A thesis submitted in partial fulfillment for
Master in Civil Engineering
in Section of Geo–Engineering
at Delft University of Technology

Presented by:

Beiyang Yu

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

September, 2022

Committee

Prof.dr. M.A. Hicks (Chair)	Geo–Engineering, TU Delft
Dr.ir. A.P. (Bram) van den Eijnden	Geo–Engineering, TU Delft
Dr. G. (Guillaume) Rongier	Applied-Geology, TU Delft
Dr. D. Varkey	Geo–Engineering, TU Delft

Preface

"Do not go gentle into that good night. Old age should burn and rave at close of day. Rage, Rage against the dying light."

—— *Dylan Thomas*

"We've always defined ourselves by the ability to overcome the impossible. And we count these moments. These moments when we dare to aim higher, to break barriers, to reach for the stars, to make the unknown known. We count these moments as our proudest achievements. But we lost all that. Or perhaps we've just forgotten that we are still pioneers. And we've barely begun. And that our greatest accomplishments cannot be behind us, because our destiny lies above us"

—— *Interstellar*

Beiyang Yu

Delft, September 2022

Abstract

The need of shear strength measurements of soil in the design phase of geotechnical engineering is almost indispensable. Many methods have been applied to estimate the shear strength of soil, including various laboratory test, in-situ test and analytical methods. As an in-situ test method, cone penetration test (CPT) is a powerful and cost-effective tool for the investigation of subsoil conditions. CPT data is usually complemented by the laboratory test data for verification. The laboratory-based studies of subsoil, however, can be not only a complex but also tedious and expensive task for large projects involving large amount of data. Therefore, new approaches for estimating the soil shear strength are demanded. Having demonstrated superior predictive ability for many material properties compared to traditional methods, machine learning methods have been increasingly popular and widely used. This thesis focus on the prediction of soil undrained shear strength through cone penetration test data. The major objectives of this master thesis include testing how machine learning could help us lower the need for laboratory test data.

At first, the research starts with a literature review of various methods used to evaluate the soil shear strength. Comparing to the machine learning methods, the laboratory and in-situ test methods are relatively more time-consuming, costly and labour-intensive. And the analytical methods are considered lacking in precision.

Then the training dataset which consists of 526 samples is introduced. In each sample, there are four input variables obtained from cone penetration test, namely the effective stress (σ'_v), cone tip resistance ($q_t - \sigma_v$), effective cone tip resistance ($q_t - u_2$) and the excess pore pressure ($u_2 - u_0$). The undrained shear strength obtained from laboratory test is taken as the output variable.

Next, the training dataset is fed to five machine learning techniques, namely the artificial neural network, support vector machine, Gaussian process regression, random forest and XGBoost, to train models. The hyperparameters are tuned with k-fold and group k-fold cross-validation strategies in the validation process.

After that, the testing dataset which consists of 20 samples is established. Cone penetration test data that are in close vicinity to the location of the samples are processed by Gaussian process regression to obtain representative cone penetration test data at the sample location, which is taken as the inputs in the testing dataset. The undrained shear strengths of the samples are measured by Consolidated-Undrained shear test and are taken as the outputs of the testing dataset.

Finally, the five machine learning models are tested on the testing dataset. The cross-validation results, together with the prediction results of the models on the training and testing dataset are evaluated, gathered and compared by various statistic metrics to show the relative performance of the models. XGBoost appears to be the most accurate of all the tested algorithms on this dataset. And Gaussian process regression is chosen as the second option due to its ability to capture uncertainties. The robustness of these two models are then validated from a statistical point of view by applying Monte Carlo analysis. The importance of the input parameters in this study is evaluated by applying random forest for the sensitivity analysis. The results from random forest indicate that the excess pore pressure and the cone tip resistance - total vertical stress are the most influential inputs to the undrained shear strength.

Contents

List of Figures

List of Tables

Nomenclature

γ	Effective soil unit weight
ϕ	Friction angle
ϕ'	Effective friction angle
σ_f	Normal stress on the failure plane
σ'_f	Effective stress at failure
σ_h	Total horizontal stress
σ'_h	Effective horizontal stress
σ_v	Vertical stress
σ'_v	Vertical effective stress
τ_f	Shear strength at failure
τ'_f	Effective shear strength at failure
a	Net area ratio tip
B	Penetrometer diameter
c'	Effective cohesion
C_c	Compression index
e	Void ratio
f_s	Sleeve friction

LL Liquid limit

N_q Bearing capacity factor

q Effective stress

q_c Cone resistance

q_t Corrected cone resistance

R_f Friction ratio

S_u Undrained shear strength

u_0 Hydrostatic pore pressure

u_1 Pore pressure in the middle of the tip height

u_2 Pore pressure behind cone

u_3 Pore pressure above the friction sleeve

C Cohesion

Acronyms

AAE	absolute average error.
ANFIS	adaptive neuro-fuzzy inference system.
ANN	artificial neural network.
BNN	biological neural network.
CART	classification and regression trees.
CD	consolidated drained triaxial tests.
CPT	cone penetration test.
CSSM	critical state soil mechanics.
CU	consolidated undrained triaxial tests.
CV	cross-validation.
E	Nash–Sutcliffe coefficient of efficiency.
EHO	elephant herding optimization.
ELM	extreme learning machine.
FN	functional networks.
GANFIS	genetic algorithm - genetic adaptive neuro-fuzzy inference system.
GPR	Gaussian process regression.

HGSO	Henry gas solubility optimization.
HPC	high-performance concrete.
KNN	K-Nearest Neighbor.
MAE	mean absolute error.
ML	machine learning.
MLP	multi-layer perceptron.
MSE	mean squared error.
OCR	over-consolidation ratio.
PANFIS	particle swarm optimization - adaptive network-based fuzzy inference system.
PDP	partial dependence plot.
PNN	probabilistic neural network.
R	correlation coefficient.
R^2	coefficient of determination.
RBF	radial basis function.
RF	random forest.
RMSE	root mean squared error.
RSS	residual sum of squares.
SDMT	seismic diameter.
SFLA	shuffled frog leaping algorithm.
SHANSEP	stress history and normalized soil engineering properties.
SPT	standard penetration test.
SSA	salp swarm algorithm.
SVM	support vector machine.

SVR	support vector regression.
UC	unconsolidated compression tests.
UCS	unconfined compressive strength.
UU	unconsolidated undrained triaxial tests.
VT	vane test.
WDO	wind-driven optimization.

Chapter 1

Introduction

1.1 Background

As a broad field of engineering covering many specialities, civil engineering deals with numerous aspects of our life, including buildings that offer human beings comfortable places to work and live in, bridges providing passage over obstacles, roads providing the infrastructure for the transport of people and goods, etc.

The study of underground problems led to geotechnical engineering, a sub-discipline of civil engineering. For geotechnical engineers, the soil has always been playing an essential role in the construction projects, such as foundations, earthen dams, embankments, excavations, retaining walls, etc. Foundations are able to transfer the load that the structure bears to the ground. Earthen dams withstand the pressure from the water, preventing flooding of the adjacent area. And embankments raise structures above flooding level. Retaining walls are solid walls supporting lateral soil thus keeping the soil behind it from sliding. In all those examples, soil mechanics plays a key role in site construction and avoiding failures. However, unlike steel or concrete, whose material properties are relatively clear and determined, the material properties and behaviour of soil are hard to predict due to its variability and limitation of the investigation. Moreover, the strength of the soil is nonlinear, that is to say, it is stress-dependent. The volume of soil also changes with the application of shear stress, which is called dilatancy, making studying soil mechanics more challenging (Mitchell et al., 2005). Being a branch of soil physics and applied mechanics, soil mechanics is concerned with the investigation of the behaviour and application of soil as materials for construction (Ly and Pham, 2020). Different from fluid mechanics and solid mechanics, soils consist of a heterogeneous mixture of air, water and particles, organic solids and other matter. The particles are usually clay, silt, sand, and gravel. This multi-phase composition of soil makes the engineering properties of soil unique and hard to predict (Terzaghi et al., 1996).

Shear strength has always been one of the most important parameters in soil mechanics studies. It is defined as the capability of soils to withstand internal movement or slippage when subjected to an imposed load. This shearing resistance induced in a soil mass is composed of two types of friction, namely the sliding friction, also called the angle of shearing resistance and glue friction, which is provided by the property of soil named

cohesion. The angle of internal friction is affected by many factors such as dry density, water content, particle size distribution, the shape of particles, and surface texture. Cohesion also depends upon the types of clay minerals, the proportion of the clay, the size of clayey particles, and the valence bond between the particles (Kiran et al., 2016). Shear strength is widely employed in the design phase of many large-scale infrastructure projects including foundations, embankments, earthen dams, roads, pavements, excavations, slopes and retaining walls, etc (Vanapalli et al., 1996). For instance, in the design of foundations, the evaluation of bearing capacity is dependent on the shear strength. As for the design of embankments for dams, roads, pavements, excavations, levees etc., the analysis of the stability of the slope is done using shear strength. In the design of earth retaining structures like retaining walls, sheet piles, coffer dams, bulks heads, and other underground structures etc., shear strength also plays a crucial role.

In general, shear strength parameters can be estimated, both in the field as well as in the laboratory. The in-situ tests include standard penetration test, cone penetration test, piezo-cone, field vane shear test and pressure meter reading test. They all have their own strengths and weaknesses (Kiran et al., 2016). In the laboratory, the parameters of soil shear strength are generally determined through three experiments: direct shear test, triaxial shear test, and unconfined compression test with three other experimental diagrams according to the working conditions of the soil unconsolidated undrained triaxial tests (UU), consolidated undrained triaxial tests (CU) and consolidated drained triaxial tests (CD) (Craig, 2004).

As machine learning (ML) techniques have become increasingly popular, there have been an increasing number of applications of ML in diverse areas of science, especially in the last decade. In the context of soil research, notably in pedometrics, statistical models have been used to infer how soil is distributed both in space and time and try to grasp the theory behind it (McBratney et al., 2019). The availability of an increasing number of large datasets of soil together with the open-sources ML techniques have contributed to the increasing use of ML techniques in soil studies, such as inferring the classification of soil types or prediction of soil properties via soil data using ML techniques (Padarian et al., 2020). It is worth mentioning that it is large for the soil mechanics community. For the ML community, it is still considered a very small dataset.

1.2 Research objectives

cone penetration test (CPT) is a powerful and cost-effective tool for the investigation of subsoil conditions, and various empirical correlations are available for interpreting CPT data. However, these correlations are not universally applicable to all soils and subsurface conditions. Therefore, 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 be not only more complex and tedious but also a more expensive task, compared with CPT. Instead, ML models based on, for example, random forest (RF) or artificial neural network (ANN) algorithms can be used, which makes the task much more efficient and economical. Thus, the motivation of this thesis is to review and investigate the relative performance of a range of ML algorithms for predicting the undrained shear strength through CPT data. The outcome of this thesis can provide a reference for selecting an effective ML algorithm to predict

soil undrained shear strength through CPT data. In order to fulfil this motivation, seven detailed objectives are as follows:

- Analyzing and preprocessing a worldwide CPT dataset, which will be used for training and validation of the ML models;
- Training the ML models on the worldwide CPT dataset, choosing an appropriate cross-validation (CV) strategy for tuning the hyperparameters of the ML models in the validation process;
- Processing another CPT dataset together with the corresponding laboratory results given by de Gast (2020), which will be used as a testing set;
- Evaluating and comparing the prediction results in the training and testing dataset and the CV results in the training dataset with statistical metrics to select an effective ML algorithm to predict the soil undrained shear strength through CPT;
- Conducting Monte Carlo analysis for further validation of the chosen algorithms;
- Conducting sensitivity analysis to investigate the relative importance of the input parameters on the output undrained shear strength;
- Discussing the limitations of the study, proposing an improvement plan and making recommendations for future studies.

1.3 Thesis outline

This thesis consists of 9 chapters in total, constructed in a logical order of how the models are trained, validated and tested.

Chapter 1 introduces the background of the topic of the thesis, starting with the importance of civil engineering and geotechnical engineering. Followed by an introduction of one of the most important parameters in soil mechanics studies, the shear strength, in terms of the definitions, significance and measuring methods. Lastly, the applications of ML techniques to soil research are briefly put forward.

Chapter 2 reviews the previous research about undrained shear strength. Conventional methods are firstly presented, including experimental studies using direct shear tests, triaxial tests and in-situ tests, etc. Followed by some analytical analysis. Next, various ML methods for predictions of material properties are presented and summarized. Lastly, novel ML technique applications are briefly introduced.

Chapter 3 firstly presents the training dataset. Then the input variables from CPT and the output variable from the laboratory test are illustrated. After that, the dataset is preprocessed. Lastly, the dataset is analyzed using descriptive statistics and a pairplot.

Chapter 4 renders a brief but comprehensive introduction of the machine learning techniques implemented in this study. The basic machine learning background is presented at the beginning in order to clarify all the terms related to machine learning in the study. Then five machine learning algorithms implemented in this research, namely the artificial neural network (ANN), support vector machine (SVM), Gaussian process regression

(GPR), random forest (RF) and XGBoost, are illustrated. To get an intuition of how they work, some simple examples are provided in the Appendix B.

Chapter 5 applies five machine learning models with five different algorithms to the training dataset. Then the hyperparameters in each model are calibrated with various cross-validation techniques using grid search CV or random search CV. After constructing the machine learning models, the Monte Carlo analysis is conducted for further validation.

Chapter 6 compiles the CPT and laboratory test data provided by de Gast (2020) to form the testing dataset. CPT data is interpreted and processed using GPR to form the input variables in the testing dataset. Laboratory test data is processed to form the output variable in the testing dataset.

Chapter 7 displays the calibrated machine learning models, the training and testing results for the machine learning methods, the Monte Carlo Analysis results and the sensitivity analysis results.

Chapter 8 summarizes the final conclusions.

Chapter 9 discusses the limitations of this study. Correspondingly, possible improvements are then given. Finally, recommendations for future investigation are brought out.

Chapter 2

Literature review

2.1 Introduction

This chapter first provides an overview of the previous related research on soil shear strength using conventional methods, including the direct shear test methods, triaxial shear test methods, in-situ test methods and analytical methods. Then a detailed review of previous research about applying various machine learning techniques, such as RF, ANN, SVM, GPR, etc. for the prediction of soil properties is provided. Lastly, a concise review of the recent development of the applications of novel machine learning techniques in soil research is presented.

2.2 Conventional methods

2.2.1 Direct shear test methods

Gan et al. (1988) modified a conventional direct shear apparatus so that the axis-translation technique for direct shear tests can be implemented on unsaturated soil, which makes it possible for the soil to be subjected to a wide range of matric suctions. Multistage direct shear tests were performed on both saturated and unsaturated specimens of a compacted glacial till in the study. Nonlinearity in the failure envelope was then discovered concerning different matric suctions. Accordingly, instructions on how to handle nonlinearity properly from a practical engineering standpoint were made. To study the shear strength of unsaturated loess, which is of great importance to the stability analysis of soil slope and foundation and calculation of earth pressure, Tian et al. (2021) carried out extensive direct shear tests of unsaturated loess. The intact and remoulded loess, together with sandy silt, quartz flour, and quartz sand for comparison were applied to direct shear tests in undrained conditions under different water content, dry density, and clay content. The test results indicate that both cohesion and the internal friction angle of loess piecewise functionally decrease with the increase of water content. The shear strength shows positive linear correlations with the dry density and the internal friction angle shows an upward quadratic function relation with the increase of clay content. Accordingly, the equation of shear strength of unsaturated loess was proposed for practical engineering uses. Being progressively applied in geotechnical engineering, geosynthetics are capable of conducting multiple tasks due to their excellent material properties. Possessing rather

high tensile strength, leading to an increase in the shear strength of soil, geotextiles are a particular type of geosynthetics. Shear box tests were carried out by Zhu and Anderson (1998) on samples containing various types of geotextiles to evaluate the variation in sand shear strength properties.

2.2.2 Triaxial test methods

The triaxial tests (Fig. 2.1) have always been one of the most commonly used experimental methods for the determination of soil shear strength. Sridharan et al. (1971) conducted isotropically consolidated undrained tests with pore water pressure measurements to investigate the shear strength characteristics of saturated, remoulded, montmorillonite and Kaolinite clays. it is discussed in the study that the shear strength is influenced by the soil structure, whose changes are generally associated with the changes in initial moulding water content, stress history and type of cation and its concentration in the electrolyte system. The test results indicate that a definite cohesion intercept exists even for saturated normally consolidated clays, under certain conditions. Markou and Droudakis (2013) implemented both single and multi-stage unconsolidated–undrained triaxial compression tests to investigate the shear strength of microfine cement grouted sands. Three different kinds of microfine cement were obtained by pulverising ordinary cement produced in Greece. It was observed that the character of the grouted sands conformed to the Mohr-Coulomb failure criterion and grouting with microfine cement suspensions proved to be able to increase the shear strength dramatically by adding cohesion.

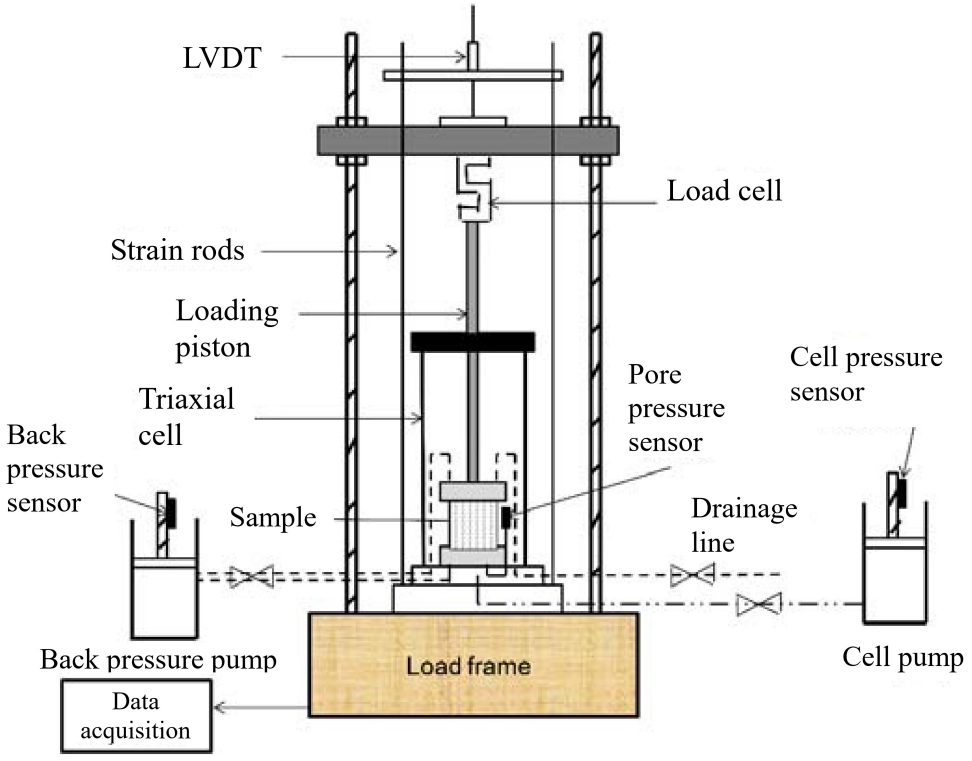


Figure 2.1. Schematic diagram of the triaxial testing system (Omar and Sadrekarimi, 2014).

Conventional triaxial tests, however, when applied on peats, have been strongly criticised for obtaining excessive-high shear strength parameters from standard data elaboration. The exaggerated shear strength parameters will lead to unrealistic factors of safety when applied in geotechnical design and evaluation which can raise a serious problem. To overcome this difficulty, numerous operational approaches have been given in many pieces of literature, nonetheless, they show a lack of consistency in mechanical background. For instance, the non-uniform stress and strain states developing in the samples resulting from end restraint effects, well before failure is reached, is one of the known issues related to the evaluation of peats from triaxial tests. Muraro and Jommi (2021) implemented undrained triaxial compression tests on reconstituted peat to examine the end restraint effects on the deviatoric stress, excess pore pressure and deviatoric strain response. In the study, the samples were tested both with the standard rough end platens and modified platens so that the friction between the samples and bottom and top caps could be reduced. They implemented four different initial height-to-diameter ratios to reduce the effects of rough end platens on the sample response. The testing results demonstrated that end restraint drastically contributes to the overestimation of the undrained shear strength of peat, as a result of the increase in both the calculated deviatoric stress and the measured excess pore pressure at the bottom of the sample. In this way, suggestions were then given to evaluate the impact that the end restraint effects have on the interpretation of laboratory results.

In addition, adequate approaches to obtain the undrained shear strength parameters with higher reliability were also proposed. In general, the shear strength of soils can be determined through laboratory direct shear tests and triaxial tests. Different testing methods, however, may probably result in large sets of scattered shear strength data, which makes it difficult to select the appropriate design parameters. Xu et al. (2018) carried out extensive laboratory tests to investigate the shear strength parameters of the compacted clay, which is needed for the safety design and stability analyses in the McArthur River Mine. Single-stage and multi-stage direct shear tests were accomplished under both dry and wet conditions under several different normal stresses. Different shearing rates were also applied to investigate the impact they have on the shear strength result. Additionally, several unconsolidated compression tests (UC), unconsolidated undrained triaxial tests (UU) and consolidated undrained triaxial tests (CU) were implemented under different equivalent effective confining pressure. A more reliable approach to the determination of peak and ultimate shear strength parameters of compacted clay in the design phase was given at length.

2.2.3 In-situ test methods

Moreover, shear strength parameters can also be estimated from in-situ tests, for instance, the cone penetration test (Fig. 2.2). Penetrometers are not only able to measure the tip resistance, but also the induced pore pressure. This enabled a new approach for the interpretation of soft soil undrained shear strength proposed by Konrad and Law (1987), considering the measured pore pressure. In this research, being one of the components of the tip resistance, the shear strength is assumed to be purely associated with the ultimate cavity expansion pressure. And the other component is calculated assuming that the effective friction is developed at the cone-soil interface. To validate the proposed model, tests were conducted at three different sites where the characteristics of soft sensitive

clay, stiff sensitive clay, and clayey silt. The outcome of the proposed model was in accordance with the known soil behaviours at those three sites. Młynarek et al. (2012) investigated the geotechnical parameters of alluvial soil represented by silts found near Poznań and Elbląg by applying the Piezo Cone Penetration Test (CPT), seismic diameter (SDMT) method and the vane test (VT). An analysis of the overconsolidation process was provided together with an equation reflecting the relationships between the undrained shear strength, the plasticity of the silts analyzed and the over-consolidation ratio (OCR) value, establishing a solid foundation for engineering projects on the grounds tested. Based on various tests, a conclusion was drawn that the most appropriate shear strength parameters can be achieved when the silt is compacted below the optimum water content.

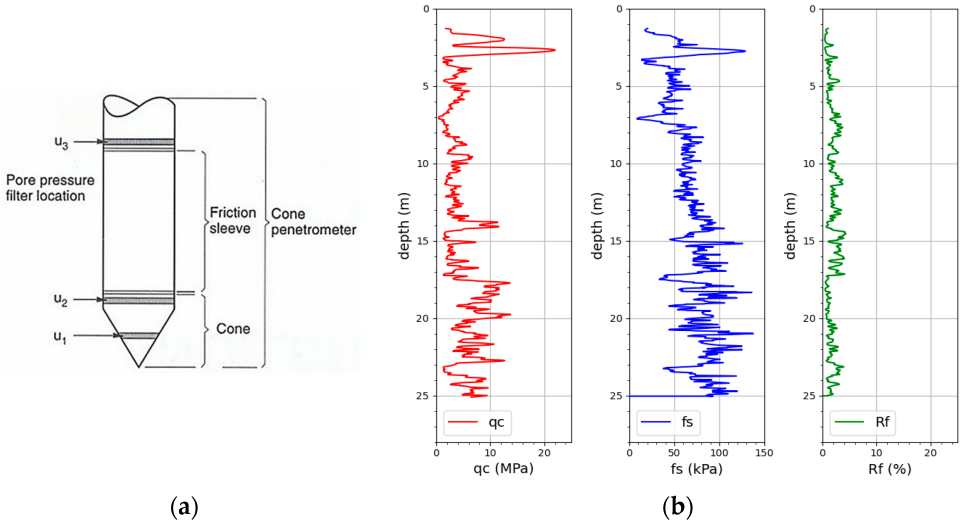


Figure 2.2. (a) Scheme of the CPT probe, which is pushed into the subsoil (Wang et al., 2021) (b) Example of measured data from the cone penetration test (cone resistance q_c, sleeve friction f_s and the friction ratio R_f) (Rauter and Tschuchnigg, 2021).

2.2.4 Analytical methods

Various analytical analysis has also been applied in soil shear strength research. Traditionally, the determination of shear strength parameters through CPT data is based on bearing capacity and cavity expansion theories. Motaghedi and Eslami (2014) proposed a new analytical approach Eq. (2.1) for shear strength prediction using quantities, q_c, u_2, and f_s from CPT taking into account the bearing capacity mechanism of failure at cone tip and direct shear failure along the penetrometer sleeve. Using all three outputs from CPTu, this new approach is considered more accurate in the case of erroneous data. To validate the advancement of this new approach, two sets of nonlinear equations proposed by this approach together with the existing correlations of C and ϕ angle parameters were applied to a database compiled from six sources. The results were compared with the corresponding laboratory tests. The internal friction angle obtained by existing correlations was comparably higher than the value measured by the laboratory test. Also, the predicted C and ϕ angle parameters from the proposed approach were in accordance with the measured values, indicating the effectiveness of the proposed approach for optimizing the design for geotechnical engineering issues.

$$\begin{cases} u_2 + \gamma B \tan\phi + qN_{\mathrm{q}} + \gamma B N_{\mathrm{q}} \tan^2\phi + C\left(\frac{N_{\mathrm{q}}-1}{\tan\phi}\right) = q_{\mathrm{t}} \\ C + 7.89\times10^{-4}(1-\sin\phi)\sigma'_{\mathrm{v0}}\tan\left(\frac{2}{3}\phi\right)\left[\frac{q_{\mathrm{t}}-\left(\frac{\sigma_{\mathrm{v}}-2\sigma_{\mathrm{h}}}{3}\right)}{\left(\frac{\sigma'_{\mathrm{v}}-2\sigma'_{\mathrm{h}}}{3}\right)}\right]^{1.44} = f_{\mathrm{s}}. \end{cases} \tag{2.1}$$

where u_2 is the pore pressure; γ is the effective soil unit weight; B is the penetrometer diameter (35.7mm in this study); ϕ is the friction angle; q is the effective stress; N_q is the bearing capacity factor; C is the cohesion; q_t is the total cone tip resistance; σ_{v} and σ'_{v} are the total vertical stress and effective vertical stress respectively; σ_{h} and σ'_{h} are the total horizontal stress and effective horizontal stress respectively; f_s is the sleeve friction.

Dirgėlienė et al. (2017) conducted experimental and numerical analysis on the direct shear test. In different laboratory tests, the soil was loaded in a different way under constant vertical stress and constant sample volume. This had an impact on the stress-strain distribution and was validated by the finite-element method, in which the stress and strain in the sample during the direct shear test exhibited non-uniformity. Knowing a more accurate distribution of stress and strain in the sample, the soil shear strength parameters can be determined more precisely. Ahmadi Naghadeh and Toker (2019) proposed an exponential equation to predict the nonlinear variation of shear strength with matric suction for unsaturated soils. The proposed equation involved two shear strength parameters and the maximum capillary cohesion. It was validated with a series of constant-suction consolidated drained triaxial tests on samples reconstituted by isotropic consolidation from the slurry state. In addition, the proposed equation was applied to the test results of five other soils of low-suction range, and it ended up with a better prediction than the other six proposed shear strength equations, further proving the validity of the equation. Analytical approaches have also been applied to conduct uncertainty analysis related to soil shear strength. Knuuti and Länsivaara (2019) studied uncertainty originating from three different transformation models used for the evaluation of undrained shear strength of Finnish clays from CPTu borings. The results showed that the uncertainty for the net cone resistance and pore pressure transformation model was the lowest, indicating a promising practical use. Tian and Sheng (2020) developed Bayesian approaches for characterizing the probabilistic of undrained shear strength using CPT data and prior information. Illustrated using CPT data at a clay site in Shanghai, the Bayesian approach proved to be an effective tool for selecting a proper random field model for probabilistic characterization. D'Ignazio et al. (2021) combined the stress history and normalized soil engineering properties (SHANSEP) model and critical state soil mechanics (CSSM) model in order to investigate the uncertainties in modelling the undrained shear strength of clays. SHANSEP is an empirical model that describes the undrained shear strength of clays with normalized properties. The soil shear strength is considered a function of the OCR and two material coefficients that require empirical calibration. And the CSSM model is able to provide analytical solutions to define the undrained shear strength as a function of preconsolidation stress and the friction angle at the critical state. The effectiveness of this proposed hybrid model was then validated by comparing the prediction result of undrained shear strength with an existing multivariate database of field vane data points from Finland.

2.3 Machine learning methods

Having gone through imprecise physical processes during the formation process, geotechnical material has always exhibited diverse and uncertain behaviours. Modelling the behaviour of such materials is so complicated that conventional physical methods are usually incompetent in finishing such tasks. Having demonstrated superior predictive ability compared to traditional methods in soil mechanics, machine learning methods have been more and more popular and they have been widely used for predictions of material properties in the past decade for regression problems (Shahin, 2013; Shahin et al., 2009).

Starting with the soil shear strength related studies, Kanungo et al. (2014) assessed the effectiveness of ANN and regression tree (classification and regression trees (CART)) techniques in predicting the shear strength parameters. Using four different combinations of the six inputs, namely, gravel %, sand %, silt %, clay %, dry density, and plasticity index, four models were constructed and applied to predict the undrained shear strength in order to estimate the degree of effects that the inputs have on the output. Correlation coefficient and root mean squared error (RMSE) were used as the evaluation metrics. The results of four models using ANN are presented in Fig. 2.3. Comparing the results of all of these four models for the prediction of c and ϕ, it is easy to conclude that Model II, considering 5 input parameters such as gravel %, sand %, silt %, clay % and dry density, has given the highest accuracies with a 5/16/2 neural network. However, in the case of Model IV which considers all of the six input parameters, the difference between training and testing correlation coefficient (R) values obtained with a 6/2/2 neural network is the lowest compared with the other three models. The training and testing R values for this 6/2/6 neural network in Model IV are illustrated in Fig. 2.4. Taking into account the idea put forward by Sietsma and Dow (1991) that when faced with several neural networks with almost identical performance, the simplest one (i.e., the one that has the smallest number of weights and biases) will, on average, generalize best, Model IV is considered as the most appropriate model for the prediction of both two shear strength parameters in undrained conditions. As for the result for the regression tree, the most appropriate trees were also achieved with Model IV, using all 6 input parameters as inputs. These two regression trees for the prediction of cohesion and internal friction angle are presented in Fig. 2.5. To conclude, the effectiveness of both techniques on the prediction of friction angle turned out to be almost identical, while for the prediction of cohesion, ANN was superior to CART. In addition, in order to acquire the optimum weights and bias for the best neural network, Garson, and proposed Weight-bias approaches were carried out which were able to evaluate the influence of input variables on the output variables, and the shear strength parameters.

ANN architecture	*R*		*RMSE*	
	Training	Testing	Training	Testing
Model I (GP, SP, STP and CP as inputs)				
4/1/2	0.904788	0.709213	0.128307	0.202459
4/6/2	**0.950486**	**0.820566**	**0.094369**	**0.094369**
4/7/2	0.938912	0.73586	0.103917	0.203841
4/20/2	0.983056	0.723992	0.056938	0.229616
4/24/2	0.98938	0.709713	0.045527	0.226808
4/32/2	0.984646	0.724235	0.052902	0.215475
4/40/2	0.991034	0.58296	0.040481	0.27849
Model II (GP, SP, STP, CP and DD as inputs)				
5/1/2	0.894149	0.779981	0.134873	0.177807
5/3/2	0.894153	0.894153	0.134871	0.177657
5/6/2	0.933653	0.85501	0.108095	0.15316
5/16/2	**0.978625**	**0.915253**	**0.062174**	**0.121163**
5/19/2	0.993896	0.861629	0.033275	0.16182
5/32/2	0.995472	0.886153	0.02864	0.135251
5/40/2	0.890366	0.671108	0.282153	0.37535
Model III (GP, SP, STP, CP and PI as inputs)				
5/1/2	0.887056	0.735596	0.139076	0.192906
5/7/2	0.967071	0.773272	0.076868	0.201282
5/13/2	0.98884	0.800253	0.047095	0.19212
5/18/2	**0.988724**	**0.810958**	**0.045282**	**0.180246**
5/21/2	0.986577	0.804543	0.04952	0.19896
5/29/2	0.993524	0.779988	0.034499	0.20616
5/40/2	0.996953	0.639055	0.023531	0.2481
Model IV (GP, SP, STP, CP, DD and PI as inputs)				
6/1/2	0.900495	0.801914	0.130986	0.170389
6/2/2	**0.939658**	**0.879199**	**0.103081**	**0.136468**
6/5/2	0.974072	0.87192	0.069304	0.147565
6/7/2	0.98054	0.877744	0.059174	0.156894
6/15/2	0.975545	0.861556	0.066319	0.16628
6/22/2	0.996851	0.848852	0.023911	0.169109
6/40/2	0.999205	0.761987	0.012016	0.216977

Figure 2.3. Combined accuracies in terms of R and RMSE's for both the shear parameters (c and f) for all 4 models for some selected neural networks (Kanungo et al., 2014).

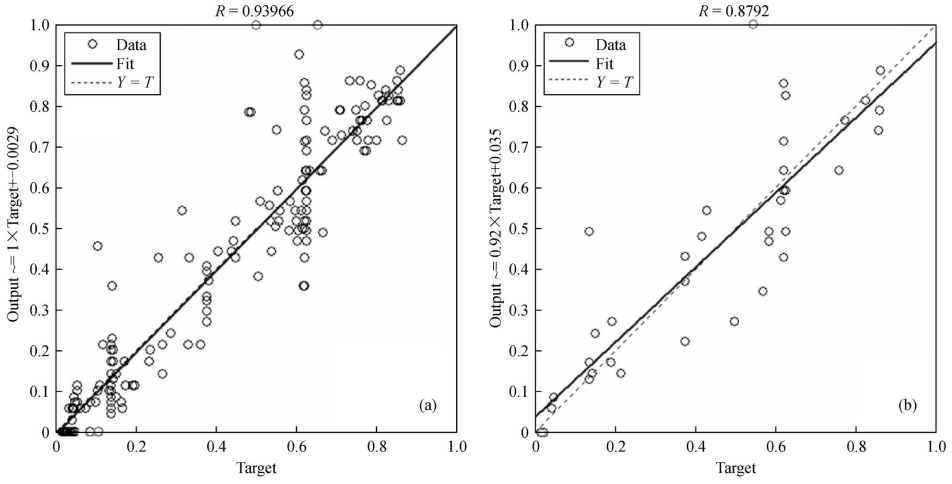


Figure 2.4. Correlation coefficients as obtained for Model IV for the 6/2/2 neural network: (a) training and (b) testing (Kanungo et al., 2014).

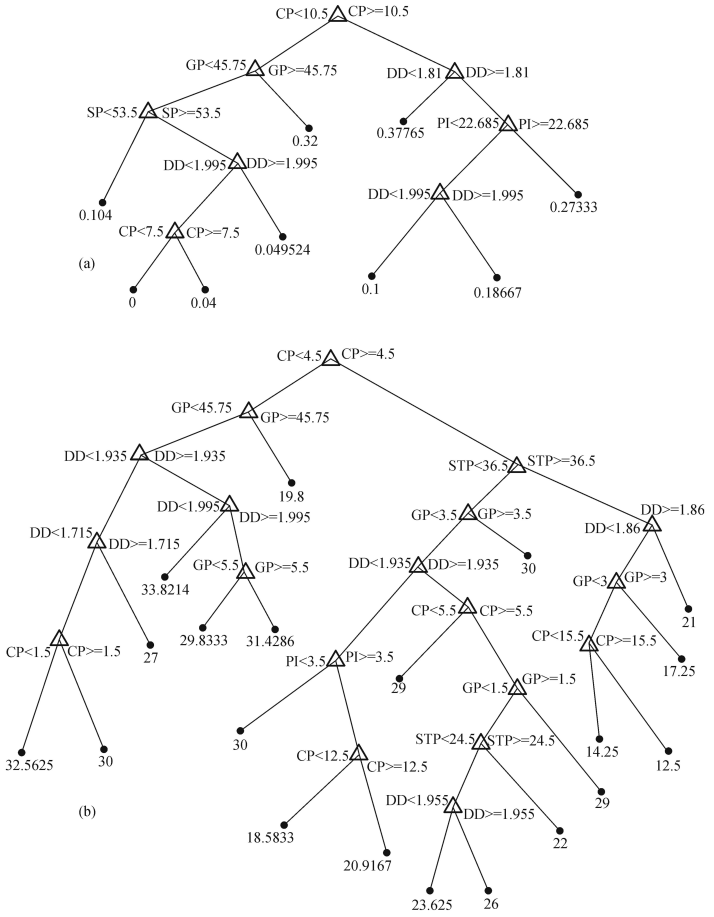


Figure 2.5. Most appropriate regression tree in the case of Model IV for predicting (a) cohesion and (b) angle of internal friction (Kanungo et al., 2014).

Using functional networks (FN), Khan et al. (2016) investigated the prediction of the residual strength of clay, which is one of the most important parameters considering the stability of slopes or landslides. The effectiveness of FN was compared with ANN and SVM through various statistical metrics, namely, the R, Nash–Sutcliffe coefficient of efficiency (E), absolute average error (AAE), maximum average error and RMSE. FN is superior to ANN, but inferior to SVM in terms of R and E. Iyeke et al. (2016) developed ANN model for the prediction of strength parameters of lateritic soils in central and southern areas of Delta State in order to save both time and money needed for acquiring geotechnical data during both design and construction phase. 83 soil samples were collected from various locations in the Delta State of Nigeria. The optimum ANN architecture for predicting the cohesion and friction angle turned out to be 3/9/1 and 3/11/1 respectively. Comparing the prediction results with that of some existing empirical correlations using the coefficient of determination and root mean square, the proposed ANN model proved to be the preferable one. Pham et al. (2020a) investigated the prediction of shear strength of soil under a different selection of input variables using the extreme learning machine (ELM) algorithm, which is an advanced ML technique. To evaluate the importance of the input variables to the prediction, feature backward elimination supported by Monte Carlo simulations was applied. 538 samples collected from the Long Phu 1 power plant project constituted the database and the R, RMSE, and MAE was

adopted as the statistical metrics. To find out the most relevant input variable, 30,000 simulations were conducted using an elimination process to select the most relevant variables. The result indicates that the performance of ELM is promising but it is sensitive to the input variable. The moisture content, liquid limit, and plastic limit were recognized as the most crucial inputs. Ly and Pham (2020) studied the prediction of shear strength of soil by applying SVM based on 6 input parameters, namely clay content, moisture content, specific gravity, void ratio, liquid limit and plastic limit. More than 500 samples were collected from the Long Phu 1 power plant project's technical reports and the performance of the proposed SVM model was evaluated using statistical metrics such as R, RMSE and MAE. The validation results were promising with the R ranging from 0.90 to 0.95. In addition, the most appropriate SVM model was used to investigate the shear strength prediction result by adopting partial dependence plot (PDP), as shown in Fig. 2.6, indicating that the moisture content, liquid limit and plastic limit were the three most important inputs to the prediction of soil shear strength. Kiran et al. (2016) also did similar studies but used probabilistic neural network (PNN) instead. By applying both statistical methods and ANN methods, Goktepe et al. (2008) compared the performance of these two methods in establishing correlations between index properties and soil shear strength parameters and the results indicated that ANN was the superior one. Das et al. (2011) described the prediction of the residual strength of soil based on index properties using ANN and SVM.

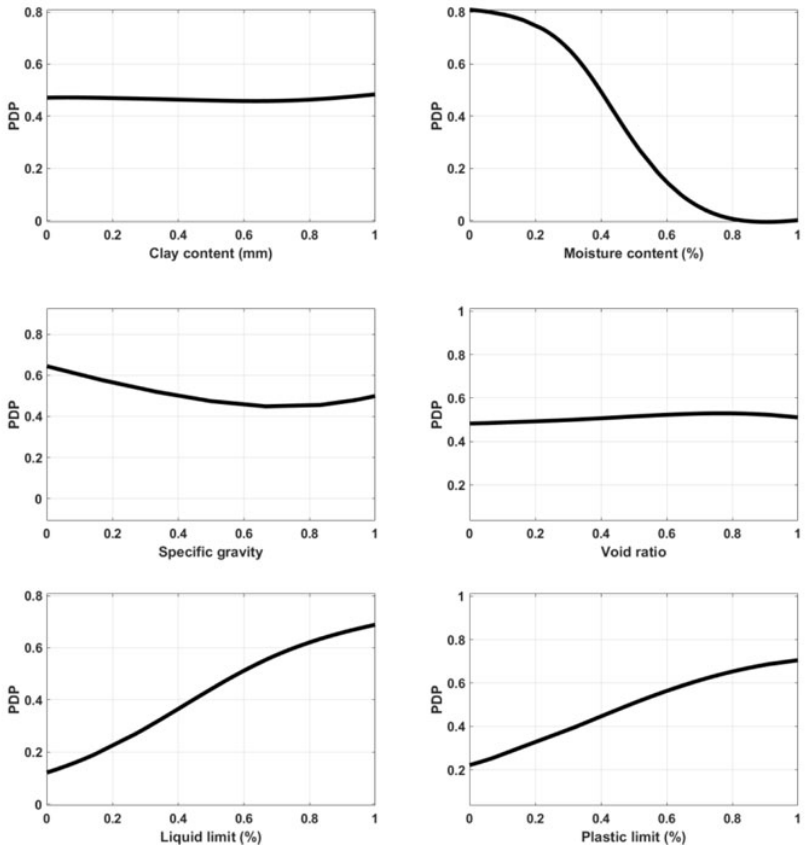


Figure 2.6. Partial dependence plots of the input variables used in this study (Ly and Pham, 2020).

Apart from the prediction of soil shear strength, many other material properties have also been investigated by applying ML techniques. Samui and Sitharam (2011) applied ANN and SVM to predict the liquefaction susceptibility of soil based on the standard penetration test (SPT) data from the 1999 Chi-Chi, Taiwan earthquake. An illustration of the support vectors is shown in Fig. 2.7. Samui (2008) explored the potential of SVM on the prediction of the friction capacity of driven piles in clay. Rigidly depending on the statistical learning theory, the SVM performed the regression technique by introducing an accuracy (ϵ) insensitive loss function, which defines an ϵ tube (Fig. 2.8). In this way, if the predicted point lies within the tube, the loss is zero, and nothing is added to the loss function; if the predicted point is outside the tube, the loss equals the absolute value of deviation minus the ϵ. More details of the SVM are illustrated in the Chapter 4. Comparing the results of SVM with ANN, SVM demonstrated an overall better performance than ANN, showing the potential of being a practical algorithm for the prediction of friction capacity of driven piles in clay.

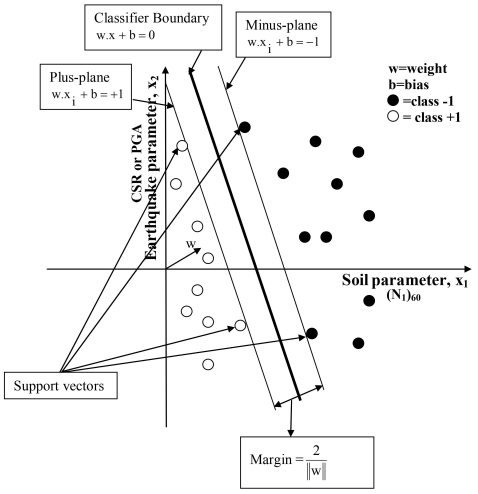


Figure 2.7. Support vectors with maximum margin (Samui and Sitharam, 2011).

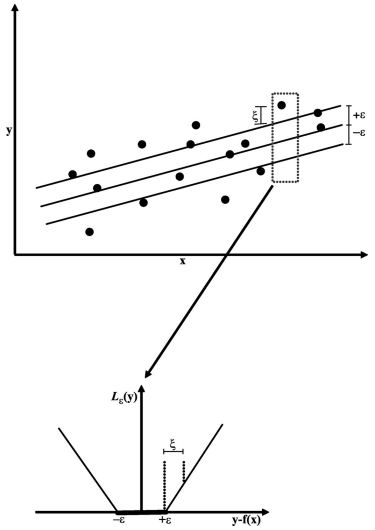


Figure 2.8. Prespecified accuracy ϵ and slack variable ξ in support vector regression (Schölkopf, 1997).

Ly and Thai Pham (2020) investigated the possibility of application of the RF algorithm to predict the soil's unconfined compressive strength (UCS), which is one of the most important mechanical properties of soils. Hoang et al. (2016) employed the GPR for modelling the compressive strength of high-performance concrete (HPC). Nonlinear functional mapping was established between the compressive strength and HPC ingredients with GPR. 239 HPC experimental tests were collected from an overpass construction project in Danang City (Vietnam) to build the dataset for the training and validation of the GPR model. Having the advantage of providing the uncertainty with respect to each predicted output, the GPR results are presented in Fig. 2.9 with an interval with a 95% level of confidence and were proved to be superior to those of the Least Squares SVM and the ANN. Similarly, Dao et al. (2020a) analyzed the performance of GPR, with five different kernels(Matern32, Matern52, Exponential, Squared Exponential, and Rational Quadratic) and ANN on the prediction of compressive strength of HPC by using Monte Carlo Simulation. As a result, Matern32 was chosen as the most appropriate kernel function. Dao et al. (2020b) optimized the structure of ANN in the prediction of compressive strength of foamed concrete, which is a promising material in civil engineering applications. Tsiaousi et al. (2018) adopted ANN techniques for the prediction of soil shear wave velocity based on approximately 300 raw CPTs and 11 seismic CPTs, from a large levees design project in the Netherlands. Puri et al. (2018) studied the prediction of several geotechnical parameters, such as the prediction of in-place density using SPT N-value, the prediction of compression index (C_c) using the liquid limit (LL) and void ratio (e), and the prediction of shear strength parameters cohesion (c) and angle of internal friction (ϕ) using SPT N-value, by applying ML techniques. Pham et al. (2020c) predicted the pile axial bearing capacity using ANN and RF. The results showed that RF outperformed ANN and the sensitivity analysis indicated that the average SPT value and the pile tip elevation were the most influential factors for the prediction of the axial bearing capacity of piles.

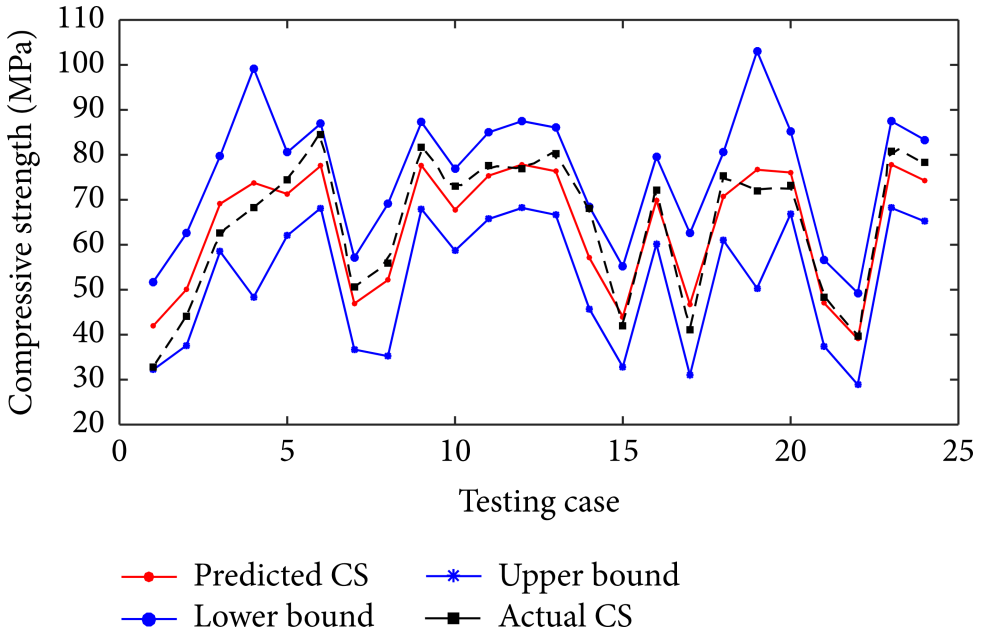


Figure 2.9. GPR prediction result with prediction interval (Hoang et al., 2016).

The articles mentioned in this section are summarized in Table 2.1. The summary consists of the source, the machine learning methods used, the inputs and the output(s) variables.

Source	Methodology	Inputs	Output(s)
Kanungo et al. (2014)	ANN; CART	gravel %; sand %; silt %; clay %; dry density; plasticity index	Undrained shear strength of soil
Khan et al. (2016)	FN	Liquid limit; plastic index; deviation from A-line; residual friction angle; clay friction	Residual strength of clay
Iyeke et al. (2016)	ANN	Plasticity index; percentage of particles passing sieve No.200; specific gravity; liquid limit; plastic limit	Shear strength of lateritic soils
Pham et al. (2020a)	ELM	Moisture content %; clay content %; void ratio; plastic limit %; liquid limit %; specific gravity	Undrained shear strength of soil
Ly and Pham (2020)	SVM	Clay content; moisture content; specific gravity; void ratio; liquid limit; plastic limit	Shear strength of soil
Kiran et al. (2016)	PNN	Water content; plastic index; dry density; gravel %; sand %; silt %; clay %	Shear strength of soil
Goktepe et al. (2008)	Statistical methods; ANN	Water content; plastic index	Shear strength of plastic clays
Das et al. (2011)	ANN; SVM	Liquid limit; plastic index; deviation from A-line; clay fraction	Residual strength of clay
Samui and Sitharam (2011)	ANN; SVM	corrected SPT value [(N1)60]; cyclic shear stress ratio (CSR)	Soil liquefaction susceptibility

Samui (2008)	SVM	Pile length; pile diameter; effective vertical overburden stress; undrained shear strength	Friction capacity of driven piles in clay
Ly and Thai Pham (2020)	RF	Clay content; moisture content; specific gravity; void ratio; liquid limit; plastic limit	Soil unconfined compressive strength
Hoang et al. (2016)	GPR	Cement; fine aggregate; small coarse aggregate; medium coarse aggregate; water; superplasticizer; concrete age	Compressive strength of high-performance concrete
Dao et al. (2020a)	GPR; ANN	Contents of cement; blast furnace slag; fly ash; water; superplasticizer; coarse aggregates; fine aggregates; concrete age	Compressive strength of high-performance concrete
Dao et al. (2020b)	ANN	Dry density; water/cement ratio; sand/cement ratio	Compressive strength of foamed concrete
Tsiaousi et al. (2018)	ANN	Cone tip resistance; sleeve friction; friction ratio versus elevation	Soil shear wave velocity
Puri et al. (2018)	Linear regression; ANN; SVM; RF; M5 model trees	SPT N-value; liquid limit; void ratio;	Inplace density; compression index; shear strength parameters
Pham et al. (2020c)	ANN; RF	Pile diameter; pile segments length; ground elevation; pile top elevation; pile tip elevation; average standard penetration test; etc.	Pile axial bearing capacity

Table 2.1. Summary of representative studies of predictions of material properties applying machine learning methods.

2.4 Recent development

The recent development of machine learning and optimization has resulted in some new promising soft computing methods i.e. particle swarm optimization - adaptive network-based fuzzy inference system (PANFIS), genetic algorithm - genetic adaptive neuro-fuzzy inference system (GANFIS). PANFIS and GANFIS are state-of-the-art methods that were formed by integrating meta-heuristic optimization algorithms and neural fuzzy models (Pham et al., 2018). In essence, the objective of using these optimization algorithms is to calibrate the hyperparameters in the machine learning models. Related recent research is briefly introduced in this section in consideration of the fact that this study addresses a relatively small dataset for which optimization algorithms are unnecessary.

Ding et al. (2021) innovatively combined an adaptive neuro-fuzzy inference system (ANFIS) model with an optimization technique i.e., Henry gas solubility optimization (HGSO) for solving a nonlinear and complex problem related to soil shear strength prediction. The new hybrid model is thus called HGSO-ANFIS. The HGSO optimization algorithm is based on the huddling behaviour of gas for the purpose of finding the global minima of the loss function in machine learning techniques and avoiding being trapped in the local minima. Taking the liquid limit, specific gravity, clay content, moisture content, void ratio, and plastic limit as the inputs for the prediction of shear strength, the proposed model was tested with real data and the result was compared with results of other ANFIS-based models. The comparison results showed that the new hybrid HGSO-ANFIS model outperformed the other ANFIS-based models, thus it can be applied for various prediction and optimization problems. Pham et al. (2020b) developed a novel hybrid soft computing model RF-PSO and used it to estimate the undrained shear strength of soil based on 6 inputs. Validation of the models indicated that the RF-PSO model is superior to the single RF model without optimization. Pham et al. (2018) investigated and compared the performance of four machine learning methods, PANFIS, GANFIS, support vector regression (SVR), and ANN, for predicting the strength of soft soils. And concluded that out of four models the PANFIS indicated a promising technique for the prediction of the strength of soft soils. Moayedi et al. (2020) combined four novel optimization algorithms, namely the elephant herding optimization (EHO), shuffled frog leaping algorithm (SFLA), salp swarm algorithm (SSA), and wind-driven optimization (WDO) with ANN to create four hybrid wise neural-metaheuristic paradigms in predicting soil shear strength. The results indicated that the proposed SSA-MLP model had the best performance out of the four models, and thus can be used as a replacement for traditional methods.

2.5 Conclusion

This chapter reviews the research of conventional methods, including laboratory tests, in-situ tests and analytical methods for the estimation of soil shear strength and the applications of both ordinary and novel machine learning methods on geotechnical problems.

For the prediction of soil shear strength, the disadvantages of conducting laboratory tests and in-situ tests are that they are more time-consuming, costly, and labour-intensive compared to applying machine learning methods. And the disadvantage of using analytical methods is that they might lack accuracy.

To develop a machine learning model for the prediction of soil shear strength, variables that are possibly highly correlated with soil shear strength should first be selected as the inputs for the model carefully, which will be discussed in the Chapter 3. Then different machine learning models should be trained and tested. Next, the performance of the proposed models on the prediction of soil shear strength should be evaluated with statistical metrics. After that, the

most adequate model is constructed and tested on an unseen dataset for further verification. There is no guarantee of the most appropriate machine learning model for a specific problem before applying the above process since it depends on a case-by-case analysis. In addition, novel optimization algorithms can be adopted for more accurate and effective calibration of the hyperparameters of the machine learning model when the training dataset is large.

Chapter 3

Training dataset

3.1 Introduction

To train the proposed machine learning models, establishing a training dataset is a prerequisite. The training dataset is fed to the machine learning algorithms so that the models are taught how to perform a certain task, in this study, which is making predictions of undrained shear strength.

The Clay/6/535 database is chosen to be the preliminary dataset in this study. It comprises 535 data points of lightly overconsolidated clay data from 40 sites with complete measurement of 6 parameters of interest in Ching et al. (2014)'s study, namely the normalized undrained shear strength ($\frac{Su}{sigv'}$), overconsolidation ratio (OCR), normalized cone tip resistance ($\frac{qt-sigv}{sigv'}$), normalized effective cone tip resistance ($\frac{q_t-u_2}{sigv'}$), normalized excess pore pressure ($\frac{u_2-u_0}{sigv'}$) and the pore pressure ratio ($\frac{u_2-u_0}{q_t-sigv}$), together with the effective stress ($sigv'$) and the depth of each measured point. These 40 sites are located in the following geographical regions: Brazil, Canada, Hong Kong, Italy, Malaysia, Norway, Singapore, Sweden, UK, USA, and Venezuela.

This chapter first selects the appropriate input variables in the training dataset for the machine learning models. Then the CPT test methods for obtaining these input variables are briefly introduced. Next, the output undrained shear strength is put forward, together with a short explanation of the laboratory tests used to obtain the undrained shear strengths in the training dataset. After that, several data preprocessing methods in terms of the machine learning application are illustrated, including handling null values, feature scaling, the spilt of data. Finally, the established training dataset is analyzed with descriptive statistics and a pairplot which is able to visualize the correlations between each variable in this study.

3.2 Input variables

To predict the undrained shear strength of soil, the input variables related to the undrained shear strength of soil should be selected and validated (Ly and Pham, 2020). To start with, the overconsolidation ratio (OCR) is not taken into consideration since it cannot be obtained with CPT while the objective of this study is to predict the undrained shear strength through CPT data. The pore pressure ratio ($\frac{u_2-u_0}{q_t-\sigma_v}$) is also discarded since it simply is the normalized excess pore pressure over the normalized cone tip resistance. In addition, the depth is not considered as the effect of depth on undrained shear strength is already captured by the features effective stress and pore pressure. Moreover, the inputs are not normalized by the effective stress since

there could be a risk of missing out on some important information, for instance, when there are higher-order relationships between the variables. Therefore, a total of four input variables are chosen in the prediction of the shear strength of soil, including: the effective stress (σ'_v), cone tip resistance ($q_t - \sigma_v$), effective cone tip resistance ($q_t - u_2$) and the excess pore pressure ($u_2 - u_0$). They are obtained from cone penetration test.

In a CPT, a standard probe with a conic tip is pushed vertically into the ground at a constant rate while the resistances are registered. Starting to be used in the 1930s in Denmark (Lunne et al., 2002), the test was initially performed using mechanical cones and only two parameters were able to be registered, which were the tip resistance q_c and sleeve friction f_s. Based on the measured data, various soil behaviour charts could then be developed to identify the soil strata and soil behaviour types (Robertson, 2009, 2010, 2016, 1990; Robertson et al., 1986). The intervals of the survey profile at the beginning, however, were relatively large, being up to tens of centimetres long owing to the design of mechanical cones (Stacul et al., 2020). Nowadays, with the advent of electrical cones, the intervals of the survey profile have been much smaller, typically around $0.5 - 2cm$. Inside the electric cone, various measurement and transmission systems have been adopted, i.e., measuring elements equipped with strain gauges or piezoelectric crystals and transmission systems equipped with electrical cables, radio waves, or even acoustic waves transmitted through the rods (Pieczyńska-Kozłowska et al., 2021). Among all these types of electric cones, the most frequently used type is the piezocone. Having the ability to quasi-continuously register three quantities, it is not only capable of registering the sleeve friction and tip resistance mentioned above, but also the pore pressure. A typical piezocone with a $36.0mm$ diameter, a $10cm^2$ base and $150cm^2$ external surface at the tube shape is presented in Fig. 3.1. The tests performed with piezocone probes are referred to as CPTu tests and the figure clearly shows the locations of the sensors for the three quantities. Furthermore, there are actually three locations to install the pore pressure sensor. The pore pressure filter presented in Fig. 3.1 is placed between the tip and the friction sleeve, and the pore pressure measured here is denoted as u_2. The other two possible locations for the pore pressure sensors have been shown in Fig. 2.2, one is in the middle of the tip height and the other is above the friction sleeve. The measurements are denoted as u_1 and u_3 respectively.

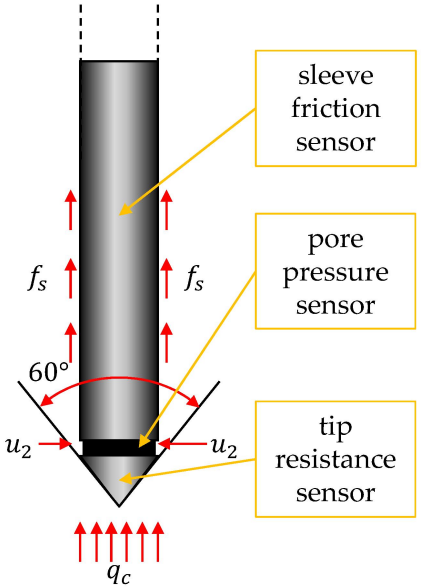


Figure 3.1. Schematic view of CPTu piezocone probe (Pieczyńska-Kozłowska et al., 2021).

In the Clay/6/535 database, all the q_c values measured in the CPTu tests have been converted into the corrected cone tip resistance q_t using Eq. (3.1) in consideration of the effect of pore pressure generated behind the cone.

$$q_t = q_c + u_2(1 - a) \tag{3.1}$$

where a is the net area ratio tip, usually ranging from 0.55 to 0.9 Lunne et al. (2002) depending on the probe design. Ideally, only data sources with complete documentation should be used. Nonetheless, perfect data sources are rare and all current statistical characterization studies in the literature involve making appropriate assumptions to ensure that the sample size is large enough to produce reasonably robust statistics. Therefore, for the 15 sites where the net area ratio tip a was not recorded, an assumption that $a = 0.7$ had been made. And for another 2 sites where the pore pressure u_2 was not recorded, the correlation equations given by Mayne et al. (1990) had been adopted to estimate u_2 based on q_c. All measured water pressures had been converted into u_2 in consideration that u_1 and u_3 are often damaged in practical. For the 9 sites where u_1 rather than u_2 was measured, the correlation equations given by Mayne et al. (1990) had been adopted to convert u_1 into u_2 (Ching et al., 2014). These can be regarded as data preprocessing for the raw CPT data in essence. The data preprocessing for the Clay/6/535 in this study will be described in Section 3.4.

3.3 Output variable

The shear strength of soil is considered an output variable. In geotechnical practice, the Mohr-Coulomb failure criterion is commonly used for determining soil shear strength. Generally, the failure criterion is based on the Eq. ((3.2)):

$$\tau_{\mathrm{f}} = \mathrm{c} + \sigma_{\mathrm{f}} \tan \phi \tag{3.2}$$

where c is the cohesion; ϕ is the angle of internal friction; σ_f is the normal stress on the failure plane; τ_f is the shear strength.

In saturated soils, however, the stress that soil particles receive is called effective stress while the stress that water receives is called pore water pressure. The shear stress in the soil can only be resisted by the skeleton of soil particles. Therefore, the shear strength of the soil must be expressed as a function of effective stress at the failure σ'_f and the shear strength parameters in the effective state (c' and ϕ') as shown in the Eq. (3.3) (Craig, 2004).

$$\tau_{\mathrm{f}} = \mathrm{c}' + \sigma'_{\mathrm{f}} \tan \phi' \tag{3.3}$$

To calculate the shear strength of soil (S_u), the parameters such as c and φ are often determined in the laboratory through the experiments mentioned in the introduction, then the shear strength of soil can be calculated using Eq. 3.2 or Eq. 3.3 with unit normal stress on the failure plane. In the Clay/6/535 database, the laboratory tests for obtaining the shear strength include unconsolidated undrained compression test, unconfined compression test, triaxial isotropically consolidated undrained compression test, triaxial K_0 consolidated undrained compression test, and field vane test (Ching et al., 2014). These S_u values obtained from different laboratory tests, however, cannot be compared directly since the S_u values are usually affected by a variety of factors, e.g., strain rate, stress state, sampling disturbance, etc. Therefore, they had all been converted into the equivalent CIUC values based on various empirical correlations (Ching et al., 2014; Kulhawy and Mayne, 1990; Chen and Kulhawy, 1993; Bjerrum, 1972).

3.4 Data preprocessing

After selecting the input variables for the machine learning model, the training dataset used for this research can be established. In order to generate the datasets for modelling, the shear strength of soil is considered as an output variable (Y) whereas the other four features namely the effective stress, cone tip resistance, effective cone tip resistance and excess pore pressure are considered as input variables X1, X2, X3 and X4 respectively. For each data sample, these 4 variables are used to predict the undrained shear strengths Y measured by the laboratory tests as stated above.

Data preprocessing is an integral step in ML as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we preprocess our data before feeding it into our model (Kumar, 2018). In the preprocessing step, the dataset is evaluated with regard to its completeness and amount of data (Rauter and Tschuchnigg, 2021).

3.4.1 Handling Null Values

In any real-world dataset, there are always a few null values. No matter it is a regression or classification or any other kind of problem, no model can handle these null values on its own so we need to intervene. There are various ways to handle this problem. The easiest way to solve this problem is by dropping the rows or columns that contain null values. However, it can result in significant information loss. If there are thousands of data points then removing 2–3 rows won't affect the dataset much but if there are only 100 data points and out of which 20 have null values for a particular field then you can't simply drop those rows. In this case, rather than dropping these values, we need to somehow substitute the missing values with the help of imputation, which is simply the process of substituting the missing values of our dataset.

In this study, there are in total 529 rows left in the training dataset left after moving out 6 rows with null values.

3.4.2 Feature scaling

Some datasets have multiple features spanning varying degrees of magnitude, range, and units. This is a significant obstacle as a few machine learning algorithms are highly sensitive to these features (Bhandari, 2020). Machine learning algorithms have been divided into the following three broad categories so as to address this issue:

1. Machine learning algorithms like linear regression, logistic regression, neural network, etc. are gradient descent based algorithms (Except for linear regression in its simplest form which uses least-squares). They use gradient descent as an optimization technique that requires data to be scaled. As an example, the formula of the gradient descent algorithm for linear regression is presented in the Eq. (3.4):

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta \left(x^{(i)} \right) - y^{(i)} \right) x_j^{(i)} \tag{3.4}$$

   where θ_j is the parameter in the cost function of the linear regression model that needs to be calibrated; α is the learning rate; m is the size of the training set; h_θ is a hypothesis; x is the model input; y is the true value.

The presence of feature value x in the formula will affect the step size of the gradient descent. The difference in ranges of features will cause different step sizes for each feature. To ensure that the gradient descent moves smoothly towards the minima and that the steps for gradient descent are updated at the same rate for all the features, we need to scale the data before feeding it to the model.

2. Algorithms like K-Nearest Neighbor (KNN), K-means, and SVM are distance-based algorithms. They are most affected by the range of features. This is because behind the scenes they are using distances between data points to determine their similarity. For example, if both the features have different scales, there is a chance that higher weightage is given to features with higher magnitude. This will impact the performance of the machine learning algorithm and obviously, we do not want our algorithm to be biased towards one feature. Therefore, we scale our data before employing a distance-based algorithm so that all the features contribute equally to the result.

3. Tree-based algorithms, on the other hand, are fairly insensitive to the scale of the features. A decision tree is only splitting a node based on a single feature. The decision tree splits a node on a feature that increases the homogeneity of the node (see Section 4.4.1). This split on a feature is not influenced by other features. So, there is virtually no effect of the remaining features on the split and this is why they are invariant to the scale of the features.

The followings are two feature scaling techniques, normalization and standardization.

3.4.2.1 Standardization

Standardization is a scaling technique where the values are centred around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation. Eq. (3.5) is the formula for normalization:

$$X' = \frac{X - \mu}{\sigma} \tag{3.5}$$

Here, μ is the mean of the feature values and σ is the standard deviation of the feature values.

Standardization, on the other hand, can be helpful in cases where the data follows a Gaussian distribution. However, this does not have to be necessarily true. Also, unlike normalization, standardization does not have a bounding range. So, even if there are outliers in the dataset, they will not be affected by standardization (Bhandari, 2020).

3.4.2.2 Normalization

Normalization is another scaling technique in which values are shifted and rescaled so that they end up ranging between the given range on the training set, i.e. between 0 and 1. It is also known as Min-Max scaling. Eq. (3.6) is the basic formula for normalization:

$$X' = m + (n - m) * \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{3.6}$$

Here, the data is rescaled to between m and n, X_{max} and X_{min} are the maximum and the minimum values of the feature respectively.

Normalization is good to use when it is known that the distribution of the data does not follow a Gaussian distribution. This can be useful in algorithms that do not assume any distribution of the data like KNN and neural networks (Bhandari, 2020).

In this study, the values in the training dataset are scaled to between -1 and 1 for the ANN, SVM and GPR algorithms since they are sensitive to feature scaling. While for the other two tree-based algorithms, RF and XGBoost, which are insensitive to feature scaling, the values in the training dataset are not scaled.

3.4.3 Split of the data

Splitting the dataset is essential for an unbiased evaluation of prediction performance. In most cases, it's enough to split the dataset randomly into three subsets (Pedregosa et al., 2011):

1. The training set is applied to train, or fit, the model. For example, the training set can be used to find the optimal weights, or coefficients, for linear regression, logistic regression, or neural networks.

2. The validation set is used for unbiased model evaluation during hyperparameter tuning. For example, to find out the optimal number of neurons in a neural network or the best kernel for a support vector machine, different values should experiment with. For each considered setting of hyperparameters, the model is fitted with the training set and its performance is assessed with the validation set.

3. The testing set is needed for an unbiased evaluation of the final model. The test datasets should never be used for training.

As the training dataset is too small in a machine learning context in this study, cross-validation strategies are applied to the training dataset (see Section 5.2). The ratio of training and validation of data is defined as 90/10. The testing dataset for this study is introduced in Chapter 6.

3.5 Analysis

After removing the 3 outliers in which the Y is too high, over 200 kPa, there are in total 526 data points left. These data points are compiled into the training dataset used in this research, as presented in the Appendix A.1. Descriptive statistics of this training dataset are shown in the form of Table 3.1.

As stated above, for the ANN, SVM and GPR algorithms in this study, the training dataset is normalized by the normalization transformation. With this estimator, each feature is scaled and translated individually such that they are all in the given range, between -1 and 1 in this research (see Section 3.4.2.2).

The correlation between input variables and output after feature scaling is then displayed in Fig. 3.2. According to this figure, the correlation between the variables can be discovered. It can be seen that the trends are fairly linear, showing good positive correlations.

Properties(kPa)	Symbol	Coding	Lowest	Highest	Mean	Median
Effective stress	σ'_v	X1	7.27	319.35	80.55	63.00
Cone tip resistance - total vertical stress	$q_t - \sigma_v$	X2	116.70	3707.05	623.39	476.74
Cone tip resistance - pore pressure behind cone	$q_t - u_2$	X3	13.92	1960.00	329.62	246.88
Excess pore pressure	$u_2 - u_0$	X4	9.71	2330.43	341.86	259.03
Undrained shear strength	S_u	Y	5.83	184.88	44.09	35.57

Table 3.1. Descriptive statistics of the data used in this study.

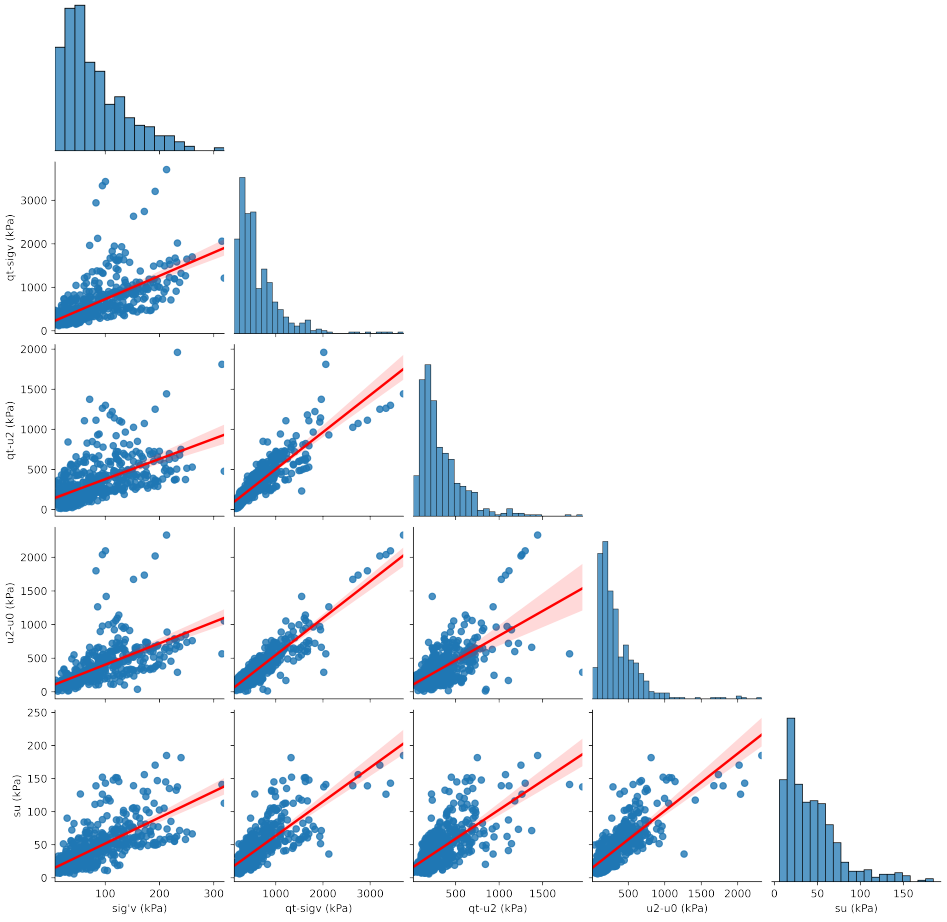


Figure 3.2. Correlation analysis between inputs and the output variables in this study.

3.6 Conclusion

Starting with the preliminary dataset Clay/6/535, the input variables for the training dataset are selected based on the following criteria: being able to be obtained through the CPT test and being correlated to the output undrained shear strength. Then the CPT test is briefly explained. Next, some transformations of raw CPT data through empirical correlations are described. Next, the output undrained shear strength is put forward together with a description of the laboratory tests used for obtaining it.

After that, the established training dataset is preprocessed considering the machine learning technique requirements. To start with, the null values in the training dataset need to be handled. 6 rows with null values were dropped out since it is comparably a small amount of data compared to in total 535 data samples. Then 3 outliers in which the shear strength is over 200 kPa are moved away and there are in total 526 data points left in the training dataset. Next, feature scaling is conducted with the training dataset to prepare for those machine learning algorithms that are sensitive to feature scaling. Next, cross-validation strategies are applied in the training dataset with the ratio of training and validation defined as 90/10. An unseen dataset will be the testing dataset will be presented in Chapter 6.

Finally, the training dataset is analyzed with descriptive statistics and a pairplot which shows good positive correlations between the inputs and output. In fact, this training dataset consisting of 526 data samples is considered as a relatively small dataset in a machine learning context. It is fed to the machine learning methods that will be introduced in Chapter 4.

Chapter 4

Methodology

4.1 Introduction

machine learning (ML) is a subset of artificial intelligence. With the help of historical data, which is also known as training data, ML algorithms construct a mathematical model to help make predictions or decisions without being explicitly programmed (Samuel, 1967). The performance of ML algorithms can be improved by providing more training data.

In this chapter, the ML techniques applied in this study are briefly introduced. Starting with the basic concepts in ML, supervised learning is put forward, which is a typical type of ML model approach. Then the validation metrics together with the gradient descent algorithm which is one of the optimization methods used for optimizing the validation metrics is shortly discussed. Next, the bias and variance trade-off which is an important property of ML models is presented.

Five machine learning algorithms used in this research are then introduced together with some simple examples for illustration as presented in the Appendix B. Using linear regression as a starter, three nonlinear regression models are presented, which are the artificial neural network (ANN), support vector machine (SVM) and Gaussian process regression (GPR). Among them, ANN can be regarded as a combination of linear regressions to obtain a more general-purpose regression. SVM and GPR are both kernel-function (a.k.a. covariance function)-based algorithms that are capable of dealing with nonlinear regression. GPR differs from SVM that it is also able to provide uncertainty estimates for its predictions. Finally, two tree-based algorithms, random forest (RF) and XGBoost, together with the ensemble methods are presented.

4.2 Basic concepts

4.2.1 Supervised learning

ML problems can generally be assigned to two main categories: supervised learning and unsupervised learning. In supervised learning, models are trained using a "labelled" dataset, which means that the input data in the training dataset is already tagged with the correct output. On the basis of this, a supervised machine learning algorithm aims to find a mapping function to transform the input variable into the output. After learning the information given in the training dataset, the model can then be tested on the testing dataset to give a prediction of the output. The process of supervised learning is illustrated with a simple example as presented in Fig. 4.1. In this example, the training dataset consists of different shapes, which have already been correctly labelled with square, triangle or hexagon. After the training of the model, it is

then tested on the testing dataset with the task of identifying the shape. The model is able to classify the new data it encounters in the testing dataset since it has already "learnt" how to do so in the training dataset.

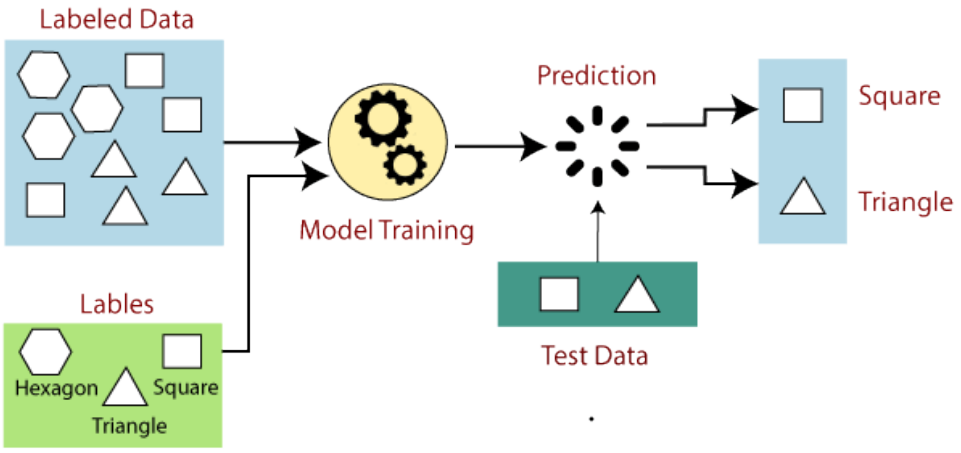


Figure 4.1. An example of supervised learning (Roy, 2019).

In a mathematical form, a typical training dataset of supervised learning can be presented as shown in Fig. 4.2. The training dataset is the collection of labelled examples. Each example contains several features, for instance, feature $x^{(1)}$ contains height (cm), feature $x^{(2)}$ contains weight (kg), feature $x^{(3)}$ contains gender and so on. These features together form a vector having m dimensions, which is called a feature vector and it is denoted as $x_{(j)}$. The term label is used to denote the output y_i which can be either an element belonging to a finite set of classes or a real number. In this way, supervised learning can deal with both classification and regression problems. This study applies supervised learning to solve a regression problem.

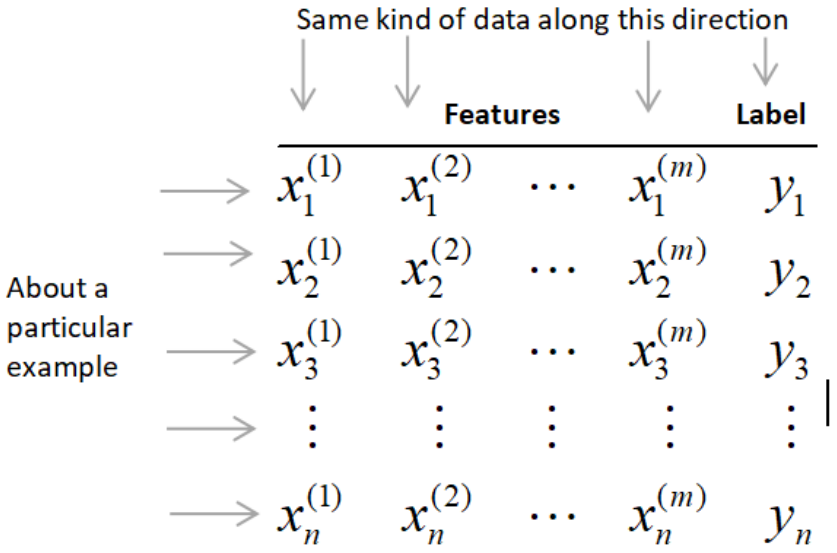


Figure 4.2. A typical training dataset of supervised learning (Roy, 2019).

In unsupervised learning, the model is trained on raw and unlabelled training data. Different from the supervised learning model, the unsupervised model has to find the hidden pattern and insights from the given data itself. As unsupervised learning is not adopted in this research, it's not further discussed.

4.2.2 Validation metrics and gradient descent

The validation metrics are able to quantify the difference between the predicted values of a ML model and the actual values, thus measuring how good the predictions are. They are used for two purposes: measuring the error for tuning the hyperparameters and evaluating the accuracy of the prediction. Hyperparameters refer to the parameters in the ML models that need to be set before the learning process begins. Three commonly used basic validation metrics for regression problems are introduced as follows:

The equation for the mean absolute error (MAE) is presented in (4.1). The smaller MAE is, the better the prediction is. It is the simplest validation metric which literally calculates the absolute difference (discards the sign) between the actual and predicted values and takes its mean. The advantage of MAE is that it is computationally inexpensive due to its simplicity. However, there are two main drawbacks of it. One of them is that it calculates all the errors on the same scale, making the alternation of the weights in the back-propagation algorithm (introduced in 4.3.1.2) difficult. The other is that its linearity may lead to a convergence problem in finding the minimum in the back-propagation algorithm (Hirekerur, 2020).

$$MAE = \frac{1}{N}\sum_{I=1}^{N}\left|Y_{i_{\text{observed}}} - Y_{i_{\text{predicted}}}\right| \tag{4.1}$$

where N is the number of samples; $Y_{i_{\text{observed}}}$ is the observed actual value; $Y_{i_{\text{predicted}}}$ is the prediction given by the ML models.

The equation for the root mean squared error (RMSE) is presented in (4.2). The smaller RMSE is, the better the prediction is. Being the square root of mean squared error (MSE), it is still a linear scoring method. Compared with MAE, it gives comparatively more penalization to larger errors since the errors are squared at the beginning, and therefore being more sensitive to the outliers (Hirekerur, 2020).

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left[\left(Y_{i_{\text{observed}}} - Y_{i_{\text{predicted}}}\right)\right]^2} \tag{4.2}$$

The equation for the coefficient of determination (R^2) is presented in (4.3). It ranges between 0 and 1. The closer it is to 1, the better the prediction is.

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(Y_{i_{\text{predicted}}} - Y_{i_{\text{observed}}}\right)^2}{\sum_{i=1}^{N}\left(Y_{i_{\text{observed}}} - \bar{Y}_{\text{observed}}\right)^2} \tag{4.3}$$

In this study, for the minimization of the validation metrics, only R^2 is applied. For the evaluation of the final prediction performance of the ML models, all these three validation metricss are used.

As discussed above, the values of the validation metrics reflect the performance of a ML model.

Taking RMSE as an example, the lower the RMSE is, the better the performance is. So ML model try to minimize the RMSE. This is achieved by applying various optimization algorithms. Gradient descent is one of the optimization algorithms to identify the optimal set of hyperparameters for optimizing the validation metrics. An illustration of the algorithm is presented in Fig. 4.3. It is an iterative optimization algorithm for finding the local minimum of a function, in this case, that is the validation metric in the ML. To find the local minimum of a function using gradient descent, two steps are taken iteratively. First, the gradient, which is the first order derivative of the validation metric at that point is computed. Second, move a step in the direction opposite to the gradient from the current point. The length of the step equals alpha times the gradient at that point. Alpha is called the learning rate, which is a hyperparameter in the optimization process that decides the length of the steps. If the learning rate is too high, we might overshoot the minima and keep bouncing, without reaching the minima. If the learning rate is too small, the training might turn out to be too long. Therefore it is crucial to choose an adequate learning rate is crucial. It is also worth noticing that the validation metric might consist of several minimum points. The optimization using the gradient descent algorithm may settle on any one of the minima, which depends on the initial point and the learning rate (M, 2020).

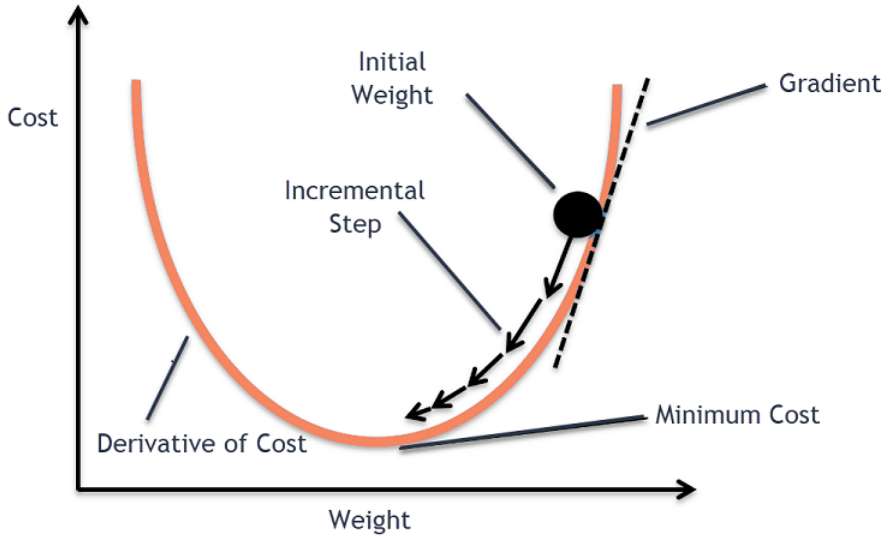


Figure 4.3. An illustration of the gradient descent algorithm (M, 2020).

4.2.3 Bias and variance trade-off

The bias and variance trade-off is the property of a ML model that the variance of the parameter estimated across samples can be reduced by increasing the bias in the estimated parameters (Kohavi et al., 1996). Bias is the difference between the average prediction of our model and the correct value that we are trying to predict. Variance is the variability of model prediction for a given data point. When tuning the hyperparameter, the conflict in trying to simultaneously minimize these two sources of error that prevent supervised learning algorithms from generalizing beyond their training set has to be properly dealt with (Von Luxburg and Schölkopf, 2011). An illustration of the bias and variance trade-off is shown in Fig. 4.4. It can be observed that the more complex a model is, the higher the variance becomes, the lower the bias is and the more the model is prone to overfitting. Conversely, the simpler a model is, the higher the bias becomes, the lower the variance is and the more the model is prone to underfitting.

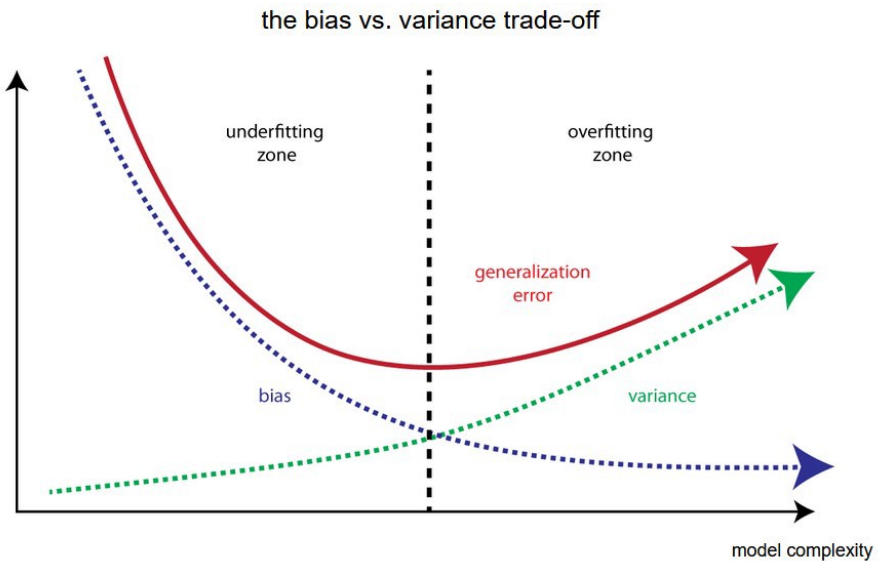


Figure 4.4. The bias and variance trade-off (Chakraborty, 2021).

4.3 Nonlinear regression

To start with, linear regression is the process of finding a line (in 2D cases) that best fits the data points available on the plot, so that we can use it to predict output values for inputs that are not present in the data set we have, with the belief that those outputs would fall on the line (Al-Masri, 2021). To introduce nonlinearity, different models apply various techniques. This section introduces three nonlinear regression algorithms, namely the ANN, SVM and GPR. Among them, ANN can be regarded as introducing nonlinearity through a combination of linear regressions. SVM and GPR are both kernel-function (a.k.a. covariance function)-based algorithms. GPR differs from SVM that it is also able to provide uncertainty estimates for its predictions.

4.3.1 Artificial Neural Network

In the following subsections, the architecture of ANN will first be briefly introduced. Then the detailed process of a feed-forward ANN using back-propagation algorithms is introduced and illustrated with a simple example as presented in the Appendix B.1.

4.3.1.1 Architecture of ANN

Artificial neural network (ANN) refers to a biologically inspired approach of ML modelled after the brain (Bhardwaj, 2022). Fig. 4.5 illustrates the typical diagram of a biological neural network (BNN). Similar to the human brain which has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. The first type of ANN historically, namely the perceptron is shown in Fig. 4.6. These neurons are known as nodes. Correspondingly, dendrites from BNN represent inputs in ANN, cell nucleus represents nodes, synapse represents weights, and axon represents output. 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). Due to the adaptability and learning capabilities, ANN is able to learn how complex non-linear systems perform when supplied with sufficient data. It can be used to perform both regression and classification analysis.

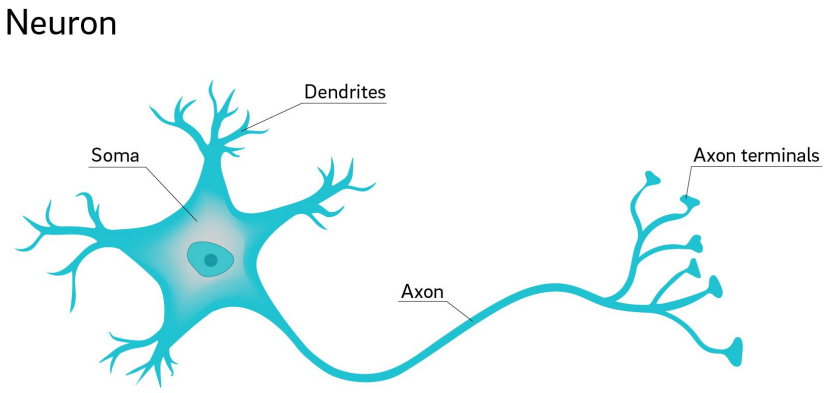


Figure 4.5. Typical diagram of a Biological Neural Network (Baillot, 2018).

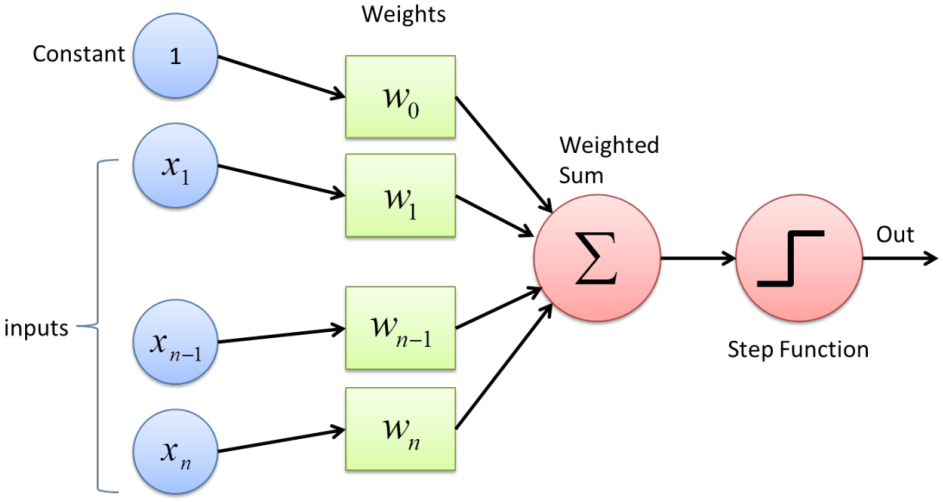


Figure 4.6. Typical diagram of an Artificial neural network (A perceptron using step function as activation function) (Qamar, 2020).

To get an overview of the architecture of an ANN, by far the most extensively used particular type of ANN namely multi-layer perceptron (MLP) is shown in the figure 4.7. Neural networks are typically arranged into an input layer, a hidden layer(s), and an output layer. The number of input and output nodes is the engineering problem in question. The number of hidden neurons is one of the hyperparameters that needs to be tuned on a problem-by-problem basis. To find the output of the neuron in ANN, firstly we take the weighted sum of all the inputs, weighted by the weights of the connections from the inputs to the neuron. Then, we add a bias term to this sum. This weighted sum is then passed through an activation function which is usually nonlinear to produce the output. More details will be explained in the following section. In this study, a feed-forward MLP using the back-propagation learning algorithm is applied.

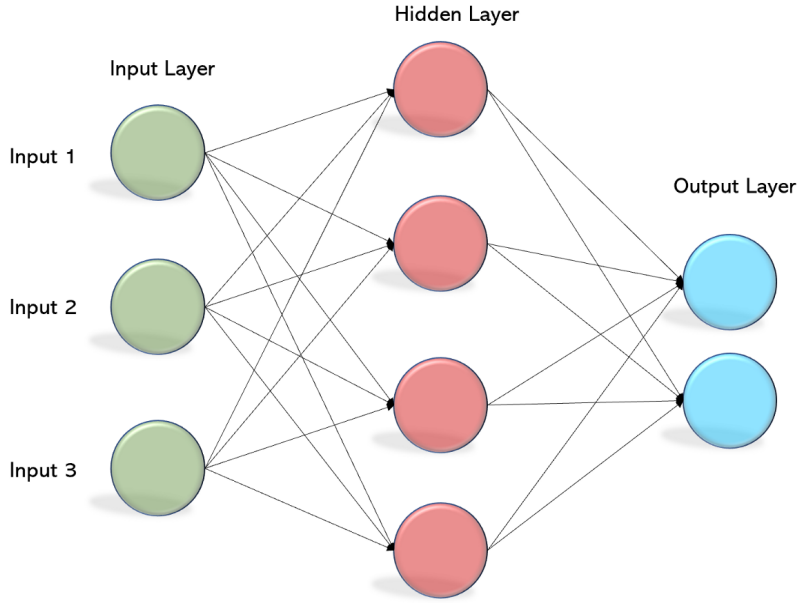


Figure 4.7. Typical architecture of an Artificial neural network (Multi-Layer Perceptron) (Mohanty, 2019).

4.3.1.2 Feed-forward MLP using back-propagation algorithms

Feed-forward Neural Networks are ANNs where the node connections do not form a cycle (Edgell, 2021). 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 some 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.

4.3.2 Support Vector Machine

Support Vector Machine (Figure. 4.8) 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 (ε) ε-insensitive loss function. There are three distinct characteristics when SVM is used to estimate the regression function: the type of kernel function, the optimum capacity factor C, and the optimum error insensitive zone ε. Consider a set of training data $\{(x_1, y_1), \ldots, (x_1, y_1)\}, x \in R^n, y \in r_1$, where x is the input, y is the output, R^N is the N-dimensional vector space and r is the one-dimensional vector space. The ε-insensitive loss function can be described as Eq. (4.4):

$$L_\varepsilon(y) = 0 \text{ for } |f(x) - y| < \varepsilon \text{ otherwise } L_\varepsilon(y) = |f(x) - y| - \varepsilon \tag{4.4}$$

This defines an ε tube such that if the predicted value is within the tube, the loss is zero, while if the predicted point is outside the tube, the loss is equal to the absolute value of the deviation minus ε. The main aim of SVM is to find a function f(x) that gives a deviation of ε from the

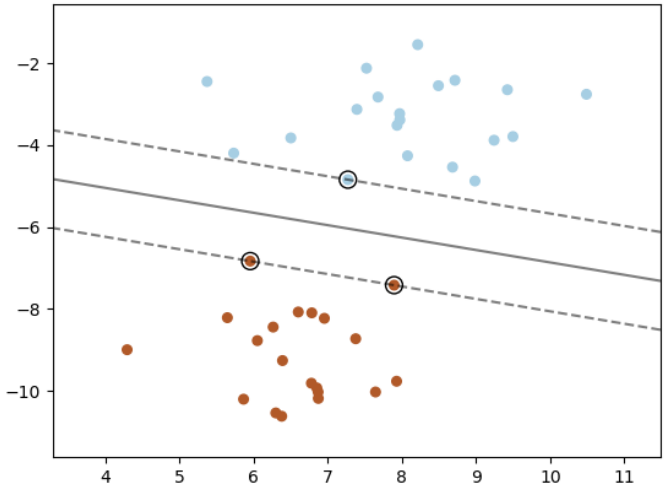


Figure 4.8. An example of a Support Vector Classifier using a linear kernel in 2D space (Pedregosa et al., 2011).

actual output and at the same time is as flat as possible. Assume a linear function f as Eq. (4.5):

$$f(x) = (w \cdot x) + b, w \in R^n, b \in r \tag{4.5}$$

where, w = an adjustable weight vector and b = the scalar threshold.
Flatness in Eq. (4.5) means a small value of w. One way of obtaining this is by minimizing the Euclidean norm $\|w\|^2$. This is equivalent to the following convex optimization problem:

Minimize: $\frac{1}{2}\|\mathrm{w}\|^2$
Subjected to:
$y_i - (\langle w \cdot x_i \rangle + b) \leq \varepsilon, i = 1, 2, \ldots, l$
$(\langle w \cdot x_i \rangle + b) - y_i \leq \varepsilon, i = 1, 2, \ldots, l$

The above convex optimization problem is feasible, sometimes, however, this may not be the case, we may also want to allow for some errors. The parameters ξ_i, ζ_i^* are slack variables that determine the degree to which samples with errors more than ε be penalized. In other words, any error smaller than ε does not require ξ_i or ξ_i^* and hence does not enter the objective function because these data points have a value of zero for the loss function. The slack variables (ξ_i, ξ_i^*) have been introduced to avoid infeasible constraints in the optimization problem below:
Minimize:
$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{1}(\xi_i + \xi_i^*)$
Subjected to:
$y_i - (\langle w \cdot x_i \rangle + b) \leq \varepsilon + \xi_i, i = 1, 2, \ldots, l$
$(\langle w \cdot x_i \rangle + b) - y_i \leq \varepsilon + \xi_i^*, i = 1, 2, \ldots, l$
$\xi_i \geq 0$ and $\xi_i^* \geq 0, i = 1, 2, \ldots, l$

The constant $C(0 < C < \infty)$ determines the trade-off between the flatness of f and the amount up to which deviations larger than ε 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). This process is illustrated in Fig. 4.9. The two steps in this exercise are, firstly, carrying out a

fixed nonlinear mapping of the data onto the feature space and, secondly, carrying out a linear regression in the high dimensional space. The input data are mapped onto the feature space by a map Φ. The dot product given by $\Phi(x_i) \cdot \Phi(x)$ is computed as a linear combination of the training points. The concept of a kernel function $[K(x_i, x) = \Phi(x_i) \cdot \Phi(x)]$ has been introduced to reduce the computational demand.

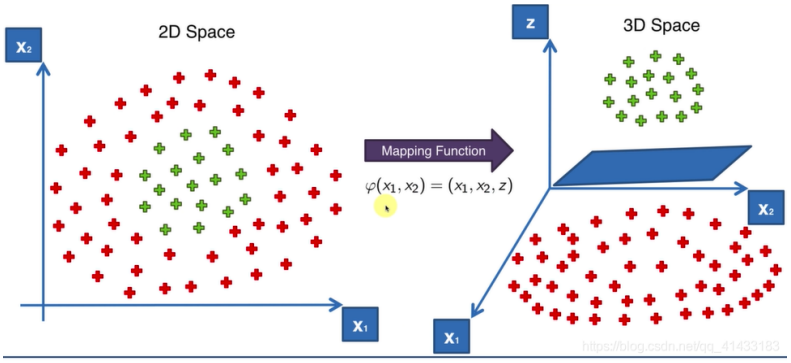


Figure 4.9. An illustration of mapping the input data from 2d to 3d through a kernel function (Saxena, 2020).

Some common kernels, such as homogeneous polynomial expressions, 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).

4.3.3 Gaussian Process Regression

Being a probabilistic supervised ML model, the Gaussian process model has been widely used for both regression and classification tasks. A Gaussian process regression (GPR) model can make predictions incorporating prior knowledge through kernel functions and provide uncertainty measures over predictions (Wang, 2020). Different from 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 it is 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 confident the predictions are. The process of conducting regressions by the Gaussian processes model is illustrated in Fig. 4.10: given the observed data (red points) and a mean function f (blue line) estimated by these observed data points, we make predictions at new points $\mathbf{X}_*$ as $\mathbf{f}(\mathbf{X}_*)$ (Wang, 2020). See Section 6.2 for an application of GPR.

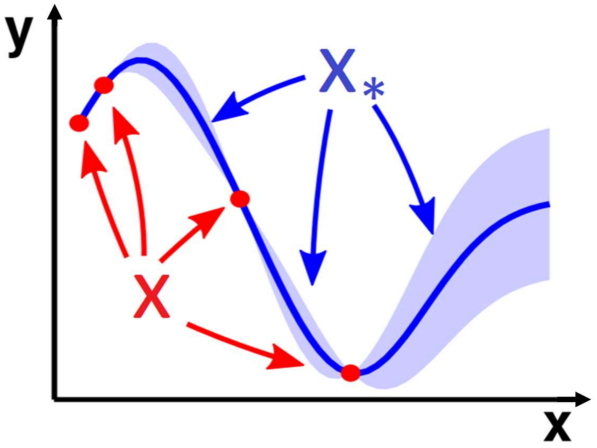


Figure 4.10. An illustrative process of conducting regressions by Gaussian processes. The red points are observed data, the blue line represents the mean function estimated by the observed data points, and the predictions will be made at new blue points (Wang, 2020).

4.4 Tree-based algorithms and ensemble methods

In the following subsections, the decision tree learning algorithm will first be explained. Then the ensemble learning methods will be introduced. Lastly, a brief introduction to RF and XGBoost algorithms will be provided. The detailed processes of RF and XGBoost algorithms are illustrated with simple examples in the Appendix B.2 and B.3 respectively.

4.4.1 Decision tree

Decision trees are a supervised learning approach, which can be applied to both regression and classification problems. In keeping with the tree analogy, decision trees implement a sequential decision process. Starting from the root node, a feature is evaluated and one of the two nodes (branches) is selected, each node in the tree is basically a decision rule. This procedure is repeated until a final leaf is reached, which normally represents the target (R, 2021). If a leaf node contains multiple samples from the training set, the average of the target variables of this leaf node is the output for any test sample that follows this decision path. Decision trees are also attractive models if we care about interpretability.

CART is one of the algorithms for creating decision trees. In this study, the Scikit-learn machine learning toolkit (Pedregosa et al., 2011) uses an optimised version of the CART algorithm. For classification problems, at each node, CART constructs binary trees using the feature and threshold that yield the largest information gain, which is a metric used to choose the attribute on which the data has to be split at the node. As for regression problems, the standard deviation is used as the metric. In other words, while the attribute with the highest information gain is chosen as the attribute to split the data in the case of classification, the attribute with the highest decrease in standard deviation is used in the case of regression (R, 2021). To get an intuition of the decision tree, a simple example is provided together with RF in the Appendix B.2.

4.4.2 Ensemble learning methods

Ensemble methods are techniques that create multiple base models and then combine them to improve the results. The underlying idea is that combining multiple models together often produces a much more powerful model. The term “model” here describes the output of the algorithm that is trained with data. And this model is then used for making predictions. This algorithm can be any machine learning algorithm such as logistic regression, decision tree, etc. These models, when used as inputs of ensemble methods, are called ”base models”. After the base models are selected, they need to be aggregated, for which, there are in general two main kinds of methods, namely bagging and boosting. Bagging often considers homogeneous base models, learns them independently from each other in parallel and combines them following some kind of deterministic averaging process. Boosting, however, often considers homogeneous base models, and learns them sequentially in an adaptative way, which means that one base model depends on the previous ones, and then combines them following a deterministic strategy (Demir, 2016).

The evolution of decision-tree-based algorithms is shown in Figure 4.11. To begin with, the decision tree is a graphical representation of possible solutions. Then, RF is based on a bagging algorithm which consists of bootstrapping and aggregating, but only a subset of features are selected at random to build a forest. Next, gradient boosting is established by employing a gradient descent algorithm to minimize errors in sequential models. Finally, XGBoost is one step further, it is based on gradient boosting together with some advanced optimizations, for instance, adding regularization terms in the loss function to avoid overfitting. The details of these techniques will be illustrated in the examples.

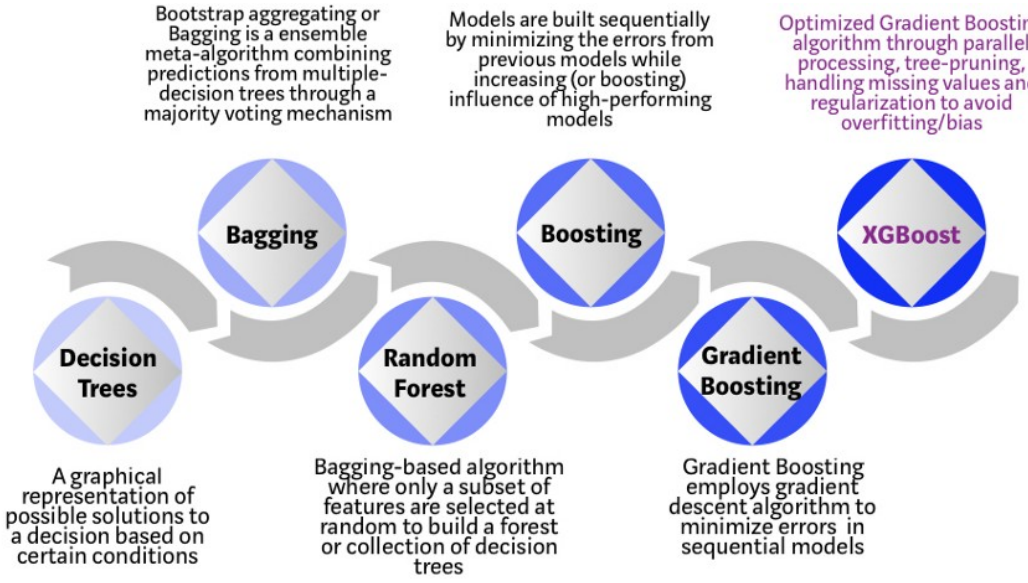


Figure 4.11. The evolution of decision-tree-based algorithms (Morde, 2019).

4.4.3 Random Forest

RF consists of a committee of decision trees (also known as classification trees or "CART" regression 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. The flowchart of RF for regression is shown in Figure 4.12.

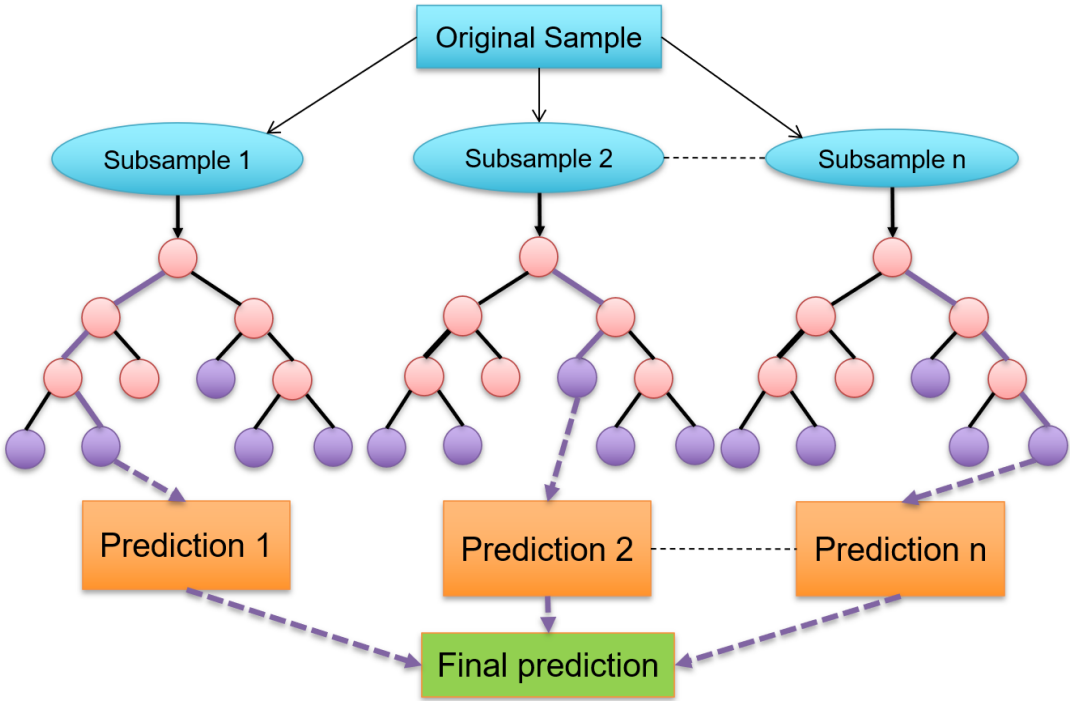


Figure 4.12. The flowchart of random forest for regression.

First of all, subsamples are generated from original samples 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. On average, 63.2% of the original data are presented in an average bootstrap sample. This result can be derived using elementary probability. Suppose the original data contains n observations. A bootstrap sample is generated by sampling with replacement from the data. The probability that a particular observation is not chosen from a set of n observations is $1 - 1/n$, so the probability that the observation is not chosen n times is $(1 - 1/n)^n$. This is the probability that the observation does not appear in a bootstrap sample. Therefore, when n is large, the probability that an observation is not chosen is approximately $1/e \approx 0.368$ since the limit as $\lim_{n\to\infty}(1 - 1/n)^n = 1/e$. So the answer would be $1 - 0.368 = 63.2\%$.

Next, the models are built by constructing a decision tree for each subsample 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 similar which increases variance. Further illustrations are presented with a simple example in the Appendix B.2.

4.4.4 XGBoost

As mentioned above, XGBoost 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 L1 and L2 regularization terms are added into the loss function to give penalization to complex models in order to avoid overfitting, improving the

model generalization capabilities. The key difference between these two regularization terms is the penalty term. L1 and L2 add "absolute value of magnitude" of coefficient and "squared magnitude" of coefficient as penalty terms to the loss function, respectively. In addition, the developers of XGBoost have made a number of important performance enhancements to different parts of the implementation which make a big difference in speed and memory utilization. An example is provided in the Appendix B.3 to give an intuition of how XGBoost works.

4.5 Conclusion

This chapter gives a brief introduction to the ML techniques used in this study. Some basic ML concepts are first discussed. Starting with a typical type of ML, supervised learning. Then some validation metrics in ML is explained, together with a typical optimization algorithm, the gradient descent algorithm which is used for optimizing the validation metrics. Next, an important property of ML models, the bias and variance trade-off is put forward, which is the key to the hyperparameter tuning process.

After that, using linear regression as a starter, three nonlinear regression algorithms, namely the ANN, SVM and GPR are offered. Among them, ANN can be regarded as a generalized-linear model. SVM and GPR are both kernel-function based algorithms. SVM is able to transfer the data onto a higher dimensional feature space through a kernel-function, and then carry out a linear regression in the high dimensional space. While GPR can make predictions incorporating prior knowledge through kernel-functions and provide uncertainty measures over predictions. In addition, some simple examples are provided for the algorithms in the Appendix B to get an intuition of how they work.

Chapter 5

Model implementation

5.1 Introduction

In this section, the machine learning techniques introduced in the Chapter 4 are implemented on the training dataset (A.1). Starting with the cross-validation (CV) strategies, two types of CV strategies are introduced, namely the k-fold CV and group k-fold CV. Next, two methods for tuning the hyperparameters, which is the key step in training a machine learning model, are introduced, namely the grid search CV and random search CV. They are capable of finding the best set of hyperparameters. After that, the machine learning models are constructed. The Monte Carlo simulation is performed for the machine learning models in order to take into account the random splitting effect in the dataset. Lastly, a sensitivity analysis is carried out to evaluate the importance of input parameters for modelling using partial dependence plots and feature importance functions.

5.2 Cross-validation Strategies

Cross-validation is one of the techniques used to test the effectiveness of a machine learning 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 is available, which is the case in this study. Therefore, it is necessary to consider selecting proper CV strategies at the beginning of the study.

There are various kinds of CV strategies. Here, which strategy we choose actually depends on what kind of scenario we want to simulate for the future application of the model. Starting with the most commonly used k-fold CV, a visualization example of which is shown in Fig. 5.1, the dataset is split into k consecutive folds, each fold is used once as 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 in each iteration using statistic metrics which in this study is the coefficient of determination (R^2). Finally, take the average of the scores in all the iterations and that will be the performance metric for the model. This strategy reproduces a scenario where new samples are added to sites where we already have data, whereas group k-fold reproduces a scenario where we add a new site.

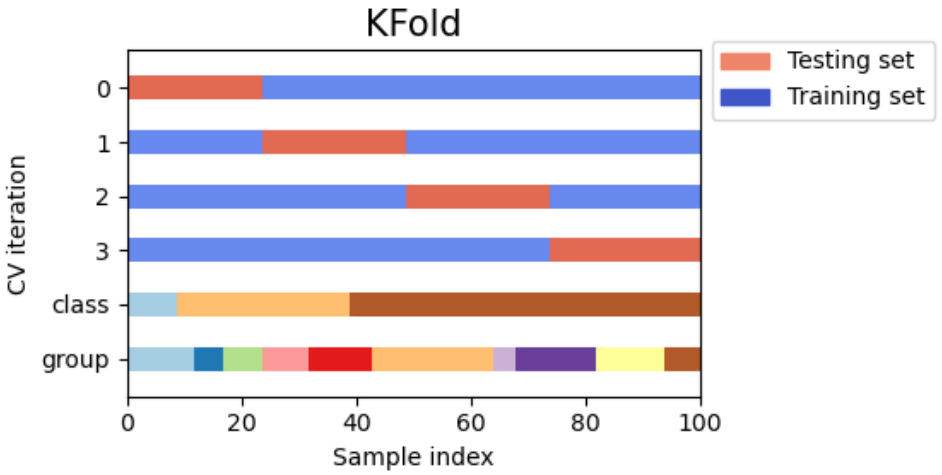


Figure 5.1. Visualization of a k-fold CV with 3 classes and 4 iterations (Pedregosa et al., 2011).

Choosing an appropriate value for k is fairly crucial since a poor selection may result in a misrepresentative idea of the performance of the model. For instance, if the value of k is too small, then the error estimation will probably end up with a score with a high bias, which means that the result may change a lot based on the data used to fit the model. Conversely, if the k is too large, e.g. to the extreme, being equal to the number of samples, the error estimation will be high in bias, showing an overestimation of the skill of the model.

Three common tactics for choosing a value for k are as follows (Brownlee, 2018):

- Representative: The value for k is chosen such that each train/test group of data samples is large enough to be statistically representative of the broader dataset;
- $k = 10$: The value for k is fixed to 10, a value that has been found through experimentation to generally result in a model skill estimate with low bias and a modest variance;
- $k = n$: The value for k is fixed to n, where n is the size of the dataset to give each test sample an opportunity to be used in the hold-out dataset. This approach is computationally expensive and it is called leave-one-out cross-validation;

To summarize, there is no formal rule for which one to choose and the choice of k is usually 5 or 10. Typically, given the considerations of the bias-variance trade-off, it is chosen as 5 or 10 since these values have been shown empirically to yield test error rate estimates that suffer neither from excessively high bias nor from very high variance (James et al., 2013). Thus in this research, the value of k is chosen as 10.

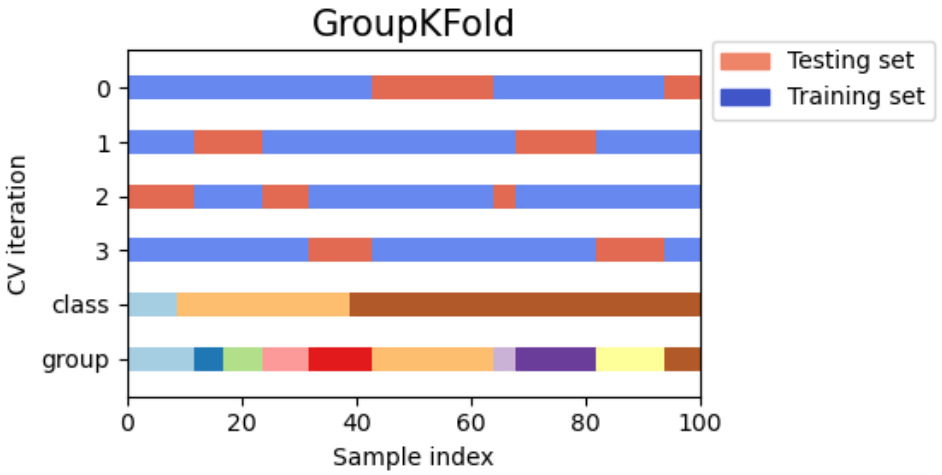


Figure 5.2. Visualization of a group k-fold CV with 3 classes and 4 iterations (Pedregosa et al., 2011).

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. A visualization example of it is shown in Fig. 5.2. This strategy best simulates the scenarios where the model will be tested on completely unseen data, which in this study is the data from new countries. This would be an invaluable application for machine learning, 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 a machine learning context because training and testing population share the same distribution, which in essence means 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 testing distribution can be very different from the training distribution, bringing a tougher challenge to the machine learning algorithms, thus usually leading to a relatively poor result. Implementing both of these two CV strategies for training the models, we end up with two sets of 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 machine learning models.

5.3 Hyperparameter tuning

In the training of the machine learning models, the hyperparameters discussed in Chapter 4 need to be calibrated based on their performance on the validation set. A grid search is a powerful tool to use. It is able to exhaustively search over specified parameter values for an estimator. An example of using grid search CV (k-fold CV) is given in Fig. 5.3. Using grid search CV, five hyperparameters in the Random Forest model are tuned based on their given ranges. The result shows that the R^2 value of the best combination of these five hyperparameters is 0.749.

However, the grid search CV is truly a computationally expensive method due to its exhaustiveness. It takes so long to train some models with many hyperparameters that a new method is required. That's where random search comes into play. In contrast to grid search, not all parameter values are tried out, but rather a fixed number of parameter settings is sampled from the specified distributions. It has been 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). Fig. 5.4 provides a simple illustration of the superiority of applying random search. In this illustration,

```
#KFold (squared_error)
param_grid = {'n_estimators': np.arange(1,100+1,1), 'max_depth': np.arange(1, 1+20, 1),
              'min_samples_split': np.arange(2, 2+20, 1), 'min_samples_leaf':np.arange(1, 1+10, 1),
              'max_leaf_nodes': np.arange(25, 50, 1)              }
regr = RandomForestRegressor(random_state=randomstate, criterion='squared_error')
cv = KFold(n_splits=10, random_state=randomstate, shuffle=True)
GS = RandomizedSearchCV(regr, param_grid, cv = cv, scoring = 'r2', n_iter = 1000)
GS.fit(X,y)
print(GS.best_params_, GS.best_score_)

{'n_estimators': 92, 'min_samples_split': 4, 'min_samples_leaf': 2, 'max_leaf_nodes': 48, 'max_depth': 9} 0.7494733716229676
```

Figure 5.3. Using a grid search CV for tuning the hyperparameters of Random Forest.

a grid of points distribute evenly in the original 2-d space, but their projections onto either the unimportant or the important parameter subspace produce an inefficient coverage of the subspace. Conversely, the random points are slightly less evenly distributed in the original 2-d space, however, much more evenly distributed in the two subspaces. It can be concluded that random search, in essence, trades a small reduction in efficiency in low-dimensional spaces for a large improvement in efficiency in high-dimensional search spaces.

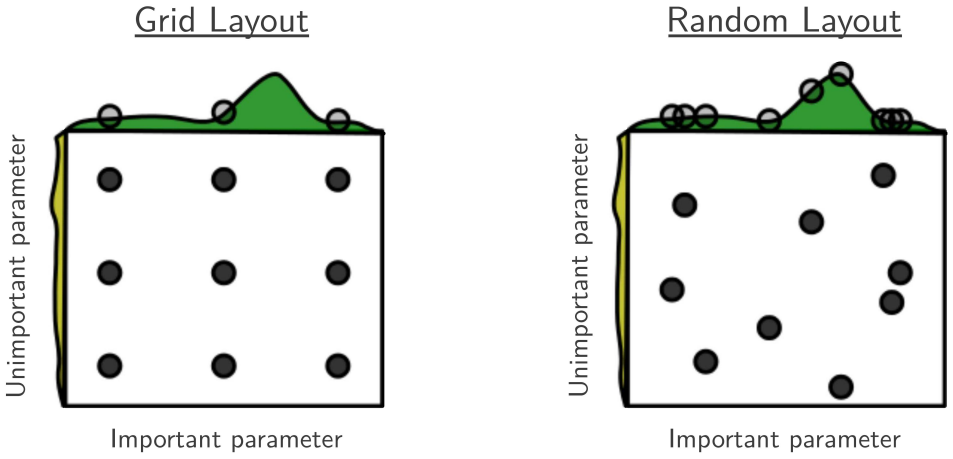


Figure 5.4. Grid and random search of nine trials for optimizing a function $f(x, y) = g(x) + h(y) \approx g(x)$ with low effective dimensionality. Above each square g(x) is shown in green, and on the left of each square h(y) is shown in yellow. With grid search, nine trials only test g(x) in three distinct places. With random search, all nine trials explore distinct values of g. This failure of grid search is the rule rather than the exception in high dimensional hyper-parameter optimization (Bergstra and Bengio, 2012).

In this research, grid search CV is applied to SVM and GPR since there are only 3 or 4 hyper-parameters that require tuning. While random search CV has been applied to RF, XGBoost and ANN since they have a lot more hyperparameters to tune.

5.4 Monte Carlo analysis

The robustness of the algorithms can also be evaluated from a statistical point of view. As stated in the section 5.2, in the k-fold CV, the value of k is chosen as 10, which means that in each iteration, 90% of the experimental data were randomly selected in order to train and construct the machine learning model. Then in each iteration, the performance of the model is evaluated on the validation set which contains 10% of the experimental data. Lastly, the performance of the

model in each iteration is averaged to give a final performance score. This performance score of a model, however, can inevitably be influenced by the choice of the sample indexes. Therefore, a total number of 1000 numerical simulations are next carried out, taking into account the random splitting effect in the dataset. To be more specific, the constructed model is tested on 1000 subsets generated with 1000 different random seeds in the train_test_split process in Pedregosa et al. (2011). This repetition of a simulation taking into account the random effect of input could also be called the Monte Carlo simulation (Ly et al., 2019). The results of the Monte Carlo analysis are presented in the section 7.4.

5.5 Sensitivity Analysis

In order to evaluate the importance of input parameters for modelling, a sensitivity analysis is carried out using partial dependence plots (PDP), which is an efficient way to investigate the relationship between inputs and output. Partial dependence plots show the dependence between the target response and a set of input features of interest, marginalizing the values of all other input features. The partial dependence function for regression is defined as Molnar (2020):

$$\hat{f}_S\left(x_S\right)=\frac{1}{n}\sum_{i=1}^{n}\hat{f}\left(x_S,x_C^{(i)}\right) \tag{5.1}$$

where X_s are the features that we are interested in and for which the partial dependence function should be plotted; X_c are the other features used in the machine learning model (using their actual values); n is the number of instances in the training dataset.

The function shows the relationship between the features in set S we are interested in and the predicted outcome. It is also worth mentioning that an assumption of the PDP is that the features in C are not correlated with the features in S. If this assumption is violated, the averages calculated for the partial dependence plot will include data points that are very unlikely or even impossible.

Another way of evaluating the importance of input parameters is by implementing a feature importance function in Pedregosa et al. (2011) on RF. As discussed in section 4.4.3, many individual decision trees are constructed in the training process of RF. Then the prediction of all the trees is averaged to make the final prediction. RF is referred to as the ensemble technique since it uses a collection of results to make a final decision. In essence, the feature importance in RF is calculated as the decrease in node impurity weighted by the probability of samples reaching that node. As this study deals with a regression problem, the node impurity here refers to the variance reduction after the split of a node evaluated by the MSE (or MAE as an alternative). And the node probability is calculated by dividing the number of samples reaching the node by the total number of samples. The higher the obtained final value is, the more important the feature is. To be specific, assuming there are only two child nodes for one node (binary tree) in RF, scikit-learn calculates the importance of a node using Gini importance (Ronaghan, 2019):

$$ni_j=w_jC_j-w_{left(j)}C_{left(j)}-w_{\text{right}(j)}C_{\text{right }(j)} \tag{5.2}$$

where ni_j is the importance of node j; w_j is the weighted number of samples reaching node j; C_j is the impurity value of node j; left(j) is the child node from left split on node j; right(j) is the child node from right split on node j.

The importance of each feature on a decision tree is then calculated as (Ronaghan, 2019):

$$fi_i = \frac{\sum_{j:\text{ node } j \text{ splits on feature } i} ni_j}{\sum_{k\in \text{ all nodes}} ni_k} \tag{5.3}$$

where fi_i is the importance of feature i; ni_j is the importance of node j.

These can then be normalized to a value between 0 and 1 by dividing by the sum of all feature importance values (Ronaghan, 2019):

$$\text{normf } i_i = \frac{fi_i}{\sum_{j\in all\text{features}} fi_j} \tag{5.4}$$

The final feature importance, at the Random Forest level, is its average over all the trees. The sum of the feature's importance value on each tree is calculated and divided by the total number of trees (Ronaghan, 2019):

$$RFfi_i = \frac{\sum_{j\in \text{ all trees}} \text{norm } fi_{ij}}{T} \tag{5.5}$$

where $RFfi_i$ is the importance of feature i calculated from all trees in the RF model; $normfi_{ij}$ is the normalized feature importance for i in tree j; T is the total number of trees.

The result of the PDP and feature importance values will be presented in the section 7.5.

5.6 Conclusion

This chapter presents the methods that have been adopted in the implementation of the machine learning models on the training dataset. In the first place, as the training dataset applied in this study is small, the cross-validation strategies are introduced in order to divide the dataset multiple times, and then average the results of multiple evaluations, so as to eliminate the adverse effects caused by the randomness in a single dataset division. Two types of CV strategies are proposed, namely the k-fold CV and the group k-fold CV. Comparably speaking, the k-fold CV simulates a more mild scenario, thus leading to a more optimistic estimation of the performance of the machine learning models. Contrarily, group k-fold CV simulates a more complex scenario, thus leading to a relatively more conservative estimation of the performance of the models. Utilizing both of these two CV strategies together enables us to provide a more comprehensive estimation of the performance of the machine learning models. Next, two methods for the hyperparameter tuning are put forward, namely the grid search CV and random search CV. Grid search is able to exhaustively search over the specified parameter values for an estimator, suitable for tuning a relatively small number of hyperparameters. In contrast, random search tries out a fixed number of parameter settings sampled from the specified distribution, applicable for tuning a large number of hyperparameters. After that, the Monte Carlo analysis is carried out to evaluate the robustness of the model on the validation set using 1000 different random seeds. After all, a sensitivity analysis is proposed by utilizing the partial dependence plots and the feature importance function to estimate the importance of input parameters for modelling. Above is all the training and constructing process of the machine learning models in the training and validation set. The testing dataset used in this study will be introduced in Chapter 6.

Chapter 6

Testing dataset

6.1 Introduction

After applying the machine learning models to the training dataset as presented in Chapter 5, the performance of the models is further tested with an unseen testing dataset, which is introduced in this chapter. 20 laboratory test samples obtained in a location, provided by de Gast (2020) are taken as the 20 data samples in the testing dataset in this research. These 20 samples were collected from 11 semi-continuous boreholes at different depths. This chapter starts with introducing input variables in the testing dataset, which are identical to the ones stated in Chapter 3, namely the effective stress, cone tip resistance, effective cone tip resistance and excess pore pressure. The data of the CPTs (provided by de Gast (2020)) that are in the vicinity of the 11 boreholes of the 20 laboratory samples is interpreted and then processed using GPR in order to obtain representative values of the input variables at the location of those laboratory samples. Then a brief introduction to achieving the output variable, the undrained shear strength of soil through laboratory tests is provided.

6.2 Input variables

In de Gast (2020)'s research, a site investigation was completed at Leendert de Boerspolder (Fig. 6.1), a polder located close to Leiden in the Netherlands. This place is typical of the western Netherlands, with a dyke founded on soft material in order to defend the land from water. This particular dyke has been on the maps since 1611, and it has been maintained first by the local farmers and then by the local water authority named 'Hoogheemraadschap van Rijnland'. The dyke is made of sand, silt, clay and rubble. Having been constructed and maintained over the years, this man-made embankment has caused the soft layers to compress (de Gast, 2020).

100 CPTs, class 1 accuracy according to NEN-EN-ISO 22476-1, were conducted in location C in Fig. 6.1. The data was collected over a period of two weeks in a location where the original ground surface had been partly compressed by an old dyke. The site investigation was performed in a grid as shown in Fig. 6.2. The grid of CPTs was parallel to the dyke, with CPT indexes 34-44 and 69-86 located on the crest of the dyke (Zone 1, Line 7), CPT indexes 45-54, 92-94, 97 and 98 on the slope of the dyke (Zone 2, Line 6), and CPT indexes 23-33 and 95-96 located at the toe of the dyke (Zone 2, Line 5). The remaining CPTs were located in the polder next to the dyke (Zone 3, Lines 1-4).

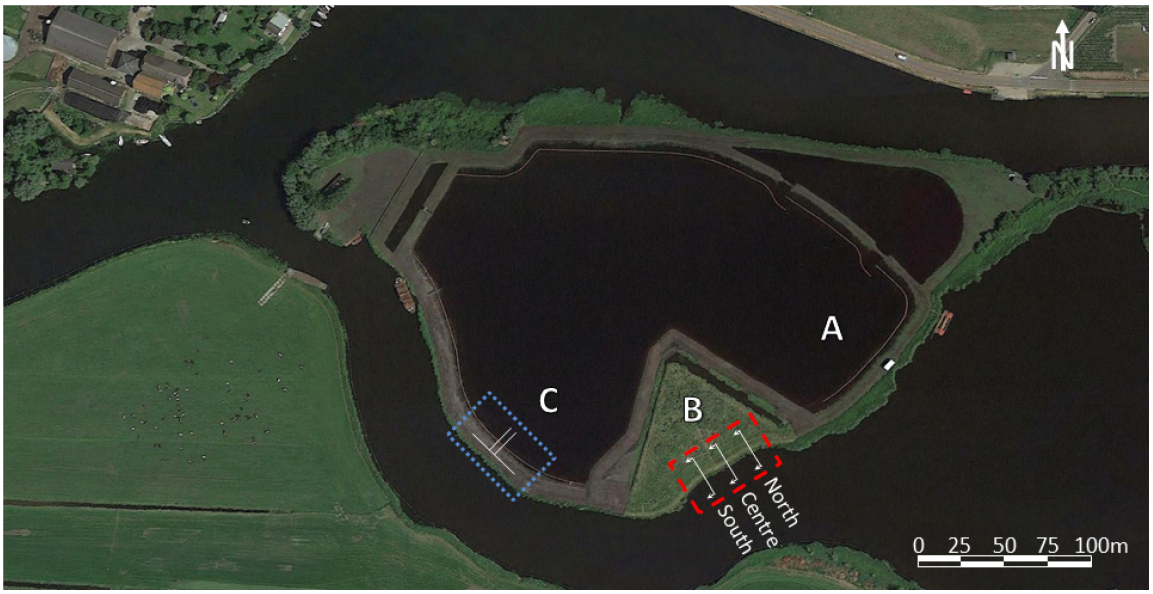


Figure 6.1. An aerial photograph of Leendert de Boerspolder, taken in 2015, indicated are: (A) and (B) locations not related to this study, (C) location of where the 100 CPTs were conducted and where 20 samples were collected for laboratory tests (de Gast, 2020).

As stated in the introduction of this chapter, there were also 20 samples for the laboratory tests collected from 11 semi-continuous boreholes in location C. The locations of these 11 boreholes are shown in Fig. 6.3 and their locations in relation to the CPTs are presented in Fig. 6.4. With this figure, the CPTs that are in the vicinity of the 11 boreholes of the 20 laboratory samples can be selected. Then the data of the selected CPTs are automatically interpreted using a script. The interpreted results are then fed to GPR to obtain the representative input at the location of the laboratory samples, that is, the final input in the testing dataset.

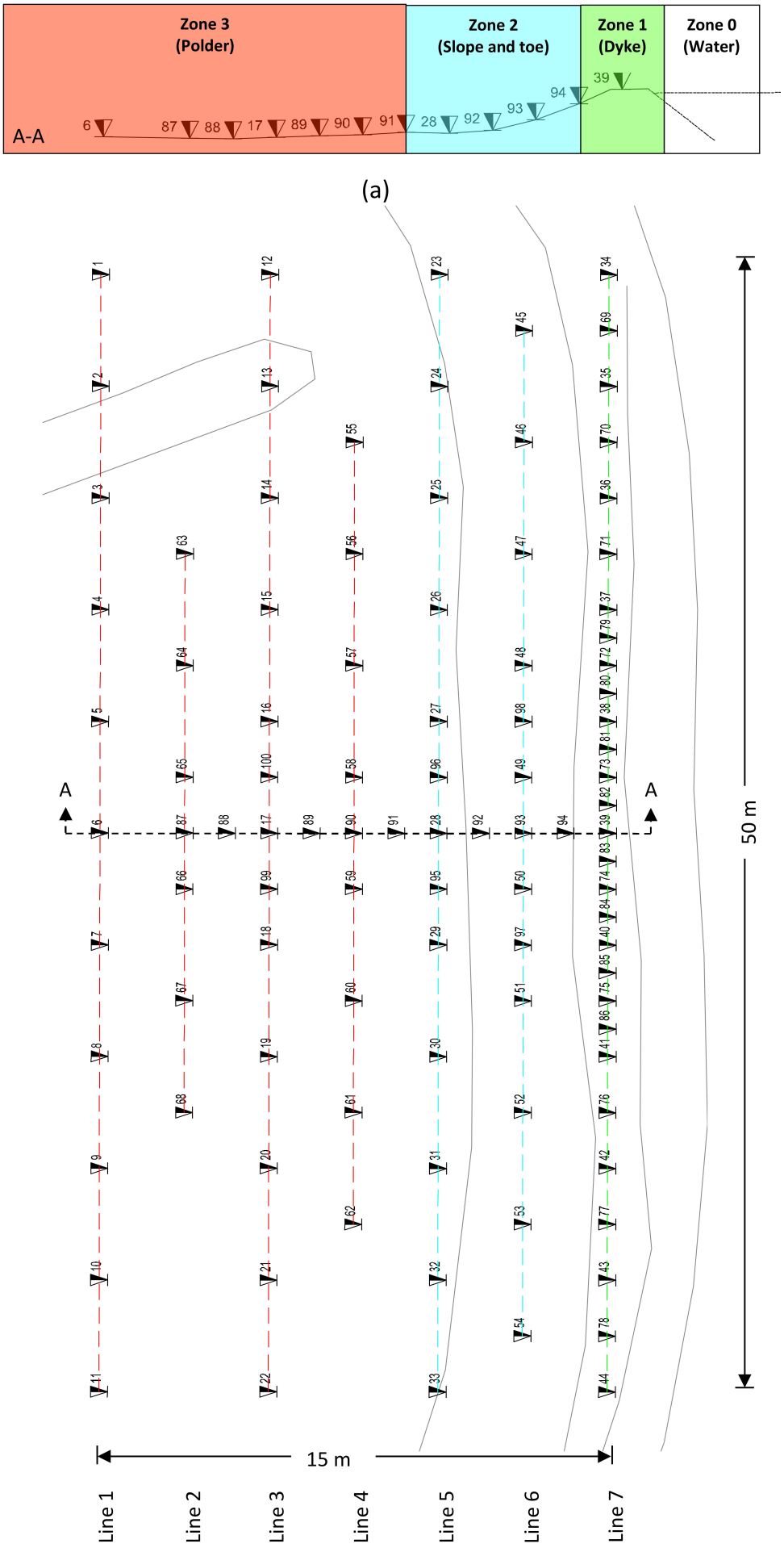


Figure 6.2. CPT grid (50m x 15m): (a) main testing zones, cross section (A-A) illustrated; (b) plan view of CPT locations (de Gast, 2020).

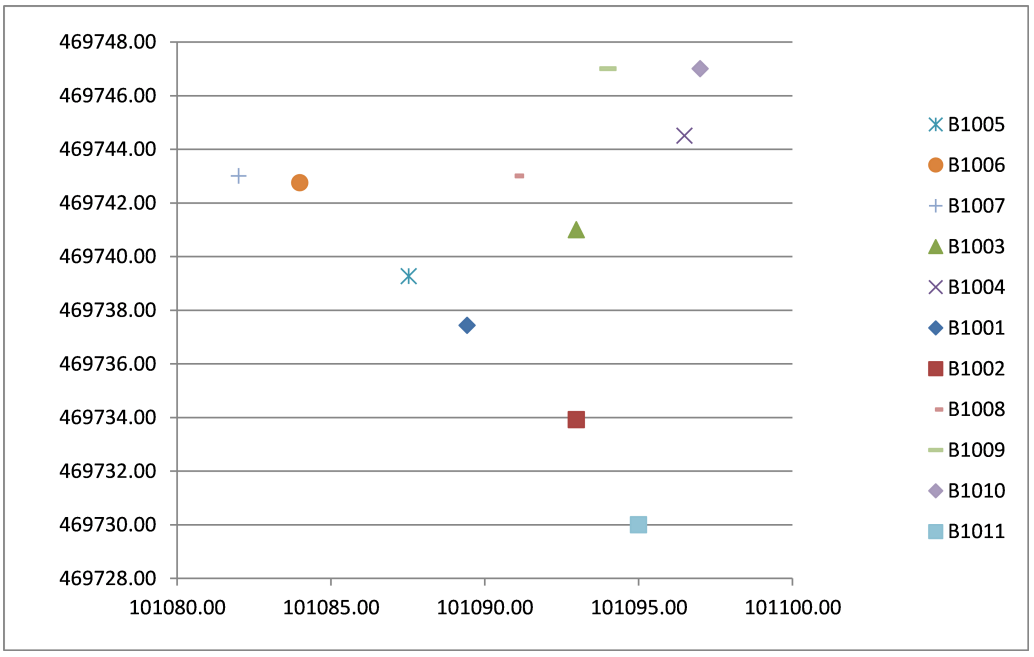


Figure 6.3. The location of 11 boreholes for the laboratory tests in the location C using Global RD (Dutch reference) XYZ-coordinates (de Gast, 2020).

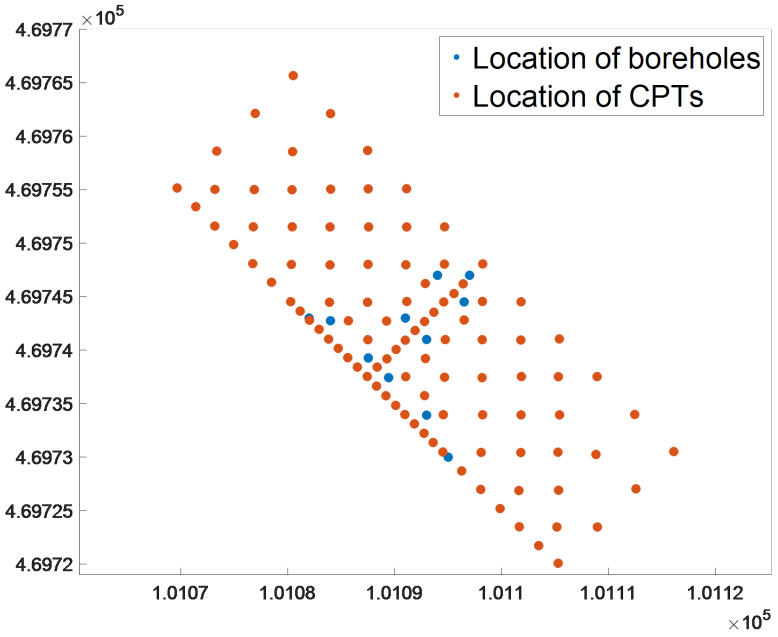


Figure 6.4. The location of 11 boreholes for the laboratory tests and 100 CPTs in the location C using Global RD (Dutch reference) XYZ-coordinates.

In addition, an illustration of the application of GPR on the prediction of representative CPT data at the location of the laboratory borehole through several CPTs that are in close vicinity is provided in Fig. 6.5 (The confidence interval is not included since multiple inputs with uncertainties is not considered). To obtain the representative CPT data at the location of borehole B1009 which is for the laboratory tests, CPT index 66, 88 and 99 are selected since they are in close vicinity. The raw CPT data of these three boreholes are first interpreted with

a script. After the interpretation, the corrected cone tip resistances (q_t) of the three boreholes 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.

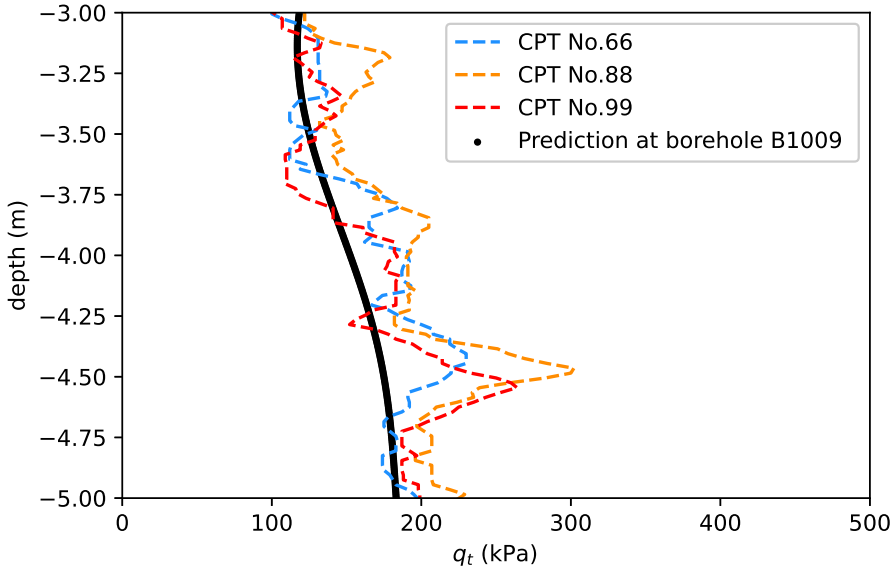


Figure 6.5. An illustration of the application of Gaussian process regression on processing CPT data.

Table 6.1 shows the 20 laboratory samples and the corresponding selected CPTs.

Laboratory samples	B1007-8 B1007-11	B1006-3 B1006-7	B1010-4 B1010-6 B1010-9	B1009-4 B1009-5 B1009-8
CPT boreholes	75	85, 40, 51, 97	6, 87	66, 88, 99
Laboratory samples	B1008-4 B1008-6	B1005-3 B1005-7	B1001-5 B1001-6	B1011-5 B1011-7 B1011-10 B1011-11
CPT boreholes	59, 91, 95, 28	50, 94, 83, 39	93, 94, 73, 49, 82	37

Table 6.1. 20 laboratory samples and the corresponding selected CPTs.

Table 6.2 shows the selected CPTs together with the corresponding representative CPT data after processing using GPR. The final inputs in the testing dataset can be found in the Appendix A.2

CPT boreholes	Depth (m)	sigv' (kPa)	qt (kPa)	sigv (kPa)	u2 (kPa)	u0 (kPa)
75	4.76	35.00	272.00	78.00	5.93	44.00
	6.31	41.00	181.00	100.00	26.56	59.00
85, 40,	2.17	18.00	246.25	35.50	22.24	18.00
51, 97	4.37	30.75	224.50	70.25	36.82	40.00
6, 87	4.06	24.00	174.50	61.00	56.60	37.00
	5.06	28.50	176.50	75.00	88.46	47.00
	6.66	35.00	235.00	98.00	134.96	63.00
66, 88, 99	3.92	24.33	178.33	59.67	49.65	35.00
	5.02	29.00	205.75	75.33	84.67	46.00
	6.12	33.67	224.00	91.33	118.20	57.00
59, 91,	3.88	24.50	154.50	59.50	69.09	35.00
95, 28	4.68	28.25	225.13	71.50	98.60	43.00
50, 94,	2.39	18.75	261.33	38.50	13.91	20.00
83, 39	4.54	31.25	218.00	72.25	40.80	42.00
93, 94, 73,	3.23	24.20	250.20	52.80	31.01	28.00
49, 82	4.13	29.60	204.80	66.60	61.85	37.00
37	3.15	24.00	358.25	52.00	-2.92	28.00
	4.25	31.00	225.00	70.00	18.25	39.00
	5.85	39.00	164.25	94.00	49.87	55.00
	6.45	42.00	182.25	102.25	70.17	61.00

Table 6.2. 20 laboratory samples with their selected CPTs, together with the corresponding representative CPT data after processing.

6.3 Output

The Mohr-Coulomb failure criterion is generally applied to estimate the shear strength of soil. Based on the presence of soil failure on any plane when the resolved shear strength (3.2) reaches the critical state, it is essentially concerned with the stress state on the potential rupture planes in soil (Goktepe et al., 2008).

To illustrate, the stress state at a point in the soil can be graphically presented using Mohr's circle, as presented in Fig. 6.6. In a Cartesian coordinate system of normal stress and shear stress, the Mohr's circle can be plotted taking $\frac{\sigma_1+\sigma_3}{2}$ as the centre and $\frac{\sigma_1-\sigma_3}{2}$ as the radius, where σ_1 and σ_3 are the maximum and minimum principal stress respectively. Demonstrably, the coordinates of each point on the Mohr's circle represent normal stress and shear stress of the point in the corresponding plane, which means that Mohr's circle can represent the stress state of a point in the soil. If the soil shear strength parameters c and ϕ are given, the shear strength envelope can be plotted together with Mohr's circle. Then, if the whole circle is below the failure envelope, the shear stress at this point in any plane is less than the shear strength, thus there is no shear failure. If Mohr's circle is tangent to the shear strength envelope, then the shear stress is equal to the shear strength in the plane represented by the tangent point. This point is considered to be at the critical state and this Mohr's circle is called the limiting stress circle.

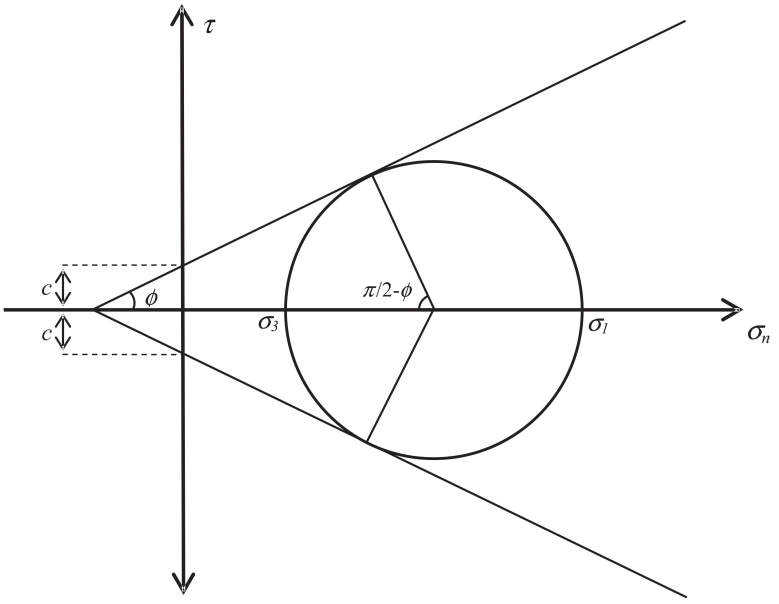


Figure 6.6. Geometrical view of Mohr-Coulomb failure criterion (Goktepe et al., 2008).

Consolidated undrained (CU) tests were carried out on the 20 laboratory samples in de Gast (2020)'s study. In a CU-test, water is filled into the pressure chamber where the sample is placed. Then the sample is subjected to a constant confining pressure σ_3. Meanwhile, the drain valve is opened, allowing the drainage consolidation of the sample. After the sample is consolidated and stable, the drain valve is closed and vertical compressive stress σ_1 is applied to the sample through a load ram. In such a manner, the vertical principal stress gradually increases while the horizontal principal stress stays constant. The sample is subjected to shear failure in the undrained condition in the end. The undrained shear strength and effective stress obtained from the CU tests for the 20 samples are presented in Table 6.3. This effective stress will be compared

with the effective stress obtained through the CPT test to ensure safety. And the undrained shear strength is taken as the output of the testing dataset.

Laboratory	B1007	B1007	B1006	B1006	B1010	B1010	B1010	B1009	B1009	B1009
boreholes	-8	-11	-3	-7	-4	-6	-9	-4	-5	-8
S_u	16.02	19.59	26.95	15.68	11.34	9.93	17.67	11.49	10.90	18.82
σ'_v	32.26	44.19	62.91	30.56	20.67	18.00	42.35	21.98	19.60	43.65
Laboratory	B1008	B1008	B1005	B1005	B1001	B1001	B1011	B1011	B1011	B1011
boreholes	-4	-6	-3	-7	-5	-6	-5	-7	-10	-11
S_u	9.61	14.11	19.70	13.16	20.47	25.80	24.97	13.30	20.61	20.61
σ'_v	17.90	36.21	47.40	26.24	45.46	53.60	58.83	28.13	45.21	45.21

Table 6.3. The undrained shear strength and effective stress obtained from the CU tests for the 20 samples.

6.4 Conclusion

This chapter introduces the testing dataset applied in this study. To start with, the location of the site investigation in de Gast (2020) is presented. In this site investigation, 100 CPT tests were conducted, together with 11 boreholes for the sampling of 20 samples for the laboratory tests. Then the CPT grid and the locations of the laboratory boreholes are provided. Plotting the locations of the CPT and laboratory boreholes together, the CPTs around the laboratory boreholes are then selected for generating representative CPT data at the location of the laboratory boreholes. The CPT data of the selected boreholes are then interpreted and applied to GPR. This representative CPT data at the location of the laboratory boreholes are taken as the inputs of the testing dataset. After that, the undrained shear strengths of the 20 samples collected from the 11 laboratory boreholes are determined through CU tests, which are taken as the output of the testing dataset. Finally, combining the data in Table 6.2 and Table 6.3, the testing dataset is displayed in the Appendix A.2. The results of the ML techniques on the testing dataset will be presented in Chapter 7.

Chapter 7

Results and discussion

7.1 Introduction

This chapter presents all the results obtained in this study. Starting with the hyperparameter tuning results of the five algorithms used, two models are constructed for each algorithm. One of them is obtained by adopting k-fold CV and the other is obtained by adopting group k-fold CV. Then the predicted values in the training and testing dataset together with their relative error and error distributions using each model are presented. The results are then compared and discussed in order to select the most appropriate model for the prediction of undrained shear strength from CPT data. Next, the results of the Monte Carlo analysis are provided for the evaluation of the robustness of the model on the validation set. Finally, the results of the sensitivity analysis are shown for the analysis of the importance of the input variables.

7.2 Hyperparameter tuning results

The hyperparameter tuning results of the ANN are presented in Table 7.1. During the calibration of the hyperparameters, 25 architectures of the neural network have been experimented with. The number of the hidden layers ranges from 1 to 5 and the number of neurons in each hidden layer is tried with 2, 4, 8, 16, and 32. The options for the activation function include the tanh function, relu function and logistic function. Two optimization solvers are tried, namely the adam and lbfgs. 8 alphas are sampled log-uniformly distributed between 10^{-8} and 0.1. 7 initial learning rates are sampled log-uniformly distributed between 10^{-6} and 1.0. The size of the mini-batches that have been experimented with is 16, 32, 64, 128 and 256.

The tuned hyperparameters of SVM are presented in Table 7.2. Possible kernel-functions to use include: linear function, polynomial function, radial basis function, sigmoid function and pre-computed function. 50 kernel coefficients are evenly sampled from 0.01 to 30. 20 regularization parameters are evenly sampled from -10 to 20. 5 epsilon values are evenly sampled from 0.1 to 0.5.

For the hyperparameter tuning of GPR, two kernel-functions, namely the radial basis function (RBF) and the Matern have been experimented with. They are both multiplied by a constant kernel and then added with a white kernel function before being fitting to the training data. The tuning results for k-fold CV and group k-fold CV are identical: the Matern kernel-function is chosen and the alpha which is the value added to the diagonal of the kernel matrix during fitting equals 1.0.

Parameters	Description	Value (k-fold)	Value (group k-fold)
hidden_layer _sizes	The ith element represents the number of neurons in the ith hidden layer	(8,8)	(16,16, 16,16)
activation	Activation function for the hidden layer	relu	tanh
solver	The solver for weight optimization	lbfgs	adam
alpha	Strength of the L2 regularization term	10^{-7}	10^{-7}
learning_ rate_init	The initial learning rate used	/	0.01
batch_size	Size of minibatches for stochastic optimizers	/	16

Table 7.1. ANN parameters tuned with k-fold CV and group k-fold CV.

Parameters	Description	Value (k-fold)	Value (group k-fold)
kernel	Specify the kernel type to be used	rbf	rbf
C	Kernel coefficient	1.60	30.00
gamma	Regularization parameter	3.06	0.06
epsilon	Specify the ϵ-tube within which no penalty is associated in the training loss function	0.10	0.10

Table 7.2. SVM parameters tuned with k-fold CV and group k-fold CV.

The tuned hyperparameters of RF are presented in Table 7.3. The number of trees has experimented from 1 to 100. The number of features to use is tried from 2 to 4. The minimum number of samples required to split an internal node has been experimented with from 2 to 20. The minimum number of samples required to be at a leaf node is experimented with from 1 to 10. The maximum number of leaf nodes a decision tree can have is tried with 25 to 50. The maximum depth of the tree is sampled from 1 to 20. The function to measure the quality of a split is chosen from RMSE and MAE.

Parameters	Description	Value (k-fold)	Value (group k-fold)
n_estimators	Number of trees in the forest	92	90
max_features	Number of features to consider when looking for the best split	4	4
min_samples_split	Minimum number of samples required to split an internal node	4	13
min_samples_leaf	Minimum number of samples required to be at a leaf node	2	6
max_leaf_nodes	Maximum number of leaf nodes a decision tree can have	48	47
max_depth	Maximum depth of the tree.	9	2
criterion	The function to measure the quality of a split	squared _error	squared _error

Table 7.3. RF parameters tuned with k-fold CV and group k-fold CV.

The tuned hyperparameters of XGBoost are presented in Table 7.4. The number of trees has experimented from 1 to 100. 6 ratios of the subsample are evenly sampled from 0.6 to 1.0. 25 learning rates are evenly sampled from 0.01 to 0.25. The maximum depth of a tree that has been experimented with is from 2 to 10. 20 L1 and 50 L2 regularization terms on weights are evenly sampled from 0 to 0.2 and from 0 to 5 respectively. 50 values of gamma are evenly sampled from 0 to 5. The boosters to choose from include gbtree and dart.

Parameters	Description	Value (k-fold)	Value (group k-fold)
n_estimators	Number of trees in the forest	47	14
subsample	Subsample ratio of the training instances	0.50	0.60
learning_rate	Step size shrinkage used in update to prevent overfitting	0.11	0.11
max_depth	Maximum depth of a tree	6	2
reg_alpha	L1 regularization term on weights	2.60	1.00
reg_lambda	L2 regularization term on weights	0.10	1.60
gamma	Minimum loss reduction required to make a further partition on a leaf node of the tree	4.80	3.90
booster	Which booster to use	gbtree	gbtree

Table 7.4. XGBoost parameters tuned with k-fold CV and group k-fold CV.

7.3 Results in the training and testing dataset

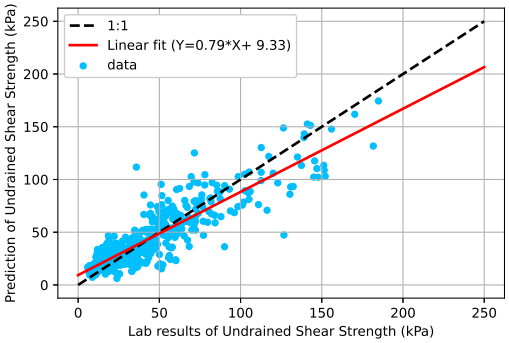
The results of ANN in the training and testing dataset are presented in Fig. 7.1 and Fig. 7.2 respectively.

The results of SVM in the training and testing dataset are presented in Fig. 7.3 and Fig. 7.4 respectively.

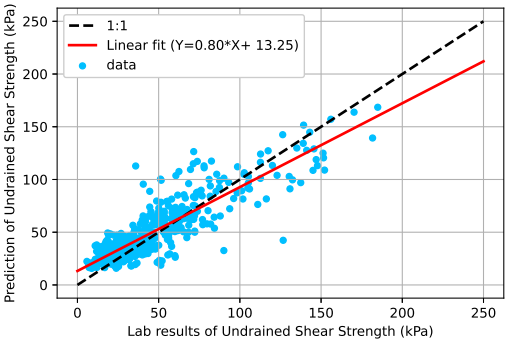
The results of GPR in the training and testing dataset are both presented in Fig. 7.5 since the tuning results from the k-fold and group k-fold CV are identical. In addition, the uncertainty quantification of the prediction using GPR is provided in Fig. 7.6. The blue area shows the 95% confidence interval which is obtained by adding and subtracting 1.96σ from the prediction. The σ is provided together with the prediction in GPR.

The results of RF in the training and testing dataset are presented in Fig. 7.7 and Fig. 7.8 respectively.

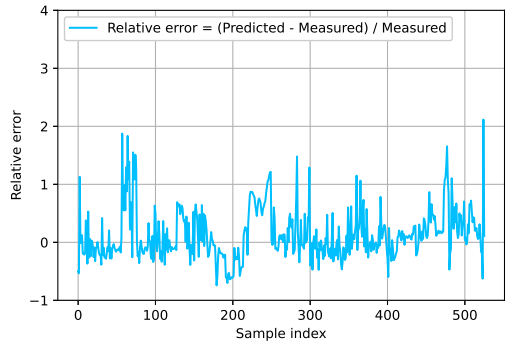
The results of XGBoost in the training and testing dataset are presented in Fig. 7.9 and Fig. 7.10 respectively.



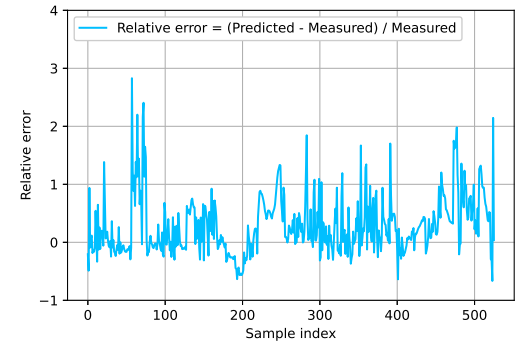
(a) ANN regression (k-fold)



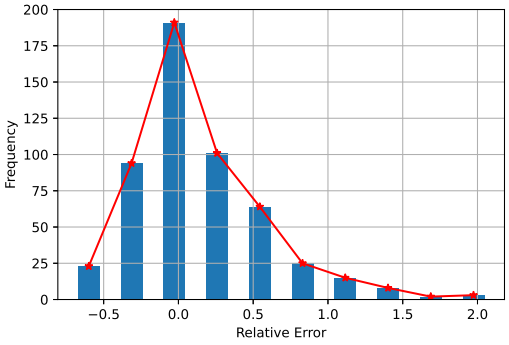
(b) ANN regression (group k-fold)



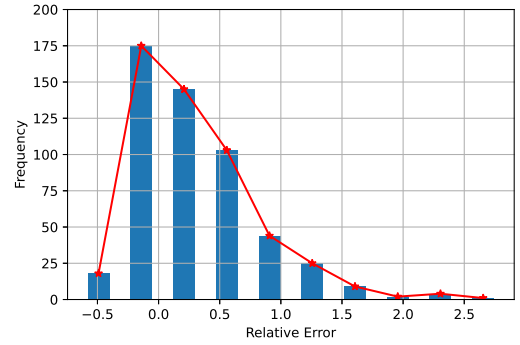
(c) Relative Error of ANN (k-fold)



(d) Relative Error of ANN (group k-fold)

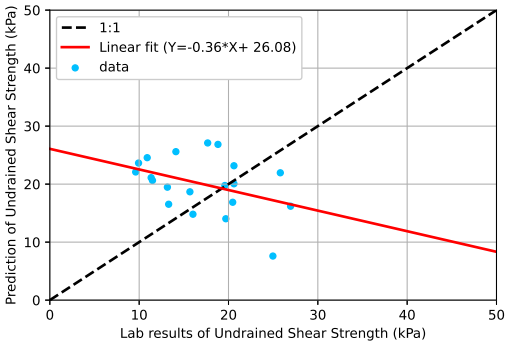


(e) Relative Error Distribution of ANN (k-fold)

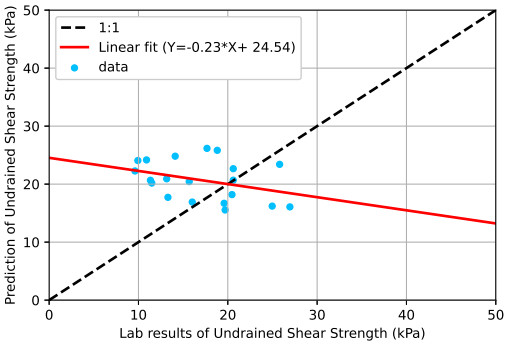


(f) Relative Error Distribution of ANN (group k-fold)

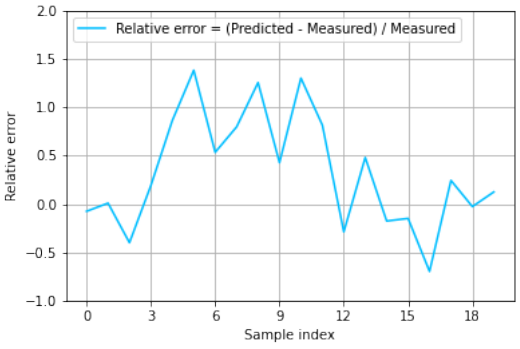
Figure 7.1. Best ANN models from k-fold and group k-fold CV on the training set.



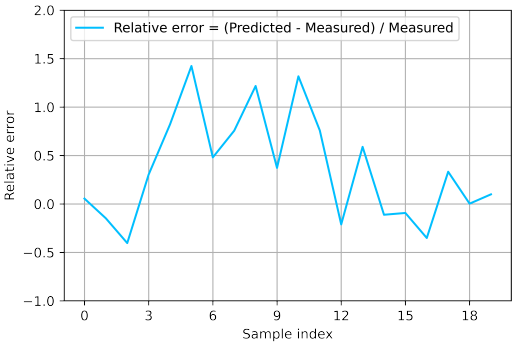
(a) ANN regression (k-fold)



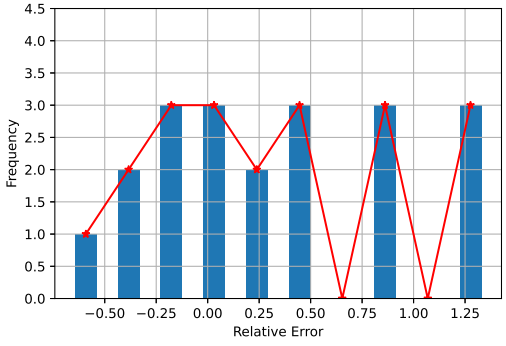
(b) ANN regression (group k-fold)



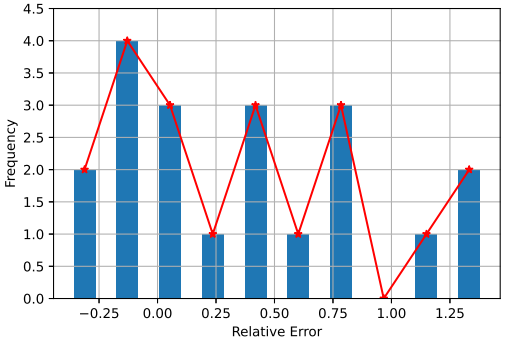
(c) Relative Error of ANN (k-fold)



(d) Relative Error of ANN (group k-fold)

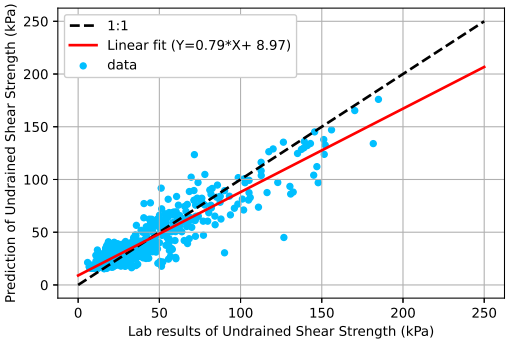


(e) Relative Error Distribution of ANN (k-fold)

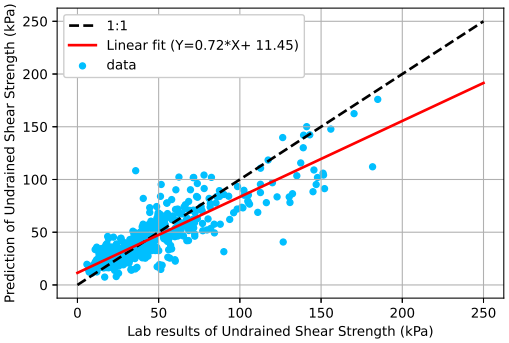


(f) Relative Error Distribution of ANN (group k-fold)

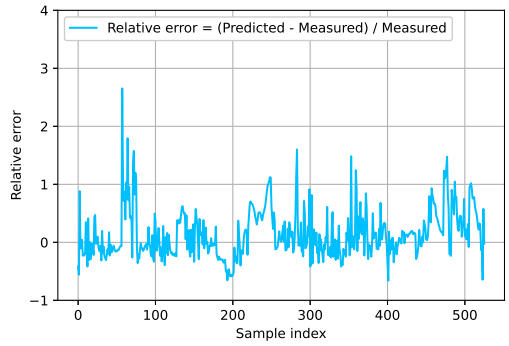
Figure 7.2. Best ANN models from k-fold and group k-fold CV on the testing set.



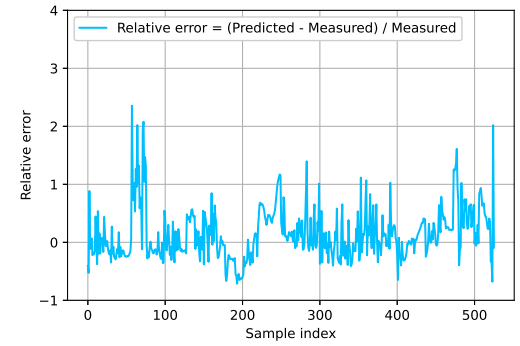
(a) SVM regression (k-fold)



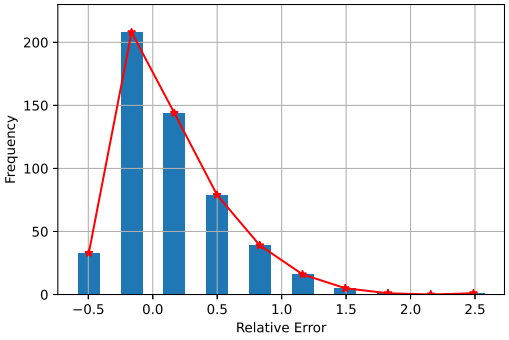
(b) SVM regression (group k-fold)



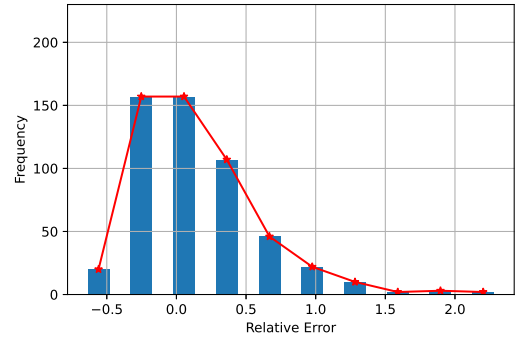
(c) Relative Error of SVM (k-fold)



(d) Relative Error of SVM (group k-fold)

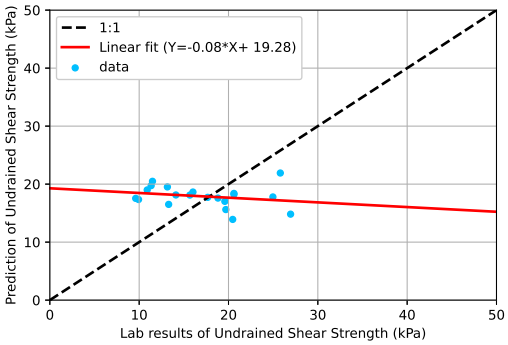


(e) Relative Error Distribution of SVM (k-fold)

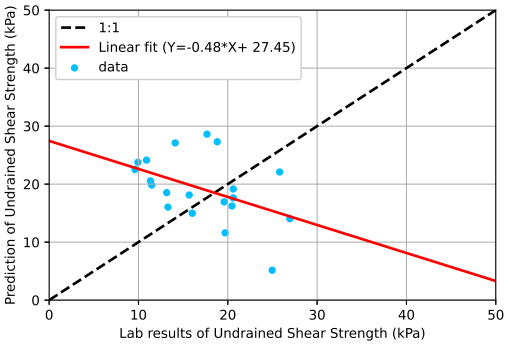


(f) Relative Error Distribution of SVM (group k-fold)

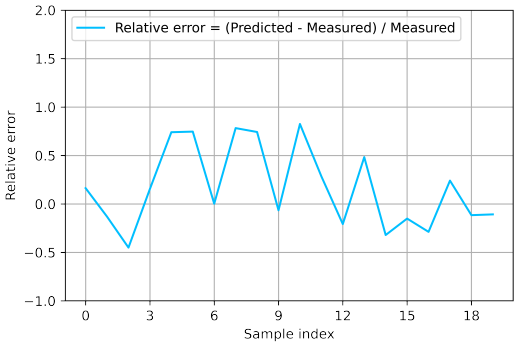
Figure 7.3. Best SVM models from k-fold and group k-fold CV on the training set.



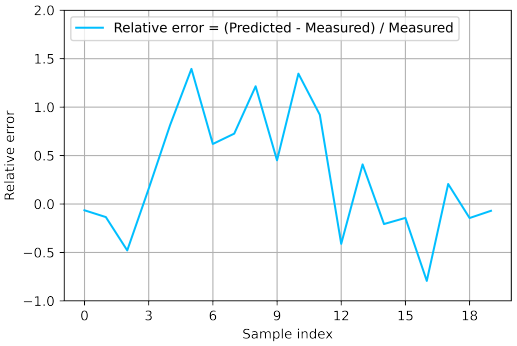
(a) SVM regression (k-fold)



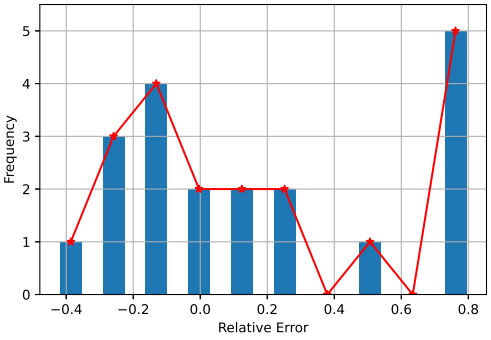
(b) SVM regression (group k-fold)



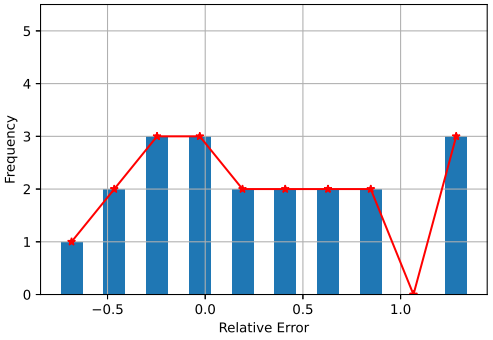
(c) Relative Error of SVM (k-fold)



(d) Relative Error of SVM (group k-fold)

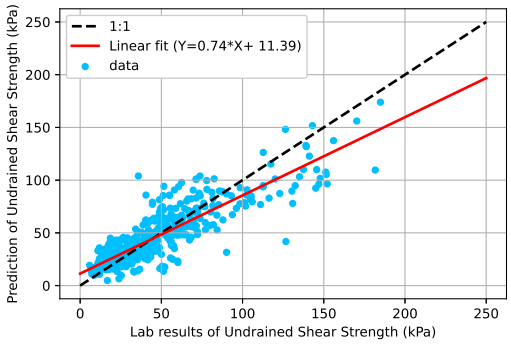


(e) Relative Error Distribution of SVM (k-fold)

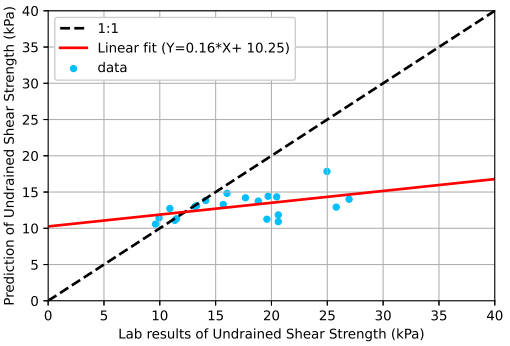


(f) Relative Error Distribution of SVM (group k-fold)

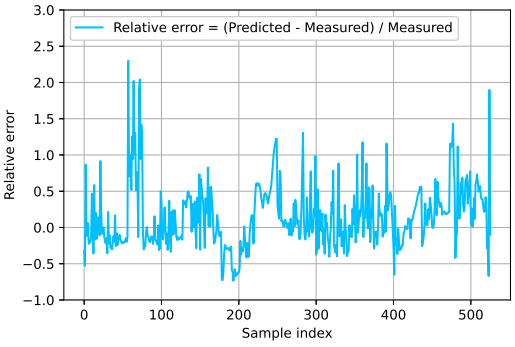
Figure 7.4. Best SVM models from k-fold and group k-fold CV on the testing set.



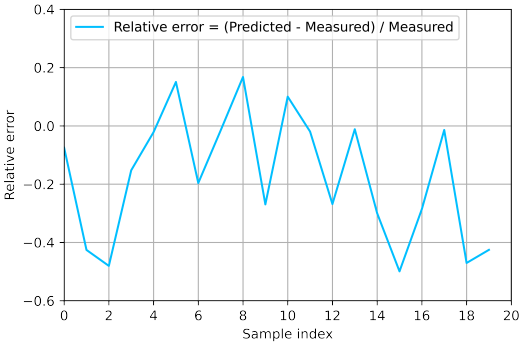
(a) GPR regression in the training dataset



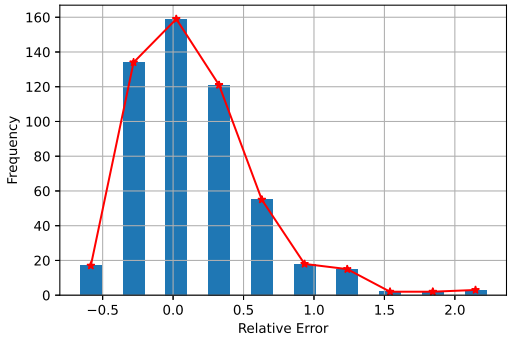
(b) GPR regression in the testing dataset



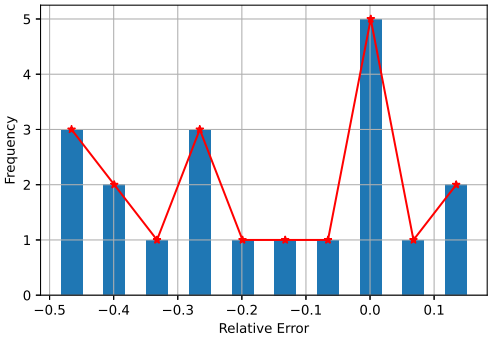
(c) Relative Error of GPR in the training dataset



(d) Relative Error of GPR in the testing dataset



(e) Relative Error Distribution of GPR in the training dataset



(f) Relative Error Distribution of GPR in the testing dataset

Figure 7.5. Best GPR model from k-fold and group k-fold CV on the training and testing set (the best models are identical from both CV strategies).

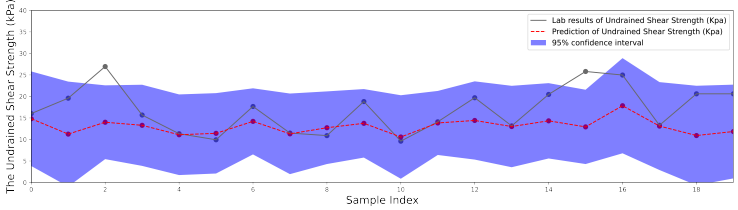
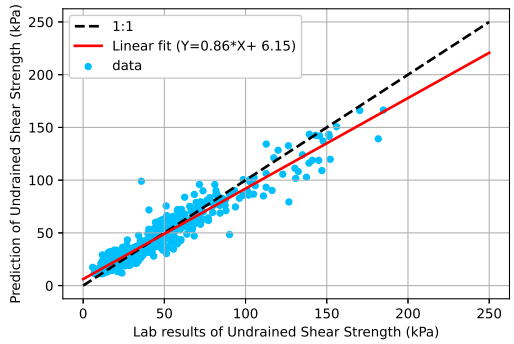
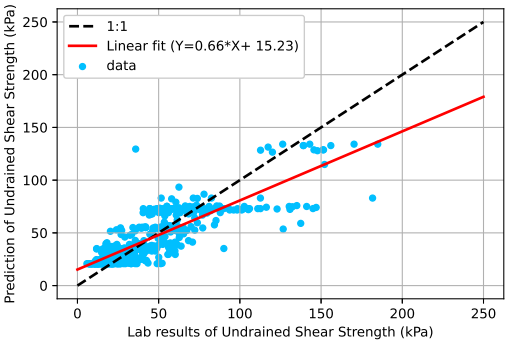


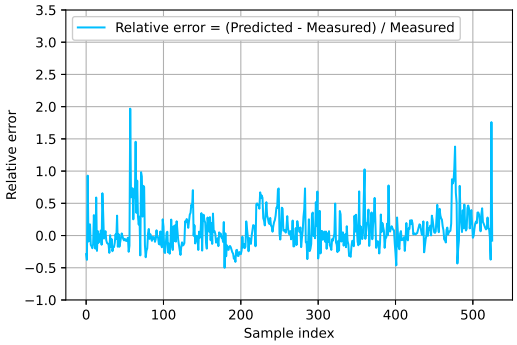
Figure 7.6. The uncertainty quantification on the testing set using GPR.



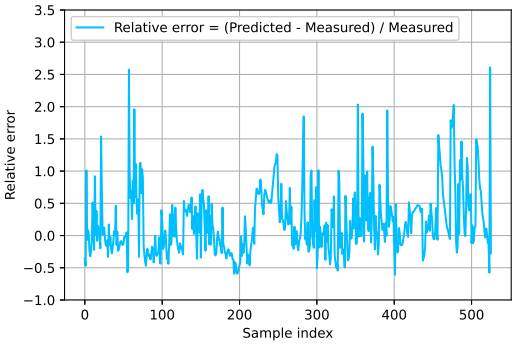
(a) RF regression (k-fold)



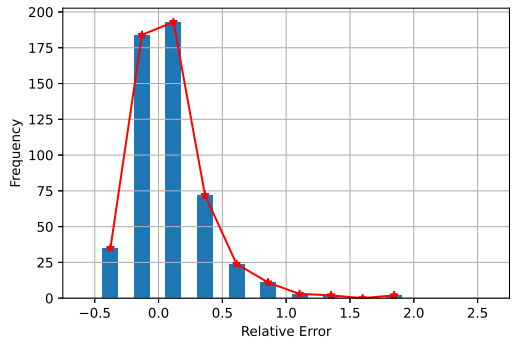
(b) RF regression (group k-fold)



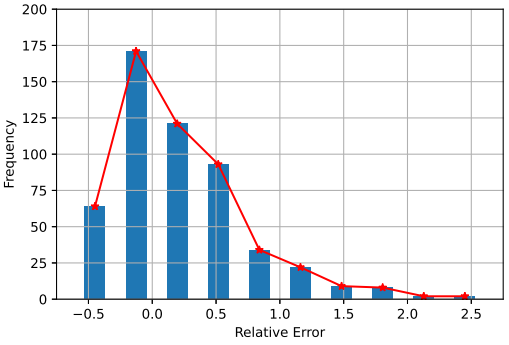
(c) Relative Error of RF (k-fold)



(d) Relative Error of RF (group k-fold)

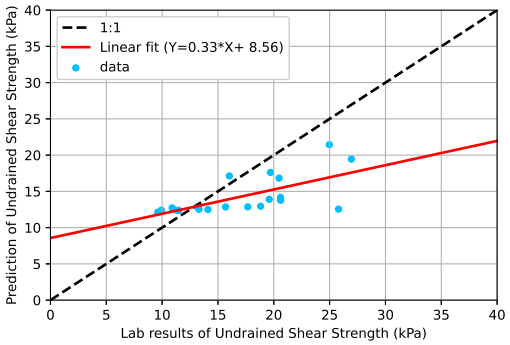


(e) Relative Error Distribution of RF (k-fold)

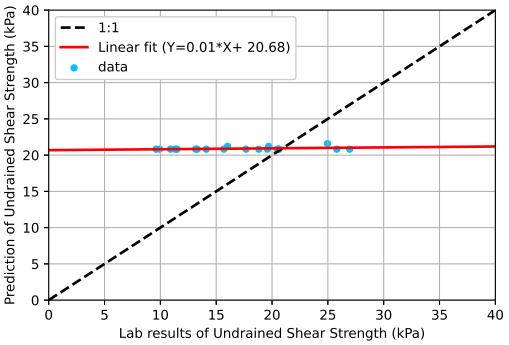


(f) Relative Error Distribution of RF (group k-fold)

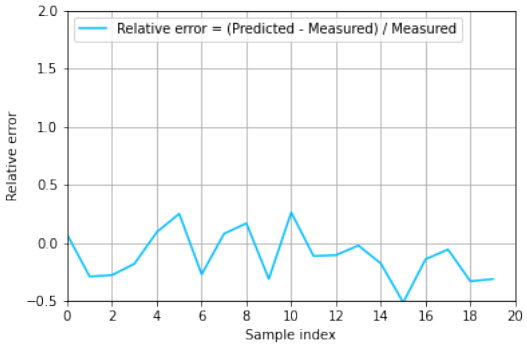
Figure 7.7. Best RF models from k-fold and group k-fold CV on the training set.



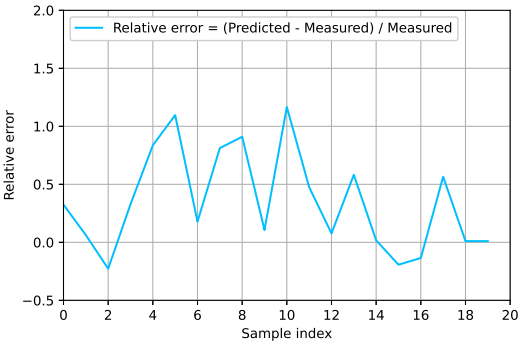
(a) RF regression (k-fold)



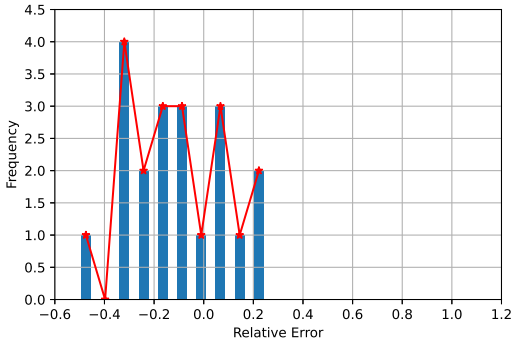
(b) RF regression (group k-fold)



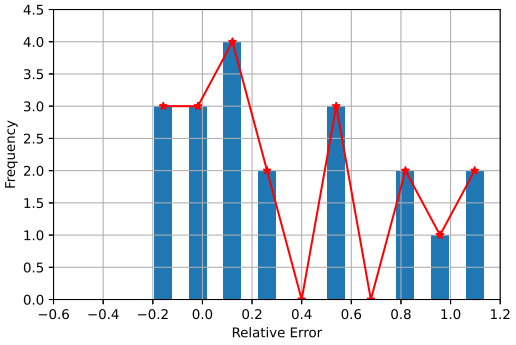
(c) Relative Error of RF (k-fold)



(d) Relative Error of RF (group k-fold)

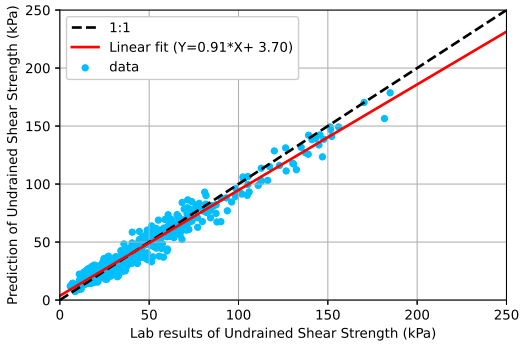


(e) Relative Error Distribution of RF (k-fold)

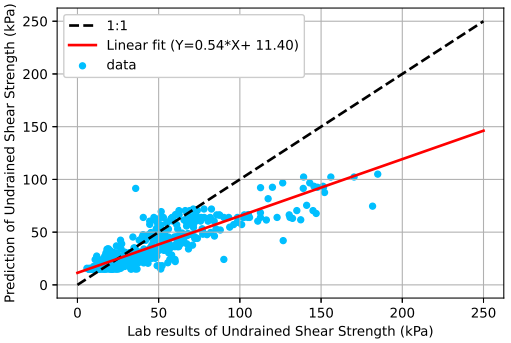


(f) Relative Error Distribution of RF (group k-fold)

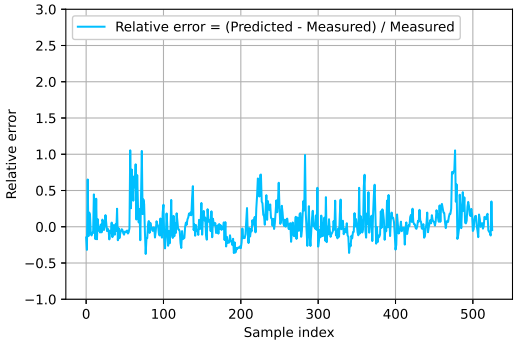
Figure 7.8. Best RF models from k-fold and group k-fold CV on the testing set.



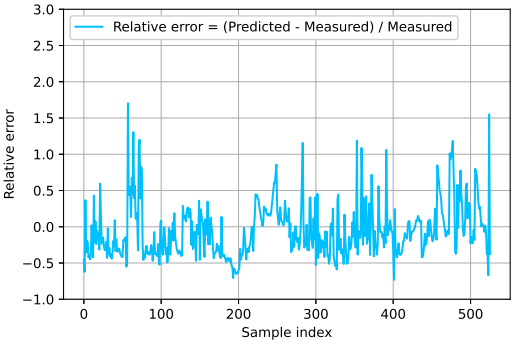
(a) XGBoost regression (k-fold)



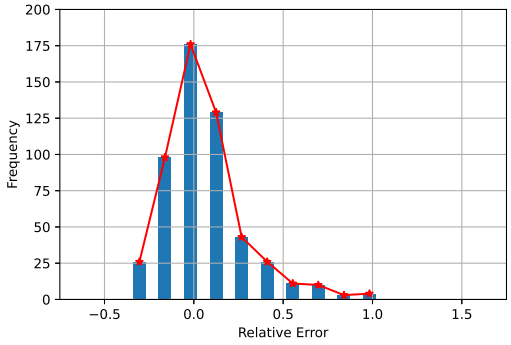
(b) XGBoost regression (group k-fold)



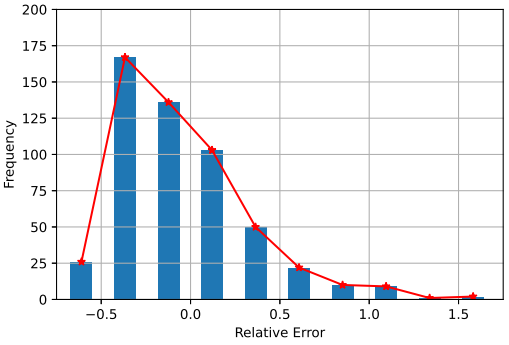
(c) Relative Error of XGBoost (k-fold)



(d) Relative Error of XGBoost (group k-fold)

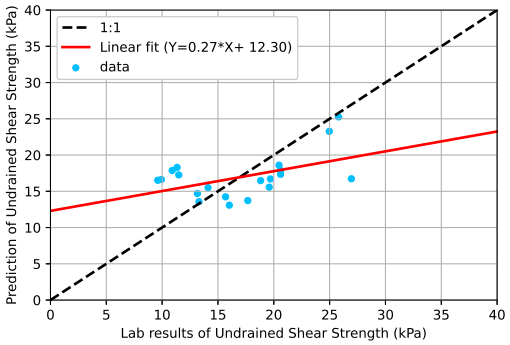


(e) Relative Error Distribution of XGBoost (k-fold)

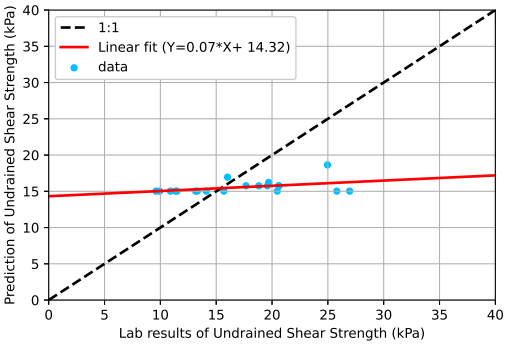


(f) Relative Error Distribution of XGBoost (group k-fold)

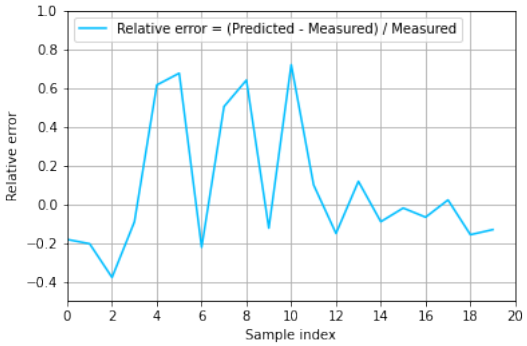
Figure 7.9. Best XGBoost models from k-fold and group k-fold CV on the training set.



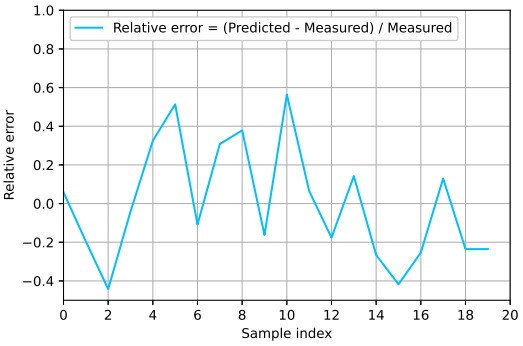
(a) XGBoost regression (k-fold)



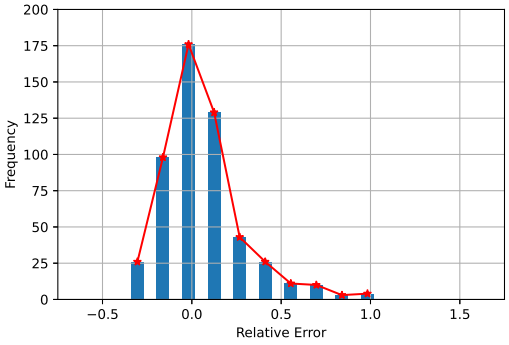
(b) XGBoost regression (group k-fold)



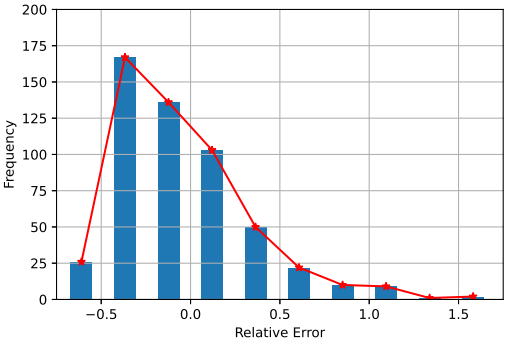
(c) Relative Error of XGBoost (k-fold)



(d) Relative Error of XGBoost (group k-fold)



(e) Relative Error Distribution of XGBoost (k-fold)



(f) Relative Error Distribution of XGBoost (group k-fold)

Figure 7.10. Best XGBoost models from k-fold and group k-fold CV on the testing set.

A summary of the prediction capabilities of the algorithms using k-fold CV and group k-fold are presented in Table 7.5 and Table 7.6 respectively. The GPR results in the two tables are identical since the result of hyperparameter tuning of k-fold CV and group k-fold CV are the same.

Part	Method	R^2	MAE	RMSE	merror(%)	StDerror
Training	ANN	0.79	10.69	14.96	13.37%	0.43
	SVM	0.83	9.85	13.38	14.42%	0.42
	GPR	0.76	11.30	15.81	14.51%	0.43
	RF	0.91	6.92	9.69	8.65%	0.30
	XGBoost	0.96	5.17	6.67	5.87%	0.23
Testing	ANN	-1.85	7.30	8.77	33.00%	0.58
	SVM	-0.31	5.09	5.94	16.68%	0.41
	GPR	-0.38	4.43	6.09	17.55%	0.21
	RF	0.13	3.75	4.85	10.98%	0.21
	XGBoost	0.23	3.73	4.55	7.78%	0.34

Table 7.5. Summary of the prediction capabilities of the algorithms using k-fold CV.

Part	Method	R^2	MAE	RMSE	merror(%)	StDerror
Training	ANN	0.76	11.87	15.88	28.72%	0.50
	SVM	0.76	11.14	15.82	14.02%	0.44
	GPR	0.76	11.30	15.81	14.51%	0.43
	RF	0.71	12.49	17.60	20.46%	0.51
	XGBoost	0.62	13.06	20.04	5.48%	0.37
Testing	ANN	-1.36	6.78	7.98	36.08%	0.54
	SVM	-2.28	7.87	9.40	29.02%	0.61
	GPR	-0.38	4.43	6.09	17.55%	0.21
	RF	-0.53	5.31	6.42	35.02%	0.42
	XGBoost	0.03	4.22	5.11	0.25%	0.29

Table 7.6. Summary of the prediction capabilities of the algorithms using group k-fold CV.

The CV results in the training dataset are presented in Table 7.7.

CV	Method	R^2	StDR^2	MAE	StDMAE	RMSE	StDRMSE
k-fold	ANN	0.76	0.06	10.87	1.64	15.49	2.96
	SVM	0.76	0.06	11.13	1.93	15.32	3.10
	GPR	0.73	0.07	11.74	1.61	16.34	2.84
	RF	0.75	0.08	10.89	1.97	15.70	3.32
	XGBoost	0.76	0.06	10.79	1.58	15.45	2.69
group k-fold	ANN	0.13	0.79	16.44	7.39	21.21	8.29
	SVM	-0.29	1.57	17.76	6.12	22.76	7.88
	GPR	-0.41	1.75	15.89	7.71	20.21	8.95
	RF	-0.95	2.90	17.27	5.27	22.45	7.58
	XGBoost	-0.12	0.78	16.61	6.97	22.03	9.75

Table 7.7. The CV results in the training dataset.

To start with, it is not surprising to see that the result of group k-fold CV is poorer both in the training dataset compared with that of the k-fold CV. However, in the testing dataset, the result of group k-fold CV is also poorer. Therefore, group k-fold CV is not chosen as the CV strategy in this research.

Then, the results can be visualized intuitively with the plots. It is also not surprising to see that the prediction results in the training dataset are overall good. However, it doesn't mean anything since the model is fitted to the training dataset. These results can only be used to determine whether the model is overfitting or not. The results of the testing dataset show how the models perform on unseen data, which has always been crucial information to consider. However, as the testing dataset only consists of 20 samples, which is way too small for the ML problem, the results in the testing dataset can only be considered as a reference. The prediction of RF and XGBoost is better than the others, which can also be verified with the tables.

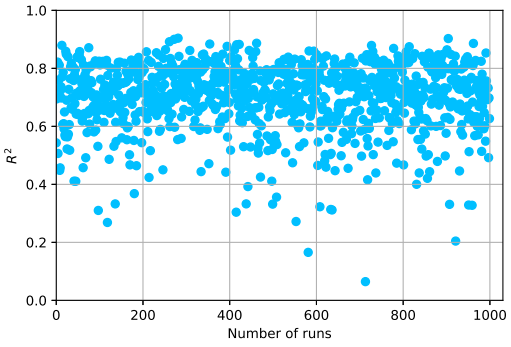
With the information in the tables, the results can be analyzed more objectively. The statistic metrics need to be considered comprehensively since one statistic metric is more representative than the others. For instance, if the R^2 of an algorithm is relatively low but the MAE is completely within the acceptable margin of error, then the algorithm is still practical to use. Table 7.7 contains important information about this research. The CV results in the training dataset show the average capabilities of the algorithms to learn from 90% of the training dataset and then predict on the rest of 10%. The CV results of ANN, SVM and XGBoost are fairly close, showing overall good performance.

Combining the above information, XGBoost is chosen as the most appropriate algorithm for the prediction of undrained shear strength from CPT due to its high accuracy. It is worth mentioning that this is only true with the specific testing dataset used in this study and it cannot be generalized to other datasets. Being a Bayesian method, GPR is chosen as the second option due to its capability of uncertainty quantification and also due to its adaptability to dealing with a small dataset (Lora, 2019). The robustness of these two models is further validated in the validation set in the next section.

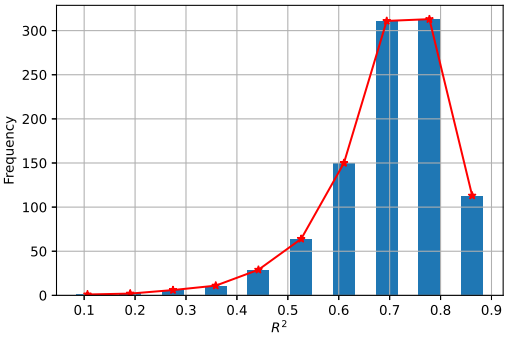
7.4 Monte Carlo analysis results

The robustness of the selected two models needs to be validated from a statistical point of view. As mentioned already in the section 5.4, in each iteration, 90% of the experimental data are randomly selected in order to train and construct the machine learning model. Then the model is tested on the validation set which contains 10% of the experimental data. Therefore a total number of 1000 numerical simulations are carried out for XGBoost and GPR on the validation set, taking into account the random splitting effect in the dataset. This is also called the Monte Carlo simulation.

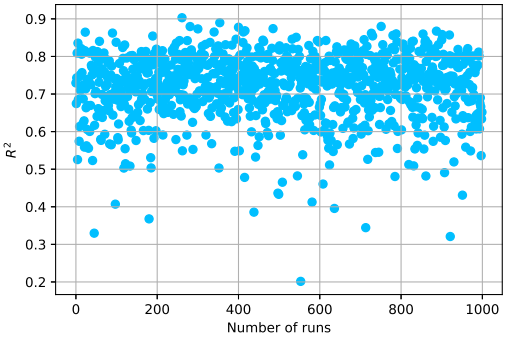
The R^2 values of these simulations in the validation datasets are plotted in Fig. 7.11(a) and 7.11(c), and the corresponding histograms are plotted in Fig. 7.11(b) and 7.11(d). It is observed that the proposed XGBoost model gives R^2 values within the range of 0.06 to 0.90. The most frequent R^2 obtained over 1000 simulations is $R^2 = 0.778$ with a frequency of 313. Besides, the accuracy R^2 values of GPR ranges from 0.20 to 0.90 with the most frequent values of $R^2 = 0.73$ with a frequency of 351. Monte Carlo analysis is also carried out for the ANN, SVM and RF. The performances of the five models for the validation part are summarized together and presented in Table 7.8.



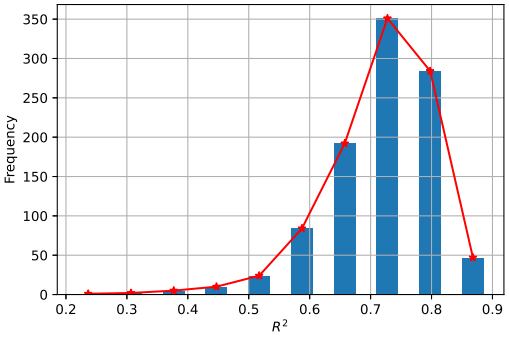
(a) R^2 for 1000 simulations for XGBoost



(b) Histograms of R^2 for XGBoost



(c) R^2 for 1000 simulations for GPR



(d) Histograms of R^2 for GPR

Figure 7.11. Scatter plots and the corresponding histograms of R^2 values for 1000 simulations for XGBoost and GPR.

Part	Method	Mode	Median	Avr. R2	StD. R2
Validation	ANN	0.723	0.759	0.738	0.105
	SVM	0.496	0.518	0.521	0.093
	GPR	0.728	0.732	0.719	0.087
	RF	0.779	0.742	0.717	0.110
	XGBoost	0.778	0.719	0.701	0.113

Table 7.8. Summary of the Monte Carlo simulations.

It can thus be concluded that XGBoost and GPR algorithms have the potential to predict the undrained shear strength of the soil. And they yielded close results with median $R^2 = 0.719$ and 0.732 respectively. ANN and RF are also fairly robust, but it turns out that SVM lacks robustness.

7.5 Sensitivity analysis results

A sensitivity analysis is carried out to evaluate the importance of the input parameters for modelling using partial dependence plots, which is an efficient way to investigate the relationship between inputs and output. Partial dependence plots show the dependence between the target response and a set of input features of interest, marginalizing the values of all other input features. Fig. 7.12 and Fig. 7.13 illustrate the partial dependence of the output on the inputs using RF and XGBoost respectively (The y-axes show the undrained shear strength (kPa)). As shown, the undrained shear strength had an overall positive correlation with the input variables. The variation of shear strength is significant with the cone tip resistance and the excess pore pressure.

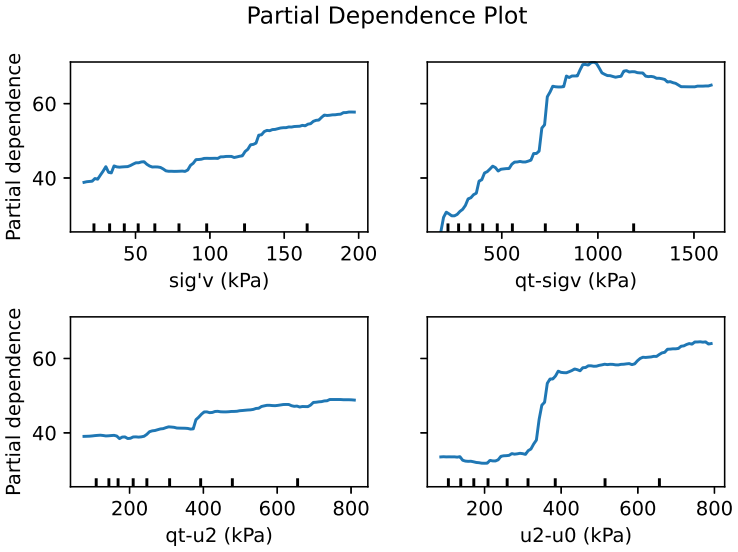


Figure 7.12. Partial dependence plots of the input variables used in this study using RF.

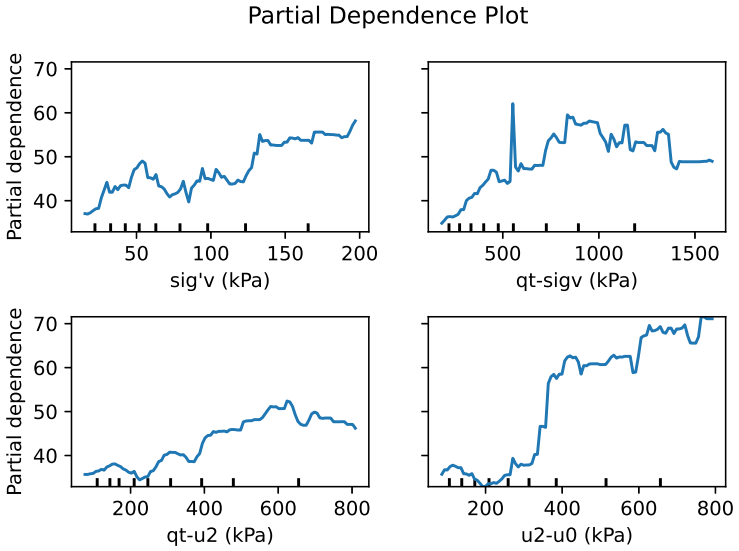


Figure 7.13. Partial dependence plots of the input variables used in this study using XGBoost.

Using RF, the relative importance scores of the inputs to the output are presented in Fig. 7.14. It can be seen that the excess pore pressure is the most significant variable for the shear strength of soil, which achieved an average importance score of 0.433. The cone tip resistance ranked second with an average importance score of 0.387, followed by the effective stress (0.099) and the effective cone tip resistance (0.080). This result agrees well with Fig. 7.12 and Fig. 7.13.

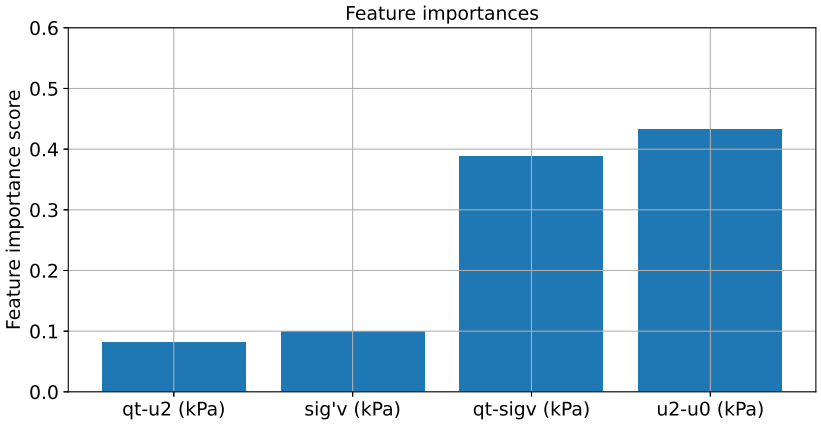


Figure 7.14. Variable importance analysis using random forest.

7.6 Conclusion

This chapter first presents the hyperparameter tuning results of the five algorithms. Each algorithm has been tuned with k-fold CV and group k-fold CV, resulting in two models with different sets of hyperparameters.

Then the results in the training and testing dataset are provided using figures and tables. For each algorithm, the figures first present the prediction results in the training dataset, together

with the corresponding relative error and relative error distribution, with two CV strategies. Next, the same is done in the testing dataset. The prediction capabilities of the algorithms using k-fold CV and group k-fold CV are then summarized in the Table 7.5 and Table 7.6. Finally, the CV results in the training dataset are summarized in Table 7.7. XGBoost is chosen as the most appropriate algorithm for the prediction of undrained shear strength from CPT. GPR as a Bayesian method is chosen as the second option due to its unique ability of uncertainty analysis and its adaptability to dealing with a small dataset.

Next, the robustness of the selected two models is validated from a statistical point of view by applying Monte Carlo analysis. The performance of the XGBoost and GPR is relatively stable, with most of the R^2 values ranging between 0.7 to 0.8, further validating the potential of these two algorithms to predict the undrained shear strength.

At last, a sensitivity analysis is carried out to evaluate the importance of the input parameters for modelling in this study using RF. The partial dependence plots and the variable relative importance score both indicate that the excess pore pressure and the cone tip resistance are the most influential inputs to the undrained shear strength.

Chapter 8

Conclusions

This thesis evaluates the relative performance of five machine learning algorithms, namely the ANN, SVM, GPR, RF and XGBoost to the prediction of undrained shear strength through CPT. The conclusions are summarized as follows:

- The training dataset used in this study consists of 526 data points from 40 different sites in 13 countries. There are four inputs in each sample, namely the effective stress (σ'_v), cone tip resistance ($q_t - \sigma_v$), effective cone tip resistance ($q_t - u_2$) and the excess pore pressure ($u_2 - u_0$). The inputs are all obtained from CPT. The output is the undrained shear strength obtained through laboratory tests. The pair plot shows an overall good positive correlation between the inputs and the output. For ANN, SVM and GPR which are sensitive to the scale of the inputs, feature scaling is conducted to the training dataset to scale the training data from -1 to 1. The training dataset is used for training and validation with the ratio of training and validation defined as 90/10.

- The CV strategy is chosen based on how the algorithms will be used in practice. The validation should be able to mimic how the algorithms will be applied. The group k-fold CV best suits the situation when the sample generation process is known to have a group structure, which means that the assumption that the random variables are independent and identically distributed in ML is broken. The poor performance of models tuned with group k-fold CV on the testing datasets indicates that the data is site-specific in this study. More data is required for a more robust conclusion. In addition, for the hyperparameter tuning, RandomizedSearchCV is chosen to replace grid search CV when tuning more than 3 hyperparameters at the same time for efficiency.

- GPR is a reliable algorithm to use for processing the CPT data that are in close vicinity to the boreholes in order to predict the representative CPT data at the location of the laboratory boreholes. The representative CPT data is used as the input of the testing dataset. The output in the testing dataset is acquired by conducting a CU test on the samples from the boreholes.

- The performance of the algorithm in the training dataset indicates how much the model has learned from the training dataset. If the performance is too poor, then the model is underfitting, which means that the model has not learned enough information from the training dataset. Conversely, if the performance of the model in the training dataset is too good, then the model might be overfitting, which means that it has probably learned too much information from the training dataset. In this case, the model may fail to generalize

in the testing dataset. The performance of models in the testing dataset shows their generalization abilities. However, considering that the testing dataset consists of only 20 samples in this study, which is simply too few, the result in the testing dataset is taken as a reference. The CV results nicely reflect the learning and generalizing abilities of the models in the training dataset, thus being very valuable for reference. It is worth noticing that the statistic metrics need to be taken into account as a whole when evaluating the performance of the models since only one statistic sometimes does not tell the whole story. For instance, when an algorithm has a relatively poor R^2 but with a MAE that is within the margin of error, it is sometimes still practical to use.

- Not surprisingly, the performance of the five algorithms is overall pretty good in the training dataset. This shows that the models have all learned enough information from the training dataset. And as expected, the models obtained with group k-fold CV exhibit a poorer prediction compared with the models obtained with k-fold CV since the models face a much more severe learning task when using group k-fold CV as discussed before. In the testing dataset, the results of the models using group k-fold CV are still relatively poor, thus group k-fold CV is not considered in this study. Nevertheless, this doesn't mean that these models are meaningless considering that the testing dataset consists of only 20 samples, which is simply too few. The models tuned with group k-fold CV might be able to show their strength in another testing dataset in a future study. The results of XGBoost and RF in the testing dataset are relatively better than the others. And the CV results of ANN, SVM and XGBoost are considerably the best. Combining the information in the plots and tables, the results of XGBoost is chosen as the best out of the five algorithms. Being a Bayesian method, GPR involves constructing a prior distribution over functions rather than over parameters and updating this distribution by conditioning on the data. And Lora (2019)'s study suggests that the Bayesian method is adept at dealing with a small dataset. Therefore GPR is chosen as the second option due to its unique ability of uncertainty analysis and its adaptability to dealing with small datasets. The result of RF is slightly worse compared with that of the XGBoost. The CV results of ANN and SVM are pretty good, but they fail to generalize in the testing dataset, thus they are not chosen either. It is worth mentioning that the result of ANN in the testing dataset is in line with the expectations since the ANN has been considered not suitable for learning information from a relatively small dataset.

- The robustness of the XGBoost and GPR is validated by applying Monte Carlo analysis. ANN and RF are also proved to be robust but it turns out that SVM lack robustness. The Monte Carlo resampling method makes it possible to build the distribution of R^2 values on the validation set. This is then used to generate confidence intervals on the parameter estimates, further validating the proposed model.

- In the sensitivity analysis, it is observed that the undrained shear strength increases slowly with the increase of the effective stress and the effective cone tip resistance. The undrained shear strength first increases slowly with the growth of the cone tip resistance, then increases rapidly when the cone tip resistance is around 750kPa, and finally, it levelled off. The undrained shear strength first stabilizes with the growth of excess pore pressure, then increases rapidly when the excess pore pressure is around 300kPa, and finally increases slowly. The relative importance scores can be obtained by applying RF. The ranking of the scores is as follows: the excess pore pressure (0.433), the cone tip resistance (0.387), the effective stress (0.099), and the effective cone tip resistance (0.080). The results are in accordance with the observations from the partial dependence plots.

Chapter 9

Limitations and recommendations

9.1 Limitations

This thesis contributes to the prediction of undrained shear strength using ML techniques. Reviewing the whole research process, there are still many limitations in this study. The limitations are summarized as follows:

- The prediction results should be compared to the results obtained with empirical correlations to reflect the superiority of ML techniques to the empirical analytical methods. The reason why this is not achieved is that it is difficult to find a complete training set that contains all the parameters in the empirical formula.
- Being an important parameter that can be gained from CPT, the friction ratio (R_f), defined as the ratio between sleeve friction and cone resistance, is not included as the input variable in the training dataset as there is no suitable training dataset.
- Even with 526 samples, the training dataset is still too small in a ML context. The fewer samples for training, the more models can fit our data. For instance, in an extreme example which contains only one training example, any model will be able to "explain" it, however simple or complex the model may be. As more samples are added to the training dataset, fewer models will be able to explain them. That way, for a dataset with only 526 samples, we need to be very careful not to be fooled by overfitting. Common measures for dealing with a small dataset include: using simple models, removing the outliers, deleting features that do not contribute to the prediction, etc.
- Containing only 20 samples, the testing dataset is apparently also way too small in a ML context. This makes it more difficult to assess the generalizability of the ML models. Moreover, there are errors both in the inputs and the output of the testing dataset. As for the inputs, the positions of the CPTs are not taken as the inputs for generating the representative CPT data at the location of the boreholes when applying GPR, thus there is still room for improvement in the accuracy of the prediction for the representative CPT data. As for the output, there are certainly errors in the undrained shear strength obtained by the laboratory tests. To be more specific, Table 9.1 shows the effective stresses of the samples obtained by the laboratory tests, the representative effective stresses at the location of the samples and their ratios. For the samples with ratios that are too high (greater than 1.5), the errors between the parameters measured by the laboratory tests and CPTs are considered too high, which requires further analysis. In addition, the soil

heterogeneity of the borehole samples in the testing dataset which has been discussed in de Gast (2020)'s research is not taken into account in this study.

Laboratory	B1007	B1007	B1006	B1006	B1010	B1010	B1010	B1009	B1009	B1009
boreholes	-8	-11	-3	-7	-4	-6	-9	-4	-5	-8
$\sigma'_v(lab)$	32.26	44.19	62.91	30.56	20.67	18.00	42.35	21.98	19.60	43.65
$\sigma'_v(CPT)$	35.00	41.00	18.00	30.75	24.00	28.50	35.00	24.33	29.00	33.67
$\frac{\sigma'_v(lab)}{\sigma'_v(CPT)}$	0.92	1.08	3.50	0.99	0.86	0.63	1.21	0.90	0.68	1.30
Laboratory	B1008	B1008	B1005	B1005	B1001	B1001	B1011	B1011	B1011	B1011
boreholes	-4	-6	-3	-7	-5	-6	-5	-7	-10	-11
$\sigma'_v(lab)$	17.90	36.21	47.40	26.24	45.46	53.60	58.83	28.13	45.21	45.21
$\sigma'_v(CPT)$	24.50	28.25	18.75	31.25	24.20	29.60	24.00	31.00	39.00	42.00
$\frac{\sigma'_v(lab)}{\sigma'_v(CPT)}$	0.73	1.28	2.53	0.84	1.88	1.81	2.45	0.91	1.16	1.08

Table 9.1. The effective stresses of the samples obtained by the laboratory tests and the representative effective stresses at the location of the samples.

9.2 Recommendations

The work of this master thesis promotes the research in the prediction of undrained shear strength through CPT data using various ML techniques. With the advancement of ML techniques and more available datasets, better ML models for the prediction or classification of all kinds of soil properties will be developed. The following recommendations are drawn for future research:

- According to Section 2.4, novel optimization techniques can be applied for the calibration of the hyperparameters in the ML models when dealing with large datasets.
- As the training dataset consists of worldwide data while the data in the testing dataset is from the Netherlands only, a relatively large prediction error is inevitable in this research. In the future study, site-specific training datasets can be established to train and construct ML models belonging to different regions in order to provide more accurate site-specific information.
- Considering the errors in both the inputs and the output of the ML models, it is more convincing to analyze the variables in terms of the confidence intervals rather than specific estimates of the values. Therefore, multivariable error analysis is recommended for future studies. In addition, it is suggested to apply the statistical analysis for more effective analysis and comparison of the ML model performance results.
- Combining the results from multiple models is recommended in future studies in order to get much more accurate predictions. For instance, a final prediction calculated as a weighted average of predictions from various individual models will have significantly lower variance and improved generalizability compared to the predictions from each individual

model. Similarly, combining the predictions from the same model using different values of hyperparameters is also recommended (Koidan, 2019).

- In this research, the CPT data is considered less labour-intensive and more cost-effective to obtain than the laboratory data. In future research, a wider range of data sources that are less labour-intensive and more cost-effective than CPT can be used. For instance, train the ML models on the geophysical data to develop ML models that are capable of characterizing the site conditions.

Appendix A

Dataset

A.1 Training dataset

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Canada	36.75	667.3065	565.95	138.1065	56.154
Canada	36.75	376.32	248.724	164.346	55.6395
Canada	45.07	409.8215	169.4181	285.4734	16.72097
Canada	47.16	422.7894	169.4459	300.5035	36.87912
Canada	51.32	405.1714	162.2225	294.3202	31.20256
Canada	53.4	383.145	155.0202	281.5248	29.2098
Canada	53.4	380.0478	144.18	289.2678	40.1034
Canada	59.64	367.621	136.9931	290.2679	38.28888
Canada	61.73	396.0597	144.2013	313.5884	41.9764
Canada	67.97	410.063	140.562	337.4031	41.12185
Canada	72.13	414.3147	151.4009	335.0439	24.66846
Canada	74.21	417.2086	144.19	347.2286	35.6208
Canada	80.46	511.5647	212.6558	379.3689	67.34502
Canada	82.54	503.6591	205.4421	380.757	27.2382
Canada	95.03	495.3914	180.2719	410.2445	54.07207
Canada	97.11	523.6171	183.8292	436.8979	39.42666
Canada	103.35	557.5733	223.4427	437.4806	59.63295

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Canada	105.43	610.9669	241.5401	474.9622	47.97065
Canada	111.68	648.6374	241.5638	518.7536	50.92608
Canada	120	580.2	299.16	401.04	51.6
Canada	132.49	659.4027	255.9707	536.0545	69.95472
Canada	31.38	823.2857	841.9568	12.7089	20.30286
Canada	32.76	401.2772	416.4779	17.55936	19.0008
Canada	36.21	228.5575	159.6137	105.1538	20.56728
Canada	36.9	216.972	148.707	105.1281	19.9998
Canada	40.69	208.5363	144.0833	105.143	18.51395
Canada	45.52	208.4361	139.1091	114.8925	21.98616
Canada	48.97	210.0813	149.0157	109.9866	24.63191
Canada	49.66	208.4727	148.1358	109.9969	25.3266
Canada	53.45	190.9769	129.5628	114.8641	22.8766
Canada	54.48	208.0591	147.6408	114.8438	31.76184
North Sea	31	279	120.001	189.999	17.763
UK	30	266.01	174	111.81	24.3
UK	30.02	264.9865	167.9919	115.607	21.25416
UK	34.01	267.9648	169.982	115.8041	26.69785
UK	34.96	302.0544	189.0287	129.5618	29.12168
UK	41.03	375.9579	238.015	151.6469	35.40889
UK	41.96	381.0807	243.0323	151.2658	31.30216
UK	47	401.004	250.98	160.787	29.845
UK	47.02	404.9833	247.9835	166.7329	32.86698
UK	50	420.95	267	162.5	23.25
UK	58.99	496.9908	341.965	157.5033	41.17502
UK	59	497.016	342.023	157.412	33.099
UK	74.04	583.9535	414.9942	163.3322	46.34904
UK	74.99	587.9966	341.9544	238.6932	45.59392

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
UK	77.03	597.9069	351.9501	238.6389	43.21383
UK	96.95	733.039	378.0081	335.7379	52.9347
UK	97.98	740.0429	390.0584	330.4865	57.3183
UK	97.98	740.0429	389.9604	330.3886	59.96376
UK	100.03	760.0279	434.0302	305.7917	61.61848
UK	99.96	764.0942	408.0367	334.2662	62.37504
UK	116.02	893.9341	484.0354	380.4296	70.7722
UK	116.02	893.9341	484.0354	380.4296	71.58434
UK	116.02	893.9341	484.0354	380.1975	69.72802
UK	117.98	903.0189	478.9988	393.2273	67.60254
UK	233	2017.081	1959.996	290.085	137.47
UK	315	2060.1	1809.99	565.11	141.12
Norway	42.24	229.7856	193.1635	78.86208	5.82912
Norway	45.11	273.2313	232.4518	85.88944	10.96173
Norway	50.84	276.9255	221.6116	106.1031	13.42176
Norway	58	319.638	226.142	151.496	13.224
Norway	60.86	327.5485	221.287	167.1216	18.07542
Norway	64.44	362.7328	245.3231	181.8497	13.2102
Norway	68.02	299.9682	181.2733	186.7149	14.14816
Norway	72.32	468.3443	348.4378	192.2266	11.86048
Norway	75.9	369.0258	246.2955	198.6303	15.8631
Norway	80.91	399.129	264.8184	215.2206	14.88744
Norway	82.34	390.5386	370.283	102.678	20.9967
Norway	84.49	359.2515	234.4598	209.2817	18.67229
Norway	85.92	411.8146	322.9733	174.8472	21.99552
Norway	90.21	286.958	211.2718	165.806	34.09938
Norway	98.81	339.5112	245.5429	192.7783	11.65958
Norway	103.82	311.3562	214.0768	201.0993	10.9011

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Norway	117.42	483.3007	398.2886	202.4321	22.19238
Norway	122.43	476.7424	337.1722	262.0002	18.60936
Norway	126.73	427.2068	314.6706	239.2662	19.13623
Norway	40	297	170	167	24.76
USA	24.86	324.1247	176.6303	172.3544	32.21856
USA	127.15	587.0516	563.1474	151.0542	63.8293
USA	132.88	372.8613	235.5962	270.0122	56.07536
USA	138.62	502.9134	320.0736	321.3212	54.47766
USA	140.53	519.1178	333.1966	326.4512	51.01239
USA	146.27	536.3721	345.7823	336.8598	60.11697
USA	150.09	460.7763	280.218	330.6483	57.18429
USA	152	456	286.216	321.784	60.648
USA	158.7	432.1401	277.4076	313.4325	61.4169
USA	165.39	484.7581	308.7831	341.365	62.68281
USA	167.3	520.1357	319.543	367.8927	63.9086
USA	175.9	472.2915	296.2156	351.9759	62.6204
USA	182.6	465.63	316.4458	331.7842	70.8488
USA	184.51	472.3456	307.2092	349.4619	65.31654
USA	192.16	550.7306	369.3315	373.559	53.42048
USA	199.8	1002.796	726.4728	476.1234	102.4974
USA	208.41	888.035	577.2957	519.1493	105.2471
Canada	133.42	904.1873	397.5916	639.8823	98.33054
Canada	138.38	830.5568	398.8112	570.1256	96.1741
Canada	173.36	761.0504	434.2668	500.1436	71.42432
Canada	177.87	978.4629	424.7536	731.5793	130.3787
Canada	230.67	908.8398	377.6068	761.903	132.4046
Canada	40.09	272.171	126.3637	185.8973	14.95357
Canada	40.09	279.6678	125.1209	194.5969	16.75762

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Canada	40.95	274.6107	127.6412	187.9196	26.33085
Canada	41.82	244.187	103.8809	182.1679	25.59384
Canada	45.27	249.5735	118.879	175.9645	21.36744
Canada	47	296.57	143.914	199.656	19.317
Canada	49.6	361.8816	167.6976	243.784	27.8752
Canada	56.51	293.5695	102.6222	247.4573	34.35808
Canada	57.37	313.0107	101.3728	269.0079	30.46347
Canada	59.1	337.3428	125.1147	271.3281	42.4929
Canada	65.15	407.7087	165.1553	307.7035	39.41575
Canada	66.88	417.7994	171.4134	313.199	25.68192
Canada	67.74	420.0557	183.9141	303.8816	40.44078
Canada	76.38	433.151	147.6425	361.8884	57.36138
Canada	78.11	450.6947	168.9519	359.8528	32.33754
Canada	86.75	579.4033	231.449	434.6175	54.73925
Canada	87.62	624.2925	263.9991	447.9134	41.88236
Canada	98.85	575.7024	212.7252	461.8272	56.04795
UK	16.18	351.2354	268.41	99.00542	17.0699
UK	21.9	344.5089	235.7097	130.6992	20.1042
UK	27.82	275.1676	173.0682	129.9194	22.36728
UK	31.54	276.2589	175.4255	132.3734	24.9166
UK	37.95	315.3266	189.4844	163.7922	27.28605
UK	48.22	358.1782	216.3631	190.035	33.07892
UK	52.35	395.3996	249.1337	198.6683	34.8651
UK	70.23	459.6554	288.2942	241.5912	42.48915
UK	75.9	474.6027	285.9912	264.5115	44.7051
UK	80.88	528.2273	319.3142	289.793	47.55744
UK	90.77	566.6771	338.5721	318.875	55.73278
Sweden	20.48	284.2624	119.4598	164.7821	13.53728

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Sweden	23.14	314.0792	133.5872	180.492	15.22612
Sweden	26.1	316.4103	135.9027	180.5076	15.6861
Sweden	28.17	311.7856	126.0608	185.7248	15.2118
Sweden	32.02	307.2319	121.4839	185.748	16.26616
Sweden	34.38	304.8818	113.9009	190.9465	15.05844
Sweden	37.34	311.789	115.5673	196.2217	14.41324
Sweden	39.41	309.4867	108.0228	201.4639	14.38465
Sweden	46.22	320.9979	106.4447	214.5532	16.08456
Sweden	51.84	348.5722	113.0112	235.5091	17.72928
Sweden	56.87	392.2324	122.6117	269.6207	19.3358
Haga	70	737.03	266	541.03	60.83
Haga	88	957	382.976	662.024	55.44
Haga	118	767	231.044	653.956	67.496
Haga	135	788.94	189	734.94	64.395
Haga	24	736.008	430.008	330	33.336
Haga	65	1094.99	689.975	470.015	72.085
Haga	109	1669.989	1180.034	598.955	116.303
China	61.6	1214.444	1106.706	169.3384	41.1488
China	66.43	1250.279	828.3157	488.3934	66.82858
China	71.26	1964.353	1374.178	661.4353	71.33126
China	76.09	1356.837	848.0991	584.8277	44.51265
China	80.92	1150.925	631.4188	600.4264	111.0222
China	84.78	1376.912	810.6664	651.0256	48.7485
China	88.65	787.6553	360.5396	515.7657	40.4244
China	91.54	779.4631	373.117	497.8861	43.84766
China	94.44	757.6921	359.1553	492.9768	46.08672
China	99.27	782.3469	358.3647	523.2522	63.23499
China	103.14	829.9676	393.4791	539.7316	93.75426

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
China	106.04	901.5521	422.9936	584.5985	51.4294
China	109.9	1095.593	598.7352	606.7579	65.6103
China	112.8	1830.067	1221.737	721.1304	73.32
China	116.66	1948.222	1142.218	922.6639	51.56372
China	120.53	1245.798	632.7825	733.5456	87.50478
China	123.43	1400.437	721.8186	802.0481	60.60413
China	126.32	1701.53	1107.321	720.5293	87.41344
China	130.19	1932.54	1090.862	971.8684	62.4912
China	134.05	1632.863	822.6649	944.2482	71.3146
China	136.95	1792.949	966.0453	963.8541	77.92455
China	151.44	824.8937	419.3374	556.9963	76.78008
China	156.27	867.2985	410.9901	612.5784	101.1067
China	158.2	944.9286	434.8918	668.2368	105.203
China	175.59	1015.788	462.3285	729.0497	102.7202
USA	82.85	2943.329	1113.67	1799.502	139.0223
USA	94.66	3336.292	1262.575	2039.26	126.3711
USA	100.3	3431.965	1299.888	2095.568	143.0278
UK	152	2632.944	1024.936	1672.152	139.232
UK	172	2744.948	1073.452	1735.48	156.004
UK	192	3206.016	1250.496	2019.456	170.304
UK	213	3707.052	1442.649	2330.433	184.884
Canada	12.53	357.4057	136.1385	9.71075	16.77767
Canada	20.48	164.4339	68.05504	19.41504	23.87968
Canada	24.87	214.7027	89.55687	53.39589	25.31766
Canada	28.32	274.6757	114.611	150.4925	28.26336
Canada	35.65	354.7175	150.443	228.1957	33.5823
Canada	39.43	345.6828	150.386	247.5415	37.41907
Canada	42.97	386.0425	168.3135	252.4058	40.77853

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Canada	46.68	406.5361	179.0645	267.0096	43.08564
Canada	49.88	418.4433	186.2519	271.846	43.8944
Canada	53.58	399.9211	182.6542	281.5629	45.75732
Canada	57.29	400.9154	186.2498	276.7107	47.95173
Canada	60.74	441.3976	204.1471	291.2483	48.04534
Canada	64.28	452.5312	211.2884	300.959	48.8528
Malaysia	13.5	212.895	136.053	73.6965	32.886
Malaysia	18.89	201.0841	122.105	68.53292	33.22751
Malaysia	22.75	219.2873	113.7728	86.177	51.324
Malaysia	26.61	224.9077	108.0632	95.02431	35.57757
Malaysia	31.62	257.6081	111.6186	116.4565	36.6792
Malaysia	35.48	275.786	122.2641	118.2903	49.06884
Malaysia	40.1	305.7224	130.2849	135.2974	52.13
Malaysia	42.03	319.5121	142.2295	135.925	50.436
Malaysia	50.9	376.7109	158.9607	172.551	60.2147
Malaysia	56.68	405.4887	166.9793	186.3638	60.02412
Malaysia	62.08	425.248	157.9936	214.176	56.86528
Canada	64.72	751.1403	342.4335	473.4268	68.53848
Canada	67.15	699.0987	320.977	445.3388	84.3404
Canada	72.84	767.1509	406.8114	433.1795	79.7598
Canada	82.17	603.0456	307.8088	377.4068	72.22743
Canada	93.54	558.9015	248.3487	404.0928	65.29092
Singapore	44.82	224.8171	179.5937	49.302	16.00074
Singapore	49.39	220.2794	140.6133	69.6399	19.16332
Singapore	69.67	211.2394	99.83711	110.9146	25.42955
Singapore	87.56	316.8796	223.3656	105.5974	41.76612
Singapore	110.24	327.964	231.063	115.5315	34.39488
Singapore	115.03	345.6652	247.0844	111.349	38.42002

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Singapore	135.31	347.882	269.5375	100.5353	50.0647
Singapore	152.63	656.309	439.4218	235.2028	52.50472
Singapore	170.52	728.2909	438.2364	320.9186	53.88432
Singapore	188.42	687.9214	389.6526	341.6055	50.11972
Singapore	193.2	694.3608	348.9192	382.7292	49.4592
Singapore	205.74	726.8794	380.8247	385.1453	53.4924
Singapore	197.41	746.4072	353.1665	408.0465	76.9899
Sweden	27.54	254.8827	126.3811	128.5016	14.76144
Sweden	29.87	268.8599	135.9981	132.8618	13.17267
Sweden	31.2	286.2912	146.9208	139.3704	12.0432
Sweden	32.52	293.1678	147.3156	145.8197	12.12996
Sweden	33.85	296.6614	144.3364	152.325	12.62605
Sweden	35.18	317.5699	158.697	158.8377	13.12214
Sweden	37.5	359.2875	187.4625	171.825	14.4375
Sweden	38.83	387.1351	202.3043	184.8308	15.68732
Sweden	40.16	383.6886	188.0291	195.6595	16.50576
Sweden	41.82	373.2435	173.2184	200.0251	17.60622
Sweden	43.15	373.2475	168.9323	204.3584	18.42505
Sweden	44.48	373.2762	168.935	204.3411	19.61568
Sweden	45.8	423.0088	195.6576	227.3512	20.4268
Sweden	48.13	449.7267	204.3119	245.4149	20.16647
Sweden	49.46	476.7449	220.6905	256.0544	21.0205
Sweden	51.12	487.1736	220.8384	266.3352	21.726
Sweden	52.44	494.3519	219.4614	274.8905	22.65408
Sweden	53.77	470.595	193.6258	276.9693	23.06733
Sweden	55.1	468.1296	190.7011	277.4285	23.5828
Sweden	57.76	492.4618	206.0299	286.4318	23.104
Sweden	59.08	504.6614	208.3752	296.3453	23.0412

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Sweden	61.74	534.8536	223.931	310.9226	23.58468
Sweden	64.06	540.0258	217.804	322.2218	22.99754
Sweden	66.39	538.954	217.3609	321.5932	21.11202
Sweden	69.04	531.608	210.572	321.036	19.40024
Sweden	71.7	523.41	209.5074	313.9026	17.925
Sweden	74.69	508.7883	202.0365	306.8265	16.87994
Sweden	77.34	527.3815	217.016	310.2881	16.93746
Sweden	80.33	560.0608	225.4863	334.6548	17.51194
Sweden	83.32	582.0735	241.7946	340.1956	18.16376
Norway	45.9	152.0208	67.2894	130.6314	12.9438
Norway	48.57	163.778	70.32936	142.0187	15.00813
Norway	51.58	191.7744	101.0452	142.3092	17.58878
Norway	78.61	327.1748	173.5709	232.2139	17.37281
Norway	114.32	470.7698	233.3271	351.877	26.8652
Venezuela	209.25	1492.999	485.46	631.7258	57.54375
Venezuela	219.17	1528.272	488.9683	657.7292	58.51839
Venezuela	231.66	1583.628	504.5555	691.0418	62.5482
Venezuela	250.4	1647.882	514.3216	735.9256	69.1104
Venezuela	260.32	1699.89	529.2306	760.9154	66.3816
Venezuela	164.57	1374.489	556.5757	458.8212	47.06702
Venezuela	188.98	1486.517	600.3895	515.1595	52.9144
Venezuela	200.9	1542.711	616.1603	547.2516	58.261
Venezuela	217.93	1619.438	625.4591	603.6661	62.32798
Venezuela	228.72	1671.486	639.9586	635.8416	60.38208
Brazil	11.36	139.9098	64.3544	77.816	11.97344
Brazil	21.21	244.0423	103.823	137.5681	17.56188
Brazil	31.39	232.6313	72.60507	149.3536	14.37662
Brazil	79.19	501.827	428.2595	152.7575	42.04989

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Brazil	81.89	490.2754	356.5491	215.6983	40.86311
Brazil	89.99	577.1059	427.3625	239.7334	47.42473
Brazil	92.68	538.3781	401.0264	230.0318	32.25264
Brazil	105.28	505.7651	377.7446	233.3005	42.6384
Brazil	115.17	582.1844	453.885	243.4694	35.24202
Brazil	131.36	467.6416	368.0707	230.7995	34.8104
Brazil	139.46	544.3124	429.9552	253.8172	45.74288
Brazil	144.86	555.1035	432.8417	267.1218	50.12156
Brazil	159.25	728.8873	850.5543	37.42375	49.686
Brazil	20.35	270.8178	183.1907	107.9771	21.978
Brazil	22.27	228.2675	146.158	104.3795	20.08754
Brazil	25.16	203.8966	150.9348	78.1218	16.63076
Brazil	29.01	191.2339	126.9768	93.26715	12.38727
Brazil	30.94	189.3528	136.2598	84.03304	8.75602
Brazil	33.82	189.1891	110.4561	112.553	7.30512
Brazil	36.71	188.8362	114.1681	111.3781	15.12452
Brazil	42.97	224.7761	142.1018	125.6443	22.12955
Brazil	45.86	159.5469	57.04984	148.3571	21.04974
Brazil	53.07	260.043	134.957	178.2091	24.67755
Brazil	59.81	272.2551	160.8889	171.1762	17.58414
Brazil	67.03	326.6372	198.0737	195.5935	30.23053
Brazil	73.28	351.8906	201.8864	223.2842	32.38976
Brazil	79.06	440.6014	266.8275	252.7548	42.05992
Singapore	13.9	233.8119	215.8114	31.9144	10.7725
Singapore	25.48	521.8814	395.0674	152.294	17.27544
Singapore	120.49	1001.272	723.6629	398.099	72.41449
Singapore	142.11	966.9164	655.2692	453.7572	81.71325
Singapore	163.38	1114.578	702.8608	575.0976	90.18576

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Singapore	19.94	224.5643	109.331	135.1733	13.4595
Singapore	31.32	339.5714	159.7007	211.1594	21.17232
Singapore	42.7	370.0382	183.0549	229.6833	14.091
Singapore	62.34	300.6658	149.1173	213.9509	43.45098
Singapore	123.24	570.2315	297.1316	396.3398	77.14824
Singapore	13.84	254.393	232.941	35.292	10.33848
Singapore	27.99	521.3417	437.9875	111.3442	42.51681
Singapore	40.57	599.0566	464.8105	174.8161	30.79263
Singapore	59.76	647.2606	471.5064	235.5739	35.01936
Singapore	78.63	726.4626	510.8591	294.2335	52.76073
Singapore	97.49	772.2183	522.4489	347.2594	61.4187
Singapore	118.72	873.5418	685.608	306.6538	53.424
Singapore	17.39	222.4181	146.6673	93.12345	20.64193
Singapore	29.44	387.4304	282.0058	134.8941	25.64224
Singapore	49.51	350.4318	302.6051	97.33666	39.95457
Singapore	69.58	471.2653	395.4231	145.4222	29.2236
Singapore	88.64	1098.25	941.3568	245.5328	61.78208
Singapore	109.72	850.6592	563.1928	397.1864	61.11404
Singapore	129.79	791.3296	578.9932	342.1264	73.85051
Singapore	149.86	1054.265	687.7075	516.4176	104.902
Singapore	169.93	910.8248	635.0284	445.7264	130.8461
Singapore	190	1186.74	722.76	653.79	112.67
Singapore	13.42	450.496	366.7283	97.18764	24.19626
Singapore	24.6	623.61	486.9078	161.3268	26.1006
Singapore	35.78	707.3706	547.0762	196.0744	34.56348
Singapore	45.84	707.2195	543.3874	209.6263	24.2952
Singapore	152.23	962.2458	600.2429	514.0807	105.9521
Singapore	174.83	994.0834	545.1199	623.7934	122.9055

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Singapore	195.97	1148.972	664.5343	680.2119	146.9775
Singapore	218.2	1228.466	696.9308	749.7352	135.0658
Singapore	239.71	1322.72	749.8129	812.6169	181.7002
Singapore	13.9	403.9757	327.9288	89.9469	16.1101
Singapore	25.48	627.9036	488.0694	165.3397	18.93164
Singapore	37.06	679.8286	558.0865	158.8392	44.73142
Singapore	48.64	692.3418	569.2339	171.6992	39.73888
Singapore	60.22	724.0251	590.2764	193.9686	40.88938
Singapore	133.7	1010.505	699.9195	444.2851	101.8794
Singapore	156.77	1120.278	724.1206	552.9278	93.12138
Singapore	180.84	1188.3	732.402	636.5568	98.5578
Singapore	211.41	960.2242	513.7263	657.9079	145.0273
Singapore	14.14	252.1869	160.8425	105.4844	12.88154
Singapore	25.92	422.1072	285.2755	162.7517	26.8272
Singapore	37.7	477.6213	308.4614	206.8599	37.0968
Singapore	49.48	497.5709	307.3203	239.7306	57.99056
Singapore	63.49	719.0243	589.1237	193.3905	68.37873
Singapore	14.51	306.9445	190.3277	131.1414	28.97647
Singapore	25.04	483.5725	349.333	159.2544	28.24512
Singapore	36.42	501.7583	351.6715	186.5068	36.34716
Singapore	47.8	501.5654	424.3684	124.997	36.6626
Singapore	14.53	386.5125	289.147	111.8955	26.11041
Singapore	32.65	406.6231	302.0452	137.228	21.05925
Singapore	50.62	507.7186	429.9663	128.4229	31.58688
Singapore	20.97	246.6282	128.9865	138.6117	12.47715
Singapore	40.15	428.3202	246.6816	221.7886	25.3748
Singapore	60.91	1033.947	618.1147	476.7426	84.17762
Singapore	93.95	758.0826	347.5211	504.5115	68.0198

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Singapore	16.42	190.0451	152.8538	53.6113	6.86356
Singapore	46.71	756.6553	474.5736	328.7917	53.20269
Singapore	71.28	790.2101	455.9069	405.5832	53.88768
Singapore	95.85	1321.196	778.302	638.6486	66.42405
Singapore	120.01	1432.679	921.0768	631.6126	73.92616
Singapore	144.99	1574.736	808.0293	911.6971	82.49931
Singapore	12.72	140.874	106.5936	47.0004	7.19952
Singapore	28.29	537.1705	334.1049	231.3556	15.50292
Singapore	86.09	981.7704	571.4654	496.3949	80.40806
Singapore	13.94	154.957	84.95036	83.94668	10.77562
Singapore	31.01	464.6538	245.5372	250.1267	27.66092
Singapore	48.08	544.9387	286.076	306.9427	28.94416
Singapore	19.35	535.7822	273.1833	281.9489	17.8407
Singapore	44.23	496.526	209.8714	330.8404	32.50905
Singapore	70.43	902.8422	400.7467	572.5255	67.19022
Singapore	23.75	732.8538	417.5963	339.0075	29.165
Singapore	8.67	322.8708	158.3662	173.1746	13.19574
Singapore	27.24	468.3373	213.7523	281.825	39.47076
Singapore	79.98	1092.927	632.4019	540.4249	58.86528
Singapore	7.27	132.9029	109.6607	30.51219	8.74581
Singapore	21.8	240.9336	153.4502	109.2834	11.0526
Singapore	70.59	633.2629	323.3022	380.4801	46.30704
Singapore	94.75	794.0998	427.3225	461.5273	87.928
Singapore	8.25	228.03	134.1203	102.1598	15.96375
Singapore	19.63	244.3346	139.5497	124.4149	19.13925
Singapore	31.01	257.7551	149.5922	139.1729	21.0868
Singapore	42.39	274.9839	153.4518	163.9221	29.58822
Singapore	55.8	361.1376	206.3484	210.5892	28.3464

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Singapore	8.69	305.4796	253.8436	60.32598	11.96613
Singapore	21.56	432.7308	360.7204	93.5704	21.64624
Singapore	44.63	510.835	422.78	132.685	26.86726
Singapore	56.31	522.782	435.5015	143.5905	29.67537
Singapore	61.7	570.9718	457.3204	175.3514	32.6393
Singapore	8.25	239.8275	155.9168	92.16075	17.6055
Singapore	19.63	289.0518	189.881	118.8008	19.55148
Singapore	31.01	319.527	213.2558	137.3123	24.37386
Singapore	42.39	349.2936	237.7655	153.9181	19.75374
Singapore	87.56	698.4661	421.8641	364.162	57.00156
Singapore	7.97	469.2577	348.2412	128.9785	11.7956
Singapore	19.35	588.5303	432.279	175.5819	21.53655
Singapore	30.73	639.1533	463.9001	205.9525	30.88365
Singapore	42.11	700.8367	518.5004	224.4463	32.0036
Singapore	53.49	822.8367	598.3926	277.934	35.57085
Singapore	64.87	907.8557	656.2898	316.4359	38.72739
Singapore	76.25	905.3925	651.0988	330.62	41.9375
Singapore	87.91	938.1755	669.3467	356.7388	53.80092
Singapore	7.27	238.9213	155.0546	91.13672	16.1394
Singapore	17.13	292.7174	185.5008	124.3467	26.32881
Singapore	29.64	465.348	220.492	274.5257	90.13524
Singapore	46.02	1044.01	658.2701	431.7596	48.5511
Singapore	62.4	534.456	197.184	399.672	55.4112
Canada	90.71	1369.812	564.4883	896.0334	112.6618
Canada	94.67	1333.427	450.8185	977.2784	152.1347
Canada	101.73	1548.229	231.2323	1418.625	117.3964
Canada	105.86	1491.462	599.6969	997.6246	120.0452
Canada	114.65	1703.355	795.7857	1022.219	147.7839

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Canada	119.3	1631.07	689.9119	1060.338	151.511
Canada	122.05	1603.737	631.1206	1094.666	151.22
Canada	124.98	1627.615	611.5271	1141.067	145.6017
Canada	15.57	266.3404	138.5419	117.6158	21.36204
Canada	24.37	344.08	168.9572	155.6756	21.6893
Canada	38.05	376.1243	184.6947	174.269	24.5042
Canada	54.67	489.0778	233.9876	225.4044	28.64708
Canada	58.58	506.3069	239.5922	237.7176	30.52018
Canada	77.16	582.095	280.7081	267.2051	36.11088
Canada	81.07	584.1094	279.8536	272.9627	37.53541
Canada	98.66	791.2532	368.1991	379.7423	44.19968
Canada	119.19	843.746	394.7573	405.1268	50.77494
Canada	141.67	1018.041	487.0615	479.553	58.22637
USA	60.02	700.3134	257.8459	440.1267	55.15838
USA	84.08	807.4202	273.5963	527.0134	52.55
USA	95.18	838.3454	272.8811	556.0416	52.15864
USA	95.18	838.3454	272.8811	556.0416	52.15864
USA	114.62	858.733	248.4962	599.1187	51.46438
USA	118.32	873.0833	245.2774	614.1991	50.40432
USA	122.02	887.9395	250.8731	622.7901	50.0282
USA	131.28	893.4917	235.7789	642.2218	49.23
USA	155.34	907.9623	253.0489	633.0105	48.46608
USA	178.48	950.049	266.8276	655.7355	48.90352
USA	190.51	993.8907	292.8139	673.0718	49.5326
USA	196.98	1017.993	289.3636	695.7334	50.62386
USA	227.52	1195.618	369.265	786.9917	55.05984
USA	227.52	1195.618	369.265	785.8541	55.05984
USA	247.88	1265.427	373.803	846.2623	58.00392

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
USA	52.61	310.5042	152.2007	210.9135	35.40653
USA	57.5	326.5425	146.855	237.1875	35.305
USA	61.58	338.6284	152.5337	247.7363	33.5611
USA	64.84	365.957	146.279	284.5179	31.18804
USA	68.92	397.6684	171.5419	294.9776	25.63824
USA	79.52	516.9595	259.3942	337.0853	37.93104
USA	85.23	556.6371	283.8159	358.0512	44.57529
USA	90.13	570.9736	260.9264	400.1772	46.59721
USA	100.73	607.0997	260.3871	447.4427	42.10514
USA	107.26	670.2677	293.2488	484.2789	52.12836
USA	121.94	748.8335	423.3757	447.3979	45.23974
USA	126.83	783.0484	462.4222	447.4562	50.09785
USA	130.91	805.0965	488.5561	447.4504	56.02948
USA	141.52	855.913	497.4428	499.9902	72.31672
USA	148.04	862.185	489.1242	521.1008	75.79648
USA	150.49	906.7023	509.8601	547.3321	76.14794
USA	161.09	969.4396	577.8298	552.5387	68.46325
USA	167.62	1047.625	662.6019	552.6431	51.1241
USA	183.12	1104.946	688.165	599.9011	69.5856
USA	188.01	1106.627	694.697	599.9399	78.02415
Sweden	13.97	174.3875	102.5258	85.84565	8.13054
Sweden	16.18	146.8011	81.91934	81.0618	8.72102
Sweden	18.38	139.7431	71.33278	86.79036	9.17162
Sweden	22.06	139.4192	71.12144	90.3357	10.01524
Sweden	25	151.55	81.225	95.35	10.3
Sweden	28.31	149.7316	71.39782	106.6438	10.95597
Sweden	31.99	150.481	67.65885	114.8121	12.70003
Sweden	36.03	167.0351	75.87918	127.2219	13.51125

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Sweden	37.87	184.7299	85.01815	137.5817	14.3906
Sweden	40.07	204.2368	96.76905	147.5778	15.42695
Sweden	43.38	213.82	104.2855	152.9579	16.65792
Sweden	46.32	241.4662	119.274	168.5122	18.25008
Sweden	50.37	257.6929	127.5368	180.5765	19.24134
Sweden	54.41	266.1737	131.509	189.0748	20.24052
Sweden	58.09	293.0641	143.0176	208.1365	21.55139
Sweden	62.5	310.375	157.3125	215.5625	22.8125
Sweden	10.43	270.9818	189.68	81.29142	7.46788
Sweden	11.28	266.4787	162.6125	103.8662	7.7832
Sweden	11.91	261.8294	158.0338	103.7957	7.64622
Sweden	14.04	250.7404	139.3891	111.3372	7.28676
Sweden	16.38	261.8834	140.524	121.3758	6.8796
Sweden	19.57	277.483	138.5752	138.9274	9.88285
Sweden	22.77	304.3894	150.3503	154.0391	12.20472
North Sea	29	136.996	107.996	58	22.852
North Sea	62	264.988	157.976	169.012	28.892
North Sea	96	465.024	263.04	297.984	44.736
North Sea	136	1167.968	680.952	623.016	40.528
North Sea	237	1112.952	678.057	671.895	71.574
Norway	14.72	116.7002	54.93504	76.48512	9.58272
Norway	20.38	142.7823	76.91412	86.24816	11.12748
Norway	26.04	142.2305	60.43884	107.8316	8.463
Norway	33.97	161.969	60.43263	135.5063	10.83643
Norway	40.76	226.8294	93.42192	174.1675	12.0242
Norway	46.42	219.0096	82.3955	182.9876	15.87564
Norway	56.61	269.0673	87.91533	237.762	20.71926
Norway	71.32	333.8489	115.3958	289.8445	23.96352

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Norway	78.11	385.9415	159.3444	304.7852	21.24592
Norway	86.04	427.7048	109.8731	403.8718	24.00516
Norway	95.09	497.5109	148.3404	444.2605	26.53011
Norway	101.89	547.1493	153.8539	495.2873	38.00497
Norway	113.21	649.9386	236.2693	526.8793	39.84992
Norway	117.73	571.3437	225.2175	463.8562	34.1417
Norway	126.79	934.4423	362.6194	698.6129	44.50329
Norway	136.98	1112.689	472.444	777.2245	85.74948
Norway	155.09	836.4004	335.1495	656.3409	61.72582
Norway	165.28	900.776	406.5888	659.4672	81.8136
Norway	174.33	961.7786	395.5548	740.5538	81.06345
Norway	213.95	1114.68	439.4533	888.9623	73.81275
Norway	319.35	1211.933	478.067	1053.216	112.7306
Sweden	14.91	153.2897	17.23596	150.9638	8.3496
Sweden	18.36	158.2816	13.91688	162.7247	8.55576
Sweden	21.8	172.4816	25.506	168.7756	8.7854
Sweden	24.67	174.5649	17.46636	181.7686	9.71998
Sweden	28.3	178.9692	17.1215	190.1477	11.1502
Sweden	31.93	206.0443	28.76893	209.2054	12.00568
Sweden	35.18	211.7836	30.74732	216.2163	12.34818
Sweden	39.2	205.9176	24.4216	220.696	13.8768
Sweden	43.4	233.926	38.7128	238.6132	15.7108
Sweden	47.04	259.3315	57.05952	249.312	16.98144
Sweden	50.48	255.4793	50.7324	255.2269	18.57664
Sweden	55.26	279.9472	69.40656	265.8006	21.38562
Sweden	61.19	308.7647	88.7255	281.2292	23.19101
Sweden	67.5	363.8925	130.0725	301.32	22.815
Sweden	74.79	321.6718	98.64801	297.8138	22.96053

Country	σ'_v **(kPa)**	$q_t - \sigma_v$ **(kPa)**	$q_t - u_2$ **(kPa)**	$u_2 - u_0$ **(kPa)**	S_u **(kPa)**
Italy	38.28	734.8994	283.004	490.2137	72.19608
Italy	42.37	825.1134	317.4784	549.9626	82.02832
Italy	54.18	553.9363	222.5714	385.5449	126.6728
Italy	85.99	2124.813	930.7558	1264.311	35.85783
Canada	221	708.968	474.929	455.039	85.306

A.2 Testing dataset

Laboratory samples	σ'_v(kPa)	$q_t - \sigma_v$(kPa)	$q_t - u_2$(kPa)	$u_2 - u_0$(kPa)	S_u(kPa)
B1007-8	35.00	194.00	266.07	-38.07	16.02
B1007-11	41.00	81.00	154.44	-32.44	19.59
B1006-3	18.00	210.75	224.01	4.24	26.95
B1006-7	30.75	154.25	187.68	-3.18	15.68
B1010-4	24.00	113.50	117.90	19.60	11.34
B1010-6	28.50	101.50	88.04	41.46	9.93
B1010-9	35.00	137.00	100.04	71.96	17.67
B1009-4	24.33	118.66	128.68	14.65	11.49
B1009-5	29.00	130.42	121.08	38.67	10.90
B1009-8	33.67	132.67	105.80	61.20	18.82
B1008-4	24.50	95.00	85.41	34.09	9.61
B1008-6	28.25	153.63	126.53	55.60	14.11
B1005-3	18.75	222.83	247.42	-6.09	19.70
B1005-7	31.25	145.75	177.20	-1.20	13.16
B1001-5	24.20	197.40	219.19	3.01	20.47
B1001-6	29.60	138.20	142.95	24.85	25.80
B1011-5	24.00	306.25	361.17	-30.92	24.97
B1011-7	31.00	155.00	206.75	-20.75	13.30
B1011-10	39.00	70.25	114.38	-5.13	20.61
B1011-11	42.00	80.00	112.08	9.17	20.61

Appendix B

Simple Examples for Machine Learning Algorithms

B.1 Artificial Neural Network

In this simple example (Mazur, 2022), we're going to use a neural network with two inputs, one hidden layer with two hidden neurons, two output neurons. Additionally, the hidden and output neurons will include a bias. Here, the bias can be thought of as analogous to the role of a constant in a linear function, whereby the line is effectively transposed by the constant value. The basic structure of this neural network is presented in Fig. B.1.

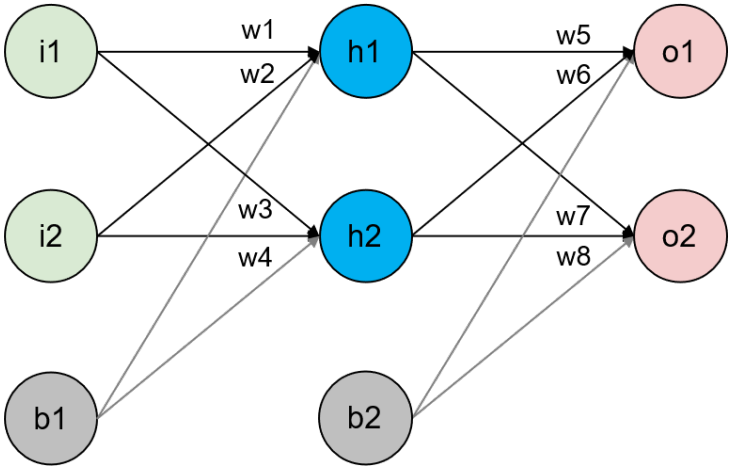


Figure B.1. Basic structure of a MLP with two inputs, one hidden layer and two outputs.

The initial weights, the biases, and training inputs/outputs are given randomly as presented in Fig. B.2. We start with a single training set: given inputs 0.05 and 0.10, we need the neural network to come up with output 0.01 and 0.99.

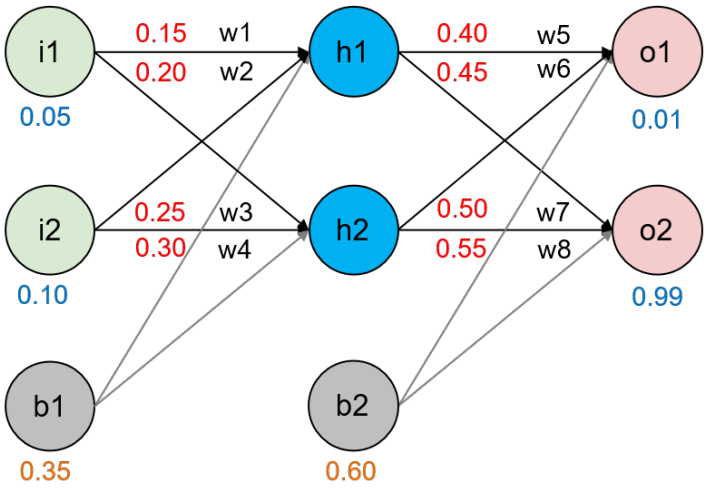


Figure B.2. Initial settings of the MLP.

B.1.1 The Forward Pass

To begin with, the inputs 0.05 and 0.10 are fed forward through the network to see the current prediction with the given initial weights and biases. Then, we figure out the total net input to each hidden layer neuron. Next, we squash it using an activation function. Finally, we repeat the process with the output layer neurons. As mentioned above, the purpose of the activation function is to introduce non-linearity into the network as a neuron network without an activation function is just a linear combination of inputs and a bias, no matter how many layers it had. The activation function used here is logistic function as presented in Fig. B.3 where input values over the entire real number range are transformed to values in the range $[0, 1]$.

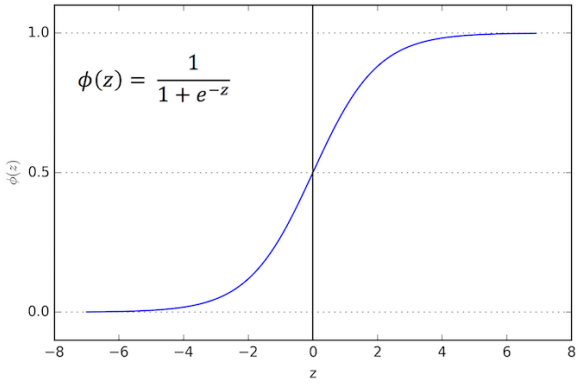


Figure B.3. Logistic function.

The computation of total net input for h1 are as follow:

$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$

$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

We then squash it using the logistic function to get the output of h1:

$$out_{h1} = \frac{1}{1 + e^{-net_{h1}}} = \frac{1}{1 + e^{-0.3775}} = 0.593269992$$

Carrying out the same process for h2 and then for the output layer neurons, using the output from the hidden layer neurons as inputs:

$$out_{h2} = 0.596884378, \quad out_{o1} = 0.75136507, \quad out_{o2} = 0.772928465$$

Now the error for each output neuron can be calculated using the squared error function and then summed up to get the total error:

$$E_{\text{total}} = \sum \frac{1}{2}(\text{target} - \text{output})^2$$

Since the target output for o1 is 0.01 but the neural network's output is 0.75136507:

$$E_{o1} = \frac{1}{2}\left(\text{target}_{o1} - \text{out}_{o1}\right)^2 = \frac{1}{2}(0.010 - 0.75136507)^2 = 0.274811083$$

The same is done with o2: $E_{o2} = 0.023560026$ The total error for the neural network is the sum of these two errors:

$$E_{\text{total}} = E_{o1} + E_{o2} = 0.274811083 + 0.023560026 = 0.298371109$$

B.1.2 The Backwards Pass

After the calculation of the total error in this first iteration, back-propagation algorithm is then applied to adjust the weights of neural network, so that they cause the actual output to be closer the target output, thereby minimizing the error for each output neuron and the network as a whole.

Starting with w5, in order to know how much a change in w5 affects the total error, $\frac{\partial E_{\text{total}}}{\partial w_5}$ is calculated by applying the following chain rule. And a visualization of the chain rule is given in Fig. B.4.

$$\frac{\partial E_{\text{total}}}{\partial w_5} = \frac{\partial E_{\text{total}}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

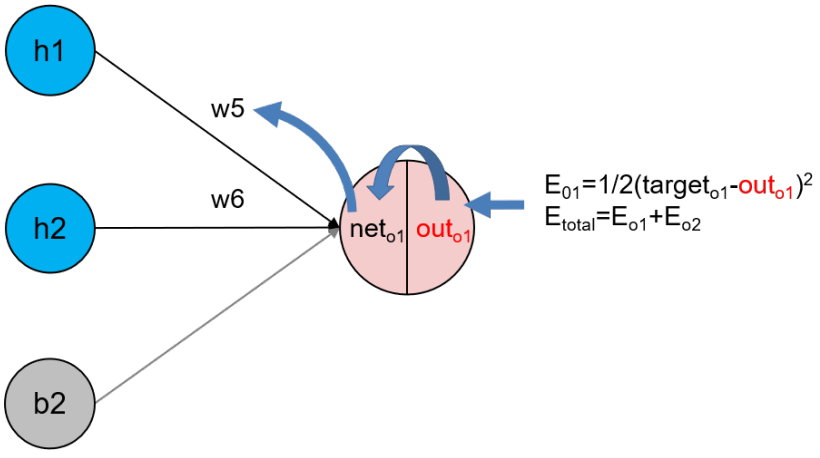


Figure B.4. Visualization of updating the weights in the output layer.

Then, each piece in this equation is figured out step by step as follow, starting with $\frac{\partial E_{\text{total}}}{\partial out_{o1}}$:

$$E_{\text{total}} = \frac{1}{2}\left(\text{target}_{o1} - \text{out}_{o1}\right)^2 + \frac{1}{2}\left(\text{target}_{o2} - \text{out}_{o2}\right)^2$$
$$\frac{\partial E_{\text{total}}}{\partial \text{out}_{o1}} = 2 * \frac{1}{2}\left(\text{target}_{o1} - \text{out}_{o1}\right)^{2-1} * (-1) + 0$$
$$\frac{\partial E_{\text{total}}}{\partial \text{out}_{o1}} = -\left(\text{target}_{o1} - \text{out}_{o1}\right) = -(0.010 - 0.75136507) = 0.74136507$$

Next $\frac{\partial net_{o1}}{\partial w_5}$:

$$\text{out}_{o1} = \frac{1}{1 + e^{-net_{o1}}}$$

$$\frac{\partial_{\text{out}o_{o1}}}{\text{net}_{o1}} = \text{out}_{o1}\left(1 - \text{out}_{o1}\right) = 0.75136507(1 - 0.75136507) = 0.186815602$$

Finally $\frac{\partial net_{o1}}{\partial w_5}$:

$$\text{net}_{o1} = w_5 * \text{out}_{h1} + w_6 * \text{out}_{h2} + b_2 * 1$$
$$\frac{\partial \text{net}_{o1}}{\partial w_5} = 1 * \text{out}_{h1} * w_5^{(1-1)} + 0 + 0 = out_{h1} = 0.593269992$$

Putting them all together:

$$\frac{\partial E_{\text{total}}}{\partial w_5} = \frac{\partial E_{\text{total}}}{\partial \text{out}_{\text{o1}}} * \frac{\partial\ \text{out}_{o1}}{\partial \text{net}_{\text{o1}}} * \frac{\partial \text{net}_{o1}}{\partial w_5}$$
$$\frac{\partial E_{\text{total}}}{\partial w_5} = 0.74136507 * 0.186815602 * 0.593269992 = 0.082167041$$

To decrease the error, gradient descent is then applied by subtracting this value from the current weight (optionally multiplied by some learning rate, eta, which is set to 0.5 in this example):

$$w_5^+ = w_5 - \eta * \frac{\partial E_{\text{total}}}{\partial w_5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

This process is repeated to get the new weights w_6, w_7, and w_8:

$$w_6^+ = 0.408666186, \quad w_7^+ = 0.511301270, \quad w_8^+ = 0.561370121$$

The actual updates in the neural network are performed after we have the new weights leading into the hidden layer neurons. That is to say, we use the original weights, not the updated weights, when we continue the back-propagation algorithm below.

Next, we'll continue the backwards pass by calculating new values for w_1, w_2, w_3, and w_4.

We're going to use a similar process as we did for the output layer, but slightly different to account for the fact that the output of each hidden layer neuron contributes to the output (and therefore error) of multiple output neurons. We know that out_{h1} affects both out_{o1} and out_{o2} therefore when calculating $\frac{\partial E_{total}}{\partial out_{h1}}$, we need to take into consideration the effect of w_1 on the both output neurons:

$$\frac{\partial E_{\text{total}}}{\partial w_1} = \frac{\partial E_{\text{total}}}{\partial \text{out}_{h1}} * \frac{\partial \text{out}_{h1}}{\partial net_{h1}} * \frac{\partial \text{net}_{h1}}{\partial w_1}$$

in which,

$$\frac{\partial E_{\text{total}}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$

Visually:

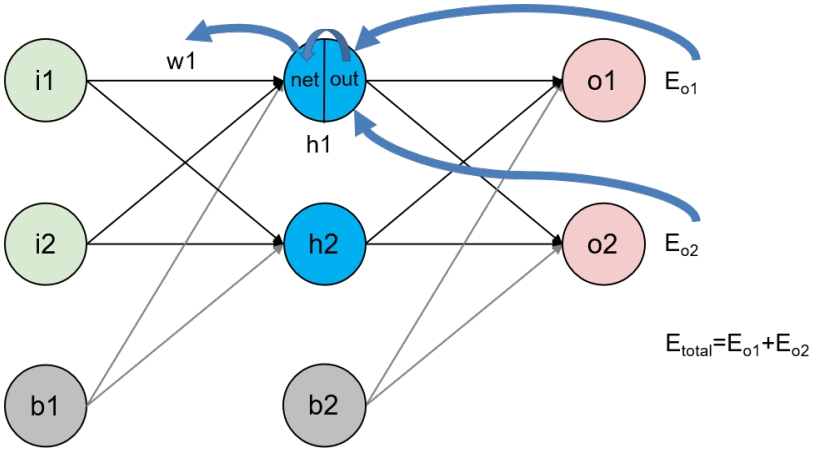


Figure B.5. Visualization of updating the weights in the hidden layer.

Starting with $\frac{\partial E_{o1}}{\partial out_{h1}}$:

$$\frac{\partial E_{o1}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial out_{h1}}$$

We can then calculate $\frac{\partial E_{o1}}{\partial net_{o1}}$ using values we calculated earlier:

$$\frac{\partial E_{o1}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{h1}} = 0.74136507 * 0.186815602 = 0.138498562$$

And $\frac{\partial net_{o1}}{\partial out_{h1}}$ is equal to w_5:

$$\text{net}_{o1} = w_5 * \text{out}_{h1} + w_6 * \text{out}_{h2} + b_2 * 1$$
$$\frac{\partial \text{net}_{o1}}{\partial \text{out}_{h1}} = w_5 = 0.40$$

Plugging them in:

$$\frac{\partial E_{o1}}{\partial \text{out}_{h1}} = \frac{\partial E_{o1}}{\partial \text{net}_{o1}} * \frac{\partial \text{net}_{o1}}{\text{out}_{h1}} = 0.138498562 * 0.40 = 0.055399425$$

Following the same process for $\frac{\partial E_{o2}}{\partial out_{h1}}$, we get:

$$\frac{\partial E_{o2}}{\partial \text{out}_{h1}} = -0.019049119$$

Therefore:

$$\frac{\partial E_{\text{total}}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}} = 0.055399425 + (-0.019049119) = 0.036350306$$

Now that we have $\frac{\partial E_{total}}{\partial out_{h1}}$, we need to figure out $\frac{\partial out_{h1}}{\partial net_{h1}}$ and then $\frac{\partial net_{h1}}{\partial w}$ for each weight:

$$\text{out}_{h1} = \frac{1}{1 + e^{-\text{net}_{h1}}}$$

$$\frac{\text{out}_{h1}}{\partial \text{net}_{h1}} = \text{out}_{h1}\,(1 - \text{out}_{h1}) = 0.59326999(1 - 0.59326999) = 0.241300709$$

We calculate the partial derivative of the total net input to h_1 with respect to w_1 the same as we did for the output neuron:

$$\text{net}_{h1} = w_1 * i_1 + w_3 * i_2 + b_1 * 1$$

$$\frac{\partial net_{h1}}{\partial w_1} = i_1 = 0.05$$

Putting it all together:

$$\frac{\partial E_{\text{total}}}{\partial w_1} = \frac{\partial E_{\text{total}}}{\partial \text{out}_{h1}} * \frac{\partial \text{out}_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\frac{\partial E_{\text{total}}}{\partial w_1} = 0.036350306 * 0.241300709 * 0.05 = 0.000438568$$

We can now update w_1:

$$w_1^+ = w_1 - \eta * \frac{\partial E_{\text{total}}}{\partial w_1} = 0.15 - 0.5 * 0.000438568 = 0.149780716$$

Repeating this for w_2, w_3, and w_4

$$w_2^+ = 0.19956143$$

$$w_3^+ = 0.24975114$$

$$w_4^+ = 0.29950229$$

Finally, all the weights have been updated. The original error on the network was 0.299, When we fed forward the 0.05 and 0.1 inputs. After this first round of back-propagation, the total error is now down to 0.291. It might not seem like much, but after repeating this process 10,000 times, for example, the error plummets to 0.0000351085. At this point, when we feed forward 0.05 and 0.1, the two outputs neurons generate 0.015912196 (vs 0.01 target) and 0.984065734 (vs 0.99 target).

B.2 Random Forest

A simple example of how a decision tree is built is firstly given below (R, 2021). It shows how CART works in regression with one input feature. CART in regression cases uses least squares as the optimization method. Intuitively, splits are chosen to minimize the residual sum of squares (RSS) between the observation and the mean in each node. Mathematically, we can write residual as follow:

$$\varepsilon_i = y_i - \hat{y}_i \tag{B.1}$$

Mathematically, we can write RSS as follow:

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{B.2}$$

$$RSS = \varepsilon_1^2 + \varepsilon_2^2 + .. + \varepsilon_n^2 \tag{B.3}$$

In order to find out the "best" split, we must minimize the RSS. In this simple example, a simulation using a "dummy" dataset is given as follows.

	Price ($)	Review Scores Rating
0	10.0	10
1	11.0	10
2	12.0	10
3	13.0	10
4	14.0	13
5	15.0	20
6	17.5	35
7	18.0	44
8	18.5	52
9	19.0	55
10	21.0	80
11	22.0	83
12	23.0	80
13	24.0	83
14	25.0	85
15	28.0	100
16	29.0	100
17	30.0	100
18	31.0	100
19	31.0	100

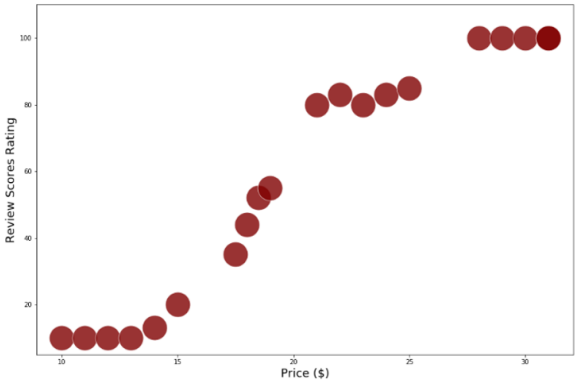


Figure B.6. The flowchart of random forest for regression (R, 2021).

First, we calculate RSS by split into two regions, starting with index 0.

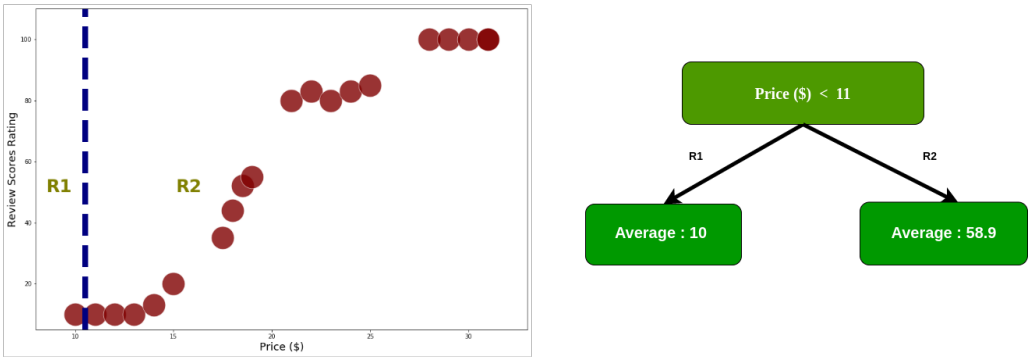


Figure B.7. Split the data within index 0 (R, 2021).

The data is already split into two regions. We use Eq. B.2 to calculate the RSS by adding up the squared residual for every data in both regions.

$$RSS = (10-10)^2 + (10-58.9)^2 + (10-58.9)^2 \ldots (10-58.9)^2 = 21139.78$$

Then, the same is done with all the other splits within each index. This process continues until the calculation of RSS in the last index ends. The final result is shown below in Fig. B.8. It's clear to see that price with a threshold of 19 has the smallest RSS, in R1 there are 10 data within prices less than 19, so we'll split the data in R1.

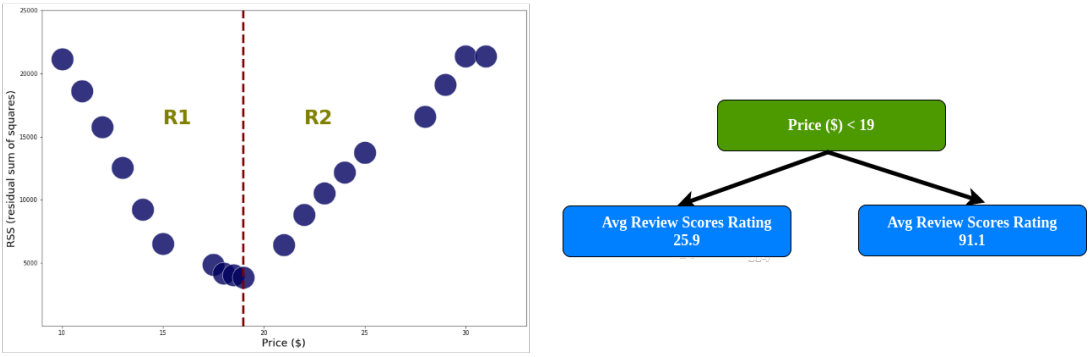


Figure B.8. RSS with respect to the different split of prices (R, 2021).

Next, the same is done on all the other branches. Finally, the end result of a tree, in this case, is shown in Fig. B.9.

In the case of splitting the dataset with multiple input features, the process remains the same as above. That is, we start by calculating the minimum RSS for every input feature as is done above. Then compare them to find the lowest RSS. Finally, the input feature with the lowest RSS is chosen to split the data, using the threshold that has the minimum RSS among all the splits.

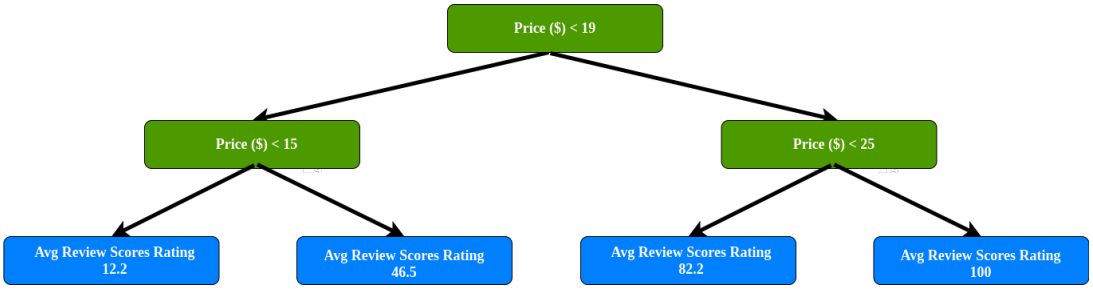


Figure B.9. The final result of the decision tree (R, 2021).

After all the trained trees are constructed, they are combined in the aggregating process to give a final prediction by averaging the results from all constructed trees whenever a test sample enters this RF. The effectiveness of bootstrapping and bagging lies in that the decision trees are trained on various random samples and their results can vary greatly, while their errors, the MSE, for example, are mutually compensated in the aggregating process. In this way, the variance of the trained regressor is decreased, making the model less overfitted.

B.3 XGBoost

In this example, we are building gradient boosting regression trees step by step using the sample shown in Fig. B.10, which has a nonlinear relationship between x and y to intuitively understand how it works.

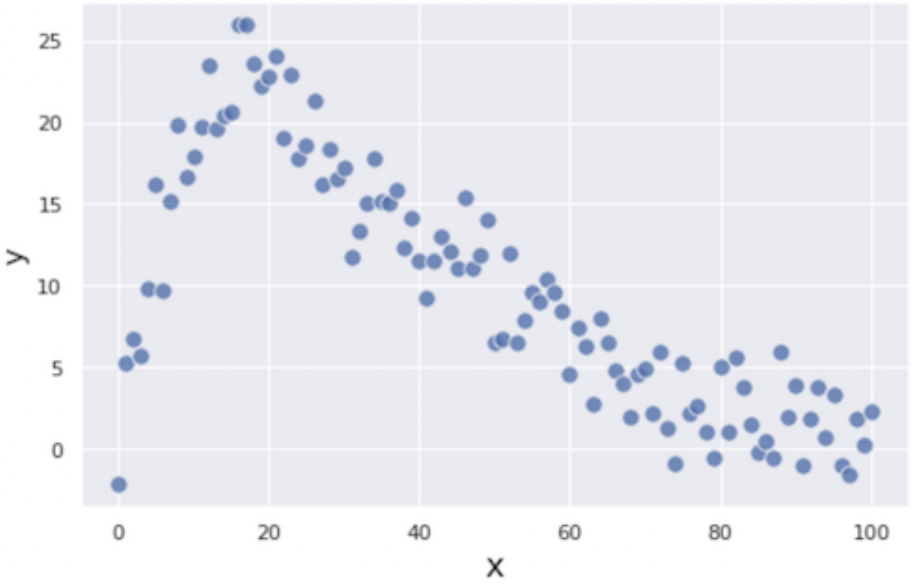


Figure B.10. Sample for a regression problem (Masui, 2022).

The first step is making a very naive prediction on the target y (see Fig. B.11 for example), followed by adding more weak models to it to improve our predictions. The resulting residual, i.e. prediction residuals (shown as the vertical blue lines in Fig. B.12), are minimised using the procedure below to make an improved prediction.

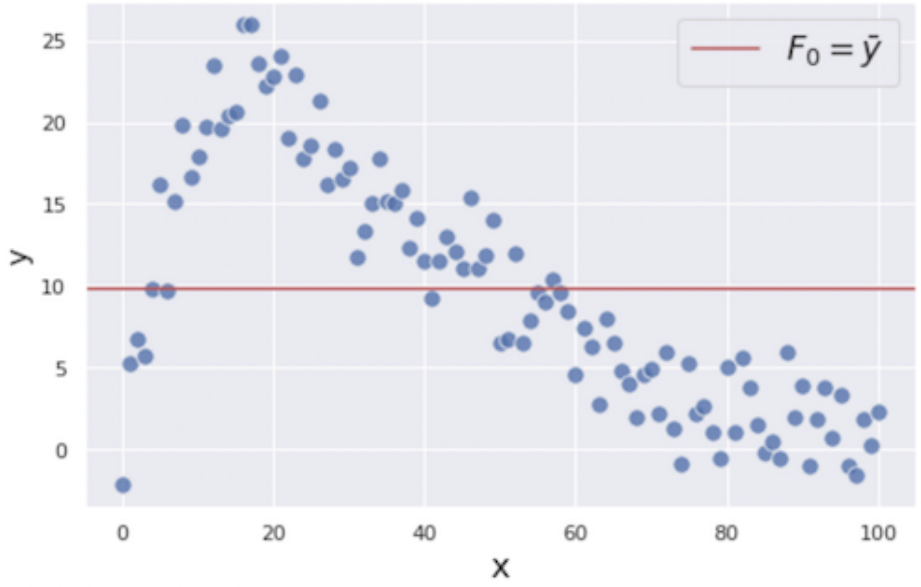


Figure B.11. Initial prediction: F_0 = mean(y) (Masui, 2022).

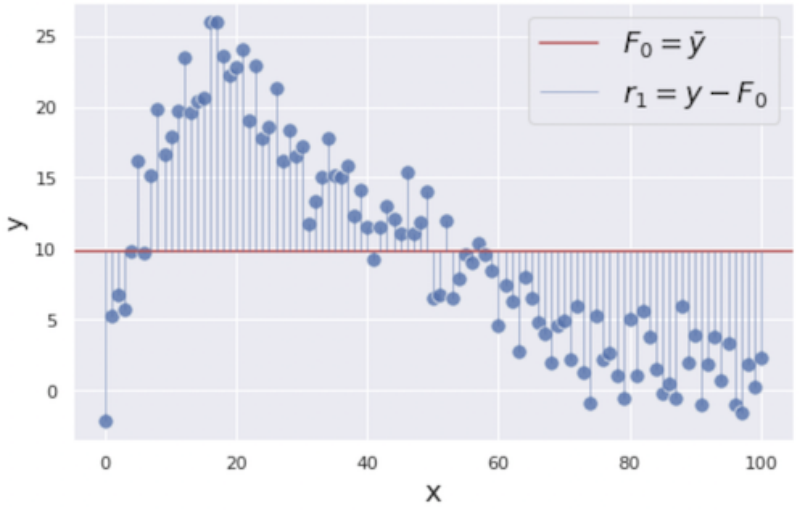


Figure B.12. The residuals of the initial prediction r_1 (Masui, 2022).

To minimize these residuals, we are building a regression tree model with x as its feature and the residuals r_1 = y - mean(y) as its target. The reasoning behind that is if we can find some patterns between x and r_1 by building the additional weak model, we can reduce the residuals by utilizing it. To simplify the demonstration, we are building very simple trees each that only has one split and two terminal nodes. Note that gradient boosting trees usually have a little deeper trees such as ones with 8 to 32 terminal nodes. As shown in Fig. B.13, we are creating the first tree predicting the residuals with two different values $\gamma 1 = \{6.0, -5.9\}$(γ denotes the prediction).

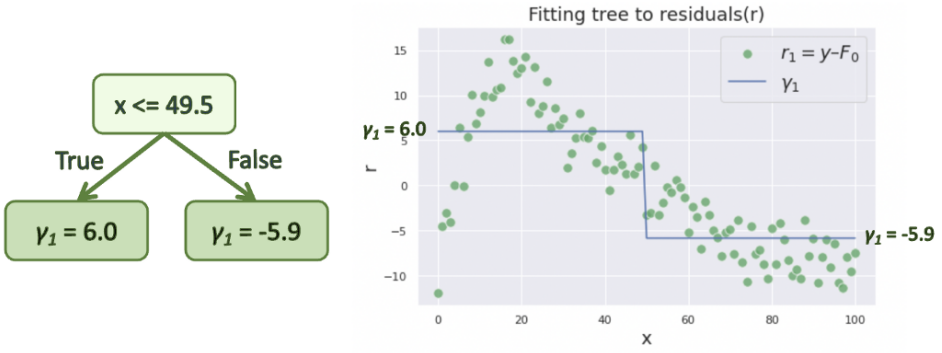


Figure B.13. Fitting the first tree to residuals r_1 (Masui, 2022).

This prediction γ_1 is added to our initial prediction F_0 to reduce the residuals. In fact, the gradient boosting algorithm does not simply add γ to F as it makes the model overfit the training data. Instead, γ is scaled down by learning rate ν which ranges between 0 and 1, and then added to F.

$$F_1 = F_0 + v \cdot \gamma_1 \tag{B.4}$$

In this example, we use a relatively big learning rate $\nu = 0.9$ to make the optimization process easier to understand, but it is usually supposed to be a much smaller value such as 0.1. After the update, our combined prediction F_1 becomes:

$$F_1 = \begin{cases} F_0 + v \cdot 6.0 & \text{if } x \leq 49.5 \\ F_0 - v \cdot 5.9 & \text{otherwise} \end{cases} \tag{B.5}$$

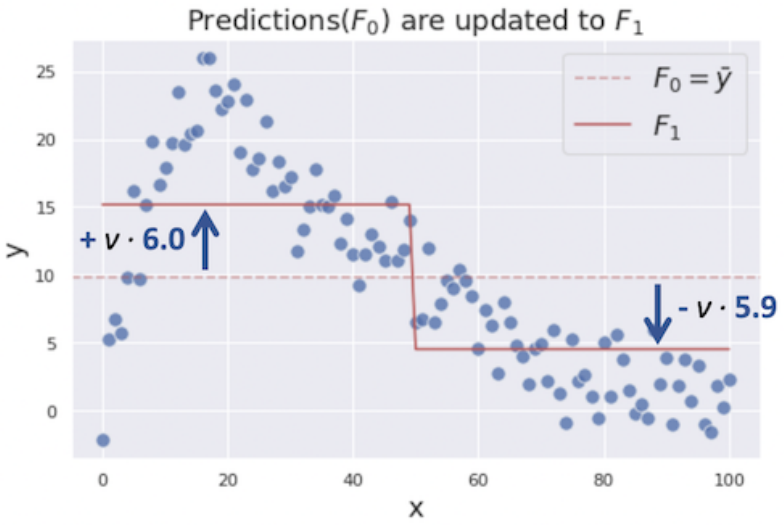


Figure B.14. Predictions(F_0) updated to F_1 (Masui, 2022).

Now, the updated residuals r_2 look like this:

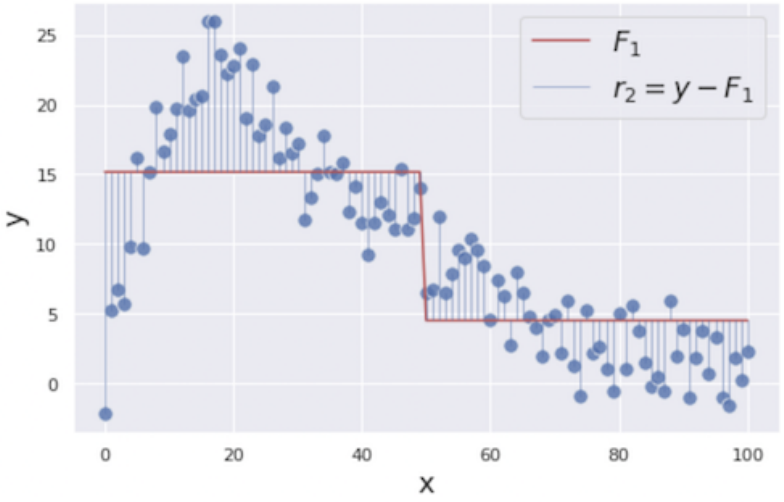


Figure B.15. The updated residuals r_2 (Masui, 2022).

In the next step, we are creating a regression tree again using the same x as the feature and the updated residuals r_2 as its target. Here is the created tree:

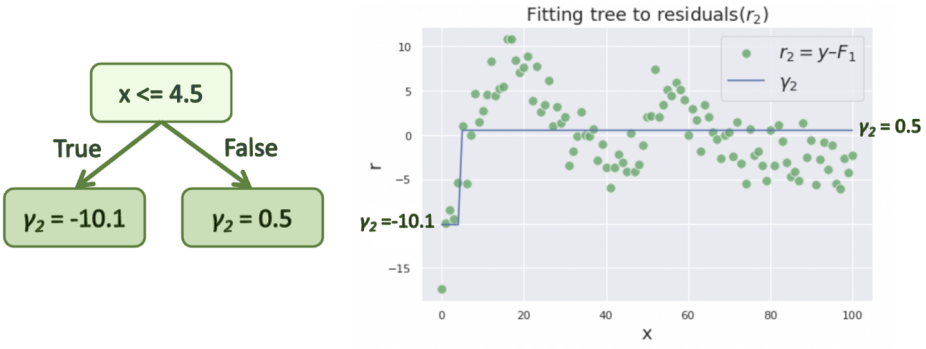


Figure B.16. Fitting the second tree to residuals r_2 (Masui, 2022).

Then, we are updating our previous combined prediction F_1 with the new tree prediction F_2.

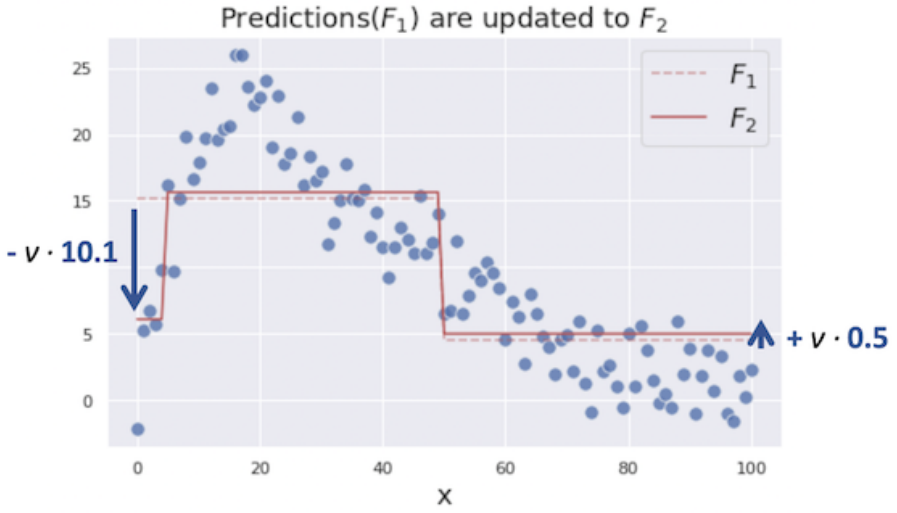


Figure B.17. Predictions(F_1) updated to F_2 (Masui, 2022).

We iterate these steps until the model prediction stops improving. The figures below show the optimization process from 0 to 6 iterations.

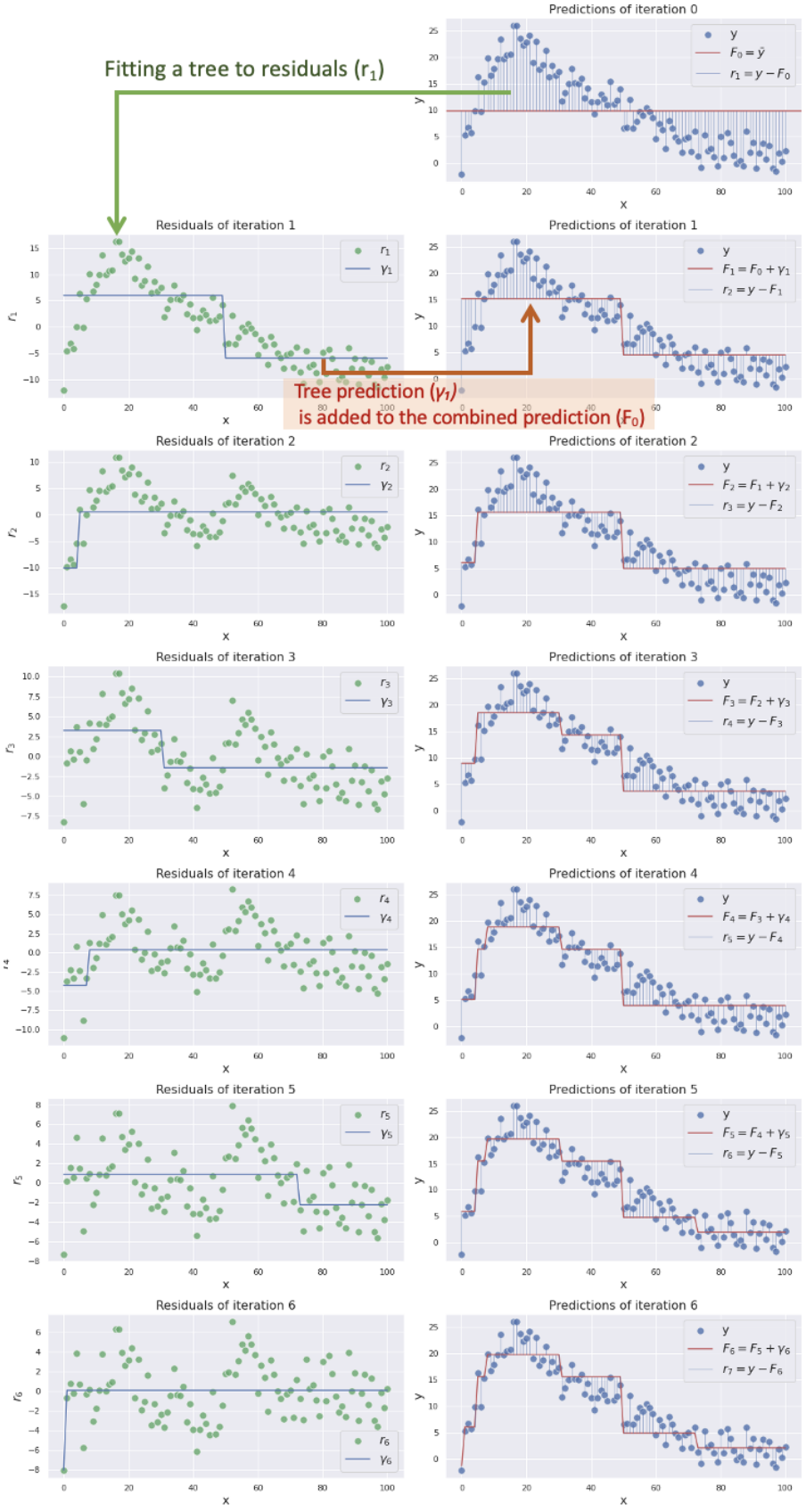


Figure B.18. Fitting trees to the residuals (the learning rate "v" is missing in the legends: $F_n = F_{n-1} + Y_n * v$) (Masui, 2022).

We can see the combined prediction F_m is getting closer to our target y as we add more trees into the combined model. This is how gradient boosting works to predict complex targets by combining multiple weak models.

To sum up, we initially take a giant step by creating a decision tree for the raw dataset. This is followed by several steps of tuning and boosting by creating decision trees that are based on the errors of the previous tree.

References

Ahmadi Naghadeh, R. and Toker, N. K. (2019). Exponential equation for predicting shear strength envelope of unsaturated soils. *International Journal of Geomechanics*, 19(7):04019061.

Al-Masri, A. (2021). How does linear regression actually work?

Baillot, D. (2018). Why are neuron axons long and spindly? study shows they're optimizing signaling efficiency.

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).

Bhandari, A. (2020). Feature scaling for machine learning: Understanding the difference between normalization vs. standardization. *Analytics Vidhya*.

Bhardwaj, A. (2022). What are artificial neural network (ann)?

Bjerrum, L. (1972). Embankments on soft ground, performance of earth and earth-supported structures. In *Proc. ASCE Specialty Conf.*, volume 2, pages 1–54.

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

Brownlee, J. (2018). A gentle introduction to k-fold cross-validation. *Machine learning mastery*, 2019.

Chakraborty, S. (2021). Understanding deep learning requires re-thinking generalization.

Chen, Y.-J. and Kulhawy, F. H. (1993). Undrained strength interrelationships among ciuc, uu, and uc tests. *Journal of Geotechnical Engineering*, 119(11):1732–1750.

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.

Craig, R. F. (2004). *Craig's soil mechanics*. CRC press.

Dao, D. V., Adeli, H., Ly, H.-B., Le, L. M., Le, V. M., Le, T.-T., and Pham, B. T. (2020a). A sensitivity and robustness analysis of gpr and ann for high-performance concrete compressive strength prediction using a monte carlo simulation. *Sustainability*, 12(3):830.

Dao, D. V., Ly, H.-B., Vu, H.-L. T., Le, T.-T., and Pham, B. T. (2020b). Investigation and optimization of the c-ann structure in predicting the compressive strength of foamed concrete. *Materials*, 13(5):1072.

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*. PhD thesis, Delft University of Technology.

Demir, N. (2016). Ensemble methods: Elegant techniques to produce improved machine learning results.

Ding, W., Nguyen, M. D., Mohammed, A. S., Armaghani, D. J., Hasanipanah, M., Van Bui, L., and Pham, B. T. (2021). A new development of anfis-based henry gas solubility optimization technique for prediction of soil shear strength. *Transportation Geotechnics*, 29:100579.

Dirgėlienė, N., Skuodis, Š., and Grigusevičius, A. (2017). Experimental and numerical analysis of direct shear test. *Procedia Engineering*, 172:218–225.

D'Ignazio, M., Phoon, K., and Länsivaara, T. (2021). Uncertainties in modelling undrained shear strength of clays using critical state soil mechanics and shansep. In *IOP Conference Series: Earth and Environmental Science*, volume 710, page 012075. IOP Publishing.

Edgell, A. (2021). Feedforward neural networks.

Gan, J., Fredlund, D., and Rahardjo, H. (1988). Determination of the shear strength parameters of an unsaturated soil using the direct shear test. *Canadian Geotechnical Journal*, 25(3):500–510.

Goktepe, A. B., Altun, S., Altintas, G., and Tan, O. (2008). Shear strength estimation of plastic clays with statistical and neural approaches. *Building and Environment*, 43(5):849–860.

Hirekerur, R. (2020). A comprehensive guide to loss functions-part 1 : Regression.

Hoang, N.-D., Pham, A.-D., Nguyen, Q.-L., and Pham, Q.-N. (2016). Estimating compressive strength of high performance concrete with gaussian process regression model. *Advances in Civil Engineering*, 2016.

Iyeke, S., Eze, E., Ehiorobo, J., and Osuji, S. (2016). Estimation of shear strength parameters of lateritic soils using artificial neural network. *Nigerian Journal of Technology*, 35(2):260–269.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer.

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.

Khan, S., Suman, S., Pavani, M., and Das, S. (2016). Prediction of the residual strength of clay using functional networks. *Geoscience Frontiers*, 7(1):67–74.

Kiran, S., Lal, B., and Tripathy, S. (2016). Shear strength prediction of soil based on probabilistic neural network. *Indian J. Sci. Technol*, 9(41):1–6.

Knuuti, M. and Länsivaara, T. (2019). Variation of cptu-based transformation models for undrained shear strength of finnish clays. *Georisk: Assessment and Management of Risk for Engineered Systems and Geohazards*, 13(4):262–270.

Kohavi, R., Wolpert, D. H., et al. (1996). Bias plus variance decomposition for zero-one loss functions. In *ICML*, volume 96, pages 275–83.

Koidan, K. (2019). 7 effective ways to deal with a small dataset.

Konrad, J.-M. and Law, K. T. (1987). Undrained shear strength from piezocone tests. *Canadian Geotechnical Journal*, 24(3):392–405.

Kulhawy, F. H. and Mayne, P. W. (1990). Manual on estimating soil properties for foundation design. Technical report, Electric Power Research Inst., Palo Alto, CA (USA); Cornell Univ., Ithaca ....

Kumar, D. (2018). Introduction to data preprocessing in machine learning.

Lora, P. J. (2019). What about small data?

Lunne, T., Powell, J. J., and Robertson, P. K. (2002). *Cone penetration testing in geotechnical practice*. CRC Press.

Ly, H.-B., Desceliers, C., Minh Le, L., Le, T.-T., Thai Pham, B., Nguyen-Ngoc, L., Doan, V. T., and Le, M. (2019). Quantification of uncertainties on the critical buckling load of columns under axial compression with uncertain random materials. *Materials*, 12(11):1828.

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).

Ly, H.-B. and Thai Pham, B. (2020). Soil unconfined compressive strength prediction using random forest (rf) machine learning model. *The Open Construction & Building Technology Journal*, 14(1).

M, R. (2020). The ascent of gradient descent.

Markou, I. N. and Droudakis, A. I. (2013). Shear strength of microfine cement grouted sands. *Proceedings of the Institution of Civil Engineers-Ground Improvement*, 166(3):177–186.

Masui, T. (2022). All you need to know about gradient boosting algorithm - part 1. regression.

Mayne, P. W., Kulhawy, F. H., and Kay, J. N. (1990). Observations on the development of pore-water stresses during piezocone penetration in clays. *Canadian Geotechnical Journal*, 27(4):418–428.

Mazur, M. (2022). A step by step backpropagation example.

McBratney, A., de Gruijter, J., and Bryce, A. (2019). Pedometrics timeline. *Geoderma*, 338:568–575.

Mitchell, J. K., Soga, K., et al. (2005). *Fundamentals of soil behavior*, volume 3. John Wiley & Sons New York.

Młynarek, Z., Stefaniak, G., and Wierzbicki, J. (2012). Geotechnical parameters of alluvial soils from in-situ tests. *Archives of Hydro-Engineering and Environmental Mechanics*, 59(1-2):63–81.

Moayedi, H., Gör, M., Khari, M., Foong, L. K., Bahiraei, M., and Bui, D. T. (2020). Hybridizing four wise neural-metaheuristic paradigms in predicting soil shear strength. *Measurement*, 156:107576.

Mohanty, A. (2019). Multi layer perceptron (mlp) models on real world banking data.

Molnar, C. (2020). *Interpretable machine learning*. Lulu. com.

Morde, V. (2019). Xgboost algorithm: Long may she reign. *Towards Data Science*.

Motaghedi, H. and Eslami, A. (2014). Analytical approach for determination of soil shear strength parameters from cpt and cptu data. *Arabian Journal for Science and Engineering*, 39(6):4363–4376.

Muraro, S. and Jommi, C. (2021). Experimental determination of the shear strength of peat from standard undrained triaxial tests: correcting for the effects of end restraint. *Géotechnique*, 71(1):76–87.

Omar, T. and Sadrekarimi, A. (2014). Effects of multiple corrections on triaxial compression testing of sands. *Journal of GeoEngineering*, 9(2):75–83.

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.

Pham, B. T., Hoang, T.-A., Nguyen, D.-M., Bui, D. T., et al. (2018). Prediction of shear strength of soft soil using machine learning methods. *Catena*, 166:181–191.

Pham, B. T., Nguyen-Thoi, T., Ly, H.-B., Nguyen, M. D., Al-Ansari, N., Tran, V.-Q., and Le, T.-T. (2020a). Extreme learning machine based prediction of soil shear strength: a sensitivity analysis using monte carlo simulations and feature backward elimination. *Sustainability*, 12(6):2339.

Pham, B. T., Qi, C., Ho, L. S., Nguyen-Thoi, T., Al-Ansari, N., Nguyen, M. D., Nguyen, H. D., Ly, H.-B., Le, H. V., and Prakash, I. (2020b). A novel hybrid soft computing model using random forest and particle swarm optimization for estimation of undrained shear strength of soil. *Sustainability*, 12(6):2218.

Pham, T. A., Ly, H.-B., Tran, V. Q., Giap, L. V., Vu, H.-L. T., and Duong, H.-A. T. (2020c). Prediction of pile axial bearing capacity using artificial neural network and random forest. *Applied Sciences*, 10(5):1871.

Pieczyńska-Kozłowska, J., Bagińska, I., and Kawa, M. (2021). The identification of the uncertainty in soil strength parameters based on cptu measurements and random fields. *Sensors*, 21(16):5393.

Puri, N., Prasad, H. D., and Jain, A. (2018). Prediction of geotechnical parameters using machine learning techniques. *Procedia Computer Science*, 125:509–517.

Qamar, H. (2020). Perceptron learning.

R, A. (2021). Regrssion in decision tree!!!FIX ME!!!-!!!FIX ME!!!a step by step cart (classification and regression tree).

Rauter, S. and Tschuchnigg, F. (2021). Cpt data interpretation employing different machine learning techniques. *Geosciences*, 11(7):265.

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.

Robertson, P. K. (1990). Soil classification using the cone penetration test. *Canadian geotechnical journal*, 27(1):151–158.

Robertson, P. K. (2009). Interpretation of cone penetration tests—a unified approach. *Canadian geotechnical journal*, 46(11):1337–1355.

Robertson, P. K. (2010). Soil behaviour type from the cpt: an update. In *2nd International symposium on cone penetration testing*, volume 2, page 8. Cone Penetration Testing Organizing Committee.

Robertson, P. K. (2016). Cone penetration test (cpt)-based soil behaviour type (sbt) classification system—an update. *Canadian Geotechnical Journal*, 53(12):1910–1927.

Robertson, P. K., Campanella, R. G., Gillespie, D., and Greig, J. (1986). Use of piezometer cone data. In *Use of in situ tests in geotechnical engineering*, pages 1263–1280. ASCE.

Ronaghan, S. (2019). The mathematics of decision trees, random forest and feature importance in scikit-learn and spark.

Roy, A. (2019). Supervised learning.

Samuel, A. L. (1967). Some studies in machine learning using the game of checkers. ii—recent progress. *IBM Journal of research and development*, 11(6):601–617.

Samui, P. (2008). Prediction of friction capacity of driven piles in clay using the support vector machine. *Canadian Geotechnical Journal*, 45(2):288–295.

Samui, P. and Sitharam, T. (2011). Machine learning modelling for predicting soil liquefaction susceptibility. *Natural Hazards and Earth System Sciences*, 11(1):1–9.

Saxena, S. (2020). The gaussian rbf kernel in non linear svm.

Schölkopf, B. (1997). Support vector learningphd thesis, technischen universität berlin.

Shahin, M. A. (2013). Artificial intelligence in geotechnical engineering: applications, modeling aspects, and future directions. *Metaheuristics in water, geotechnical and transport engineering*, 169204.

Shahin, M. A., Jaksa, M. B., and Maier, H. R. (2009). Recent advances and future challenges for artificial neural systems in geotechnical engineering applications. *Advances in Artificial Neural Systems*, 2009.

Sietsma, J. and Dow, R. J. (1991). Creating artificial neural networks that generalize. *Neural networks*, 4(1):67–79.

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

Sridharan, A., Rao, S. N., and Rao, G. V. (1971). Shear strength characteristics of saturated montmorillonite and kaolinite clays. *Soils and Foundations*, 11(3):1–22.

Stacul, S., Magalotti, A., Baglione, M., Meisina, C., and Lo Presti, D. (2020). Implementation and use of a mechanical cone penetration test database for liquefaction hazard assessment of the coastal area of the tuscany region. *Geosciences*, 10(4):128.

Terzaghi, K., Peck, R. B., and Mesri, G. (1996). *Soil mechanics in engineering practice*. John Wiley & Sons.

Tian, M. and Sheng, X. (2020). Cpt-based probabilistic characterization of undrained shear strength of clay. *Advances in Civil Engineering*, 2020.

Tian, W.-t., Dong, J.-h., Sun, J.-j., and Yang, B. (2021). Experimental study on main physical parameters controlling shear strength of unsaturated loess. *Advances in Civil Engineering*, 2021.

Tsiaousi, D., Travasarou, T., Drosos, V., Ugalde, J., and Chacko, J. (2018). Machine learning applications for site characterization based on cpt data. In *Geotechnical Earthquake Engineering and Soil Dynamics V: Slope Stability and Landslides, Laboratory Testing, and In Situ Testing*, pages 461–472. American Society of Civil Engineers Reston, VA.

Vanapalli, S., Fredlund, D., Pufahl, D., and Clifton, A. (1996). Model for the prediction of shear strength with respect to soil suction. *Canadian geotechnical journal*, 33(3):379–392.

Von Luxburg, U. and Schölkopf, B. (2011). Statistical learning theory: Models, concepts, and results. In *Handbook of the History of Logic*, volume 10, pages 651–706. Elsevier.

Wang, H., Wu, S., Qi, X., and Chu, J. (2021). Site characterization of reclaimed lands based on seismic cone penetration test. *Engineering Geology*, 280:105953.

Wang, J. (2020). An intuitive tutorial to gaussian processes regression. *arXiv preprint arXiv:2009.10862*.

Xu, Y., Wu, S., Williams, D. J., and Serati, M. (2018). Determination of peak and ultimate shear strength parameters of compacted clay. *Engineering geology*, 243:160–167.

Zhu, J.-H. and Anderson, S. (1998). Determination of shear strength of hawaiian residual soil subjected to rainfall-induced landslides. *Géotechnique*, 48(1):73–82.

Acknowledgments

The authors would like to thank the members of the TC304 Committee on Engineering Practice of Risk Assessment & Management of the International Society of Soil Mechanics and Geotechnical Engineering for developing the database 304dB used in this study and making it available for scientific inquiry. We also wish to thank <J Ching jyching@gmail.com> for contributing this database to the TC304 compendium of databases.

It has been a truly incredible and unforgettable two years studying at the Delft University of Technology.

I gratefully acknowledge my thesis supervisor Prof. dr. M.A. Hicks, who provided me with such an interesting and meaningful research subject, invited me to a fantastic workshop and generously supported my academic development. I have also received much assistance from my thesis committee. Thanks to Dr. D. Varkey for having meetings with me every week, discussing all kinds of questions with me passionately, and helping me revise the paper enthusiastically. Thanks to Dr.ir. A.P. (Bram) van den Eijnden, who has been keeping in touch with me for more than a year, guiding me with the way to go and providing me with numerous valuable suggestions. And a great many thanks to Dr. G. (Guillaume) Rongier who has taught me a lot about machine learning techniques during the meetings with great passion and professionality and revised my paper attentively.

I would like to thank all my friends in the Geo– Engineering Section for so many wonderful memories in the office, in the restaurants, in the sports centre and on the trips. Special thanks to Tianyang Lu, Xilin Yin, Yixuan Liu and Jin Yan for accompanying me in the two–year life at Delft.

I must of course in the end make my genuine acknowledgements to my parents for unconditionally supporting me in pursuing my MSc degree, providing me with adequate freedom, and guiding my life with invaluable wisdom.