Journal of Building Engineering 72 (2023) 106523

Contents lists available at ScienceDirect

Journal of Building Engineering

journal homepage: www.elsevier.com/locate/jobe

Multitarget regression models for predicting compressive strength and chloride resistance of concrete

Woubishet Zewdu Taffese *, Leonardo Espinosa-Leal

School of Research and Graduate Studies, Arcada University of Applied Sciences, Helsinki, Finland

ARTICLE INFO

Keywords:
Concrete
Multitarget
Regression
Model
Machine learning
Compressive strength
Chloride penetration
Durability

ABSTRACT

This work develops single multitarget regression models that predict compressive strength and non-steady-state chloride migration coefficients (D_{nssm}) of concrete simultaneously using machine learning algorithms. The data for this study are obtained from research projects and internationally published articles. Following data preprocessing, the compressive strength ranged from 21 to 80 MPa, while the D_{nssm} ranged from 1.57 to 31.30 $\times$ 10^{-12} m^2/s. The algorithms used are five decision tree-based ensemble methods: bagging, random forest, AdaBoost, Gradient boosting, and XGBoost. In the development of the models, two scenarios are considered. Scenario 1 employs the default hyperparameter settings, while Scenario 2 employs hyperparameters chosen from among those identified through training single-target models. The performance evaluation results confirm that Gradient boosting is the best performing algorithm and Scenario 2 is the most appropriate modeling strategy for the considered dataset. It predicts compressive strength with (MAE = 6.683, MSE = 83.369, and RMSE = 9.131) and D_{nssm} with (MAE = 1.363, MSE = 3.712, and RMSE = 1.927). The potential of the developed multitarget model to design concrete with the intended strength and D_{nssm} is supported by its remarkable generalization ability. However, in order to ensure the model's versatility, it is necessary to improve it by incorporating comprehensive datasets that include a broad range of concrete properties.

1. Introduction

Reinforced concrete (RC) structures are essential in civil infrastructure development, but they deteriorate over time due to various reasons. Chloride attack is one of the most significant threats to durability of RC structures exposed to marine environments and/or chloride-containing de-icing salts in cold climates [1]. Chloride penetration itself does not cause concrete damage, but once the concentration of chloride ions reaches a certain level at the steel reinforcement bar, it leads to depassivation and subsequent corrosion [2]. Corrosion negatively impacts the serviceability and safety of RC structures and results in significant economic losses due to premature rehabilitation of civil infrastructures [3–5]. Some developed countries spend up to 3.5–4.5% of their GDP on corrosion-related damage and control [3].

To avoid premature failure of RC structures due to corrosion and its associated costs, the concrete must be designed to resist chloride attack. Concrete mix design methods tend to prioritize the workability and strength properties of the concrete, often neglecting to address its durability. The majority of concrete mix design techniques, such as those developed by the American Concrete Institute (ACI) [6] and the British Department of Environment (DoE) [7], are intricate and involve multiple steps that depend on

* Corresponding author.
E-mail address: woubishet.taffese@arcada.fi (W.Z. Taffese).

https://doi.org/10.1016/j.jobe.2023.106523
Received 18 January 2023; Received in revised form 26 March 2023; Accepted 9 April 2023
Available online 15 April 2023

mathematical computations and laboratory experiments. The proportion of concrete mix constituents that is determined through analytical calculations must be confirmed by laboratory tests, which are typically conducted on trial mixes after 28 days. The proportions of ingredients are then adjusted until an acceptable mix is achieved through an iterative process. This approach is not only time-consuming but also expensive and consumes natural resources.

There are several laboratory testing techniques suggested to quantify the chloride diffusion coefficient of concrete, in order to determine its resistance to chloride. The Nordic standard NT Build 492 is among the accelerated testing methods utilized for determining the chloride diffusion coefficient [8]. The chloride diffusion coefficient determined by this method is referred to as the "non-steady-state migration coefficient" or D_{nssm}. Although this test method yields quick results, it is usually conducted 28 days after concrete production because curing is required. In recent years, significant efforts have been made to develop analytical and statistical models for predicting chloride diffusion coefficients in an effort to shorten the duration of this procedure [9–11]. While these models are significant, many of them do not account for non-steady-state migration coefficient and focus mainly on diffusion coefficients obtained from other types of tests. Additionally, some models only consider a limited number of factors and overlook several essential ones that define concrete's microstructure.

Incorporating supplementary cementitious materials (SCMs) that come from industrial and agricultural by-products such as ground granulated blast-furnace slag (GGBS), fly ash (FA), silica fume (SF), and rice husk ash (RHA), further complicates the traditional concrete mix design process, owing to the intricate properties of each material. For example, the effectiveness of RHA as a pozzolanic material is influenced not only by its amorphous content but also by factors such as its specific surface area and particle fineness [12]. Due to such a diverse properties and compositions of SCMs often make it difficult for conventional mix design methods to fully address the related complexities. Thus, to determine the proportions of ingredients that meet the required properties, an intelligent guess based on the predefined relationships must eventually be made. As a result, concrete characteristics are often designed with an excessive margin of safety as a preventative measure, due to concerns of not achieving the required properties and incurring added expenses. Moreover, empirical techniques are practical only when a limited number of parameters are taken into account. Incorporating further important properties that describe the durability of concrete, such as its resistance to chloride, would result in a substantial increase in the number of concrete samples that must be prepared for laboratory testing.

All these limitations underscore the necessity for a reliable, efficient, straightforward, and cost-effective method that empowers engineers to create concrete that can withstand chloride attack with minimal steps. The use of advanced machine learning algorithms is a practical method to meet these requirements. Machine learning algorithms can detect the implicit patterns among numerous features and establish intricate relationships without necessitating explicit knowledge.

Over the past few years, several research paper [13–18] have indicated that machine learning techniques have the ability to create models for predicting chloride penetration and compressive strength in concrete. However, these models focused on only a single parameter at a time, either describing strength or chloride penetration, rather than considering both parameters simultaneously. Developing separate models for each parameter in isolation without taking into account their interdependence does not facilitate analysis of trade-offs. This process is undeniably complicated, time-consuming, and results in suboptimal concrete mix optimization to meet the desired strength and chloride resist level. To overcome this challenge, the current study proposes the development of single multitarget models that can simultaneously predict both compressive strength and chloride penetration using machine learning algorithms that are capable of capturing all dependencies and internal relationships between the two parameters. In addition, using a single multitarget model is easier to interpret than using multiple single-target models, and it also ensures higher prediction accuracy [19], resulting in optimal concrete mix.

The remainder of the paper is organized as follows. Section 2 gives a general overview of machine learning. Section 3 describes ensemble methods in detail, as all five algorithms used in this work are ensemble methods that use multiple machine learning algorithms to achieve better prediction performance. Section 4 describes the materials and methods used. This section describes in detail the dataset used for model training and testing, as well as all activities associated with model development. Section 5 discusses the research results. Finally, Section 6 contains the conclusions.

2. Overview of machine learning

Machine learning is a subfield of artificial intelligence involving the design and implementation of algorithms to recognise complex patterns in data and make rational decisions [20]. Machine learning models can be descriptive to reveal knowledge from data or predictive to carry out prediction, or both, without relying on predefined equations [20]. The data could come in the form of a digitized human-labeled training dataset or other sorts of information collected through interaction with the environment. The quality and size of the data are critical in all circumstances for the learning algorithm's predictions to be successful. Even though machine learning originated from the quest for artificial intelligence, it has a broader scope and applications. It incorporates concepts from various fields, including neuroscience, information theory, statistics, and computational linguistics, among others.

A machine learning method that maps an input to an output based on input-output pairs is known as supervised learning. This learning type can be divided into two categories based on the nature of the desired output (target feature): classification and regression. Classification refers to supervised learning issues in which the target feature is a discrete set of classes. In contrast, regression refers to situations in which the value of the target feature is continuous. Generally, regression problems are solved by creating a functional model that is the best predictor of y based on a given input x retaining a specific training dataset $D = \{y_i, x_i\}_1^N$ as shown in Equation (1).

$$y = \widehat{F}(x_1, x_2, \ldots, x_n) = \widehat{F}(X), \tag{1}$$

where y_i is the output feature, x_i is the input vector made of all the feature values for the $i-th$ observation, n is the number of features, and N is the number of observations.

The primary aim of this study is to utilize regression learning methods to predict the continuous target features of compressive strength and D_{nssm} of concrete. To achieve this, the study employs ensemble machine learning algorithms that have demonstrated their ability to handle complex nonlinear regression problems in various civil engineering disciplines [21–29]. The following section describes the disciplines principle of ensemble methods.

3. Ensemble methods

Ensemble methods are designed to create a powerful predictive model by combining multiple machine learning models that each solve the same original task [30]. They are applicable to the two most important predictive modeling tasks, classification and regression. In both cases, the ensemble method outperforms a single model, but requires more computation time because multiple models need to be built. To take advantage of the superior predictive power of the ensemble method, proper base model building and aggregation techniques are required [31]. The base models can be multiple independent similar or different models. Model aggregation involves the integration of base models $m_1, m_2, \ldots, m_k$ into an ensemble model M through the development of a prediction combination strategy that computes $M(x)$ based on $m_1(x), m_2(x), \ldots, m_k(x)$ for any $x \in X$. The combined model M is represented by all of its base models as well as the strategy used to integrate their predictions.

In this work, decision trees are used as base models to build various ensemble methods. Decision trees are fast learners with a high level of interpretability that handle complex nonlinear problems with large numbers of observations and input variables by decomposing them into manageable levels and applying the same approach to the subproblems recursively. After a discussion of how decision trees work, there follows a brief discussion of the integration methods applied, used to form the ensemble methods, which are either bagging or boosting.

3.1. Decision tree

A decision tree is essentially an acyclically connected graph structure made up of nodes, branches, and leaves. A decision tree used to solve regression problems can be referred as a regression tree. Fig. 1 shows the basic structure of a regression tree. The left subfigure depicts the data points and their partitions, while the right subfigure depicts the structure of the corresponding regression tree. Decision nodes represent domain areas that must be broken down into smaller areas by dividing them. Leaf nodes represent domain areas where further divisions are not possible. The root node is the highest node in the tree structure. Branches are connected to descendant or leaf nodes in proportion to certain split results. Splits are determined by some relational circumstances based on the selected instances, which can have two or more outcomes. A split can be formally defined by a test function $t : X \rightarrow R_t$ that maps instances into split outcomes. Each possible result of splitting a node is assigned its own outgoing branch. The relationship between the parent node and its descendant nodes, which is theoretically defined by the branches connecting the former to the latter, need not always have to be clearly defined in the decision tree's data structure. If the result of a split can be determined explicitly for each reachable instance, then the domain is partitioned into disjoint subsets proportional to the outgoing branches. As a result, as presented in Equation (2), each node n of a decision tree corresponds to an area (subset) of the domain determined by the sequence of splits $t_1, t_2, \ldots, t_k$ and their results $r_1, r_2, \ldots, r_k$ that occur on the path from a root to leaf nodes [31].

$$X_n = \{x \in X | t_1(x) = r_1 \wedge t_2(x) = r_2 \wedge \cdots \wedge t_k(x) = r_k\} \tag{2}$$

3.2. Bagging

Bagging, also known as bootstrap aggregation, is a general technique for combining the predictions of multiple models to form an ensemble method. The base models in a bagging-based ensemble learning are built by randomly selecting bootstrapped samples from the dataset. This procedure is repeated multiple times until a significant subset of training datasets is generated and the same samples can be collected more than once. Each bootstrap training dataset formed contains an average of $N\left(1-\frac{1}{e}\right) \approx 0.63N$ instances, where N is the total number of samples in the dataset. The instances that were left out are known as out-of-bag observations, and they are used to evaluate the model's performance. The final output of the ensemble method is the average of the predicted outputs of the individual

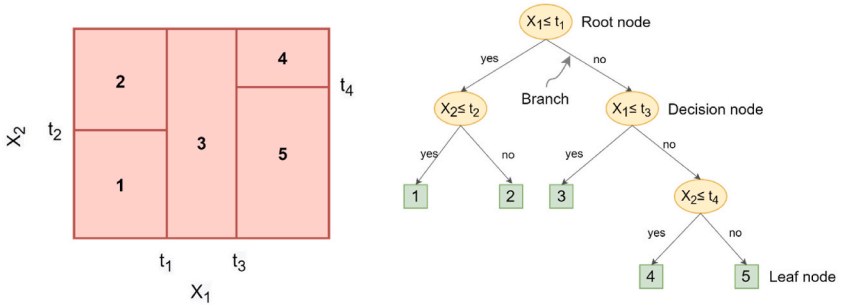

Fig. 1. An example of a dataset and its associated regression tree.

base models, which reduces variance and produces better stability [30]. Each base model fits the training dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ to achieve the tree's prediction $\widehat{f}(x)$ at input vector x. Bagging takes the average of this prediction over a series of bootstrap samples. The model predicts $\widehat{f}^{*t}(x)$ for each bootstrap sample $D^{*t}, t = 1, 2, \ldots, T$, the bagged estimate is the mean prediction at x from T trees, as shown in Equation (3).

$$\widehat{f}_{bag}(x) = \frac{1}{T}\sum_{t=1}^{T}\widehat{f}^{*t}(x) \tag{3}$$

Another improved bagging variant is the *Random Forest*, which is essentially an ensemble of base models trained with a bagging mechanism. Breiman [32] introduced feature randomness into the bagging procedure, resulting in an uncorrelated forest of decision trees. It is a significant improvement over bagged decision trees and is one of the ensemble methods used in this work.

3.3. Boosting

Boosting is a bagging improvement that spans multiple base models by shifting focus to cases that are struggling to perform well [31]. Unlike bagging, boosting builds simple base models serially and fuses them with improvements from one model to the next to improve the performance of the ensemble method. Each base model is built from a training dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ using knowledge from previously built base models. An appropriate algorithm is used to fit the training datasets $D^{(t)}, t = 1, 2, \ldots, T$ with a sequence of different weights $w^{(1)}, w^{(2)}, \ldots, w^{(T)}$, returning the predictions of the base models $\widehat{f}^{(1)}(x), \widehat{f}^{(2)}(x), \ldots, \widehat{f}^{(T)}(x)$ for each input vector x and its corresponding weight vector w. Typically, the weight vector is generated by implementing an initial weight $w^{(1)}$ and is continuously adjusted in each base model built based on perceived errors. The weight is increased in cases where the base model produces large errors and decreased in cases where the model generates small errors. The final output of the model is a weighted sum of the individual base model outputs, as expressed in Equation (4).

$$\widehat{f}_{boost}(x) = \sum_{t=1}^{T}\widehat{f}^{(t)}(x)w^{t} \tag{4}$$

The way boosting algorithms create and aggregate weak learners during the sequential process can vary. The three most popular ensemble methods based on boosting are as follows. They are all used in this work.

Adaptive boosting or AdaBoost: This algorithm was developed by Yoav Freund and Robert Schapire [33]. It was originally designed to solve classification problems but was later extended to solve regression problems. The core principle of AdaBoost is to fit a sequence of weak learners on repeatedly modified versions of the data and adjust their weights to minimize training error. The model continues sequential optimization until it finds the best predictor.

Gradient boosting: Jerome H. Friedman [34] invented gradient boosting, which works by sequentially adding predictors to an ensemble, each correcting for its predecessor's errors. However, unlike AdaBoost, gradient boosting trains on the residuals of the previous predictor instead of changing the weights of data points. The term gradient boosting refers to the combination of the gradient descent algorithm and the boosting method.

Table 1
Description of the selected features from the raw dataset.

Feature category	No.	Feature subcategory	Description	Unit
Concrete mix ingredients	1	Cement types	CEM I CEM II: CEM II/A-S, CEM II/B–S, CEM II/A-D, CEM II/A-V, CEM II/B–V, CEM II/A-L, CEM II/B-L, CEM II/A-LL, CEM II/A-M, CEM II/B-M CEM III: CEM III/A, CEM III/B CEM IV: CEM IV/A, CEM IV/B	–
	2	Water content		[kg/m^3]
	3	Cement content	Cement	[kg/m^3]
	4	Supplementary cementitious materials	Slag	[kg/m^3]
	5		Fly ash	[kg/m^3]
	6		Silica fume	[kg/m3]
	7	Water-to-binder ratio		–
	8	Total aggregate		[kg/m^3]
	9	Chemical admixtures content	Plasticizer	[% by binder wt.]
	10		Superplasticizer	[% by binder wt.]
	11		Air-entraining agent	[% by binder wt.]
Hardened concrete properties	12	Mechanical properties	Compressive strength	[MPa]
	13		Concrete age at compressive strength test	[days]
	14	Migration properties	Concrete age at migration test	[days]
	15		Non-steady-state migration coefficient (D_{nssm})	[x10^{-12} m^2/s]

Extreme gradient boosting or XGBoost: It was initially created by Tianqi Chen in 2016 [35], and it now has many contributors. XGBoost was developed using the general principles of gradient boosting and is designed for computational speed and scaling. It makes use of multiple central processing unit (CPU) cores, allowing for parallel learning during training.

4. Materials and methods

In this section, the dataset used and the development process of machine learning based models for predicting compressive strength and D_{nssm} of concrete are discussed. The dataset includes various types of concrete that employed a variety of binders and admixtures. The main activities involved in the model development process are data preprocessing, model training, and model evaluation. All are described in detail below.

4.1. Raw dataset

The raw dataset used in this study was originally created by the authors to develop data-driven non-steady-state migration coefficients prediction models. The data were collected from research projects and internationally published journal articles by accessing the databases Web of Science and Scopus, which are abstract and citation databases of peer-reviewed literature, delivering a complete citation search by giving access to numerous databases. The set of queries, ("strength" AND "migration coefficient" AND "concrete"), which comprise the title, abstract, and keywords of the articles, were run on both databases to retrieve works reporting on concrete mixes with their fresh and hardened properties, including chloride resistance properties. All duplicate records in the databases were removed and suitable ones selected manually. A total of 22 scientific papers [36–57], including a research project report, were identified that were suitable for creating a dataset for the study. The dataset contains twenty-three features that describe the mix ingredients as well as the properties of fresh and hardened concrete. Following a preliminary assessment of the data quality and relevance, fifteen features that provide information on the concrete mix ingredients and its hardened properties are selected. Table 1 list all the selected features. Features that describe the concrete mix ingredients includes cement types, amount of water content (unit kg/m^3), content of binders: cement and supplementary cementitious materials (slag, FA, and SF) (unit kg/m^3), w/b ratio, amount of total aggregate (unit kg/m^3), content of chemical additives: plasticizer, superplasticizer, and air-entraining agent (AEA) in (% by binder wt.). The hardened concrete property test includes test results of compressive strength (unit MPa) and non-steady-state migration coefficients (unit of $x10^{-12}$ m^2/s) performed at different maturity ages. All the concrete in the dataset was cured at the age of 28 days.

As presented in Table 1, the dataset only includes two concrete property tests: compressive strength and non-steady-state migration coefficient. The first describes the quality of the concrete from a strength standpoint, while the second describes the durability standpoint. Based on the value of the non-steady-state migration coefficient, the concrete's resistance to chloride ion penetration can be classified as given in Table 2.

4.2. Pipeline

The pipeline of the models is shown in Fig. 2. Obtaining the dataset containing information on the concrete constituents and proportions, compressive strength and the chloride migration coefficient, which describes the concrete's resistance to chloride attack. Then comes data preprocessing, which is the most important step in developing machine learning models, since data from real-world scenarios is generally noisy, contains missing values, and may even be in an unusable format that cannot be used directly in a machine learning model. To make the data suitable for a machine learning model, a variety of activities are typically performed during data processing, including missing data processing, outlier detection and treatment, data encoding, and splitting the dataset into a training and test dataset. The next step is performing model training with appropriate learning algorithms using the training dataset. The models' performance is then evaluated using a test dataset that the models have not seen before. The process of training and testing the model is repeated until the best results are obtained by optimizing the hyperparameters. The model development process is carried out by writing code in Python in the Jupyter notebook environment, which is an open-source web application that provides an interactive computing environment for creating, executing, and visualizing interactive data offers in a variety of programming languages.

4.3. Data preprocessing

Data preprocessing is a crucial step in the development of any machine learning-based models. Missing data processing, outlier detection and handling, data encoding, data normalization, and data partitioning are among the common tasks of data preprocessing. All the data preprocessing tasks applied in this work are detailed in the following subsections.

Table 2
Concrete chloride penetration resistance classification criteria [37].

D_{nssm} ($x10^{-12}$ m^2/s)	Chloride Penetration Resistance of Concrete
>15	Low
10–15	Moderate
5–10	High
2.5–5	Very high
<2.5	Extremely high

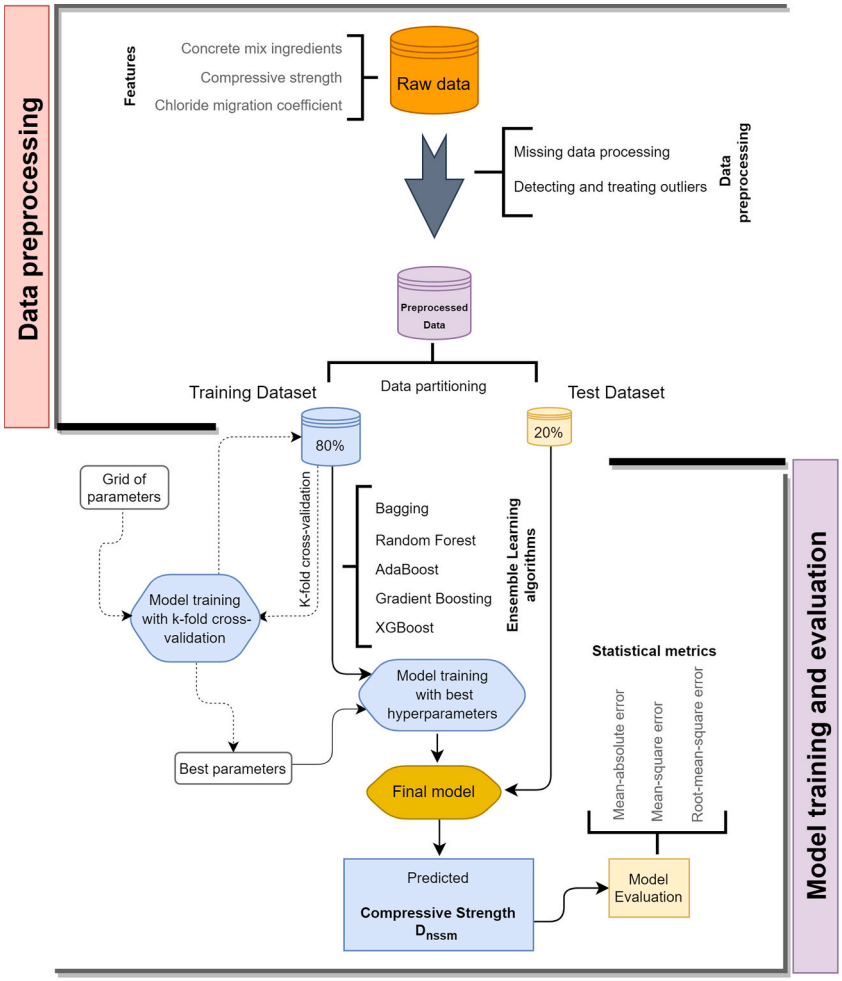

Fig. 2. Pipeline of the models.

4.3.1. *Missing data processing*

One of the most important components to improving the predictive accuracy of a data-driven model is the quality of the input data. Missing data is defined as values that do not exist in the given dataset for some features. It reduces the predictive ability of machine learning models and causes the model to become biased, and it is a problem affecting almost all scientific fields. For this study, all instances (but not the features) that contained missing values were removed from the dataset.

4.3.2. *Outlier detection and handling*

Outliers are unusual observations that are extremely distant from the rest of the population. Any data-driven model development process should include outlier detection and handling, since the performance of the model depends on the quality of the data. In this work, Mahalanobis Distance (MD) approach is used to detect multivariant outliers. This method is a widely used distance measure in multivariate space that accounts for the mean and covariance of the data and returns larger distances for observations that deviate from the mean in directions of less covariance. To identify multivariate outliers using MD, it is necessary to compare the Mahalanobis distance to a threshold value obtained from the chi-square distribution. Any instance is considered a multivariate outlier in this work if the probability associated with its Mahalanobis distance is 0.001 or less. Five instances were found to be outliers and discarded from the dataset. The MD between two objects X_A and X_B as well as the MD from each observation to the data center can be calculated by Equations (5) and (6) [58], respectively.

$$d = \left[(X_B - X_A)^T . C^{-1} . (X_B - X_A)\right]^{0.5} \tag{5}$$

$$d_i = \left[(X_i - \overline{X})^T . C^{-1} . (X_i - \overline{X})\right]^{0.5}, \tag{6}$$

where C is the covariance matrix of the sample, X_i is an object vector, and $\overline{X}$ is an arithmetic mean vector.

4.3.3. Data partitioning

Typically, training/test partitioning involves partitioning the data into a training and a test dataset in a specific ratio. The training dataset is used to train the model, while the test dataset is used to evaluate the fitted model's predictive performance on data it has never seen before. In this work, the data are randomly divided into two parts: 80% and 20%. Eighty percent of the data is used to train the models, and twenty percent of the dataset is used to test the performance of the models. The reason for the random 80/20 split ratio is to strike a balance between overfitting and underfitting. By using 80% of the data for training, the model can learn from an ample amount of data, while still leaving a sufficient amount of data for testing and evaluating how well the model generalizes to new data that it hasn't seen during training.

4.4. Data after preprocessing

Table 3 displays descriptive statistics for the preprocessed datasets of numeric input features used to train and validate the models. It can be noted that the final dataset has 76 observations. In this dataset, the w/b of the concrete ranged from 0.3 to 0.6. The cement content varies from 217 to 525 kg/m^3. The amount of SCMs fly ash, and silica fume used to partially replace Portland cement reached up to 216 kg/m^3 and 60 kg/m^3, respectively. The wide range of ingredient amounts confirms the dataset's inclusion of various types of concrete. The feature slag is not present in the cleaned dataset, as shown in Table 3. This is because the number of instances where slag was used as a partial replacement for cement was small, all instances with slag and the feature slag itself were removed from the dataset. The distribution of the target features, compressive strength and D_{nssm}, is shown in Fig. 3. It can be seen that the dataset covers a wide range of compressive strength (21–80 MPa) and D_{nssm} (1.57–31.30 $\times$ 10^{-12} m^2/s).

Despite the fact that a variety of non-numerical factors influence the properties of concrete in both direct and indirect ways, the only non-numerical component present in the raw dataset taken into consideration in this work is cement type. As with slag, when instances with missing values and outliers were removed from the dataset, only few types of cement remained. However, with the exception of CEM I (Portland cement), the number of observations was insignificant, retaining them in the data can cause a bias in model training, and thus all instances in which their cement types other than CEM I have been removed from the dataset. For this reason, the feature "Cement types" is no longer available in the preprocessed dataset.

Fig. 4 depicts the Pearson correlation coefficients between all possible features to aid in understanding the dependency between the features under consideration. It is the most commonly used metric for determining a linear relationship between two features that are measured on the same interval or ratio scale. Pearson coefficients range from +1 to −1, with +1 indicating a total positive correlation, −1 indicating a total negative correlation, and 0 indicating no correlation [59]. As shown in Fig. 4, some features, such as the w/b ratio and D_{nssm}, show positive strong correlations. There are also features with insignificant correlation, meaning the Pearson coefficient is less than 0.05, denoted by X.

4.5. Model training and evaluation

In this work, the model training activity takes into account two scenarios: Scenario 1 and 2. Both scenarios use the same decision tree ensemble methods, namely Bagging, random forest, AdaBoost, Gradient Boosting, and XGBoost. The only deference is the use of hyperparameters. Scenario 1 uses the default hyperparameter settings. Scenario 2 uses selected hyperparameters for each learning algorithms identified by training single-target models. The aim of this scenario is to investigate the effect of the identified hyperparameters based on single-target models on the performance of multitarget models. To realize this scenario, first, single-target models are trained with concrete mix ingredients as input features and compressive strength as a target feature. The hyperparameters of the adopted algorithms are then optimized using a grid search method, which performs an exhaustive search through a manually defined subset of a learning algorithm's hyperparameter space. The grid search was guided by k-fold cross-validation method. The value of k has a substantial impact on computation time, particularly when a grid search technique is used. After doing a sensitivity analysis on a range of values, it was found that a k value of 5 strikes a good balance between computational cost and a low bias in assessing model performance. Hence, 5-fold cross-validation coupled with grid search were employed to identify the optimal hyperparameters for each model. The hyperparameters with the highest prediction accuracy were then used to fit the multitarget regression, with compressive strength and D_{nssm} as target features. Table 4 summarizes all of the considered hyperparameters, as well as their grid search ranges and the identified optimal hyperparameters with their description.

The validity of any machine learning model must be assessed using a test dataset derived from the original data but not included in the training dataset. This is because the models can contain errors due to high bias and variance during training. High bias can lead to underfitting as the algorithm misses the relevant connection between the input and the target features. High variance can lead to

Table 3
Descriptive statistics of the preprocessed dataset's input features.

	w/b	Water	Cement	Fly ash	Silica fume	Total aggregate	Plasticizer	Superplasticizer	Air entraining
count	76	76	76	76	76	76	76	76	76
mean	0.42	172.09	384.14	26.24	7.26	1563.90	0.12	0.47	0.24
std	0.06	21.10	75.45	51.32	16.31	276.55	0.30	0.41	0.89
min	0.30	122.50	217.00	0.00	0.00	839.00	0.00	0.00	0.00
25%	0.35	158.00	323.75	0.00	0.00	1376.77	0.00	0.00	0.00
50%	0.41	171.00	383.00	0.00	0.00	1615.00	0.00	0.54	0.00
75%	0.45	189.00	450.00	13.50	0.00	1810.75	0.00	0.73	0.00
max	0.60	217.00	525.00	216.00	60.00	1950.00	0.89	1.56	5.80

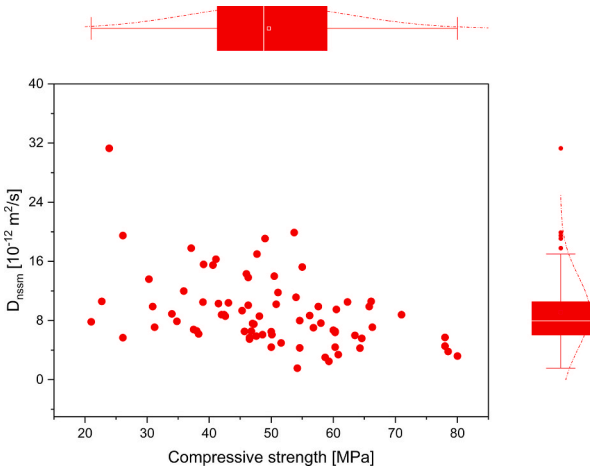

Fig. 3. Distribution of the target features.

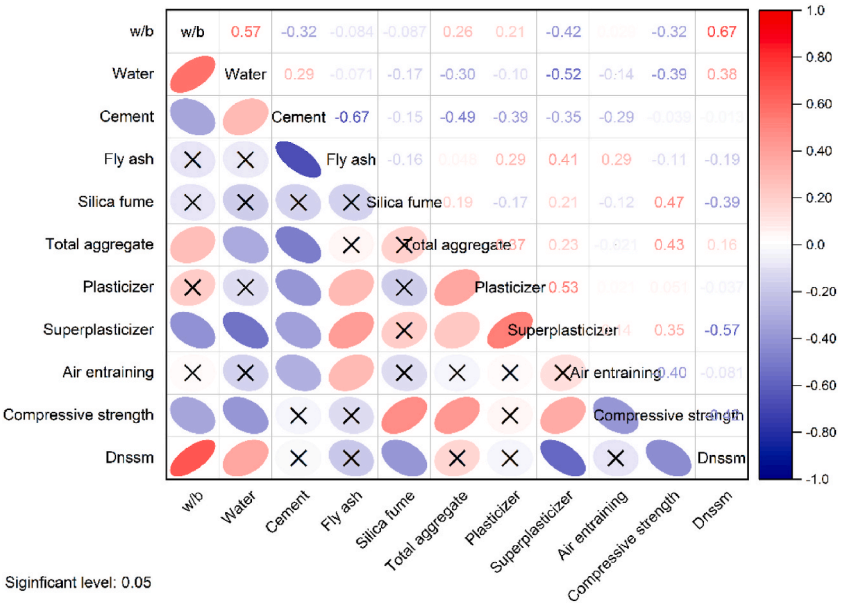

Fig. 4. Correlation plot showing the Pearson correlation of the features.

overfitting, causing the algorithm to model the random noise in the training dataset rather than the expected outputs [20]. The performance of all trained models was evaluated using the metrics of mean-absolute error (MAE), mean-square error (MSE), and root-mean-square error (RMSE) on unseen data. These metrics are the most widely used to evaluate the performance of regression models [60].

MAE is an average of the absolute errors (the difference between the actual and the predicted value), as in Equation (7) and is measured in the same units as the target feature. MAE is also known as the absolute loss.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{7}$$

MSE is the most commonly used loss function in regression models. It is computed by averaging the squared difference between the actual and predicted values, as specified by Equation (8).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{8}$$

RMSE is equal to the square root of the MSE. Sometimes it is preferred over MSE because MSE values are harder to understand due to the squaring effect. This is especially true when the target represents values in units of measurement. The RMSE is computed using

Table 4
Description of the considered range of hyperparameters with the optimal ones.

Learning algorithms	Hyperparameters	Grid search ranges	Optimal hyperparameters	Description of the hyperparameters
Bagging	n_estimators	[50,100,200]	50	The number of base estimators in the ensemble.
	max_features	[1,2,4,6,8]	8	The number of features to draw from the training input samples to train each base estimator.
	max_samples	[0.5,0.1]	0.5	The number of samples to draw from the training input samples to train each base estimator.
	bootstrap	[True, False]	True	Whether samples are drawn with replacement.
	bootstrap_features	[True, False]	False	Whether features are drawn with replacement.
Random forests	bootstrap	[True]	True	Whether bootstrap samples are used when building trees.
	max_depth	[80, 90, 100, 110]	110	The maximum depth of the tree.
	max_features	['auto', 'sqrt', 'log2']	log2	The number of features to consider when looking for the best split.
	min_samples_leaf	[3–5]	3	The minimum number of samples required to be at a leaf node.
	min_samples_split	[8,10,12]	8	The minimum number of samples required to split an internal node.
	n_estimators	[100, 200, 300, 1000]	300	The number of trees in the forest.
AdaBoost	learning_rate	[0.05,0.1,0.2,0.6,0.8,1]	1	Weight applied to each regressor at each boosting iteration.
	n_estimators	[50,60,100]	100	The maximum number of estimators at which boosting is terminated.
	loss	['linear', 'square', 'exponential']	linear	The loss function to use when updating the weights after each boosting iteration.
Gradient boosting	max_depth	[3,5–7]	0.1	Maximum depth of the individual regression estimators.
	max_features	['auto', 'sqrt', 'log2']	sqrt	The number of features to consider when looking for the best split.
	min_samples_split	[2,3,10]	10	The minimum number of samples required to split an internal node.
	min_samples_leaf	[1,3,10]	1	The minimum number of samples required to be at a leaf node.
	learning_rate	[0.05,0.1,0.2]	0.1	Shrinks the contribution of each tree.
	n_estimators	[10,30,50,70]	50	The number of boosting stages to perform.
XGBoost	n_estimators	[20, 50, 100, 300,500]	50	Number of gradient boosted trees.
	max_depth	[2,4,6,8,10]	2	Maximum depth of a tree.
	gamma	[0.0001, 0.001, 0.01]	0.0001	Minimum loss reduction required to make a further partition on a leaf node of the tree.
	learning_rate	[0.001, 0.01, 0.1, 0.3]	0.1	Step size shrinkage used in update to prevents overfitting.
	booster	['gbtree']	gbtree	The type of booster.

Equation (9).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} = \sqrt{MSE} \tag{9}$$

where n is the number of observations, y_i is the actual target value, $\hat{y}_i$ is the predicted output value.

5. Results and discussion

The performance of all the trained models on test dataset is evaluated using the statistical measures, namely MAE, MSE, and RMSE, and the results are presented in Table 5. The smaller these statistical errors, the better the model performs. After carefully examining the statistical measures of each learning algorithm, the best performing models that predict the compressive strength or D_{nssm} from each scenario are identified. Except for the prediction of compressive strength in Scenario 2, the Gradient boosting algorithm produces the best results almost in all conditions. The best performing models among all the choices are highlighted in green, while the second best are highlighted in dark yellow. The XGBoost algorithm predicts compressive strength best with test errors of (MAE = 6.264, MSE = 73.890, and RMSE = 8.596), and D_{nssm} Gradient Boosting performs best with (MAE = 1.363, MSE = 3.712, and RMSE = 1.927).

The residuals of the best performing models predicting compressive strength and D_{nssm} from both scenarios are shown in Fig. 5 as violin plots. The plots show the distribution of the residuals smoothed with kernel density curves that allow easy comparison. A small boxplot is located in the center of each density curve. The box portion of the boxplot defines the 25th and 75th percentiles. The 25th percentile is the value at which 25% of the data values are less than this value. The middle 50% of the data values fall between the 25th and 75th percentiles, and this range is known as the interquartile range (IQR). The lines extending parallel from the boxes are known as whiskers, and they are used to indicate variability outside the 25th and 75th percentiles. Whiskers are typically extended to 1.5 times the IQR. The IQR of Scenario 1 of the compressive strength is smaller than that of Scenario 1. Though the 50th percentile of the residuals in Scenario 1 is lower than those in Scenario 2, the median residual in Scenario 2 are significantly lower, almost zero, when

Table 5
Statistical validation metrics of the models.

			Algorithms				
			Bagging	Random forest	AdaBoost	Gradient boosting	XGBoost
Compressive strength	Scenario 1	MAE	6.986	7.254	8.152	5.918	7.602
		MSE	93.147	109.144	120.382	85.930	120.331
		RMSE	9.651	10.447	10.972	9.27	10.970
	Scenario 2	MAE	7.099	8.189	7.727	6.683	6.264
		MSE	93.311	122.101	114.646	83.369	73.890
		RMSE	9.660	11.050	10.707	9.131	8.596
D_{nssm}	Scenario 1	MAE	1.756	1.765	1.707	1.546	1.949
		MSE	6.934	5.948	5.259	4.695	5.595
		RMSE	2.633	2.439	2.293	2.167	2.365
	Scenario 2	MAE	1.743	1.788	1.612	1.363	1.588
		MSE	6.131	5.408	4.644	3.712	4.781
		RMSE	2.476	2.325	2.155	1.927	2.186

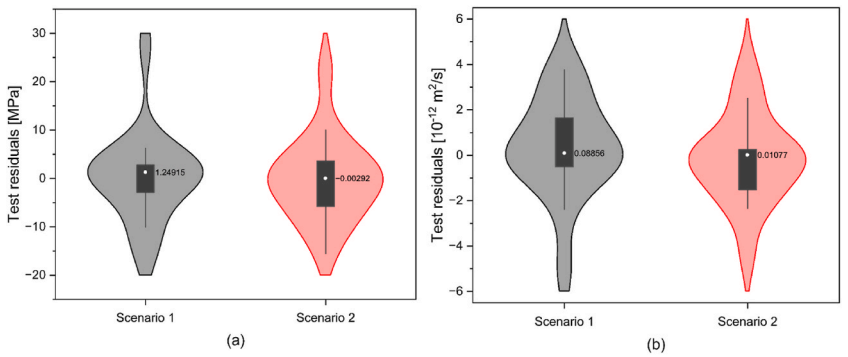

Fig. 5. Residual distribution of best-performing models: a) compressive Strength, (b) D_{nssm}.

compared to Scenario 1. This indicating it is superiority. In the case of D_{nssm}, it is noticeable that the median of Scenario 2 is smaller than Scenario 1. From a distribution standpoint, both scenarios of models that predict compressive strength have nearly identical distributions. However, in the case of D_{nssm}, the width of the probability density curve of Scenario 1 around the upper whisker area is wider. This means that residuals occur more frequently in this area, confirming that the model under Scenario 1 is less superior.

According to statistical evidence, the XGBoost and Gradient boosting algorithms best predict compressive strength and D_{nssm}. Nonetheless, unlike separate single-target models, this work aims to optimize concrete mixes that yield the desired properties of

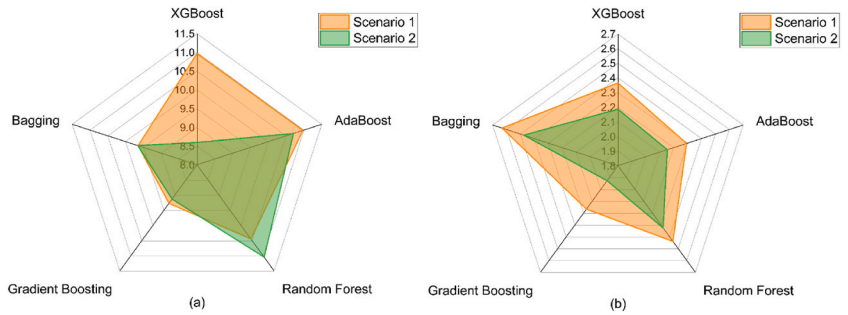

Fig. 6. RMSE of the models: (a) compressive strength, (b) D_{nssm}.

compressive strength and chloride resistance while simultaneously using these properties as multitarget features; thus, identifying a single best algorithm is critical. For this purpose, RMSE and MAE of each algorithm used to predict compressive strength and D_{nssm} of concrete in both scenarios are presented using radar plots in Figs. 6 and 7. The best algorithms have the lowest RMSE and MAE. As illustrated in the figures the Gradient boosting algorithm outperforms all other algorithms except for compressive strength prediction in Scenario 2, where it ranks in second next to XGBoost. As a result, the Gradient boosting algorithm could be used to optimize the concrete mix in order to produce concrete with the desired strength and chloride resistance.

Besides choosing the best performing algorithm, it is of paramount importance to identify the best modeling strategy between the two, Scenario 1 or Scenario 2. It can be seen from Figs. 6 and 7 that the RMSE and MAE of all algorithms that predict the D_{nssm} from Scenario 2 are noticeably lower than those from Scenario 1. In terms of compressive strength, most Scenario 2 learning algorithms outperform Scenario 1. This corroborates that the Scenario 2 modelling strategy is the best approach for accurately predicting compressive strength and D_{nssm} of concrete, thereby assisting in the optimization of concrete mix ingredients. This scenario first identifies optimal hyperparameters based on single-target models, then trains multitarget models with compressive strength and D_{nssm} as target features using the identified hyperparameters.

Among the five algorithms used, the Gradient boosting algorithm produced the best results, and from the two scenarios of modelling strategy considered, Scenario 2 appears to be the best, so the Gradient boosting algorithm of Scenario 2 can be chosen as the final compressive strength and D_{nssm} prediction model. As proofed by the statistical measurements, the performance of the model is quite remarkable, given that the datasets contain a small number of observations with different types of mix proportions, obtaining a wide range of strength and chloride resistance properties. As a result, it is possible to design optimal concrete mixes that meet the desired strength and resistance to chloride attack.

The performance ranking 1 to 5 (1 = top performer and 5 = least performer) of all the adopted learning algorithms is illustrated in Fig. 8. The ranking is based solely on prediction accuracy because the size of the dataset is small and including computational effort in the performance ranking makes no sense. It can be noticed that the ranking of all the algorithms in the case of compressive strength differs from Scenario 1 to Scenario 2. Gradient boosting, for example, ranks first in Scenario 1 but second in Scenario 2. AdaBoost ranks fifth (worst) in Scenario 1, but fourth in Scenario 2. In contrast to algorithms that predict compressive strength, the ranking of all algorithms that predict D_{nssm} obtained the same ranking in both scenarios. It is also worth noting that the Gradient boosting algorithm's high performance in predicting compressive strength and D_{nssm} does not imply that this algorithm is always the best. The performance may differ if the dataset is changed, so it is always worthwhile to compare different algorithms for the best results.

Compared to the conventional method, the machine learning-based model proposed in this work eliminates the need for error-prone and complicated analytical models. As a result, it reduces the requirement for time-consuming and resource-intensive laboratory tests. In addition, concrete design engineers do not need to possess extensive knowledge or experience in concrete science to use the proposed model. Moreover, the rapid delivery of results makes it a more appealing option than the traditional method. All of these factors, along with the model's anticipated economic impacts, will render the developed model highly valuable.

The limitation of this work could be attributed to the absence of extensive datasets and other features such as aggregate type, slump, various types of cement, and SCMs that could provide insights into the properties of concrete. The model's applicability is thus limited to a certain range of concrete types since it performs best when the input features fall within a specific range. This means that the model may not perform well for a wider range of concrete compositions beyond the input range it was trained on. To make the model universally applicable, it is necessary to expand the dataset to include a greater variety of concrete mixes, each with different ingredients and their fundamental properties. This would allow the model to learn from a more diverse set of data and be better equipped to handle a wider range of concrete compositions.

One way to obtain additional data is by collecting it from previously published studies, as was done in this work. However, collecting data from published studies presented some challenges. These included dealing with multiple measurement units, missing features, and varying methods of presenting data. In order to obtain comprehensive and accurate data, all of the collected information had to be translated into appropriate units and formats, which was a time-consuming and meticulous process. To address this issue, an open data exchange platform could be developed, allowing the scientific and concrete communities to share data in a standardized format. This would streamline the data collection process and make it easier to obtain a larger and more diverse dataset, ultimately leading to more robust and accurate models. The dataset used in this work is included as supplementary materials in this article.

6. Conclusions

In this work, single multitarget models were developed that use five tree-based machine learning algorithms to predict compressive strength and non-steady-state migration coefficient of concrete. Two scenarios were considered in the development of five models: Scenario 1 utilized the default hyperparameter settings, while Scenario 2 employed hyperparameters chosen from among those identified through training single-target models. The key findings from this research are as follows.

- Performance: The assessment outcomes validated that Gradient boosting was the most effective algorithm out of all the algorithms assessed, and Scenario 2 was the optimal modeling approach. It predicts compressive strength with (MAE = 6.683, MSE = 83.369, and RMSE = 9.131) and D_{nssm} with (MAE = 1.363, MSE = 3.712, and RMSE = 1.927).
- Implication: The high generalizability of the model corroborates its usefulness in designing optimal concrete mixes that meet the desired strength and resistance to chloride attack. Compared to traditional methods, it is more efficient, eliminating the need for multiple lab tests and saving time, money, and resources.

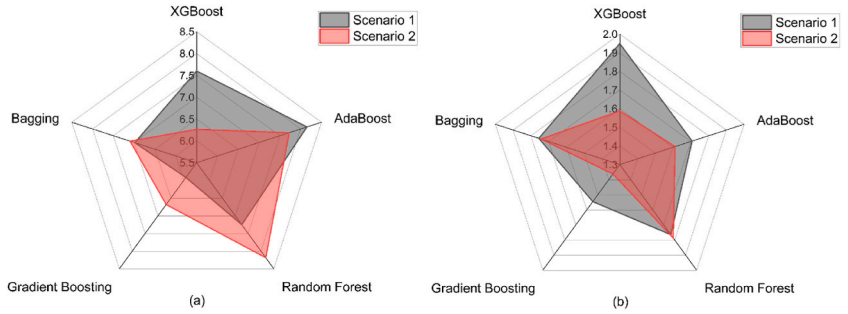

Fig. 7. MAE of the models: (a) compressive strength, (b) D_{nssm}.

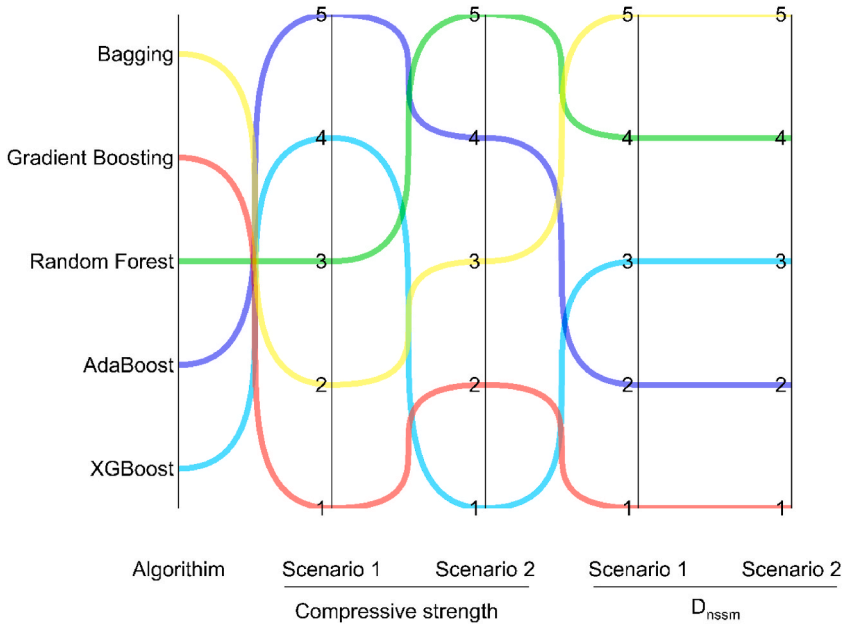

Fig. 8. Performance ranking of the adopted algorithms.

- Ease of use: The models are also simple to use and allow anyone to quickly produce the optimal concrete mixes without needing any prior knowledge or experience in concrete science.
- Adoptability: The model development strategy utilized can be replicated to carry out similar scientific investigations not only in the field of concrete science but also in other domains.
- Limitation: The study's limitation stems from the inadequate availability of comprehensive datasets and other features that could depict a vast array of concrete properties, constraining its practicality.

CRediT author statement

Woubishet Zewdu Taffese: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization, Funding acquisition. Leonardo Espinosa-Leal: Writing - Review & Editing, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data used in this work are included as a supplementary file.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.jobe.2023.106523.

References

[1] T. Luping, L.-O. Nilsson, P.M. Basheer, Resistance of Concrete to Chloride Ingress: Testing and Modelling, CRC Press, Boca Raton, FL, 2012, https://doi.org/10.1201/b12603.
[2] C. Andrade, D. Izquierdo, Statistical treatments of chloride threshold and corrosion propagation rate, Corrosion. Mater. Degrad. 3 (2022) 598–611, https://doi.org/10.3390/cmd3040032.
[3] R. Singh, Corrosion Control for Offshore Structures: Cathodic Protection and High Efficiency Coating, first ed., Gulf Professional Publishing, Oxford, UK, 2015 https://doi.org/10.1016/C2012-0-01231-8.
[4] W.Z. Taffese, E. Nigussie, Autonomous corrosion assessment of reinforced concrete structures: feasibility study, Sensors (Switzerland) 20 (2020) 6825, https://doi.org/10.3390/s20236825.
[5] R. Homayoonmehr, A.A. Ramezanianpour, M. Mirdarsoltany, Influence of metakaolin on fresh properties, mechanical properties and corrosion resistance of concrete and its sustainability issues: a review, J. Build. Eng. 44 (2021), 103011, https://doi.org/10.1016/j.jobe.2021.103011.
[6] ACI PRC-211, Standard Practice for Selecting Proportions for Normal, Heavyweight, and Mass Concrete, 2002.
[7] DoE, Design of Normal Concrete Mixes, 1988.
[8] NT BUILD 492, Concrete, Mortar and Cement-Based Repair Materials: Chloride Migration Coefficient from Non-steady-state Migration Experiments, NORDTEST, 1999.
[9] M. Shafikhani, S.E. Chidiac, Quantification of concrete chloride diffusion coefficient – a critical review, Cem. Concr. Compos. 99 (2019) 225–250, https://doi.org/10.1016/j.cemconcomp.2019.03.011.
[10] B. Yu, Q. Ma, H. Huang, Z. Chen, Probabilistic prediction model for chloride diffusion coefficient of concrete in terms of material parameters, Construct. Build. Mater. 215 (2019) 941–957, https://doi.org/10.1016/j.conbuildmat.2019.04.147.
[11] R. Homayoonmehr, A. Rahai, A. Akbar Ramezanianpour, Predicting the chloride diffusion coefficient and surface electrical resistivity of concrete using statistical regression-based models and its application in chloride-induced corrosion service life prediction of RC structures, Construct. Build. Mater. 357 (2022), 129351, https://doi.org/10.1016/j.conbuildmat.2022.129351.
[12] S.A. Endale, W.Z. Taffese, D.-H. Vo, M.D. Yehualaw, Rice husk ash in concrete, Sustainability 15 (2022) 137, https://doi.org/10.3390/su15010137.
[13] J. Zhang, Y. Huang, Y. Wang, G. Ma, Multi-objective optimization of concrete mixture proportions using machine learning and metaheuristic algorithms, Construct. Build. Mater. 253 (2020), 119208, https://doi.org/10.1016/j.conbuildmat.2020.119208.
[14] P. Ziolkowski, M. Niedostatkiewicz, S.B. Kang, Model-based adaptive machine learning approach in concrete mix design, Materials 14 (2021) 1661, https://doi.org/10.3390/ma14071661.
[15] P. Ziolkowski, M. Niedostatkiewicz, Machine learning techniques in concrete mix design, Materials 12 (2019) 1256, https://doi.org/10.3390/ma12081256.
[16] P. Pereira Dias, L. Bhagya Jayasinghe, D. Waldmann, Machine learning in mix design of Miscanthus lightweight concrete, Construct. Build. Mater. 302 (2021), 124191, https://doi.org/10.1016/j.conbuildmat.2021.124191.
[17] E. Mohammadi Golafshani, M. Arashpour, A. Kashani, Green mix design of rubbercrete using machine learning-based ensemble model and constrained multi-objective optimization, J. Clean. Prod. 327 (2021), 129518, https://doi.org/10.1016/j.jclepro.2021.129518.
[18] W.Z. Taffese, L. Espinosa-Leal, A machine learning method for predicting the chloride migration coefficient of concrete, Construct. Build. Mater. 348 (2022), 128566, https://doi.org/10.1016/j.conbuildmat.2022.128566.
[19] H. Borchani, G. Varando, C. Bielza, P. Larrañaga, A Survey on Multi-Output Regression, vol. 5, Wiley Interdiscip Rev Data Min Knowl Discov, 2015, pp. 216–233, https://doi.org/10.1002/widm.1157.
[20] S. Marsland, Machine Learning: an Algorithmic Perspective, second ed., CRC Press, Boca Raton, FL, USA, 2015.
[21] W.Z. Taffese, K.A. Abegaz, Prediction of compaction and strength properties of amended soil using machine learning, Buildings 12 (2022) 613, https://doi.org/10.3390/buildings12050613.
[22] E. Kalkan, S. Akbulut, A. Tortum, S. Celik, Prediction of the unconfined compressive strength of compacted granular soils by using inference systems, Environ. Geol. 58 (2009) 1429–1440, https://doi.org/10.1007/s00254-008-1645-x.
[23] W.Z. Taffese, K.A. Abegaz, Artificial intelligence for prediction of physical and mechanical properties of stabilized soil for affordable housing, Appl. Sci. 11 (2021) 7503, https://doi.org/10.3390/app11167503.
[24] H.-V.T. Mai, T.-A. Nguyen, H.-B. Ly, V.Q. Tran, Prediction compressive strength of concrete containing GGBFS using random forest model, Adv. Civ. Eng. (2021) 1–12, https://doi.org/10.1155/2021/6671448, 2021.
[25] W.Z. Taffese, E. Sistonen, Significance of chloride penetration controlling parameters in concrete: ensemble methods, Construct. Build. Mater. 139 (2017) 9–23, https://doi.org/10.1016/j.conbuildmat.2017.02.014.
[26] S. Pan, Z. Zheng, Z. Guo, H. Luo, An optimized XGBoost method for predicting reservoir porosity using petrophysical logs, J. Pet. Sci. Eng. 208 (2022), 109520, https://doi.org/10.1016/j.petrol.2021.109520.
[27] W.Z. Taffese, L. Espinosa-Leal, Prediction of chloride resistance level of concrete using machine learning for durability and service life assessment of building structures, J. Build. Eng. 60 (2022), 105146, https://doi.org/10.1016/j.jobe.2022.105146.
[28] X. Guo, P. Hao, Using a random forest model to predict the location of potential damage on asphalt pavement, Appl. Sci. 11 (2021), 10396, https://doi.org/10.3390/app112110396.
[29] W.Z. Taffese, Data-driven Method for Enhanced Corrosion Assessment of Reinforced Concrete Structures, University of Turku, 2020. https://www.utupub.fi/handle/10024/149752.
[30] E. Alpaydin, Introduction to Machine Learning, second ed., MIT press, Cambridge, MA, USA, 2020 https://doi.org/10.1017/S0269888910000056.
[31] P. Cichosz, Data Mining Algorithms: Explained Using R, John Wiley & Sons, Ltd, Chichester, United Kingdom, 2015, https://doi.org/10.1002/9781118950951.
[32] L. Breiman, Bagging predictors, Mach. Learn. 24 (1996) 123–140, https://doi.org/10.1007/BF00058655.
[33] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (1997) 119–139, https://doi.org/10.1006/jcss.1997.1504.
[34] J.H. Friedman, Greedy function approximation: a gradient boosting machine, Ann. Stat. 29 (2001) 1189–1232, https://doi.org/10.1214/aos/1013203451.
[35] T. Chen, C. Guestrin, XGBoost: a scalable tree boosting system, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794, https://doi.org/10.1145/2939672.2939785.
[36] H. Kuosa, Concrete Durability Field Testing in DuraInt-Project: Field and Laboratory Results 2007 - 2010, Espoo, 2011.
[37] J. Pontes, J.A. Bogas, S. Real, A. Silva, The rapid chloride migration test in assessing the chloride penetration resistance of normal and lightweight concrete, Appl. Sci. 11 (2021) 7251, https://doi.org/10.3390/app11167251.
[38] Y.C. Choi, B. Park, G.S. Pang, K.M. Lee, S. Choi, Modelling of chloride diffusivity in concrete considering effect of aggregates, Construct. Build. Mater. 136 (2017) 81–87, https://doi.org/10.1016/j.conbuildmat.2017.01.041.
[39] V. Elfmarkova, P. Spiesz, H.J.H. Brouwers, Determination of the chloride diffusion coefficient in blended cement mortars, Cement Concr. Res. 78 (2015) 190–199, https://doi.org/10.1016/j.cemconres.2015.06.014.

[40] K. Audenaert, Q. Yuan, G. De Schutter, On the time dependency of the chloride migration coefficient in concrete, Construct. Build. Mater. 24 (2010) 396–402, https://doi.org/10.1016/j.conbuildmat.2009.07.003.
[41] M. Marks, M.A. Glinicki, K. Gibas, Prediction of the chloride resistance of concrete modified with high calcium fly ash using machine learning, Materials 8 (2015) 8714–8727, https://doi.org/10.3390/ma8125483.
[42] M. Marks, D. Jóźwiak-NiedzWiedzka, M.A. Glinicki, Automatic categorization of chloride migration into concrete modified with CFBC ash, Comput. Concr. 9 (2012) 375–387, https://doi.org/10.12989/cac.2012.9.5.375.
[43] F.K. Sell Junior, G.B. Wally, F.R. Teixeira, F.C. Magalhães, Experimental assessment of accelerated test methods for determining chloride diffusion coefficient in concrete, Revista IBRACON De Estruturas E Materiais 14 (2021), https://doi.org/10.1590/s1983-41952021000400007.
[44] H.B. Hou, G.Z. Zhang, Assessment on chloride contaminated resistance of concrete with non-steady-state migration method, J. Wuhan Univ. Technol.-Materials Sci. Ed. 19 (2004) 6, https://doi.org/10.1007/bf02841355.
[45] R.W. Shiu, C.C. Yang, Evaluation of migration characteristics of opc and slag concrete from the rapid chloride migration test, J. Mar. Sci. Technol. 28 (2020) 69–79, https://doi.org/10.6119/JMST.202004_28(2).0001.
[46] M. Maes, E. Gruyaert, N. De Belie, Resistance of concrete with blast-furnace slag against chlorides, investigated by comparing chloride profiles after migration and diffusion, Mater. Struct. 46 (2013) 89–103, https://doi.org/10.1617/s11527-012-9885-3.
[47] J.A. Bogas, A. Gomes, Non-steady-state accelerated chloride penetration resistance of structural lightweight aggregate concrete, Cem. Concr. Compos. 60 (2015) 111–122, https://doi.org/10.1016/j.cemconcomp.2015.04.001.
[48] J. Jain, N. Neithalath, Electrical impedance analysis based quantification of microstructural changes in concretes due to non-steady state chloride migration, Mater. Chem. Phys. 129 (2011) 569–579, https://doi.org/10.1016/j.matchemphys.2011.04.057.
[49] X. Liu, K.S. Chia, M.H. Zhang, Water absorption, permeability, and resistance to chloride-ion penetration of lightweight aggregate concrete, Construct. Build. Mater. 25 (2011) 335–343, https://doi.org/10.1016/j.conbuildmat.2010.06.020.
[50] S. Real, J.A. Bogas, J. Pontes, Chloride migration in structural lightweight aggregate concrete produced with different binders, Construct. Build. Mater. 98 (2015) 425–436, https://doi.org/10.1016/j.conbuildmat.2015.08.080.
[51] C. Naito, J. Fox, P. Bocchini, M. Khazaali, Chloride migration characteristics and reliability of reinforced concrete highway structures in Pennsylvania, Construct. Build. Mater. 231 (2020), 117045, https://doi.org/10.1016/j.conbuildmat.2019.117045.
[52] J.-I. Park, K.-M. Lee, S.-O. Kwon, S.-H. Bae, S.-H. Jung, S.-W. Yoo, Diffusion decay coefficient for chloride ions of concrete containing mineral admixtures, Adv. Mater. Sci. Eng. (2016) 11, https://doi.org/10.1155/2016/2042918, 2016.
[53] X. Liu, H. Du, M.H. Zhang, A model to estimate the durability performance of both normal and light-weight concrete, Construct. Build. Mater. 80 (2015) 255–261, https://doi.org/10.1016/j.conbuildmat.2014.11.033.
[54] R. Van Noort, M. Hunger, P. Spiesz, Long-term chloride migration coefficient in slag cement-based concrete and resistivity as an alternative test method, Construct. Build. Mater. 115 (2016) 746–759, https://doi.org/10.1016/j.conbuildmat.2016.04.054.
[55] R.M. Ferreira, J.P. Castro-Gomes, P. Costa, R. Malheiro, Effect of metakaolin on the chloride ingress properties of concrete, KSCE J. Civ. Eng. 20 (2016) 1375–1384, https://doi.org/10.1007/s12205-015-0131-8.
[56] A. Pilvar, A.A. Ramezanianpour, H. Rajaie, S.M.M. Karein, Practical evaluation of rapid tests for assessing the Chloride resistance of concretes containing Silica Fume, Comput. Concr. 18 (2016) 793–806, https://doi.org/10.12989/cac.2016.18.6.793.
[57] J. Liu, X. Wang, Q. Qiu, G. Ou, F. Xing, Understanding the effect of curing age on the chloride resistance of fly ash blended concrete by rapid chloride migration test, Mater. Chem. Phys. 196 (2017) 315–323, https://doi.org/10.1016/j.matchemphys.2017.05.011.
[58] K. Varmuza, P. Filzmoser, Introduction to Multivariate Statistical Analysis in Chemometrics, CRC Press, 2016.
[59] D. Nettleton, Selection of variables and factor derivation, in: Commercial Data Mining: Processing, Analysis and Modeling for Predictive Analytics Projects, Morgan Kaufmann, 2014, pp. 79–104, https://doi.org/10.1016/B978-0-12-416602-8.00006-6.
[60] S. Vieira, W.H. Lopez Pinaya, A. Mechelli, Main concepts in machine learning, in: A. Mechelli, S. Vieira (Eds.), Machine Learning: Methods and Applications to Brain Disorders, Elsevier, 2020, pp. 21–44, https://doi.org/10.1016/B978-0-12-815739-8.00002-X.