Jennifer Neumaier

Institute of Genomics, University of Tartu

Master Thesis

*Benchmarking machine learning performances with compositional* *data*

Table of Contents

[Introduction 2](#_Toc96333815)

[Goal 2](#_Toc96333816)

[Background 2](#_Toc96333817)

[Microbiome Data is Compositional 2](#_Toc96333818)

[Current Solutions for Compositional Data in Statistical Analysis 4](#_Toc96333819)

[Log-Ratio Transformations make compositional data analysis a lot easier 6](#_Toc96333820)

[Compositional Data in Machine learning 8](#_Toc96333821)

[Methodology 11](#_Toc96333822)

[Pipeline 11](#_Toc96333823)

[Data Set 12](#_Toc96333824)

[Pre-Processing 12](#_Toc96333825)

[Standard Microbiome Approaches 13](#_Toc96333826)

[Transformations 13](#_Toc96333827)

[Machine Learning Models 14](#_Toc96333828)

[Publication bibliography 16](#_Toc96333829)

# Introduction

## Goal

Machine learning in microbiome studies is widely used and the interest is growing. However, there is no universal understanding of the algorithmic approaches that can best utilize the information present in the microbiome data. Thus, this is an interesting and widely discussed topic that can have a great impact on the potential applications leveraging microbiome data. A key topic in microbiome research is the sample space of the input data. The sequencing data appears as count data, but, only relative abundance of the microbial features can be observed, commonly called “compositional data”. Thus, transforming the read counts to relative abundances is usually the first step and machine learning methods are usually applied on relative abundances. However, relative abundances raise several limitations, which can have an impact on the performance of the prediction models. Therefore, log-ratio transformations are a proposition made by several studies now, however their impact on machine learning performances has never been tested in large-scale studies. The goal of this benchmarking project is to rectify that and conduct several machine learning models under several log-ratio transformations in comparison to standard microbiome approaches like *selbal* or *CoDaCoRe*. This way it will become clearer if a scientist should make the effort in learning about machine learning methods, when automated algorithms perform well enough and no heavy prior machine learning knowledge is necessary.

## Background

### Microbiome Data is Compositional

Microbiome data is achieved by taking a population of (total or fractionated) RNA, converting them to a library of cDNA fragments, optionally amplifying the fragments, and then sequencing those fragments in a ‘high-throughput manner’ (Quinn et al. 2018). This methodology is known as next generation sequencing. The result of NGS is a virtual library of many short sequence fragments that are converted to a numeric dataset through alignment (most often to a previously established reference genome or transcriptome) and quantification (Griffith et al. 2015). Thus, sequence abundances are not absolute abundances because the total number of sequences measured by sequencing machines ultimately depends on the chemistry of the assay, not the input material (Quinn et al. 2018).

This leads to the illusion that sequencing data appears as count data, but in reality, only relative abundance of the microbial features can be observed (Gloor et al. 2017), since the abundances for each sample are constrained by an arbitrary total sum (Quinn et al. 2018). Thus, the individual values of the observed counts are irrelevant (Quinn et al. 2018). The following figure displays this problematic visually:

Diagram, schematic

Description automatically generated

Figure : Characteristics of compositional data

Taken from (Gloor et al. 2017). (A) After sequencing the data observed from a bacterial population cannot inform on the absolute abundance of molecules. The number of counts in a high throughput sequencing dataset reflect the proportion of counts per feature per sample, multiplied by the sequencing depth. Therefore, only relative abundances are available. The consequences are portrayed in (B). The bar plots show the difference between the count of molecules and the proportion of molecules for two features, A (red) and B (gray) in three samples. The top bar graphs show the total counts for three samples, and the height of the color illustrates the total count of the feature. When the three samples are sequenced, we lose the absolute count information and only have relative abundances, proportions, or “normalized counts” as shown in the bottom bar graph. Note that features A and B in samples 2 and 3 appear with the same relative abundances, even though the counts in the environment are different.

Thus, relative abundance data - and microbiome data - are mathematically considered “compositional data”, with its own mathematical theory and properties. Compositional data lives in the positive simplex space and not in real Euclidean space, which is assumed by commonly used data analysis (Quinn et al. 2018). Thus, compositional data is very awkward to handle due to its scarcity of meaningful definitions of independence (Aitchison 1982). Luckily, the relative abundances of microbial features still carry meaning. Several propositions have been made in the last few years to acknowledge compositional data in statistical analysis (Aitchison 1982) and increase its interpretability.

### ~~Current Solutions for Compositional Data in Statistical Analysis~~

~~Gloor et al. (2017) pointed out the importance of an alternative tool kit for compositional data. Table

Description automatically generatedOne of the first analysis steps in traditional analysis is the calculation of a distance or dissimilarity (DD) matrix from the data after rarefaction or count normalization. Figure 2 shows a standard microbiome toolkit and its alternatives for compositional data.~~

Figure 2: Standard microbiome analysis tool kit and compositional replacements

Figure was taken from Gloor et al. (2017). It depicts a simplified standard microbiome computational workflow.

~~Common in microbiome analysis are UniFrac, Bray-Curtis and Jensen-Shannon divergence. Inherently, DD methods are sensitive to the total read depth of a sample. Thus, they do not account for the compositional nature of the data and since they largely discriminate between samples based on the most relative abundant features, instead the most variable, this can lead to drastic changes when different features are included or excluded in the dataset (Gloor et al. 2017). Therefore, Aitchison proposed the so called “Aitchison distance”. It is more stable to sub setting and aggregating of the data, and being a true linear distance (Gloor et al. 2017).~~

~~The major uses for DD matrices are ordination and clustering (Gloor et al. 2017). Using the Aitchison distance solves the problem of sensitivity in ordination. However, it has been shown that differential abundance tools are sensitive to sparsity and correlation is not reliable or a reproducible indicator when dealing with compositional data (Gloor et al. 2017). The replacement for β-diversity exploration of microbiome data is the variance-based compositional principal component (PCA) biplot (Gloor et al. 2017). It has the advantage that exploratory data analysis is not driven by the presence-absence relationships in the data nor by excessive sparsity. Also, it does not rely on an underlying phylogenetic tree.~~

~~Severe problems with correlation in compositional data were noted early (Gloor et al. 2017), as compositional data have a negative correlation bias and a different correlation structure than the underlying count data (Gloor et al. 2017). This is a severe problem in compositional data analysis. Possible approaches to analyze correlation are SPARCC and SpiecEasi, which both assume a sparse data matrix, as well as two metrices which require a non-sparse matrix (Gloor et al. 2017). Finally, differential abundance of OTUs in compositional data can be analyzed by ANCOM, which performs statistical tests on point estimates of data transformed by an ALR. ALDEx2 performs statistical tests on log-transformed values from a modelled probability distribution of the data set (Gloor et al. 2017).~~

~~The described methods clearly show the problems when trying to analyze compositional data in Euclidean space. They successfully work around the characteristics of compositional data however their interpretability and practicality leave much to wish for. Additionally, these methods are also not feasible for machine learning purposes, as it would increase the computational complexity dramatically. Thankfully, there is a very elegant way of solving this predicament: log-ratio transformations.~~

### Log-Ratio Transformations make compositional data analysis a lot easier

The difficulty of confined data points has already been commented on by Pearson (1897) in the context of spurious correlations and has been taken up by Aitchison 1982 in an attempt to overcome the “bounded sum problem”. Although compositional data exist in the simplex, Aitchison first documented that these data could get mapped into real space by use of the log-ratio transformation (Quinn et al. 2018; Aitchison 1982). This does not normalize the data (does not “open it”), but makes the interpretation of the transformed data dependent on the reference used and aim for a straight-forward univariate interpretation of the data (Quinn et al. 2018). Furthermore, it allows to employ standard analysis methods instead of the more complex alternatives introduced in the chapter before.

For all log-ratio transformations, relationships between the features in the data set are captured and taking the logarithm of these ratios makes the data symmetric and linearly related. It moves the simplex into real space and imparts key properties to the data set: scale invariance (performance does not change with e.g., sequencing depth), perturbation invariance (i.e., converting a composition between equivalent units will not change the results), and permutation invariance (i.e., changing the order of the components within a composition will not change the results).

Two more important properties exist that are transformation-specific: sub-compositional coherence (i.e., identical results are enforced when components are included in compositions or sub-compositions), and sub-compositional dominance (i.e., using a subset of a complete composition carries less information than using the whole) (Quinn et al., 2018). It has been shown recently that quasi-coherence is sufficient in practice, as well as quasi-isometry (Greenacre et al. 2022), as true isometry is difficult to interpret. However, keeping these characteristics in mind when choosing log-ratio transformations is important, as not every log-ratio transformation inherently incorporates all traits from Euclidean space.

Typical transformation techniques in compositional data consist of ALR (additive log-ratio), ILR (isometric log-ratio) and CLR (centered log-ratio). The latter uses the geometric mean of all input variables in place of the reference feature (Gloor et al. 2017). It has the advantage of being scale invariant and a good interpretability which makes it very practical. However, is not very useful in sparse data containing a lot of 0s. Later on, methodologies will be discussed to deal with 0 count values (Gloor et al. 2017).

Diagram

Description automatically generated with medium confidence

Figure : Equation for CLR

Equation describes calculation of CLR, with xj as vector of sample features, Dj the total number of features, and g(x) the geometric mean of sample vector x. Log-ratio transformations are applied within a feature (i.e., column-wise).

More complex is ILR (isometric log-ratio), which transforms the data with respect to an orthonormal coordinate system that is constructed from sequential binary partitions of features (Quinn et al. 2018). The ILR-transform maps a composition in the D-part Aitchison-simplex isometrically to a D-1 dimensional Euclidian vector, with clr(x) the centered log-ratio transform and V a matrix which columns form an orthonormal basis of the clr-plane (Greenacre et al. 2022).

Figure : Equation of ILR

Equation describes calculation of ILR, with xas vector of sample features, V a matrix which columns form on orthonormal basis of the clr-plane. Log-ratio transformations are applied within a feature (i.e., column-wise).

Isometric log-ratios are the “gold standard” of log-transformations, as they engender exactly the same multivariate geometric structure of the sample points as that of all the pairwise log-ratios, called the “log-ratio geometry” or also “Aitchison geometry” (Greenacre et al. 2021). Unfortunately, isometric log-ratios are particularly problematic when the numbers of components in the geometric means are high and thus lack interpretability (Greenacre et al. 2021).

A picture containing diagram

Description automatically generatedTherefore, transformations such as ALR (additive log-ratio) are re-evaluated in their effectiveness. In ALR, the logarithm is taken of each measurement within a composition and divided by a reference feature.

Figure : Equation ALR

Equation describes calculation of ALR, with xj as vector of sample features, Dj the total number of features, and xDj the reference feature. Log-ratio transformations are applied within a feature (i.e., column-wise).

Here, a small loss of isometry is traded off in favor of the benefit of a simpler and clearer interpretation of the log-ratio variables, as the interpretation of the results is always according to the chosen reference. When choosing a reference, Greenacre et al. 2021 propose to use three criteria to find a good reference for the denominator: (i) the reference component should maximize the Procrustes correlation between the additive log-ratio geometry and the exact log-ratio geometry, (ii) the reference should minimize the variance the relative abundances of log-transformed components, and (iii) it should be a well populated component. Using these guidelines produces additive log-ratios close to being isometric, which would make them a favorable log-transformation. Further discussable log-ratios are IQLR (inter-quartile log-ratio), PWLR (pair-wise log-ratio), and SLR (summed log-ratio). As those log-ratios come with higher complexity in terms of computational power and interpretability, they will not be part of this master thesis.

For machine learning purposes, it is still unclear if any log-ratio transformation improves the performance in a prediction task. This will be one of the core goals of this benchmarking project and previous studies and results will be described in the next section.

### Compositional Data in Machine learning

Predictive methods such as random forests (RF), artificial neural networks (ANN), deep learning (DL) or support-vector machines (SVM) and other methods have become in the last years increasingly popular (Tolosana-Delgado et al. 2019). Traditional machine learning methods can provide added predictive power with the price of limited explainability. Thus, balancing the predictive power with explainability becomes important for the conclusions.

In terms of statistical analysis, machine learning models are of great interest for microbiome analysis, as they allow predictions of biomarkers, phenotypes or microbial taxa, as well as other interesting tasks, that are not possible with the standard microbiome tool kit (Marcos-Zambrano et al. 2021). Therefore, a correct application of machine learning models is key to reproducible and interpretable research results. Several studies (Zhang and Shi 2019; Coenders and Greenacre 2021) showed log-ratio transformations in machine learning models with mixed performances. In 2019, Zhang and Shi compared several machine learning algorithms on geological compositional data and showed that overall, RF was the best performing model and that ILR and CLR were superior to ALR (Zhang and Shi 2019). Tolosana-Delgado et al. (2019) showed that ridge regression and SVM both need ILR. More observations were also made by Quinn et al. 2020. They performed linear discriminant analysis (LDR) on ILR-transformed data and partial least squares (PLS) to CLR-transformed data and showed good predictive results (Quinn and Erb 2020). Neural Networks require further research, but does not seem to be equivariant (Tolosana-Delgado et al. 2019), i.e. not any log-ratio works similarly well.

These observations demonstrate the core problematic of compositional data. Log-ratios in linear and generalized linear models are not easily chosen and depend heavily on the observations at hand. The reason why ALR was outperformed by Zhang and Shis study (2019) for example, was probably due to a badly chosen reference and this makes the direct comparison of several studies difficult. In general, log-ratio transformations seem to outperform raw proportions for classification tasks, but it is not clear how log-ratio transformations relate to the changes in predictive performance. Furthermore, employing log-ratio transformations leads to an increase in complexity in the correct application of machine learning models. Thus, it is of increasing importance to create a practical guide for all scientists who want to employ such analysis.

The question arises if machine learning models are “worth the hassle” considering microbiome-specific algorithms like “*selbal*” and “*CoDaCoRe*” exist, that are faster and do not require a lot of background knowledge to use. The next section describes these models more in detail and their potential effectiveness.

### Microbiome-specific Algorithms

#### CoDaCore

2021 Gordon-Rodriguez et al. published a novel learning algorithm for finding balances called “CoDaCoRe” (Compositional Data via Continuous Relaxations). Balances are defined as the log-ratios between geometric means of two subsets (or features) of the input variables. Translated, CoDaCoRe finds ratios between two features that are explanatory for the given classification task. Such ratios are commonly used as biomarkers of gut health e.g., the Firmicutes-to-Bacteroidetes ratio (Crovesy et al., 2020; Magne et al.; 2020).

Balances are essentially pairwise log-ratios; however, they allow the aggregation of more than one variable in the numerator and denominator of the log-ratio. This leads to a richer set of features and therefore more flexible models. As mentioned in the section about log-ratio transformation, pairwise log-ratios are not included in the master thesis, since they are very computationally taxing. However, in CoDaCoRe Gordon-Rodrigues et al. use deep learning technology called “continuous relaxation” and only approximate the optimization problem, which has the advantage of greatly reducing the runtime.

In its basic formulation, CoDaCoRe learns a regression function, which uses balances as weights. The goal of CoDaCoRe is to find the balance that is maximally associated with the response variable by minimizing the cross-entropy loss. The continuous relaxation approximates the geometric averages over subsets of the inputs, by weighted geometric averages of all components. This makes the relaxation and balances differentiable and allows the use of gradient descent. This has the advantage of a linearly scaling computational cost instead of exponential, which reduces the runtime drastically.

At this step, weighted geometric averages or not easily interpretable. Therefore, CoDaCoRe implements a discretization procedure, i.e., fitting a linear model to assess if the previously found balance is impactful. This step can be regularized by influencing lambda in the model creation, which becomes a regularization hyperparameter that can be tuned. In practice, lower lambda is more useful when the emphasis is on predictive accuracy rather than interpretability or sparsity.

In summary, in the full CoDaCoRe algorithm, multiple regressors are trained in a stage-wise additive fashion and afterwards each successive balance is fitted on the residual from the current model. Thus, CoDaCoRe identifies a sequence of balances, in decreasing order of importance, each of which is sparse and interpretable.

CoDaCoRe is a promising algorithm that is created to also work efficiently on big data sets with a lot of features. In their paper, the authors compare CoDaCoRe against several machine learning models (Lasso, RF and XGBoost) and show that their algorithm does not sacrifice interpretability nor predictive accuracy. As they do not show various machine learning models and their performances over various log-ratio transformations against their algorithm, one goal of this master thesis will be to include such tests.

#### ~~Selbal~~

~~The selbal algorithm, published 2018 by Rivera-Pinto et al., has gained popularity as a method for automatically identifying balances that predict a response variable. It is a greedy stepwise algorithm that searches for a sparse model that adequately explains the response variable of interest. In multiple regression a new taxon is added to the model each time and assessed if its relative abundance (or balance) is predictive of the outcome (Rivera-Pinto et al. 2018). It has been developed specifically for microbiome data and has been shown to work effectively. However, this algorithm scales poorly in the dimensions of the input.~~

### Implementation

The former sections describe the problems and uncertainties regarding log-ratio transformation in machine learning procedures. This project focuses on collecting insights on the performances of machine learning models, but also practicality. Recent years made it clear that machine learning is a tool that should be available to all biologists, but comes with high complexity and its own pitfalls, even without the addition of mathematical characteristics of compositional data and log-ratio transformations.

Therefore, the focus will be on standard machine learning models, that are already incorporated in easy-to-use packages in R: generalized linear models (GLMs) and XGBoost as non-linear approach.

As several authors pointed out (Quinn and Erb 2020; Gloor et al. 2017), machine learning performance is also influenced by data size. Therefore, three data sets were chosen accordingly to include direct comparison of performances of small and large data sets, as well as high and low known correlations microbiome and host, as well as continuous and discrete predictive variables.

First is a Colorectal Cancer (CRC) set. The CRC data set was first used and described by (Wirbel et al. 2019) in their meta-analysis for colorectal cancer. This data set is well known and contains 7727 features with 695 samples. It shows clear correlations between microbiomes and colorectal cancer. It is therefore helpful to show the behavior of transformations and machine learning algorithms on small but highly specific data sets. A second data set is the Polycystic Ovary Syndrome (PCOS) data set from Kreete et al. (2020). It observed 312 individuals, with two-thirds of them being healthy, and 72738 features. It is a valuable addition as it is a small data set that shows no (low) correlation between the disease and microbiome structure. Lastly, is the Estonian Biobank microbiome cohort (EstMB). This data set includes 2509 individuals with several phenotypical markers collected over time and 17180 features overall, which makes it by far the largest of all three sample-wise. All data sets contain at least one discrete and one continuous response variable.

Furthermore, five log-ratio transformations will be compared: TSS (total sum scaling transformation), CLR (centered log-ratio), and three ALR methods (worst, random and optimal). The performance of these transformations is directly compared to the CoDaCoRe performance in the following conceptual pipeline:

Diagram

Description automatically generated

Figure 5: Used Pipeline

The graph shows the proposed pipeline for the benchmarking project. Data sets will be collected by their characteristics large/small, high/low correlations and continuous/discrete variables. Afterwards, data sets will be pre-processed by zero-imputation methods and filtering. Microbiome-native methods will be employed and compared to the data being log-transformed and used in machine learning models.

The general pipeline will be constructed of the following building blocks: Pre-processing, Imputation, Transformation, and Machine Learning Models/Microbiome Approaches. The core idea is to observe statistical fluctuations in all models given the same training data set. After the data split, a repeated cross-validation is used to find the best model and its performance is saved for plotting.

An additional point of interest is the impact of data leakage on transformations and machine learning models. It is considered good standard-of-practice to split the data before any pre-processing or imputation is conducted. As transformations fall under the category “pre-processing” and are conducted column-wise in nature (especially ALR), it should be tested how machine learning models are impacted under data leakage and non-data-leakage circumstances. We hypothesize that the concept of data leakage is of less importance than the problematics that arise when splitting the data before transformations. Especially in ALR, different denominators could be chosen for test and training set, which would impact the interpretability of the results.

Furthermore, an additional qualitative analysis will show, if picked ratios are consistent throughout all repetitions in codacore and also ALR, especially for the high correlation data sets.

In summary, a lot of the provided information show promises in terms of predictive performances of log-ratio transformations for machine learning models compared to no transformation. However, considering the small number of studies and its benchmarking aspect, it context for this project they should be taken more as a guideline instead of face-value. Therefore, re-validating their results could prove to be beneficial for the scientific community. Furthermore, the observations from all these papers show, that the selection and performance of the best algorithm is heavily dependent on the dataset, its research hypotheses, and models. It is therefore difficult to understand and handle for non-experts, but unfortunately vital to the scientific community. Thus, this benchmarking project will focus on establishing a pipeline, as well as recommendations and guidelines that reduce human error and hopefully improves quality management in machine learning methodology.

## Pipeline

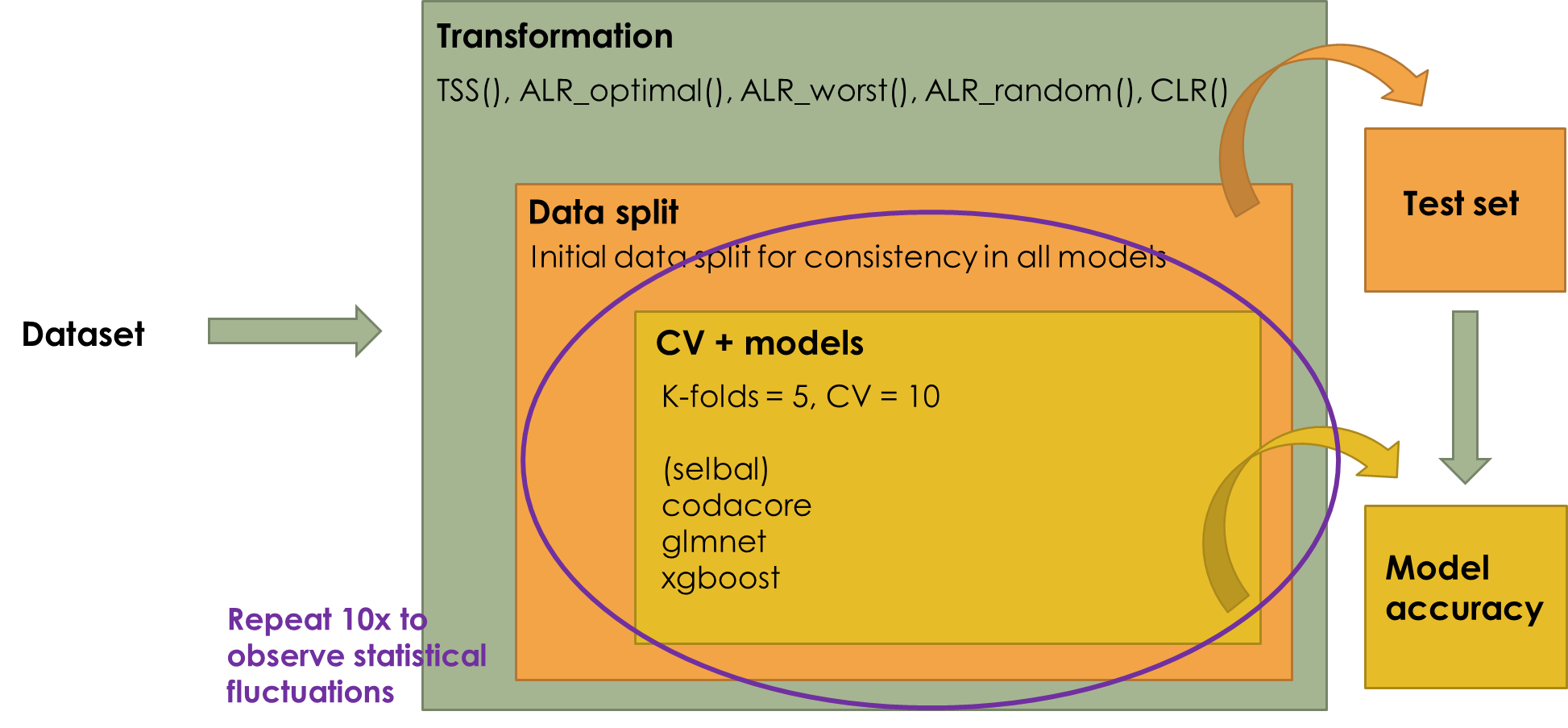


Figure : Used Pipeline

The graph shows the proposed pipeline for the benchmarking project. Data sets will be collected by their characteristics large/small, high/low correlations and continuous/discrete variables. Afterwards, data sets will be pre-processed by zero-imputation methods and filtering. Microbiome-native methods will be employed and compared to the data being log-transformed and used in machine learning models.

The goal of this pipeline is to compare the performance of classic machine learning models after different log-ratio transformations with microbiome-native and automated algorithms such as *selbal* and *CoDaCoRe*. This way it will become clearer if a scientist should make the effort in learning about machine learning methods, when automated algorithms perform well and no prior machine learning knowledge is necessary.

There are three common types of analyses conducted in microbiomics with machine learning approaches (Marcos-Zambrano et al. 2021): (i) classification and prediction of microbial taxa, (ii) prediction of host phenotype, and (iii) usage of microbial communities for understanding disease mechanisms (i.e., biomarker-finding).

To keep the size of this project manageable, the focus will be on prediction and classification tasks. This partially includes feature selection, e.g., in ElasticNet (ENET) and RF models. Additionally, as several authors pointed out (Quinn and Erb 2020; Gloor et al. 2017), machine learning performance is also influenced by data size. Therefore, data sets should be chosen accordingly to include direct comparison of performances of small and large data sets. Furthermore, phenotype variables with high and low known correlations between microbiome and host will be chosen, as well as continuous and discrete predictive variables.

## Machine Learning Models

Using Machine Learning models always includes some form of cross-validation to ensure a low bias in machine learning models. One of the recurring methodologies is nested cross-validation algorithm. This is an approach to model hyperparameter optimization and model selection that attempts to overcome the problem of overfitting the training data set which often happens in standard cross-validation procedures (Cawley and Talbot 2010). Typically, the k-fold cross-validation procedure involves fitting a model on all folds but one and evaluating the fit model on the holdout fold. Each training dataset is then provided to a hyperparameter optimized procedure that finds an optimal set of hyperparameters for the model (Cawley and Talbot 2010). Additionally, stratification will be included. In stratified nested cross-validation during splitting of data into folds it is ensured that each fold has the same proportion of observations to ensure balancing. Here, a 10-fold stratified nested cross-validation protocol will be implemented, as it is standard now in various microbiome analyses (Marcos-Zambrano et al. 2021; Wirbel et al. 2019).

Tsamardinos et al. (2015) showed that a stratified nested cross-validation algorithm shows the least bias compared to standard cross-validation algorithms. They also propose to always include repetitions of inner CV loop for small data sets to reduce variances (Tsamardinos et al. 2015). Their computation of bias could be implemented as a control before feeding the data into machine learning models. The bias is computed as L(hold-out) – L(estimation), with L(hold-out) being the performance of 70% of the data set, whereas 30% of each data set were used for sub-sampling (here n = 30) and training of the model (Tsamardinos et al. 2015).

It is a general consensus in the statistical community that most problems can be described via classical machine learning models (Marcos-Zambrano et al. 2021). Therefore, this pipeline will only include standard and most frequently used models. In microbiome analyses, most applications for machine learning are classification tasks in supervised learning. Therefore, ElasticNet (ENET) will be used as regression model and XGBoost (XGB) as random forest approach, also to have a direct comparison to *selbal* and *CoDaCoRe.* Additionally, Linear Discriminant Analysis (LDA) will be conducted.

As most models will employ binary classification tasks, the following performance metrics will be proposed: steadily recurring performance metrics are of course AUROC, and Accuracy metrics, as well as MAE (mean absolute error).

To assess if the difference in model performances is statistically significant, Statnikov et al. (2013) employed Random Permutation testing. Additional methods mentioned in literature are McNemar’s test, 5x2-fold cross-validation with modified paired students t-test and Wilcoxon signed-rank test.

## Standard Microbiome Approaches

Using tested and published libraries for microbiome analysis is the easiest way to reduce human error and improve quality management. Two approaches are used frequently, and they will present the baseline comparison if one should use those packages or a machine learning model. One is called *selbal* and was proposed by Rivera-Pinto et al. (2018). It is based on standard generalized linear models. The second one will be *CoDaCoRe* proposed by (Gordon-Rodriguez et al. 2021), which is based on random forest analysis.

# Methodology

For the data analysis and model pipelines, the script language R (v4.1.3) in combination with RStudio (v2022.02.1+461) has been used. For data cleaning and filtering the main libraries is “tidyverse” (1.3.1). Imputation was conducted with “zCompositions” (1.4.0.1), and transformations were mostly done with “easyCODA” (0.34.3). Models were constructed with “mikropml” (1.2.2), “tidymodels” (0.2.0) and “codacore” (0.0.3).

Additionally, own scripts were created for convenience purposes. All scripts can be found on Github JenniferNeumaier/ml\_coda.

## Pre-Processing

### Cleaning and filtering

First, all data sets were cleaned in order to remove NAs in predictor columns or patients that have no sequencing data. In EstMB data set, 21 rows removed in metadata due to NA and 21 patients respectively cut out of abundance table. This leads to 2485 final sample-size. In CRC, 128 rows were removed due to NA in feature “BMI”, leading to 567 samples overall. Additionally, the column “X.1” has been removed as it is only a sum of all abundances per row. In PCOS, 6 rows removed in abundance table because no matching patient has been found in metadata, reducing the number of samples to 304.

As microbiome data usually has a lot of features, the computational work can be taxing. Therefore, filters were applied to all three data sets. In this benchmarking project, taxa with ≥10% abundance in samples will be discarded. Additionally, a filter of ≥50% abundance in samples will be applied, as well as a mean relative abundance filter for 0.001. For 10% abundance filters EstMB keeps 9738 features, CRC 650 features and PCOS 1154 features. Respectively, for 50% abundance filters EstMB keeps 5233 features, CRC 189 features and PCOS 120 features. For EstMB data, 90% abundance was used, as the data was otherwise not practically usable without heavy computational power. EstMB keeps 3062 features after filtering.

### Imputation

One of the main problems of microbiome data is its sparse nature. When working with relative abundances this is annoying but doesn’t have any mathematical consequences. On the other hand in log-ratio transformation zeros lead to problems, as log(0) is undefined. Therefore, one of the first steps after filtering and before log-ratio transformation is zero-imputation. Introduced by (Palarea-Albaladejo and Martín-Fernández 2015) is pseudocount. It has been frequently used for statistical analysis of microbiome data. It adds a pseudo-count of 1E-05 to avoid non-finite values resulting from log(0). All three data sets were imputed with Geometric Bayesian Multiplicative (GBM) and output form “p-counts”.

## Transformations

As mentioned in the introduction, choosing a log-ratio is not an easy decision. In order to stay with the goal of improving quality management and reducing human error, ILR will left out, as it is the most difficult one to work with and interpret. Similarly, pair-wise log-ratio transformations will also not be tested, as they are very computationally taxing. It has been decided to use TSS (total sum scaling transformation), which is standard relative abundance data, and compare it to CLR and ALR transformed data.

As ALR would be the most promising log-ratio transformation in terms of interpretability and its closeness to ILR, we will compare ALR transformation in three ways: (i) a random reference will be picked as denominator, to assess the average performance of machine learning models for ALR, (ii) find the most optimal denominator and (iii) worst ALR denominator via Greenacre et al. (2021) proposed way of finding a reference. Included in the package “easyCODA” is the function ALR() that assesses the abundances and variances of features in a data matrix, followed by a Procrustes analysis to assess their geometry. This leads to a list of possible good denominators for the respective data set if the top results are chosen or worst denominators, if the bottom results are selected. Similarly, “easyCODA” also contains the function CLR() to compute the centered log-ratio.

## Machine LEarning models

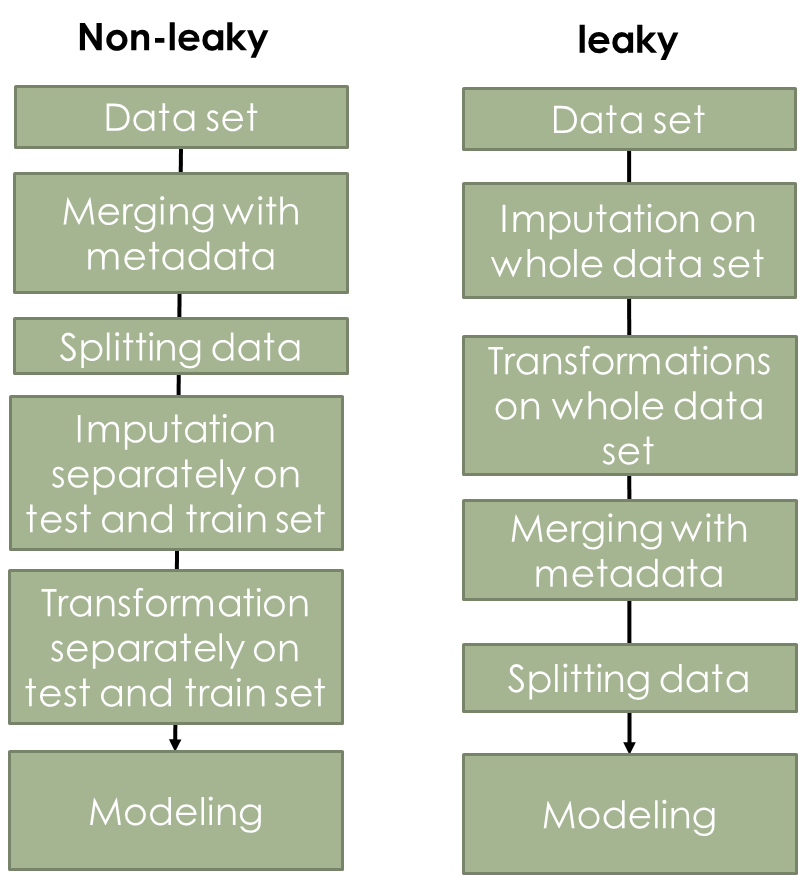
Figure .. shows the pipeline for comparing the performance of machine learning models. First, the preprocessed (and imputed) data set was transformed using TSS, CLR, and various ALR options. Afterwards, the data set was split 80/20 with stratification with the package “tidymodels”. The train set given to the function run\_ml() from “mikropml” by. This package nicely compacts the use of standard machine learning models to a few lines of code and supports the use of GLMs (glmnet), as well as XGBoost (xgbTree). As shown in the pipeline, it was of interest to control the initial split into test and train data, which is also allowed by mikropml. The training set is then split into 5 folds and the best model assessed via 10-fold cross validation and the final test and training scores of the best model are saved for plotting. This procedure is repeated 10x for each model and each data set to assess statistical fluctuations of model performances and accuracies.

Diagram

Description automatically generated

As it was of interest to compare the machine learning model performances to codacore, the pipeline includes codacore directly. The first initial data split is fed to the function codacore(). Two codacore models are trained, one with lambda = 0 and the other with lambda = 1. For both models, AUC values were predicted with all balances taken into consideration and one were only the best balance was used. This creates four performance scores per repetition and saved for further plotting. The codacore function is also repeated 10x to catch statistical fluctuations under the same data split. For discrete response variables, AUC is chosen as performance score, and for continuous response variables RMSE.

## Data Leakage in Transformations

In order to assess the influence of transformation on the concept of “data leakage” (machine learning best practice paper), a small test was conducted. In this, the CRC and PCOS data sets were used, and it was compared how the test and training set performances behave with imputation and transformation before data merging vs. after data merging (see figure).

In the non-leaky procedure, the processed data set has been first merged with the metadata to include the predictor column. Afterwards, the data is split into train and test set and imputation, as well as three transformations (TSS, CLR and optimal ALR) are performed separately on both. Finally, train and test set are fed into a glmnnet model.

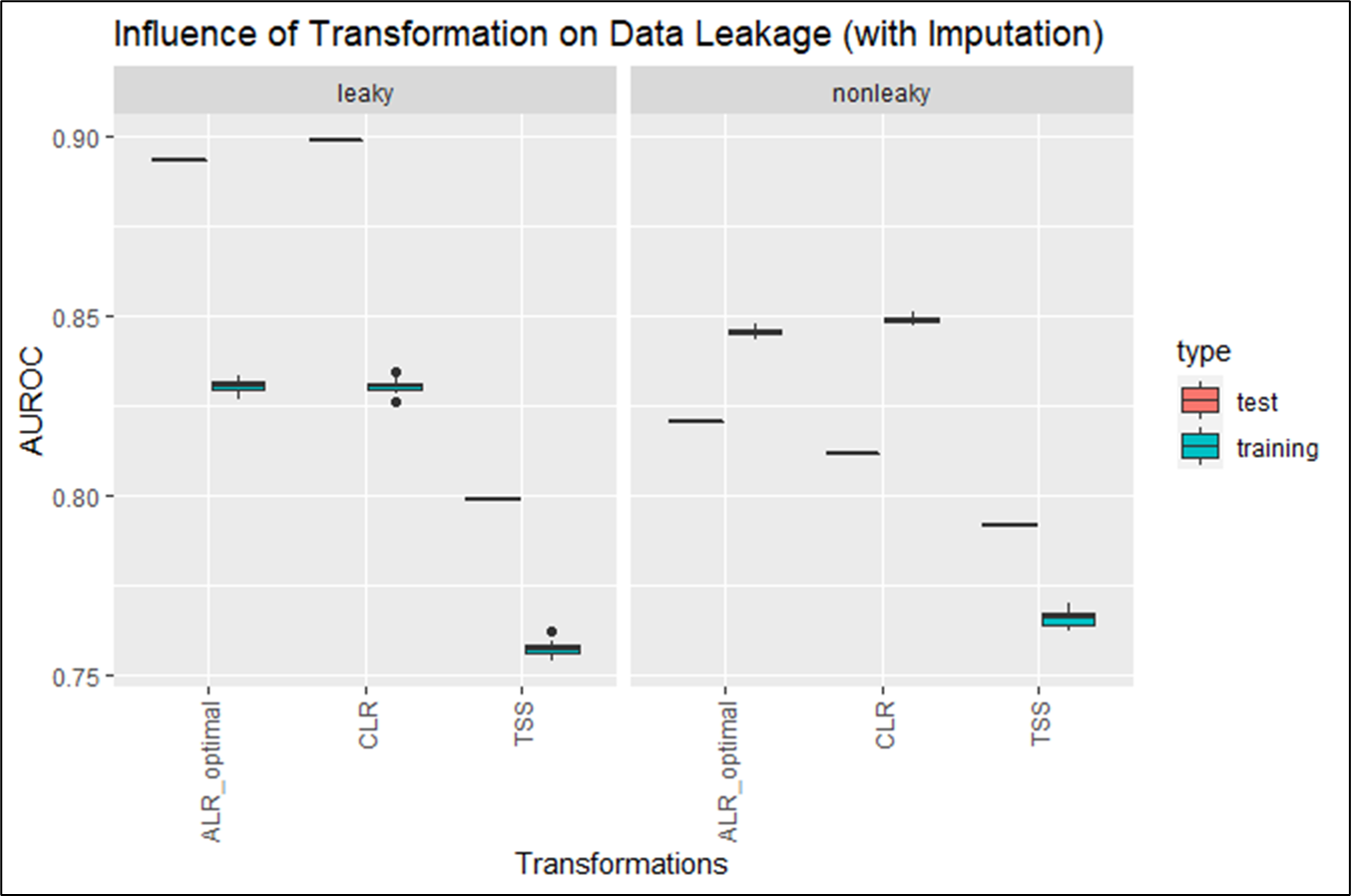
In the leaky procedure, imputation and transformations are conducted on the whole data set and afterwards the data is merged with the metadata and split into train and test set and fed into the glmnet model. Both procedures were repeated 10x to observe statistical fluctuations. Only 10% abundance filter data sets have been used.

# Results

In the following section all results gathered throughout the project are introduced and described. This section is divided into three parts, each containing the results of one dataset, comparing different transformations directly with codacore and selbal results.

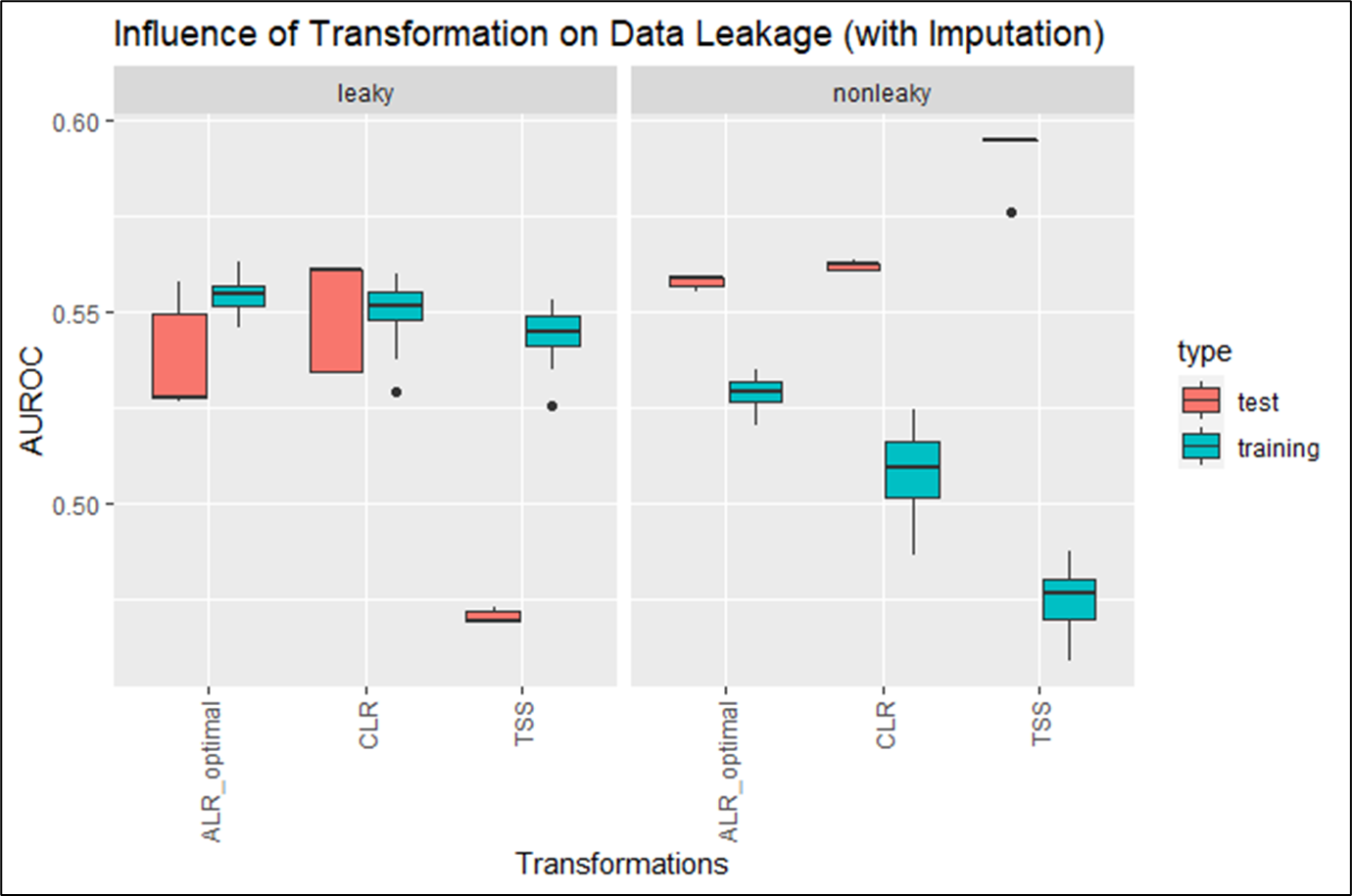
## Leaky Preprocessing

In these results, CRC (10% filtering and imputed) was predicted once on GER and once on FRA as test set and both results combined for this figure.



The figure compares the behavior of AUC in train and test set in a leaky and non-leaky procedure. It can be seen that the achieved AUCs for ALR\_optimal and CLR are both around 0.86, however the leaky procedure shows a lower variance compared to non-leaky. TSS shows in both procedures a AUC between 0.6 and 0.78, with the training set converging on a AUC of around 0.78.

The same procedure was used on the PCOS data set to estimate the influence of data leakage in less clear correlations.



Similar to figure () leaky and non-leaky behavior is compared directly. In PCOS the AUC varies between 0.55 and 0.6, with the training sets converging on 0.55. The test sets however show higher variances.

## Colorectal Cancer Dataset (CRC)

### GLMNET

#### Classification

In this figure, the predictor column was “Group”, i.e. the column that determined if someone had cancer or not. All models were trained on the same training set and the same test set. A holdout set was not used here.

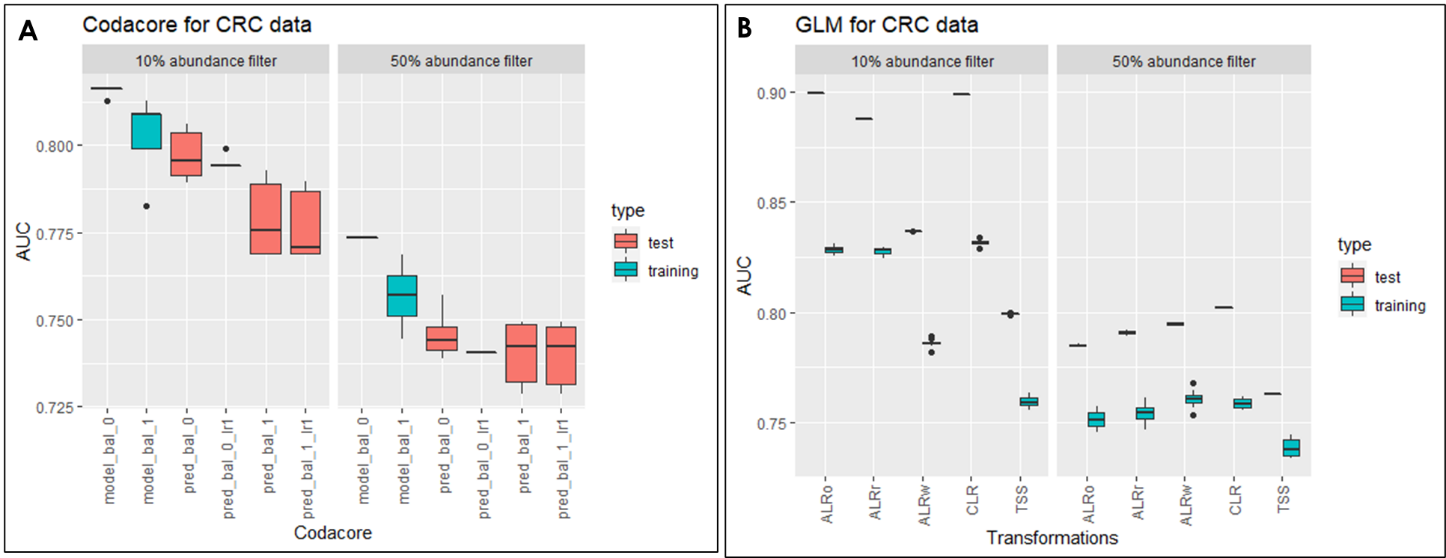


Figure A shows the result for codacore split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the AUC performances and on the x-axis are all codacore models. In both data sets the training set shows higher performances than the test set, with an AUC for 10% abundance filter data set of 0.8 and 0.775 for 50% abundance filter set. The test set ranges between 0.8 for 10% abundance filter and 0.75 for 50% abundance filter for all codacore models.

Figure B shows the result for glmnet split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the AUC performances and on the x-axis are the different transformations. Performances for train and test set vary greatly over all transformations for both data sets, with the train tests showing significantly higher performances between 0.8 and 0.9 for 10% abundance data set and constant 0.8 for 50% filter set. The training set is in general significantly lower than the respective test set with the lowest performances in the 50% abundance filter set with AUC performances around 0.75.

#### Regression

In this figure, the predictor column was “BMI”. All models were trained on the same training set and the same test set. A holdout set was not used here.

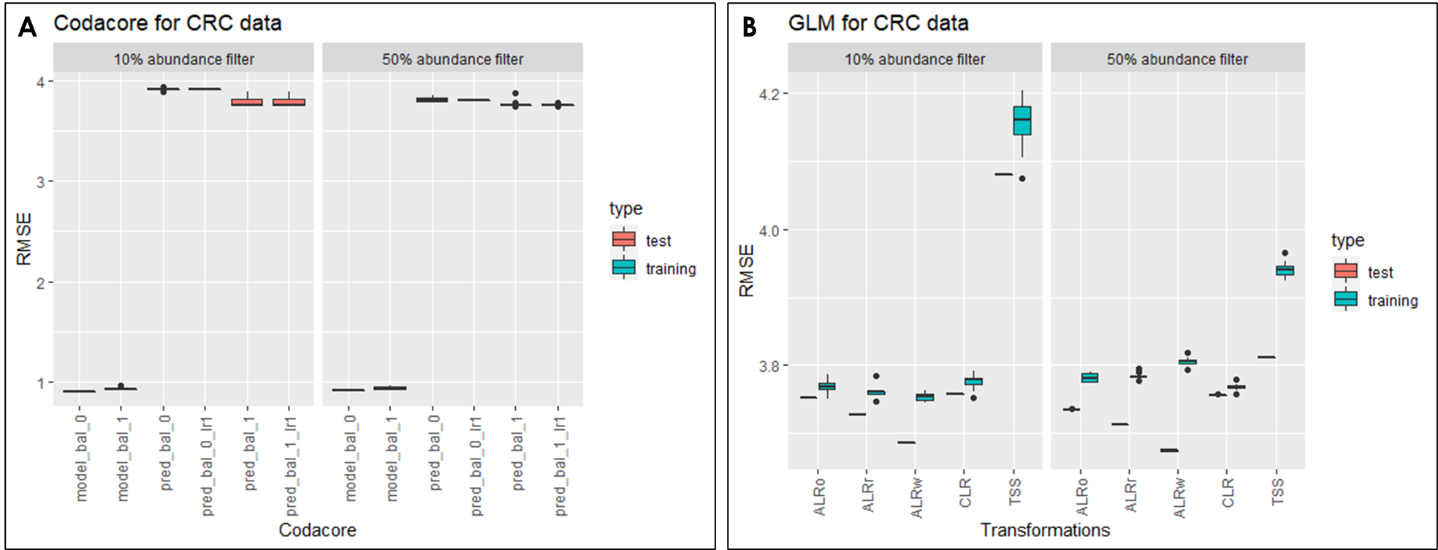


Figure A shows the result for codacore split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the RMSE performances and on the x-axis are all codacore models. In both data sets the test set shows higher performances than the training set, with an RMSE between 1 for all training sets and 4 for all test sets with very low variances.

Figure B shows the result for glmnet split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the RMSE performances and on the x-axis are the different transformations. Performances for train and test sets sit around 3.8 for all transformations besides TSS which shows performances for train and test set around 4.2 for 10% abundance filtering and 3.9 for 50% abundance filters. In general, test sets show lower performances than training sets.

## Polycystic Ovary Syndrome Dataset (PCOS)

The following section describes the result for several machine learning models and their performances on the PCOS data set under various transformations.

### GLMNET

#### Classification

In this figure, the predictor column was “PCOS\_Riikka”, i.e. the column that determined if someone had PCOS or not. All models were trained on the same training set.

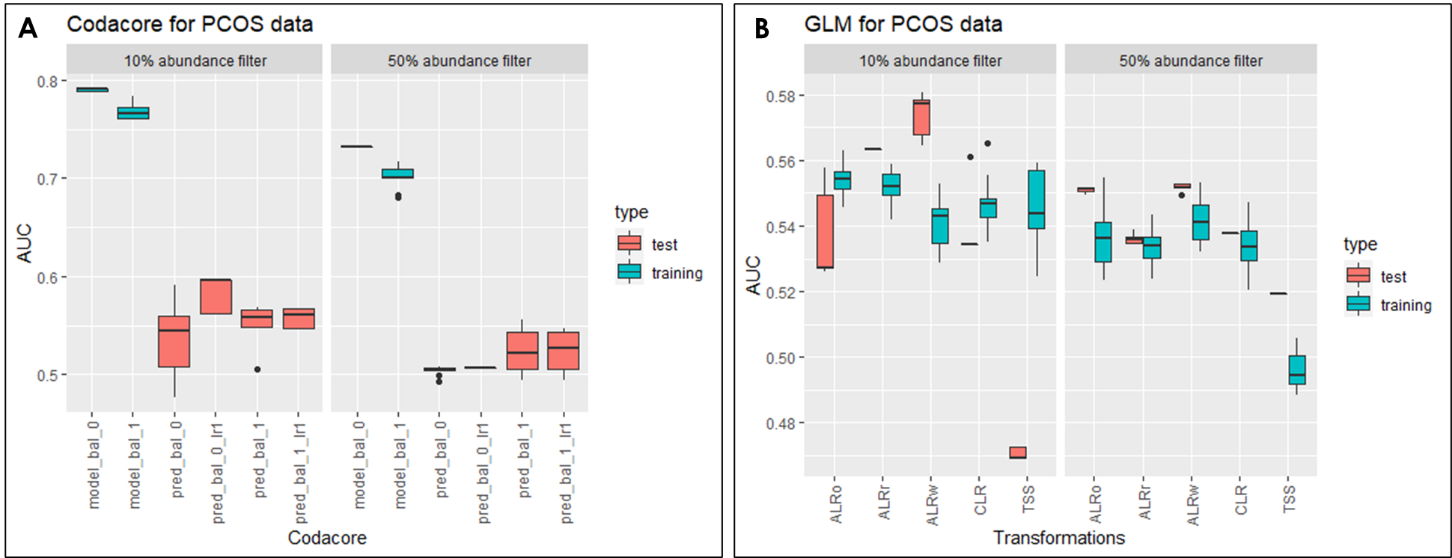


Figure 6

ALRw = worst ALR, ALRo = optimal ALR, ALRr = random ALR.

Figure A shows the result for codacore split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the AUC performances and on the x-axis are all codacore models. In both data sets the training set shows higher performances than the test set, with an AUC between 0.7 and 0.8. The test set ranges between 0.5 and 0.6 for all codacore models.

Figure B shows the result for glmnet split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the AUC performances and on the x-axis are the different transformations. Performances for train and test set vary greatly over all transformations for both data sets, with the train tests showing also higher variances. In general, the AUC trends between 0.48 and 0.58, with the test set for TSS as the lowest and worst ALR as the highest with 0.58.

#### Regression

In this figure, the predictor column was “BMI”. All models were trained on the same training set.

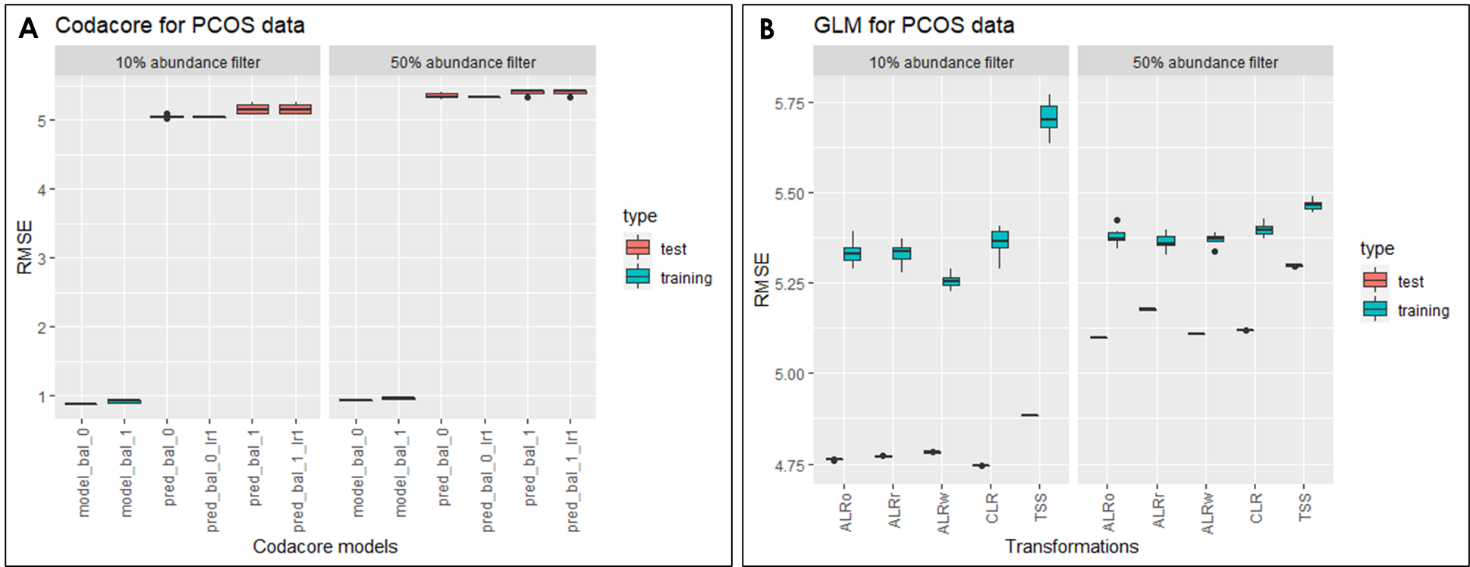


Figure 7

ALRw = worst ALR, ALRo = optimal ALR, ALRr = random ALR.

Figure A shows the result for codacore split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the RMSE performances and on the x-axis are all codacore models. In both data sets the test set shows higher performances than the training set, with an RMSE between 1 for all training sets and 5 for all test sets with very low variances.

Figure B shows the result for glmnet split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the RMSE performances and on the x-axis are the different transformations. Performances for train and test set vary greatly over all transformations for both data sets, with the train tests showing also higher variances. In general, the RMSE for training sets trends between 5.25 and 5.75, and the test set around 4.75 for 10% abundance and 5.1 to 5.25 for 50% filtering.

### xgboost

#### classification

#### regression

## Estonian Biobank Dataset (EstMB)

# Discussion

Performance selbal and codacore:

* takes quite a while for 500x1000 data set
* at least 5-10 minutes for selbal
* codacore faster
* with CV even more
* tensorflow necessary for codacore -> installation problems

comparison codacore and mikropml:

* impact of filtering -> 50% seems to lose too many features, performance generally worse than 10%.
* PCOS: regression overfitting, classification all over the place
* CRC: regression pretty constant (varies only in second decimal place), classification underfitting
* ALRo and CLR seem to have very similar results over both data sets and model types
* Even ALRw is better than TSS for both high and low correlation and regression and classification -> supports former papers that suggest using transformations for compositional data -> also for machine learning concepts
* Do CRC holdout AUCs match paper? Yes
* Transformations plus standard split seems to UNDERFIT data?
* 50% abundance is too few features so not even transformations impact performances

This is because an underfit model has low variance and high bias. Variance refers to how much the model is dependent on the training data. For the case of a 1 degree polynomial, the model depends very little on the training data because it barely pays any attention to the points! Instead, the model has high bias, which means it makes a strong assumption about the data. For this example, the assumption is that the data is linear, which is evidently quite wrong. When the model makes test predictions, the bias leads it to make inaccurate estimates. The model failed to learn the relationship between x and y because of this bias, a clear example of underfitting. (<https://towardsdatascience.com/overfitting-vs-underfitting-a-complete-example-d05dd7e19765>) -> xgboost performance in comparison?

**Q: how is codacore working?**

Data leakage:

* in data sets with clear correlation, performance is similarly good, with leaky procedure showing lower variance and therefore preferable.
* Reasoning: using test and train set to perform transformation could potentially not be big enough and therefore lead to higher variances in transformation results. Also, denominators for ALR were different for test and train set -> both denominators were removed for modeling
* Makes interpretability even harder -> use fixed denominator for test set (i.e. denominator from train set)?
* As leaky procedure does not seem problematic it is practicable to conduct transformations on the whole data set
* For data sets that show low correlations and are difficult for models, the nonleaky prodecure seems to work better. However, the AUCs are very similar.

**Q: How is imputation influencing leakiness? -> include imputation in pipeline**

Publication bibliography

Aitchison, J. (1982): The Statistical Analysis of Compositional Data. In *J. Royal Statistical Society* 44, pp. 139–177.

Cawley, Gavin C.; Talbot, Nicola L. C. (2010): On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. In *Journal of Machine Learning Research* 11, pp. 2079–2107.

Coenders, Germa; Greenacre, Michael (2021): Three approaches to supervised learning for compositional data with pairwise logratios. Available online at http://arxiv.org/pdf/2111.08953v1.

Gloor, Gregory B.; Macklaim, Jean M.; Pawlowsky-Glahn, Vera; Egozcue, Juan J. (2017): Microbiome Datasets Are Compositional: And This Is Not Optional. In *Frontiers in microbiology* 8, p. 2224. DOI: 10.3389/fmicb.2017.02224.

Gordon-Rodriguez, Elliott; Quinn, Thomas P.; Cunningham, John P. (2021): Learning Sparse Log-Ratios for High-Throughput Sequencing Data.

Greenacre, Michael; Grunsky, Eric; Bacon-Shone, John; Erb, Ionas; Quinn, Thomas (2022): Aitchison's Compositional Data Analysis 40 Years On: A Reappraisal. Available online at http://arxiv.org/pdf/2201.05197v1.

Greenacre, Michael; Martínez-Álvaro, Marina; Blasco, Agustín (2021): Compositional data analysis of microbiome and any-omics datasets: a revalidation of the additive logratio transformation.

Griffith, Malachi; Walker, Jason R.; Spies, Nicholas C.; Ainscough, Benjamin J.; Griffith, Obi L. (2015): Informatics for RNA Sequencing: A Web Resource for Analysis on the Cloud. In *PLoS Comput Biol* 11 (8), e1004393. DOI: 10.1371/journal.pcbi.1004393.

Marcos-Zambrano, Laura Judith; Karaduzovic-Hadziabdic, Kanita; Loncar Turukalo, Tatjana; Przymus, Piotr; Trajkovik, Vladimir; Aasmets, Oliver et al. (2021): Applications of Machine Learning in Human Microbiome Studies: A Review on Feature Selection, Biomarker Identification, Disease Prediction and Treatment. In *Frontiers in microbiology* 12, p. 634511. DOI: 10.3389/fmicb.2021.634511.

Palarea-Albaladejo, Javier; Martín-Fernández, Josep Antoni (2015): zCompositions — R package for multivariate imputation of left-censored data under a compositional approach. In *Chemometrics and Intelligent Laboratory Systems* 143, pp. 85–96. DOI: 10.1016/j.chemolab.2015.02.019.

Quinn, Thomas P.; Erb, Ionas (2020): Interpretable Log Contrasts for the Classification of Health Biomarkers: a New Approach to Balance Selection. In *mSystems* 5 (2). DOI: 10.1128/mSystems.00230-19.

Quinn, Thomas P.; Erb, Ionas; Richardson, Mark F.; Crowley, Tamsyn M. (2018): Understanding sequencing data as compositions: an outlook and review. In *Bioinformatics (Oxford, England)* 34 (16), pp. 2870–2878. DOI: 10.1093/bioinformatics/bty175.

Rivera-Pinto, J.; Egozcue, J. J.; Pawlowsky-Glahn, V.; Paredes, R.; Noguera-Julian, M.; Calle, M. L. (2018): Balances: a New Perspective for Microbiome Analysis. In *mSystems* 3 (4). DOI: 10.1128/mSystems.00053-18.

Tolosana-Delgado, Raimon; Khodadadzadeh, Mahdi; Talebi, Hassan (Eds.) (2019): On machine learning algorithms and compositional data.

Tsamardinos, Ioannis; Rakhshani, Amin; Lagani, Vincenzo (2015): Performance-Estimation Properties of Cross-Validation-Based Protocols with Simultaneous Hyper-Parameter Optimization. In *Int. J. Artif. Intell. Tools* 24 (05), p. 1540023. DOI: 10.1142/S0218213015400230.

Wirbel, Jakob; Pyl, Paul Theodor; Kartal, Ece; Zych, Konrad; Kashani, Alireza; Milanese, Alessio et al. (2019): Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer. In *Nature medicine* 25 (4), pp. 679–689. DOI: 10.1038/s41591-019-0406-6.

Zhang, Mo; Shi, Wenjiao (2019): Systematic comparison of five machine-learning methods in classification and interpolation of soil particle size fractions using different transformed data.

# Supplementary

## Influence of Transformation on Holdout vs. 80/20

This test was focused on showing if there are differences in the model performances in various transformations when choosing a specific holdout set vs a standard 80/20 set. As can be read in the CRC paper, it has been decided to use a holdout set and leave-one-out principle to validate the model. As the other data sets do not have the option and it would be of interest to see if transformation impact the choice of test sets, the AUC performances of a standard 80/20 split were compared to performances for every holdout group in the CRC metadata.

Chart

Description automatically generated

The figure shows the result for glmnet split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the AUC performances and on the x-axis are the different transformations. The data set has been trained on the predictor “Group”. In these boxplots, performances for training and test were combined to capture the distances between them. In the supplementary a figure is added that shows every data point. Using USA as holdout set leads to the biggest distance between training and test performance (0.7 to 0.9) for all transformations and data set types. Using GER as holdout set produces the smallest distance between test and training set performances. In general, TSS performs worse compared to all other transformations, with CLR and ALR random and ALR optimal having the highest test set performances for 10% abundance filter.

Separating train and test performances and instead combining all holdout performances support the claim for consistency of 80/20:

Chart, scatter chart, box and whisker chart

Description automatically generated

The figure shows the result for glmnet split into the performances for 10% abundance (left) and 50% abundance (right). The y-axis contains the AUC performances and on the x-axis are the different transformations. The data set has been trained on the predictor “Group”. It can be seen that the performances on the standard test split show higher training and test performances than the combined holdout set performances. Interestingly, the standard split seems to be underfitting the data, compared to the holdout set method. Holdout AUCs match compared to original paper (in range of 0.7 and 0.8). Standard split seems to perform even better.

