Michel Philipp

Contributions to Machine Learning and Psychometrics

Computational, Graphical, and Statistical Methods for Assessing Stability

Cuvillier Verlag Göttingen
Internationaler wissenschaftlicher Fachverlag

Contributions to Machine Learning and Psychometrics

CONTRIBUTIONS TO MACHINE LEARNING AND PSYCHOMETRICS

Computational, Graphical, and Statistical Methods for Assessing Stability

Michel Philipp

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

1. Aufl. - Göttingen: Cuvillier, 2017
Zugl.: Zürich, Univ., Diss., 2016

This work was accepted as a PhD thesis by the Faculty of Arts and Social Sciences, University of Zurich in the fall semester 2016 on the recommendation of the Doctoral Committee: Prof. Dr. Carolin Strobl (main supervisor) and Prof. Dr. Klaus Oberauer.

Nonnenstieg 8, 37075 Göttingen
Telefon: 0551-54724-0
Telefax: 0551-54724-21
www.cuvillier.de

1. Auflage, 2017
Gedruckt auf umweltfreundlichem, säurefreiem Papier aus nachhaltiger Forstwirtschaft

ISBN 978-3-7369-9447-8
eISBN 978-3-7369-8447-9

Acknowledgments

My considerable gratitude goes to all those who supported me during my PhD studies.

First of all, I would like to extend my sincere thanks to my main supervisor Carolin Strobl for many valuable discussions, her expert advice and her tireless support. I am very grateful to her for giving me the opportunity to write this thesis and to gain valuable consulting and teaching experience during my PhD studies. I also want to thank Klaus Oberauer, the second member of the dissertation committee, for his involvement.

In addition, I would like to acknowledge the scientific support that I received from the following people. In particular, a special thanks goes to Achim Zeileis for his expert support and advice concerning R implementations and statistical methods in general, which have broadened my understanding of statistics and programming in many ways. Thanks also go to Jimmy de la Torre, Thomas Rusch, and Kurt Hornik for reading, commenting, and making suggestions for improving the manuscripts that form the basis of this thesis. I would also like to thank Reto Bürgin, Beate Sick, and Matthias Keller for reading and discussing parts of the thesis.

Thanks also go to René Locher, Andreas Ruckstuhl, and Werner Stahel who have motivated me to study data analysis and statistics and/or supervised my diploma and master thesis. I wish to thank Franziskus Liem, Maike Debus, Jeanine Grütter, Andreas Walther, and Albert Steiner for various collaborations over the past years. Also, I would like to thank my former and current coworkers at the research unit for Psychological Methods, Evaluation, and Statistics at the University of Zurich for the scientific exchange. A very special thank you goes to Nina Kramer for her ever present kindness.

Moreover, I would like to thank my fellow students Katja Machmutow, Andreas Küffer, and Marcel Niklaus for involving me in healthy distractions such as running, hiking, or mountaineering and the members of the MuSt Peer Mentoring Group for retreats and meetings with lively discussions about statistics.

Finally, my sincere thanks go to my family Thomas, Gabriela, Stefanie, and Nadine for their magnificent support and understanding during all my studies and of course to my girlfriend Martina Ramel for her love, support, and inspiration.

Abstract

This thesis contributes to the development of methods for assessing the stability of results from statistical data analysis. For applications in science, business, or industry stability is a general requirement to draw consistent conclusions. This is only the case if statistical results generated with slightly perturbed data as well as with different data sets drawn from the same data-generating process lead to the same interpretation. In psychometric modeling, the stability of item parameters across different subsets of the data is a particularly important requirement to ensure fair comparisons between individuals in educational or psychological tests.

The first article in this thesis suggests descriptive measures and graphical illustrations for a detailed stability assessment of the variable and cutpoint selection in recursive partitioning. The second article proposes a general framework for assessing and comparing the stability of results generated by supervised statistical learning algorithms. The third article discusses the estimation of standard errors in cognitive diagnosis models, a particular family of psychometric models. An additional chapter presents the unpublished results of a comparison of statistical tests to detect parameter instability in cognitive diagnosis models. All proposed methods are implemented in add-on packages for the free open source software system R for statistical computing.

Zusammenfassung

Die vorliegende Arbeit umfasst mehrere Forschungsbeiträge über die Entwicklung neuer Methoden zur Erhebung der Stabilität statistischer Datenanalysen, die in Forschung und Praxis durchgeführt werden. Stabilität ist eine wichtige Voraussetzung damit aus den Ergebnissen statistischer Datenanalysen konsistente Schlussfolgerungen gezogen werden können. Dies ist jedoch nur möglich, wenn Analysen, die auf leicht veränderten oder auf komplett unterschiedlichen Datensätzen vom selben datengenerierenden Prozess beruhen, zu vergleichbaren Interpretationen führen. Ausserdem ist Stabilität eine zentrale Eigenschaft vieler psychometrischer Modelle, um objektive und faire Vergleiche zwischen Personen zu gewährleisten.

Im ersten Beitrag dieser Arbeit werden deskriptive Statistiken und graphische Illustrationen zur detaillierten Stabilitätsanalyse der Variablen- und Bruchpunktselektion bei Entscheidungsbäumen vorgestellt. Im zweiten Beitrag wird ein generelles Framework zur Analyse und zum Vergleich der Stabilität von Ergebnissen, die durch maschinelles Lernen erzeugt werden, vorgeschlagen. Im dritten Beitrag wird die Schätzung von Standardfehlern bei Kognitiven Diagnosemodellen, einer neuen Familie psychometrischer Modelle, diskutiert. Im letzten Kapitel werden unveröffentlichte Resultate zur Erkennung von Parameterinstabilitäten präsentiert. Alle vorgeschlagenen Methoden sind als Zusatzpakete für die freie Open Source Software R verfügbar.

Contents

Scope of this work

Statistical data analysis carried out in industry, business, or scientific research are often concerned with generating an understanding about "how things really are" (Tukey, 1962). In the social sciences and other areas, for example, scientific findings are obtained by contrasting theoretical considerations (substantial theory) with reality (observed data in a study or an experiment) by using statistical methods. However, the conclusions and their implications are strongly based on the *stability* of the results from the statistical data analysis (Stodden, 2015; Turney, 1995; Yu, 2013). This thesis contributes to the development of statistical methods to investigate stability.

According to Stodden (2015), stability is a major requirement to ensure the reproducibility of scientific findings. This has previously been pointed out by influential statisticians. Yu (2013), for example, stated: "More often than not, modern scientific findings rely on statistical analysis of high-dimensional data, and reproducibility is imperative for any scientific discovery. Scientific reproducibility therefore is a responsibility of statisticians. At a minimum, reproducibility manifests itself in the stability of statistical results relative to 'reasonable' perturbations to data and to the method or model used." (p. 1485) and further gives a broad definition of statistical stability: "We say statistical stability holds if statistical conclusions are robust or stable to appropriate perturbations to data." (p. 1489).

To be concrete, Yu (2013) mentioned the bootstrap and subsampling as reasonable forms of data perturbation when the data units are i.i.d. In certain applications, however, stable statistical conclusions (i.e., invariance properties) are required between subsets of the data formed by given characteristics of the observational units (an example in psychological and educational testing is given below). Moreover, for the reproducibility of scientific findings in general, stable results are also expected between completely different data sets drawn from the same data-generating process (DGP).

Depending on the method used to analyze data, different aspects of stability may be relevant and various approaches may be used to investigate it. In a parametric model, for example, instability is revealed when the estimated parameters vary across different subsets of the data formed by known or unknown subgroups of observations. This is known as *parameter instability* in economics, as violations of *measurement invariance*

in psychological and educational measurement or more generally as *varying coefficients* in statistics.

The detection of parameter instability provides information about the heterogeneity in the data and may generate new knowledge that contributes to scientific findings. In some areas (e.g., in the social and behavioral sciences), however, the concept is used to detect violations of model assumptions. For example, when measuring latent traits such as ability, personality, or attitudes in a questionnaire or an assessment using psychometric models, a particular goal of studying parameter instability is the detection of *differential item functioning* (DIF; see, e.g., Holland & Wainer, 1993) – that is the identification of items that violate the general assumption of constant item parameters across different groups of test takers in psychometric models.

A common way to detect parameter instability in this setting is by using statistical inference procedures, such as likelihood ratio, Wald, or Lagrange multiplier tests (see, e.g., Merkle & Zeileis, 2013). These methods can be used to detect parameter instability with respect to two subgroups specified by a dichotomous variable (e.g., gender) that may also be formed by splitting a continuous variable at a known "cutpoint". In practice, however, the cutpoint is often unknown and therefore chosen arbitrary (e.g., age splitted at the median). One particular approach (that will again appear in Chapter 4) is to estimate the model parameters separately for the two subgroups and then test the null hypothesis of equal parameter values in both groups using the (multivariate) Wald statistic.

Thus, with these methods parameter instability can only be tested for the groups that are proposed explicitly by the researcher (Strobl, Kopf, & Zeileis, 2015). However, as Merkle and Zeileis (2013) stated: "[...], if the pattern is unknown, it is difficult to develop a single test that is well-suited for all conceivable patterns." (p. 61). Therefore, statistical inference procedures have been proposed that can also detect variation in the parameter values with respect to a numerical (e.g., time or age), an unordered (e.g. ethnic groups), or an ordered (e.g., educational degree levels) multicategorical variable without prior knowledge about the group structure that causes parameter instability (Merkle, Fan, & Zeileis, 2014; Merkle & Zeileis, 2013). These approaches are based on stochastic processes of the individual likelihood contributions (scores) and can be seen as a generalization of the score test that is also known as Lagrange multiplier (LM) test. Nevertheless, the tests only work with one variable at a time that must be preselected by the user.

Recursive partitioning (see, e.g. Strobl, Malley, & Tutz, 2009) is a suitable solution to this constraint. With the development of the model-based recursive partitioning (MOB, Zeileis, Hothorn, & Hornik, 2008) algorithm that is based on statistical inference procedures (Hothorn, Hornik, & Zeileis, 2006), it has become possible to *explore* parameter instability in parametric models with respect to multiple variables of mixed types and has lead to a growing number of applications for various models across dif-

ferent disciplines (see, e.g., Bürgin & Ritschard, 2015; Fokkema, Smits, & Zeileis, 2015; Komboz, Strobl, & Zeileis, 2016; Strobl et al., 2015; Strobl, Wickelmaier, & Zeileis, 2011; Wickelmaier, 2016).

In recursive partitioning, the observations are repeatedly separated into subgroups – formed by a set of candidate variables – that differ with respect to one or more parameters in the model (see, e.g., Strobl et al., 2009, for an introduction to recursive partitioning). Simple and widely used algorithms for recursive partitioning are the well-known classification and regression trees (Breiman, Friedman, Stone, & Olshen, 1984). In its original version, the split variable and the cutpoints are selected by the largest impurity reduction in the response variable. In the MOB algorithm, however, the split variables are selected by means of a parameter instability test (the score test) and the optimal cutpoint is determined by the maximum log-likelihood of the model over all candidate cutpoints. This procedure is computationally efficient and therefore an attractive approach to detect parameter instability with respect to known or unknown subgroups formed by several variables and interactions between them. The resulting partition can be illustrated in the form of a decision tree that is easy and intuitive to interpret for substantive researchers.

An example of a model-based tree is given in Figure 1. The figure illustrates a "Rasch tree" which is a method proposed by Strobl et al. (2015) to investigate DIF in the Rasch model, a popular psychometric model for analyzing dichotomous response data. The data used for this example originate from an online test on general knowledge conducted by the German news magazine SPIEGEL in 2009. The respondents answered 45 items from the knowledge domains politics, history, economy, culture, and natural sciences as well as sociodemographic questions (gender, age, etc.). For this example, a subsample containing the responses of 1075 university students from Bavaria to the economy domain was analyzed. The complete data was analyzed and discussed in Trepte and Verbeet (2010). Code to reproduce the example using the free open source software R for statistical computing (R Core Team, 2016) can be found in Appendix C.

The mere fact that the tree illustrated in Figure 1 shows one or more splits indicates that some of the items were subject to DIF (Strobl et al., 2015). The method detected four subgroups that differed with respect to the item difficulty parameters in the Rasch model. Those are displayed in the terminal nodes using profiles plots. In each terminal node, the level of the difficulty for the particular subgroup is illustrated on the y-axis (lower values indicate easier items) for each of the nine items (x-direction). The most striking difference was observed for item 1 that is highlighted for presentation purposes. Given the same level of ability, item 1 was harder to be answered correctly for female students than for male students. This conclusion follows from the comparison of the difficulty of item 1 compared to the other items in nodes 3 and 4 (terminal nodes to the left) versus the difficulty in nodes 6 and 7 (terminal nodes to the right). Within both subgroups (males and females), the item was additionally slightly harder to be answered

Figure 1: Tree representation of the result from applying the Rasch tree method for detecting DIF in the Rasch model (Strobl et al., 2015) to data from an online quiz on "general knowledge" conducted by the German news magazine SPIEGEL in 2009. The method was applied to a subsample of university students from Bavaria answering questions about economics. The terminal nodes illustrate the item difficulty parameters in the Rasch model for each item (lower values indicate easier items) for different subgroups formed by sociodemographic characteristics of the participants.

correctly for students who visited the website of the SPIEGEL online magazine less then two times per week (`spon` $\leq$ 2–3/week) than for students who visited the website two or more times per week. This can be concluded by comparing the difficulty of item 1 between the nodes 3 and 4 as well as between the nodes 6 and 7. The task in item 1 was to recognize Dieter Zetsche, CEO of Daimler AG (a German car manufacturer), from a picture. The DIF analysis indicated that male students and students who visited the SPIEGEL website regularly had an advantage for this question. Note that the Rasch tree investigates DIF jointly for all items and that in this example the difficulty parameters of other items also varied slightly between the subgroups.

The Rasch tree example illustrates that tree-based methods are very useful for exploratory data analysis or for testing model assumptions. Tree methods, however, are prone to generate unstable results (Breiman, 1996b). Instability is revealed when the selected variables and cutpoints vary after small data perturbations or when a different learning sample from the same DGP is used to generate the tree. Hence, the tree is in itself a result from a statistical analysis that needs to be investigated for stability.

The first methodological result presented in Chapter 1 of this thesis is concerned with the stability of tree-based models. The stability of trees was extensively studied in the literature (see, e.g., Briand, Ducharme, Parache, & Mercat-Rommens, 2009; Ciampi & Thiffault, 1988; Miglio & Soffritti, 2004; Ntoutsi, Kalousis, & Theodoridis, 2008). Some work has focused on assessing the stability of the structure of trees, in particular, the sequence of the splits. In Chapter 1, however, it will be demonstrated by means of a simple hypothetical example that trees can lead to the same interpretation, even if the order of the splits is different. Thus, for assessing the stability of a tree with respect to its interpretation, it is important to investigate the stability of the variable and cutpoint selection rather than its paths. Chapter 1 therefore presents a toolkit of simple graphical and computational methods to assess the stability of the variable and cutpoint selection in tree-based models. For this, the algorithm is trained on several learning samples generated by resampling from the original data set.

The concept of measuring stability by resampling can be extended to results from supervised statistical learning in general. This idea is pursued in Chapter 2. Adaptive and flexible methods such as recursive partitioning, neural networks, or support vector machines have become widely used in business, industry, and many different scientific areas (see, e.g., Byun & Lee, 2003; Vellido, Lisboa, & Vaughan, 1999; H. Zhang & Singer, 2010, for surveys of applications). These methods are commonly used for *predictive modeling*, where the goal is to achieve accurate predictions. A higher predictive power can often be achieved at the cost of a lower interpretability (Breiman, 2001). Methods that generate less interpretable results are often described as "black boxes".

However, attempts to peek into the black boxes have been made to find out how such methods exploit the association between the predictors and the response in a data set. In the field of neural computing, for example, a wide range of studies propose

approaches for the extraction of classification rules from trained neural networks (see, e.g., Andrews, Diederich, & Tickle, 1995, for a survey) or support vector machines (see, e.g., Barakat & Bradley, 2010, for a review). A general approach to visualize the results from supervised statistical learning via individual conditional expectation plots was presented by Goldstein, Kapelner, Bleich, and Pitkin (2015) and is available in the R package **ICEbox**.

Moreover, it is widely known that some supervised statistical learning algorithms can generate unstable results (e.g., recursive partitioning and neural networks, see Breiman, 1996b). The predictions from an unstable result can be stabilized by using ensemble methods such as bagging (Breiman, 1996a, 1998), boosting (e.g., Schapire, Freund, Bartlett, & Lee, 1998), or random forests (Breiman, 2001). The big drawback of these approaches is that any potential interpretability of a single result is largely lost through the aggregation. Thus, ensemble methods also generate black boxes.

Generally, when the aim is explanation or description rather than prediction (see, e.g., Breiman, 2001; Shmueli, 2010, for discussions on various modeling cultures), stability is again a key requirement to draw reliable conclusions. Moreover, since many algorithms for supervised statistical learning are not based on a stochastic model for the data, they also do not provide measures of confidence for their results. Thus, there is a need for methods to quantify the uncertainty of results from supervised statistical learning.

Therefore, in Chapter 2, a very general framework is proposed that can be used for measuring the stability of results from supervised statistical learning. Moreover, the framework will be used to show that even algorithms usually considered as unstable, such as algorithms for recursive partitioning, can generate stable results if the functional form of the relationship between the predictors and the conditional distribution of the response matches to what the algorithm learns.

The first two chapters cover the idea of assessing stability by means of resampling from the original data. In the context of the main topic of this thesis – stability of statistical conclusions in general – the contributions following in the remaining chapters should be regarded as applications of statistical methods to investigate the uncertainty and the instability of parameters in a particular family of parametric models. To be more specific, Chapters 3 and 4 are concerned with the estimation of standard errors and the assessment of differential item functioning (DIF) in cognitive diagnosis models (CDM), a particular family of psychometric models.

While methods for detecting DIF have a long research tradition in the framework of item response models (Fischer & Molenaar, 2012), they are relatively new in CDMs, just like the models themselves. CDMs are used primarily in educational measurement to measure the strength and weaknesses of students. An item from basic probability calculus, for example, may require one or more different abilities to be solved correctly: The calculation of the probability of an event, the probability of the complement of

an event, the probability of the union of two disjoint events, or the probability of two independent events (see Section 3.3.2 for a real data example). CDMs can be used to assess the mastery or nonmastery of such fine-grained abilities, so-called attributes, for each individual.

Several different versions of CDMs have been proposed over the last two decades that differ with respect to various characteristics of the tests (e.g., dichotomous or polytomous responses, compensatory, or noncompensatory processes). Rupp, Templin, and Henson (2010) provided a taxonomy of different CDMs. Many of these can be subsumed within a more general framework, such as the generalized deterministic input, noisy "and" gate (G-DINA; de la Torre, 2011) model.

Chapter 3 is concerned with the estimation of standard errors in the G-DINA model. A widespread approach to estimate the model parameters of CDMs is via marginal maximum likelihood. In this approach, a probability distribution is imposed for the attribute patterns of the individuals, the so-called latent class distribution. According to likelihood theory, the standard errors can then be computed via the inverse of the information matrix. A common mistake in the CDM literature is to compute the information matrix only for the item parameters of the model, while ignoring the parameters used to specify the latent class distribution. We show mathematically and by means of simulations that this approach leads to an underestimation of the standard errors for the item parameters.

This can have severe consequences for inference techniques that rely on standard errors (or the entire covariance matrix) including tests for parameter instability. The Wald test, that was for example proposed for detecting DIF, showed Type I error inflation in previous studies (Hou, de la Torre, & Nandakumar, 2014; X. Li & Wang, 2015). In Chapter 4 it is shown by means of simulations that the Type I error inflation vanishes when the correct computation of the covariance matrix is used. Additionally, the performance (i.e., the Type I error and the power) of the Wald test is compared to the LM test that could be used alternatively for detecting parameter instability in CDMs.

In summary, this thesis highlights the importance of stability for reliable interpretations of results from statistical data analysis and contributes to statistical approaches for assessing the stability of the variable and cutpoint selection in tree-based models (Chapter 1), the stability of the predictions of results from supervised statistical learning in general (Chapter 2), and the uncertainty and the stability of parameters in cognitive diagnosis models (Chapters 3 and 4). Add-on software packages are provided to perform the proposed stability analyzes in practice.

Contributing articles, manuscripts, and software

Parts of this dissertation are based on published articles, submitted manuscripts, working papers, and add-on software packages for the free open source software R for statistical computing (R Core Team, 2016).

- Chapter 1 corresponds (up to minor changes) to the published article:

  Philipp, M., Zeileis, A., and Strobl, C. (2016). A toolkit for stability assessment of tree-based learners. In A. Colubi, A. Blanco, and C. Gatu (Eds.), *Proceedings of compstat 2016 – 22nd international conference on computational statistics* (pp. 315–325). The International Statistical Institute/International Association for Statistical Computing.

  Michel Philipp developed and designed the descriptive measures and graphics proposed in the article, constructed the examples that are used throughout the article, and drafted the manuscript. Carolin Strobl initiated the project. Michel Philipp and Achim Zeileis developed the R package **stablelearner**. Achim Zeileis and Carolin Strobl contributed to the methodology and the manuscript.

- Chapter 2 is based on the manuscript (that is currently under review):

  Philipp, M., Rusch, T., Hornik, K., and Strobl, C. (2016). *Measuring the stability of results from statistical learning.* (Manuscript under review)

  Michel Philipp studied the literature, designed and conducted the simulation and benchmark experiments, and developed the R code to perform stability analyzes. Michel Philipp and Carolin Strobl identified the key arguments for measuring stability by comparing predictions, designed the framework, and drafted the manuscript. Thomas Rusch and Kurt Hornik contributed to the methodology and made suggestions to improve the manuscript.

- Chapter 3 is based on the manuscript (that is accepted for publication):

  Philipp, M., Strobl, C., de la Torre, J., and Zeileis, A. (in press). On the estimation of standard errors in cognitive diagnosis models. *Journal of Educational and Behavioral Statistics*.

  Michel Philipp studied the literature, set up the mathematical derivation and the proof of the underestimation of the standard errors, developed the R package **Rcdm**, designed and conducted the simulation study and the real data example, and drafted the manuscript. Achim Zeileis recognized a key argument of the article and contributed to the methodology, the notation, and the initial R-code. Jimmy de la Torre contributed to the design of the simulation studies and made suggestions to improve the manuscript. Carolin Strobl contributed to the manuscript.

- Chapter 4 is based on additional results.

  Michel Philipp studied the literature, designed and conducted the simulation studies, and drafted the chapter. Achim Zeileis contributed to the methodology and Carolin Strobl contributed to the manuscript.

Chapter 1

A toolkit for stability assessment of tree-based learners

Michel Philipp, *University of Zurich*
Achim Zeileis, *Universität Innsbruck*
Carolin Strobl, *University of Zurich*

Abstract Recursive partitioning techniques are established and frequently applied for exploring unknown structures in complex and possibly high-dimensional data sets. The methods can be used to detect interactions and nonlinear structures in a data-driven way by recursively splitting the predictor space to form homogeneous groups of observations. However, while the resulting trees are easy to interpret, they are also known to be potentially unstable. Altering the data slightly can change either the variables and/or the cutpoints selected for splitting. Moreover, the methods do not provide measures of confidence for the selected splits and therefore users cannot assess the uncertainty of a given fitted tree. We present a toolkit of descriptive measures and graphical illustrations based on resampling that can be used to assess the stability of the variable and cutpoint selection in recursive partitioning. The summary measures and graphics available in the toolkit are illustrated using a real world data set and implemented in the R package **stablelearner**.

Keywords: stability, recursive partitioning, variable selection, cutpoint selection, decision trees

1.1 Introduction

Recursive partitioning approaches, such as classification and regression trees (CART; Breiman et al., 1984), conditional inference trees (Hothorn et al., 2006), or model-based recursive partitioning (Zeileis et al., 2008), are widely used for modeling complex and possibly high-dimensional data sets (Strobl et al., 2009). The methods are able to detect high-degree interactions and nonlinear structures in a data-driven way. Therefore, these methods have been frequently applied in many scientific disciplines, as well as in many industries for predictive modeling purposes (Kuhn & Johnson, 2013).

Nowadays, more complex and more flexible methods exist for predictive learning that often achieve a better prediction accuracy (e.g., random forests, boosting, support vector machines, neural networks). Recursive partitioning, however, is still a popular method in situations where the aim is to infer and interpret the structure of the underlying process that has generated the data. For this purpose, recursive partitioning is often favored over other methods, since the results can be illustrated in the form of decision trees, which are relatively easy to interpret. Therefore tree-based methods are widely used as exploratory modeling techniques in many fields, such as social and behavioral sciences (see, e.g., Kopf, 2013).

Recursive partitioning algorithms recursively split the predictor space $\mathcal{X} \in \mathbb{R}_p$ to form homogenous groups of observations. The various algorithms that have been proposed in the literature mainly differ with respect to the criteria for selecting the split variable, choosing the cutpoint and stopping the recursion (see Hothorn et al., 2006). CART, for example, selects the variable and the cutpoint that best unmixes the classes in case of a classification problem, or that most reduces the squared error loss in case of a regression problem. Conditional inference trees, on the other hand, perform splitting and stopping based on a statistical inference procedure.

Despite their popularity, a major drawback of recursive partitioning methods is their instability. By studying the predictive loss of different regularization techniques, Breiman (Breiman, 1996b) identified recursive partitioning (among others) as unstable. It is well known that small changes in the training data can affect the selection of the split variable and the choice of the cutpoint at any stage in the recursive procedure, such that the resulting tree can take a very different form (Kuhn & Johnson, 2013; Strobl et al., 2009; Turney, 1995). Moreover, recursive partitioning methods do not provide measures of confidence for the results. Therefore, users cannot assess the degree of certainty for selected variables and cutpoints. Hence, the question remains to what extend one can rely on the splits in a single tree to draw conclusions.

Previous research has already focused on assessing the stability of trees from different perspectives and with different goals (see, e.g., Bar-hen, Gey, & Poggi, 2015; Briand et al., 2009; Miglio & Soffritti, 2004). Their methods are commonly based on a measure

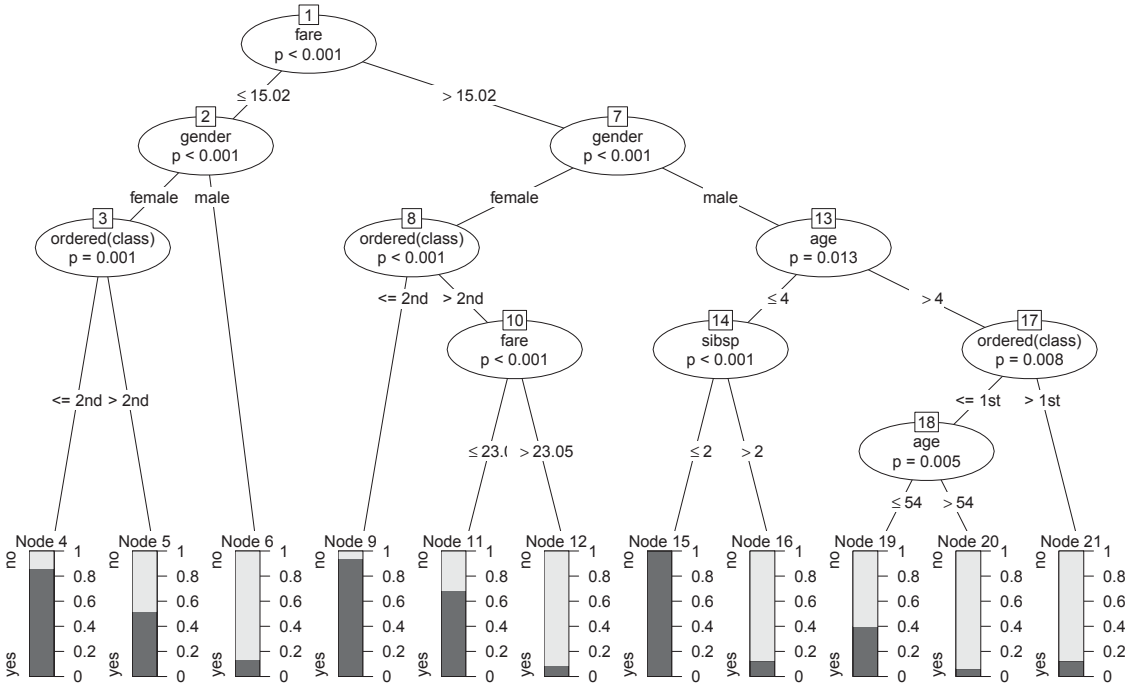
that is used to compare the distance (or similarity) between pairs of trees. In (Briand et al., 2009), for example, a measure of similarity between trees was proposed to stabilize the selection of the splits in a specific tree algorithm. And more recently, measures and theory were proposed to detect observations that influence the prediction or the pruning in CART (Bar-hen et al., 2015). While in these approaches the prediction, partitioning, and the structure of the trees are considered separately, they may also be combined in one measure (Miglio & Soffritti, 2004). Thus, while previous research has focused on reducing instability, measuring the influence of individual observations, or assessing the distance between trees, we focus on assessing and visualizing two important aspects that reveal the stability of a tree resulting for a given data set: the variable and the cutpoint selection.

In this paper, we first discuss instability of results from recursive partitioning using a practical example. In the second part, we present a computational procedure and a number of graphical tools that support users for assessing the stability of the variable and the cutpoint selection. The proposed methods are implemented in the software package **stablelearner** (currently available from `https://R-Forge.R-project.org/projects/stablelearner/`) for the free R system for statistical computing (R Core Team, 2016). By using a real world data set, the package will be used throughout the article for illustrating the proposed methods.

1.2 Instability of trees

To illustrate the instability of trees we have used recursive partitioning to predict the survival of the passengers during the sinking of the RMS Titanic in 1912 by several passenger characteristics. A complete passenger list is available online on `http://www.encyclopedia-titanica.org/` (accessed on 2016-04-05). According to the list, 1317 passengers (excluding crew members) were aboard from which 500 survived the sinking. The passenger information that was used for training the tree was gender, age, fare, class (1st, 2nd, or 3rd), place of embarkment (B = Belfast, C = Cherbourg, Q = Queenstown, S = Southampton), number of siblings/spouses aboard (abbreviated as sibsp), and number of parents/children aboard (abbreviated as parch). The last two features were obtained from an overlapping data set available on `http://biostat.mc.vanderbilt.edu/wiki/Main/DataSets`. The tree was generated using the function `ctree` from the **partykit** package (Hothorn et al., 2006; Hothorn & Zeileis, 2015) that performs recursive partitioning in a conditional inference framework in R and is illustrated in the form of a tree in the upper panel of Figure 1.1. In the following, we will refer to this result as the original tree, since the partitioning was performed on the original passenger data (as opposed to random samples drawn from the original data set employed subsequently).

(a) Tree based on the original data set:



(b) Tree based on a bootstrap sample drawn from the original data set:

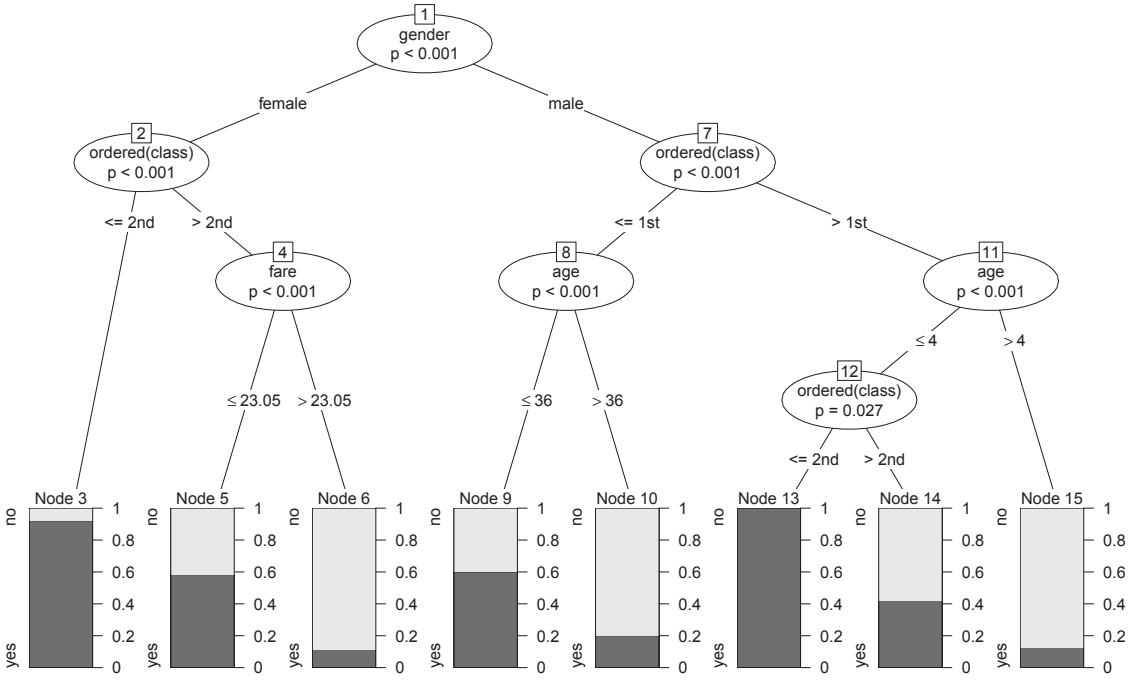


Figure 1.1: Tree representation of results from recursive partitioning for the RMS Titanic passenger data.

Based on a bootstrap sample taken from the original passenger data, we generated a second tree, which is illustrated in the lower panel of Figure 1.1. The structures of the trees look quite different at first sight, which suggests a large instability of the tree. However, when looking closer one can identify variables that were selected in both trees and split at the same or a similar cutpoint. For example, the numerical variable `age` was split at 4 and 54 in the original tree and at the values 4 and 36 in the bootstrap tree, or the numerical variable `fare` was split twice in the original tree at 15.02 and 23.05 and at 23.05 in the bootstrap tree. Thus, many splits appeared in both trees, only the order and the cutpoints for numerical variables were slightly different.

As Turney (1995) elaborated in his work, two trees that are structurally different can be logically equivalent. This means that two trees can lead to very similar or even the same interpretation although their structures (in particular the order of the splits) look very different. To illustrate this principle, we consider two hypothetic trees for a simple classification problem with two classes and a two-dimensional predictor space. Note, however, that the statement also holds for any type of response variable and also for predictor spaces with more dimensions. Figure 2.1 shows two trees (upper row) and representations of the corresponding partitioning of the feature space (bottom row). In the illustration the predicted class in each terminal node is indicated by the colors red and green. According to the figures in panel (a) and (b), the tree structures differ by the split variable in their root node, their path structure, and the sequence of split variables in the paths between the root node and the leafs. Yet, though the two trees are structurally different, the predictions are equivalent for any point in the predictor space. By mentally merging the two partitioning representations, it becomes evident that the two trees have identical splits that only appear in a different order in the tree representation.

To assess whether a tree is stable or not, it is therefore principally important to investigate the stability of the splits, rather than the stability of the entire tree structure. From a single tree representation it is not possible to identify which splits are stable. It is possible, however from an ensemble of trees, e.g., generated by resampling from the original data. From the ensemble, the stability of the splits can be assessed by investigating the variable selection frequency and the cutpoint variability.

1.3 Measuring variable selection and cutpoint stability

In the following we will outline what steps are necessary from a conceptual point of view to assess the stability of variable and cutpoint selection in trees. Subsequently, these steps will be illustrated for a binary classification tree modeling survival vs. non-survival on the RMS Titanic.

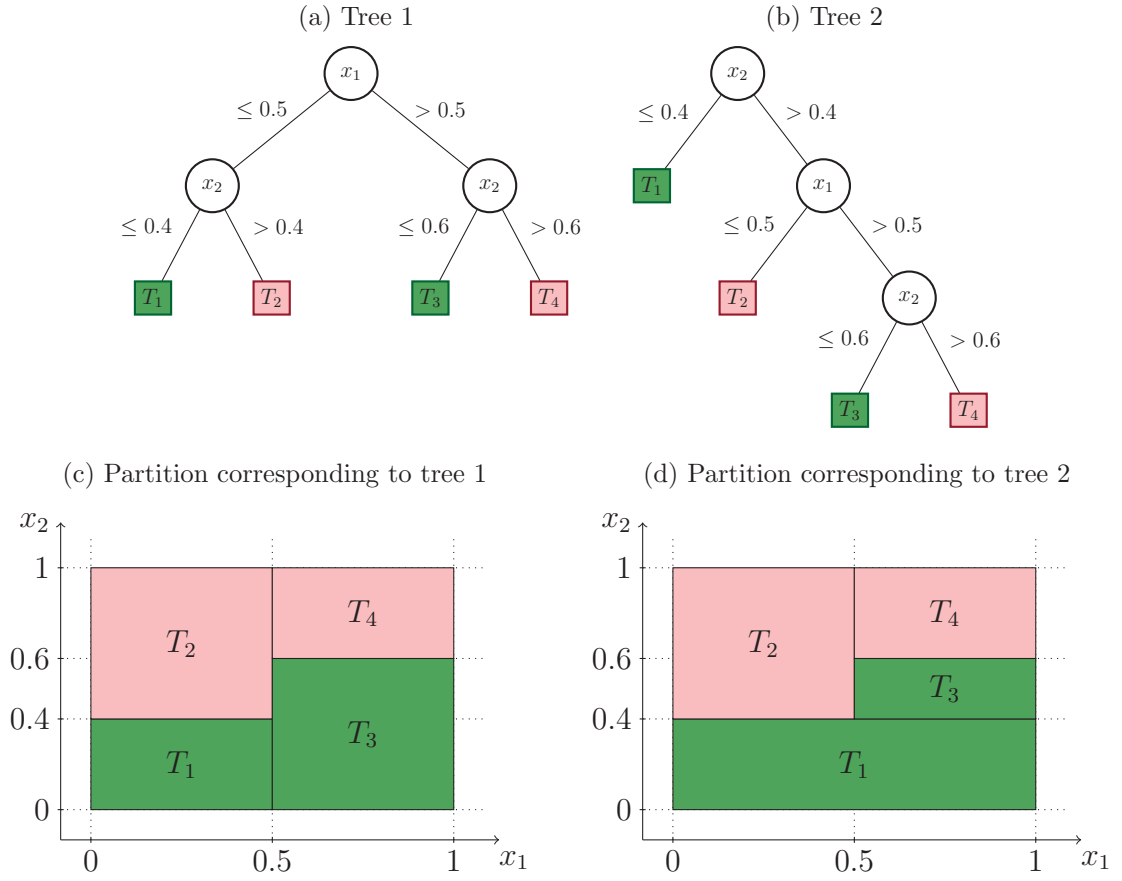


Figure 1.2: Examples of different tree structures, but equivalent partitions and interpretations.

The first step to assess stability is to draw several samples from the original data. The second step is to compute the descriptive measures and graphics provided in our toolkit over all samples. The options implemented in the package for generating samples in the first step are bootstrap sampling (sampling with replacement), subsampling (sampling without replacement), k-fold sample splitting (partitioning the original data into k equally sized samples), leave-k-out jackknife sampling, or further user-defined strategies. Since each option has its specific peculiarities, they will likely generate different results. For the further illustration we will focus on bootstrap sampling, which is most widely used and was chosen as the default option in the function `stabletree()` that performs the resampling and extracts the required information from the ensemble for further analysis:

```
R> library("stablelearner")
R> data("titanic", package = "stablelearner")
R> m <- ctree(survived ~ gender + age + fare + ordered(class) + embarked +
+    sibsp + parch, data = subset(titanic, class %in% c("1st", "2nd", "3rd")))
R> s <- stabletree(m, B = 500)
```

The function `stabletree()` requires a tree-based model object that either inherits from class `party` (like, e.g., the result of `ctree()` or `glmtree()`) or can be coerced to it (like, e.g., the results of `rpart()` or `J48()`). Additionally, parallelization can easily be utilized with a convenience option for multicore computation based on **parallel** (for platforms that support this).

In the remaining part of this section, descriptive measures and graphical illustrations are introduced for investigating the stability of the splits, specifically for the variable and the cutpoint selection. First, the measures will be briefly discussed and then illustrated for the Titanic example.

1.3.1 Variable selection analysis

The aim of the variable selection analysis is to investigate *whether* variables that are selected for splitting in the original tree are also consistently selected for splitting in the resampled data sets. Furthermore, it can be compared *how often* (on average) a variable is selected within the original tree and the repetitions, respectively.

The first descriptive measure is simply the relative frequency of selecting variable x_j for splitting, computed over all repetitions in the procedure. Let $b = 1, \ldots, B$ denote the index for the repetitions and $j = 1, \ldots, p$ the index of the variables considered for partitioning. Further, let $\mathbf{S} = \{s_{bj}\}$ be a binary matrix, where $s_{bj} = 1$ if variable x_j was selected for splitting in repetition b and 0 otherwise. Then, the relative *variable selection frequency* is computed by $100 \cdot \frac{1}{B} \sum_{b=1}^{B} s_{bj}$ and is expected to be large (i.e., close to 100%) for those variables selected in the original tree, if the result is stable. The variable selection frequency can be illustrated graphically using a `barplot()` method that generates the barplot depicted in the left panel of Figure 1.3. The variables depicted on the x-axis are sorted in decreasing order with respect to their variable selection frequencies (here and in all the following graphical tools). The bars of variables selected in the original tree are colored in dark gray and the corresponding labels are underlined. Thus, from the plot we can infer that the variables `gender`, `class`, `age`, `fare`, and `sibsp` were selected for splitting in the original tree. The height of the bars corresponds to the variable selection frequency depicted on the y-axis. The first two bars reach the upper limit of 100%, which means that the variables `gender` and `class` were selected for splitting in each repetition. The variable `age`, represented by the third bar, was selected slightly less than 100% (but still very often) over the repetitions. The variables `fare` and `sibsp`, represented by the fourth and the fifth bar, were selected in the original tree, but not as frequently over all repetitions. This indicates that the splits in those variables in the original tree must be considered less reliable compared to the splits of the variable `gender`, `class`, and `age`. The last two bars represent the variables `embarked` and `parch`, which were not selected in the original tree. They were selected for splitting in less than 50% of the repetitions. This indicates that although those

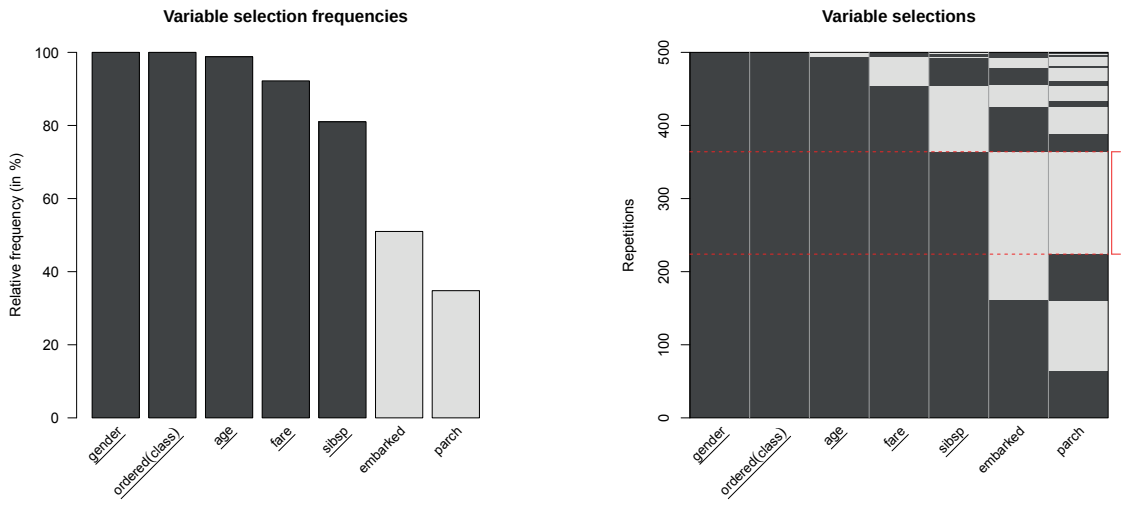


Figure 1.3: Graphical variable selection analysis.

variables seem to carry some information that is useful for predicting survival, they are not predominant. From a content perspective one may assume for this example that, over the repetitions, the variables `embarked` and `parch` occasionally acted as a proxy for the other variables in the data set.

The `summary()` method prints the corresponding table with the variable selection frequency (termed `freq`) in the first column for each variable. The second column (headed by an asterisk) indicates whether the variable was selected for splitting in the original tree:

```
                freq *  mean *
gender         1.000 1 1.644 2
ordered(class) 1.000 1 2.578 3
age            0.988 1 2.316 2
fare           0.922 1 1.676 2
sibsp          0.810 1 1.132 1
embarked       0.510 0 0.638 0
parch          0.348 0 0.444 0
(* = original tree)
```

The third column in the table (termed `mean`) contains the values of another descriptive measure and denotes the average count splitting in variable x_j per tree. Let $\mathbf{C} = \{c_{bj}\}$ be an integer matrix, where c_{bj} equals the number of times x_j was used for splitting in the tree for repetition b. Note that this number can be greater than one, because the same variable may be used for splitting in different parts of the tree. The *average variable split count* is computed by $\frac{1}{B}\sum_{b=1}^{B} c_{bj}$ and is expected to be close to the count

of splitting in variable x_j in the original tree. The last column in the table (also headed by an asterisk) indicates how many times the variable was selected for splitting in the original tree. For example, the variable `gender`, was used on average 1.644 times over all repetitions and twice in the original tree. It is possible that the variable `gender` was often split on a higher level (and thus less often used for splitting) in the repetitions, as compared to the original tree. The reverse may be assumed for the variable `age`, which was on average more often used for splitting over the repetitions than it was used for splitting in the original tree. Similar interpretations follow from the information for the other variables.

Furthermore, we can investigate the combinations of variables selected in the various trees over the repetitions. This can be illustrated using the function `image()`. The resulting plot that is illustrated in the right panel of Figure 1.3 is a graphical illustration of the binary matrix $\mathbf{S}$ that contains the variable selections over the repetitions. A fine grid of rectangles is drawn for each element in $\mathbf{S}$, which are colored dark gray if $s_{bj} = 1$ and light gray if $s_{bj} = 0$. The repetitions (illustrated in the y direction) are ordered such that similar combinations of selected variables are grouped together. The combination of variables used for splitting in the original tree is marked on the right side of the plot using a thin solid red line. The area representing the combination is additionally enclosed by two dashed red lines. Notice that this is also the most frequent combination of variables selected over all repetitions. Repetitions that included additional variables beyond the combination in the original tree are illustrated below the marked area. Hence, we can deduce from the illustration that the variables `embarked` and `parch` were sometimes additionally used for splitting. In the replications above the marked area some splitting variables from the original tree were substituted with other variables.

1.3.2 Cutpoint analysis

The variable selection analysis showed that there are some variables which are consistently used for splitting, indicating that those variables are more relevant in predicting survival than others. However, even when the same variables are selected, the splits may still vary with respect to the cutpoints chosen for splitting. Therefore a further important step in assessing the stability of the splits is the analysis of the cutpoints, which provides more detailed information about the variability of the splits.

We suggest different graphical illustrations for analyzing the variability of the cutpoints for numerical, unordered categorical and ordered categorical variables. Using the function `plot()` these illustrations can be generated for all variables specified in the model. According to the type of variable the correct illustration is chosen automatically and the variable names are underlined if the variable was selected for splitting in the original tree. Figure 1.4 illustrates these plots for the variables in the Titanic passenger data set.

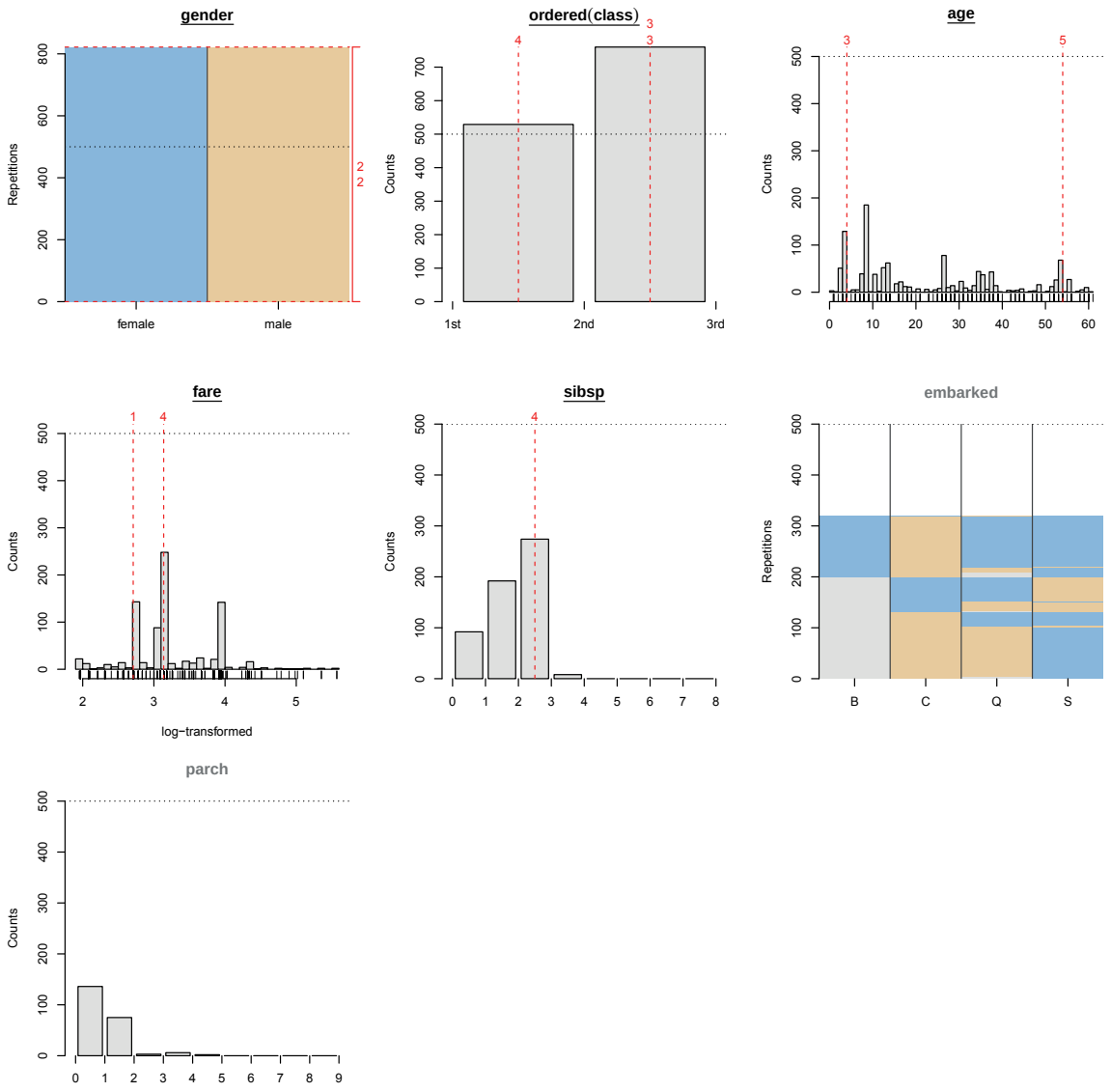


Figure 1.4: Graphical cutpoint analysis.

To analyze the cutpoints for ordered categorical variables, we suggest to use a barplot that shows the frequency of all possible cutpoints. Those are sorted on the x-axis by their natural order that arises from the ordering of the categories of the variables. Examples are given for the variables `class`, `sibsp`, and `parch` in Figure 1.4. Additionally, the cutpoints chosen in the original tree are marked using a vertical dashed red line. The number above each line indicates at which level the split occurred in the original tree. For example, the cutpoint between the first and the second class is selected more than 500 times (the number of repetitions in this example). This means that for some repetitions the split appeared several times in different positions in the same tree (for example in parallel branches). However, the passengers were split even

more often between the second and the third class. The illustration indicates that the observations were consistently split by their class affiliation over the repetitions to predict survival of the passengers. The cutpoint in the variable `sibsp`, on the other hand, was less stable. Although the variable was quite frequently selected for splitting, the variable was often split between lower categories over the repetitions as compared to the original tree. The variable `parch`, which was not used in the original tree, was split only few times between the lower categories and can thus be considered as not very relevant.

To analyze the partition for unordered categorical variables (avoiding ambiguities by using the term "partition" rather than "cutpoint" here), we suggest to use image plots, as illustrated for the variables `gender` and `embarked` in Figure 1.4. When using binary splits, observations with the same categories are partitioned into the left or the right daughter node. Thus, the categories are assigned to the left or to the right branch of the split, respectively. For visualizing the partitions over the repetitions, categories that are directed to the same daughter node are illustrated by the same color. For the variable `gender`, there is only one possible split between the two categories `Female` and `Male`. The plot illustrates, however, that this split occurs many times (more than 500) over all repetitions, which underscores the relevance of the split. The combination of categories that represent a partition as it occurred in the original tree, is marked on the right side of the plot using a thin solid red line. The area representing the partition is additionally enclosed by two dashed lines (this is a little hard to see here, because the binary variable `gender` only offers one possible partition). Furthermore, the number(s) on the right side of the marking also represent(s) the level(s) of the corresponding split(s) in the structure of the original tree. The two numbers on the right side of the illustration for the variable `gender` in Figure 1.4 indicate that `gender` was split twice on the second level in the original tree.

The plot becomes more detailed for variables with more than two categories such as the variable `embarked`. This variable, however, was not used for splitting in the original tree. Nevertheless it was used relatively often for splitting over all repetitions. In this illustration the additional color light gray is used when a category was no more represented by the observations left for partitioning in the particular node. The partitions over all repetitions are ordered such that equal partitions are grouped together. The most frequent partitions are $[C, Q]$ versus $[S]$ and $[C]$ versus $[B, Q, S]$. Since passengers from the different classes tended to embark in different cities (e.g., most third class passengers embarked in Southampton), the variable `embarked` may in some repetitions (but not in the original tree) have been used as a proxy for the variable `class` in parts of the tree.

To analyze the cutpoints for numerical variables, we suggest to use a histogram, as illustrated for the variables `age` and `fare`. According to the distribution illustrated for the variable `age`, the cutpoints selected over the repetitions spread over the complete

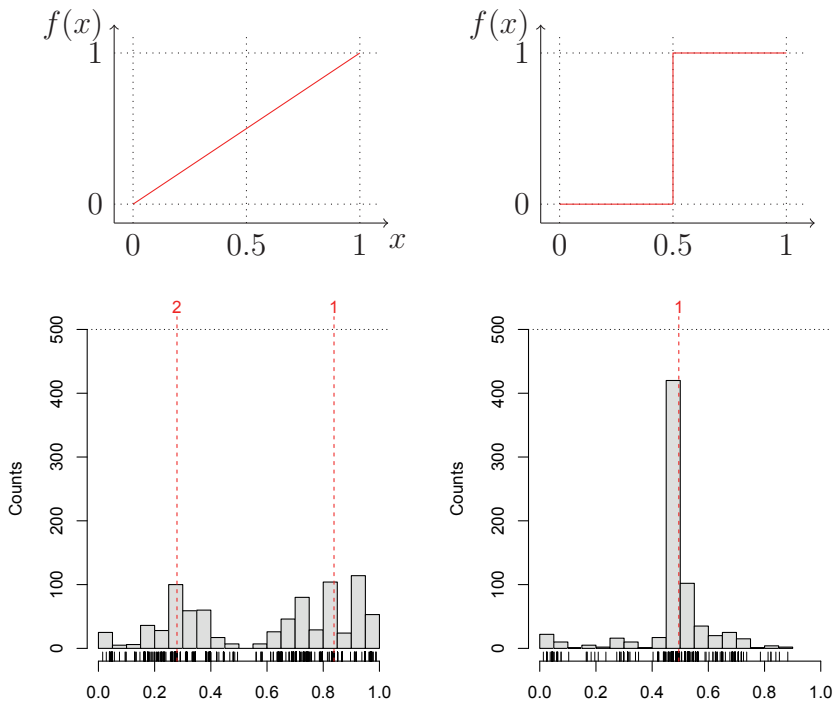


Figure 1.5: Cutpoint analysis for artificial regression problem.

range of possible cutpoints. Although some cutpoints were selected more frequently than others, there were no unique cutpoints that were selected over most repetitions. The selected cutpoints of the variable `fare` are illustrated on a logarithmic scale in Figure 1.4, as it makes the picture easier to read. Again, the cutpoints selected over the repetitions spread over the complete range of possible cutpoints. However, the cutpoints selected in the original tree match two of three distinct peaks in the histogram and can be considered slightly more stable as compared to the cutpoints within the variable `age`.

From a conceptual point of view, the cutpoint pattern reflects the underlying functional shape. Due to the recursive nature of trees, smooth functions need to be approximated by several splits while piecewise constant step functions can be described straightforwardly by individual splits (see also Strobl et al., 2009). This is illustrated in Figure 1.5. The upper left panel illustrates a linear relationship. To approximate this functional form, a tree algorithm will split the variable several times at different cutpoints. Altering the data would thus very likely lead to a different cutpoint. For a piecewise constant function like the one illustrated in the upper right panel of Figure 1.5, on the other hand, the functional form is captured by a single split that is relatively easy to detect and will not change much if the data are altered.

To further demonstrate how the cutpoint stability plots can reflect the underlying functional form, we have simulated 500 observations from the model $y = f(x) + \varepsilon$ for

each of the two functions displayed in the top row of Figure 1.5. The variable x was sampled from a uniform distribution $\in [0, 1]$ and ε was sampled from a standard normal distribution. In the bottom row of Figure 1.5 the stability of the cutpoints for the variable x is illustrated for the two artificial examples. As expected, the identification of a stable cutpoint failed for the example with the linear relationship (see lower-left panel). In the example with the piecewise constant relationship, however, the cutpoint at 0.5 was correctly recovered over most repetitions (see lower-right panel). For the Titanic example illustrated in Figure 1.4 this means that the cutpoints selected in the original tree for the variable `age` are rather unstable and should not be overinterpreted because the underlying functions seems to be smooth rather than piecewise constant. The cutpoints selected for the variable `fare` are slightly more stable.

To sum up, the stability analysis of the binary classification tree fitted for the Titanic data revealed that many splits in the original tree illustrated in Figure 1.1 were rather stable, but some parts were quite variable. First, the splits of the variables `gender` and `class` can be considered as important and stable. Second, the splits of the variable `age` are ambiguous, although the variable is definitely relevant for predicting survival of the passengers. Furthermore, the splits of the variable `fare` are fairly stable, but the variable was a few times not selected for splitting over the repetitions. Thus, if the data were altered slightly, the variable might also had been omitted for splitting in the original tree. And finally, the split of the variable `sibsp` is least stable and should not be overinterpreted.

1.4 Discussion

In this paper we have presented a toolkit of descriptive measures and graphical illustrations that can be used to investigate the stability of the variable and cutpoint selection in models resulting from recursive partitioning. It was demonstrated how the tools are used and illustrated how intuitive they are by a real world data set. The analysis revealed that many aspects of the fitted tree were rather stable, but some parts were quite variable. Notice that the toolkit is not limited to classification trees, but can also be used to investigate the stability of regression trees or model-based trees. It was further illustrated that clear cutpoints from piecewise constant functions in the underlying data generating process, can be identified using the proposed graphics for the cutpoint analysis.

To acknowledge some limitations associated with the tools it should be mentioned that they produce less meaningful results for very large trees with many splits. If the structure of the underlying data generating process is complex, the sample size or the number of predictors is large, it can become tedious to interpret a tree. Assessing the variable selection and cutpoint stability of such trees is computationally very intensive

and the result might be unclear. However, the complexity of a tree can be reduced by modifying the settings (i.e., the pruning rule or the stopping criteria) of the recursive partitioning algorithm. Furthermore one should always be aware that any resampling scheme can only mimic what would happen if a new sample could be drawn from the population. And finally, the proposed tools do not assess the predictive stability of trees, which is another important aspect for their interpretation, as we briefly saw in Section 1.2. This aspect will be addressed in future research.

Chapter 2

Measuring the stability of results from supervised statistical learning

Michel Philipp, *University of Zurich*
Thomas Rusch, *WU Vienna University of Economics and Business*
Kurt Hornik, *WU Vienna University of Economics and Business*
Carolin Strobl, *University of Zurich*

Abstract Stability is a major requirement to draw reliable conclusions when interpreting results from supervised statistical learning. In this paper, we present a general framework for assessing and comparing the stability of results that can be used in real-world statistical learning applications or in benchmark studies. We use the framework to show that stability is a property of both the algorithm and the data-generating process. In particular, we demonstrate that unstable algorithms (such as recursive partitioning) can produce stable results when the functional form of the relationship between the predictors and the response matches the algorithm. Typical uses of the framework in practice would be to compare the stability of results generated by different candidate algorithms for a data set at hand or to assess the stability of algorithms in a benchmark study. Code to perform the stability analysis is provided in the form of an R package in the supplementary material that is available online.

Keywords: Resampling, Recursive Partitioning, R package **stablelearner**

2.1 Introduction

Influential statisticians have previously pointed out the importance of stability to draw reliable conclusions (or more generally speaking for reproducibility) when interpreting results from statistical learning (see, e.g., Stodden, 2015; Turney, 1995; Yu, 2013). Yu (2013), for example, stated: "More often than not, modern scientific findings rely on statistical analysis of high-dimensional data, and reproducibility is imperative for any scientific discovery. Scientific reproducibility therefore is a responsibility of statisticians." (p. 1485). To meet this demand in practical applications as well as in methodological research, we here present a framework that can be used for measuring the stability of results from statistical learning methods. In simple terms, stability is revealed when the interpretation of results generated by using different data sets drawn from some data-generating process (DGP) lead to identical or at least very similar conclusions.

In the following we will distinguish between the *algorithm* that is the general rule used for learning from data (such as a routine for estimating the coefficients in a logistic regression model or an algorithm for generating classification trees) and the *result* of applying the algorithm to a learning sample that is the fitted model or fitted rule (such as a fitted logistic regression model with estimated parameter values or a fitted classification tree with selected variables and cutpoints). Other terms used in the literature (although not consistently) for *algorithm* include terms like model, learner, or method, for *result* terms like fitted model, trained algorithm/learner, or classifier.

Algorithmic methods such as recursive partitioning, support vector machines, neural networks, and k-nearest neighbors, among many others, have become widely used in research and industry (Kuhn & Johnson, 2013). An established approach, known as *predictive modeling*, is to train an algorithm on a learning sample and then use it for predicting the response of new observations. In many applications, the focus lies on generating results with a high predictive performance. In substantive research, however, some algorithms are becoming more and more popular to achieve not only precise predictions but also in-depth knowledge about the relationship between the predictors and the response. Under this approach, that has been characterized as *explanatory modeling* (Shmueli, 2010), the goal is to learn about the nature of the DGP by interpreting the result. For discussions of different modeling cultures in statistics, we refer to Breiman (2001) and Shmueli (2010). In the present paper, however, we focus on the stability of results when conducting explanatory data analysis.

A popular method for explanatory modeling is recursive partitioning (see, e.g., Strobl et al., 2009, for an introduction). In recursive partitioning, the predictor space is repeatedly split to identify groups of observations with similar values in the response variable. The simplest cases are the well-known classification and regression trees (Breiman et al., 1984). More generally, however, the splitting can be conducted with

respect to the instability of the parameter values in a particular statistical model, for example, a generalized linear model (e.g., Rusch & Zeileis, 2013). This idea is implemented in the emerging model-based recursive partitioning method (Hothorn et al., 2006; Zeileis et al., 2008). The resulting partition of the predictor space and the estimated model parameters in each region can be used to explain differences between groups of observations. The partition is usually illustrated in the form of a decision tree that is easy to interpret. Therefore, tree-based methods are still very popular, although they have inferior predictive power compared to other algorithms for statistical learning or ensemble methods that have a lower interpretability.

When a single result is interpreted, the aim is to draw conclusions about the nature of the DGP. Therefore, stability is a major requirement, since we would expect to draw the same or at least very similar conclusions from interpreting results generated with a given algorithm on different random samples from the same DGP. Unfortunately, some algorithms are prone to generate unstable results (Breiman, 1996b). This means that small random changes in the data can cause large changes in the interpretation of a result. For these situations, it is important that we develop a means of quantifying the reproducibility of the results in terms of stability.

In this paper, we present a general framework to investigate and quantify the stability of results for both, when a data set (in real world supervised learning problems) or the DGP (in simulation studies) is available. The framework can support practitioners in judging the stability of a single analysis result or in choosing the most stable algorithm among a set of candidates with respect to a consistent interpretation of the results. In addition to this, it can also be used as a performance criterion in benchmark experiments (see Hothorn, Leisch, Zeileis, & Hornik, 2005). Code to perform the stability assessment in practice is provided in the form of a software package for the free open source software R for statistical computing (R Core Team, 2016) that is available in the supplementary material (version 0.1-2).

The remainder of the article is organized as follows. We first present our theoretical considerations and our new framework for measuring stability in Section 2.2, and describe a variety of specifications of the framework in Section 2.3. Section 2.4 contains simulation experiments to demonstrate some important properties of the framework and an illustrative example with real data sets how the framework can be used in practice. Section 2.5 concludes the article with a discussion and ideas for future research.

2.1.1 Related work

The stability of algorithms for statistical learning has been studied in statistics and machine learning for quite some time (see, e.g., Bousquet & Elisseeff, 2002; Breiman, 1996b; Mukherjee, Niyogi, Poggio, & Rifkin, 2006; Poggio, Rifkin, Mukherjee, & Niyogi,

2004). These articles identified algorithms that follow a certain definition of being "stable" and have provided important results for the development of new algorithms. According to Bousquet and Elisseeff (2002), for example, an algorithm is said to be "uniformly stable", if the (largest) change in the output due to small changes in the data is bounded and the bound decreases as $\mathcal{O}(n^{-1})$. Thus, this line of research mainly labels algorithms as being stable or not.

In our view, however, stability is not only a property of the algorithm but always depends on how well "what the algorithm learns matches to the DGP". Depending on the functional shape of the underlying relationship between the predictors and the response, for example, an algorithm may generate more or less stable results. One can think of a regression tree as the algorithm of interest. When this algorithm is trained on 100 data sets from a DGP in which the conditional distribution of the response depends on a smooth function of the predictors, it can only approximate this function and will thus give a different result every time. If however, the tree is trained on data sets from a DGP in which the conditional distribution of the response depends on a simple piecewise constant function of the predictors in a small predictor space, it will give virtually the same result in every tree. Additionally, the stability of the result depends on the model that was specified. One can imagine that a correctly specified model generates more stable results than an incorrectly specified model. In the present paper, we therefore claim that the stability of a result depends on several components, including the algorithm, the specified model, and the DGP and focus on empirically measuring the stability of a result from a practitioner's perspective.

Ideas for measuring the stability of results have been presented previously in Turney (1995), Lange, Braun, Roth, and Buhmann (2002), and more recently in Lim and Yu (2016). There are also approaches for measuring the stability especially for trees (see, e.g., Briand et al., 2009; Ntoutsi et al., 2008). These interesting concepts are all introduced either for the classification or the regression case. Unfortunately, neither of these approaches covers both cases in the same framework. Moreover, they are not general enough to be expanded to the emerging model-based recursive partitioning methods. Extending this work, we therefore introduce a very general framework for assessing the stability of results from statistical learning that is applicable to many situations. In particular, the procedure involves the pairwise comparison of results generated from learning samples randomly drawn from the original data set or – in the case of a simulation study – directly from the DGP. For the comparison of the results, a similarity (or dissimilarity) measure plus an additional evaluation sample will be required.

2.2 Stability measuring framework

Throughout this article, we focus on supervised learning problems by assuming predictor-response data $Z = (Y, X)$. Let Y denote the (possibly multivariate) response from some sample space $\mathcal{Y}$ and $X = (X_1, \ldots, X_p)$ the p-dimensional vector of predictors from some sample space $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_p$. For generality, let us assume some data-generating process (DGP) based on the joint probability $\mathsf{P}_Z = \mathsf{P}_{Y,X}$ that characterizes the population of interest.

We further assume that the conditional distribution of the response $\mathsf{P}_{Y|X}$ depends on a function f of the predictors. The goal in explanatory modeling is to reliably estimate f, such that we can draw conclusions about the relation between X and Y. To make an example, Y could represent the medical diagnosis, which is either positive ($Y = 1$) or negative ($Y = 0$) and X could be a vector of characteristics about the patient (age, gender, blood values, body mass index, etc.). When assuming a binomial distribution for the response, f could represent the unknown function that describes the relationship between the predictors and the conditional expectation of the diagnosis that is in this case,

$$\mathsf{E}(Y|X_1, \ldots, X_p) = \mathsf{P}(Y = 1|X_1, \ldots, X_p) = f(X_1, \ldots, X_p).$$

Further, let $\mathcal{A}$ be an algorithm for statistical learning. For example, an algorithm for recursive partitioning – such as CART (Breiman et al., 1984), conditional inference trees (Hothorn et al., 2006), or C4.5 (Quinlan, 1993) – or an algorithm to train neural networks, support vector machines, k-nearest neighbors, various types of regression models, etc.

Additionally, let $\mathcal{M}$ be the model that specifies the assumed relationship between the predictors and the response. The level of details required in $\mathcal{M}$ varies between algorithms. At minimum, however, $\mathcal{M}$ specifies the response and the predictor variables.

The goal in supervised learning is to approximate f by an algorithm $\mathcal{A}$ for the specified model $\mathcal{M}$ and a given learning sample $\mathfrak{L}' = \{z_1, \ldots, z_n\}$ generated by the DGP (where n is the number of observations). The result is a function denoted by $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}') = \hat{f}(x)$ that is an estimate of the unknown function $f(x)$ and can be used to predict the response for new instances of x. In a (univariate) regression problem, for example, the prediction of $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}')$ is a real value $\hat{y}$. In a (univariate) classification problem, $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}')$ predicts the class or class probabilities $\hat{\pi} = \{\hat{\pi}_k\}$ for $k = 1, \ldots K$, where K is the number of classes.

Now, suppose a different learning sample, say $\mathfrak{L}''$, was generated by the DGP. The result from learning the algorithm $\mathcal{A}$ for the model $\mathcal{M}$ on the learning sample $\mathfrak{L}'$ is denoted by $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}'')$. Due to the sampling variability in the learning samples, the

two results $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}')$ and $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}'')$ may be not the same. (Note that in addition to the sampling variability some algorithms add more randomness to the result, for example, by using random starting values). That is to say, both estimates of $f(x)$ could be slightly different and their predictions could vary (at least for some x). Hence, one can think of $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}')$ and $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}'')$ as realizations of a random function $r_{\mathcal{A},\mathcal{M}}(x)$ in the domain of functions that could result from learning on different samples generated by the DGP. The distribution of $r_{\mathcal{A},\mathcal{M}}(x)$ depends on several components; Most importantly on the algorithm, the specified model, the DGP, and the sample size n:

$$r_{\mathcal{A},\mathcal{M}}(x) \sim \mathsf{P}_r(\mathcal{A}, \mathcal{M}, \text{DGP}, n, \ldots).$$

In many practical situations, the result $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}')$ – for example a tree – is interpreted to learn about the DGP. Then, the question arises whether the interpretation of the result would have been the same if a different learning sample $\mathfrak{L}''$ had been used to generate it.

The basic idea discussed in this article is to assess the stability of a result by quantifying the similarity of realizations from the distribution P_r. Therefore, a measure is required to assesses the similarity[1] between the results generated by training the algorithm $\mathcal{A}$ on two different learning samples, $\mathfrak{L}'$ and $\mathfrak{L}''$. Large similarity indicates that two results lead to the same or similar conclusions about the underlying DGP and, thus, implies high stability. But before we present the details of the procedure to assess the stability of a result, we discuss how two results can be compared with respect to their interpretation.

2.2.1 Semantic versus structural similarity

The literature provides a vast number of semantic and structural measures to quantify the similarity between results (see, e.g., M. Banerjee, Ding, & Noone, 2012; Briand et al., 2009; Miglio & Soffritti, 2004; Ntoutsi et al., 2008; Shannon & Banks, 1999; Turney, 1995, for a selection of approaches only in recursive partitioning). Whereas semantic measures compare the logical meaning of two results by their prediction, structural measures compare the appearance of two results by their structural elements (e.g., by the nodes and the branches in a tree). For the purpose of comparing results with respect to their interpretation and the conclusion that can be drawn about the underlying DGP, we recommend that a semantic measure should be used.

[1]Note that the term similarity is related to other terms that have an inverse meaning, such as distance or divergence. For reasons of consistency, we will distinguish between similarity (proximity, agreement, correlation, etc.) and dissimilarity (distance, divergence, etc.) throughout the article.

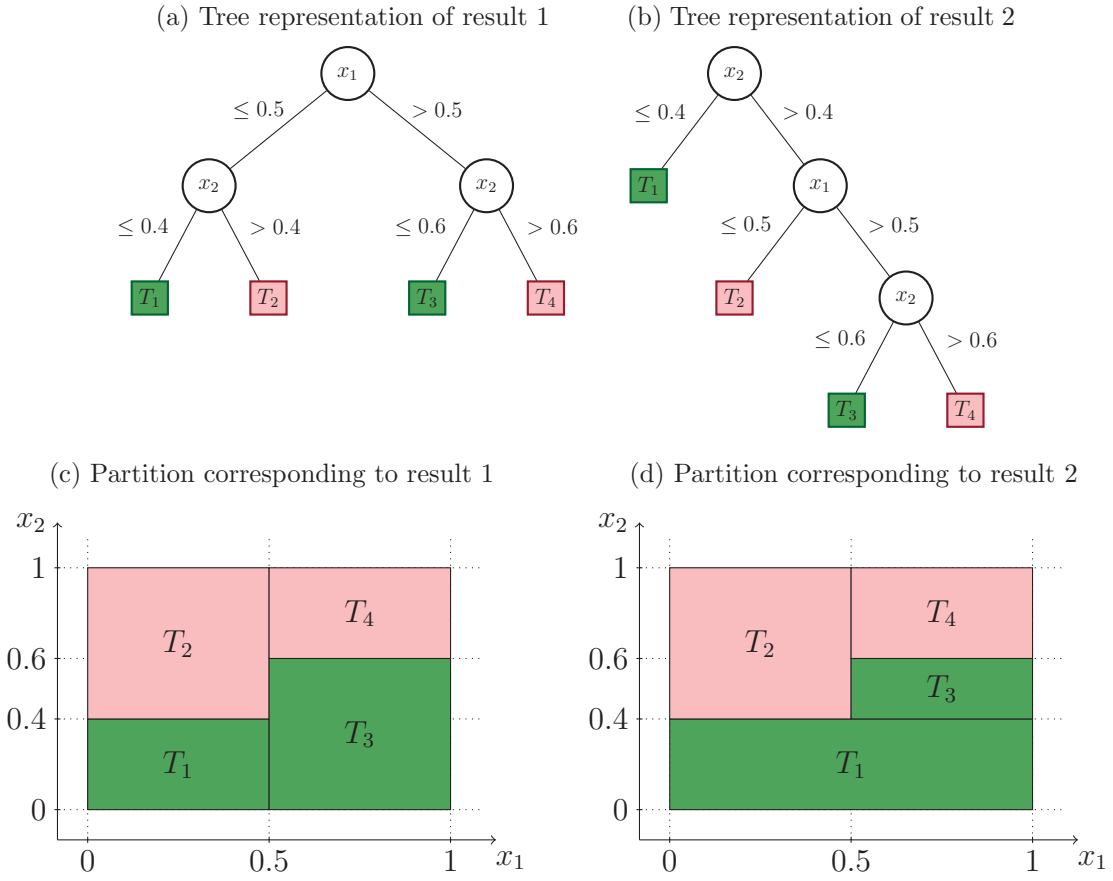


Figure 2.1: Examples of different tree structures, but equivalent partitions and interpretations.

This recommendation is based on our theoretical considerations that are in line with Turney (1995). As Turney elaborated in his work, two results that are structurally different can be logically equivalent and lead to the same interpretation. To illustrate this principle, we consider trees applied to a simple classification problem with two classes and a two-dimensional predictor space. Note, however, that the argument holds for more general cases and other algorithms as well. Figure 2.1 shows the two exemplary trees (first row) and the corresponding partitions of the predictor space (second row), where the predicted class in each terminal node is indicated by two different colors.

As one can see in the first row, the structures of the trees differ with respect to their shapes and the order in which the variables appear (in reality, they might additionally differ with respect to the cutpoints used for splitting). Nevertheless, the second row shows that the predictions are equivalent for any point in the predictor space. Thus, the two trees have the exact same semantic and lead to the same substantive conclusions.

In addition to those trees shown in Figure 2.1, there exist three more trees with exactly four leafs, which also have different structures, but correspond perfectly to the partitions shown in Figure 2.1. This example illustrates the fact that the same conclusions can be drawn from structurally different trees.

Note that, in addition to the argument above, fair comparisons of the stability of results between different algorithms are only feasible when a semantic similarity measure is used as well. Therefore, we recommend the use of a semantic, rather than a structural, similarity measure for assessing stability.

2.2.2 Measuring similarity based on predictions

As illustrated above, the similarity of two results is best assessed by comparing their predictions over the entire predictor space. Following Turney (1995), we thus propose to assess the agreement between two results of the same algorithm by means of a semantic similarity measure. We consider a similarity measure as a function $s(\cdot, \cdot)$ that generates real values. Measures that are particularly useful for computing the similarity of predictions in the context of regression and classification problems are discussed in Section 2.3.1.

To compute the similarity between the predictions of the two results, an evaluation sample is needed to generate predictions from the results. We suggest that more emphasis is given to areas in the predictor space where new observations are more likely to occur. This can be accomplished when the evaluation sample is drawn from P_X. In practice, however, stability might be of relevance for a particular region of the predictor space $\mathcal{X}$ (e.g., a group of high risk patients in a medical study). In this situation, the evaluation sample may also be defined specifically for that region.

Let $\mathfrak{E}$ be such an evaluation sample that contains m observations, that is $|\mathfrak{E}| = m$. Then the predictions of the results are given by

$$\hat{y}' = \{r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}') : x \in \mathfrak{E}\} \quad \text{and} \quad \hat{y}'' = \{r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}'') : x \in \mathfrak{E}\},$$

and their similarity can be calculated by $s\big(\hat{y}', \hat{y}''\big)$.

As mentioned above, both results are realizations of a random function. The similarity between the predictions of the two results can therefore again be seen as a realization of a random variable S. The domain of possible similarity values depends on the similarity function and via the prediction of the results on the algorithm, the specified model, the DGP, and the sample size. Thus, the distribution of the similarity is given by

$$S \sim \mathsf{P}_S(s(\cdot, \cdot), \mathcal{A}, \mathcal{M}, \text{DGP}, n, \ldots).$$

Finally, the stability of a result for a given algorithm, specified model and DGP can be assessed by studying characteristics (e.g., the first and second moments) of P_S.

Therefore, we suggest that P_S is approximated by generating many realizations $s(\hat{y}'_b, \hat{y}''_b)$ $(b = 1, \ldots, B)$ with the generic procedure described in Section 2.2.3.

2.2.3 Stability measurement procedure

Similar to Hothorn et al. (2005), we distinguish between the following two[2] situations:

- The *real data problem* (as encountered in real-world statistical learning applications or in benchmark experiments with real data sets) where the DGP is unknown and all information available is a fixed set of n observations $\{z_1, \ldots, z_n\} \sim \mathsf{P}_{Z_n}$, denoted as the original data set.
- The *simulation study problem* (as encountered in benchmark experiments with artificial data sets) where the DGP is known precisely, such that an arbitrary number of learning samples $\{z_1, \ldots, z_n\} \sim \mathsf{P}_{Z_n}$ can be generated by the DGP.

For empirically measuring stability, we suggest the following generic procedure. For iteration $b = 1, \ldots, B$,

(1) Generate two learning samples $\mathfrak{L}'_b$ and $\mathfrak{L}''_b$ plus an evaluation sample $\mathfrak{E}_b$ by sampling from $\mathcal{F}$ that is a proxy for P_Z. To do this in practice, the plug-in principle can be used. Thus, for the

$$\begin{aligned} \text{real data problem:} \quad \mathcal{F} &\mathrel{\widehat{=}} \widehat{F}_n, \\ \text{simulation study problem:} \quad \mathcal{F} &\mathrel{\widehat{=}} \mathsf{P}_Z, \end{aligned}$$

where $\widehat{F}_n$ is an approximation of P_Z that represents the DGP well when n is large.

(2) Generate the results $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}'_b)$ and $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}''_b)$ by training the algorithm on both learning samples.

(3) Compute $s(\hat{y}'_b, \hat{y}''_b)$ using the evaluation sample $\mathfrak{E}_b$.

Drawing samples from $\widehat{F}_n$ is equivalent to resampling from the original data set (see, e.g., Wasserman, 2004, chap. 8). Thus, resampling from the original data set can be used to generate the learning and evaluation samples in the real data problem.

[2] In practice, one might encounter a third situation where the DGP is unknown but a large number of independent learning samples are available from an ongoing data-generation pipeline as found in informatics (network data), finance, meteorology, and so on. In the context of measuring the stability of a result, this situation is closely related to the simulation study problem.

Following the notation above, this is comparable to sampling directly from the DGP, as carried out in the simulation study problem.

To apply the procedure in practice, users have to choose a similarity measure $s(\cdot, \cdot)$ and, for real data problems, a resampling and an evaluation method. Sensible choices are presented in the following section.

2.3 Framework settings

In the first part of the following section we discuss a few exemplary options of similarity and dissimilarity measures for regression and classification problems. In the second part of the section, different methods for generating the learning and the evaluation samples for the case of a real data problem are presented.

2.3.1 Similarity and dissimilarity measures

We distinguish between similarity and dissimilarity measures, denoted by $s(\cdot, \cdot)$ and $d(\cdot, \cdot)$, respectively. Most similarity and dissimilarity measures have a lower and/or an upper bound (e.g., $-\infty < s_{min} \leq s(\cdot, \cdot) \leq s_{max} < \infty$, where s_{min} and s_{max} are the lower and upper bounds of the measure, respectively). Some measures are additionally normalized, for example between -1 and 1 (most correlation measures) or between 0 and 1 (some distance measures). Normalized measures have the additional advantage that their range has an absolute meaning.

For reasons of comparability, dissimilarity measures may need to be converted into similarity measures. Nonnormalized dissimilarity measures may be converted into similarity using $s(\cdot, \cdot) = -d(\cdot, \cdot)$. To convert a normalized dissimilarity measure into a normalized similarity, it may be appropriate to use the upper bound as a reference, that is, $s(\cdot, \cdot) = d_{max} - d(\cdot, \cdot)$, where d_{max} is the upper bound of the dissimilarity measure.

A practically relevant task is the comparison of stability assessments from different results. To compare the stability of two or more results generated using data sets or DGPs, in which the responses were measured on different scales, we recommend to use a similarity measure that is invariant to changes in the scale. We denote a similarity measure as scale-invariant if $s(a + b \cdot \hat{y}', a + b \cdot \hat{y}'') = s(\hat{y}', \hat{y}'')$ for $a, b \neq 0$.

Please be aware that only *point-wise* similarity measures should be used for measuring stability in our framework, which guarantees that comparisons are made between predictions for the same point in the predictor space (as illustrated previously in Section 2.2.1).

Regression

Let us first consider the regression case. Thus, in the univariate case, $\hat{y}'$ and $\hat{y}''$ are numeric vectors of length m ($\hat{y}', \hat{y}'' \in \mathbb{R}$). The literature provides a vast number of similarity and dissimilarity measures used in diverse fields (e.g., in medicine, psychology, image registration, clustering, etc.) to compare numeric measurements.

The well-known *Euclidean distance* (ED) is a dissimilarity measure that can be used to compute the similarity of numeric predictions. It is computed as

$$d_{\text{ED}}(\hat{y}', \hat{y}'') = \sqrt{\sum_{i=1}^{m} (\hat{y}'_i - \hat{y}''_i)^2},$$

and has a lower bound of $d_{min} = 0$ that is approached if and only if $\hat{y}' = \hat{y}''$. A normalized version of the ED is available through the *Gaussian radial basis function* (GRBF) kernel that is commonly used in support vector machines to assess the proximity between two data points. It is defined as

$$s_{\text{GRBF}}(\hat{y}', \hat{y}''; \sigma) = \exp\left(-\frac{d_{\text{ED}}(\hat{y}', \hat{y}'')^2}{2\sigma^2}\right),$$

and can be interpreted as a similarity measure. σ is a free parameter that can be used to regulate the "sensitivity" of the measure, where smaller values lead to less similar predictions (i.e., $s_{\text{GRBF}}(\hat{y}', \hat{y}''; \sigma_1) < s_{\text{GRBF}}(\hat{y}', \hat{y}''; \sigma_2)$ when $\sigma_1 < \sigma_2$). Please note that, although this measure is normalized, it is not scale-invariant.

A scale-invariant option is the *concordance correlation coefficient* (CCC, Lin, Hedayat, Sinha, & Yang, 2002) that is a normalized version of the well-known *mean squared deviation* (MSD). It is defined as

$$s_{\text{CCC}}(\hat{y}', \hat{y}'') = \frac{2\sigma_{\hat{y}'\hat{y}''}}{\sigma^2_{\hat{y}'} + \sigma^2_{\hat{y}''} + (\mu_{\hat{y}'} - \mu_{\hat{y}''})^2},$$

and can be computed by plugging in the standard sample estimates for means, variances, and covariances. When using the divisor m (instead of $m - 1$) in the sample estimates, the values of the CCC are bounded within $[-1, 1]$, where the lower bound indicates perfect disagreement and the upper bound indicates perfect agreement between the predictions, respectively.

Other measures presented in the literature may also be used in our framework. A small selection of measures that we consider as useful for the regression case (without any claim to completeness) is provided in the supplementary material in Appendix A.1.

Classification

Let us now consider the classification case. A simple way to assess the similarity between two classification results is the *average class agreement* (ACA) between the predicted class labels, computed by

$$s_{\text{ACA}}(\hat{y}', \hat{y}'') = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\hat{y}_i' = \hat{y}_i''},$$

where $\mathbb{1}$ is the indicator function. By definition, it is normalized between 0 and 1. In case of unbalanced response classes, the *Kappa statistic* may be used to additionally account for their distribution (see, e.g., Kuhn & Johnson, 2013, chap. 11).

For a probabilistic classification result, however, much more detailed information is provided by the predicted class probabilities. Therefore, we suggest using a more precise similarity estimation via

$$s(\hat{\pi}', \hat{\pi}'') = 1 - \frac{1}{m} \sum_{i=1}^{m} \delta(\hat{\pi}_i', \hat{\pi}_i''),$$

where $\delta(\cdot)$ is a distance measure between two discrete probability distributions (see, e.g., Cha, 2007, for a survey on distance measures between probability distributions).

A small selection that we consider as useful and computationally feasible distance measures for discrete probability distribution (without any claim to completeness) is provided in the supplementary material in Appendix A.1. A well-known version is the *total variation distance* (TVD) that is computed via

$$\delta_{\text{TVD}}(\hat{\pi}_i', \hat{\pi}_i'') = \frac{1}{2} \sum_{k=1}^{K} |\hat{\pi}_{ik}' - \hat{\pi}_{ik}''|.$$

Due to the l_1-norm, it is more sensitive to small changes in the predicted class probabilities compared to other distance measures.

2.3.2 Resampling and evaluation methods

Let us consider the real data problem where the DGP is unknown. To assess the stability in this situation, two samples $\mathfrak{L}_b'$ and $\mathfrak{L}_b''$ for generating the results and a third sample $\mathfrak{E}_b$ for evaluating the similarity have to be generated in each iteration by resampling from the original data set $\mathfrak{L} = \{z_1, \ldots, z_n\}$ (as previously described in Section 2.2.2).

Learning overlap

An important aspect concerns the intersection between the learning samples, given by $\mathfrak{L}'_b \cap \mathfrak{L}''_b$, that we call the *learning overlap* as opposed to the *evaluation overlap* (see below).

In principle, the similarity of the predictions from two results depends on the similarity of the observations in the learning samples that are used to generate it (see, e.g., Ntoutsi et al., 2008). It is to be expected, however, that the similarity of the predictions is additionally affected when observations are sampled into both learning samples, that is, when $\mathfrak{L}' \cap \mathfrak{L}'' \neq \emptyset$. One can think of the jackknife approach as the most extreme form of learning overlap, since except for a single observations, both samples share the same observations. In this case, the focus would rest on the *robustness* of a result with respect to small changes in the given learning sample. With less learning overlap, the focus shifts towards the *generalizability* of a result for independent draws from the DGP.

The learning overlap varies between the following exemplary resampling methods:

- Bootstrap sampling: We refer to bootstrap sampling as randomly drawing r observations with replacement from $\mathfrak{L}$, which is done twice per iteration to generate two learning samples. The probability that observation z_i appears in both samples can be shown to be $\left[1 - (1 - \frac{1}{n})^r\right]^2$. If $r = n$, the expected size of the overlap between $\mathfrak{L}'_b$ and $\mathfrak{L}''_b$ is $\approx 40\%$ of the size of $\mathfrak{L}$ and decreases if $r < n$. Thus, the overlap can be reduced by choosing $r < n$.

- Subsampling: We refer to subsampling as randomly drawing r observations without replacement from $\mathfrak{L}$, which is done twice per iteration to generate two learning samples. The probability that observation z_i appears in both samples is given by $\left[\frac{r}{n}\right]^2$ and is obviously 1 if $r = n$. For all r, subsampling generates a larger learning overlap than bootstrap sampling, while the size of the learning samples is always smaller.

- Splithalf: We refer to splithalf sampling as splitting the learning sample into two disjoint sets of observations. To generate $\mathfrak{L}'_b$, sample $\lfloor \frac{n}{2} \rfloor$ observations without replacement from $\mathfrak{L}$. Then, let $\mathfrak{L}''_b = \mathfrak{L} \setminus \mathfrak{L}'_b$. The overlap is zero by definition. However, at the same time, the size of the learning samples is halved compared to the bootstrap approach.

Hence, the resampling methods differ with respect to the sample size and the overlap of the learning samples.

Evaluation overlap

Another aspect concerns the sample $\mathfrak{E}_b$ used for evaluating the predictions and its overlap with the learning samples, given by $\mathfrak{E}_b \cap \{\mathfrak{L}'_b \cup \mathfrak{L}''_b\}$, that we call the *evaluation overlap*. It corresponds to the set of observations in $\mathfrak{E}_b$ previously used for generating either result. This is an important issue when the aim is prediction error estimation: In order to avoid overfitting, the learning and test sample should not overlap and there is always a tradeoff between keeping enough observations in the learning sample for precisely estimating the model versus reserving enough observations as a test sample for precisely estimating the prediction accuracy (see, e.g., Molinaro, Simon, & Pfeiffer, 2005). Since in our framework for stability assessment, however, we do not compare the predictions with the true response but only two predictions with each other, the evaluation overlap (related to the detectability of overfitting) is less of an issue, as will be illustrated in Section 2.4.3.

The evaluation overlap varies between the following exemplary evaluation methods:

- In-sample (ALL): The complete set of observations available is used for evaluating the similarity (i.e., $\mathfrak{E}_b = \mathfrak{L}$). How many of the observations in this sample were previously used for generating the results depends on the resampling method.
- Out-of-bag (OOB): The observations not used for learning either of the two results are used for evaluating the similarity (i.e., $\mathfrak{E}_b = \mathfrak{L} \setminus \{\mathfrak{L}'_b \cap \mathfrak{L}''_b\}$). Hence, none of the observations in the evaluation sample was also present in any of the learning samples.
- Out-of-sample (OOS): A completely separate set of observations is used for evaluating the distance (i.e., $\mathfrak{E}_b \cap \mathfrak{L} = \emptyset$), which could result from separating the original data set into a learning and a test sample. None of the observations used in the evaluation sample was used for generating any of the results in the entire stability assessment procedure. Note that with this method, fewer observations are available for generating the results in the first place.

Hence, the evaluation methods differ with respect to the sample size and the overlap of the evaluation sample with the learning samples.

The impact of the different combinations of resampling and evaluation methods will be illustrated by the simulation experiments presented in Section 2.4.3. The most strict way to investigate the practically relevant question of how dissimilar the interpretation of a result could be when a different learning sample was used for training, is when there is no learning and evaluation overlap and when the size of the learning and the evaluation samples remain equal to the size of the original data set. Later on, we therefore consider the stability assessed by means of independent draws from a known

DGP (as carried out in the simulation study problem) as the "reference stability"; that is, as the gold standard. In situations where the DGP is unknown, we aim to come as close as possible to the reference.

Reweighting

A computationally efficient way for implementing the resampling in practice is by reweighting the observations in the original data set $\mathfrak{L}$. Instead of generating the learning samples $\mathfrak{L}'_b$ and $\mathfrak{L}''_b$ by resampling from the original data set, the n-dimensional vectors $w'_b = \{w'_{ib}\}$ and $w''_b = \{w''_{ib}\}$ $(i = 1, \ldots, n)$ that contain nonnegative case-weights are defined. The case-weights of observation i are given by

$$w'_{ib} = \#\{i : x_i \in \mathfrak{L}'\} \quad \text{and} \quad w''_{ib} = \#\{i : x_i \in \mathfrak{L}''\}.$$

By means of these case-weights, all common resampling schemes can be represented. For example, the case-weights $w'_{ib}, w''_{ib} \in \{0, 1\}$ apply for subsampling and splithalf sampling and the case-weights $w'_{ib}, w''_{ib} \in \{0, 1, 2, 3, \ldots\}$ apply for bootstrap sampling.

The results $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}, w'_b)$ and $r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}, w''_b)$ are then generated using w'_b and w''_b, respectively, and the predictions are given by

$$\hat{y}' = \{r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}, w'_b) : x \in \mathfrak{L}\} \quad \text{and} \quad \hat{y}'' = \{r_{\mathcal{A},\mathcal{M}}(x; \mathfrak{L}, w''_b) : x \in \mathfrak{L}\}.$$

However, whether case-weights can be applied in practice depends on the algorithm and its software implementation.

To implement a specific evaluation method or restrict the analysis to a particular region of the predictor space, one can define $w^e_b = \{w^e_{ib}\}$ $(i = 1, \ldots, n)$ that is also a n-dimensional vector with nonnegative case-weights. The predictions used for estimating the similarity $s(\hat{y}'_b, \hat{y}''_b)$ are now selected from $\hat{y}'$ and $\hat{y}''$ according to the counts in w^e_b.

2.4 Simulation and benchmark experiments

In the next section, we will first describe the architecture of the DGPs that were used for the simulation experiments. Then, we will present the results of two small simulation studies. In the first study, we analyzed the effect of different characteristics of the DGP on the stability of a result. In particular, we illustrate our claim that the stability of a result is a property of both the algorithm and the DGP, not the algorithm alone. In the second study, we analyzed the effect of different combinations of resampling and evaluation methods on the stability of a result to investigate our expectation that a

larger learning overlap leads to higher similarity values. The section ends with a small benchmark experiment to illustrate how the framework can be applied in practice.

We restricted our analyzes to binary classification problems and two methods that are widely used in statistical learning, recursive partitioning and logistic regression. Besides the fact that both methods can be used for classification and produce interpretable results, they are also very different. Whereas it is widely known that tree-algorithms generate unstable results, logistic regression is known as a stable method. Moreover, unlike logistic regression, the original tree-algorithms are not based on a stochastic model for the data and, thus, do not provide statistical inference procedures. In our analyzes, we used the following implementations available in R: The function **ctree()** from the **partykit** package for conditional inference trees and the function **glm()** with `family = "binomial"` for logistic regression. For both algorithms, the default settings were used.

2.4.1 Data-generating processes

Our aim was to investigate the impact of several characteristics of the DGP on the stability assessment in the framework. In a preliminary study (results not shown for brevity) we observed notable impact for the sample size, the distribution of the response classes, and the dimension of the predictor space for different algorithms. Additionally, we were interested in the effect of different forms of the underlying function describing the association between the predictors and the conditional distribution of the response. Therefore, the DGPs were set up as follows:

Dimensionality For all DGPs, the predictor variables were sampled from a multivariate standard normal distribution, $x_i \sim \mathcal{N}_p(0, I_p)$, where p was the number of predictor variables comprising $q \leq p$ signal and $p-q$ noise variables. The dimensionality was chosen at $p = 20$ and $p = 40$, from which the first $q = 4$ and $q = 8$ were signal variables, respectively.

Functional form The binary response was sampled from a Bernoulli distribution with a conditional probability of success given by

$$\pi_i = \mathsf{P}(Y_i = 1|x_i) = \text{logit}^{-1}\left(\beta_0 + \sum_{j=1}^{q} f(x_{ij})\right).$$

The intercept β_0 specifies the baseline probability and was used to control the class distribution (see below). Two different functions were selected for $f(x)$: The identity function, $f(x) = x$, and the signum function, $f(x) = \text{sgn}(x)$. The choice of $f(x)$ was

motivated by the algorithms considered for the simulations. Whereas logistic regression can model linear effects exactly as simulated by the DGPs with the identity function, trees model nonlinear effects using piecewise constant functions exactly as simulated by the DGPs with the signum function. Thus, we had DGPs with a functional form that matched perfectly to either algorithm.

Sample size Different sample sizes were selected to illustrate asymptotic behavior. For presentation purposes, the samples sizes were selected according to the binary logarithm:

$$n = 2^7 (= 128),\ 2^8 (= 256),\ \ldots,\ 2^{15} (= 32\,768).$$

Class distribution Three different class distributions were investigated for the response: Equally balanced classes (50%/50%), weakly unbalanced classes (30%/70%) and strongly unbalanced classes (10%/90%). To approximately achieve these allocations, the intercept β_0 was chosen at 0, 1.417, and 3.447 for the lower dimensionality ($p = 20$) and at 0, 1.774, and 4.315 for the larger dimensionality ($p = 40$).

2.4.2 Study 1: Impact of the DGP

Reference stability

In the first study, we investigated the reference stability of results generated by training **ctree** and **glm** on DGPs that differed by the sample size, the dimension of the predictor space, the class distribution, and the functional form. We first illustrate and discuss the results when the correct algorithm was selected for the functional form of the DGP. We here present and discuss only the results for **ctree**; the results for **glm** can be found in the supplementary material in Appendix A.2. The findings in case the algorithm did not match to the functional form of the DGP are illustrated and discussed separately in Section 2.4.2 for both algorithms.

The results for **ctree** are illustrated in Figure 2.2. The panels separate the three different class distributions, the two dimensions of the predictor space are displayed in different colors, and the sample sizes are depicted on the x axis. To estimate the similarity distribution as precisely as possible, the procedure was repeated $B = 5000$ times (in practice, however, fewer iterations are sufficient). Thus, for each condition, 5000 similarity values were computed. The dashed lines correspond to the median similarity. The shaded areas mark the range between the lower and upper quartile of the similarity distribution in each DGP. The similarity was computed using the total variation distance as described in Section 2.3.1.

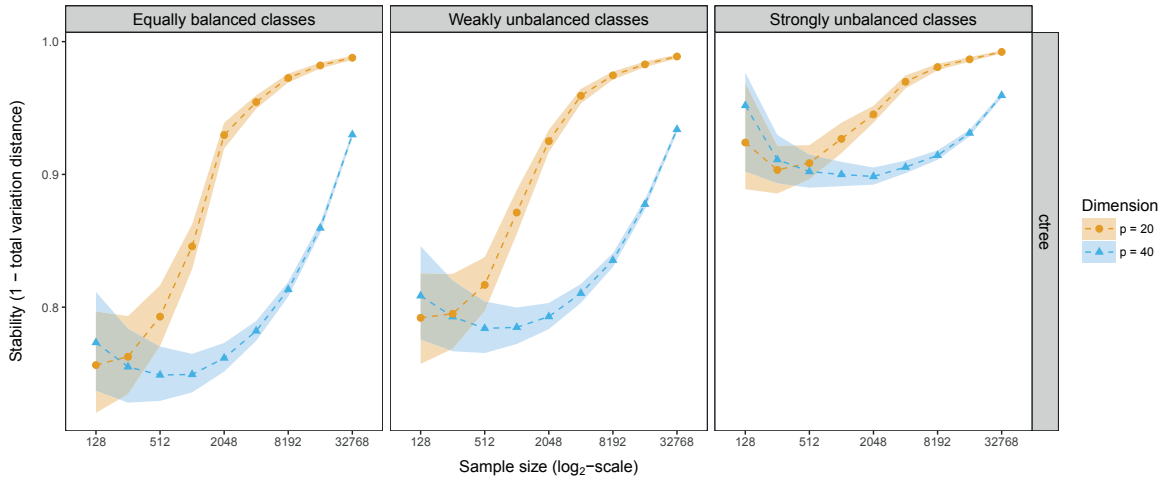


Figure 2.2: Stability assessment of results generated by training the algorithm **ctree** on DGPs based on the signum function, but with different sample sizes, distribution of the response classes and dimension of the predictor space. The dashed line corresponds to the 50%-quantile of the estimated similarities. The highlighted area marks the simlarities between the 25%- and the 75%-quantile.

First of all, the results became more stable as the sample size increased, as was to be expected. For the DGPs with lower dimension ($p = 20$, depicted in orange color) the similarity values converged towards the upper bound of the similarity measure. For the largest sample size considered, the **ctree** results were almost perfectly stable. For the DGPs with larger dimension ($p = 40$, depicted in blue color) the convergence was slower and never reached the maximum similarity within the examined sample sizes.

The stability was smaller for DGPs with balanced classes than for DGPs with unbalanced classes. This can be explained by the declining range of values from which the conditional probabilities were sampled in the unbalanced case, where the baseline probability was higher. Thus, any differences between the predictions of the results became smaller and the stability correctly increased.

In all scenarios, the variation among the similarity values declined as the sample size increased. Additionally, for $p = 40$, the similarity values were higher for small samples, then decreased for medium-sized samples before they again increased for large samples (see "U-shape" for $p = 40$). Both artifacts are caused by the circumstance that trees with fewer splits (resulting for small samples) were generally more stable than trees with more splits (resulting for medium-sized samples), before the trees again became more stable due to the increased sample size. Trees with fewer splits had, on the other hand, a weaker prediction accuracy. This highlights that – perhaps counterintuitively – a high stability does not always go hand in hand with a high prediction accuracy, so that a measure of stability can give additional information, as we will also show in the application example in Section 2.4.4.

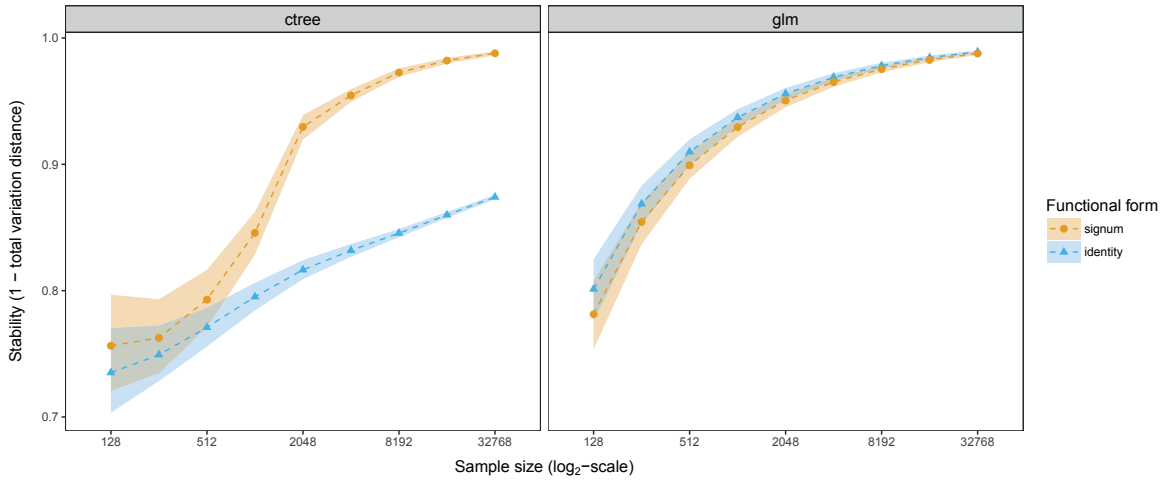


Figure 2.3: Stability assessment of DGPs with different functional forms (signum and identity) and two algorithms (**ctree** and **glm**) for different sample sizes. The dashed line corresponds to the 50%-quantile of the estimated similarities. The highlighted area marks the simlarities between the 25%- and the 75%-quantile.

Match between DGP and algorithm

Further, we investigated how the match between the functional form of the DGP and the algorithm affects the stability of a result. For a better presentation, the study was restricted to DGPs with equally balanced classes and lower dimensionality ($p = 20$). The results for the remaining conditions can be found in the supplementary material in Appendix A.2.

The results are illustrated in Figure 2.3 using again the medians as well as the lower and the upper quartiles of the similarity distributions. The left panel shows the stability of results generated from **ctree** and the right panel for results generated from **glm**, respectively. The two different functional forms of the DGPs are depicted in orange (identity function) and blue (signum function). Thus, in the left panel, the **ctree** algorithm perfectly matched to the DGPs with the signum function illustrated in orange. In the right panel, however, the **glm** algorithm perfectly matched to the DGPs with the identity function illustrated in blue.

At first sight, we find that for both algorithms the DGP with the corresponding functional form leads to more stable results. Since the DGPs with different functional forms are not directly comparable, however, minor differences (as in the right panel for **glm**) should not be overinterpreted. Still, we do see clearly that the stability of **ctree** is much higher for the DGPs with the piecewise constant signum function (see left panel). This supports our expectation that the stability is a property of both the algorithm and the functional form of the DGP.

Table 2.1: Resampling and evaluation methods used in the simulation study. The columns three and four show the expected size of the learning sample and the expected learning overlap; the columns five and six show the expected size of the evaluation sample and the expected evaluation overlap. The values are given relative to the original sample size n or to the size of the learning samples r; the expected learning overlap in column four is for completeness additionally given relative to n (in parentheses).

Resampling	Evaluation	Learning $\mathfrak{L}', \mathfrak{L}''$		Evaluation $\mathfrak{E}$	
		$\frac{\lvert\mathfrak{L}'\rvert}{n} = \frac{\lvert\mathfrak{L}''\rvert}{n}$	$\frac{\lvert\mathfrak{L}' \cap \mathfrak{L}''\rvert}{r}$	$\frac{\lvert\mathfrak{E}\rvert}{n}$	$\frac{\lvert\mathfrak{E} \cap \{\mathfrak{L}' \cup \mathfrak{L}''\}\rvert}{r}$
Bootstrap sampling	Out-of-bag	100%	$\approx$ 40% (40%)	$\approx$ 13.5%	0%
Bootstrap sampling	In-sample	100%	$\approx$ 40% (40%)	100%	$\approx$ 86.5%
Bootstrap sampling	Out-of-sample	75%	$\approx$ 40% (30%)	25%	0%
Subsampling	Out-of-bag	80%	$\approx$ 80% (64%)	$\approx$ 4%	0%
Subsampling	In-sample	80%	$\approx$ 80% (64%)	100%	$\approx$ 96.0%
Subsampling	Out-of-sample	60%	$\approx$ 80% (48%)	25%	0%
Splithalf sampling	In-sample	50%	0% (0%)	100%	100%
Splithalf sampling	Out-of-sample	37.5%	0% (0%)	25%	0%

2.4.3 Study 2: Impact of resampling and evaluation methods

Method comparison

In the second study, we investigated the impact of eight different combinations of resampling and evaluation methods listed in Table 2.1 using the following procedure for each algorithm and DGP:

1. First, we assessed the reference stability for the selected algorithm and the DGP and computed the mean of the corresponding similarity distribution, denoted by $\bar{s}_0$.
2. For $l = 1, \ldots, 100$ we repeated the following steps:
   (a) Draw a learning sample $\mathfrak{L}$ from the DGP and generate the result $r_a(x; \mathfrak{L})$.
   (b) Assess the stability of $r_a(x; \mathfrak{L})$ as described in Section 2.2.3 with $B = 500$ using the resampling combinations listed in Table 2.1.
   (c) For each combination, compute the mean of the corresponding similarity distribution, denoted by $\bar{s}_l$.
   (d) Compute the mean difference between $\bar{s}_l - \bar{s}_0$.

Thus, for each combination, we estimated 100 mean differences to the reference stability that are illustrated by means of boxplots in Figure 2.4. In this illustration, values below zero imply that the mean of the reference stability was underestimated. Results are

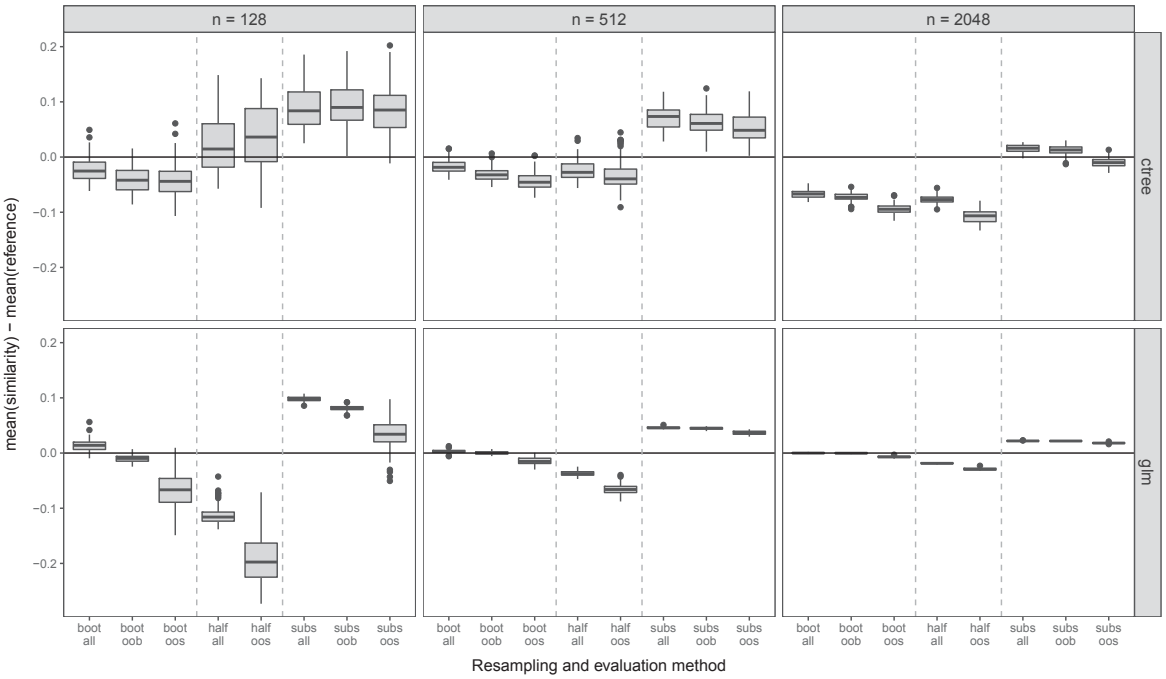


Figure 2.4: Boxplots of 100 differences between the mean of the reference similarity distribution and the mean of similarity distributions generated with different combinations of resampling (boot = bootstrap sampling, half = splithalf sampling, subs = subsampling) and evaluation methods (all = in-sample, oob = out-of-bag, oos = out-of-sample). The upper and the lower row represent the results for ctree (for DGPs with indicator function) and glm (for DGPs with signum function), respectively.

shown for DGPs with equally balanced response classes, lower dimensionality, and the functional form to which either algorithm perfectly matched. The results for the remaining conditions can be found in the supplementary material in Appendix A.2. In the upper row, the mean differences are illustrated for **ctree** and in the lower row for **glm**, respectively. The columns separate the three different samples sizes and the resampling combinations are illustrated in the x direction of the graph. For computational reasons, this study was only conducted for samples sizes $n = 128$, 512, and 2048. Recall that the learning and the evaluation overlaps as well as the effective sample sizes for learning and evaluation differ between the methods (see Table 2.1).

For **glm** (see lower row), the results were in line with our expectations. First, the stability increased with a higher learning overlap: *half* < *boot* < *subs*. Second, the stability decreased with increasing size of the evaluation sample: *oob* > *oos* within the same resampling method. Finally, the stability (mostly) remained constant for increasing evaluation overlap: *oob* ≈ *all* within the same resampling method. Overall, the mean reference stability $\bar{s}_0$ was captured well by bootstrap sampling (*boot*) with in-sample (*all*) or out-of-bag (*oob*) evaluation. With splithalf sampling (*half*) and subsampling (*subs*), on the other hand, the reference stability was under- or overestimated within the investigated sample sizes.

For **ctree** (see upper row), the results were not so clear and none of the combinations approached the mean reference stablity $\bar{s}_0$ reasonably well over all sample sizes. This might be due to a confounding between the properties of the resampling combination and the trees' tuning parameters.

Therefore, the experiment did not reveal one generally optimal resampling combination, but did highlight that framework-specific factors influence the stability assessment and should be kept constant across studies for comparable results.

Note that the resampling combinations we investigated have the advantage that they are widely known and practically applicable to real data problems. However, they do not allow us to single out individual effects, for example, of the learning and the evaluation overlap, because these are confounded with each other, as illustrated in Table 2.1. Therefore, we will further investigate individual effects in an artificial setting that does not correspond to any practically applicable resampling or evaluation method, in the next section.

Learning and evaluation overlap

To demonstrate that the stability is largely affected by the learning overlap, but much less by the evaluation overlap, we conducted a controlled computer experiment in which each overlap was systematically and individually manipulated to assess its partial impact. We only performed this analysis for DGPs with the lower dimensionality ($p = 20$) and equally balanced classes.

In each of $l = 1, \ldots, 100$ repetitions we first drew an original data set $\mathfrak{L}$ of size $n = 1536$ ($= 3 \times 512$) from the DGPs with the functional forms matching to the algorithms **ctree** and **glm**. Next, the stability was assessed for two results: One generated by means of an overspecified model (using all predictor variables) and another generated by means of a correctly specified model (using only signal predictors). Then, in each of the $b = 1, \ldots, 500$ iterations of the stability assessment, we assembled the observations from the original data set $\mathfrak{L}$ into the three samples $\mathfrak{L}'_b$, $\mathfrak{L}''_b$ and $\mathfrak{E}$ of size $r = 512$, such that either the learning overlap or the evaluation overlap ranged from 0%, 10%, 20%, etc. to 100% (relative to the size of the learning and evaluation samples). Thus, in this study, the reference stability corresponded to the situation with 0% overlap. Finally, the median similarity was computed in each repetition.

Thus, we ended up with a distribution of 100 median values from which again the median as well as the lower and the upper quartile are illustrated in Figure 2.5. The different overlaps are depicted on the x-axis. The colors orange and blue distinguish the correctly specified from the overspecified model.

We consider first the left panel that illustrates the results for the varying learning overlap, while the evaluation overlap was kept constant at zero. For both algorithms and

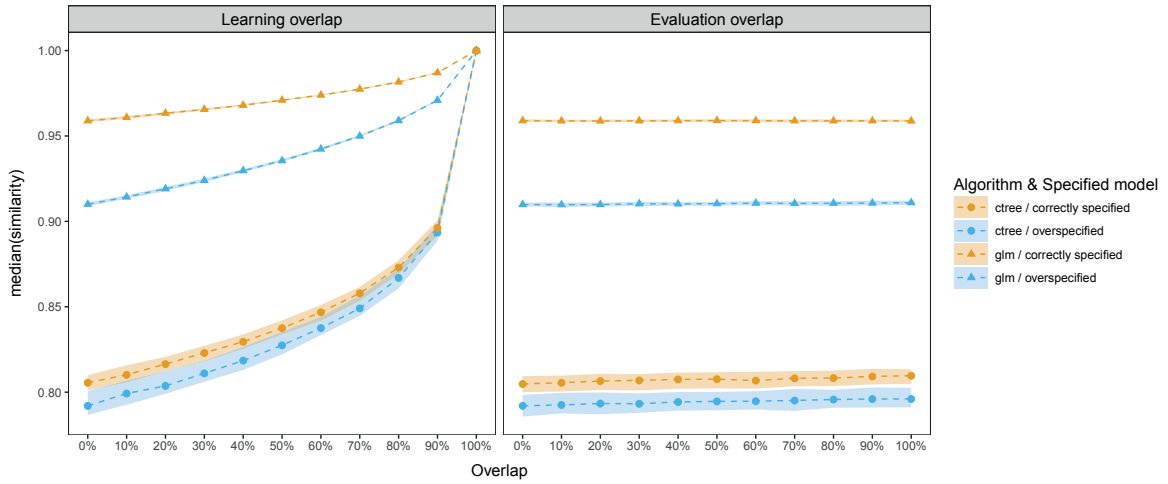


Figure 2.5: Stability assessment of results generated by experimentally manipulating the learning and the evaluation overlap. The dashed line corresponds to the 50%-quantile of $\bar{s}_l$ for **ctree** (cirlces) and **glm** (triangles) and the colors distinguish the specified models. The highlighted area marks values between the 25%- and the 75%-quantile.

models and with increasing overlap, the stability converged as expected towards the upper bound of the similarity measure (which was 1.00 in the case of the total variation distance used here). Although the largest increase in stability was between 90% and 100%, we also observed substantial differences between 0% (corresponding to the overlap for splithalf sampling) and 80% (corresponding to the overlap for subsampling).

Thus, in practice the learning overlap should be kept as low as possible in order to avoid large distortions from the reference stability. However, splithalf sampling, although without overlap, is not recommended because the size of the learning sample is significantly reduced, which also affects the stability assessment for small to medium-sized data sets. Therefore, taking all our results into account, we recommend bootstrap sampling as a good compromise.

Moreover, the results show that the stability was generally higher for **glm** (filled triangles) than for **ctree** (filled points) as previously demonstrated in Section 2.4.2. A new insight, that was in line with our expectation, was that the stability was higher with the correctly specified model (illustrated in orange) than for the overspecified model with additional noise variables (illustrated in blue). This is true for both algorithms, but note that the **ctree** algorithm showed overall lower stability but was less affected by noise variables due to its automatic variable selection.

Now consider the right panel that illustrates the results for the varying evaluation overlap, while the learning overlap was kept constant at zero. With increasing overlap, the stability remained (almost) constant for both algorithms and models. A very small and negligible increase can be detected with **ctree** for both models and with **glm** for the overspecified model. According to preliminary results (not shown for brevity), this

Table 2.2: Median and IQR of stability and accuracy values for **ctree** on well-known classification problems. Results are listed in decreasing order of the median stability. Legend: n = sample size, p = number of predictors, K = number of classes.

Data set	n	p	K	Stability		Accuracy		CPU time [sec]
				Median	IQR	Median	IQR	
Iris	150	4	3	0.956	0.048	0.919	0.146	4.7
Breast Cancer	699	9	2	0.933	0.029	0.864	0.080	13.4
Titanic	1317	7	2	0.925	0.027	0.561	0.086	13.6
Ionosphere	351	34	2	0.900	0.058	0.788	0.132	10.8
Pima	768	8	2	0.835	0.036	0.403	0.130	13.8
Satellite	6435	36	6	0.819	0.013	0.363	0.025	631.3
Sonar	208	60	2	0.728	0.098	0.412	0.231	12.6
Vehicle	846	18	4	0.723	0.050	0.552	0.084	50.3
Glass	214	9	6	0.694	0.107	0.219	0.185	12.9

increase in slightly more pronounced for smaller samples sizes. Thus, compared to the issue of overfitting when assessing prediction accuracy (see comment in Section 2.3.2), the evaluation overlap has only little impact on the stability assessment.

2.4.4 Benchmark experiment

To illustrate the stability assessment in a practical scenario, we trained **ctree** on eight well-known benchmarking problems for classification from the UC Irvine meachine learning repository (Lichman, 2013) and the Titanic data from the R package **stablelearner**. The stability was assessed via the total variation distance as described in Section 2.3.1 with bootstrap sampling and out-of-bag evaluation using $B = 500$ iterations. For comparison, the prediction accuracy was assessed simultaneously via the *Kappa statistic* that takes the original distribution of the response class into account (see, e.g., Kuhn & Johnson, 2013, chap. 11). The values zero/one indicate no/perfect agreement between the true and the predicted class. A computer with a four core Intel i7-2600 processor running with 3.7 GHz in total and 8 GB RAM on a 64-bit Linux (Ubuntu 14.04.5 LTS) operating system was used to perform the analyzes. The results are given in Table 2.2. For each data set, the table lists the median and the interquartile range (IQR) of the stability and the accuracy values, as well as the CPU time in seconds and additional information about the data sets.

Generally, the results were more stable on data sets with fewer predictors and response classes. The result on the Ionosphere data set is a notable exception from this observation. Moreover, for most problems, the stability increased along with the accuracy. However, as mentioned earlier, there might be exceptions, as shown here exemplary by

the results for the Titanic and the Pima data set. In practice, we conclude, one would be relatively safe to interpret a tree generated by **ctree** for the Iris data set, whereas for the Glass data set it is not recommended to overinterpret the result.

Finally, the stability assessment was relatively fast for small data sets but took clearly longer for large data sets (more than ten minutes for the Satellite data set). Note that the computation time would also vary largely between algorithms.

2.5 Discussion

2.5.1 Summary and recommendations

Stability is an important property for drawing reliable conclusions from a statistical learning result because the interpretation of an unstable result can lead to wrong conclusions and errors in decision making. Moreover, many algorithms do not provide measures of stability for their results. Thus, for practitioners, it is crucial to be able to judge the stability of their results before interpreting them.

In this article, we have presented a framework to assess the stability. It can be used in real-world statistical learning applications to assess the stability of a result for a given algorithm and data set. Moreover, the framework can be applied in benchmark experiments to either compare the stability of results generated by training different algorithms on a certain data set or to compare the stability of results generated by training different data sets on a certain algorithm. Instead of real data sets, one can also assess the stability of results for artificial DGPs.

The stability is assessed by repeatedly computing the similarity between two results generated with learning samples drawn directly from the DGP or by resampling from the original data set. Therefore, a measure of similarity is required. To compute the similarity of the substantial meaning between the results, we strongly recommend to use a measure that compares two results by their predictions and not by their structure. We have presented a few exemplary measures for classification and regression problems. We would like to emphasize that our selection is not complete and that the framework can be combined with other measures as well. Note also that by choosing an appropriate similarity measure, the framework could be extended to unsupervised learning applications.

Unlike previous research, our aim was not a theoretical investigation of the stability of learning algorithms alone. Instead, our motivation was to measure the stability of the result of an algorithm trained on a specific learning sample. Thereby, we have shown by our reasoning and in a simulation experiment that the stability of a result is not only a property of the algorithm alone but also of the DGP that has generated the learning sample.

In an exemplary illustration, we have further studied the impact of different data characteristics (sample size, dimensionality, and class distribution) on the stability of results generated by logistic regression and conditional inference trees for binary classification. We observed a large impact on the stability for all investigated factors. Although we do not claim that our results can be generalized to other algorithms or DGPs beyond the simulation study, they provide useful insights into the mechanisms of the stability assessment itself.

As a side note, we would like to address the wide-spread opinion that the correlation among the predictor variables decreases the stability. This may be correct when a structural similarity measure is used. For a semantic similarity measure, however, this need not be the case. According to our preliminary studies (results not shown for brevity), the stability can even slightly increase with correlated variables. A possible explanation for this is that a narrower distribution of the data in the predictor space lead to less variable solutions. This, however, needs to be further investigated in future research.

A practically relevant task is the comparison of stability assessments from different results. Such comparisons should only be conducted when either the response variables were measured on the same scale – which is generally the case for classification but not necessarily for regression problems – or when a scale-invariant similarity measure is used.

When assessing the stability of a result for a real data set, a resampling and an evaluation method must additionally be selected. We have investigated different resampling and evaluation methods for logistic regression and conditional inference trees. Our investigations revealed that none of the methods performed best in all simulation scenarios. Hence, there is nothing like the one true approach for all situations. However, the similarity distribution was clearly affected by the learning overlap. For reasons of comparability, we therefore suggest that users report the similarity measure, the resampling and evaluation method, and the number or iterations they employed when presenting stability results.

It is further important to always assess stability jointly with prediction accuracy since results with weak predictive performance can be very stable and vice versa. Stumps (tree without splits), for example, are very stable, although they have a weak prediction accuracy. As illustrated by our benchmark experiment, however, both aspects can often go hand in hand.

2.5.2 Limitations and future research

A specific aspect of the stability assessment procedure is the pairwise comparison of results generated from different learning samples. This approach was also proposed

by other authors (Lange et al., 2002; Turney, 1995). With this procedure, we aim at the essential question of how much two results were alike if two different learning samples would have been used to generate them and the idea that the learning sample was generated from a DGP that could (theoretically) produce many learning samples. However, there are alternative ways to estimate the similarity distribution P_s. One could, for example, estimate P_s by repeatedly computing the similarity between results generated on resampled data sets and the result generated on the original data set (see, e.g., Bar-hen et al., 2015). Another option could be to estimate P_s from the similarities between all possible pairs of results generated on resampled data sets (see, e.g., Lim & Yu, 2016). Future research should compare the different approaches to reveal what properties they have in common.

Another important open question is, how stability is connected to variability. Suppose that several results have been generated on different learning samples. It can be shown that, for a given point in the predictor space, the sum of the squared distances between the predictions of all results is proportional to the covariance of the predictions at this point (Y. Zhang, Wu, & Cheng, 2012). Thus, the expected variability over the predictor space is proportional to the expected similarity computed by the Euclidean distance. It is, however, not clear whether such a connection can be found for other similarity measures as well. A better knowledge about these connections could possibly help to reduce the computational costs in assessing the (expected) stability.

This points out that in our framework, stability is a global characteristic over the complete predictor space. A result is, however, almost certainly more stable in certain areas of the predictor space than in others. It would be interesting for future research to examine different ways to graphically illustrate the local stability (e.g., using contour plots) and to develop methods for statistical inference (e.g., confidence intervals for predictions, see Wager, Hastie, & Efron, 2014) to investigate the local stability of results.

We have presented a list of similarity measures for classification and regression problems and highlighted some of their properties. However, there is a need for a comprehensive survey of similarity and dissimilarity measures that may also be used in our framework. Future research should as well examine their properties with respect to stability assessment, such as scale-invariance. Another important aspect is to know which similarity and dissimilarity measures fulfill the conditions of a metric and to study the consequences for the stability assessment, if not.

Moreover, it will be interesting for future research to extend the framework to other statistical learning problems. An example of an emerging and promising type of supervised learning applications is model-based recursive partitioning (see, e.g., Strobl et al., 2015; Zeileis et al., 2008) in which a parametric model can be fitted to the observations in each terminal node. The popular survival trees (see Bou-Hamd, Larocque, & Ben-Ameur, 2011, for an overview) are examples of model-based recursive partitioning

approaches for censored data. The resulting tree can also be used for prediction. However, the type of prediction will depend on the model used in the terminal nodes. In the case of a survival tree, it could be the estimated survival curve for each individual. To compute the similarity between two survival trees, one could then, for example, calculate the mean of the log-rank statistic that is often used to compare two estimated survival functions, over all predicted individuals.

2.6 Implementation

An implementation of the stability measuring framework presented in this paper is available in the form of an add-on package for the free open source software R for statistical computing (R Core Team, 2016). The workhorse function `stability()` implements the stability assessment procedure with bootstrap sampling and out-of-bag evaluation as the default resampling and evaluation method and with $B = 500$. Several similarity and dissimilarity measures for regression and classification are implemented. Stability can be assessed for one or more results generated by a few predefined algorithms (see `?LearnerList`), but new algorithms can be integrated by the user. Parallelization can be utilized with a convenience option for multicore computation based on **parallel** (for supported platforms). More detailed stability analyzes based on descriptive measures and graphical illustrations can be conducted for results generated by tree-based algorithms via the function `stabletree()` as described in Philipp, Zeileis, and Strobl (2016).

2.7 Acknowledgments

The authors acknowledge the University of Zurich S3IT: Service and Support for Science IT, for providing the support and the computational resources that have contributed to the research results reported in this publication. URL: `http://www.s3it.uzh.ch`.

Chapter 3

On the estimation of standard errors in cognitive diagnosis models

Michel Philipp, *University of Zurich*
Carolin Strobl, *University of Zurich*
Jimmy de la Torre, *The University of Hong Kong*
Achim Zeileis, *Universität Innsbruck*

Abstract Cognitive diagnosis models (CDMs) are an increasingly popular method to assess mastery or nonmastery of a set of fine-grained abilities in educational or psychological assessments. Several inference techniques are available to quantify the uncertainty of model parameter estimates, to compare different versions of CDMs, or to check model assumptions. However, they require a precise estimation of the standard errors (or the entire covariance matrix) of the model parameter estimates. In this article, it is shown analytically that the currently widely used form of calculation leads to underestimated standard errors because it only includes the item parameters, but omits the parameters for the ability distribution. In a simulation study, we demonstrate that including those parameters in the computation of the covariance matrix consistently improves the quality of the standard errors. The practical importance of this finding is discussed and illustrated using a real data example.

Keywords: cognitive diagnosis model, G-DINA, standard errors, information matrix

3.1 Introduction

Cognitive diagnosis models (CDMs) are restricted latent class models that can be used to analyze response data from educational or psychological tests. In the educational context, they are becoming a popular method for measuring mastery or nonmastery of a set of fine-grained abilities (called attributes) that can be used, for example, to support teachers to recognize strengths and weaknesses of students. Lee, Park, and Taylan (2011) and H. Li (2011) are examples of cognitive diagnostic analyses of mathematics and language skills in large-scale assessments. However, the method has also been suggested to identify the presence or absence of psychological disorders (de la Torre, van der Ark, & Rossi, 2015; Templin & Henson, 2006), or can be used for a detailed measurement of fluid intelligence using abstract reasoning tasks (Rupp et al., 2010; Yang & Embretson, 2007).

The field of cognitive diagnostic assessments has also become a popular area for methodological research over the past 20 years. Many different versions of CDMs have been proposed to analyze responses from tests with various characteristics (e.g., models for dichotomous and polytomous responses, compensatory and noncompensatory processes). See Rupp et al. (2010) for a taxonomy of CDMs. Many of these models can be subsumed within a more general framework, such as the generalized deterministic input, noisy "and" gate (G-DINA; de la Torre, 2011) model, the log-linear CDM (LCDM; Henson, Templin, & Willse, 2009), or the general diagnostic model (GDM; von Davier, 2008). Aside from Bayesian approaches, which are presented in the literature for different versions of CDMs (see e.g., Culpepper, 2015), the model parameters are usually estimated via marginal maximum likelihood estimation (MMLE) using, for example, the EM algorithm (Dempster, Laird, & Rubin, 1977; McLachlan & Krishnan, 2007). In the marginal formulation of the model, a probability distribution that models the attribute space is imposed in conjunction with the traditional item response function, that models the conditional probability of a correct response given the attributes.

An important step of any practical analysis is to assess the uncertainty of the estimated model parameters using confidence intervals or significance tests. Furthermore, several techniques are available to investigate the model fit or to check the model assumptions of a CDM, including tests for (item-level) model comparisons (de la Torre & Lee, 2013) and to detect differential item functioning (Hou et al., 2014). These methods require a precise estimation of the model parameters and their standard errors (or the entire covariance matrix).

However, according to the CDM literature (see e.g., J. Chen & de la Torre, 2013; de la Torre, 2009, 2011; George, 2013; Rojas, 2013; Song, Wang, Dai, & Ding, 2012) and open source software implementations (e.g., in the R package **cdm**, version 4.991-1), it is common to compute the standard errors only for the parameters which are used to specify the item response function while ignoring the parameters used to specify the

joint distribution of the attributes. Consequently, this approach is frequently applied in substantive as well as in many methodological research applications.

Unfortunately, this widely used approach can lead to underestimated standard errors, as we will demonstrate in this paper. The aim of this article is to provide detailed guidance on how standard errors for cognitive diagnosis models should be computed correctly. In addition to analytic arguments, we will investigate the quality of the standard errors using simulations.

The severity of the underestimation varies considerably depending on some known factors (e.g., test length and number of attributes in the assessment), as well as unknown factors (e.g., parameters of the item response function and distribution of the attributes). In some situations, the incremental improvement with the correct approach may become negligibly small (e.g., for high test lengths). However, because the factors potentially causing underestimation are manifold, practioners cannot know upfront whether the data being analyzed is subject to underestimation of standard errors, and how severe the underestimation might be. Given that the necessary computations are straightforward, using the correct approach presented in this article is recommended to be on the "safe side". The additional computations only involve components that are already provided by the results of the estimation routine, and we provide free and open-source software for obtaining the results in practice.

In many situations the underestimation can seriously deteriorate the quality of confidence intervals and statistical tests. Hou et al. (2014), for example, proposed the Wald test to detect differential item functioning in CDMs, and encountered serious Type I error inflation (up to 18%). X. Li and Wang (2015) later found that this was caused by a substantial underestimation of the standard errors with the marginal maximum likelihood estimation (MMLE) approach. Although it is not clear whether the underestimation they observed in their study was caused by the incorrect computation of the standard errors or otherwise, it demonstrates how the performance of the Wald test can be negatively affected by underestimated standard errors (or the entire covariance matrix). Several studies in the field of item response theory (IRT) have also demonstrated the influence of the estimation approach on the quality of procedures that require a covariance matrix. Woods, Cai, and Wang (2012), for example, found better controlled Type I error in the Wald test to detect differential item functioning in the Rasch model if the covariance matrix was computed using the supplemented EM algorithm (Cai, 2008).

Other statistical issues might also cause biases in standard errors for CDMs when using MMLE. Similar to traditional latent class analysis, for example, parameter estimates sometimes converge towards the boundary of the parameter space for small data sets. This causes numerical problems in the calculation of the information matrix, which is inverted to get the covariance matrix. Posterior mode (PM) estimation has been suggested to overcome these problems (DeCarlo, 2011; Garre & Vermunt, 2006). However,

in the CDM literature and in some frequently used software packages, the traditional maximum likelihood (ML) estimation is prevalent. Therefore, we will focus on the estimation of standard errors in this framework for this article.

The rest of the article is organized as follows. The next section contains a short formal introduction of CDMs before the correct estimation of the standard errors is discussed in detail. Later in that section, the G-DINA model will be introduced for the remaining aspects discussed in the article. In the section after next, the quality of the standard errors is investigated using simulation studies and a real data example. The last section concludes with a discussion. To simplify notation and language, we will focus on CDMs for dichotomous responses in the context of educational assessments for the rest of the article. Please note, however, that the calculation of the standard errors described here holds for all types of CDMs estimated via MMLE.

3.2 Cognitive diagnosis models

The primary goal in cognitive diagnosis modeling is to infer mastery or nonmastery of K attributes from the responses of each individual to J items in an assessment. For this task a $J \times K$ Q-matrix (Tatsuoka, 1983) must be specified to identify the cognitive specification of the items, where $Q = \{q_{jk}\}$ and $q_{jk} = 1$ if attribute k $(k = 1, \ldots, K)$ is required to solve item j $(j = 1, \ldots, J)$, and 0 otherwise. The Q-matrix requires domain-specific knowledge, and should ideally be specified together with experts from the field for which the assessment will be needed.

Let $\boldsymbol{X}_i = \{X_{ij}\}$ be the binary response pattern of individual i $(i = 1, \ldots, N)$. The conditional probability of a correct response to item j given the unobserved attribute profile $\boldsymbol{\alpha}_i = \{\alpha_{ik}\}$ is parametrized using a specific item response function, denoted by $P_j(\boldsymbol{\alpha}_i) = \mathsf{P}(X_{ij} = 1|\boldsymbol{\alpha}_i)$. Furthermore, let $\boldsymbol{\delta}_j$ denote the vector of all parameters used to specify $P_j(\boldsymbol{\alpha}_i)$ and, let $\boldsymbol{\delta} = (\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_J)^\top$ denote the vector of parameters that contains all item parameters. For reasons of consistency, it is usually suggested to estimate $\boldsymbol{\delta}$ and $\boldsymbol{\alpha}_i$ using a marginal maximum likelihood approach (de la Torre, 2009; Neyman & Scott, 1948). The marginal probability is given by the sum over all $L = 2^K$ possible attribute patterns, called latent classes:

$$\mathsf{P}(\boldsymbol{X}_i = \boldsymbol{x}_i) = \sum_{l=1}^{L} p(\boldsymbol{\alpha}_l) \cdot \mathsf{P}(\boldsymbol{X}_i = \boldsymbol{x}_i|\boldsymbol{\alpha}_l),$$

where $\mathsf{P}(\boldsymbol{X}_i = \boldsymbol{x}_i|\boldsymbol{\alpha}_l) = \prod_{j=1}^{J} P_j(\boldsymbol{\alpha}_l)^{x_{ij}} \left[1 - P_j(\boldsymbol{\alpha}_l)\right]^{1-x_{ij}}$.

A distribution $p(\boldsymbol{\alpha}_l)$ is imposed to specify a prior probability for each latent class. Let $\boldsymbol{\pi}$ be the vector of all parameters used in the model that specifies $p(\boldsymbol{\alpha}_l)$. For this

article, we choose a saturated model by estimating a probability $\pi_l = p(\boldsymbol{\alpha}_l)$ for each latent class, where $\pi_L = 1 - \sum_{l=1}^{L-1} \pi_l$. Different models can be assumed to reduce the number of parameters (de la Torre & Douglas, 2004; Rupp et al., 2010).

Thus, let $\boldsymbol{\vartheta} = (\boldsymbol{\delta}, \boldsymbol{\pi})^\top$ be the complete vector of all model parameters of a CDM, and further $p = \dim(\boldsymbol{\delta})$ and $q = \dim(\boldsymbol{\pi})$. The marginal log-likelihood that is maximized to estimate $\boldsymbol{\vartheta}$ given the data sample $\boldsymbol{X} = \{\boldsymbol{x}_i\}$ for individuals $i = 1, \ldots, N$, is given by

$$\ell(\boldsymbol{\vartheta}; \boldsymbol{X}) = \log\left[L(\boldsymbol{\vartheta}; \boldsymbol{X})\right] = \log \prod_{i=1}^{N} \sum_{l=1}^{L} \pi_l \cdot \mathsf{P}(\boldsymbol{X}_i = \boldsymbol{x}_i | \boldsymbol{\alpha}_l),$$

and can be maximized using the EM algorithm as described in de la Torre (2009). The estimation procedure provides the posterior probability for each latent class, $\widehat{\mathsf{P}}(\boldsymbol{\alpha}_l | \boldsymbol{x}_i)$, that can be used to find $\hat{\boldsymbol{\pi}}$ and the attribute profiles $\widehat{\boldsymbol{\alpha}}_i$. However, the aim of this article is to discuss the estimation of standard errors for the estimated model parameters $\widehat{\boldsymbol{\vartheta}}$, which will be the focus of the next section.

3.2.1 Theory and estimation of standard errors

The standard errors of the estimated model parameters $\widehat{\boldsymbol{\vartheta}} = \left(\widehat{\boldsymbol{\delta}}, \widehat{\boldsymbol{\pi}}\right)^\top$ can be computed as the square root of the diagonal elements of the covariance matrix of $\widehat{\boldsymbol{\vartheta}}$. Regarding the two types of parameters, $\boldsymbol{\delta}$ and $\boldsymbol{\pi}$, the covariance matrix of $\widehat{\boldsymbol{\vartheta}}$ can be divided into four blocks:

$$\mathrm{Cov}(\widehat{\boldsymbol{\vartheta}}) = V_{\boldsymbol{\vartheta}} = \begin{pmatrix} V_{\boldsymbol{\delta}} & V_{\boldsymbol{\delta},\boldsymbol{\pi}} \\ V_{\boldsymbol{\pi},\boldsymbol{\delta}} & V_{\boldsymbol{\pi}} \end{pmatrix},$$

where $V_{\boldsymbol{\delta}} = \mathrm{Cov}(\widehat{\boldsymbol{\delta}})$ is the covariance matrix of the parameters used to specify the item response function, $V_{\boldsymbol{\pi}} = \mathrm{Cov}(\widehat{\boldsymbol{\pi}})$ is the covariance matrix of the parameters used to specify the distribution of the latent classes and $V_{\boldsymbol{\delta},\boldsymbol{\pi}} = V_{\boldsymbol{\pi},\boldsymbol{\delta}}^\top = \mathrm{Cov}(\widehat{\boldsymbol{\delta}}, \widehat{\boldsymbol{\pi}})$ is the covariance matrix between the two types of parameters.

Complete and incomplete information matrix

The (asymptotic) covariance matrix of $\widehat{\boldsymbol{\vartheta}}$ is equal to the inverse of the information matrix, $V_{\boldsymbol{\vartheta}} = \mathcal{I}_{\boldsymbol{\vartheta}}^{-1}$, which is defined as

$$\mathcal{I}_{\boldsymbol{\vartheta}} = E\left[\psi(\boldsymbol{\vartheta})\psi(\boldsymbol{\vartheta})^\top\right], \tag{3.1}$$

where

$$\psi(\boldsymbol{\vartheta}) = \big(\psi(\boldsymbol{\delta}), \psi(\boldsymbol{\pi})\big)^\top = \left(\frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x})}{\partial \delta_1}, \ldots, \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x})}{\partial \delta_p}, \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x})}{\partial \pi_1}, \ldots, \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x})}{\partial \pi_q}\right)^\top$$

is the score function (i.e., the partial derivatives of the log-likelihood with respect to all model parameters).

Similar to the covariance matrix, the information matrix can be divided into four blocks:

$$\mathcal{I}_{\boldsymbol{\vartheta}} = \begin{pmatrix} \mathcal{I}_{\boldsymbol{\delta}} & \mathcal{I}_{\boldsymbol{\delta},\boldsymbol{\pi}} \\ \mathcal{I}_{\boldsymbol{\pi},\boldsymbol{\delta}} & \mathcal{I}_{\boldsymbol{\pi}} \end{pmatrix} = E\left[\begin{pmatrix} \psi(\boldsymbol{\delta})\psi(\boldsymbol{\delta})^\top & \psi(\boldsymbol{\delta})\psi(\boldsymbol{\pi})^\top \\ \psi(\boldsymbol{\pi})\psi(\boldsymbol{\delta})^\top & \psi(\boldsymbol{\pi})\psi(\boldsymbol{\pi})^\top \end{pmatrix}\right],$$

where $\mathcal{I}_{\boldsymbol{\delta}}$ is the information matrix for the parameters used to specify the item response function, $\mathcal{I}_{\boldsymbol{\pi}}$ is the information matrix for the parameters used to specify the distribution of the latent classes and $\mathcal{I}_{\boldsymbol{\delta},\boldsymbol{\pi}} = \mathcal{I}_{\boldsymbol{\pi},\boldsymbol{\delta}}^\top$ is the information matrix for the two types of parameters.

In many practical applications (e.g., tests for differential item functioning) researchers are primarily interested in the parameters $\boldsymbol{\delta}$, and thus they incorrectly compute the covariance matrix for $\widehat{\boldsymbol{\delta}}$ via the inverse of the *incomplete* information matrix $\mathcal{I}_{\boldsymbol{\delta}}$. This approach, however, considers only a submatrix of the *complete* information matrix including all model parameters $\mathcal{I}_{\boldsymbol{\vartheta}}$. It is important to note that, since $\boldsymbol{\delta}$ and $\boldsymbol{\pi}$ are generally not mutually independent in CDMs (i.e., $\mathcal{I}_{\boldsymbol{\delta},\boldsymbol{\pi}} = \mathcal{I}_{\boldsymbol{\pi},\boldsymbol{\delta}}^\top \neq \mathbf{0}$), inverting the incomplete information matrix $\mathcal{I}_{\boldsymbol{\delta}}$ systematically underestimates the standard errors for $\widehat{\boldsymbol{\delta}}$. In some cases, only the *item-wise* information matrix $\mathcal{I}_{\boldsymbol{\delta}_j}$ (a submatrix of $\mathcal{I}_{\boldsymbol{\delta}}$) is computed and inverted to get the covariance matrix of the parameter vector $\boldsymbol{\delta}_j$. However, similar to traditional IRT models (Yuan, Cheng, & Patton, 2014), $\mathcal{I}_{\boldsymbol{\delta}}$ is not block-diagonal. And thus, inverting the item-wise information matrix underestimates the standard errors even stronger compared to the incomplete information matrix approach.

The above statement can be derived in a formal way using matrix algebra. Let $(\mathcal{I}_{\boldsymbol{\delta}})^{-1}$ be the covariance of $\widehat{\boldsymbol{\delta}}$, based on the incomplete information matrix and let $V_{\boldsymbol{\delta}}$ be the covariance of $\widehat{\boldsymbol{\delta}}$, based on the complete information matrix. From blockwise matrix inversion (see e.g., S. Banerjee & Roy, 2014), it follows, that

$$V_{\boldsymbol{\delta}} \;=\; (\mathcal{I}_{\boldsymbol{\delta}})^{-1} + \Delta, \tag{3.2}$$

[1]The inverse exists in many practical cases. However, it does not exist, e.g., when the parameters lie at the boundary of the parameter space (but estimating standard errors for such parameters is not

with $\Delta = (\mathcal{I}_{\boldsymbol{\delta}})^{-1}\mathcal{I}_{\boldsymbol{\delta},\boldsymbol{\pi}}V_{\boldsymbol{\pi}}\mathcal{I}_{\boldsymbol{\pi},\boldsymbol{\delta}}(\mathcal{I}_{\boldsymbol{\delta}})^{-1}$. If the inverse of $\mathcal{I}_{\boldsymbol{\vartheta}}$ exists[1] and $\mathcal{I}_{\boldsymbol{\delta},\boldsymbol{\pi}} = \mathcal{I}_{\boldsymbol{\pi},\boldsymbol{\delta}}^{\top} \neq \mathbf{0}$, then the diagonal elements of all terms in (3.2) are positive (see Appendix B.1), which implies,

$$\sqrt{\left[V_{\boldsymbol{\delta}}\right]_{r,r}} > \sqrt{\left[(\mathcal{I}_{\boldsymbol{\delta}})^{-1}\right]_{r,r}} \qquad r = 1, \ldots, p.$$

This means that the standard errors of the estimated parameters $\widehat{\boldsymbol{\delta}}$ are consistently underestimated if the incomplete or the item-wise – instead of the complete – information matrix is used. Later, in Section 3.3, we will demonstrate by means of simulations that standard errors computed using the complete information matrix are of better quality. But first, we will discuss two important techniques to estimate the information matrix.

Estimating the information matrix and standard errors

Computing the (expected) information matrix by evaluating the expected value at the maximum likelihood estimate is infeasible for large assessments. The expectation must be taken over the support of the random response vector $\boldsymbol{x}_i$, which becomes very large even if J (the number of items) is only moderately large (e.g., $J = 25$) and computation becomes very slow due to memory limitation.

Thus, the information matrix is often estimated by the empirical counterpart of Equation 3.1, given by

$$\mathcal{J}_{\boldsymbol{\vartheta},OPG} = \frac{1}{N}\left[\sum_{i=1}^{N}\psi(\boldsymbol{\vartheta};\boldsymbol{x}_i)\psi(\boldsymbol{\vartheta};\boldsymbol{x}_i)^{\top}\right]\Bigg|_{\vartheta=\widehat{\vartheta}}, \tag{3.3}$$

also known as the "outer product of gradients" (OPG) estimator, where $\psi(\boldsymbol{\vartheta};\boldsymbol{x}_i)$ is the contribution of individual i to the score function.

Another estimator follows from the fact that under the true parameter values and standard regularity conditions the information matrix (as defined in Equation 3.1) is equivalent to the expected value of the negative Hessian matrix of the log-likelihood. Thus, the information matrix may also be estimated via

$$\mathcal{J}_{\boldsymbol{\vartheta},Hess} = -\frac{1}{N}\left[\sum_{i=1}^{N}\frac{\partial^2\ell(\boldsymbol{\vartheta};\boldsymbol{x}_i)}{\partial\boldsymbol{\vartheta}\partial\boldsymbol{\vartheta}^{\top}}\right]\Bigg|_{\vartheta=\widehat{\vartheta}}. \tag{3.4}$$

meaningful anyway), or when the latent classes are not completely identified by the items in the test.

In practice, however, (3.3) and (3.4) are evaluated at the estimated parameter values and, thus, the two estimators differ by

$$\mathcal{J}_{\boldsymbol{\vartheta},Hess} - \mathcal{J}_{\boldsymbol{\vartheta},OPG} = \frac{1}{N}\left[\sum_{i=1}^{N}\frac{1}{L(\boldsymbol{\vartheta};\boldsymbol{x}_i)}\frac{\partial^2 L(\boldsymbol{\vartheta};\boldsymbol{x}_i)}{\partial\boldsymbol{\vartheta}\partial\boldsymbol{\vartheta}^\top}\right]\Bigg|_{\vartheta=\widehat{\vartheta}}.$$

Often (3.3) is easier to compute, but (3.4) promises a better finite sample approximation of the information matrix (McLachlan & Krishnan, 2007).

From the above definitions, the standard error for the parameter ϑ_r $(r = 1, \ldots, p+q)$, can be computed via the inverse of the complete information matrix, using

$$\widehat{\mathrm{se}}(\widehat{\vartheta}_r) = \sqrt{\left[(\mathcal{J}_{\boldsymbol{\vartheta},OPG})^{-1}\right]_{r,r}} \qquad \text{or} \qquad \widehat{\mathrm{se}}(\widehat{\vartheta}_r) = \sqrt{\left[(\mathcal{J}_{\boldsymbol{\vartheta},Hess})^{-1}\right]_{r,r}},$$

estimated via the outer-products of gradients or the Hessian matrix, respectively. Since the differences between the OPG and the Hessian approach turned out to be negligibly small for simple CDMs (i.e., for the DINA model introduced below, but results are not shown), we will only consider the OPG estimator for the rest of the article.

In Section 3, the improvement of the quality of the standard errors by using the inverse of the complete information matrix will be illustrated using three specific versions of CDMs. Therefore, we will briefly introduce the generalized DINA model framework proposed by de la Torre (2011), which covers other CDMs as special cases. For a comprehensive description of the framework, its relation to other general CDMs and parameter estimation, we refer the reader to the original article.

3.2.2 The G-DINA model

A comprehensive and very flexible version of a CDM is the generalized deterministic input, noisy "and" gate (G-DINA) model (de la Torre, 2011). Due to its general formulation, it includes many other (more restrictive) CDMs as special cases.

For each item in the assessment, the individuals are separated into $2^{K_j^*}$ latent groups, where K_j^* is the number of attributes required by item j (i.e., the sum of the jth row in the Q-matrix). Presence or absence of all the other attributes does not affect the group membership of an individual. Consequently, the attribute vector $\boldsymbol{\alpha}_i$ can be reduced to the attributes required by the particular item.

Let $\boldsymbol{\alpha}_{ij}^* = (\alpha_{i1}, \ldots, \alpha_{iK_j^*})$ denote the reduced attribute vector of individual i for item j. The conditional probability to answer item j correctly is then defined by the item response function

$$P_j(\boldsymbol{\alpha}_{ij}^*) = h^{-1}\left(\delta_{j0} + \sum_{k=1}^{K_j^*}\delta_{jk}\alpha_{ik} + \sum_{k=1}^{K_j^*-1}\sum_{k'=k+1}^{K_j^*}\delta_{jkk'}\alpha_{ik}\alpha_{ik'} + \ldots + \delta_{j12\ldots K_j^*}\prod_{k=1}^{K_j^*}\alpha_{ik}\right),$$

where $h(\cdot)$ is a link function, such as *identity*, *log* or *logit*.

The $\boldsymbol{\delta}_j$ are the model parameters of item j. In case of the identity link, δ_{j0} represents the baseline probability for correctly answering item j when none of the required attribute has been mastered (i.e., a lucky guess); δ_{jk} is the main effect that increases (or in rare cases decreases) the probability for correctly answering item j when attribute k has been mastered; and the rest of the parameters represent interaction terms that can increase or decrease the response probability when two or more of the required attributes have been mastered.

Other CDMs can be obtained by restricting the parameters in the G-DINA model. An intuitive, simple and parsimonious CDM is the deterministic input, noisy "and" gate (DINA; Haertel, 1989; Junker & Sijtsma, 2001) model. In the DINA model the individuals are separated into two latent groups, depending on whether they have mastered all the attributes required to solve the item or not. Thus, the DINA model is a completely noncompensatory (or conjunctive) model, which means that having mastered only part of the required attributes does not increase the probability of answering the item correctly. It can be obtained from the G-DINA model by restricting all parameters except δ_{j0} and $\delta_{j12\ldots K_j^*}$ to zero. Thus, $= g_j$ is called the *guessing* probability, since individuals that have not mastered all attributes required by the item can only guess the correct response. On the other hand, $1 - (\delta_{j0} + \delta_{j12\ldots K_j^*}) = s_j$ is called the *slip* probability, since in this probabilistic model individuals that have mastered all attributes required by the item may still slip and give the wrong response.

Another CDM that can be obtained from the G-DINA model is the *additive* CDM (A-CDM). It is slightly more flexible than the DINA model because the conditional response probability can increase (or in some cases decrease) for each attribute that has been mastered. It can be obtained from the G-DINA model by restricting all interaction parameters to zero.

Score contributions for parameters in the G-DINA model

To estimate the information matrix of the model parameters of the G-DINA model via OPG, the contributions of individual i to the score function, $\psi(\boldsymbol{\vartheta}; \boldsymbol{x}_i)$, are required. They are given by the first-order derivative of the casewise log-likelihood contribution with respect to the model parameters:

$$\begin{aligned}\psi(\boldsymbol{\vartheta}; \boldsymbol{x}_i) &= \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \boldsymbol{\vartheta}} = \frac{\partial \log L(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \boldsymbol{\vartheta}} \\ &= \frac{1}{L(\boldsymbol{\vartheta}; \boldsymbol{x}_i)} \cdot \frac{\partial L(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \boldsymbol{\vartheta}} = \frac{1}{L(\boldsymbol{\vartheta}; \boldsymbol{x}_i)} \cdot \frac{\partial}{\partial \boldsymbol{\vartheta}} \left(\sum_{l=1}^{L} \pi_l \cdot \mathsf{P}(\boldsymbol{x}_i | \boldsymbol{\alpha}_l) \right).\end{aligned}$$

Using formula (A6) from the Appendix in de la Torre (2009) for the partial derivative of the conditional likelihood, the score contributions of the parameters of item j can be computed via

$$\frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \boldsymbol{\delta}_j} = \sum_{l=1}^{L} \mathsf{P}(\boldsymbol{\alpha}_l|\boldsymbol{x}_i) \cdot \left[\frac{x_{ij} - P_j(\boldsymbol{\alpha}_{lj}^*)}{P_j(\boldsymbol{\alpha}_{lj}^*)(1 - P_j(\boldsymbol{\alpha}_{lj}^*))} \right] \cdot \frac{\partial P_j(\boldsymbol{\alpha}_{lj}^*)}{\partial \boldsymbol{\delta}_j}. \tag{3.5}$$

To estimate the score contributions, we plug-in the estimated parameters $\widehat{\boldsymbol{\delta}}_j$ to get $P_j(\boldsymbol{\alpha}_{lj}^*)$ and use $\mathsf{P}(\boldsymbol{\alpha}_l|\boldsymbol{x}_i)$ that is also available from the estimation procedure. The last term in Equation (3.5) depends on the type of CDM that is used. It is also possible to compute the score contributions directly for the conditional response probabilities. In this case, the last term in Equation (3.5) needs to be derived with respect to the conditional response probability of interest.

For the score contributions of the latent class probabilities, the constraint $\pi_L = 1 - \sum_{l=1}^{L-1} \pi_l$ must be taken into account, and thus,

$$\begin{aligned}
\frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \pi_l} &= \frac{1}{L(\boldsymbol{\vartheta}; \boldsymbol{x}_i)} \frac{\partial}{\partial \pi_l} \left(\sum_{l=1}^{L-1} \pi_l \cdot \mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_l) + \pi_L \cdot \mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_L) \right) \\
&= \frac{1}{L(\boldsymbol{\vartheta}; \boldsymbol{x}_i)} \frac{\partial}{\partial \pi_l} \left(\sum_{l=1}^{L-1} \pi_l \cdot \mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_l) + \left(1 - \sum_{l=1}^{L-1} \pi_l \right) \cdot \mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_L) \right) \\
&= \frac{1}{L(\boldsymbol{\vartheta}; \boldsymbol{x}_i)} \frac{\partial}{\partial \pi_l} \left(\sum_{l=1}^{L-1} \pi_l \cdot \Big(\mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_l) - \mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_L) \Big) + \mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_L) \right) \\
&= \frac{1}{L(\boldsymbol{\vartheta}; \boldsymbol{x}_i)} \Big(\mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_l) - \mathsf{P}(\boldsymbol{x}_i|\boldsymbol{\alpha}_L) \Big).
\end{aligned}$$

Since the parameters in the last iteration of the EM algorithm are computed from the posterior values $\mathsf{P}(\boldsymbol{\alpha}_l|\boldsymbol{x}_i)$, it is more precise to also compute the score function for the latent class probabilities using the posterior values, via

$$\frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \pi_l} = \frac{1}{\pi_l} \Big(\mathsf{P}(\boldsymbol{\alpha}_l|\boldsymbol{x}_i) - \mathsf{P}(\boldsymbol{\alpha}_L|\boldsymbol{x}_i) \Big).$$

Nonidentifiability of latent classes

In the theory about standard errors of parameters that is presented above, it is assumed that the inverse of the complete information matrix $\mathcal{I}_{\boldsymbol{\vartheta}}$ exists. This, however, is not always the case in practical applications due to different causes. The most common cause has previously been discussed in Haertel (1989) as the nonidentifiability of latent classes. The problem arises whenever a test does not involve a single-attribute item for

each of the K attributes (see Chiu, Douglas, & Li, 2009, for a theoretical discussion of the completeness of a Q-matrix in the DINA model, and Chiu & Köhn, 2015, for CDMs in general). The G-DINA model can still be estimated, but some of the latent classes are not identified and the estimates of the corresponding latent class probabilities are equivalent. Moreover, when computing the covariance matrix using the complete information matrix, the corresponding columns and rows in the information matrix are alike (i.e., they are linearly dependent). Thus, the information matrix is nonsingular and cannot be inverted.

To avoid problems of identification in practice, it is therefore recommended that, whenever possible a single-attribute item is included for each of the K attributes when developing new tests for cognitive diagnostic assessment. For researchers who perform a cognitive diagnostic analysis of data from an existing assessment (so-called retrofitting), the inversion problem can be circumvented by pooling latent classes that cannot be separated from each other.

3.3 Illustrations

Following the theoretical derivation of the underestimation of the standard errors – resulting from the inversion of the incomplete or the item-wise information matrix – the goal of this section is to illustrate the extent of this underestimation, and its effect on confidence intervals for the parameter estimates. In addition, we show for an exemplary real data set how much the standard errors may be underestimated in practice when the wrong methods are used. For both illustrations, the OPG estimator was used to estimate the covariance matrix of the model parameter estimates.

3.3.1 Coverage study

In the first study, we compare the quality of the standard error estimates based on the complete, the incomplete, and the item-wise information matrix (see Section 3.2.1), by estimating the coverage probability of the true parameter in a Wald-type confidence interval that uses a normal approximation given by $\left[\widehat{\vartheta} \pm z_{\frac{\alpha}{2}} \cdot \widehat{\mathrm{se}}(\widehat{\vartheta})\right]$, and by computing the empirical bias of the standard errors.

Four different sample sizes ($N = 500, 1000, 2000, 5000$) were investigated using the Q-matrix given in Table 3.1. The Q-matrix included five attributes and was constructed such that each attribute was measured equally often (equal row sums in the table) and that the number of items that required the same number of attributes was equally distributed (i.e., five single-attribute items, five two-attribute items, and five three-attribute items). Thus, the Q-matrix represented a test with $J = 15$ items.

Table 3.1: Transposed Q-matrix used in the simulation study.

	Items															
Attributes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	$\sum_k$
α_1	1	0	0	0	0	1	0	0	0	1	1	0	0	1	1	6
α_2	0	1	0	0	0	1	1	0	0	0	1	1	0	1	0	6
α_3	0	0	1	0	0	0	1	1	0	0	1	1	1	0	0	6
α_4	0	0	0	1	0	0	0	1	1	0	0	1	1	0	1	6
α_5	0	0	0	0	1	0	0	0	1	1	0	0	1	1	1	6
$\sum_j$	1	1	1	1	1	2	2	2	2	2	3	3	3	3	3	

The DINA model and the A-CDM were used to generate response data. For each item, the true value of the baseline parameter (δ_{j0}) was set to 0.2. In case of the DINA model, the true value of the interaction parameter between all attributes required by the item ($\delta_{j12\ldots K_j^*}$) was set to 0.6. Therefore, the guessing and the slip probabilities were both equal to 0.2. In case of the A-CDM, the main effect parameters were set to $\delta_{jk} = 0.6/K_j^*$. Thus, with each additionally mastered attribute, the conditional response probability increased by the same amount. The K attributes for each individual were sampled independently from a Bernoulli distribution with probability $\mathsf{P}(\alpha_k = 1) = 0.5$, for all $k = 1 \ldots K$. The joint distribution of the attributes (i.e., the latent class distribution) is then given by a categorical distribution with equal probabilities $\pi_l = \mathsf{P}(\alpha_l) = 1/(2^K)$. Responses that were simulated under the DINA model were analyzed using the DINA and the G-DINA model using the identity link. Note, that the G-DINA is also correct for data that were generated under the DINA model. It was fitted in addition to the DINA model because in practice the true model is usually unknown. However, in this situation the G-DINA model is overspecified, due to the many additional parameters estimated, for which the true values are zero according to the data generating model. Responses that were simulated under the A-CDM were accordingly analyzed using the A-CDM and the G-DINA model using the identity link. Again, the G-DINA is also correct – yet overspecified – for data generated under the A-CDM. To estimate the models and the standard errors, the EM algorithm was implemented in R (R Core Team, 2016) based on the description in de la Torre (2009), but including our new suggestions on how the standard errors should be estimated.

Figures 3.1 and 3.2 illustrate the coverage probabilities for the data generated under the DINA model and the A-CDM, respectively. For all sample sizes and models, the coverage probabilities were computed for the $\boldsymbol{\delta}$ parameters using standard errors based on the complete information matrix $\mathcal{J}_{\boldsymbol{\vartheta}}$ (correct approach), and the incomplete information matrix $\mathcal{J}_{\boldsymbol{\delta}}$ and the item-wise information matrix $\mathcal{J}_{\boldsymbol{\delta}_j}$ (incorrect approaches). It turned out that the asymptotically expected standard errors of the item parameters are identical across items that require the same number of attributes. In the DINA model,

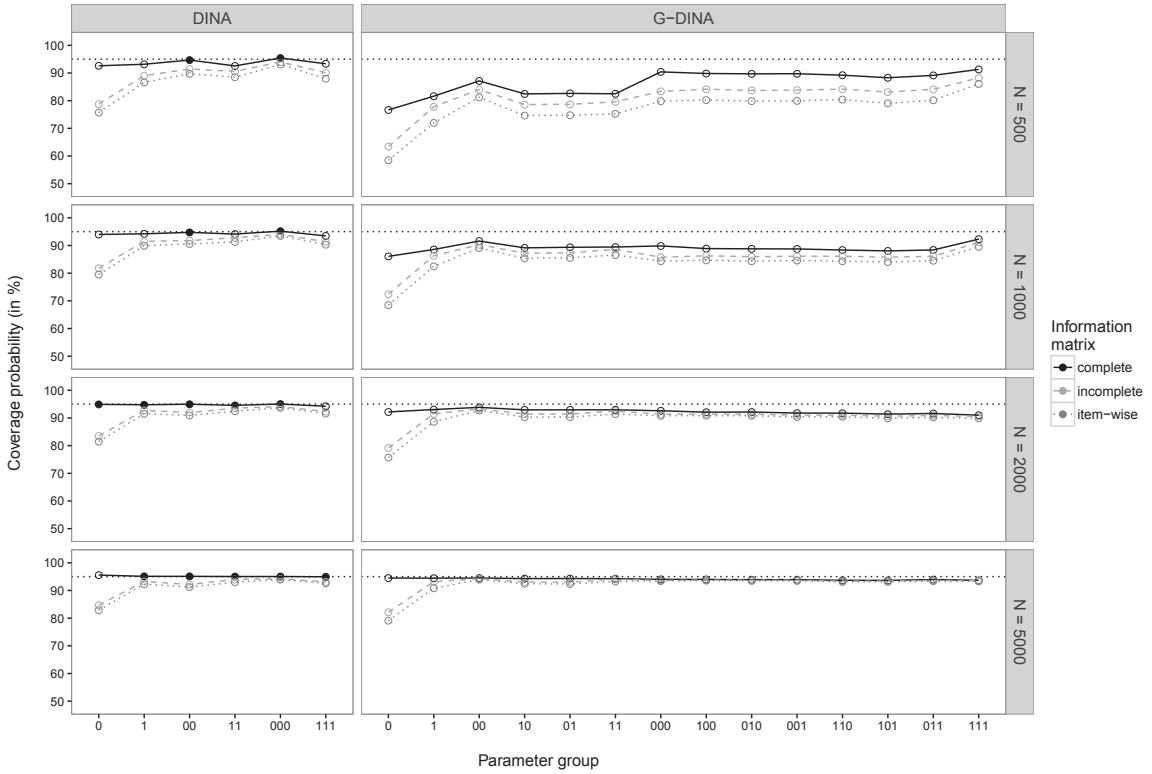


Figure 3.1: Coverage probabilities of 95% Wald-type confidence intervals for data generated under the DINA model are illustrated (on the y-axis) separately for parameters of items that require the same number of attributes (= parameter groups on the x-axis) using three different calculation methods for the standard errors. For ease of readability, values sufficiently close to the nominal coverage probability are depicted as solid circles, all others as empty circles.

for example, the baseline (guessing) probabilities of all single-attribute items share the same asymptotic standard error, no matter which of the attributes is required. This also holds for other item parameters, items that require more attributes and different models. Therefore, the coverage probabilities were averaged over the parameters within those groups, which are illustrated on the x-axis of the graph. The parameter group "0", for example, represents the baseline probability of all single-attribute items. The parameter group "111" represents the parameter of the three-way interaction of all three-attribute items.

By definition, the coverage probability of a 95% confidence interval has an expected nominal coverage rate of 95%. However, due to sampling error, the estimated coverage probabilities may randomly deviate from this nominal value. To achieve a high precision of the estimated coverage probabilities, each configuration was repeated 10,000 times. Assuming an exact binomial distribution for the coverage probabilities, the sampling error was equal to $\sqrt{\frac{0.95 \cdot 0.05}{10{,}000}} \approx 0.002$. Thus, based on a Wald-type confidence interval, we would consider coverage probabilities within $\left[94.6\%, 95.4\%\right]$ as sufficiently close to the nominal rate. Numbers within this interval are depicted with solid circles (otherwise empty circles) in Figures 3.1 and 3.2.

Table 3.2: Coverage probabilities of 95% Wald-type confidence intervals and average estimated bias of the standard errors for data generated under the DINA model and fitted to the DINA model.

		Coverage probabilities			Average estimated bias		
N	η	Complete	Incomplete	Item-wise	Complete	Incomplete	Item-wise
500	0	0.9261	0.7877	0.7571	0.0061	0.0248	0.0275
	1	0.9315	0.8902	0.8662	0.0056	0.0147	0.0191
	00	0.9468	0.9148	0.8974	0.0004	0.0037	0.0050
	11	0.9256	0.9057	0.8849	0.0044	0.0094	0.0136
	000	0.9541	0.9397	0.9312	−0.0006	0.0007	0.0014
	111	0.9334	0.9010	0.8796	0.0036	0.0127	0.0173
5000	0	0.9556	0.8467	0.8289	−0.0005	0.0051	0.0057
	1	0.9511	0.9328	0.9228	−0.0002	0.0016	0.0024
	00	0.9511	0.9213	0.9126	0.0000	0.0009	0.0011
	11	0.9504	0.9399	0.9299	0.0000	0.0008	0.0016
	000	0.9504	0.9423	0.9394	0.0000	0.0002	0.0003
	111	0.9494	0.9313	0.9262	0.0000	0.0019	0.0024

Additionally, Tables 3.2 to 3.5 list the exact values of the coverage probabilities and the empirical bias of the standard errors for the smallest ($N = 500$) and the largest ($N = 5000$) sample sizes (the intemediate sample sizes were omitted for brevity, but can be requested from the corresponding author) and each parameter group (labeled by η). The average empirical bias corresponds to the average of the empirical biases over all replications, that were computed by subtracting the estimated standard errors $\widehat{\text{se}}(\widehat{\vartheta}_r)$ from the empirical standard error of the estimated parameter values over all replications.

Figure 3.1 shows the coverage probabilities for the data generated under the DINA model. When the DINA model was used to analyze the data (see left column in Figure 3.1 and exact values reported in Table 3.2), the coverage probabilities for the standard errors based on the complete information matrix (solid line) were reasonably close to the expected coverage rate for small data samples, and converged quickly toward the nominal rate with increasing sample size N. The coverage rates for the standard errors based on the incomplete (dashed line) or the item-wise (dotted line) information matrix, however, were systematically smaller than the nominal coverage probability, particularly for the first parameter groups. Even for the largest sample size considered, their coverage probability does not converge towards the nominal rate. This is caused by the structural underestimation of the standard errors discussed earlier. We observed the largest underestimation for the baseline probabilities of single-attribute items (parameter group "0"). For the other parameters, the difference to the correct

Table 3.3: Coverage probabilities of 95% Wald-type confidence intervals and average estimated bias of the standard errors for data generated under the DINA model and fitted to the G-DINA model.

		Coverage probabilities			Average estimated bias		
N	η	Complete	Incomplete	Item-wise	Complete	Incomplete	Item-wise
500	0	0.7666	0.6343	0.5850	0.0304	0.0440	0.0482
	1	0.8162	0.7782	0.7198	0.0346	0.0403	0.0483
	00	0.8715	0.8403	0.8121	0.0194	0.0231	0.0269
	10	0.8244	0.7861	0.7465	0.0441	0.0512	0.0583
	01	0.8265	0.7871	0.7477	0.0444	0.0515	0.0587
	11	0.8250	0.7960	0.7527	0.0622	0.0702	0.0824
	000	0.9040	0.8335	0.7982	0.0443	0.0643	0.0712
	100	0.8981	0.8413	0.8022	0.0505	0.0790	0.0903
	010	0.8969	0.8368	0.7984	0.0534	0.0854	0.0973
	001	0.8973	0.8383	0.7993	0.0508	0.0801	0.0917
	110	0.8920	0.8415	0.8041	0.0548	0.0980	0.1159
	101	0.8829	0.8312	0.7906	0.0643	0.1069	0.1252
	011	0.8913	0.8407	0.8014	0.0549	0.0994	0.1175
	111	0.9130	0.8820	0.8603	0.0411	0.0999	0.1270
5000	0	0.9449	0.8213	0.7910	0.0004	0.0063	0.0073
	1	0.9444	0.9319	0.9082	0.0006	0.0018	0.0036
	00	0.9451	0.9423	0.9394	0.0004	0.0005	0.0007
	10	0.9429	0.9304	0.9243	0.0009	0.0024	0.0030
	01	0.9432	0.9301	0.9235	0.0010	0.0025	0.0031
	11	0.9426	0.9402	0.9327	0.0014	0.0019	0.0030
	000	0.9402	0.9361	0.9342	0.0017	0.0019	0.0021
	100	0.9399	0.9377	0.9357	0.0027	0.0029	0.0033
	010	0.9391	0.9369	0.9348	0.0028	0.0030	0.0034
	001	0.9390	0.9370	0.9348	0.0028	0.0030	0.0034
	110	0.9368	0.9337	0.9313	0.0044	0.0051	0.0057
	101	0.9365	0.9340	0.9314	0.0046	0.0053	0.0059
	011	0.9394	0.9363	0.9340	0.0039	0.0047	0.0053
	111	0.9366	0.9355	0.9329	0.0061	0.0066	0.0077

approach is smaller, but still lower than for the correct approach and notably below the nominal rate. A similar pattern can be observed when the G-DINA model was used to analyze the data generated under the DINA model (see right column in Figure 3.1 and exact values reported in Table 3.3). However, for smaller sample sizes the coverage probabilities were generally estimated considerably below the nominal coverage rate of 95%. This artifact may be explained by several circumstances. First, the normal approximation underlying the Wald-type confidence intervals might fail, particularly

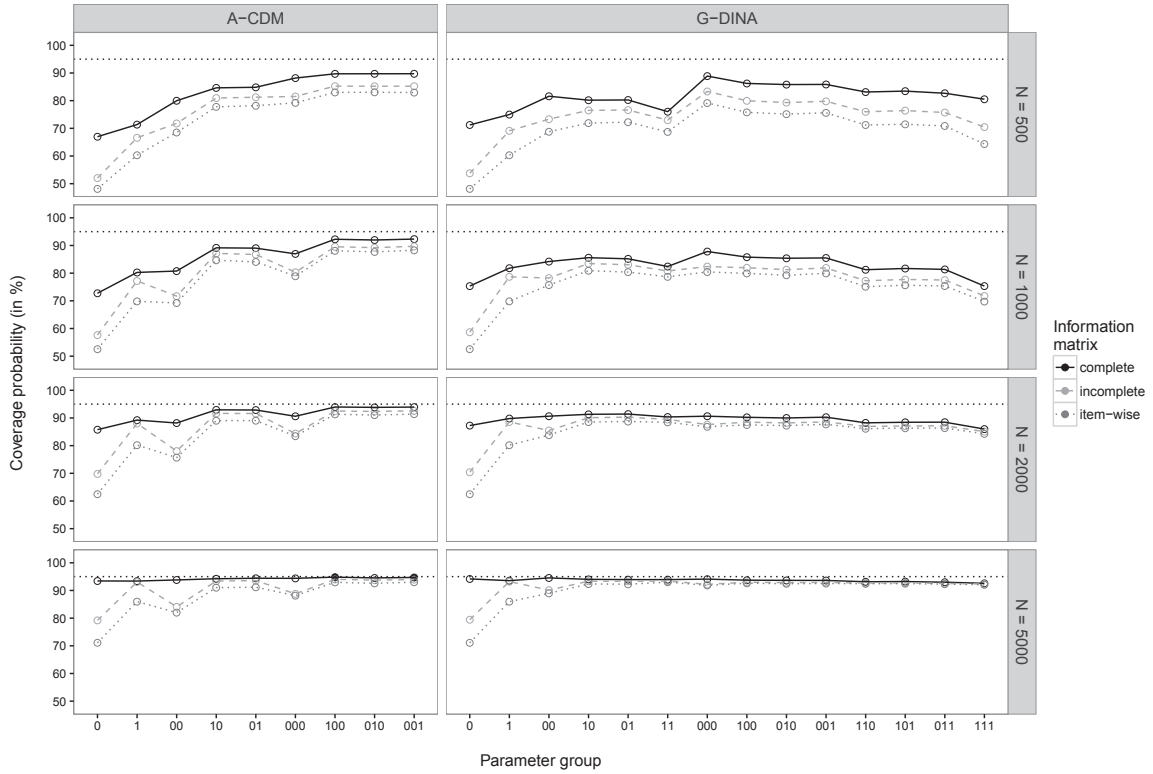


Figure 3.2: Coverage probabilities of 95% Wald-type confidence intervals for data generated under the A-CDM are illustrated (on the y-axis) separately for parameters of items that require the same number of attributes (= parameter groups on the x-axis) using three different calculation methods for the standard errors. For ease of readability, values sufficiently close to the nominal coverage probability are depicted as solid circles, all others as empty circles.

for the baseline probabilities that are restricted between zero and one. Second, for smaller data sets and more complex models, the conditional response probabilities and the parameters used to specify the attribute distribution are often estimated on the boundary of the parameter space. As mentioned earlier, this causes numerical problems in the calculation of the information matrix. Finally, the ratio between the number of estimated parameters per observation is larger for more general models. Thus, inferior asymptotic convergence has to be reckoned with the G-DINA when compared to the DINA model. Nevertheless, the complete information matrix approach clearly provided more accurate results in all conditions considered.

Similar and related conclusions can be drawn from the average empirical biases reported in Tables 3.2 and 3.3. They were (in absolute terms) always smaller when the complete information matrix instead of the incomplete or the item-wise information matrix approaches were used. Please note, that when the correct DINA model was fitted to the simulated data (see values reported in Table 3.2), and when the standard errors were estimated with the complete information matrix approach, the bias almost completely vanished for the larger sample size, whereas with the two incorrect

Table 3.4: Coverage probabilities of 95% Wald-type confidence intervals and average estimated bias of the standard errors for data generated under the A-CDM and fitted to the A-CDM.

		Coverage probabilities			Average estimated bias		
N	η	Complete	Incomplete	Item-wise	Complete	Incomplete	Item-wise
500	0	0.6698	0.5205	0.4813	0.0496	0.0633	0.0678
	1	0.7137	0.6659	0.6030	0.0687	0.0757	0.0843
	00	0.7999	0.7179	0.6852	0.0268	0.0352	0.0382
	10	0.8463	0.8096	0.7778	0.0252	0.0308	0.0350
	01	0.8484	0.8124	0.7817	0.0247	0.0302	0.0343
	000	0.8817	0.8153	0.7923	0.0139	0.0243	0.0270
	100	0.8971	0.8526	0.8299	0.0134	0.0217	0.0250
	010	0.8972	0.8521	0.8304	0.0142	0.0225	0.0258
	001	0.8973	0.8525	0.8294	0.0137	0.0220	0.0252
5000	0	0.9343	0.7922	0.7111	0.0016	0.0099	0.0127
	1	0.9340	0.9307	0.8597	0.0025	0.0029	0.0087
	00	0.9380	0.8411	0.8200	0.0009	0.0061	0.0069
	10	0.9424	0.9341	0.9102	0.0007	0.0014	0.0031
	01	0.9442	0.9357	0.9119	0.0006	0.0013	0.0029
	000	0.9439	0.8878	0.8815	0.0005	0.0041	0.0044
	100	0.9483	0.9403	0.9296	0.0003	0.0010	0.0017
	010	0.9453	0.9373	0.9258	0.0004	0.0011	0.0019
	001	0.9471	0.9394	0.9300	0.0002	0.0008	0.0016

approaches still had significant biases at the larger sample size. This, however, was not the case when the overspecified G-DINA model was fitted to the data simulated under the DINA model (see values reported in Table 3.3). The average estimated biases reported for the complete information matrix approach did not coverage zero, although it was always smaller than for the incomplete and the item-wise approaches. Despite this finding, the complete information matrix approach provided the most accurate standard error estimates of all estimation approaches considered in this study.

Figure 3.2 shows the coverage probabilities for the data generated under the A-CDM (for exact values, see Tables 3.4 and 3.5). For the same reasons as discussed above, the coverage probabilities were estimated below the nominal rate for smaller samples. As the sample size increased, the coverage probabilities computed with the standard errors based on the complete information matrix again approached the nominal rate for the A-CDM and the G-DINA model. The coverage probabilities computed with the standard errors based on the incomplete or the item-wise information matrix, however, were again systematically underestimated. Overall, the complete information matrix approach again provided more accurate results across all conditions considered.

Table 3.5: Coverage probabilities of 95% Wald-type confidence intervals and average estimated bias of the standard errors for data generated under the A-CDM and fitted to the G-DINA model.

		Coverage probabilities			Average estimated bias		
N	η	Complete	Incomplete	Item-wise	Complete	Incomplete	Item-wise
500	0	0.7126	0.5377	0.4815	0.0453	0.0614	0.0678
	1	0.7497	0.6909	0.6029	0.0639	0.0720	0.0843
	00	0.8160	0.7332	0.6877	0.0322	0.0412	0.0464
	10	0.8017	0.7649	0.7190	0.0679	0.0762	0.0853
	01	0.8027	0.7663	0.7225	0.0680	0.0762	0.0854
	11	0.7604	0.7300	0.6874	0.1240	0.1344	0.1476
	000	0.8890	0.8336	0.7918	0.0228	0.0405	0.0479
	100	0.8622	0.7997	0.7579	0.0612	0.0913	0.1048
	010	0.8583	0.7933	0.7516	0.0642	0.0957	0.1095
	001	0.8587	0.7978	0.7562	0.0628	0.0922	0.1056
	110	0.8312	0.7598	0.7120	0.1125	0.1642	0.1867
	101	0.8344	0.7640	0.7145	0.1136	0.1649	0.1873
	011	0.8269	0.7573	0.7087	0.1168	0.1676	0.1899
	111	0.8049	0.7044	0.6430	0.1607	0.2422	0.2772
5000	0	0.9417	0.7945	0.7111	0.0008	0.0098	0.0127
	1	0.9355	0.9322	0.8597	0.0023	0.0027	0.0087
	00	0.9453	0.9013	0.8898	0.0005	0.0040	0.0047
	10	0.9402	0.9335	0.9234	0.0015	0.0024	0.0037
	01	0.9393	0.9332	0.9226	0.0016	0.0025	0.0038
	11	0.9393	0.9340	0.9300	0.0029	0.0041	0.0048
	000	0.9410	0.9219	0.9184	0.0017	0.0039	0.0042
	100	0.9372	0.9299	0.9262	0.0038	0.0051	0.0058
	010	0.9365	0.9283	0.9244	0.0042	0.0056	0.0063
	001	0.9362	0.9290	0.9253	0.0040	0.0053	0.0060
	110	0.9315	0.9267	0.9243	0.0081	0.0095	0.0103
	101	0.9323	0.9275	0.9247	0.0081	0.0094	0.0102
	011	0.9299	0.9254	0.9228	0.0087	0.0101	0.0109
	111	0.9259	0.9234	0.9207	0.0166	0.0178	0.0189

The average empirical biases reported in Tables 3.4 and 3.5 were again always smaller with the complete information approach for all parameter groups and sample sizes. However, as discussed above for the data simulated under the DINA model, the bias did not converge toward zero for the larger sample size, when the overspecified G-DINA model was used to estimate the data simulated under the A-CDM (see values reported in Table 3.5).

Table 3.6: Transposed Q-matrix used for analyzing the elementary probability theory data.

	Items												
Attributes	1	2	3	4	5	6	7	8	9	10	11	12	$\sum_k$
pb	1	0	0	0	1	1	1	1	1	0	1	1	8
cp	0	1	0	0	1	1	0	0	0	1	1	0	5
un	0	0	1	0	0	0	1	1	0	0	0	1	4
id	0	0	0	1	0	0	0	0	1	1	1	1	5
$\sum_j$	1	1	1	1	2	2	2	2	2	2	3	3	

3.3.2 Empirical example

To illustrate the practical importance of estimating standard errors via the complete information matrix, data from a real assessment was analyzed using CDMs. The data stem from a learning experiment at the University of Tuebingen in Germany and is available in the R package **pks** (Heller & Wickelmaier, 2013). The participants were required to answer 12 items about elementary probability theory. For example, "A box contains 30 marbles in the following colors: 8 red, 10 black, 12 yellow. What is the probability that a randomly drawn marble is yellow?". Four different attributes (concepts) were tested: How to calculate

- the classic probability of an event (pb),
- the probability of the complement of an event (cp),
- the probability of the union of two disjoint events (un),
- the probability of two independent events (id).

These concepts were combined to form the 12 items. Therefore, the Q-matrix (see Table 3.6) was defined by the design of the items. The first four items required only one attribute, the items 5 to 10 required two attributes and the items 11 and 12 required three attributes. For this illustration, the responses of 504 participants from the first part of the experiment were analyzed.

The data was fitted using the DINA, the A-CDM and the G-DINA model with the resulting BIC values of 5200.46 ($df = 39$), 5154.58 ($df = 49$) and 5241.70 ($df = 63$), respectively. The results of the A-CDM – which had the lowest BIC value – are illustrated in Table 3.7. The table summarizes the estimated parameters, the corresponding standard errors based on the complete, the incomplete and the item-wise information matrix, and the relative change in the standard errors between the correct and the two incorrect approaches (in parentheses).

Table 3.7: Estimates and standard errors of parameters for the elementary probability theory data. Numbers in brackets correspond to the relative change to the standard errors based on the complete information matrix.

			Standard errors				
Item	Attribute	Estimate	Complete	Incomplete		Item-wise	
1	baseline	0.224	0.065	0.061	(−0.071)	0.052	**(−0.203)**
	pb	0.710	0.067	0.063	(−0.061)	0.055	**(−0.186)**
2	baseline	0.275	0.105	0.080	**(−0.241)**	0.068	**(−0.356)**
	cp	0.699	0.105	0.081	**(−0.232)**	0.069	**(−0.346)**
3	baseline	0.097	0.060	0.055	(−0.082)	0.048	**(−0.194)**
	un	0.864	0.061	0.056	(−0.082)	0.050	**(−0.188)**
4	baseline	0.125	0.038	0.035	(−0.072)	0.032	**(−0.159)**
	id	0.837	0.039	0.037	(−0.064)	0.034	**(−0.140)**
5	baseline	0.201	0.067	0.055	**(−0.187)**	0.048	**(−0.288)**
	pb	0.364	0.116	0.101	**(−0.130)**	0.094	**(−0.191)**
	cp	0.293	0.125	0.111	**(−0.116)**	0.103	**(−0.181)**
6	baseline	0.194	0.062	0.058	(−0.058)	0.051	**(−0.185)**
	pb	0.462	0.085	0.080	(−0.053)	0.074	**(−0.125)**
	cp	0.308	0.083	0.081	(−0.021)	0.077	(−0.071)
7	baseline	0.278	0.071	0.068	(−0.049)	0.062	**(−0.126)**
	pb	0.292	0.095	0.088	(−0.078)	0.083	**(−0.127)**
	un	0.372	0.116	0.105	(−0.094)	0.097	**(−0.164)**
8	baseline	0.430	0.087	0.076	**(−0.132)**	0.063	**(−0.277)**
	pb	0.065	0.095	0.066	**(−0.297)**	0.059	**(−0.371)**
	un	0.462	0.111	0.088	**(−0.212)**	0.079	**(−0.293)**
9	baseline	0.116	0.045	0.043	(−0.042)	0.038	**(−0.145)**
	pb	0.510	0.084	0.079	(−0.060)	0.074	**(−0.113)**
	id	0.154	0.075	0.070	(−0.065)	0.065	**(−0.124)**
10	baseline	0.083	0.050	0.044	**(−0.115)**	0.037	**(−0.248)**
	cp	−0.056	0.060	0.055	(−0.086)	0.048	**(−0.190)**
	id	0.781	0.036	0.035	(−0.027)	0.034	(−0.062)
11	baseline	0.053	0.049	0.045	(−0.086)	0.038	**(−0.229)**
	pb	0.010	0.106	0.086	**(−0.184)**	0.080	**(−0.244)**
	cp	−0.037	0.094	0.084	**(−0.109)**	0.078	**(−0.173)**
	id	0.672	0.034	0.033	(−0.030)	0.032	(−0.060)
12	baseline	0.000	0.039	0.036	(−0.090)	0.029	**(−0.269)**
	pb	0.140	0.469	0.191	**(−0.592)**	0.169	**(−0.640)**
	un	0.000	0.452	0.181	**(−0.600)**	0.162	**(−0.643)**
	id	0.660	0.046	0.042	(−0.067)	0.042	(−0.084)

Note. Strongest relative changes are printed in bold letters for better readability.

For each item, the first parameter estimate represents the baseline probability (i.e., the probability of correctly answering the item when the attributes required by the item have not been mastered). Thus, large values for this guessing probability are unusual. For item 8, however, a value of over 0.4 is reported. A possible explanation is that the item – "What is the probability of obtaining an odd number when throwing a dice?" – was not very difficult, even for individuals without knowledge in basic probability theory. Further parameter estimates represent the amount of increase (or seldom decrease) in probability of answering an item correctly when the corresponding attribute had been mastered. For example, the probability of answering item 1 increased by 0.71 when attribute "pb" had been mastered.

The relative change between the standard errors based on the complete and the incomplete information matrix showed substantial differences (highlighted by bold letters in Table 3.7) for both parameters of the single-attribute item 2, for some of the parameters of the two-attribute items 5, 8 and 10, and for some of the parameters of the three-attribute items 11 and 12. The underestimation of the standard errors based on the item-wise information matrix was even worse. For 30 out of 34 item parameters the standard error was underestimated.

It should be noted that ten out of 48 conditional response probabilities and four out of 16 parameters of the latent class probabilities were estimated at the boundary of the parameter space (not displayed in Table 3.7). As mentioned earlier, this can cause numerical problems in computing the information matrix. According to the previous simulation study, where a similar scenario was investigated (see top-left panel in Figure 3.2 for the same model and a nearly equal sample size), it must be assumed that some of the standard errors reported for this data are generally underestimated. Nevertheless, just like in the simulation study – and as expected from our theoretical considerations – the additional severe underestimation caused by the wrong computation of the information matrix can easily be avoided by using the complete information matrix.

3.4 Discussion

Standard errors are an important measure to quantify the uncertainty of an estimate. They are required for many different statistical techniques to evaluate model fit or to check model assumptions. It is therefore crucial in practical research to estimate standard errors as precisely as possible. In the commonly used approach for computing standard errors in CDMs, however, the information matrix is based only on those parameters which are used to specify the item response function. The parameters used to specify the joint distribution of the attributes (i.e., latent class distribution) are not incorporated in the computation.

In this article, we have shown that with this approach, the standard errors for the parameters of the item response function are systematically underestimated. We therefore strongly recommend to compute the standard errors based on the complete information matrix, which also includes the parameters used to specify the latent class distribution. In addition to the clear theoretical result, we have also illustrated by means of simulations that our approach leads to a higher quality of Wald-type confidence intervals and lower empirical bias. An additional benefit of using the complete instead of the incomplete information matrix is that it also provides the information required to compute standard errors for the parameters used to specify the latent class distribution.

We assume that the incomplete information matrix approaches have only become widely used in the CDM literature because previous authors might have assumed that the off-diagonal elements of the information matrix would have negligible impact under certain conditions. With respect to the item-wise computation of the standard errors, the CDM literature may be partially influenced by the traditional IRT literature, where approaches exist that lead to block diagonal information matrices (e.g., in Thissen & Wainer, 1982), in which case an item-wise computation of the standard errors is possible. However for CDMs, as we showed analytically and illustrated with examples, the complete information matrix approach clearly generates better standard errors than the incomplete and the item-wise approaches and is computationally well feasible. Similar to our results, Yuan et al. (2014) showed that the item-wise computation of the standard errors in IRT models also leads to undersized standard errors.

In the simulation study, we did not specifically vary design factors, such as the Q-matrix, the true values of the item parameters, or the latent class distribution. Varying these factors might positively or negatively affect the severity of underestimation. In a preliminary study with the DINA model, we found that longer tests and highly discriminating items can alleviate the underestimation. It should be highlighted, however, that the proposed method for estimating the standard errors cannot make the quality of the standard errors worse. In practical situations, however, it is difficult (or even impossible) to control the factors that have a large impact on the underestimation. As such it is always preferable to compute standard errors using the complete information matrix.

We note that differences between the approaches are not only expected for the standard errors, but for the entire covariance matrix of the model parameters (although not generally in the same direction). Thus, many techniques used to investigate a fitted model may be affected. The impact of under- or overestimation of the entire covariance matrix will be multiplied for multivariate methods. It is therefore worth in any circumstances to estimate standard errors (and also the entire covariance matrix) from the complete information matrix. As we did not specifically investigate the impact of misestimating the entire covariance matrix on multivariate techniques, it will be interesting for future research to investigate how much the quality of the covariance matrix can be improved by using the complete information matrix in computing it.

The results of the simulation study revealed problems of asymptotic convergence when more complex models were fitted to smaller data sets. This might partially be caused by boundary problems that often occur for smaller data sets. DeCarlo (2011) suggested posterior mode (PM) estimation to overcome these problems. Whether PM estimation leads to more accurate parameter and standard error estimates than the traditional ML approach in CDMs was not the scope of this work, but something that can be investigated in future research. Moreover, the normal approximation of the ML estimates might be more accurate on the real line under the logit link rather than on the (bounded) probability scale under the identity link. However, this not only concerns the estimation of standard errors but of the model parameters in general. Therefore, this is beyond the scope of this manuscript and is not pursued here. It might be of interest for future research, though, to explore the potential benefits of different link functions. In general, the results from our simulation study suggest that it is recommended to use simpler models whenever possible and appropriate because it may avoid boundary problems or problems with asymptotic convergence.

Finally, in the present article, we assumed that the Q-matrix is known or well specified for an assessment. However, in practice (especially when retrofitting CDMs to existing data), the Q-matrix may be unknown or misspecified, which can affect parameter estimation and classification accuracy (de la Torre, 2008; Rupp & Templin, 2007). To minimize the impact of a misspecified Q-matrix, several methods have been proposed. De la Torre (2008) proposed an iterative procedure to evaluate the correctness of the Q-matrix specification in the context of the DINA model. The approach was extended by de la Torre and Chiu (2016) to apply generally to other CDMs. Other recent approaches include that of Y. Chen, Liu, Xu, and Ying (2015), which estimates the Q-matrix of the DINA model using regularization, whereas Chiu (2013) proposed a nonparametric approach to Q-matrix validation that does not require specifying the exact form of the CDM, only that the underlying process is conjunctive in nature. Future research should examine the extent of the impact of Q-matrix mispecifications on standard error estimation, and whether specific steps can be taken to minimize such an impact.

3.5 Computational details

The estimation routines used in this study were written in the free and open-source software R (R Core Team, 2016) for statistical computing. Functions to estimate the parameters and the standard errors in the G-DINA model are provided in the form of the add-on package **Rcdm**, available online at `https://github.com/mphili/cdm` under the terms of the GNU General Public License (Version 2 or 3).

Chapter 4

Detecting DIF in cognitive diagnosis models

Michel Philipp, *University of Zurich*
Carolin Strobl, *University of Zurich*
Achim Zeileis, *Universität Innsbruck*

Abstract In cognitive diagnosis models (CDMs) – similar as in traditional IRT modeling – differential item functioning (DIF) needs to be investigated to ensure test fairness and the validity of the results. The CDM literature has suggested various approaches for different types of CDMs, including the Wald test for detecting DIF in the DINA model between a focal and a reference group. In this chapter it is shown by means of simulations that the frequently applied incorrect computation of the covariance matrix based on an incomplete information matrix causes a Type I error inflation that was observed in previous simulation studies. Moreover, the Lagrange multiplier (LM) test is investigated as an alternative testing approach and the performance is compared to the Wald test. The LM test employed here can be generalized to a more general framework for testing measurement invariance that (if proven successful) allows DIF detection with respect to known and unknown subgroups formed by ordered and unordered categorical as well as continuous variables in different types of CDMs.

Keywords: differential item functioning, DINA model, Wald test, Lagrange multiplier test, score test, information matrix.

4.1 Introduction

Differential item functioning (DIF; Holland & Wainer, 1993) is becoming a relevant research topic in the family of cognitive diagnosis models (CDMs). Over the last decade, several articles and doctoral theses have been published on this subject. Most of this work has focused on the parsimonious deterministic input, noisy "and" gate (DINA; Haertel, 1989; Junker & Sijtsma, 2001) model, which has also been used in several practical applications to analyze response data from large scale assessments, such as the Trends in International Mathematics and Science Study (TIMSS) 2007 (e.g., in Lee et al., 2011).

In a recent publication, X. Li and Wang (2015) have summarized previous research about DIF detection for the DINA model by W. Zhang (2006), F. Li (2008), and Hou et al. (2014) and highlighted their limitations. Among others issues, it is criticized that previous research has only focused on DIF assessments between two subgroups, a reference and a focal group, given by a dichotomous variable (e.g., gender). For example, the use of the Wald test for DIF detection, as proposed by Hou et al. (2014), is limited in this regard, since it can only be used to detect DIF between two subgroups of the individuals. In practice, however, DIF assessments may also be required for more subgroups given by a nominal variable with more than two categories (e.g., ethnic groups) or subgroups formed by a continuous variable (e.g., age).

As a solution, X. Li and Wang (2015) proposed an extended version of the log-linear cognitive diagnosis models (LCDM; Henson et al., 2009) that is modified to incorporate student covariates. This approach, however, is limited to CDMs that fit into the LCDM framework. Here, we therefore study the Lagrange multiplier (LM) test as an alternative approach. It was previously proposed for DIF assessments in traditional IRT models for known subgroups (Glas, 1998) and later as part of a more general family of tests for detecting measurement invariance (Merkle et al., 2014; Merkle & Zeileis, 2013) in psychometric models in general. The framework described by Merkle and Zeileis (2013) can also be used for detecting DIF when the subgroups are unknown. This means that, although in this chapter we only consider the case of detecting DIF for two known subgroups by using the LM test, it is in principle possible to generalize the approach. As such, it would be possible to detect DIF with respect to unordered and ordered categorical variables as well as continuous variables (Merkle et al., 2014; Merkle & Zeileis, 2013) without the need to previously specify a focal and a reference group. Moreover, the tests can in principle be applied to all types of CDMs and all general CDM frameworks.

Further, in the simulation study conducted in Hou et al. (2014) the Wald test exhibited inflated Type I error rates. X. Li and Wang (2015) found that this was due to a substantial underestimation of the standard errors with the marginal maximum likelihood estimation (MMLE) approach. We argue that it is not just the MMLE, but

the incorrect computation of the covariance matrix by omitting the parameters of the latent class distribution (discussed in Chapter 3) when using MMLE that causes the Type I error inflation in the Wald test.

Another limitation of most previous work is that they did not consider ability differences between the individuals in their simulation studies. Similar as in traditional IRT modeling, differences in the ability distribution (that corresponds to the latent class distribution in CDMs, see below) between the subgroups, also referred to as *impact*, can cause Type I error inflation in DIF detection (DeMars, 2010). We believe that a DIF test is only useful for practical application when it is guaranteed that true ability differences are not by mistake taken as DIF. For the Wald test impact is not an issue, since the model (and, thus, also the ability distribution) is estimated separately for the focal and the reference group (Hou et al., 2014). In DIF detection approaches for which the model is estimated jointly for all individuals, however, the influence of impact is currently unclear. Since impact is a realistic scenario in practice, X. Li and Wang (2015) simulated individuals from three subgroups that differed in their ability and the same subgroups were used to simulate the DIF conditions. With the LM test, a DIF detection approach is investigated in this chapter, for which the model is estimated jointly for all individuals. Therefore, we also consider ability differences between the subgroups.

This chapter is concerned with the question whether the LM test can be used to detect DIF in CDMs. We present the results of a preliminary simulation study that compares the performance (i.e., the Type I error and the power) of the Wald test and the LM test. For this preliminary study, however, we restrict our investigations to uniform DIF with respect to a dichotomous variable. To demonstrate our claim that the Type I error inflation in the Wald test is caused by the incorrect computation of the covariance matrix, we will further study the performance of the Wald test under test statistics that are based on three different computations of the covariance matrix (discussed previously in Chapter 3). Finally, to meet our own expectation that the performance of a DIF detection approach should be tested under a realistic scenario with ability differences, we will (similar to X. Li & Wang, 2015) take impact into account in our simulation study.

The chapter is organized as follows. A brief review of the DINA model, an introduction into the concept of DIF in the DINA model and the theory of the tests are given in Section 4.2. The design and the results of the simulation study is presented in Section 4.3. Section 4.4 concludes with a short discussion.

4.2 Theoretical background

4.2.1 The DINA model

An intuitive, simple, and parsimonious version of a CDM is the deterministic input, noisy "and" gate (DINA, Haertel, 1989; Junker & Sijtsma, 2001) model. The DINA model has been intensively studied by methodological researchers (e.g., DeCarlo, 2011; de la Torre & Lee, 2010) and can be seen as a special case of general CDMs, such as the generalized DINA model (de la Torre, 2011) presented in Chapter 3.

For each item $j = 1, \ldots, J$ in the assessment, the individuals $i = 1, \ldots, N$ are separated into two latent groups, depending on whether they have mastered all the attributes required to solve the item or not. This latent group membership is identified from the unobserved K-dimensional attribute profile $\boldsymbol{\alpha}_i$ and the corresponding row from the Q–matrix (see Chapter 3) using an AND-gate operation and can be computed via

$$\eta_{ij} = \prod_{k=1}^{K} \alpha_{ik}^{q_{jk}},$$

where $\eta_{ij} = 1$, if individual i has mastered all attributes required to solve item j, and 0 otherwise. The probability to answer item j correctly is then defined by the item response function

$$P_j(\boldsymbol{\alpha}_i) = \Pr(X_{ij} = 1|\boldsymbol{\alpha}_i) = g_j^{1-\eta_{ij}}(1 - s_j)^{\eta_{ij}},$$

where g_j is called the "guessing" and s_j is called the "slip" parameter. Let

$$\boldsymbol{\delta} = (\boldsymbol{\delta}_j, \ldots, \boldsymbol{\delta}_J)^\top = ((g_1, s_1), \ldots, (g_J, s_J))^\top$$

denote the p-dimensional vector of item parameters in the DINA model, where $p = 2J$.

The parameters in the DINA model are often estimated by maximizing the marginal likelihood as described in de la Torre (2009). With this approach, an additional distribution is imposed for the attribute profiles. Let $\boldsymbol{\pi}$ denote the q-dimensional vector of parameters used to specify this distribution and $\boldsymbol{\vartheta} = (\boldsymbol{\delta}, \boldsymbol{\pi})^\top$ the complete vector of all model parameters.

4.2.2 DIF in the DINA model

We consider DIF with respect to two known subgroups, a focal and a reference group, given by $F = \{i\}_{v_i=1}$ and $R = \{i\}_{v_i=0}$, where v_i is a dummy coded grouping variable

(e.g., gender). Hou et al. (2014) defined DIF as follows: "An item exhibits DIF in the context of CDMs if the probabilities of success on the item are different for individuals that have the same attribute mastery profile but are from different groups" (p. 101). In CDMs, the ability is represented by the attribute profile $\boldsymbol{\alpha}_i$, which is – in case of the DINA model (as described above) – translated into two latent groups for each item. Please note that the latent groups may or may not correspond to the focal and the reference group.

Thus, item j exhibits DIF if $\Pr(X_j = 1|\eta_j)_F \neq \Pr(X_j = 1|\eta_j)_R$. For individuals in the latent group where not all required skills have been mastered ($\eta_j = 0$), this corresponds to

$$\Delta_{g_j} = \Pr(X_j = 1|\eta_j = 0)_F - \Pr(X_j = 1|\eta_j = 0)_R = g_{F_j} - g_{R_j} \neq 0$$

and, for individuals in the latent group where all required skills have been mastered ($\eta_j = 1$), to

$$\Delta_{s_j} = \Pr(X_j = 1|\eta_j = 1)_F - \Pr(X_j = 1|\eta_j = 1)_R = s_{R_j} - s_{F_j} \neq 0.$$

According to Hou et al. (2014), and in line with DIF in the traditional IRT framework, a relevant distinction is between *uniform* and *nonuniform* DIF. For a better understanding, the two concepts are exemplary illustrated in Figure 4.1. Under uniform DIF, the probability of answering an item correctly is higher (top left panel) or lower (bottom left panel) for individuals in the focal group than for individuals in the reference group in both latent groups. Thus, an item exhibits uniform DIF when $\Delta_{g_j} (= g_{F_j} - g_{R_j})$ and $\Delta_{s_j} (= s_{R_j} - s_{F_j})$ have equal signs. Under nonuniform DIF, however, the probability of answering an item correctly is higher (top right panel) for individuals in the focal group than for individuals in the reference group for nonmasters $\eta_j = 0$ and lower for individuals in the focal group than for individuals in the reference group for masters $\eta_j = 1$, or vise versa (bottom right panel). Thus, an item exhibits nonuniform DIF when Δ_{g_j} and Δ_{s_j} have unequal signs. Of course, Δ_{g_j} and Δ_{s_j} may also differ in magnitude which is (for brevity) not considered in this chapter.

The Wald test

To detect DIF with the Wald method as described in Hou et al. (2014), the DINA model needs to be estimated separately for the focal and the reference group, yielding the item parameter estimates $\widehat{\boldsymbol{\delta}}_{R_j} = \left(\widehat{g}_{R_j}, \widehat{s}_{R_j}\right)^\top$ and $\widehat{\boldsymbol{\delta}}_{F_j} = \left(\widehat{g}_{F_j}, \widehat{s}_{F_j}\right)^\top$ for item j with the corresponding covariance matrices, $\widehat{V}_{\boldsymbol{\delta}_{R_j}} = \text{Cov}(\widehat{\boldsymbol{\delta}}_{R_j})$ and $\widehat{V}_{\boldsymbol{\delta}_{F_j}} = \text{Cov}(\widehat{\boldsymbol{\delta}}_{F_j})$.

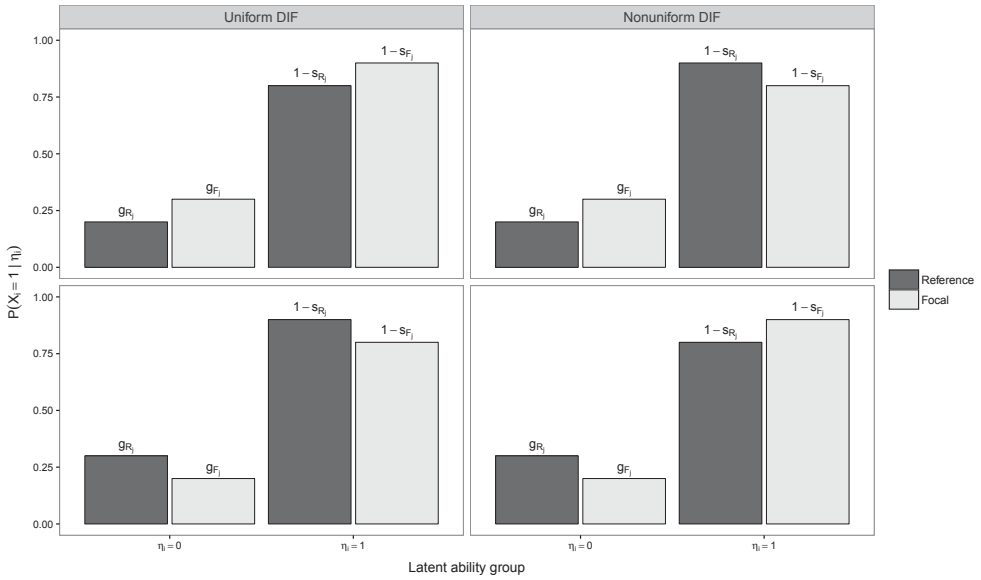


Figure 4.1: Exemplary illustration of uniform and nonuniform DIF in the DINA model.

Hou et al. (2014) suggested to use the Wald statistic to test the null hypothesis

$$H_{0_j} : \begin{pmatrix} \Delta_{g_j} \\ \Delta_{s_j} \end{pmatrix} = \begin{pmatrix} g_{F_j} - g_{R_j} \\ s_{R_j} - s_{F_j} \end{pmatrix} = 0, \tag{4.1}$$

for item j that is given by (see e.g., X. Li & Wang, 2015):

$$W_j = \left(\widehat{\boldsymbol{\delta}}_{R_j} - \widehat{\boldsymbol{\delta}}_{F_j} \right)^\top \left(\widehat{V}_{\boldsymbol{\delta}_{R_j}} + \widehat{V}_{\boldsymbol{\delta}_{F_j}} \right)^{-1} \left(\widehat{\boldsymbol{\delta}}_{R_j} - \widehat{\boldsymbol{\delta}}_{F_j} \right).$$

Under the null hypothesis, W_j is asymptotically χ^2-distributed with 2 degrees of freedom.

As discussed in Chapter 3, the covariance matrix can be computed via the inverse of the information matrix. In the CDM literature and software, however, it is common to compute the information matrix only for the item parameters (which we call the *incomplete* information matrix) or sometimes even separately for each item (which we call the *item-wise* information matrix). We have shown that both approaches can lead to underestimated standard errors, especially for parameters of items that require only few attributes. Thus, the diagonal elements of the covariance matrix are potentially to small, which can lead to an overestimation of the test statistic W_j. In the simulation study presented in Section 4.3 we will therefore present the Type I error and power rates that would result for detecting DIF with the Wald test, when the test statistic is based on the covariance matrix computed via the incorrect *item-wise* or *incomplete* versus the correct *complete* information matrix (for details, see Chapter 3).

The generalized LM test

The generalized LM test was suggested in several articles for detecting DIF in the IRT framework (e.g., Merkle et al., 2014; Merkle & Zeileis, 2013; Strobl et al., 2015). Here we only give a nontechnical description of the general idea of the test. For a detailed account of the inference procedures involved, we refer to Merkle and Zeileis (2013) and Merkle et al. (2014). Note that the test can be extended straightforwardly to more general situations (e.g., more than two subgroups or unknown subgroups formed by continuous variables) than the one considered here.

We focus on the situation where the DINA model was estimated jointly for all individuals via maximum likelihood estimation or, equivalently, where the parameters were estimated by solving

$$\sum_{i=1}^{N} \psi(\boldsymbol{\vartheta}; \boldsymbol{x}_i) = 0,$$

where $\psi(\boldsymbol{\vartheta}; \boldsymbol{x}_i)$ is the score function (i.e., the partial derivatives of the casewise log-likelihood contribution with respect to all model parameters):

$$\psi(\boldsymbol{\vartheta}) = \big(\psi(\boldsymbol{\delta}), \psi(\boldsymbol{\pi})\big)^\top = \left(\frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \delta_1}, \ldots, \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \delta_p}, \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \pi_1}, \ldots, \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \pi_q}\right)^\top.$$

Evaluating the score function for each individual i using the maximum likelihood estimate, reveals the individual deviations from the estimated model over all individuals for each model parameter. The general idea of the LM test is that under the null hypothesis of no parameter instability the individual deviations randomly fluctuate around zero. If, however, an item exhibits DIF with respect to a focal and a reference group, we would expect systematic deviations from zero within each subgroup for the score contributions from the parameters of the particular item.

To statistically test for a systematic difference, the individual score contributions are first rearranged with respect to the ordering of the variable of interest. Thus, in our case of two subgroups, the scores of the individuals in the reference group are placed in front of the scores of the individuals in the focal group. Then, the cumulative sum process of the score contributions, denoted by $\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}})$, is computed (see, e.g., Merkle et al., 2014). The next step is to decorrelate the cumulative sum processes, which can be done by an estimate of the covariance matrix of the scores. At this point, it is again important that all model parameters (including the parameters of the latent class distribution $\boldsymbol{\pi}$) are taken into account, although in DIF detection only the item parameters are of interest. Under the null hypothesis of parameter stability, $\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}})$ can be shown to converge (in distribution) to a k-dimensional Brownian bridge (Hjort &

Koning, 2002), where k is the number of model parameters. This result can be used to test the null hypothesis by constructing scalar test statistics and comparing them to the same statistic applied to the Brownian bridge.

In the categorical case the cumulative scores are additionally aggregated within the categories, yielding $\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}})_R$ and $\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}})_F$ in the situation considered here. A reasonable statistic for testing both item parameters (i.e., the guessing and the slip parameter) for DIF between the focal and the reference group is the "unordered" LM statistic (called LM_{uo} by Merkle & Zeileis, 2013) that is given by the sum of the squared differences between $\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}})_R$ and $\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}})_F$, for the scores of the parameters of item j:

$$LM_j = \sum_{l=2j-1}^{2j} \Big(\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}})_{R,l} - \boldsymbol{B}(\widehat{\boldsymbol{\vartheta}})_{F,l} \Big)^2 .$$

The score function for the parameters of the DINA model can be derived as described in Chapter 3 and are given for the item parameters by

$$\begin{aligned} \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial g_j} &= \sum_{l=1}^{L} \Pr(\boldsymbol{\alpha}_l | \boldsymbol{x}_i) \cdot \left[\frac{x_{ij} - P_j(\boldsymbol{\alpha}_l)}{P_j(\boldsymbol{\alpha}_l)(1 - P_j(\boldsymbol{\alpha}_l))} \right] \cdot \frac{\partial P_j(\boldsymbol{\alpha}_l)}{\partial g_j} \\ \frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial s_j} &= \sum_{l=1}^{L} \Pr(\boldsymbol{\alpha}_l | \boldsymbol{x}_i) \cdot \left[\frac{x_{ij} - P_j(\boldsymbol{\alpha}_l)}{P_j(\boldsymbol{\alpha}_l)(1 - P_j(\boldsymbol{\alpha}_l))} \right] \cdot \frac{\partial P_j(\boldsymbol{\alpha}_l)}{\partial s_j}, \end{aligned}$$

and the partial derivatives of the item response function for the DINA are given by

$$\frac{\partial P_j(\boldsymbol{\alpha}_l)}{\partial g_j} = (1 - \eta_{lj}) \cdot \left(\frac{1 - s_j}{g_j} \right)^{\eta_{lj}} \quad \text{and} \quad \frac{\partial P_j(\boldsymbol{\alpha}_l)}{\partial s_j} = (-\eta_{lj}) \cdot \left(\frac{g_j}{1 - s_j} \right)^{1 - \eta_{lj}}.$$

The scores of the parameters for the latent class distribution are given by

$$\frac{\partial \ell(\boldsymbol{\vartheta}; \boldsymbol{x}_i)}{\partial \pi_l} = \frac{1}{\pi_l} \Big(\Pr(\boldsymbol{\alpha}_l | \boldsymbol{x}_i) - \Pr(\boldsymbol{\alpha}_L | \boldsymbol{x}_i) \Big).$$

To evaluate the scores, we simply plug-in the estimated parameters $\widehat{g}_j$ and $\widehat{s}_j$ and use $\Pr(\boldsymbol{\alpha}_l | \boldsymbol{x}_i)$ that results as a by-product from the estimation procedure with the EM algorithm.

4.3 Simulation study

The goal of the simulation study presented in this section is to demonstrate that the correct estimation of the covariance matrix discussed in Section 3 reduces the Type I

error inflation previously encountered in simulation studies for the Wald test as a DIF detection approach in the DINA model and to compare the performance of the Wald test to that of the LM test.

The simulation study was set up similarly to the previous studies. Response data was simulated under the DINA model for an assessment with $J = 30$ items. The same Q-matrix as in Hou et al. (2014) and X. Li and Wang (2015) was used that included ten single-attribute items ($K_j = 1$), ten two-attribute items ($K_j = 2$), and ten three-attribute items ($K_j = 3$) and was constructed such that each attribute was measured equally often. Three different sample sizes were considered, $N = 1000, 2000,$ and 4000, and the true guessing and slip parameters were set at $g_j = s_j = 0.1, 0.2,$ and 0.3. For this preliminary investigation, only uniform DIF was simulated for two subgroups of size $\frac{N}{2}$ such that the response probability was higher for individuals in the focal group (corresponding to the illustration in the top left panel of Figure 4.1). Thus, $\Delta_{g_j} = \Delta_{s_j}$ was chosen at 0 (no DIF), 0.05 (small DIF), and 0.1 (large DIF). For reasons of comparability, all 30 items were simulated as having DIF, as in Hou et al. (2014). From a practical point of view, it is rather unrealistic that all items is a test are subject to DIF and would require an essential revision of the assessment. However, the Wald test should be able to handle this "extreme" situation, since the model is estimated separately for both subgroups. To make our results comparable to the previous studies we therefore inherited this design. Please see the comment in Section 4.3.2 to what extent this affects the performance of the LM test.

To consider impact (ability differences between the focal and the reference group) the simulation study was replicated twice. In the first run, the attributes were sampled independently from a Bernoulli distribution with probability $\Pr(\alpha_k = 1) = 0.5$ for all individuals. This results in a latent class distribution with equal probability for each attribute pattern. In the second run, the individuals in the focal and the reference group were sampled from different ability distributions by setting the probability that attribute k had been mastered to $\Pr(\alpha_k = 1) = 0.6$ for individuals in the focal group and to $\Pr(\alpha_k = 1) = 0.4$ for individuals in the reference group. This corresponds to a fairly large ability difference and should be considered as a "severe test" for the DIF detection approaches.

The DINA model was estimated using the EM algorithm as described in de la Torre (2009) and each condition was replicated 1000 times. The estimation routine and the Wald test were implemented in the free R system for statistical computing (R Core Team, 2016). The code is available online (`https://github.com/mphili/cdm`) in the form of an R package called **Rcdm** for other users. The R package **strucchange** (Zeileis, Leisch, Hornik, & Kleiber, 2002) was additionally used to perform the LM tests.

4.3.1 Type I error study

We first summarize the results from the 18 conditions without DIF. Table 4.1 lists the rejection rates for the three different Wald tests and the LM test when the null hypothesis (no DIF) was true, tested on a significance level of $\alpha = 0.05$. The rejection rates were additionally aggregated over items that required the same number of attributes. For the significance level of 0.05 and 10×1000 trials, we would expect the rejection rate within the interval $[0.046, 0.054]$ by assuming an exact binomial distribution. Rates within this interval are illustrated in italic letters and rates below the upper margin of this interval are illustrated in bold letters to better distinguish inflated from deflated Type I error rates.

First of all, the simulation replicated the inflation of the Wald test encountered in previous studies for the Wald tests with test statistics based on the incomplete and the item-wise information matrix. The inflation was bigger for larger true parameter values ($g_j = s_j = 0.2$ and 0.3) and did not vanish (although slightly decline) with increasing sample size. The Type I error of the Wald test in which the test statistic was based on the complete information matrix, however, was mostly conservative for the smallest sample size, but converged towards the nominal rate with increasing sample size. This suggests that the poor convergence properties of the model discussed earlier in Chapter 3 causes deviations from the nominal rate for smaller sample sizes.

For the conditions with impact, the Wald test with the test statistic based on the complete information matrix also showed inflated Type I error rates for the smaller samples sizes ($N = 1000$ and $N = 2000$) and large parameter values ($g_j = s_j = 0.3$). This inflation, however, vanished for the largest samples size ($N = 4000$). For the incomplete and the item-wise approach, however, the inflation was much higher. Thus, large heterogeneity in the data can have a particularly negative influence on the performance of the Wald test for DIF detection when the incorrect computation of the covariance matrix is used as our simulation study demonstrated.

The Type I error of the LM test was very well-controlled and almost always located within the expected interval over all conditions considered without impact. When impact was present, however, the Type I error rates were either deflated or inflated. Inflated rates were found for large true parameter values $g_j = s_j = 0.3$ and even increased with larger sample size. This is presumably due to the fact that the model is estimated jointly for all individuals to compute the score contributions for the LM test (see the comment in Section 4.4 on this matter).

4.3.2 Power study

For a better readability, the power rates are illustrated graphically (instead of tabularly) in Figure 4.2 that reads as follows. The rows separate the conditions with varying true

Table 4.1: Type I error rates for detecting uniform DIF using the Wald test (with test statistics based on the complete, the incomplete, or the item-wise information matrix) and the LM test ($\alpha = 0.05$). Rates ≤ 0.054 are printed in bold letters and rates in $[0.046, 0.054]$ are additionally given in italics.

Impact	N	$g_j = s_j$	Wald test (complete)			Wald test (incomplete)			Wald test (item-wise)			LM test		
			$K_j = 1$	$K_j = 2$	$K_j = 3$	$K_j = 1$	$K_j = 2$	$K_j = 3$	$K_j = 1$	$K_j = 2$	$K_j = 3$	$K_j = 1$	$K_j = 2$	$K_j = 3$
no	1000	0.1	**0.030**	**0.028**	**0.022**	**0.039**	**0.035**	**0.029**	0.059	***0.052***	***0.048***	***0.049***	***0.047***	**0.043**
		0.2	**0.034**	**0.031**	**0.030**	0.057	**0.044**	**0.039**	0.087	0.070	0.063	***0.052***	***0.051***	***0.048***
		0.3	0.069	0.057	***0.049***	0.175	0.116	0.084	0.230	0.164	0.128	***0.054***	***0.050***	***0.051***
	2000	0.1	**0.038**	**0.037**	**0.037**	**0.044**	**0.041**	**0.041**	***0.053***	***0.049***	***0.053***	***0.047***	***0.049***	***0.050***
		0.2	**0.043**	**0.042**	**0.041**	0.062	***0.052***	***0.048***	0.080	0.066	0.060	***0.050***	***0.051***	***0.052***
		0.3	0.055	***0.052***	***0.048***	0.155	0.099	0.076	0.186	0.123	0.098	0.054	***0.049***	***0.051***
	4000	0.1	***0.047***	**0.043**	**0.042**	***0.050***	***0.047***	**0.044**	0.058	***0.053***	***0.050***	***0.051***	***0.051***	***0.049***
		0.2	**0.041**	**0.043**	**0.042**	0.059	***0.051***	**0.045**	0.073	0.059	***0.052***	***0.048***	***0.048***	***0.048***
		0.3	***0.048***	***0.048***	**0.046**	0.133	0.096	0.067	0.155	0.111	0.076	***0.046***	***0.049***	***0.048***
yes	1000	0.1	**0.031**	**0.034**	**0.023**	**0.043**	**0.043**	**0.030**	0.064	0.065	***0.048***	**0.032**	**0.026**	**0.021**
		0.2	**0.037**	**0.036**	**0.029**	0.064	0.056	**0.043**	0.095	0.086	0.074	**0.041**	**0.032**	**0.025**
		0.3	0.100	0.077	0.056	0.223	0.153	0.090	0.281	0.217	0.142	0.063	***0.048***	**0.032**
	2000	0.1	**0.036**	**0.038**	***0.048***	**0.042**	**0.043**	***0.054***	0.055	***0.054***	0.066	**0.034**	**0.026**	**0.022**
		0.2	***0.046***	**0.041**	**0.043**	0.068	***0.054***	***0.052***	0.090	0.069	0.068	***0.049***	**0.032**	**0.023**
		0.3	0.067	0.070	0.058	0.184	0.130	0.095	0.220	0.158	0.124	0.081	0.058	**0.034**
	4000	0.1	**0.045**	**0.044**	***0.046***	***0.050***	***0.047***	***0.049***	0.057	***0.054***	***0.054***	**0.039**	**0.028**	**0.022**
		0.2	**0.045**	**0.045**	***0.047***	0.065	0.055	***0.053***	0.081	0.064	0.063	0.070	**0.040**	**0.027**
		0.3	***0.053***	***0.054***	***0.053***	0.162	0.112	0.083	0.188	0.126	0.097	0.131	0.081	**0.037**

Note. g_j = guessing parameter for item j, s_j = slip parameter for item j, N = number of examinees,
K_j = Number of attributes required by item j.

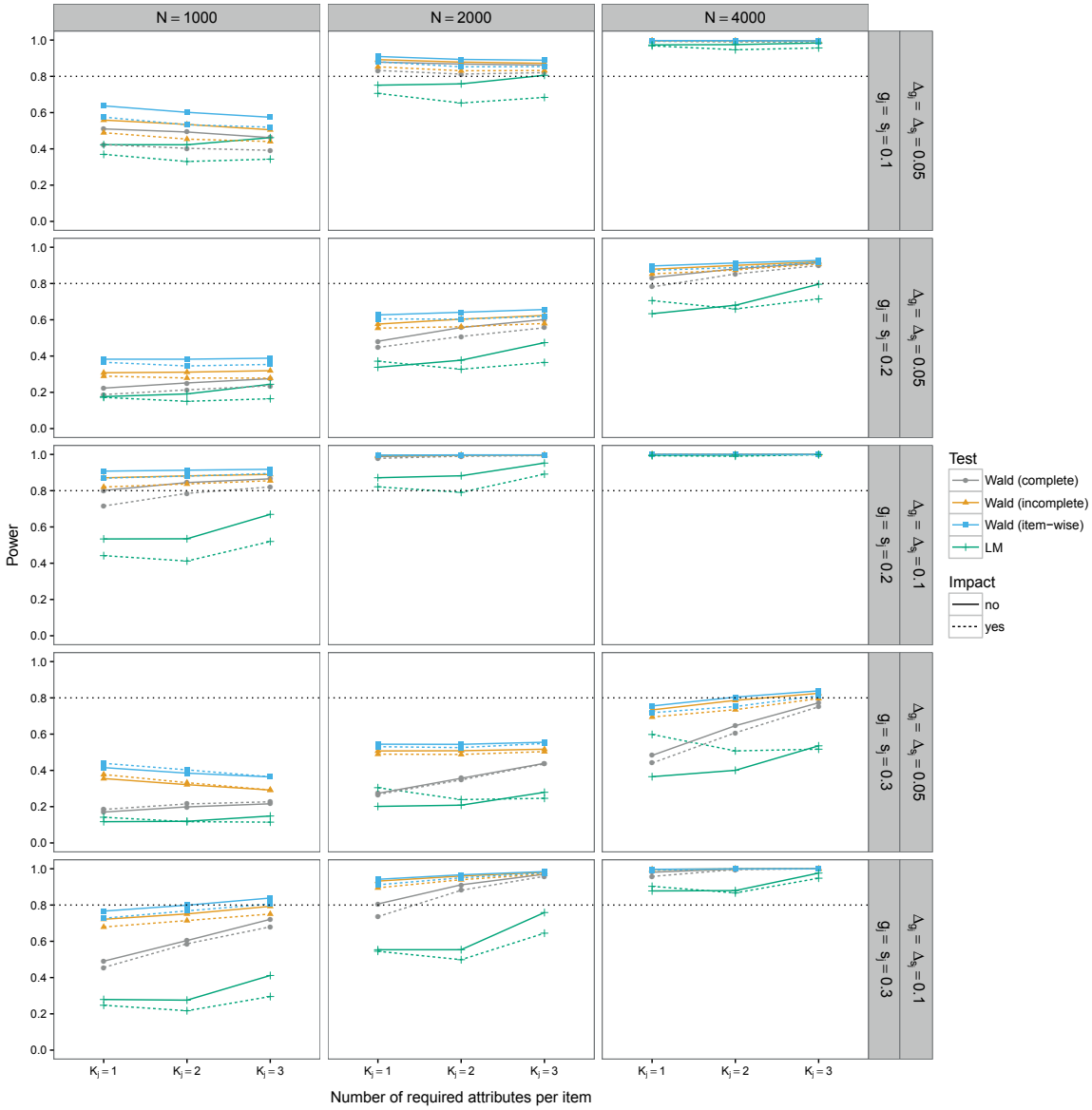


Figure 4.2: Power rates for detecting uniform DIF using the Wald test (where the test statistic is based on the complete, the incomplete, or the item-wise information matrix) and the LM test ($\alpha = 0.05$).

parameter values (first vs. second and third vs. fourth and fifth row) and DIF size (first, second and fourth vs. third and fifth row). The power rates are illustrated in the y-direction of the graph, separately for items that required the same number of attributes (in the x-direction). The impact and no impact conditions are separated using solid and dashed lines, respectively, and the four investigated tests are distinguished using different colors and symbols.

As one would expect, the power rates increased with the sample size (from left to right column) and with larger DIF (compare second vs. third and fourth vs. fifth rows). With increasing true parameter values the power rates decreased (compare,

e.g., first vs. third vs. fifth row), because larger parameter values indicate lower discrimination between the two latent groups. The Wald test showed power rates above 0.8 (usually considered sufficient in practice) in several conditions, especially for the largest sample size, large DIF, or for small true parameter values. Deviations (beyond sampling variability) resulted between the simulations with and without impact. Since the direction of the deviation depended on several factors (number of attributes, DIF amount and true parameter values), it is not further interpreted here, but may be interesting to investigate in future research.

Note that the power rates of the Wald test with the test statistics based on the incomplete or item-wise computation of the information matrix were always higher than when the test statistic was based on the complete computation of the information matrix. The underestimation of the standard errors in CDMs discussed in Chapter 3 and the results of the Type I error study presented in the last section suggest that, when the test statistic is based on an incomplete computation of the information matrix, the test statistic is overestimated and, thus, the power rates are inflated. This means that the Wald test with the test statistic based on the incorrect computation of the covariance matrix is generally too liberal.

The LM test, on the other hand, had the lowest power rates in all condition considered. This is most likely caused by the very strong violation of the model assumption by adding DIF to all items for this simulation study (see the comment in Section 4.4 on this matter). Similar as in the Type I error study, the power rates were inflated for the condition with large true parameter values ($g_j = s_j = 0.3$), when impact was present. For the rest of the conditions, the power rates were slightly lower under impact, but the deviation vanished with increasing sample size.

4.4 Discussion

The simulation study presented in this chapter demonstrated that the previously encountered Type I error inflation in the Wald test for DIF detection in the DINA model (see, e.g. Hou et al., 2014; X. Li & Wang, 2015) was due to an overestimation of the test statistic caused by an incorrect computation of the covariance matrix (based on an incomplete information matrix, see Chapter 3) when the models are estimated via MMLE. When the covariance matrix was correctly computed via the complete information matrix, however, the Wald test showed well-controlled Type I error rates. At the same time, the power rates were lower in some of the investigated conditions than those that had been reported in previous studies, which was due to the same reason and suggests that the test was generally too liberal when the covariance matrix used to compute the test statistic was based on an incomplete information matrix.

In the conditions without impact, the LM test showed well-controlled Type I error rates, but also lower power rates compared to the Wald test. This was presumably caused by the fact that DIF was present in every item, which is in fact a strong violation of the models assumption that the parameters of all items are constant across all individuals. Different than for the Wald test, the items are estimated jointly for all individuals, such that the violations can, for example, affect the ability estimates of individuals in both subgroups. To check our claim, we repeated the simulation study with only 20% DIF items (namely the items 1, 2, 11, 12, 21, and 22). The results (not shown for brevity) supported our assertion in the sense that the power rates of the LM test were comparable and sometimes even higher to the one from the Wald test with the test statistic based on the complete information matrix.

In this additional study, we also checked the Type I error rates for the non-DIF items in the conditions where the DIF items were subject to DIF and noted a small inflation of the Type I error in the LM test for single-attribute items in the conditions with large DIF and true parameter values $g_j, s_j \geq 0.2$. The Type I error of the Wald test with the test statistic based on the correct computation of the covariance matrix, however, was not affected. This indicates a principle problem that can occur through the joint estimation of all individuals applied for the LM test in which the presence of DIF in some items may deteriorate the score contributions for the non-DIF items. To devote the LM test as real alternative to the Wald test as an item-wise DIF detection approach, this issue needs to be further investigated in future research. The LM test, however, could also be employed as a global DIF detection approach by testing all items jointly for DIF. Therefore, the test statistic is simply computed over the scores of all item parameters. In this case, the situation described above is no longer problematic.

Similar to X. Li and Wang (2015), this study investigated impact (ability differences) with respect to two subgroups. For the Wald test with the test statistic based on the correct computation of the covariance matrix, some conditions had slightly increased Type I error rates that vanished with increasing samples size. The impact, on the other hand, did affect the Type I error rates of the LM test. They were either deflated for small true parameter values or inflated for large true parameter values. These effects did not vanish with increasing sample size.

In CDMs, the ability distribution is freely estimated from the data. Moreover, it is easy to show by rearranging terms that separate ability distributions for subgroups are combined in a mixed ability distribution that holds for all individuals. Therefore, we would not expect problems with the LM test under impact. However, the results of the simulation study contradict our expectations and indicate that violations of measurement invariance in the parameters of the latent class distribution can deteriorate the score contributions for the item parameters. Previously, de la Torre and Lee (2010) encountered bias in the item parameter estimates for individuals formed by subgroups with different ability distributions, when the model was estimated jointly for all in-

dividuals. Thus, it is possible that the origin of this problem is in the estimation procedure, rather than in the LM test.

There is a need for future research to investigate these issues. If thereby the LM test proves successful and existing problems can be solved, however, it has a great potential to be used as a general and flexible DIF detection approach for many different types of CDMs and general CDM families.

Conclusion and outlook

The aim of this work was to highlight the importance of stability to draw reliable conclusions from the results of statistical data analysis. Graphical, computational, and statistical methods were provided and/or evaluated, that can be used to investigate the stability of results generated by parametric and nonparametric methods applied in practice and research. This chapter summarizes important findings and limitations of the contributions and suggests ideas for future research.

Approaches to assess the stability of results generated by nonparametric, adaptive, and flexible methods were addressed in Chapters 1 and 2. The first contribution presented in Chapter 1 is a toolkit of descriptive measures and graphical illustrations to investigate the stability of results generated by tree-based methods. It was found that the stability of trees should be evaluated by analyzing the variable and cutpoint selection. The stability assessment was illustrated by applying the tools to a real data set. The second contribution presented in Chapter 2 is a framework to measure the stability of results from supervised statistical learning in general. In the framework, the stability is assessed by pairwise comparison of results generated on resampled learning samples. It was illustrated by means of simulations that the stability of results depends on the algorithm, the DGP, and – most importantly – how well the algorithm can approximate the true functional form between the predictors and the conditional distribution of the response in the DGP. The stability assessment also depends on framework specific factors, such as the selected resampling method.

The approaches presented in Chapters 1 and 2 are based on the key theoretical argument that the equivalence of the substantive meaning of two results can be assessed by the comparison of their predictions; as is was previously suggested by Turney (1995). The substantive meaning is what researchers interpret to draw scientific conclusions. From the decision tree generated for the Titanic data set in Chapter 1, for example, one can conclude that women who traveled first class (a subgroup formed by the predictors) had a high chance to survive the sinking of the RMS Titanic (the predicted response for the subgroup). Consequently, when the goal is to assess the stability with respect to the interpretation of a result it is possible to study the stability of its prediction.

Moreover, it was found that the interpretation of two results from statistical learning can be identical although their structure differs (e.g., the order of the splits in a tree).

This rational is very relevant for practitioners who use their results for interpretation and means that – by inversion of the argument – just because two results (e.g., trees) look very different at first sight, this does not automatically imply that they also lead to fundamentally different interpretations. Although deviations in their interpretations are very likely (e.g., due to randomness in the data), they still might be relatively similar.

This further implies that, besides the prediction itself, only the structural elements that are required to generate the prediction should be used to assess stability. These are, for example, the selected variables and cutpoins in a tree or the estimated coefficients in a parametric model. Thus, for various algorithms, more detailed measures can be developed that may provide an in-depth assessment of the stability of their results, such as the tools presented for tree-based methods in Chapter 1. However, only the framework presented in Chapter 2 can be used for a fair comparison of the stability between results generated by different algorithms.

Taking these findings into account, future research should put effort in developing sophisticated descriptive measures and graphical tools for assessing the stability of results from tree-based methods or other statistical learning algorithms. Similar to Wager et al. (2014) it may, for example, be possible to develop a theory based on the resampling method that allows the computation of a lower bound for acceptable variable selection proportions or confidence intervals for selected cutpoints in tree-based methods.

Effort should also be aimed at investigating the stability of results generated by algorithms that are used in practice and research for statistical data analysis; especially when the aim is to draw conclusions about the underlying data-generating process. A starting point could be a comprehensive study in which different algorithms applied to well-known benchmarking problems for regression or classification, such as those from the UC Irvine Machine Learning repository Lichman (2013) are compared. When applying methods for statistical learning in practice, people should choose the algorithm (and its hyperparameters) not only with respect to prediction accuracy, but also with respect to the stability of the generated results when conducting explanatory data analysis.

Recursive partitioning was used throughout Chapters 1 and 2 to motivate stability assessments, because this method is widely used for explanatory modeling and because it is well-known that the methods can generate unstable results. Moreover, Breiman (1996b) identified other methods as unstable, including multivariate additive regression splines (MARS) and neural networks.

Although the majority of applications in business, industry, and science are of predictive nature, there is often an ex post interest in understanding the relation between the observed variables. Therefore, more and more methods are available to peek into those

black boxes generated by powerful, yet purely predictive approaches, such as random forests or deep learning algorithms. Thus, whenever the focus shifts from prediction towards generating an understanding about the relation between the observed variables, stability is again important. Irrespective whether the used algorithm is known to be stable or not, the framework is generally useful to investigate the stability of a result, since the stability always also depends on the characteristics of the DGP and on the match between the algorithm and the DGP.

Stability assessments can also guide the choice of users between ensemble methods and single results. If in an explanatory data analysis, where explanation is typically more important than prediction, the stability of a single result is sufficiently high, there is no need for generating a less interpretable result through bagging or boosting. Thus, the methods presented in this thesis can, for example, help users to make a stability-based choice between a single tree and a random forest.

To address some problems or limitations with the methods and studies presented in Chapters 1 and 2, it should be mentioned that, as opposed to **glm**, it was not possible to identify a resampling and evaluation method for **ctree** that recovered the reference stability well for all investigated DGPs. This highlights that the quality of the stability assessment depends on the algorithm and perhaps additionally on its tuning parameters. Thus, comparisons of stability between different algorithms may be conflated and therefore this issue should be investigated in future research.

In practical applications, the stability in a specific region could be more relevant than in others. Such as in medical applications where a specific group of patients (e.g., pregnant women or children) might be more vulnerable to wrong treatments than other groups. If a user wants to limit the stability assessment to a particular region of the predictor space, reweighting the observations in the evaluation sample is the right approach. However, we only discussed nonnegative case-weights in Chapter 2, although proportionality weights could be more suitable for practical applications (similar to misclassification costs in predictive modeling). But whether proportionality weights can be used or not depends on the similarity measure and future research should examine how this could be implemented in a general way for any similarity measure.

The framework presented in Chapter 2 can be used to investigate the *global* stability of a result over the complete predictor space in terms of a single similarity distribution. However, results from statistical learning are – almost certainly – more stable in some regions of the predictor space than in others. A possible way to address this limitation is by investigating the *local* stability of a result. This could be done by specifying a grid of points over the complete predictor space and then illustrate the stability measure at these points. For two predictor variables, the resulting similarity values could be illustrated by contour plots. With this approach one would generate a local stability map over the predictor space as opposed to a single similarity distribution that represents the complete preditor space, as it is done in our framework.

Future research should examine ways to compute the local stability efficiently and how it could be meaningfully reported and illustrated when the number of predictors is larger than two. The idea of individual conditional expectation presented in Goldstein et al. (2015) could be used as a starting point.

Approaches to assess the stability of results generated with parametric models were addressed exemplary for cognitive diagnosis models (CDMs) in Chapters 3 and 4. These models are becoming popular in the field of educational and psychological measurement to assess the mastery or non-mastery of a set of fine-grained attributes (e.g., particular mathematical skills).

The third contribution presented in Chapter 3 is a discussion on the estimation of the standard errors in CDMs. Standard errors are important to assess the reliability of parameter estimates in a parametric model. Thus, they should be accurate and correctly computed. A common way to compute the standard errors in models estimated by the maximum likelihood (ML) approach, is based on the inversion of the information matrix. In Chapter 3 it is found by theoretical considerations and illustrated by means of simulations that an incomplete computation of the information matrix can lead to an underestimation of the standard errors for the specific parameters of interest, when the model parameters are not independent. Although, this finding is valid for parametric models in general, it is of specific relevance in CDMs for reasons discussed in Chapter 3.

The fourth contribution presented in Chapter 4 is a discussion of two statistical tests for parameter instability in CDMs. An important assumption to guarantee fair comparisons between subgroups is that the item parameters are invariant. In the field of educational and psychological measurement, it is therefore important to investigate the instability of the item parameters, called differential item functioning (DIF), with respect to subgroups formed by the covariates of the respondents. Among other approaches, the Wald test has been proposed to detect DIF in CDMs that, however, suffered from Type I error inflation for items with low discrimination as shown in previous simulation studies. In Chapter 4 it was demonstrated by means of simulations that the Type I error inflation is caused by the incorrect computation of the covariance matrix discussed in Chapter 3. With the correct computation of the covariance matrix, the Wald test had well-controlled Type I error and acceptable power rates. Note that in Chapter 4, only uniform DIF was investigated.

Further, the LM test was investigated that could in principle be extended to detect DIF between known or unknown subgroups formed by categorical and numerical variables. Although the LM test performed equally well as the Wald test in most investigated conditions, it suffered from Type I error inflation under impact (ability differences between the subgroups tested for DIF). Future research should examine the LM test more closely in this situation and come up with solutions to deal with the impact

Table 4.2: Decision tables for classical DIF testing and equivalence testing.

	Classical DIF testing $H_0 : g_{j_R} = g_{j_F}$, $H_A : g_{j_R} \neq g_{j_F}$		Equivalence testing $H_0 : g_{j_R} \neq g_{j_F}$, $H_A : g_{j_R} = g_{j_F}$	
Test	H_0 true (no DIF)	H_0 false (DIF)	H_0 true (DIF)	H_0 false (no DIF)
rejects H_0	Incorrect decision: Replace item that has no DIF	Correct decision: Replace item that has DIF	Incorrect decision: Keep item that has DIF	Correct decision: Keep item that has no DIF
	Type I error $= \alpha$	Power $= 1 - \beta$	Type I error $= \alpha$	Power $= 1 - \beta$
fails to reject H_0	Correct decision: Keep item that has no DIF	Incorrect decision: Keep item that has DIF	Correct decision: Replace item that has DIF	Incorrect decision: Replace item that has no DIF
	Specifity $= 1 - \alpha$	Type II error $= \beta$	Specifity $= 1 - \alpha$	Type II error $= \beta$

problem, since the approach could provide a powerful alternative to the Wald test for DIF detection in various testing situations.

An issue that arises with the tests for DIF detection discussed in this Chapter 4, is that the burden of proof rests on the presence of DIF and not on the actual research hypothesis that the studied item does not have DIF. This issue was previously mentioned in the DIF literature (Kopf, 2013) and is exemplified by means of decision tables in Table 4.2.

By defining the equivalence of the item parameters as the null hypothesis, as it is done in classical DIF testing (see left part of Table 4.2), the actual "worst case" scenario of keeping an item that has DIF is no longer controlled by the α-level of the test. Instead, it is given by the Type II error (β) of incorrectly not rejecting the null hypothesis although it is false that depends on the power of the test ($1 - \beta$). Thus, when the power of the test is low (e.g., due to a small sample size, large variance in the response probabilities, or small DIF) the Type II error is large and the "worst case" scenario is more likely to occur. This can be a serious problem for the tests discussed in Chapter 4, since the power was not always high enough to rule out this issue.

The so-called equivalence tests (Walker & Nowacki, 2011) could provide a solution to this problem (see right part of Table 4.2). The nonequivalence of the item parameters (i.e., the item has DIF) is defined as the null hypothesis by specifying a minimum discrepancy in the parameter values between the focal and the reference group. As such, the actual research hypothesis (no DIF) becomes the alternative, as it is common in null hypothesis testing, and the "worst case" scenario of keeping an item although it has DIF is controlled by the α-level of the test. It is left for future research to investigate this idea for the tests used for DIF detection in CDMs.

Which approach to use, however, is also a matter of perspective. The classical tests for nonequivalence (see left part of Table 4.2) may be appropriate for testing companies due to the costs of generating test items. To avoid errorneously excluding an expensive item, it is of advantage when the incorrect decision of replacing an item that has no DIF is stricly controlled by the α-level.

Finally, a general limitation of CDMs is the poor asymptotic convergence of the parameter estimates to the normal distribution for complex models, even for relatively large data sets. This is caused by the large number of parameters estimated in a saturated model, such as the G-DINA model. Reduced models (e.g., DINA, DINO, or A-CDM) that require fewer parameters, can be estimated separately for each item in the G-DINA model framework. To identify the appropriate reduced model for each item, de la Torre and Lee (2013) proposed the Wald test that makes a comparison of the item level fit between the reduced and the saturated model. The number of comparisons depends on the number of items and the number of reduced models of interest. A drawback of this approach is that the quality of the Wald test depends on the asymptotic convergence of the model parameters and its covariance matrix.

Another approach could be to find the best rule for each item via step-wise model selection. For items that require two or more items one could sequentially select the rule that leads to the model fit with the lowest AIC or BIC value, while all other items keep the rule from the initial model, for example, the saturated G-DINA model. This approach is already implemented in the **Rcdm** package for research purposes. It should, however, not be used for practical applications before its quality is investigated in future research – but first results looked promising.

Yet a different approach to obtain reduced models – that could be investigated in future research – is to use regularization methods to shrink dispensable parameters (or groups of parameters) to zero while estimating the saturated G-DINA model. Compared to the approach based on the Wald test, there would not be need for additional hypothesis testing after the saturated model was estimated.

List of Figures

List of Tables

Appendix A

Supplementary material: Chapter 2

A.1 Similarity measures

- Table A.1 lists similarity measures for the regression case.
- Table A.2 lists similarity and distance measures for the classification case.

A.2 Simulation experiments

- Figure A.1 and Figure A.2 illustrate the complete results from Study 1 presented and discussed in Section 2.4.2.
- Figures A.3-A.8 illustrate the complete results from Study 2 presented and discussed in Section 2.4.3.

Table A.1: Similarity measures for the regression case.

Name	Formula or definition	Range	Scale-invariance	R package
Euclidean distance	$d_{\mathrm{ED}}(\hat{y}', \hat{y}'') = \sqrt{\sum_{i=1}^{m}(\hat{y}_i' - \hat{y}_i'')^2}$	$[0, \infty]$	no	`edist()`
Mean squared distance	$d_{\mathrm{MSD}}(\hat{y}', \hat{y}'') = \frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i' - \hat{y}_i'')^2$	$[0, \infty]$	no	`msdist()`
Root mean squared distance	$d_{\mathrm{RMSD}}(\hat{y}', \hat{y}'') = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i' - \hat{y}_i'')^2}$	$[0, \infty]$	no	`rmsdist()`
Mean absolute distance	$d_{\mathrm{MSD}}(\hat{y}', \hat{y}'') = \frac{1}{m}\sum_{i=1}^{m}\lvert\hat{y}_i' - \hat{y}_i''\rvert$	$[0, \infty]$	no	`madist()`
Quantile of absolute distance	$d_{\mathrm{QAD}}(\hat{y}', \hat{y}''; p) = \mathsf{P}(\lvert\hat{y}' - \hat{y}''\rvert \leq \kappa) = 1 - p$	$[0, \infty]$	no	`qadist()`
Coverage probability	$s_{\mathrm{CP}}(\hat{y}', \hat{y}''; \kappa) = \frac{1}{m}\sum_{i=1}^{m}\mathbb{1}_{\lvert\hat{y}_i' - \hat{y}_i''\rvert \leq \kappa}$	$[0, 1]$	no	`cprob()`
Gaussian radial basis function kernel	$s_{\mathrm{GRBF}}(\hat{y}', \hat{y}''; \sigma) = \exp\left(-\frac{d_{\mathrm{ED}}(\hat{y}', \hat{y}'')^2}{2\sigma^2}\right)$	$[0, 1]$	no	`rbfkernel()`
Tanimoto coefficient	$s_{\mathrm{TC}}(\hat{y}', \hat{y}'') = \frac{\sum_{i=1}^{m}\hat{y}_i'\hat{y}_i''}{\sum_{i=1}^{m}(\hat{y}_i')^2 + \sum_{i=1}^{m}(\hat{y}_i'')^2 - \sum_{i=1}^{m}\hat{y}_i'\hat{y}_i''}$	$[-\frac{1}{3}, 1]$	no	`tanimoto()`
Cosine similarity	$s_{\mathrm{CS}}(\hat{y}', \hat{y}'') = \frac{\sum_{i=1}^{m}\hat{y}_i'\hat{y}_i''}{\sqrt{\sum_{i=1}^{m}(\hat{y}_i')^2}\sqrt{\sum_{i=1}^{m}(\hat{y}_i'')^2}}$	$[-1, 1]$	no	`cosine()`
Concordance correlation coefficient	$s_{\mathrm{CCC}}(\hat{y}', \hat{y}'') = \frac{2\sigma_{\hat{y}'\hat{y}''}}{\sigma_{\hat{y}'}^2 + \sigma_{\hat{y}''}^2 + (\mu_{\hat{y}'} - \mu_{\hat{y}''})^2}$	$[-1, 1]$	yes	`ccc()`
Pearson correlation coefficient	$s_{\mathrm{PCC}}(\hat{y}', \hat{y}'') = \frac{\sum_{i=1}^{m}(\hat{y}_i' - \mu_{\hat{y}'})(\hat{y}_i'' - \mu_{\hat{y}''})}{\sqrt{\sum_{i=1}^{m}(\hat{y}_i' - \mu_{\hat{y}'})^2}\sqrt{\sum_{i=1}^{m}(\hat{y}_i'' - \mu_{\hat{y}''})^2}}$	$[-1, 1]$	yes	`pcc()`

Note. m is the size of the evaluation sample and $\mathbb{1}$ denotes the indicator function.

Table A.2: Similarity and distance measures for the classification case.

Name	Formula or definition	Range	R package
Average class agreement	$s_{\text{ACA}}(\hat{y}', \hat{y}'') = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\hat{y}_i' = \hat{y}_i''}$	$[0, 1]$	`clagree()`
Cohen's kappa	$s_{\text{CK}}(\hat{y}', \hat{y}'') = 1 - \frac{1 - s_{\text{ACA}}(\hat{y}', \hat{y}'')}{1 - \frac{1}{m} \sum_{k \in \mathcal{K}} \sum_{i=1}^{m} \mathbb{1}_{\hat{y}_i' = k} \sum_{i=1}^{m} \mathbb{1}_{\hat{y}_i' = k}}$	$[0, 1]$	`ckappa()`
Bhattacharyya distance	$\delta_{\text{BD}}(\hat{\pi}_i', \hat{\pi}_i'') = 1 - \sum_{k=1}^{K} \sqrt{\hat{\pi}_{ik}' \cdot \hat{\pi}_{ik}''}$	$[0, 1]$	`bdist()`
Total variation distance	$\delta_{\text{TVD}}(\hat{\pi}_i', \hat{\pi}_i'') = \frac{1}{2} \sum_{k=1}^{K} \lvert \hat{\pi}_{ik}' - \hat{\pi}_{ik}'' \rvert$	$[0, 1]$	`tvdist()`
Hellinger distance	$\delta_{\text{HD}}(\hat{\pi}_i', \hat{\pi}_i'') = \frac{1}{\sqrt{2}} \sqrt{\sum_{k=1}^{K} \left(\sqrt{\hat{\pi}_{ik}'} - \sqrt{\hat{\pi}_{ik}''} \right)^2}$	$[0, 1]$	`hdist()`
Jensen-Shannon divergence	$\delta_{\text{JSD}}(\hat{\pi}_i', \hat{\pi}_i'') = \frac{1}{2} \sum_{k=1}^{K} \left(\hat{\pi}_{ik}' \cdot \log \frac{\hat{\pi}_{ik}'}{m_{ik}} + \hat{\pi}_{ik}'' \cdot \log \frac{\hat{\pi}_{ik}''}{m_{ik}} \right), \; m_{ik} = \frac{1}{2} (\hat{\pi}_{ik}' + \hat{\pi}_{ik}'')$	$[0, 1]$	`jsdist()`

Note. m is the size of the evaluation sample, $\mathbb{1}$ denotes the indicator function and $\mathcal{K}$ denotes the set of class labels with $|\mathcal{K}| = K$.

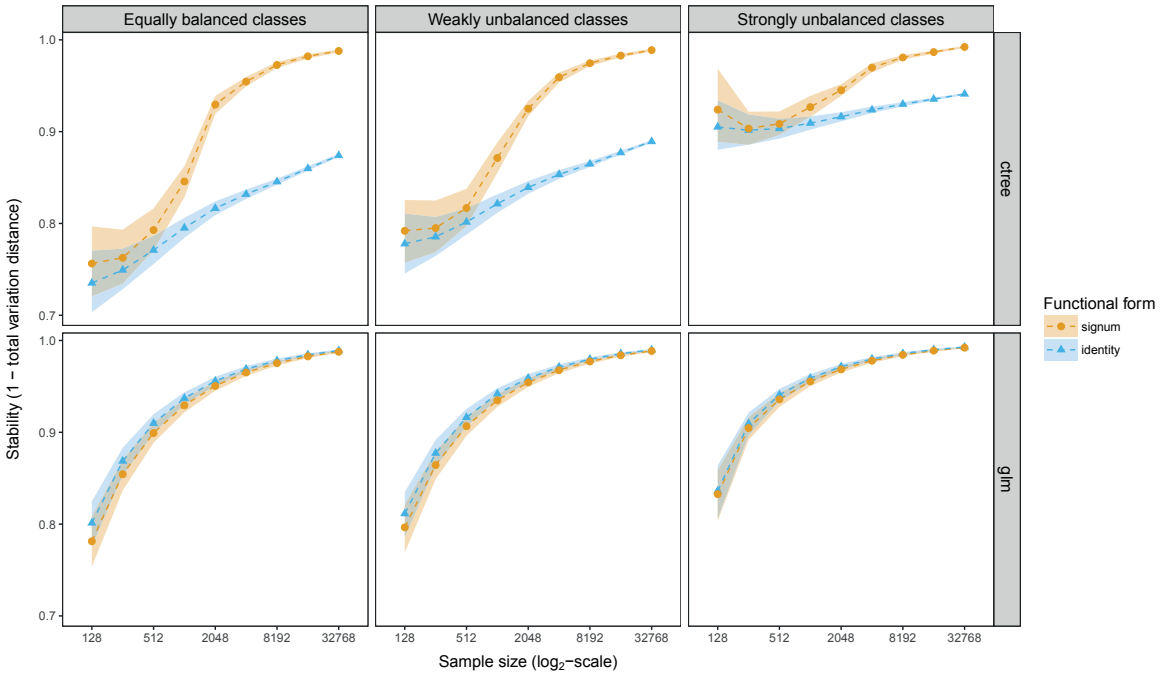


Figure A.1: Reference stability (results of Study 1) for DGPs with low dimension ($p = 20$).

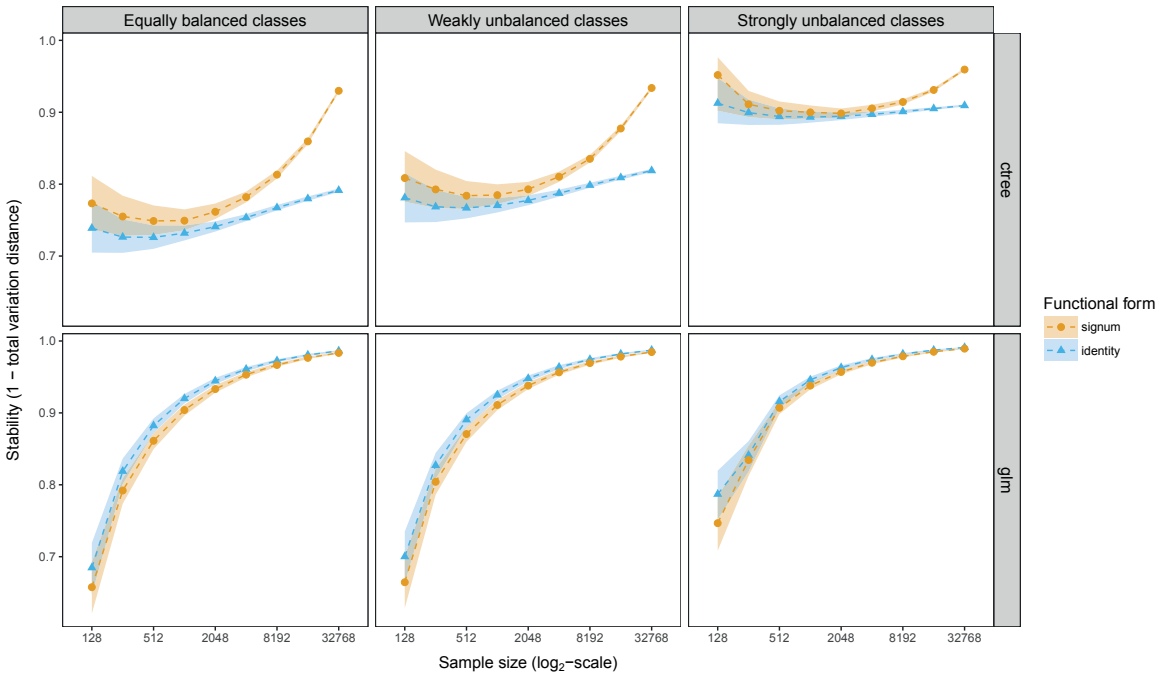


Figure A.2: Reference stability (results of Study 1) for DGPs with high dimension ($p = 40$).

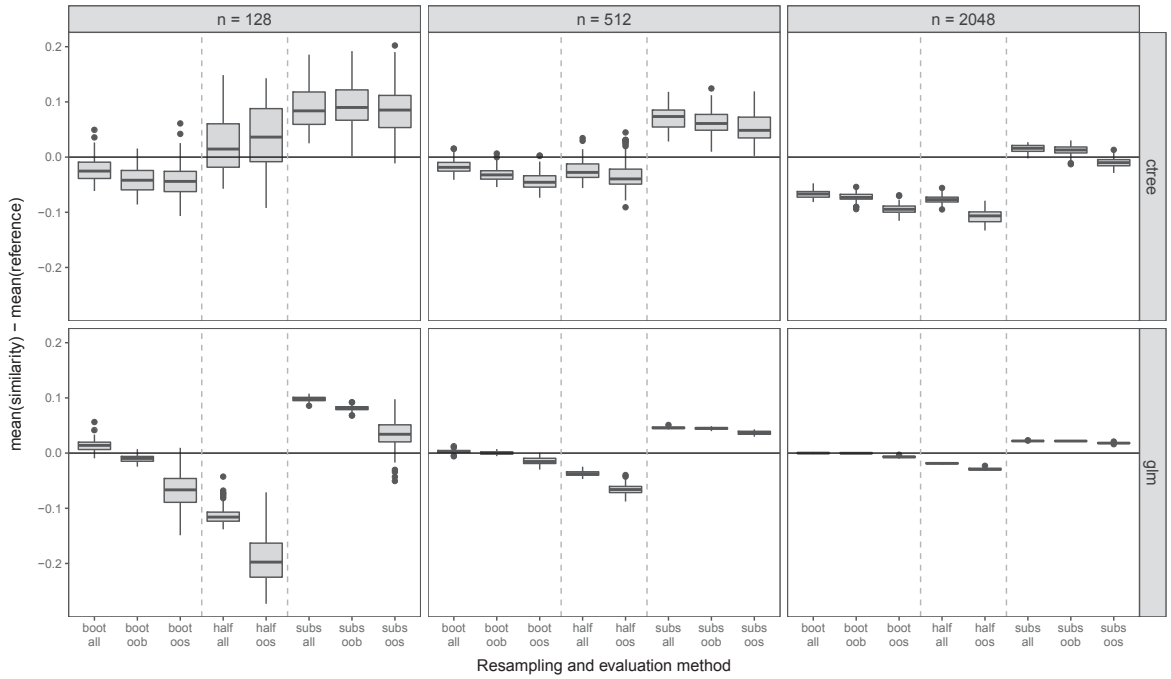


Figure A.3: Results of Study 2 for DGPs with low dimension ($p = 20$) and equally balanced classes.

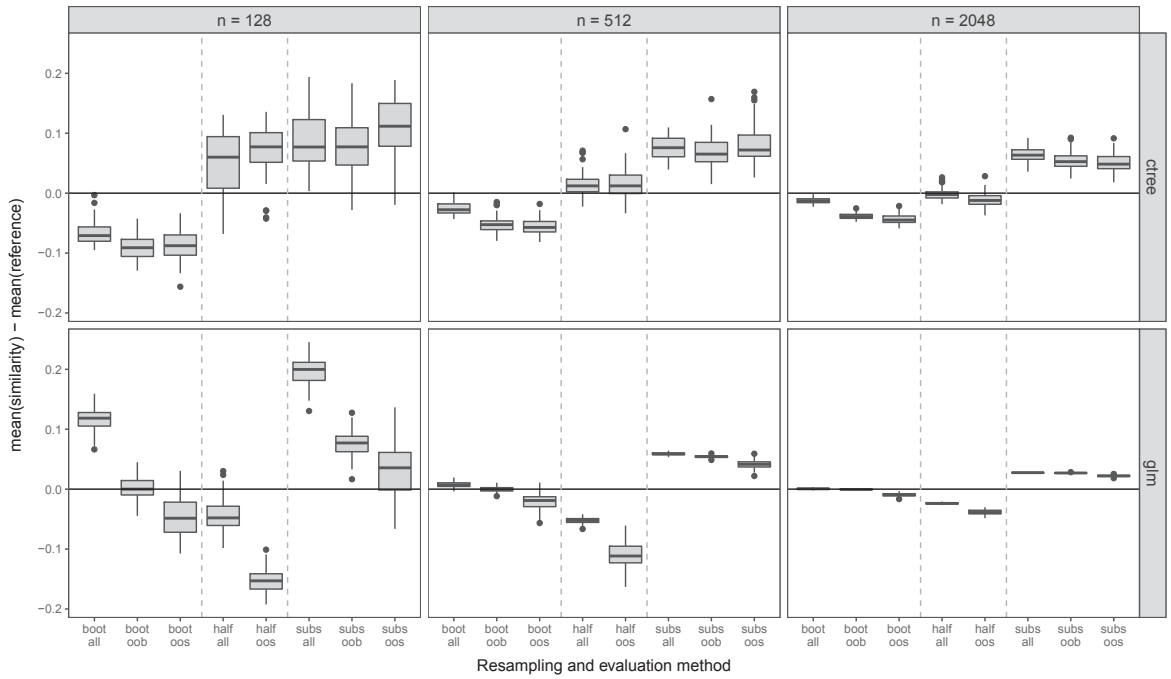


Figure A.4: Results of Study 2 for DGPs with high dimension ($p = 40$) and equally balanced classes.

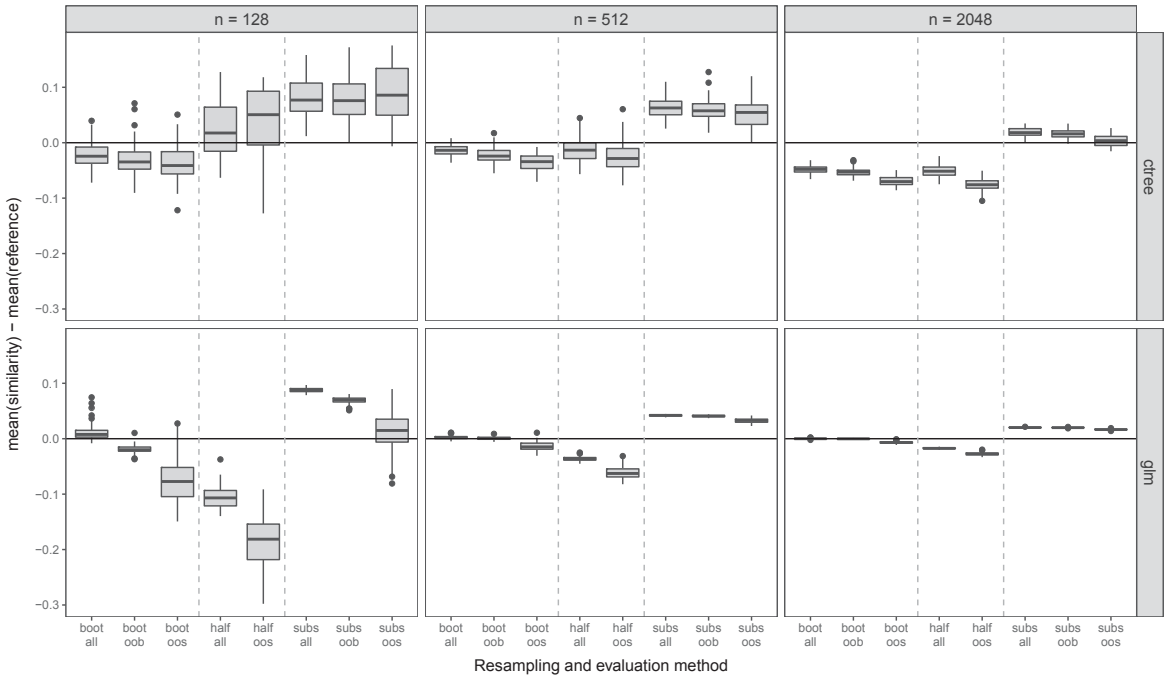


Figure A.5: Results of Study 2 for DGPs with low dimension ($p = 20$) and weakly unbalanced classes.

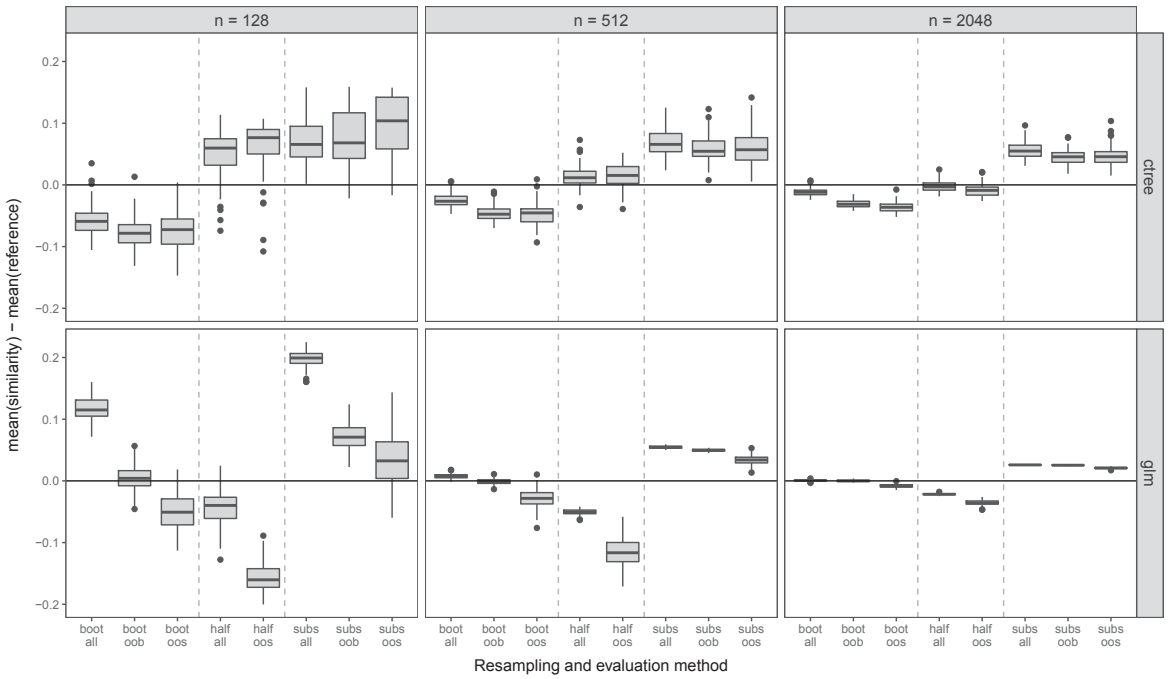


Figure A.6: Results of Study 2 for DGPs with high dimension ($p = 40$) and weakly unbalanced classes.

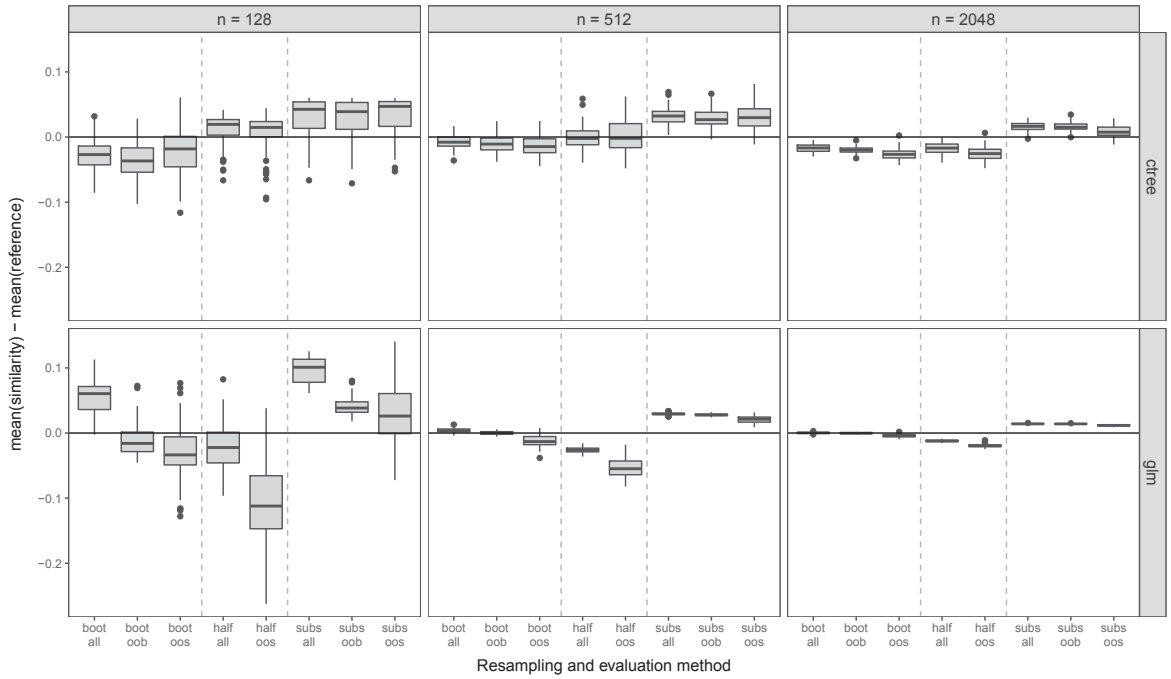


Figure A.7: Results of Study 2 for DGPs with low dimension ($p = 20$) and highly unbalanced classes.

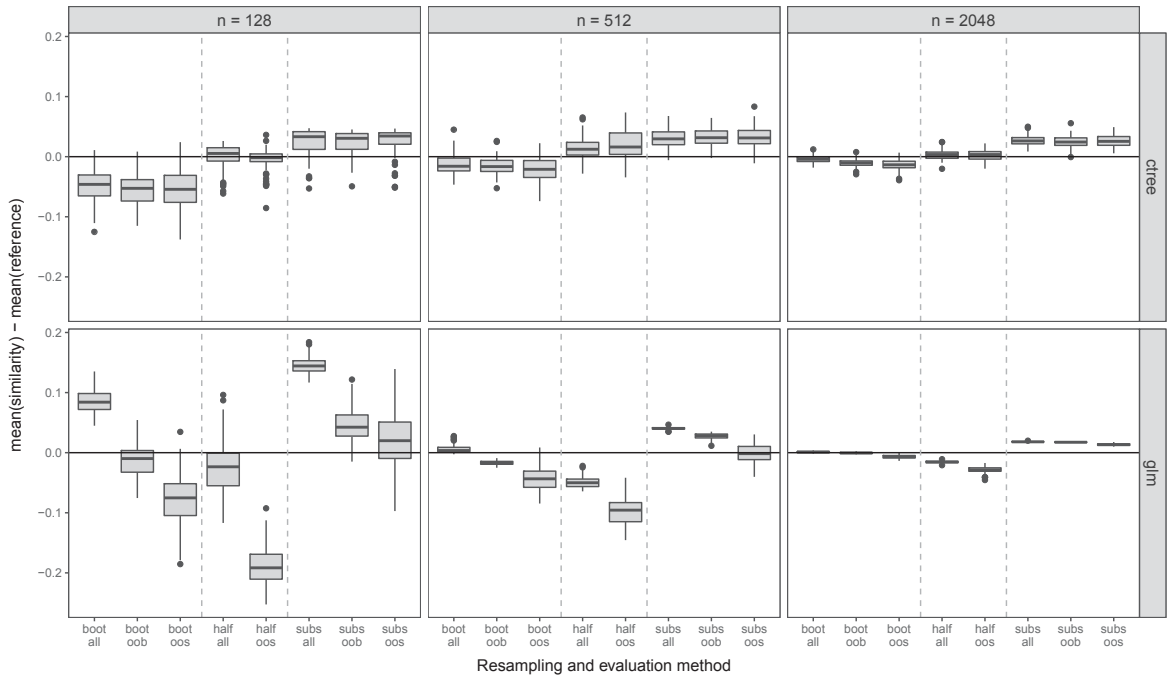


Figure A.8: Results of Study 2 for DGPs with high dimension ($p = 40$) and highly unbalanced classes.

Appendix B

Supplementary material: Chapter 3

B.1 Blockwise matrix inversion

The following statements about blockwise matrix inversion of a symmetric matrix can be used to establish the inequality between standard errors based on the complete and the incomplete information matrix discussed in Section 3.2.1. The corresponding theorems (and proofs) can be found in Chapter 13 of S. Banerjee and Roy (2014), if not stated otherwise.

Let $\boldsymbol{A}$ be a positive definite (*p.d.*) symmetric matrix, i.e. the inverse $\boldsymbol{A}^{-1}$ exists and is also *p.d.*. Suppose $\boldsymbol{A}$ is partitioned as

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{12}^\top & \boldsymbol{A}_{22} \end{pmatrix},$$

where $\boldsymbol{A}_{11}$ is $p \times p$, $\boldsymbol{A}_{12}$ is $p \times q$ and $\boldsymbol{A}_{22}$ is $q \times q$. Then its principal submatrices $\boldsymbol{A}_{11}$ and $\boldsymbol{A}_{22}$ are also invertible and *p.d.*. Let $\boldsymbol{B} = \boldsymbol{A}^{-1}$ be partitioned (similar to $\boldsymbol{A}$) as

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} \\ \boldsymbol{B}_{12}^\top & \boldsymbol{B}_{22} \end{pmatrix},$$

where $\boldsymbol{B}_{11} = \left(\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{12}^\top\right)^{-1}$ and $\boldsymbol{B}_{22} = \left(\boldsymbol{A}_{22} - \boldsymbol{A}_{12}^\top\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\right)^{-1}$ are given by the inverse of the *Schur* complements of $\boldsymbol{A}_{22}$ and $\boldsymbol{A}_{11}$, respectively, which are also *p.d.*.

By the *Sherman-Woodbury-Morrison* formula (see e.g., S. Banerjee & Roy, 2014, p. 82),

$$\begin{aligned}\left(\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{12}^{\top}\right)^{-1} &= \boldsymbol{A}_{11}^{-1} + \boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\left(\boldsymbol{A}_{22} - \boldsymbol{A}_{12}^{\top}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\right)^{-1}\boldsymbol{A}_{12}^{\top}\boldsymbol{A}_{11}^{-1} \\ \boldsymbol{B}_{11} &= \boldsymbol{A}_{11}^{-1} + \boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\boldsymbol{B}_{22}\boldsymbol{A}_{12}^{\top}\boldsymbol{A}_{11}^{-1} \\ \boldsymbol{B}_{11} &= \boldsymbol{A}_{11}^{-1} + \boldsymbol{C}^{\top}\boldsymbol{B}_{22}\boldsymbol{C}.\end{aligned}$$

where $\boldsymbol{C} = \boldsymbol{A}_{12}^{\top}\boldsymbol{A}_{11}^{-1} = (\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12})^{\top}$. For the diagonal elements, we have

$$\mathrm{diag}(\mathrm{B}_{11}) = \mathrm{diag}(\mathrm{A}_{11}^{-1}) + \mathrm{diag}(\mathrm{C}^{\top}\mathrm{B}_{22}\mathrm{C}),$$

where $\boldsymbol{B}_{11}$ and $\boldsymbol{A}_{11}^{-1}$ are both positive definite, i.e., their diagonal elements are positive.

Lemma 1. *If $\boldsymbol{B}_{22}$ and $\boldsymbol{A}_{11}^{-1}$ are positive definite and $\boldsymbol{A}_{12} \neq \boldsymbol{0}$, then each diagonal element of $\boldsymbol{C}^{\top}\boldsymbol{B}_{22}\boldsymbol{C}$ is positive.*

Proof. Since $\boldsymbol{B}_{22}$ is positive definite, $\boldsymbol{x}^{\top}\boldsymbol{B}_{22}\boldsymbol{x} > 0$ whenever $\boldsymbol{x} \neq 0$. Choosing $\boldsymbol{x} = \boldsymbol{C}\boldsymbol{e}_i$ reveals that

$$\boldsymbol{x}^{\top}\boldsymbol{B}_{22}\boldsymbol{x} = \boldsymbol{e}_i^{\top}\boldsymbol{C}^{\top}\boldsymbol{B}_{22}\boldsymbol{C}\boldsymbol{e}_i > 0,$$

where $\boldsymbol{e}_i$ is the ith unit vector that is used to extract the ith diagonal element from $\boldsymbol{C}^{\top}\boldsymbol{B}_{22}\boldsymbol{C}$. Hence, the diagonal elements in $\boldsymbol{C}^{\top}\boldsymbol{B}_{22}\boldsymbol{C}$ are also positive. □

So, if $\boldsymbol{A}_{12} \neq \boldsymbol{0}$, all diagonal elements in $\boldsymbol{C}^{\top}\boldsymbol{B}_{22}\boldsymbol{C}$ are positive and therefore,

$$\mathrm{diag}(\boldsymbol{B}_{11})_r > \mathrm{diag}(\boldsymbol{A}_{11}^{-1})_r \qquad \forall\, r \in \{1, \ldots, p\}.$$

To obtain the inequality of the standard errors as stated in Section 3.2.1, use $\boldsymbol{A} = \mathcal{I}_{\boldsymbol{\vartheta}}$ and $\boldsymbol{B} = V_{\boldsymbol{\vartheta}}$ and let $\mathcal{I}_{\boldsymbol{\beta},\boldsymbol{\pi}} \neq \boldsymbol{0}$.

Please note that the symmetric information matrix $\mathcal{I}_{\boldsymbol{\vartheta}}$ is only positive semidefinite. A positive semidefinite symmetric matrix is, however, positive definite if and only if it is nonsingular (see e.g., Harville, 2008, Corollary 14.3.12). Thus, the inequality holds if $\mathcal{I}_{\boldsymbol{\vartheta}}$ is invertible, which is required anyway to compute the standard errors.

Appendix C

R codes

The appendix demonstrates the usage of the R packages **stablelearner** and **Rcdm**. For brevity, the output of some commands is suppressed or truncated.

C.1 R package stablelearner

Functions to perform stability assessments presented in Chapters 1 and 2 are provided in the form of an add-on package for the free open source software R for statistical computing. The package is called **stablelearner** and is available online under the terms of the GNU General Public License 2 on `https://R-Forge.R-project.org/projects/stablelearner/`.

C.1.1 Installation

The package can be directly installed from R-Forge (Theußl & Zeileis, 2009) via

```
> install.packages("stablelearner", repos = "http://R-Forge.R-project.org")
```

Additional packages (Hornik, Buchta, & Zeileis, 2009; Hothorn & Zeileis, 2015; Kuhn, Weston, Coulter, & Culp, 2015; Meyer, Dimitriadou, Hornik, Weingessel, & Leisch, 2015; Ripley, 2016) that are only required to run the R commands presented below can be installed via

```
> install.packages(c("RWeka", "partykit", "C50", "e1071", "tree"))
```

The package **RWeka** is an interface to Weka (Frank, Hall, & Witten, 2016) that is written in (and requires) Java. For more information see `http://www.cs.waikato.ac.nz/ml/weka/`.

C.1.2 Stability of tree-based methods

The package must be loaded for the current R session:

```
> library("stablelearner")
```

The data used throughout this example can be retrieved directly from the package:

```
> data("titanic", package = "stablelearner")
> titanic_passenger <- subset(titanic, class %in% c("1st", "2nd", "3rd"))
```

The example illustrated here, demonstrates how to assess the stability of a result from recursive partitioning using `ctree()` from the **partykit** package that can be generated:

```
> res <- ctree(survived ~ gender + age + fare + ordered(class) + embarked +
+                sibsp + parch, data = titanic_passenger)
```

Now, the model object can be passed to the function `stabletree()` to assess its stability:

```
> stab <- stabletree(res)
```

A summary of the stability assessment can be shown by:

```
> summary(stab)

Call:
ctree(formula = survived ~ gender + age + fare + ordered(class) +
    embarked + sibsp + parch, data = titanic_passenger)

Sampler:
B = 500
Method = Bootstrap sampling

Variable selection overview:

                freq *  mean *
gender         1.000 1 1.000 1
ordered(class) 1.000 1 2.608 2
age            0.994 1 2.356 2
fare           0.790 1 0.956 1
```

```
sibsp          0.778 1 1.046 1
embarked       0.438 0 0.502 0
parch          0.240 0 0.274 0
(* = original tree)
```

In order to speed up the stability assessment process, parallelization may be activated via **parallel** for supported platforms:

```
> stabletree(res, cores = 4)
```

Visualizing tree stability assessments

The variable selection proportion can be illustrated via:

```
> barplot(stab)
```

Labels and variable ordering can be changed (for details, see `?barplot.stabletree`).

The variable selections of replications can be illustrated via:

```
> image(stab)
```

Labels and variable ordering can be changed (for details, see `?image.stabletree`).

The graphical cutpoint analysis can be illustrated via:

```
> plot(stab)
```

Several options are available to change the appearance of the plot (for details, see `?plot.stabletree`).

Sampler functions

The default sampler is `bootstrap`. A different sampler can be selected via:

```
> stabletree(res, sampler = subsampling(v = 0.8))
> stabletree(res, sampler = samplesplitting(k = 10))
> stabletree(res, sampler = jackknife(d = 1))
```

C.1.3 Stability of results from supervised statistical learning

To demonstrate the implementation of the stability measuring framework, we continue with a smaller example based on the well-known *iris* data set (for details, see `?iris`) that is a classification problem with three classes and four numeric predictors. We use recursive partitioning, but other methods can be applied as well.

```
> r1 <- ctree(Species ~ ., data = iris)
```

The stability of the results can be assessed by:

```
> stab <- stability(r1)
```

A summary of the sampling and the estimated similarity values can be shown by:

```
> summary(stab)

Call:
ctree(formula = Species ~ ., data = iris)

Sampler:
B = 500
Resampling method = Bootstrap sampling
Evaluation method = OOB

Sampling summary:
                           avg min max
Learning sample size   150.000 150 150
Learning overlap        60.204  43  72
Evaluation sample size  20.314  10  33
Evaluation overlap       0.000   0   0

Similarity summary:

, , Total variation distance (reversed)

      5%    25%   50%   75%   95%
party 0.871 0.923 0.952 0.980 0.995
```

The default similarity measure for classification problems is the total variation distance (TVD). The values are reversed by default (such that higher values indicate higher stability). This can be suppressed by:

```
> summary(stab, reverse = FALSE)
```

The number of repetitions and the similarity measure can be changed to one or more other similarity measures using `stab_control()` and the `control` argument:

```
> stability(r1, control = stab_control(B = 250, measure = list(bdist, ...)))
```

A list of implemented similarity measures can be found via `?similarity_measures`.

Comparing algorithms

The package can be used to compare the stability of results generated by different algorithms that are implemented in R. Here, we compare different algorithms for recursive partitioning.

Therefore, the packages that provide the functions for the different algorithms must be loaded to the current R session:

```
> library("rpart")
> library("C50")
> library("RWeka")
> library("tree")
```

Then, for each algorithm a result must be generated for the same model and data set:

```
> r2 <- rpart(Species ~ ., data = iris)
> r3 <- tree(Species ~ ., data = iris)
> r4 <- J48(Species ~ ., data = iris)
> r5 <- C5.0(Species ~ ., data = iris)
```

The stability can be assessed by passing all objects to the `stability()` function. For comparability, a seed may be set via the `control` argument to assess the stability of each result using the same resamplings. Labels may be passed via the argument `names` for a better recognition of the results in the summary output:

```
> stab <- stability(r1, r2, r3, r4, r5, control = stab_control(seed = 1234))
> summary(stab, names = c("ctree", "rpart", "tree", "J48", "C5.0"))

Call:
ctree(formula = Species ~ ., data = iris)
rpart(formula = Species ~ ., data = iris)
tree(formula = Species ~ ., data = iris)
```

```
J48(formula = Species ~ ., data = iris)
C5.0.formula(formula = Species ~ ., data = iris)

Sampler:
B = 500
Resampling method = Bootstrap sampling
Evaluation method = OOB

Sampling summary:
                          avg min max
Learning sample size   150.000 150 150
Learning overlap        60.108  43  73
Evaluation sample size  20.348  10  32
Evaluation overlap       0.000   0   0

Similarity summary:

, , Total variation distance (reversed)

      5%    25%   50%   75%   95%
ctree 0.871 0.930 0.956 0.978 0.994
rpart 0.849 0.902 0.941 0.976 0.992
tree  0.880 0.935 0.960 0.980 1.000
J48   0.878 0.932 0.955 0.988 1.000
C5.0  0.857 0.916 0.947 0.982 0.993
```

Illustrate similarity distributions

The similarity distribution may be illustrated graphically. A simple plot can be generated with the generic standard R plotting function:

```
> boxplot(stab, main = "Compare the Stability of some Tree-based Learners")
```

More fancy plots may be generated manually. For example by using **ggplot2** (Wickham, 2009):

```
> library("ggplot2")
> sval <- similarity_values(stab)
> pdat <- as.data.frame.table(sval) # or use melt from package reshape
> colnames(pdat) <- c("Repetition", "Learner", "Measure", "Value")
> ggplot(pdat, aes(x = Learner, y = Value)) +
```

```
+   geom_violin(aes(fill = Learner)) +
+   geom_boxplot(width = 0.1) +
+   facet_wrap(~ Measure) + # in case more than one measure is investigated
+   ylab("Stability") + theme_bw() +
+   ggtitle("Compare the Stability of some Tree-based Learners")
```

Resampling and evaluation methods

The default combination of sampling and evaluation method is bootstrap sampling with out-of-bag evaluation. The methods can be changed by the `control` argument.

For bootstrap sampling with in-sample or out-of-sample evaluation:

```
> stability(r1, control = stab_control(evaluate = "ALL"))
> stability(r1, control = stab_control(evaluate = "OOS"))
```

For subsampling with out-of-bag, in-sample or out-of-sample evaluation:

```
> stability(r1, control = stab_control(sampler = subsampling))
> stability(r1, control = stab_control(sampler = subsampling, evaluate = "ALL"))
> stability(r1, control = stab_control(sampler = subsampling, evaluate = "OOS"))
```

For splithalf sampling with in-sample or out-of-sample evaluation:

```
> stability(r1, control = stab_control(sampler = splithalf, evaluate = "ALL"))
> stability(r1, control = stab_control(sampler = splithalf, evaluate = "OOS"))
```

More details on the available resampling and evaluation methods and all other options can be found via `?stab_control`.

Define algorithms

The number of algorithms for supervised statistical learning provided in R via add-on packages is too big to be supported by the package. Thus, only very few algorithms are implemented for presentation purposes (see `?LearnerList`). The idea is that the algorithms of interest are defined by the user. An example how this can be done is given below.

Suppose we want to investigate the stability of a result generated by the support vector machine algorithm implemented in the package **e1071** that is not predefined in the package **stablelearner**.

The first step is to load the package and generate a result by:

```
> library("e1071")
> res <- svm(Species ~ ., data = iris, probability = TRUE, gamma = 0.5, cost = 4)
```

The algorithm can be added to `LearnerList` for the current R session by:

```
> newlearner <- list(
+   class   = "svm",
+   package = "e1071",
+   method  = "Support Vector Machine",
+   predfun = function(x, newdata, yclass = NULL) {
+     if(match(yclass, c("ordered", "factor"))) {
+       attr(predict(x, newdata = newdata, probability = TRUE), "probabilities")
+     } else {
+       predict(x, newdata = newdata)
+     }
+   })
> addLearner(newlearner)
```

The class of the fitted model object can be extracted using `class()`. In many cases, the generic method `predict()` is defined for the class, but the arguments tend to differ between implementations. For more information, the documentation of the corresponding package should be considered.

After all, stability can be assessed as before using `stability()`. Note that **stablelearner** currently only supports algorithms that use the standard R formula interface and the function `call` must be stored as part of the object generated by the algorithm, since the refitting is done via `update()` (see `?update` for details).

Using reweighting

Case-weighting can be activated by:

```
> stability(r1, weights = TRUE)
```

for algorithms that support these. The weights are computed according to the selected sampling and evaluation method.

User-defined case-weights can be submitted via the `weights` argument. A 3-dimensional array of size `n*B*3` must be submitted, where the first two matrices contain the learning weights and the third matrix contains the evaluation weights.

For example, to randomly select observations (without replacement) in the resampling and evaluation process one can use:

```
> n <- 150
> B <- 500
> w <- array(sample(c(0, 1), size = n*B*3, replace = TRUE), dim = c(n, B, 3))
> stability(r1, weights = w)
```

Weights can also be used to restrict the stability assessment to a particular region of the predictor space by:

```
> w <- array(NA, dim = c(n, B, 3))
> w[,,1:2] <- sample(c(0, 1), size = n*B*2, replace = TRUE)
> w[,,3]   <- iris$Sepal.Length > 4 & iris$Sepal.Width > 2 &
+   iris$Petal.Length > 4 & iris$Petal.Width > 2
> stability(r1, weights = w)
```

Artificial data

It is possible to assess the stability for a known data-generating process (DGP) given by a function that is passed to the `data` argument. The function must return a an object of class `data.frame` containing the simulated observations.

First, a DGP-function must be defined, for example, via the predefined DGP-function for two-class problems available in the package:

```
> my_dgp <- function() dgp_twoclass(n = 100, p = 2, noise = 4, rho = 0.2)
```

Then, a first result must be generated by using the function and the algorithm of interest:

```
> res <- ctree(class ~ ., data = my_dgp())
```

Now, the result and the DGP-function are passed to the `stability()` function:

```
> stability(res, data = my_dgp)
```

C.2 R package Rcdm

Functions to estimate the parameters and the standard errors in the G-DINA model (see Chapter 3) and to perform DIF detection for the DINA model (see Chapter 4) are provided in the form of an add-on package for the free open source software R for statistical computing. The package is called **Rcdm** and is freely available online under the terms of the GNU General Public Licence 2 on `https://github.com/mphili/cdm`.

C.2.1 Installation

Step 1 Install the packages that are required by **Rcdm**:

```
> install.packages(c("Rcpp", "RcppArmadillo", "limSolve", "Matrix"))
```

Install the packages that are required to run the R commands presented below:

```
> install.packages(c("strucchange", "pks"))
```

Step 2 Download the zip-file of the package from `https://github.com/mphili/cdm`, unpack it and install it via:

```
> install.packages("<path-to-directory>", repos = NULL, type = "source")
```

Alternatively, **Rcdm** can be installed directly from GitHub via **devtools** (Wickham & Chang, 2016):

```
> # install.packages("devtools")
> library("devtools")
> install_github("mphili/cdm")
```

C.2.2 Data preparation

For the examples below we use the data from the **pks** (Heller & Wickelmaier, 2013) package (see Section 3.3.2 or `?pks::probability` for a description).

The data can be loaded into the current R session via:

```
> data("probability", package = "pks")
```

Only the responses to the first set of problems are analyzed below:

```
> items <- sprintf("b1%.2i", 1:12)
> resp <- probability[, items]
```

The Q-matrix must be defined first. For this example it is given by:

```
> qmat <- t(read.table(header = FALSE, text = "
+     0    1    0    0    1    1    0    0    0    1    1    0
+     0    0    0    1    0    0    0    0    1    1    1    1
+     1    0    0    0    1    1    1    1    1    0    1    1
+     0    0    1    0    0    0    1    1    0    0    0    1
+ "))
> colnames(qmat) <- c("cp", "id", "pb", "un")
> rownames(qmat) <- colnames(resp)
```

C.2.3 Model fitting

For each new R session, the package must be loaded via:

```
> library("Rcdm")
```

The GDINA model can be fitted with the generic `gdina()` function:

```
> mGDINA <- gdina(x = resp, q = qmat)
```

Reduced models can be specified with the `rule` argument, either jointly for all items:

```
> mDINA <- gdina(x = resp, q = qmat, rule = "DINA")
> mDINO <- gdina(x = resp, q = qmat, rule = "DINO")
> mACDM <- gdina(x = resp, q = qmat, rule = "ACDM")
```

or for each item separately:

```
> itemrules <- c("DINA", "DINA", "DINA", "DINA", "ACDM", "ACDM",
+                "ACDM", "DINO", "ACDM", "DINA", "ACDM", "DINA")
> mSEP <- gdina(x = resp, q = qmat, rule = itemrules)
```

C.2.4 Model diagnosis

The fitted model and the estimated parameters can be summarized using:

```
> summary(mGDINA)

Call:
gdina(x = resp, q = qmat)
```

```
Number of items: 12
Number of skills: 4
Number of latent classes: 16
Number of respondents: 504
Number of parameters: 63

Item parameters:

   Item Itemno  Name   Rule Estimate Std.Err
1  b101      1   d_0 G-DINA   0.2026  0.0712
2  b101      1   d_1 G-DINA   0.7329  0.0728
3  b102      2   d_0 G-DINA   0.2846  0.1161
4  b102      2   d_1 G-DINA   0.7154  0.1718
5  b103      3   d_0 G-DINA   0.1043  0.0702
6  b103      3   d_1 G-DINA   0.8562  0.0713
7  b104      4   d_0 G-DINA   0.1231  0.0401
8  b104      4   d_1 G-DINA   0.8396  0.0416
...
```

For a graphical illustration of the estimates conditional response probabilities type:

```
> plot(mGDINA)
```

and for a graphical illustration of the (estimated) latent class distribution:

```
> barplot(mGDINA)
```

Wald-type confidence intervals can be computed for the item parameters via:

```
> confint(mGDINA)
              2.5 %  97.5 %
b101.d_0     0.0631  0.3422
b101.d_1     0.5902  0.8756
b102.d_0     0.0570  0.5122
b102.d_1     0.3786  1.0522
b103.d_0    -0.0333  0.2419
b103.d_1     0.7165  0.9959
b104.d_0     0.0445  0.2017
b104.d_1     0.7580  0.9211
...
```

The item-level fit can be tested as described in (de la Torre & Lee, 2013) via:

```
> item_level_fit(mGDINA)
            Kj   W value df    Pr(>W)
b101  1.000000        NA NA        NA
b102  1.000000        NA NA        NA
b103  1.000000        NA NA        NA
b104  1.000000        NA NA        NA
b105  2.000000 14.529871  2 0.0006996 ***
b106  2.000000 38.464606  2 4.441e-09 ***
b107  2.000000  7.259821  2 0.0265186 *
b108  2.000000  9.369927  2 0.0092331 **
b109  2.000000 33.230430  2 6.083e-08 ***
b110  2.000000 18.538968  2 9.426e-05 ***
b111  3.000000  7.210987  6 0.3017754
b112  3.000000  0.046081  6 0.9999980
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

C.2.5 Model comparison

Non-nested models can be compared via:

```
> BIC(mDINA, mDINO, mACDM, mGDINA) # or use AIC() instead
       df      BIC
mDINA  39 5200.459
mDINO  39 5369.075
mACDM  49 5156.883
mGDINA 63 5243.999
```

Nested models can be compared using a likelihood ratio test that is implemented in the `anova` function:

```
> anova(mDINA, mACDM, mGDINA)
Analysis of Variance Table

   Npar  logLik    AIC    BIC Df Deviance Pr(>Chi)
m1   39 -2478.9 5035.8 5200.5
m2   49 -2426.0 4950.0 5156.9 10    105.8   <2e-16 ***
m3   63 -2426.0 4978.0 5244.0 14      0.0        1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

C.2.6 DIF detection

To test an item for DIF using the Wald test (only implemented for the DINA rule):

```
> mR <- update(mDINA, x = resp[probability$sex == "m",])
> mF <- update(mDINA, x = resp[probability$sex == "f",])
> difwald(mR, mF, item = 1)

Wald-test for DIF detection

data:  objR, objF
Wj = 0.013466, df = 2, p-value = 0.9933
```

To test an item for DIF using the LM-test based on **strucchange** (Zeileis et al., 2002):

```
> difscore(mDINA, z = probability$sex, item = 1)

M-fluctuation test

data:  obj
f(efp) = 1.5877, p-value = 0.4521
```

C.3 Rasch tree example

The following R code can be used to reproduce the example of the Rasch tree.

```
> library("partykit")    # install.packages("partykit")
> library("psychotree")  # install.packages("psychotree")
```

The SPISA data can be loaded into the current R session via (more information on the data set is available via `?SPISA`):

```
> data("SPISA")
```

The Rasch tree can be generated for the items 19 to 27 of the economy domain and plotted via:

```
> SPISA_eco <- SPISA
> SPISA_eco$spisa <- SPISA_eco$spisa[,19:27]
> m <- raschtree(spisa ~ ., data = SPISA_eco)
> plot(m)
```

Appendix D

Publications

List of publications with peer review during the dissertation in chronological order of release:

- Liem, F., Mérillat, S., Bezzola, L., Hirsiger, S., Philipp, M., Madhyastha, T., and Jäncke, L. (2015). Reliability and statistical power analysis of cortical and subcortical FreeSurfer metrics in a large sample of healthy elderly. *NeuroImage*, *108*, 95–109.

- Philipp, M., Zeileis, A., and Strobl, C. (2016). A toolkit for stability assessment of tree-based learners. In A. Colubi, A. Blanco, and C. Gatu (Eds.), *Proceedings of compstat 2016 – 22nd international conference on computational statistics* (pp. 315–325). The International Statistical Institute/International Association for Statistical Computing.

- Walther, A., Phillip, M., Lozza, N., and Ehlert, U. (2016). The rate of change in declining steroid hormones: A new parameter of healthy aging in men? *Oncotarget*, *7*(38), 60844–60857.

- Philipp, M., Strobl, C., de la Torre, J., and Zeileis, A. (in press). On the estimation of standard errors in cognitive diagnosis models. *Journal of Educational and Behavioral Statistics*.

References

Andrews, R., Diederich, J., & Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, *8*(6), 373–389.

Banerjee, M., Ding, Y., & Noone, A.-M. (2012). Identifying representative trees from ensembles. *Statistics in Medicine*, *31*(15), 1601–1616.

Banerjee, S., & Roy, A. (2014). *Linear algebra and matrix analysis for statistics*. Boca Raton, FL: CRC Press.

Barakat, N., & Bradley, A. P. (2010). Rule extraction from support vector machines: A review. *Neurocomputing*, *74*(1-3), 178–190.

Bar-hen, A., Gey, S., & Poggi, J.-M. (2015). Influence measures for CART classification trees. *Journal of Classification*, *32*(1), 21–45.

Bou-Hamd, I., Larocque, D., & Ben-Ameur, H. (2011). A review of survival trees. *Statistics Surveys*, *5*, 44–71.

Bousquet, O., & Elisseeff, A. (2002). Stability and generalization. *Journal of Machine Learning Research*, *2*(3), 499–526.

Breiman, L. (1996a). Bagging predictors. *Machine Learning*, *24*(2), 123–140.

Breiman, L. (1996b). Heuristics of instability and stabilization in model selection. *The Annals of Statistics*, *24*(6), 2350–2383.

Breiman, L. (1998). Arcing classifier. *The Annals of Statistics*, *26*(3), 801–849.

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

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. Monterey, CA: Wadsworth.

Briand, B. B., Ducharme, G. R., Parache, V., & Mercat-Rommens, C. (2009). A similarity measure to assess the stability of classification trees. *Computational Statistics & Data Analysis*, *53*(4), 1208–1217.

Bürgin, R., & Ritschard, G. (2015). Tree-based varying coefficient regression for longitudinal ordinal responses. *Computational Statistics & Data Analysis*, *86*, 65–80.

Byun, H., & Lee, S.-W. (2003). A survey on pattern recognition applications of support vector machines. *International Journal of Pattern Recognition and Artificial Intelligence*, *17*(03), 459–486.

Cai, L. (2008). SEM of another flavour: Two new applications of the supplemented EM algorithm. *British Journal of Mathematical and Statistical Psychology*, *61*(2), 309–329.

Cha, S.-h. (2007). Comprehensive survey on distance / similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, *1*(4), 300–307.

Chen, J., & de la Torre, J. (2013). A general cognitive diagnosis model for expert-defined polytomous attributes. *Applied Psychological Measurement*, *37*(6), 419–437.

Chen, Y., Liu, J., Xu, G., & Ying, Z. (2015). Statistical analysis of Q-matrix based diagnostic classification models. *Journal of the American Statistical Association*, *110*(510), 850–866.

Chiu, C.-Y. (2013). Statistical refinement of the Q-matrix in cognitive diagnosis. *Applied Psychological Measurement*, *37*(8), 598–618.

Chiu, C.-Y., Douglas, J. A., & Li, X. (2009). Cluster analysis for cognitive diagnosis: Theory and applications. *Psychometrika*, *74*(4), 633–665.

Chiu, C.-Y., & Köhn, H.-F. (2015). A general proof of consistency of heuristic classification for cognitive diagnosis models. *British Journal of Mathematical and Statistical Psychology*, *68*(3), 387–409.

Ciampi, A., & Thiffault, J. (1988). Recursive partition in biostatistics: Stability of trees and choice of the most stable classification. In D. Edwards & N. E. Raun (Eds.), *Proceedings of compstat 1988 – 8th symposium on computational statistics* (pp. 267–272). Heidelberg, Germany: Physica-Verlag.

Culpepper, S. A. (2015). Bayesian estimation of the DINA model with Gibbs sampling. *Journal of Educational and Behavioral Statistics*, *40*(5), 454–476.

DeCarlo, L. T. (2011). On the analysis of fraction subtraction data: The DINA model, classification, latent class sizes, and the Q-matrix. *Applied Psychological Measurement*, *35*(1), 8–26.

de la Torre, J. (2008). An empirically based method of Q-matrix validation for the DINA model: Development and applications. *Journal of Educational Measurement*, *45*(4), 343–362.

de la Torre, J. (2009). DINA model and parameter estimation: A didactic. *Journal of Educational and Behavioral Statistics*, *34*(1), 115–130.

de la Torre, J. (2011). The generalized DINA model framework. *Psychometrika*, *76*(2), 179–199.

de la Torre, J., & Chiu, C.-Y. (2016). A general method of empirical Q-matrix validation. *Psychometrika*, *81*, 253–273.

de la Torre, J., & Douglas, J. A. (2004). Higher-order latent trait models for cognitive diagnosis. *Psychometrika*, *69*(3), 333–353.

de la Torre, J., & Douglas, J. A. (2008). Model evaluation and multiple strategies in cognitive diagnosis: An analysis of fraction subtraction data. *Psychometrika*, *73*(4), 595–624.

de la Torre, J., & Lee, Y.-S. (2010). A note on the invariance of the DINA model parameters. *Journal of Educational Measurement*, *47*(1), 115–127.

de la Torre, J., & Lee, Y.-S. (2013). Evaluating the Wald test for item-level comparison of saturated and reduced models in cognitive diagnosis. *Journal of Educational Measurement*, *50*(4), 355–373.

de la Torre, J., van der Ark, L. A., & Rossi, G. (2015). Analysis of clinical data from cognitive diagnosis modeling framework. *Measurement and Evaluation in Counseling and Development*. (Advance online publication)

DeMars, C. E. (2010). Type I error inflation for detecting DIF in the presence of impact. *Educational and Psychological Measurement*, *70*(6), 961–972.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, *39*(1), 1–38.

Fischer, G., & Molenaar, I. (2012). *Rasch models: Foundations, recent developments, and applications*. New York, NY: Springer.

Fokkema, M., Smits, N., & Zeileis, A. (2015). *Detecting treatment-subgroup interactions in clustered data with generalized linear mixed-effects model trees* (Working Paper No. 2015-10). Austria: Research Platform Empirical and Experimental Economics, Universität Innsbruck.

Frank, E., Hall, M. A., & Witten, I. H. (2016). *The weka workbench. online appendix for "data mining: Practical machine learning tools and techniques"* (4th ed.). Morgan Kaufmann.

Garre, F. G., & Vermunt, J. K. (2006). Avoiding boundary estimates in latent class analysis by bayesian posterior mode estimation. *Behaviormetrika*, *33*(1), 43–59.

George, A. C. (2013). *Investigating CDMs: Blending theory with practicality* (Doctoral dissertation, TU Dortmund University, Germany). Retrieved from `https://eldorado.tu-dortmund.de/`

Glas, C. A. W. (1998). Detection of differential item functioning using lagrange multiplier tests. *Statistica Sinica*, *8*(3), 647–667.

Goldstein, A., Kapelner, A., Bleich, J., & Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, *24*(1), 44–65.

Goshtasby, A. A. (2012). *Image registration: Principles, tools and methods.* London, England: Springer.

Haertel, E. H. (1989). Using restricted latent class models to map the skill of achievement structure items. *Journal of Educational Measurement*, *26*(4), 301–321.

Harville, D. A. (2008). *Matrix algebra from a statistician's perspective.* New York, NY: Springer.

Heller, J., & Wickelmaier, F. (2013). Minimum discrepancy estimation in probabilistic knowledge structures. *Electronic Notes in Discrete Mathematics*, *42*, 49–56.

Henson, R. A., Templin, J. L., & Willse, J. T. (2009). Defining a family of cognitive diagnosis models using log-linear models with latent variables. *Psychometrika*, *74*(2), 191–210.

Hjort, N. L., & Koning, A. (2002). Tests for constancy of model parameters over time. *Journal of Nonparametric Statistics*, *14*(1-2), 113–132.

Holland, P. W., & Wainer, H. (Eds.). (1993). *Differential item functioning.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Hornik, K., Buchta, C., & Zeileis, A. (2009). Open-source machine learning: R meets Weka. *Computational Statistics*, *24*(2), 225–232.

Hothorn, T., Hornik, K., & Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, *15*(3), 651–674.

Hothorn, T., Leisch, F., Zeileis, A., & Hornik, K. (2005). The design and analysis of benchmark experiments. *Journal of Computational and Graphical Statistics*, *14*(3), 675–699.

Hothorn, T., & Zeileis, A. (2015). partykit : A toolkit for recursive partytioning. *Journal of Machine Learning Research*, *16*, 3905–3909.

Hou, L., de la Torre, J., & Nandakumar, R. (2014). Differential item functioning assessment in cognitive diagnostic modeling: Application of the Wald test to investigate DIF in the DINA model. *Journal of Educational Measurement*, *51*(1), 98–125.

Junker, B. W., & Sijtsma, K. (2001). Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. *Applied Psychological Measurement*, *25*(3), 258–272.

Komboz, B., Strobl, C., & Zeileis, A. (2016). Tree-based global model tests for polytomous rasch models. *Educational and Psychological Measurement*. (Advance online publication)

Kopf, J. (2013). *Model-based Recursive Partitioning Meets Item Response Theory* (Doctoral dissertation, Ludwig-Maximilians-Universität München, Germany). Retrieved from `https://edoc.ub.uni-muenchen.de/`

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*. New York, NY: Springer.

Kuhn, M., Weston, S., Coulter, N., & Culp, M. (2015). C50: C5.0 decision trees and rule-based models [Computer software manual]. (R package version 0.1.0-24)

Lange, T., Braun, M. L., Roth, V., & Buhmann, J. M. (2002). Stability-based model selection. *Advances in Neural Information Processing Systems*, *15*, 617–624.

Lee, Y.-S., Park, Y. S., & Taylan, D. (2011). A cognitive diagnostic modeling of attribute mastery in Massachusetts, Minnesota, and the U.S. national sample using the TIMSS 2007. *International Journal of Testing*, *11*(2), 144–177.

Li, F. (2008). *A modified higher-order DINA model for detecting differential item functioning and differential attribute functioning* (Doctoral dissertation, University of Georgia, Athens). Retrieved from `http://athenaeum.libs.uga.edu/`

Li, H. (2011). A cognitive diagnostic analysis of the MELAB reading test. *Spaan Fellow Working Papers in Second or Foreign Language Assessment*, *9*, 17–46.

Li, X., & Wang, W.-c. (2015). Assessment of differential item functioning under cognitive diagnosis models: The DINA model example. *Journal of Educational Measurement*, *52*(1), 28–54.

Lichman, M. (2013). *UC irvine machine learning repository.* URL: `http://archive.ics.uci.edu/ml`.

Liem, F., Mérillat, S., Bezzola, L., Hirsiger, S., Philipp, M., Madhyastha, T., & Jäncke, L. (2015). Reliability and statistical power analysis of cortical and subcortical FreeSurfer metrics in a large sample of healthy elderly. *NeuroImage*, *108*, 95–109.

Lim, C., & Yu, B. (2016). Estimation stability with cross validation. *Journal of Computational and Graphical Statistics*, *25*(2), 464–492.

Lin, L., Hedayat, A. S., Sinha, B., & Yang, M. (2002). Statistical methods in assessing agreement. *Journal of the American Statistical Association*, *97*(457), 257–270.

McLachlan, G., & Krishnan, T. (2007). *The EM algorithm and extensions* (2nd ed.). New York, NY: Wiley.

Merkle, E. C., Fan, J., & Zeileis, A. (2014). Testing for measurement invariance with respect to an ordinal variable. *Psychometrika*, *79*(4), 569–584.

Merkle, E. C., & Zeileis, A. (2013). Tests of measurement invariance without subgroups: A generalization of classical methods. *Psychometrika*, *78*(1), 59–82.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., & Leisch, F. (2015). e1071: Misc functions of the department of statistics, probability theory group (formerly: E1071), tu wien [Computer software manual]. (R package version 1.6-7)

Miglio, R. R., & Soffritti, G. (2004). The comparison between classification trees through proximity measures. *Computational Statistics & Data Analysis*, *45*(3), 577–593.

Molinaro, A. M., Simon, R., & Pfeiffer, R. M. (2005). Prediction error estimation: A comparison of resampling methods. *Bioinformatics*, *21*(15), 3301–3307.

Mukherjee, S., Niyogi, P., Poggio, T., & Rifkin, R. (2006). Learning theory: Stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, *25*(1-3), 161–193.

Neyman, J., & Scott, E. L. (1948). Consistent estimates based on partially consistent observations. *Econometrica*, *16*(1), 1–32.

Ntoutsi, I., Kalousis, A., & Theodoridis, Y. (2008). A general framework for estimating similarity of datasets and decision trees: Exploring semantic similarity of decision trees. In M. Zaki, H. Park, C. J. Apte, & K. Wang (Eds.), *Proceedings of the 2008 siam international conference on data mining* (pp. 810–821). Philadelphia, PA: Society for Industrial and Applied Mathematics.

Paek, I., & Cai, L. (2014). A comparison of item parameter standard error estimation procedures for unidimensional and multidimensional item response theory modeling. *Educational and Psychological Measurement*, *74*(1), 58–76.

Philipp, M., Rusch, T., Hornik, K., & Strobl, C. (2016). *Measuring the stability of results from statistical learning.* (Manuscript under review)

Philipp, M., Strobl, C., de la Torre, J., & Zeileis, A. (in press). On the estimation of standard errors in cognitive diagnosis models. *Journal of Educational and Behavioral Statistics*.

Philipp, M., Zeileis, A., & Strobl, C. (2016). A toolkit for stability assessment of tree-based learners. In A. Colubi, A. Blanco, & C. Gatu (Eds.), *Proceedings of compstat 2016 – 22nd international conference on computational statistics* (pp. 315–325). The International Statistical Institute/International Association for Statistical Computing.

Poggio, T., Rifkin, R., Mukherjee, S., & Niyogi, P. (2004). General conditions for predictivity in learning theory. *Nature*, *428*, 419–422.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Meteo, CA: Morgan Kaufmann Publishers Publishers Inc.

R Core Team. (2016). R: A language and environment for statistical computing [Computer software manual]. Vienna, Austria.

Ripley, B. (2016). tree: Classification and regression trees [Computer software manual]. (R package version 1.0-37)

Rojas, G. (2013). *Cognitive diagnosis models: Attribute classification, differential item functioning and applications* (Doctoral dissertation, Universidad Autónomia de Madrid, Spain). Retrieved from `https://repositorio.uam.es/`

Rupp, A. A., & Templin, J. (2007). The effects of Q–matrix misspecification on parameter estimates and classification accuracy in the DINA model. *Educational and Psychological Measurement*, *68*(1), 78–96.

Rupp, A. A., Templin, J., & Henson, R. A. (2010). *Diagnostic measurement: Theory, methods, and applications*. New York, NY: Guilford Press.

Rusch, T., Hofmarcher, P., Hatzinger, R., & Hornik, K. (2012). Model trees with topic model pre-processig: An approach for data journalism illustrated with the Wikileaks Afghanistan war logs. *Annals of Applied Statistics*, *7*(2), 613–639.

Rusch, T., & Zeileis, A. (2013). Gaining insight with recursive partitioning of generalized linear models. *Journal of Statistical Computation and Simulation*, *83*(7), 1301–1315.

Schapire, R. E., Freund, Y., Bartlett, P., & Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, *26*(5), 1654–1686.

Shannon, W. D., & Banks, D. (1999). Combining classification trees using MLE. *Statistics in Medicine*, *18*(6), 727–740.

Shmueli, G. (2010). To explain or to predict? *Statistical Science*, *25*(3), 289–310.

Song, L., Wang, W., Dai, H., & Ding, S. (2012). The revised DINA model parameter estimation with EM algorithm. *International Journal of Digital Content Technology and its Applications*, *6*(9), 85–92.

Stodden, V. (2015). Reproducing statistical results. *Annual Review of Statistics and Its Application*, *2*(1), 1–19.

Strobl, C., Kopf, J., & Zeileis, A. (2015). Rasch Trees: A new method for detecting differential item functioning in the Rasch model. *Psychometrika*, *80*(20), 289–361.

Strobl, C., Malley, J., & Tutz, G. (2009). An introduction to recursive partitioning: Rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological methods*, *14*(4), 323–348.

Strobl, C., Wickelmaier, F., & Zeileis, A. (2011). Accounting for individual differences in Bradley-Terry models by means of recursive partitioning. *Journal of Educational and Behavioral Statistics*, *36*(2), 135–153.

Tatsuoka, K. K. (1983). Rule space: An approach for dealing with misconceptions based in item response theory. *Journal of Educational Measurement*, *20*(4), 345–354.

Tatsuoka, K. K. (1990). Toward an integration of item-response theory and cognitive error diagnosis. In N. Frederiksen, R. Glaser, A. Lesgold, & M. Shafto (Eds.), *Diagnostic monitoring of skill and knowledge acquisition* (pp. 453–488). Hillsdale, NJ: Lawrence Erlbaum Associates.

Templin, J. L., & Henson, R. a. (2006). Measurement of psychological disorders using cognitive diagnosis models. *Psychological Methods*, *11*(3), 287–305.

Theußl, S., & Zeileis, A. (2009). Collaborative software development using R-Forge. *The R Journal*, *1*(1), 9–14.

Thissen, D., & Wainer, H. (1982). Some standard errors in item response theory. *Psychometrika*, *47*(4), 397–412.

Trepte, S., & Verbeet, M. (Eds.). (2010). *Allgemeinbildung in Deutschland – Erkenntnisse aus dem SPIEGEL Studentenpisa-Test*. Wiesbaden, Germany: VS Verlag.

Tukey, J. W. (1962). The future of data analysis. *The Annals of Mathematical Statistics*, *33*(1), 1–67.

Turney, P. (1995). Technical note: Bias and the quantification of stability. *Machine Learning*, *20*(1-2), 23–33.

Vellido, A., Lisboa, P. J. G., & Vaughan, J. (1999). Neural networks in business: A survey of applications (1992–1998). *Expert Systems with Applications*, *17*(1), 51–70.

von Davier, M. (2008). A general diagnostic model applied to language testing data. *British Journal of Mathematical and Statistical Psychology*, *61*(2), 287–307.

Wager, S., Hastie, T., & Efron, B. (2014). Confidence intervals for random forests: The jackknife and the infinitesimal jackknife. *Journal of Machine Learning Research*, *15*, 1625–1651.

Walker, E., & Nowacki, A. S. (2011). Understanding equivalence and noninferiority testing. *Journal of general internal medicine*, *26*(2), 192–6.

Walther, A., Phillip, M., Lozza, N., & Ehlert, U. (2016). The rate of change in declining steroid hormones: A new parameter of healthy aging in men? *Oncotarget*, *7*(38), 60844–60857.

Wasserman, L. (2004). *All of statistics: A concise course in statistical inference*. New York, NY: Springer.

Wickelmaier, F. (2016). *Using recursive partitioning to account for parameter heterogeneity in multinomial processing tree models* (Working Paper No. 2016-26). Austria: Research Platform Empirical and Experimental Economics, Universität Innsbruck.

Wickham, H. (2009). *ggplot2: Elegant graphics for data analysis*. New York, NY: Springer.

Wickham, H., & Chang, W. (2016). devtools: Tools to make developing r packages easier [Computer software manual]. (R package version 1.12.0)

Woods, C. M., Cai, L., & Wang, M. (2012). The Langer-improved Wald test for DIF testing with multiple groups: Evaluation and comparison to two-group IRT. *Educational and Psychological Measurement*, *73*(3), 532–547.

Yang, X., & Embretson, S. E. (2007). Construct validity and cognitive diagnostic assessment. In J. P. Leighton & M. J. Gierl (Eds.), *Cognitive diagnostic assessment for education: Theory and applications* (pp. 119–145). New York, NY: Cambridge University Press.

Yu, B. (2013). Stability. *Bernoulli*, *19*(4), 1484–1500.

Yuan, K. H., Cheng, Y., & Patton, J. (2014). Information matrices and standard errors for MLEs of item parameters in IRT. *Psychometrika*, *79*(2), 232–254.

Zeileis, A., Hothorn, T., & Hornik, K. (2008). Model-based recursive partitioning. *Journal of Computational and Graphical Statistics*, *17*(2), 492–514.

Zeileis, A., Leisch, F., Hornik, K., & Kleiber, C. (2002). strucchange : An R package for testing for structural change in linear regression models. *Journal of Statistical Software*, *7*(2), 1–38.

Zhang, H., & Singer, B. (2010). *Recursive partitioning and applications*. New York, NY: Springer.

Zhang, W. (2006). *Detecting differential item functioning using the DINA model* (Doctoral dissertation, University of North Carolina, Chapel Hill). Retrieved from `https://libres.uncg.edu/ir/uncg/`

Zhang, Y., Wu, H., & Cheng, L. (2012). Some new deformation formulas about variance and covariance. In *Proceedings of 2012 international conference on modelling, identification and control* (pp. 1042–1047). Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Zhong, Y., Meacham, C. A., & Pramanik, S. (1997). A general method for tree-comparison based on subtree similarity and its use in a taxonomic database. *Bio Systems*, *42*(1), 1–8.