# Machine learning for prediction of undrained shear strength from cone penetration test data

Yu, Beiyang; Varkey, Divya; van den Eijnden, Abraham P.; Rongier, Guillaume; Hicks, Michael A.

*14th International Conference on Applications of Statistics and Probability in Civil Engineering, ICASP14*
*Dublin, Ireland, July 9-13, 2023*

# Machine Learning for Prediction of Undrained Shear Strength from Cone Penetration Test Data

Beiyang Yu
*Dept. of Geoscience and Engineering, Delft University of Technology, Delft, Netherlands*

Divya Varkey
*Dept. of Geoscience and Engineering, Delft University of Technology, Delft, Netherlands*

Abraham P. van den Eijnden
*Dept. of Geoscience and Engineering, Delft University of Technology, Delft, Netherlands*

Guillaume Rongier
*Dept. of Geoscience and Engineering, Delft University of Technology, Delft, Netherlands*

Michael A. Hicks
*Dept. of Geoscience and Engineering, Delft University of Technology, Delft, Netherlands*

ABSTRACT: This research focuses on investigating the relative performance of a range of machine learning algorithms, namely the artificial neural network, support vector machine, Gaussian process regression, random forest, and XGBoost, for predicting the undrained shear strength from cone penetration test data. This is to assess how machine learning could help us lower the need for laboratory test data. The training dataset compiles 526 data from 12 regions and the testing dataset consists of 20 data from a polder located close to Leiden in the Netherlands. In addition, k-fold and group k-fold cross-validation strategies are both applied to validate the models. The poor performance of the models during group k-fold cross-validation suggests that, while machine learning techniques can perform well when site-specific data are included during training, they struggle to generalize without site-specific data. This highlights the difficulty of capturing soil heterogeneity and suggests that either machine learning methods should be trained on specific sites for which some data are already available, or much larger training datasets are needed.

## 1. INTRODUCTION

Soil shear strength, defined as the capability of soils to withstand internal movement or slippage when subjected to an imposed load, has always been one of the most important parameters in soil mechanics studies. It plays a crucial role in the design phase of various infrastructures, such as foundations, embankments, earthen dams, and retaining walls.

In general, shear strength parameters can be estimated in the field as well as in a laboratory. In-situ tests include cone penetration tests, standard penetration tests, piezo-cone, field vane shear tests and pressure meter tests. In the laboratory, the parameters of soil shear strength are generally determined through the direct shear test or various forms of triaxial shear test, namely unconsolidated undrained triaxial test, consolidated undrained triaxial test and consolidated drained triaxial test.

As machine learning (ML) techniques have become increasingly popular, there has been an increasing number of applications of ML in diverse areas of science, especially in the last decade. In the context of soil research, the availability of an increasing number of large datasets of soil together

with open-source ML techniques has contributed to the increasing use of ML techniques in soil studies, such as in the prediction of soil properties via soil data using ML techniques (Padarian et al., 2020). For instance, Kanungo et al. (2014) assessed the effectiveness of artificial neural network (ANN) and regression tree techniques in predicting shear strength parameters. Ly and Pham (2020) studied the prediction of soil shear strength by applying a support vector machine (SVM) based on six input parameters, namely clay content, moisture content, specific gravity, void ratio, liquid limit, and plastic limit. It is worth mentioning that the datasets used in those studies are large for the soil mechanics community, but still considered very small in the ML community.

The cone penetration test (CPT) is a powerful and cost-effective tool for investigating subsoil conditions, and various empirical correlations are available for interpreting CPT data. These correlations, however, are not universally applicable to all soils and subsurface conditions. Therefore, in practice, CPT test data are usually complemented by laboratory test data to verify the applicability of the correlations. For large projects involving large amounts of data, however, laboratory-based studies of the subsoil can not only be more complex and tedious, but also more expensive. Instead, ML models based on, for example, random forest (RF) or ANN algorithms can be used, which makes the task much more efficient and economical. Thus, the motivation of this paper is to review and investigate the relative performance of a range of ML algorithms for predicting undrained shear strength through CPT data and provide a reference for selecting an effective ML algorithm.

## 2. MACHINE LEARNING METHODS

### *2.1. Nonlinear regression*

From linear regression to nonlinear regression, different models apply various techniques to introduce nonlinearity. ANN can be regarded as introducing nonlinearity through a combination of generalized linear models. SVM and Gaussian process regression (GPR) are both kernel-function (i.e. covariance function) -based algorithms. GPR differs from SVM in that it is also able to provide uncertainty estimates for its predictions.

#### *2.1.1. Artificial neural network*

ANN refers to a biologically inspired approach of ML modeled on the brain. Similar to the human brain, which has neurons interconnected with one another, ANNs have neurons that are interconnected to one another in various layers of a network. The interconnected artificial neural elements work in unison, sharing information to develop an awareness of the relationship between different parameters in order to learn or emulate how a system functions (Reale et al., 2018). Neural networks are typically arranged into an input layer, one or several hidden layers, and an output layer. The number of input and output nodes depends on the engineering problem being considered. The number of hidden neurons is one of the hyperparameters that needs to be tuned on a problem-by-problem basis. In this study, a feed-forward multi-layer perceptron using the back-propagation learning algorithm is applied. Feed-forward Neural Networks are ANNs where the node connections do not form a cycle. In this way, each layer's outputs serve as the input to the next layer. This allows using the back-propagation algorithm to efficiently train the neural network. Here, the output values are compared with the correct answer to compute the value of some predefined error function. By applying the automatic differentiation technique, the error is then fed back through the network. Using this information, the algorithm is able to adjust the weights of each connection in order to reduce the value of the error function by a small amount. After repeating this process for a sufficiently large number of training cycles, the network will usually converge to some state where the error of the calculations is small.

#### *2.1.2. Support vector machine*

The support vector machine originated from the concept of statistical learning theory pioneered by Boser et al. (1992). In this study, we use the SVM as a regression technique by introducing an error-insensitive loss function. There are three distinct characteristics when an SVM is used to estimate the regression function: the type of kernel func-

tion $f$, the optimum capacity factor $C$, and the optimum error insensitive zone $\varepsilon$. The main aim of an SVM is to find a function that gives a deviation of $\varepsilon$ from the actual output and, at the same time, is as flat as possible. The constant $C(0 < C < \infty)$ determines the trade-off between the flatness of $f$ and the amount up to which deviations larger than $\varepsilon$ are tolerated (Smola and Schölkopf, 2004). When linear regression is not appropriate, then input data have to be mapped into a high dimensional feature space through some nonlinear mapping technique (Boser et al., 1992). There are two steps in this exercise: firstly, carry out a fixed nonlinear mapping of the data onto the feature space; and secondly, carry out a linear regression in the high dimensional space. The input data are mapped onto the feature space. The concept of a kernel function has been introduced to reduce the computational demand. Some common kernels, such as homogeneous and non-homogeneous polynomial expressions, radial basis functions, Gaussian functions and sigmoid functions, and their combinations, have been used for nonlinear cases (Das et al., 2011).

#### *2.1.3. Gaussian process regression*

Being a probabilistic supervised ML model, the Gaussian process model has been widely used for both regression and classification tasks. A GPR model can make predictions incorporating prior knowledge through kernel functions and provide uncertainty measures over predictions. In contrast to the traditional nonlinear regression methods that typically give one function that is considered to fit the dataset best, a Gaussian process model is able to describe a probability distribution over possible functions that fit a set of points. Through this probability distribution over all possible functions, the mean function can be calculated and taken as the prediction. The prediction is updated as the number of observation points increases, and the variance can also be used to indicate how uncertain the predictions are.

### *2.2. Tree-based algorithms*

#### *2.2.1. Random forest*

The RF consists of a committee of decision trees. Each individual tree is a fairly simple model that has branches, nodes and leaves. The purpose of building a decision tree is to create a model that predicts the value of the target variable depending on several input variables. Firstly, sub-samples are generated from the training data by drawing with replacement. This is called bootstrap sampling. This random sampling with replacement ensures that we are not using the same data for every tree, so it helps our model to be less sensitive to the original training data. Next, the models are built by constructing a decision tree for each sub-sample based on a random set of features. This random selection of features is important, since, if every feature is used, then most of the trees will have the same decision nodes and will act very similarly which decreases the variance.

#### *2.2.2. XGBoost*

XGBoost is a tree-based method that uses trees in a sequential manner (so that a tree is trained to improve the prediction of the previous tree), in contrast to RF which uses trees in parallel. It is based on gradient boosting together with some advanced optimizations. Gradient boosting is an iterative optimization algorithm used in ML to minimize the loss function, which is a measure of how good the prediction model does in terms of being able to predict the expected outcome. XGBoost, however, is an optimized gradient-boosting ML library. It is a more regularized form of gradient boosting using advanced regularization, as the loss function in XGBoost includes two basic components, training loss, and regularization. Training loss measures how well the model fits into the training data. Regularization measures the complexity of the model. The regularization terms are added into the loss function to penalize complex models to avoid overfitting, thereby improving the model generalization capabilities.

## 3. DATASET

### *3.1. Training dataset*

The Clay/6/535 database (Ching et al., 2014) was chosen to be the preliminary dataset in this study. It comprises 535 data points of lightly over-consolidated clay data from 40 sites with the following measurement information: normalized

undrained shear strength ($\frac{Su}{\sigma'_v}$), over-consolidation ratio ($OCR$), normalized cone tip resistance ($\frac{q_t-\sigma_v}{\sigma'_v}$), normalized effective cone tip resistance ($\frac{q_t-u_2}{\sigma'_v}$), normalized excess pore pressure ($\frac{u_2-u_0}{\sigma'_v}$) and pore pressure ratio ($\frac{u_2-u_0}{q_t-\sigma_v}$), together with the effective stress ($\sigma'_v$) and the depth of each measured point. The 40 sites are located in the following geographical regions: Brazil, Canada, Hong Kong, Italy, Malaysia, Norway, Singapore, Sweden, UK, USA, the North Sea, and Venezuela.

The preliminary dataset is pre-processed by selecting the input variables, handling null values, removing the outliers, and scaling the features. A total of four input variables are chosen in predicting the shear strength of the soil, including the effective stress ($\sigma'_v$), cone tip resistance ($q_t - \sigma_v$), effective cone tip resistance ($q_t - u_2$) and excess pore pressure ($u_2 - u_0$). After removing 9 outliers, in total 526 data points are compiled into the training dataset used in this research. As the training dataset is too small in an ML context, cross-validation (CV) strategies are applied to the training dataset. The ratio of training data to validation data is taken as 90/10.

### 3.2. *Testing dataset*

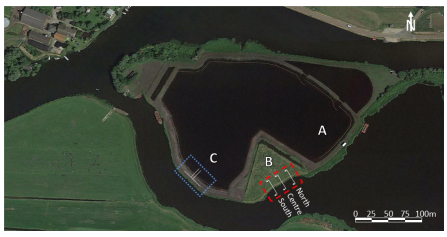



*Figure 1: An aerial photograph of Leendert de Boerspolder, taken in 2015, indicated are: (A) and (B) locations not related to this study; (C) location where 100 CPTs were conducted and where 20 samples were collected for laboratory tests (de Gast, 2020).*

The testing dataset is from a polder (see Figure 1) in the Netherlands, in which 100 CPTs were performed and 20 samples were collected from 11 boreholes (de Gast, 2020). The relative locations of the CPTs and boreholes are illustrated in Figure 2.

The CPTs in the vicinity of the boreholes utilize GPR to get representative inputs at the locations of the boreholes. An illustration of the application of GPR on the prediction of representative CPT data at the location of a borehole, through several CPTs that are in close vicinity to it, is provided in Figure 3. (The confidence interval is not included since multiple inputs with uncertainties are not considered.) To obtain the representative CPT data at the location of borehole B1009, which is for providing samples for laboratory tests, CPT index 66, 88, and 99 are selected since they are in close vicinity. The raw data of these three CPTs are first interpreted with a script. After the interpretation, the corrected cone tip resistances ($q_t$) of the three CPTs are plotted with dotted lines in the figure. Then GPR, which applies a Matern kernel function, is applied to these three sets of data to figure out the final prediction which is plotted with a black solid line in the figure. Lastly, the representative corrected cone tip resistance value of the laboratory sample B1009-4, which is located at a depth of 3.92 m, can be evaluated through the prediction line.

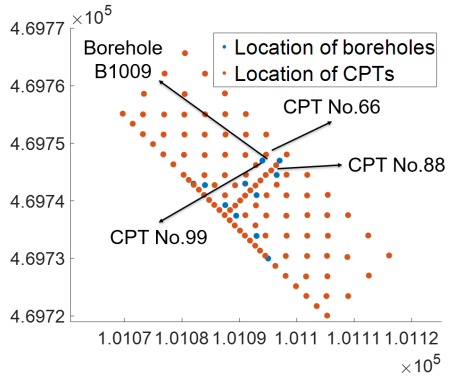



*Figure 2: The locations of 11 boreholes for the laboratory tests and 100 CPTs at location C using Global RD (Dutch reference) XYZ-coordinates.*

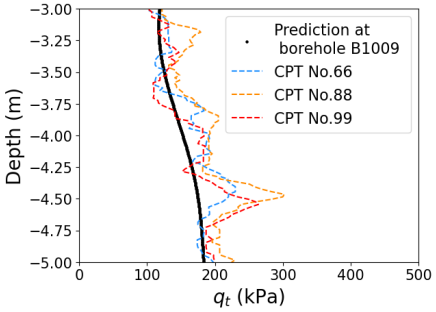
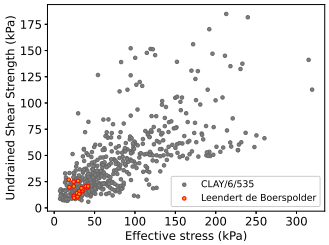


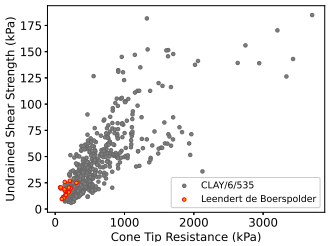

*Figure 3: An illustration of the application of Gaussian process regression for processing CPT data.*

### *3.3. Comparison between training and testing dataset*

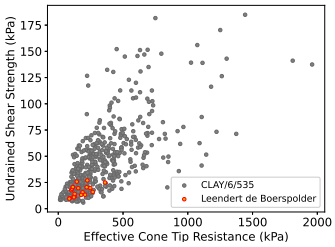
In order to investigate the applicability of the tuned ML models to the testing dataset, Figure 4 shows the correlations between the inputs and the output for the two datasets. The figure demonstrates that the testing dataset is sufficiently within the Clay/6/535 database and that the correlations are not site-specific.



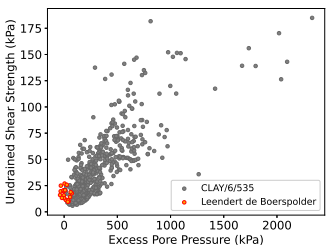

*(a)* $\sigma'_v$ *vs.* $S_u$.




*(b)* $(q_t - \sigma_v)$ *vs.* $S_u$.




*(c)* $(q_t - u_2)$ *vs.* $S_u$.




*(d)* $(u_2 - u_0)$ *vs.* $S_u$.

*Figure 4: Correlations between the inputs and the output in the training and testing dataset.*

## 4. MODEL IMPLEMENTATION

### *4.1. Cross-validation strategies*

Cross-validation is one of the techniques used to test the effectiveness of an ML model, by testing the model on some unseen data. It is also a resampling procedure used to evaluate a model if only a limited amount of data are available, which is the case in this study. There are various kinds of CV strategies. The strategy to perform CV depends on the scenario in the future application. Starting with the most commonly used k-fold CV, a visualization of which is shown in Figure 5, the dataset is split into $k$ consecutive folds, with each fold being used once as the validation, while the $k-1$ remaining folds form the training set. Subsequently, the model is fitted to the training set, and evaluated on the validation set using statistical metrics, namely $R^2$, mean absolute error (MAE) and root mean square error (RMSE). Finally, the average of the scores in $k$ iterations is taken as the performance metric for the model.

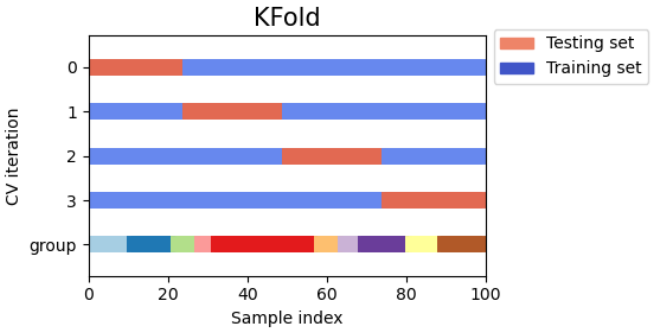



*Figure 5: Visualization of a 4-fold CV.*

Group k-fold CV is a variation of k-fold CV which ensures that the same group is not represented in both testing and training sets. An example visualization is shown in Figure 6, where in this paper, a 'group' corresponds to a 'site'. This strategy best simulates the scenarios where the model will be tested on completely unseen data, which in this study is data from new sites. This would be an invaluable application for ML, allowing us to start the exploration of a site even before gathering any data.

To sum up, k-fold CV is a typical CV strategy. It can be regarded as an ideal CV strategy in an ML context because the training and testing population share the same distribution, which in essence

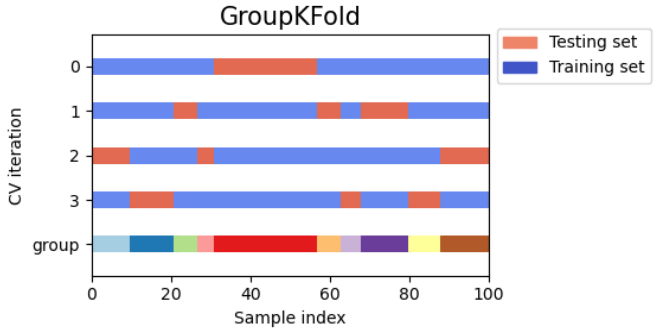



*Figure 6: Visualization of a 4-fold group CV.*

means that the scenario it is simulating tends to be milder, thus leading to a relatively good result. On the contrary, the group k-fold CV simulates a more complex scenario, in which the testing distribution can be very different from the training distribution. This constitutes a tougher challenge to ML algorithms, which generally leads to a relatively poor result. Implementing these CV strategies for training the models and assuming $k = 10$, we end up with two sets of trained models for the testing dataset: one is relatively conservative and the other is more radical. This assures a more objective and comprehensive evaluation of the performance of the ML models.

### *4.2. Hyperparameter tuning*

In training the ML models, the hyperparameters need to be calibrated based on their performance on a validation set. Grid search is a powerful tool to use for this purpose as it can exhaustively search over specified parameter values for an estimator (Pedregosa et al., 2011). However, grid search CV can quickly become too computationally expensive due to its exhaustiveness. Conversely, in random search CV, not all parameter values are tried out, but a fixed number of parameter settings is sampled from specified distributions. The latter has proven to be able to find models that are as good or even better within a small fraction of the computation time (Bergstra and Bengio, 2012). In this research, grid search CV is applied to SVM and GPR, since there are only 3 or 4 hyperparameters that require tuning. Meanwhile, random search CV has been applied to RF, XGBoost, and ANN, since they have a lot more hyperparameters to tune.

## 5. RESULTS

The hyperparameters in each ML model are first calibrated using grid search CV or random search CV based on their averaged performance on the validation sets. As mentioned in Section 3.1, the validation dataset constitutes 10% of the traning dataset. As an example, the XGBoost parameters tuned with k-fold CV and group k-fold CV are present in Table 1.

*Table 1: XGBoost parameters tuned with k-fold CV and group k-fold CV.*

| Parameters | Value (k-fold) | Value (group k-fold) |
|---|---|---|
| n_estimators | 47 | 14 |
| subsample | 0.50 | 0.60 |
| learning_rate | 0.11 | 0.11 |
| max_depth | 6 | 2 |
| reg_alpha | 2.60 | 1.00 |
| reg_lambda | 0.10 | 1.60 |
| gamma | 4.80 | 3.90 |
| booster | gbtree | gbtree |

The CV results in the validation dataset are summarized in Table 2. It is taken as the main reference for evaluating the relative performance of the algorithms in this study. The results of the k-fold CV are close and satisfying overall. However, the results of the group k-fold CV are rather poor.

*Table 2: The CV results in the validation dataset.*

| CV | Method | $R^2$ | MAE | RMSE |
|---|---|---|---|---|
| k-fold | ANN | 0.76 | 10.87 | 15.49 |
| | SVM | 0.76 | 11.13 | 15.32 |
| | GPR | 0.73 | 11.74 | 16.34 |
| | RF | 0.75 | 10.89 | 15.70 |
| | XGBoost | 0.76 | 10.79 | 15.45 |
| group k-fold | ANN | 0.13 | 16.44 | 21.21 |
| | SVM | -0.29 | 17.76 | 22.76 |
| | GPR | -0.41 | 15.89 | 20.21 |
| | RF | -0.95 | 17.27 | 22.45 |
| | XGBoost | -0.12 | 16.61 | 22.03 |

After tuning the hyperparameters, the constructed ML models are applied to the whole training and testing dataset. For instance, the results of

XGBoost in the training and testing dataset are presented in Figures 7 and 8 respectively.

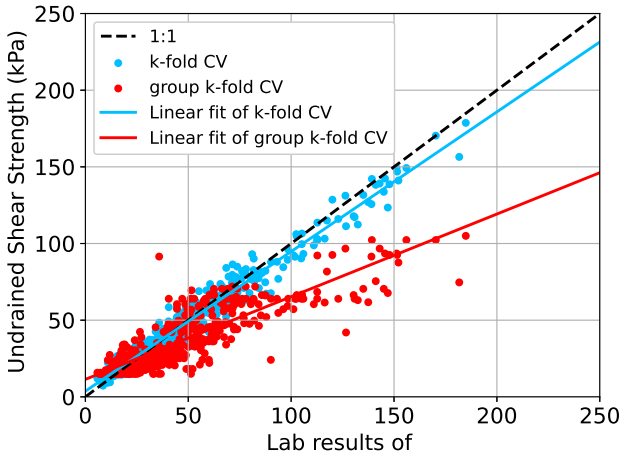



*Figure 7: Best XGBoost models from k-fold and group k-fold CV on the training set.*

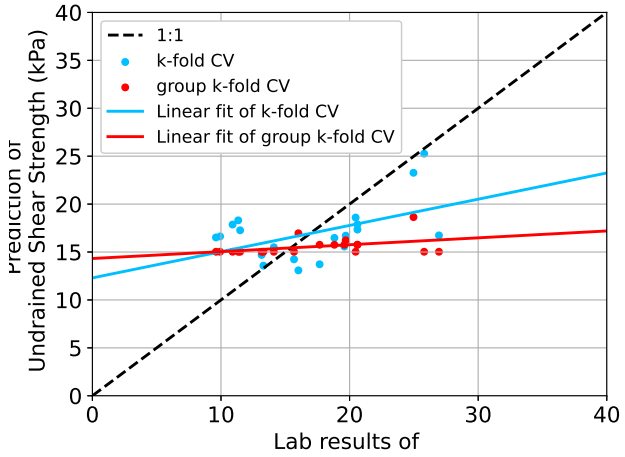



*Figure 8: Best XGBoost models from k-fold and group k-fold CV on the testing set.*

A summary of the predictive capabilities of the algorithms using k-fold and group k-fold CV are presented in Tables 3 and 4 respectively. Note that GPR shows identical performance in both tables since the models obtained with both CV strategies were identical.

*Table 3: Summary of the prediction capabilities of the algorithms whose hyperparameters are selected using k-fold CV.*

| Part | Method | $R^2$ | MAE | RMSE |
|---|---|---|---|---|
| Training | ANN | 0.79 | 10.69 | 14.96 |
| | SVM | 0.83 | 9.85 | 13.38 |
| | GPR | 0.76 | 11.30 | 15.81 |
| | RF | 0.91 | 6.92 | 9.69 |
| | XGBoost | 0.96 | 5.17 | 6.67 |
| Testing | ANN | -1.85 | 7.30 | 8.77 |
| | SVM | -0.31 | 5.09 | 5.94 |
| | GPR | -0.38 | 4.43 | 6.09 |
| | RF | 0.13 | 3.75 | 4.85 |
| | XGBoost | 0.23 | 3.73 | 4.55 |

*Table 4: Summary of the prediction capabilities of the algorithms whose hyperparameters are selected using group k-fold CV.*

| Part | Method | $R^2$ | MAE | RMSE |
|---|---|---|---|---|
| Training | ANN | 0.76 | 11.87 | 15.88 |
| | SVM | 0.76 | 11.14 | 15.82 |
| | GPR | 0.76 | 11.30 | 15.81 |
| | RF | 0.71 | 12.49 | 17.60 |
| | XGBoost | 0.62 | 13.06 | 20.04 |
| Testing | ANN | -1.36 | 6.78 | 7.98 |
| | SVM | -2.28 | 7.87 | 9.40 |
| | GPR | -0.38 | 4.43 | 6.09 |
| | RF | -0.53 | 5.31 | 6.42 |
| | XGBoost | 0.03 | 4.22 | 5.11 |

The results in the training dataset using both CV strategies are satisfying overall, showing that the ML models have successfully learned and captured the data characterization in the training dataset. The results of the testing dataset show how the models perform on completely different data, which have always been crucial information to consider. As can be seen from the poor prediction capabilities of the ML models on validation and testing datasets by using group k-fold CV (see Tables 2 and 4), the models struggle to generalize without site-specific data. On the other hand, using k-fold CV leads to being too optimistic about the true performance of the models (Table 2), as it assumes data to be independent and identically distributed, whereas the predictions are rather poor on unseen data (Table 3).

## 6. CONCLUSIONS

The effectiveness of using the five ML algorithms in this study in predicting the shear strength of the soil is validated when site-specific data are available. The performance of the algorithms is close and satisfying overall. However, the performance of the models during group k-fold CV is rather poor. Meanwhile, using k-fold leads to being too optimistic about the true performance of the algorithms because CPT data are not independent and identically distributed, but this is an assumption behind k-fold. Therefore, the poor performance of models validated with group k-fold CV indicates that while ML techniques can perform well when site-specific data are included during training, they struggle to generalize without site-specific data. This highlights the difficulty of capturing soil heterogeneity and suggests that either ML methods should be trained and used on specific sites for which some data are already available, or much larger training datasets are needed in order to construct a model that can be applied on a global scale.

There are, of course, limitations in this study, the main limitation being that the dataset we used is small. This is a common problem whenever researchers apply ML techniques to geotechnical problems, since the amount of data in geotechnical engineering is very limited in an ML context. To minimize the negative impact of this limitation, an appropriate CV strategy that best simulates future applications should be used to avoid overly optimistic estimates of results.

Looking to the future, CPT data are considered less labour-intensive and more cost-effective to obtain than laboratory data. This leads to the idea that a wider range of data sources that are less labour-intensive and more cost-effective than CPT can be used in the future. For instance, it should be possible to use geophysical data to develop ML models that are capable of predicting the CPT profile of the soil, thereby better characterizing site conditions.

## 7. ACKNOWLEDGEMENTS

The authors wish to thank ProRail for their financial support of the second author through the project RESET (Reliable Embankments for the Safe Expansion of Rail Traffic).

## 8. REFERENCES

Bergstra, J. and Bengio, Y. (2012). "Random search for hyper-parameter optimization." *Journal of Machine Learning Research*, 13(2), 281–305.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). "A training algorithm for optimal margin classifiers." *Proceedings of the fifth annual workshop on Computational Learning Theory*, 144–152.

Ching, J., Phoon, K.-K., and Chen, C.-H. (2014). "Modeling piezocone cone penetration (cptu) parameters of clays as a multivariate normal distribution." *Canadian Geotechnical Journal*, 51(1), 77–91.

Das, S., Samui, P., Khan, S., and Sivakugan, N. (2011). "Machine learning techniques applied to prediction of residual strength of clay." *Open Geosciences*, 3(4), 449–461.

de Gast, T. (2020). "Dykes and embankments: a geostatistical analysis of soft terrain." Ph.D. thesis, Delft University of Technology, The Netherlands.

Kanungo, D., Sharma, S., and Pain, A. (2014). "Artificial neural network (ann) and regression tree (cart) applications for the indirect estimation of unsaturated soil shear strength parameters." *Frontiers of Earth Science*, 8(3), 439–456.

Ly, H.-B. and Pham, B. T. (2020). "Prediction of shear strength of soil using direct shear test and support vector machine model." *The Open Construction and Building Technology Journal*, 14(1), 268–277.

Padarian, J., Minasny, B., and McBratney, A. B. (2020). "Machine learning and soil sciences: A review aided by machine learning tools." *Soil*, 6(1), 35–52.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). "Scikit-learn: Machine learning in Python." *Journal of Machine Learning Research*, 12, 2825–2830.

Reale, C., Gavin, K., Librić, L., and Jurić-Kaćunić, D. (2018). "Automatic classification of fine-grained soils using cpt measurements and artificial neural networks." *Advanced Engineering Informatics*, 36, 207–215.

Smola, A. J. and Schölkopf, B. (2004). "A tutorial on support vector regression." *Statistics and Computing*, 14(3), 199–222.