mlr3shiny: A graphical user interface for easy machine learning in R

31 January 2020

Introduction and Statement of Need

Within recent years, machine learning (ML) applications have entered many industries and open source software is readily available such as the powerful object-orientated and standardized ML framework mlr3 [@mlr3] an evolution of mlr `[@mlr]` dating back to 2010 `[@ifcs]` and based on R `[@rcore]`. Despite the easy acces to frameworks, their advanced and highly parameterizable functionalities may quickly overwhelm the soaring number of inexperienced newcomers to machine learning.

For this reason, the R package `mlr3shiny` has been developed as a simple accessible and user-friendly web-application, combining the graphical user interface (GUI) offered by Shiny `[@shiny]` with the state-of-the-art ML functionalities provided by mlr3. The resulting application enables users to set up machine learning workflows in a very fast way while familiarizing with the basic steps of a machine learning process. Thus, modern ML functionalites can be referenced and applied in an easy to use point-and-click fashion freeing users from coding in R. Especially the latter makes this new package also a valuable tool to be used for teaching introductory ML courses as it is done e.g. at Stralsund University of Applied Sciences (Germany). Working with `mlr3shiny` facilitates the next step to leverage the entire power of mlr3.

Description of Features

The layout of the application visually guides its users chronologically through the different steps of the machine learning workflow as presented in Figure 1.

Figure 2 demonstrates the GUI of the application which can be started via:

```
mlr3shiny::launchMlr3Shiny()
```

The ordered tabs of the application represent the core functionalities of `mlr3shiny` corresponding to the different steps of the workflow as given in Figure 1. A typical ML process starts as follows:

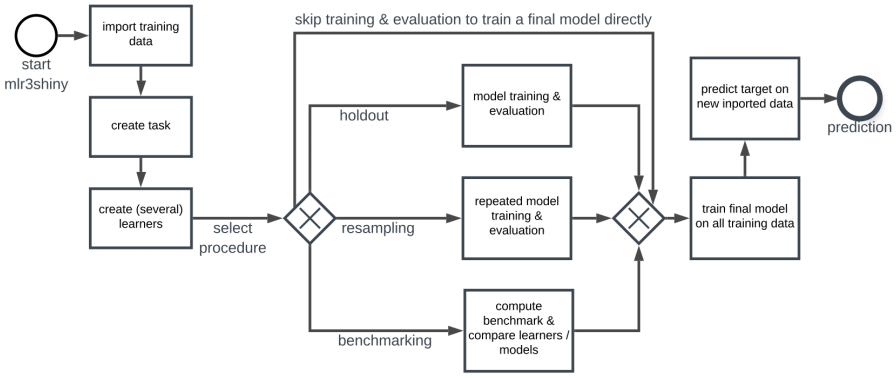

Figure 1: Workflow with mlr3shiny

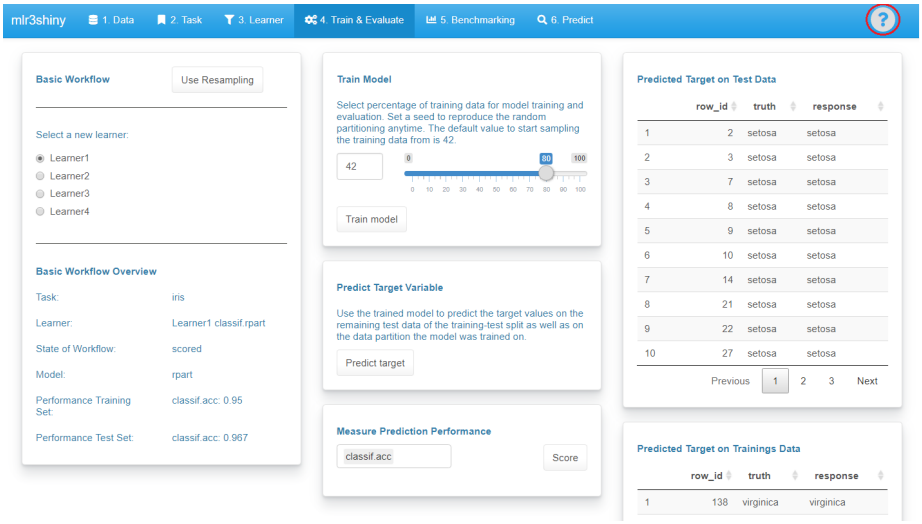

Figure 2: The GUI of `mlr3shiny`

- **Data** import
- Creation of a supervised learning **task** which combines the data with required meta information such as the target variable and the type of task to be performed (either regression or classification).
- Selection of the desired set of **learners** (algorithms) to be used in the ML experiment. Currently supported are linear and logistic regression, decicion trees `[@rpart]`, random forests `[@ranger]` and support vector machines `[@e1071]`. Each learner can be parameterized. In contrast to other GUI based Ml frameworks such as rattle `[@rattle]`, several instances of the same learner class can be defined in order to compare their performance.

The next three tabs correspond to the different branches in the workflow diagram:

- Typically, a single learner out of the specified ones is **trained and evaluated** on the holdout sample with the holdout sample being a randomly selected portion of the data. Figure 2 depicts this step of the modelling process for a classification task. Via *use resampling* resampling techniques such as cross-validation or bootstrap can be used. A trained model can be further applied to both training and holdout data (predict) and further evaluated (score) on both data sets w.r.t some performance metric. At the moment of writing the metrics classification error (ce) and accuracy (ac) are supported for classification tasks and mean squared error (mse) as well as mean absolute error (mae) for regression tasks.
- Several learners can be **benchmarked** and compared against each other w.r.t. a selected performance measure and one of the previuosly mentioned resampling strategies.
- Finally, a model can be applied to **predict** new data. For an optimal performance, it is meaningful to first retrain the learner on the entire data that have been used so far while the former two steps rather serve to evaluate a model with its parameters and assess its predictive performance.

The whole application is designed in such a way that also users new to this field can successfully conduct machine learning experiments. All steps are accompagnied by a help page (red circled question mark in the top right corner) which provides a brief general *description* of the methodology as well as the *functionality* for each step. As an important feature, results can be saved for further use: either benchmark results, a final retrained model on the entire data or the predictions on new data. To guard new users from committing easy mistakes and to provide orientation, default input values and examples are given in accordance with current practices. On top, it is ensured that conducted experiments are replicable.

Summary

In summary, the R package `mlr3shiny` provides a user-friendly web-application that implements the basic steps of a machine learning workflow. As such the access to ML for users unfamiliar with R or coding is facilitated and an easy

entry into the universe of machine learning is provided based on the workflow of one of the state-of-the-art ML frameworks.

Acknowledgements

We greatly acknowledge Michel Lang and the mlr3 developer team for helpful discussions and feedback on the design and the architecure of the application. We would further like to acknowledge Rabea Aschenbruck for her constructivism and Stralsund University of Applied Sciences for providing the creative environment that led to this development.

References