Ludwig-Maximilians-University
Munich

Institute for Statistics

Master Thesis

A comparison study of prediction approaches for multiple training data sets and test data with block-wise missing values

Author:
Frederik
Ludwigs

Supervisor:
Dr. Roman
Hornung

July 16, 2020

Abstract

This thesis compares different prediction approaches for training- and test-sets with block-wise missing values. It introduces five random forest based approaches to deal with block-wise missingness. Additionally the adaptions of the priority-Lasso from Hagenberg's thesis [1] and the mdd-sPLS method from Lorenzo et al. [2] are briefly introduced. The predictive performances of the approaches are investigated and compared based on various data sets with different patterns of block-wise missingness. Among the random forest based approaches, the 'Imputation', 'Fold-wise' and 'Block-wise' approach provide promising results. The idea of the 'Fold-'/ 'Block-wise' approach was taken from [3]/ [4] and extended with a new weighting scheme. This new weighting scheme leads to better results compared to the original approaches without the weighting scheme. The mdd-sPLS method performs rather bad and is always outperformed by the priority-Lasso adaptions, as well as most of the random forest based approaches. Most of the priority-Lasso adaptions perform quite well and outperform the diverse random forest based approaches. The drawback of the different priority-Lasso adaptions is that they require prior knowledge of the data - e.g. which feature-blocks are how important for the target variable. With the random forest based adaptions, this is not necessary. Hence the optimal approach for data with block-wise missingness depends on the prior knowledge of the user. If the user is familiar with the data and knows which feature-blocks are rather important and which are not, then the priority-Lasso adaptions can be recommended. If the user is unfamiliar with the data and does not know which feature-blocks are important and which are not, then the random forest based adaptions can be recommended.

Contents

1 Introduction

On October 1, 1990 the international scientific research project named *Human Genome Project* was launched, with the aim to sequence the first complete human genome ever [5]. After investments of totally $2.7 billion and 13 years of research, the sequencing was officially finished in 2003 [6]. Since then, on the one hand, there have been biomedical advances that have led to the identification of disease genes which in turn have led "to improved diagnosis and novel approaches in therapy" [[7], p. 14]. On the other hand there has been an "extraordinary progress [...] in genome sequencing technologies" [[8], p. 333] leading to a sharp drop in sequencing prices. Nowadays whole genome sequencing is available and affordable for everyone - e.g. 'Veritas Genomics' offers whole genome sequencing for ∼$700 [9].
Besides the 'genome' that carries the whole genetic material of an organism, there are also other types of '-omes', such as 'epigenomes', 'transcriptomes', 'proteomes' and 'microbiomes'. The time and costs to collect data from these different types of '-omes' have been reduced drastically ever since the completion of the Human Genome Project [[10], [11], [12], [13], [14], [15]]. The methods for "fast, automated analyses of large numbers of substances including DNA, RNA, proteins, and other types of molecules" [16] are summarized under the term 'High Throughput Technologies'. These technologies make data from molecular processes available for many patients on a large scale.
The collected data from any type of '-omes' is commonly referred to as 'omics data'. In the clinical context, it is of utmost interest to incorporate such omics data into different statistical approaches. A typical example in this context is the survival time prediction for cancer patients, where in addition to the regular clinical data 'gene expression' data has been incorporated into the survival models. This additional omics data has "often been found to be useful for predicting [the] survival response" [[11], p. 1]. In "the beginning, only data from single omics was used to build such prediction models, together or without [...] clinical data" [[17], p. 1]. The usage of multiple distinct types of '-omes' in a single prediction approach was the next logical step and coined the term 'multi-omics data'. The theoretical aspects of integrating multiple omics types into a single prediction approach and how to deal with the block-wise structures have been the topic of several papers already - e.g. [17], [18], [19], [20], [21].
This thesis deals with a special type of missing data "that is common in practice, particular in the context of multi-omics data" [3] - the so-called 'block-wise missingness'. Data with block-wise missingness consists of different folds and feature-blocks. While a feature-block stands for a collection of

associated covariates, a fold represents a set of observations with the same observed feature-blocks. In data sets with block-wise missingness, there is always at least one fold, with a missing feature-block, such that not all observations have the same observed feature-blocks.
Most statistical methods require fully observed data for their training and predictions. Data with block-wise missingness does not fulfil this requirement, such that either the approaches need methodical adjustment or the data itself needs to be processed. This foundational problem raises the following challenges and questions: How can we fit a model on the block-wise missing data, without removing observations or whole feature-blocks? Does imputation work appropriately in these settings? How does a model that uses single feature-blocks only perform in comparison? How can a model predict on observations with missing feature-blocks?
In addition to the problem of block-wise missingness, there is also the challenge of "inherent high dimensionality" [[4] p. 93], when working with multi-omics data. Data from a single omics type can easily exceed thousands of covariates, and the corresponding data sets usually consist of fewer observations than features [17]. Besides the predictive performance of an approach, it is furthermore important for the approach to be sparse. "Sparsity is [...] an important aspect of the model which contributes to its practical utility" [[19], p. 3], as it makes the model much more interpretable than models including several thousands of variables.
A method that handles high dimensional data, even if the number of observations is lower than the amount of features, is the random forest method [17]. The method additionally handles different input types, does not need a lot of tuning and yields comparable predictive performances [22]. The only drawback is that it is not as interpretable as "models yielding [*in*] coefficient estimates of few relevant features" [[17], p. 35], as penalised regression approaches for example. Nevertheless, variable importance measures can be extracted with the random forest method, as well as partial dependencies. Furthermore it has already been used successfully in various articles dealing with multi-omics data - e.g. [17], [18]. Moreover, there have been proposals by Hornung et al. [3] and Krautenbacher [4] that modify the random forest approach, such that it can directly handle data with block-wise missingness. The different adaptations of penalised regression, as the priority-Lasso [19] can also be modified so they can directly deal with block-wise missing data. The theoretical aspects of these approaches are only briefly explained, while a closer explanation is in Hagenberg's thesis [1]. Nevertheless, the performances of the different random forest approaches and penalised regression adaptations are compared in this thesis as well.

Even though the problem of block-wise missingness is common in multi-omics data, there are, to my knowledge, no comparison studies of such prediction approaches yet. Krautenbacher has already stated that "reliable analysis strategies for multi-omics data [...] [with block-wise missingness are] urgently needed" [[4] p. 94]. The thesis at hand aims to provide such a large scale comparative study of prediction approaches capable of dealing with block-wise missingness and shall help to find a reliable analysis strategy.
This paper compares the predictive performance of two naive random forest approaches, a random forest approach on imputed data, two random forest adaptations, the mdd-sPLS method and the adaptions of penalised regression on data with block-wise missing values. In the second chapter, firstly the term 'block-wise missingness' is defined in more detail and how it can arise in multi-omics data. Then a theoretical explanation of the random forest method for classification is given. Following three data processing approaches are explained - these process the block-wise missing data such that a regular random forest can be trained with it. Moreover, two methodological adaptations of the random forest method are illustrated. These adaptations allow the random forest approach to deal with block-wise missing data directly. The first part of the third chapter covers general information on the used metrics and evaluation techniques, while the second part introduces the different data sources and corresponding data sets. These data sets are then used to validate the performances of the various approaches. In the penultimate chapter, all methods are analysed, and the predictive performance is compared. The last section of this thesis discusses all findings, draws a conclusion and gives an outlook.

2 Methods

This section deals with the theory of the random forest model and the different adaptions of it to handle data with block-wise missingness.
In the beginning, block-wise missingness is defined in more detail, and it is shown how it can arise in multi-omics data. Afterwards, the theory of the random forest method for classification is illustrated. Subsequent three approaches that process the data with block-wise missingness, such that a regular random forest can be fit on them, are described. The last two sections of this chapter present two different adaptations of the random forest method. These adaptions enable the random forest method to deal with block-wise missing data directly.

2.1 Block-wise missingness

Collecting omics data has become significantly cheaper and faster ever since the completion of the Human Genome Project. As a result, this type of data is used more and more frequently in the biomedical research - e.g. risk prediction of childhood asthma [4]. Even though the integration of multiple types of '-omes' into a single prediction approach seems promising, there are still challenges to face. One of these challenges is a special type of missingness that is common in the context of multi-omics data, the so-called block-wise missingness [3].
The term block-wise missingness needs to be defined in more detail before clarifying how it can arise in multi-omics data. Table 1 shows a minimalist example for a data set with block-wise missingness, whereby the data consists of eight observations, 105 covariates and the binary response variable Y. While the covariates 'weight', 'height', 'income' and 'education' are pretty much self-explanatory, the features 'g_1', ..., 'g_{100}' could be any type of omics data. Data with block-wise missingness always consist of different blocks and folds. On the one hand, a **block** describes a set of covariates containing all features collected based on a characteristic - basically all covariates that are related in content. The data in table 1 has three blocks in total. 'Block 1' consists of the variables 'weight' and 'height' representing the physical properties. 'Block 2' contains the variables 'income' and 'education' standing for economic properties. 'Block 3' includes the remaining variables 'g_1', ..., 'g_{100}' that are measurements from a single omics type and represent genetic properties. On the other hand, a **fold** represents a set of observations with the same observed feature-blocks - basically all observations with the same observed features. The data set in table 1 consists of three folds in total.

'Fold 1' holds the observations 1, 2 and 3, as these have the same observed feature-blocks ('Block 1' & 'Block 2'). 'Fold 2' holds the observations 4 and 5, while 'Fold 3' consists of the remaining observations 6, 7 and 8. As each fold has different observed feature-blocks, each fold is unique, and every observation belongs to exactly one of them. The only variable all folds must have in common is the target variable.

ID	$weight$	$height$	$income$	$education$	g_1	$\cdots$	g_{100}	Y	
1	65.4	187	2.536	*Upper*				1	Fold1
2	83.9	192	1.342	*Lower*				0	
3	67.4	167	5.332	*Upper*				1	
4			743	*Lower*	-0.42	$\cdots$	1.43	1	Fold2
5			2.125	*Lower*	0.52	$\cdots$	-1.37	0	
6	105.2	175			-1.53	$\cdots$	2.01	0	Fold3
7	71.5	173			0.93	$\cdots$	0.53	0	
8	73.0	169			0.31	$\cdots$	-0.07	1	
	$Block1$		$Block2$		$Block3$				

Table 1: A data set with block-wise missingness - consisting of three feature-blocks, three folds and the binary target variable 'Y'.

Multi-omics data with block-wise missingness have a structure as displayed in table 1, but the single feature-blocks are usually much higher dimensional than in the given example. When working with multi-omics data this type of missingness is a common problem. There are two main reasons for this:
The first one is related to the costs of collecting omics data. Even though the costs have been reduced drastically over the last 15 years, collecting omics data is still more complex and expensive than obtaining standard clinical data, as for example 'weight', 'height' and 'smoking status'. As a consequence, omics data can not always be collected for all participants of a study. Therefore participants from the same study can end up with different observed feature-blocks, such that the data for the whole study contains block-wise missingness.
The second reason is related to the collection of data sets from different sources - e.g. various hospitals. Even though the different sources do research regarding the same response variable the surveyed feature-blocks can still differ. Therefore the concatenation of such data sets can result in a data set with block-wise missingness. This scenario is illustrated in figure 1.
In the top of the figure the three different data sources are displayed - 'Hospital 1', 'Hospital 2' and 'Hospital 3'. Each source consists of the target

variable 'Y' and two feature-blocks as covariates - e.g. 'Hospital 2' consists of the target variable 'Y' and the feature-blocks 'RNA' and 'Clinical'. The feature-blocks 'RNA', 'miRNA' and 'CNV' represent high dimensional omics data, while the 'Clinical' feature-block stands for several clinical features. Even though the target variable 'Y' is the same for all data sources, the observed feature-blocks still differ. The concatenation of the data sets results in data with block-wise missingness and is displayed in the bottom of figure 1. In the concatenated data an observed block is marked with a green tick and a missing block with a red cross. The fold 'Hospital 2' only has 'RNA' and 'Clinical' as observed feature-blocks, such that the observations from this fold miss all the features from the blocks 'CNV' and 'miRNA'. The concatenated data consists of three unique folds and four different feature-blocks.

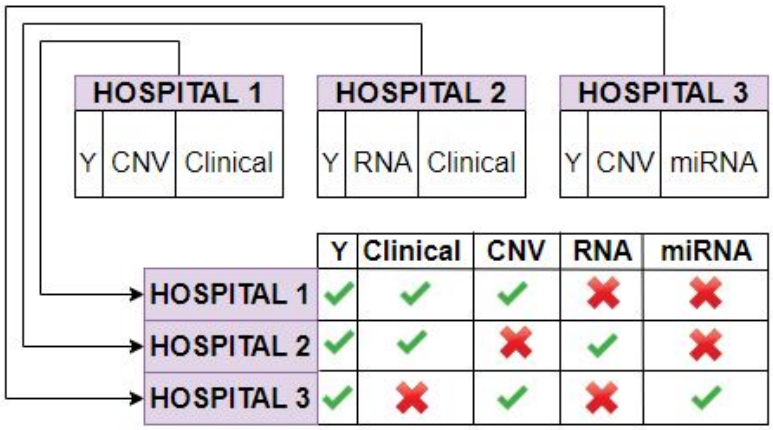

Figure 1: Block-wise missingness, when concatenating data from diverse sources.

Training a prediction model is for most approaches not directly possible on data with block-wise missingness. Either the methods have to be adopted or the data processed. As block-wise missingness can also affect the test data, it raises the following question. How can a model predict for an observation that misses feature-blocks the model has been trained with? This challenge has to be taken into account when proposing methods capable of dealing with block-wise missingness.
The remaining sections in this chapter focus on the approaches and adaptations of the random forest method. Firstly the concept of the random forest for classification is explained and then the different approaches and adaptions to handle data with block-wise missingness.

2.2 Random Forest

This chapter illustrates the random forest method that has already been applied in several articles dealing with multi-omics data [[17], [18], [19]]. It is a "powerful prediction method [...] able to capture complex dependency patterns between the outcome and the covariates" [[18] p. 2]. Furthermore, it does not need a lot of tuning and naturally handles high-dimensional data with more covariates than observations [17]. The random forest method can be applied to classification-, regression- and even survival-problems. Latter was added in 2008 by Ishwaran et al. [23]. As this thesis focuses on classification tasks, only the random forest for classification is explained. Nevertheless, all of the approaches and adaptions described in sections 2.3 to 2.7 can also be applied in regression- and survival-problems.
The random forest model is a tree-based ensemble method that was introduced by Breiman in 2001 [24]. An ensemble is a concept from machine learning that "train[s] multiple models using the same learning algorithm" [25]. Therefore an ensemble consists of η identical so-called 'base learners'. The base learner of the random forest method is a 'decision tree'. The decision tree is an excellent base learner for an ensemble, as it can capture complex interactions and has a relatively low bias if grown sufficiently deep. Especially as single decision trees are known to be noisy, they benefit from the ensemble [22]. Since decision trees are the basis of the random forest method, it is crucial to understand how these work to understand the random forest method properly.

2.2.1 Decision Tree

A decision tree is a supervised learning method that was introduced by Breiman et al. in 1984 [26]. It has a hierarchical nature, is easy to interpret and non-model based [27]. It applies recursive binary splitting to "partition the feature space into a set of rectangles" [[22] p. 305], such that the resulting squares are as pure as possible in terms of the target variable. A prediction is generated by assigning an observation to one of the rectangles in the partitioned feature space. The prediction then equals the distribution of the target variable within the assigned rectangle. E.g. an observation that falls into a rectangle with three negative and seven positive responses has predicted a probability of 70% for a positive response.
To partition the feature space into the purest rectangles possible the algorithm iterates over all possible split variable/ split value combinations. For each of these possible splits, the observations from the parent node N are divided - with respect to the split variable x_j at split point t - into the

child nodes N_1 and N_2 [[28], p. 10]:

$$N_1(x_j, t) = \{(x, y) \in N : x_j \geq t\} \tag{1}$$
$$N_2(x_j, t) = \{(x, y) \in N : x_j < t\} \tag{2}$$

N_1 contains all observations from the parent node N with $x_j \geq t$, while N_2 contains all observations from the parent node N with $x_j < t$. The point (x_j, t) therefore creates a binary split and partitions the data from the parent node N in the two subspaces N_1 and N_2. The split variable x_j and split point t are chosen such that the resulting child nodes N_1 and N_2 have the greatest possible purity [28]. To measure the impurity of a node N regarding a categorical response with g classes the 'Gini-Index' (3), 'Misclassification-Error' (4) or 'Shannon-Entropy' (5) can be used [[28], p. 12]:

$$I(N) = \sum_{k=1}^{g} \hat{\pi}_{k,N} \cdot (1 - \hat{\pi}_{k,N}) \tag{3}$$
$$I(N) = 1 - \max_k \hat{\pi}_{k,N} \tag{4}$$
$$I(N) = -\sum_{k=1}^{g} \hat{\pi}_{k,N} \cdot \log(\hat{\pi}_{k,N}) \tag{5}$$

- $\hat{\pi}_{k,N}$: Relative frequency of category k in node N

For all of these impurity measures applies: The lower $I(N)$, the purer the node N and a node N is completely pure, when it only contains observations of the same response class - $I(N) = 0$. The corresponding plots of these impurity functions for a binary target variable are in the attachment in figure A-1. The reduction of the impurity when splitting the parent node N into the child nodes N_1 and N_2 is calculated by [[28], p. 10]:

$$I(N) - \frac{|N_1|}{|N|} \cdot I(N_1) - \frac{|N_2|}{|N|} \cdot I(N_2) \tag{6}$$

- $|N|$: Number of observations in the parent node N
- $|N_1|$: Number of observations in child node N_1
- $|N_2|$: Number of observations in child node N_2

This equation calculates how strong the impurity from the parent node N is reduced for a given split point that divides the observations to the child nodes N_1 and N_2. This impurity reduction is calculated for every possible split. The final split variable x_j and split point t are chosen, such that the impurity is maximally reduced.

For illustrative purposes, the single partition steps of a classification tree are displayed in figure 2. The figure consists of three plots in total, whereby each is a scatter plot of 'weight' and 'height' for the observations from 'Fold 1' and 'Fold 3' in table 1. Observations with a positive outcome are marked in blue, while negative outcomes are labelled in red.
In the very beginning, all observations are in the same feature space that has not been divided yet 'N1' - the so-called 'root node'. This situation is displayed in the leftmost plot of figure 2. The node contains three observations with a positive and three with a negative response - hence the class distribution in this node is 50|50. The node is not pure regarding its responses and all possible impurity measures [(3), (4), (5)] have the highest possible value. The algorithm now iterates over all features, and for each feature over all possible split points and calculates the impurity of the resulting child nodes for each of these possible splits. The split variable and corresponding split value are chosen, such that the impurity reduction according to equation (6) is maximised. In the example of figure 2 the first split variable is chosen as 'weight' with the split value 69. Therefore the data from the root node - 'N1' - is split into the two child nodes 'N2' and 'N3' - central plot in figure 2. 'N2' contains the observations with weight ≥ 69, while 'N3' consists of the observations with a weight < 69. The distribution of the target variable in 'N2' is 25|75 and in 'N3' 100|0. Hence both resulting child nodes are purer than their parent node 'N1'. The node 'N3' only contains observations with a positive response, therefore it is completely pure and can not be split any further - all possible impurity measures [(3), (4), (5)] have the lowest possible value. The node 'N2' on the other hand, is not completely pure yet and can be split further. 'N2' is now the parent node, and the algorithm tries all possible splits on this segmented feature space. The highest impurity reduction of 'N2' is achieved with the split-variable 'height' on the value 171. 'N2' is therefore further split into 'N4' - all observations from 'N2' with a height ≥ 171 - and 'N5' - all observations from 'N2' with a height < 171. As well 'N4' as 'N5' are completely pure and the impurity of these nodes can not be reduced any further. The final partitioned feature space is displayed on the rightmost plot in figure 2. Based on this final partitioned feature space predictions can be done by assigning observations to one of the segments in the feature space. An observation with weight = 90 and height = 185 for example falls into the segment 'N4' and has a predicted class probability of 100% for response class 0 then.
Hence the decision tree algorithm splits the feature space, such that the resulting child nodes maximally gain purity regarding the target variable. This is done with an exhaustive search, trying all possible split variables and corresponding split points.

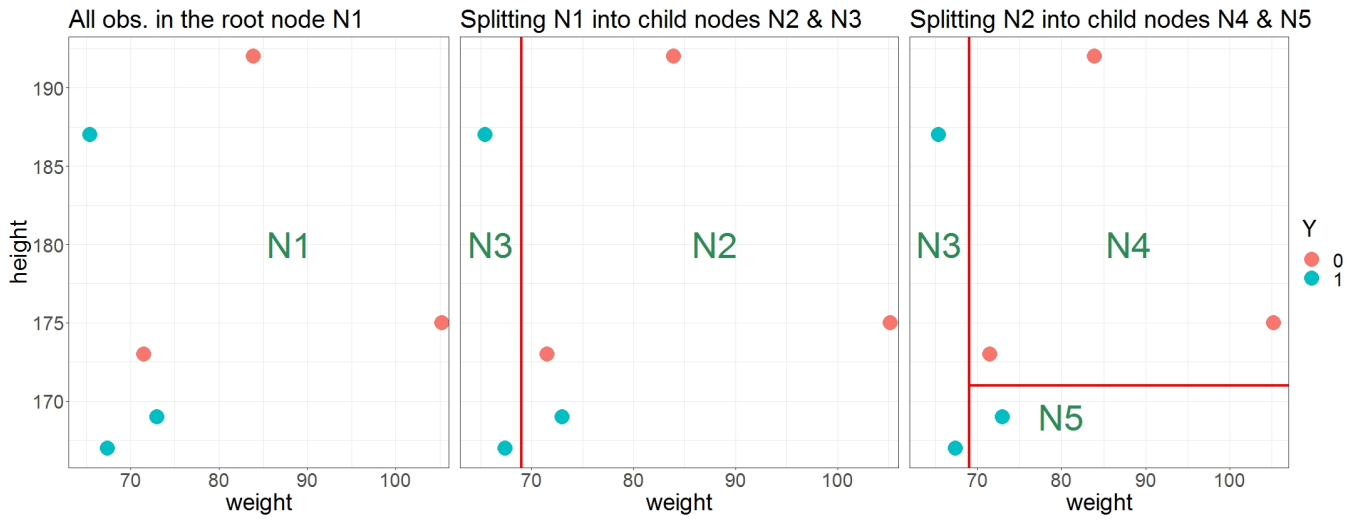

Figure 2: Recursive binary splitting of a decision tree on a two-dimensional feature space.

A convenient property of a decision tree consists of its natural graphical display, which makes it extremely easy to interpret - even for people without a mathematical background. This visualisation is especially useful when the training data for the decision tree holds more than two covariates and can not be displayed as scatter plot [27].
The segmentation of the feature space from figure 2 is displayed as a graphical decision tree in figure 3. Each square in the figure represents a node of the decision tree. Each of these nodes display the response class with the highest proportion (top), the distribution of the response classes (mid) and the fraction of observations they contain (bottom). The split variables and split values are displayed below each node - nodes without a split variable/ value are so-called 'terminal nodes'. The prediction for a test observation with figure 3 is straightforward and intuitive. The test observation is passed down the decision tree until it reaches a terminal node. This is shown for an observation with weight = 90 and height = 185. The first node splits on the variable 'weight' with the value 69. As the test observation has a weight $\geq$ 69, it is sent down to the left child node. The next node splits on the variable height with the value 171. As the test observation is taller than 171cm, it is sent to the left child node. The observation is then in the node on the leftmost in the bottom of figure 3. This is a terminal node and can not be divided any further. The distribution of this node equals 100|0, and the prediction for the observations is, therefore, class 0, with a probability of 100%.

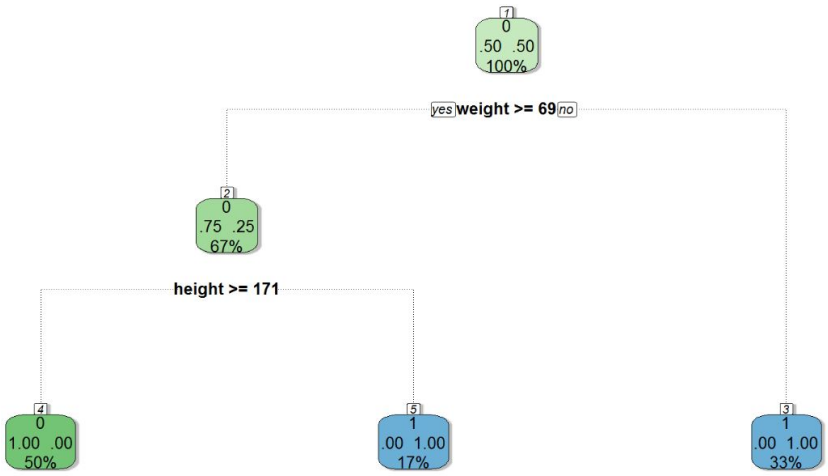

Figure 3: Corresponding decision tree for the segmented feature space on the rightmost plot in figure 2.

The complexity of a decision tree grows with the number of used splits and resulting terminal nodes [28]. The more complex a decision tree, the higher the chances of overfitting, but a tree with not enough complexity "might not capture the important structure[s]" [[27], p. 20]. So when should a tree stop with the binary partition of the feature space? There are multiple stopping criteria to control this, whereby the two most commonly used arguments are [27]:

- MinSplit:
  "The minimum number of observations that must exist in a node in order for a split to be attempted" [[29], p. 22]

- Complexity:
  "Split tree nodes only if the decrease in impurity due to splits exceeds some threshold" [[27], p. 20]

Both arguments have a considerable impact on the complexity of a decision tree, as they control when the tree stops the partition of the feature space. The 'MinSplit' argument forces the tree to stop the partition, as soon as a potential parent node contains less than 'MinSplit' observations. The higher this argument, the earlier has the tree to stop growing and hence the less complex the resulting tree. 'Complexity', on the other hand only allows splits that lead to a decrease of impurity of a given threshold when splitting the parent node to its child nodes. The drawback of this argument is that it is rather short-sighted, as a "seemingly worthless split might lead to a very good split below" [[27], p. 20]. Hence the 'MinSplit' argument is the preferred

argument to control the complexity of a decision tree.
The advantages of the decision tree method are numerous. It is easy to interpret, has no problems with outliers, captures interaction effects between features, handles categorical features and scales well with larger data [27]. Besides all these advantages, unfortunately, there is also a huge disadvantage. A decision tree is highly unstable meaning that "small changes in the data could lead to completely different splits, thus, to a completely different tree" [[28], p. 26]. Even the removing of a single observation/ feature from the training data can lead to a completely different decision tree.
The next chapter explains, how this alleged disadvantage of high instability of a decision tree is exploited by the random forest method to create better predictions based on multiple decision trees.

2.2.2 Random Forest Model

As already mentioned at the beginning of this chapter, the random forest model is an ensemble method that uses the decision tree as a base learner. The random forest model, therefore, consists of multiple decision trees. To train the diverse decision trees of a random forest model, the random forest method uses a modified version of bagging that was initially proposed by Breiman in 1996 [30]. As bagging is an essential component of the random forest method, it is explained in more detail now.
To train M base learners on a single data set, each base learner needs to be fit on a modified data set, else all resulting learners are completely identical. To generate a different data set for each of the M base learners, bagging - short for **B**ootstrap **Agg**regation - is applied to the original data. It is a "type of resampling where large numbers of [...] samples of the same size are repeatedly drawn, with replacement, from a single original sample" [31]. The bootstrapping therefore generates M different bootstrap samples of the original data and trains any base learner B on these M bootstrapped data sets. To obtain a prediction, each of the M fitted base learners is asked for a prediction - $B_m(x)$. These M different predictions are then aggregated for a final prediction: $B(x) = \frac{1}{M} \sum_{i=1}^{M} B_m(x)$ [[32], p. 4]. Bagging works best for learners with a high variance - e.g. a decision tree - as it reduces the variance of the base learner and only increases the bias in return [32].
The random forest method uses a slight modification of the bagging algorithm to construct bootstrapped decorrelated decision trees [22]. To do so, the random forest algorithm does not only fit each decision tree on a separate bootstrapped data set but decreases the correlation of these as well by randomly drawing 'mtry' features as possible split candidates at each split point instead of having all 'p' features as possible split candidates [32]. The

standard value for 'mtry' with a categorical response class is $\lceil\sqrt{p}\rceil$, whereby p equals the number of covariates in the data. [33]. Hence at each node of a decision tree, only a subset of the available features are drawn as possible split variables. Therefore every single decision tree has a different set of possible split variables for each of their nodes. This modification of the original bagging algorithm ensures that the trees are grown more diversely, and the resulting trees are less correlated as with the regular bagging algorithm. The modified bagging algorithm to fit a random forest is the following [[22], p. 588]:

Algorithm 1: Growing a random forest

Input : D $\leftarrow$ data with n observations & p features
M $\leftarrow$ number of trees in the forest
n_{min} $\leftarrow$ 'MinSplit' argument of a decision tree
mtry $\leftarrow$ number of variables to draw at each split

for $m \leftarrow 1$ **to** M **do**

1. Draw a bootstrap sample **Z*** of size 'n' from 'D';

2. Based on **Z*** grow a decision tree, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached;

2.1 Randomly draw 'mtry' of the 'p' available variables;

2.2 Pick the best splitting point among the 'mtry' variables;

2.3 Split the node into two daughter nodes;

The procedure to receive a prediction from a random forest model is the same as in the original bagging algorithm. The input x is passed to each decision tree in the random forest model and each of these trees creates a prediction - details in section 2.2.1. The final prediction of a categorical response can either be the average of the M predicted class probabilities or the label that was predicted by the majority of the trees.

2.2.3 Out-of-bag error

A convenient property of the random forest method is the so-called out-of-bag error (OOB error). The random forest model consists of multiple decision trees, whereby the data for each of these decision trees are obtained by drawing observations with replacement from the original data. For each tree, the average probability for an observation not to be drawn is ~ 0.37 [[32], p. 12]:

$$P(\text{Obs. not drawn}) = (1 - \frac{1}{n})^n \quad \xrightarrow{n \to \infty} \quad \frac{1}{e} \approx 0.37 \tag{7}$$

- n: Amount of observations in the data

The observations that are not used to grow the decision tree can be used to estimate its predictive performance - the so-called OOB error. Before explaining the OOB error, let's have a look at figure 4. The figure displays the M different decision trees of a random forest model that were originally supplied with data of n observations and p features. Under each tree the data used for growing is displayed - a pink background indicates that an observation was 'in-bag' and hence used in the training of the decision tree. In contrast, a grey background means that an observation is out-of-bag and was not used in the training of the decision tree. It should be noticed that the observations are drawn with replacement so that an observation can enter the in-bag samples more than once.

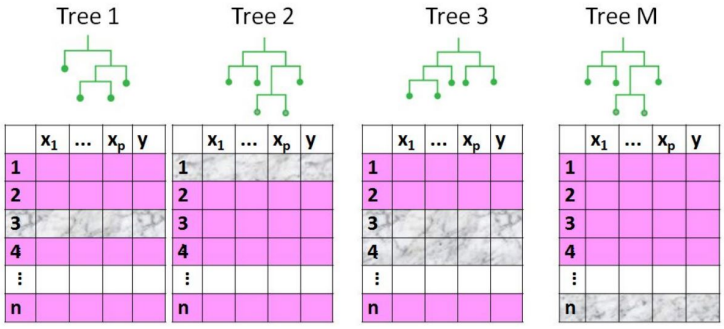

Figure 4: The data used to grow the M different decision trees of a random forest. Below each decision tree the in-bag observations are labelled in pink, while out-of-bag observations are labelled in grey [[32], p. 13].

To receive the OOB error of a random forest model, each of the M decision trees is asked for a prediction for the current observation i. Only those trees that have observation i as an out-of-bag observation create a prediction. This results in $\psi_i \leq M$ predictions for the observation i, whereby the final out-of-bag estimation for observation i equals the average of the ψ_i predictions. After receiving this out-of-bag prediction for all n observations, the final OOB error of the random forest model can be calculated. To compare the predicted classes and the true response class of the n observations, any metric

can be used - e.g. Accuracy, F-1 score. The OOB error "is almost identical to that obtained by N-fold cross-validation" [[22], p. 593]. Therefore, unlike most other prediction models, the random forest can be fit and evaluated in one single step - an extremely handy property.

2.2.4 Variable importance

In most applications, not all feature variables are equally important, and mostly only a few have a relevant influence. Therefore the property of variable importance in any prediction method has high practical usage. Even though the single decision trees of a random forest model are highly interpretable, the random forest model itself "lose[s] this important feature, and must, therefore, be interpreted in a different way" [[22], p. 367].
One possibility to measure the variable importance in a random forest model is based on the permutations of the out-of-bag observations. For each decision tree T_m, the corresponding out-of-bag observations get a prediction and the accuracy of a decision tree is calculated - $acc_{\text{m, without permutation}}$. To obtain the importance of a variable x_l, the out-of-bag observations of a decision tree T_m are permuted in the variable x_l, such that all out-of-bag observations receive a different value for the variable x_l. For each decision tree T_m, the corresponding permuted out-of-bag observations get a prediction and the accuracy of a decision tree is calculated again - $acc_{\text{m, with permutation } x_l}$. The difference between the regular OOB accuracy - $acc_{\text{m, without permutation}}$ - and the OOB accuracy with permuted variable x_l - $acc_{\text{m, with permutation in } x_l}$ - is used as measure for the importance of the l-th variable in the decision tree T_m. The average importance of the l-th variable of the M decision trees equals the variable importance for x_l for the whole random forest model [32]. This technique to access the importance for the different variables is displayed in figure 5 for the variable x_1. For the decision trees 1 and M the data used to train these is displayed below. A grey background marks the out-of-bag observations. Based on these observations the out-of-bag accuracy can be calculated for each of the M trees - this results in $acc_{\text{m, without permutation}}$ for each tree. Then the values of the variable x_1 are permuted for the out-of-bag observations in each decision tree. Following the out-of-bag accuracy is calculated with the permuted variable x_1 resulting in $acc_{\text{m, with permutation in } x_1}$ for each tree. The difference diff_m between $acc_{\text{m, with permutation in } x_1}$ and $acc_{\text{m, without permutation}}$ represents the importance of variable x_1 in the decision tree T_m. The final importance of variable x_1 then equals the average over all these differences [[32], p. 16]: $\frac{1}{M}\sum_{i=1}^{M}\text{diff}_i$

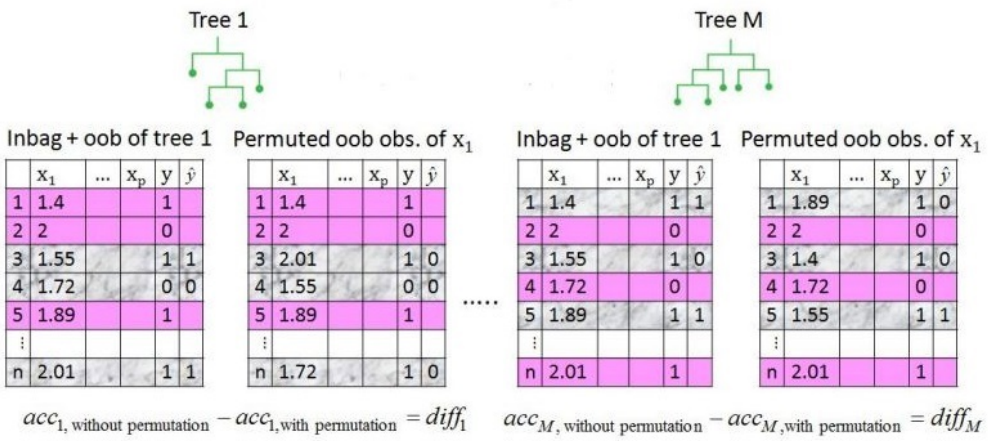

Figure 5: Calculation of the variable importance of x_1 for a random forest model consisting of M decision trees [[32], p. 16].

2.3 Complete-Case Approach

In this section, the first baseline approach to handle data with block-wise missingness is explained - the so-called 'Complete-Case' approach. This approach does not modify the random forest model itself, but processes the training data, such that it does not contain any missing values afterwards. This processing has the advantage that every prediction model - e.g. a random forest model - can be trained regularly on the processed data and the disadvantage of not using all available folds and feature-blocks. Therefore it is a rather simple approach and shall serve as a first baseline. The results from this first baseline approach are a hurdle to overcome for the more sophisticated methods from the sections 2.5 - 2.7.
Let's have a look at the approach itself. As block-wise missingness can affect the test data as well as the training data, the test observations may be missing feature-blocks - even if these are available in the training data. The 'Complete-Case' approach removes all folds from the training data that miss at least one of the available feature-blocks from the test data. The feature-blocks of the training data that are not available in test data are removed as well. After the processing, the training data only consists of the feature-blocks that are available in the test data and only of observations that are completely observed in these feature-blocks. Based on this processed training data, a random forest can be trained regularly. The prediction on the test observations with such a fitted model can then be done completely regular, as the model does not use any split variables that are not available for the test observations. To make the processing of the training data easier to un-

derstand two examples are shown in figure 6 and 7. In these examples, the concatenated data with block-wise missingness from figure 1 is used as an exemplary training data:

1. Example: This example is displayed in figure 6. The test-set is shown in the top of the figure and only has two observed feature-blocks - 'Clinical' and CNV'. The train-set is displayed below and consists of four feature-blocks and three unique folds. The 'Complete-Case' approach processes the training data, such that it removes all observations that miss at least one of the available feature-blocks of the test-set. Therefore only observations from the fold 'Hospital 1' can be used, as all other folds either miss the feature-block 'Clinical' or 'CNV'. The fold and feature-blocks that can be used for the model fitting are marked with a green box. On this processed data, a regular random forest model can be trained and used to create predictions for the test observations then. The processed training data contains two feature-blocks and two folds less than the original training data, as these were removed by the processing of the 'Complete-Case' approach.

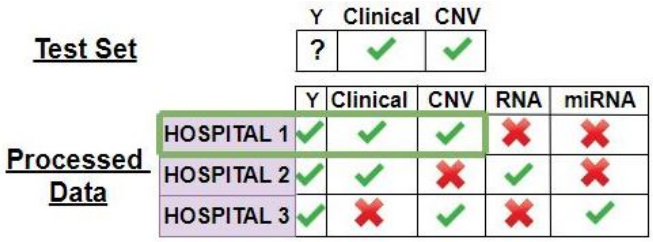

Figure 6: The 'Complete-Case' processing of the training data according to the available feature-blocks in the test-set.

2. Example: This example is displayed in figure 7. The available train-set is displayed at the bottom of the figure and consists of four feature-blocks, while the test-set is only observed in the feature-block 'CNV'. The 'Complete-Case' approach removes all folds from the train-set that do not have an observed 'CNV' feature-block. Therefore only the observations from the folds 'Hospital 1' and 'Hospital 3' can be used as training data. For these folds, only the feature-block 'CNV' can be used for training, and all other feature-blocks are discarded. The folds and feature-block that can be used for the model fitting are marked with a green box. On this data, a regular random forest model can be trained and used to create predictions for the test observations then. As in the example before, the 'Complete-Case' approach discards much of the original training data.

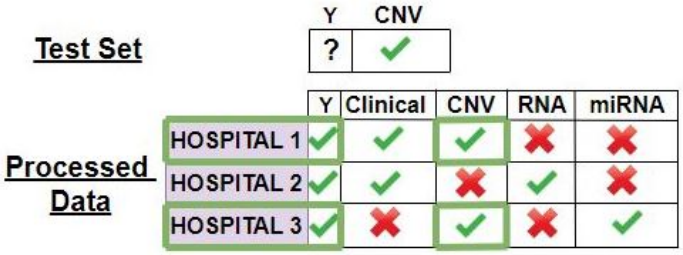

Figure 7: The 'Complete-Case' processing of the training data according to the available feature-blocks in the test-set.

Besides the generous discarding of training data, the method has another disadvantage. As the 'Complete-Case' approach removes all observations from the train-set that miss at least one of the available feature-blocks of the test-set, there may be no training observations left after the processing of the train-set. In situations like this, the 'Complete-Case' approach can not provide predictions for the test-set then.

In summary, the 'Complete-Case' approach removes all observations from the train-set that miss at least one of the observed feature-blocks in the test-set. Also, all feature-blocks from the train-set that are not available in the test-set are removed. This data processing approach can discard a big part of the original train-set and hence it does not handle the data very efficiently.

2.4 Single-Block Approach

The second baseline approach to handle data with block-wise missingness is the 'Single-Block' approach. As well as the 'Complete-Case' approach, it does not modify the random forest model itself but processes the training data, such that it does not contain any missing values afterwards. As well as the 'Complete-Case' approach, the 'Single-Block' approach discards much of the available training data. As this approach is rather naive, it is the second baseline approach. It shall serve as another lower limit for the performances of the more sophisticated methods from the following sections 2.5 - 2.7.
As the name of the approach already suggests, it only uses a single feature-block to train a random forest model and predicts on the test-set then. The model must be trained with a feature-block that is available in the test-set to create predictions for the test-set. Else the fitted model can not predict on the test-set, as it uses split variables that are not available for the observations in the test-set. Hence the single feature-blocks from the training data

that can be used to train a model depend on the observed feature-blocks in the test-set. The concept of this approach is now explained with the example in figure 8. The training data in this example has already been introduced in section 2.1 and has been used as an example in the previous section as well:

Example: The test-set for this example is displayed in the top of figure 8 and contains two different feature-blocks - 'Clinical' and 'CNV'. The training data consists of four different feature-blocks and three different folds in total. The 'Single-Block' approach processes the training data in multiple ways to get rid of the block-wise missingness in the train-set. For each available feature-block in the test-set, it is checked, whether the train-set involves the feature-block as well. For each feature-block that the test and train-set have in common, a separate random forest model is fitted and used to predict the outcome of the test observations. In the current example, it is firstly checked whether the training data involves a 'Clinical' or a 'CNV' feature-block. In this example, the training data involves both feature-blocks of the test-set. For each of the feature-blocks, the test- and train-set have in common a separate processed data set is created.

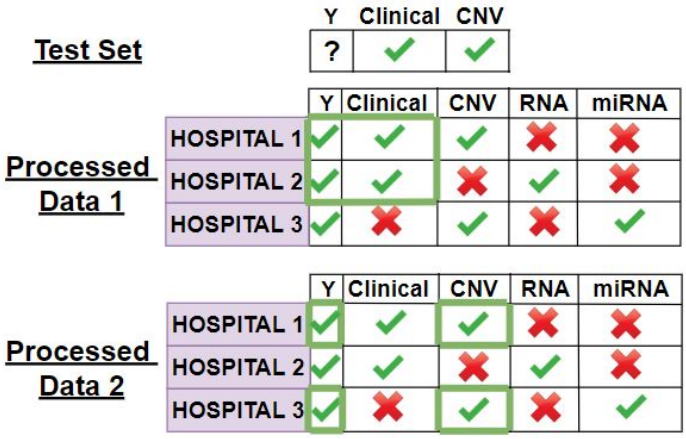

Figure 8: 'Single-Block' processing of the training data so a random forest model can be regularly trained with each of these processed data sets.

Processed Data 1: As the test- and train-set have the feature-block 'Clinical' in common, the first processed training data only consists of the response Y and the feature-block 'Clinical' for the observations that have been observed in this block. This subset of the data is displayed in the middle of figure 8 and marked with a green box. Based on this subset, a random forest model can be fit regularly and used to create predictions for the test-set - for the predictions, only the features from the 'Clinical' feature-block are used.

Processed Data 2: As the test- and train-set also have the feature-block 'CNV' in common, another processed train-set is created. This processed training data only consists of the response Y and the feature-block 'CNV' for the observations that were observed in the 'CNV' block. This subset of the data is displayed in the bottom of figure 8 and marked with a green box. Based on this data, a random forest model can be regularly fit and used to create predictions for the test-set. For the predictions on the test-set, only the features from the 'CNV' feature-block are used.

Predictions: As the processing of the training data with the 'Single-Block' approach results in two processed train-sets, this approach consists of two different fitted models then. One random forest model was fitted on the 'Clinical' feature-block and the other one on the 'CNV' feature-block. Both of the fitted models can create predictions for the test-set based on the features they have been trained with. The 'Single-Block' approach can, therefore, result in multiple predictions for the observations in the test-set.

In summary, the 'Single-Block' approach creates an own processed train-set for each of the feature-blocks the test- and train-set have in common. Each of the resulting processed train-sets consists of only one single feature-block and do not contain any missing data, as the observations with missing values are removed. On each of these processed train-sets, a random forest model can be trained and used for predictions on the test-set. As the 'Complete-Case' approach, the 'Single-Block' does not handle the data very efficiently.

2.5 Imputation Approach

This section introduces an approach that deals with block-wise missingness by imputing the missing values - the so-called 'Imputation' approach. Other than the 'Complete-Case' and 'Single-Block' approach, the 'Imputation' approach does not discard any of the available training data and hence uses the data more efficiently. After the missing values in the training data have been imputed, any prediction model can be fit on this data in a regular way and provide predictions for a test observation based on a single feature-block or based on multiple different feature-blocks.
"Many established [prediction] methods [...] require fully observed datasets without any missing values" [[34], p. 112] - in data with block-wise missingness this requirement is clearly not fulfilled. The idea to deal with this missingness by imputing the missing values seems natural. The data set does not contain any missing values after the imputation, such that a prediction method can be fitted regularly then. There are two big drawbacks when imputing missing

values in multi-omics data. Firstly, multi-omics data with block-wise missingness can consist of "many missing values making imputation techniques unreliable" [3]. Secondly, if the data is a concatenation of data sets from different sources, the imputation is "performed across different, potentially heterogeneous data sets" [3] - another reason for the unreliability of the imputation. Despite these disadvantages, the 'Imputation' approach is still worth being compared to the other approaches in this study.
Firstly a suitable imputation approach has to be found. This is not trivial, as multi-omics data usually do not only have fewer observations than features but also a mixture of continuous and categorical variables as features [34]. Furthermore "such datasets often contain complex interactions and non-linear relation structures which are notoriously hard to capture" [[34], p. 112]. The 'k nearest neighbours' imputation method [35] requires at least one complete case observation, an assumption that is not always true for multi-omics data with block-wise missingness. The 'Amelia' imputation [36] "assumes the data is distributed multivariate normal" [[37], p. 7]. Most multi-omics data sets do not fulfil this assumption and need a transformation to fulfil it - a not very handy property with such high-dimensional data [37]. There are more imputation methods, but most of these "are restricted to one type of variable" [[34], p. 112] or "make assumptions about the distribution of the data" [[34], p. 112]. An imputation method that can handle any type of feature variables and makes as few as possible assumptions about the data is based on the random forest method - the so-called 'MissForest' [34]. This imputation approach needs "no tuning parameter, and hence it is easy to use and needs no prior knowledge about the data" [[34], p. 113]. Additionally it was shown that the 'MissForest' approach is competitive to the 'k nearest neighbours' and 'MICE' imputation [34]. Furthermore the 'MissForest' imputation method can handle "mixed-type data and is known to perform very well under barren conditions like high dimensions, complex interactions and non-linear data structures" [[34], p. 113]. Because of all these mentioned advantages, the 'MissForest' method is used as the imputation method in this thesis. It is explained in more detail in the following paragraph.

MissForest: The 'MissForest' imputation method was proposed by Stekhoven and Bühlmann in 2012 [34] and builds upon the random forest method. For the imputation of missing values, a random forest is trained on the observed parts of the data and then used to predict the missing values in the data. For the explanation, assume D to be a $n \times p$ dimensional data set with missing values in the diverse variables. For the imputation of a variable X_j with missing values at the entries $i_{mis}^{(j)} \subseteq \{1, \ldots, n\}$ the data set D is separated into four parts [34]:

1 y^j_{obs}: Observed values of variable X_j

2 y^j_{mis}: Missing values of variable X_j

3 x^j_{obs}: Variables other than X_j with observations $i^{(j)}_{obs} = \{1, \ldots, n\} \backslash i^{(j)}_{mis}$
typically not fully observed as the index $i^{(j)}_{obs}$
corresponds to the observed values in X_j

4 x^j_{mis}: Variables other than X_j with observations in $i^{(j)}_{mis}$
typically not completely missing as the index $i^{(j)}_{obs}$
corresponds to the observed values in X_j

The imputation procedure is explained and shown in algorithm 2 - algorithm and explanation are based on [34]:
In the beginning, all missing values in the data set D are imputed with an initial guess - e.g. mean imputation. In the next step, the variables with missing values are ordered accordingly to the number of missing values - starting with the variable with the fewest missing values. For each of these variables, a random forest model is fitted with the response y^j_{obs} and x^j_{obs} as predictor variables. With this fitted random forest model, the missing values y^j_{mis} are imputed by the predictions of the random forest model based on x^j_{mis}. This procedure is repeated for a fixed amount of iterations or until the stopping criterion γ is met.

Algorithm 2: Imputation procedure of the 'MissForest'

Input : D $\leftarrow$ data of n observations & p features
$\gamma \leftarrow$ stopping criterion

1. Make initial guess for missing values - e.g. mean imputation;
2. **k** $\leftarrow$ vector of sorted indices of the variables in D w.r.t. increasing amount of missing values;
while not γ **do**
  1. $D^{\text{IMP}}_{old} \leftarrow$ store previously imputed data;
  for $s \in \boldsymbol{k}$ **do**
    1. Fit a random forest: $y^s_{obs} \sim x^s_{obs}$;
    2. Predict: y^s_{mis} using x^s_{obs};
    3. $D^{\text{IMP}}_{new} \leftarrow$ update imputed matrix, using predictions y^s_{mis};
  2. Update γ;
3. Return the imputed data set D^{IMP};

The stopping criterion γ measures the difference between the newly imputed data matrix D_{new}^{IMP} and the previous imputed data D_{old}^{IMP}. For continuous variables $\mathbf{N}$ the difference is calculated via [[34], p. 113]:

$$\triangle_{\mathbf{N}} = \frac{\sum_{j \in \mathbf{N}} (D_{new}^{\text{IMP}} - D_{old}^{\text{IMP}})^2}{\sum_{j \in \mathbf{N}} (D_{new}^{\text{IMP}})^2} \tag{8}$$

And for categorical variables $\mathbf{F}$ it is calculated via [[34], p. 113]:

$$\triangle_{\mathbf{F}} = \frac{\sum_{j \in \mathbf{F}} \sum_{i=1}^{n} \mathbb{1}_{D_{new}^{\text{IMP}} \neq D_{old}^{\text{IMP}}}}{\#\text{NA}} \tag{9}$$

- #NA: Number of missing values in the categorical variables

The stopping criterion γ is fulfilled as soon as "the difference between the newly imputed data [...] and the previous one increases for the first time with respect to both variables" [[34], p. 113]. An alternative to the stopping criterion γ is a fixed amount of iterations for the imputation.

Predictions: Now that is has been clarified how the 'MissForest' imputation works, the procedure of the 'Imputation' approach can be explained in more detail based on the example in figure 9. The training data has already been introduced in the section 2.1 and was used as example in the previous sections as well.
In the top of the figure, the train-set with block-wise missingness is displayed. The very first step of the 'Imputation' approach is to impute the missing values in the train-set with the 'MissForest' method. After the imputation has taken place, the train-set does not contain anymore missing values - displayed right below the original training data. As this data has no missing values at all, a random forest model can be fit regularly. But as the test-set might miss feature-blocks, the random forest model is only trained with the feature-blocks that the train- and test-set have in common. Else it might be possible that the fitted random forest model can not create predictions for the test-set, as it uses split variables that are not available for the test-set. Therefore, all feature-blocks from the train data that are not available for the test-set have to be removed. Hence the imputed train-set that can be used to train a random forest model only consists of the feature-blocks that are also in the test-set then. Based on this usable train-set a random forest model can be fit regularly and provide predictions for the test-set then.

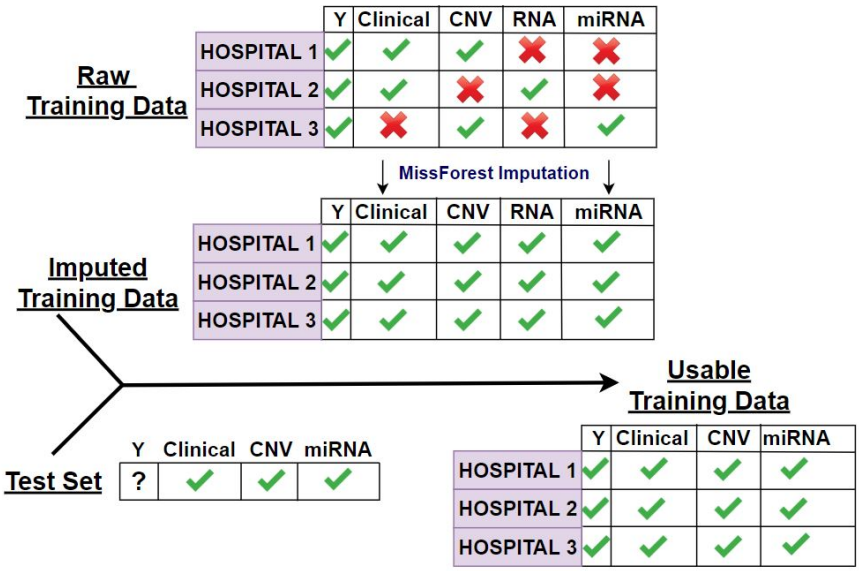

Figure 9: 'Imputation' approach to deal with block-wise missingness

Summary: With the 'Imputation' approach, the block-wise missing values in the train-set are imputed by the 'MissForest' method. After this imputation step, the train-set does not consist of missing values anymore. Based on the feature-blocks, the train- and test-set have in common a regular random forest model can be trained. This fitted random forest model can then provide predictions for the test-set.

2.6 Block-wise Approach

This section introduces the 'block-wise' approach that was initially proposed by Krautenbacher in 2018 [4]. Other than the methods from the previous sections 2.3 - 2.5 this approach does not modify the training data, but the random forest model itself. The 'block-wise' approach can directly handle block-wise missingness in the training data and does not need to process the data at all. Therefore it uses the available training data efficiently and does not discard any observations or feature-blocks. Furthermore, a 'block-wise' fitted random forest model is flexibly applicable and can provide predictions for test data that is not observed in all features from the training data.
As the name of the approach already suggests, the random forest model is fitted in a 'block-wise' manner to the training data. In the beginning, all available feature-blocks of the training data are extracted. On each of these feature-blocks, a random forest model is separately fitted. This enables "all

observations per [feature-block] [...] to be utilised for learning" [[4], p. 102] and no observation or feature-block has to be left out. With the 'block-wise' approach as many separate random forest models are fitted as the training data has feature-blocks. To create a prediction for a test observation then, each block-wise fitted random forest model is asked for a prediction. The models that were fitted on a feature-block that is not available for the test observation can not create a prediction, as these use split variables that are not available for the test observation. The remaining random forest models can create a prediction for the test observation by using the features from the test observation the models have originally been trained with. The predictions from the separate block-wise fitted models can then be aggregated to obtain a final prediction. The separate model fitting is explained in more detail with the example in figure 10. The training data in this example has already been introduced in section 2.1 and has been used as an example in the previous sections as well:

Model Fitting: The training data is displayed in the top of figure 10 and consists of four feature-blocks and three folds. To fit a separate random forest model on each feature-block, the training data needs to be split, such that each feature-block can be used to train a random forest model. This is done by merging each feature-block and the response Y to a separate train-set. In figure 10 these separate train-sets are displayed as data frames with a green background below the original training data. From each of these separate train-sets, all folds that contain missing values in the corresponding feature-block are removed - e.g. in the separate 'Clinical' train-set all observations from 'Hospital 3' had to be removed, as this hospital did not collect any clinical data. The folds that had to be removed from these separate train-sets are marked with a red horizontal line, while the usable folds are marked with a green tick. Based on each of these four different train-sets, a random forest model can be trained. This results in four distinct random forest models in total - $RF_{Clinical}$, RF_{CNV}, RF_{RNA} and RF_{miRNA}. Each of these models has been trained with a single feature-block only - e.g. RF_{RNA} was only trained with the feature-block 'RNA'.

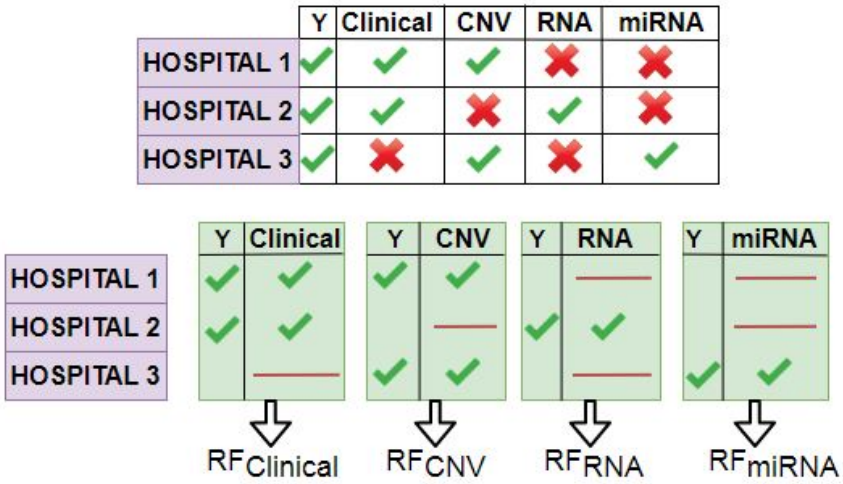

Figure 10: Training of random forest models with the 'block-wise' approach.

The 'block-wise' approach trains a separate random forest model on each of the distinct feature-blocks in the training data. It has as many separate random forest models as the training data consists of distinct feature-blocks. But how can these models be used to create a final prediction? As already mentioned, the block-wise predictions from the different random forest models need to be aggregated for a final prediction. This aggregation is explained in the following paragraph based on the example in figure 11.

Predictions: Assume that the four block-wise fitted random forest models from figure 10 can be used for the example in figure 11. The test-set is displayed at the top of figure 11. For the observations in this test-set, the outcome 'Y' needs to be predicted. Other than the training data from figure 10, the test-set only contains three feature-blocks and misses the 'CNV' feature-block from the training data. To create predictions for the observations in the test-set, each of the four block-wise fitted random forest models is asked for predictions on the test observations. As the test-set contains the feature-blocks 'Clinical', 'RNA' and 'miRNA', only the corresponding random forest models $RF_{Clinical}$, RF_{RNA} and RF_{miRNA} can provide predictions for the test-set. The random forest model RF_{CNV} can not create predictions on this test-set, as the feature-block, 'CNV' is not available. Each of the three block-wise fitted models $RF_{Clinical}$, RF_{RNA} and RF_{miRNA} create a prediction for each observation in the test-set, by only using the variables from the feature-block the models have originally been trained with. Therefore each model generates a prediction for each observation in the test-set, such

that there are three predicted outcomes for each observation - $\text{Preds}_{Clinical}$, Preds_{RNA} and Preds_{miRNA}. These predictions represent the probabilities for each of the possible response classes. The final prediction for the target variable 'Y' equals a weighted average of these predictions.

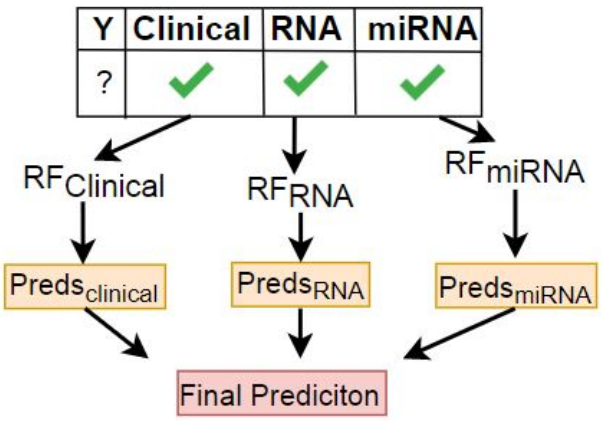

Figure 11: Prediction on test data with the 'block-wise' approach. The fitting of the random forest models was described with figure 10.

To create a meaningful weighted average of the different block-wise predictions, different techniques can be applied. The simplest method is giving each block-wise fitted model the same weight and return the simple average over all block-wise predictions. But as the block-wise fitted models have been trained on different feature-blocks, this might not always be optimal, as the models might differ sharply in their prediction quality. To make this clear let us assume that the feature-block 'miRNA' is not related at all to the outcome 'Y', while the feature-block 'Clinical' is strongly related to it. In this case, it can be assumed that the predictions based on the 'miRNA' feature-block are worse than the predictions based on the 'Clinical' feature-block. Therefore it would be meaningful to put a higher weight on the predictions from the $RF_{Clinical}$ model than on the predictions from the RF_{miRNA} model. Usually, the real strength of the relation between a feature-block and the target variable is unknown. Therefore the predictive quality of the different feature-blocks needs to be estimated. This can be done with the out-of-bag error of the block-wise fitted models. For each block-wise fitted random forest model, the predicted classes for all out-of-bag observations are generated - see chapter 2.2.3 for details. Based on the predicted outcomes and the actual responses, any metric can be calculated to judge the predictive performance of a block-wise fitted random forest model then. In this thesis, either the

accuracy or F-1 score is used as a metric to judge the predictive quality - details on the metrics in chapter 3.1.1. The better the out-of-bag accuracy/ F-1 score of a block-wise fitted model, the higher the estimated predictive quality of the model. The higher the predictive quality of a model, the higher its weight and therefore, the higher its contribution to the final prediction. The reason to use the F-1 score beside the accuracy is that the F-1 score is sensitive to class imbalances in the target variable. In contrast, the accuracy only represents the fraction of correctly classified observations.
Let us have a look at a minimalist example to make the idea of the weighted average clearer. Assume the block-wise fitted random forest models from figure 11 have the following out-of-bag accuracy and predicted probabilities for the observation i:

$$\begin{aligned} OOB_{Acc}(RF_{Clinical}) &= 0.67 & \quad \text{Preds}_{Clinical}(\text{Obs}_i) &= 0.19 \\ OOB_{Acc}(RF_{RNA}) &= 0.86 & \quad \text{Preds}_{RNA}(\text{Obs}_i) &= 0.33 \\ OOB_{Acc}(RF_{miRNA}) &= 0.21 & \quad \text{Preds}_{miRNA}(\text{Obs}_i) &= 0.99 \end{aligned}$$

The predictions of the models represent the probability for a positive response, such that all probabilities < 0.5 result in a negative predicted class, while probabilities ≥ 0.5 result in a positive predicted response class. The actual response class for the observation i is negative, as well as the predictions of $RF_{Clinical}$ and RF_{RNA}. Only the RF_{miRNA} model predicts the response for observation i wrongly as positive. When calculating the simple average of all these predicted probabilities, it results in 0.503. Therefore the final predicted probability is ≥ 0.5, and the predicted class is a positive - which is wrong for observation i. If we use the out-of-bag accuracy of the models as weights for a weighted average, the final predicted probability is $0.355 < 0.5$, and therefore the predicted class is the negative class - which is correct for observation i. So instead of giving all block-wise predictions the same weight, the predictive power of the single feature-blocks can be estimated with any out-of-bag metric and used to weight the block-wise predictions. The better the out-of-bag metric, the higher the weight for the predictions from the given feature-block.

In summary: With the 'block-wise' approach, a separate random forest model is fitted on each feature-block of the training data. For a prediction on a test observation, all block-wise fitted models are asked for a prediction. Only those models that have been trained with a feature-block that is available for the test observation can create a prediction. These predictions can then be averaged in a weighted/ unweighted way to create a final prediction for the test observation.

2.7 Fold-wise Approach

This section introduces the 'fold-wise' approach that was initially proposed by Hornung et al. [3]. This approach was actually not proposed to deal with block-wise missingness in multi-omics data, but to deal with multiple train-sets with the same target variable and different partly overlapping feature-blocks. Nevertheless, this approach can also deal with block-wise missingness in multi-omics data. Other than the approaches from the sections 2.3 - 2.5 this approach does not modify the training data, but the random forest model itself. The 'fold-wise' approach can directly handle block-wise missingness in the training data and does not need to process the data at all. Therefore it does not discard any of the available observations or feature-blocks and uses the available training data efficiently. Furthermore a 'fold-wise' fitted random forest model is flexibly applicable and can "provide predictions for test data that do not feature all covariates available from training" [3].
The random forest model is fitted in a 'fold-wise' manner to the training data. In the beginning, all available folds of the training data are extracted. On each of these folds, a random forest model is then separately fitted. This results in as many fold-wise fitted random forest models as the training data has folds. As the different folds of the training data usually consist of multiple feature-blocks, each fold-wise fitted random forest model incorporates the covariates from multiple different feature-blocks. For a prediction on test data, only "the subsets of covariates included in the test data that are also included in at least one of the" [3] train-sets is used. The prediction of a single fold-wise fitted random forest model is then obtained as follows:
Firstly, remove all trees from the fold-wise fitted random forest that use a split variable as first split that is not available for the test observations. These trees can not even split the test data once, as the first split variable is not available for the test observations. Therefore these decision trees are of no value for the given test observations. Secondly, for each remaining decision tree "follow each branch of the tree and cut the branch as soon as a covariate is used for splitting that is not available" [3] for the test data. This process of cutting branches is called 'pruning'. A node that has to be pruned is a new terminal node of the decision tree then.
After these two steps have been applied to the fold-wise fitted model, the predictions can be obtained as for a standard random forest model. The predictions from the separate fold-wise fitted models can then be aggregated to obtain a final prediction. The fold-wise model fitting is explained in more detail with the example in figure 12. The training data in this example has already been introduced in section 2.1 and has been used as an example in the previous sections as well:

Model Fitting: The training data is displayed at the top of figure 12 and consists of four feature-blocks and three folds. To fit a separate random forest model on each fold, the training data needs to be split, such that each fold can be used to train a random forest model. This is done by merging the feature-blocks of a fold and the corresponding response Y to a separate train-set. The feature-blocks that were not observed for a certain fold are removed from the fold-wise training data - e.g. the feature-blocks 'RNA' and 'MIRNA' were not observed for the fold 'Hospital 1' and had to be removed from the training data of the fold. In figure 12, these separate train-sets are displayed as data frames with a green background below the original training data. Based on each of these three different train-sets, a random forest model can be trained. This results in three random forest models in total - $RF_{Hospital1}$, $RF_{Hospital2}$ and $RF_{Hospital3}$. Each of these models has only been trained with the observed feature-blocks of the different folds - e.g. $RF_{Hospital1}$ was trained with the feature-blocks 'Clinical' and 'CNV'.

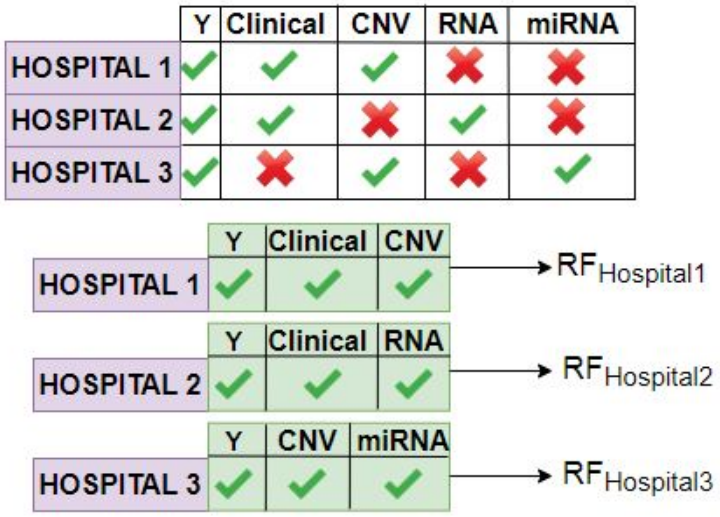

Figure 12: Training of random forest models with the 'fold-wise' approach.

The 'fold-wise' approach trains separate random forest models on the distinct folds of the training data. It consists of as many separate random forest models as the training data consists of unique folds. But how can these models be used to create a prediction? As already mentioned, the fold-wise predictions from the different random forest models need to be aggregated for a final prediction. To receive a prediction from a fold-wise fitted random forest model the single decision trees of such a model might be pruned. Be-

fore explaining the aggregation of the fold-wise predictions, it is important to understand the pruning process. It is explained in the following paragraph with the help of figure 13:

Pruning: Pruning actually describes a process applied to a decision tree to avoid overfitting. But it can also be applied to a decision tree if it contains split variables that are not available for a test observation. The latter idea is used in the 'fold-wise' approach and explained in more detail with the help of figure 13. On the left of the figure, the original decision tree from figure 3 can be seen. It was grown based on the two feature variables 'weight' and 'height'. To obtain a prediction, the observation is passed down the tree until it reaches a terminal node. The predicted probabilities equal the distribution of the target variable in the terminal node. But how can this decision tree predict on an observation with an unknown 'height'? To receive such a prediction, the original decision tree has to be pruned. For this, all nodes that split with the variable 'height' needs to be cut off. This is displayed on the right side of figure 13. The scissors indicate the pruning at the node that uses 'height' as a split variable - this node is a terminal node then. The pruned tree has one terminal node less than the original decision tree. It can create predictions for observations without a 'height' variable, as the pruned decision tree does not use this variable as a split variable anymore.

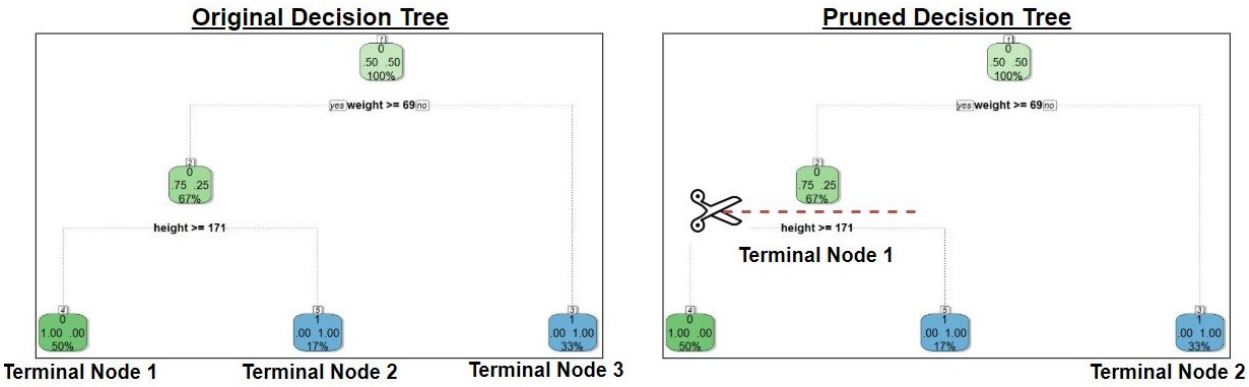

Figure 13: The pruning of a single decision tree. The decision tree was originally introduced in figure 3

The process of receiving a final prediction for an observation based on the predicted classes from multiple fold-wise fitted random forest models is explained in the next paragraph based on the example in figure 14.

Predictions: Assume that the three fold-wise fitted random forest models from figure 12 can be used for the example in figure 14. The test-set is displayed at the leftmost of figure 14, whereby the outcome 'Y' needs to be predicted. Other than the training data from figure 12, the test-set contains only three feature-blocks and misses the 'CNV' block. To create predictions for the observations in the test-set, the three fold-wise fitted random forest models from figure 12 can be used. Each of these models is asked for a prediction. As $RF_{Hospital2}$ was only trained on feature-blocks that are also available in the test-set - 'Clinical' and 'RNA' - no single decision tree of $RF_{Hospital2}$ needs to be pruned. The prediction on the test-set with $RF_{Hospital2}$ is therefore completely regular. The fold-wise fitted random forest models $RF_{Hospital1}$ and $RF_{Hospital3}$ were both trained under the inclusion of the feature-block 'CNV'. This feature-block is not available for the test observations from figure 14. Therefore the single decision trees in $RF_{Hospital1}$ and $RF_{Hospital3}$ need to be pruned, as these trees could contain nodes with split variables that are not available for the test observations. Firstly all decision trees of $RF_{Hospital1}$ and $RF_{Hospital3}$ that use a 'CNV' covariate as the first split variable have to be removed, as these trees can not even partition the test data once. Secondly, all remaining trees are pruned, as explained in the paragraph before. After applying these two steps to the models $RF_{Hospital1}$ and $RF_{Hospital3}$, the predictions for the test observations can be obtained "as in the case of a standard" [18] random forest model. In the end, each fold-wise fitted model creates a prediction for each observation in the test-set, such that there are three predicted outcomes for each observation - $\text{Preds}_{Hospital1}$, $\text{Preds}_{Hospital2}$ and $\text{Preds}_{Hospital3}$. These predictions represent the probabilities for each of the possible response classes. The final predictions for the target variable 'Y' equal a weighted average of these predictions then.

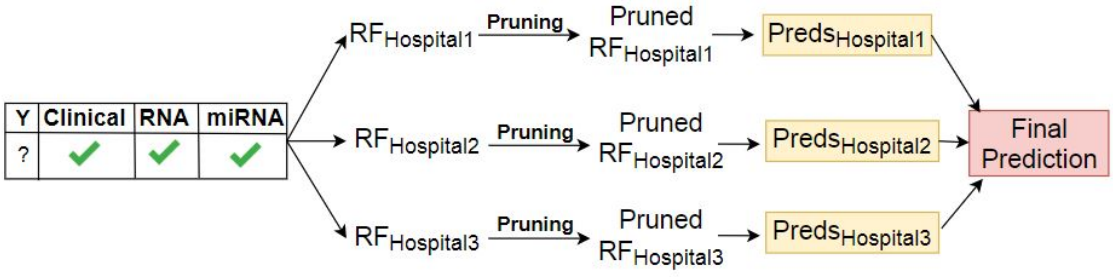

Figure 14: The prediction on test data with the 'fold-wise' approach. The training of the models was described with figure 12.

Different techniques can be applied to create a meaningful weighted average of the different fold-wise predictions. The simplest method is giving each fold-wise fitted model the same weight and return the simple average over all the fold-wise predictions. This might not always be optimal, as the fold-wise fitted random forest models were trained with different combinations of feature-blocks and might differ in their predictive quality. As for the 'block-wise' approach from section 2.6, the predictive quality of the various prediction models can be estimated with an out-of-bag metric - e.g. Accuracy or F-1 score. The out-of-bag metrics of the fold-wise fitted random forest models can then be used to weight the predictions of the different models. The higher the out-of-bag metric for a model, the higher the contribution of the model to the final prediction. There is only one difference in the out-of-bag calculation between the block- and fold-wise approach. As the fold-wise fitted random forest models might be pruned - depending on the test data - the out-of-bag calculation is based on the pruned trees of the random forest then. This is meaningful, as pruning might reduce the predictive power of a model.

In summary: The fold-wise approach fits a separate random forest model on each fold of the training data. Depending on the available feature-blocks in the test-set, the fold-wise fitted random forest models might be pruned. After the pruning process, each fold-wise fitted model can generate predictions for the test-set. These predictions can then be averaged in a weighted/ unweighted way to create final predictions for the test-set.

3 Benchmark Experiments

This section deals with the benchmark experiment for the comparison of the predictive performance of different approaches on data with block-wise missingness. It is not expected that such comparison experiments "result [in] an absolute truth applicable to all situations" [[38], p. 14f], but with enough different data sets and sufficient objective evaluation criteria general trends can be determined [38].
In the beginning, the different metrics used in this thesis are introduced and defined. Afterwards, it is explained how the predictive performance of an approach can be estimated. Subsequently, the different data sources and corresponding data sets are investigated, as well as the diverse structures of block-wise missingness. Furthermore, the evaluation technique for each data source is explained separately after the data sources have been introduced and investigated.

3.1 Assessing the Performance

This part of the thesis supplies the essential information needed to estimate the predictive performance of a classification model. In the first chapter, different metrics are introduced and defined. These are used to rate and compare the predictive performance of the different approaches. Subsequently 'k-fold cross-validation' is introduced and explained. This technique is used to estimate the predictive performance of a model when there is no separate test-set to do so.

3.1.1 Metrics

This subsection introduces the diverse metrics used in this thesis to rate the predictive performances of the different approaches. Metrics are needed "in order to evaluate the performance of a statistical learning method on a given data set [...] [and] measure how well its predictions actually match the observed data" [[39], p. 29]. As this thesis aims to compare classification approaches, only performance metrics suitable for classification problems are introduced. The thesis only uses data sets with a binary response variable to compare the different approaches. Therefore the metrics are only explained for a binary response, but most of the metrics can also be applied to multi-class problems. The selection of metrics is crucial, as it influences how the performance of the approaches are measured and compared.

Confusion Matrix

The confusion matrix itself is not a performance measure, but the basis for most classification metrics [40]. It is used in classification problems, and the matrix always consists of as many rows and columns, as the response variable has unique classes.
An example of a confusion matrix for a binary target can be seen in table 2 - it displays a matrix with the dimensions 2×2. The rows of the table represent the predicted class - $\hat{Y} \in [0; 1]$ -, while the columns represent the actual class - $Y \in [0; 1]$ [41]. Each cell of the matrix displays the amount of observations that were classified correctly | wrongly:

- **TP:** True Positives - amount of observations where the true outcome and the predicted outcome is both positive ($Y = 1$ & $\hat{Y} = 1$)
- **FN:** False Negatives - amount of observations where the true outcome is positive ($Y = 1$), but the predicted outcome is negative ($\hat{Y} = 0$)
- **FP:** False Positives - amount of observations where the true outcome is negative ($Y = 0$), but the predicted outcome is positive ($\hat{Y} = 1$)
- **TN:** True Negatives - amount of observations where the true outcome and the predicted outcome is both negative ($Y = 0$ & $\hat{Y} = 0$)

	Y = 1	Y = 0	
$\hat{Y} = 1$	**TP**	**FP**	} Predicted Classes
$\hat{Y} = 0$	**FN**	**TN**	
	True Classes		

Table 2: Confusion matrix for a binary response.

Therefore **TP** and **TN** show the amount of observations that were labelled correctly, while **FN** and **FP** represent the wrongly labelled observations. Hence the confusion matrix shows the amount of correctly and wrongly labelled observations and can be used for the calculation of sophisticated classification metrics.
The confusion matrix from table 2 is used as a running example for the introduction of the different metrics in the next paragraphs.

Error-rate & Accuracy

"The most common approach for quantifying the accuracy [...] [of a prediction model is the] error-rate" [[39], p. 37]. It represents the proportion of mistakes a prediction model did when predicting the classes for the given set of observations - it can be calculated with help of the confusion matrix: $\textbf{Error-rate} = \frac{\textbf{FP+FN}}{\textbf{TP+TN+FN+FP}}$ [[41], p. 4].
The accuracy on the other hand represents the exact opposite, as it represents the fraction of correctly classified observations. It can also be calculated directly from the confusion-matrix: $\textbf{Accuracy} = \frac{\textbf{TP+TN}}{\textbf{TP+TN+FN+FP}}$ [[41], p. 4].
The error-rate and accuracy are rather simple metrics and intuitively understandable. The accuracy is $\in [0, 1]$ and is better, the higher its value, while the error-rate is $\in [0, 1]$ and is worse, the higher its value.
Both metrics are proper measures when the classes of the response variable are nearly balanced, but should not be used if the classes of the target variable are highly imbalanced [40]. Think of a a data set where only 1% of the observations have a 'positive' response and the remaining 99% a 'negative' one. If a model is naive and always predicts a negative outcome, it still has an accuracy of 99% even though the model is terrible at predicting the actual outcome. To overcome this drawback of the error-rate and accuracy, other metrics are introduced in the following.

Precision

The precision metric represents the fraction of observations with an predicted positive response - $\hat{Y} = 1$ - that also have an actual positive response - $Y = 1$. On basis of the confusion matrix it can be calculated via: $\textbf{Precision} = \frac{\textbf{TP}}{\textbf{TP+FP}}$ [[41], p. 4].
As the precision metric only respects the observations with a predicted positive class, it is less sensitive to class imbalances than the accuracy and error-rate. The precision metric is $\in [0, 1]$, and the lower its value, the worse the predictive performance. Another common term for this metric is 'positive predictive value'.

Recall

The recall metric represents the fraction of observations with an actual positive response - $Y = 1$ - that were correctly classified as positive - $\hat{Y} = 1$. It can be calculated on basis of the confusion matrix via: $\textbf{Recall} = \frac{\textbf{TP}}{\textbf{TP+FN}}$ [[42], p. 2].
As it measures how accurate the predictions for observations with an actual positive outcome are, the recall metric ignores the observations with an actual

negative outcome. Therefore this metric is less sensitive to class imbalances than the accuracy and error-rate metric. The recall metric is $\in [0, 1]$ and the higher its value, the better the predictive performance. Another common term for this metric is 'sensitivity'.

F-1 Score

The F-1 score is a widespread metric that represents the two metrics precision and recall at once. It is the harmonic mean of both metrics and based on the precision and recall it can be calculated via [[41], p. 4]:

$$\textbf{F-1 Score} = 2 * \frac{\textbf{Precision} * \textbf{Recall}}{\textbf{Precision} + \textbf{Recall}} \tag{10}$$

The F-1 score is $\in [0, 1]$ and the better the predictive performance of a model the higher the F-1 score.

Balanced Accuracy

The balanced accuracy is a metric that "avoids inflated performance estimates on imbalanced data sets" [43]. It calculates the accuracy for each possible response class separately and returns the average of these class-wise accuracies. This average of the class-wise accuracies can be determined with the confusion matrix [44]:

$$\textbf{Balanced Accuracy} = \frac{\frac{\textbf{TP}}{\textbf{TP+FN}} + \frac{\textbf{TN}}{\textbf{TN+FP}}}{2} \tag{11}$$

As the regular accuracy, it is $\in [0, 1]$ and the better the higher its value.

Matthews correlation coefficient

The Matthews correlation coefficient (MCC) is a metric that is insensitive to class imbalances. It is a "widely used performance measure in biomedical research" [[42], p. 1] and can be seen as a "discretization of the Pearson correlation for binary variables" [[42], p. 1]. It can directly be calculated with the confusion matrix via [[45], p. 415]:

$$\textbf{MMC} = \frac{\textbf{TP} * \textbf{TN} - \textbf{FP} * \textbf{FN}}{\sqrt{(\textbf{TP} + \textbf{FP}) * (\textbf{TP} + \textbf{FN}) * (\textbf{TN} + \textbf{FP}) * (\textbf{TN} + \textbf{FN})}} \tag{12}$$

The MMC is in "essence a correlation coefficient" [43] with values $\in [-1, 1]$. 1 is the best possible value and represents perfect predictions, 0 indicates that the predictions are random on average and a value of -1 indicates the worst possible predictive performance [43].

3.1.2 k-fold Cross-Validation

'K-fold cross-validation' is a technique to estimate the predictive performance of a model when there is no separate test-set. Before explaining the method in more detail, it is essential to understand the difference between the calculation of a metric on the test- and train-set.
Regardless of the metric, there is a big difference whether it is calculated on the train- or the test-set. The train-set consists of the data used to train a prediction model, while the data of the test-set "was not used in training the model" [[39], p. 176]. The results of calculating a metric on the train-set are quite different from the metric obtained when calculating it on the test-set - "in particular the former can dramatically underestimate the latter" [[39], p. 176]. The calculation of a metric on the train-set is therefore over-optimistic, as "the same data is being used to fit the method and assess its [predictive performance]" [[22], p. 228]. Hence the calculation of a metric should be done with data that has not been used to train the model. In case of having no designated test-set, there are diverse techniques to estimate the predictive performance by using only the available training data.
A straightforward approach to obtain a metric on a test-set is the so-called 'hold-out' method. With this technique a data set D is split to a train-set D_{train} and a test-set D_{test}. A model can then be trained on D_{train} and evaluated on D_{test} to see how well that model performs on unseen data. Such a single train-test split can be problematic and lead to distorted estimates, as the metric highly depends on how the data is split into train- and test-set. Different train-test splits of the data can, therefore, lead to completely distinct results. In general, the smaller the test-set D_{test} "the higher the variance of our estimated metric" [[46], p. 18]. The smaller the train-set D_{train}, the bigger the introduced pessimistic bias, as the model is trained on fewer data and will, therefore, learn less and perform worse [46].
To avoid this bias-variance trade-off with the 'hold-out' method, there are techniques that use the data more efficiently through resampling. Resampling, in general, describes the process of repeatedly splitting the training data D into train- and tests-sets, whereby the resulting metrics can be calculated for each of these splits and aggregated then [46]. This leads to a more stable performance estimation. A well-known method to do so is the so-called 'k-fold cross-validation'.
'K-fold cross-validation' divides the available training data into k groups of approximately equal size. By using only k-1 of these folds for the training of the model, the trained model can then be evaluated on the remaining fold - the so-called held-out fold. Based "on the observations in the held-out fold" [[39], p. 181] any metric can then be calculated. This process is repeated until

each fold has been the held-out fold once. This results in k estimates of a metric, whereby the final k-fold cross-validation estimation for this metric equals the average of the k values [39]: $CV(k) = \frac{1}{k} \sum_{i=1}^{k} \text{Metric}_i$
The process of the 'k-fold cross-validation' method is illustrated in figure 15. In the top of the figure, the whole data is displayed. In the first step of the cross-validation, the data is shuffled and partitioned into five equally sized folds. In the first of the five iterations, the first fold is used as a held-out fold, while the remaining folds are used to train the model. The fitted model is then evaluated with the observations from the first fold, which results in Metric_1. The same process is repeated for the remaining iterations 2-5. In each of these iterations, a different fold is used as held-out fold. Therefore each of these iterations results in an estimated metric based on a different held-out fold - Metric_1, ..., Metric_5. The final estimation of the metric equals the average of the metric overall five iterations: $\frac{1}{5} \sum_{i=1}^{5} \text{Metric}_\text{i}$

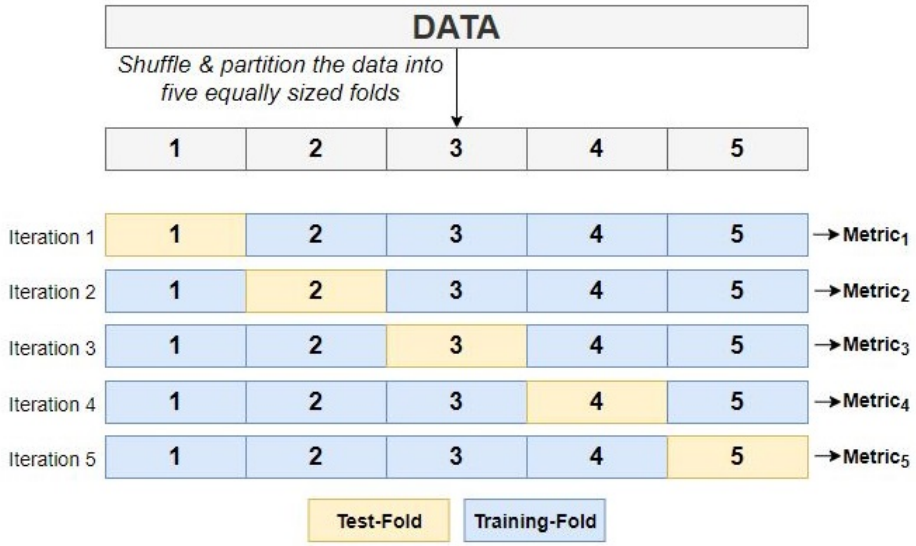

Figure 15: Example for 5-fold cross-validation.

In summary: 'K-fold cross-validation' is a technique to estimate the predictive performance of an approach on unseen data. It can be applied if there is only a train-set, but no separate test-set.

3.2 Data

In this subsection, the data sets from the diverse sources are investigated. In total, there are two distinct sources of data for the comparison of the different approaches. The comparison of approaches with an insufficient number of data sets leads to underpowered results, as the performance of the approaches across the different data sets can be highly variable [38]. Therefore enough data sets from different sources are needed for a meaningful comparison of the approaches. In the coming sections, the different sources and corresponding data sets are introduced in more detail, as well as their different structures of block-wise missingness. After the introduction of each source, the data source specific technique for the evaluation is explained.

3.2.1 TCGA Data

The first source for multi-omics data is 'The Cancer Genome Atlas' (TCGA) that provides "real multi-omics data sets [...], where each of these data sets contains the measurements of patients with a certain cancer type" [[18], p. 14]. The data was not directly accessed via TCGA, but provided by R. Hornung who has used the data in one of his articles already [18]. In total, 21 different data sets were provided, whereby these data sets have already been processed. The processing included the imputation of "missing values in the clinical block" [[18], p. 14] and the transformation of categorical feature variables into binary numerical features. For the imputation of the clinical covariates, the 'k nearest neighbours imputation' and 'univariate logistic regression' have been used - further details can be found in the article itself [18]. Hence the 21 provided data sets do not miss any values and only consist of numerical covariates.
The 21 data sets have a survival outcome, but as this thesis aims to compare classification approaches, this outcome is not used as the target variable. Instead, the covariate 'gender' from the 'Clinical' feature-block is used as a binary categorical response variable. This is not an unusual procedure and has been applied in other studies already - e.g. [47]. Even though the 'gender' variable "is not a clinically meaningful outcome in biomedical applications, it features major advantages for a purely methodological investigation" [[47], p. 5]. Seven of the 21 available data sets do not contain a 'gender' covariate and had to be removed, such that a total of 14 data sets remained.
Each of the 14 remaining data sets is completely observed and consists of the same five feature-blocks. Even though the number of available covariates for each feature-block differs between the 14 data sets, they are still in a similar field. The average amount of covariates over the 14 data sets for

each feature-block is displayed in table 3. While the 'Clinical' feature-block represents clinical information like 'weight', 'height' and 'smoking status' the remaining four feature-blocks are different types of omics data and represent biological properties. The 'Clinical' feature-block has the lowest number of covariates on average, while the 'CNV' block has by far the most.

Feature-Block	**Average amount of covariates**
Clinical	3.5
miRNA	770
Mutation	16218
CNV	57964
RNA	23559

Table 3: The average amount of covariates in each feature-block over the 14 TCGA data sets.

The average amount of observations for the 14 data sets is ∼280. Hence, the amount of features is on much higher than the number of observations - a characteristic property of multi-omics data.

Reducing the dimensionality of the omics feature-blocks

A problem that comes with the choice of 'gender' as target variable are the "overly strong biological signal[s]" [[47], p. 5] contained in the different omics feature-blocks. It is aimed to reduce these biological signals by only using a subset of variables for each omics block, and "thus make it comparable to signals observed in applications of clinical relevance" [[47], p. 5]. Another reason for using only a subset of the available covariates per feature-block is the computational expense of evaluating models on such high dimensional data. The reduction of the computational effort is especially important for the 'fold-wise' approach from section 2.7. This approach was implemented on the basis of the 'simpleRF' package [48] in plain R, as no package offered a method to prune the single trees of a random forest model dynamically. Therefore this approach is computational much slower than the other approaches from sections 2.3 - 2.6, as these were implemented with the 'randomForestSRC' package [33] that directly builds upon 'Java' and 'C'.
To find a reasonable subset for each omics feature-block, the performance of a random forest model is evaluated on each of the single feature-blocks and their corresponding subsets. The predictive performance of the models on the different omics blocks is evaluated with 5-fold cross-validation and rated

with the accuracy - general details on the metrics/ k-fold cross-validation in the section 3.1.1/ 3.1.2. The predictive performance of a random forest model on the single omics blocks and their corresponding subsets - based on all 14 available TCGA data sets - is displayed in figure 16.
The figure consists of totally four plots, whereby each plot represents a single omics feature-block. Each of these plots contains eight boxplots representing the accuracy - y-axis - of the model for different subsets of used covariates - x-axis. The predictive performance for the feature-blocks 'CNV' and 'RNA' is, in general, the best, while the feature-blocks 'miRNA' and 'Mutation' are much worse. The feature-blocks 'miRNA' and 'Mutation' are, therefore only trimmed to reduce the computational effort. As the 'miRNA' block only consists of 770 covariates on average, only 50% of the available covariates are removed. The feature-block 'Mutation' has 16218 covariates on average from which 90% are removed. While the trimming of these two feature-blocks reduces the computational effort, it does not reduce the predictive quality of these single feature-blocks too much. The number of covariates in the 'CNV' and 'RNA' feature-blocks have to be trimmed to reduce the biological signals in these blocks. 85% of the available covariates were removed from the 'RNA' block. This leads to a lower but still reasonable predictive performance with this block. The 'CNV' block leads to the best predictive performances and seems to contain a lot of strong biological signals. Therefore 95% of the

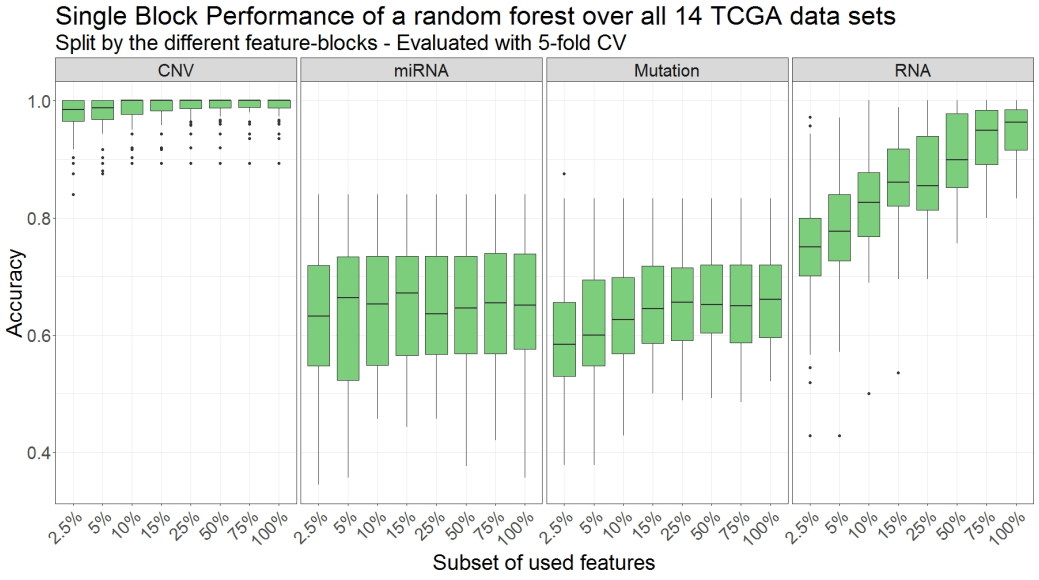

Figure 16: The accuracy of a random forest model evaluated on the single omics feature-blocks for a range of possible subsets on all 14 TCGA data sets.

available covariates from this feature-block were removed. The predictive performance with the trimmed 'CNV' block is still by far the best and not much worse than on the complete 'CNV' feature-block. The same plot but with the F-1 score as a metric is displayed in figure A-2 in the attachment - the results are about the same.

Removing 50% of the covariates from the 'miRNA' block, 90% from the 'Mutation' block, 85% from the 'RNA' block and 95% from the 'CNV' block lead to data sets with much lower dimensions than originally. Nonetheless, the essential multi-omics property of more observed covariates than observations is still valid for the trimmed data sets. The average amount of covariates over the 14 data sets for each reduced feature-block is displayed in table 4.

Reduced Feature-Block	**Average amount of covariates**
Clinical	3.5
miRNA	385
Mutation	1616
CNV	2898
RNA	3555

Table 4: The average dimensionality of the five trimmed feature-blocks for the 14 TCGA data set.

Inducing block-wise missingness

To estimate the predictive performance of the different approaches, k-fold cross-validation is applied. The 14 provided TCGA data sets are completely observed and do not contain any missing values. But as these data sets shall be used for the investigation of different approaches capable of dealing with block-wise missingness, block-wise missingness needs to be inducted into the training-sets in the cross-validation - more details on this in algorithm 3.
Each of the 14 TCGA data sets consists of the response variable 'Y' and the five feature-blocks 'Clinical', 'CNV', 'RNA', 'Mutation' and 'miRNA'. These 14 data sets are induced with block-wise missingness according to the following patterns. In the tables of these patterns, a red cross indicates a missing feature-block for a fold, while a green tick indicates an observed feature-block.

Pattern 1 The first pattern of block-wise missingness is displayed in table 5. Every fold consists of the same amount of observations, and to each of them, there is an observed response variable 'Y', the 'Clinical' feature-block and one additional omics feature-block. Therefore every single fold consists of two feature-blocks as observed covariates - one 'Clinical' and one omics block. The 'Clinical' block is the same for all folds, while the omics blocks differ from fold to fold.

	Y	**Clinical**	**CNV**	**RNA**	**Mutation**	**miRNA**
Fold 1	✓	✓	✓	✗	✗	✗
Fold 2	✓	✓	✗	✓	✗	✗
Fold 3	✓	✓	✗	✗	✓	✗
Fold 4	✓	✓	✗	✗	✗	✓

Table 5: The first block-wise missingness pattern for the TCGA data sets.

Pattern 2 The second pattern of block-wise missingness is displayed in table 6. The amount of observations in the different folds is equal for all folds, while the number of observed feature-blocks differ from fold to fold. The order of the feature-blocks was randomly changed in comparison to 'Pattern 1'. The last omics block 'CNV' is observed for all folds and the first omics block 'miRNA' only for the first fold. The amount of available feature-blocks decreases from 'Fold 1' to 'Fold 4'. 'Fold 1' is completely observed within all five feature-blocks and 'Fold 4' in only two feature-blocks - 'Clinical' and 'CNV'.

	Y	**Clinical**	**miRNA**	**RNA**	**Mutation**	**CNV**
Fold 1	✓	✓	✓	✓	✓	✓
Fold 2	✓	✓	✗	✓	✓	✓
Fold 3	✓	✓	✗	✗	✓	✓
Fold 4	✓	✓	✗	✗	✗	✓

Table 6: The second block-wise missingness pattern for the TCGA data sets.

Pattern 3 The third pattern of block-wise missingness is displayed in table 7. The amount of observations is equally split into the four folds. For each of these folds, it is randomly drawn which feature-blocks are observed. The probability for any feature-block to be observed equals $\frac{2}{3}$ and the probability of not observing a block, therefore, $\frac{1}{3}$. The 'Clinical', as well

as the 'RNA' feature-block, were sampled for every fold. Therefore the four folds only differ in the feature-blocks 'CNV', 'Mutation' and 'miRNA'. The same feature-blocks were sampled for 'Fold 3' and 'Fold 4', such that these folds have the same covariates. Therefore these two folds are a single fold, and the data consists of only three unique folds then.

	Y	Clinical	CNV	RNA	Mutation	miRNA	
Fold 1	✓	✓	✓	✓	✗	✓	
Fold 2	✓	✓	✓	✓	✗	✗	
Fold 3	✓	✓	✗	✓	✓	✓	} **Fold3**
Fold 4	✓	✓	✗	✓	✓	✓	

Table 7: The third block-wise missingness pattern for the TCGA data sets.

Pattern 4 The fourth and last pattern of block-wise missingness is displayed in table 8. Other than in the three patterns before, the amount of observations is equally split in two instead of four folds. The next difference to the first three patterns is that the amount of feature-blocks is reduced from five to three. For this two of each of the four available omics feature-blocks were combined into a single omics feature-block - this has been done randomly. The feature-blocks 'RNA' and 'miRNA' were combined to a single feature-block, as well as the feature-blocks 'Mutation' and 'CNV'. Therefore the whole data consists of three feature-blocks - 'Clinical', 'RNA & miRNA' and 'Mutation & CNV' - and two folds.

	Y	Clinical	RNA & miRNA	Mutation & CNV
Fold 1	✓	✓	✓	✗
Fold 2	✓	✓	✗	✓

Table 8: The fourth block-wise missingness pattern for the TCGA data sets.

Evaluation Technique

The 14 data sets from the TCGA source are used to assess the predictive performance of the different random forest adaptions for data with block-wise missingness. All data sets are completely observed and to none of these exists a separate test-set. A slightly modified version of 5-fold cross-validation is applied to assess the predictive performance of the different approaches on this data. The process is illustrated in algorithm 3.

In the very beginning a data set D, an approach App and a block-wise missingness pattern $Patt$ have to be selected. The data set D is split into five equally sized folds, whereby a single fold is used as test-set and the remaining four folds as train-set. Then block-wise missingness is induced into the train-set according to the selected pattern of missingness $Patt$ and the model is fitted on this data. The predictive performance of an approach does not only depend on the train-set, but also on the available feature-blocks in the test-set. Hence the approach App is evaluated not only on a fully observed test-set, but also on the same test-set, but with different combinations of missing feature-blocks. This is relevant, as block-wise missingness can affect the train-set as well as the test-set. Firstly the approach is evaluated on the test-set with all feature-blocks available (2.3.1). Then it is further evaluated on the same test-sets with one missing feature-block (2.3.2) - e.g. missing 'CNV' feature-block -, with two missing feature-blocks (2.3.3) - e.g. missing 'CNV' & 'RNA' feature-block -, with three missing feature-blocks (2.3.4) - e.g. missing 'CNV' & 'RNA' & 'miRNA' feature-block -, and finally on the same test-sets with only a single observed feature-block (2.3.5) - e.g. only has the 'CNV' feature-block. For each of these possible test-sets, the metrics are calculated. When the evaluation of an approach App on a data set D with the block-wise missingness pattern $Patt$ is done, to each possible test-set, there are five values of each metric. The average of these metrics for each possible test-set is then used to compare the predictive performance of the different approaches - section 4.1.

In summary: To evaluate an approach on a TCGA data set 5-fold cross-validation is applied. The training data is induced with a block-wise missingness pattern, and the model is fitted on this data. The model is then not only evaluated on the fully observed test-set, but also on the same test-set for all possible combinations of block-wise missingness.

Algorithm 3: Evaluation of the approaches with the TCGA data

Input : D ← TCGA data set with n observations & p features
App ← Approach
$Patt$ ← Pattern of block-wise missingness

1. Split the fully observed data set D into five equally sized folds
2. for $k \leftarrow 1$ **to** 5 **do**
- **2.1** Use fold k as test-set and the remaining four folds as train-set;
- **2.2** Induce the block-wise missingness according to $Patt$ into the train-set;
- **2.3** Fit the App on the train-set with block-wise missingness;
- **2.4** Evaluate the predictive performance of the approach App for different combinations of observed feature-blocks in the test-set;
  - **2.4.1** Evaluation on the fully observed test-set;
  - **2.4.2** Evaluation on test-sets with one missing feature-block;
  - **2.4.3** Evaluation on test-sets with two missing feature-blocks;
  - **2.4.4** Evaluation on test-sets with three missing feature-blocks;
  - **2.4.5** Evaluation on test-sets with a single feature-block;

3.2.2 Clinical asthma data

This subsection introduces the second source for multi-omics data - the so-called 'clinical asthma data'. It is a real-world data set with block-wise missingness that was provided by the group of Prof. Dr. med. Bianca Schaub at the 'paediatric clinic Dr. von Haunersches Kinderspital'. The data was collected as part of a clinical case-control study in the field of asthma research, whereby the target variable of the data is binary and defined as the presence of asthma. The data set contains 521 observations in total (age 5-16), whereby 265 observations have a negative response and the remaining 256 observations have a positive response. The data comprises five different data sources, with 268 variables in total. 44 of those variables originated from a questionnaire, 16 from a clinical routine diagnostic, 19 from the information on allergen sensitization, 29 are from cytokine expression data, and 166 from gene expression data. As the feature space of the gene expression data had a block-wise missing structure itself, the gene expression data was split into two data sets - 'gene expression data I' (82 variables) and 'gene expression data II' (84 variables). The number of observations and variables in the different data sources is displayed in table 9. The amount of observations in the various data sources inversely reflects the effort of generating the data - e.g. the fewer observations, the more valuable the measured features.

ID	Feature-Block	Number of observations	Number of variables
1	Questionaire	521	44
2	Clinical routine diagnostics	385	16
3	Allergen sensitization	472	19
4	Cytokine expression data	149	29
5	Gene expression data I	66	82
6	Gene expression data II	46	84

Table 9: The number of observations and variables for the different data sets in the clinical asthma data.

The data preparation was carried out by the project partner and conducted as follows:

(1) Feature variables (especially from the questionnaire) were selected such that they are reasonable as predictor variables and either continuous or binary distributed.

(2) All data sources were cleaned by removing variables with a high proportion of missing values ($\geq$30%) and imputing the remaining missing values with the 'missForest' imputation approach.

(3) The data sets from the different sources were then combined into a single data set.

The merged data set has the dimensions 521×274 and consists of six different feature-blocks in total - each data source corresponds to a feature-block. The single variable names of the different feature-blocks were censored because of data protection regulations. Every single of the 521 observations in the data misses at least one feature-block, such that there is no single observation that is completely observed in all six feature-blocks. The structure of the block-wise missingness is displayed in figure 17. In this figure, each row corresponds to an observation in the data set, and each column stands for a single data source. A green line indicates that an observation was observed entirely for a given feature-block. In contrast, a red line indicates that the observation was not observed at all for the corresponding feature-block. The 'Questionaire' feature-block was observed for every patient of the data set, and there is no single patient that is observed in both 'gene expression data I' and 'gene expression data II' feature-blocks.

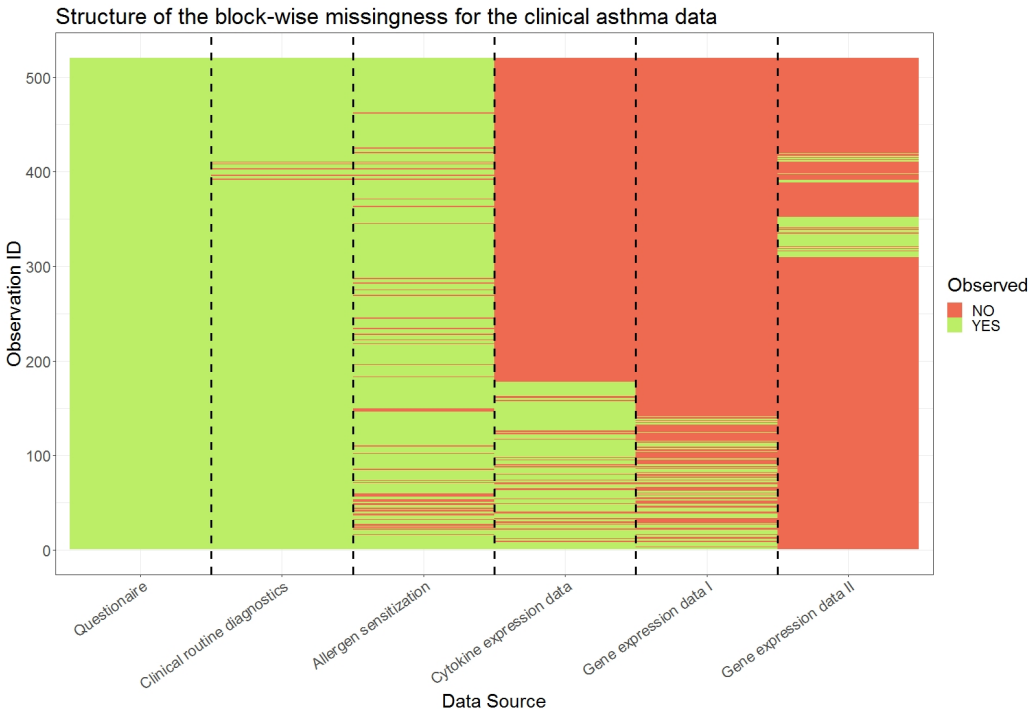

Figure 17: Structure of block-wise missingness in the clinical asthma data.

Evaluation Technique

The clinical asthma data is used to assess the predictive performance of the different approaches for data with block-wise missingness. Besides the comparison of the predictive performance of the various random forest based approaches, also the performance of the mdd-sPLS method and the different priority-Lasso adaptions from Hagenberg [1] are compared on basis of this data. To assess the predictive performance of the different approaches on this data, regular 5-fold cross-validation is applied - the distribution of the target variable is approximately equal for all folds (50:50). Other than the TCGA data, the clinical asthma data already consists of block-wise missingness. As the data is randomly split into five folds, each of these folds consist of observations with diverse block-wise missing values. Hence not only the training-sets consist of different block-wise missingness structures but also the test-sets ($\widehat{=}$ held-out folds) contain observations with different observed feature-blocks. This is completely different compared to the cross-validation on the TCGA data, where the observations in the test-set always have the same observed feature-blocks. The 5-fold cross-validation technique for the clinical asthma data is explained now and illustrated in algorithm 4.

In the beginning the clinical asthma data is split into five equally sized folds. In each iteration of the 5-fold cross-validation one of the folds is used as test-set (held-out fold) and the remaining folds as train-set. Firstly, the different patterns of block-wise missingness are extracted from the test-set. For each pattern of block-wise missingness in the test-set, the approach *App* is fitted on the train-set, such that it can provide predictions for the test-observations with the current pattern of block-wise missingness. After the outcome was predicted for each observation in the test-set, the predicted classes are compared with the true classes and the resulting metric for the current cross-validation fold is returned.

Algorithm 4: Evaluation of the approaches with the clinical asthma data

Input : D $\leftarrow$ clinical asthma data with block-wise missingness
$App \leftarrow$ Approach

1. Split the clinical asthma data D into five equally sized folds

2. for $k \leftarrow 1$ **to** 5 **do**

- **2.1** Use fold k as test-set and the remaining four folds as train-set;
- **2.2** Extract the unique patterns of block-wise missingness in the test-set - $test_{patterns}$;
- **2.3 for** $pattern_{current} \in test_{patterns}$ **do**
  - **2.3.1** Fit *App* on the train-set, such that it can be used for the prediction on $pattern_{current}$;
  - **2.3.2** Generate predictions for the test-observations with block-wise missingness according to $current_{pattern}$
- **2.4** Evaluate the predictive performance of the approach *App* by comparing the predicted classes with the true classes;

4 Results

This section presents the results of the different approaches for data with block-wise missing values on the diverse data sources. Firstly the results of the various random forest adaptions on the TCGA data are shown, compared and analysed. Afterwards, the results of the different random forest adaptions, the mdd-sPLS method and the several priority-Lasso adaptions from Hagenberg [1] are shown, compared and analysed based on the clinical asthma data.

4.1 TCGA

This subsection contains all cross-validation results of the different random forest adaptions for the diverse patterns of block-wise missingness in the TCGA data. Detailed information on the evaluation technique with the TCGA data can be found in the paragraph 'Evaluation Technique' of section 3.2.1. Details on the diverse patterns of block-wise missingness can be found in the paragraph 'Inducing block-wise missingness' of section 3.2.1. Firstly, the results for each approach on the various patterns of block-wise missingness ('Pattern 1' - 'Pattern 4') are shown and explained separately before the different approaches are compared then.
The figures that show the results for the different approaches always have a similar structure. Each figure consists of four separate plots - one for each pattern of block-wise missingness in the training data. Each of these plots show the results as boxplots, whereby the y-axis shows the F-1 score and the x-axis the different combinations of feature-blocks in the test-set - e.g. 'miss2_BD' shows the results of the approach on a test-set with missing feature-blocks 'B' and 'D'. For each possible combination of feature-blocks in the test-set - so-called 'test-situations' - the results are plotted as a boxplot. The letters A, B, C & D represent the feature-blocks in the diverse patterns. The order is as in the tables 5 - 8 - e.g. 'A' in pattern 1 stands for 'CNV'. The data in the different test-situations is always the same and only differs in the observed feature-blocks - e.g. in a given pattern, the test-situations 'miss1_A' (observed feature-blocks: [*B, C, D & Clin*]) and 'miss1_B'(observed feature-blocks: [*A, C, D & Clin*]) only differ in the observed feature-blocks, while everything else is equal (the observations and feature-blocks [*C, D & Clin*] are the same for both test-situations). As the training data in 'Pattern 4' only has three feature-blocks and not five as in the remaining three patterns, there are fewer test-situations. The test-situations that are only available for the first three patterns but not for 'Pattern 4' are marked with an orange 'X'.

4.1.1 Complete-Case Approach

This subsection introduces the results of the complete-case approach on the diverse patterns of block-wise missingness on the TCGA data - details on the approach can be found in section 2.3.

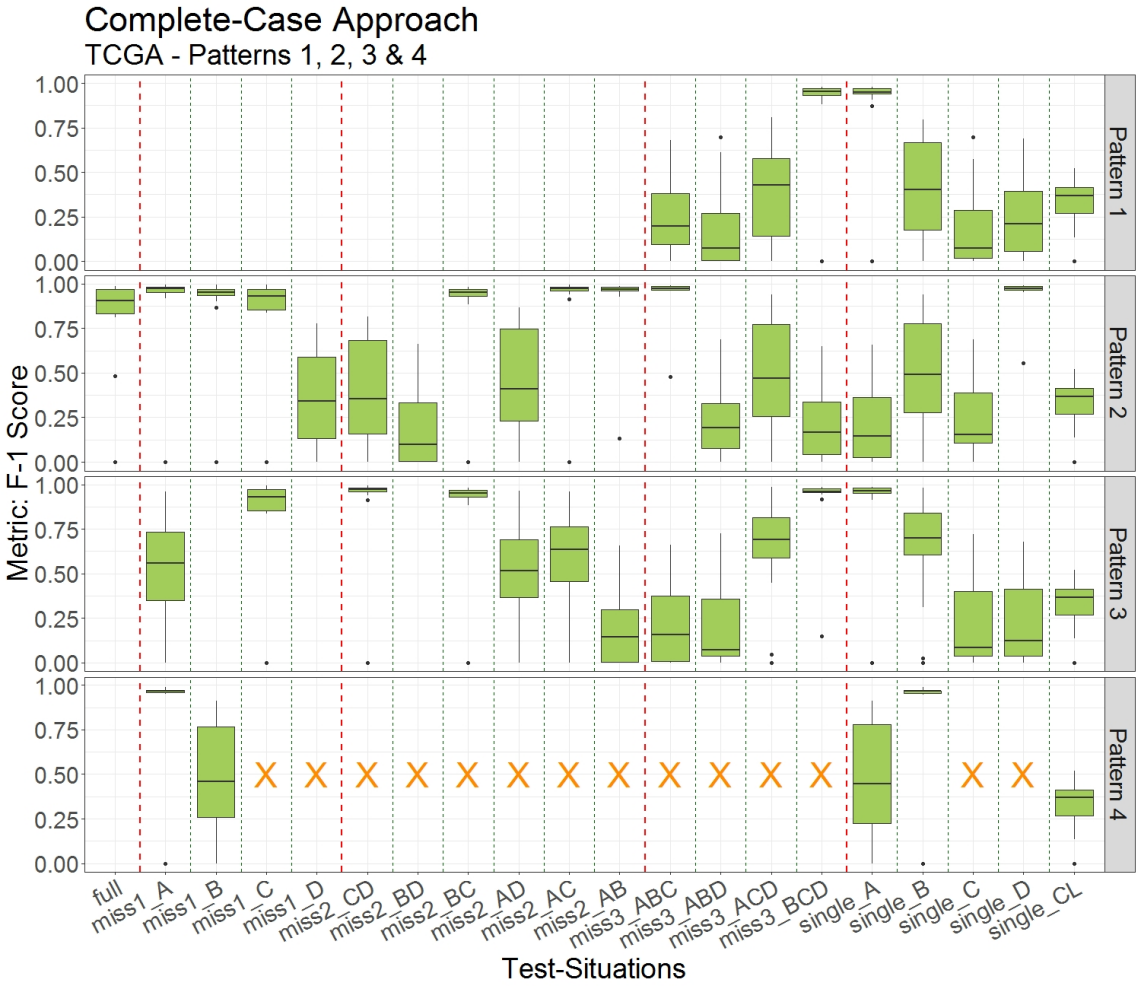

Figure 18: Results of the 'Complete-Case' approach on the TCGA data with induced block-wise missingness according to pattern 1, 2, 3 & 4. Test-Situations that are not available for 'Pattern 4' are marked with an orange cross.

Figure 18 shows the average results of the 5-fold cross-validation for the complete-case approach on the 14 TCGA data sets for the various patterns of block-wise missingness. The first thing that stands out is that there are not always results for each test-situation - e.g. with 'Pattern 1' there are no

results for the situations where the test-set is fully observed ('full'), misses a single feature-block (e.g. 'miss1_A') or two (e.g. 'miss2_CD'). The complete-case approach only uses observations that are observed in the same feature-blocks as the test-set to train a random forest model. As the training-set with 'Pattern 1' only contains folds with an observed clinical block and one additional omics block, there are simply no complete case observations for these test-situations. Hence the complete-case approach can not generate predictions for these test-situations. Only with 'Pattern 2' the complete-case approach can create predictions for all possible test-situations. In 'Pattern 1' and 'Pattern 3' it is noticeable that the results are much better in test-situations with the feature-block 'A' - e.g. 'miss3_BCD'/ 'single_A' (in both patterns 'A' stands for the 'CNV' block). In 'Pattern 2' the results are the best in test-situations with the feature-block 'D' - e.g. 'miss1_A'/ 'miss1_B'/ 'miss1_C' (in this pattern 'D' stands for the 'CNV' block). In 'Pattern 4' the results are the best for test-situations with the feature-block 'B' - e.g. 'miss1_A'/ 'single_B' (in this pattern 'B' stands for the 'CNV' & 'Mutation' block).

4.1.2 Single-Block Approach

This subsection introduces the results of the single-block approach on the diverse patterns of block-wise missingness on the TCGA data - details on the approach can be found in section 2.4.
Figure 19 shows the average results of the 5-fold cross-validation for the single-block approach on the 14 TCGA data sets for the various patterns of block-wise missingness. To each test-situation, the results are structured according to the feature-block used for the training of the single-block approach. The first thing that stands out is that the single-block approach can not create predictions for all test-situations. It can only provide predictions when the test-set contains the feature-block that has been initially used for the training. E.g. when training a random forest model on the 'Clinical' feature-block, the resulting model can only generate predictions for observations/ test-sets that are observed in this feature-block. Moreover, the performance of the single-block approach on a given feature-block is the same for all test-situations. This makes sense, as the test-sets in the different test-situations only differ in their observed feature-blocks, and as the single-block approach uses the same feature-block for the predictions, the performance is equal for the different test-situations. Regarding the predictive performance, it is clear to see that in 'Pattern 1' and 'Pattern 3' the performance is the best when using the feature-block 'A' (in both patterns 'A' stands for the 'CNV' block). In 'Pattern 2' the single-block approach has its best performance

with the feature-block 'D' (in this pattern 'D' stands for the 'CNV' block), and in 'Pattern 4' it is the best with the feature-block 'B' (in this pattern 'B' stands for the 'CNV' & 'Mutation' block).

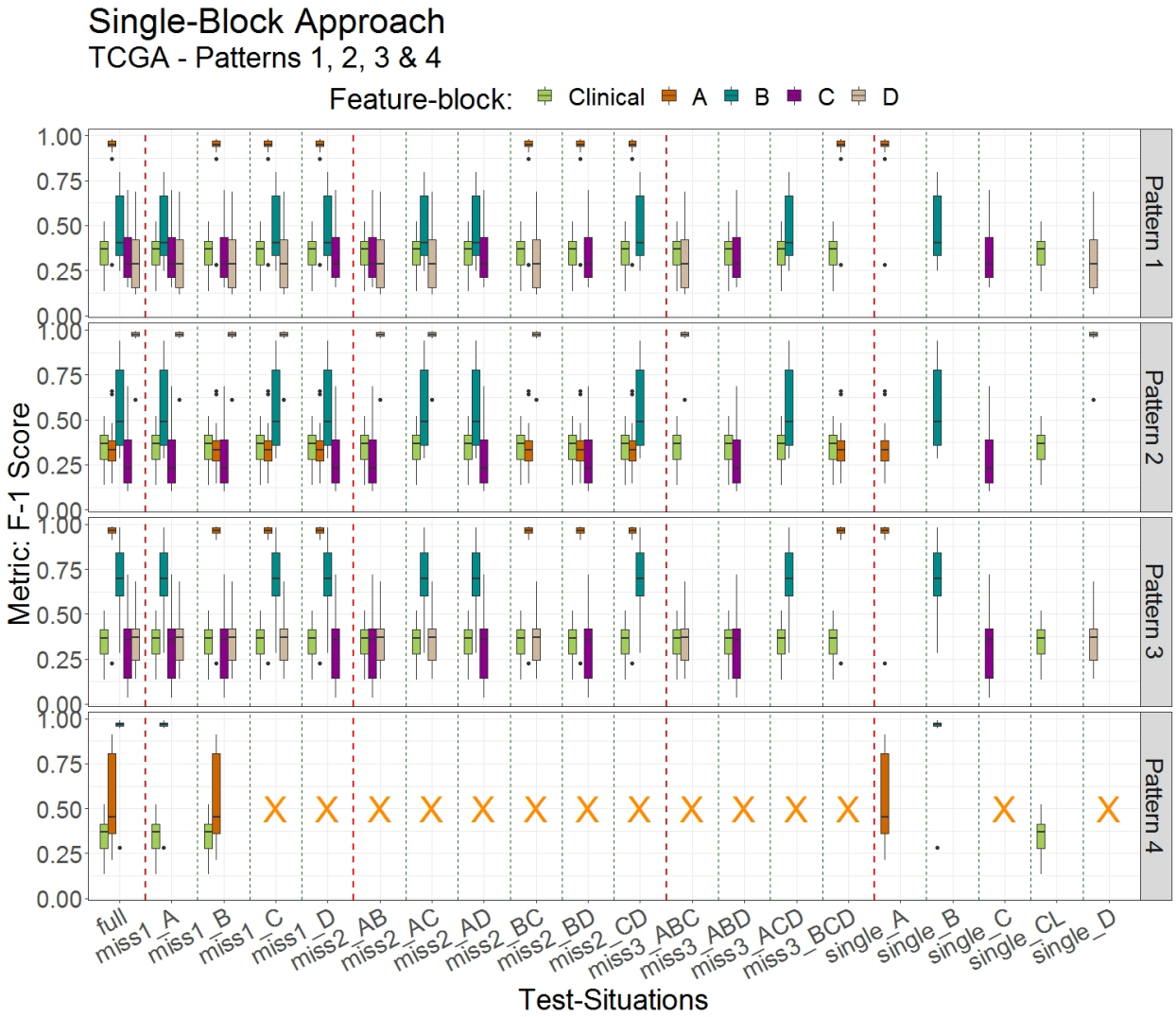

Figure 19: Results of the 'Single-Block' approach on the TCGA data with induced block-wise missingness according to pattern 1, 2, 3 & 4. Test-Situations that are not available for 'Pattern 4' are marked with an orange cross.

4.1.3 Imputation Approach

This subsection introduces the results of the imputation approach on the diverse patterns of block-wise missingness on the TCGA data - details on the approach can be found in section 2.5.

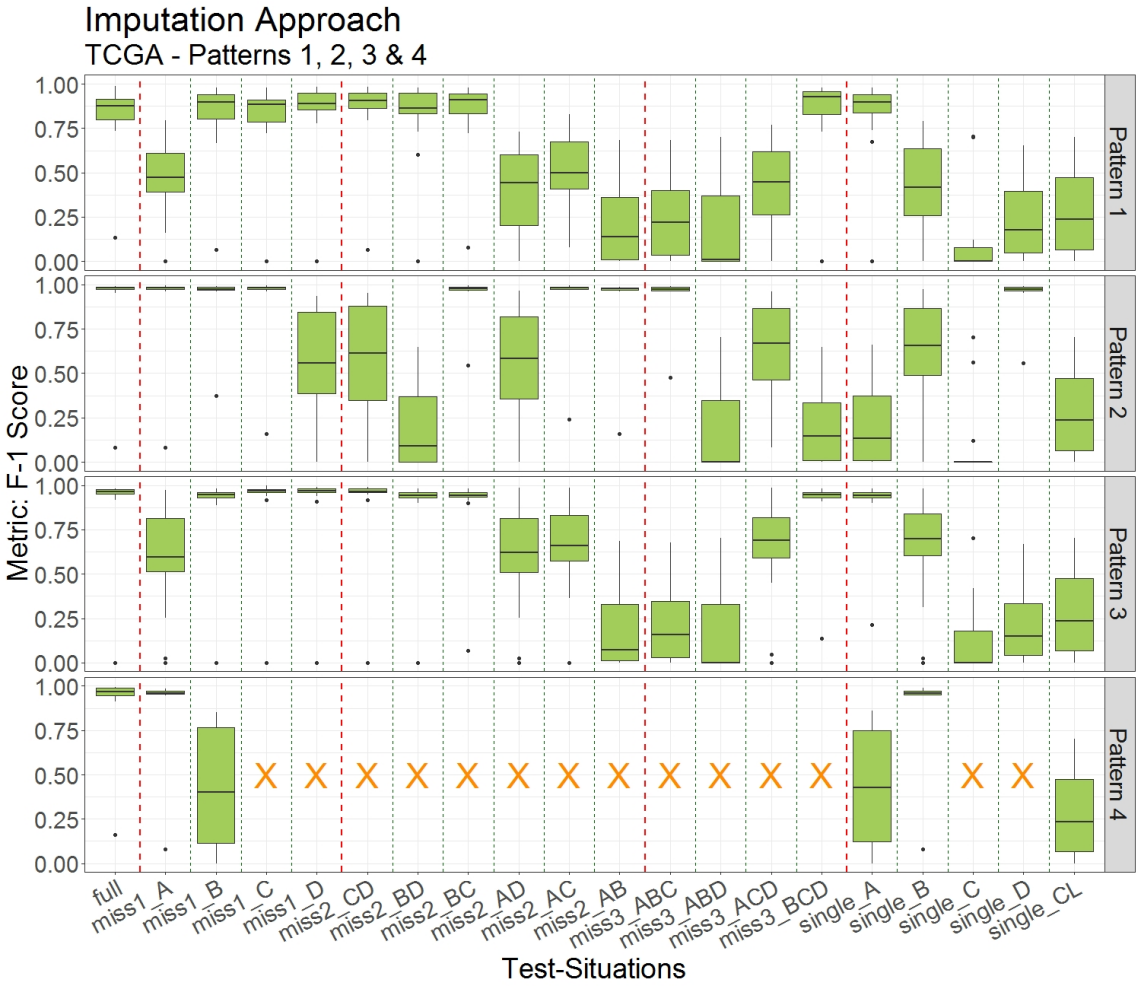

Figure 20: Results of the 'Imputation' approach on the TCGA data with induced block-wise missingness according to pattern 1, 2, 3 & 4. Test-Situations that are not available for 'Pattern 4' are marked with an orange cross.

Figure 20 shows the average results of the 5-fold cross-validation for the imputation approach on the 14 TCGA data sets for the various patterns of block-wise missingness. The first thing that stands out is that the imputation approach can provide predictions for any possible test-situation. This is

possible, as the imputation approach firstly imputes all missing values in the training data, such that the data is completely observed then. With this wholly observed data, a random forest model can be fitted regularly and provide predictions for all possible test-situations then. Regarding the predictive performance in 'Pattern 1' and 'Pattern 3', it is the best in the test-situations with the feature-block 'A' (in both patterns 'A' stands for the 'CNV' block). In 'Pattern 2' the imputation approach has its best performance with the feature-block 'D' (in this pattern 'D' stands for the 'CNV' block), and in 'Pattern 4' it is the best with the feature-block 'B' (in this pattern 'B' stands for the 'CNV' & 'Mutation' block). Therefore, as with the previous two approaches, the performance in the diverse test-situations is always the best in test-situations with the feature-block 'CNV'.

4.1.4 Block-wise Approach

This subsection introduces the results of the block-wise approach for the diverse patterns of block-wise missingness on the TCGA data - details on the approach can be found in section 2.6.
Figure 21 shows the average results of the 5-fold cross-validation for the block-wise approach on the 14 TCGA data sets for the various patterns of block-wise missingness. For each test-situation, the results of the block-wise approach are shown structured according to the weight metric used to combine the block-wise predictions. The first thing that stands out is that the block-wise approach can provide predictions for any test-situation. As the block-wise approach combines the predictions of the single feature-blocks it has been trained with, it is flexibly applicable in different test-situations. Furthermore, the performance in the test-situations with a single feature-block only - e.g. 'single_A' - is equal for the different weight metrics used for the aggregation of the block-wise predictions. That is meaningful, as the predictions in these test-situations only depend on a single feature-block, such that there is no aggregation of block-wise predictions. Regarding the predictive performance, it is clear to see that it is the best when using the 'F-1 Score' as weight metric. For all patterns of block-wise missingness and almost all test-situations, the median F-1 score is the best with the 'F-1 Score' as weight metric. The only situations in which the block-wise approach with the 'F-1 Score' as weight metric does not have the best results are the test-situations with a single feature-block ('single_A' - 'single_CL'). In these test-situations, the used weight metric has no influence, as there is no aggregation of the block-wise predictions. Regarding the predictive performance, it is the same as with the previous approaches. In 'Pattern 1' and 'Pattern 3' the predictive performance is the best in the test-situations

with the feature-block 'A' (in both patterns 'A' stands for the 'CNV' block). While in 'Pattern 2' the block-wise approach has the best performance with the feature-block 'D' (in this pattern 'D' stands for the 'CNV' block). For 'Pattern 4' it is the best with the feature-block 'B' (in this pattern 'B' stands for the 'CNV' & 'Mutation' block).

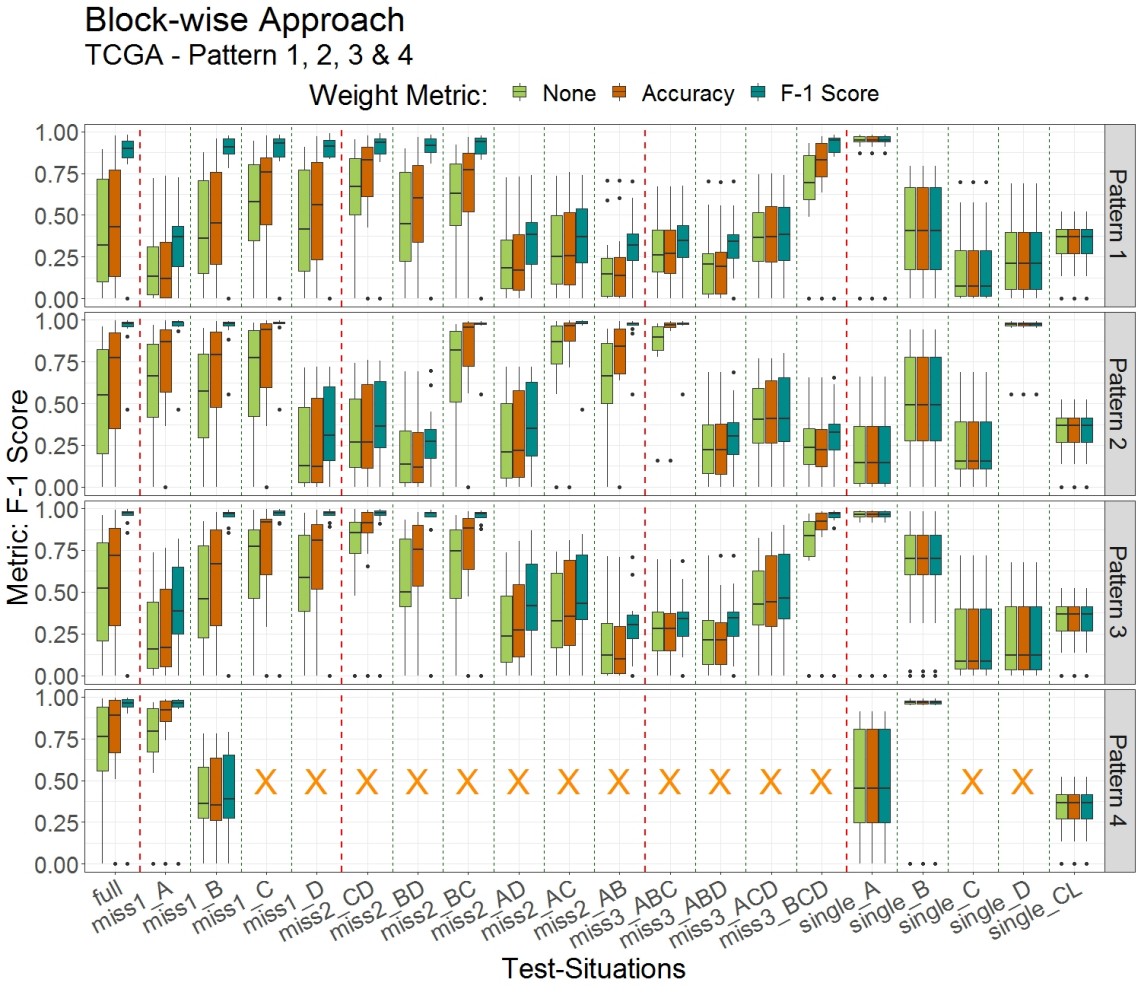

Figure 21: Results of the 'Block-wise' approach on the TCGA data with induced block-wise missingness according to pattern 1, 2, 3 & 4. Test-Situations that are not available for 'Pattern 4' are marked with an orange cross.

4.1.5 Fold-wise Approach

This subsection introduces the results of the fold-wise approach for the diverse patterns of block-wise missingness on the TCGA data - details on the approach can be found in section 2.7.
Figure 22 shows the average results of the 5-fold cross-validation for the block-wise approach on the 14 TCGA data sets for the various patterns of block-wise missingness. For each test-situation, the results of the fold-wise approach are shown structured according to the weight metric used to combine the fold-wise predictions. As the fold-wise approach combines the predictions of its diverse fold-wise fitted random forest models, it is flexibly applicable and can provide predictions for any test-situation. Furthermore, the performance of the fold-wise approach is identical for the different weight metrics in the test-situations that only consist of feature-blocks that are only available for a single fold in the training data. For example, with 'Pattern 1' the performance in the test-situations 'single_A', 'single_B', 'single_C' and 'single_D' is equal for the different weight metrics. That is reasonable, as in these test-situations only a single fold-wise fitted random forest model can create predictions, as the training-set only has a single fold that can be used for these test-situations. Hence there is only one fold in the training data with an observed feature-block 'A', such that only this fold can be used to train a random forest model and create a prediction for the test-set 'single_A' then. Regarding the predictive performance, it is quite clear to see that it is the best when using the 'F-1 Score' as weight metric. With 'Pattern 1' the median F-1 score is the best in 15 out of the 20 test-situations with the 'F-1 Score' as weight metric. With 'Pattern 2' the median F-1 score is the best in 9 out of the 20 test-situations with the 'F-1 Score' as weight metric, and with 'Pattern 3' it is the best in 8 out of the 20 test-situations with the 'F-1 Score' as weight metric. In 'Pattern 4' median F-1 score is once the best with the 'F-1 Score' as weight metric, and once when using no weight metric. With the 'Accuracy' as weight metric, the results were never the best in any pattern. As with the previous approaches, in 'Pattern 1' and 'Pattern 3' the predictive performance is the best in the test-situations that contain the feature-block 'A' (in both patterns 'A' stands for the 'CNV' block). While in 'Pattern 2' the block-wise approach has the best performance with the feature-block 'D' (in this pattern 'D' stands for the 'CNV' block). For 'Pattern 4' it is the best with the feature-block 'B' (in this pattern 'B' stands for the 'CNV' & 'Mutation' block).

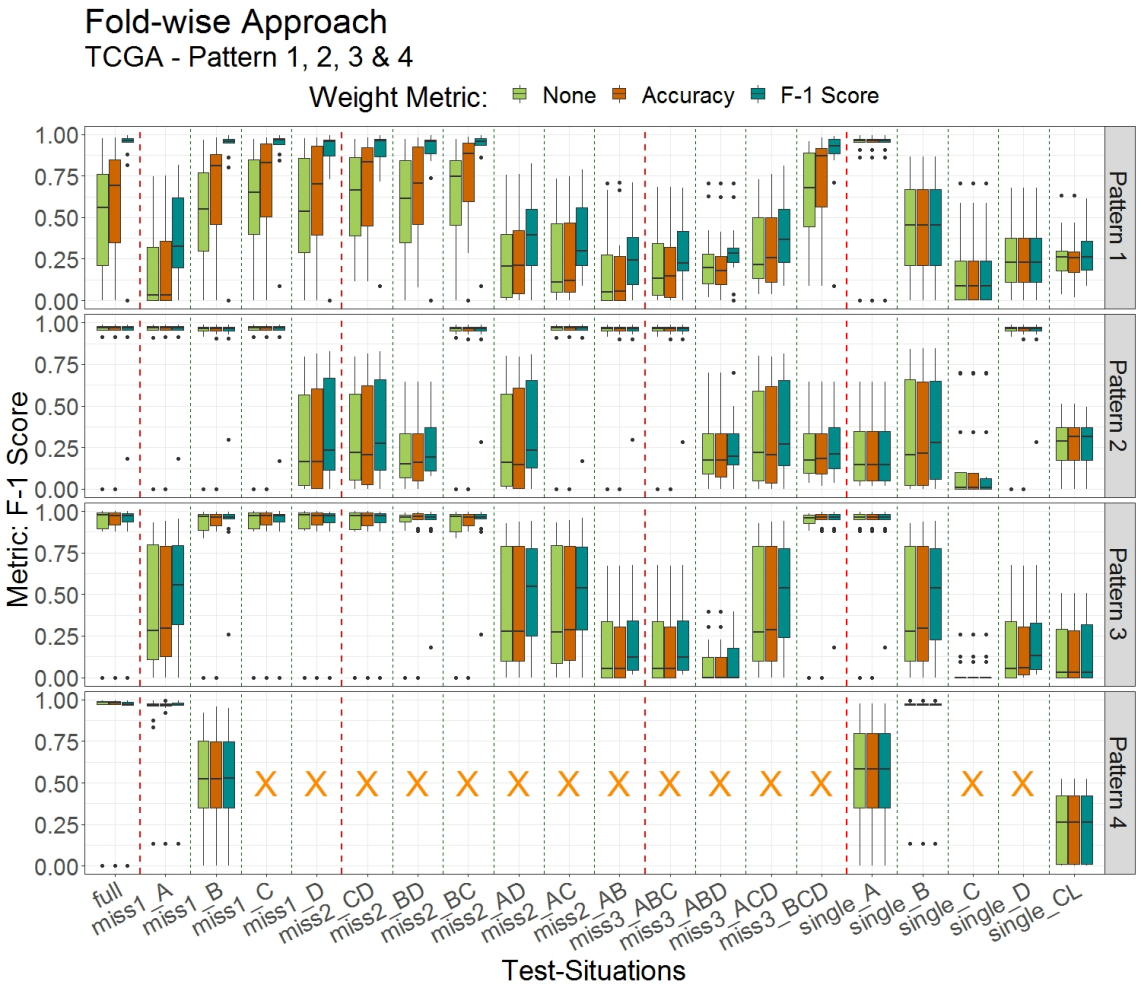

Figure 22: Results of the 'Fold-wise' approach on the TCGA data with induced block-wise missingness according to pattern 1, 2, 3 & 4. Test-Situations that are not available for 'Pattern 4' are marked with an orange cross.

4.1.6 Comparison of the Approaches

This part of the thesis compares the results of the different random forest based approaches on the TCGA data for its various patterns of block-wise missingness. For the comparison of the diverse approaches, only the best settings of each approach are used. For the block- & fold-wise approach the results were the best with the 'F-1 Score' as weight metric, and for the single-block approach, the performance was the best, when using the feature-block 'CNV' - in 'Pattern 1' and 'Pattern 3' 'A' stands for the 'CNV' block | in

'Pattern 2' 'D' stands for the 'CNV' block | in 'Pattern 4' 'B' stands for the 'CNV' and 'Mutation' block.

Pattern 1

Figure 23 shows the boxplots for the average results of the 5-fold cross-validation of the diverse approaches on the 14 TCGA data sets with block-wise missingness in the training-set according to 'Pattern 1'. In the test-situations that do not contain the feature-block 'A', the predictive performance is in general much worse, as in the test-situations that contain the feature-block 'A' (in this pattern 'A' stands for the 'CNV' block). The single-block and complete-case approach can not create predictions for all test-situations, while the fold-wise, block-wise, and imputation approach can do so. Regarding the predictive performance, the fold-wise approach has the best median metric in 11/20 test-situations, the block-wise approach has the best median metric in 3/20 test-situations, and the imputation approach has the best median metric in 4/20 test-situations. The single-block approach never has the best median performance in any of the test-situations, while the complete-case approach has the best median performance in only one. In this setting, it is clear to say that the fold-wise approach has the best predictive performance. The second-best approach in this scenario is the imputation approach which is only marginally better than the block-wise approach. The single-block and complete-case approach have a rather bad predictive performance compared to the other three approaches. Additionally, these approaches have the drawback that they can not generate predictions for all possible test-situations.
Summary: The fold-wise approach has by far the best predictive performance and can create predictions for every test-situation. The imputation and block-wise approach can create predictions for all test-situations as well but have a worse predictive performance than the fold-wise approach. The complete-case and single-block approach are inflexible and can only provide predictions in certain test-situations. But even in these test-situations, the approaches are usually worse than at least one of the remaining approaches. The results with the balanced accuracy/ MCC as metric are similar - corresponding figures A-3/ A-4 can be found in the attachment.

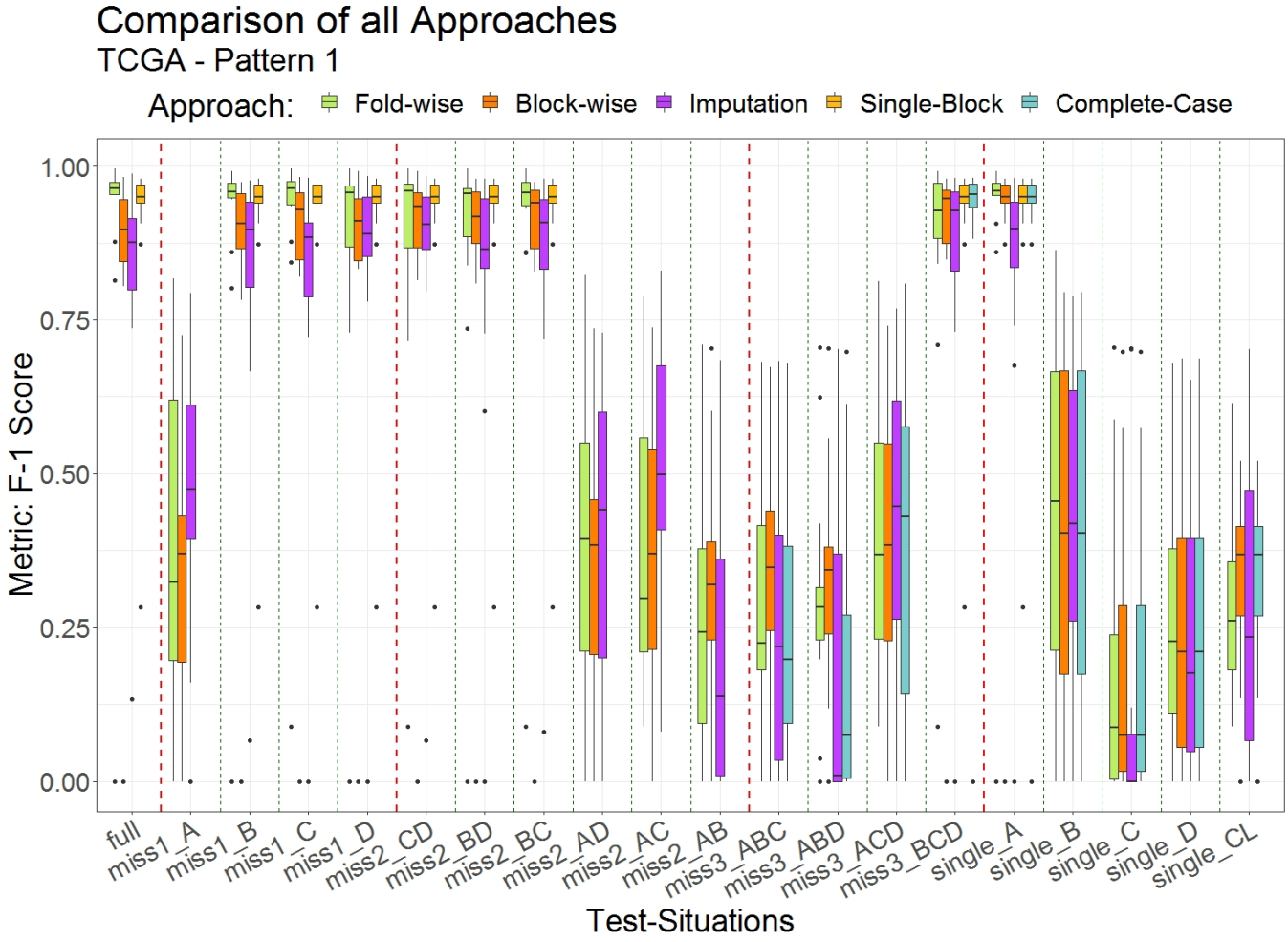

Figure 23: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 1.

Pattern 2

Figure 24 shows the boxplots for the average results of the 5-fold cross-validation of the diverse approaches on the 14 TCGA data sets with block-wise missingness in the training-set according to 'Pattern 2'. In the test-situations that do not contain the feature-block 'D', the predictive performance is in general much worse, as in the test-situations that contain the feature-block 'D' (in this pattern 'D' stands for the 'CNV' block). The complete-case approach can not create predictions for all test-situations, while the other four approaches can do so. Regarding the predictive performance, the imputation and block-wise approach have each the best

median F-1 score in 8/20 test-situations each. The fold-wise, complete-case, and single-block approaches never have the best median F-1 score in any test-situation. In this setting, it is clear to say then that the imputation and block-wise approach have the best predictive performance. The fold-wise approach performs really poor in this setting and can be compared with the single-block and complete-case approach.

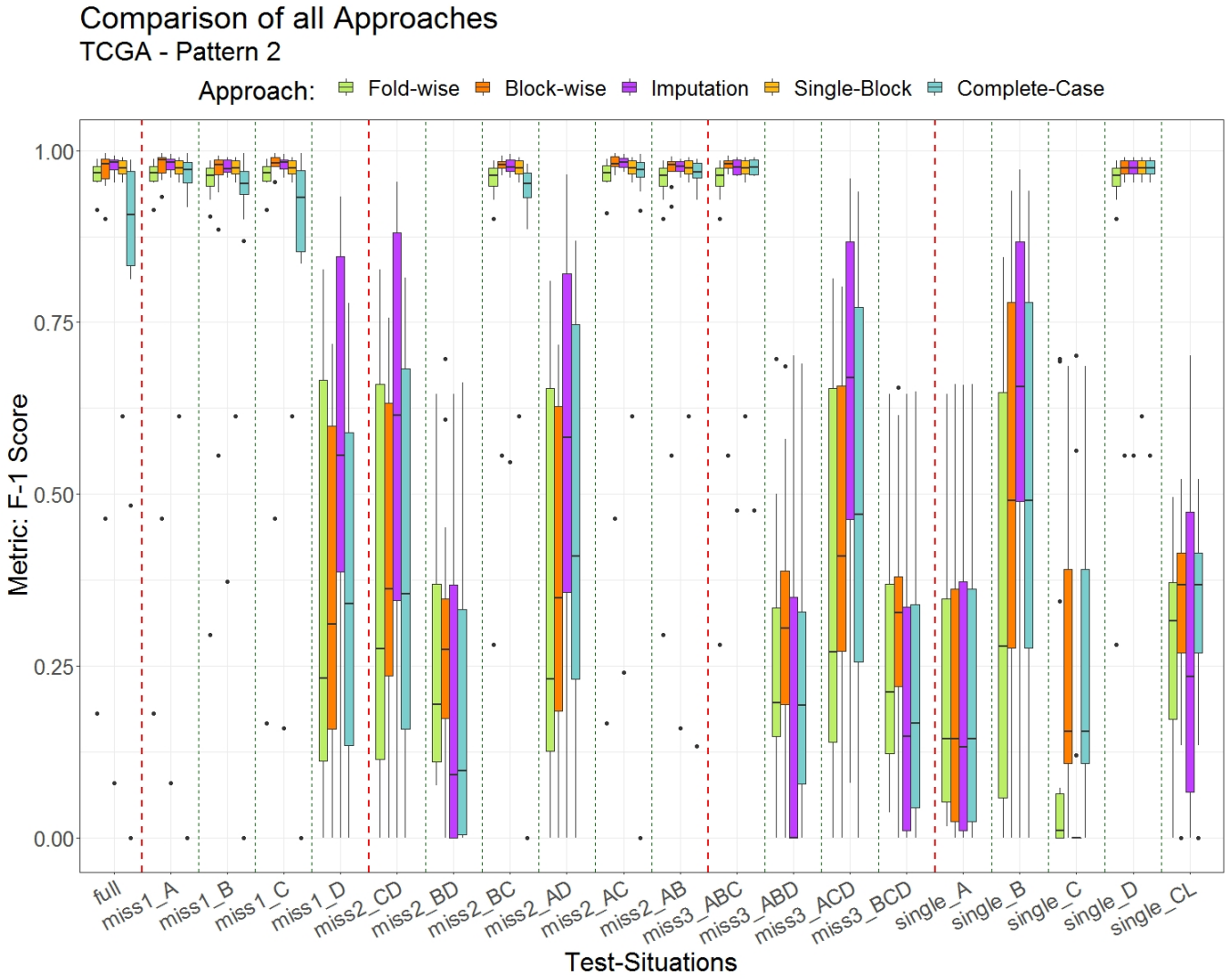

Figure 24: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 2.

Summary: The imputation and block-wise approach have the best predictive performance and can create predictions for every possible test-situation. The fold-wise and complete-case approach can create predictions for all test-situations as well but have a worse predictive performance. The single-block approach is inflexible and can only provide predictions in certain test-situations - regarding the predictive performance, the single-block approach never leads to the best results. The results with the balanced accuracy/ MCC as metric are similar - corresponding figures A-5/ A-6 can be found in the attachment.

Pattern 3

Figure 25 shows the boxplots for the average results of the 5-fold cross-validation of the diverse approaches on the 14 TCGA data sets with block-wise missingness in the training-set according to 'Pattern 3'. In the test-situations that do not contain the feature-block 'A' the predictive performance is in general much worse, as in the test-situations that contain the feature-block 'A' (in this pattern 'A' stands for the 'CNV' block). Only the single-block and complete-case approach can not create predictions for all test-situations. Regarding the predictive performance, the block-wise approach has the best median F-1 score in ten out of the 20 test-situations. The imputation approach has the best performance in four test-situations, while the fold-wise approach has the best performance in only two test-situations. The complete-case and single-block approaches never have the best median F-1 score. In this setting, it is clear to say then that the block-wise approach has the best predictive performance. The single-block approach can provide predictions only for the test-situations with the feature-block 'A' and there is no single test-situation where this approach has the best median F-1 score.
Summary: The block-wise approach has the best predictive performance and can create predictions for every possible test-situation. The imputation and fold-wise approach can create predictions for all test-situations as well, but have a worse predictive performance than the block-wise approach. The single-block approach is inflexible and can only provide predictions in certain test-situations - regarding the predictive performance the single-block approach never leads to the best results. The results with the balanced accuracy/ MCC as metric are similar - corresponding figures A-7/ A-8 can be found in the attachment.

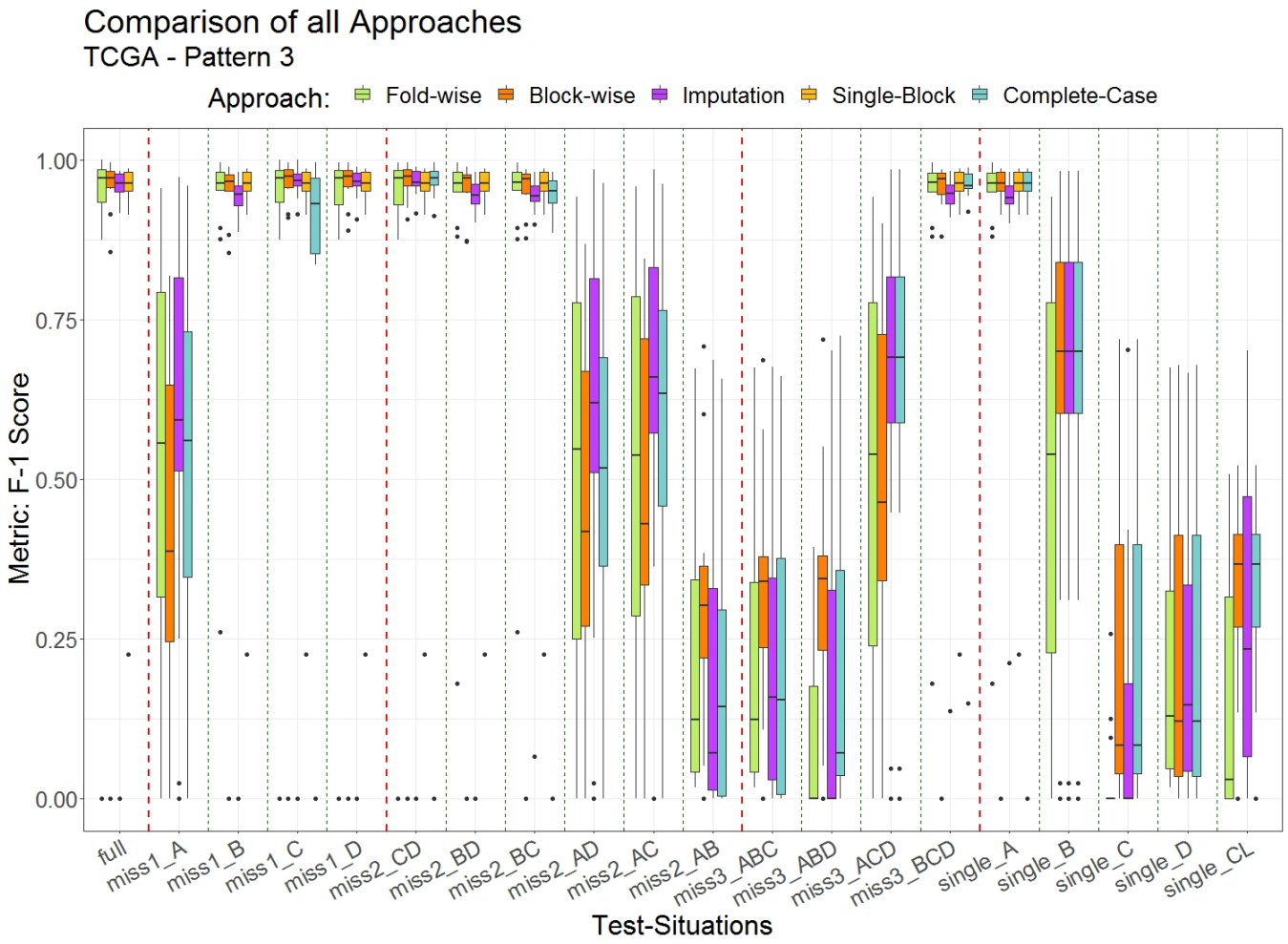

Figure 25: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 3.

Pattern 4

Figure 26 shows the boxplots for the average results of the 5-fold cross-validation of the diverse approaches on the 14 TCGA data sets with block-wise missingness in the training-set according to 'Pattern 4'. In the test-situations that do not contain the feature-block 'B', the predictive performance is in general much worse, as in the test-situations that contain the feature-block 'B' (in this pattern 'B' stands for the 'CNV' & 'Mutation' block). Only the single-block and complete-case approach can not create predictions for all test-situations. Regarding the predictive performance, the fold-wise approach has the best median F-1 score in four out of the six

test-situations. The complete-case approach has the best median F-1 score once. In contrast, the remaining three approaches (block-wise, imputation and single-block) never resulted in the best median F-1 score in any test-situation. In this setting, it is clear to say that the fold-wise approach has the best predictive performance. The single-block approach can provide predictions only for the test-situations with the feature-block 'A', and there is no single test-situation where this approach has the best median F-1 score.

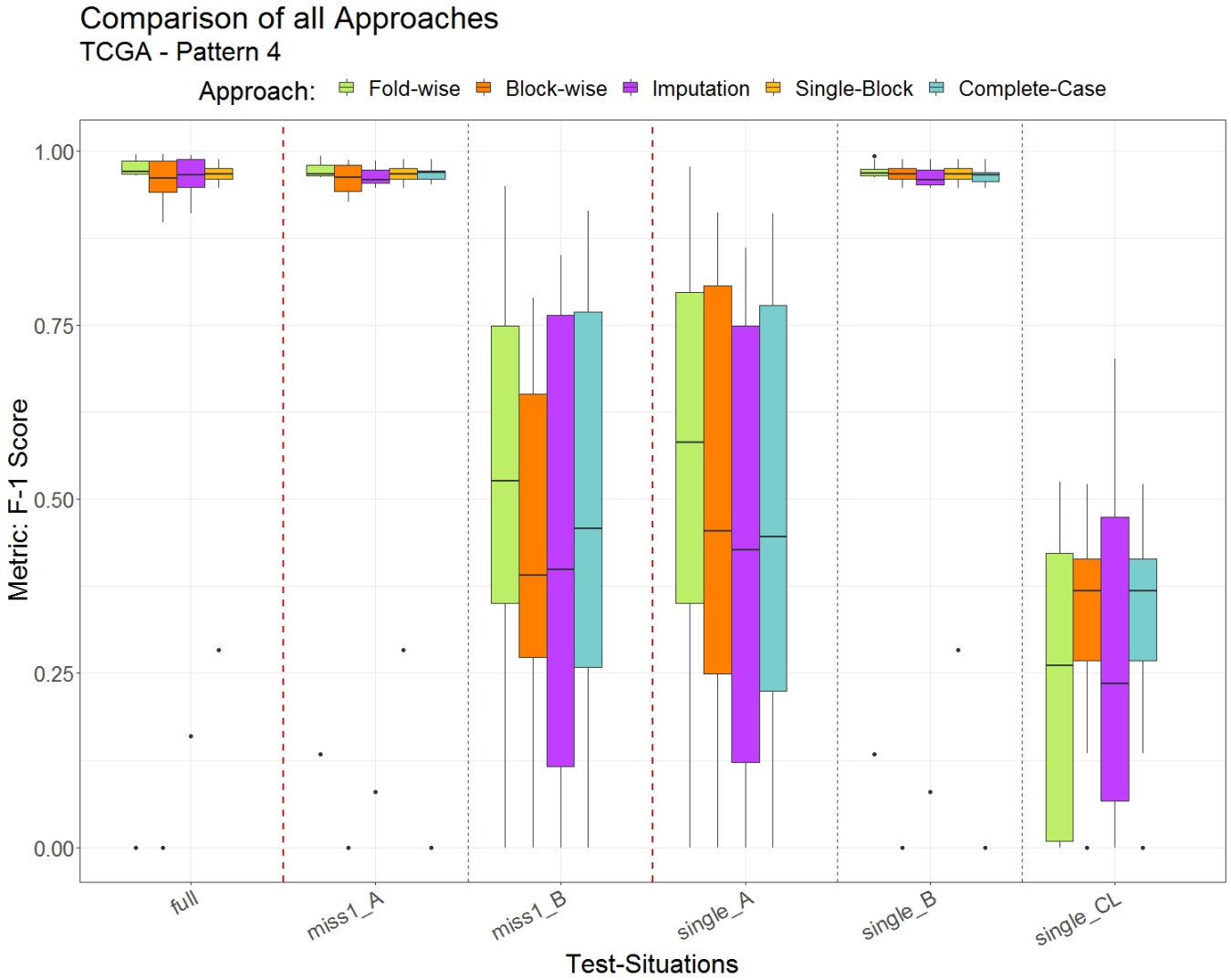

Figure 26: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 4.

Summary: The fold-wise approach has the best predictive performance and can create predictions for every possible test-situation. The imputation and block-wise approach can create predictions for all test-situations as well but never have the best median F-1 score. The single-block and complete-case approach are inflexible and can only provide predictions in certain test-situations - regarding the predictive performance the single-block approach never leads to the best results, while the complete-case approach has the best median F-1 score in one test-situation. The results with the balanced accuracy/ MCC as metric are similar - corresponding figures A-9/ A-10 can be found in the attachment.

4.2 Clinical asthma data

This subsection contains all cross-validation results on the clinical asthma data for the different random forest based approaches, as well as for the adaptions of the priority-Lasso from Hagenberg [1] and the mdd-sPLS method. Firstly the results of the different approaches are shown and explained separately before the results of the approaches are compared then.

4.2.1 Random Forest

This part of the thesis introduces all results of the different random forest adaptions from this thesis on the clinical asthma data. Detailed information on the evaluation technique on the clinical asthma data can are in the paragraph 'Evaluation Technique' of section 3.2.2.

Complete-Case Approach

This paragraph shows the results of the complete-case approach for the clinical asthma data - details on the approach itself in section 2.3.
Figure 27 shows the results of the 5-fold cross-validation with the complete-case approach on the clinical asthma data. The resulting F-1 scores are between 0.81 and 0.87, whereby the median and mean are slightly above 0.83. As the observed feature-blocks for the observations in the test-set differ, the predictions need to be generated separately for each fold in the test-set. To create predictions for a given fold in the test-set it is checked, whether the training-set contains complete-cases regarding the current fold. If there are complete-cases, a random forest model is trained with these observations then and used to create predictions for the current fold. In the 5-fold cross-validation on the clinical asthma data, there was no single

case, where the complete-case approach could not generate a prediction for a test-observation.

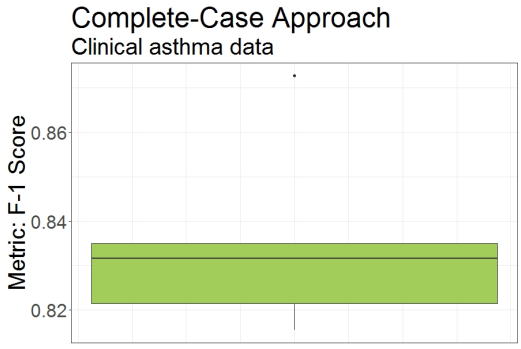

Figure 27: Results of the 'Complete-Case' approach on the clinical asthma data.

Single-Block Approach

This paragraph shows the results of the single-block approach for the clinical asthma data - details on the approach itself in section 2.4.
Figure 28 shows the results of the 5-fold cross-validation with the single-block approach on the clinical asthma data. The x-axis shows the different feature-blocks that have been used for the training of the single-block approach - e.g. the boxplot on the far left in the figure shows the results of the 5-fold cross-validation when training a random forest model on the 'Questionaire' feature-block and using this model to predict on the test-set then. As the observations in the test-set are observed in different feature-blocks, the single-block approach can not always generate predictions for all observations in the test-set. Instead of evaluating the observations for which no prediction could be made as misclassified, the predictions for these observations equal the distribution of the predicted classes in the test-set. E.g. if the model labels three test-observations as 'positive' and seven as 'negative', the predicted probability for the remaining test-observations the model could not create predictions for, is set to 0.7 for 'negative'. The single-block approach with the 'Questionaire' feature-block could always generate predictions for the on average 104.2 observations per held-out-fold. With the 'Clinical routine diagnostics' feature-block, on average one test-observation out of the 104.2 test-observations per held-out fold had to be guessed and with the 'Allergen sensitization' feature-block it were on average 8.2 test-observations per held-out fold. The remaining feature-blocks have only been observed for a small fraction of the observations in the clinical asthma data, such that the average

amount of test-observations that could not be predicted is much higher. For the 'Cytokine expression data', it were 74.4 out of 104.2 test-observations per held-out fold. With the 'Gene expression data I', it were 91 out of 104.2 test-observations per held-out fold and with the 'Gene expression data II' on average even 95. With a median F-1 score of 0.81, the best predictive performance is achieved with the 'Questionaire' feature-block. The second-best performance is achieved with the feature-block 'Allergen sensitization' that has a median F-1 score of 0.77. With any other feature-block, the median F-1 score is below 0.72.

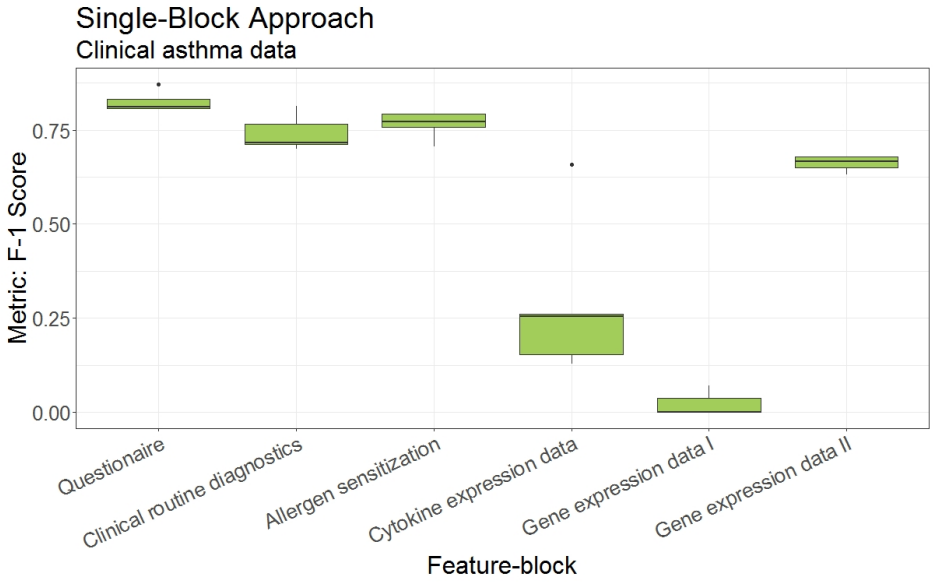

Figure 28: Results of the 'Single-Block' approach on the clinical asthma data.

Imputation Approach

This paragraph shows the results of the imputation approach for the clinical asthma data - details to the approach itself in section 2.5.
Figure 29 shows the results of the 5-fold cross-validation with the imputation approach on the clinical asthma data as a boxplot. The resulting F-1 score metrics are between 0.83 and 0.87, whereby the median is above 0.84 and the mean around 0.85. As the observed feature-blocks for observations in the held-out fold differ, the predictions are generated separately for each fold in the test-set. To create a prediction for a given fold from the test-set, firstly the observed feature-blocks for the given fold have to be extracted. From

the imputed training data, only the available feature-blocks of the current fold are then used to train a random forest model. This model is then able to create predictions for the current fold then. In the 5-fold cross-validation on the clinical asthma data, there was no single case, where the imputation approach could not generate a prediction for a test-observation.

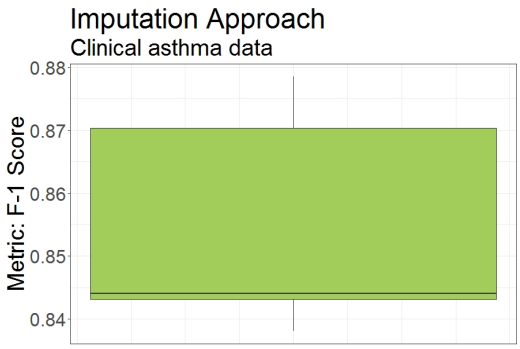

Figure 29: Results of the 'Imputation' approach on the clinical asthma data.

Block-wise Approach

This paragraph shows the results of the block-wise approach for the clinical asthma data - details on the approach itself in section 2.5.
Figure 30 shows the results of the 5-fold cross-validation with the block-wise approach on the clinical asthma data. The results are split by the weight metric used to aggregate the diverse block-wise predictions. With this approach, predictions were always possible for the observations in the test-set, as the block-wise fitted model is flexibly applicable to observations with different patterns of block-wise missingness. The predictive performance with the block-wise approach is the best when using the 'F-1 Score' as weight metric for the aggregation of the block-wise predictions. The median F-1 score metric is around 0.79, and around 0.78 when using the 'Accuracy' as weight-metric. When using no metric for the aggregation of the block-wise predictions, the performance is the worst - the median is 0.77. Even if the results are close, it can be said that the best performance on this data set is achieved when using the 'F-1 Score' as weight metric.

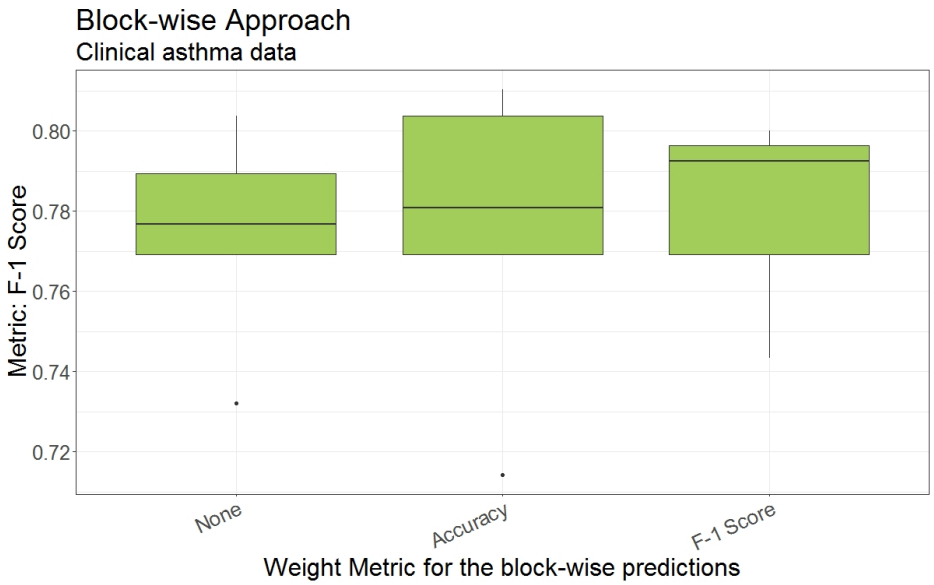

Figure 30: Results of the 'Block-wise' approach on the clinical asthma data.

Fold-wise Approach

This paragraph shows the results of the fold-wise approach for the clinical asthma data - details on the approach itself in section 2.6.
Figure 31 shows the results of the 5-fold cross-validation with the fold-wise approach on the clinical asthma data. The results are split by the weight metric used to aggregate the diverse fold-wise predictions. As the fold-wise approach combines the predictions of its different fold-wise fitted random forest models, it is flexibly applicable and can provide predictions for all observations in each held-out fold. Regarding the predictive performance, it is clear to see that it is the best when using the 'F-1 Score' as weight metric - median F-1 score of˜0.83. With the 'Accuracy' as weight metric, the median is around 0.7, and when using no metric for the aggregation, the median equals 0.65.

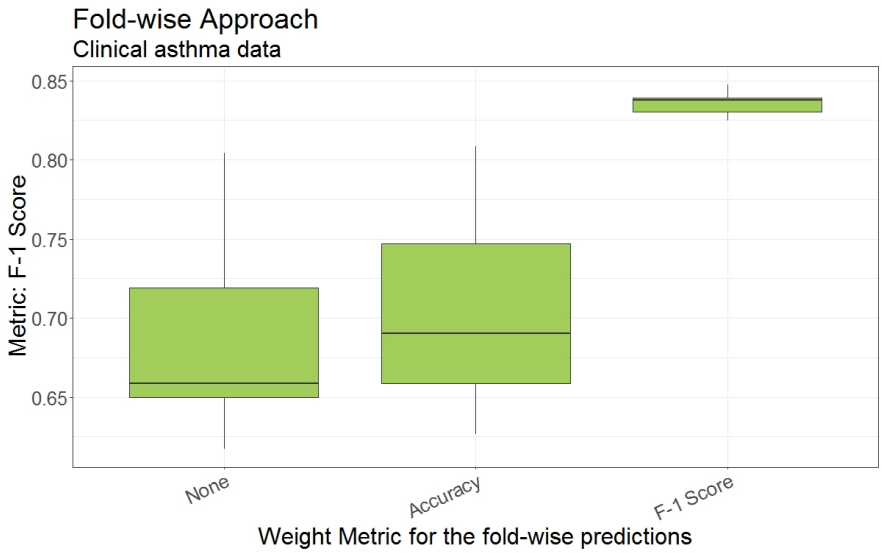

Figure 31: Results of the 'Fold-wise' approach on the clinical asthma data.

4.2.2 Priority-Lasso & mdd-sPLS

This part of the thesis introduces the predictive performances of the different priority-Lasso adaptions from Hagenberg [1] and of the mdd-sPLS method from Lorenzo et al. [2] on the clinical asthma data. These diverse methods can directly deal with block-wise missingness. In the beginning, the theoretical principles are briefly explained, before the predictive performance of the different approaches is investigated then.

Theoretical principles - Priority-Lasso

The priority-Lasso is an extension of the lasso method. Lasso stands for *least absolute shrinkage and selection operator* and was introduced by Tibshirani in 1996 [49]. The lasso method is a regularised least squares estimator that adds the absolute values of all parameters as an additional penalty. The loss function is defined as:

$$\|y - \beta_0 \mathbb{1} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{13}$$

In the equation 13 the parameter λ is a hyperparameter ≥ 0. The bigger λ, the stronger the added penalty for each $\beta \neq 0$. The additional penalty leads

to two major advantages over the regular least squares estimator. Firstly, the lasso model can be fitted on data with more features than observations. Secondly, the lasso method does variable selection, as only parameters of the important variables are estimated with a value different from 0.
In 2018 the priority-Lasso was introduced by Klau et al. [19] as an extension of the classical lasso method. The priority-Lasso can be applied to data with different feature-blocks. It is described as a "hierarchical regression method which builds prediction rules for patient outcomes [...] from different blocks of variables including high-throughput molecular data" [[19], p. 2]. Firstly the feature-blocks of the data are ordered in descending order of priority for the target variable - the stronger the relation of a feature-block to the target variable, the higher its priority. Then a separate lasso model is fitted to each of these feature-blocks. The models that were fitted on the later blocks (the later, the lower the importance of the block for the target variable) are only used to improve the prediction from the previous blocks. "In order to assure that later blocks only improve the prediction of the previous blocks, the fitted linear predictor of block m is used as the offset when fitting block m + 1" [1]. More details on this are in the second chapter of Hagenberg's thesis [1] or in Klau et al. [19].
Hagenberg [1] developed different adaptions of the priority-Lasso such that it can directly deal with block-wise missingness. As for the priority-Lasso, a model is fitted on each feature-block of the data. Mainly there are two methods to deal with the missing data:

(1) Ignore: The "lasso model for every block is only fitted with the observations that have no missing values" [1]. For observations that have missing values in the current feature-block, no offset can be calculated for the next block. For these observations the offset (incl. intercept) "from the previous block is carried forward" [1]. For observations that lack values of the first feature-block, the offset for the second feature-block is either set to zero - *PL-ignore, zero* - or to the estimated intercept of the first feature-block - *PL-ignore, intercept*.

(2) Impute: As in the "Ignore" approach, the lasso model fitted on each feature-block only uses the observations that have no missing values in this block. For observations with missing values, the offset from the previous block is not carried forward but imputed. "This has the advantage that instead of imputing a possibly very high number of covariates, only one value [- the offset -] is imputed" [1]. In the beginning, the unique patterns of block-wise missingness Q_m are extracted from all the observations that miss the feature-block m. Then for each pattern of block-wise missingness $q \in Q_m$ all

observations from the training data that can be used for an imputation model are found. These observations need to be observed in the feature-block m and in at least one feature-block that is observed in the pattern q. Then for every pattern of block-wise missingness, it is counted how many observations can be used for the imputation model. The pattern q that the training observations must have to be usable for the imputation model is either chosen according to the pattern "that has the most observations that can be used for the imputation model" [1] [*max. n*] or according to the pattern "that uses the most high priority blocks" [*max. blocks*] [1].

In total there are four different adaptions of the priority-Lasso - 'ignore, zero', 'ignore, intercept', 'impute, max. n' and 'impute. max. blocks'. All models are fitted sequentially and only differ in the way how the offset is carried forward for observations with missing values and how predictions are made. Closer information on the theoretical principles of these adoptions are in chapter three of Hagenberg's thesis [1].

Theoretical principles - mdd-sPLS

The mdd-sPLS method was introduced by Lorenzo et al. in 2019 [2] and can be applied in "supervised high dimensional settings with a large number of variables and a low number of individuals" [[2], p. 1]. It can either be used for variable selection or as a prediction model. In the thesis of Hagenberg [1], it is used as a prediction method for the comparison with the different priority-Lasso adaptions. The mdd-sPLS method "is based on singular value decomposition (SVD) of the covariance matrix between the outcome Y and the covariates X" [1]. The maximising of the covariance between X and Y leads to latent variables for the covariates and the target variable. "The latent variables can then be used to approximate X and Y and regress $\hat{Y}$ onto $\hat{X}$ [(latent variables)]" [1]. Lorenzo et al. [2] use a soft-threshold covariance matrix for the SVD, such that all entries with an absolute value smaller than a threshold are set to zero - the method is called 'covariance-thresholding sparse PLS' (ct-sPLS). To combine the information from multiple feature-blocks, the ct-sPLS method is applied separately to all blocks. For the combination of the information in the different blocks "the weights from the SVD used to generate the latent variables are pooled. The pooling is performed in such a way that the subsequent regression is done in a common subspace over all blocks" [1]. This method is called 'Multi-Data-Driven-sPLS (mdd-sPLS)'. Missing values in the training data are mean imputed before the mdd-sPLS algorithm is applied. Only variables that have been initially missing and received a mean value as a first imputation get improved imputations then. To

do this, another mdd-sPLS model is built. This model predicts the "variables used in the first model from information about the outcome and is used to impute the missing values. The process of building a model to predict the outcome and then another model to impute the missing variables is repeated until convergence of the latent variables of the covariates" [1].
Missing values in the test data are imputed by a mdd-sPLS model on the observed data. Based on the complete test-set, predictions can then be generated.

The mdd-sPLS method can be fitted directly on data with block-wise missing values and provide predictions for test observations with block-wise missing values. Further details on the theoretical properties can either be found in [2] or in Hagenberg's thesis [1].

Naive block priority

In this setting, the analysis was performed with a block order that gives a higher priority to feature-blocks with less missing observations. This is a somewhat naive assumption, as the priority of the feature-blocks should be justified by a professional - e.g. a doctor. The results of the 5-fold cross-validation on the clinical asthma data for the different priority-Lasso adaptions and the mdd-sPLS method are shown as boxplots in figure 32. The x-axis shows the different approaches and the y-axis the corresponding F-1 scores. The overall F-1 scores are between 0.75 and 0.88. Among the priority-Lasso adaptions, the 'PL - impute, maximise blocks' approach has the worst median F-1 score with 0.81. The second worst adaption is the 'PL - impute, maximise n' with a median F-1 score of 0.82. Hence the 'imputation' priority-Lasso adaptions perform worse than the 'ignore' adaptions. The median F-1 score for the 'PL - ignore, zero' approach is 0.82 and only slightly lower than of the 'PL - ignore, intercept' approach that has a median F-1 score of 0.83. The mdd-sPLS method has a median F-1 score of 0.81 and is only slightly better than the 'PL - impute, maximise blocks' adaption.
In this setting the 'ignore' priority-Lasso adaptions perform better than the 'imputation' adaptions. The best performance is achieved with the 'PL - ignore, intercept' approach that has a median F-1 score of 0.83. The mdd-sPLS approach is the second-worst approach and is only better than the worst priority-Lasso adaption - 'PL - impute, maximise blocks'.

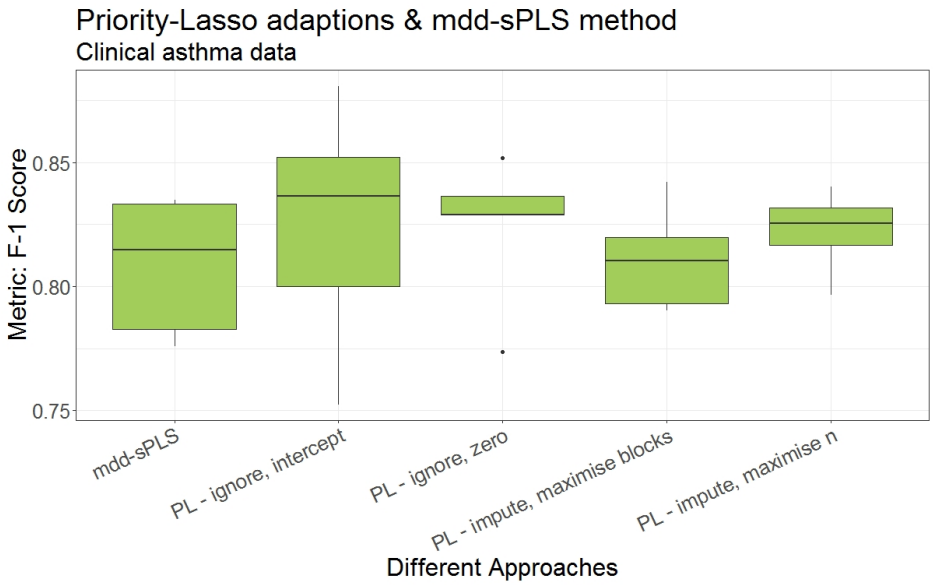

Figure 32: Results of the priority-Lasso adaptions and the mdd-sPLS approach on the clinical asthma data.

Influence of using different block combinations for the prediction

In this setting, the influence of different block combinations for the prediction is investigated. The priority of the feature-blocks is the same naive one as in the previous paragraph. The different priority-Lasso adaptions are trained on all blocks, but the predictions are "based on a subset of the blocks, adding one after the other" [1]. The results for the different priority-Lasso adaptions are shown as boxplots in figure 33. The x-axis shows the different priority-Lasso adaptions and the y-axis the corresponding F-1 scores. For each approach, the F-1 score is shown for the predictions based on the different block combinations. E.g. the red boxplots show the results of the diverse approaches when only using block 1 for the predictions, whereby the green boxplots show the results of the same approaches when using block 1, 2 and 3 for the predictions.
When using only block 1 for the prediction, the median F-1 score for the 'ignore' priority-Lasso adaptions are around 0.81, and for the 'imputation' priority-Lasso adaptions around 0.82. The addition of block 2 increases the median F-1 score for all approaches except for 'PL - impute, maximise blocks'. The addition of block 3 increases the median F-1 scores for all approaches. When additionally using block 4 for the prediction, it worsens the median F-1 score for all approaches except for 'PL - ignore, intercept'. The additions

of block 5 and 6 lead to a significant decrease in the predictive performance for all the different approaches.
The predictive performance is the best for all approaches when only using the feature-blocks 1, 2 and 3 for the predictions. As in the previous setting, the 'ignore' priority-Lasso adaptions perform better than the 'imputation' adaptions. The best predictive performance is achieved with the 'PL - ignore, zero' approach that has a median F-1 score of 0.88. As the mdd-sPLS method always uses all feature-blocks for the prediction, the performance in this setting is the same as in the previous paragraph - median F-1 score of 0.81.

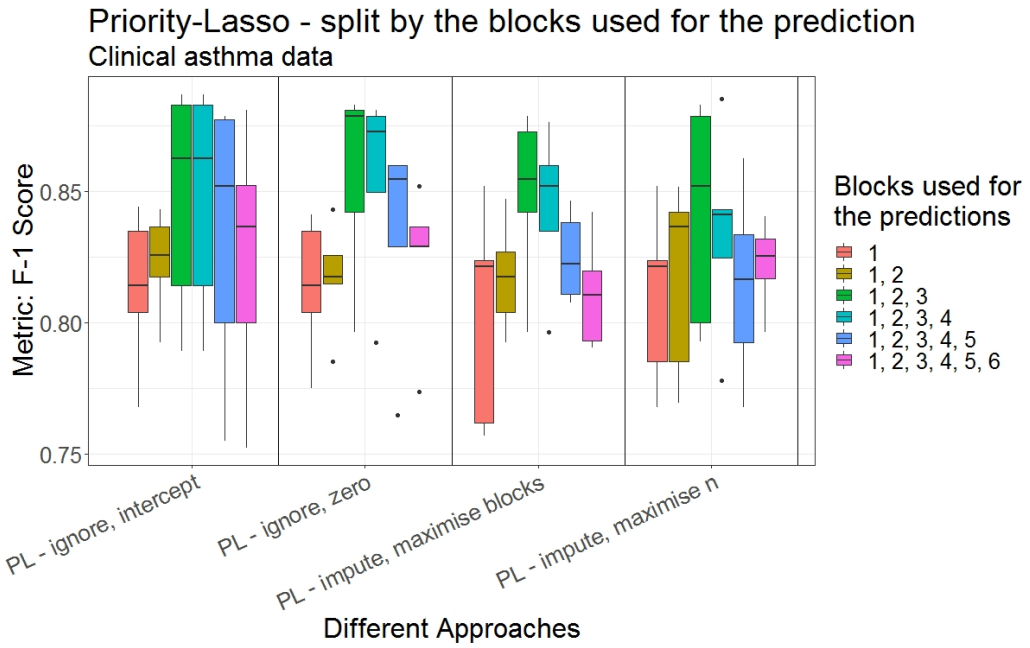

Figure 33: Results of the priority-Lasso adaptions on the clinical asthma data, when using subsets of blocks for the predictions.

Influence of different block priorities

One of the key features of the priority-Lasso is that the priority of the blocks can be set according to the needs of the user. In the previous two paragraphs, the priority of the blocks was set according to the number of missing observations in the feature-blocks. This does not necessarily reflect a professional opinion. Therefore the analysis of the approaches was repeated for a different order of the block priorities. The priority of the blocks was set according to the opinion of the project partner and set as follows (ascending order): cytokine expression data (ID 4), clinical routine diagnostics (ID 2),

questionnaire (ID 1), and lastly the allergen sensitisation (ID 3), the gene expression data I (ID 5) and the gene expression data II (ID 6) have the same priority. "With this block order, not all models with both gene expression data blocks could be estimated. Therefore, only the [...] block combinations where only one gene expression data block is included were analysed" [1]. Thus one of the analyses is based on the feature-blocks 1, 2, 3, 4 and 5 and the other one based on 1, 2, 3, 4 and 6.

Block Priority [4, 2, 1, 3, 5]: In this part, the predictive performance of the different priority-Lasso adaptions and the mdd-sPLS approach based on the feature-blocks 1, 2, 3, 4 and 5 are analysed - the priority is set to 4, 2, 1, 3, 5. The predictive performances of the different approaches are shown as boxplots in figure 34. The x-axis shows the different approaches and the y-axis the corresponding F-1 scores. For each approach, the F-1 scores are shown separately when the "predictions are based only on the block with the highest priority, based on the blocks with the two highest priorities and so on" [1].
The mdd-sPLS method always uses all feature-blocks for the prediction, such that there is only one single boxplot for this approach. The median F-1 score is around 0.81 and hence more or less the same as with the 'naive block priority'. For the different priority-Lasso adaptions, the predictive performance based only on block 4 is rather bad. Adding the blocks 2 and 1 increases the predictive performances respectively for all approaches. The addition of block 3 does not change too much, while the addition of block 5 has two different effects. While the 'ignore' priority-Lasso adaptions can increase their median F-1 score, the median F-1 score of the 'imputation' adaptions is heavily reduced. The best predictive performance of the 'ignore' priority-Lasso adaptions is achieved on all five feature-blocks - the median F-1 score for 'PL - ignore, zero' is around 0.85 and slightly above the median of the 'PL - ignore, intercept' approach. The 'imputation' priority-Lasso adaptions have the best predictive performance on the feature-blocks 4, 2, 1 and 3 - the predictive performance is the same for both approaches, and the F-1 scores are only slightly below 0.85. Therefore the 'PL - ignore, zero' achieves the best performance in this setting. The differences between the predictive performance of the diverse priority-Lasso approaches are shallow in this setting.

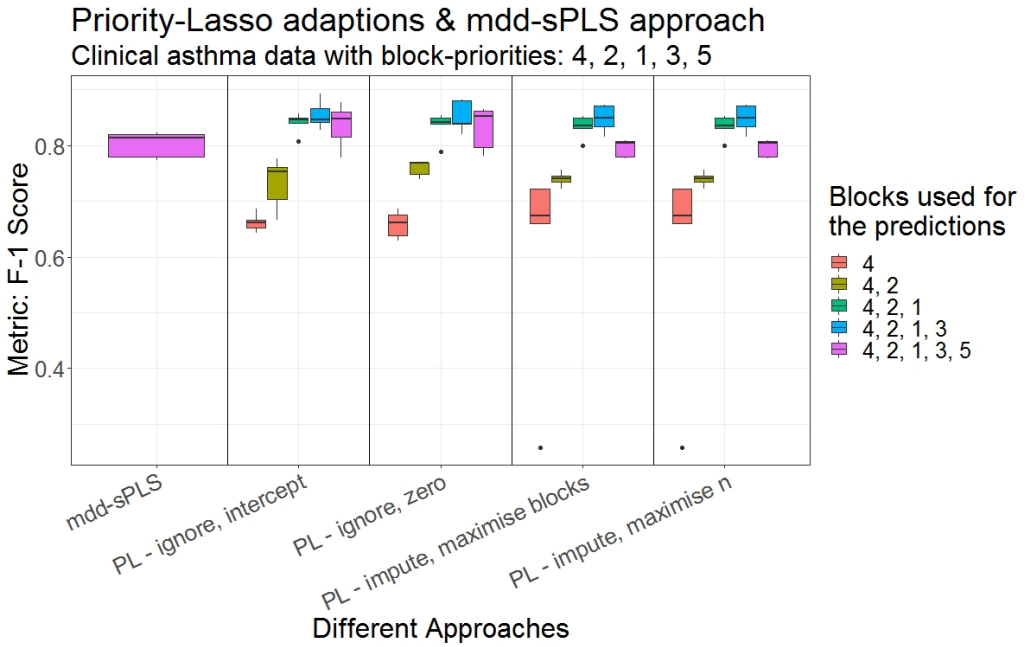

Figure 34: Results of the priority-Lasso adaptions and the mdd-sPLS approach on the clinical asthma data, when using subsets of blocks for the predictions and setting the block priorities according to the opinion of the project partner.

Block Priority [4, 2, 1, 3, 6]: In this part, the predictive performance of the different priority-Lasso adaptions and the mdd-sPLS approach based on the feature-blocks 1, 2, 3, 4 and 6 are analysed - the priority is set to 4, 2, 1, 3, 6. The predictive performances of the different approaches are shown as boxplots in figure 35. The x-axis shows the different approaches and the y-axis the corresponding F-1 scores. For each approach, the F-1 scores are shown separately when the "predictions are based only on the block with the highest priority, based on the blocks with the two highest priorities and so on" [1].
The mdd-sPLS method always uses all feature-blocks for the prediction, such that there is only one single boxplot for this approach. The median F-1 score is around 0.83 and therefore 0.02 higher than with the priority order [4, 2, 1, 3, 5], as well as with the naive block priority. For the different priority-Lasso adaptions, the predictive performance based on the first four blocks is exactly the same as in the paragraph before - the more of the first four blocks are used for the prediction, the better the predictive performance. But the addition of block 6 decreases the predictive performance for all

priority-Lasso adaptions. Therefore the best performance of the priority-Lasso adaptions is achieved again based on the feature-blocks 4, 2, 1 and 3. As in the paragraph before, both 'ignore' priority-Lasso adaptions, as well as both 'imputation' priority-Lasso adaptions have a median F-1 score of around 0.85. The 'PL - ignore, intercept' approach is again minimally better than the other approaches regarding the median and mean F-1 score.

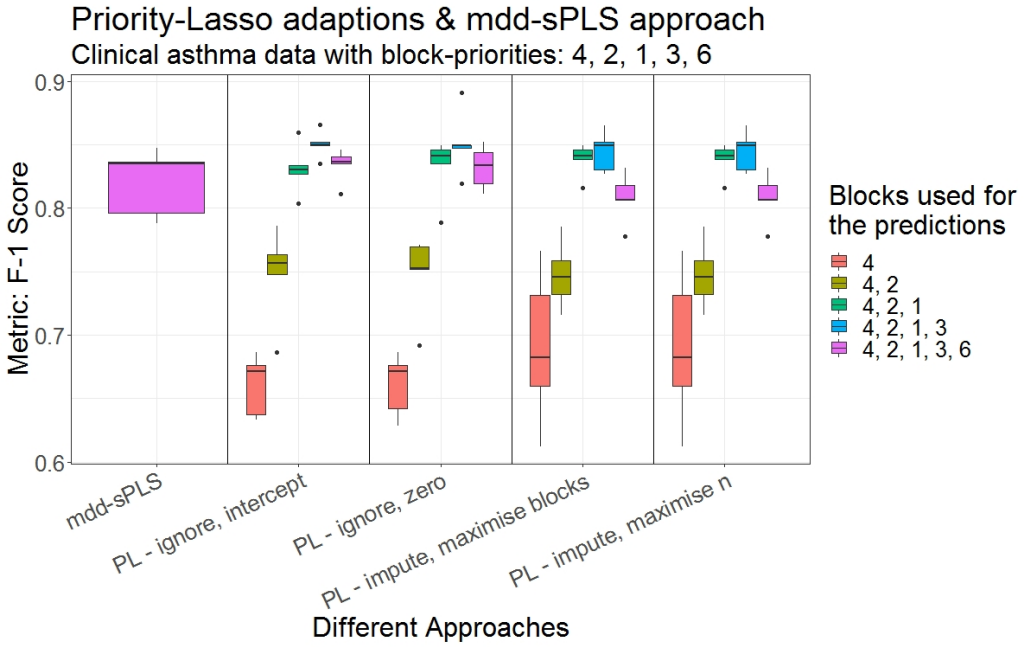

Figure 35: Results of the priority-Lasso adaptions and the mdd-sPLS approach on the clinical asthma data, when using subsets of blocks for the predictions and setting the block priorities according to the opinion of the project partner.

4.2.3 Comparison of the Approaches

This chapter compares the different approaches from Hagenberg's thesis [1] and the diverse random forest based approaches from this thesis based on the clinical asthma data.
Only the best approaches of Hagenberg's and this thesis are used for the comparison on the clinical asthma data. The single-block approach had the best performance based on the 'questionaire' feature-block. The block- and fold-wise approach had the best results with the 'F-1 Score' as weight metric. With the naive block order, the 'PL - ignore, intercept' approach had the best performance on the whole set and the 'PL - ignore, zero' approach with

the feature-blocks 1, 2 and 3 for the prediction. With the adjusted block priorities, [4, 2, 1, 3, 5] the performance was the best for the 'PL - ignore, zero' approach on all five blocks. With the adjusted block priorities [4, 2, 1, 3, 6] the performance was the best for the 'PL - ignore, intercept' approach on the first four blocks. The mdd-sPLS method had the best performance with the adjusted block priorities [4, 2, 1, 3, 6].
The predictive performances of the different approaches are shown as boxplots in figure 36. The x-axis shows the different approaches and the y-axis the corresponding F-1 scores. The approaches that are in the focus of this thesis are green, the priority-Lasso adaptions from Hagenberg's thesis [1] are purple, and the mdd-sPLS method is dark golden. For the single-block approach, the feature-block that was used is noted in the square brackets. For the block- and fold-wise approach the weight metric used for the aggregation is also indicated in square brackets. For the different approaches from Hagenberg, the used feature-blocks for the prediction inclusive their priority is noted in square brackets.
Among the approaches in figure 36, the block-wise approach has the worst performance - the median F-1 score is 0.79. The second worst approach has the single-block approach on the questionnaire feature-block with a median F-1 score of 0.81. The complete-case approach achieves the third-worst performance - it has a median F-1 score of 0.83. The approaches 'mdd-sPLS [4, 2, 1, 3, 6]' and 'PL - ignore, intercept [1, 2, 3, 4, 5, 6]' are only slightly better than the complete-case approach and have a median slightly above 0.83. The fold-wise approach has a median F-1 score of 0.84 and is only slightly worse than the imputation approach that has a median F-1 score slightly above 0.84. The approaches 'PL - ignore, zero [4, 2, 1, 3, 5]' and 'PL - ignore, zero [4, 2, 1, 3]' both have a median F-1 score of 0.85, but the average F-1 score is considerably higher with the feature-blocks [4, 2, 1, 3]. The overall best performance has the approach 'PL - ignore, zero [1, 2, 3]'. The median F-1 score is close to 0.88 and therefore significantly better than of all other approaches. The results with the balanced accuracy / Matthews correlation coefficient (MCC) as metric are in the attachment in figure A-11 / A-12 - the results are about the same.

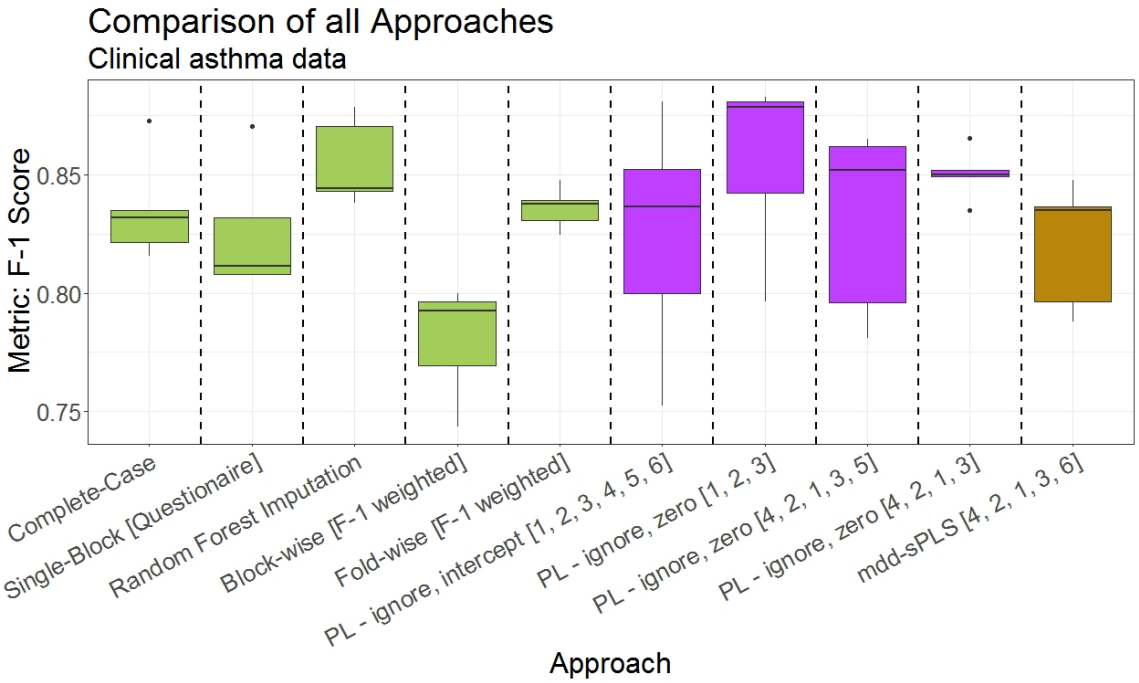

Figure 36: Comparison of the random forest adaptions, the priority-Lasso adaptions and the mdd-sPLS method on the clinical asthma data.

In summary, the best performance was achieved by a priority-Lasso adaption that ignores three out of the six available feature-blocks for the prediction. The second and third best predictive performances have also been achieved with priority-Lasso adaptions. The priority-Lasso adaptions, in general, perform better than the random forest adaptions on the clinical asthma data. But the priority-Lasso adaptions highly depend on the priority of the blocks and the blocks that are actually used for the prediction. Without any prior knowledge about the data, the priority-Lasso adaptions need to be fitted with a naive block priority. The results on such a naive block order can be worse than with a block priority according to a professional opinion. Furthermore, 'selection bias' might have been introduced, as much more variants of the priority-Lasso (different block-priorities and different subset of blocks for the prediction) have been tried out than with the random forest based approaches. The best random forest based approach on the clinical asthma data is the imputation approach, while the block-wise approach has by far the worst predictive performance. The second best random forest based approach is the fold-wise approach.

5 Discussion and Conclusion

This chapter of the thesis discusses the results of the different approaches in the benchmark experiments, draws a conclusion to the approaches and gives an outlook for possible future research. Firstly, the results of the diverse random forest based approaches are compared and summarized. Then the different methods from Hagenberg's thesis [1] are compared and summarized, before the approaches from this and Hagenberg's thesis [1] are compared with each other. Following a conclusion on the different approaches is drawn, and suggestions for future research are made.

The complete-case and single-block approach are the simplest random forest based approaches for data with block-wise missingness. In the diverse patterns of block-wise missingness in the TCGA data, the single-block approach never had the best predictive performance in any test-situation. In contrast, the complete-case approach had the best performance in just two test-situations (out of 66 test-situations in total). On the clinical asthma data, these approaches were only better than the block-wise approach, whereby the block-wise approach had the worst predictive performance on this data. The imputation approach had the best predictive performance among all random forest based approaches on the clinical asthma data and on the TCGA data with block-wise missingness according to pattern 2. On the TCGA data with block-wise missingness according to pattern 1 and 3, the imputation approach had the second-best results. The block-wise approach had its best predictive performance in all test-situations when using the 'F-1 Score' as weight metric for the aggregation of the block-wise predictions. This approach had the worst predictive performance on the clinical asthma data, but the best predictive performance on the TCGA data with block-wise missingness according to pattern 2 and 3. On the TCGA data with block-wise missingness according to pattern 1 and 4, the block-wise approach had a rather bad predictive performance. The fold-wise approach had its best predictive performance in all test-situations when using the 'F-1 Score' as weight metric for the combination of the fold-wise predictions. It was the second-best random forest based approach on the clinical asthma data and the best approach on the TCGA data with block-wise missingness according to pattern 1 and 4. On the TCGA data with block-wise missingness according to pattern 2 and 3, the fold-wise approach had a bad predictive performance and was just better than the complete-case and single-block approach.

In summary to the predictive performance of the random forest based approaches, it can be said that the single-block and complete-case approach had a bad predictive performance. The fold- and block-wise approach

outperformed the single-block and complete-case approach, and had a similar predictive performance. In settings, where the fold-wise approach performed good, the block-wise approach performed bad and vice versa, such that one of these two approaches always resulted in a good predictive performance. The imputation approach also had a good predictive performance comparable with the fold- and block-wise approach. It was also by far the slowest algorithm, as the imputation in high-dimensional data needs a lot of computational power and time. The fold-wise approach was the second slowest approach, as it was implemented in plain R. The single-block, complete-case and block-wise approach were much faster, as these approaches are based on the 'randomForestSRC' package [33] that directly builds upon 'Java' and 'C'.

The thesis of Hagenberg [1] contains diverse adaptions of the priority-Lasso and the mdd-sPLS method from Lorenzo et al. [2]. These approaches were evaluated on the clinical asthma data, whereby the diverse priority-Lasso approaches were evaluated with different block-priorities and different subsets of blocks for the prediction. In all these evaluations, the 'priority-Lasso - impute' adaptions ended up with worse results than the 'priority-Lasso - ignore' adaptions. Hence the 'impute' approach with the priority-Lasso seems less promising than the 'ignore' approach. The two 'ignore' adaptions of the priority-Lasso had a very similar predictive performance. The results of the priority-Lasso approaches with the naive block-priority were the worst. When adjusting the priority of the blocks and/ or only using a subset of the available feature-blocks of the test-set for the prediction, the predictive performance of the priority-Lasso adaptions was increased. The best performance was achieved with the naive block priority, whereby only the first three feature-blocks were used for the prediction on the test-set. The mdd-sPLS approach always had a worse predictive performance than the 'priority-Lasso - ignore' adaptions.

The approaches from Hagenberg's thesis [1] and the random forest based approaches that are in the focus of this thesis were compared based on the clinical asthma data. Among the random forest based approaches, the 'imputation' and 'fold-wise' approach had the best predictive performance on the clinical asthma data - median F-1 score of˜0.84. These approaches had a better predictive performance than the mdd-sPLS method, as well as the best priority-Lasso approach on the naive block-priority [1, 2, 3, 4, 5, 6]. The priority-Lasso adaptions that outperformed the 'imputation' and 'fold-wise' approach were achieved with different block-priorities and using only a subset of available feature-blocks for the prediction. Compared to the priority-Lasso adaptions, the random forest based adaptions were applied to clinical asthma data only once, and no prior knowledge was needed. When fitting

the priority-Lasso adaptions to the clinical asthma data without any prior knowledge, the predictive performance was worse than with the 'imputation' and 'fold-wise' approach. A 'selection bias' might have been introduced, as more variants of the priority-Lasso have been tried out than with the random forest adaptions - using different block-priorities and different subsets of all feature-blocks for the prediction. Hence the choice of the approach highly depends on the prior knowledge of the data. If the user does not know which feature-blocks might be more important than others for the target-variable, the random forest based approaches seem to be the better choice. Especially the fold- and block-wise approach should be used in these situations, as these use the OOB metric to estimate the importance of the diverse folds/ feature-blocks. If a user knows which feature-blocks might be more important than others for the target variable, the 'ignore' priority-Lasso adaptions can be recommended. With these approaches the user can set the priority of the feature-blocks according to their needs - this is not possible with the different random forest based adaptions.
In summary, it can be said that the priority-Lasso adaptions are more appropriate if the user knows something about the relevance of the single feature-blocks. If the relevance of the single feature-blocks for the target variable is unknown, it is recommendable to apply the imputation-, fold- and block-wise approach, as for these approaches no prior knowledge to the data is needed.
For further research, it might be interesting to compare the predictive performances of the random forest based approaches and the approaches from Hagenberg's thesis [1] not only on the clinical asthma data but also on the TCGA data with its different patterns of block-wise missingness. Furthermore, it might be interesting to combine the idea of the 'Block Forests' paper [18] with the fold-wise approach. With the fold-wise approach, a random forest model is fit based on a single fold and the corresponding feature-blocks, whereby the split point selection happens without the incorporation of the block structure. Combining the idea of incorporating the block structure for the split point selection from the 'Block Forests' paper [18] and the idea of the fold-wise approach seems promising. Moreover, an direct implementation of the fold-wise approach in 'C'/ 'Java' would massively reduce the computational time of the approach.

6 Bibliography

[1] Jonas Hagenberg. "Penalized regression approaches for prognostic modelling using multi-omics data with block-wise missing values". manuscript - unpublished yet. - in prep.

[2] Hadrien Lorenzo, Jérôme Saracco, and Rodolphe Thiébaut. "Supervised Learning for Multi-Block Incomplete Data". In: *arXiv preprint arXiv:1901.04380* (2019).

[3] Roman Hornung et al. "Random forests for multiple training data sets with varying covariate sets". manuscript - unpublished yet. - in prep.

[4] Norbert Krautenbacher. "Learning on complex, biased, and big data: disease risk prediction in epidemiological studies and genomic medicine on the example of childhood asthma". PhD thesis. Technische Universität München, 2018.

[5] Francis S Collins. "Medical and societal consequences of the Human Genome Project". In: *New England Journal of Medicine* 341.1 (1999), pp. 28–37.

[6] *National Human Genome Research Institute*. `https://www.genome.gov/about-genomics/fact-sheets/Sequencing-Human-Genome-cost`. Accessed: 2020-01-07.

[7] Belinda JF Rossiter and C Thomas Caskey. "Impact of the Human Genome Project on medical practice". In: *Annals of surgical oncology* 2.1 (1995), pp. 14–25.

[8] Sara Goodwin, John D McPherson, and W Richard McCombie. "Coming of age: ten years of next-generation sequencing technologies". In: *Nature Reviews Genetics* 17.6 (2016), p. 333.

[9] *Veritas - The Genome Company*. `https://www.veritasgenetics.com/myGenome`. Accessed: 2020-01-19.

[10] Ke Bi et al. "Transcriptome-based exon capture enables highly cost-effective comparative genomic data collection at moderate evolutionary scales". In: *BMC genomics* 13.1 (2012), p. 403.

[11] Anne-Laure Boulesteix et al. "IPF-LASSO: Integrative-penalized regression with penalty factors for prediction based on multi-omics data". In: *Computational and mathematical methods in medicine* 2017 (2017).

[12] Valeria D'Argenio. "The high-throughput analyses era: are we ready for the data struggle?" In: *High-throughput* 7.1 (2018), p. 8.

[13] Gregory B Gloor et al. “Microbiome profiling by illumina sequencing of combinatorial sequence-tagged PCR products”. In: *PloS one* 5.10 (2010).

[14] Shrutii Sarda and Sridhar Hannenhalli. “Next-generation sequencing and epigenomics research: a hammer in search of nails”. In: *Genomics & informatics* 12.1 (2014), p. 2.

[15] Forest M White. “The potential cost of high-throughput proteomics”. In: *Sci. Signal.* 4.160 (2011), pp. 8.

[16] *National Institutes of Health*. `https://commonfund.nih.gov/arra/highthroughput`. Accessed: 2020-01-30.

[17] Moritz Herrmann. “Large-scale benchmark study of prediction methods using multi-omics data”. PhD thesis. 2019.

[18] Roman Hornung and Marvin N Wright. “Block Forests: random forests for blocks of clinical and omics covariate data”. In: *BMC bioinformatics* 20.1 (2019), p. 358.

[19] Simon Klau et al. “Priority-Lasso: a simple hierarchical approach to the prediction of clinical outcome using multi-omics data”. In: *BMC bioinformatics* 19.1 (2018), p. 322.

[20] Stefanie Hieke et al. “Integrating multiple molecular sources into a clinical risk prediction signature by extracting complementary information”. In: *BMC bioinformatics* 17.1 (2016), p. 327.

[21] Qing Zhao et al. “Combining multidimensional genomic measurements for predicting cancer prognosis: observations from TCGA”. In: *Briefings in bioinformatics* 16.2 (2015), pp. 291–303.

[22] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[23] Hemant Ishwaran et al. “Random survival forests”. In: *The annals of applied statistics* 2.3 (2008), pp. 841–860.

[24] Leo Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32.

[25] *What is the difference between Bagging and Boosting?* `https://quantdare.com/what-is-the-difference-between-bagging-and-boosting/`. Accessed: 2020-03-06.

[26] Leo Breiman et al. *Classification and regression trees*. CRC press, 1984.

[27] Wenbin Lu. *Lecture 21: Classification and Regression Trees*. Department of Statistics North Carolina State University, 2019.

[28] Bernd Bischl and Christoph Molnar. *Introduction to Machine Learning - Chapter 13: Trees*. Department of Statistics - LMU Munich, WinterTerm 2017/18.

[29] Terry Therneau and Beth Atkinson. *rpart: Recursive Partitioning and Regression Trees*. R package version 4.1-15. 2019. URL: `https://CRAN.R-project.org/package=rpart`.

[30] Leo Breiman. "Bagging predictors". In: *Machine learning* 24.2 (1996), pp. 123–140.

[31] *Bootstrap Sample: Definition, Example*. `https://www.statisticshowto.com/bootstrap-sample/`. Accessed: 2020-03-04.

[32] Bernd Bischl and Christoph Molnar. *Introduction to Machine Learning - Chapter 15: Bagging and Random Forests*. Department of Statistics - LMU Munich, WinterTerm 2017/18.

[33] H. Ishwaran and U.B. Kogalur. *Fast Unified Random Forests for Survival, Regression, and Classification (RF-SRC)*. R package version 2.9.3. manual, 2020. URL: `https://cran.r-project.org/package=randomForestSRC`.

[34] Daniel J Stekhoven and Peter Bühlmann. "MissForest—nonparametric missing value imputation for mixed-type data". In: *Bioinformatics* 28.1 (2012), pp. 112–118.

[35] Olga Troyanskaya et al. "Missing value estimation methods for DNA microarrays". In: *Bioinformatics* 17.6 (2001), pp. 520–525.

[36] Gary King et al. "Analyzing incomplete political science data: An alternative algorithm for multiple imputation". In: *American political science review* 95.1 (2001), pp. 49–69.

[37] James Honaker, Gary King, and Matthew Blackwell. "Amelia II: A Program for Missing Data". In: *Journal of Statistical Software* 45.7 (2011), pp. 1–47. URL: `http://www.jstatsoft.org/v45/i07/`.

[38] Anne-Laure Boulesteix, Sabine Lauer, and Manuel JA Eugster. "A plea for neutral comparison studies in computational sciences". In: *PloS one* 8.4 (2013).

[39] Gareth James et al. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[40] *Performance Metrics for Classification problems in Machine Learning*. https://medium.com/thalus-ai/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b. Accessed: 2020-04-22.

[41] Mohammad Hossin and MN Sulaiman. "A review on evaluation metrics for data classification evaluations". In: *International Journal of Data Mining & Knowledge Management Process* 5.2 (2015), p. 1.

[42] Sabri Boughorbel, Fethi Jarray, and Mohammed El-Anbari. "Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric". In: *PloS one* 12.6 (2017).

[43] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct (2011), pp. 2825–2830.

[44] *Balanced accuracy: what and why?* http://mvpa.blogspot.com/2015/12/balanced-accuracy-what-and-why.html. Accessed: 2020-04-23.

[45] Pierre Baldi et al. "Assessing the accuracy of prediction algorithms for classification: an overview ". In: *Bioinformatics* 16.5 (May 2000), pp. 412–424. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/16.5.412. eprint: https://academic.oup.com/bioinformatics/article-pdf/16/5/412/476945/160412.pdf. URL: https://doi.org/10.1093/bioinformatics/16.5.412.

[46] Bernd Bischl and Christoph Molnar. *Introduction to Machine Learning - Chapter 9: Performance Estimation*. Department of Statistics - LMU Munich, WinterTerm 2017/18.

[47] Nicole Schüller et al. "Improved outcome prediction across data sources through robust parameter tuning". In: (2019).

[48] Marvin N Wright. *SimpleRF*. https://github.com/mnwright/simpleRF. 2018.

[49] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996), pp. 267–288.

7 Attachment

7.1 Availability of data and materials

The whole repository of this master thesis is on 'Github':
`https://github.com/rollator3000/MS_omics`

It contains the entire code and sources of this master thesis. Additionally, a README with documentation to the structure of the repository, quick explanations to the different scripts and a summary of the different approaches is included.
The whole code was written in R - version 3.6.1. The used libraries and their corresponding versions are listed in the README.

The clinical asthma data is not part of the repository because of data protection regulations. For the TCGA data, only the subsetted data sets are part of the repository, as the original TCGA data sets are too big to be uploaded. If interested in the original TCGA data of this thesis contact me via e-mail: 'f.ludwigs@yahoo.de'

7.2 Acknowledgements

I want to thank Dr. rer. nat. Roman Hornung for his excellent supervision of the thesis, the provision of the processed TCGA data and for his constant feedback - especially during COVID-19.

I would also like to thank Matthias Assenmacher, who took care of my access to the LRZ servers and could always help quickly.

I also want to thank J. Hagenberg for the great communication and fruitful cooperation of our master theses.

7.3 Figures

Impurity functions (3), (4) and (5) for a binary target variable

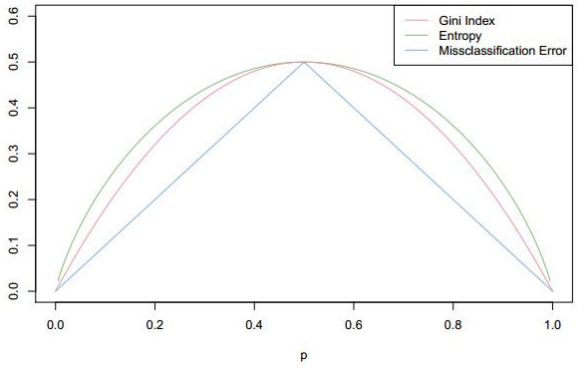

Figure A-1: The different impurity functions (3), (4) and (5) plotted for a given fraction of a binary target variable within any node N [[28], p. 13]

.

Predictive performance of a random forest model on the single feature-blocks of the 14 TCGA data sets

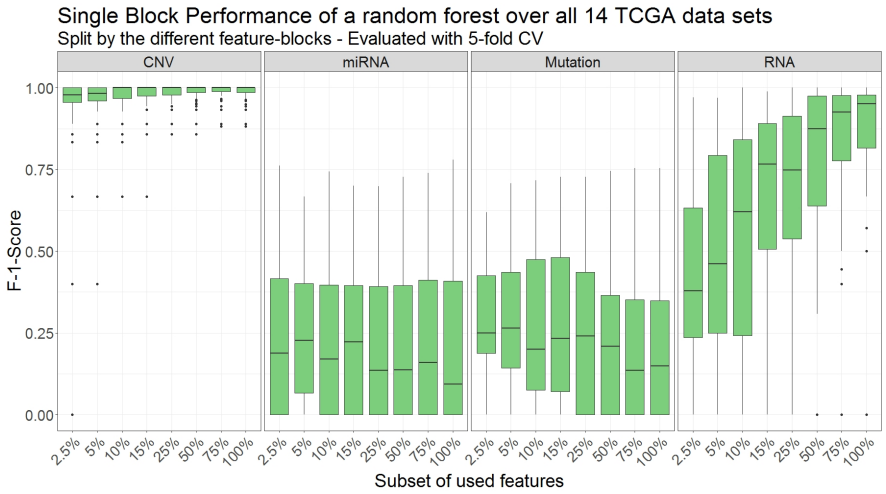

Figure A-2: The F-1 score of a random forest model evaluated on the single omics feature-blocks for a range of possible subsets. The results were obtained on basis of the 14 TCGA data sets.

Comparison of the approaches on the TCGA data with induced block-wise missingness according to pattern 1 - Balanced Accuracy

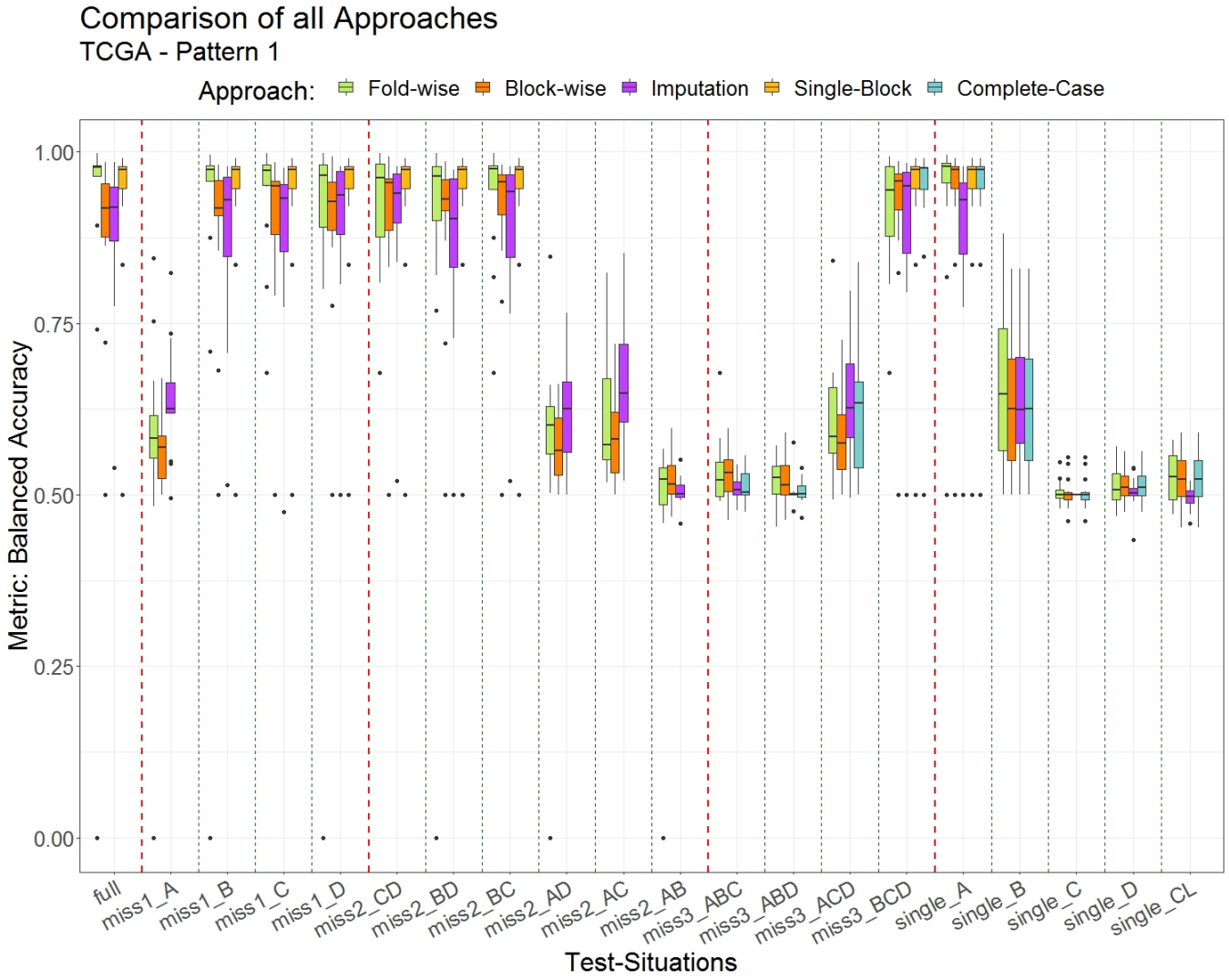

Figure A-3: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 1 - Balanced Accuracy metric

Comparison of the approaches on the TCGA data with induced block-wise missingness according to pattern 1 - MMC

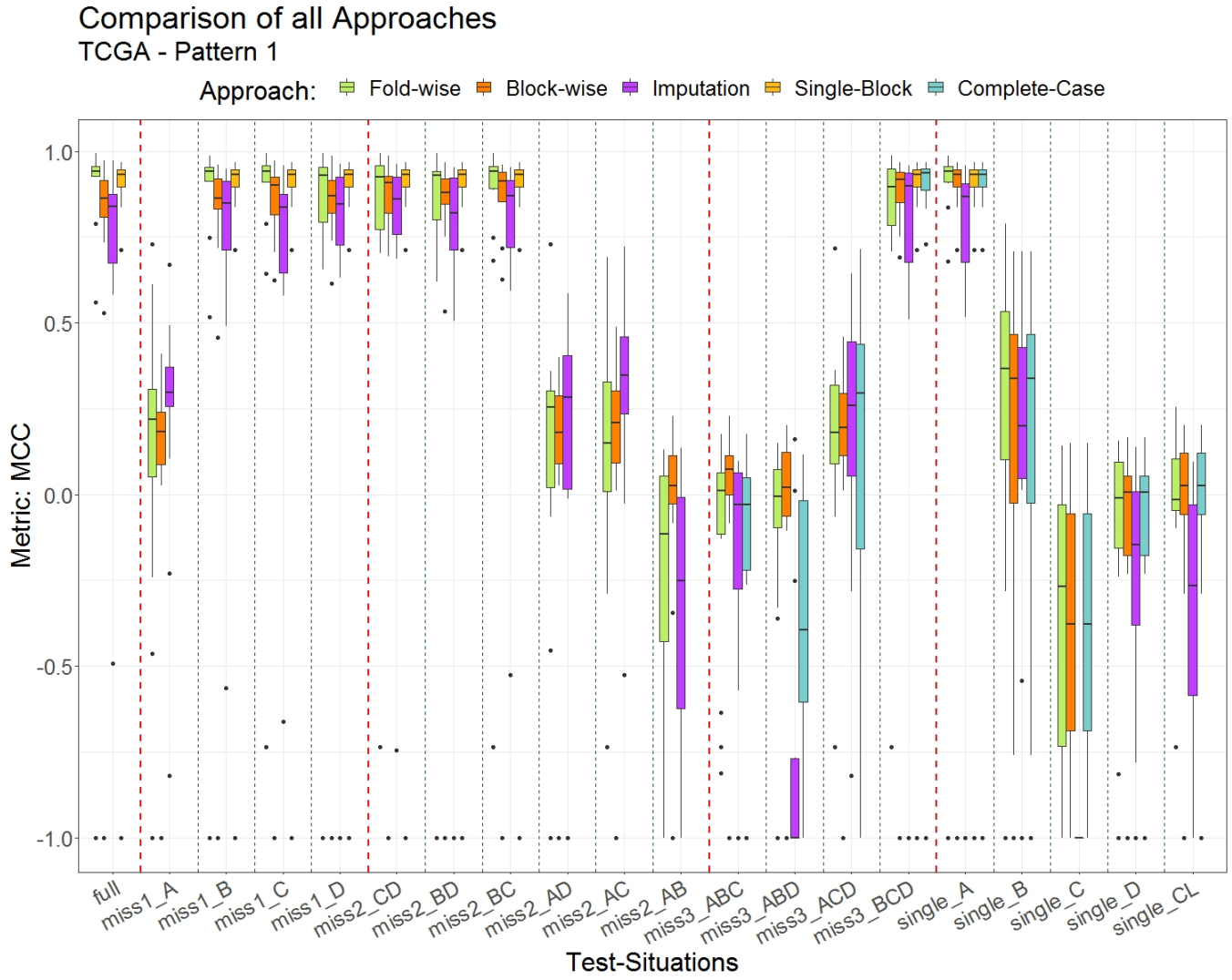

Figure A-4: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 1 - MMC metric

Comparison of the approaches on the TCGA data with induced block-wise missingness according to pattern 2 - Balanced Accuracy

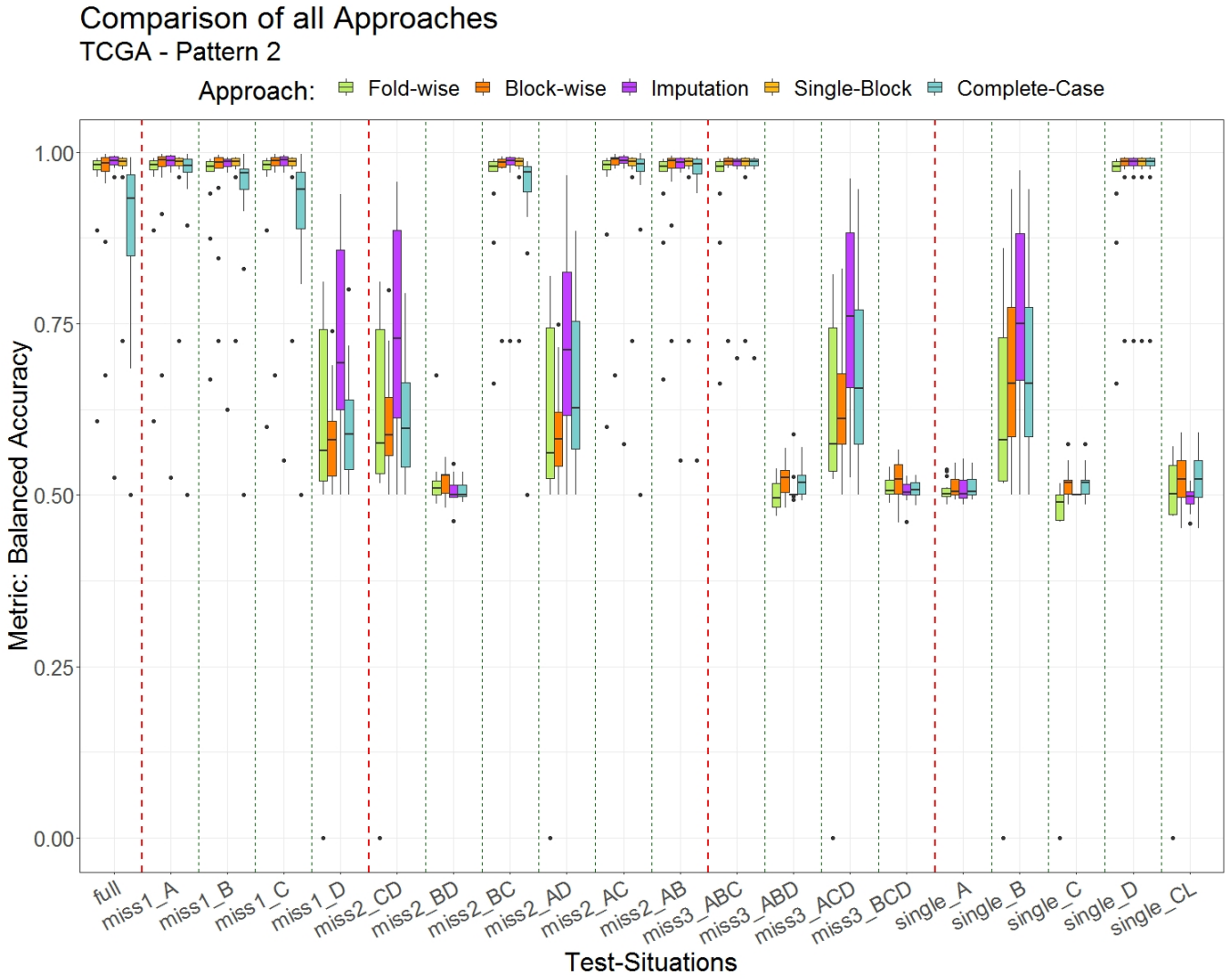

Figure A-5: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 2 - Balanced Accuracy metric

Comparison of the approaches on the TCGA data with induced block-wise missingness according to pattern 2 - MMC

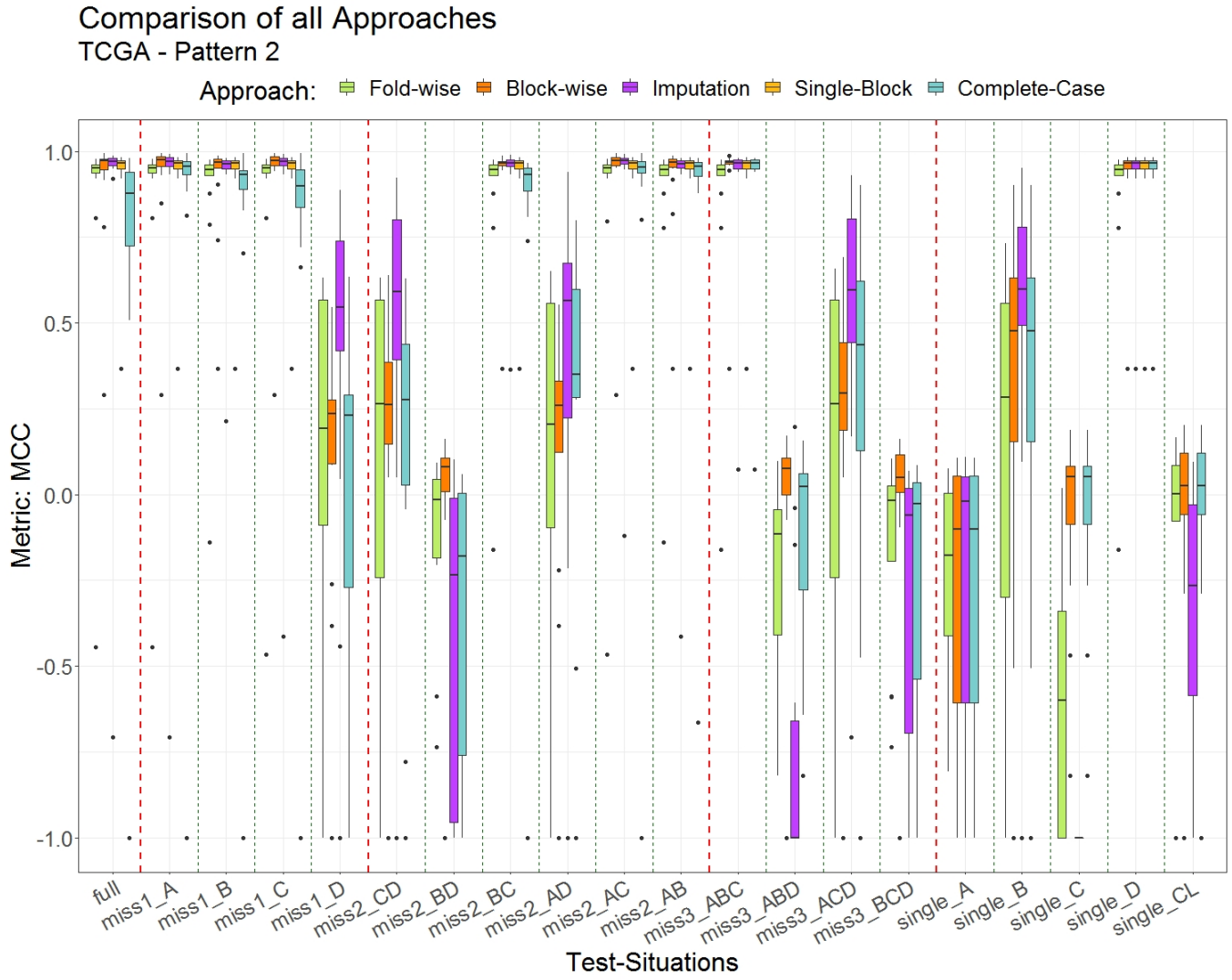

Figure A-6: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 2 - MMC metric

Comparison of the approaches on the TCGA data with induced block-wise missingness according to pattern 3 - Balanced Accuracy

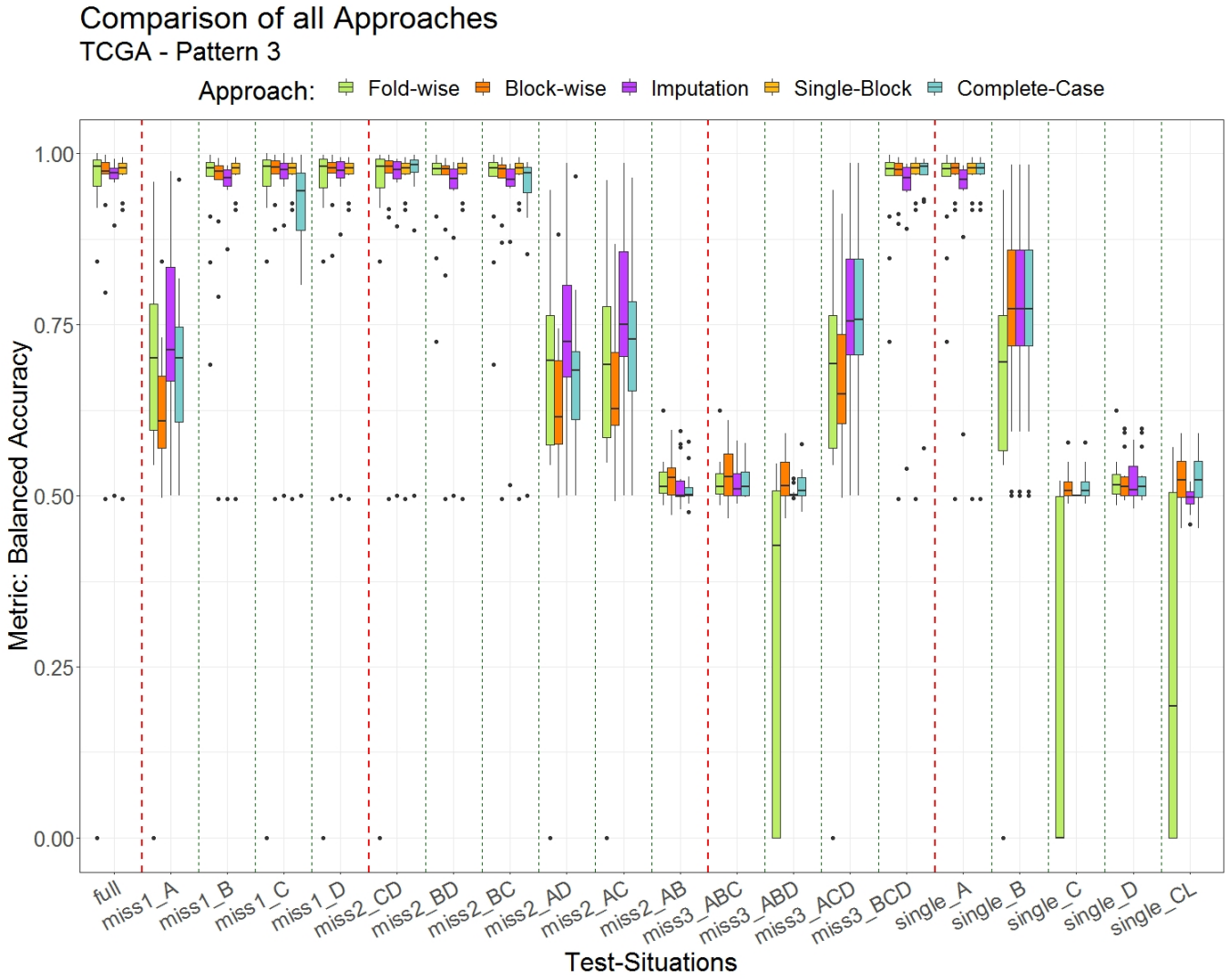

Figure A-7: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 3 - Balanced Accuracy metric

Comparison of the approaches on the TCGA data with induced block-wise missingness according to pattern 3 - MMC

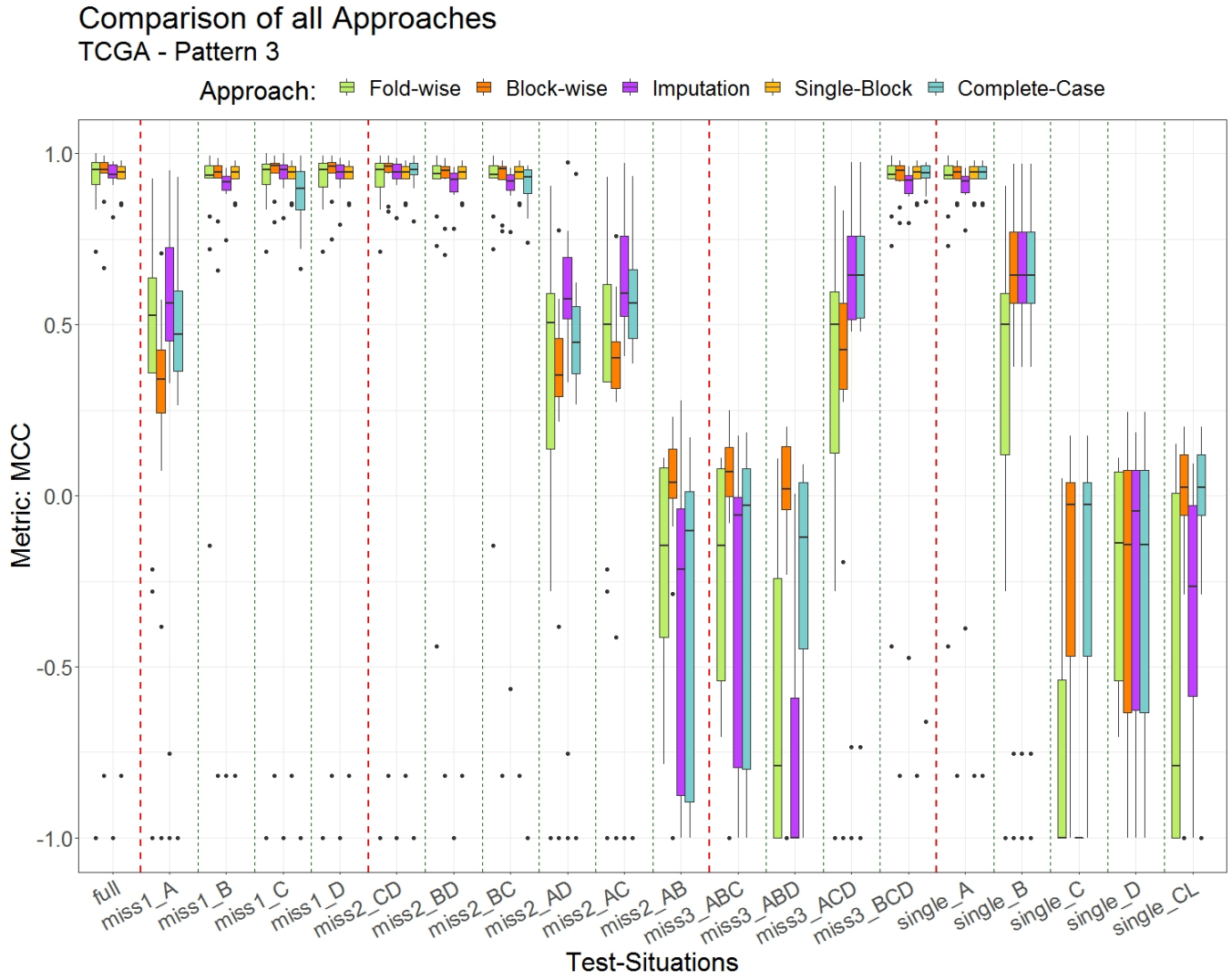

Figure A-8: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 3 - MMC metric

Comparison of the approaches on the TCGA data with induced block-wise missingness according to pattern 4 - Balanced Accuracy

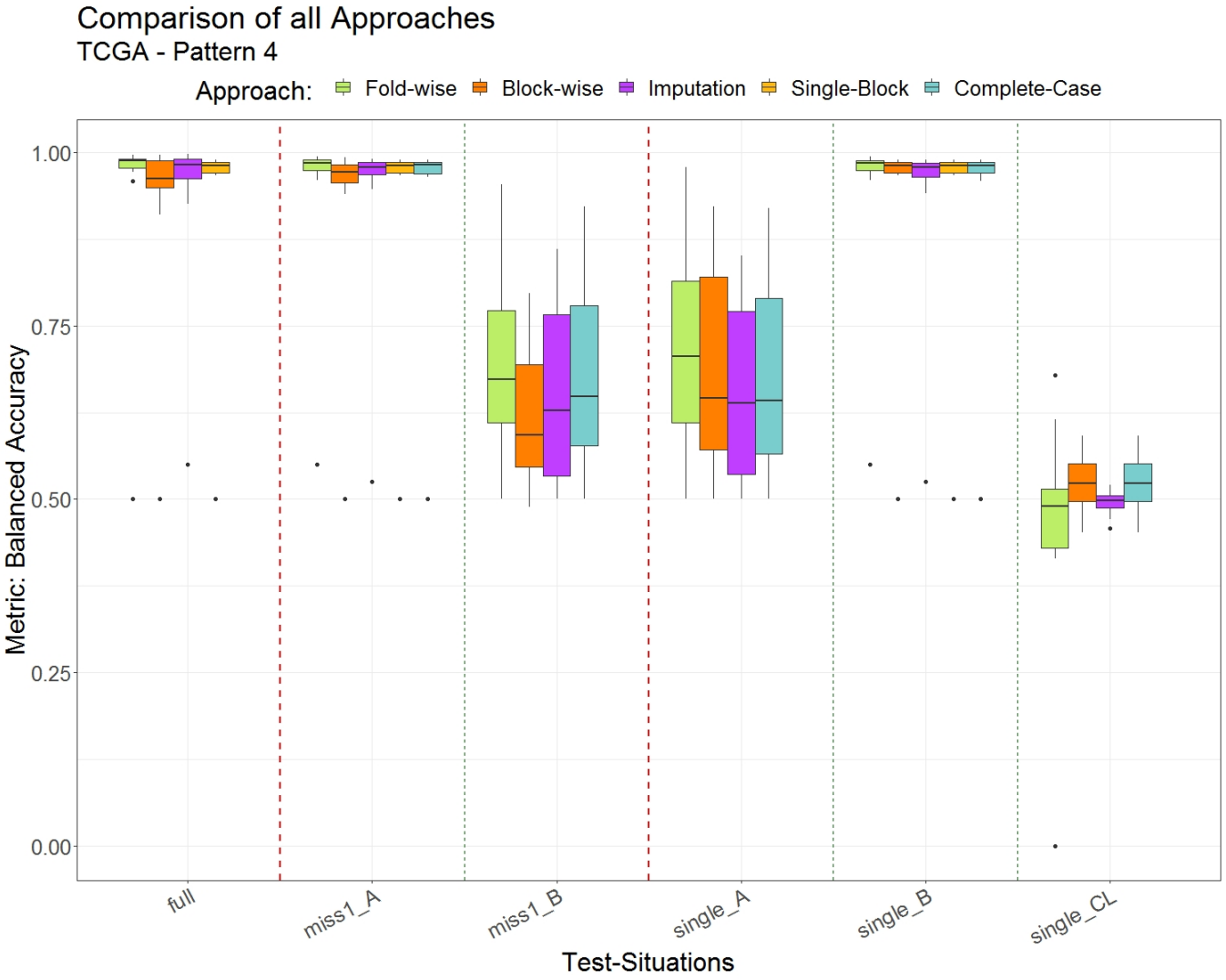

Figure A-9: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 4 - Balanced Accuracy metric

Comparison of the approaches on the TCGA data with induced block-wise missingness according to pattern 4 - MMC

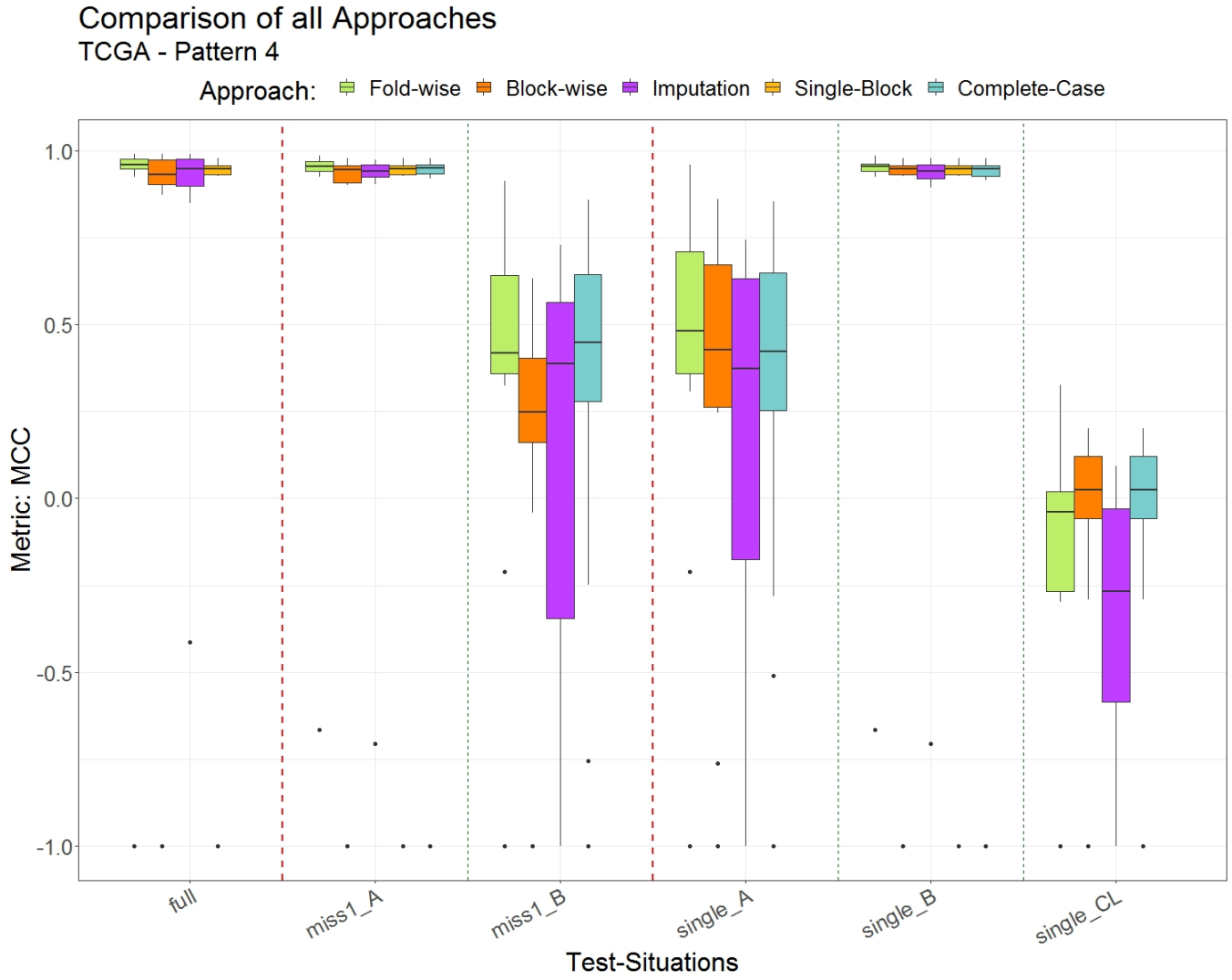

Figure A-10: Comparison of the different approaches on the TCGA data with induced block-wise missingness according to pattern 4 - MMC metric

Comparison of the approaches on the clinical asthma data - Balanced Accuracy

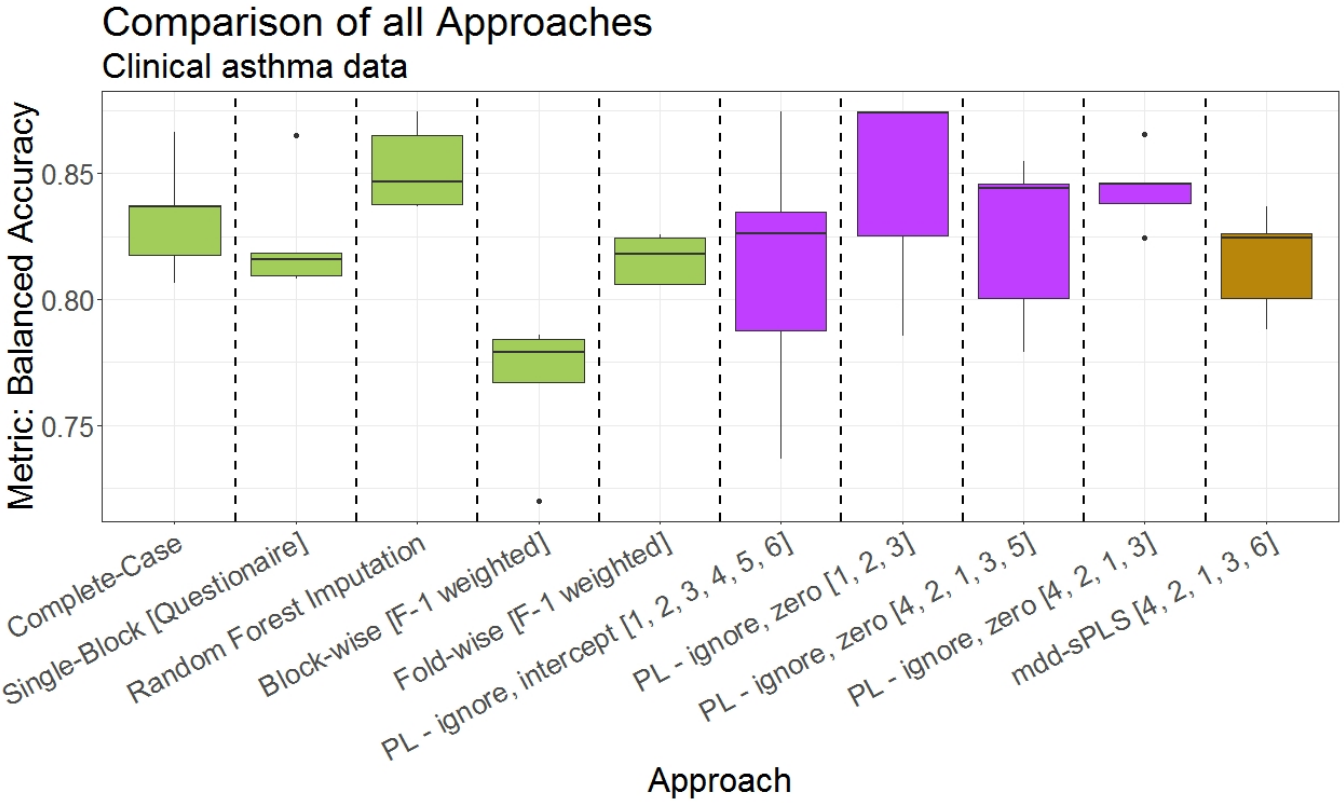

Figure A-11: Comparison of the random forest adaptions, the priority-Lasso adaptions and the mdd-sPLS method on the clinical asthma data - Balanced Accuracy metric

Comparison of the approaches on the clinical asthma data - MMC

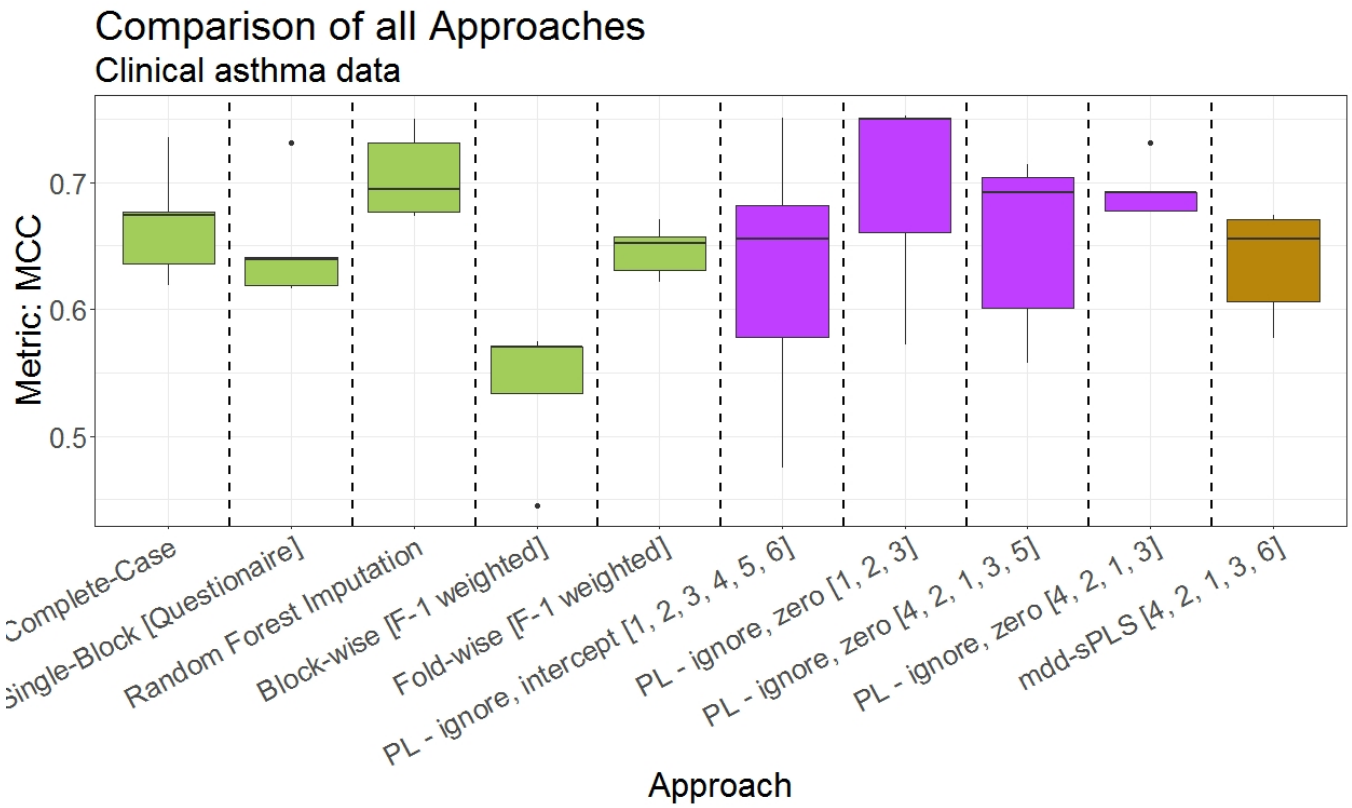

Figure A-12: Comparison of the random forest adaptions, the priority-Lasso adaptions and the mdd-sPLS method on the clinical asthma data - MMC metric

List of Figures

List of Tables

List of Algorithms