Assessment, Testing and Comparison of Statistical Models using R

21 March 2021

Summary

A crucial aspect in statistical analysis, particularly with regression models, is to evaluate the quality of fit. During data analysis, researchers should investigate how well models fit to the data to find out whether the best model has been chosen. In the context of presenting results, fit indices should be reported, so that readers can judge the quality of regression models. Functions to create diagnostic plots or to compute fit measures do exist, however, these are located in many different packages, and there is no unique and consistent approach to assess the model quality for different kind of models. This makes it hard for researchers to discover the package they need or to find out whether any packages for specific regression models exist at all.

Aims of the Package

performance is an R-package (R Core Team, 2021) that provides utilities for computing measures to assess model quality, which are not directly provided by R's *base* or *stats* packages. These include measures like R^2, intraclass correlation coefficient, root mean squared error, etc., or functions to check models for overdispersion, singularity or zero-inflation, and more. Functions apply to a large variety of regression models, including generalized linear models, mixed effects models, and Bayesian models.

performance is part of the *easystats* ecosystem, a collaborative project created to facilitate the usage of R for statistical analyses (Ben-Shachar, Makowski, & Lüdecke, 2020; Lüdecke, Ben-Shachar, Patil, Waggoner, & Makowski, 2020; Makowski, Ben-Shachar, & Lüdecke, 2019; Makowski, Ben-Shachar, Patil, & Lüdecke, 2020).

Comparison to other Packages

- *lmtest* (Zeileis & Hothorn, 2002)
- *MuMIn::r.squaredGLMM()* (Barton, 2020)
- *car* (Fox & Weisberg, 2019)
- *broom::glance()* [robinson_broom_2020]

Features

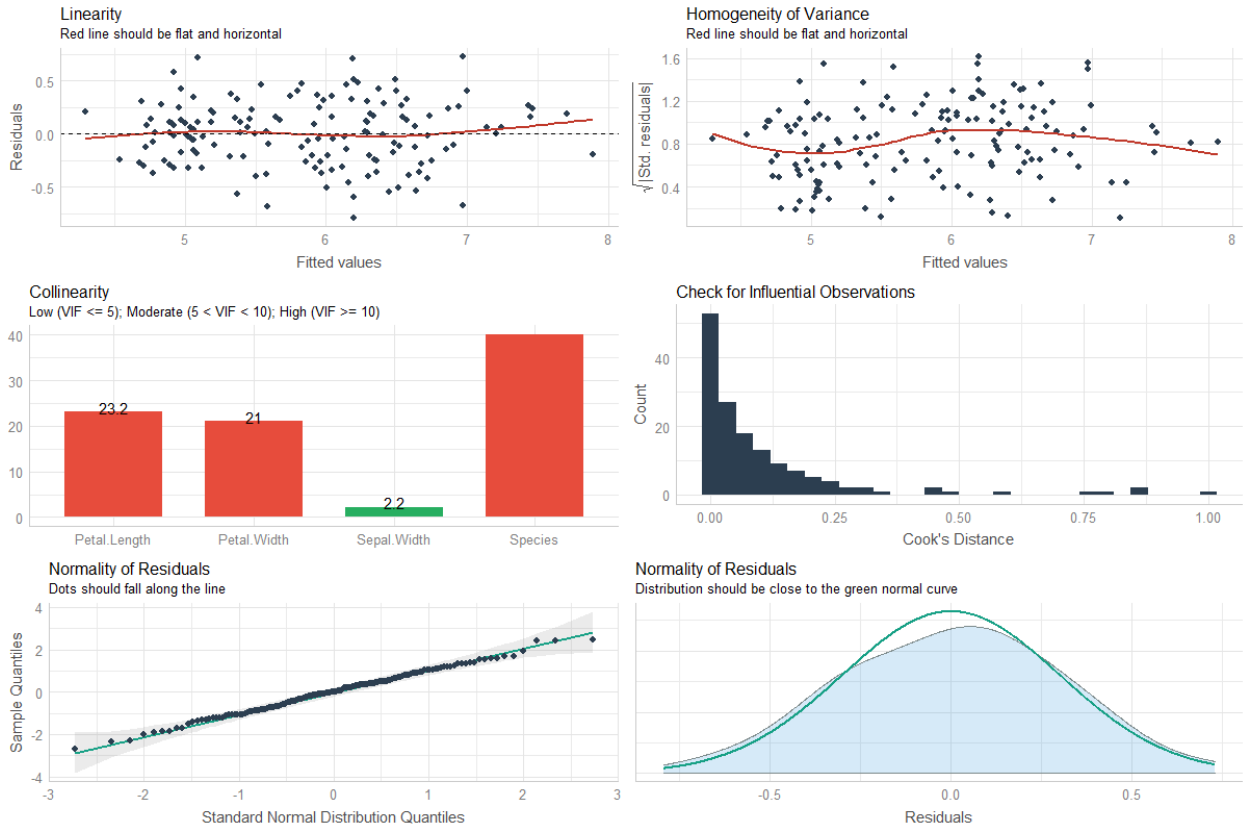
performance functions also include plotting capabilities via the *see* package (Lüdecke, Ben-Shachar, Patil, Waggoner, & Makowski, 2020). A complete overview of plotting functions is available at the *see* website (https://easystats.github.io/see/articles/performance.html).

Checking if a Model is Valid

In addition to providing numerical indices of model fits, *performance* also provides convenience functions to *visually* assess statistical assumptions for regression models. Moreover, these visual checks adjust to the

object entered and support various regression models, like linear models, linear mixed-effects models, their Bayesian equivalents, and more. Here we show what the function output for linear models:

```
library(see)
model <- lm(Sepal.Length ~ Species + Sepal.Width +
            Petal.Length + Petal.Width, data = iris)
check_model(model)
```



Computing Indices of Performance

The `model_performance()` function is the workhorse of this package and allows you to extract a comprehensive set of model fit indices from various models in a consistent way. Depending on the regression model object, the list of computed indices might include R^2, AIC, BIC, RMSE, ICC, LOOIC, etc.

Example with linear model

```
m1 <- lm(mpg ~ wt + cyl, data = mtcars)
model_performance(m1)
#> # Indices of model performance
#>
#> AIC     |     BIC |    R2 | R2 (adj.) |  RMSE | Sigma
#> ----------------------------------------------------------
#> 156.010 | 161.873 | 0.830 |     0.819 | 2.444 | 2.568
```

Example with linear mixed model:

```
library(lme4)
m3 <- lmer(Reaction ~ Days + (1 + Days | Subject), data = sleepstudy)
model_performance(m3)
```

```
#> # Indices of model performance
#>
#> AIC      |      BIC | R2 (cond.) | R2 (marg.) |   ICC |   RMSE |  Sigma
#> ----------------------------------------------------------------------------
#> 1755.628 | 1774.786 |      0.799 |      0.279 | 0.722 | 23.438 | 25.592
```

Comparing Multiple Models

For multiple models, one can obtain a useful table to compare these indices at a glance using the `compare_performance()` function.

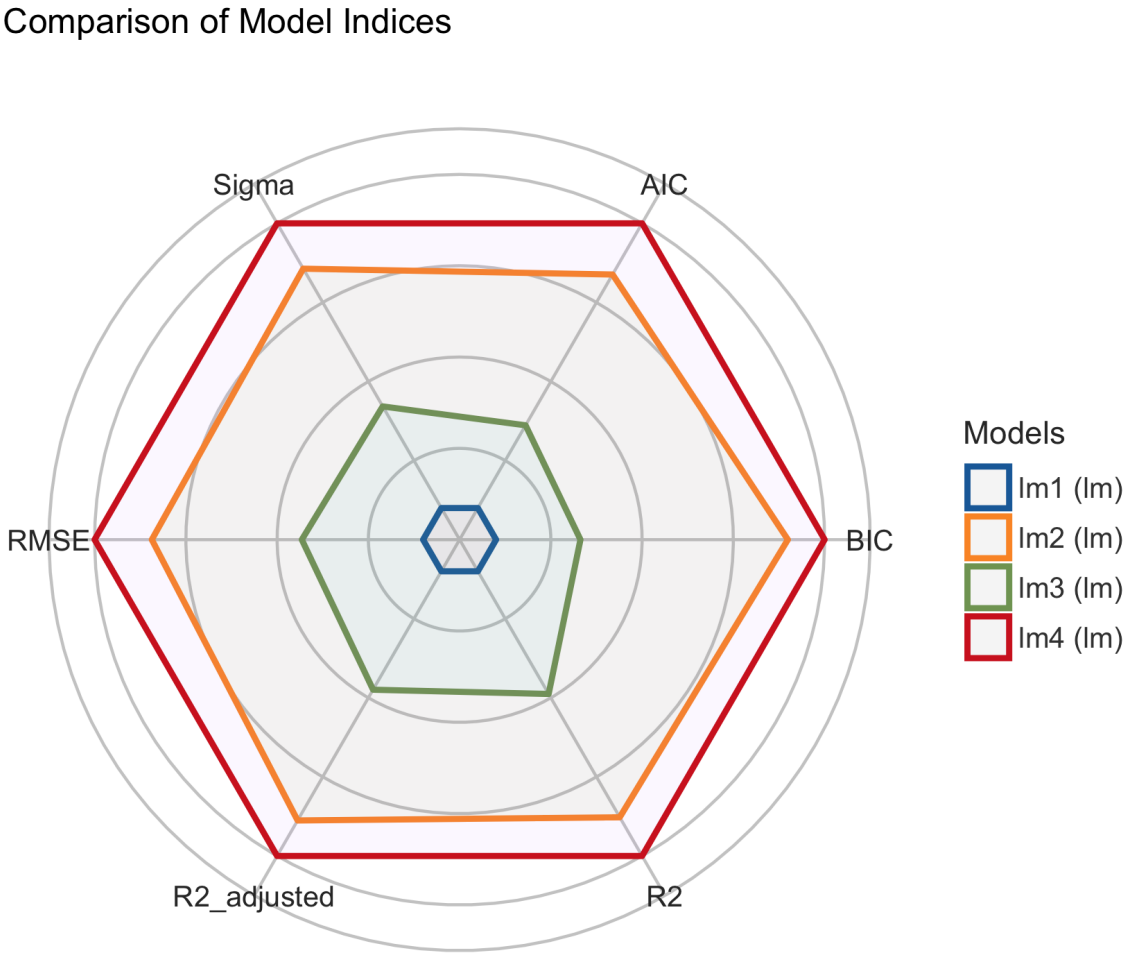
```
data(iris)

lm1 <- lm(Sepal.Length ~ Species, data = iris)
lm2 <- lm(Sepal.Length ~ Species + Petal.Length, data = iris)
lm3 <- lm(Sepal.Length ~ Species * Sepal.Width, data = iris)
lm4 <- lm(Sepal.Length ~ Species * Sepal.Width +
          Petal.Length + Petal.Width, data = iris)

compare_performance(lm1, lm2, lm3, lm4)
#> # Comparison of Model Performance Indices
#>
#> Name | Model |     AIC |     BIC |    R2 | R2 (adj.) |  RMSE | Sigma
#> ------------------------------------------------------------------------
#> lm1  |    lm | 231.452 | 243.494 | 0.619 |     0.614 | 0.510 | 0.515
#> lm2  |    lm | 106.233 | 121.286 | 0.837 |     0.833 | 0.333 | 0.338
#> lm3  |    lm | 187.092 | 208.167 | 0.727 |     0.718 | 0.431 | 0.440
#> lm4  |    lm |  78.797 | 105.892 | 0.871 |     0.865 | 0.296 | 0.305
```

Similarly, in addition to the formal tests to compare several models, *performance* also provides visual ways to compare model fit indices.

```
library(see)
plot(compare_performance(lm1, lm2, lm3, lm4))
```



Testing Models

While **comparing** these indices is often useful, making a decision (for instance, which model to keep or drop) can often be hard, as the indices can give conflicting suggestions. Additionally, it is sometimes unclear which index to favour in the given context.

This is one of the reason why tests are useful, as they facilitate decisions via "significance" indices, like *p*-values (in Frequentist framework) or Bayes Factors (in Bayesian framework).

The generic `test_performance()` runs the most relevant and appropriate tests based on the input. For instance, in the example below, the results from *Vuong's Test* are displayed:

```
test_performance(lm1, lm2, lm3, lm4)
#> Name | Model | Omega2 | p (Omega2) |    LR | p (LR)
#> ------------------------------------------------
#> lm1  |    lm |        |            |       |
#> lm2  |    lm |   0.69 |     < .001 | -6.25 | < .001
#> lm3  |    lm |   0.36 |     < .001 | -3.44 | < .001
#> lm4  |    lm |   0.73 |     < .001 | -7.77 | < .001
#> Each model is compared to lm1.
```

For Bayesian framework, *performance* also provides `test_bf()` function to compare models.

An overview of different test functions is available here.

Licensing and Availability

performance is licensed under the GNU General Public License (v3.0), with all source code stored at GitHub (https://github.com/easystats/performance), and with a corresponding issue tracker for bug reporting and feature enhancements. In the spirit of honest and open science, we encourage requests/tips for fixes, feature updates, as well as general questions and concerns via direct interaction with contributors and developers.

Acknowledgments

performance is part of the collaborative *easystats* ecosystem. Thus, we would like to thank the members of easystats as well as the users.

References

Barton, K. (2020). *MuMIn: Multi-model inference*. Retrieved from https://CRAN.R-project.org/package=MuMIn

Ben-Shachar, M. S., Makowski, D., & Lüdecke, D. (2020). effectsize: Compute and interpret indices of effect size. *CRAN*. https://doi.org/10.5281/zenodo.3952214

Fox, J., & Weisberg, S. (2019). *An R companion to applied regression* (Third). Retrieved from https://socialsciences.mcmaster.ca/jfox/Books/Companion/

Lüdecke, D., Ben-Shachar, M. S., Patil, I., Waggoner, P., & Makowski, D. (2020). see: Visualisation toolbox for 'easystats' and extra geoms, themes and color palettes for 'ggplot2'. *CRAN*. https://doi.org/10.5281/zenodo.3952153

Makowski, D., Ben-Shachar, M., & Lüdecke, D. (2019). bayestestR: Describing effects and their uncertainty, existence and significance within the bayesian framework. *Journal of Open Source Software*, *4*(40), 1541. https://doi.org/10.21105/joss.01541

Makowski, D., Ben-Shachar, M. S., Patil, I., & Lüdecke, D. (2020). Methods and algorithms for correlation analysis in r. *Journal of Open Source Software*, *5*(51), 2306. https://doi.org/10.21105/joss.02306

R Core Team. (2021). *R: A language and environment for statistical computing*. Retrieved from https://www.R-project.org/

Zeileis, A., & Hothorn, T. (2002). Diagnostic checking in regression relationships. *R News*, *2*(3), 7–10. Retrieved from https://CRAN.R-project.org/doc/Rnews/