

Predicting Compressive Strength and Density of Ecuadorian Natural Mordenite-Based Geopolymer Using Machine Learning Algorithms

Carolina Massay

Facultad de Ingeniería Mecánica y Ciencias de la Producción, Escuela Superior Politécnica de Litoral, ESPOL

Mauricio Cornejo

Facultad de Ingeniería Mecánica y Ciencias de la Producción, Escuela Superior Politécnica de Litoral, ESPOL

Haci Baykara

Facultad de Ingeniería Mecánica y Ciencias de la Producción, Escuela Superior Politécnica de Litoral, ESPOL

<https://doi.org/10.5109/7326955>

出版情報 : Evergreen. 11 (4), pp.3183-3203, 2024-12. Interdisciplinary Graduate School of Engineering Sciences, Kyushu University, Japan

バージョン :

権利関係 : Creative Commons Attribution 4.0 International



Predicting Compressive Strength and Density of Ecuadorian Natural Mordenite-based Geopolymer Using Machine Learning Algorithms

Carolina Massay^{1,2}, Mauricio Cornejo^{1,2}, Hacı Baykara^{1,*}

¹Facultad de Ingeniería Mecánica y Ciencias de la Producción, Escuela Superior Politécnica de Litoral, ESPOL, Campus Gustavo Galindo km 30.5 Vía Perimetral, Guayaquil, Ecuador

²Center of Nanotechnology Research and Development (CIDNA), Escuela Superior Politécnica de Litoral, ESPOL, Campus Gustavo Galindo km 30.5 Vía Perimetral, Guayaquil, Ecuador

*Author to whom correspondence should be addressed:

E-mail: hbaykara@espol.edu.ec

(Received April 17, 2024; Revised October 10, 2024; Accepted October 24, 2024).

Abstract: Nowadays, in the construction industry, it is crucial to utilize machine learning (ML) to predict the properties of construction materials, minimize potential errors, and improve the desired features. This study focused on estimating the compressive strength (CS) and density of mordenite-based geopolymers using ML models. Optimization techniques are used to improve the performance of the models. Extra Tree using Bayesian optimization is the best model for CS. The mentioned ML model achieved a determination coefficient of 0.8852 with prediction errors MAE=0.8649, MSE=1.3932, RMSE=1.1803, RMSLE=0.3884, and MAPE=6.1239. Similarly, Extra Tree using Bayesian tuning is the best model for density, yielding a determination coefficient of 0.6791 with prediction errors MAE=0.0404, MSE=0.0024, RMSE=0.0494, RMSLE=0.0178, and MAPE=0.0228.

Keywords: Compressive strength; density; machine learning; mechanical properties prediction; mordenite based-geopolymers

1. Literature review and problem statement

In traditional research, several samples are made with different proportions that must be tested experimentally to determine the relationship between the independent and dependent variables. However, developing prototypes and carrying out tests involves investment in the acquisition of raw materials, equipment, tools, software, and time used to carry out the research. This reality applies in the case of geopolymer preparation and characterization, which requires various mechanical, physical, or chemical tests to determine the optimum mix design. The processes mentioned require considerable time and significant cost¹.

Geopolymers are inorganic synthetic polymers created from a chemical reaction between an alkali solution and a solid aluminosilicate². They have been investigated as an alternative to Ordinary Portland Cement (OPC) due to their resistance to compression, temperature, and corrosion, as well as being eco-friendly³. Because traditional production of raw materials used in the construction industry, like OPC, generates carbon dioxide emissions and energy consumption⁴. Another environmentally harmful raw material utilized in construction is the sand river, as its excessive collection to produce concrete causes soil erosion. Singh *et al.*⁵ studied

copper slag, quarry dust, foundry sand and saw dust as fine aggregate to replace sand river. Hence, sustainable and environmentally friendly construction and building materials production are among the main goals of the construction industry. So, the research on a potential replacement for OPC-based materials to reduce CO₂ emissions is crucial⁶.

Geopolymer cement and geopolymer-based materials used in the construction sector show superior safety and mechanical and chemical resistance properties. Therefore, tests must be conducted to measure geopolymer compressive strength (CS) and guarantee its structural safety⁷. In this sense, machine learning (ML) algorithms provide reliable and accurate predictions, reducing the time and cost of experiments testing⁸⁻¹⁴. In data science, a dataset is utilized to train and test ML models through a supervised model that involves pre-establishing input and target variables, enabling, in this manner, the identification of hidden patterns within the analyzed process¹⁵. The model dataset is usually divided into training and testing data. In the training step, the model is built, and it learns about the underlying rules from the training data analyzed. This trained model is used in the testing step to obtain predictions from unseen data.

Classification models are used when the target is discrete, but regression models are used when the target is continuous¹⁶⁾. As CS and density are continuous output variables, regression models are used in this study.

Also, the validation and evaluation steps are necessary to determine the ability of the trained model to predict the target variable, giving new, unknown data. This is important because it helps measure the reliability of the model. The performance of a regression ML model can be assessed using statistical metrics that indicate the error between the experimental and predicted values. A method to verify the performance of the ML models is the cross-validation method using a k-fold technique to determine the variability of the predicted results¹⁷⁾. A critical factor for model performance is the hyperparameter settings. They do not learn from the dataset and must be adjusted before running the model¹⁸⁾. One way to achieve better performance for the ML model is to tune the default set of hyperparameters using optimization techniques^{19–21)}. All ML models had default values for the hyperparameter set, but if the optimal values are in an established space range for each hyperparameter, the statistical metrics of the ML models and, therefore, their performance can be improved.

Previous studies^{7,22–24)} developed ML models to predict the properties of geopolymers and demonstrate their feasibility. Su *et al.*⁷⁾ implemented a multivariate polynomial regression model using 120 data for the geopolymers based on blast furnace slag and metakaolin, obtaining values of the coefficient of determination of 0.946 and 0.927 for the compressive strength at 7 and 28 days, correspondingly. Ahmad *et al.*²²⁾ performed a neural adaptive fuzzy inference system model using 90 data for the geopolymer based on fly ash, obtaining a determination coefficient of 0.9 for the compressive strength. Huo *et al.*²³⁾ investigated several machine learning models, such as bootstrap and boosting, using 557 data for the calcium-based geopolymer. Extreme Gradient Boosting (EGB), Gradient Boosting (GB), Decision Trees Regression (DT), and Extra Tree (ET) obtained 0.91, 0.91, and 0.90, in that order, for the coefficient of determination in the testing step to predict CS. Kuang *et al.*²⁴⁾ used backpropagation neural networks (BPNN), random forest (RF), and k-nearest neighbors (KNN) to predict CS with 191 data for the geopolymer based on metakaolin, slag, and fly ash, obtaining determination coefficients of 0.9394, 0.9274, and 0.9059, respectively. Kumar *et al.*²⁵⁾ studied the synthesis of concrete geopolymers that used fly ash, silica fume, ground granulated blast slag as raw material, and polyvinyl alcohol and steel fibers as reinforced materials. In the study, the implementation of SVM and ANN models was performed to predict the compressive strength of these reinforced composites.

Besides, subsequent studies predict properties such as compressive strength and slump for fly ash concrete and mortar geopolymers^{8,26–36)}. These studies use many types of artificial intelligence models, such as BPNN, Support

Vector Machine (SVM), extreme learning machine, artificial neural network (ANN), genetic programming, DT, bagging regressor, AdaBoost regressor, deep neural network, deep residual network, RF, Ridge regression model, light gradient boosting machine (LGBM), EGB, and GB. Additionally, Alakara *et al.*³⁷⁾ investigated the prediction of compressive strength using TreeNet and ANN for Ground Blast Furnace Slag-raw perlite powder-geopolymer mortar. Similarly, Paswan *et al.*³⁸⁾ developed two models (KNN and EGB) to predict the tensile strength of ground-granulated blast furnace slag-based geopolymer concrete, and it was determined that the EGB model has a better performance with lower statistical error metrics than KNN.

Several studies have used natural zeolite as a solid precursor for synthesizing geopolymers^{39,40)}. Other researchers have also synthesized and predicted the mechanical properties of natural zeolite-incorporated geopolymer concrete^{1,41)}, and OPC based-concrete⁴²⁾. Besides, prior studies have analyzed ash from the combustions of fossil fuels in thermoelectric⁴³⁾, coal bottom ash⁴⁴⁾, and groundnut shell⁴⁵⁾ as a solid precursor to replace OPC for concrete. Analogously, Sharma *et al.*⁴⁶⁾ implemented an ANN model to predict CS and flexural strength for concrete that used eggshell powder as a raw material. The model reached values of 0.9915 and 0.9921 of the coefficients of determination for CS and flexural strength, confirming that the model performs well and has a high accuracy.

In addition, hyperparameter optimization can enhance the performance of the ML models. For this purpose, Zhang *et al.*⁴⁷⁾ studied the development of machine learning to predict the CS of recycled aggregate concrete. The current study analyzed the three optimization methods: Grid search, Random search, and Bayesian. The findings indicated Bayesian optimization had a lower processing time to search the optimal hyperparameter set than the other two optimization methods. Daniel *et al.*⁴⁸⁾ implemented six ML models: AdaBoost, GB, RF, ET, EGB, and Light Gradient Boosting Machine. These six models were enhanced using a hyperparameter optimization with the Bayesian method. As a result, the ET model optimized by the Bayesian method achieved the highest performance with a coefficient of determination of 0.9898 and 0.9876 for training and testing steps.

Moreover, several studies have established the contribution and relevance of each input variable by performing a feature importance analysis in the implemented ML models. Chen *et al.*⁴⁹⁾ developed a Shapley additive explanation-based method (SHAP) to determine the importance of the input variables regarding the target. This study reported that the amount of cement and water had a high effect on the compressive strength of high-strength concrete. Similarly, Wang *et al.*⁵⁰⁾ explored the contribution of the input parameters by performing a SHAP analysis in an EGB model for the CS prediction. It was determined that the age, the percentage of air content,

the slump, and the water-binder ratio are the most relevant input variables for the CS of concrete.

However, no scientific articles have predicted the mechanical properties of Ecuadorian zeolite-based geopolymers. Therefore, this study focuses on implementing machine learning algorithms on the Google Colab online platform using the Pycaret ML library. This study aims to implement ML algorithms to successfully predict geopolymers' compressive strength and density using Ecuadorian mordenite as precursors. For this reason, the five models with the best performance were selected according to the values of the statistical metrics when predicting compressive strength and density. In addition, to improve the performance of the initial ML models, hyperparameter optimization was conducted using Bayesian⁵¹⁾ and Random Search⁵²⁾ methods that have been used in the literature, as previously mentioned. Besides, a feature importance analysis was performed to evaluate the influence of the input variables in the machine learning models. This way, artificial intelligence tools can predict geopolymers' mechanical and physicochemical properties, reducing the research time and the cost associated with experimental design and mechanical testing.

2. Methodology

2.1 Model Structure

Data normalization and transformation, training, testing, validation, and hyperparameter optimization in ML algorithms are realized in Python using the low-code ML library, Pycaret⁵³⁾, on the Google Collaboratory environment⁵⁴⁾. Pearson's correlation heatmap is produced using the Seaborn library⁵⁵⁾, and multivariable analysis is made using the matplotlib library⁵⁶⁾, both on the Google Collaboratory environment⁵⁴⁾. The steps to develop the ML model are detailed in the next section.

2.1.1 Dataset compilation.

Sixty-six data sets were used to build the ML models. The data sets were recollected from a previous research article³⁹⁾. This research studied the influence of the curing temperature and three components of geopolymer regarding CS for developing the best mix design with the highest value of CS. The raw materials used to synthesize are an aluminosilicate source (mordenite), an activation solution (NaOH and sodium silicate), and calcium hydroxide.

First, the sodium hydroxide concentration was varied (8M, 9M, 10M, 11M, 12M, and 14M), and then the ratio between the sodium silicate and sodium hydroxide solutions was changed (0.5, 1, 1.5, 2, 2.5 and 3). After that, the mass of calcium hydroxide, based on the zeolite mass, was modified (2g, 3g, 4g, 5g, 6g, 7 g, and 8g). Finally, the curing temperature was switched (23°C, 40°C, 50°C, 60°C, 70°C, and 80°C). Therefore, the input parameters are curing temperature (°C), the mass of calcium hydroxide (g), sodium hydroxide concentration (M), the volume of sodium hydroxide (mL), the volume of sodium silicate

(mL), and the sodium silicate/sodium hydroxide ratio. The output parameters are CS (MPa) and density (g/cm³).

The quantity of each component in geopolymer production, such as sodium hydroxide and silicate, should be considered, and the specific ratios between the two with specific amounts have been included among the input values. Moreover, variables such as the ratio between silicon oxide and sodium oxide in sodium silicate, the solid-to-liquid ratio, or the amount of zeolite were not considered input variables because they remained constant in the experimental design, and any change was applied to them.

In the current study, the triplicated results were used as input for a feasible ML analysis instead of the mean CS values of twenty-one mix series of geopolymers. To build the model, the Pycaret library⁵³⁾ separated the datasets: seventy percent (46 data sets) represents training data, and thirty percent remaining (20 data sets) is testing data. These percentages of dataset distribution were applied in other previous studies^{42,57)}.

2.1.2 Data Pre-Processing.

Data were analyzed using basic statistics. Also, a multivariate analysis was performed between input and output variables. In addition, a Pearson correlation coefficient matrix was conducted to determine the correlation between the input and output. The dataset was normalized so that one or more variables with large values do not dominate the ML model. Furthermore, the dataset was transformed into a normal distribution using a data transformation technique called Yeo-Johnson. It is an extension of the Box-Cox transformation that transforms only positive variables, but the Yeo-Johnson transformation transforms both positive and negative variables. This transformation technique is applied in ML algorithms to improve the estimated results' accuracy⁵⁸⁾.

2.1.3 Model selection.

Using the *compare_models* function⁵⁹⁾ from the Pycaret library to select the ten best ML models that predict CS and density. Five ML models are used for CS, and the other five are chosen for density. The ML algorithms used are explained in section 2.2.

2.1.4 Hyperparameter optimization.

The models selected in section 2.1.3 were optimized by the Bayesian method⁵¹⁾ using the Scikit-optimize⁶⁰⁾ library and the Random Search method⁵²⁾ with the Optuna library⁶¹⁾. Bayesian is an iterative technique that uses global and local search to optimize the set of hyperparameters in the ML model. This optimization approach consists of creating a probabilistic model (Gaussian Process) to estimate the objective function and upload it after each iteration to select the next set of hyperparameters to evaluate from the search space. This process uses all available solutions from past evaluations to search new regions in the search space, improving the

evaluation metrics of algorithms⁵¹⁾. Random search uses a set of hyperparameters randomly selected from a defined range. The results predicted by the model are evaluated on the selected set of hyperparameters, and if this result performs better than the previous one, these hyperparameters are chosen. This process is repeated until the set of hyperparameters provides the best performance for the model⁵²⁾.

In addition, regression ML models apply early stopping as a regularization technique to avoid overfitting, capture the underlying patterns that characterize the new unseen dataset, and increase their reliability. It is used when the dataset is noisy or limited. This method consists of stopping training before convergence because the model's performance does not improve or decrease, and it saves the model with the best evaluation metrics on the validation set⁶²⁾.

2.1.5 Model Validation.

A cross-validation method is used to evaluate the reliability of the ML algorithm. The k-fold technique involves splitting the dataset into k-groups, where the subsets are approximately equal. K-1 groups are trained, and the remaining groups are used for testing and repeated k-times. Every k-group is once used for testing. The average of all k sets of results must be calculated to obtain the final performance of the model. This approach reduces the problem of overfitting when the built model does not perform well in making predictions on new, unseen data during the test step⁶³⁾. The selected k-value affects the performance of the ML algorithm, as using different k-values results in different performances⁶⁴⁾. This validation method can be used in training, testing, and hyperparameter optimization. This study used 5-fold cross-validation for CS prediction and 3-fold for density prediction. Because only 46 datasets are used in the training stage, the number of groups into which the data can be divided is limited.

2.1.6 Model Evaluation.

The initial and optimized ML models are evaluated and compared using six statistical parameters. These statistical metrics are coefficient of determination (R^2), Mean Absolute Error (MAE), Mean Square Error (MSE), Root Mean Square Error (RMSE), Root Mean Square Logarithmic Error (RMSLE), and Mean Absolute Percentage Error (MAPE). Five statistical parameters, i.e., MAPE, MSE, MAE, RMSE, and RMSLE, were used to analyze the residual errors of the select models. The remaining parameter is the coefficient of determination. R^2 is the proportion of variation in the dependent variable explained by the independent variable and is used to assess the reliability of the regression model. Values closer to 1 indicate a better fit, while values closer to zero or negative indicate that the model does not fit⁶⁵⁾. MAE measures the absolute mean between the experimental and predicted values and gives an idea of the error magnitude

in the predictions from a regression model. MAE does not consider the error direction (positive or negative values) because it uses absolute values⁶⁶⁾. MSE measures the mean square between the experimental and predicted values⁶⁷⁾. RMSE measures the square root of the mean square between the experimental and predicted values. RMSE explains the average distance between the experimental and predicted values⁶³⁾. RMSLE measures the square root of the mean square on a logarithmic scale between the experimental and predicted values. RMSLE is used when the output variable has an extensive range of values⁶⁸⁾. MAPE measures the mean percentage between the experimental and predicted values. MAPE is used when the output variable is a ratio or a percentage⁶⁹⁾ values closer to zero for MAE, MSE, RMSE, RMSLE, and MAPE indicate better reliability of the regression model. The equations used to calculate the statistical parameters are defined in equations 1-6.

$$R^2 = 1 - \frac{SSR}{SST} \quad (1)$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (2)$$

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (3)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (4)$$

$$RMSLE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\log(y_i + 1) - \log(\hat{y}_i + 1))^2} \quad (5)$$

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (6)$$

Where SSR is the sum squared regression, SST is the total sum of squares, n is the total number of datasets, y_i $\hat{y}_i$ are the real and the predicted values, respectively.

2.1.7 Feature importance

A feature importance analysis was conducted to determine and quantify the relevance of each of the input variables. Feature importance plots were performed for each ML model selected in section 2.1.3 using the function “plot_model” from the ML library, Pycaret⁵³⁾. The input variables can take values from 0 to 100 according to their importance in the ML model.

2.2 Machine learning algorithms.

2.2.1 Extreme Gradient Boosting.

This algorithm was proposed by Chen and Guestrin in 2016⁷⁰). It is an ensemble method and scalable system for the GB machine. This model uses a sparsity-aware split finding for sparse data. If a value is missing, it is classified in a default direction. Optimal default directions are selected from the data. Also, it implements a weighted quantile sketch algorithm to manage the weighted data. Besides, it proposes a cache-aware algorithm with a non-continuous memory access. This algorithm additionally needs to indirectly fetch the gradients statistic by row index, decreasing the runtime overhead if the number of rows is extensive. For these innovations, the model is scalable in all scenarios and uses minimum resources⁷⁰).

2.2.2 Decision Tree Regressor.

The DT regressor was proposed by Breiman in 1984⁷¹). This algorithm uses nodes and branches. The nodes use the input variable to make decisions, and the branches predict the possible outcomes of the node decisions. However, the leaves are the prediction result value for the output variable. Selecting a feature that becomes in the first node is essential to build this model. Nevertheless, a single feature usually cannot perfectly predict the output variable, which is called impurity. For a continuous variable, the Gini method selects the feature that fits well on the data. This process is repeated for the remaining features until all the data is used⁷²).

2.2.3 Gradient Boosting.

The GB regressor was proposed by Friedman⁷³), as an ensemble learning method that uses boosted trees. In this method, training and combining multiple weak learners into one strong learner is more straightforward than developing a complex ML model. Decision trees are sequentially added to the learner, correcting the residual errors in the previous decision tree. A key characteristic of this model is that it uses a gradient descent algorithm that minimizes the loss function and improves its performance in each new iteration⁷⁴).

2.2.4 AdaBoost.

The AdaBoost or Adaptive Boosting was proposed by Freund and Schapire⁷⁵). Like GB, AdaBoost is also an ensemble learning tool that uses boosted trees. This model uses trees called Decision Stumps, with one node and two leaves. In each iteration, the weak learner is trained on the data by assigning weights to their weak learner predictions. In addition, it is adaptive because it assigns higher weights to difficult-to-classify subsets. In the model, the weights are assigned depending on the performance of the weak learners. In this case, the higher the performance of the weak learner, the higher its weight in the final prediction⁷⁴).

2.2.5 Random Forest Regressor.

In 2001, the RF regressor was proposed by Breiman⁷⁶). RF is an algorithm of supervised learning using multiple decision trees (random forest). It is an ensemble learning that uses a bootstrap technique that randomly splits and replaces the datasets. The model is trained with a random subset of the training data and a random subset of the features. Each one of the decision trees predicted a result. This model uses the average result from multiple decision trees to estimate a single final prediction. The mean result makes the RF model better than a single decision tree because it reduces the overfitting caused by a single decision tree and improves its performance⁷⁷).

2.2.6 Extra Tree Regressor.

The extremely randomized tree is an ensemble method. In 2005, this algorithm was proposed by Geurts⁷⁸). It is built with many decision trees like RF Regressor. ET randomly split the training data without replacement. ET does not use bootstrapping to construct the subset for the trees. For the splitting features, this model uses a split function optimization that first selects a certain number of features and then determines the best feature depending on the objective function. In regression, the objective function is to reduce the output variance. The output prediction is the mean of the result of each tree⁷⁹).

3 Results and discussion

3.1 Data distribution analysis

Table 1 shows a statistical analysis of the input and output variables. Temperature, the volume of NaOH, and sodium silicate had the highest standard deviation, meaning that the data for these variables was far from the average. This occurs because these variables in the analysis dataset are design parameters that can be varied to develop an optimal mix design. Normalization of the dataset is necessary because one or more variables with large magnitude values can dominate the ML model during the training step. The zscore method⁸⁰) is used to normalize the data. The formula for this normalization method is detailed in equation 7:

$$zscore = (x - u)/s \quad (7)$$

Where x is the original dataset, u is the average of the original dataset, and s is the standard deviation of the original dataset.

Figure 1 presents the multivariate analysis between inputs, CS, and density. When the concentration of sodium hydroxide is 10M, the volume of sodium hydroxide is 23.3 mL, the volume of sodium silicate is 11.6 mL, the sodium silicate/sodium hydroxide ratio is 0.5, the mass of calcium hydroxide is 8 grams, and the temperature is 60°C, the compression strength showed higher values. In contrast to density, its value does not vary much concerning the input variables.

3.2 Correlation coefficient analysis

The analysis of the correlation coefficient matrix is shown in Figure 2. It illustrates the relationship between the six inputs, CS, and density variables. The Pearson correlation coefficient can take values from 1 (dark blue) to -1 (light blue). This coefficient can be positive or negative. When it is positive, the correlation between two variables is direct, but when it is negative, the correlation

is inverse. When the coefficient is close to one, the correlation is stronger between the two variables, but it is the opposite when the coefficient is close to zero. The correlation coefficient values between inputs and compression strength ranged from -0.55 to 0.6, and for inputs and density ranged from -0.58 to 0.56. Hence, the correlation between the inputs and targets can be considered moderate^{23,57,81}.

Table 1. Summary of statistics analysis of all variables (inputs and targets)

Variables	Count	Mean	Standard Deviation	Minimum	Maximum
Concentration of Sodium Hydroxide	66.00	10.18	1.08	8.00	14.00
Volume of Sodium Hydroxide	66.00	17.79	6.31	8.70	23.30
Volume of Sodium Silicate	66.00	17.15	6.35	11.60	26.20
Sodium Silicate/Sodium Hydroxide Ratio	66.00	1.29	0.96	0.50	3.00
Mass of Calcium Hydroxide	66.00	3.86	2.09	2.00	8.00
Temperature	66.00	58.34	10.24	23.5	80
CS	66.00	4.51	3.09	0.02	13.52
Density	66.00	1.78	0.08	1.59	1.91

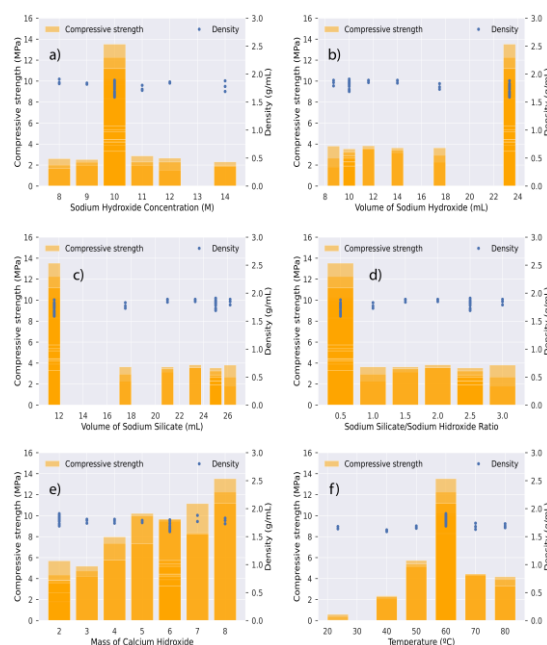


Fig. 1: Multivariate analysis between inputs, CS, and density: a) Sodium hydroxide vs. CS and density; b) Volume of sodium hydroxide vs. CS and density; c) Volume of sodium silicate vs. CS and density; d) Sodium silicate/sodium hydroxide ratio vs. CS and density; e) Mass of calcium hydroxide vs. CS and density, and f) Temperature vs. CS and density.

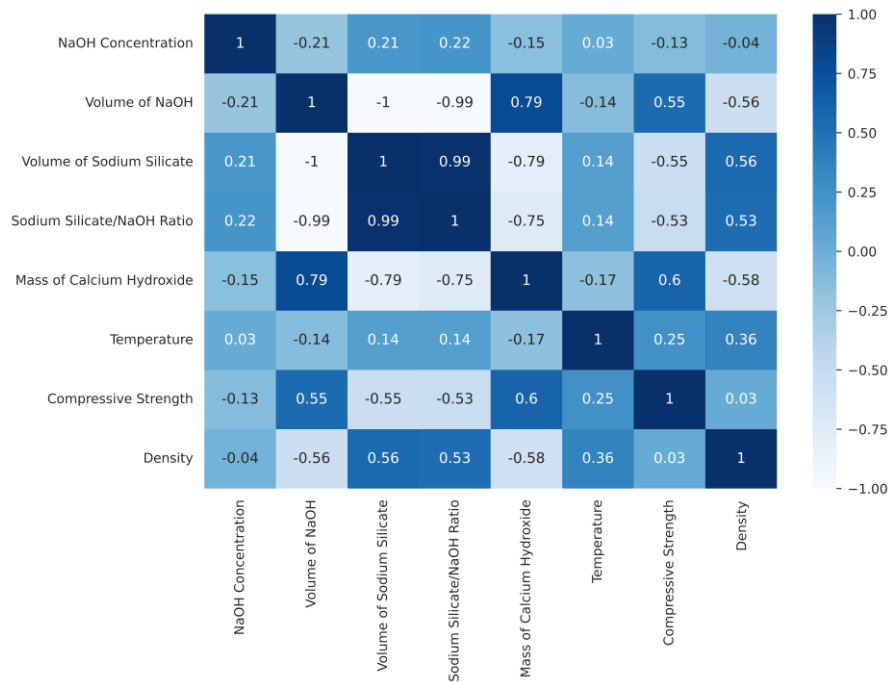


Fig. 2: Pearson's correlation heatmap of input and output parameters.

3.3 Compressive strength prediction results

Table 2 compares all the regression ML models from the Pycaret library⁵³⁾ using the six statistical scores previously mentioned in the methodology section. The first five models are selected due to their higher performance than the remaining models in Table 2. These models are EGB⁷⁰⁾, DT Regressor⁷¹⁾, ET Regressor⁷⁸⁾, GB Regressor⁷³⁾, and AdaBoost Regressor⁷⁵⁾. These initial ML models use the default values of the hyperparameters. Although RF Regressor⁷⁶⁾ also had a higher value of R^2 scores in the testing step, this model was not chosen because it had a notable difference with R^2 scores in the training step, unlike the fifth first model.

On the contrary, The Huber Regressor⁸²⁾, LGBM⁸³⁾, Dummy Regressor⁸⁴⁾, Passive Aggressive Regressor⁸⁵⁾, and Least Angle Regression⁸⁶⁾ models present negative values of the coefficient of determination for CS prediction. A negative coefficient of determination value means that the ML model's prediction results are unreliable. This negative score also indicates that the model's performance is deficient and implies that the model is overfitting^{87,88)}.

The coefficient of determination of the EGB⁷⁰⁾ model in the training and testing steps of the CS prediction results are 0.8391 and 0.8611, respectively. Instead, DT Regressor⁷¹⁾, ET Regressor⁷⁸⁾, GB Regressor⁷³⁾, and AdaBoost Regressor⁷⁵⁾ reached values of 0.8610, 0.8610, 0.8593, and 0.8439 for testing prediction scores. Tables 2, 3, and 4 show differences between training and testing MAPE values, but the other training and testing statistical parameter values are similar. In this case, MAPE is a percent error. It indicates that the CS had slight variations in the values predicted by ML models due to this study not considering a mean value of CS for each geopolymer mix.

Rather, contemplating the CS triplicate results for each mix and using different values entails a slight standard deviation for the CS results. However, triplicate results of CS are considered to provide more realistic results in the construction materials field.

In the testing step, the EGB⁷⁰⁾ model had the best R^2 value, while the AdaBoost⁷⁵⁾ Regressor model had the lowest value. The initial models with default hyperparameter values performed well in the CS prediction.

A hyperparameter optimization was performed to improve the initial ML models for CS prediction. Bayesian optimization⁵¹⁾ from the scikit-optimize⁶⁰⁾ library and Random Search optimization⁵²⁾ from the optuna⁶¹⁾ library were used. One hundred iterations were performed to obtain the optimum hyperparameters for the ten models.

Table 3 details the prediction results of the Random Search⁵²⁾ optimization for CS. The EGB⁷⁰⁾ Regressor obtained 0.8697 and 0.8541 for the coefficient of determination value in the testing and training steps, respectively. Instead, DT Regressor⁷¹⁾, ET Regressor⁷⁸⁾, GB Regressor⁷³⁾, and AdaBoost Regressor⁷⁵⁾ reached values of 0.8813, 0.8482, 0.8599, and 0.8513 for testing prediction scores.

The evaluation of the performance model using Bayesian optimization for CS prediction is indicated in Table 4. ET Regressor⁷⁸⁾ model obtained 0.8548 and 0.8852 for the coefficient of determination value in the training and testing steps, respectively. Instead, EGB Regressor⁷⁰⁾, DT Regressor⁷¹⁾, GB Regressor⁷³⁾, and AdaBoost Regressor⁷⁵⁾ reached values of 0.8660, 0.8801, 0.8517, and 0.8488 for testing prediction scores.

Therefore, the top three tuning models for CS prediction are the ET Regressor⁷⁸⁾ model with Bayesian optimization⁵¹⁾, the EGB Regressor⁷⁰⁾ model using Random Search tuning⁵²⁾, and the EGB Regressor⁷⁰⁾ model using Bayesian tuning⁵¹⁾. Although DT Regressor⁷¹⁾ had a higher coefficient of determination for the testing step than the other optimized models, there is a noteworthy difference in the R^2 scores for the training and testing steps, so for this reason, this model was not chosen as the best-optimized model for CS prediction.

The improvement in the coefficient of determination in the initial models regarding the optimized model is remarkable. In the same way, the five error metrics in the tuning models are lower than in the initial models. Except for the ET Regressor⁷⁸⁾ that uses the Random Search tuning⁵²⁾ and GB Regressor⁷³⁾ optimized by Bayesian tuning⁵¹⁾ where the testing R^2 scores are less than the initial model. In both cases, the optimization methods do not find a better set of hyperparameters to replace the set of hyperparameters by default. Table 5 indicates the default and optimized values for the hyperparameters. The k-fold validation was implemented for all five selected ML models to evaluate the variability of the performance. For the best initial and optimized model, statistical metrics, such as the coefficient of determination, RMSLE, and MAPE in the training step, had a lower standard deviation in the 5-fold validation. Whereas MAE, MSE, and RMSE had a slight standard deviation, this probably occurred due to the variation of CS prediction results related to considered triplicated data.

The distributions of CS prediction results of the five initial ML models and the three optimized models with the

highest performance are shown in Figure 3 and 4. The experimental and predicted values distribution had a similar behavior because the ML models had a similar value for the coefficient of determination in the testing step.

Wu and Zhou⁸⁹⁾ developed the SVM model with grid search optimization to predict the CS of sustainable concrete using 559 datasets. Better model performance is obtained by tuning to the SVM model. The optimized model's R^2 score, MAE and RMSE are 0.93, 3.52, and 4.89, respectively. Instead, the initial model's determination coefficient, MAE and RMSE are 0.85, 5.22, and 7.01, respectively⁸⁹⁾. Rathakrishnan *et al.*⁵⁷⁾ developed five boosting ML algorithms to predict a mechanical property (CS) of concrete using 152 datasets. This study compared all five initial ML models using default hyperparameters and optimized models with a random search method. The best model is the GB Regressor. For the optimized model, the determination coefficient, MAE, MSE, RMSE, RMSLE, and MAPE values are 0.96, 2.73, 11.53, 3.40, 0.03, and 0.03, respectively. Instead, the determination coefficient MAE, MSE, RMSE, RMSLE, and MAPE values for the initial model are 0.93, 3.24, 17.82, 4.22, 0.05, and 0.03, respectively. The slightly lower performance of the models to forecast CS regarding the statistical metrics reported by other studies can be attributed to the dataset size used in this study. This study only used sixty-six data to implement the ML models, and it is reported that a dataset's size affects the ML models' performance⁹⁰⁾.

Table 2. Parameter evaluation for CS prediction ML models without optimization.

Models		R^2	MAE	MSE	RMSE	RMSLE	MAPE
EGB	Train	0.8391	0.7726	1.2495	1.0415	0.1876	0.2020
	Test	0.8611	0.9814	1.6847	1.2979	0.3858	6.0500
DT Regressor	Train	0.8491	0.7330	1.1174	0.9930	0.1841	0.1982
	Test	0.8610	0.9818	1.6859	1.2984	0.3858	6.0489
ET Regressor	Train	0.8457	0.7573	1.1556	1.0097	0.1861	0.2021
	Test	0.8610	0.9818	1.6859	1.2984	0.3858	6.0489
GB Regressor	Train	0.8410	0.7845	1.2091	1.0295	0.1834	0.2019
	Test	0.8593	0.9918	1.7062	1.3062	0.3890	6.1580
AdaBoost Regressor	Train	0.8281	0.8638	1.3258	1.0779	0.1845	0.2170
	Test	0.8439	1.1015	1.8939	1.3762	0.3959	6.2355
Random Forest	Train	0.7876	1.5727	1.1778	0.9550	0.1970	0.2350
	Test	0.8874	0.9400	1.3662	1.1688	0.3906	6.2272
K Neighbors	Train	0.4554	1.2737	2.9810	1.6729	0.2886	0.3648
	Test	0.6805	1.5586	3.8762	1.9688	0.5342	9.6365
Bayesian Ridge	Train	0.1947	1.6525	5.3009	2.2611	0.3500	0.4166
	Test	0.2627	2.2170	8.9443	2.9907	0.6789	14.7139
Elastic Net	Train	0.1839	1.7335	5.7776	2.3267	0.3689	0.4526

Linear Regression	Test	0.1535	2.3367	10.2684	3.2044	0.7056	15.7926
	Train	0.1731	1.5552	5.1599	2.2495	0.3386	0.3787
Ridge Regression	Test	0.2822	2.2668	8.7075	2.9508	0.6791	14.6314
	Train	0.1590	1.6540	5.2781	2.2730	0.3516	0.4173
Lasso Regression	Test	0.2855	2.2202	8.6674	2.9440	0.6763	14.6221
	Train	0.1084	1.8933	6.5252	2.4537	0.4008	0.5091
Lasso Least Angle Regression	Test	0.1421	2.3641	10.4063	3.2259	0.7007	15.0433
	Train	0.1084	1.8933	6.5252	2.4537	0.4008	0.5091
Orthogonal Matching Pursuit	Test	0.1421	2.3641	10.4063	3.2259	0.7007	15.0433
	Train	0.0602	1.7703	5.8400	2.3931	0.3772	0.4577
Hubber Regressor	Test	0.1372	2.3615	10.4661	3.2351	0.7410	18.4493
	Train	-0.1025	1.6972	6.8751	2.5923	0.3662	0.4079
Light Gradient Boosting Machine	Test	0.4131	1.9831	7.1198	2.6683	0.5446	8.4539
	Train	-0.2179	2.2965	8.7249	2.8516	0.4807	0.6372
Dummy Regressor	Test	0.1103	2.4489	10.7931	3.2853	0.7360	17.7614
	Train	-0.2179	2.2965	8.7249	2.8516	0.4807	0.6372
Passive Aggressive Regressor	Test	-0.0110	2.7087	12.2639	3.5020	0.6991	12.1952
	Train	-0.9094	2.5270	11.3464	3.2409	0.5354	0.7113
Least Angle Regression	Test	0.2944	2.2043	8.5593	2.9256	0.4334	2.3592
	Train	-15.3100	3.5267	53.1718	5.0715	0.5808	1.0158
	Test	0.2823	2.2659	8.7062	2.9506	0.6790	14.6311

Table 3. Parameter evaluation for CS prediction ML models (Random search optimization with the Optuna library)

Models		R ²	MAE	MSE	RMSE	RMSLE	MAPE
EGB	Train	0.8541	0.7140	1.0929	0.9930	0.1750	0.1798
	Test	0.8697	0.9192	1.5802	1.2571	0.3871	6.0941
DT Regressor	Train	0.7727	0.8377	1.5075	1.1519	0.1808	0.1931
	Test	0.8813	0.9232	1.4400	1.2000	0.3838	6.0425
ET Regressor	Train	0.7835	0.9725	1.6400	1.2072	0.2044	0.2453
	Test	0.8482	1.0669	1.8413	1.3569	0.4584	8.1220
GB Regressor	Train	0.8482	0.7410	1.1368	1.0012	0.1832	0.1977
	Test	0.8599	1.0005	1.6991	1.3035	0.3867	6.0496
AdaBoost Regressor	Train	0.8441	0.8051	1.2379	1.0423	0.1766	0.2042
	Test	0.8513	1.0393	1.8041	1.3432	0.3931	6.2624

Table 4. Parameter evaluation for CS prediction ML models (Bayesian optimization with the Scikit-optimize library)

Models		R ²	MAE	MSE	RMSE	RMSLE	MAPE
EGB	Train	0.8592	0.6929	1.0613	0.9740	0.1596	0.1711
	Test	0.8660	0.9758	1.6257	1.2750	0.3934	6.2300
DT Regressor	Train	0.7764	0.8450	1.4908	1.1446	0.1784	0.1920
	Test	0.8801	0.9387	1.4550	1.2062	0.3846	6.0466
ET Regressor	Train	0.8548	0.7045	1.1255	0.9860	0.1746	0.1811
	Test	0.8852	0.8649	1.3932	1.1803	0.3884	6.1239
GB Regressor	Train	0.8767	0.6837	0.9146	0.8936	0.1616	0.1818
	Test	0.8517	1.0197	1.7994	1.3414	0.4048	6.4706
	Train	0.8453	0.7967	1.3087	1.0544	0.1682	0.1842

AdaBoost Regressor	Test	0.8488	1.0783	1.8345	1.3545	0.3951	6.2708
-------------------------------	------	--------	--------	--------	--------	--------	--------

Table 5. Defaults and optimized hyperparameter values for the top three models to CS prediction.

Top three CS models				
	Hyperparameter	ET using Bayesian tuning	EGB using Random Search tuning	EGB using Bayesian tuning
Default Value	max_depth	None	None	None
	max_features	1	-	-
	min_impurity_decrease	0	-	-
	Colsample_bytree	-	None	None
	Learning_rate	-	None	None
	Min_child_weight	-	None	None
	ccp_alpha	0	-	-
	min_weight_function_leaf	0	-	-
	min_samples_leaf	1	-	-
	min_samples_split	2	-	-
	n_estimators	10	None	None
Optimized Value	max_depth	11	9	6
	max_features	1	-	-
	min_impurity_decrease	6.47×10^{-9}	-	-
	Colsample_bytree	-	0.9304	0.7132
	Learning_rate	-	0.1146	0.4953
	Min_child_weight	-	2	2
	ccp_alpha	0	-	-
	min_weight_function_leaf	0	-	-
	min_samples_leaf	1	-	-
	min_samples_split	2	-	-
	n_estimators	300	216	252

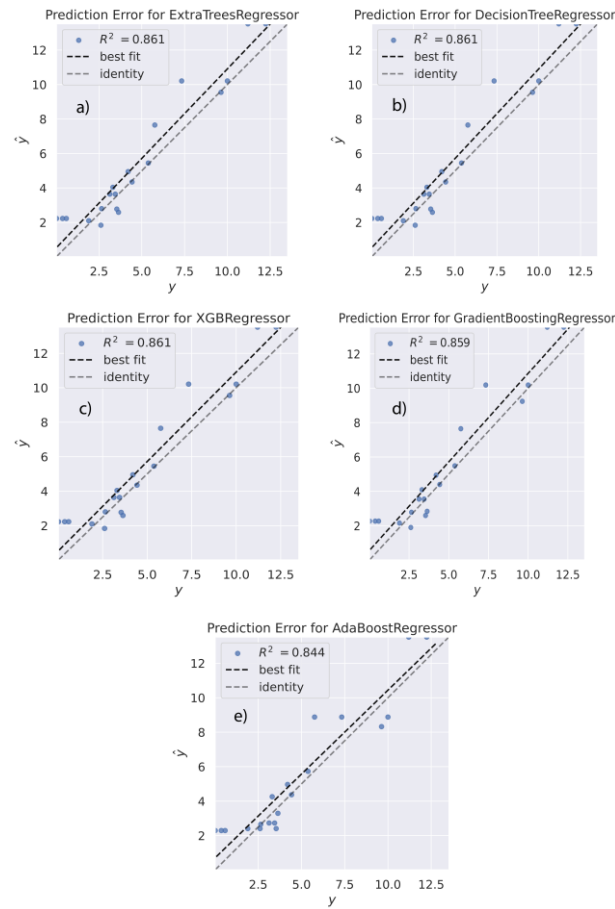


Fig. 3: The best-fit line for CS prediction distribution in the testing step (Models without optimization): a) ET initial model, b) DT initial model, c) EGB initial model, d) GB initial model, and e) AdaBoost initial model.

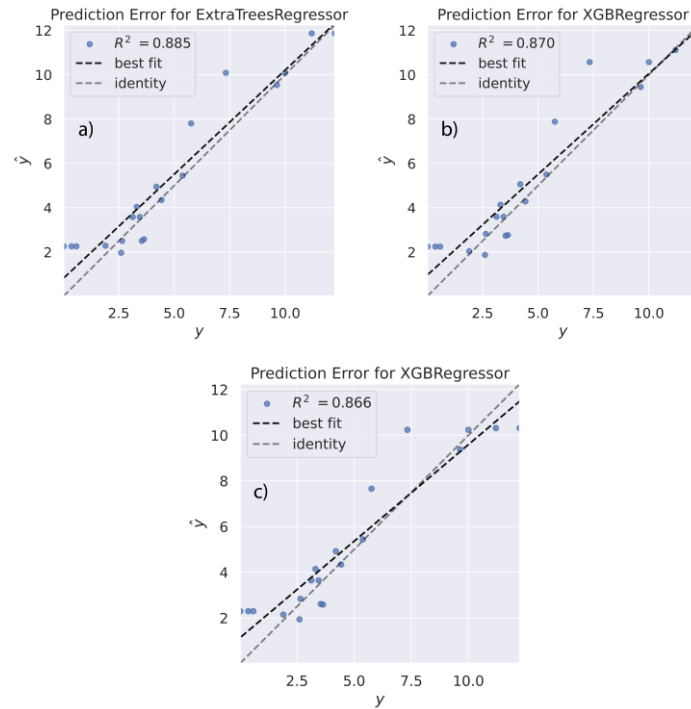


Fig. 4: The best-fit line for CS prediction distribution in the testing step for the top 3 optimized models: a) ET model using Bayesian tuning, b) EGB model using Random search tuning, and c) EGB model using Bayesian tuning.

3.5 Density prediction results

Table 6 enlists and compares all the regression models from the Pycaret library⁵³⁾ to predict density using the six scores for the training and testing step. The Passive Aggressive Regressor⁸⁵⁾, Lasso Regression⁹¹⁾, Elastic Net⁹²⁾, Lasso Least Angle Regression⁹³⁾, LGBM Regressor⁸³⁾, and Dummy Regressor⁸⁴⁾ models had negative values of the coefficient of determination for density prediction.

This negative score suggests that the ML model performs poorly and can be overfitted^{87,88)}.

The five selected machine learning models for density prediction are RF Regressor⁷⁶⁾, ET Regressor⁷⁸⁾, GB Regressor⁷³⁾, EGB⁷⁰⁾, and DT Regressor⁷¹⁾. These initial ML models are implemented with the hyperparameter set by default. The GB Regressor⁷³⁾ performed better, achieving coefficients of determination of 0.5365 and 0.5632 for training and testing steps, respectively. Instead, RF Regressor⁷⁶⁾, ET Regressor⁷⁸⁾, EGB Regressor⁷⁰⁾, and DT Regressor⁷¹⁾ models reached 0.6549, 0.5548, 0.5594, and 0.5548 for testing prediction scores. Nevertheless, RF Regressor⁷⁶⁾ had a higher coefficient of determination than the other four models, and it was not chosen because there was a difference between their training and testing values for the coefficient of determination.

Hyperparameter optimization was conducted to improve these previous initial ML models and predict density using two optimization methods. The methods implemented are Bayesian optimization⁵¹⁾ from the scikit-optimize⁶⁰⁾ library and Random Search optimization⁵²⁾ from the optuna⁶¹⁾ library. One hundred iterations were executed to get the optimum values of the hyperparameter set for each ML model.

The statistical parameter evaluation for the optimized models of density prediction is detailed in Tables 7 and 8. The training and testing scores for the optimized models using Random Search optimization⁵²⁾ are indicated in Table 7. ET Regressor⁷⁸⁾ has the better coefficient of determination in testing with 0.6550 and a training value of 0.6261. RF Regressor⁷⁶⁾, GB Regressor⁷³⁾, EGB Regressor⁷⁰⁾, and DT Regressor⁷¹⁾ reached values of 0.6552, 0.6612, 0.6726, and 0.6624 for testing. However, these four models had a difference between the training and testing scores of the coefficient of determination; for this reason, they were not selected as the best model.

Similarly, Table 8 shows the training and testing scores for Bayesian optimization⁵¹⁾. ET Regressor⁷⁸⁾ has the better coefficient of determination in training with 0.6659 and a testing value of 0.6791. RF Regressor⁷⁶⁾, GB Regressor⁷³⁾, EGB Regressor⁷⁰⁾, and DT Regressor⁷¹⁾ obtained 0.7135, 0.6281, 0.6199, and 0.6516 for the coefficient of determination in the testing step. In the RF Regressor⁷⁶⁾ the coefficient of determination in training is

notably lower than in testing, so this model was not considered.

Hence, the top three optimized models for density prediction are the ET Regressor⁷⁸⁾ model using Bayesian optimization⁵¹⁾, the ET Regressor⁷⁸⁾ model using Random Search optimization⁵²⁾, and the EGB Regressor⁷⁰⁾ using Bayesian optimization⁵¹⁾.

Furthermore, in another study,⁹⁴⁾ the CS and density of foamed concrete were predicted using a gene expression programming model. The built model uses 166 data points from the literature, 70% for training and 30% for testing. Additionally, 15 extra data points collected from laboratory tests were used to validate the model. The input variables were the cement content in kilograms, water-cement ratio, sand content in kilograms, and foam volume in cubic meters. CS and density were evaluated after the samples had been cured for twenty-eight days. The coefficient of determination for the density prediction model was 0.95 in training, 0.98 in testing, and 0.79 in validation. Also, the coefficient of determination for the compressive strength prediction model was 0.95 in training, 0.94 in testing, and 0.94 in validation⁹⁴⁾.

The improvement in the initial models is observed concerning the optimized models because the R^2 scores in the optimized algorithms are higher than in the initial algorithms, and the prediction errors in the tuning models are lower than in the initial models. The ET Regressor⁷⁸⁾ models for CS and density that implemented the Bayesian optimization⁵¹⁾ had the better performance. This is caused by the Bayesian optimization⁵¹⁾ method using a probabilistic model (Gaussian Process) to choose the next set of hyperparameters instead of the Random Search optimization⁵²⁾ selects randomly them from a search space. Besides, the Bayesian optimization⁵¹⁾ method also uses previous results to find regions of the search space that improve the statistical metrics of the model⁵¹⁾.

Table 9 illustrates the default and optimized values for the hyperparameters in the top three ML models for density prediction. A k-fold validation was performed to ensure a lower variability in the ML model. For density prediction, a value of k equal to three was selected to examine the variability of all ML models that predict density. For the best initial and optimized model, it was identified that the error metrics had a slight standard deviation, but the coefficient of determination had a remarkable standard deviation. These findings indicated that the performance of the density models had a higher variability. Moreover, the distribution of density prediction results for five initial ML models and the top three optimized models are shown in Figure 5 and 6. The distribution fit lines for density prediction illustrate that there is a relevant distance between the predicted values and the best-fit line of the models, and it probably causes the lower performance of the density prediction.

Table 6. Parameter evaluation for density prediction ML models without optimization.

Models		R ²	MAE	MSE	RMSE	RMSLE	MAPE
RF Regressor	Train	0.6038	0.0374	0.0021	0.0436	0.0157	0.0212
	Test	0.6549	0.0389	0.0026	0.0512	0.0185	0.0220
ET Regressor	Train	0.5450	0.0388	0.0024	0.0486	0.0175	0.0219
	Test	0.5548	0.0450	0.0034	0.0582	0.0210	0.0253
GB Regressor	Train	0.5365	0.0398	0.0024	0.0485	0.0175	0.0225
	Test	0.5632	0.0448	0.0033	0.0576	0.0208	0.0252
EGB	Train	0.5188	0.0408	0.0025	0.0497	0.0180	0.0231
	Test	0.5594	0.0448	0.0033	0.0579	0.0209	0.0252
DT Regressor	Train	0.5160	0.0404	0.0025	0.0500	0.0181	0.0228
	Test	0.5548	0.0450	0.0034	0.0582	0.0210	0.0253
K Neighbors	Train	0.5000	0.0408	0.0028	0.0505	0.0183	0.0232
	Test	0.6990	0.0341	0.0023	0.0478	0.0174	0.0195
AdaBoost Regressor	Train	0.4807	0.0420	0.0028	0.0505	0.0182	0.0238
	Test	0.5547	0.0471	0.0034	0.0582	0.0210	0.0266
Linear Regression	Train	0.2910	0.0481	0.0037	0.0594	0.0216	0.0274
	Test	0.4381	0.0551	0.0043	0.0654	0.0236	0.0310
Least Angle Regression	Train	0.2892	0.0482	0.0038	0.0596	0.0216	0.0274
	Test	0.4379	0.0551	0.0043	0.0654	0.0236	0.0311
Ridge Regression	Train	0.2425	0.0507	0.0040	0.0618	0.0224	0.0288
	Test	0.4303	0.0559	0.0043	0.0658	0.0238	0.0315
Bayesian Ridge	Train	0.2116	0.0518	0.0042	0.0638	0.0232	0.0295
	Test	0.4604	0.0549	0.0041	0.0640	0.0232	0.0310
Orthogonal Matching Pursuit	Train	0.2106	0.0508	0.0042	0.0643	0.0234	0.0289
	Test	0.4355	0.0598	0.0043	0.0655	0.0238	0.0340
Huber Regressor	Train	0.0782	0.0515	0.0048	0.0661	0.0239	0.0294
	Test	0.3703	0.0590	0.0048	0.0692	0.0251	0.0335
Passive Aggressive Regressor	Train	-0.0983	0.0632	0.0058	0.0760	0.0274	0.0360
	Test	0.3973	0.0583	0.0046	0.0677	0.0244	0.0329
Lasso Regression	Train	-0.1437	0.0642	0.0062	0.0786	0.0286	0.0367
	Test	-0.0231	0.0745	0.0078	0.0882	0.0320	0.0428
Elastic Net	Train	-0.1437	0.0642	0.0062	0.0786	0.0286	0.0367
	Test	-0.0231	0.0745	0.0078	0.0882	0.0320	0.0428
Lasso Least Angle Regression	Train	-0.1437	0.0642	0.0062	0.0786	0.0286	0.0367
	Test	-0.0231	0.0745	0.0078	0.0882	0.0320	0.0428
Light Gradient Boosting Machine	Train	-0.1437	0.0642	0.0062	0.0786	0.0286	0.0367
	Test	0.4543	0.0588	0.0041	0.0644	0.0234	0.0336
Dummy Regressor	Train	-0.1437	0.0642	0.0062	0.0786	0.0286	0.0367
	Test	-0.0231	0.0745	0.0078	0.088	0.0320	0.0428

Table 7. Parameter evaluation for density prediction models (Random search optimization with the Optuna library)

Models		R ²	MAE	MSE	RMSE	RMSLE	MAPE
RF Regressor	Train	0.5802	0.0390	0.0023	0.0457	0.0165	0.0221
	Test	0.6552	0.0366	0.0026	0.0512	0.0185	0.0207
ET Regressor	Train	0.6261	0.0355	0.0020	0.0429	0.0154	0.0200
	Test	0.6550	0.0403	0.0026	0.0512	0.0185	0.0228
GB Regressor	Train	0.5748	0.0392	0.0023	0.0460	0.0166	0.0222
	Test	0.6612	0.0374	0.0026	0.0508	0.0183	0.0211
EGB	Train	0.6003	0.0378	0.0021	0.0449	0.0162	0.0214
	Test	0.6726	0.0392	0.0025	0.0499	0.0180	0.0221
DT Regressor	Train	0.5935	0.0377	0.0022	0.0448	0.0161	0.0214
	Test	0.6624	0.0367	0.0026	0.0507	0.0183	0.0208

Table 8. Parameter evaluation for density prediction ML models (Bayesian optimization with the Scikit-optimize library)

Models		R ²	MAE	MSE	RMSE	RMSLE	MAPE
RF Regressor	Train	0.5848	0.0377	0.0023	0.0451	0.0163	0.0213
	Test	0.7135	0.0337	0.0022	0.0467	0.0168	0.0191
ET Regressor	Train	0.6659	0.0336	0.0018	0.0406	0.0146	0.0190
	Test	0.6791	0.0404	0.0024	0.0494	0.0178	0.0228
GB Regressor	Train	0.5984	0.0366	0.0021	0.0439	0.0159	0.0208
	Test	0.6281	0.0406	0.0028	0.0532	0.0192	0.0229
EGB	Train	0.6211	0.0364	0.0020	0.0436	0.0157	0.0206
	Test	0.6199	0.0417	0.0029	0.0538	0.0193	0.0234
DT Regressor	Train	0.5949	0.0381	0.0022	0.0448	0.0161	0.0216
	Test	0.6516	0.0370	0.0026	0.0515	0.0186	0.0210

Table 9. Defaults and optimized hyperparameter values for the top three models to density prediction.

Top three density models				
Hyperparameter		ET using Bayesian tuning	ET using Random Search tuning	GBR using Bayesian tuning
Default Value	max_depth	None	None	3
	max_features	1	1	None
	min_impurity_decrease	0	0	0
	min_samples_leaf	1	1	1
	min_samples_split	2	2	2
	n_estimators	100	100	100
	learning rate	-	-	0.1
Optimized Value	max_depth	6	6	1
	max_features	1	0.9632	1
	min_impurity_decrease	1*10 ⁻⁹	5.42*10 ⁻⁷	1.28*10 ⁻⁸
	min_samples_leaf	1	1	2
	min_samples_split	2	2	10
	n_estimators	10	282	244
	learning rate	-	-	0.1723

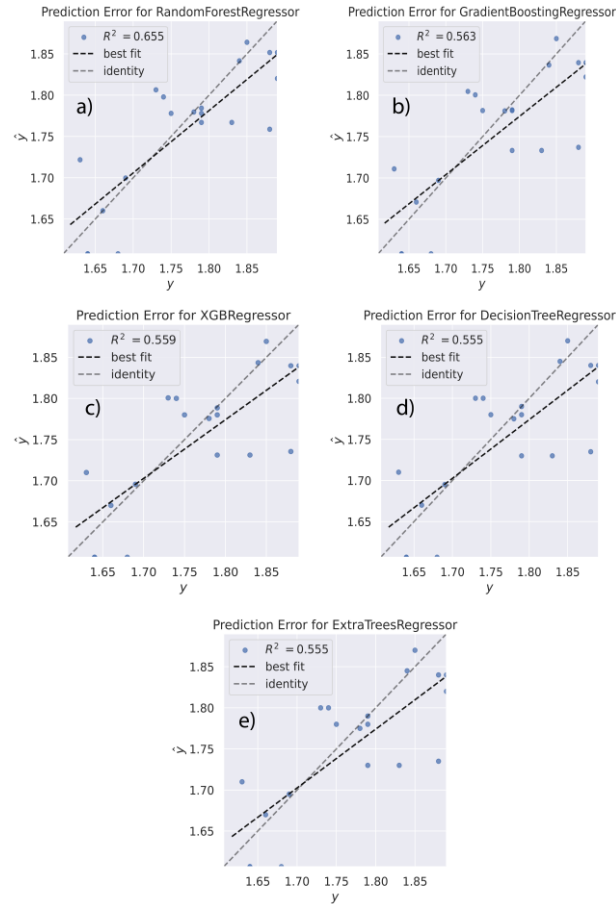


Fig. 5: The best-fit line for density prediction distribution in the testing step (Models without optimization): a) RF initial model, b) GBR initial model, c) EGB initial model, d) DT initial model, and e) ET initial model

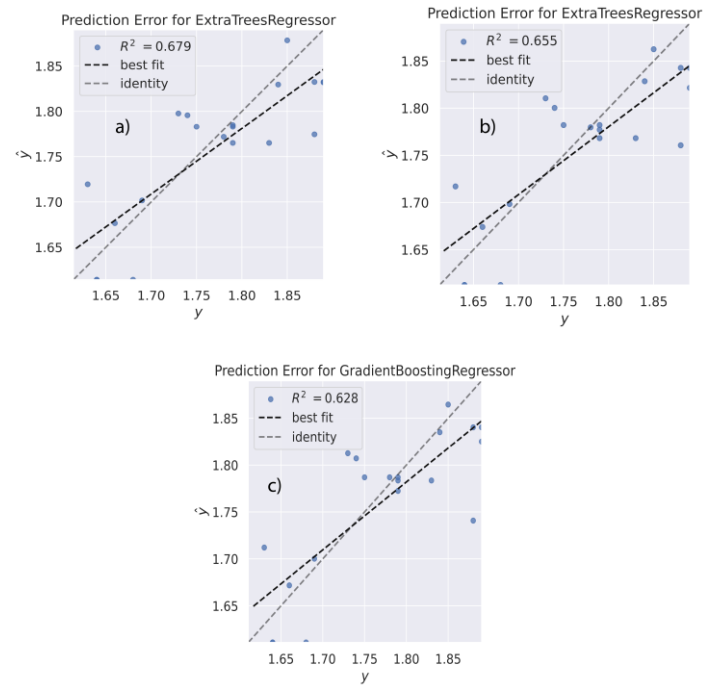


Fig. 6: The best-fit line for density prediction distribution in the testing step for the top 3 optimized models: a) ET model using Bayesian tuning, b) et model using Random Search tuning, and c) gb model using Bayesian tuning.

3.6 Feature importance analysis

Figure 7 shows the feature importance analysis for the top three tuning models of CS prediction. ET Regressor⁷⁸⁾ model that used Bayesian optimization⁵¹⁾ and EGB Regressor⁷⁰⁾ that used Random Search⁵²⁾ give great importance to the variables like the mass of calcium hydroxide and temperature. However, the temperature is more relevant for ET Regressor⁷⁸⁾, and for EGB Regressor⁷⁰⁾, the mass of calcium hydroxide. Instead, EGB Regressor⁷⁰⁾ with Bayesian optimization⁵¹⁾ emphasizes the variable Mass of Calcium Hydroxide. This model does not consider the NaOH concentration, volume of sodium silicate, and the ratio between sodium silicate and sodium hydroxide relevant. Similarly, for EGB Regressor⁷⁰⁾ which used Random Search⁵²⁾ the ratio

is not essential, and for ET Regressor⁷⁸⁾ model that used Bayesian optimization⁵¹⁾ the sodium hydroxide is not considered significant.

The feature importance analysis for the top 3 tuning models of density prediction is shown in Figure 8. All the models give great importance to the Temperature, the Mass of Calcium Hydroxide, and the NaOH concentration. The variables of Sodium Silicate/NaOH ratio, NaOH volume, and sodium silicate volume are not considered keys for regression models. Harmaji *et al.*⁹⁵⁾ also developed a feature importance analysis to predict workability and compressive strength. For workability, the percentage of calcium oxide is the most relevant input variable, and for the CS, it is the mass of fly ash (aluminosilicate source) and the amount of water.

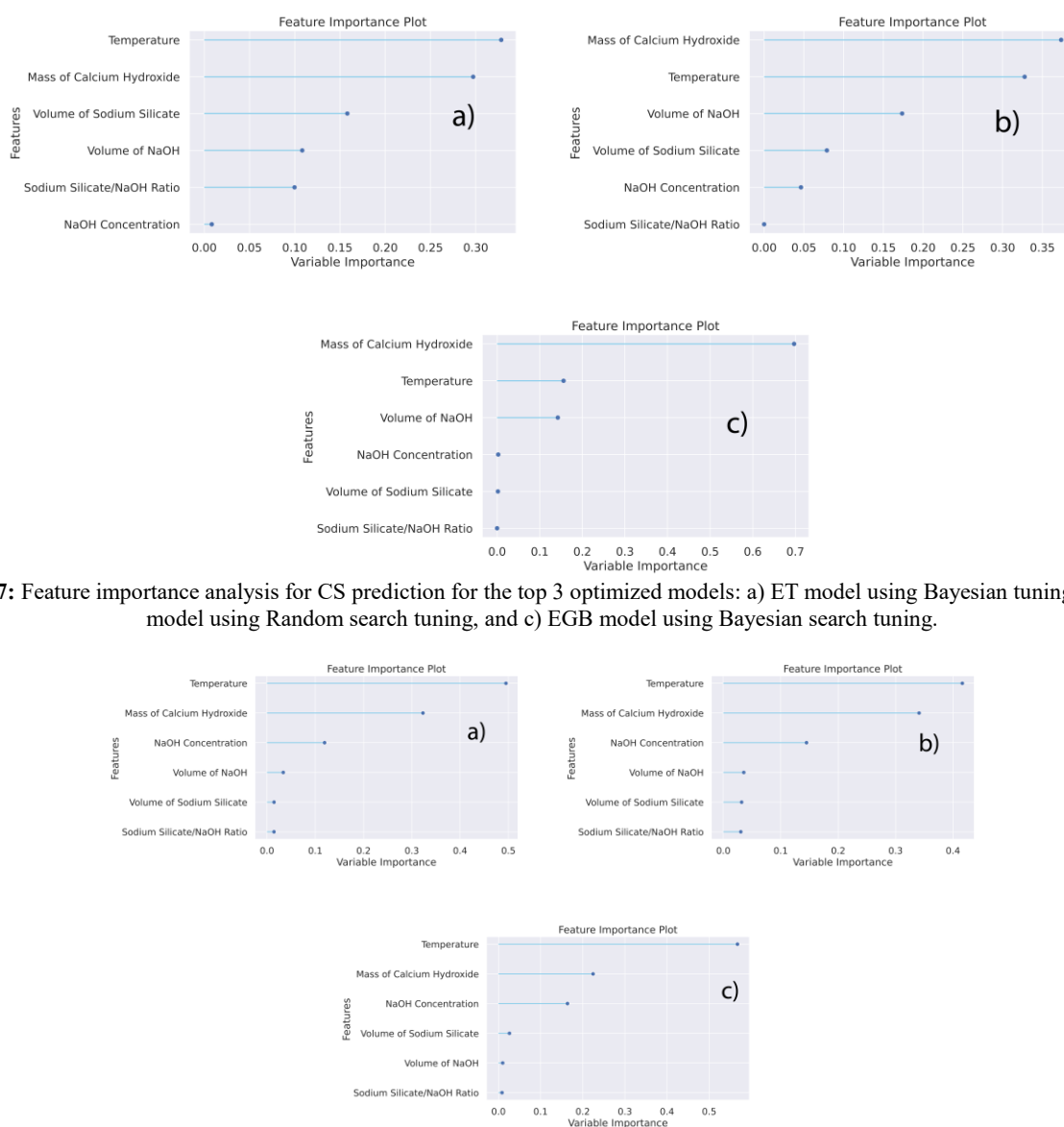


Fig. 7: Feature importance analysis for CS prediction for the top 3 optimized models: a) ET model using Bayesian tuning, b) EGB model using Random search tuning, and c) EGB model using Bayesian search tuning.

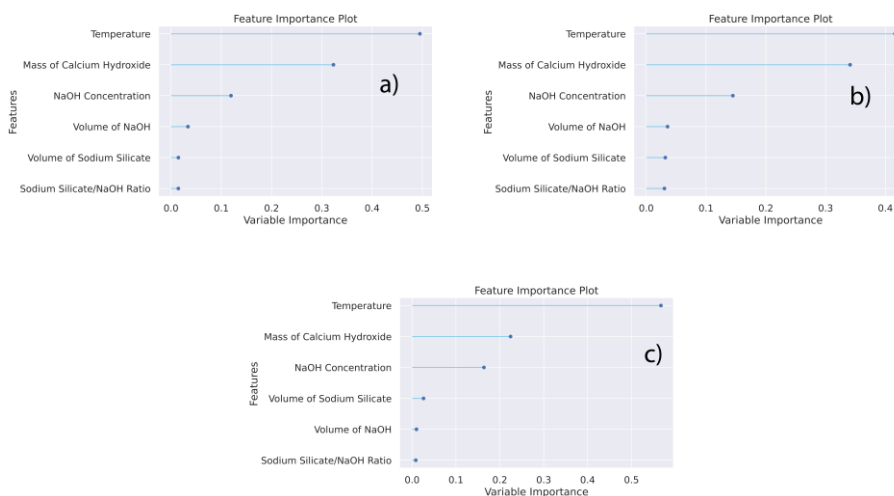


Fig. 8: Feature importance analysis for density prediction for the top 3 optimized models: a) ET model using Bayesian tuning, b) ET model using Random search tuning, and c) GB model using Bayesian tuning.

4. Conclusion

The regression algorithms were successfully implemented to predict Ecuadorian natural mordenite-based geopolymer's compressive strength and density. EGB, DT, ET, GB, and AdaBoost were utilized to forecast CS in terms of their statistical metrics. A hyperparameter optimization was performed using the Bayesian and Random Search methods. It was determined that the Extra Tree Regressor optimized by Bayesian tuning had the highest performance compared to the other machine learning models. Similarly, RF, ET, GB, EGB, and DT were the chosen models to implement density. The Extra Tree Regressor with Bayesian optimization had the best performance, but their statistical metrics were lower than CS due to the variability of the predicted results.

Also, the importance of features in the models was analyzed. The mass of calcium hydroxide and temperature are essential variables in predicting CS. Instead, for density prediction, temperature, sodium hydroxide concentration, and the mass of calcium hydroxide are the most crucial variables.

Larger datasets and inputs will be considered to improve the prediction accuracy of density, compressive strength, and other mechanical properties in future works.

5. Data availability

The datasets used in the machine learning models in this study and the code are available on Google Colab: <https://colab.research.google.com/drive/1KtJOa0rR4UpVrmYTDqIBIKH7OkbA1V8M?usp=sharing>.

References

- 1) A.A. Shahmansouri, H. Akbarzadeh Bengar, and E. Jahani, "Predicting compressive strength and electrical resistivity of eco-friendly concrete containing natural zeolite via gep algorithm," *Constr Build Mater*, **229** 116883 (2019). doi:10.1016/J.CONBUILDMAT.2019.116883.
- 2) M. Nawaz, A. Heitor, and M. Sivakumar, "Geopolymers in construction - recent developments," *Constr Build Mater*, **260** 120472 (2020). doi:10.1016/J.CONBUILDMAT.2020.120472.
- 3) Y. Wu, B. Lu, T. Bai, H. Wang, F. Du, Y. Zhang, L. Cai, C. Jiang, and W. Wang, "Geopolymer, green alkali activated cementitious material: synthesis, applications and challenges," *Constr Build Mater*, **224** 930–949 (2019). doi:10.1016/J.CONBUILDMAT.2019.07.112.
- 4) Tri Muji Susantoro, Sugihardjo, K. Wikantika, D. Sunarjanto, U. Pasarai, B. Widarsono, A. Rahmadi, M. Romli, P. Wahyudi, and S. Kepies, "CCUS-eor optimization to achieve zero emission program targets in northwest java basin," *Evergreen*, **10** (3) 1809–1818 (2023). doi:10.5109/7151730.
- 5) S. Singh, Suraj Kumar Singh, R. Kumar, A. Shrama, and S. Kanga, "Finding alternative to river sand in building construction," *Evergreen*, **9** (4) 973–992 (2022). doi:10.5109/6625713.
- 6) M. Schneider, M. Romer, M. Tschudin, and H. Bolio, "Sustainable cement production—present and future," *Cem Concr Res*, **41** (7) 642–650 (2011). doi:10.1016/J.CEMCONRES.2011.03.019.
- 7) M. Su, Q. Zhong, and H. Peng, "Regularized multivariate polynomial regression analysis of the compressive strength of slag-metakaolin geopolymer pastes based on experimental data," *Constr Build Mater*, **303** (March) 124529 (2021). doi:10.1016/j.conbuildmat.2021.124529.
- 8) Y. Peng, and C. Unluer, "Analyzing the mechanical performance of fly ash-based geopolymer concrete with different machine learning techniques," *Constr Build Mater*, **316** (November 2021) 125785 (2022). doi:10.1016/j.conbuildmat.2021.125785.
- 9) P. Saraswat, and R. Agrawal, "Artificial intelligence as key enabler for sustainable maintenance in the manufacturing industry: scope & challenges," *Evergreen*, **10** (4) 2490–2497 (2023). doi:10.5109/7162012.
- 10) S. Akhai, "Navigating the potential applications and challenges of intelligent and sustainable manufacturing for a greener future," *Evergreen*, **10** (4) 2237–2243 (2023). doi:10.5109/7160899.
- 11) R. Huerta-Soto, F. Francis, M. Asís-López, and J. Panduro-Ramirez, "Implementation of machine learning in supply chain management process for sustainable development by multiple regression analysis approach (mraa)," *Evergreen*, **10** (2) 1113–1119 (2023). doi:10.5109/6793671.
- 12) A. Arunika, Jaka Fajar Fatriansyah, and Venia Andira Ramadheena, "Detection of asphalt pavement segregation using machine learning linear and quadratic discriminant analyses," *Evergreen*, **9** (1) 213–218 (2022). doi:10.5109/4774236.
- 13) Gde Dharma Nugraha, B. Sudiarto, and K. Ramli, "Machine learning-based energy management system for prosumer," *Evergreen*, **7** (2) 309–313 (2020). doi:10.5109/4055238.
- 14) Tejas G. Patil, and Sanjay P. Shekhawat, "Artificial neural based quality assessment of guava fruit," *Evergreen*, **9** (2) 389–395 (2022). doi:10.5109/4794164.
- 15) I. El Naqa, and M.J. Murphy, "What Is Machine Learning?," in: *Machine Learning in Radiation Oncology*, Springer International Publishing, 2015: pp. 3–11. doi:10.1007/978-3-319-18305-3_1.
- 16) Z.-H. Zhou, "Machine Learning," Springer Singapore, Singapore, 2021. doi:10.1007/978-981-15-1967-3.
- 17) T. Hastie, R. Tibshirani, and J. Friedman, "Model Assessment and Selection," in: 2009: pp. 1–41. doi:10.1007/b94608_7.
- 18) H.J.P. Weerts, A.C. Mueller, and J. Vanschoren, "Importance of tuning hyperparameters of machine

- learning algorithms,” (2020). <http://arxiv.org/abs/2007.07588>.
- 19) P. Probst, and B. Bischl, “Tunability: Importance of Hyperparameters of Machine Learning Algorithms,” 2019. <http://jmlr.org/papers/v20/18-444.html>.
- 20) Joshila Grace L.K, A. Maneengam, Pradeep Kumar K V, and J. Alanya-Beltran, “Design and implementation of machine learning modelling through adaptive hybrid swarm optimization techniques for machine management,” *Evergreen*, **10** (2) 1120–1126 (2023). doi:10.5109/6793672.
- 21) M. Ghalan, and Rajesh Kumar Aggarwal, “Advancing human activity recognition: a novel waecn-bo approach for distinguishing highly correlated actions,” *Evergreen*, **10** (4) 2398–2411 (2023). doi:10.5109/7160930.
- 22) M. Ahmad, K. Rashid, Z. Tariq, and M. Ju, “Utilization of a novel artificial intelligence technique (anfis) to predict the compressive strength of fly ash-based geopolymer,” *Constr Build Mater*, **301** (July) 124251 (2021). doi:10.1016/j.conbuildmat.2021.124251.
- 23) W. Huo, Z. Zhu, H. Sun, B. Ma, and L. Yang, “Development of machine learning models for the prediction of the compressive strength of calcium-based geopolymers,” *J Clean Prod*, **380** (P2) 135159 (2022). doi:10.1016/j.jclepro.2022.135159.
- 24) F. Kuang, Z. Long, D. Kuang, X. Liu, and R. Guo, “Application of back propagation neural network to the modeling of slump and compressive strength of composite geopolymers,” *Comput Mater Sci*, **206** (January) 111241 (2022). doi:10.1016/j.commatsci.2022.111241.
- 25) P. Kumar, S. Sharma, and B. PRATAP, “Prediction of compressive strength of geopolymer fiber reinforced concrete using machine learning,” *Civil Engineering Infrastructures Journal*, (2024). doi:10.22059/cej.2024.364871.1956.
- 26) A. Bagheri, A. Nazari, and J. Sanjayan, “The use of machine learning in boron-based geopolymers: function approximation of compressive strength by ann and gp,” *Measurement (Lond)*, **141** 241–249 (2019). doi:10.1016/j.measurement.2019.03.001.
- 27) A. Ahmad, W. Ahmad, F. Aslam, and P. Joyklad, “Compressive strength prediction of fly ash-based geopolymer concrete via advanced machine learning techniques,” *Case Studies in Construction Materials*, **16** (December 2021) e00840 (2022). doi:10.1016/j.cscm.2021.e00840.
- 28) D.A. Emarah, “Compressive strength analysis of fly ash-based geopolymer concrete using machine learning approaches,” *Results in Materials*, **16** (July) 100347 (2022). doi:10.1016/j.rinma.2022.100347.
- 29) K.T. Nguyen, Q.D. Nguyen, T.A. Le, J. Shin, and K. Lee, “Analyzing the compressive strength of green fly ash based geopolymer concrete using experiment and machine learning approaches,” *Constr Build Mater*, **247** 118581 (2020). doi:10.1016/j.conbuildmat.2020.118581.
- 30) E. Gomaa, T. Han, M. ElGawady, J. Huang, and A. Kumar, “Machine learning to predict properties of fresh and hardened alkali-activated concrete,” *Cem Concr Compos*, **115** (October 2020) 103863 (2021). doi:10.1016/j.cemconcomp.2020.103863.
- 31) S.F.A. Shah, B. Chen, M. Zahid, and M.R. Ahmad, “Compressive strength prediction of one-part alkali activated material enabled by interpretable machine learning,” *Constr Build Mater*, **360** (July) 129534 (2022). doi:10.1016/j.conbuildmat.2022.129534.
- 32) B.S.R. Yeddula, and S. Karthiyaini, “Experimental investigations and gep modelling of compressive strength of ferrosialate based geopolymer mortars,” *Constr Build Mater*, **236** 117602 (2020). doi:10.1016/j.conbuildmat.2019.117602.
- 33) F. Rehman, S.A. Khokhar, and R.A. Khushnood, “ANN based predictive mimicker for mechanical and rheological properties of eco-friendly geopolymer concrete,” *Case Studies in Construction Materials*, **17** (July) e01536 (2022). doi:10.1016/j.cscm.2022.e01536.
- 34) K. Khan, W. Ahmad, M.N. Amin, A. Ahmad, S. Nazar, and M.A. Al-Faiad, “Assessment of artificial intelligence strategies to estimate the strength of geopolymer composites and influence of input parameters,” *Polymers (Basel)*, **14** (12) (2022). doi:10.3390/polym14122509.
- 35) A. Gogineni, I.K. Panday, P. Kumar, and R.Kr. Paswan, “Predicting compressive strength of concrete with fly ash and admixture using xgboost: a comparative study of machine learning algorithms,” *Asian Journal of Civil Engineering*, **25** (1) 685–698 (2024). doi:10.1007/s42107-023-00804-0.
- 36) B. Pratap, S. Sharma, P. Kumari, and S. Raj, “Mechanical properties prediction of metakaolin and fly ash - based geopolymer concrete using svr,” *Journal of Building Pathology and Rehabilitation*, **9** (1) 1 (2024). doi:10.1007/s41024-023-00360-9.
- 37) E.H. Alakara, S. Nacar, O. Sevim, S. Korkmaz, and I. Demir, “Determination of compressive strength of perlite-containing slag-based geopolymers and its prediction using artificial neural network and regression-based methods,” *Constr Build Mater*, **359** (September) 129518 (2022). doi:10.1016/j.conbuildmat.2022.129518.
- 38) R.K. Paswan, A. Gogineni, S. Sharma, and P. Kumar, “Predicting split tensile strength in portland and geopolymer concretes using machine learning algorithms: a comparative study,” *Journal of Building Pathology and Rehabilitation*, **9** (2) 129 (2024). doi:10.1007/s41024-024-00485-5.
- 39) H. Baykara, M.H. Cornejo, R. Murillo, A. Gavilanes, C. Paredes, and J. Elsen, “Preparation, characterization and reaction kinetics of green cement: ecuadorian natural mordenite-based

- geopolymers,” *Materials and Structures/Materiaux et Constructions*, **50** (3) 1–12 (2017). doi:10.1617/s11527-017-1057-z.
- 40) J.L.V. Lynch, H. Baykara, M. Cornejo, G. Soriano, and N.A. Ulloa, “Preparation, characterization, and determination of mechanical and thermal stability of natural zeolite-based foamed geopolymers,” *Constr Build Mater*, **172** 448–456 (2018). doi:10.1016/j.conbuildmat.2018.03.253.
- 41) A.A. Shahmansouri, M. Yazdani, S. Ghanbari, H. Akbarzadeh Bengar, A. Jafari, and H. Farrokh Ghatte, “Artificial neural network model to predict the compressive strength of eco-friendly geopolymer concrete incorporating silica fume and natural zeolite,” *J Clean Prod*, **279** 123697 (2021). doi:10.1016/j.jclepro.2020.123697.
- 42) J. Li, Q. Qin, Y. Wang, J. Wang, Q. Dong, K. Li, and X. Wang, “Properties prediction for self-compacting concrete incorporating activated fiber and stone chips,” *Mater Today Commun*, **33** 104615 (2022). doi:10.1016/J.MTCOMM.2022.104615.
- 43) G. Zhakypova, S. Uderbayev, N. Saktaganova, G. Abyieva, A. Budikova, and A. Zhapakhova, “Properties of fine-grained concrete using ash of kazakhstan,” *Evergreen*, **10** (2) 830–841 (2023). doi:10.5109/6792835.
- 44) D. Shekhar, and J. Godihal, “Exploring the mechanical and microstructural characteristics of recycled concrete hollow blocks: transforming waste into valuable resources,” *Evergreen*, **10** (4) 2195–2206 (2023). doi:10.5109/7160894.
- 45) P. Dwivedi, M. Maurya, K. Maurya, K. Srivastava, S. Sharma, and A. Saxena, “Utilization of groundnut shell as reinforcement in development of aluminum based composite to reduce environment pollution: a review,” *Evergreen*, **7** (1) 15–25 (2020). doi:10.5109/2740937.
- 46) S. Sharma, A. Kumar, S. Bano, and P. Kumar, “Soft computing techniques for analysing the mechanical properties of egg shell powder-based concrete,” *Advances in Civil and Architectural Engineering*, **15** (28) 119–132 (2024). doi:10.13167/2024.28.9.
- 47) X. Zhang, C. Dai, W. Li, and Y. Chen, “Prediction of compressive strength of recycled aggregate concrete using machine learning and bayesian optimization methods,” *Front Earth Sci (Lausanne)*, **11** (2023). doi:10.3389/feart.2023.1112105.
- 48) C. Daniel, X. Yin, X. Huang, J.A. Busari, A.I. Daniel, H. Yu, and Y. Pan, “Bayesian optimization-enhanced ensemble learning for the uniaxial compressive strength prediction of natural rock and its application,” *Geohazard Mechanics*, **2** (3) 197–215 (2024). doi:10.1016/j.ghm.2024.05.002.
- 49) H. Chen, X. Li, Y. Wu, L. Zuo, M. Lu, and Y. Zhou, “Compressive strength prediction of high-strength concrete using long short-term memory and machine learning algorithms,” *Buildings*, **12** (3) 302 (2022). doi:10.3390/buildings12030302.
- 50) W. Wang, Y. Zhong, G. Liao, Q. Ding, T. Zhang, and X. Li, “Prediction of compressive strength of concrete specimens based on interpretable machine learning,” *Materials*, **17** (15) 3661 (2024). doi:10.3390/ma17153661.
- 51) J. Snoek, H. Larochelle, and R.P. Adams, “Practical bayesian optimization of machine learning algorithms,” *In Advances in Neural Information Processing Systems*, 2951–2959 (2012). doi:https://doi.org/10.48550/arXiv.1206.2944.
- 52) J. Bergstra, and Y. Bengio, “Random search for hyper-parameter optimization,” *Journal of Machine Learning Research*, **13** 281–305 (2012).
- 53) M. Ali, “PyCaret: an open source, low-code machine learning library in python,” (2020). https://pycaret.gitbook.io/docs/.
- 54) Google Research, “Google colab,” (n.d.). https://colab.research.google.com/.
- 55) M. Waskom, “Seaborn: statistical data visualization,” *J Open Source Softw*, **6** (60) 3021 (2021). doi:10.21105/joss.03021.
- 56) J.D. Hunter, “Matplotlib: a 2d graphics environment,” *Comput Sci Eng*, **9** (3) 90–95 (2007). doi:10.5281/zenodo.7570264.
- 57) V. Rathakrishnan, S. Bt. Beddu, and A.N. Ahmed, “Predicting compressive strength of high-performance concrete with high volume ground granulated blast-furnace slag replacement using boosting machine learning algorithms,” *Sci Rep*, **12** (1) 1–16 (2022). doi:10.1038/s41598-022-12890-2.
- 58) G. Maroljlis, and A.J. Thakkar, “A new family of power transformations to improve normality or symmetry,” *Biometrika*, **156** (1) 87–90 (1989).
- 59) M. Ali, “PyCaret: an open source, low-code machine learning library in python. training functions in pycaret,” (2020). https://pycaret.gitbook.io/docs/get-started/functions/train#compare_models.
- 60) T. Head, M. Kumar, H. Nahrstaedt, G. Louppe, and I. Shcherbatyi, “Scikit-optimize/scikit-optimize,” (2021). doi:10.5281/ZENODO.5565057.
- 61) T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, “Optuna: A Next-generation Hyperparameter Optimization Framework,” in: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Association for Computing Machinery, 2019: pp. 2623–2631. doi:10.1145/3292500.3330701.
- 62) L. Prechelt, “Early stopping – but when? neural networks: tricks of the trade,” *Springer-Verlag Berlin Heidelberg*, **1524** 55–69 (1998). https://link.springer.com/chapter/10.1007/3-540-49430-8_3.
- 63) M. Kuhn, and K. Johnson, “Applied Predictive Modeling,” Springer, 2013. doi:10.1007/978-1-4614-6849-3.

- 64) I. Kofi Nti, O. Nyarko-Boateng, and J. Aning, "Performance of machine learning algorithms with different k values in k-fold crossvalidation," *International Journal of Information Technology and Computer Science*, **13** (6) 61–71 (2021). doi:10.5815/ijitcs.2021.06.05.
- 65) D.C. Montgomery, E.A. Peck, and G.G. Vining, "Introduction to Linear Regression Analysis," Fifth edit, Wiley series in probability and statistics, New Jersey, 2012.
- 66) J. Brownlee, "Machine Learning Mastery With Python.," 2016.
- 67) G. James, D. Wittern, T. Hastie, and R. Tibshirani, "An Introduction to Statistical Learning with Applications in R," Springer, 2013. doi:10.1007/978-1-4614-7138-7.
- 68) A. Nafees, M.F. Javed, S. Khan, K. Nazir, F. Farooq, F. Aslam, M.A. Musarat, and N.I. Vatin, "Predictive modeling of mechanical properties of silica fume-based green concrete using artificial intelligence approaches: mlpnn, anfis, and gep," *Materials*, **14** (24) 1–28 (2021). doi:10.3390/ma14247531.
- 69) A. De Myttenaere, B. Golden, B. Le Grand, and F. Rossi, "Mean absolute percentage error for regression models," *Neurocomputing*, **192** 38–48 (2016). doi:10.1016/J.NEUCOM.2015.12.114.
- 70) T. Chen, and C. Guestrin, "XGBoost: A scalable tree boosting system," in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, 2016: pp. 785–794. doi:10.1145/2939672.2939785.
- 71) L. Breiman, J.H. Friedman, C.J. Stone, and R.A. Olshen, "Classification and regression trees," Chapman and Hall, 1984. doi:https://doi.org/10.1201/9781315139470.
- 72) S. Raschka, and V. Mirjalili, "Python Machine Learning," Packt Publishing Ltd., 2019.
- 73) J.H. Friedman, "Greedy function approximation: a gradient boosting machine," *Ann Stat*, **29** (5) 1189–1232 (2001). doi:10.1214/aos/1013203451.
- 74) A.J. Ferreira, and A.T. Figueiredo, "Ensemble Machine Learning," in: Ensemble Machine Learning, 2012. doi:10.1007/978-1-4419-9326-7.
- 75) Y. Freund, and R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J Comput Syst Sci*, **55** 119–139 (1997).
- 76) L. Breiman, "Random forests," *Mach Learn*, **45** 5–32 (2001). doi:https://doi.org/10.1023/A:1010933404324.
- 77) G. Rebala, R. Ajay, and S. Churiwala, "An introduction to machine learning," 2019. doi:10.4018/978-1-7998-0414-7.ch003.
- 78) P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach Learn*, **63** (1) 3–42 (2006). doi:10.1007/s10994-006-6226-1.
- 79) R. Marée, L. Wehenkel, and P. Geurts, "Extremely Randomized Trees and Random Subwindows for Image Classification, Annotation, and Retrieval," in: A. Criminisi, J. Shotton (Eds.), Springer London, London, 2013: pp. 125–141. doi:10.1007/978-1-4471-4929-3_10.
- 80) M. Ali, "PyCaret: an open source, low-code machine learning library in python. scale and transform.," (2020). <https://pycaret.gitbook.io/docs/get-started/preprocessing/scale-and-transform>.
- 81) K.C. Onyelowe, A.M. Ebid, A. Riofrio, H. Baykara, A. Soleymani, H.A. Mahdi, H. Jahangir, and K. Ibe, "Multi-objective prediction of the mechanical properties and environmental impact appraisals of self-healing concrete for sustainable structures," *Sustainability (Switzerland)*, **14** (15) (2022). doi:10.3390/su14159573.
- 82) P.J. Huber, "Robust estimation of a location parameter," *The Annals of Mathematical Statistics*, **35** (1) 73–101 (1964). doi:10.1214/aoms/1177703732.
- 83) G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.Y. Liu, "LightGBM: a highly efficient gradient boosting decision tree," *Adv Neural Inf Process Syst*, **2017-Decem (Nips)** 3147–3155 (2017).
- 84) F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn. dummy regressor.," (2011). <https://scikit-learn.org/stable/modules/generated/sklearn.dummy.DummyRegressor.html>.
- 85) K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *Journal of Machine Learning Research*, **7** 551–585 (2006).
- 86) B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, **32** (2) 385–479 (2004). doi:10.1214/009053604000000067.
- 87) Z. Wan, Y. Xu, and B. Šavija, "On the use of machine learning models for prediction of compressive strength of concrete: influence of dimensionality reduction on the model performance," *Materials*, **14** (4) 1–23 (2021). doi:10.3390/ma14040713.
- 88) G. Bonaccorso, "Machine Learning Algorithms," 2023. doi:10.4018/978-1-7998-9220-5.ch054.
- 89) Y. Wu, and Y. Zhou, "Hybrid machine learning model and shapley additive explanations for compressive strength of sustainable concrete," *Constr Build Mater*, **330** 127298 (2022). doi:10.1016/J.CONBUILDMAT.2022.127298.
- 90) P. Mehta, M. Bukov, C.-H. Wang, A.G.R. Day, C. Richardson, C.K. Fisher, and D.J. Schwab, "A high-bias, low-variance introduction to machine learning

- for physicists,” *Phys Rep*, **810** 1–124 (2019). doi:10.1016/j.physrep.2019.03.001.
- 91) R. Shrinkage, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society. Series B (Methodological)*, **58** (1) 267–288 (1996). [jstor.org/stable/2346178](https://www.jstor.org/stable/2346178).
 - 92) H. Zou, and T. Hastie, “Regularization and variable selection via the elastic net,” *J R Stat Soc Series B Stat Methodol*, **67** (2) 301–320 (2005). doi:10.1111/j.1467-9868.2005.00503.x.
 - 93) F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn. lasso least angle regression,” (2011). https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LassoLars.html (accessed March 9, 2023).
 - 94) H.S. Ullah, R.A. Khushnood, J. Ahmad, and F. Farooq, “Predictive modelling of sustainable lightweight foamed concrete using machine learning novel approach,” *Journal of Building Engineering*, **56** (2022). doi:10.1016/j.job.2022.104746.
 - 95) A. Harmaji, M.C. Kirana, and R. Jafari, “Machine learning to predict workability and compressive strength of low- and high-calcium fly ash-based geopolymers,” *Crystals (Basel)*, **14** (10) 830 (2024). doi:10.3390/cryst14100830.