Neural Computing and Applications (2021) 33:3053–3068
https://doi.org/10.1007/s00521-020-05172-3

ORIGINAL ARTICLE

Super ensemble learning for daily streamflow forecasting: large-scale demonstration and comparison with multiple machine learning algorithms

Hristos Tyralis[1,2] · Georgia Papacharalampous[1] · Andreas Langousis[3]

Received: 28 December 2019 / Accepted: 26 June 2020 / Published online: 6 July 2020

Abstract
Daily streamflow forecasting through data-driven approaches is traditionally performed using a single machine learning algorithm. Existing applications are mostly restricted to examination of few case studies, not allowing accurate assessment of the predictive performance of the algorithms involved. Here, we propose super learning (a type of ensemble learning) by combining 10 machine learning algorithms. We apply the proposed algorithm in one-step-ahead forecasting mode. For the application, we exploit a big dataset consisting of 10-year long time series of daily streamflow, precipitation and temperature from 511 basins. The super ensemble learner improves over the performance of the linear regression algorithm by 20.06%, outperforming the "hard to beat in practice" equal weight combiner. The latter improves over the performance of the linear regression algorithm by 19.21%. The best performing individual machine learning algorithm is neural networks, which improves over the performance of the linear regression algorithm by 16.73%, followed by extremely randomized trees (16.40%), XGBoost (15.92%), loess (15.36%), random forests (12.75%), polyMARS (12.36%), MARS (4.74%), lasso (0.11%) and support vector regression (– 0.45%). Furthermore, the super ensemble learner outperforms exponential smoothing and autoregressive integrated moving average (ARIMA). These latter two models improve over the performance of the linear regression algorithm by 13.89% and 8.77%, respectively. Based on the obtained large-scale results, we propose super ensemble learning for daily streamflow forecasting.

Keywords Combining forecasts · Ensemble learning · Hydrology · Stacking

1 Introduction

Streamflow forecasting at various temporal scales and time steps ahead is important for engineering purposes (e.g. hydro-power generation, dam regulation and other water resources engineering purposes), as well as environmental and societal purposes (e.g. flood protection and long-term water resources planning). Here, we are interested in one-step-ahead daily streamflow forecasting.

In streamflow forecasting, the predictive ability of the implemented model is of high importance; therefore, more flexible albeit less interpretable models (e.g. machine learning algorithms) are acceptable, given that they are more accurate. While accuracy is important in engineering, the current trend in the field of hydrology favours model interpretability (see, for example, [11]). The reader is referred to [18, 45, pp 24–26)] and [73], for a general discussion on the issue of interpretability versus flexibility

✉ Hristos Tyralis
hristos@itia.ntua.gr; montchrister@gmail.com

Georgia Papacharalampous
papacharalampous.georgia@gmail.com

Andreas Langousis
andlag@alum.mit.edu

1 Department of Water Resources and Environmental Engineering, School of Civil Engineering, National Technical University of Athens, Iroon Polytechniou 5, 157 80 Zografou, Greece

2 Air Force Support Command, Hellenic Air Force, Elefsina Air Base, 192 00 Elefsina, Greece

3 Department of Civil Engineering, School of Engineering, University of Patras, University Campus, Rio, 26 504 Patras, Greece

or, equivalently, understanding versus prediction in algorithmic modelling. Here, focus is on accuracy.

The dominant approach in daily streamflow forecasting is the implementation of machine learning regression algorithms, while linear models (mostly time series models) have been found to be more competitive at larger time scales (e.g. monthly and annual; [62, 63]). Regression algorithms model the dependent variable (streamflow at some time) as function of a set of selected predictor variables (e.g. past streamflow values, past precipitation values and past temperature values, with the latter two types of information being collectively referred to as "exogenous predictor variables" for this particular forecasting problem). In the case of machine learning regression, this function is learnt directly from data through an algorithmic approach. Popular algorithms include neural networks (see, for example, [1, 25, 54, 78]), support vector machines [69], decision trees, random forests and their variants [85], with numerous algorithmic variants (see, for example, [28] for the most representative ones) having been more or less applied to hydrologic case studies. Note, however, that existing approaches to daily streamflow forecasting are mostly based on the implementation of a single machine learning algorithm.

Combining forecasts from different methods has been proved to increase the forecasting accuracy. This point was initially raised by [7], while the argumentation in favour of forecast combinations, referred to as *"ensemble learning"* in the literature, was further strengthened in the early 90s (see, for example, [37, 93]). The "*no free lunch theorem*" [103] implies that no universally best machine learning algorithm exists. Thus, ensemble learning, i.e. combining multiple machine learning algorithms (hereinafter termed as base-learners) instead of using a single one, may increase the predictive accuracy of the forecasts. Overviews of model combinations in general and ensemble learning in particular can be found in [29] and [72], respectively. Here, we are interested in stacked generalization (also referred to as stacking), a particular type of ensemble learning where base-learners are properly weighted, so certain performance metrics are minimized (see, for example, [66, 87] for specific applications in probabilistic hydrological post-processing), which was initially suggested by [102] and later investigated by [16] for regression.

The simplest combination of models is equal weight averaging. The latter combination approach has been proved *"hard to beat in practice"* by more complex combination methods, a finding that has been termed "*forecast combination puzzle*" by [79]. While research on the causes of the "*forecast combination puzzle*" is currently inconclusive (see, for example, [23, 76, 83]), one can intuitively attribute its sources to the fact that as the level of uncertainty (or equivalently the number of base-learners) increases, weight optimization may not lead to significant improvements relative to simple averaging, i.e. a uniform weighting scheme that assigns equal weights to all base-learners (see [87]).

Most published studies focusing on daily streamflow forecasting use small datasets (e.g. data collected from a couple of rivers) to present some type of new method, usually referred to as hybrid when combining, for example, neural networks with an optimization algorithm. While such studies may be useful from a hydrological standpoint, the obtained results cannot be conclusive regarding the accuracy of the proposed method, due to the high degree of randomness induced by sample variability. While small-scale applications were acceptable in the early era of neural network hydrology, the current status of data availability allows for large-scale applications. Actually, recent studies based on big datasets have revealed ground breaking results in the field of hydrological forecasting (see, for example, [62, 65]), as large-scale applications allow for less biased simulation designs to assess the relative performance of new and existing methods (see, for example, the commentary in [12]).

The aim of our study is to propose a new practical system for streamflow forecasting based on a stacking algorithm, specifically super ensemble learning [90]. We conduct a large-scale investigation and find that the proposed practical system outperforms a diverse and wide variety of methods that are commonly used in hydrology for daily streamflow forecasting. Along with the introduction of the new practical system, our study aims at advancing the existing knowledge and current state of the art in the field of machine learning by:

a. Introducing a super ensemble learning framework to combine 10 machine learning algorithms, together with a predictor variable selection scheme based on random forests importance metrics, and comparing super ensemble learning with the "*hard to beat in practice*" equal weight combiner.
b. Assessing the relative performance of two time series models and 10 individual machine learning algorithms in daily streamflow forecasting, and comparing them with the super ensemble learning framework.
c. Using more than 500 streamflow time series to support the quantitative conclusions reached.

Beyond presentation of the new practical system, we consider remarks (b) and (c) above equally important, since most studies in the field use small datasets (i.e. formed by a single-digit number of time series) to compare a limited number of machine learning algorithms. Use of big datasets can provide insights and facilitate understanding and contrasting of the properties of various algorithms in predicting daily streamflow, consisting an important asset for engineering applications.

2 Methods

In this section, we present short descriptions of the individual machine learning algorithms (base-learners) used (please note that an exhaustive presentation of the algorithms is out of the scope of the present study), the three combiner learners (i.e. super ensemble learner, equal weight combiner and best learner), the variable selection methodology, the statistical time series forecasting methods, the metrics used to assess the relative performance of the algorithms and the testing procedure.

2.1 Statistical time series forecasting methods

Here, we present the statistical time series forecasting methods that are compared to the proposed practical system. These methods are well established in the literature, while their implementation is fully automated in the forecast R package [43, 44]; therefore, in what follows, short descriptions are provided. A typical property of such models is that they are fitted to the time series of interest (i.e. the streamflow time series in our case), thereby not exploiting available information from other predictor variables (i.e. temperature and precipitation variables in our case). Furthermore, such models can exploit temporal dependencies in the observations [14], while most machine learning algorithms cannot. Details regarding the training periods of the time series models can be found in Sect. 2.6.

2.1.1 Exponential smoothing method

Simple exponential smoothing methods compute weighted moving averages of past time series values. They were introduced by [19, 41, 101]. Variants of exponential smoothing models that can account for drifts and seasonality also exist. Here, we used the automated method of the forecast R package, which employs a procedure for automatic estimation of trend parameters. We did not let the algorithm estimate the seasonality of the data, because this would result in unstable forecasts, given that 365 seasons should be estimated. An alternative option is to fit different models to each month, but we did not choose this option, due to the secondary benchmarking role of the model. It should be noted that the first application of exponential smoothing models in geophysical time series forecasting can be found in [27], and that a large-scale comparison with other models can be found in [65]. While use of exponential smoothing models has been limited to geophysical time series forecasting, these algorithms are popular in other fields (econometrics, etc.).

2.1.2 ARIMA models

Autoregressive Integrated Moving Average (ARIMA) stochastic processes model time series by combining autoregressive (where dependent variables depend linearly on their previous values) and moving average (where dependent variables depend linearly on previous white noise terms) stochastic schemes, and simultaneously model trends. They have been popularized by [14], while a more recent treatment can be found in [15]. A first application in hydrology can be found in [20]. Here, we use the automated forecasting procedure implemented in the forecast R package.

2.2 Base-learners

A detailed description of the majority of the base-learners exploited herein is out of the scope of the manuscript and can be found in [39, 45]. All algorithms have been implemented and documented in the R programming language. Details on their software implementation can be found in "Appendix". To ensure reproducibility of the results, "Appendix" also includes the versions of the software packages used herein.

2.2.1 Linear regression

Linear regression is the simplest model used herein. It is described in detail by [39 pp 43–55]. The dependent variable is modelled as a linear combination of the predictor variables, while the weights are estimated by minimizing the residual sum of squares (least squares method).

2.2.2 Lasso

The least absolute shrinkage and selection operator (lasso) algorithm [82] performs variable selection and regularization by imposing the lasso penalty (L_1 shrinkage) in the least squares method, aiming to shrink its coefficients, while allowing for elimination of non-influential predictor variables by nullifying their coefficients.

2.2.3 Loess

Locally estimated scatterplot smoothing (loess, [24]) fits a polynomial surface (determined by the predictor variables) to the data by using local fitting. Here, we used a second-degree polynomial.

2.2.4 Multivariate adaptive regression splines

Multivariate adaptive regression splines (MARS, [30, 31]) is a weighted sum of basis functions, with total number and associated parameters (i.e. product degree and knot locations, respectively) being automatically determined from data. Here, we build an additive model (i.e. a model without interactions), where the predictor variables enter the regression through a linear sum of hinge basis functions.

2.2.5 Multivariate adaptive polynomial spline regression

Multivariate adaptive polynomial spline regression (polyMARS, [49, 80]) is an adaptive regression procedure that uses piecewise linear splines to model the dependent variable. It is similar to MARS, with main differences being that "*(a) it requires linear terms of a predictor to be in the model before nonlinear terms using the same predictor can be added and (b) it requires a univariate basis function to be in the model before a tensor-product basis function involving the univariate basis function can be in the model*" [48].

2.2.6 Random forests

Random forests [17] are bagging (abbreviation for bootstrap aggregation) of regression trees with an additional degree of randomization, i.e. they randomly select a fixed number of predictor variables as candidates when determining the nodes of the decision tree.

2.2.7 XGBoost

Extreme Gradient Boosting (XGBoost, [21]) is an implementation of gradient boosted decision trees (see, for example, [32, 56, 58], albeit considerably faster and better performing. Gradient boosting is an approach that creates new models (in this case decision trees) to predict the errors of prior models. The final model adds all fitted models. A gradient descent algorithm is used to minimize the loss function when adding new decision trees. XGBoost uses a model formalization that is more regularized to control overfitting. This procedure renders XGBoost more accurate than gradient boosting.

2.2.8 Extremely randomized trees

Extremely randomized trees [36] are similar to random forests. These two models mostly differ in the splitting procedure. Contrary to random forests, in extremely randomized trees the cut-point is fully random.

2.2.9 Support vector machines

The principal concept of support vector regression is to estimate a linear regression model in a high-dimensional feature space. In this space, the input data are mapped using a (nonlinear) kernel function [77, 91]. Here, we used a radial basis kernel.

2.2.10 Neural networks

The principal concept of neural networks is to extract linear combinations of the predictor variables as derived features and then model the dependent variable as a nonlinear function of these features [39, p 389]. Here, we used feed-forward neural networks [70, pp 143–180].

2.3 Super ensemble learning

Super ensemble learner is a convex weighted combination of multiple machine learning algorithms, with weights that sum to unity and are equal or higher than zero (see [88–90]). The weights are estimated through a k-fold cross-validation procedure (here, we choose $k = 5$) in the training set (see Sect. 2.6), so that a properly selected loss function is minimized. Here, we minimize the quadratic loss function, which is equivalent to minimizing the root-mean-squared error (RMSE). Then, the base-learners are retrained in the full training dataset, and the super ensemble learner predictions are obtained as the weighted sum (using the estimated weights of the cross-validation procedure) of the retrained base-learners. The design of the algorithm is presented in Fig. 1. Super ensemble learning (as every stacking algorithm) can combine ensemble learners (e.g. bagging algorithms, boosting algorithms and more) and different types of base-learners, while, for example, bagging or boosting algorithms use a single type of base-learners.

Algorithm 1 presents the formal procedure of super ensemble learning for a training set of N observations. Some theoretical results and recommendations for the implementation of super learning algorithms can be found in [90]. In particular:

a. It is recommended to use as many as possible sensible base-learners.
b. Due to the cross-validation procedure, overfitting is avoided.
c. Different loss functions can be applied. For instance, by construction random forests are not a minimization procedure. However, if one wants to minimize the L_1 loss, one can still include random forests in the mix of base-learners, as the optimization procedure of super

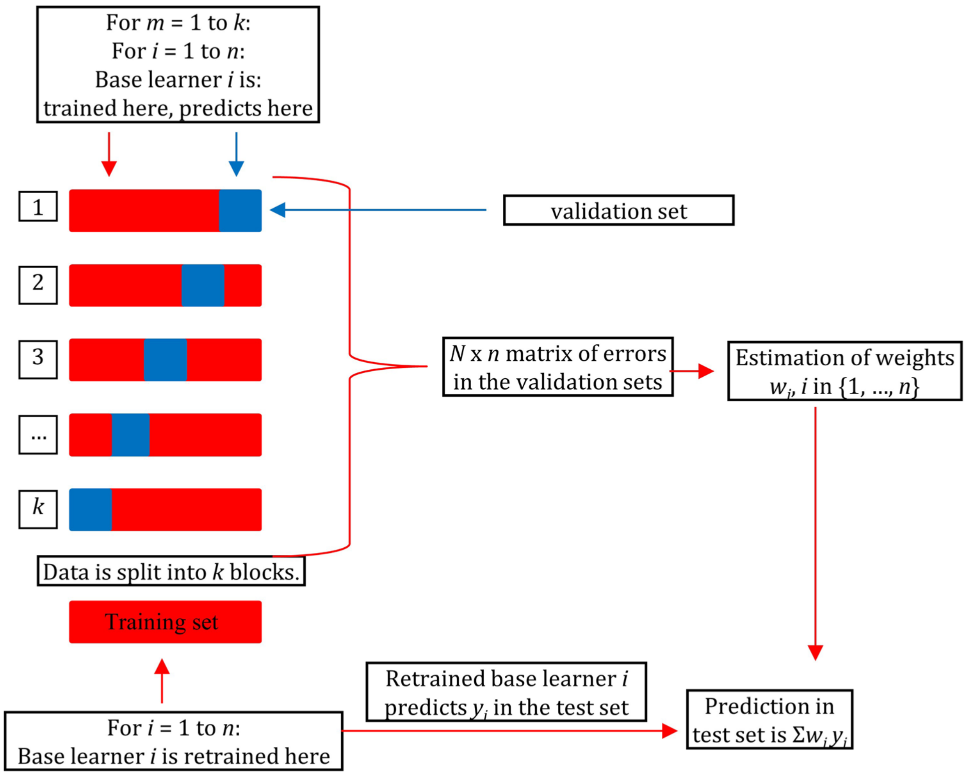

For m = 1 to k:
For i = 1 to n:
Base learner i is:
trained here, predicts here
1
2
3
...
k
validation set
N x n matrix of errors in the validation sets
Estimation of weights w_i, i in {1, ..., n}
Data is split into k blocks.
Training set
For i = 1 to n:
Base learner i is retrained here
Retrained base learner i predicts y_i in the test set
Prediction in test set is $\Sigma w_i y_i$

Fig. 1 Design of the super ensemble learner corresponding to Algorithm 1. Red blocks in the training dataset are used for training, and blue blocks are used for validation

ensemble learning will assign appropriate weights to them.

d. The super ensemble learner will perform asymptotically as well as the best base-learner.

Algorithm 1. The super ensemble learning algorithm.

```
Step 1:
a. Let $N$ be the size of the training set and $z_i$, $i \in \{1, ..., N\}$ be the $i^{th}$ value of the dependent variable.
b. Split the training set to $k$ mutually exclusive sets.
c. Select $n$ base-learners.
Step 2:
For $m = 1$ to $k$:
For $i = 1$ to $n$:
a. Train $i^{th}$ base-learner to $k - 1$ sets.
b. Predict $k^{th}$ set using the $i^{th}$ base-learner. Let $y_{ji}$, $j \in \{1, ..., N\}$, $i \in \{1, ..., n\}$ be the predicted value of the corresponding $z_j$ value by the $i^{th}$ base-learner.
Step 3:
For $i = 1$ to $n$:
Train $i^{th}$ base-learner to training set.
Step 4: Create $[y_{ji}]$ $N \times n$ matrix of step 2d $N$ predictions and $n$ base-learners
Step 5: Optimize $\sum_{j \in \{1, ..., N\}} (z_j - \sum_{i \in \{1, ..., n\}} \beta_i y_{ji})^2$ subject to $\sum_{i \in \{1, ..., n\}} \beta_i = 1, \beta i \geq 0$
Step 6: Predict new data with weighted average of $n$ trained base-learners of step 3, using weights estimated in step 5.
```

2.4 Other ensemble learners

In addition to super ensemble learning, we applied the equal weight combiner by assigning a uniform weighting scheme (i.e. weights equal to 1/10) to all base-learners. Furthermore, we used an ensemble learner (referred to as best learner), which selects the best base-learner based on its performance in the k-fold cross-validation procedure in the training set.

2.5 Variable selection

Variable selection constitutes a complex problem that has been extensively investigated with no exact solution (see, for example, [40]) as selection of variables is strictly linked to the problem at hand. In daily streamflow forecasting, daily streamflow q_i may depend on past streamflow, precipitation and temperature values (q_j, p_j, t_j, $j = 1, \ldots, i - 1$, respectively). Past precipitation and temperature values are exogenous predictor variables in our problem. If non-informative predictor variables are included in the model, the performance of some algorithms (e.g. linear regression) may decrease considerably, while if too many predictor variables are included, the computational burden may become prohibitive. Missing some informative predictor variables may also harm the performance of the model.

Several strategies can be employed to select predictor variables, e.g. an exhaustive search [84], use of correlation measures, partial mutual information [55] and the like. An overview of variable selection procedures in water resources engineering can be found in [13].

Here, we selected to use the permutation variable importance metric (VIM) used by random forests algorithms for variable selection. The permutation VIM measures the mean decrease in accuracy in the out-of-bag (OOB) sample by randomly permuting the predictor variable of interest. OOB samples are the samples remaining after bootstrapping the training set (see also Sect. 2.2.6). VIM permits ranking of the relative significance of predictor variables [85] and is a commonly used variable selection procedure. We computed VIM of daily values of streamflow, precipitation and temperature of the last month, i.e. 90 predictor variables in total. We selected the resulting five most important predictor variables for each process type (i.e. streamflow, precipitation and temperature). The fitting problem is formulated as:

$$q_i = f(\{q_j, p_k, t_l, \}), j, k, l \in \{\text{five values in } (i-30, \ldots, i-1)\} \tag{1}$$

If some of the best ranked possible predictor variables display negative VIM values, they are excluded from the set of predictor variables, since they are non-informative (see, for example, [86] and references therein). In this case, the set of predictor variables includes less than 15 variables.

2.6 Training and testing

Machine learning algorithms in regression settings approximate the function f in Eq. (1) through training on data. During training, some hyperparameter optimization can be performed to enhance the performance of the model. However, default hyperparameter values used in software implementations usually display favourable properties, as, for example, proved in large-scale empirical studies in hydrology [64], while hyperparameter optimization may be computationally costly with little improvement in performance. That said, in the present study, we decided to use default hyperparameter values, as suggested in the corresponding software implementations (see "Appendix").

Time series models are fitted in the training period using the automated procedure of the forecast R package. One-step-ahead forecasts are delivered in the testing period by using the estimated parameters during the training phase.

To estimate the generalization error of the implemented algorithms, one should compare to an independent set, i.e. a set not used for training, termed as test set. Following recent theoretical studies [4], we use training and test sets of equal size (i.e. each one corresponding to 50% of the full time series) to assess the performance of the algorithms.

2.7 Metrics

Although the super ensemble learner is optimized with respect to RMSE, we use multiple metrics to understand the effect of this optimization and quantitatively assess the relative performance of the algorithms. An overview of metrics that can be used to assess the performance of forecasting methods can be found in [42]. Here, we use RMSE, the mean of absolute errors (MAE), the median of absolute errors (MEDAE) and the squared correlation r^2 between the forecasts f_n and the observations o_n. All metrics, defined by the following equations, are computed in the testing period.

$$E_n := f_n - o_n \tag{2}$$

$$\text{MAE} := (1/|N|) \sum\nolimits_n |E_n| \tag{3}$$

$$\text{RMSE} := ((1/|\text{N}|) \sum\nolimits_n \text{E}^2 n)^{1/2} \tag{4}$$

$$\text{MEDAE} := \text{median}_n\{|E_n|\} \tag{5}$$

$$r^2 := (\text{corr}(\boldsymbol{f}, \boldsymbol{o}))^2 \tag{6}$$

In Eq. (6), $\boldsymbol{f}$ and $\boldsymbol{o}$ denote the vectors of the forecasts and observations, respectively, in the testing period. MAE, RMSE and MEDAE take values in the range $[0, \infty)$, with 0 values indicating perfect forecasts. r^2 takes values in the range [0, 1], with values equal to 1 denoting perfect forecasts.

Following relevant suggestions by [6], we do not use hypothesis tests to report the significance of the differences between forecasting performances of pairs of methods, as their use in the field of forecasting may lead to misinterpretations. Instead, we prefer to use "*effect sizes*", as, for example, done in forecasting competitions [6], which in our case are "*percent error reductions*" in terms of a specified metric. Our choice also overcomes the problems of (a) computing significance of the forecasting performance differences between every pair of the implemented algorithms and (b) using some type of scaled metrics (e.g. the widely used in hydrology Nash–Suttcliffe efficiency) which is usually accompanied by other disadvantages.

3 Data and application

3.1 Data

We used CAMELS (Catchment Attributes and MEteorology for Large-sample Studies) dataset, which is used for benchmarking purposes in hydrology [61] and can be found online in [2, 59]. A detailed documentation of the dataset can be found in [3, 60]. The dataset includes daily minimum temperatures, maximum temperatures, precipitation and streamflow data from 671 small- to medium-sized basins in the contiguous US (CONUS). Temperature and precipitation time series for the needs of the analysis have been obtained by processing the daily dataset by [81]. The mean daily temperature was estimated by averaging the minimum and maximum daily temperatures. Changes in the basins due to human influences are minimal. Here, we focus on the 10-year period 2004–2013, while basins with missing data or other inconsistencies have been excluded. The final sample consists of 511 basins representing diverse climate types over CONUS; see Fig. 2.

3.2 Implementation of methods

In what follows, we detail the implementation of the algorithms and their testing, while the workflow is presented in Fig. 3.

a. The training and testing periods (hereafter denoted by T_1 and T_2, respectively) are set to T_1 = {2004-01-01, …, 2008-12-31} and T_2 = {2009-01-01, …, 2013-12-31}.
b. For an arbitrary basin, random forests VIM approach (see Sect. 2.5) is applied in period T_1 by using q_j, p_k, t_l, j, k, $l \in \{i-30, \ldots, i-1\}$ as predictor variables (90 predictor variables in total) and q_i as dependent variable. The training sample includes 1827 instances, i.e. as many as the number of days in period T_1. The five most important predictor variables for each process type (i.e. q, p, t) are selected based on their VIM values (see Sect. 2.5) and used for training of the algorithms. In the case, when less than five predictor variables have positive VIM values for a certain process type, the predictor variables with negative (or zero) VIM values are excluded and the number of the selected predictor variables reduces to less than 15. The selected predictor variables are used in the next steps.
c. All algorithms of Sects. 2.2.1–2.2.10 are trained in period T_1 in a fivefold cross-validation setting.
d. The time series models of Sect. 2.1 are trained in period T_1 using the procedure of the forecast R package.
e. The super ensemble learner (composed by the ten base-learners of step (c); see Sect. 2.3) is also trained in period T_1 using fivefold cross-validation. This is done by estimating the fivefold cross-validated risk for each base-learner in step (c) and computing its weight.
f. The ten trained base-learners of step (c) are retrained in the full T_1 period and predict streamflow in period T_2. The testing sample includes 1826 instances, i.e. as many as the number of days in period T_2.
g. The super ensemble learner (which uses the estimated weights of step (e) and weights the retrained base-learners), the equal weight combiner (which averages the 10 retrained base-learners; see Sect. 2.4) and the best learner (i.e. the retrained base-learner with the least fivefold cross-validated risk in period T_1; see Sect. 2.4 and step (c)) predict daily streamflow in period T_2.
h. The metrics of Sect. 2.7 are computed for each of the 15 algorithms (see steps (e) and (f)) in period T_2.
i. Finally, the metric values are summarized for all basins in period T_2.

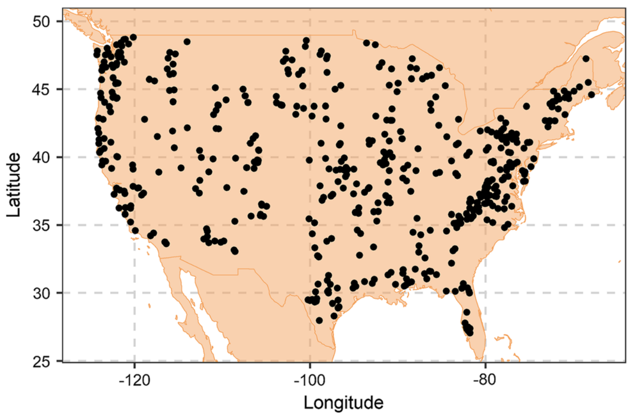


Fig. 2 The 511 basins over CONUS used in the study

4 Results

Here, we summarize the predictive performance of the 15 algorithms in period T_2 for the 511 basins. We present the rankings of the algorithms (Sect. 4.1) and their relative improvements with respect to the linear regression

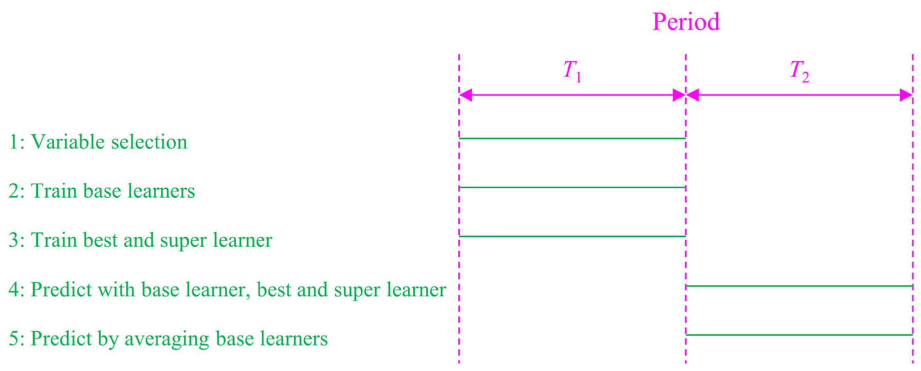

Period
T_1
T_2
1: Variable selection
2: Train base learners
3: Train best and super learner
4: Predict with base learner, best and super learner
5: Predict by averaging base learners

Fig. 3 Workflow of the proposed practical system. Time periods in which the models are applied. T_1 is the training period, while T_2 is the testing period

benchmark (Sect. 4.2). An investigation on the estimated weights of the super ensemble learner is also presented (Sect. 4.3).

4.1 Ranking of methods

Figure 4 presents the mean rankings of the 15 algorithms according to their performance in terms of the examined metrics. Rankings range from 1 to 15, with lower values indicating better performance. For instance, when examining an arbitrary basin, the 15 algorithms are ranked according to their performance in terms of each metric separately. Then, these rankings are averaged over all basins, conditional on the metric.

Super ensemble learner is the best performing algorithm in terms of RMSE and MAE, and the second best algorithm in terms of MEDAE and r^2; nonetheless, its difference from the best performing algorithm in terms of r^2 (i.e. the equal weight combiner) is minimal. In terms of RMSE, the equal weight combiner is the second best performing algorithm, followed by the best learner. From the base-learners, neural networks, extremely randomized trees and loess are the best performing algorithms (ranked from best to worst) in terms of RMSE, while support vector machines are worse compared to the linear regression benchmark. It is remarkable that time series models seem to outperform some base-learners in terms of RMSE, although they do not exploit information from exogenous predictor variables. Two possible explanations are that: (a) temporal dependencies include rich information, which is exploited by time series models but not by regression algorithms, and (b) additional information introduced by exogenous variables is relatively limited.

When focusing on metrics other than RMSE, one sees that the rankings of the algorithms remain similar, albeit not identical. For instance, while MARS is not well performing in terms of RMSE, it is the best performing learner in terms of MEDAE, contrary to the equal weight combiner, which does not perform well.

Figure 5 presents the rankings of the 15 algorithms according to their performance in terms of RMSE for the 511 basins considered. While, in general, one sees similar rankings of an algorithm at all basins (i.e. similar colours dominate a given row), there are cases where the rankings of an algorithm vary with respect to its mean performance. Take, for instance, the super ensemble learner. While it is on average the best performing algorithm, there are cases where other algorithms perform better.

4.2 Relative improvements

The median relative improvement introduced by each algorithm with respect to the linear regression benchmark is important for understanding whether a more flexible (yet less interpretable) algorithm is indeed worth implementing. In this context, Fig. 6 presents the percentage of decrease

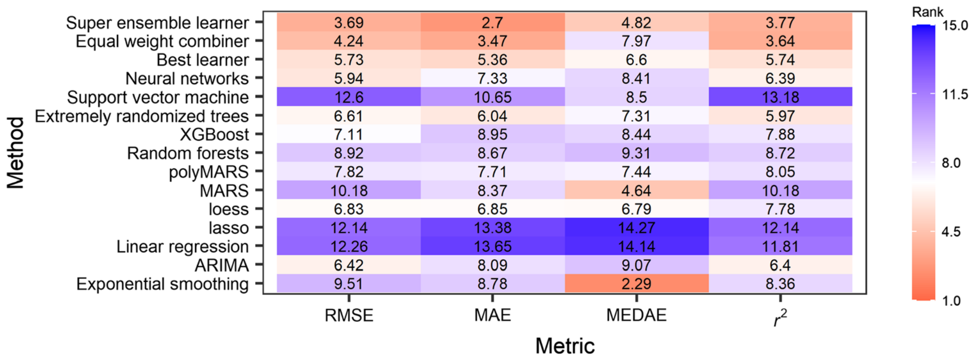


Fig. 4 Mean rankings of the 15 algorithms according to their performance in the 511 basins

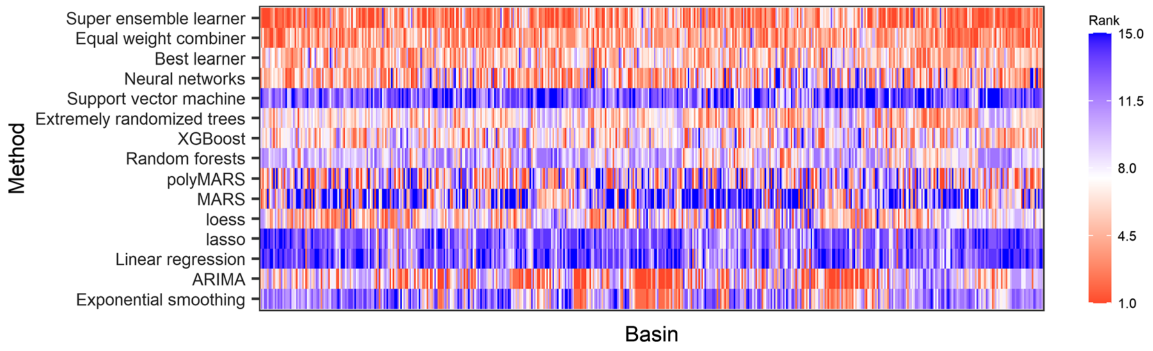
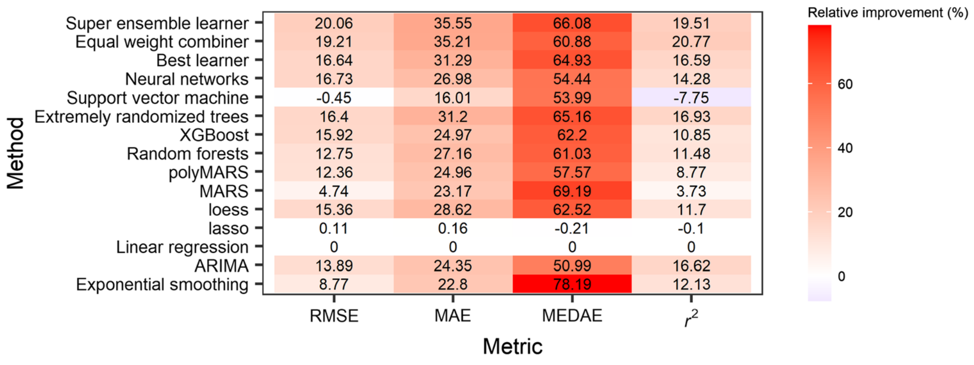


Fig. 5 Rankings of the 15 algorithms according to their performance in terms of RMSE for the 511 basins considered



Method	RMSE	MAE	MEDAE	r^2
Super ensemble learner	20.06	35.55	66.08	19.51
Equal weight combiner	19.21	35.21	60.88	20.77
Best learner	16.64	31.29	64.93	16.59
Neural networks	16.73	26.98	54.44	14.28
Support vector machine	-0.45	16.01	53.99	-7.75
Extremely randomized trees	16.4	31.2	65.16	16.93
XGBoost	15.92	24.97	62.2	10.85
Random forests	12.75	27.16	61.03	11.48
polyMARS	12.36	24.96	57.57	8.77
MARS	4.74	23.17	69.19	3.73
loess	15.36	28.62	62.52	11.7
lasso	0.11	0.16	-0.21	-0.1
Linear regression	0	0	0	0
ARIMA	13.89	24.35	50.99	16.62
Exponential smoothing	8.77	22.8	78.19	12.13

Fig. 6 Median relative improvements of the 15 algorithms with respect to the linear regression benchmark in the 511 basins considered

in the RMSE, MAE, MEDAE and r^2 introduced by each of the 15 examined learners relative to that of the linear regression benchmark.

Focusing on RMSE, super ensemble learner improves over the performance of the linear regression algorithm by 20.06%. The improvement introduced by the equal weight combiner is equal to 19.21% (not negligible as well), followed by the best learner with relative improvement equal to 16.64%. The best base-learner is neural networks, which improves over the performance of the linear regression algorithm by 16.73%, followed by extremely randomized trees (16.4%), XGBoost (15.92%) and loess (15.36%).

An important note to be made here is that the specific ranking of an algorithm in terms of the improvement it introduces relative to the linear regression benchmark depends significantly on the metric used, i.e. RMSE, MAE, MEDAE and r^2. For instance, while the equal weight combiner is the second best performing learner in terms of RMSE, MAE and r^2, it is the fourth worst performing in terms of MEDAE. In addition, please note that the magnitudes of the relative improvements differ considerably for the various metrics. For instance, relative improvements in terms of MAE mostly range between 25 and 35%, while the respective relative improvements in terms of RMSE are mostly between 10 and 20%.

To facilitate understanding of the range of forecast errors, Fig. 7 presents boxplots of the RMSE values for all 15 algorithms considered. While in most cases the forecast errors lie below 5 mm/day, one sees that MARS and PolyMARS forecasts may fail considerably (see the exceptionally high outliers), and this is the case for neural networks as well. This form of instability could also explain why MARS is amongst the best performing methods in terms of MEDAE (a metric based on medians), while it appears to be less performing when assessed using metrics based on mean errors (i.e. RMSE, MAE and r^2).

Values of r^2 are also of interest. Close inspection of Fig. 8 reveals that the super ensemble learner and the equal weight combiner display values that lie mostly in the range 0.60–0.65, while the best learner exhibits somewhat lower values. The remainder base-learners display, in general, lower r^2 values, while the mean r^2 of linear regression is somewhat higher than 0.5.

Figure 9 presents a comparison of the two best performing methods in terms of RMSE (left panel) and ranking (right panel) for each of the 511 considered basins. In terms of RMSE, the performances of the two methods seem similar, with the equal weight combiner being slightly more stable (see the few points lying above the 45° line). This behaviour should be attributed to the fact that, in some basins, the super ensemble learner may assign higher weights to inferior base-learners. Note, however, that based on Figs. 4 and 6, the median behaviour of the super ensemble learner is better than that of the equal weight

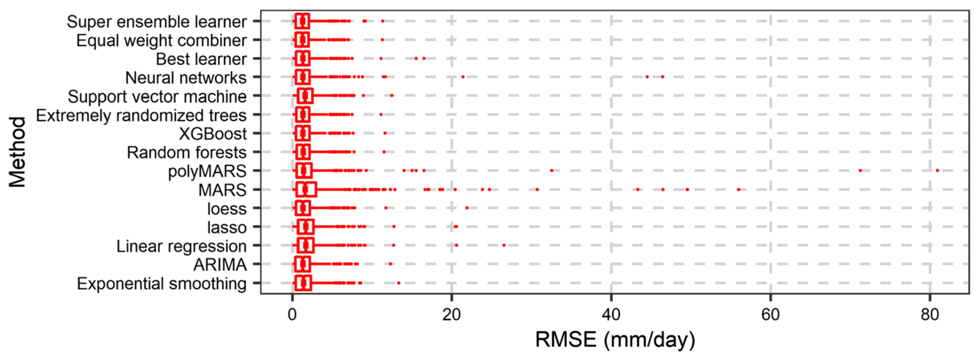


Fig. 7 Boxplots of the RMSE values computed for the 15 algorithms in the 511 basins considered

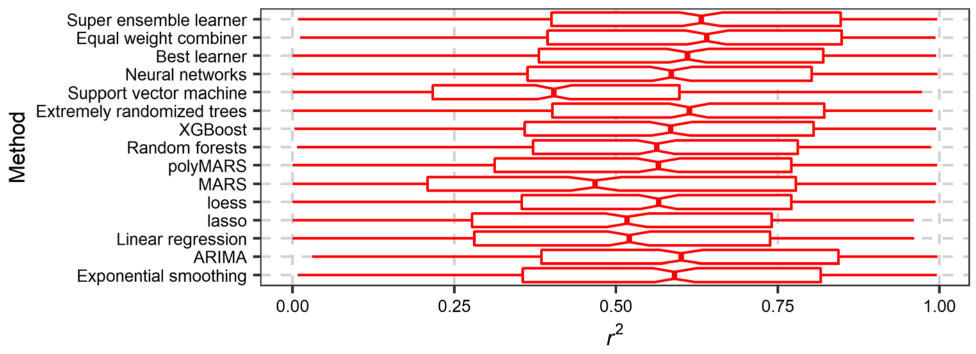


Fig. 8 Boxplot of the r^2 values computed for the 15 algorithms in the 511 basins considered

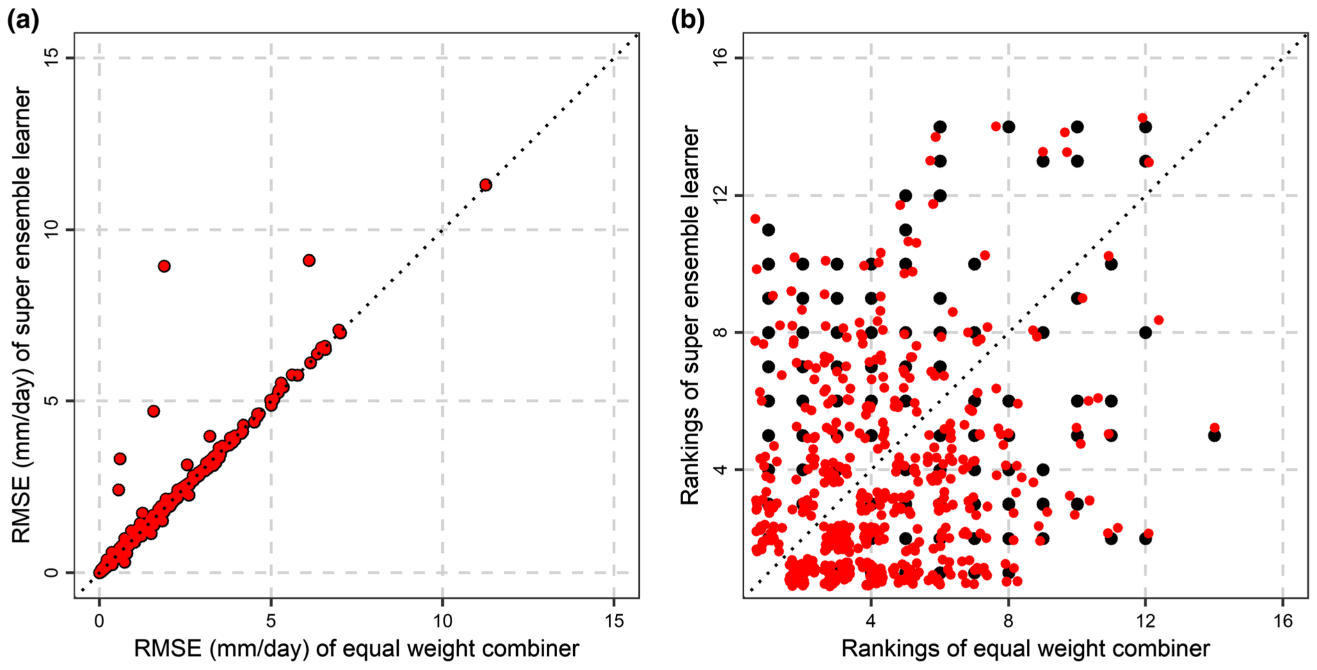


Fig. 9 Visual comparison between the equal weight combiner and the super ensemble learner, based on their performances in the testing set for each of the 511 basins considered (red points): **a** scatterplot of RMSEs. **b** Scatterplot with jitter of the ranking of the two methods (i.e. multiple co-located points are randomly displaced to appear as clusters (red points) around their exact location (black dots)) (color figure online)

combiner, and this is also observable in Fig. 9b, where the number of red points lying below the 45° line (a total of 313) is larger than those lying above (a total of 198). In other words, the super ensemble learner is ranked higher relative to the equal weight combiner in 313 out of the 511 basins considered.

4.3 Weights

The weights of the base-learners (used to compose the super ensemble learner) are strongly linked to the performance of the 10 base-learners in the test set. This becomes apparent from Fig. 10, which presents the weights assigned to the 10 base-learners per basin. More precisely, close inspection of Fig. 10 alongside with Fig. 6 reveals that less performing algorithms in the test set are assigned smaller weights.

The boxplots in Fig. 11 also confirm the aforementioned observation/finding, i.e. that the best performing methods in the cross-validation set (i.e. methods that are assigned higher weights) are those displaying the highest performance in the test set (see also Fig. 6). The highest weights are assigned to XGBoost, which is one of the best performing algorithms.

The boxplots in Fig. 12 summarize results from all basins considered and present how the weights assigned to the base-learners composing the super ensemble learner are related to their individual rankings within the testing period. Clearly, the higher the weight, the better the performance of the algorithm in the testing period.

5 Discussion

An advantage of super ensemble learning is that it can be optimized with respect to any loss function; in our case, this loss function was RMSE. Albeit base-learners may be designed to optimize other loss functions, a combination approach (such as the super ensemble learner proposed herein) may be useful to extract their advantages with respect to a specific loss function. In general, other loss functions could also be used for optimizing the super ensemble learner.

Regarding the usefulness of the proposed method, one should consider that it is fully automated and does not need any assumptions, since it exploits a *k*-fold cross-validation procedure (in contrast, for example, to Bayesian model averaging, which is widely used in hydrology).

In this paper, it is empirically shown that predictive performance improvements can be obtained by combining algorithms. We would like to emphasize that the simplest combination methods (best learner and simple averaging) resulted in significant improvements with respect to the exploited base-learners. Therefore, it is worth applying as many algorithms as possible, with the aim to further combine them. Moreover, it is empirically shown that the exploitation of exogenous predictor variables can lead to considerable improvements in forecasting performance (especially when forecasts are made by ensemble learning algorithms), relative to forecast schemes (e.g. ARIMA and exponential smoothing methods) that exclusively use past information.

Due to its automation, the super ensemble learner can be considered a practical system for hydrological time series forecasting based on available data. Furthermore, it can be integrated with weather forecasts of the exogenous variables of interest, which can be provided a day earlier and can be incorporated into the practical system. Perhaps, weather forecasts for the day of interest can provide significant information, in addition to observed data from previous days. Another extension of the practical system would be to use the full available information of precipitation and temperature from weather stations, instead of averaging this information over the basin area.

The results of the present study can improve understanding of the relative performance of the implemented base-learners and time series models when used for daily streamflow forecasting, while allowing for interpretations beyond the area of hydrological applications. Considering that many large datasets are available in hydrology and atmospheric sciences, information from these fields could benefit machine learning applications by facilitating better understanding of algorithmic properties.

6 Conclusions

We presented a new method for daily streamflow forecasting. This method is based on super ensemble learning. The introduced algorithm combines 10 base-learners and

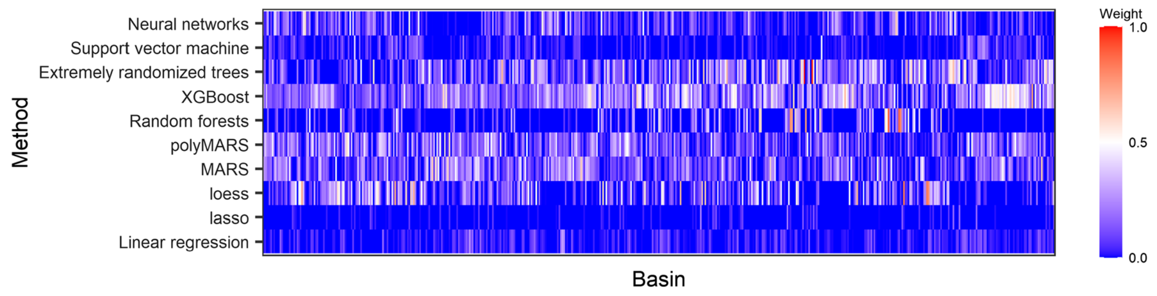


Fig. 10 Weights assigned to the 10 base-learners in the 511 basins considered

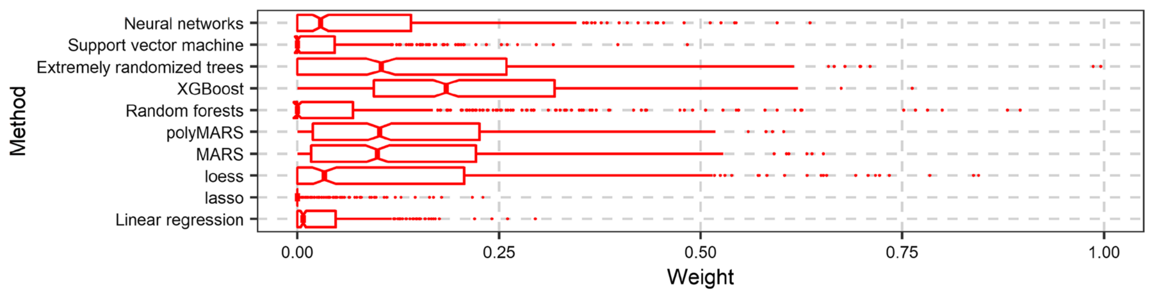


Fig. 11 Boxplot of the weights assigned to the 10 base-learners in the 511 basins considered

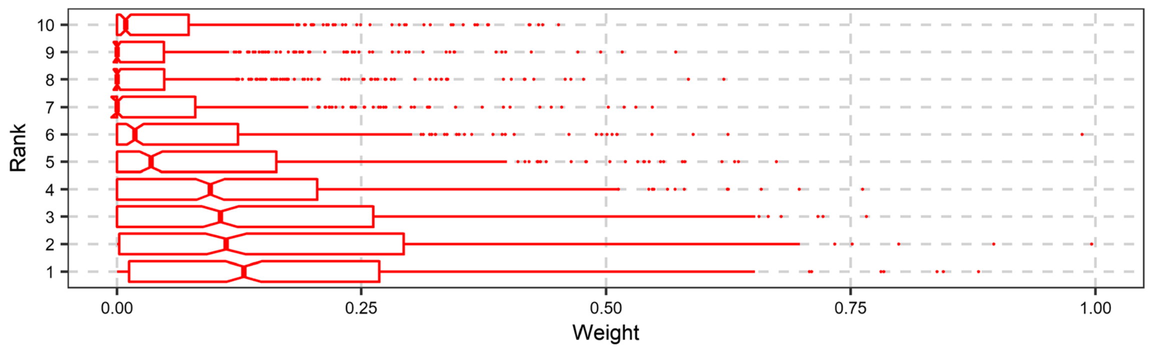


Fig. 12 Boxplots of the weights assigned to the 10 base-learners conditional on their ranking in terms of RMSE

was compared to an equal weight combiner and a best learner (identified in the cross-validation procedure). We applied the algorithms to a dataset consisting of 511 river basins with 10 years of daily streamflow, precipitation and temperature. The machine learning algorithms modelled the relationship between next-day streamflow and daily streamflow, precipitation and temperature up to the present day.

The super ensemble learner improved over the performance of the linear regression benchmark by 20.06% in terms of the RMSE, while the respective improvements provided by the other ensemble learners were 19.21% (equal weight combiner) and 16.64% (best learner). The best base-learner was neural networks (16.73%), followed by extremely randomized trees (16.40%), XGBoost (15.92%), loess (15.36%), random forests (12.75%), polyMARS (12.36%), MARS (4.74%), lasso (0.11%) and support vector machines (– 0.45%). Exponential smoothing and ARIMA time series models improved over the linear regression benchmark by 13.89% and 8.77%, respectively.

All ensemble learners improved over the performance of the single base-learners. The performance of the super ensemble learner was somewhat higher than that of the equal weight combiner, which according to the "forecast combination puzzle" is a "*hard to beat in practice*" combination method. Consequently, we consider that the equal weight combiner can be effectively used as a benchmark for new combination methods, while super ensemble learning can result in better performances. One could claim that based on statistical tests, this difference may be insignificant; however, as mentioned by [6], these tests should be avoided when comparing forecasting methods, as they can be misleading.

We emphasize that our results are based on a big dataset comprising of 511 basins with 10 years of daily data each. Therefore, the reported relative improvements against the linear regression benchmark (i.e. in the range 0–20% in terms of RMSE, 0–35.5% in terms of MAE, 0–70% in terms of MEDAE and 0–21% in terms of r^2) can be considered realistic and can provide insightful guidance in understanding whether results reported in the literature (e.g. single case studies indicating improvements more than 50% in terms of RMSE) could be attributed to chance related to the use of small datasets. Assessments based on big datasets can emulate neutral comparison studies, i.e. studies focusing on comparison rather than aiming to promote a single method [12].

Future research could focus on improving the variable selection procedure and comparing the ensemble learner with optimized base-learners, while testing on different datasets could be also useful. Furthermore, pre-processing approaches based on clustering techniques, as well as frameworks formulated in a reinforcement learning context (e.g. [50, 51, 53]), including spatial information (e.g. [52]), may improve the performance of the proposed practical system. Besides machine learning, other techniques (e.g. graphs [38]) can also be tested in such problems. An additional topic of potential interest is to compare super ensemble learning with other combination methods, e.g. Bayesian model averaging or stacking using more flexible combiners.

Acknowledgments We thank the Associate Editor and two anonymous Reviewers for their constructive comments and suggestions, which helped us to substantially improve the manuscript.

Compliance with ethical standards

Conflicts of interest We declare no conflict of interest.

Appendix: Used software

We used the R programming language [68] to implement the algorithms of the study, and to report and visualize the results.

For data processing, we used the contributed R packages data.table [26], gdata [94], readr [98], stringi [35], stringr [96], tidyr [97].

The algorithms were implemented by using the contributed R packages earth [57], extraTrees [74, 75], forecast [43, 44], glmnet [33, 34], kernlab [46, 47], Matrix [8], nnet [71, 92], polspline [48], ranger [104, 105], SuperLearner [67], xgboost [22].

The performance metrics were computed by implementing the contributed R package mlr [9, 10].

Visualizations were made by using the contributed R package ggplot2 [95, 99].

Reports were produced by using the contributed R packages devtools [100], knitr [106–108], rmarkdown [5, 109].

References

1. Abrahart RJ, Anctil F, Coulibaly P, Dawson CW, Mount NJ, See LM, Shamseldin AY, Solomatine DP, Toth E, Wilby RL (2012) Two decades of anarchy? emerging themes and outstanding challenges for neural network river forecasting. Progress Phys Geogr Earth Environ 36(4):480–513. https://doi.org/10.1177/0309133312444943
2. Addor N, Newman AJ, Mizukami N, Clark MP (2017) Catchment attributes for large-sample studies. UCAR/NCAR, Boulder. https://doi.org/10.5065/D6G73C3Q
3. Addor N, Newman AJ, Mizukami N, Clark MP (2017) The CAMELS data set: catchment attributes and meteorology for large-sample studies. Hydrol Earth Syst Sci 21:5293–5313. https://doi.org/10.5194/hess-21-5293-2017
4. Afendras G, Markatou M (2019) Optimality of training/test size and resampling effectiveness in cross-validation. J Stat Plan Inference 199:286–301. https://doi.org/10.1016/j.jspi.2018.07.005
5. Allaire JJ, Xie Y, McPherson J, Luraschi J, Ushey K, Atkins A, Wickham H, Cheng J, Chang W, Iannone R (2019) rmarkdown: Dynamic documents for R. R package version 1.15. https://CRAN.R-project.org/package=rmarkdown
6. Armstrong JS (2007) Significance tests harm progress in forecasting. Int J Forecast 23(2):321–327. https://doi.org/10.1016/j.ijforecast.2007.03.004
7. Bates JM, Granger CWJ (1969) The combination of forecasts. J Oper Res Soc 20(4):451–468. https://doi.org/10.1057/jors.1969.103
8. Bates D, Maechler M (2019) Matrix: sparse and dense matrix classes and methods. R package version 1.2-17. https://CRAN.R-project.org/package=Matrix
9. Bischl B, Lang M, Kotthoff L, Schiffner J, Richter J, Studerus E, Casalicchio G, Jones ZM (2016) mlr: Machine learning in R. J Mach Learn Res 17(170):1–5
10. Bischl B, Lang M, Kotthoff L, Schiffner J, Richter J, Jones Z, Casalicchio G, Gallo M, Schratz P (2019) mlr: machine learning in R. R package version 2.15.0. https://CRAN.R-project.org/package=mlr
11. Blöschl G, Bierkens MFP, Chambel A, Cudennec C, Destouni G, Fiori A, Kirchner JW, McDonnell JJ, Savenije HHG, Sivapalan M et al (2019) Twenty-three unsolved problems in hydrology (UPH)—a community perspective. Hydrol Sci J 64(10):1141–1158. https://doi.org/10.1080/02626667.2019.1620507
12. Boulesteix AL, Binder H, Abrahamowicz M, Sauerbrei W, for the Simulation Panel of the STRATOS Initiative (2018) On the necessity and design of studies comparing statistical methods. Biom J 60(1):216–218. https://doi.org/10.1002/bimj.201700129
13. Bowden GJ, Dandy GC, Maier HR (2005) Input determination for neural network models in water resources applications. Part 1—background and methodology. J Hydrol 301(1–4):75–92. https://doi.org/10.1016/j.jhydrol.2004.06.021
14. Box GEP, Jenkins GM (1970) Time series analysis: forecasting and control. Holden-Day Inc., San Francisco
15. Box GEP, Jenkins GM, Reinsel GC, Ljung GM (2015) Time series analysis: forecasting and control. Wiley, Hoboken. ISBN 978-1-118-67502-1
16. Breiman L (1996) Stacked regressions. Mach Learn 24(1):49–64. https://doi.org/10.1007/BF00117832
17. Breiman L (2001) Random forests. Mach Learn 45(1):5–32. https://doi.org/10.1023/a:1010933404324
18. Breiman L (2001) Statistical modeling: the two cultures. Stat Sci 16(3):199–231. https://doi.org/10.1214/ss/1009213726
19. Brown RG (1959) Statistical forecasting for inventory control. McGraw-Hill Book Co., New York
20. Carlson RF, MacCormick AJA, Watts DG (1970) Application of linear random models to four annual streamflow series. Water Resour Res 6(4):1070–1078. https://doi.org/10.1029/wr006i004p01070
21. Chen T, Guestrin C (2016) XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. pp 785–794. https://doi.org/10.1145/2939672.2939785
22. Chen T, He T, Benesty M, Khotilovich V, Tang Y, Cho H, Chen K, Mitchell R, Cano I, Zhou T, Li M, Xie J, Lin M, Geng Y, Li Y (2019) xgboost: Extreme gradient boosting. R package version 0.90.0.2. https://CRAN.R-project.org/package=xgboost
23. Claeskens G, Magnus JR, Vasnev AL, Wang W (2016) The forecast combination puzzle: a simple theoretical explanation. Int J Forecast 32(3):754–762. https://doi.org/10.1016/j.ijforecast.2015.12.005

24. Cleveland WS, Grosse E, Shyu WM (1992) Local regression models. In: Chambers JM, Hastie TJ (eds) Statistical models in S. Chapman and Hall, London, pp 309–376
25. Dawson CW, Wilby RL (2001) Hydrological modelling using artificial neural networks. Progress Phys Geogr Earth Environ 25(1):80–108. https://doi.org/10.1177/030913330102500104
26. Dowle M, Srinivasan A (2019) data.table: Extension of 'data.-frame'. R package version 1.12.2. https://CRAN.R-project.org/package=data.table
27. Dyer TGJ (1977) On the application of some stochastic models to precipitation forecasting. Q J R Meteorol Soc 103(435):177–189. https://doi.org/10.1002/qj.49710343512
28. Efron B, Hastie T (2016) Computer age statistical inference. Cambridge University Press, New York. ISBN 9781107149892
29. Fletcher D (2018) Model averaging. Springer, Berlin. https://doi.org/10.1007/978-3-662-58541-2
30. Friedman JH (1991) Multivariate adaptive regression splines. Ann Stat 19(1):1–67. https://doi.org/10.1214/aos/1176347963
31. Friedman JH (1993) Fast MARS. Stanford University, Department of Statistics. Technical Report 110. https://statistics.stanford.edu/sites/g/files/sbiybj6031/f/LCS%20110.pdf
32. Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann Stat 29(5):1189–1232. https://doi.org/10.1214/aos/1013203451
33. Friedman JH, Hastie T, Tibshirani R (2010) Regularization paths for generalized linear models via coordinate descent. J Stat Softw 33(1):1. https://doi.org/10.18637/jss.v033.i01
34. Friedman J, Hastie T, Tibshirani R, Simon N, Narasimhan B (2019) glmnet: Lasso and elastic-net regularized generalized linear models. R package version 2.0-18. https://CRAN.R-project.org/package=glmnet
35. Gagolewski M (2019) stringi: Character string processing facilities. R package version 1.4.3. https://CRAN.R-project.org/package=stringi
36. Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Mach Learn 63(1):3–42. https://doi.org/10.1007/s10994-006-6226-1
37. Granger CWJ, Ramanathan R (1984) Improved methods of combining forecasts. J Forecast 3(2):197–204. https://doi.org/10.1002/for.3980030207
38. Hao F, Park DS, Li S, Lee HM (2016) Mining λ-maximal cliques from a fuzzy graph. Sustainability 8(6):553. https://doi.org/10.3390/su8060553
39. Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning. Springer, New York. https://doi.org/10.1007/978-0-387-84858-7
40. Heinze G, Wallisch C, Dunkler D (2018) Variable selection—a review and recommendations for the practicing statistician. Biom J 60(3):431–449. https://doi.org/10.1002/bimj.201700067
41. Holt CC (2004) Forecasting seasonals and trends by exponentially weighted moving averages. Int J Forecast 20(1):5–10. https://doi.org/10.1016/j.ijforecast.2003.09.015
42. Hyndman RJ, Koehler AB (2006) Another look at measures of forecast accuracy. Int J Forecast 22(4):679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001
43. Hyndman RJ, Khandakar Y (2008) Automatic time series forecasting: the forecast package for R. J Stat Softw 27(3):1–22. https://doi.org/10.18637/jss.v027.i03
44. Hyndman R, Athanasopoulos G, Bergmeir C, Caceres G, Chhay L, O'Hara-Wild M, Petropoulos F, Razbash S, Wang E, Yasmeen F (2020) forecast: Forecasting functions for time series and linear models. R package version 8.11. https://CRAN.R-project.org/package=forecast
45. James G, Witten D, Hastie T, Tibshirani R (2013) An introduction to statistical learning. Springer, New York. https://doi.org/10.1007/978-1-4614-7138-7
46. Karatzoglou A, Smola A, Hornik K, Zeileis A (2004) kernlab—An S4 package for kernel methods in R. J Stat Softw 11(9):1–20. https://doi.org/10.18637/jss.v011.i09
47. Karatzoglou A, Smola A, Hornik K (2018) kernlab: Kernel-based machine learning lab. R package version 0.9-27. https://CRAN.R-project.org/package=kernlab
48. Kooperberg C (2019) polspline: Polynomial spline routines. R package version 1.1.15. https://CRAN.R-project.org/package=polspline
49. Kooperberg C, Bose S, Stone CJ (1997) Polychotomous regression. J Am Stat Assoc 92(437):117–127. https://doi.org/10.1080/01621459.1997.10473608
50. Korda N, Szorenyi B, Li S (2016) Distributed clustering of linear bandits in peer to peer networks. Proc Mach Learn Res 48:1301–1309
51. Li S (2016) The Art of clustering bandits. Dissertation, Università degli Studi dell'Insubria. http://insubriaspace.cineca.it/handle/10277/729
52. Li S, Hao F, Li M, Kim HC (2013) Medicine rating prediction and recommendation in mobile social networks. In: Park JJH, Arabnia HR, Kim C, Shi W, Gil JM (eds) Grid and pervasive computing. Springer, Berlin, pp 216–223. https://doi.org/10.1007/978-3-642-38027-3_23
53. Li S, Karatzoglou A, Gentile C (2016) Collaborative filtering bandits. SIGIR '16. In: Proceedings of the 39th international ACM SIGIR conference on research and development in information retrieval. pp. 539–548. https://doi.org/10.1145/2911451.2911548
54. Maier HR, Jain A, Dandy GC, Sudheer KP (2010) Methods used for the development of neural networks for the prediction of water resource variables in river systems: current status and future directions. Environ Model Softw 25(8):891–909. https://doi.org/10.1016/j.envsoft.2010.02.003
55. May RJ, Maier HR, Dandy GC, Gayani Fernando TMC (2008) Non-linear variable selection for artificial neural networks using partial mutual information. Environ Model Softw 23(10–11):1312–1326. https://doi.org/10.1016/j.envsoft.2008.03.007
56. Mayr A, Binder H, Gefeller O, Schmid M (2014) The evolution of boosting algorithms. Methods Inf Med 53(06):419–427. https://doi.org/10.3414/me13-01-0122
57. Milborrow S (2019) earth: Multivariate adaptive regression splines. R package version 5.1.1. https://CRAN.R-project.org/package=earth
58. Natekin A, Knoll A (2013) Gradient boosting machines, a tutorial. Front Neurorobotics 7:21. https://doi.org/10.3389/fnbot.2013.00021
59. Newman AJ, Sampson K, Clark MP, Bock A, Viger RJ, Blodgett D (2014) A large-sample watershed-scale hydrometeorological dataset for the contiguous USA. UCAR/NCAR, Boulder. https://doi.org/10.5065/D6MW2F4D
60. Newman AJ, Clark MP, Sampson K, Wood A, Hay LE, Bock A, Viger RJ, Blodgett D, Brekke L, Arnold JR, Hopson T, Duan Q (2015) Development of a large-sample watershed-scale hydrometeorological data set for the contiguous USA: data set characteristics and assessment of regional variability in hydrologic model performance. Hydrol Earth Syst Sci 19:209–223. https://doi.org/10.5194/hess-19-209-2015
61. Newman AJ, Mizukami N, Clark MP, Wood AW, Nijssen B, Nearing G (2017) Benchmarking of a physically based hydrologic model. J Hydrometeorol 18:2215–2225. https://doi.org/10.1175/jhm-d-16-0284.1
62. Papacharalampous G, Tyralis H, Koutsoyiannis D (2018) One-step ahead forecasting of geophysical processes within a purely statistical framework. Geosci Lett 5(12). https://doi.org/10.1186/s40562-018-0111-1

63. Papacharalampous G, Tyralis H, Koutsoyiannis D (2018) Predictability of monthly temperature and precipitation using automatic time series forecasting methods. Acta Geophys 66(4):807–831. https://doi.org/10.1007/s11600-018-0120-7
64. Papacharalampous G, Tyralis H, Koutsoyiannis D (2018) Univariate time series forecasting of temperature and precipitation with a focus on machine learning algorithms: a multiple-case study from Greece. Water Resour Manag 32(15):5207–5239. https://doi.org/10.1007/s11269-018-2155-6
65. Papacharalampous G, Tyralis H, Koutsoyiannis D (2019) Comparison of stochastic and machine learning methods for multi-step ahead forecasting of hydrological processes. Stoch Environ Res Risk Assess 33(2):481–514. https://doi.org/10.1007/s00477-018-1638-6
66. Papacharalampous G, Tyralis H, Langousis A, Jayawardena AW, Sivakumar B, Mamassis N, Montanari A, Koutsoyiannis D (2019) Probabilistic hydrological post-processing at scale: why and how to apply machine-learning quantile regression algorithms. Water 11(10):2126. https://doi.org/10.3390/w11102126
67. Polley E, LeDell E, Kennedy C, van der Laan MJ (2019) SuperLearner: super learner prediction. R package version 2.0-25. https://CRAN.R-project.org/package=SuperLearner
68. R Core Team (2019) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/
69. Raghavendra S, Deka PC (2014) Support vector machine applications in the field of hydrology: a review. Appl Soft Comput 19:372–386. https://doi.org/10.1016/j.asoc.2014.02.002
70. Ripley BD (1996) Pattern recognition and neural networks. Cambridge University Press, Cambridge. https://doi.org/10.1017/cbo9780511812651
71. Ripley BD (2016) nnet: Feed-forward neural networks and multinomial log-linear models. R package version 7.3-12. https://CRAN.R-project.org/package=nnet
72. Sagi O, Rokach L (2018) Ensemble learning: a survey. Wiley Interdiscip Rev Data Min Knowl Discov 8(4):e1249. https://doi.org/10.1002/widm.1249
73. Shmueli G (2010) To explain or to predict? Stat Sci 25(3):289–310. https://doi.org/10.1214/10-sts330
74. Simm J, de Abril IM (2014) extraTrees: Extremely randomized trees (ExtraTrees) method for classification and regression. R package version 1.0.5. https://CRAN.R-project.org/package=extraTrees
75. Simm J, de Abril IM, Sugiyama M (2014) Tree-based ensemble multi-task learning method for classification and regression. IEICE Trans Inf Syst E97.D(6):1677–1681. https://doi.org/10.1587/transinf.e97.d.1677
76. Smith J, Wallis KF (2009) A simple explanation of the forecast combination puzzle. Oxford Bull Econ Stat 71(3):331–355. https://doi.org/10.1111/j.1468-0084.2008.00541.x
77. Smola AJ, Schölkopf B (2004) A tutorial on support vector regression. Stat Comput 14(3):199–222. https://doi.org/10.1023/b:stco.0000035301.49549.88
78. Solomatine DP, Ostfeld A (2008) Data-driven modelling: some past experiences and new approaches. J Hydroinformatics 10(1):3–22. https://doi.org/10.2166/hydro.2008.015
79. Stock JH, Watson MW (2004) Combination forecasts of output growth in a seven-country data set. J Forecast 23(6):405–430. https://doi.org/10.1002/for.928
80. Stone CJ, Hansen MH, Kooperberg C, Truong YK (1997) Polynomial splines and their tensor products in extended linear modeling. Annal Stat 25(4):1371–1470. https://doi.org/10.1214/aos/1031594728
81. Thornton PE, Thornton MM, Mayer BW, Wilhelmi N, Wei Y, Devarakonda R, Cook RB (2014) Daymet: Daily surface weather data on a 1-km grid for North America, Version 2. ORNL DAAC, Oak Ridge, Tennessee, USA. Date accessed: 2016/01/20. https://doi.org/10.3334/ORNLDAAC/1219
82. Tibshirani R (1996) Regression shrinkage and selection via the lasso. J Roy Stat Soc B 58(1):267–288. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x
83. Timmermann A (2006) Chapter 4 forecast combinations. In: Arrow KJ, Intriligator MD (eds) Handbook of economic forecasting. Springer, Berlin, pp 135–196. https://doi.org/10.1016/s1574-0706(05)01004-9
84. Tyralis H, Papacharalampous G (2017) Variable selection in time series forecasting using random forests. Algorithms 10(4):114. https://doi.org/10.3390/a10040114
85. Tyralis H, Papacharalampous G, Langousis A (2019) A brief review of random forests for water scientists and practitioners and their recent history in water resources. Water 11(5):910. https://doi.org/10.3390/w11050910
86. Tyralis H, Papacharalampous G, Tantanee S (2019) How to explain and predict the shape parameter of the generalized extreme value distribution of streamflow extremes using a big dataset. J Hydrol 574:628–645. https://doi.org/10.1016/j.jhydrol.2019.04.070
87. Tyralis H, Papacharalampous G, Burnetas A, Langousis A (2019) Hydrological post-processing using stacked generalization of quantile regression algorithms: large-scale application over CONUS. J Hydrol 577:123957. https://doi.org/10.1016/j.jhydrol.2019.123957
88. van der Laan MJ, Rose S (2011) Targeted learning. Springer, New York. https://doi.org/10.1007/978-1-4419-9782-1
89. van der Laan MJ, Rose S (2018) Targeted learning in data science. Springer, Berlin. https://doi.org/10.1007/978-3-319-65304-4
90. Van der Laan MJ, Polley EC, Hubbard AE (2007) Super learner. Stat Appl Genet Mol Biol. https://doi.org/10.2202/1544-6115.1309
91. Vapnik VN (1995) The nature of statistical learning theory. Springer, New York. https://doi.org/10.1007/978-1-4757-2440-0
92. Venables WN, Ripley BD (2002) Modern applied statistics with S. Springer, New York. https://doi.org/10.1007/978-0-387-21706-2
93. Wallis KF (2011) Combining forecasts—forty years later. Appl Financ Econ 21(1–2):33–41. https://doi.org/10.1080/09603107.2011.523179
94. Warnes GR, Bolker B, Gorjanc G, Grothendieck G, Korosec A, Lumley T, MacQueen D, Magnusson A, Rogers J (2017) gdata: Various R programming tools for data manipulation. R package version 2.18.0. https://CRAN.R-project.org/package=gdata
95. Wickham H (2016) ggplot2. Springer, New York. https://doi.org/10.1007/978-0-387-98141-3
96. Wickham H (2019) stringr: Simple, consistent wrappers for common string operations. R package version 1.4.0. https://CRAN.R-project.org/package=stringr
97. Wickham H, Henry L (2019) tidyr: Easily tidy data with 'spread()' and 'gather()' functions. R package version 0.8.3. https://CRAN.R-project.org/package=tidyr
98. Wickham H, Hester J, Francois R (2018) readr: Read rectangular text data. R package version 1.3.1. https://CRAN.R-project.org/package=readr
99. Wickham H, Chang W, Henry L, Pedersen TL, Takahashi K, Wilke C, Woo K, Yutani H (2019) ggplot2: Create elegant data visualisations using the grammar of graphics. R package version 3.2.1. https://CRAN.R-project.org/package=ggplot2
100. Wickham H, Hester J, Chang W (2019) devtools: Tools to make developing R packages easier. R package version 2.1.0. https://CRAN.R-project.org/package=devtools
101. Winters PR (1960) Forecasting sales by exponentially weighted moving averages. Manag Forecast 6(3):324–342. https://doi.org/10.1287/mnsc.6.3.324

102. Wolpert DH (1992) Stacked generalization. Neural Netw 5(2):241–259. https://doi.org/10.1016/s0893-6080(05)80023-1
103. Wolpert DH (1996) The lack of a priori distinctions between learning algorithms. Neural Comput 8(7):1341–1390. https://doi.org/10.1162/neco.1996.8.7.1341
104. Wright MN (2019) ranger: A Fast Implementation of random forests. R package version 0.11.2. https://CRAN.R-project.org/package=ranger
105. Wright MN, Ziegler A (2017) ranger: A fast implementation of random forests for high dimensional data in C ++ and R. J Stat Softw. https://doi.org/10.18637/jss.v077.i01
106. Xie Y (2014) knitr: A Comprehensive tool for reproducible research in R. In: Stodden V, Leisch F, Peng RD (eds) Implementing reproducible computational research. Chapman and Hall/CRC, London
107. Xie Y (2015) Dynamic documents with R and knitr, 2nd edn. Chapman and Hall/CRC, London
108. Xie Y (2019) knitr: A general-purpose package for dynamic report generation in R. R package version 1.24. https://CRAN.R-project.org/package=knitr
109. Xie Y, Allaire JJ, Grolemund G (2018) R Markdown, 1st edn. Chapman and Hall/CRC, London. ISBN:13: 978-1138359338