TẠP CHÍ KHOA HỌC & CÔNG NGHỆ ĐẠI HỌC DUY TÂN
DTU Journal of Science and Technology 3(58) (2023) 3-11
TRƯỜNG ĐẠI HỌC DUY TÂN
DUYTAN UNIVERSITY

XGBoost regression for estimating bearing capacity of concrete piles

Sử dụng hồi quy XGBoost để đánh giá sức chịu tải của cọc bê tông

Tran Thu Hien[a*], Hoang Nhat Duc[a,b]
Trần Thu Hiền[a*], Hoàng Nhật Đức[a,b]

[a]*Faculty of Civil Engineering, Duy Tan University, 55000, Danang, Vietnam*
[a]*Khoa Xây dựng, Trường Đại học Duy Tân, Đà Nẵng, Việt Nam*
[b] *Institute of Research and Development, Duy Tan University, Da Nang, 55000, Vietnam*
[b] *Viện Nghiên cứu và phát triển Công nghệ Cao, Trường Đại học Duy Tân, Đà Nẵng, Việt Nam*

(Ngày nhận bài: 04/3/2023, ngày phản biện xong: 13/3/2023, ngày chấp nhận đăng: 10/5/2023)

Abstract

This paper uses XGBoost to predict bearing capacity of concrete piles. The proposed model is trained and tested against a dataset of 472 samples collected from static load tests in Vietnam. The results indicate that the default XGBoost model consistently outperforms the Deep Neural Network (DNN) regression. XGBoost is a suitable tool for engineers to predict pile bearing capacity.

Keywords: Pile bearing capacity; Machine learning; XGBoost.

Tóm tắt

Trong bài báo này, XGBoost được sử dụng để dự đoán sức chịu tải của cọc bê tông. Mô hình đề xuất được huấn luyện và kiểm tra với 472 dữ liệu, thu thập từ các thí nghiệm nén tĩnh tại Việt Nam. Kết quả thu được chỉ ra rằng, phương pháp hồi quy XGBoost cho kết quả chính xác hơn so với phương pháp DNN. XGBoost có thể được sử dụng để dự đoán sức chịu tải của cọc bê tông.

Từ khóa: Sức chịu tải của cọc; Máy học; XGBoost;

1. Research background and motivation

Deep foundation is a common and obligatory type of foundation to support superstructure having heavy loads or laying on weak ground. Besides drilled shafts, driven piles made of timber, steel, precast concrete and composite are also an effective solution in terms of cost and quality. In order to design pile foundation, the axial pile bearing capacity is regarded as the most important parameter. Therefore, estimating this parameter has been the subject of numerous theoretical and experimental researches in geotechnics.

Overall, there are five main methods to evaluate the pile bearing capacity, namely the static analysis, dynamic analysis, dynamic testing, pile load test, and in-situ testing [1]. Design guidelines based on static analysis often

[*]*Corresponding Author:* Tran Thu Hien, Faculty of Civil Engineering, Duy Tan University, 55000, Danang, Vietnam; Faculty of Civil Engineering, Duy Tan University, 55000, Danang, Vietnam;
Email: tranthuhien197@gmail.com

recommend using the critical depth concept. However, the critical depth is an idealization that has neither theoretical nor reliable experimental support, and it contradicts physical laws.

Dynamic analysis methods are based on wave mechanics for the hammer-pile-soil system. The ambiguity in the hammer impact effect, as well as changes in soil strength from the conditions at the time of pile driving, and at the time of loading, cause uncertainties in bearing capacity determination. Dynamic testing methods are based on monitoring acceleration and strain near the pile head during driving. However, the measurements can only be analyzed by an experienced person. In addition, another considerable limitation is that the capacity estimation is not available until the pile is driven. The pile load test, a field measurement of full-scale pile settlement subjected to static load, is believed to provide the most accurate results. However, this method is time consuming and costly. Therefore, developing a simple, economical and accurate method is highly desired.

The measurements of soil properties by in-situ test methods have developed rapidly since 1970's. Concurrent with this development is the increasing use of in-situ test data in prediction of pile bearing capacity. The common tests include: standard penetration test (SPT), cone penetration test (CPT), flat dilatometer test (DMT), pressuremeter test (PMT), plate loading test (PLT), dynamic probing test (DP), press-in and screw-on probe test (SS) and field vane test (FVT). Each test applies different loading schemes to measure the corresponding soil response in an attempt to evaluate material characteristics. Among these in-situ test data, the SPT is commonly used to predict the bearing capacity of piles [2].

Different SPT data based methods for determining the bearing capacity of piles have been proposed in the literature. They can be categorized into two main approaches, direct and indirect methods. Amongst the two, the direct methods are more widely accepted among field engineers due to the ease of computation. For example, Decourt [3] proposed SPT direct methods for sandy or clayed soil proposed. For a case study in Iran [4], the authors analyzed the pile by a finite element method and compared it with four different SPT direct methods to find a reasonable prediction for its bearing capacity. However, according to Shariatmadari et al. [5], all of these empirical formulations have some inadequacies. Therefore, researchers have been exploring other ways to utilize SPT data to predict pile bearing capacity and using machine learning algorithms is a viable option.

Machine learning (ML), a branch of artificial intelligence, that mimics the operation of human brain, can non-linearly infer new facts from adaptively learning historical data. Moreover, the performance of machine learning (ML)-based models can be improved gradually along with the increase of learning data, so they can be kept up-to-date with high requirements of accuracy for complex engineering problems. Many contributions have demonstrated the effectiveness and efficiency of ML-based models to deal with civil engineering-related problems, for example, predicting mechanical properties (compressive/tensile strength/shear) of hardened concrete [6], estimation of tribological and rheological properties of fresh concrete [7], ultimate bond strength of corroded reinforcement and surrounding concrete [8], evaluation and detection of concrete structure deterioration.

ML-based models, especially Artificial Neural Network (ANN), have also been used to

predict pile bearing capacity. Early works in this direction include Lee and Lee [9]; Teh et al. [10], where ANN with error back propagation is utilized.

Recently, in the field of machine learning, hybrid models that incorporate mutual merits of different techniques have attracted increasing interest because of their robustness, efficiency, and superiority to baseline models. For instance, ensemble models use boosting, stacking, or voting strategies to compensate errors of each constituent model [1]. Especially, Extreme Gradient Boosting (XGBoost), an ensemble tree model, has become very popular (XGBoost 2021). Le et al. [11] recently utilized XGBoost-based ensemble model for predicting heating load of buildings for smart city planning and concluded that the proposed ensemble model is the most robust in comparing with other machine learning models, including classical XGBoost model, SVM, Random Forest (RF), Gaussian process (GP), and classification and regression trees (CART). Nguyen et al. [12] recently demonstrated XGBoost to be a promising tool to assist civil engineers in forecasting deflections of reinforced-concrete members. In addition, the outstanding performance of XGBoost-based models has been further convincingly demonstrated in a variety of practical problems [13,14].

Motivated by the successes of XGBoost-based ensemble models, this study aimed to investigate an XGBoost-based model to predict bearing capacity of reinforced concrete pile and compared its performance with those of other baseline popular machine learning models, including deep ANN and Random Forrest.

2. Research methodology

2.1. The collected dataset of static load tests

To train and validate the proposed machine learning method, this study relies on a dataset of static load test of driven reinforced concrete piles. This dataset includes 472 complied in the previous work of Pham et al. [15]. This is a fairly large dataset and highly appropriate for constructing and verifying sophisticated machine learning models. It is noted that pre-cast piles with closed tips are driven into soil layers with the employment of hydraulic pile driven machine to record capacity of piles. **Fig. 1** demonstrates the experimental set-up used for data measurement. **Fig. 2** provides illustrations for the pile structure, its geometrical variables, and soil stratigraphy. Herein, the predictor variables include pile diameter (X1), thickness of the first soil layer (X2), thickness of the second soil layer (X3), thickness of the third soil layer (X4), elevation of the natural ground (X5), top of pile elevation (X6), elevation of the extra segment of pile top (X7), depth of pile tip (X8), mean value of SPT blow count along pile shaft (X9), and mean value of SPT blow count at pile tip (X10). Those ten conditioning factors are employed to predict the dependent variable Y which is the axial pile bearing capacity. In addition, **Table 1** reports statistical descriptions of the predictor and dependent variables.

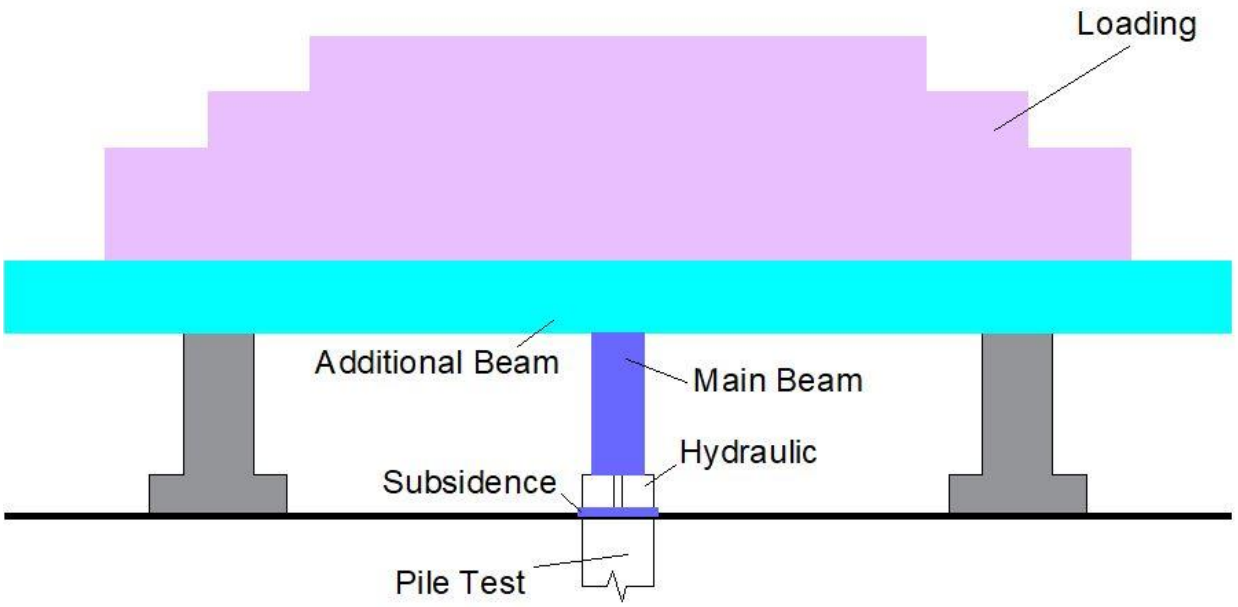

Fig. 1 Static load test experiment for measuring pile bearing capacity

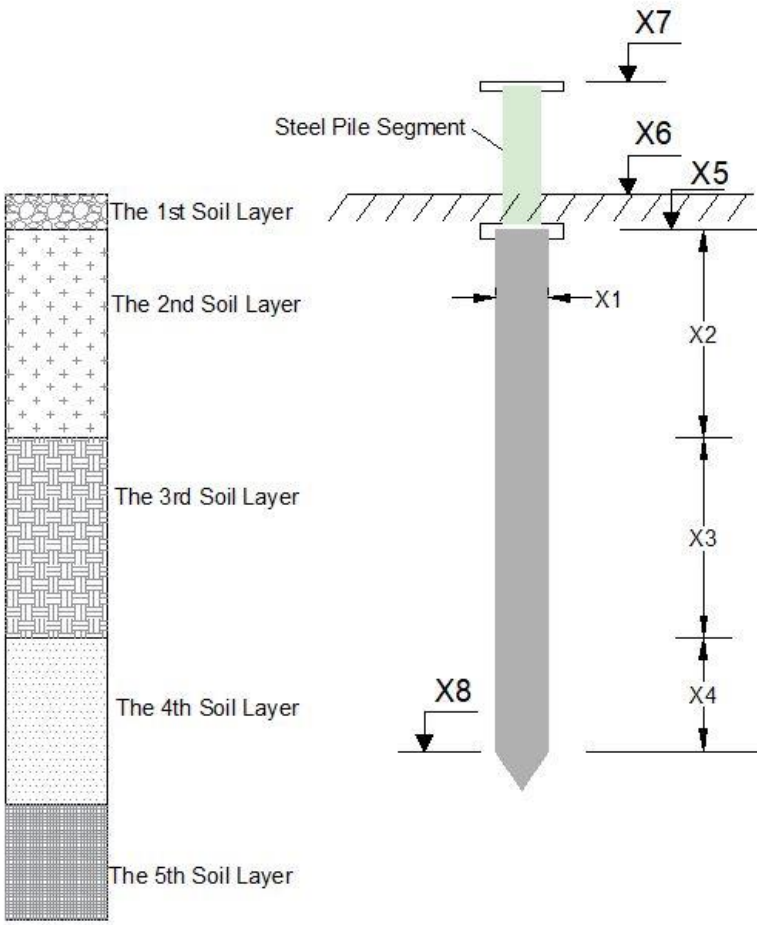

Fig. 2 Demonstration of the pile structure and soil stratigraphy

Table 1. Statistical descriptions of the employed variables

Variables	Notation	Min	Average	Std.	Max
Diameter of pile (mm)	X1	300.00	363.77	48.12	400.00
The thickness of the first soil layer that pile embedded (m)	X2	3.40	3.83	0.48	5.72
The thickness of the second soil layer that pile embedded (m)	X3	1.50	6.58	1.64	8.00
The thickness of the third soil layer that pile embedded (m)	X4	0.00	0.33	0.46	1.69
Pile top elevation (m)	X5	0.68	2.80	0.62	3.40
Natural ground elevation (m)	X6	3.04	3.50	0.08	4.12
The elevation of extra segment pile top (m)	X7	1.03	2.92	0.60	4.35
The depth of pile tip (m)	X8	8.30	13.54	1.80	16.09
The average SPT blow count along the pile shaft	X9	5.60	10.74	2.26	15.41
The average SPT blow count at the pile tip	X10	4.38	7.06	0.66	7.75
The axial bearing capacity load of pile (kN)	Y	407.20	984.20	353.21	1551.00

2.2. Extreme Gradient Boosting (XGBoost) machine

XGBoost is an open-source library that provides machine learning algorithms, both regression and classification, in the *gradient boosting framework*. It was originated from an academic research project but has become a widely used library in both academia and industry. The library is highly efficient, flexible and portable. It supports multiple languages, including C++, python, R, Java, Scala and Julia. The library also supports distributed training on clusters on cloud computing platforms, such as Amazon Web Services, Google Cloud Platform and Microsoft Azure.

Under the hood, the XGBoost algorithm builds a series of weak leaners, which are *classification or regression trees* (CART). These weak learners are then combined to form the final prediction model. Like other boosting methods, XGBoost do not build all the regression trees at the same time but step by step. The tree in the current step is constructed in such a way that it minimizes the average value of the loss function of all the steps on the training set.

More specifically, let the training data be $D = \{x_i, y_i\}_{i=1}^{n}$, where $x_i \in R^m$ is an *input vector* with m features, and $y_i \in R$ is the corresponding *output*. Assume $\hat{y}_i^{(t-1)}$ is the prediction output at step $t-1$. Then, at step t, XGBoost builds the tree that minimizes the following objective function:

$$L^t = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \Omega(f_t) = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \quad (1)$$

where l is any convex and differentiable *loss function* measuring the difference of the prediction and the provided output, $f_t: R^m \to R, f_t(x) = w_{q(x)}$ is the prediction function of the tree, where $w \in R^T$ is the vector of scores on leaves, $q: R^m \to \{1, 2, \ldots, T\}$ is a function assigning each data point to the corresponding leaf, and T is the number of leaves. The last term, Ω, is the regularization term. Its purpose is to reduce overfitting, a common issue in machine learning. This term penalizes complex trees with many leaves and

gives priority to more simple and predictive trees, more specifically

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \quad (2)$$

where γ and λ are parameters.

Approximating the right hand side of Eq. (1) by using the second-order Taylor expansion of l w.r.t to the second variable we have:

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

where $g_i = \partial_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$ and $h_i = \partial^2_{\hat{y}_i^{(t-1)}} l\left(y_i, \hat{y}_i^{(t-1)}\right)$.

Removing the term $\sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)}\right)$, which does not depend on the choice of the decision tree in the current step, from Eq. (2) and collecting the terms associated with the same score (the data points on the same leaf gets the same score), we are left with a simpler objective function that needs to be minimized:

$$\begin{aligned}\tilde{L}^t &= \sum_{i=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \Omega(f_t) \\ &= \sum_{j=1}^{T}\left[\left(\sum_{i\in I_j} g_i\right)w_j + \frac{1}{2}\left(\sum_{i\in I_j} h_i + \lambda\right)w_j^2\right] + \gamma T \end{aligned} \quad (4)$$

where I_j is the subset of the input set associated with leaf j, i.e., $I_j = \{i : q(x_i) = j\}$.

It can be noted that the first term of **Eq. (4)** is a sum of independent quadratic functions in w_j. Therefore, the optimal w_j^* and the minimal objective $\tilde{L}^{*t}$ are given by:

$$w_j^* = -\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i + \lambda}, \quad \tilde{L}^{*t}(q) = -\frac{1}{2}\sum_{j}^{T}\frac{\left(\sum_{i\in I_j} g_i\right)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T \quad (5)$$

This equation can be used to measure how good a tree is as a candidate for the current step. However, these optimal values can only be calculated when the structure of the tree in the current step has already determined. As it is not feasible to consider all the possible tree structures, XGBoost built trees iteratively.

At the beginning, XGBoost sorts the input data set according to feature values to form a tree with zero depth. Then in each step, a new tree is created by an optimal branch splitting. According to **Eq. (5),** this splitting, which maximizes the lost reduction, is calculated by

$$\tilde{L}_{split} = -\frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} + \frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I} g_i\right)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma \quad (6)$$

where I_L is the subset of input indices on the left of the split and I_R is the subset of indices on the right of the split.

The XGBoost algorithm is summarized in **Fig. 3**. In addition to using the regularization term Ω in **Eq. (1)**, XGBoost let users specify two parameters, max_depth and *learning rate* η, to combat overfitting. The parameter max_depth, as suggested by the name, limits the maximal tree depth that is allowed by XGBoost. The possible range for this parameter is $[0, \infty]$ and the default value is $max\_depth = 6$. The learning rate, also called shrinkage, scales the prediction of newly built tree by a factor $0 < \eta < 1$ to reduce the influence of each individual tree and allow trees

in the later steps chances to improve the model. More specifically:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta f_t(x_i) \quad (71)$$

The default value of η is 0.3.

The XGBoost algorithm

Specify the algorithm parameters
Sort the input set by feature values
For each iteration t
  Calculate g_i and h_i
    Examine the current tree and decide the best split using **Eq. (6)**
    The weights of the new tree $f_t(x)$ is computing using **Eq. (5)**
    The new prediction is computed using **Eq. (7)**

Return the trained XGBoost model

Fig. 3 The XGBoost algorithm

3. Experimental results and discussion

For our experiments, we have developed XGBoost model in Python using the following packages: i) XGBoost Python package version 0.90, the official implementation of XGBoost in Python ii) scikit-learn version 0.23.2, a machine learning library for Python language. For XGBoost, the objective is set to be reg: squarederror (regression with square error) and the number of boost rounds (number of iterations) is chosen to be 100. The data consisting of 472 samples is randomly split into two sets: a training set of 424 samples (90%) and a testing set of 48 samples (10%).

In order to accurately assess the predictive capability of different models, the following performance metrics are considered: RMSE, the mean absolute percentage error (MAPE), the mean absolute error (MAE) and the coefficient of determination (R^2). RMSE has been introduced in the previous section. MAPE, MAE, and R^2 are given by:

$$MAPE = \frac{100\%}{N}\sum_{i=1}^{N}\frac{|Y_{A,i}-Y_{P,i}|}{Y_{A,i}} \quad (8)$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\frac{|Y_{A,i}-Y_{P,i}|}{Y_{A,i}} \quad (9)$$

$$R^2 = 1 - \sum_{i=1}^{N}(Y_{A,i}-Y_{P,i})^2 / \sum_{i=1}^{N}(Y_{A,i}-\bar{Y})^2 \quad (10)$$

where $Y_{A,i}$ and $Y_{P,i}$ are the actual and the predicted bearing capacity, respectively; N is the number of data instances; $\bar{Y}$ is the average of actual values. It should be noted that smaller RMSE, MAE or MAPE is better, while higher R^2 is better.

Table 2 presents the performance of XGBoost with default parameters recommended by the XGBoost toolbox.

Table 2. Performance of XGBoost

Phase	Metrics	XGBoost
Training	RMSE	39.64
	MAE	21.02
	MAPE (%)	2.51
	R^2	0.99
Testing	RMSE	101.30
	MAE	73.77
	MAPE (%)	7.78
	R^2	0.92

In this section, to confirm the predictive capability of the newly model XGBoost used for pile bearing capacity prediction, its performance is compared to the capable machine-learning-based models based on Deep Neural Network (DNN) for regression. The DNN has been trained with the state-of-the-art Adam optimizer [16] and implemented via the scikit-learn Python library [17]. The DNN has been constructed with 2, 3, 4, and 5 hidden layers. The DNN is selected as benchmark models in this section because neural networks have been extensively and successfully employed in data-driven pile capacity estimation. In DNN, ReLU (Rectified Linear Unit) is employed and the number of training epochs is set to be 1000; the number of neurons in the hidden layers is selected via a five-fold cross validation.

Table 3. Performance statistic in the training phase

Indices	DNN 2 Layers		DNN 3 Layers		DNN 4 Layers		DNN 5 Layers		XGB	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
RMSE	93.85	3.22	89.37	4.25	90.71	1.29	94.97	3.57	39.64	2.25
MAE	72.47	2.11	68.94	3.4	70.19	0.73	74.35	3.59	21.02	1.47
MAPE	7.63	0.21	7.24	0.34	7.33	0.06	7.8	0.42	2.51	0.19
R^2	0.93	0.005	0.94	0.006	0.93	0.03	0.93	0.006	0.99	0.004

Table 4. Performance statistic in the testing phase

Indices	DNN 2 Layers		DNN 3 Layers		DNN 4 Layers		DNN 5 Layers		XGB	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
RMSE	98.85	11.58	98.59	15.13	95.49	18.9	98.9	10.22	86.99	9.59
MAE	75.58	9.61	74.54	10.23	73.53	10.6	76.85	8.35	73.77	8.79
MAPE	8.06	0.9	7.82	0.95	7.7	0.97	8.07	0.84	7.78	0.8
R^2	0.92	0.02	0.92	0.03	0.93	0.03	0.93	0.02	0.92	0.03

In **Table 3** and **Table 4**, the means and standard deviations of all the metrics are provided for all considered models in both training and testing phases. Variants of DNN with different number of hidden layers (2, 3, 4 and 5) appear to have similar performances, especially in the testing phase. Among them, the variant with 4 layers is the better one. However, even this variant lags behind XGBoost in both training and testing phase. The reductions in the average of testing RMSE of XGBoost compared to DNN with 2, 3, 4 and 5 layers are roughly 12%, 11.7%, 9% and 12% respectively. Not only XGBoost has better mean but it also has better median and interquartile range in all metrics. The performances of XGBoost in all metrics are also very robust.

4. Concluding remarks

In this paper, we have formulated and tested the XGBoost model in predicting bearing capacity of concrete piles. XGBoost is the crucial part of the model providing the prediction from a set of ten feature variables, including thickness of the first, second and third soil layer, pile diameter, elevation of the natural ground, of top of pile and of the extra segment of pile top, depth of pile tip, mean value of SPT blow count along pile shaft and of SPT blow count at pile tip.

The model is trained, validated and compared on subsets of a dataset consisting of 472 samples. The results indicate that the default XGBoost model is more accurate and robust than the DNN models. Therefore, it is highly recommended to use in pile bearing capacity prediction. Incorporating advanced feature selection and utilizing other state-of-the-art metaheuristic are a few lines of research that can be considered to advance further the study we have proposed.

References

[1] K. Pham, D. Kim, S. Park, H. Choi. (2021). "Ensemble learning-based classification models for slope stability analysis". *CATENA 196*:104886. DOI: 10.1016/j.catena.2020.104886

[2] A. Benali, A. Nechnech, A. Bouafia. (2013). "Bored Pile Capacity by Direct SPT Methods Applied to 40 Case Histories". *Civil and Environmental Research 5*, 118-122.

[3] L. Decourt. (1995). "Prediction of load settlement relationships for foundations on the basis of the SPT-T". *Ciclo de Conferencias Inter"Leonardo Zeevaert", UNAM, Mexico,* 85–104.

[4] I. Shooshpasha, A. Hasanzadeh, A. Taghavi. (2013). *"*Prediction of the axial bearing capacity of piles by SPT-based and numerical design methods". *International Journal of GEOMATE 4*, 560-565.

[5] N. Shariatmadari, A. Eslami, M. Karimpour-Fard. (2008). *"*Bearing capacity of driven piles in sands from SPT-applied to 60 case histories". *Iranian Journal of Science & Technology, Transaction B, Engineering 32*,125-140.

[6] D.K. Bui, T. Nguyen, J.S. Chou, H. Nguyen-Xuan, T.D. Ngo. (2018). "A modified firefly algorithm-artificial neural network expert system for predicting compressive and tensile strength of high-performance concrete". *Construction and Building Materials 180*, 320-333. DOI: 10.1016/j.conbuildmat.2018.05.201.

[7] T.D. Nguyen, T.H. Tran, N.D. Hoang. (2020). "Prediction of interface yield stress and plastic viscosity of fresh concrete using a hybrid machine learning approach". *Advanced Engineering Informatics 44*,101057. DOI: 10.1016/j.aei.2020.101057.

[8] N.D. Hoang, X.L. Tran, H. Nguyen H. (2020). "Predicting ultimate bond strength of corroded reinforcement and surrounding concrete using a metaheuristic optimized least squares support vector regression model". *Neural Computing and Applications 32*, 7289-7309. DOI: 10.1007/s00521-019-04258-x.

[9] I.M. Lee, J.H. Lee. (1996). "Prediction of pile bearing capacity using artificial neural networks". *Computers and Geotechnics 18*, 189-200. DOI: 10.1016/0266-352X(95)00027-8

[10] C.I. Teh, K.S. Wong, A.T.C. Goh, S. Jaritngam. (1997). "Prediction of Pile Capacity Using Neural Networks". *Journal of Computing in Civil Engineering 11*, 129-138.

[11] L.T. Le, H. Nguyen, J. Zhou, J. Dou, H. Moayedi. (2019). "Estimating the Heating Load of Buildings for Smart City Planning Using a Novel Artificial Intelligence Technique PSO-XGBoost". *Applied Sciences 9*, 2714. DOI: 10.3390/app9132714

[12] H. Nguyen, N.M. Nguyen, M.T. Cao, N.D. Hoang, X.L. Tran. (2022). *"*Prediction of long-term deflections of reinforced-concrete members using a novel swarm optimized extreme gradient boosting machine". *Engineering with Computer 38*, 1255-1267. DOI: 10.1007/s00366-020-01260-z

[13] W. Dong, Y. Huang, B. Lehane, G. Ma. (2020). "XGBoost algorithm-based prediction of concrete electrical resistivity for structural health monitoring". *Automation in Construction 114*, 103155. DOI: 10.1016/j.autcon.2020.103155

[14] J. Duan, P.G. Asteris, H. Nguyen, X.N. Bui, H. Moayedi. (2021). "A novel artificial intelligence technique to predict compressive strength of recycled aggregate concrete using ICA-XGBoost model". *Engineering with Computers 37*, 3329-3346. DOI: 10.1007/s00366-020-01003-0

[15] T.A. Pham, V.Q. Tran, H.L.T. Vu, H.B. Ly. (2020). "Design deep neural network architecture using a genetic algorithm for estimation of pile bearing capacity". *PLOS ONE 15(12):* e0243030. DOI: 10.1371/journal.pone.0243030.

[16] D.P. Kingma, J. Ba. (2015). "Adam: A Method for Stochastic Optimization". *Proceedings of the 3rd International Conference on Learning Representations (ICLR), San Diego*.

[17] F. Pedregosa et al. (2011). "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research 12*, 2825−2830.