American Psychological Association

Psychological Methods

ISSN: 1082-989X

2020, Vol. 25, No. 6, 776–786
http://dx.doi.org/10.1037/met0000262

One Model to Rule Them All? Using Machine Learning Algorithms to Determine the Number of Factors in Exploratory Factor Analysis

David Goretzko and Markus Bühner
Ludwig Maximilians University Munich

Abstract

Determining the number of factors is one of the most crucial decisions a researcher has to face when conducting an exploratory factor analysis. As no common factor retention criterion can be seen as generally superior, a new approach is proposed—combining extensive data simulation with state-of-the-art machine learning algorithms. First, data was simulated under a broad range of realistic conditions and 3 algorithms were trained using specially designed features based on the correlation matrices of the simulated data sets. Subsequently, the new approach was compared with 4 common factor retention criteria with regard to its accuracy in determining the correct number of factors in a large-scale simulation experiment. Sample size, variables per factor, correlations between factors, primary and cross-loadings as well as the correct number of factors were varied to gain comprehensive knowledge of the efficiency of our new method. A gradient boosting model outperformed all other criteria, so in a second step, we improved this model by tuning several hyperparameters of the algorithm and using common retention criteria as additional features. This model reached an out-of-sample accuracy of 99.3% (the pretrained model can be obtained from https://osf.io/mvrau/). A great advantage of this approach is the possibility to continuously extend the data basis (e.g., using ordinal data) as well as the set of features to improve the predictive performance and to increase generalizability.

Translational Abstract

Determining the number of factors is one of the most important decisions a researcher has to face when conducting an exploratory factor analysis. No common method for this purposes is always superior, so a new approach is proposed. The new approach combines an extensive data simulation with machine learning algorithms (complex statistical modeling). In a first step, data that reflect typical application contexts are created. Then in a second step, the machine learning modeling is used to find data characteristics that are able to predict the dimensionality (the number of factors). In a large-scale simulation study, this new approach was compared to four common methods varying the sample size, the number of variables per factor, the correlations between factors, loading magnitudes and the true number of factors. The new approach outperformed all four common approaches. Further, it was shown that the accuracy of the new approach could be increased by improving the machine learning model and the data basis. As the data basis can be easily extended for further application contexts, our new method promises better decision-making in exploratory factor analysis.

Keywords: exploratory factor analysis, number of factors, machine learning, factor retention criteria, factorial validity

Supplemental materials: http://dx.doi.org/10.1037/met0000262.supp

Exploratory factor analysis (EFA) is a commonly used statistical method to explore latent psychological concepts. Its exploratory nature allows researchers to carve out the structure of new constructs, but also reflects a threat to the validity of its results as several methodological decisions have to be taken by the researchers—decisions that can strongly shape the outcome and future research in the respective field (Fabrigar, Wegener, MacCallum, & Strahan, 1999; Goretzko, Pham, & Bühner, 2019). The most crucial decision might be determining the number of factors that should be extracted. Extracting too few factors (underfactoring) or too many (overfactoring) can have adverse effects on the estimated factor scores (Fava & Velicer, 1996) and leads to estimation problems such as Heywood cases (De Winter & Dodou, 2012). Overfactoring is generally regarded as less critical because the actual relations between the manifest variables and the latent variables can be estimated more accurately than in cases of underfactoring (Fabrigar et al., 1999).

There are numerous ways to determine the number of factors. In some cases, theoretical considerations set the number of factors,

This article was published Online First March 5, 2020.

David Goretzko and Markus Bühner, Department of Psychology, Ludwig Maximilians University Munich.

Correspondence concerning this article should be addressed to David Goretzko, Department of Psychology, Ludwig Maximilians University Munich, Leopoldstraße 13, 80802 Munich, Germany. E-mail: david.goretzko@psy.lmu.de

but often such implications are missing and the number has to be estimated based on the empirical data. The majority of the various methods that have been developed for this issue evaluate the eigenvalue-structure of the item correlations. There are traditional approaches like the scree test (Cattell, 1966), the Kaiser-Guttman rule (Kaiser, 1960), and the parallel analysis (Horn, 1965) as well as modern approaches like the comparison data (CD) approach (Ruscio & Roche, 2012) or the empirical Kaiser criterion (EKC, Braeken & Van Assen, 2017). Parallel analysis (PA) became some kind of gold-standard (e.g., Fabrigar et al., 1999; Goretzko et al., 2019) due to its robustness against varying distributional assumptions (Dinno, 2009) and its comparably good performance under various conditions such as sample sizes between 30 and 360 and number of variables between nine and 72 (Peres-Neto, Jackson, & Somers, 2005; Zwick & Velicer, 1986). Nonetheless, new methods like CD (Ruscio & Roche, 2012), the hull method proposed by Lorenzo-Seva, Timmerman, and Kiers (2011) and EKC (Braeken & Van Assen, 2017) showed superiority for specific data conditions.[1] A broad simulation study by Auerswald and Moshagen (2019) evaluated these (and other) criteria under various conditions (number of items: four to 60, sample sizes: 100 to 1,000, number of factors: one to five, variables per factor: four to 12, varying loading magnitudes and interfactor correlations) and recommended combining different methods. They found combinations of PA (based on principal component analysis), EKC, the hull method, and CD (when sample sizes were sufficiently large) to provide the best results.

Aim of the Study

Combining various methods is also recommended by several other authors (Fabrigar et al., 1999; Goretzko et al., 2019), yet this suggestion can be unsettling and frustrating for practitioners. For this reason, a new approach is tested in this study—combining extensive data simulation with machine learning algorithms to find a model that is predictive (can determine the number of factors correctly) under a broad range of conditions. We focus on two major approaches: random forest (Breiman, 2001) and extreme gradient boosting (Chen, He, Benesty, Khotilovich, & Tang, 2018) as well as an automatic gradient boosting approach (Thomas, Coors, & Bischl, 2018) because both random forest and gradient boosting are able to reflect nonlinearities and complex interactions.

Random Forest

Random forests are based on n_{tree} bootstrap samples drawn from the empirical data. A regression or classification tree is grown on each bootstrap sample by recursive binary splitting. This growth-process stops when a minimum threshold for observations in each terminal node is reached. The algorithm randomly uses m_{try} variables at each node, of which one variable is selected that allows for the best split (Breiman, 2001). The resulting trees are not pruned like single decision trees, because overfitting is prevented by averaging over the n_{tree} trees. These trees can vary a lot as only m_{try} of all features are used for each possible split and the bootstrapped samples do not consist of all original observations. The process of building a random forest by averaging the m_{try} trees provides reliable results (James, Witten, Hastie, & Tibshirani, 2013 for a more detailed introduction). Common values for m_{try} are $\frac{p}{3}$ or $\sqrt{p}$ (with p being the number of variables or features). $\frac{p}{3}$ is preferred for regression task while $\sqrt{p}$ might be favorable for classification purposes (Breiman, 1999).

Extreme Gradient Boosting and Automatic Gradient Boosting

The principal idea of boosting is to sequentially perform a prediction task (regression or classification) with comparably simple (or "weak") methods like decision trees (Friedman, Hastie, & Tibshirani, 2000). Contrary to random forests or bagging approaches, no bootstrap samples are drawn. In fact, a number of decision trees (n_{tree}) is grown sequentially using the residuals of the complete model containing all previous trees. A shrinkage parameter (e.g., λ) also known as learning rate regulates the updating speed:

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^{new}(x)$$

where $\hat{f}(x)$ is the iteratively updated model, which yields $\hat{f}(x) = \sum_{n=1}^{n_{tree}} \lambda \hat{f}^{n}(x)$ for the complete boosted model. λ is usually chosen as $\lambda = 0.01$ or $\lambda = 0.001$ (James et al., 2013).

For gradient boosting this rate is not fixed for each new tree, but rather computed by minimizing the residuals in the respective step given a predefined loss-function (Friedman, 2001). As the *xgboost* algorithm (Chen & Guestrin, 2016; Chen et al., 2018) used for this study contains several hyperparameters (e.g., a learning rate that shrinks the step weights, a L_1-regularization and a L_2-regularization as well as tree-specific parameters like the minimal node size), tuning the *xgboost* model might be promising. Thomas, Coors, and Bischl (2018) provided an automatized version—the automatic gradient boosting or *autoxgboost* which applies model-based optimization (Bayesian optimization) to find the best set of hyperparameters.

Method

The idea of this study was to find a machine learning model that is predictive for the true number of factors (the number of latent dimensions underlying the data generating process) in the context of EFA. We therefore simulated several data sets with given factor structures that reflect realistic conditions in psychological research. Afterward, three machine learning models (random forest, gradient boosting, and automatic gradient boosting) were trained on this data (see Figure 1 for a flowchart demonstrating the approach). The performance of the resulting three trained models were compared with the performance of parallel analysis, the comparison data approach and the empirical Kaiser criterion as well as the common Kaiser criterion. This was done using new simulated data that also covered a broad range of conditions usually found in psychological literature.

Creating a Machine Learning Model as Factor Retention Criterion

The underlying data basis was simulated assuming multivariate normality. Sample sizes were between 200 and 1,000, the true

[1] CD was superior to PA especially in conditions with few factors ($k < 5$), while the EKC was superior for conditions with oblique factor structures and the Hull method outperformed PA when overdetermination was high.

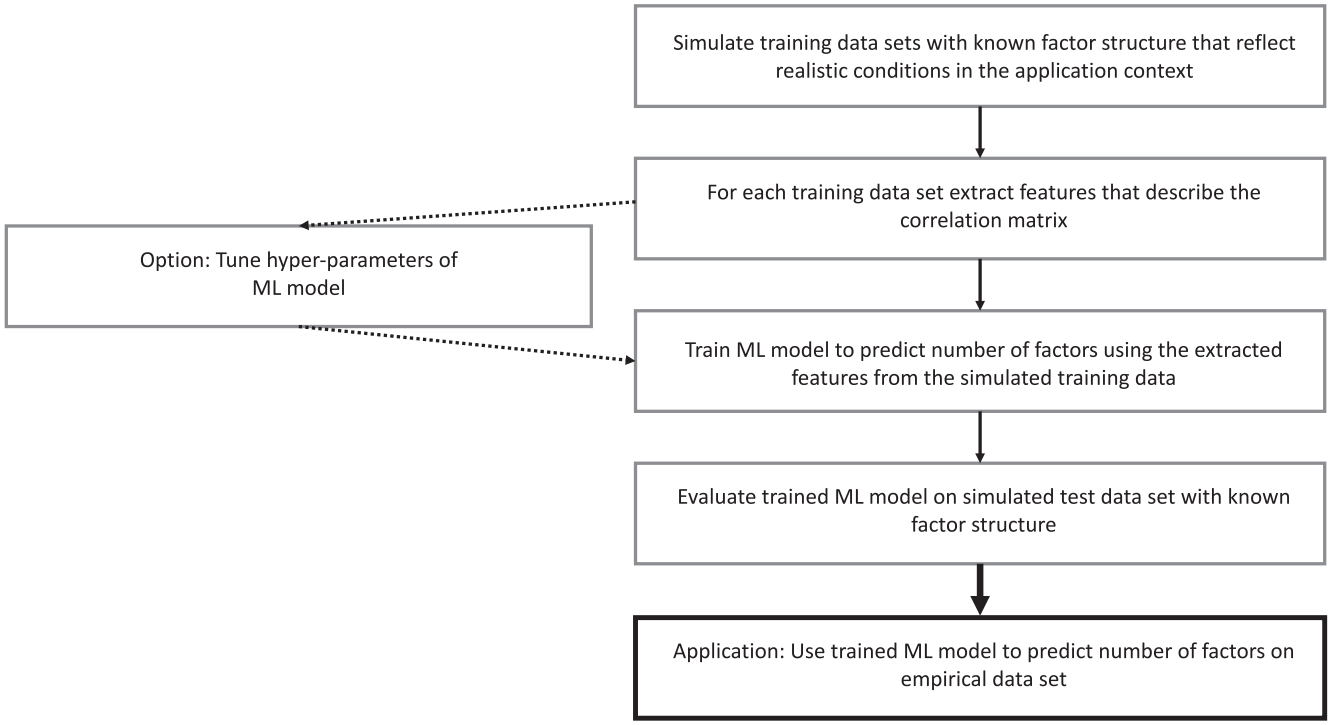

Figure 1. Visualization of the new factor retention approach. ML = machine learning.

number of factors (k) ranged from one to eight factors, variables per factor (vpf) varied between three and 10, factor correlations were set to values between 0 and 0.4, primary loadings ranged from 0.35 to 0.80 and cross-loadings from 0.00 and 0.20.[2]

A population correlation matrix was created for each data set based on the following decomposition:

$$\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Phi}\boldsymbol{\Lambda}^{\top} + \boldsymbol{\Psi}^2$$

with $\boldsymbol{\Lambda}$ being the true loading matrix, $\boldsymbol{\Phi}$ being the factor correlation matrix and $\boldsymbol{\Psi}^2$ being a diagonal matrix containing the unique variance of each variable. The true loading matrix contained all primary and cross-loadings drawn from different uniform distributions (e.g., when primary loadings should be high a uniform distribution between 0.65 and 0.80 was used). For $\boldsymbol{\Phi}$ a matrix that consists of the value one on the diagonal and equal values for the interfactor correlations on the off-diagonal (0, 0.1, 0.2, 0.3, or 0.4) was chosen, while $\boldsymbol{\Psi}^2$ was calculated from $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Phi}\boldsymbol{\Lambda}^{\top} + \mathbb{1}_{p\times p} - diag(\boldsymbol{\Lambda}\boldsymbol{\Phi}\boldsymbol{\Lambda}^{\top})$. Data simulation and analysis were conducted with R (R Development Core Team, 2008) while the article was written with the *papaja* package (Aust & Barth, 2018) and graphics were created with the *ggplot2* package (Wickham, 2016). We used the *mvtnorm* package (Genz et al., 2018) to simulate multivariate normal data with the respective correlation matrix $\boldsymbol{\Sigma}$ (consequently, all manifest variables had unit variance) and N observations (N was drawn from a uniform distribution between 200 and 1,000).

Feature engineering. One-hundred and 81 features were computed for each simulated data set to create the respective training data for the machine learning algorithms. The following features were used for training: the sample size N, the number of variables p, the number of eigenvalues that are greater than one, the relative proportion of the first eigenvalue, the relative proportion of the first two eigenvalues, the relative proportion of the first three eigenvalues, the number of eigenvalues that are greater than 0.7, the standard deviation of all eigenvalues, the number of eigenvalues that accounts for 50% of the variance, the number of eigenvalues that accounts for 75% of the variance, the L_1-norm of the correlation matrix, the Frobenius-norm of the correlation matrix, the maximum-norm of the correlation matrix, the spectral-norm of the correlation matrix, the average of the off-diagonal correlations, the number of correlations smaller or equal to 0.1, the average of the initial communality estimates, the determinant of the correlation matrix, the measure of sampling adequacy (MSA after Kaiser (1970)), the Gini-coefficient (Gini, 1921) of the correlation matrix, the Kolm measure of inequality (Kolm, 1999) of the correlation matrix, all p eigenvalues as well as all p eigenvalues of the factor model.[3]

We simulated 500,000 data sets varying the sample size, primary and cross-loadings, the number of factors, the variables per factor, and the factor correlations. For some of the random combinations, the resulting $\boldsymbol{\Sigma}$-matrix was not positive semidefinite or the calculation of

[2] Primary loadings were either small ($\lambda_{ij} \in [0.35; 0.50]$), medium ($\lambda_{ij} \in [0.50; 0.65]$), or high ($\lambda_{ij} \in [0.65; 0.80]$) and cross-loadings were either nonexisting, small ($\lambda_{ij} \in [0.00; 0.10]$), or medium sized ($\lambda_{ij} \in [0.10; 0.20]$) representing different levels of communalities.

[3] For both common eigenvalues and eigenvalues based on the factor model, the maximum number was 80 as $p = k \times vpf$ could have been $p = 8 \times 10$ in maximum. Missing values (for each simulated data set with $p < 80$) were coded with $-1{,}000$.

all features was not feasible, so our effective training sample consisted of 498,971 simulated data sets with 181 features.

Model training. Based on the simulated data, we used the machine learning framework *mlr* (Bischl et al., 2016) to train a random forest (*ranger*, Wright & Ziegler, 2017), the extreme gradient boosting model (*xgboost*, Chen et al., 2018), and the automatic gradient boosting model (*autoxgboost*, Thomas et al., 2018). Determining the true number of factors k was implemented as a classification task (multiclass) with no specific cost-matrix—the algorithms were trained to maximize the accuracy of the suggested factor solution which means that it was not differentiated among falsely classified cases (e.g., suggesting four factors when $k = 2$ had the same costs as suggesting five factors).

The *ranger* was applied with default settings ($n_{tree} = 500$, which seems to be a good trade-off between performance and the need for computational resources (Genuer, Poggi, & Tuleau, 2008) and $m_{try} = floor(\sqrt{p})$ which is the rounded down square root of the number of features as recommended by Breiman (1999) - in our case $m_{try} = floor(\sqrt{181}) = 13$). We also used the default settings of the *xgboost,*[4] but set the number of iterations (the trees that are sequentially build on the residuals) to 500 for better comparison with the *ranger*. The *autoxgboost* was used with default settings as well. We only increased the time budget[5] of the algorithm from 1 hr to 2 hrs.

We saved the three trained models to evaluate them on new data. The *ranger* model reached an in-sample accuracy of 97.2% while the *xgboost* model had an in-sample accuracy of 99.0% and the *autoxgboost* model an in-sample accuracy of 95.8%.[6]

Evaluation of the Machine Learning Models and Four Common Factor Retention Criteria

To evaluate the performance of the three models in more detail and on new data, we created several experimental conditions and simulated multivariate normal data. We varied the following conditions: sample size was $N = 250$, 500, or 1,000; variables per factor were either $vpf = 4$, 5, or 7; the true number of factors was $k = 1$, 2, 4, or 6; between factor correlations[7] were $\rho = 0$, 0.1, 0.2, 0.3, or 0.5 and loadings varied between 0.35 and 0.80 for primary loadings (different conditions with small ($\lambda_{ij} \in [0.35; 0.50]$), medium ($\lambda_{ij} \in [0.50; 0.65]$) and high ($\lambda_{ij} \in [0.65; 0.80]$) primary loadings) and 0.00 and 0.20 for cross-loadings (different conditions with no, small ($\lambda_{ij} \in [0.00; 0.10]$) and medium sized ($\lambda_{ij} \in [0.10; 0.20]$) cross-loadings). This gave us 3,204 conditions in total as we excluded combinations that could yield improper solutions for Σ. Each condition was replicated 500 times, so 1,512,000 data sets were evaluated. Data simulation was conducted analogously to the simulation of the data basis for the training set.

We calculated all necessary features and saved the predictions of the three models for each data set. We also collected the suggested number of factors for four common factor retention criteria (Kaiser criterion, PA, CD, EKC) for comparison. Accuracies, ratios of under- or overfactoring as well as minimum and maximum of the suggested number of factors per conditions were then calculated. For the sake of clarity, we used only four common criteria for the comparison that can be used as a baseline[8] for our new approach (e.g., foregoing the hull method which is superior to PA only in rather special conditions with high overdetermination and the minimum average partial test (MAP; Velicer, 1976) which is designed for principal component analyses rather than EFA in a narrow sense and which is not able to outperform PA (Zwick & Velicer, 1986)). For the same reason, we also focused on one implementation of the parallel analysis (using the 95% quantile of the eigenvalue distribution of random data for comparison as implemented in the *psych* package by Revelle (2018)), so all results concerning PA are related to a specific implementation and cannot be generalized to other types of parallel analyses. The simulation studies of Auerswald and Moshagen (2019) and Lim and Jahng (2019) provide further insights on these different types of parallel analysis.

Results

Averaged over all 3,204 conditions, both trained models *ranger* and *xgboost* had a higher accuracy than the common factor retention criteria. When considering conditions with $k = 1$, 2, 4, and 6 separately, the *xgboost* model had the highest accuracy on average for two-factor, four-factor, and six-factor solutions and fell short closely behind the EKC when one-factor solutions were evaluated (99.7% to 99.9%). While all retention criteria general had lower accuracies when k was higher, the *xgboost* model exclusively reached accuracies higher than 85% on average. Table 1 shows the accuracies of all methods averaged over all conditions as well as the accuracies for all different values of k separately.

Besides having the highest overall accuracy the *xgboost* model showed no signs of biased estimation of the number of factors (estimated bias smaller than 0.01). The *ranger* showed no clear signs of bias in the estimation as well, yet it tended to overfactor when $k = 6$ (10.9% of the cases). In contrast, PA and CD tended to underfactor suggesting less than k factors for all values of k—a tendency that increased with higher values of k (e.g., when $k = 6$ PA underestimated k in 22.6% of the cases while CD did so in 23.3% of the cases). The Kaiser-Guttman-rule rather tended to overfactor (20.3% of the cases with $k = 4$ and 25.9% of the cases with $k = 6$), while EKC was nearly unbiased for $k = 1$ and $k = 2$ and suggested less than k factors when four or six factors (22.0% of the cases) were in the data generating model. In Table 2, the averaged suggested numbers of factors are presented for each criterion and the four different values of k, while the proportions of under- and overfactoring are displayed in greater detail in the online supplementary materials (Supplementary Table 2).

[4] The default settings were used as we wanted to know whether the *xgboost* algorithm is useful at all. As there are many possible tuning parameters for this algorithm, the *autoxgboost* implementation was tested as well.

[5] The time budget is the maximum time that will be used for tuning the parameters of the underlying boosting algorithm via Bayesian optimization.

[6] The apparently lower accuracy (in-sample) of the *autoxgboost* compared with the *xgboost* indicates that the time budget was too short to find optimal settings for the hyperparameters of the gradient boosting algorithm.

[7] We evaluated oblique structures with (nearly) simple structure rather than the related orthogonal structures with higher cross-loadings as researchers almost always search for simple structure and many common rotation methods were designed to provide solutions with simple structure (Browne, 2001; Fabrigar et al., 1999; Goretzko et al., 2019).

[8] We focused on common factor retention criteria that are applied by practitioners and seemed to be promising for our data conditions. Including methods for this baseline made sense for those criteria that were shown to be superior to PA for some of these data conditions.

Table 1
Accuracy of Factor Retention Criteria Averaged Over All Conditions and for Different Factor Solutions Separately

Method	Acc	Acc_1	Acc_2	Acc_4	Acc_6
xgboost	0.92886	0.99663	0.95883	0.90646	0.85002
ranger	0.91508	0.99625	0.94913	0.90314	0.80701
axgboost	0.85600	0.99583	0.93834	0.79607	0.68620
PA	0.82506	0.76951	0.90248	0.85971	0.76590
CD	0.81304	0.85624	0.90297	0.79599	0.69157
EKC	0.88432	0.99916	0.95634	0.82596	0.74983
Kaiser	0.74644	0.96389	0.85003	0.64023	0.52161

Note. PA = parallel analysis; CD = comparison data; EKC = empirical Kaiser criterion. Acc is the overall accuracy of each method, whereas Acc1 is the accuracy for single factor conditions, Acc2 for conditions with two factors and so on.

All factor retention criteria improved their performance with increasing sample size. Especially the simulation based approaches PA and CD as well as the Kaiser criterion strongly benefited from greater samples. EKC, on the contrary, showed almost the same performance for all three values of *N* (mean accuracies: 87.7%, 88.5%, 89.1%). Table 3 displays the averaged accuracies for all sample sizes separately.

The accuracy of factor retention varied for different levels of variables per factor (the item-to-factor ratio) as well as for different combinations of primary and cross-loadings. Figures 2–5 show the performance of all methods for conditions with $k = 1$ (see Figure 2), $k = 2$ (see Figure 3), $k = 4$ (see Figure 4), and $k = 6$ (see Figure 5) and all combinations of these three variables (*vpf,* primary loadings and cross-loadings). All figures can also be found in Supplementary B where they are presented in color and in greater detail (different values of ρ and *N*).

When $k = 1$ and primary loadings were high, all methods except CD achieved almost perfect accuracy, while PA failed to retain the correct number of factors more often than all other methods when primary loadings were small. Especially when $vpf = 4$ or 5, PA yielded an averaged accuracy below 30%. The three ML models, EKC, and the Kaiser criterion achieved very high accuracies throughout all respective conditions.

When $k = 2$ and primary loadings were small, the Kaiser criterion and PA performed worse than the other retention criteria. While, in general, a higher number of variables per factor yielded better results, the Kaiser criterion and EKC showed opposing tendencies (when primary loadings were small) with worse results the more variables per factor were present. When cross-loadings were medium and primary loadings were small, all methods had averaged accuracies below 90% with EKC being the only exception when $vpf = 7$. In conditions with high primary loadings, all criteria performed reasonably well, yet CD failed to retain the true number of factors more often than the other methods (see Figure 3).

Figure 4 and Figure 5, show the averaged accuracy for conditions with four and six true factors respectively. When $k = 4$ and primary loadings were small the Kaiser criterion performed quite poorly, while the empirical Kaiser criterion yielded high accuracies. The *xgboost* model was often superior to the other retention criteria, especially when cross-loadings got higher. When primary loadings were high all criteria performed better, yet the EKC showed some problems with relatively high cross-loadings and only four variables per factor. When $k = 6$ all methods lacked accuracy for conditions with small primary loadings and comparably high cross-loadings. Only the *xgboost* model reached an accuracy higher than 50% on average in these conditions. When $k = 6$, $vpf = 7$, and primary loadings were small (no cross-loadings), the Kaiser criterion was not able to retain the correct number of factors in a single data set. It rather suggested more than six factors for each case.

Additional Conditions

As the data sets used for this evaluation were based on loading matrices containing solely positive loadings, we added 16 data conditions with both negative and positive loadings to find out whether our new approach (the *xgboost* model as it outperformed both the

Table 2
Mean and Median Solution for Each Number of Factors

Method	M_1	$Bias_1$	$Median_1$	M_2	$Bias_2$	$Median_2$	M_4	$Bias_4$	$Median_4$	M_6	$Bias_6$	$Median_6$
xgboost	1.00404	0.00404	1.00000	1.99281	−0.00719	1.99383	4.00270	0.00270	4.00247	6.00720	0.00720	6.02067
ranger	1.00465	0.00465	1.00000	1.98084	−0.01916	1.99506	4.01826	0.01826	4.00000	6.03094	0.03094	6.03682
axgboost	1.00491	0.00491	1.00000	1.99562	−0.00438	1.99383	4.08578	0.08578	4.05123	6.02002	0.02002	6.02196
PA	0.83837	−0.16163	0.77778	1.87148	−0.12852	1.89383	3.74010	−0.25990	3.78272	5.30216	−0.69784	5.33269
CD	1.16709	0.16709	1.00000	2.00030	0.00030	1.96235	3.79098	−0.20902	3.75432	5.45288	−0.54712	5.39664
EKC	1.00079	0.00079	1.00000	1.96730	−0.03270	1.97284	3.69512	−0.30488	3.71358	5.35788	−0.64212	5.40310
Kaiser	1.03779	0.03779	1.03704	2.11237	0.11237	2.09136	4.16895	0.16895	4.16543	6.21663	0.21663	6.20155

Note. PA = parallel analysis; CD = comparison data; EKC = empirical Kaiser criterion. Mean1 means the average suggested number of factors for conditions with one true factor, Median1 is the respective median number, and Bias1 the average deviance in these conditions.

Table 3
Accuracy of Factor Retention Criteria for Different Sample Sizes

Method	Acc_{250}	Acc_{500}	Acc_{1000}
xgboost	0.88754	0.93549	0.96355
ranger	0.88408	0.92132	0.93985
axgboost	0.81523	0.86006	0.89271
PA	0.75991	0.83494	0.88032
CD	0.74920	0.82486	0.86505
EKC	0.87652	0.88499	0.89145
Kaiser	0.65478	0.75597	0.82856

Note. PA = parallel analysis; CD = comparison data; EKC = empirical Kaiser criterion. Acc_{250} is the mean accuracy for conditions with $N = 250$, Acc_{500} for conditions with $N = 500$ and Acc_{1000} for conditions with $N = 1{,}000$.

ranger and the *autoxgboost*) can handle respective data. We also evaluated the performance of the *xgboost* model compared to the four common retention criteria under a condition based on a random intercept model with 10 variables loading equally on the first factor and unequally on a second factor. This random intercept condition can provide first insights on how the *xgboost* model behaves in conditions fundamentally different to conditions based on a (near) simple structure. Table 4 shows that the *xgboost* model performed quite well under these additional conditions being able to retain the correct number of factors almost in every case, while the EKC struggled with the random intercept model (1% accuracy).

Feature Importance

While common retention criteria are derived from statistical theory (e.g., EKC) or are grounded on it, the machine learning models only use components of these criteria (e.g., eigenvalues) as features and provide a prediction for the dimensionality *k* based on rather complex interaction patterns. Despite their black box character, it is possible to calculate measures of feature importance indicating the influence of features on the prediction. In case of the best performing machine learning model, the *xgboost* model, both "inequality" measures—Kolm measure and Gini coefficient—had the highest importance followed by several corrected eigenvalues of the factor model and the averaged value of the bivariate correlations (see Table 5). The feature importance values for all 181 features can be found in the online supplementary materials (Supplementary Tables 3–6).

Tuning the Best Model

Using default settings in this study, the *xgboost* model outperformed both the random forest and all common retention criteria. Although, the in-sample accuracy (the accuracy based on the training data) had already been quite high, tuning hyperparameters of the algorithm might raise the predictive power of the model, so we decided to improve the final model in a second step. In addition to hyperparameter-tuning, the results of PA, CD and EKC were added as features for training the machine learning model in this second step as well.

Data were simulated as described before. In total, we used 246,355[9] simulated data sets to reduce computational costs. Additionally, 70% of these data sets served as training data. As the *xgboost* implementation allows for tuning several hyperparameters, we used six out of eight parameters that were defined as the simple space for the *autoxgboost* algorithm (Thomas et al., 2018). Table 6 shows these six parameters with the lower and upper bounds chosen for tuning the model.

Again, we applied the *mlr* framework (Bischl et al., 2016) to train the model and to tune the six parameters. Eighty percent of the data from the training set (137,959 data sets) were used as the actual training set while 20% served as an internal test set (34,489 data sets) for an early stopping rule as implemented in the *autoxgboost* algorithm (Thomas et al., 2018). We set the *early_stopping_rounds* argument of this implementation to five[10] and used the fast histogram optimized algorithm (https://github.com/dmlc/xgboost/issues/1950) as the tree construction algorithm to save computation time.

For tuning purposes, we decided not to rely on a predefined grid, but use the iterative racing procedure (*irace*) by López-Ibáñez, Dubois-Lacoste, Pérez Cáceres, Stützle, and Birattari (2016). It allows for automatic parameter configuration as it samples possible parameter configurations from iteratively updated distributions in the parameter space. With regard to the sample size of the training set and the idea that the training data were simulated as "representative" for real-world scenarios, we used a holdout set ($\frac{1}{3}$ of the actual training set which equaled 45,986 data sets) for the tuning procedure.

After six iterations with 2,483 so-called experiments (configurations tested), the following hyperparameter set was selected $\eta = 0.158$, $\gamma = 0.015$, *max_depth* = 4, *colsample_bytree* = 0.789, $\lambda = 0.005$ and *subsample* = 0.812. The tuned *xgboost* model reached an out-of-sample accuracy of 99.3% on the test set (30% of the 246,355 data sets = 73,907 data sets).[11]

Discussion

In this study, we present a new approach to determine the number of factors in EFA. We combined different machine learning algorithms with a large data simulation to build a model that can predict the true number of factors based on features of the empirical correlation matrix. The used *xgboost* model was able to constantly outperform the common retention criteria, even under conditions that were outside the range that the model was trained on (e.g., $\rho = 0.5$ and the random intercept model). Because the simulation study had to focus on some general data conditions in order to ensure an adequate scope, we were not able to evaluate the performance of the *xgboost* model under all potential conditions. Therefore, some condition variations, such as different numbers of variables for each factor, were not considered. However, many conditions not covered by the evaluation study were still included in the test set for the tuned *xgboost* model. Thus, the new approach should be able to deal with this kind of data. Hence, our study can

[9] 250,000 data sets were simulated originally, but in 3,645 cases either population correlation matrices were not semipositive definite or internal simulations of comparison data (CD approach) were causing errors.

[10] This means that when no improvement in the performance measure is indicated for five iterations the algorithm stops.

[11] We applied a slightly different importance measure to look at the feature importance of the tuned *xgboost* model. All three added factor retention criteria were among the 20 most important features and helped to improve the prediction performance.

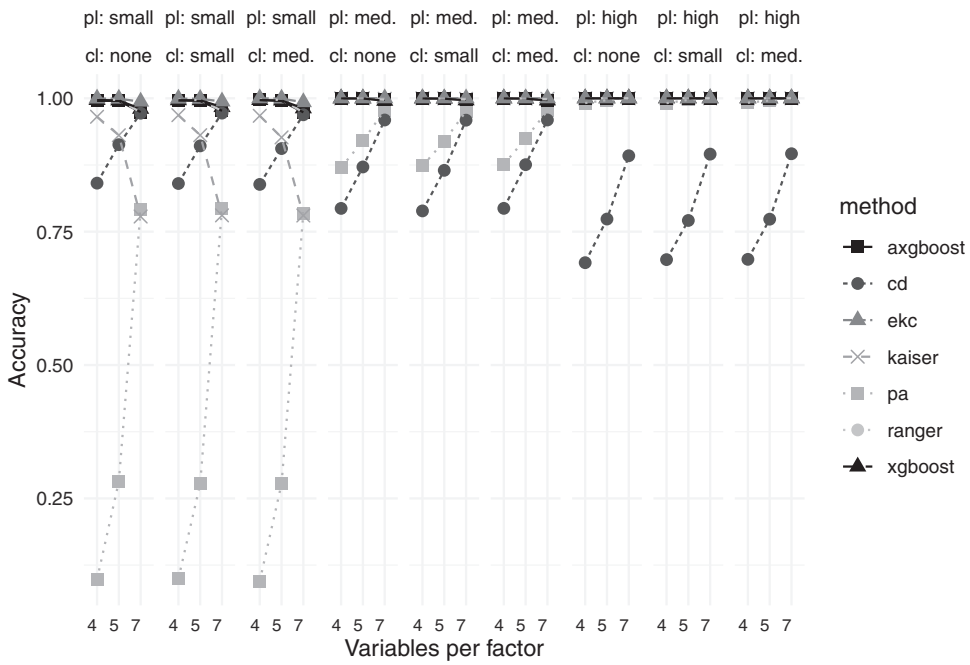

Figure 2. Accuracy of retention criteria for conditions with one factor averaged over *N*. PA = parallel analysis; CD = comparison data; EKC = empirical Kaiser criterion.

be seen as a proof of concept. While all common factor retention criteria showed some tendencies of bias (i.e., over- or underfactoring) the *xgboost* model estimated the number of factors without bias (for all *k*). This is a great advantage of the trained model as all common retention criteria perform poorly under some circumstances which is why several authors recommend combinations of different criteria (Auerswald & Moshagen, 2019; Fabrigar et al., 1999; Goretzko et al., 2019). The machine learning models, are not theoretically founded like the EKC, for example. Nevertheless, they are able to reflect the complex relations between the number

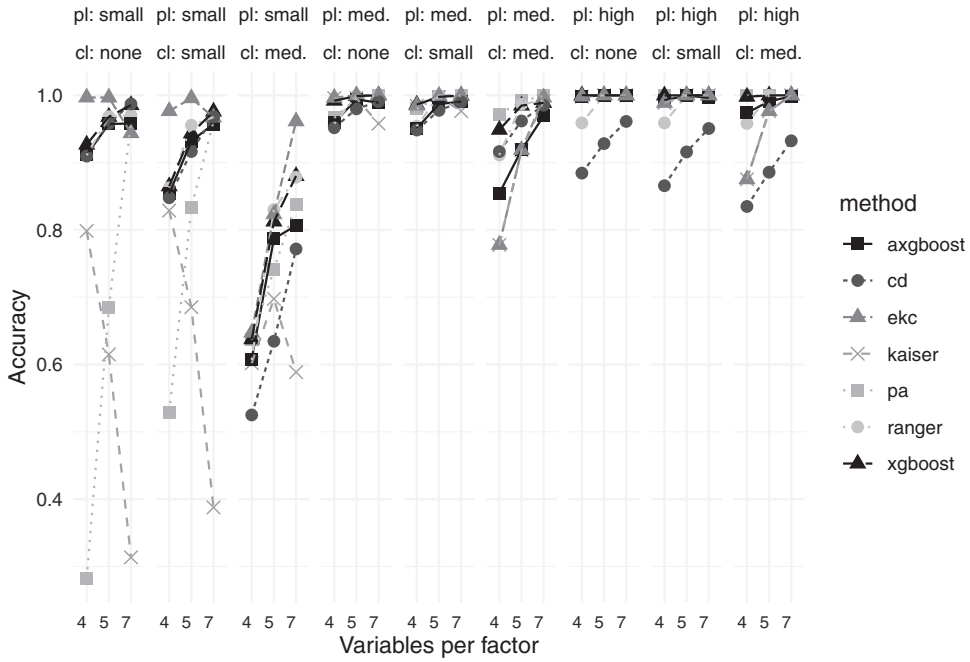

Figure 3. Accuracy of retention criteria for conditions with two factors averaged over *N* and ρ. PA = parallel analysis; CD = comparison data; EKC = empirical Kaiser criterion.

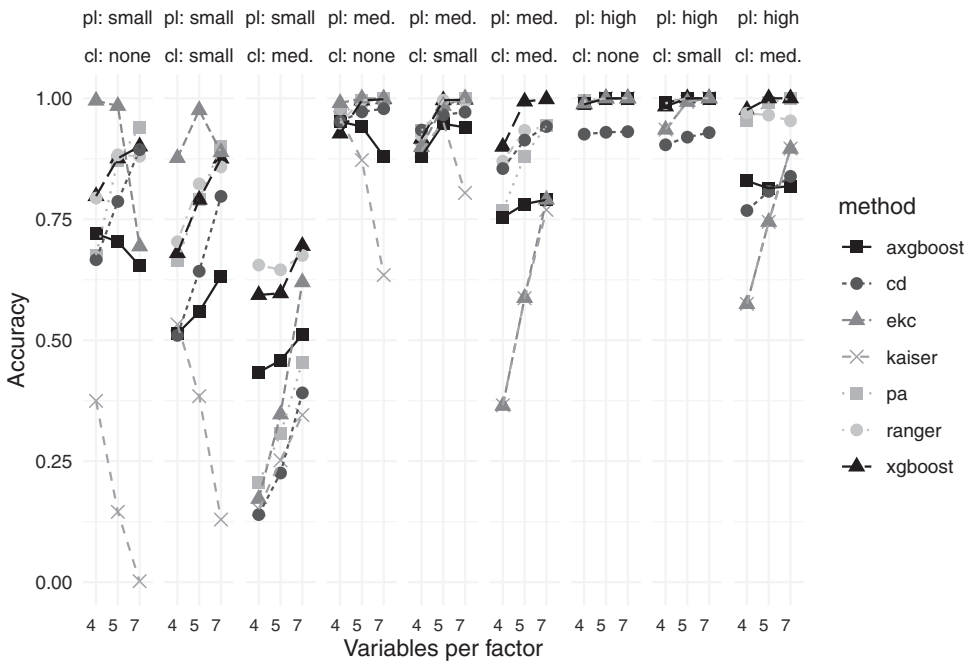

Figure 4. Accuracy of retention criteria for conditions with four factors averaged over *N* and ρ. pa = parallel analysis; cd = comparison data; ekc = empirical Kaiser criterion.

of factors and the data characteristics (in this case described by 181/184 features) almost perfectly as demonstrated by the tuned *xgboost* model with its out-of-sample accuracy of 99.3%.

The dependency on the simulated data basis can be seen as a weakness of the new approach. When empirical data is fundamentally different to this data basis, model predictions are probably invalid and not trustworthy. However, this study showed that the performance of the *xgboost* model was quite good in conditions not completely covered in the data basis it was trained on. In fact, when the data basis is sufficiently large and all possible data

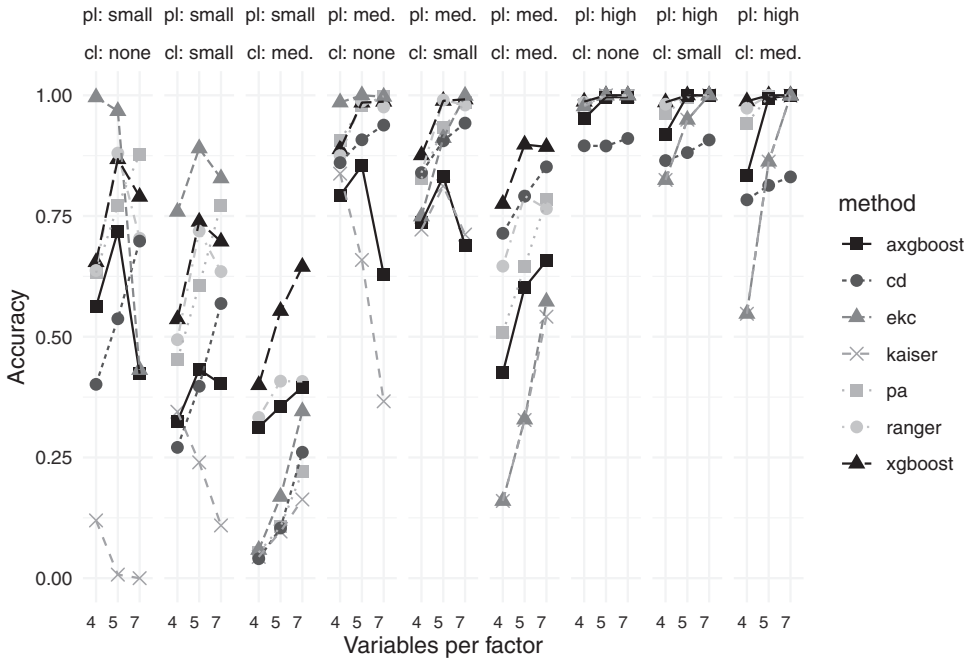

Figure 5. Accuracy of retention criteria for conditions with six factors averaged over *N* and ρ. pa = parallel analysis; cd = comparison data; ekc = empirical Kaiser criterion.

Table 4
Accuracy of the Factor Retention Criteria for Data Conditions With Negative and Positive Loadings (A) and a Data Condition Based on a Random Intercept Model (B)

N	vpf	k	ρ	Primary load.	Cross-load.	Acc_{xgb}	Acc_{ekc}	Acc_{cd}	Acc_{pa}	Acc_{kc}
A										
500	5	2	0.0	High	Medium	1.00	1.00	0.89	1.00	1.00
500	5	2	0.0	High	None	1.00	1.00	0.94	1.00	1.00
500	5	2	0.0	Low	Medium	1.00	0.98	0.98	0.87	0.88
500	5	2	0.0	Low	None	1.00	1.00	0.96	0.81	0.70
500	5	2	0.3	High	Medium	1.00	1.00	0.85	1.00	1.00
500	5	2	0.3	High	None	1.00	1.00	0.93	1.00	1.00
500	5	2	0.3	Low	Medium	1.00	0.95	0.99	0.94	0.80
500	5	2	0.3	Low	None	1.00	1.00	0.98	0.90	0.77
500	5	4	0.0	High	Medium	1.00	1.00	0.95	1.00	1.00
500	5	4	0.0	High	None	1.00	1.00	0.86	1.00	1.00
500	5	4	0.0	Low	Medium	1.00	0.91	0.99	0.99	0.60
500	5	4	0.0	Low	None	1.00	1.00	1.00	0.89	0.01
500	5	4	0.3	High	Medium	1.00	1.00	0.93	1.00	1.00
500	5	4	0.3	High	None	1.00	1.00	0.97	1.00	1.00
500	5	4	0.3	Low	Medium	0.95	0.64	0.97	0.97	0.38
500	5	4	0.3	Low	None	0.98	0.65	0.95	0.90	0.00
B										
500	NA	2	0.0	Equal	Unequal	1.00	0.01	0.96	1.00	0.88

Note. pa = parallel analysis; cd = comparison data; ekc = empirical Kaiser criterion; xgb = xgboost model; kc = Kaiser criterion; Acc = accuracy. vpf = Variables per factor; k = number of factors; A = data conditions with both negative and positive loadings; B = data condition based on random intercept model with all variables loading equally on the first factor and unequally on the second factor.

conditions are included, the machine learning models (the *xgboost* model in particular) are able to outperform all common criteria. So providing a wide-ranging training set allows us to rely on a single model rather than combining several criteria. One specific advantage of this approach is that the data basis can be extended easily and the model can be improved if necessary when specific conditions have to be considered that have been left out previously. Further research should also focus on the evaluation of the model under other data conditions (e.g., extending our additional analyses: conditions with different numbers of variables for each factor, more complex factor structures like the random intercept model or more conditions with negative loadings).

It might also be possible to further improve the model by adding new features (as we did in the tuned version adding the solutions of PA, CD, and EKC as features) and extending the data basis for specific applications (e.g., panel data with far more factors and variables). So far, the data basis is solely based on data following a multivariate normal distribution, yet data in psychological research is often of ordinal nature, so in a next step a model trained on ordinal data has to be developed as well. The procedure can easily be applied to both ordinal data and other somehow exceptionally distributed data (e.g., count data from observational studies). Accordingly, the new approach provides a framework that is less dependent on distributional assumptions than other criteria (e.g., EKC and CD relying on normally distributed items).

Another advantage of this approach is the possibility to get not just an estimate for *k* but also a probability estimate for several values of *k*. With this option, the *xgboost* model provides an implicit uncertainty measure that enables the user (researcher) to assess how convincing a particular solution is. Common retention criteria, on the contrary, only return an estimate for *k* or in case of the scree test an ambiguous plot that has to interpreted,[12] but usually no information reflecting the estimation uncertainty (due to sampling error) is given. Our study showed that the accuracy of all factor retention criteria is influenced by such sampling error as reflected by the comparably poor performance of PA, CD, and the Kaiser criterion when $N = 250$—a sample size that is not necessarily reached in current research (Goretzko et al., 2019).

Table 5
Feature Importance: 15 Most Important Features of the xgboost Model

Feature	Type	Importance
Kolm	Inequality measure	0.248
Gini	Inequality measure	0.103
fa_eigval2	Eigenvalue factor model	0.085
fa_eigval3	Eigenvalue factor model	0.082
fa_eigval8	Eigenvalue factor model	0.081
fa_eigval4	Eigenvalue factor model	0.079
fa_eigval6	Eigenvalue factor model	0.072
fa_eigval7	Eigenvalue factor model	0.071
fa_eigval5	Eigenvalue factor model	0.045
eigval2	Eigenvalue	0.041
eigval3	Eigenvalue	0.011
avgcor	Correlation size	0.010
N	Sample	0.008
fa_eigval9	Eigenvalue factor model	0.006
eigval5	Eigenvalue	0.005

Note. fa_eigval2 means the second eigenvalue of the factor model, while eigval2 is the second eigenvalue of the correlation matrix. avgcor is the averaged inter-item correlation.

[12] Note that there are ways to objectify this procedure, like the Cattell-Nelson-Gorsuch approach (e.g. Nasser, Benson, & Wisenbaker, 2002).

Table 6
Hyperparameter Space for Tuning

Name	Lower	Upper	log2_scale
Eta	0.01	0.20	FALSE
Gamma	−7.00	6.00	TRUE
max_depth	3.00	20.00	FALSE
colsample_bytree	0.50	1.00	FALSE
Lambda	−10.00	10.00	TRUE
Subsample	0.50	1.00	FALSE

Note. log2_scale means that values are transformed according to the binary logarithm. Parameter names are identical to names in the xgboost implementation in R.

Understanding the Black Box

Practitioners might be bothered by the black box character of the model which hampers its interpretability. Hence, one can use tools like the local interpretable model-agnostic explanations (LIME; Ribeiro, Singh, & Guestrin, 2016) for each new empirical data set that is evaluated with the *xgboost* model. We present a short example on how this could work. For this purpose, we chose a data set containing 1,369 observations of 50 items of a *BIG5*-inventory constructed by Goldberg (1990) that can be retrieved from https://openpsychometrics.org/_rawdata/ (version of 11/8/2018). Applying the tuned *xgboost* model to this data yielded six factors (estimated probability for $k = 6$ was 97.3%) while CD suggested five, EKC six, PA eight, and the Kaiser-Guttman rule nine factors. Approximating the complex *xgboost* model locally with (generalized) linear models, LIME provides the best features to explain this six factor solution.[13] Nine out of the 10 most important features explaining the six-factor solution were different eigenvalues with the sixth eigenvalue of the factor model being greater than 0.5115 having the highest explanatory power and $p = 50 > 35$ being third (Supplementary Table 1). There were also several features having negative explanatory power which means that these features and its values would speak against the final prediction of the complex model based on the simple approximation (e.g., the second eigenvalue being greater than 2.4). Approximating the complex model locally using (generalized) linear models provides insights on how the complex model estimates the number of factors *k*. Even though LIME might improve the interpretability of the black box model, researchers have to be cautious as the (local) approximation with simple models might not fully reflect the complex interactions among some features that indicate a specific *k*-factor solution (here: rather weak $R^2 = 0.235$ of the explaining model).

Conclusion

This study shows that the new approach combining extensive data simulation and machine learning techniques to determine the number of factors provides very good results, outperforming common criteria. Based on data that cover a wide range of conditions, the new approach promises to tackle the ambiguous decision of how many factors to extract in EFA. Extending the data basis as well as the features might improve the method even further. Further research could also evaluate other ML algorithms, even though the performance of the tuned *xgboost* model seems to be tough to beat. Adaptation to ordinal data should follow, so that Likert-type data will be specifically accounted for. As the new approach is computationally costly, further research should aim to provide pretrained models for different data types. Accordingly, the tuned *xgboost* model of this study can be obtained from https://osf.io/mvrau/ to determine the number of factors in data conditions that are similar to those in our training set.

[13] Note that the estimated R^2 of the model used for the approximation was 0.235 in this case.

References

Auerswald, M., & Moshagen, M. (2019). How to determine the number of factors to retain in exploratory factor analysis: A comparison of extraction methods under realistic conditions. *Psychological Methods, 24,* 468–491. http://dx.doi.org/10.1037/met0000200

Aust, F., & Barth, M. (2018). *papaja: Create APA manuscripts with R Markdown*. Retrieved from https://github.com/crsh/papaja

Bischl, B., Lang, M., Kotthoff, L., Schiffner, J., Richter, J., Studerus, E., . . . Jones, Z. M. (2016). Mlr: Machine learning in R. *The Journal of Machine Learning Research, 17,* 5938–5942.

Braeken, J., & Van Assen, M. A. (2017). An empirical Kaiser criterion. *Psychological Methods, 22,* 450–466. http://dx.doi.org/10.1037/met0000074

Breiman, L. (1999). *Random forest*. Retrieved from http://machinelearning202.pbworks.com/w/file/fetch/60606349/breiman_randomforests.pdf

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

Browne, M. W. (2001). An overview of analytic rotation in exploratory factor analysis. *Multivariate Behavioral Research, 36,* 111–150. http://dx.doi.org/10.1207/S15327906MBR3601_05

Cattell, R. B. (1966). The scree test for the number of factors. *Multivariate Behavioral Research, 1,* 245–276. http://dx.doi.org/10.1207/s15327906mbr0102_10

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794). New York, NY: ACM. http://dx.doi.org/10.1145/2939672.2939785

Chen, T., He, T., Benesty, M., Khotilovich, V., & Tang, Y. (2018). Xgboost: Extreme gradient boosting (R package version 0.6. 4.1) [Computer software]. Retrieved from https://CRAN.R-project.org/package=xgboost

De Winter, J. C., & Dodou, D. (2012). Factor recovery by principal axis factoring and maximum likelihood factor analysis as a function of factor pattern and sample size. *Journal of Applied Statistics, 39,* 695–710. http://dx.doi.org/10.1080/02664763.2011.610445

Dinno, A. (2009). Exploring the sensitivity of Horn's parallel analysis to the distributional form of random data. *Multivariate Behavioral Research, 44,* 362–388. http://dx.doi.org/10.1080/00273170902938969

Fabrigar, L. R., Wegener, D. T., MacCallum, R. C., & Strahan, E. J. (1999). Evaluating the use of exploratory factor analysis in psychological research. *Psychological Methods, 4,* 272–299. http://dx.doi.org/10.1037/1082-989X.4.3.272

Fava, J. L., & Velicer, W. F. (1996). The effects of underextraction in factor and component analyses. *Educational and Psychological Measurement, 56,* 907–929. http://dx.doi.org/10.1177/0013164496056006001

Friedman, J. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics, 29,* 1189–1232. Retrieved from https://www.jstor.org/stable/2699986

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics, 28,* 337–407. http://dx.doi.org/10.1214/aos/1016218223

Genuer, R., Poggi, J.-M., & Tuleau, C. (2008). Random forests: Some methodological insights. *arXiv Preprint arXiv:0811.3619*. Retrieved from https://arxiv.org/abs/0811.3619

Genz, A., Bretz, F., Miwa, T., Mi, X., Leisch, F., Scheipl, F., & Hothorn, T. (2018). *mvtnorm: Multivariate normal and t distributions*. Retrieved from https://CRAN.R-project.org/package=mvtnorm

Gini, C. (1921). Measurement of inequality of incomes. *The Economic Journal, 31,* 124–126.

Goldberg, L. R. (1990). An alternative "description of personality": The big-five factor structure. *Journal of Personality and Social Psychology, 59,* 1216–1229. http://dx.doi.org/10.1037/0022-3514.59.6.1216

Goretzko, D., Pham, T. T. H., & Bühner, M. (2019). Exploratory factor analysis: Current use, methodological developments and recommendations for good practice. *Current Psychology*. http://dx.doi.org/10.1007/s12144-019-00300-2

Horn, J. L. (1965). A rationale and test for the number of factors in factor analysis. *Psychometrika, 30,* 179–185. http://dx.doi.org/10.1007/BF02289447

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning*. New York, NY: Springer.

Kaiser, H. F. (1960). The application of electronic computers to factor analysis. *Educational and Psychological Measurement, 20,* 141–151. http://dx.doi.org/10.1177/001316446002000116

Kaiser, H. F. (1970). A second generation little jiffy. *Psychometrika, 35,* 401–415. http://dx.doi.org/10.1007/BF02291817

Kolm, S.-C. (1999). The rational foundations of income inequality measurement. In J. Silber (Ed.), *Handbook of income inequality measurement* (pp. 19–100). Dordrecht, the Netherlands: Springer.

Lim, S., & Jahng, S. (2019). Determining the number of factors using parallel analysis and its recent variants. *Psychological Methods, 24,* 452–467. http://dx.doi.org/10.1037/met0000230

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Stützle, T., & Birattari, M. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives, 3,* 43–58. http://dx.doi.org/10.1016/j.orp.2016.09.002

Lorenzo-Seva, U., Timmerman, M. E., & Kiers, H. A. L. (2011). The hull method for selecting the number of common factors. *Multivariate Behavioral Research, 46,* 340–364. http://dx.doi.org/10.1080/00273171.2011.564527

Nasser, F., Benson, J., & Wisenbaker, J. (2002). The performance of regression-based variations of the visual scree for determining the number of common factors. *Educational and Psychological Measurement, 62,* 397–419. http://dx.doi.org/10.1177/00164402062003001

Peres-Neto, P. R., Jackson, D. A., & Somers, K. M. (2005). How many principal components? Stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis, 49,* 974–997.

R Development Core Team. (2008). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from http://www.R-project.org

Revelle, W. (2018). *Psych: Procedures for psychological, psychometric, and personality research*. Evanston, IL: Northwestern University. Retrieved from https://CRAN.R-project.org/package=psych

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). Model-agnostic interpretability of machine learning. *arXiv Preprint arXiv:1606.05386*. Retrieved from https://arxiv.org/abs/1606.05386

Ruscio, J., & Roche, B. (2012). Determining the number of factors to retain in an exploratory factor analysis using comparison data of known factorial structure. *Psychological Assessment, 24,* 282–292. http://dx.doi.org/10.1037/a0025697

Thomas, J., Coors, S., & Bischl, B. (2018). Automatic gradient boosting. *arXiv Preprint arXiv:1807.03873*. Retrieved from https://arxiv.org/abs/1807.03873

Velicer, W. F. (1976). Determining the number of components from the matrix of partial correlations. *Psychometrika, 41,* 321–327. http://dx.doi.org/10.1007/BF02293557

Wickham, H. (2016). *Ggplot2: Elegant graphics for data analysis*. New York, NY: Springer-Verlag. Retrieved from https://ggplot2.tidyverse.org

Wright, M. N., & Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software, 77,* 1–17. http://dx.doi.org/10.18637/jss.v077.i01

Zwick, W. R., & Velicer, W. F. (1986). Comparison of five rules for determining the number of components to retain. *Psychological Bulletin, 99,* 432–442. http://dx.doi.org/10.1037/0033-2909.99.3.432

Received June 10, 2019
Revision received January 22, 2020
Accepted January 23, 2020 ■