Journal of Classification (2021) 38:212–231
https://doi.org/10.1007/s00357-020-09368-z

Improved Outcome Prediction Across Data Sources Through Robust Parameter Tuning

Nicole Ellenbach[1,2] · Anne-Laure Boulesteix[1] · Bernd Bischl[3] · Kristian Unger[2,4] · Roman Hornung[1]

Published online: 6 July 2020

Abstract

In many application areas, prediction rules trained based on high-dimensional data are subsequently applied to make predictions for observations from other sources, but they do not always perform well in this setting. This is because data sets from different sources can feature (slightly) differing distributions, even if they come from similar populations. In the context of high-dimensional data and beyond, most prediction methods involve one or several tuning parameters. Their values are commonly chosen by maximizing the cross-validated prediction performance on the training data. This procedure, however, implicitly presumes that the data to which the prediction rule will be ultimately applied, follow the same distribution as the training data. If this is not the case, less complex prediction rules that slightly underfit the training data may be preferable. Indeed, a tuning parameter does not only control the degree of adjustment of a prediction rule to the training data, but also, more generally, the degree of adjustment to the *distribution of* the training data. On the basis of this idea, in this paper we compare various approaches including new procedures for choosing tuning parameter values that lead to better generalizing prediction rules than those obtained based on cross-validation. Most of these approaches use an external validation data set. In our extensive comparison study based on a large collection of 15 transcriptomic data sets, tuning on external data and robust tuning with a tuned robustness parameter are the two approaches leading to better generalizing prediction rules.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s00357-020-09368-z) contains supplementary material, which is available to authorized users.

✉ Nicole Ellenbach
nellenbach@ibe.med.uni-muenchen.de

1 Institute for Medical Information Processing, Biometry and Epidemiology, University of Munich, Munich, Germany

2 Research Unit Radiation Cytogenetics, Helmholtz Zentrum Munich, German Research Center for Environmental Health GmbH, Neuherberg, Germany

3 Department of Statistics, University of Munich, Munich, Germany

4 Department of Radiation Oncology, University Hospital, University of Munich, Munich, Germany

Keywords Prediction · Robust modeling · Tuning parameter value optimization · Batch effects

1 Introduction

In the context of high-dimensional data and beyond, most prediction methods feature one or several so-called tuning parameters. As opposed to parameters that are fitted during the training process (such as regression coefficients), tuning parameters have to be chosen beforehand. Their values affect the performances of the resulting prediction rules, often to a substantial extent. That is why it is crucial that these values are well chosen. Usually, the values of the tuning parameters are chosen as those that maximize the performance of the prediction rule as estimated through cross-validation (CV) within the training data. This approach is denoted as internal tuning in the following.

Internal tuning does not take into account data from independent sources, denoted as external data from now on, for evaluating the prediction rules. Its principle is based on the implicit assumption that the data to which the prediction rule is intended to be ultimately applied (the test data) follow the same distribution as the training data. If this is not the case (as common if the test data are external), the chosen tuning parameter value may not be optimal. That is because tuning parameters do not only control the degree of adjustment of the prediction rule to the specific observations in the training data, but also the degree of adjustment to the specific *distribution* of the training data (Hornung 2016). While internal tuning prevents over-fitting to the specific observations of the training data, it does not prevent over-fitting to the specific *distribution* of the training data.

In many applications the test data do not follow the (exact) same distribution as the training data. For example, in the special case of phenotype prediction using high-dimensional biomolecular data considered here, there are frequently strong so-called batch effects. These effects are systematic differences between measurements from different studies which are performed in different laboratories, by different staff, with different instrument types, etc. (Scherer 2009). Batches can generally be defined as groups of observations that show differences in measured data that are not caused by the biological signal of interest. Beyond high-dimensional biomolecular data, also clinical prediction rules often perform considerably worse when applied to external data, which is why it is essential that such rules are also assessed using external validation on independent validation data sets (Collins et al. 2014; Bleeker et al. 2003; Siontis et al. 2015). Following the terminology of Bernau et al. (2014), in the following we use the term "cross-study prediction" to describe any situation in which the test data does not have the same distribution as the training data.

As noted above, in such situations internal tuning may be a suboptimal procedure for obtaining suitable tuning parameter values. In this paper we consider alternatives to internal tuning that address this shortcoming. We conjecture that underfitting the training data may lead to a stronger prediction performance for data from external sources on average. A prediction rule that underfits the training data takes into account only strong dependency structures between the outcome and the covariates. Strong dependency structures are more likely to be universal, that is, present in any data source relevant to the prediction problem of interest. By contrast, weaker dependency structures are more likely to be specific to the considered training data. For example, in the context of gene expression analysis, there may be factors with subtle systematic influences on the molecular phenotype that are specific to the subjects included in the training data, such as differences in age, ethnicity, or hospital-/culture-specific delivery of therapeutic treatments. Such associations, specific to

the training data, should ideally not be captured by the prediction rule as they tend to partly mask the actually relevant, universal association patterns.

In this spirit, Zhang et al. (2020) demonstrated in a recent extensive empirical study that in biomolecular data analysis the dependency structures between covariates and outcome differ frequently across sources. They found this phenomenon to be the most important reason for the worse results obtained in cross-study prediction compared with the prediction of the outcome of observations from the same source. Recently, Dondelinger et al. (2020) presented the joint lasso, a regression approach for high-dimensional covariate data that allows for differing dependency structures between covariates and outcome across different groups of observations.

If we had a large number of data sets from different sources available, an ideal tuning strategy would be to choose that tuning parameter value that leads to an optimal mean performance over these sources. Provided that the number of data sets considered in this approach is large, the chosen tuning parameter value would subsequently also be associated with an optimal mean performance on independent, future external data different from that used for tuning. Unfortunately, this approach is almost never applicable, because in practice there are usually not many data sets available for the same prediction problem. For this reason, we will not investigate this approach in this paper. However, *one* external data set is often available for testing purposes. This external data set might in principle be used to choose a tuning parameter value suitable for cross-study prediction by maximizing the prediction performance on this data set. We will consider this approach in this paper in different variations.

The idea of optimizing tuning parameter values using external data has to our knowledge been previously considered merely in a single publication: Rohart et al. (2017) present a penalized variant of Partial Least Squares Discriminant Analysis for multiple training data sets, where for tuning parameter value optimization a procedure referred to as "Leave-One-Group-Out Cross-Validation" is used. It is analogous to CV, with the difference that the concatenation of training data sets is not randomly split into folds, but instead the folds correspond to the different data sources. This procedure reflects the intended use of the prediction rule, that is, applying it to independent external data. Rohart et al. (2017) introduced this approach in the context of their new prediction method for the case of training data sets consisting of several data sources. In contrast, the approaches for tuning parameter optimization introduced in the present paper are applicable to prediction methods for single training data sets that feature tuning parameters.

Note that after using the external data set for tuning parameter value optimization, it cannot be used for assessing the performance of the optimally selected prediction rule anymore. Such an assessment would be over-optimistic (see Hornung et al. 2015 for a discussion of the consequences of partly taking the test data into account while training or selecting prediction rules). To estimate the prediction performance unbiasedly, we need a second external data set that has not been involved in constructing the prediction rule. However, in many applications two external data sets may not be available. In Section 3.4 we will present a pragmatic procedure for estimating the prediction performance based on the external data set in a biased but conservative way.

A strategy that does not in its general form require a second external data set is the following. First, optimize the tuning parameter using internal tuning and, second, modify the optimized tuning parameter value by a reasonable degree in order to obtain a less complex prediction rule. This idea is already realized for Lasso and Ridge in the R package `glmnet` (Friedman et al. 2010). It allows the use of a stronger penalizing λ value `lambda.1se` than the one optimized using internal tuning (see Section 2.5.2 for details). In this paper, we gene-

ralize this approach to other prediction methods and suggest improvements. In summary, all approaches to parameter tuning considered in this paper for obtaining robust prediction rules are based on such a modification of the results of internal tuning (denoted as "robust tuning") and/or on the use of external data sets (denoted as "external tuning"). The methods are compared using 15 real gene expression data sets.

Throughout our analyses we use simple grid search to optimize the tuning parameter values; that is, we restrict the search for suitable values of the tuning parameters to sets of values given through prespecified grids. This strategy is valid, as it can be expected to deliver optimized tuning parameter values that are close to the actual optimal values. However, there exist many, more sophisticated approaches to tuning parameter optimization that approximate the optimal tuning parameter values more effectively. Examples of modern tuning parameter optimization approaches include: Bayesian optimization (Snoek et al. 2012; Bischl et al. 2017), gradient descent search (Chapelle et al. 2002), and swarm algorithms (Lin et al. 2008) (see Claesen and De Moor 2015 for an overview). The procedures presented in this paper may be used in combination with these approaches.

The paper is structured as follows. In Section 2 we describe the data used in our extensive comparison studies as well as the study designs and provide detailed outlines of each of the approaches considered in these studies. Subsequently, we present and interpret the results of our studies in Section 3. Section 4 concludes the paper with a discussion of heterogeneity between data sources in the context of biomedical data and a set of recommendations derived from our study.

2 Methods

2.1 Data

We use 15 publicly available microarray data sets of chip type HG-U133PLUS2 with different numbers of observations. To ensure that the numbers of observations in the data sets are not too small for prediction modeling, these 15 data sets are the subset of the 25 data sets studied in Hornung et al. (2017) that feature at least 50 observations. They are independent of each other and contain the same gene expression variables as covariates and the same dependent variable, "gender." Each gene expression variable provides the gene expression value of a specific gene for the considered patient. Gene expression values are metric scores, the levels of which indicate, how active the respective genes are for the different patients. For some genes, these levels of activation have an important influence on the outcome of prediction applications. We do not use all 54675 gene expression variables, but merely a random subset of 2500 variables (the same for all 15 data sets). This subsetting of the covariate space is performed in order to reduce the overly strong biological signal contained in the gene expression data for explaining the dependent variable "gender" and thus make it comparable to signals observed in applications of clinical relevance. While "gender" is not a clinically meaningful outcome in biomedical applications, it features major advantages for a purely methodological investigation on cross-study prediction such as the one performed in this paper (see Hornung et al. 2017 for a thorough justification of this choice).

2.2 Prediction Methods Considered in Comparison Study

We include five different prediction methods for binary dependent variables in our studies: Lasso regression (Tibshirani 1996), Ridge regression (Hoerl and Kennard 1970),

component-wise boosting (Buehlmann and Yu 2003), Support Vector Machine (Cortes and Vapnik 1995), and Random Forest (Breiman 2001) (see Table 1 for details on how each method is used, in particular with respect to the involved tuning parameters). Since centering and standardization was found to affect cross-study performance in a previous study (Hornung et al. 2017), we initially considered two variants for each of the methods Lasso, Ridge, component-wise boosting, and Support Vector Machine. For Lasso, Ridge, and Support Vector Machine, in the first variant (suffix "_unstand") the variables were left as they are and in the second variant they were standardized to have mean zero and variance one before fitting ("_stand"). Similarly, for component-wise boosting in the first variant the variables were again left as they are ("_uncenter") and in the second variant they were centered ("_center") to have mean zero. When standardizing respectively centering was applied in the training data, the same transformation was performed for the test data using the variances respectively means of the variables estimated from the training data. With the exception of component-wise boosting we, however, did not observe any relevant differences between the two variants with respect to the performances of the compared approaches relative to each other. Therefore, in order to make the presentation of the results more clear, we will provide the results obtained for the unstandardized versions of Lasso, Ridge, and Support Vector Machine only in the supplement (Online Resource 1). For the sake of simplicity, we will, moreover, leave out the suffix "_stand" in the designations of these methods in the following. In the case of Random Forest the variables were left unstandardized.

2.3 General Design of the Comparison Study

Before detailing each considered tuning approach, we first describe the general design of our study. Most of the considered approaches use two data sets A and B. For example, for one of the approaches ("`Ext`"), training is conducted on data set A, while data set B is used for external tuning. We consider all $\binom{15}{2}2 = 210$ possible combinations of two data sets out of all available 15 data sets. The prediction performance of the resulting prediction rules is measured by the area under the receiver operating characteristic curve (AUC) estimated based on a large independent test data set, which is obtained by combining the data of all $13 = 15 - 2$ remaining data sets. For better comparability, this procedure is also adopted for internal tuning, even though for this approach the second data set is not used.

The optimal tuning parameters are always chosen from a grid of values (see Table 1). We always choose the tuning parameter value from the respective grid that delivers the highest AUC value, where the data used to calculate these AUC values depends on the specific approach considered (see Sections 2.4 and 2.5). If the AUC is maximal for more than one tuning parameter value, we choose among these values the one that leads to the least complex (i.e., most robust) prediction rule (see Table 1 for what "more robust" means for each of the different methods).

In this paper, we exclusively consider binary classification. However, the considered approaches are extendable to other types of dependent variables, such as metric or survival outcome, by considering performance measures other than the AUC and corresponding variants of the respective prediction methods.

2.4 Preliminary Study: Conceptual Comparison of External and Internal Tuning

The comparison study presented in this section aims at conceptually comparing internal and external tuning. In contrast to the rest of the paper, in this preliminary study the term "internal tuning" does not refer to the CV-based tuning parameter optimization procedure,

Table 1 Setups used for prediction methods and respective tuning parameters

Method	Label	Tuning parameter to optimize	Other (tuning) parameters	Software
Lasso	`Lasso`	Shrinkage parameter λ used grid: 200 values as chosen by R package `glmnet` (uses data-driven procedure to determine the range of the grid), sequence determined on whole training data set also when using internal tuning the larger λ, the less complex/more robust the prediction rule	–	R package `glmnet` (Friedman et al. 2010)
Ridge	`Ridge`	Description identical to that for `Lasso`	–	R package `glmnet`
Component-wise boosting	`Boost`	Number of boosting steps m_{stop} used grid: 5, 10, 15, ..., 5000 the smaller m_{stop}, the less complex/more robust the prediction rule	Step size ν; set to 0.1	R package `mboost` (Hothorn et al. 2018)
Support Vector Machine	`SVM`	Cost parameter C used grid: $2^{-15}, 2^{-14}, \ldots, 2^{15}$ the smaller C, the less complex/more robust the prediction rule	Kernel type; set to "linear"	R packages `mlr` (Bischl et al. 2016) and `e1071` (Meyer et al. 2019)
Random Forest	`RF`	Minimal node size n_{node} used grid: $1, 2, 3, \ldots, n-1$, where n denotes the sample size the larger n_{node}, the less complex/more robust the prediction rule	(1) number of trees $ntree$; set to 500; (2) number of variables sampled per split $mtry$; set to $\sqrt{p}$, where p is the number of variables	R packages `mlr` and `ranger` (Wright and Ziegler 2017)

but merely to the fact that training and tuning are performed on data from the same source. We want to evaluate whether external tuning (i.e., tuning conducted on data from a different source) performs better than such an internal tuning, *the size of the data used for tuning being equal for both procedures*. For this purpose, we artificially make the sizes of the data sets equal for these two approaches by random subsetting.

Consider a data set A of size n_A, used for training, and a data set B of size n_B, used for tuning and denoted as *external data set*. With the aim of conceptually comparing internal and external tuning while eliminating the effect of the size of the data used for tuning, we proceed as follows to obtain subsets of A and B to be used for training, external tuning and internal tuning. If $n_A \leq 3n_B$, we randomly draw a subset of size $2n_A/3$ from data set A, which is used for training. The rest of data set A (of size $n_A/3$) is used for internal tuning. The data to be used for external tuning is obtained by randomly sampling from the data set B a subset of size $n_A/3$. Since this is not possible if $n_A > 3n_B$, in this case the whole data set B is used for external tuning, a random subset from A of size n_B is used for internal tuning, and training is performed using the other $n_A - n_B$ observations from A. In this way, the only difference between internal and external tuning is the source (but not the size) of the data sets used for tuning. In order to decrease the variability of the results, for each pair of training data set and external data set, 10 different random subsets of data used for training and tuning are considered (the same for both tuning approaches). Subsequently, the 10 optimized tuning parameter values and the 10 AUC values obtained on the combined data of the 13 remaining data sets are averaged.

In contrast to the approaches considered in this conceptual study, however, in practice all available data would be exploited as effectively as possible, as described in the next section.

2.5 Main Study: Internal vs. External Tuning and Procedures for Robust Tuning

Figure 1 gives an overview of all (internal, external or robust) tuning approaches considered in our main study and described in this section (see Figure S1 in Section A of Online Resource 1 for an extended version of this overview that includes the approaches referred to in Section 2.5.5). Note that these graphical representations are not self-explanatory, but should help to maintain the overview of the different approaches subsequent to reading the corresponding text passages (even if we will not refer explicitly to this figure again).

2.5.1 `Ext` and `Int`

In contrast to the procedures described in Section 2.4, when applying external tuning in practice, the whole data set A is used for training and the whole external data set B is used for tuning. This procedure is denoted as `Ext`.

As indicated in the introduction, a standard procedure used for internal tuning is CV-based tuning. In our analysis this procedure is performed as follows. We use 5-fold CV. For each CV iteration, four folds are used for training the prediction rule using each tuning parameter value from the considered grid successively and the remaining fold is used to estimate the AUC. If this fifth part only contains either females or males as response, the AUC value is set to missing for this fold. For each considered tuning parameter value, the AUC values are then averaged over the five CV folds. The optimal tuning parameter value is chosen as the one yielding the highest average AUC (see Section 2.3 for details). Finally, the prediction rule is trained on the whole training data set using the chosen tuning parameter value. We denote this procedure as `Int`.

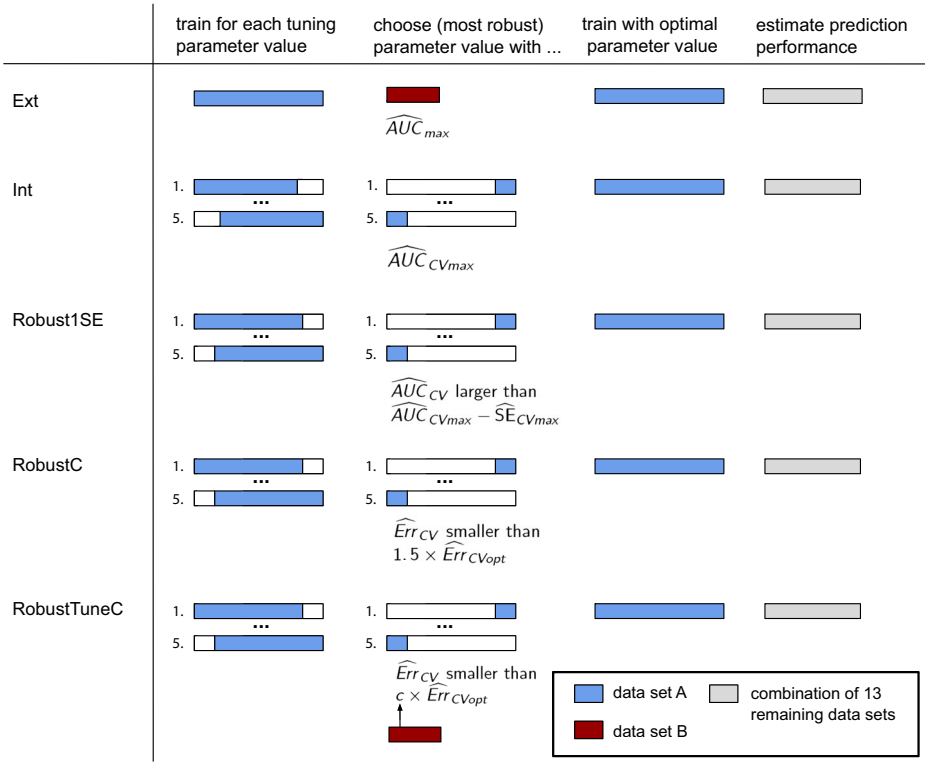

Fig. 1 Overview of the approaches from Section 2.5 for external/ internal tuning and the procedures for robust tuning. See the text for more details on `RobustC` and `RobustTuneC`

2.5.2 `Robust1SE`

In this and in the two next subsections we present approaches denoted as "robust tuning": `Robust1SE`, `RobustC` and `RobustTuneC`. The idea consists of intentionally selecting parameter values that lead to underfitting when performing internal tuning using data set A, with the aim of obtaining prediction rules that generalize better to data from other distributions.

The first variant, `Robust1SE`, is inspired by the R package `glmnet` (Friedman et al. 2010) implementing Lasso and Ridge: the most robust tuning parameter value (in the case of Lasso and Ridge, the largest λ) is chosen "such that [the] error is within 1 standard error of the minimum" (quotation from the `glmnet` manual).

In this vein, our approach `Robust1SE` consists of choosing the tuning parameter value that leads to the least complex prediction rule for which the cross-validated AUC value is larger than . $\widehat{AUC}_{CVmax} - \widehat{SE}_{CVmax}$. Here, $\widehat{AUC}_{CVmax}$ denotes the largest cross-validated AUC value and $\widehat{SE}_{CVmax}$ stands for the quantity that is referred to as "standard error of the [maximum]" in the `glmnet` manual and calculated as

$$\widehat{SE}_{CVmax} = \sqrt{\frac{1}{K(K-1)} \sum_{k=1}^{K} \left(\widehat{AUC}_k - \widehat{AUC}_{CVmax} \right)^2},$$

where K is the number of CV folds and $\widehat{AUC}_k$ denotes the estimate from the kth CV fold obtained with the parameter value that yields $\widehat{AUC}_{CVmax}$. Note that $\widehat{SE}_{CVmax}$ is not intended as an estimator of the standard error of the estimator $\widehat{AUC}_{CVmax}$. This standard error is very difficult to estimate due to the dependencies between CV folds and to the maximization process. Instead, $\widehat{SE}_{CVmax}$ should be seen as a pragmatic measure of the reliability of the performance estimation.

Note that the original main motivation of the implementation of this approach in `glmnet` was likely not cross-study prediction. Instead, its main purpose is to favor sparse models. Further, for large data sets $\widehat{SE}_{CVmax}$ approaches zero and $\widehat{AUC}_{CVmax} - \widehat{SE}_{CVmax}$ becomes thus close to $\widehat{AUC}_{CVmax}$. For this reason `Robust1SE` delivers similar results as `Int` for large data sets. This is problematic in the context of cross-study prediction, because the disparity between the training data set and the data set to which the rule is intended to be applied does not depend on the size of the training data set.

2.5.3 `RobustC`

Motivated by the pitfall of `Robust1SE` outlined above, we suggest an alternative approach for robust tuning, `RobustC`. Similarly to `Robust1SE`, the idea is to choose a parameter value leading to a less complex prediction rule than the value that is optimal in terms of prediction error on observations from the same source (as estimated by CV on data set A). In contrast to `Robust1SE`, however, the sacrifice in terms of prediction error on observations from the same source is independent of the sample size. In this section, we consider $Err = 1 - AUC$ as a measure for prediction error, where perfect and useless prediction rules have values of 0 and 0.5, respectively.

`RobustC` consists of choosing the tuning parameter value leading to the least complex rule with cross-validated $\widehat{Err}$ smaller than $c \times \widehat{Err}_{CVopt}$, where $c \geq 1$ and $\widehat{Err}_{CVopt}$ is the cross-validated error achieved by the optimal parameter value on data set A.

The constant c controls the sacrifice in terms of prediction error one is willing to make for observations from the same distribution as data set A in the hope of obtaining a robust prediction rule that performs well for other distributions. The complexity of the resulting prediction rule decreases with increasing c. If the data to which the prediction rule is intended to be applied follows the same distribution as data set A, the value $c = 1$ is optimal. In contrast, if the distributions are substantially different, prediction rules obtained from data set A with larger values of c are likely to perform better. For the investigation of `RobustC`, we fix c to $c = 1.5$. A more flexible variant that considers c itself as a tuning parameter is described in Section 2.5.4.

Regardless of the choice of c, using $c \times \widehat{Err}_{CVopt}$ as a threshold, as described above, may be suboptimal in the case of a high $\widehat{Err}_{CVopt}$. Unless c is very close to 1, the sacrificed prediction error may then be unacceptable. The resulting prediction rule may indeed not only substantially underfit the distribution of data set A, but perhaps also the distribution(s) of the data to which it is intended to be applied. To address this pitfall, we modify the procedure described above as follows. We choose the tuning parameter value that yields the least complex prediction rule with cross-validated $\widehat{Err}$ smaller than:

$$\begin{cases} c \times \widehat{Err}_{CVopt}, & \text{if } c \times \widehat{Err}_{CVopt} < 0.4, \\ 0.4, & \text{if } c \times \widehat{Err}_{CVopt} \geq 0.4 \\ & \text{and } \widehat{Err}_{CVopt} < 0.4, \\ \widehat{Err}_{CVopt}, & \text{if } \widehat{Err}_{CVopt} \geq 0.4. \end{cases}$$

2.5.4 RobustTuneC

As outlined above, the best value of the parameter c depends, roughly speaking, on the difference between the distributions of the training data set A and the data set to which the prediction rule is intended to be applied. In a variant procedure which we denote as RobustTuneC, we thus suggest considering several candidate values of c successively and finally selecting the one that leads to the best performing rule as assessed based on data set B. Here we assume that external data other than data set B differ to a similar degree from data set A, as does data set B. Note that with this procedure the *best* performance obtained on data set B cannot be used as an estimate of the performance on independent data sets, since it is the result of an optimization (across the candidate values of c). We will suggest a conservative surrogate estimate of this performance when discussing the results in Section 3.4.

We suggest the following sequence to be used for the value of c: 1, 1.1, 1.3, 1.5, 2. By including small c values in this prespecified sequence, we avoid the risk of obtaining an overly robust prediction rule with bad performance. This is because for some data, the prediction rules obtained for high c values will be too robust. In such situations a small c value will be chosen by the procedure, which will result in a merely moderately robust prediction rule or, if the chosen c value is equal to one, in an internally tuned prediction rule.

2.5.5 Outlook: Procedures Using both A and B for Training

When having two data sets A and B available, some practitioners may be inclined to maximize the number of observations used for training by combining both data sets A and B for training. Therefore, in addition to Ext and Int, we considered corresponding procedures that use both data sets for training. However, since including the validation data set for training does not seem to be very common in practice and because this proceeding does not belong to the core topics of this paper, we describe these approaches and the results obtained using them in Online Resource 1 (for the description of these approaches, see Section B of Online Resource 1).

2.6 Additional Study: Optimistic Bias by Using the External Data Set for both Tuning and Prediction Performance Estimation

In practice, there is often only one external data set available in addition to the training data set. In such situations, when applying external tuning it would be practical if the external data set could also be used for estimating the performance of the resulting prediction rule on independent data. However, as mentioned in the introduction, this approach, which we denote as ExtValidNoCV here, yields optimistic estimates. This issue is well-known in the context of parameter tuning with CV and motivates the use of nested CV (Varma and Simon 2006; Hornung et al. 2015). In order to assess the extent and reasons of this overestimation in the different context of *external* tuning, we perform an additional study.

There are two distinct reasons for the optimistic bias of the prediction performance estimate obtained by ExtValidNoCV. The first reason is known from classical tuning with CV. AUC values calculated on the external data set B are not the true AUC values but estimates fluctuating around the corresponding true AUC values. Selecting the maximum observed AUC value will in general overestimate the maximum true AUC value, a problem commonly known as the winner's curse. The second reason for the optimistic bias of the prediction performance estimate obtained by ExtValidNoCV, however, is specific to

external tuning: observations to which the prediction rule is intended to be applied do not follow the same distribution as data set B used for external tuning, a problem which also contributes to the bias of `ExtValidNoCV`.

To assess the contribution of these two sources of bias, we consider a procedure `ExtValidCV` that eliminates the first source of bias and compare it with `ExtValidNoCV`. The idea of `ExtValidCV` is to use different parts of data set B for choosing the tuning parameter value and estimating the prediction performance. To do this, data set B is split randomly into five approximately equally sized folds. For each fold k, the following procedure is repeated: the four other folds are used to choose the tuning parameter value, while fold k is used to estimate the resulting performance. The performance is then averaged over the five folds.

3 Results

All R codes written to produce and evaluate our results are available online with the Electronic Appendix (Online Resource 2).

3.1 Preliminary Study: Conceptual Comparison of External and Internal Tuning

Figure 2 shows the AUC values obtained for external and internal tuning (see Figures S2 and S3 in Section C of Online Resource 1 for the extended results, that is, those including

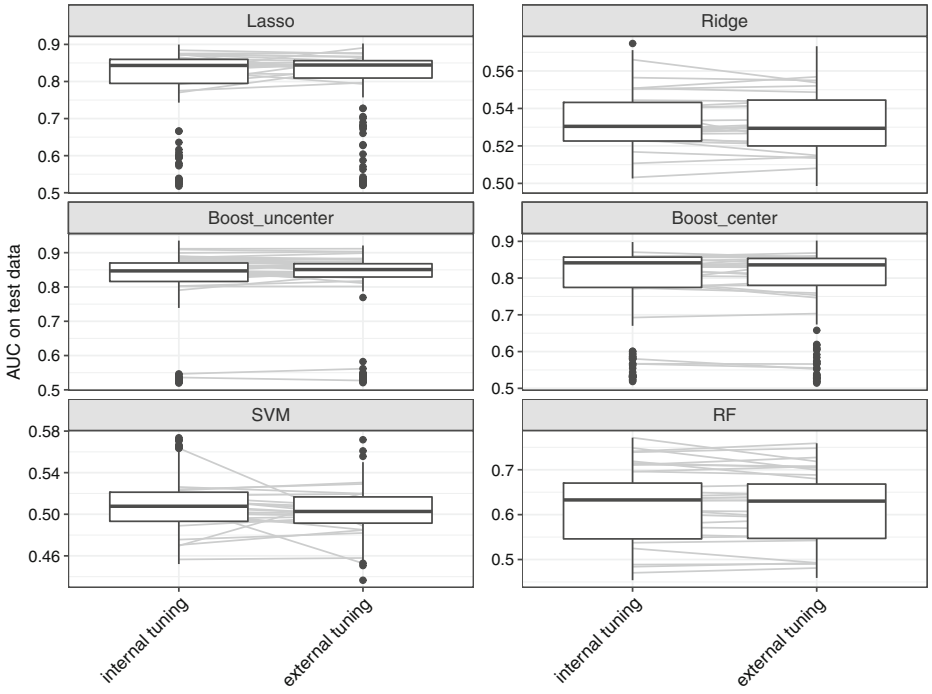

Fig. 2 Preliminary study. Prediction performance estimates based on independent test data (cf. Section 2.4) in the conceptual comparison of external and internal tuning. The gray lines connect the values of pairs that share the same training data sets, where in each case, for the sake of clarity, we do not show a line for each of the pairs, but merely for a random subset of 30 pairs

in addition the unstandardized versions of Lasso, Ridge, and Support Vector Machine). Obviously, the AUC values hardly differ between external and internal tuning for any of the prediction methods considered. Nevertheless, for some prediction methods we do observe differences with respect to the tuning parameter values chosen by the two tuning approaches (Figure 3). In each case, for which we can observe differences between external and internal tuning, the tuning parameter values chosen by external tuning are associated with more robust prediction rules. For example, in the case of `Boost_uncenter`, the values of m_{stop} are smaller for external tuning and smaller values of m_{stop} are associated with more robust boosting models.

The above results suggest that, when comparing external and internal tuning fairly, external tuning can lead to slightly more robust prediction rules, but the differences to internal tuning are not strong enough to manifest themselves in notably differing prediction performances. Note again that for the comparisons presented in this subsection we artificially made the training and tuning data of equal size between external and internal tuning (and thus did not fully exploit the size of the data sets). In the following subsection, we will present comparisons of the practically relevant approaches to external and internal tuning. For these, the whole training data set and the whole external data set are exploited.

3.2 Main Study: Internal vs. External Tuning and Procedures for Robust Tuning

Figure 4 shows the AUC values obtained for the approaches described in Section 2.5 (see Figures S4 and S5 in Section D of Online Resource 1 for the extended results including,

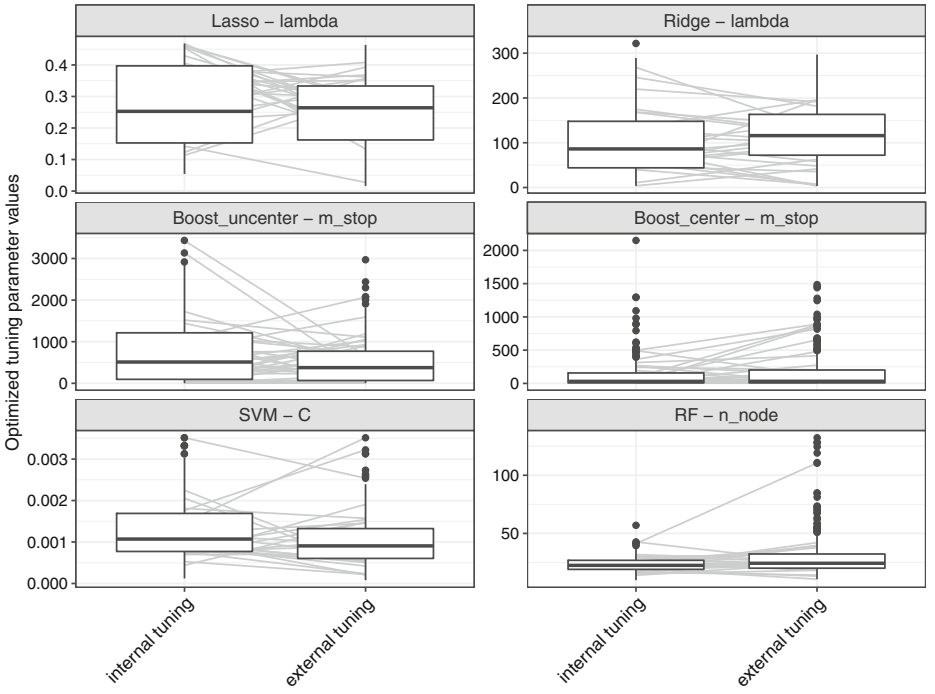

Fig. 3 Preliminary study. Chosen tuning parameter values (cf. Section 2.4) in the conceptual comparison of external and internal tuning. The gray lines connect the values of pairs that share the same training data sets, where in each case, for the sake of clarity, we do not show a line for each of the pairs, but merely for a random subset of 30 pairs

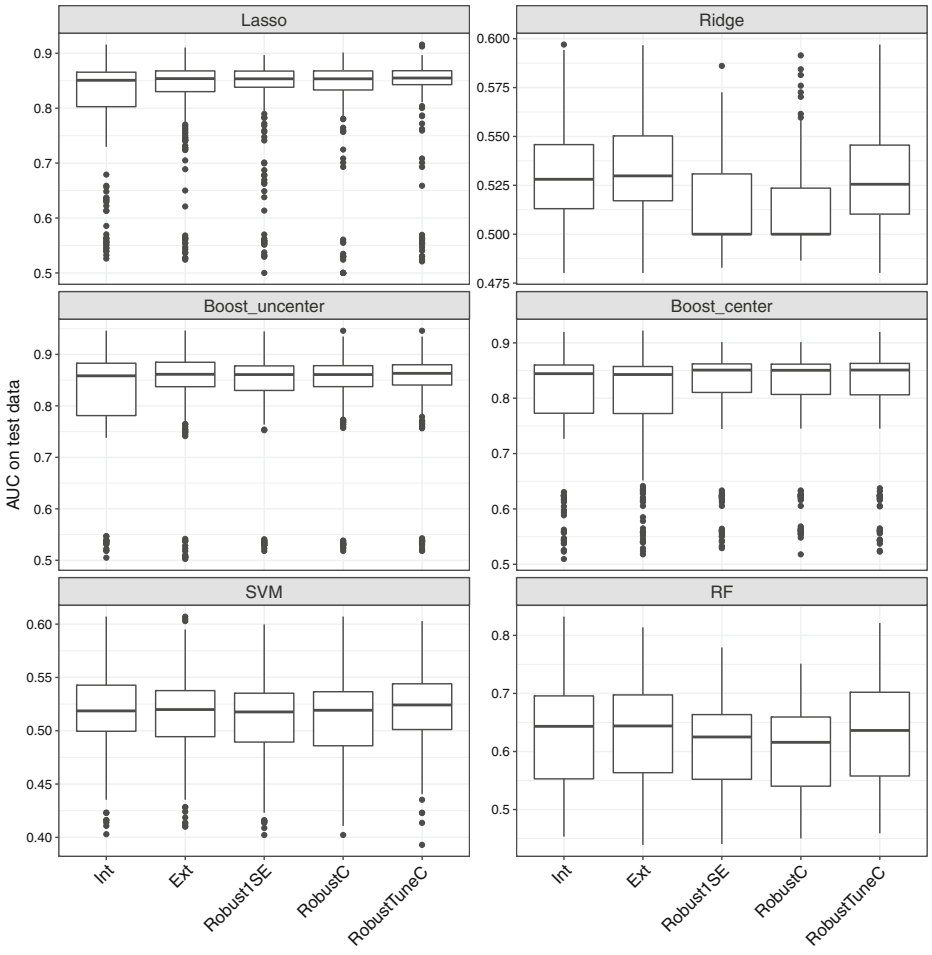

Fig. 4 Main study. Prediction performance estimates based on independent test data (cf. Section 2.3) for various practically motivated approaches to external, internal, and robust tuning

in addition, the approaches mentioned in Section 2.5.5 and the unstandardized versions of Lasso, Ridge, and Support Vector Machine). When comparing `Ext` and `Int`, that is, the two basic practical approaches to external and internal tuning, we observe a clear improvement by external tuning for `Lasso` and `Boost_uncenter`, while there are no noteworthy differences between the two tuning approaches for the other prediction methods. The tuning parameter values chosen using each approach are shown in Figure 5 (see Figures S6 and S7 in Section E of Online Resource 1 for the extended results). For `Lasso` `Ext` delivered slightly smaller λ values than `Int`. However, in the cases of the other prediction methods, either the tuning parameter values do not differ systematically between `Ext` and `Int` or more robust prediction rules are obtained using `Ext`. The observation that `Ext` was associated with smaller optimized λ values than `Int` for `Lasso` is probably strongly related to the combination of two facts: first, the variance of the λ values optimized using `Ext` is smaller and, second, the λ values are bounded by zero. If two random variables X and Y have the same type of probability distribution with support $[0, \infty)$, where X has higher

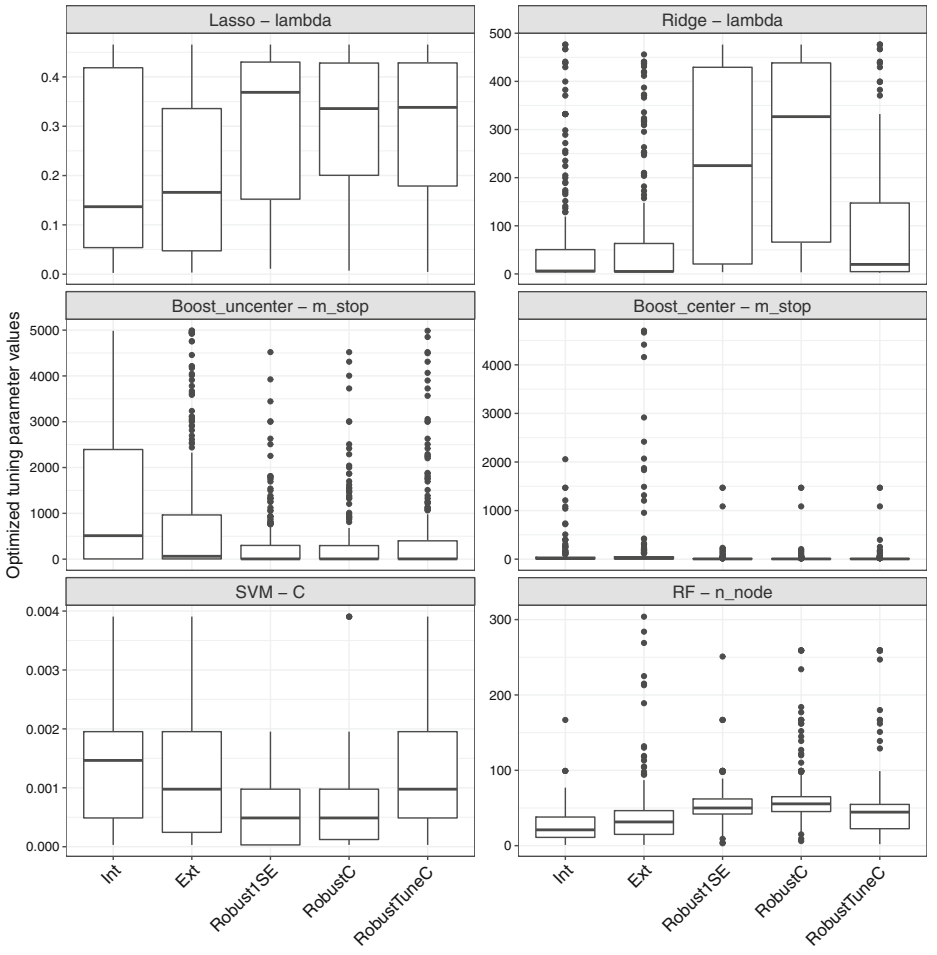

Fig. 5 Main study. Chosen tuning parameter values for various practically motivated approaches to external, internal, and robust tuning

variance than Y, then it is likely that the mean of X is also higher than that of Y. This is because the large variance of X will express itself rather in larger realizations of X than in smaller realizations, because there is less allowance for smaller realizations due to the support being bound by zero. This positive relation between the mean and the variance of probability distributions with support $[0, \infty)$ is well seen in the example of the gamma distribution (and its special cases exponential distribution and chi-square distribution), which has mean $k\theta$ and variance $k\theta^2$.

The two robust tuning approaches, `Robust1SE` and `RobustC`, perform very similarly in all cases. For some prediction methods, these two approaches perform similarly well or even better than `Ext`, while they perform worse for other prediction methods. More precisely, we observe that these two approaches perform badly for those prediction methods for which the cross-study prediction performance is bad in general, namely for `Ridge`, `SVM`, and `RF`. Inspecting the tuning parameter values chosen by `Robust1SE` and `RobustC` for these prediction methods reveals that they differ strongly from those chosen by the other

tuning approaches in the direction of more robust prediction rules. This suggests that for these prediction methods `Robust1SE` and `RobustC` deliver prediction rules that do not use enough information from the training data. This might be explained as follows: if the cross-validated prediction error measure $\widehat{Err}_{CVopt}$ is large, as is the case for prediction methods with bad cross-study prediction performance, then both $\widehat{Err}_{CVopt} + \widehat{SE}_{CVopt}$ and $c \times \widehat{Err}_{CVopt}$ are large as well (the variance of large $\widehat{Err}_{CVopt}$ values is larger).

With respect to the results obtained using `RobustTuneC`, we can make the following two main observations. First, for those prediction methods for which the two robust tuning approaches `Robust1SE` and `RobustC` perform better or equally well as external tuning, `RobustTuneC` also performed well. Second, for those prediction methods for which `Robust1SE` and `RobustC` perform worse than external tuning, `RobustTuneC` performed equally well as external tuning (with one slight exception, `Ridge`, where `RobustTuneC` performed very slightly worse than external tuning). Thus, depending on the specific setting, `RobustTuneC` performed either better or equally well as external tuning, avoiding the disadvantages of `Robust1SE` and `RobustC` of delivering overly robust prediction methods in cases in which the cross-study prediction performance is low in general. The tuning parameter values optimized with `RobustTuneC` (Figure 5) are congruent with the results obtained with respect to the prediction performance of this approach: in the cases of the prediction methods, for which the robust tuning methods (including `RobustTuneC`) performed better than external tuning, the tuning parameter values optimized using `RobustTuneC` are similar to those optimized with `Robust1SE` and `RobustC`. By contrast, for those prediction methods for which `RobustTuneC` and external tuning performed better than `Robust1SE` and `RobustC`, the tuning parameter values optimized using `RobustTuneC` are similar to those optimized with external tuning. Figure S8 (Online Resource 1, Section F) shows the frequencies of selection of the considered candidate values of c: 1, 1.1, 1.3, 1.5, 2. Interestingly, for those prediction methods for which the robust tuning methods performed better than external tuning, the highest value $c = 2$ was chosen in the vast majority of cases. This suggests that the optimal value of c is probably even larger than 2 for these methods. The prespecified sequence for the c values might thus be expanded by one or several values larger than 2. However, note that allowing very large values for c might lead to a bad cross-study prediction performance. This can occur in situations in which the distribution of the external data set used for choosing the c value happens to differ much more strongly from that of the training data set than do the distributions of other external data for which the prediction rule is intended. For the remaining prediction methods, for which the value $c = 2$ was not chosen frequently, in the majority of cases small c values tended to be chosen.

Finally, the results obtained for the procedures that include the external data set for training are discussed in Section G of Online Resource 1.

3.3 Additional Study: Optimistic Bias by Using the External Data Set for both Tuning and Prediction Performance Estimation

Figure 6 shows the prediction performance estimates obtained based on data set B using `ExtValidNoCV` and `ExtValidCV` in comparison with the unbiased estimates based on independent test data (see Figure S9 in Section H of Online Resource 1 for the extended results). Obviously, both approaches lead to a substantial overestimation of the prediction performance. Interestingly, the degrees of overestimation do not noteworthily differ between `ExtValidNoCV` and `ExtValidCV`. This shows that the fact that the observed AUC values are maximally selected in `ExtValidNoCV`, does not or hardly contributes to

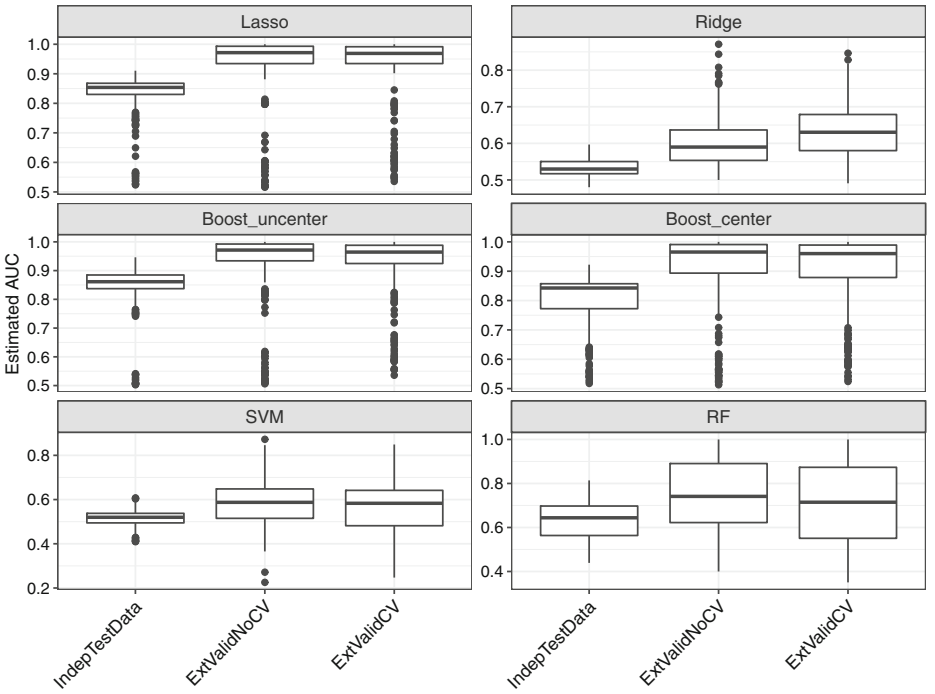

Fig. 6 Additional study. Prediction performance estimates based on independent test data `IndepTestData` (left) compared with estimates `ExtValidNoCV` (middle) and `ExtValidCV` (right)

the observed bias associated with this procedure. Instead, the main reason for the upward bias in the AUC estimates obtained with `ExtValidNoCV` and `ExtValidCV` is the fact that the same source is used for both tuning and prediction performance estimation.

Figure 6 clearly indicates that the prediction performance estimates obtained using both `ExtValidNoCV` and `ExtValidCV` are not useful for measuring cross-study prediction performance. However, those obtained using `ExtValidCV` are by construction valid for estimating the prediction performance on observations that follow the same distribution as the external data set used for tuning. Against this background, the fact that the AUC estimates obtained using `ExtValidCV` are considerably higher than the independent test data based prediction performance estimates shows the following: the tuning parameter values obtained using external tuning are more suitable for observations from the same source as the external data set than they are for other data. In other words, we observe an overadaptation of the chosen tuning parameter value to the source of the data used for external tuning. However, as seen in the previous Section 3.2, for some prediction methods we can still expect improvement by performing external tuning in contrast to internal tuning in terms of prediction performance.

3.4 A Conservative Prediction Performance Estimation Procedure

The analysis presented in Section 3.3 revealed that the external data set used for tuning cannot be used for testing the performance of the prediction rule without introducing a major optimistic bias. Thus, in order to obtain a realistic prediction performance estimate it would

be necessary to use a second external data set that was neither used for training nor for tuning. However, a second external data set may often not be available in applications or the effort of obtaining one may not be justifiable as the improvement in prediction performance through using external tuning is not huge in general.

We suggest the following pragmatic procedure to obtain a conservative prediction performance estimate. Regardless of the procedure used for parameter tuning (internal, external, or robust tuning), (i) construct a prediction rule based on data set A using internal tuning, and (ii) estimate its prediction performance on the external data set B. This estimate can be seen as a—according to our results, somewhat pessimistic—surrogate of the prediction performance obtained using other (external or robust) tuning approaches. This (slightly) conservative estimate should be appropriate in many cases, because, in applications, it is most important that prediction performance is not overestimated. This procedure is not only parsimonious (no second external data set is needed) but also allows the analyst to try several tuning approaches without generating an optimistic bias in the performance estimate when selecting the approach that performed best on the external data set.

4 Discussion and Conclusions

4.1 Sources and Implications of Batch Effects in Biomedical Data

An important application field of the statistical methodology considered in this paper are biomedical data. Batch effects are a significant source of variation within high- and low-dimensional biomedical data that are not related to the underlying biology. These effects negatively affect various kinds of studies, in particular those aiming to identify feature signatures (i.e., genes, micro RNAs, proteins, metabolites, etc.) and predict outcomes of interest such as time-to-event endpoints (e.g., survival or disease recurrence) or binary or categorical endpoints (e.g., diseased/not diseased). The causes of batch effects are manifold and can be distinguished into two major classes: batch effects of experimental data and batch effects of clinical data. For experimental data, batch effects can occur in low- as well as in high-throughput data. However, for the latter they can be easier identified (Leek et al. 2010). As noted in the introduction, batches, in our definition, are groups of observations that show differences in measured data that are not caused by the biological signal of interest. Batch effects (i.e., systematic differences between measurements from different batches) can be, for instance, introduced by the technical personnel preparing and processing biological samples within one lab that apply slightly different routines while sticking to standard protocols or due to inter-lab specific differences with regards to personnel or instrumental differences. Beyond these specific examples, multiple batch effect causing sources are known (see Irizarry et al. 2005; Goh et al. 2017; Leek et al. 2010 for extensive reviews of these sources). Although omics data generation, at least at the genomic, epigenomic, transcriptomic and posttranscriptomic levels, is increasingly switching to highly standardized next-generation sequencing technologies, it is unlikely that batch effects will disappear in future applications—so the problem remains relevant to high-dimensional data analysis. Tom et al. (2017) present a workflow for the identification and mitigation of batch effects in whole genome sequencing data and one of the mostly used Bioconductor R packages, DESeq2, has an implementation of linear-model based batch correction of RNAseq data (Love et al. 2014).

Compared with batch effects of high-dimensional data, which are comprehensively assessed, little is known about batch effects in clinical data. Clinical data usually provide

time-to-event endpoints such as survival or recurrence in addition to tumor size, stage, grading, hormone receptor status and other prognostic or predictive factors. When assessed in the frame of monocenter studies, batch effects should be small due to consistent procedures and common rules implemented in the same clinics. However, changes in protocols over time can also introduce batch effects that should be considered in the analysis of data, making use of clinical endpoints. When assessing clinical endpoints in multi-center settings, batch effects can occur due to slightly different definitions of these endpoints. Mathews et al. (2016) present an approach for the elimination of batch effects in histopathological images which are the basis for a lot of prognostic factors in clinical data. However, since for most clinical data it will remain difficult to identify batch effects, communication between different centers for the purpose of data harmonization should be the strongest tool for ruling out such effects. Batch effects between data from different, unrelated sources can be expected to be strong due to the absence of data harmonization efforts between unrelated sources. In this context, the disparity between the prediction performance estimate of a clinical prediction rule obtained on external data and that obtained on observations from the same source can be huge (Bleeker et al. 2003).

To conclude, batch effects have various sources in biomedical data and their strength depends on the specific application field under consideration.

4.2 Summary and Conclusion

We compared various strategies for parameter tuning with respect to their performance in cross-study prediction. Several of these strategies were designed to deliver robust prediction rules that can perform better in cross-study prediction than conventional strategies, because they avoid over-fitting to the distribution of the training data.

The robust tuning procedure `RobustTuneC` can be used as a default tuning procedure. Nevertheless, external tuning (using `Ext`) may also be considered. External tuning has the advantage that it is easier to perform using any standard statistical software. Nevertheless, the analysis presented in Section 3.3 suggests that external tuning may lead to tuning parameter values that are too specific to the considered external data set. This problem can be expected to be less prevalent for `RobustTuneC`, unless a very large number of c values are considered with this procedure. Moreover, given that all considered c values are larger than or equal to one, this robust tuning procedure has the further advantage that it always delivers prediction rules that are at least as robust as those obtained using internal tuning.

Summarizing, we recommend using, as a first choice, `RobustTuneC` and, as a second choice, the more easily implementable external tuning. Moreover, regardless of the used tuning procedure, we suggest reporting the prediction performance estimate obtained on the external data with internal tuning—as a conservative surrogate of the performance on independent data (see Section 3.4). We plan to implement `RobustTuneC` and external tuning (using `Ext`) for the prediction methods considered in this paper in a CRAN R package, `RobustPrediction`. Readers may also consult the Electronic Appendix (Online Resource 2) for R code implementing `RobustTuneC` and `Ext`.

In this paper, we focused exclusively on high-dimensional data, which is why our conclusions are only valid for applications to data of this type. However, there might also be low-dimensional data applications for which there is relevant heterogeneity among data sources. In such applications, the methods considered in our paper may also yield improvements compared with internal tuning. Comparison studies, analogous to those presented in this paper, would be required to determine which of the considered approaches could be valuable in applications to low-dimensional data.

Acknowledgments We thank Alethea Charlton for making valuable language corrections. This work has been partially supported by the German Research Foundation [grant number BO3139/4-3 to ALB] and the German Federal Ministry of Education and Research (BMBF) [grant number 01IS18036A to BB and ALB (Munich Center of Machine Learning)]. The authors of this work take full responsibilities for its content.

Funding Information Open Access funding provided by Projekt DEAL.

Compliance with Ethical Standards

Conflict of Interest The authors declare that they have no conflict of interest.

References

Bernau, C., Riester, M., Boulesteix, A.L., Parmigiani, G., Huttenhower, C., Waldron, L., Trippa, L. (2014). Cross-study validation for the assessment of prediction algorithms. *Bioinformatics*, *30*(12), i105–i112.

Bischl, B., Lang, M., Kotthoff, L., Schiffner, J., Richter, J., Studerus, E., Casalicchio, G., Jones, Z.M. (2016). mlr: machine learning in R. *Journal of Machine Learning Research*, *17*(170), 1–5.

Bischl, B., Richter, J., Bossek, J., Horn, D., Thomas, J., Lang, M. (2017). mlrMBO: a modular framework for model-based optimization of expensive black-box functions, arXiv:1703.03373.

Bleeker, S.E., Moll, H.A., Steyerberg, E.W., Donders, A.R.T., Derksen-Lubsen, G., Grobbee, D.E., Moons, K.G.M. (2003). External validation is necessary in prediction research: a clinical example. *Journal of Clinical Epidemiology*, *56*, 826–832.

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

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

Chapelle, O., Vapnik, V., Bousquet, O., Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, *46*, 131–159.

Claesen, M., & De Moor, B. (2015). Hyperparameter search in machine learning, arXiv:1502.02127.

Collins, G.S., de Groot, J.A., Dutton, S., Omar, O., Shanyinde, M., Tajar, A., Voysey, M., Wharton, R., Yu, L.M., Moons, K.G., Altman, D.G. (2014). External validation of multivariable prediction models: a systematic review of methodological conduct and reporting. *BMC Medical Research Methodology*, *14*, 40.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*, 273–297.

Dondelinger, F., Mukherjee, S., The Alzheimer's Disease Neuroimaging Initiative (2020). The joint lasso: high-dimensional regression for group structured data. *Biostatistics*, *21*, 219–235.

Friedman, J., Hastie, T., Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, *33*(1), 1–22.

Goh, W.W.B., Wang, W., Wong, L. (2017). Why batch effects matter in omics data, and how to avoid them. *Trends in Biotechnology*, *35*, 498–507.

Hoerl, A.E., & Kennard, R.W. (1970). Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, *12*, 55–67.

Hornung, R., Bernau, C., Truntzer, C., Wilson, R., Stadler, T., Boulesteix, A.L. (2015). A measure of the impact of CV incompleteness on prediction error estimation with application to PCA and normalization. *BMC Medical Research Methodology*, *15*, 95.

Hornung, R. (2016). *Preparation of high-dimensional biomedical data with a focus on prediction and error estimation*. Dissertation: University of Munich.

Hornung, R., Causeur, D., Bernau, C., Boulesteix, A.L. (2017). Improving cross-study prediction through addon batch effect adjustment or addon normalization. *Bioinformatics*, *33*, 397–404.

Hothorn, T., Buehlmann, P., Kneib, T., Schmid, M., Hofner, B. (2018). mboost: model-based boosting, R package version 2.9-1.

Irizarry, R.A., Warren, D., Spencer, F., Kim, I.F., Biswal, S., Frank, B.C., Gabrielson, E., Garcia, J.G., Geoghegan, J., Germino, G., Griffin, C., Hilmer, S.C., Hoffman, E., Jedlicka, A.E., Kawasaki, E., Martinez-Murillo, F., Morsberger, L., Lee, H., Petersen, D., Quackenbush, J., Scott, A., Wilson, M., Yang, Y., Ye, S.Q., Yu, W. (2005). Multiple-laboratory comparison of microarray platforms. *Nature Methods*, *2*, 345–350.

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

Lin, S.W., Ying, K.C., Chen, S.C., Lee, Z.J. (2008). Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert Systems with Applications*, *35*, 1817–1824.

Love, M.I., Huber, W., Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, *15*, 550.

Mathews, A., Simi, I., Kizhakkethottam, J.J. (2016). Efficient diagnosis of cancer from histopathological images by eliminating batch effects. *Procedia Technology*, *24*, 1415–1422.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F. (2019). e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien, r package version 1.7-0.1.

Rohart, F., Eslami, A., Matigian, N., Bougeard, S., Lê Cao, K.A. (2017). MINT: A multivariate integrative method to identify reproducible molecular signatures across independent experiments and platforms. *BMC Bioinformatics*, *18*, 128.

Scherer, A. (Ed.) (2009). *Batch effects and noise in microarray experiments: sources and solutions wiley series in probability and statistics*. Wiley: Hoboken.

Siontis, G.C.M., Tzoulaki, I., Castaldi, P.J., Ioannidis, J.P.A. (2015). External validation of new risk prediction models is infrequent and reveals worse prognostic discrimination. *Journal of Clinical Epidemiology*, *68*, 25–34.

Snoek, J., Larochelle, H., Adams, R.P. (2012). Practical Bayesian optimization of machine learning algorithms. In Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (Eds.) *Advances in Neural Information Processing Systems*, (Vol. 25 pp. 2951–2959): Curran Associates, Inc.

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, *58*, 267–288.

Tom, J.A., Reeder, J., Forrest, W.F., Graham, R.R., Hunkapiller, J., Behrens, T.W., Bhangale, T.R. (2017). Identifying and mitigating batch effects in whole genome sequencing data. *BMC Bioinformatics*, *18*, 351.

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

Wright, M.N., & Ziegler, A. (2017). ranger: a fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, *77*(1), 1–17.

Zhang, Y., Bernau, C., Parmigiani, G., Waldron, L. (2020). The impact of different sources of heterogeneity on loss of accuracy from genomic prediction models. *Biostatistics*, *21*, 253–268.