Compressive Strength Predictions for Concrete using Various Machine Learning Techniques

Abstract: Machine Learning techniques have been popularly used for prediction and statistical optimization. Due to the lack of any empirical relation for the compressive strength of concrete and the emergence of new concrete mixtures, this work has been conducted to acquire reliable Machine Learning models for accurate predictions of concrete mechanical properties. Algorithms ranging from basic linear regressions to multilayered Artificial Neural Network, individual Support Vector Regressions to ensemble methods like voting, stacking, or bagging is used for which train-validation-test split is incorporated to reduce overfitting. This extensive research yields familiarity with the application of regularly used Machine Learning methods, thereby bridging the knowledge gap. The application of each model for predicting compressive strength of concrete with various statistical measures is critiqued. These comparison results deem Random Forest, Support Vector Regressions, Artificial Neural Network, and stacking of Gradient Boosting, Random Forest, and Linear Regression efficient in making the required prediction.

Keywords: *Compressive Strength of Concrete, Machine Learning, Statistical Learning, Regression, Artificial Neural Network, Decision Trees, Support Vector Matrices, Ensemble Methods.*

1. Introduction

Concrete, a mixture of various strengthening and binding components, is the most preliminary material used in the construction process. Thus, knowledge of different mechanical properties for random mixes, especially compressive strength, is necessary for modern construction phases. The compressive strength depends on numerous other factors like the degree of compaction, curing, or weather conditions, to name a few. Inclusion of such plentiful factors manufacture problems like a lack of empirical equations or statistical measures. Additional fine particles like Fly Ash or Blast Furnace Slag increase concrete performance in terms of compressive strength and durability. Thus, a precise prediction of compressive strength can provide structural engineers with vital data leading towards durable and economical designs.

There have been numerous efforts made in predicting the compressive strength of concrete using different Machine Learning (ML) techniques. Linear and non-linear regression analysis are common prediction techniques in the field of statistics. However, getting precise equations from these simple mathematical models is quite difficult [1,2,3,4]. Furthermore, to overcome the drawbacks of conventional methods, advanced individual methods were adopted. Support Vector Regression (SVR) is a hybrid baseline approach, which yields accurate results in a confined space, also known as a black-box approach [5,6,7,8,9]. Along with the development of time-efficient computing systems, the popularity of tree-based models and boosting-based models have increased in the Civil engineering domain [10,11,12,13]. M5P model was used by

Behnood et al. [14] for analysis of compressive, split tensile and flexural strength of concrete. Yan and Shi reported that SVR was better than other considered models for predicting elastic modulus of normal and high strength concrete [15]. The most sophisticated machine learning (ML) technique, Artificial Neural Networks, performs meticulously over substantial datasets and hence, is used widely in material science studies as it gives efficient results. [5,6,16,17,18,19,20]. Ensemble learning methods are claimed to be superior to the individual learning models [6], but the literature review exhibits no extensive studies for comparison of the whole spectrum wide ML algorithms, both individual and ensemble-based.

To conclude, the primary objectives of this study include reviewing well known ML algorithms used for research purposes, comparing the said algorithms, both individual and ensemble-based, and finally develop the best prediction model with the highest R^2 Score which can serve as a baseline model for future studies and predictions of the compressive strength. The R^2 Score range between 0 and 1 depending upon the "goodness of the fit." Models with higher coefficients exhibit better performances.

Nomenclature is included if necessary:

1. LR – Linear Regression
2. kNN – k Nearest Neighbors
3. RF – Random Forest
4. DT – Decision Trees
5. SVM – Support Vector Machine
6. GB – Gradient Boosting
7. SVR – Support Vector Regressions
8. MSE – Mean Squared Error

2. Processing of Dataset

2.1 Description of Dataset

Data for ML Predictions was collected from reliable published studies conducting laboratory experiments [35]. Along with basic mix constituents like cement, water, fine aggregates, and coarse aggregates, various supplementary materials, like Blast Furnace Slag, Fly Ash, and Superplasticizers, are added to enhance the compressive strength and durability property of concrete. The final set of 1030 samples for compressive strength for High-Performance Concrete with varying ages was cured under normal conditions and evaluated from numerous university research labs. All tests were performed on 15-cm cylindrical specimens of concrete prepared using standard procedures. Effects of various elements of concrete and its relation with compressive strength is discussed in further section.

2.1.1 Cement

The compressive strength of concrete significantly changes with an increase in cement content, particularly at an early age; however, other factors like water to cement ratio have a greater effect on strength. Once a particular value for compressive strength is met, the addition of cement will have an insignificant effect on the same [22].

2.1.2 Water

Water content in concrete does have a significant effect on the compressive strength of concrete and the compressive strength decreases with an increase in water content [23].

2.1.3 Aggregates

Though influence of aggregate size has insignificant role on both, normal and high-strength concrete, coarse aggregate size does have a direct relation with compressive strength to a certain extent [24].

2.1.4 Superplasticizer

Effect of dosage of admixture presents different behavior on compressive strength. The increase in dosage until optimal dosage point will increase the compressive strength since the addition of Superplasticizer will give more workability but subsequently, overdosage causes deflocculation of binding particles resulting in a drastic decrease [25].

2.1.5 Blast Furnace Slag

The Blast Furnace Slag has a significant effect on the compressive strength of concrete as the compressive strength of the concrete increases along with an increase in the proportion of Blast Furnace Slag [26].

2.1.6 Fly Ash

The appropriate amount of fly ash depends on the application, as well as the composition and proportions of all the ingredients in the concrete mix, the conditions during placement, particularly the temperature, building procedures such as curing, and the exposure conditions. The general trend follows an increase in compressive strength leading up to this optimal value succeeded by a sharp decrease.

Table 1 shows the minimum and maximum values of input parameters which will be used further for building the models described in this paper. Figure 1 shows the correlation matrix derived from the data set for building the model in the paper.

Table 1. Description of the Dataset [35]

Input Parameters	**Units**	**Minimum**	**Maximum**	**Average**
Cement	kg/m^3	102.00	540.00	281.17
Blast Furnace Slag	kg/m^3	0.00	359.40	73.90
Fly Ash	kg/m^3	0.00	200.10	54.19
Water	kg/m^3	121.75	247.00	181.57
Superplasticizer	kg/m^3	0.00	32.20	6.20
Coarse Aggregate	kg/m^3	801.00	1145.00	972.92
Fine Aggregate	kg/m^3	594.00	992.60	773.58
Age of Testing	days	1	365	45.66
Output Parameters				
	Units	**Minimum**	**Maximum**	**Average**
Compressive Strength	MPa	2.33	82.60	35.82

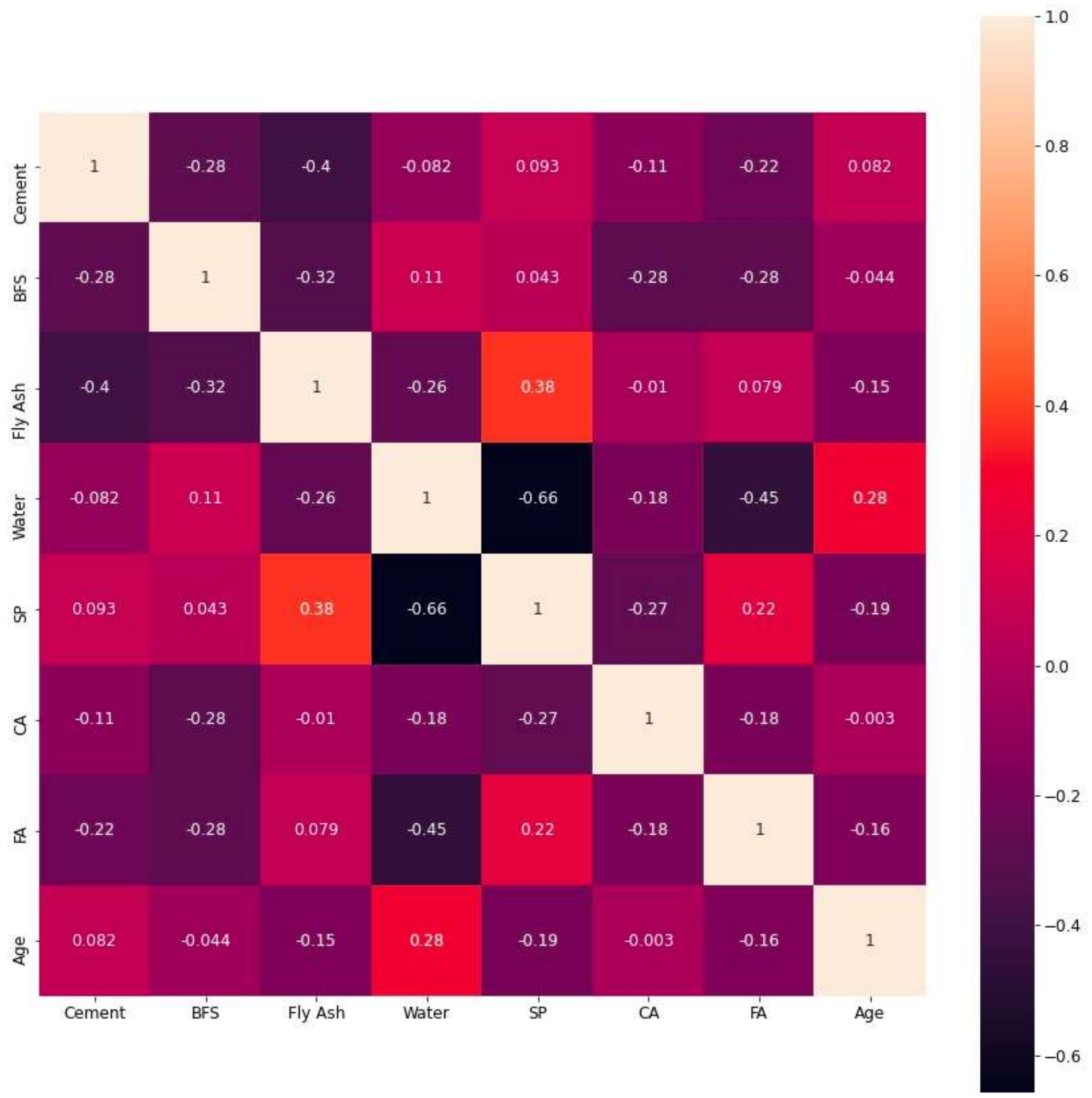


Fig 1: Correlation Matrix for the Dataset

2.2 Refining of Data

In order to increase the model efficiency, reduce computational time and achieve good efficiency by removing outliers, refining of data is necessary. Various methods like data scaling or normalization and data splitting can be used for data refinement as described below.

2.2.1 Data Scaling

The goal of Feature Scaling, generally known as Normalization [28], is to change the values in dataset on a common scale, such that distortion between different features is reduced without losing any information in the dataset. Normalization of dataset is performed using Z-score. Here, $[x]$ is the raw score for the observation, μ is the mean of the set and σ is the standard deviations of the set as given in equation 1.

$$[x] = [x] - \mu/\sigma \qquad (1)$$

2.2.2 Data Splitting

The train-test split procedure is used to estimate the performance of machine learning (ML) algorithms when they are used to make predictions on data not used to train the model. Dataset is divided into two subsets; namely, training data, used for training the models, and testing data for evaluating performance of different Machine Learning Models. Train-Test split of 80% is taken in formulations of different Machine Learning Algorithms for this study.

2.3 Performance Indices

In order to evaluate performance of various models for predicting compressive strength various performance indices are used as described below.

2.3.1 R^2 Score

The coefficient of determination, R^2 score, is a statistical measure of how close the predicted data fits the actual scatter. These values range between 0 and 1 depending upon the "goodness of the fit." Models with higher coefficients exhibit better performances.

$$R^2 = 1 - \sum_{i=1}^{n}(y-\hat{y})^2 \Big/ \sum_{i=1}^{n}(y-\bar{y})^2 \qquad (2)$$

In equation 2, $\hat{y}$ is the predicted value, y is the actual value, $\bar{y}$ is the mean of actual values and n is the number of data samples.

2.3.2 R Score

R Score also known as Variance Regression Score Function is another parameter, like coefficient of determination, ranging between 0 and 1 where higher value indicates better fit.

$$R = 1 - var(y-\hat{y})/var(y) \qquad (3)$$

$$var(y) = \sum_{i-1}^{n}(y-\bar{y})^2/n \qquad (4)$$

In equation 3 and 4, $\hat{y}$ is the predicted value, y is the actual value, $\bar{y}$ is the mean of actual values and n is the number of data samples.

2.3.3 Root Mean Squared Error (RMSE)

The general loss function used in regression analysis is statistically known as the mean squared error function. Dimensional rectification yields the RMSE value, which is smaller for better models.

$$RMSE = \sqrt{\sum_{i=1}^{n}(y-\hat{y})^2/n} \qquad (5)$$

In equation 5, $\hat{y}$ is the predicted value, y is the actual value and n is the number of data samples.

2.3.4 Mean Absolute Error (MAE)

MAE is the arithmetic mean of the deviation, expressed as the absolute value minus the total mean of each measurement.

$$\text{MAE} = \sum_{i-1}^{n}|(y-\hat{y})|/n \qquad (6)$$

In equation 6, $\hat{y}$ is the predicted value, y is the actual value and n is the number of data samples.

2.3.5 Median Absolute Error (MedAE)

MedAE is median of absolute deviation of each point in dataset and henceforth, robust to outliers.

$$MedAE = median\{\bigcup_{i=1}^{n}|(y_i - \hat{y}_i)|\} \qquad (7)$$

In equation 7, $\hat{y}$ is the predicted value, y is the actual value and n is the number of data sample.

3. MACHINE LEARNING MODELS

Machine Learning models like linear regression, ridge regression, lasso regression, k-Nearest Neighbor (kNN) Regression, polynomial regression, forest regression, decision tree regression, gradient boosting regression, support vector regression and various neural network models were used to predict compressive strength results. These state-of-the-art models were considered in this study because of factors like simplicity, effectiveness, efficiency and relevance.

3.1 Linear Regression

Linear Regression (LR) is the most basic Supervised Learning Method for making predictions. The algorithm works by using the input parameters and output parameters in the training set and finding the optimized set of coefficients, formally known as parameters. The Gradient Descent techniques on Mean Squared Loss functions are used in this study for the yield of the optimized vector of Parameters [27,28].

$$\hat{y} = \theta_0 + \theta_0 x_1 + \theta_2 x_2 + \ldots + \theta_m x_m \qquad (8)$$

$$J(\theta) = \sum_{i-1}^{n} (y - \hat{y})^2 / 2n \qquad (9)$$

In equation 8 and 9, θ represents the corresponding parameters and J (θ) is the loss function.

3.2 Ridge Regression

Ridge Regression has a similar hypothesis to that for Linear Regression (LR) with added L2-Norm Squared Regulation, which does not apply when assessing performance in a test set or predicting an actual sample. Hyperparameter controls how much the model will be regulated. When the hyperparameters are very large, all parameters tend to be zero and eventually become a horizontal line over the average of the data [28].

$$J(\theta) = \sum_{i-1}^{n} (y - \hat{y})^2 / 2n + \alpha \sum_{i-1}^{m} \theta_i^2 \qquad (10)$$

In equation 10, y represents truth value, $\hat{y}$ represents predictions, θ the corresponding parameters, α is the penalty and J (θ) is the loss function.

3.3 Lasso Regression

Lasso Regression has a similar hypothesis to that for Linear regression (LR) with added L1-Norm Absolute Regulation which does not apply when assessing performance in a test set or predicting an actual sample. Hyperparameters control how much the model will be regulated. When the hyperparameters are very large, all parameters tend to be zero and eventually become a horizontal line over the average of the data [28].

$$J(\theta) = \sum_{i=1}^{n} (y - \hat{y})^2 / 2n + \alpha \sum_{i=1}^{m} |\theta_i| \qquad (11)$$

In equation 11, y represents truth value, $\hat{y}$ represents predictions, θ the corresponding parameters, α is the penalty and J (θ) is the loss function.

Figure 2, shows the graph of gradient descent which give an intuitive idea of walking downhill to reduce the losses.

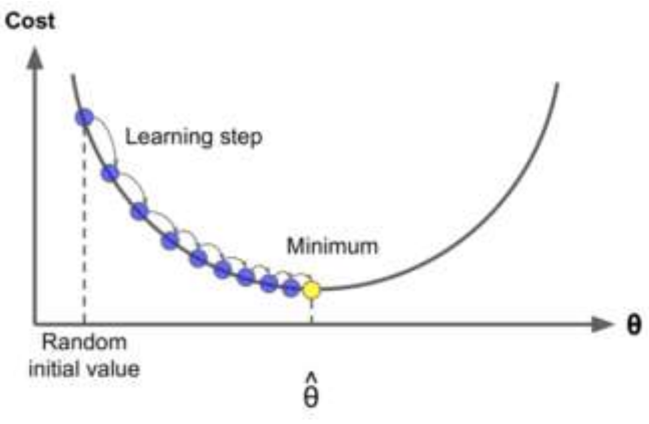


Fig. 2. Gradient Descent [29]

3.4 k-Nearest Neighbor Regression

k-Nearest Neighbor (kNN) Algorithm, although generally used for classification, can yield accurate predictions on an optimized k value for large datasets. A standard kNN algorithm consists of assigning similar weightage to all training data points, calculating Euclidean distances and use the average of a data space comprised of predefined k data points to record observations for unknown values.

$$d = \sqrt{(x - x_i)^2 + (y - y_i)^2 + .. + (z - z_i)^2} \tag{12}$$

In equation 12, x, y, z represents new points and x_i, y_i, z_i represents existing points.

3.5 Polynomial Regression

For complex data, other than a straight line, polynomial regression is incorporated. The higher-order terms can be viewed as additional features with a similar hypothesis to that for linear regression. The chances of overfitting increase with the degree of the polynomial [21], and thus, bias-variance trade-off is essential.

3.6 Random Forest Regression

Using a random forest (RF) is a way of constructing a regression tree ensemble to reduce the fluctuation of individual trees. Decision trees gather to build a forest, by using the concept of ‘‘bootstrap aggregation” (bagging) to create many similar datasets sampled from the same source dataset. Bagging is a method of combining a trained basic model for training data. Because of its small bias and great variance, the tree is vulnerable to overfitting. The Random Forest (RF) algorithm introduces extra randomness when growing trees; instead of searching for the very best feature when splitting a node, therein reduces instability. However, the drawback of the decision tree is that it tends to overfit training data which can be regulated [28,29].

3.7 Decision Tree Regression

Decision Tree, generally used for classification, is a node-based method with parameters like maximum depth of trees. Each node predicts a value after going through a series of true/false “branches”. The Classification and Regression Tree (CART) training algorithm is used for bifurcation of the values at the node, hence minimizing the mean-squared error for both nodes. The cost function for a decision tree node can be written as,

$$J(\theta) = m_L MSE_L + m_R MSE_R / m \tag{13}$$

$$MSE_{node} = \sum_{i\in node} (y-\hat{y})^2/2m_{node} \tag{14}$$

In equation 13 and 14, MSEL means the mean squared error L branches, y represents truth value, yˆ represents predictions, θ the corresponding parameters, α is the penalty and J (θ) is the loss function.

Although good for handling both categorical and numerical data and easier to understand, careful investigation is required as trees are prone to overfitting as a result of orthogonal boundaries and instabilities associated with it. A trivial change can result in completely different tree structures and consequently different predictions [28]. Figure 3, describes the flow chart for decision tree model for compressive strength evaluation.

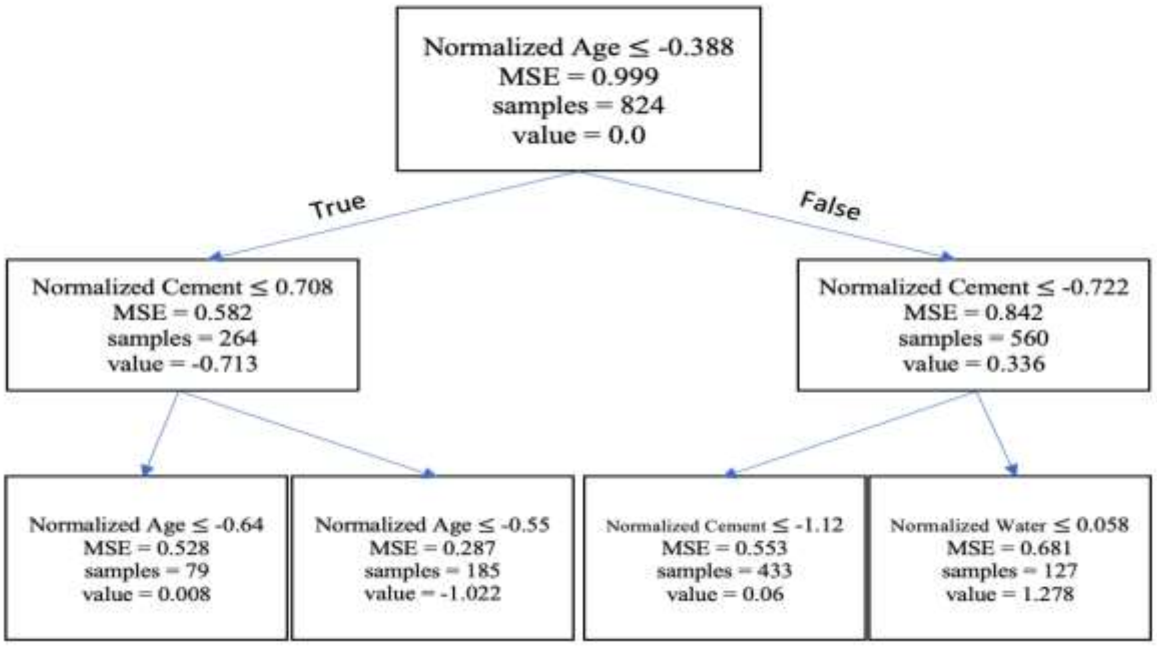


Fig. 3. Starting nodes for Decision Tree

3.8 Gradient Boosting Regression

Gradient Boosting Regression works by adding predictors to an ensemble in a sequential manner, each one correcting the previous one. Unlike simpler boosting algorithms, which change the instance weights at each iteration, this method seeks to adapt a new predictor to the residual errors created by the preceding predictor [28].

A gradient boosting tree usually has less than five levels, which utilises less memory and allows for faster predictions. The basic principle of gradient boosting is to connect a number of simple models, such as weak learners, which are shallow trees. Only a small portion of the data can be predicted by each tree. The performance of the system can be improved by adding more trees [20].

3.9 Support Vector Regression

Support Vector Regression (SVR), the “black-box” algorithm was proposed by Vapnik et al. [30]. The intuition behind SVR algorithm is to fit maximum number of possible points on a predefined street, either polynomial or exponential, and reduce instances of “off-street violations” [21]. The width of the said street is controlled by a hyperparameter ε. The prediction values and the loss function for SVR for kernel *f(x)* can be written as:

$$\hat{y} = \theta_0 + \sum_{i-1}^{n} \theta_i f(x_i) \qquad (15)$$

$$J(\theta) = \begin{cases} 0 \; if \; |y-\hat{y}| < \varepsilon \\ |y-\hat{y}| \; else \end{cases} \qquad (16)$$

For hard margins, restrict all margin violations, the hypothesis can be written as:

$$minimize \; (1/2)\theta\theta^T \; or \; (1/2)\|\theta\|^2 \qquad (17)$$

$$subjected \; to \; |y-\hat{y}| \le \varepsilon \; for \; \forall i \qquad (18)$$

For soft margins, allowing certain instances to override violation protocols, the hypothesis has an additional slack variable with measures the allowance of various instances;

$$minimize \; (1/2)\|\theta\|^2 + C\sum_{i-0}^{n} \zeta_i \qquad (19)$$

$$subjected \; to \begin{cases} |y-\hat{y}| \le \varepsilon + \zeta \\ \zeta > 0 \end{cases} fo \qquad (20)$$

In equation 16, 17, 18, 19 and 20, y represents truth value, $\hat{y}$ represents predictions, θ the corresponding parameters, α is the penalty and J (θ) is the loss function.

Figure 4 shows how well the kernels perform, where red dots indicate the data points used for training the models.

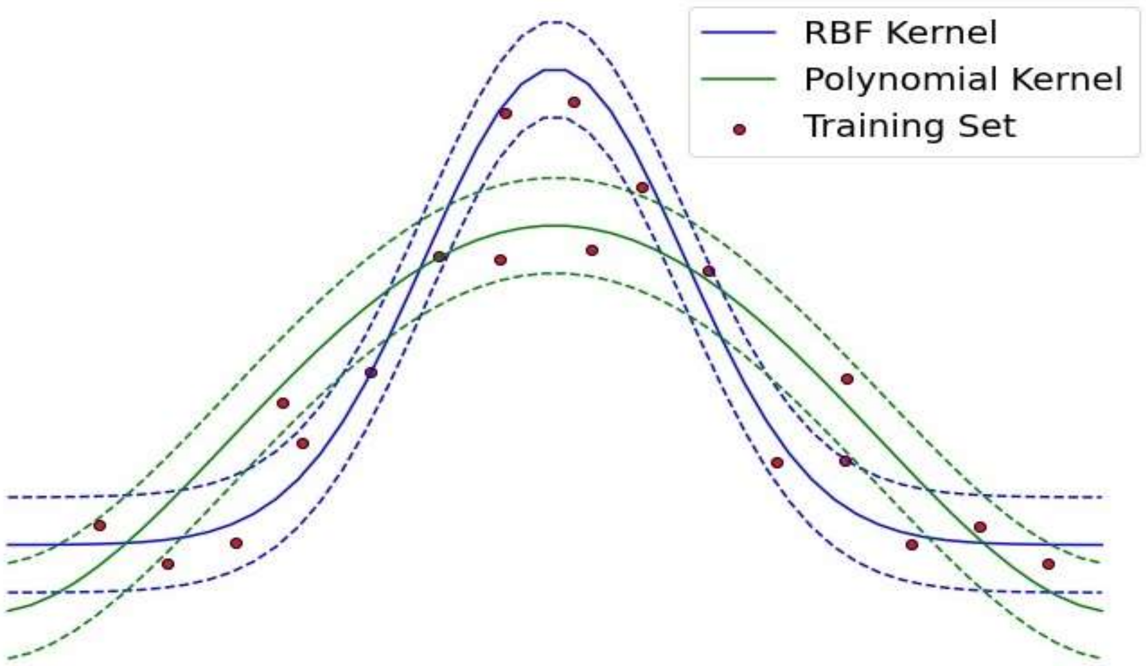


Fig. 4. Kernels for SVR

3.10 Voting

The simplest ensemble method, combination of multiple algorithms, is voting. Various algorithms are pooled, and the numerical output can be derived by different combinations of probability estimates. By a combination of three and four better-performing regressions, five different voting regressors are developed for this study. From the results of individual models it was observed Random Forest (RF), Decision Trees (DT), Gradient Boosting (GB), Support Vector Regressions (SVR) are better performing models and Linear Regression (LR) was considered as a baseline.

3.11 Stacking

Stacking short for Stacked Generalization [31], is a multi-level hierarchy of various regressors. The first layer of single regressors, known as stacked regressor is used to train the second level of the hierarchy. After blending, the final outputs are made. Unlike static regressions by voting, stacking is deemed a more “trainable” technique. By a combination of three and four better-performing regressions, five different stacking regressors are developed for this study. From the results of individual models it was observed Random Forest (RF), Decision Trees (DT), Gradient Boosting (GB), Support Vector Regressions (SVR) are better performing models and Linear Regression (LR) was considered as a baseline.

3.12 Bagging

The bagging method uses the bootstrap method to train several classifiers independently and with different training sets [32]. Bootstrapping builds numerous replicate training datasets to construct numerous independent regressor by randomly re-sampling and replacing the original training dataset. Various bagging regressors considering LR ensemble, Gradient Boosting ensemble, RF ensemble, SVM ensemble, and DT ensemble are developed under this study.

3.13 Artificial Neural Networks

“A Logical Calculus of Ideas Immanent in Nervous Activity,” McCulloch and Pitts presented a simplified computational model of how biological neurons might work together in animal brains to perform complex computations using propositional logic. This was the first artificial neural network architecture. It includes the main features of biological neural networks: parallelism and high connectivity. Artificial neuron structures consist of a total of five parts: input, weights, the sum function, the activation function, and output. [33]

The basic idea of Neural Networks oscillates between processes like forward propagation, backpropagation, and activation functions. The forward propagation works like a basic regression for each layer with different weighted functions and applying activation functions to it.

$$x^{(n+1)} = g(\theta^{T} \cdot x^{(n)}) \qquad (21)$$

Where, x represents the data points and θ represents the parameters. General architecture of neural network is represented in figure 5.

After the forward propagation for all layers, the algorithms dives backwards to estimated errors, selected from a pool including mean squared error, mean absolute error or median absolute errors. This process is known backpropagation for Neural Networks.

$$\delta^{l} = (\theta^{l+1})^{T}\delta^{l+1} \cdot g'(x(n)) \qquad (22)$$

$$\frac{dC}{d\theta_{ik}^{l}} = \hat{y}_{k}^{l-1}\delta_{j}^{l} \qquad (23)$$

In equation 22 and 23, y represents truth value, $\hat{y}$ represents predictions, θ the corresponding parameters, δ represents the gradient of the super-scripted layer and *g(x)* is the activation function.

Table 2 presents the general idea of activation functions that can be used to obtain output at any layer of neural network.

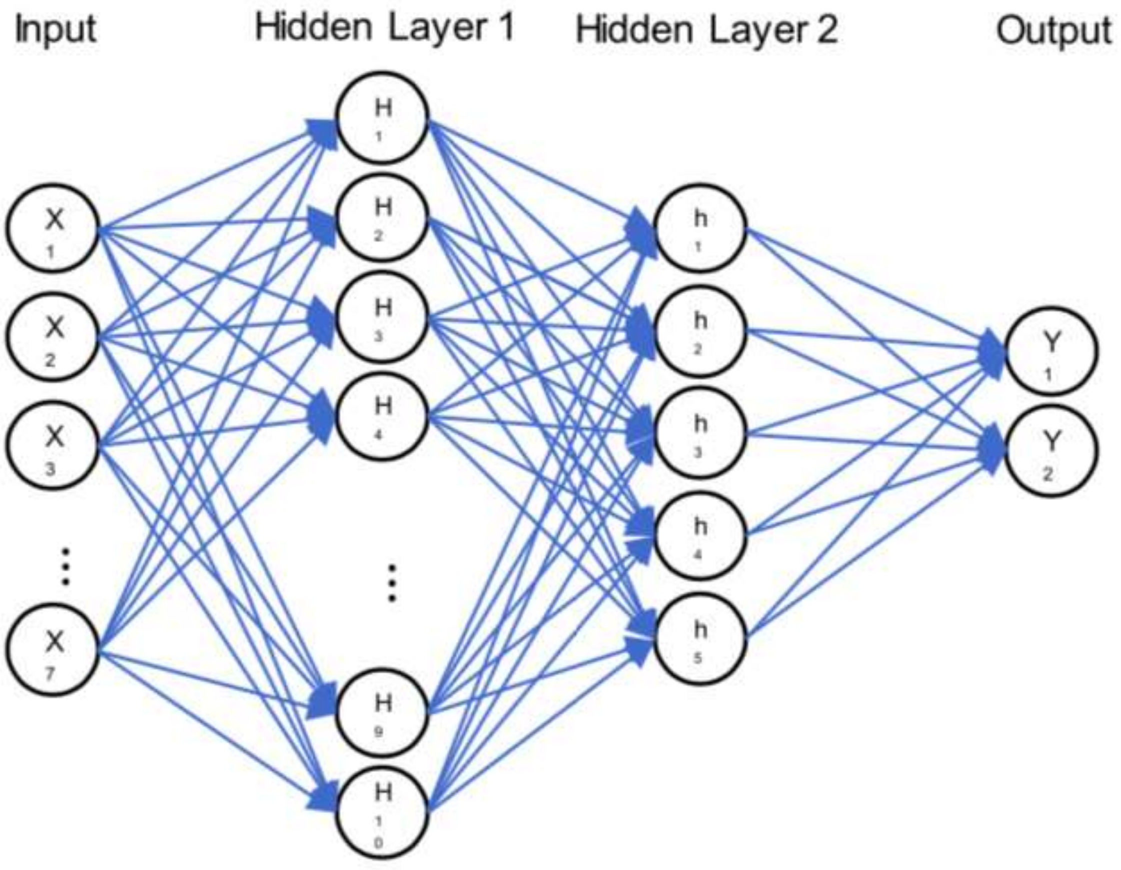
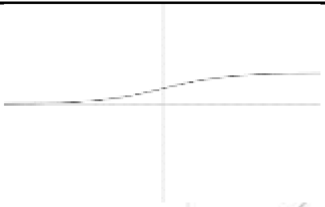
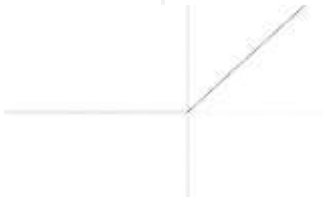
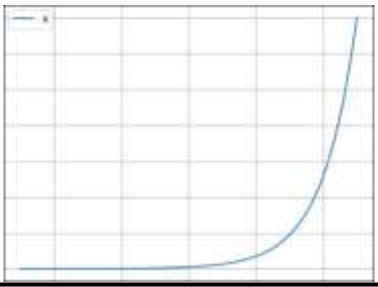


Fig. 5. Architecture for 2-Layered ANN [29]

Table 2. Activation Functions utilized

Activation Function	Graphs	Mathematical Models
Sigmoid		$g(x) = (1 + e^{-x})^{-1}$
ReLu (Rectified Linear Unit)		$g(x) = max(0, x)$

Softmax		$g(\vec{x}) = \frac{e^{x_i}}{\sum e^{x_i}}$

3.14 Gradient Descent

Almost every Machine Learning (ML) hypothesis runs on minimization of errors which can be imagined as descending a gradient. Different algorithms can be used to “climb down the hill” in correspondence to runtime or accuracy. Methods like AdaGrad or Adam Gradient Descent are swift in application while others like Stochastic or Momentum Gradient Descent, although slower in converging, are easier to apply. As evident from the pictorial representation (fig. 6) [18] , the red line depicting stochastic gradient descent converge slowly to the center of the image and gets wavy and unstable near the center. The green and blue lines depicting momentum and nesterov respectively, also show similar behavior. However, consistency is clearly visible for adagrad, rmsprop and adam optimizers, however, the trade off, here is that the algorithm poorly converges to minimum and the results are not the best.

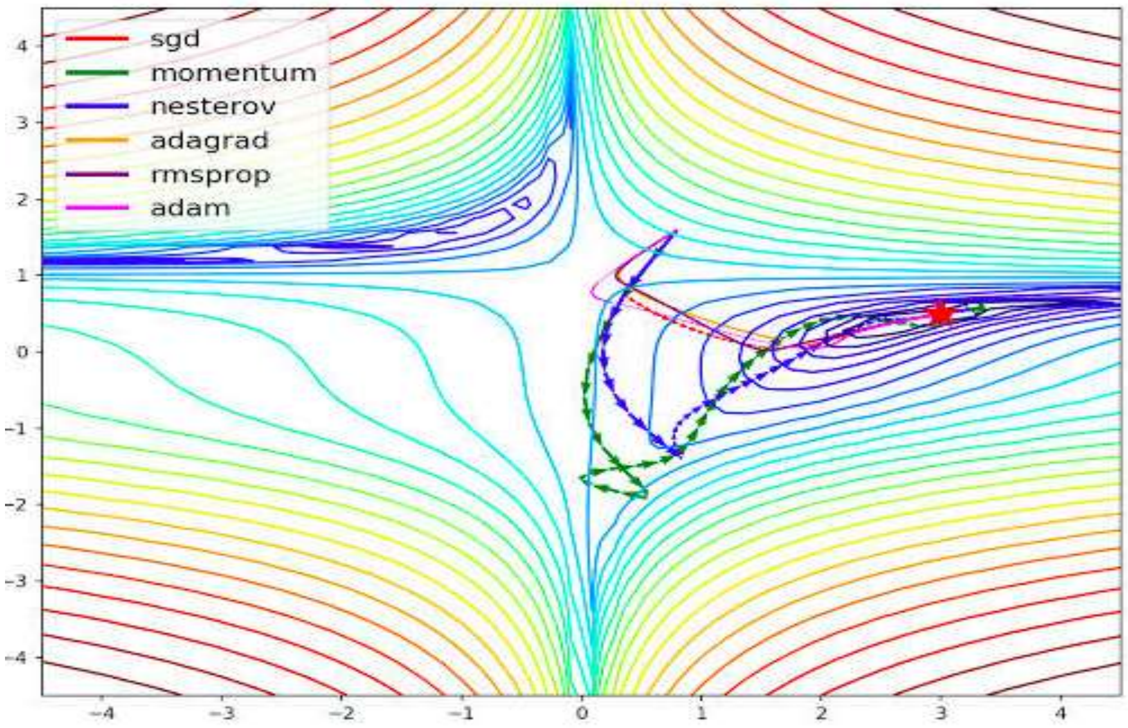


Fig. 6. NumPy Gradient Descent Function [18]

3.15 Model specifications

For modifications in Linear Regression (LR), the modification factor alpha=5 was used. Cost Function used for both Lasso Regression and Ridge Regression was mean squared error loss. The advantages for this costing function over others include, parabolic losses which converge readily, optional use as measures of uncertainty in forecasting. Normal Gradient Descent was used for Lasso Regressions but for Ridge, a linear algebraic approach of Normal Equation was used.

In k-Nearest Neighbors (kNN) algorithm, basic parameters were implemented. Statistical average of 2 neighbors determined by the Euclidean distance was used for predictions. For Random forest (RF) regression, mean squared error (MSE) loss was advantageous as it includes, parabolic losses which converge readily, optional use as measures of uncertainty in forecasting. The maximum depth of the forest was set to be 100. For SVM, the C parameter suggests the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors. In RBF and polynomial modelling, C=100, gamma=0.05, epsilon=0.2 parameters are used. Polynomial kernel pertains to the 4th degree.

To summarize, the architecture of all the models and its specifications are mentioned in Table-3.

Table 3. Model Description

Model	**Description**
	alpha=5
Ridge Regression	Normal Equation
	Mean Squared Error Loss
	alpha=5
Lasso Regression	Gradient Descent
	Mean Squared Error Loss
	k=2
kNN	Euclidean Distance
	Statistical Average Output
Random Forest	n=100
	Mean Squared Error Loss
	C=100,
Polynomial SVM	gamma=0.05,
	degree=4, epsilon=0.2
	C=100,
RBF SVM	gamma=0.05,
	epsilon=0.2

Furthermore, in training neural networks, validation losses are implemented to track the points for which it drops beyond cost function descent. The patience for validation losses is taken as 100 with a batch size of 32. Cost Function of mean squared error (MSE) loss was advantageous as it includes, parabolic losses which converge readily, optional use as measures of uncertainty in forecasting. The layers with 1000 neurons were used as described with Adam gradient descent. Adam can be looked at as a combination of RMSprop and Stochastic Gradient Descent with momentum. Bias-correction helps Adam slightly outperform RMSprop towards the end of optimization as gradients become sparser [18]. The architecture of neural network models deployed for compressive strength predictions of concrete in this project is summarized in Table-4.

Table 4. Neural Network Description

ANN Model	**Description**
	3 Sigmoid Layers
	Validation Batch=32
Sigmoid Activation	Patience = 100
	Adam Gradient Descent
	Mean Squared Error Loss
	3 ReLu Layers
	Validation Batch=32
ReLu Activation	Patience = 100
	Adam Gradient Descent
	Mean Squared Error Loss
	3 Softmax Layers
	Validation Batch=32
Softmax Activation	Patience = 100
	Adam Gradient Descent
	Mean Squared Error Loss
	2 ReLu + 1 Sigmoid Layer
	Validation Batch=32
ReLu + Sigmoid	Patience = 100
	Adam Gradient Descent
	Mean Squared Error Loss

4. Results and Discussions

The compressive strength of concrete was predicted through various machine learning techniques. The predicted values are compared to actual values in the dataset and various statistical ratios are calculated. Any hypothesis having performance scores on a higher side or lower statistical errors are said to be performing well.

For individual models like Random Forest (RF) Regression or Radial-Basis Support Vector Machines have good performance scores. Amongst both methods having similar performance scores, Random Forest Regressions has the least statistical error. Similarly, all best combinations for all ensemble methods have R-Scores within a particular range but the least statistical error is offered by Voting amongst Gradient Boosting, Random Forest Regression, Radial-Basis Support Vector Machines and Linear Regression. Artificial Neural Network with Sigmoid and ReLu Activation is distinctly the best performing in the said category.

Linear Regression, Lasso Regression and Ridge Regression are basic statistical regressions having R^2 scores of 0.48. This suggests that predicted parameters never overfit the sample points in the dataset. Quadratic Regression or Cubic Regression can be included in the family pertaining to basic statistical regressions with certain modifications in degrees and an increase in parameters, which increases R^2 scores to 0.75 and 0.85 respectively. k-Nearest Neighbors, Random Forests, Decision Trees fall into the category of classification algorithms. With certain modelling changes these methods yield R^2 scores of 0.68, 0.89 and 0.83 respectively.

Support Vector Machines, regarded as the black-box technique, and Gradient Boosting Algorithms are advanced regression models. Radial Basis Function kernel performs better than the polynomial kernel for the dataset with scores of 0.89 and 0.87 respectively. Overall, SVMs are more successful than Gradient Boosting Techniques with R^2 score of 0.83. For ensemble methods, voting amongst GB+DT+SVR+LR, stacking of GB+RF+SVR+LR and bagging on SVR are the best performing models with the R^2 scores of 0.90, 0.91 and 0.90 respectively. Neural Networks, incorporating the correlation between different parameters, are observed to the best performing. For different activation functions like sigmoid, ReLu and softmax, R^2 scores of 0.87, 0.92 and 0.92 respectively. On combining ReLu and softmax layers in a single network, R^2 score of 0.93 is obtained. To summarize, a comparative analysis of the R^2 and R scores of all the models deployed for predicting compressive strength of concrete are mentioned in Table-5.

Furthermore, statistical errors have been calculated using the metrics Root mean square error(RMSE), Mean absolute error(MAE) and Median absolute error(MedAE) for all the models deployed for predicting compressive strength of concrete in this paper as given in Table-6. It can be observed that the RMSE values are considerably higher for each model than MAE and MedAE scores as for regression the errors are squared which leads to higher errors. Ideally, A lower RMSE score indicates that the model fits better and the predictions are accurate. Same is true for MAE. Median absolute error doesn’t prove much of the performance of the models in our case. It can be observed that linear regression, Lasso regression, ridge regression and k-nearest neighbors models perform almost identical. While the performance is better for polynomial models with a score of 8.12, 5.83, 4.09 for quadratic and 6.69,4.60, 2.97 for RMSE, MAE and MedAE respectively. However, the best performance is shown by Random forest technique with a score of 5.42, 3.82, 2.60 for RMSE, MAE and MedAE respectively when deployed individually. Further complex techniques of voting and stacking gives even better performance. Gradient boosting, support vector regression, decision trees and linear regression when deployed together using voting techniques, the model performs best amongst all with a score of 5.08, 3.71, 2.50 for RMSE, MAE and MedAE respectively. The same when deployed using stacking techniques with a minor change of replacing decision tree model with random forest performs almost identical with scores of 5.08, 3.75 and 2.63 for RMSE, MAE and MedAE respectively. The SVR based model deployed using bagging techniques also performs quite well with scores of 5.60, 4.02 and 3.24 for RMSE, MAE and MedAE respectively. However, as expected, the sophistication of neural network models gives a slight edge over other traditional machine learning models with statistical error scores of 4.91, 3.40 and 2.51 for RMSE, MAE and MedAE respectively when sigmoid and ReLu activation functions are stacked together.

The graphs are plotted for all data points in the Train-Test Split with Predicted Data on abscissa and Actual Data on Ordinate. For any model, more clustered the points are about the line L:x=y, the better performing it is. For most hypotheses included in the study, the training points do form a good cluster, but testing points do not and hence the performance cannot be justified. For example, training set clusters for gradient booster regression are unmatched yet in the lower end of the performance hierarchy as shown in figure 7 (). Random Forest Regression, Radial-Basis Support Vector Machines, All Ensemble Methods plotted and, Artificial Neural Network with Sigmoid and ReLu Activation can be entitled “well” performing in the study as observed pictorially.

It can be observed from figure-7 (a), (b) and (c) representing predictions and model performance for linear regression, ridge regression, and lasso regressions respectively, that the predictions are scattered around y=x line that is our model unevenly and hence the model techniques linear regression, ridge regression and lasso regression do not perform satisfactorily. Similarly figure-7 (f), (g), (h), (i), (l), and (n) representing predictions and model performance for k-nearest neighbor regression, cubic regression, quadratic regression, radial basis kernel SVM, polynomial kernel SVM, voting, and bagging models respectively, the performance is considerably good as can be viewed pictorially that the predictions lie mostly around y=x line in closed clusters. However, the most ideal graphs can be visible for figure 7- (d), (e), (j) and (k) representing predictions and model performance for random forest regression, decision tree regression, gradient booster regression and stacking techniques respectively, where the predictions are mostly captured and clustered by y=x line which essentially means that the predicted values match the actual data. However, there are outliers for all the model techniques which can be viewed as dots at a significant distance away from y=x line.

Table 5. Performance Scores

Algorithm	R^2	R
Individual Models		
LR	0.48	0.67
Ridge	0.48	0.67
Lasso	0.48	0.67
kNN	0.68	0.71
Quadratic	0.75	0.78
Cubic	0.85	0.85
RF	**0.89**	**0.90**
DT	0.83	0.80
Poly SVM	0.87	0.87
RBF SVM	**0.89**	**0.89**
GB	0.84	0.82
Voting		
GB+RF+LR	0.85	0.89
GB+SVR+LR	0.86	0.89
GB+DT+LR	0.88	0.90
GB+RF+SVR+LR	0.88	0.91
GB+DT+SVR+LR	**0.90**	**0.91**
Stacking		
GB+RF+LR	0.90	0.90
GB+SVR+LR	0.90	0.90
GB+DT+LR	0.90	0.89
GB+RF+SVR+LR	**0.91**	**0.91**
GB+DT+SVR+LR	0.91	0.91
Bagging		
GB Based	0.86	0.87
RF Based	0.87	0.89

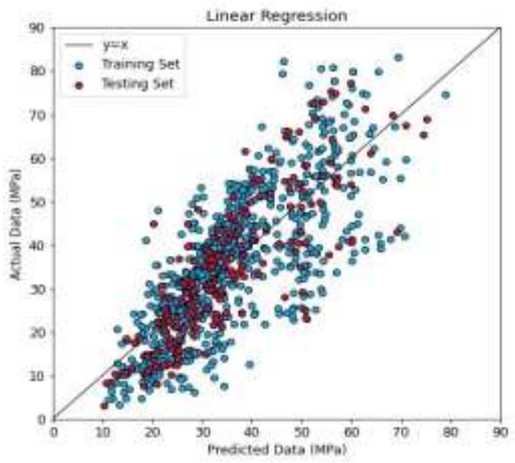
LR Based	0.47	0.67
SVR Based	**0.90**	**0.89**
DT Based	0.86	0.87
Neural Networks		
Sigmoid Activation	0.87	0.89
ReLu Activation	0.92	0.92
Softmax Activation	0.90	0.90
Sigmoid + ReLu	**0.93**	**0.92**

Table 6. Statistical Scores

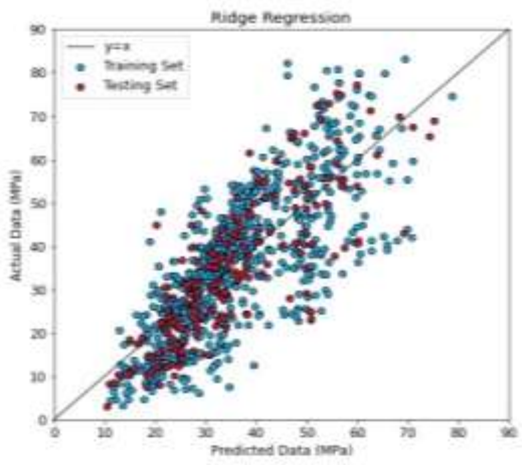
Algorithm	RMSE (MPa)	MAE (MPa)	MedAE (MPa)
Individual Models			
LR	9.89	7.84	6.31
Ridge	9.89	7.84	6.28
Lasso	9.90	7.85	6.11
kNN	9.32	6.81	4.64
Quadratic	8.12	5.83	4.09
Cubic	6.69	4.60	2.97
RF	**5.42**	**3.82**	**2.60**
DT	7.79	5.07	2.77
Poly SVM	6.22	4.29	3.09
RBF SVM	**5.74**	**4.06**	**2.72**
GB	7.40	5.49	4.10
Voting			
GB+RF+LR	5.71	4.31	3.23
GB+SVR+LR	5.67	4.19	3.06
GB+DT+LR	5.46	4.14	3.13
GB+RF+SVR+LR	5.30	3.86	2.68
GB+DT+SVR+LR	**5.08**	**3.71**	**2.50**
Stacking			

GB+RF+LR	5.46	4.10	3.00
GB+SVR+LR	5.51	4.25	3.37
GB+DT+LR	5.63	4.34	4.34
GB+RF+SVR+LR	**5.08**	**3.75**	**2.63**
GB+DT+SVR+LR	5.15	3.94	2.90
Bagging			
GB Based	6.14	4.62	3.54
RF Based	5.76	4.16	2.91
LR Based	9.87	7.85	6.22
SVR Based	**5.60**	**4.02**	**3.24**
DT Based	6.20	4.45	3.08
Neural Networks			
Sigmoid Activation	5.78	4.30	3.15
ReLu Activation	5.00	3.50	2.50
Softmax Activation	5.48	3.75	2.61
Sigmoid + ReLu	**4.91**	**3.40**	**2.51**

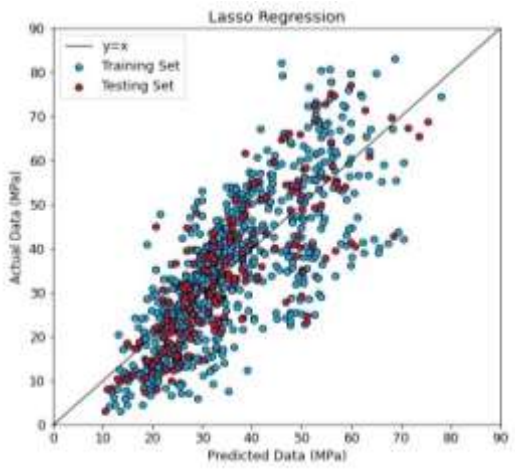
In Table 5 and 6, the best performing algorithm from the group is boldfaced for distinction. For example, Sigmoid + ReLu activation layers offers the least error in the group of Neural Networks.



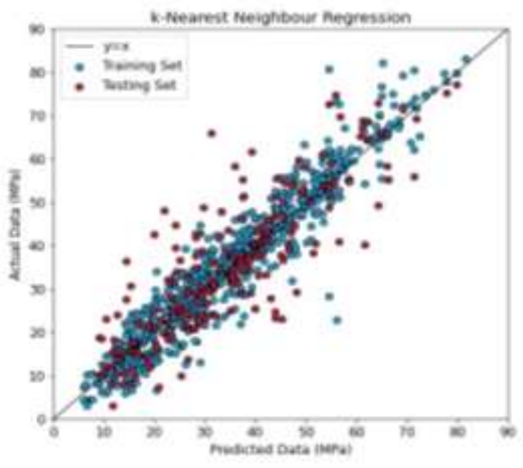
a



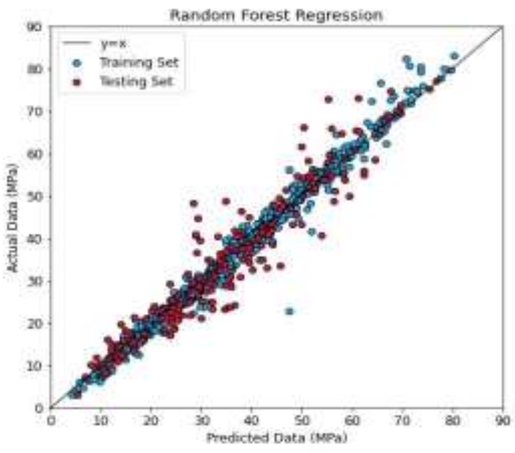
b


Lasso Regression
y=x
Training Set
Testing Set
Actual Data (MPa)
Predicted Data (MPa)

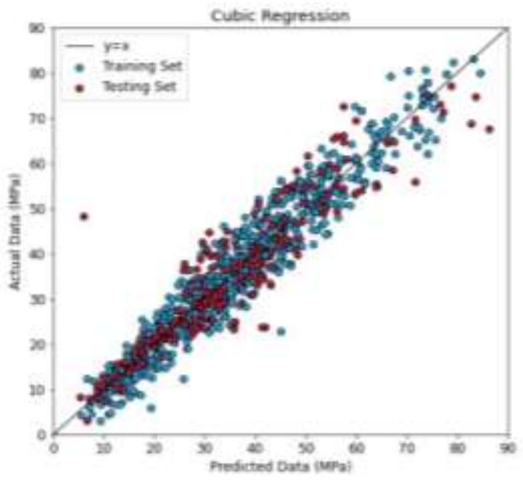
c


k-Nearest Neighbour Regression
y=x
Training Set
Testing Set
Actual Data (MPa)
Predicted Data (MPa)

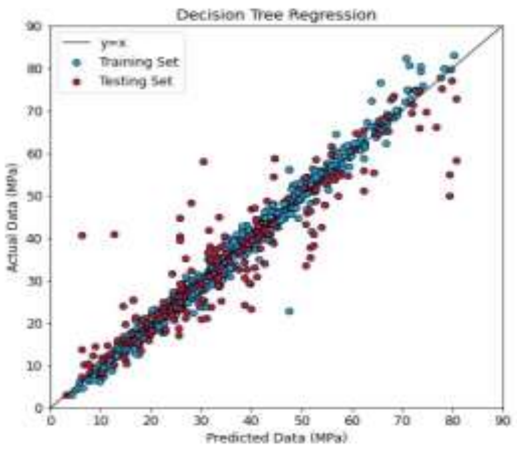
f


Random Forest Regression
y=x
Training Set
Testing Set
Actual Data (MPa)
Predicted Data (MPa)

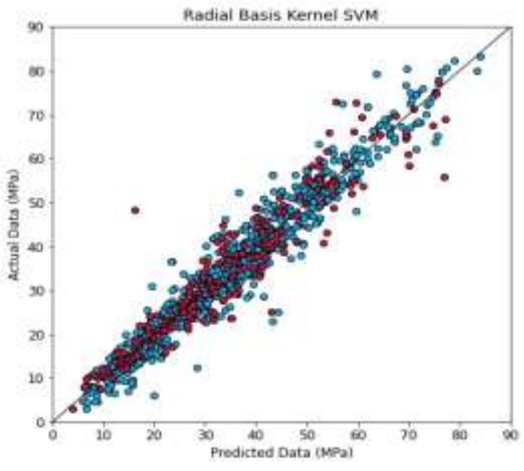
d


Cubic Regression
y=x
Training Set
Testing Set
Actual Data (MPa)
Predicted Data (MPa)

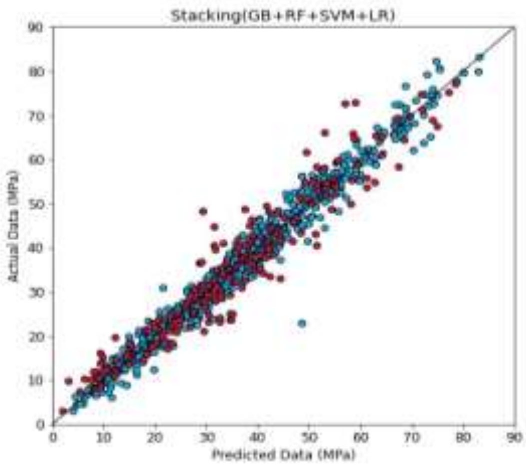
g


Decision Tree Regression
y=x
Training Set
Testing Set
Actual Data (MPa)
Predicted Data (MPa)

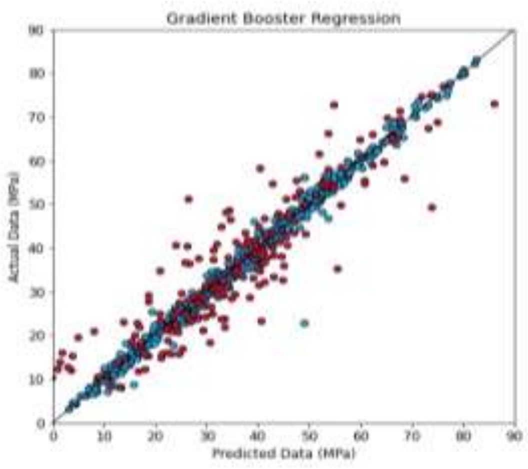
e


Radial Basis Kernel SVM
Actual Data (MPa)
Predicted Data (MPa)

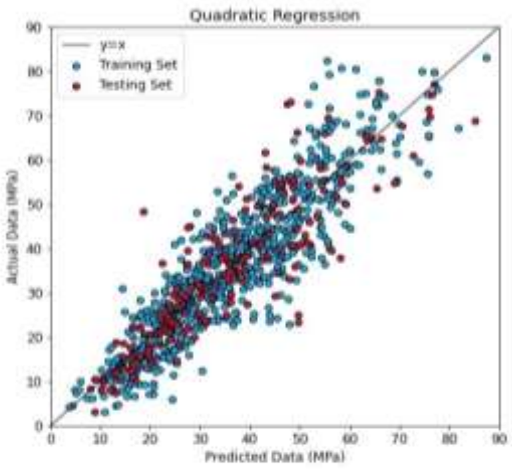
h



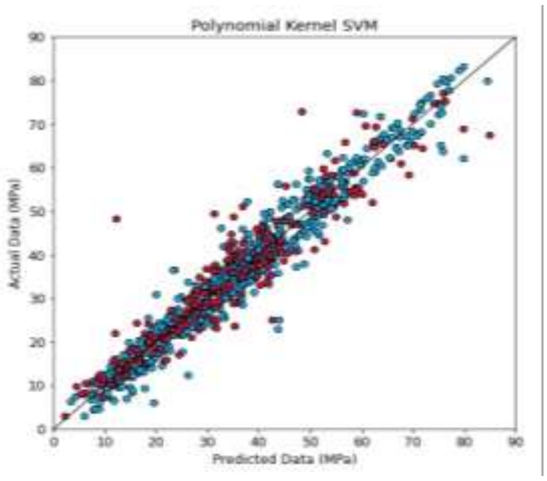
i



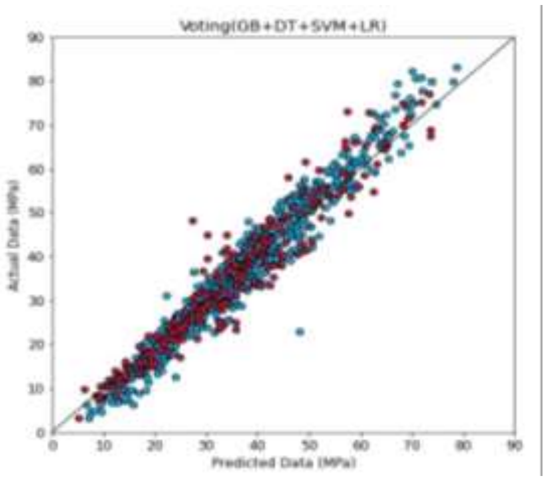
k



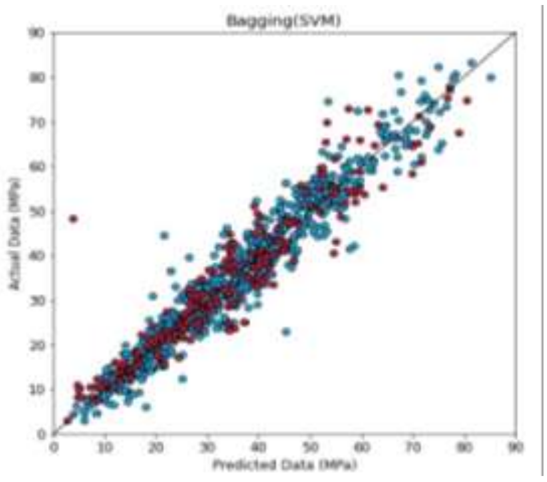
j



l



m



n

Fig. 7. Comparison of Compressive Strength for various algorithms

The statistical errors for voting, stacking and bagging techniques deployed for various combinations of gradient boosting (GB), support vector machines (SVM), random forest (RF), linear regression (LR), and decision trees (DT) is presented graphically as shown in fig. 8, with lowest bar levels performing the best of all.

The statistical errors for voting, stacking and bagging techniques deployed for various combinations of gradient boosting (GB), support vector machines (SVM), random forest (RF), linear regression (LR), and decision trees (DT) is presented graphically as shown in fig. 8, with lowest bar levels performing the best of all.

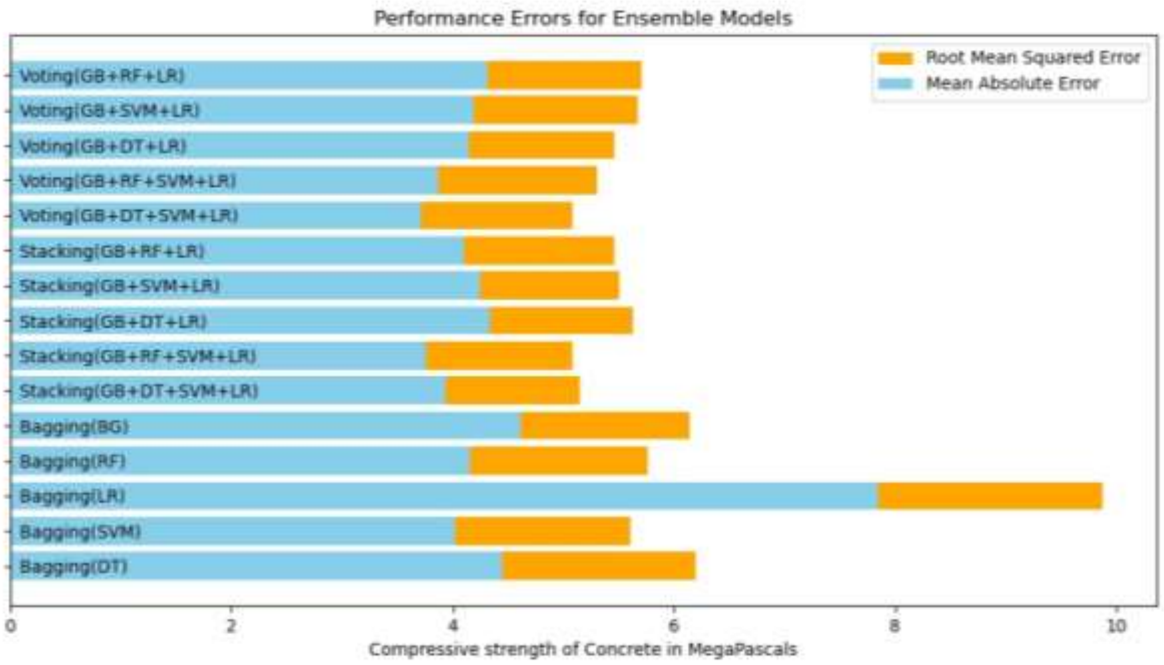
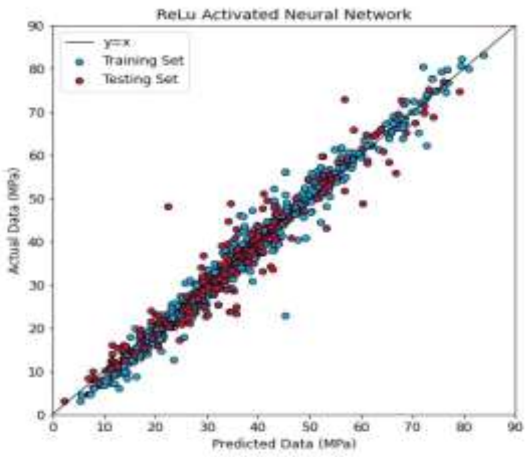
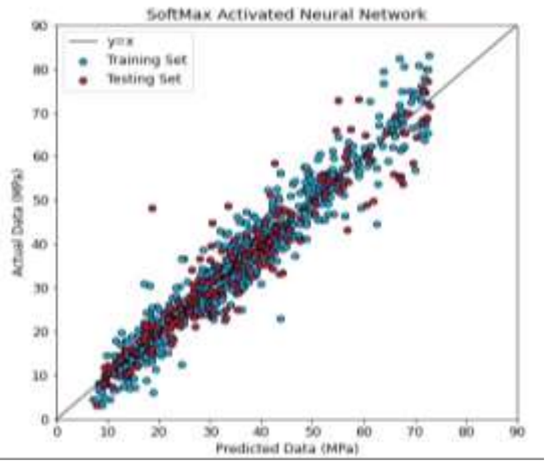


Fig. 8. Performance Errors for Ensemble Models

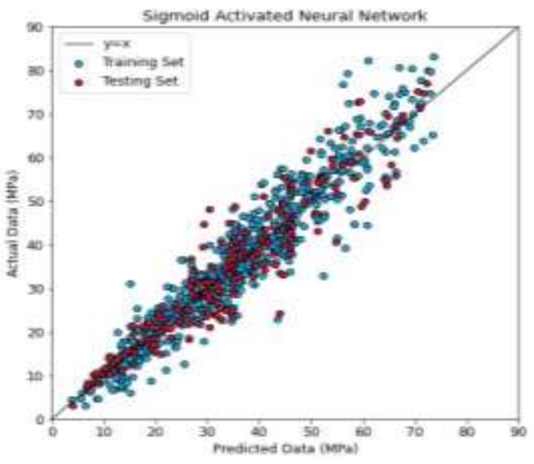
Similarly for neural network models deployed for predicting compressive strength of concrete in this paper, fig. 9-(b), and (c) representing model performance and predictions for models with sigmoid and softmax as activation functions respectively, the model performance is considerably good. However, the best performance can be observed when ReLu activation function, and ReLu along with sigmoid activation functions are used, as pictorially visible in fig. 9- (a) and (d) respectively.



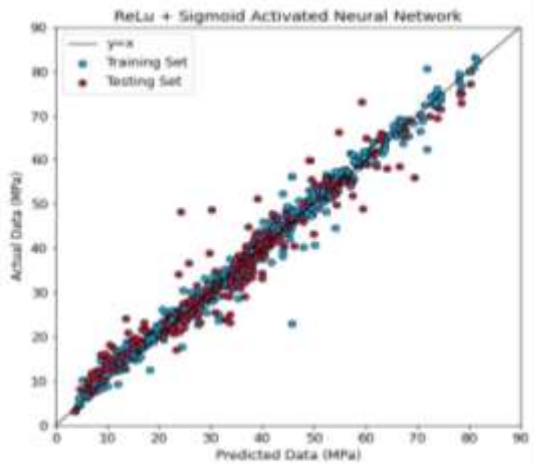
(a)



(c)



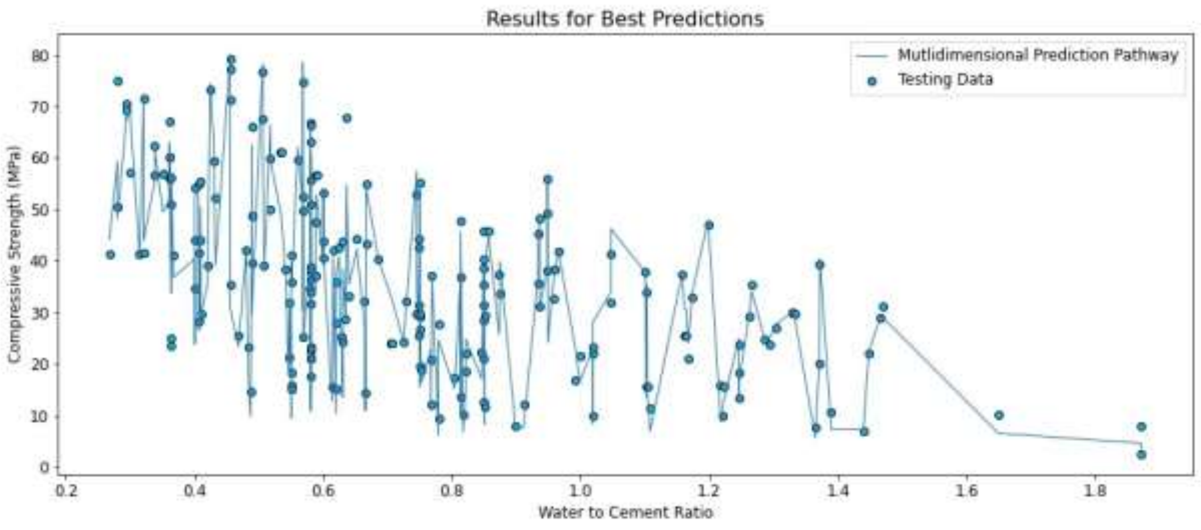
(b)



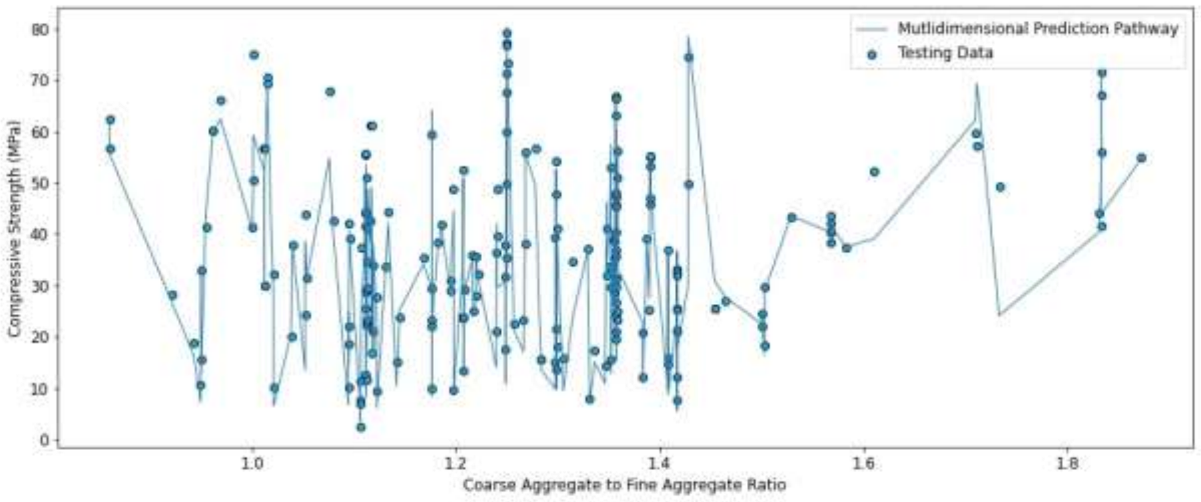
(d)

Fig. 9. Comparison of Compressive Strength for various ANN setups

Furthermore, fig. 10 (a) and (b) for Water-Cement Ratio and Coarse Aggregate to Fine Aggregate Ratio respectively, shows the predictions tracking the parameters water-cement ratio, and coarse aggregate to fine aggregate ratio, to compressive strength. Water-Cement Ratio and Coarse Aggregate to Fine Aggregate Ratio respectively are important terms for concreting procedures. The Water-Cement ratio plays very vital role in concrete mixture. The improper or random selection of Water-Cement ratio leads in various defects in fresh and hardened concrete. Water-Cement ratio should be optimum, which depends on grade of concrete and exposure conditions. Coarse Aggregate to Fine Aggregate ratio also has an impact on compressive strength and flexural strength of concrete. These ratios are calculated for all data points in testing split and a comparison in predicted values and specified values.



(a)



(b)

Fig. 10. Graphs for commonly used ratios under MLP/ANN

5. Conclusion

Scaling and normalization of dataset is very important for convergence of machine learning algorithms. Different statistical ratios and performance scores are to be calculated on the testing and training splits of the dataset to check the "goodness" of the fit whilst avoiding overfitting or underfitting. There is a strong correlation within the constitutive components of concrete with its compressive strength and hence preliminary regressions like linear regression, lasso regression, ridge regression, polynomial regression or kNN regression do not provide satisfactory results.

Some advanced classification techniques, turned into regression, work quite well incorporating inter parameteric interactions. Combining different regressions in an ensemble is apt for making "good" predictions. The epitome of performance scores is obtained by neural networks. The glibness of the whole algorithm including different activation functions, different network layers, forward propagation and back propagation algorithm, interconnectedness of the complete network yield pre-eminent results.

Water-cement ratio and Fine Aggregate to Coarse aggregate ratio are the most sought-after parameters for predicting compression strength of concrete on construction sites. The results shown in this paper are well in line with expectations of traditional construction practices and the model with superlative results can be opted for and deployed on site for predicting compressive strength of concrete.

6. Future Scope

The most effective Machine Learning Algorithms devised from this study can be utilized for various dataset for compressive strength, and best method for prediction of compressive strength of any concrete sample can be singled out for all future calculations.

Furthermore, similar studies can be conducted for prediction of compressive strength of Fiber Reinforced Concrete, Concrete Filled Tubes, High Performance Concrete, Lightweight Concrete and more. Along with it, various different properties like Flexural Strength, Split Tensile Strength, Electrical Resistivity, Modulus of Elasticity or Rigidity, Permeance can be predicted using appropriate Machine Learning hypothesis.

References

1. Bharatkumar BH, Narayanan R, Raghuprasad BK, Ramachandramurthy DS. Mix proportioning of high performance concrete. Cement and Concrete Composites 2001;23(1):71–80. https://doi.org/10.1016/S0958-9465(00)00071-8
2. Bhanja S, Sengupta B. Investigations on the compressive strength of silica fume concrete using statistical methods. Cement and Concrete Research 2002;32(9):1391–4. https://doi.org/10.1016/S0008-8846(02)00787-1
3. Atici U. Prediction of the strength of mineral-addition concrete using regression analysis. In: Concrete Research. Thomas Telford Ltd.; 2010. p. 585–92. https://doi.org/10.1680/macr.2010.62.8.585
4. Zain MFM, Abd SM. Multiple regression model for compressive strength prediction of high performance concrete. J Appl Sci 2009;9(1):155–60. https://doi.org/10.3923/jas.2009.155.160
5. K.O. Akande, T.O. Owolabi, S. Twaha, S.O. Olatunji, Performance comparison of SVM and ANN in predicting compressive strength of concrete, IOSR Journal of Computer Engineering 16 (5) (2014) 88–94. https://doi.org/10.9790/0661-16518894
6. J.S. Chou, C.F. Tsai, A.D. Pham, Y.H. Lu, Machine learning in concrete strength simulations: Multi-nation data analytics, Construction and Building Materials 73 (2014) 771– 780. https://doi.org/10.1016/j.conbuildmat.2014.09.054
7. Chen B-T, Chang T-P, Shih J-Y, Wang J-J. Estimation of exposed temperature for fire-damaged concrete using support vector machine. Computational Material Science 2009;44(3):913–20. https://doi.org/10.1016/j.commatsci.2008.06.017
8. Majid A, Khan A, Javed G, Mirza AM. Lattice constant prediction of cubic and monoclinic perovskites using neural networks and support vector regression. Computational Materials Science 2010;50(2):363–72. http://dx.doi.org/10.1016/j.commatsci.2010.08.028
9. Gupta S. Support vector machines based modelling of concrete strength. In: Proceedings of world academy of science: engineering & technology, vol. 36; 2007. https://doi.org/10.1.1.360.1088

10. J. Duan, P.G. Asteris, H. Nguyen, X.N. Bui, H. Moayedi, A novel artificial intelligence technique to predict compressive strength of recycled aggregate concrete using ICA-XGBoost model, Engineering with Computers (2020). https://doi.org/ 10.1007/s00366-020-01003-0
11. J.S. Chou, A.D. Pham, Enhanced artificial intelligence for ensemble approach to predicting high performance concrete compressive strength, Construction and Building Materials 49 (2013) 554–563. https://doi.org/ 10.1016/j.conbuildmat.2013.08.078
12. C. Deepa, K. SathiyaKumari, V.P. Sudha, Prediction of the compressive strength of high performance concrete mix using tree based modeling, International Journal of Computer Applications 6 (5) (2010) 18–24. https:doi.org/ 10.5120/1076-1406
13. H.I. Erdal, Two-level and hybrid ensembles of decision trees for high performance concrete compressive strength prediction, Engineering Applications of Artificial Intelligence 26 (7) (2013) 1689–1697. http://dx.doi.org/10.1016%2Fj.engappai.2013.03.014
14. Behnood, E.M. Golafshani, Machine learning study of the mechanical properties of concretes containing waste foundry sand, Construction and Building Materials 243 (2020) 118152. http://dx.doi.org/10.1016/j.conbuildmat.2020.118152
15. Yan K, Shi C. Prediction of elastic modulus of normal and high strength concrete by support vector machine. Construction and Building Materials 2010;24(8):1479–85. http://dx.doi.org/10.1016%2Fj.conbuildmat.2010.01.006
16. B.A. Young, A. Hall, L. Pilon, P. Gupta, G. Sant, Can the compressive strength of concrete be estimated from knowledge of the mixture proportions?: New insights from statistical analysis and machine learning methods, Cement and Concrete Research- 115 (2019) 379–388. http://dx.doi.org/10.1016/j.cemconres. 2018.09.006
17. Topçu I_B, Sarıdemir M. Prediction of compressive strength of concrete containing fly ash using artificial neural networks and fuzzy logic. Computational Materials Science 2008;41(3):305–11. http://dx.doi.org/10.1016%2Fj.commatsci.2007.04.009
18. Reich Y. Machine learning techniques for civil engineering problems. Computer Aided Civil Infrastructure Engineering 1997;12(4):295–310. http://dx.doi.org/10.1111/0885-9507.00065
19. Dantas ATA, Batista Leite M, de Jesus Nagahama K. Prediction of compressive strength of concrete containing construction and demolition waste using artificial neural networks. Construction Building Material 2013;38:717–22. http://dx.doi.org/10.1016/j.conbuildmat.2012.09.026
20. Min-Chang Kang, Doo-Yeol Yoo, Rishi Gupta (2020), Machine learning-based prediction for compressive and flexural strengths of steel fiber-reinforced concrete, Construction and Building Materials Vol. 266; 121-117 http://dx.doi.org/10.1016/j.conbuildmat.2020.121117
21. Yeh IC. Modeling of strength of high-performance concrete using artificial neural networks. Cement Concrete Research 1998;28(12):1797–808. https://doi.org/10.1016/S0008-8846(98)00165-3
22. Caleb Joshua Lebow, Effect of Cement Content on Concrete Performance, University of Arkansas, Fayetteville. https://scholarworks.uark.edu/etd/3000
23. Xudong Chen, Wanshan Huang & Jikai Zhou (2012), Effect of moisture content on compressive and split tensile strength of concrete, Indian Journal of

Engineering & Materials Sciences, Vol. 19, December 2012, pp. 427-435 http://hdl.handle.net/123456789/15819

24. Rozalija Kozul, David Darwin (1997), Effects of Aggregate Type, Size, and Content on Concrete Strength and Fracture Energy, Structural Engineering and Engineering Materials SM Report No. 43 https://iri.drupal.ku.edu/sites/iri.drupal.ku.edu/files/files/pdf/publications/sm43.pdf
25. Muhsen Salam Mohammed, Salahaldein Alsadey Mohamed, Megat Azmi Megat Johari (2012), Influence of Superplasticizer Compatibility on the Setting Time, Strength and Stiffening Characteristics of Concrete, Advances in Applied Sciences 1(2); 30-36 http://dx.doi.org/10.11648/j.aas.20160102.12
26. A.O. Familusi, B.E. Adewumi, F.I. Oladipo, D.A. Ogundare and J.O. Olusami (2017), Effects of Blast Furnance Slag as a partial replacmenet for Cement in Concrete, 9th International Conference on Sciences, Engineering and Environmental Technology held at Federal Polytechnic Ede, September 2017.
27. Bhanja S, Sengupta B. Investigations on the compressive strength of silica fume concrete using statistical methods. Cement and Concrete Research 2002;32(9):1391–4.
28. Liang H, Song W. Improved estimation in multiple linear regression models with measurement error and general constraint. J. Multivariate. Analysis. 2009; 100(4):726–41. http://dx.doi.org/10.1016/j.jmva.2008.08.003
29. Frosyniotis D, Stafylopatis A, Likas A. A divide-and-conquer method for multi-net classifiers. Pattern Analysis and Applications 2003;6(1):32–40. http://dx.doi.org/10.1007/s10044-002-0174-6
30. Vapnik, V., Golowich, S. E., & Smola, A. J. (1997). Support vector method for function approximation, regression estimation and signal processing, Advances in neural information processing systems (pp. 281-287).
31. Wolpert DH. Stacked generalization. Neural Networks 1992;5(2):241–59. https://doi.org/10.1016/S0893-6080(05)80023-1
32. Breiman L. Bagging predictors. Maching Learning 1996;24(2):123–40.
33. I.N. Da Silva, D.H. Spatti, R.A. Flauzino, L.H.B. Liboni, S.F. dos Reis Alves, Artificial neural networks, Springer International Publishing, Cham, 2017, p. 39.
34. Michael Thomas(2007), Optimizing the Use of Fly Ash in Concrete, University of New Brunswick http://www.cement.org/docs/default-source/fc_concrete_technology/is548-optimizing-the-use-of-fly-ash-concrete.pdf
35. https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength, Compressive Strength Dataset