Research Article – Vol. 02, Issue 03, No. 09

Advanced in Engineering and Intelligence Systems

Journal Web Page: https://aeis.bilijipub.com

Predict the Maximum Dry Density of soil based on Individual and Hybrid Methods of Machine Learning

Ghanshyam G. Tejani[1] , Behnam Sadagat[2,*] , Sumit Kumar[3]

[1] *1Department of Mechanical Engineering, School of Technology, GSFC University, Vadodara, Gujarat, India*

[2] *Department of Civil and Water Engineering, University of Tabriz, Tabriz, Iran*

[3] *Australian Maritime College, College of Sciences and Engineering, University of Tasmania, Launceston, 7248, Australia*

Highlights

- Introduces a novel technique for precise maximum dry density (MDD) forecasting in soil stabilization blends.
- Utilizes Naive Bayes, Artificial Rabbits Optimization, and Gradient-based Optimizer for robust models.
- NBAR model demonstrates exceptional performance with high R2 (0.9903) and low RMSE (34.563).
- A promising method for reliable MDD prediction with implications for robust infrastructure in the construction industry.

Article Info

Received: 31 August 2023
Received in revised: 23 September 2023
Accepted: 28 September 2023
Available online: 30 September 2023

Keywords

Maximum Dry Density;
Naive Bayes;
Artificial Rabbits Optimization;
Gradient-based optimizer;

Abstract

This article introduces a novel technique to accurately forecast the maximum dry density of soil stabilization blends. The Naive Bayes algorithm is employed to develop detailed and accurate models that use various natural soil characteristics, such as particle size distribution, plasticity, linear shrinkage, and stabilizing additives' type and amount, to relate to the MDD of stabilized soil. To ensure the model's accuracy, the study integrates two meta-heuristic algorithms: Artificial Rabbits Optimization and Gradient-based Optimizer. The models undergo validation using MDD samples of various soil types acquired from previously published stabilization test results. The results reveal three distinct models: NBAR, NBGB, and an individual NB model. Among these, the NBAR model stands out with exceptional performance, boasting a high R2 value of 0.9903 and a remarkably low RMSE value of 34.563. These results demonstrate the precision and reliability of the NBAR model and signify its effectiveness in predicting soil stabilization outcomes. Overall, this approach offers a promising way to accurately predict the MDD of soil stabilization mixtures in various engineering applications. Integrating meta-heuristic algorithms into the analysis increases the accuracy of the models and provides more reliable predictions, which has significant implications for the construction industry, where soil stabilization is critical for building robust and long-lasting infrastructure.

Nomenclature

AI	*Artificial intelligence*	*ML*	*Machine Learning*
ARO	*Artificial Rabbits Optimization*	*NB*	*Naive Bayes*
GBO	*Gradient-based Optimizer*	*PI*	*performance index*
LL	*Liquid Limit*	*PI*	*Plasticity Index*
MAE	*Mean Absolute Error*	*PL*	*Plastic Limit*
MAPE	*mean absolute percentage error*	*RMSE*	*Root Mean Square Error*
MDD	*Maximum Dry Density*	R^2	*coefficient of determination*

1. Introduction

Chemical stabilization is a crucial process in that chemicals involve the addition of lime, asphalt, cement, or their combinations to react with the natural constituents of soil, resulting in enhanced strength, changes in porosity, volume, permeability, density, waterproofing, and reduced surface abrasion. Among these, the Maximum Dry Density (MDD) of stabilized soils is particularly important in

* Corresponding Author: Behnam Sadagat
Email: Bhnamsedaghat07@gmail.com

assessing soil suitability for stabilization. MDD represents the quality of stabilized soil after compaction and before curing and serves as a performance criterion for stabilization effectiveness [1]. By establishing such models, one can select a chemical stabilizer appropriately and capture the complex relationships between the properties and influential parameters. While some research studies have investigated changes in material properties before or after stabilization, there have been limited attempts to use these soil properties to predict MDD accurately. To advance the field of chemical stabilization, further research should focus on developing robust predictive models that can aid in optimizing stabilizer selection and enhancing construction practices for various civil engineering projects [2]–[4].

However, whether in their original state or enhanced through techniques like compaction, reinforcement, and consolidation, soil and rock remain indispensable components in the construction field [5]–[7]. Mechanical soil compaction is a widely employed technique to enhance soil engineering properties. Its effectiveness is commonly evaluated by comparing the achieved dry density with the MDD. In construction projects like earth dams, road and railway embankments, landfill liners, and backfilling of retaining structures, comprehending the soil compaction characteristics, particularly MDD, is of utmost importance. MDD plays a critical role as a performance indicator for stabilization and offers valuable insights into the soil quality post-compaction but before any treatment. Understanding the MDD helps ensure the stability and durability of these structures, making it an essential factor in the construction process [8]–[11]. Creating a mathematical model for estimating MDD values is recommended to reduce the requirement for extensive laboratory testing of MDD for each new construction project and to gain insights into the intricate connections between soil properties and influencing factors. This model should consider various soil characteristics before stabilization, such as texture, ductility, linear shrinkage, and the type and quantity of stabilizing additives. Nevertheless, these tests are time-consuming, costly, and heavily reliant on the expertise and experience of operators in both sample collection and result verification [12].

Machine learning (ML) is a potent tool that empowers machines to process data and derive valuable insights, mainly when manual analysis is challenging. The growing availability of vast datasets has spurred the demand for ML applications across diverse industries. The primary objective of ML is to create algorithms capable of learning from data without explicit programming, leading to significant advancements in the field. Mathematicians and programmers employ various methods to tackle this challenge and develop ML algorithms that handle large and intricate datasets [13]–[15]. ML proves especially adept at identifying patterns and making precise predictions. Numerous methodologies have been developed and well-documented, including supervised and unsupervised learning, reinforcement learning, and deep learning. Its widespread adoption in civil engineering, finance, manufacturing, transportation, and other industries has demonstrated remarkable success in enhancing performance and efficiency [16]–[18]. Through the utilization of data, the ML model accurately predicts the MDD of soil. One of the model's most significant advantages is its ability to handle non-linear relationships between input and output variables.

Additionally, the model only requires optimization of a few parameters, effectively mitigating the risk of overfitting. As a result, the model proves to be a reliable and efficient tool for predicting MDD in civil engineering projects. Its extensive use in research and practical applications attests to its credibility and effectiveness [19]–[26].

The current study introduces a novel machine learning method to accurately predict crucial soil properties, specifically focusing on MDD outputs, which play a vital role in civil engineering projects. The study adopts the Naive Bayes (NB) algorithm to overcome the challenges of obtaining experimental data. The key to achieving optimal NB model performance lies in parameter optimization, for which integrating two algorithms, Artificial Rabbits Optimization (ARO) and Gradient-based Optimizer (GBO), proves highly effective. This combination significantly enhances the accuracy and efficiency of the NB model, showcasing its positive impact on the infrastructure sector by optimizing MDD structure design and construction processes. Evaluating the proposed framework with a comprehensive MDD dataset and conducting comparative analyses demonstrate its effectiveness. The study's findings provide valuable insights into efficiently predicting MDD in civil engineering projects, presenting a promising method by incorporating the NB algorithm within the ML approach. This research offers practical solutions and valuable knowledge for tackling MDD prediction critical aspects of soil behavior in civil engineering projects.

2. Materials and Methodology

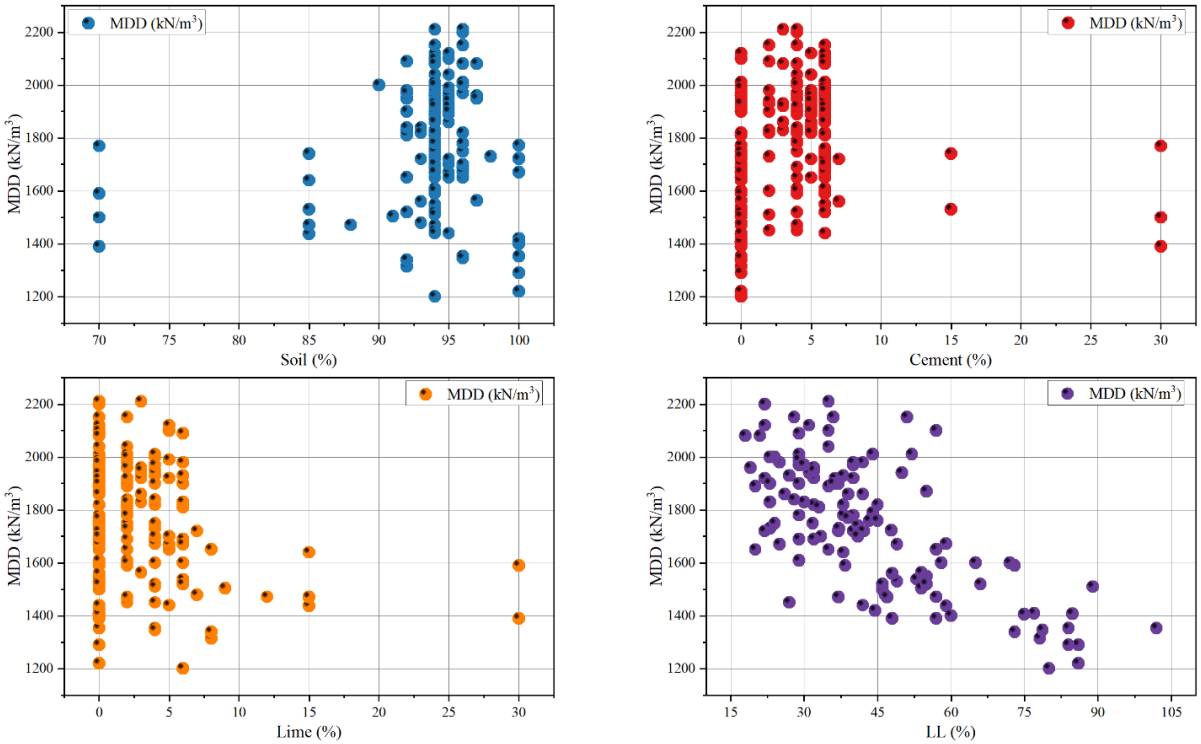
2.1. Data gathering

A more advanced approach has been developed to estimate soil's Maximum Dry Density (MDD), which involves six variables. The six variables used for this estimation are the percentages of cement and lime, as well

as the Atterberg Limits test results, which include the Liquid Limit (LL), Plasticity Index (PI), and Plastic Limit (PL). The MDD prediction is based on an NB model, and soil samples are obtained from different sources and analyzed in the laboratory [7]. Table 1 provides information about the sample origin, quantity, and variable range. LL indicates the moisture content at which the soil transitions from a plastic to a liquid state, PL represents the moisture content at which the soil transitions from a plastic to a semi-solid state, and PI measures the soil's plasticity, which is the difference between LL and PL. By considering the properties of the soil and the percentages of cement, lime, LL, PL, and PI obtained from collected data, custom equations, and correlations are employed to forecast the MDD of the soil. These equations and correlations are specifically designed to deliver accurate predictions of MDD based on the available information. In addition, the relation and correlations between input and output parameters are illustrated in Figure 1 and Table 2, respectively. According to Figure 1 and Table 2, if the PL, Lime, and cement possess a lower ratio, it will result in higher MDD values. A higher Soil, LL, and PI amounts can increase the values of MDD. In summary, the input parameters of Soil, Cement, LL, PL, PI, and Lime can all impact the elastic modulus of RAC. The desired properties and performance of the MDD will be achievable by optimizing these parameters.

Table 1. Statistical properties of inputs and MDD.

Indicators	Variables						
	Input						Targets
	Soil (%)	Cement (%)	Lime (%)	LL (%)	PL (%)	PI (%)	MDD ($kN/m3$)
Max	100	30	30	102	58.24	70	2210
Min	70	0	0	18	12	0	1200
Avg	93.604	3.807	2.588	39.428	22.673	16.755	1780.61
St. Dev.	4.6366	4.316	4.086	16.763	9.412	12.694	227.508



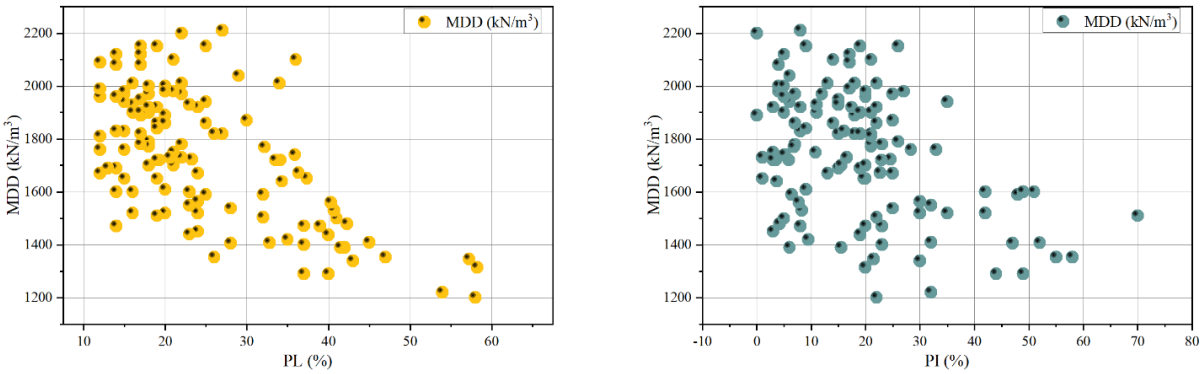


Fig. 1. The scatter plot for the relation between input and output.

Table. 2. The correlation for input and output.

	Soil (%)	Cement (%)	Lime (%)	LL (%)	PL (%)	PI (%)	MDD
Soil	1						
Cement	-0.58545	1					
Lime	-0.51634	-0.39199	1				
LL	0.036907	-0.16005	0.127181	1			
PL	-0.24418	0.025072	0.250608	0.660584	1		
PI	0.229777	-0.22993	-0.01786	0.830769	0.130912	1	
MDD	0.11241	0.119095	-0.25336	-0.66095	-0.57443	-0.4469	1

2.2. *Naive Bayes (NB)*

The *NB* classifier is a probabilistic classifier that utilizes Bayes' theorem while assuming strong independence between features. Its primary advantage is its simplicity in design, as it does not necessitate complex iterative parameter estimation methods. Furthermore, the *NB* classifier is not easily affected by noise or irrelevant attributes, as stated by Das et al. [27]. The *NB* classifier is based on the following equation:

$$y = \underset{y_i=\{landslide,non-landslide\}}{\arg\max} P(y_i) \prod_{i=1}^{14} P\left(\frac{x_i}{y_i}\right) \quad (1)$$

where $P(y_i)$ is the prior probability of y_i, $P(\frac{x_i}{y_i})$ is the posterior probability, and it can be calculated by:

$$P\left(\frac{x_i}{y_i}\right) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(x_i-\mu)^2}{2\sigma^2}} \quad (2)$$

Where μ is the mean, and σ is the standard deviation of x_i.

2.3. *Artificial Rabbits Optimization (ARO)*

The ARO algorithm drew inspiration from the survival tactics adopted by rabbits in their natural habitat. The detour foraging strategy, where rabbits venture away from their nests to seek food, was the basis for this approach [28]. To evade predators and hunters, rabbits construct burrows near their nests. They tend to search for food if they possess high energy levels or ample. Rabbits tend to forage for food at distant locations from their nests when they possess high or sufficient energy levels (detour foraging). Conversely, when their energy levels are low, they randomly take shelter in nearby burrows around their nests. Figure 2 illustrates the process of ARO [29].

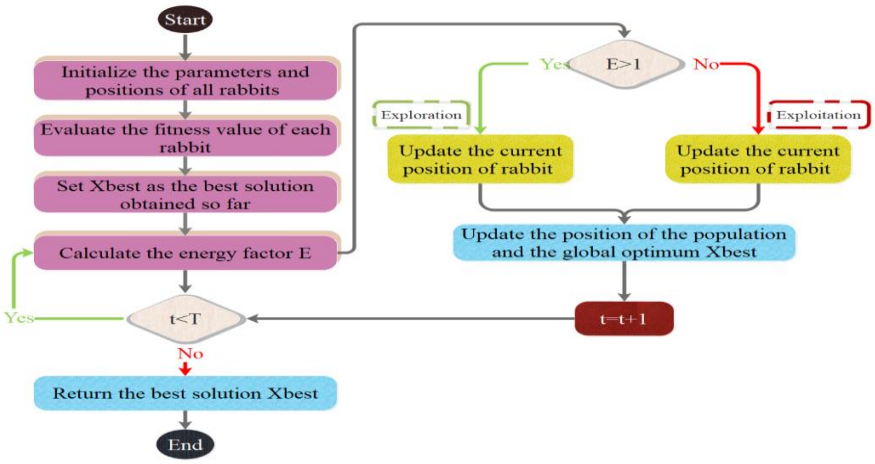


Fig. 2. The process of ARO

2.3.1 *Energy Shrink (Switch between Exploration and Exploitation)*

Rabbits can either engage in random hiding or detour foraging, contingent on their energy levels. To simulate the rabbit's decision, an energy factor A(t) is computed using Eq. (3). If $A(t)$ exceeds 1, the rabbit will opt for detour foraging, whereas if A(t) is less than or equal to 1, it will choose random hiding.

$$A(t) = 4\left(1 - \frac{t}{T}\right) ln\frac{1}{r} \tag{3}$$

Here, r is a randomly selected number between 0 and 1.

2.3.2 *Detour Foraging (Exploration)*

As per Eq. (4), rabbits randomly search for food based on the positions of their peers while foraging for food at a distance from their nests to protect them from potential predators.

$$\vec{S_i}(t+1) = \vec{x_j}(t) + U \times \left(\vec{x_i}(t) - \vec{x_j}(t)\right) + round(0.5 \times (0.05) + r_1)) \times n_1, i, j = 1, \dots, n \text{ and } j \neq i \tag{4}$$

$$U = O \times C \tag{5}$$

$$O = (e - e^{\left(\frac{t-1}{T}\right)} \times \sin(2\pi r_2) \tag{6}$$

$$C(k) = \begin{cases} 1 & if\ k = g(O) \\ 0 & else \end{cases} \quad k = 1, \dots, d \text{ and } l = 1, \dots, [r_3, d] \tag{7}$$

$$g = randperm(d) \tag{8}$$

$$n_1 \sim N(0,1) \tag{9}$$

The equation involves several parameters. At time t, $\vec{x_i}(t)$ denotes the position of the ith rabbit, while $\vec{S_i}(t+1)$ represents the candidate's position at time $t+1$. The maximum number of iterations is T, and the movement pace of rabbits is represented by L. The rabbit population size is denoted by n, and d represents the number of variables in the optimization problem. Additionally, r_1, r_2, and r_3 are three random numbers between (0,1) and n_1 follows the standard normal distribution. The mapping vector is C, and the running operator that simulates the rabbits' running characteristics is denoted by U.

2.3.3 *Random Hiding (Exploitation)*

Eq. (10) generates a set of d burrows around the nest of each rabbit. The rabbit randomly selects these burrows to take shelter and evade potential predators.

$$\overrightarrow{b_{i,j}}(t) = \vec{x_i}(t) + H \times g \times \vec{x_i}(t), i = 1, \dots, n \text{ and } j = 1, \dots, d \tag{10}$$

$$H = \frac{T - t + 1}{T} \times r_4 \tag{11}$$

$$g(k) = \begin{cases} 1 & if\ k = j \\ 0 & else \end{cases} \quad k = 1, \dots, d \tag{12}$$

$$\vec{S_i}(t+1) = \vec{x_j}(t) + U \times \left(r_4 \times \overrightarrow{b_{i,r}}(t) - \vec{x_i}(t)\right) i = 1, \dots, n \tag{13}$$

$$gr(k) = \begin{cases} 1 & if\ k = |r_5 \times d| \\ 0 & else \end{cases} \quad k = 1, \dots, d \tag{14}$$

$$\overrightarrow{b_{i,r}}(t) = \vec{x_i}(t) + H \times gr \times \vec{x_i}(t), i = 1, \dots, n \tag{15}$$

$$\vec{x_i}(t+1) = \begin{cases} \vec{x_i}(t) f(\vec{x_i}(t)) \leq f(\vec{s_i}(t+1)) \\ \vec{s_i}(t+1) f(\vec{x_i}(t)) > f(\vec{s_i}(t+1)) \end{cases} \quad k = 1, \dots, d \tag{16}$$

Eq. (15) highlights $\overrightarrow{b_{i,r}}(t)$ as the randomly chosen burrow for the ith rabbit to take shelter in, where H represents the hiding parameter, $\overrightarrow{b_{i,j}}$ denotes the jth burrow for the ith rabbit, and r_4, and r_5 are random numbers between (0,1).

2.4. *Gradient-based optimizer (GBO)*

The *GBO* is a technique that combines population-based methods with a gradient technique to address complex optimization problems. The *GBO* algorithm leverages *Newton's* method to guide the search agents' direction as they navigate through the problem space. Its

fundamental components include the gradient search rule and the locale escaping operator [30].

2.4.1 Initialization stage

The GBO, similar to many other metaheuristic algorithms, commences the optimization procedure by generating an initial population from a uniform random distribution. Each agent in the population is referred to as a "$vector$", and the population consists of N agents represented as vectors in a search space with $D-dimensions$. The initialization process is executed in the following manner:

$$X_n = X_{min} + rand(0,1) \times (X_{max} - X_{min}) \quad (17)$$

where X_{min}, and X_{max} are represent the limits of decision variables X, and $rand(0,1)$ is a random number defined in the range $[0, 1]$.

2.4.2 Gradient search rule stage

The GBO algorithm, which was previously discussed, starts by generating a $random$ set of initial solutions. Subsequently, the position of each agent gets updated according to a designated gradient direction. To ensure a balance between exploring essential regions of the search space and exploiting them to reach optimal and global points, a significant factor ρ_1 is utilized in the following manner:

$$\rho_1 = 2 \times rand \times \alpha - \alpha \quad (18)$$

$$\alpha = \left| \beta \times sin\left(\frac{3\pi}{2} + sin\left(\beta \times \frac{3\pi}{2}\right)\right)\right| \quad (19)$$

$$\beta = \beta_{min} + (\beta_{max} - \beta_{min}) \times \left(1 - \left(\frac{m}{M}\right)^3\right)^2 \quad (20)$$

where β_{min} and β_{max} are values of constant 0.2 and 1.2, respectively, m indicates the present iteration number, while M represents the total number of iterations. To achieve a balance between exploration and exploitation, the parameter ρ_1 based on changes in the sine function α. The parameter value varies during the iterations. Initially, it is set to a significant value during the early optimization iterations to enhance population diversity. As the iterations progress, the value gradually decreases, speeding up the convergence of the population. Within a specified range $[550, 750]$, the parameter value increases during specific iterations, promoting solution diversity and leading the algorithm to converge around the best solution found and explore additional solutions. Consequently, this approach helps the algorithm avoid getting stuck in local sub-regions. In summary, GSR can be defined as follows:

$$GSR = randn \times \rho_1 \times \frac{2\Delta x \times x_n}{(x_{worst} - x_{best} + \varepsilon)} \quad (21)$$

The GBO algorithm benefits from the concept of GSR as it enables random behavior in each iteration, which enhances exploration behavior and prevents getting stuck in local optima. The formula in Eq. (21) includes a factor called Δx, which calculates the difference between the best solution (x_{best}) and a solution selected randomly x_{r1}^m. To ensure that Δx varies across iterations, the value of δ is determined using Eq. (24). Moreover, to further promote exploration, some random $(randn)$ is incorporated into this equation.

$$\Delta x = rand(1{:}N) \times |step| \quad (22)$$

$$step = \frac{(x_{best} - x_{r1}^m) + \delta}{2} \quad (23)$$

$$\delta = 2 \times rand \times \left(\left|\frac{x_{r1}^m + x_{r2}^m + x_{r3}^m + x_{r4}^m}{4} - x_n^m\right|\right) \quad (24)$$

The vector $rand(1{:}N)$ contains N random values that fall within the range of $[0, 1]$. Four different integers, $r_1, r_2, r_3, and\ r_4$, are randomly selected from the range of $[1, N]$, with the condition that $(r_1 \neq r_2 \neq r_3 \neq r_4 \neq n)$. The step size, which is determined by x_{best} and x_{r1}^m, is denoted as "$step$". The movement direction (DM) is used to approach the solution area around x_n. To achieve this, the current vector (x_n) is moved in the direction of $(x_{best} - x_n)$ using the best vector. This process significantly affects the GBO convergence by providing a useful local search tendency. The formula for computing the DM is as follows:

$$DM = rand \times \rho_2 \times (x_{best} - x_n) \quad (25)$$

The value of $rand$ is generated uniformly within the range $[0, 1]$, and ρ_2 is a random parameter used to adjust the step size of each vector agent during the GBO exploration process ρ_2 is composed of important parameters and is calculated according to the following formula:

$$\rho_2 = 2 \times rand \times \alpha - \alpha. \quad (26)$$

The current vector position (x_n^m) is updated using Eqs. (27) and (28), which depend on the terms GSR and DM.

$$X1_n^m = x_n^m - GSR + DM \quad (27)$$

The new vector, $X1_n^m$, is obtained by updating x_n^m. This can be expressed as a reformulation of Eqs. (20) and (25):

$$X1_n^m = x_n^m - randn \times \rho_1 \times \frac{2\Delta x \times x_n^m}{(yp_n^m - yq_n^m + \varepsilon)} + randn \times \rho_2 \times (x_{best} - x_n^m) \quad (28)$$

The values of yp_n^m and yq_n^m correspond to $y_n + \Delta x$ and $y_n - \Delta x$, respectively. The vector y_n is the average of the current solution vector, x_n, and the vector z_{n+1}, and is calculated as follows:

$$z_{n+1} = x_n - randn \times \frac{2\Delta x \times x_n}{(x_{worst} - x_{best} + \varepsilon)} \quad (29)$$

The current solution vector is represented by x_n, while $randn$ is a randomly generated solution vector of size n. x_{worst} and x_{best} correspond to the $worst$ and $best$ solutions,

respectively, and Δx is determined by Eq. (22). By substituting the current solution vector, x_n^m, for the best solution vector, x_{best} in the formula mentioned above, obtain $X2_n^m$, which is expressed as follows:

$$X2_n^m = x_{best} - randn \times \rho_1 \times \frac{2\Delta x \times x_n^m}{(yp_n^m - yq_n^m + \varepsilon)} + randn \times \rho_2 \times (x_{r1}^m - x_{r2}^m) \quad (30)$$

The objective of the GBO algorithm is to *improve both* the exploration and exploitation phases by utilizing Eq. (28) to enhance global search during the exploration phase, and Eq. (30) to boost local search capability during the exploitation phase. As a result, the next iteration's updated solution is created in the following manner:

$$x_n^{m+1} = r_a \times (r_b \times X1_n^m + (1 - r_b) \times X2_n^m) + (1 - r_a) \times X3_n^m \quad (31)$$

where r_a, and $r_b$are random numbers determined in the range $[0, 1]$, and $X3_n^m$ is defined as:

$$X3_n^m = X_n^{m+1} - \rho 1 \times (X2_n^m - X1_n^m) \quad (32)$$

2.4.3 Local escaping operator stage

To enhance the efficacy of an optimization algorithm in addressing intricate problems, the Local Escaping Operator (LEO) is introduced. By updating the solution's position, the LEO is adept at helping the algorithm escape local optima points and accelerate convergence. The LEO employs targets to create a new solution, X_{LEO}^m, which outperforms several other solutions, including the best solution, x_{best}, as well as solutions $X1_n^m$ and $X1_n^m$ selected randomly from the population, as well as solutions X_{r1}^m and $X1_{r2}^m$ created randomly. The LEO efficiently updates the current solutions, and this operation follows a specific scheme.

$$If\ rand < prX_{LEO}^m = \begin{cases} X_n^{m+1} + f_1(u_1 x_{best} - u_2 x_k^m) + f_2 \rho 1\left(u_3(X2_n^m - X1_n^m)\right) + \dfrac{u_2(X2_{r1}^m - X1_{r2}^m)}{2}, & if\ rand < 0.5 \\ X_n^{m+1} + f_1(u_1 x_{best} - u_2 x_k^m) + f_2 \rho 1\left(u_3(X2_n^m - X1_n^m)\right) + \dfrac{u_2(X2_{r1}^m - X1_{r2}^m)}{2}, & otherwise \\ End \end{cases} \quad (33)$$

Where pr is a chance value, $pr = 0.5$, the values f_1, and f_2 are random numbers following a uniform distribution $\in [-1, 1]$, and u_1, u_2, u_3 are random values generated as follows:

$$u_1 = \begin{cases} 2 \times rand & if \mu_1 < 0.5 \\ 1 & otherwise \end{cases} \quad (34)$$

$$u_2 = \begin{cases} rand & if \mu_1 < 0.5 \\ 1 & otherwise \end{cases} \quad (35)$$

$$u_3 = \begin{cases} rand & if \mu_1 < 0.5 \\ 1 & otherwise \end{cases} \quad (36)$$

Where $rand$ denotes a random value $\in [0, 1]$, and μ_1 is a number within the specified range $[0, 1]$. The previous equations for u_1, u_2, u_3, can be explained as follows:

$$u_1 = L_1 \times 2 \times rand + (1 - L_1) \quad (37)$$

$$u_2 = L_1 \times rand + (1 - L_1) \quad (38)$$

$$u_3 = L_1 \times rand + (1 - L_1) \quad (39)$$

The binary parameter L_1 can take on values of either $0\ or\ 1$. If the parameter μ_1 is less than 0.5, then L_1 is assigned a value of 1; $otherwise, L_1$ is assigned a value of 0. The solution x_k^m is created using the following method:

$$x_k^m = \begin{cases} x_{rand} & if \mu_2 < 0.5 \\ x_p^m & otherwise \end{cases} \quad (40)$$

The solution generated by x_{rand} is a result of a formula that produces random outputs.

$$x_{rand} = X_{min} + rand(0,1) \times (X_{max} - X_{min}) \quad (41)$$

A solution called x_p^m is chosen at random from a given population, and a random number between $0\ and\ 1$, denoted by μ_2, is also selected.

Algorithm 1 describes the details $pseudo - code$ of GBO algorithm.

Algorithm 1 The Gradient-Based Optimizer's Pseudo Code.

Step 1. *Initialization process*
Provide values to parameters pr, ε, M
Formulate an initial population $X_0 = [x_{0,1}, x_{0,2}, \dots, x_{0,D}$
Evaluate the objective function value $f(X_0) =, n = 1, \dots, N$
Determine the best and worst outcomes x_{best}^m and x_{worst}^m
Step 2. *Primary iteration*
while $m < M$ ***do***
  for $n = 1\ to\ N$ ***do***
    for $n = 1\ to\ D$ ***do***
      Select randomly $r1 \neq r2 \neq r3 \neq r4 \neq n$ *in the range of* $[1, N]$
      Determine the position $x_{n,i}^{m+1}$using $Eq.$ (31)
    end for
    if $rand < pr$ ***then***
      Determine the position x_{LEO}^m using $Eq.$ (33)
      $x_n^{m+1} = x_{LEO}^m$
    end if
    Modify the positions x_{best}^m and x_{worst}^m
  end for
  $m = m + 1$
end while
Step 2. *Return* x_{best}^m

2.5. Performance evaluation methods

Different criteria for evaluating hybrid models based on their error level and correlation were presented in this

section. The evaluation metrics discussed here include Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Coefficient Correlation (R^2), mean absolute percentage error (MAPE), scattered index (SI), and performance index (PI). The formulas for each of these metrics are provided below. An algorithm with a high R^2 value close to 1 is considered to perform well in both the training and testing stages, while lower values of metrics such as RMSE, MAPE, and MSE are preferred as they indicate a lower error level in the model.

$$R^2 = \left(\frac{\sum_{i=1}^{w}(p_i - \bar{p})(q_i - \bar{q})}{\sqrt{[\sum_{i=1}^{w}(p_i - p)^2][\sum_{i=1}^{w}(q_i - \bar{q})^2]}} \right)^2 \quad (42)$$

$$RMSE = \sqrt{\frac{1}{w}\sum_{i=1}^{w}(q_i - p_i)^2} \quad (43)$$

$$MSE = \frac{1}{w}\sum_{i=1}^{w} q_i^2 \quad (44)$$

$$MAPE = \frac{100}{W}\sum_{i}^{w}\frac{|q_i|}{|p_i|} \quad (45)$$

$$PI = \frac{1}{\bar{\bar{q}}}\frac{RMSE}{\sqrt{R^2 + 1}} \quad (46)$$

$$SI = \frac{RMSE}{\bar{q}} \quad (47)$$

$Eqs.(42-47)$ use the variables w to indicate the number of samples, p_i to represent the predicted value, $\bar{p}$ and $\bar{q}$ to denote the mean predicted and measured values, respectively, and q_i to indicate the measured value.

3. Results and discussion

3.1. Results of Hyperparameters

In machine learning, hyperparameters are pre-defined parameters before training and remain constant during training. This research set up the optimizers by specifying hyperparameters such as alpha and binarize. By tuning these hyperparameters effectively, enhancing the optimizer's performance and avoiding problems such as underfitting or overfitting the model is feasible. Consequently, selecting suitable hyperparameters is essential in building a dependable and precise machine-learning model.

Table 2. The hyperparameters of developed models.

HPC Features	models	Hyperparameter	
		alpha	binarize
MDD	NBAR	4	55
	NBGB	1	667

Table 3. The obtained results for the predicted value.

Phase	Models	Evaluators					
		RMSE	R2	MSE	MAPE	PI	SI
Train	NB	83.694	0.9651	7004.8	3.84	0.023	0.047
	NBAR	34.563	0.9903	1194.6	1.65	0.009	0.019
	NBGB	46.968	0.9764	2206	2.20	0.013	0.0264
Validation	NB	62.186	0.9044	3867.1	2.69	0.017	0.035
	NBAR	31.006	0.9733	961.42	1.43	0.008	0.017
	NBGB	49.340	0.9281	2434.5	2.33	0.014	0.028
Test	NB	94.399	0.9682	8911.3	4.49	0.027	0.053
	NBAR	41.358	0.9899	1710.5	2.09	0.012	0.023
	NBGB	59.084	0.9844	3491	2.97	0.017	0.033

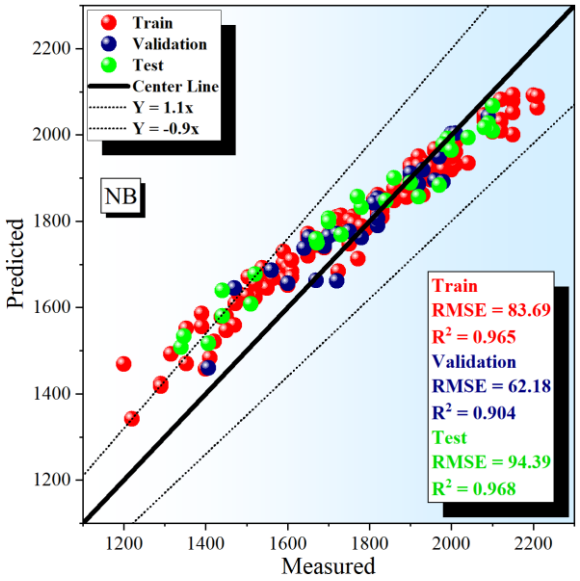
3.2. *Comparing models' performance*

The study aimed to predict MDD using three models, NB, NBAR, and NBGB, and evaluate their performance against experimental measurements during the training, validation, and testing phases. The experimental data were divided into training, validation, and testing sets to ensure an unbiased evaluation. The study used six statistical metrics, including R^2, RMSE, MAPE, PI, SI, and MAE, to comprehensively evaluate and compare the algorithms used in this study. The models were evaluated based on their R^2 values, which indicate the amount of variance in the dependent variable that the independent variable can explain. The NBAR model exhibited excellent predictive accuracy, with the highest R^2 values of 0.9903, 0.9733, and 0.9682 in the training, validation, and testing phases, respectively, while the NB model showed slightly lower R^2 values of 0.9651, 0.9044, and 0.9899, respectively. Additionally, the study analyzed other error indicators such as RMSE, which ranged from 31.006 to 83.694, with the NBAR model showing the fewest errors in the validation phase and the NB model showing the most errors in the training phase. The MAPE value was also analyzed, with the NBAR model showing the lowest value of 1.43 in the validation phase, indicating the most suitable modeling,

while the lowest value of 4.49 belonged to the NB training phase.

Considering the results, it can be concluded that the NBAR model outperformed the NB and NBGB models in all three stages. However, other factors such as model complexity, computational efficiency, and ease of implementation should also be considered when selecting a model for real-world applications. Overall, the results indicate that ARO optimization improved NB's ability to predict MDD. Using the NBAR model for predicting MDD in real-world applications could be a viable and reliable option.

Figure 3 presents a scatter plot for the correlation between measured and predicted MDD values, where the R^2 and RMSE metrics influence the point distribution. Higher R^2 values lead to dispersed points, while lower RMSE values result in denser points. The current illustration includes various elements, such as the central line at Y=X, a linear regression model, and two lines positioned below and above the central line at Y=0.9X and Y=1.1X. When the upper and lower endpoints of these lines intersect, it results in incorrect predictions of values exceeding or falling short of the actual values. The plot indicates that NBAR has high precision, exhibiting less dispersion across all phases. In contrast, NB shows more excellent dispersion, leading to overestimating and underestimating, resulting in less accurate predictions than NBAR. Overall, NBAR is more suitable than other hybrid models for accuracy and performance during the training, validation, and testing phases.



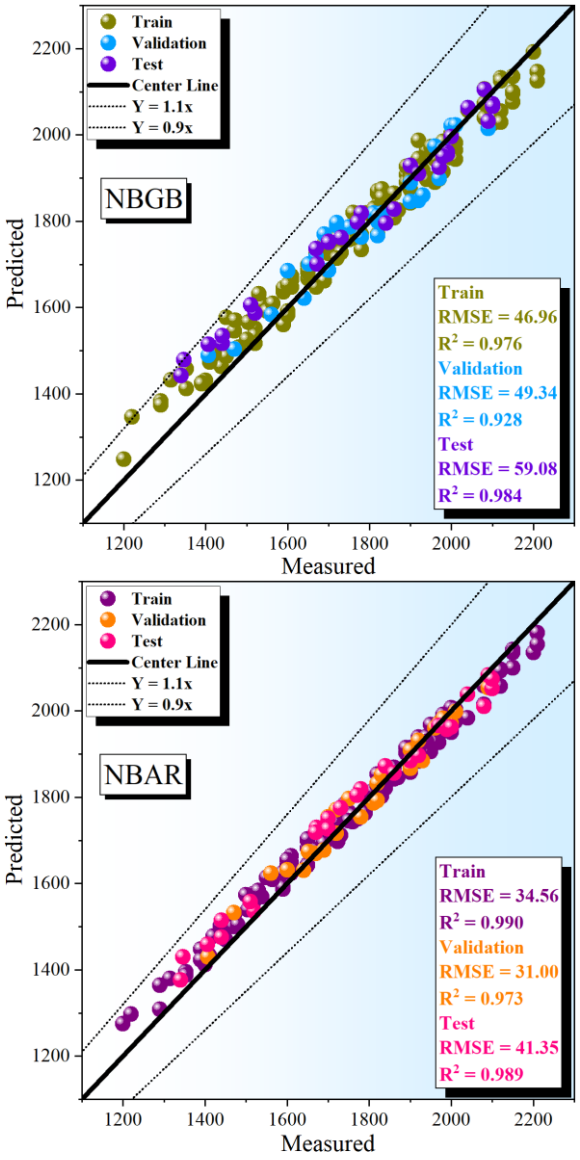


Fig. 3. The scatter plot for the correlation between predicted and measured MDD.

Figure 4 compares the predicted and measured samples, categorized into three parts: training, validation, and testing. The analysis of the NBAR model reveals a slight dissimilarity between the measured values in the training and testing phases, with the latter being higher in most instances. Likewise, the predicted points in NBGB exhibit a minor difference from the measured values, although its precision is inferior to NBAR. The NB model performs less effectively than the other two models and shows a relatively considerable variance.

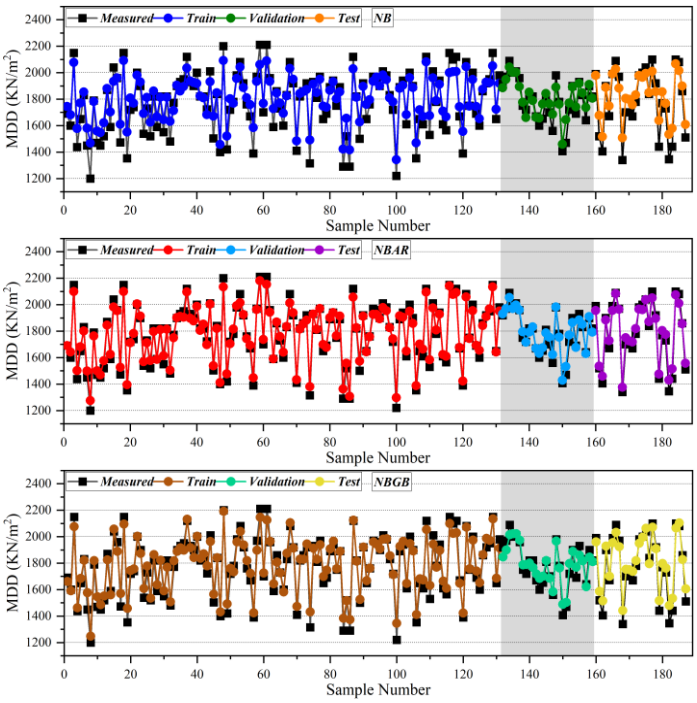
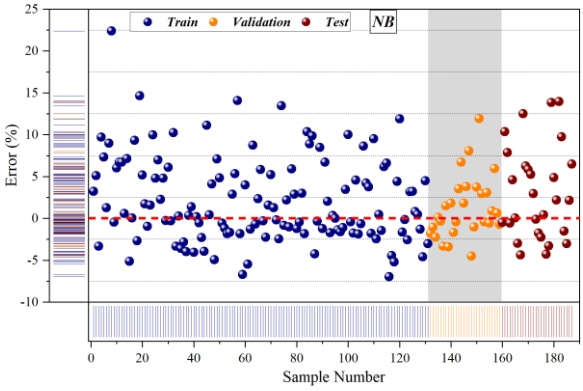


Fig.4. The comparison between predicted and measured values.

As depicted in Figure 5, the NB model exhibits the highest scatter percentage and dispersion among all the models. The introduction of these optimizers had a positive impact on the model's performance. Upon comparing the obtained models, NBAR stood out with the lowest dispersion across all phases. Notably, it displayed the most promising results during the initial evaluation, achieving a testing error rate of merely 7%. In contrast, NBGB showed a higher testing error rate of 10%. Finally, the NB model had a training error rate of 22%. These findings emphasize the significance of optimization techniques in enhancing model performance, with NBAR emerging as the most effective model due to its superior performance in both dispersion reduction and error rate minimization.



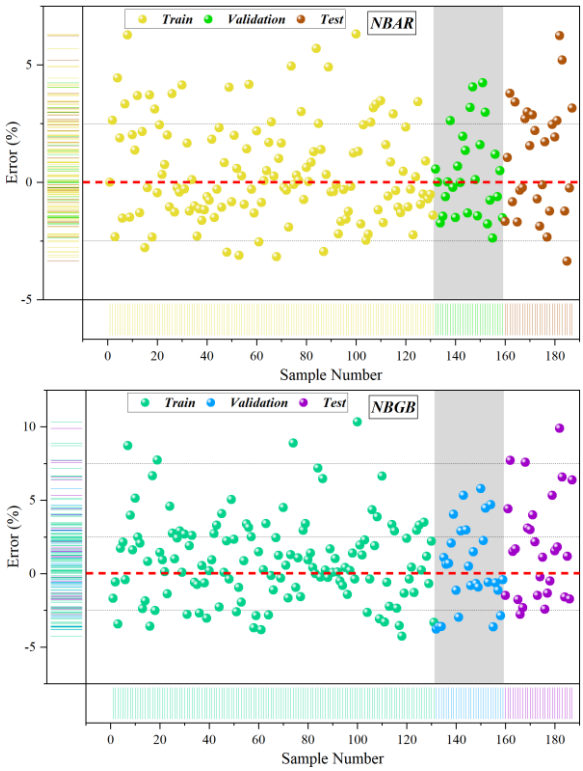


Fig. 5. The error percentage of developed models.

Figure 6 showcases a violin diagram that presents the models' error percentages. According to the diagram, NBAR demonstrated an average error rate of 5%, with minimal dispersion and a sharp normal distribution. In contrast, NB exhibited dispersion in all three phases, with a flatter normal distribution. However, it achieved its highest accuracy with an error rate below 20%. While NBGB had the most significant and diverse errors, an outlier data point exceeding 10% of the data was observed. NB had a more widely dispersed distribution than the other two models, with a lower frequency of around zero.

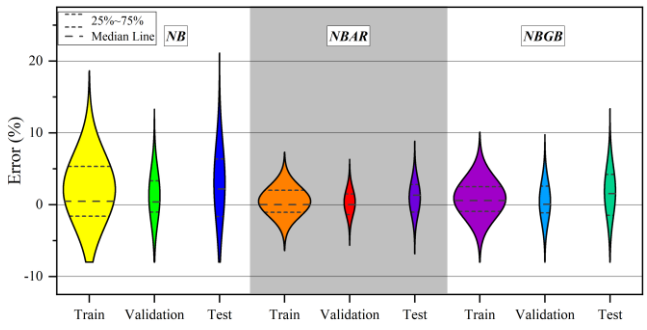


Fig. 6. The violin diagram for the error percentage of presented models.

The exhibition of Taylor diagrams for the applied NB, NBAR, and NBGB predictive models is shown in Figure 7. This diagram is a statistical summary of the observed and predicted MDD, which combines the root mean square errors (RMSE), correlation coefficients (CC), and the normalized standard deviations. From Figure 7, the NBAR

model (Combination of the NB model and ARO optimizer) is considered the best predictive model, and its results are closest to the ideal benchmark observed experimental data.

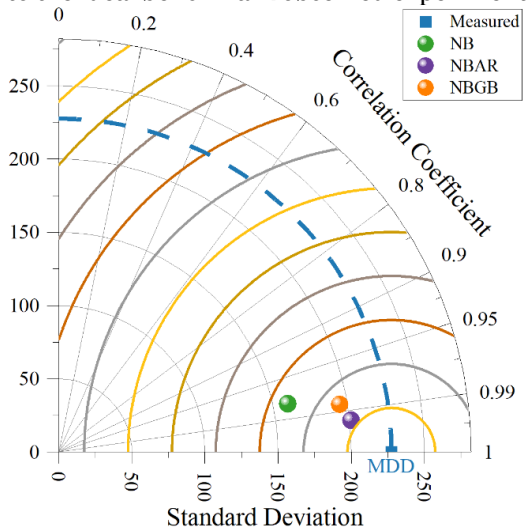


Fig. 7. The Taylor diagram for the developed models

4. Conclusion

The stability of soil depends significantly on its maximum dry density (MDD), which is influenced by factors such as soil type, compaction effort, and moisture content. Ensuring an accurate measurement of MDD is essential in engineering and construction to guarantee the safety and durability of structures. This study proposed a machine learning model based on the NB algorithm to predict MDD. To enhance the model's accuracy and minimize errors, ARO and GBO meta-heuristic algorithms were employed, resulting in the development of three models: NBAR, NBGB, and an individual NB model. Laboratory samples from published articles were utilized in the training, validation, and testing phases to validate these models. Several evaluation metrics, including R^2, RMSE, MSE, PI, and MAPE, were used to compare the performance of the models. The study's findings revealed that the NBAR models achieved the highest R^2 values, while the NB model exhibited the lowest R^2 value, with a difference of 3%. Throughout all three phases, NBAR consistently outperformed other methods in accurately predicting MDD with high precision. This superiority was evident from the significantly lower error rates, with a 62% lower RMSE and an 89% lower MSE than NB. While the performance of NB and NBGB was weaker than NBAR based on all statistical indices, their results were still deemed acceptable with values of criteria assessments. In contrast, the NBAR model displayed the most appropriate performance in the training, validation, and testing stages. Overall, machine learning models can be relied upon as an alternative to experimental methods for evaluating MDD, leading to significant time and energy savings. The research highlighted the effectiveness of combining the ARO optimizer with NB, resulting in a favorable combination that led to accurate predictions of MDD.

COMPETING OF INTERESTS

The authors declare no competing interests.

AUTHORSHIP CONTRIBUTION STATEMENT

Behnam Sadagat: Writing-Original draft preparation, Conceptualization, Supervision.

Ghanshyam G: Project administration, Methodology.

Sumit Kumar: Software, Validation.

DATA AVAILABILITY STATEMENT

Some or all data, models, or codes that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Preethi, R. B. Tangadagi, M. Manjunatha, and A. Bharath, "Sustainable effect of chemically treated aggregates on bond strength of bitumen," *J. Green Eng*, vol. 10, no. 9, pp. 5076–5089, 2020.

[2] Z. S. Janjua and J. Chand, "Correlation of CBR with index properties of soil," *International Journal of Civil Engineering and Technology*, vol. 7, no. 5, pp. 57–62, 2016.

[3] J. Duque, W. Fuentes, S. Rey, and E. Molina, "Effect of grain size distribution on california bearing ratio (CBR) and modified proctor parameters for granular materials," *Arabian Journal for Science and Engineering*, vol. 45, pp. 8231–8239, 2020.

[4] S. N. Mehdi Yaltaghian Khiabani1, Behnam sedaghat 2, Parisa Ghorbanzadeh3, Negin Porroustami4, Seied Mehdy Hashemy Shahdany5, Yousef Hassani6, "Application of a Hybrid Hydro-economic Model to Allocate Water over the Micro- and Macro-scale Region for Enhancing Socioeconomic Criteria under the Water Shortage Period," *Water Economics and Policy*, 2023.

[5] E. G. Akpokodje, "The stabilization of some arid zone soils with cement and lime," *Quarterly journal of engineering geology*, vol. 18, no. 2, pp. 173–180, 1985.

[6] F. G. Bell, "Lime stabilization of clay minerals and soils," *Engineering geology*, vol. 42, no. 4, pp. 223–237, 1996.

[7] A. H. Alavi, A. H. Gandomi, M. Gandomi, and S. S. Sadat Hosseini, "Prediction of maximum dry density and optimum moisture content of stabilised soil using RBF neural networks," *The IES Journal Part A: Civil & Structural Engineering*, vol. 2, no. 2, pp. 98–106, 2009.

[8] A. B. Ngowi, "Improving the traditional earth construction: a case study of Botswana," *Construction and Building Materials*, vol. 11, no. 1, pp. 1–7, 1997.

[9] C. KS, Y. M. Chew, M. H. Osman, and M. G. SK, "Estimating maximum dry density and optimum moisture content of compacted soils," in *international conference on advances in civil and environmental engineering*, 2015,

pp. 1–8.
[10] A. Bharath, M. Manjunatha, T. V Reshma, and S. Preethi, “Influence and correlation of maximum dry density on soaked & unsoaked CBR of soil,” *Materials Today: Proceedings*, vol. 47, pp. 3998–4002, 2021.
[11] S. Suman, M. Mahamaya, and S. K. Das, “Prediction of maximum dry density and unconfined compressive strength of cement stabilised soil using artificial intelligence techniques,” *International Journal of Geosynthetics and Ground Engineering*, vol. 2, pp. 1–11, 2016.
[12] A. Hossein Alavi, A. Hossein Gandomi, A. Mollahassani, A. Akbar Heshmati, and A. Rashed, “Modeling of maximum dry density and optimum moisture content of stabilized soil using artificial neural networks,” *Journal of Plant Nutrition and Soil Science*, vol. 173, no. 3, pp. 368–379, 2010.
[13] Q. Zhang and M. Afzal, “Prediction of the elastic modulus of recycled aggregate concrete applying hybrid artificial intelligence and machine learning algorithms,” *Structural Concrete*, vol. 23, no. 4, pp. 2477–2495, Aug. 2022, doi: 10.1002/suco.202100250.
[14] F. Masoumi, S. Najjar-Ghabel, A. Safarzadeh, and B. Sadaghat, “Automatic calibration of the groundwater simulation model with high parameter dimensionality using sequential uncertainty fitting approach,” *Water Supply*, vol. 20, no. 8, pp. 3487–3501, Dec. 2020, doi: 10.2166/ws.2020.241.
[15] L. Huang, W. Jiang, Y. Wang, Y. Zhu, and M. Afzal, “Prediction of long-term compressive strength of concrete with admixtures using hybrid swarm-based algorithms,” *Smart Structures and Systems*, vol. 29, no. 3, pp. 433–444, 2022.
[16] A. H. Alavi, A. H. Gandomi, and A. Mollahasani, “A genetic programming-based approach for the performance characteristics assessment of stabilized soil,” *Variants of evolutionary algorithms for real-world applications*, pp. 343–376, 2012.
[17] W. Z. Taffese and K. A. Abegaz, “Prediction of compaction and strength properties of amended soil using machine learning,” *Buildings*, vol. 12, no. 5, p. 613, 2022.
[18] S. K. Das, P. Samui, and A. K. Sabat, “Application of artificial intelligence to maximum dry density and unconfined compressive strength of cement stabilized soil,” *Geotechnical and Geological Engineering*, vol. 29, pp. 329–342, 2011.
[19] B. Mahesh, “Machine learning algorithms-a review,” *International Journal of Science and Research (IJSR).[Internet]*, vol. 9, pp. 381–386, 2020.
[20] Z.-H. Zhou, *Machine learning*. Springer Nature, 2021.
[21] H. Wang, Z. Lei, X. Zhang, B. Zhou, and J. Peng, “Machine learning basics,” *Deep learning*, pp. 98–164, 2016.
[22] G. Biau, “Analysis of a random forests model,” *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 1063–1095, 2012.
[23] K. Kim, “Financial time series forecasting using support vector machines,” *Neurocomputing*, vol. 55, no. 1–2, pp. 307–319, 2003.
[24] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245, pp. 255–260, 2015.
[25] T. Chen, J. Morris, and E. Martin, “Gaussian process regression for multivariate spectroscopic calibration,” *Chemometrics and Intelligent Laboratory Systems*, vol. 87, no. 1, pp. 59–71, 2007.
[26] W. Ni, L. Nørgaard, and M. Mørup, “W. Ni, L. Norgaard, M. Morup, Non-linear calibration models for near infrared spectroscopy, Analytica Chimica Acta 813 (2014) 1–14.,” *Analytica chimica acta*, vol. 813, pp. 1–14, 2014.
[27] I. Das, A. Stein, N. Kerle, and V. K. Dadhwal, “Landslide susceptibility mapping along road corridors in the Indian Himalayas using Bayesian logistic regression models,” *Geomorphology*, vol. 179, pp. 116–125, 2012.
[28] L. Wang, Q. Cao, Z. Zhang, S. Mirjalili, and W. Zhao, “Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems,” *Engineering Applications of Artificial Intelligence*, vol. 114, p. 105082, 2022.
[29] A. J. Riad, H. M. Hasanien, R. A. Turky, and A. H. Yakout, “Identifying the PEM Fuel Cell Parameters Using Artificial Rabbits Optimization Algorithm,” *Sustainability*, vol. 15, no. 5, p. 4625, 2023.
[30] I. Ahmadianfar, O. Bozorg-Haddad, and X. Chu, “Gradient-based optimizer: A new metaheuristic optimization algorithm,” *Information Sciences*, vol. 540, pp. 131–159, 2020.