Validation of machine learning algorithms

Tomasz Burzykowski,[a,b] Melvin Geubbelmans,[a] Axel-Jan Rousseau,[a] and Dirk Valkenborg[a]
Hasselt, Belgium, and Bialystok, Poland

Machine-learning (ML) algorithms based on statistical models "learn" by building models using a *training dataset*. Several choices are to be made during the model building. First, various algorithms can be considered for solving an ML problem. The structure of an algorithm is also an important consideration, for instance, whether it assumes a linear or a nonlinear relationship. Further, one must decide which explanatory variables should be included in the model. Finally, some algorithms apply the so-called hyperparameters that have to be fine-tuned. An essential issue in the model-building process is the risk of *overfitting*, ie, the risk that the obtained model "follows" a particular training dataset too closely to the extent that it may capture random aspects in the data. In that case, the model may fail to provide reliable predictions for another set of observations because the random aspects may be different or no longer present in the new dataset.

Figure 1 schematically illustrates this issue. The *blue* and *red* curves show the change in prediction error on the basis of the complexity of the model. This complexity is roughly the same as the number of explanatory variables included therein. The *blue* curve presents the error for a training dataset, whereas the *red* curve presents the error for an independent testing dataset. The *blue* curve indicates that prediction error decreases when more complex models are considered for a particular training dataset. Hence, one might be inclined to use a complex model to reduce the error. However, the *red* curve shows that, at some point, increasing the model's complexity will become counterproductive, because it will increase the prediction error for an independent dataset. Unfortunately, it is not generally possible to specify the optimal model-complexity threshold at which an ML model would lead to the minimum prediction error in an independent dataset.

In practice, various *model validation* strategies can be used to prevent overfitting. In an ideal situation, the available data would be split into 3 parts: *training*, *validation*, and *testing datasets*.[1] This strategy is sometimes referred to as a *holdout method*. The training dataset would be used to build models of different forms and/or complexity. Estimates of the prediction error for those models, computed using the training data, would be too optimistic and should not serve for model selection, as suggested by the *blue* curve in Figure 1, which shows a much lower error than the *red* curve of the independent set. Instead, the validation dataset would be used to select the model with the smallest prediction error (in this case sometimes called the *training error*[1]). Finally, the selected model would be applied to the testing dataset that is held out (hence the name of the strategy) to obtain an independent assessment of the prediction error (in this case, sometimes called the *generalization* or *test error*[1]). This model-selection procedure minimizes the test error by the so-called bias-variance trade-off explained in an earlier article.

However, very often, the amount of available data is limited and splitting the data as described above is infeasible. In that case, the validation process can be mimicked by applying *cross-validation* (CV). In the *K-fold CV* (left-hand-side panel, Fig 2), the data are split into K parts of roughly equal size. Each part is considered in turn and set aside to play the role of a testing dataset. The model is then built by considering the data from the remaining K-1 parts as a training dataset and validated on the part set aside for testing. This is repeated for each part set aside. After obtaining the predictions for observations in each of the K parts, the prediction error is estimated by taking the average of the prediction error for all the observations.

When K equals the number of observations, the procedure is called *leave-one-out CV*, as a single observation is set aside each time. However, the computational

[a]Data Science Institute and Center for Statistics, Hasselt University, Hasselt, Belgium.
[b]Department of Biostatistics and Medical Informatics, Medical University of Bialystok, Bialystok, Poland.
This research received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" program.
Address correspondence to: Tomasz Burzykowski, Hasselt University - Data Science Institute, Agoralaan 1, Building D, B-3590 Diepenbeek, Belgium; e-mail, tomasz.burzykowski@uhasselt.be.
Am J Orthod Dentofacial Orthop 2023;164:295-7
0889-5406/$36.00

https://doi.org/10.1016/j.ajodo.2023.05.007

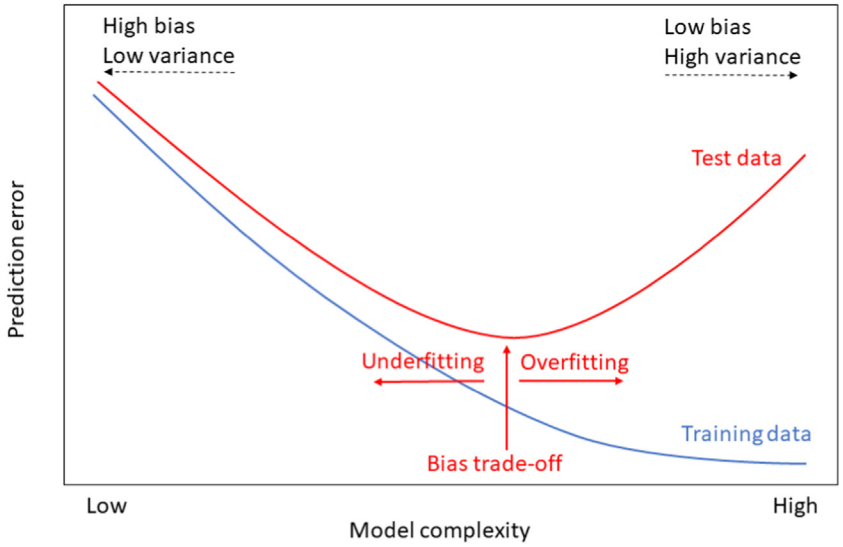

Fig 1. Prediction error in the function of model complexity.

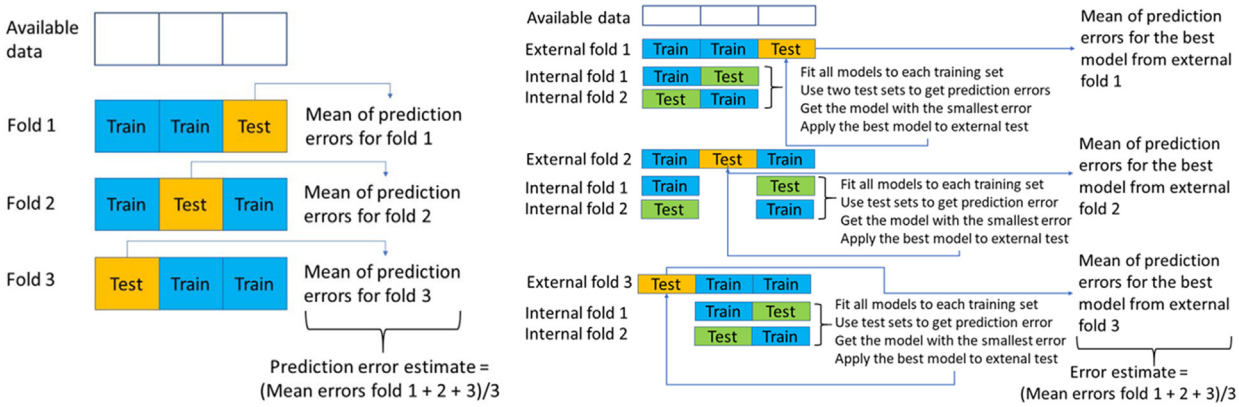

Fig 2. A scheme for 3-fold CV (left panel): in each fold, a model is applied to training data and the mean of prediction errors is obtained for observations in the testing data. The model's prediction error is estimated by the average of the fold-specific mean errors. A scheme for double CV for model selection with external 3-fold CV and internal 2-fold CV (right panel): the internal CV serves to select the best model from several candidates. The external CV estimates the prediction error for the model-selection procedure that can be applied to the available data.

burden may be large in that case, as the model has to be fitted to the data as many times as the number of observations in the original dataset. In addition, the resulting precision of estimation of the prediction error is low. For that reason, in practice, K = 5 or 10 is often used, as it reduces the computational burden and improves precision.

It is worth noting that K-fold CV estimates the prediction error of a selected *model-building procedure* rather than of a fixed model (as in the holdout method). The obtained estimate can be interpreted as an estimate of the true error for the models that we would obtain if we applied the model-building algorithm to the entire dataset.[2]

A vital consequence of this remark is that all steps of the model-building algorithm should be considered in CV. For instance, in some situations, a limited number of explanatory variables is preselected from a large pool of variables before a model is fitted to data. In that case, the feature selection forms a part of the

model-building algorithm and should therefore be included in the CV loop.[1-3] More specifically, the selection of the variables should be applied before fitting the model to the training data specific to a fold. Note that this means that, as a result, different sets of variables may be selected for different folds. These differences should not be seen as an issue with a CV. They reflect the behavior of the model-building algorithm across different datasets and, as indicated above, allow a proper evaluation of the prediction error of the algorithm as a whole.

Similar considerations apply when the model-building algorithm involves selection of a model with the "best" predictive performance from several possible models. *Double CV* (also termed *nested CV*) should be applied in that case.[4] In double CV, there are 2 CV loops: an external K_e-fold CV that is used to evaluate the prediction error of the model-building algorithm and an internal K_i-fold CV (in which K_i can be different from K_e) that is used for the model selection (right-hand-side panel, Fig 2). Note that different "best" models may be selected for different folds of the external K_e-fold CV. These different choices of models reflect the behavior of the model-building algorithm across different datasets and contribute to a proper evaluation of the prediction error.

Even a 3- or 5-fold CV may be problematic for a limited dataset, resulting in small training datasets in each fold. In such a case, an alternative approach to model validation is to use *bootstrap*.[1,3] The idea is to randomly select (*sample*) observations from the original data to create multiple, say B, datasets of the same size as the original one. These datasets are called *bootstrap samples*. Note that several copies of a particular observation can be included in the same sample (*sampling with replacement*). The model is then built for each of the B bootstrap samples, and the prediction error is computed for each observation from the original data. Subsequently, for each observation, the B prediction errors are averaged. Finally, the overall prediction error is estimated by averaging the mean observation-specific prediction errors.

However, an issue is that the original data are used to create the bootstrap samples and to estimate the prediction error in the bootstrap procedure described above. As a result, the obtained estimate of the error is usually too low.[1,3] A better estimate can be obtained by a modified procedure that computes the average prediction errors for each observation from the original data only for those bootstrap samples that do not contain the observation. This is called *leave-one-out bootstrap*.[1,3] An even better estimate comes from computing a weighted average of the prediction error obtained by using the model fitted to the original data and by using the leave-one-out bootstrap. Depending on the applied weights, the so-called *0.632 bootstrap* or *0.632+ bootstrap estimators* are created.[1,3]

To conclude, constructing an optimal model that avoids overfitting is a real issue for ML algorithms, as overfitting may lead to an overly optimistic estimation of the prediction error. Thus, preventing overfitting and properly evaluating the error on an independent dataset is paramount. Toward this aim, CV or bootstrap may be used. The choice of the method may depend on particular characteristics of the problem at hand, such as, for instance, the amount of data available for developing a statistical-model-based ML algorithm.

REFERENCES

1. Hastie T, Tibshirani R, Friedman J. The Elements of Statistical Learning. 2nd ed. New York: Springer; 2009.
2. Varma S, Simon R. Bias in error estimation when using cross-validation for model selection. BMC Bioinformatics 2006;7:91.
3. Ambroise C, McLachlan GJ. Selection bias in gene extraction on the basis of microarray gene-expression data. Proc Natl Acad Sci U S A 2002;99:6562-6.
4. Hawkins DM, Kraker J. Deterministic fallacies and model validation. J Chemometrics 2010;24:188-93.