Boosting algorithms for projecting streamflow in the Lower Godavari Basin for different climate change scenarios

Article *in* Water Science & Technology · January 2024

DOI: 10.2166/wst.2024.011

CITATIONS
0

READS
243

5 authors, including:

Bhavesh Rahul Mishra
BITS Pilani, Hyderabad
3 PUBLICATIONS **4** CITATIONS
SEE PROFILE

Rishith Kumar Vogeti
BITS Pilani, Hyderabad
5 PUBLICATIONS **7** CITATIONS
SEE PROFILE

Rahul Jauhari
Birla Institute of Technology and Science Pilani
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

D Nagesh Kumar
Indian Institute of Science
267 PUBLICATIONS **7,835** CITATIONS
SEE PROFILE

Water Science & Technology

Water Science & Technology Vol 89 No 3, 613 doi: 10.2166/wst.2024.011

Boosting algorithms for projecting streamflow in the Lower Godavari Basin for different climate change scenarios

Bhavesh Rahul Mishra[a], Rishith Kumar Vogeti[b], Rahul Jauhari[c], K. Srinivasa Raju[b,*] and D. Nagesh Kumar[d]

[a] Department of Electrical and Electronics Engineering, BITS Pilani Hyderabad Campus, Hyderabad, India
[b] Department of Civil Engineering, BITS Pilani Hyderabad Campus, Hyderabad, India
[c] Department of Computer Science and Information Systems, BITS Pilani Hyderabad Campus, Hyderabad, India
[d] Department of Civil Engineering, Indian Institute of Science, Bangalore, India
*Corresponding author. E-mail: ksraju@hyderabad.bits-pilani.ac.in

ABSTRACT

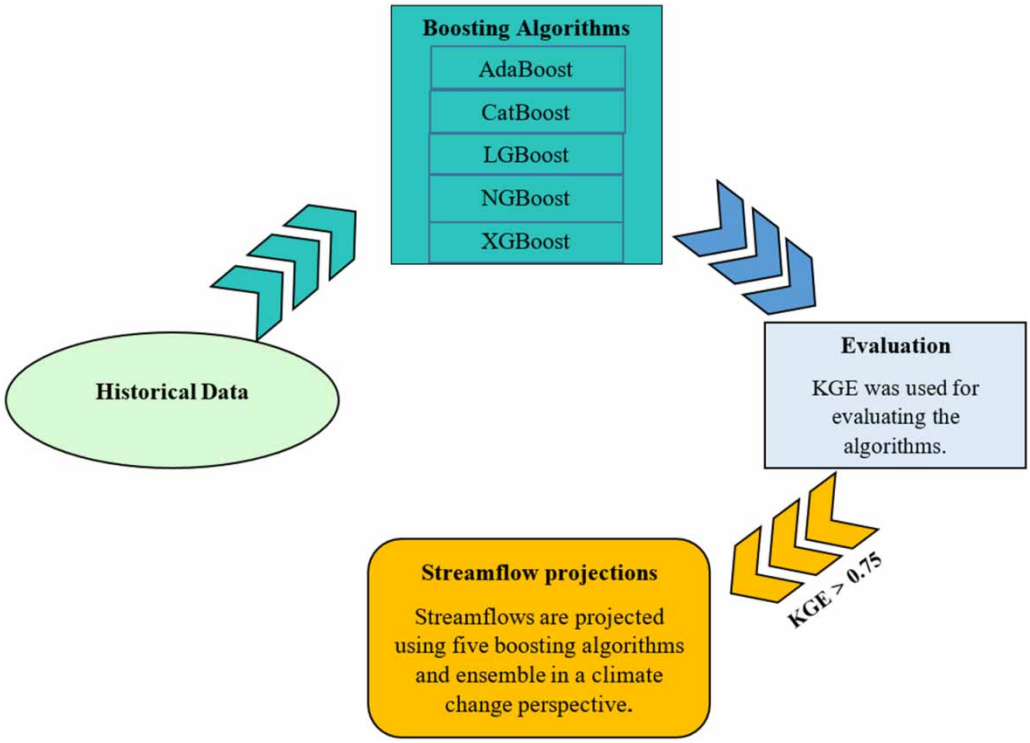
The present study investigates the ability of five boosting algorithms, namely Adaptive Boosting (AdaBoost), Categorical Boosting (CatBoost), Light Gradient Boosting (LGBoost), Natural Gradient Boosting (NGBoost), and eXtreme Gradient Boosting (XGBoost) for simulating streamflow in the Lower Godavari Basin, India. Monthly rainfall, temperatures, and streamflow from 1982 to 2020 were used for training and testing. Kling Gupta Efficiency (KGE) was deployed to assess the ability of the boosting algorithms. It was observed that all the boosting algorithms had shown good simulating ability, having KGE values of AdaBoost (0.87, 0.85), CatBoost (0.90, 0.78), LGBoost (0.95, 0.93), NGBoost (0.95, 0.95), and XGBoost (0.91, 0.90), respectively, in training and testing. Thus, all the algorithms were used for projecting streamflow in a climate change perspective for the short-term projections (2025–2050) and long-term projections (2051–2075) for four Shared Socioeconomic Pathways (SSPs). The highest streamflow for all four SSPs in the case of NGBoost is more than the historical scenario (9382 m^3/s), whereas vice-versa for the remaining four. The effect of ensembling the outputs of five algorithms is also studied and compared with that of individual algorithms.

Key words: boosting, KGE, Lower Godavari Basin, SSP, streamflow

HIGHLIGHTS

- All the boosting algorithms have shown good simulating ability and were used in projecting streamflows for short-term projections (2025–2050) and long-term projections (2051–2075).
- The highest streamflow for all four SSPs in the case of NGBoost is more than the historical scenario (9382 m^3/s), whereas vice-versa for the remaining four.
- The effect of ensembling the outputs of five algorithms is also studied and compared with that of individual algorithms.

GRAPHICAL ABSTRACT



1. INTRODUCTION

Streamflow forecasts in the climate change framework are critical in basin planning and management decisions (Liu *et al.* 2020). They deliver the essential information that enables decision-makers, water managers, and communities to make proactive and informed decisions about available water resources (Speight *et al.* 2021). It also helps mitigate the risks associated with water-related hazards, prioritize flood-prone areas by detecting early peak flows (Oborie & Rowland 2023), and contribute to sustainable water management (Marques *et al.* 2022). Simulating streamflow is complex due to the involvement of several hydrological processes (Guo *et al.* 2021). In this regard, semi-distributed, distributed, process-based deterministic, and stochastic hydrological models (Nesru 2023) were employed to simulate these complex hydrological processes. However, high spatio-temporal dynamics (Ibrahim *et al.* 2021; Markhali *et al.* 2022), the necessity of advanced calibration procedures (Asgari *et al.* 2022; Jin 2022), and the accurate estimation of initial hydrologic conditions (Manikanta & Umamahesh 2023) pose challenges to the simulation efficacy of these models.

Machine learning algorithms were explored as promising alternatives to overcome these challenges as they do not entail the details of underlying physical mechanisms (Kim *et al.* 2021). However, selecting the most suitable machine learning algorithm for calculating streamflow is also challenging. Some researchers made an effort to analyze various machine learning algorithms, including boosting algorithms, in terms of improved accuracy, less prone to overfitting, the ability to handle imbalanced and noisy data, and versatility across domains. Accordingly, boosting algorithms are proposed in the present study for simulating streamflow in the Lower Godavari Basin, India. A brief but relevant review of the literature is given in the following.

Zounemat-Kermani *et al.* (2021) reviewed resampling ensemble methods and stacking in hydrology and their advantages over individual models. Troin *et al.* (2021) extensively reviewed ensemble forecasting related to hydrology, including limitations, challenges, and the road ahead. Başağaoğlu *et al.* (2022) reviewed Interpretable and eXplainable Artificial Intelligence models in a hydroclimatic context.

Fan *et al.* (2019) employed Light Gradient Boosting (LGBoost), M5 Tree, Random Forest, and four empirical models to compute evapotranspiration in the Jiangxi province of China. They employed Root Mean Square Error. LGBoost was superior to the remaining models. Machine learning models were suggested as an alternative to the empirical models.

McCarty *et al.* (2020) employed Random Forest, Support Vector Machine, and LGBoost in the Rhine–Ruhr Functional Urban Region. They used metrics such as accuracy, and Kappa. LGBoost obtained the highest overall accuracy, followed by Random Forest and Support Vector Machine. Ni *et al.* (2020) developed a Gaussian mixture model- XGBoost, XGBoost, and Support Vector Machine for streamflow forecasting in the Yangtze River Basin. Gaussian mixture model- XGBoost proved to be the best.

Ma *et al.* (2021) employed XGBoost and the Least Square-Support Vector Machine to develop a flash flood risk map for Yunnan Province, China. They considered five performance indices. XGBoost performed exceptionally well. Hu *et al.* (2021) combined a physics-based model with XGBoost to forecast the occurrence and severity of empirical orthogonal functions runoff in the Great Lakes region. A hybrid approach is found to perform well. Khosravi *et al.* (2021) applied M5 Prime, M5 Rule, six hybrid bagging algorithms, Autoregressive integrated moving average, and Random Forest for streamflow prediction of the Taleghan catchment. Different input combinations were explored for streamflow prediction. They employed six performance indicators for evaluating the efficacy of models. Hybrid bagging algorithms-M5 Prime outperformed the other. They also compared the previous studies on the three hydrological models. They proved that Artificial Intelligence models are superior in performance. Adnan *et al.* (2021) combined a locally weighted learning technique with various Machine learning algorithms. They applied ensemble modeling to the Jhelum Catchment, Pakistan, for streamflow prediction. Ensemble locally weighted learning-additive regression is found to be the best. Pham *et al.* (2021) employed Adaptive Boosting (AdaBoost), Boosted Generalized Linear Models, XGBoost, and Deep Boosting to identify flood hazard areas in the Talar watershed, Mazandaran province, Iran. The area under the curve is the indicator. These values of Deep Boosting are slightly higher than the remaining three. In general, boosting algorithms performed well.

Shen *et al.* (2022) employed Natural Gradient Boosting (NGBoost) with Tree-structured Parzen Estimator optimization in the upper Yangtze River. They used metrics to evaluate the model performance and discussed high-flow cases, etc. Yiming *et al.* (2022) applied AdaBoost, Random Forest, and Support Vector Machine to Wangjiaba and Bengbu stations in Huai River Basin, China, for streamflow prediction. Three machine learning algorithms performed well, but Random Forest and Support Vector Machines had low prediction rates on high discharge values. AdaBoost performed better than Random Forest and Support Vector Machine. Khan *et al.* (2023) applied AdaBoost, Random Forest, Gradient Boosting, and K-Nearest Neighbors to simulate streamflow in the Hunza River basin, Pakistan. AdaBoost has surpassed all the other models with a more remarkable simulation ability. The study also recommended that AdaBoost can be further explored for reliable forecasting from a climate change perspective. Szczepanek (2022) employed XGBoost, LGBoost, Categorical Boosting (CatBoost), Multiple linear regression, and Random Forest to forecast daily streamflow in the mountainous Skawa River catchment. LGBoost is the best among the three, followed by CatBoost.

Dastour & Hassan (2023) employed a number of models for the Athabasca River Basin. Gradient boosting with bagging regressor performed well in seven out of 26 streamflow situations. They also developed a mechanism to handle the missing data in 26 stations. Kilinc *et al.* (2023) employed the CatBoost-Genetic Algorithm, Long Short-Term Memory (LSTM), CatBoost, and Linear Regression in Sakarya Basin, Turkey. CatBoost-Genetic Algorithm was found to capture streamflow prediction effectively. Kumar *et al.* (2023) employed several models to predict the river inflow, the Garudeshwar watershed, in India. CatBoost performed remarkably compared to other chosen methods. They also discussed the limitations of the techniques. In peninsular India, Sharma *et al.* (2023) applied four models, for Suvarna, Aghanashini, and Kunderu River Basins. Different models performed well for each river basin. However, all four models are satisfactory for implementation in a watershed. Xu *et al.* (2023) employed LGBoost to Haidian Island, China. A Personal Computer Storm Water Management Model provides data for the LGBoost. Hyperparameter tuning procedures were used to obtain optimum parameters. Benchmarking models used for comparison are Random Forest, XGBoost, and K-Nearest Neighbors. LGBoost is found to be the best among the chosen models. Rathnayake *et al.* (2023) employed XGBoost, LGBoost, CatBoost, and Adaptive Network-based Fuzzy Inference System to simulate river flows of Malwathu Oya, Sri Lanka. CatBoost has an edge over others.

Several researchers considered Coupled Model Intercomparison Project Phase-6 (CMIP6)-Global Climate Models (GCMs) framework and Shared Socioeconomic Pathways (SSPs) to consider the influence of climate change on streamflow using Machine learning algorithms. Singh *et al.* (2023) applied six Machine learning models, to simulate streamflow in Sutlej River Basin, India. It was found that Random Forest performed better. It is further used in projecting streamflow using a mean ensemble of six GCMs for SSP245 and SSP585. It was concluded that the mean annual streamflow in the monsoon would increase in both scenarios and decrease in pre-monsoon and winter in 2050 and 2080 s. Zhang *et al.* (2023) employed a Support Vector Machine, Artificial Neural Network, and Multiple Variable Regression for simulating streamflows across

river basins, namely, Mackenzie, Lena, Ob & Yenisei. It was found that the Support Vector Machine demonstrated superior performance and can be further used in projecting streamflows. The mean ensemble of five GCMs in 2021–2100 for SSP245 and SSP585 were considered. It is to minimize the biases associated with individual models. The study reveals minimal streamflow variation in the Ob basin, whereas increment is observed in the remaining basins. Adib & Harun (2022) used a Support Vector Regression and Random Forest ensemble to simulate streamflow in Kurau Basin, Malaysia. They have shown good simulating ability. Further, an ensemble of five GCMs for SSP126, SSP245, and SSP585 were explored to predict streamflow in 2021–2080. Streamflow significantly decreased in April and May for all three SSPs.

Yang *et al.* (2023) combined the Soil Water Assessment Tool (SWAT) and Bi-directional-LSTM and developed two hybrid models, SWAT-T-LSTM and SWAT-D-LSTM, for simulating daily streamflow in Kelantan River Basin, Malaysia. SWAT-T-LSTM has performed ahead of the other three. Further, this hybrid was used in projecting streamflows. An ensemble of five GCMs using SSP126, SSP245, and SSP585 were considered. It was concluded that average and extreme streamflow would increase in the northwest monsoon.

Graf & Vyshnevskyi (2022) employed XGBoost in climate change perspective for projecting streamflows in Prut, Styr, and Sula catchments of Ukraine using CMIP5-Representative Concentration Pathways-2.6 and 4.5. Decreased streamflows in Prut and Styr, whereas increased Sula catchments were observed.

The existing literature review reveals several research gaps in the context of streamflow projections. First, there is a lack of exclusive comparisons among boosting algorithms to identify the most suitable one. Second, limited studies have utilized an ensemble of multiple CMIP6-GCMs to generate future climate data within a boosting framework. Last, ensemble streamflow projections are seldom studied, limiting a comprehensive understanding of variability and uncertainty. Accordingly, the following objectives are formulated for the Lower Godavari Basin, India.

1. To study the suitability of boosting algorithms, AdaBoost, CatBoost, LGBoost, NGBoost, and XGBoost for simulating streamflow.
2. To forecast streamflow utilizing a competent algorithm(s) from a climate change framework using an ensembling of CMIP6-GCMs and four SSPs.
3. To explore the ensembling of projected streamflow.

The study area was chosen due to the complex hydrologic process resulting from diverse land cover, soil, highly varying meteorological conditions, and its proximity to the Bay of Bengal.

2. STUDY AREA AND DATA

The Lower Godavari Basin is located at the latitude 17° 00′ to 19° 00′ N and longitude 80° 00′ to 83° 4′ E. Figure 1 presents a digital elevation model of the basin. It shares boundaries with five states. It has a catchment area of 39,180 km^2 with the highest elevation of 1,642 m. The basin receives an annual rainfall of 1,096 mm, whereas evaporation is from 1,401 to 2,606.40 mm (Jhajharia *et al.* 2021; Sarkar 2022). Primary land is mainly comprised of cultivated land and forests.

The basin is expected to meet various demands, including drinking and industrial needs in Visakhapatnam, generation of hydropower, domestic water supply for the number of villages, water requirements for Odisha state, and more similar minor requirements (Polavaram Project Authority 2021). Accordingly, accurate streamflow estimation is crucial for water security, keeping the effect of climate change on runoff (Hengade *et al.* 2018). Furthermore, with the growing demands in every sector, it is felt that the present study will be helpful to policymakers. For example, the increase in demand due to population growth and other factors is to be understood. Considering these factors, the present study is proposed to analyze streamflow projections.

The meteorological data for 52 grid locations of the study area from 1982 to 2020 is from the India Meteorological Department, whereas observed streamflow is from the Central Water Commission. Future meteorological data for 2025–2075 were derived from 13 CMIP6-GCMs for four SSPs, 126, 245, 370, and 585 (Mishra *et al.* 2020).

3. METHODOLOGY

The philosophy of boosting algorithms is to build an ensemble of decision trees to minimize the loss function. They learn iteratively to solve various classification and regression problems. Employed algorithms here are based on natural Gradient (NGBoost), level-wise (CatBoost and AdaBoost), leaf-wise (LGBoost), and depth-wise (XGBoost) and are explained as follows:

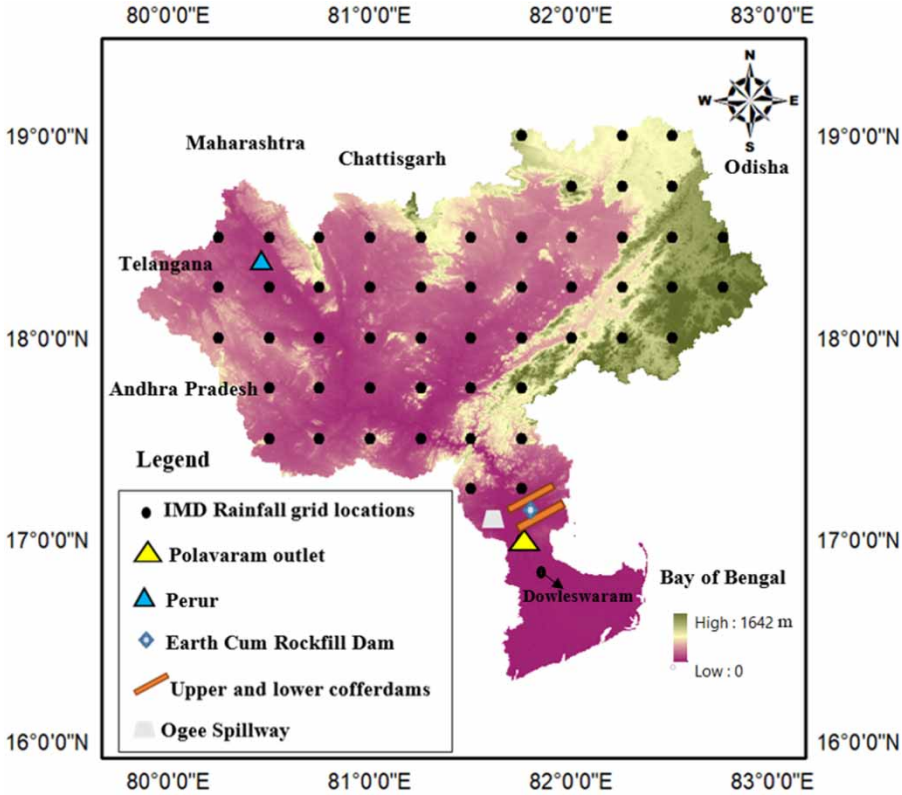


Figure 1 | Digital elevation model of the Lower Godavari Basin.

3.1. AdaBoost

AdaBoost (Wang *et al.* 2019) combines multiple weak learners to create a strong regression model. It focuses on instances with high loss function to improve the overall performance. Input information comprises (x_i, y_i) which represents the input feature and target variables at ith instance, respectively. First, a base learner $h_0(x_i)$ is used as an initial guess. In continuation, multiple base learners $h_t(x_i)$ are iterated, and the weighted error at iteration t (ε_t) is given by (Equation (1)):

$$\varepsilon_t = \sum_{1}^{t} w_i^{(t)} \cdot h_t(x_i) \tag{1}$$

The weight updation for successive iteration $(t+1)$ is given by (Equation (2)):

$$w_i^{(t+1)} = w_i^{(t)} e^{\alpha_t . h_t(x_i)} \tag{2}$$

where $w_i^{(t)}$ and $w_i^{(t+1)}$ are the weights assigned at (t) and $(t+1)$ iterations, respectively; α_t refers to the contribution factor of weak learner for t iterations and are expressed as follows (Equation (3)):

$$\alpha_t = \frac{1}{2} ln \frac{1-\varepsilon_t}{\varepsilon_t} \tag{3}$$

The $h_t(x_i)$, ε_t, $w_i^{(t+1),}$ and α_t are continuously updated until they meet the desired criteria. The final output $(H_t(x_i))$ is obtained by combining several base learners and is expressed as (Equation (4)):

$$H_t(x_i) = \sum_{1}^{t} \alpha_t \cdot h_t(x_i) \tag{4}$$

3.2. CatBoost

The mathematical expression of CatBoost (Mehraein *et al.* 2022) is almost similar to that of AdaBoost. Here, the decision tree is developed based on a symmetric-wise strategy (refer to Figure 2) and specifically designed to handle categorical features more efficiently. It uses an ordered boosting technique and combines target encoding and gradient-based splitting to handle categorical features.

3.3. LGBoost

Unlike another traditional decision tree, *LGBoost* (Fan et al. 2019) *does not use bootstrap sampling to subsample the data*. It uses a leaf-wise strategy, which allows it to process the data faster than traditional level-based methods. A leaf structure that minimizes the negative gradient of loss function is highly preferred (Figure 3).

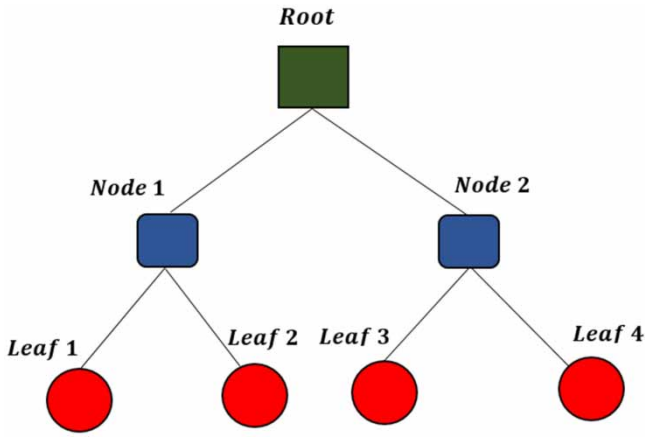


Figure 2 | Tree formation in CatBoost.

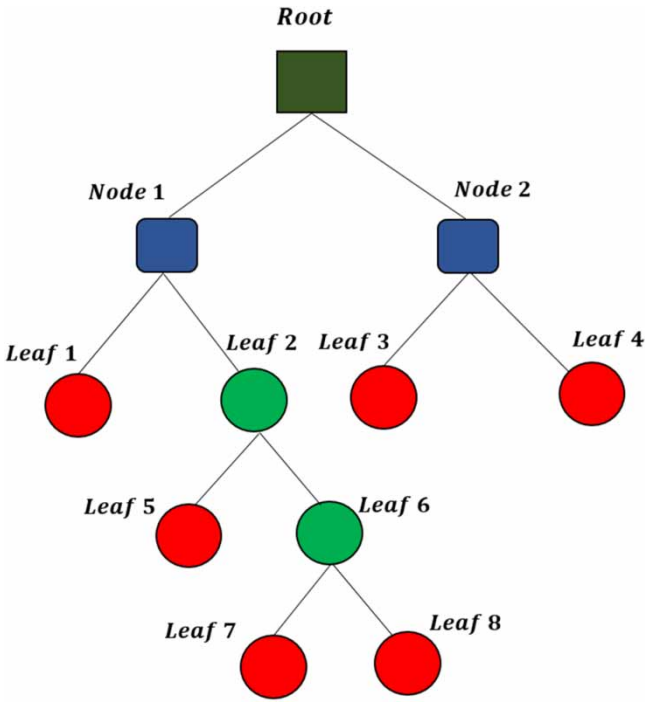


Figure 3 | Tree formation in LGBoost.

This leaf structure depends on gradient-based one-side sampling and exclusive feature bundling. Gradient-based one-side sampling helps in determining the best splits for each feature. It ranks various split points based on their gradients and ensures the identification of more informative splits, resulting in better predictions. Exclusive feature bundling reduces features by bundling several features into a few densely packed ones and lessens computational efforts. Accordingly, the term *light* is used due to its ability to achieve faster convergence. The mathematical expressions involved are as follows:

For the given input information, $I = \{x_i,\ y_i\}_{i=1}^{n}$, *the* algorithm aims to identify a base learner $\hat{f}(x_i)$ for achieving minimal loss function $L(y_i,\ f(x_i)$ and is expressed as (Equation (5)):

$$\hat{f}(x_i) = argmin_f\ E_{y_i,I} L(y_i,\ f(x_i)) \tag{5}$$

It integrates T base learners $\sum_{t=1}^{T} f_t(x_i)$ to find the final model $f_T(x_i)$ (Equation (6)):

$$f_T(x_i) = \sum_{t=1}^{T} f_t(x_i) \tag{6}$$

The base learners can be expressed as $w_{d(x_i)}$, $d \in \{1,\ 2,\ \ldots l\}$. where l indicates the number of leaves, d refers to the decision rules of the regression trees.

The combined form of the base learners at step t ($ב_t$) in the training process is presented as (Equation (7)):

$$ב_t = \sum_{i=1}^{n} (L(y_i,\ f_{t-1}(x_i) + f_t(x_i)) \tag{7}$$

Equation (7) is further simplified by discarding the constant term, and the transformed (Equation (8)) is as follows:

$$ב_t = \sum_{i=1}^{n} \left(g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right) \tag{8}$$

where g_i and h_i represent the first and second-order gradients of the loss function. Consider S_l is the sample set of leaf l, the equation is further transformed to (Equation (9)) as:

$$ב_t = \sum_{l=1}^{l} \left(\left(\sum_{i \in S_l} g_i \right) w_l + \frac{1}{2} \left(\sum_{i \in S_l} h_i + \lambda \right) w_l^2 \right) \tag{9}$$

For $d(x)$ tree structure optimal leaf weight scores w_l^* and the extreme value of $ב_T$ are computed (Equations (10) and (11)) as follows:

$$w_l{}^* = -\frac{\sum_{i \in S_l} g_i}{\sum_{i \in S_l} h_i + \lambda} \tag{10}$$

$$ב_T{}^* = -\frac{1}{2} \sum_{l=1}^{L} \frac{\left(\sum_{i \in S_l} g_i \right)^2}{\sum_{i \in S_l} h_i + \lambda} \tag{11}$$

where, $ב_T{}^*$ represents the scoring function. The final objective function (O) after adding the splits (Equation (12)) is as follows:

$$O = -\frac{1}{2} \left(\frac{\left(\sum_{i \in S_L} g_i \right)^2}{\sum_{i \in S_L} h_i + \lambda} + \frac{\left(\sum_{i \in S_R} g_i \right)^2}{\sum_{i \in S_R} h_i + \lambda} + \frac{\left(\sum_{i \in S_l} g_i \right)^2}{\sum_{i \in S_l} h_i + \lambda} \right) \tag{12}$$

Here S_L, S_R represents the sample sets of left and right branches, respectively. λ refers to the regularization parameter.

3.4. NGBoost

NGBoost (Duan *et al.* 2020) employs a natural rather than a conventional gradient to produce a probabilistic prediction with significantly higher accuracy. The predictions majorly rely on components such as base learners (f), distribution (P_θ), and the proper scoring rule (S). In general, gradient boosting methods determine the base learner that generates point estimates. On the other hand, base learners are probabilistic predictions that follow specific probability distribution functions, such as Gaussian or Laplace, in the case of NGBoost. The scoring rule is an objective function or criterion to decide the convergence of the solution. At least two degrees of freedom are required to effectively compute magnitude and prediction uncertainty. It takes time to perform simultaneous boosting with multiple parameters from base learners. NGBoost addresses this shortcoming through multiparameter boosting and using the natural gradient.

First, the information on input features (x_i) and target variable (y_i) are imported into the framework and are denoted as:

$$I = \{x_i,\ y_i\}_{i=1}^{n}$$

Based on the given input feature, initialize the appropriate base probabilistic model (Equation (13)):

$$f^{(m)}(x_i) = P_\theta\,(y_i|x_i) \tag{13}$$

Here θ is not limited to a scalar value, unlike traditional boosting methods. These values determine the probabilistic predictions of $P_\theta\,(y_i|x_i)$. For the chosen distribution having parameters μ and $log\sigma$, the base learners are $f_\mu^{(m)}$ and $f_{log\sigma}^{(m)}$ and are cumulatively represented as $f^{(m)}$ (Equation (14)):

$$f^{(m)} = (f_\mu^{(m)}, f_{log\sigma}^{(m)}) \tag{14}$$

The natural Gradient varies with $S(\theta,\ y_i)$ and is presented as (Equation (15)) follows:

$$\tilde{\nabla} S(\theta,\ y_i)\ \propto\ \omega_s(\theta)^{-1}\nabla S((\theta,\ y_i)) \tag{15}$$

Maximum likelihood estimation, Negative logarithmic likelihood, or Continuous ranked probability score can be used as a loss function.

The parameter updation after several boosting iterations is presented as (Equation (16)):

$$\theta_i^{(m)} = \theta_i^{(m-1)} - \eta \sum_{m=1}^{M} \rho^{(m)} f^{(m)}(x_i) \tag{16}$$

Initial parameter updation (Equation (17)) is presented as:

$$\theta^0 = argmin_\theta \sum_{i=1}^{n} \rho^{(m)} S(\theta,\ y_i) \tag{17}$$

$\rho^{(m)}$ refers to the scaling factor and is expressed as (Equation (18)):

$$\rho^{(m)} =\ argmin_\rho\, S(\theta_i^{(m)},\ \ y_i) \tag{18}$$

$g_i^{(m)} = \tilde{\nabla} S(\theta,\ y_i)$ represents the generalized natural Gradient and is presented as (Equation (19)):

$$g_i^{(m)} =\ \ \omega_s(\theta)^{-1}\nabla S((\theta,\ y_i)) \tag{19}$$

$f^{(m)}$ refers to the output of the base learner (Equation (20)):

$$f^{(m)} = \text{Fit}(\{x_i,\ g_i^{(m)}\}_{i=1}^{n} \tag{20}$$

where, $\theta_i^{(m)}$, $\theta_i^{(m-1)}$ indicate the parameter predicted after m and $m-1$ boosting iterations, respectively, and η refers to the learning rate.

3.5. XGBoost

XGBoost (Wang *et al.* 2022) is a gradient boosting technique used in ensembles. The decision tree is constructed in a depth or level-wise manner. It employs a similar mechanism to CatBoost, but the depth of the tree determines its formation. These are developed based on the similarity scores. The size varies depending on the leaves, nodes, and splits. Decision stump refers to a shorter decision tree possessing a single split. Typically, tree splits range between 4 and 8 for larger ones. However, splits can be increased further, requiring high computational power. The new trees are added to the ones that already exist. The losses incurred by adding trees are taken care of with the help of the gradient descent procedure. The mathematical expressions involved in XGBoost are presented (Equations (21)–(23)) as follows:

$$y_i^{(t)} = y_i^{(t-1)} - \eta.\omega_j^* \tag{21}$$

$$\omega_j^* = \frac{a_i + b_i}{2 + \lambda} \tag{22}$$

$$L(y_i,\ p_i) = y_i^{(t-1)} - \frac{1}{2}(y_i - p_i)^2 \tag{23}$$

Here, $y_i^{(t)}$ is the estimated value; $y_i^{(t-1)}$ is the initial prediction; ω_j^* represent the j^{th} leaf node's similarity score; y_i, p_i are observed and predicted values; a_i, b_i are the ordinates of a specific leaf node.

The formation of the decision tree continues until $L(y_i,\ p_i)$ meets the termination criteria. Lower learning rates are preferable as they facilitate sufficient time for capturing the input features, ensuring prediction accuracy.

KGE is the evaluation metric to judge the model performance as it considers the data correlation, bias, and variability (Newman *et al.* 2021). KGE greater than 0.75 is considered good (Towner *et al.* 2019). It is expressed as (Equation (24)):

$$\text{KGE} = 1 - \sqrt{(r-1)^2 + \left(\frac{\sigma_S}{\sigma_O} - 1\right)^2 + \left(\frac{\mu_S}{\mu_O} - 1\right)^2} \tag{24}$$

μ_O, σ_O are mean and standard deviations of observed data, whereas μ_s, σ_s are for the simulated. r is the correlation coefficient.

Comprehensive conceptual differences between the employed boosting algorithms are presented in Table 1.

Table 1 | Conceptual differences of boosting algorithms

	Algorithms				
Features	**AdaBoost**	**CatBoost**	**LGBoost**	**NGBoost**	**XGBoost**
Formation of decision tree	Asymmetric level-wise	Symmetric level-wise growth	Asymmetric leaf-wise growth	Level-wise growth	Depth-wise growth
Splitting method	Greedy splitting method	Greedy splitting method	Gradient-based one-side sampling	Natural gradient	Histogram based
Handling categorical features	No	Yes	No	No	No
Regularization	No	Yes	Yes	Yes	Yes
Memory consumption	Low	High	Low	High	Moderate
Feature importance	Available	Available	Available	Available	Available
Scalability	Fast	Moderate	Fast	Low	Fast

4. RESULTS

4.1. Hyperparameter tuning

Hyperparameters are the explicitly defined parameters that control the learning process and significantly impact algorithm performance. The parameters obtained for individual boosting algorithms using the grid search method are presented in Table 2. A few remarks on hyperparameters are also part of Table 2.

4.2. Boosting algorithms performance for historical data

The chosen algorithms were examined for 70:30 as well as 80:20 train-test split ratios. The KGE value for both split ratios in training and testing periods is presented in Figure 4. The 80:20 train-test split ratio performed slightly better than the 70:30 split ratio for all boosting algorithms. Accordingly, related results are presented only for 80:20 to minimize the repetition.

Figure 5(a) and 5(b) presents scatter plots for training and testing, corresponding to 374 training and 94 testing records. Brief but relevant inferences for AdaBoost, CatBoost, LGBoost, NGBoost, and XGBoost (in the same order) are as follows: [additional plots are presented in Supplementary material for the benefit of readers].

Table 2 | Hyperparameter-tuned boosting method parameters

Parameter (range) (1)	AdaBoost, CatBoost, LGBoost, NGBoost, XGBoost (2)	Remarks (3)
Learning rate (0–1)	0.2, 0.03, 0.03, 0.4, 0.3	The learning rate is the number of steps to update the weights to obtain the minimal loss function. CatBoost and LGBoost have optimized at a lower learning rate of 0.03, seven to ten times less than others.
Number of estimators (1–1,000)	100, 150, 50, 10, 100	The number of estimators is the threshold limit at which the boosting is terminated. NGBoost requires only 6.67% of estimators compared to CatBoost. It is 33.33% of CatBoost for LGBoost and 66.67% of CatBoost for AdaBoost and XGBoost, respectively. It is because of a lower learning rate where fewer estimators are sufficient for simulation.
Mini-batch fraction (0–1)	NA, NA, NA, 0.75, NA	Mini-batch fraction refers to the fraction of rows that require subsampling. This feature is exclusively present in NGBoost and will help avoid overfitting.
Colsample bytree (0.5–1)	1, NA, 0.7, 1, 0.9	Colsample bytree is an important hyperparameter that governs decision tree development and represents the percentage of columns requiring subsampling. A value of 0.5 indicates that the model will randomly sample half of the training data before growing trees. This will help to avoid overfitting and has the potential to speed up the training process. This parameter is required by all algorithms except CatBoost, as they are prone to overfitting.
Maximum depth (1–15)	3, 6, 4, 4, 2	Maximum depth refers to the complexity of the decision tree. Higher maximum depths than optimal value may lead to overfitting. CatBoost is relatively complex.
Minimum child weight (0–10)	NA, NA, 9.1, NA, 7.3	The minimum child weight determines the decision tree partitioning. With higher values, the tree partitions more conservatively. Lower values generate more decision trees and improve convergence but are more expensive to compute and essential in the LGBoost and XGBoost.
Number of leaves (1–1,000)	31, 31, 40, 31, 31	The number of leaves denotes the maximum number of splitting criteria in a decision tree. Except for LGBoost, all others were found to be identical in this instance.
Alpha (α) (0–5)	0, NA, 3.25, 1, 1.25	α denotes lasso-type regularization. Increasing their value causes the model to become more conservative. It is a critical hyperparameter for XGBoost, LGBoost, and NGBoost.
Lambda (Λ) (0–5)	0, 0.2, 4.5, 1, 2.5	Λ refers to ridge-type regularization. Increasing their value will make the algorithm more conservative and a crucial hyperparameter in the case of XGBoost, LGBoost, and NGBoost.
Subsample (0–1)	1, 1, 0.75, 1, 0.65	Subsample refers to the portion of training data chosen before forming the decision tree. XGBoost and LGBoost consider fractions of 0.65 and 0.75, which are lower than others.

NA, not applicable.

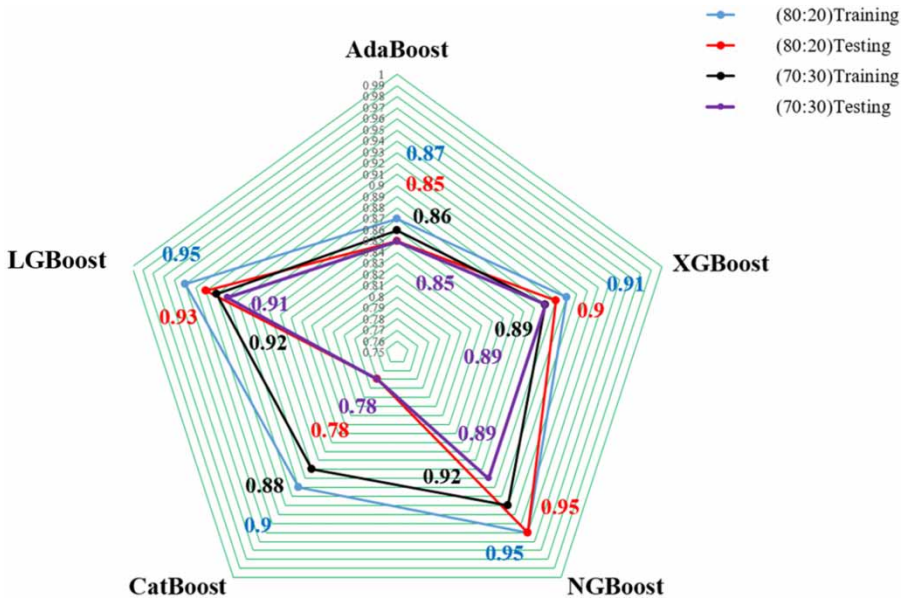


Figure 4 | Radar plot of KGE for boosting algorithms in 80:20 and 70:30 train-test split ratios.

In training:

- *Simulated maximum streamflow (year and under or overestimation)*: August 1990 (−3.70%), August 1990 (−0.33%), August 1990 (−11.41%), August 1990 (−3.51%), August 1990 (−55.39%) as compared to the maximum observed (+ indicates higher, −indicates lower than historical scenario).
- *Average simulated streamflow*: +9.01%, −1.88%, +0.24%, +2.82%, −2.54% compared to the average observed streamflow.
- *Year of occurrence of highest streamflow underestimation (overestimation)*: September 1987 (August 1986), September 2010 (January 2010), September 1994 (August 1986), September 1987 (August 1986), August 1990 (July 2010).
- Interquartile ranges are 1,304.43, 908.83, 997.43, 1,019.42, and 908.79 m^3/s. Upper extreme whisker values are 3,419.34, 2,354.21, 2,599.77, 2,653.26, and 2,357.74 m^3/s. Outliers identified above upper extreme whisker *values (in percentages)* are 5.35, 13.64, 13.10, 13.64, and 14.17 (Figure 6(a)).
- KGE values in training are 0.87, 0.90, 0.95, 0.95 and 0.91

In testing:

- Simulated maximum streamflow (year and under or overestimation): August 2013 (+6.39%), August 2013 (−27.60%), August 2013 (+3.23%), August 2013 (+4.6%), August 2013 (−15.73%) as compared to the maximum observed.
- Average simulated streamflow: +15.32%, +5.24%, +2.10%, +3.92%, +2.27% compared to average observed streamflow.
- Year of occurrence of highest streamflow underestimation (overestimation): July 2013 (June 2013), August 2013 (July 2018), September 2019 (July 2018), July 2013 (July 2018), August 2013 (July 2018).
- Interquartile ranges are 1,412.43, 1,517.37, 1,136.11, 1,175.83 and 1,153.44 m^3/s. Upper extreme whisker values are 3,685.33, 3,889.26, 2,944.8, 3,042.44 and 2,955.85 m^3/s. Outliers identified above upper extreme whisker values (in percentages) are 6.38, 4.26, 8.51, 6.38, and 10.64. It is noticed that the outliers are comparatively low in testing over the training period (Figure 6(b)).
- KGE values in testing are 0.85, 0.78, 0.93, 0.95 and 0.90.

All these boosting algorithms are further considered (as a condition of KGE more than 0.75 is satisfied) for projecting streamflow using 13 CMIP6-GCMs for four SSPs for two time splits: short-term projections (2025–2050) and long-term projections (2051–2075). For now, they are mentioned as short-term and long-term throughout the manuscript.

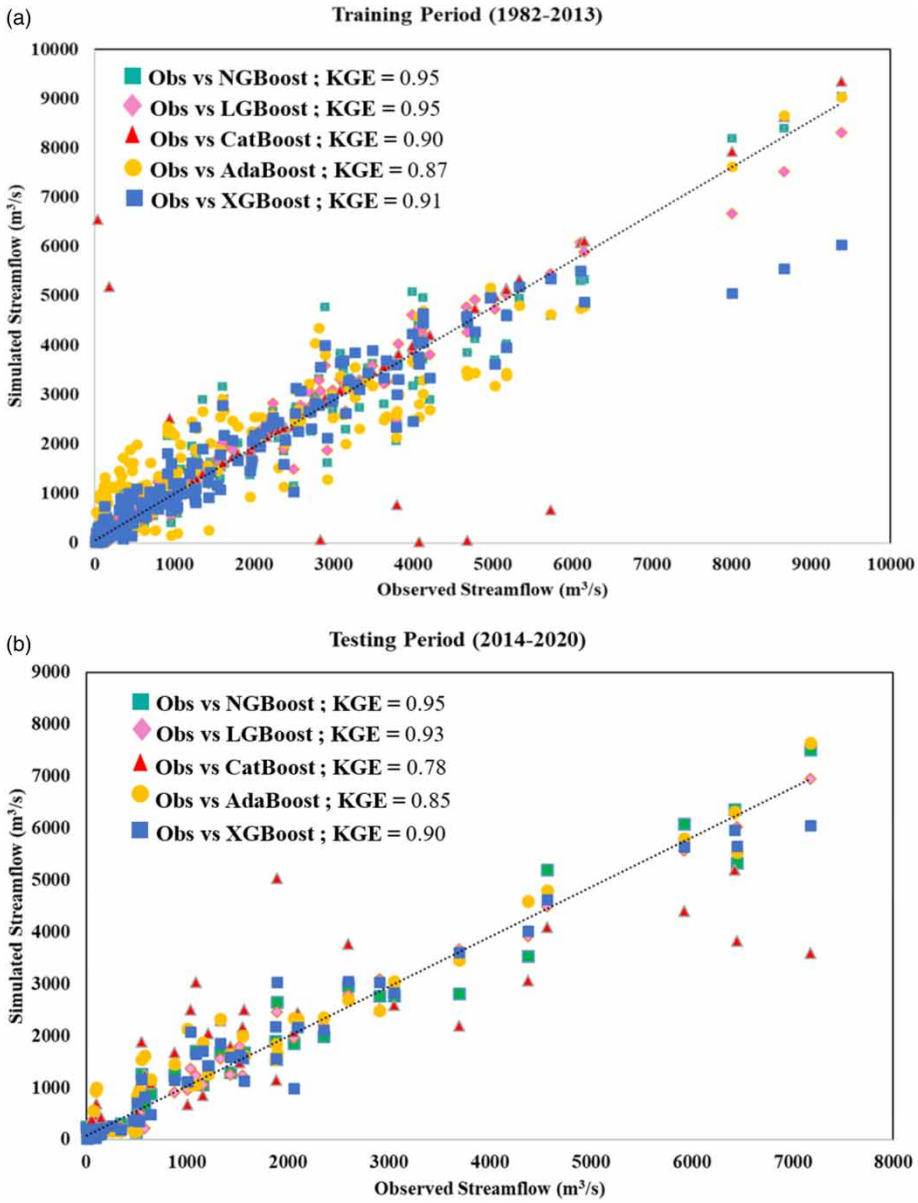


Figure 5 | Scatter plots for boosting algorithms in (a) training and (b) testing periods for 80:20 ratio.

4.3. Streamflow projections for short-term and long-term

To understand the fluctuations, the average, maximum, and minimum projected stream flow in the short-term and long-term were compared with historical scenarios and presented in Table 3. Results are presented in the order of the following algorithms: AdaBoost, CatBoost, LGBoost, NGBoost, and XGBoost.

Short-term:

- The discharge line plots follow a regular trend across the boosting algorithms. The highest and lowest fluctuations were observed in NGBoost and XGBoost, as presented in Figure 7(a)–7(e).
- *Minimum streamflow*: 1,860.22 m^3/s (all four SSPs), 785.35 m^3/s (SSP585), 3,507.82 m^3/s (SSP585), 130.23 m^3/s (SSP585), 2,701.23 m^3/s (SSP370).
- *Maximum streamflow*: 4,581.04 m^3/s (SSP245), 3,558.72 m^3/s (SSP370), 6,116.03 m^3/s (SSP245), 10,252.47 m^3/s (SSP370), 6,962.72 m^3/s (SSP126).

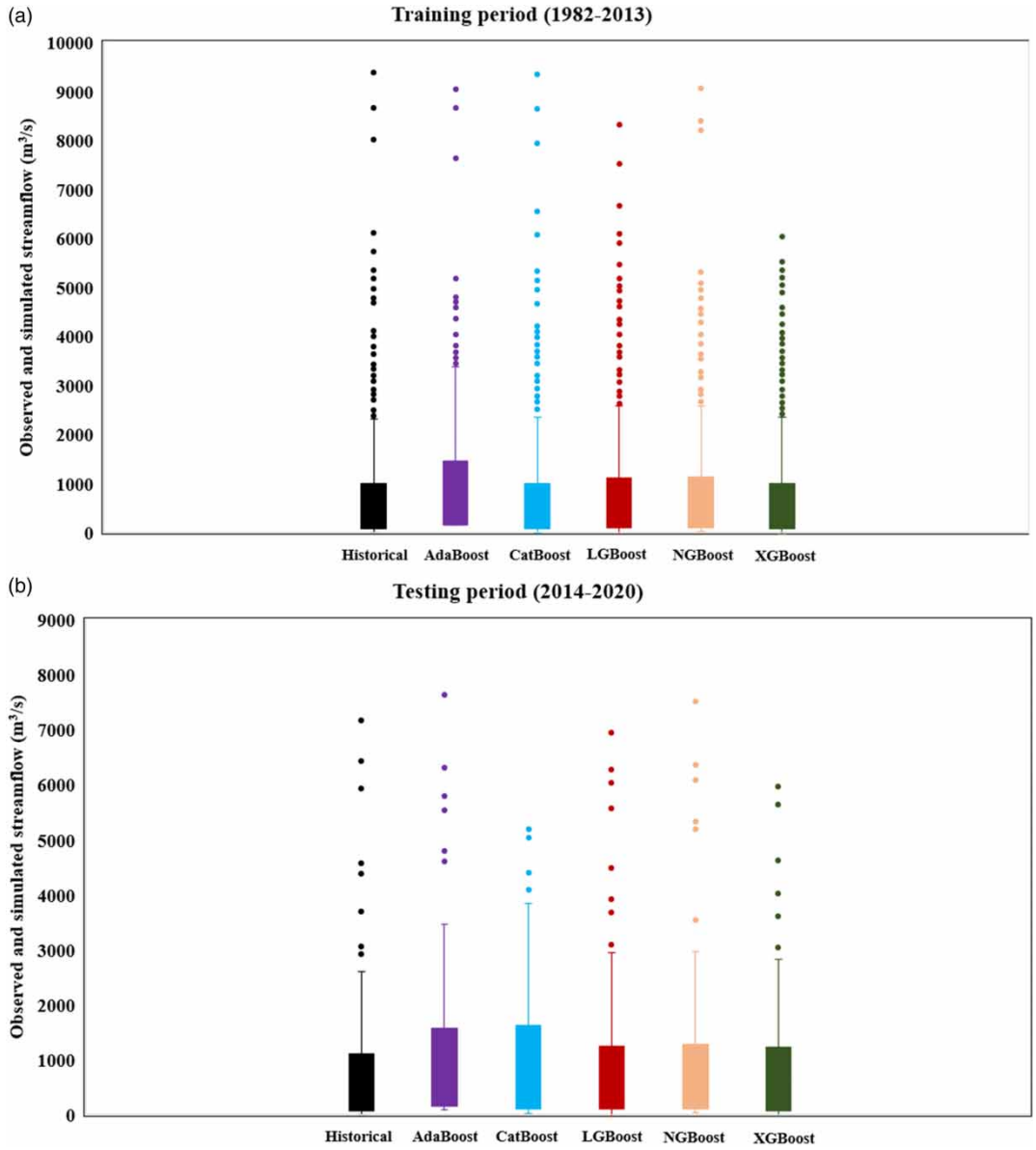


Figure 6 | Box plots of boosting algorithms in (a) training and (b) testing periods for 80:20 ratio (* dots above upper whisker represents the outliers).

- *Maximum average streamflow*: 2,762.03 m^3/s (SSP245), 2,193.01 m^3/s (SSP126), 5,223.24 m^3/s (SSP245), 5,602.52 m^3/s (SSP126), 5,739.18 m^3/s (SSP126).
- *Average streamflow*: An increase from SSP126 to SSP245 and a further decline in SSP370 and SSP585 is observed for AdaBoost. In the case of CatBoost, the decrease is observed from SSP126 to SSP245, a further increment in SSP370, and a decline in SSP585. NGBoost shows the opposite of the trend observed in AdaBoost. The decreasing trend from SSP126 to SSP585 is observed in the case of XGBoost.

Long-term:

- The discharge line plots follow a similar trend as short-term across the boosting algorithms and as presented in Figure 7(a)–7(e).

Table 3 | Projected streamflow information using five boosting algorithms and weighted ensemble (all units are in m^3/s)

		Short-term (2025–2050)				Long-term (2051–2075)			
Models	**Aspects**	**SSP126**	**SSP245**	**SSP370**	**SSP585**	**SSP126**	**SSP245**	**SSP370**	**SSP585**
AdaBoost	A	2,737.41	**2,762.03**	2,751.19	2,699.42	2,662.48	2,677.67	2,625.79	2,493.86
	Min	**1,860.22**	**1,860.22**	**1,860.22**	**1,860.22**	1,836.20	1,813.78	1,813.78	1,813.78
	Max	3,782.34	**4,581.04**	4,034.55	3,555.66	3,514.14	3,924.49	3,514.14	3,514.14
CatBoost	A	**2,193.01**	2,162.62	2,169.38	2,151.91	2,149.23	2,165.10	2,131.95	2,092.87
	Min	1,542.26	1,406.37	1,261.32	785.35	1,287.79	1,561.68	1,566.31	**1,568.47**
	Max	3,446.19	3,375.58	**3,558.72**	3,293.30	3,328.22	3,276.30	3,107.12	3,021.09
LGBoost	A	5,088.53	5,223.24	5,116.53	5,150.70	5,087.79	**5,271.58**	5,140.03	5,137.63
	Min	**3,578.25**	**4,276.73**	**3,666.38**	3,507.82	**3,117.54**	3,940.61	3,807.19	3,494.22
	Max	5,955.94	6,116.03	5,984.29	6,089.25	6,295.86	**6,394.36**	6,055.87	6,264.96
NGBoost	A	5,602.52	4,753.41	5,395.90	5,336.07	**5,731.18**	4,738.92	5,463.92	5,149.72
	Min	**1,999.77**	1,469.10	1,910.15	130.23	1,298.04	938.72	1,909.33	1,772.30
	Max	**10,206.34**	**10,179.85**	**10,252.47**	**10,117.49**	**10,015.78**	**10,477.52**	**9,730.26**	**9,732.85**
XGBoost	A	**5,739.18**	5,708.37	5,702.23	5,683.71	5,676.24	5,675.74	5,676.26	5,674.72
	Min	3,507.38	**4,642.03**	2,701.23	**3,608.55**	3,048.62	**4,890.46**	4,675.70	4,586.85
	Max	6,962.72	6,516.09	6,652.09	6,615.51	6,646.65	6,474.66	**6,987.93**	6,712.82
Weighted ensemble	A	**4,307.27**	4,151.96	4,260.91	4,238.63	4,298.37	4,136.93	4,243.49	4,144.88
	Min	2,837.43	3,148.64	3,143.09	2,176.15	2,385.01	3,137.09	**3,180.01**	3,244.70
	Max	5,843.96	**6,017.61**	5,713.89	5,866.37	5,381.66	5,897.91	5,374.81	5,344.49

A, Min, and Max indicate, respectively, the average, minimum, and maximum in m^3/s.

- Minimum streamflow: 1,813.78 m^3/s (SSP245, SSP370 and SSP585), 1,287.79 m^3/s (SSP126), 3,117.54 m^3/s (SSP126), 938.72 m^3/s (SSP245), 3,048.62 m^3/s (SSP126).
- Maximum streamflow: 3,924.49 m^3/s (SSP245), 3,328.22 m^3/s (SSP126), 6,394.36 m^3/s (SSP245), 10,477.52 m^3/s (SSP245), 6,987.93 m^3/s (SSP370).
- Maximum average streamflow: 2,677.67 m^3/s (SSP245), 2,165.10 m^3/s (SSP245), 5,271.58 m^3/s (SSP245), 5,731.18 m^3/s (SSP126), 5,676.26 m^3/s (SSP370).
- Average streamflow: A streamflow decrease is observed from SSP126 to SSP245, increased in SSP370, and declined in SSP585 in the case of NGBoost. The trend almost remains flat in the case of XGBoost. The remaining is the same as short-term.

4.4. Weighted ensembling-based streamflow projections

A weighted ensembling is used to provide unified projections for the benefit of policymakers and is necessitated as all algorithms satisfy the requirements of KGE. Accordingly, all five were considered for the ensembling. The information on average, maximum, and minimum projected stream flow of weighted ensembling in two time periods SSP-wise were compared with that of the historical scenario and are presented in Table 3. A brief discussion is presented as follows:

- The discharge line plots of weighted ensembling follow a regular trend across SSPs of two time periods, as presented in Figure 8.
- The minimum streamflow in short-term and long-term are (2,387.43 m^3/s, 2,385.01 m^3/s) for SSP126; (3,148.64 m^3/s, 3,137.09 m^3/s) for SSP245; (3,143.09 m^3/s, 3,180.01 m^3/s) for SSP370 and (2,122.24 m^3/s, 3,244.7 m^3/s) for SSP585 respectively.
- The maximum streamflow in short-term and long-term are (5,843.96 m^3/s, 5,381.66 m^3/s) for SSP126; (6,017.61 m^3/s, 5,897.91 m^3/s) for SSP245; (5,713.89 m^3/s, 5,374.81 m^3/s) for SSP370 and (5,866.37 m^3/s, 5,344.49 m^3/s) for SSP585 respectively.
- The average streamflow in short-term and long-term are (4,307.27 m^3/s, 4,298.37 m^3/s) for SSP126; (4,151.96 m^3/s, 4,136.93 m^3/s) for SSP245; (4,260.91 m^3/s, 4,243.49 m^3/s) for SSP370 and (4,238.46 m^3/s, 4,144.88 m^3/s) for SSP585 respectively.
- It is observed that the magnitude of minimum streamflows increased drastically in all the SSPs, corresponding to short-term and long-term compared to the historical scenario.

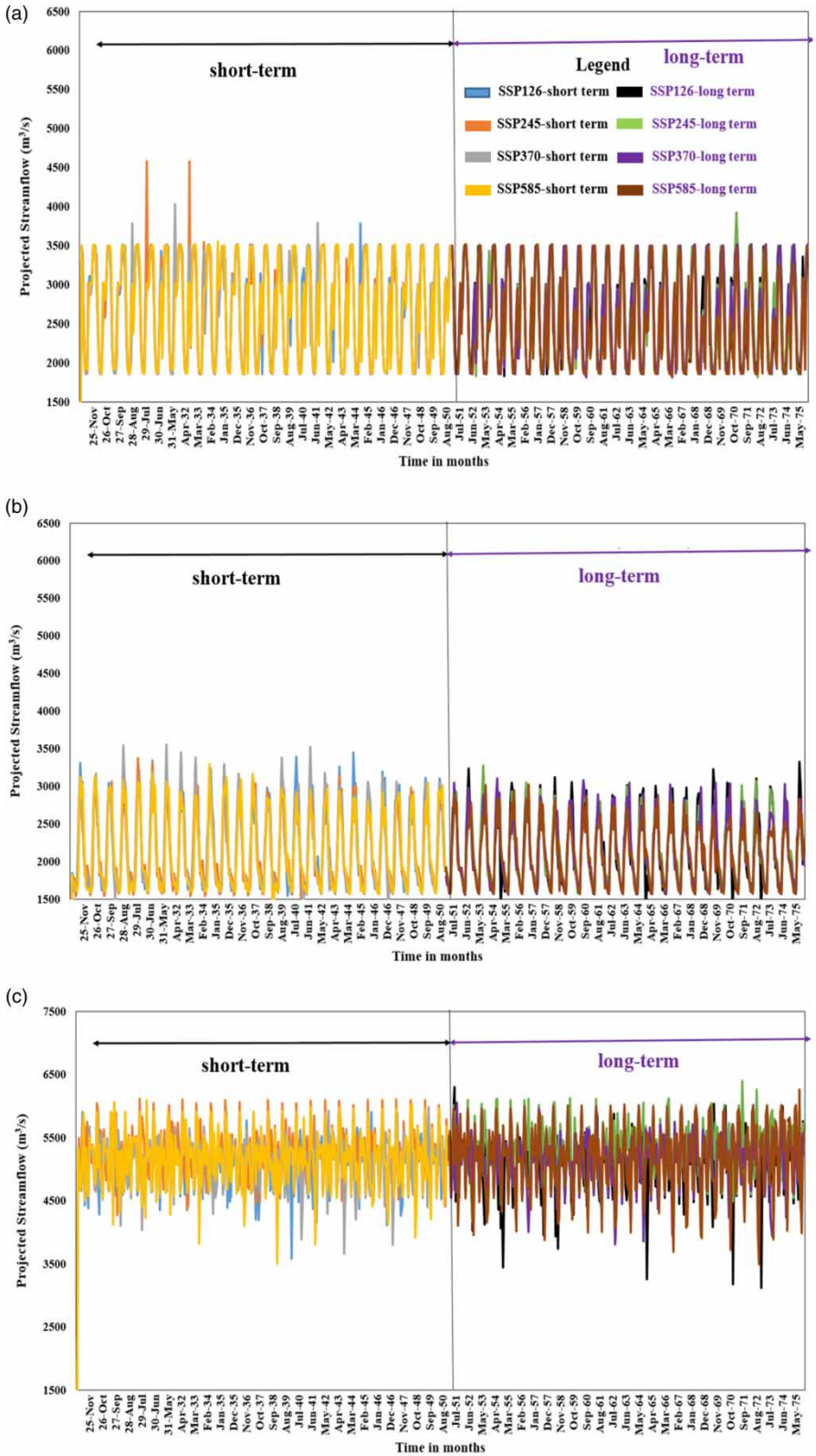


Figure 7 | Projected streamflow for four SSPs in short-term and long-term: (a) AdaBoost; (b) CatBoost; (c) LGBoost; (d) NGBoost; and (e) XGBoost. *(Continued.)*

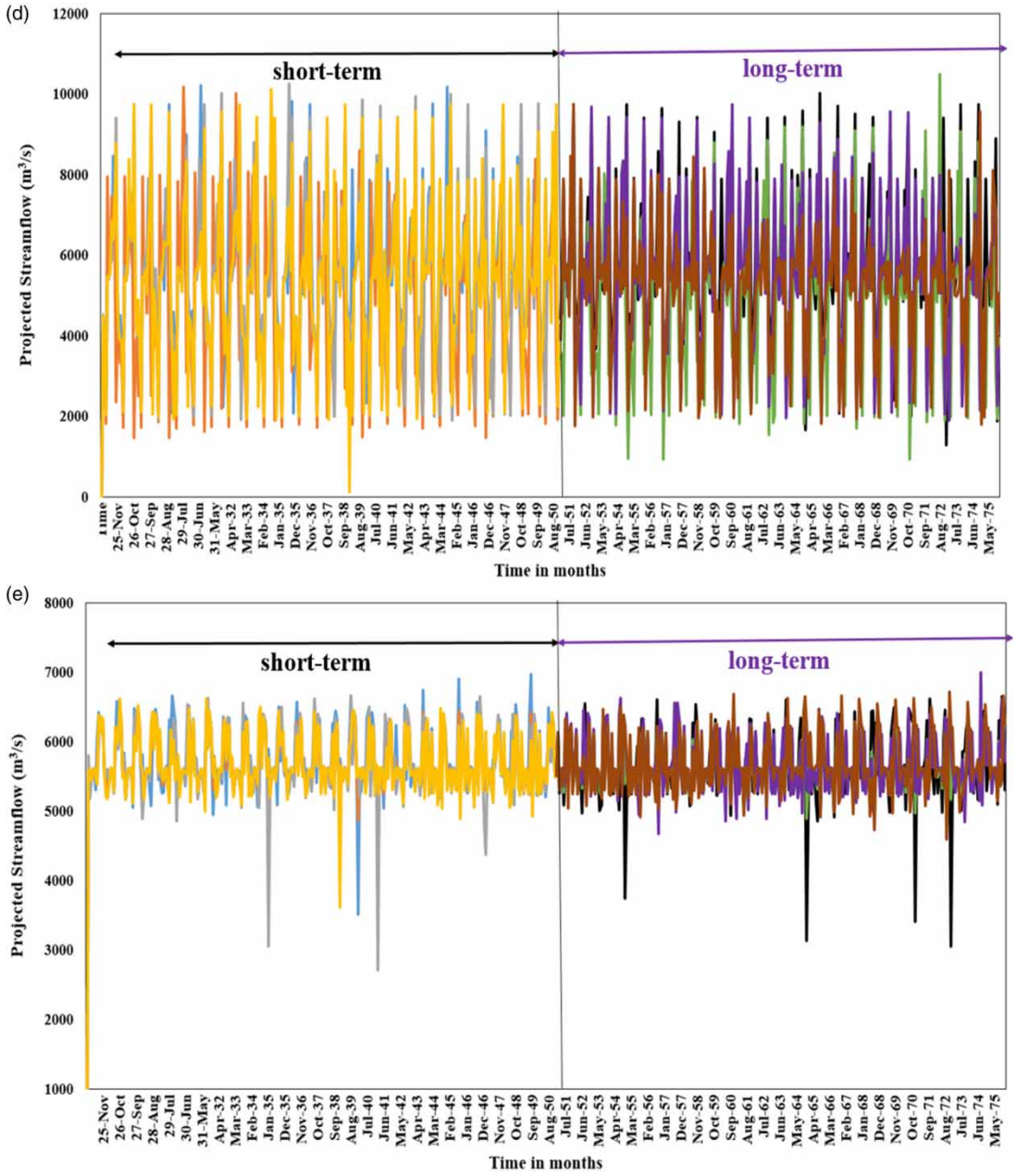


Figure 7 | *Continued.*

- The maximum streamflows are 0.57–0.65 times in the short- and long-term across the SSPs compared with the maximum streamflow observed in the historical scenario.
- The increase in average streamflow for short-term and long-term across SSPs is about 4.66–4.85 times that of the historical scenario. The positive side of this situation is that there is scope for the basin to receive sufficient water to fulfill its water demands.

4.4.1. Lowest and highest number of events

The lowest and highest number of events in Table 4 are based on the four SSPs over two periods, short-term and long-term (refer to Table 4). Only one value is presented for the historical scenario. The boosting algorithms obtained lower and higher values among eight possibilities, i.e., four SSP and two time periods. Salient observations from Table 4 are as follows:

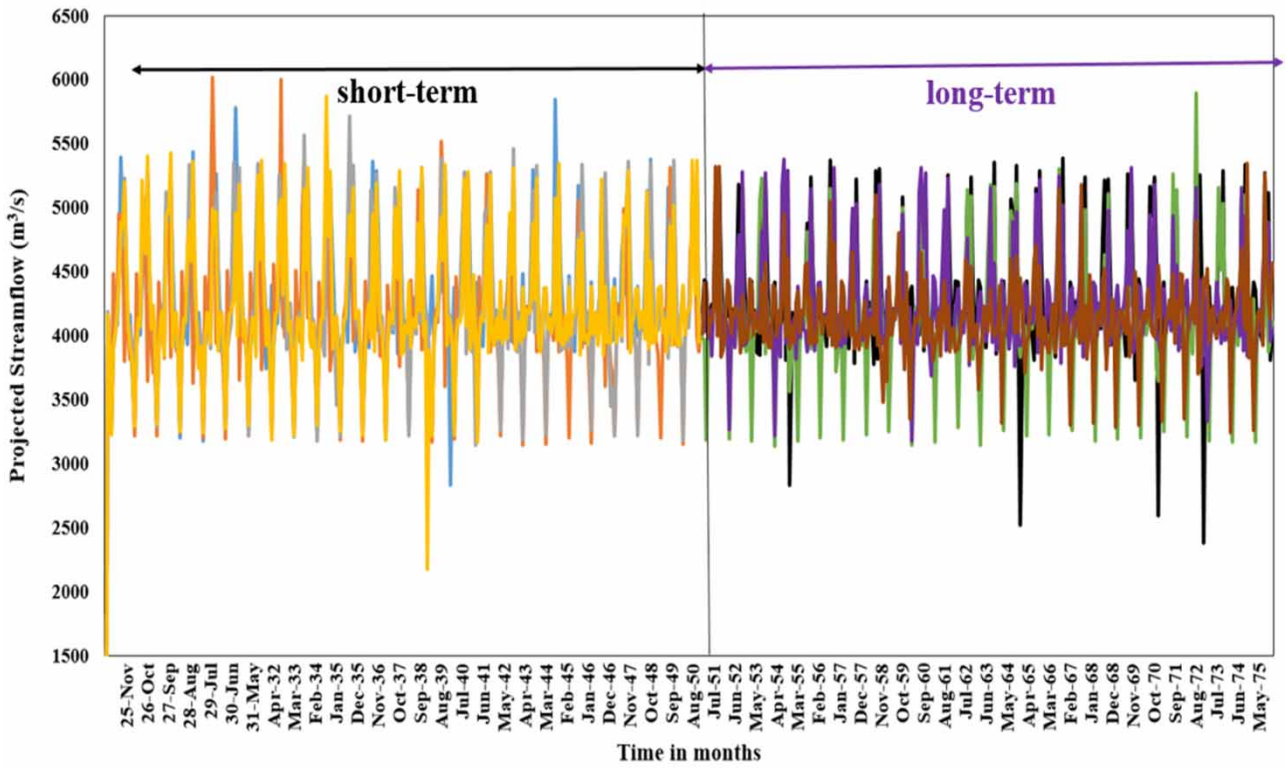


Figure 8 | Projected streamflow for four SSPs in short-term and long-term using weighted ensemble.

- In the historical scenario category, three hundred fifty events are less than 1,000 m^3/s. NGBoost has recorded the highest number in the less than 1,000 m^3/s segment, followed by CatBoost. No events were identified in other algorithms.
- CatBoost has the highest number of events in 1,000–2,000 m^3/s and 2,000–3,000 m^3/s segments, whereas no events were identified in LGBoost.
- AdaBoost has the highest number of events in the 3,000–4,000 m^3/s segment.
- Weighted ensembling has the highest events in the 4,000–5,000 m^3/s segment, whereas no events were identified in CatBoost.
- XGBoost has the highest number of events in 5,000–6,000 m^3/s and 6,000–7,000 m^3/s segments, whereas no events were identified in AdaBoost and CatBoost.
- NGBoost has highest number of events in 7,000–8,000, 8,000–9,000, 9,000–10,000, and $>$ 10,000 m^3/s segments, whereas no events were reported in other algorithms.
- NGBoost has reported at least one event in the 1,000–2,000 to 8,000–9,000 m^3/s category, similar to the historical scenario. This situation is not observed in other algorithms.

5. DISCUSSION

In the present study, NGBoost demonstrated an exceptional Kling Gupta Efficiency value of 0.95 in both the training and testing periods, affirming its effectiveness in simulating streamflow. A similar inference was made by Shen *et al.* (2022).

Several comparative studies have presented contrasting views to the present study by the authors, suggesting that AdaBoost (Khan *et al.* 2023), XGBoost (Ma *et al.* 2021), CatBoost (Rathnayake *et al.* 2023), and LGBoost (Fan *et al.* 2019; McCarty *et al.* 2020; Szczepanek 2022; Xu *et al.* 2023) exhibit greater simulating abilities other than NGBoost. However, none of these studies have applied NGBoost to validate or confirm the dominance of those algorithms.

Table 4 | Lowest and highest number of events (based on four SSPs over short-term and long-term)

	Segment										
Algorithm	**<1,000 m^3/s**	**1,000–2,000 m^3/s**	**2,000–3,000 m^3/s**	**3,000–4,000 m^3/s**	**4,000–5,000 m^3/s**	**5,000–6,000 m^3/s**	**6,000–7,000 m^3/s**	**7,000–8,000 m^3/s**	**8,000–9,000 m^3/s**	**9,000–10,000 m^3/s**	**>10,000 m^3/s**
Historic scenario	350	42	29	19	11	9	4	1	2	1	0
AdaBoost	(0,0)	(69,96)	(85,117)	(87,150)	(0, 2)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)
CatBoost	(0,1)	(125,164)	(123,173)	(2,29)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)
LGBoost	(0,0)	(0,0)	(0,0)	(0,9)	(73,124)	(178,200)	(0,34)	(0,0)	(0,0)	(0,0)	(0,0)
NGBoost	(0,3)	(1,31)	(13,29)	(23,85)	(20,55)	(78,147)	(14,40)	(13,35)	(7,17)	(0,22)	(0,3)
XGBoost	(0,0)	(0,0)	(0,1)	(0,4)	(1,11)	(198,233)	(63,92)	(0,0)	(0,0)	(0,0)	(0,0)
Weighted ensemble	(0,0)	(0,0)	(0,4)	(66,120)	(165,202)	(8,43)	(0,2)	(0,0)	(0,0)	(0,0)	(0,0)

Furthermore, the development of hybrid boosting algorithms such as Gaussian Mixture Model-XGBoost (Ni *et al.* 2020), Genetic Algorithm-CatBoost (Kilinc *et al.* 2023), Gradient Boosting Regressor-Bagging Regressor (Dastour & Hassan 2023), and Deep Boost (Pham *et al.* 2021) are also found to help improve simulation efficacy.

The choice of hyperparameter tuning method is equally important for achieving better simulation efficacy. The present study utilizes the grid search approach to optimize selected boosting algorithms that resonate with the prior works (Fan *et al.* 2019; Ni *et al.* 2020; Adib & Harun 2022; Szczepanek 2022; Xu *et al.* 2023). This extensive adoption of grid search highlights its effectiveness in thoroughly examining a predetermined range of values inside the hyperparameter space, which is important when complex interactions between hyperparameters are unclear.

Notably, Shen *et al.* (2022) and Kilinc *et al.* (2023) employed hyperparameter methods like Tree-structured Parzen Estimator and Genetic Algorithm, differing from the choice of the present study. 'Shen *et al.* (2022) reported optimal values, including 900 learners, a learning rate of 0.03722, a subsample percentage of 0.6, and a base learner depth of 3. However, a notable deviation in the number of learners is observed between the present study and Shen *et al.* (2022). In contrast, no significant deviation exists regarding learning rate and base learner depth. For LGBoost, Fan *et al.* (2019) identified optimal values of 100 trees, a maximum depth of 3, and a learning rate of 0.05, differing findings from the present study, i.e., 50 trees, a depth of 4, and a learning rate of 0.03. McCarty *et al.* (2020) reported an optimal number of leaves as 1,024, with a remarkably low learning rate of 0.005, in contrast to the present study. Szczepanek (2022) found optimal values of the number of estimators and learning rate as 200 and 0.5, respectively, nearly four times and 17 times higher than reported in the present study.

Furthermore, Xu *et al.* (2023) found the number of leaves, number of estimators, and learning rate, to be 12, 450, and 0.11, respectively. All the aforementioned hyperparameters deviated significantly from the optimal values obtained in the study. These differences underscore the sensitivity of hyperparameter optimization to specific contexts and datasets, emphasizing the importance of tailored tuning for betterment of algorithm performance in streamflow simulations.

The reliability of future climate data is imperative for generating credible streamflow projections. In this regard, Adib & Harun (2022) and Singh *et al.* 2023) preferred a mean ensemble of multiple GCMs. Adib & Harun (2022) employed five GCMs with SSP245 and SSP585, whereas Singh *et al.* (2023) employed six GCMs with SSP126, SSP245 and SSP585. The present study also focuses on a similar framework employing a mean ensemble of thirteen GCMs with SSP126, SSP245, SSP370, and SSP585.

6. SUMMARY AND CONCLUSIONS

Five boosting algorithms, AdaBoost, CatBoost, LGBoost, NGBoost, and XGBoost, were implemented to simulate the streamflow of the Lower Godavari Basin. Grid search-based hyperparameter tuning was utilized. Rainfall, temperatures, and streamflow were utilized for training and testing. Train-test split ratios of 70:30 and 80:20 were examined in this study. The model performances were diagnosed using KGE. The following are the salient conclusions drawn from this study:

- 80:20 train-test split ratio has shown slightly better performance than 70:30 in training and testing periods.
- All the boosting algorithms have been further used for projecting streamflows for the short-term and long-term due to their higher KGE values.
- Minimum and maximum streamflow of 130.23 and 10,252.47 m^3/s were observed in SSP585 and SSP370 respectively, based on all short-term projections.
- Minimum and maximum streamflow of 938.72 and 10,477.52 m^3/s were observed in SSP245 based on all long-term projections.
- The average streamflow trends for LGBoost, NGBoost, and weighted ensembling were the same for the short-term and long-term. At the same time, they differ in the case of AdaBoost, CatBoost, and XGBoost in the short and long term.
- The highest number of events were recorded for CatBoost in (1,000–2,000 and 2,000–3,000 m^3/s) segments; AdaBoost in (3,000–4,000 m^3/s); weighted ensembling in (4,000–5,000 m^3/s); XGBoost in (5,000–6,000 and 6,000–7,000 m^3/s); and NGBoost in (7,000–8,000, 8,000–9,000, 9,000–10,000, and $>$10,000 m^3/s) segments.

The study primarily focuses on boosting algorithms with grid search optimization methods. All five boosting algorithms achieve greater simulation efficacy, as evidenced by the employed metric. It indicates the excellent potential of boosting algorithms to simulate complex problems such as rainfall–runoff modeling. In addition, the impact of ensembling five algorithms to generate streamflows in the climate change framework is a new dimension to researchers in this field and allied fields.

However, specific challenges need to be addressed for further advancements. In this regard, it is essential to emphasize the development of hybrid algorithms and robust hyperparameter tuning techniques. Additionally, incorporating a fuzzy logic layer within the boosting algorithms can effectively tackle the challenge of uncertain input data, improving the understanding of the relationship between input variables and streamflow. Furthermore, exploring novel combinations of ensemble models in the climate change perspective holds promise for gaining new insights and deriving more reliable predictions. Overcoming these limitations and challenges is crucial to driving progress in climate change research.

ACKNOWLEDGEMENTS

This research work is sponsored by CSIR, New Delhi, through Project no. 22(0782)/19/EMR-II dated 24.7.19. The authors thank the officials for their support.

DATA AVAILABILITY STATEMENT

Data cannot be made publicly available; readers should contact the corresponding author for details.

CONFLICT OF INTEREST

The authors declare there is no conflict.

REFERENCES

Adib, M. N. M. & Harun, S. 2022 Metalearning approach coupled with CMIP6 multi-GCM for future monthly streamflow forecasting. *Journal of Hydrologic Engineering* **27** (6), 05022004.

Adnan, R. M., Jaafari, A., Mohanavelu, A., Kisi, O. & Elbeltagi, A. 2021 Novel ensemble forecasting of streamflow using locally weighted learning algorithm. *Sustainability* **13** (11), 5877.

Asgari, M., Yang, W., Lindsay, J., Tolson, B. & Dehnavi, M. M. 2022 A review of parallel computing applications in calibrating watershed hydrologic models. *Environmental Modelling & Software* **151**, 105370.

Başağaoğlu, H., Chakraborty, D., Lago, C. D., Gutierrez, L., Şahinli, M. A., Giacomoni, M. & Şengör, S. S. 2022 A review on interpretable and explainable artificial intelligence in hydroclimatic applications. *Water* **14** (8), 1230.

Dastour, H. & Hassan, Q. K. 2023 A machine-learning framework for modeling and predicting monthly streamflow time series. *Hydrology* **10** (4), 95.

Duan, T., Anand, A., Ding, D. Y., Thai, K. K., Basu, S., Ng, A. & Schuler, A. 2020 NGBoost: Natural Gradient boosting for probabilistic prediction, Proceedings of the 37th International Conference on Machine Learning, *July 13–18, 2020 (Virtual Event)* 119, 2690–2700.

Fan, J., Ma, X., Wu, L., Zhang, F., Yu, X. & Zeng, W. 2019 Light gradient boosting machine: An efficient soft computing model for estimating daily reference evapotranspiration with local and external meteorological data. *Agricultural Water Management* **225**, 105758.

Graf, R. & Vyshnevskyi, V. 2022 Forecasting monthly river flows in Ukraine under different climatic conditions. *Resources* **11** (12), 111.

Guo, Y., Zhang, Y., Zhang, L. & Wang, Z. 2021 Regionalization of hydrological modeling for predicting streamflow in ungauged catchments: A comprehensive review. *Wiley Interdisciplinary Reviews: Water* **8** (1), e1487.

Hengade, N., Eldho, T. I. & Ghosh, S. 2018 Climate change impact assessment of a river basin using CMIP5 climate models and the VIC hydrological model. *Hydrological Sciences Journal* **63** (4), 596–614.

Hu, Y., Fitzpatrick, L., Fry, L. M., Mason, L., Read, L. K. & Goering, D. C. 2021 Edge-of-field runoff prediction by a hybrid modeling approach using causal inference. *Environmental Research Communications* **3** (7), 075003.

Ibrahim, T., Geremew, B. & Tesfay, F. 2021 Spatio-temporal dynamic of land use and land cover in Andit Tid Watershed, Wet Frost/Afro-Alpine Highland of Ethiopia. *Edelweiss Applied Science and Technology* **5** (1), 33–38.

Jhajharia, D., Gupta, S., Mirabbasi, R., Kumar, R. & Patle, G. T. 2021 Pan evaporative changes in transboundary Godavari River basin, India. *Theoretical and Applied Climatology* **145**, 1503–1520.

Jin, W. 2022 Cognitive radio spectrum allocation based on IOT and genetic Algorithm. *Journal of Commercial Biotechnology* **27** (1). https://doi.org/10.5912/jcb1071.

Khan, S., Khan, M., Khan, A. U., Khan, F. A., Khan, S. & Fawad, M. 2023 Monthly streamflow forecasting for the Hunza River Basin using machine learning techniques. *Water Practice & Technology* **18** (8), 1959–1969.

Khosravi, K., Golkarian, A., Booij, M. J., Barzegar, R., Sun, W., Yaseen, Z. M. & Mosavi, A. 2021 Improving daily stochastic streamflow prediction: Comparison of novel hybrid data-mining algorithms. *Hydrological Sciences Journal* **66** (9), 1457–1474.

Kilinc, H. C., Ahmadianfar, I., Demir, V., Heddam, S., Al-Areeq, A. M., Abba, S. I. & Yaseen, Z. M. 2023 Daily scale river flow forecasting using hybrid gradient boosting model with genetic algorithm optimization. *Water Resources Management* **37**, 3699–3714.

Kim, T., Yang, T., Gao, S., Zhang, L., Ding, Z., Wen, X. & Hong, Y. 2021 Can artificial intelligence and data-driven machine learning models match or even replace process-driven hydrologic models for streamflow simulation?: a case study of four watersheds with different hydro-climatic regions across the CONUS. *Journal of Hydrology* **598**, 126423.

Kumar, V., Kedam, N., Sharma, K. V., Mehta, D. J. & Caloiero, T. 2023 Advanced machine learning techniques to improve hydrological prediction: A comparative analysis of streamflow prediction models. *Water* **15** (14), 2572.

Liu, Y., Zhang, J., Wang, G., Wang, G., Jin, J., Liu, C. & He, R. 2020 How do natural climate variability, anthropogenic climate and basin underlying surface change affect streamflows? A three-source attribution framework and application. *Journal of Hydro-Environment Research* **28**, 57–66.

Ma, M., Zhao, G., He, B., Li, Q., Dong, H., Wang, S. & Wang, Z. 2021 XGBoost-based method for flash flood risk assessment. *Journal of Hydrology* **598**, 126382.

Manikanta, V. & Umamahesh, N. V. 2023 Performance assessment of methods to estimate initial hydrologic conditions for event-based rainfall-runoff modelling. *Journal of Water and Climate Change* **14** (7), 2277–2293.

Markhali, S. P., Poulin, A. & Boucher, M. A. 2022 Spatio-temporal discretization uncertainty of distributed hydrological models. *Hydrological Processes* **36** (6), e14635.

Marques, A. C., Veras, C. E. & Rodriguez, D. A. 2022 Assessment of water policies contributions for sustainable water resources management under climate change scenarios. *Journal of Hydrology* **608**, 127690.

McCarty, D. A., Kim, H. W. & Lee, H. K. 2020 Evaluation of light Gradient boosted machine learning technique in large scale land use and land cover classification. *Environments* **7** (10), 84.

Mehraein, M., Mohanavelu, A., Naganna, S. R., Kulls, C. & Kisi, O. 2022 Monthly streamflow prediction by metaheuristic regression approaches considering satellite precipitation data. *Water* **14** (22), 3636.

Mishra, V., Bhatia, U. & Tiwari, A. D. 2020 Bias-corrected climate projections for South Asia from coupled model intercomparison project-6. *Scientific Data* **7** (1), 338.

Nesru, M. 2023 A review of model selection for hydrological studies. *Arabian Journal of Geosciences* **16** (2), 102.

Newman, A. J., Stone, A. G., Saharia, M., Holman, K. D., Addor, N. & Clark, M. P. 2021 Identifying sensitivities in flood frequency analyses using a stochastic hydrologic modeling system. *Hydrology and Earth System Sciences* **25** (10), 5603–5621.

Ni, L., Wang, D., Wu, J., Wang, Y., Tao, Y., Zhang, J. & Liu, J. 2020 Streamflow forecasting using extreme Gradient boosting model coupled with Gaussian mixture model. *Journal of Hydrology* **586**, 124901.

Oborie, E. & Rowland, E. D. 2023 Flood influence using GIS and remote sensing based morphometric parameters: A case study in Niger delta region. *Journal of Asian Scientific Research* **13** (1), 1–15.

Pham, Q. B., Pal, S. C., Chakrabortty, R., Norouzi, A., Golshan, M., Ogunrinde, A. T. & Anh, D. T. 2021 Evaluation of various boosting ensemble algorithms for predicting flood hazard susceptibility areas. *Geomatics, Natural Hazards and Risk* **12** (1), 2607–2628.

Polavaram Project Authority Annual Report 2020–21 Ministry of Jal Shakti, Department of Water Resources, River Development & Ganga Rejuvenation, Government of India.

Rathnayake, N., Rathnayake, U., Dang, T. L. & Hoshino, Y. 2023 Water level prediction using soft computing techniques: A case study in the Malwathu Oya, Sri Lanka. *Plos One* **18** (4), e0282847.

Sarkar, S. 2022 Drought and flood dynamics of Godavari basin, India: A geospatial perspective. *Arabian Journal of Geosciences* **15** (8), 772.

Sharma, R. K., Kumar, S., Padmalal, D. & Roy, A. 2023 Streamflow prediction using machine learning models in selected rivers of Southern India. *International Journal of River Basin Management*. doi:10.1080/15715124.2023.2196635.

Shen, K., Qin, H., Zhou, J. & Liu, G. 2022 Runoff probability prediction model based on natural Gradient boosting with tree-structured parzen estimator optimization. *Water* **14** (4), 545.

Singh, D., Vardhan, M., Sahu, R., Chatterjee, D., Chauhan, P. & Liu, S. 2023 Machine-learning-and deep-learning-based streamflow prediction in a hilly catchment for future scenarios using CMIP6 GCM data. *Hydrology and Earth System Sciences* **27** (5), 1047–1075.

Speight, L. J., Cranston, M. D., White, C. J. & Kelly, L. 2021 Operational and emerging capabilities for surface water flood forecasting. *Wiley Interdisciplinary Reviews: Water* **8** (3), e1517.

Szczepanek, R. 2022 Daily streamflow forecasting in mountainous catchment using XGBoost, LightGBM and CatBoost. *Hydrology* **9** (12), 226.

Towner, J., Cloke, H. L., Zsoter, E., Flamig, Z., Hoch, J. M., Bazo, J. & Stephens, E. M. 2019 Assessing the performance of global hydrological models for capturing peak river flows in the Amazon basin. *Hydrology and Earth System Sciences* **23** (7), 3057–3080.

Troin, M., Arsenault, R., Wood, A. W., Brissette, F. & Martel, J. L. 2021 Generating ensemble streamflow forecasts: A review of methods and approaches over the past 40 years. *Water Resources Research* **57** (7), e2020WR028392.

Wang, F., Li, Z., He, F., Wang, R., Yu, W. & Nie, F. 2019 Feature learning viewpoint of AdaBoost and a new algorithm. *IEEE Access* **7**, 149890–149899.

Wang, S., Peng, H., Hu, Q. & Jiang, M. 2022 Analysis of runoff generation driving factors based on hydrological model and interpretable machine learning method. *Journal of Hydrology: Regional Studies* **42**, 101139.

Xu, K., Han, Z., Xu, H. & Bin, L. 2023 Rapid prediction model for urban floods based on a light Gradient boosting machine approach and hydrological–hydraulic model. *International Journal of Disaster Risk Science* **14** (1), 79–97.

Yang, S., Tan, M. L., Song, Q., He, J., Yao, N., Li, X. & Yang, X. 2023 Coupling SWAT and Bi-LSTM for improving daily-scale hydro-climatic simulation and climate change impact assessment in a tropical river basin. *Journal of Environmental Management* **330**, 117244.

Yiming, H. U., Teng, C. H., Xuyi, L. U., Chao, T. A. & Zhongmin, L. 2022 Medium to long term runoff forecast for the Huai River Basin based on machine learning algorithm. *Earth Science Frontiers* **29** (3), 284.

Zhang, S., Gan, T. Y., Bush, A. B. & Zhang, G. 2023 Evaluation of the impact of climate change on the streamflow of major pan-Arctic river basins through machine learning models. *Journal of Hydrology* **619**, 129295.

Zounemat-Kermani, M., Batelaan, O., Fadaee, M. & Hinkelmann, R. 2021 Ensemble machine learning paradigms in hydrology: A review. *Journal of Hydrology* **598**, 126266.

First received 18 August 2023; accepted in revised form 29 December 2023. Available online 18 January 2024