah.

Ta tema magistrske naloge je vključevala razvoj cevovoda za strojno učenje za binarno kategorizacijo borelioze. Prikazali smo nov potek dela, ki uporabnikom omogoča uvajanje in usposabljanje modelov z uporabo visokorazsežnih (biomedicinskih) podatkov. Koraki postopka cevovoda vključujejo organizacijo podatkov, čiščenje in obravnavo manjkajočih vrednosti, izbiro lastnosti, vizualizacijo podatkov, prilagajanje modela in oceno modela, ki jo sestavljajo grafične primerjave metrik uspešnosti modela. Razvili smo odprtokodno izvajanje cevovoda, ki ga najdete na naslovu [11].

9 REFERENCES

[1] Russ B Altman and Michael Levitt. What is biomedical data science and do we need an annual review of it? *Annual Review of Biomedical Data Science*, 1:i–iii, 2018. *(Cited on page 1.)*

[2] Rémi Bardenet, Mátyás Brendel, Balázs Kégl, and Michele Sebag. Collaborative hyperparameter tuning. In *International conference on machine learning*, pages 199–207. PMLR, 2013. *(Cited on page 12.)*

[3] Anne-Laure Boulesteix and Gerhard Tutz. Identification of interaction patterns and classification with applications to microarray data. *Computational statistics & data analysis*, 50(3):783–802, 2006. *(Cited on page 13.)*

[4] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016. *(Cited on page 10.)*

[5] Christopher J Clarke and John N Haselden. Metabolic profiling as a tool for understanding mechanisms of toxicity. *Toxicologic Pathology*, 36(1):140–147, 2008. *(Cited on page 33.)*

[6] Daniel JB Clarke, Alison W Rebman, Allison Bailey, Megan L Wojciechowicz, Sherry L Jenkins, John E Evangelista, Matteo Danieletto, Jinshui Fan, Mark W Eshoo, Michael R Mosel, et al. Predicting lyme disease from patients' peripheral blood mononuclear cells profiled with rna-sequencing. *Frontiers in immunology*, 12:636289, 2021. *(Cited on page 5.)*

[7] Allan Donner and Neil Klar. The statistical analysis of kappa statistics in multiple samples. *Journal of clinical epidemiology*, 49(9):1053–1058, 1996. *(Cited on page 12.)*

[8] Thomas Edgar and David Manz. *Research methods for cyber security*. Syngress, 2017. *(Cited on page 7.)*

[9] Yan He, J Gregory Caporaso, Xiao-Tao Jiang, Hua-Fang Sheng, Susan M Huse, Jai Ram Rideout, Robert C Edgar, Evguenia Kopylova, William A Walters, Rob Knight, et al. Stability of operational taxonomic units: an important but neglected

property for analyzing microbial diversity. *Microbiome*, 3(1):1–10, 2015. *(Cited on page 6.)*

[10] Eric R Kehoe, Bryna L Fitzgerald, Barbara Graham, M Nurul Islam, Kartikay Sharma, Gary P Wormser, John T Belisle, and Michael J Kirby. Biomarker selection and a prospective metabolite-based machine learning diagnostic for lyme disease. *Scientific reports*, 12(1):1–14, 2022. *(Cited on pages 1, 4, 15 in 34.)*

[11] Dorde Klisura. Pipeline dorde. `https://gitlab.com/openresearchlabs/pipeline-dorde`, 2022. *(Cited on pages 24, 30, 32 in 35.)*

[12] Sotiris B Kotsiantis. Decision trees: a recent overview. *Artificial Intelligence Review*, 39(4):261–283, 2013. *(Cited on page 8.)*

[13] Roshan Kumari and Saurabh Kr Srivastava. Machine learning: A review on binary classification. *International Journal of Computer Applications*, 160(7), 2017. *(Cited on page 7.)*

[14] Jae Won Lee, Jung Bok Lee, Mira Park, and Seuck Heun Song. An extensive comparison of recent classification tools applied to microarray data. *Computational Statistics & Data Analysis*, 48(4):869–885, 2005. *(Cited on page 13.)*

[15] Zachary Chase Lipton, Charles Elkan, and Balakrishnan Narayanaswamy. Thresholding classifiers to maximize f1 score. *arXiv preprint arXiv:1402.1892*, 2014. *(Cited on page 12.)*

[16] Huan Liu and Lei Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on knowledge and data engineering*, 17(4):491–502, 2005. *(Cited on page 13.)*

[17] Rui Liu, Xiangdong Wang, Kazuyuki Aihara, and Luonan Chen. Early diagnosis of complex diseases by molecular biomarkers, network biomarkers, and dynamical network biomarkers. *Medicinal research reviews*, 34(3):455–478, 2014. *(Cited on page 1.)*

[18] Jake Luo, Min Wu, Deepika Gopukumar, and Yiqing Zhao. Big data application in biomedical research and health care: a literature review. *Biomedical informatics insights*, 8:BII–S31559, 2016. *(Cited on page 1.)*

[19] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. *(Cited on page 14.)*

[20] Yazdani Mehrdad, Bryn C Taylor, Justine W Debelius, Weizhong Li, Rob Knight, and Larry Smarr. Using machine learning to identify major shifts in human gut microbiome protein family abundance in disease. In *2016 IEEE international conference on big data (big data)*, pages 1272–1280. IEEE, 2016. *(Cited on page 2.)*

[21] Kevin M Mendez, Stacey N Reinke, and David I Broadhurst. A comparative evaluation of the generalised predictive ability of eight machine learning algorithms across ten clinical metabolomics data sets for binary classification. *Metabolomics*, 15(12):1–15, 2019. *(Cited on page 9.)*

[22] Tim Menzies, Leandro Minku, and Fayola Peters. The art and science of analyzing software data; quantitative methods. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 2, pages 959–960. IEEE, 2015. *(Cited on page 8.)*

[23] Tomasz Niedoba. Multi-parameter data visualization by means of principal component analysis (pca) in qualitative evaluation of various coal types. *physicochemical problems of Mineral processing*, 50, 2014. *(Cited on page 20.)*

[24] Margaret Sullivan Pepe. *The statistical evaluation of medical tests for classification and prediction*. Oxford University Press, USA, 2003. *(Cited on page 11.)*

[25] Patrick Ray, Yannick Le Manach, Bruno Riou, Tim T Houle, and David S Warner. Statistical evaluation of a biomarker. *The Journal of the American Society of Anesthesiologists*, 112(4):1023–1040, 2010. *(Cited on page 12.)*

[26] Karen Segers, Sven Declerck, Debby Mangelings, Yvan Vander Heyden, and Ann Van Eeckhaut. Analytical techniques for metabolomic studies: a review. *Bioanalysis*, 11(24):2297–2318, 2019. *(Cited on page 1.)*

[27] Eugene D Shapiro. Borrelia burgdorferi (lyme disease). *Pediatrics in review*, 35(12):500–509, 2014. *(Cited on page 31.)*

[28] Shu-juan, Qiao Wang, Yu, Rong-cun Jiang, and Hong-yang Wang. A feature selection approach based on a similarity measure for software defect prediction. *Frontiers of Information Technology & Electronic Engineering*, 18(11):1744–1753, 2017. *(Cited on page 13.)*

[29] Marc A Sze and Patrick D Schloss. Leveraging existing 16s rrna gene surveys to identify reproducible biomarkers in individuals with colorectal tumors. *MBio*, 9(3):e00630–18, 2018. *(Cited on page 16.)*

[30] Kashvi Taunk, Sanjukta De, Srishti Verma, and Aleena Swetapadma. A brief review of nearest neighbor algorithm for learning and classification. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pages 1255–1260. IEEE, 2019. *(Cited on page 11.)*

[31] Begüm D Topçuoğlu, Nicholas A Lesniak, Mack T Ruffin IV, Jenna Wiens, and Patrick D Schloss. A framework for effective application of machine learning to microbiome-based classification problems. *MBio*, 11(3):e00434–20, 2020. *(Cited on pages VII, 1, 2, 4, 12, 15, 16, 30 in 34.)*

[32] Begüm D. Topçuoğlu, Zena Lapp, Kelly L. Sovacool, Evan Snitkin, Jenna Wiens, and Patrick D. Schloss. mikropml: User-friendly r package for supervised machine learning pipelines. *Journal of Open Source Software*, 6(61):3073, May 2021. *(Cited on page 16.)*

[33] Vicken Totten, Erin L Simon, Mohammad Jalili, and Hendry R Sawe. Acquiring data in medical research: A research primer for low-and middle-income countries. *African Journal of Emergency Medicine*, 10:S135–S139, 2020. *(Cited on page 1.)*

[34] Luke K Ursell, Jessica L Metcalf, Laura Wegener Parfrey, and Rob Knight. Defining the human microbiome. *Nutrition Reviews*, $70(\text{suppl}_1) : S38 - -S44, 082012.(Citedonpages\ 6in\ 33.)$

Appendices

APPENDIX A Study replication

```r
df1 <- read.table(file = 'data2.shared', sep = '\t', header = TRUE)
df2_2 <- read.table(file = 'metadata.tsv', sep = '\t', header = TRUE)

# left join
df2<- df2_2 %>%
  select("sample","dx")

dff<-left_join(df1, df2, by = c("Group" = "sample"))

# dropping the unnecessary columns
df <- dff %>%
  select(-c("X", "label","Group","numOtus"))

df$dx <- case_when(df$dx=="normal" ~ "healthy",
                   df$dx == "adenoma" ~ "SRN",
                   df$dx == "cancer" ~ "SRN")

# preprocessing function
df <- preprocess_data(dataset = df,
                      outcome_colname = 'dx')
df <- df$dat_transformed

# Running the pipeline - function created
pipe <- function(df, method){
  run_ml(df,
         method,
         kfold = 5,
         cv_times = 100,
         training_frac = 0.8,
         seed = 2022)
}
```

APPENDIX B Preprocessing data

```r
df<- read.csv("PeakTable_data.csv")
df1<- read.csv("HCvsEL_clinicaldata.csv")

# deleting from the column names dates (after the dot)
colnames(df) <- gsub("\\..*","",colnames(df))

#transposing
dff <- transpose(df, keep.names = "rn")

# adding an output column
cols <- df1[,c("My.Number","Sample.Type")]

# merge by matching the two data frames to create an output column
dff$outcome_long <- cols$Sample.Type[match(dff$rn, cols$My.Number)]

# delete first column with names
dff <- subset(dff, select = -c(rn))

dff$outcome_long <- case_when(dff$outcome_long=="Early Disseminated Lyme" ~ "EDL",
                              dff$outcome_long == "Early Localized Lyme" ~ "ELL",
                              dff$outcome_long == "Healthy Non-enemic control CO" ~ "HCN",
                              dff$outcome_long == "Heatlhy Controls - Endemic Dr. Wormser" ~ "HCE1")

dff$outcome_long #we have around 15 missing outcomes
dff$outcome_long <- as.factor(dff$outcome_long)

# KNN imputation
dff[dff == 0] <- NA
sum(is.na(dff)) # 19.327 missing values
df.feat <- dff[1:4851] # we take everything but the outcome column
res<- kNN(df.feat, k = 5) # perform kNN with k=5 (default)
col.outcome <- dff[4852]
res$outcome_long <- col.outcome #adding outcome column
dataset <- res[grep("_imp", colnames(res), invert = TRUE)] #deleting unnecessary produced columns _imp
```

```r
dataset <- kNN(dataset, variable = "outcome_long") # kNN imputation for the missing values in outcome column
fin<- dataset[1:4852]
sum(is.na(dataset)) # 0

# binary outcome
dataset$outcome_long <- case_when(fin$outcome_long == "EDL" ~ "lyme",
                                  fin$outcome_long == "ELL" ~ "lyme",
                                  fin$outcome_long == "HCN" ~ "healthy",
                                  fin$outcome_long == "HCE1" ~ "healthy")
```

APPENDIX C Data visualizion

```
# R E L E V A N T   F I G U R E S

# plotting function
df <- read.csv(file="dataset_after_KNN.csv")
df.cluster <- df %>%
  select(-c(X, outcome))

df<- df %>%
  mutate(ID = row_number()) %>%
  select(-c(X))

# categorical
df.labels <- df[, c("outcome", "ID")]

# apply Uniform Manifold Approximation and Projection
set.seed(142)

df.umap <- df %>%
  select(where(is.numeric)) %>%
  column_to_rownames("ID") %>%
  log() %>%
  scale() %>%
  umap()

umap_df <- df.umap$layout %>%
  as.data.frame()%>%
  rename(UMAP1="V1",
         UMAP2="V2") %>%
  mutate(ID=row_number())%>%
  inner_join(df.labels, by="ID")


umap_df %>%
  ggplot(aes(x = UMAP1,
             y = UMAP2,
             color = outcome))+
  geom_point(size = 2)+
  labs(x = "UMAP1",
       y = "UMAP2",
```

```
       subtitle = "UMAP plot")+
  theme_bw()
ggsave("UMAP_plot.png")

umap_df %>%
  ggplot(aes(x = UMAP1,
             y = UMAP2,
             color = outcome)) +
  geom_point(size=2, alpha=0.5)+
  facet_wrap(~outcome)+
  labs(x = "UMAP1",
       y = "UMAP2",
       subtitle="UMAP plot")+
  theme_bw()
ggsave("UMAP_plot_facet_wrap.png")


#### PCA VISUALIZATION

# reading binary options - healthy and lyme (diagnosed)
df.binary <- read.csv(file="binary_outcome.csv")
df.binary <- subset(df.binary, select = -c(X))

pca.res <- df.binary %>%
  select(-c("outcome"))

pca_res <- prcomp(pca.res, scale. = TRUE)

autoplot(pca_res, data = df.binary, colour = 'outcome')

# clusters
autoplot(fanny(pca.res, 2), frame = TRUE)

# probability ellipse
autoplot(pam(pca.res, 2), frame = TRUE, frame.type = 'norm')
```

APPENDIX D Feature selection

```r
library("mt")
library(tidyverse)
library(mikropml)

data <- read.csv("binary_outcome.csv") %>%
  select(-X)

x <- data %>%
  select(-outcome)
y <- data[,"outcome"]

# log transform
x <- preproc(x, method="log10")

# Wilcoxon
result <- fs.wilcox(x, y)

# function for displaying the best x number of features
disp_features <- function(result, number){
  features<- result$fs.order[number]

  return(features)
}
#display best 45 features
disp_features(result,1:45)
```

APPENDIX E Trained models

```
1
2  # we use the multicore plan to split the work across 2 cores
3  doFuture::registerDoFuture()
4  future::plan(future::multicore, workers = 2)
5
6  # applying the pipeline
7  df.binary <- read.csv(file = "binary_outcome.csv")
8  df.binary <- df.binary %>%
9    select(-X)
10
11 # logistics model with 3 random splits
12 result_glmnet <- future.apply::future_lapply(seq(2020, 2022), function
       (seed) {
13   run_ml(df.binary,
14          'glmnet',
15          outcome_colname = "outcome",
16          training_frac = 0.75,
17          seed = seed)
18 }, future.seed = TRUE)
19 saveRDS(result_glmnet, file="result_glmnet.Rds")
20
21 #function for displaying metrics
22 extr <- function(model) {
23   a<- model[[1]]$performance %>%
24     select(cv_metric_AUC, AUC, Accuracy, F1, Specificity, Sensitivity,
       Balanced_Accuracy, Kappa, method)
25   b<- model[[2]]$performance %>%
26     select(cv_metric_AUC, AUC, Accuracy, F1, Specificity, Sensitivity,
       Balanced_Accuracy, Kappa, method)
27   c<- model[[3]]$performance %>%
28     select(cv_metric_AUC, AUC, Accuracy, F1, Specificity, Sensitivity,
       Balanced_Accuracy, Kappa, method)
29
30   df_list <- list(a,b,c)
31   df<- df_list %>%
32     reduce(full_join)
33
34   return(df)
35 }
```

```r
# displaying the main characteristics
extr(result_glmnet)

# checking the random baseline result for logistics
random_glmnet <- transform(df.binary, outcome = sample(outcome))
result_glmnet_random <- run_ml(random_glmnet,
                               'glmnet',
                               outcome_colname = 'outcome',
                               kfold = 5,
                               cv_times = 100,
                               training_frac = 0.8,
                               seed = 2019)
# using the created function for other methods
pipe <- function(df, method){
  run_ml(df,
         method,
         kfold = 5,
         cv_times = 100,
         training_frac = 0.8,
         seed = 2022)
}
```

APPENDIX F The method

```
pipeline <- function(dataset, outcome_variable, method, cv_times){
  # KNN imputation
  dataset<- kNN(dataset, k = 5)
  dataset <- knn_res[grep("_imp", colnames(knn_res), invert = TRUE)]
  # features selection using Wilcox test
  x <- dataset %>%
    select(-c(outcome_variable))
  y <- dataset[,outcome_variable]
  # log transform
  x <- preproc(x, method="log10")
  wilcox <- fs.wilcox(x, y)
  wilcox_order <- wilcox$fs.order

  # VISUALIZATIONS
  df<- dataset %>%
    mutate(ID = row_number())
  # categorical
  df.labels <- df[, c(outcome_variable, "ID")]
  # apply Uniform Manifold Approximation and Projection
  set.seed(142)
  df.umap <- df %>%
    select(where(is.numeric)) %>%
    column_to_rownames("ID") %>%
    log() %>%
    scale() %>%
    umap()

  umap_df <- df.umap$layout %>%
    as.data.frame()%>%
    rename(UMAP1="V1",
           UMAP2="V2") %>%
    mutate(ID=row_number())%>%
    inner_join(df.labels, by="ID")

  pic<- umap_df %>%
    ggplot(aes(x = UMAP1,
               y = UMAP2,
               color = outcome))+
    geom_point(size = 2)+
```

```
    labs(x = "UMAP1",
         y = "UMAP2",
         subtitle = "UMAP plot")+
    theme_bw()
  #facet wrap plot
  pic2<- umap_df %>%
    ggplot(aes(x = UMAP1,
               y = UMAP2,
               color = outcome)) +
    geom_point(size=2, alpha=0.5)+
    facet_wrap(~outcome)+
    labs(x = "UMAP1",
         y = "UMAP2",
         subtitle="UMAP plot")+
    theme_bw()
  # PCA figure
  features <- dataset %>%
    select(-c(outcome_variable))
  pca_res <- prcomp(features, scale. = TRUE)
  pic3<- autoplot(pca_res, data = dataset, colour = 'outcome')
  #clusters
  pic4<- autoplot(fanny(features, 2), frame = TRUE)
  #probability ellipse
  pic5<- autoplot(pam(features, 2), frame = TRUE, frame.type = 'norm')

  # PIPELINE
  model_result <- run_ml(dataset,
                         method,
                         outcome_colname = outcome_variable,
                         kfold = 5,
                         cv_times = cv_times,
                         training_frac = 0.8,
                         seed = 2022)
  model<-saveRDS(model_result, file="model_result.Rds")

  return(list("Dataset after KNN" = dataset,
              "Feature selection" = wilcox_order,
              "UMAP plot"=pic,
              "UMAP facet wrap"=pic2,
              "PCA plot" = pic3,
              "Clusters" = pic4,
              "Probability ellipse" = pic5,
              "Model performance" = model_result))
}
```