Comparison Studies Between Machine Learning Optimisation Technique on Predicting Concrete Compressive Strength

Vimal Rathakrishnan (SC22732@student.uniten.edu.my)
Universiti Tenaga Nasional
Salmia Beddu
Universiti Tenaga Nasional
Ali Najah Ahmed
Universiti Tenaga Nasional

Research Article

Keywords: Concrete Compressive Strength, Machine Learning Optimisation, Prediction, XGBoost, Grid Search, Random Search, Bayesian Optimisation

Posted Date: April 12th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-381936/v1

Comparison Studies Between Machine Learning Optimisation Technique on Predicting Concrete Compressive Strength

Vimal Rathakrishnan[1], Salmia Bt. Beddu[1] and Ali Najah Ahmed[1]

[1]Universiti Tenaga Nasional, Malaysia

Abstract

In this research, a comparison study of the machine learning (ML) optimisation technique to predict the compressive strength of concrete is discussed. In previous studies, researchers focused on identifying the machine learning model by comparing, ensemble, bagging, and fusion methods in predicting the concrete strength. In this research, an ML model hyper-parameter optimisation is used to improve the prediction accuracy and performance of the model. Extreme gradient boosting (XGBoost) is used as the base model to perform the prediction, as the XGBoost has a built-in model ensemble, bagging, and boosting algorithms. Grid Search, Random Search, and Bayesian Optimisation are selected and used to optimise the hyperparameters of the XGBoost model. For this particular prediction study, the optimised models based on Random Search performed better than other optimisation methods. The Random Search optimisation method showed substantial improvements in prediction accuracy, modelling error and computation time.

Keyword

Concrete Compressive Strength, Machine Learning Optimisation, Prediction, XGBoost, Grid Search, Random Search, Bayesian Optimisation

1. Introduction

1.1 Literature Review

Concrete has been commonly used in construction and architecture due to its favourable engineering properties. Combining the ability to cast and harden at ambient temperatures, concrete is a popular option in constructing structural elements, especially in high-rise buildings [1]. The benefits of concrete include high compressive strength and excellent water resistance, allowing it to be the material of choice for structures that need a solid foundation to withstand critical environmental conditions such as tunnels, dams, and reservoirs [2].

In general, concrete comprises four primary components: coarse aggregate, fine aggregate, cement, and water. Concrete's economic value allows it to be widely used in constructions and the accessibility to the material which is available in the local market. It also demonstrates excellent benefits over other construction materials such as steel, and concrete can be produced with minimum effort. In certain instances, supplementary materials like fly ash (PFA)[1][2], blast furnace slag (GGBS)[3], silica fume[4], and other industrial waste are added in concrete to enhance the mechanical properties of the concrete [3]. The introduction of industrial waste[5][6] into concrete offers environmental benefits while increasing the longevity and resiliency of concrete structures.

1.2 Problem Statement

When dealing with concrete materials, one of the critical issues is selecting the concrete materials and predicting the mechanical properties of concrete, i.e., compressive strength due to the heterogeneous mixture that contains various types of ingredients. So, it is essential to provide stable and accurate predictive models at the early stage of construction, so the cost associated with the risk of non-compliance concrete during construction can be reduced [4].

With the use of suitable models, it can lead to success in finding combination inputs that can achieve meaningful outcomes and, at the same time, saves considerable time and money. However, it is challenging to construct a prediction model that reliably predicts concrete strength due to the nonlinear relationship between materials and concrete strength.

In recent years, there has been extensive research and development in artificial intelligence (AI). Machine learning is a sub-class of AI that self-learning through algorithms and enables it to improve its performance based on previous datasets/experience. With minimal human input, machine learning algorithms will automatically learn and improves over time [5].

Given the popularity of machine learning, especially in concrete technology, various studies have been conducted using machine learning approaches[7]. Many empirical and statistical models, i.e., linear and nonlinear regression algorithms, were employed to predict the properties of concrete [4]. Multiple Linear Regression (MLR)[3][4], Support Vector Machine(SVR)[6][7][1][8], Multilayer Perceptron(MLP)[9][1][10], Artificial Neural Network(ANN)[11][8][7][10][12] and Gradient Boosting[1][13][8] are most commonly used machine learning algorithms/models to predict the mechanical and chemical properties of concrete. In general, the compressive strength prediction was undertaken for several type of concrete i.e., normal concrete [8][1][14], high-performance concrete [13][15][16], ultra-high-performance concrete [11][17], and green concrete with supplementary cementitious material i.e., fly ash [12][18], blast furnace slag [3] and recycled aggregates [19]. Machine learning is also used to predict other mechanical and chemical properties of concrete, i.e., prediction of concrete shear strength [20], tensile strength [1], flexural strength [14], the thermal conductivity of concrete [9], and chloride concentration of concrete [10][21].

Among all the machine learning models used for concrete compressive strength prediction, XGBoost models are more accurate, stable, and efficient than others [8][1]. Tianqi Chen and

Guestrin developed XGBoost, and the method uses the conventional tree gradient boosting algorithm [8][22] to create state-of-the-art algorithms, the 'extreme gradient boosting' [23]. The multiple Kaggle competition winner 'XGBoost' is a highly effective machine learning algorithm due to its scalable tree boosting system and sparsity-aware algorithm in modelling structured datasets. However, in most previous studies, the XGBoost model was used for comparison studies and evaluated its performance.

1.3 Objectives

In this paper, several hyper-parameter optimisation methods, namely 'Grid Search' and 'Random Search' and 'Bayesian Optimisation,' are used to optimise the prediction accuracy of the XGBoost models. As a reference, several regression-based machine learning models were used to evaluate and compare the performance of XGBoost models.

The fundamentals behind XGBoost models are defined in Section 2 and followed by the statistical properties of the dataset & modelling discussed in Section 3. Results of the XGBoost models with various models and optimisations are compared, and the importance of each hyper-parameter optimisation is analysed in Section 4. The conclusion and significance of model optimisation in XGBoost modelling are discussed in Section 5.

2 XGBoost

2.1 Fundamental Theoretical of XGBoost

XGBoost is a decision tree-based ensemble Machine Learning algorithm that uses gradient boosting to make predictions for unstructured data, i.e., images. The algorithm has been the source of countless cutting-edge applications, and it has been the driving force behind many of these recent advances. It's been widely used as industrial solutions such as customer churn prediction[9], applicant risk assessment[10], malware detection[11], stock market selection[12], classification of traffic accidents[13], diseases identification[14], and even in predicting the death of patience during SARS-COV-2(Covid-19) treatment[15]. In general, the XGBoost algorithms are the evolution of decision tree algorithms that were improved over time. Figure 1 below shows the development of decision tree-based algorithms to XGBoost.

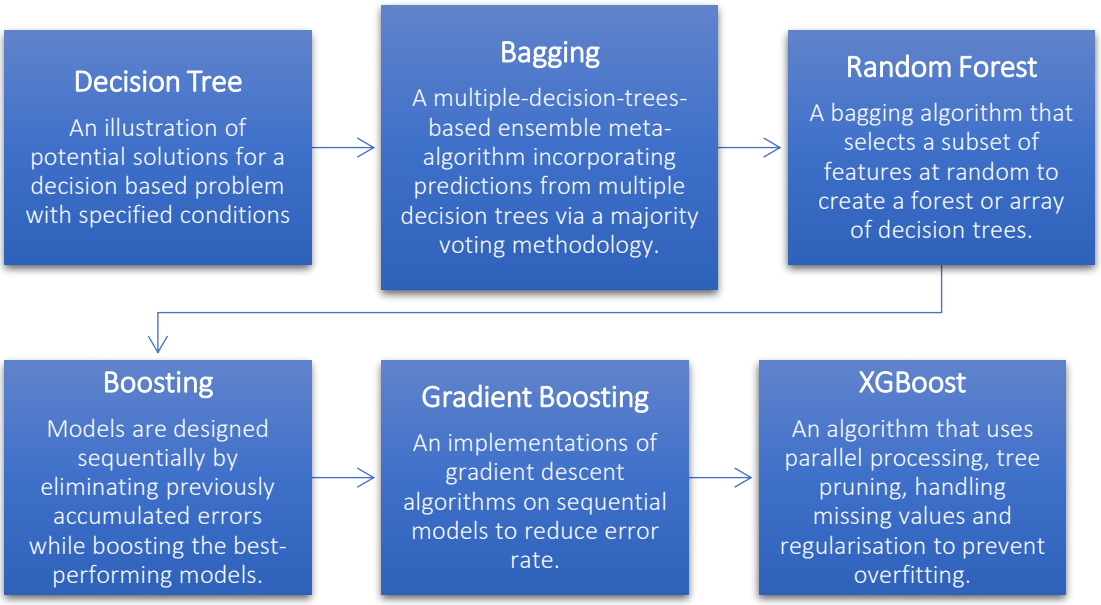

Figure 1 The Evolution of XGBoost

The most significant benefit of XGBoost is its scalability across any condition [16]. Like GBM's ensemble tree method, XGBoost employs the concept of boosting weak learners using gradient descent architecture. However, XGBoost outperforms the GBM algorithm by superior

optimisation and algorithmic improvement. Figure 2 illustrates the advantages of XGBoost, which contributes to an exceptional prediction result.

Parallelization	XGBoost builds parallel trees based on sequential implementation. The loop order is reversed using initialisation via a global search and sorting using parallel threads. This increases algorithmic efficiency by allowing parallelization without any overhead.
Tree Pruning	The greedy stopping criterion allows the loss to be below some threshold at the split point. Using a max depth parameter, the XGBoost algorithm begins pruning the trees backwards. This 'depth-first' approach increases computational efficiency.
Hardware Optimization	XGBoost is a cache aware algorithm which allocate internal buffers in each thread to store gradient statistics. In addition, it is enhanced with out-of-core computation when handling large datasets.
Regularization	The approach penalises models that are too complex by applying LASSO and Ridge regularisation to reduce model overfitting.
Sparsity Awareness	XGBoost automatically learns sparse features for inputs by naturally handling different degrees of sparsity in the data, and effectively handles different kinds of sparsity in the training dataset.
Weighted Quantile Sketch	The distributed weighted Quantile Sketch algorithm in XGBoost finds the optimal split points between weighted datasets.
Cross Validation	The built-in cross-validation in XGBoost takes away the need to specify the exact number of boosting iterations needed in a single run.

Figure 2 The Advantages of XGBoost

The XGBoost algorithm applies an adaptive regularisation technique to the objective function, and regularisation is used to keep the complexity of the model low and prevent overfitting, as shown in Eq. (1),

$$Obj(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \quad (1)$$

where $\Omega(f) = \gamma T + \frac{1}{2}\lambda\omega^2$, (regularisation term)

and y_i = predicted value,

$\hat{y}_i$ = real value,

f_k = decision tree,

T = leaves of each decision tree,

ω = complexity of decision tree,

$\gamma\ and\ \lambda$ = penalty coefficient

The objective function built by the iteration of the XGBoost is shown in Eq. (2):

$$Obj^t = \sum_{i=1}^{n} l\left(y_i, \hat{y_i}^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \quad (2)$$

Eq. (3) gives the Taylor expansion of the objective function. Taylor's method in the second-order makes the convergence speed of the model increase, which allows for the optimal global solution to be obtained.

$$Obj^t = \sum_{i=1}^{n} l\left(y_i, \hat{y_i}^{(t-1)} + g_i f_t(x_i)\right) + \frac{1}{2} h_i f_t^2(x_i) + \Omega(f_t) \quad (3)$$

where $g^i = \partial_{\hat{y}^{(t-1)}}\, l\left(y_i, \hat{y_i}^{(t-1)}\right)$ is a first-order derivative and $h^i = \partial^2_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y_i}^{(t-1)}\right)$ is a second derivative. The aim is to establish optimal tree structures by incrementally adding partitions to the current leaf nodes.

When each decision tree splits, two branches are formed. We can measure the information gain from the split by measuring the target function before and after the split. Eq. (4). If the fixed gain is less than the splitting gain or the number of divisions exceeds the maximum depth of division, the split ends and the final model is obtained.

$$Gain = \frac{1}{2}\left[\frac{g_l^2}{h_l^2 + \lambda} + \frac{g_r^2}{h_r^2 + \lambda} + \frac{(g_l + g_r)^2}{h_l + h_r + \lambda}\right] - \gamma \quad (4)$$

2.2 Hyperparameters in XGBoost

The most commonly used machine-learning algorithms are famous for learning patterns and regularities in the data automatically by tuning the hyperparameters. In machine learning models, the hyperparameters choose decision variables at each node and the numeric thresholds that influence the predictions. In XGBoost, hyperparameters are categorised into three, i.e., general parameters, booster parameters, and learning task parameters. Table 1 below shows the categories of hyperparameters and the critical parameter in XGBoost.

Table 1 Summary of Key Hyperparameter in XGBoost

Category	Description	Hyperparameters
General Parameters	Hyperparameters used for boosting, i.e., tree or linear model selection	• booster (model boosting parameter)
Booster Parameters	Optimisation of booster parameters to prevent overfitting	• eta (learning rate) • min_child_weight (minimum sum of instance weights needed in a child) • max_depth (maximum depth of a tree) • gamma (minimum loss reduction required to do a split) • subsample (section of observations to be randomly sampled for each tree) • colsample_bytree (section of columns to be randomly sampled for each tree)

		• colsample_bylevel (subsample ratio of columns for each split in each level) • lambda (L2 regularisation term on weights) • alpha (L1 regularisation term on weights)
Learning Task Parameters	Hyperparameters used to define optimisation objective to the calculated metric	• objective (minimisation of loss function) • eval_metric (evaluation metric for data validation)

The influence of hyperparameters on a model is well known; however, it is difficult to determine the best values for a hyperparameter or the best combinations of hyperparameters for a given dataset. Hyperparameter optimisation or hyperparameter tuning is an approach used to evaluate various values for model hyperparameters and select a subset that results in a model with the best predictive results on a given dataset.

In optimisation, a search space is established as an n-dimensional volume, where each hyperparameter characterises a different dimension, and the value of that dimension may be a true, integer, or categorical value. Points in the search space are vectors that have the required values for a given set of hyperparameters. Optimisation aims to find a parameter that performs in the best output of the model after training, i.e., the most accurate or the least error model. Various optimisation algorithms can be used, with two of the most reliable and simplest methods are Grid Search and Random Search. As a comparison, advanced optimisation techniques, i.e., Bayesian Optimisation, were also used in this research.

2.3 Grid Search

Grid Search optimisation is a comprehensive parameter searching approach where the model is tested and evaluated with every combination of the parameter values, and the optimal combination is selected. In Grid Search, the parameters are successively allocated to each model, generating a multidimensional mesh grid theoretically created with different values. Each mesh grid node represents a set of parameters, and model calibration are established at each node according to the predetermined values of parameters. To confirm the nodes are the most efficient, the algorithm continues by screening the entire grid, similar to the partial exhaustion method. Considering simplicity and representative, the best possible outcomes are to be obtained with cross-validation [17].

2.4 Random Search

The random search focuses on the use of random combinations to optimise the hyperparameters of a built model. It measures random combinations of a set of values to optimise decent outcomes, with the function tested at any number of random combinations in the parameter space. The chances of discovering the optimal parameter are relatively higher in random search due to various search patterns in the model are trained on the optimised parameters without aliasing. Random search is best for lower dimensional data as this method takes less time and iterations to find the right parameter combination [18].

2.5 Bayesian Optimisation

The fundamental difference between Bayesian optimisation and other methods is that it generates a probabilistic model and then uses it to decide where to test the function next while

simultaneously controlling for uncertainty. The basic principle is to use all available information in the model and use gradient and Hessian approximations only if possible [19].

This allows for finding the minimum of complex non-convex functions with fewer evaluations at the expense of evaluating where to look next. When using Bayesian optimisation, two essential decisions must be made. Firstly, to express assumptions about the function being optimised, a prior over function must be chosen. As this suits our purposes, we will use the Gaussian method for its simplicity and tractability. Secondly, an acquisition function is selected to establish a utility function from the model posterior, which is then used to find the next evaluation stage.

3 Methodology

3.1 Model Structure

In general, the machine learning model structure involves several key processes. Figure 3 demonstrates the step-by-step process for training, optimising, and validating the XGBoost model for concrete compressive strength prediction. There are six main processes which involve in the development of the optimised XGBoost model. Listed below brief explanation for each step:

i. Data Collection – This involves data collection from the research paper/repository and compiling the data accordingly.
ii. Data Pre-Processing – This step is required to identify and arrange the collected data by sorting out the missing values and normalise the data set for model development
iii. Model Selection - For this study, XGBoost has used the machine learning algorithm for compressive strength prediction.

iv. Hyper-parameter Optimisation – Three different optimisation algorithms, i.e., Grid Search, Random Search, and Bayesian Optimisation was used to optimise the hyper-parameter of XGBoost

v. Model Evaluation – All the models are compared, and the best performing algorithms are selected based on evaluation metrics, i.e., R^2, MSE, RMSE, MAE.

vi. Analysis and Reporting – A case study is performed based on comparing various machine learning models, optimisation parameters, and evaluation metrics.

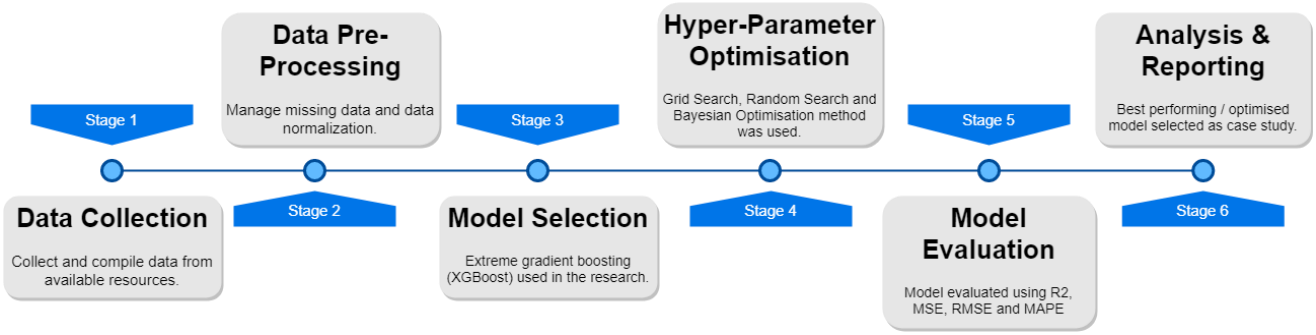

Figure 3 Step by step of XGBoost Modelling and Optimisation

3.2 Data Collection and Pre-Processing

Dataset of concrete compressive strength with 1030 samples is collected from the UCI Machine Learning Repository [20]. Generally, there are seven main ingredients, i.e., OPC, GGBS, PFA, Water, Admixture, Aggregate, and Sand, available in the dataset with strength results from 1 day to 365 days. Out of 1030 batches, 564 batches contain GGBS, and 464 batches contain PFA. Details of statistical metrics such as count, mean, standard deviation, maximum, and the minimum value are listed in Table 2 below.

Table 2 Summary of Statistical Analysis of the Concrete Material Composition

	OPC	GGBS	PFA	Water	Admixture	Aggregate	Sand	Age	Strength
Count	1030	564	464	1030	651	1030	1030	1030	1030
Mean	281.17	134.95	120.29	181.57	9.81	972.92	773.58	45.66	35.82
Standard Deviation	104.51	73.15	33.68	21.36	4.58	77.75	80.18	63.17	16.71
Min	102.00	0.02	24.46	121.75	1.72	801.00	594.00	1.00	2.33
Max	540.00	359.40	200.10	247.00	32.20	1145.00	992.60	365.00	82.60

Statistical analysis was performed to gain a deeper understanding of the correlation between all input and output features. The statistical measurement is significant as it defines the function in terms of its relationship to other parameters. Eventually, the statistical analysis can contribute to the optimisation of the predictive model that maximises predicting outcomes. Among those available in the literature, Pearson's approach will be used to calculate the correlation coefficient as follows

$$Pearson\ Correlation\ Coefficeint, r = \frac{\sum(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum(x_i-\bar{x})^2 \sum(y_i-\bar{y})^2}} \quad (5)$$

where;

r = correlation coefficient

x_i = values of the x-variable in a sample

$\bar{x}$ = mean of the values of the x-variable

y_i = values of the y-variable in a sample

$\bar{y}$ = mean of the values of the y-variable

The Pearson correlation coefficient, also known as the r coefficient, indicates the strength of a linear relationship between two variables. A Pearson correlation seeks to map a best-fit line by two variables, and the Pearson correlation coefficient, r, represents how far apart from all of the data points are from this line.

Pearson's correlation coefficient (r) can range from +1 to -1, and zero reflects no correlation between the two variables. The greater the connection between the two variables, the closer the Pearson correlation coefficient, r, would be to either +1 or -1. If all the data points are found on the line of best fit, the value of +1 or -1 has been reached; no data points are showing any deviation away from this line. For this research, the correlation between all the parameters was analysed and illustrated in the correlation heatmap in Figure 4.

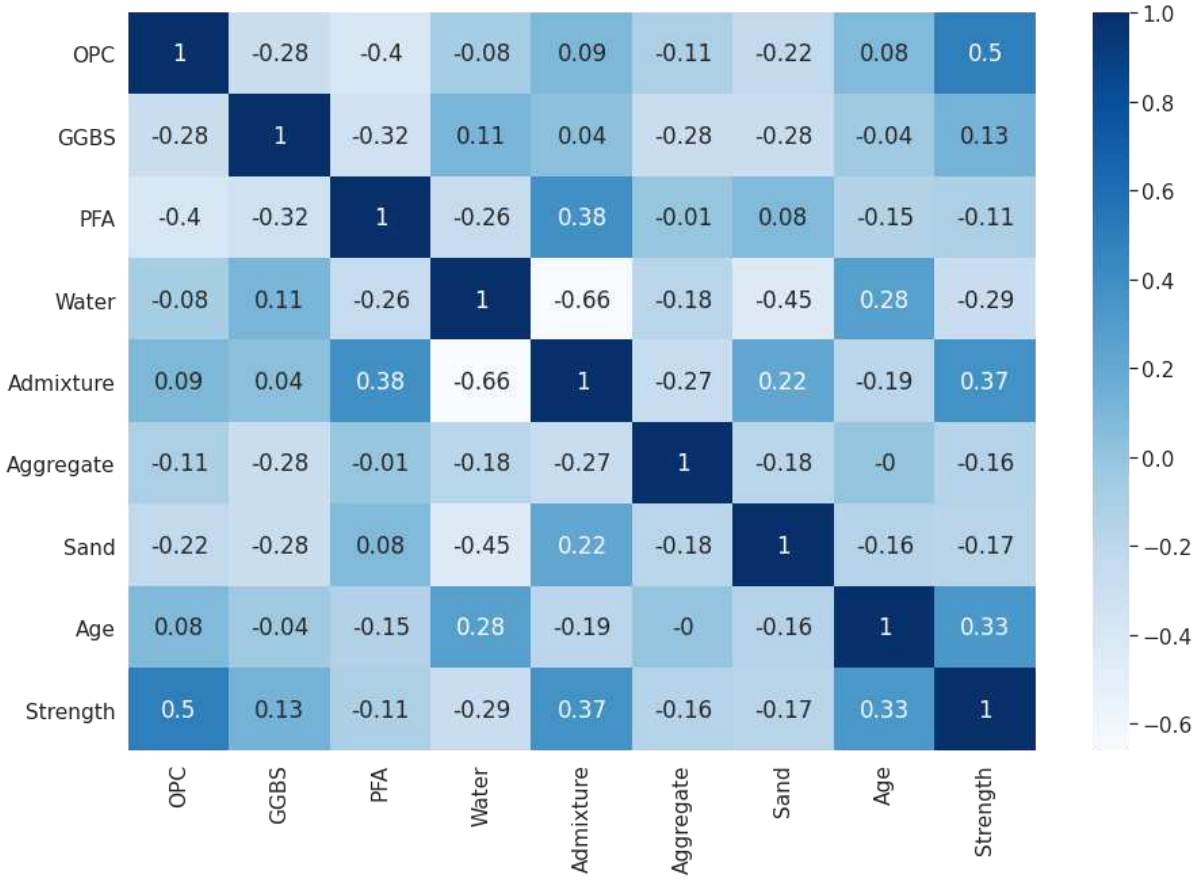

Figure 4 Pearson’s Correlation Heatmap

As shown in Figure 4 above, it can be observed that the correlation between input and output parameters is relatively low. The correlation coefficient is generally in the range of -0.66 to 0.50, and only aggregate is negatively correlated with all other parameters.

An additional statistical analysis of each parameter was also performed to ensure that the training dataset comprised input parameters with overall values encompassing the entire range of the dataset. Figure 5-11 shows the distribution and correlation between input parameters with output parameters, i.e., strength.

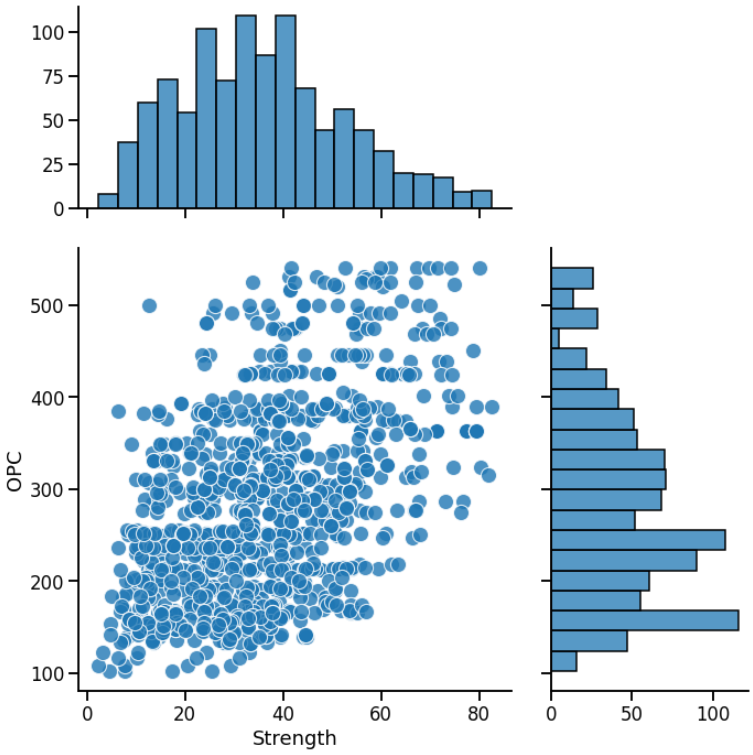

Figure 5 Correlation Between OPC and Strength

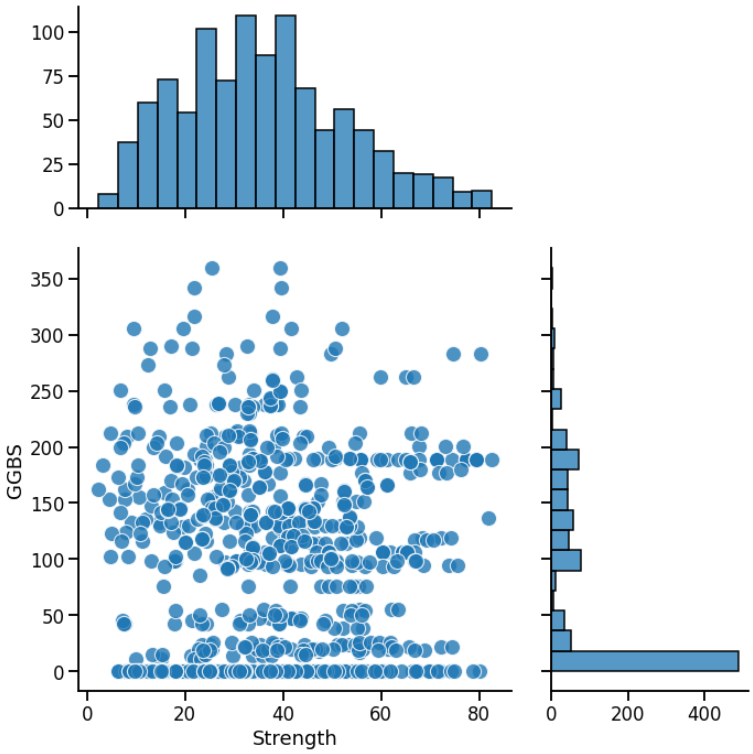

Figure 6 Correlation Between GGBS and Strength

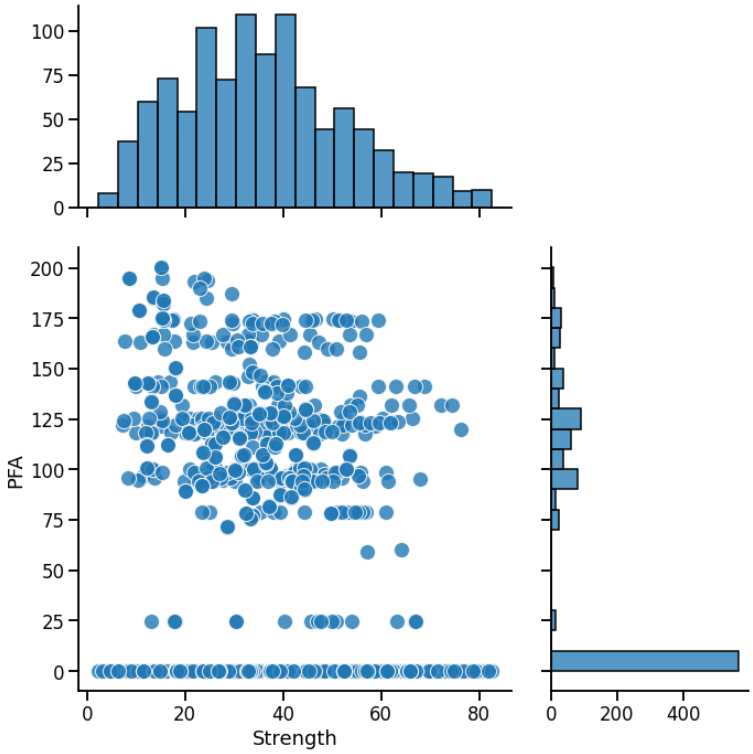

Figure 7 Correlation Between PFA and Strength

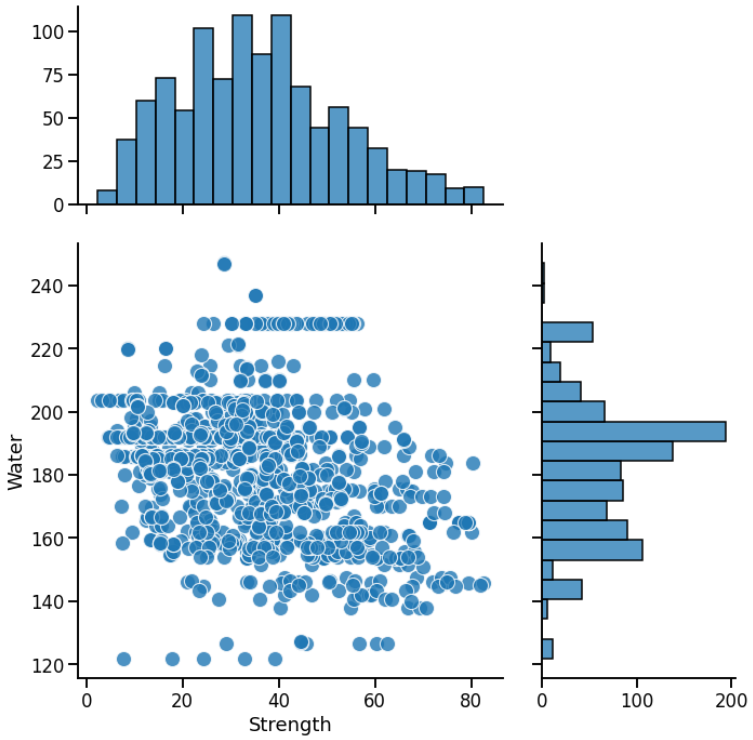

Figure 8 Correlation Between Water and Strength

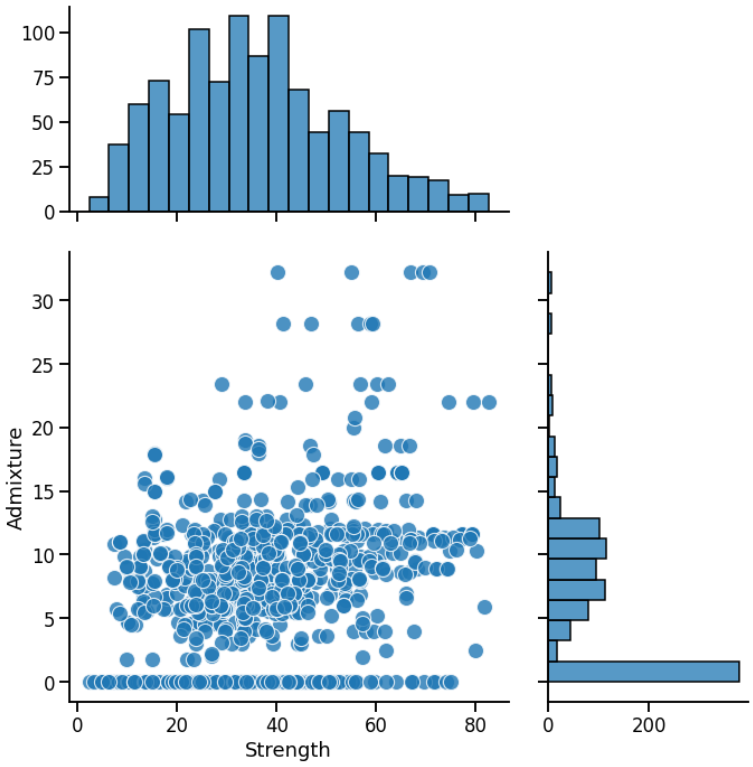

Figure 9 Correlation Between Admixture and Strength

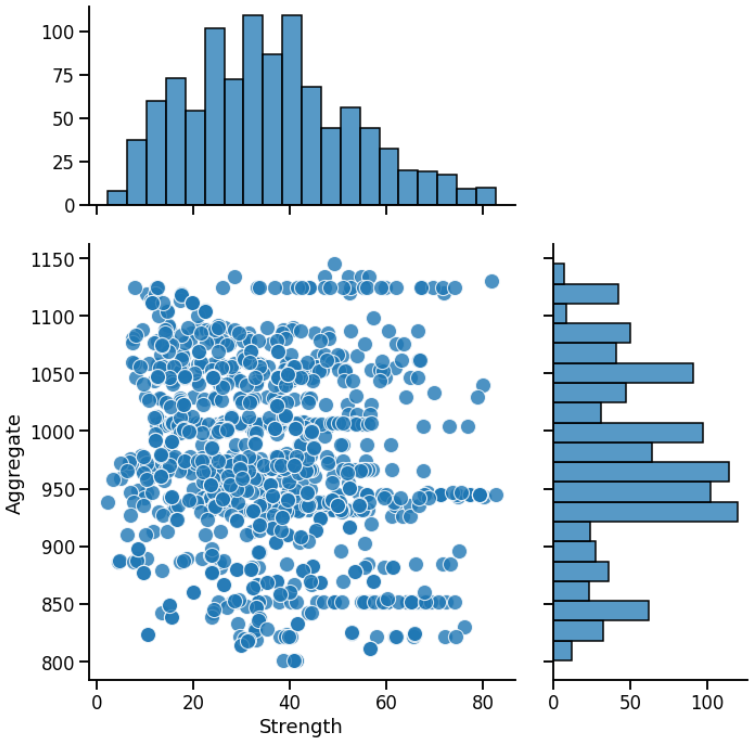

Figure 10 Correlation Between Aggregate and Strength

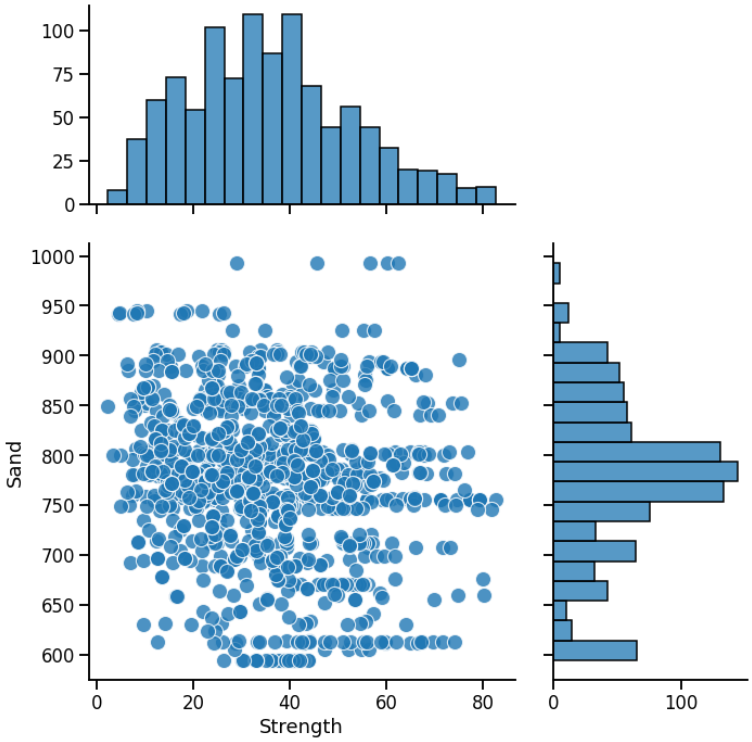

Figure 11 Correlation Between Sand and Strength

Figure 5-11 indicates that the input parameter data is well distributed, and only for GGBS, PFA, and admixture, a high distribution for value 0 is noticed. However, this is not likely to influence the modelling as the 0 value was considered that the mix design contains no GGBS, PFA, or admixture rather than treating it as missing values.

To train and evaluate XGBoost model prediction results, the dataset was randomly partitioned into two sets, i.e., a training set and a testing set. Around 75% of the primary dataset's data records were used to train the XGBoost models, while 25% were used for testing. This theoretical percentage split of 75% to 25% is widely used in past research [21][22][4][8].

Before training machine learning models, pre-processing data is required. To prevent training from being dominated by one or a few features with large magnitude, features should be normalised so that their range is consistent. The normalisation process is achieved by applying the normalisation process to the range from 0 to 1 before training. For each input element, the data points are divided by the highest magnitude. In the training process, the performance features' predictive effects are mapped back to their original scale.

3.3 Model Evaluation

In this paper, four separate statistical measurement parameters were used to calculate the prediction efficiency of the XGBoost models. In simpler terms, the evaluation parameters estimate the accumulated error in predictions concerning actual observations. The statistical parameters are: coefficient of determination (R2), mean square error (MSE), mean absolute error (MAE), and root mean squared error (RMSE). These mathematical formulations are defined in Eq. 6-10; in this case, n is the total number of test dataset records while y′ and y are the predicted and measured values, respectively. The values of R^2 would range from 0 to 1 – the closer the value is to 1, the higher fitting optimisation of the model is. The values MAE,

MSE, and RMSE are used to evaluate model quality – the smaller the value, the lesser the difference between the predicted value and the measured value, that is, the best the prediction of the model.

$$R^2 = \left[\frac{n\sum y.\dot{y} - (\sum y)(\sum \dot{y})}{\sqrt{n(\sum y^2) - (\sum y)^2}\sqrt{n(\sum \dot{y}^2) - (\sum \dot{y})^2}}\right]^2 \quad (6)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{i=n}|y - \dot{y}| \quad (8)$$

$$MSE = \frac{1}{n}\sum_{i=1}^{i=n}|y - \dot{y}|^2 \quad (9)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{i=n}|y - \dot{y}|^2} \quad (10)$$

4 Results and Discussion

4.1 Initial Modelling

Initial modelling was undertaken using the default hyper-parameter setting in the XGBoost algorithm. For this research, five hyperparameters were selected for optimisation. Table 3 summarises all five hyperparameters and the default value used on the initial XGBoost modelling.

Table 3 Summary of Hyper-Parameter for Optimisation

Hyper - Parameter	**Default Value**
colsample_bytree	1.0
learning_rate	0.3
max_depth	6
min_child_weight	1

subsample	1.0

The initial modelling reached a score of 0.95 and 0.88 for the training and testing models. The coefficient of determination, $R^{2,}$ reached 0.88, and other evaluation metrics for the initial modelling are listed in Table 4. Figure 12 and 13 illustrates the distribution of predicted results compared to actual results and the best fit line for the prediction distribution. Based on the evaluation metrics and prediction results, the initial modelling indicates a fair modelling result; however, it can be further optimised using Grid Search, Random Search, or Bayesian Optimisation Models, which is discussed in the following section.

Table 4 Evaluation Metric of Initial Modelling

Evaluation Metrics	Reference Model
Training Score	0.95
Test Score	0.88
R^2	0.88
MSE	28.75
RMSE	5.36
MAE	4.02

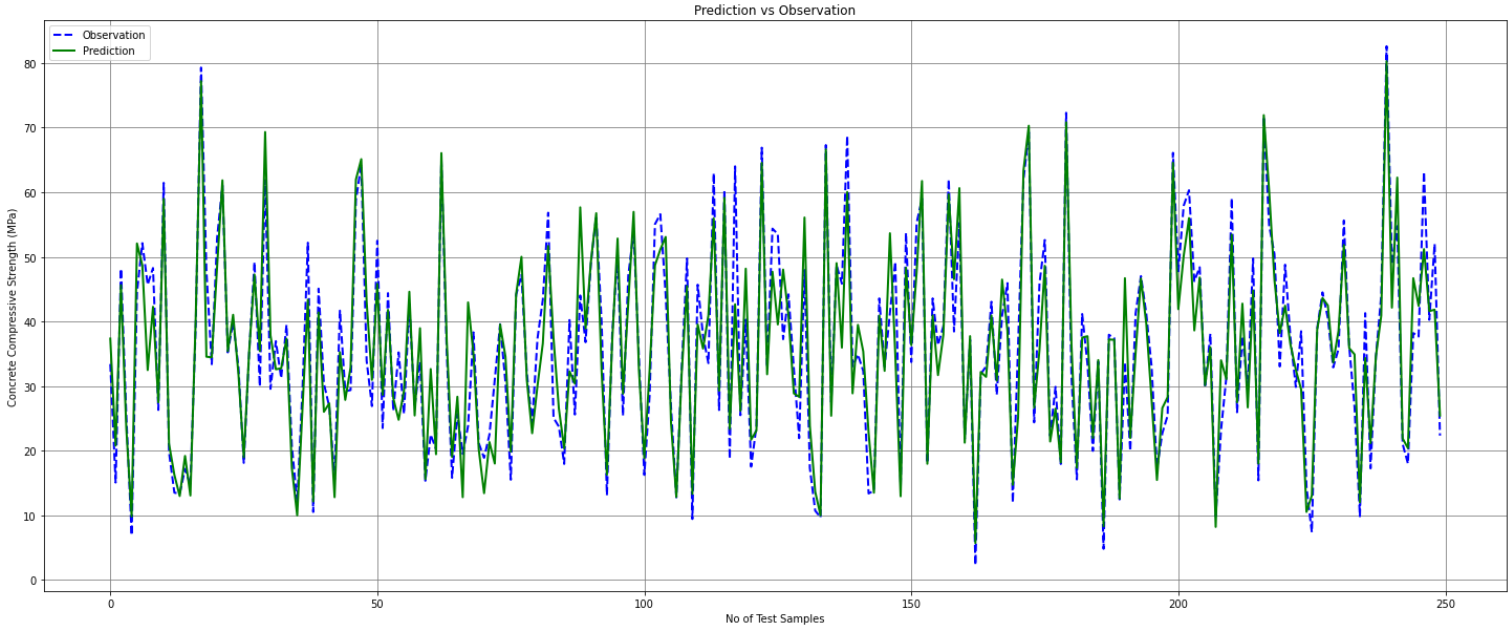

Figure 12 Prediction and Actual Result Comparison (Initial Modelling)

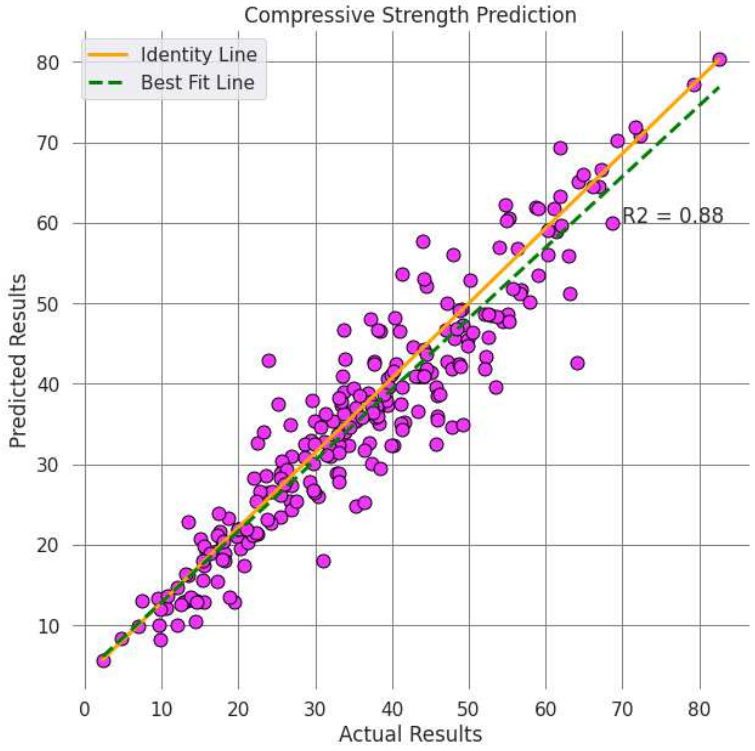

Figure 13 Best Fit Line for Prediction Distribution (Initial Modelling)

4.2 Comparison of Grid Search, Random Search, and Bayesian Optimisation Models

Table 5 below shows the summary of hyper-parameter optimised value for the proposed Grid Search, Random Search, and Bayesian Optimisation Models. The initial model was used as the reference model, and all three optimisation algorithms were performed to optimise the five default hyper-parameter to reach the best RMSE value. As indicated in Table 5, each optimisation algorithms returns a unique value for all five hyper-parameters. Moreover, the optimisation duration was also recorded for each optimisation algorithm. The grid search optimisation shows a significantly longer optimisation duration than Random Search and Bayesian Optimisation, which is recorded around 1 hour.

Table 5 Summary of Hyper-Parameter Optimised Values

Optimisation Parameter	**Optimised Value**			
	Reference Model (Default Value)	Grid Search	Random Search	Bayesian Optimisation
colsample_bytree	1.0	0.8	0.8	0.5
learning_rate	0.3	0.2	0.3	0.5
max_depth	6	4	5	3
min_child_weight	1	1	5	3
subsample	1.0	0.7	0.7	0.8
Optimisation Duration, s	NA	3,696	2.5	19.0

Based on the optimised hyper-parameter values in Table 5, the model is evaluated using the evaluation metrics listed in Table 6. All the optimisation algorithms reached the maximum score of 0.99 for model training. In comparison to all three optimisation models, Random

Search demonstrates the best prediction result. The Grid Search and Bayesian Optimisation model reached almost identical prediction results with only a marginal difference. Overall, the Random Search model has achieved the best R2, MSE, RMSE, MAE values, and fastest optimisation execution compared to Grid Search and Bayesian Optimisation method.

Table 6 Summary of Evaluation Metric for Optimised Models

Evaluation Metrics	**Optimised Value**			
	Reference Model	Grid Search	Random Search	Bayesian Optimisation
Training Score	0.95	0.99	0.99	0.99
Test Score	0.88	0.90	0.91	0.90
R^2	0.88	0.90	0.91	0.90
MSE	28.75	24.36	21.64	24.32
RMSE	5.36	4.94	4.65	4.93
MAE	4.02	3.40	3.11	3.39

Figure 14-19 illustrates the prediction results compared to actual results for Grid Search, Random Search, and Bayesian Optimisation Models. All the optimisation method indicates a similar prediction pattern and fairly distributed along the identity line. However, Random Search shows a higher model fitting and lower prediction errors with an R^2 value of 0.91. In contrast, the Grid Search and Bayesian Optimisation model shows an identical R^2 value of 0.90 and marginally higher error in prediction as the MAE value reached 3.40 and 3.39, respectively.

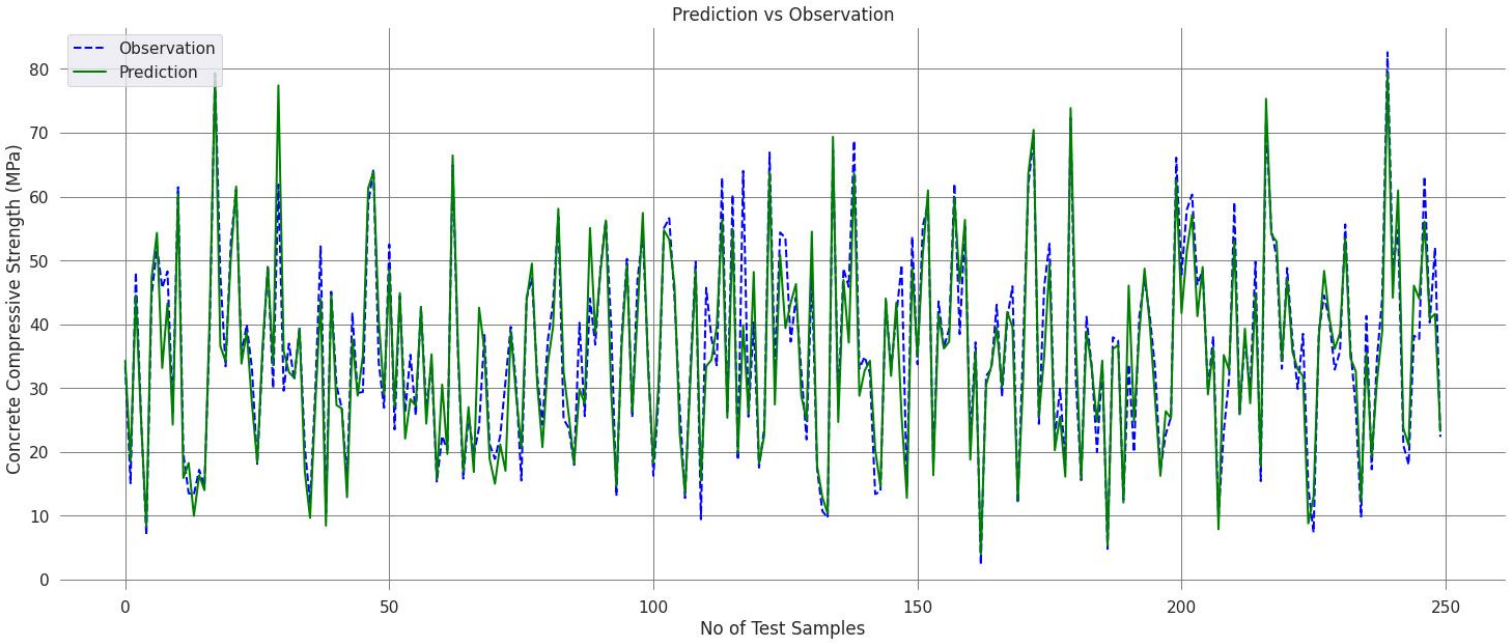

Figure 14 Prediction and Actual Result Comparison (Grid Search)

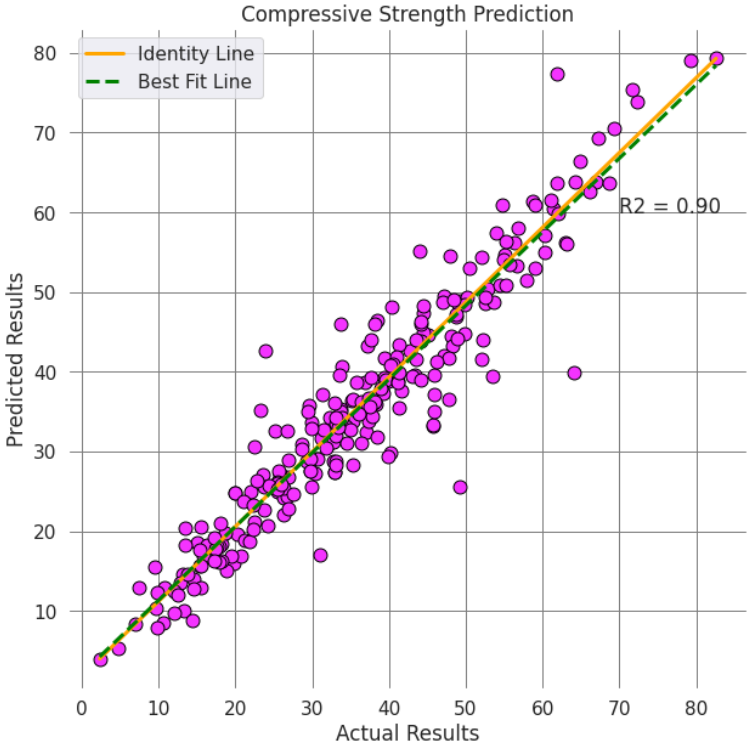

Figure 15 Best Fit Line for Prediction Distribution (Grid Search)

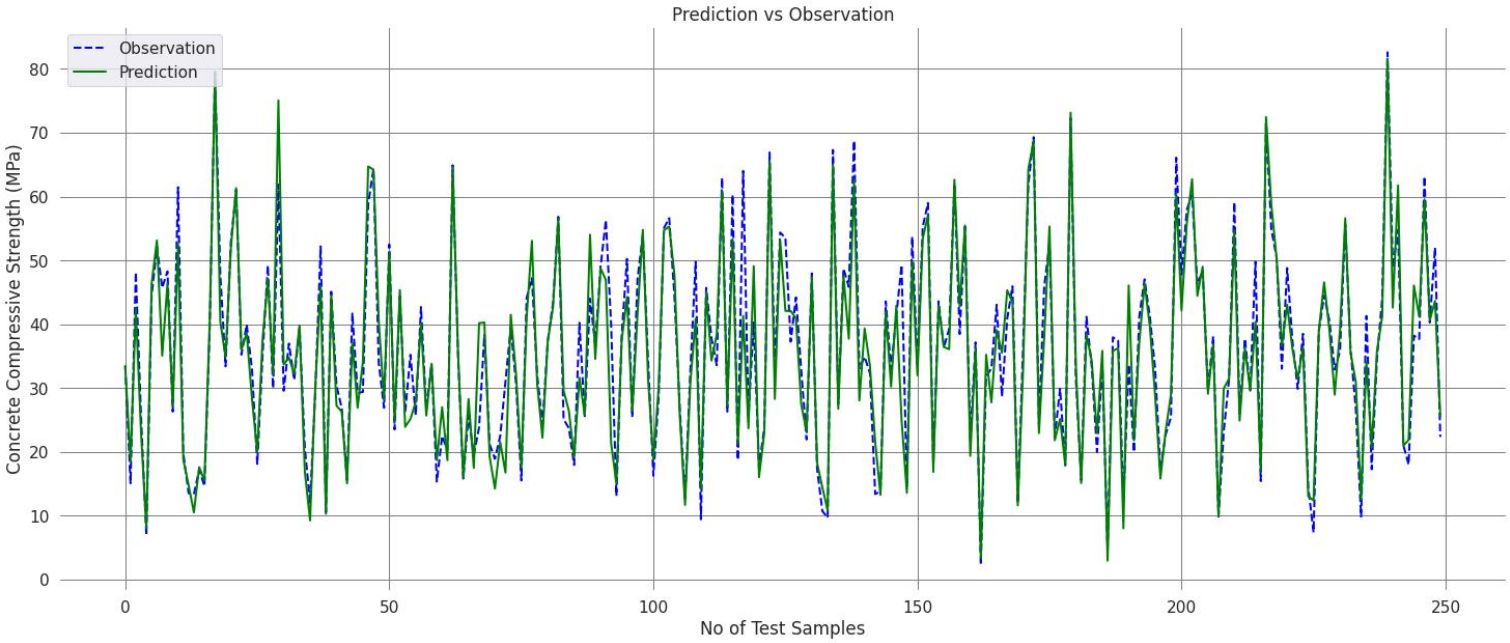

Figure 16 Prediction and Actual Result Comparison (Random Search)

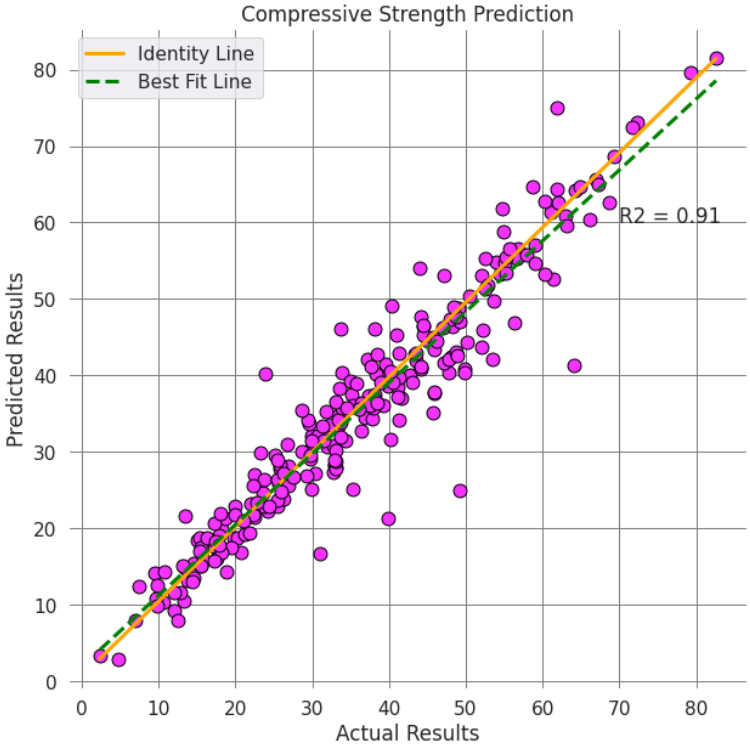

Figure 17 Best Fit Line for Prediction Distribution (Random Search)

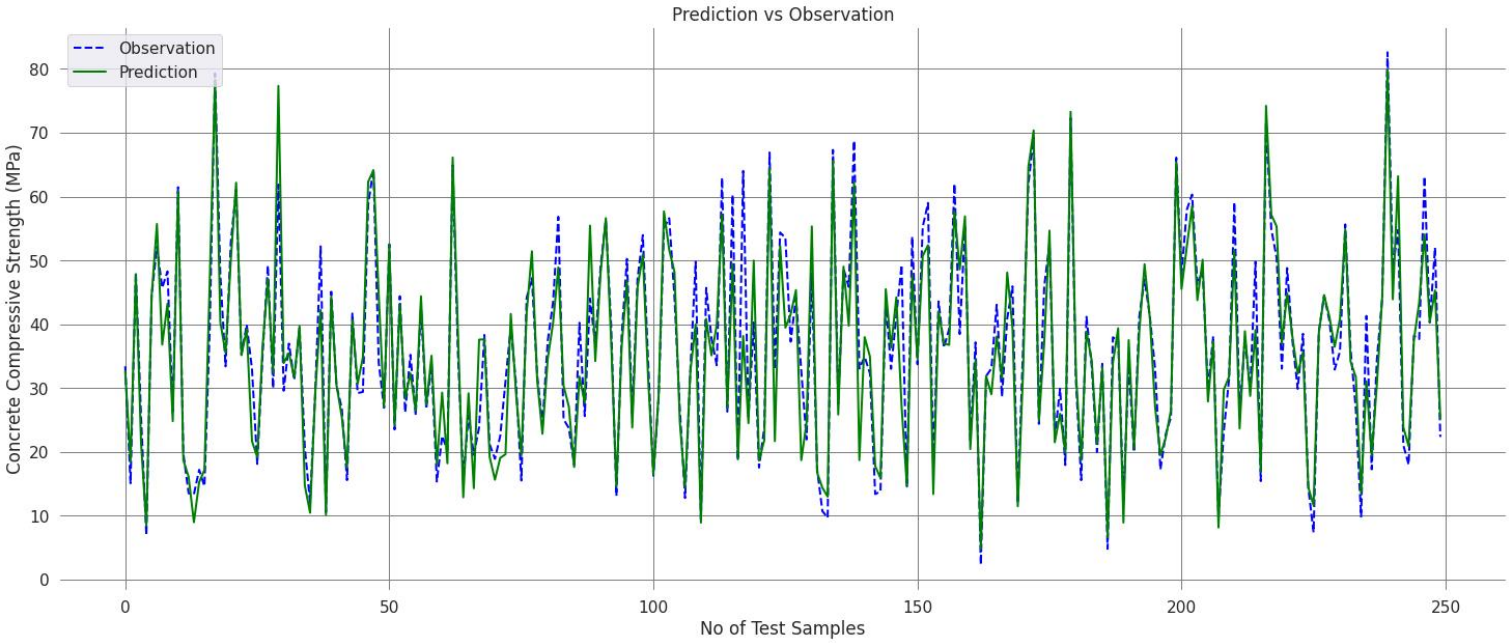

Figure 18 Prediction and Actual Result Comparison (Bayesian Optimisation)

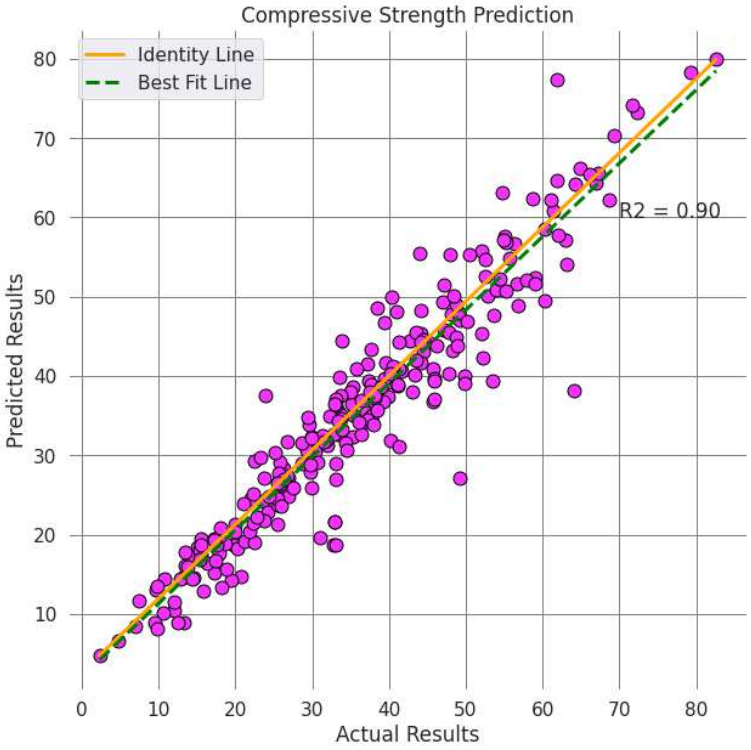

Figure 19 Best Fit Line for Prediction Distribution (Bayesian Optimisation)

4.3 Features Importance Analysis (Sensitivity Analysis)

In addition to model optimisation, sensitivity analysis or feature importance analysis is performed to understand the influence of each feature/component of concrete in predicting the compressive strength of concrete. Figure 20 below displays all the features used in the compressive strength prediction model and the relative importance. Age is expressed as the most essential feature and followed by OPC as the second most important feature. Aggregate is recorded as the least important feature and has a notably low effect on strength prediction.

Furthermore, a separate model was used to evaluate the influence of the least important feature in the model. In the new model, the modelling was performed using all input parameters, excluding 'Aggregate'. The optimised XGBoost with Random Search was used to model the new model. Finally, the model was compared with the original model that contains all features, and the findings are summarised in Table 7. Generally, a marginal improvement was recorded in the evaluation metric compared to the original Random Search Model. This indicates that the model is considered stable and able to perform with a few input parameters.

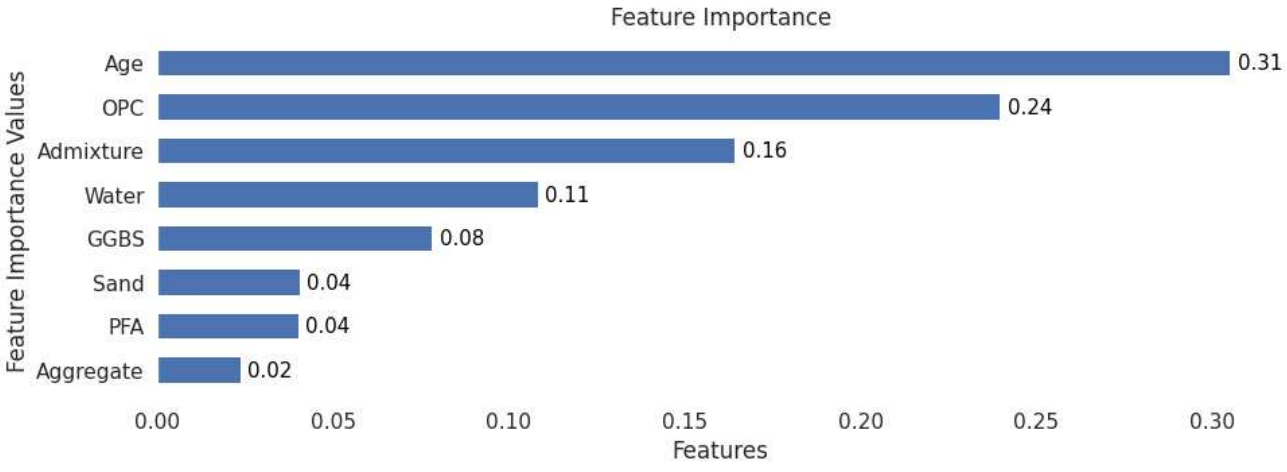

Figure 20 Features Importance for Concrete Strength Prediction Model

Table 7 Summary of Evaluation Metric for Optimised Models

Evaluation Metrics	Value
Training Score	0.99
Test Score	0.92
R^2	0.92
MSE	19.16
RMSE	4.38
MAE	3.01

4.4 Uncertainty Analysis

As the final step of the analysis, a new dataset was feed into the model to assess the performance and behaviour of the model. A separate 30 batches of the dataset were used to predict using the final model. The evaluation metric is listed in Table 8. The distribution of predicted results compared to actual results and the best fit line for the prediction distribution is shown in Figure 21 and Figure 22, respectively. The final model with the new dataset shows acceptable prediction results and comparatively with better evaluation metrics. The prediction distribution and the best fit line also indicate an improved prediction with R^2 reached 0.95.

Table 8 Summary of Evaluation Metric for Final Model with New Data

Evaluation Metrics	Value
R^2	0.95
MSE	6.80
RMSE	2.61
MAE	1.72

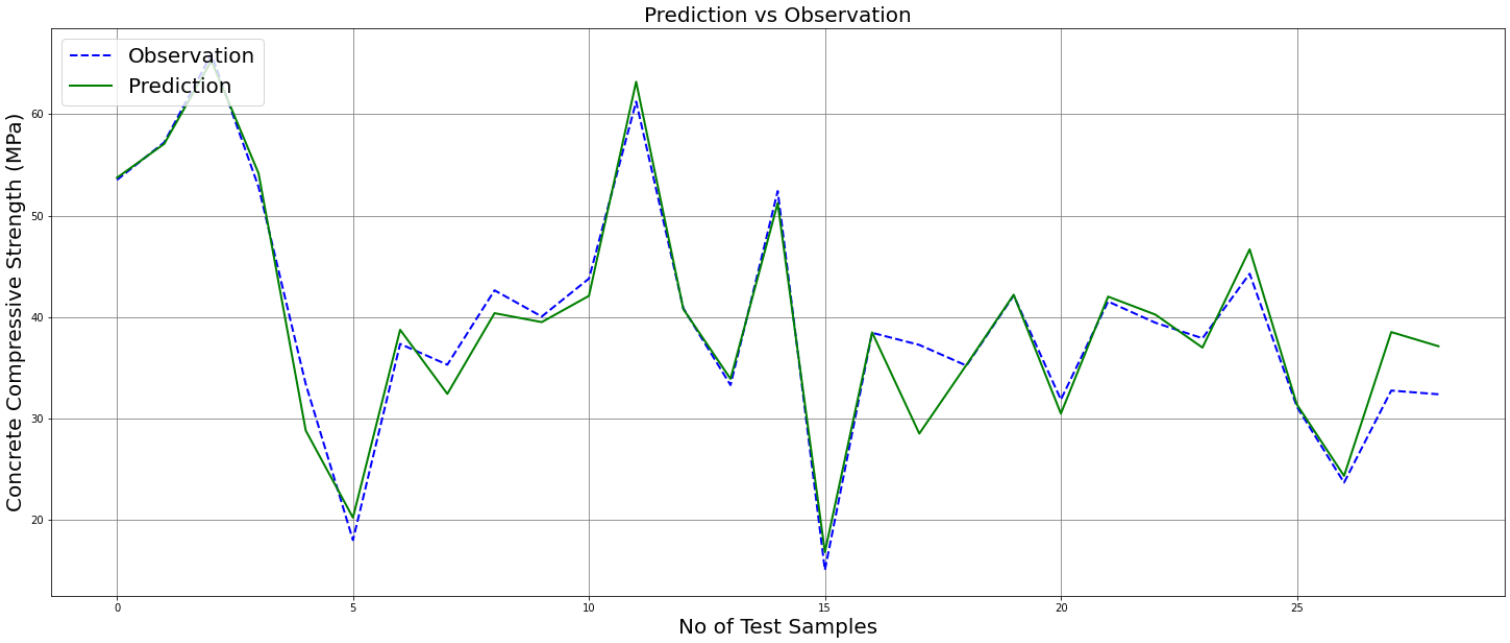

Figure 21 Prediction and Actual Result Comparison (Final Model with New Data)

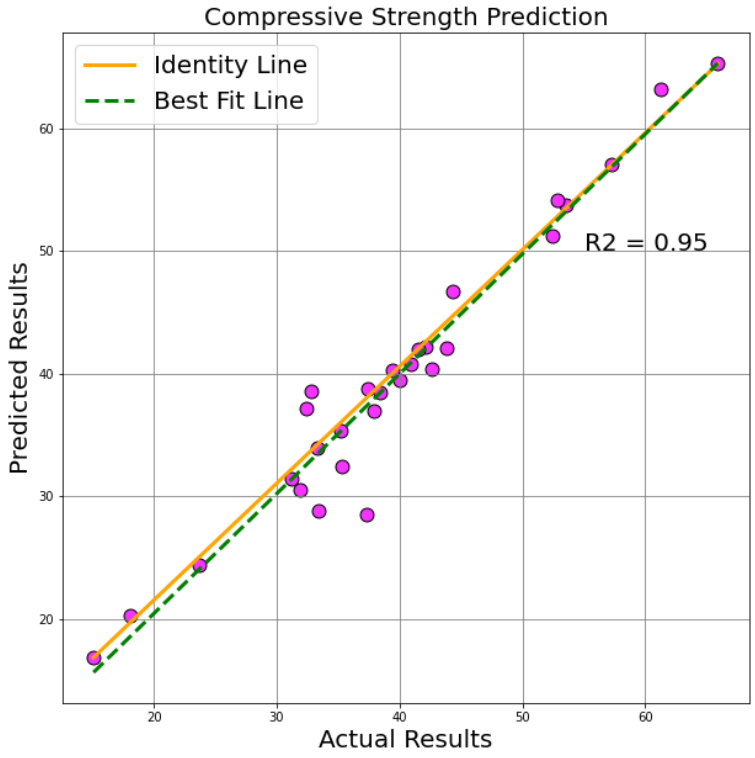

Figure 22 Prediction and Actual Result Comparison (Final Model with New Data)

4.5 Comparison Between Previous Studies and Various ML Algorithms

Furthermore, this paper's results are also compared to previous studies that used the same dataset for research. In the previous research, several combinations, i.e., ensemble model, stacking model, and fusion of deep learning and machine learning algorithms, i.e., ANN, SVM, CART, CHAID, MFA, and LR [23][24][25] was used by the author. The comparison results of the evaluation metrics, i.e., R^2, RMSE, and MAE of previous studies and this research, are summarised in Table 9. Most of the models reported in the earlier studies performed well in fitting the model or high coefficient of determination, R^2; however, no improvement was noticed for modelling errors, i.e., RMSE and MAE values.

Comparably, the XGBoost model optimised with Random Search surpassed all previous research in terms of modelling errors, i.e., RMSE and MAE. In this paper, the RMSE and MAE were improved significantly using the hyper-parameter optimisation algorithms. RMSE and MAE are an essential indicator that measures the prediction accuracy of the model using the new dataset, and at the same time to fit the primary purpose of a predictive model.

Table 9 Summary of Comparison Between Previous Studies

Description / Reference	**Evaluation Metrics**		
	R2	RMSE	MAE
ANN [23]	0.93	6.33	4.42
SVMs [23]	0.92	6.91	4.76
ANNs + CHAID [23]	0.92	7.03	4.67
ANNs + SVMs [23]	0.94	6.17	4.24
CHAID + SVMs [23]	0.93	6.69	4.58
ANNs + SVMs + CHAID [23]	0.94	6.23	4.28

MFA + ANN [24]	0.95	4.85	3.41
SVM [25]	-	5.59	3.75
CART + SVM + LR [25]	-	5.08	3.52
XGBoost *	0.88	5.36	4.02
XGBoost (Random Search) *	0.91	4.65	3.11

* Results from this paper

Additionally, various machine learning algorithms were also modelled and compared to optimised XGBoost with Random Search algorithm. Table 10 shows the summary of multiple ML models and the evaluation metrics for predicting the compressive strength of concrete. Generally, ten models were developed and compared to XGBoost, and only the 'Extra Trees Regressor' model achieved suggestively good prediction results with R^2 of 0.91 and RMSE of 5.41.

Table 10 Summary of Comparison Between Previous Studies

ML Model	**Evaluation Metrics**		
	R^2	RMSE	MAE
Extra Trees Regressor	0.91	5.41	3.41
Random Forest Regressor	0.89	5.45	3.84
Decision Tree Regressor	0.82	6.94	4.66
Linear Regression	0.81	7.17	5.55
Ridge Regression	0.81	7.18	5.58
Bayesian Ridge	0.81	7.22	5.56
Least Angle Regression	0.64	9.55	7.19
Lasso Regression	0.64	10.01	7.96
Huber Regressor	0.61	10.35	8.12

K Neighbours Regressor	0.48	11.97	9.46
XGBoost *	0.88	5.36	4.02
XGBoost (Random Search) *	0.91	4.65	3.11

* Results from this paper

5.0 Conclusion and Recommendation

In conclusion, the XGBoost model with hyperparameter optimisation using Random Search notably improved the model prediction accuracy and reduced the modelling errors. It also showed that the optimised XGBoost algorithms surpassed all other machine learning models reported in previous studies. Nevertheless, the model can be further enhanced by optimising other hyper-parameter in the XGBoost algorithm. For future research, different boosting machine learning algorithms, i.e., CatBoost, Light BGM, AdaBoost, and BGM, can be evaluated to understand the model's performance compared to the XGBoost algorithm.

6.0 Dataset Contribution Statement

Original Dataset Owner and Donor - I-Cheng Yeh

http://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength

7.0 Reference

[1] K. T. Nguyen, Q. D. Nguyen, T. A. Le, J. Shin, and K. Lee, "Analysing the compressive strength of green fly ash based geopolymer concrete using experiment and machine learning approaches," *Constr. Build. Mater.*, vol. 247, p. 118581, 2020, doi: 10.1016/j.conbuildmat.2020.118581.

[2] 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.*, vol. 115, no. October 2020, p. 103863, 2021, doi: 10.1016/j.cemconcomp.2020.103863.

[3] F. H. Chiew, "Prediction of blast furnace slag concrete compressive strength using artificial neural networks and multiple regression analysis," *Proc. - 2019 Int. Conf. Comput. Drone Appl. IConDA 2019*, pp. 54–58, 2019, doi: 10.1109/IConDA47345.2019.9034920.

[4] M. C. Kang, D. Y. Yoo, and R. Gupta, "Machine learning-based prediction for compressive and flexural strengths of steel fiber-reinforced concrete," *Constr. Build. Mater.*, vol. 266, p. 121117, 2021, doi: 10.1016/j.conbuildmat.2020.121117.

[5] T. Han, A. Siddique, K. Khayat, J. Huang, and A. Kumar, "An ensemble machine learning approach for prediction and optimisation of modulus of elasticity of recycled aggregate concrete," *Constr. Build. Mater.*, vol. 244, p. 118271, 2020, doi: 10.1016/j.conbuildmat.2020.118271.

[6] P. Singh and P. Khaskil, "Prediction of compressive strength of green concrete with admixtures using neural networks," *2020 IEEE Int. Conf. Comput. Power Commun. Technol. GUCON 2020*, no. cm, pp. 714–717, 2020, doi: 10.1109/GUCON48875.2020.9231230.

[7] W. Ben Chaabene, M. Flah, and M. L. Nehdi, “Machine learning prediction of mechanical properties of concrete: Critical review,” *Constr. Build. Mater.*, vol. 260, p. 119889, 2020, doi: 10.1016/j.conbuildmat.2020.119889.

[8] T. Nguyen-Sy, J. Wakim, Q. D. To, M. N. Vu, T. D. Nguyen, and T. T. Nguyen, “Predicting the compressive strength of concrete from its compositions and age using the extreme gradient boosting method,” *Constr. Build. Mater.*, vol. 260, p. 119757, 2020, doi: 10.1016/j.conbuildmat.2020.119757.

[9] Q. Tang, G. Xia, X. Zhang, and F. Long, “A Customer Churn Prediction Model Based on XGBoost and MLP,” *Proc. - 2020 Int. Conf. Comput. Eng. Appl. ICCEA 2020*, pp. 608–612, 2020, doi: 10.1109/ICCEA50009.2020.00133.

[10] W. F. Mustika, H. Murfi, and Y. Widyaningsih, “Analysis Accuracy of XGBoost Model for Multiclass Classification - A Case Study of Applicant Level Risk Prediction for Life Insurance,” *Proceeding - 2019 5th Int. Conf. Sci. Inf. Technol. Embrac. Ind. 4.0 Towar. Innov. Cyber Phys. Syst. ICSITech 2019*, pp. 71–77, 2019, doi: 10.1109/ICSITech46713.2019.8987474.

[11] D. Wu, P. Guo, and P. Wang, “Malware Detection based on Cascading XGBoost and Cost Sensitive,” *Proc. - 2020 Int. Conf. Comput. Commun. Netw. Secur. CCNS 2020*, pp. 201–205, 2020, doi: 10.1109/CCNS50731.2020.00051.

[12] J. Li and R. Zhang, “Dynamic Weighting Multi Factor Stock Selection Strategy Based on XGboost Machine Learning Algorithm,” *Proc. 2018 IEEE Int. Conf. Saf. Prod. Informatiz. IICSPI 2018*, pp. 868–872, 2019, doi: 10.1109/IICSPI.2018.8690416.

[13] Y. Qu, Z. Lin, H. Li, and X. Zhang, “Feature Recognition of Urban Road Traffic Accidents Based on GA-XGBoost in the Context of Big Data,” *IEEE Access*, vol. 7, pp. 170106–170115, 2019, doi: 10.1109/ACCESS.2019.2952655.

[14] H. Xu *et al.*, "Identifying diseases that cause psychological trauma and social avoidance by GCN-Xgboost," *BMC Bioinformatics*, vol. 21, pp. 2–6, 2020, doi: 10.1186/s12859-020-03847-1.

[15] M. Kivrak, E. Guldogan, and C. Colak, "Prediction of death status on the course of treatment in SARS-COV-2 patients with deep learning and machine learning methods," *Comput. Methods Programs Biomed.*, vol. 201, p. 105951, 2021, doi: 10.1016/j.cmpb.2021.105951.

[16] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, vol. 13-17-Augu, pp. 785–794, 2016, doi: 10.1145/2939672.2939785.

[17] H. Chen, Z. Liu, K. Cai, L. Xu, and A. Chen, "Grid search parametric optimisation for FT-NIR quantitative analysis of solid soluble content in strawberry samples," *Vib. Spectrosc.*, vol. 94, pp. 7–15, 2018, doi: 10.1016/j.vibspec.2017.10.006.

[18] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimisation," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.

[19] R. P. A. Jasper.S, Hugo.L, "Practical Bayesian Optimisation of Machine Learning Algorithms," *Relig. Arts*, vol. 17, no. 1–2, pp. 57–73, 2013, doi: 10.1163/15685292-12341254.

[20] I. C. Yeh, "Modeling of strength of high-performance concrete using artificial neural networks," *Cem. Concr. Res.*, vol. 28, no. 12, pp. 1797–1808, 1998, doi: 10.1016/S0008-8846(98)00165-3.

[21] H. Nguyen, T. Vu, T. P. Vo, and H. T. Thai, "Efficient machine learning models for prediction of concrete strengths," *Constr. Build. Mater.*, vol. 266, p. 120950, 2021,

doi: 10.1016/j.conbuildmat.2020.120950.

[22] R. Sen Fan, Y. Li, and T. T. Ma, “Research and Application of Project Settlement Overdue Prediction Based on XGBOOST Intelligent Algorithm,” *iSPEC 2019 - 2019 IEEE Sustain. Power Energy Conf. Grid Mod. Energy Revolution, Proc.*, pp. 1213–1216, 2019, doi: 10.1109/iSPEC48194.2019.8975056.

[23] J. S. Chou and A. D. Pham, “Enhanced artificial intelligence for ensemble approach to predicting high performance concrete compressive strength,” *Constr. Build. Mater.*, vol. 49, pp. 554–563, 2013, doi: 10.1016/j.conbuildmat.2013.08.078.

[24] D. K. Bui, T. Nguyen, J. S. Chou, H. Nguyen-Xuan, and T. D. Ngo, “A modified firefly algorithm-artificial neural network expert system for predicting compressive and tensile strength of high-performance concrete,” *Constr. Build. Mater.*, vol. 180, pp. 320–333, 2018, doi: 10.1016/j.conbuildmat.2018.05.201.

[25] J. S. Chou, C. F. Tsai, A. D. Pham, and Y. H. Lu, “Machine learning in concrete strength simulations: Multi-nation data analytics,” *Constr. Build. Mater.*, vol. 73, pp. 771–780, 2014, doi: 10.1016/j.conbuildmat.2014.09.054.

Figures

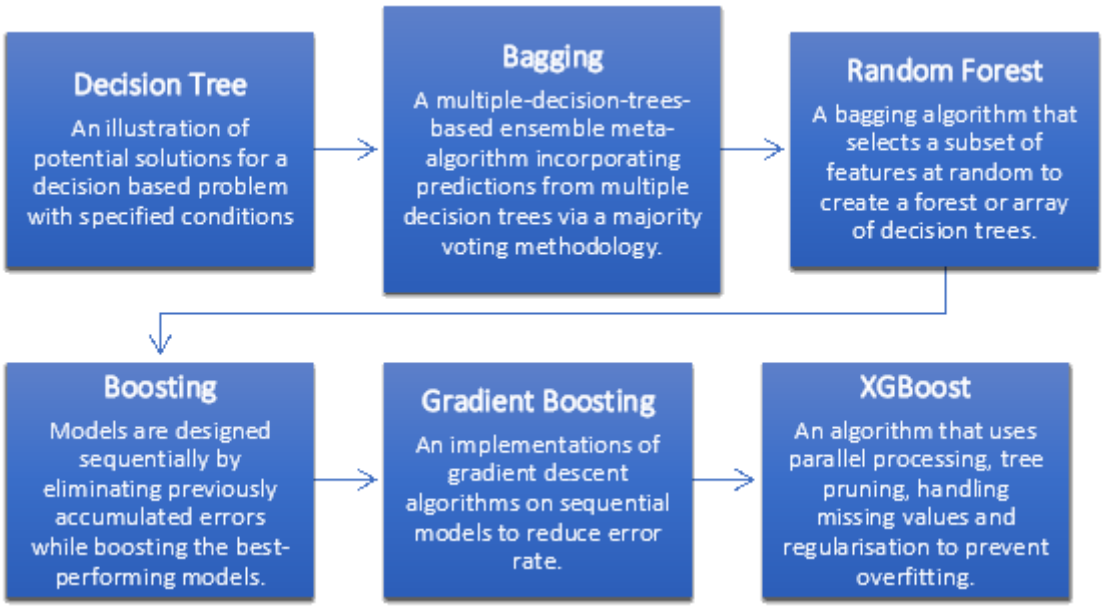

Figure 1

The Evolution of XGBoost

Parallelization	XGBoost builds parallel trees based on sequential implementation. The loop order is reversed using initialisation via a global search and sorting using parallel threads. This increases algorithmic efficiency by allowing parallelization without any overhead.
Tree Pruning	The greedy stopping criterion allows the loss to be below some threshold at the split point. Using a max depth parameter, the XGBoost algorithm begins pruning the trees backwards. This 'depth-first' approach increases computational efficiency.
Hardware Optimization	XGBoost is a cache aware algorithm which allocate internal buffers in each thread to store gradient statistics. In addition, it is enhanced with out-of-core computation when handling large datasets.
Regularization	The approach penalises models that are too complex by applying LASSO and Ridge regularisation to reduce model overfitting.
Sparsity Awareness	XGBoost automatically learns sparse features for inputs by naturally handling different degrees of sparsity in the data, and effectively handles different kinds of sparsity in the training dataset.
Weighted Quantile Sketch	The distributed weighted Quantile Sketch algorithm in XGBoost finds the optimal split points between weighted datasets.
Cross Validation	The built-in cross-validation in XGBoost takes away the need to specify the exact number of boosting iterations needed in a single run.

Figure 2

The Advantages of XGBoost

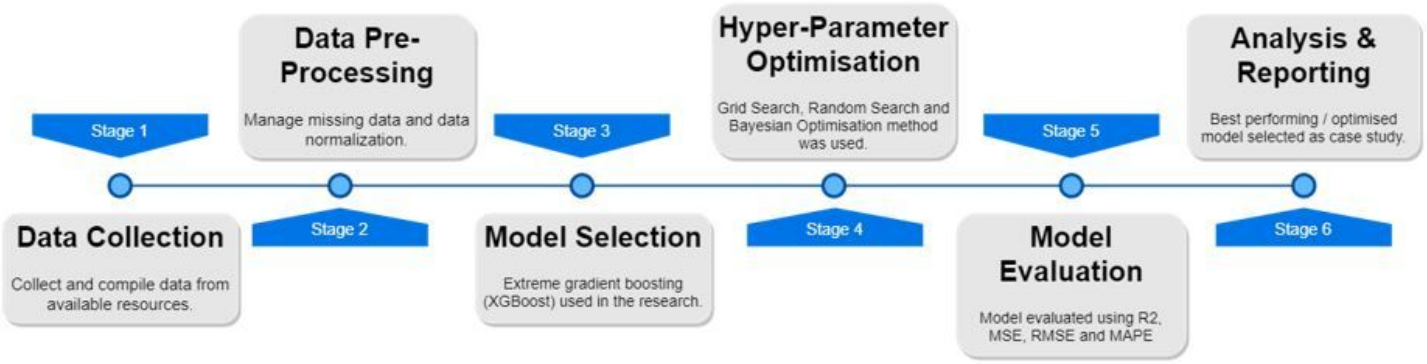

Figure 3

Step by step of XGBoost Modelling and Optimisation

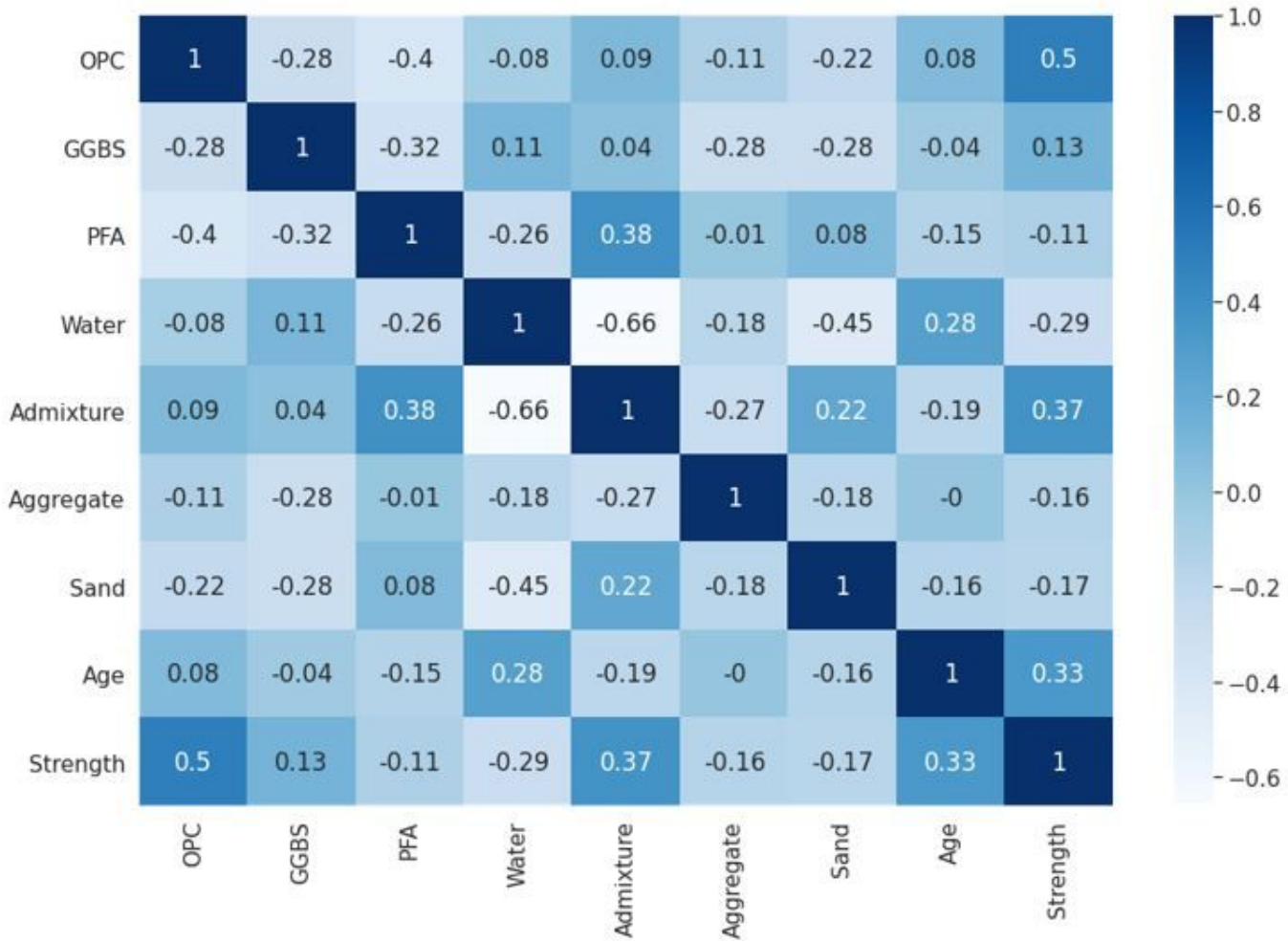

Figure 4

Pearson's Correlation Heatmap

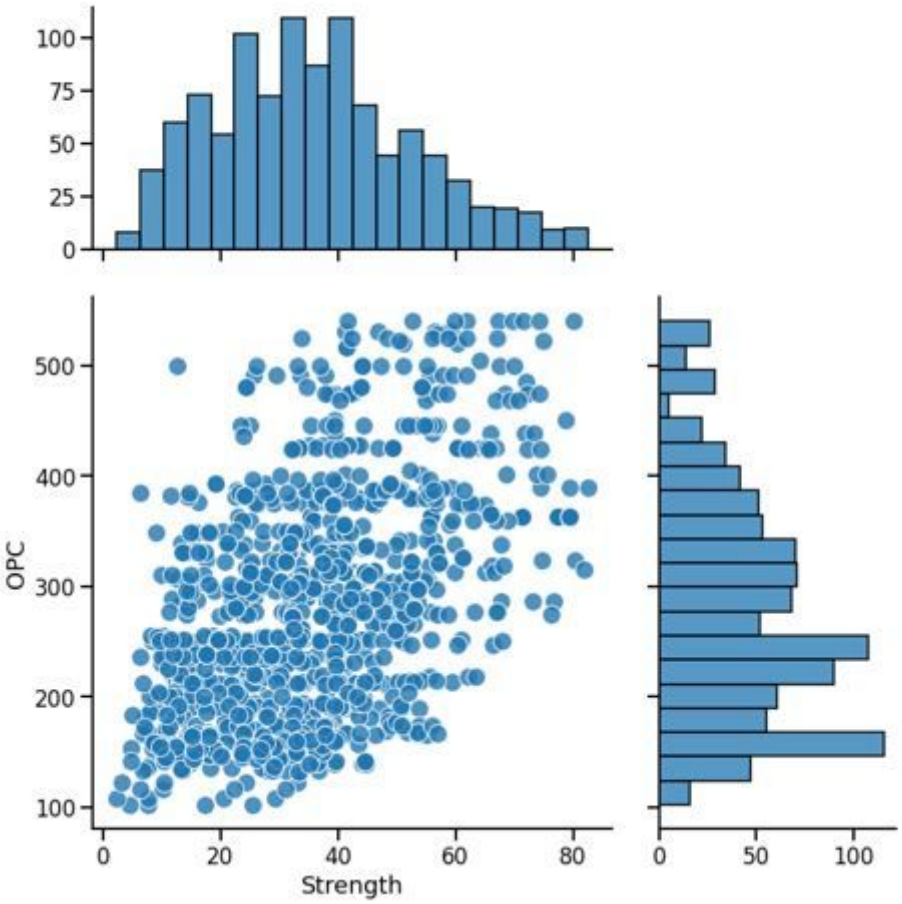

Figure 5

Correlation Between OPC and Strength

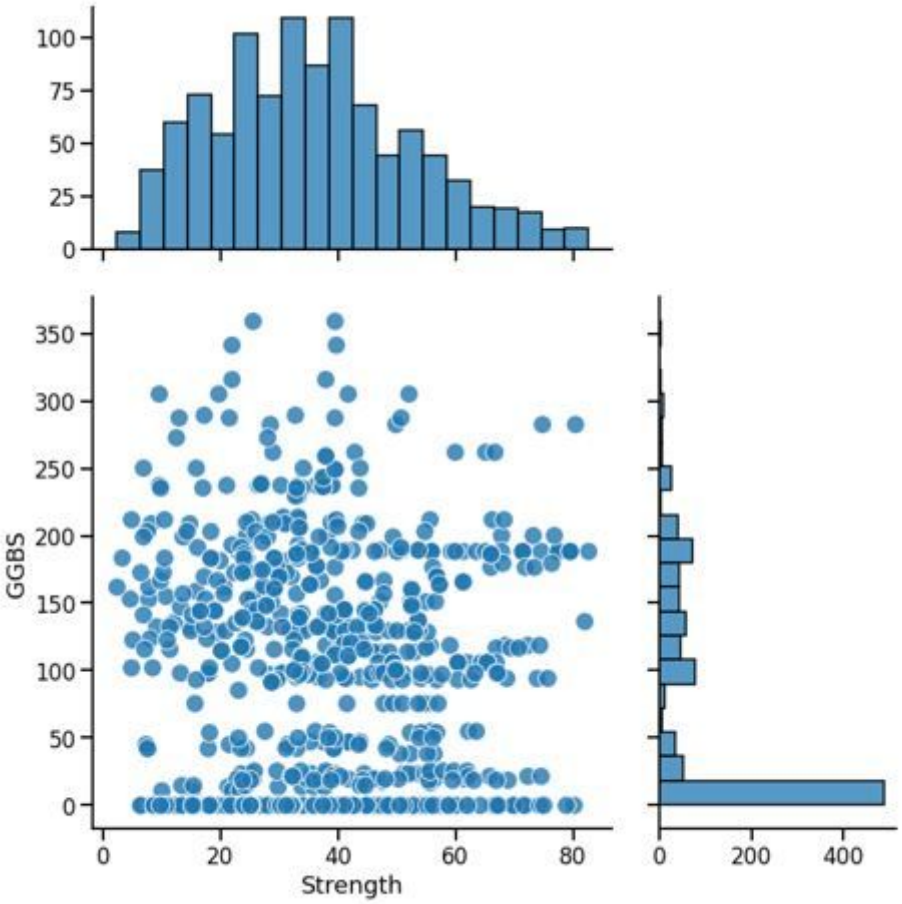

Figure 6

Correlation Between GGBS and Strength

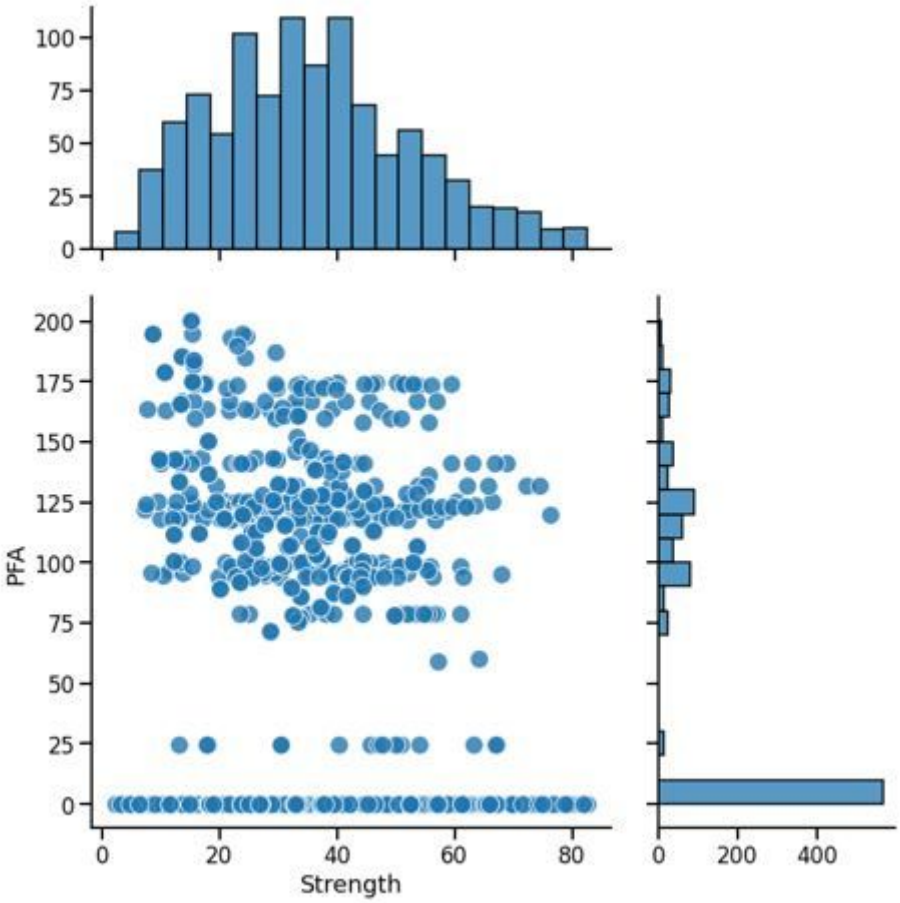

Figure 7

Correlation Between PFA and Strength

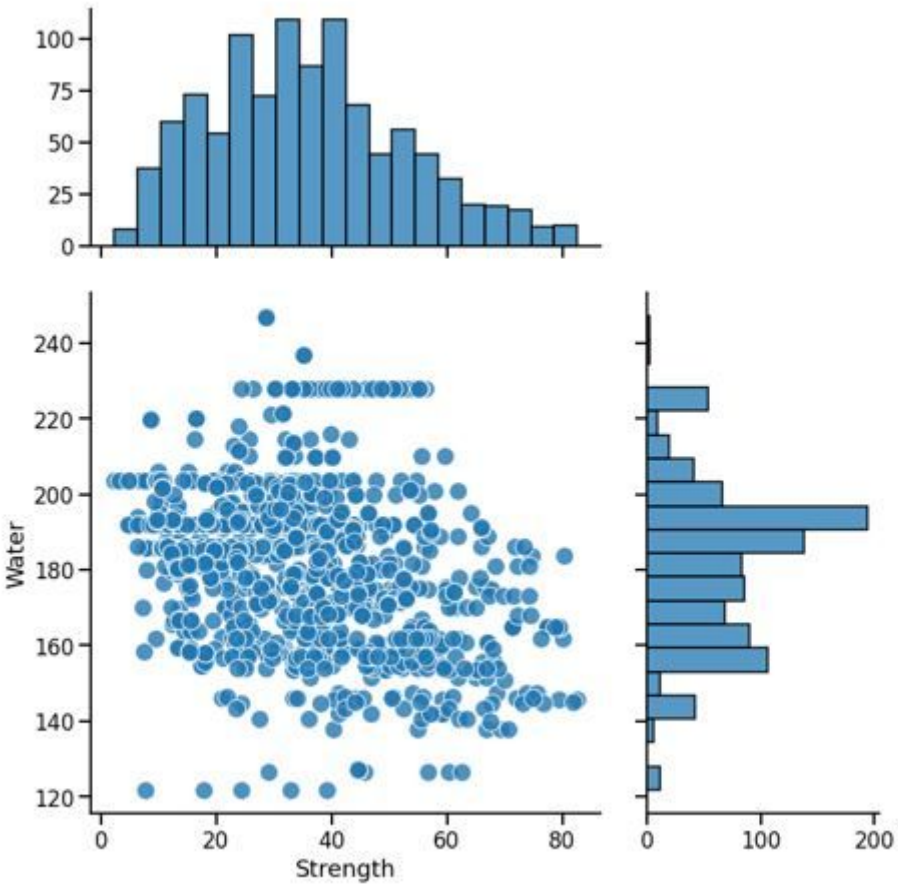

Figure 8

Correlation Between Water and Strength

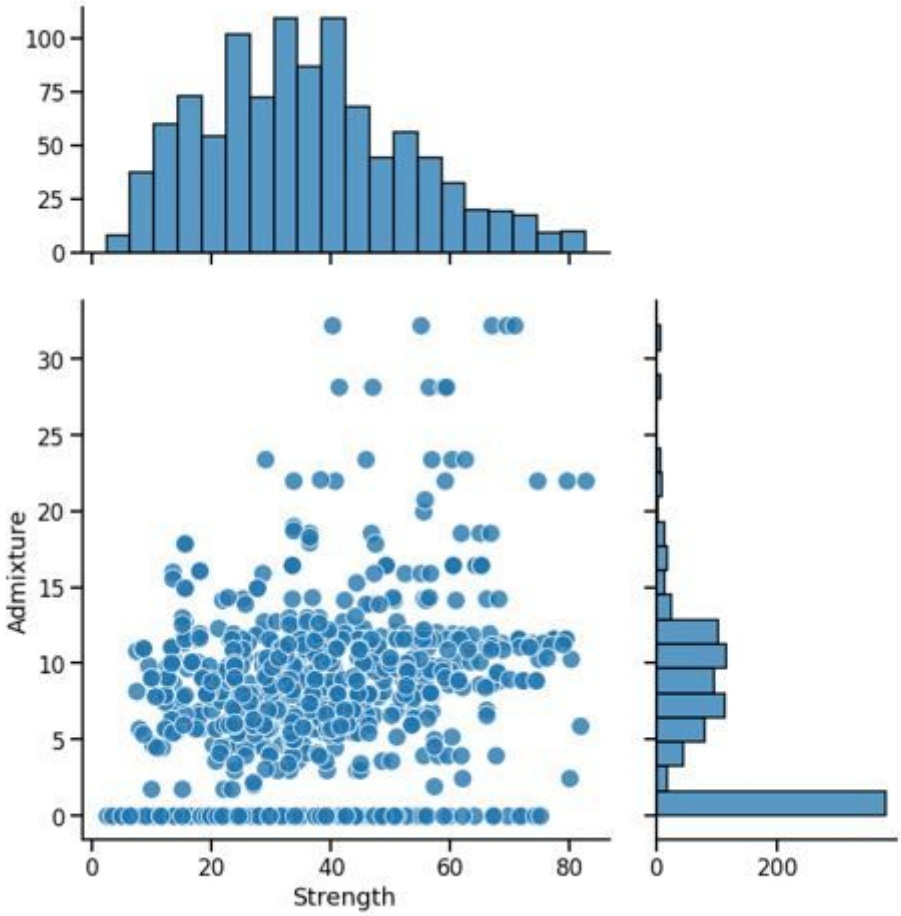

Figure 9

Correlation Between Admixture and Strength

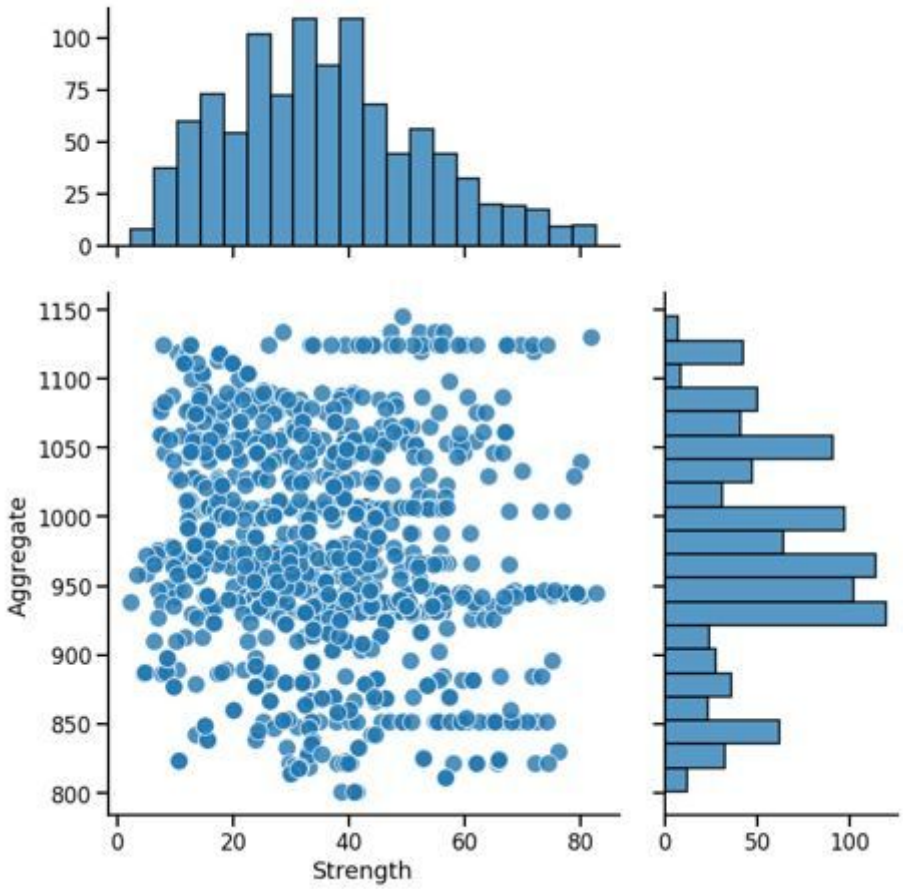

Figure 10

Correlation Between Aggregate and Strength

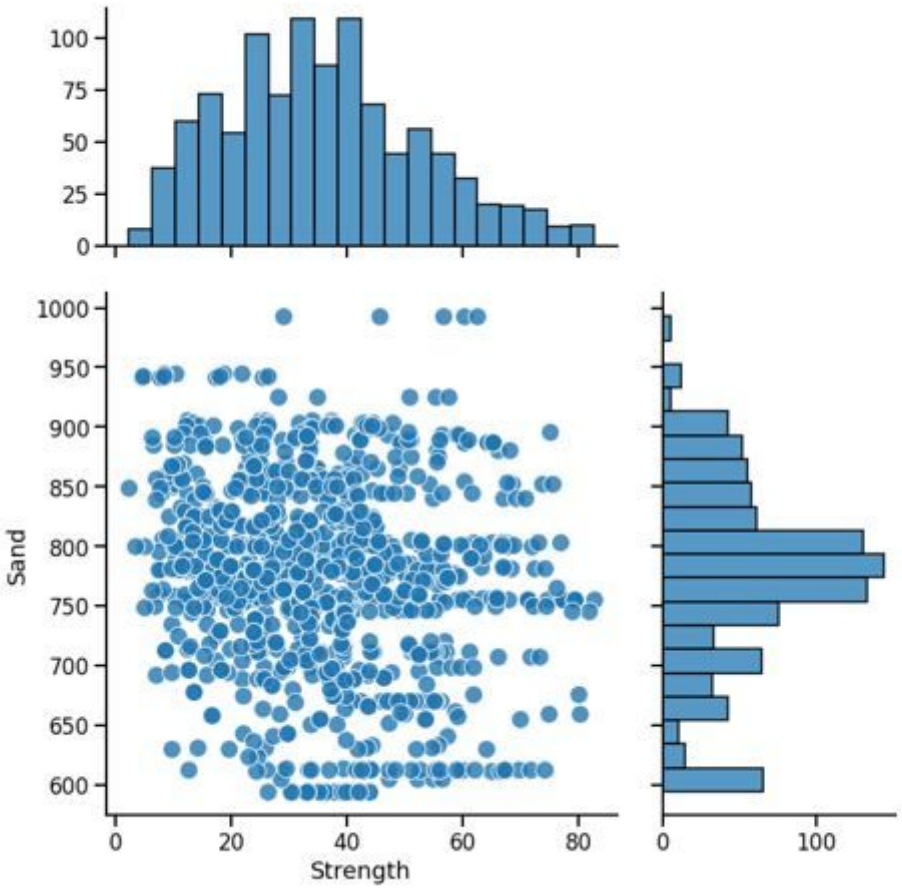

Figure 11

Correlation Between Sand and Strength

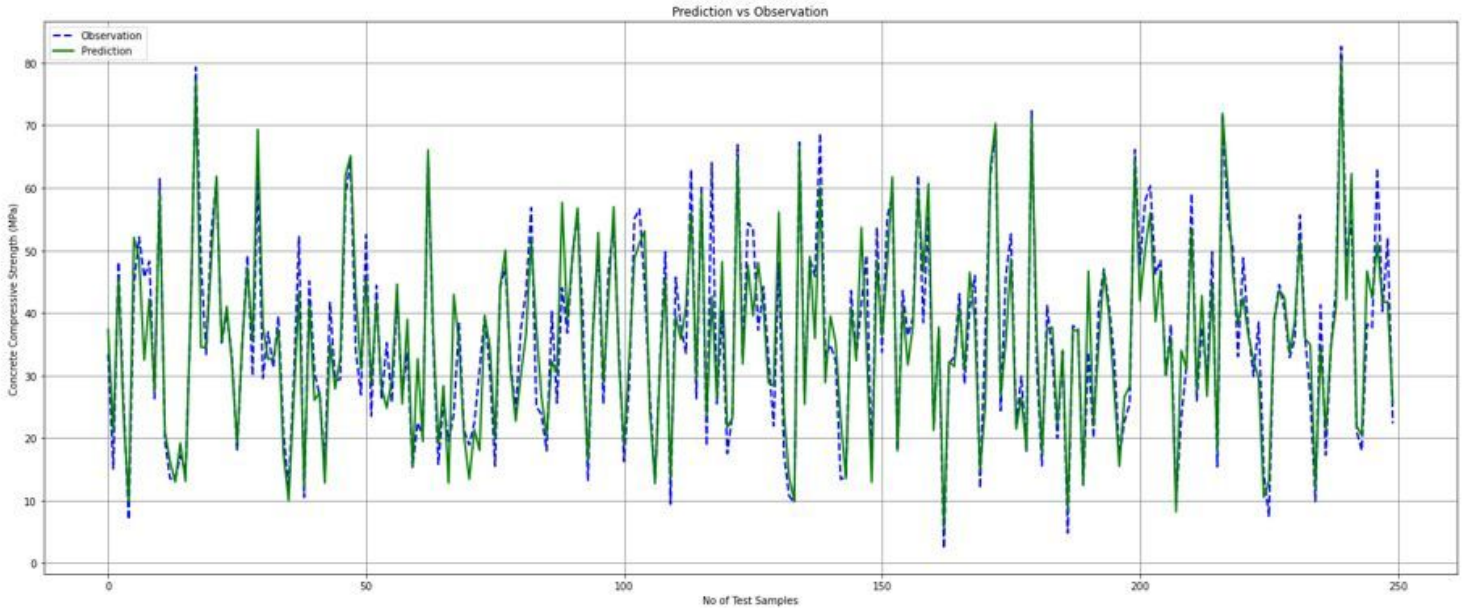

Figure 12

Prediction and Actual Result Comparison (Initial Modelling)

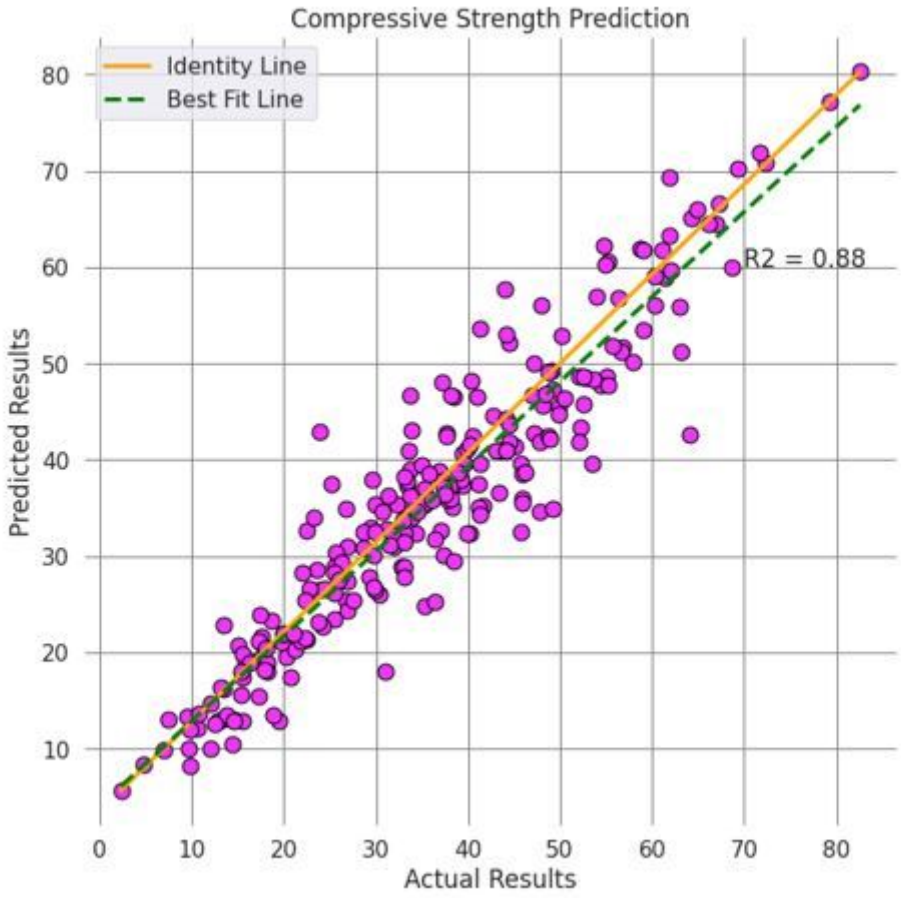

Figure 13

Best Fit Line for Prediction Distribution (Initial Modelling)

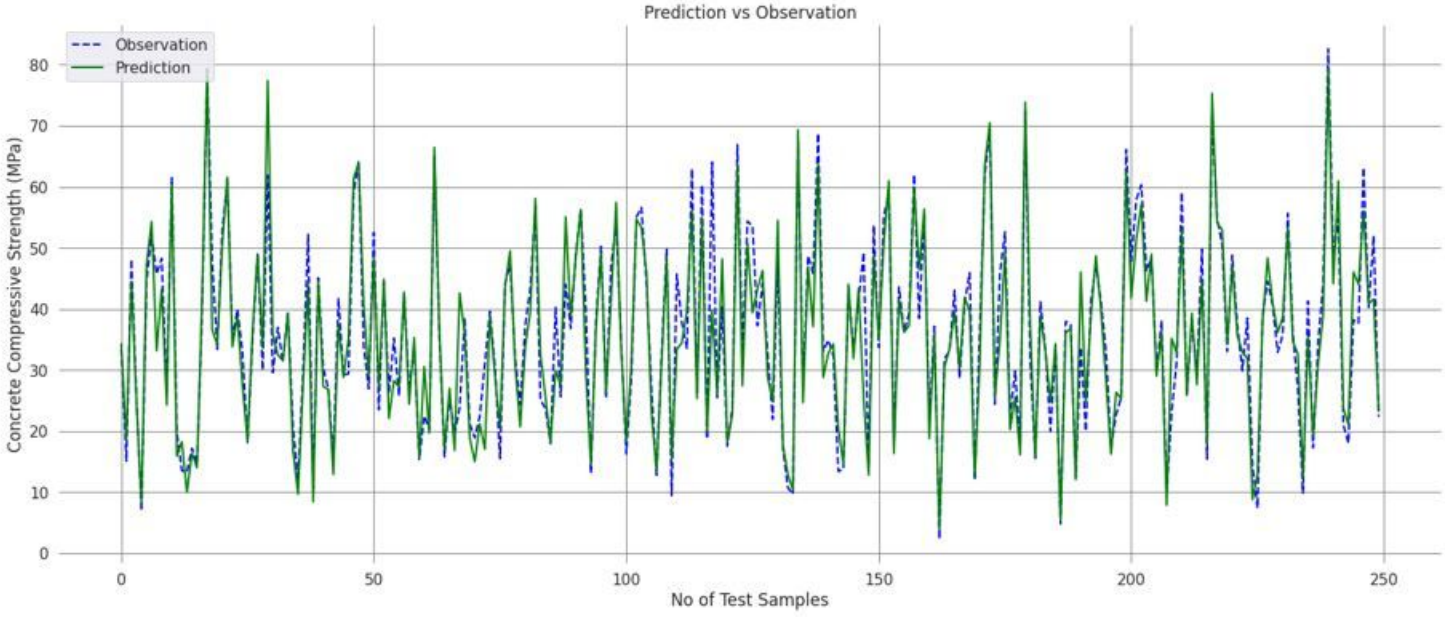

Figure 14

Prediction and Actual Result Comparison (Grid Search)

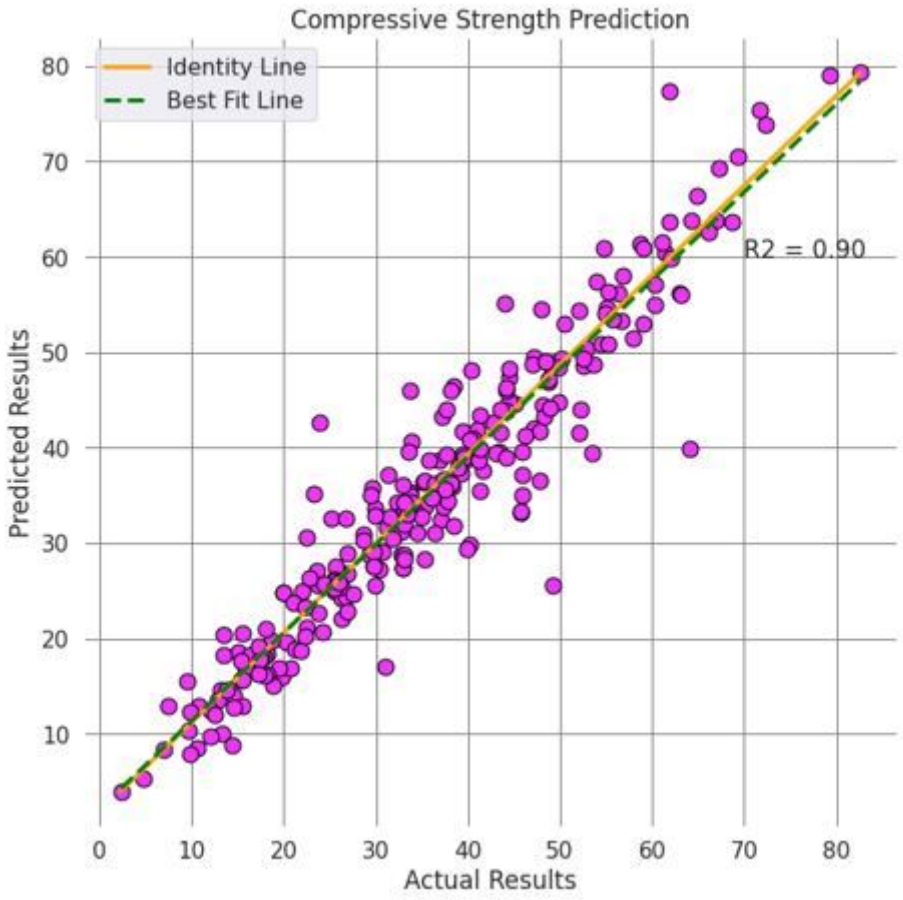

Figure 15

Best Fit Line for Prediction Distribution (Grid Search)

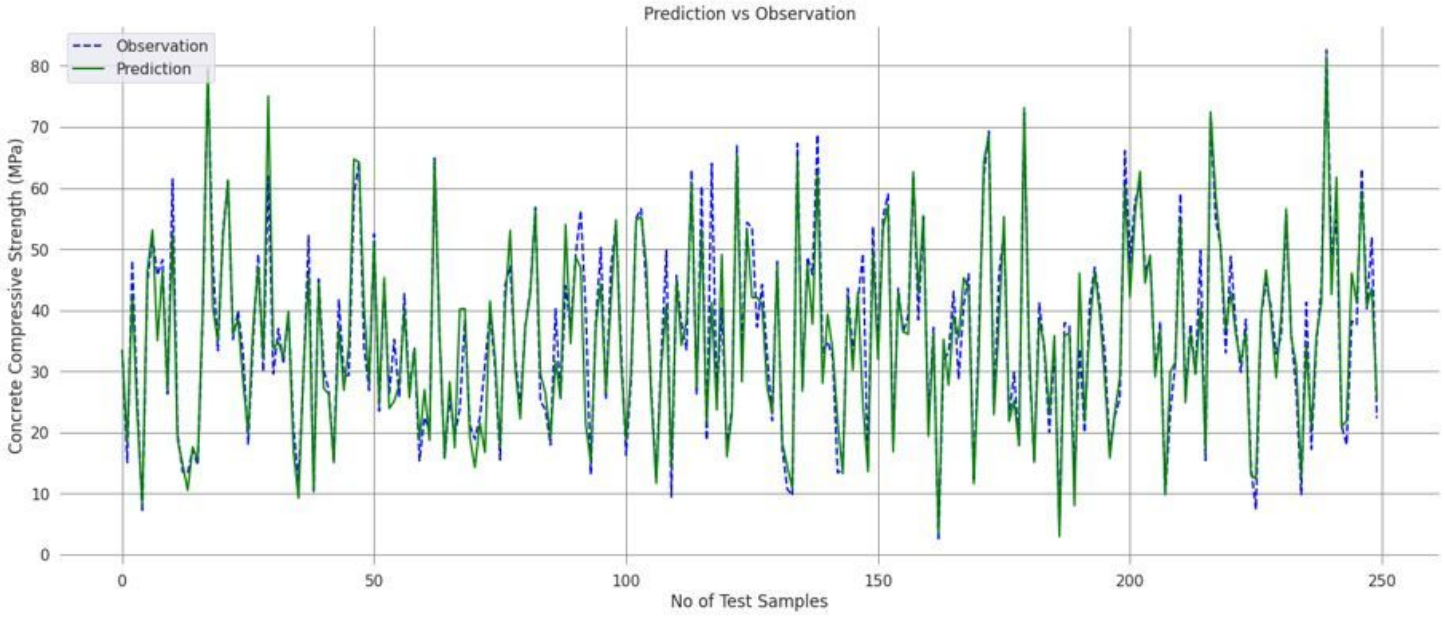

Figure 16

Prediction and Actual Result Comparison (Random Search)

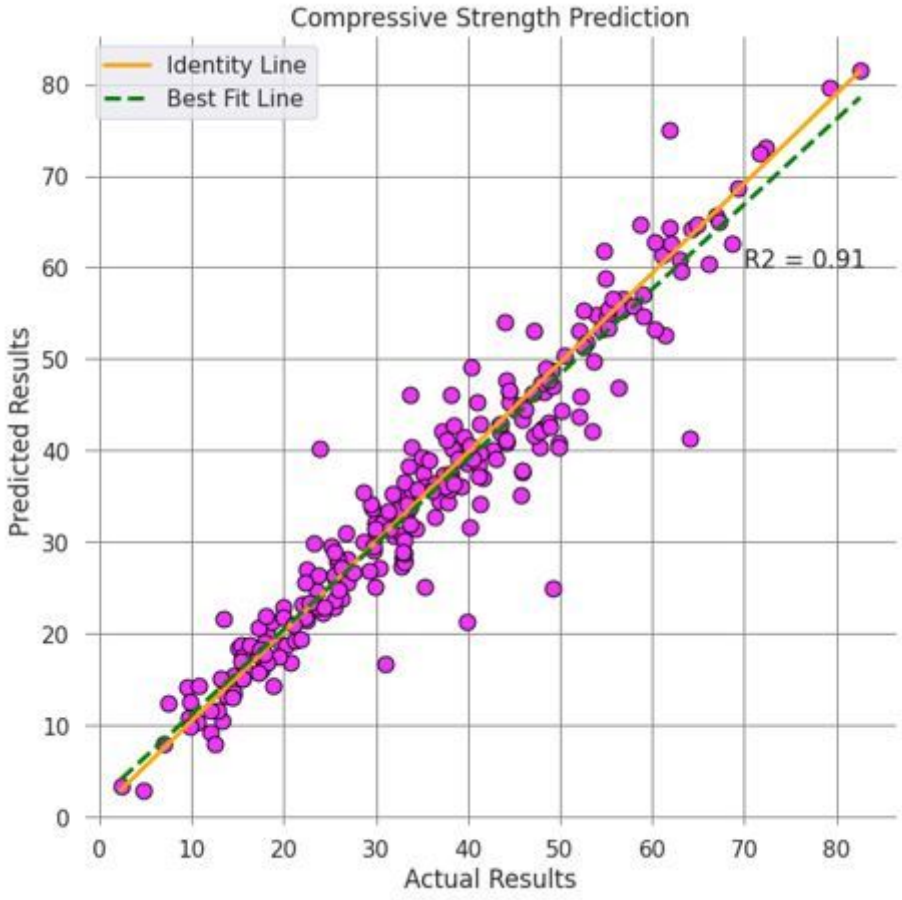

Figure 17

Best Fit Line for Prediction Distribution (Random Search)

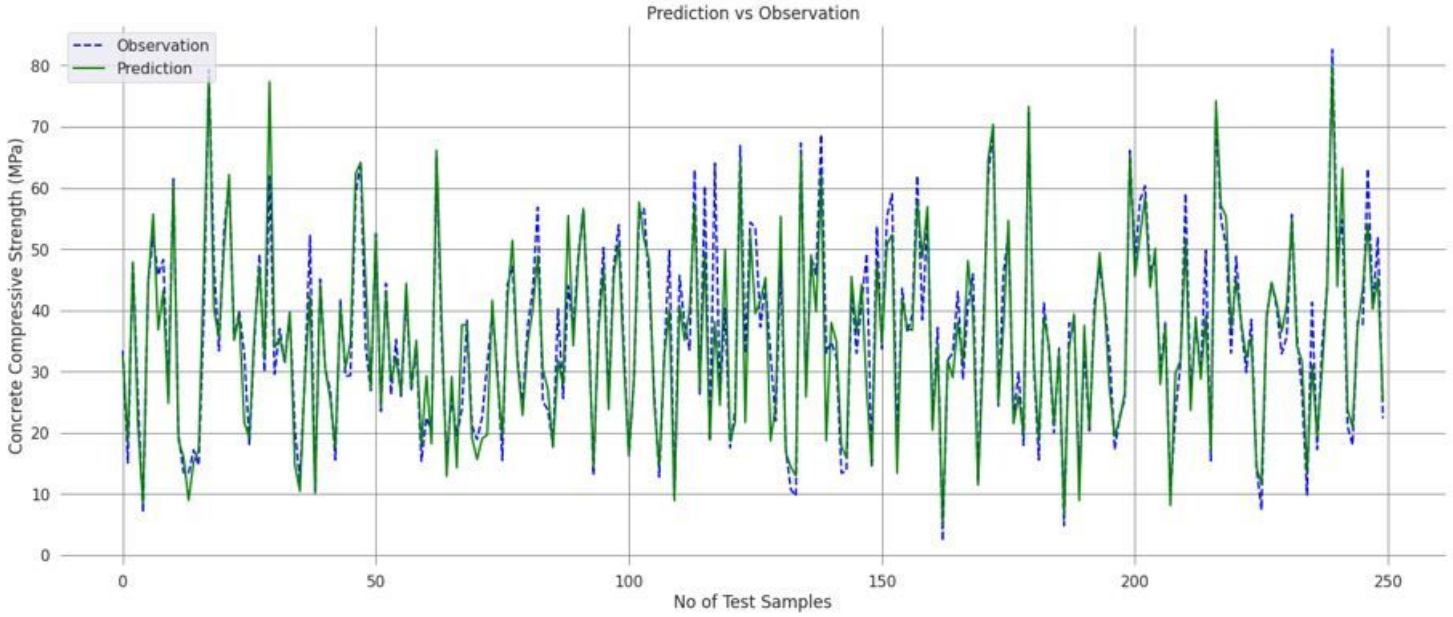

Figure 18

Prediction and Actual Result Comparison (Bayesian Optimisation)

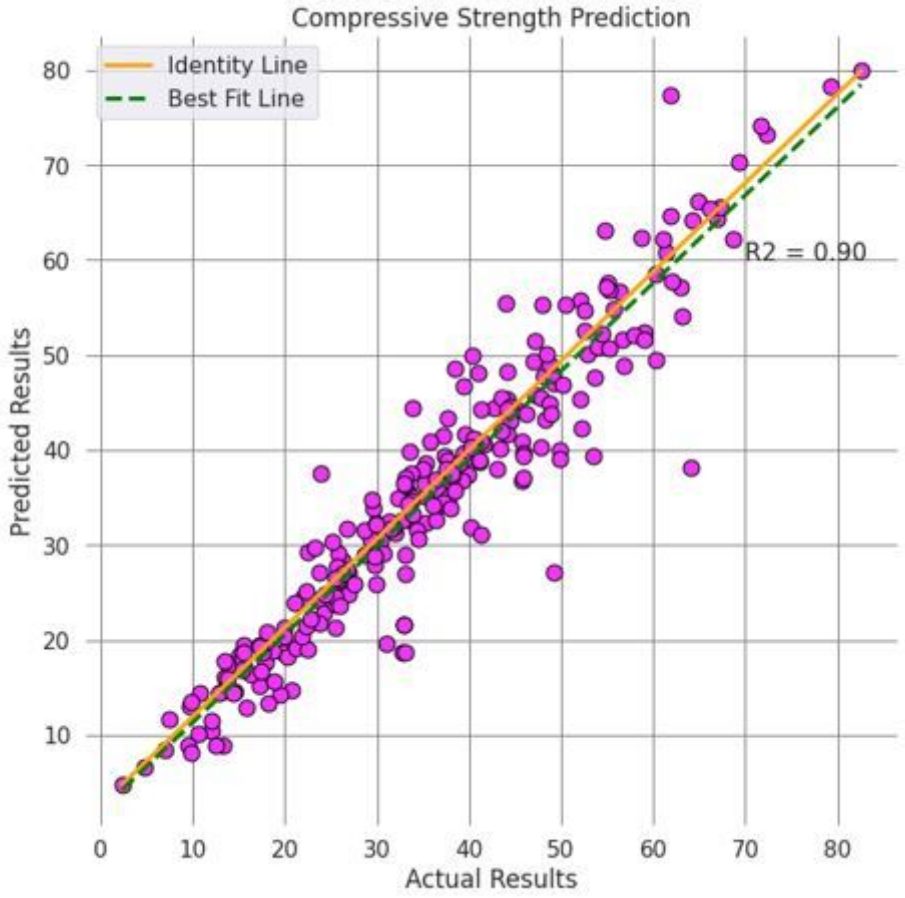

Figure 19

Best Fit Line for Prediction Distribution (Bayesian Optimisation)

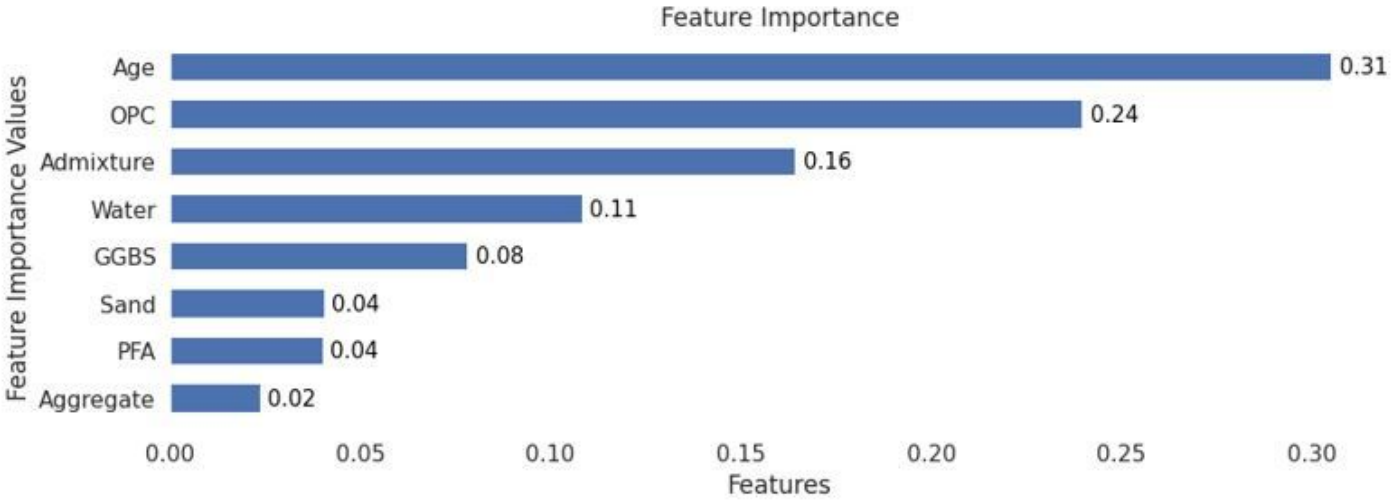

Figure 20

Features Importance for Concrete Strength Prediction Model

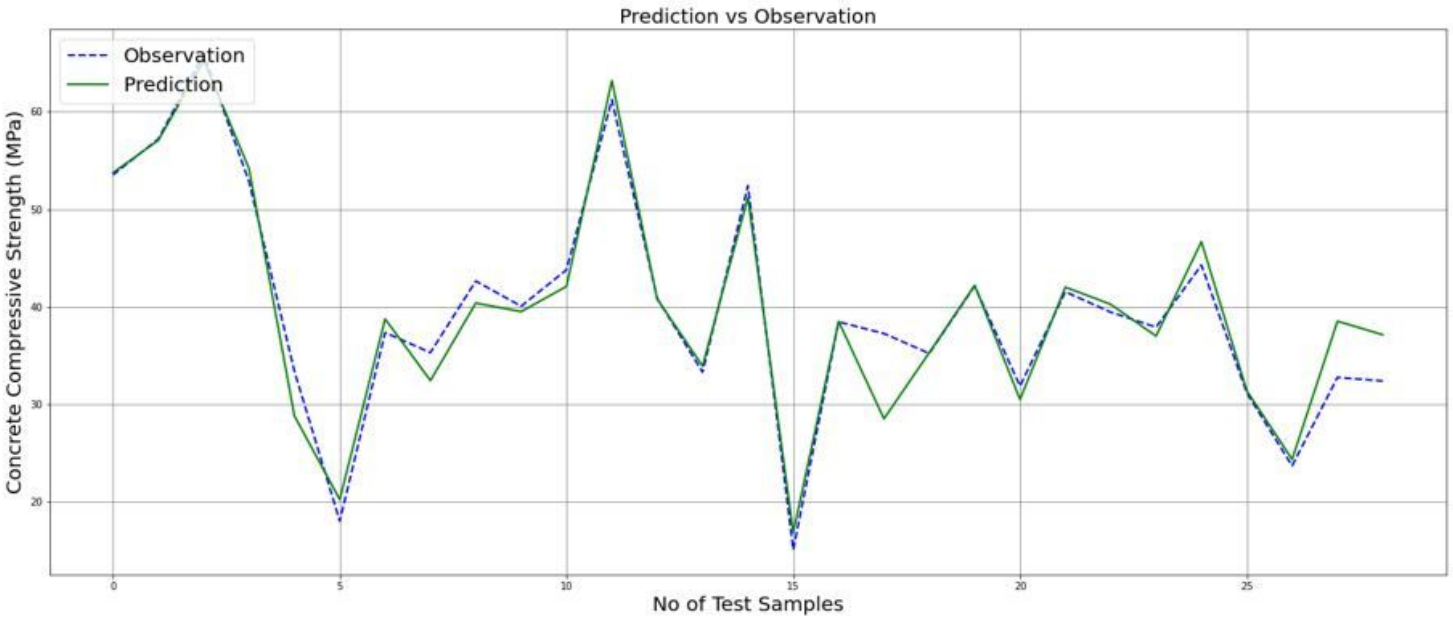

Figure 21

Prediction and Actual Result Comparison (Final Model with New Data)

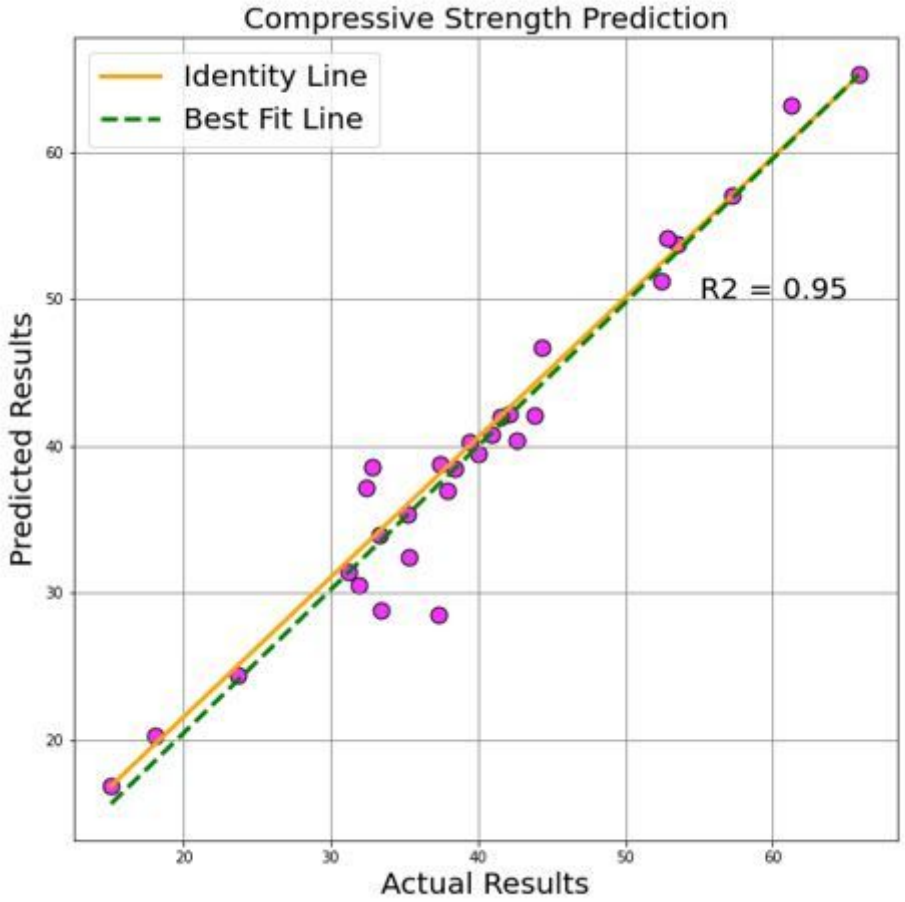

Figure 22

Prediction and Actual Result Comparison (Final Model with New Data)