technische universität
dortmund

fakultät
statistik

Integration of Feature Selection Stability in Model Fitting

Dissertation

in Fulfillment of
the Requirements for the Degree of
Doktor der Naturwissenschaften

Submitted to
the Faculty of Statistics
of the TU Dortmund University

by
Andrea Martina Bommert

November 20, 2020

Referees:
Prof. Dr. Jörg Rahnenführer
Prof. Dr. Claus Weihs

Date of Oral Thesis Defense:
January 20, 2021

Acknowledgements

I would like to express my gratitude to Prof. Dr. Jörg Rahnenführer for supporting my research and for giving me this opportunity. I would like to thank Prof. Dr. Jörg Rahnenführer, Dr. Michel Lang, and Prof. Dr. Bernd Bischl for the helpful discussions. Also, I would like to thank my family for always supporting me.

This work was supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG), project RA 870/7-1 and Collaborative Research Center SFB 876, project A3. Also, I gratefully acknowledge the computing time provided on the Linux HPC cluster at TU Dortmund University (LiDO3), partially funded in the course of the Large-Scale Equipment Initiative by the German Research Foundation as project 271 512 359.

Contents

Chapter 1

Introduction

Feature selection is one of the most fundamental problems in data analysis, machine learning, and data mining. Recently, it has drawn increasing attention due to high-dimensional data sets emerging from many different fields. Especially for high-dimensional data sets, it is often advantageous with respect to predictive performance, run time, and interpretability to disregard the irrelevant and redundant features. This can be achieved by choosing a suitable subset of features that are relevant for target prediction. In the context of this thesis, relevant features are features that are important for target prediction, that is, features that allow building predictive models with high predictive accuracy. Redundant features are features that carry the same information as other features. So, if irrelevant or redundant features are removed from a predictive model, the prediction accuracy on new data does not decrease (John et al., 1994).

In fields like bioinformatics, feature selection often allows identifying the features that are important for biological processes of interest. Particularly in domains where the selected features will be analyzed in expensive experimental studies, it is desired to keep the number of selected features as small as possible. In such situations, the goal is to select a subset of features so that all relevant information is captured while avoiding the selection of irrelevant or redundant features. In other words, the task of feature selection is to select the smallest subset of features that has maximal predictive quality for an outcome of interest. Also, it is important that the feature selection is stable. This means that the set of selected features is robust with respect to different data sets from the same data generating distribution. An example for an unstable feature selection are sets of selected features differing strongly for data sets obtained with the same experiment conducted at different places such that the assumption of the data sets coming from the same data generating process is valid. An unstable feature selection can make practitioners question the reliability of the resulting models (Kalousis et al., 2007).

This thesis deals with four aspects connected to feature selection: The first contribution of the thesis is a benchmark of filter methods for feature selection. The second aspect is a comparison of measures for the assessment of feature selection stability. In the third part, a multi-criteria approach for obtaining desirable models is proposed. In this context, models with high predictive accuracy, high feature selection stability and a small number of selected features are considered desirable. The fourth part is about fitting models as well, but with focus on data sets with many similar, for example highly correlated, features. In this part, the goal is fitting models that possess high predictive accuracy and that include all relevant information but no irrelevant or redundant features.

1.1 Benchmark of Filter Methods

There exists a variety of methods for feature selection. Feature selection methods can be categorized into three classes (Guyon and Elisseeff, 2003): Filter methods rank features by calculating a score for each feature independent of a model. Either the l features with the highest scores or all the features whose scores exceed a threshold τ are selected (with $l \in \mathbb{N}$ or $\tau \in \mathbb{R}$ being pre-specified). For many filter methods, the score calculation can be done in parallel. Wrapper methods (for example sequential forward search) consider subsets of the set of all features. For each of the subsets, a predictive model is fitted. The subsets are evaluated by a performance measure calculated on the resulting model (for example classification accuracy). Embedded methods like lasso regression include the feature selection in the model fitting process. Regarding the comparison of different methods, benchmark studies have gained increasing attention in the machine learning community.

The first part of this thesis focuses on the comparison of filter methods for feature selection. The focus is based on the following considerations. Most wrapper methods are computationally infeasible for high-dimensional data sets. Embedded methods require that a certain predictive model is used. Most filter methods, however, are fast to calculate and can be combined with any kind of predictive method, even methods with embedded feature selection, see Bommert et al. (2017). Also, they can heavily reduce the run time of machine learning algorithms. So, for data sets with really large numbers of features, it can be necessary to pre-filter the data set in order to make further analyses possible.

In this thesis, 20 filter methods from different toolboxes are benchmarked based on 12 data sets from various domains. The filter methods are representatives of the most prominent general concepts for filter methods. These classes of filter methods are univariate statistical tests, univariate predictive performance indicators, feature variance, random forest importance, and information theoretic measures. Most of the compared filter methods have been integrated into the machine learning R package *mlr* (Bischl et al., 2016) and are ready to use. *mlr* is a comprehensive package for machine learning and a standard in the R community.

The aim of benchmarking the filter methods is finding the best filter methods, so that these methods can be employed in future data analyses. The best filter methods are assessed with respect to predictive performance when combined with a predictive model and with respect to run time. Additionally, the empirical similarity of the filter methods is analyzed. For finding groups of similar filter methods, it is investigated which methods select the features of a data set in a similar order. Also, the scaling behavior of the filter methods is compared. All of these analyses are based on the analyses in Bommert et al. (2020).

1.2 Comparison of Stability Measures

For the assessment of feature selection stability, many stability measures exist and it is unclear which measure is best for evaluating the stability. To find out which measure should be used, 20 stability measures are compared both theoretically and empirically.

For the theoretical comparison, it is investigated which of the desirable theoretical properties defined in Nogueira (2018) and in this thesis are fulfilled by the measures. For the empirical comparison, both a small example, for which all combinations of feature sets can be taken into account, and feature sets selected from real data sets are considered. One part of the empirical comparison is finding out which of the measures assess the stability similarly, that is, consider the same situations as stable or unstable. Another part is investigating the

impact of the number of selected features on the stability measures. These analyses extend the analyses in Bommert et al. (2017) and Bommert and Rahnenführer (2020).

1.3 Finding Desirable Configurations by Multi-Criteria Tuning

Finding a good predictive model for a data set can be challenging. The predictive accuracy of a predictive model is important and is usually the only criterion considered in model selection. When trying to discover relevant features, for example for understanding the underlying biological process, the reliability and the interpretability of the model are important as well. For data from such domains, it is not only necessary to find a model with high predictive accuracy, but it is also relevant that the model includes only few features and that the selection of these features is stable.

To reach all three goals simultaneously, in this thesis, the following strategy is suggested: perform the hyperparameter tuning of the considered methods in a multi-criteria fashion with respect to predictive accuracy, stability of the feature selection, and number of selected features. This strategy is evaluated on 12 data sets from various domains. It is investigated, whether it is possible to find configurations that perform a more stable selection of fewer features without losing much predictive accuracy compared to model fitting only considering the predictive performance. These analyses extend the analyses in Bommert et al. (2017). In this thesis, the term "configuration" is used when talking about a predictive method with specific hyperparameter values while the term "model" refers to a configuration fitted to the data.

1.4 Fitting Models on Data Sets with Similar Features

In many applications, data sets with similar features are generated. An example are gene expression data sets where genes of the same biological processes are often highly positively correlated. For continuous features, the Pearson correlation is often used to quantify the similarity between features. Other criteria, possibly also measuring non-linear associations, can be considered as well. For categorical features, information theoretic quantities like mutual information can be employed.

For data sets with similar features, feature selection is very challenging, because it is more difficult to avoid the selection of relevant but redundant features. Most established methods like univariate filters, lasso, boosting methods, and random forests are not able to select only one feature out of a group of similar features. They select either several or none of the similar features. One method that is able to perform such a selection among similar features is L_0-regularized regression. L_0-regularized regression is a state-of-the-art technique for fitting sparse models. A different approach for the analysis of data sets with similar features would be pre-processing the data by keeping only one feature per group of similar features. The huge disadvantage of this procedure is that it is not guaranteed that the best features for prediction are kept.

For data sets with similar features, also the evaluation of feature selection stability is more difficult. The commonly used stability measures see features with different identifiers as different features. Consider a situation with one set containing a feature X_A and another set not including X_A but instead an almost identical feature X_B. Even though X_A and X_B provide almost the same information, traditional stability measures consider the selection of X_B instead of X_A (or vice versa) as a lack of stability. Existing stability measures that take

into account similarities between features have major theoretical drawbacks. To overcome this deficiency of stability measures for data sets with similar features, we have defined new stability measures. In this thesis, stability measures that consider similarities between features are called "adjusted", the others are called "unadjusted".

The goal in this part of the thesis is to find sparse models that have a high predictive accuracy for data sets with similar features. These models are supposed to include a subset of features such that all relevant information is captured while not including irrelevant or redundant features. For achieving this, we propose the following approach: Use L_0-regularized regression as predictive method and tune its hyperparameter considering both the predictive accuracy and the feature selection stability. Assess the stability of the feature selection with an adjusted stability measure. For choosing the best configuration, that is, the best hyperparameter value, employ ϵ-constraint selection, an algorithm introduced in this thesis that focuses on predictive accuracy and employs the stability as a secondary criterion.

Performing hyperparameter tuning with respect to predictive accuracy is state-of-the-art. Considering also the feature selection stability works as a regularization, avoiding the selection of irrelevant or redundant features. In the proposed approach, L_0-regularized regression could potentially be replaced by any other feature selection method that is able to select only one feature out of a group of similar features, like a sequential forward search taking into account similarities between features. This, however, is out of scope for this thesis.

The proposed approach is evaluated based on both simulated and real data sets of different sizes and with different similarity structures between the features. It is investigated whether the approach is beneficial for fitting models with high predictive accuracy on independent data and that do not include irrelevant or redundant features.

1.5 Structure of the Thesis

The remainder of this thesis is organized as follows: Chapter 2 contains the current state of research in the fields that are considered in this thesis. In Chapter 3, the filter and classification methods used in this thesis are explained (Sections 3.1 and 3.2). Also, feature selection stability is explained in detail, stability measures are defined, and stability selection is described (Section 3.3). Details about multi-criteria optimization are given in Section 3.4. In Chapter 4, the data sets that are used for the following analyses are presented. Chapters 5 to 8 contain the four analysis parts of the thesis. In Chapter 5, the benchmark of filter methods is conducted. The stability measures are compared both theoretically and empirically in Chapter 6. In Chapter 7, the multi-criteria approach for finding desirable models is evaluated. In Chapter 8, the approach for fitting models to data sets with similar features is analyzed. Chapter 9 contains a summary and a discussion of the results as well as an outlook for further aspects of research.

Chapter 2

Current State of Research

The current state of the research in the fields that are considered in this thesis is presented in this chapter. An overview of the literature of feature selection and especially of filter methods for feature selection is given in Section 2.1. Section 2.2 provides references to literature related to feature selection stability. Literature about feature selection for data sets with similar features is discussed in Section 2.3.

2.1 Filter Methods for Feature Selection

In the past decades, many feature selection methods have been proposed. The methods can be categorized into three classes (Guyon and Elisseeff, 2003): Filter methods rank features by calculating a score for each feature independent of a model. Lazar et al. (2012) give an extensive overview of existing filter methods. Wrapper methods (Kohavi and John, 1997) consider subsets of the set of all features. For each of the subsets, a predictive model is fitted and the subsets are evaluated by a performance measure calculated on the resulting model. Wrapper methods include simple approaches like greedy sequential searches (Kittler, 1978), but also more elaborate algorithms like recursive feature elimination (Huang et al., 2018b) as well as evolutionary and swarm intelligence algorithms for feature selection (Yang and Honavar, 1998; Xue et al., 2016; Brezočnik et al., 2018). Embedded methods include the feature selection in the model fitting process. Examples for predictive methods that perform embedded feature selection are lasso regression (Izenman, 2013, pp. 150 ff.), tree based methods like classification and regression trees (Izenman, 2013, pp. 281 ff.) or random forests (Izenman, 2013, pp. 536 ff.), and gradient boosting (Hofner et al., 2014; Bühlmann and Yu, 2003). There are many overview papers that describe in detail, categorize, and suggest how to evaluate existing feature selection methods, for example Guyon and Elisseeff (2003), Liu and Yu (2005), Saeys et al. (2007), Chandrashekar and Sahin (2014), Tang et al. (2014), Hira and Gillies (2015), Jović et al. (2015), Cai et al. (2018), Li et al. (2018), and Venkatesh and Anuradha (2019).

There are also several papers in which feature selection methods are compared. In many of these, the feature selection methods are combined with classification methods in order to assess the predictive performance of the selected features. Liu et al. (2002) compare filter methods based on two gene expression data sets, counting the number of misclassified samples. Bolón-Canedo et al. (2013) analyze the classification accuracy of different filter, wrapper, and embedded methods on several artificial data sets. Bolón-Canedo et al. (2014) and Inza et al. (2004) compare filter methods with respect to classification accuracy based on microarray data sets. Forman (2003) and Aphinyanaphongs et al. (2014) conduct extensive comparisons based on text classification data sets. They analyze filter and wrapper methods, respectively. Darshan and Jaidhar (2018) compare filter methods with respect to classification accuracy on malware detection data. Liu (2004) and Peng et al. (2005) study filter methods on large

data sets, analyzing the predictive accuracy with respect to the number of features to select. Dash and Liu (1997) and Sánchez-Maroño et al. (2007) use small artificial data sets to assess whether the correct features are selected. Dash and Liu (1997) compare different feature selection methods while Sánchez-Maroño et al. (2007) consider filter methods only. Wah et al. (2018) compare filter and wrapper methods on large simulated data sets with respect to the correctness of the selected features. Additionally, they conduct comparisons with respect to classification accuracy on real data sets. Xue et al. (2015) comprehensively compare filter and wrapper methods with respect to classification accuracy and run time, considering each of the two objectives separately. Most of the data sets on which the comparison is based contain a small or medium number of features.

In some papers, only filter methods whose scores are based on similar concepts are compared. Both Meyer et al. (2008) and Brown et al. (2012) compare several filter methods that are based on mutual information. Meyer et al. (2008) analyze the accuracy and the run time of the methods separately. Additionally, they take into account theoretical properties and look at the percentages of correctly identified features on artificial data. Brown et al. (2012) analyze the classification accuracy with respect to the number of selected features and search for Pareto optimal methods considering the accuracy and feature selection stability. Hall (1999) conducts an extensive study of correlation based feature selection. The author analyzes the classification accuracy based on real data sets as well as the choice of relevant or irrelevant features based on artificial data sets.

Several authors introduce new feature selection methods and compare them in a benchmark study to competing approaches. Zhu et al. (2007) and Mohtashami and Eftekhari (2019) present new wrapper methods. The comparisons they conduct for the new methods also include some filter methods. The authors assess the classification accuracy of the methods on several data sets. Zhu et al. (2007) also consider the number of features for which the best performance is achieved. Fleuret (2004), Yu and Liu (2004), and Ke et al. (2018) present new filter methods and conduct a comparison of their new methods with established feature selection methods. All of them consider the classification accuracy of the feature selection methods and the run time separately. Hoque et al. (2018) develop an ensemble filter method that aggregates the scores of several filter methods and compare it to the single filter methods. They compare the methods with respect to classification accuracy considering both small and high-dimensional data sets. Ghosh et al. (2019) create an ensemble filter method for guiding an evolutionary feature selection algorithm. This algorithm is compared to evolutionary algorithms guided by single filter methods with respect to classification accuracy. The comparison is based on high-dimensional data sets and small numbers of features to select.

In Bommert et al. (2020), we conduct an extensive comparison of 22 filter methods on 16 large or high-dimensional data sets with respect to both classification accuracy and run time jointly. Also, we analyze the empirical similarity of the filter methods based on the rankings of all features of all considered data sets. To the best of our knowledge, the approaches of examining the accuracy and the run time jointly as well as studying the similarity of the feature rankings have not been analyzed before by researchers in a filter comparison study.

2.2 Feature Selection Stability

Over the past decade, a variety of frameworks for stability evaluation has been proposed. References for measures for the evaluation of feature selection stability are given in Subsection 3.3.2 where several stability measures are presented. Overviews of existing stability

measures are given in He and Yu (2010) and Lausser et al. (2013). The theoretical properties of different stability measures are studied in Nogueira and Brown (2016) and Nogueira (2018). Pitfalls with respect to interpreting the values of stability measures are discussed in Alelyani et al. (2011). Experimental setups for stability evaluation are presented in Wang et al. (2012). Ensemble methods for making feature selection more stable than a single feature selection method are proposed in Meinshausen and Bühlmann (2010), Boulesteix and Slawski (2009), and Lee et al. (2014). The research that has been done in all of the aforementioned aspects of stability assessment is reviewed in Awada et al. (2012).

Various feature selection methods including ensemble methods are analyzed with respect to stability for example in Davis et al. (2006), Saeys et al. (2008), Abeel et al. (2010), Somol and Novovičová (2010), Dittman et al. (2011), Haury et al. (2011), Dessì et al. (2013), and Lee et al. (2013). Schirra et al. (2016) show that conducting a stable feature selection before fitting a classification model can increase the predictive performance of the model. Most of the works that analyze feature selection methods consider both a high stability and a high predictive accuracy of a resulting classification model as target criteria. But they do not consider the number of selected features as a third target criterion.

In Bommert et al. (2017), we investigate two aspects. First, we conduct an extensive empirical comparison of 9 stability measures based on 3 high-dimensional data sets. We analyze whether the stability measures consider the same situations as stable or unstable. Also, we analyze the connection between the stability values attained for real feature sets and the number of selected features. Moreover, we discuss whether the considered measures fulfill the theoretical properties defined in Nogueira and Brown (2016). Second, we propose and evaluate the approach of performing multi-criteria optimization with respect to predictive accuracy, feature selection stability, and number of selected features for obtaining desirable models. To the best of our knowledge, a similar comparison of stability measures has not been conducted and the approach for obtaining desirable models has not been proposed before.

2.3 Feature Selection on Data Sets with Similar Features

Performing feature selection on data sets with similar features is an issue that has received comparably little attention in the literature. Some benchmarks for feature selection methods include scenarios with similar features, see for example Koller and Sahami (1996), Dash and Liu (1997), and Hall (1999). Also, feature selection methods for selecting relevant and avoiding redundant features have been defined, for example in Yu and Liu (2004) and Brown et al. (2012). These methods are greedy forward search algorithms that measure the redundancy of features by their similarity to the already selected features.

For high-dimensional data sets from the bioinformatics' domain, mostly the Pearson correlation between features is considered to measure their similarity (Toloşi and Lengauer, 2011; Zuber and Strimmer, 2009). Toloşi and Lengauer (2011) report the instability of established feature selection methods such as lasso regression or random forest for data sets with correlated features. They criticize that such methods select an arbitrary feature out of a group of similar features. However, their claim does not coincide with the observations that we have made in preliminary studies. We have observed that with such methods, several of the similar and relevant features are selected, see Section 8.1. Selecting all of the similar and relevant features is what Toloşi and Lengauer (2011) find desirable. This can also be

achieved with techniques such as group lasso (Meier et al., 2008), where groups of similar features are only allowed to enter or leave the model jointly.

In contrast to Tolo̧si and Lengauer (2011) and in accordance with Yu and Liu (2004) and Brown et al. (2012), we find it desirable to have only one feature per group of relevant and similar features included in the model. This way, redundant features are avoided which allows an easier interpretation, because the model is more sparse, and also reduces over-fitting. A method that is able to perform such a selection among similar features is L_0-regularized regression. For recent work on best subset selection and efficient computation of L_0-regularized regression, see for example Bertsimas et al. (2016), Huang et al. (2018a), and Hazimeh and Mazumder (2020).

The lack of stability stated by Tolo̧si and Lengauer (2011) is in our opinion a deficiency of the stability measure that considers almost identical features as different. With adjusted stability measures, that is, stability measures that take into account similarities between features, the feature selection stability can be assessed better for such data sets. Zucknick et al. (2008), Zhang et al. (2009), and Sechidis et al. (2020) define stability measures that consider correlations between features. In Bommert and Rahnenführer (2020), we define and analyze new adjusted stability measures.

Chapter 3

Statistical Methods

The statistical methods used in this thesis are explained in this chapter. In Section 3.1, filter methods for feature selection are described. In Section 3.2, classification methods are explained. Section 3.3 contains various aspects related to feature selection stability. In Section 3.4, details about multi-criteria optimization are given.

3.1 Filter Methods for Feature Selection

For feature selection, filter methods play an important role. They can be combined with any machine learning model and they can heavily reduce run time of machine learning algorithms. All filter methods described in this section are applicable for classification data sets with continuous features. Some of the methods also work for categorical features and require a discretization of continuous features. Two kinds of filter methods are presented: Most filter methods calculate a score for all features and then select the features with the highest scores. Some filter methods, however, select features iteratively in a greedy forward fashion. For these filters, in each iteration, the feature with the maximal score is selected but the scores of different iterations are not comparable. Notation-wise, a data set with n observations of the p features $X_1, \ldots, X_p$ and class variable Y is considered. The filter methods in Subsections 3.1.1 to 3.1.3 are univariate, that is, they do not consider interactions between the p features. Most of the filter methods in Subsections 3.1.4 and 3.1.5 are multivariate.

3.1.1 Univariate Statistical Tests

Filter *anova.test* performs for each feature an analysis of variance (Rasch et al., 2011, pp. 241 ff.) where the class variable is explained by the respective feature. The value of the F statistic is used as filter score. The higher the value of the F statistic, the more different are the mean values of the corresponding feature between the classes.

Both filters *limma* and *sam* perform a moderated version of the F test. The basic idea of both approaches is to stabilize the estimation of the standard deviation by using information about the variation of all features. For *limma*, this is done with a linear regression model (Smyth, 2004). For *sam*, the observed statistics are compared to the expected values of the statistics based on permutations (Tusher et al., 2001). Both methods are popular for gene expression data analysis.

Filter *kruskal.test* applies for each feature a Kruskal-Wallis rank sum test (Kruskal and Wallis, 1952) which is the non-parametric equivalent of an analysis of variance. Analogously, the test statistic is used as the filter score and higher values of this test statistic mean that the values of the corresponding feature differ more between the classes.

Filter *chi.squared* performs for each feature a χ^2 test of independence (Rasch et al., 2011, pp. 221 ff.) between a dichotomized transformation of this feature and the class variable.

The value of the χ^2 statistic is used as the score. The higher the value of the χ^2 statistic, the higher the dependency between the corresponding feature and the class variable. To calculate this test, continuous features have to be discretized. For the implementation in the toolbox *FSelector* (Romanski and Kotthoff, 2018), this is done with the MDL method, see Subsection 3.1.6.

3.1.2 Univariate Predictive Performance

The score of the *auc* filter represents the classification accuracy when each feature is used directly and separately for class prediction for data sets with two classes. For each feature X_k, the following prediction rule for the class variable Y is used: $\widehat{Y} = \mathbb{I}_{[c,\infty)}(X_k)$, with $\mathbb{I}$ denoting the indicator function. The receiver operating curve displays the sensitivity and specificity of a classification rule for all choices of a threshold c, see Sammut and Webb (2011). The area under the receiver operating curve (AUC) of the classification rule $\widehat{Y} = \mathbb{I}_{[c,\infty)}(X_k)$ measures how well each feature separates the target variable. An AUC value of 1 means that there is a threshold c for which the prediction rule is perfectly accurate. An AUC value of 0 means that there is a threshold c for which the rule predicts all labels wrongly which implies that X_k can achieve perfect classification with the rule $\widehat{Y} = \mathbb{I}_{(-\infty,c)}(X_k)$. A value of 0.5 is the worst possible in this application. When the feature and the class variable are independent, the expected AUC value is 0.5. Therefore, $|0.5 - \text{AUC}|$ is used as the AUC filter score. This filter is only applicable for two-class data sets.

The idea of the simple association rule filter *oneR* is to predict the class based on the value of a single feature. For this, continuous features have to be discretized in advance. For the implementation in the toolbox *FSelector* (Romanski and Kotthoff, 2018), this is done using the MDL method, see Subsection 3.1.6. The score of a feature X_k is calculated in the following way: Let $V^{(k)}$ denote the set of possible values for feature X_k. For each value $v \in V^{(k)}$, let $n_{vi}^{(k)}$ denote the number of observations with $X_k = v$ and class i, $i \in \{1, \ldots, l\}$. A simple classification rule for observations with $X_k = v$ is predicting the class i with the highest count $n_{vi}^{(k)}$. The proportion of correctly classified observations by this rule without conducting any resampling,

$$\frac{1}{n} \sum_{v \in V^{(k)}} \max_{i \in \{1,\ldots,l\}} n_{vi}^{(k)},$$

is used as filter score. The higher the accuracy of the rule, the more feature X_k is considered to be suitable for univariate class prediction.

3.1.3 Feature Variance

The *variance* filter uses the variance of a feature as its score. The idea of this filter is to remove features that only consist of noise and therefore have very little variation. This filter only makes sense for data sets where the features are measured on the same scale and have not been scaled to unit variance.

3.1.4 Random Forest Importance

Random forests are bagging ensembles with trees as base learners, see Subsection 3.2.5. There are two popular ways of measuring feature importance based on a random forest: permutation importance and impurity importance. To calculate the permutation importance, the out of bag (oob) observations for each tree, that is, the observations that were not used for fitting this tree, are considered. For the oob observations of each tree, feature X_k is permuted. Then,

the permuted observations are classified by the corresponding trees. The resulting predictive accuracy is compared to the predictive accuracy without permuting feature X_k. The score of a permutation importance filter is the decrease in predictive accuracy of the random forest from original oob observations to permuted observations (Izenman, 2013, pp. 542 ff.). Features that are important for class prediction cause a large decrease in accuracy because their relevant information is not available when the feature is permuted. Filter *permutation* is a permutation importance filter for a forest of classification trees.

Filter *impurity* considers the node impurities of the trees. Nodes containing only observations of one class are called pure, nodes with many observations of different classes are called impure. For each node in each tree of the forest, the impurity is measured before and after the split is made. This can be done for example with the Gini index. The filter score of feature X_k is the mean decrease in impurity due to the splits based on X_k (Izenman, 2013, pp. 542 ff.). A feature that is important for class prediction causes on average a large decrease in impurity.

3.1.5 Mutual Information

Let X and Y be two discrete variables with respective (empirical) probability mass function p. Then the entropy of Y is defined as

$$H(Y) = -\sum_y p(y) \log_2 (p(y))$$

and the conditional entropy of Y given X is given by

$$H(Y|X) = \sum_x p(x) H(Y|X = x) = \sum_x p(x) \left(-\sum_y p(y|x) \log_2 (p(y|x)) \right).$$

The entropy measures the uncertainty of the variable. When all possible values occur with roughly the same probability, the entropy is high. If the probabilities of occurrence are very different from each other, the entropy is low. The mutual information of two variables is defined as

$$I(Y; X) = H(Y) - H(Y|X).$$

It can be interpreted as the decrease in uncertainty about Y conditional on knowing X. Considering the symmetry property $I(Y; X) = I(X; Y)$, it can also be seen as the amount of information shared by X and Y. There are several filter methods based on mutual information, see Hall (1999) and Brown et al. (2012). Continuous features have to be discretized before applying these filters. In the following, filters from two toolboxes are described. They calculate similar scores but differ in the way they discretize the features. The filters *info.gain*, *gain.ratio*, and *sym.uncert* from the toolbox *FSelectorRcpp* (Zawadzki and Kosinski, 2018) use the MDL method for discretization, while the filters *MIM*, *MRMR*, *JMI*, *JMIM*, *DISR*, *NJMIM*, and *CMIM* from the toolbox *praznik* (Kursa, 2018) perform a discretization into equally spaced intervals. Both discretization methods are explained in Subsection 3.1.6.

Filter *info.gain* uses

$$J_{\text{info.gain}}(X_k) = I(Y; X_k),$$

the reduction of uncertainty about the class variable Y due to feature X_k, as the score for X_k.

The score of filter *gain.ratio* is the ratio of the mutual information and the entropy of feature X_k

$$J_{\text{gain.ratio}}(X_k) = \frac{I(Y; X_k)}{H(X_k)}.$$

Out of two features with the same information about Y, this filter favors the feature with the smaller entropy, for example the feature that attains less different values. The reason for dividing by the entropy is to balance out the bias of the mutual information towards selecting features that take many different values like credit card numbers or similar.

For filter *sym.uncert*, the score

$$J_{\text{sym.uncert}}(X_k) = \frac{2 \cdot I(Y; X_k)}{H(X_k) + H(Y)}$$

is used. This score also reduces the bias towards features with many values and additionally, the score is normalized to the range [0,1].

Filter *MIM* ranks all features according to the information they share with the class variable Y

$$J_{\text{MIM}}(X_k) = I(Y; X_k).$$

This is the same score that filter *info.gain* uses as well. However, the filters differ in the way they discretize the features.

The following filter methods calculate the scores of all features iteratively. Thus, the features are selected in a greedy forward manner. Let S denote the set of features that are already selected. S is initialized as $S = \{X_k\}$ with $k = \underset{j \in \{1,\ldots,p\}}{\text{argmax}}\, I(Y; X_j)$. In each iteration, the feature that maximizes the respective score is added to S.

Filter *MRMR* uses the score

$$J_{\text{MRMR}}(X_k) = I(Y; X_k) - \frac{1}{|S|} \sum_{X_j \in S} I(X_k; X_j).$$

The term $I(Y; X_k)$ measures the relevance of the feature based on the information this feature has about Y. The term $\frac{1}{|S|} \sum_{X_j \in S} I(X_k; X_j)$ judges the redundancy of X_k by assessing the mean information that the feature shares with the features in S. The idea is to find maximally relevant and minimally redundant (MRMR) features.

For filter *JMI*, the score

$$J_{\text{JMI}}(X_k) = \sum_{X_j \in S} I(Y; X_k, X_j)$$

is employed. $I(Y; X_k, X_j)$ is the amount of information about Y that X_k and X_j provide jointly. This quantity can be calculated by using the multivariate variable $X = (X_k, X_j)^\top$ and its multivariate probability mass function in the definition of mutual information. The idea of this score is to include features that are complementary to the already selected features.

Filter *JMIM* is a modification of filter *JMI*. The score

$$J_{\text{JMIM}}(X_k) = \min_{X_j \in S} \{I(Y; X_k, X_j)\}$$

considers the minimal joint information over all already selected features instead of the sum.

For filter *DISR*, the score

$$J_{\text{DISR}}(X_k) = \sum_{X_j \in S} \frac{I(Y; X_k, X_j)}{H(Y, X_k, X_j)}$$

is used. Like *JMI*, it uses the information about Y provided jointly by X_k and X_j. But additionally, this information is divided by the joint entropy of Y, X_k, and X_j. To obtain

this entropy, consider the multivariate variable $\widetilde{Y} = (Y, X_k, X_j)^\top$ and plug it into the above definition of the entropy. The reason for dividing by the entropy is avoiding the selection of features that for example attain many different values, see filter *gain.ratio*.

Filter *NJMIM* is a modification of filter *DISR*. Its score

$$J_{\text{NJMIM}}(X_k) = \min_{X_j \in S} \left\{ \frac{I(Y; X_k, X_j)}{H(Y, X_k, X_j)} \right\}$$

considers the minimal relative joint information over all already selected features instead of the sum.

Filter *CMIM* has the score

$$J_{\text{CMIM}}(X_k) = \min_{X_j \in S} \left\{ I(Y; X_k | X_j) \right\}.$$

It uses the conditional mutual information

$$I(Y; X_k | X_j) = H(Y | X_j) - H(Y | X_k, X_j)$$

that can be interpreted as the difference in uncertainty about Y before and after X_k is known, while X_j is known anyway. The idea is to select features that provide much information about the class variable, given the information of the already selected features.

3.1.6 Discretization

Fayyad and Irani (1993) define the minimal description length (MDL) discretization method for continuous features. Their discretization method works recursively: A feature is split into two categories at an optimal cut point. Then, these categories are split recursively until a stopping condition is reached. Let $a_1 < \ldots < a_m$ denote the distinct attained values of the feature to be discretized. Then the points $\frac{1}{2}(a_1 + a_2), \ldots, \frac{1}{2}(a_{m-1} + a_m)$ are considered as cut points. The criterion for determining which of these cut points is optimal is the difference in entropy of the class variable before and after the split. The cut point with the greatest decrease in entropy is considered the best cut point. However, this split is only made if the decrease in entropy exceeds a threshold. This threshold is chosen such that the decrease in entropy is greater than the threshold if and only if the costs of performing the split are lower than the costs of not performing it. The costs are assessed by the minimal description length, which is an information theoretic measure. For details and formulas see Fayyad and Irani (1993). Not carrying out a split because of falling below the threshold works as stopping condition. When all recursions have stopped, the cut points $\tilde{a}_1 < \ldots < \tilde{a}_k$ have been selected. The feature is then discretized into the categories $(-\infty, \tilde{a}_1], (\tilde{a}_1, \tilde{a}_2], \ldots, (\tilde{a}_k, \infty)$. Note that this method discretizes all values of a feature into one single category if the best cut point does not cause enough decrease in entropy.

Kursa (2018) discretizes continuous features by cutting their ranges into q equally spaced intervals and uses these intervals as categories. The number of intervals is determined as $q = \max\left\{\min\left\{\frac{n}{3}, 10\right\}, 2\right\}$ where n is the number of observations in the data set.

3.1.7 Overview of all Considered Filter Methods

Table 3.1 gives an overview of all filter methods considered in this thesis. If the class variable can be of type multi-class, also binary class variables are allowed. If categorical features are required, continuous features are automatically discretized by the filter methods.

Filter	Concept	Class Variable	Features
anova.test	univariate statistical test	multi-class	continuous or binary
limma	univariate statistical test	multi-class	continuous or binary
sam	univariate statistical test	multi-class	continuous or binary
kruskal.test	univariate statistical test	multi-class	continuous or binary
chi.squared	univariate statistical test	multi-class	categorical
auc	univariate predictive performance	two-class	continuous
oneR	univariate predictive performance	multi-class	categorical
variance	feature variance	arbitrary	continuous
permutation	random forest importance	multi-class	continuous or categorical
impurity	random forest importance	multi-class	continuous or categorical
info.gain	mutual information	multi-class	categorical
gain.ratio	mutual information	multi-class	categorical
sym.uncert	mutual information	multi-class	categorical
MIM	mutual information	multi-class	categorical
MRMR	mutual information	multi-class	categorical
JMI	mutual information	multi-class	categorical
JMIM	mutual information	multi-class	categorical
DISR	mutual information	multi-class	categorical
NJMIM	mutual information	multi-class	categorical
CMIM	mutual information	multi-class	categorical

Table 3.1: Requirements on the class variable and on the features for all filter methods.

3.2 Classification Methods

Consider a data set with numeric features $X_1, \ldots, X_p$ and class variable Y. In this thesis, only data sets with two classes are considered. Classification methods aim to predict the class of a new observation, given the values of the features $X_1, \ldots, X_p$. For this, they learn a classification rule based on training data for which both the values of the features and the class variable are available. In the following, different classification methods are presented.

3.2.1 Support Vector Machine

Support vector machines (Izenman, 2013, pp. 369 ff.) use the hyperplane in feature space which is optimal with respect to the maximum margin principle as decision boundary for class prediction. An illustration of this principle is given in Figure 3.1. If the classes are linearly separable, the maximum margin principle means finding the hyperplane for which all observations of the same class lie on the same side of the hyperplane and for which the distance between the hyperplane and the closest observations per class is maximal. For an illustration of the linear separable case, see Figure 3.1 and ignore the grey points. If the classes are not linearly separable, slack variables are introduced to quantify how far the observations lie on the wrong side of the separating hyperplane or inside the margin. This is depicted by the grey points in Figure 3.1. Support vector machines have one hyperparameter $C > 0$ that balances the maximization of the margin and the sum of the values of the slack variables, that is, the importance of all observations lying on the correct side of the

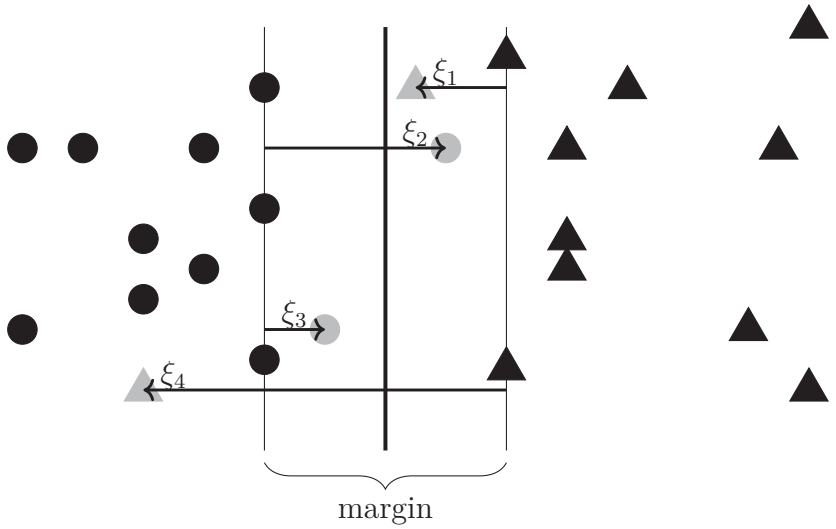


Figure 3.1: Illustration of linear support vector machines. Circles denote observations of the one class, triangles observations of the other class. Linear separable case: without grey points, linear non-separable case: with grey observations. $\xi_1, \ldots, \xi_4$ are slack variables.

hyperplane and outside the margin. The larger the value of C, the less importance is given to the maximization of the margin.

To change the shape of the separating hyperplane into something non-linear, kernel functions are used. A popular type of kernel functions are radial basis functions (RBF). Radial basis functions have an additional hyperparameter $\sigma > 0$ that determines the kernel width and therefore the shape of the transformed hyperplane. Support vector machines use all features, so no embedded feature selection is conducted.

3.2.2 *k* Nearest Neighbors

The method k nearest neighbors classifies a new observations by a majority vote of the k closest observataions in the training data set (Larose and Larose, 2014, pp. 149 ff.). For measuring the distance between observations, the Euclidean norm based on all features is used. $k \in \mathbb{N}$ is a hyperparameter. For small values of k, only few observations that lie close to the new observation are considered for the majority vote. For large values of k, many observations are included in the vote, some of them possibly lying far away from the new observation.

3.2.3 Regularized Logistic Regression

The ordinary logistic regression model is given by

$$P(Y = 1|x_1, \ldots, x_p) = \frac{\exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p)}{1 + \exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p)}$$

with $Y \in \{0, 1\}$ denoting the class variable and $x_1, \ldots, x_p$ are the values of the features $X_1, \ldots, X_p$ for a given observation. The regression coefficients $\beta_0, \ldots, \beta_p$ are estimated by maximizing the log likelihood (LL) of the model (Izenman, 2013, pp. 250 ff.). If the number of features exceeds the number of observations in the training data set, the estimation of

the regression coefficients is not possible. To overcome this problem, regularization can be applied. Regularization means that for the estimation of $\beta := (\beta_0, \ldots, \beta_p)^\top$ the function

$$r_q(\beta) = LL - \lambda \left|\left|\beta\right|\right|_q$$

instead of just the log likelihood is maximized (Izenman, 2013, pp. 150 ff.). $\left|\left|\beta\right|\right|_q$ denotes the L_q-norm of β with $q \in \mathbb{N}_0$. $\lambda \geq 0$ is a hyperparameter that balances the goodness of the fit and the size of the regression coefficients. The larger the value of λ, the more importance is given to a small size of the regression coefficients measured by the L_q-norm.

For $q = 2$, the regularization corresponds to the ridge penalty. The ridge penalty is known to shrink all coefficients towards zero but it does not perform feature selection. For $q = 1$, the regularization corresponds to the lasso penalty. With the lasso penalty, typically only a subset of the coefficients obtains non-zero estimates. By this, embedded feature selection is conducted. Depending on the implementation, the L_1-penalty fails at picking only one feature out of a group of identical features. The reason is that the L_1-penalty attains the same value if only one of these features has a non-zero coefficient or if this coefficient value is divided among several of the features.

For $q = 0$, the penalty term counts the number of non-zero coefficients. This leads to sparse models, also in the case of identical features. Finding the optimum of r_0 is NP-hard but can be achieved with mixed integer optimization for data sets with less than 1 000 features (Hazimeh and Mazumder, 2020). To solve the optimization problem also for larger data sets within acceptable run time, Hazimeh and Mazumder (2020) present an approximate algorithm based on coordinate descent and local combinatorial optimization. For numerical reasons, in this thesis, $\widetilde{r}_0(\beta) = LL - \lambda \left|\left|\beta\right|\right|_0 - 0.01 \left|\left|\beta\right|\right|_2$ is optimized instead of r_0.

3.2.4 Gradient Boosting

Gradient boosting (Hofner et al., 2014; Bühlmann and Yu, 2003) combines many weak classification methods, so called base learners, into one strong classification method. The resulting classification rule minimizes the mean misclassification costs on the training data measured by an arbitrary loss function. For generalized linear models (glm) boosting, the loss function is chosen as the negative log likelihood that is used for the estimation of the regression coefficients of the generalized linear model. So, the minimization problem can be written as

$$R(f) = \frac{1}{n} \sum_{i=1}^{n} \rho(y_i, f(x_{i1}, \ldots, x_{ip})) \rightarrow \min!$$

where f denotes the optimal classification rule to be found, ρ is the loss function, and $y_i, x_{i1}, \ldots, x_{ip}$ are the observed values for the i-th observation, $i = 1, \ldots, n$. To solve the minimization problem, gradient descent is applied. Gradient descent is an iterative optimization method that starts with a solution $f^{[0]}$ and updates this solution in each iteration by taking a step of length ν in the direction of steepest descent:

$$f^{[j]} = f^{[j-1]} - \nu \widehat{\frac{\partial R(f)}{\partial f}}\bigg|_{f=f^{[j-1]}}.$$

Let $h_1, \ldots, h_b$ denote the b base learners. Typically, $b = p$ and each base learner uses only one of the features $X_1, \ldots, X_p$. Each base learner has a parameter vector θ. For example, $h_1, \ldots, h_b$ can be univariate linear models and $\theta_1, \ldots, \theta_b$ the respective regression coefficients. For initialization, $f^{[0]}$ is chosen as the constant function that minimizes R among all constant

functions. It serves as an offset and the base learners consequently do not need an intercept term. In the j-th iteration, the negative gradient

$$-\left.\frac{\partial R(f)}{\partial f}\right|_{f=f^{[j-1]}} = \frac{1}{n}\sum_{i=1}^{n} \underbrace{-\left.\frac{\partial \rho\left(y_i, f\right)}{\partial f}\right|_{f=f^{[j-1]}(x_{i1},\ldots,x_{ip})}}_{:=u_i}$$

needs to be approximated by a base learner. To do so, all base learners are fitted to predict the u_i based on the values $x_{i1}, \ldots, x_{ip}$, $i = 1, \ldots, n$. That is, for each base learner h_k, $\hat{\theta}_k$ is chosen such that

$$U(h_k) = \frac{1}{n}\sum_{i=1}^{n} \rho\left(u_i, h_k\left(\hat{\theta}_k, x_{i1}, \ldots, x_{ip}\right)\right)$$

is minimal. Among the fitted base learners, the one that approximates the negative gradient best, that is, the one with smallest value $U(h_k)$ is chosen and denoted as $h_*^{[j]}$. Then the gradient descent update is calculated as

$$f^{[j]} = f^{[j-1]} + \nu h_*^{[j]}.$$

After *n.iter* iterations, gradient descent terminates. The final classification rule is

$$\hat{f} = f^{[0]} + \sum_{j=1}^{n.iter} \nu h_*^{[j]}.$$

For class prediction of a new observation, the *n.iter* fitted base learners $h_*^{[1]}, \ldots, h_*^{[n.iter]}$ classify the new observation. Then, the vote

$$\hat{f}(x_1, \ldots, x_p) = f^{[0]}(x_1, \ldots, x_p) + \sum_{j=1}^{n.iter} \nu h_*^{[j]}(x_1, \ldots, x_p)$$

is calculated and the class is predicted as $\hat{y} = \text{sign}(\hat{f}(x_1, \ldots, x_p))$, assuming that the classes are labeled "1" and "-1". Note that the predicted values of the base learners are not required to equal 1 or -1.

The number of boosting iterations *n.iter* is a hyperparameter. It is an upper limit for the number of base learners and thereby for the number of features that can be part of the classification rule. So, embedded feature selection is performed by the selection of the corresponding base learners in the boosting update iterations.

3.2.5 Random Forest

Random forests (Izenman, 2013, pp. 536 ff.) are ensembles of classification trees (Izenman, 2013, pp. 281 ff.). For class prediction, a new observation is classified by all trees and then a majority vote is conducted. To construct a random forest, *ntree* classification trees are built. For each tree, a bootstrap sample of the training data set is drawn and the tree is built using only these observations. For finding the best split in each node, *mtry* randomly selected features are considered. Splitting is conducted until further splits would lead to terminal nodes with less than *nodesize* observations. The trees are not pruned. Due to the randomness in its construction process, the resulting random forest is not deterministic. That is, two random forests constructed with the same values of the hyperparameters for the same data set will generally be different, unless the same seed for the random number generator is used.

The more trees there are in the forest, the better the model is fitted to the training data set. The larger the value of *mtry*, the less diverse are the trees. The larger the value of *nodesize*, the lower is the depth of the trees. For more details about the influence of the hyperparameters see Probst et al., 2019. The features that are included in a random forest model can be assessed by checking which features are assigned variable importance values (see Subsection 3.1.4) greater than 0.

3.2.6 Classification Accuracy

To evaluate the predictive performance of a classification method on a binary classification data set, the accuracy is defined as

$$\text{accuracy} = \frac{\text{number of correctly classified observations}}{\text{number of all classified observations}}.$$

The accuracy takes values in the interval $[0, 1]$. The higher the accuracy, the better the predictive performance of the classification method.

3.3 Feature Selection Stability

This section is about feature selection stability. In Subsection 3.3.1, feature selection stability is explained. Many measures for the assessment of feature selection stability are defined in Subsection 3.3.2. In Subsection 3.3.3, stability selection is described.

3.3.1 Definition of Feature Selection Stability

Kalousis et al. (2007) define the stability of a feature selection algorithm as the robustness of the set of selected features towards different data sets from the same data generating distribution. They explain that feature selection stability indicates how strongly different training data sets affect the sets of selected features. For similar data sets, a stable feature selection algorithm selects similar sets of features. An example for similar data sets could be data coming from different studies measuring the same features, possibly conducted at different places and times, as long as the assumption of the same underlying distribution is valid. Another example is that one data set is split into parts for resampling.

Especially in domains where the selected features are subject to further research, the stability of feature selection is very important. If for similar training data sets very different sets of features are selected, this questions not only the reliability of the resulting models but could also lead to unnecessary expensive experimental research.

A lack of stability has three main reasons: too few observations in a data set, highly similar features and equivalent sets of features. Consider a group of data sets, for which the number of observations does not greatly exceed the number of features, from the same data generating process. The subsets of features with maximal predictive quality on the respective data sets often differ between these data sets. One reason is that there are features that seem beneficial for prediction, but that only help on the specific data set and not on new data from the same process. Selecting such features and including them in a predictive model typically causes over-fitting. Another reason is that there are features with small differences in predictive quality even though they are unrelated with respect to their content. Due to the small number of observations, chance has a large influence on which of these features has highest predictive quality. The instability of feature selection resulting from both reasons is

undesirable. Regarding the case of highly similar and therefore almost identical features, it is likely that for some data sets from the underlying data generating process, one feature is selected and for other data sets from the same process, another one of the similar features is selected. As the features are almost identical, it makes sense to label this as stable because the feature selection algorithm always chooses a feature with the same information - just with a different identifier. Therefore, it is desirable to have a stability measure that takes into account the reason for the differences in the sets of selected features. However, most existing stability measures treat both situations equally: if the identifiers of the selected features are different, the feature selection is rated unstable. Regarding the case of equivalent feature sets, for some data sets, there are different sets of features that contain exactly the same information. Finding all equivalent optimal subsets of features is an active field of research, see for example Statnikov et al. (2013), and worst-case intractable. The selection of equivalent subsets of features is evaluated as unstable by all existing stability measures. Creating stability measures that can recognize equivalent sets of features is out of scope for this thesis.

3.3.2 Stability Measures

In this subsection, 20 stability measures are defined. For this, the following notation is used: Assume that there is a data generating process that generates observations of the p features $X_1, \ldots, X_p$. Further, assume that there are m data sets that are generated by this process. A feature selection method is applied to all data sets. Let $V_i \subseteq \{X_1, \ldots, X_p\}$, $i = 1, \ldots, m$, denote the set of selected features for the i-th data set and $|V_i|$ the cardinality of this set. The feature selection stability is assessed based on the similarity of the sets $V_1, \ldots, V_m$.

Stability measures that only allow for $m = 2$ sets or require $|V_1| = \ldots = |V_m|$ are not considered in this thesis because they are not suitable for the applications in Chapters 7 and 8. Remarks on the requirements with respect to the sets $V_1, \ldots, V_m$ in order for each stability measure to be well-defined are given in Appendix B.1. Most stability measures are undefined in some extreme situations with all or no features being selected which invokes a division by zero. For all stability measures, large values correspond to high stability and small values correspond to low stability. Properties of the measures are discussed in Section 6.1.

3.3.2.1 Intersection Based Stability Measures

The following stability measures assess the similarity of the sets $V_1, \ldots, V_m$ based on the cardinalities of all pairwise intersections $|V_i \cap V_j|$. Two sets V_i and V_j are considered to be similar if their overlap is large, that is, if $|V_i \cap V_j|$ takes a large value. The measures standardize the cardinalities of the intersections in different ways. Then, they calculate the average of all standardized scores. Three simple intersection based stability measures are

$$\text{SMJ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j|}{|V_i \cup V_j|} \quad \text{(Jaccard, 1901)},$$

$$\text{SMD} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j|}{\frac{1}{2}\left(|V_i| + |V_j|\right)} \quad \text{(Dice, 1945), and}$$

$$\text{SMO} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j|}{\sqrt{|V_i| \cdot |V_j|}} \quad \text{(Ochiai, 1957)}.$$

The scores differ in the choice of denominator. All of the denominators are upper bounds for $|V_i \cap V_j|$, making 1 the maximum value of each score.

Dunne et al. (2002) present a stability measure that is based on the Hamming distance between V_i and V_j and therefore considers both the number of features that are included in both V_i and V_j and the number of features that is included in neither of them:

$$\text{SMH} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| + \left|V_i^c \cap V_j^c\right|}{p}.$$

For all of the aforementioned measures, the expected value for a random feature selection depends on the number of selected features, see Section 6.1. The more features are selected, the larger is the expected stability value. To avoid this, the following stability measures subtract $\frac{|V_i| \cdot |V_j|}{p}$ in the numerator. This is the expected value of $|V_i \cap V_j|$ for two random sets of cardinalities $|V_i|$ and $|V_j|$ and equal selection probabilities for all features. By doing this, the expected stability values for a random feature selection becomes 0, independent of the number of selected features. The measures differ in the way they normalize the pairwise scores in the denominator.

$$\text{SML} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\min\{|V_i|, |V_j|\} - \max\{0, |V_i| + |V_j| - p\}} \text{ (Lustgarten et al., 2009)}$$

$$\text{SMW} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\min\{|V_i|, |V_j|\} - \frac{|V_i| \cdot |V_j|}{p}} \quad \text{(Wald et al., 2013)}$$

$$\text{SMU} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i| \cdot |V_j|}{p}} \quad \text{(Bommert and Rahnenführer, 2020)}$$

$$\text{SMK} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\frac{|V_i| + |V_j|}{2} - \frac{|V_i| \cdot |V_j|}{p}} \quad \text{(Carletta, 1996)}$$

$$\text{SMP} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot \left(1 - \frac{|V_i|}{p}\right) \cdot |V_j| \cdot \left(1 - \frac{|V_j|}{p}\right)}} \quad \text{(Nogueira and Brown, 2016)}$$

For SMK and SMP, the pairwise scores are coefficients for the agreement in contingency tables. They correspond to the κ-coefficient (SMK) and to the ϕ-coefficient (SMP).

3.3.2.2 Frequency Based Stability Measures

Frequency based stability measures assess the stability based on the selection frequencies of all features $X_1, \ldots, X_p$. They evaluate such situations as stable in which all features are either selected for all of the sets or for none of them. A feature that is only selected for some of the m sets decreases the stability score. Let h_j, $j = 1, \ldots, p$, denote the number of sets V_i that contain feature X_j so that h_j is the absolute frequency with which feature X_j is selected. Also, let $q = \sum_{j=1}^{p} h_j = \sum_{i=1}^{m} |V_i|$ denote the total number of selected features.

Novovičová et al. (2009) present the entropy-based stability measure

$$\text{SME} = \frac{1}{q \log_2(m)} \sum_{j: X_j \in \bigcup_{i=1}^{m} V_i} h_j \log_2(h_j).$$

Davis et al. (2006) define the stability measure

$$\text{SMD-}\alpha = \max\left\{0,\ \frac{1}{\left|\bigcup_{i=1}^{m} V_i\right|}\sum_{j=1}^{p}\frac{h_j}{m} - \frac{\alpha}{p}\cdot \text{median}\left(|V_1|,\ldots,|V_m|\right)\right\}.$$

The minuend rewards high selection frequencies while the subtrahend penalizes large sets of selected features. The larger the value of $\alpha \geq 0$, the more important it is to the stability measure that only few features are selected.

The relative weighted consistency is defined by Somol and Novovičová (2008) as

$$\begin{aligned}
\text{SMS} &= \frac{\left(\sum_{j=1}^{p}\frac{h_j}{q}\frac{h_j-1}{m-1}\right) - c_{\min}}{c_{\max}-c_{\min}} \quad \text{with}\\
c_{\min} &= \frac{q^2 - p(q - q \bmod p) - (q \bmod p)^2}{pq(m-1)},\\
c_{\max} &= \frac{(q \bmod m)^2 + q(m-1) - (q \bmod m)\,m}{q(m-1)}.
\end{aligned}$$

Calculating $\frac{h_j-1}{m-1}$ scales the positive absolute frequencies to [0,1]. All scaled frequencies with $h_j > 0$ are assigned the weight $\frac{h_j}{q}$. The standardization with $c_{\min}$ and $c_{\max}$ makes 0 the minimum and 1 the maximum attainable value for SMS.

Nogueira (2018) presents a stability measure that is based on the variances s_j^2 of the Bernoulli variables that describe the selection of the features:

$$\begin{aligned}
\text{SMN} &= 1 - \frac{\frac{1}{p}\sum_{j=1}^{p} s_j^2}{\frac{q}{mp}\left(1-\frac{q}{mp}\right)} \quad \text{with}\\
s_j^2 &= \frac{m}{m-1}\frac{h_j}{m}\left(1-\frac{h_j}{m}\right).
\end{aligned}$$

3.3.2.3 Adjusted Stability Measures

All of the previously defined stability measures do not take into account similarities between the features $X_1,\ldots,X_p$. These measures are called "unadjusted" in this thesis. Now, stability measures that include an adjustment for similar features are presented. The basic idea of adjusted stability measures is to adjust the scores $|V_i \cap V_j|$ in a way that different but highly similar features count towards stability.

Zucknick et al. (2008) extend the stability measure SMJ, considering the correlations between the features:

$$\begin{aligned}
\text{SMZ} &= \frac{2}{m(m-1)}\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}\frac{|V_i\cap V_j| + C(V_i,V_j) + C(V_j,V_i)}{|V_i\cup V_j|} \quad \text{with}\\
C(V_i,V_j) &= \sum_{x\in V_i}\frac{1}{|V_j|}\sum_{y\in V_j\setminus V_i}|\text{Cor}(x,y)|\ \mathbb{I}_{[\theta,\infty)}\left(|\text{Cor}(x,y)|\right).
\end{aligned}$$

$|\text{Cor}(x,y)|$ is the absolute Pearson correlation between x and y, $\theta \in [0,1]$ is a threshold, and $\mathbb{I}$ denotes the indicator function. This stability measure can be generalized by allowing

arbitrary similarity values in the interval $[0, 1]$ instead of the absolute correlations. If there are no similar features, SMZ is identical to SMJ.

Sechidis et al. (2020) extend the stability measure SMN so that it takes into account feature similarities. They call their new measure "effective stability" because it measures the stability of the information effectively contained in the sets of selected features. The stability measure is defined as

$$\text{SMES} = 1 - \frac{\text{trace}(CS)}{\text{trace}(C\Sigma)}.$$

$C \in \mathbb{R}^{p\times p}$ denotes the matrix of feature similarities. Matrix C can be chosen arbitrarily as long as the requirements $c_{ij} = c_{ji}\ \forall i \neq j$, $c_{ij} \in [0, 1]\ \forall i \neq j$, and $c_{ii} = 1\ \forall i \in \{1, \ldots, p\}$ are met. The matrices $S \in \mathbb{R}^{p\times p}$ and $\Sigma \in \mathbb{R}^{p\times p}$ are defined as

$$S_{ij} = \frac{m}{m-1}\left(\frac{h_{ij}}{m} - \frac{h_i}{m}\frac{h_j}{m}\right),$$

$$\Sigma_{ij} = \begin{cases} \frac{q}{mp}\left(1 - \frac{q}{mp}\right), & i = j, \\ \frac{\frac{1}{m}\sum_{i=1}^{m}|V_i|^2 - \frac{q}{m}}{p^2-p} - \frac{q^2}{m^2p^2}, & i \neq j, \end{cases}$$

with h_j denoting the number of sets that include feature X_j, h_{ij} indicating the number of sets that include both X_i and X_j, and $q = \sum_{j=1}^{p} h_j = \sum_{i=1}^{m} |V_i|$. Matrix S is an empirical covariance matrix, generalizing the variance terms s_j^2 in SMN. Matrix Σ is the theoretical covariance matrix under the assumption of a random feature selection with identical selection probabilities for all features and generalizes the denominator in SMN. If C equals the identity matrix, SMES is identical to SMN.

Zhang et al. (2009) also present stability measures that take into account similarities. Their scores are developed for the comparison of two gene lists. The scores they define are

$$\text{nPOGR}_{ij} = \frac{K + O_{ij} - E\left[K + O_{ij}\right]}{|V_i| - E\left[K + O_{ij}\right]}$$

with $ij \in \{12, 21\}$. K is defined as the number of genes that are included in both lists and regulated in the same direction. O_{ij} denotes the number of genes in list i that are not in list j but significantly positively correlated with at least one gene in list j. For each pair of gene lists, two stability scores are obtained.

Yu et al. (2012) combine the two scores nPOGR_{ij} and nPOGR_{ji} into one score for the special case $|V_i| = |V_j|$:

$$\text{nPOGR} = \frac{K + \frac{O_{ij}+O_{ji}}{2} - E\left[K + \frac{O_{ij}+O_{ji}}{2}\right]}{|V_i| - E\left[K + \frac{O_{ij}+O_{ji}}{2}\right]}$$

We generalize this score to be applicable in the general context of feature selection with arbitrary feature sets $V_1, \ldots, V_m$ by ...

1. ... replacing the quantity K by $|V_i \cap V_j|$.

2. ... allowing the similarities between the features to be assessed by an arbitrary similarity measure instead of only considering significantly positive correlations, that is, replacing $\frac{O_{ij}+O_{ji}}{2}$ by $\frac{A(V_i,V_j)+A(V_j,V_i)}{2}$ with A defined below.

3. ... replacing $|V_i|$, which is the maximum value of $K + \frac{O_{ij}+O_{ji}}{2}$, by $\frac{|V_i|+|V_j|}{2}$, the maximum value of $|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}$ (see Appendix B.1, p. 159 f.).

4. ... calculating the average of the scores for all pairs V_i, V_j, $i < j$.

As a result, the stability measure

$$\text{SMY} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2} - E\left[|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}\right]}{\frac{|V_i|+|V_j|}{2} - E\left[|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}\right]}$$

$$\text{with} \quad A(V_i, V_j) = |\{x \in (V_i \setminus V_j) : \exists y \in (V_j \setminus V_i) \text{ with similarity}(x, y) \geq \theta\}|$$

is obtained. E denotes the expected value for a random feature selection and can be assessed with the same Monte-Carlo procedure as described on page 25 for SMA. Similarity$(x, y) \in [0, 1]$ quantifies the similarity of the two features x and y and $\theta \in [0, 1]$ is a threshold.

If there are no similar features, SMY is identical to SMK. In situations where V_i and V_j greatly differ in size and contain many similar features, the value of SMY may be misleading. Consider a scenario with $|V_i| \gg |V_j|$, $|V_i \cap V_j| = 0$, $A(V_i, V_j) = |V_i|$, and $A(V_j, V_i) = |V_j|$. In such situations, there are many features in the larger set that are similar to the same feature in the smaller set. Even though the sets V_i and V_j greatly differ with respect to feature redundancy and resulting effects for model building such as over-fitting, the stability score attains its maximum value.

To overcome this drawback, in Bommert and Rahnenführer (2020), we define a new stability measure employing an adjustment $\text{Adj}(V_i, V_j)$ different from $\frac{A(V_i,V_j)+A(V_j,V_i)}{2}$ that fulfills

$$\max\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right] \leq \max\left[\left|\widetilde{V}_i \cap \widetilde{V}_j\right|\right] \text{ with } \left|\widetilde{V}_i\right| = |V_i| \text{ and } \left|\widetilde{V}_j\right| = |V_j|\,.$$

This means that the adjusted score for V_i and V_j cannot exceed the value of $\left|\widetilde{V}_i \cap \widetilde{V}_j\right|$ that would be obtained if two sets $\widetilde{V}_i$ and $\widetilde{V}_j$ with $\left|\widetilde{V}_i\right| = |V_i|$ and $\left|\widetilde{V}_j\right| = |V_j|$ were selected such that their overlap is maximal. This happens when $\widetilde{V}_i \subseteq \widetilde{V}_j$ or $\widetilde{V}_j \subseteq \widetilde{V}_i$. The resulting measure is

$$\text{SMA} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| + \text{Adj}(V_i, V_j) - E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right]}{\text{UB}\left[|V_i \cap V_j|\right] - E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right]}$$

with $\text{UB}\left[|V_i \cap V_j|\right]$ denoting an upper bound for $|V_i \cap V_j|$. Four different adjustments are considered. They are listed in Table 3.2.

For the adjustment of SMA-MBM, a graph is constructed. In this graph, each feature of $(V_i \setminus V_j) \cup (V_j \setminus V_i)$ is represented by a vertex. Vertices $x \in V_i \setminus V_j$ and $y \in V_j \setminus V_i$ are connected by an edge, if and only if similarity$(x, y) \geq \theta$. Figure 3.2 shows an example of such a graph. An edge in the graph means that the corresponding features of the two connected vertices should be seen as exchangeable for stability assessment. A matching of a graph is defined as a subset of its edges such that none of the edges share a vertex (Rahman, 2017, p. 63). A maximum matching is a matching that contains as many edges as possible. The size of the maximum matching is the number of edges that are included in the maximum matching. In Figure 3.2, the size of the maximum matching is 2. The size of the maximum matching can be interpreted as the maximum number of features in $V_i \setminus V_j$ and $V_j \setminus V_i$ that should be seen as exchangeable for stability assessment with the restriction that each feature in $V_i \setminus V_j$ may only be seen as exchangeable with at most one feature in $V_j \setminus V_i$ and vice versa. There are no edges between vertices that both correspond to features of $V_i \setminus V_j$ or $V_j \setminus V_i$, so

Name	Adjustment
SMA-MBM	$\text{Adj}_{\text{MBM}}(V_i, V_j) = \text{size of maximum bipartite matching}\,(V_i \setminus V_j, V_j \setminus V_i)$
SMA-Greedy	$\text{Adj}_{\text{Greedy}}(V_i, V_j) =$ greedy choice of most similar pairs of features determined by Algorithm 3.1 (introduced on page 25)
SMA-Count	$\text{Adj}_{\text{Count}}(V_i, V_j) = \min\{A(V_i, V_j), A(V_j, V_i)\}$ with A as defined for SMY
SMA-Mean	$\text{Adj}_{\text{Mean}}(V_i, V_j) = \min\{M(V_i, V_j), M(V_j, V_i)\}$ with $M(V_i, V_j) = \sum_{x \in V_i \setminus V_j : \left\lvert G_x^{ij} \right\rvert > 0} \frac{1}{\left\lvert G_x^{ij} \right\rvert} \sum_{y \in G_x^{ij}} \text{similarity}(x, y)$ and $G_x^{ij} = \{y \in V_j \setminus V_i : \text{similarity}(x, y) \geq \theta\}$

Table 3.2: Adjustment functions for the stability measure SMA.

the graph is bipartite (Rahman, 2017, p. 17). For the calculation of a maximum matching for a bipartite graph, there exist suitable algorithms (Hopcroft and Karp, 1973).

The calculation of the maximum bipartite matching has the computational complexity $\mathcal{O}\left((\text{number of vertices} + \text{number of edges}) \cdot \sqrt{\text{number of vertices}}\right)$ (Hopcroft and Karp, 1973) and hence can be very time consuming. Therefore, a new greedy algorithm for choosing the most similar pairs of features is introduced in Algorithm 3.1. It is used to calculate the adjustment in SMA-Greedy. The return value of the algorithm is always smaller than or equal to the size of the maximum bipartite matching of the corresponding graph, as shown in Appendix B.1 on page 160. The computational complexity of the algorithm is dominated by the sorting of the edges and therefore equals $\mathcal{O}\,(\text{number of edges} \cdot \log(\text{number of edges}))$.

For SMA-Count, $A(V_i, V_j)$ is the number of features in V_i that are not included in V_j but that have a similar feature in $V_j \setminus V_i$. The minimum $\min\{A(V_i, V_j), A(V_j, V_i)\}$ is used in order to guarantee that the adjusted score for V_i and V_j cannot exceed the value of $\left|\widetilde{V}_i \cap \widetilde{V}_j\right|$ that would be obtained if two sets $\widetilde{V}_i$ and $\widetilde{V}_j$ with $\left|\widetilde{V}_i\right| = |V_i|$ and $\left|\widetilde{V}_j\right| = |V_j|$ were selected such

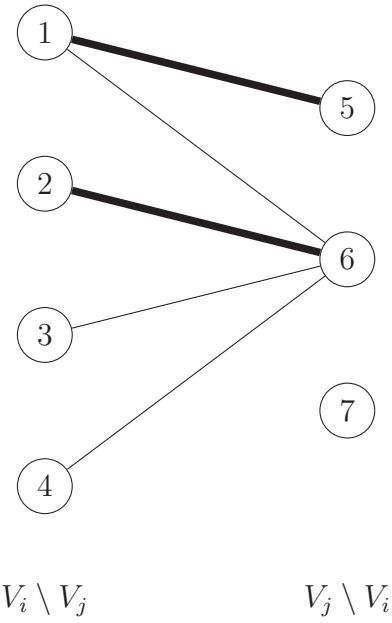


Figure 3.2: Example for a maximum bipartite matching. The bold edges belong to the maximum matching. The maximum matching is not unique for this graph.

input : Sets V_i and V_j and similarity values similarity(x, y) $\forall x \in V_i \setminus V_j, y \in V_j \setminus V_i$
output : $\text{Adj}_{\text{Greedy}}(V_i, V_j)$

```
size = 0
L_A = [X, Y, S] = list of tuples x ∈ V_i \ V_j, y ∈ V_j \ V_i, similarity(x, y) with
  similarity(x, y) ≥ θ, sorted decreasingly by similarity values
L_B = empty list
while length of L_A > 0 do
    [x, y, s] = first tuple of L_A
    add [x, y, s] to L_B
    remove all tuples in L_A that contain x or y
end
return length of L_B
```

Algorithm 3.1: Greedy choice of the most similar pairs of features.

that their overlap is maximal. For the situation in Figure 3.2, the adjustment of SMA-Count also has the value 2, because $A(V_i, V_j) = 4$ and $A(V_j, V_i) = 2$. $\min\{A(V_i, V_j), A(V_j, V_i)\}$ is always larger than or equal to the size of the maximum bipartite matching, see Appendix B.1 page 160 f. In Figure 3.2, $\min\{A(V_i, V_j), A(V_j, V_i)\}$ would be larger than the size of the maximum bipartite matching if an edge between the vertices 1 and 7 was added.

The adjustment of SMA-Mean is very similar to the one of SMA-Count. While $A(V_i, V_j)$ counts the number of features in $V_i \setminus V_j$ that have a similar feature in $V_j \setminus V_i$, $M(V_i, V_j)$ sums up the mean similarity values of the features in $V_i \setminus V_j$ to their similar features in $V_j \setminus V_i$. If there are no similarity values of features in $V_i \setminus V_j$ and $V_j \setminus V_i$ in the interval $[\theta, 1)$, the adjustments of SMA-Count and SMA-Mean are identical. Otherwise, the adjustment of SMA-Mean is smaller than the adjustment of SMA-Count, see Appendix B.1 page 160 f.

The expected values $E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right]$ cannot be calculated with a universal formula as they depend on the data specific similarity structure. However, they can be estimated by repeating the following Monte-Carlo procedure N times: 1. Randomly draw sets $\widetilde{V}_i \subseteq \{X_1, \ldots, X_p\}$ and $\widetilde{V}_j \subseteq \{X_1, \ldots, X_p\}$, with $\left|\widetilde{V}_i\right| = |V_i|$, $\left|\widetilde{V}_j\right| = |V_j|$, and equal selection probabilities for all features. 2. Calculate the score $\left|\widetilde{V}_i \cap \widetilde{V}_j\right| + \text{Adj}(\widetilde{V}_i, \widetilde{V}_j)$. An estimate for the expected value $E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right]$ is the average of the N scores. Zhang et al. (2009) suggest conducting $N = 10\,000$ repetitions. In situations with very small values of p and m, the expected values can be determined exactly by considering all possible combinations of sets $\widetilde{V}_i \subseteq \{X_1, \ldots, X_p\}$ and $\widetilde{V}_j \subseteq \{X_1, \ldots, X_p\}$ with $\left|\widetilde{V}_i\right| = |V_i|$ and $\left|\widetilde{V}_j\right| = |V_j|$.

Concerning the upper bounds $\text{UB}\left[|V_i \cap V_j|\right]$, it can easily be seen that $\min\{|V_i|, |V_j|\}$ is the tightest upper bound for $|V_i \cap V_j|$. However, this upper bound is not a good choice for $\text{UB}\left[|V_i \cap V_j|\right]$ because the stability measure could attain its maximum value for sets $V_i \subsetneqq V_j$ or $V_j \subsetneqq V_i$. To avoid this, $\text{UB}\left[|V_i \cap V_j|\right]$ must depend on both $|V_i|$ and $|V_j|$. Possible choices are for example $\frac{|V_i|+|V_j|}{2}$ or $\sqrt{|V_i| \cdot |V_j|}$. These choices are upper bounds for $|V_i \cap V_j|$ and they are met if and only if $V_i = V_j$. For $|V_i| \neq |V_j|$, the bounds differ and $\min\{|V_i|, |V_j|\} \leq \sqrt{|V_i| \cdot |V_j|} \leq \frac{|V_i|+|V_j|}{2}$ holds (see Appendix B.1, p. 150 f.) which makes $\sqrt{|V_i| \cdot |V_j|}$ more suitable. Therefore, we use $\text{UB}\left[|V_i \cap V_j|\right] = \sqrt{|V_i| \cdot |V_j|}$. If there are no similar features in the data set, all SMA variants are identical to SMU, independent of the choice of adjustment.

3.3.3 Stability Selection

Stability selection (Meinshausen and Bühlmann, 2010; Shah and Samworth, 2013) is a flexible framework for selecting a stable subset of features. It can be combined with any feature selection algorithm for which the number of features to choose can be set. The basic idea is to repeatedly apply the feature selection algorithm on subsamples of a given data set and to finally select the features that have been selected for sufficiently many subsamples.

More precisely, B subsamples of size $\lfloor n/2 \rfloor$, with n denoting the number of observations in the data set, are drawn and the remaining parts of the data set are used as subsamples as well (complementary pairs, Shah and Samworth, 2013), resulting in $2B$ subsamples. For each of the subsamples, the feature selection algorithm is applied and selects q features. Then, for each feature X_i, the selection frequency $\hat{\pi}_i$, that is, the fraction of subsamples for which feature X_i has been selected, is calculated. The set of finally selected features is defined as $\{X_i : \hat{\pi}_i \geq \pi_{\text{thr}}\}$ where $\pi_{\text{thr}} \in [0, 1]$ is an arbitrary threshold. This procedure controls the per-family error rate (PFER), that is, the expected number of uninformative features that are selected. An upper bound is given by PFER $\leq \frac{q^2}{(2\pi_{\text{thr}}-1)p}$ (Meinshausen and Bühlmann, 2010) where p denotes the total number of features in the data set. q, π_{thr}, and PFER are hyperparameters and it is not obvious to which values they should be set. It is sufficient to set two of them, the third one can be calculated based on the upper bound for PFER. A major drawback of stability selection is its enormous run time.

3.4 Multi-Criteria Optimization

In this section, the multi-criteria optimization problem is defined and an algorithm for solving it is presented. As the solution of a multi-criteria optimization problem usually is a set of points, an algorithm for automatically selecting a good point from this set is introduced.

3.4.1 Optimization Problem

Let M be some finite set and $f : M \to \mathbb{R}^t$ an objective function to maximize. For example, M could be the set of configurations of classification methods that have been evaluated and $f_1, \ldots, f_t$ could be t performance criteria. Note that each minimization problem can be transformed into a maximization problem by multiplication with -1. If $t = 1$ holds, all points in the image $f(M) = \{y \in \mathbb{R}^t : \exists x \in M \text{ with } f(x) = y\}$ are comparable and therefore $f(M)$ has a distinct maximum. However, when $t \geq 2$ holds, some of the elements of $f(M)$ may not be comparable: they may be smaller in one component and larger in another one. The set $f(M)$ thus does not necessarily have a distinct maximum. Instead, there will likely be a set of incomparable maximal points which is called the Pareto front. A point $y \in \mathbb{R}^t$ Pareto dominates another point $z \in \mathbb{R}^t$ if and only if $\forall i = 1, \ldots, t : y_i \geq z_i$ and $\exists j \in \{1, \ldots, t\} : y_j > z_j$. The Pareto front is the subset of $f(M)$ that contains no dominated points: $\text{PF} = \{y \in f(M) : \nexists z \in f(M) \text{ that dominates } y\}$. The set $\{x \in M : f(x) \in \text{PF}\}$ is called Pareto set. For a more detailed introduction in multi-criteria optimization and Pareto optimality see Miettinen (2004, pp. 5 ff.).

3.4.2 Optimization Algorithm

There exist many optimization algorithms for finding the Pareto front or the Pareto set, respectively. Popular approaches include evolutionary and model-based algorithms. Coello Coello et al. (2007) present an overview of evolutionary algorithms and Horn et al. (2015)

give an overview of model-based algorithms for multi-objective optimization. All of these optimization strategies are iterative procedures and therefore have long run times, especially if the evaluation of the objective function f takes long.

A very simple optimization algorithm is random search (Bergstra and Bengio, 2012): N values $x \in M$ are drawn at random and $f(x)$ is calculated. N denotes a given budget of evaluations of f. Then, the Pareto front and the Pareto set are estimated by calculating the subset of non-dominated points of the set of evaluated x-values. For $N \to \infty$, these estimates converge with probability 1 to the true Pareto front and Pareto set (Laumanns and Zenklusen, 2011). The main advantage of a random search is that the calculations of $f(x)$ for the N randomly selected x-values can be conducted in parallel. Therefore, on high performance compute clusters, performing a random search is faster than the more elaborate iterative algorithms by orders of magnitude.

3.4.3 Selection of Points from Pareto Front

Having obtained a Pareto front, it often is desirable to choose one point from the front that provides a good compromise between the objectives. In the following, an algorithm for choosing such a point in an automated way is presented for the bi-objective maximization problem of finding a configuration with maximal predictive accuracy and maximal feature selection stability. The algorithm can be used for arbitrary bi-objective maximization problems and it can easily be extended for more than two objectives. Here, accuracy and stability are used as objectives, because this is the use case in this thesis.

Algorithm 3.2 states the algorithm "ϵ-constraint selection". It implements an a posteriori ϵ-constraint scalarization method, see Miettinen (2008) for more details on scalarization methods. Figure 3.3 illustrates the procedure of Algorithm 3.2. With ϵ-constraint selection, only the configurations that achieve high predictive accuracy are considered. More precisely, only configurations, whose predictive accuracy deviates at most *acc.const* from the best accuracy that was observed among all configurations, are considered. If *acc.const* is set to 0, this corresponds to single-objective optimization of the predictive accuracy. Of the remaining configurations, the configurations with comparably low feature selection stability are discarded. Then, the best configuration is chosen based on predictive accuracy. So, the focus of ϵ-constraint selection is selecting a configuration with high predictive accuracy. Avoiding configurations with low stability is the secondary criterion.

input : All configurations, constants *acc.const*, *stab.const* $\in [0, 1]$
output : Best configuration
Determine maximal accuracy *acc.max* among all configurations.
Remove all configurations with accuracy $<$ *acc.max* $-$ *acc.const*.
Among the remaining configurations, determine the maximal stability *stab.max*.
Remove all configurations with stability $<$ *stab.max* $-$ *stab.const*.
Among the remaining configurations, determine the maximal accuracy *acc.max.end*.
Remove all configurations with accuracy $<$ *acc.max.end*.
if more than one configuration left **then**
    Among the remaining configurations, determine the maximal stability *stab.max.end*.
    Remove all configurations with stability $<$ *stab.max.end*.
end
if more than one configuration left **then**
    Randomly choose one of the remaining configurations.
end
return Best configuration.

Algorithm 3.2: ϵ-constraint selection

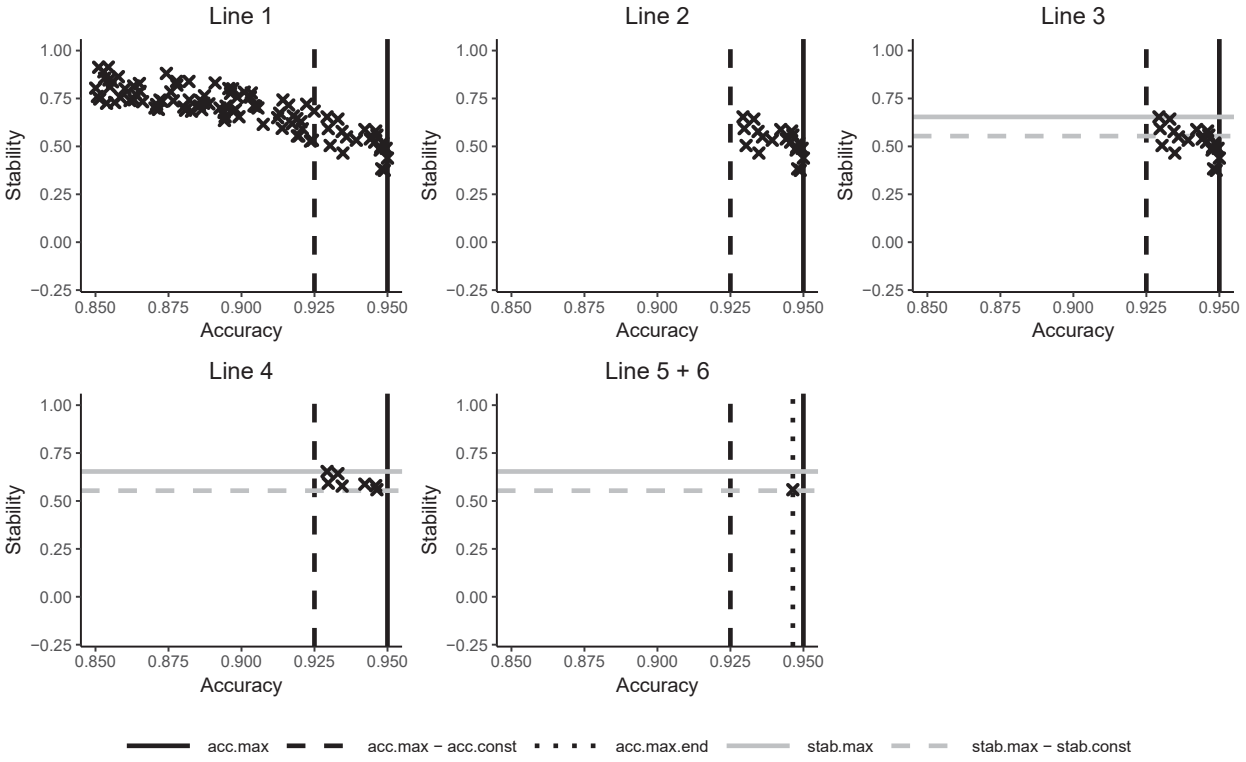


Figure 3.3: Illustration of Algorithm 3.2.

Chapter 4

Data Sets and Software

In this chapter, the data sets and software used for the analyses in the following chapters are presented.

4.1 Data Sets

Most analyses are based on 12 classification data sets from various domains. These data sets have been selected considering four categories of data set sizes with p denoting the number of features and n denoting the number of observations in the data sets:

small $p \leq n$:	$p \in [50, 500]$	and	$n \in [p, 3\,000]$
large $p \leq n$:	$p \in (500, 2\,000]$	and	$n \in [p, 5\,000]$
small $p > n$:	$p \in (100, 3\,000]$	and	$n \in (100, p)$
large $p > n$:	$p \in (3\,000, 15\,000]$	and	$n \in (100, p)$

To find suitable data sets for all categories, a search on the machine learning platform OpenML (Vanschoren et al., 2013; Casalicchio et al., 2017) and within the R package *datamicroarray* (Ramey, 2016) has been conducted. Data sets with more than two classes were converted into two-class data sets by choosing two classes whose discrimination is not trivial. For all data sets, constant features were removed. The criteria and characteristics that are considered in the following, were assessed for the selected part of the data sets. The search has been performed according to the following criteria:

1. The data sets must contain real data (not artificial data).
2. There must not be any missing values.
3. The data sets may only consist of numeric features. Not more than half of the features per data set may have five or less unique values.
4. The data sets should not be extremely unbalanced: the majority class must not contain more than 75% of all observations.

For all candidate data sets, the matrix of absolute Pearson correlations between all features was analyzed as a measure of similarity between the features. For each feature, the number of highly correlated other features (absolute correlation of at least 0.9) was assessed. For aggregation, the mean number of highly correlated other features per feature was calculated for each data set. For each of the four categories of data sets, it was aimed to choose one data set with (almost) no similar features, one data set with a medium number of similar features, and one data set with many similar features. For the data set categories "small $p \leq n$", "large $p \leq n$", and "large $p > n$" this could be achieved, but for category "small

$p > n$", no data sets with a medium number of similar features could be found. Therefore, two data sets with almost no similar features were chosen instead. An overview of the data set characteristics of all 12 data sets is given in Table 4.1.

The task of data set *sonar* (Gorman and Sejnowski, 1988) is to discriminate between a metal cylinder and an approximately cylindrical rock based on sonar signals. The features correspond to the off bouncing sonar signals transmitted from various angles and under different conditions.

The context of data set *kc1-binary* (Koru and Liu, 2005) is the prediction of the defective modules in the NASA product KC1. The observations of this data set are classes (units of object oriented software) and the features describe properties of these classes. Originally, the properties were assessed separately for all methods[1] of a class. The data set contains the information aggregated over all the methods of a class by using several aggregation functions (minimum, maximum, sum, and average).

The task of data set *tecator* (Borggaard and Thodberg, 1992; Thodberg, 1993) is to predict the fat content of a meat sample based on its near infrared absorbance spectrum. The data was recorded on a Tecator Infratec Food and Feed Analyzer. The features of the data set are 100 near infrared absorbance values, the first 22 principal components of these values (with weights calculated only on part of the data set), the protein content of the meat, and its moisture. To define two classes, the mean fat content was calculated and an observation was classified as "positive" if its fat content was below the average fat content, otherwise it was classified as "negative".

The task of data set *har* is to recognize human activities based on data recorded with a smartphone. The data was obtained in an experiment. Volunteers performed different activities wearing a smartphone around their waist and were recorded on video for labeling. The sensor signals obtained with the smartphone (accelerometer and gyroscope) were preprocessed by applying noise filters and then sampled in fixed-width sliding windows of 2.56 seconds and 50% overlap. From each window, a vector of features was obtained by calculating variables from the time and frequency domain. Here, only the two classes "sitting" and "standing" are considered to obtain a two-class-data set. These are two classes whose separation is not trivial (Anguita et al., 2013).

The task of data set *gina_agnostic* (Guyon, 2006) is to discriminate between even and odd handwritten two-digit-numbers. For each number, 485 pixels describe the first digit and 485 the second digit. The features state the grayness of the pixels. The data set was part of the Agnostic Learning vs. Prior Knowledge Challenge.

The data set *dilbert* was part of the AutoML challenge (Guyon et al., 2016). The identity of the data set and the type of data are concealed. However, it is known that it is a real world data set. The original data set consists of 5 classes. Here, to obtain a two-class-data set, only the two largest classes are considered because no further information is available.

The task of data set *lsvt* (Tsanas et al., 2014) is to assess whether voice rehabilitation treatment of Parkinson's disease patients leads to phonations considered acceptable or unacceptable. The features of this data set are various dysphonia measures of the speech input.

The task of data set *christensen* (Christensen et al., 2009) is to discriminate between DNA from different anatomic sites based on methylation data. Non-pathologic human tissues from 10 anatomic sites were analyzed at various autosomal CpG sites that are associated with cancer-related genes using Illumina GoldenGate methylation bead arrays. Each feature contains the methylation information obtained at one autosomal CpG site. The data set is

[1]Methods in object oriented programming are similar to functions in R. Each method is defined within a class and belongs to this class.

Name	Reference	p	n	PP	MHCF	ID
sonar	Gorman and Sejnowski, 1988	60	208	0.53	0.10	40
kc1-binary	Koru and Liu, 2005	86	145	0.59	6.81	1 066
tecator	Borggaard and Thodberg, 1992	124	240	0.57	91.40	851
har	Anguita et al., 2013	561	3 683	0.52	5.28	1 478
gina_agnostic	Guyon, 2006	970	3 468	0.51	0.00	1 038
dilbert	Guyon et al., 2016	2 000	4 095	0.50	66.24	41 163
lsvt	Tsanas et al., 2014	307	126	0.67	18.42	1 484
christensen	Christensen et al., 2009	1 413	217	0.61	1.00	–
gravier	Gravier et al., 2010	2 905	168	0.66	0.13	–
eating	Hantke et al., 2016	6 365	273	0.51	4.51	1 233
arcene	Guyon et al., 2004	9 961	200	0.56	29.97	1 458
chiaretti	Chiaretti et al., 2004	12 625	128	0.74	0.05	–

Table 4.1: Information about the data sets: number of features (p), number of observations (n), proportion of majority class (PP), mean number of highly correlated other features per feature (MHCF), and identification number of the data set on OpenML (ID).

available in the R package *datamicroarray*, which downloads it from Array Express (Athar et al., 2019). In the R package *datamicroarray*, the DNA from blood and placenta are considered as separate classes, while all the other tissues are combined to one class called "other". To obtain a two-class data set, the samples from placenta (the smaller of the two separate classes) are added to the class "other" here.

The task of data set *gravier* (Gravier et al., 2010) is to discriminate between two groups of breast cancer patients. The two groups are patients with no event at 5 years after diagnosis and patients with early metastasis. The tissue of small invasive ductal carcinomas without axillary lymph node involvement (T1T2N0) was analyzed with comparative genomic hybridization (CGH) array. Array CGH compares the patients' genomes against a reference genome with respect to copy number variations, utilizing the principle of competitive fluorescence. The features of the data set describe the differences between the two genomes based on the fluorescence data. The data set is available in the R package *datamicroarray*, which downloads it from Array Express (Athar et al., 2019).

The task of data set *eating* (Hantke et al., 2016) is to determine what a person is eating while speaking. The data comes from volunteers eating different types of food while speaking. The features are computed from the recorded speech sounds using the software openSMILE (Eyben et al., 2010). The data set was part of the Interspeech 2015 Computational Paralinguistics Challenge (Schuller et al., 2015). The original data set contains 7 classes (types of food). Here, only the classes "apple" and "nectarine" are considered to obtain a two-class-data set. These are two classes whose separation is not trivial (Wagner et al., 2015).

The task of data set *arcene* (Guyon et al., 2004) is to distinguish cancer from normal patterns based on mass-spectrometric data. The data set was part of the NIPS 2003 feature selection challenge. It was created by merging three mass-spectrometry data sets to obtain enough samples. The samples include patients with cancer (ovarian or prostate cancer) and healthy or control patients. The original features indicate the abundance of proteins with a given mass value in human sera. For the challenge, a number of distractor features was added. The distractor features were drawn at random from a distribution resembling the distribution of the real features, but carrying no information about the class labels. 30% of the features in the data set are distractor features. The order of the features and patterns was randomized.

The task of data set *chiaretti* (Chiaretti et al., 2004) is to discriminate between T-cell and B-cell acute lymphocytic leukemia based on microarray data. The gene expression data was measured on Affimetrix HGU95aV2 chips. The features contain the gene expression information corresponding to the different positions on the microarray chip. The data set is available in the R package *ALL* (Li, 2018). Here, a logarithm transformation is applied to the gene expression data for normalization. The data set is also available in the R package *datamicroarray*, but there the various genetic translocations (instead of T-cell and B-cell) are used as classes.

Figures 4.1 and 4.2 show PCA plots of the data sets. For Figure 4.1, the data is not scaled for the computation of the principal components. For Figure 4.2, all features are scaled to unit variance. The data sets *kc1-binary*, *lsvt*, and *eating* contain outlying observations, which make the scaled plots more suitable for interpretation. The plots provide insight into the distribution of the classes and give an impression about how easy discriminating between the two classes per data set will be. However, it should be kept in mind that only part of the data variation can be displayed in two dimensions. For data sets *sonar*, *kc1-binary*, *lsvt*, *gravier*, *eating*, and *arcene*, the classes seem quite difficult to separate. For the data sets *har* and *gina_agnostic*, the class discrimination tasks appear to be at medium difficulty. For data sets *tecator*, *dilbert*, *christensen*, and *chiaretti*, the classes seem to be separable easily. The PCA plots for data sets *christensen* and *arcene* show different clusters within the classes, that is, the distributions are not homogeneous within the classes. Figure A.1 in Appendix A.1 shows the PCA plots for data set *christensen* with all original classes being indicated. It can be observed that the two clusters of class "blood" are formed by the two subclasses "blood" and "guthrie blood" (blood from newborn infants). The subcluster of class "other", which is visible in the plot with the unscaled data, contains the measurements from placenta tissue. For data set *arcene*, the cluster structure is likely due to *arcene* being created by merging three data sets.

4.2 Software

All of the following analyses have been conducted with R version 3.5.1 (R Core Team, 2018). The experiments have been run on the high performance Linux HPC cluster at TU Dortmund University (LiDO3) using the R package *batchtools* (Lang et al., 2017). Within the experiments, the machine learning framework provided by the R package *mlr* (Bischl et al., 2016) was used. For calculations related to feature selection stability, the R package *stabm* (Bommert and Lang, 2020) was employed. The graphics for the analyses were created with the R package *ggplot2* (Wickham, 2016). Further R packages that were only used for some analyses are indicated in the respective chapters. The R source code for all analyses presented in this thesis is publicly available at `https://github.com/bommert/phd-thesis`.

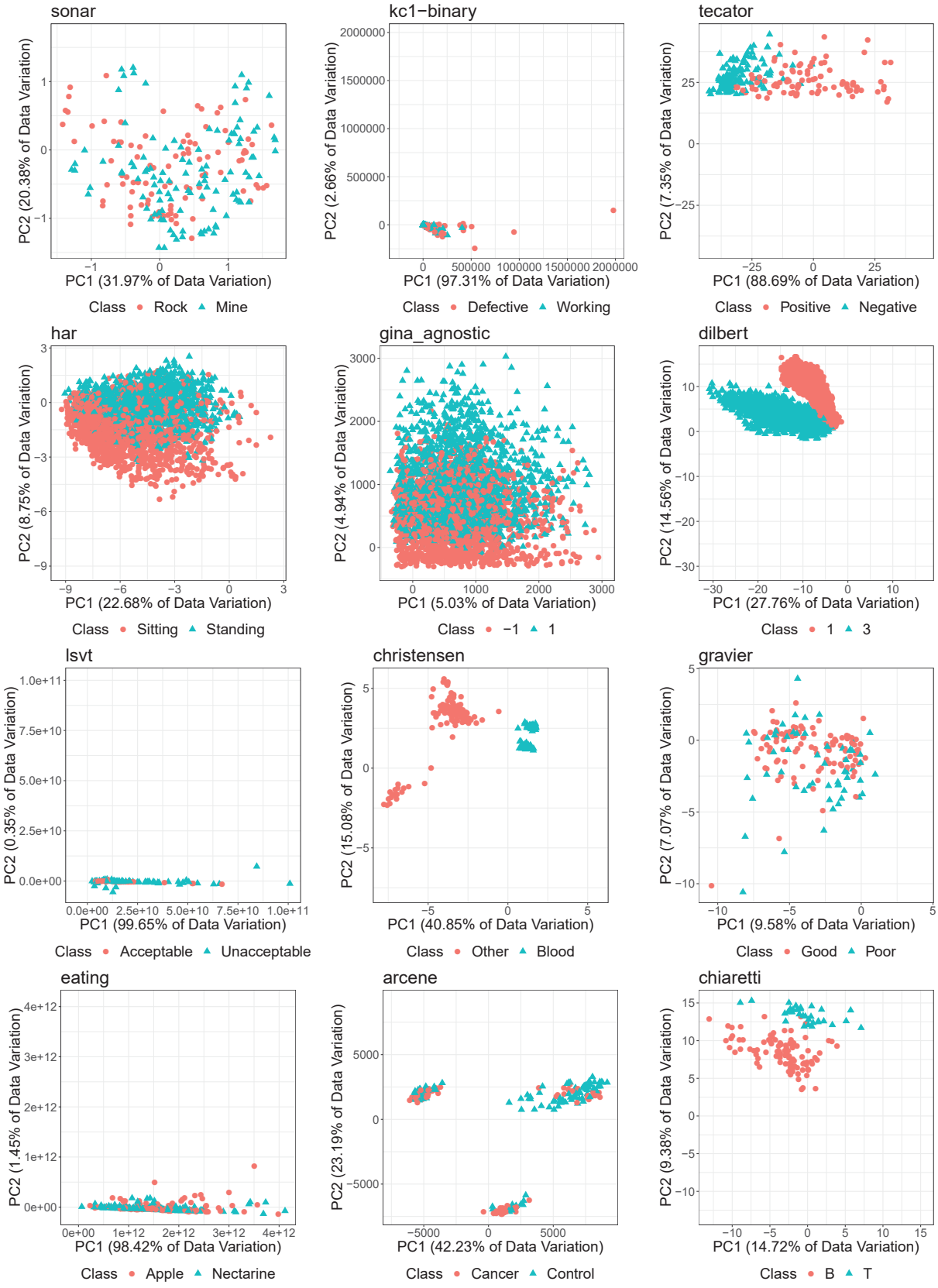


Figure 4.1: PCA plots of all 12 data sets. The data is not scaled for the computation of the principal components.

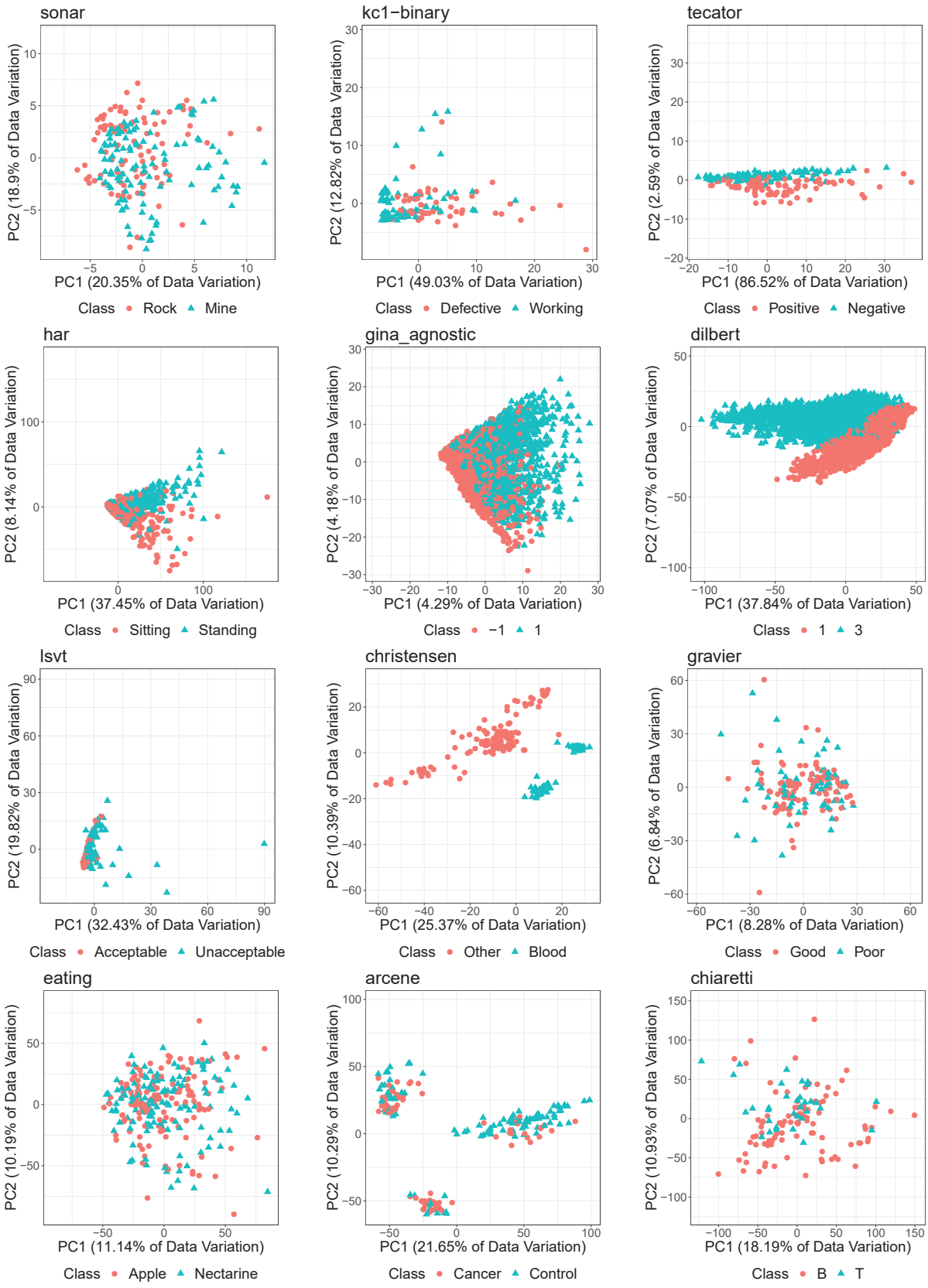


Figure 4.2: PCA plots of all 12 data sets. The data is scaled for the computation of the principal components.

Chapter 5

Benchmark of Filter Methods

The filter methods presented in Section 3.1 are compared and benchmarked in this chapter. The aim of these analyses is selecting the best filter methods, so that these methods can be employed in future analyses, instead of having to try all filter methods. In Section 5.1, the scaling behavior of the filter methods is analyzed. In Section 5.2, the filters are compared empirically with respect to the orders in which they select the features. The goal of this analysis is finding groups of filter methods that rank the features in a similar way. In Section 5.3, the best filter methods with respect to predictive performance when combined with a predictive model and with respect to run time are assessed.

For all of the filter methods described in Section 3.1, there are already implementations in R packages and most of them have been integrated into the machine learning package *mlr* (Bischl et al., 2016). Table 5.1 lists for all filter methods the R package in which they are implemented and their names in *mlr*.

The analyses conducted in this chapter are based on the analyses in Bommert et al. (2020). In Bommert et al. (2020), all of the filter methods in Table 5.1 as well as two additional methods are analyzed. The filter methods omitted here are a random forest permutation importance filter and a filter based on univariate model scores. The first one is left out because it takes very long to compute and there already is a random forest permutation importance filter in this study. The second one is omitted because it is not clear which model and which resampling strategy for performance evaluation should be used. Its default implementation in *mlr* is subject to strong variations due to stochastic effects. The data sets, based on which the filter methods are compared, differ between this thesis and the study in Bommert et al. (2020). Four data sets are used in both studies. Eight data sets are considered in this thesis, but not in Bommert et al. (2020); twelve data sets are considered in Bommert et al. (2020), but not in this thesis. In both works, the analyses of the filters are conducted in analogous ways and the results are similar.

5.1 Comparison with Respect to Scaling Behavior

In this section, all filters are analyzed with respect to their scaling behavior. First, it is investigated how the run times change, when for a given data set the number of features to select is increased. Second, the run times for calculating the scores for all features for data sets with an increasing number of observations or an increasing number of features are analyzed.

5.1.1 Experimental Setup

All run time analyses are based on a reduced version of data set *gina_agnostic*. The number of observations is reduced to 970, so that the number of observations equals the number of

Filter	Name in *mlr*	R package
anova.test	anova.test	*mlr* (Bischl et al., 2016)
limma	—	*limma* (Ritchie et al., 2015)
sam	—	*samr* (Tibshirani et al., 2011)
kruskal.test	kruskal.test	*mlr* (Bischl et al., 2016)
chi.squared	FSelector_chi.squared	*FSelector* (Romanski and Kotthoff, 2018)
auc	auc	*mlr* (Bischl et al., 2016)
oneR	FSelector_oneR	*FSelector* (Romanski and Kotthoff, 2018)
variance	variance	*mlr* (Bischl et al., 2016)
permutation	ranger_permutation	*ranger* (Wright and Ziegler, 2017)
impurity	ranger_impurity	*ranger* (Wright and Ziegler, 2017)
info.gain	FSelectorRcpp_information.gain	*FSelectorRcpp* (Zawadzki and Kosinski, 2018)
gain.ratio	FSelectorRcpp_gain.ratio	*FSelectorRcpp* (Zawadzki and Kosinski, 2018)
sym.uncert	FSelectorRcpp_symmetrical.uncertainty	*FSelectorRcpp* (Zawadzki and Kosinski, 2018)
MIM	praznik_MIM	*praznik* (Kursa, 2018)
MRMR	praznik_MRMR	*praznik* (Kursa, 2018)
JMI	praznik_JMI	*praznik* (Kursa, 2018)
JMIM	praznik_JMIM	*praznik* (Kursa, 2018)
DISR	praznik_DISR	*praznik* (Kursa, 2018)
NJMIM	praznik_NJMIM	*praznik* (Kursa, 2018)
CMIM	praznik_CMIM	*praznik* (Kursa, 2018)

Table 5.1: Names of the filter methods as introduced in Section 3.1, names in *mlr*, and R packages from which the implementations are taken.

features in the resulting data set. For creating the reduced data set, 970 observations are selected at random, such that 50% of the observations of the new data set belong to the positive and 50% to the negative class. The reduced data set is called "base data set" for the remainder of this section.

For the first analysis scenario, all filter methods are applied to the base data set. The filter hyperparameter, which indicates the percentage of features to select, is varied from 10% to 100% in steps of 10%. For the second analysis, new data sets of different sizes are created from the base data set. For the study with an increasing number of observations, new data sets with all features and 10% to 100% of the observations (with steps of 10%) are generated. All of these newly created data sets consist of 50% positive and 50% negative observations. For the analysis with an increasing number of features, new data sets with all observations and 10% to 100% of the features (with steps of 10%) are created. Then, all filter methods are applied to these new data sets, calculating scores for all features. For all run time analyses, the median run time of 1 000 repetitions is computed. These repetitions are computed on exactly the same data sets. The time measurements have been conducted on a high performance compute cluster in a randomized order.

5.1.2 Results

Regarding the first analysis scenario, Figure A.2 in Appendix A.2 displays the run times of all filter methods for the different percentages of features to select. For each filter method, an individual plot is shown because of the large differences in run time. Many of the run time lines are very jagged. For these filter methods, the run time does not depend on the number of features to select. The scores for all features are calculated and then, the best features are selected. The differences in run times are due to effects of the high performance compute cluster. Also, the differences are very small, as the scales of the y-axes show. For the filter methods from the *praznik* package except for *MIM*, the run times increase with the number of features to select. As described in Section 3.1, the *praznik* filters except for *MIM* perform an iterative forward selection and hence only calculate as many filter scores as needed. To compare the differences in scaling behavior between the filter methods, Figure 5.1 shows relative run times for all filter methods. For each filter, the run times are divided by the respective run time for selecting 10% of the features. So, the relative run time for selecting 10% of the features equals 1 for all filter methods. The run time of filter *CMIM* increases the strongest among all filter methods. For *NJMIM* and *JMIM*, strong increases in run time can be observed as well. For *DISR*, *MRMR*, and *JMI*, the run times increase only slightly. For all other filter methods, the run times are constant with respect to the number of selected features.

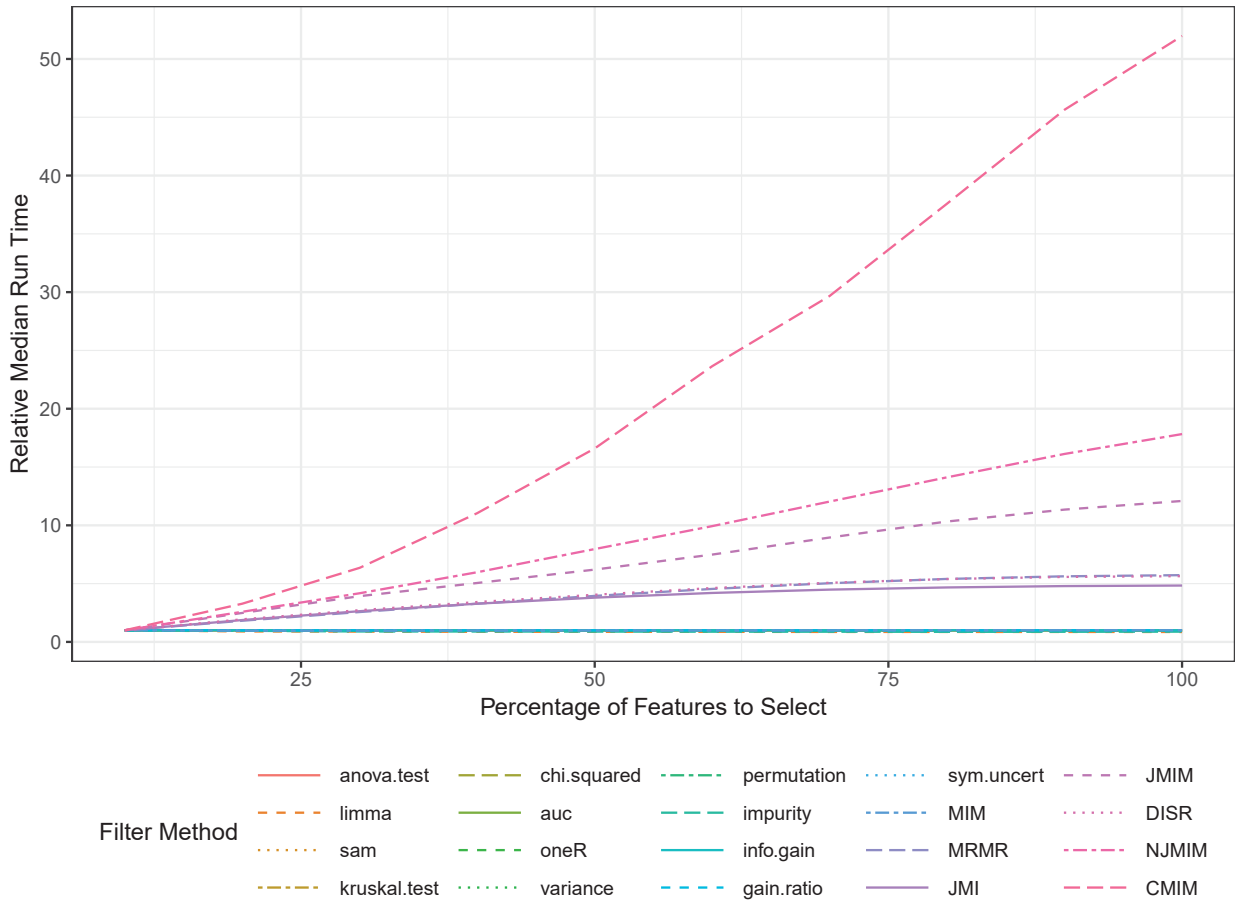


Figure 5.1: Relative run time for filtering with respect to the percentage of features to select. For each filter, the run times are divided by the respective run time for selecting 10% of the features.

Regarding the second analysis scenario, Figure A.3 in Appendix A.2 displays the run times for the calculation of the scores of all features for data sets with an increasing number of observations. For all filter methods, the increase in run time is linear with respect to the number of observations in the data set. For a comparison of the scaling behaviors between filter methods, Figure 5.2 shows relative run times. For each filter, the run times are divided by the respective run time for a data set with 10% of the observations of the base data set. The increase in run time is the strongest for filter *permutation*. For the filters *sym.uncert*, *info.gain*, *gain.ratio*, and *impurity*, strong increases in run time are observed as well. For the other filters, the run time increases only slightly with the number of observations in the data set. The smallest increases in run time are observed for filters *anova.test* and *variance*.

Figure A.4 in Appendix A.2 displays the run times for the calculation of the scores of all features for data sets with an increasing number of features. For all filter methods except for the iterative *praznik* filters (*MRMR*, *JMI*, *JMIM*, *DISR*, *NJMIM*, and *CMIM*), the increases in run time are linear with respect to the number of features in the data set. For the iterative *praznik* filters, the run times increase more than linearly, possibly quadratically. To compare the scaling behaviors between the filters, Figure 5.3 shows relative run times. For each filter, the run times are divided by the respective run time for a data set with 10% of the features of the base data set. It can be observed that there are large differences in the increases in run time between the iterative *praznik* filters and the rest of the filters. For the iterative filters,

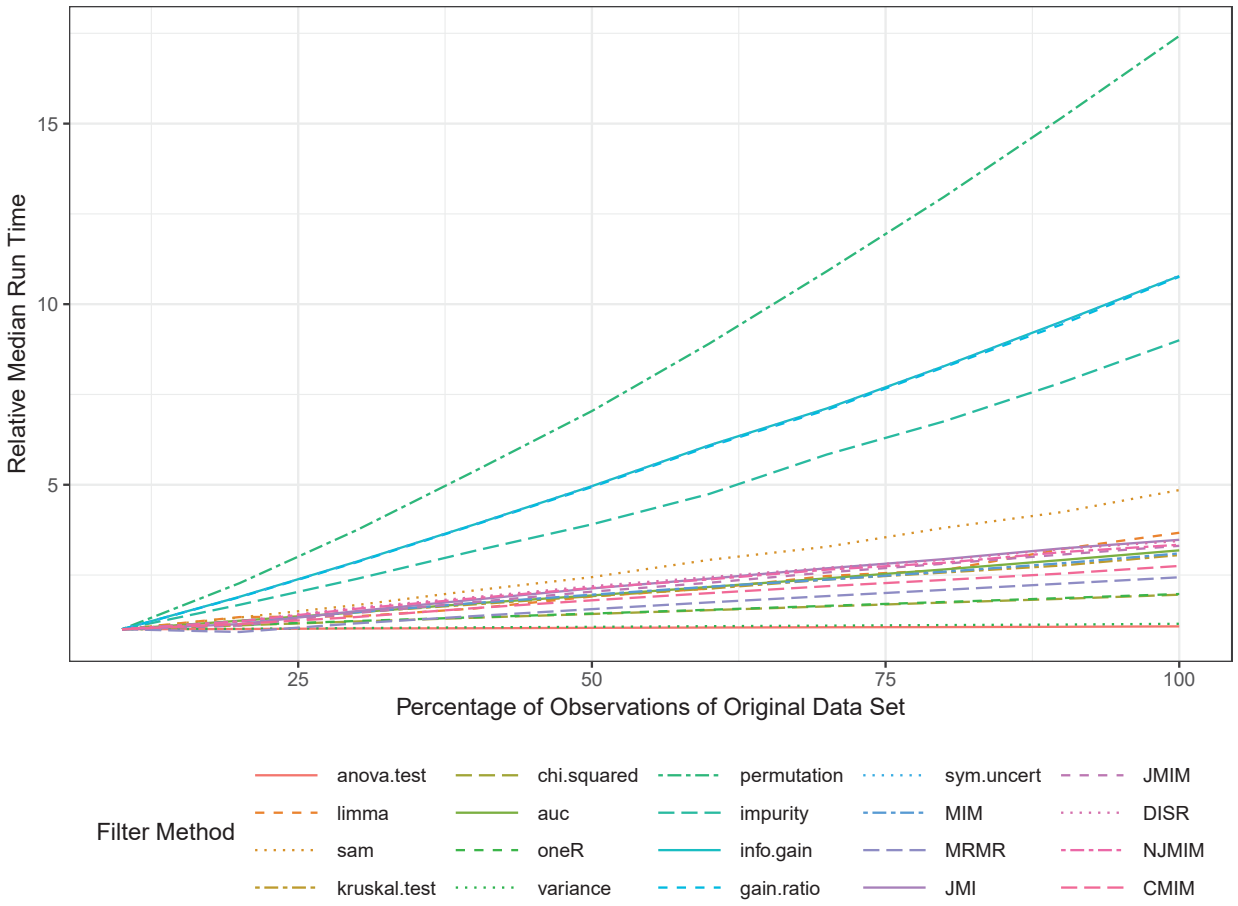


Figure 5.2: Relative run time for calculating the scores for all features for data sets with different numbers of observations (identical numbers of features). For each filter, the run times are divided by the respective run time for a data set with 10% of the observations of the base data set.

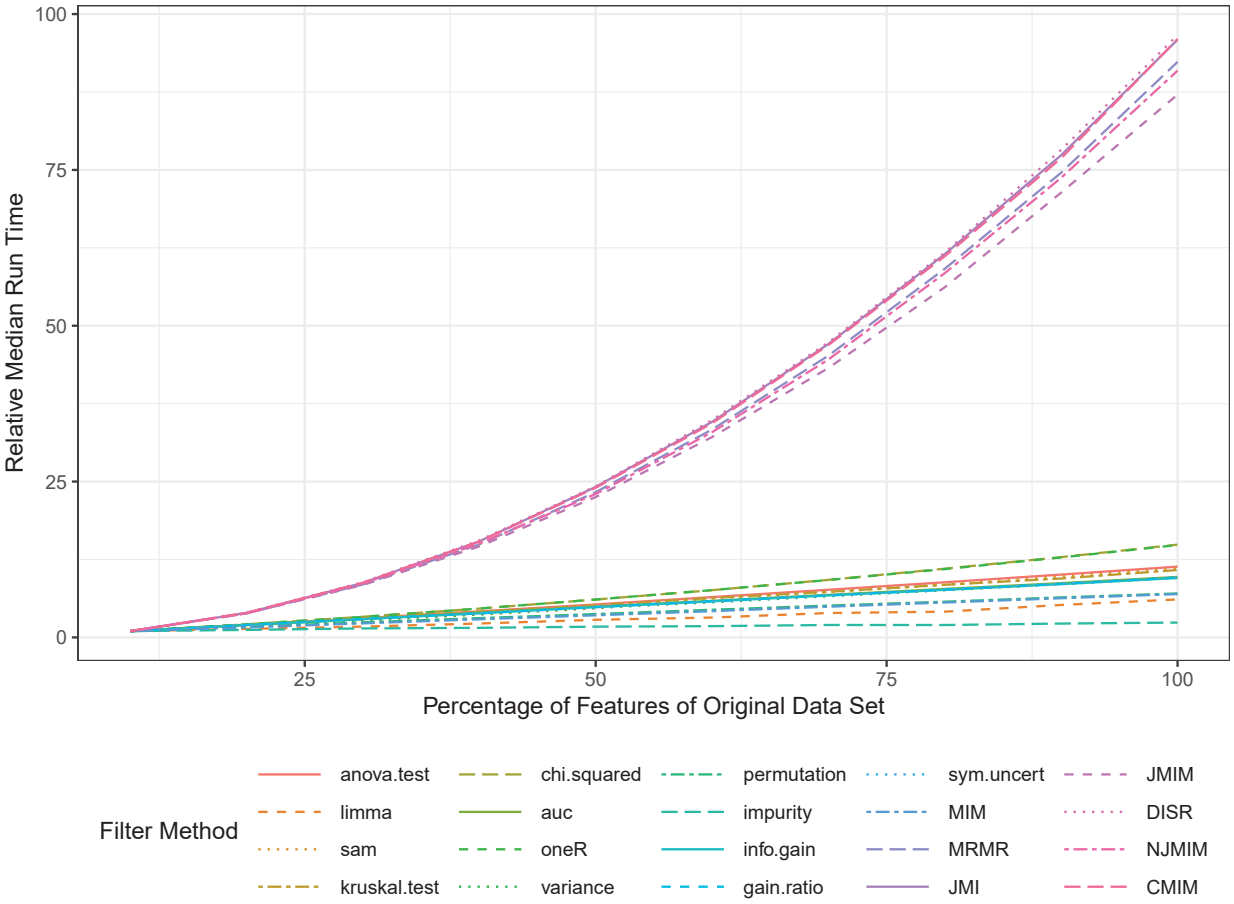


Figure 5.3: Relative run time for calculating the scores for all features for data sets with different numbers of features (identical numbers of observations). For each filter, the run times are divided by the respective run time for a data set with 10% of the features of the base data set.

the run times increase much more than for the other filters. Within the two groups, the increasements in run time with respect to the number of features in the data set are similar.

5.2 Comparison with Respect to Feature Ranking

In this analysis, the aim is finding out which filter methods are similar with respect to feature ranking. The question behind this analysis is if the filter methods can be grouped into sets with similar behavior and if these groups contain redundant information such that some members of each group can be neglected. For each filter method and each data set presented in Section 4.1, the filter scores for all features are computed.

To assess the similarity of the filter methods, the orders in which they select the features are compared. For each data set, the Spearman rank correlations between the selection orders of all pairs of filter methods are determined. Rank correlations are considered because the associations between different filter scores are not necessarily linear. The results are displayed in Figures A.5 and A.6 in Appendix A.2. To draw conclusions based on all data sets, the rank correlations are averaged across data sets with the arithmetic mean. Figure 5.4 displays the mean rank correlations between all pairs of filter methods. The higher the rank correlation between two filter methods, the more similar they are.

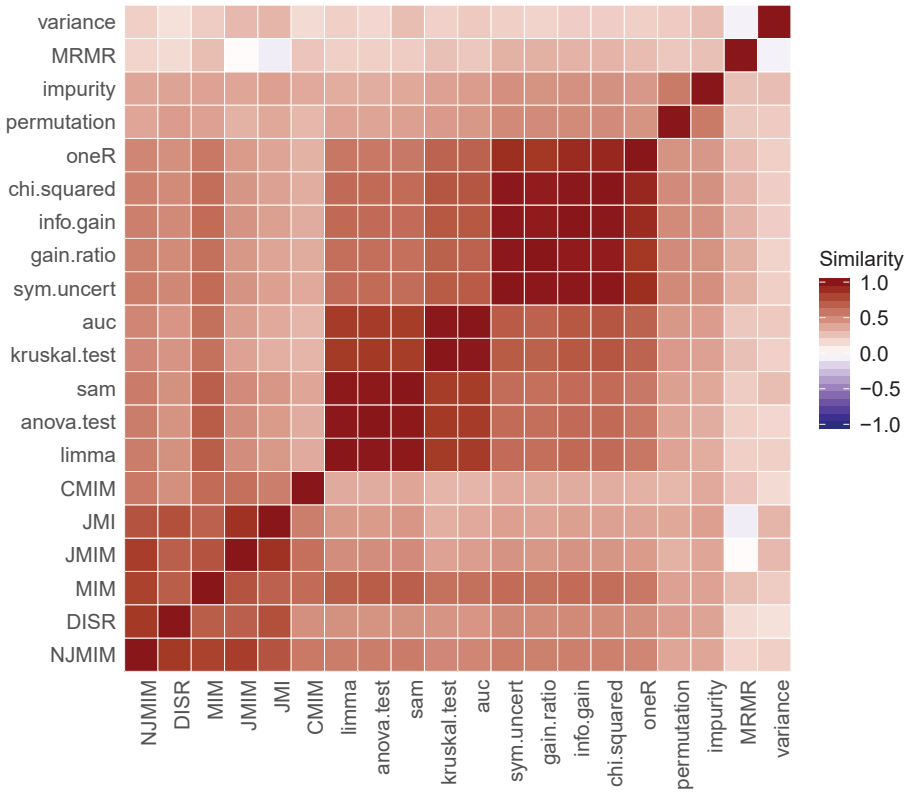


Figure 5.4: Rank correlations between the selection order of all features for all pairs of filter methods on all 12 data sets, averaged by the arithmetic mean. The filter methods are ordered by average linkage hierarchical clustering using the mean rank correlation as similarity measure.

Figure 5.4 shows four groups of similar filter methods. The first group consists of 6 out of the 7 *praznik* filters: *NJMIM*, *DISR*, *MIM*, *JMIM*, *JMI*, and *CMIM*. The second group is formed by the filters *limma*, *anova.test*, *sam*, *kruskal.test*, and *auc*. The third group contains the filters *sym.uncert*, *gain.ratio*, *info.gain*, *chi.squared*, and *oneR* from the toolboxes *FSelector* and *FSelectorRcpp*. The fourth group consists of the random forest importance filters *permutation* and *impurity*. The filter methods *MRMR* and *variance* are not similar to any other filter method.

The average similarity value in Figure 5.4 is 0.5241. The highest mean rank correlations are observed between *info.gain* and *chi.squared* (0.9950), *kruskal.test* and *auc* (0.9931), *limma* and *anova.test* (0.9903), *sym.uncert* and *info.gain* (0.9892), *sym.uncert* and *gain.ratio* (0.9860), *sym.uncert* and *chi.squared* (0.9813), *limma* and *sam* (0.9798), *anova.test* and *sam* (0.9758), *gain.ratio* and *info.gain* (0.9618), and *gain.ratio* and *chi.squared* (0.9513).

The similarity of the filters *limma*, *anova.test*, *sam*, *kruskal.test*, and *auc* is easy to understand. The Kruskal-Wallis test can be seen as the non-parametric equivalent of the analysis of variance. The filters *limma* and *sam* perform a moderated version of the F test conducted by filter *anova.test*. Also, there are strong links between the area under the curve and the Kruskal-Wallis test (Hanley and McNeil, 1982).

Considering the group of similar *praznik* filters, it appears plausible that they select features in a similar order because their scores are all based on mutual information. Within this group, *JMIM* and *JMI* as well as *NJMIM* and *DISR* are especially similar to each other.

This makes sense as the scores are modified versions of each other, see Section 3.1. The filter *MRMR* comes from the same toolbox, but does not appear to be very similar to the other *praznik* filters. In contrast to the other *praznik* filters, *MRMR* also considers mutual information between features, see Section 3.1.

The filters *MIM* and *info.gain* use the same score. However, the different discretization methods used by the two toolboxes cause an average rank correlation of only 0.6272 between the two filter methods. The filters from the toolboxes *FSelector* and *FSelectorRcpp* use different filter criteria but all employ the same discretization method. This makes it seem likely that the similarity of these filter methods is due to the same discretization method.

All of the previous analyses are based on the averaged rank correlations. A brief analysis of the plots in Figures A.5 and A.6 in Appendix A.2 shows that the similarity structure displayed in Figure 5.4 does not represent the similarity structure for all single data sets. Instead, there are mainly two groups of data sets. In the first group, the similarity structure resembles the one in Figure 5.4. For the data sets *gravier*, *eating*, *arcene*, and *chiaretti*, the resemblance is very strong. The data sets *sonar*, *tecator*, *har*, and *lsvt* also show the group structure, but less clearly. The second group consists of the data sets *kc1-binary*, *gina_agnostic*, *dilbert*, and *christensen*. In the second group, most filter methods select the features in a similar order. There are only few filter methods that are not similar to the other methods, and these methods differ between the data sets of the second group. The data sets *gravier*, *eating*, *arcene*, and *chiaretti*, which show the filter group structure very clearly, are the four data sets that contain the most features. The other groups of data sets cannot be explained by the data set characteristics given in Table 4.1.

5.3 Optimal Filter Methods with Respect to Predictive Performance and Run Time

In the previous section, groups of similar filter methods have been discovered. For deciding which of the filter methods in the groups of similar filters can be neglected, all filter methods are compared with respect to their predictive performance and run time.

5.3.1 Experimental Setup

For determining the predictive performance of subsets of features selected by a filter method, the three classification methods support vector machine, k nearest neighbors, and ridge logistic regression are considered. Specifically these classification methods are chosen because they are popular methods that do not perform embedded feature selection. This is important for judging the direct impact of the feature selection conducted by the filter on the prediction performance, not in combination with a subsequent embedded feature selection by the classification algorithm.

Each filter method is combined with a classification method such that the combined methods first apply the filter, selecting a given percentage of features, and then learn the classification rule using only the remaining features. To ensure a fair comparison of the filter methods, the predictive performances of the filters are compared in combination with the best classifier and the best hyperparameters per filter. More precisely, the classification method to be combined with a filter method is considered as a hyperparameter as well, which results in a hierarchical search space. For each filter method and each data set, the classification method, the corresponding hyperparameters and the percentage of features to select are tuned simultaneously. Possible configurations of the search space are for example

(12%, KNN, $k = 7$) or (94%, SVM, $C = 2^{-1.3}$, $\sigma = 2^{13.28}$). Percentages of features to select are allowed in the range $[0\%, 100\%]$. The sets of considered values for the hyperparameters of the classification methods are given in Table 5.2. To find the best configuration, a random search with 100 iterations is conducted. To obtain unbiased estimates of the performances of the filters, nested cross-validation with 10 outer and 10 inner iterations is performed (Bischl et al., 2012). It is made sure that the considered configurations are identical for all filter methods on the same data set in the same outer iteration, in order not to favor any filter method because it happens to be evaluated in combination with better configurations. Also, all filter methods use the same cross-validation splits.

For each filter method and each data set, the following is done: In each outer iteration, 10% of the data set is used as evaluation data. On the remaining 90% of the data, 10-fold cross-validation is conducted to estimate the performance of 100 randomly drawn configurations. The best configuration is selected based on maximal mean classification accuracy. Ties are resolved by choosing the configuration with the smallest run time. Either the median time for training the combined model or the median time for filtering only are considered. For the analyses taking into account the time for filtering, the respective tuning results are used. For all other analyses, the tuning results with ties resolved by the times for training the combined models are employed. The median is used for aggregating the run times in order to obtain an estimate that is robust against variation caused by the high performance compute cluster. Note that this tuning procedure makes sense because in practice, one is interested in filter methods that allow a good predictive performance, and among these methods, one is interested in methods with short run times.

The selected configuration is then evaluated on the evaluation data, calculating the classification accuracy and measuring the time for training the combined model and for filtering. This way, for each data set and each filter method, 10 evaluations of the best configurations from the 10 outer iterations are obtained. The performance values are aggregated by calculating the mean classification accuracy and the median run times. So, in the end there are three performance values per filter method and data set: mean classification accuracy, median time for filtering, and median entire run time. Note that with the two strategies for resolving ties during tuning, different configurations may be chosen, so the accuracies on the test data may differ.

As an important baseline, the results of the filters are compared to results without filtering. For classification without filtering, a random search with the same cross-validation splits and configurations as above (ignoring the percentage of features to select) is conducted. The same performance measures are assessed and the best configurations are selected with respect to the same criteria.

There are 14 configurations for which no results could be obtained. As all filters try the same configurations, this means that for all filters the results for the corresponding

Method	Hyperparameters	R package
Support Vector Machine	$C, \sigma \in \{2^x : x \in [-15, 15]\}$	*kernlab* (Karatzoglou et al., 2004)
k Nearest Neighbors	$k \in \{1, 2, \ldots, 20\}$	*kknn* (Schliep and Hechenbichler, 2016)
Ridge Logistic Regression	$\lambda \in \{2^x : x \in [-15, 15]\}$	*glmnet* (Simon et al., 2011)

Table 5.2: Classification methods with corresponding hyperparameters and sets of values used for tuning as well as the R packages in which the methods are implemented.

configurations are missing. Of theses configurations, 7 are tried on data set *sonar*, 3 on *kc1-binary*, 3 on *tecator*, and 1 on *lsvt*. The failing of all configurations is caused by the selection of less than two features by the filters and the implementation of ridge logistic regression in *glmnet*, which requires at least two features. The configurations for which no results were obtained are ignored for the analyses.

5.3.2 Results

First, the filter methods are compared only with respect to predictive performance. Figure 5.5 shows for two data sets the classification accuracies of the 10 best configurations corresponding to the 10 outer cross-validation iterations, separately for all filter methods. As the performances are evaluated on data that is not used for tuning, the performance values may be interpreted as unbiased estimates of the performances on new data from the data generating process that created the respective data set. The results for the other 10 data sets are displayed in Figures A.7 and A.8 in Appendix A.2.

The most obvious observation from all plots is that there are large differences in predictive performance for the same filter method across cross-validation iterations, compared to the differences between the filter methods. For some data sets, there are noticeable differences between the filter methods, for others there are not. Figure 5.5 displays an example for each situation. The left plot in Figure 5.5 shows that for data set *gina_agnostic*, some filters perform considerably better than others. The filter *impurity* performs quite well while the filters *DISR*, *MRMR*, and *variance* perform comparably bad. All filters lead on median to a better predictive performance than "no filter". The right plot in Figure 5.5 demonstrates that for data set *sonar*, there are only little differences in the central locations of the classification accuracies, taking into account the variations. These variations should be kept in mind, when the predictive performance of the filter methods will be expressed by the mean classification accuracy across cross-validation iterations in the following analyses.

Figure 5.6 shows the number of times that the filter methods outperform each other based on the mean accuracies. The number displayed in the row of filter A and the column of filter B indicates the number of data sets on which filter A is better than filter B with respect to the mean classification accuracy. Given that two filter methods are equally good but never have exactly the same performance, it is expected that each of the filters outperforms the

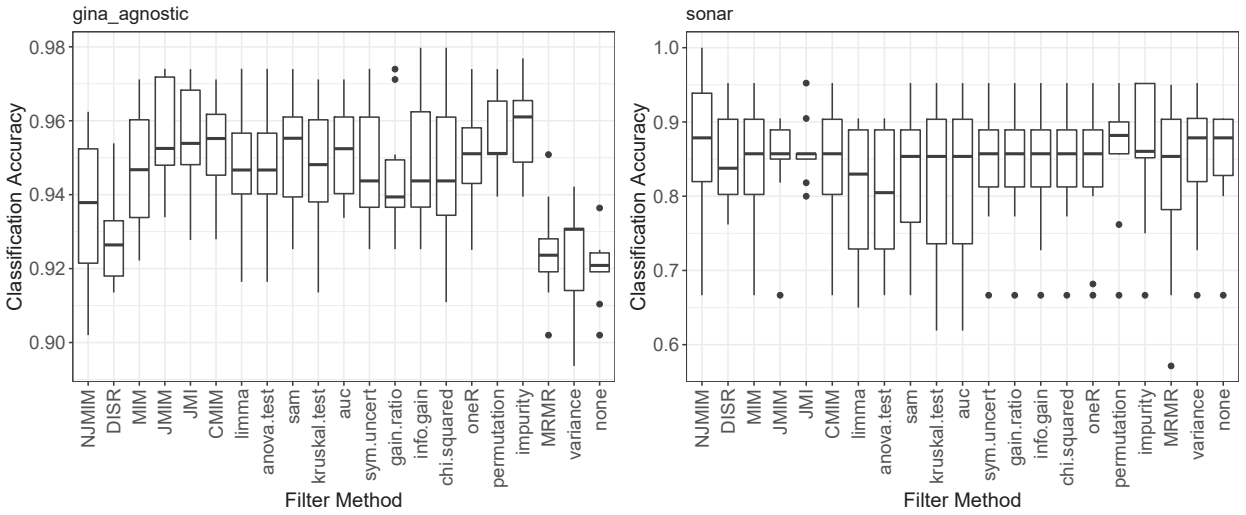


Figure 5.5: Boxplots of the classification accuracies of the best configurations in the 10 outer cross-validation iterations per filter method and data set.

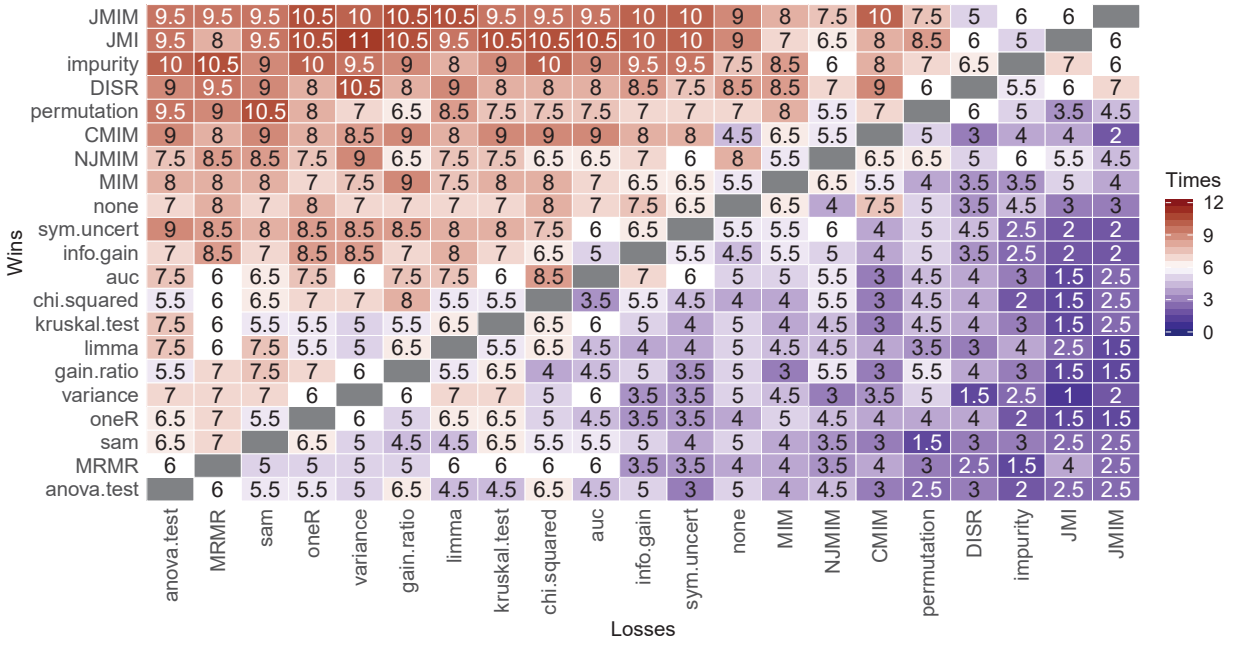


Figure 5.6: Number of data sets on which the filter method in the row has a higher mean classification accuracy than the filter method in the column. Ties are counted as 0.5 for both filters. The filter methods are sorted decreasingly by row sums.

other on approximately 6 of the 12 data sets. It is marked in red when a filter method is better than another filter method more often than 6 times and in blue when this happens less than 6 times. Note that there are several ties with respect to the classification accuracy of the filter methods. It can be observed that filters *JMIM*, *JMI*, *impurity*, and *DISR* perform best across data sets. However, there is no filter method that is better than all the other methods on all data sets. The filters *anova.test*, *MRMR*, *sam*, *oneR*, *variance*, *gain.ratio*, *limma*, *kruskal.test*, and *chi.squared* are outperformed by many other filter methods.

Now, the filter methods are compared with respect to both predictive performance and run time across data sets. For the analysis presented in Figure 5.6, it was only looked at good or bad predictive performances in comparison to the other filter methods. This analysis also takes into account how much the filter methods differ in performances. In the following, first, the run time for filtering only and later the entire run time for filtering and fitting the best classification model are considered.

Figure A.9 in Appendix A.2 displays the mean classification accuracies and the median run times for filtering, separately for all data sets. Figure 5.7 aggregates the information of Figure A.9 into one graphic. To compare the predictive accuracy across data sets, relative accuracies are considered. More precisely, the difference between the mean classification accuracy of each filter method and the highest mean classification accuracy observed on the same data set is calculated. The filter method with the best predictive performance per data set has relative mean classification accuracy 0. The smaller the relative classification accuracy, the better the predictive performance of the filter method when combined with the best configuration. Because some very long run times are observed (see Figure A.9), logarithmic run times are considered. To be able to apply the logarithm, a small constant is added to all run times, as not filtering at all has zero run time. The lowest logarithmic median run time for filtering per data set is subtracted from all logarithmic median run times for filtering on the same data set. So, the lowest relative logarithmic median run time for filtering per

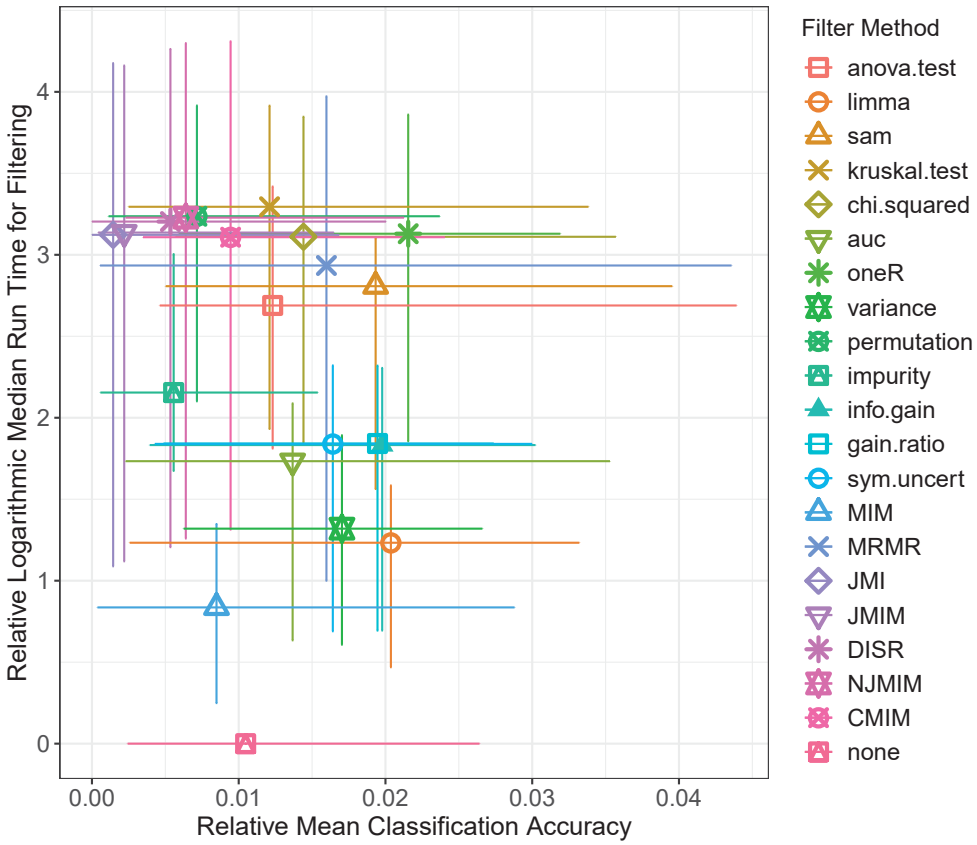


Figure 5.7: Relative mean classification accuracy and relative logarithmic median run time for filtering of the filter methods with best configurations aggregated over all 12 data sets. The median of both performance measures (relative mean classification accuracy and relative logarithmic median filtering time) across all data sets is displayed by a symbol. The upper and lower quartiles are located at the respective ends of the horizontal and vertical lines. The best filter method would be located in $(0, 0)^\top$.

data set is 0. Regarding the interpretation of the transformed performance criteria, a relative mean classification accuracy of x means that the mean classification accuracy of the filter (with the best configuration found) is worse than the mean classification accuracy of the best filter on the same data set by an additive factor of x. A relative logarithmic median run time of x means that the median run time of the filter equals the median run time of the best filter on the same data set multiplied with 10^x. The transformed performance criteria are aggregated over the data sets by computing the median values as well as the upper and lower quartiles. The median value provides information about the central location of the performance measures. The distance of the quartiles shows the variation of the performances across data sets. Figure 5.7 displays for each filter method the median of the performance criteria values by a symbol. The quartiles are located at the respective ends of the horizontal and vertical lines. The longer the lines, the larger the variation across data sets. The best filter method would be located in $(0, 0)^\top$.

Figure 5.7 shows that the relative mean classification accuracy and the relative logarithmic median run time for filtering vary a lot across data sets for all filter methods. Compared to the differences between the filter methods, the variations across data sets are quite large. Considering the median values in both criteria, it can be seen that the filters *JMI*, *impurity*, and *MIM* as well as "no filter" are Pareto optimal. Pareto optimality means that there is

no other method that performs as good in both criteria and better in at least one criterion. Considering the non-Pareto optimal filter methods in Figure 5.7, at least one of the Pareto optimal filters performs better in both criteria. Filter *JMI* is Pareto optimal because it has the best median relative classification accuracy among all filter methods. Filter methods *impurity* and *MIM* have a good median relative classification accuracy and a low median relative run time for filtering. Applying no filter takes no time, which is always faster than applying any filter and therefore this is always Pareto optimal, independent of its classification accuracy. If instead of the median performance values the upper quartiles are considered, only filter *impurity* and "no filter" are Pareto optimal. It should be noted that the configurations have been chosen primarily based on classification accuracy. Because the run times of the filters from package *praznik* depend on the number of selected features (see Section 5.1), it might be possible to make them Pareto optimal by selecting fewer features. This would make them faster at the cost of predictive accuracy. As the run times depend on the implementation, any filter method could potentially become Pareto optimal if it was implemented efficiently enough.

Now, the predictive accuracy is analyzed in combination with the entire run time for filtering and fitting the best classification model. Considering the entire run time makes sense because it also assesses how long it takes to fit the best classification model with the features selected by the filter. As most filters rank all features and because there are groups of filters with very similar run times (see Section 5.1), it is likely that many filters achieve similar run times even if they select different numbers of features. However, selecting more features may have a huge impact on the run time for fitting the predictive model of interest.

Figure A.10 in Appendix A.2 shows the mean classification accuracies and the median run times for fitting the combined models for all data sets. Figure 5.8 aggregates the information of Figure A.10, resulting in an analogous plot to Figure 5.7. In comparison to Figure 5.7, for example the relative run times for *auc* and *sam* are lower, compared to the other filter methods. For these filters, the filtering takes comparably long but the best predictive model using the selected features can be fitted comparably fast. When considering the median values of the performance measures, the filter methods *JMIM*, *JMI*, *impurity*, and *MIM* are Pareto optimal. Compared to the Pareto set resulting from Figure 5.7, "no filter" is replaced by *JMIM* and the rest is left unchanged. With respect to the upper quartiles of the performance measures, here *JMI*, *impurity*, and *MIM* are Pareto optimal. Compared to the respective Pareto set from Figure 5.7, "no filter" is replaced by *JMI* and *MIM*. Like in the discussion of Figure 5.7, it also has to be mentioned that filter methods could become Pareto optimal by sacrificing classification accuracy and thus saving run time. Here, this is possible for all filter methods by choosing a configuration that is faster but provides less predictive accuracy.

5.4 Stochasticity of Filter Scores

The scores of the filter methods *impurity* and *permutation* are stochastic. For both filters, random forests are constructed which is a process subject to randomness, see Section 3.1. For *permutation*, the feature importance values are determined based on random permutations which adds even more stochasticity. The other filter methods are deterministic.

For *impurity* and *permutation*, the number of trees has an influence on how much the filter scores vary when the same filter is applied several times to exactly the same data set. Figure 5.9 and Figure A.11 in Appendix A.2 show the variation of the filter scores for random forests with different numbers of trees. To generate the plots, the filter methods *impurity* and *permutation* are applied 10 times to each data set. Then, for each feature, the ranks of the

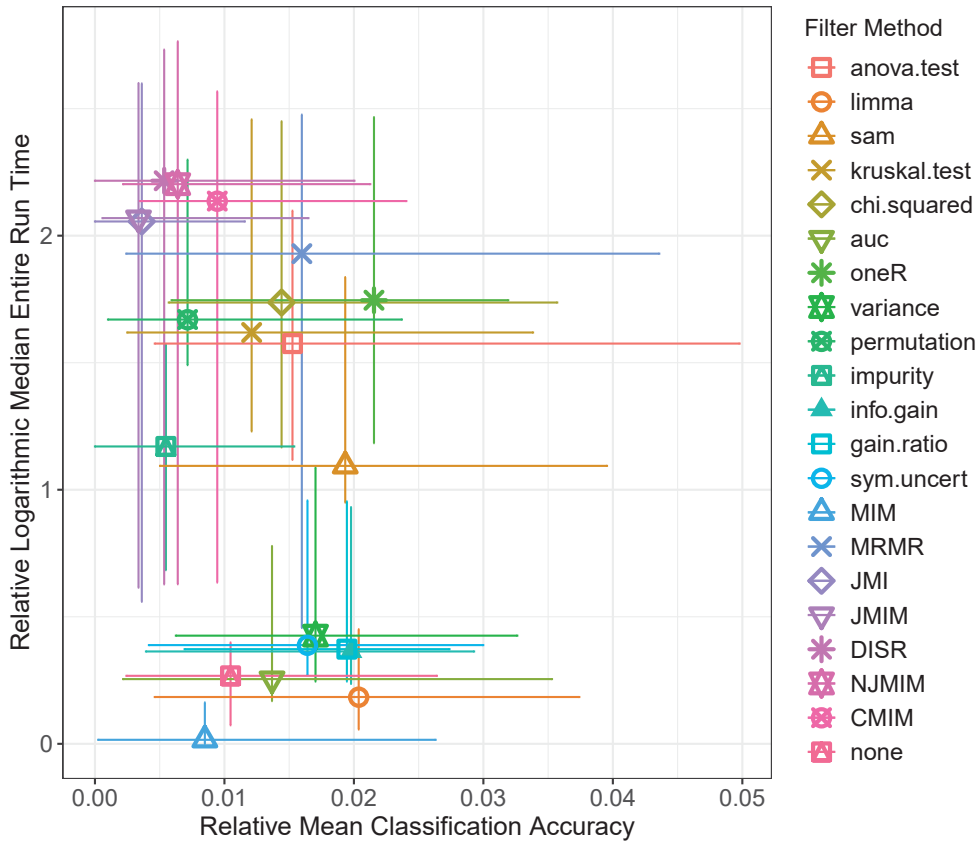


Figure 5.8: Relative mean classification accuracy and relative logarithmic median run time for filtering and model fitting of the filter methods with best configurations aggregated over all 12 data sets. The median of both performance measures (relative mean classification accuracy and relative logarithmic median run time) across all data sets is displayed by a symbol. The upper and lower quartile are located at the respective ends of the horizontal and vertical lines. The best filter method would be located in $(0, 0)^\top$.

filter scores are assessed and their variation is quantified by calculating the interquartile range. Finally, for each data set, the interquartile ranges of the ranks are sorted increasingly. So, the resulting functions can be interpreted as quantile functions of the variations in the feature ranks. For most data sets, the variation in the filter scores decreases with an increasing number of trees in the forest. Based on these results, it seems advisable to use a large number of trees. However, no clear conclusion can be drawn on how many trees are enough.

Figure A.12 in Appendix A.2 shows the run times for calculating the filter scores for random forests with different numbers of trees. The run times of the 10 replications are aggregated using the median. The increase in run time with respect to the number of trees in the forest is linear for both *impurity* and *permutation*. Especially for filter *permutation*, the run times become unbearably long for large numbers of trees, compared to the benefit in reduced variation. In the previous sections, *impurity* and *permutation* have been used with 500 trees which provides a good compromise between variability and run time. Also, this is the default in the software implementation.

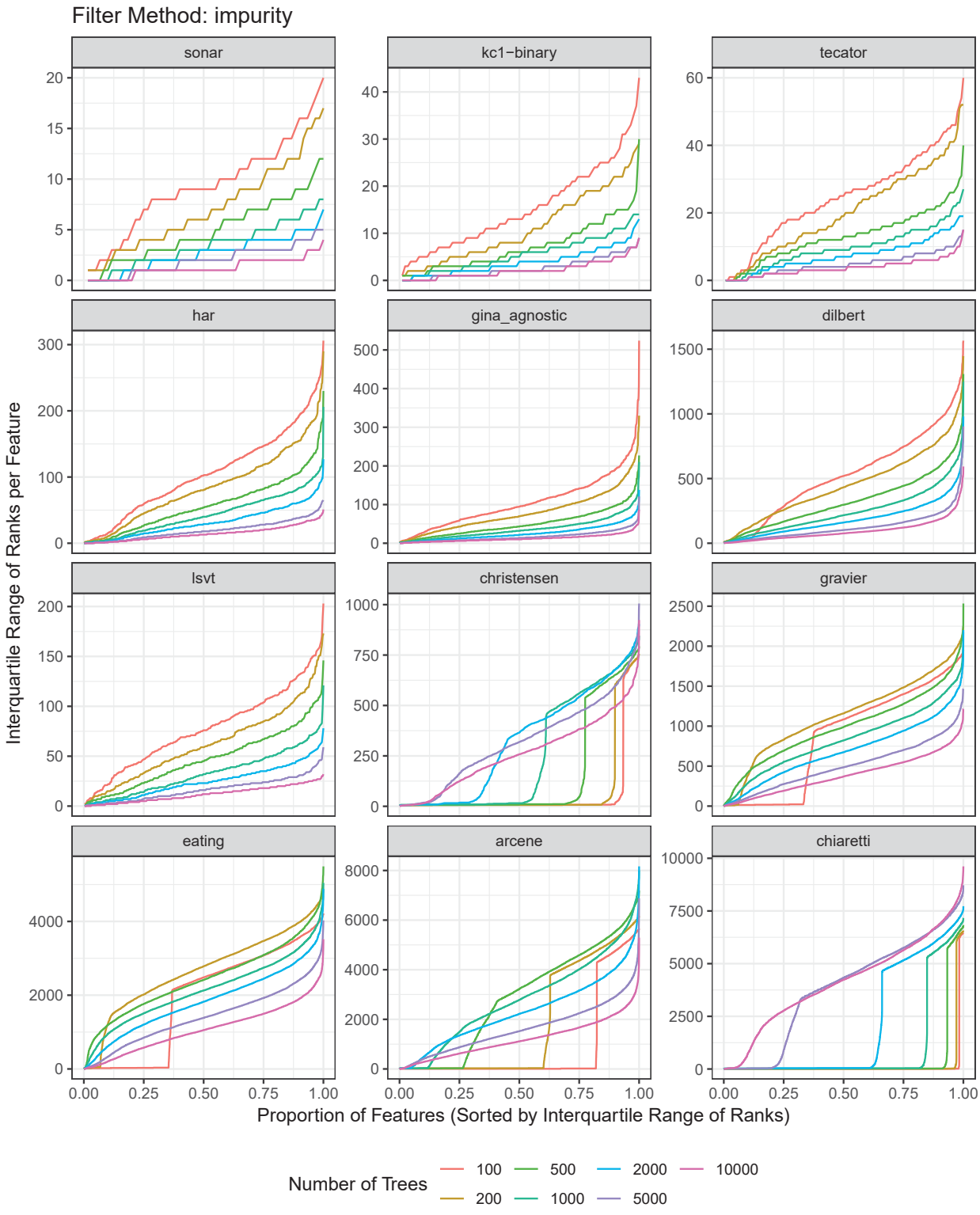


Figure 5.9: Interquartile ranges of the ranks of the scores calculated with filter *impurity* for each feature. The features are sorted increasingly by the interquartile range of their ranks.

5.5 Conclusions

Based on empirical analyses, the filter methods can be divided into groups of filter methods that rank the features similarly. One group consists of the *praznik* filters *NJMIM*, *DISR*, *MIM*, *JMIM*, *JMI*, and *CMIM*. Another group is formed by the filters *limma*, *anova.test*, *sam*, *kruskal.test*, and *auc*, which are all related to the analysis of variance. The next group contains the filters *sym.uncert*, *gain.ratio*, *info.gain*, *chi.squared*, and *oneR* from the toolboxes *FSelector* and *FSelectorRcpp*. The last group consists of the random forest importance filters *permutation* and *impurity*. The filter methods *MRMR* and *variance* are not similar to any other filter method.

With respect to the scaling behavior, the filters *NJMIM*, *DISR*, *JMIM*, *JMI*, *CMIM*, and *MRMR* deviate from all other filters. For these filters, the filter scores are calculated in an iterative way, whereas for all other methods, the scores of all features are calculated at once.

There is no subset of filter methods that outperforms all other filter methods with respect to predictive accuracy. Which filters work best, depends on the data set. The results are subject to variation within and across data sets. Nevertheless, the filter methods *JMI*, *MIM*, and *impurity* work well in most data situations.

Regarding the question whether some of the filter methods can be neglected when searching for a good filter method, it seems reasonable to limit the search space to *JMI*, *MIM*, and *impurity*. *JMI* and *MIM* are representatives of the group of *praznik* filters. *JMI* selects the features in an iterative way, while *MIM* calculates all scores at once. Filter *impurity* is a representative of the group of random forest filters. One could add *auc* and *sym.uncert* to the search space as representatives of their groups of filter methods, because they are the ones that perform best within these groups with respect to predictive accuracy. Filters *MRMR* and *variance* can be neglected due to bad predictive performance.

Choosing the best filter method for a new data set also is a matter of available computational resources. Based on the analyses, the following recommendations can be made: If the computational resources only allow trying one filter method, *impurity* is a good choice. This filter method allowed fitting classification models with high classification accuracy on most of the data sets. If some computational resources are available for finding a suitable filter method, *JMI*, *MIM*, and *impurity* should be tried. If the resources allow trying all filter methods, this is recommended, because only this way the very best filter method for a new data set can be found.

Chapter 6

Comparison of Stability Measures

The 20 stability measures presented in Subsection 3.3.2 are compared both theoretically and empirically in this chapter. The goal of the analyses is finding the best measures for stability assessment. In Section 6.1, a set of desirable theoretical properties is given and it is analyzed which of these properties are fulfilled by each measure. In the empirical comparison, the empirical behavior of the stability measures is investigated. One aim of the comparison is finding groups of similar stability measures. For this, it is analyzed, whether the stability measures consider the same situations as stable or unstable. Another aim of the analyses is investigating the impact of the number of selected features on the stability measures. In Section 6.2, the stability measures are compared based on a small example for which all combinations of feature sets can be taken into account. In Section 6.3, the comparison of the stability measures is conducted based on feature sets selected for real data sets.

The analyses in this chapter extend the analyses in Bommert et al. (2017) and Bommert and Rahnenführer (2020). In Bommert et al. (2017), only 9 stability measures are considered. Here, all of these measures and 11 additional stability measures are investigated. The analyses in Bommert et al. (2017) are based on feature sets resulting from 3 gene expression data sets. In this chapter, both artificial feature sets and real feature sets obtained for the 12 data sets presented in Section 4.1 are used. For the 9 stability measures that have been analyzed in Bommert et al. (2017) as well, similar results are obtained here. In Bommert et al. (2017), theoretical properties of the stability measures are stated and motivated, but no extensive proofs are given. Here, all proofs of properties are given in Section 6.1 or Appendix B. In Bommert and Rahnenführer (2020), 6 of the adjusted stability measures presented in Subsection 3.3.2.3 are compared to each other and to an unadjusted stability measure. The comparisons are based on the artificial feature sets introduced in Section 6.2 as well as on a subset of the feature sets considered in Section 6.3. The analyses in Bommert and Rahnenführer (2020) are a subset of the analyses in this thesis. Here, more stability measures and data sets are considered and more analyses are conducted.

6.1 Theoretical Comparison

In this section, the stability measures are compared with respect to theoretical properties. For this, the notation of Subsection 3.3.2 is used. Nogueira (2018) defines five properties that are desirable for stability measures:

1. Fully defined: The stability measure should not require the cardinalities $|V_1|, \ldots, |V_m|$ to be identical.

2. Monotonicity: The stability measure should be a strictly decreasing function of the sample variance s_j^2 of the selection of each feature. Nogueira (2018) shows that this

property is equivalent to asking for the stability measure to be a strictly increasing function of all cardinalities of pairwise intersections $|V_i \cap V_j|$.

3. Correction for chance: The expected value of the stability measure for a random feature selection with equal selection probabilities should be constant and therefore not depend on the number of selected features.

4. Bounds: The stability measure should have an upper and a lower bound. Both bounds should be finite and neither depend on p nor on the number of selected features.

5. Maximum: Maximum stability $\Leftrightarrow$ deterministic selection: The stability measure should attain its maximum value if and only if $V_1 = \ldots = V_m$.

The idea of the monotonicity property is to assess whether a stability measure takes larger values in situations that are intuitively considered to be "more stable". The properties 3 to 5 serve the assessment whether the stability values can be interpreted well. In addition to the five properties stated by Nogueira (2018), we define three further properties:

6. Adjusted: The stability measure takes into account similarities between the features, such that different but highly similar features count towards stability.

7. Adjusted monotonicity: The stability measure should be a strictly increasing function of all cardinalities of pairwise adjusted intersections $|V_i \cap V_j| + \text{Adj}(V_i, V_j)$. The adjustment depends on the stability measure and for unadjusted stability measures, $\text{Adj}(V_i, V_j) \equiv 0$.

8. Maximum for equal cardinalities: The maximum value should only be attainable in situations where all feature sets have the same cardinality, that is, $|V_1| = \ldots = |V_m|$. Also, there should exist situations with $|V_1| = \ldots = |V_m|$ in which the stability measure attains its maximum value.

For adjusted stability measures, it is not desirable to fulfill the monotonicity or the maximum property. These measures are specifically designed so that similar features are viewed as exchangeable for stability assessment. Therefore, the measures should not only increase when the cardinality of a pairwise intersection increases and the maximum value should not only be attainable for identical sets. The "adjusted monotonicity" property extends the monotonicity property such that similar features are considered. The "maximum for equal cardinalities" property is a relaxation of the maximum property. Note that for all properties, only such feature sets and similarities between the features are considered for which the stability measures are well-defined. The necessary conditions for each stability measure to be well-defined are stated in Appendix B.1.

Table 6.1 shows for the 20 stability measures if they fulfill the 8 properties. For SMJ, SMD, SMO, SMH, SML, SMW, SMP, SMD-α, SMS, and SMN, the properties 1 to 5 are analyzed in Nogueira (2018) or Nogueira and Brown (2016). Two proofs given in Nogueira (2018) are incorrect. They are based on the statement that SMD-α with $\alpha = 0$ would be equal to another stability measure not considered in this thesis, but this statement is wrong. Therefore, proofs for the two properties of SMD-α are given in this thesis. The properties 1 to 5 for the stability measures SMU, SMK, SME, SMZ, SMES, SMY, and SMA as well as the properties 6 to 8 for all measures are discussed in this thesis. The results for all measures are reported in Table 6.1. Because the four considered variants of SMA (SMA-MBM, SMA-Greedy, SMA-Count, and SMA-Mean) do not differ with respect to these properties, they are analyzed jointly as SMA.

Name	Reference	Fully defined	Monotonicity	Correction	Bounds	Maximum	Adjusted	Adjusted monotonicity	Maximum for equal cardinalities
SMJ	Jaccard (1901)	×	×	—	×	×	—	×	×
SMD	Dice (1945)	×	×	—	×	×	—	×	×
SMO	Ochiai (1957)	×	×	—	×	×	—	×	×
SMH	Dunne et al. (2002)	×	×	—	×	×	—	×	×
SML	Lustgarten et al. (2009)	×	×	×	×	—	—	×	—
SMW	Wald et al. (2013)	×	×	×	×	—	—	×	—
SMU	Bommert and Rahnenführer (2020)	×	×	×	×	×	—	×	×
SMK	Carletta (1996)	×	×	×	×	×	—	×	×
SMP	Nogueira and Brown (2016)	×	×	×	×	×	—	×	×
SME	Novovičová et al. (2009)	×	×	—	×	×	—	×	×
SMD-0	Davis et al. (2006)	×	—	—	×	×	—	—	×
SMD-α, $\alpha > 0$	Davis et al. (2006)	×	—	—	×	—	—	—	—
SMS	Somol and Novovičová (2008)	×	×	—	×	—	—	×	—
SMN	Nogueira (2018)	×	×	×	×	×	—	×	×
SMZ	Zucknick et al. (2008)	×	—	—	×	—	×	×	×
SMES	Sechidis et al. (2020)	×	—	—	—	—	×	×	—
SMY	Yu et al. (2012)	×	—	×	—	—	×	×	—
SMA	Bommert and Rahnenführer (2020)	×	—	×	—	—	×	×	×

Table 6.1: Properties of the considered stability measures.

Fully Defined The stability measures SMU, SMK, SME, SMZ, SMY, and SMA are fully defined. Only stability measures that are fully defined are introduced in Subsection 3.3.2.

Monotonicity The monotonicity property holds for SMU and SMK. These measures are defined as averages of normalized versions of $|V_i \cap V_j|$ such that they are strictly increasing functions of $|V_i \cap V_j|$, see Subsection 3.3.2.1. For SME, the monotonicity property is fulfilled as well: $h_j \log_2(h_j)$ for $h_j \in \{1, \ldots, m\}$ is a convex function with minimum in $h_j = \frac{m}{2}$ and s_j^2 is a concave function of h_j with maximum in $h_j = \frac{m}{2}$. So, if s_j^2 increases, the value of SME decreases. In Appendix B.2, a proof for SMD-α not fulfilling the monotonicity property is given. The adjusted measures SMZ, SMES, SMY, and SMA do not fulfill the monotonicity property, see Appendix B.2.

Correction for Chance SMU, SMK, SMY, and SMA are corrected for chance. For these measures, in the pairwise scores, the expected value of $|V_i \cap V_j|$ (or $|V_i \cap V_j|$ + adjustment, respectively) for a random feature selection with equal selection probabilities is subtracted in the numerator. Because of this, the expected values of all pairwise scores are 0 and consequently, the expected values of the stability measures are 0, too. This holds independent of the number of selected features. SMZ is not corrected for chance. The more features are selected, the higher the expected value of SMZ. In the special case where there are no highly correlated features in $\bigcup_{i=1}^{m} V_i$, SMZ and SMJ are identical and SMJ is not corrected for chance (Nogueira, 2018). SME and SMES are not corrected for chance either. The expected values of SME and SMES for a random feature selection depend on the number of selected features, see Appendix B.3. Regarding the stability measure SMS, Nogueira (2018) shows that SMS is not corrected for chance, but it is asymptotically corrected for chance for $m \to \infty$.

Bounds Table 6.2 displays the ranges of possible values for all stability measures considered in this thesis. Before the bounds property is discussed, first, we analyze the upper and lower bounds and their tightness for all stability measures. Proofs are given in Appendix B.1. For all measures except SMES, the maximum value is 1 and it is attained if all sets $V_1, \ldots, V_m$ are identical, independent of the value of m or the number of selected features. For SML, the upper bound is only attained asymptotically for $p \to \infty$. For SMD-α, the upper bound is only tight for $\alpha = 0$. For SMES, no upper bound that is independent of p and the data specific similarity structure can be given, see Appendix B.1. This is a problem with respect to interpretability, because it cannot be known if a stability value obtained with SMES indicates rather high or rather low stability.

All lower bounds are tight if $m = 2$. The lower bound of SML is only tight asymptotically for $p \to \infty$. For SMD-α, the lower bound can only be attained if α is large enough, see Appendix B.1. For the measures SML, SMW, SMU, SMK, and SMP, tight lower bounds are not known for $m > 2$. This, however, is not a problem with respect to the interpretability of the stability values. For these measures, a value of 0 means that a feature selection algorithm is as stable as a random feature selection and lower values indicate an even worse stability.

For SMY and SMA, universal lower bounds cannot be given, as they depend on the data specific similarities between the features. The more similar features there are, the closer are the expected values to the maximum values and therefore, the smaller is the minimum stability value. For the same reasons as for SML, SMW, SMU, SMK, and SMP, this is not a problem with respect to the interpretability of the stability values. For SMES, no lower bound is known either. The value that SMES attains for a random feature selection is not constant. It depends on the number of selected features, see the discussion of the "correction

Name	Reference	Lower Bound	Upper Bound	Tightness of Lower Bound for $m > 2$
SMJ	Jaccard (1901)	0	1	yes
SMD	Dice (1945)	0	1	yes
SMO	Ochiai (1957)	0	1	yes
SMH	Dunne et al. (2002)	0	1	no
SML	Lustgarten et al. (2009)	-1	1	no
SMW	Wald et al. (2013)	$1-p$	1	no
SMU	Bommert and Rahnenführer (2020)	-1	1	no
SMK	Carletta (1996)	-1	1	no
SMP	Nogueira and Brown (2016)	-1	1	no
SME	Novovičová et al. (2009)	0	1	yes
SMD-α	Davis et al. (2006)	0	1	depends on α
SMS	Somol and Novovičová (2008)	0	1	yes
SMN	Nogueira (2018)	$\frac{-1}{m-1}$	1	yes
SMZ	Zucknick et al. (2008)	0	1	yes
SMES	Sechidis et al. (2020)	—	—	—
SMY	Yu et al. (2012)	—	1	—
SMA	Bommert and Rahnenführer (2020)	—	1	—

Table 6.2: Ranges of the stability measures.

for chance" property. Therefore, for SMES, the lack of a lower bound is a problem with respect to interpretability.

Regarding the bounds property, SMU, SMK, SME, and SMZ fulfill this property because they have finite upper and lower bounds. For SMES, SMY, and SMA, the bounds property does not hold because for them, universal lower bounds are not known. For SMES, a finite upper bound is not known either.

Maximum For SMU, SMK, and SME, the maximum value is reached if and only if $V_1 = \ldots = V_m$, see Appendix B.4. For $\alpha = 0$, SMD-α fulfills the maximum property as shown in Appendix B.4. For $\alpha > 0$, SMD-α does not fulfill this property, because the maximum value is not attained for identical feature sets, see Appendix B.1. For SMZ, SMY, and SMA, the maximum stability value can also be obtained for sets that are not identical, see Appendix B.2. For SMES, the maximum property does not hold because for SMES, no maximum value is known.

Adjusted Out of the 20 stability measures, only SMZ, SMES, SMY, and SMA are adjusted. The others do not take into account similarities between features.

Adjusted Monotonicity The "adjusted monotonicity" property is equivalent to the monotonicity property for unadjusted stability measures. Therefore, the unadjusted measures fulfill the "adjusted monotonicity" property if and only if they fulfill the monotonicity property. For the adjusted measures SMZ, SMY, and SMA, the "adjusted monotonicity" property holds because these measures are defined as averages of pairwise scores such that each pairwise score is a strictly increasing function of the cardinality of the respective adjusted intersection, see Subsection 3.3.2.3. For SMES, the property holds as well. Sechidis et al. (2020) show that their stability measure can be rewritten as a function of all pairwise adjusted intersections

and this function is strictly increasing with respect to the cardinalities of all pairwise adjusted intersections.

Maximum for Equal Cardinalities The "maximum for equal cardinalities" property is an implication of the maximum property. If a stability measure attains its maximum value only for identical feature sets, then it only attains this value for sets with equal cardinalities. And if a stability measure attains its maximum value for identical feature sets, then the maximum value is attainable in a situation with all feature sets having the same cardinality. For SML, the "maximum for equal cardinalities" property is not fulfilled, because SML does not attain its maximum value for finite values of p, see Appendix B.1. SMW, SMS, and SMY do not fulfill this property because they can attain their maximum values for feature sets with unequal cardinalities, see Appendix B.5. SMD-α fulfills the "maximum for equal cardinalities" property for $\alpha = 0$ and not for $\alpha > 0$ due to the same reasons that the maximum property is fulfilled or not fulfilled, respectively. SMZ fulfills the property because it attains its maximum value only in situations in which all feature sets have the same cardinality and it attains its maximum value if all sets are identical, see the discussion of the maximum property for SMZ. For SMA, the property is fulfilled as well, as shown in Appendix B.5. SMES does not fulfill the property because no maximum value is known.

Conclusion For the unadjusted stability measures, it is desirable to fulfill all of the properties defined by Nogueira (2018). Only SMU, SMK, SMP, and SMN possess all properties. For the adjusted stability measures, it is desirable to fulfill the "maximum for equal cardinalities" property instead of the maximum property. Among the adjusted stability measures, SMZ and the four variants of SMA possess the largest number of desirable properties. However, none of them fulfills all desirable properties. SMZ is not corrected for chance and for SMA, no universal lower bound is known.

6.2 Empirical Comparison Considering All Combinations of Feature Sets

To compare the stability measures empirically, first, a comparison in a situation with only 7 features is conducted. The advantage of this comparison is that all possible combinations of 2 sets of selected features can be analyzed, because there are only $2^7 \cdot 2^7 = 16\,384$ possible combinations. Also, many of these 16 384 combinations are symmetric versions of each other, that is, they describe the same situation but the features in the two sets have different identifiers.

6.2.1 Experimental Setup

The values of the 20 stability measures presented in Subsection 3.3.2 are calculated for all 16 384 possible combinations of 2 feature sets being selected from a total number of 7 features. With the notation of Subsection 3.3.2, this means that $p = 7$ and $m = 2$. For SMD-α, the value $\alpha = 0$ is used. For the adjusted stability measures presented in Subsection 3.3.2.3, the similarities between the 7 features have to be defined. The similarity matrix used for this analysis is displayed in Figure 6.1. For the threshold θ, the value $\theta = 0.9$ is used. There is one block of three features that are similar to each other, that is, their similarity values exceed the threshold θ. Also, there are two blocks with two similar features each. For SMZ, the similarity matrix in Figure 6.1 is assumed to be a matrix of absolute Pearson correlations.

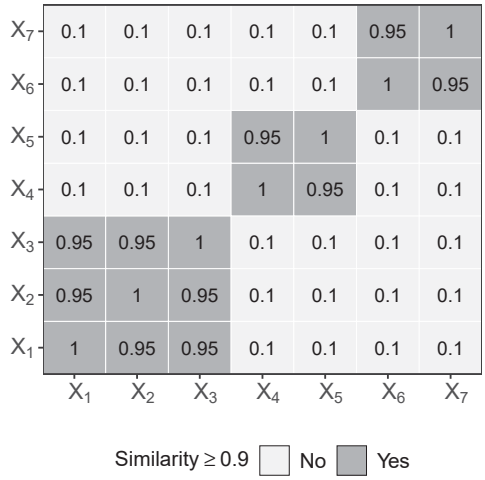


Figure 6.1: Similarity matrix for the 7 features. Similarity values must be within the interval $[0, 1]$.

For SMES, the similarity matrix C (see Subsection 3.3.2.3) is determined by setting all similarity values in Figure 6.1 that are smaller than the threshold $\theta = 0.9$ to 0. For SMY, SMA-MBM, SMA-Greedy, SMA-Count, and SMA-Mean, the expected values of the pairwise scores are determined exactly by considering all possible pairs of sets of the same cardinalities. Some stability measures are not defined for some of the 16 384 combinations of feature sets. Table 6.3 displays the number of scenarios with missing values for each stability measure. The situations in which each stability measure is not defined are stated in Appendix B.1. Missing values are omitted for the analyses.

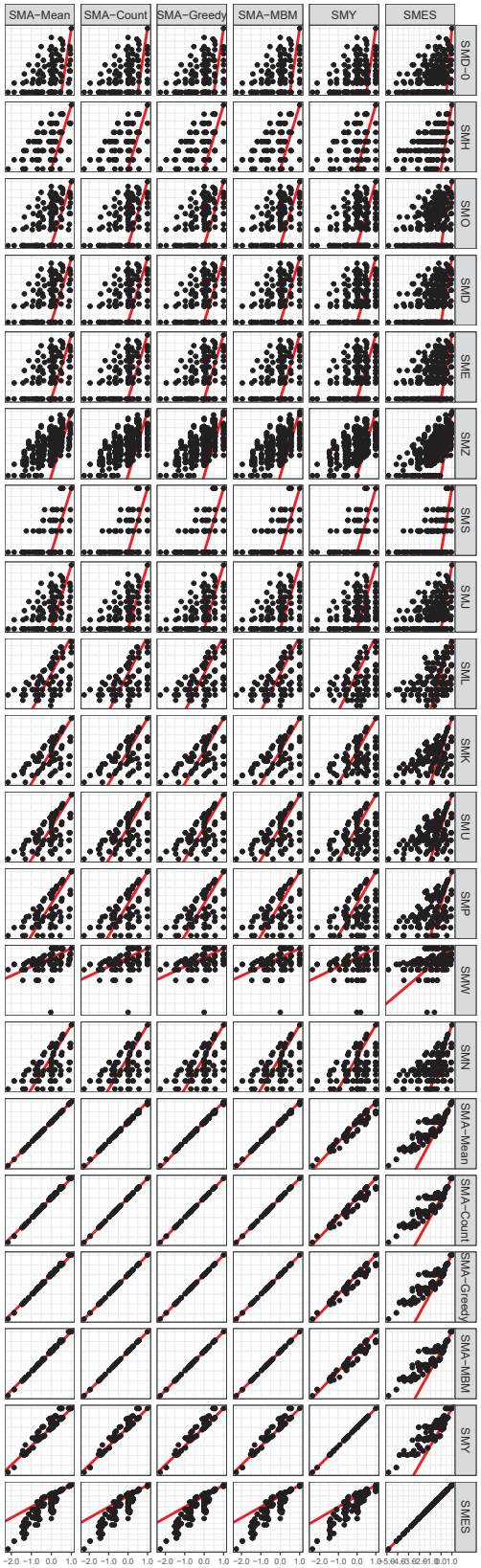
6.2.2 Results

To compare all stability measures, in Figure 6.2 scatter plots of all pairs of stability measures are shown. This plot allows determining three groups of stability measures. In the upper left corner of Figure 6.2, there is the group of stability measures SMD-0, SMH, SMO, SMD, SME, SMZ, SMS, and SMJ. All of these measures are not corrected for chance, see Section 6.1. The second group consists of the stability measures SML, SMK, SMU, SMP, SMW, and SMN. These measures are corrected for chance, but do not consider similarities between features. The third group consists of SMA-Mean, SMA-Count, SMA-Greedy, SMA-MBM, SMY, and SMES. These measures are adjusted and – except for SMES – they are corrected for chance.

Stability measure(s)	Number of scenarios with missing values
SMH	0
SMJ, SMD, SME, SMD-0, SMZ	1
SMK, SMN, SMES, SMY	2
SMS	30
SMO	255
SMU, SMA-MBM, SMA-Greedy, SMA-Count, SMA-Mean	256
SML, SMW, SMP	508

Table 6.3: Number of scenarios with missing values out of all 16 384 scenarios.

Figure 6.2: Scatter plots of the stability values obtained with the 19 stability measures for the 16 384 combinations. The red line in each plot indicates the identity. The order of the stability measures is determined by conducting a PCA with the stability measures as observations and all stability values as features, and ordering the measures by the values of the first principal component.


SMA−Mean
SMA−Count
SMA−Greedy
SMA−MBM
SMY
SMES
SMD−0
SMH
SMO
SMD
SME
SMZ
SMS
SMJ
SML
SMK
SMU
SMP
SMW
SMN
SMA−Mean
SMA−Count
SMA−Greedy
SMA−MBM
SMY
SMES

All groups are rather homogeneous. In the first group, especially SMO, SMD, and SME take almost identical values. SMD-0 and SMJ agree very much with each other and with SMO, SMD, and SME when considering the ranks of the stability values. SMH takes 8 and SMS only 5 distinct values for the 16 384 combinations. For SMH, there are many combinations in which SMH takes a value greater than 0 while the other measures of the group take the value 0. For these combinations, there are no features that are included in both sets, but there are features that are included in neither of the sets. For SMS, there are many combinations in which SMS assigns the value 0 and the other measures of the group take values larger than 0. This effect is due to the attempt of a correction for chance that SMS includes. SMZ is very different from all the other stability measures. The largest agreement between SMZ and any other measure can be found between SMZ and SMJ. The values of SMZ are always larger than or equal to the values of SMJ. This can be explained by SMZ being a modification of SMJ for taking into account similar features.

In the second group, SMK, SMU, and SMP are almost identical. SML and SMW differ from the other stability measures of the group due to the way they normalize the pairwise scores. Their normalizations result in the stability measures not fulfilling the maximum property, see Section 6.1. SMN takes similar stability values as SMK, SMU, and SMP, when all of them attain high values. For combinations where SMK, SMU, and SMP take values around 0 or smaller, the values of SMN can differ quite much. The reason is that SMK, SMU, and SMP subtract the expected values of $|V_i \cap V_j|$, given that feature sets of cardinalities $|V_i|$ and $|V_j|$ are selected, in the pairwise scores. So, for example, when $V_i \subsetneqq V_j$ and $|V_j| = p$, SMK and SMU have the value 0, independent of $|V_i|$. However, intuitively, in this example the stability scores should increase with increasing $|V_i|$. SMN can represent the intuitive differences in stability by assigning different stability scores.

Comparing the first two groups, it can be observed that the uncorrected measures take larger or equal values compared to the corrected measures for almost all feature set combinations. This effect was expected, because the motivation for correcting a measure for chance is avoiding high stability values that are simply due to the selection of many features.

The adjusted stability measures differ strongly from the unadjusted measures with respect to their stability assessment behavior. Within the group of adjusted measures, SMA-Mean, SMA-Count, SMA-Greedy, and SMA-MBM take almost identical values for all combinations. The values assigned by SMY and by the SMA variants are also quite similar. However, for combinations that are assigned large stability values by all of these measures, SMY often attains larger values than the SMA measures. These are combinations for which several features from the one set are mapped to the same feature of the other set, see the discussion in Subsection 3.3.2.3. This undesired behavior of SMY occurs for large stability values. This is problematic because large stability values are what an optimizer will be searching for later on, see Chapters 7 and 8. Stability measure SMES differs noticeably from the other measures in the group. In many situations, SMES attains much smaller values. For SMES, the similarity values of the features included in the same set are important. Table 6.4 provides an explanatory example. In both of the scenarios in Table 6.4, the intersection of the two sets of selected features is empty and none of the features in the one set is similar to any of

V_1	V_2	SMA-Mean	SMA-Count	SMA-Greedy	SMA-MBM	SMY	SMES
$\{X_4, X_5\}$	$\{X_1, X_7\}$	-1.1238	-1.1832	-1.1832	-1.1832	-1.2500	-1.6686
$\{X_4, X_5\}$	$\{X_6, X_7\}$	-1.1238	-1.1832	-1.1832	-1.1832	-1.2500	-2.5280

Table 6.4: Stability values for two combinations of sets of selected features, V_1 and V_2.

the features in the other set. In the first scenario, the features in set V_2 are not similar to each other, in the second scenario they are, see Figure 6.1. In the second scenario, SMES takes a much smaller value than in the first scenario. The other stability measures attain the same value in both scenarios because they do not consider the similarities between features in the same set. It seems intuitive that a stability measure should attain the same value in both scenarios because the similarity values of X_1 to X_4 and X_5 are exactly identical to the similarity values of X_6 to X_4 and X_5.

Now, the dependence of the stability values on the number of selected features is analyzed. Figure 6.3 displays for each stability measure the attained values for feature sets with different mean cardinalities. Also, the mean stability value per mean number of selected features is displayed. Remember that in this analysis, all possible combinations of feature sets are considered. Therefore, the mean stability value is equal to the expected stability value for a random feature selection of two sets with the respective mean cardinality. For the corrected measures, the expected stability value is constant with respect to the number of selected features. This was expected because this is the definition of correction for chance. For SMN, the expected value is smaller than 0 whereas it is 0 for all other corrected measures. For SMD-0, SMO, SMD, SME, SMZ, and SMJ, the expected value increases with the number of selected features. For SMH, the expected value is smallest for a mean cardinality of 3.5 features, which is half of the 7 existing features. For SMS and SMES, it is largest if 3.5 features are selected on average. For SMES, the expected value varies only little with respect to the mean number of selected features. This makes it plausible that SMES belongs to the group of adjusted and corrected stability measures in Figure 6.2, even though it is not corrected for chance.

Figure 6.3 also provides information about which stability values are attainable with respect to the number of selected features. All corrected measures, SMH, and SMES can only attain their smallest values for a mean cardinality of 3.5 features. SMD-0, SMO, SMD, SME, SMZ, and SMJ can attain their minimum value for all mean cardinalities up to 3.5. Only SMS can take its minimum value for arbitrary mean numbers of selected features.

SMS, SMW, and SMY can take their maximum value for arbitrary mean numbers of selected features, also for feature sets of different sizes. These effects have been described in Section 6.1 with the measures violating the “maximum for equal cardinalities” property. For all other measures, the maximum values can only be reached for sets of equal cardinalities. SML does not attain its maximum value. The largest value that SML reaches is attained in situations where both sets have the same cardinality of 1 or 6. All other stability measures for which a maximum value is known attain their maximum value.

Another observation of Figure 6.3 is that for the corrected measures, there seem to be the fewer distinct attainable values, the more the mean number of selected features deviates from 3.5. The reason is that there are more possibilities for two feature sets to have a mean cardinality around 3.5 than there are for having a very small or very large mean cardinality. Also, in this limited scenario with only 7 features, there are only very few combinations with extreme mean cardinalities that are different from each other and not just symmetric versions of each other.

6.3 Empirical Comparison on Real Feature Sets

Now, the stability measures are compared based on feature sets that are selected for real data sets. While in Section 6.2 all possible sets for a limited scenario were considered, this analysis is based on realistic feature sets resulting from real applications.

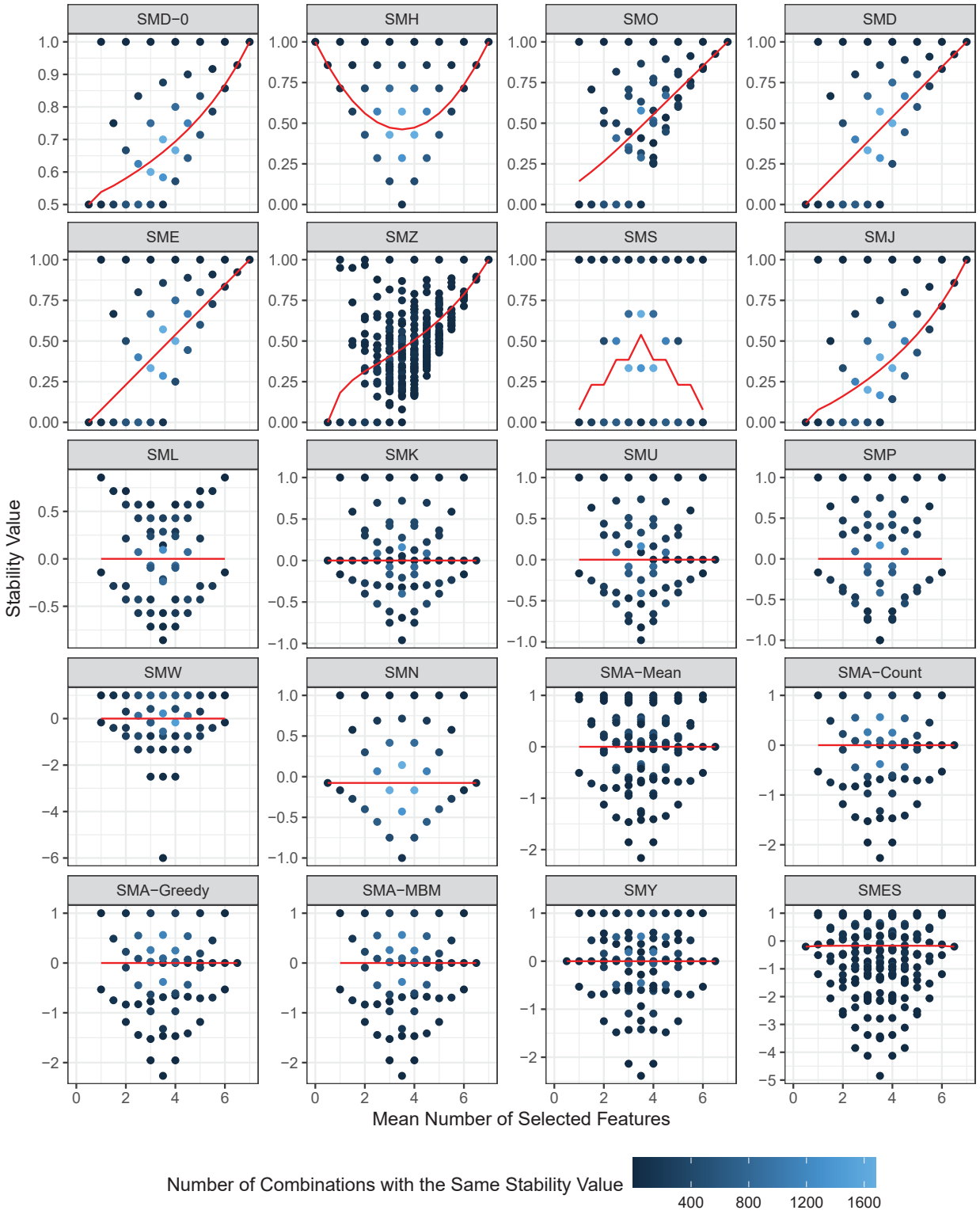


Figure 6.3: Dependence of the stability values on the number of selected features. The red line indicates the mean stability value per mean number of selected features. The order of the stability measures is the same as in Figure 6.2.

6.3.1 Experimental Setup

The setup of this study is designed so that the results can be used for the analyses in Chapter 7 as well. The analyses are based on all data sets presented in Section 4.1. Each data set is split into two equally sized halves, with equal class ratios in both halves (stratified split). The halves will be the training and the test data in Chapter 7. In this analysis, only the training half of each data set is considered.

For feature selection, combined methods that consist of a filter method and a classification method, like in Chapter 5, are applied. These combined methods first apply a filter and then learn the classification rule on the remaining features. Based on the results of the analyses in Chapter 5, the filter methods *JMI*, *MIM*, and *impurity* as well as applying no filter are considered. Not filtering at all is included in the search space because in this analysis, also classification methods that perform an embedded feature selection are employed. The filter methods have one hyperparameter indicating the percentage of features to select. The range of this hyperparameter is $[0\%, 100\%]$. The classification methods that are considered for this analysis are displayed in Table 6.5 with their corresponding hyperparameters. The range for the hyperparameter *nodesize* is taken from Probst et al. (2019). Note that lasso logistic regression, random forest, and glm boosting conduct an embedded feature selection, that is, the set of selected features is determined by both the filter and the classification method. Support vector machine, k nearest neighbors, and ridge logistic regression use all features, so the feature selection is only conducted by the filter method. Each filter method is combined with each classification method, resulting in $4 \cdot 6 = 24$ combined methods. For each of the combined methods, 200 hyperparameter configurations are drawn randomly. So, for each data set, $24 \cdot 200 = 4\,800$ configurations are analyzed.

For each data set, 10 parts of this data set are generated from the training half with 10-fold cross-validation. Then, the configurations are applied to the 10 parts. So, for each original data set, 10 models are fitted. Based on these models, the sets of selected features

Method	Hyperparameters	R package
Support Vector Machine	$C, \sigma \in \{2^x : x \in [-15, 15]\}$	*kernlab* (Karatzoglou et al., 2004)
k Nearest Neighbors	$k \in \{1, 2, \ldots, 20\}$	*kknn* (Schliep and Hechenbichler, 2016)
Ridge Logistic Regression	$\lambda \in \{2^x : x \in [-15, 15]\}$	*glmnet* (Simon et al., 2011)
Lasso Logistic Regression	$\lambda \in \{2^x : x \in [-15, 15]\}$	*glmnet* (Simon et al., 2011)
Random Forest	*ntree* $\in \{100, 200, 500, 1\,000, 2\,000, 5\,000\}$, *nodesize* $\in \{[(n \cdot 0.2)^x] : x \in [0, 1]\}$, *mtry* $\in \{1, 2, \ldots, p_{\text{filter}}\}$	*ranger* (Wright and Ziegler, 2017)
GLM Boosting	*n.iter* $\in \{[2^x] : x \in [0, 15]\}$	*mboost* (Hothorn et al., 2018)

Table 6.5: Classification methods with corresponding hyperparameters and R package of which the implementation is used. $[\cdot]$ denotes rounding to the closest integer, p_{filter} is the number of features in the data set after filtering and n is the number of observations in the data set.

are determined and the mean number of selected features is recorded. In the analyses in this section, the proportion of selected features is considered instead of the number of selected features for comparability across data sets. Also, the stability of the feature selection of the configurations is evaluated with all stability measures based on the 10 feature sets obtained from the 10 models. For all adjusted stability measures, the similarities between the features are quantified with the absolute Pearson correlation. For SMES, the similarity matrix C (see Subsection 3.3.2.3) is determined by setting all values in the correlation matrix that are smaller than the threshold $\theta = 0.9$ to 0. For SMY, SMA-MBM, SMA-Greedy, SMA-Count, and SMA-Mean, the expected values of the pairwise scores are estimated based on 10 000 values, as described in Subsection 3.3.2.3. In preliminary studies, 10 000 values have shown to provide a good compromise between convergence and run time. Here, it is computationally infeasible to determine the expected values exactly like it has been done in Section 6.2.

For all configurations, the same cross-validation splits are used. The advantage of cross-validation compared to other resampling strategies is that the overlap of each pair of data set parts is known. This is relevant for stability assessment, because the similarity of the data sets influences the stability of the feature selection, see Alelyani et al. (2011). For 10-fold cross-validation, the data set parts share $\frac{80}{90}$ of their observations.

There are 54 configurations for which no results could be obtained: 23 on *sonar*, 14 on *kc1-binary*, 9 on *tecator*, 1 on *har*, 1 on *gina_agnostic*, 2 on *dilbert*, and 4 on *lsvt*. Like in Section 5.3, the failing of all configurations is caused by the selection of less than two features by the filters and the implementation of ridge logistic regression and lasso logistic regression in *glmnet*, which requires at least two features. The configurations for which no results were obtained are ignored for the analyses.

6.3.2 Results

To start the comparison of the stability measures based on real feature sets, an overview of the values attained by the stability measures is given. Figure 6.4 shows the values that the stability measures take for the features selected by the 4 800 configurations per data set. It illustrates the ranges of attained values with respect to the proportion of selected features. The plots in Figure 6.4 are similar to the ones in Figure 6.3, but they are based on the stability values calculated for feature sets resulting from real data.

Like in Figure 6.3, it can be observed here that SME, SMD, SMO, SMD-0, SMJ, and SMZ do not attain small values if many features are selected. Also, SMH does not take small values for large or small numbers of selected features and SML does not attain large values for medium numbers of selected features. For the corrected measures, there are no systematic restrictions on the attained values. Unlike in Figure 6.3, the corrected but not adjusted measures SMW, SMN, SMK, SMP, and SMU hardly take any negative values here. So, in realistic feature selection situations, almost none of the feature selections are less stable than a random feature selection.

To analyze the similarities between the stability measures, Pearson correlations between all pairs of stability measures are calculated for all data sets. The resulting plots are shown in Figures A.13 and A.14 in Appendix A.3. Then, the correlations are aggregated across data sets by calculating the arithmetic mean. Figure 6.5 displays the results. There are three groups of similar stability measures. SMH is one group by itself. The rest of the uncorrected stability measures except for SMS and SMES form the second group. The third group consists of all corrected measures, SMS, and SMES. All groups are very homogeneous.

This group structure is observed for most data sets. For *tecator*, *lsvt*, and *arcene*, the group of corrected measures is split into adjusted and unadjusted stability measures. Also,

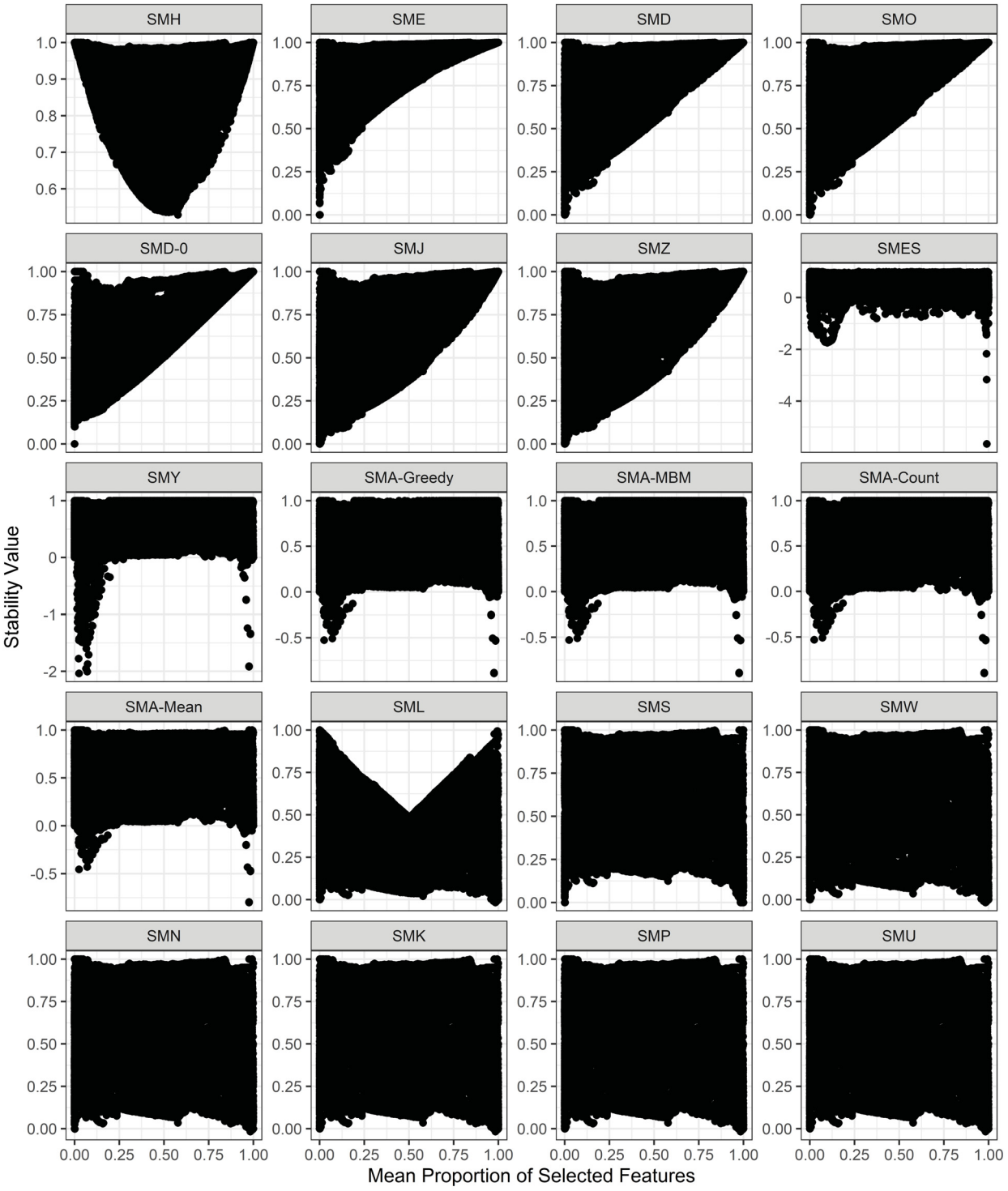


Figure 6.4: Dependence of the stability values on the proportion of selected features. The order of the stability measures is the same as in Figure 6.5.

SMES is a group by itself for these data sets. The data sets *tecator*, *lsvt*, and *arcene* are the ones with many similar features. It is expected that the adjusted stability measures only differ noticeably from the unadjusted measures if there is a sufficient number of similar features in a data set. Also, as demonstrated in Table 6.4, SMES deviates strongly from the adjusted and corrected stability measures when sets of similar features are selected. The adjusted but uncorrected measure SMZ belongs to the group of uncorrected and unadjusted measures for all data sets. So, the existing similarities in the considered data sets do not seem

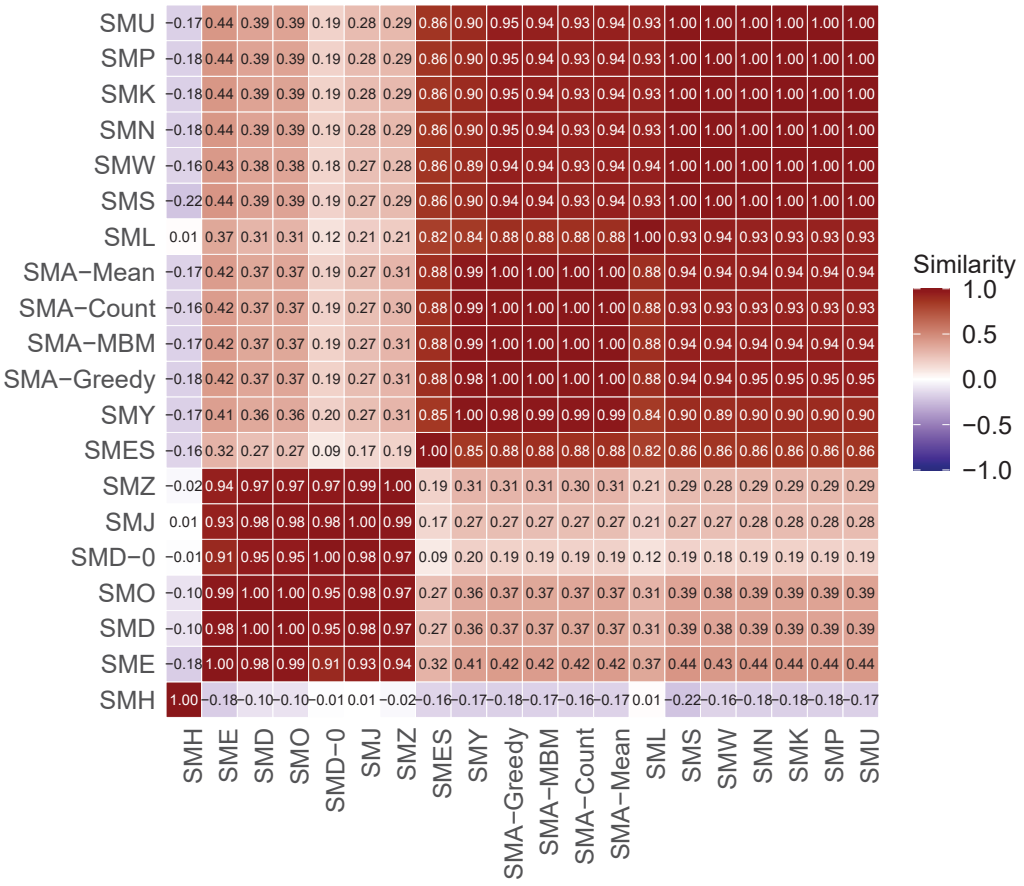


Figure 6.5: Mean Pearson correlations between all pairs of stability measures rounded to two digits. The correlations between the stability measures are averaged across data sets. The order of the stability measures is determined by average linkage hierarchical clustering using the mean Pearson correlation as similarity measure between stability measures.

to be sufficient for showing differences between SMZ and the unadjusted and uncorrected measures.

Compared to the similarity analysis of the stability measures in Section 6.2, the groups differ slightly. The main difference is that in this analysis, the stability measures SMH and SMS are not part of the group of uncorrected stability measures. SMS belonging to the group of corrected measures can be explained by its asymptotic correction for chance (Nogueira, 2018). The corrected measures, SMS, and SMES being grouped together in Figure 6.5, even though some of them are adjusted and others are not, is due to the choice of data sets. Many of the data sets contain only few similar features, while in Section 6.2, a scenario with many similar features was considered. Also, here, the stability assessment behavior of SMN is almost identical to the way the other corrected but unadjusted stability measures assign stability values. This also means that the advantageous behavior of SMN for unstable feature selections does not come into play for the real data situations considered here. In Section 6.2, the advantageous behavior was observed for situations in which SMN attained values of 0 and below and such situations hardly occurred for the real data feature sets, see Figure 6.4.

To analyze the differences of the stability measures more in detail, scatter plots of the stability values are shown in Figure 6.6. Because the members of the groups are so highly correlated, only representatives of the groups shown. SMJ represents the group of uncorrected and unadjusted measures, SMU represents the group of corrected and unadjusted measures,

and SMA-Count represents the group of corrected and adjusted measures. It can be observed that SMH and SMJ differ especially in situations in which only few features are selected on average. For such situations, SMH attains large values because the number of features not included in any of the feature sets is high, while SMJ often attains lower values. If many features are selected on average, the measures attain very similar values. Compared to SMU, SMJ takes larger values if many features are selected. For small numbers of selected features, SMJ and SMU attain fairly similar values. SMH attains larger values than SMU for almost all of the considered feature sets. Comparing SMU and SMA-Count, it can be observed that many values are almost identical. These values result from the data sets with only few similar features. For data sets with many similar features, the measures attain different values and a tendency can be observed that SMA-Count assigns larger values than SMU in situations with many selected features. In situations with only few selected features, SMU has a tendency to

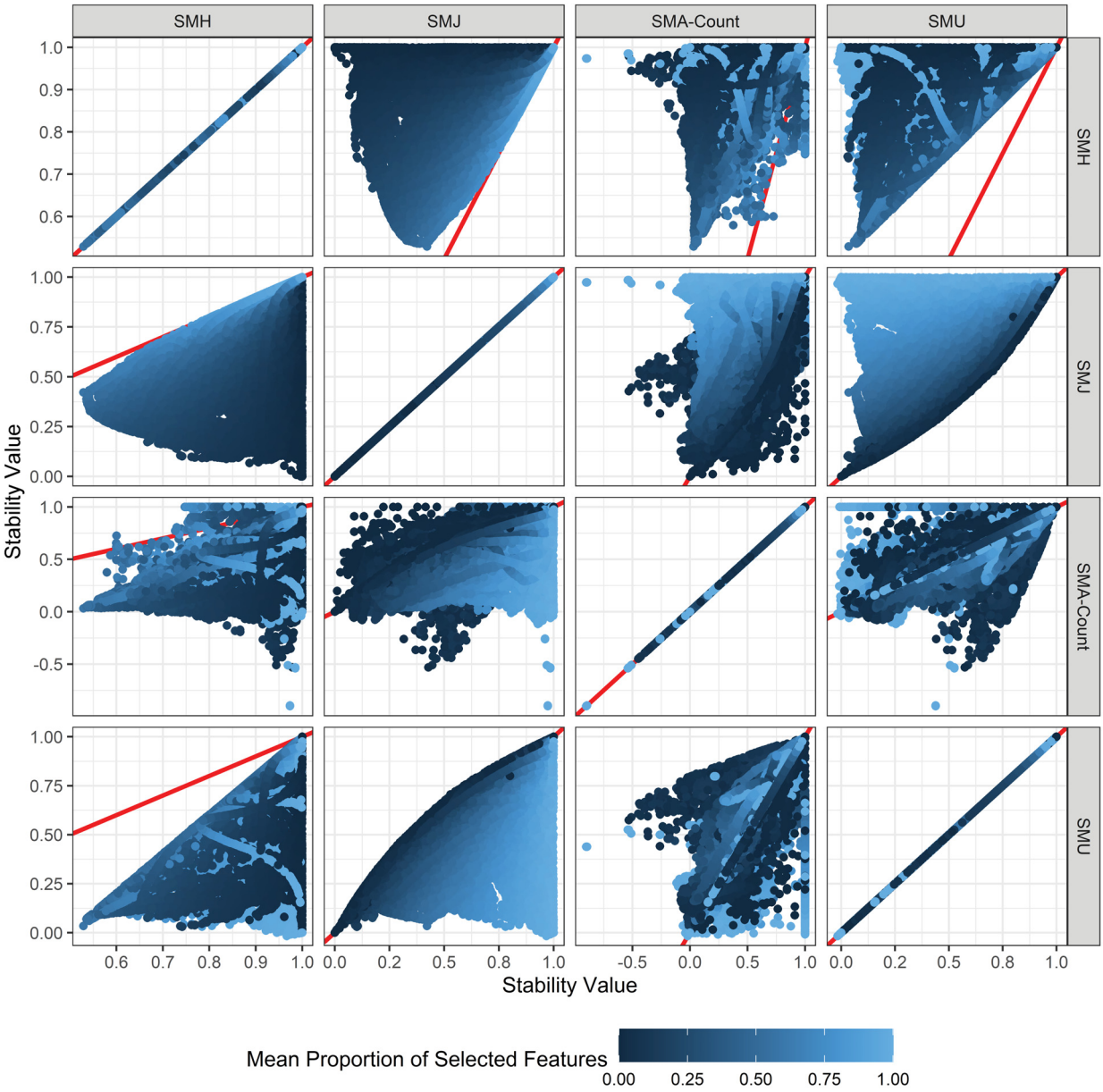


Figure 6.6: Scatter plots of the stability values for all configurations and all 12 data sets. The displayed stability measures are representatives of the groups found in Figure 6.5. The red line indicates the identity.

assign larger values than SMA-Count. This can be explained by the probability that features that are similar to other features in the data set are among the selected features being larger if more features are selected. If more similar features are among the selected features, it is more likely that the adjustment takes a positive value and that therefore, SMA-Count attains a larger value than SMU.

Now, the run times of the stability measures are compared. Figure 6.7 displays the run times for calculating the values of the stability measures for all data sets. Each boxplot per data set is based on the same feature sets that are selected by the 4 800 configurations. The computation of the unadjusted stability measures is very fast. Calculating the value of the adjusted but uncorrected measures SMZ and SMES takes a bit longer, but still can be done in less than one second for most data sets. The run times of the corrected and adjusted measures are much longer than the run times of the other measures. The reason is that the expected values of the scores have to be estimated, which involves frequently repeated evaluation of the adjustments.

Figure A.15 in Appendix A.3 displays the run times of the adjusted and corrected stability measures only to allow a better comparison between these measures. While for the unadjusted measures, the differences in run time do not really matter because all of them are fast to compute, the differences do matter for the stability measures that are both adjusted and corrected. For all data sets, SMY and SMA-Count require the least time for calculation among the adjusted and corrected measures. For most data sets, the calculation of SMA-Mean, SMA-Greedy, and SMA-MBM takes much longer. For large data sets, the latter computation times are not acceptable.

6.4 Conclusions

The empirical comparisons based on both artificial and real feature sets have revealed that there are groups of stability measures with very similar stability assessment behavior. One group is formed by the stability measures that are not corrected for chance except SMES. The second group consists of the measures that are corrected for chance but do not take into account similarities between features. The third group is composed of the adjusted measures except SMZ. For the feature sets obtained from real data sets, the uncorrected stability measure SMH behaves so differently from the other uncorrected measures that it forms a group by itself. To reduce the number of considered stability measures, one representative of each group of similar stability measures can be chosen. For selecting a suitable measure for each group, the theoretical properties analyzed in Section 6.1 and the run times analyzed in Section 6.3 are taken into account.

For the group of adjusted stability measures, the four variants of SMA fulfill more desirable properties than SMY and SMES. The sets of desirable properties fulfilled by SMY and SMES are subsets of the properties fulfilled by SMA-Count, SMA-MBM, SMA-Greedy, and SMA-Mean. The four variants of SMA fulfill the same theoretical properties. It makes sense to choose the fastest of these measures, because their run times are quite long. Therefore, SMA-Count is chosen for the group of corrected and adjusted stability measures. Regarding the newly proposed stability measure SMA, the four considered variants differ in the way they take into account similar features when evaluating the stability. Even though the adjustments for similar features are conceptually different for the four variants, the resulting stability values are very similar both on artificial and on real feature sets.

Within the group of corrected but unadjusted measures, only SMN, SMK, SMP, and SMU fulfill all of the desired theoretical properties. These measures have shown to be very similar, especially in the comparison based on real feature sets, so any of the measure could be

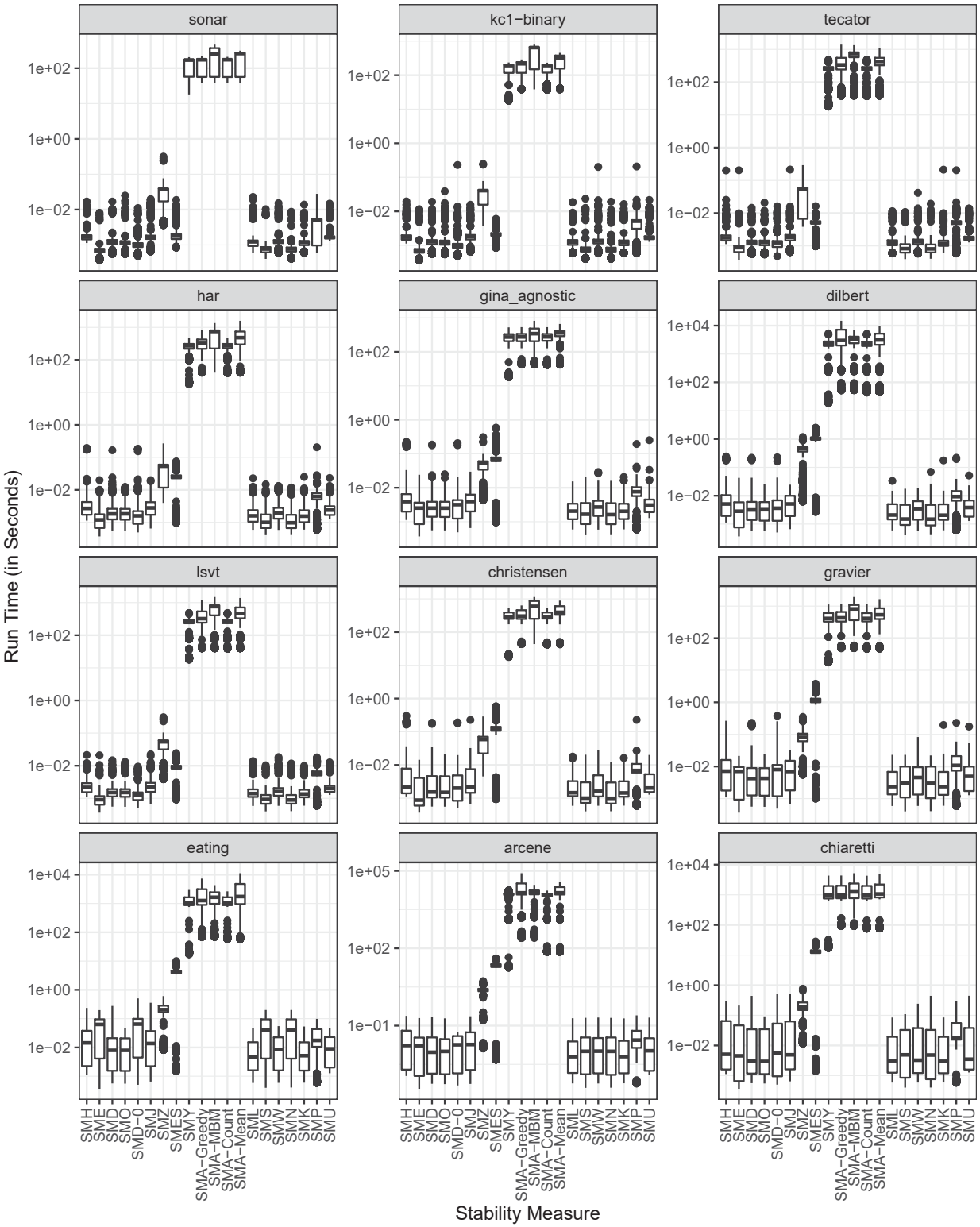


Figure 6.7: Run times of all stability measures and all 12 data sets. Note that the y-axis is scaled logarithmically. The order of the stability measures is the same as in Figure 6.5.

selected. Here, SMU is chosen because, if there are no similar features, SMU and SMA-Count are identical and therefore, the effect of similar features can be seen best.

Of the group of uncorrected measures, SME, SMD, SMO, and SMJ fulfill all theoretical properties, except for the correction for chance. The run time is not an issue for these measures, as they are all very fast to compute. One possible choice would be SMJ as the most popular and the most widely used of these measures. Another possible choice would be SMD as the measure that is on average the most similar to all other measures of this group. Here, SMJ is chosen. For further analyses, it should be kept in mind that the uncorrected stability measures lack an important property, so it can make sense to only consider SMU and SMA-Count instead of a representative of each group.

Chapter 7

Finding Desirable Configurations by Multi-Criteria Tuning

In this chapter, a strategy for finding configurations with high predictive accuracy and a stable selection of a small number of features is analyzed. The aim of the analysis is investigating whether it is possible with the proposed approach to find configurations that perform a more stable selection of fewer features without losing much predictive accuracy compared to model fitting only considering the predictive performance.

We have proposed this strategy in Bommert et al. (2017) and evaluated it based on 3 gene expression data sets. In this chapter, the strategy is evaluated on 12 different data sets, but the same conclusions are drawn.

7.1 Proposed Approach

For finding configurations with high predictive accuracy and a stable selection of a small number of features, the hyperparameter tuning of the considered predictive methods is performed in a multi-criteria fashion. First, the Pareto optimal configurations with respect to maximal predictive accuracy, maximal feature selection stability, and minimal number of selected features are determined. Then, a subset of these configurations is selected. This subset consists of the Pareto optimal configurations whose predictive accuracy is not lower than $acc.max - 0.05$ where $acc.max$ is the best predictive accuracy observed on the same data set. In the following, these configurations are called "PA-best" configurations.

It is reasonable to base the analyses on these configurations, because the prediction accuracy of a predictive model is its most important performance criterion - often the only one considered during tuning. Allowing configurations with a bit lower accuracy permits analyzing potential gains in stability and number of selected features when not the very best configuration with respect to predictive accuracy is chosen. The PA-best configurations are compared to the best configurations obtained with single-criteria tuning with respect to predictive accuracy. The latter configurations are named "A-best" configurations for the remainder of this chapter.

7.2 Experimental Setup

A random search for configurations that lead to sparse and stable models with high predictive accuracy is conducted. This study builds upon the study conducted in Section 6.3. For the study in Section 6.3, the 12 data sets presented in Section 4.1 were used. Each of them was split into two halves of equal size with stratification of the class variable. Here, one half is used as training data, the other half as test data. Combined methods consisting of a filter and a classification method were considered. For each of the 24 combined methods,

200 configurations were drawn randomly for each data set, resulting in 4 800 configurations to evaluate, see Subsection 6.3.1. There are 54 configurations for which no results could be obtained and which are ignored for the analyses, see Subsection 6.3.1

Now, for selecting the PA-best and A-best configurations, all configurations are evaluated on the training data. For this, 10-fold cross-validation is performed, that is, 10 models are fitted, each based on 90% of the observations. For each model, the class is predicted on the 10% of the observations that were not used for fitting the model and the classification accuracy is calculated. Then, the mean value of the 10 classification accuracy rates is assessed as a measure of the predictive performance of the configuration. Additionally, for each model, the set of selected features is determined. To assess the mean size of the models, the mean cardinality of the sets of selected features is determined (mean number of selected features). Also, the stability of the feature selection of the configuration is evaluated based on the 10 features sets obtained from the 10 models. For stability assessment, the stability measures SMU, SMA-Count, SMJ, and SMH are chosen based on the results of the analyses in Chapter 6.

In Section 6.3, the stability of the feature selection of all configurations was analyzed. Here, the PA-best and A-best configurations are chosen and evaluated on the test halves of the data sets. For the evaluation on the test data, 10-fold cross-validation is performed and the mean classification accuracy, the mean number of selected features, and the stability are determined. Evaluating the chosen configurations on data that has not been used for choosing the configurations allows to assess unbiased estimates for the three target criteria. It is necessary to perform resampling on both halves of the data sets in order to be able to evaluate the stability on both halves. By using this procedure, the number of observations is roughly equal for all model fits on both training and test data. For all configurations evaluated on the same data set, the same cross-validation splits of both the training and the test data set are used.

7.3 Results

The proposed approach for finding desirable configurations is evaluated. In Subsection 7.3.1, the PA-best configurations obtained with the different stability measures are compared. In Subsection 7.3.2, the PA-best configurations are compared to the A-best configurations with respect to predictive accuracy, feature selection stability, and number of selected features on the test data.

7.3.1 Comparison of Optimal Configurations Between the Stability Measures

As discussed in Chapter 6, the stability measures SMU, SMA-Count, SMJ, and SMH differ with respect to their stability assessment behavior. These differences also result in different sets of Pareto optimal configurations when considering the classification accuracy, stability, and number of selected features. Table 7.1 displays the number of Pareto optimal and PA-best configurations per data set and stability measure as well as the number of A-best configurations per data set. For all data sets, using SMJ leads to the largest number of Pareto optimal and PA-best configurations. For data sets *dilbert*, *christensen*, and *chiaretti*, many of the 4 800 configurations per data set achieve the same best classification accuracy on the training data. As discussed in Section 4.1, these are data sets for which the separation of the classes seems to be comparably easy.

Data set	SMU (all)	SMA-Count (all)	SMJ (all)	SMH (all)	SMU (PA)	SMA-Count (PA)	SMJ (PA)	SMH (PA)	A-best
sonar	97	88	132	65	17	16	31	20	1
kc1-binary	95	30	170	51	25	8	38	7	1
tecator	37	41	57	24	31	35	51	23	1
har	507	492	553	502	22	19	68	23	1
gina_agnostic	139	90	260	124	37	36	111	62	1
dilbert	650	551	856	713	137	108	342	209	954
lsvt	558	501	623	475	21	18	42	21	3
christensen	483	483	483	483	67	67	67	67	3414
gravier	687	563	1027	506	6	6	6	4	2
eating	946	867	1139	505	5	4	5	3	1
arcene	825	538	1160	593	69	25	84	7	2
chiaretti	744	549	1010	656	126	64	282	170	2207

Table 7.1: Number of Pareto optimal configurations per data set and stability measure (all), number of PA-best configurations per data set and stability measure (PA), and number of A-best configurations per data set (A-best).

To analyze the differences in the sets of optimal configurations between the stability measures, Figure 7.1 shows the PA-best configurations per stability measure and data set. For most data sets, there are some configurations that are PA-best for all stability measures, while most configurations are only PA-best for some of the measures. The agreement on PA-best configurations is largest for the data sets *christensen*, *gravier*, and *eating*. For most data sets, there are configurations that are just optimal for SMJ or for SMJ and SMH. For the data sets *tecator* and *lsvt*, which include many similar features, there are some configurations that are only optimal for the adjusted stability measure SMA-Count.

Figure 7.2 as well as Figures A.16, A.17, and A.18 in Appendix A.3 show the classification and filter methods of the PA-best configurations for the different stability measures. Figure A.19 in Appendix A.3 displays analogous plots for the A-best configurations. The classification and filter methods of the best configurations differ strongly between the data sets. For all data sets, it can be observed that for the classification methods that perform embedded feature selection (lasso logistic regression, random forest, and glm boosting) many of the best configurations employ a filter. So, even if there already is feature selection included in the model fitting process, it seems beneficial to apply a filter first. For SMU and SMA-Count, it can be observed that among the PA-best configurations there are no methods that do not perform feature selection at all. Such methods would consist of one of the classification methods that do not perform embedded feature selection (support vector machine, k nearest neighbors, and ridge logistic regression) and no filter. For SMJ and SMH, such configurations are Pareto optimal because both stability measures attain their maximum value if all features are included in all of the 10 models. For some data sets, such configurations are among the PA-best configurations for SMJ and SMH.

7.3.2 Performances of the Optimal Configurations

The PA-best configurations per data set are determined based on their performance (mean classification accuracy, mean number of selected features, and stability) on the training data. Figure 7.3 and Figures A.20 and A.21 in Appendix A.3 display the differences in performance

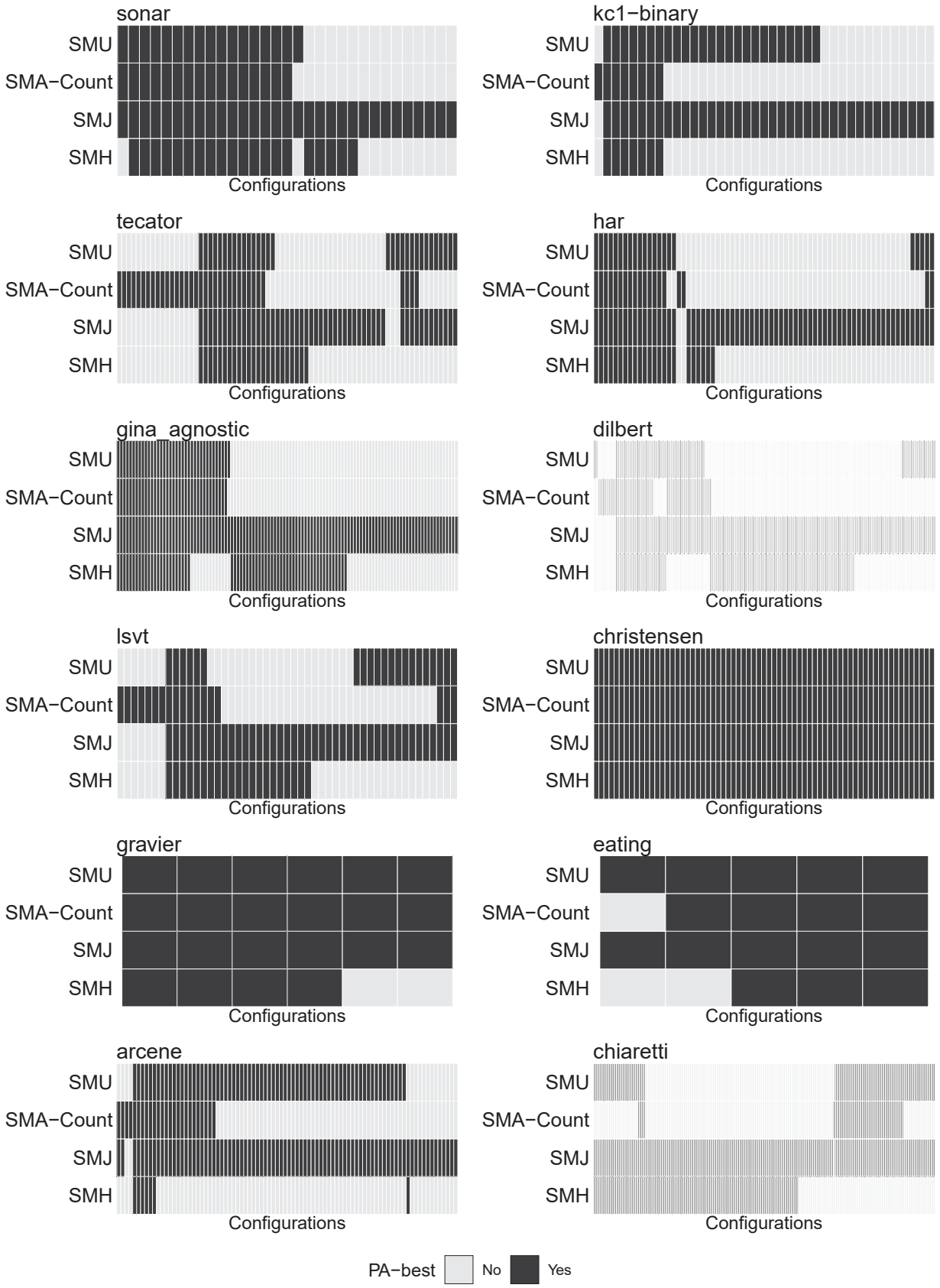


Figure 7.1: PA-best configurations per stability measure and data set. Only the configurations that are PA-best for at least one stability measure are displayed.

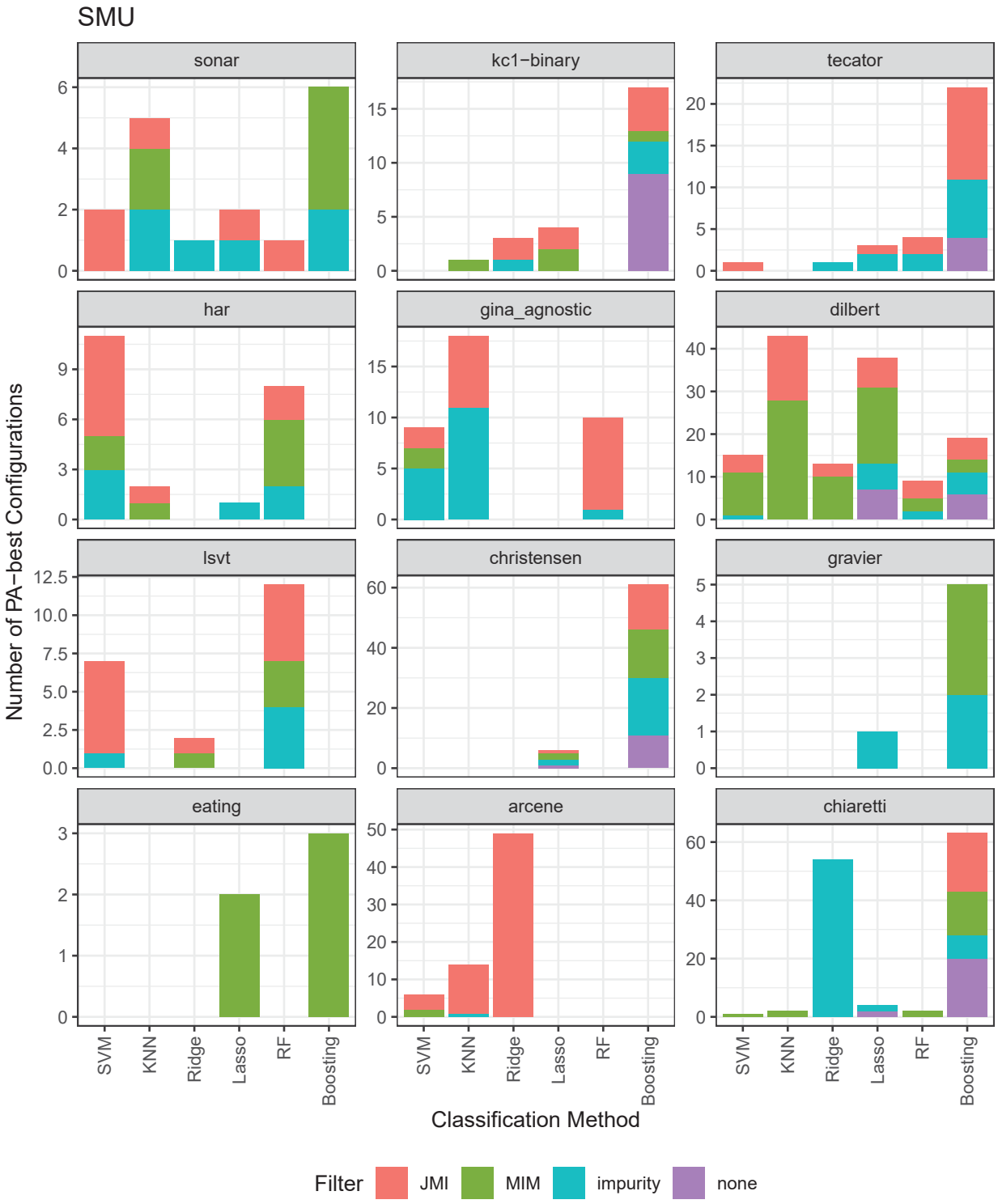


Figure 7.2: Classification and filter methods of the PA-best configurations per data set for the stability measure SMU.

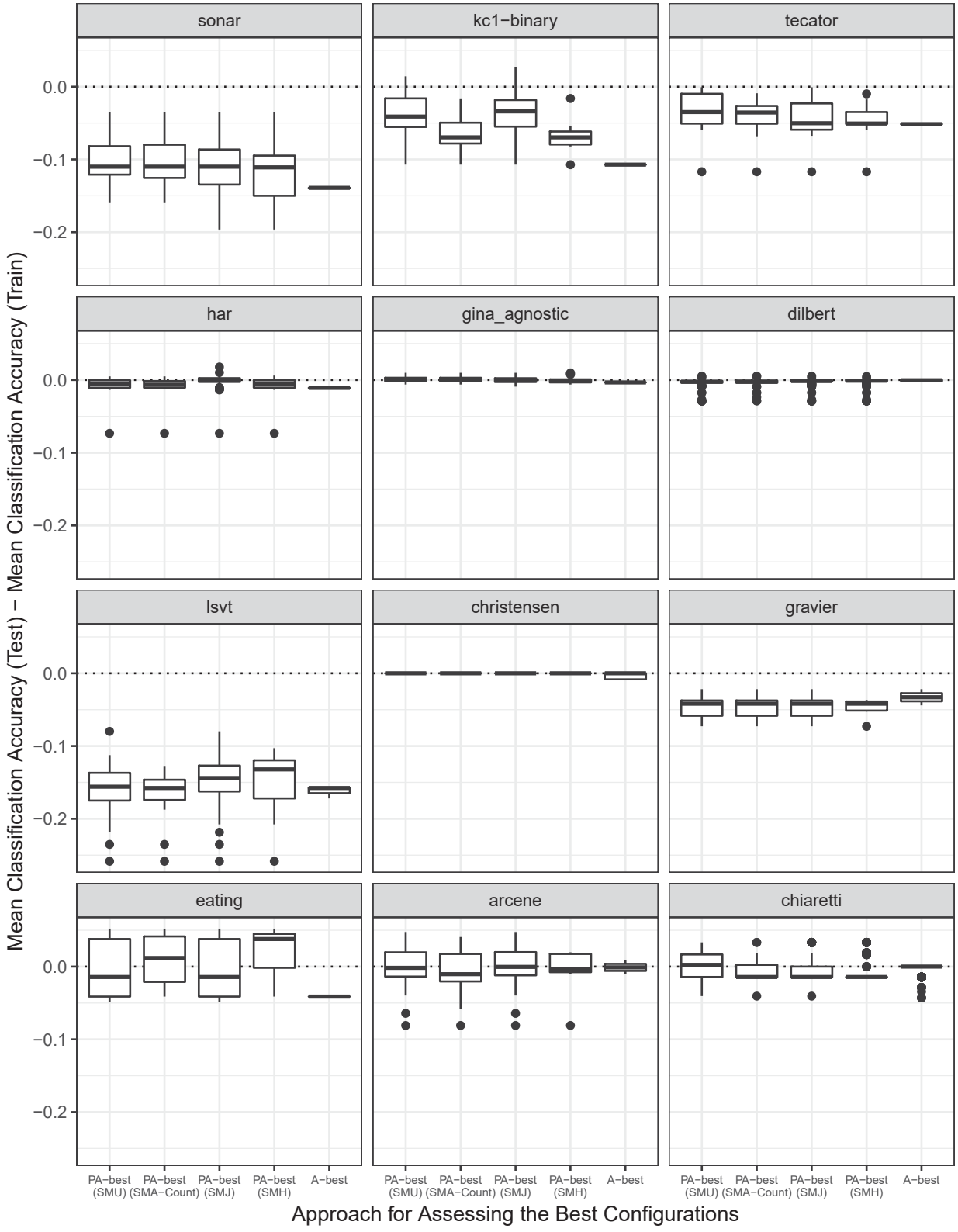


Figure 7.3: Differences in mean classification accuracy between training and test data for the PA-best and A-best configurations. Values above the dotted line indicate a higher mean classification accuracy on the test data than on the training data.

between the training and test data for the PA-best configurations. Figure 7.3 additionally displays the differences in predictive performance for the A-best configurations.

Figure 7.3 shows that the differences in predictive accuracy for the PA-best configurations are small for most data sets. For *har*, *gina_agnostic*, *dilbert*, and *christensen*, the PA-best configurations have almost the same predictive accuracy on the test data as on the training data. *har*, *gina_agnostic*, and *dilbert* are data sets with very many observations. For *sonar*, *kc1-binary*, *tecator*, *lsvt*, and *gravier*, the classification accuracy on the test data is noticeably lower than on the training data. The largest differences are observed for data set *lsvt*. For *eating*, *arcene*, and *chiaretti*, there are configurations with higher predictive accuracy on the test data than on the training data. These are the data sets with very many features and more features than observations. Especially for *sonar*, *kc1-binary*, *tecator*, and *eating*, it can be observed that the decrease in classification accuracy from training to test data is much larger for the A-best configurations than for the PA-best configurations. This means that the best configurations found with single-criteria tuning over-fit the training data much more than the configurations obtained with multi-criteria tuning. This observation applies to all of the considered data sets, except for the gene expression data sets *gravier* and *chiaretti*.

Figure A.20 demonstrates that the numbers of selected features do not differ much between training and test data for all data sets. This is plausible because the number of selected features is almost fixed for given hyperparameter values. For the classification methods without embedded feature selection, the number of selected features is completely determined by the filter percentage. For the other classification methods, the number of selected features can vary between training and test data. However, for these methods, the hyperparameters (for example the number of boosting iterations or the weight of the lasso penalty) only allow the number of selected features to vary within a small range.

Regarding the differences in feature selection stability, Figure A.21 shows that for most data sets, the configurations are equally stable on the training and test data. The largest differences are observed for PA-best configurations determined with the stability measure SMA-Count on data sets *tecator* and *lsvt* as well as for the PA-best configurations for data set *sonar* determined with SMU or SMA-Count. In all of these cases, the stability on the test data is lower than on the training data.

Summarizing the content of Figures 7.3, A.20, and A.21, for most data sets, the performances of the PA-best configurations on the training and test data are very similar. In the following, plots are analyzed, in which the performances of the PA-best configurations on the test data are shown. For this, it should be kept in mind that the performances on the test data are most likely very similar to the performances on the training data. Six-dimensional plots displaying all performance values per configuration cannot be shown.

Figures 7.4 and 7.5 display the performances on the test data of the PA-best configurations for all considered stability measures and data sets. Also, the performances of the A-best configurations are shown. The sets of PA-best and A-best configurations overlap. Some of the A-best configurations are also PA-best configurations, but not all of them, and vice versa. A-best configurations that have the same classification accuracy but lower stability or a larger number of selected features than at least one PA-best configuration on the same training data set are not Pareto optimal and therefore not PA-best either. Whether a configuration is PA-best, A-best or both is indicated by the shape of its point. The stability and the mean number of selected features are displayed by the position of the points. The mean classification accuracy is displayed by color. Optimal configurations with respect to all three target criteria would be located in the upper left corner (minimal number of selected features and maximal stability) and their color would be dark green. The performances of the configurations on the test data can be seen as unbiased estimates for the performance

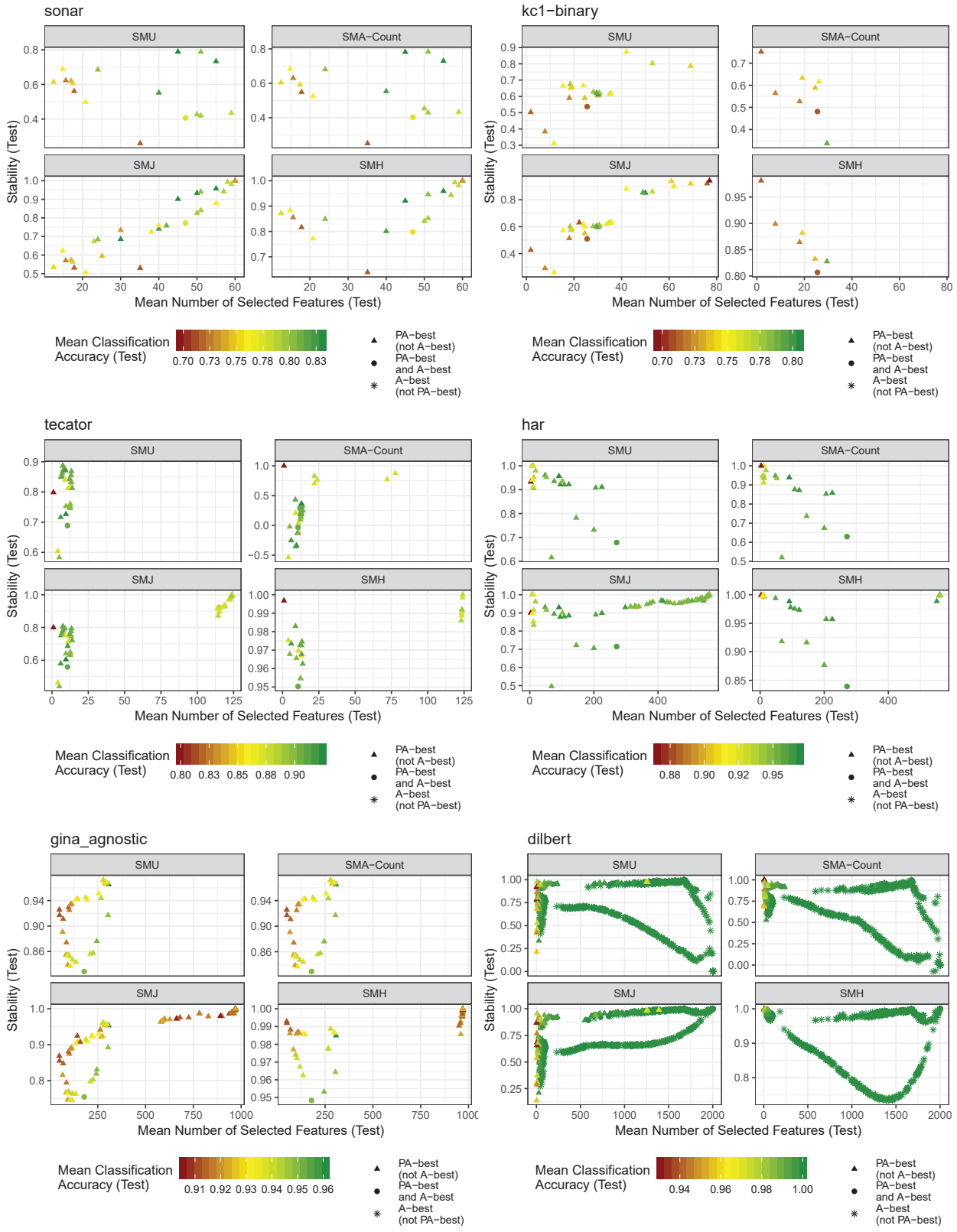


Figure 7.4: Stability, mean number of selected features, and mean classification accuracy on the test data of the PA-best and A-best configurations for data sets *sonar*, *kc1-binary*, *tecator*, *har*, *gina_agnostic*, and *dilbert*.

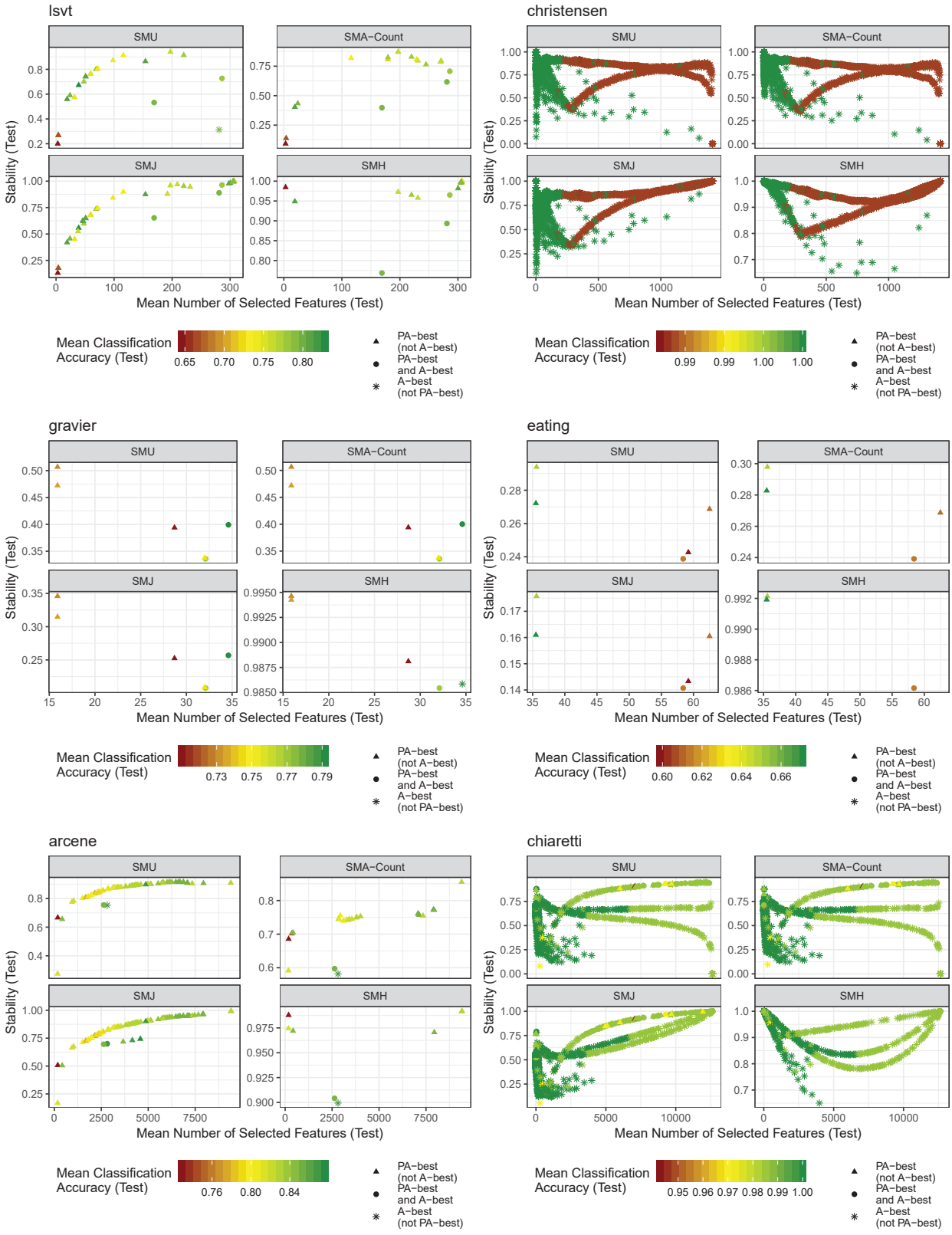


Figure 7.5: Stability, mean number of selected features, and mean classification accuracy on the test data of the PA-best and A-best configurations for data sets *lsvt*, *christensen*, *gravier*, *eating*, *arcene*, and *chiaretti*.

values on new data because they are assessed on data that is not considered for tuning or model fitting.

To investigate whether multi-criteria tuning and selecting the PA-best configurations provides an advantage over single-criteria tuning, Figures 7.4 and 7.5 are analyzed. Triangles indicate configurations that are PA-best but that are not optimal when only considering the predictive accuracy. If the plots contain triangles located more upper-left than the dots and stars, this means that with multi-criteria tuning, it is possible to obtain configurations, that perform a more stable selection and select fewer features, than with single-criteria tuning. The potential loss in predictive accuracy, which choosing such a configuration instead of an A-best configuration invokes, can be assessed by comparing the colors of the respective dots, stars, and triangles. Even though the use of different stability measures leads to different sets of PA-best configurations, the conclusions regarding the benefit of multi-criteria tuning over single-criteria tuning are identical.

For data sets *sonar*, *kc1-binary*, *tecator*, *har*, *lsvt*, and *eating*, it is possible to choose PA-best configurations with higher classification accuracy, higher stability, and a lower number of selected features on the test data compared to the A-best configurations. For *gina_agnostic*, *gravier*, and *arcene*, with multi-criteria tuning, configurations can be found that lead to a more stable selection of fewer features at the cost of a noticeable loss in predictive accuracy compared to the configurations obtained with single-criteria tuning. For data sets *dilbert*, *christensen*, and *chiaretti*, there are several configurations that achieve the best classification accuracy on the training data. When considering the stability and number of selected features during tuning, it is possible to find configurations with a comparably stable feature selection of few features without any loss of predictive accuracy.

For data sets *tecator*, *har*, and *gina_agnostic*, differences between the results for the four stability measures can be observed. For the uncorrected measures SMJ and SMH, the PA-best configurations include configurations with many selected features. For the corrected measures SMU and SMA-Count, such configurations are not among the PA-best configurations.

7.4 Conclusions

For finding configurations with high predictive accuracy and with a stable selection of a small number of features, the strategy of considering all three target criteria simultaneously during hyperparameter tuning is analyzed. The Pareto optimal configurations with respect to the three criteria are determined and based on these, the PA-best configurations are selected. The PA-best configurations are defined in this thesis as the subset of the Pareto optimal configurations whose predictive accuracy is not lower than $acc.max - 0.05$ where $acc.max$ is the best predictive accuracy observed on the same data set.

Summarizing the results for all data sets, it can be concluded that with this strategy, it is possible to choose configurations with a more stable selection of fewer features without losing much predictive accuracy compared to model fitting only based on predictive performance. Depending on the data set, the losses in predictive accuracy range from noticeable over neglectable to gains in predictive accuracy. With the proposed strategy, models can be obtained that over-fit the training data less than the models obtained with single-criteria tuning only with respect to predictive accuracy.

The best results are obtained when the feature selection stability is evaluated with the stability measures SMU or SMA-Count. The other considered stability measures lack the theoretical property “correction for chance”, see Section 6.1, and therefore favor models with a large number of selected features for some data sets.

Chapter 8

Fitting Models on Data Sets with Similar Features

For data sets with similar features, feature selection is very challenging. The reason is that most established feature selection methods are not able to select only one feature out of a group of similar features. Instead, they select either several or none of the similar features. An example for this is given in Section 8.1. For data sets with similar features, also the evaluation of feature selection stability is more difficult. Unadjusted stability measures consider features that are almost identical as different features. Adjusted stability measures on the other hand take into account the similarities between the features but take more time for calculation, see Section 6.3.

In this chapter, strategies for fitting models on data sets with similar features are analyzed. When fitting a predictive model on a data set, the standard approach is tuning the hyperparameters only with respect to predictive accuracy. In Section 8.2, the proposed approach of considering the feature selection stability as a second target criterion during hyperparameter tuning is described. Also, competing approaches are explained. All approaches are compared based on both simulated data (Section 8.3) and real data (Section 8.4).

The aim of the analyses on simulated data is finding out whether the proposed approach allows fitting models that include all features that were used for target generation and no irrelevant or redundant features. On real data, the features that generate the target variable are unknown, but the performance of the models with respect to predictive accuracy and number of selected features can be examined. If the same or higher predictive accuracy is achieved with fewer features, this can be due to two reasons: 1. The additional features are irrelevant or redundant and therefore should not be selected. 2. All of the features in the larger set are relevant and not redundant, but the same information can be captured by a smaller set of different features. In both cases, the smaller set of features is preferable, for example with respect to domain specific interpretation of the features and cost for subsequent experimental analyses. In both the analyses on simulated data and on real data, it is investigated whether the feature selection stability must be assessed with an adjusted stability measure or if an unadjusted stability measure suffices for fitting models that achieve the respective goals.

8.1 Feature Selection on Data Sets with Similar features

Consider a data set with groups of similar, for example highly correlated, features. Many feature selection methods are not able to select only one feature out of a group of similar features. We have observed this behavior for various feature selection methods. For methods like univariate filter methods, this behavior is intuitive: highly similar features are likely to be assigned similar filter scores. For predictive methods that perform embedded feature

selection, the behavior is less intuitive. For lasso regression, it is even stated in the literature that this method would select only one feature out of a group of similar features, see for example Tolo̧si and Lengauer (2011). This, however, is not what we observed. We provide a small example demonstrating the feature selection behavior of several classification methods with embedded feature selection on data sets with similar features.

8.1.1 Experimental Setup

Data sets with 15 features and 100 observations are created in the following way. The values for the 15 features are drawn from the multivariate normal distribution $\mathcal{N}(0, M)$ using the R package *mvtnorm* (Genz and Bretz, 2009). The covariance matrix M is displayed in Figure 8.1. Then, the target is created as $Y = \text{sign}(X_1 + X_2 + E)$ with $E \sim \mathcal{N}(0, 0.25)$. X_1 is independent of all other features $X_2, \ldots, X_{15}$. X_2, X_3, and X_4 are perfectly positively correlated. X_5, X_6, X_7, and X_8 are correlated with X_2. The strengths of the correlations decrease from very high (X_5) to moderate (X_8). The features X_9 to X_{15} are noise features. So, X_3, and X_4 are redundant, X_9 to X_{15} are irrelevant and X_5 to X_8 are both redundant and irrelevant to some degree.

The classification methods glm boosting, lasso logistic regression, random forest and L_0-regularized logistic regression are analyzed. The implementations of glm boosting and random forest are taken from *mboost* (Hothorn et al., 2018) and *ranger* (Wright and Ziegler, 2017), respectively. For lasso logistic regression, the implementations in *glmnet* (Simon et al., 2011) and *LiblineaR* (Helleputte, 2017) are considered. The hyperparameters of all methods are tuned using 10-fold cross-validation. For glm boosting, lasso logistic regression and random forest, a random search with 50 iterations for the best hyperparameter values is performed. For L_0-regularized logistic regression, the implementation in *L0Learn* (Hazimeh and Mazumder, 2018) is used. This implementation has one integer hyperparameter with feasible values ranging from 1 to the number of features in the data set. Because the number of features in the data set is smaller than the number of tuning iterations for the other methods, all feasible values are tried here. For more details on the hyperparameter of *L0Learn* see Subsection 8.3.1. The best hyperparameter values are determined based on maximal classification accuracy. For each classification method, a model with the best hyperparameter

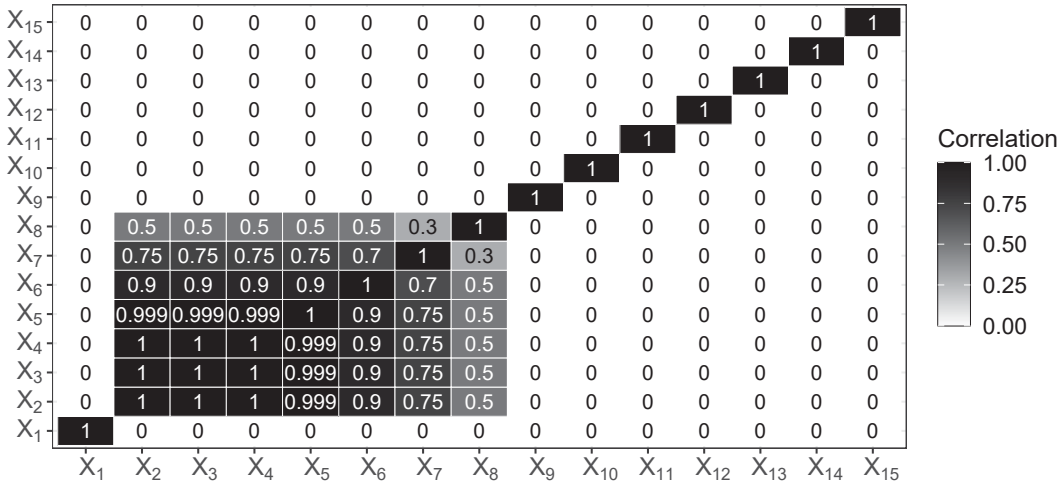


Figure 8.1: Covariance matrix for the 15 features. Note that the correlation matrix is identical to the covariance matrix because all features have unit variance.

values is fitted on the entire data set and the selected features are assessed. This procedure is repeated 1 000 times based on 1 000 data sets.

8.1.2 Results

Figure 8.2 displays for each feature, how many times it has been selected by each of the classification methods. The random forest models contain all features on all data sets. So, even though random forest can perform embedded feature selection, it does not exclude any of the features here, neither the irrelevant nor the redundant features. For the four other methods, irrelevant features are included only in some models. Also, the features X_6 to X_8 are selected only slightly more frequently than the irrelevant features. The two implementations of lasso logistic regression lead to different results. With *LiblineaR*, most models include all of the features X_2 to X_5. This means that there are redundant features in the resulting models. With *glmnet*, the selection frequencies decrease from X_2 to X_4. For both glm boosting and L_0-regularized logistic regression, the selection frequencies of X_2 to X_4 are quite similar among these features. Only a small decrease from X_2 to X_4 can be observed. The relevant feature X_1 which has no similar features is included in all models by all methods. Among the five methods, L_0-regularized logistic regression selects irrelevant features least frequently.

Figure 8.3 permits analyzing the redundancy in the models fitted with all methods. The first plot displays the number of features selected out of the group of perfectly positively correlated features X_2, X_3, and X_4. It shows that glm boosting, L_0-regularized logistic regression and lasso with *glmnet* on median select exactly one feature out of this group, which is optimal. Lasso with *LiblineaR* and random forest almost always select all three features. When analyzing the selection of $\{X_2, \ldots, X_5\}$, three perfectly and one highly correlated

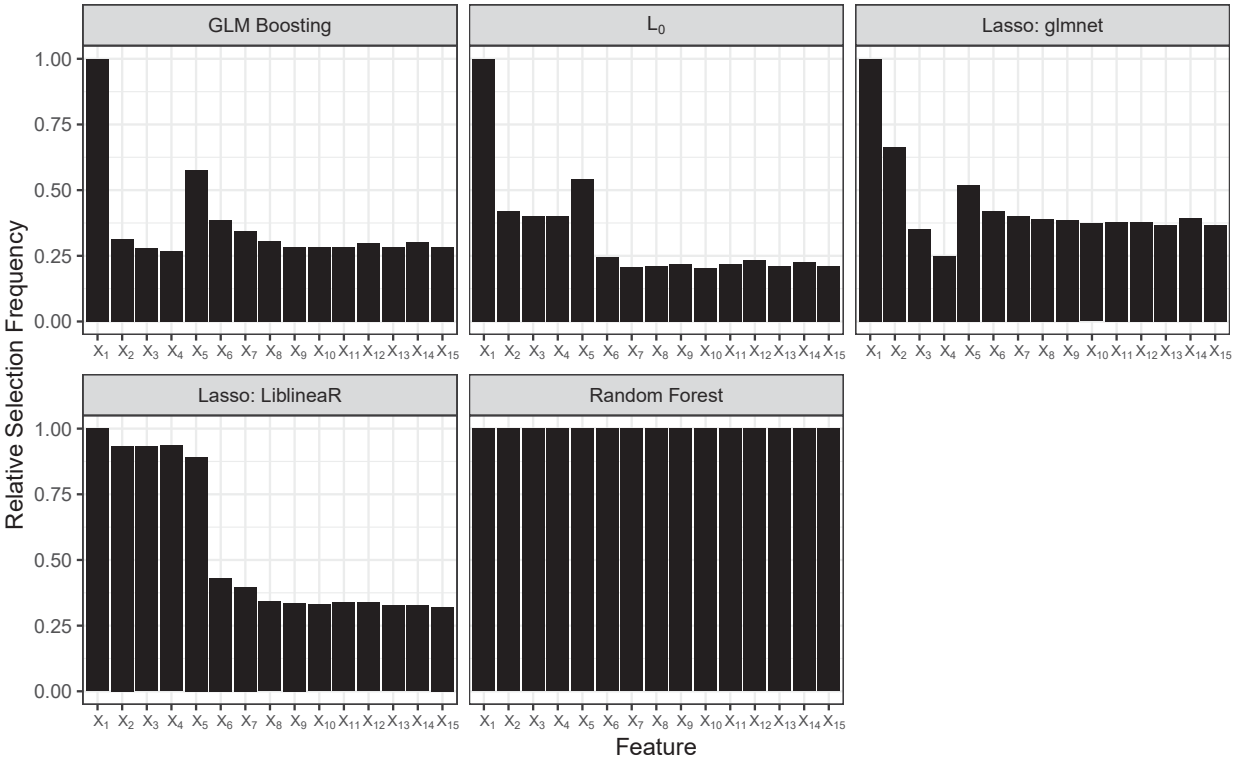


Figure 8.2: Selection frequencies of all 15 features divided by the number of data sets for all considered classification methods.

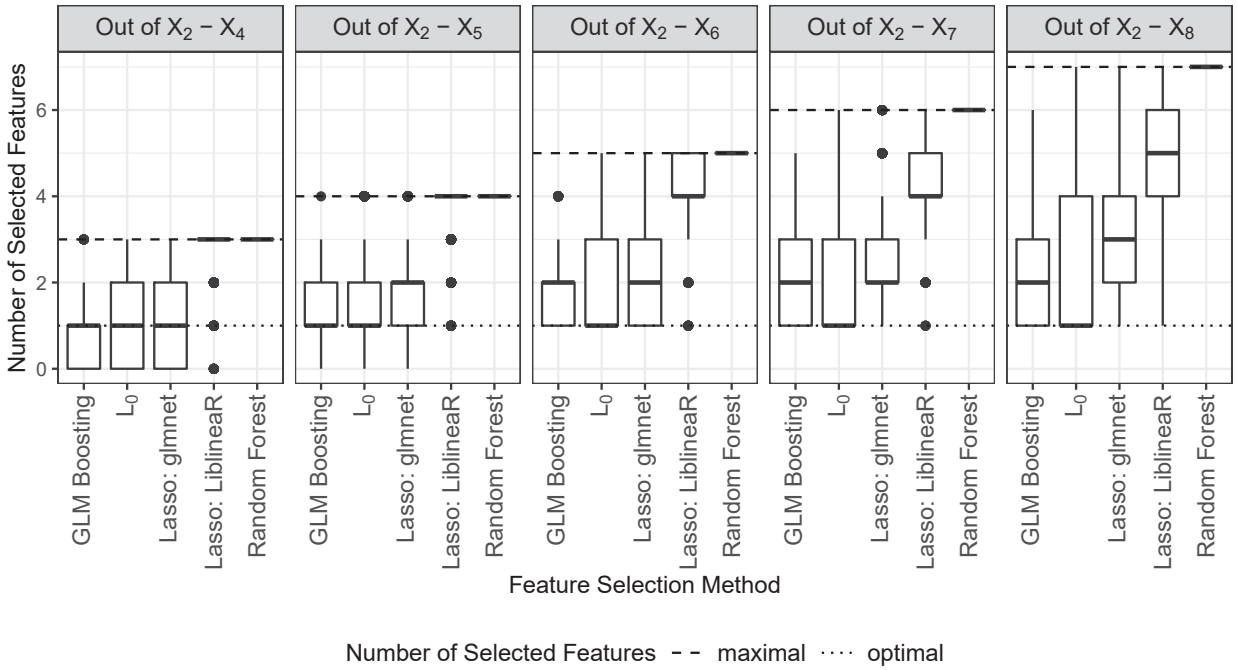


Figure 8.3: Number of selected features out of the sets $\{X_2, \ldots, X_i\}$ with $i \in \{4, \ldots, 8\}$. Optimally, only one of the features is selected per model. Maximally, all $i - 1$ features can be selected.

features are considered. In this scenario, glm boosting and L_0-regularized logistic regression still select only one of the features on median. Lasso with *glmnet* on median selects two of these features. Lasso with *LiblineaR* and random forest almost always select all of the features. In the three remaining plots, the features X_6, X_7 and X_8, which are decreasingly similar to X_1, are considered additionally. For all methods, the number of selected features out of $\{X_2, \ldots, X_i\}$ increases noticeably with increasing value of i.

Figure 8.4 allows comparing the five methods more easily. It displays the median number of selected features presented in Figure 8.3 for all classification methods in one plot. It can be observed that L_0-regularized logistic regression on median selects the optimal number, which is only one feature, out of all groups $\{X_2, \ldots, X_i\}$, $i \in \{4, \ldots, 8\}$. Glm boosting and lasso with *glmnet* select comparably few redundant features. Lasso with *LiblineaR* and random forest generate models with many redundant features. Concludingly, L_0-regularized logistic regression performs best at selecting only one feature out of a group of similar features. Therefore, in the analyses in this chapter, L_0-regularized logistic regression is used as classification and feature selection method.

8.2 Proposed Approach and Competing Approaches

The goal is finding sparse models with high predictive accuracy that include all relevant information for target prediction but no irrelevant or redundant features. As demonstrated in the previous section, L_0-regularized logistic regression is a method that is able to perform feature selection among similar features, that is, select only one feature out of a group of relevant and redundant features. Therefore, for classification data sets with similar features, we propose the following approach:

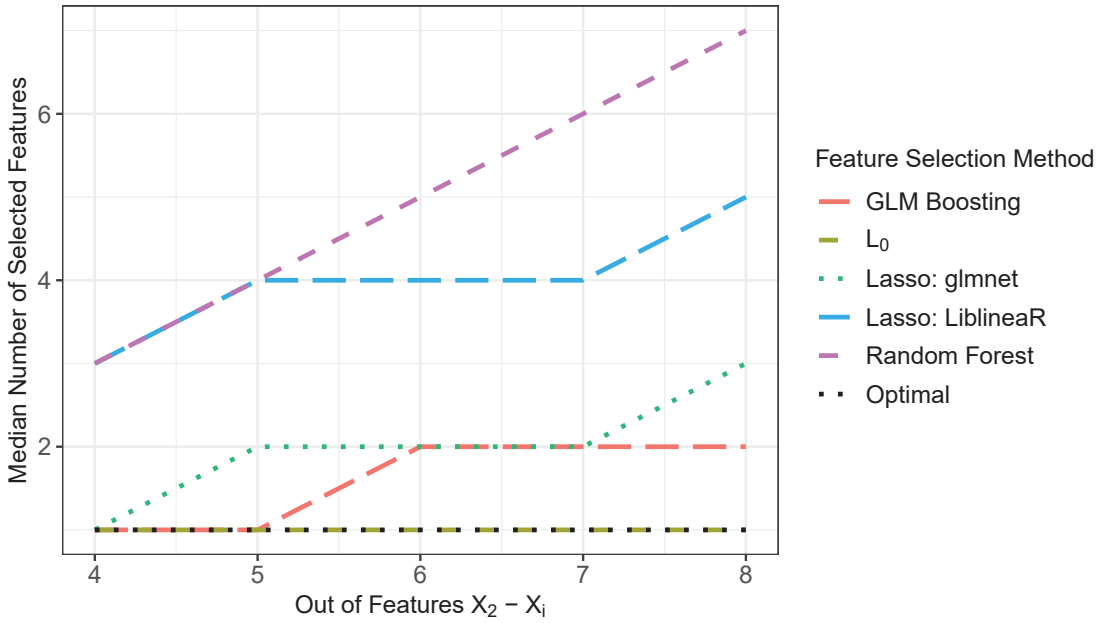


Figure 8.4: Median number of selected features out of the sets $\{X_2, \ldots, X_i\}$ with $i \in \{4, \ldots, 8\}$. Optimally, only one of the features is selected per model.

"adj": Use L_0-regularized logistic regression as predictive method and tune its hyperparameter with respect to predictive accuracy and to feature selection stability. This way, a set of Pareto optimal configurations is obtained. Choose the best configuration with ϵ-constraint selection, see Subsection 3.4.3. For assessing the stability of the feature selection during hyperparameter tuning, employ an adjusted stability measure.

The motivation for tuning the hyperparameter of L_0-regularized logistic regression this way is enhancing the standard approach of single-criteria tuning with respect to predictive accuracy. Configurations with a stable feature selection select almost the same features for all data sets (here: cross-validation splits). If the same features are selected for slightly varying data sets, these features are presumably relevant and not redundant. If the feature selection stability is low, many features are only included in some of the models. In this case, it is likely that these features are either redundant or do not carry much information for target prediction. Therefore, considering the feature selection stability during hyperparameter tuning should lead to models which include neither irrelevant nor redundant features. For data sets with similar features, it is expected that the selected features vary such that features with different identifiers but almost identical information are selected. This is taken into account when employing an adjusted stability measure. With ϵ-constraint selection, a configuration with high feature selection stability is chosen among the configurations that achieve high predictive accuracy. The proposed approach is compared to three other approaches:

"unadj": Proceed as in "adj", but employ an unadjusted stability measure instead of an adjusted measure.

"acc": L_0-regularized logistic regression with hyperparameter tuning only with respect to predictive accuracy. Single-criteria hyperparameter tuning with respect to predictive accuracy is a standard approach and serves as baseline. Either a single best configuration or a set of configurations with the same predictive accuracy on

the training data is obtained. In the latter case, one of these configurations is chosen at random.

"stabs": Perform feature selection with stability selection. Then fit an unregularized logistic regression model including the selected features. For stability selection, employ L_0-regularized logistic regression as feature selection method and tune the hyperparameters of stability selection with respect to predictive accuracy. Stability selection, see Subsection 3.3.3, is a popular framework for stable feature selection and a state-of-the-art method. It does not take into account the similarities between the features. With this approach, either a single best configuration or a set of configurations with the same predictive accuracy on the training data is obtained. In the latter case, one of these configurations is chosen at random.

8.3 Experimental Results on Simulated Data

First, the approaches are compared on simulated data. On simulated data, it is known which features have been used for creating the target variable and therefore should be selected and included in a predictive model. In the following, features with an absolute Pearson correlation of 0.9 or more are interpreted as exchangeable and they are called "similar features". In many fields, an absolute correlation of 0.9 or more is interpreted as a "strong" or even "very strong" association (Akoglu, 2018).

8.3.1 Experimental Setup

Data Sets The data sets are created in the following way. First, a covariance matrix Σ is defined. Two types of covariance matrices are considered. For matrix type "blocks", the features within a block all have correlation 0.95 to each other and 0.1 to features that are not in this block. For this matrix, the features within a block are interpreted as similar to each other. For each feature, the number of similar other features is given by the size of the block minus 1 (for excluding the feature itself as a similar feature). For the second matrix type, "Toeplitz", the covariance matrix is defined as $\Sigma_{ij} = \rho^{|i-j|}$ with $\rho = 0.9^{2/(\text{number of similar other features})}$. In the "Toeplitz" scenario, for the majority of features, the number of other features that have a correlation of at least 0.9 to the considered feature is given by "number of similar other features". This holds if "number of similar other features" is chosen as an even number. The "number of similar other features"/2 first and last features in the order of the covariance matrix are similar to less than "number of similar other features" other features. For both matrix types, all features have unit variance. Therefore, the covariance matrices are equal to the respective correlation matrices. Given the covariance matrix, the data is drawn from the multivariate normal distribution $\mathcal{N}(0, \Sigma)$ using the R package *mvtnorm* (Genz and Bretz, 2009). Then, five features $X_1, \ldots, X_5$ are chosen such that the absolute correlations between them are small. Then, the class variable Y_i is sampled from a Bernoulli distribution with probability $P(Y_i = 1) = \frac{\exp(\eta_i)}{1+\exp(\eta_i)}$ with $\eta_i = x_{1,i} + x_{2,i} + x_{3,i} + x_{4,i} + x_{5,i}$, for $i = 1, \ldots, n$. In total, $4 \cdot 2 \cdot 4 = 32$ simulation scenarios that differ with respect to data set dimensions, covariance matrix type, and number of similar features in the data set are considered. These scenarios are defined by all possible combinations of the values given in Table 8.1.

Setup for Tuning L_0-Regularized Logistic Regression L_0-regularized logistic regression (see Subsection 3.2.3) has only one hyperparameter that needs to be tuned: $\lambda \in \mathbb{R}_0^+$ balances the importance of a good fit in terms of maximum likelihood and the sparsity

Parameter	Values
Number of observations n × number of features p	100×200, $1\,000 \times 200$, $100 \times 2\,000$, $100 \times 10\,000$
Covariance matrix type	"Blocks", "Toeplitz"
Number of similar other features	0, 4, 14, 24

Table 8.1: Parameters for data set creation.

of the model. The implementation in the R package *L0Learn* (Hazimeh and Mazumder, 2018), which is used for the experiments, does not allow the user to tune the hyperparameter λ directly. Instead, one has to specify *max.feats*, the maximum number of features to be included in the model. The smaller the value of λ, the more features are selected. *L0Learn* uses an increasing sequence of λ-values and attempts to find a model with exactly *max.feats* features. Because *max.feats* is of type integer, a grid search for the best value is performed, trying all even values from 2 to 100. Note that the value *max.feats* = 5 is not tried, in order not to exploit the knowledge that exactly 5 features are used for generating the target variable. For continuous hyperparameters, a random search is more efficient than a grid search (Bergstra and Bengio, 2012), but when randomly drawing 50 integer values between 1 and 100, duplicate values are very likely. Ten-fold cross-validation is conducted to evaluate the performance of each hyperparameter value that is considered.

For "acc", the mean classification accuracy of the 10 models on the respective left-out data for the 10 cross-validation iterations is assessed. For "adj" and "unadj", the mean classification accuracy and the feature selection stability are evaluated. The feature selection stability is quantified based on the 10 sets of features that are included in the 10 models. In the "adj" approach, the stability measure SMA-Count is employed for stability assessment. For SMA-Count, the absolute Pearson correlation is used as a measure of feature similarity and the threshold θ is set to $\theta = 0.9$. For estimating the expected values, 10 000 replications are conducted, like it has been done in Section 6.3 as well. In the "unadj" approach, the stability measure SMU is employed. The two stability measures are chosen based on the analyses in Chapter 6.

Based on the performance values, the best configuration, that is, the best value for *max.feats*, is selected. For "acc" the configuration with highest cross-validated classification accuracy is chosen. If there are several configurations that attain the best classification accuracy, one of these configurations is chosen at random. For "adj" and "unadj", the best configuration is chosen based on ϵ-constraint selection, which is introduced in Subsection 3.4.3. For the cutoff values, *acc.const* = 0.025 and *stab.const* = 0.1 are used. These values have been determined in preliminary studies on other data sets.

Setup for Tuning Stability Selection Stability selection is combined with L_0-regularized logistic regression as feature selection method. The implementation of stability selection from the R package *stabs* (Hofner and Hothorn, 2017) is used with $B = 50$ complementary subsamples. The two real-valued hyperparameters PFER and π_{thr} have to be tuned. A random search with 50 iterations is performed to find the best PFER $\in [0.1, 10]$ and $\pi_{\text{thr}} \in [0.6, 0.9]$. The range for π_{thr} is suggested by Hofner and Hothorn (2017). The range for PFER is taken from Thomas et al. (2017) and slightly extended. To determine the best hyperparameter values of stability selection, an unregularized logistic regression model with the features obtained from stability selection is fitted. Its classification accuracy is evaluated using 10-fold cross-validation. The hyperparameter values of stability selection that lead to the model

with the highest cross-validated classification accuracy are chosen. If several values lead to the same best classification accuracy, one of these configurations is chosen at random. The hyperparameter of L_0-regularized logistic regression, *max.feats*, does not require tuning because it corresponds to the hyperparameter q of stability selection (see Subsection 3.2.3). Note that in the implementation of *L0Learn*, only the maximum number of features to select can specified, while q denotes the exact number of selected features. However, in most situations, *L0Learn* selects exactly the specified maximum number of features, not fewer features.

Evaluation For each approach, a final model is built based on the entire training data set. For "adj", "unadj", and "acc", a L_0-regularized logistic regression model with the best value of *max.feats* is fitted. For "stabs", stability selection is conducted with the best hyperparameter values. Then, an unregularized logistic regression model is built with the selected features. Additionally, an unregularized logistic regression model with the five features that were used for generating the target variable is fitted. This provides an upper bound for the predictive accuracy that can be achieved and will be denoted by "truth".

Based on the final models, three performance measures are calculated: the classification accuracy on new test data, the number of false positive features, and the number of false negative features. For evaluating the test accuracy, a new test data set of the same size is created in the same way as the training data set. Then, the classification accuracy of the final models is assessed. The number of false positive features is the number of irrelevant or redundant features that have been selected for the final models. The number of false negative features is the number of relevant and not redundant features that have not been selected for the final models. For the assessment of the number of false positive and false negative features, features with an absolute correlation of at least 0.9 are interpreted as exchangeable. So, if instead of a feature that was used for generating the target variable, an other feature with absolute correlation of at least 0.9 is selected, this other feature is accepted as well. More precisely, the quantity

$$\begin{aligned} a &= |V \cap C| + \min\{A(V, C), A(C, V)\} \quad \text{with} \\ A(V_i, V_j) &= |\{x \in (V_i \setminus V_j) : \exists y \in (V_j \setminus V_i) \text{ with } |\operatorname{Cor}(x, y)| \geq 0.9\}| \end{aligned}$$

as defined in Subsection 3.3.2.3 is calculated. V denotes the set of selected features and C denotes the set consisting of the five features used for target generation. Then,

$$\text{number of false positive features} = \text{number of selected features} - a$$

and

$$\text{number of false negative features} = 5 - a.$$

From the two performance measures, "number of false positive features" and "number of false negative features" the number of selected features can be calculated:

$$\text{n. o. selected features} = \text{n. o. false positive features} - \text{n. o. false negative features} + 5.$$

Note that the measure A, which is used for determining the number of false positive and false negative features, is also employed in the stability measure SMA-Count. Also, note that in the considered simulation scenarios, the data is generated from a logistic regression model and all approaches fit logistic regression models. In this case, the best subset of features with respect to predictive accuracy of the resulting model should consist exactly of the features

used for target creation. In situations where the data generation process and the models differ, the best subset of features generally depends on the model.

To ensure a fair comparison, all approaches use the same training and test data as well as the same cross-validation splits. For each simulation scenario, 50 training and test data sets are created.

8.3.2 Results

The results for the five approaches in the simulation scenarios with $n = 100$ observations and $p = 200$ features are displayed in Figure 8.5. In Appendix A.5, analogous plots for the scenarios with $n = 1\,000$ and $p = 200$ (Figure A.22), $n = 100$ and $p = 2\,000$ (Figure A.23), and $n = 100$ and $p = 10\,000$ (Figure A.24) are shown. The columns contain the results for the different numbers of similar other features per feature in the data set. For each approach, the classification accuracy on independent test data as well as the number of false positive and false negative features of the respective best configurations are shown. As 50 replications have been performed, each boxplot consists of 50 data points.

First, the scenarios with similar features are analyzed. In these scenarios, the predictive performances of the models obtained with "adj", "unadj", and "acc" are very similar. Also, in the scenarios with $p = 200$, the classification accuracies are quite close to the upper bound: the classification accuracy of a model that contains exactly the five features used for target generation. In the scenarios with $p = 2\,000$ and $p = 10\,000$, the predictive performances of the models resulting from the three approaches are noticeably lower than the upper bound, especially in the scenarios with few similar features.

Using the adjusted stability measure during tuning leads to much fewer false positive features compared to single-criteria tuning and to tuning using the unadjusted stability measure. So, the models obtained with "adj" contain fewer irrelevant or redundant features than the models resulting from "unadj" and "acc". This advantage comes at the small drawback of a slightly increased number of false negative features which, however, does not result in a decreased predictive performance. In the situations with $n = 100$ and $p = 200$, these observations can be made for all simulation scenarios with similar features. For $n = 1\,000$ and $p = 200$, this can only be observed if most features are similar to at least 14 other features. In the settings with $n = 100$ and $p = 2\,000$, 4 similar other features are necessary in the "blocks" scenarios and 14 in the "Toeplitz" scenarios. For $n = 100$ and $p = 10\,000$, only the scenario with 24 similar other features and matrix type "blocks" shows the advantage of the "adj" approach. It should be noted that in the high-dimensional simulation scenarios, the relative number of similar features is much lower than in the low-dimensional settings.

The stability selection approach performs worse than the other methods in terms of predictive accuracy in the scenarios with similar features. The low classification accuracy is due to too few relevant features being selected: the number of false negative features is high. The models obtained with "stabs" are very sparse. They contain almost no irrelevant or redundant features, but also not many relevant features.

When there are no similar features, all approaches lead to models with similar classification accuracy. With the stability selection approach, the resulting models contain the fewest irrelevant or redundant features among the compared approaches. In the scenarios without similar features, the proposed approach does not perform worse than the standard approach "acc", even though the proposed approach was specifically designed for situations with similar features. When there are no similar features, the approaches "adj" and "unadj" are identical because in this situation, the stability measures SMU and SMA-Count are identical.

Figure 8.6 and Figures A.25 to A.29 in Appendix A.5 show the influence of the simulation

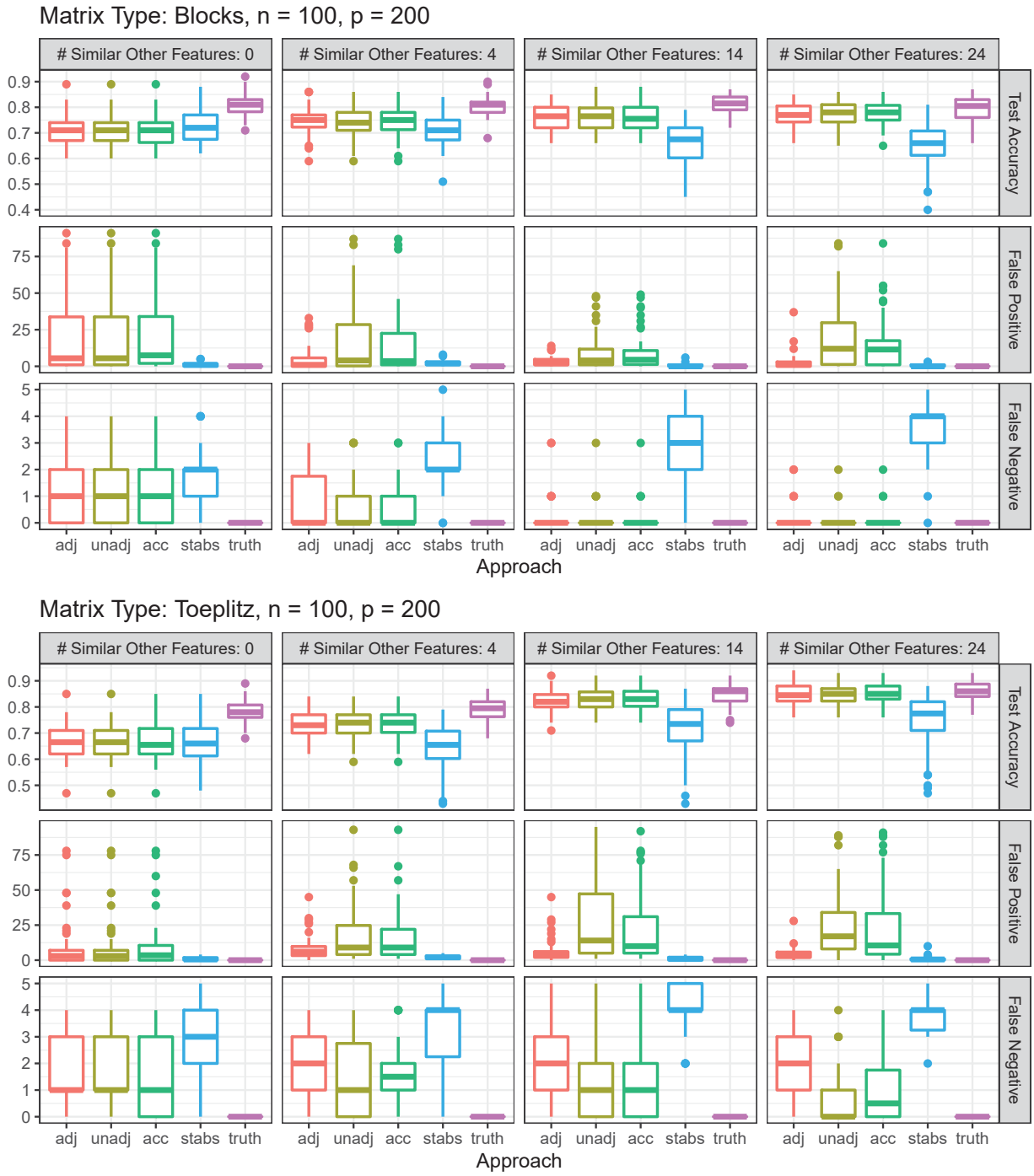


Figure 8.5: Results for the simulation scenarios with $n = 100$ and $p = 200$. "# Similar Other Features": Number of features in the data set that each feature is similar to (other than itself). Two features are interpreted as similar if their absolute correlation is at least 0.9. "Test Accuracy": classification accuracy on independent test data. "False Positive": number of irrelevant or redundant features that have been selected. "False Negative": number of relevant and not redundant features that have not been selected.

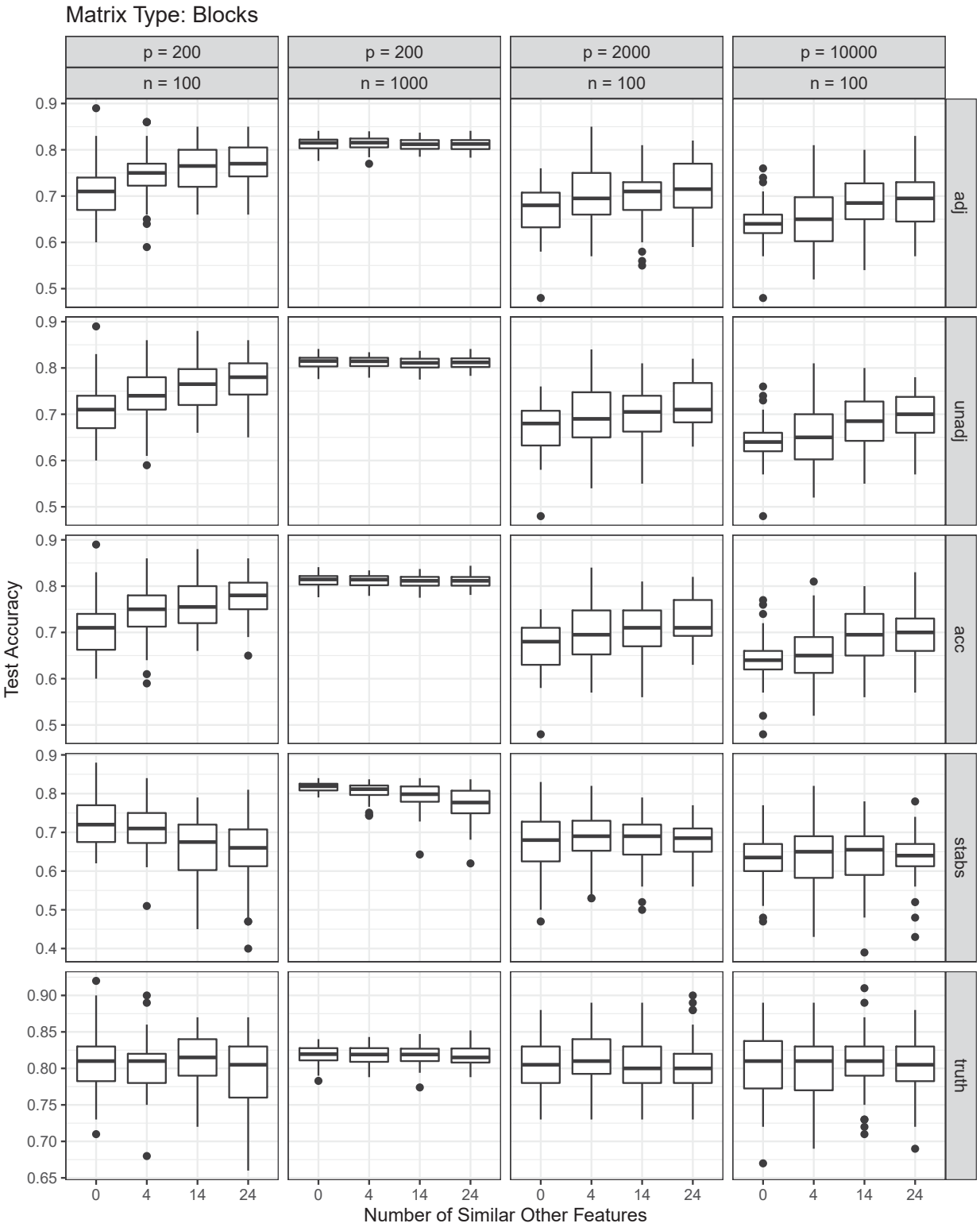


Figure 8.6: Test accuracy for all considered approaches in the simulation scenarios with matrix type "blocks".

scenarios on the performance of each approach. Figures 8.6 and A.25 display the classification accuracy on test data for all approaches and all simulation scenarios. In the scenarios with $p = 200$ and $n = 1\,000$, the highest predictive accuracy values are obtained, followed by the scenarios with $p = 200$ and $n = 100$. This is not surprising as the task of feature selection usually is easier if the total number of features is smaller.

When the "blocks" covariance matrix is used to generate the data, the predictive accuracy of the "truth" approach is independent of the number of similar features in the data set. This seems intuitive as this approach only considers the features used for generating the target variable. However, in the "Toeplitz" scenarios with $p = 200$, the predictive accuracy of "truth" increases with the number of similar features. The reason is that the 5 features chosen for target creation are correlated comparably strong. If each feature is supposed to be similar to 14 or 24 other features in a data set with $p = 200$ features, it simply is not possible in the "Toeplitz" scenario to find 5 features that are only weakly correlated. If the features used for target generation are positively correlated, most of the probabilities $P(Y_i = 1)$ are close to 0 or 1. Therefore, the classification tasks are easier than in the other scenarios.

For the approaches "adj", "unadj", and "acc" and all settings with $p > n$, the predictive accuracy is the higher, the more similar features there are. The increase in predictive accuracy is stronger in the "Toeplitz" scenarios than in the "blocks" scenarios. The larger the "number of similar other features", the more features are similar to the features used for target generation. So, it becomes easier to select features with information for target prediction, because there are more of them. In the "Toeplitz" scenarios, this effect is stronger than in the "blocks" scenarios. In all scenarios, for each relevant feature, there are "number of similar other features" features that contain almost the same information. For matrix type "blocks", the remaining features provide almost no information for target prediction. In the "Toeplitz" scenarios, the remaining features contain information for target prediction. In order for two features to be seen as similar, their absolute correlation must at least have the value 0.9. But even if a feature is not considered to be similar to any of the features used for target generation, the feature can still contain a lot of information for target prediction.

The predictive accuracy of the "stabs" approach depends on the number of similar features in the settings with $p = 200$. For matrix type "blocks", the predictive accuracy decreases with an increasing number of similar features. This is due to the increasing number of false negative features for an increasing number of similar features, see Figures A.28 and A.29. The omission of many relevant and not redundant features results in missing relevant information in the predictive model and therefore in a low predictive accuracy. For matrix type "Toeplitz", the accuracy increases with the number of similar features. As discussed for "truth", the classification tasks are easier in the scenarios with $p = 200$ and many similar other features per feature. In the scenarios with $p = 2\,000$ and $p = 10\,000$, the classification accuracy of "stabs" is more or less constant with respect to the number of similar features in the data set. This is because the number of false negative features is almost constant in these scenarios, see Figures A.28 and A.29.

Figures A.26 and A.27 display the effect of the simulation scenarios on the number of false positive features. These figures do not contain results for "truth" because these models include exactly the features used for target generation by definition. The more features there are in the data set, the more false positive features are selected. This is intuitive because the feature selection task becomes harder with increasing number of features in the data set. In the settings with $n > p$, comparably few false positive features are selected. In these scenarios, there are more observations which makes it easier to identify the relevant and not redundant features.

In all scenarios, for "adj", the number of false positive features decreases with an increasing number of similar features. The greater the number of similar other features per feature, the fewer irrelevant and the more redundant features there are in the data set. So, it can be concluded that with "adj", especially the selection of redundant features is prevented. For "unadj" and "acc", an analogous decrease in the number of false positive features can only be observed in the settings with $p = 10\,000$ and $n = 100$. For "stabs" as well as for "unadj" and "acc" in the other scenarios, no monotonic effect of the number of similar features can be detected.

Figures A.28 and A.29 show the effect of the simulation scenarios on the number of false negative features. Here, the number of features in the data set only has a small effect. For the data sets with 10 000 features, the number of relevant features that have not been selected is slightly larger than for the other scenarios with the same number of observations. In the $n > p$ settings, almost no relevant features are omitted by all approaches. As discussed before, the reason is that more observations allow a better detection of the relevant features.

For "adj", "unadj", and "acc", in almost all scenarios, the number of false negative features decreases with an increasing number of similar features in the data set. As described before, the greater the number of similar other features per feature, the more relevant features there are in the data set. So, it is more likely that the methods succeed at selecting relevant features. For "stabs", the number of false negative features increases with the number of similar features in the scenarios with $p = 200$. Recall that L_0-regularized logistic regression, which is used as the feature selection method in the "stabs" approach, usually selects only one feature out of a group of similar features. When repeatedly performing feature selection on the subsamples, it is likely that each time, only one feature out of each group of similar and relevant features is selected and that the selection frequencies within a group are fairly equal. So, if there are many similar features, the selection frequencies become very small. If the highest of the selection frequencies is below any reasonable value of the threshold π_{thr}, none of the features is included in the final model. In the scenarios with $p = 2\,000$ and $p = 10\,000$, the number of false negative features is almost constant with respect to the number of similar features. Independent of the number of similar features in the data set, comparably many relevant and not redundant features are not included in the final models. So, the stability selection approach generally seems not to work well for data sets with many features, even if there are no similar features among them.

Figure 8.7 shows the run times for all simulation scenarios and all approaches. In all scenarios, the run times of the stability selection approach are much longer than the run times of all competing approaches. Also, it can be observed that calculating the unadjusted stability measure is very fast. So, tuning with respect to classification accuracy and stability assessed by the unadjusted stability measure is about as fast as tuning only with respect to classification accuracy. Calculating the adjusted stability measure takes a bit longer. But it is faster than the stability selection approach by orders of magnitude.

8.4 Experimental Results on Real Data

Now, these findings are validated based on the 12 real data sets given in Section 4.1. Whenever possible, the analyses here are conducted analogously to the analyses in the previous section. However, on real data sets, the truly relevant features are not known.

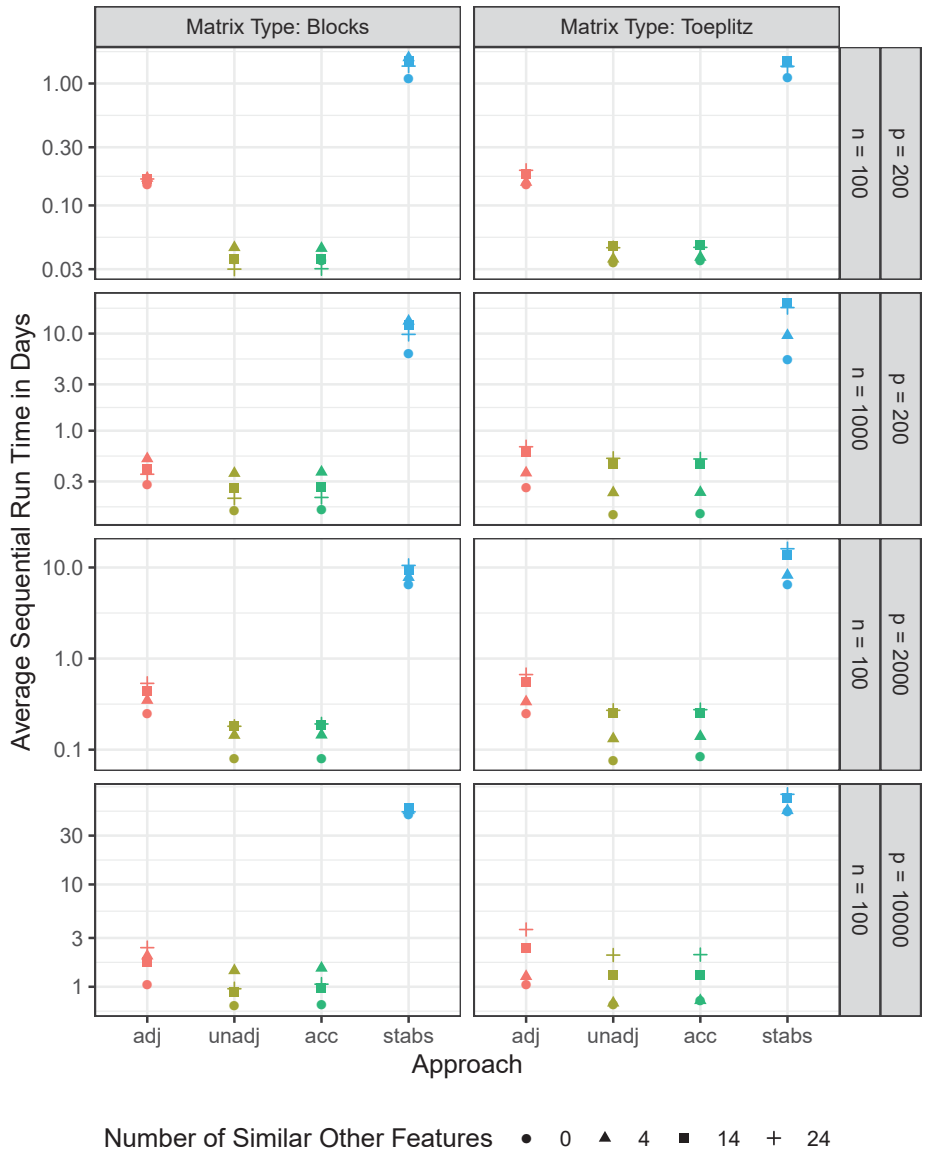


Figure 8.7: Average sequential run times for all considered simulations scenarios and all considered approaches. Each symbol represents the average sequential run time over the 50 replications that have been conducted for the respective scenario and approach. The sequential run time is the sum of the run times of the 50 iterations for hyperparameter tuning. Note that the y-axis is scaled logarithmically.

8.4.1 Experimental Setup

In this study, the same approaches are compared as in the simulation study. As far as possible, tuning and evaluation is performed in the same way. Because it is not possible here to generate independent test data sets, nested cross-validation with 10 inner and 10 outer iterations is used (Bischl et al., 2012). The inner iterations are used to determine the best configurations and the left-out data of the outer iterations is used to evaluate the predictive accuracy. So, for each approach and each data set, 10 predictive performance values are obtained. In the simulation study, an upper bound for the classification accuracy is obtained with a logistic regression model with the features used for target generation. This is not possible here because the truly relevant features are not known here. For the same reason, it is not possible to assess the number of false positive and false negative features. Instead, the

number of features included in the 10 models, whose predictive accuracy is evaluated on the left-out test data, is recorded.

8.4.2 Results

Figures 8.8 and 8.9 display the evaluation results of the best configurations on real data. Figure 8.8 shows the classification accuracy of the best configurations. Figure 8.9 displays the number of features that are selected for these configurations. As nested cross-validation with 10 outer iterations has been performed, each boxplot consists of 10 data points.

For data sets *sonar*, *tecator*, *har*, *dilbert*, *lsvt*, *christensen*, *arcene*, and *chiaretti*, it is beneficial to perform multi-criteria tuning with respect to both classification accuracy and stability and choosing the best configuration based on ϵ-constraint selection, compared to single-criteria tuning only with respect to classification accuracy. A comparable or even better predictive performance is achieved with multi-criteria tuning and the fitted models include fewer features. Among these data sets, for *tecator*, *har*, *dilbert*, *lsvt*, and *arcene*, the adjusted stability measure must be used for achieving this benefit. These data sets contain many similar features. For data sets *sonar*, *christensen*, and *chiaretti*, which contain only few similar features, the unadjusted stability measure is sufficient. For data sets *kc1-binary*, *gina_agnostic*, *gravier*, and *eating*, multi-criteria tuning does not provide a benefit over single-criteria tuning. Still, for these data sets considering the feature selection stability during tuning does not decrease the predictive performance or increase the number of selected features of the resulting models in comparison to single-criteria tuning.

Comparing the results of the proposed approach and stability selection, the proposed approach performs better on the majority of data sets. For data sets *sonar*, *har*, *gina_agnostic*, *lsvt*, *gravier*, *eating*, and *arcene*, stability selection leads to a worse predictive accuracy than the other approaches. For data sets *christensen* and *chiaretti*, it fails at excluding irrelevant or redundant features. Only for data set *tecator*, models with higher predictive accuracy are obtained with stability selection compared to the other approaches. For *kc1-binary* and *dilbert*, more sparse models with the same predictive quality can be fitted with the stability selection approach. On most data sets, the stability selection approach leads to comparably sparse models, often at the expense of a comparably low predictive accuracy. This has been observed on simulated data as well.

8.5 Conclusions

L_0-regularized logistic regression is a classification method with embedded feature selection that - in contrast to many state-of-the-art feature selection methods - is able to select only one feature out of a group of similar features in a data set. The approach of tuning its hyperparameter in a multi-criteria fashion with respect to predictive accuracy and feature selection stability is proposed and evaluated. It is compared to the standard approach of single-criteria tuning of the hyperparameter as well as to the state-of-the-art technique stability selection with L_0-regularized logistic regression as feature selection method.

On simulated data, especially in the scenarios with many similar features, tuning the hyperparameter of L_0-regularized logistic regression with respect to both predictive accuracy and stability is beneficial for avoiding the selection of irrelevant or redundant features compared to single-criteria tuning. To obtain this benefit, the feature selection stability must be assessed with an adjusted stability measure. Measuring the stability with an unadjusted measure does not outperform single-criteria tuning in most scenarios. Also, considering the stability during tuning does not decrease the predictive accuracy of the resulting models.

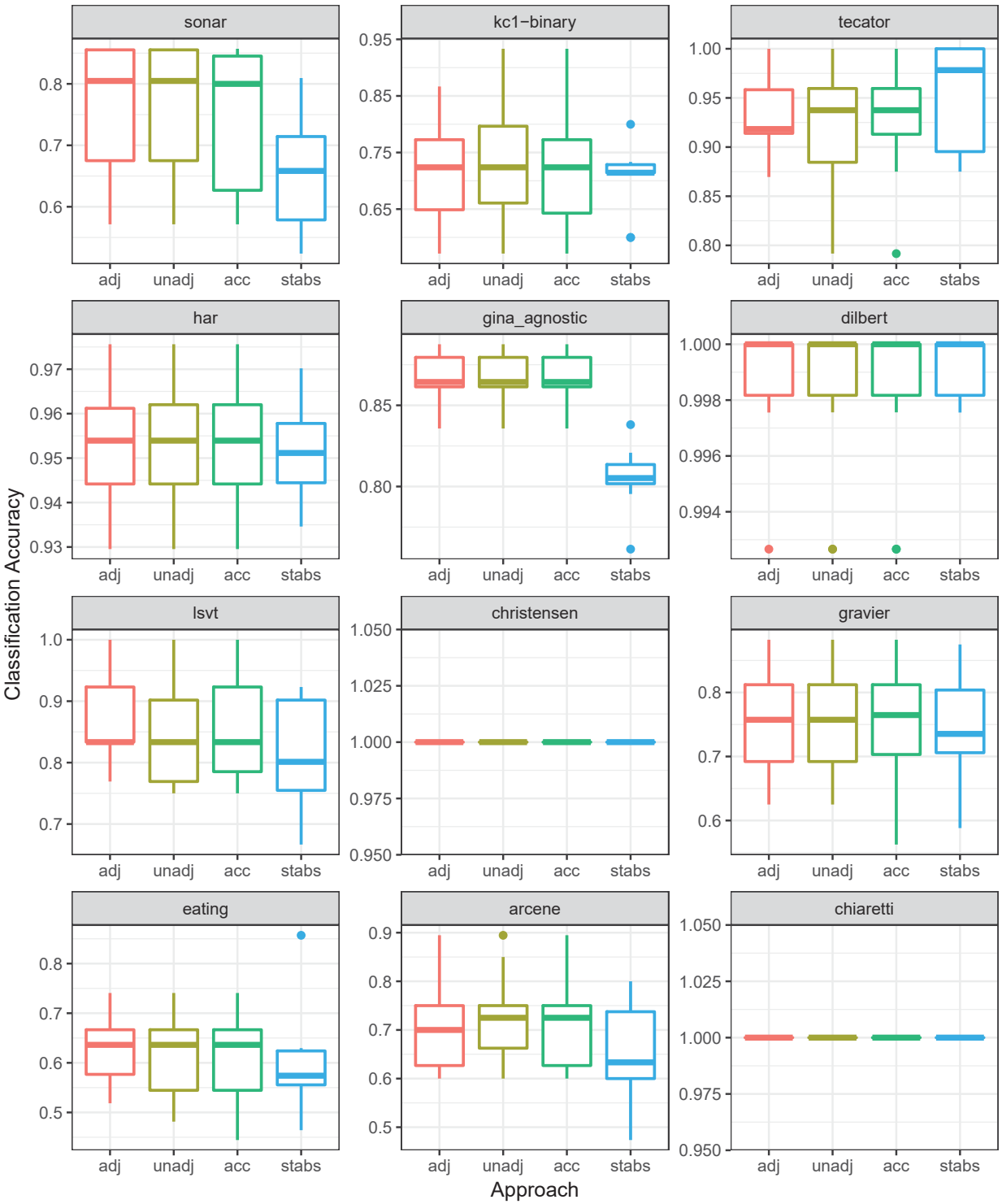


Figure 8.8: Comparison of the approaches on real data. Classification accuracy of the best configurations on the left-out test data of the outer cross-validation iterations.

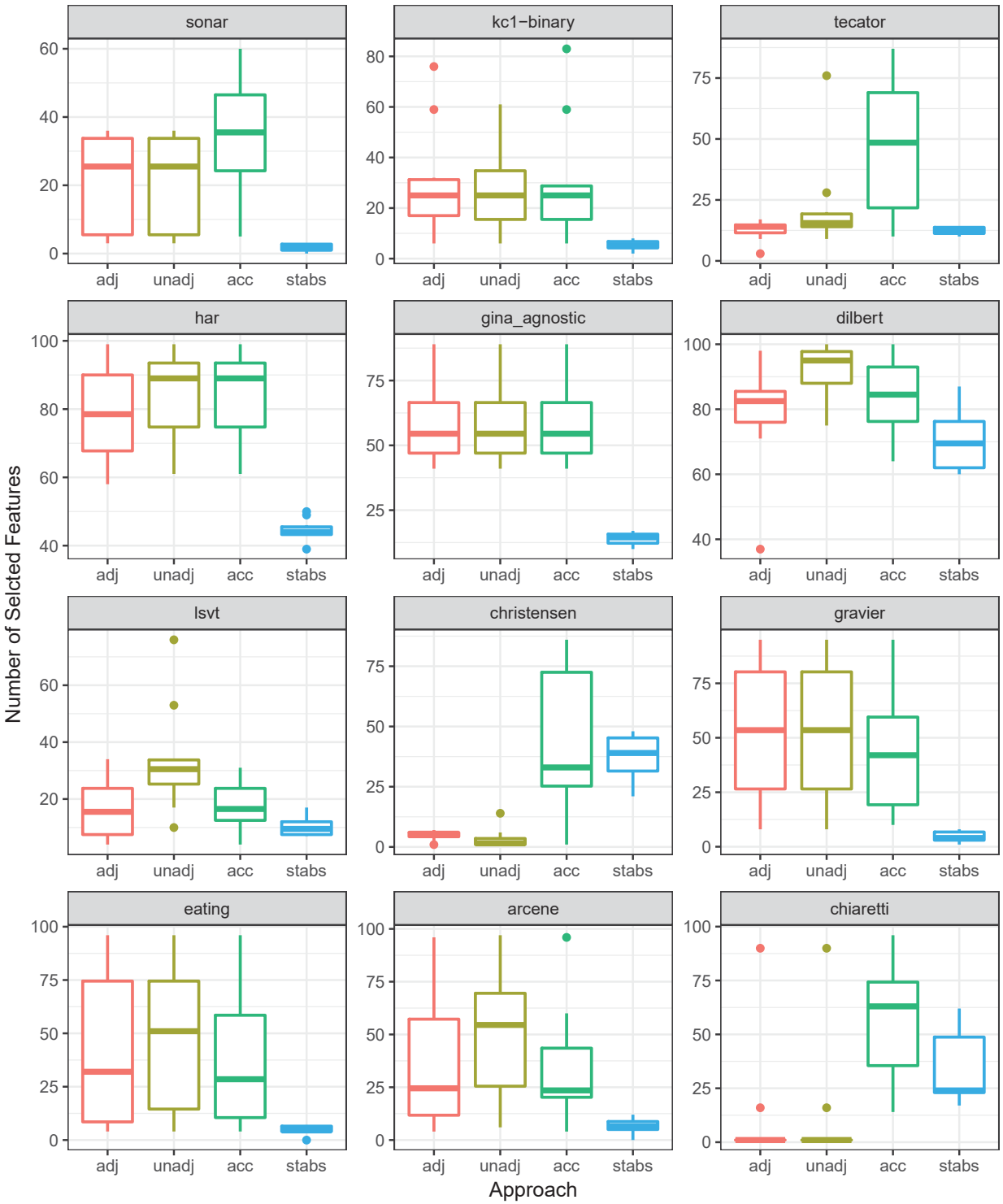


Figure 8.9: Comparison of the approaches on real data. Number of selected features of the best configurations per outer cross-validation iteration.

On real data, performing hyperparameter tuning with respect to both predictive accuracy and feature selection stability is beneficial for fitting models with fewer features without losing predictive accuracy compared to single-criteria tuning. Possibly, with the proposed approach, fewer irrelevant or redundant features are selected, but this cannot be known for sure. For data sets with many similar features, an adjusted stability measure must be used to obtain this benefit. For data sets with only few similar features, an unadjusted stability measure suffices and is faster to compute. For all data sets, no predictive accuracy is lost by additionally considering the stability.

Compared to the stability selection approach, models with higher predictive accuracy are fitted with the proposed approach, especially in simulation scenarios with many similar features. On real data sets, the proposed approach outperforms stability selection on many of the data sets. Both on simulated data and on real data, with the stability selection approach, comparably sparse models are fitted. These models, however, often do not include enough relevant features and therefore obtain a comparably low predictive accuracy. Also, the stability selection approach takes much more time for computing. For large data sets, it is infeasible without a high performance compute cluster.

Chapter 9

Summary, Discussion, and Outlook

Feature selection is a key part of data analysis, machine learning, and data mining. Often it is advantageous with respect to predictive performance, run time, and interpretability to disregard the irrelevant and redundant features by choosing a suitable subset of features. Especially in such domains where the selected features are subject to further costly analyses, it is important to select a subset of features such that all relevant information is captured while avoiding the selection of irrelevant or redundant features.

In this thesis, four aspects connected to feature selection were analyzed: Firstly, a benchmark of filter methods for feature selection was conducted. Secondly, measures for the assessment of feature selection stability were compared both theoretically and empirically. Thirdly, a multi-criteria approach for obtaining desirable models with respect to predictive accuracy, feature selection stability, and sparsity was proposed and evaluated. Fourthly, an approach for finding desirable models for data sets with many similar features was suggested and evaluated. All analyses that were conducted on real data, were based on the same 12 classification data sets. These data sets come from various domains and differ in size as well as in the similarity structure between the features.

For the benchmark of filter methods, 20 filter methods were analyzed. The analyses were based on the analyses in Bommert et al. (2020). First, the filter methods were compared with respect to the order in which they rank the features and with respect to their scaling behavior in order to identify groups of similar filter methods. Especially for the data sets that contain a large number of features, there were four groups of filter methods that rank the features in a similar way and some filter methods that were not similar to any other method. For the other data sets, most filter methods were very similar. The filters that were similar to each other mostly came from the same toolboxes. Regarding the scaling behavior of the filter methods, there were two groups of filters with similar behaviors: the filter methods that calculate all scores at the same time and the methods that calculate the scores iteratively.

Next, the predictive accuracy of the filter methods when combined with a predictive model and the run time needed for feature selection as well as for building a good predictive model based on the selected features were analyzed. There was no subset of filter methods that performed better than the rest of the filter methods on all data sets. Instead, the best filter methods differed between the data sets. Also, the differences in performance between the filter methods on the same data set were not large for many data sets, compared to the observed variation. Regarding the aim of choosing a subset of filter methods that perform well for most data sets, the mutual information based filter methods *JMI* and *MIM* as well as the random forest importance filter *impurity* seem advisable. For *JMI*, the best predictive accuracies and for *MIM*, the lowest run times among all filter methods were observed across data sets. When searching for a good filter method for a new data set, it appears reasonable to limit the search space to *JMI*, *MIM*, and *impurity*.

The analyses could be extended by additionally considering other feature selection methods apart from pure filter methods, which was out of scope for this analysis. Also, here, tuning

was performed with respect to predictive performance and then the best configurations not only with respect to predictive performance but also with respect to run time were analyzed. In future analyses, one could perform multi-criteria tuning with respect to both performance criteria at the drawback of a much more complicated aggregation of the results. Another interesting aspect of research would be discovering data set characteristics that indicate which filter and classification methods perform best. If such characteristics were available, much run time could be saved in the tuning process. Ideally, the characteristics should be cheap to compute in order to obtain a benefit in run time over trying different methods. In the field of optimization, exploratory landscape analysis (Kerschke and Trautmann, 2019) is an approach for defining characteristics of objective functions in order to automatically select the best optimization algorithm. In future research, this approach could be transferred to model selection.

To find suitable measures for stability assessment, 20 stability measures were compared based on both theoretical properties and on their empirical behavior. Five of the measures were newly proposed by us. Regarding the theoretical comparison, it was assessed which of the desirable criteria for stability measures defined in Nogueira (2018) are fulfilled by the stability measures. Additionally, new desirable properties were defined and investigated in this thesis. For the empirical comparison, two scenarios were considered: In the first scenario, all possible combinations of sets of selected features were analyzed for a very small number of features. In the second scenario, feature sets selected from real data sets were investigated. In both scenarios, groups of similar stability measures were identified and the impact of the number of selected features on the stability values was studied. Additionally, in the scenario with the real feature sets, the run times for calculating the stability measures were analyzed. These analyses are extensions of the analyses in Bommert et al. (2017) and Bommert and Rahnenführer (2020).

When analyzing the stability assessment behavior of the stability measures, it was investigated whether the measures consider the same feature sets as stable or unstable. Based on this behavior, four homogeneous groups of stability measures could be identified. One group was formed by stability measures that do not possess the theoretical property "correction for chance". These measures take the higher values, indicating a more stable feature selection, the more features are selected. The second group consisted of stability measures that fulfill this property but that do not take into account similarities between the features. The third group was created by stability measures that both are corrected for chance and consider similarities between features. These measures were named "adjusted" (for similar features). The fourth group existed only for the analysis on real feature sets and only contained one uncorrected stability measure that - in contrast to the other uncorrected stability measures - is not able to take small values for small numbers of selected features. The stability measures that are corrected for chance do not have restrictions on attainable values. The run times for most stability measures were almost neglectable. For the adjusted and corrected measures however, the run times were quite long, especially for large data sets with many similar features.

For stability assessment in future analyses, the stability measures SMU, SMA-Count, SMJ, and SMH were chosen as representatives of the four groups, based on their properties and on run time considerations. SMU, the representative of the group of corrected but unadjusted measures, and SMA-Count, the representative of the group of corrected and adjusted measures, are suitable measures for stability assessment. When employing the uncorrected measures SMJ and SMH, it should be kept in mind that they do not fulfill the important theoretical property "correction for chance".

When searching for a good predictive model for a given data set, the predictive accuracy is usually the only criterion considered in the model finding process. In this thesis, the benefits of not only considering the predictive accuracy but also the feature selection stability and the number of selected features were investigated. To find desirable configurations with respect to all three performance criteria, the hyperparameter tuning of combined methods consisting of a filter and a classification method was performed in a multi-criteria fashion: the subset of Pareto optimal configurations, whose predictive accuracy is at most 0.05 below the accuracy of the configuration with highest accuracy obtained for the same data set, was determined. This way, it could be investigated whether it is possible to find configurations that perform a more stable selection of fewer features without losing much predictive accuracy compared to model fitting only considering the predictive performance. These analyses are extensions of the analyses in Bommert et al. (2017).

It could be concluded that with multi-criteria tuning, it is possible to choose such configurations. The resulting Pareto fronts gave several options to choose sparse and stable configurations, which at the same time had high predictive accuracy. The losses in predictive accuracy that went along with the gains in stability and sparsity were different between the data sets and ranged from noticeable over neglectable to gains in predictive accuracy. Also, with multi-criteria tuning, models were obtained that over-fit the training data less than the models obtained with single-criteria tuning only with respect to predictive accuracy. The configurations that were obtained with multi-criteria tuning differed a bit when different stability measures were used. Nevertheless, the conclusions about the benefits of multi-criteria tuning applied to all of the considered stability measures. All in all, the multi-criteria approach of considering predictive accuracy, stability, and sparsity jointly during hyperparameter tuning proved to be advantageous.

In future work, model based optimization could be used instead of a random search for finding desirable configurations more efficiently. Also, the entire sets of Pareto optimal configurations could be analyzed, accepting the drawback of visualization difficulties. Moreover, it would be interesting to analyze whether configurations obtained with the multi-criteria approach select the same features on entirely new data sets from the same application instead of just considering test data sets created using resampling.

For data sets with many similar, for example highly correlated features, feature selection is especially challenging. Most established feature selection methods are not able to select only one feature out of a group of similar features. They select either several or none of them. L_0-regularized logistic regression is a method that is able to perform such a feature selection. Therefore, for data sets with similar features, we proposed the approach of using L_0-regularized logistic regression and tuning its hyperparameter in a multi-criteria fashion with respect to both predictive accuracy and feature selection stability. The idea was that considering also the feature selection stability works as a regularization, avoiding the selection of irrelevant or redundant features. We suggested to assess the stability with an adjusted stability measure, that is, a stability measure that takes into account similarities between features. For selecting the best configuration, the proposed algorithm ϵ-constraint selection was employed. The approach was evaluated based on both simulated and real data sets. It was compared to the standard approach of L_0-regularized logistic regression with hyperparameter tuning only with respect to predictive accuracy, to L_0-regularized logistic regression with hyperparameter tuning with respect to predictive accuracy and stability assessed with an unadjusted stability measure, and to the state-of-the-art method stability selection with L_0-regularized logistic regression as feature selection method.

Based on simulated data, it was observed that the proposed approach achieved the same or better predictive performance compared to the two established approaches. Considering

the stability during tuning did not decrease the predictive accuracy of the resulting models. The proposed approach succeeded at selecting the relevant features while avoiding irrelevant or redundant features. With the proposed approach, much fewer irrelevant or redundant features were selected than with the other approaches. Especially in situations with many similar features, the proposed approach outperformed the competing approaches. The single-criteria approach failed at avoiding irrelevant or redundant features and the stability selection approach failed at selecting enough relevant features for achieving acceptable predictive accuracy.

On real data, performing hyperparameter tuning with respect to both predictive accuracy and stability was beneficial for fitting models with fewer features without losing predictive accuracy compared to single-criteria tuning. For data sets with many similar features, it was necessary to use an adjusted stability measure for obtaining this benefit. For data sets with very few similar features, it was still beneficial to consider the stability during tuning, but for these data sets, an unadjusted stability measure was sufficient and faster to compute. By additionally considering the stability, no predictive accuracy was lost and for many data sets, the resulting models were more sparse. The stability selection approach was outperformed by the proposed approach on many of the data sets. With the stability selection approach, comparably sparse models were fitted. These models, however, often did not include enough relevant features and therefore obtained a comparably low predictive accuracy. Also, the proposed approach could easily be applied to all of the considered data sets while the stability selection approach would have been infeasible without a high performance compute cluster for the larger data sets because of its enormous run time.

In future analyses, more simulation scenarios could be investigated. In this thesis, the data was drawn from a multivariate normal distribution with a specified correlation matrix. This could have biased the results towards the proposed approach because for the adjusted stability measure, the similarity between features was quantified using the absolute Pearson correlation. Likewise, other measures for quantifying the similarity between features could be employed. Also, in this thesis, the threshold for identifying similar features was set to a fixed value. In further analyses, the threshold could be considered as an additional hyperparameter and hence be tuned as well. Moreover, other feature selection methods that are able to select only one feature out of a group of similar features could be considered. These further methods could include other L_0-regularized methods like L_0-regularized support vector machines or L_0-regularized boosting methods. Additionally, other feature selection methods, like a sequential forward search taking into account similarities between features could be employed. Furthermore, the number of features that are included in the model could be considered as a third criterion during hyperparameter tuning. The tuning of integer hyperparameters could also be conducted with a random search with sampling without replacement instead of a grid search. Another aspect of research would be modifying stability selection in such a way that it also takes into account similarities between features.

All of the analyses in this thesis were based on classification data sets. In future analyses, also regression or survival data sets could be considered. Another aspect for further research could be the connection between the stability of the feature selection and the similarity of the data sets on which the feature selection is performed. As pointed out by Alelyani et al. (2011), the more similar the data sets are, the more stable the feature selection is expected to be. To investigate this connection, a suitable method for quantifying the similarity of potentially high-dimensional data sets is needed. Suggestions for methods for assessing the similarity of data sets include the comparison of the data densities by estimating them based on classification trees (Ntoutsi et al., 2008) or probability binning (Roederer et al., 2001) and calculating a similarity measure of the estimated densities (Cha, 2007). Other proposals are

the assessment of the agreement of predictive models fitted on the data sets (Ganti et al., 1999) and the analysis if the same predicitve models perform best (Leite et al., 2012). To deal with high-dimensionality, a dimension reduction technique like calculating principal components could be applied beforehand.

Summarizing the contributions of this thesis, new stability measures and properties for stability measures were defined and analyzed, two extensive comparative studies were conducted, and two strategies for finding desirable predictive models were proposed and evaluated thoroughly. In this thesis, new stability measures and new theoretical properties for identifying suitable stability measures were introduced and investigated thoroughly. With the adjusted stability measures proposed in this thesis, the need for adjusted stability measures that do not lack important theoretical properties is met. Regarding the comparative studies, both filter methods for feature selection and measures for the assessment of feature selection stability were analyzed. These analyses did not only result in recommendations on which of the methods or measures to use in future analyses but also provided insight into similarities between the methods or measures. Regarding the proposed strategies for finding desirable models, first, we suggested a strategy for finding configurations with high predictive accuracy, high feature selection stability, and a small number of selected features. Second, we proposed an approach for fitting models on data sets with similar features. The models are supposed to possess high predictive accuracy and select a subset of features such that all relevant information is contained but no irrelevant or redundant features are selected. For the second approach, a newly defined adjusted stability measure was employed. With both proposed strategies, the respective aims were reached and state-of-the-art strategies were outperformed. All stability measures considered in this thesis have been implemented and made publicly available in the open source R package *stabm* (Bommert and Lang, 2020).

Bibliography

Abeel, T., Helleputte, T., Van de Peer, Y., Dupont, P., and Saeys, Y. (2010). "Robust Biomarker Identification for Cancer Diagnosis with Ensemble Feature Selection Methods". In: *Bioinformatics* 26(3), pp. 392–398.

Akoglu, H. (2018). "User's Guide to Correlation Coefficients". In: *Turkish Journal of Emergency Medicine* 18(3), pp. 91–93.

Alelyani, S., Zhao, Z., and Liu, H. (2011). "A Dilemma in Assessing Stability of Feature Selection Algorithms". In: *2011 IEEE International Conference on High Performance Computing and Communications*, pp. 701–707.

Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. (2013). "A Public Domain Dataset for Human Activity Recognition Using Smartphones". In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pp. 437–442.

Aphinyanaphongs, Y., Fu, L. D., Li, Z., Peskin, E. R., Efstathiadis, E., Aliferis, C. F., and Statnikov, A. (2014). "A Comprehensive Empirical Comparison of Modern Supervised Classification and Feature Selection Methods for Text Categorization". In: *Journal of the Association for Information Science and Technology* 65(10), pp. 1964–1987.

Athar, A., Füllgrabe, A., George, N., Iqbal, H., Huerta, L., Ali, A., Snow, C., Fonseca, N. A., Petryszak, R., Papatheodorou, I., Sarkans, U., and Brazma, A. (2019). "ArrayExpress Update – From Bulk to Single-Cell Expression Data". In: *Nucleic Acids Research* 47(D1), pp. D711–D715.

Awada, W., Khoshgoftaar, T. M., Dittman, D., Wald, R., and Napolitano, A. (2012). "A Review of the Stability of Feature Selection Techniques for Bioinformatics Data". In: *2012 IEEE International Conference on Information Reuse and Integration*, pp. 356–363.

Bergstra, J. and Bengio, Y. (2012). "Random Search for Hyper-Parameter Optimization". In: *Journal of Machine Learning Research* 13, pp. 281–305.

Bertsimas, D., King, A., and Mazumder, R. (2016). "Best Subset Selection via a Modern Optimization Lens". In: *The Annals of Statistics* 44(2), pp. 813–852.

Bischl, B., Mersmann, O., Trautmann, H., and Weihs, C. (2012). "Resampling Methods for Meta-Model Validation with Recommendations for Evolutionary Computation". In: *Evolutionary Computation* 20(2), pp. 249–275.

Bischl, B., Lang, M., Kotthoff, L., Schiffner, J., Richter, J., Studerus, E., Casalicchio, G., and Jones, Z. M. (2016). "mlr: Machine Learning in R". In: *Journal of Machine Learning Research* 17, Article ID: 170.

Bolón-Canedo, V., Sánchez-Maroño, N., and Alonso-Betanzos, A. (2013). "A Review of Feature Selection Methods on Synthetic Data". In: *Knowledge and Information Systems* 34(3), pp. 483–519.

Bolón-Canedo, V., Sánchez-Marono, N., Alonso-Betanzos, A., Benítez, J. M., and Herrera, F. (2014). "A Review of Microarray Datasets and Applied Feature Selection Methods". In: *Information Sciences* 282, pp. 111–135.

Bommert, A. and Lang, M. (2020). "stabm: Stability Measures for Feature Selection". R package version 1.1.3. URL: `https://CRAN.R-project.org/package=stabm`.

Bommert, A. and Rahnenführer, J. (2020). "Adjusted Measures for Feature Selection Stability for Data Sets with Similar Features". In: *Machine Learning, Optimization, and Data Science*, pp. 203–214.

Bommert, A., Rahnenführer, J., and Lang, M. (2017). "A Multicriteria Approach to Find Predictive and Sparse Models with Stable Feature Selection for High-Dimensional Data". In: *Computational and Mathematical Methods in Medicine* 2017, Article ID: 7907163.

Bommert, A., Sun, X., Bischl, B., Rahnenführer, J., and Lang, M. (2020). "Benchmark for Filter Methods for Feature Selection in High-Dimensional Classification Data". In: *Computational Statistics & Data Analysis* 143, Article ID: 106839.

Borggaard, C. and Thodberg, H. H. (1992). "Optimal Minimal Neural Interpretation of Spectra". In: *Analytical Chemistry* 64(5), pp. 545–551.

Boulesteix, A.-L. and Slawski, M. (2009). "Stability and Aggregation of Ranked Gene Lists". In: *Briefings in Bioinformatics* 10(5), pp. 556–568.

Brezočnik, L., Fister, I., and Podgorelec, V. (2018). "Swarm Intelligence Algorithms for Feature Selection: A Review". In: *Applied Sciences* 8(9), Article ID: 1521.

Brown, G., Pocock, A., Zhao, M.-J., and Luján, M. (2012). "Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection". In: *Journal of Machine Learning Research* 13, pp. 27–66.

Bühlmann, P. and Yu, B. (2003). "Boosting With the L_2 Loss". In: *Journal of the American Statistical Association* 98(462), pp. 324–339.

Cai, J., Luo, J., Wang, S., and Yang, S. (2018). "Feature Selection in Machine Learning: A New Perspective". In: *Neurocomputing* 300, pp. 70–79.

Carletta, J. (1996). "Assessing Agreement on Classification Tasks: The Kappa Statistic". In: *Computational Linguistics* 22(2), pp. 249–254.

Casalicchio, G., Bossek, J., Lang, M., Kirchhoff, D., Kerschke, P., Hofner, B., Seibold, H., Vanschoren, J., and Bischl, B. (2017). "OpenML: An R Package to Connect to the Machine Learning Platform OpenML". In: *Computational Statistics*, pp. 1–15.

Cha, S.-H. (2007). "Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions". In: *Journal of Mathematical Models and Methods in Applied Sciences* 1(4), pp. 300–307.

Chandrashekar, G. and Sahin, F. (2014). "A Survey on Feature Selection Methods". In: *Computers & Electrical Engineering* 40(1), pp. 16–28.

Chiaretti, S., Li, X., Gentleman, R., Vitale, A., Vignetti, M., Mandelli, F., Ritz, J., and Foa, R. (2004). "Gene Expression Profile of Adult T-Cell Acute Lymphocytic Leukemia Identifies Distinct Subsets of Patients with Different Response to Therapy and Survival". In: *Blood* 103(7), pp. 2771–2778.

Christensen, B. C., Houseman, E. A., Marsit, C. J., Zheng, S., Wrensch, M. R., Wiemels, J. L., Nelson, H. H., Karagas, M. R., Padbury, J. F., Bueno, R., Sugarbaker, D. J., Yeh, R.-F., Wiencke, J. K., and Kelsey, K. T. (2009). "Aging and Environmental Exposures Alter Tissue-Specific DNA Methylation Dependent Upon CpG Island Context". In: *PLoS Genetics* 5(8), Article ID: e1000602.

Coello Coello, C. A., Lamont, G. B., and Van Veldhuizen, D. A. (2007). "Evolutionary Algorithms for Solving Multi-Objective Problems". 2nd edition. Springer, New York, USA.

Darshan, S. S. and Jaidhar, C. (2018). "Performance Evaluation of Filter-Based Feature Selection Techniques in Classifying Portable Executable Files". In: *Procedia Computer Science* 125, pp. 346–356.

Dash, M. and Liu, H. (1997). "Feature Selection for Classification". In: *Intelligent Data Analysis* 1, pp. 131–156.

Davis, C. A., Gerick, F., Hintermair, V., Friedel, C. C., Fundel, K., Küffner, R., and Zimmer, R. (2006). "Reliable Gene Signatures for Microarray Classification: Assessment of Stability and Performance". In: *Bioinformatics* 22(19), pp. 2356–2363.

Dessì, N., Pascariello, E., and Pes, B. (2013). "A Comparative Analysis of Biomarker Selection Techniques". In: *BioMed Research International* 2013, Article ID: 387673.

Dice, L. R. (1945). "Measures of the Amount of Ecologic Association Between Species". In: *Ecology* 26(3), pp. 297–302.

Dittman, D., Khoshgoftaar, T. M., Wald, R., and Wang, H. (2011). "Stability Analysis of Feature Ranking Techniques on Biological Datasets". In: *2011 IEEE International Conference on Bioinformatics and Biomedicine*, pp. 252–256.

Dunne, K., Cunningham, P., and Azuaje, F. (2002). "Solutions to Instability Problems with Sequential Wrapper-Based Approaches to Feature Selection". Technical Report. Trinity College, Dublin, Ireland.

Eyben, F., Wöllmer, M., and Schuller, B. (2010). "openSMILE: The Munich Versatile and Fast Open-Source Audio Feature Extractor". In: *Proceedings of the 18th ACM International Conference on Multimedia*, pp. 1459–1462.

Fayyad, U. and Irani, K. (1993). "Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning". Technical Report. California Institute of Technology.

Fleuret, F. (2004). "Fast Binary Feature Selection with Conditional Mutual Information". In: *Journal of Machine Learning Research* 5, pp. 1531–1555.

Forman, G. (2003). "An Extensive Empirical Study of Feature Selection Metrics for Text Classification". In: *Journal of Machine Learning Research* 3, pp. 1289–1305.

Ganti, V., Gehrke, J., Ramakrishnan, R., and Loh, W.-Y. (1999). "A Framework for Measuring Changes in Data Characteristics". In: *Proceedings of the 18th Symposium on Principles of Database Systems*, pp. 126–137.

Genz, A. and Bretz, F. (2009). "Computation of Multivariate Normal and t Probabilities". Lecture Notes in Statistics. Springer-Verlag, Heidelberg. ISBN: 978-3-642-01688-2.

Ghosh, M., Adhikary, S., Ghosh, K. K., Sardar, A., Begum, S., and Sarkar, R. (2019). "Genetic Algorithm Based Cancerous Gene Identification from Microarray Data Using Ensemble of Filter Methods". In: *Medical & Biological Engineering & Computing* 57(1), pp. 159–176.

Gorman, R. P. and Sejnowski, T. J. (1988). "Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets". In: *Neural Networks* 1(1), pp. 75–89.

Gravier, E., Pierron, G., Vincent-Salomon, A., Gruel, N., Raynal, V., Savignoni, A., De Rycke, Y., Pierga, J.-Y., Lucchesi, C., Reyal, F., Fourquet, A., Roman-Roman, S., Radvanyi, F., Sastre-Garau, X., Asselain, B., and Delattre, O. (2010). "A Prognostic DNA Signature for T1T2 Node-Negative Breast Cancer Patients". In: *Genes, Chromosomes and Cancer* 49(12), pp. 1125–1134.

Guyon, I. (2006). "Datasets for the Agnostic Learning vs. Prior Knowledge Competition". Technical Report. ClopiNet.

Guyon, I., Chaabane, I., Escalante, H. J., Escalera, S., Jajetic, D., Lloyd, J. R., Macià, N., Ray, B., Romaszko, L., Sebag, M., Statnikov, A., Treguer, S., and Viegas, E. (2016). "A Brief Review of the ChaLearn AutoML Challenge: Any-Time Any-Dataset Learning Without Human Intervention". In: *Proceedings of Machine Learning Research* 64, pp. 21–30.

Guyon, I. and Elisseeff, A. (2003). "An Introduction to Variable and Feature Selection". In: *Journal of Machine Learning Research* 3, pp. 1157–1182.

Guyon, I., Gunn, S., Ben-Hur, A., and Dror, G. (2004). "Result Analysis of the NIPS 2003 Feature Selection Challenge". In: *Proceedings of the 17th International Conference on Neural Information Processing Systems*, pp. 545–552.

Hall, M. A. (1999). "Correlation-Based Feature Selection for Machine Learning". PhD thesis. University of Waikato, Hamilton, New Zealand.

Hanley, J. A. and McNeil, B. J. (1982). "The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve". In: *Radiology* 143(1), pp. 29–36.

Hantke, S., Weninger, F., Kurle, R., Ringeval, F., Batliner, A., Mousa, A. E.-D., and Schuller, B. (2016). "I Hear You Eat and Speak: Automatic Recognition of Eating Condition and Food Type, Use-Cases, and Impact on ASR Performance". In: *PLoS ONE* 11(5), Article ID: e0154486.

Haury, A.-C., Gestraud, P., and Vert, J.-P. (2011). "The Influence of Feature Selection Methods on Accuracy, Stability and Interpretability of Molecular Signatures". In: *PloS ONE* 6(12), Article ID: e28210.

Hazimeh, H. and Mazumder, R. (2018). "L0Learn: Fast Algorithms for Best Subset Selection". R package version 1.0.9. URL: `https://CRAN.R-project.org/package=L0Learn`.

Hazimeh, H. and Mazumder, R. (2020). "Fast Best Subset Selection: Coordinate Descent and Local Combinatorial Optimization Algorithms". In: *Operations Research: Articles in Advance*, Article ID: 1526-5463.

He, Z. and Yu, W. (2010). "Stable Feature Selection for Biomarker Discovery". In: *Computational Biology and Chemistry* 34(4), pp. 215–225.

Helleputte, T. (2017). "LiblineaR: Linear Predictive Models Based on the LIBLINEAR C/C++ Library". R package version 2.10-8.

Hira, Z. M. and Gillies, D. F. (2015). "A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data". In: *Advances in Bioinformatics* 2015, Article ID: 198363.

Hofner, B. and Hothorn, T. (2017). "stabs: Stability Selection with Error Control". R package version 0.6-3. URL: `https://CRAN.R-project.org/package=stabs`.

Hofner, B., Mayr, A., Robinzonov, N., and Schmid, M. (2014). "Model-Based Boosting in R: A Hands-On Tutorial Using the R Package mboost". In: *Computational Statistics* 29(1–2), pp. 3–35.

Hopcroft, J. E. and Karp, R. M. (1973). "An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs". In: *SIAM Journal on Computing* 2(4), pp. 225–231.

Hoque, N., Singh, M., and Bhattacharyya, D. K. (2018). "EFS-MI: An Ensemble Feature Selection Method for Classification". In: *Complex & Intelligent Systems* 4(2), pp. 105–118.

Horn, D., Wagner, T., Biermann, D., Weihs, C., and Bischl, B. (2015). "Model-Based Multi-Objective Optimization: Taxonomy, Multi-Point Proposal, Toolbox and Benchmark". In: *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 64–78.

Hothorn, T., Bühlmann, P., Kneib, T., Schmid, M., and Hofner, B. (2018). "mboost: Model-Based Boosting". R package version 2.9-1. URL: `https://CRAN.R-project.org/package=mboost`.

Huang, J., Jiao, Y., Liu, Y., and Lu, X. (2018a). "A Constructive Approach to L_0 Penalized Regression". In: *Journal of Machine Learning Research* 19, Article ID: 10.

Huang, X., Zhang, L., Wang, B., Li, F., and Zhang, Z. (2018b). "Feature Clustering Based Support Vector Machine Recursive Feature Elimination for Gene Selection". In: *Applied Intelligence* 48(3), pp. 594–607.

Inza, I., Larrañaga, P., Blanco, R., and Cerrolaza, A. J. (2004). "Filter Versus Wrapper Gene Selection Approaches in DNA Microarray Domains". In: *Artificial Intelligence in Medicine* 31(2), pp. 91–103.

Izenman, A. J. (2013). "Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning". 2nd edition. Springer, New York, USA.

Jaccard, P. (1901). "Étude Comparative de la Distribution Florale Dans une Portion des Alpes et du Jura". In: *Bulletin de la Société Vaudoise des Sciences Naturelles* 37, pp. 547–579.

John, G. H., Kohavi, R., and Pfleger, K. (1994). "Irrelevant Features and the Subset Selection Problem". In: *Machine Learning Proceedings 1994*, pp. 121–129.

Jović, A., Brkić, K., and Bogunović, N. (2015). "A Review of Feature Selection Methods with Applications". In: *38th International Convention on Information and Communication Technology, Electronics and Microelectronics*, pp. 1200–1205.

Kalousis, A., Prados, J., and Hilario, M. (2007). "Stability of Feature Selection Algorithms: A Study on High-Dimensional Spaces". In: *Knowledge and Information Systems* 12(1), pp. 95–116.

Karatzoglou, A., Smola, A., Hornik, K., and Zeileis, A. (2004). "kernlab – An S4 Package for Kernel Methods in R". In: *Journal of Statistical Software* 11, Article ID: 9.

Ke, W., Wu, C., Wu, Y., and Xiong, N. N. (2018). "A New Filter Feature Selection Based on Criteria Fusion for Gene Microarray Data". In: *IEEE Access* 6, pp. 61065–61076.

Kerschke, P. and Trautmann, H. (2019). "Automated Algorithm Selection on Continuous Black-Box Problems by Combining Exploratory Landscape Analysis and Machine Learning". In: *Evolutionary Computation* 27(1), pp. 99–127.

Kittler, J. (1978). "Feature Set Search Algorithms". In: *Pattern Recognition and Signal Processing*. Sijthoff and Noordhoff, Alphen aan den Rijn, Netherlands, pp. 41–60.

Kohavi, R. and John, G. H. (1997). "Wrappers for Feature Subset Selection". In: *Artificial Intelligence* 97(1–2), pp. 273–324.

Koller, D. and Sahami, M. (1996). "Toward Optimal Feature Selection". Technical Report. Stanford InfoLab.

Koru, A. G. and Liu, H. (2005). "An Investigation of the Effect of Module Size on Defect Prediction Using Static Measures". In: *ACM SIGSOFT Software Engineering Notes* 30(4), pp. 1–5.

Kruskal, W. H. and Wallis, W. A. (1952). "Use of Ranks in One-Criterion Variance Analysis". In: *Journal of the American Statistical Association* 47(260), pp. 583–621.

Kursa, M. B. (2018). "praznik: Collection of Information-Based Feature Selection Filters". R package version 5.0.0. URL: `https://CRAN.R-project.org/package=praznik`.

Lang, M., Bischl, B., and Surmann, D. (2017). "batchtools: Tools for R to Work on Batch Systems". In: *Journal of Open Source Software* 2(10).

Larose, D. T. and Larose, C. D. (2014). "Discovering Knowledge in Data". 2nd edition. John Wiley & Sons, Inc., Hoboken, NJ, USA.

Laumanns, M. and Zenklusen, R. (2011). "Stochastic Convergence of Random Search Methods to Fixed Size Pareto Front Approximations". In: *European Journal of Operational Research* 213(2), pp. 414–421.

Lausser, L., Müssel, C., Maucher, M., and Kestler, H. A. (2013). "Measuring and Visualizing the Stability of Biomarker Selection Techniques". In: *Computational Statistics* 28(1), pp. 51–65.

Lazar, C., Taminau, J., Meganck, S., Steenhoff, D., Coletta, A., Molter, C., Schaetzen, V. de, Duque, R., Bersini, H., and Nowe, A. (2012). "A Survey on Filter Techniques for Feature Selection in Gene Eexpression Microarray Analysis". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 9(4), pp. 1106–1119.

Lee, H. W., Lawton, C., Na, Y. J., and Yoon, S. (2013). "Robustness of Chemometrics-Based Feature Selection Methods in Early Cancer Detection and Biomarker Discovery". In: *Statistical Applications in Genetics and Molecular Biology* 12(2), pp. 207–223.

Lee, S., Rahnenführer, J., Lang, M., De Preter, K., Mestdagh, P., Koster, J., Versteeg, R., Stallings, R. L., Varesio, L., Asgharzadeh, S., Schulte, J. H., Fielitz, K., Schwermer, M.,

Morik, K., and Schramm, A. (2014). "Robust Selection of Cancer Survival Signatures from High-Throughput Genomic Data Using Two-Fold Subsampling". In: *PLoS ONE* 9(10), Article ID: e108818.

Leite, R., Brazdil, P., and Vanschoren, J. (2012). "Selecting Classification Algorithms with Active Testing". In: *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pp. 117–131.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. (2018). "Feature Selection: A Data Perspective". In: *ACM Computing Surveys* 50(6), Article ID: 94.

Li, X. (2018). "ALL: A Data Package". R package version 1.24.0. URL: `https://bioconductor.org/packages/ALL`.

Liu, H. and Yu, L. (2005). "Toward Integrating Feature Selection Algorithms for Classification and Clustering". In: *IEEE Transactions on Knowledge and Data Engineering* 17(4), pp. 491–502.

Liu, H., Li, J., and Wong, L. (2002). "A Comparative Study on Feature Selection and Classification Methods Using Gene Expression Profiles and Proteomic Patterns". In: *Genome Informatics* 13, pp. 51–60.

Liu, Y. (2004). "A Comparative Study on Feature Selection Methods for Drug Discovery". In: *Journal of Chemical Information and Computer Sciences* 44(5), pp. 1823–1828.

Lustgarten, J. L., Gopalakrishnan, V., and Visweswaran, S. (2009). "Measuring Stability of Feature Selection in Biomedical Datasets". In: *AMIA Annual Symposium Proceedings 2009*, pp. 406–410.

Meier, L., Van de Geer, S., and Bühlmann, P. (2008). "The Group Lasso for Logistic Regression". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70(1), pp. 53–71.

Meinshausen, N. and Bühlmann, P. (2010). "Stability Selection". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72(4), pp. 417–473.

Meyer, P. E., Schretter, C., and Bontempi, G. (2008). "Information-Theoretic Feature Selection in Microarray Data Using Variable Complementarity". In: *IEEE Journal of Selected Topics in Signal Processing* 2(3), pp. 261–274.

Miettinen, K. (2008). "Introduction to Multiobjective Optimization: Noninteractive Approaches". In: *Multiobjective Optimization: Interactive and Evolutionary Approaches*. Springer, Berlin, pp. 1–26.

Miettinen, K. M. (2004). "Nonlinear Multiobjective Optimization". 4th edition. Springer, New York, USA.

Mohtashami, M. and Eftekhari, M. (2019). "A Hybrid Filter-Based Feature Selection Method via Hesitant Fuzzy and Rough Sets Concepts". In: *Iranian Journal of Fuzzy Systems* 16(2), pp. 165–182.

Nogueira, S. (2018). "Quantifying the Stability of Feature Selection". PhD thesis. University of Manchester, United Kingdom.

Nogueira, S. and Brown, G. (2016). "Measuring the Stability of Feature Selection". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 442–457.

Novovičová, J., Somol, P., and Pudil, P. (2009). "A New Measure of Feature Selection Algorithms' Stability". In: *2009 IEEE International Conference on Data Mining Workshops*, pp. 382–387.

Ntoutsi, I., Kalousis, A., and Theodoridis, Y. (2008). "A General Framework for Estimating Similarity of Datasets and Decision Trees: Exploring Semantic Similarity of Decision

Trees". In: *Proceedings of the 2008 SIAM International Conference on Data Mining*, pp. 810–821.

Ochiai, A. (1957). "Zoogeographic Studies on the Soleoid Fishes Found in Japan and Its Neighbouring Regions". In: *Bulletin of the Japanese Society for the Science of Fish* 22(9), pp. 526–530.

Peng, H., Long, F., and Ding, C. (2005). "Feature Selection Based on Mutual Information Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(8), pp. 1226–1238.

Probst, P., Wright, M. N., and Boulesteix, A.-L. (2019). "Hyperparameters and Tuning Strategies for Random Forest". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 9(3), Article ID: e1301.

R Core Team (2018). "R: A Language and Environment for Statistical Computing". R Foundation for Statistical Computing, Vienna, Austria. URL: `https://www.R-project.org/`.

Rahman, M. S. (2017). "Basic Graph Theory". Springer, New York, USA.

Ramey, J. A. (2016). "datamicroarray: Collection of Data Sets for Classification". URL: `https://github.com/ramhiser/datamicroarray`.

Rasch, D., Kubinger, K. D., and Yanagida, T. (2011). "Statistics in Psychology Using R and SPSS". John Wiley & Sons, Inc., Hoboken, NJ, USA.

Ritchie, M. E., Phipson, B., Wu, D., Hu, Y., Law, C. W., Shi, W., and Smyth, G. K. (2015). "limma Powers Differential Expression Analyses for RNA-Sequencing and Microarray Studies". In: *Nucleic Acids Research* 43(7), Article ID: e47.

Roederer, M., Moore, W., Treister, A., Hardy, R. R., and Herzenberg, L. A. (2001). "Probability Binning Comparison: A Metric for Quantitating Multivariate Distribution Differences". In: *Cytometry* 45(1), pp. 47–55.

Romanski, P. and Kotthoff, L. (2018). "FSelector: Selecting Attributes". R package version 0.31. URL: `https://CRAN.R-project.org/package=FSelector`.

Saeys, Y., Abeel, T., and Van de Peer, Y. (2008). "Robust Feature Selection Using Ensemble Feature Selection Techniques". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 313–325.

Saeys, Y., Inza, I., and Larrañaga, P. (2007). "A Review of Feature Selection Techniques in Bioinformatics". In: *Bioinformatics* 23(19), pp. 2507–2517.

Sammut, C. and Webb, G. I. (2011). "Encyclopedia of Machine Learning". Springer, New York, USA.

Sánchez-Maroño, N., Alonso-Betanzos, A., and Tombilla-Sanromán, M. (2007). "Filter Methods for Feature Selection – A Comparative Study". In: *International Conference on Intelligent Data Engineering and Automated Learning*, pp. 178–187.

Schirra, L.-R., Lausser, L., and Kestler, H. A. (2016). "Selection Stability as a Means of Biomarker Discovery in Classification". In: *Analysis of Large and Complex Data*. Springer, New York, USA, pp. 79–89.

Schliep, K. and Hechenbichler, K. (2016). "kknn: Weighted k-Nearest Neighbors". R package version 1.3.1. URL: `https://CRAN.R-project.org/package=kknn`.

Schuller, B., Steidl, S., Batliner, A., Hantke, S., Hönig, F., Orozco-Arroyave, J. R., Nöth, E., Zhang, Y., and Weninger, F. (2015). "The INTERSPEECH 2015 Computational Paralinguistics Challenge: Nativeness, Parkinson's & Eating Condition". In: *16th Annual Conference of the International Speech Communication Association*, pp. 478–482.

Sechidis, K., Papangelou, K., Nogueira, S., Weatherall, J., and Brown, G. (2020). "On the Stability of Feature Selection in the Presence of Feature Correlations". In: *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, pp. 327–342.

Shah, R. D. and Samworth, R. J. (2013). "Variable Selection with Error Control: Another Look at Stability Selection". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 75(1), pp. 55–80.

Simon, N., Friedman, J., Hastie, T., and Tibshirani, R. (2011). "Regularization Paths for Cox's Proportional Hazards Model via Coordinate Descent". In: *Journal of Statistical Software* 39, Article ID: 5.

Smyth, G. K. (2004). "Linear Models and Empirical Bayes Methods for Assessing Differential Expression in Microarray Experiments". In: *Statistical Applications in Genetics and Molecular Biology* 3(1), Article ID: 3.

Somol, P. and Novovičová, J. (2008). "Evaluating the Stability of Feature Selectors that Optimize Feature Subset Cardinality". In: *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition and Structural and Syntactic Pattern Recognition*, pp. 956–966.

Somol, P. and Novovičová, J. (2010). "Evaluating Stability and Comparing Output of Feature Selectors That Optimize Feature Subset Cardinality". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32(11), pp. 1921–1939.

Statnikov, A., Lytkin, N. I., Lemeire, J., and Aliferis, C. F. (2013). "Algorithms for Discovery of Multiple Markov Boundaries". In: *Journal of Machine Learning Research* 14, pp. 499–566.

Tang, J., Alelyani, S., and Liu, H. (2014). "Feature Selection for Classification: A Review". In: *Data Classification: Algorithms and Applications*. CRC Press, Boca Raton, FL, USA, pp. 37–64.

Thodberg, H. H. (1993). "Ace of Bayes: Application of Neural Networks with Pruning". Technical Report. Danish Meat Research Institute.

Thomas, J., Hepp, T., Mayr, A., and Bischl, B. (2017). "Probing for Sparse and Fast Variable Selection with Model-Based Boosting". In: *Computational and Mathematical Methods in Medicine* 2017, Article ID: 1421409.

Tibshirani, R., Chu, G., Narasimhan, B., and Li, J. (2011). "samr: SAM: Significance Analysis of Microarrays". R package version 2.0. URL: `https://CRAN.R-project.org/package=samr`.

Toloşi, L. and Lengauer, T. (2011). "Classification with Correlated Features: Unreliability of Feature Ranking and Solutions". In: *Bioinformatics* 27(14), pp. 1986–1994.

Tsanas, A., Little, M. A., Fox, C., and Ramig, L. O. (2014). "Objective Automatic Assessment of Rehabilitative Speech Treatment in Parkinson's Disease". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 22(1), pp. 181–190.

Tusher, V. G., Tibshirani, R., and Chu, G. (2001). "Significance Analysis of Microarrays Applied to the Ionizing Radiation Response". In: *Proceedings of the National Academy of Sciences of the United States of America* 98(9), pp. 5116–5121.

Vanschoren, J., Van Rijn, J. N., Bischl, B., and Torgo, L. (2013). "OpenML: Networked Science in Machine Learning". In: *ACM SIGKDD Explorations Newsletter* 15(2), pp. 49–60.

Venkatesh, B. and Anuradha, J. (2019). "A Review of Feature Selection and Its Methods". In: *Cybernetics and Information Technologies* 19(1), pp. 3–26.

Wagner, J., Seiderer, A., Lingenfelser, F., and André, E. (2015). "Combining Hierarchical Classification with Frequency Weighting for the Recognition of Eating Conditions". In: *16th Annual Conference of the International Speech Communication Association*, pp. 889–893.

Wah, Y. B., Ibrahim, N., Hamid, H. A., Abdul-Rahman, S., and Fong, S. (2018). "Feature Selection Methods: Case of Filter and Wrapper Approaches for Maximising Classification Accuracy". In: *Pertanika Journal of Science & Technology* 26(1), pp. 329–340.

Wald, R., Khoshgoftaar, T. M., and Napolitano, A. (2013). "Stability of Filter- and Wrapper-Based Feature Subset Selection". In: *2013 IEEE 25th International Conference on Tools with Artificial Intelligence*, pp. 374–380.

Wang, H., Khoshgoftaar, T. M., Wald, R., and Napolitano, A. (2012). "A Novel Dataset-Similarity-Aware Approach for Evaluating Stability of Software Metric Selection Techniques". In: *2012 IEEE International Conference on Information Reuse and Integration*, pp. 1–8.

Wickham, H. (2016). "ggplot2: Elegant Graphics for Data Analysis". Springer, New York, USA.

Wright, M. N. and Ziegler, A. (2017). "ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R". In: *Journal of Statistical Software* 77, Article ID: 1.

Xue, B., Zhang, M., and Browne, W. N. (2015). "A Comprehensive Comparison on Evolutionary Feature Selection Approaches to Classification". In: *International Journal of Computational Intelligence and Applications* 14(2), Article ID: 1550008.

Xue, B., Zhang, M., Browne, W. N., and Yao, X. (2016). "A Survey on Evolutionary Computation Approaches to Feature Selection". In: *IEEE Transactions on Evolutionary Computation* 20(4), pp. 606–626.

Yang, J. and Honavar, V. (1998). "Feature Subset Selection Using a Genetic Algorithm". In: *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Springer, New York, USA, pp. 117–136.

Yu, L., Han, Y., and Berens, M. E. (2012). "Stable Gene Selection from Microarray Data via Sample Weighting". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 9(1), pp. 262–272.

Yu, L. and Liu, H. (2004). "Efficient Feature Selection via Analysis of Relevance and Redundancy". In: *Journal of Machine Learning Research* 5, pp. 1205–1224.

Zawadzki, Z. and Kosinski, M. (2018). "FSelectorRcpp: 'Rcpp' Implementation of 'FSelector' Entropy-Based Feature Selection Algorithms with a Sparse Matrix Support". R package version 0.3.0. URL: `https://CRAN.R-project.org/package=FSelectorRcpp`.

Zhang, M., Zhang, L., Zou, J., Yao, C., Xiao, H., Liu, Q., Wang, J., Wang, D., Wang, C., and Guo, Z. (2009). "Evaluating Reproducibility of Differential Expression Discoveries in Microarray Studies by Considering Correlated Molecular Changes". In: *Bioinformatics* 25(13), pp. 1662–1668.

Zhu, Z., Ong, Y.-S., and Dash, M. (2007). "Wrapper-Filter Feature Selection Algorithm Using a Memetic Framework". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 37(1), pp. 70–76.

Zuber, V. and Strimmer, K. (2009). "Gene Ranking and Biomarker Discovery Under Correlation". In: *Bioinformatics* 25(20), pp. 2700–2707.

Zucknick, M., Richardson, S., and Stronach, E. A. (2008). "Comparing the Characteristics of Gene Expression Profiles Derived by Univariate and Multivariate Classification Methods". In: *Statistical Applications in Genetics and Molecular Biology* 7(1), Article ID: 7.

Appendix A

Further Figures

A.1 Data Sets

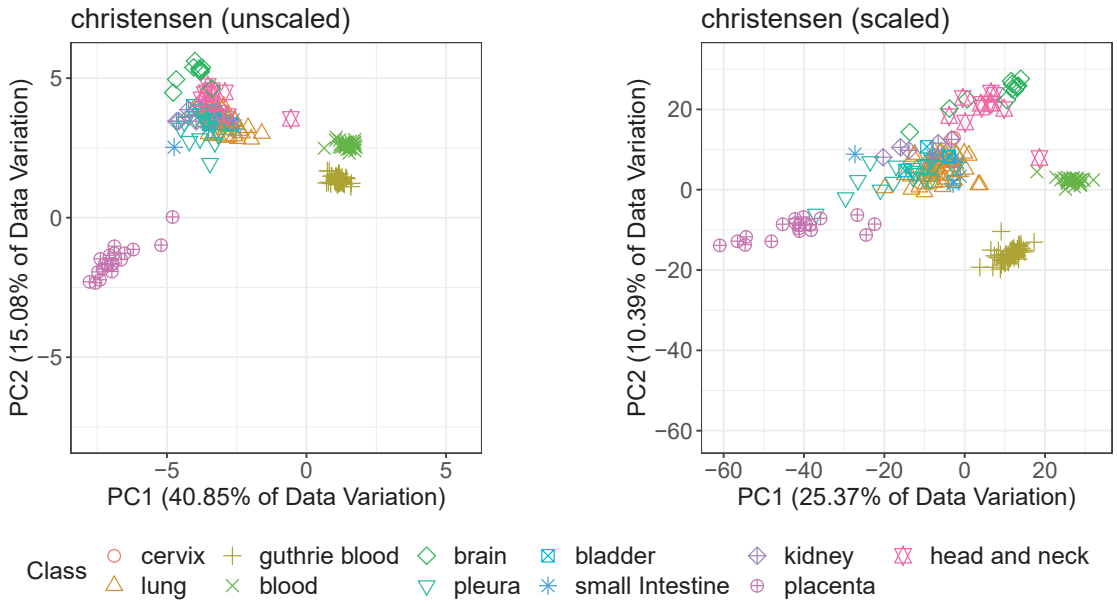


Figure A.1: PCA plots data set *christensen* with the original classes. For the right plot, the data is scaled for the computation of the principal components, for the left plot it is not.

A.2 Benchmark of Filter Methods

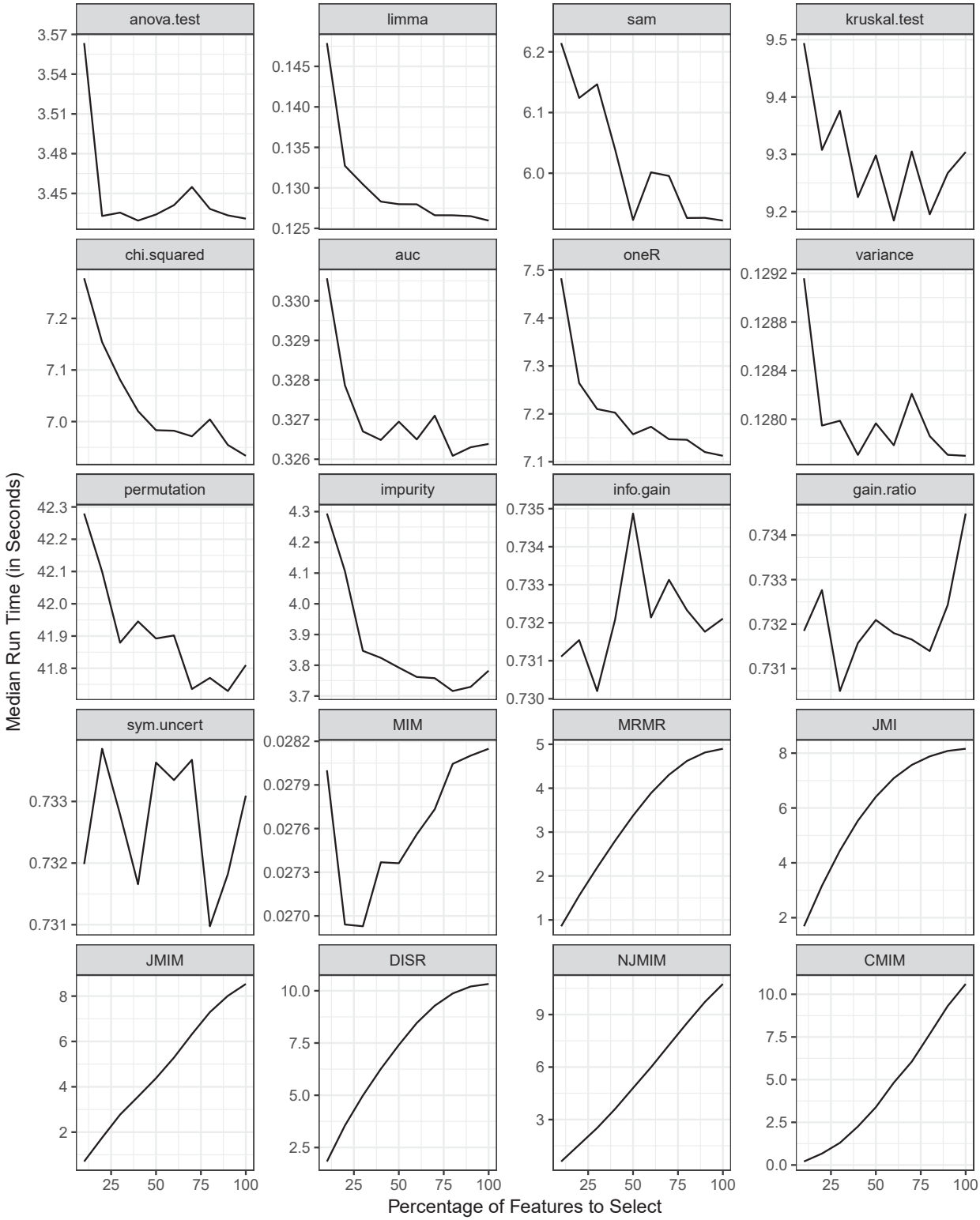


Figure A.2: Scaling behavior of the filter methods: median time for filtering with respect to the percentage of features to select.

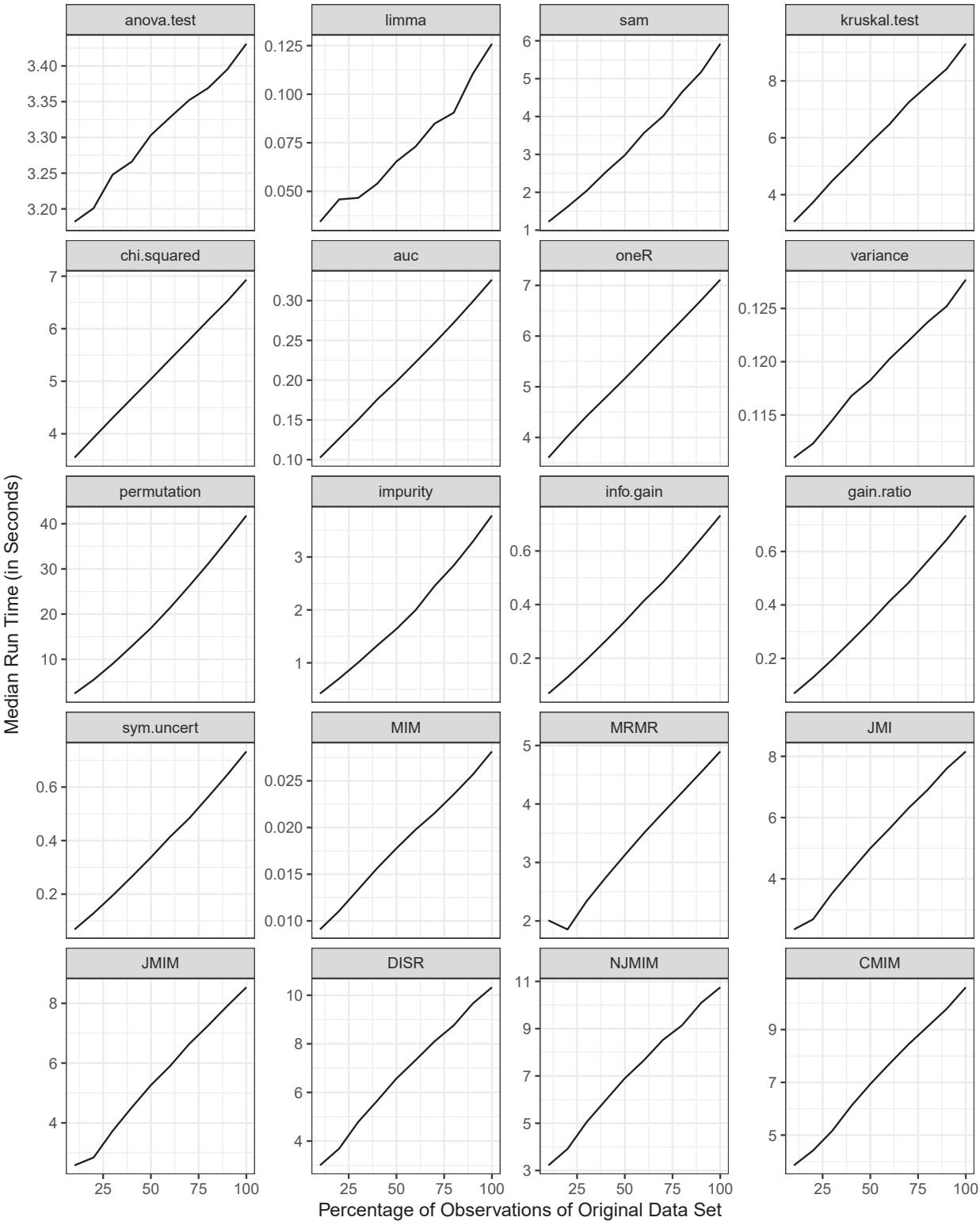


Figure A.3: Scaling behavior of the filter methods: median time for calculating the scores for all features for data sets with different numbers of observations (identical numbers of features).

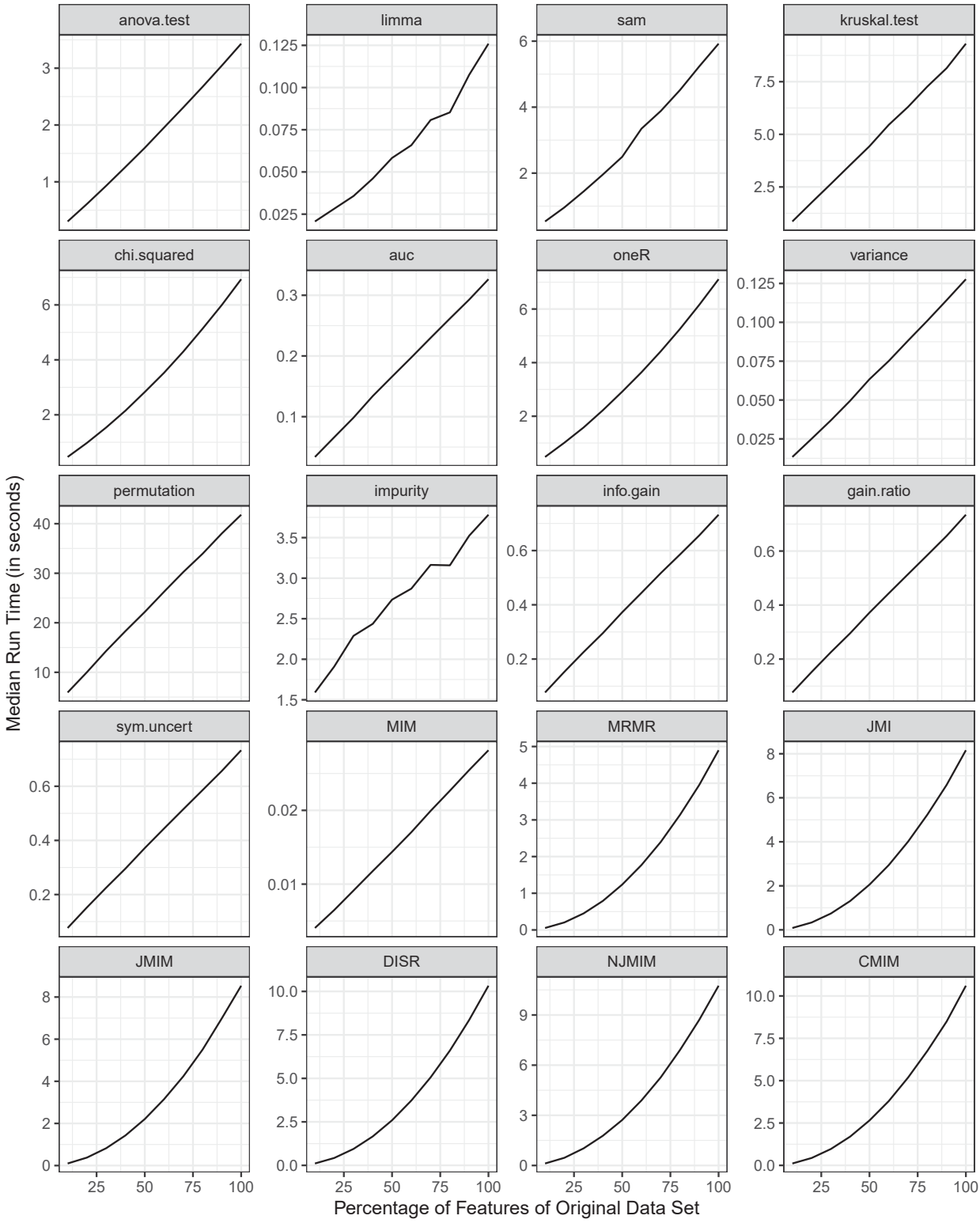


Figure A.4: Scaling behavior of the filter methods: median time for calculating the scores for all features for data sets with different numbers of features (identical numbers of observations).

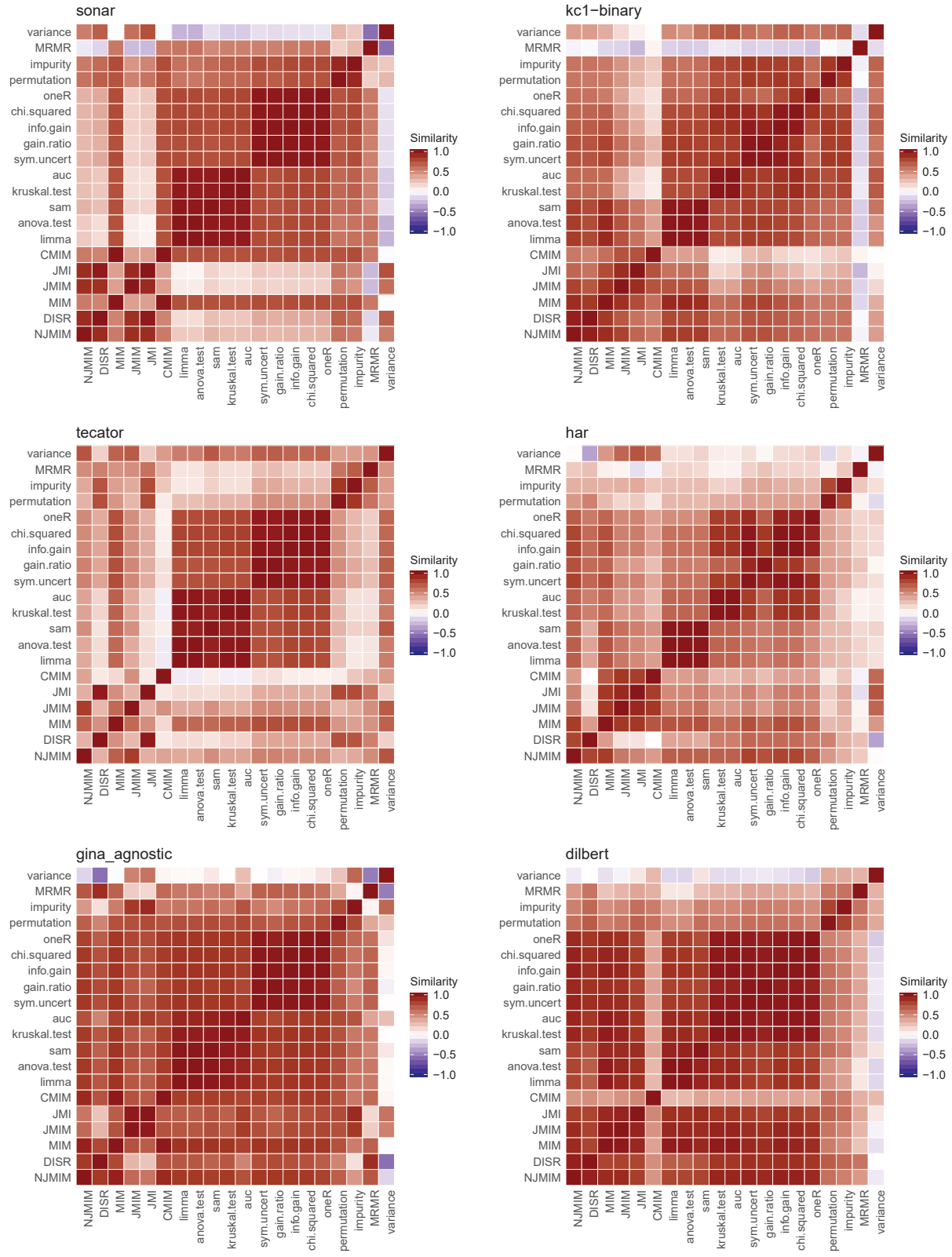


Figure A.5: Rank correlations between the selection order of all features for all pairs of filter methods per data set. The filter methods are ordered by average linkage hierarchical clustering using the mean rank correlation as similarity measure.

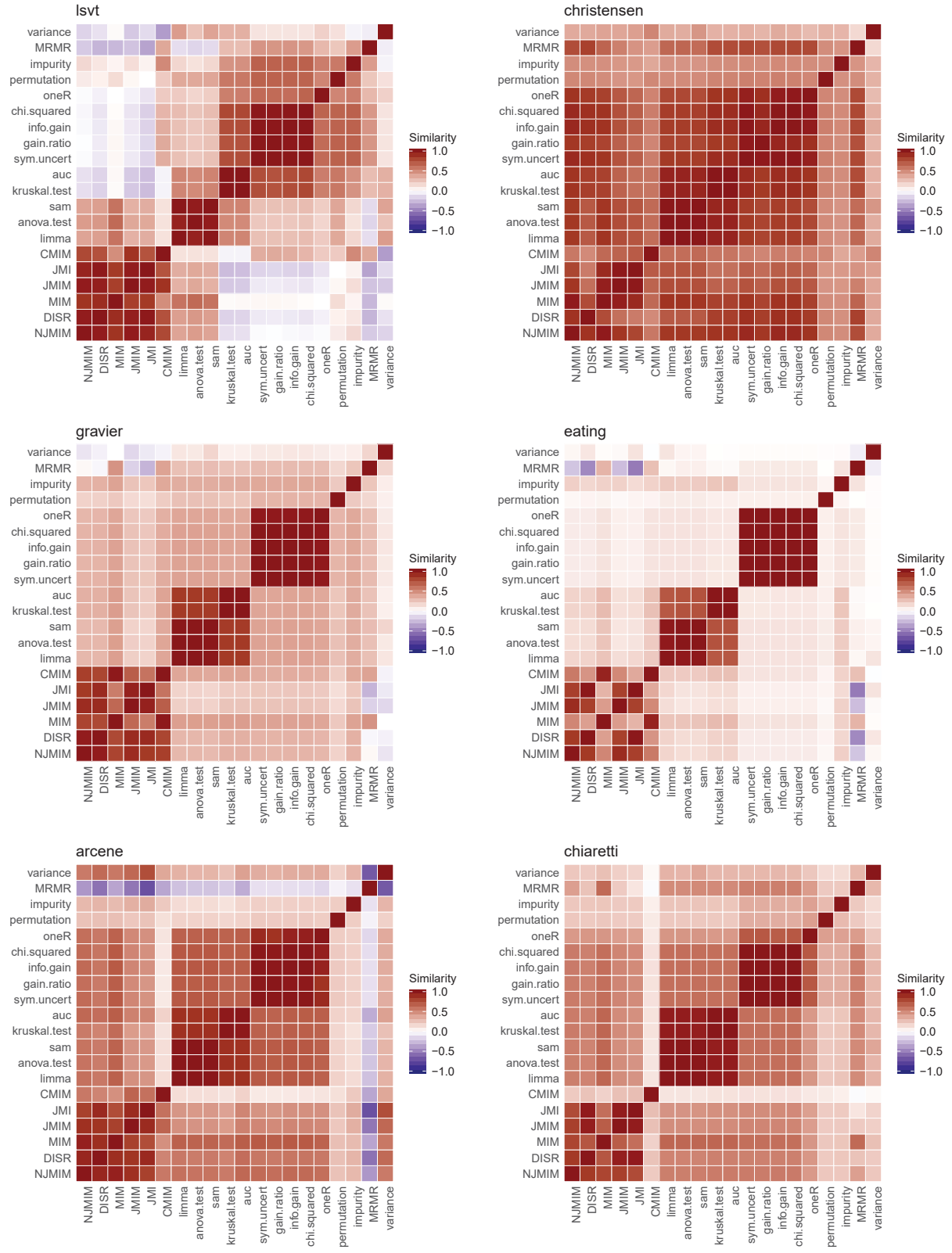


Figure A.6: Rank correlations between the selection order of all features for all pairs of filter methods per data set. The filter methods are ordered by average linkage hierarchical clustering using the mean rank correlation as similarity measure.

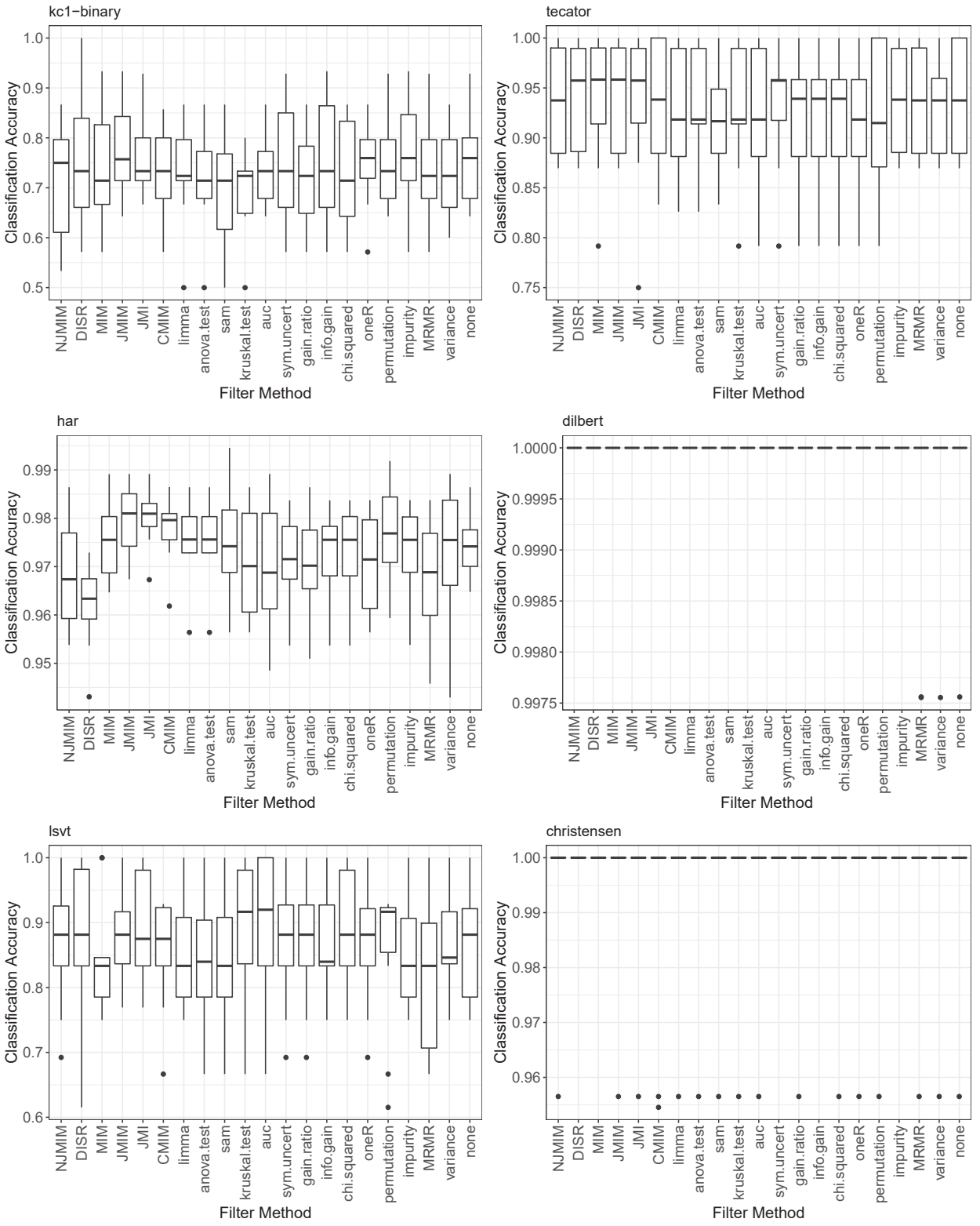


Figure A.7: Boxplots of the classification accuracies of the best configurations in the 10 outer cross-validation iterations per filter method and data set.

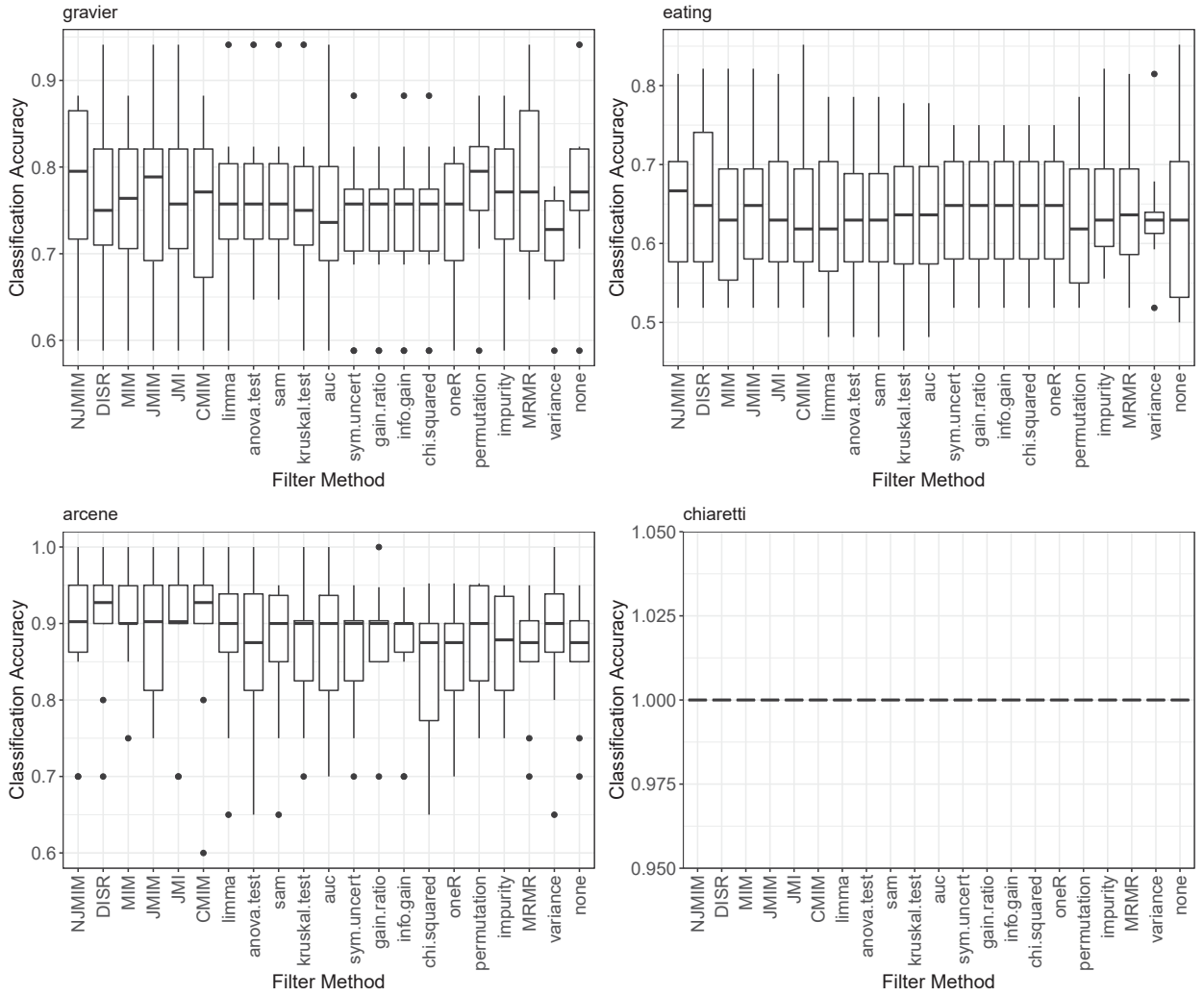


Figure A.8: Boxplots of the classification accuracies of the best configurations in the 10 outer cross-validation iterations per filter method and data set.

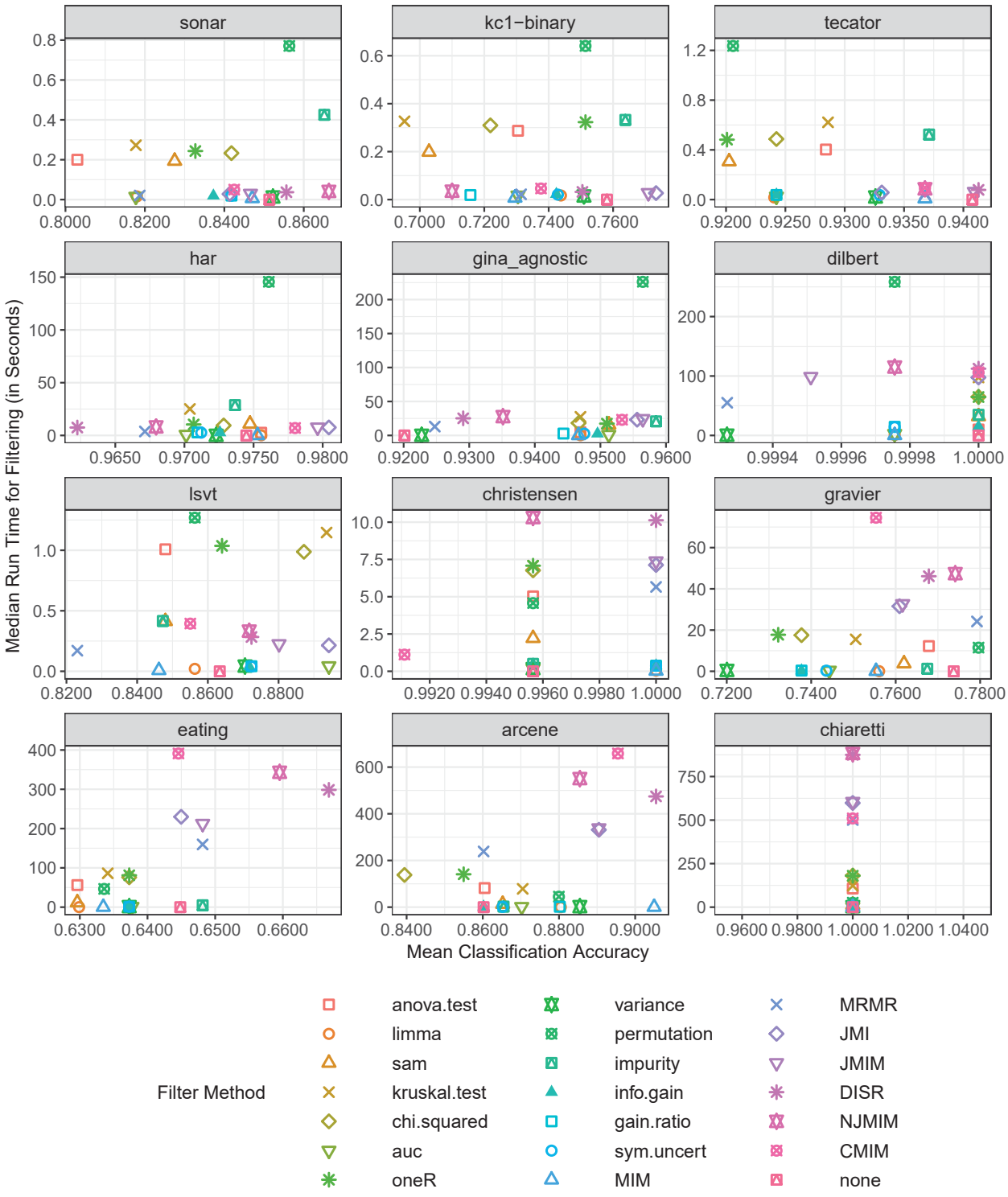


Figure A.9: Mean classification accuracy and median run time for filtering of all considered filter methods with optimal configurations per data set.

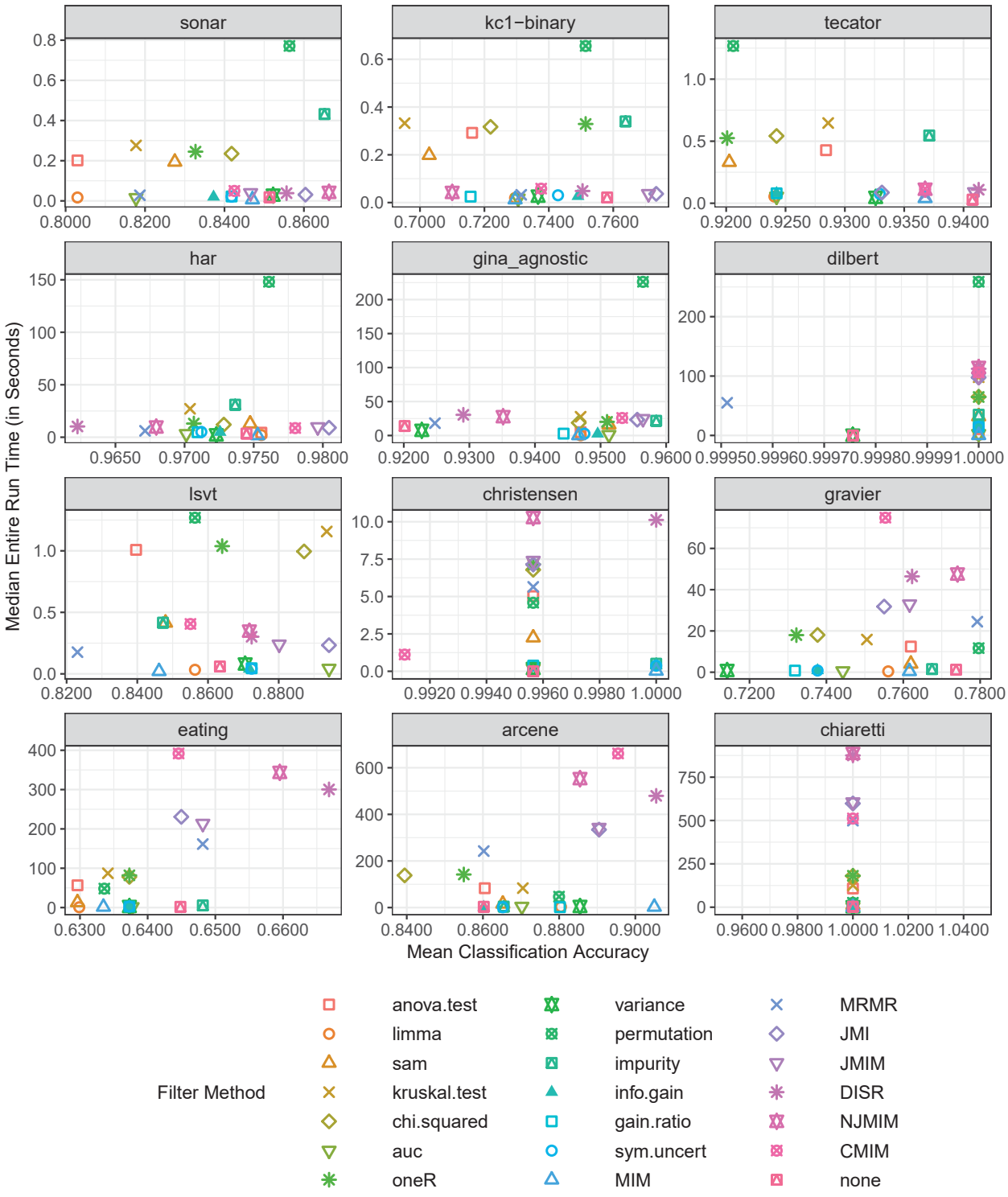


Figure A.10: Mean classification accuracy and median run time for fitting the combined model of all considered filter methods with optimal configurations per data set.

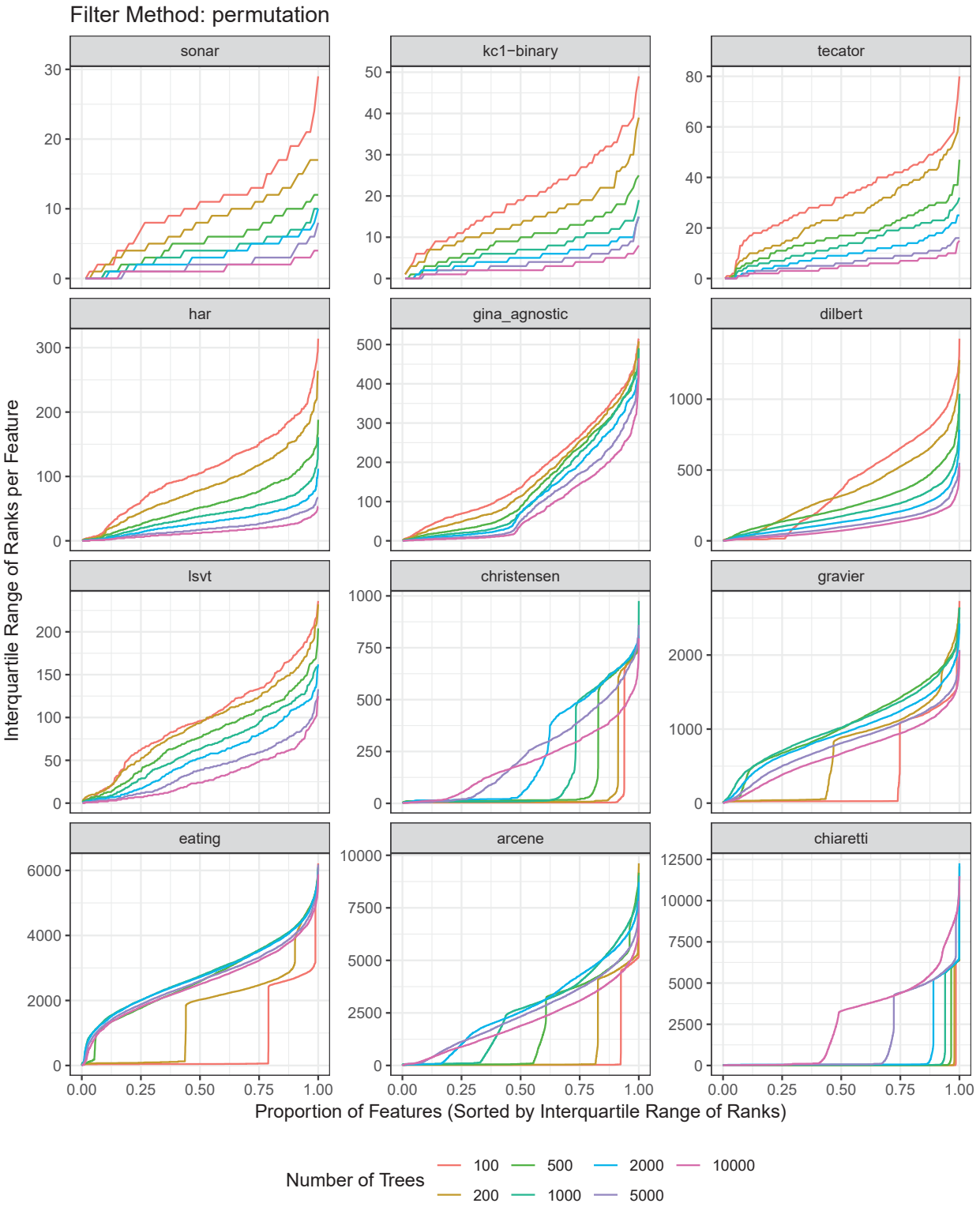


Figure A.11: Interquartile ranges of the ranks of the scores calculated with filter *permutation* for each feature. The features are sorted increasingly by the interquartile range of their ranks.

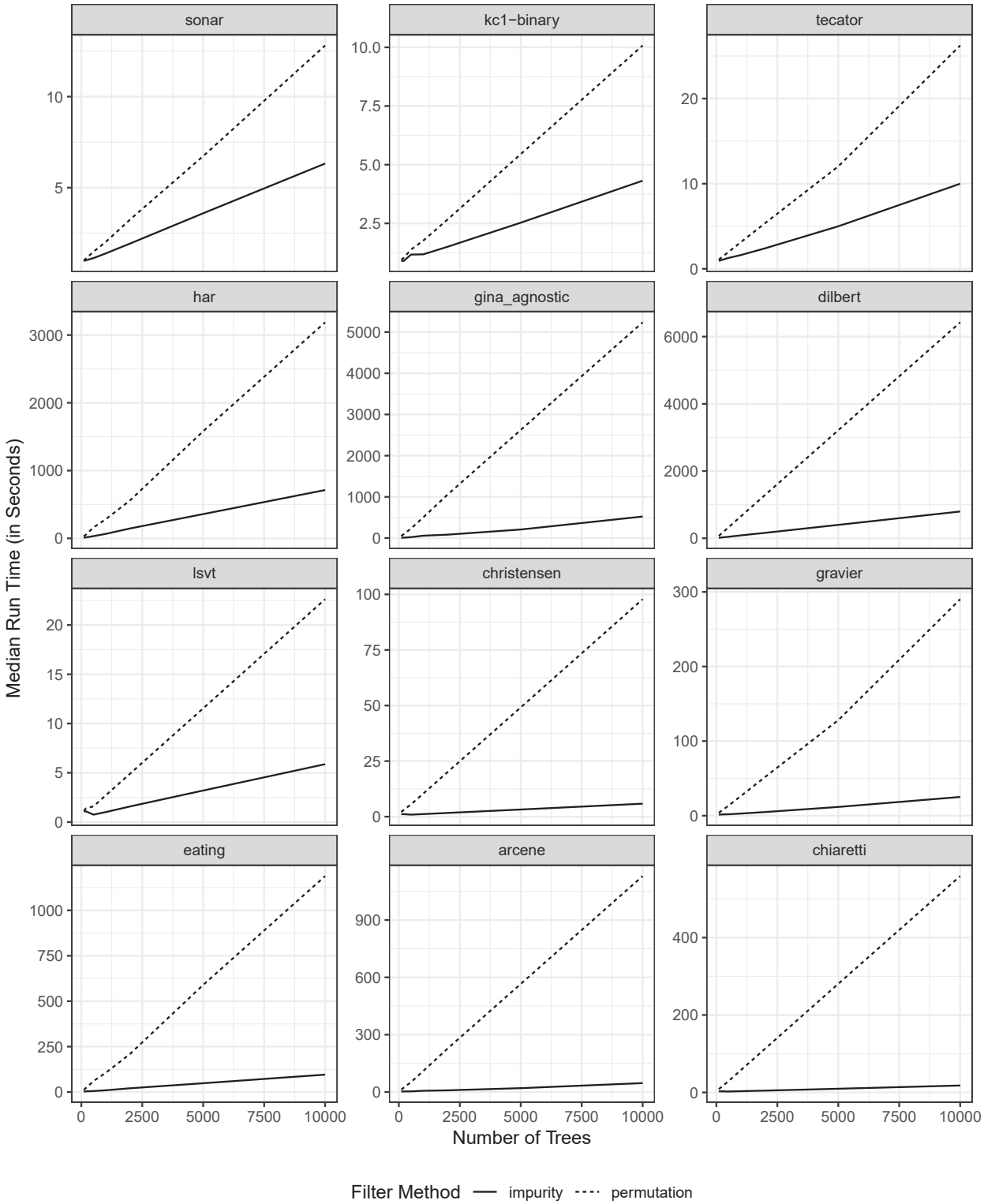


Figure A.12: Median run times of the filter methods *permutation* and *impurity* depending on the number of trees in the random forests.

A.3 Comparison of Stability Measures

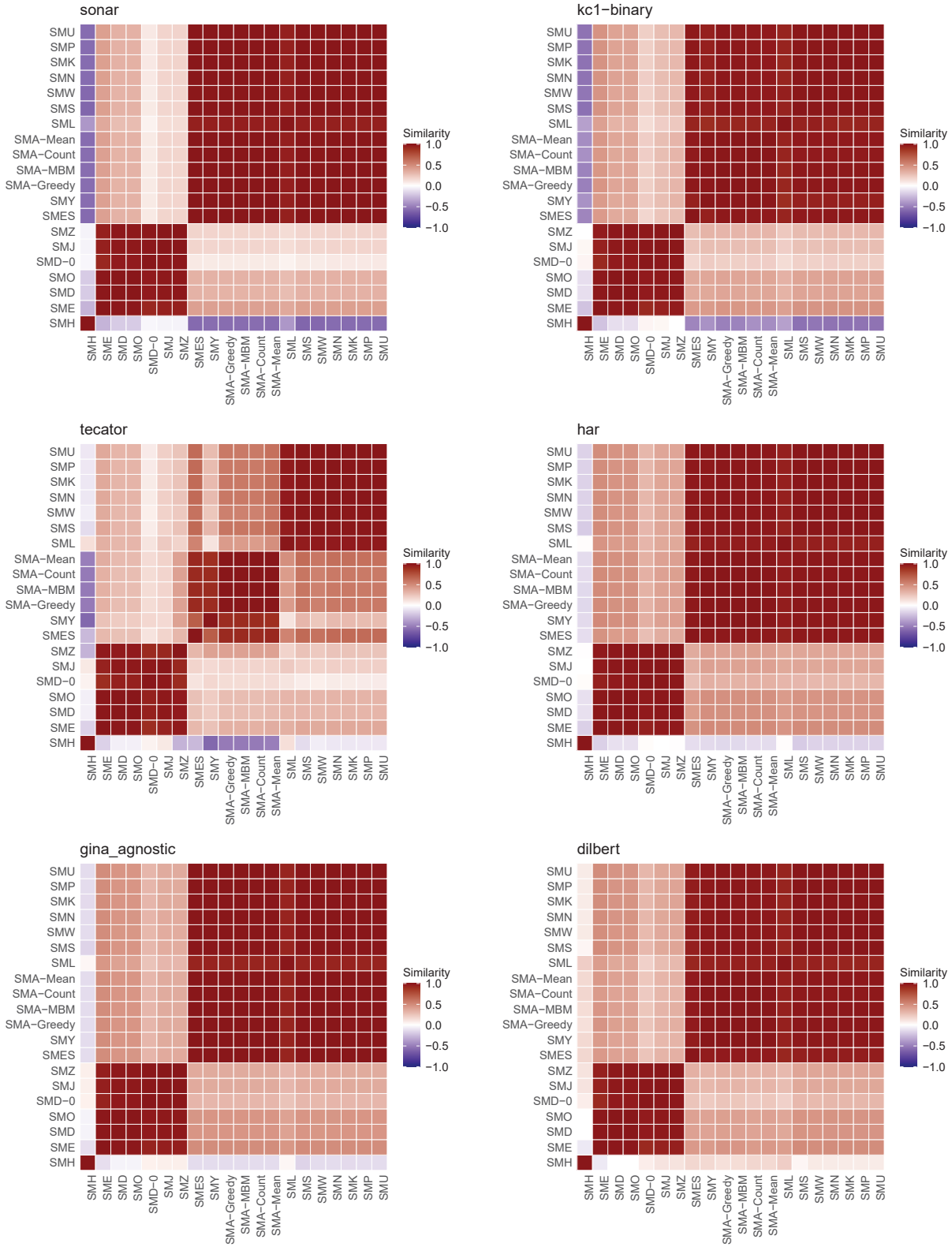


Figure A.13: Pearson correlations between all pairs of stability measures per data set. The order of the stability measures is the same as in Figure 6.5.

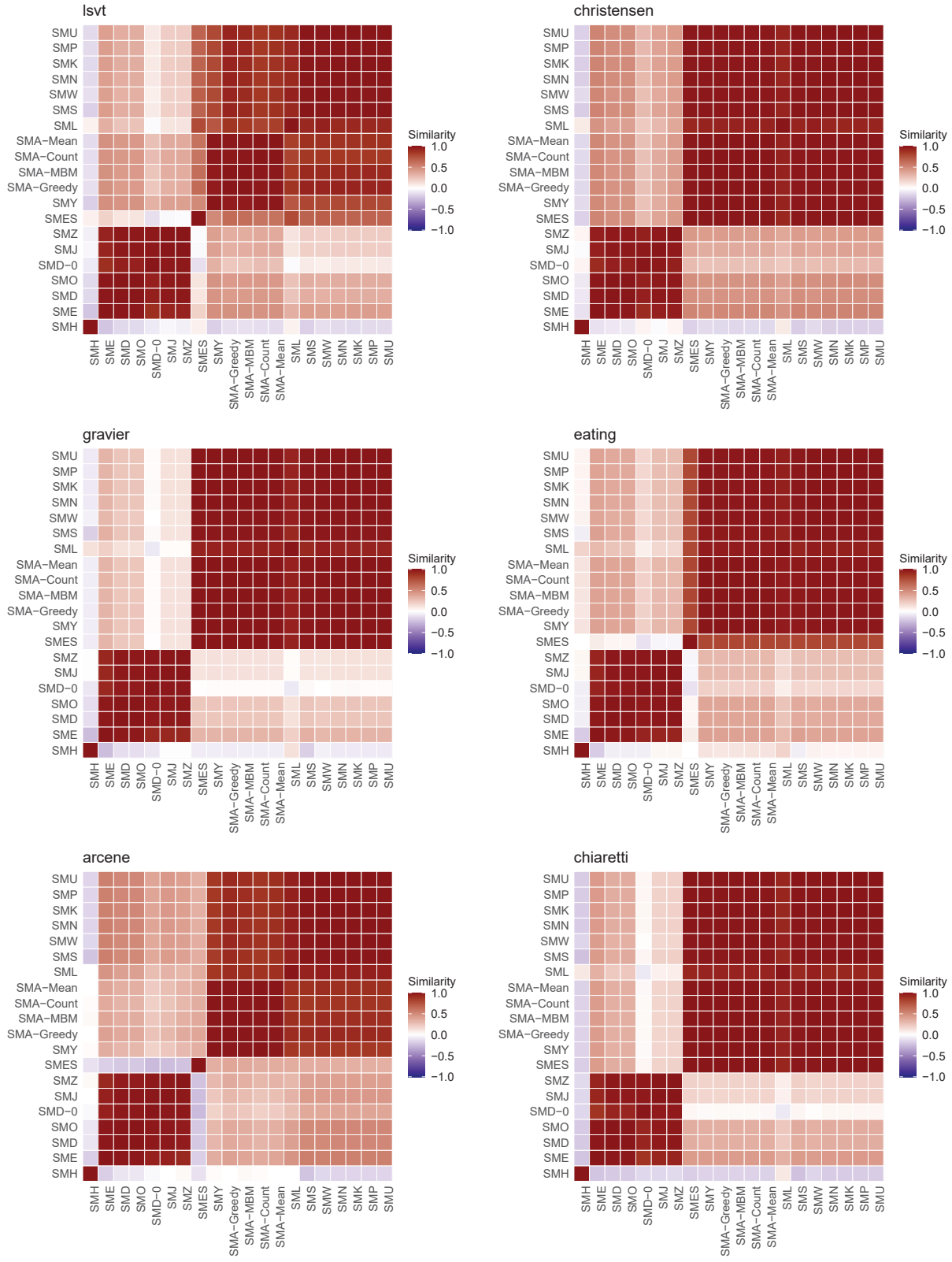


Figure A.14: Pearson correlations between all pairs of stability measures per data set. The order of the stability measures is the same as in Figure 6.5.

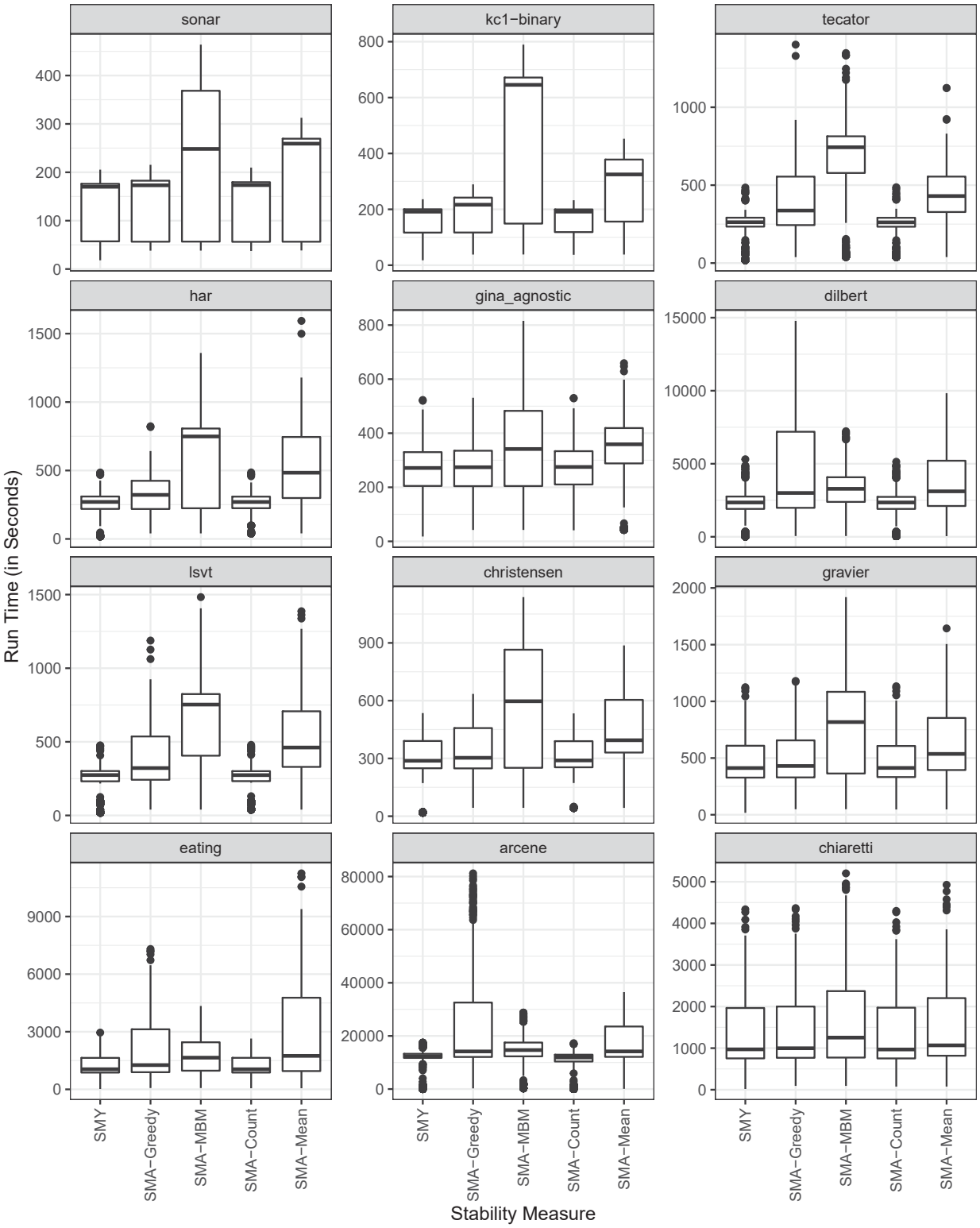


Figure A.15: Run times of the adjusted and corrected stability measures for all 12 data sets.

A.4 Finding Desirable Configurations by Multi-Criteria Tuning

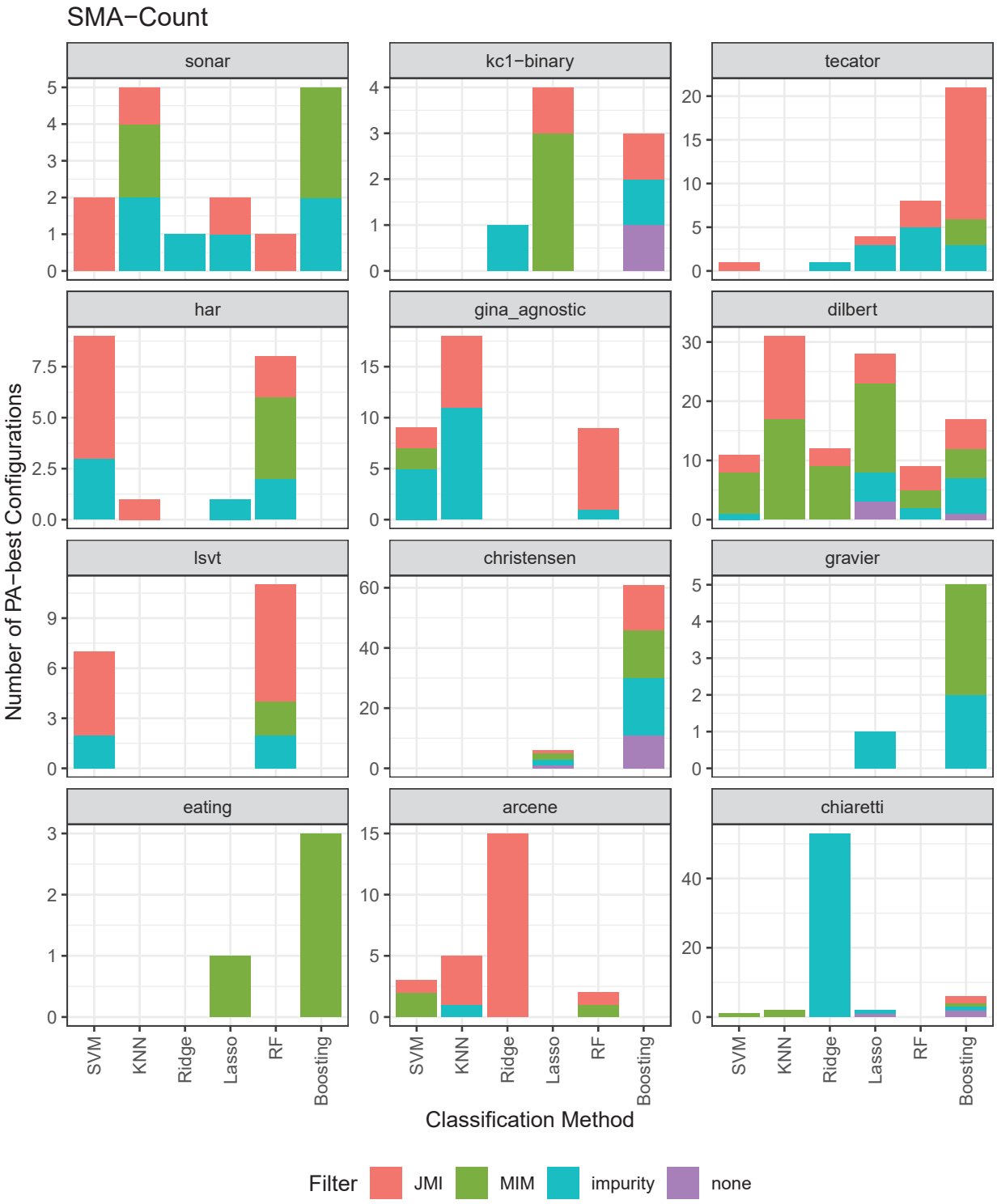


Figure A.16: Classification and filter methods of the PA-best configurations per data set for the stability measure SMA-Count.

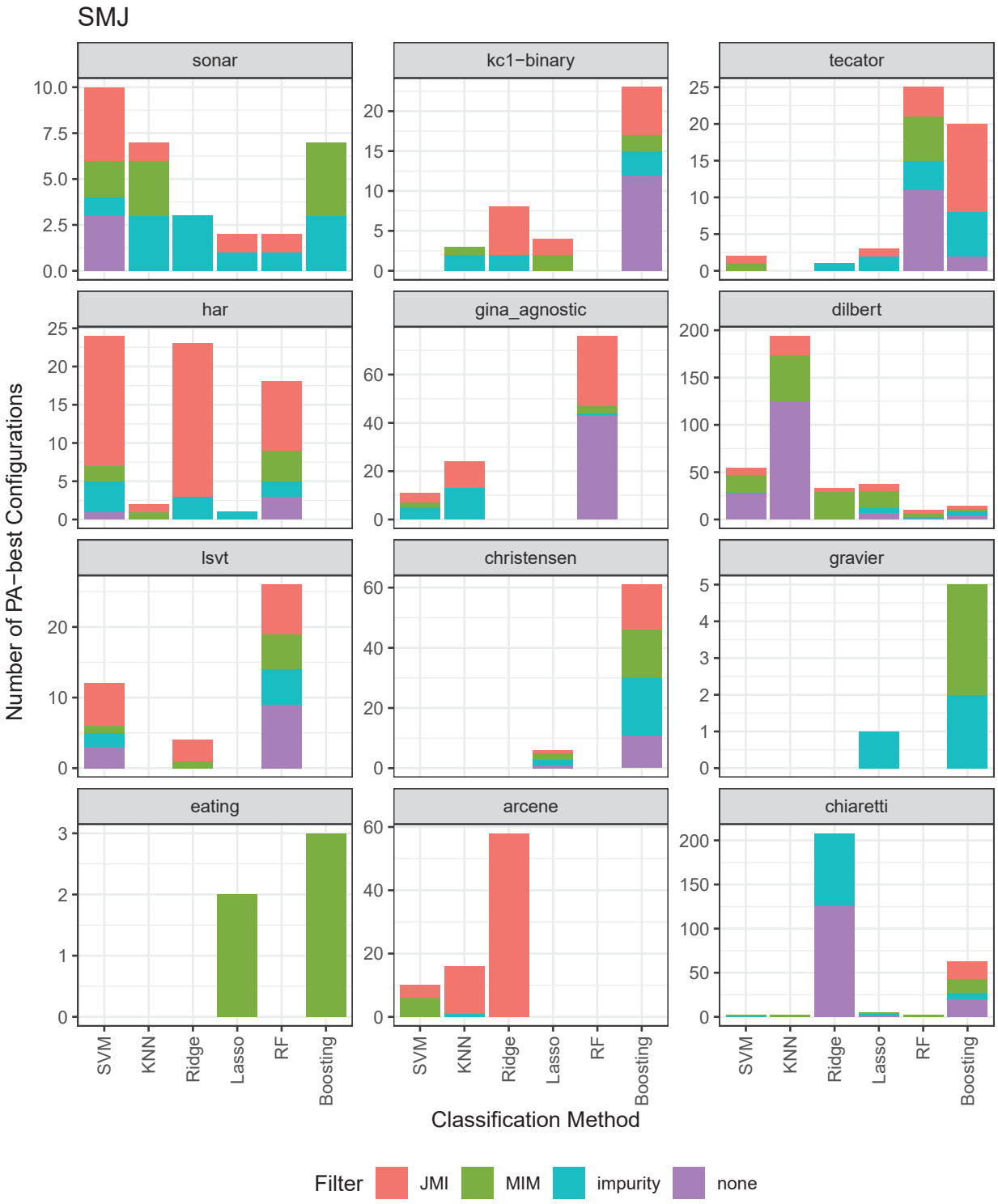


Figure A.17: Classification and filter methods of the PA-best configurations per data set for the stability measure SMJ.

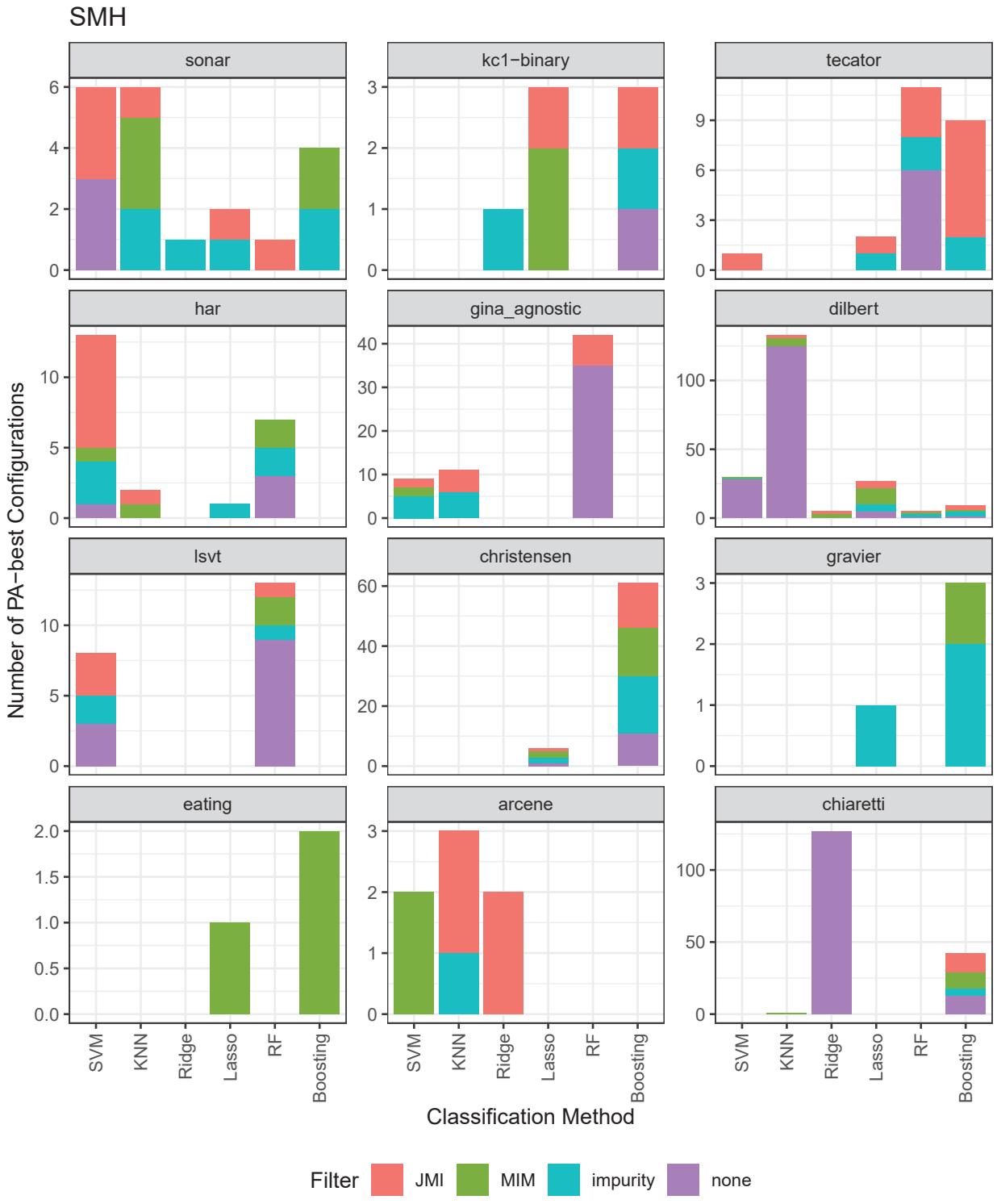


Figure A.18: Classification and filter methods of the PA-best configurations per data set for the stability measure SMH.

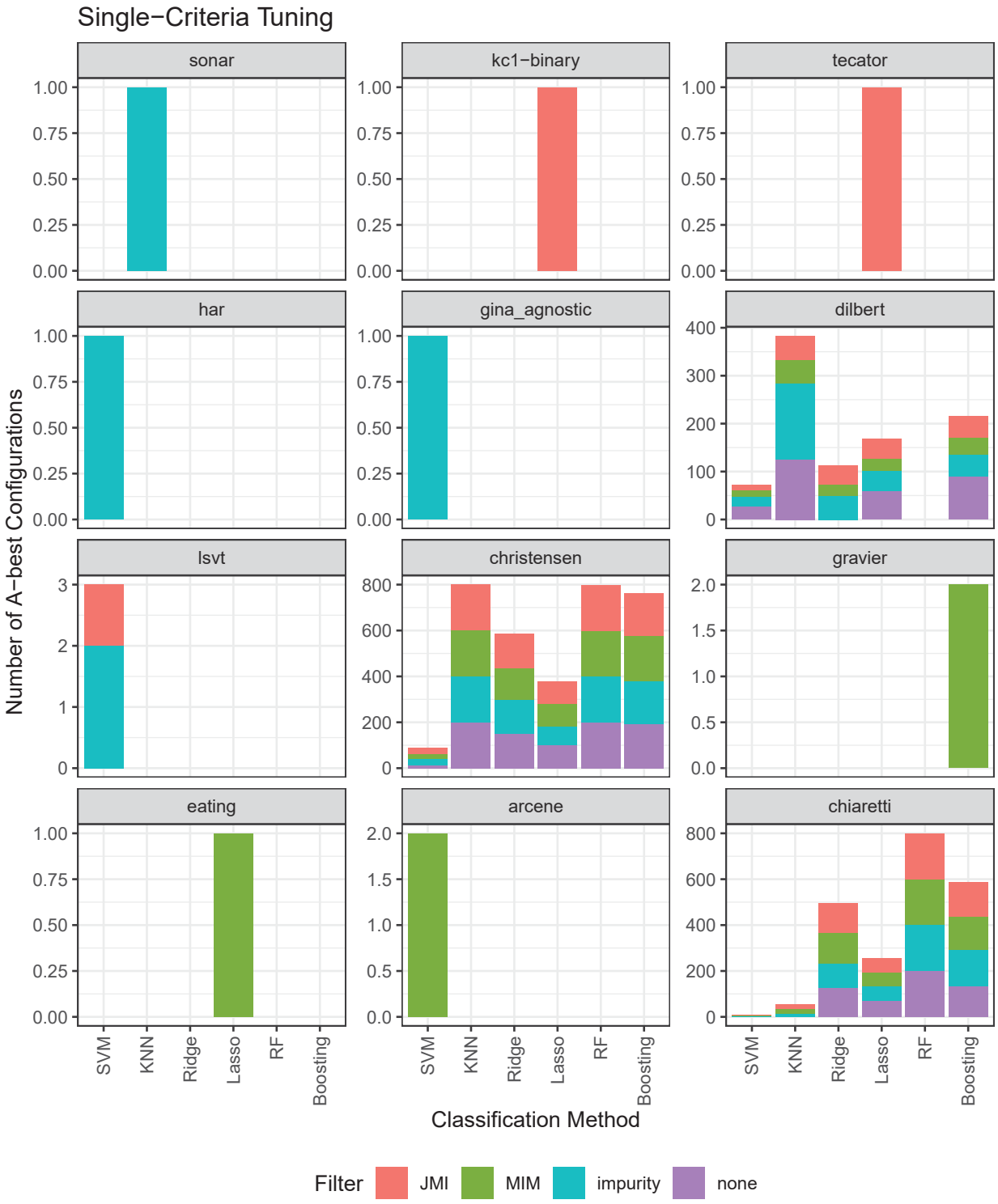


Figure A.19: Classification and filter methods of the A-best configurations per data set.

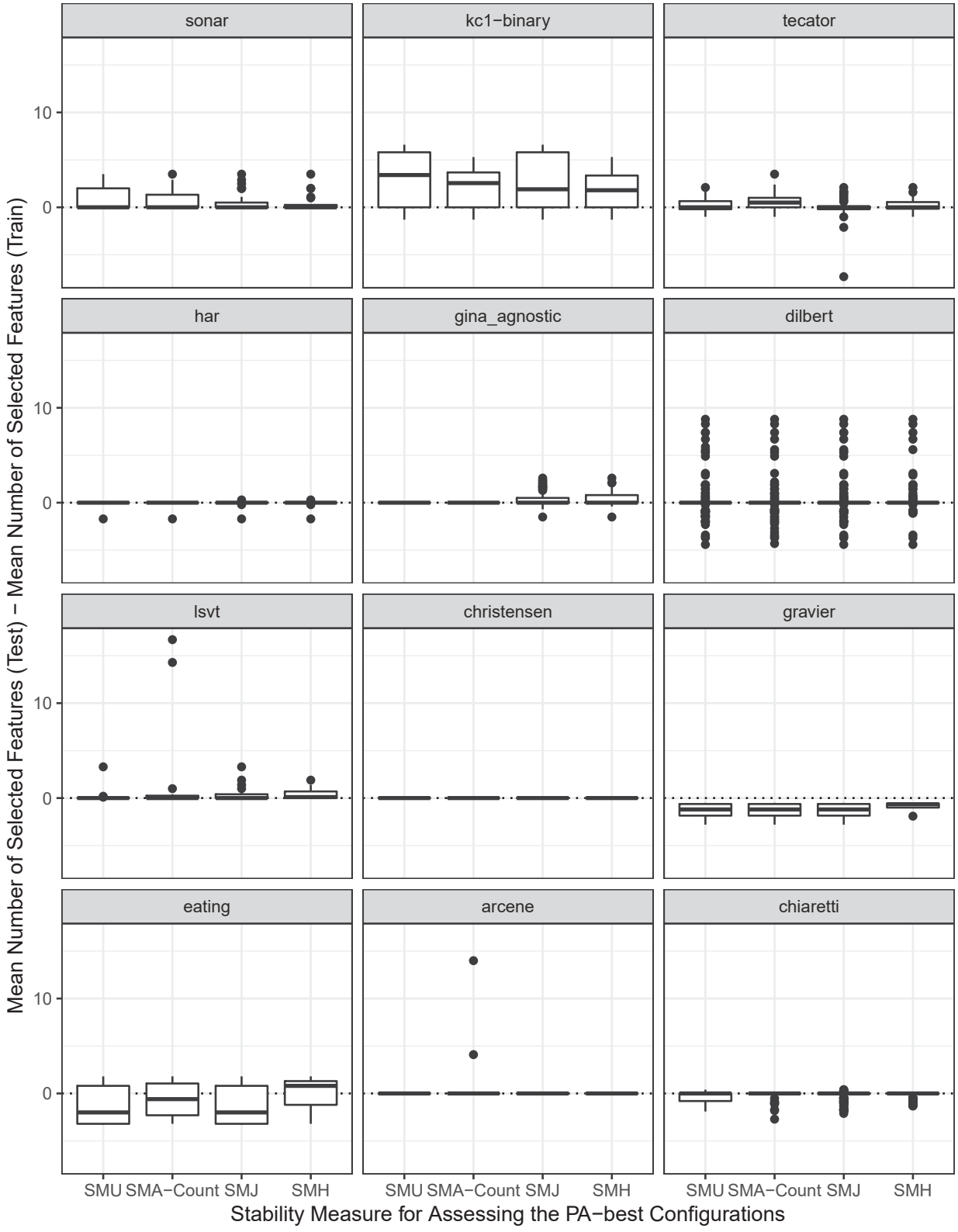


Figure A.20: Differences in mean number of selected features between training and test data for the PA-best configurations. Values above the dotted line indicate a higher mean number of selected features on the test data than on the training data.

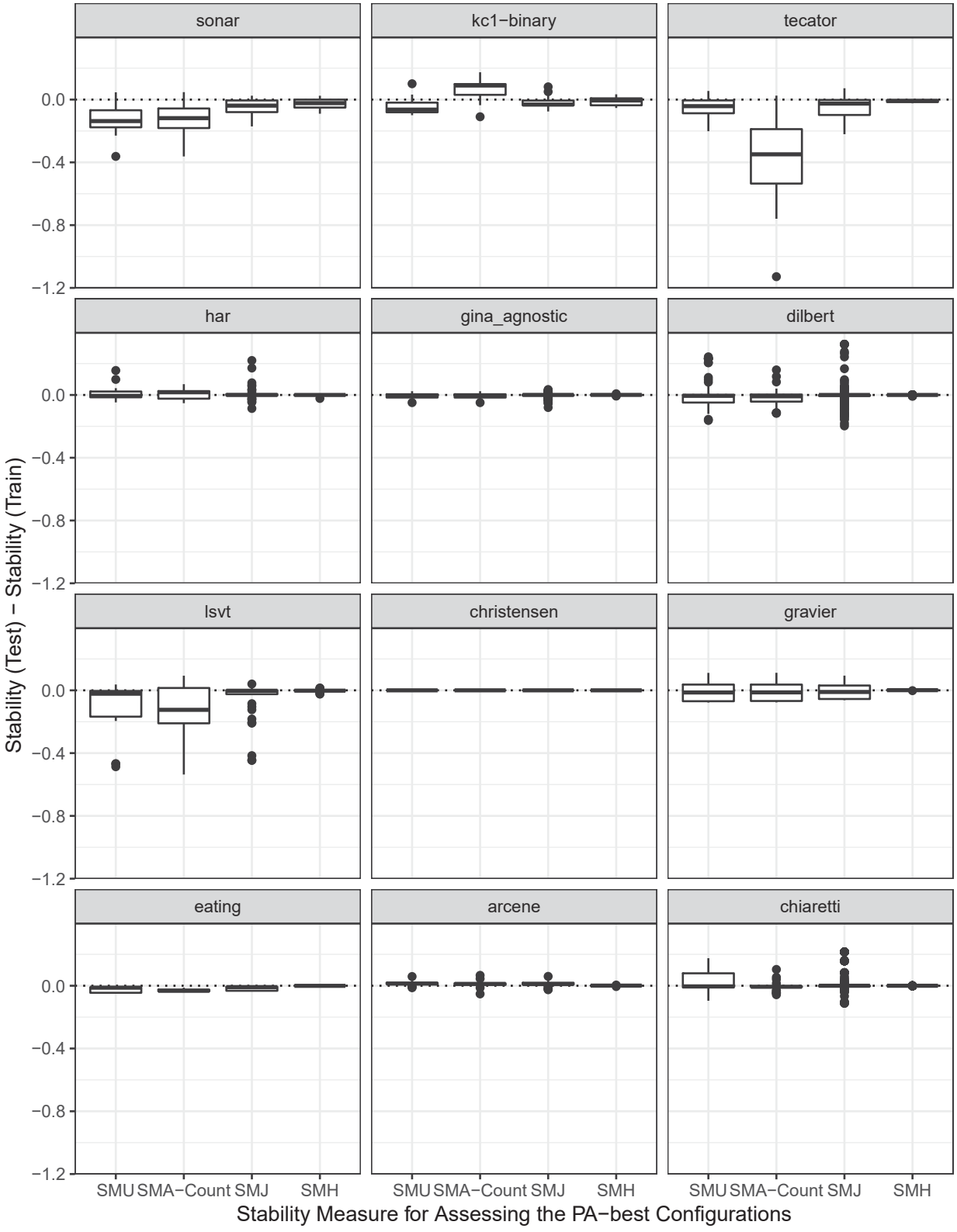


Figure A.21: Differences in stability between training and test data for the PA-best configurations. Values above the dotted indicate a higher stability on the test data than on the training data. The difference in stability is assessed with the same stability measure that was used to determine the PA-best configurations.

A.5 Fitting Models on Data Sets with Similar Features

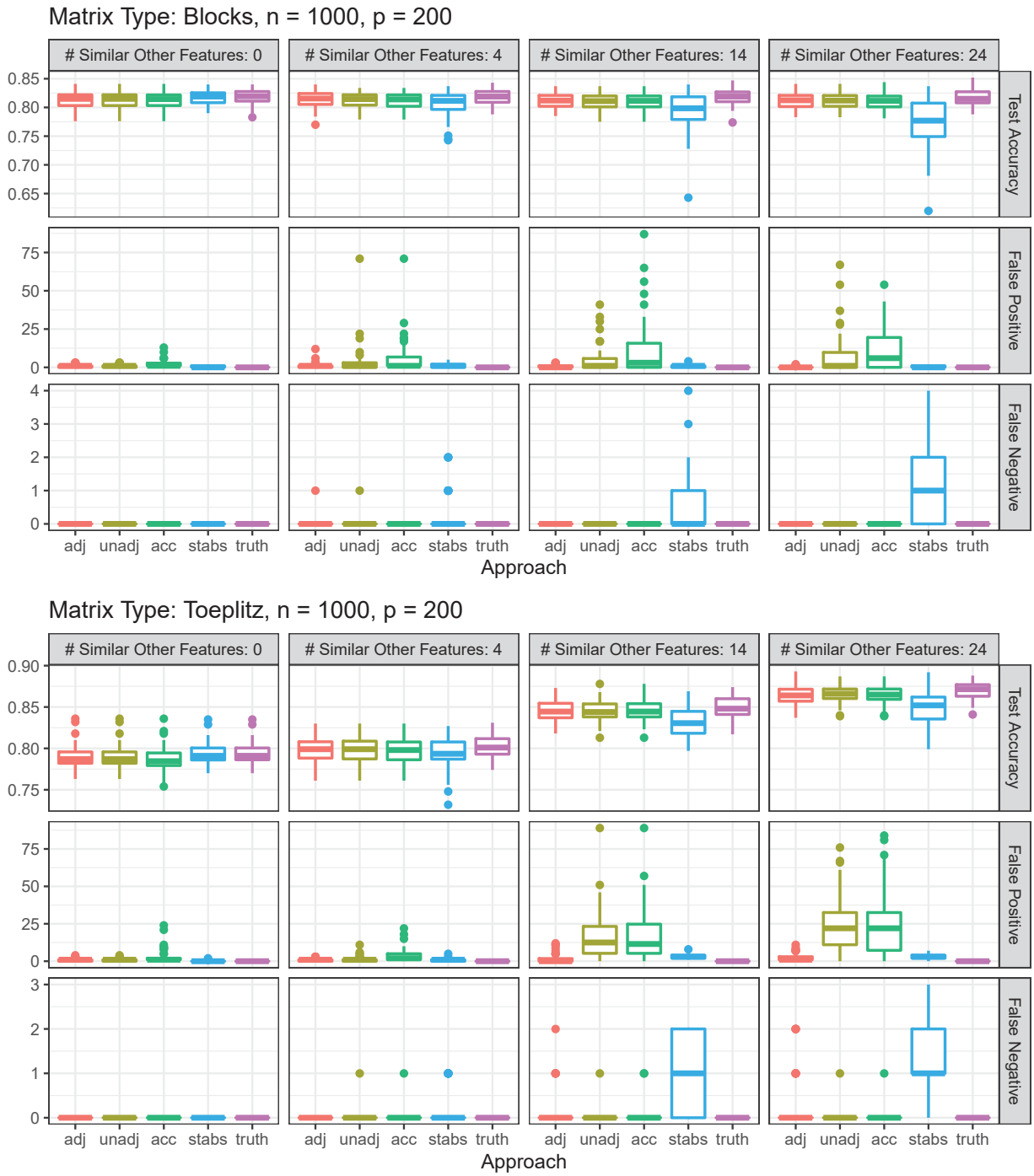


Figure A.22: Results for the simulation scenarios with $n = 1\,000$ and $p = 200$. "# Similar Other Features": Number of features in the data set that each feature is similar to (other than itself). Two features are interpreted as similar if their absolute correlation is at least 0.9. "Test Accuracy": classification accuracy on independent test data. "False Positive": number of irrelevant or redundant features that have been selected. "False Negative": number of relevant and not redundant features that have not been selected.

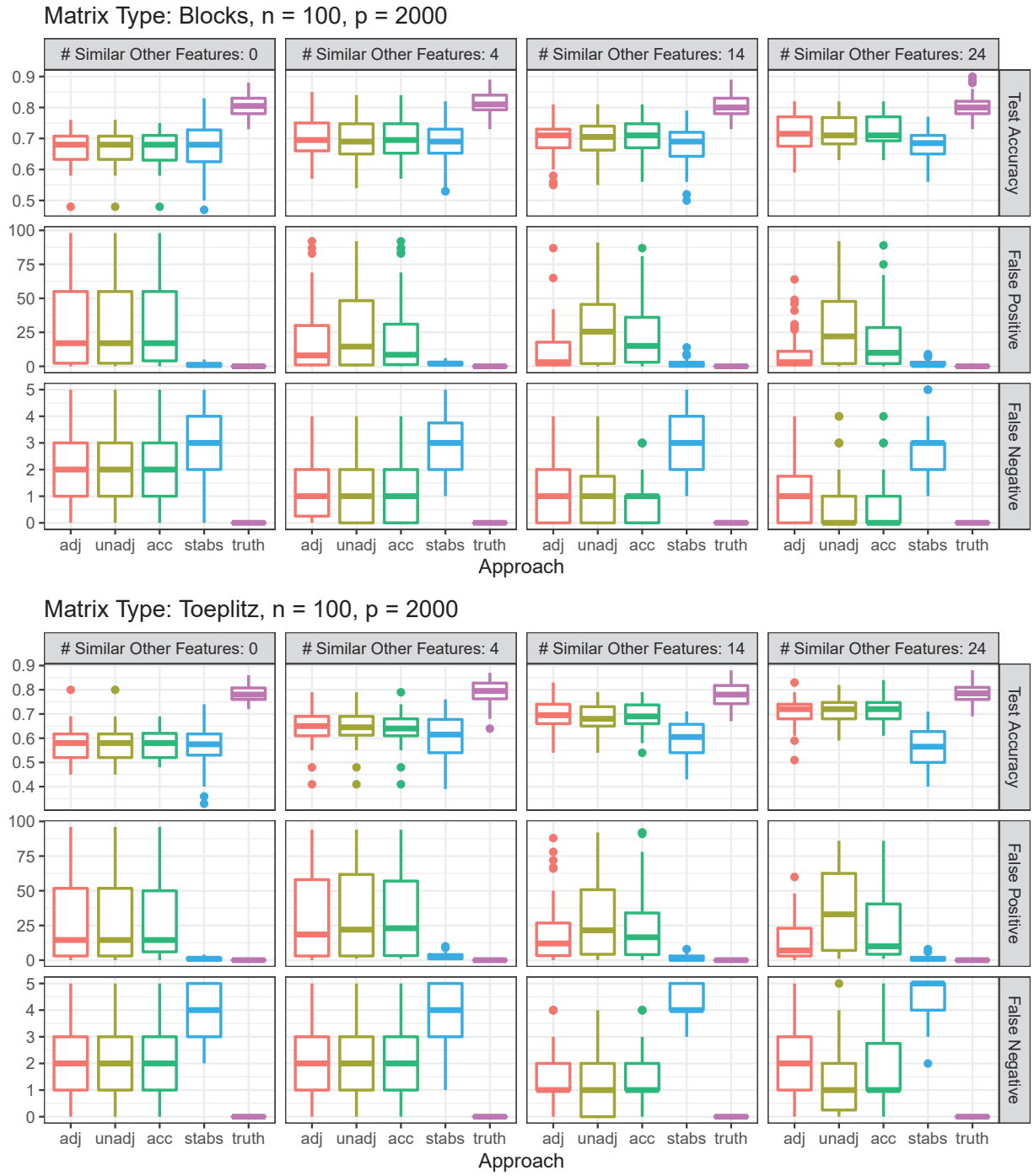


Figure A.23: Results for the simulation scenarios with $n = 100$ and $p = 2\,000$. "# Similar Other Features": Number of features in the data set that each feature is similar to (other than itself). Two features are interpreted as similar if their absolute correlation is at least 0.9. "Test Accuracy": classification accuracy on independent test data. "False Positive": number of irrelevant or redundant features that have been selected. "False Negative": number of relevant and not redundant features that have not been selected.

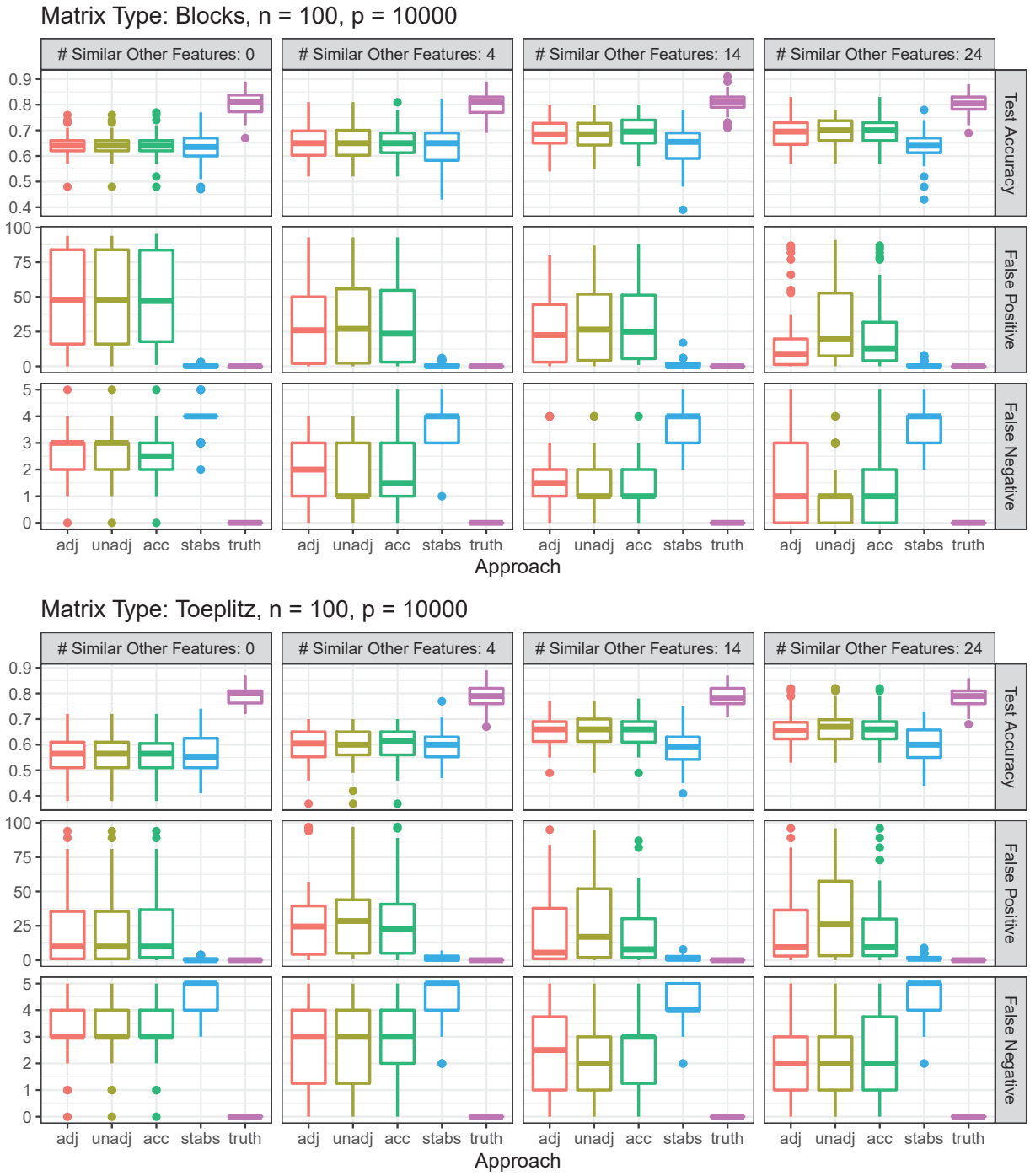


Figure A.24: Results for the simulation scenarios with $n = 100$ and $p = 10\,000$. "# Similar Other Features": Number of features in the data set that each feature is similar to (other than itself). Two features are interpreted as similar if their absolute correlation is at least 0.9. "Test Accuracy": classification accuracy on independent test data. "False Positive": number of irrelevant or redundant features that have been selected. "False Negative": number of relevant and not redundant features that have not been selected.

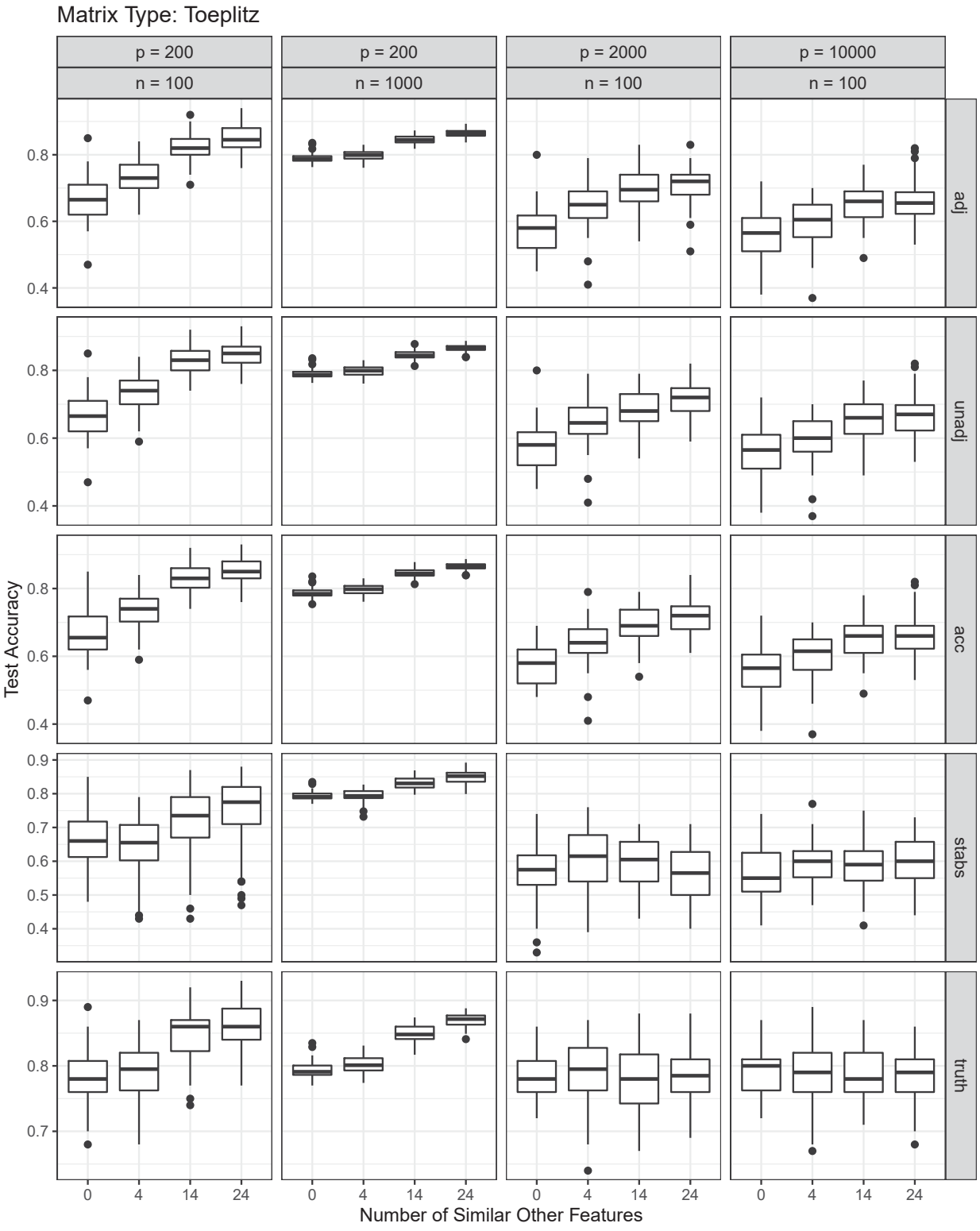


Figure A.25: Test accuracy for all considered approaches in the simulation scenarios with matrix type "Toeplitz".

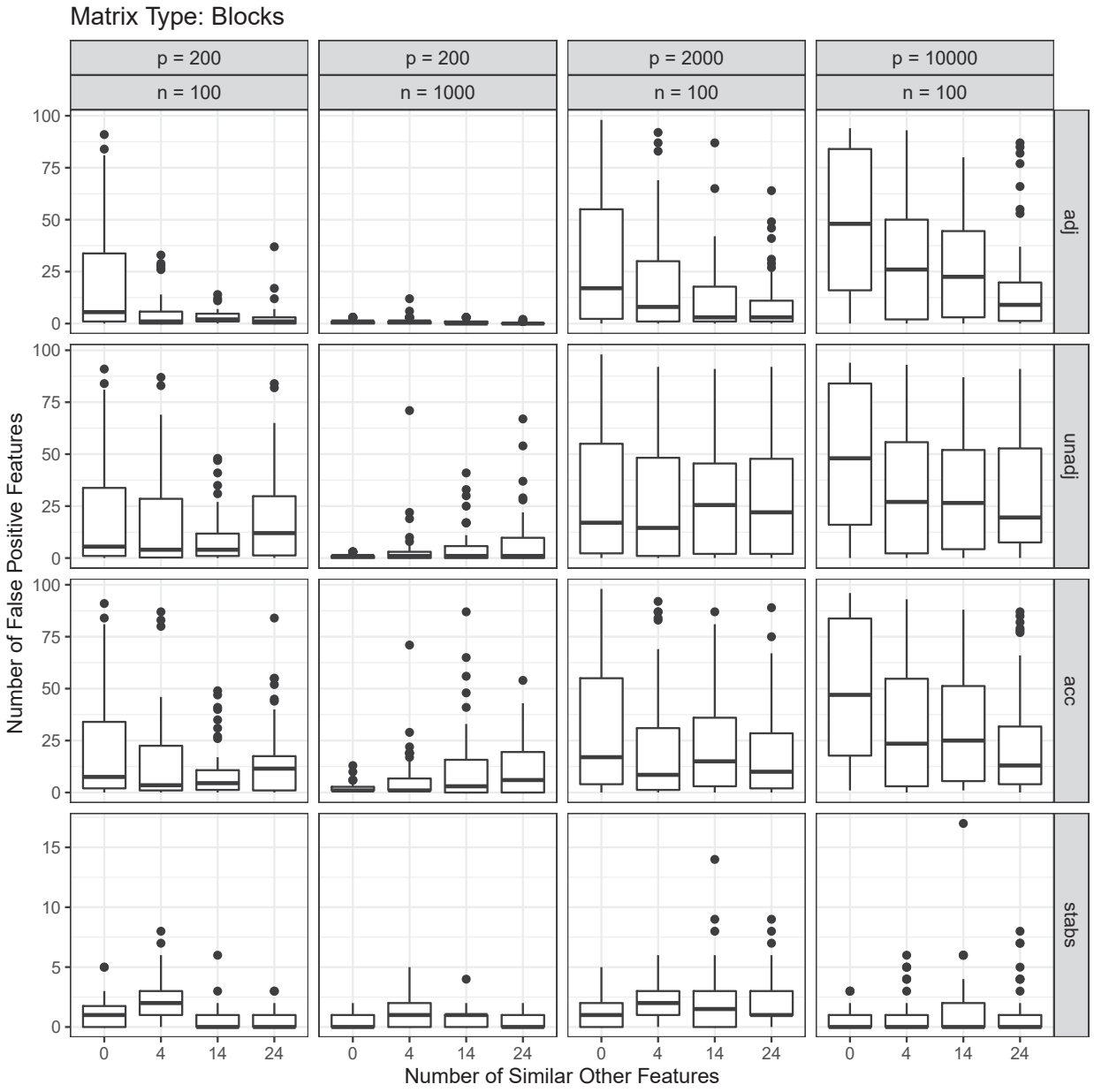


Figure A.26: Number of false positive features for all considered approaches except "truth" in the simulation scenarios with matrix type "blocks".

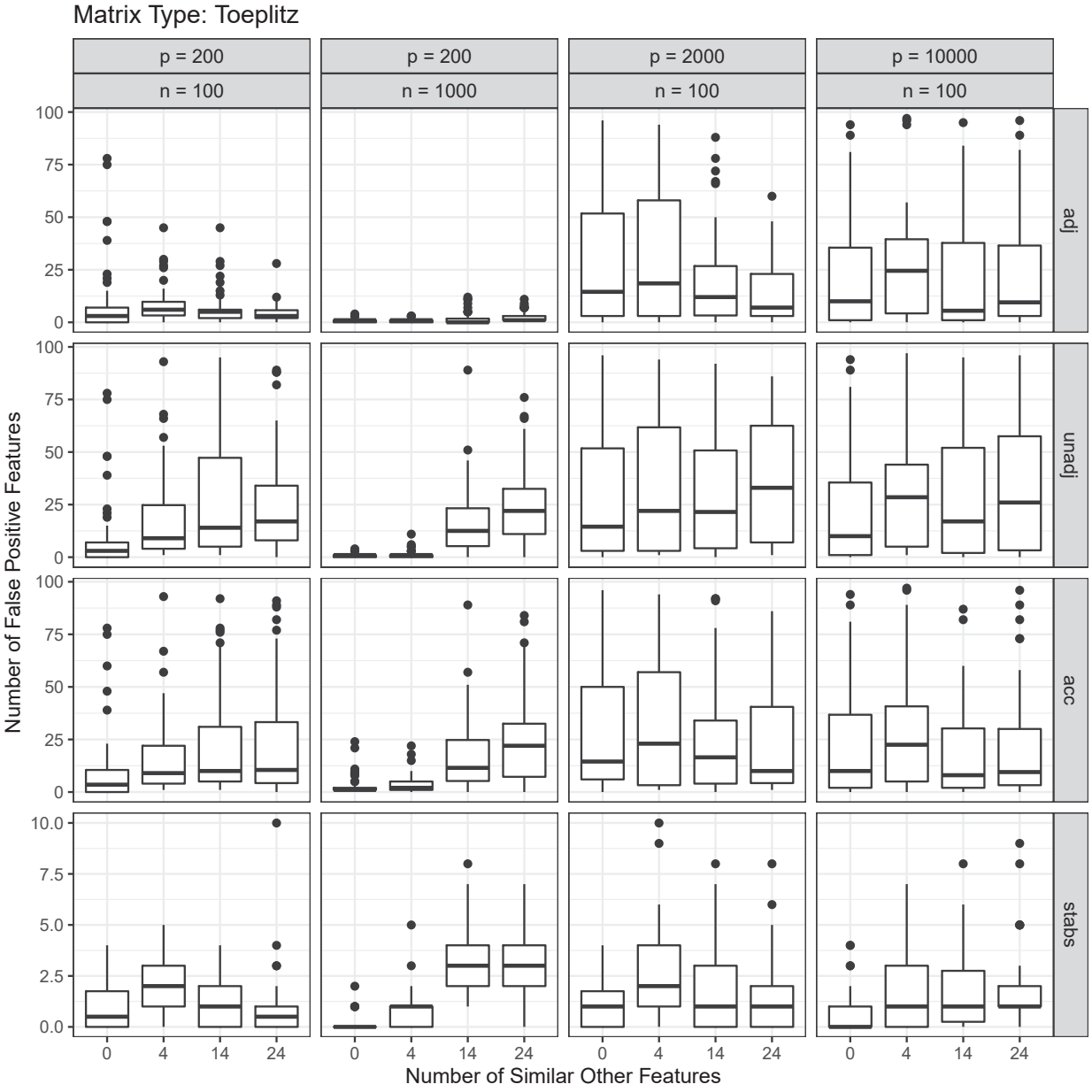


Figure A.27: Number of false positive features for all considered approaches except "truth" in the simulation scenarios with matrix type "Toeplitz".

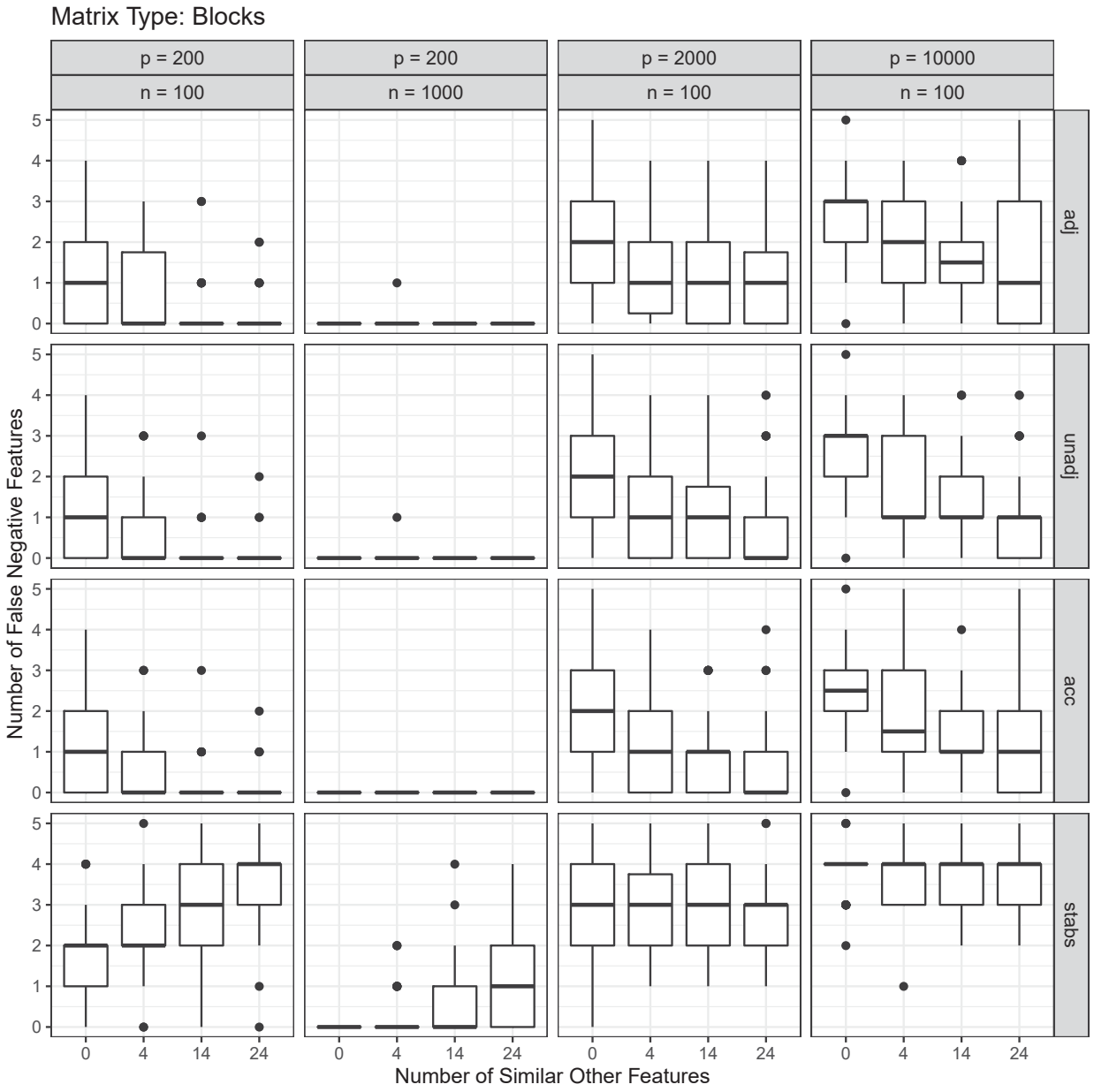


Figure A.28: Number of false negative features for all considered approaches except "truth" in the simulation scenarios with matrix type "blocks".

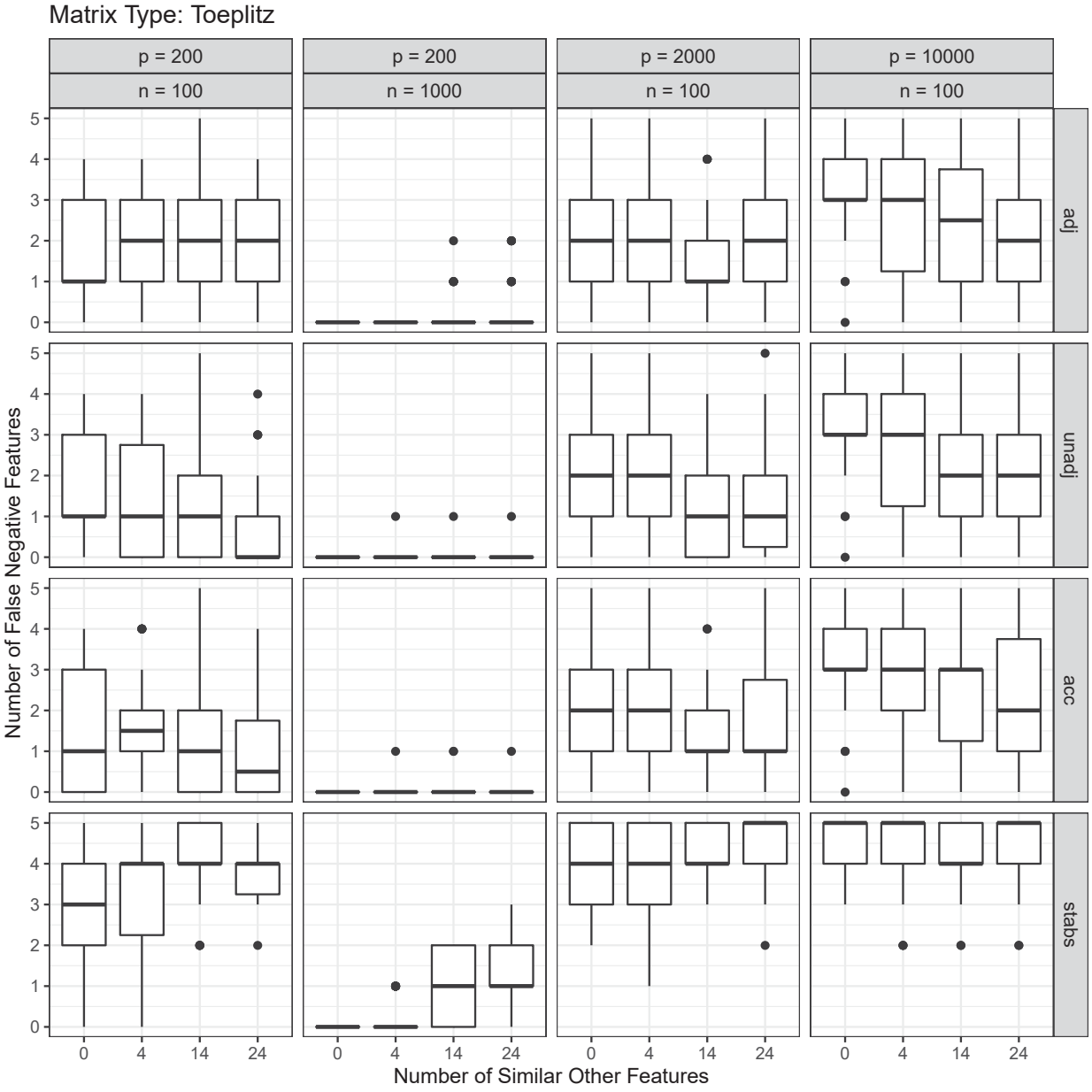


Figure A.29: Number of false negative features for all considered approaches except "truth" in the simulation scenarios with matrix type "Toeplitz".

Appendix B

Proofs of Properties

For all proofs, it is assumed that the stability measures are well-defined. This means that implicitly only such feature sets $V_1, \ldots, V_m$ and similarity structures between features are considered, for which the stability measures are well-defined. In Section B.1, for each stability measure, the necessary conditions for it to be well-defined are stated.

B.1 Bounds

SMJ

SMJ is well-defined if and only if $|V_i \cup V_j| \neq 0 \, \forall i < j$, that is, if not more than one of the sets $V_1, \ldots, V_m$ is empty.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$(V_i \cap V_j) \subseteq (V_i \cup V_j) \; \forall i < j \quad \Rightarrow \quad |V_i \cap V_j| \leq |V_i \cup V_j| \; \forall i < j$$
$$\Rightarrow \quad \text{SMJ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j|}{|V_i \cup V_j|}}_{\leq 1} \leq 1$$

For $V_1 = \ldots = V_m$:

$$(V_i \cap V_j) = (V_i \cup V_j) \; \forall i < j \Rightarrow \quad |V_i \cap V_j| = |V_i \cup V_j| \; \forall i < j$$
$$\Rightarrow \quad \text{SMJ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j|}{|V_i \cup V_j|}}_{=1} = 1$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$|V_i \cap V_j| \geq 0 \text{ and } |V_i \cup V_j| \geq 0 \; \forall i < j$$
$$\Rightarrow \quad \text{SMJ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j|}{|V_i \cup V_j|}}_{\geq 0} \geq 0$$

For $V_1, \ldots, V_m$ pairwise disjoint:

$$|V_i \cap V_j| = 0 \text{ and } |V_i \cup V_j| \geq 0 \; \forall i < j$$
$$\Rightarrow \quad \text{SMJ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j|}{|V_i \cup V_j|}}_{=0} = 0$$

SMD

SMD is well-defined if and only if $|V_i| + |V_j| \neq 0 \, \forall i < j$, that is, if not more than one of the sets $V_1, \ldots, V_m$ is empty.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$(V_i \cap V_j) \subseteq V_i \text{ and } (V_i \cap V_j) \subseteq V_j \; \forall i < j \quad \Rightarrow \quad 2\,|V_i \cap V_j| \leq |V_i| + |V_j| \; \forall i < j$$
$$\Rightarrow \quad \text{SMD} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{2\,|V_i \cap V_j|}{|V_i| + |V_j|}}_{\leq 1} \leq 1$$

For $V_1 = \ldots = V_m$:

$$(V_i \cap V_j) = V_i \text{ and } (V_i \cap V_j) = V_j \; \forall i < j \quad \Rightarrow \quad 2\,|V_i \cap V_j| = |V_i| + |V_j| \; \forall i < j$$
$$\Rightarrow \quad \text{SMD} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{2\,|V_i \cap V_j|}{|V_i| + |V_j|}}_{=1} = 1$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$|V_i \cap V_j| \geq 0, \; |V_i| \geq 0, \text{ and } |V_j| \geq 0 \; \forall i < j$$
$$\Rightarrow \quad \text{SMD} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{2\,|V_i \cap V_j|}{|V_i| + |V_j|}}_{\geq 0} \geq 0$$

For $V_1, \ldots, V_m$ pairwise disjoint:

$$|V_i \cap V_j| = 0, \; |V_i| \geq 0, \text{ and } |V_j| \geq 0 \; \forall i < j$$
$$\Rightarrow \quad \text{SMD} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{2\,|V_i \cap V_j|}{|V_i| + |V_j|}}_{=0} = 0$$

SMO

SMO is well-defined if and only if $\sqrt{|V_i| \cdot |V_j|} \neq 0 \, \forall i < j$, that is, if none of the sets $V_1, \ldots, V_m$ is empty.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\begin{aligned} & (V_i \cap V_j) \subseteq V_i \text{ and } (V_i \cap V_j) \subseteq V_j \; \forall i < j \\ \Rightarrow \quad & |V_i \cap V_j| = \underbrace{\sqrt{|V_i \cap V_j|}}_{\leq \sqrt{|V_i|}} \cdot \underbrace{\sqrt{|V_i \cap V_j|}}_{\leq \sqrt{|V_j|}} \leq \sqrt{|V_i| \cdot |V_j|} \; \forall i < j \\ \Rightarrow \quad & \text{SMO} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j|}{\sqrt{|V_i| \cdot |V_j|}}}_{\leq 1} \leq 1 \end{aligned}$$

For $V_1 = \ldots = V_m$:

$$\begin{aligned} & (V_i \cap V_j) = V_i \text{ and } (V_i \cap V_j) = V_j \; \forall i < j \quad \Rightarrow \quad |V_i \cap V_j| = \sqrt{|V_i| \cdot |V_j|} \; \forall i < j \\ \Rightarrow \quad & \text{SMO} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j|}{\sqrt{|V_i| \cdot |V_j|}}}_{=1} = 1 \end{aligned}$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\begin{aligned} & |V_i \cap V_j| \geq 0, \; |V_i| \geq 0, \text{ and } \; |V_j| \geq 0 \; \forall i < j \\ \Rightarrow \quad & \text{SMO} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j|}{\sqrt{|V_i| \cdot |V_j|}}}_{\geq 0} \geq 0 \end{aligned}$$

For $V_1, \ldots, V_m$ pairwise disjoint:

$$\begin{aligned} & |V_i \cap V_j| = 0, \; |V_i| \geq 0, \text{ and } \; |V_j| \geq 0 \; \forall i < j \\ \Rightarrow \quad & \text{SMO} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j|}{\sqrt{|V_i| \cdot |V_j|}}}_{=0} = 0 \end{aligned}$$

SMH

SMH is well-defined for arbitrary sets $V_1, \ldots, V_m$.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\begin{aligned} & |V_i \cap V_j| + \left|V_i^c \cap V_j^c\right| \leq p \ \forall i < j \\ \Rightarrow \quad & \text{SMH} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| + \left|V_i^c \cap V_j^c\right|}{p}}_{\leq 1} \leq 1 \end{aligned}$$

For $V_1 = \ldots = V_m$:

$$\begin{aligned} & |V_i \cap V_j| + \left|V_i^c \cap V_j^c\right| = |V_i| + |V_i^c| = p \ \forall i < j \\ \Rightarrow \quad & \text{SMH} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| + \left|V_i^c \cap V_j^c\right|}{p}}_{=1} = 1 \end{aligned}$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\begin{aligned} & |V_i \cap V_j| \geq 0 \text{ and } \left|V_i^c \cap V_j^c\right| \geq 0 \ \forall i < j \\ \Rightarrow \quad & \text{SMH} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| + \left|V_i^c \cap V_j^c\right|}{p}}_{\geq 0} \geq 0 \end{aligned}$$

For $m = 2$ and $V_1 = \{X_1, \ldots, X_p\} \setminus V_2$:

$$\begin{aligned} & |V_1 \cap V_2| = 0 \text{ and } |V_1^c \cap V_2^c| = 0 \\ \Rightarrow \quad & \text{SMH} = \frac{|V_1 \cap V_2| + |V_1^c \cap V_2^c|}{p} = 0 \end{aligned}$$

For $m > 2$, the lower bound is not tight. It is not possible to choose $V_1, \ldots, V_m$ such that $|V_i \cap V_j| = 0$ and $\left|V_i^c \cap V_j^c\right| = 0 \ \forall i < j$ for $m > 2$.

SML

SML is well-defined if and only if $\min\{|V_i|, |V_j|\} - \max\{0, |V_i| + |V_j| - p\} \neq 0 \, \forall i < j$, that is, if none of the sets $V_1, \ldots, V_m$ is empty or contains all features.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

- $|V_i \cap V_j| \leq \min\{|V_i|, |V_j|\}$
- $\frac{|V_i| \cdot |V_j|}{p} \geq \max\{0, |V_i| + |V_j| - p\}$ because
  - $\frac{|V_i| \cdot |V_j|}{p} \geq 0$ and
  - $\frac{|V_i| \cdot |V_j|}{p} \geq |V_i| + |V_j| - p \Leftrightarrow |V_i| \cdot |V_j| \geq p \cdot |V_i| + p \cdot |V_j| - p^2 \Leftrightarrow p(p - |V_j|) \geq |V_i|(p - |V_j|) \Leftrightarrow p \geq |V_i|$

$$\Rightarrow \quad \text{SML} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\min\{|V_i|, |V_j|\} - \max\{0, |V_i| + |V_j| - p\}}}_{\leq 1} \leq 1$$

For $V_1 = \ldots = V_m$:

$$\text{SML} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_1| - \frac{|V_1|^2}{p}}{|V_1| - \max\{0, 2|V_1| - p\}} = \begin{cases} 1 - \frac{|V_1|}{p}, & |V_1| < \frac{p}{2} \\ \frac{|V_1|}{p}, & |V_1| \geq \frac{p}{2} \end{cases}$$

As $0 < |V_1| < p$, the upper bound is not tight. For $p \to \infty$, the bound can be reached asymptotically.
Lower bound:
For arbitrary sets $V_1, \ldots, V_m$ with w.l.o.g. $|V_i| \leq |V_j| \ \forall i < j$:
If $|V_i| + |V_j| \leq p$, then

$$\frac{\overbrace{|V_i \cap V_j|}^{\geq 0} - \frac{|V_i| \cdot |V_j|}{p}}{\underbrace{\min\{|V_i|, |V_j|\}}_{=|V_i|} - \underbrace{\max\{0, |V_i| + |V_j| - p\}}_{=0}} \geq \frac{-\frac{|V_i| \cdot |V_j|}{p}}{|V_i|} = -\frac{|V_j|}{p} \geq -1.$$

If $|V_i| + |V_j| > p$, then $|V_i \cap V_j| > 0$, more precisely $|V_i \cap V_j| \geq |V_i| + |V_j| - p$. Show that

$$\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\min\{|V_i|, |V_j|\} - \max\{0, |V_i| + |V_j| - p\}} \geq -1$$

by proof by contradiction:

$$\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\underbrace{\min\{|V_i|, |V_j|\}}_{=|V_i|} - \underbrace{\max\{0, |V_i| + |V_j| - p\}}_{=|V_i|+|V_j|-p}} = \frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{p - |V_j|} < -1$$

$$\Leftrightarrow \quad |V_i \cap V_j| < |V_j| - p + |V_i| \cdot \underbrace{\frac{|V_j|}{p}}_{\leq 1} \leq |V_i| + |V_j| - p \leq |V_i \cap V_j| \ \lightning$$

$$\Rightarrow \quad \text{SML} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\min\{|V_i|, |V_j|\} - \max\{0, |V_i| + |V_j| - p\}}}_{\geq -1} \geq -1$$

For $m = 2$ and V_1, V_2 with $|V_1 \cap V_2| = 0$, $|V_1| = 1$, and $|V_2| = p - 1$:

$$\text{SML} = \frac{|V_1 \cap V_2| - \frac{|V_1| \cdot |V_2|}{p}}{\min\{|V_1|, |V_2|\} - \max\{0, |V_1| + |V_2| - p\}} = \frac{-\frac{p-1}{p}}{1} = -1 + \frac{1}{p} \overset{p \to \infty}{\longrightarrow} -1$$

For $m > 2$, the lower bound is not tight.

SMW

SMW is well-defined if and only if $\min\{|V_i|, |V_j|\} - \frac{|V_i|\cdot|V_j|}{p} \neq 0 \, \forall i < j$, that is, if none of the sets $V_1, \ldots, V_m$ is empty or contains all features.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\begin{aligned} & |V_i \cap V_j| \leq \min\{|V_i|, |V_j|\} \;\; \forall i < j \\ \Rightarrow \quad & \text{SMW} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\min\{|V_i|, |V_j|\} - \frac{|V_i|\cdot|V_j|}{p}}}_{\leq 1} \leq 1 \end{aligned}$$

For $V_1 = \ldots = V_m$:

$$\begin{aligned} & |V_i \cap V_j| = \min\{|V_i|, |V_j|\} \;\; \forall i < j \\ \Rightarrow \quad & \text{SMW} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\min\{|V_i|, |V_j|\} - \frac{|V_i|\cdot|V_j|}{p}}}_{=1} = 1 \end{aligned}$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$ with w.l.o.g. $|V_i| \leq |V_j| \;\; \forall i < j$:

$$\frac{\overbrace{|V_i \cap V_j|}^{\geq 0} - \frac{|V_i|\cdot|V_j|}{p}}{\underbrace{\min\{|V_i|, |V_j|\}}_{=|V_i|} - \frac{|V_i|\cdot|V_j|}{p}} \geq \frac{-\frac{|V_i|\cdot|V_j|}{p}}{|V_i| - \frac{|V_i|\cdot|V_j|}{p}} = \frac{-\frac{|V_j|}{p}}{1 - \frac{|V_j|}{p}} = \frac{1}{1 - \underbrace{\frac{p}{|V_j|}}_{|V_j|<p}} \geq \frac{1}{1 - \frac{p}{p-1}} = 1 - p$$

$$\Rightarrow \quad \text{SMW} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\min\{|V_i|, |V_j|\} - \frac{|V_i|\cdot|V_j|}{p}}}_{\geq 1-p} \geq 1 - p$$

For $m = 2$ and V_1, V_2 with $|V_1 \cap V_2| = 0$, $|V_1| = 1$, and $|V_2| = p - 1$, SMW attains the lower bound $1 - p$. For $m > 2$, the lower bound is not tight.

SMU

SMU is well-defined if and only if $\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i|\cdot|V_j|}{p} \neq 0 \, \forall i < j$, that is, if none of the sets $V_1, \ldots, V_m$ is empty and if not more than one set contains all features.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\begin{aligned} & |V_i \cap V_j| \leq \sqrt{|V_i| \cdot |V_j|} \; \forall i < j \text{ (see SMO)} \\ \Rightarrow \quad & \text{SMU} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i|\cdot|V_j|}{p}}}_{\leq 1} \leq 1 \end{aligned}$$

For $V_1 = \ldots = V_m$:

$$|V_i \cap V_j| = \sqrt{|V_i| \cdot |V_j|} \; \forall i < j \text{ (see SMO)}$$

$$\Rightarrow \quad \text{SMU} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i| \cdot |V_j|}{p}}}_{=1} = 1$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i| \cdot |V_j|}{p}} \geq -1$$

as shown by proof by contradiction:

$$\underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i| \cdot |V_j|}{p}}}_{\geq 0 \; (\star)} < -1$$

$$\Leftrightarrow \quad |V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p} < \frac{|V_i| \cdot |V_j|}{p} - \sqrt{|V_i| \cdot |V_j|}$$

$$\Leftrightarrow \quad |V_i \cap V_j| < 2 \cdot \frac{|V_i| \cdot |V_j|}{p} - \sqrt{|V_i| \cdot |V_j|}$$

$$\Leftrightarrow \quad |V_i \cap V_j| < \frac{\sqrt{|V_i| \cdot |V_j|}}{p} \left(2 \cdot \sqrt{|V_i| \cdot |V_j|} - p \right) \overset{(\star\star)}{\leq} \underbrace{\frac{\sqrt{|V_i| \cdot |V_j|}}{p}}_{\leq 1} (|V_i| + |V_j| - p)$$

$$\leq |V_i| + |V_j| - p \leq |V_i \cap V_j| \; \lightning$$

($\star$): $|V_i| \leq p$ and $|V_j| \leq p \Rightarrow \sqrt{|V_i| \cdot |V_j|} \leq p \Leftrightarrow \frac{|V_i| \cdot |V_j|}{p} \leq \sqrt{|V_i| \cdot |V_j|}$
($\star\star$) holds because

$$2 \cdot \sqrt{|V_i| \cdot |V_j|} \leq |V_i| + |V_j| \quad \Leftrightarrow \quad 0 \leq \left(\sqrt{|V_i|} - \sqrt{|V_j|} \right)^2.$$

$$\Rightarrow \quad \text{SMU} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i| \cdot |V_j|}{p}}}_{\geq -1} \geq -1$$

For $m = 2$ and V_1, V_2 with $|V_1 \cap V_2| = 0$, $|V_1| = \frac{p}{2}$, and $|V_2| = \frac{p}{2}$, SMU attains the lower bound -1. For $m > 2$, the lower bound is not tight.

SMK

SMK is well-defined if and only if $\frac{|V_i|+|V_j|}{2} - \frac{|V_i|\cdot|V_j|}{p} \neq 0 \, \forall i < j$, that is, if not more than one of the sets $V_1, \ldots, V_m$ is empty or contains all features.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$|V_i \cap V_j| \leq \frac{|V_i| + |V_j|}{2} \; \forall i < j \text{ (see SMD)}$$

$$\Rightarrow \quad \text{SMK} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\frac{|V_i|+|V_j|}{2} - \frac{|V_i|\cdot|V_j|}{p}}}_{\leq 1} \leq 1$$

For $V_1 = \ldots = V_m$:

$$|V_i \cap V_j| = \frac{|V_i| + |V_j|}{2} \; \forall i < j \text{ (see SMD)}$$

$$\Rightarrow \quad \text{SMK} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\frac{|V_i|+|V_j|}{2} - \frac{|V_i|\cdot|V_j|}{p}}}_{=1} = 1$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\frac{|V_i|+|V_j|}{2} - \frac{|V_i|\cdot|V_j|}{p}} \geq -1$$

as shown by proof by contradiction:

$$\underbrace{\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\frac{|V_i| + |V_j|}{2} - \frac{|V_i| \cdot |V_j|}{p}}}_{\geq 0 \text{ (see SMU)}} < -1$$

$$\Leftrightarrow \quad |V_i \cap V_j| < 2\frac{|V_i| \cdot |V_j|}{p} - \underbrace{\frac{|V_i| + |V_j|}{2}}_{\geq \sqrt{|V_i|\cdot|V_j|}} \leq 2\frac{|V_i| \cdot |V_j|}{p} - \sqrt{|V_i| \cdot |V_j|} \leq |V_i| + |V_j| - p, \; \lightning$$

see proof of lower bound for SMU.

$$\Rightarrow \quad \text{SMK} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i|\cdot|V_j|}{p}}{\frac{|V_i|+|V_j|}{2} - \frac{|V_i|\cdot|V_j|}{p}}}_{\geq -1} \geq -1$$

For $m = 2$ and V_1, V_2 with $|V_1 \cap V_2| = 0$, $|V_1| = \frac{p}{2}$, and $|V_2| = \frac{p}{2}$, SMK attains the lower bound -1. For $m > 2$, the lower bound is not tight.

SMP

SMP is well-defined if and only if $\sqrt{|V_i| \cdot \left(1 - \frac{|V_i|}{p}\right) \cdot |V_j| \cdot \left(1 - \frac{|V_j|}{p}\right)} \neq 0 \, \forall i < j$, that is, if none of the sets $V_1, \ldots, V_m$ is empty or contains all features.
As shown by Nogueira and Brown (2016),

$$\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot \left(1 - \frac{|V_i|}{p}\right) \cdot |V_j| \cdot \left(1 - \frac{|V_j|}{p}\right)}}$$

is identical to the Pearson correlation $\mathrm{Cor}(z_i, z_j)$ between the two vectors $z_i, z_j \in \{0,1\}^p$ that represent the selected features in a binary way: The k-th component of $z_l, l \in \{i, j\}$ has value 1 if and only if $X_k \in V_l$. The Pearson correlation is known to be bounded by -1 and 1.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\mathrm{SMP} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot \left(1 - \frac{|V_i|}{p}\right) \cdot |V_j| \cdot \left(1 - \frac{|V_j|}{p}\right)}}}_{\leq 1} \leq 1$$

For $V_1 = \ldots = V_m$:

$$\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot \left(1 - \frac{|V_i|}{p}\right) \cdot |V_j| \cdot \left(1 - \frac{|V_j|}{p}\right)}} = \frac{|V_i| - \frac{|V_i|^2}{p}}{|V_i| \left(1 - \frac{|V_i|}{p}\right)} = \frac{|V_i| \left(1 - \frac{|V_i|}{p}\right)}{|V_i| \left(1 - \frac{|V_i|}{p}\right)} = 1$$

$$\Rightarrow \quad \mathrm{SMP} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot \left(1 - \frac{|V_i|}{p}\right) \cdot |V_j| \cdot \left(1 - \frac{|V_j|}{p}\right)}}}_{=1} = 1$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\mathrm{SMP} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot \left(1 - \frac{|V_i|}{p}\right) \cdot |V_j| \cdot \left(1 - \frac{|V_j|}{p}\right)}}}_{\geq -1} \geq -1$$

For $m = 2$ and V_1, V_2 with $|V_1 \cap V_2| = 0$, $|V_1| = \frac{p}{2}$, and $|V_2| = \frac{p}{2}$, SMP attains the lower bound -1. For $m > 2$, the lower bound is not tight.

SME

SME is well-defined if and only if $q \neq 0$, that is, if not more than one of the sets $V_1, \ldots, V_m$ is empty.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\text{SME} = \frac{1}{q \log_2(m)} \sum_{j:X_j \in \bigcup_{i=1}^m V_i} h_j \log_2(h_j) \quad \text{with} \quad q = \sum_{j=1}^{p} h_j = \sum_{i=1}^{m} |V_i|$$

$$\Rightarrow \quad \text{SME} = \frac{1}{q} \sum_{j:X_j \in \bigcup_{i=1}^m V_i} h_j \underbrace{\frac{\log_2(h_j)}{\log_2(m)}}_{\leq 1} \leq \frac{1}{q} \sum_{j:X_j \in \bigcup_{i=1}^m V_i} h_j \leq \frac{1}{q} \underbrace{\sum_{j=1}^{p} h_j}_{=q} = 1$$

For $V_1 = \ldots = V_m$:

$$X_j \in \bigcup_{i=1}^{m} V_i \quad \Rightarrow \quad h_j = m \quad \text{and} \quad X_j \notin \bigcup_{i=1}^{m} V_i \quad \Rightarrow \quad h_j = 0$$

$$\Rightarrow \quad \text{SME} = \frac{1}{q \log_2(m)} \sum_{j:X_j \in \bigcup_{i=1}^m V_i} h_j \log_2(h_j) = \frac{1}{|V_1| \cdot m \cdot \log_2(m)} \sum_{j:X_j \in \bigcup_{i=1}^m V_i} m \log_2(m)$$

$$= \frac{|\bigcup_{i=1}^m V_i|}{|V_1|} = 1$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$X_j \in \bigcup_{i=1}^{m} V_i \quad \Rightarrow \quad h_j \geq 1$$

$$\Rightarrow \quad \text{SME} = \frac{1}{q \log_2(m)} \sum_{j:X_j \in \bigcup_{i=1}^m V_i} \underbrace{h_j}_{\geq 1} \underbrace{\log_2(h_j)}_{\geq 0} \geq 0$$

For $V_i = \{X_i\}$, $i = 1, \ldots, m$:

$$\text{SME} = \frac{1}{m \log_2(m)} \cdot 0 = 0$$

SMD-α

SMD-α is well-defined if and only if $|\bigcup_{i=1}^m V_i| \neq 0$, that is, if not more than one of the sets $V_1, \ldots, V_m$ is empty.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\sum_{j=1}^{p} \frac{h_j}{m} = \frac{1}{m} \sum_{i=1}^{m} |V_i| \leq \max\{|V_1|, \ldots, |V_m|\} \leq \left| \bigcup_{i=1}^{m} V_i \right|$$

$$\Rightarrow \quad \text{SMD-}\alpha = \max \left\{ 0, \underbrace{\frac{1}{|\bigcup_{i=1}^m V_i|} \sum_{j=1}^{p} \frac{h_j}{m}}_{\leq 1} - \underbrace{\frac{\alpha}{p} \cdot \text{median}(|V_1|, \ldots, |V_m|)}_{\geq 0} \right\} \leq 1$$

For $V_1 = \ldots = V_m$:

$$\text{SMD-}\alpha = \max\left\{0,\ \frac{1}{|V_1|}\cdot|V_1| - \frac{\alpha}{p}\cdot|V_1|\right\} = \max\left\{0,\ 1 - \frac{\alpha}{p}\cdot|V_1|\right\} = 1 \quad \Leftrightarrow \quad \alpha = 0$$

For $\alpha > 0$ the upper bound cannot be reached.
Lower bound:
For arbitrary sets $V_1, \ldots, V_m$, the lower bound of

$$\text{SMD-}\alpha = \max\left\{0,\ \frac{1}{|\bigcup_{i=1}^{m} V_i|}\sum_{j=1}^{p}\frac{h_j}{m} - \frac{\alpha}{p}\cdot\text{median}\left(|V_1|,\ldots,|V_m|\right)\right\}$$

is 0 by definition. For small values of α, the lower bound cannot be attained, because

$$\frac{1}{|\bigcup_{i=1}^{m} V_i|}\sum_{j=1}^{p}\frac{h_j}{m} > 0.$$

If α is large enough, the lower bound can be attained, for example for $|V_1| = \ldots = \left|V_{\lceil\frac{m+1}{2}\rceil}\right| = p$ and $\left|V_{\lceil\frac{m+1}{2}\rceil+1}\right| = \ldots = |V_m| = 0$:

$$\text{SMD-}\alpha = \max\left\{0,\ \frac{1}{p}\cdot\frac{p\left\lceil\frac{m+1}{2}\right\rceil}{m} - \frac{\alpha}{p}\cdot p\right\} = 0 \quad \Leftrightarrow \quad \alpha \geq \frac{\left\lceil\frac{m+1}{2}\right\rceil}{m}$$

SMS

SMS is well-defined if and only if $c_{\max} \neq c_{\min}$. Finding the values for p, q, and m for which this is true, means solving equations with integer constraints. Computations for all possible combinations of $p \in \{1,\ldots,10\,000\}$ and $m \in \{2,\ldots,100\}$ show that SMS is well-defined if and only if $p > 1$, $q > 1$, and $q < pm - 1$. That is, the sum of all selected features $q = \sum_{i=1}^{m}|V_i|$ has to be larger than 1 and smaller than $pm - 1$.
Somol and Novovičová (2008) prove for

$$\text{SMS} = \frac{\left(\sum_{j=1}^{p}\frac{h_j}{q}\frac{h_j-1}{m-1}\right) - c_{\min}}{c_{\max} - c_{\min}} \quad \text{with}$$

$$c_{\min} = \frac{q^2 - p(q - q \bmod p) - (q \bmod p)^2}{pq(m-1)} \quad \text{and}$$

$$c_{\max} = \frac{(q \bmod m)^2 + q(m-1) - (q \bmod m)\,m}{q(m-1)}$$

that

$$c_{\min} \leq \sum_{j=1}^{p}\frac{h_j}{q}\frac{h_j-1}{m-1} \leq c_{\max}.$$

Therefore, $0 \leq \text{SMS} \leq 1$ holds for arbitrary sets $V_1, \ldots, V_m$.

Both bounds can be attained: For $V_1 = \ldots = V_m$:

$$\begin{aligned} & h_j = m \Leftrightarrow X_j \in V_1 \; \forall j = 1, \ldots, p, \quad h_j = 0 \Leftrightarrow X_j \notin V_1 \; \forall j = 1, \ldots, p, \quad \text{and} \quad q = m\,|V_1| \\ \Rightarrow \quad & \sum_{j=1}^{p} \frac{h_j}{q} \frac{h_j - 1}{m-1} = |V_1| \cdot \frac{m}{q} \cdot \frac{m-1}{m-1} = 1 \quad \text{and} \quad c_{\max} = \frac{0^2 + q(m-1) - 0}{q(m-1)} = 1 \\ \Rightarrow \quad & \text{SMS} = \frac{\sum_{j=1}^{p} \frac{h_j}{q} \frac{h_j - 1}{m-1} - c_{\min}}{c_{\max} - c_{\min}} = 1 \end{aligned}$$

For $V_1, \ldots, V_m$ pairwise disjoint with $|V_i| = \frac{p}{m} \; \forall i = 1, \ldots, m$:

$$\begin{aligned} & h_j = 1 \; \forall j = 1, \ldots, p \quad \text{and} \quad q = p \\ \Rightarrow \quad & \sum_{j=1}^{p} \frac{h_j}{q} \frac{h_j - 1}{m-1} = 0 \quad \text{and} \quad c_{\min} = \frac{p^2 - p(p-0) - 0^2}{p^2(m-1)} = 0 \\ \Rightarrow \quad & \text{SMS} = \frac{\sum_{j=1}^{p} \frac{h_j}{q} \frac{h_j - 1}{m-1} - c_{\min}}{c_{\max} - c_{\min}} = 0 \end{aligned}$$

SMN

SMN is well-defined if and only if $\frac{q}{mp}\left(1 - \frac{q}{mp}\right) \neq 0$, that is, if at least one of the sets $V_1, \ldots, V_m$ is not empty or contains less than p features.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\begin{aligned} & s_j^2 = \frac{m}{m-1} \underbrace{\frac{h_j}{m}}_{\geq 0} \underbrace{\left(1 - \frac{h_j}{m}\right)}_{\geq 0} \geq 0 \quad \text{and} \quad q = \sum_{i=1}^{m} |V_i| \leq mp \\ \Rightarrow \quad & \text{SMN} = 1 - \frac{\overbrace{\frac{1}{p} \sum_{j=1}^{p} s_j^2}^{\geq 0}}{\underbrace{\frac{q}{mp}\left(1 - \frac{q}{mp}\right)}_{\geq 0}} \leq 1 \end{aligned}$$

For $V_1 = \ldots = V_m$:

$$\begin{aligned} & h_j = m \Leftrightarrow X_j \in V_1 \; \forall j = 1, \ldots, p \quad \text{and} \quad h_j = 0 \Leftrightarrow X_j \notin V_1 \; \forall j = 1, \ldots, p \\ \Rightarrow \quad & s_j^2 = 0 \; \forall j = 1, \ldots, p \quad \Rightarrow \quad \text{SMN} = 1 \end{aligned}$$

Lower bound:
Nogueira (2018) shows that $\text{SMN} \geq -\frac{1}{m-1}$ for arbitrary sets $V_1, \ldots, V_m$. This bound is

attained for $V_1, \ldots, V_m$ pairwise disjoint with $|V_i| = \frac{p}{m}$, $i = 1, \ldots, m$:

$$h_j = 1 \; \forall j = 1, \ldots, p \quad \Rightarrow \quad s_j^2 = \frac{m}{m-1} \frac{h_j}{m} \left(1 - \frac{h_j}{m}\right) = \frac{m}{m-1} \cdot \frac{1}{m} \left(1 - \frac{1}{m}\right) = \frac{1}{m}$$

$$q = p \quad \Rightarrow \quad \frac{q}{mp} \left(1 - \frac{q}{mp}\right) = \frac{1}{m} \left(1 - \frac{1}{m}\right) = \frac{m-1}{m^2}$$

$$\Rightarrow \quad \text{SMN} = 1 - \frac{\frac{1}{m}}{\frac{m-1}{m^2}} = -\frac{1}{m-1}$$

SMZ

SMZ is well-defined if and only if $|V_i \cup V_j| \neq 0 \, \forall i < j$, that is, if not more than one of the sets $V_1, \ldots, V_m$ is empty.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$C(V_i, V_j) = \sum_{x \in V_i} \frac{1}{|V_j|} \sum_{y \in V_j \setminus V_i} \underbrace{|\text{Cor}(x,y)| \; \mathbb{I}_{[\theta,\infty)}(|\text{Cor}(x,y)|)}_{\leq 1} \leq \frac{|V_i|}{|V_j|} \cdot |V_j \setminus V_i| \quad \text{and}$$

$$C(V_j, V_i) \leq \frac{|V_j|}{|V_i|} \cdot |V_i \setminus V_j|$$

$$\Rightarrow \quad |V_i \cap V_j| + C(V_i, V_j) + C(V_j, V_i) \leq |V_i \cap V_j| + \frac{|V_i|}{|V_j|} \cdot |V_j \setminus V_i| + \frac{|V_j|}{|V_i|} \cdot |V_i \setminus V_j|$$

$$= |V_i \cap V_j| + \frac{|V_i|}{|V_j|} \cdot (|V_j| - |V_i \cap V_j|) + \frac{|V_j|}{|V_i|} \cdot (|V_i| - |V_i \cap V_j|)$$

$$= |V_i \cap V_j| \underbrace{\left(1 - \frac{|V_i|}{|V_j|} - \frac{|V_j|}{|V_i|}\right)}_{\leq -1 (*)} + |V_i| + |V_j| \leq |V_i \cup V_j|$$

$$\Rightarrow \quad \text{SMZ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| + C(V_i, V_j) + C(V_j, V_i)}{|V_i \cup V_j|}}_{\leq 1} \leq 1$$

$$(*): \quad 1 - \frac{|V_i|}{|V_j|} - \frac{|V_j|}{|V_i|} \leq -1$$

$$\Leftrightarrow \quad \frac{|V_i|}{|V_j|} + \frac{|V_j|}{|V_i|} \geq 2$$

$$\Leftrightarrow \quad \frac{|V_i|^2 + |V_j|^2}{|V_i| \cdot |V_j|} \geq 2$$

$$\Leftrightarrow \quad |V_i|^2 + |V_j|^2 \geq 2 \cdot |V_i| \cdot |V_j|$$

$$\Leftrightarrow \quad |V_i|^2 + |V_j|^2 - 2 \cdot |V_i| \cdot |V_j| = (|V_i| - |V_j|)^2 \geq 0$$

For $V_1 = \ldots = V_m$:

$$|V_i \cap V_j| = |V_i \cup V_j|, \quad C(V_i, V_j) = 0, \quad \text{and} \quad C(V_j, V_i) = 0 \quad \forall i < j$$
$$\Rightarrow \quad \text{SMZ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| + C(V_i, V_j) + C(V_j, V_i)}{|V_i \cup V_j|}}_{=1} = 1$$

Lower bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$\text{SMZ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{\overbrace{|V_i \cap V_j|}^{\geq 0} + \overbrace{C(V_i, V_j)}^{\geq 0} + \overbrace{C(V_j, V_i)}^{\geq 0}}{\underbrace{|V_i \cup V_j|}_{\geq 0}} \geq 0$$

For $V_1, \ldots, V_m$ pairwise disjoint and $|\text{Cor}(x, y)| < \theta \ \forall x, y \in \{X_1, \ldots, X_p\}$:

$$\text{SMZ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{\overbrace{|V_i \cap V_j|}^{=0} + \overbrace{C(V_i, V_j)}^{=0} + \overbrace{C(V_j, V_i)}^{=0}}{\underbrace{|V_i \cup V_j|}_{\geq 0}} = 0$$

SMES

SMES is well-defined if and only if $p > 1$ and $\text{trace}(C\Sigma) \neq 0$ which depends on the data specific similarity structure. SMES is undefined if all of the sets $V_1, \ldots, V_m$ are empty or contain all p features.
Upper bound:
No upper bound is known. It seems that SMES can attain arbitrarily large values, depending on the number of features in the data set, p, and the similarity matrix C. Consider the scenario with $m = 2$, $V_1 = \left\{X_1, \ldots, X_{\lceil \frac{p}{2} \rceil}\right\}$, $V_2 = \left\{X_{\lceil \frac{p}{2} \rceil + 1}, \ldots, X_p\right\}$, and

$$C_{ij} = \begin{cases} 1, & i = j, \\ \tau, & (X_i \in V_1 \text{ and } X_j \in V_2) \text{ or } (X_i \in V_2 \text{ and } X_j \in V_1), i \neq j, \\ 0, & (X_i \in V_1 \text{ and } X_j \in V_1) \text{ or } (X_i \in V_2 \text{ and } X_j \in V_2), i \neq j. \end{cases}$$

Such a similarity matrix is plausible if only similarity values that are larger than or equal to a threshold are kept and the other ones are set to 0. The top plot in Figure B.1 shows the values that SMES attains in this scenario for $p \in \{2, \ldots, 100\}$ and $\tau \in \{0.90, 0.91, \ldots, 1\}$. It can be observed that the value of SMES increases with increasing p and that this increase is the stronger, the larger the value of τ.
Lower Bound:
No lower bound is known. It seems that SMES can attain arbitrarily small values, depending on p and C. Consider the scenario with $m = 2$, $V_1 = \{X_1\}, V_2 = \{X_2\}$, and

$$C_{ij} = \begin{cases} 1, & i = j, \\ \tau, & i \notin \{1, 2\} \text{ and } j \notin \{1, 2\}, i \neq j, \\ 0, & i \in \{1, 2\} \text{ or } j \in \{1, 2\}, i \neq j. \end{cases}$$

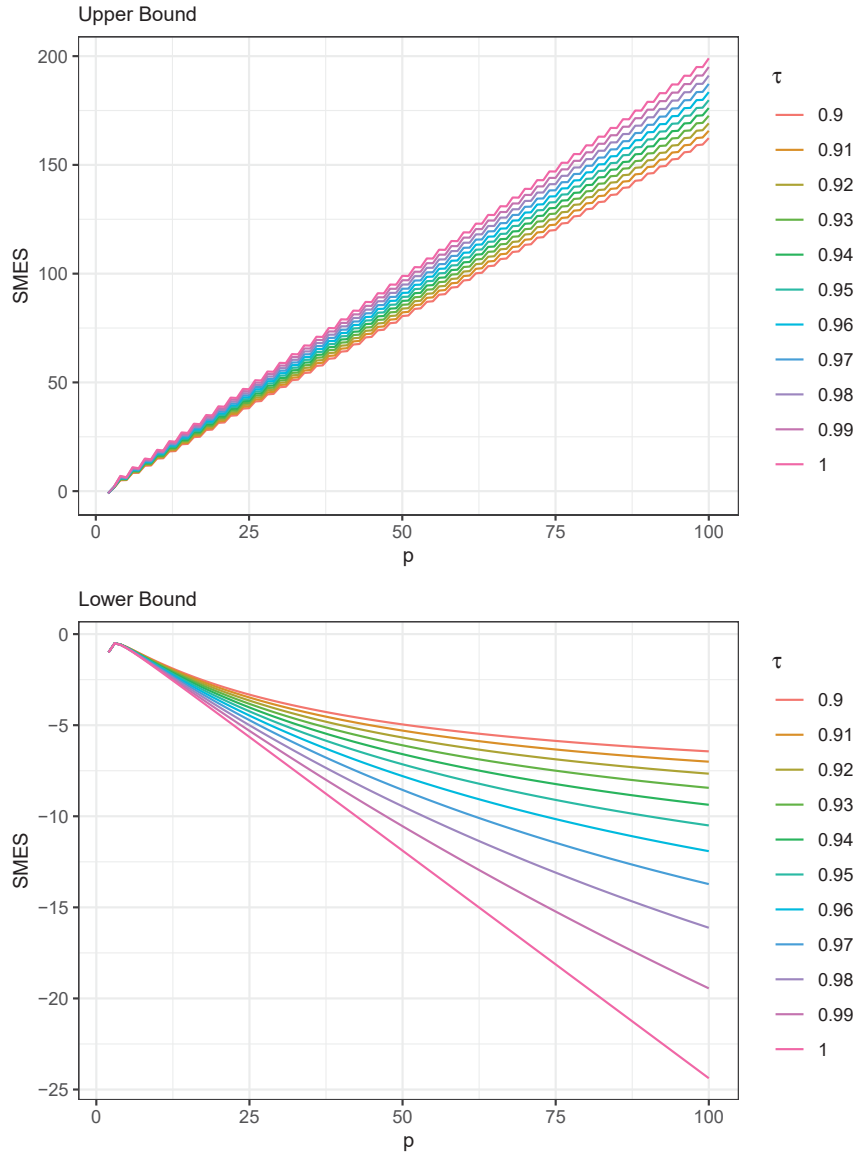


Figure B.1: Value of SMES, depending on the number of features in the data set, p, and the similarity matrix C described above with parameter τ.

The two selected features are not similar to each other or to any of the other features. The remaining features in the data set are similar to each other. The bottom plot in Figure B.1 displays the values that SMES attains in this scenario for $p \in \{2, \ldots, 100\}$ and $\tau \in \{0.90, 0.91, \ldots, 1\}$. The value of SMES decreases with increasing p and this decrease is the stronger, the larger the value of τ.

SMY

SMY is well-defined if and only if not more than one of the sets $V_1, \ldots, V_m$ is empty and

$$E\left[|V_i \cap V_j| + \frac{A(V_i, V_j) + A(V_j, V_i)}{2}\right] < \frac{|V_i| + |V_j|}{2} \; \forall i < j$$

which depends on the data specific similarity structure.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:

$$
\begin{aligned}
& A(V_i, V_j) = |\{x \in (V_i \setminus V_j) : \exists y \in (V_j \setminus V_i) \text{ with similarity}(x, y) \geq \theta\}| \leq |V_i \setminus V_j| \ \forall i, j \\
\Rightarrow \quad & |V_i \cap V_j| + \frac{A(V_i, V_j) + A(V_j, V_i)}{2} \leq |V_i \cap V_j| + \frac{|V_i \setminus V_j| + |V_j \setminus V_i|}{2} \\
& = \frac{|V_i| + |V_j|}{2} \ \forall i < j \\
\Rightarrow \quad & \text{Score}_{\text{SMY}}(V_i, V_j) := \frac{|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2} - E\left[|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}\right]}{\frac{|V_i|+|V_j|}{2} - E\left[|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}\right]} \leq 1 \\
\Rightarrow \quad & \text{SMY} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \text{Score}_{\text{SMY}}(V_i, V_j) \leq 1
\end{aligned}
$$

For $V_1 = \ldots = V_m$:

- $|V_i \cap V_j| = \frac{|V_i|+|V_j|}{2} \ \forall i < j$ (see SMD)
- $A(V_i, V_j) = \left| \{x \in \underbrace{(V_i \setminus V_j)}_{=\emptyset} : \exists y \in (V_j \setminus V_i) \text{ with similarity}(x, y) \geq \theta\} \right| = 0 \ \forall i, j$

$$
\begin{aligned}
\Rightarrow \quad & \text{Score}_{\text{SMY}}(V_i, V_j) = \frac{|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2} - E\left[|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}\right]}{\frac{|V_i|+|V_j|}{2} - E\left[|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}\right]} = 1 \\
\Rightarrow \quad & \text{SMY} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \text{Score}_{\text{SMY}}(V_i, V_j) = 1
\end{aligned}
$$

SMA

SMA is well-defined if and only if none of the sets $V_1, \ldots, V_m$ is empty and

$$E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right] < \sqrt{|V_i| \cdot |V_j|} \ \forall i < j$$

which depends on the data specific similarity structure.
Upper bound:
For arbitrary sets $V_1, \ldots, V_m$:
First, it is shown that $\text{Adj}_{\text{Count}}$ is the largest of the considered adjustments:

1. $\text{Adj}_{\text{Greedy}} \leq \text{Adj}_{\text{MBM}}$
2. $\text{Adj}_{\text{MBM}} \leq \text{Adj}_{\text{Count}}$
3. $\text{Adj}_{\text{Mean}} \leq \text{Adj}_{\text{Count}}$

1. The tuples $[x, y, s]$ in Algorithm 3.1 represent edges in the graph described for SMA-MBM. L_B is constructed in such a way that all its tuples do not share any vertices x or y. Therefore, L_B is a matching. The size of any matching of a graph cannot exceed the size of the maximum matching of this graph by definition of the maximum matching. Therefore, $\text{Adj}_{\text{Greedy}}(V_i, V_j) \leq \text{Adj}_{\text{MBM}}(V_i, V_j)$ must hold for arbitrary sets V_i and V_j.

2. Consider arbitrary sets V_i and V_j. By definition of Adj_{MBM}, there must be $\text{Adj}_{\text{MBM}}(V_i, V_j)$ edges in the graph constructed for SMA-MBM that belong to the maximum matching. This means that there must be $\text{Adj}_{\text{MBM}}(V_i, V_j)$ vertices that represent features of $V_i \setminus V_j$ and $\text{Adj}_{\text{MBM}}(V_i, V_j)$ vertices that represent features of $V_j \setminus V_i$ that are connected by the edges in the matching. Because of the way the graph is constructed, there must be at least $\text{Adj}_{\text{MBM}}(V_i, V_j)$ features in $V_i \setminus V_j$ with a feature in $V_j \setminus V_i$ with similarity value of at least θ, that is,

$$\begin{aligned} A(V_i, V_j) &= |\{x \in (V_i \setminus V_j) : \exists y \in (V_j \setminus V_i) \text{ with similarity}(x, y) \geq \theta\}| \\ &\geq \text{Adj}_{\text{MBM}}(V_i, V_j). \end{aligned}$$

   Analogously, $A(V_j, V_i) \geq \text{Adj}_{\text{MBM}}(V_i, V_j)$ must hold. Therefore,

$$\text{Adj}_{\text{Count}}(V_i, V_j) = \min\{A(V_i, V_j), A(V_j, V_i)\} \geq \text{Adj}_{\text{MBM}}(V_i, V_j).$$

3. Consider arbitrary sets V_i and V_j.

$$\frac{1}{\left|G_x^{ij}\right|} \sum_{y \in G_x^{ij}} \text{similarity}(x, y) \leq 1$$

   holds, because $\text{similarity}(x, y) \in [0, 1]$. Therefore,

$$M(V_i, V_j) = \sum_{x \in V_i \setminus V_j : \left|G_x^{ij}\right| > 0} \frac{1}{\left|G_x^{ij}\right|} \sum_{y \in G_x^{ij}} \text{similarity}(x, y) \leq \left|\left\{x \in V_i \setminus V_j : \left|G_x^{ij}\right| > 0\right\}\right|$$

   with $|G_x^{ij}| = |\{y \in V_j \setminus V_i : \text{similarity}(x, y) \geq \theta\}|$, that is,

$$M(V_i, V_j) \leq |\{x \in (V_i \setminus V_j) : \exists y \in (V_j \setminus V_i) \text{ with similarity}(x, y) \geq \theta\}| = A(V_i, V_j).$$

   So, $M(V_i, V_j) \leq A(V_i, V_j)$ and analogously $M(V_j, V_i) \leq A(V_j, V_i)$ and therefore

$$\begin{aligned} \text{Adj}_{\text{Mean}}(V_i, V_j) &= \min\{M(V_i, V_j), M(V_j, V_i)\} \\ &\leq \min\{A(V_i, V_j), A(V_j, V_i)\} = \text{Adj}_{\text{Count}}(V_i, V_j). \end{aligned}$$

W.l.o.g. assume that $|V_i| \leq |V_j| \ \ \forall i < j$.
SMA-Count:

$$\begin{aligned} & A(V_i, V_j) \leq |V_i \setminus V_j| \quad \text{and} \quad A(V_j, V_i) \leq |V_j \setminus V_i| \quad \text{(see SMY)} \\ \Rightarrow \quad & \text{Adj}_{\text{Count}}(V_i, V_j) = \min\{A(V_i, V_j), A(V_j, V_i)\} \leq \min\{|V_i \setminus V_j|, |V_j \setminus V_i|\} = |V_i \setminus V_j| \, \forall i < j \\ \Rightarrow \quad & |V_i \cap V_j| + \text{Adj}_{\text{Count}}(V_i, V_j) \leq |V_i \cap V_j| + |V_i \setminus V_j| = |V_i| \leq \sqrt{|V_i| \cdot |V_j|} \ \forall i < j \\ \Rightarrow \quad & \text{Score}_{\text{Count}}(V_i, V_j) := \frac{|V_i \cap V_j| + \text{Adj}_{\text{Count}}(V_i, V_j) - E\left[|V_i \cap V_j| + \text{Adj}_{\text{Count}}(V_i, V_j)\right]}{\sqrt{|V_i| \cdot |V_j|} - E\left[|V_i \cap V_j| + \text{Adj}_{\text{Count}}(V_i, V_j)\right]} \leq 1 \\ \Rightarrow \quad & \text{SMA-Count} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \text{Score}_{\text{Count}}(V_i, V_j) \leq 1 \end{aligned}$$

SMA-MBM:

$$\begin{aligned}
& \mathrm{Adj}_{\mathrm{MBM}}(V_i, V_j) \leq \mathrm{Adj}_{\mathrm{Count}}(V_i, V_j)\ \forall i < j \\
\Rightarrow\quad & |V_i \cap V_j| + \mathrm{Adj}_{\mathrm{MBM}}(V_i, V_j) \leq |V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Count}}(V_i, V_j) \leq \sqrt{|V_i| \cdot |V_j|}\ \forall i < j \\
\Rightarrow\quad & \mathrm{Score}_{\mathrm{MBM}}(V_i, V_j) := \frac{|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{MBM}}(V_i, V_j) - E\left[|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{MBM}}(V_i, V_j)\right]}{\sqrt{|V_i| \cdot |V_j|} - E\left[|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{MBM}}(V_i, V_j)\right]} \leq 1 \\
\Rightarrow\quad & \text{SMA-MBM} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \mathrm{Score}_{\mathrm{MBM}}(V_i, V_j) \leq 1
\end{aligned}$$

SMA-Greedy:

$$\begin{aligned}
& \mathrm{Adj}_{\mathrm{Greedy}}(V_i, V_j) \leq \mathrm{Adj}_{\mathrm{Count}}(V_i, V_j)\ \forall i < j \\
\Rightarrow\quad & |V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Greedy}}(V_i, V_j) \leq |V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Count}}(V_i, V_j) \leq \sqrt{|V_i| \cdot |V_j|}\ \forall i < j \\
\Rightarrow\quad & \mathrm{Score}_{\mathrm{Greedy}}(V_i, V_j) := \frac{|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Greedy}}(V_i, V_j) - E\left[|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Greedy}}(V_i, V_j)\right]}{\sqrt{|V_i| \cdot |V_j|} - E\left[|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Greedy}}(V_i, V_j)\right]} \leq 1 \\
\Rightarrow\quad & \text{SMA-Greedy} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \mathrm{Score}_{\mathrm{Greedy}}(V_i, V_j) \leq 1
\end{aligned}$$

SMA-Mean:

$$\begin{aligned}
& \mathrm{Adj}_{\mathrm{Mean}}(V_i, V_j) \leq \mathrm{Adj}_{\mathrm{Count}}(V_i, V_j)\ \forall i < j \\
\Rightarrow\quad & |V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Mean}}(V_i, V_j) \leq |V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Count}}(V_i, V_j) \leq \sqrt{|V_i| \cdot |V_j|}\ \forall i < j \\
\Rightarrow\quad & \mathrm{Score}_{\mathrm{Mean}}(V_i, V_j) := \frac{|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Mean}}(V_i, V_j) - E\left[|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Mean}}(V_i, V_j)\right]}{\sqrt{|V_i| \cdot |V_j|} - E\left[|V_i \cap V_j| + \mathrm{Adj}_{\mathrm{Mean}}(V_i, V_j)\right]} \leq 1 \\
\Rightarrow\quad & \text{SMA-Mean} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \mathrm{Score}_{\mathrm{Mean}}(V_i, V_j) \leq 1
\end{aligned}$$

For $V_1 = \ldots = V_m$:

- $V_i \setminus V_j = \emptyset$ and $\mathrm{V}_j \setminus V_i = \emptyset$
  $\Rightarrow \mathrm{Adj}_{\mathrm{MBM}}(V_i, V_j) = \mathrm{Adj}_{\mathrm{Greedy}}(V_i, V_j) = \mathrm{Adj}_{\mathrm{Count}}(V_i, V_j) = \mathrm{Adj}_{\mathrm{Mean}}(V_i, V_j) = 0\ \forall i < j$
- $|V_i \cap V_j| = \sqrt{|V_i| \cdot |V_j|}\ \forall i < j$ (see SMO)
  $\Rightarrow$ SMA-MBM = SMA-Greedy = SMA-Count = SMA-Mean = 1

B.2 Monotonicity

SMD-α

Consider the situation with $m = 5$ and

(a) $V_1 = \{X_1, X_2\}$ and $V_2 = V_3 = V_4 = V_5 = \{X_2\}$, that is, $h_1 = 1$, $h_2 = 5$, and $h_j = 0\ \forall j \in \{3, \ldots, p\}$,

(b) $V_1 = V_2 = \{X_1, X_2\}$ and $V_3 = V_4 = V_5 = \{X_2\}$, that is, $h_1 = 2$, $h_2 = 5$, and $h_j = 0\ \forall j \in \{3, \ldots, p\}$.

In scenario (a):

$$\begin{aligned}\text{SMD-}\alpha &= \max\left\{0,\ \frac{1}{|\{X_1, X_2\}|}\left(\frac{1}{5}+\frac{5}{5}\right) - \frac{\alpha}{p}\cdot \text{median}\left(|V_1|, \ldots, |V_5|\right)\right\} \\ &= \max\left\{0, \frac{1}{2}\cdot\left(\frac{1}{5}+\frac{5}{5}\right) - \frac{\alpha}{p}\cdot 1\right\} = \max\left\{0, 0.6 - \frac{\alpha}{p}\right\}\end{aligned}$$

In scenario (b):

$$\begin{aligned}\text{SMD-}\alpha &= \max\left\{0,\ \frac{1}{|\{X_1, X_2\}|}\left(\frac{2}{5}+\frac{5}{5}\right) - \frac{\alpha}{p}\cdot \text{median}\left(|V_1|, \ldots, |V_5|\right)\right\} \\ &= \max\left\{0, \frac{1}{2}\cdot\left(\frac{2}{5}+\frac{5}{5}\right) - \frac{\alpha}{p}\cdot 1\right\} = \max\left\{0, 0.7 - \frac{\alpha}{p}\right\}\end{aligned}$$

So, the value of SMD-α is higher in scenario (b) than in scenario (a) or it is equal for $\alpha \geq 0.7p$. The sample variance s_j^2 of feature X_j is defined by Nogueira (2018) as

$$s_j^2 = \frac{m}{m-1}\frac{h_j}{m}\left(1 - \frac{h_j}{m}\right).$$

In scenario (a):

$$s_1 = \frac{5}{4}\cdot\frac{1}{5}\cdot\left(1-\frac{1}{5}\right) = 0.2$$

In scenario (b):

$$s_1 = \frac{5}{4}\cdot\frac{2}{5}\cdot\left(1-\frac{2}{5}\right) = 0.3$$

As the sample variance of feature X_1 is higher in scenario (b) than in scenario (a), SMD-α cannot be a strictly decreasing function of the sample variance of the selection for each feature.

SMZ

Consider the situation with $p = 9$, $m = 2$, and

$$\text{Cor}(X_i, X_j) = \begin{cases} 1, & i = j, \\ 1, & i \in \{6,7,8,9\} \text{ and } j \in \{6,7,8,9\}, \\ 0, & \text{otherwise.}\end{cases}$$

The features X_6 to X_9 are similar to each other, while the features X_1 to X_5 are neither similar to each other nor to the other features. Now consider the scenarios:

(a) $V_1 = \{X_1, X_2, X_3\}$ and $V_2 = \{X_1, X_4, X_5\}$,

(b) $V_1 = \{X_6, X_7, X_8\}$ and $V_2 = \{X_1, X_9\}$.

These sets are chosen such that $|V_1 \cap V_2|$ varies while $|V_1 \cup V_2|$ remains constant.

$$\text{SMZ} = \frac{2}{m(m-1)}\sum_{i=1}^{m-1}\sum_{j=i+1}^{m}\frac{|V_i \cap V_j| + C(V_i, V_j) + C(V_j, V_i)}{|V_i \cup V_j|} \quad \text{with}$$

$$C(V_i, V_j) = \sum_{x \in V_i}\frac{1}{|V_j|}\sum_{y \in V_j \setminus V_i}|\text{Cor}(x,y)|\ \mathbb{I}_{[\theta,\infty)}\left(|\text{Cor}(x,y)|\right).$$

In scenario (a):

$$\begin{aligned} |V_1 \cap V_2| &= 1 \\ C(V_1, V_2) &= 0 \\ C(V_2, V_1) &= 0 \\ \Rightarrow \text{SMZ} &= \frac{1+0+0}{5} = 0.2 \end{aligned}$$

In scenario (b):

$$\begin{aligned} |V_1 \cap V_2| &= 0 \\ C(V_1, V_2) &= 3\left(\frac{1}{2}(0+1)\right) = 1.5 \\ C(V_2, V_1) &= \frac{1}{3}(1+1+1) + 0 = 1 \\ \Rightarrow \text{SMZ} &= \frac{0+1.5+1}{5} = 0.5 \end{aligned}$$

So, $|V_1 \cap V_2|$ is larger in scenario (a), but the value of SMZ is larger in scenario (b). Therefore, stability measure SMZ cannot be a strictly increasing function of all cardinalites of pairwise intersections.

SMES

Consider the situation with $p = 7$, $m = 2$, and similarity matrix

$$C_{ij} = \begin{cases} 1, & i = j, \\ 1, & i \in \{4,5,6,7\} \text{ and } j \in \{4,5,6,7\}, \\ 0, & \text{otherwise.} \end{cases}$$

The features X_4 to X_7 are similar to each other, while the features X_1 to X_3 are neither similar to each other nor to the other features. Now consider the scenarios:

(a) $V_1 = \{X_1, X_2\}$ and $V_2 = \{X_1, X_3\}$,

(b) $V_1 = \{X_4, X_5\}$ and $V_2 = \{X_6, X_7\}$.

These sets are chosen such that $|V_1 \cap V_2|$ varies while $|V_1|$ and $|V_2|$ remain constant.

$$\text{SMES} = 1 - \frac{\text{trace}(CS)}{\text{trace}(C\Sigma)}$$

with

$$S_{ij} = \frac{m}{m-1}\left(\frac{h_{ij}}{m} - \frac{h_i}{m}\frac{h_j}{m}\right),$$

$$\Sigma_{ij} = \begin{cases} \frac{q}{mp}\left(1 - \frac{q}{mp}\right), & i = j, \\ \frac{\frac{1}{m}\sum_{i=1}^{m}|V_i|^2 - \frac{q}{m}}{p^2 - p} - \frac{q^2}{m^2 p^2}, & i \neq j. \end{cases}$$

In scenario (a):

$$S_{ij} = \begin{cases} 0.5, & i = j = 2 \text{ or } i = j = 3, \\ -0.5, & (i = 2 \text{ and } j = 3) \text{ or } (i = 3 \text{ and } j = 2), \\ 0, & \text{otherwise.} \end{cases}$$

$$\text{trace}(CS) = 0 + 0.5 + 0.5 + 0 + 0 + 0 + 0 = 1$$

$$\Sigma_{ij} = \begin{cases} \frac{10}{49}, & i = j, \\ \frac{-5}{147}, & i \neq j. \end{cases}$$

$$\text{trace}(C\Sigma) = \frac{1}{147}(30 + 30 + 30 + 15 + 15 + 15 + 15) = \frac{150}{147}$$

$$\Rightarrow \text{SMES} = 1 - \frac{1}{\frac{150}{147}} = \frac{3}{150} = 0.02$$

In scenario (b):

$$S_{ij} = \begin{cases} 0.5, & i, j \in \{4, 5\} \text{ or } i, j \in \{6, 7\}, \\ -0.5, & (i \in \{4, 5\} \text{ and } j \in \{6, 7\}) \text{ or } (i \in \{6, 7\} \text{ and } j \in \{4, 5\}), \\ 0, & \text{otherwise.} \end{cases}$$

$$\text{trace}(CS) = 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 0$$

$$\Sigma_{ij} = \begin{cases} \frac{10}{49}, & i = j, \\ \frac{-5}{147}, & i \neq j. \end{cases}$$

$$\text{trace}(C\Sigma) = \frac{1}{147}(30 + 30 + 30 + 15 + 15 + 15 + 15) = \frac{150}{147}$$

$$\Rightarrow \text{SMES} = 1 - \frac{0}{\frac{150}{147}} = 1$$

In scenario (a), $|V_1 \cap V_2|$ is larger than in scenario (b), but the value of SMES is larger in scenario (b) than in scenario (a). Therefore, stability measure SMES cannot be a strictly increasing function of all cardinalites of pairwise intersections.

SMY

Consider the same scenarios as for SMES on page 164.

$$\text{SMY} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2} - E\left[|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}\right]}{\frac{|V_i|+|V_j|}{2} - E\left[|V_i \cap V_j| + \frac{A(V_i,V_j)+A(V_j,V_i)}{2}\right]}$$

$$\text{with} \quad A(V_i, V_j) = |\{x \in (V_i \setminus V_j) : \exists y \in (V_j \setminus V_i) \text{ with similarity}(x, y) \geq \theta\}|$$

In scenario (a):

$$\begin{aligned} |V_1 \cap V_2| &= 1 \\ A(V_1, V_2) &= 0 \\ A(V_2, V_1) &= 0 \\ \frac{|V_1| + |V_2|}{2} &= 2 \\ E\left[|V_1 \cap V_2| + \frac{A(V_1, V_2) + A(V_2, V_1)}{2}\right] &= \frac{8}{7} \\ \Rightarrow \text{SMY} = \frac{1 + 0 - \frac{8}{7}}{2 - \frac{8}{7}} &= -\frac{1}{6} \end{aligned}$$

In scenario (b):

$$\begin{aligned} |V_1 \cap V_2| &= 0 \\ A(V_1, V_2) &= 2 \\ A(V_2, V_1) &= 2 \\ \frac{|V_1| + |V_2|}{2} &= 2 \\ E\left[|V_1 \cap V_2| + \frac{A(V_1, V_2) + A(V_2, V_1)}{2}\right] &= \frac{8}{7} \\ \Rightarrow \text{SMY} = \frac{0 + \frac{2+2}{2} - \frac{8}{7}}{2 - \frac{8}{7}} &= 1 \end{aligned}$$

Note that for assessing the expected value, the score "$\left|\widetilde{V}_1 \cap \widetilde{V}_2\right| + \frac{A(\widetilde{V}_1, \widetilde{V}_2) + A(\widetilde{V}_2, \widetilde{V}_1)}{2}$" is calculated for all 441 combinations of two sets with $\left|\widetilde{V}_1\right| = \left|\widetilde{V}_2\right| = 2$ and then the average value is assessed. In scenario (a), $|V_1 \cap V_2|$ is larger than in scenario (b), but the value of SMY is larger in scenario (b) than in scenario (a). Therefore, stability measure SMY cannot be a strictly increasing function of all cardinalites of pairwise intersections.

SMA

Consider the same scenarios as for SMES on page 164. In this situation, the four variants SMA-MBM, SMA-Greedy, SMA-Count and SMA-Mean are identical, because the respective adjustment functions are identical. As shown in Appendix B.1 on page 160 f., $\text{Adj}_{\text{Count}}$ and Adj_{Mean} are identical if and only if all non-zero similarity values are equal to 1 which is fulfilled in the considered situation. Also, $\text{Adj}_{\text{Greedy}} \leq \text{Adj}_{\text{MBM}} \leq \text{Adj}_{\text{Count}}$ and in the considered situation, $\text{Adj}_{\text{Greedy}} = \text{Adj}_{\text{MBM}} = \text{Adj}_{\text{Count}}$. In the considered situation, there are two sets with cardinality $\left|\widetilde{V}_1\right| = \left|\widetilde{V}_2\right| = 2$.

- If $\left|\widetilde{V}_1 \cap \widetilde{V}_2\right| = 2$, then $\text{Adj}_{\text{Greedy}}$ and $\text{Adj}_{\text{Count}}$ take the value 0 because both difference sets $\widetilde{V}_1 \setminus \widetilde{V}_2$ and $\widetilde{V}_2 \setminus \widetilde{V}_1$ are empty.
- If $\left|\widetilde{V}_1 \cap \widetilde{V}_2\right| = 1$, then there are two cases. W.l.o.g. let $\widetilde{V}_1 \setminus \widetilde{V}_2 = \{X_A\}$ and $\widetilde{V}_2 \setminus \widetilde{V}_1 = \{X_B\}$.
  - If $\text{similarity}(X_A, X_B) = 1$, then $\text{Adj}_{\text{Greedy}}$ and $\text{Adj}_{\text{Count}}$ take the value 1.
  - Otherwise they both take the value 0.

- If $\left|\widetilde{V}_1 \cap \widetilde{V}_2\right| = 0$, then there are six cases which will be presented by their partial similarity matrices M. W.l.o.g. assume that the first two features in M belong to $\widetilde{V}_1 \setminus \widetilde{V}_2$ and the last two to $\widetilde{V}_2 \setminus \widetilde{V}_1$. Similarity values of features of the same set are denoted with $\star$ because they are irrelevant for the adjustment functions.
  - If $M = \begin{pmatrix} 1 & \star & 0 & 0 \\ \star & 1 & 0 & 0 \\ 0 & 0 & 1 & \star \\ 0 & 0 & \star & 1 \end{pmatrix}$, then $\mathrm{Adj}_{\mathrm{Greedy}}$ and $\mathrm{Adj}_{\mathrm{Count}}$ take the value 0.
  - If $M = \begin{pmatrix} 1 & \star & 1 & 0 \\ \star & 1 & 0 & 0 \\ 1 & 0 & 1 & \star \\ 0 & 0 & \star & 1 \end{pmatrix}$, then $\mathrm{Adj}_{\mathrm{Greedy}}$ and $\mathrm{Adj}_{\mathrm{Count}}$ take the value 1.
  - If $M = \begin{pmatrix} 1 & \star & 1 & 1 \\ \star & 1 & 0 & 0 \\ 1 & 0 & 1 & \star \\ 1 & 0 & \star & 1 \end{pmatrix}$, then $\mathrm{Adj}_{\mathrm{Greedy}}$ and $\mathrm{Adj}_{\mathrm{Count}}$ take the value 1.
  - If $M = \begin{pmatrix} 1 & \star & 1 & 0 \\ \star & 1 & 0 & 1 \\ 1 & 0 & 1 & \star \\ 0 & 1 & \star & 1 \end{pmatrix}$, then $\mathrm{Adj}_{\mathrm{Greedy}}$ and $\mathrm{Adj}_{\mathrm{Count}}$ take the value 2.
  - If $M = \begin{pmatrix} 1 & \star & 1 & 1 \\ \star & 1 & 0 & 1 \\ 1 & 0 & 1 & \star \\ 1 & 1 & \star & 1 \end{pmatrix}$, then $\mathrm{Adj}_{\mathrm{Greedy}}$ and $\mathrm{Adj}_{\mathrm{Count}}$ take the value 2.
  - If $M = \begin{pmatrix} 1 & \star & 1 & 1 \\ \star & 1 & 1 & 1 \\ 1 & 1 & 1 & \star \\ 1 & 1 & \star & 1 \end{pmatrix}$, then $\mathrm{Adj}_{\mathrm{Greedy}}$ and $\mathrm{Adj}_{\mathrm{Count}}$ take the value 2.

Now, the four variants SMA-MBM, SMA-Greedy, SMA-Count and SMA-Mean are analyzed jointly as SMA.
In scenario (a):

$$\begin{aligned} |V_1 \cap V_2| &= 1 \\ \mathrm{Adj}(V_1, V_2) &= 0 \\ \sqrt{|V_1| \cdot |V_2|} &= 2 \\ E\left[|V_1 \cap V_2| + \mathrm{Adj}(V_1, V_2)\right] &= \frac{156}{147} \\ \Rightarrow \mathrm{SMA} = \frac{1 + 0 - \frac{156}{147}}{2 - \frac{156}{147}} &= -\frac{9}{138} \end{aligned}$$

In scenario (b):

$$\begin{aligned} |V_1 \cap V_2| &= 0 \\ \text{Adj}(V_1, V_2) &= 2 \\ A(V_2, V_1) &= 2 \\ \sqrt{|V_1| \cdot |V_2|} &= 2 \\ E\left[|V_1 \cap V_2| + \text{Adj}(V_1, V_2)\right] &= \frac{156}{147} \\ \Rightarrow \text{SMA} &= \frac{0 + 2 - \frac{156}{147}}{2 - \frac{156}{147}} = 1 \end{aligned}$$

Note that for assessing the expected value, the score "$\left|\widetilde{V}_1 \cap \widetilde{V}_2\right| + \text{Adj}(\widetilde{V}_1, \widetilde{V}_2)$" is calculated for all 441 combinations of two sets with $\left|\widetilde{V}_1\right| = \left|\widetilde{V}_2\right| = 2$ and then the average value is assessed. In scenario (a), $|V_1 \cap V_2|$ is larger than in scenario (b), but the value of SMA is larger in scenario (b) than in scenario (a). Therefore, stability measure SMA cannot be a strictly increasing function of all cardinalites of pairwise intersections.

B.3 Correction for Chance

SME

If $V_1, \ldots, V_m$ are random feature sets of cardinality k, the probability that feature X_j is included in set V_i is $\frac{k}{p}$ ($\binom{p-1}{k-1}$ of $\binom{p}{k}$ possibilities) for all $j = 1, \ldots, p$ and $i = 1, \ldots, m$. As the sets are independent, the probability that feature X_j is selected exactly h_j times equals

$$\binom{m}{h_j} \left(\frac{k}{p}\right)^{h_j} \left(1 - \frac{k}{p}\right)^{m-h_j}.$$

Calculating the expected value of SME gives

$$\begin{aligned} E(\text{SME}) &= E\left(\frac{1}{q \log_2(m)} \sum_{j: X_j \in \bigcup_{i=1}^m V_i} h_j \log_2(h_j)\right) \\ &= \frac{1}{mk \log_2(m)} \sum_{j=1}^{p} E\left(\mathbb{I}_{\{1,\ldots,m\}}(h_j)\, h_j \log_2(h_j)\right) \\ &= \frac{p}{mk \log_2(m)} \cdot E\left(\mathbb{I}_{\{1,\ldots,m\}}(h_1)\, h_1 \log_2(h_1)\right) \\ &= \frac{p}{mk \log_2(m)} \cdot \sum_{h=1}^{m} h \log_2(h) \cdot P(h_1 = h) \\ &= \frac{p}{mk \log_2(m)} \cdot \sum_{h=1}^{m} h \log_2(h) \cdot \binom{m}{h} \left(\frac{k}{p}\right)^{h} \left(1 - \frac{k}{p}\right)^{m-h} \\ &= \frac{1}{m \log_2(m)} \cdot \sum_{h=1}^{m} h \log_2(h) \cdot \binom{m}{h} \left(\frac{k}{p}\right)^{h-1} \left(1 - \frac{k}{p}\right)^{m-h}. \end{aligned}$$

So, the expected value of stability measure SME for a random feature selection depends on the ratio of selected features $\frac{k}{p}$. Figure B.2 illustrates the dependence of the expected value of SME on the ratio of selected features.

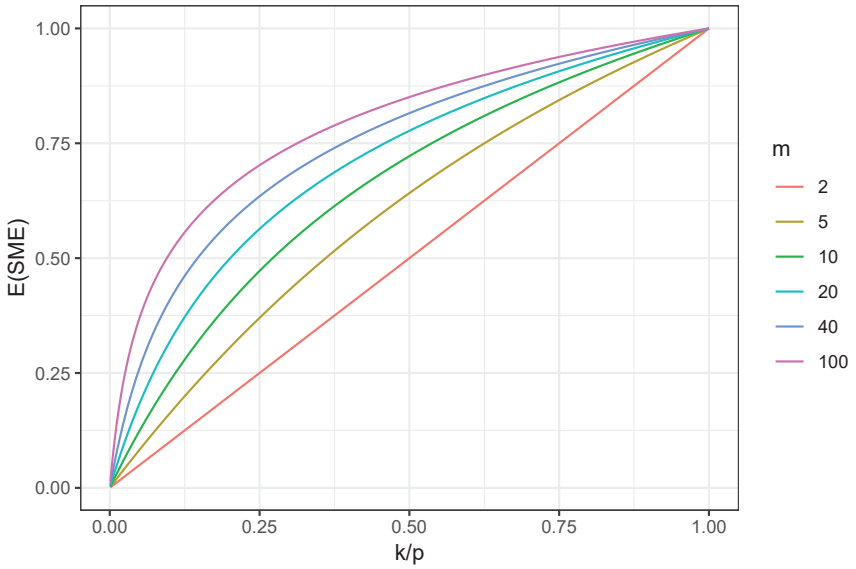


Figure B.2: Dependence of the expected value of the stability measure SME on the proportion of selected features k/p for different numbers of feature sets (m).

SMES

Consider the situation with $p = 2$, $m = 2$, similarity matrix

$$C = \begin{pmatrix} 1 & 0.95 \\ 0.95 & 1 \end{pmatrix},$$

and the two scenarios

(a) $|V_1| = 1$ and $|V_2| = 1$,

(b) $|V_1| = 1$ and $|V_2| = 2$.

In scenario (a), there are four cases:

- If $V_1 = \{X_1\}$ and $V_2 = \{X_1\}$, then SMES $= 1$.
- If $V_1 = \{X_2\}$ and $V_2 = \{X_2\}$, then SMES $= 1$.
- If $V_1 = \{X_1\}$ and $V_2 = \{X_2\}$, then SMES $= -1$.
- If $V_1 = \{X_2\}$ and $V_2 = \{X_1\}$, then SMES $= -1$.

The expected value of SMES in this scenario is the average of these four values: $E(\text{SMES}) = 0$. In scenario (b), there are two cases:

- If $V_1 = \{X_1\}$ and $V_2 = \{X_1, X_2\}$, then SMES $= -\frac{39}{41}$.
- If $V_1 = \{X_2\}$ and $V_2 = \{X_1, X_2\}$, then SMES $= -\frac{39}{41}$.

The expected value of SMES in this scenario is the average of these two values: $E(\text{SMES}) = -\frac{39}{41}$.
So, the expected value of SMES depends on the number of selected features. Therefore, stability measure SMES is not corrected for chance.

B.4 Maximum

SMU

$$\text{SMU} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i| \cdot |V_j|}{p}}}_{\leq 1, \text{ see Appendix B.1, page 150 f.}} = 1$$

$$\begin{aligned}
&\Leftrightarrow \quad \forall i < j : \frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\sqrt{|V_i| \cdot |V_j|} - \frac{|V_i| \cdot |V_j|}{p}} = 1 \\
&\Leftrightarrow \quad \forall i < j : |V_i \cap V_j| = \sqrt{|V_i| \cdot |V_j|} \\
&\Leftrightarrow \quad \forall i < j : \underbrace{\sqrt{\frac{|V_i \cap V_j|}{|V_i|}}}_{\leq 1} \cdot \underbrace{\sqrt{\frac{|V_i \cap V_j|}{|V_j|}}}_{\leq 1} = 1 \\
&\Leftrightarrow \quad \forall i < j : \frac{|V_i \cap V_j|}{|V_i|} = 1 \quad \text{and} \quad \frac{|V_i \cap V_j|}{|V_j|} = 1 \\
&\Leftrightarrow \quad \forall i < j : |V_i \cap V_j| = |V_i| = |V_j| \quad \Leftrightarrow \quad \forall i < j : V_i = V_j \\
&\Leftrightarrow \quad V_1 = \ldots = V_m
\end{aligned}$$

SMK

$$\text{SMK} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\frac{|V_i| + |V_j|}{2} - \frac{|V_i| \cdot |V_j|}{p}}}_{\leq 1, \text{ see Appendix B.1, page 152}} = 1$$

$$\begin{aligned}
&\Leftrightarrow \quad \forall i < j : \frac{|V_i \cap V_j| - \frac{|V_i| \cdot |V_j|}{p}}{\frac{|V_i| + |V_j|}{2} - \frac{|V_i| \cdot |V_j|}{p}} = 1 \\
&\Leftrightarrow \quad \forall i < j : |V_i \cap V_j| = \frac{|V_i| + |V_j|}{2} \\
&\Leftrightarrow \quad \forall i < j : \underbrace{|V_i \cap V_j|}_{\leq |V_i|} + \underbrace{|V_i \cap V_j|}_{\leq |V_j|} = |V_i| + |V_j| \\
&\Leftrightarrow \quad \forall i < j : |V_i \cap V_j| = |V_i| = |V_j| \quad \Leftrightarrow \quad \forall i < j : V_i = V_j \\
&\Leftrightarrow \quad V_1 = \ldots = V_m
\end{aligned}$$

SME

$$\text{SME} = \frac{1}{q\log_2(m)} \sum_{j:X_j \in \bigcup_{i=1}^m V_i} h_j \log_2(h_j) = 1 \quad \text{with} \quad q = \sum_{j=1}^p h_j$$

$$\Leftrightarrow \quad \sum_{j:X_j \in \bigcup_{i=1}^m V_i} h_j \log_2(h_j) = \sum_{j=1}^p h_j \log_2(m)$$

$$\Leftrightarrow \quad \sum_{j=1}^p \left\{ \begin{array}{ll} h_j \underbrace{\log_2(h_j)}_{\leq \log_2(m)}, & h_j > 0 \\ 0, & h_j = 0 \end{array} \right\} = \sum_{j=1}^p h_j \log_2(m)$$

$$\Leftrightarrow \quad \forall j \in \{1, \ldots, p\} : h_j = m \text{ or } h_j = 0$$

$$\Leftrightarrow \quad V_1 = \ldots = V_m$$

SMD-0

For $\alpha = 0$:

$$\text{SMD-0} = \max\left\{0, \ \frac{1}{|\bigcup_{i=1}^m V_i|} \sum_{j=1}^p \frac{h_j}{m} - \frac{0}{p} \cdot \text{median}\left(|V_1|, \ldots, |V_m|\right)\right\} = \frac{1}{|\bigcup_{i=1}^m V_i|} \sum_{j=1}^p \frac{h_j}{m}$$

$$\text{SMD-0} = \frac{1}{|\bigcup_{i=1}^m V_i|} \sum_{j=1}^p \frac{h_j}{m} = 1$$

$$\Leftrightarrow \quad \sum_{j=1}^p \frac{h_j}{m} = \left|\bigcup_{i=1}^m V_i\right| \quad \Leftrightarrow \quad \sum_{j=1}^p h_j = m \left|\bigcup_{i=1}^m V_i\right| \quad \Leftrightarrow \quad \sum_{j=1}^m |V_j| = \sum_{j=1}^m \left|\bigcup_{i=1}^m V_i\right|$$

$$\Leftrightarrow \quad \sum_{j=1}^m \underbrace{\left(|V_j| - \left|\bigcup_{i=1}^m V_i\right|\right)}_{\leq 0} = 0 \quad \Leftrightarrow \quad \forall j \in \{1, \ldots, m\} : |V_j| = \left|\bigcup_{i=1}^m V_i\right|$$

$$\Leftrightarrow \quad \forall j \in \{1, \ldots, m\} : V_j = \bigcup_{i=1}^m V_i \quad \Leftrightarrow \quad V_1 = \ldots = V_m$$

SMZ

$$\text{SMZ} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| + C(V_i, V_j) + C(V_j, V_i)}{|V_i \cup V_j|}}_{\leq 1, \text{ see Appendix B.1, page 157}} = 1$$

$$\Leftrightarrow \quad \forall i < j : \frac{|V_i \cap V_j| + C(V_i, V_j) + C(V_j, V_i)}{|V_i \cup V_j|} = 1$$

$$\Leftrightarrow \quad \forall i < j : |V_i \cap V_j| = |V_i \cup V_j| \quad \text{or} \quad C(V_i, V_j) + C(V_j, V_i) = |V_j \setminus V_i| + |V_i \setminus V_j|$$

$$\Leftrightarrow \quad \forall i < j : V_i = V_j \quad \text{or} \quad (|V_i| = |V_j| \quad \text{and} \quad |\text{Cor}(x, y)| = 1 \ \forall x \in V_i, y \in V_j \setminus V_i$$
$$\text{and} \quad |\text{Cor}(x, y)| = 1 \ \forall x \in V_j, y \in V_i \setminus V_j)$$

$$\Leftrightarrow \quad \forall i < j : V_i = V_j \quad \text{or} \quad (|V_i| = |V_j| \quad \text{and} \quad |\text{Cor}(x, y)| = 1 \ \forall x, y \in V_i \cup V_j)$$

So, consider the situation with $m = 2$, $V_1 = \{X_1\}$, $V_2 = \{X_2\}$, and $\text{Cor}(X_1, X_2) = 1$. Then, $|V_i \cap V_j|$, $C(V_1, V_2) = 1$, and $C(V_2, V_1)$, that is, $\text{SMZ} = \frac{0+1+1}{2} = 1$. Therefore, SMZ can attain its maximum value for unequal sets of selected features.

B.5 Maximum for Equal Cardinalities

SMW

Consider the situation with $m = 2$ and $V_1 \subsetneqq V_2$. Then $|V_1| < |V_2|$ and

$$\text{SMW} = \frac{|V_1 \cap V_2| - \frac{|V_1|\cdot|V_2|}{p}}{\min\{|V_1|, |V_2|\} - \frac{|V_1|\cdot|V_2|}{p}} = \frac{|V_1| - \frac{|V_1|\cdot|V_2|}{p}}{|V_1| - \frac{|V_1|\cdot|V_2|}{p}} = 1.$$

So, SMW can attain its maximum value for feature sets with unequal cardinalities.

SMS

Consider the example presented in Nogueira (2018): $p = 4$, $m = 4$, $V_1 = V_2 = \{X_1, X_2\}$, and $V_3 = V_4 = \{X_1, X_2, X_3\}$. So, $h_1 = h_2 = 4$, $h_3 = 2$, $h_4 = 0$, and $q = \sum_{j=1}^{4} h_j = 10$.

$$c_{\min} = \frac{10^2 - 4 \cdot (10 - 10 \bmod 4) - (10 \bmod 4)^2}{4 \cdot 10 \cdot (4-1)} = \frac{100 - 32 - 4}{120} = \frac{8}{15}$$

$$c_{\max} = \frac{(10 \bmod 4)^2 + 10 \cdot (4-1) - (10 \bmod 4) \cdot 4}{10 \cdot (4-1)} = \frac{4 + 30 - 8}{30} = \frac{13}{15}$$

$$\sum_{j=1}^{p} \frac{h_j}{q} \frac{h_j - 1}{m-1} = 2 \cdot \frac{4}{10} \cdot \frac{3}{3} + \frac{2}{10} \cdot \frac{1}{3} = \frac{4}{5} + \frac{1}{15} = \frac{13}{15}$$

$$\Rightarrow \quad \text{SMS} = \frac{\left(\sum_{j=1}^{p} \frac{h_j}{q} \frac{h_j - 1}{m-1}\right) - c_{\min}}{c_{\max} - c_{\min}} = \frac{\frac{13}{15} - \frac{8}{15}}{\frac{13}{15} - \frac{8}{15}} = 1$$

So, SMS can attain its maximum value for feature sets with unequal cardinalities.

SMY

Consider the situation with $p = 4$, $m = 2$, $V_1 = \{X_1\}$, $V_2 = \{X_2, X_3\}$, and similarity matrix

$$\begin{array}{c} \\ X_1 \\ X_2 \\ X_3 \\ X_4 \end{array} \begin{array}{c} \begin{array}{cccc} X_1 & X_2 & X_3 & X_4 \end{array} \\ \begin{pmatrix} 1 & \theta & \theta & 0 \\ \theta & 1 & \theta & 0 \\ \theta & \theta & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Then,

$$\text{SMY} = \frac{|V_1 \cap V_2| + \frac{A(V_1,V_2)+A(V_2,V_1)}{2} - E\left[|V_1 \cap V_2| + \frac{A(V_1,V_2)+A(V_2,V_1)}{2}\right]}{\frac{|V_1|+|V_2|}{2} - E\left[|V_1 \cap V_2| + \frac{A(V_1,V_2)+A(V_2,V_1)}{2}\right]}$$

with $A(V_i, V_j) = |\{x \in (V_i \setminus V_j) : \exists y \in (V_j \setminus V_i) \text{ with similarity}(x, y) \geq \theta\}|$.

- $|V_1 \cap V_2| + \frac{A(V_1,V_2)+A(V_2,V_1)}{2} = 0 + \frac{1+2}{2} = 1.5$
- $\frac{|V_1|+|V_2|}{2} = 1.5$
- $E\left[|V_1 \cap V_2| + \frac{A(V_1,V_2)+A(V_2,V_1)}{2}\right] = \frac{1}{24}\left(3 \cdot 0 + 18 \cdot 1 + 3 \cdot 1.5\right) = \frac{22.5}{24}$, see below.

V_1	V_2	$\lvert V_1 \cap V_2\rvert + \frac{A(V_1,V_2)+A(V_2,V_1)}{2}$
$\{X_1\}$	$\{X_1, X_2\}$	$1 + \frac{0+0}{2} = 1$
$\{X_1\}$	$\{X_1, X_3\}$	$1 + \frac{0+0}{2} = 1$
$\{X_1\}$	$\{X_1, X_4\}$	$1 + \frac{0+0}{2} = 1$
$\{X_1\}$	$\{X_2, X_3\}$	$0 + \frac{1+2}{2} = \frac{3}{2}$
$\{X_1\}$	$\{X_2, X_4\}$	$0 + \frac{1+1}{2} = 1$
$\{X_1\}$	$\{X_3, X_4\}$	$0 + \frac{1+1}{2} = 1$
$\{X_2\}$	$\{X_1, X_2\}$	$1 + \frac{0+0}{2} = 1$
$\{X_2\}$	$\{X_1, X_3\}$	$0 + \frac{1+2}{2} = \frac{3}{2}$
$\{X_2\}$	$\{X_1, X_4\}$	$0 + \frac{1+1}{2} = 1$
$\{X_2\}$	$\{X_2, X_3\}$	$1 + \frac{0+0}{2} = 1$
$\{X_2\}$	$\{X_2, X_4\}$	$1 + \frac{0+0}{2} = 1$
$\{X_2\}$	$\{X_3, X_4\}$	$0 + \frac{1+1}{2} = 1$
$\{X_3\}$	$\{X_1, X_2\}$	$0 + \frac{1+2}{2} = \frac{3}{2}$
$\{X_3\}$	$\{X_1, X_3\}$	$1 + \frac{0+0}{2} = 1$
$\{X_3\}$	$\{X_1, X_4\}$	$0 + \frac{1+1}{2} = 1$
$\{X_3\}$	$\{X_2, X_3\}$	$1 + \frac{0+0}{2} = 1$
$\{X_3\}$	$\{X_2, X_4\}$	$0 + \frac{1+1}{2} = 1$
$\{X_3\}$	$\{X_3, X_4\}$	$1 + \frac{0+0}{2} = 1$
$\{X_4\}$	$\{X_1, X_2\}$	$0 + \frac{0+0}{2} = 0$
$\{X_4\}$	$\{X_1, X_3\}$	$0 + \frac{0+0}{2} = 0$
$\{X_4\}$	$\{X_1, X_4\}$	$1 + \frac{0+0}{2} = 1$
$\{X_4\}$	$\{X_2, X_3\}$	$0 + \frac{0+0}{2} = 0$
$\{X_4\}$	$\{X_2, X_4\}$	$1 + \frac{0+0}{2} = 1$
$\{X_4\}$	$\{X_3, X_4\}$	$1 + \frac{0+0}{2} = 1$

$$\Rightarrow \quad \text{SMY} = \frac{1.5 - \frac{22.5}{24}}{1.5 - \frac{22.5}{24}} = 1$$

So, SMY can attain its maximum value for feature sets with unequal cardinalities.

SMA

As shown in Appendix B.1, page 160 f., the adjustment $\text{Adj}_{\text{Count}}$ is the largest of the considered adjustments, that is $\text{Adj}_{\text{MBM}} \leq \text{Adj}_{\text{Count}}$, $\text{Adj}_{\text{Greedy}} \leq \text{Adj}_{\text{Count}}$, and $\text{Adj}_{\text{Mean}} \leq \text{Adj}_{\text{Count}}$. For $\text{Adj} \in \{\text{Adj}_{\text{Count}}, \text{Adj}_{\text{MBM}}, \text{Adj}_{\text{Greedy}}, \text{Adj}_{\text{Mean}}\}$ the following holds:

$$\text{SMA} = \frac{2}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \underbrace{\frac{|V_i \cap V_j| + \text{Adj}(V_i, V_j) - E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right]}{\sqrt{|V_i| \cdot |V_j|} - E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right]}}_{\leq 1, \text{ see Appendix B.1, page 160 f.}} = 1$$

$$\Leftrightarrow \quad \forall i < j : \frac{|V_i \cap V_j| + \text{Adj}(V_i, V_j) - E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right]}{\sqrt{|V_i| \cdot |V_j|} - E\left[|V_i \cap V_j| + \text{Adj}(V_i, V_j)\right]} = 1$$

$$\Leftrightarrow \quad \forall i < j : |V_i \cap V_j| + \text{Adj}(V_i, V_j) = \sqrt{|V_i| \cdot |V_j|}$$

Consider arbitrary non-empty sets V_i and V_j and w.l.o.g. assume that $|V_i| \leq |V_j|$.

- Case 1: $\text{Adj}(V_i, V_j) = 0$:
  As shown for SMU on page 170, $|V_i \cap V_j| = \sqrt{|V_i| \cdot |V_j|} \Leftrightarrow |V_i| = |V_j|$
- Case 2: $\text{Adj}(V_i, V_j) > 0$:
  Show $|V_i| = |V_j|$ by proof by contradiction:
  Assume that $|V_i| < |V_j|$. Then, $\sqrt{|V_i| \cdot |V_j|} = |V_i \cap V_j| + \text{Adj}(V_i, V_j) \leq |V_i \cap V_j| + \text{Adj}_{\text{Count}}(V_i, V_j) \leq |V_i \cap V_j| + |V_i \setminus V_j| = |V_i| < \sqrt{|V_i| \cdot |V_j|}$ ↯, see Appendix B.1, page 160 f.

Concludingly, $\text{SMA} = 1 \Rightarrow |V_1| = \ldots = |V_m|$. Also, it was shown in Appendix B.1, page 160 f., that SMA-Count, SMA-MBM, SMA-Greedy, and SMA-Mean attain their maximum value 1 for $V_1 = \ldots = V_m$, which is a situation with all feature sets having the same cardinality.

Eidesstattliche Erklärung

Hiermit erkläre ich, Andrea Martina Bommert, dass ich die vorliegende Dissertation mit dem Titel “Integration of Feature Selection Stability in Model Fitting” selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Dissertation ist bisher keiner anderen Fakultät vorgelegt worden. Ich erkläre, dass ich bisher kein Promotionsverfahren erfolglos beendet habe und dass keine Aberkennung eines bereits erworbenen Doktorgrades vorliegt.