2nd International Conference
on Automated Machine Learning:
AutoML 2023 Conference, Potsdam

forester: A Novel Approach to Accessible and Interpretable AutoML for Tree-Based Modeling

Hubert Ruczyński, Anna Kozak
Warsaw University of Technology

forester: A Novel Approach to Accessible and Interpretable AutoML for Tree-Based Modeling

Anna Kozak[1] **Hubert Ruczyński**[1]

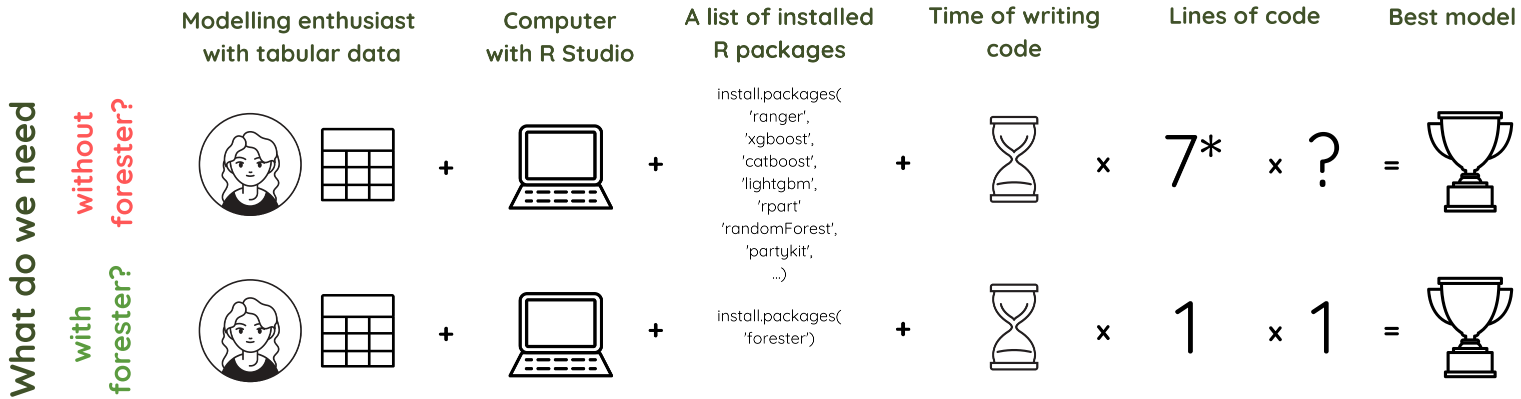
[1]Warsaw University of Technology

Abstract The majority of AutoML solutions are developed in Python. However, a large percentage of data scientists are associated with the R language. Unfortunately, there are limited R solutions available with high entry level which means they are not accessible to everyone. To fill this gap, we present the *forester* package, which offers ease of use regardless of the user's proficiency in the area of machine learning.

The *forester* package is an open-source AutoML package implemented in R designed for training high-quality tree-based models on tabular data. It supports regression and binary classification tasks. A single line of code allows the use of unprocessed datasets, informs about potential issues concerning them, and handles feature engineering automatically. Moreover, hyperparameter tuning is performed by Bayesian optimization, which provides high-quality outcomes. The results are later served as a ranked list of models. Finally, the *forester* package offers a vast training report, including the ranked list, a comparison of trained models, and explanations for the best one.

AutoML 2023 Workshop Track

How to build models in R?



Modelling enthusiast with tabular data

Computer with R Studio

A list of installed R packages

Time of writing code

Lines of code

Best model

What do we need

without forester?

```
install.packages(
'ranger',
'xgboost',
'catboost',
'lightgbm',
'rpart'
'randomForest',
'partykit',
...)
```

+ + + × 7* × ? =

with forester?

```
install.packages(
'forester')
```

+ + + × 1 × 1 =

* dependent on the number of packages used

How to use it?

```
library(forester)
data(`lisbon`)
train_output <- train(lisbon,`Price`)
```

What is the forester?

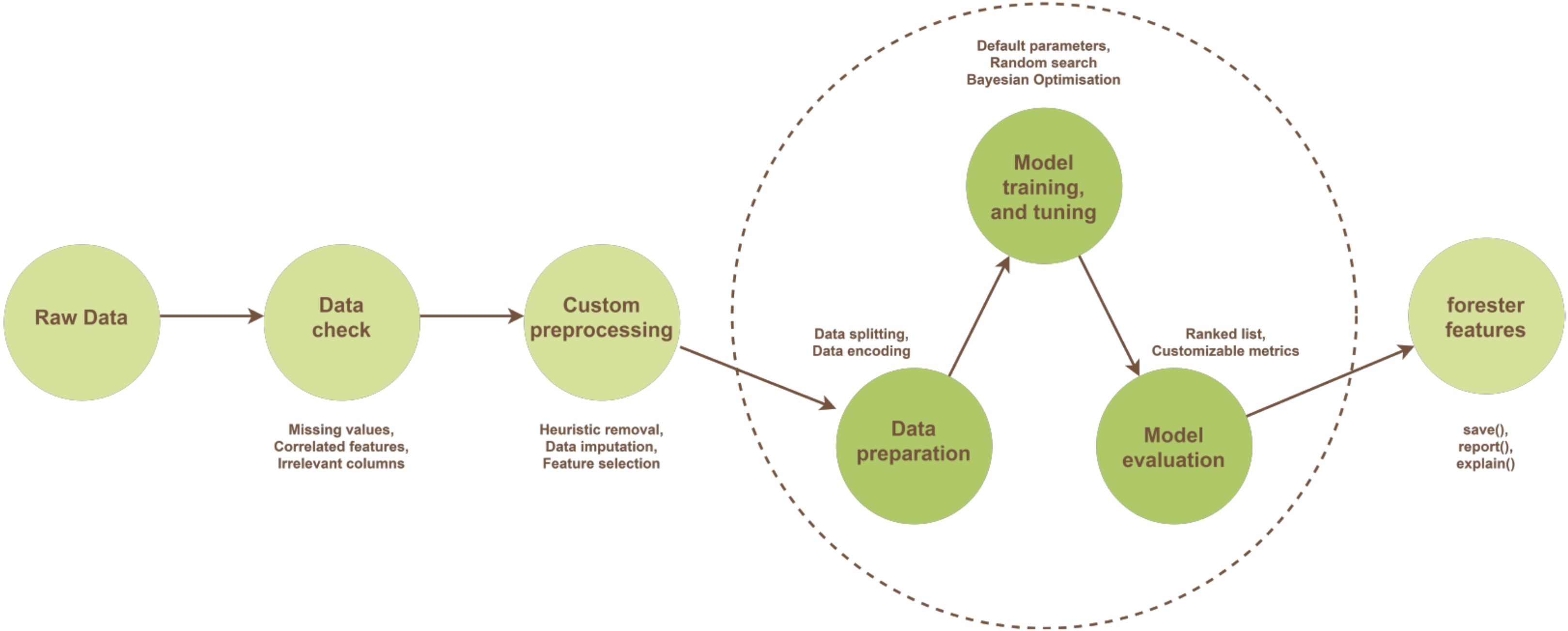
The forester is an AutoML tool in R for **tabular data regression** and **binary classification tasks***, that wraps up all machine learning processes into a single `train()` function, which includes:

1. rendering a brief **data check report**,
2. **preprocessing** the initial dataset enough for models to be trained,
3. **training 5 tree-based models** with default parameters, random search and Bayesian optimization,
4. evaluating them and providing a **ranked list**.

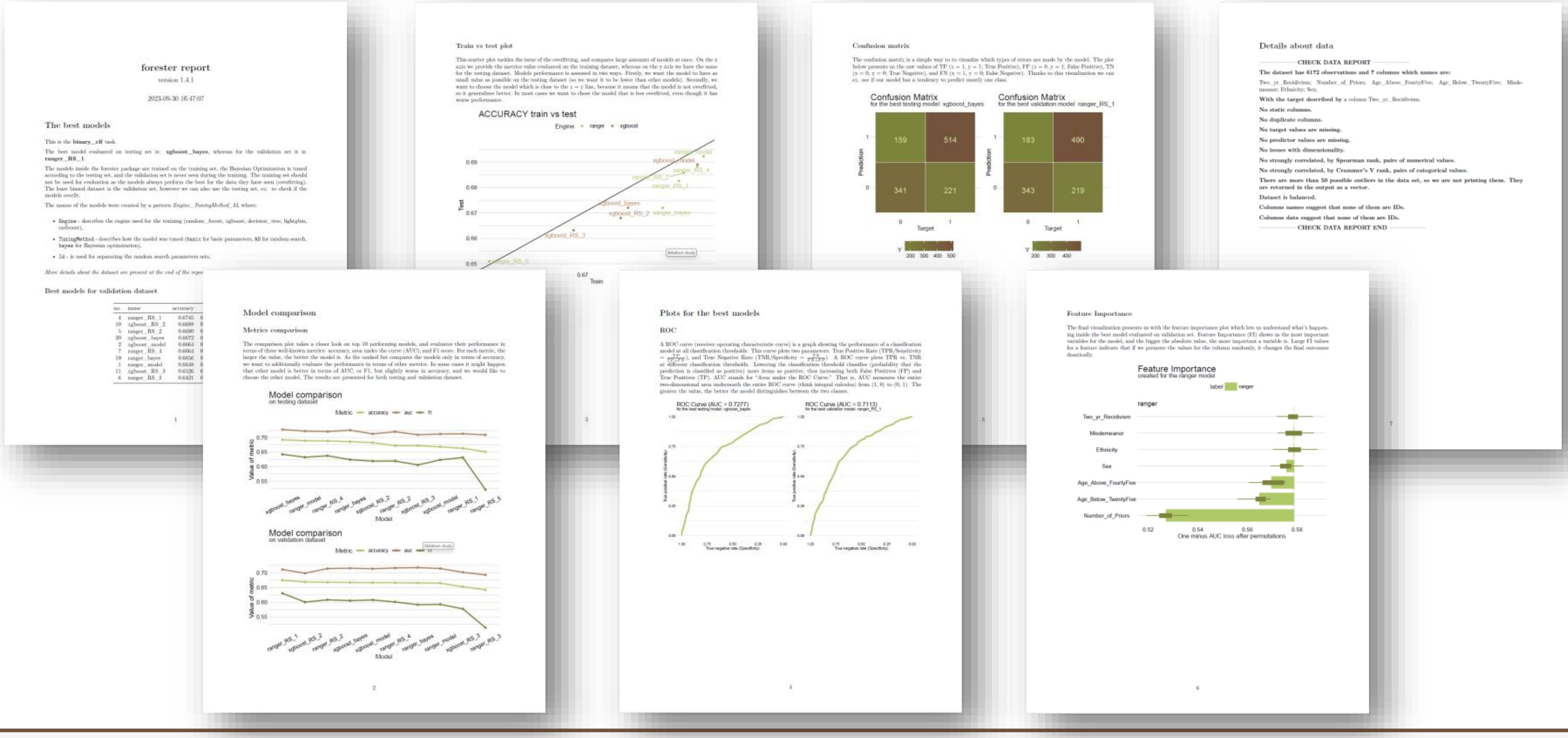
However, that's not everything that the forester has to offer. Via additional functions, the user can easily explain created models with the usage of *DALEX* or generate one of the predefined **reports** including:

1. information about the dataset,
2. in-depth parameters of trained models,
3. visualizations comparing the best models,
4. explanations of the aforementioned models.

forester pipeline



Automatic reports and XAI



forester report

version 1.4.1

2023-09-30 16:47:07

The best models

This is the **binary_clf** task.

The best model evaluated on testing set is: **xgboost_bayes**, whereas for the validation set it is: **ranger_RS_1**.

The models inside the forester package are trained on the training set, the Bayesian Optimization is tuned according to the testing set, and the validation set is never seen during the training. The training set should not be used for evaluation as the models always perform the best for the data they have seen (overfitting). The least biased dataset is the validation set, however we can also use the testing set, ex. to check if the models overfit.

The names of the models were created by a pattern *Engine_TuningMethod_Id*, where:

- `Engine` - describes the engine used for the training (random_forest, xgboost, decision_tree, lightgbm, catboost),
- `TuningMethod` - describes how the model was tuned (`basic` for basic parameters, `RS` for random search, `bayes` for Bayesian optimization),
- `Id` - is used for separating the random search parameters sets.

More details about the dataset are present at the end of the repor

Best models for validation dataset

no.	name	accuracy
4	ranger_RS_1	0.6745
10	xgboost_RS_2	0.6688
5	ranger_RS_2	0.6680
20	xgboost_bayes	0.6672
2	xgboost_model	0.6664
7	ranger_RS_4	0.6664
19	ranger_bayes	0.6656
1	ranger_model	0.6648
11	xgboost_RS_3	0.6526
6	ranger_RS_3	0.6421

Model comparison

Metrics comparison

The comparison plot takes a closer look on top 10 performing models, and evaluates their performance in terms of three well-known metrics: accuracy, area under the curve (AUC), and F1 score. For each metric, the larger the value, the better the model is. As the ranked list compares the models only in terms of accuracy, we want to additionally evaluate the performance in terms of other metrics. In some cases it might happen that other model is better in terms of AUC, or F1, but slightly worse in accuracy, and we would like to choose the other model. The results are presented for both testing and validation dataset.

Train vs test plot

This scatter plot tackles the issue of the overfitting, and compares large amounts of models at once. On the x axis we provide the metrics value evaluated on the training dataset, whereas on the y axis we have the same for the testing dataset. Models performance is assessed in two ways. Firstly, we want the model to have as small value as possible on the testing dataset (so we want it to be lower than other models). Secondly, we want to choose the model which is close to the x = y line, because it means that the model is not overfitted, so it generalizes better. In most cases we want to chose the model that is less overfitted, even though it has worse performance.

Plots for the best models

ROC

A ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate (TPR/Sensitivity $= \frac{TP}{TP+FN}$), and True Negative Rate (TNR/Specificity $= \frac{TN}{TN+FP}$). A ROC curve plots TPR vs. TNR at different classification thresholds. Lowering the classification threshold classifies (probability that the prediction is classified as positive) more items as positive, thus increasing both False Positives (FP) and True Positives (TP). AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (1, 0) to (0, 1). The greater the value, the better the model distinguishes between the two classes.

ROC Curve (AUC = 0.7277)
for the best testing model: xgboost_bayes
ROC Curve (AUC = 0.7113)
for the best validation model: ranger_RS_1
1.00
0.75
0.50
0.25
0.00
True positive rate (Sensitivity)
1.00
0.75
0.50
0.25
0.00
True negative rate (Specificity)

Confusion matrix

The confusion matrix is a simple way to to visualize which types of errors are made by the model. The plot below presents us the raw values of TP (x = 1, y = 1; True Positive), FP (x = 0, y = 1; False Positive), TN (x = 0, y = 0; True Negative), and FN (x = 1, y = 0; False Negative). Thanks to this visualization we can ex. see if our model has a tendency to predict mostly one class.

Feature Importance

The final visualization presents us with the feature importance plot which lets us understand what's happening inside the best model evaluated on validation set. Feature Importance (FI) shows us the most important variables for the model, and the bigger the absolute value, the more important a variable is. Large FI values for a feature indicate that if we permute the values for the column randomly, it changes the final outcomes drastically.

Details about data

——— **CHECK DATA REPORT** ———

The dataset has 6172 observations and 7 columns which names are:

Two_yr_Recidivism; Number_of_Priors; Age_Above_FourtyFive; Age_Below_TwentyFive; Misdemeanor; Ethnicity; Sex;

With the target described by a column Two_yr_Recidivism.

No static columns.

No duplicate columns.

No target values are missing.

No predictor values are missing.

No issues with dimensionality.

No strongly correlated, by Spearman rank, pairs of numerical values.

No strongly correlated, by Crammer's V rank, pairs of categorical values.

There are more than 50 possible outliers in the data set, so we are not printing them. They are returned in the output as a vector.

Dataset is balanced.

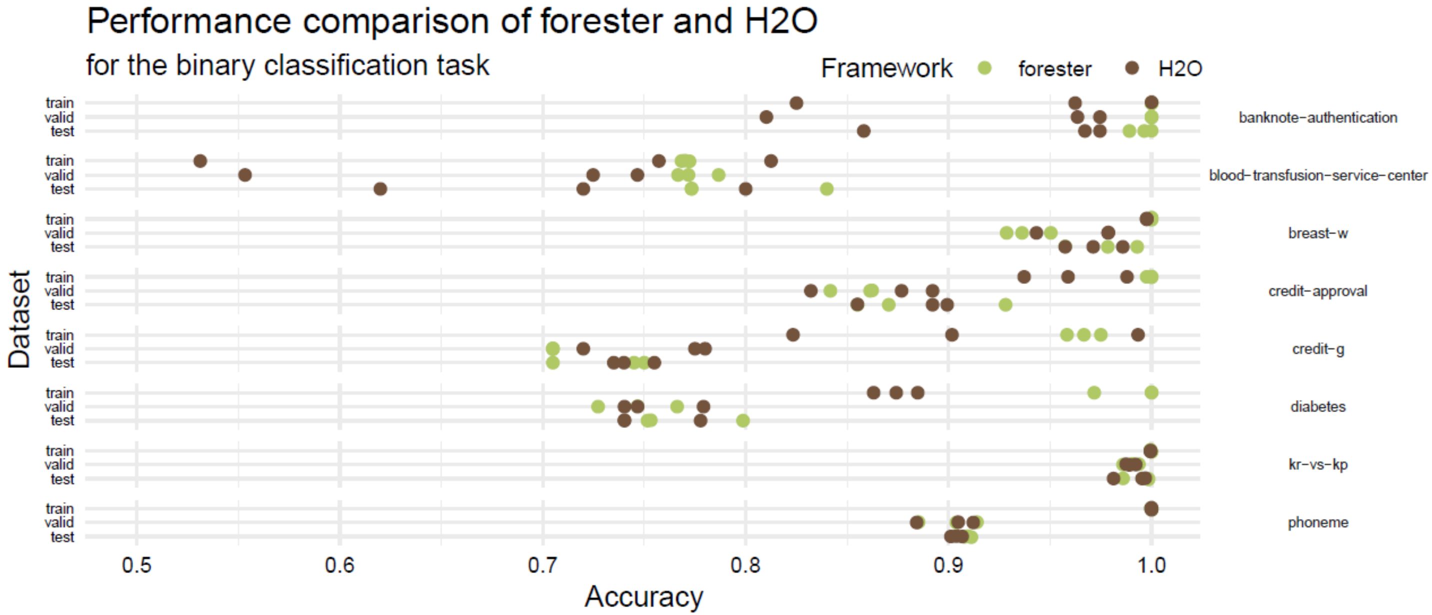
Columns names suggest that none of them are IDs.

Columns data suggest that none of them are IDs.

——— **CHECK DATA REPORT END** ———

Evaluation

Name	Number of columns	Number of rows
kr-vs-kp	37	3196
breast-w	10	699
credit-approval	16	690
credit-g	21	1000
diabetes	9	768
phoneme	6	5404
banknote-authentication	5	1372
blood-transfusion-service-center	5	748

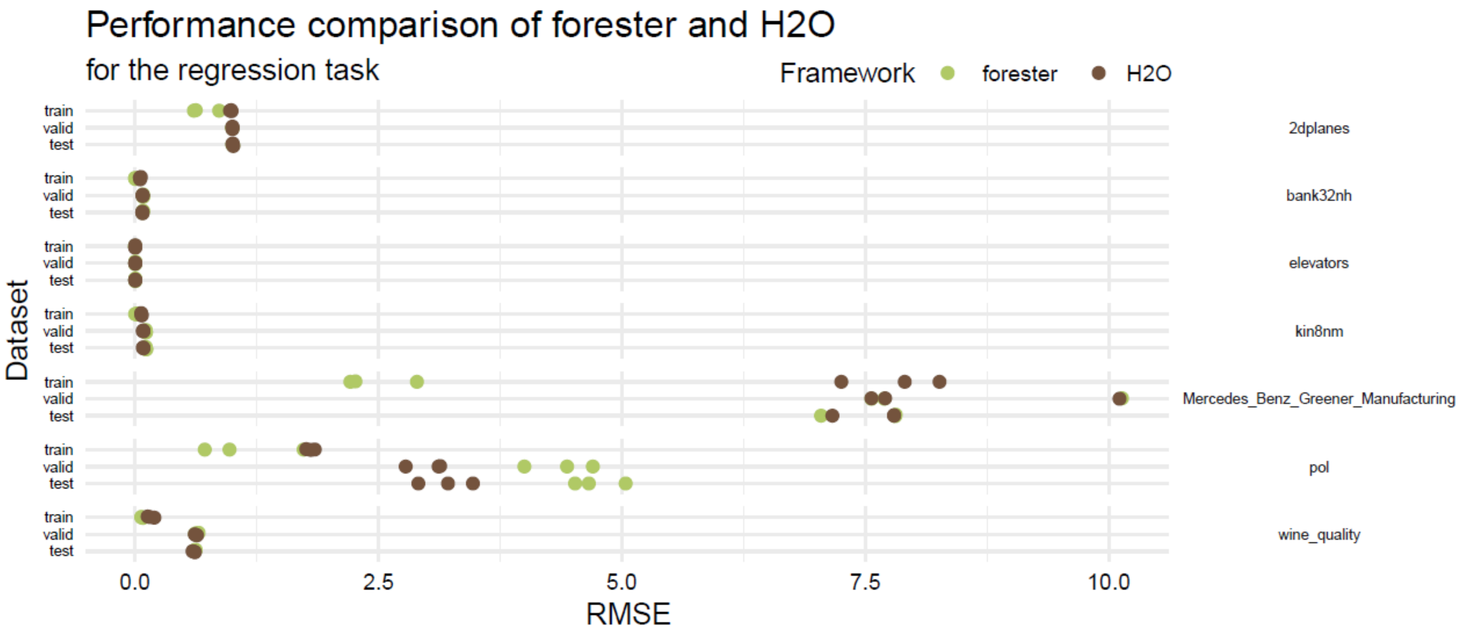


Performance comparison of forester and H2O

for the binary classification task

Evaluation

Name	Number of columns	Number of rows
bank32nh	33	8192
wine_quality	12	6497
Mercedes_Benz_Greener_Manufacturing	378	4209
kin8nm	9	8192
pol	49	15000
2dplanes	11	40768
elevators	19	16599


Performance comparison of forester and H2O

for the regression task

Evaluation

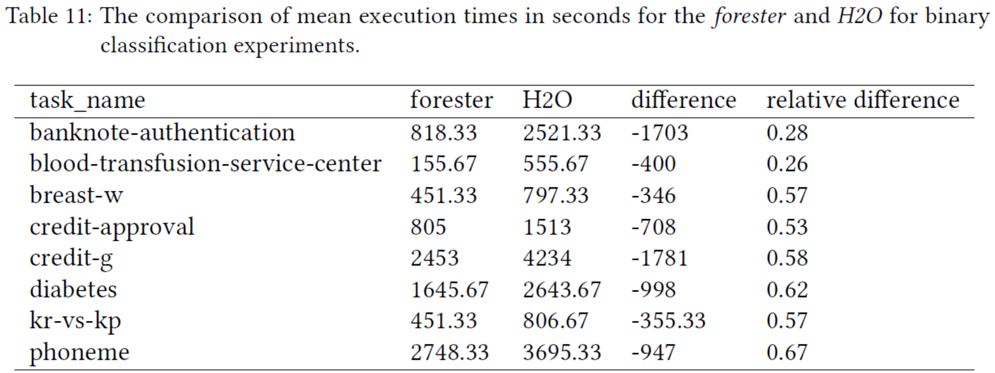


Table 11: The comparison of mean execution times in seconds for the *forester* and *H2O* for binary classification experiments.

task_name	forester	H2O	difference	relative difference
banknote-authentication	818.33	2521.33	-1703	0.28
blood-transfusion-service-center	155.67	555.67	-400	0.26
breast-w	451.33	797.33	-346	0.57
credit-approval	805	1513	-708	0.53
credit-g	2453	4234	-1781	0.58
diabetes	1645.67	2643.67	-998	0.62
kr-vs-kp	451.33	806.67	-355.33	0.57
phoneme	2748.33	3695.33	-947	0.67

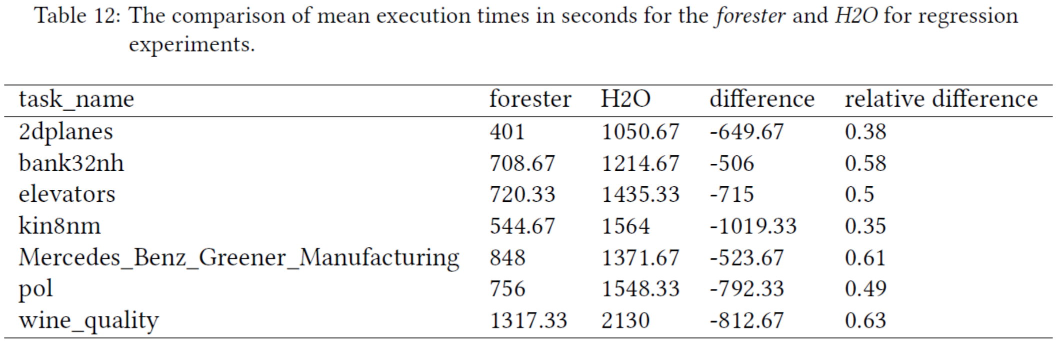
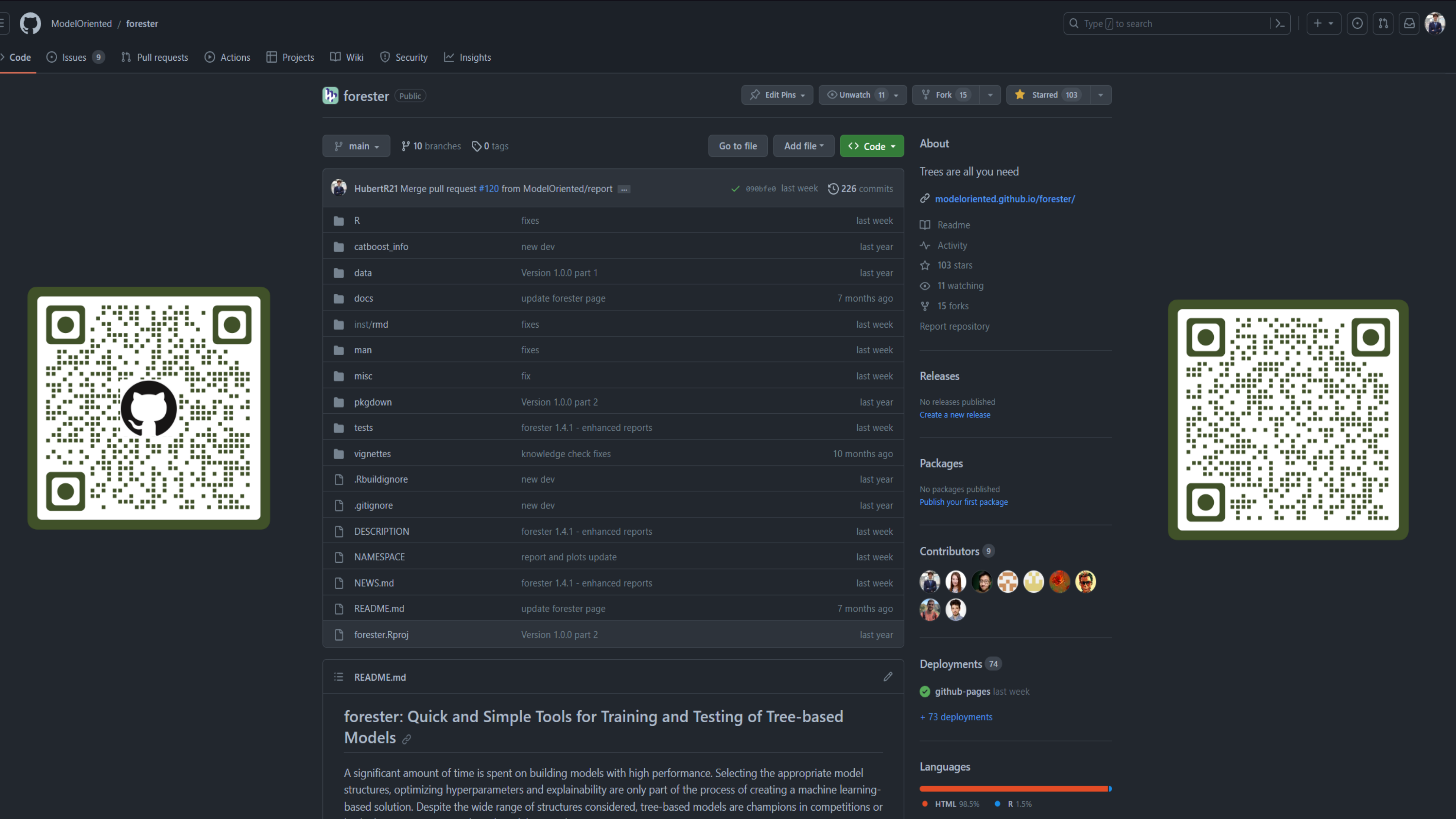


Table 12: The comparison of mean execution times in seconds for the *forester* and *H2O* for regression experiments.

task_name	forester	H2O	difference	relative difference
2dplanes	401	1050.67	-649.67	0.38
bank32nh	708.67	1214.67	-506	0.58
elevators	720.33	1435.33	-715	0.5
kin8nm	544.67	1564	-1019.33	0.35
Mercedes_Benz_Greener_Manufacturing	848	1371.67	-523.67	0.61
pol	756	1548.33	-792.33	0.49
wine_quality	1317.33	2130	-812.67	0.63



HubertR21 Merge pull request #120 from ModelOriented/report ... ✓ 090bfe0 last week 226 commits

R	fixes	last week
catboost_info	new dev	last year
data	Version 1.0.0 part 1	last year
docs	update forester page	7 months ago
inst/rmd	fixes	last week
man	fixes	last week
misc	fix	last week
pkgdown	Version 1.0.0 part 2	last year
tests	forester 1.4.1 - enhanced reports	last week
vignettes	knowledge check fixes	10 months ago
.Rbuildignore	new dev	last year
.gitignore	new dev	last year
DESCRIPTION	forester 1.4.1 - enhanced reports	last week
NAMESPACE	report and plots update	last week
NEWS.md	forester 1.4.1 - enhanced reports	last week
README.md	update forester page	7 months ago
forester.Rproj	Version 1.0.0 part 2	last year

README.md

forester: Quick and Simple Tools for Training and Testing of Tree-based Models

A significant amount of time is spent on building models with high performance. Selecting the appropriate model structures, optimizing hyperparameters and explainability are only part of the process of creating a machine learning-based solution. Despite the wide range of structures considered, tree-based models are champions in competitions or

About

Trees are all you need

modeloriented.github.io/forester/

Readme
Activity
103 stars
11 watching
15 forks
Report repository

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 9

Deployments 74

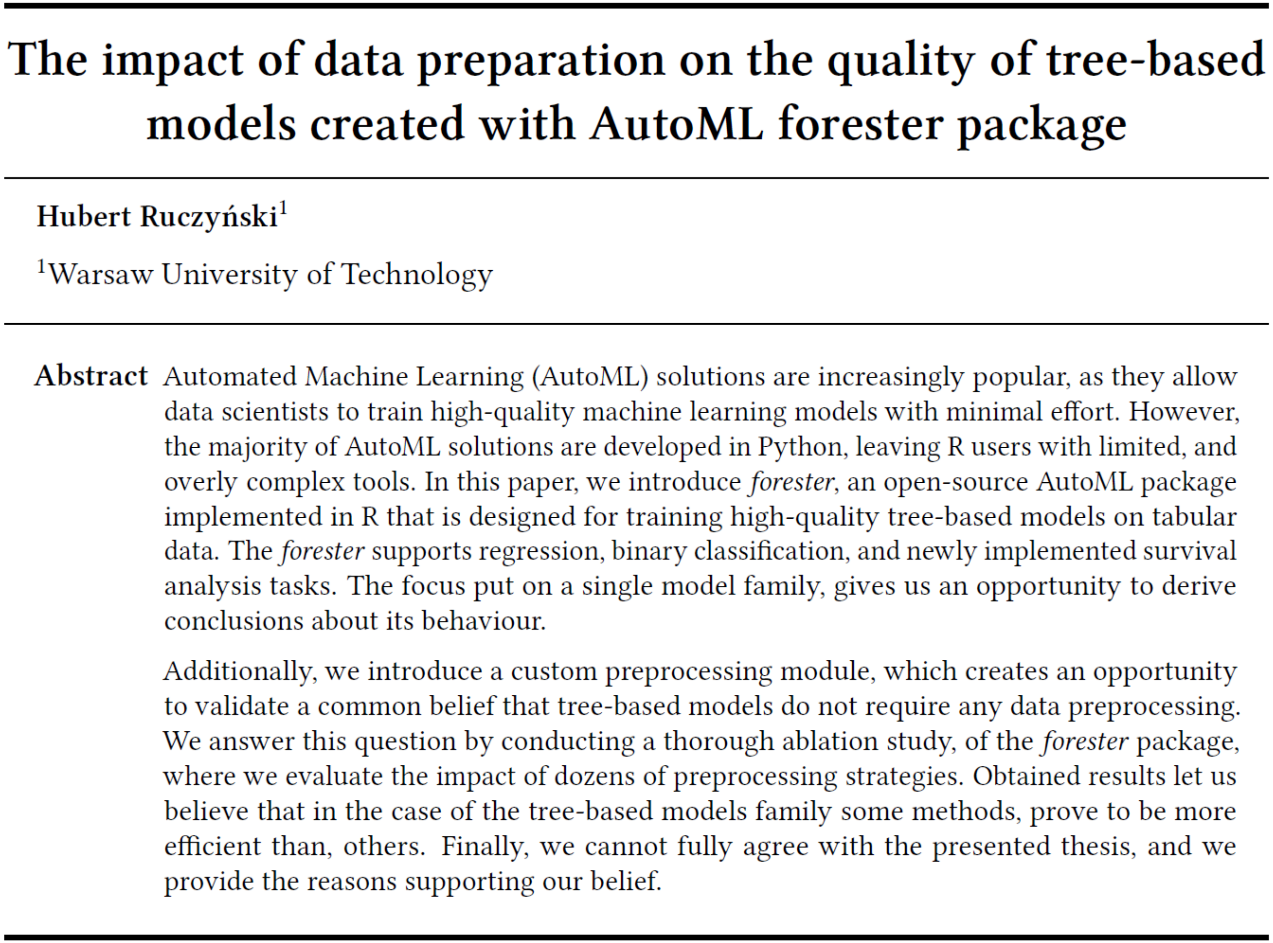
✓ github-pages last week

+ 73 deployments

Languages

● HTML 98.5% ● R 1.5%

Current directions



The impact of data preparation on the quality of tree-based models created with AutoML forester package

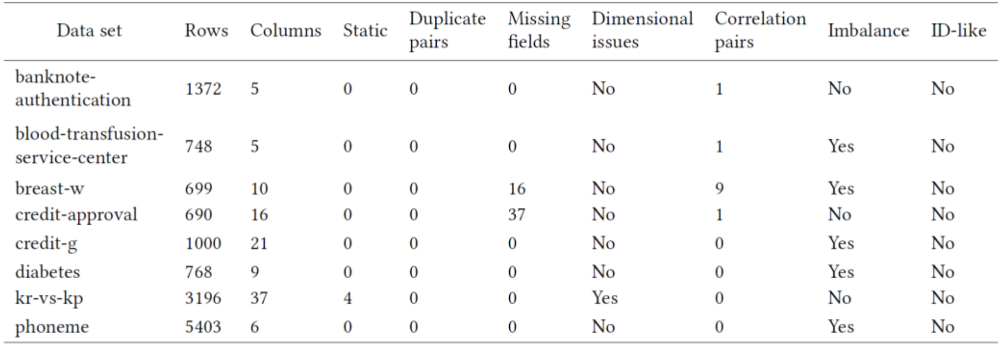
Hubert Ruczyński[1]

[1]Warsaw University of Technology

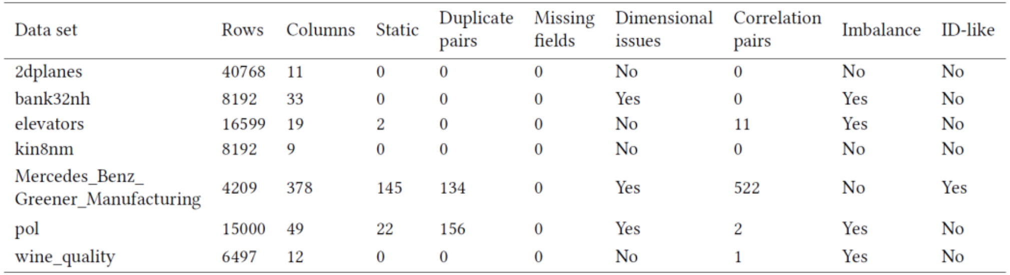
Abstract Automated Machine Learning (AutoML) solutions are increasingly popular, as they allow data scientists to train high-quality machine learning models with minimal effort. However, the majority of AutoML solutions are developed in Python, leaving R users with limited, and overly complex tools. In this paper, we introduce *forester*, an open-source AutoML package implemented in R that is designed for training high-quality tree-based models on tabular data. The *forester* supports regression, binary classification, and newly implemented survival analysis tasks. The focus put on a single model family, gives us an opportunity to derive conclusions about its behaviour.

Additionally, we introduce a custom preprocessing module, which creates an opportunity to validate a common belief that tree-based models do not require any data preprocessing. We answer this question by conducting a thorough ablation study, of the *forester* package, where we evaluate the impact of dozens of preprocessing strategies. Obtained results let us believe that in the case of the tree-based models family some methods, prove to be more efficient than, others. Finally, we cannot fully agree with the presented thesis, and we provide the reasons supporting our belief.

Ablation study

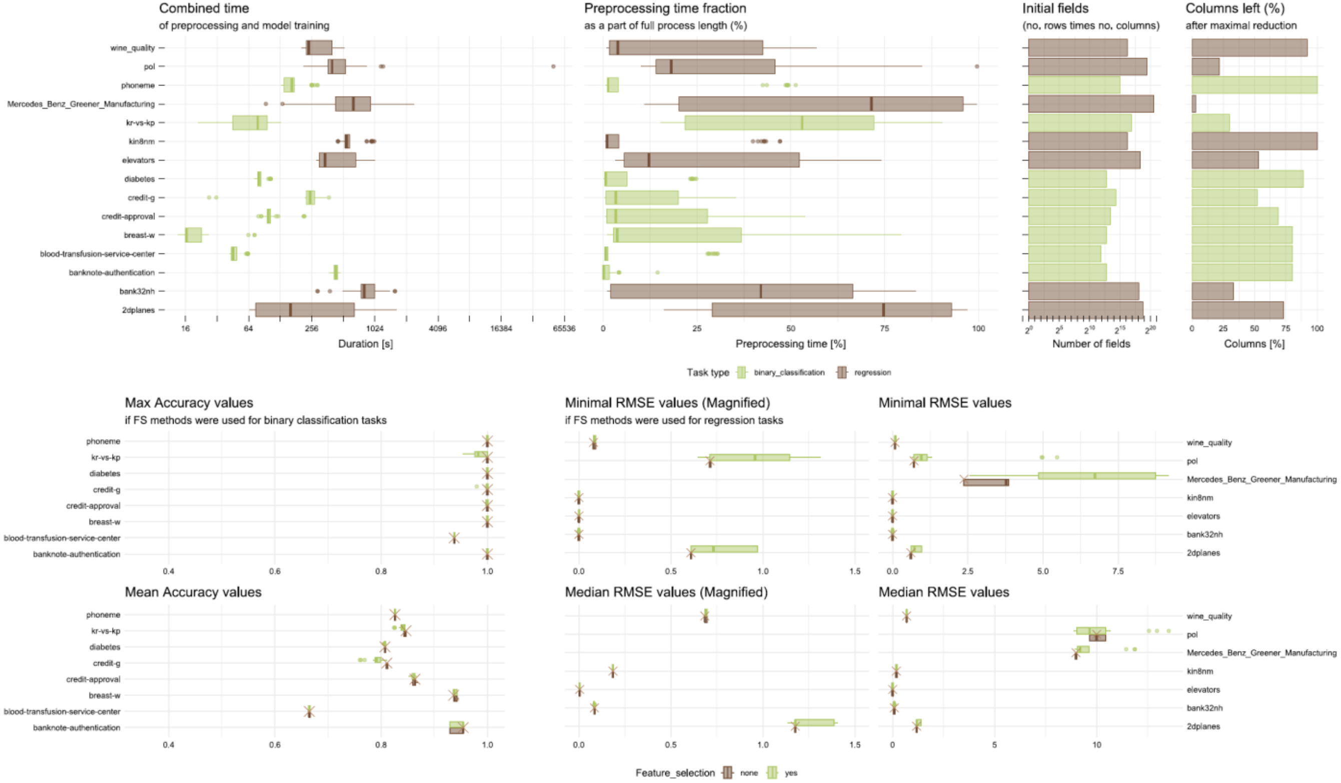


Data set	Rows	Columns	Static	Duplicate pairs	Missing fields	Dimensional issues	Correlation pairs	Imbalance	ID-like
banknote-authentication	1372	5	0	0	0	No	1	No	No
blood-transfusion-service-center	748	5	0	0	0	No	1	Yes	No
breast-w	699	10	0	0	16	No	9	Yes	No
credit-approval	690	16	0	0	37	No	1	No	No
credit-g	1000	21	0	0	0	No	0	Yes	No
diabetes	768	9	0	0	0	No	0	Yes	No
kr-vs-kp	3196	37	4	0	0	Yes	0	No	No
phoneme	5403	6	0	0	0	No	0	Yes	No



Data set	Rows	Columns	Static	Duplicate pairs	Missing fields	Dimensional issues	Correlation pairs	Imbalance	ID-like
2dplanes	40768	11	0	0	0	No	0	No	No
bank32nh	8192	33	0	0	0	Yes	0	Yes	No
elevators	16599	19	2	0	0	No	11	Yes	No
kin8nm	8192	9	0	0	0	No	0	No	No
Mercedes_Benz_Greener_Manufacturing	4209	378	145	134	0	Yes	522	No	Yes
pol	15000	49	22	156	0	Yes	2	Yes	No
wine_quality	6497	12	0	0	0	No	1	Yes	No

Ablation study



Master’s degree

1. **Implementation of multiclass classification (MC).**

2. **Adding a few more datasets for current tasks.**

3. **Conducting preprocessing methods ablations study for MC.**

4. **Evaluation of various preprocessing strategies on new data.**

5. **Uploading forester to CRAN Repository.**

Thank you for attention!

forester: an R package for automated building of tree-based machine learning models

Anna Kozak, Adrianna Grudzień, Hubert Ruczyński, Patryk Słowakiewicz
MI2.AI Group, Faculty of Mathematics and Information Science, Warsaw University of Technology

Introduction

A significant amount of time is spent on building models with high performance. Selecting the appropriate model structures, optimising hyperparameters and explainability are only part of the process of creating a machine learning-based solution. Despite the wide range of structures considered, tree-based models are champions in competitions or hackathons. So, aren't tree-based models enough? They are, and that's why **we want to fully automate the process of training tree-based models so that even the newcomers can easily build**, train and understand these powerful prediction tools. At the same time, **the experienced users gain a powerful tool for making high-quality baseline models** for new tasks, they start working with.

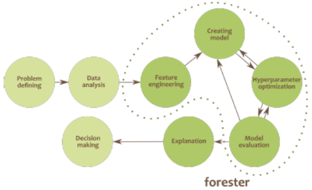
What is the *forester*?

The *forester* is **an autoML tool in R** that wraps up all machine learning processes into a single `train()` function, which includes:

1. rendering a brief data check report,
2. preprocessing initial dataset enough for models to be trained,
3. training 5 tree-based models with default parameters, random search and Bayesian optimisation,
4. evaluating them and providing a ranked list.

However, that's not everything that the *forester* has to offer. Via additional functions, the user can easily explain created models with the usage of *DALEX* or generate one of the predefined reports including:

1. information about the dataset,
2. in-depth parameters of trained models,
3. visualisations comparing the best models,
4. explanations of the aforementioned models.



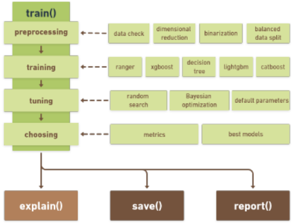
Why tree-based models?

Tree-based models, especially **XGBoost are extremely popular amongst winners in Kaggle competitions** and they firmly show their superiority with tabular data, not only in terms of fast computations. Moreover, the researchers also prove that **tree-based models are superior to deep learning neural networks** because they don't suffer from uninformative columns presence and are not biased toward overly smoothed solutions.

MI RESEARCH

Package structure

With functions in *forester* package users can create a well-tuned tree-based model with a unified, simple formula. With the usage of only two required parameters: the raw, not preprocessed dataset and target column name, the user is able to achieve satisfying results. The *forester* automatically handles the "ugly" part for you.



For whom is this package created?

The *forester* is designed for beginners in data science, but also for more experienced users. They get an easy-to-use tool that can be used to prepare high-quality baseline models for comparison with more advanced methods or a set of output parameters for more thorough optimisations. **Tree-based models are created in just one line of code.** The package differentiates itself in this aspect from powerful autoML frameworks like *mlr3* and *H2O*.

	forester	mlr3	H2O
easy to use	✓		
preprocesing	✓	✓	
autoML	✓	✓	✓
feature selection	🕒	✓	✓
model tuning	✓	✓	✓
vizualization	✓		✓
explanation	✓		✓
report	✓		✓

Contact info

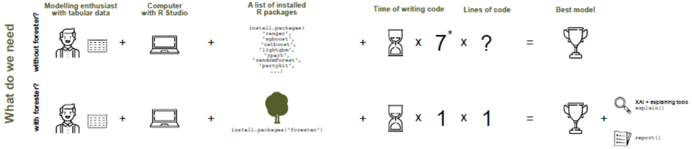
✉ anna.kozak@pw.edu.pl
✉ grudziena@outlook.com
✉ hruczynski21@interia.pl
✉ slowakiewiczpatryk@outlook.com
https://github.com/ModelOriented/forester

MI

forester: growing transparent tree-based models for everyone

Anna Kozak[1], Adrianna Grudzień[1], Hubert Ruczyński[1],
Patryk Słowakiewicz[1], Przemysław Biecek[1,2]
[1]MI2.AI, Warsaw University of Technology [2]MI2.AI, University of Warsaw

Let's talk about AutoML, tree-based models, explainable AI (XAI), exploratory data analysis (EDA)!


What do we need
without forester?
with forester?
Modelling enthusiast with tabular data
Computer with R Studio
A list of installed R packages
Time of writing code
Lines of code
Best model
install.packages('ranger', 'xgboost', 'catboost', 'lightgbm', 'rpart', 'randomForest', 'partykit', ...)
install.packages('forester')
+
x 7* x ?
=
x 1 x 1
XAI + explaining tools explain()
report()

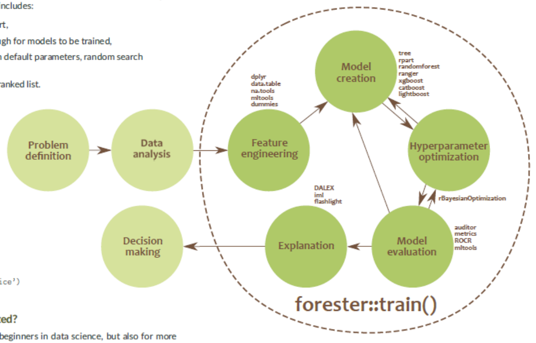
How to build tree-based models in R?

What is *forester*?

- full automation of the process of training tree-based models
- no demand for ML expertise
- powerful tool for making high-quality baseline models for experienced users

The *forester* package is **an AutoML tool in R** that wraps up all machine learning processes into a single `train()` function, which includes:

1. rendering a brief **data check** report,
2. **preprocessing** initial dataset enough for models to be trained,
3. **training** 5 tree-based models with default parameters, random search and Bayesian optimisation,
4. **evaluating** them and providing a ranked list.



How to use it?

```
library(forester)
data('lisbon')
train_output <- train(lisbon, 'Price')
```

For whom is this package created?

The *forester* package is designed for beginners in data science, but also for more experienced users. They get an easy-to-use tool that can be used to prepare high-quality baseline models for comparison with more advanced methods or a set of output parameters for more thorough optimisations.

Contact info

✉ anna.kozak@pw.edu.pl
https://github.com/ModelOriented/forester

References

P. Biecek. DALEX: Explainers for Complex Predictive Models in R. *Journal of Machine Learning Research*, 19(84):1–5, 2018. URL https://jmlr.org/papers/v19/18-416.html.

A. Kozak, H. Ruczyński, P. Słowakiewicz, A. Grudzień, and P. Biecek. *forester: Quick and Simple Tools for Training and Testing of Tree-based Models*, 2022. URL https://github.com/ModelOriented/forester. R package version 1.0.0.

Prepare meaningful report less than in 60 seconds!

As data scientists, we are fully aware that there are some time expensive processes in out work. One of them is creating a report with meaningful results. That's why one of the most powerful *forester* feature, which makes it a efficient tool for both experienced users and the newcomers, is a `report()` function. This single-line command is designed to **provide a holistic view on the outcomes of the ML process** happening inside of the *forester*.

See the report yourself!

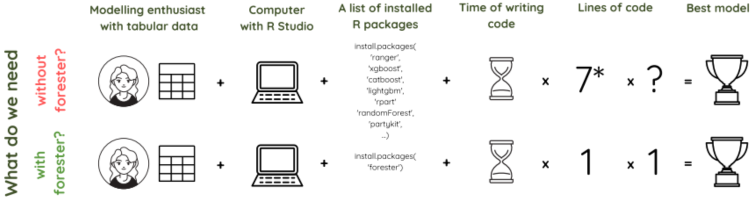
forester: A Novel Approach to Accessible and Interpretable AutoML for Tree-Based Modeling

Anna Kozak [1], Hubert Ruczyński[1]

[1]Warsaw University of Technology

Let's talk about AutoML, tree-based models, explainable AI (XAI), and exploratory data analysis (EDA)!

How to build tree-based models in R?



```
install.packages(
'ranger',
'xgboost',
'catboost',
'lightgbm',
'rpart'
'randomForest',
'partykit',
...)
```

```
install.packages(
'forester')
```

* dependent on the number of packages used

What is *forester?*

- Full automation of the process of training tree-based models,
- No demand for ML expertise,
- Powerful tool for making high-quality baseline models for experienced users.

The *forester* package is **an AutoML tool in R** that wraps up all machine learning processes into a single `train()` function, which includes:

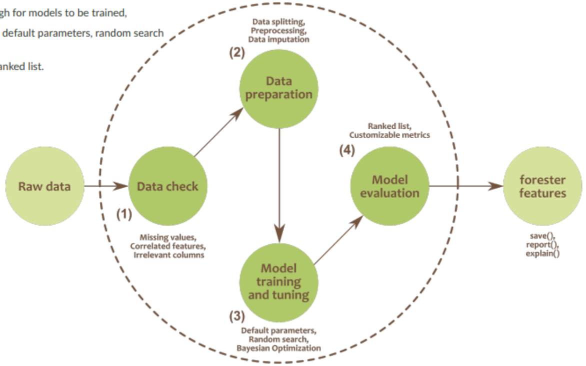
1. Rendering a brief **data check** report,
2. **Preprocessing** initial dataset enough for models to be trained,
3. **Training** 5 tree-based models with default parameters, random search and Bayesian optimisation,
4. **Evaluating** them and providing a ranked list.

For whom is this package created?

The *forester* package is designed for beginners in data science, but also for more experienced users. They get an easy-to-use tool that can be used to prepare high-quality baseline models for comparison with more advanced methods or a set of output parameters for more thorough optimisations.

How to use it?

```
library(forester)
data('lisbon')
train_output <- train(lisbon, 'Price')
```

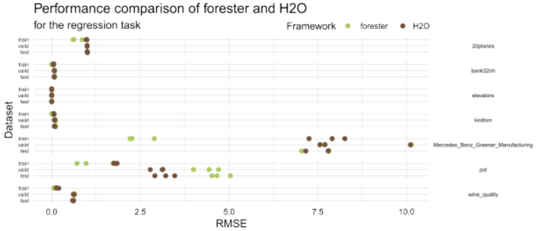


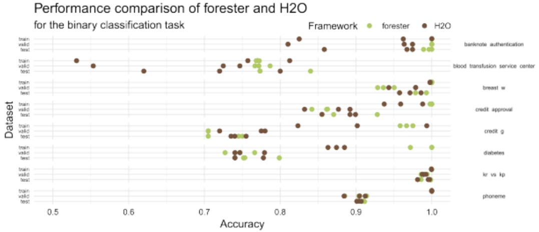
Prepare meaningful report less than in 60 seconds!

As data scientists, we are fully aware that there are some time expensive processes in out work. One of them is creating a report with meaningful results. That's why one of the most powerful *forester* feature, which makes it a efficient tool for both experienced users and the newcomers, is a `report()` function. This single-line command is designed to **provide a holistic view on the outcomes of the ML process** happening inside of the *forester*.

Simple doesn't mean worse!

According to our experiments, the forester package achieves competitive results in much shorter time in comparison to well-known H2O AutoML tool. We have compared their performance on 8 binary classification, and 7 regression tasks, and the calculations were repeated 3 times for each dataset and framework. The forester outperformed H2O most of the times, even though the latter package's training lasted 2 times longer on average.


Performance comparison of forester and H2O
for the regression task


Performance comparison of forester and H2O
for the binary classification task

Contact info

✉ anna.kozak@pw.edu.pl
✉ hruczynski21@interia.pl
https://github.com/ModelOriented/forester

Paper

GitHub

References

L. Grinsztajn, E. Oyallon, and G. Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=Fp7__phQszn.

A. Kozak, H. Ruczyński, P. Słowakiewicz, A. Grudzień, and P. Biecek. *forester: Quick and Simple Tools for Training and Testing of Tree-based Models*, 2023. URL https://github.com/ModelOriented/forester. R package version 1.1.4.

Investigating the Efficiency of Tree-based Models for Tabular Data with *forester* Package

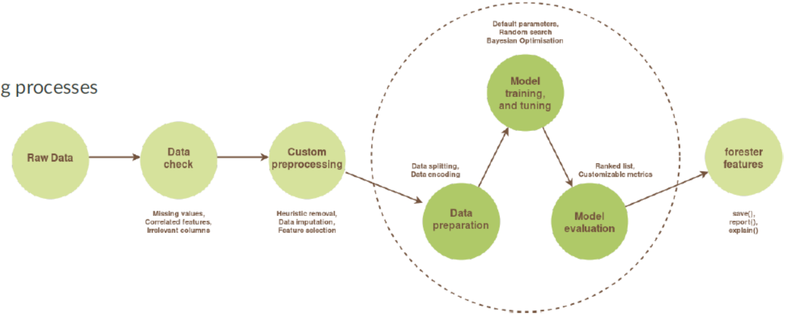
Hubert Ruczyński, Anna Kozak

Warsaw University of Technology

What is *forester?*

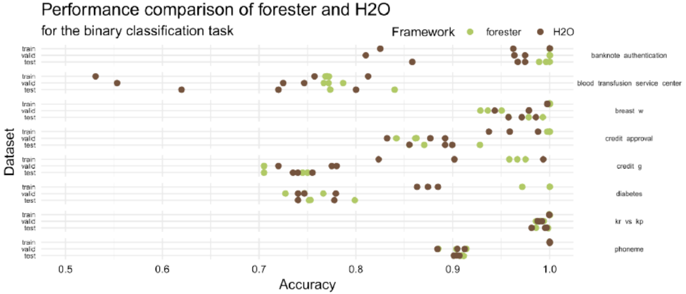
The *forester* package is **an AutoML tool in R** that wraps up all machine learning processes into a single `train()` function, which includes:

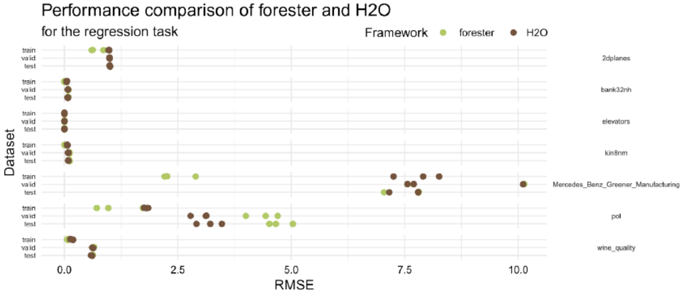
1. rendering a brief **data check** report,
2. **preprocessing** initial dataset enough for models to be trained,
3. **training** 5 tree-based models with default parameters, random search and Bayesian optimization,
4. **evaluating** them and providing a ranked list.

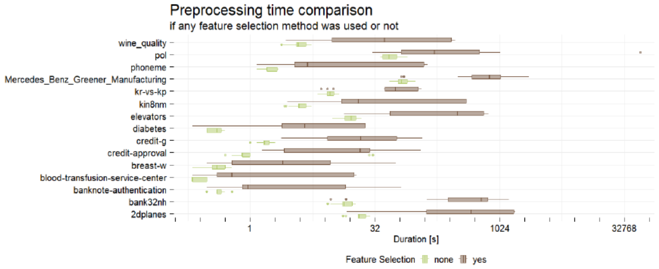

Raw Data
Data check
Missing values, Correlated features, Irrelevant columns
Custom preprocessing
Heuristic removal, Data imputation, Feature selection
Data preparation
Data splitting, Data encoding
Model training, and tuning
Default parameters, Random search Bayesian Optimisation
Model evaluation
Ranked list, Customizable metrics
forester features
save(), report(), explain()

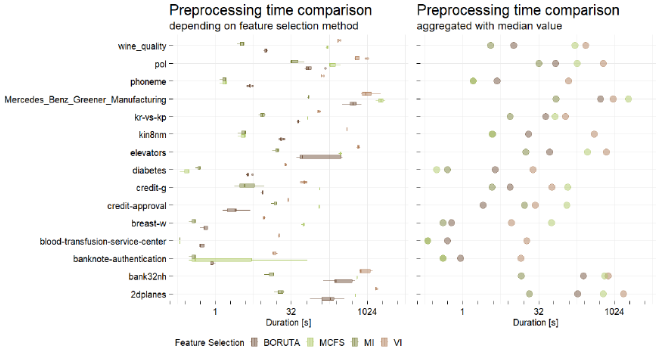
Simple doesn't mean worse!

According to our experiments, the forester package achieves competitive results in much shorter time in comparison to well-known H2O AutoML tool. We have compared their performance on 8 binary classification, and 7 regression tasks, and the calculations were repeated 3 times for each dataset and framework. The forester outperformed H2O most of the times, even though the latter package's training lasted 2 times longer on average.



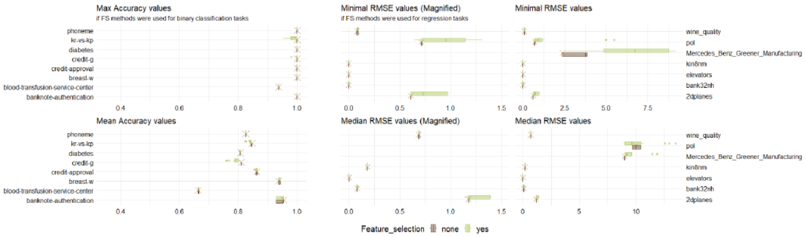


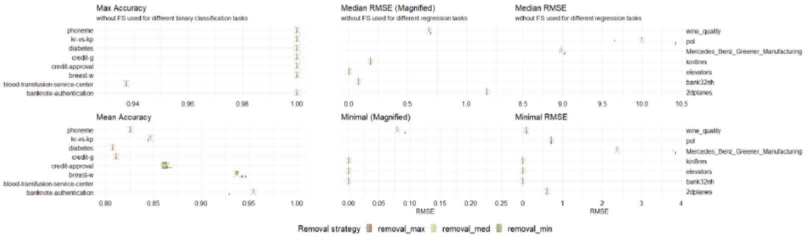




Time complexity

The most important findings consider feature selection (FS) methods. They are the most expensive part of preprocessing, and their execution times differ significantly between the methods. The right plot shows us that Mutual Information (MI) based selection method, and BORUTA are relatively fast, whereas Monte Carlo Feature Selection (MCFS), and Variable Importance (VI) are rather slow.





Feature selection impact on performance

FS methods are responsible for unstable results, and in most cases, its usage leads to worse results than for baseline methods marked with **X**. In some cases however, with FS methods we can obtain better results.
When we consider preprocessing strategies based on heuristic removals, the results are less significant, but in most cases lead to enhancements of the results.

Contact info

✉ hruczynski21@interia.pl
https://github.com/ModelOriented/forester

References

A. Kozak and H. Ruczyński. forester: A Novel Approach to Accessible and Interpretable AutoML for Tree-Based Modeling. In *AutoML Conference 2023 (ABCD Track)*, 2023. URL https://openreview.net/forum?id=Q3DWpGoX7PD.