RESEARCH ARTICLE OPEN ACCESS

Rethinking the Handling of Method Failure in Comparison Studies

Milena Wünsch[1,2] | Moritz Herrmann[1,2] | Elisa Noltenius[3] | Mattia Mohr[3] | Tim P. Morris[4] | Anne-Laure Boulesteix[1,2]

[1]Institute for Medical Information Processing, Biometry, and Epidemiology, Faculty of Medicine, LMU Munich, Munich, Germany | [2]Munich Center for Machine Learning (MCML), Munich, Germany | [3]Department of Statistics, LMU Munich, Munich, Germany | [4]MRC Clinical Trials Unit at UCL, University College London, London, UK

Correspondence: Milena Wünsch (milena.wuensch@ibe.med.uni-muenchen.de)

Received: 16 August 2024 | **Revised:** 29 July 2025 | **Accepted:** 28 August 2025

Funding: This work is supported in part by funds from the German Research Foundation (DFG: BO3139/7 and BO3139/9-1). TPM was funded by the UK Medical Research Council (Grant No. MC_UU_00004/09).

Keywords: benchmark studies | comparison studies | non-convergence | simulation studies

ABSTRACT

Comparison studies in methodological research are intended to compare methods in an evidence-based manner to help data analysts select a suitable method for their application. To provide trustworthy evidence, they must be carefully designed, implemented, and reported, especially given the many decisions made in planning and running. A common challenge in comparison studies is to handle the "failure" of one or more methods to produce a result for some (real or simulated) data sets, such that their performances cannot be measured in those instances. Despite an increasing emphasis on this topic in recent literature (focusing on non-convergence as a common manifestation), there is little guidance on proper handling and interpretation, and reporting of the chosen approach is often neglected. This paper aims to fill this gap and offers practical guidance on handling method failure in comparison studies. After exploring common handlings across various published comparison studies from classical statistics and predictive modeling, we show that the popular approaches of discarding data sets yielding failure (either for all or the failing methods only) and imputing are inappropriate in most cases. We then recommend a different perspective on method failure—viewing it as the result of a complex interplay of several factors rather than just its manifestation. Building on this, we provide recommendations on more adequate handling of method failure derived from realistic considerations. In particular, we propose considering fallback strategies that directly reflect the behavior of real-world users. Finally, we illustrate our recommendations and the dangers of inadequate handling of method failure through two exemplary comparison studies.

1 | Introduction

In methodological research, comparison studies aim to compare the performance of methods to provide empirical evidence on their behavior and help data analysts choose appropriate methods for their setting. To be reliable, they require careful planning of the design, execution, and reporting of the results. These issues have gained increasing attention in recent years, both in the context of simulation studies based on artificially generated data [1–3] and in the context of real data-based benchmark studies [4, 5].

In this work, we focus on a particular aspect of the design of comparison studies that affects both simulation and benchmark studies, namely "method failure." Method failure occurs when a method under investigation fails to produce an output for a (real or simulated) data set considered in the comparison study. The issue of method failure in comparison studies has begun to attract attention in recent years [1–3], often focusing on non-convergence. However, there is little guidance on how to handle method failure appropriately, whether it occurs as non-convergence or in another form. In particular, published reviews have shown that method failure goes unreported in the majority of cases. For instance, in a systematic review of 42 published simulation studies on the analysis of complex longitudinal patient-reported outcomes data, less than half *"acknowledge that model non-convergence might occur"* [6]. Even more notably, only 12 of 85 applicable research articles reviewed by Morris et al. [1] report convergence as a performance measure. This contrasts with our perception—both through our own experiences and informal discussions with colleagues—that the problem of method failure is highly prevalent. In particular, signs of method failure are often traceable in the code provided for reproducibility, even when not reported in the manuscript. This is often clear through the use of explicit error-handling mechanisms (e.g., functions `try()` or `tryCatch()` in `R` or `try-except` constructs in `Python`).

To illustrate the difficulty of handling method failure adequately, imagine a researcher conducting a comparison study in statistics or predictive modeling to compare the performance of three methods based on four simulation/benchmark data sets. In the context of classical statistics, they likely focus on comparing the methods' performances (e.g., bias) in *"estimating one or more population quantities"* [1], sometimes called "estimand(s)," and base the analysis on data simulated using a certain data-generating mechanism (DGM). In predictive modeling, they are likely to compare the methods' predictive performances (e.g., accuracy), that is, their ability to make predictions on a new set of observations. Here, the analysis is typically performed on real benchmark data sets. For simplicity, we refer to both settings as "simulation studies" and "benchmark studies," respectively (though admitting that there may be intermediate situations, e.g., simulation studies investigating predictive performance on artificial data).

During the experiments, the researcher encounters the failure of some methods in one or more of the following ways: (i) they receive an error message or output of *NA* ("not applicable") or *NaN* ("not a number"), implying that a certain calculation could not be completed (e.g., non-convergence), (ii) their system crashes, or (iii) the experiment runs for an excessively long time, forcing the researcher to abort.

While the majority of real published comparison studies do not report method failure, there are positive examples from classical statistics and predictive modeling in which the occurrence (and oftentimes also the applied handling) of method failure is described. For instance, van Smeden et al. [7] investigate why different simulation studies offer conflicting minimal "events per variable (EPV)" recommendations for logistic regression. All three studies featured, exploring the effect of EPV on performance measures such as bias and coverage, encounter separated data sets (i.e., outcome variable is perfectly predicted by one or a combination of multiple covariates), which lead to *convergence issues* of the maximum likelihood process. That is, convergence is not achieved within the pre-specified number of iterations, or the process converges to a point that is not the maximum likelihood estimate [7]. In a study from the statistical learning field, Gijsbers et al. [8] conduct a comparison study of nine automated machine learning methods ("AutoML frameworks"). They encounter and report method failure in the forms of (i) *exceeding available memory* (or other memory-related issues), (ii) *exceeding time limits*, (iii) *failure related to data set characteristics* (such as highly imbalanced data), and (iv) failure caused by *errors in the implementation* of the AutoML method [8].

Regardless of the form in which a researcher encounters method failure, it leads to NA for the associated method-data set combination, where otherwise an estimate (for simulation studies) or a performance value (for benchmark studies) is expected. Table 1 depicts a corresponding summary of the results of the hypothetical researcher's study across all simulation repetitions/benchmark data sets and methods. In simulation studies, the NAs complicate the derivation of performance (e.g., bias as the mean deviation of the estimates from the true value across all repetitions). In benchmark studies, on the other hand, they hinder the aggregation of performance values into an average performance (e.g., average estimated accuracy as the mean fraction of correct predictions across all benchmark data sets). In both cases, method failure complicates the assessment and especially

TABLE 1 | Overview table of (a) a fictive simulation study across four repetitions (rows) and (b) a fictive benchmark study across four benchmark data sets (rows) for three methods (columns). Method failure results in undefined values for the affected combination of data set and method, marked by "NA."

(a) Fictive simulation study: Overview of estimates, with the true value being equal to 4. Method failure first leads to an undefined estimate value for the given method—data set combination, complicating the derivation of bias for that method.

Repetition	Method 1	Method 2	Method 3
1	3.89	4.23	4.08
2	3.78	4.13	4.11
3	**NA**	3.69	4.23
4	3.75	4.24	**NA**
Bias	**?**	0.07	**?**

(b) Fictive benchmark study: Overview of estimated accuracies across all method—data set combinations. Method failure directly leads to an undefined performance (i.e., accuracy) value for the given method—data set combination.

Benchmark data set	Method 1	Method 2	Method 3
1	0.85	0.88	0.87
2	0.9	0.91	0.86
3	**NA**	0.80	0.82
4	0.78	0.76	**NA**
Average accuracy	**?**	0.84	**?**

the comparison of performance of the methods under investigation, such that the researcher finds themself wondering how to proceed adequately.

The above-shown examples of published comparison studies alone already illustrate different viewpoints on method failure and correspondingly different ways in which it is handled in practice. Van Smeden et al. [7] observe that separated *data sets are commonly removed* from the analysis. This, however, affects bias, mean squared error (MSE), and confidence interval width, even if the frequency of separated data sets is small [7]. Gijsbers et al. [8] argue against discarding data by describing that method failure is typically correlated with data set characteristics. Instead, they *impute* by the performance of a constant predictor.

Examining additional published studies, which we do throughout this paper, shows that imputation and discarding data sets associated with failure are among the most common practices. However, it also becomes clear that they are only two of many handlings applied in practice and that others, such as modifying the affected methods' implementations and changing software, are common as well — an indicator that proper guidance is commonly lacking. Our paper aims to fill this gap and provides detailed instructions on how to handle method failure. In particular, we illustrate why usual handlings of method failure derived from treating the resulting NAs as regular "missing data," and especially imputation and discarding data, are inappropriate in most settings (leading us to refer to "undefined" rather than "missing" performance values in instances of method failure). As an alternative, we promote considering method failure from a different viewpoint, namely as the result of a complex and individual interplay of multiple factors — which we use to derive alternative and more appropriate measures that can be applied when encountering method failure.

Our paper targets both the planning and the execution stages of a comparison study. While many instances of method failure can only be directly observed — and therefore investigated — after the study is executed, certain considerations should be made while planning to anticipate and ideally reduce the potential for failure.

The rest of this paper is structured as follows. In Section 2, we give an overview of the status quo by reviewing and summarizing the different manifestations and handlings of method failure reported in published comparison studies, followed by a detailed discussion of why discarding data or imputing values is usually inadequate when encountering method failure. Building on this, Section 3 is a compilation of alternative recommendations for handling method failure, which differentiates between different scopes of the study and research stages in which it may take place. In particular, we promote using fallback strategies when encountering method failure, which enables the aggregation of performance across data sets even when method failure occurs, and on top of that, reflects the behavior of real method users across many settings. Considering possible time constraints, we also discuss "slimmed-down" ways in which authors can implement part of our recommendations if unable to follow them completely. Finally, in Section 4, we demonstrate our recommendations using two fictive comparison studies that illustrate the complexity of the discussed issues.

2 | Method Failure in Published Comparison Studies

In Section 2.1, we summarize common manifestations and handlings of method failure reported in the literature. Since some authors focus on how method failure is handled rather than explaining how it occurred, the corresponding studies are discussed only in Section 2.1.2. An overview of all study contexts, manifestations, and handlings of method failure is provided in Table 2.

2.1 | Status Quo: Occurrence and Handling of Method Failure

2.1.1 | Common Manifestations of Method Failure

Non-convergence and other calculation issues Authors often encounter the inability of a method to perform a calculation necessary for prediction or estimation, either implied through an error, a warning, or a meaningless output. For methods employing an iterative procedure (e.g., an iterative maximum-likelihood procedure), this is called "non-convergence," that is, the method fails to produce a valid output within the pre-specified number of iterations.

Examples Zapf et al. [9] encounter non-convergence for two of the three methods under investigation. One returns an error, while the other outputs the current, non-converged estimate in these instances [9]. Masaoud and Stryhn [12] encounter instances of *"non-convergence or non-sensible estimates"* [12] for some method-data set combinations. In the study by Crowther et al. [13], the methods are based on numerical integration and encounter estimation difficulties (non-convergence) with *"challenging data sets"* [13]. Hornung et al. [11] report instances of method failure when methods are *"not applicable"* [11] to certain differences between training and test sets regarding the patterns of block-wise missingness. Additionally, they report calculation issues for one method, explicitly disallowing certain structures in the training data set, resulting in failure. Calculation issues are also reported in the study by Ruxton and Neuhäuser [10], where some odds ratio estimation methods fail due to division by 0 when the 2×2 contingency table contains one or more zero entries. Fernández-Delgado et al. [15] report several data set characteristics associated with issues in required calculations, including *"collinearity of data, singular covariance matrices, and equal [predictor values] for all the training [observations], [. . .] discrete predictor variables, classes with low populations, or too few classes"* [15]. Additionally, a group of methods in their study requires a certain minimum number of observations per outcome class and is therefore not applicable to particularly small data sets. Another case of calculation issues can be found in the simulation study conducted by Dunias et al. [14]. For some data sets, the statistical learning method LASSO does not select any predictor variable, leading to a constant predictor and preventing the assessment of performance regarding calibration slope.

Memory issues Even before exceeding possible time budgets, some methods crash due to memory issues. That is, the given method consumes more memory than is available.

TABLE 2 | Selection of comparison studies from statistics and predictive modeling that report the occurrence and handling of method failure.

Comparison study		Manifestations			Handlings						
					Resolve method failure			Summarize despite method failure			
Author	**Context of the study**	**Non-convergence (and others)**	**Runtime issues**	**Memory issues**	**Remove method**	**Modify implementation**	**Change software**	**Discard data for failing methods**	**Discard data for all methods**	**Impute**	**Include performance measure**
van Smeden et al. [7][a]	Simulation study to investigate why different published simulation studies offer conflicting minimal "events per variable (EPV)" recommendations for logistic regression. Features three published simulation studies exploring the effect of EPV on performance measures.	×							×		
Gijsbers et al. [8]	Comparison study of nine automated machine learning ("AutoML") methods.	×	×	×						×	
Zapf et al. [9]	Simulation study to compare three frequentist methods for the meta-analysis of diagnostic accuracy studies regarding sensitivity and specificity.	×						×			×
Ruxton and Neuhäuser [10]	Simulation study on the empirical coverage of eight methods for calculating confidence intervals for the odds ratio for binary outcome and exposure.	×			×						
Hornung et al. [11]	Benchmark study using thirteen multi-omics data sets to assess the performance of seven prediction methods for multi-omics data that can handle block-wise missingness by artificially inducing specific patterns of block-wise missing values into initially complete data.	×		×					×		
Masaoud and Stryhn [12]	Two simulation studies to compare the performance of seven marginal and random effects estimation methods for binary longitudinal data (e.g., GEE, Bayesian Markov Chain Monte Carlo).	×			×	×	×				

(Continues)

TABLE 2 | (Continued)

Comparison study		Manifestations			Handlings						
					Resolve method failure			Summarize despite method failure			
Author	**Context of the study**	**Non-convergence (and others)**	**Runtime issues**	**Memory issues**	**Remove method**	**Modify implementation**	**Change software**	**Discard data for failing methods**	**Discard data for all methods**	**Impute**	**Include performance measure**
Crowther et al. [13]	Simulation study to assess the performance of a multilevel mixed effects parametric survival model using adaptive and nonadaptive Gauss-Hermite quadrature.	×				×					
Dunias et al. [14]	Simulation study to compare the performance of five hyperparameter tuning procedures for clinical prediction models built upon statistical learning methods.	×								×	
Fernández-Delgado et al. [15]	Benchmark study of 179 supervised classification methods across 121 benchmark data sets.	×		×				×		×	
Herrmann et al. [16]	Benchmark study on the predictive performance of thirteen survival time prediction methods based on eighteen high-dimensional multi-omics data.	*	×	*	×[b]		×			×	
Bischl et al. [17]	Comparison study of eleven classification methods on 49 real benchmark and simulated data sets.	*	*	*						×	
Niessl et al. [18][c]	Metascientific study to investigate the effect of different imputation procedures on the performance assessment and ranking of the methods. Considered alternative procedures include imputation by the worst possible value and by a value that weighs the method's overall proportion of method failure against its performance in the remaining folds.	*	*	*						×	

[a] van Smeden et al. [7] *observe* in published comparison studies that separated data sets (i.e., data sets leading to non-convergence) are commonly omitted from analyses.
[b] Herrmann et al. [16] apply a moderate form of data removal by restricting the analysis to the subset of smaller benchmark data sets for one method.
[c] Nießl et al. [18] perform an example benchmark study based on the results of the study by Herrmann et al. [16] to illustrate how different ways of imputation can influence the results of a benchmark study.
* The authors did not report details on the manifestation of method failure.

Examples Gijsbers et al. [8] report memory constraint violations, segmentation faults, and Java server crashes as instances of method failure related to memory. Hornung et al. [11] also encounter memory issues, which they attribute to a specific missingness pattern induced into the training data. Fernández-Delgado et al. [15] encounter memory issues in combination with large data sets.

Runtime issues Another form of method failure often reported in comparison studies is excessive runtime. Some studies set fixed time budgets, so exceeding them is considered method failure. Others without fixed budgets encounter method failure when experiments for single methods run for an excessive amount of time (e.g., days), forcing the researcher to abort. Unlike non-convergence and memory issues, method failure is *actively declared* by the researchers when runtime issues occur, as without these (naturally necessary) limitations, a method might eventually generate an output at some point. Resulting implications are discussed in Section 3.1.6.

Examples While Gijsbers et al. [8] encounter the excess of predefined time budgets for model training, Herrmann et al. [16] face runtime issues without fixed budgets. They encounter *"computations lasting several days for one single model fit for large data sets"* [16] for one method.

2.1.2 | Common Approaches to Handling Method Failure

Based on the published comparison studies, the handling of method failure can be commonly divided into two general approaches. Note that a literature review by Pawel et al. [19], yielding a structured investigation of the frequencies of handling approaches from published simulation studies, results in four main handlings, which are among those we have commonly observed.

Resolving method failure Some authors resolve method failure completely so that no instances remain. A straightforward way observable in the literature is to completely *remove the affected method from the analysis*. For instance, Ruxton and Neuhäuser [10] exclude all methods that fail with zeros in the 2×2 contingency table. Masaoud and Stryhn [12] use method removal as a last resort in their sequence of handlings when previous measures (see below) have been unsuccessful. This illustrates well that oftentimes, different handlings are applied within the same study, sometimes switching between approaches or applying them sequentially until successful. Note that a more moderate form of method removal can be observed in the benchmark study by Herrmann et al. [16] For the method encountering runtime issues, they restrict the analysis to the subset of smaller benchmark data sets and report the corresponding results separately from the remaining methods.

An alternative to removing methods is to *modify their implementation* such that an output can be successfully produced. For instance, in the study by Masaoud et Stryhn [12], *"different optimization techniques were tried"* [12] for one method when the default procedure resulted in failure. Crowther et al. [13] react to method failure by successively increasing the number of quadrature points until convergence is reached.

Lastly, *changing software* for the method affected by method failure can also be observed in the literature. Switching the operating system can be observed in the study by Herrmann et al. [16], who, for one method, encountered *"a fatal error in R under Windows, but not using the Linux distribution Ubuntu 14.04"* [16]. Masaoud et al. [12], while remaining in the same operating system, generate the required output for one method in `SAS` when it fails in `R`.

Summarizing performance despite method failure Instead of resolving method failure, some authors summarize the methods' performances despite existing failures. A widespread approach is to discard data sets associated with failure. The first version of this approach, consisting of *discarding data sets for the failing methods only*, is followed by Zapf et al. [9] for the method affected by errors due to non-convergence. Fernández-Delgado et al. [15], comparing the methods' performances based on two performance measures, discard the associated data sets for the methods affected by method failure in the comparison based on average accuracy. The second version of discarding data, namely *discarding the corresponding data sets for all methods*, is applied by Hornung et al. [11]. Particularly, they argue against discarding the data sets for the failing methods only by saying that it would not yield a fair comparison since predictive performance is generally associated with the induced pattern of block-wise missingness [11].

Instead of discarding data, it is also common to *impute performance values* for the instances of method failure. Note that imputation yields a variety of options. Fernández-Delgado et al. [15], for instance, impute by the mean accuracy of all remaining methods on the given data set for their second performance measure (Friedman ranking) when a method fails, as it requires the same number of performance values across all methods. On the other hand, recall Gijsbers et al. [8] and their comparison study of AutoML methods in which they impute with a constant predictor when a method fails. They choose this *"very penalizing imputation strategy"* [8], which can be viewed as a form of imputation by the worst possible value, with a similar reasoning to Hornung et al. [11]. That is, failure may be correlated with data set characteristics such that discarding data sets for the failing methods or imputing by the average performance of the given method on the remaining data sets may unfairly favor the failing methods [8]. Note that Dunias et al. [14] also impute by a constant predictor when a statistical learning method fails.

An alternative way of imputing is employed by Bischl et al. [17], who impute performance values based on the proportion of repetitions in which the affected method fails. Below 20%, they impute by sampling from a normal distribution estimated from the remaining performance values of that model. Above 20%, on the other hand, they impute by the worst possible value with the reasoning that the method's behavior is *"too unreliable for the current data set"* [17]. A similar imputation strategy is employed by Herrmann et al. [16]. While using the same imputation rule when a method fails on more than 20% of data sets, they otherwise impute by the mean performance of the method in the remaining "successful" data sets. In a subsequent project, Nießl et al.

[18] consider alternative procedures, including imputation by the worst possible value and by a value that weights the method's overall proportion of method failure against its performance in the remaining folds. They show that the approach to imputing the undefined performance values has a notable effect on the performance ranking between the methods.

When summarizing performance despite the failure of an existing method, some authors *report the proportion of (real or simulated) data sets for which each method fails*. For instance, Hornung et al. [11] list "the frequency of data sets with missing results" [11] for the methods affected by method failure. Gijsbers et al. [8] provide multiple failure plots, consisting of the number of failures per method with differentiation of the different failure types, the number of failures by size of the associated data set, and boxplots of the training durations for each method with indicators of excessive runtimes. Zapf et al. [9] even include convergence as a performance measure, supported by a table of the number of converged simulation runs for each simulation scenario across the methods in the supplement.

2.2 | Why Popular Handlings are Often Inadequate

2.2.1 | Why Discarding Data is Inadequate

Currently, there is no one-size-fits-all solution to handling method failure, exemplified by the examples in Section 2. This section discusses some principles to guide decisions. To illustrate ideas, we begin with two simple examples and some useful terminology. Suppose a simulation study aims to compare the performance of method A with method B in terms of bias. Method A fails in 20% of repetitions while method B does not fail in any. Alternatively, suppose a benchmark study aims to compare the average accuracy of two methods. Again, method A fails in 20% of benchmark data sets while method B does not fail in any. The hypothetical results for the two cases are given in Table 3a,b.

TABLE 3 | Overview table of an illustrative comparison study comparing the performance of (a) two estimation methods regarding bias and (b) two prediction methods regarding average accuracy. Method A fails in 20% of (a) simulation repetitions and (b) benchmark data sets.

(a) Hypothetical simulation study

Bias	Unconditional (all repetitions)	Conditional (repetitions where A does not fail)
Method A	—	0.09
Method B	0.14	0.09

(b) Hypothetical benchmark study

Average accuracy	Unconditional (all data sets)	Conditional (data sets where A does not fail)
Method A	—	0.85
Method B	0.72	0.85

We contrast the terms *unconditional versus conditional* performance and *absolute versus relative* performance. *Unconditional* performance simply means "in all data sets," while *conditional* means "in data sets where this method did not fail." The *unconditional* bias of method B is 0.14, while the unconditional bias of method A is not defined. The conditional bias of both methods is 0.09. Parallel descriptions apply to the hypothetical benchmark study in Table 3b. *Absolute* performance means "considered on its own" and *relative* performance means "when compared with another method."

These terms help to distill the problems with method failure in comparison studies. Clearly, we want to compare the methods' unconditional performances. However, in both Table 3a,b, the relative unconditional performance of methods cannot be compared because the unconditional performance of method A is undefined. In contrast, the relative conditional performance can be quantified and compared ("discarding data sets associated with failure for all methods"): In Table 3a, the two methods have the same bias conditional on non-failure of method A; in Table 3b, the two methods have the same accuracy conditional on non-failure of method A. However, the *conditional* performance of method B should be of little interest. Why should a method's performance depend on the behavior of another method regarding failure? In Table 3, method B's performance improves when assessed based on only those data sets where A does not fail. This becomes even more complex as additional methods are included in the comparison, since discarding the corresponding data sets for all methods requires conditioning on the non-failure of each one.

Alternatively, one could intuitively proceed to compare the unconditional performance of method B with the conditional performance of method A ("discard data sets for the failing methods only"). However, comparisons should be performed vertically, or we are not comparing like with like: In the context of Table 3, this would reflect poorly on method B, even though both perform equally well for the data sets without failure. This is all the more problematic since method B, in this case, is actually preferable to method A, as it outputs a result for all data sets! Again, this becomes even more pronounced as additional methods are added to the comparison, since different methods typically fail on different datasets (see Section 3.1.1).

2.2.2 | Why Imputation is Inadequate

Another superficial solution often proposed when looking at results such as those in Table 3 is to interpret method failure as causing missing values and use missing data methods, such as imputation, to infer the unconditional performance of method A. After all, results given by different methods tend to be positively correlated, so results given by method B could inform the imputation (analogously, it may be intuitive to impute with the results of method A on the remaining data sets, as results tend to be correlated).

We view it as mistaken to regard method failure as a form of missing data and have been careful to refer to "undefined" rather than "missing" values. The missing data literature defines missing values as those that exist and were intended to be recorded, but for one reason or another are *"not observed"* [20, 21]. Method failure is far closer to what is called a truncating event in the estimands literature [22, 23], which denotes a value that does not exist. That is, when a method fails, the result we wished to record simply does not exist for that data set. Method failure should, therefore, not be viewed as obscuring the method's results on a given data set. Instead, failure is the result we get! For this reason, we should be cautious about applying missing data approaches, and particularly imputation, to method failure. Note that this also includes measures of "weighted performance," which weight the performance of a method according to the proportion of data sets with failure for that method. Using the correspondence between weighting and imputation approaches, this implies that weighting implicitly assigns some performance value in instances of method failure. The consequence of our dissuasion from "missing data-inspired" approaches is that our work focuses intensively on alternative ways to deal with method failure.

Note that there is an exception to method failure not leading to "missing" data. This is when a "failure" is recorded, but a result could have been obtained; for example, if failure occurs due to excessive runtime, but the definition of excessive in the comparison study can safely be judged as shorter than in practice. We refer to this case in Section 3.1.6.

3 | Principles and Recommendations for Handling Method Failure

The following section presents guidelines for handling method failure that are more appropriate than discarding data or imputation. Since, as previously mentioned, method failure can only be directly observed after the execution stage, the order of our recommendations aligns with the natural process of investigating these failures post-execution. However, we also highlight key considerations that can be addressed during the planning stage to help reduce both the potential for and unpredictability of failure in the first place. An overview of the recommendations can be found in Table 4.

Using the terminology of Heinze et al. [24], we distinguish between comparison studies in "early" and "late" research phases. Early phase studies investigate if a given method can be "used with caution" in settings similar or only slightly different from its original target setting, possibly refining or extending it while also making limited comparisons. Late phase studies, however, focus on realistic comparisons across diverse scenarios to guide real-world users. We make differentiated recommendations when necessary.

3.1 | Practical Recommendations for the Handling of Method Failure

3.1.1 | Consider Method Failure as the Result of a Complex Interplay

When looking beyond the manifestations of method failure summarized in Section 2.1.1, it becomes evident that these are just the tip of the iceberg. In the end, regardless of how method failure manifests, it is rooted in an individual and complex interplay of interconnected factors, that is,

> **Certain data set characteristics cause the method to fail in its current implementation (and on the given hardware and software).**

Besides influencing runtime and memory consumption [25], data set characteristics contributing to method failure are highlighted by the observation that a method usually fails for specific data sets, not all of them. Certain data set characteristics can complicate or completely prevent required calculations of a method within the maximum number of iterations (non-convergence), time, or memory budgets. Note that this association is addressed in several comparison studies listed in Section 2.1.1. For instance, the study by van Smeden et al. [7] addresses *perfect separation* of the outcome variable by one or more predictor variables as a cause of non-convergence of logistic regression algorithms. Recall also the data set characteristics associated with failure that are reported by Fernández-Delgado et al. [15], including *"collinearity of data, singular covariance matrices, and equal [predictor values] for all the training [observations], [. . .] discrete predictor variables, classes with low populations, or too few classes"* [15].

Conversely, the association of method failure with the (implementation of) a given method becomes clear from the usual observation that not all methods fail for the same data sets. That is, methods often vary in their ability to handle specific data set characteristics. For instance, in our illustrative study in Section 4.1, some odds ratio estimation methods are equipped with a way around "zeros," while others fail.

In addition to the interplay of method and data set characteristics, however, software can also affect the (non-)failure of a method such that results are obtained in one software but not another. Consider, for instance, that Herrmann et al. [16] report that one of the methods under investigation is completely unusable due to an error in R that occurred using Windows as an operating system, while it did not occur under Ubuntu. The dependency on hardware, on the other hand, becomes clear in how working memory, besides being directly associated with possible memory issues, can also considerably affect runtimes.

Ultimately, the core of method failure—stemming from the interplay of data set, method, software, and hardware—is considerably more informative for addressing method failure than the manifestation of failure itself. We, therefore, recommend that authors of comparison studies investigate and understand the interplay underlying failure for each instance after the execution phase. Ideally, they should identify if particular (and if different) data set characteristics cause the failures, leading to evaluations such as "method A fails when a predictor variable is constant across all observations and when a binary predictor has highly imbalanced classes." As a preparatory measure in the planning stage, authors should check user manuals and help pages for any described problems related to certain data structures for all methods under investigation.

TABLE 4 | Checklist of recommendations for handling method failure.

Recommended error handling workflow	Section
1. **Investigate and understand the interplay underlying each instance of method failure**. Interplay may consist of data set characteristics, the method's implementation, the given hardware, and software.	3.1.1
2. **Check for the correct use of the method**. Distinguish between "use beyond original scope" and "method misuse" when detecting that a failing method is used differently than intended by its developers.	3.1.2
3. **Check if data matches the scope of the study**. Check if the characteristics of the data associated with failure (and the failure itself) are realistic for the targeted users.	3.1.3
4. **Be realistic when modifying method implementations (for early-stage research)**. Restrict yourself to parameter adjustments that are realistic for future targeted users.	3.1.4
5. **Consider a fallback strategy (for late-stage research)**. Evaluate the performance of method pipelines of the type "use method A when it successfully produces an output, use method B otherwise." Define the pipelines based on preliminary considerations and avoid selectively reporting the "best" pipelines.	3.1.5
6. **Special case: Runtime issues**. When real users running the affected method only on a single data set can be assumed to successfully obtain a model output, run the method on a preselected random subset of all data sets (selection best made in the planning stage).	3.1.6
7. **Report failure proportions, underlying interplays, applied approaches, and share code**. Provide valuable insights to real users into the methods' usability and enhance a deeper understanding of method failure in general.	3.1.7
8. **Think ahead and pre-register, but remain flexible**. Conduct a pilot study before pre-registering the handling of method failure. Be aware that unforeseen failures may still occur; therefore, include error-handling mechanisms in your code from the start.	3.1.8
Slimmed-down error handling workflow when time constraints prevent recommendations 1–8	
Report (i) performance results after discarding the data sets for all methods, (ii) performance results after discarding the data sets for failing methods only, and (iii) failure proportions for all methods. Note that this approach does **not** provide a comparison of unconditional performance.	3.2

3.1.2 | Check for the Correct Use of the Method

When encountering method failure in the execution stage, authors should first check whether the method was truly developed for the given estimation or prediction task and the type of data (based on the investigations from the previous Section 3.1.1), and if it is used as intended by the developers. If this is not the case, a distinction should be made between two scenarios. First, a method in the study is used in a slightly different (but not contraindicated) context than originally intended by its developers. We call this "use beyond the original scope." Including this method in the study may be justified when real users commonly use this method "beyond its original scope," however, authors should make a corresponding remark in their manuscript. Second, if using the method in the given context is explicitly contraindicated by the method developers, including it in the study would correspond to method misuse. A method should not be penalized for failing on a data set or a task it was explicitly not intended for. Even if commonly (but probably unknowingly) "misused" by real users, this method should be excluded from the study, and the contraindication should be stated clearly in the manuscript. If none of the scenarios apply, authors should resort to Section 3.1.3.

All assessments regarding the use of a given method should be based on the familiarization with all methods from the experiment, which should already take place in the planning stage. This underlines the importance of carefully reading corresponding user manuals and checking for publications from real-world applications in which these methods are commonly used. This aspect should also be checked when planning the data generation in simulation studies and selecting benchmark data for benchmark studies. When in doubt, authors may also consult the developers of the individual methods (and possibly involve them in the study).

3.1.3 | Check if Data Matches Scope of Study

If method failure persists after ruling out an incorrect use in the execution stage, authors should assess whether real-world users are likely to encounter such data (and, therefore, this method failure). This assessment is shaped by the study's scope and is naturally not purely objective. Importantly, associated considerations should be reported in the study since they carry valuable information and affect the interpretation of the study's results.

Consider the case of method failure caused by the method's inability to make required calculations due to an "extreme" structure of the given simulated data set. For instance, White et al. [26] conduct an illustrative simulation study that examines methods for handling missing values in confounders in epidemiological data. They observe failure of an imputation method, and investigation of the corresponding data reveals that it occurs when outcome or exposure is highly imbalanced for individuals with an observed confounding variable. From this information, they conclude that *"the data-generating mechanism is too extreme and should be changed to generate more outcome events"* [26].

If, given the scope of the study, certain data set characteristics are deemed unrealistic for the targeted readers, authors may consider making sensible parameter changes in the DGM to generate data that is more realistic, in line with White et al. [26]. Similarly, "too extreme" data may also affect real data-based studies. If one of the (real) benchmark data sets is considered to be irrelevant to the scope of the study after careful examination, authors may consider removing the data set from the study completely. See Section 4.2 for an illustration. However, if the data sets causing failure are deemed relevant to the (simulation or benchmark) study's scope, the researchers conducting the comparison study should—more appropriately than discarding or imputing in most cases—proceed by considering how real-world users are likely to behave in this situation. In particular, they may modify certain parameters of the concerned method(s) (see Section 3.1.4) or resort to a fallback (see Section 3.1.5).

To be able to assess whether data structures associated with failure are too extreme, certain considerations should already be made in the planning stage. Researchers should be aware of the scope and goal of their study: Is it the goal to examine the general behavior of the method(s) (i.e., earlier stages of research) or to provide concrete recommendations for real-world practitioners (later stages of research)? Is the focus on testing the methods under regular or more extreme circumstances? If aiming to provide recommendations to real-world users, what degree of extremity in the data remains realistic for them?

3.1.4 | Be Realistic when Modifying Method Implementations

In earlier stages of research, it may make sense to modify a method's implementation entirely (i.e., for all simulation repetitions/benchmark data sets) when encountering method failure during execution. For instance, authors may conclude after execution that the default number of maximum likelihood iterations is generally too low, or that the given optimization algorithm may fail frequently, and that an alternative optimizer is more reliable. This way, authors may modify a greater variety of parameters to see when the method is successful and when it fails. Although this early-stage research is not yet directly aimed at real-world users, authors should remember that it will be used by them later. Therefore, certain practical limitations on parameter modifications should apply to ensure that the method remains feasible for future users.

Also, authors should be careful about increasing the runtimes of affected method(s). Especially, raising the number of iterations in optimization algorithms can increase runtime or memory consumption, potentially shifting method failure rather than solving it. A special case of modifying a method's implementation often only arises after the execution stage of the study: Investigations following the failure of a method reveal code bugs in the implementation of a method. Especially when their programming experience is high, authors may fix this bug and notify the method developers. Ultimately, the performance assessment of the given method is only meaningful if the authors stress and justify their parameter choices when reporting the study.

3.1.5 | Consider a Fallback Strategy

When a comparison study aims to provide guidance to real-world users (i.e., in later phases of research), it makes sense to mimic their behavior when encountering method failure in the execution stage. How would they proceed when their method of choice failed on their data set? Morris et al. [1] argue that they would usually not give up but naturally resort to an alternative method, a *fallback*, that successfully provides an output. Therefore, it can make sense to transfer this *fallback strategy* into the context of method comparison studies and evaluate the performance of *method pipelines* of the type

> Use method A if it successfully produces an output,
> use method B otherwise.

Incorporating fallbacks indirectly yields automatic handling of method failure in case they affect method A but not method B, allowing for the comparison of (absolute and relative) *unconditional* performance when chosen carefully and in a way that there is a suitable fallback for each instance of method failure. Note that when using fallback strategies, we no longer evaluate the "pure" performance of method A. However, it is clear from Section 2.2.1 that this is unattainable in the first place.

The choice of a suitable fallback, again, depends on the scope of the study and the targeted users. Should it serve as a guide for experienced or rather inexperienced users? When targeting inexperienced real-world users, authors may choose a completely different method as a fallback. For instance, Zapf et al. [9] suggest from the results of their study that *"the approach by Frömke* et al. *(2022) could be used as a fallback strategy in case of non-convergence [of the overall best-performing method] because it always yields results"* [9], where the approach by Frömke et al. [27] is a non-iterative procedure, therefore never affected by non-convergence by design. When addressing experienced users, an alternative implementation of the given method (in variation of Section 3.1.4) may make a suitable fallback.

Many considerations and decisions regarding fallback strategies should already be made in the planning stage of a study. This includes, first and foremost, whether using fallbacks is suitable for the given study, and based on this decision and the scope of the study, whether a completely new method or a modified implementation should be used as a fallback. Authors should further be aware that the neutrality principle of late phase comparison studies should always apply; that is, authors should avoid putting considerably more effort into only a few methods when choosing fallbacks for them. This becomes clear, for instance, when modifying the implementation of a method, as, for example, parameter modifications can strongly influence performance [28]. In the same context, it is important to avoid selective reporting as it can lead to over-optimistic results; that is, authors should disclose *all* tested pipelines, not just the best ones. It is therefore advisable to reduce the data-driven construction and the complexity of pipelines, which can be achieved by the following considerations. First, authors should decide on a maximum number of fallbacks that can be considered to realistically reflect the behavior of real users (as they are unlikely to try out an indefinite number of methods). Suitable combinations of methods and fallbacks may be preliminarily constructed based on the similarity of ease of use (e.g., a user is unlikely to resort to a fallback that is considerably more difficult to apply than the original method). Alternatively, inspired by Zapf et al. [9], it may make sense to define a default fallback to all instances of method failure if a method has proven to be particularly computationally robust in previous comparison studies.

Finally, authors should be aware that instances of all fallbacks failing on a given data set cannot always be avoided, even when using a computationally robust "default" fallback. Many fallbacks (and generally many methods) failing on the same data set may indicate that it is particularly "challenging." Authors should then investigate the data set closely to check whether it suits the scope of the study (see Section 3.1.3) and consider removing it if it is deemed unrealistic for real users. Otherwise, reporting the corresponding characteristics of the data set alongside the methods that successfully produce an output may provide valuable insights to real users.

3.1.6 | Special Case: Runtime Issues

In contrast to non-convergence and memory issues, method failure *is actively declared* by the authors when runtime issues occur in the execution stage. That is, running the methods under investigation on hundreds or thousands of data sets requires time restrictions, which are set by the authors arbitrarily, and without which the method would likely produce a valid output. On the other hand, it can sometimes be safely said that real (targeted) users, running the method on a single data set only, are likely to obtain an output. In these cases, when method failure leads to *missing* rather than *undefined* performance values, it may make sense to run the experiment on a randomly selected subset of all simulation repetitions/benchmark data sets for the method affected by runtime issues ("failure by design"). To ensure that this "failure" is independent of data characteristics, it is important to select the subset of data independently of the data sets for which the method has particularly long runtimes. Researchers should therefore think about possible runtime problems and carry out the corresponding random selection of data sets already in the planning stage.

3.1.7 | Report Failure Proportions, Underlying Interplays, Applied Approaches, and Share Code

Authors of comparison studies should *report any occurrences and proportions of method failure*. This can serve as an important criterion for real-world users selecting a suitable method. Even the "best performing" method is of limited use if the number of settings in which it can be applied without frequent failures is limited. Ideally, the proportion of method failure should be included as an additional performance measure for all considered methods (see Morris et al. [1]); see Zapf et al. [9] for an example.

Additionally, *the interplay of method and data causing failure should be reported*. Adding to the previous paragraph, a corresponding reporting for each method can look like "method A, with a failure proportion of $X\%$, fails when a predictor variable is constant across all observations and when a binary predictor has highly imbalanced classes." In addition to reporting the corresponding failure proportions, readers can learn in *which* settings a method is applicable, and no less importantly, in which it is *not*. Additionally, these insights serve as justification for the chosen handling of method failure. An example of a corresponding reporting can be found in Hornung et al. [11], who provide detailed descriptions in the supplement of their paper.

Authors of comparison studies should additionally *report their applied handlings* with sufficient detail, that is, any actions resulting from the above-described in-depth investigations and recommendations regarding method failure. This includes removing any misused methods, removing data (or modifying DGMs) that do not match the scope of the study, any changes in implementations, and incorporating fallback strategies. Additionally, authors should justify their choices given the scope of the study (i.e., which users it is aimed at) and the research stage in which it takes place.

The complexity of the occurrence and handling of method failure, which becomes clear from the previous sections, emphasizes all the more the value of *sharing code, data, and intermediate results*. Allowing authors of other, "new" comparison studies to conduct in-depth investigations of the specific instances of method failure in the given study may help them gain a greater understanding of this topic—and enhance fruitful discussions on suitable handlings.

3.1.8 | Finally: Think Ahead and Pre-Register, but Remain Flexible

The previous examples and those that will be outlined in Section 4 make clear that method failure and its cause(s), related to complex dynamics between factors such as method and data set, are often difficult to anticipate. While pre-specifying the handling of undefined values has the advantage of reducing analytical uncertainty and its dangers in terms of selective reporting [3], pre-specifying a standard procedure before conducting the study may sometimes be delicate. Making method failure less unexpected (and pre-specifying a procedure for handling method failure less hazardous) can be achieved through pilot experiments. They test whether an experimental design is appropriate for the specific context in which it is applied, and are recommended in several textbooks on *Design and Analysis of Experiments* [29–31]. For example, a pilot experiment might use a fraction of the available data sets (e.g., 20%) in the comparison study to test whether the specified method implementations produce reasonable results. The remaining 80% is used for the main experiment, which may be refined based on what is learned in the pilot. Pilot studies can also help authors anticipate possible method failures for which they can then pre-specify adequate handlings. Note that it is important to clearly report the results of the pilot study.

Despite pre-registration, researchers should be aware that unforeseen issues may appear and require adaptations of the pre-specified strategy (see Lakens [32] for a discussion of the consequences of deviations from pre-registration on the severity and validity of inferences). As a result, error-handling mechanisms should be included in the code right from the beginning, including the storage of error messages [26].

3.2 | Slimmed-Down Recommendations for the Busy Researcher (Looking to Do Just Enough)

For some authors, there may be time constraints that do not allow for any (in-depth) implementations of our recommendations from the previous Section 3.1; however, authors nevertheless wish to aggregate results over data sets despite the existence of undefined values caused by method failure. While we recommend implementing our in-depth recommendations as comprehensively as possible, we suggest the following slimmed-down analysis and reporting strategy when time and resources are limited. Authors may present performance results after (i) discarding the corresponding data sets *for the failing methods*, (ii) discarding these data sets *for all methods*, and (iii) *report the failure proportions* for all methods, ideally as an additional performance measure. This three-fold analysis and reporting enables more differentiated assessments of the methods' performances and avoid the preference of methods based solely on how data sets associated with method failure are discarded. It also reveals how successful methods perform on data sets for which others fail (and how this influences the method comparison). When the graphical comparison involves boxplots of performance values, authors could indicate method failure (and the corresponding frequency) as scatters of contrasting color above or below the box. For instance, for bias, they could be indicated above the plot (as a value higher than all biases observed in the study), whereas for accuracy, it could be indicated below the plot.

However, following Section 2.2.1, authors should be aware and clearly state in their manuscript that any form of aggregating performance results *despite* existing undefined values does not allow an assessment of *unconditional* performance of all methods under investigation.

4 | Illustrations

In the following, we illustrate the guidelines presented in Section 3 through two fictive comparison studies featuring method failure: A statistical simulation study and a benchmark study at the interface between statistics and predictive modeling.

Note that our focus is exclusively on the handling of method failure and its consequences on the method comparison rather than the method comparison itself. This means that both illustrations represent only a small excerpt (e.g., a few or a single scenario) of what would be a typical publishable study, exclusively focusing on those aspects of the studies that are relevant to the issue of method failure. Importantly, our goal is *not* to provide insights into the actual performance of the compared methods.

4.1 | Example 1: Comparing Methods for the Estimation of Odds Ratios

4.1.1 | General

This fictive statistical simulation study deals with the comparison of odds ratio (OR) estimation methods. Outcome and exposure are binary ($X, Y \in \{0, 1\}$) and the observed number of subjects with $X = i$ and $Y = j$ is denoted as n_{ij}. The simulation scenarios are defined by the sample size *n.obs*, the underlying true OR, and the probability of exposure p_x. Especially when the sample size is low, the underlying true OR is far from 1, or p_x is close to 0 or 1, some simulated data sets may not contain any subjects for one or more of the four outcome-exposure combinations ($n_{ij} = 0$). These "sampling zeros" cause calculation issues (e.g., division by 0) for many OR estimation methods, leading to either returning an error or, depending on the position of the sampling zero in the contingency table, non-meaningful OR estimations of 0 or "∞" for the corresponding data sets. In practice, users often apply the "Haldane–Anscombe correction" [33, 34] to handle sampling zeros in the OR estimation. This correction consists of adding an offset of +0.5 to all four cells in the contingency table to avoid issues in computation.

4.1.2 | Design of the Fictive Study

We vary the underlying *true* OR ($OR \in \{2, 5\}$) and exposure probability p_x ($p_x \in \{0.25, 0.5\}$) while keeping the sample size fixed at the relatively low value of $n.obs = 50$. This results in four distinct simulation scenarios (see Table 5 for an overview). For each scenario, we simulate 100 000 outcome-exposure data sets. Five OR estimation methods are applied to the simulated data to obtain a point estimate for the true OR. "Manual" corresponds to the manual and straightforward computation $\widehat{OR} = \frac{n_{11} \cdot n_{00}}{n_{01} \cdot n_{10}}$. The remaining four methods, namely "Fisher," "Midp," "Small" (all three implemented in R package epitools [35]), and "Woolf" [36] (implemented in R package pairwiseCI [37]) employ more complex modeling strategies. From the five estimators under investigation, only Small and Woolf can handle sampling zeros internally. For each simulation scenario, the estimators are compared regarding bias on the log-scale (log-transformation ensuring symmetry around the value of "no association").

Our analysis consists of two parts. First, we compare the performance of the OR estimation methods when *discarding data sets* with sampling zeros for *all* methods versus only for the *failing* methods ("ad hoc approaches"; Section 4.1.3). Though real users often apply the Haldane–Anscombe correction when encountering sampling zeros, this first part illustrates how the preference of some methods may mainly be driven by the way data sets with sampling zeros are discarded.

Second, we repeat the analysis implementing the principle of fallback strategies suggested in Section 3.1.5. This includes important considerations that must be made when choosing suitable

TABLE 5 | Overview of the proportion of the 100 000 simulated data sets (fixed sample size $n.obs = 50$) with sampling zeros across the simulation scenarios. p_x: Exposure probability.

Simulation scenario	*true* OR	p_x	Sampling zero proportion
1	2	0.25	1.28%
2	5	0.25	12.0%
3	2	0.5	0.01%
4	5	0.5	1.27%

fallbacks. For instance, for the manual estimation of the OR (method "Manual"), only the Haldane–Anscombe correction is applied as a fallback, assuming that a user applying the manual computation in the first place, which is very easy to implement, is unlikely to resort to a method that requires notably more effort. On the other hand, for methods Midp and Fisher for data sets with sampling zeros, we either (i) apply the Haldane–Anscombe correction, (ii) use method Small, or (iii) use method Woolf, assuming that corresponding users may also employ fallbacks that are slightly more difficult to apply. Together with Woolf and Small, this leads to nine "methods" ("pure" methods or pipelines) being compared in total. The results are presented in Section 4.1.4.

4.1.3 | Occurrence of Method Failure and Ad Hoc Handlings

Table 5 displays the proportion of the 100 000 simulated data sets containing at least one sampling zero across the four simulation scenarios. As expected, this proportion increases with an increasing true OR > 1, while decreasing when exposure X is more balanced.

Ad hoc approaches In the first analysis, Figure 1A demonstrates that the way of discarding data sets with sampling zeros can substantially affect the results of the method comparison, with discrepancies regarding performance ranks already observable at a sampling zero proportion of just over 1%. This impact becomes more pronounced as the proportion of sampling zeros increases, making a difference in the observed performance by up to three ranks. In particular, it can affect which method is perceived as best-performing. Note that a more detailed investigation of Woolf's performance shows that for the data sets *without* sampling zeros, Woolf slightly underestimates the true OR, while for the few data sets *with* sampling zeros, Woolf notably overestimates it. This leads to top performance when all data sets are included in its performance assessment (while discarding data sets with sampling zeros for the failing methods).

4.1.4 | Recommended Handling: Using Fallbacks

Woolf's overall good performance across all scenarios is underpinned when using fallbacks for all methods affected by method failure (see Figure 1B). Furthermore, the comparison between Woolf and Midp, another method performing well in most scenarios, becomes clearer through the use of fallbacks. While they often alternate in rank (and especially between rank 1 and 2) when removing data sets (Figure 1A), their performance comparison is constant across all scenarios when a fallback is applied to Midp (Figure 1B). Since this applies to both fallbacks, the Haldane–Anscombe correction and Small, it is debatable which fallback is preferable. Both fallbacks for Midp frequently switch ranking positions across the performance measures and simulation scenarios; however, fallback Small might be even easier to

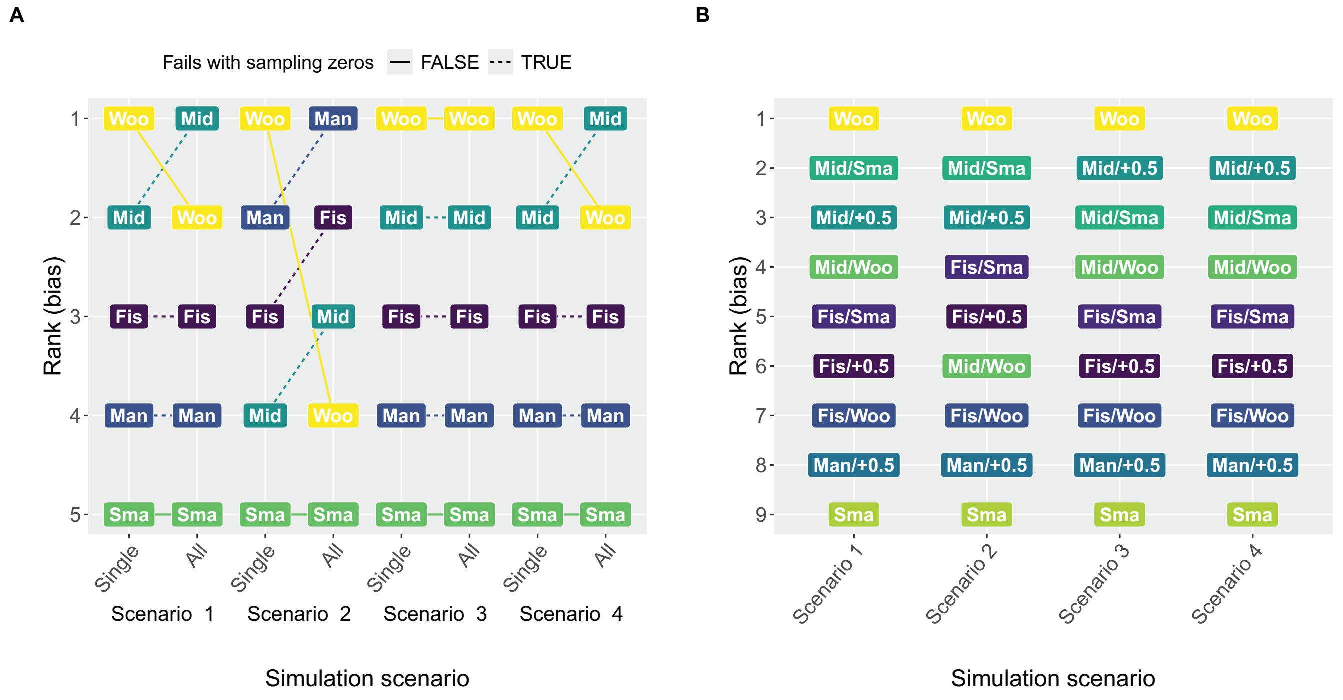

FIGURE 1 | Performance ranks of the OR estimation methods across the eight simulation scenarios based on empirical bias on the log-scale, where a rank of 1 means that the method has the lowest empirical bias in the given scenario. **(A)** Ranks are compared between **discarding data sets** for the individual failing methods only (**"Single"** on the x-axis) versus for all methods (**"All"** on the x-axis). **(B)** The ranks are compared after **applying fallback strategies** to methods Midp, Fisher, and Manual. For Midp and Small, the fallbacks Small, Woolf, and the Haldane–Anscombe correction ("+0.5") are used. For Manual, only the Haldane–Anscombe correction is used as a fallback. Small and Woolf can deal with sampling zeros internally. Abbreviation of methods: "Fis": Fisher; "Mid": Midp; "Sma": Small; "Woo": Woolf; "Man": Manual (i.e., manual estimation of the OR). **(B)**: The combination of original method m_1 and fallback m_2 is marked by "m_1/m_2."

implement than the Haldane–Anscombe correction, requiring only a simple code modification from Midp.

4.2 | Example 2: Comparing Methods for Computing Confidence Intervals for the Generalization Error

4.2.1 | General

The fictive study aims to compare the performance of two methods for constructing confidence intervals (CIs) for the generalization performance of a prediction model based on resampling-based estimates, such as cross-validation. The "naive" method "N" for constructing the CIs ignores the dependence between the generalization performances in the resampling repetitions, while method "C" corrects for it. The performance of both methods is assessed in terms of *coverage*. Note that a comprehensive benchmark study of methods for constructing CIs for the generalization error is performed by Schulz-Kümpel et al. [38].

4.2.2 | Design of the Fictive Study

Classification trees [39] are considered as models for predictive modeling, and the data set provided by the SAPA project [40], further processed according to Klau et al. [41], is used as the benchmark data set. The response variable is binary. The generalization performance is quantified using the Area Under the Curve (AUC). The AUC is estimated using 15 repetitions of repeated subsampling ($i = 1, \ldots, 15$) with a 4:1 ratio (i.e., the data is randomly split 15 times into 80% on which the model is trained and 20% on which the AUC is obtained [42]).

Method C suggested by Nadeau and Bengio [43] computes the CI for the generalization AUC as

$$\left[\overline{\widehat{AUC}} - t_{0.975,14} \cdot \sqrt{\left(\frac{1}{15} + c\right) \cdot S^2_{\widehat{AUC_i}}}, \right. \\ \left. \overline{\widehat{AUC}} + t_{0.975,14} \cdot \sqrt{\left(\frac{1}{15} + c\right) \cdot S^2_{\widehat{AUC_i}}} \right] \tag{1}$$

where $\overline{\widehat{AUC}}$ represents the mean AUC estimate and $S^2_{\widehat{AUC_i}}$ the sample variance over the 15 subsampling repetitions. $t_{0.975,14}$ is the corresponding quantile of the t-distribution. A fixed correction term $c > 0$ incorporates the correlation structure between the subsampling repetitions for method C (see Nadeau and Bengio [43] for the construction of c). Method N, on the other hand, sets $c = 0$ in the above formula.

To estimate the coverages of methods N and C, the following procedure is repeated for 1000 iterations (see Figure 2). The SAPA data set is randomly split into a large test set D_{test} (80% of observations) and a training set D_{train} (20% of observations). The *true* AUC is approximated by applying the trained model to D_{test}. The corresponding *estimated* AUC, on the other hand, is obtained via repeated subsampling on D_{train}. Confidence intervals for the true AUC are constructed according to Equation (1) with corresponding values of c for methods N and C. Finally, the estimated coverages of methods N and C are the proportions of the 1000 iterations where the respective confidence intervals cover the approximated true AUC.

The study is conducted in `R` using package `mlr3`, method N is implemented using function `t.test()` from package `base` and for method C, $\overline{\widehat{AUC}}$ and $S^2_{\widehat{AUC_i}}$ are obtained using functions `mean()` and `sd()` (from the same package).

4.2.3 | Occurrence of Method Failure and Ad Hoc Handlings

For method N, the implementation of the above-described procedure returns the following error message for 300 of the 1000 iterations in `R`:

```
Error in t.test.default: Data are
essentially constant.
```

Hence, the confidence intervals cannot be constructed for method N in these iterations, making it impossible to assess whether they cover the true AUC. Method C is not affected by these errors. At first glance, the error message gives little indication of how to resolve it. In the following, we first apply three ad hoc approaches to handle these errors. Then, by following our in-depth guidelines from Section 3, we demonstrate how these ad hoc approaches prove to be inadequate, and illustrate more suitable ways to handle method failure in this example.

Ad hoc approaches One could be inclined to apply one of the following ad hoc approaches:

1. For method N, ignore all data sets in which it fails (method C is unaffected).
2. For both methods, ignore those iterations in which N fails.
3. For method N, set "undefined" confidence intervals to "not covering the true AUC."

Note that the intuition to approach 3, which can be viewed as a form of imputation, is that a CI that does not exist cannot cover any value. It thus induces a negative correlation between the proportion of undefined confidence intervals and the coverage of method N.

The coverages resulting from the ad hoc approaches are displayed in Table 6a. While the coverage of method C is constant across all three ad hoc approaches, method N, while too liberal across all approaches, is even more liberal using the ad hoc imputation approach.

4.2.4 | Recommended Handling

We now closely inspect the modeling processes underlying the study to find the root cause of the error message and to adapt the handling of method failure accordingly. Investigations show that the errors in method N occur when the estimated AUCs are

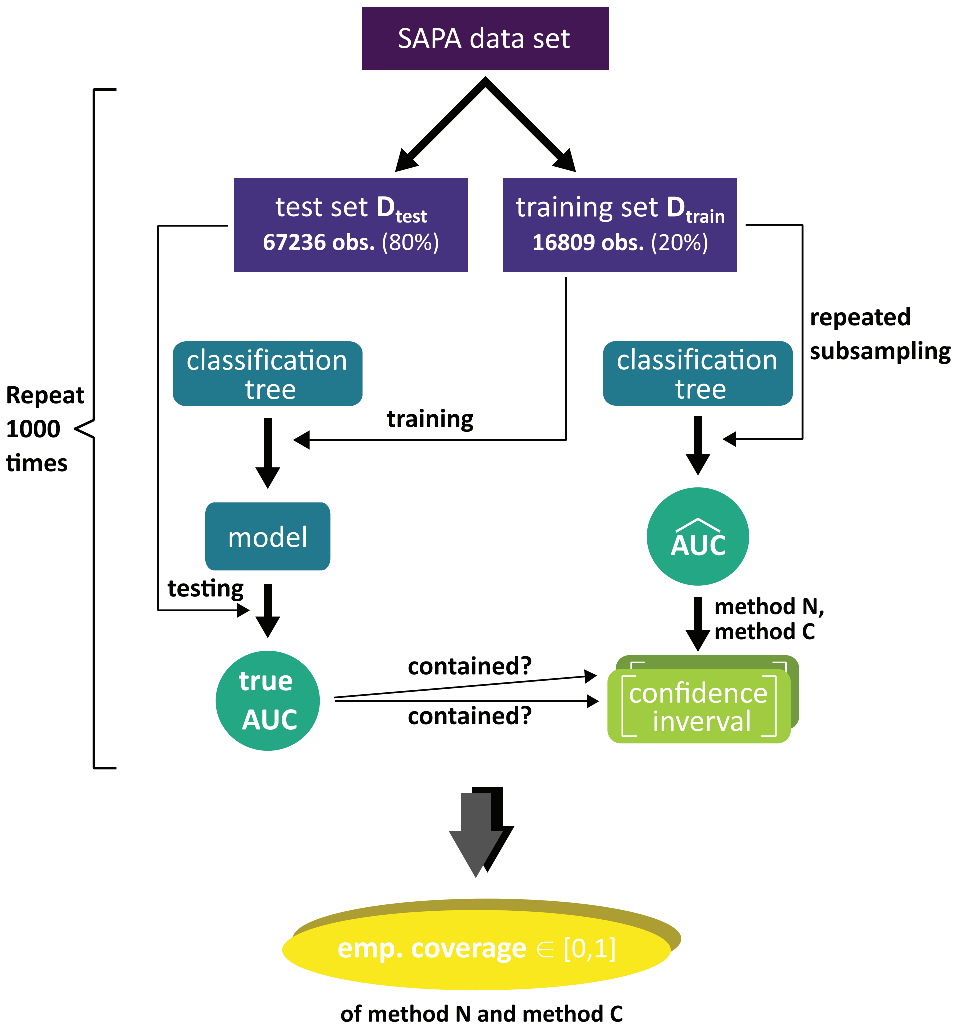

FIGURE 2 | Design of the fictive study to obtain the empirical coverage of methods C and N. An approximation of the "true AUC" is obtained based on the very large D_{test}.

TABLE 6 | Empirical coverages of methods N and C resulting for (a) the three "ad hoc" approaches and (b) setting undefined CIs to zero-width CIs.

	(a) Ad hoc approaches: NA-CIs are			**(b) NA-CIs are**
	discarded for method *N* only	**discarded for methods *N* and *C***	**set to "not covering true AUC"**	**set to zero-width interval**
Method N	0.54	0.54	0.38	0.68
Method C	1	1	1	1

equal across all 15 subsampling repetitions, resulting in a sample variance $S^2_{\widehat{AUC}i}$ of 0. This zero variance in the denominator of the t-statistic causes an error in the function `t.test()`. Focusing on the confidence interval, however, a sample variance of 0 simply implies a zero-width CI $\{\widehat{AUC}\}$. Therefore, method N not yielding a CI (and returning an error) is a technical artifact of its implementation. A straightforward solution would be to modify the implementation of method N, using `mean()` and `sd()` instead of `t.test()`, so that it computes the zero-width CI, which can enter the calculation of coverage as any other CI (see Table 6b for the resulting coverage).

However, upon further investigations, it becomes clear that a sample variance $S^2_{\widehat{AUC}i} = 0$ always coincides with all 15 estimated (and the approximate true) AUC being 0.5. Ultimately, this worst possible performance of the classification tree ("random prediction") occurs when the classification tree does not perform any splitting. This is related to the characteristics of the data set. For example, the problem disappears if we include additional predictors: Splitting then occurs, leading to an AUC higher than 0.5.

This illustrates well that the further the investigations are taken, the apparent adequate handling of method failure can change notably. In this case, based on the information gained, adequate handling depends on the aim of the study and requires further considerations, such as the following. A prediction problem where splitting often does not occur can be viewed as an edge case. If the study's scope includes such edge cases, one could attempt to suitably modify the parameters of the classification tree to favor splitting. If, however, the study aims to compare both methods in regular scenarios, one may add additional predictors to favor splitting more strongly, resort to an alternative prediction model (e.g., logistic regression) or an alternative measure of generalization performance (e.g., accuracy), or use a different benchmark data set.

5 | Discussion

Method failure in comparison studies, often manifesting as issues regarding calculation (e.g., non-convergence), runtime, or memory issues, complicates the desired comparison of *unconditional* performance between the methods under investigation, that is, across all simulated or benchmark data sets. Our informal literature review and illustrations both make clear that authors should not blindly rely on the initially perceived manifestation of failure, but rather make extensive investigations of the underlying factors causing the failure, which often guides the choice of handling the resulting undefined values.

Even if we argue for decisions on a case-by-case basis after careful inspection of the failures, we show and illustrate that some general principles hold in most situations. In particular, method failure usually leads to non-existing values instead of just "missing," that is, existent but unobserved values. This means imputation, a common but missing-data-inspired approach to handling method failure, is usually inadequate. The same applies to the popular and seemingly logical approach of discarding data associated with failure (either for all or the failing methods), which almost never enables the comparison of unconditional performance. In this work, we provide alternative approaches that might be more appropriate to most occurrences of method failure.

An approach that in our view deserves more attention than it receives currently in the literature consists of evaluating whole "pipelines" of methods, including one or several fallbacks in case of method failure. This approach reflects the typical behavior of most users of methods in practice, while enabling the comparison of unconditional performance if constructed carefully: If one method does not work, they resort to an alternative one. Future work is required to collect experiences and define standards regarding such pipelines and their evaluation. Above all, transparent reporting of the frequency of failure and of the corresponding handling is always recommended.

A limitation of our recommendations is that, beyond obvious cases, they typically require both high time resources and thorough expertise in the underlying modeling processes and methods. Researchers conducting comparison studies may not always have these resources to the full extent. This particularly applies to a nuanced examination of the interactions between methods and data sets, leading to method failure in different instances. We understand that this bottleneck may impair the full implementation of our recommendations in practice, hence our attempt to provide a slimmed-down version of the recommendations. When time, resources, and expertise are restricted, transparent reporting is all the more important. Furthermore, this underlines the necessity for openly sharing code, data, and ideally intermediate results, as this allows other researchers to explore the observed instances of method failure in greater detail. Also, our viewpoint on method failure as leading to "undefined" performance values does not always hold unrestrictedly, which can complicate the implementation of our recommendations. However, with method failure being a complex matter in and of itself, this underscores the importance of a case-by-case evaluation, as described above.

In this article, method failure has been treated as black-or-white. However, in practice, whether or not a method has failed is itself a value of judgment. One such example is near-separation [7], which does not necessarily make itself obvious, hence the existence of dedicated methods to detect it [44]. Such gray-area failures add further complexities to benchmark and simulation studies: Overall performance may depend on the specific criteria used to judge failure.

Partial failure of a method is also possible. For example, in a simulation study, a method might return a valid point estimate but not its standard error, or a point estimate for one estimand but not another. Many of the principles discussed in this article apply in such situations, for example, the principle to avoid imputing estimates where a method has failed, and the notion of a pipeline (when the standard error estimator fails, revert to a backup estimator if available). There are, however, further subtleties. For example, it may be inadvisable for the fallback approach to keep the point estimate from method A and fallback on the standard error estimated from method B.

To sum up, we believe there is not "one" right way to handle method failure in all situations. Instead, it is important to carefully choose an approach depending on the context, and then report, explain (and ideally discuss) the decisions made. With our work, we hope to contribute meaningful ideas on how to think about method failure. They lead to helpful strategies for dealing with it, as outlined as part of our recommendations. However, we also believe that method failure is only one of many issues related to comparison studies that deserve much more time, effort, and methodological strength towards more reliable evidence on the behavior of methods in the context of a "replication crisis in methodological research" [45]. We need more well-designed neutral comparison studies [4]—late phase studies according to the terminology of Heinze et al. [24]. In such studies, no time is spent on the development of new methods. This leaves time for implementing our recommendations and diving into the technical details of the methods under investigation.

Acknowledgments

The authors thank Julian Lange for useful literature input and Clemens Kreutz for valuable comments. The authors further thank Savanna Ratky for language corrections and Luzia Hanßum for support with formatting the manuscript. We acknowledge the use of the large language model GPT-4o (OpenAI) in the preparation of this manuscript. It was consulted exclusively for linguistic refinement and copy-editing of text originally written by the authors; no content was generated by the model. Open Access funding enabled and organized by Projekt DEAL.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data that support the findings of this study are openly available in Zenodo at https://doi.org/10.5281/zenodo.15020209.

References

1. T. P. Morris, I. R. White, and M. J. Crowther, "Using Simulation Studies to Evaluate Statistical Methods," *Statistics in Medicine* 38, no. 11 (2019): 2074–2102.

2. S. Pawel, L. Kook, and K. Reeve, "Pitfalls and Potentials in Simulation Studies: Questionable Research Practices in Comparative Simulation Studies Allow for Spurious Claims of Superiority of any Method," *Biometrical Journal* 66, no. 1 (2024): 2200091.

3. B. S. Siepe, F. Bartoš, T. P. Morris, A. L. Boulesteix, D. W. Heck, and S. Pawel, "Simulation Studies for Methodological Research in Psychology: A Standardized Template for Planning, Preregistration, and Reporting," *Psychological Methods* (2024), https://doi.org/10.1037/met0000695.

4. A. L. Boulesteix, R. Wilson, and A. Hapfelmeier, "Towards Evidence-Based Computational Statistics: Lessons From Clinical Research on the Role and Design of Real-Data Benchmark Studies," *BMC Medical Research Methodology* 17 (2017): 138.

5. S. Friedrich and T. Friede, "On the Role of Benchmarking Data Sets and Simulations in Method Comparison Studies," *Biometrical Journal* 66, no. 1 (2024): 2200212.

6. A. M. Hinds, T. T. Sajobi, V. Sebille, R. Sawatzky, and L. M. Lix, "A Systematic Review of the Quality of Reporting of Simulation Studies About Methods for the Analysis of Complex Longitudinal Patient-Reported Outcomes Data," *Quality of Life Research* 27 (2018): 2507–2516.

7. v M. Smeden, d J A. Groot, K. G. Moons, et al., "No Rationale for 1 Variable per 10 Events Criterion for Binary Logistic Regression Analysis," *BMC Medical Research Methodology* 16 (2016): 163.

8. P. Gijsbers, M. L. Bueno, S. Coors, et al., "AMLB: An AutoML Benchmark," *Journal of Machine Learning Research* 25, no. 101 (2024): 1–65.

9. A. Zapf, C. Frömke, J. Hardt, G. Rücker, D. Voeltz, and A. Hoyer, "Meta-Analysis of Diagnostic Accuracy Studies With Multiple Thresholds: Comparison of Approaches in a Simulation Study," *Biometrical Journal* 66, no. 7 (2024): e202300101.

10. G. D. Ruxton and M. Neuhäuser, "Review of Alternative Approaches to Calculation of a Confidence Interval for the Odds Ratio of a 2×2 Contingency Table," *Methods in Ecology and Evolution* 4, no. 1 (2013): 9–13.

11. R. Hornung, F. Ludwigs, J. Hagenberg, and A. L. Boulesteix, "Prediction Approaches for Partly Missing Multi-Omics Covariate Data: A Literature Review and an Empirical Comparison Study," *Wiley Interdisciplinary Reviews: Computational Statistics* 16, no. 1 (2024): e1626.

12. E. Masaoud and H. Stryhn, "A Simulation Study to Assess Statistical Methods for Binary Repeated Measures Data," *Preventive Veterinary Medicine* 93, no. 2–3 (2010): 81–97.

13. M. J. Crowther, M. P. Look, and R. D. Riley, "Multilevel Mixed Effects Parametric Survival Models Using Adaptive Gauss–Hermite Quadrature With Application to Recurrent Events and Individual Participant Data Meta-Analysis," *Statistics in Medicine* 33, no. 22 (2014): 3844–3858, https://doi.org/10.1002/sim.6191.

14. Z. S. Dunias, B. Van Calster, D. Timmerman, A. L. Boulesteix, and v M. Smeden, "A Comparison of Hyperparameter Tuning Procedures for Clinical Prediction Models: A Simulation Study," *Statistics in Medicine* 43, no. 6 (2024): 1119–1134.

15. M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do We Need Hundreds of Classifiers to Solve Real World Classification Problems?," *Journal of Machine Learning Research* 15, no. 1 (2014): 3133–3181.

16. M. Herrmann, P. Probst, R. Hornung, V. Jurinovic, and A. L. Boulesteix, "Large-Scale Benchmark Study of Survival Prediction Methods Using Multi-Omics Data," *Briefings in Bioinformatics* 22, no. 3 (2021): bbaa167.

17. B. Bischl, J. Schiffner, and C. Weihs, "Benchmarking Local Classification Methods," *Computational Statistics* 28, no. 6 (2013): 2599–2619.

18. C. Nießl, M. Herrmann, C. Wiedemann, G. Casalicchio, and A. L. Boulesteix, "Over-Optimism in Benchmark Studies and the Multiplicity of Design and Analysis Options When Interpreting Their Results," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 12, no. 2 (2022): e1441.

19. S. Pawel, F. Bartoš, B. S. Siepe, and A. Lohmann, "Handling Missingness, Failures, and Non-Convergence in Simulation Studies: A Review of Current Practices and Recommendations," arXiv preprint arXiv:2409.18527 (2024).

20. D. B. Rubin, *Multiple Imputation for Nonresponse in Surveys*, vol. 81 (John Wiley & Sons, 2004).

21. J. R. Carpenter, J. W. Bartlett, T. P. Morris, A. M. Wood, M. Quartagno, and M. G. Kenward, *Multiple Imputation and Its Application* (John Wiley & Sons, 2023).

22. B. F. Kurland, L. L. Johnson, B. L. Egleston, and P. H. Diehr, "Longitudinal Data With Follow-Up Truncated by Death: Match the Analysis Method to Research Aims," *Statistical Science* 24, no. 2 (2009): 211, https://doi.org/10.1214/09-sts293.

23. B. C. Kahan, J. Hindley, M. Edwards, S. Cro, and T. P. Morris, "The Estimands Framework: A Primer on the ICH E9(R1) Addendum," *BMJ (Clinical Research Ed.)* 384 (2024): e076316, https://doi.org/10.1136/bmj-2023-076316.

24. G. Heinze, A. L. Boulesteix, M. Kammer, T. P. Morris, I. R. White, and STRATOS Initiative o. tSP, "Phases of Methodological Research in Biostatistics—Building the Evidence Base for New Methods," *Biometrical Journal* 66, no. 1 (2024): 2200222.

25. A. Zimmermann, "Method Evaluation, Parameterization, and Result Validation in Unsupervised Data Mining: A Critical Survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 10, no. 2 (2020): e1330.

26. I. R. White, T. M. Pham, M. Quartagno, and T. P. Morris, "How to Check a Simulation Study," *International Journal of Epidemiology* 53, no. 1 (2024): dyad134.

27. C. Frömke, M. Kirstein, and A. Zapf, "A Semiparametric Approach for Meta-Analysis of Diagnostic Accuracy Studies With Multiple Cut-Offs," *Research Synthesis Methods* 13, no. 5 (2022): 612–621.

28. L. M. Weber, W. Saelens, R. Cannoodt, et al., "Essential Guidelines for Computational Method Benchmarking," *Genome Biology* 20 (2019): 1–12.

29. P. R. Cohen, *Empirical Methods for Artificial Intelligence* (MIT Press, 1995).

30. A. Dean, D. Voss, and D. Draguljić, *Design and Analysis of Experiments*, 2nd ed. (Springer, 2017).

31. D. C. Montgomery, *Design and Analysis of Experiments*, 10th ed. (John Wiley & Sons, Inc, 2020).

32. D. Lakens, "When and How to Deviate From a Preregistration," *Collabra: Psychology* 10, no. 1 (2024): 117094.

33. J. Haldane, “The Estimation and Significance of the Logarithm of a Ratio of Frequencies,” *Annals of Human Genetics* 20, no. 4 (1956): 309–311.

34. F. J. Anscombe, “On Estimating Binomial Response Relations,” *Biometrika* 43, no. 3/4 (1956): 461–464.

35. T. J. Aragon, M. P. Fay, D. Wollschlaeger, and A. Omidpanah, “Epitools: Epidemiology Tools,” R package version 0.5–10.1 (2020), https://doi.org/10.32614/CRAN.package.epitools.

36. B. Woolf, “On Estimating the Relation Between Blood Group and Disease,” *Annals of Human Genetics* 19, no. 4 (1955): 251–253.

37. F. Schaarschmidt and D. Gerhard, “pairwiseCI: Confidence Intervals for Two Sample Comparisons,” R package version 0.1–27 (2019), https://doi.org/10.32614/CRAN.package.pairwiseCI.

38. H. Schulz-Kümpel, S. Fischer, R. Hornung, A. L. Boulesteix, T. Nagler, and B. Bischl, “Constructing Confidence Intervals For’the’generalization Error–a Comprehensive Benchmark Study,” arXiv preprint arXiv:2409.18836 (2024).

39. L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees* (Chapman and Hall/CRC, 1984).

40. D. M. Condon, E. Roney, and W. Revelle, “A SAPA Project Update: On the Structure of Phrased Self-Report Personality Items,” *Journal of Open Psychology Data* 5 (2017): 3.

41. S. Klau, F. D. Schönbrodt, C. J. Patel, J. P. Ioannidis, A. L. Boulesteix, and S. Hoffmann, “Comparing the Vibration of Effects due to Model, Data Pre-Processing and Sampling Uncertainty on a Large Data Set in Personality Psychology,” *Meta-Psychology* 7 (2023): MP.2020.2556.

42. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. (Springer, 2009).

43. C. Nadeau and Y. Bengio, “Inference for the Generalization Error,” *Machine Learning* 52, no. 3 (2003): 239–281.

44. M. A. Mansournia, A. Geroldinger, S. Greenland, and G. Heinze, “Separation in Logistic Regression: Causes, Consequences, and Control,” *American Journal of Epidemiology* 187, no. 4 (2017): 864–870, https://doi.org/10.1093/aje/kwx299.

45. A. L. Boulesteix, S. Hoffmann, A. Charlton, and H. Seibold, “A Replication Crisis in Methodological Research?,” *Significance* 17, no. 5 (2020): 18–21.