Computers and Concrete, *Vol. 32, No. 3 (2023) 233-246*
https://doi.org/10.12989/cac.2023.32.3.233

Enhancing prediction accuracy of concrete compressive strength using stacking ensemble machine learning

Yunpeng Zhao*, Dimitrios Goulias[a] and Setare Saremi[b]

Department of Civil & Environmental Engineering, University of Maryland, College Park, MD, 20740, USA

(*Received June 23, 2022, Revised February 21, 2023, Accepted February 21, 2023*)

Abstract. Accurate prediction of concrete compressive strength can minimize the need for extensive, time-consuming, and costly mixture optimization testing and analysis. This study attempts to enhance the prediction accuracy of compressive strength using stacking ensemble machine learning (ML) with feature engineering techniques. Seven alternative ML models of increasing complexity were implemented and compared, including linear regression, SVM, decision tree, multiple layer perceptron, random forest, Xgboost and Adaboost. To further improve the prediction accuracy, a ML pipeline was proposed in which the feature engineering technique was implemented, and a two-layer stacked model was developed. The k-fold cross-validation approach was employed to optimize model parameters and train the stacked model. The stacked model showed superior performance in predicting concrete compressive strength with a correlation of determination (R^2) of 0.985. Feature (i.e., variable) importance was determined to demonstrate how useful the synthetic features are in prediction and provide better interpretability of the data and the model. The methodology in this study promotes a more thorough assessment of alternative ML algorithms and rather than focusing on any single ML model type for concrete compressive strength prediction.

Keywords: concrete mixture optimization; concrete strength; ensemble learning; feature engineering; machine learning; quality assurance; stacking

1. Introduction

Compressive strength is one of the most critical parameters to assess concrete quality in engineering applications (ACI 318 2019, Kosmatka and Wilson 2010). In order to meet compressive strength requirements, concrete mixture proportioning is often based on empirical prescriptive and/or performance-based mixture design methodologies as recommended by American Concrete Institute (ACI) (ACI 318 2019). A concrete mixture consists of various ingredients including cement, water, coarse and fine aggregate, and in several cases additives and admixtures. Understanding the relationship between concrete ingredients and strength is essential for optimizing concrete mixture proportioning and predicting compressive strength. However, the relationships among ingredients and compressive strength are often complex and highly nonlinear (Chou *et al.* 2011). Therefore, developing a comprehensive model for accurately predicting compressive strength is challenging. On the other hand, with the development of alternative machine learning (ML) techniques in recent years, there is growing interest in predicting concrete compressive strength with such a data-driven approach (Young *et al.* 2019). ML-based predictions have a significant advantage over the traditional approaches especially for handling nonlinear problems (Harun 2022, Kaveh *et al.* 2021, Onur 2021, Ahmed *et al.* 2021, Chou *et al.* 2011, Young *et al.* 2019, Salehi and Burgueño 2018, Yeh 1998). ML algorithms have also been extensively used for predicting the performance of other civil engineering materials and structures (Emad *et al.* 2022, Harandizadeh *et al.* 2021, Asteris *et al.* 2019, 2022, Lemonis *et al.* 2022, Kardani *et al.* 2022 and Psyllaki *et al.* 2018). ML algorithms consider the type and quantities of concrete ingredients as input variables to predict compressive strength (i.e., output/target variables). ML algorithms are capable of learning the relationship between the target and input variables without constraints on presumption. This approach also further provides greater flexibility to capture hidden, non-intuitive feature patterns directly from the input data.

*Corresponding author, Ph.D. Student
E-mail: zyp1015@umd.edu
[a]Associate Professor
E-mail: dgoulias@umd.edu
[b]Ph.D. Student
E-mail: sghahris@umd.edu

Literature review

The most common ML models used for forecasting the compressive strength of concrete can be categorized into four types: support vector machine (SVM), decision trees (DT), artificial neural networks (ANN), and ensemble learning algorithms (EA). Numerous studies demonstrated that ML techniques provide higher accuracy for concrete strength prediction than traditional statistical analysis (i.e., multivariate regression). For instance, a neural network with backpropagation was adapted to predict the compressive strength of high-performance concrete (HPC)

(Yeh 1998). The model incorporated concrete mixture ingredients and age as input variables. Results demonstrated that ANN exhibited good prediction performance, outperforming regression models in terms of accuracy. Several other studies have also reported the use of ANN in predicting concrete strength. For example, Duan *et al.* (2013) proposed an ANN model to predict the compressive strength of recycled aggregate concrete (RAC). The performance assessment revealed that ANN offered a fairly high accuracy with a coefficient of determination (R^2) of 0.995. However, this study is limited to an extremely small dataset (i.e., n=168) which may not sufficiently represent the predictor variable space. As such, the generalization ability of ANN needs to be validated. Siddique *et al.* (2011) compared the predictive performance of a simple backpropagation neural network on two concrete compressive strength databases. Their study also presented the relative importance of each input variable in prediction. Asteris *et al.* (2022) proposed an ANN model to predict the compressive strength of concrete with metakaolin and superplasticizer. The proposed model reveals the non-linearity between mixture proportions and strength, and presents the most important parameters (i.e., binder to sand ratio, water to binder ratio and coarse to fine aggregate ratio) affecting concrete strength. Asteris *et al.* (2018) also demonstrated the ability of ANN for accurately predicting the compressive strength of masonry.

Gupta (2008) investigated the potential use of SVM for predicting concrete compressive strength using two relatively small datasets (i.e., n=181 and 190). The results showed that SVM with radial basis function (RBF) can effectively predict compressive strength and provide a correlation of coefficient of 0.994. Ling *et al.* (2019) proposed an SVM model combined with a k-fold cross-validation technique to predict the compressive strength of concrete in the marine environment. The results showed that the ANN model outperformed ANN and DT. Chou *et al.* (2011) attempted to optimize the prediction accuracy of compressive strength of HPC by comparing different ML models, including multiple additive regression trees (MART), SVM, bagging, and ANN. The performance comparison indicated MART that was superior in prediction accuracy, computational efficiency, and aversion to overfitting. Young *et al.* (2019) compared the performance of four ML models (i.e., linear regression, ANNs, SVM, and random forest) for predicting both field and laboratory concrete data. In their study, the random forest provided the best prediction for both field (R^2=0.60) and laboratory data (R^2=0.86).

Feng *et al.* (2020) proposed an ensemble ML model (i.e., AdaBoost) that employed an adaptive boosting algorithm to establish a strong learner by integrating several weaker learners and reported promising accuracy. Zhang *et al.* (2019) developed a random forest (RF) to predict the compressive strength of self-compacting concrete. The model achieved high predictive accuracy indicated by a high correlation coefficient (R=0.97). Farooq *et al.* (2021) compared ensemble approaches with individual machine learners for predicting the compressive strength of HPC. The results revealed that ensemble models with boosting and bagging showed robust performance as compared to the individual approach (i.e., DT, ANN, and SVM). Duan *et al.* (2021) used four ML algorithms (i.e., SVR, ANN, Xgboost and adaptive neuro fuzzy inference system, ANFIS) with metaheuristic search algorithm (i.e., ICA) to predict the compressive strength of concrete with recycled aggregates. The results demonstrated that the proposed ICA-Xgboost model outperformed other models. Nguyen *et al.* (2021) proposed a semi-empirical framework to predict the concrete compressive strength and reveal the nonlinear relationships between the strength and mixtures proportions.

Since each ML technique has various advantages and drawbacks, the selection of the most suitable model is based on different criteria (i.e., the nature of the data, predictive accuracy, computational time). In practice it is impossible to know in advance which algorithm is most suitable for the particular dataset and prediction task at hand. As such, a systemic comparison of all common ML techniques for predicting concrete compressive strength is needed. Therefore, we considered a larger set of representative algorithms of different complexity (i.e., base models), from parametric regressions (i.e., linear regression) to non-parametric ML algorithms such as SVM, DT, multiple layer perceptron (MLP, a class of feedforward ANN), and ensemble models (i.e., RF, Adaboost, and Xgboost). A two-layer stacked ensemble ML model was developed using the based models to further improve the prediction accuracy and generalization performance. The stacked ensemble model alleviates concerns over selecting the one “right” algorithm while benefiting from considering a diverse set.

Additionally, the performance of the proposed ML models for predicting concrete strength from literature was often optimized by extensively tuning up the hyperparameters of each algorithm or developing a hybrid model based on conventional ML algorithms, which is considered time-consuming and computationally expensive as it requires testing numerous combinations before attaining the optimum values or models. Alternatively, this study presents the importance of exploratory data analytics and featuring engineering using domain knowledge to improve the prediction performance of concrete compressive strength. Feature (i.e., input variable) importance was computed to demonstrate the contribution of the synthetic features on prediction accuracy as compared to original features. Finally, the model performance from this study was compared with other previous from the literature to show the superiority of the proposed methodology.

Research significance

This study proposes a two-layer stacked ensemble model and presents the implementation of such model for predicting concrete compressive strength. The proposed model creates the optimal weighted combination of predictions from the candidate base algorithms, and thus provides superior prediction accuracy. In relation to the “model averaging” approach proposed by other studies, the suggested stacking used in this study has the advantage that any base model that performs poorly does not harm the performance of the stacking ensemble, since the meta learner would assign a weight of zero to such base leaner. Thus, such stacking ensemble model uses combinations of predictions from other learners to produce a superior prediction. This research also demonstrates the importance of the feature engineering, which promotes using domain knowledge to create synthetic features for improving the concrete strength prediction. The proposed approach encourages a more thorough assessment of alternative ML algorithms rather than focusing on any single ML model type for the focus case study of concrete compressive strength prediction.

2. Machine learning algorithms

This section briefly reviews the ML algorithms investigated and compared for optimizing the accuracy of concrete compressive strength.

2.1 Linear regression

Linear regression algorithm establishes a linear relationship between input and the target variable. For multidimensional inputs, this algorithm learns a mapping from D-dimensional inputs to scaler outputs: $x \in \mathbb{R}^D, y \in \mathbb{R}$ with the following equation

$$f(x) = w^T x + b = \sum_{i=1}^{D} w_i x_i + b \tag{1}$$

where w_i is the D-dimensional coefficient of the input variable x_i. In the proposed regression model, y represents concrete compressive strength, and x_i represents mixture components. The algorithm learns a vector of weights w by minimizing the least-squares cost function given by

$$E(\widetilde{w}) = \sum_{i=1}^{N} (y_i - \widetilde{w}^T \tilde{x})^2 \tag{2}$$

where

$$\widetilde{W} = \begin{bmatrix} w_1 \\ \vdots \\ w_D \\ b \end{bmatrix}, \tilde{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_D \\ l \end{bmatrix} \tag{3}$$

2.2 Decision tree algorithms

Decision tree algorithms build regression or classification models and represent the data in the form of a tree structure. The algorithm breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The condition, or test, is represented as the “leaf” (node) and the possible outcomes as “branches” (edges). This splitting process continues until no further gain can be made or a preset rule is met (i.e., the maximum depth of the tree is reached). The core algorithm, ID3, for building decision trees employs a top-down, greedy search through the space of possible branches with no backtracking. The algorithm creates a multi-way tree in which each node can have two or more edges to detect the categorical feature that will maximize the information gain using the impurity criterion-entropy. The ID3 algorithm can be used to construct a decision tree for regression by replacing information gain with standard deviation reduction. Classification and regression trees (CART) is another widely used decision tree algorithm developed by Breiman *et al.* (1984). The algorithm creates a binary tree (each node has exactly two outgoing edges) finding the best numerical or categorical feature to split using an appropriate impurity criterion. For regression, CART introduced variance reduction using least mean square error (MSE).

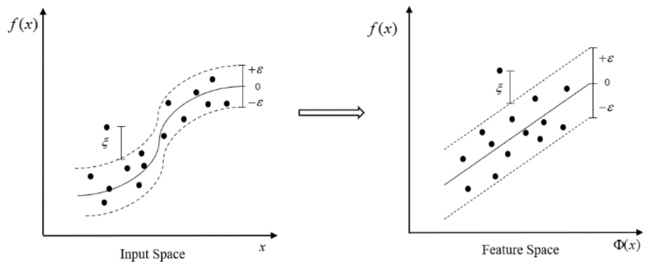


Fig. 1 Nonlinear transformation

2.3 Support vector machines

SVM is a supervised learning method that can be used for both nonlinear regression and classification analysis based on structural risk minimization (SRM). SVM was first proposed by Cortes and Vapnik (1995). The main concept of the SVM method is using an effective separation by a hyperplane, with the largest distance to the nearest training-data point and the lowest generalization error, to allow the SVM to minimize the error and obtain a better generalization. In support vector regression (SVR), the inputs from lower-dimensional input space are firstly mapped to a higher-dimension feature space using a nonlinear kernel function (e.g., polynomial function, sigmoidal function and Gaussian radial basis kernel functions), Fig. 1. In this feature space, the SVM attempts to construct a linear objective function so that its output has a maximum deviation of ε from the actual targets, y_i, in the training dataset. The linear objective function in the high-dimensional feature space is expressed as

$$f(x, w) = \sum_{i=1}^{n} w_i g_i(x) + b \tag{4}$$

Where $g_i(x)$ is a set of n nonlinear transformations, w_i represents the weight vector, and b is a bias term.

Compared to other machine learning algorithms, SVM has multiple advantages including a unique optimization approach and the effective utilization of high-dimensional feature spaces and computational learning theory.

2.4 Multiple layer perceptron

Artificial neural networks are models that attempt to determine the relationship between input and output variables by simulating the structure of the biological neural network (human brain). ANN models can efficiently solve multi-dimensional or multi-variable problems. A neural network consists of an input layer, output layer, and hidden layer(s) of neurons. The input-output relationship is contained in the connections between neurons. Neurons in the hidden layers represent the features of input relevant to output. Backpropagation and gradient descent algorithms were employed to update the weights and minimize the error. To train the neural network, first, the total error, which is the difference between NN output and target, is calculated. Then, the weight contributions (i.e., gradients) to the error are calculated at each connection. The error is calculated from the output, then moving back toward the inputs to calculate the gradient. Each weight keeps being updated and moves in the negative direction of the gradient to minimize the error, and until the desired error level is reached. The main challenge of ANN is that the selection of network size and parameters can be time-consuming.

2.5 Ensemble learning

Ensemble learning is the approach by which multiple learners (e.g., SVM, linear regression, decision tree, and NN) are strategically generated and combined to one predictive model in order to improve predictive performance or reduce bias and variance (Feng *et al.* 2020, Breiman 1996, Bühlmann and Yu 2002). There are three main classes of ensemble learning methods including bagging, boosting and stacking.

Bagging was first defined by Breiman as a method to generate different versions of a predictor leading to a more robust prediction (Breiman 1996). It is an ensemble algorithm that reduces the variance by applying bootstrap sampling to obtain data subsets and using these different versions of datasets to generate multiple models. The final prediction is obtained by averaging the outcomes of these models for regression and using plurality voting for classification. For example, N different trees are trained on different subsets of the data (chosen randomly with replacement) and the ensemble is computed as follows

$$f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x) \tag{8}$$

Random Forest is a typical bagging technique capable of performing both regression and classification with the use of decision trees as a base learner. The basic idea is to combine and train multiple decision trees using only a random subset of the input variables to determine the final output rather than relying on an individual decision tree.

In boosting, the base learners are built sequentially such that each subsequent learner aims to reduce the errors of the previous learner. The main principle of boosting is to fit a sequence of weak learners to weighted versions of the data. More weight is given to examples that were misclassified by earlier rounds. The predictions are then combined through a weighted majority vote (i.e., classification) or a weighted sum (i.e., regression) to produce the final prediction. The most widely used form of boosting algorithms is adaptive boosting (Adaboost) proposed by Freund *et al.* (1996). In the Adaboost algorithm, the first base learner (i.e., decision tree) is trained using equal weighting coefficients on the original dataset. Then the weighting coefficients are adjusted according to the error of the current prediction in the subsequent boosting rounds. In this method, more weights are assigned to incorrectly classified or predicted samples. In the end, the weak learners are assembled with different weights to a strong and robust learner.

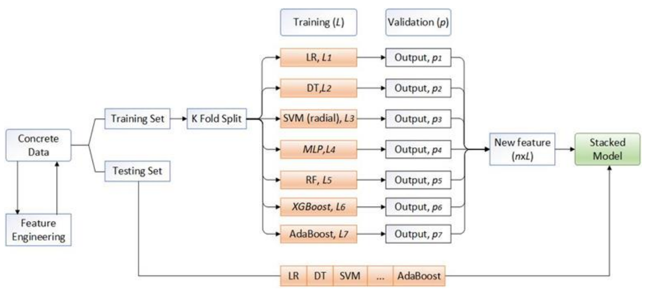


Fig. 2 Flow diagram of the proposed methodology

Stacking generalization is an ensemble learning method that learns how to optimally combine multiple ML algorithms using a new algorithm (i.e., meta model). In this study the base models were trained on a complete training dataset, while the meta model was trained on the final predictions of all the base-level models. The stacking approach considers heterogeneous weak learners whereas bagging and boosting consider mainly homogeneous weak learners. Stacking learns to combine the base models using a meta model while bagging and boosting combine weak learners following deterministic algorithms. As mentioned earlier, the stacking ensemble model approach used in this study learns how to optimally combine the predictions from the base models to improve predictions. In relation to the “model averaging” suggested in other studies, the suggested stacking approach used in this study has the advantage that any base model that performs poorly does not harm the performance of the stacking ensemble, since the meta learner would assign a weight of zero to such base leaner. Thus, such stacking ensemble model uses combinations of predictions from other learners to produce a superior prediction.

3. Methodology and implementation

Fig. 2 illustrates the implementation process of the proposed methodology, which consists of three steps: (1) data preparation and feature engineering, (2) training and optimizing individual models, (3) training the stacked model and generating predictions. The modelling and analysis are implemented in Python programming.

3.1 Data preparation

The database employed in this study includes 1031 laboratory measured HPC compressive strength from various mixtures (Yeh 1998). Even though this dataset has been used by several researchers for evaluating ML algorithms, a comparison of the model performances, included later in this paper, demonstrates the superiority of proposed models and feature engineering techniques. There

Table 1 Statistics of concrete mixtures variables

Features	Mean	Standard deviation	Minimum	Maximum	Correlation coefficient (with strength)
Cement (kg/m^3)	281.17	104.51	102.00	540.00	0.48
Fly ash (kg/m^3)	54.19	64.00	0.00	200.10	-0.05
Blast furnace slag (kg/m^3)	73.90	86.28	0.00	359.40	0.14
Water (kg/m^3)	181.57	21.35	121.80	247.00	-0.37
Superplasticizer (kg/m^3)	6.02	5.97	0.00	32.20	0.40
Coarse aggregate (kg/m^3)	972.92	77.75	801.00	1145.00	-0.17
Fine aggregate (kg/m^3)	773.58	80.18	594.00	992.60	-0.16
Age (days)	45.66	63.17	1.00	365.00	0.52
Water to cementitious materials ratio	0.46	0.12	0.25	0.90	-0.66
Aggregate to cement ratio	7.33	2.88	3.10	17.93	-0.47
Fresh density (kg/m^3)	2345.87	61.05	2194.60	2551.00	0.45
Compressive strength (MPa)	35.82	16.70	2.33	82.60	1.00

are eight independent variables (e.g., cement, blast furnace slag, fly ash, water, superplastic, coarse and fine aggregate, age) and one dependent variable (i.e., compressive strength). All the records are numeric, and no missing values have been observed in the dataset (Table 1). Data preparation involves best exposing the unknown underlying structure of the problem to learning algorithms by implementing the following tasks: exploratory data analysis, data cleaning, data transformation and scaling, feature engineering, and selection. The inter-quartile range (IQR) method was employed to perform outlier detection. Based on the results from IQR, a small number of data points (n=68) were considered as outliers. However, the study decided to keep these data since they may represent the variability inherent in the concrete mixture data. In addition to outlier detection, normalization and transformation are carried out to convert the input variables into the best structural representation for ML algorithms. Data is normalized through scale and center transformation to convert the initial variables to have a mean of 0 and a standard deviation of 1. The input variables are standardized using the following formula

$$z_i = \frac{(x_i - u)}{\sigma} \tag{9}$$

Where z_i is the standardized value of the original input variable x_i, μ *and* σ are mean and standard deviations of the input variables.

Power transforms are techniques for transforming numerical input variables to have a Gaussian or more-Gaussian-like probability distribution, which is preferred by many ML algorithms. In this study, a Box-Cox transformation is implemented to transform non-normal dependent variables into normal shapes. The formulation and implementation of such transformation can be found elsewhere (Box and Cox 1964).

Feature engineering is the process of using domain knowledge to extract features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data. Three synthetic features were created including water to cementitious materials ratio (*W/C* ratio), fresh density, and aggregate to cement ratio. Table 1 summarizes the general statistics of each variable. All the three new variables were extracted from the eight original dependent variables. For instance, the *W/C* ratio is obtained by dividing water by cementitious materials (i.e., cement, fly ash, and Blast furnace slag), while fresh density is calculated by the summation of seven ingredients in one cubic meter of the mixture. The correlation coefficients between the input and output variables were also presented in Table 1. These additional features show a relative strong correlation to the compressive strength and thus are anticipated to be useful for improving the predictive performance as they have been proven to have considerable influences on concrete compressive strength (Popovics and Ujhelyi 2008, Ni and Wang 2000). The feature importance was calculated in the later section to demonstrate how useful they are at predicting the strength as compared to the original variables.

3.2 Training base models

As shown in Fig. 2, seven base models were investigated including four conventional modes (i.e., LR, DT, and SVM) and three ensemble learners (RF, Xgboost, and Adaboost). Prior to model training and evaluation, the dataset was randomly portioned into training (85%) and testing (15%) sets. This ensures to examine the generalization ability of predictive models and avoid data leakage as the testing set does not involve model training in any way. The seven base ML models are separately trained and tuned using the training data. K-fold cross-validation (where k=5) is carried out to train the models and avoid overfitting. The hyperparameters were tuned using randomized search with cross-validation to optimize the models' predictions. Randomized search was used in which random combination of hyperparameters were selected to train each model. Firstly, the distribution for each hyperparameter was defined. The randomized search

Table 2 ML models with optimized hyperparameters

Model	Parameters	Setting
LR	polynomial features	2 degrees
	max depth	20
DT	min_samples_leaf	2
	min_samples_split	6
	kernel	RBF
SVM	regularization (C)	100
	gamma	2.0
	hidden layers	2
MLP	neurons	32
	activation	Relu
	number of trees	400
RF	min_samples_split	2
	max features	6
	number of trees	600
	learning rate	0.1
XGBoost	max depth	4
	subsample	0.8
	colsample_bytree	0.8
	number of trees	200
AdaBoost	loss	Square
	learning rate	0.3

algorithm was then randomly selected sample values for each hyperparameter from the corresponding distribution to train each model using such values. This process was repeated for a specified number of iterations, and the optimal hyperparameters were chosen based on the performance of the models. The root means square error (RMSE) is selected as the primary performance metric during the training process. The optimized hyperparameters for each model are listed in Table 2.

3.3 Training stacked model and generating predictions

Stacked generalization or stacking is an ensemble technique that uses a new algorithm (i.e., metal model) to learn how to best combine the predictions from two or more models trained on the dataset. In this study, a two-layer staked model is implemented to combine the base models and model configurations that were investigated. The first layer consists of the 7 base models (Fig. 2), and the outputs of these base models are used to train the stacked model (i.e., the second layer or meta-model). Firstly, a list of L base models (i.e., L=7) with specific model parameters is specified (Table 2). To train the stacked model, we need to make a loop for k-fold cross-validation. In each iteration, the training data is randomly divided into k blocks. The k-fold cross-validation is performed on each of the base models where all models use the same k-fold of the data, and the cross-validated predictions are collected. The predicted values represent $p_1, p_2 \cdots p_L$ (Eq. (5)). The n cross-validated predicted values from each of the L algorithms can be combined to form a new n×L feature matrix represented by Z in Eq. (10).

$$n\left\{\begin{bmatrix} p_1 \end{bmatrix}\begin{bmatrix} p_2 \end{bmatrix}\cdots\begin{bmatrix} p_L \end{bmatrix}\right. \rightarrow n\left\{\overbrace{\begin{bmatrix} \quad Z \quad \end{bmatrix}}^{L}\right.\begin{bmatrix} y \end{bmatrix} \tag{10}$$

Where n=number of rows in the training set. y=original response vector (i.e., target variable)

Once the feature matrix is obtained, a meta-learning algorithm needs to be specified. The meta-learning algorithm can be any of the base models investigated, but it is often suggested to use some form of regularization regression (Combrisson 2015). As such, lasso regression was selected as the meta-model. It should be noted that the cross-validated predictions from each of the base models became new features (i.e., feature matrix Z) for the meta-model. The feature matrix, along with the original response vector y was used to train the meta-model.

$$y = f(Z) \tag{11}$$

To make ensemble predictions, the predictions from each base model on the testing set need to be generated. These predictions were fed into the meta-model to generate the final prediction from the meta-model.

4. Model performance evaluation

4.1 Performance measures

The performance and accuracy of each ML model were evaluated by calculating performance measures including, root mean square error (RMSE), coefficient of determination (R^2), and mean absolute error (MAE). RMSE quantifies the averaged Euclidean distance between predicted and true strength data in the test set, while R^2 evaluates the accuracy of the predicted strength in terms of how close the data are from the fitted regression line (a perfect fit would provide an R^2=1). The performance measures are calculated based on the following equations

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_{pred}-y_{obs})^2}{\sum_{i=1}^{n}(y_{obs}-\bar{y}_{obs})^2} \tag{12}$$

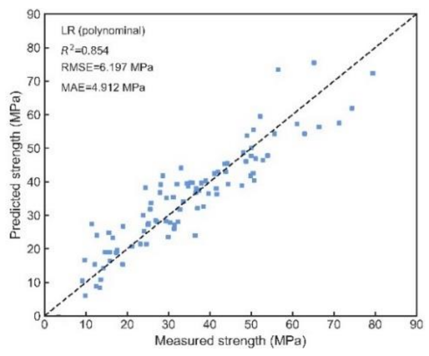
$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_{pred}-y_{obs})^2}{n}} \tag{13}$$

$$MAE = \frac{\sum_{i=1}^{n}|y_{pred}-y_{obs}|}{n} \tag{14}$$

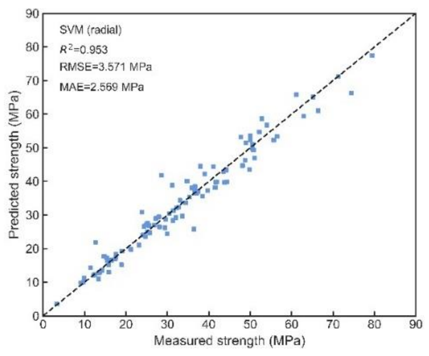
where y_{pred} and y_{obs} are the predicted and observed values, respectively, $\bar{y}_{obs}$ is the mean value of the observed data, n is the total number of samples in the data set.

4.2 Performance of base models

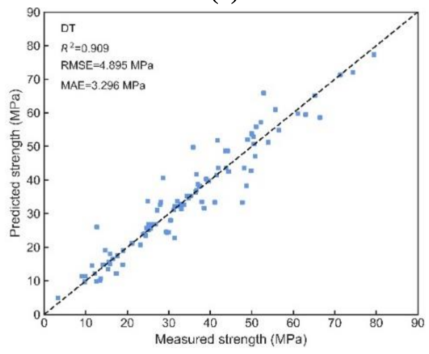
The prediction performance of the seven base models on the testing dataset is presented in Fig. 3. The scatter plots show the relation between the predicted and measured



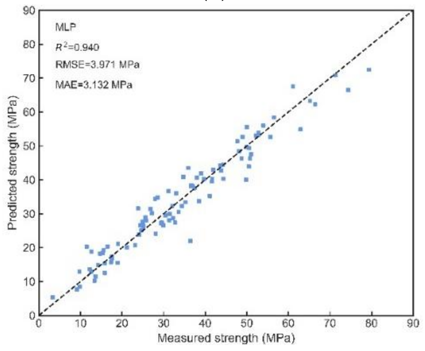
(a)



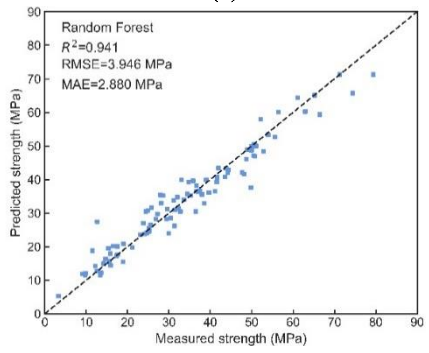
(b)



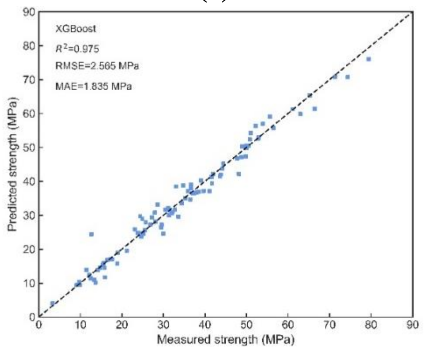
(c)



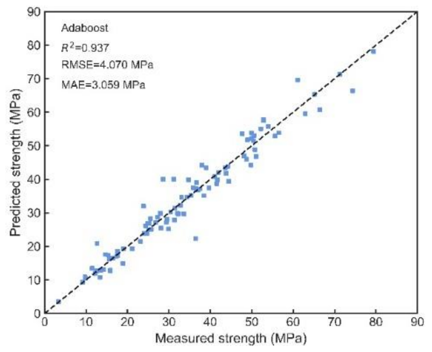
(d)



(e)



(f)



(g)

Fig. 3 Relationship between predicted and measured compressive strength (a) Linear regression, (b) Support vector machine, (c) Decision tree, (d) Multiple layer perceptron, (e) Random forest, (f) Xgboost and (g) Adaboost

strength for each model. A good model should have minimal discrepancies between the predicted and the measured values, or, in other words, all the data should be close to the regressed diagonal line. Linear regression is the first base model investigated in this analysis. The performance of linear regression is considered as a baseline for comparing the model performance of other ML models. For the linear regression model, a polynomial feature transform is implemented which raises the input variables to a polynomial power of two. This approach can help to better expose the complexity of interpreting the input variables and their relationships. The linear regression has a testing R^2, RMSE and MAE of 0.854, 6.197 MPa and 4.912 MPa, respectively, which is significantly better than linear models (i.e., R^2 of 0.611 and 0.66) developed on the same dataset with the original features (Chou *et al.* 2011 and Young *et al.* 2019). This demonstrates that polynomial transformation of the predictor variables improves the performance of the linear regression. Among the three individual ML models, the SVM made the best prediction (i.e., R^2=0.953, RMSE=2.569) which outperformed DT and MLP with a R^2 of 0.907 and 0.940 respectively. The performance of SVM with RBF kernels also demonstrates the importance of input variables transformation to linearly separate the patterns that exist among concrete ingredients.

The ensemble learning models show a better performance than the single learning model proposed in this study, except that AdaBoost generates a slightly lower accuracy (R^2=0.937, RMSE=2.569) than SVM and MLP. This indicates the reliable prediction capabilities of generalization for these ML techniques. In particular, The Xgboost exhibits the best predictive capability, which explains more variance in the data (R^2=0.975) and achieves higher accuracy (RMSE=2.565, MAE=1.835) in concrete strength prediction. The second-best predictive model is RF with an R^2, RMSE, and MAE of 0.941, 3.946 MPa, and 2.880 MPa, respectively. The superior prediction performance of Xgboost and RF may be attributed to the ability of tree-based methods to learn inconsistent variable importance in the data.

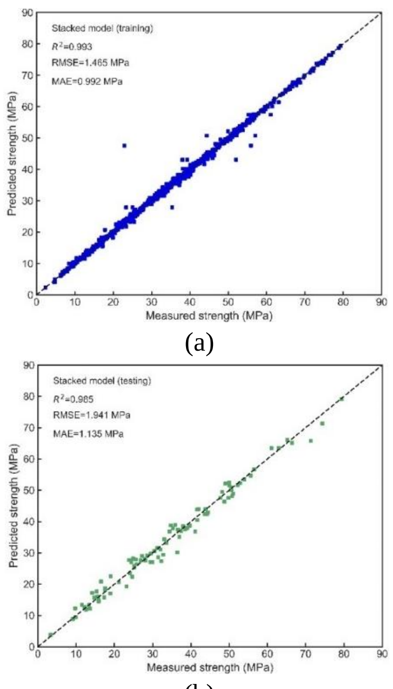


Fig. 4 Predicted vs. measured strength for stacked model (a) On training dataset and (b) On testing dataset

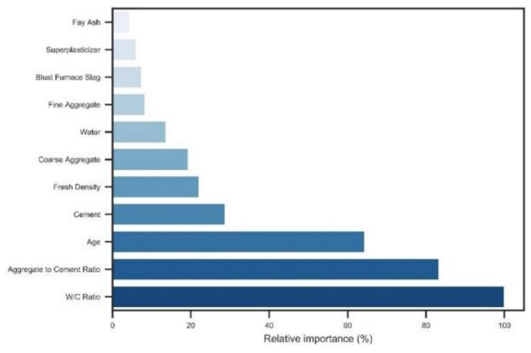
4.3 Performance of stacked model

The predictive performance of the stacked ensemble is presented in Fig. 4. For both training and testing dataset, a strong linear relation (R^2 of 0.993 and 0.985, respectively) between the predicted and measured values is observed. The stacked model achieves a considerably high accuracy on the testing data with a RMSE of 1.941 MPa and MAE of 1.135 MPa, which outperforms any of the base models. This indicates that the stacked model learns the optimal combination of the base learners, and thus improves the prediction performance. The theoretical detail on the optimality of stacking is explained by Van der Laan *et al.* (2007). As shown in Fig. 4, most of the points for testing data are either on or close to the diagonal line indicating high accuracy of prediction. However, for extremely large values (i.e., compressive strength >70 MPa), the model slightly underestimates the actual compressive strength. The reason could be that the base models steadily underestimate the actual strength for these data points as shown in Fig. 3. Further residual or error analysis (i.e., residual normality and heteroscedasticity) could be conducted to check whether these data points are outliers.

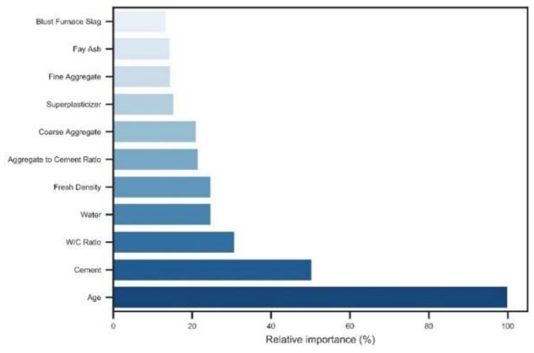
In order to assess which learning algorithm (i.e., "learner") is more suitable for the particular dataset, different types of base algorithms are explored. When the underlying functional form, relating the various material properties to strength in this case, is simple each algorithm may be able to provide good predictions. However, in this case due to the complexity of such relationship, one type of algorithm may be more successful than another. For instance, unlike a main terms parametric model, a tree-based algorithm like random forest inherently considers interactions and is unaffected by monotone transforms of the data. Since the true functional form of the parameters is unknown, there is a need to explore alternative base learners. Therefore, in this study alternative models were considered, from the standard parametric models (e.g., linear regression) to the more complex data-adaptive models (e.g., SVM, tree-based algorithms, and multiple layer perceptron). Furthermore, in this effort when a base learner performs poorly it does not harm the performance of the stacking ensemble since the meta learner would assign a weight of zero to such leaner. The advantage of the stacking ensemble model is that considers a combination of predictions from other learners to produce a superior prediction.

The meta-learning algorithm is often some sort of regularized linear model which provides a smooth interpretation prediction of the predictions generated from the base models (Phillips *et al.* 2022). Using a simple linear regression as the meta model may also reduce the chance of overfitting the predictions from the base models Moreover, predictions from the base models are usually strongly correlated, as they are trying to predict the same relationship. Therefore, a lasso regression with regularization parameters was used to deal with the correlations between the predictions of the base models.

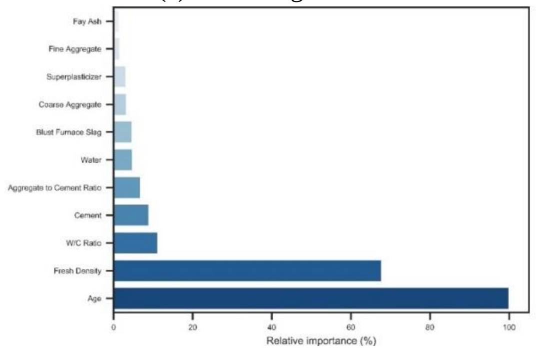
Although different combinations of base models may affect slightly the accuracy of the stacked model, the stacking is expected to perform better than or equal to any of the base models. Higher performance gains are usually produced when stacking base learners have high variability and uncorrelated predicted values (Grolemund 2014). The cross-validated predictions for the base models represent the model's generalization capability in making predictions



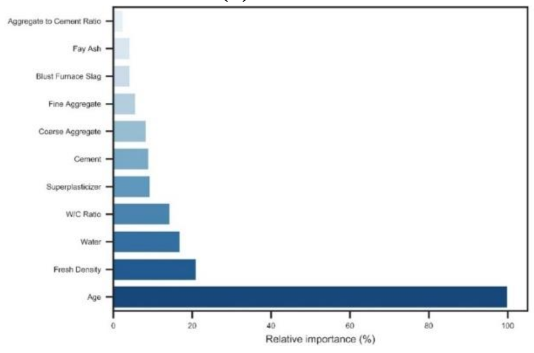
(a) Linear regression



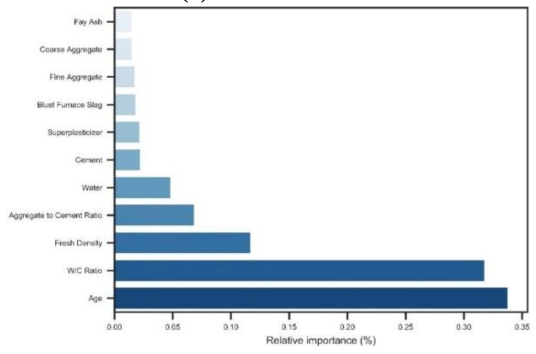
(b) SVM



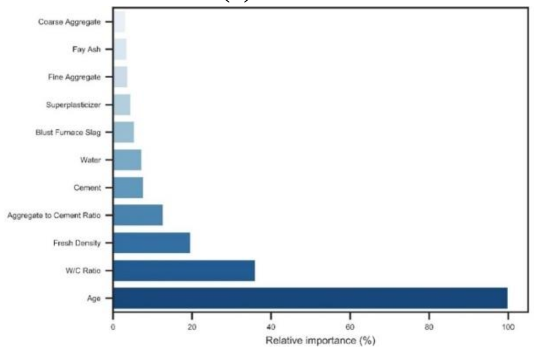
(c) Decision tree



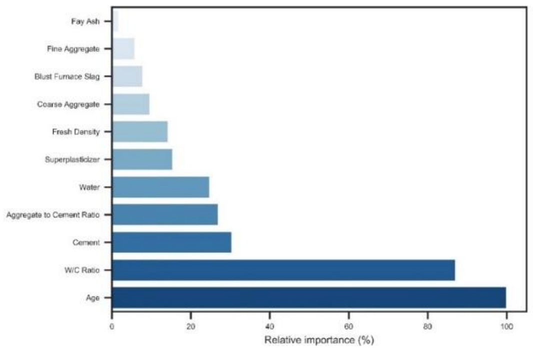
(d) MLP



(e) Random forest



(f) Xgboost



(g) Adaboost

Fig. 5 Relative importance of input variables for each model

on data not seen during training (i.e., cross-validation). A 5-fold cross-validation was used to find the optimal weighted combination of predictions from the candidate algorithms (i.e., base models) By training a meta-model with out-of-sample (i.e., testing data) predictions of the base models, the meta-model learns how to both correct and best combine the out-of-sample predictions from multiple models. As such the stacked model is unlikely to overfit the training data. When comparing to the performance of the base models (i.e., Xgboost), the stacked model increases 0.011 in terms of the R^2 score. The dataset used in this study is relatively small and less complex, which means that it is not easy to capture any of the patterns that the base algorithms (i.e., RF, Xgboost) could capture already. However, the results still demonstrate the superiority of stacking which produces better prediction accuracy. The

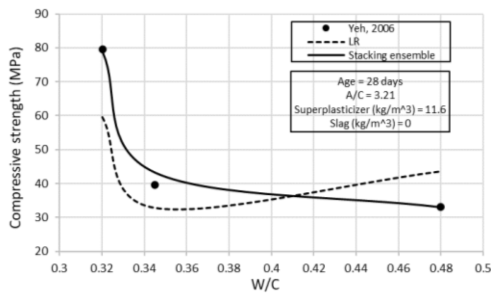


Fig. 6 Predicted compressive strength in relation to w/c

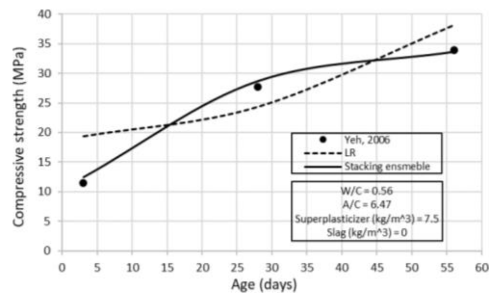


Fig. 7 Predicted compressive strength in relation to age

prediction performance is expected to be optimized to a higher degree for more complex concrete datasets (e.g., field concrete data).

4.4 Feature importance

Feature importance is used to describe how important the feature is at predicting the target variable. More precisely, it is referred to as a measure of the individual contribution of the corresponding feature for a particular model, regardless of the shape (e.g., linear or nonlinear relationship) or direction of the feature effect (Combrisson 2015, Fisher *et al.* 2019, Zhang *et al.* 2021). This means that the feature importance of the input data depends on the corresponding ML model. Feature importance provides better interpretability of the data and the model, and sometimes improves the performance of the model by implementing feature selection techniques.

There are three main types of metrics for determining feature importance, including model coefficients, the mean decrease in impurity (MDI), and permutation feature importance. For parametric linear models (e.g., LR, logistic regression, lasso, and ridge), the input variables are evaluated against the absolute normalized values of regression coefficients. Tree-based models (e.g., DT, RF and Xgboost) provide an alternative measure of feature importance based on MDI, which is quantified by the splitting of the decision tree. For example, the input variables of RF are evaluated by the mean square error (MSE) on the out-of-bag data for each tree before and after permuting the variables. The variable importance is determined by the averages and normalized variations of MS. For Xgboost, in each boosting iteration, the decrease in the loss function (i.e., MSE) in association with each input variable at each split is noted and summed (Zhang *et al.* 2021). Permutation feature importance is a technique for determining a relative importance score that is dependent on the model employed. As such it is used for models (e.g., SVM and MLP) that do not support native feature importance scores. The importance of a feature is measured by calculating the increase in the model's prediction error after permuting the feature (Interpretable machine learning reference). A feature is considered as "important" if shuffling its values increases the model error because in this case, the model relied on the feature for the prediction. The employed algorithm for the permutation feature is based on Fisher *et al.* (2019).

The relative importance of the input variables for each model is presented in Fig. 5. It can be observed that age, water, fresh density, cement, W/C ratio and aggregate to cement ratio play a more significant role than the rest of the variables for predicting compressive strength. This also demonstrates the value of synthetic features (i.e., fresh density, W/C ratio, and aggregate to cement ratio) and the importance of feature engineering for improving prediction performance. In particular, age is determined to be the most important variable in prediction for nonlinear models, while the W/C ratio made the biggest contribution for linear regression. This may be because there is a strong correlation (-0.66) between W/C and compressive strength, which can be easily captured by linear models. Even though no significant correlation of coefficient is observed between age and strength, the degree of hydration is synonymous with the age of concrete, which significantly increases the compressive strength by 28 days. ML models provide greater flexibility to capture such nonlinear patterns. The other two new features (i.e., aggregate to cement ratio and density) are also proven to be useful in predicting concrete strength as shown in Fig. 5. These observations tend to be consistent with engineering practice and the physical properties of regular concrete mixtures. Therefore, the ML models and feature importance techniques may be useful and reliable to analyze the relation and interaction between the ingredients and strength of innovative mixtures (e.g., lightweight, self-healing and green concrete mixtures).

To reveal the nonlinearity between concrete mixture ingredients and compressive strength, the predicted strength from the stacking ensemble model was plotted for different w/c and age while other input parameters (e.g., coarse aggregates, fine aggregates and slag) remained constant. As shown in Fig. 6, a nonlinear relationship between compressive strength and w/c is observed. Overall, the compressive decreases as the w/c increases. Fig. 7 indicates that the compressive strength increases significantly during the early ages, whereas the rate of increase is diminished after day 30. Such findings are in agreement with the findings reported in past studies (Asteris *et al.* 2021 and 2022). Figs. 6 and 7 also demonstrate that the stacking ensemble is capable of capturing the nonlinear relationship between input parameters and strength without overfitting the dataset.

5. Comparison with models from previous studies

Several studies have proposed models to predict

Table 3 Comparison of ML model performance on the HPC dataset

ML algorithms	R^2	RMSE (MPa)	MAE (MPa)	Remarks**	Reference
LR	0.854	6.197	4.912		*
	0.779	N/A	N/A	Input variables: water to binder ratio and age	Yeh (1998)
	0.611	10.428	N/A		Chou *et al.* (2011)
	0.660	8.800	N/A		Young *et al.* (2019)
DT	0.909	4.895	3.296		*
	0.911	4.948	N/A	Employed MART (i.e., gradient boosting)	Chou *et al.* (2011)
	N/A	7.840	5.860		Chou *et al.* (2014)
	N/A	7.37	4.62		Farooq *et al.* (2021)
SVM	0.953	3.571	2.569	Radial kernel	*
	0.886	5.572	N/A		Chou *et al.* (2011)
	N/A	5.590	3.750		Chou *et al.* (2014)
	0.830	6.400	N/A		Young *et al.* (2019)
ANN	0.940	3.971	3.132	Simple MLP with two hidden layers	*
	0.942	4.050	2.850	Proposed a high order deep neural network	Nguyen *et al.* (2019)
	0.953	5.750	4.830	Employed gradient boosted ANN	Erdal *et al.* (2013)
	0.914	N/A	N/A	Hand tuning for hyperparameters	Yeh *et al.* (1998)
RF	0.941	3.946	2.880		*
	0.850	5.800	N/A		Young *et al.* (2019)
	0.965	4.433	3.105	new feature: coarse aggregate to binder ratio	Han *et al.* (2019)
	0.922	4.6	3.23		Farooq *et al.* (2021)
Xgboost	0.975	2.565	1.835		*
	0.930	4.640			*
	0.980	2.650	1.890	With feature selection and hyperparameter optimization	Chakraborty *et al.* (2021)
	0.902	5.170	3.710		Farooq *et al.* (2021)
Adaboost	0.937	4.070	3.059		Nguyen-Sy *et al.* (2020)
	0.982	2.200	1.640	With hyperparameter optimization	Feng *et al.* (2020)
	0.919	5.220	3.690		Farooq *et al.* (2021)
Stacked model	**0.985**	**1.94**	**1.135**		*
	N/A	5.08	3.520	Stacking of three learners: SVM, DT, and LR	Chou *et al.* (2014)

Note: *models developed in this study, **1031 samples/datapoints and the following parameters were considered, cement, fly ash, blast furnace slag, water, superplasticizer, coarse aggregate, fine aggregate, age, unless otherwise specified in the “Remarks” column.

concrete compressive strength using the adapted HPC dataset. Table 3 provides a comparison of models’ performance. For the four individual models, this study significantly improves the prediction performance except for ANN, which may be largely attributed to the newly created variables. For instance, the linear regression achieves an R^2 of 0.859 which is significantly higher than other developed linear models. There are two possible reasons for such an improvement. One is that the power transformation makes the input variables more normally distributed which is preferred in linear models. A second possible reason is that W/C ratio has a higher linear correlation with strength which improves the prediction performance. This is also demonstrated by Yeh’s results where an R^2 of 0.779 is obtained using only the W/C ratio and age as input variables (Yeh 1998). Additionally, this study improves the performance of the SVM model significantly with an R^2 of 0.953 compared to Chou et al with R^2 of 0.953 (Chou *et al.*, 2011). Even though the ANN generates slightly lower accuracy as compared to Nguyen *et al.* and Erdal *et al.*, it should be noted that this study simply employs an MLP with 2 hidden layers while other studies developed much more complex hybrid models which could be time-consuming and computationally expensive (Nguyen-Sy *et al.* 2020, Erdal *et al.* 2013).

In terms of ensemble learning, the accuracy of the RF model in this study is slightly lower compared to Han *et al.* (2019). It is worth mentioning that they also created a new variable (i.e., coarse aggregate to binder ratio) to train the RF model and conducted hyperparameter optimization. The Xgboost model developed in the present study provides similar performance in terms of R^2, RMSE and MAE compared to Chakraborty *et al.* (2021). The AdaBoost of Feng *et al.* provided a better prediction performance than the present study by performing a hyperparameter optimization (Feng *et al.* 2020). Chou *et al.* (2019) implemented stacking of LR, DT, and SVM, however, resulted in much lower accuracy. Their results showed that the stacking outperformed any of the base learners, however, due to the poor performance of the base models, the stacking did not provide a competitive prediction ability. Notably, the stacked model proposed in this study outperforms other models reported from the literature. For the two-layer stacked model, ensembling stronger regressors in the first layer is likely to improve the performance of the model, while a simple, explainable, parametric model (i.e., Lasso) serves better in the second layer. The mechanism behind the more efficient learning ability of the proposed model is related to the elements in the proposed methodology (Fig. 2), including data processing, feature engineering, base model development, hyperparameters tuning, and stacking.

6. Conclusions

The goal of this study was to explore alternative ML models for enhancing the prediction of compressive strength as a function of mixture ingredients and proportions. Seven individual ML techniques were considered and compared including LR, DT, SVM, MLP, RF, Xgboost, and Adaboost. To improve the prediction accuracy, a methodology was proposed in which the feature engineering technique was implemented, and a two-layer stacked model was proposed. The k-fold cross-validation approach was employed to optimize model parameters and train the stacked model. The performance of these models was evaluated and compared with R^2, RMSE, and MAE. Furthermore, the relative importance of the input variables in predicting compressive strength was assessed.

The results showed that, among the seven individual models, the Xgboost exhibited the best predictive performance with an R^2 of 0.975 followed by SVM which generated an R^2 of 0.953. The results for feature importance showed that age, water, fresh density, cement, W/C ratio, and aggregate to cement ratio play a higher role in predicting compressive strength than the rest of the variables. The proposed stacked models in this study outperformed other models reported in the literature with a testing R^2, RMSE, and MAE of 0.985, 1.941 MPa, and 1.135 MPa. This approach encourages a more thorough consideration of alternative ML algorithms rather than focusing on a single machine learning approach. While the implementation process for the proposed methodology was presented, the study results indicated that the ML models (i.e., SVM, stacked model) can provide fairly high accuracy of concrete strength predictions. Thus, such ML models can be used to accurately predict compressive strength, minimizing the need for time-consuming and costly testing for mix design optimization and quality assurance during production.

While the ML models proposed herein have shown to be very effective in predicting concrete strength, model generalization to a wider set of data and concrete mixtures will be beneficial for their widespread calibration and use. The superiority of the proposed methodology and stacking is expected to be further optimized with more complex concrete data. Such effort is currently underway with datasets from field-produced concrete where higher noise and variance caused by uncontrolled or unreported production processing and construction variables may exist. Such datasets may eventually include concrete mixtures such as self-consolidated concrete, fiber-reinforced concrete, and other mixtures. In a similar direction, the proposed ML models, and/or modeling techniques, could be calibrated to other regions by using concrete datasets reflecting concrete materials and mixtures elsewhere.

Acknowledgements

This research is funded by the Maryland Transportation Institute from the University of Maryland, College Park. The authors are thankful to Dr. Mark Fuge for giving insightful suggestions for the research.

References

ACI Committee 318 (2019), Building Code Requirements for Reinforced Concrete, American Concrete Institute, Farmington Hills, MI, USA.

Armaghani, D.J., Hatzigeorgiou, G.D., Karamani, C., Skentou, A., Zoumpoulaki, I. and Asteris, P.G. (2019), “Soft computing-based techniques for concrete beams shear strength”, *Procedia Struct. Integr.*, **17**, 924-933. https://doi.org/10.1016/j.prostr.2019.08.123.

Asteris, P.G., Argyropoulos, I., Cavaleri, L., Rodrigues, H., Varum, H., Thomas, J. and Lourenço, P.B. (2018), “Masonry compressive strength prediction using artificial neural networks”, *International Conference on Transdisciplinary Multispectral Modeling and Cooperation for the Preservation of Cultural Heritage*, Athens, Greece, October.

Asteris, P.G., Lourenço, P.B., Adami, C.A., Roussis, P.C., Armaghani, D.J., Cavaleri, L., Chalioris, C.E., Hajihassani, M., Lemonis, M.E., Mohammed, A.S. and Pilakoutas, K. (2021), “Revealing the nature of metakaolin-based concrete materials using artificial intelligence techniques”, *Constr. Build. Mater.*, **322**, 126500. https://doi.org/10.1016/j.conbuildmat.2022.126500.

Asteris, P.G., Nozhati, S., Nikoo, M., Cavaleri, L. and Nikoo, M. (2019), “Krill herd algorithm-based neural network in structural seismic reliability evaluation”, *Mech. Adv. Mater. Struct.*, **26**(13), 1146-1153. https://doi.org/10.1080/15376494.2018.1430874.

Asteris, P.G., Rizal, F.I.M., Koopialipoor, M., Roussis, P.C., Ferentinou, M., Armaghani, D.J. and Gordan, B. (2022), “Slope stability classification under seismic conditions using several tree-based intelligent techniques”, *Appl. Sci.*, **12**, 1753.

https://doi.org/10.3390/app12031753.

Box, G.E.P. and Cox, D.R. (1964), "An analysis of transformations", *J. Royal Stat. Soc. Ser. B: Stat. Methodol.*, **26**(2), 211-243. https://doi.org/10.1111/j.2517-6161.1964.tb00553.x.

Breiman, L. (1996), "Bagging predictors", *Mach. Learn.*, **24**, 123-140. https://doi.org/10.1007/BF00058655.

Breiman, L., Friedman, J., Stone, C.J. and Olshen, R. (1984), *Classification and Regression Trees*, Taylor & Francis, New York, NY, USA.

Bühlmann, P. and Yu, B. (2002), "Analyzing bagging", *Ann. Stat.*, **30**(4), 927-961. https://doi.org/10.1214/aos/1031689014.

Chakraborty, D., Awolusi, I. and Gutierrez, L. (2021), "An explainable machine learning model to predict and elucidate the compressive behavior of high-performance concrete", *Result. Eng.*, **11**, 100246. https://doi.org/10.1016/j.rineng.2021.100245.

Chou, J.S., Chiu, C.K., Farfoura, F. and Al-Taharwa, I. (2011), "Optimizing the prediction accuracy of concrete compressive strength based on a comparison of data-mining techniques", *J. Comput. Civil Eng.*, **25**(3), 242-253. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000088.

Combrisson, E. and Jerbi, K. (2015), "Exceeding chance level by chance: The caveat of theoretical chance levels in brain signal classification and statistical assessment of decoding accuracy", *J. Neurosci. Method.*, **250**, 126-136. https://doi.org/10.1016/j.jneumeth.2015.01.010.

Cortes, C. and Vapnik, V. (1995), "Support-vector networks", *Mach. Learn.*, **20**, 273-297. https://doi.org/10.1007/BF00994018.

Duan, J., Asteris, P.G., Nguyen, H. Bui, X.N. and Moayedi, H. (2020), "A novel artificial intelligence technique to predict compressive strength of recycled aggregate concrete using ICA-XGBoost model", *Eng. Comput.*, **37**, 3329-3346. https://doi.org/10.1007/s00366-020-01003-0.

Duan, Z.H., Kou, S.C. and Poon, C.S. (2013), "Prediction of compressive strength of recycled aggregate concrete using artificial neural networks", *Constr. Build. Mater.*, **40**, 1200-1206. https://doi.org/10.1016/j.conbuildmat.2012.04.063.

Erdal, H.I., Karakurt, O. and Namli, E. (2013), "High performance concrete compressive strength forecasting using ensemble models based on discrete wavelet transform", *Eng. Appl. Artif. Intell.*, **26**, 1246-1254. https://doi.org/10.1016/j.engappai.2012.10.014.

Farooq, F., Ahmed, W., Akbar, A., Aslam, F. and Alyousef, R. (2021), "Predictive modeling for sustainable high-performance concrete from industrial wastes: A comparison and optimization of models using ensemble learners", *J. Clean. Prod.*, **292**, 126032. https://doi.org/10.1016/j.jclepro.2021.126032.

Feng, D.C., Liu, Z.T., Wang, X.D., Chen, Y., Chang., J.Q., Wei., D.F. and Jiang, Z.M. (2020), "Machine learning-based compressive strength prediction for concrete: An adaptive boosting approach", *Constr. Build. Mater.*, **230**, 117000. https://doi.org/10.1016/j.conbuildmat.2019.117000.

Fisher, A., Rudin, C. and Dominici, F. (2019), "All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously", *J. Mach. Learn. Res.*, **20**(177), 1-81.

Freund, Y. and Schapire, R.E. (1996), "Experiments with a new boosting algorithm", *Machine Learning: Proceedings of the Thirteenth International Conference*, Bari, Italy, July.

Grolemund, G. (2014), *Hands-On Programming with R*, O'Reilly Media, Inc., Sebastopol, CA, USA.

Gupta, S. (2008), "Support vector machines based modeling of concrete strength", *World Acad. Sci. Eng. Technol.*, **36**, 305-311.

Han, Q., Gui, C., Xu, J. and Lacidogna, G. (2019), "A generalized method to predict the compressive strength of high-performance concrete by improved random forest algorithm", *Constr. Build. Mater.*, **226**, 734-742. https://doi.org/10.1016/j.conbuildmat.2019.07.315.

Harandizadeh, H., Armaghani, D.J., Asteris, P.G. and Gandomi, A.H. (2021), "TBM performance prediction developing a hybrid ANFIS-PNN predictive model optimized by ICA", *Neural Comput. Appl.*, **33**, 16149-16179. https://doi.org/10.1007/s00521-021-06217-x.

Kardani, N., Bardhan, A., Samui, P., Nazem, M., Asteris, P.G. and Zhou, A. (2022), "Predicting the thermal conductivity of soils using integrated approach of ANN and PSO with adaptive and time-varying acceleration coefficients", *Int. J. Therm. Sci.*, **173**, 107427. https://doi.org/10.1016/j.ijthermalsci.2021.107427.

Kosmatka, S.H. and Wilson, M.L. (2011), *Design and Control of Concrete Mixtures*, Portland Cement Association, Skokie, IL, USA.

Lemonis, M.E., Daramara, A.G., Georgiadou, A.G., Siorikis, V.G., Tsavdaridis, K.D. and Asteris, P.G. (2022), "Ultimate axial load of rectangular concrete-filled steel tubes using multiple ANN activation functions", *Steel Compos. Struct.*, **2**(4), 459-475. https://doi.org/10.12989/scs.2022.42.4.459.

Ling, H., Qian, C., Kang, W., Liang, C. and Chen, H. (2019), "Combination of support vector machine and K-Fold cross-validation to predict compressive strength of concrete in marine environment", *Constr. Build. Mater.*, **206**, 355-363. https://doi.org/10.1016/j.conbuildmat.2019.02.071.

Lu, S., Koopialipoor, M., Asteris, P.G., Bahri, M. and Armaghani, D.J. (2020), "A novel feature selection approach based on tree models for evaluating the punching shear capacity of steel fiber-reinforced concrete flat slabs", *Mater.*, **13**, 3902. https://doi.org/10.3390/ma13173902.

Miladirad, K., Golafshani, E.M., Safehian, M. and Sarkar, A. (2021), "Modeling the mechanical properties of rubberized concrete using machine learning methods", *Comput. Concrete*, **28**(6), 567-583. https://doi.org/10.12989/cac.2021.28.6.567.

Nguyen, N.H., Vo, T.P., Lee, S. and Asteris, P.G. (2021), "Heuristic algorithm-based semi-empirical formulas for estimating the compressive strength of the normal and high-performance concrete", *Constr. Build. Mater.*, **304**, 124467. https://doi.org/10.1016/j.conbuildmat.2021.124467.

Nguyen-Sy, T., Wakim, J., To, Q.D., Vu, M.N., Nguyen, T.D. and Nguyen, T.T. (2020), "Predicting the compressive strength of concrete from its compositions and age using the extreme gradient boosting method", *Constr. Build. Mater.*, **260**, 119757. https://doi.org/10.1016/j.conbuildmat.2020.119757.

Ni, H.G. and Wang, J.Z. (2000), "Prediction of compressive strength of concrete by neural networks", *Concrete Cement Res.*, **30**, 1245-1250. https://doi.org/10.1016/S0008-8846(00)00345-8.

Oner, E. (2021), "Frictionless contact mechanics of an orthotropic coating/isotropic substrate system", *Comput. Concrete*, **28**(2), 209-220. https://doi.org/10.12989/cac.2021.28.2.209.

Phillips, R.V., van der Laan, M.J., Lee, H. and Gruber, S. (2022), "Practical considerations for specifying a super learner", *arXiv preprint arXiv*, **2204**, 06139. https://doi.org/10.48550/arXiv.2204.06139.

Popovics, S. and Ujhelyi, J. (2008), "Contribution to the concrete strength versus water cement ratio relationship", *J. Mater. Civil Eng.*, **20**(7), 459-463. https://doi.org/10.1061/(ASCE)0899-1561(2008)20:7(459).

Psyllaki, P., Stamatiou, K., Iliadis, I., Mourlas, A., Asteris, P. and Vaxevanidis, N. (2018), "Surface treatment of tool steels against galling failure", *MATEC Web Conf.*, **188**, 04024. https://doi.org/10.1051/matecconf/201818804024.

Salehi, H. and Burgueño, R. (2018), "Emerging artificial intelligence methods in structural engineering", *Eng. Struct.*, **171**, 170-189. https://doi.org/10.1016/j.engstruct.2018.05.084.

Tahwia, A.M., Heniegal, A., Elgamal, M.S. and Tayeh, B.A. (2021), "The prediction of compressive strength and non-destructive tests of sustainable concrete by using artificial neural networks", *Comput. Concrete*, **27**(1), 21-28. https://doi.org/10.12989/cac.2021.27.1.021.

Tanyildizi, H. (2022), "Predicting bond strength of corroded reinforcement by deep learning", *Comput. Concrete*, **29**(3), 145-159. https://doi.org/10.12989/cac.2022.29.3.145.

van der Laan, M.J., Polley, E.C. and Hubbard, A.E. (2007), "Super Learner", *Stat. Appl. Genet. Molecul. Biol.*, **6**(1), 1. https://doi.org/10.2202/1544-6115.1309.

Yeh, I.C. (1998), "Modeling of strength of high-performance concrete using artificial neural networks", *Cement Concrete Res.*, **28**(12), 1797-1808. https://doi.org/10.1016/S0008-8846(98)00165-3.

Young, B.A., Hall, A., Pilon, L., Gupta, P. and Sant, G. (2019), "Can the compressive strength of concrete be estimated from knowledge of the mixture proportions? New insights from statistical analysis and machine learning methods", *Cement Concrete Res.*, **115**, 379-388. https://doi.org/10.1016/j.cemconres.2018.09.006.

Zhang, J., Ma, G., Huang, Y., Sun, J., Aslani, F. and Nener, B. (2019), "Modelling uniaxial compressive strength of lightweight self-compacting concrete using random forest regression", *Constr. Build. Mater.*, **210**, 713-719. https://doi.org/10.1016/j.conbuildmat.2019.03.189.

Zhang, X., Akber, M.Z. and Zhang, W. (2021), "Prediction of seven-day compressive strength of field concrete", *Constr. Build. Mater.*, **305**, 124604. https://doi.org/10.1016/j.conbuildmat.2021.124604.

HK