Machine Learning-based Modelling of Soil Properties for Geotechnical Design: Review, Tool Development and Comparison

Pin ZHANG[1], Zhen-Yu YIN[1,*] and Yin-Fu JIN[1]

1 Department of Civil and Environmental Engineering, Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, China

* Corresponding author: Dr Zhen-Yu YIN, Tel: +852 3400 8470; Fax: +852 2334 6389; E-mail: zhenyu.yin@polyu.edu.hk; zhenyu.yin@gmail.com

Abstract: Machine learning (ML) holds significant potential for predicting soil properties in geotechnical design but at the same time poses challenges, including those of how to easily examine the performance of an algorithm and how to select an optimal algorithm. This study first comprehensively reviewed the application of ML algorithms in modelling soil properties for geotechnical design. The algorithms were categorized into several groups based on their principles, and the main characteristics of these ML algorithms were summarized. After that six representative algorithms are further detailed and selected for the creation of a ML-based tool with which to easily build ML-based models. Interestingly, automatic determination of the optimal configurations of ML algorithms is developed, with an evaluation of model accuracy, application of the developed ML model to the new data and investigation of relationships between the input variables and soil properties. Furthermore, a novel ranking index is proposed for the model comparison and selection, which evaluates a ML-based model from five aspects. Soil maximum dry density is selected as an example to allow examination of the performance of different ML algorithms, the applicability of the tool and the model ranking index to determining an optimal model.

Keywords: Machine learning; model selection; correlations; soils; physical properties; mechanical properties

List of notations:

$\boldsymbol{b}_i$: bias vector of the ith hidden layer

$\boldsymbol{c}$: constant coefficient vector

C: regularization parameter

$\mathbf{E}$: exponent matrix

F: function set

gen: number of iterations

$\mathbf{H}_i$: output of the ith hidden layer

m: number of datasets

$mtry$: number of features at each node

n: dimension of input variables

n_t: dimension of transformed variables

$ntree$: number of decision trees

N: stochastic calculation times

p: dropout probability

p_c: probability of crossover

p_m: probability of mutation

pop: size of population

r: Bernoulli distribution with probability of p

$\mathbf{W}_i$: weight matrix of the ith hidden layer

$x_{i,\max}$: maximum value of the variable $\boldsymbol{x}_i$

$x_{i,\min}$: minimum value of the variable $\boldsymbol{x}_i$

x_{norm}: normalized value of a dataset

$\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$: matrix of input variables

$\mathbf{XT}$: matrix of transformed variables

y_i^a: actual value of the output variable

y_i^p: predicted value of the output variable

$\overline{y}_i^a$: mean value of the actual output variable

$\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$: output of the output layer

γ: kernel coefficient

ξ: slack parameter (default value: 0.1).

σ: activation function

$\mathbb{E}$: mean value of output

Introduction

Estimation of soil physical and mechanical properties is an inevitable step in geotechnical design for achieving the trade-off between the cost and engineering safety, such as the estimation of bearing capacity and long-term deformation.~~, and~~ Site-specific laboratory testing is the most direct way of obtaining soil properties. For a certain type of soils, empirical or semi-empirical correlations have been developed for use in predicting soil properties [1-7]. However, these conventional regression-based correlations have limited prediction capabilities and cannot uncover the essential interconnections between input variables and the studied soil properties, hindering the development of a general prediction model and the use of existing data.

Machine learning (ML) provides novel insights into this issue because of its strong nonlinear fitting capability [8-14]. ML algorithms directly learn the relationships between the governing input variables and the studied soil properties [15-17]. In view of the large volume of research works regarding the application of ML algorithms to the prediction of soil properties, herein the well-known articles in the engineering, geology and methodology categories have been checked based on the Web of Science (Fig. 1). This type of research shows exponential growth and has prevailed since 2018 associated with the blossoming of ML.

The rapid development of ML has created a significant opportunity for predicting soil properties but at the same time poses challenges, such as how to easily assess an algorithm's performance and how to select an optimal algorithm [18-23]. Thus, an easy-to-use tool is needed that can assist researchers or engineers in rapidly developing various ML-based models

so that they can avoid having to learn ML from scratch. The development of such a tool would allow ready comparison of the performance of different ML-based models and selection of the optimal model based on a reasonable criterion.

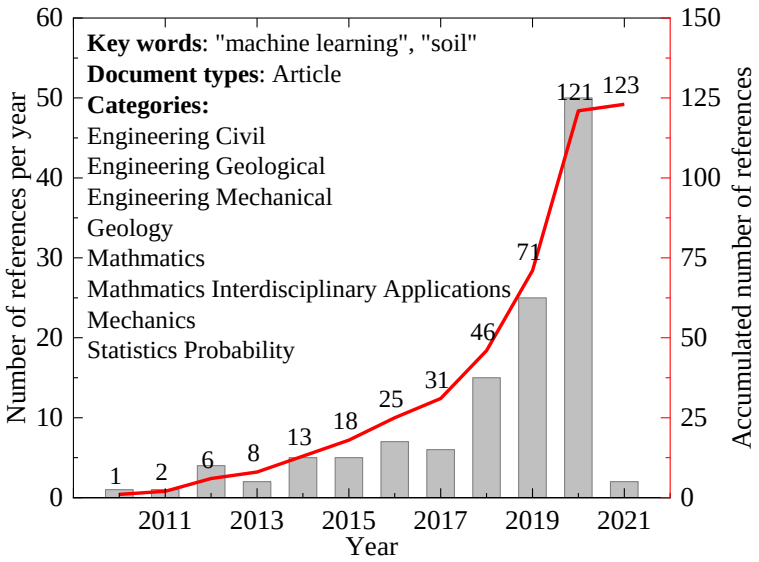


Fig. 1 Number of publications relating to development and application of ML algorithms to predicting soil properties

This study first reviews the application of ML algorithms to predict soil properties, with more specified attention on six representative ML algorithms, i.e. genetic programming (GP), exponential polynomial regression (EPR), support vector regression (SVR), random forest (RF), feedforward neural network (FNN) and Monte Carlo dropout based artificial neural network (ANN_MCD), for application to prediction of soil properties. An easy-to-use ML-based modelling tool, a graphical user interface (GUI) platform co-called ErosMLM, is developed for use in building these ML-based models, with functions including automatic determination of optimal configurations of ML algorithms, evaluation of model accuracy, application of the developed ML-based model to new data and investigation of relationships between the input variables and soil properties. Meanwhile, a novel ranking index is proposed

for model selection, which evaluates a ML-based model from five aspects, such as accuracy for the training set, accuracy for the testing set, ability to exploit and explore new data, monotonicity evaluation and model complexity. Soil maximum dry density is taken as an example when examining the developed tool, comparing model performance and selecting a model.

Review of ML algorithms for modelling of soil properties

Overview of applied ML algorithms

Linear and basic nonlinear regression methods aside, ML algorithms can be classified into different categories in geotechnical practice, as Table 1 shows. Symbolic regression algorithm-based models such as GP and EPR can be presented through an explicit formulation and thus are easy to use. SVR and tree-based algorithms such as RF are two types of classical ML algorithms that have shown excellent performance when handling high-dimensional data. FNN has an exceptional fitting capacity, particularly its network complexity can increase its adaptability to a high volume of data. A Bayesian-based model is characterized by uncertainty representation, in which the final prediction is assigned through an uncertainty evaluation. It should be noted that integrated ANN and Monte Carlo dropout (ANN_MCD) was recently proposed for approximating Bayesian interference [24-26] and showed similarly excellent performance [27]. ~~but has not yet been used in geotechnical design and~~ ANN_MCD is thus worth trying, because the uncertain algorithm is suitable for the reliability-based design in geotechnical engineering [28-31].

To this end, six ML algorithms – GP, EPR, SVR, RF, FNN and ANN_MCD (Fig. 2) –

were ultimately selected and implemented for use in the development of ErosMLM. The principles of these algorithms are briefly introduced in the following sections. Their advantages or drawbacks are compared in the later sections.

Table 1 Classification of ML algorithms adopted for modelling soil properties

Classification	Algorithm	Main characteristics	Relevant publications
symbolic regression	MARS	simple and explicit expression	pile drivability [32]
	GP		soil suction [33]
	EPR		soil compressibility [34]
SVM	SVR	structural risk minimization	landslide displacement [35]
	LSSVM		settlement [36]
tree-based	DT	flowchart-like structure	soil erosion [37]
	GBDT	ensemble algorithm	stope displacement [38]
	RF		creep behavior [39]
ELM	ELM	single layer feedforward	soil temperature [40]
k-NN	*k*-NN	instance-based learning	soil moisture [41]
ANN	FNN	error backpropagation	tunneling-induced responses [42]
	RBFNN	single layer feedforward	hydraulic conductivity [43]
	GRNN	single layer feedforward	loadbearing capacity of piles [44]
Bayesian	BN	uncertainty	tunnel squeezing [45]
	BNN		drilled shafts [46]

Note: MARS = multivariate adaptive regression splines; GP = genetic programming; EPR = evolutionary polynominal regression; SVM = support vector machine; SVR = support vector regression; LSSVM = least-squares support vector machine; DT = decision tree; GBDT = gradient boosting decision tree; RF = random forest; ELM = extreme learning machine; *k*-NN = *k*-nearest neighbor; FNN = feedforward neural network; RBFNN = radial basis function neural network; GRNN = general regression neural network; BN = Bayesian network; BNN = Bayesian neural network.

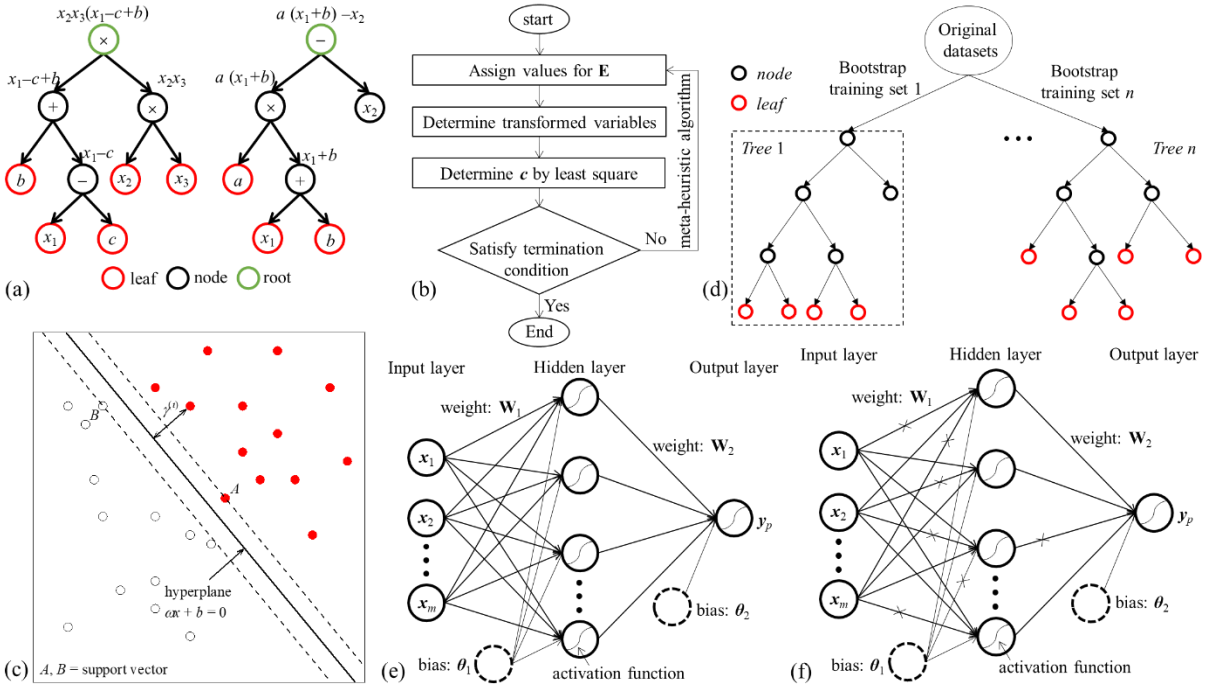


Fig. 2 Schematic view of internal mechanism of adopted six ML algorithms: (a) GP; (b) EPR; (c) SVR; (d) RF; (e) FNN; (f) ANN_MCD

Genetic programming

GP is a type of symbolic regression method, consisting of terminals and functions [33,47] (Fig. 2a). Terminals are conceptionally identical to the input variables $\mathbf{X} = (x_1, x_2, \ldots, x_n)$, and the function set F is a combination of various arithmetic operation such as {+, –, ×, /} [48]. Therefore, the output y of GP can be expressed using an explicit formulation using:

$$y = f\left(\mathbf{X}, F, \boldsymbol{c}\right) \tag{1}$$

where $\boldsymbol{c}$ is a constant coefficient vector, which is generally obtained by the least square. The binary tree is the commonly used architecture in GP. The growth of a tree is controlled by the loss function, which is generally defined as the difference between the predicted and actual result. The tree evolves until obtaining the optimal structure that can minimize the loss value. Meanwhile, the operations during the evolution of the tree involve crossover and mutation. Crossover means the branch under a node has a probability of p_c to be exchanged with different

branches under other nodes, and the mutation means the arithmetic operation at each node has a probability of p_m to be randomly changed.

Evolutionary polynomial regression

EPR is also a type of symbolic regression method [49] and its primary characteristic is that the original input variables are combined to form new ~~random number of~~ transformed variables **XT** ~~are pre-defined~~ (Fig. 2b). Each transformed variable is the combination of original input variables, thereby the output $\boldsymbol{y}$ of EPR can be obtained using:

$$\boldsymbol{y} = f\left(\mathbf{X}, \mathbf{E}, \boldsymbol{c}\right) \tag{2}$$

where **E** is an exponent matrix, and it is generally determined using meta-heuristic algorithms such as genetic algorithm and particle swarm optimization (PSO) [50-52]. The **XT** can be obtained by:

$$\mathbf{XT}_{i,\ j} = \mathbf{X}_{i,1}^{\mathbf{E}_{j,1}} \mathbf{X}_{i,2}^{\mathbf{E}_{j,2}} \ldots \mathbf{X}_{i,n}^{\mathbf{E}_{j,n}} \tag{3}$$

where n is the dimension of **X**. The ~~determination of the~~ dimension of **XT** (n_t) and the values of elements in the exponent matrix are required to be prescribed, which are vitally important for the performance of EPR.

Support vector regression

SVR is characterized by structural risk minimization (Fig. 2c). For datasets that cannot be separated in the low-dimensional space, SVR can use kernel function to map datasets into a more high-dimensional space to find a hyperplane to separate all datasets [53]. The training of SVR is to find a hyperplane that can separate all datasets with a largest "gap", which can be mathematically expressed using:

$$\min_{\xi,\mathbf{W},\boldsymbol{b}} \frac{1}{2}\|\mathbf{W}\|^2 + C\sum_{i=1}^{m}\xi_i$$
$$s.t.\ \boldsymbol{y}\left[\left(\mathbf{W}\right)^T \phi\left(\mathbf{X}\right)+\boldsymbol{b}\right] \geq 1-\xi_i,\ i=1,2,\ldots,m,\ \xi_i \geq 0 \quad (4)$$

where m is a total of datasets $\mathbf{X}$; ξ is the slack parameter, indicating the allowable error; C is the regularization parameter for imposing a penalty on error. The radial basis function is the commonly used kernel type, which can be obtained using:

$$\phi\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \exp\left(-\gamma\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2\right) \quad (5)$$

where γ is the kernel coefficient.

Random forest

Random forest (RF) is an ensemble ML algorithm, consisting of numerous decision trees (DTs) [54], in which each DT is an independent predictor (Fig. 2d). The final output of RF is the mean value of outputs generated by all DTs. Therefore, the most important hyper-parameters in RF are the number of DTs (*ntree*) and the number of features (*mtry*) at each node for dividing datasets. The number of datasets used for developing DT is determined using the bagging method [55], which generates a bootstrap set by sampling with replacement. In general, each bootstrap set based on bagging includes 2/3 datasets of the training set, and the remaining 1/3 referred to as out of the bag are used to estimate the error of DT [56].

$$\boldsymbol{y} = \frac{1}{ntree}\sum_{i=1}^{ntree}\boldsymbol{y}_i\left(\mathbf{X}\right) \quad (6)$$

Feedforward neural network

FNN is the most commonly used ML algorithm, consisting of input, hidden and output layers (Fig. 2e). The input data flows from the input layer to the output layer and the error is

propagated from the output layer for updating all weights and biases until the difference between the predicted and actual results minimizes [57]. The weights and biases are generally updated using the gradient descend algorithm. The mathematic formulations of FNN can be expressed by:

$$\mathbf{H} = \sigma\left(\mathbf{W}_1\mathbf{X} + \boldsymbol{b}_1\right) \tag{7}$$

$$\boldsymbol{y} = \sigma\left(\mathbf{W}_2\mathbf{H} + \boldsymbol{b}_2\right) \tag{8}$$

where σ is the activation function; $\mathbf{W}$ and $\boldsymbol{b}$ are the weight matrix and bias vector, respectively; $\mathbf{H}$ is the output in the hidden layer.

Artificial neural network with Monte Carlo dropout

The predicted results of the formerly mentioned ML algorithms are deterministic, which means these algorithms can always output a fixed result as long as input a set of datasets (Fig. 2f). Gal and Ghahramani [24] thus proposed a novel theoretical framework that casts Monte Carlo dropout training in an ANN (ANN_MCD) to account for epistemic uncertainty. Dropout is a widely used overfitting prevention method [58], which inactivates each hidden neuron with a probability of p using:

$$\mathbf{H}_{i+1} = \sigma\left(r_i\mathbf{W}_{i+1}\mathbf{H}_i + \boldsymbol{b}_{i+1}\right), \; r_i \sim Bernoulli(p) \tag{9}$$

In the conventional ANN, dropout only activates in the training phase, but in ANN_MCD, it activates in both training and testing phases. It means each hidden neuron has a fixed probability to be inactivated in the testing phase, i.e., the architecture of the ANN is not consistent. The output is different at each implementation, and this characteristic is used to represent model uncertainty. By performing stochastic calculation T times, the final output and

uncertainty can be obtained using:

$$\mathbb{E} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{y}_i\left(\mathbf{X}, \mathbf{W}_i, \boldsymbol{b}_i\right) \tag{10}$$

$$Var = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{y}_i\left(\mathbf{X}, \mathbf{W}_i, \boldsymbol{b}_i\right)^T \boldsymbol{y}_i\left(\mathbf{X}, \mathbf{W}_i, \boldsymbol{b}_i\right) - \mathbb{E}^T\mathbb{E} \tag{11}$$

ANN_MCD has not been applying to model correlations of materials, but its output with an uncertainty evaluation is significant for the design and risk evaluation in civil engineering.

Summary and suggestions

Based on the introduction of above mentioned six ML algorithms, each algorithm has its advantages and limitations (Table 2). The selection of an optimal ML algorithm thus depends on a comprehensive evaluation and detailed requirements of the studied issue.

Table 2 Main advantages and limitations of adopted ML algorithms

Algorithms	Advantages	Limitations
GP	Explicit expression; Simplicity	Complex structure; Limited non-linear mapping ability; Single output prediction
EPR	Explicit expression; Simplicity	Limited non-linear mapping ability; single output prediction
SVM	Structural risk minimization; Fast training speed	Poor readability and interpretability; Single output prediction
RF	Strong non-linear mapping ability	Single output prediction; Poor extrapolation ability
FNN	Strong non-linear mapping ability; Multi-output prediction	Numerous hyper-parameters; Huge computational cost
ANN_MCD	Strong non-linear mapping ability; Multi-output prediction; Uncertainty evaluation	Numerous hyper-parameters; Huge computational cost

Development of ML-based modelling tool

Considering the fast development in the ML domain, ML algorithms have been increasingly proposed, which leads to difficulties in advancing the understanding of these algorithms and

applying them in engineering practice. To this end, the ErosMLM tool is developed, aiming to offer a platform to easily build ML-based models. The main interface is presented in Fig. 3a. It is programmed using Python language, and an open-source code *tkinter* is utilized for establishing a GUI framework. Four submodules are created: (1) "Help" is designed consisting of a manual for explaining the operation of ErosMLM; (2) "Develop" is designed to determine the optimal configuration of ML algorithms; (3) "Predict" is designed to directly use the existing ML-based models; and (4) "Exit" is designed for the close of ErosMLM. The development of this platform can be divided into six steps (see Fig. 3): (1) selecting a ML algorithm; (2) inputting the database in a required format; (3) preprocessing of datasets; (4) automatically determining the optimal configuration of the selected ML algorithm; (5) applying the ML-based model; (6) saving the results. The details of 3 key steps from 3 to 5 are presented as follows.

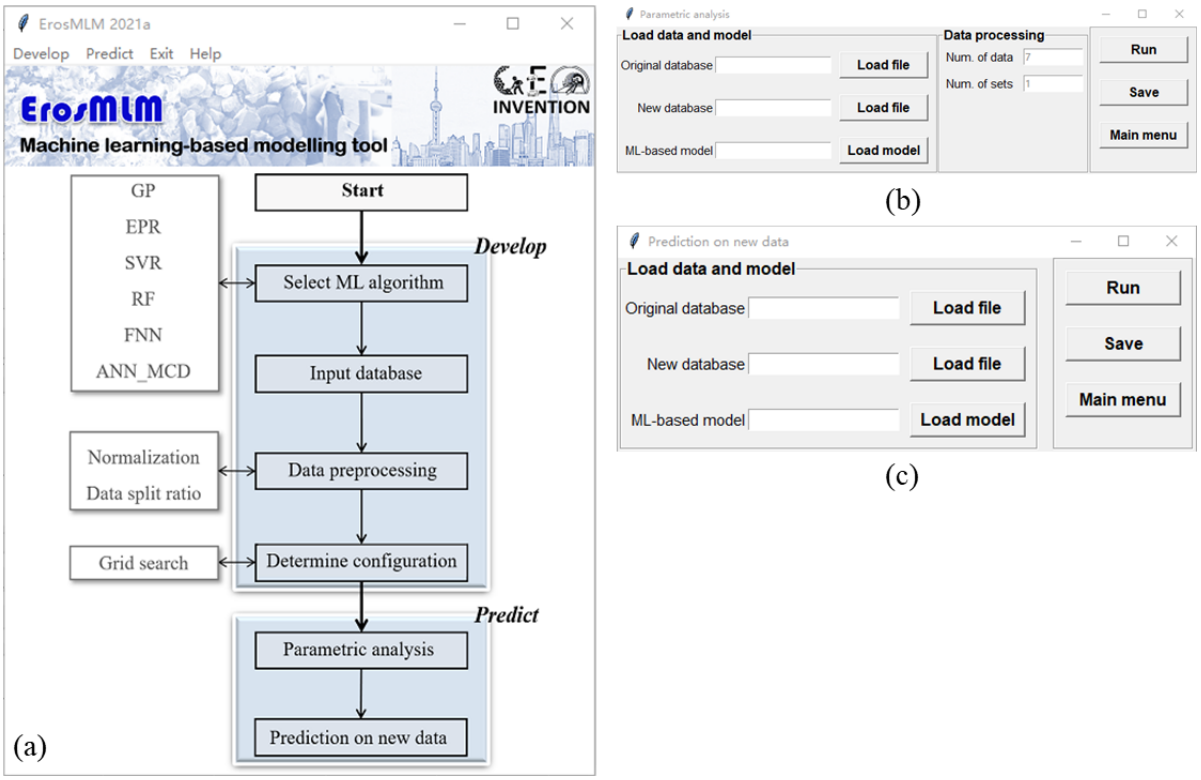


Fig. 3 Introduction of ErosMLM: (a) main interface; (b) "Parametric analysis" module; (c) "Prediction on new data" module

Data preprocessing

Herein, the former mentioned six algorithms can be selected. After selecting a ML algorithm, the implementation panel pops up. The grey value in each entry is the commonly used values, which guides users to assign values. The explanation of the label can also pop up when the cursor hovers over the label. The first step is to input datasets and implement data preprocessing (using the "Load file" button). The preprocessing of datasets involves the normalization, assignment of input and output dimensions and split of training and testing datasets. The min-max normalization method is set as the default (see Eq. (12)), in which all datasets are rescaled into the range (–1, 1) for eliminating the scaling effect of input variables. It only requires assigning the dimension of the output variable, thereafter the dimension of input variables can be automatically calculated. The split ratio specifies the proportion of datasets in the training set, which is generally set as 80%, and the data in the training and testing sets is determined by the random seed. It should be noted that additional processing procedures including *k*-fold cross-validation and batch size are implemented in the FNN and ANN_MCD algorithms, which are two commonly used strategies adopted in FNN and its variants for enhancing the robustness of the model and optimization process.

$$x_{norm} = \frac{2x - x_{i,\max} - x_{i,\min}}{x_{i,\max} - x_{i,\min}} \tag{12}$$

where $x_{i,\max}$ and $x_{i,\min}$ are the maximum and minimum values of the parameter $\boldsymbol{x}_i$, respectively; x_{norm} is the normalized value.

Automatic determination of optimal configuration

Once completing the data pre-processing, the datasets are employed to develop a ML-based model. A necessary task but difficult for beginners for developing a ML-based model is to determine its hyper-parameters. In the submodule "Develop", the grid search method is used to search the optimal combination of hyper-parameters because it can systematically search the whole studied space. The ranges of one or two hyper-parameters are pre-assigned, while the remaining hyper-parameters are fixed. The relationships between the performance of a ML-based model and the hyper-parameters can be comprehensively investigated under this condition. Herein, the coefficient of determination (R^2) is the only applied indicator, because multiple indicators may trigger contradiction for the model selection and R^2 is a suitable indicator to measure the agreement between the predicted and measured results.

$$R^2 = 1 - \frac{\sum_{i=1}^{m}\left(y_i^p - y_i^a\right)^2}{\sum_{i=1}^{m}\left(y_i^a - \bar{y}_i^a\right)^2} \tag{13}$$

where y_i^a and y_i^p are the actual and predicted values for the output variable, respectively; $\bar{y}_i^a$ is the mean value of the output variable. The "Training" button is designed by clicking which, the training process can be activated. The optimal configurations are thereafter determined and shown in the "Output" panel, and the relationships between the studied hyper-parameters and the model performance are shown in the bottom figures. The "Save" button is designed by clicking which, the optimal model, file and figures can be saved. Note that values shown in the panel can also be changed manually for advanced users or teaching/learning purpose.

Application of the trained ML-based model

Trigger of the "Main menu" button can return to the main interface, and then the users can enter the "Predict" submodule for applying the trained ML-based model. A module under the "Predict" is termed "Parametric analysis" to examine the monotonicity of the model (see Fig. 3b). Another module under the "Predict" is termed the "Prediction on new data" (see Fig. 3c).

Parametric analysis

This module aims to investigate the relationships between the input variable and soil properties in the ML-based model. The first step is to load the original, new data file and the model. It should be noted that loading the original data file is to unify the upper and lower bounds with the original database, because the ML-based model is trained based on normalized value, which is a relative value. It is necessary the ensure the same criterion when implementing the normalization operation on the new data. As designed, after clicking the "Run" and "Save" button, the prediction results on the new datasets can be generated and saved.

Regarding the "Parametric analysis", the preparation of new data files obeys the following three constraints required by the tool: (1) only the value of the studied variable varies at each set while the values of the remaining variables maintain unchanged; (2) for each studied variable, the number of data at each set is the same and the number of sets for each variable is also identical; and (3) the data for all variables are required to be prepared equally and the order of preparation is identical to the order of input variables.

It should be noted that the parametric analysis for the ML-based model is affected by the selection of input data. To obtain an objective result, herein a method is recommended. The

distribution of each variable in the original dataset is known. Therefore, the values of the studied variable at the 20%, 30%, 40%, 50%, 60%, 70% and 80% percentile are selected, while the values of the remaining variables can be represented by their mean values (Num. of data = 7, Num. of sets = 1, see Fig. 3b), thereafter a set of data for investigating such studied variable is generated. The reason is that the relationship between the studied variable and the output variable may not be truly represented if the value of the input variable exceeds the normal range (less than the value at 20% and larger than the value at 80% percentile). Herein, the values of the remaining variables can also be represented by several representative values (Num. of sets > 1), thereafter obtaining several different relationships between the studied variable and output variables, and the final result of the parametric analysis can be obtained by mean trend. A noteworthy concern is the constraint relations among variables should be satisfied.

Prediction on the new data

In this module, users can load the trained ML-based model and apply it to new data. It aims to examine the exploitation and extrapolation ability of the trained ML-based model, because it still has a probability to meet unknown data that exceed the range of variables in the training set no matter how big the training set is. The results directly determine whether the trained model is sufficiently robust and can be further applied in engineering practice. The detailed implementation process can refer to the "Parametric analysis" module.

Comparative study

The maximum dry density with a relatively high number of input variables is taken as an example to examine the developed tool and compare the performance of different typical ML

algorithms.

Data source

The datasets used in this study are from Wang and Yin [59], who collected a total of 226 datasets, including gravel, sand and fines, from open literature. The six input variables were gravel content C_G, sand content C_S, fines content C_F, liquid limit LL, plastic limit PL and compaction energy E, in addition to one output variable, maximum dry density $\rho_{d,\max}$. Much research has revealed the strong relationships between these six input variables on $\rho_{d,\max}$, so it is reasonable to use ML algorithms to develop prediction models of $\rho_{d,\max}$ based on these six variables. For this research, 80% of the data were randomly selected to train six ML-based models, with the remaining 20% used to test models.

Development of ML-based models

All results presented in this section are obtained following the "Develop" section.

Genetic programming

Fig. 4a shows the initial settings of GP parameters for use in determining optimal GP configuration. The value of p_c is assigned in the range 0.1–0.9 with an interval of 0.1, and the value of p_m is assigned in the range 0.05–0.2, with an interval of 0.05. The function set includes 14 commonly used arithmetic operations, with 6 operations used: {+, –, ×, /, $\sqrt{\ }$, log}. The population is set to 50 and the number of iterations to 1000, large enough to guarantee the convergence of loss value and the search for an optimal result. Mean square error (MSE) is set as the loss function, which can be formulated using

$$L = \frac{1}{m}\sum_{i=1}^{m}\left(y_i^a - y_i^p\right)^2 \tag{14}$$

Figs. 4b and 4c show the R^2 values for the training and testing sets, respectively, which are generated by GP using different combinations of p_c and p_m. There is no deterministic relationship between the value of p_c or p_m and the model performance. Models featuring p_c = 0.1 or p_m = 0.15 performed best with an R^2 of 0.76 and 0.71 on the training and testing sets, respectively; the scatter plot is presented in Fig. 4d, compared with the actual results. The prediction performance of the GP-based model is poor when $\rho_{d,\max}$ is less than 1.5 or larger than 2.25, for which data are sparse. GP fails to capture the internal correlations in this area, owing to insufficient data, so that a large error is created.

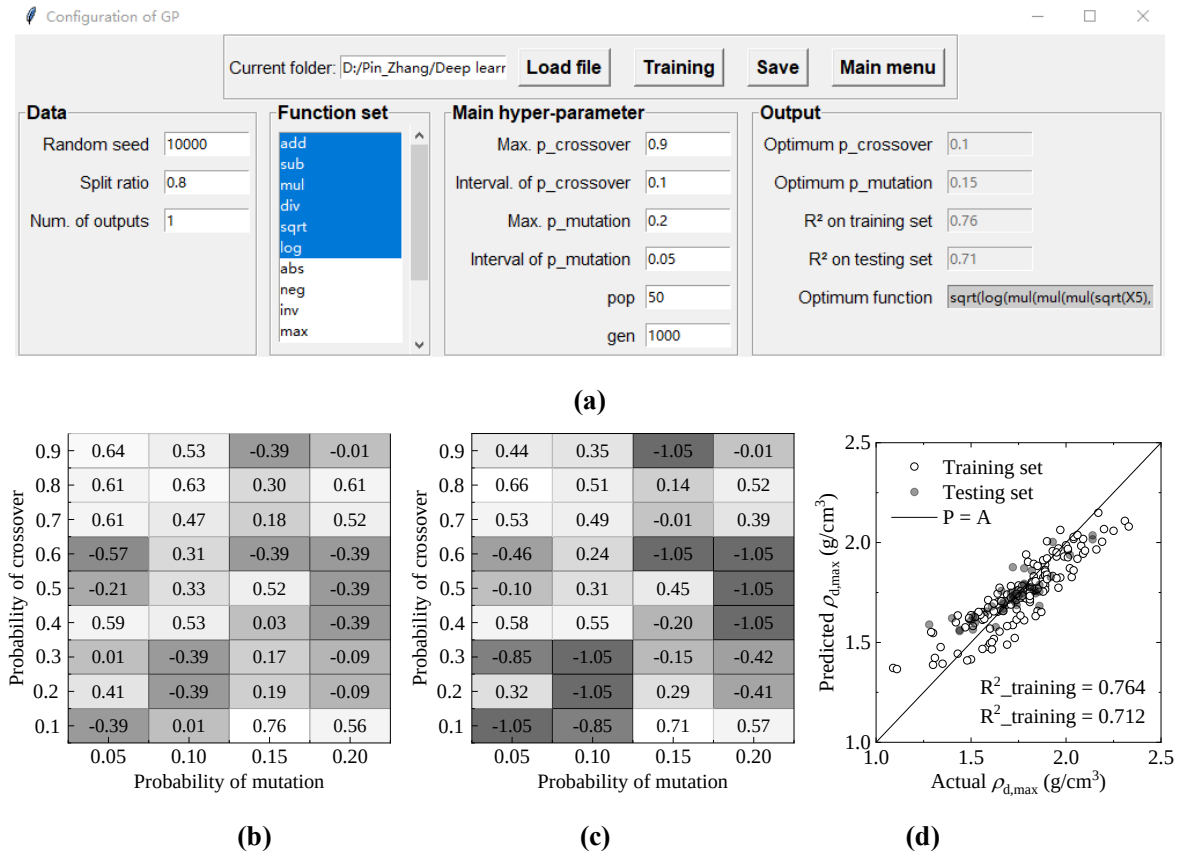
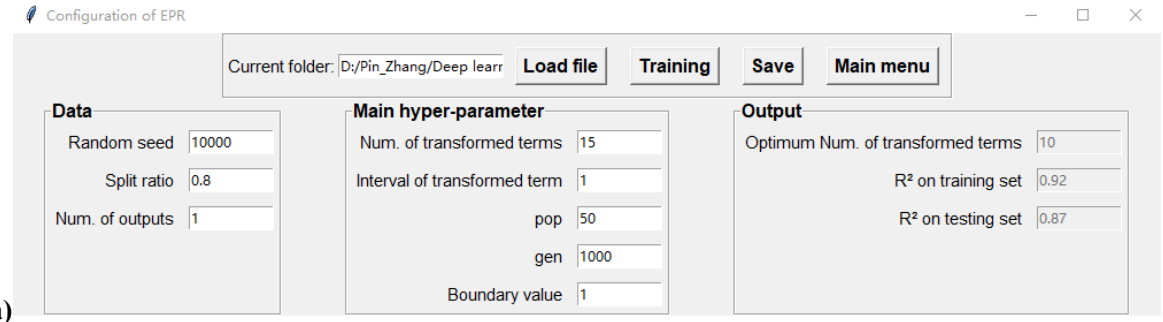


Fig. 4 Performance of GP-based model with various configurations: (a) initial settings of GP (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Evolutionary polynomial regression

The initial settings of EPR are shown in Fig. 5a. The number of transformed terms in EPR varies from 1 to 15 with an interval of 1, and the corresponding R^2 values are presented in Fig. 5b. To avoid complex formulations and overflow errors, the value of elements in the exponent matrix is limited to three discrete values [–1, 0, 1]. Besides, PSO is used to search for the optimal discrete values of elements in the exponent matrix, with MSE used as the loss function. Population size and the number of iterations are the same as the values assigned in the GP.

Fig. 5b presents R^2 values generated by the EPR-based model for various numbers of transformed terms. As the number of transformed terms increases, the accuracy of the training set improves. The accuracy is roughly saturated on the training set as the number of transformed terms reaches 9, but the model's performance on the testing set still fluctuates. The optimal number of transformed terms is 10, which represents the ideal trade-off between accuracy and simplicity for this model. The corresponding R^2 values for the training and testing sets are 0.92 and 0.87, respectively. The $\rho_{d,\max}$ predicted using the optimal EPR-based model shows excellent agreement with the actual $\rho_{d,\max}$ and clearly outperforms the GP-based model, particularly in predicting small values of $\rho_{d,\max}$ (Fig. 5c). For several data with large values, the EPR-based model still produces a discernible error.



(a)

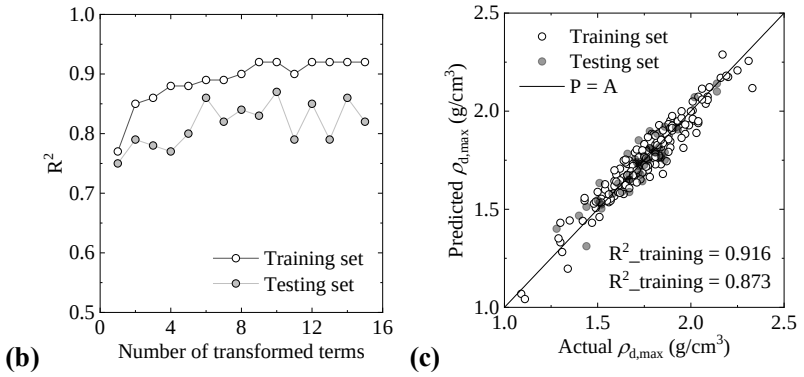


Fig. 5 Performance of EPR-based model with various configurations: (a) initial settings of EPR;(b) on the training and testing sets; (c) predicted results using the optimal configuration

Support vector regression

Fig. 6a presents the initial SVR settings. The loss function is MSE, and the values of C and γ increase by a constant factor of 2. Figs. 6b and 6c present variations in model performance with various combinations of C and γ for the training and testing sets, respectively, clearly showing that the accuracy of the SVR-based model increases with C, because larger values of C impose a greater penalty for incorrect prediction; excessively large values of C may also create overfitting issues. Increasing values of γ also increase the model's accuracy until it reaches the peak value, after which they can reduce its accuracy, indicating that an optimal value of γ in the kernel function can maximise SVR performance. The optimal combination of C and γ is 128 and 0.25, respectively, with corresponding R^2 values for the training and testing sets of 0.95 and 0.89, respectively. All predicted data points are close to the line with a slope of 1, as Fig. 6d shows. The SVR-based model greatly improves prediction performance in comparison with explicit formulation-based models such as GP- and EPR-based models.

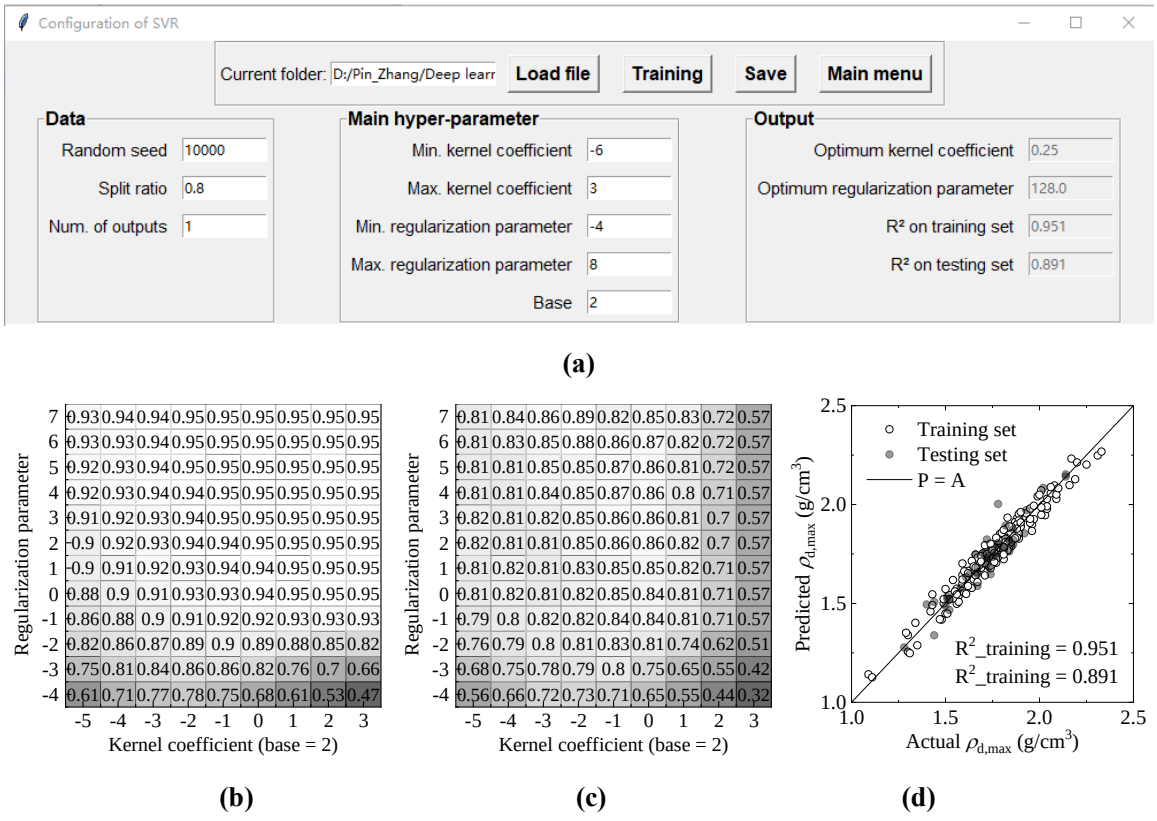


Fig. 6 Performance of SVR-based model with various configurations: (a) initial settings of SVR; (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Random forest

The initial settings of RF are presented in Fig. 7a. MSE is set as the loss function. The *ntree* increases from 10 to 80 with an interval of 10, and *mtry* increases from 1 to 6 as the dimension of input data. The performance of the RF-based model with various combinations of *ntree* and *mtry* on the training and testing sets is presented in Figs. 7b and 7c. It can be seen that the RF-based model shows excellent fitting ability in capturing the relationships between the input and output variables. The model developed based on random combinations of *ntree* and *mtry* can achieve excellent performance, which indicates the RF-based model is less sensitive to variations in hyper-parameters. The optimal *ntree* and *mtry* are 40 and 3, respectively, and the corresponding R^2 values for the training and testing sets are 0.99 and 0.91, respectively. Fig.

7d presents the scatter plot of the predicted and measured $\rho_{d,\max}$. The data points are close to the line with a slope of 1, particularly for the training set, for which the predicted results are perfectly consistent with the actual results.

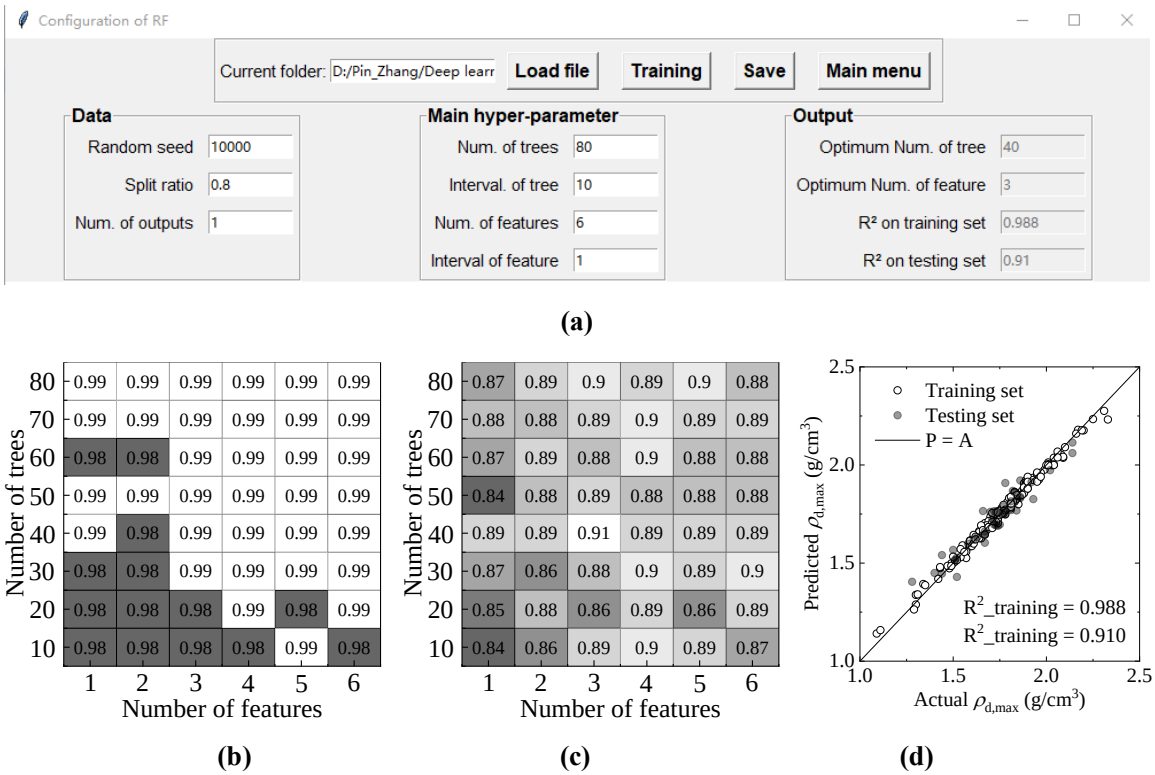


Fig. 7 Performance of RF-based model with various configurations: (a) initial settings of RF; (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Feedforward neural network

The performance of the FNN-based model depends on numerous configurations. Herein, the effects of N_l and N_n on the performance of the FNN-based model are investigated, in which the number of hidden layers increases from 1 to 4 and the number of hidden neurons from 10 to 80. The remaining configurations are assigned by commonly used methods, as Fig. 8a shows. The activation function in the hidden layer is ReLU. Because the prediction of $\rho_{d,\max}$ is a regression issue, activation in the output layer is *linear* and the loss function is MSE. Because the performance of FNN relies heavily on initial weights and biases, the mean performance of

a FNN-based model with 10 different initial weights and biases is used to represent the real performance of this model. The batch size is set as half of the training datasets.

The increasing complexity of the neural network can improve the fitting ability of the FNN, thereby showing higher accuracy for the training set. Simultaneously, it may induce a more severe overfitting issue, further reducing the accuracy for the testing set (see Figs. 8b and 8c). Finally, the model with $N_l = 1$ and $N_n = 80$ shows optimal performance, with R^2 values for the training and testing sets of 0.96 and 0.86, respectively. Fig. 8d presents the scatter plot; it can be seen that the predicted $\rho_{d,\max}$ shows excellent agreement with the actual $\rho_{d,\max}$.

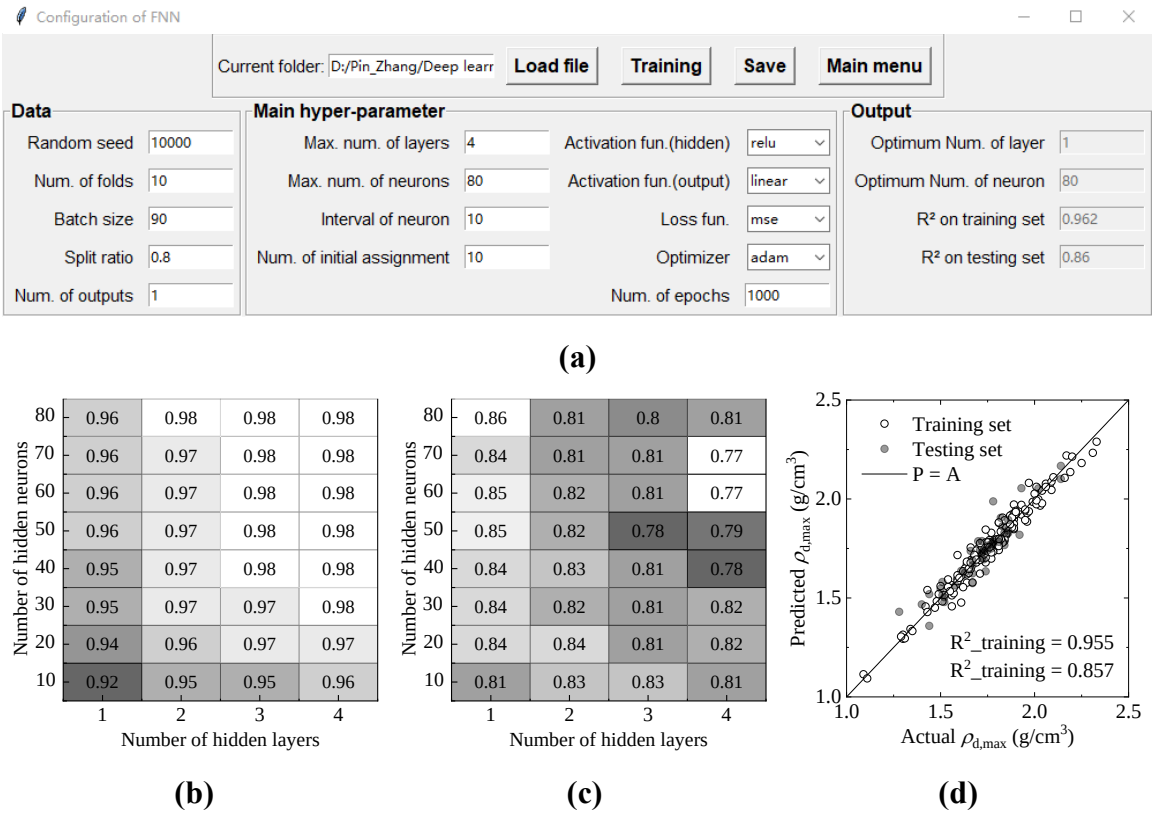


Fig. 8 Performance of FNN-based model with various configurations: (a) initial settings of FNN; (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

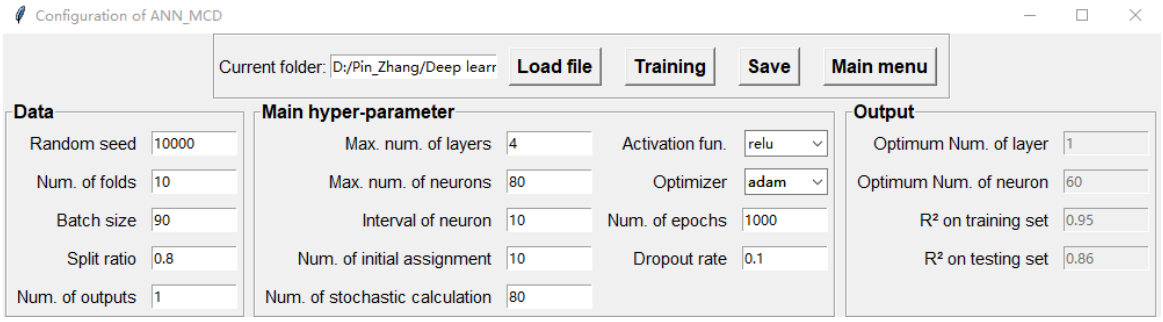
Artificial neural network with Monte Carlo dropout

The implementation of ANN_MCD is similar to that of ANN. The configurations are identical to the FNN irrespective of the loss function and the activation function used in the output layer,

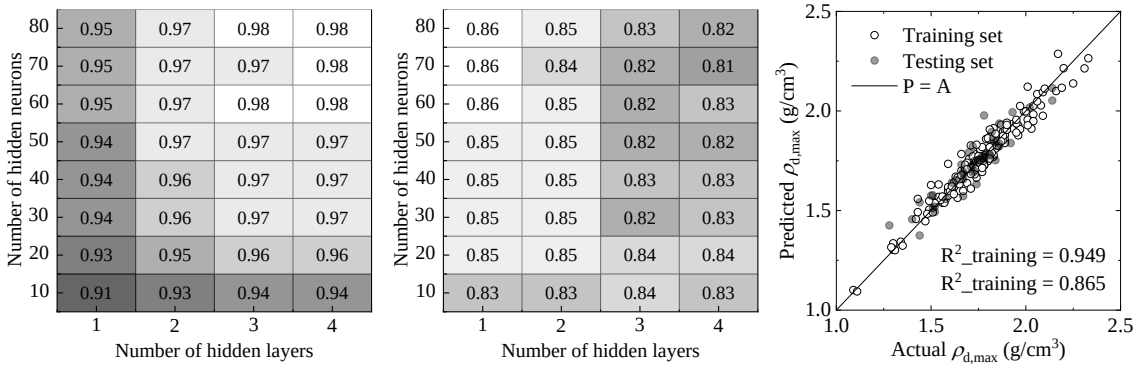
which is assigned as the predefined default. Fig. 9a presents the initial settings of ANN_MCD. Negative log-likelihood (NLL) is defined as the loss function for the uncertainty issue, intended to maximize the likelihood of approximating the actual result. Combined with *k*-fold cross-validation, the loss function can be formulated using

$$L = -\frac{1}{k}\sum_i y_i^a \log\left(y_i^p\right) \tag{15}$$

It should be noted that the dropout probability activates in the testing phase. Figs. 9b and 9c show the performance of the ANN_MCD-based model with various combinations of N_l and N_n for the training and testing sets, respectively. Much as for the FNN-based model, the increasing complexity of the neural network improves accuracy for the training set, but an overfitting issue also arises after N_l exceeds 1. Finally, the model with $N_l = 1$ and $N_n = 60$ shows optimal performance, with R^2 values for the training and testing sets of 0.95 and 0.86, respectively. The accuracy for the training set is less than for the results generated using the ANN-based model; the scatter plot is shown in Fig. 9d, which indicates that the uncertainty of the ANN_MCD sacrifices a certain degree of accuracy.



(a)



(b) **(c)** **(d)**

Fig. 9 Performance of ANN_MCD-based model with various configurations: (a) initial settings of ANN_MCD; (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Based on the foregoing analysis results, the ML algorithms exhibit excellent potential for modelling correlations that involve soil properties. Table 3 summarizes the performance of the six ML algorithms for training and testing sets alike.

Table 3 Comparison of ML-based models

Algorithm	Training set	Testing set	Parametric analysis	Application on new data	Overall R_{MS}
GP	0.764 (6)	0.712 (6)	(1)	0.444 (6)	4.5 (6)
EPR	0.916 (5)	0.873 (3)	(1)	0.795 (5)	3.4 (5)
SVR	0.951 (3)	0.891 (2)	(3)	0.801 (4)	3.2 (4)
RF	0.988 (1)	0.910 (1)	(4)	0.817 (3)	2.7 (3)
FNN	0.962 (2)	0.860 (5)	(2)	0.819 (2)	2.6 (2)
ANN_MCD	0.949 (4)	0.865 (4)	(2)	0.837 (1)	2.2 (1)

Note: the number in the bracket indicates the rank of model performance, and 1 is optimal

Parametric analysis

To enhance the interpretability of the ML-based model, an investigation of what has been learnt by the ML-based model is needed. To this end, the relationships between the input and output variables in each ML-based model are revealed. Preparation of data obeys the constraints

already mentioned. For each studied input variable, after obtaining its distribution in the original database, 7 values at the 20–80% integer percentile are extracted and form a set of data, with the values of the remaining input variables maintaining the individual mean value.

The results of the parametric analysis are presented in Fig. 10, conducted using the "Parametric analysis" module. The values of input as well as output variables are normalized by the individual maximum value, with their relationships roughly identical in six ML-based models. The relationships generated by the RF-based model are not as smooth as other ML algorithm-based models, because the output of RF is discrete owing to its binary tree structure [19,39]. A slight difference is observed in the relationships of C_S versus $\rho_{d,\max}$ and E versus $\rho_{d,\max}$ in SVR when the input variable value exceeds the range of most data in the database or the output of the model is not sensitive to the input variable. The relationships in the GP- and EPR-based models can be expressed by explicit formulations, thus showing the smoothest curves. The FNN- and ANN_MCD-based models also perform well at capturing the relationships between the input and output variables, showing a trend similar to that generated by the GP- and EPR-based model but exhibiting slight variation for C_F versus $\rho_{d,\max}$ (FNN) and LL versus $\rho_{d,\max}$ (ANN_MCD).

Overall, $\rho_{d,\max}$ has a positive relationship with C_G, C_S and E and is negative to C_F, LL and PL, consistent with the experimental observations. Increases in LL and PL generally mean increases in clay content, and increases in C_F induces the decrease in $\rho_{d,\max}$ [60]. Increases in E within a certain range can improve compactness and produce a larger $\rho_{d,\max}$ [61]. These results explain what has been learned using the ML-based models, indicating their

reasonableness.

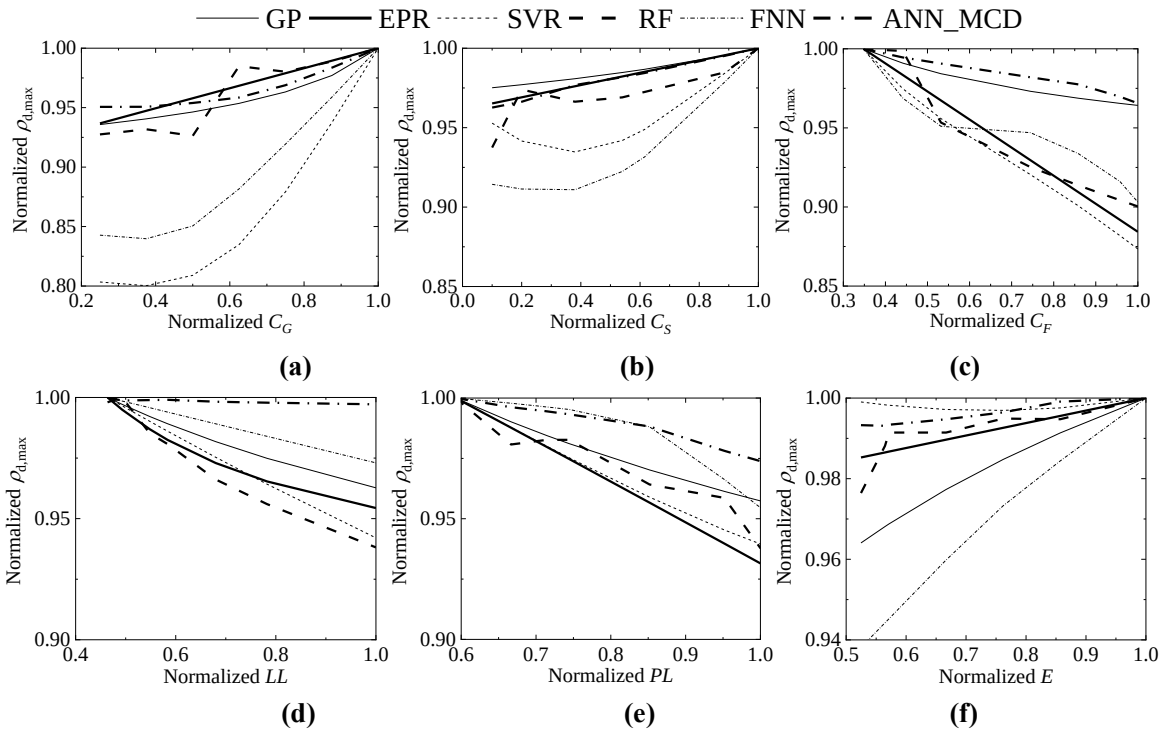


Fig. 10 Relationships generated by six ML-based models: (a) C_G–$\rho_{d,max}$; (b) C_S–$\rho_{d,max}$; (c) C_F–$\rho_{d,max}$; (d) LL–$\rho_{d,max}$; (e) PL–$\rho_{d,max}$; (f) E–$\rho_{d,max}$

Prediction on new data

After the six ML-based models are well trained, they can be directly saved and applied to predict unknown data that are not exposed in the original database. This process can be implemented using the "Prediction on new data" module, with the intent of examining the exploitation and exploration abilities of ML-based models. An additional 19 data were collected from the literature [60,62]. Fig. 11 compares the results of given input variables to the six ML-based models with the actual results. Table 3 summarizes the values of R^2 for the new data. The ranking of each model's performance differs slightly from the results for the original training and testing sets, with the ANN_MCD-based model outperforming the RF-based model and showing the best performance. ANN_MCD is an uncertain algorithm, in

which the predicted result is assigned through reliability evaluation. In other words, ANN_MCD knows what it does not know, allowing it to be reasonably deduced that ANN_MCD is more robust for unknown data. The FNN-based model also shows stable prediction performance for unknown data. The accuracy of the SVR- and RF-based models show a clear decrease, indicating poor robustness, perhaps related to its failure to capture the relationships between some input parameters and the studied property. The GP- and EPR-based models still expose low prediction accuracy, relating to their limited mapping capability. Overall, the prediction accuracy of RF-, FNN- and ANN_MCD-based models for unknown data can be acceptable, offering a promising scenario for application.

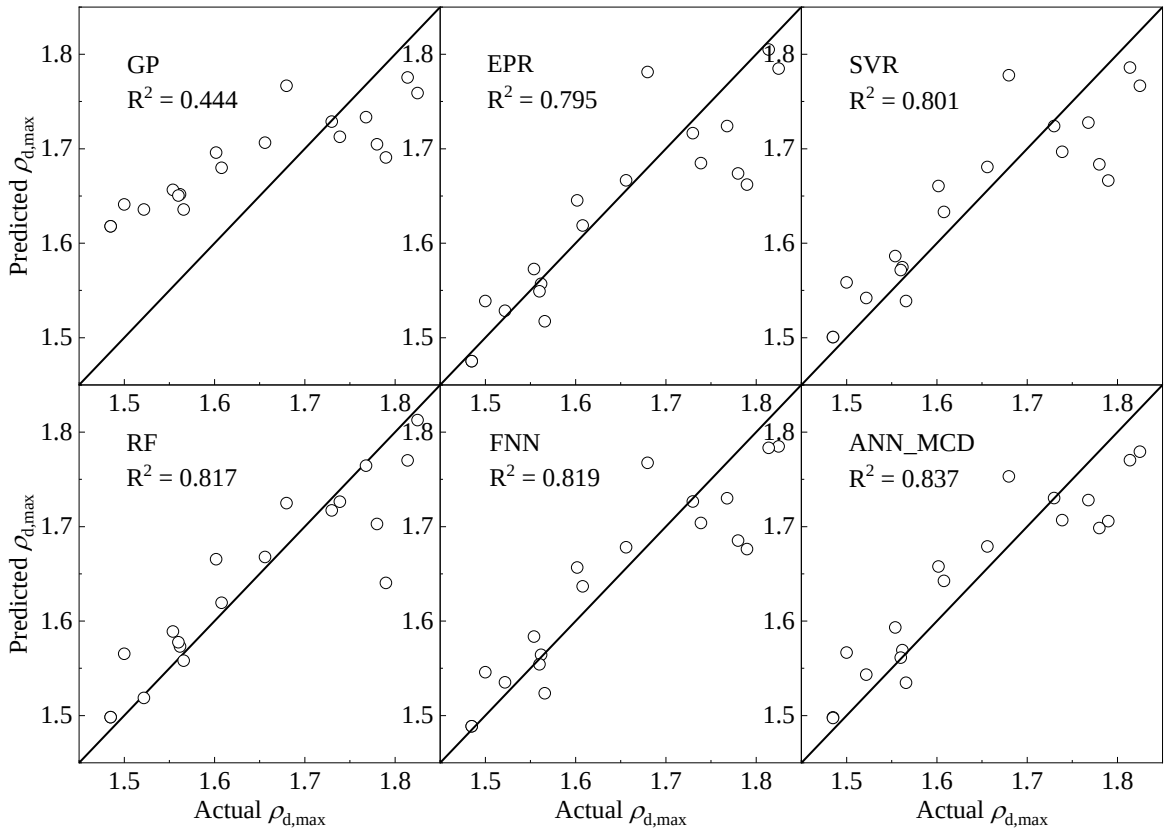


Fig. 11 Prediction on the unknown data

Model selection

Model selection has always garnered much attention in various domains, from different viewpoints. To the authors' best knowledge, researchers have tended to approach its implementation from three aspects [63]: selection of (1) input variables, (2) algorithms or approaches, and (3) hyper-parameters of the applied algorithm or approach. Current methods and theories for model selection focus mainly on the first category, in which models with combinations of different input variables are developed and their performance compared to selecting the optimal one [39]. In the second category, benchmark testing has generally been conducted to evaluate the performance of the selected algorithms or approaches [19]. In the third category, trial and error, grid search and hybrid algorithms have been the most commonly used methods [64].

In this study, the same input variables are used for all ML-based models, with their hyperparameters able to be directly determined by the search method built into the GUI platform. To this end, the model selection focuses on algorithm selection in this section. Researches have generally used error indicators to measure the performance of different ML-based models for the training and testing sets, a limited approach that cannot sufficiently evaluate the performance of a ML-based model. As already noted, this study has revealed several aspects of model performance, indicating that no model always outperforms all other models in all aspects. To overcome this issue, Zorlu et al. [65] developed a ranking method of comparing model performance, in which model performance for the training and testing sets are ranked by the values of error indicators. By summing the ranks for the training and tests, a

final ranking of model performance can be produced and the optimal model identified. However, this ranking method is limited, because it merely accounts for model performance for the training and testing sets. Meanwhile, model performance for the training and testing sets was assigned the same weights, although performance on the testing set is more important.

Motivated by Zorlu et al. [65], a novel ranking index (R_{MS}; see Eq. (16)) is proposed with which to comprehensively and objectively evaluate the performance of various ML-based models from five aspects, i.e., accuracy for the training set, accuracy for the testing set, monotonicity, exploitation and exploration ability for new data, and model complexity that describes the model topology and the number of parameters. Meanwhile, the weights in the different aspects vary. This ranking index can be mathematically expressed by

$$\begin{aligned} &R_{MS} = p_{TR} \times R_{TR} + p_{TE} \times R_{TE} + p_{PA} \times R_{PA} + p_{EE} \times R_{EE} + p_{MC} \times R_{MC} \\ &\text{with } p_{TR} + p_{TE} + p_{PA} + p_{EE} + p_{MC} = 1 \end{aligned} \tag{16}$$

where p_{TR}, p_{TE}, p_{PA}, p_{EE} and p_{MC} are the weights of the aforementioned five aspects on the model selection, respectively, and R_{TR}, R_{TE}, R_{PA}, R_{EE} and R_{MC} are the rank of the performance of the studied model among all developed ML-based models in the aforementioned five aspects, respectively. Smaller R_{MS} values indicate better performance, providing a direct method of model selection. Note that this is only a simple form for ranking index, more forms are also worth trying.

It should be noted that model complexity is a significant factor for the comparison of model performance. This study completes all implementations in the GUI, and because each ML-based model can be easily used by loading it, model complexity need not be considered. Based on the introduction in the former section, the most important factors for evaluating a

ML-based model are its generalization ability on the unknown data and whether it can learn the correct relationships between the input variables and the studied soil property. Two such factors directly determine the applicability and the internal mechanism of a ML-based model. The model's performance during the developing phase is less important, but performance on the testing set should draw more attention to the training set. Accordingly, p_{TR}, p_{TE}, p_{PA}, p_{EE} and p_{MC} are ultimately assigned values of 0.1, 0.2, 0.3, 0.4 and 0, respectively. Table 3 provides detailed results for R_{TR}, R_{TE}, R_{PA} and R_{EE}, with the final rank presented in its last column. ANN_MCD shows the best performance for modelling correlations of $\rho_{d,\max}$, followed by models of FNN, RF, SVR, EPR and GP. Noted that this is only a simple example, values of weights can be changed according to the problem given.

Conclusions

This study first comprehensively reviewed the application of ML algorithms in modelling soil properties for geotechnical design. The algorithms were categorized into several groups based on their principles, and the main characteristics of these ML algorithms were summarized. Six representative ML algorithms were further detailed, including genetic programming (GP), evolutionary polynomial regression (EPR), support vector regression (SVR), random forest (RF), feedforward neural network (FNN) and artificial neural network integrated with Monte Carlo dropout (ANN_MCD).

These six algorithms were selected to develop a machine learning (ML)-based graphical user interface (GUI) platform. This GUI provides users with an easy-to-use tool for building a ML-based model, in which the entire process can be completed, including determination of

optimal configurations for ML algorithms, evaluation of model accuracy, investigation of relationships between the input variables and soil properties, and application of the developed model to new data. The goal of developing this platform was to facilitate the application of ML algorithms to geotechnical design, but it is versatile and can be used in any domain.

In order to compare the performance of different ML algorithms, the soil maximum dry density $\rho_{d,\max}$ was selected as an example. Their performance on five aspects, accuracy for the training set, accuracy for the testing set, exploitation and exploration of the new data, parametric analysis and model complexity, was revealed. The RF-based model showed the highest accuracy for the training and testing sets, explicit formulation-based models (GP- and EPR-based) can capture the smoothest relationships between the input variables and the studied soil properties, and ANN-based models (FNN- and ANN_MCD-based) showed the greatest robustness for unknown data.

Considering that the ML-based model exhibited varying performance in different aspects, a novel ranking index was proposed with which to comprehensively evaluate the performance of various ML-based models on five aforementioned aspects, further facilitating model selection. ANN_MCD presented the optimal generalization ability for modelling correlations of $\rho_{d,\max}$, followed by FNN, RF, SVR, EPR and GP.

Data Availability Statement

(GUI download available: https://www.researchgate.net/publication/348617390_ErosMLM)
All data used during the study are available from the corresponding author by request.

Acknowledgements

This research was financially supported by the Research Grants Council (RGC) of Hong Kong Special Administrative Region Government (HKSARG) of China (Grant No.: R5037-18F).

Credit author statement

Pin Zhang: conceptualization, methodology, analysis, writing-review and original draft.

Zhen-Yu Yin: supervision, methodology, visualization, writing-review and editing.

Yin-Fu Jin: validation, visualization and editing.

Conflicts of Interest

The authors declare that the work described has not been published before; that it is not under consideration for publication anywhere else; that its publication has been approved by all co-authors; that there is no conflict of interest regarding the publication of this article.

References

1. Yin J.H. (1999) Properties and behaviour of Hong Kong marine deposits with different clay contents. Can Geotech J, 36(6):1085-1095
2. Nagaraj T.S., Murthy B.R.S. (1986) A critical reappraisal of compression index equations. Géotechnique, 36(1):27-32
3. Ouyang Z., Mayne P.W. (2019) Modified NTH method for assessing effective friction angle of normally consolidated and overconsolidated clays from piezocone tests. J Geotech Geoenviron Eng, 145(10):04019067
4. Hattab M., Hammad T., Fleureau J.M. (2015) Internal friction angle variation in a kaolin/montmorillonite clay mix and microstructural identification. Géotechnique, 65(1):1-11
5. Yoon G.L., Kim B.T., Jeon S.S. (2004) Empirical correlations of compression index for marine clay from regression analysis. Can Geotech J, 41(6):1213-1221
6. Kootahi K. (2017) Simple index tests for assessing the recompression index of fine-grained soils. J Geotech Geoenviron Eng, 143(4):06016027
7. Hayden C.P., Purchase-Sanborn K., Dewoolkar M. (2018) Comparison of site-specific and empirical correlations for drained residual shear strength. Géotechnique, 68(12):1099-1108
8. Zhang P., Jin Y.-F., Yin Z.-Y., Yang Y. (2020) Random forest based artificial intelligent model for predicting failure envelopes of caisson foundations in sand. Appl Ocean Res, 101:102223
9. Hochreiter S., Schmidhuber J. (1997) Long Short-Term Memory. Neural Comput., 9(8):1735–1780

10. Krizhevsky A., Sutskever I., Hinton G., ImageNet classification with deep convolutional neural networks, NIPS, 2012.
11. LeCun Y., Bengio Y., Hinton G. (2015) Deep learning. Nature, 521:436-444
12. Shen S.-L., Atangana Njock P.G., Zhou A., Lyu H.-M. (2021) Dynamic prediction of jet grouted column diameter in soft soil using Bi-LSTM deep learning. Acta Geotech, 16:303–315
13. Atangana Njock P.G., Shen S.-L., Zhou A., Lyu H.-M. (2020) Evaluation of soil liquefaction using AI technology incorporating a coupled ENN / t-SNE model. Soil Dyn Earthq Eng, 130:105988
14. Lin S.-S., Shen S.-L., Zhang N., Zhou A. (2021) Comprehensive environmental impact evaluation for concrete mixing station (CMS) based on improved TOPSIS method. Sustainable Cities and Society, 69:102838
15. Kohestani V.R., Hassanlourad M. (2016) Modeling the mechanical behavior of carbonate sands using artificial neural networks and support vector machines. Int J Geomech, 16(1):04015038
16. Penumadu D., Zhao R.D. (1999) Triaxial compression behavior of sand and gravel using artificial neural networks (ANN). Comput Geotech, 24:207-230
17. Zhang N., Shen S.-L., Zhou A., Jin Y.-F. (2021) Application of LSTM approach for modelling stress–strain behaviour of soil. Appl Soft Comput, 100:106959
18. Zhang P., Wu H.N., Chen R.P., Chan T.H.T. (2020) Hybrid meta-heuristic and machine learning algorithms for tunneling-induced settlement prediction: A comparative study. Tunnell Undergr Space Technol, 99:103383
19. Zhang P., Yin Z.Y., Jin Y.F., Chan T., Gao F.P. (2021) Intelligent modelling of clay compressibility using hybrid meta-heuristic and machine learning algorithms. Geosci Front, 12(1):441-452
20. Qi C.C., Tang X.L. (2018) Slope stability prediction using integrated metaheuristic and machine learning approaches: A comparative study. Comput Ind Eng, 118:112-122
21. Feng Y., Cui N., Hao W., Gao L., Gong D. (2019) Estimation of soil temperature from meteorological data using different machine learning models. Geoderma, 338:67-77
22. Zhang P., Yin Z.Y., Jin Y.F. (2021) State-of-the-art review of machine learning applications in constitutive modeling of soils. Arch Comput Method Eng, 10.1007/s11831-020-09524-z
23. Elbaz K., Shen S.L., Zhou A.N., Yin Z.Y., Lyu H.M. (2021) Prediction of disc cutter life during shield tunnelling with AI via incorporation of genetic algorithm into GMDH-type neural network. Engineering, 7(2):238-251
24. Gal Y., Ghahramani Z. (2015) Dropout as a Bayesian approximation: representing model uncertainty in deep learning. arXiv:1506.02142
25. Blundell C., Cornebise J., Kavukcuoglu K., Wierstra D. (2015) Weight uncertainty in neural networks. arXiv:1505.05424v2
26. Graves A. (2011) Practical variational inference for neural networks. NIPS
27. Zhang P., Jin Y.F., Yin Z.Y. (2021) Machine learning–based uncertainty modelling ofmechanical properties of soft clays relating totime-dependent behavior and its application. Int J Numer Anal Methods Geomech, 10.1002/nag.3215

28. Tan F., Zhou W.-H., Yuen K.-V. (2018) Effect of loading duration on uncertainty in creep analysis of clay. Int J Numer Anal Methods Geomech, 42(11):1235-1254
29. Zhou W.H., Tan F., Yuen K.V. (2018) Model updating and uncertainty analysis for creep behavior of soft soil. Comput Geotech, 100:135-143
30. Tan F., Zhou W.-H., Yuen K.-V. (2016) Modeling the soil water retention properties of same-textured soils with different initial void ratios. J Hydrol, 542:731-743
31. Zhou W.-H., Yuen K.-V., Tan F. (2014) Estimation of soil–water characteristic curve and relative permeability for granular soils with different initial dry densities. Eng Geol, 179:1-9
32. Zhang W., Goh A.T.C. (2016) Multivariate adaptive regression splines and neural network models for prediction of pile drivability. Geosci Front, 7(1):45-52
33. Cheng Z.-L., Zhou W.-H., Garg A. (2020) Genetic programming model for estimating soil suction in shallow soil layers in the vicinity of a tree. Eng Geol, 268:105506
34. Yin Z.Y., Jin Y.F., Huang H.W., Shen S.L. (2016) Evolutionary polynomial regression based modelling of clay compressibility using an enhanced hybrid real-coded genetic algorithm. Eng Geol, 210:158–167
35. Wang R., Zhang K., Wang W., Meng Y., Yang L., Huang H. (2020) Hydrodynamic landslide displacement prediction using combined extreme learning machine and random search support vector regression model. Eur J Environ Civ Eng:1-13
36. Samui P., Sitharam T.G. (2008) Least‐square support vector machine applied to settlement of shallow foundations on cohesionless soils. Int J Numer Anal Met, 32(17):419-427
37. Ai L., Fang N.F., Zhang B., Shi Z.H. (2013) Broad area mapping of monthly soil erosion risk using fuzzy decision tree approach: integration of multi-source data within GIS. Int J Geogr Inf Sci, 27(6):1251-1267
38. Qi C., Fourie A., Zhao X. (2018) Back-analysis method for stope displacements using gradient-boosted regression tree and firefly algorithm. J Comput Civil Eng, 32(5):04018031
39. Zhang P., Yin Z.Y., Jin Y.F., Chan T.H.T. (2020) A novel hybrid surrogate intelligent model for creep index prediction based on particle swarm optimization and random forest. Eng Geol, 265:105328
40. Sanikhani H., Deo R.C., Yaseen Z.M., Eray O., Kisi O. (2018) Non-tuned data intelligent model for soil temperature estimation: A new approach. Geoderma, 330:52-64
41. Yamaç S.S., Şeker C., Negiş H. (2020) Evaluation of machine learning methods to predict soil moisture constants with different combinations of soil input data for calcareous soils in a semi arid area. Agric Water Manage, 234:106121
42. Chen R.P., Zhang P., Kang X., Zhong Z.Q., Liu Y., Wu H.N. (2019) Prediction of maximum surface settlement caused by EPB shield tunneling with ANN methods. Soils Found, 59(2):284–295
43. Yilmaz I., Marschalko M., Bednarik M., Kaynar O., Fojtova L. (2012) Neural computing models for prediction of permeability coefficient of coarse-grained soils. Neural Comput Appl, 21(5):957-968
44. Kiefa M.A.A. (1998) General regression neural networks for driven piles in cohesionless soils. J Geotech Geoenviron Eng, 124(12):1177-1185

45. Feng X., Jimenez R. (2015) Predicting tunnel squeezing with incomplete data using Bayesian networks. Eng Geol, 195:214-224
46. Goh A.T.C., Kulhawy F.H., Chua C.G. (2005) Bayesian neural network analysis of undrained side resistance of drilled shafts. J Geotech Geoenviron Eng, 131(1):84-93
47. Koza J.R., Genetic programming: on the programming of computers by natural selection, MIT Press, Cambridge, MA., 1992.
48. Sette S., Boullart L. (2001) Genetic programming: principles and applications. Eng Appl Artif Intel, 14:727-736
49. Giustolisi O., Savic D.A. (2006) A symbolic data-driven technique based on evolutionary polynomial regression. J Hydroinform, 8(4):235-237
50. Jin Y.F., Yin Z.Y. (2020) Enhancement of backtracking search algorithm for identifying soil parameters. Int J Numer Anal Methods Geomech, 44(9):1239-1261
51. Jin Y.F., Yin Z.Y. (2020) An intelligent multi-objective EPR technique with multi-step model selection for correlations of soil properties. Acta Geotech, 15(8):2053-2073
52. Yin Z.Y., Jin Y.F., Shen J.S., Hicher P.Y. (2018) Optimization techniques for identifying soil parameters in geotechnical engineering: Comparative study and enhancement. Int J Numer Anal Methods Geomech, 42(1):70-94
53. Cortes C., Vapnik V. (1995) Support-Vector networks. Mach Learn, 20:273-297
54. Breiman L. (2001) Random Forests. Mach Learn, 45:5–32
55. Breiman L. (1996) Bagging Predictors. Mach Learn, 24(2):123-140
56. Zhang P., Chen R.P., Wu H.N. (2019) Real-time analysis and regulation of EPB shield steering using Random Forest. Automat Constr, 106:102860
57. Rumelhart D.E., Hinton G.E., Williams R.J. (1986) Learning representations by back-propagating errors. Nature, 323(9):533–536
58. Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R. (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res, 15:1929–1958
59. Wang H.L., Yin Z.Y. (2020) High performance prediction of soil compaction parameters using multi expression programming. Eng Geol, 276:105758
60. Ören A.H. (2014) Estimating compaction parameters of clayey soils from sediment volume test. Appl Clay Sci, 101:68-72
61. Gurtug Y., Sridharan A. (2004) Compaction behaviour and prediction of its characteristics of fine grained soils with particular reference to compaction energy. Soils Found, 44(5):27-36
62. AI-Khafaji A.N. (1993) Estimation of soil compaction parameters by means of Atterberg limits. Q J Eng Geol, 26:359-368
63. Luo G. (2016) A review of automatic selection methods for machine learning algorithms and hyper-parameter values. Netw Model Anal Health Inform Bioinforma, 5:18
64. Zhang P., Yin Z.Y., Jin Y.F., Ye G.L. (2020) An AI-based model for describing cyclic characteristics of granular materials. Int J Numer Anal Methods Geomech, 44(9):1315-1335
65. Zorlu K., Gokceoglu C., Ocakoglu F., Nefeslioglu H.A., Acikalin S. (2008) Prediction of uniaxial compressive strength of sandstones using petrography-based models. Eng Geol, 96(3-4):141-158

Figure Caption

Fig. 1 Number of publications relating to development and application of ML algorithms to predicting soil properties

Fig. 2 Schematic view of internal mechanism of adopted six ML algorithms: (a) GP; (b) EPR; (c) SVR; (d) RF; (e) FNN; (f) ANN_MCD

Fig. 3 Introduction of ErosMLM: (a) main interface; (b) "Parametric analysis" module; (c) "Prediction on new data" module

Fig. 4 Performance of GP-based model with various configurations: (a) initial settings of GP (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Fig. 5 Performance of EPR-based model with various configurations: (a) initial settings of EPR;(b) on the training and testing sets; (c) predicted results using the optimal configuration

Fig. 6 Performance of SVR-based model with various configurations: (a) initial settings of SVR; (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Fig. 7 Performance of RF-based model with various configurations: (a) initial settings of RF; (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Fig. 8 Performance of FNN-based model with various configurations: (a) initial settings of FNN; (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Fig. 9 Performance of ANN_MCD-based model with various configurations: (a) initial settings of ANN_MCD; (b) on the training set; (c) on the testing set; (d) predicted results using the optimal configuration

Fig. 10 Relationships generated by six ML-based models: (a) C_G–$\rho_{d,\max}$; (b) C_S–$\rho_{d,\max}$; (c) C_F–$\rho_{d,\max}$; (d) LL–$\rho_{d,\max}$; (e) PL–$\rho_{d,\max}$; (f) E–$\rho_{d,\max}$

Fig. 11 Prediction on the unknown data