Journal of Geomine (2024), 2(3), 156-175
DOI: 10.22077/jgm.2025.9350.1047

Machine learning-based predictions for Toyoura sand constitutive behaviors under triaxial compression monotonic loadings

Mojtaba Rahimi[1], Ali Akbari[2]

Recived: 2025 May 08, Revised: 2025 May 28, Online Published: 2025 Aug. 01

ABSTRACT

The mechanical behaviors of sands exhibit nonlinear stress-strain relationships under applied loads, involving a complex interaction between volumetric and deviatoric responses. An accurate understanding of constitutive behaviors is crucial for predicting how sand responds under various loading conditions. However, laboratory investigations of sand behavior under monotonic loading are challenging due to the intricate nature of stress-strain responses. Moreover, traditional constitutive models are often time-consuming, computationally intensive, and require extensive calibration. Machine learning (ML) techniques provide a promising alternative by learning patterns and trends from experimental or modeled data. In this study, three ML methods, namely Decision Trees (DT), K-Nearest Neighbors (KNN), and Random Forests (RF), were employed to evaluate the constitutive behaviors of Toyoura sand under drained and undrained triaxial compression monotonic loadings. The models' performance was assessed using R2, Mean Absolute Deviation (MAD), and Root Mean Square Error (RMSE). In this context, "high accuracy" refers to R^2 values close to 1, coupled with low MAD and RMSE values, indicating a strong correlation between predicted and actual responses. Under drained conditions, the ML models achieved high accuracy across varying initial void ratios, with R2 values up to 0.9992 for volumetric strain and 0.9980 for deviatoric stress, along with minimal prediction errors and zero training-phase error, reflecting a perfect model fit. Among the models, KNN demonstrated superior performance in most drained cases, likely due to its effectiveness in capturing local nonlinear trends within the dataset. Under undrained conditions and across a wide range of confining pressures (Pc = 100–2000 kPa), the ML models maintained robust predictive capability. High R2 values (up to 0.9998) and low error metrics confirmed the models' reliability, showing excellent agreement with training data. These findings validate the efficacy of ML algorithms in accurately modeling complex mechanical behaviors, including deviatoric and volumetric responses under different confining pressures and initial void ratios.

KEYWORDS

Machine learning, Toyoura sand, constitutive modeling, K-Nearest Neighbors, Random Forests, Decision Trees

I. INTRODUCTION

The mechanical behaviors of sands show nonlinear stress-strain relationships under monotonic loadings with a complex interplay between volumetric and deviatoric responses. Critical state conditions emerge in these granular materials only after significant shearing occurs (usually when the axial strain exceeds 10%) (Liu et al., 2022; Tarhouni & Amer, 2021). An accurate understanding of constitutive behaviors is paramount for predicting how sand will behave under different loading conditions. This understanding is crucial for designing structures like foundations and retaining walls, where accurate estimations of sand behavior under mechanical loads can prevent structural failures, and also for analysis of other geotechnical research topics, including seismic analysis of earth structures and buried pipeline stability (Latini et al., 2017; Tarhouni & Amer, 2021). To optimize structural safety factors, the industry needs an accurate prediction of sand response under various stress paths. Current design practices focus mainly on laboratory testing techniques, such as triaxial compression tests, which demand accurate control of strain rates and drainage conditions to maintain specimen integrity during the tests (Liu et al., 2022; Tarhouni & Amer, 2021). However, laboratory measurements face inherent challenges in replicating in-situ fabric conditions and scaling effects (Latini et al., 2017). Moreover, laboratory characterization of sand behavior under monotonic loadings is also challenging due to the complex nature of sand's response to stress, which involves both deviator strain and volumetric strain (De Silva & Koseki, 2012). Constitutive models assist us in predicting the behavior of sand under monotonic loading. These models are employed in the construction of infrastructure like bridges and foundations, where understanding sand behavior under applied loads is critical for stability issues (De Silva & Koseki, 2012; Sassel & O'Sullivan, 2024). While

[1] Department of Petroleum Engineering, Kho.C., Islamic Azad University, Khomeinishahr, Iran, [2] Stone Research Center, Kho.C., Islamic Azad University, Khomeinishahr, Iran, [3] Department of Petroleum Engineering, Faculty of Petroleum, Gas, and Petrochemical Engineering, Persian Gulf University, Bushehr, Iran
✉ M. Rahimi: mrahimi@iau.ac.ir

traditional constitutive models provide a foundation for understanding sand behavior, they often need extensive calibration and may not capture all the complexities of sand's response. Furthermore, critical state-based constitutive models are complicated because accurately capturing the non-linear and path-dependent behavior of sand requires sophisticated models with numerous material parameters (Yeh et al., 2023). Besides, critical state-based constitutive models may have limitations in capturing the complex nature of sand behavior, especially under complex loading conditions. They may not be able to represent the anisotropic elastic response and the influence of intermediate principal stresses. This is because they usually assume simplified isotropic elasticity and do not consider the intermediate principal stress in the definition of the yield function. The accuracy of the predictions of these constitutive models also depends on the type of model parameters and their calibration. The accurate determination of these material parameters might be challenging and require adequate experimental data. The development of more sophisticated critical state constitutive models tailored for more complex loading conditions requires a more profound understanding of critical state soil mechanics and involves increasing mathematical and numerical complexities. Machine learning (ML) techniques offer a great solution to overcoming these limitations through data-driven pattern recognition. ML techniques offer a promising approach to modeling sand behavior by learning from the pattern and trend of existing experimental/constitutive modeling data. These models can improve prediction accuracy and reduce the need for extensive parameter calibration (Najjar & Zhang, 2000; Su et al., 2023). Unlike conventional plasticity models, ML methods bypass restrictive assumptions regarding hardening laws and flow rules by directly learning from the existing databases (Chen et al., 2021) .

Kohestani and Hassanlourad (2016) discussed the modeling of the mechanical behavior of carbonate sands using ML techniques, specifically support vector machines (SVMs) and artificial neural networks (ANNs). They highlighted the unusual characteristics of carbonate sands, such as particle crushability and compressibility, which distinguish them from other soil types. The study compared the performance of these ML models in predicting the mechanical behavior of various carbonate sands, utilizing a comprehensive database of triaxial test results. The findings indicated that both methods are reliable for representing the mechanical behavior of carbonate sands .

Gao (2018) presented a comprehensive review of the identification of geomaterial constitutive models using computational intelligence methods. The review was organized into four key aspects: the approach, description, selection, and construction of constitutive models by ANNs and evolutionary computation. The study reported that challenges presented by the complexity and numerous parameters of traditional models highlight the importance of developing models that effectively describe real engineering behaviors through back analysis. The document also discussed the advantages and disadvantages of current research directions and suggested future research to focus on identifying a geomaterial constitutive model based on computational intelligence. Pouragha et al. (2020) explored the integration of artificial intelligence (AI) in geotechnical studies and the modeling of geomaterials. They addressed the fundamental questions concerning the capability of AI-generated models to represent the constitutive behavior of geomaterials. The authors emphasized the importance of understanding the long-term impacts of AI in geomechanics and the potential shifts in theoretical constitutive modeling. Zhang et al. (2021) discussed the ability of ML to learn from raw data and offered a detailed comparison of different ML algorithms in predicting and modeling soil behaviors. This review paper highlighted the principles of various ML algorithms, their characteristics, limitations, and methodologies in constructing ML-based soil constitutive models. The findings indicated that long short-term memory (LSTM) networks and their variants are particularly effective for this purpose .

Zhang et al. (2022) discussed the establishment of a constitutive model for sand under monotonic loading using SVM technology. They explained the complexity of sand's deformation mechanisms, especially under varying stress paths, and how traditional mathematical models may not capture the responses adequately. The authors conducted triaxial tests on Fujian sand to gather data, which was then employed to train SVM models using different optimization algorithms. The study found that incorporating PSO (particle swarm optimization) and GWO (grey wolf optimization) algorithms in the SVM model offers better predictions for the deformation modulus of sand, with GWO-SVM being the most effective under monotonic loading conditions. Wu and Wang (2022) presented a novel approach to constitutive modeling of natural sands by incorporating the effects of particle shape using deep learning techniques. Traditional methods often overlook these effects, leading to limitations in understanding granular material behavior. An LSTM network, a type of deep learning model, was developed to analyze how particle shape, confining pressure, and initial sample density affect the constitutive behavior of the sands. The effectiveness of this model was validated through comparisons with numerical simulation results and triaxial test data. Eghbalian et al. (2023) presented a physics-informed deep neural network architecture developed for surrogate modeling in classical elasto-plasticity, termed the Elasto-Plastic Neural Network (EPNN). This model incorporates essential physics aspects, facilitating efficient network training with reduced data while enhancing extrapolation

capabilities. The EPNN was adaptable to various elastoplastic materials, including sand. Zhang et al. (2023) discussed the application of ML in the constitutive modeling of sand and clay, focusing on improving interpretability and reducing dependency on large datasets. They proposed a data-driven approach that incorporates established theoretical knowledge and uncertainty in predictions. The study evaluated the performance of pure and physics-constrained data-driven models, highlighting the efficacy of the latter in predicting soil behavior, especially in cases with sparse data. Guan and Yang (2023) presented a novel hybrid deep learning model designed to predict the monotonic and cyclic responses of sand under various loading conditions. They utilized LSTM and temporal convolutional networks (TCN). The study involved analyzing a synthetic dataset generated by a constitutive model to determine the optimal arrangement of LSTM and TCN layers, comparing the performance of the hybrid model against individual LSTM and TCN models. Results demonstrated the superior predictive performance of the hybrid model, achieving high accuracy in replicating the constitutive responses of sand under both monotonic and cyclic loading. Wu et al. (2023) discussed an ML framework for predicting the stress-strain behavior and shear-induced contact fabric evolution of granular materials during triaxial shearing tests. They emphasized the use of a multi-layer perceptron (MLP) neural network that requires only the initial void ratio of the granular sample as input to predict its constitutive response. Their findings suggested that ML-based constitutive modeling can effectively capture the behavior of granular materials. Dornheim et al. (2024) focused on the application of neural networks in constitutive modeling. They outlined the evolution of constitutive models from traditional physics-based approaches to advanced ML methods, with a particular emphasis on the capabilities of neural networks. The authors highlighted the advantages of ML techniques in performing rapid numerical simulations compared to more complex physics-based models. Wang et al. (2024) provided a comprehensive overview of the applications of ML in the study of granular materials. They detailed various ML methods, particularly neural networks, and their effectiveness in modeling the constitutive behavior of granular materials. Eidgahee and Shiri (2024) discussed the modeling of the stress-strain behavior of frozen sandy soil using ML techniques, specifically ANNs. A comprehensive database from triaxial tests was created to train the ANN models on the relationships between deviatoric stresses, volumetric strains, and key variables such as temperature and confining pressure. The findings suggested that the models can predict the stress-strain behavior of frozen soil with significant accuracy. Noor et al. (2025) presented a study on a recursive Bayesian neural network (rBNN) framework designed for the constitutive modeling of sands under monotonic loading. They highlighted the importance of data-driven deep learning models in creating predictive constitutive models. Validation of the proposed framework was conducted using two datasets, demonstrating its capability to provide robust predictions. Yao et al. (2025) presented a study on a short-sequence ML framework designed for predicting the constitutive relationships of sand. They reported that classical numerical methods face challenges with complex material behavior and iterative processes. The study highlighted the effectiveness of the full sequence strategy using MLP and LSTM models .

In summary, the review of relevant literature indicates that a notable surge has occurred in the application of ML for sand constitutive modeling since 2020. This is because ML techniques offer a promising approach to modeling sand behavior by learning from existing experimental/modeling data and capturing complex patterns. These models reduce the need for extensive calibration of material parameters. To replicate the constitutive behaviors of Toyoura sand under triaxial compression monotonic loadings, we initially employed a well-established critical state constitutive model (Imam, 1999; Imam et al. 2005) and re-validated the constitutive model based on existing experimental observations of Toyoura sand. ML techniques have not been applied to the critical state constitutive model mentioned above. In this research, we focused on predicting Toyoura sand behavior under monotonic loadings by applying three different ML techniques to the numerical predictions of the aforementioned constitutive model. We adopted four inputs (namely axial strain, void ratio, critical state void ratio, and dilatancy rate) generated using the validated critical state constitutive model to estimate the deviator and volumetric behaviors of Toyoura sand. We utilized different ML methods to achieve this goal and to address the constitutive behaviors of Toyoura sand under triaxial drained and undrained compression monotonic loadings.

II. CONSTITUTIVE MODELING

The critical state constitutive model developed by Imam (1999) and Imam et al. (2005) was initially adopted in this research. To ensure that the numerical results generated by this critical state constitutive modeling are valid and correct, the constitutive modeling was implemented numerically, and the consistency condition was satisfied for all strain increments. To avoid excessive lengthening of the manuscript, the reader is referred to Imam et al. (2005) for full details of the constitutive model, including elasticity, yield function, stress-dilatancy relationship, hardening law, stress-strain relationships, and calibration procedure. A single set of material parameters was used in all predictions made by the

constitutive model. Table 1 presents the values assigned for the model parameters in this study. Experimental data points were extracted from Imam et. al. (2005).

In Table 1, φ_μ is almost equal to the interparticle friction angle, φ_{cs} is the critical state friction angle, G_a and K_a are reference elastic moduli, e_{cs} is the critical state void ratio, p is the mean effective normal stress, and the other parameters are constitutive model parameters.

Fig. 1 shows experimental observations and constitutive modeling results for Toyoura sand under drained triaxial compression loadings. The laboratory data have been obtained under three different initial void ratios (e=0.81, e=0.886, and e=0.96) and a constant confining pressure (Pc=500 kPa). As observed, there are perfect agreements between experimental observations and constitutive modeling results in all cases. This re-confirms the validity of the constitutive model developed by Imam (1999) and Imam et al. (2005). Figs. 2-3 exhibit experimental and constitutive modeling results for Toyoura sand under undrained triaxial compression loadings. The laboratory data have been acquired under two different initial void ratios (e=0.735 and e=0.833) and three confining pressures (Pc=100 kPa, Pc=1000 kPa, and Pc=2000 kPa). The comparison between experimental and modeling results suggests the great capability of the constitutive model, and it once again verifies the validity of the proposed constitutive model.

Since a continuous and incremental record of experimental data is not usually publicly available to researchers, using data from a constitutive model whose validity has already been verified is an alternative way of applying data-driven approaches for engineering purposes. In the following sections, the data generated through the validated constitutive model are employed to predict Toyoura sand behaviors under drained and undrained triaxial compression loadings.

Table 1. Material parameters and their values assigned in the constitutive modeling adopted in the current study

k_p	φ_μ	a_p	φ_{cs}	k_{PT}	a_{PT}	G_a	K_a	h	C	k_f	e_{cs}
1.2	21	0.18	31	1.25	0.15	8e6	8.5e6	1	6e-3	0.75	$-0.0063477p^3 + 0.0367p^2 - 0.11991p + 0.92548$ (p in Mpa)

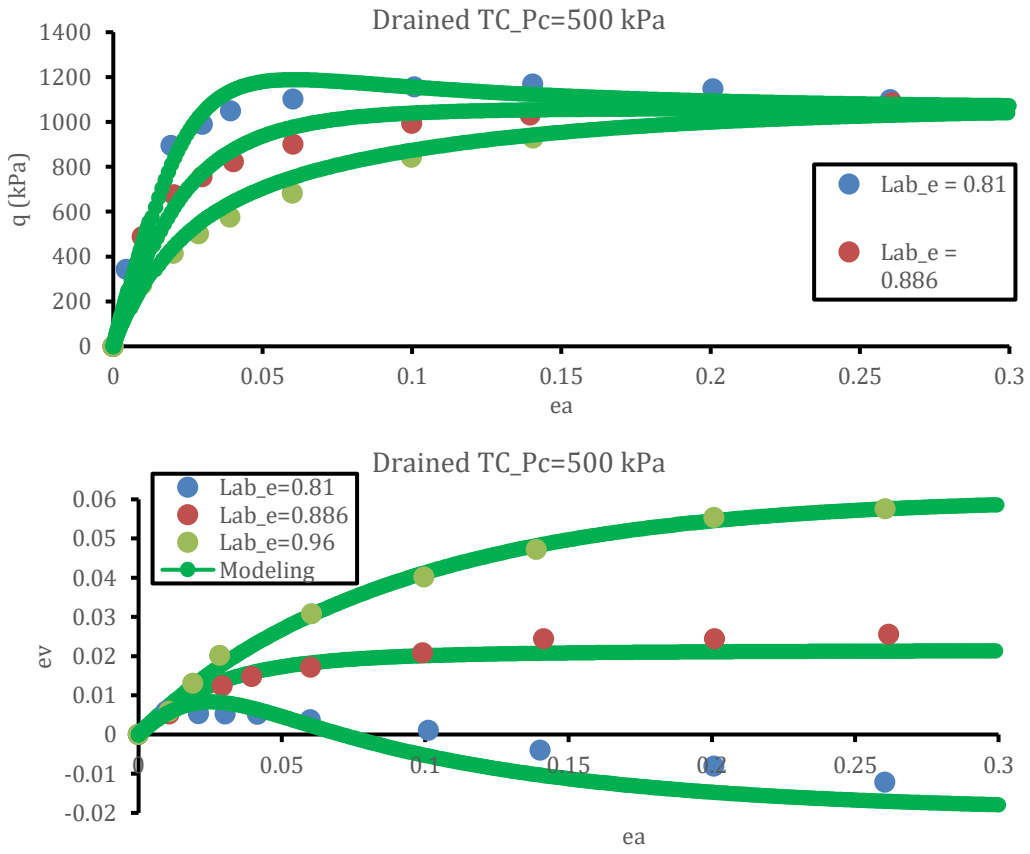


Fig. 1. Laboratory observations and predicted deviator and volumetric behaviors by constitutive modeling for Toyoura sand under drained triaxial compression loadings

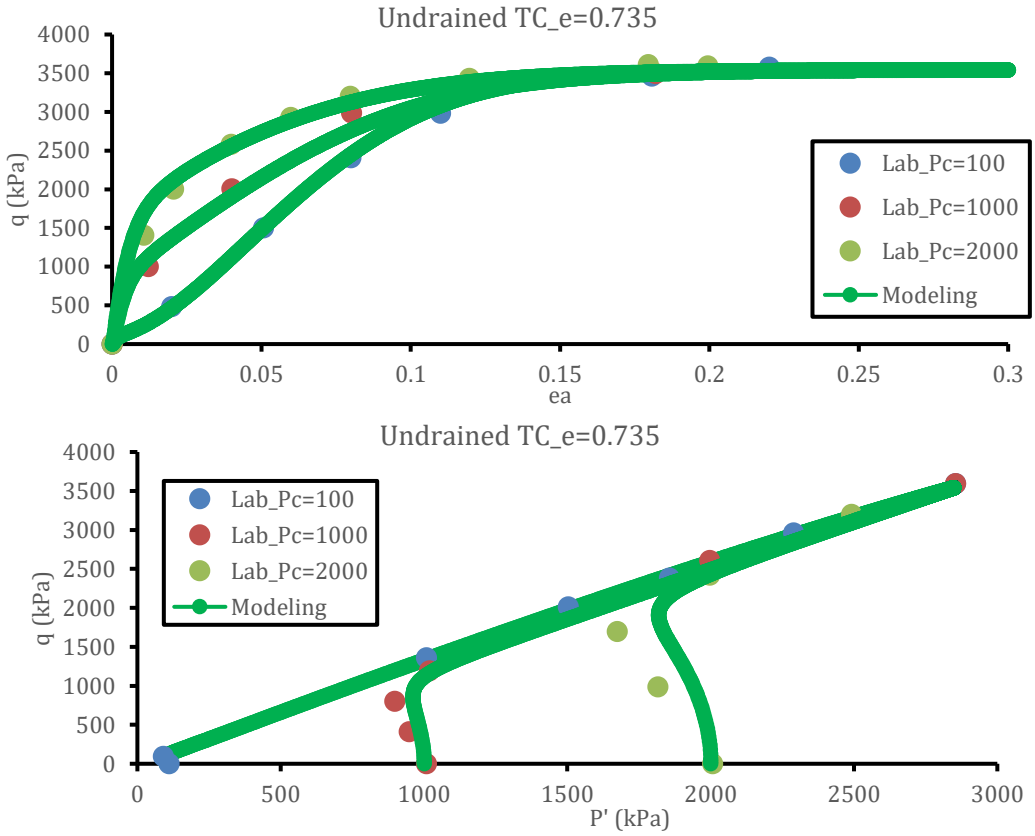


Fig. 2 Laboratory observations and predicted deviator stress vs. axial strain and deviator stress vs. mean effective normal stress by constitutive modeling for Toyoura sand under undrained triaxial compression loadings (initial void ratio=0.735)

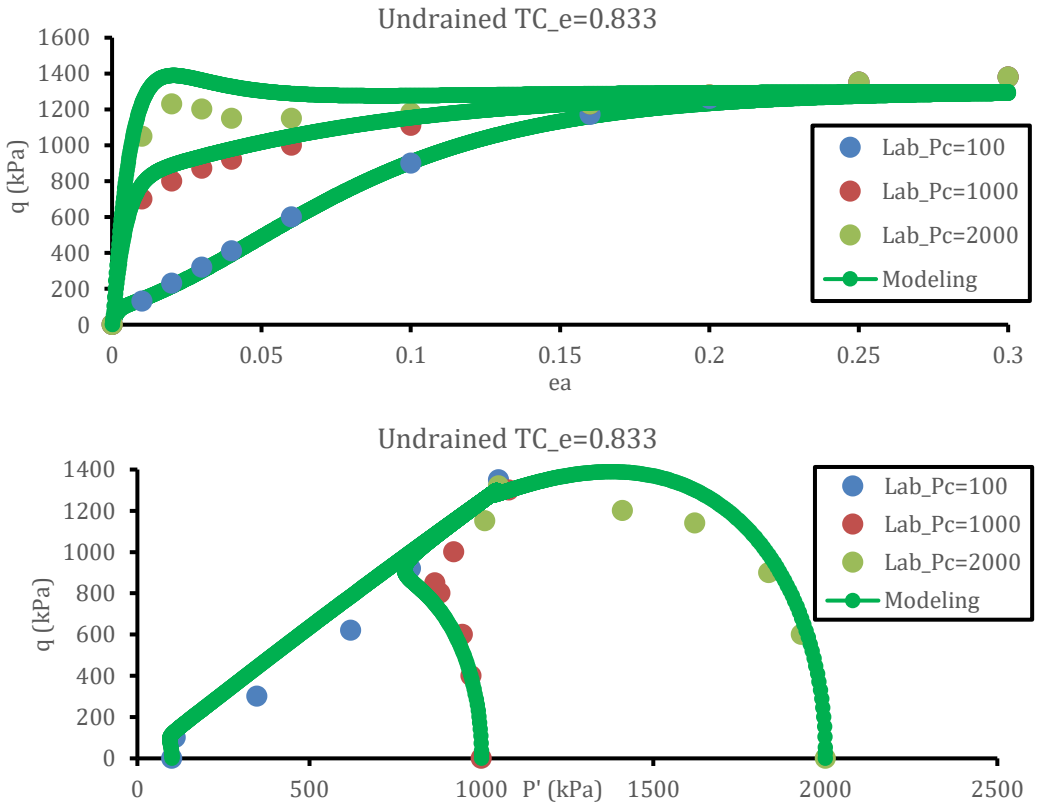


Fig. 3 Laboratory observations and predicted deviator stress vs. axial strain and deviator stress vs. mean effective normal stress by constitutive modeling for Toyoura sand under undrained triaxial compression loadings (initial void ratio=0.833)

III. METHODOLOGY

A. Data Collection and Processing

The dataset employed in this study comprises a set of key parameters associated with the constitutive behaviors of the Toyoura sand samples extracted from the constitutive modeling validated above. These parameters include axial strain (e_a), void ratio (e), critical state void ratio (e_{cs}), dilatancy rate (d), deviator stress (q), volumetric strain (e_v), and mean effective normal stress (p). These parameters play a crucial role in analyzing the mechanical behavior of sands, and hence, they have been selected as influential parameters in ML analyses. The value of axial strain is of paramount importance in strain-control physical or numerical tests, making its consideration essential in ML studies. The value of the void ratio and its relationship with the corresponding critical state void ratio, which determines the values of the state parameter in conventional critical state models, suggests whether the sand is on the loose/wet side or the dense/dry side of the critical state line. This directly affects the deviatoric and volumetric responses of sands tested under a given confining pressure. Therefore, considering these parameters as influential parameters on the mechanical behaviors of sands is logical. The value of the dilatancy rate (d) determines whether the sand is under compression or dilation. Besides, the change in the sign of d (namely a zero value for d) dictates a temporary steady state, which is called phase transformation and it is a critical parameter in the sand constitutive behavior. Thus, it is rational to take parameter d into account during ML analysis.

The collected data were subjected to comprehensive statistical and quantitative preprocessing procedures. This included conducting correlation analysis to identify the relationships between variables, performing sensitivity analysis to determine the influence of each parameter, and carrying out thorough evaluation and validation processes to ensure data reliability and robustness. Following the preprocessing phase, various ML models were developed and applied to both training and testing datasets. The performance of these models was systematically evaluated based on appropriate metrics to determine their predictive capabilities and generalization potential. A detailed summary of the statistical analyses and model evaluation results is provided in Table 2.

In addition to standard descriptive statistics, skewness and kurtosis values were examined to assess the distribution characteristics of the dataset. Skewness provides insight into the asymmetry of the data, while kurtosis reflects the presence of outliers or extreme values through the "tailedness" of the distribution. High skewness may indicate the presence of bias in certain input features, which can affect the sensitivity of ML models, particularly distance-based algorithms like KNN. Similarly, high kurtosis can suggest the presence of heavy tails or outliers, which may influence models such as decision trees or random forests, potentially leading to overfitting or decreased generalization (See Table 2). Recognizing these distribution properties helped guide the preprocessing stage and informed the selection and tuning of ML algorithms to ensure robustness and accuracy.

B. ML Methods

The model outputs correspond to q and e_v (drained tests) and q and p' (undrained tests), and reflect the results derived from analyzing and processing the input parameters within the proposed framework. To examine the influence of each input parameter on the target output, their variations concerning the output are illustrated in Fig. 4 according to the constitutive modeling results. This figure aids in a better understanding of the relationships between model variables and demonstrates how the system responds to changes in different parameters. In the subsequent sections of this study, a brief description of the algorithms employed will be provided to familiarize readers with the methodologies applied in the modeling process. These explanations will serve as a foundation for a deeper comprehension of the model's performance and the validity of the obtained results.

The selection of DT, KNN, and RF in this study was motivated by their complementary strengths in modeling nonlinear relationships and handling complex datasets. DT offers high interpretability and is capable of capturing nonlinear patterns with a simple tree-based structure. KNN is a non-parametric method that performs well in capturing local relationships without making strong assumptions about the underlying data distribution. RF, as an ensemble method, combines multiple decision trees to enhance prediction accuracy and reduce overfitting. These algorithms were chosen due to their proven effectiveness in regression tasks, their adaptability to various data characteristics, and their relatively low requirement for parameter tuning compared to more complex ML models such as deep neural networks.

1) Decision Tree (DT)

The Decision Tree (DT) algorithm is widely recognized in ML for its effectiveness in both classification and regression tasks. It operates as a predictive model that systematically partitions data into subsets based on feature values. The process begins by selecting a feature to divide the dataset, aiming to create the most informative split possible. At each decision node, a single feature is chosen to separate the data into two or more branches, to maximize the purity of the resulting groups (De Ville, 2013).

Table 2. Data Statistics

Drained

	e = 0.810						e = 0.886						e = 0.960					
	ea	e	ecs	d	q (kPa)	ev	ea	e	ecs	d	q (kPa)	ev	ea	e	ecs	d	q (kPa)	ev
Max	0.301	0.846	0.874	0.621	1189	0.008	0.300	0.886	0.874	0.736	1061	0.021	0.300	0.960	0.874	0.842	1042	0.059
Min	0.000	0.795	0.843	-0.177	0.010	-0.018	0.000	0.846	0.846	0.001	0.010	0.000	0.000	0.848	0.846	0.016	0.010	0.000
Range	0.301	0.048	0.031	0.798	1189	0.026	0.300	0.040	0.028	0.735	1061	0.021	0.300	0.112	0.028	0.826	1042	0.059
Media	0.122	0.825	0.845	-0.041	1091	-0.008	0.124	0.848	0.846	0.010	1051	0.021	0.127	0.871	0.849	0.135	923	0.046
Q1	0.035	0.809	0.844	-0.102	1074	-0.015	0.034	0.846	0.846	0.003	821	0.016	0.033	0.854	0.847	0.047	585	0.020
Q3	0.211	0.838	0.846	-0.016	1134	0.001	0.213	0.859	0.852	0.151	1060	0.021	0.215	0.923	0.858	0.435	1009	0.056
Mean	0.127	0.822	0.850	0.036	918	-0.007	0.128	0.856	0.852	0.154	835	0.016	0.129	0.889	0.854	0.273	741	0.037
Varian	0.009	0.000	0.000	0.057	15532	0.000	0.009	0.000	0.000	0.066	14215	0.000	0.009	0.002	0.000	0.083	13051	0.000
Skewn	0.174	-0.218	1.611	1.589	-1.584	0.233	0.145	1.313	1.451	1.465	-1.423	-1.328	0.122	0.726	1.164	0.997	-1.121	-0.724
Kurtos	-1.288	-1.432	0.826	1.009	0.741	-1.423	-1.308	-0.004	0.353	0.404	0.271	0.042	-1.329	-1.050	-0.254	-0.570	-0.344	-1.045

Undrained – 0.735

	Pc = 100					Pc = 1000					Pc = 2000				
	ea	ecs	d	q (kPa)	p' (kPa)	ea	ecs	d	q (kPa)	p' (kPa)	ea	ecs	d	q (kPa)	p' (kPa)
Max	0.300	0.914	0.433	3542	2848	0.300	0.839	0.562	3541	2847	0.300	0.791	0.647	3541	2847
Min	0.000	0.735	-0.456	0.100	99	0.000	0.735	-0.218	0.100	961	0.000	0.735	-0.107	0.100	1818
Range	0.300	0.179	0.889	3542	2749	0.300	0.103	0.780	3540	1886	0.300	0.056	0.754	3541	1029
Media	0.150	0.742	-0.024	3429	2739	0.150	0.742	-0.016	3413	2743	0.150	0.738	-0.006	3474	2794
Q1	0.075	0.735	-0.200	2308	1768	0.075	0.736	-0.094	2673	2146	0.075	0.735	-0.040	3073	2477
Q3	0.225	0.793	-0.001	3536	2742	0.225	0.774	-0.002	3528	2837	0.225	0.757	-0.001	3533	2842
Mean	0.150	0.772	-0.113	2790	2217	0.150	0.759	-0.046	2981	2415	0.150	0.749	-0.012	3179	2611
Varian	0.008	0.003	0.024	11604	76875	0.008	0.001	0.007	62993	34405	0.008	0.000	0.007	35634	10690
Skewn	0.000	1.407	-1.033	-1.312	-1.234	0.000	1.276	1.278	-1.543	-1.296	0.000	1.219	4.648	-2.380	-1.302
Kurtos	-1.200	0.695	-0.027	0.283	0.059	-1.200	0.310	9.851	1.447	0.305	-1.200	0.018	28.316	6.271	0.237

Undrained – 0.833

	Pc = 100					Pc = 1000					Pc = 2000				
	ea	ecs	d	q (kPa)	p' (kPa)	ea	ecs	d	q (kPa)	p' (kPa)	ea	ecs	d	q (kPa)	p' (kPa)
Max	0.300	0.915	0.594	1289	1036	0.300	0.852	0.713	1291	1038	0.300	0.834	0.793	1389	2000
Min	0.000	0.834	-0.202	0.100	94	0.000	0.833	-0.034	0.100	776	0.000	0.782	-0.001	0.100	1030
Range	0.300	0.081	0.795	1289	943	0.300	0.018	0.748	1291	262	0.300	0.052	0.794	1389	970
Media	0.150	0.842	-0.030	1133	905	0.150	0.837	-0.007	1226	988	0.150	0.833	0.000	1292	1041
Q1	0.075	0.835	-0.108	718	566	0.075	0.834	-0.019	1094	896	0.075	0.833	-0.001	1282	1035
Q3	0.225	0.868	-0.008	1258	1010	0.225	0.843	-0.002	1274	1025	0.225	0.834	0.010	1297	1043
Mean	0.150	0.855	-0.059	956	765	0.150	0.839	0.006	1156	954	0.150	0.830	0.043	1279	1099
Varian	0.008	0.001	0.006	13906	89294	0.008	0.000	0.007	31172	6947	0.008	0.000	0.015	14233	31331
Skewn	0.000	1.138	1.074	-1.013	-0.967	0.000	0.900	5.555	-2.596	-0.855	0.000	-3.332	3.753	-7.308	3.483
Kurtos	-1.200	-0.043	10.897	-0.333	-0.456	-1.200	-0.546	33.455	9.626	-0.634	-1.200	10.768	14.642	60.151	11.922

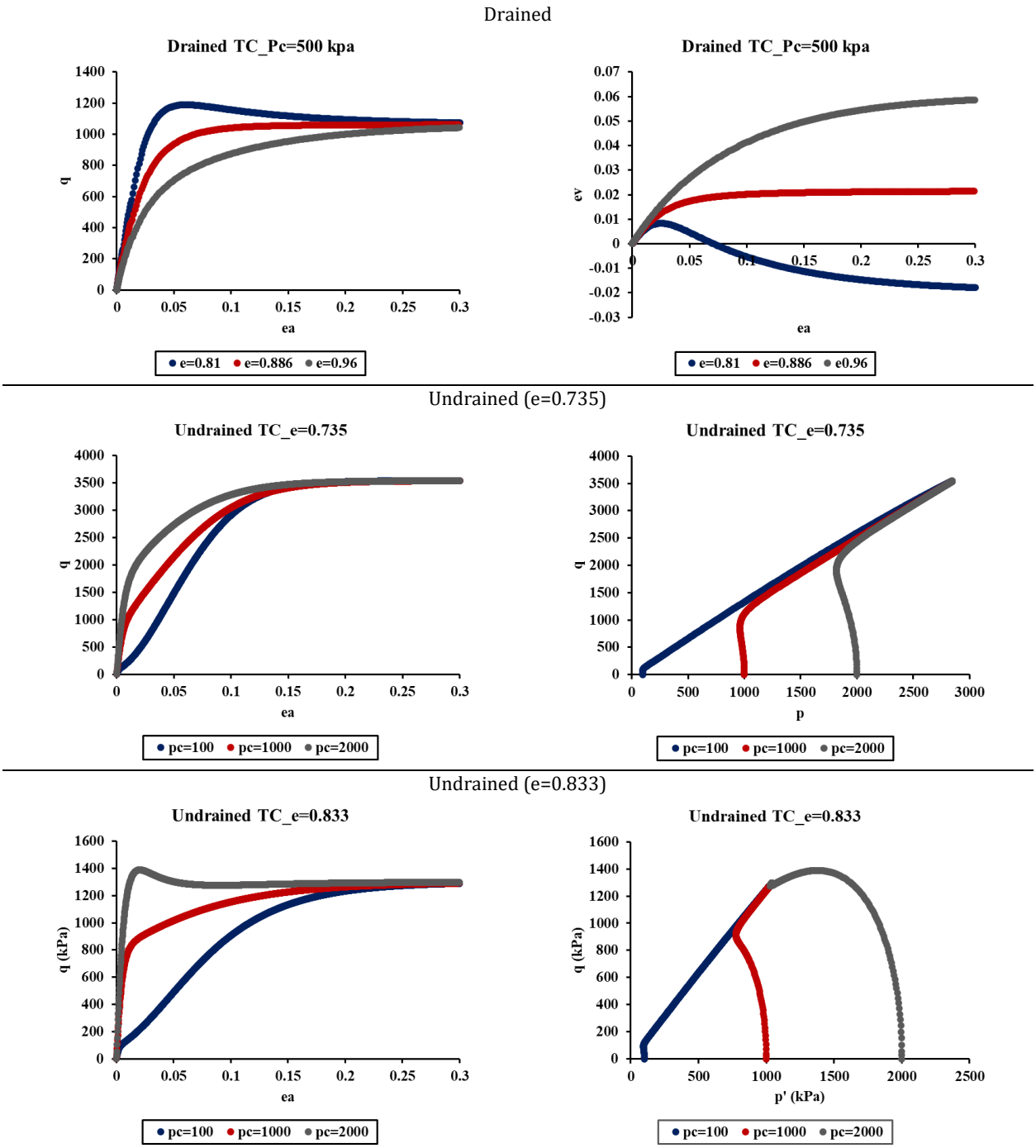


Fig. 4. Constitutive Responses/Behaviors Used as References for Developing ML Models

This selection is guided by a cost function or splitting criterion that assesses the quality of each potential split. Commonly used criteria include Entropy, which is typically applied in classification problems, and the Gini Index, often used in regression. The tree continues to split the data recursively until it satisfies certain stopping conditions—such as all samples in a node belonging to the same class, reaching a predefined maximum depth, or exhausting all available features.

To assess the effectiveness of each split, the DT algorithm relies on metrics like Entropy and the Gini Index. Entropy quantifies the amount of uncertainty or randomness in the dataset, and the goal during training is to reduce entropy as much as possible with each split, leading to clearer, more accurate decision paths. The formula for calculating entropy is as follows:

$$Entropy(S) = -\sum_{i=1}^{k} P_i \log_2(P_i) \quad (1)$$

Where P_i is the probability of a data point belonging to class i, and k is the number of classes. Conversely, the Gini Index evaluates the "impurity" of the data during the splitting process. The formula for calculating the Gini Index is as follows:

$$Gini(S) = 1 - \sum_{i=1}^{k} P_i^2 \quad (2)$$

Where P_i is the probability of a data point belonging to class i, and k is the number of classes. Furthermore, the Information Gain (IG) criterion is employed to identify the optimal feature for splitting. Information Gain measures the reduction in entropy resulting from the split, and its formula is:

$$IG(S,A) = Entropy(S) - \sum_{veValues(A)} \frac{|S_v|}{|S|}.Entropy(S_v) \quad (3)$$

Where S is the dataset, A is the selected feature for splitting, and S_v represents the subsets of data that the feature A divides. $|S_v|$ is the number of data points in the subset S_v. Due to its simplicity and interpretability, DTs are widely recognized as one of the most popular algorithms in ML.

2) K-Nearest Neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm is a popular ML method utilized for both classification and regression tasks. As a non-parametric and instance-based learning technique, KNN does not assume a predefined mathematical model for the data. Instead, it stores the training dataset and makes predictions by evaluating the similarity between data points (Peterson, 2009). KNN is especially useful for problems with complex, nonlinear decision boundaries (Fig. 5).

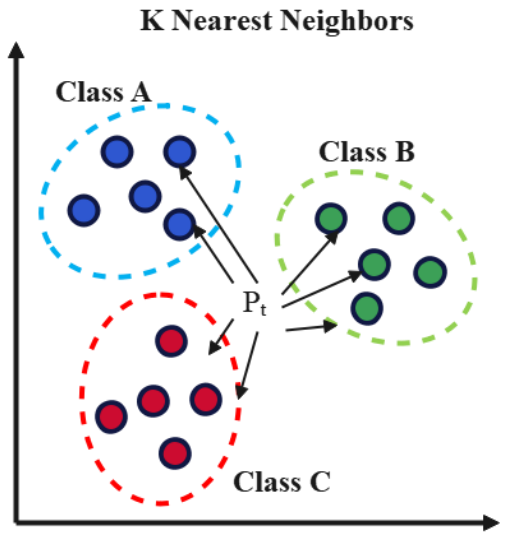


Fig. 5 KNN algorithm

The KNN algorithm functions on the principle of similarity between data points. When making predictions for a new data point, the algorithm identifies the K nearest neighbors from the training dataset and assigns an output based on their values. The K value is a hyperparameter that significantly affects the model's performance. A smaller K results in a more flexible decision boundary, while a larger K creates a smoother decision boundary by incorporating a greater number of neighbors.

To determine the nearest neighbors, KNN uses a distance metric. The Euclidean distance is the most commonly employed metric, which is calculated as:

$$d(X_i, X_j) = \sqrt{\sum_{m=1}^{n} (x_{i,m} - x_{j,m})^2} \quad (4)$$

Where X_i and X_j are two data points, and n is the number of features. Other distance metrics, such as Manhattan distance and Minkowski distance, may also be utilized based on the specific requirements of the problem. In a classification problem, KNN assigns a class label to a new data point based on the majority vote of its K nearest neighbors. The predicted class $\hat{y}$ is determined as:

$$\hat{y} = \arg max \sum_{i=1}^{K} I(y_i = c) \quad (5)$$

Where y_i represents the class of the $i-th$ neighbor, and $I(y_i = c)$ is an indicator function that equals 1 if y_i belongs to class c, and 0 otherwise. In certain situations, a weighted voting approach is applied, where closer neighbors have a greater influence on the decision-making process.

For regression tasks, KNN predicts the output by computing the average of the K nearest neighbors' values:

$$\hat{y} = \frac{1}{K} \sum_{i=1}^{K} y_i \quad (6)$$

Another approach is to use distance-weighted regression, where nearer neighbors have a larger influence on the final prediction.

$$\hat{y} = \frac{\sum_{i=1}^{K} \frac{y_i}{d(X, X_i)}}{\sum_{i=1}^{K} \frac{1}{d(X, X_i)}} \quad (7)$$

Because KNN does not build an explicit model, it is commonly classified as a lazy learning algorithm. Rather than performing calculations during the training phase, it stores all training instances and computes distances only when making predictions. This method makes KNN very flexible but also computationally costly, particularly with large datasets. Despite its simplicity, KNN is still a popular algorithm in ML due to its capacity to manage nonlinear decision boundaries and its versatility in handling various types of data.

3) Random Forest (RF)

Random Forest (RF) is a widely used ML method recognized for its efficiency in classification and regression tasks, especially when working with large and intricate datasets. Its effectiveness in reducing variance and preventing overfitting has led to its broad adoption. RF is an ensemble learning technique that merges multiple decision trees (DTs), where each tree is trained independently on a random subset of the data (Rigatti, 2017). The final prediction is generated by averaging the outputs from these individual trees (Fig. 6).

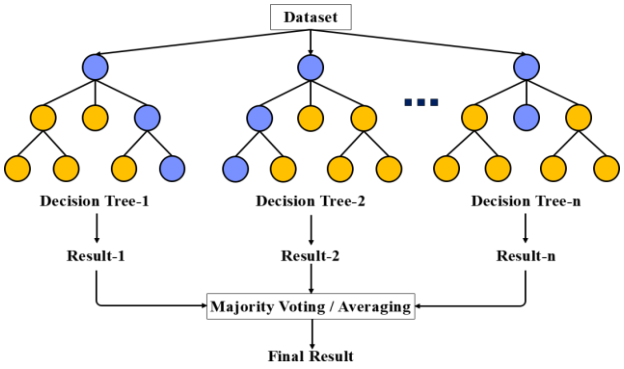


Fig. 6. The RF algorithm

The RF algorithm starts by generating random samples from the training dataset through bootstrap sampling (sampling with replacement). Each decision tree (DT) is trained on a distinct random subset of the data. To increase diversity among the trees, a random selection of features is made at each node. When making predictions, the results from all trees are combined. For classification tasks, the final prediction is determined by a majority vote, whereas for regression tasks, the predictions are averaged.

In classification, the final prediction for a new sample x is computed using the following formula, where T_1, $T_2, \dots, T_n$ represent the decision trees and $C_1, C_2, \dots, C_n$ are the possible classes:

$$Prediction\ (x) = \arg max\left(\sum_{j=1}^{n} I\left(T_j(x) = C_i\right)\right) \quad (8)$$

Here, I is an indicator function that equals 1 if $T_j(x)$ equals C_i and 0 otherwise.

For regression tasks, the final prediction for a new sample x is calculated as the average of the predictions from all the decision trees:

$$Prediction\ (x) = \frac{1}{n}\sum_{j=1}^{n} T_j(x) \quad (9)$$

where $T_j(x)$ is the predicted value from the decision tree T_j for the sample x.

4) Strengths and Limitations of Algorithms

ML algorithms are powerful tools for analyzing large and complex datasets, enabling effective data processing and providing accurate and reliable results. These algorithms are capable of identifying patterns and relationships within the data, facilitating predictions and informed decision-making across various scientific and industrial fields. However, each algorithm comes with its own set of features, advantages, and limitations that need to be carefully considered. The subtle differences between algorithms highlight the importance of a thorough evaluation of their unique characteristics and constraints, which is crucial for making informed decisions when selecting and implementing the appropriate model. The advantages and disadvantages of the models used in this study are clearly outlined in Table 3, providing a better understanding of their capabilities and the challenges associated with their use.

PSO is a population-based optimization algorithm inspired by the social behavior of birds flocking or fish schooling. Each individual in the population, called a particle, represents a potential solution. These particles explore the solution space by adjusting their positions based on their own experience and that of their neighbors. The algorithm updates the velocity and position of each particle to converge towards the global optimum. Due to its simplicity and efficiency, PSO is commonly used for optimizing hyperparameters in ML models and has been successfully applied in various geotechnical studies.

IV. Results and Discussion

A. *Data Division into Training and Testing Sets*

At the outset of this study, all input parameters were comprehensively collected and thoroughly analyzed to ensure the quality and adequacy of the initial dataset for subsequent processing. For the computational procedures, ML algorithms were selected and implemented in the MATLAB environment. After identifying and selecting the most relevant parameters, the input data were divided into two independent sets: one for training the models and the other for testing, to evaluate the performance of the algorithms. Finally, the accuracy and efficiency of each method were calculated, and the results were presented graphically through comparative charts to enable a more detailed analysis and interpretation.

The random selection of input data based on predefined ratios for dataset partitioning can significantly influence the final accuracy of ML algorithms. To mitigate the potential effects of randomness in data splitting and to enhance the reliability of the results, each algorithm was evaluated through ten independent runs. In each run, the dataset was randomly divided, and the model's performance was assessed using the coefficient of determination (R^2) as the evaluation metric. The average R^2 values obtained from these ten repetitions were reported as the final results for each algorithm. This multi-run evaluation strategy ensures that the outcomes are stable, reproducible, and generalizable, thereby providing a dependable representation of the factual accuracy and performance of the algorithms.

In this section of the study, various data percentages ranging from 10% to 80%, with a step size of 10%, were analyzed. These percentages serve as a key parameter in determining the training-to-testing data ratio during the data-splitting process in ML. The primary objective of this analysis is to assess the impact of different training-to-testing ratios on the performance of ML models and to optimize the training process

accordingly. This evaluation helps to understand better how the amount of training data influences prediction accuracy and model generalization. The findings can provide useful guidance for selecting an optimal data split ratio, ensuring sufficient data for learning while maintaining reliable evaluation capabilities. The corresponding numerical values and results are presented in Table 4.

The primary criterion for selecting the final results in this study is based on the overall training/testing ratio (combined ratio). Accordingly, for each of the evaluated ML algorithms, the maximum value obtained for the performance metric (such as the coefficient of determination, R^2) is identified and used as the basis for the final analysis. Subsequently, to enhance understanding and visualize the performance of these algorithms under optimal conditions, corresponding regression plots are generated. These plots, presented in Fig. 7, illustrate the relationship between the actual (modeling) and ML-predicted values, highlighting the models' ability to reconstruct real data accurately. This approach provides a comprehensive view of the models' effectiveness across different scenarios and serves as a foundation for a more precise comparison of algorithmic performance.

B. Performance of Each Method in the Training and Testing Phases

The results obtained from the regression analyses and R^2 values are presented in graphical form, where the training set (Train), test set (Test), and their combined performance are simultaneously displayed. These plots clearly illustrate the model's fit to the data and provide a visual representation of prediction accuracy.

In this study, the regression equations have been specifically designed and implemented for analyzing and predicting the behavior of various systems. To achieve higher accuracy, diverse ML methods have been utilized, each of which has been able to simulate the complex relationships between inputs and outputs and provide precise predictions. These models are capable of generating accurate and corresponding outputs based on input data and can be effectively used in decision-making processes and optimization.

Table 3. Strengths and Limitations of ML Algorithms

Algorithm	Strength	Limitation
DT	Simple to understand and explain Capable of processing both numerical and categorical variables	Tends to overfit the data Can be affected by minor fluctuations in the dataset
KNN	Straightforward and easy-to-understand Eliminates the need for a training phase	Requires significant computational resources at the prediction stage Susceptible to the influence of noisy data
RF	Efficiently processes large volumes of data Effectively resists overfitting tendencies	May lack interpretability Requires significant computational resources

Table 4. Final R^2 Values for Each Layer Across ML Methods

		Drained																	
		e = 0.810, q			e = 0.810, ev			e = 0.886, q			e = 0.886, ev			e = 0.960, q			e = 0.960, ev		
	Precision	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF
Test/Train (All)	10-90%	0.9772	0.9980	0.9760	0.7359	0.7204	0.8692	0.9587	0.9978	0.9668	0.9582	0.9980	0.9313	0.9535	0.9970	0.9747	0.9733	0.9992	0.9655
	20-80%	0.3953	0.9562	0.8484	0.7448	0.7428	0.7416	0.8358	0.7456	0.8957	0.5401	0.8829	0.5143	0.9421	0.8905	0.8000	0.8497	0.9295	0.8928
	30-70%	0.1602	0.6007	0.9116	0.7441	0.7341	0.7489	0.9071	0.7124	0.9084	0.3157	0.6812	0.6268	0.9419	0.8938	0.7552	0.5468	0.6244	0.9574
	40-60%	0.4993	0.2434	0.7446	0.7430	0.7588	0.7441	0.6111	0.7560	0.8114	0.3778	0.5527	0.7070	0.9271	0.9412	0.9209	0.4570	0.5762	0.9597
	50-50%	0.4507	0.7856	0.8254	0.7562	0.7241	0.7557	0.8861	0.6501	0.8076	0.4064	0.5860	0.3770	0.9474	0.8618	0.3538	0.7780	0.7885	0.9598
	60-40%	0.6597	0.8954	0.9055	0.7125	0.7235	0.7694	0.9022	0.7224	0.8770	0.5694	0.8165	0.9081	0.9473	0.6482	0.8237	0.4454	0.8952	0.9598
	70-30%	0.5448	0.6315	0.6790	0.7530	0.7739	0.7408	0.8027	0.9345	0.8936	0.2819	0.5428	0.6794	0.9380	0.9015	0.7096	0.8918	0.8693	0.9599
	80-20%	0.4132	0.9034	0.9455	0.7891	0.9800	0.7991	0.9082	0.4769	0.3404	0.2668	0.7885	0.8548	0.9487	0.2824	0.7609	0.8688	0.6036	0.9399
		Undrained (e=0.735)																	
		Pc = 100, p			Pc = 100, q			Pc = 1000, p			Pc = 1000, q			Pc = 2000, p			Pc = 2000, q		
	Precision	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF
Test/Train (All)	10-90%	0.9966	0.9998	0.9971	0.9964	0.9997	0.9973	0.9973	0.9997	0.9987	0.9915	0.9977	0.9883	0.9958	0.9995	0.9989	0.8567	0.9685	0.7906
	20-80%	0.1250	0.9200	0.1333	0.5346	0.7900	0.6720	0.4400	0.5100	0.1577	0.2160	0.7900	0.5546	0.2405	0.5900	0.1462	0.6020	0.9300	0.1032
	30-70%	0.3484	0.9100	0.7031	0.2920	0.8900	0.4888	0.2484	0.8900	0.4968	0.7440	0.5900	0.3304	0.1360	0.8000	0.4959	0.6972	0.5000	0.6935
	40-60%	0.6384	0.5600	0.1496	0.1617	0.8400	0.4320	0.7626	0.7700	0.5766	0.8184	0.9000	0.5106	0.5104	0.8900	0.5214	0.3240	0.7200	0.2021
	50-50%	0.6935	0.7800	0.1760	0.2303	0.7300	0.2720	0.4914	0.7600	0.7644	0.8170	0.4200	0.3150	0.1980	0.8900	0.1008	0.0864	0.9200	0.3854
	60-40%	0.2542	0.4200	0.3864	0.5733	0.4000	0.1920	0.0798	0.8300	0.7533	0.1328	0.6500	0.2160	0.4224	0.3800	0.4346	0.5920	0.8700	0.2958
	70-30%	0.1870	0.8200	0.1800	0.7216	0.6700	0.5460	0.6900	0.7100	0.0968	0.1935	0.2400	0.5278	0.3024	0.5600	0.4675	0.4080	0.9200	0.7980
	80-20%	0.4293	0.5200	0.5561	0.7626	0.2900	0.2162	0.4788	0.8300	0.4144	0.2262	0.8600	0.1326	0.2430	0.4200	0.2590	0.4599	0.8500	0.5658
		Undrained (e=0.833)																	
		Pc = 100, p			Pc = 100, q			Pc = 1000, p			Pc = 1000, q			Pc = 2000, p			Pc = 2000, q		
	Precision	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF	DT	KNN	RF
Test/Train (All)	10-90%	0.9957	0.9998	0.9996	0.9499	0.9614	0.8525	0.9971	0.9985	0.9990	0.9593	0.9786	0.9376	0.9776	0.9972	0.9925	0.8567	0.9685	0.7906
	20-80%	0.3626	0.9500	0.2080	0.6417	0.8600	0.5850	0.3008	0.7400	0.0792	0.3978	0.8700	0.2816	0.1224	0.3700	0.0700	0.5950	0.9500	0.3560
	30-70%	0.4524	0.8900	0.1806	0.3348	0.5400	0.1365	0.5467	0.5900	0.1053	0.0532	0.6000	0.3915	0.4278	0.9400	0.3186	0.3504	0.7700	0.3552
	40-60%	0.5551	0.9300	0.1080	0.4209	0.3100	0.3588	0.1482	0.9000	0.5859	0.4785	0.3700	0.3192	0.5104	0.6900	0.7743	0.2204	0.6100	0.1242
	50-50%	0.2268	0.8600	0.3705	0.3864	0.6900	0.7553	0.2814	0.8700	0.7410	0.3219	0.9500	0.1950	0.3723	0.8700	0.4263	0.4020	0.3200	0.4888
	60-40%	0.2625	0.5200	0.3120	0.5920	0.5200	0.1560	0.6660	0.6300	0.7056	0.2418	0.4000	0.5329	0.2914	0.7600	0.1988	0.2130	0.4400	0.5304
	70-30%	0.5369	0.4700	0.4088	0.3108	0.7600	0.2277	0.6808	0.1400	0.4180	0.4345	0.9000	0.5688	0.7636	0.4000	0.6164	0.5328	0.4200	0.2596
	80-20%	0.2590	0.8400	0.3150	0.4824	0.8300	0.2688	0.2214	0.6800	0.2871	0.1513	0.8000	0.2697	0.5440	0.6900	0.2967	0.2862	0.9500	0.4698

Three different ML methods were employed for analysis and prediction to evaluate the performance of each algorithm under various conditions. These methods were specifically chosen to simulate and model the complex relationships among the data and were selected to address the particular problems of this research. The selection of these algorithms was based on their ability to accurately simulate system behavior and provide reliable predictions under different scenarios.

After applying these algorithms and analyzing the results obtained, the best outcomes and performance of each method were determined based on prediction accuracy and how well they aligned with real data. These results are comprehensively presented in Table 5, which serves as a basis for evaluating and comparing the performance of different models in accurately predicting system behavior. This analysis can also guide us in selecting the most suitable model based on the specific needs of the research.

DT, KNN, and RF were utilized for data analysis and prediction. These methods were selected due to their ability to analyze complex and nonlinear relationships in the data and provide accurate predictions in various applications. Each of these algorithms is widely used in ML tasks for its high capacity to model system behaviors and deliver reliable results.

To enhance the accuracy of the models and optimize their performance, the Particle Swarm Optimization (PSO) algorithm was employed. PSO, as an evolutionary search method, was applied in the process of fine-tuning and selecting the optimal set of parameters for each model. This algorithm continuously searches the parameter space to find the best combination of parameters that maximizes the prediction accuracy of the models. The integration of PSO significantly improved the hyperparameter tuning process and increased the efficiency of the ML techniques in data analysis and predictive modeling. This approach, particularly beneficial in cases with complex and nonlinear data, has a significant impact on improving the accuracy of predictions and optimizing the results derived from the models. Overall, combining these ML algorithms with PSO contributes to increased accuracy, efficiency, and robustness in generating more precise and reliable predictions.

In ML models, various parameters control the performance of the model. Each of these parameters directly impacts the model's results and needs to be carefully tuned to achieve optimal performance. Below is an explanation of some of the most important parameters for different ML models.

For the DT model, the MinLeafSize parameter defines the minimum number of data samples in each leaf, which influences the model's complexity and overfitting. This parameter is directly related to overfitting because if the number of samples in the leaves is too small, the model may become overly sensitive to the details in the training data. On the other hand, the MaxNumSplits parameter, which specifies the maximum number of allowed splits in the tree, affects the tree's depth and the model's ability to distinguish between data points. This parameter is also critical in preventing excessive complexity in the tree, which can reduce the risk of overfitting.

For the KNN model, the NumNeighbors parameter determines the number of nearest neighbors considered during prediction. This parameter impacts the predictions as too few neighbors can make the model highly sensitive to small changes in the data, while too many neighbors may lead to the loss of important features in the data. Additionally, the DistanceMetric parameter, which measures the similarity between samples, affects how distances between data points are calculated and how the best predictions are chosen.

In the RF model, the NumTrees parameter defines the number of decision trees in the forest. The more trees there are, the more data the model can analyze, and the higher the prediction accuracy. The NumPredictorsToSample parameter, which controls the number of features to be considered for each tree, is crucial in preventing overfitting and enhancing the model's diversity. By selecting random features for each tree, the model avoids overfitting and gains the ability to generalize to new data.

Overall, these parameters are critical for fine-tuning the performance of ML models. They have a profound impact on factors such as sensitivity, accuracy, and the likelihood of overfitting. Optimizing these parameters can significantly improve model performance and prediction accuracy, making the process of selecting and adjusting these parameters an essential part of designing effective ML models.

Table 6 presents the optimal parameter values for each ML method using the PSO algorithm. These values have been carefully selected to achieve the best possible performance for each model and accurately represent the optimal settings for various parameters in each algorithm. The selection of these values ensures that each model, when using the optimized parameters, can deliver more precise and efficient results. The use of the PSO algorithm as the optimizer has significantly improved the accuracy and efficiency of the parameter search process. In general, PSO, as a metaheuristic search method, has been able to perform the parameter tuning process more effectively, leading to more accurate results in selecting the optimal values. This improvement in the parameter selection process has contributed significantly to enhanced prediction accuracy and the overall efficiency of the ML models.

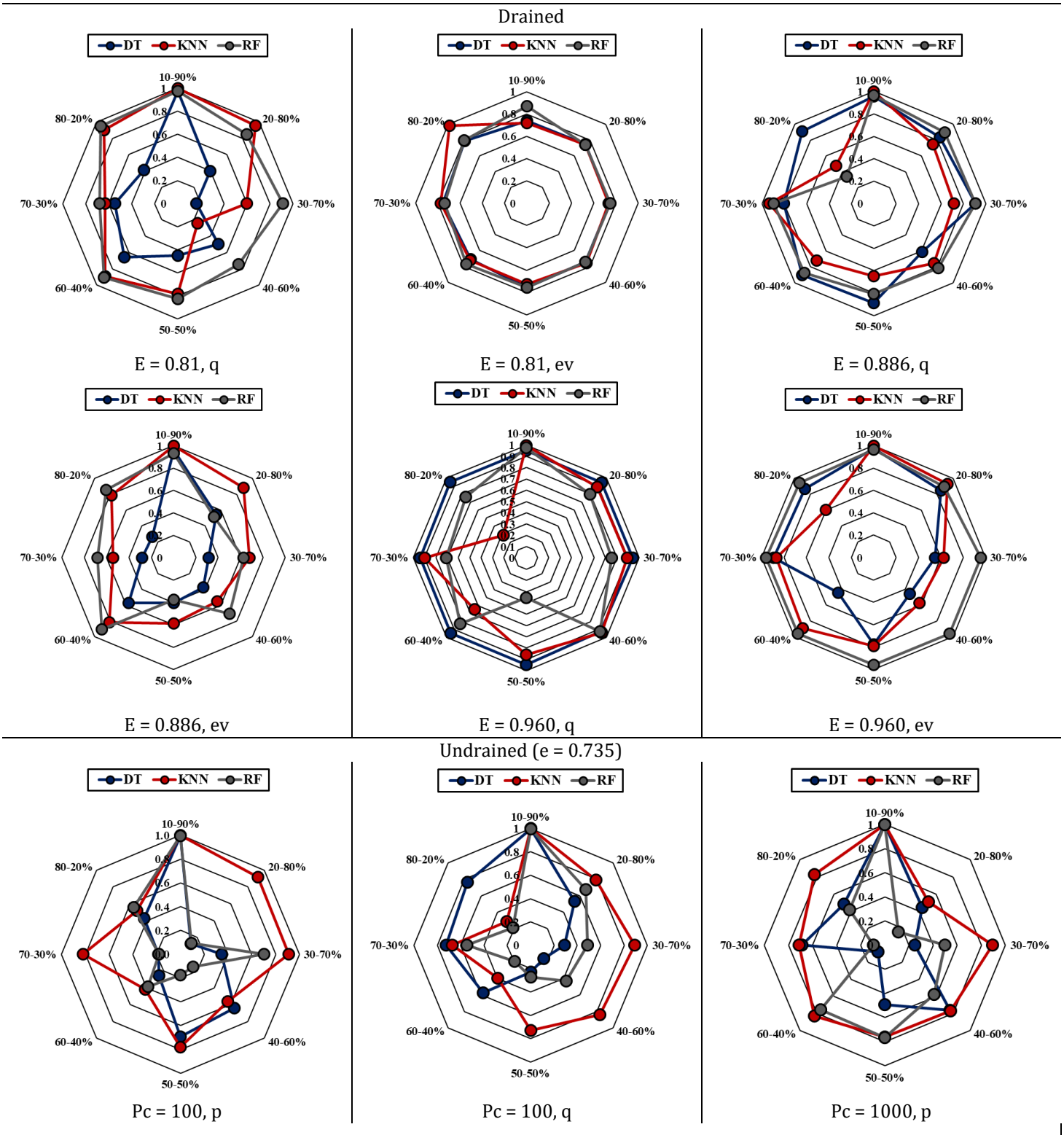
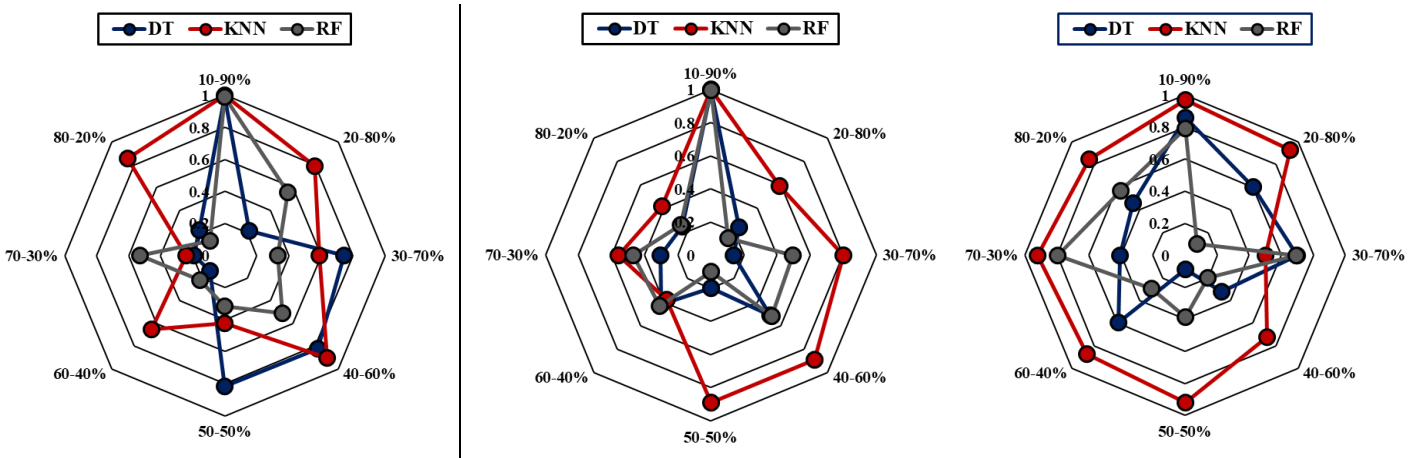
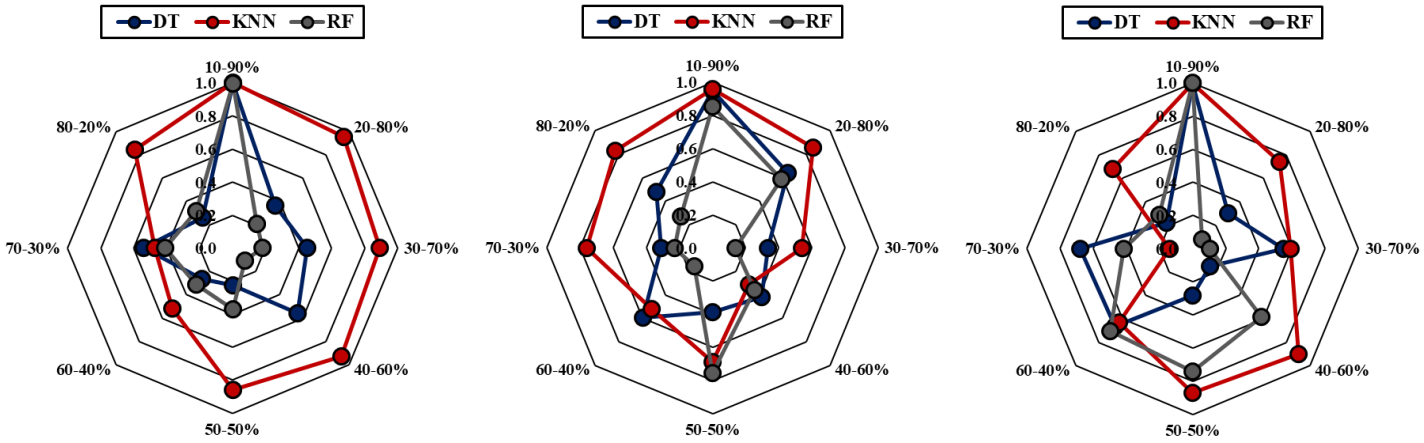


Fig. 7. Accuracy of different ML methods in different percentages of training-to-test data on Test/Train (All)

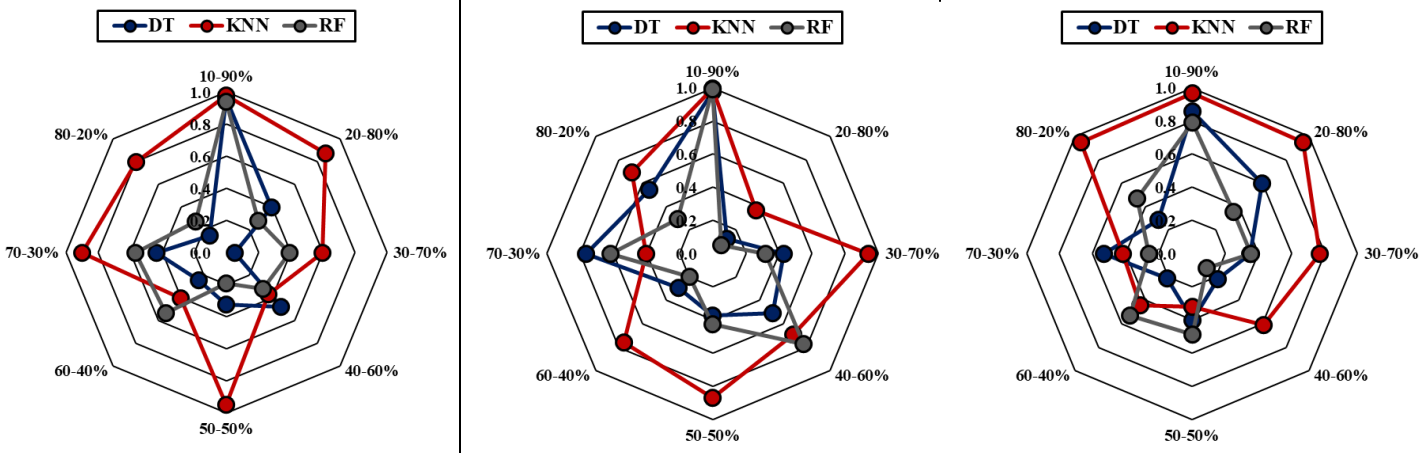


Pc = 1000, q | Pc = 2000, p | Pc = 2000, q

Undrained (e = 0.833)



Pc = 100, p | Pc = 100, q | Pc = 1000, p



Pc = 1000, q | Pc = 2000, p | Pc = 2000, q

Fig. 7. (*continued*)

Table 5. The best results

Drained				Undrained (e=0.735)				Undrained (e=0.833)			
e = 0.810, q	DT	Test	0.9670	Pc = 100, p	DT	Test	0.9953	Pc = 100, p	DT	Test	0.9951
		Train	0.9875			Train	0.9979			Train	0.9964
		All	0.9772			All	0.9966			All	0.9957
	KNN	Test	0.9959		KNN	Test	0.9995		KNN	Test	0.9997
		Train	1.0000			Train	1.0000			Train	1.0000
		All	0.9980			All	0.9998			All	0.9998
	RF	Test	0.9775		RF	Test	0.9963		RF	Test	0.9995
		Train	0.9744			Train	0.9978			Train	0.9998
		All	0.9760			All	0.9971			All	0.9996
e = 0.810, ev	DT	Test	0.6230	Pc = 100, q	DT	Test	0.9954	Pc = 100, q	DT	Test	0.9154
		Train	0.9994			Train	0.9973			Train	0.9857
		All	0.7891			All	0.9964			All	0.9499
	KNN	Test	0.6404		KNN	Test	0.9993		KNN	Test	0.9243
		Train	1.0000			Train	1.0000			Train	1.0000
		All	0.8002			All	0.9997			All	0.9614
	RF	Test	0.7697		RF	Test	0.9972		RF	Test	0.8787
		Train	0.9817			Train	0.9974			Train	0.8270
		All	0.8692			All	0.9973			All	0.8525
e = 0.886, q	DT	Test	0.9465	Pc = 1000, p	DT	Test	0.9961	Pc = 1000, p	DT	Test	0.9962
		Train	0.9711			Train	0.9985			Train	0.9980
		All	0.9587			All	0.9973			All	0.9971
	KNN	Test	0.9956		KNN	Test	0.9994		KNN	Test	0.9970
		Train	1.0000			Train	1.0000			Train	1.0000
		All	0.9978			All	0.9997			All	0.9985
	RF	Test	0.9642		RF	Test	0.9985		RF	Test	0.9986
		Train	0.9695			Train	0.9989			Train	0.9994
		All	0.9668			All	0.9987			All	0.9990
e = 0.886, ev	DT	Test	0.9520	Pc = 1000, q	DT	Test	0.9907	Pc = 1000, q	DT	Test	0.9643
		Train	0.9645			Train	0.9923			Train	0.9542
		All	0.9582			All	0.9915			All	0.9593
	KNN	Test	0.9960		KNN	Test	0.9954		KNN	Test	0.9576
		Train	1.0000			Train	1.0000			Train	1.0000
		All	0.9980			All	0.9977			All	0.9786
	RF	Test	0.9360		RF	Test	0.9890		RF	Test	0.9366
		Train	0.9267			Train	0.9876			Train	0.9386
		All	0.9313			All	0.9883			All	0.9376
e = 0.960, q	DT	Test	0.9493	Pc = 2000, p	DT	Test	0.9948	Pc = 2000, p	DT	Test	0.9747
		Train	0.9578			Train	0.9968			Train	0.9806
		All	0.9535			All	0.9958			All	0.9776
	KNN	Test	0.9939		KNN	Test	0.9991		KNN	Test	0.9944
		Train	1.0000			Train	1.0000			Train	1.0000
		All	0.9970			All	0.9995			All	0.9972
	RF	Test	0.9637		RF	Test	0.9988		RF	Test	0.9918
		Train	0.9858			Train	0.9991			Train	0.9933
		All	0.9747			All	0.9989			All	0.9925
e = 0.960, ev	DT	Test	0.9746	Pc = 2000, q	DT	Test	0.8486	Pc = 2000, q	DT	Test	0.8486
		Train	0.9720			Train	0.8648			Train	0.8648
		All	0.9733			All	0.8567			All	0.8567
	KNN	Test	0.9983		KNN	Test	0.9379		KNN	Test	0.9379
		Train	1.0000			Train	1.0000			Train	1.0000
		All	0.9992			All	0.9685			All	0.9685
	RF	Test	0.9608		RF	Test	0.8240		RF	Test	0.8240
		Train	0.9703			Train	0.7586			Train	0.7586
		All	0.9655			All	0.7906			All	0.7906

Table 6. The optimal parameter values for ML methods using PSO

Drained				Undrained (e=0.735)				Undrained (e=0.833)			
e = 0.810, q	DT	MinLeafSize	2	Pc = 100, p	DT	MinLeafSize	5	Pc = 100, p	DT	MinLeafSize	5
		MaxNumSplits	78			MaxNumSplits	53			MaxNumSplits	48
	KNN	NumNeighbors	1		KNN	NumNeighbors	1		KNN	NumNeighbors	1
		DistanceMetric	5			DistanceMetric	2.12			DistanceMetric	5
	RF	NumTrees	20		RF	NumTrees	10		RF	NumTrees	142
		NumPredictorsToSample	3			NumPredictorsToSample	3			NumPredictorsToSample	3
e = 0.810, ev	DT	MinLeafSize	5	Pc = 100, q	DT	MinLeafSize	4	Pc = 100, q	DT	MinLeafSize	2
		MaxNumSplits	93			MaxNumSplits	94			MaxNumSplits	64
	KNN	NumNeighbors	1		KNN	NumNeighbors	1		KNN	NumNeighbors	1
		DistanceMetric	4.405			DistanceMetric	5			DistanceMetric	2.36
	RF	NumTrees	96		RF	NumTrees	10		RF	NumTrees	10
		NumPredictorsToSample	4			NumPredictorsToSample	3			NumPredictorsToSample	1
e = 0.886, q	DT	MinLeafSize	2	Pc = 1000, p	DT	MinLeafSize	4	Pc = 1000, p	DT	MinLeafSize	2
		MaxNumSplits	100			MaxNumSplits	94			MaxNumSplits	100
	KNN	NumNeighbors	1		KNN	NumNeighbors	1		KNN	NumNeighbors	1
		DistanceMetric	2.14			DistanceMetric	1.46			DistanceMetric	3.35
	RF	NumTrees	10		RF	NumTrees	47		RF	NumTrees	34
		NumPredictorsToSample	4			NumPredictorsToSample	3			NumPredictorsToSample	3
e = 0.886, ev	DT	MinLeafSize	2	Pc = 1000, q	DT	MinLeafSize	4	Pc = 1000, q	DT	MinLeafSize	2
		MaxNumSplits	37			MaxNumSplits	92			MaxNumSplits	100
	KNN	NumNeighbors	1		KNN	NumNeighbors	1		KNN	NumNeighbors	1
		DistanceMetric	3.5			DistanceMetric	5			DistanceMetric	3.11
	RF	NumTrees	10		RF	NumTrees	15		RF	NumTrees	10
		NumPredictorsToSample	2			NumPredictorsToSample	3			NumPredictorsToSample	3
e = 0.960, q	DT	MinLeafSize	2	Pc = 2000, p	DT	MinLeafSize	4	Pc = 2000, p	DT	MinLeafSize	2
		MaxNumSplits	100			MaxNumSplits	82			MaxNumSplits	100
	KNN	NumNeighbors	1		KNN	NumNeighbors	1		KNN	NumNeighbors	1
		DistanceMetric	1			DistanceMetric	1.29			DistanceMetric	1
	RF	NumTrees	10		RF	NumTrees	51		RF	NumTrees	113
		NumPredictorsToSample	2			NumPredictorsToSample	3			NumPredictorsToSample	3
E = 0.960, ev	DT	MinLeafSize	2	Pc = 2000, q	DT	MinLeafSize	3	Pc = 2000, q	DT	MinLeafSize	2
		MaxNumSplits	100			MaxNumSplits	95			MaxNumSplits	64
	KNN	NumNeighbors	1		KNN	NumNeighbors	1		KNN	NumNeighbors	1
		DistanceMetric	1			DistanceMetric	1			DistanceMetric	1
	RF	NumTrees	10		RF	NumTrees	10		RF	NumTrees	10
		NumPredictorsToSample	3			NumPredictorsToSample	3			NumPredictorsToSample	1

C. Statistical Criteria in Measuring the Accuracy of ML Methods

These statistical metrics—R^2, RMSE, and MAD—are essential for evaluating the performance of ML models in this study (Table 7). Specifically, they quantify the accuracy and reliability of the predictions generated by the ML algorithms when compared to the reference values obtained from the validated constitutive model. The R^2 value indicates how well the predicted results explain the variance in the data, while RMSE and MAD provide insights into the average magnitude of prediction errors. These measures are significant for comparing the effectiveness of different ML models (DT, KNN, and RF) across various stress conditions and soil states. By incorporating these metrics, the study ensures a rigorous and objective assessment of model performance, which is critical for identifying the most accurate and generalizable prediction approach for sand constitutive behavior.

By carefully examining these results, a more precise and comprehensive analysis of the models' performance can be achieved. This detailed evaluation provides deeper insights into the performance differences among the various algorithms applied in this study. Understanding these distinctions is crucial for selecting the most suitable model for predictive tasks within the specific context of the dataset.

An important observation arises with the drained test conducted at an initial void ratio of 0.81. For this particular case, the predictions generated by the DT and RF algorithms deviate noticeably from the corresponding modeling results. This inconsistency suggests limitations in the predictive capability of these methods under certain conditions or data characteristics. On the other hand, the KNN technique stands out by producing predictions that closely align with the modeled outcomes, indicating a superior fit for this dataset and scenario. In other words, the KNN algorithm demonstrates enhanced performance compared to both RF and DT across several key evaluation metrics. One of the primary reasons for the effectiveness of KNN lies in its simplicity and adaptability. Unlike more complex models that require extensive training and tuning, KNN operates based on straightforward principles. It identifies the closest data points in the feature space relative to a given test sample

and makes predictions grounded in these local neighborhoods. This local approach allows KNN to capture subtle patterns and relationships that might be missed by more global models. This characteristic makes KNN particularly suitable for datasets characterized by complex, non-linear relationships. Since it does not rely on rigid assumptions about data distribution or underlying functional forms, KNN can flexibly adjust to varying data structures and complexities. Furthermore, the results from our dataset indicate that KNN consistently delivers reliable and stable outcomes, reflected in its superior values of accuracy (R^2), MAD, and RMSE.

Specifically, KNN achieves lower error rates and higher precision compared to RF and DT, especially in cases where data points are closely related and exhibit discernible patterns. Its ability to handle small variations within the data also contributes significantly to its robustness. Even in the presence of noise or outliers, KNN maintains a high level of predictive accuracy, a property highly desirable in practical applications where data quality can be variable.

Table 7. Plotted error values of ML methods

				Drained			Undrained (e=0.735)			Undrained (e=0.833)		
e = 0.810, q	MAD	DT	Test	36.377	Pc = 100, p	DT	Test	33.825	Pc = 100, p	DT	Test	13.56
			Train	25.061			Train	22.828			Train	10.131
		KNN	Test	10.798		KNN	Test	9.639		KNN	Test	3.078
			Train	0			Train	0			Train	0
		RF	Test	33.926		RF	Test	29.941		RF	Test	4.489
			Train	39.092			Train	21.01			Train	3.152
	RMSE	DT	Test	71.698		DT	Test	60.079		DT	Test	21.044
			Train	43.595			Train	40.205			Train	17.47
		KNN	Test	25.487		KNN	Test	19.325		KNN	Test	5.413
			Train	0			Train	0			Train	0
		RF	Test	59.596		RF	Test	53.519		RF	Test	6.985
			Train	74.488			Train	40.534			Train	4.858
e = 0.810, ev	MAD	DT	Test	0.0101	Pc = 100, q	DT	Test	38.616	Pc = 100, q	DT	Test	9.841
			Train	0.0063			Train	30.176			Train	2.933
		KNN	Test	0.0065		KNN	Test	14.067		KNN	Test	6.197
			Train	0			Train	0			Train	0
		RF	Test	0.0099		RF	Test	30.231		RF	Test	7.047
			Train	0.0078			Train	22.792			Train	14.816
	RMSE	DT	Test	0.1005		DT	Test	74.296		DT	Test	54.774
			Train	0.0794			Train	55.351			Train	12.641
		KNN	Test	0.0806		KNN	Test	28.594		KNN	Test	39.396
			Train	0			Train	0			Train	0
		RF	Test	0.0995		RF	Test	60.607		RF	Test	41.854
			Train	0.0883			Train	49.469			Train	76.67
e = 0.886, q	MAD	DT	Test	42.344	Pc = 1000, p	DT	Test	20.771	Pc = 1000, p	DT	Test	3.223
			Train	24.094			Train	12.166			Train	2.22
		KNN	Test	11.245		KNN	Test	7.55		KNN	Test	1.652
			Train	0			Train	0			Train	0
		RF	Test	200.973		RF	Test	12.493		RF	Test	1.87
			Train	244.016			Train	8.952			Train	1.157
	RMSE	DT	Test	87.985		DT	Test	37.174		DT	Test	5.142
			Train	58.421			Train	22.116			Train	3.689
		KNN	Test	25.233		KNN	Test	14.081		KNN	Test	4.576
			Train	0			Train	0			Train	0
		RF	Test	379.108		RF	Test	23.35		RF	Test	3.09
			Train	429.605			Train	19.554			Train	2.037
e = 0.886, ev	MAD	DT	Test	0.001	Pc = 1000, q	DT	Test	38.154	Pc = 1000, q	DT	Test	10.529
			Train	0.001			Train	31.902			Train	10.915
		KNN	Test	0		KNN	Test	17.454		KNN	Test	6.503
			Train	0			Train	0			Train	0
		RF	Test	0.001		RF	Test	28.361		RF	Test	10.662
			Train	0.001			Train	29.568			Train	11.34
	RMSE	DT	Test	0.002		DT	Test	76.369		DT	Test	32.924
			Train	0.001			Train	71.728			Train	41.821
		KNN	Test	0		KNN	Test	56.031		KNN	Test	40.582
			Train	0			Train	0			Train	0
		RF	Test	0.002		RF	Test	84.145		RF	Test	44.509
			Train	0.002			Train	89.078			Train	43.386

Table 7. (*continued*)

Drained					Undrained (e=0.735)				Undrained (e=0.833)			
e = 0.960, q	MAD	DT	Test	46.477	Pc = 2000, p	DT	Test	13.094	Pc = 2000, p	DT	Test	10.493
			Train	37.516			Train	8.808			Train	6.279
		KNN	Test	14.883		KNN	Test	4.561		KNN	Test	3.824
			Train	0			Train	0			Train	0
		RF	Test	40.896		RF	Test	5.911		RF	Test	5.058
			Train	27.031			Train	5.008			Train	5.974
	RMSE	DT	Test	82.376		DT	Test	24.275		DT	Test	29.883
			Train	65.773			Train	17.398			Train	20.854
		KNN	Test	27.536		KNN	Test	10.007		KNN	Test	12.82
			Train	0			Train	0			Train	0
		RF	Test	68.232		RF	Test	11.575		RF	Test	15.584
			Train	48.267			Train	9.641			Train	17.782
e = 0.960, ev	MAD	DT	Test	0.003	Pc = 2000, q	DT	Test	46.779	Pc = 2000, q	DT	Test	8.501
			Train	0.003			Train	34.371			Train	13.546
		KNN	Test	0.001		KNN	Test	11.901		KNN	Test	5.537
			Train	0			Train	0			Train	0
		RF	Test	0.003		RF	Test	22.61		RF	Test	9.18
			Train	0.003			Train	27.727			Train	12.628
	RMSE	DT	Test	0.003		DT	Test	115.554		DT	Test	44.504
			Train	0.003			Train	101.189			Train	62.195
		KNN	Test	0.001		KNN	Test	39.792		KNN	Test	31.964
			Train	0			Train	0			Train	0
		RF	Test	0.004		RF	Test	91.044		RF	Test	56.449
			Train	0.004			Train	88.456			Train	71.665

V. CONCLUSION

A critical state constitutive model developed at the University of Alberta (Canada) was employed to simulate the behavior of sands. To ensure the validity and authenticity of the numerical results generated by this model, the constitutive formulation was implemented numerically with the consistency condition satisfied for all strain increments. A single, consistent set of model parameters was used across all predictions, and the comparison between experimental data and constitutive model outputs confirmed the sound predictive capability of the model. Given the common challenge of the lack of continuous and incremental experimental data, the use of a previously validated constitutive model offers a reliable alternative for predicting the behavior of sands. Accordingly, the critical state constitutive model was adopted here specifically for simulating Toyoura sand constitutive behaviors.

The study further evaluated the drained and undrained constitutive behaviors of Toyoura sand under triaxial compression monotonic loadings by applying three ML algorithms: Decision Tree (DT), K-Nearest Neighbors (KNN), and Random Forest (RF). The performance of these algorithms was rigorously assessed using metrics such as the coefficient of determination (R^2), Mean Absolute Deviation (MAD), and Root Mean Square Error (RMSE). Among the models tested, KNN consistently demonstrated superior performance. Under drained conditions, KNN accurately predicted both deviatoric stress (q) and volumetric strain (ev) across different initial void ratios (e), with high R^2 values and minimal errors. Similarly, in undrained conditions with varying confining pressures (Pc) and void ratios, KNN maintained high predictive accuracy and robustness, as evidenced by near-zero training errors and strong agreement with experimental results. These findings underscore the capability of KNN to model complex constitutive behaviors reliably, outperforming DT and RF in this context.

Overall, this research illustrates that ML algorithms, especially KNN, serve as powerful tools for predicting the drained and undrained constitutive responses of Toyoura sand with high accuracy. These methods are effective in handling complex datasets and provide valuable insights for geotechnical engineering applications. Considering the highly complex and nonlinear nature of sand behavior under cyclic loadings involving multiple unloading and reloading cycles with hysteresis loops, future work will focus on capturing these intricate patterns through advanced ML techniques. In particular, deep learning architectures such as Long Short-Term Memory (LSTM) neural networks and Convolutional Neural Networks (CNN) will be explored to model cyclic loading data more effectively.

REFERENCES

Chen, W., Olarte, A. A. P., & Cudmani, R. (2021). Modelling the monotonic and cyclic behaviour of sands using Artificial Neural Networks. In EPJ Web of Conferences (Vol. 249, p. 11015). EDP Sciences.

De Silva, L. I. N., & Koseki, J. (2012). Modelling of sand behavior in drained cyclic shear. In 2nd International Conference on Transportation Geotechnics (ICTG) International Society of Soil Mechanics and Geotechnical Engineering (ISSMGE).

De Ville, B. (2013). Decision trees. Wiley Interdisciplinary Reviews: Computational Statistics, 5(6), 448-455.

Dornheim, J., Morand, L., Nallani, H. J., & Helm, D. (2024). Neural networks for constitutive modeling: From universal function approximators to advanced models and the integration of physics. Archives of computational methods in engineering, 31(2), 1097-1127.

Eghbalian, M., Pouragha, M., & Wan, R. (2023). A physics-informed deep neural network for surrogate modeling in classical elasto-plasticity. Computers and Geotechnics, 159, 105472.

Gao, W. (2018). A comprehensive review on identification of the geomaterial constitutive model using the computational intelligence method. Advanced Engineering Informatics, 38, 420-440.

Guan, Q. Z., & Yang, Z. X. (2023). Hybrid deep learning model for prediction of monotonic and cyclic responses of sand. Acta Geotechnica, 18(3), 1447-1461.

Imam, S. M. (1999). Modeling the constitutive behavior of sand for the analysis of static liquefaction.

Imam, S. R., Morgenstern, N. R., Robertson, P. K., & Chan, D. H. (2005). A critical-state constitutive model for liquefiable sand. Canadian geotechnical journal, 42(3), 830-855.

Kohestani, V. R., & Hassanlourad, M. (2016). Modeling the mechanical behavior of carbonate sands using artificial neural networks and support vector machines. International Journal of Geomechanics, 16(1), 04015038.

Latini, C., Zania, V., & Tamagnini, C. (2017). Modelling of constitutive behavior of sand in the low stress regime: an implementation of SANISAND. In 19th International Conference on Soil Mechanics and Geotechnical Engineering.

Liu, Y., Liang, Z., Liu, Z., & Nie, G. (2022). Post-cyclic drained shear behaviour of Fujian sand under various loading conditions. Journal of Marine Science and Engineering, 10(10), 1499.

Najjar, Y., & Zhang, X. C. (2000). Characterizing the 3D stress-strain behavior of sandy soils: A neuro-mechanistic approach. In Numerical methods in geotechnical engineering (pp. 43-57).

Noor, T., Lone, S. N., Ramana, G. V., & Nayek, R. (2025). A recursive Bayesian neural network for constitutive modeling of sands under monotonic loading. arXiv preprint arXiv:2501.10088.

Peterson, L. E. (2009). K-nearest neighbor. Scholarpedia, 4(2), 1883.

Pouragha, M., Eghbalian, M., & Wan, R. (2020). A note on applicability of artificial intelligence to constitutive modeling of geomaterials. Journal of Machine Learning for Modeling and Computing, 1(2).

Rezazadeh Eidgahee, D., & Shiri, H. (2024). Machine Learning–Enhanced Modeling of Stress–Strain Behavior of Frozen Sandy Soil. Geotechnics, 4(4), 1228-1245.

Rigatti, S. J. (2017). Random forest. Journal of insurance medicine, 47(1), 31-39.

Sassel, T. S., & O'Sullivan, C. (2024). Advancing understanding of the influence of drained cyclic loading on sand behavior using DEM. Journal of Engineering Mechanics, 150(1), 04023110.

Su, M., Guo, N., & Yang, Z. (2023). A multifidelity neural network (MFNN) for constitutive modeling of complex soil behaviors. International Journal for Numerical and Analytical Methods in Geomechanics, 47(18), 3269-3289.

Tarhouni, A., & Amer, M. (2021). Behaviour of sand in monotonic and cyclic simple shear loading at low-stress level (Doctoral dissertation, Memorial University of Newfoundland).

Wang, M., Kumar, K., Feng, Y. T., Qu, T., & Wang, M. (2024). Machine learning aided modeling of granular materials: a review. Archives of Computational Methods in Engineering, 1-38.

Wu, M., & Wang, J. (2022). Constitutive modelling of natural sands using a deep learning approach accounting for particle shape effects. Powder Technology, 404, 117439.

Wu, M., Xia, Z., & Wang, J. (2023). Constitutive modelling of idealised granular materials using machine learning method. Journal of Rock Mechanics and Geotechnical Engineering, 15(4), 1038-1051.

Yao, X., Ma, S., Li, B., Liu, H., Bai, J., & Bai, Z. (2025). Short-Sequence Machine Learning Framework for Predicting Constitutive Relationships of Sand. Geotechnical and Geological Engineering, 43(2), 101.

Yeh, F. H., Tafili, M., Prada-Sarmiento, L. F., Ge, L., & Wichtmann, T. (2023). Inspection of two sophisticated models for sand based on generalized plasticity: Monotonic loading and Monte Carlo analysis. International Journal for Numerical and Analytical Methods in Geomechanics, 47(3), 425-456.

Zhang, P., Yin, Z. Y., & Jin, Y. F. (2021). State-of-the-art review of machine learning applications in constitutive modeling of soils. Archives of Computational Methods in Engineering, 28(5), 3661-3686.

Zhang, P., Yin, Z. Y., & Sheil, B. (2023). Interpretable data-driven constitutive modelling of soils with sparse data. Computers and Geotechnics, 160, 105511.

Zhang, Y., Qiu, J., Zhang, Y. G., & Liao, R. (2022). The establishment of a constitutive model of sand under monotonic loading by adopting the support vector machine (SVM). Arabian Journal for Science and Engineering, 47(4), 4421-4435.