LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

INSTITUT FÜR STATISTIK

Roman Hornung, Christoph Bernau, Caroline Truntzer, Thomas Stadler, Anne-Laure Boulesteix

Full versus incomplete cross-validation: measuring the impact of imperfect separation between training and test sets in prediction error estimation

Technical Report Number 159, 2014
Department of Statistics
University of Munich

http://www.stat.uni-muenchen.de

Full versus incomplete cross-validation: measuring the impact of imperfect separation between training and test sets in prediction error estimation

Roman Hornung[1‡] Christoph Bernau[1] Caroline Truntzer[2]
Thomas Stadler[3] Anne-Laure Boulesteix[1]

April 17, 2014

[1] Department of Medical Informatics, Biometry and Epidemiology, University of Munich, D-81377, Munich

[2] Clinical Innovation Proteomic Platform, Centre Hospitalo-Universitaire de Dijon, F-21000, Dijon

[3] Department of Urology, University of Munich, D-81377, Munich

Abstract

In practical applications of supervised statistical learning the separation of the training and test data is often violated through performing one or several analysis steps prior to estimating the prediction error by cross-validation (CV) procedures. We refer to such practices as incomplete CV. For the special case of preliminary variable selection in high-dimensional microarray data the corresponding error estimate is well known to be strongly downwardly biased, resulting in over-optimistic conclusions regarding prediction accuracy of the fitted models. However, while other data preparation steps may also be affected by these types of problems, their impact on error estimation is far less acknowledged in the literature. In this paper we shed light on these issues. We present a new measure quantifying the impact of incomplete CV that is based on the ratio between the errors estimated by incomplete CV and by a formally correct "full CV." The new measure is illustrated through applications to several low- and high-dimensional biomedical data sets and various data preparation steps including preliminary variable selection, choice of tuning parameters, normalization of gene expression microarray data, and imputation of missing values. It may be used in biometrical applications to determine whether specific data preparation steps can be safely performed as preliminary steps before running the CV procedure, or if they should be repeatedly trained in each CV iteration.

[‡]Corresponding author. Email: hornung@ibe.med.uni-muenchen.de.

1 Introduction

In supervised statistical learning, it is widely recognized that prediction models should not be constructed and evaluated using the same data set. While the training data set is used for all steps towards obtaining the prediction rule, the test data set is used to evaluate its prediction error and should ideally not be opened during the training phase. In practice, however, the data set is often too small to be split into a training set and a test set. For example, many microarray studies include no more than 50 or 100 patients. It is then standard practice to estimate prediction error using cross-validation (CV) or a related procedure – these roughly consist of considering several splittings into training and test data successively and averaging the estimated prediction errors resulting from the different iterations. From this point let CV denote such procedures, but all ideas and procedures can be directly extended to other resampling techniques used for prediction error estimation, such as repeated subsampling or bootstrapping.

By "incomplete CV" (Simon et al., 2003), we mean a CV procedure in which the excluded folds are somehow taken into account in the construction of the prediction rules. The analyses are thereby only partially repeated in each CV iteration: some analysis steps are performed beforehand using the whole data set, i.e. for these steps no distinction is made between training and test data. In contrast, if all steps leading to the prediction rules are trained in each CV iteration on the corresponding training set, the CV procedure is denoted by "full CV." Incomplete CV is well known to yield strongly biased error estimates resulting in over-optimistic conclusions if supervised variable selection is not repeated in each CV iteration (Ambroise and McLachlan, 2002). However, it is far less acknowledged that also other cases in which analysis steps are performed on the whole data set beforehand are potentially dangerous with respect to over-optimistic error estimation.

In fact, beyond variable selection in the context of high-dimensional prediction models, many data preparation procedures are commonly conducted as preliminary steps before the main analyses are performed. For example, raw data from high-throughput biological experiments such as microarrays have to be properly background-corrected, normalized and summarized be-

fore so-called high-level analyses such as predictive modeling can be conducted. In the context of large-scale biological data, it is also common practice to exclude from further analyses those features that show poor variability across the observations — irrespective of their effect on the response variable. Another important step to be performed before the analysis of experimental data from high-throughput technologies is the removal of potential batch effects arising when data are measured, say, by different machines or in different labs (Leek et al., 2010). With any type of data, missing values are a common issue that is often addressed by imputation techniques. Dichotomization, i.e. categorization of (some of) the features into binary features, is often performed according to thresholds determined from the data — for instance the median value (or another quantile). More generally, data-driven transformations of the features are commonly applied to better take non-linear effects on the response into account, for example fractional polynomials with powers determined from the data. Data preparation techniques are by far not limited to these few examples. In particular, data with complex structures, like for example imaging data, often require sophisticated preprocessing steps for making raw data generated by instruments analysable by standard statistical tools. We do not intend to provide an exhaustive list at this stage, but stress that such preliminary data preparation steps are extremely diverse and common in most biomedical data analysis fields.

Let us now come back to the problem of prediction error estimation through CV. Both in methodological studies (dealing with the development or comparison of methods) or applications (dealing with concrete data sets of current substantive interest), such preprocessing procedures are almost always performed using the whole data set, i.e. not repeated in each CV iteration. This is rarely mentioned in the description of the CV method, in contrast to the level of detail of information usually provided on the methods used for deriving prediction rules or the parameters of the CV procedure such as the number of folds or number of replications of CV. A possible reason for this lack of attention might be that this problem is less relevant in traditional, low-dimensional statistical applications, since the steps conducted to obtain final results are typically fewer than in modern biomedical applications and more directly related to the main analysis method.

It is not clear whether it is necessary to perform data preparation steps for each training data set successively, i.e. in each iteration anew. Since CV aims to mimic the prediction error that would be observed for future independent data, it is in principle required to fully ignore the test data in each iteration until the prediction rule for this iteration has been completely derived. Theoretically, the test data should also not be taken into account also in any data preparation step either. Otherwise it would imply a violation of the principle of separation between training and test data and possibly introduce an optimistic bias in the error estimation.

However, it is not obvious whether such a violation of the separation between training and test data has a significant impact on error estimation in the case of data preparation. The answer to this question is expected to essentially depend on the considered specific data preparation procedure. Note that it is not always straightforward to set up the right procedure for preparing the test data after training the data preparation on the training data. The procedure must perform the data preparation step for an observation in the test data in exactly the same way as for a corresponding observation in the training data. Here it is important that the trained data preparation step is unaffected by the test observation — this would for example not be the case when re-training the step including the test observation. Hence we use the term "addon procedure", which was originally introduced in the specific case of normalization procedures for microarray data (Kostka and Spang, 2008) but is employed in a more general sense here for all types of data preparation steps. Note at this point that by "performing" a data preparation step we mean the following procedure: 1) Conduct the data preparation procedure on the considered data; 2) Store all information necessary for addon preparation of new observations. Addon procedures are trivial in some cases, for instance in the case of dichotomization according to a cutpoint determined from the training data. In other cases like normalization of microarray data, however, this task can be more complex. A naive and straightforward procedure would be to prepare the test data completely independently without using any information from the preparation of the training data. But this has the disadvantage of increasing the prediction error or, for example in the case of variable selection, might make training

and test data incomparable when the result of the data preparation has a different structure in the test data than in the training data. Moreover, it requires a "large enough" test data set, a condition which is not fulfilled in CV for small sample data.

To our knowledge, the potential impact on error estimation of including the test data for specific steps has not been given any attention in the literature, except for the special case of variable selection as already mentioned above. Our paper aims at filling this gap.

We present a new measure quantifying the impact of incomplete CV, denoted as CVIIM, — standing for "CV Incompleteness Impact Measure." It is based on the ratio between the CV prediction errors resulting when specific steps are trained only once using the whole data set and when these procedures are trained in each CV iteration, i.e. for each considered training data set anew and subsequently applied to the excluded fold via addon procedures. The new measure is intended to be used by methodological researchers or statisticians working on statistical learning applications to determine whether a particular preparation step should be considered as part of the prediction rule fitting process and trained in each CV iteration successively - at the price of a (substantially) higher computational effort - or whether it can be safely performed as a preliminary step on the whole data set without generating an important optimistic bias. Using several real-life low- or high-dimensional data sets from the biomedical field, we investigate CVIIM's behavior for various data preparation steps: preliminary variable selection, choice of tuning parameters, normalization and batch effect removal (in the case of high-throughput omics data), and imputation of missing values.

2 Methods

2.1 Notations and settings

The predictor space is denoted as $\mathcal{X} \subset \mathbb{R}^p$ and the space of the response variable as $\mathcal{Y} = \{1, 2\}$. Let $\boldsymbol{S} = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ be an *i.i.d.* random sample following the distribution P^n. Most importantly, in our paper $\boldsymbol{x} \in \mathcal{X}$ denotes the "raw" data, meaning that these predictors may be sub-

ject to data preparation steps (possibly modifying their number or scale of measurement) before being used as predictors in classification.

We consider a classification function $g : \mathcal{X} \mapsto \mathcal{Y}$, $\boldsymbol{x} \mapsto g(\boldsymbol{x})$ that takes the vector $\boldsymbol{x}$ as an argument and returns a prediction of the value of the target variable y. For example, consider a classification task where, based on microarray samples, patients are classified as having cancer or not using the Nearest Shrunken Centroids approach (Tibshirani et al., 2002). Then the corresponding function g would take the pre-normalized expression values as an argument, perform normalization and classify the sample using a certain value of the shrinkage parameter. These steps are assumed to be performed in an ideal way, where all occurring parameters are estimated or optimized using a hypothetical data set with sample size tending to infinity.

In practice g is estimated from the available data. We therefore denote the classification function estimated from $\boldsymbol{S}$ as $\hat{g}_{\boldsymbol{S}} : \mathcal{X} \mapsto \mathcal{Y}$, $\boldsymbol{x} \mapsto \hat{g}_{\boldsymbol{S}}(\boldsymbol{x})$. In the example outlined above, this means that the parameters involved in the normalization procedure as well as the averages and variances involved in the Nearest Shrunken Centroids classifier are estimated from $\boldsymbol{S}$ and that the shrinkage parameter is also chosen based on $\boldsymbol{S}$. The estimated classification function $\hat{g}_{\boldsymbol{S}}$ can then be used to predict y for a new observation.

Note that — as already outlined in the introduction — depending on the procedures involved in the estimation, it is not always straightforward to construct such a function $\hat{g}_{\boldsymbol{S}}$ that can be applied to predict independent data. For example, while normalization of microarray data is done easily on the training sets, it is not straightforward how to normalize a new observation using the same scheme in order to make it comparable to observations in the training set. See for instance the addon procedure described by Kostka and Spang (2008) that uses the quantiles from the training data for quantile normalization in the special case of the normalization procedure 'RMA' (Irizarry et al., 2003). From now on, we will however assume that such methods are available and that we can thus construct the function $\hat{g}_{\boldsymbol{S}}$.

It is important to assess the prediction error of $\hat{g}_{\boldsymbol{S}}$ that is defined as

$$\varepsilon[\hat{g}_{\boldsymbol{S}}] := \mathbb{E}_{(\boldsymbol{X},Y)\sim P}[L(\hat{g}_{\boldsymbol{S}}(\boldsymbol{X}), Y)] = \int_{\mathcal{X}\times\mathcal{Y}} L(\hat{g}_{\boldsymbol{S}}(\boldsymbol{x}), y)\; dP(\boldsymbol{x}, y), \qquad (1)$$

whereby $L(\cdot,\cdot)$ is an adequate loss function, for example the indicator loss yielding the misclassification error rate used in the present paper. This error is commonly termed "conditional" because it refers to the specific sample $\boldsymbol{S}$. The *average error* over all samples following P^n is referred to as the unconditional error and denoted as $\varepsilon(n) \;:=\; \mathbb{E}_{\boldsymbol{S}\sim P^n}[\,\varepsilon[\hat{g}_{\boldsymbol{S}}]\,]$.

Let from now on $\boldsymbol{s} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ denote a realization of the random sample $\boldsymbol{S}$. If we had a large independent sample at hand we could estimate $\varepsilon[\hat{g}_{\boldsymbol{s}}]$ directly by comparing the true values of the target variable in this data to the predictions made by $\hat{g}_{\boldsymbol{s}}$. Having only $\boldsymbol{s}$ at hand a naive approach would be to estimate $\varepsilon[\hat{g}_{\boldsymbol{s}}]$ using $\boldsymbol{s}$ itself as test data. This approach yields the so-called "apparent error" or "resubstitution error" that is well known to be downwardly biased (i.e., too optimistic) as an estimator of $\varepsilon[\hat{g}_{\boldsymbol{s}}]$, since estimation of the classification function and error estimation are conducted on the same data. As already noted in the introduction, resampling-based error estimation can be performed to address this issue. The sample $\boldsymbol{s}$ is iteratively split into non-overlapping training and test data sets. In each iteration the function g is estimated based on the training set and the error of this estimated function is assessed based on the test set. In this paper we consider cross-validation—the most widely used of these resampling-based approaches.

Given a random partition of the data set $\boldsymbol{s}$ into K approximately equally sized folds $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_K$, the K-fold cross-validation error estimate is given as

$$\frac{1}{K}\sum_{k=1}^{K} \frac{1}{\#\boldsymbol{s}_k} \sum_{j\,\in\,\{i\,:\,(\boldsymbol{x}_i,y_i)\,\in\,\boldsymbol{s}_k\}} L(\hat{g}_{\boldsymbol{s}\setminus\boldsymbol{s}_k}(\boldsymbol{x}_j), y_j), \qquad (2)$$

whereby # represents the cardinality, $\boldsymbol{s}\setminus\boldsymbol{s}_k$ is the training set in iteration k and $\boldsymbol{s}_k$ is the test set. Since this estimate (highly) depends on the considered random partition of the sample $\boldsymbol{s}$ into K folds, it is recommended to repeat this procedure $B > 1$ times and average the error estimates over the B repetitions. With $\boldsymbol{s}_{b1}, \ldots, \boldsymbol{s}_{bK}$ denoting the folds considered in the b-th

repetition, the repeated K-fold cross-validation error estimate is given as

$$e_K(\boldsymbol{s}) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{\#\boldsymbol{s}_{bk}}\sum_{j\,\in\,\{i\,:\,(\boldsymbol{x}_i,y_i)\,\in\,\boldsymbol{s}_{bk}\}} L(\hat{g}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}(\boldsymbol{x}_j), y_j). \qquad (3)$$

If, for simplicity, we assume that the $\boldsymbol{s}_{b1}, \ldots, \boldsymbol{s}_{bK}$ $(b = 1, \ldots, B)$ are equally sized and denote $n_{train,K} := \#\boldsymbol{s}\setminus\boldsymbol{s}_{bk}$ with $b \in \{1, \ldots, B\}$ and $k \in \{1, \ldots, K\}$, it can be easily seen that $e_K(\boldsymbol{s})$ is an unbiased estimator for $\varepsilon(n_{train,K})$ and therefore an upwardly biased estimator for $\varepsilon(n)$. This bias is called the "inherent bias" of CV in Varma and Simon (2006). Note that the notation $e_K(\boldsymbol{s})$ does not reflect the fact that the repeated K-fold cross-validation error estimate depends on the random partitions in the B iterations. For our purpose we assume B to be chosen large enough so that this dependency can be ignored.

2.2 Incomplete versus full cross-validation

As outlined in the previous section error estimation should not be performed based on the data that were used to estimate the classification function. However, a common practice already described in the introduction is to perform specific data analysis steps as "preliminary steps," using the whole sample $\boldsymbol{s}$, i.e., before splitting it into training and test data sets. With this issue in mind, we introduce the notation

$$\hat{g}^{\boldsymbol{a}_2}_{\boldsymbol{a}_1} : \mathcal{X} \mapsto \mathcal{Y} \qquad \boldsymbol{x} \mapsto \hat{g}^{\boldsymbol{a}_2}_{\boldsymbol{a}_1}(\boldsymbol{x}) \qquad \boldsymbol{a}_1 \subseteq \boldsymbol{a}_2 \subseteq \boldsymbol{s} \qquad (4)$$

to denote an estimated classification function that is estimated partly based on a sample $\boldsymbol{a}_2$ and partly based on a possibly smaller subsample $\boldsymbol{a}_1$ (i.e., some steps may be performed on a bigger sample). Let us come back to our example of the microarray-based classification. It is common practice to run the normalization procedure and often also the parameter tuning based on the whole data set $\boldsymbol{s}$, whereas the training of the classifier is performed within cross-validation, i.e., based only on the training set $\boldsymbol{s} \setminus \boldsymbol{s}_{bk}$ in each iteration k of each repetition b. In this scenario $\boldsymbol{a}_2$ would be the whole data set $\boldsymbol{s}$ and in each CV iteration $\boldsymbol{a}_1$ would be the training set $\boldsymbol{s}\setminus\boldsymbol{s}_{bk}$.

With $\boldsymbol{a}_1 = \boldsymbol{s} \setminus \boldsymbol{s}_{bk}$ and $\boldsymbol{a}_2 = \boldsymbol{s}$ for $b = 1, \ldots, B$ and $k = 1, \ldots, K$, we obtain an incomplete CV error estimate that is downwardly biased as an

estimator of $\varepsilon(n_{train,K})$:

$$e_{incompl,K}(\boldsymbol{s}) := \frac{1}{B}\sum_{b=1}^{B}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{\#\boldsymbol{s}_{bk}}\sum_{j\,\in\,\{i\,:\,(\boldsymbol{x}_i,y_i)\,\in\,\boldsymbol{s}_{bk}\}} L(\hat{g}^{\boldsymbol{s}}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}(\boldsymbol{x}_j), y_j) \quad (5)$$

where the index "incompl" indicates that the whole sample $\boldsymbol{s}$ is used for at least part of the data analysis steps required for the estimation of g, and that the resulting CV procedure can thus be seen as incomplete according to our definition given in the introduction. The extent of the downward bias of $e_{incompl,K}(\boldsymbol{s})$ depends on how strongly the specific analysis step(s) conducted on the whole data set increase(s) the apparent homogeneity of the predictors across observations and/or the apparent association between response and predictors. The estimator $e_{incompl,K}(\boldsymbol{s})$ is unbiased as an estimator of the *average incomplete error* $\varepsilon_{incompl}(n_{train,K}; n) := \mathbb{E}_{\boldsymbol{S}\sim P^n}[L(\hat{g}^{\boldsymbol{S}}_{\boldsymbol{S}_{train,K}}(\boldsymbol{X}_{n_{train,K}+1}), Y_{n_{train,K}+1})]$, with $\boldsymbol{S}_{train,K} = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_{n_{train,K}}, Y_{n_{train,K}})\}$ and $\boldsymbol{X}_{n_{train,K}+1}$ playing the role of an arbitrary test set observation. We assume exchangeability of the random observations in $\boldsymbol{S}$. The quantity $\varepsilon_{incompl}(n_{train,K}; n)$ will be important, when we define the true value of our measure in the next section.

Furthermore, since by definition $\hat{g}^{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}} = \hat{g}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}$, we obtain the usual repeated K-fold error estimate from Eq. (3) if we set $\boldsymbol{a}_1 = \boldsymbol{a}_2 = \boldsymbol{s}\setminus\boldsymbol{s}_{bk}$ for $k = 1, \ldots, K$, and $b = 1, \ldots, B$. This estimator is denoted as $e_{full,K}(\boldsymbol{s})$:

$$e_{full,K}(\boldsymbol{s}) := e_K(\boldsymbol{s}) = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{K}\sum_{k=1}^{K}\frac{1}{\#\boldsymbol{s}_{bk}}\sum_{j\,\in\,\{i\,:\,(\boldsymbol{x}_i,y_i)\,\in\,\boldsymbol{s}_{bk}\}} L(\hat{g}^{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}_{\boldsymbol{s}\setminus\boldsymbol{s}_{bk}}(\boldsymbol{x}_j), y_j), \quad (6)$$

where the index "full" underlines that *all* steps are conducted within the CV procedure, i.e., using the training sets only. Given the equivalence of $e_{full,K}(\boldsymbol{s})$ and $e_K(\boldsymbol{s})$, we will from now on also add the index "full" to the true error $\varepsilon(n_{train,K})$, that is write $\varepsilon_{full}(n_{train,K})$, to better distinguish it from $\varepsilon_{incompl}(n_{train,K}; n)$ in the notation. In the next section we present our simple measure based on $\varepsilon_{full}(n_{train,K})$ and $\varepsilon_{incompl}(n_{train,K}; n)$ to assess the impact of CV incompleteness.

2.3 A new measure of the impact of CV incompleteness (CVIIM)

Our new measure CVIIM (standing for "Cross-Validation Incompleteness Impact Measure") is defined as

$$\text{CVIIM}_{P,n,K} = \begin{cases} 1 - \dfrac{\varepsilon_{incompl}(n_{train,K}; n)}{\varepsilon_{full}(n_{train,K})} & \text{if } \varepsilon_{incompl}(n_{train,K}; n) < \varepsilon_{full}(n_{train,K}) \\ & \text{and } \varepsilon_{full}(n_{train,K}) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$\in [0, 1]$. It is based on the ratio $\varepsilon_{incompl}(n_{train,K}; n)/\varepsilon_{full}(n_{train,K})$ of the two errors, which is more appropriate than using the difference $\varepsilon_{full}(n_{train,K}) - \varepsilon_{incompl}(n_{train,K}; n)$ as a measure of the impact of CV incompleteness, since the latter would strongly depend on the value of $\varepsilon_{full}(n_{train,K})$ (large values of $\varepsilon_{full}(n_{train,K})$ leading to large differences). Truncation of $1 - \varepsilon_{incompl}(n_{train,K}; n)/\varepsilon_{full}(n_{train,K})$ at 0 prevents CVIIM from being negative in the case where $\varepsilon_{incompl}(n_{train,K}; n) > \varepsilon_{full}(n_{train,K})$, finally leading to a measure in [0,1]. A large value of CVIIM indicates that CV incompleteness implies a large underestimation of prediction error.

Since, as we have seen in the last section, $e_{incompl,K}(\boldsymbol{s})$ is an unbiased estimator for $\varepsilon_{incompl}(n_{train,K}; n)$ and $e_{full,K}(\boldsymbol{s})$ is an unbiased estimator for $\varepsilon_{full}(n_{train,K})$, we suggest replacing the true errors by their estimators in equation (7) to obtain an estimator for $\text{CVIIM}_{P,n,K}$, denoted as $\text{CVIIM}_{\boldsymbol{s},n,K}$.

3 Illustration: study design

3.1 General procedure

Using various real-life data sets, we investigate the impact of incomplete CV with respect to the following analysis steps: supervised variable selection, variable filtering by variance, choice of tuning parameters for various classification methods, imputation using a variant of k-Nearest-Neighbors (Ripley,

1996; Troyanskaya et al., 2001), normalization with the RMA method, and batch effect removal using ComBat (Johnson et al., 2007).

$\text{CVIIM}_{\boldsymbol{s},n,K}$ is calculated as described in Section 2.3. Since CV error estimates are highly variable, we perform $B = 300$ repetitions of CV. We use the same random partitions to estimate $\varepsilon_{full}(n_{train,K})$ and $\varepsilon_{incompl}(n_{train,K}; n)$. Since the procedures of parameter tuning and imputation integrate a CV procedure in the estimation (see Section 3.3) they are not completely deterministic and give differing results when repeated. Therefore if we trained them just once using the whole data set in the process of estimating $\varepsilon_{incompl}(n_{train,K}; n)$, the result would be volatile and would noticeably change when repeating the estimation procedure. As a solution to this problem in the cases of parameter tuning and imputation, the whole data set is used for training these steps anew before each of the 300 repetitions of the CV. For all other data preparation steps, the result is deterministic, wherefore these are trained just once using the whole data set and have to be repeated only in the process of estimating $\hat{g}^{s\setminus s_{bk}}_{s\setminus s_{bk}}$. We consider the following established splitting ratios between the sizes of the training and test sets: 2:1 (3-fold CV), 4:1 (5-fold CV) and 9:1 (10-fold CV).

3.2 Data sets

Table 1 gives an overview of the data sets considered in our illustrative study. The references for the individual data sets are found in Web Appendix A. The data sets `GenitInfCoww0`, ..., `GenitInfCoww4` contain data on major genital infection in cows in different weeks post-partum. The data sets `BreastCancerConcatenation`, `HCCProteom` and `SarcoidosisTranscr` contain batches; the sizes of the individual batches are given below the total number of samples in Table 1. `BreastCancerConcatenation` is thereby a concatenation of four independent transcriptomic data sets from studies of breast cancer. All data sets involve a binary response variable, which indicates whether the specific disease is present or not in the individual patients.

Name	number of samples	number of variables	% diseased	type of variables	disease
`ProstatecTranscr`	102	12,625	51%	transcriptomic	prostate cancer
`HeadNeckcTranscr`	50	22,011	50%	transcriptomic	head and neck squamous
`LungcTranscr`	100	22,277	49%	transcriptomic	lung Adenocarcinoma
`SLETranscr`	36	47,231	56%	transcriptomic	systemic lupus erythematosus
`GenitInfCoww0`	51	21	71%	various	major genital infection in cows
`GenitInfCoww1`	51	24	71%	various	major genital infection in cows
`GenitInfCoww2`	51	27	71%	various	major genital infection in cows
`GenitInfCoww3`	51	26	71%	various	major genital infection in cows
`GenitInfCoww4`	51	27	71%	various	major genital infection in cows
`ProstatecMethyl` (supplied by the fourth author)	70	222	41%	methylation	prostate cancer
`ColoncTranscr`	47	22,283	53%	transcriptomic	colon cancer
`WilmsTumorTranscr`	100	22,283	42%	transcriptomic	Wilms' tumor
`HCCProteom` *by batch* (supplied by the CIRCE group)	268 (58/60/64/86)	127	50%	proteomic	Hepatocellular carcinoma
`BreastCancerConcatenation` *by batch*	148 (86/10/19/33)	22,277	67%	transcriptomic	breast cancer
`SarcoidosisTranscr` *by batch*	58 (20/20/18)	54,675	66%	transcriptomic	sacroidosis

Table 1: Overview of used data sets

3.3 Exact (training) data preparation procedures performed in the individual analyses

In this section we outline the exact proceedings for the individual analysis procedures performed for each of the investigated data preparation steps.

Supervised variable selection For every variable a two-sample t-test is conducted with respect to the two classes. The variables with the smallest p-values are selected. Since it is expected that the result substantially depends on the number of selected variables, the analysis is repeated for different numbers of variables: 5, 10, 20 and half of the total number p of variables. When selecting 5, 10 and 20 variables we use Linear Discriminant Analysis as a classifier and when selecting half of the variables we use Diagonal Linear Discriminant Analysis, which is very similar to applying the Nearest Shrunken Centroids method with a shrinkage parameter of zero; see Hastie et al. (2009). We use the data sets `ProstatecTranscr`, `HeadNeckcTranscr`, `LungcTranscr` and `SLETranscr` for this procedure.

Variable filtering by variance For every variable the empirical variance is calculated and the $p/2$ variables with the largest variances are selected. Diagonal Linear Discriminant Analysis is again used here as a classifier. The same data sets as in the case of supervised variable selection are considered.

Optimization of tuning parameters We optimize tuning parameters on a grid for seven different classification methods: number of iterations m_{stop} in componentwise boosting with logistic loss function (grid: 50, 100, 200, 500, 1000) (Bühlmann and Yu, 2003), number of neighbors in the k-Nearest-Neighbors algorithm (grid: $1, 2, \ldots, 10$), L_1 shrinkage intensity in Lasso expressed as the fraction of the coefficient L_1-norm compared to the maximum possible L_1-norm (grid: $0.1, 0.2, \ldots, 0.9$) (Young-Park and Hastie, 2007), shrinkage intensity for the class centroids in Nearest Shrunken Centroids (grid: 0.1, 0.25, 0.5, 1, 2, 5), number of components in Linear Discriminant Analysis on Partial Least Squares components (grid: $1, 2, \ldots, 10$) (Boulesteix and Strimmer, 2007), number $mtry$ of variables randomly sampled as candidates at each split in Random Forests (grid: 1, 5, 10, 50, 100, 500) (Breiman, 2001) and cost of constraints violation in Support Vector Machines with linear kernel (grid: $\{10^{-5} \cdot 40^{k/7} : k = 0, \ldots, 7\}$) (Schölkopf

and Smola, 2002).

The optimization is done in the following way. For each candidate value of the tuning parameter on its respective grid, we perform a 3-fold CV of the classifier using this value of the tuning parameter. The value yielding the smallest 3-fold CV error is selected. Again the same data sets are used for this procedure.

Bernau et al. (2013) provide an estimation procedure for the error of wrapper algorithms which can be employed as an interesting, computationally much cheaper alternative to full CV in the case of grid-based tuning.

Imputation of missing values We perform k-Nearest-Neighbors imputation (Wong, 2013), a procedure that is commonly used for the analysis of high-dimensional microarray data. Prior to imputation the variables are centered and scaled and the estimated means and standard deviations are stored. After imputing the values, they are rescaled using the stored standard deviations and the stored means are added to retransform the data to the original level. The result of the imputation can be assumed to depend critically on the number k of nearest neighbors considered. Therefore to optimize this parameter on the grid $1, 2, 3, 5, 10, 15$ we employ 3-fold CV in an analogous way as described for tuning above. For a correct addon-imputation, besides using the means and standard deviations estimated from the training data, we also have to consider only the training data when searching for the k nearest neighbors. We use Nearest Shrunken Centroids as a classifier for the high-dimensional data set and Random Forests for the other data sets. Here for optimizing the shrinkage intensity and *mtry* respectively we again employ 3-fold CV in the described way. We consider the `GenitInfCow` data sets and `ProstatecMethyl`.

Normalization Microarray data preparation is performed via the RMA algorithm (standing for Robust Multi-array Average) including the addon procedure provided by Kostka and Spang (2008). Here the quantiles from the training data sets are used for the quantile normalization of the test data. Since background correction and summarization are performed on an array-by-array basis, no addon strategies are necessary for these procedures.

For classification, Nearest Shrunken Centroids optimizing the shrinkage intensity is employed. For this procedure we use the following data sets: `ColoncTranscr` and `WilmsTumorTranscr`.

Removal of batch effects We use the parametric version of the ComBat method (Johnson et al., 2007; Leek et al., 2012). Here a batch-specific location and scale adjustment is done on the variables, with the peculiarity that the estimation of the corresponding parameters is stabilized by fitting prior distributions in an empirical Bayes framework. An obvious restriction to the partitions into the K folds generated in the repeated CV is that batches present in a test data set have to be represented in the corresponding training data set as well. This is needed to perform addon batch effect removal, since the ComBat-model involves batch-specific parameters. More precisely it is required that there are two observations per batch present in the training data. This is because for estimating the parameters of the prior distribution for the scale-shift parameters the batch-specific variances of the variables have to be estimated. Therefore if a partition into K folds does not meet this restriction, further are drawn until an admissible partition is generated. However for our data sets non-admissible partitions occur only rarely. Nearest Shrunken Centroids optimizing the shrinkage intensity is again used as classification method. The considered data sets were `HCCProteom`, `BreastCancerConcatenation` and `SarcoidosisTranscr`.

4 Results

The results of the analyses are displayed in Figure 1 (supervised variable selection and filtering by variance), Figure 2 (optimization of tuning parameters) and Figure 3 (imputation of missing values, normalization, batch effect removal) and listed in the Web Tables 1, 2 and 3 found in Web Appendix B. In the plots the error bars represent the 25%- and 75%- quartiles (computed over the $B = 300$ iterations) of the iterationwise non-truncated incompleteness measure estimates $\text{CVIIM}_{\boldsymbol{s},n,K,b} = 1 - e_{incompl,K}(\boldsymbol{s})_b / e_{full,K}(\boldsymbol{s})_b$, where the index b indicates that these errors are obtained for iteration b (with $b = 1, \ldots, B$). While these error bars reflect the variability of the incompleteness measure over the $B = 300$ iterations we are actually interested in the variability of the $\text{CVIIM}_{\boldsymbol{s},n,K}$-values themselves, which is hard to esti-

mate. However, we assume that the variability of the $\text{CVIIM}_{\boldsymbol{s},n,K}$-values is also expressed in the observed variability of the $\text{CVIIM}_{\boldsymbol{s},n,K,b}$-values. It is important to note that the error bars should be used for comparisons between each other only, since their absolute lengths have no relevant interpretation.

The measure $\text{CVIIM}_{P,n,K}$ can be interpreted in terms of the relative reduction in the estimated error resulting from CV incompleteness with respect to the considered data preparation step(s). With this interpretation in mind and based on the results of our analyses compared to our expectations regarding the impact of the considered data preparation steps, we define the following tentative rules of thumb for categorizing the computed values: $[0, 0.02] \sim$ no influence, $]0.02, 0.1] \sim$ weak, $]0.1, 0.2] \sim$ medium, $]0.2, 0.4] \sim$ strong, $]0.4, 1] \sim$ very strong.

In three out of four data sets the errors were not smaller when performing filtering by variance on the whole data set, leading to zero values of $\text{CVIIM}_{\boldsymbol{s},n,K}$ and in the fourth data set it was almost zero (see Figure 1 and Web Table 1). According to our examples filtering by variance thus seems to be an analysis step that can be safely conducted using the whole data set. The results are very different when performing supervised variable selection. While $\text{CVIIM}_{\boldsymbol{s},n,K}$ is especially large for small numbers of selected variables, relatively large values are also observed when half of the variables are selected (with the exception of the data set with the least number of variables). Therefore, given the results for filtering by variance it seems to make a big difference whether the variables are selected in a supervised or unsupervised fashion, even when the number of selected variables is large. The differences in $\text{CVIIM}_{\boldsymbol{s},n,K}$ for the selection of 5, 10 and 20 variables are not large. Nevertheless the means over the data sets given in Web Table 1 indicate that $\text{CVIIM}_{P,n,K}$ tends to decrease with an increasing number of selected variables. The data set `SLETranscr` stands out through its noticeably larger $\text{CVIIM}_{\boldsymbol{s},n,K}$-values in all plots referring to supervised variable selection. This data set comprises only 36 observations but 47,231 variables (see Table 1), which may at least partly explain the larger values. Extreme values above 0.9, however, are surprising. This example nicely illustrates the utility of our measure as a quantitative tool to investigate the effect of

CV incompleteness.

In the case of optimization of tuning parameters none of the methods exhibit large CVIIM$_{\boldsymbol{s},n,K}$-values (see Figure 2 and Web Table 2). According to our rule of thumb only one result is classified as a medium effect—tuning of the number of components in PLS-LDA for data set `HeadNeckcTranscr`—and the rest are classified as weak effects. Although the effects are weak in most cases, small differences between the methods can be identified by considering the means over the data sets in Web Table 2. PLS-LDA exhibits the biggest means, which is however mainly due to the high values for data set `HeadNeckcTranscr`. With the exception of `LungcTranscr` the training of m_{stop} in boosting has a relatively high impact on the considered data sets compared to other methods. The same holds for training the shrinkage intensity in Nearest Shrunken Centroids, with the exception of the data set `HeadNeckcTranscr` which yields zero-values for CVIIM$_{\boldsymbol{s},n,K}$. `HeadNeckcTranscr` is also an exception in the case of tuning *mtry* in Random Forests, where it yields low values, while we observe near-zero or exactly zero-values for all other data sets. Note, however, that the above results may depend highly on the considered parameter grids, although we made an effort to choose reasonable grids.

In the case of imputation of missing values the results are encouraging overall (upper left panel of Figure 3 and Web Table 3) in the sense that it does not seem to make a big difference whether the considered imputation procedure is trained on the whole data set or based on the training data sets only. The `GenitInfCow` data sets contain proportions of missing values between ~ 8 % and ~ 19 % with tendentially lower proportions for more advanced weeks. This pattern is also reflected by the CVIIM$_{\boldsymbol{s},n,K}$-values, where we observe decreasing values for more advanced weeks, with the highest values being observed for data set `GenitInfCoww0`, the one with the highest amount of missing values. The high-dimensional data set `ProstatecMethyl` yields CVIIM$_{\boldsymbol{s},n,K}$-values of zero for all K-values. In this data set only $\sim 3\%$ of values were missing, which is—although small compared to the `GenitInfCow` data sets—a proportion within the range of proportions likely to occur in practice.

For quantile normalization, with the exception of $K = 3$, we observe zero-values for $\text{CVIIM}_{\boldsymbol{s},n,K}$ on both considered data sets (upper right panel of Figure 3 and Web Table 3). In view of the high computational burden associated with quantile normalization this is an important result. The slightly positive values for $K = 3$ may well be explained by the smaller training sets leading to a higher variance in the normalization procedure, which is related to the "inherent bias" of CV outlined in the methods section; see the discussion.

The picture is almost the same with batch effect removal using ComBat (bottom left panel of Figure 3 and Web Table 3). We observe values of almost or exactly zero in all cases except for $K = 3$ and data set `BreastCancerConcatenation`. The small positive value in the latter case should not be overinterpreted, since it might also be related to the inherent bias.

5 Discussion

In this paper we addressed a ubiquitous topic that had surprisingly not been examined in a systematic way, to our knowledge: the consequence of the violation of the separation between training and test sets during data preparation prior to the training of prediction methods. On one hand the necessary separation of training and test data is a widely acknowledged necessity for the assessment of prediction rules in the context of supervised learning. On the other hand almost all predictive modeling analyses involve some kind of data preparation. But poor attention is given in the literature to the inter-connection between these issues.

We suggested a systematic examination of this problem from a quantitative perspective through estimations of our novel measure of CV incompleteness. In our analyses of real data sets we observed that the impact of such an incomplete CV can be very different depending on the considered step. For data preparation steps that take the response variable into account—supervised variable selection and tuning—we observed distinctively higher $\text{CVIIM}_{\boldsymbol{s},n,K}$-values than for those steps where it is not explicitly used—unsupervised variable selection, imputation, normalization and batch effect

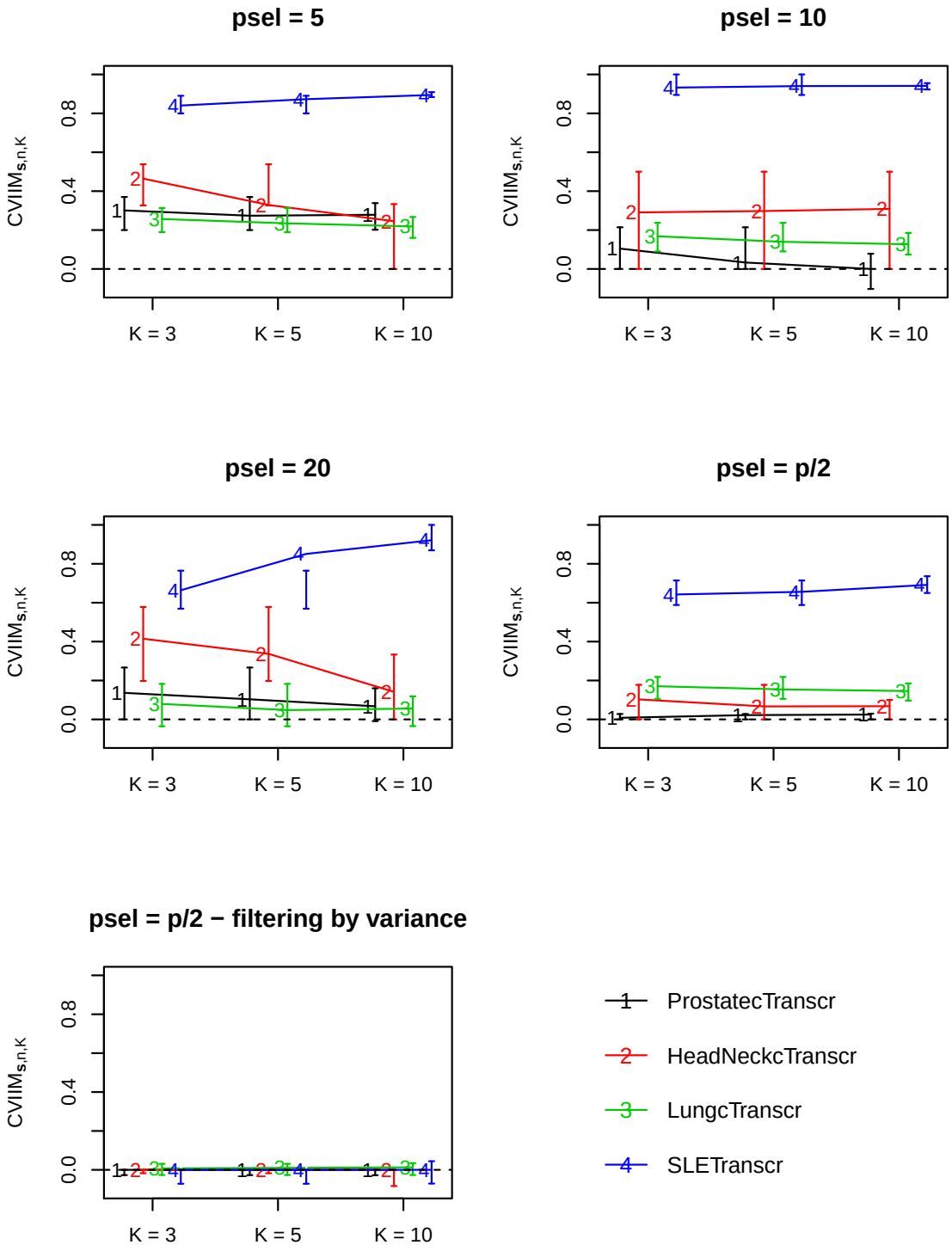

Figure 1: CVIIM$_{\boldsymbol{s},n,K}$-values for variable selection. The different numbers distinguish the data sets.

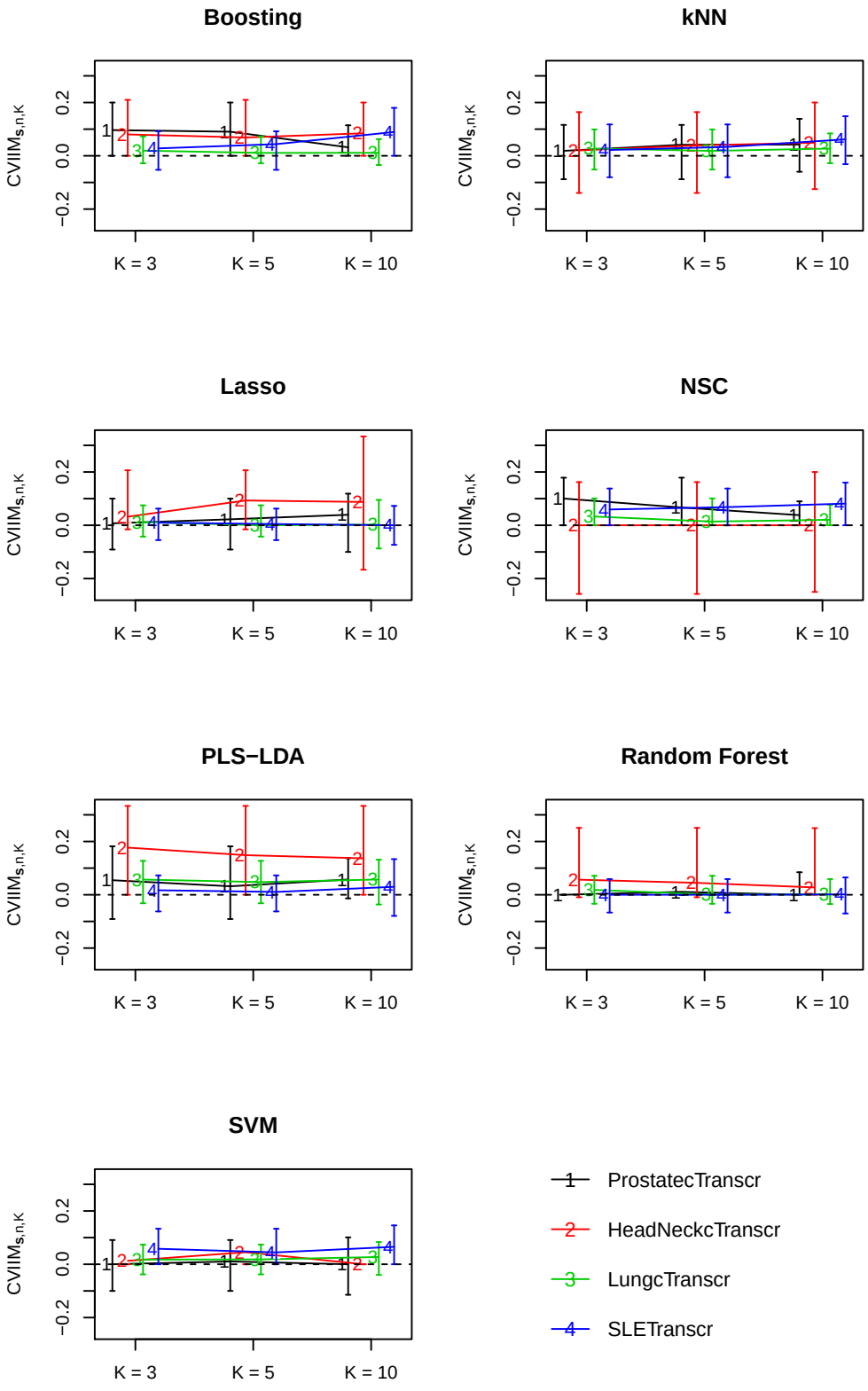

Figure 2: CVIIM$_{\boldsymbol{s},n,K}$-values for tuning. The different numbers distinguish the data sets.

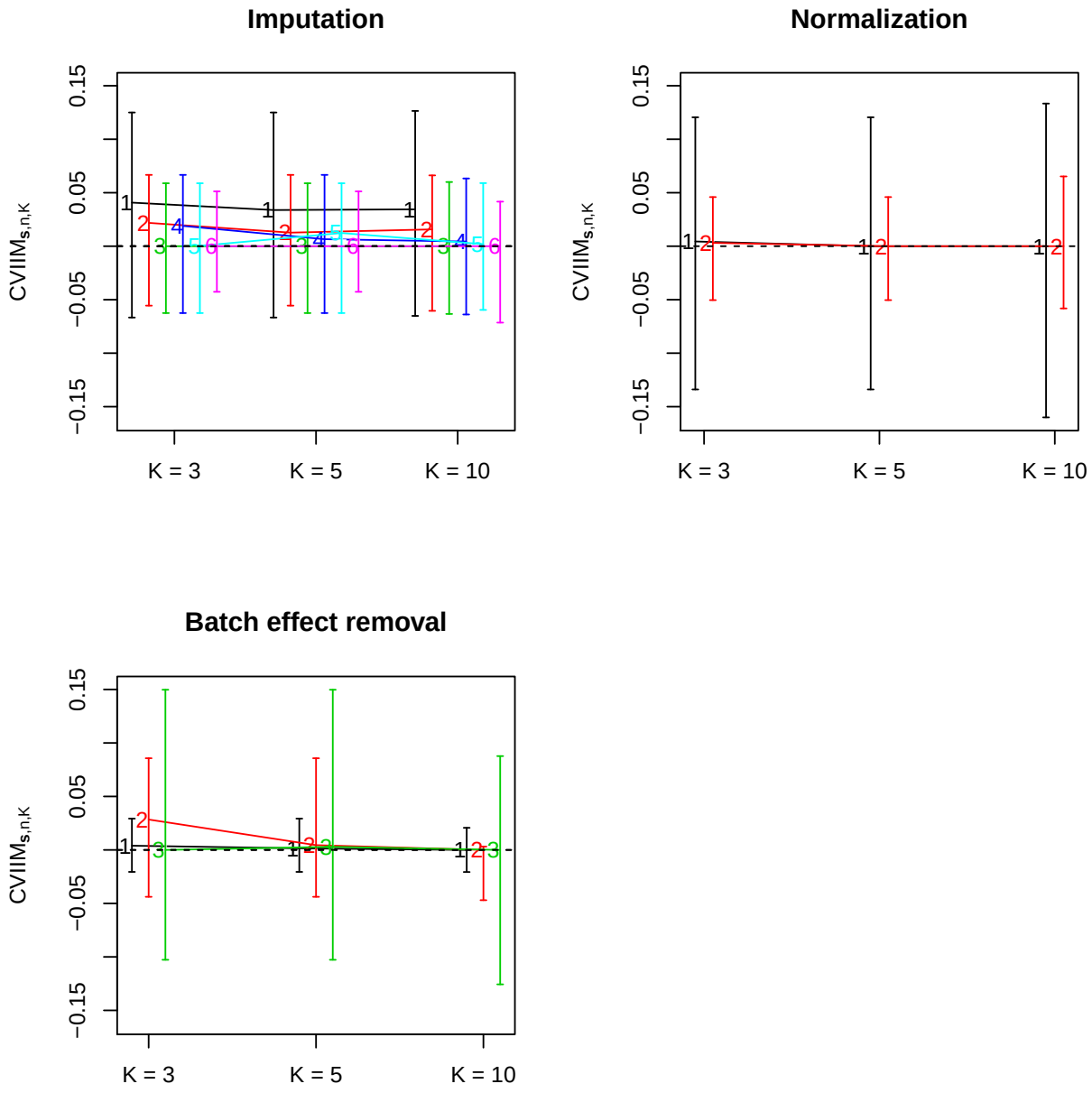

Figure 3: CVIIM$_{s,n,K}$-values for imputation, normalization and batch effect removal. The different numbers distinguish the data sets: Imputation: `GenitInfCoww0` (**-1-**), `GenitInfCoww1` (**-2-**), `GenitInfCoww2` (**-3-**), `GenitInfCoww3` (**-4-**), `GenitInfCoww4` (**-5-**), `ProstatecMethyl` (**-6-**); Normalization: `ColoncTranscr` (**-1-**), `WilmsTumorTranscr` (**-2-**); Batch effect removal: `HCCProteom` (**-1-**), `BreastCancerConcatenation` (**-2-**), `SarcoidosisTranscr` (**-3-**)

removal. For the latter data preparation steps the values were exactly or almost zero. On one hand, it is not surprising that incomplete CV has a large impact on the estimated error for data preparation steps taking the response variable into account. While aiming to increase the association between predictors and response variable, these steps also catch spurious associations – a mechanism known as overfitting –, thereby bringing the estimated model too close to the values of the response variable in the test sample when performed using both training and test data sets. On the other hand, data preparation steps that do not take the response variable into account nevertheless make the values of the predictors overly homogeneous across the observations. If performed using both training and test sets, these steps might thus also result in a seemingly better prediction performance on the test set than if performed using the training set only. However, it was not the case in our examples, where this effect is either nonexistent or small enough to be neutralized by the additional inherent upward bias of $e_{full,K}(\boldsymbol{s})$ compared to $e_{incompl,K}(\boldsymbol{s})$; see below for more details.

Note, however, that we do not claim that a high $\text{CVIIM}_{P,n,K}$-value necessarily corresponds to a step that takes the response variable into account. Some steps involving the response variable might have a negligible impact on error estimation. Likewise, steps that do not involve the response variable may have a higher impact than those investigated in our paper. The aim of this paper was neither to provide an answer for all possible steps nor to make general statements. Instead, we point out that the answer highly depends on the considered step and we recommend using our new general measure to investigate this issue in various settings.

In practice data preparation most often consists of a combination of several analysis steps. In many cases there is a natural ordering of the individual analysis steps towards obtaining a prediction rule. For example, normalization of microarray data has to be performed before variable selection. There are, however, also cases with no clear-cut ordering: for example, dichotomization might be conducted before or after variable selection. Given a specific ordering of the steps, if we include one step in the CV, for obvious technical reasons we also have to include all following steps in the CV. Of course it is also possible to compute $\text{CVIIM}_{\boldsymbol{s},n,K}$ globally for the whole

combination of steps. In Web Appendix C we consider two examples of combinations. In both cases a single analysis step was mainly responsible for the difference between $e_{full,K}(\boldsymbol{s})$ and $e_{incompl,K}(\boldsymbol{s})$.

As outlined in Section 2.1, the training sets are by definition smaller than the whole data set and the CV error estimate is thus an upwardly biased estimator of the error of the prediction rule fitted on the whole data set. This type of bias also affects the relationship between $\varepsilon_{full}(n_{train,K})$ and $\varepsilon_{incompl}(n_{train,K}; n)$. Since in $\varepsilon_{incompl}(n_{train,K}; n)$ the considered analysis step(s) is/are performed on the whole data set, the corresponding parameters are estimated more accurately than in $\varepsilon_{full}(n_{train,K})$ due to the difference in sample sizes. This leads to an additional upward bias of $e_{full,K}(\boldsymbol{s})$ compared to $e_{incompl,K}(\boldsymbol{s})$ with respect to the unconditional error when training using n samples, $\varepsilon(n)$. It may misleadingly result in slightly increased $\text{CVIIM}_{\boldsymbol{s},n,K}$-values in cases where incompleteness has in fact a poor impact on error estimation. Here the additional upward inherent bias of $e_{full,K}(\boldsymbol{s})$ might even be bigger in absolute terms than the downward incompleteness bias of $e_{incompl,K}(\boldsymbol{s})$.

If this issue had a substantial impact in our analyses, $\text{CVIIM}_{\boldsymbol{s},n,K}$ would tend to decrease with an increasing number K of CV folds—because for increasing K the size of the training sets gets closer to the full sample size, thereby diminishing the additional upward inherent bias of $e_{full,K}(\boldsymbol{s})$. This was not the case in our study: K had no clear-cut influence on the results, indicating that this problem does not play a major role in the investigated context.

In this paper our interest was strongly focused on illustrative real-data analyses: the goal of CVIIM is the study of the impact of incomplete CV with respect to specific steps when applied to biomedical data sets. The behavior of the estimator $\text{CVIIM}_{\boldsymbol{s},n,K}$ in practice will highly depend on the considered analysis step and on the specific properties of the data type(s) the step is commonly applied to. Nevertheless to investigate some of its general basic properties we conducted a simulation study for the case of supervised variable selection—that yielded the largest $\text{CVIIM}_{\boldsymbol{s},n,K}$-values in the real-data analyses. The simulation design and detailed results are pre-

sented in Web Appendix D. The data-driven simulation design is based on the `ProstatecTranscr` data set and involves 2000 correlated normally distributed predictors. In these simulations the variance of $\text{CVIIM}_{\boldsymbol{s},n,K}$ as an estimator of $\text{CVIIM}_{P,n,K}$ was relatively high and decreased with decreasing $\text{CVIIM}_{P,n,K}$-values. The bias was negligible. Note that, independently of this variance, we recommend estimating CVIIM for several data sets anyway before formulating general statements on the impact of a specific analysis step. CVIIM cannot be expected to have the same value for all data sets. In the same way as meta-analysts assume that the investigated effects vary over the considered studies, we have to assume that CVIIM will vary depending on the distribution underlying each individual data set. By computing CVIIM for a couple of data sets, we address both the variability of CVIIM within a given distribution (that is observed in our simulation study) and the variability over the data sets.

When displaying the $\text{CVIIM}_{\boldsymbol{s},n,K}$-values graphically in Section 4 we added error bars representing the variability of the (untruncated) $\text{CVIIM}_{P,n,K}$-estimates from individual repetitions of CV. Our underlying assumption that this variability measure also reflects the actual variance of $\text{CVIIM}_{\boldsymbol{s},n,K}$ was confirmed by the simulation, whereby this similarity in behavior was most pronounced for $K = 3$. Considering that the true measure $\text{CVIIM}_{P,n,K}$ was almost equal for different K-values and that the variance of $\text{CVIIM}_{\boldsymbol{s},n,K}$ was as expected smallest for $K = 3$, the simulation indicates that the choice $K = 3$ may be appropriate.

Since our analyses were restricted to binary classification problems, CVIIM was defined based on the misclassification errors corresponding to the indicator loss function. However, the concept of the ratio of the errors is directly transferable to the use of any other loss function with positive range. Most common loss functions fulfill this requirement, for example the quadratic or absolute loss for linear regression, the integrated Brier score for survival data, the check function in the case of quantile regression or the negative log-likelihood as an alternative to the error rate when the response variable is discrete.

In high-dimensional settings, prediction models are often assessed by CV

or related methods, in contrast to low-dimensional settings where parametric models and likelihood-based methods offer the analyst a variety of model checking instruments. Such instruments are usually not applicable to prediction rules trained on complex high-dimensional data, since—as demonstrated in this paper—the derivation of such prediction rules usually involves a succession of potentially complex data preparation steps. The importance of CV in this context makes it especially important to have reliable guidelines for applying these methods in practice. With the example of high-dimensional gene expression data in mind, we concentrated on a selection of important analysis steps for this data type, but such issues essentially arise as soon as CV is applied in any context, as also demonstrated in our paper through applications to other types of data. Most importantly, it is to be expected that ever-changing biotechnologies will constantly produce new types of data requiring special data preparation steps. Our new simple measure will be useful in determining the impact of CV incompleteness for these procedures.

Acknowledgements

The authors thank Rory Wilson for making language corrections and giving valuable input as well as the CIRCE group for providing the data set `HCCProteom` and Michael Schmaußer for providing the `GenitInfCow` data sets.

Supplementary Materials

Web Appendices referenced in Sections 3, 4 and 5 are available with this paper at the Biometrics website on Wiley Online Library. We strive to make all our analyses reproducible: R-Codes implementing our analyses together with data sets in the form of Rda-files can be downloaded from the companion website (URL: `http://www.ibe.med.uni-muenchen.de/organisation/mitarbeiter/070_drittmittel/hornung/cviim_suppfiles/index.html`).

References

Ambroise, C. and McLachlan, G. J. (2002). Selection bias in gene extraction on the basis of microarray gene-expression data. *Proc. Nat. Acad. Sci.*, 99:6562–6566.

Bernau, C., Augustin, T., and Boulesteix, A.-L. (2013). Correcting the optimal resampling-based error rate by estimating the error rate of wrapper algorithms. *Biometrics*, 69:693–702.

Boulesteix, A.-L. and Strimmer, K. (2007). Partial least squares: A versatile tool for the analysis of high-dimensional genomic data. *Briefings in Bioinformatics*, 8:32–44.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Bühlmann, P. and Yu, B. (2003). Boosting with the l2-loss: Regression and classification. *Journal of the American Statistical Association*, 98:324–339.

Edgar, R., Domrachev, M., and Lash, A. E. (2002). Gene expression omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Research*, 30(1):207–210.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer, New York.

Irizarry, R. A., Hobbs, B., Collin, F., Beazer-Barclay, Y. D., Antonellis, K. J., Scherf, U., and Speed, T. P. (2003). Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4:249–264.

Johnson, W. E., Rabinovic, A., and Li, C. (2007). Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics*, 8:118–127.

Kostka, D. and Spang, R. (2008). Microarray based diagnosis profits from better documentation of gene expression signatures. *PLoS Computational Biology*, 4(2):e22.

Leek, J. T., Johnson, W. E., Parker, H. S., Jaffe, A. E., and Storey, J. D. (2012). The sva package for removing batch effects and other unwanted variation in high-throughput experiments. *Bioinformatics*, 28:882–883.

Leek, J. T., Scharpf, R. B., Bravo, H. C., Simcha, D., Langmead, B., Johnson, W. E., Geman, D., Baggerly, K., and Irizarry, R. A. (2010). Tackling the widespread and critical impact of batch effects in high-throughput data. *Nature Review Genetics*, 11:733–739.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, UK.

Rustici, G., Kolesnikov, N., Brandizi, M., Burdett, T., Dylag, M., Emam, I., et al. (2013). Arrayexpress update – trends in database growth and links to data analysis tools. *Nucleid Acid Research*, 41:987–990.

Schölkopf, B. and Smola, A. J. (2002). *Learning with Kernels*. MIT Press, Cambridge, MA.

Simon, R., Radmacher, M. D., Dobbin, K., and McShane, L. M. (2003). Pitfalls in the use of dna microarray data for diagnostic and prognostic classification. *Journal of the National Cancer Institute*, 95(1):14–18.

Tibshirani, R., Hastie, T., Narasimhan, B., and Chu, G. (2002). Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc. Nat. Acad. Sci.*, 99:6567–6572.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. (2001). Missing value estimation methods for dna microarrays. *Bioinformatics*, 17:520–525.

Varma, S. and Simon, R. (2006). Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, 7:91.

Wong, J. (2013). *imputation*. R package version 2.0.1.

Young-Park, M. and Hastie, T. (2007). L1-regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society B*, 69(4):659–577.