

https://dspace.org/

Application of machine learning (ML) techniques for water quality assessment in a water treatment plant

Ndhlovu, Lindelwa Philasande Mpilonhle

2023

https://hdl.handle.net/10539/35668

Downloaded from a DSpace repository. https://dspace.org/



UNIVERSITY OF THE
WITWATERSRAND,
JOHANNESBURG

Application of machine learning (ML) techniques for water quality assessment in a water treatment plant

A Masters thesis submitted to the Faculty of Science, University of the Witwatersrand, in fulfilment of the requirements for the degree of Master of Science in Chemistry.

Lindelwa Philasande Mpilonhle Ndhlovu

Supervised by: Hlanganani Tutu

and

Yannick Nuapia

School of Chemistry

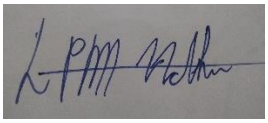
The University of the Witwatersrand, Johannesburg

May 2022

Declaration

I, Lindelwa Philasande Mpilonhle Ndhlovu (student number: 1141986), hereby declare the contents of this dissertation to be my own work. This dissertation is submitted for the degree of Master of Science in Chemistry at the University of the Witwatersrand. This work has not been submitted to any other university, or for any other degree.

Johannesburg, South Africa



Lindelwa Philasande Mpilonhle Ndhlovu, 24 May 2022

Abstract

Clean water is an important resource, one that is not readily available to everyone, especially in developing countries such as South Africa. This is due to a variety of reasons, including overpopulation, pollution, and lack of infrastructure in water treatment facilities. Gauteng Province, the economical hub, as well as the most populated province in South Africa receives its water from Rand Water, Africa's biggest water treatment facility.

Rand Water, like many other water treatment facilities uses various processes to clean raw water. Throughout these procedures, sensors and communication devices monitor chemical parameters such as pH, alkalinity, chlorine etc. in real-time. This produces enormous amounts of complex data, known as big data. It has been reported that water treatment facilities use less than 40% of the data that they produce. This has undoubtedly deprived these facilities and authorities to fully exploit such data and draw some insights that can be actionable towards better water management and planning.

Machine learning, a subfield of artificial intelligence that allows software applications to accurately predict outcomes without being explicitly programmed, has presented some of the most useful tools for processing large datasets and has found application in medicine, finance, marketing, art, music, and environmental management among other areas.

This study aimed at applying machine learning techniques to assess water quality data at one of the treatment facilities at Rand Water. The approach involved: (1) cleaning and pre-processing datasets for nephelometric turbidity units (NTU) obtained from four filter houses; (2) developing algorithms that can effectively predict future turbidity levels, namely multilayer perceptron artificial neural networks (MLP-ANN), long-short term memory (LSTM), random forest (RF), and support vector regression (SVR); and (3) testing the robustness of the models under different model input and hyperparameter settings.

For the predictive analysis, two scenarios were developed: Scenario 1 were predictions using the complete datasets and Scenario 2 were predictions using datasets that were treated for outliers. The performances of the algorithms were monitored using three loss functions, namely: root mean square error (RMSE), mean squared error (MSE) and coefficient of determination (R^2). The errors are expressed as absolute values.

For Scenario 1, with respect to the RMSE and MSE metrics, the algorithms observed similar performances, with very low errors (values closer to zero). With respect to the R^2 metric, the LSTM algorithm produced the highest average testing phase score (0.522). These results showed that the LSTM algorithm is best suited for predictive analysis when datasets are treated for outliers.

For Scenario 2, with respect to the RMSE and MSE metrics, LSTM, RF, and MLP-ANN had similar results for both the training and testing phases, which were good (errors were close to zero). The SVR algorithm gave similar performances for the training phase as the other algorithms. However, for the testing phase, the errors were elevated. The algorithm achieved its highest errors using data from Filter House 3, with an MSE of 12.577 NTU^2 and RMSE of 3.546 NTU. With respect to the R^2 metric, RF had the best performance in the testing phase (0.006) and training phase (0.238) using data from Filter House 4 C, relative to the other algorithms. RF is the only algorithm that produced positive scores. These results show that RF is best suited for predictive analysis when the datasets are not treated for outliers. However, in general (and not relative), the RF R^2 scores are very low. The algorithm may need to be improved or replaced.

For the case tests, two scenarios were tested: the effect of overfitting on the neural networks (MLP-ANN and LSTM) and the effect of data augmentation on predictions. Overfitting occurs when the algorithm successfully predicts the training dataset but fails at predicting unknown cases, and data augmentation is the process of increasing the amount of data available by mathematically introducing synthetic values. The study found that when the neural networks are not treated for overfitting, they produce poor predictions in the testing phase. The study also found that when the data is augmented, the algorithms produce poorer predictions.

The study reveals how machine learning can be implemented to treat the enormous amounts of data produced in water utilities (detection of missing values and outliers etc.) The study has also revealed the potential of the data produced in water treatment plants for predictive studies. Although the performances of the algorithms were not the best, future research can improve on these, and in turn improve the monitoring and maintenance of water utilities timeously and cost effectively.

Dedication

This thesis is dedicated to my parents, Thokozani and Thembinkosi Ndhlovu, who have been my most supportive structure. I love you both.

Acknowledgments

Great gratitude to my supervisors, Prof Hlanganani Tutu and Dr Yannick Nuapia for guiding me throughout my studies and offering valuable suggestions during the writing of this thesis. Without you, I would not have known anything about this interesting research field. I am forever grateful.

Special thanks to Rand Water and the late Miss Shakera Arendze for providing the data on which the project has been based on.

Special thanks go to The Department of Water Affairs for financial support, especially Mr Odwa Ntsika Mtembu and Miss Nobulali Mtikitiki for the additional support.

My gratitude also goes to the Environmental Analytical Chemistry group for all the extra support and for all the fun times we have had. It surely made the journey more pleasurable.

To my family and friends, I truly hope all the time I have taken to complete this research was worth it, and that I have made you proud.

Table of contents

Declaration i
Abstract ii
Dedication iv
Acknowledgments v
Table of contents vi
List of figures ix
List of tables xi
List of abbreviations and acronyms xiii
Chapter 1: Introduction 1
1.1 Research background 1
1.2 Problem statement and motivation 2
Chapter 2: Literature Review 3
2.1 Machine Learning 3
2.1.1 Grouping algorithms based on learning styles 4
2.1.1.1 Supervised learning 4
2.1.1.2 Unsupervised learning 4
2.1.1.3 Semi-supervised learning 5
2.1.2 Grouping algorithms based on similarities 5
2.1.2.1 Regression 6
2.1.2.2 Clustering 6
2.1.2.3 Artificial Neural Networks 7
2.1.2.4 Deep Learning Neural Networks 7
2.1.2.5 Ensemble Learning 7
2.1.2.6 Instance-Based Learning 7
2.1.2.7 Bayesian 7
2.1.2.8 Regularisation 8
2.1.2.9 Decision Trees 8
2.1.2.10 Dimensionality Reduction 8
2.1.2.11 Associate Rule Learning 9
2.2 Support-Vector Machines 9

2.2.1 Conceptual background 9
2.2.2 Summary of research conducted on SVM 11
2.2.3 Advantages and disadvantages 14
2.3 Random Forests 14
2.3.1 Conceptual background 14
2.3.2 Summary of research conducted on RF 16
2.3.3 Advantages and disadvantages 17
2.4 Artificial Neural Networks 18
2.4.1 Conceptual background 18
2.4.2 Summary of research conducted on ANN 19
2.4.3 Advantages and disadvantages 21
2.5 Recurrent Neural Networks 21
2.5.1 Conceptual background 21
2.5.2 Summary of research conducted on RNN (LSTM) 24
2.5.3 Advantages and disadvantages 25
2.6 Water Treatment Plants 26
Chapter 3: Aim and objectives 31
3.1 Aim 31
3.2 Objectives 31
3.3 Research Questions 31
Chapter 4: Methodology 32
4.1 Process data 32
4.2 Data pre-processing and clean up 33
4.3 Algorithm performance analysis and evaluation 35
4.4 Algorithm hyperparameter optimisation 37
4.4.1 Neural networks 37
4.4.2 Random forest 40
4.4.3 Support vector regression 43
Chapter 5: Results and discussion 46
5.1 Stationarity test results 46
5.2 Scenario 1 results 47

5.3 Scenario 2 results 57
5.4 Case test results 64
5.4.1 Case A 65
5.4.2 Case B 68
Chapter 6: Conclusions and recommendations 74
6.1 Conclusions 74
6.2 Recommendations 76
References 77

List of figures

Figure 2.1: Relationship of artificial intelligence, machine learning and deep learning 3

Figure 2.2: Mindmap of different types of machine learning algorithms 6

Figure 2.3: An SVR plot showing the hyperplane (red line) in-between two decision boundaries (black lines). Any data point outside of the decision boundaries is not a support vector..... 11

Figure 2.4: A schematic representation of an RF consisting of multiple DT algorithms 15

Figure 2.5: An MLP-ANN algorithm with two hidden layers. Information processing occurs inside the neurons, and the connection between the neurons is feedforward 19

Figure 2.6: Structure of an LSTM (the memory block) with 5 hidden neurons (states)..... 23

Figure 2.7: Illustration of the different processes used by various water treatment plants to treat their raw water. 27

Figure 2.8: The Vaal Dam and its source waters 28

Figure 4.1: Schematic diagram showing various dam flows into the Vaal Dam..... 32

Figure 4.2: Flow chart from data preparation until algorithm predictions and recommendations 33

Figure 4.3: Flowchart summarising algorithm approach 37

Figure 4.4: MLP-ANN compilation codes showing the ReLU as the chosen activation function, Adam as the optimiser, MSE as the loss function, 128 neurons for each of the two hidden layers and the construction of the early stopping technique..... 40

Figure 4.5: LSTM compilation codes showing the ReLU as the chosen activation function, Adam as the optimiser, MSE as the loss function, 128 neurons for each of the two hidden layers and the construction of the early stopping technique..... 40

Figure 4.6: RF compilation code showing the different hyperparameters that have been fine-tuned as well as their optimal values 43

Figure 4.7: SVR compilation code showing the different hyperparameters that have been fine-tuned as well as their optimal values. 45

Figure 5.1: Algorithm error averages; (a) $R^{2,}$ (b) MSE, (c) RMSE..... 50

Figure 5.2: Time-series of actual vs predicted turbidity for the testing phase using the LSTM algorithm for (a) Filter Houses 1 and 2 (b) Filter House 3 (c) Filter House 4 A and B and (d) Filter House 4 C 53

Figure 5.3: Time-series of actual vs predicted turbidity for the training phase using the LSTM algorithm for (a) Filter Houses 1 and 2 (b) Filter House 3 (c) Filter House 4 A and B and (d) Filter House 4 C 56

Figure 5.4: Time-series of actual vs predicted turbidity for both the training and testing phases using the RF algorithm for (a) Filter Houses 1 and 2 (b) Filter House 3 (c) Filter House 4 A and B and (d) Filter House 4 C 63

Figure 5.5: Algorithm error averages for predictions on datasets that excluded outliers (Scenario 1 datasets) that were augmented; (a) R^2, (b) MSE, (c) RMSE. 70

Figure 5.6: Algorithm error averages for predictions on datasets that excluded outliers (Scenario 1 datasets) that were augmented, which were further treated for outliers; (a) R^2, (b) MSE, (c) RMSE..... 72

List of tables

Table 2.1: A summary of the research reported on SVM and SVR 12

Table 2.2: Advantages and disadvantages of SVM and SVR 14

Table 2.3: A summary of the research reported on RF 16

Table 2.4: Advantages and disadvantages of RF 17

Table 2.5: A summary of the research reported on ANN 19

Table 2.6: Advantages and disadvantages of ANN 21

Table 2.7: A summary of the research reported on RNN (LSTM) 24

Table 2.8: Advantages and disadvantages of RNN (LSTM) 25

Table 4.1: The fine-tuned hyperparameters of the RF algorithm 41

Table 4.2: The fine-tuned hyperparameters for the SVR algorithm 43

Table 5.1: Results (p-values) of the ADF stationarity test 46

Table 5.2: Performance scores of the machine learning algorithms for the training and testing phases in predicting turbidity (NTU) for 4 filter houses 47

Table 5.3: Performance of the machine learning techniques for 4 filter houses when the complete datasets are considered 57

Table 5.4: Description of each case 64

Table 5.5: Performance of the machine learning techniques when two scenarios are considered. Scenario 1 are predictions with early stopping, and Scenario 2 are predictions without early stopping 65

Table 5.6: The averages of the R2 metric across all 4 filter houses for the MLP-ANN and LSTM algorithms for Scenario 1 and 2, for both the training and testing phases 67

Table 5.7: Relative errors of the MLP-ANN and LSTM algorithm between the training and testing errors for both scenarios 68

List of abbreviations and acronyms

Adam	Adaptive moment estimation
ADF	Augmented Dickey-Fuller
AI	Artificial intelligence
ANN	Artificial neural networks
BOD	Biological oxygen demand
BPNN	Backpropagation neural networks
BWTP	Brackish water treatment plant
CART	Classification and regression trees
Conv-LSTM	Conventional LSTM
DAG	Directed acyclic graph
DA-LSTM	Dual-stage attention-based LSTM
DL	Deep learning
DO	Dissolved oxygen
DT	Decision trees
EC	Electrical conductivity
FFNN	Feed-forward neural networks
GRNN	General regression neural networks
GT	Gamma test

ICA	Independent component analysis
ISODATA	Iterative self-organising data analysis technique
kNN	k-nearest neighbours
LDA	Linear discriminant analysis
LGP	Linear genetic programming
LSTM	Long-short term memory
MAPE	Mean absolute percentage error
MDP	Markov decision process
ML	Machine learning
MLP-ANN	Multilayer perceptron artificial neural networks
MLR	Multivariate linear regression
MSE	Mean squared error
NARX	Nonlinear autoregressive exogenous model
NRMSE	Normalised root mean square error
NS	Not significant
NTU	Nephelometric turbidity unit
PC	Phycocyanin
PCA	Principal component analysis
Pmiss	Miss-detection probability

Q	River discharge
R^2	Coefficient of determination
RBF	Radial basis function
ReLU	Rectified linear unit function
RF	Random forest
RFR	Multiple random forest
RMSE	Root mean square error
RNN	Recurrent neural networks
SAM	Spectral angle mapper
SAS	Systematic analysis scheme
SLRF	Self-learning random forest
SOM	Self organising maps
SVM	Support vector machine
SVR	Support vector regression
SWAT	Soil and water assessment tool
TDS	Total dissolved solids
TI	Treatability index
TS	Threshold static
WQ	Water quality

W-RFR	Wavelet transform and random forest regression
WWTP	Wastewater treatment plant

Chapter 1: Introduction

This chapter presents the background, problem statement and motivation for the study. It gives a brief overview of the gaps existing concerning effective water quality management at water treatment plants for which techniques such as machine learning can be employed.

1.1 Research background

Industrialisation has been a major driving force in development and urbanisation in many societies for decades. Even though it has its advantages, it has been identified as a major threat to the environment. It releases toxic substances to notably, water bodies. These include organic chemicals and inorganic trace elements such as lead and cadmium (Gidlow, 2004; Sharma et al., 2015). Such contamination has adversely impacted water resources, thus putting a strain on water treatment (Wang & Yang, 2016).

Conventional water treatment systems incorporate sand and carbon filters and have had some modifications in some instances to manage some emerging contaminants (Huang et al., 2021). Most of these treatment plants have recorded large amounts of data relating to quality of raw influent water and treated water for reticulation (Thompson & Kadiyala, 2014). There has been, however, limited use of this big data for meaningful water quality management, planning and predictive maintenance. Where modelling of such data has been conducted, the techniques have largely been deterministic and limited in their robustness (Newhart et al., 2019). With the advent and fast development of artificial intelligence (AI), several tools e.g., machine learning and deep learning are now available that can be usefully applied to such big data to extract patterns and predict trends from which decision making on treatment plant processes can be based.

Artificial intelligence is not a new tool, having its roots from as far back as the 1940s and 1950s when it was introduced by figures such as Alan Turing, Alan Hodgkin and Andrew Huxley (the latter two are known for the Hodgkin-Huxley algorithm) (Zaremba & Smoleński, 2000; Hodgkin & Huxley, 1952). It developed slowly over the years with cybernetics, neural networks and expert systems being some of the milestones. Improvements in computing capability, internet, and generation of large amounts of data in the past fifteen years or so have seen an exponential rise in the development and adoption of AI, especially in machine learning and deep learning. This has seen wide applications e.g. in prediction of natural disasters such as tornadoes (Abad & Iyengar,

2015) and tsunamis (Marras & Mandli, 2021); health care (Kourou et al., 2015); agriculture (Liakos et al., 2018); speech recognition and natural language processing (Van Houdt et al., 2020) and some water studies (Kirthana & Soumya, 2020; Bernardelli et al., 2020; Al-Adhaileh & Alsaade, 2021; Ramli et al., 2018).

In machine learning (which generally includes deep learning), a computer learns without being explicitly programmed. Using a mathematical notation, it receives a model from some observed dataset, also termed the training dataset and uses that model to predict for other unknown datasets, instead of using a set of codes prescribed specifically to cater for the training dataset alone (Singh & Dhiman, 2021). These tools are discussed in detail in Chapter 2.

1.2 Problem statement and motivation

In the 21st century, data is produced in a myriad of ways. These include social media activities, internet activities, scientific expeditions, business transactions, surveys, and sensor technologies, among many others. These benefit industries in understanding their operations (Gaffoor et al., 2020). Water treatment facilities often use less than 40% of the data they produce (Thompson & Kadiyala, 2014). In a world where data helps improve different subjects and departments, this is counterintuitive and presents a major disadvantage.

Water is an important natural resource and shortages are becoming more frequent. This negatively affects economies, especially in developing countries such as South Africa, which already faces water scarcity resulting in limited access to clean water for the most impoverished (Luo et al., 2015; Basson et al., 2018; Habiyaremye, 2020). Thus, proper management of this resource cannot be overemphasised. In water treatment plants, the large amounts of water quality data that are recorded provide a platform for the application of machine learning tools to predict trends, optimise plant efficiency and minimise costs through predictive maintenance e.g., filter replacement.

This study focused on the use of machine learning techniques such as multilayer perceptron artificial neural networks (MLP-ANN); a type of support-vector machine (SVM) known as support-vector regression (SVR); random forests (RF) and recurrent neural networks (RNN) based on long-short term memory (LSTM) networks to assess water quality in a water treatment plant over time. These algorithms were selected because they work best with regression problems.

Chapter 2: Literature Review

This chapter presents an overview of machine learning. The different types of machine learning algorithms are discussed as well as similarities amongst them. The chapter then expands on the four different machine learning algorithms chosen for the study (MLP-ANN, SVM (SVR), RF and RNN (LSTM)). A brief overview of the processes in water treatment facilities, focusing on the amount of data generated and its relevance.

2.1 Machine Learning

The relationship of artificial intelligence (AI), machine learning (ML) and deep learning (DL) can best be described by what is known as the Russian nested doll. DL is a subset of ML which in turn is a subset of AI (Figure 2.1).

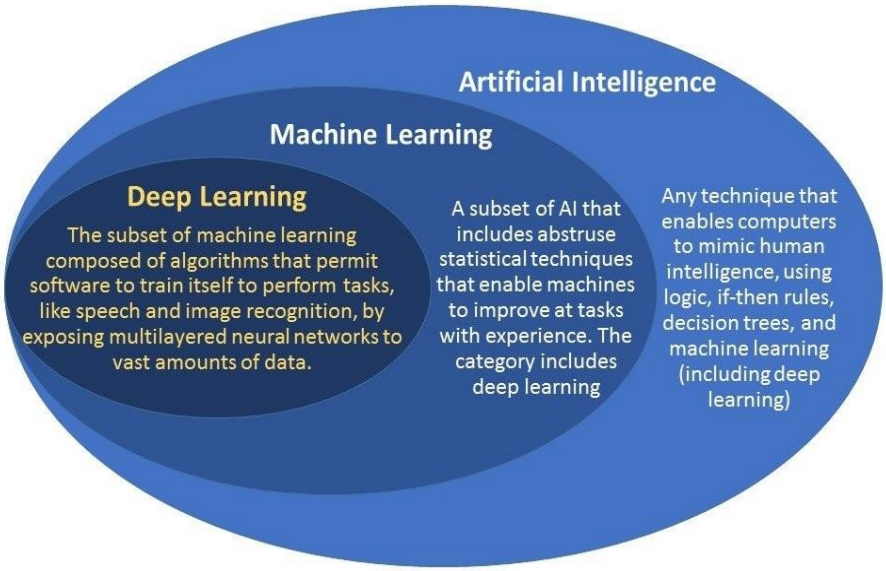


Figure 2.1: Relationship of artificial intelligence, machine learning and deep learning (https://www.geospatialworld.net/blogs/difference-between-ai%EF%BB%BF-machine-learning-and-deep-learning/ (Accessed on 31 August 2021)).

Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task, based on patterns and conjecture, without using detailed instructions and commands. Machine learning algorithms create a mathematical model based on sample data, known as the training dataset, to make predictions or decisions without the need to program them to do the task (Singh & Dhiman, 2021).

There are many machine learning algorithms e.g., k-nearest neighbours (kNN), k-means, support vector machines (SVM), artificial neural networks (ANN), decision trees, principal component analysis (PCA), etc that have been used across many disciplines. The algorithms operate differently and cater for different sorts of problems. In this section, different types of machine learning algorithms are described by grouping them based on learning styles and similarities.

2.1.1 Grouping algorithms based on learning styles

The most popular method of grouping machine learning algorithms is based on how they interact with the initial dataset, giving rise to different learning styles. Three main learning styles, namely: supervised learning, unsupervised learning and semi-supervised learning will be discussed.

2.1.1.1 Supervised learning

Supervised learning consists of a set of labelled datasets that train or supervise the algorithm into either classifying or predicting outcomes. The algorithm does not only use predetermined input variables, but also uses predetermined output variables, and traces a relationship between the two (Newhart et al., 2019). What this means is that the machine learning algorithm will take the input dataset, divide it into a training set and validation set, train using the training set and make predictions on the validation set. This continues until the highest possible accuracy between the training and validation set is achieved. Examples of supervised learning algorithms include some neural networks, naïve bayes and random forest (RF).

2.1.1.2 Unsupervised learning

Unsupervised learning algorithms use a set of unlabelled datasets. In unlabelled datasets, there is not a target variable to predict or estimate. The machine learning algorithm discovers patterns and similarities in the data and produces results from the differences (Anowar et al., 2021). Because of this, these types of algorithms are mainly used for exploratory data analysis, customer segmentation and image recognition (IBM Cloud Education, 2020). Examples of unsupervised learning algorithms include k-means, self-organising maps (SOM), independent component analysis (ICA) and principal component analysis (PCA).

2.1.1.3 Semi-supervised learning

Semi-supervised learning consists of both labelled and unlabelled datasets. The latter is often in greater amounts because it is easier to acquire and less expensive. The algorithm develops an agent which interacts with its environment, which can be the initial dataset. If the agent is successful in an estimation, it gets rewarded. If it is not successful, it receives no reward. Thus, the algorithm learns through a trial-and-error process (Ławrynowicz & Tresp, 2014). An example of a semi-supervised learning algorithm is a Markov decision process (MDP).

2.1.2 Grouping algorithms based on similarities

Another popular method of differentiating between the many different types of machine learning algorithms is to group them based on their similarities – by the way they function. Although there is almost an unlimited number of ways in which machine learning algorithms can be grouped according to their similarities, the following list is based on the most popular algorithms. A mindmap illustrating different machine learning algorithms, not exhaustive, is presented (Figure 2.2).



Figure 2.2: Mindmap of different types of machine learning algorithms (https://machinelearningmastery.com/wp-content/uploads/2013/11/Machine-Learning-Algorithms-Mindmap-Sample.png (Accessed on 31 August 2021))

2.1.2.1 Regression

A regression algorithm will trace a relationship between variables, namely, the predictor variable(s) and the target variable(s). The algorithm tries to predict the target variable and will refine the prediction by error minimisation until the highest possible estimation is achieved (Abate et al., 2020).

2.1.2.2 Clustering

Clustering algorithms divide data points in the dataset into groups based on the similarities they share. This means that data points in a particular group will be more similar than the data points in another group (Asadi-Pooya et al., 2021). There are over 100 clustering algorithms, and these are organised according to the way they operate, such as connectivity algorithms or distribution algorithms.

2.1.2.3 Artificial Neural Networks

Artificial neural networks algorithms were modelled after the way our biological neural networks function. It adds together values from the input it receives. If the values surpass a given threshold determined by an activation function, they are passed along the chain to the next neuron to be processed until the final output is calculated (Goudarzi et al., 2021). These will be discussed in detail further at a later stage.

2.1.2.4 Deep Learning Neural Networks

Deep learning neural networks algorithms are a modern update of artificial neural networks that function by constructing larger and more complex neural networks to allow for a more comprehensive analysis. Because of these complex networks, these types of algorithms can deal with large amounts of labelled datasets (Antwi-Boasiako et al., 2021). These are the algorithms that have been credited with the exponential growth and adaptation of AI e.g., in areas such as driverless cars, virtual reality and natural language processing among others.

2.1.2.5 Ensemble Learning

Ensemble learning algorithms work by combining the prediction of multiple algorithms to achieve a better overall prediction. This can also achieve greater accuracy and robustness (Guo et al., 2021). These algorithms generally use the same approach that is used to draw decisions based on "wisdom of the crowds" e.g., allowing different groups of people to work on a problem and then aggregate their decisions to make a final decision.

2.1.2.6 Instance-Based Learning

Instance-based learning, also known as memory-based learning or lazy learning is a type of machine learning algorithm that learns from the training dataset and makes generalisations on new instances based on a similarity measure. It makes assumptions from the training dataset and extends these assumptions into new entries (Aha et al., 1991).

2.1.2.7 Bayesian

Bayesian networks use the Bayes Theorem for problems such as prediction and classification. These algorithms are made up of topology structure and parameters representing the joint probability distribution of random variables. In illustrations, they are represented by directed

acyclic graphs (DAGs). These are composed of neurons and edges, which depict the random variables and the dependency relationships between these variables, respectively (Sun et al., 2021).

2.1.2.8 Regularisation

Regularisation algorithms are extensions to other algorithms that simplify these algorithms to avoid overfitting by penalising complexity (Rao & Scherer, 2010). Overfitting and underfitting are common terms to describe the behaviour of models following training and will be discussed further at a later stage.

2.1.2.9 Decision Trees

Decisions trees are algorithms that split the data continuously according to certain attributes. The simplest form consists of decision neurons and leaves. The decision neurons are used to split the data and leaves are the final output (Makariou et al., 2021). Decision trees that deal with discrete values are known as classification trees, and those that deal with continuous values are known as regression trees. Decision trees that can handle both classification and regression problems are known as classification and regression trees (CART) (Cheng et al., 2019). Random forest (RF) is a tweak on this approach where decision trees are created such that instead of selecting optimal split points, suboptimal splits are made by introducing randomness. RF is classified under ensemble methods.

2.1.2.10 Dimensionality Reduction

The performance of machine learning algorithms decreases as dimensionality increases. This is usually termed “the curse of dimensionality”. Dimensionality reduction algorithms aim to reduce the effect of dimensionality and improve the quality of the dataset by simplifying the data. This can be done through feature selection, which is the process of eliminating less informative variables. This can also be done through feature extraction, which is the process of combining existing features into a few new features through algebraic equations e.g. in PCA (Makariou et al., 2021).

2.1.2.11 Associate Rule Learning

Associate rule learning algorithms check for dependencies in variables in the dataset by assessing interesting relations and associations. The algorithm then maps these together, such as mapping similar variables in multidimensional datasets in market analysis (Shaukat Dar & Zaheer, 2015).

The following section will discuss in detail the four machine learning algorithms chosen for this study, namely: SVM (SVR), ANN (MLP), RF, and RNN (LSTM). These are best suited to handle regression problems and were selected based on their relevance for the study.

2.2 Support-Vector Machines

2.2.1 Conceptual background

Support-vector machines are supervised learning algorithms. They fit a given observation into the relevant category. For simplicity, the algorithm has been explained with the assumption that there are only two categories available. Observations are firstly placed on an n-dimensional space according to their coordinates. These are then grouped according to their similarities. A threshold is then calculated between the observations at the ends to develop the two categories. This threshold is known as a hyperplane and separates multidimensional data (Ahmed et al., 2019). Its mathematical expression is given by equation 2.1 (Moguerza & Muñoz, 2006):

$$y = w \cdot x + b \tag{2.1}$$

where: x is (x_1, x_2), the initial coordinates of a data point, b = (a, -1), a is the gradient and b is the y-intercept. The hyperplane makes predictions based on equation 2.2 (Moguerza & Muñoz, 2006):

$$h(x_i) = \begin{cases} +1 \; if \; w \cdot x + b \geq 0 \\ -1 \; if \; w \cdot x + b < 0 \end{cases} \tag{2.2}$$

A point above or on the hyperplane will be classified as class +1, and a point below the hyperplane will be classified as class -1. When a new observation enters the edge of one of the two categories, a new hyperplane is calculated using that observation. A good SVM algorithm will try to find the best hyperplane to separate the data accurately. The first type of classifier that can be developed from a hyperplane is a margin maximum classifier. The margin maximum classifier is used to calculate the shortest most possible distance between the observations on the edges of the

categories, and the distance between these observations and the hyperplane is called a hard margin (Yahya et al., 2019).

However, the maximum margin classifier is very sensitive to outliers in the training dataset. This results in an unreliable hyperplane. To make a hyperplane that is not sensitive to outliers, the SVM algorithm needs to misclassify some of the observations, so that it does not affect the classification of the next observation. Choosing a hyperplane that allows for misclassifications is an example of the Bias/Variance Trade-off that affects all of machine learning. A good hyperplane needs to have a high bias and a low variance. When misclassifications are allowed, the distance between the observations and the hyperplane is called a soft margin (Wu, 2005).

Various soft margins can be calculated for a given training dataset. Cross-validation is used to determine how many misclassifications and observations to allow inside of a soft margin to get the best classification. A soft margin classifier - also known as a support vector classifier - is used to determine the hyperplane through a soft margin (Wu, 2005). The observations used to calculate the soft margin and those inside the soft margin are called support vectors.

The type of SVM that deals with regression problems is known as a support vector regression (SVR). It uses the same principle as an SVM. However, the algorithm develops two decision boundaries in-between the hyperplane, which acts as the regression line instead of the category threshold. The objective of SVR is to only consider values within these decision boundary lines (Figure 2.3).

If the equation of a hyperplane is (2.1), then the equation of the decision boundaries become equations 2.3 and 2.4 (Sethi, 2020):

$$+a = w \cdot x + b \tag{2.3}$$

$$-a = w \cdot x + b \tag{2.4}$$

where: a is the distance between the hyperplane and a decision boundary. Any hyperplane that satisfies an SVR algorithm should satisfy equation 2.5 (Sethi, 2020):

$$a < y - w \cdot x + b < +a \tag{2.5}$$

An SVR algorithm returns continuous-valued outputs instead of outputs from a finite set, which is the reason why it is used in regression problems as opposed to SVM, which is best suited for classification problems (Awad & Khanna, 2015).

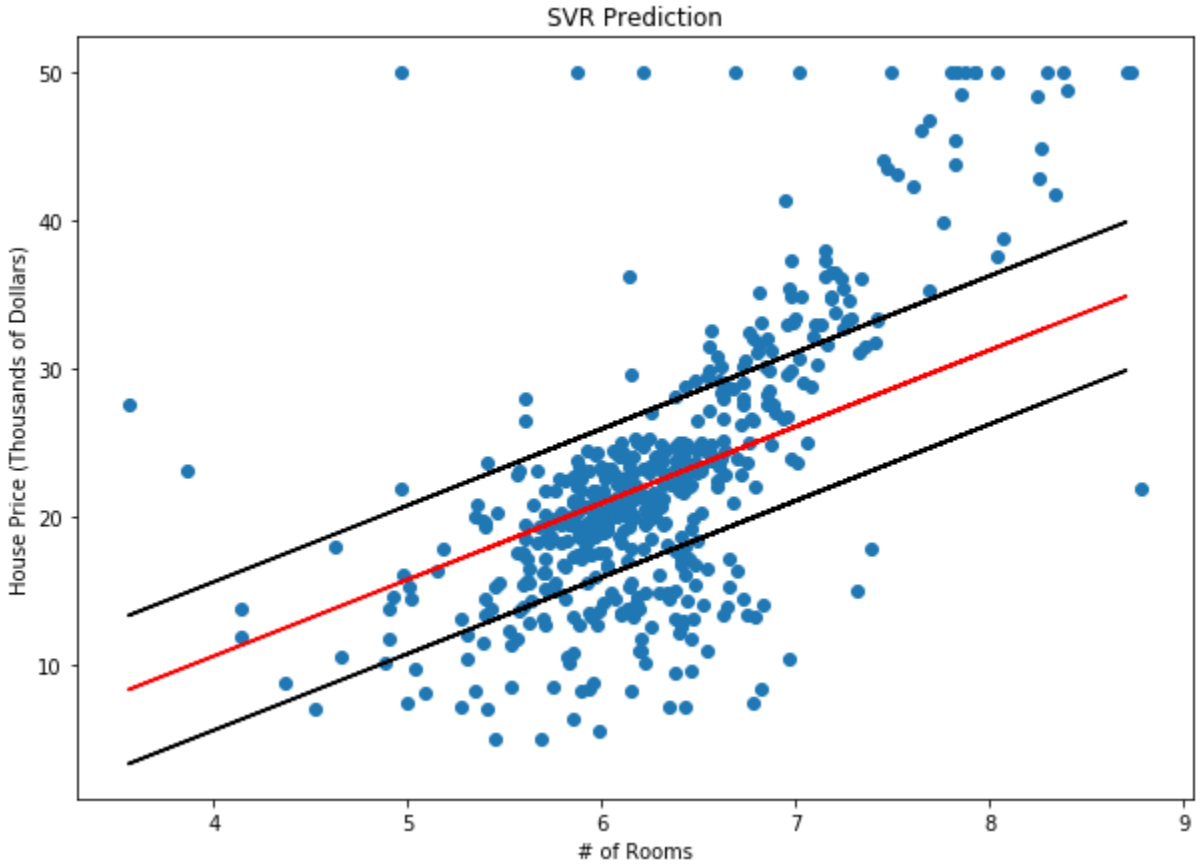


Figure 2.3: An SVR plot showing the hyperplane (red line) in-between two decision boundaries (black lines). Any data point outside of the decision boundaries is not a support vector (https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a)

2.2.2 Summary of research conducted on SVM

Several studies in water-related research have been conducted which used SVM and SVR in combination with other machine learning techniques (Table 2.1).

Table 2.1: A summary of the research reported on SVM and SVR

Author	Algorithms	Research remarks
(Olyaie et al., 2017)	MLP, RBF, LGP, SVM	The study implemented various algorithms to predict dissolved oxygen (DO) levels. The parameters used to predict DO were pH, temperature, electrical conductivity (EC) and river discharge (Q). The least mean square error (MSE) was used to produce the most efficient architectures of the algorithms. All algorithms had a not significant (NS) score larger than 0.7. An algorithm is considered very accurate if it has an NS score of 0.8 or greater. However, the algorithms were not far off and had a good performance during both the training and validation stages. SVM had a higher NS score than the other algorithms and generally performed better. Specifically, the SVM algorithm produced a lower root mean square error (RMSE), as well as a higher coefficient of determination (R^2). The SVM algorithm produced a good forecast, close to expectations when compared with other algorithms. It was concluded that SVM obtained the best relative error between DO observed and modelled.
(Kar et al., 2016)	SAM, SAM with BandMax, SVM, ISODATA	Spectral angle mapper (SAM), SAM target finder using BandMax, SVM and iterative self-organising data analysis technique (ISODATA) were used to identify spatial variability of metal pollutants in water from an image developed from remote sensing. SVM provided classes that could be correlated with natural spectral classes that were generated using the ISODATA algorithm. This is a perfect example of how well SVM

Author	Algorithms	Research remarks
		algorithms can be used in conjunction with other algorithms to produce effective results.
(Forough Kamyab-Talesh et al., 2019)	SVM	The study aimed to predict the water quality index (WQI) using SVM and identify the key attributes that influence the variability of the WQI. R^2 and RMSE statistical tools were used to validate the performance of the algorithm. The R^2 score obtained was 0.87 and the RMSE score was 0.061. The SVM algorithm was able to explain 87% of the variability of the WQI and successfully identified nitrate as the most important parameter influencing the WQI.
(Gamble & Babbar-Sebens, 2012)	PCA, DA, SVM, Kohonen SOM	The study used various methods for combining in-stream monitoring data and spatial analysis data to characterise water quality conditions. PCA was used for linear dimensionality reduction while Kohonen-SOM was used for non-linear dimensionality reduction. The results of the dimensionality reduction were assembled using kNN clustering. Linear discriminant analysis (LDA) and SVM were used to establish a relationship between the linear and non-linear variables respectively. SVM performed the best, although both algorithms performed very well.
(Fan et al., 2018)	SWAT, SVM	The study aimed to identify priority zones for river restoration. The soil and water assessment tool (SWAT) was developed to model temporal-spatial variations in environmental indicators. SVM was developed to explore the relationships between aquatic organisms and environmental indicators. The algorithms coupled could identify the relation and need for restoration.

Author	Algorithms	Research remarks
(Guan, 2020)	SVM	SVM was used to predict the scale of the hydraulic ash pipeline in a water treatment process. SVM performed better when compared to the traditional methods. Not only did it not require complex reaction modes such as chemical balance, but it also simplified intermediate links.

2.2.3 Advantages and disadvantages

Several advantages and disadvantages of SVM and SVR are listed in Table 2.2.

Table 2.2: Advantages and disadvantages of SVM and SVR

Advantages	Disadvantages
SVM is defined by a convex optimisation problem (there are no local minima) (You et al., 2013)**.**	To do multi-class classification, pairwise classifications need to be used (Karamizadeh et al., 2014).
Capable of processing binary value inputs and allows for easy adding and removal of features.	Computationally expensive.
Relatively accurate.	Very slow.
Can identify complex nonlinear relationships (Raghavendra & Deka, 2014)**.**	Has a difficult training process because of the parameter C which is determined by the analyst and determines the degree of misclassification (Modaresi & Araghinejad, 2014).

2.3 Random Forests

2.3.1 Conceptual background

Random forests (RF) are supervised learning algorithms that can accomplish both classification and regression problems, and they are based on decision trees (DT). DT consist of neurons, which

essentially relate options. Observations respond to one of these options and the tree is then grown to produce more neurons until a final output is calculated (Rakhra et al., 2021). Predictions are made by walking the splits of the tree (which is binary) until arriving at a leaf node and outputting the class value at that leaf node. DT have high variance and can yield more accurate predictions when used in an ensemble. RF is a variation of this approach in which DT are created such that rather than selecting optimal split points, suboptimal splits are made by introducing randomness.

Basically, RF consists of multiple DT which operate as an ensemble (Figure 2.4). For classification, each tree in the RF formulates a class prediction, and the class with the most votes become the algorithm's prediction. For regression, multiple trees are constructed, and the output is generated based on the average of a single-tree method. RF works because a large number of relatively uncorrelated algorithms operate as a community, which will always outperform any individual DT. The ensemble method that RF uses is called a BAGGing or Bootstrap AGGregating. The result of the regression is calculated using equation 2.6 (Gareth et al., 2013):

$$\hat{y}(x_i) = \frac{1}{B}\sum_{b=1}^{B} T_b(x_i) \tag{2.6}$$

where: x_i is the input state, B is the number of trees and T_b represents each tree.

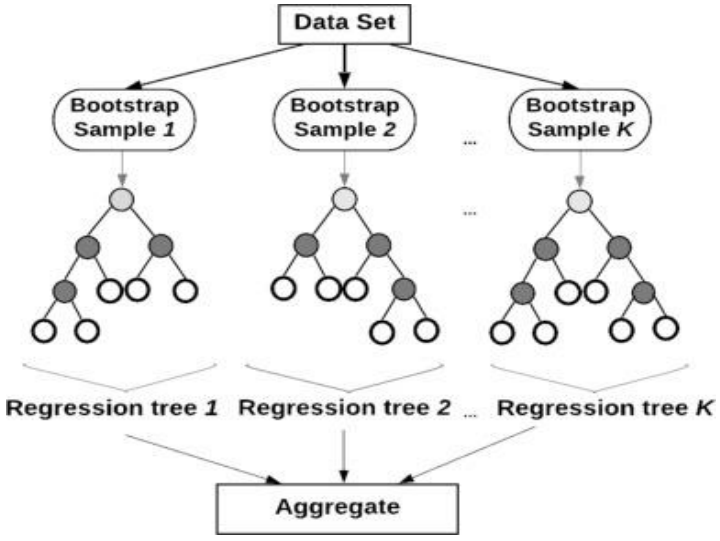


Figure 2.4: A schematic representation of an RF consisting of multiple DT algorithms (Makariou et al., 2021)

2.3.2 Summary of research conducted on RF

Several studies in water-related research have been conducted which used RF in combination with other machine learning techniques (Table 2.3).

Table 2.3: A summary of the research reported on RF

Author	Algorithms	Research remarks
(G. Chen et al., 2017)	RFR, W-RFR, FFNN	The study aimed to predict water consumption. This presented a problem for the prediction due to consumption fluctuations, given their non-stationary and non-linear characteristics. A multiple random forests (RFR) algorithm and an integrated wavelet transform and random forests regression (W-RFR) algorithm were used. Four statistical tools were used to evaluate the algorithms: R^2, mean absolute percentage error (MAPE), normalised root mean square error (NRMSE) and threshold static (TS). W-RFR could predict the basic dynamics of daily water consumption. This was further compared With RFR and feed-forward neural networks (FFNN). W-RFR had the best precision in the predictions.
(Sameen et al., 2019)	RF, SLRF, SVM, ANN, DT	The study aim was to map groundwater yield. RF was used and compared to other algorithms: SVM, ANN and DT. The RF algorithm used was a self-learning random forest (SLRF). This improved the general performance of RF by about 23%. SLRF outperformed all the other algorithms when all parameters were considered.
(Mohammed et al., 2017)	RF	This study investigated the effectiveness of an RF consisting of multiple DT in the prediction of faecal indicator organisms in raw water according to records based on conductivity, pH, colour, and turbidity, all

Author	Algorithms	Research remarks
		extracted from a drinking water source. The method was able to assess important variations in the level of microorganisms in raw water with acceptable accuracy. The algorithm predicted that the colour of the water, as well as the effect of the autumn season, were the most important parameters.
(Kutyłowska, 2017)	RF	This study used RF to predict the failure rate of water distribution pipes and house connections in selected cities in Poland. The optimal algorithm was chosen based on low costs and relatively simple architecture. The algorithm was able to isolate length out of all the variables as the most influential parameter. The algorithm received a consistency of 99%.

2.3.3 Advantages and disadvantages

Several advantages and disadvantages of RF are listed in Table 2.4.

Table 2.4: Advantages and disadvantages of RF

Advantages	Disadvantages
RF is used extensively in classification and regression problems.	They are not very accurate.
They are almost automated and very easy to compute.	They are instable. This means that they can do well in predicting the data that was used to train them. However, when new data is introduced into the algorithm, they underperform (Prajwala, 2015).
They can handle missing values through surrogate variables (Prajwala, 2015).	

2.4 Artificial Neural Networks

2.4.1 Conceptual background

Artificial neural networks (ANN) algorithms are information processors which were inspired by the way our biological nervous system functions. Our nervous system consists of trillions of neurons functioning as a collective to process information and output results. These are massive and fast. ANN algorithms work in the same way and they have been used to process large quantities of data (Chiroma et al., 2019). They have found a use (mostly as their advanced versions, deep neural networks) in self-driving cars (Haque & Riaz, 2017), image recognition (Kranthi Kumar et al., 2021), and biometric scanners (Oduah et al., 2021) and many other fields. The ability of ANN algorithms to learn quickly and intuitively detect patterns in data is what makes them so efficient.

ANN algorithms usually consist of 3 layers: an input layer, hidden layer (which can be multiple layers) and an output layer. Each layer has a certain number of neurons where the information processing takes place. Data is firstly fed into the neurons of the input layer, and this is passed along into the neurons of the hidden layer. If the hidden layer accepts n input data, then its output y is given by equation 2.7 (Li et al., 2021):

$$y = F_{act} \sum_{j=1}^{n} w_j x_j + b \tag{2.7}$$

where: x_j is the jth input data and w_j is the corresponding weight value, b is the bias value and F_{act} is the activation function. The activation function determines whether the neuron should be activated or not and determines the value of the output of the neuron from a set of input data (Li et al., 2021).

Backpropagation is used to learn the weight values and bias values of each neuron. Backpropagation is a very important machine learning algorithm used to minimise the error rate because of its ability to calculate derivatives quickly. In neural networks, it is used to fine-tune the weight values and bias values according to the error rate obtained from the previous iteration (epoch). Additional information about the network can be found in (Tuan Hoang et al., 2021).

The simplest form of an ANN is a multilayer-perceptron artificial neural network (MLP-ANN). MLP-ANN are feedforward networks, where the connection between the neurons of subsequent layers is non-cyclic (Figure 2.5). This means the neurons of any particular layer are independent of one another. In MLP-ANN, information flows forward from the input neuron(s) until the output neuron(s), and the weight values and bias values are updated after each iteration (Oduah et al., 2021).

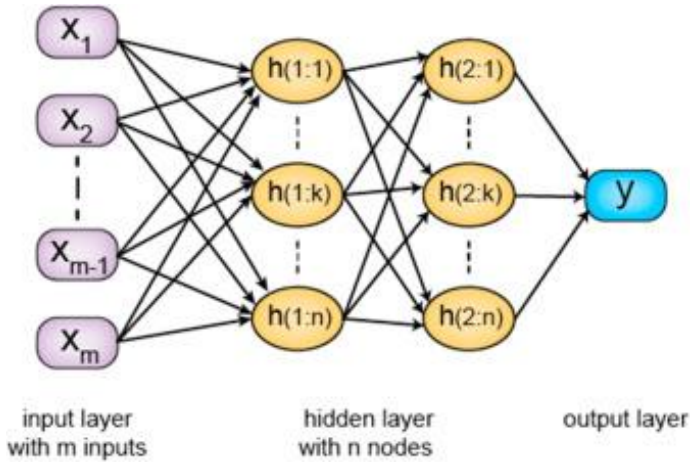


Figure 2.5: An MLP-ANN algorithm with two hidden layers. Information processing occurs inside the neurons, and the connection between the neurons is feedforward (Oduah et al., 2021)

2.4.2 Summary of research conducted on ANN

Several studies in water-related research have been conducted which used ANN in combination with other machine learning techniques (Table 2.5).

Table 2.5: A summary of the research reported on ANN

Author	Algorithms	Research remarks
(Chang et al., 2015)	ANN (NARX) with GT	The study proposed a systematic analysis scheme (SAS) to assess the spatiotemporal interrelation of WQ in an urban river, and construct WQ estimation algorithms. The study used static ANNs, and one dynamic ANN coupled with gamma test (GT). The variables used were based on hydrological, WQ and economic data. The study found that there was a WQ

Author	Algorithms	Research remarks
		problem in the area and found that nitrate was the most important parameter.
(Salami & Ehteshami, 2015)	ANN	In this study, three parameters were chosen: temperature, pH, and conductivity due to its easy and less expensive measuring methods. A package of six algorithms was submitted for evaluating DO concentration, percentage DO, biological oxygen demand (BOD), chloride, alkalinity, and total hardness. To develop the algorithms, feed-forward algorithms were used for training as well as Levenberg-Marquardt and transition functions as activation and transfer functions, respectively. The R factor (ratio of precision) was used to evaluate the algorithms and the scores were all greater than 0.77 (for all 6 trials). This proved that the package could be used to estimate the WQ parameters.
(Csábrági et al., 2015)	MLR, BPNN, GRNN	The purpose of this research was to predict one of the most important WQ parameters, DO, with the help of water pH, flow data, water temperature and conductivity data. Multivariate linear regression (MLR), backpropagation neural networks (BPNN) and general regression neural networks (GRNN) were applied and statistically compared. R^2, RMSE and MAE were used as the evaluation criteria. Using values from both the training and testing values, GRNN performed better in predicting DO. The sensitivity test revealed that DO was mostly influenced by pH and least by flow data.
(Heddam, 2016)	MLP-ANN, MLR	The study aimed to predict phycocyanin (PC) pigment using water temperature, DO, pH and specific

Author	Algorithms	Research remarks
		conductance. An MLP-ANN and an MLR were used for comparison purposes. The conclusion reached was that MLP-ANN outperformed MLR.
(Gebler et al., 2018)	ANN	The study aimed to find a correlation between ecological indices, water pollution and hydro morphological deterioration with ANN. The ANN algorithm showed favourable results despite the data being complicated and saturated.

2.4.3 Advantages and disadvantages

Several advantages and disadvantages of ANN are listed in Table 2.6.

Table 2.6: Advantages and disadvantages of ANN

Advantages	Disadvantages
Can work with incomplete information.	Difficult determination of proper network structure.
The algorithm takes a while to corrode, and corrosion slows the algorithm down.	Requires computing power.
Has parallel processing ability, which makes it possible to do more than one task at a time (Mijwil, 2018).	Once the algorithm has provided a solution, it does not give details of how the solution was obtained (unexplained behaviour of the network) (Mijwil, 2018).

2.5 Recurrent Neural Networks

2.5.1 Conceptual background

Recurrent neural networks (RNN) are a class of neural networks that are useful for modelling sequential data. They exhibit similar behaviour to the biological neural networks as well as feedforward neural networks. However, they bypass one limitation intrinsic to feedforward networks, in that the connection between the neurons of the hidden layer(s) are not only

feedforward but also feedback. This means that the neurons are not independent of one another (Yoon et al., 2021). RNNs also combine two sources of input, the present input, and the previously computed output. Therefore, an event downstream depends on one or more events that came before it. RNNs also make use of internal memory, which can remember important things about the input they receive, which allows them to be very precise in predicting future events (Kanyama, 2021). The following equation (ArunKumar et al., 2021) explains the function of a single RNN neuron:

$$h_t = F_{act}\ (w[h_{t-1}, w_t] + b \tag{2.8}$$

where: x_t is the input state, w is the weight matrix, b is the bias matrix, h_t and h_{t-1} are the hidden states for the current and previous timestamps, respectively. F_{act} is the activation function and accomplishes the same responsibilities as in feedforward neural networks. In a single RNN neuron, $h_{t-1}$and X_t are combined and multiplied with the weight matrix. The bias is then added, and the final solution is passed through an activation function to yield h_t (ArunKumar et al., 2021).

Even though RNNs have been employed in many fields for manipulating sequential data, it has a major drawback known as the vanishing gradient problem which limits its ability to train through backpropagation for long sequences of datasets. Vanishing gradient occurs when the error that is backpropagated decreases drastically with each layer that it passes, owing to the many layers that have to be traversed (Lipton et al., 2015). This is where long short-term memory (LSTM) neural networks come in. The LSTM algorithm consists of a forget gate, input gate, update gate and output gate (Figure 2.6) and was introduced by Hochreiter and Schmidhuber (1997) to overcome the vanishing gradient problem.

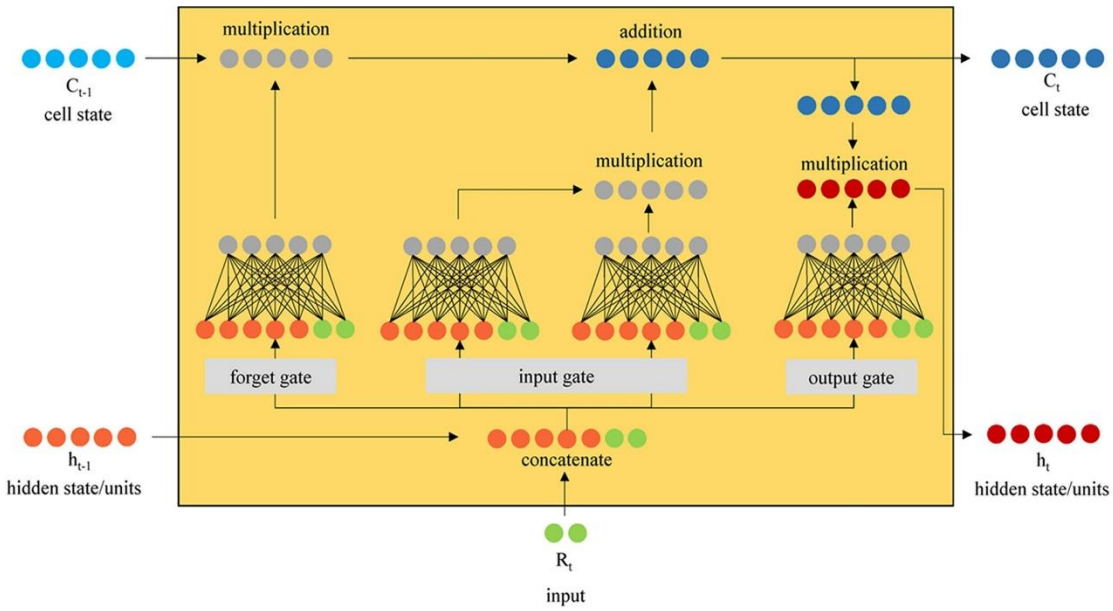


Figure 2.6: Structure of an LSTM (the memory block) with 5 hidden neurons (states) (Nourani & Behfar, 2021)

The input gate decides which information must be transferred to a neuron and is defined by equation 2.9 (ArunKumar et al., 2021):

$$i_t = F_{act}(w_i * [h_{t-1}, x_t] + b_i \quad (2.9)$$

where: * represents the element-wise multiplication of vectors. The forget gate regulates which information is worth remembering and which is not. It is defined by equation 2.10 (ArunKumar et al., 2021):

$$f_t = F_{act}(w_f * [h_{t-1}, x_t] + b_f \quad (2.10)$$

The update gate is used to update the cell, and is represented by equations 2.11 and 2.12 (ArunKumar et al., 2021):

$$\tilde{c}_t = F_{act}(w_c * [h_{t-1}, x_t] + b_c \quad (2.11)$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (2.12)$$

The output gate is used to update the hidden layer of the previous timestamp and it also updates the current output. It is defined by equations 2.13 and 2.14 (ArunKumar et al., 2021):

$$o_t = F_{act}(w_o * [h_{t-1}, x_t] + b_o \quad (2.13)$$

$$h_t = o_t * F_{act}(c_t) \quad (2.14)$$

2.5.2 Summary of research conducted on RNN (LSTM)

Several studies in water-related research have been conducted which used LSTM in combination with other machine learning techniques (Table 2.7).

Table 2.7: A summary of the research reported on RNN (LSTM)

Author	Algorithms	Research remarks
(Pisa et al., 2018)	RNN-LSTM	The study aimed to predict ammonium and total nitrogen concentration in a wastewater treatment plant (WWTP) effluent. Two RNN structures based on LSTM were used for the prediction. Results showed a miss-detection probability (Pmiss) of around 30% for both cases. In addition, low prediction errors (low MAPE) were achieved.
(Yoon et al., 2021)	Conv-LSTM, DA-LSTM	The study aimed to model the desalination of water in a brackish water treatment plant (BWTP) using two types of LSTM algorithms: conventional LSTM (Conv-LSTM) and dual-stage attention-based LSTM (DA-LSTM). DA-LSTM had better predictions with an R^2 value greater than 0.99. However, both algorithms were able to eliminate the problem of accounting for the complex causality and various correlations among the process parameters, because deep learning algorithms can describe the complexity and non-linearity of the parameters.
(Farhi et al., 2021)	LSTM	The study aimed to predict ammonia and nitrate a few hours ahead in a WWTP. Biological reactor

Author	Algorithms	Research remarks
		measurements and climate measurements were used to improve the prediction accuracy. The algorithm had an accuracy of 99% in predicting ammonia and an accuracy of 90% in predicting nitrate concentrations.
(Z. Chen et al., 2021)	TrAdaBoost-LSTM	Water treatment efficiency is plagued by missing data due to human negligence. The study aimed to improve the prediction and subsequent imputation of these missing values by exploiting the full benefits of LSTM in modelling sequential data and coupling this with instance-based learning. The resultant model is called the TrAdaBoost-LSTM and it was able to improve the imputation accuracy by 15% ~ 25%.

2.5.3 Advantages and disadvantages

Several advantages and disadvantages of LSTM are listed in Table 2.8.

Table 2.8: Advantages and disadvantages of RNN (LSTM)

Advantages	Disadvantages
The algorithm overcomes the vanishing gradient problem.	Lack of interpretability.
Remembers values over arbitrary periods (Van Houdt et al., 2020)**.**	Stochastic nature.
	Obscurity of feature contribution to predictions (Hopp, 2021).

As seen from the above literature, applications of machine learning to water research are ever becoming more important. A lot more remains to be done in this area to provide a body of work that can help transform and adapt the water sector to new technologies, given the importance of water as a resource. Water treatment plants provide the important service of sourcing, treating and reticulating water to the population. They have faced challenges of changes in water quality, which

in turn has impacted on filter maintenance and replacement. The following section intends to broaden the discussion on these plants and give a context for application of machine learning for predictive analytics.

2.6 Water Treatment Plants

In the 21st century, pollution has posed a greater than ever threat to the quality of drinking water. Many studies have found microplastics in a range of water bodies (Na et al., 2021). These have adverse effects on human beings and the surrounding ecosystem. Kim et al. (2021) found alarming levels of neonicotinoids in the influent of water treatment plants. Masindi (2020) used the Treatability Index (TI) and found that the surface water from areas like Vaalkop, Klipdrift, Wallmansthal and Cullinan in South Africa were not suitable for drinking purposes at all due to extreme levels of pollution. It is not difficult to see why water treatment should be an essential part of any evolving society.

Many water treatment plants use the traditional approach to treat raw water for drinking purposes. This involves using filters to remove solid particles. Several stages are involved, these may include coagulation, flocculation, sedimentation, stabilisation, filtration, disinfection and chlorination (Huang et al., 2021). Figure 2.7 is a summary of the process.

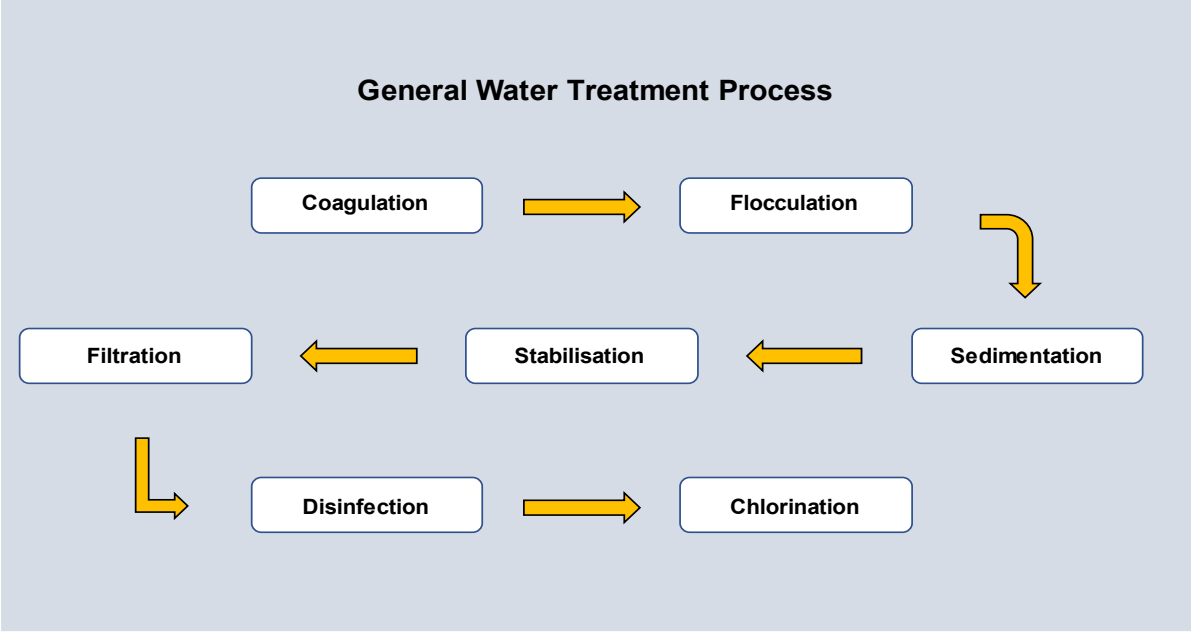


Figure 2.7: Illustration of the different processes used by various water treatment plants to treat their raw water.

The following is a brief overview of the process carried out by Rand Water, which is Africa's biggest water treatment facility.

Rand Water receives its influent from the Vaal Dam (Figure 2.8) which consists of raw water with materials that are colloidal and as a result remain distributed and suspended. Coagulation in Rand Water's systems involves destabilising these materials in a 20 – 30 second process. Calcium carbonate is fed into the kiln shift at 1200 °C to convert it to calcium oxide and carbon dioxide gas. The burnt limestone is then crushed and slaked to produce calcium hydroxide. The slake limestone is added into the raw water to destabilise the electrostatic charges of the particles in the water which is suspended. A small amount of activated sodium silicate is also added to the raw water to allow suspended particles to group and form larger groups or flocs that can then be established by gravitation. Flocculation is the clumping together of the suspended material that has undergone coagulation. The coagulated matter, known as sludge, is allowed to sediment in specially designed tanks.

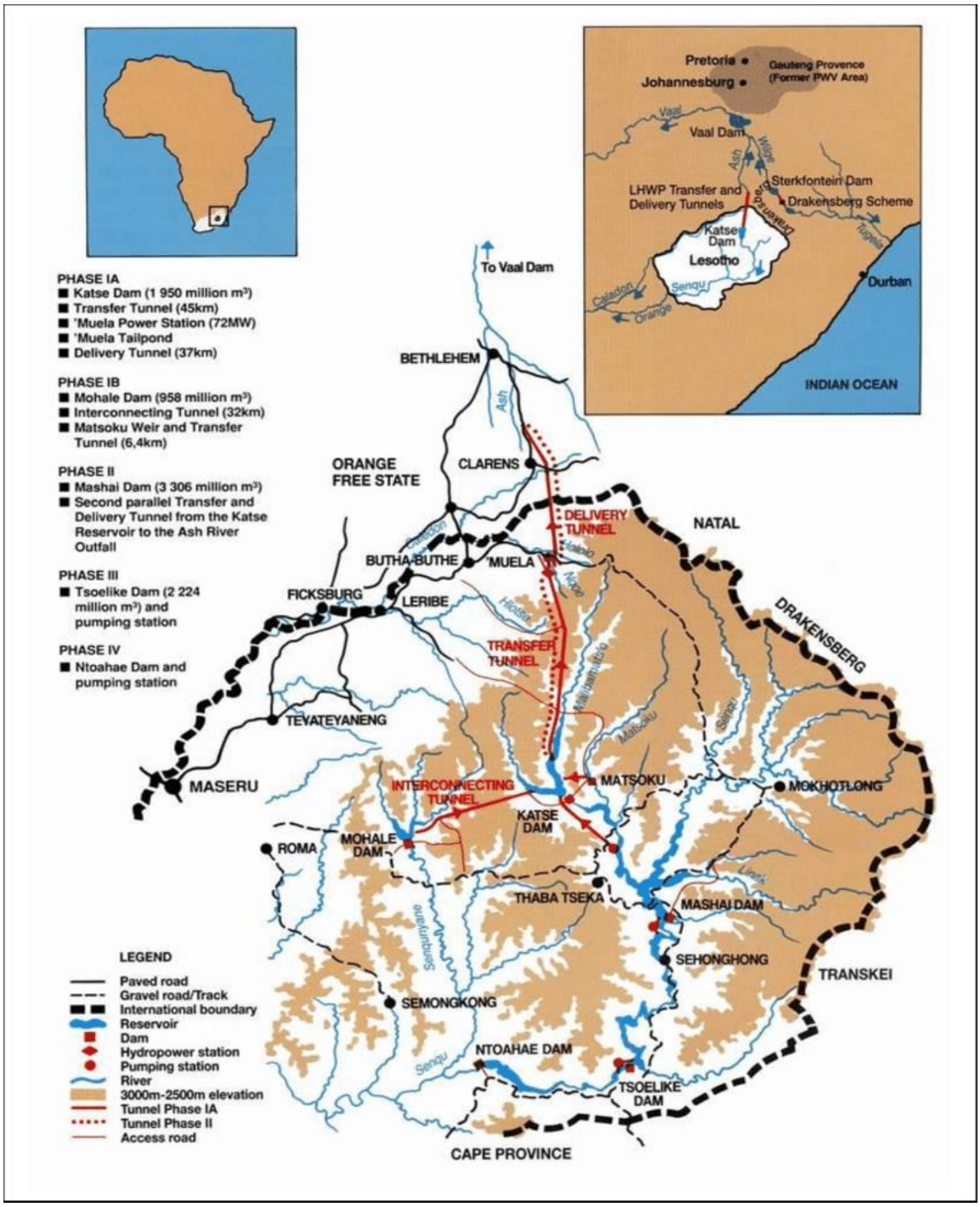


Figure 2.8: The Vaal Dam and its source waters (Turton & Meissner, 2015)

Using lime as a coagulant raises the pH of the water to about 10.5 - 11, which limits algal growth and is effective at removing heavy metals, organic materials, bacteria, and viruses. However, this pH levels are very unstable and favour scale formation. Thus, after sedimentation, the water flows

into the carbonation bays where it is stabilised by boiling the carbon dioxide gas obtained from the lime kilns through the water. This lowers the pH to 8.0- 8.4, which is necessary to produce chemically stable water, which will not cause excessive peeling or corrosion of the hardware and pipe system. The water leaving the treatment plant is disinfected with chlorine, killing microorganisms, bacteria and viruses that may be present. Chlorine is effective at disinfection but does not last very long, and the water needs to remain disinfected through the pipes until it reaches the end-user. A mixture of chlorine and ammonia is used to ensure that this is achieved.

In many water treatment utilities, throughout these different stages in the water treatment process, there are sensors and communication devices that allow for continuous monitoring (Thompson & Kadiyala, 2014; Belikova et al., 2019). With the advancement of technology, these devices monitor in real-time and with very little oversight. This produces enormous amounts of data (Shafiee et al., 2018). This data, often termed big water data, has three characteristics: volume, velocity, and variety. Volume refers to the sheer amount of the data; velocity refers to the rate that this data is generated; and variety refers to the heterogeneity of the data (Hill et al., 2014).

It comes as no surprise as to why many water facilities only use 40% of the data they generate. Sifting through this enormous amount of data would be time consuming, which is why it is rarely done. This method also presents the risk of missing important information in the data or discovering information when it is too late to make a difference. This concept is known as “Value of Now”, where information extracted from data provides the best possible solution at a given set time. In other words, before the relevance of the data expires (Thompson & Kadiyala, 2014). In addition to providing a tool for utilities to deal with the amounts of unused data they produce, the study introduces the concepts of machine learning to environmentalists and water specialist, especially those based on the African continent, and further reveals that such tools can be used to cater for datasets produced by African water utilities, in this case Rand Water.

If properly used, this big water data has the potential to close many knowledge gaps. Examples include tracing a relationship between downstream water quality and land management methods for urban growth and/or climate change; the influence of wastewater treatment practises on water quality; and the influence of energy production (hydropower) on water quality (Hill et al., 2014). Many others have already been discussed in this chapter.

Machine learning has advantages of versatility when working with large amounts of data. An array of machine learning tools can be used, thus enabling processing of data timeously, dealing with missing data and modelling relationships amongst uncorrelated data. Because of this, this study is focused on using these tools to study big datasets effectively and yield results that can be useful for the maintenance, planning, and monitoring of water quality trends in a water treatment plant.

Chapter 3: Aim and objectives

This chapter will present the aim of the research, objectives and research questions evaluated in the study.

3.1 Aim

The main aim of the study was to assess water quality with respect to process efficiency in water treatment plants using machine learning techniques.

3.2 Objectives

To achieve the main aim, the following specific objectives were pursued:

- To pre-process and clean the datasets of four filters obtained from Rand Water.
- To stationarise and then normalise the datasets.
- To optimise four machine learning algorithms for a time-series regression by fine-tuning their hyperparameters.
- To compare the performance of the machine learning algorithms.
- To test the influence of outliers on the machine learning algorithms.
- To evaluate the effectiveness of early stopping in preventing overfitting in neural networks.
- To evaluate the influence of data augmentation on the algorithms' predictions.

3.3 Research Questions

The following questions were addressed in the study:

- What is the difference between the machine learning algorithms chosen for the study and how do they handle different datasets?
- How do outliers affect the predictions and how does each machine learning algorithm deal with outliers?
- What can be done to optimise the performance of the machine learning algorithms?

Chapter 4: Methodology

This chapter presents the protocols taken to produce the time-series regressions using four machine learning algorithms. The first step involved data pre-processing and clean up. This was followed by algorithm optimisation and assessment of evaluation metrics.

4.1 Process Data

Rand Water plays an essential role in producing clean water. As stated in Chapter 2, it is the largest water utility in Africa and provides the clean water that Gauteng Province and Rustenburg use for their domestic and industrial purposes. The utility retrieves its raw water from the Vaal Dam through a canal and gravity pipeline. The Vaal Dam receives raw water from various other dams in the country (Figure 4.1). In addition, the utility also pumps water from the Vaal River Barrage Reservoir at Lethabo, Zuikerbosch and Vereeniging, and extracts a small amount of underground water from Zuurbekom.

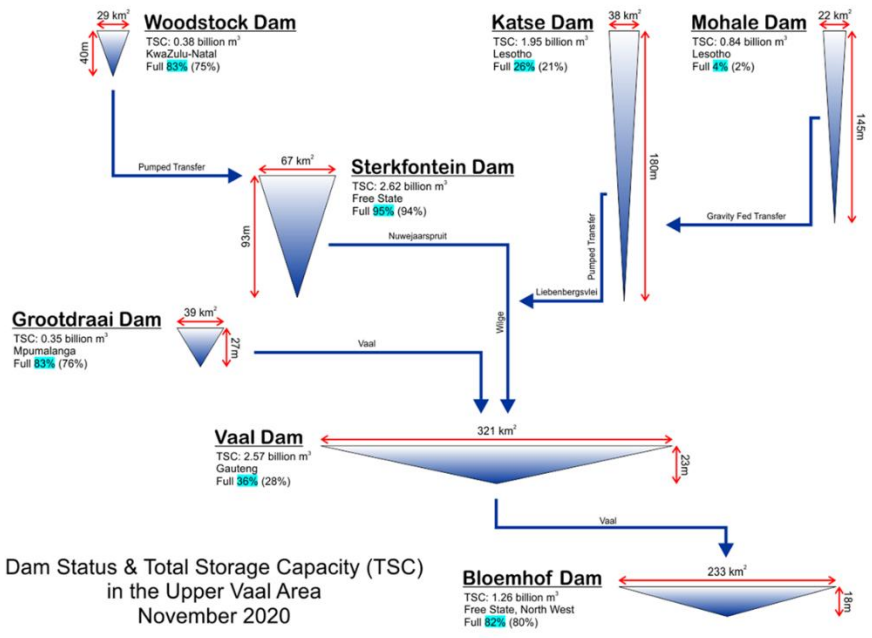


Figure 4.1: Schematic diagram showing various dam flows into the Vaal Dam (Courtesy of Francois van Wyk)

In this study, historical datasets were received from Rand Water. The datasets consisted of four filters measuring average turbidity in nephelometric turbidity units (NTU) and were used as input

data to develop the four machine learning algorithms. The four filters were labelled Filter House 1 and 2, Filter House 3, Filter House 4 A and B, and Filter House 4 C. For Filter House 1 and 2, data ranged from 1-Jan-2015 to 31-Dec-2018, resulting in four years' worth of data (1462 observations). For Filter House 3, data ranged from 1-Jan-2015 to 9-Jan-2018, resulting in just over three years' worth of data (1106 observations). For Filter House 4 A and B and Filter House 4 C data ranged from 1-Jan-2015 to 31-Dec-2017, resulting in just three years' worth of data (1097 observations).

The following sections will describe how the datasets were treated, how the four machine learning algorithms were developed and how the predictions were evaluated. The summary of the data treatment process is presented (Figure 4.2).

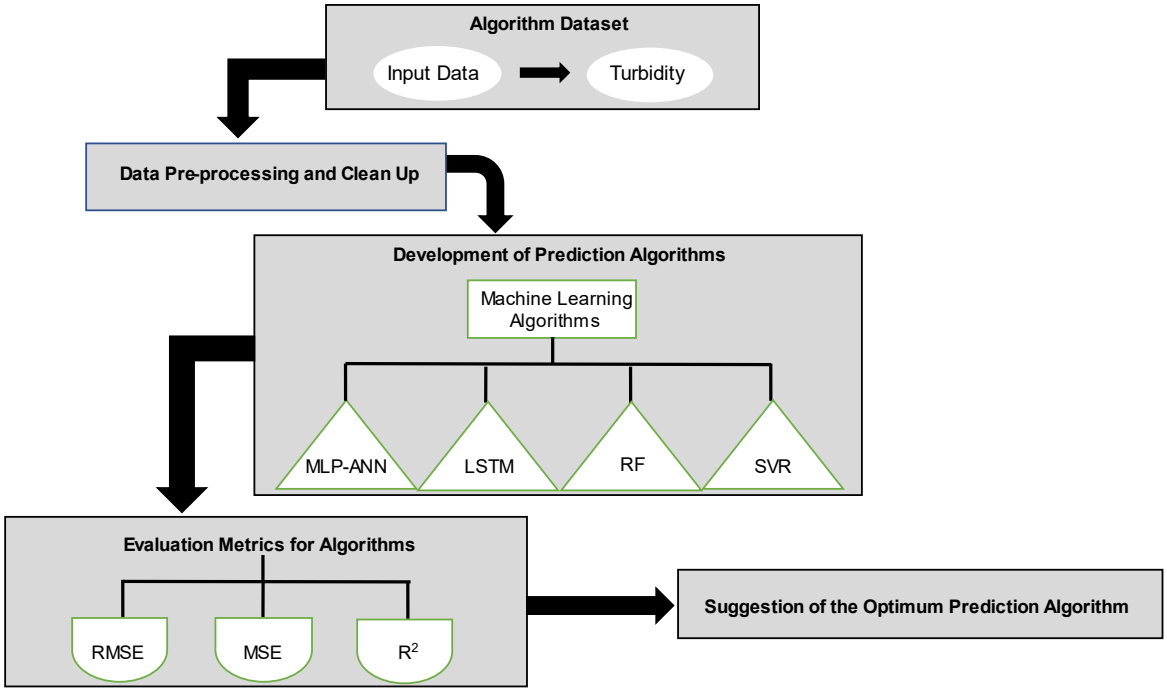


Figure 4.2: Flow chart from data preparation until algorithm predictions and recommendations

4.2 Data pre-processing and clean up

Data pre-processing is the process of detecting and/or identifying corrupt instances and replacing, modifying, or completely removing them from the database. These instances may be incomplete, incorrect, inaccurate, or irrelevant. This is done because real-life data is rarely pure and clean, and if the data is not pre-processed, it may negatively affect the predictions of the algorithms.

In this study, the first step involved the detection of missing values. Since there were minimal missing values (Filter House 1 and 2 = 7, Filter House 3 = 3, Filter House 4 A and B = 5, and Filter House 4 C = 9), the missing values were simply replaced using the mode value for each dataset (Aljuaid & Sasi, 2017).

The second step involved the detection and removal of outliers. For quantitative analysis, outliers can range from unimportant to extremely important. Unless prior knowledge of the dataset is known, is it difficult to tell whether an outlier should be excluded or included from an analysis. As such, in this study, two scenarios were formulated. Scenario 1 produced regression algorithms with possible outliers omitted. Scenario 2 produced regression algorithms with the complete datasets. The results of Scenario 1 are discussed in Chapter 5 Section 5.2 and the results of Scenario 2 are discussed in Section 5.3. Outliers were removed using equations 4.1 and 4.2 (Mirauda et al., 2020) for each filter:

$$upper\ range = Q_3 + 1.5 \times IQR \tag{4.1}$$

$$lower\ range = Q_1 - 1.5 \times IQR \tag{4.2}$$

where: Q_3 is the upper quartile, Q_1 is the lower quartile, and IQR is the difference between Q_3 and Q_1. Any data point below the lower range or above the upper range was considered an outlier and removed from the complete data set. The Pandas and Numpy libraries of Python were imported for the pre-processing and clean up of the datasets.

A reliable algorithm prediction is one that can accurately capture seasonality or trends effectively. Stationarising the datasets enables this because it standardises the training dataset with the testing dataset. This is because if one segment of the series is significantly different from the other, the algorithm may not predict successfully. A stationary series is one whose variables do not depend on the time at which the series is observed (Gagniuc, 2017). For a series to be stationary, it must have a constant mean, a constant variance, and a constant covariance between periods of identical lengths (COV (t_5, t_{15}) = COV (t_{25}, t_{35})).

In this study, the Augmented Dickey-Fuller (ADF) stationarity test was used. The test consists of a null hypothesis, which assumes that the series is non-stationary. The ADF test produces a p-value, and if the p-value of the series is below 0.05 (5%), the null hypothesis is rejected and it is

concluded that the series is stationary (Cookl, 2001). If the p-value is above 0.05, then the series needs to be corrected for stationarity. This is done by taking the 1st differential, given by equation 4.3 (Yadav et al., 2020):

$$y_i = z_i - z_{i-1} \tag{4.3}$$

where: z_i and z_{i-1} are the observations from the current and previous timestamps, respectively. The results of the stationarity test are presented in Chapter 5 Section 5.1. The Pandas and Statsmodels.tsa.stattools libraries were loaded to run the ADF stationarity test.

Subsequently, the datasets were transformed to scale through normalisation. This is to ensure that the datasets are in the same range. To avoid contaminating the datasets and giving the regressions an advantage by learning attributes from the training dataset, the scaling coefficients, which are the maximum and minimum values, were calculated from the training dataset and applied to scale the testing datasets. The mathematical expression is given by equation 4.4 (Patro & Sahu, 2015):

$$x_{normalized} = \frac{x - x_{minimum}}{x_{maximum} - x_{minimum}} \tag{4.4}$$

where: x is a data point. The Pandas and MinMaxScaller class from the Sklearn library were used for normalisation.

Finally, the datasets were split into training and testing sets. For each algorithm, 80% of the dataset was used for training and 20% of the dataset was used for testing. Empirical studies show that using approximately 80% of the data for training produces the best results (Gholamy et al., 2018). The Pandas library was used to accomplish this.

4.3 Algorithm performance analysis and evaluation

To produce the best regression algorithm, the performance of the algorithm needs to be tracked. This is done through constructive feedback. The algorithms are firstly developed, then they are evaluated and based on the results of the evaluation, suitable improvements are implemented until the highest possible accuracy is achieved. How well an algorithm performs is determined by how well it is able to predict the training dataset and new points that have not been introduced into the algorithm (Engelbrecht, 2007).

In this study, the machine learning algorithms were evaluated using three loss functions, namely: mean squared error (MSE), root mean square error (RMSE) and coefficient of determination (R^2).

MSE represents the average of the squared difference between the original and predicted values. It measures the variance in the residuals given by equation 4.5 (Aziz & Abdullah, 2015):

$$MSE = \frac{1}{N}\sum_{z=1}^{n=i} e^2 \tag{4.5}$$

where: e is the difference between the observed and predicted values, N is the total training sample and n_i is the size of class i.

RMSE is used to determine the amount of error found in the predicted values by comparing actual vs predicted results. It is the square root of the MSE given by equation 4.6 (Engelbrecht, 2007):

$$RMSE = \sqrt{\frac{1}{N}\sum_{z=1}^{n=i} e^2} \tag{4.6}$$

In general, minimum values (values closer to zero) of MSE and RMSE means the model performed better (Z. Chen et al., 2021; Jasim Saud & Kudair Abass, 2018). The loss functions were chosen for the study because they best explain the spread of the predicted values from the regression line.

R^2 shows how well the data fit the model. It measures the variability between the actual and estimated values. The mathematical expression is given by equation 4.7 (Figueiredo Filho et al., 2011):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(P_i - O_i)^2}{\sum_{i=1}^{n}(O_i - \bar{O})^2} \tag{4.7}$$

where: P_i is the predicted value, O_I is the observed value, $\bar{O}$ is the mean of the test dataset and n is the size of the test dataset. (Yoon et al., 2021) . R^2 values range from $-\infty$ to 1. (Nourani & Behfar, 2021). A value of 1 is favourable, as this suggests a perfect fit. A value of 0 suggests a constant algorithm that always predicts the expected value of the dependent variable, disregarding the input features. A negative value suggests that the algorithm performs worse than the mean value. In this

study, the errors are expressed as absolute values. R^2 was chosen for the study because it is the standard metric used for evaluating the performance of regression algorithms.

4.4 Algorithm hyperparameter optimisation

Often, choosing the optimal structure of a machine learning algorithm is a trial-and-error process. Model hyperparameters are used to control the learning process. While some algorithms come with some fixed hyperparameters, other algorithms require a lot of fine-tuning. In this section, the optimal structure of machine learning algorithms (MLP-ANN, LSTM, RF and SVR) is discussed and some basic principles behind their hyperparameters defined (Figure 4.3).

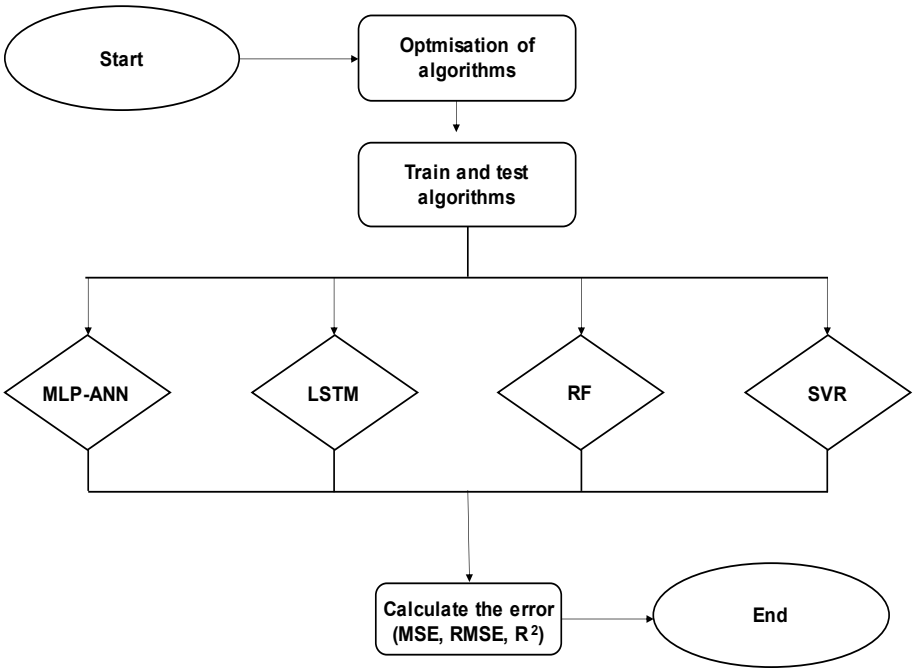


Figure 4.3: Flowchart summarising algorithm approach

4.4.1 Neural networks

In this section, MLP-ANN and LSTM (both neural networks) will be discussed. The hyperparameters that were fine-tuned in the neural networks included the number of hidden layers as well as the number of neurons for each hidden layer, the activation function, the number of epochs, the loss function, optimisation function, and the batch size.

Both algorithms consisted of two hidden layers with 128 neurons each. The way information is processed inside the neurons has already been discussed in Chapter 2. The networks used the rectified linear unit (ReLU) activation function. This is a type of non-linear activation function that improves the performance of deep learning networks and prevents the problem of vanishing gradients, which is common in networks with a large number of hidden layers, because its derivative in the positive portion of its domain is equal to 1 (Oostwal et al., 2021). It works by outputting the input directly if it is positive, otherwise, it will output zero (Kranthi Kumar et al., 2021; Khodabakhsh et al., 2020). The mathematical expression is given by equation 4.8 (Lipton et al., 2015):

$$f(x) = \max(0, x) \tag{4.8}$$

where: $f(x)$ is the activation function and x represents the input state of the neuron.

The MSE loss function was used to measure the loss between predicted and actual values (equation 4.5) and was optimised using the adaptive moment estimation (Adam) optimiser. Adam uses estimations of first and second moments of gradients to adapt the learning rate for each weight in the neural network. A moment is defined as the expected value of a variable to the power of n given by equation 4.9 (Wang et al., 2019):

$$m_n = E[x^n] \tag{4.9}$$

where: m is the moment and x is the value of the random variable. To estimate the moments, Adam uses exponentially moving gradients, computed on the gradient evaluated on a current mini batch given by equations 4.10 and 4.11 (Wang et al., 2019):

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{4.10}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{4.11}$$

where: m and v are the moving averages, g is the gradient of the current mini-batch, and β_1 and β_2 are equal to the good default values of 0.9 and 0.999, respectively, which are rarely ever changed. The moments are biased towards zero during the first few iterations. Biased corrected

first and second moments counteract this and the mathematical expressions are given by equations 4.12 and 4.13 (Wang et al., 2019):

$$\widehat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{4.12}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{4.13}$$

The moving averages are then used to scale the learning rate individually for each parameter given by equation 4.14 (Wang et al., 2019):

$$W_t = W_{t-1} - \eta \frac{\widehat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{4.14}$$

where: W_t and W_{t-1} are the weights of the current and previous timestamps, respectively and η is the step size.

The batch size was set to 32 and the number of epochs was set to 100. To prevent overfitting, the early stopping technique was used (the influence of early stopping will be discussed in detail in Chapter 5 Section 5.4.1). Overfitting occurs when an algorithm is too fine-tuned for the training dataset that it cannot cater for unknown datasets (Arias del Campo et al., 2021). The training process was set to stop if the performance on the validation set did not improve after 3 epochs (patience = 2).

The MLP-ANN algorithm expects input data to be 2 dimensional (samples and features) and the LSTM algorithm expects the data to be 3 dimensional (samples, features, and time steps). Samples are the number of rows in the dataset, features are the number of columns, and time steps are the past observations of the features. The input data was reformatted to meet these requirements.

Finally, neural networks produce varying results every time that the algorithm is run, and this is due to changing randomised weights. In this study, the experiment was performed ten times and the average scores were recorded. Figure 4.4 and Figure 4.5 show the codes used to compile the

MLP-ANN and LSTM algorithms, respectively. All the codes were developed using the Python programming language on an Intel® Core™ i5-8265U CPU @ 1.60GHz.

```
#building the model
model = Sequential() #building a linear ANN

model.add(Dense(128, input_dim=1, activation='relu'))#128 nodes in the first hidden layer

model.add(Dense(units=128, activation="relu")) #128 nodes in the next hidden layer

model.add(Dense(1))#one output node

model.compile(loss='mean_squared_error', optimizer='adam') #MSE = loss function. Adam = optimiser

early_stop = EarlyStopping(monitor='loss', patience=2, verbose=1)#use early stopping to avoid over fitting

model.fit(x_train, y_train, epochs=100, batch_size=32, verbose=1, shuffle=False, callbacks=[early_stop])
```

Figure 4.4: MLP-ANN compilation codes showing the ReLU as the chosen activation function, Adam as the optimiser, MSE as the loss function, 128 neurons for each of the two hidden layers and the construction of the early stopping technique

```
#building the model
model = Sequential() #building a linear ANN

model.add(Dense(128, input_dim=1, activation='relu'))#128 nodes in the first hidden layer

model.add(Dense(units=128, activation="relu")) #128 nodes in the next hidden layer

model.add(Dense(1))#one output node

model.compile(loss='mean_squared_error', optimizer='adam') #MSE = loss function. Adam = optimiser

early_stop = EarlyStopping(monitor='loss', patience=2, verbose=1)#use early stopping to avoid over fitting

model.fit(x_train, y_train, epochs=100, batch_size=32, verbose=1, shuffle=False, callbacks=[early_stop])
```

Figure 4.5: LSTM compilation codes showing the ReLU as the chosen activation function, Adam as the optimiser, MSE as the loss function, 128 neurons for each of the two hidden layers and the construction of the early stopping technique

4.4.2 Random Forest

Random forests are much easier to set up than neural networks. In this study, the parameters that were fine-tuned were: the number of estimators or decision trees (DT); the maximum number of features allowed on each DT; depth of the random forest, which is the distance between the root node (topmost node) and leaf node (a node without a parent); the minimum samples split, defined

as the minimum number of samples required to split a node; and the minimum samples for each leaf node, which contains the predictor which associates a pattern with the desired output.

To find the optimal values of the hyperparameters aforementioned, grid search was employed. It is a library function that helps to loop through predefined hyperparameters and fit the model on the training dataset. In this method, each combination of hyperparameter value is evaluated. This makes the process time consuming but is the most efficient method as it provides the least possibility of missing out on an optimal solution for a model.

Because the analysis involved time-series data, which is temporal data, using grid search with k-fold cross-validation would have produced incorrect results. This is because k-fold cross-validation randomly splits the data into several sections, called folds (determined by the hyperparameter k), where most of the folds are used for training and the last fold is used for testing. For a time-series analysis, this can result in the future/forecast data occurring before the training data. To circumvent this problem, the python library function TimeSeriesSplit was employed. This function breaks the data into several folds, where the test data fold consists of a single observation, and the train data folds consists of observations that occur before the observation in the test data fold. This ensures that no future observation can be used in constructing the forecast. In this study, a 10-fold TimeSeriesSplit was employed. Table 4.1 represents the results of the different hyperparameters for each filter house, for both Scenario 1 and 2.

Table 4.1: The fine-tuned hyperparameters of the RF algorithm

Hyperparameter	**Filter House**	**Scenario**	**Value**
Number of estimators	Filter House 1 and 2	1	100
		2	200
	Filter House 3	1	500
		2	100
	Filter House 4 A and B	1	100
		2	100
	Filter House 4 C	1	500
		2	100
	Filter House 1 and 2	1	5

Hyperparameter	Filter House	Scenario	Value
Minimum samples split		2	20
	Filter House 3	1	20
		2	10
	Filter House 4 A and B	1	20
		2	20
	Filter House 4 C	1	15
		2	10
Minimum samples leaf	Filter House 1 and 2	1	1
		2	20
	Filter House 3	1	15
		2	20
	Filter House 4 A and B	1	15
		2	15
	Filter House 4 C	1	20
		2	20
Maximum features	Filter House 1 and 2	1	auto
		2	auto
	Filter House 3	1	sqrt
		2	auto
	Filter House 4 A and B	1	log2
		2	log2
	Filter House 4 C	1	log2
		2	auto
Model depth	Filter House 1 and 2	1	40
		2	10
	Filter House 3	1	50
		2	30
	Filter House 4 A and B	1	30
		2	30

Hyperparameter	Filter House	Scenario	Value
	Filter House 4 C	1	30
		2	30

The RF algorithm was run ten times for each filter house and the average scores was recorded. Figure 4.6 shows an example of the code use to compile the algorithm.

```
#building the RF algorithm
model = RandomForestRegressor(n_estimators=200, #number of decision trees
                              max_features='auto', #maximum features allowed on each decision tree
                              max_depth=10, #the depth of the RF
                              min_samples_split=20, #min samples required to split a node
                              min_samples_leaf=20) #min samples per decision node
```

Figure 4.6: RF compilation code showing the different hyperparameters that have been fine-tuned as well as their optimal values

4.4.3 Support Vector Regression

The most important hyperparameters of the SVR algorithm are C, which is the penalty classifier and measures the degree of misclassification; kernel, which helps the model predict non-linear data; and lastly, gamma, which is the measure of the influence of a training sample, with low values meaning far and high values meaning close. Grid search with 10-fold TimeSeriesSplit was also used to find the optimal values of the aforementioned hyperparameters (Table 4.2).

Table 4.2: The fine-tuned hyperparameters for the SVR algorithm

Hyperparameter	Filter House	Scenario	Value
C	Filter House 1 and 2	1	1
		2	10
	Filter House 3	1	10

Hyperparameter	Filter House	Scenario	Value
		2	10
	Filter House 4 A and B	1	10
		2	1
	Filter House 4 C	1	10
		2	10
Gamma	Filter House 1 and 2	1	1
		2	0.1
	Filter House 3	1	0.1
		2	0.1
	Filter House 4 A and B	1	1
		2	1
	Filter House 4 C	1	1
		2	1
Kernel	All the datasets optimised with the RBF kernel for both Scenarios		

The optimal kernel found by the study is the radial basis function (RBF). Unlike the linear kernel, it maps out samples into higher dimensional space to trace the relationship between non-linear data points. The mathematical expression of the RBF kernel is given by (Afentoulis, 2015):

$$K(x, x') = exp \frac{\|x - x'\|^2}{2\Theta^2} \tag{4.16}$$

where: x and x' are the actual and predicted values, respectively, and θ is the free parameter. The SVR algorithm was run ten times and the average scores was recorded. Figure 4.7 shows the code used to compile the SVR algorithm.

```
#Building the SVR algorithm
clf = svm.SVR(kernel='rbf', #the kernel
              C=1, #degree of misclassification
              gamma=1) # degree of the influence of the training data
```

Figure 4.7: SVR compilation code showing the different hyperparameters that have been fine-tuned as well as their optimal values.

Chapter 5: Results and discussion

This chapter presents the results and discussions of the stationarity test, the algorithm results of Scenario 1 (predictions where the datasets were treated for outliers) and Scenario 2 (predictions where the datasets were not treated for outliers) and two case studies that looked at the influence of overfitting on neural networks and data augmentation on the performance of machine learning algorithms.

5.1 Stationarity test results

This section will present the results obtained by the ADF test. As stated in Chapter 4 Section 4.1, two scenarios were developed to test for the influence of outliers. Scenario 1 consisted of datasets where possible outliers were omitted, and Scenario 2 consisted of datasets where possible outliers were not omitted. Datasets in both scenarios were tested for stationarity. The algorithm prediction results of Scenario 1 are discussed in Section 5.2 and those of Scenario 2 are discussed in Section 5.3. Table 5.1 shows the p-values obtained for each dataset.

Table 5.1: Results (p-values) of the ADF stationarity test

Filter House	Scenario 1	Scenario 2
Filter House 1 and 2	0.000003	1.46×10^{-7}
Filter House 3	0.035576	0
Filter House 4 A and B	0.004293	0
Filter House 4 C	0.001779	1.04×10^{-14}

All the datasets had p-values below 0.05, meaning that according to the ADF test, the null hypothesis can be rejected, and the datasets are stationary. As stated in Chapter 4 Section 4.2, a stationary series has statistical properties (variance, covariance and mean) that do not change overtime. In a nonstationary series, these properties will vary with time. Algorithms such as LSTM are capable of learning nonlinearities and because of this, they are capable of learning long-term dependencies. In theory, such algorithms do not require the data to be stationary to produce

successful predictions (Preeti et al., 2019). However, a lot of data would be required to learn long-term dependencies and when the algorithms are applied over stationary datasets, the algorithms can train better and improve their predictions.

5.2 Scenario 1 results

This section will discuss the results of the predictions obtained by each algorithm using the evaluation metrics discussed in Chapter 4. Note, the datasets used to produce these results excluded outliers (Scenario 1). Table 5.2 summarises the results of both the training and testing phases. The size of the training and testing datasets will vary depending on the period length of the Filter House.

Table 5.2: Performance scores of the machine learning algorithms for the training and testing phases in predicting turbidity (NTU) for 4 filter houses

Filter House	Phase	Algorithm	R^2	MSE	RMSE
Filter House 1 and 2	Training	MLP-ANN	0.358	0.001	0.032
		LSTM	0.319	0.004	0.063
		RF	0.389	0.001	0.032
		SVR	0.358	0.001	0.032
	Testing	MLP-ANN	0.279	0.024	0.155
		LSTM	0.281	0.026	0.161
		RF	0.269	0.023	0.152
		SVR	0.281	0.024	0.155
Filter House 3	Training	MLP-ANN	0.761	0.002	0.045
		LSTM	0.765	0.001	0.032
		RF	0.774	0.001	0.032
		SVR	0.766	0.006	0.077
	Testing	MLP-ANN	0.651	0.010	0.099
		LSTM	0.660	0.010	0.099
		RF	0.650	0.009	0.094
		SVR	0.634	0.010	0.099

Filter House	Phase	Algorithm	R^2	MSE	RMSE
Filter House 4 A and B	Training	MLP-ANN	0.753	0.004	0.063
		LSTM	0.760	0.001	0.032
		RF	0.770	0.001	0.032
		SVR	0.750	0.002	0.045
	Testing	MLP-ANN	0.647	0.010	0.099
		LSTM	0.650	0.013	0.114
		RF	0.648	0.006	0.077
		SVR	0.633	0.011	0.105
Filter House 4 C	Training	MLP-ANN	0.753	0.004	0.063
		LSTM	0.763	0.004	0.063
		RF	0.769	0.002	0.045
		SVR	0.765	0.001	0.032
	Testing	MLP-ANN	0.485	0.009	0.095
		LSTM	0.495	0.010	0.099
		RF	0.460	0.009	0.095
		SVR	0.458	0.009	0.095

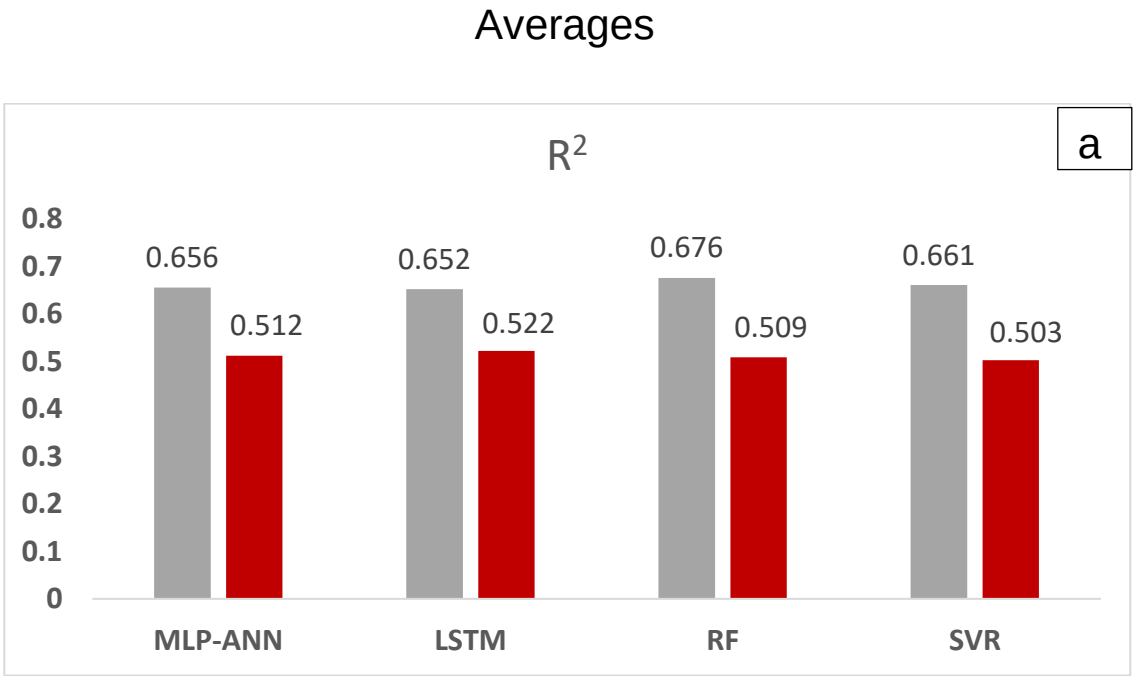
Table 5.2 shows that with regards to the RMSE and MSE error metrics in both the training and testing phases, all the algorithms had good performances since all the values were very close to zero. LSTM produced the highest MSE (0.026 NTU^2) and RMSE (0.161 NTU) for the testing phase, whilst SVR produced the highest MSE (0.006 NTU^2) and RMSE (0.077 NTU) for the training phase. Since the values were very close to zero, it indicates that the data points were not too dispersed from the regression line. There is a slight increase in the mentioned error metrics when moving from the training phase to the testing phase, which is expected. This is because the algorithms were optimised using the training dataset, while the testing dataset consists of instances that the algorithms had not observed yet, making them harder to predict.

With regards to the R^2 metric, all the algorithms performed well when using data from Filter House 3. The training phase scores were 0.761 (MLP-ANN), 0.765 (LSTM), 0.774 (RF) and 0.766 (SVR) and the testing phase scores were 0.651 (MLP-ANN), 0.660 (LSTM), 0.650 (RF) and 0.634 (SVR).

This could speak to the integrity of the dataset, meaning it was accurate and reliable. RF had the best performance during the training phase with a score of 0.774, but LSTM had the best performance during the testing phase with a score of 0.660. This adds to the ever-growing concept of the LSTM algorithm's ability to produce effective time-series results better than other algorithms. This is largely because LSTM can learn the context required to make a prediction, rather than having the context pre-specified (Khodabakhsh et al., 2020).

While RF had the best performance during the training phase using data from Filter House 3, it did not observe the best performance in the testing phase. This could be an indication of overfitting, which occurs when the algorithm is too finetuned for the training dataset that it learns the noise instead of the relationship between the data points, which results in its underperformance when trying to predict unknown instances (the testing dataset). This results in a higher R^2 score for the training phase and a lower score for the testing phase. According to Prajwala (2015), overfitting is one of the major disadvantages of RF.

All the models produced their lowest R^2 scores when using data from Filter House 1 and 2. LSTM had the worst performance in the training phase (0.319), while RF had the worst performance in the testing phase (0.269). This confirms that R^2 is sensitive to the data structure, especially when the observations are few (Akossou, 2013)



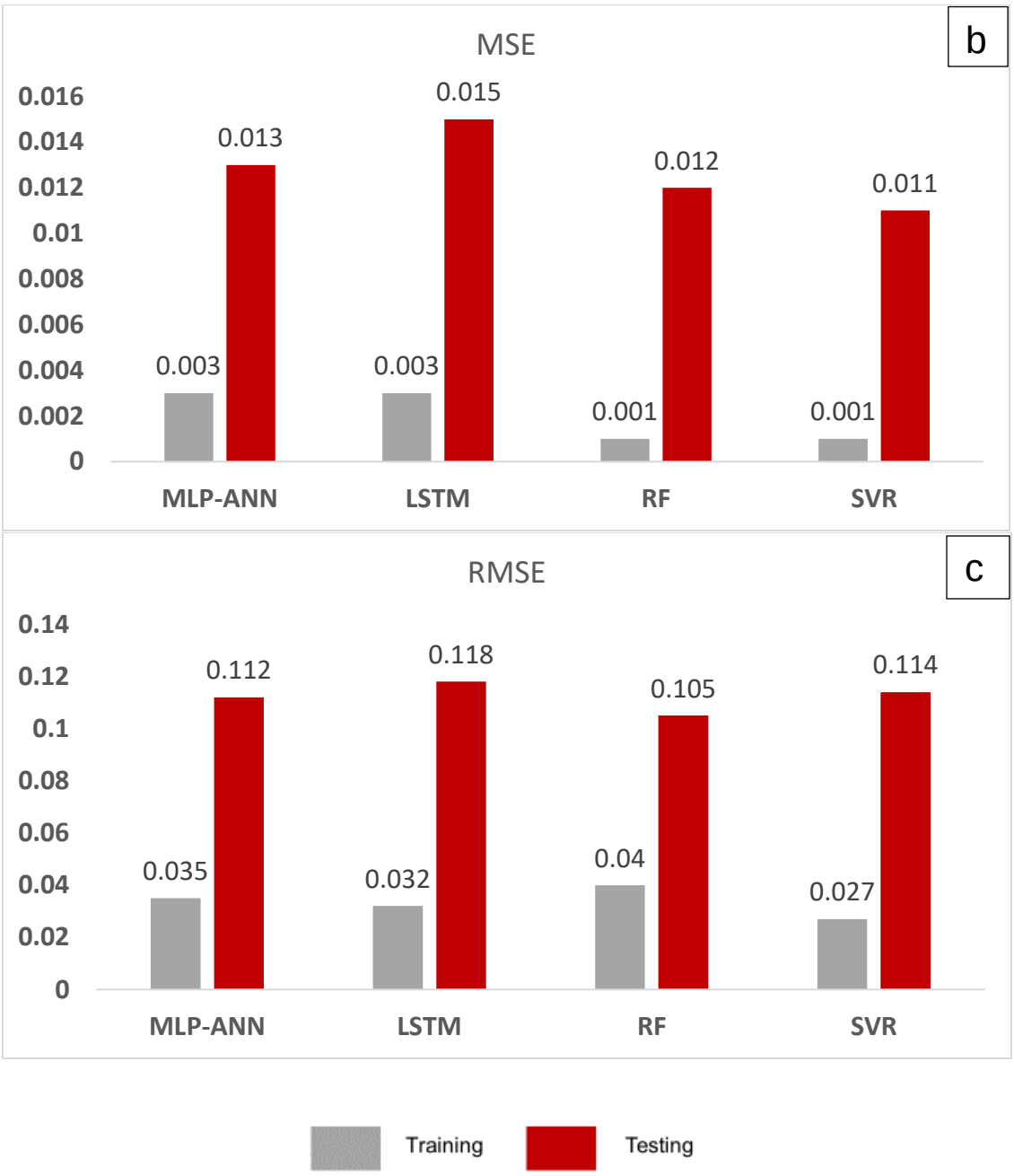


Figure 5.1: Algorithm error averages; (a) $R^{2,}$ (b) MSE, (c) RMSE

Figure 5.1 shows the evaluation metrics averages of each algorithm. Figure 5.1 (a) shows the averages of the R^2 metric. RF produced the highest average training score of 0.676, and LSTM produced the highest average testing score of 0.522. Since R^2 depends on the type of research and research parameters, there is no standard guideline to determine an acceptable prediction score. A general rule of thumb places values below 0.5 as weak, between 0.5 and 0.75 as moderate, and above 0.75 as substantial (Moore et al., 2013; Henseler et al., 2009). Even though the LSTM

algorithm produced the best average testing score, it produced the worst average training score of 0.652. One idea that has been posited to explain the poor training capabilities of the LSTM algorithm is that it requires memory-bandwidth-bound computation. This refers to the physical limits of a device in accessing global memory, which is the computer storage that can be used by a number of processors. Each cell in an LSTM requires at least 4 linear layers. These layers consist of neurons where the information processing occurs (see Chapter 2). The linear layers require large amounts of memory bandwidth to be computed. Memory bandwidth is the rate at which a given amount of data can be moved from or to a particular destination (Culurciello, 2018). SVR produced the worst average testing score of 0.503. In general, SVR algorithms are very poor at forecasting (Yarnold et al., 2013). On average, the algorithms did not perform well in the testing phase. This can be largely attributed to the influence of Filter House 1 and 2, where the algorithms achieved very poor performances.

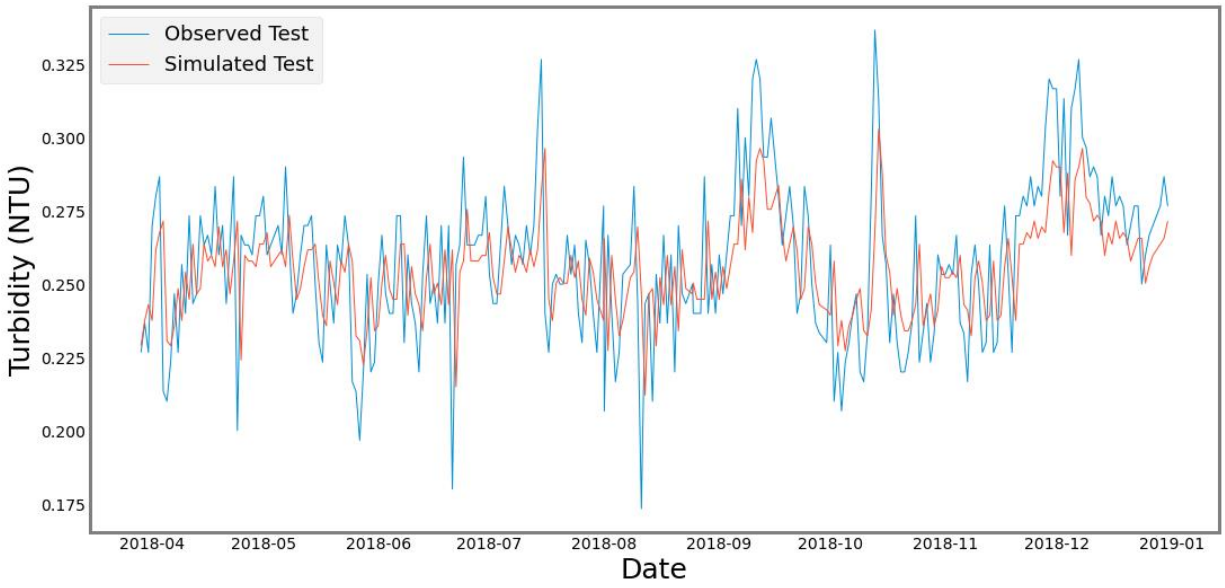
Figure 5.1 (b) shows the averages of the MSE metric. SVR and RF produced the lowest average training errors of 0.001 NTU^2 each, while SVR produced the lowest average testing error of 0.011 NTU^2. Both LSTM and MLP-ANN produced the highest average training errors of 0.003 NTU^2 each, while LSTM produced the highest average testing error of 0.015 NTU^2. When the R^2 metric was considered, LSTM had the best performance, which means it had the best fit (on average, 52.2% of the data points fit the algorithm). However, since it had the highest MSE value, this means that some of the data points that did not fit the algorithm well were greatly dispersed from the regression line as compared to the other algorithms. While the fit was the best relative to the other algorithms, LSTM underperformed when trying to identify underlying trends to the measurement of turbidity.

Figure 5.1 (c) shows the average of the RMSE metric. SVR produced the lowest average training error of 0.027 NTU, while RF produced the lowest average testing error of 0.105 NTU. However, RF produced the highest average training error of 0.040 NTU, while LSTM produced the highest average testing error of 0.118 NTU. As stated above, the MSE and RMSE errors are very small, which means overall all the algorithms did well.

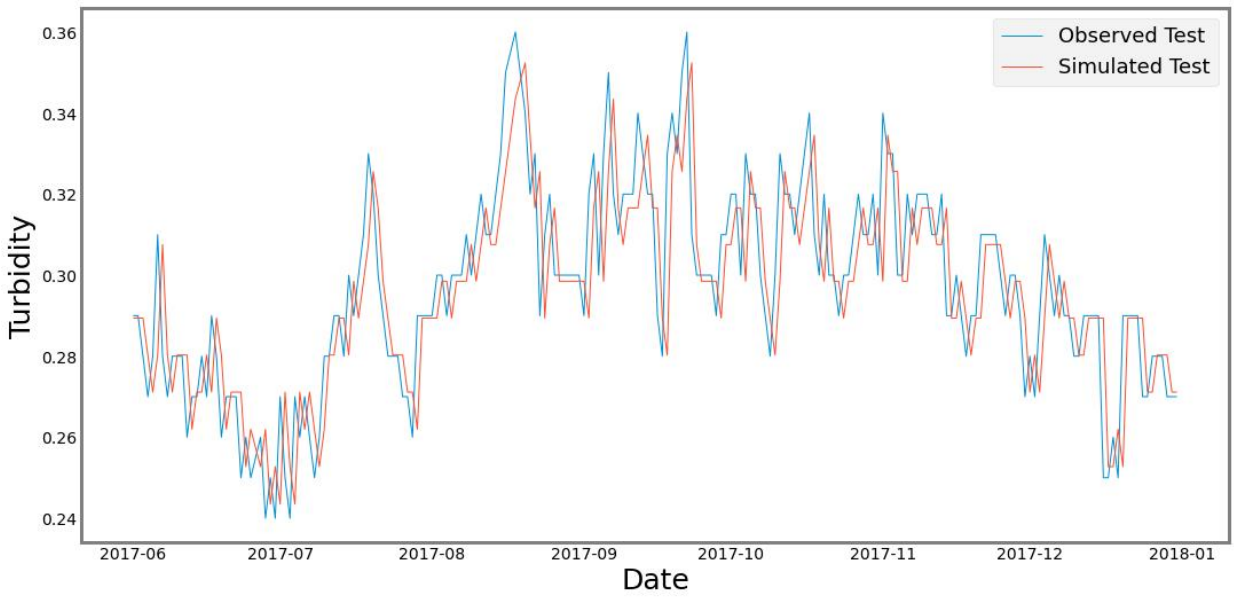
Since the algorithm performances observed a similar trend for all the filter houses, where their worst performances were observed using data from Filter House 1 and 2 and their best performances were observed using data from Filter House 3, the regressions of the LSTM

algorithm, which was the best performing algorithm in terms of R^2, will be used to explain the structure of the testing datasets of each filter house and how that could have affected the predictions.

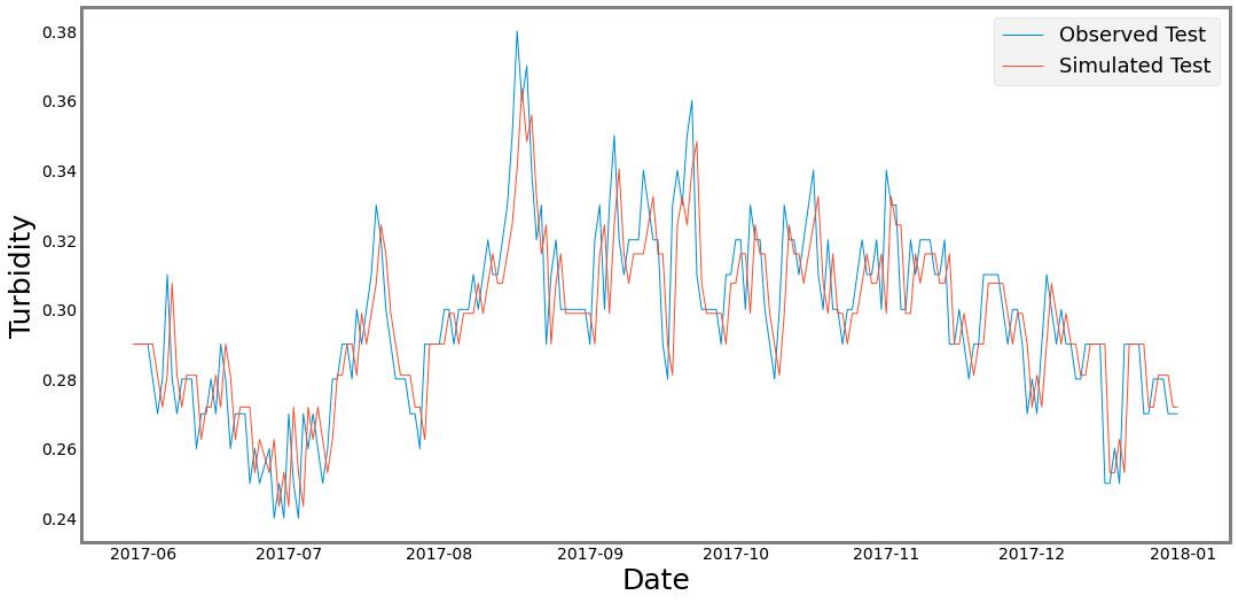
(a) Filter House 1 and 2



(b) Filter House 3



(c) Filter House 4 A and B



(d) Filter House 4 C

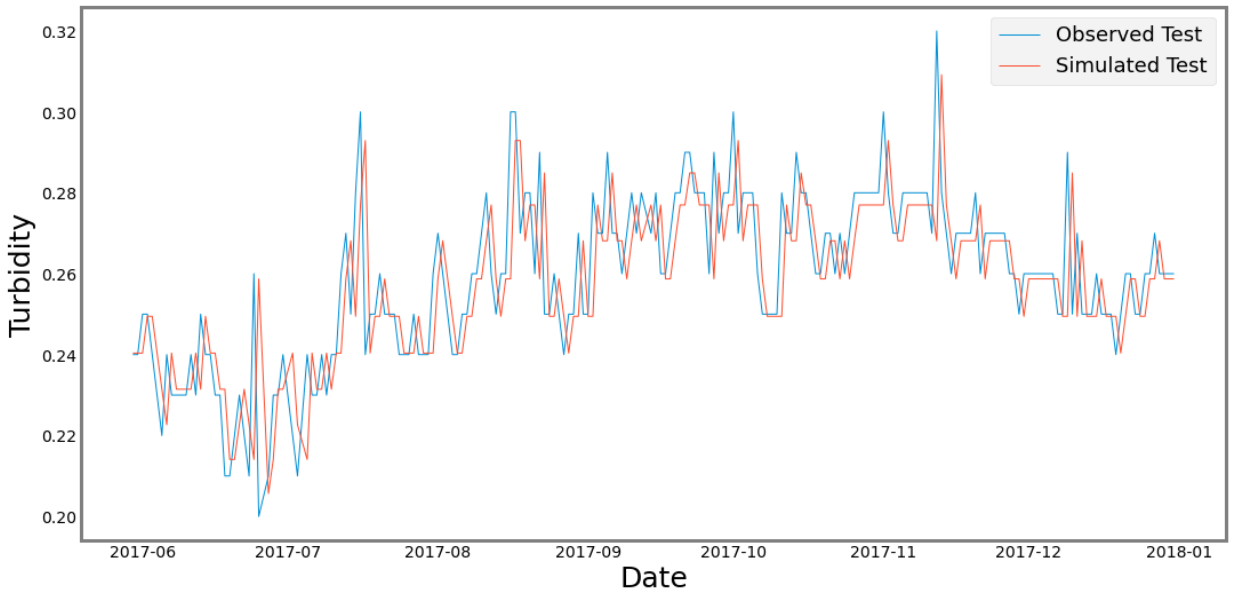


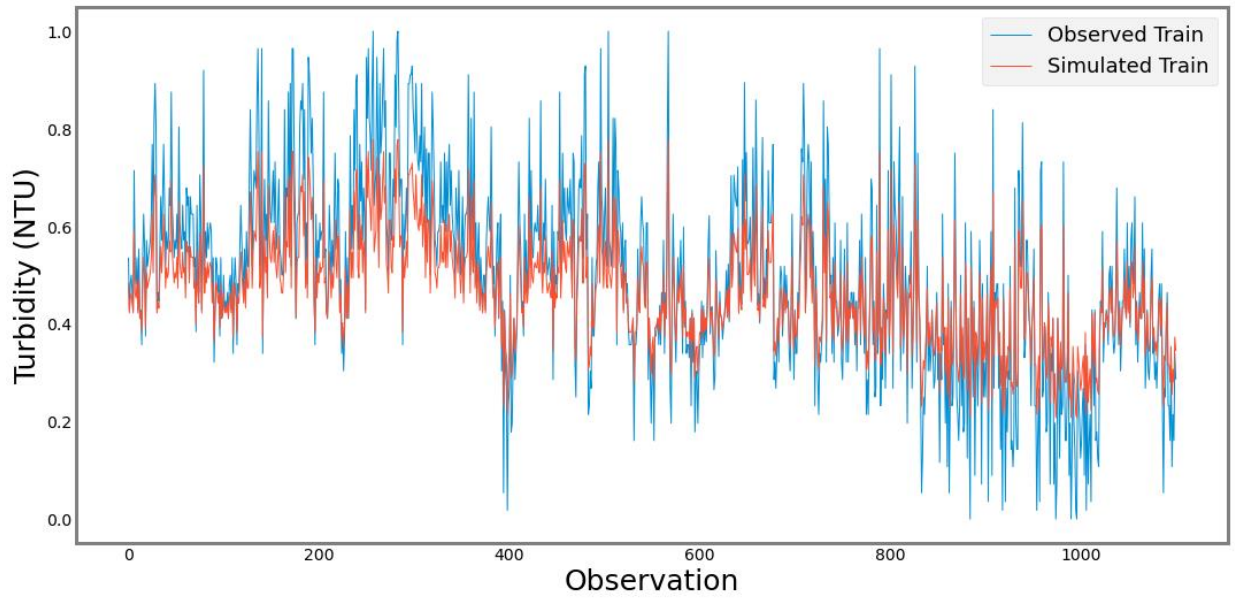
Figure 5.2: Time-series of actual vs predicted turbidity for the testing phase using the LSTM algorithm for (a) Filter Houses 1 and 2 (b) Filter House 3 (c) Filter House 4 A and B and (d) Filter House 4 C

Figure 5.2 suggests that there was a good correlation between the observed test dataset and the simulated dataset. Simulated data is the data produced by the algorithms, meaning it is the

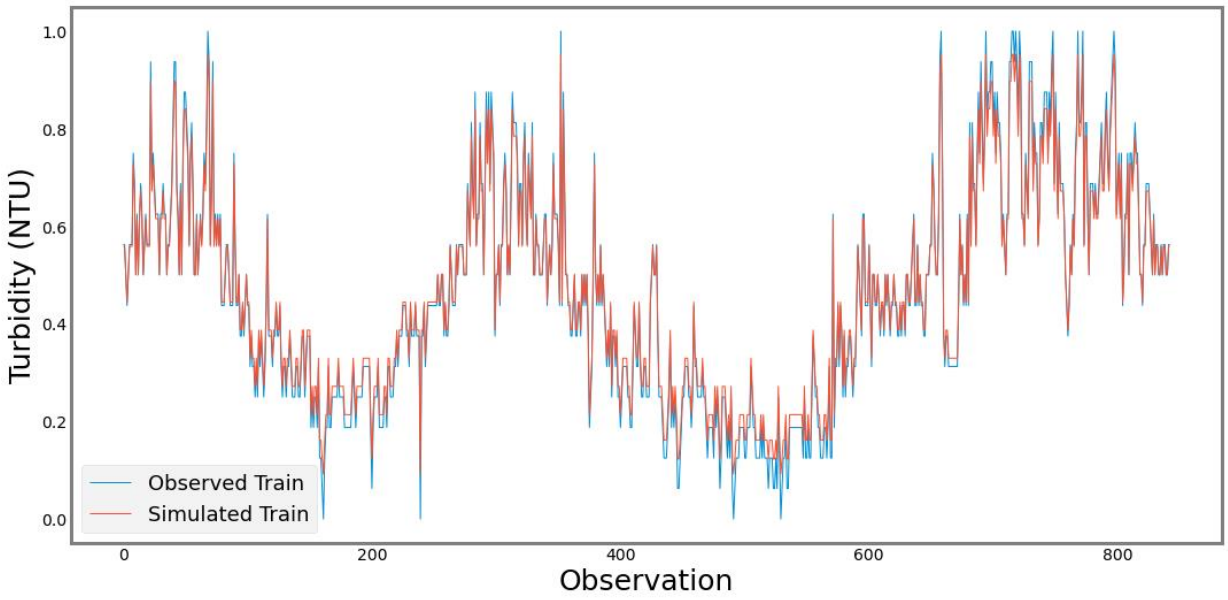
algorithm's predictions. It proves the validity of the algorithm. The correlation was best observed in Filter House 3, Filter House 4 A and B, and Filter House 4 C. The LSTM algorithm in all these aforementioned filter houses produced respectable R^2 values, with Filter House 4 C being the lowest with a score of 0.495, and Filter House 3 being the highest with a score of 0.660. The MSE and RMSE values for all the filter houses were close to zero.

When considering Filter House 3, Filter House 4 A and B and Filter House 4 C, there seems to be little irregularity in the testing dataset, which is not the case for Filter House 1 and 2. In this filter house, peaks and troughs appear randomly throughout the dataset, and the sizes of these are never the same (they are random). This results in a higher degree of variability in the dataset as compared to the other filter houses. Kanyama (2021) stated that the LSTM algorithm performs better with the more variability it sees in the training phase. Because of this, training predictions were studied. Figure 5.3 shows the regression of the training phase of the LSTM algorithm for all the filter houses.

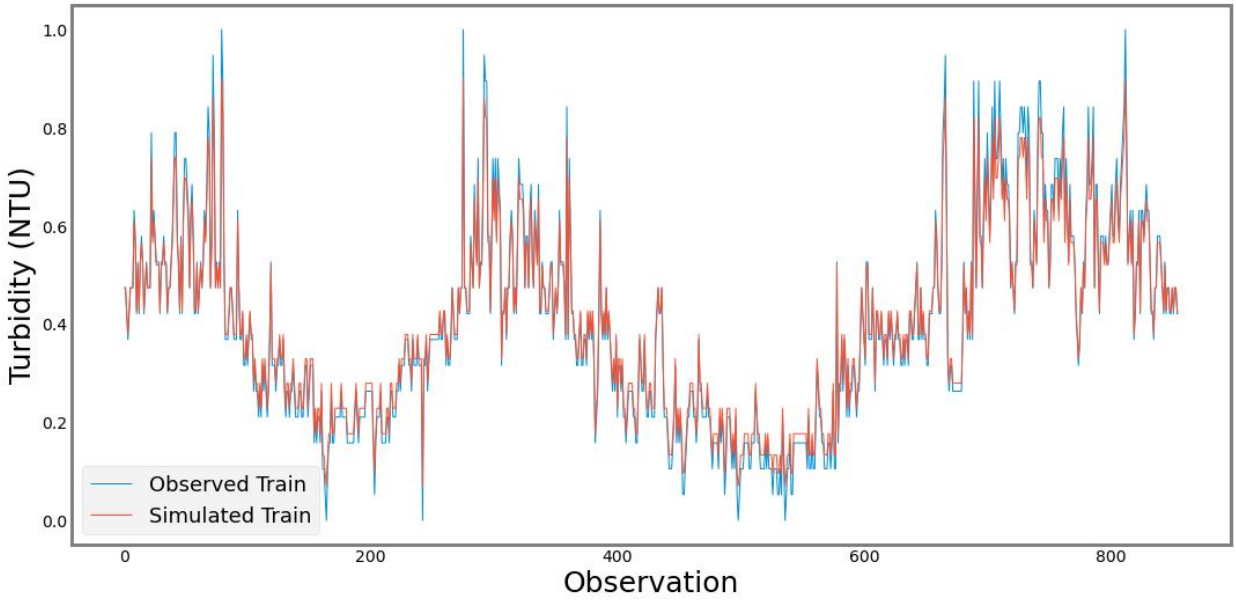
(a) Filter House 1 and 2



(b) Filter House 3



(c) Filter House 4 A and B



(d) Filter House 4 C

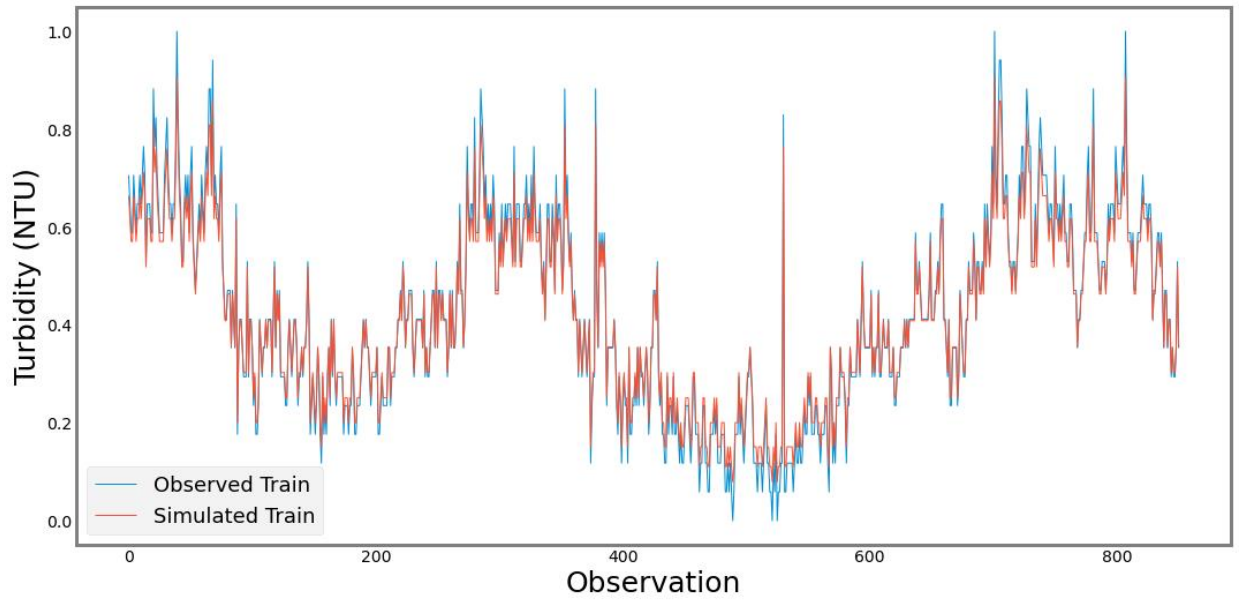


Figure 5.3: Time-series of actual vs predicted turbidity for the training phase using the LSTM algorithm for (a) Filter Houses 1 and 2 (b) Filter House 3 (c) Filter House 4 A and B and (d) Filter House 4 C

Figure 5.3 shows that in the training phase of Filter House 1 and 2, there is also a great degree of variability. This means that Filter House 1 and 2 should have produced a better R^2 score compared to the other filter houses. However, the algorithm was unable to trace the variability observed in the testing phase because the training data is not representative of the testing data (There is a multitude of random peaks and troughs throughout the training dataset that are not present in the testing dataset). This means that the variability learned in the training dataset was different from the variability present in the testing dataset. In the other filter houses, there is a greater similarity between the training and testing datasets, hence the algorithm was able to trace the variability and produce higher R^2 values.

The observed and simulated turbidity levels for Filter House 3 and Filter House 4 A and B were closely matched. This was followed by Filter House 4 C. The worst performing dataset belonged to Filter House 1 and 2.

5.3 Scenario 2 results

This section will discuss the results obtained when predictions were conducted using the complete datasets (Scenario 2). Table 5.3 summarises the results of both the training and testing phases. The size of the training and testing datasets will vary depending on the period length of the filter house.

Table 5.3: Performance of the machine learning techniques for 4 filter houses when the complete datasets are considered

Filter House	Phase	Algorithm	R^2	MSE	RMSE
Filter House 1 and 2	Training	MLP-ANN	-0.001	0.001	0.032
		LSTM	-0.000	0.007	0.084
		RF	0.110	0.003	0.055
		SVR	-2.331	0.011	0.105
	Testing	MLP-ANN	-0.051	0.156	0.395
		LSTM	-0.036	0.155	0.394
		RF	-0.139	0.170	0.413
		SVR	-8.882	1.469	1.212
Filter House 3	Training	MLP-ANN	-0.004	0.001	0.032
		LSTM	-0.003	0.001	0.032
		RF	0.023	0.001	0.032
		SVR	-7.974	0.010	0.101
	Testing	MLP-ANN	-0.170	0.010	0.101
		LSTM	-0.102	0.010	0.101
		RF	-2.776	0.033	0.182
		SVR	-1443.12	12.577	3.546
Filter House 4 A and B	Training	MLP-ANN	-0.002	0.001	0.032
		LSTM	-0.004	0.001	0.032
		RF	0.011	0.001	0.032
		SVR	-8.101	0.010	0.101

Filter House	Phase	Algorithm	R^2	MSE	RMSE
	Testing	MLP-ANN	-1.069	0.001	0.032
		LSTM	-1.075	0.002	0.045
		RF	-3.489	0.051	0.226
		SVR	-11746	7.746	2.783
Filter House 4 C	Training	MLP-ANN	-0.004	0.004	0.063
		LSTM	-0.001	0.001	0.032
		RF	0.238	0.002	0.045
		SVR	-2.111	0.007	0.084
	Testing	MLP-ANN	-1.060	0.010	0.101
		LSTM	-0.149	0.009	0.095
		RF	0.006	0.011	0.105
		SVR	-31.919	6.454	2.540

Table 5.3 shows that with regards to the MSE error metric, RF, LSTM and MLP-ANN produced good predictions for both the training and testing phases for Filter House 3, Filter House 4 A and B, and Filter House 4 C. The same trend was observed for the RMSE error metric.

There is a slight difference when considering predictions using data from Filter House 1 and 2. For the training phase, the MSE errors were 0.003 NTU^2 (RF), 0.007 NTU^2 (LSTM) and 0.001 NTU^2 (MLP-ANN) and the RMSE errors were 0.055 NTU (RF), 0.084 NTU (LSTM) and 0.032 NTU (MLP-ANN). The MSE and RMSE errors for the testing phase were slightly elevated, with MSE errors of 0.170 NTU^2 (RF), 0.155 NTU^2 (LSTM) and 0.156 NTU^2 (MLP-ANN), and RMSE errors of 0.413 NTU (RF), 0.394 NTU (LSTM) and 0.395 NTU (MLP-ANN). While these are still within the acceptable range, it is well noting that there seems to be a greater dispersion of the data points from the regression line for the testing phase for Filter House 1 and 2 when the datasets are not treated for outliers. However, it can be concluded from the observations that outliers, in general, do not seem to significantly affect the dispersion of the data points from the regression line for the algorithms discussed.

The SVR algorithm produced different results. With regards to the MSE and RMSE error metrics, the same trend as the other algorithms was observed for the training phase for all the filter houses.

However, when considering the testing phase, the algorithm produced very high errors. For Filter House 1 and 2, Filter House 3, Filter House 4 A and B and Filter House 4 C, the MSE errors were 1.469 NTU^2, 12.577 NTU^2, 7,746 NTU^2, and 6.454 NTU^2, and the RMSE errors were 1.212 NTU, 3.546 NTU, 2.783 NTU and 2.540 NTU, respectively. These values are very high, which means that the SVR algorithm could not identify underlying contributions to the prediction of turbidity in the testing phase when outliers were included in the datasets. Chuang et al. (2002) suggests that SVR algorithms present the possibility of selecting outliers as support vectors. Kanamori et al. (2014) explains this. Both the SVM and SVR algorithms consist of a feature C, which is the penalty classifier and measures the degree of misclassifications (see Chapter 2). The penalty of misclassifications is defined by a convex loss. A convex loss makes accessing the global optimum solution easier, and the global optimum solution of an optimisation procedure is the optimum value of the objective function compared to all other possible solutions. The unboundedness of this convex loss causes the sensitivity of these algorithms to outliers in the training sample, hence affecting the overall performance in predicting future cases.

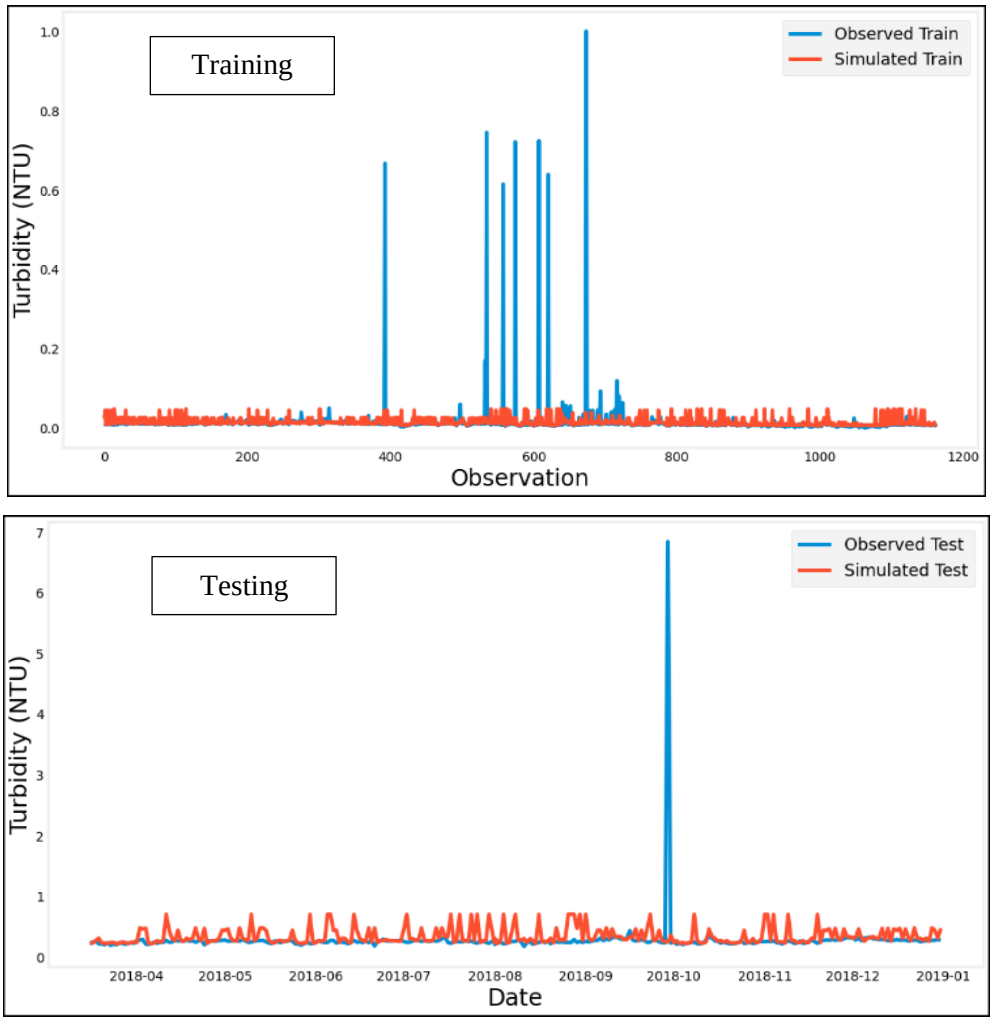
With regards to the R^2 metric, all the algorithms observed poor performances. Apart from RF, all the algorithms produced negative scores for both the training and testing phases, which means that they performed worse than the mean value. The SVR algorithm observed the worst performance with scores of -8.101 and -11746 for the training and testing phases, respectively, for Filter House 4 A and B.

The RF algorithm produced different results. For the training phase, the algorithm produced R^2 scores of 0.110 (Filter House 1 and 2), 0.023 (Filter House 3), 0.011 (Filter House 4 A and B) and 0.238 (Filter House 4 C). This means that the algorithm was able to pick up some trends in the datasets, even though it was very poorly modelled. For the testing phase, the algorithm produced scores of -0.139 (Filter House 1 and 2), -2.776 (Filter House 3), and -3.489 (Filter House 4 A and B), which are all negative scores. This means that while the algorithm was able to find a trend in the training phase, when new unknown instances were introduced (the testing dataset), the algorithm completely failed. The only difference in the predictions was observed for Filter House 4 C. Here the algorithm produced positive scores for both the training and testing phases, with values of 0.238 and 0.006, respectively. The algorithm was able to identify a trend, though very poorly. A proposed reason why RF can cater for outliers relative to the other algorithms is because

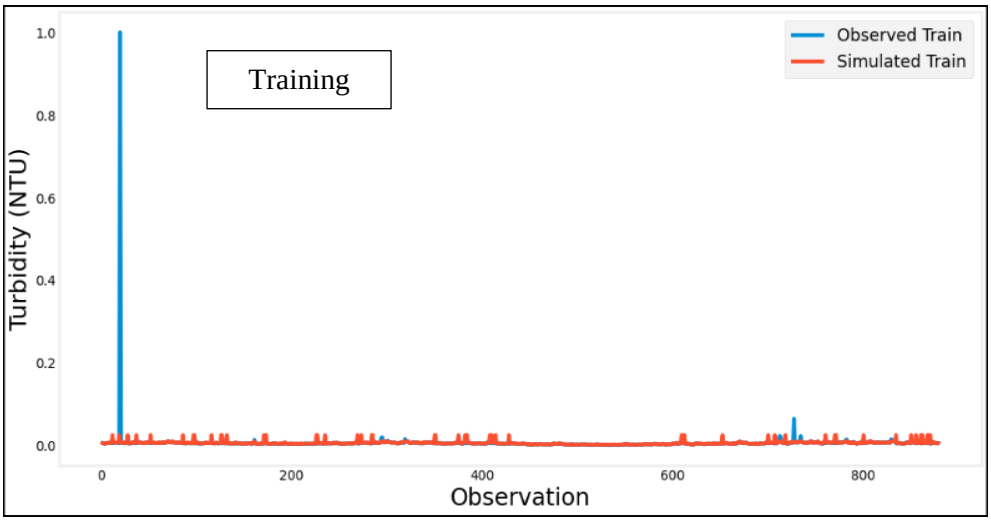
the algorithm uses trees, which split the data into groups. This means that outliers only affect the results of the leaf node they are in, and not the estimates of other leaf nodes (Elg, 2018; Diaz-Uriarte & Alvarez de Andres, 2006).

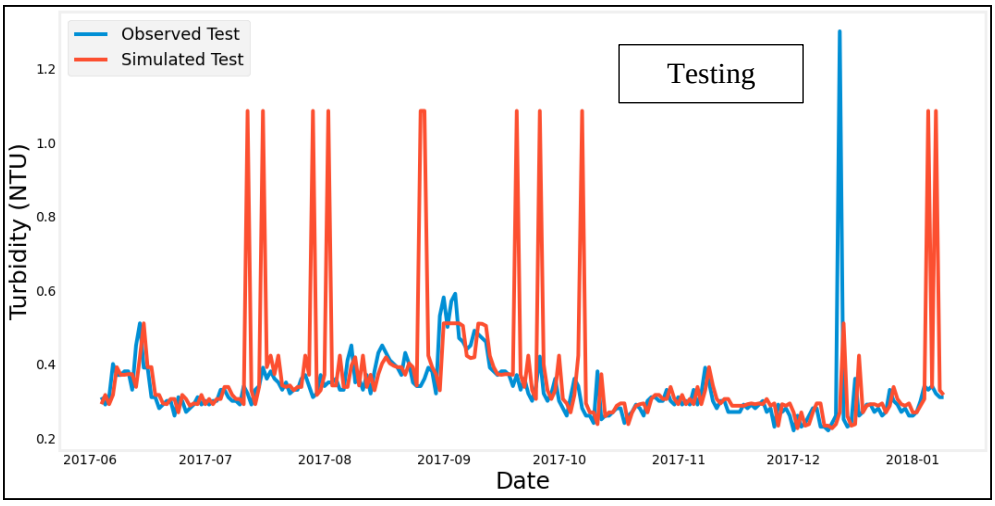
When outliers were included in the analysis, the networks performed worse with respect to the R^2 metric. This was also reported by (Satapathy et al., 2015). As was done in this study (Section 5.2), the researchers proceeded to remove the outliers. However, since outliers can arise from a variety of reasons, removal requires careful consideration (Z. Chen et al., 2021; Li et al., 2021; Nourani & Behfar, 2021). Nonetheless, The RF algorithm had the best performance. Because of this, it was used to plot the time-series regressions presented in Figure 5.4.

(a) Filter House 1 and 2

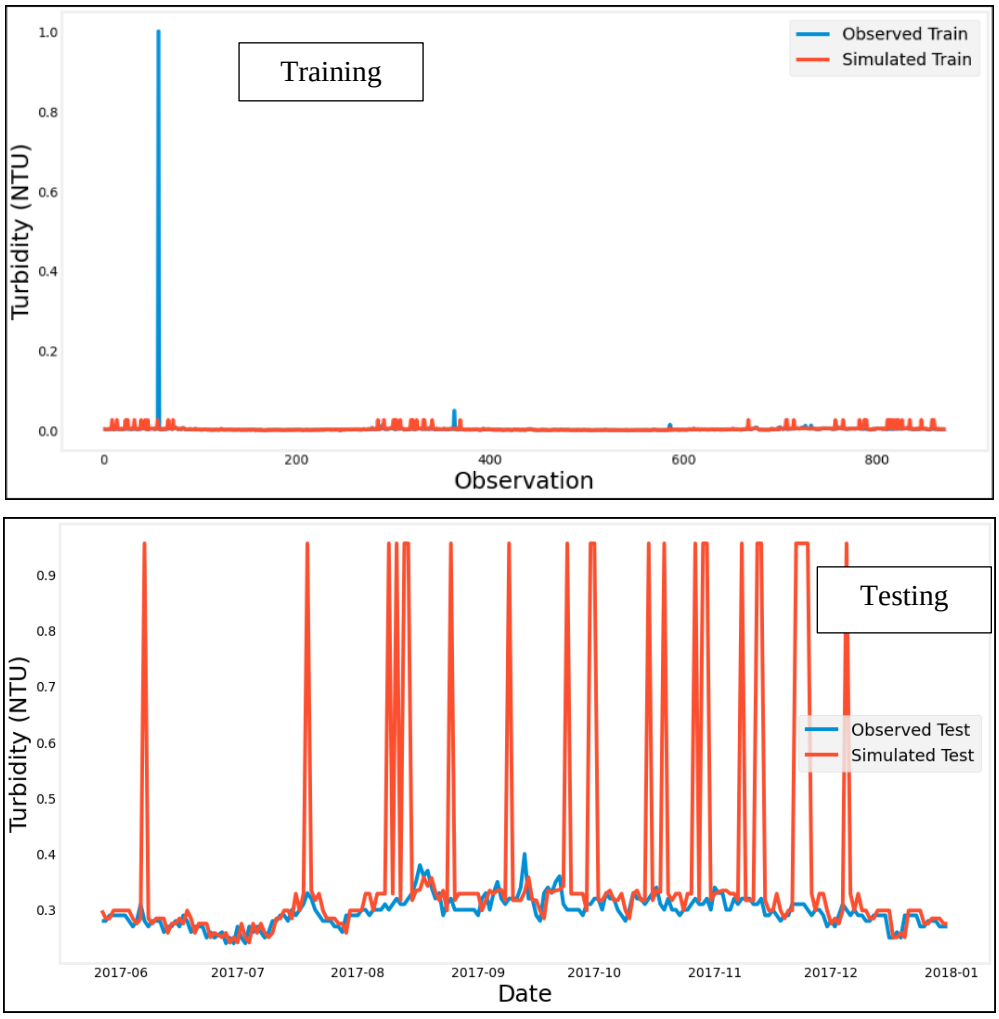


(b) Filter House 3





(c) Filter House 4 A and B



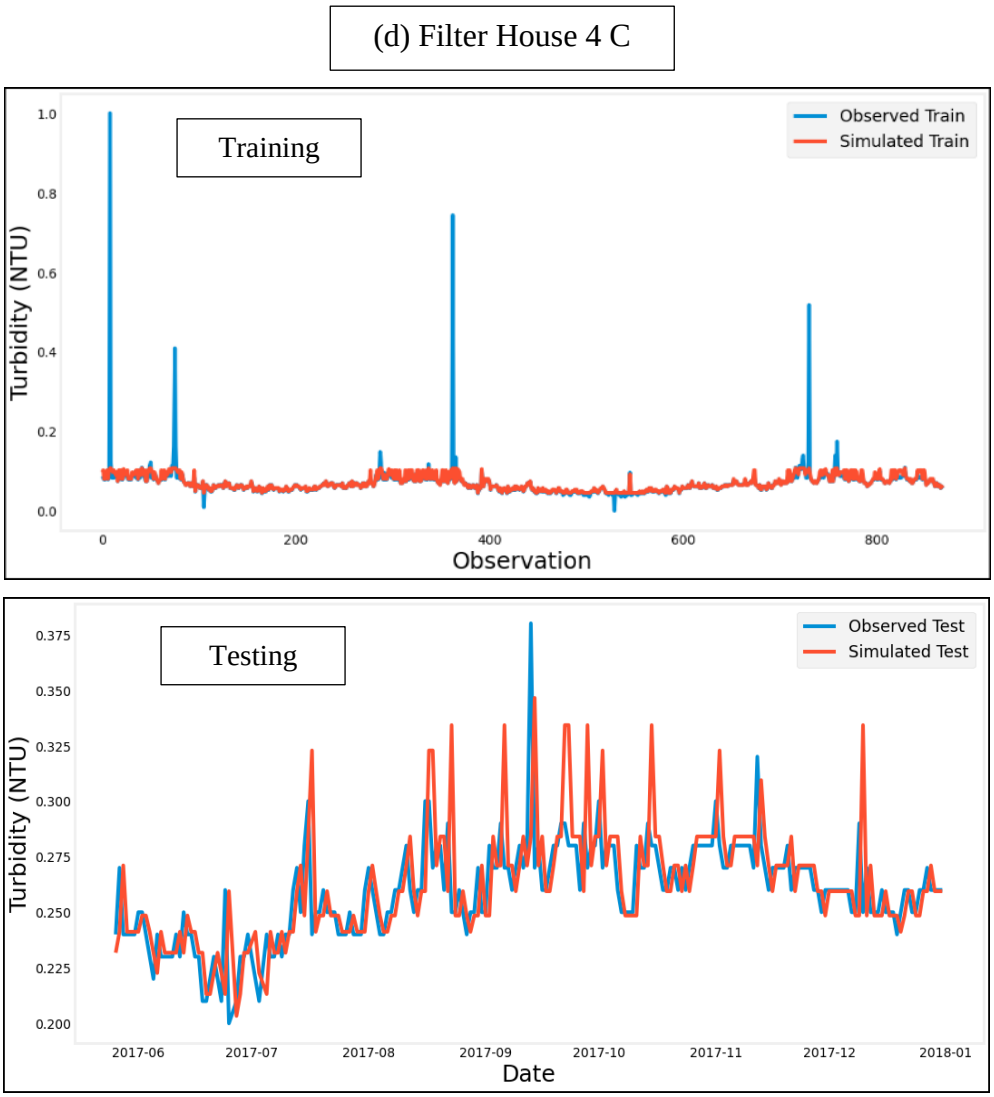


Figure 5.4: Time-series of actual vs predicted turbidity for both the training and testing phases using the RF algorithm for (a) Filter Houses 1 and 2 (b) Filter House 3 (c) Filter House 4 A and B and (d) Filter House 4 C

Figure 5.4 suggests that there were very poor correlations between observed and simulated values for the testing phase. The algorithm seems to have overestimated some of the values. It is difficult to study the training phase due to the high peaks present in the datasets. These are clearly outliers, and these outliers have drastically affected the prediction of the training phase, even though they did not result in negative scores as seen in the testing phase. This could be because the algorithm was optimised using the training datasets.

With regards to Filter House 4 C, the only filter house which produced a positive R^2 score in the testing phase (0.006), the algorithm seems to have tried to not overestimate the values in the testing phase, at least relative to the other filter houses. In the training phase of Filter House 4 C, there are several high peaks that are scattered throughout the dataset. This variability could be the reason for the improved prediction of Filter House 4 C relative to the other filter houses. Algorithms, as stated in Section 5.2, seem to perform better the more variability they see in the training phase. But in the training phase of Filter House 1 and 2, there are also several peaks present. These, however, are clustered at one point in the dataset (approximately from observation 400 to 700) and are not spread throughout the dataset, resulting in less variability. This is also the case for Filter House 3 and Filter House 4 A and B. Both filter houses have one high peak at the beginning of the time-series, but the series remains constant thereafter.

It can be concluded from the results that generally, the RF algorithm performs better than the other algorithms when outliers are included in the datasets. SVR had the worst performance. Overall, algorithms produce poor R^2 scores when they are not treated for outliers.

5.4 Case test results

In this section, the results of two sensitivity tests are discussed. These were done to see how the algorithm results could be improved and how some factors intrinsic to machine learning affect algorithm performance. The cases are described in Table 5.4.

Table 5.4: Description of each case

Case ID	Case Description
A	Test for the influence of overfitting in the LSTM and MLP-ANN algorithm by creating 2 scenarios. Scenario 1 consists of early stopping and scenario 2 does not.
B	Create new regressions with larger synthetic datasets.

5.4.1 Case A

In Case A, the influence of overfitting on LSTM and MLP-ANN (the neural networks) is discussed. As stated, overfitting occurs when the algorithm corresponds too closely to the training dataset that it fails to predict additional data or future instances. It is difficult to discern whether an algorithm is overfitting during the training phase. For the neural networks, the early stopping function in Python can sometimes help remove the influence of overfitting by stopping the training when the loss function (in this study, the MSE metric) remains constant over several iterations (in this study, after 3 iterations). RF and SVR do not have this function and because of that, they were not included in the analysis. The only way to remove overfitting in these algorithms is to finetune the hyperparameters. This is a trial-and-error method (See Chapter 4).

Two scenarios were developed. Scenario 1 are predictions with early stopping, and Scenario 2 are predictions without early stopping. Table 5.5 shows the results of the neural networks. Note, the regressions were compiled on datasets where possible outliers were removed (datasets from Section 5.2).

Table 5.5: Performance of the machine learning techniques when two scenarios are considered. Scenario 1 are predictions with early stopping, and Scenario 2 are predictions without early stopping

Filter House	Scenario	Phase	Algorithm	R^2	MSE	RMSE
Filter House 1 and 2	Scenario 1	Training	MLP-ANN	0.345	0.001	0.032
			LSTM	0.280	0.001	0.032
		Testing	MLP-ANN	0.272	0.025	0.157
			LSTM	0.284	0.027	0.165
	Scenario 2	Training	MLP-ANN	0.356	0.001	0.032
			LSTM	0.321	0.001	0.032
		Testing	MLP-ANN	0.202	0.025	0.157
			LSTM	0.272	0.026	0.161
Filter House 3	Scenario 1	Training	MLP-ANN	0.775	0.004	0.063
			LSTM	0.770	0.002	0.045

Filter House	Scenario	Phase	Algorithm	R^2	MSE	RMSE
		Testing	MLP-ANN	0.651	0.012	0.110
			LSTM	0.674	0.011	0.105
	Scenario 2	Training	MLP-ANN	0.778	0.000	0.000
			LSTM	0.773	0.010	0.099
		Testing	MLP-ANN	0.632	0.011	0.107
			LSTM	0.650	0.013	0.114
Filter House 4 A and B	Scenario 1	Training	MLP-ANN	0.754	0.002	0.045
			LSTM	0.752	0.001	0.032
		Testing	MLP-ANN	0.631	0.010	0.100
			LSTM	0.643	0.018	0.134
	Scenario 2	Training	MLP-ANN	0.759	0.006	0.075
			LSTM	0.761	0.004	0.063
		Testing	MLP-ANN	0.611	0.010	0.100
			LSTM	0.637	0.017	0.130
Filter House 4 C	Scenario 1	Training	MLP-ANN	0.749	0.009	0.096
			LSTM	0.737	0.010	0.100
		Testing	MLP-ANN	0.498	0.014	0.118
			LSTM	0.501	0.015	0.122
	Scenario 2	Training	MLP-ANN	0.756	0.011	0.105
			LSTM	0.746	0.004	0.063
		Testing	MLP-ANN	0.450	0.019	0.138
			LSTM	0.491	0.023	0.152

Table 5.5 shows that with regards to the MSE and RMSE metrics, all the algorithms observed good performances for both the training and testing phases for all the filter houses, as the values were very close to zero. There is no trend to differentiate the performance of the algorithms between the two scenarios using these metrics.

With regards to the R^2 metric, the algorithms performed better in the training phase for Scenario 2 for all the filter houses. The best performances were obtained in Filter House 3, where the scores

were 0.778 (MLP-ANN) and 0.773 (LSTM). From this, it can be concluded that when early stopping is omitted, the algorithms observe the best training. For the testing phase, both algorithms performed better in Scenario 1 for all the filter houses. The best scores were once again obtained from Filter House 3, which were 0.651 (MLP-ANN) and 0.674 (LSTM). From this, it can be concluded that when early stopping is included in the analysis, the algorithms can predict the testing dataset better.

The easiest way to confirm overfitting is when the training error is significantly lower than the testing error. This means that if there is a greater gap between the training error and testing error, the algorithm is overfitting. In this study, equation 5.1 was used to calculate the relative percentage error between the training phase and the testing phase for each scenario (Kat & Els, 2012):

$$relative\ percentage\ error = \left| \frac{testing\ error - \ training\ error}{training\ error} \right| x\ 100 \quad (5.1)$$

The higher the relative percentage error, the bigger the difference between the training and testing errors. This, however, is not a suitable method to use when trying to correct algorithms for overfitting, because this method assumes that the training error is perfect (that the training error is the one the algorithm should attain (the real/expected error)). However, for comparative analysis, it can provide some insight into the differences between the established scenarios. The R^2 averages of each algorithm for both the training and testing phases (Table 5.6) were used to calculate the relative errors (Table 5.7).

Table 5.6: The averages of the R2 metric across all 4 filter houses for the MLP-ANN and LSTM algorithms for Scenario 1 and 2, for both the training and testing phases

Phase	**Scenario**	**Algorithm**	**R^2**
Training	1	MLP-ANN	0.656
		LSTM	0.635
	2	MLP-ANN	0.662
		LSTM	0.650

Phase	Scenario	Algorithm	R^2
Testing	1	MLP-ANN	0.513
		LSTM	0.519
	2	MLP-ANN	0.473
		LSTM	0.513

Table 5.7: Relative errors of the MLP-ANN and LSTM algorithm between the training and testing errors for both scenarios

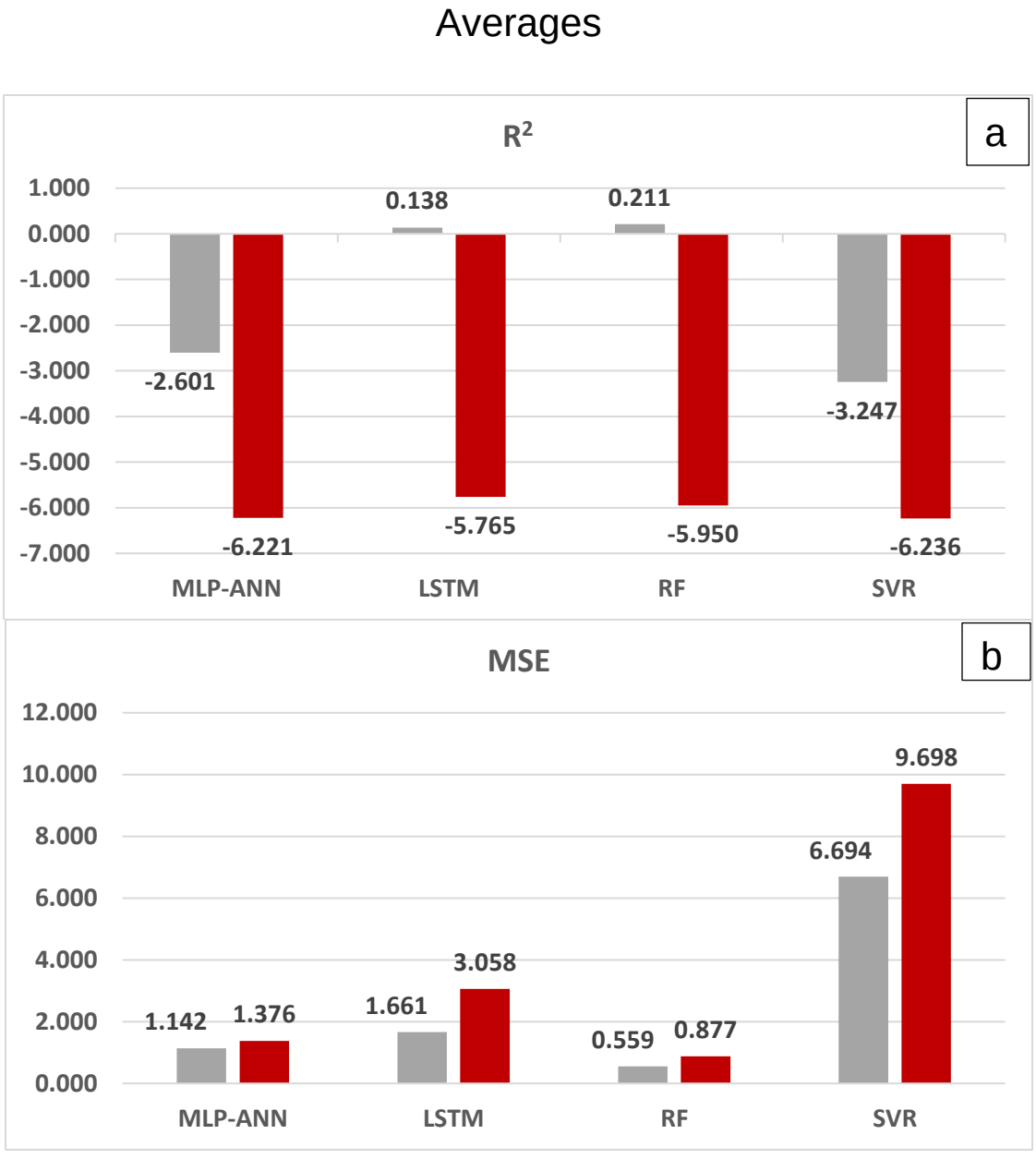
Algorithm	Scenario	Relative error
MLP-ANN	1	21.8%
	2	28.5%
LSTM	1	18.3%
	2	21.1%

Table 5.7 shows that the relative errors of the algorithms increase from Scenario 1 (with 21.8% for the MLP-ANN and 18.3% for the LSTM) to Scenario 2 with (28.5% for the MLP-ANN and 21.1% for the LSTM). This suggests that when early stopping was included in the analysis (Scenario 1), there was a smaller gap between the training error and the testing error, as compared to when early stopping was excluded (Scenario 2). From this, it can be concluded that without early stopping, the neural networks tend to overfit the data and overfitting does reduce the capabilities of the neural networks to forecast.

5.4.2 Case B

In Case B, the influence of data augmentation on algorithm predictions is discussed. Data augmentation is a technique in which new synthetic data is created from the original data to increase the overall size of the dataset. This is done on the training dataset and not on the testing dataset, as the testing dataset estimates how the algorithm works on real-world data. The augmented dataset exists to improve the training of the algorithm.

Augmentation was done on datasets that excluded outliers (datasets from Section 5.2). The augmentation process was random. Figure 5.5 shows the best obtained average results.



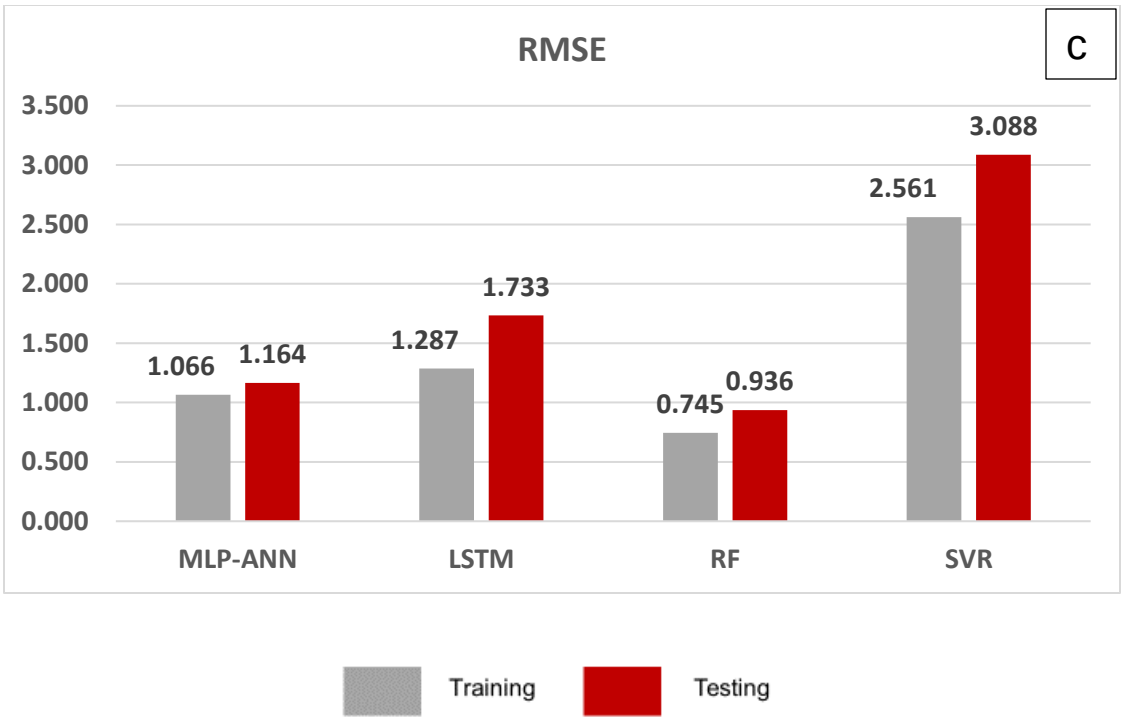


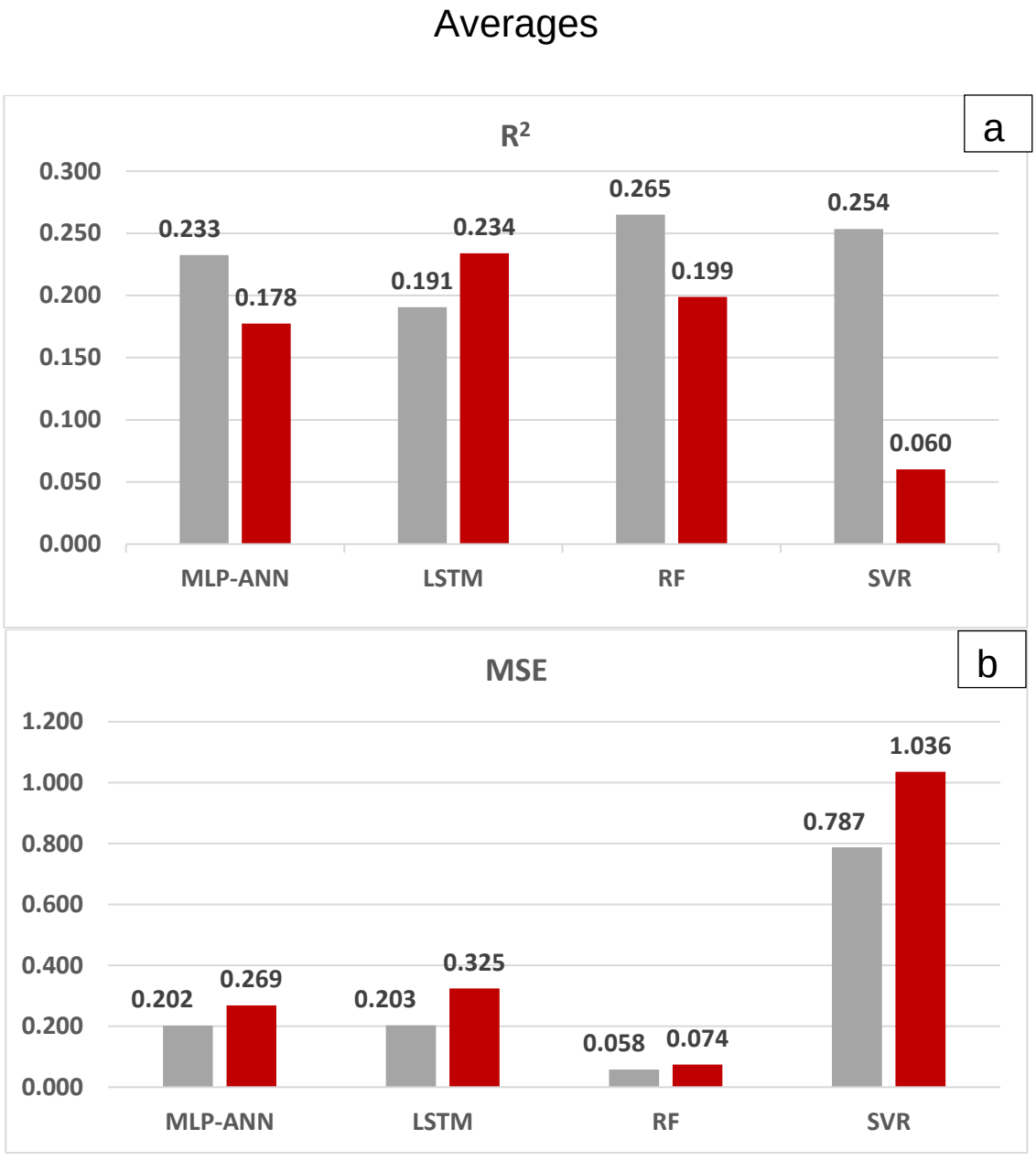
Figure 5.5: Algorithm error averages for predictions on datasets that excluded outliers (Scenario 1 datasets) that were augmented; (a) R^2, (b) MSE, (c) RMSE.

Figure 5.5 (a) shows the averages of the R^2 metric. When the datasets were augmented, the overall performances of all the machine learning algorithms greatly digressed. The average training scores were -2.601 (MLP-ANN), 0.138 (LSTM), 0.211 (RF) and -3.247 (SVR). MLP-ANN and SVR produced negative R^2 scores, which means the algorithms' training performed worse than the mean value. LSTM and RF had poor positive scores, which means that while they were able to detect a trend, the trend was poorly modelled. While generally, LSTM is the hardest of the algorithms to train, it performed better than the MLP-ANN and SVR, suggesting that when more data is used, the algorithm can learn better. The testing phase also produced poor results, with average testing scores of -6.221 (MLP-ANN), -5.765 (LSTM), -5.950 (RF) and -6.236 (SVR). As expected, SVR had the worst performance.

Figure 5.5 (b) and (c) show the averages of the MSE and RMSE metrics, respectively. All the algorithms produced very high errors, which means they could not identify the underlying contributions to the prediction of turbidity. The RF algorithm produced the least errors. For the MSE metric, the average errors were 0.559 NTU^2 and 0.877 NTU^2 for the training and testing phases, respectively. For the RMSE metric, the average errors were 0.745 NTU and 0.936 NTU

for the training and testing phases, respectively. Data augmentation seems to affect the dispersion of the data points from the regression line.

Since the R^2 scores produced were similar to those produced in Section 5.3, the augmentation procedure may have re-introduced outliers into the datasets. As such, the augmented datasets were re-treated for outliers using (4.1) and (4.2). Figure 5.6 shows the average results for the training and testing phases. These were the best-obtained average results.



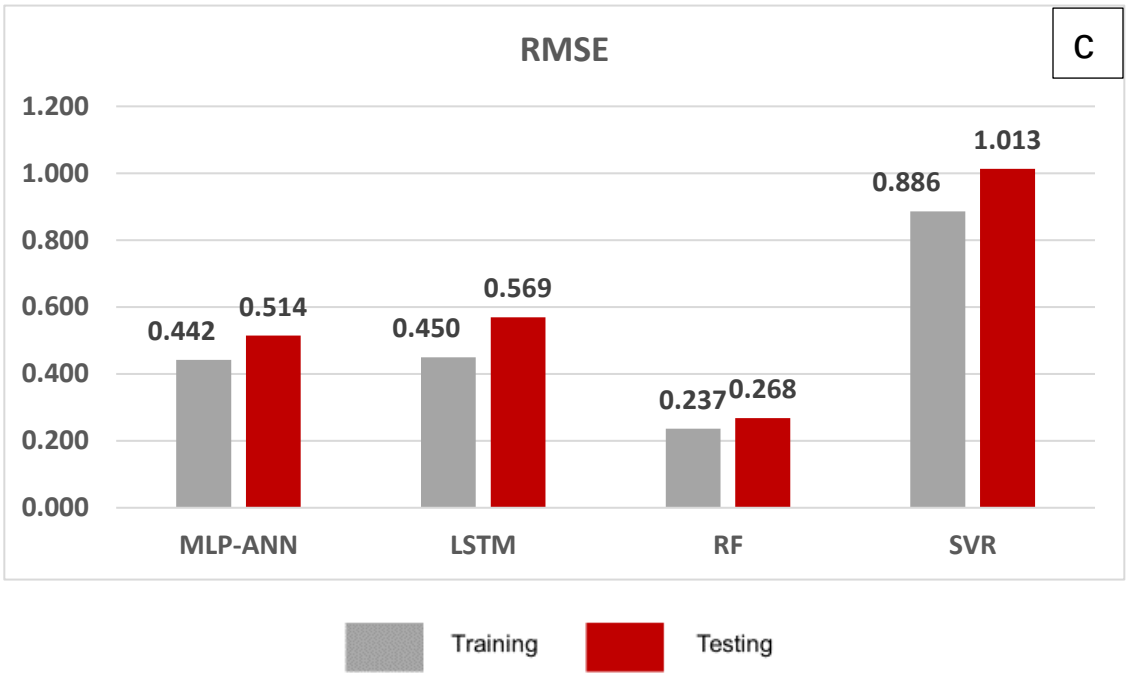


Figure 5.6: Algorithm error averages for predictions on datasets that excluded outliers (Scenario 1 datasets) that were augmented, which were further treated for outliers; (a) R^2, (b) MSE, (c) RMSE.

Figure 5.6 (a) shows that with regards to the R^2 metric, the models performed slightly better when the datasets were further treated for outliers. Like the results in Section 5.2, RF produced the highest average training score of 0.265 and LSTM produced the highest average testing score of 0.234. LSTM produced the lowest average training score of 0.191 and SVR produced the lowest average testing score of 0.060. While there is an improvement, the algorithms underperformed when compared to the results produced in Section 5.2. This means that data augmentation did not help to improve the predictions. Since overall, both the training and testing scores are low, it can be concluded that the algorithms were underfitting the data. Data augmentation introduces variety in the training dataset which reduces overfitting (Iwana & Uchida, 2021; Su et al., 2021; Jiang et al., 2021). However, sometimes the data augmentation process can result in underfitting (Sikka, 2020).

Figure 5.6 (b) shows the averages of the MSE metric. Except for SVR, the algorithms observed good performances as all the errors were very close to zero. RF produced the least errors, with an average training error of 0.058 NTU^2 and an average testing error of 0.074 NTU^2. This means that data augmentation does not greatly influence the dispersion of the data points when the augmented

data are re-treated for outliers. As shown in Figure 5.2 (b), if the augmented data is not treated for outliers, it greatly affects the dispersion of the data points. SVR observed the worst performance, with an average training error of 0.787 NTU^2 and an average testing error of 1.036 NTU^2, making it the only exception to the analysis.

Figure 5.6 (c) shows the averages of the RMSE metric. Except for RF, the algorithms observed slightly poor performances, with SVR producing the highest average errors of 0.886 NTU (training phase) and 1.013 NTU (testing phase). RF observed a good performance with an average training error of 0.237 NTU and an average testing error of 0.268 NTU. This is expected for any discrepancies introduced by the augmentation process will only affect the prediction of a single leaf node and not the whole algorithm.

Generally, the study found that when the data is augmented, the algorithms observed poorer performances than when the data is kept in its original form.

Chapter 6: Conclusions and recommendations

This chapter will present the conclusion to the study as well as recommendations for any future work on possible ways the performance of the machine learning algorithms could be improved.

6.1 Conclusions

The study made use of data obtained from Rand Water measuring turbidity (NTU). Four machine learning techniques were developed using existing code libraries in Python. Several scenarios were tested to assess how the algorithms handle different situations and to compare their performances on such data.

Scenario 1 looked at predictions when datasets were treated for outliers. All the algorithms observed commendable performances with respect to the MSE and RMSE metrics, which were very low. This means that they could capture trends in the observed turbidity levels. The LSTM algorithm observed the best performance with respect to the R^2 metric, producing an average testing score of 0.522. This means that 52.2% of the data points fit the model. A consensus grades this level of correlation as moderate. SVR observed the worst performance with an average testing score of 0.503.

All the algorithms produced good R^2 scores when using data from Filter House 3. The worst performances were observed when using data from Filter Houses 1 and 2. This was largely because the variation that the algorithms learned from the training phase was different from the variation present in the testing dataset, resulting in poor predictions. Variation in this case refers to the number of irregular peaks present in the dataset.

Scenario 2 looked at predictions when the datasets were not treated for outliers. All the algorithms observed very poor performances with respect to the R^2 metric as would be expected given the sensitivity of this metric. Except for RF, all the algorithms produced negative scores in both the training and testing phases. This means they did worse than the mean value. RF produced positive scores for the training phase for all the filter houses. For the testing phase, the algorithm produced negative scores except for Filter House 4 C in which the algorithm produced a positive score. For Filter House 4 C, the training and testing phases scores were 0.238 and 0.006, respectively. This means that the algorithm was able to pick up a trend, although it was very poorly modelled. RF

observed the best performance (relative to the other algorithms) when possible outliers were not omitted and SVR observed the worst performance.

With respect to the MSE and RMSE metrics, except for SVR, the algorithms observed good performances in both the training and testing phases for Filter House 3, Filter House 4 A and B and Filter House 4 C. For Filter House 1 and 2, the training phase errors were also good, however the testing phase errors were slightly elevated. This means that outliers do seem to affect the dispersion of the datasets from the regression line. However, since the errors were still in the acceptable range (<0.5), the influence is insignificant.

With the SVR algorithm, the same trend as the other algorithms was observed for the training phase for all the filter houses. The testing phase produced very poor errors, with Filter House 3 producing the highest MSE (12.577 NTU^2) and RMSE (3.546 NTU) errors. This means that when outliers were included in the analysis, SVR could not detect the trends in the prediction of turbidity.

The RF predictions were used to investigate the structure of the dataset of each filter house. The best performances were observed using data from Filter House 4 C in both the training (0.238) and testing (0.006) phases. This filter house observed the greatest variability in the training phase. This resulted in the best training which produced the best scores when predicting future observations. However, this is with respect to the other algorithms, as generally, the R^2 scores produced by the RF algorithm are poor.

The influence of outliers on the neural networks was also tested. The early stopping function of Python was employed to filter out overfitting. The study found that when early stopping is not used, the networks do overfit, and that overfitting poorly influences the performance of the algorithms.

The influence of data augmentation was also tested. The study found that when the data is augmented, the machine learning algorithms produce poorer predictions. The predictions were slightly better when the augmented datasets were treated for outliers, but they were still poor. Overall, data augmentation negatively affected the machine learning algorithm's predictions.

The study was a univariate analysis, which means that it involved only one variable (turbidity). This was limited as a result of the data that was available for study. To improve the quality of the predictions, more variables can be included, resulting in a multivariate analysis. More variables

can reveal additional underlying contributions to the level of turbidity predicted. Another limitation confronted by the study was the size of the datasets. Using more data can result in better training, which in turn can produce better results. This can form part of any future work.

The study reveals several important factors. The first is that the LSTM algorithm is best suited for predictive analysis when the datasets are treated for outliers and the RF algorithm is best suited for when the datasets are not treated for outliers. Secondly, data from water treatment facilities hold potential for successful predictive analysis, if utilised correctly. Thirdly, the study has shown that machine learning is a fast and an inexpensive tool that can be used for the maintenance and monitoring of water treatment plants. As shown in literature, machine learning can reduce the cost of running a treatment plant and highlight the most vulnerable aspects, including the most important chemicals that contribute to the utility's WQI.

6.2 Recommendations

- Future work should include exploring more machine learning algorithms. Examples of algorithms that will be suited for this study include convolutional neural networks (CNN) LSTM, bidirectional LTSM, and XGBoost. These have potential to extract effective features from the data, improve algorithm performance on sequence classifications on the input data, as well as achieve faster performances, to name a few.

- Additional water quality data should be collected (pH, electrical conductivity, nitrate concentration etc). This should slightly increase the volume or/and dimensionality of the complete dataset, as these qualities can help machine learning algorithms perform better.

- The degrees of freedom of the grid search function should be expanded to find the maximum optimal combination of hyperparameters for both the RF and SVR, as well as any additional algorithms. This was not accomplished in the study due to time constraints and computing power.

References

Abad, C., & Iyengar, G. (2015). Constructing an effiecient machine learning model for decision making in economics, Editors of the Series WP7 " Mathematical methods for decision making in economics, business and politics " Aleskerov Fuad, Mirkin Boris, Podinovskiy Vladislav. *SIAM Journal on Financial Mathematics*, *6*(1), 467–486. http://epubs.siam.org/doi/10.1137/140967635

Abate, A. F., Barra, P., Pero, C., & Tucci, M. (2020). Head pose estimation by regression algorithm. *Pattern Recognition Letters*, *140*, 179–185. https://doi.org/10.1016/j.patrec.2020.10.003

Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-Based Learning Algorithms. *Mach. Learn*, *6*(1), 37–66.

Ahmed, U., Mumtaz, R., Anwar, H., Shah, A. A., Irfan, R & Garcia-Nieto, J. (2019). Efficient Water Quality Prediction Using Supervised Machine Learning. *MDPI*, 1–14.

Akossou, A. (2013). Impact of data structure on the estimators R-square and adjusted R-square in linear regression. *International Journal of Mathematics and Computation*, *January*. http://www.ceser.in/ceserp/index.php/ijmc/article/view/2579

Al-Adhaileh, M. H., & Alsaade, F. W. (2021). Modelling and prediction of water quality by using artificial intelligence. *Sustainability (Switzerland)*, *13*(8), 1–18. https://doi.org/10.3390/su13084259

Aljuaid, T., & Sasi, S. (2017). Proper imputation techniques for missing values in data sets. *Proceedings of the 2016 International Conference on Data Science and Engineering, ICDSE 2016*, *August*. https://doi.org/10.1109/ICDSE.2016.7823957

Antwi-Boasiako, E., Zhou, S., Liao, Y., Liu, Q., Wang, Y., & Owusu-Agyemang, K. (2021). Privacy preservation in Distributed Deep Learning: A survey on Distributed Deep Learning, privacy preservation techniques used and interesting research directions. *Journal of Information Security and Applications*, *61*, 102949. https://doi.org/10.1016/J.JISA.2021.102949

Afentoulis, V., & Lioufi, K. (2015). SVM Classification with Linear and RBF kernels. Available online: https://www.researchgate.net/publication/279913074_SVM_Classification_with_Linear_and_RBF_kernels (accessed 4 July 2021)

Arias del Campo, F., Guevara Neri, M. C., Vergara Villegas, O. O., Cruz Sánchez, V. G., Ochoa Domínguez, H. de J., & García Jiménez, V. (2021). Auto-adaptive multilayer perceptron for univariate time series classification. *Expert Systems with Applications*, *181*(April), 115147. https://doi.org/10.1016/j.eswa.2021.115147

ArunKumar, K. E., Kalaga, D. V., Kumar, C. M. S., Kawaji, M., & Brenza, T. M. (2021). Forecasting of COVID-19 using deep layer Recurrent Neural Networks (RNNs) with Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTM) cells. *Chaos, Solitons & Fractals*, *146*, 110861. https://doi.org/10.1016/J.CHAOS.2021.110861

Asadi-Pooya, A. A., Bazrafshan, M., & Farazdaghi, M. (2021). Cluster analysis of a large dataset of patients with Lennox-Gastaut syndrome. *Seizure*, *92*, 36–39. https://doi.org/10.1016/J.SEIZURE.2021.08.005

Awad, M., & Khanna, R. (2015). Efficient learning machines: Theories, concepts, and applications for engineers and system designers. *Appress*, Berkeley - CA

Aziz, N. B. A., & Abdullah, W. F. H. (2015). Comparison between MLP and RBF network in improving CHEMFET sensor selectivity. *ISCAIE 2015 - 2015 IEEE Symposium on Computer Applications and Industrial Electronics*, 165–170. https://doi.org/10.1109/ISCAIE.2015.7298349

Basson, L., Bronkhorst, S., Fordyce, N., & Pengelley, C. (2018). Water-2018 Market Intelligence Report-18 Roeland Street, Cape Town, 8001, South Africa Editorial and review. *Technical Report*.

Belikova, V., Panchuk, V., Legin, E., Melenteva, A., Kirsanov, D., & Legin, A. (2019). Continuous monitoring of water quality at aeration plant with potentiometric sensor array. *Sensors and Actuators B: Chemical*, *282*, 854–860. https://doi.org/10.1016/J.SNB.2018.11.153

Bernardelli, A., Marsili-Libelli, S., Manzini, A., Stancari, S., Tardini, G., Montanari, D., Anceschi, G., Gelli, P., & Venier, S. (2020). Real-time model predictive control of a wastewater treatment plant based on machine learning. *Water Science and Technology*, *81*(11), 2391–2400. https://doi.org/10.2166/wst.2020.298

Chang, F. J., Tsai, Y. H., Chen, P. A., Coynel, A., & Vachaud, G. (2015). Modeling water quality in an urban river using hydrological factors - Data driven approaches. *Journal of Environmental Management*, *151*, 87–96. https://doi.org/10.1016/j.jenvman.2014.12.014

Chen, G., Long, T., Xiong, J., & Bai, Y. (2017). Multiple Random Forests Modelling for Urban Water Consumption Forecasting. *Water Resources Management*, *31*(15), 4715–4729. https://doi.org/10.1007/s11269-017-1774-7

Chen, Z., Xu, H., Jiang, P., Yu, S., Lin, G., Bychkov, I., Hmelnov, A., Ruzhnikov, G., Zhu, N., & Liu, Z. (2021). A transfer Learning-Based LSTM strategy for imputing Large-Scale consecutive missing data and its application in a water quality prediction system. *Journal of Hydrology*, *602*, 126573. https://doi.org/10.1016/J.JHYDROL.2021.126573

Cheng, R., Chen, D., Gai, W., & Zheng, S. (2019). Intelligent driving methods based on sparse LSSVM and ensemble CART algorithms for high-speed trains. *Computers & Industrial Engineering*, *127*, 1203–1213. https://doi.org/10.1016/J.CIE.2018.03.022

Chiroma, H., Abdullahi, U. A., Abdulhamid, S. M., Abdulsalam Alarood, A., Gabralla, L. A., Rana, N., Shuib, L., Targio Hashem, I. A., Gbenga, D. E., Abubakar, A. I., Zeki, A. M., & Herawan, T. (2019). Progress on Artificial Neural Networks for Big Data Analytics: A Survey. *IEEE Access*, *7*(November), 70535–70551. https://doi.org/10.1109/ACCESS.2018.2880694

Chuang, C. C., Su, S. F., Jeng, J. T., & Hsiao, C. C. (2002). Robust support vector regression networks for function approximation with outliers. *IEEE Transactions on Neural Networks*, *13*(6), 1322–1330. https://doi.org/10.1109/TNN.2002.804227

Cookl, S. (2001). Finite-sample critical values of the Augmented Dickey-Fuller statistic: a note on lag order. *Economic Issues*, *6*(2), 31–46.

Csábrági, A., Molnár, S., Tanos, P., & Kovács, J. (2015). Forecasting of dissolved oxygen in the

river danube using neural networks. *Hungarian Agricultural Engineering*, *7410*(27), 38–41. https://doi.org/10.17676/hae.2015.27.38

Culurciello, E. (2018). *The fall of RNN / LSTM*. Towards Data Science. https://towardsdatascience.com/the-fall-of-rnn-lstm-2d1594c74ce0 (accessed 19 October 2021)

Diaz-Uriarte, R., Alvarez de Andres, S. (2006). Gene Selection and Classification of Microarray Data Using Random Forest. *BMC Bioinformatics*, 7 (1) 3.

IBM Cloud Education. (2020). *Unsupervised Learning*. IBM. https://www.ibm.com/cloud/learn/unsupervised-learning (accessed 25 September 2021)

Elg, E. (2018). *Jump Start Your Modeling with Random Forests*. Elder Research. https://www.elderresearch.com/blog/jump-start-your-modeling-with-random-forests/#:~:text=Also%2C output outliers will affect,of any other leaf node.&text=So output outliers have a,prediction of a Random Forest. (accessed 21 September 2021)

Engelbrecht, A. (2007). Computational Intelligence: An Introduction (2nd ed.). *Wiley*. http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470035617.html

Fan, J., Li, M., Guo, F., Yan, Z., Zheng, X., Zhang, Y., Xu, Z., & Wu, F. (2018). Priorization of river restoration by coupling soil and water assessment tool (SWAT) and support vector machine (SVM) models in the Taizi river basin, northern China. *International Journal of Environmental Research and Public Health*, *15*(10). https://doi.org/10.3390/ijerph15102090

Farhi, N., Kohen, E., Mamane, H., & Shavitt, Y. (2021). Prediction of wastewater treatment quality using LSTM neural network. *Environmental Technology & Innovation*, *23*, 101632. https://doi.org/10.1016/j.eti.2021.101632

Figueiredo Filho, D. B., Silva Júnior, J. A., & Rocha, E. C. (2011). What is R2 all about? *Leviathan (São Paulo)*, *3*, 60. https://doi.org/10.11606/issn.2237-4485.lev.2011.132282

Forough Kamyab-Talesh, Mousavi, S. F., Khaledian, M., Yousefi-Falakdehi, O., & Norouzi-Masir, M. (2019). Prediction of Water Quality Index by Support Vector Machine: a Case Study in the Sefidrud Basin, Northern Iran. *Water Resources*, *46*(1), 112–116. https://doi.org/10.1134/S0097807819010056

Gaffoor, Z., Pietersen, K., Jovanovic, N., Bagula, A., & Kanyerere, T. (2020). Big data analytics and its role to support groundwater management in the southern african development community. *Water (Switzerland)*, *12*(10). https://doi.org/10.3390/w12102796

Gagniuc, P.A. (2017). From Theory to Implementation and Experimentation. *John Wiley and Sons*, 22(10), 12 - 16

Gamble, A., & Babbar-Sebens, M. (2012). On the use of multivariate statistical methods for combining in-stream monitoring data and spatial analysis to characterize water quality conditions in the White River Basin, Indiana, USA. *Environmental Monitoring and Assessment*, *184*(2), 845–875. https://doi.org/10.1007/s10661-011-2005-y

Gareth, J., Daniela, W., Trevor, H., & Robert., T. (2013). An Introduction to Statistical Learning: with Applications in R (2nd ed.). *Springer*. https://doi.org/10.1007/978-1-4614-7138-7

Gebler, D., Wiegleb, G., & Szoszkiewicz, K. (2018). Integrating river hydromorphology and water quality into ecological status modelling by artificial neural networks. *Water Research*, *139*, 395–405. https://doi.org/10.1016/j.watres.2018.04.016

Gholamy, A., Kreinovich, V., & Kosheleva, O. (2018). Why 70/30 or 80/20 Relation Between Training and Testing Sets : A Pedagogical Explanation. *Departmental Technical Reports (CS)*, 1–6.

Gidlow, D. A. (2004). Lead toxicity. *Occupational Medicine*, *54*(2), 76–81. https://doi.org/10.1093/occmed/kqh019

Goudarzi, G., Hopke, P. K., & Yazdani, M. (2021). Forecasting PM2.5 concentration using artificial neural network and its health effects in Ahvaz, Iran. *Chemosphere*, *283*, 131285. https://doi.org/10.1016/J.CHEMOSPHERE.2021.131285

Guan, X. M. (2020). Water Treatment Analysis Based on Support Vector Machine Regression Algorithm. *Journal of Physics: Conference Series*, *1533*(2). https://doi.org/10.1088/1742-6596/1533/2/022054

Guo, C., Liu, M., & Lu, M. (2021). A Dynamic Ensemble Learning Algorithm based on K-means for ICU mortality prediction. *Applied Soft Computing*, *103*, 107166. https://doi.org/10.1016/J.ASOC.2021.107166

Habiyaremye, A. (2020). Water innovation in South Africa: Mapping innovation successes and diffusion constraints. *Environmental Science and Policy*, *114*(October 2019), 217–229. https://doi.org/10.1016/j.envsci.2020.08.011

Haque, N., & Riaz, M. H. (2017). Autonomous Vehicle Control System as a Mobile Robot by Artificial Neural Network. *IAES International Journal of Robotics and Automation (IJRA)*, *6*(3), 200. https://doi.org/10.11591/ijra.v6i3.pp200-206

Heddam, S. (2016). Multilayer perceptron neural network-based approach for modeling phycocyanin pigment concentrations: case study from lower Charles River buoy, USA. *Environmental Science and Pollution Research*, *23*(17), 17210–17225. https://doi.org/10.1007/s11356-016-6905-9

Henseler, J., Ringle, C., & Sinkovics, R. (2009). The use of Partial Least Squares Path Modeling in International Marketing. *Advances in International Marketing (AIM)*, *20*, 277–320.

Hill, D., Kerkez, B., Rasekh, A., Ostfeld, A., Minsker, B., & Banks, M. . (2014). Sensing and cyberinfrastructure for smarter water management: The promise and challenge of ubiquity. *Journal of Water Resources Planning and Management*, *140*(7). https://doi.org/10.1061/(ASCE)WR.1943-5452.0000449

Hochreiter, S & Schimidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *19*, 1735–1780

Hodgkin, A., & Huxley, A. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*.

Huang, Y., Shang, C., & Li, L. (2021). Novel N-doped graphene enhanced ultrafiltration nano-porous polyvinylidene fluoride membrane with high permeability and stability for water treatment. *Separation and Purification Technology*, *267*, 118622. https://doi.org/10.1016/J.SEPPUR.2021.118622

Iwana, B. K., & Uchida, S. (2021). An empirical survey of data augmentation for time series classification with neural networks. *PLoS ONE*, *16*(7 July), 1–3. https://doi.org/10.1371/journal.pone.0254841

Jiang, Y., Malliaras, P., Chen, B., & Kulić, D. (2021). Model-based data augmentation for user-

independent fatigue estimation. *Computers in Biology and Medicine*, *137*, 104839. https://doi.org/10.1016/J.COMPBIOMED.2021.104839

Kanamori, T., Fujiwara, S., & Takeda, A. (2017). Breakdown point of robust support vector machines. *Entropy*, *19*(2). https://doi.org/10.3390/e19020083

Kanyama, Y. (2021). Application of machine learning techniques in predicting groundwater levels and discharge rates in the North West aquifers. *University of the Witwatersrand*, 1 - 72

Kar, S., Rathore, V. S., Champati ray, P. K., Sharma, R., & Swain, S. K. (2016). Classification of river water pollution using Hyperion data. *Journal of Hydrology*, *537*, 221–233. https://doi.org/10.1016/j.jhydrol.2016.03.047

Karamizadeh, S., Abdullah, S. M., Halimi, M., Shayan, J., & Rajabi, M. J. (2014). Advantage and drawback of support vector machine functionality. *I4CT 2014 - 1st International Conference on Computer, Communications, and Control Technology, Proceedings*, *I4ct*, 63–65. https://doi.org/10.1109/I4CT.2014.6914146

Kat, C. J., & Els, P. S. (2012). Validation metric based on relative error. *Mathematical and Computer Modelling of Dynamical Systems*, *18*(5), 487–520. https://doi.org/10.1080/13873954.2012.663392

Khodabakhsh, A., Ari, I., Bakır, M., & Alagoz, S. M. (2020). Forecasting Multivariate Time-Series Data Using LSTM and Mini-Batches. *Lecture Notes on Data Engineering and Communications Technologies*, *45*(January), 121–129. https://doi.org/10.1007/978-3-030-37309-2_10

Kim, J., Wang, W., Lee, S., Park, J. H., & Oh, J. E. (2021). Concentrations and distributions of neonicotinoids in drinking water treatment plants in South Korea. *Environmental Pollution*, *288*, 117767. https://doi.org/10.1016/J.ENVPOL.2021.117767

Kirthana, T., & Soumya, A. (2020). Machine Learning Techniques for wastwewater treatment plants. *Journal of University of Shanghai for Science and Technology*, *22*(10), 12–16.

Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V., & Fotiadis, D. I. (2015). Machine learning applications in cancer prognosis and prediction. *Computational and Structural Biotechnology Journal*, *13*, 8–17. https://doi.org/10.1016/j.csbj.2014.11.005

Kranthi Kumar, K., Dileep Kumar, M., Samsonu, C., & Vamshi Krishna, K. (2021). Role of convolutional neural networks for any real time image classification, recognition and analysis. *Materials Today: Proceedings*. https://doi.org/10.1016/J.MATPR.2021.02.186

Kutyłowska, M. (2017). Application of regression tress for prediction of water conduits failure rate. *E3S Web of Conferences*, *22*. https://doi.org/10.1051/e3sconf/20172200097

Ławrynowicz, A., & Tresp, V. (2014). Introducing machine learning. *Perspectives on Ontology Learning*, *18*(November), 35–50. https://doi.org/10.1007/978-3-030-67626-1_8

Li, L. N., Liu, X. F., Yang, F., Xu, W. M., Wang, J. Y., & Shu, R. (2021). A review of artificial neural network based chemometrics applied in laser-induced breakdown spectroscopy analysis. *Spectrochimica Acta Part B: Atomic Spectroscopy*, *180*, 106183. https://doi.org/10.1016/J.SAB.2021.106183

Liakos, K. G., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). Machine learning in agriculture: A review. *Sensors (Switzerland)*, *18*(8), 1–29. https://doi.org/10.3390/s18082674

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). *A Critical Review of Recurrent Neural Networks for Sequence Learning*. 1–38. http://arxiv.org/abs/1506.00019

Luo, T., Young, R., & Reig, P. (2015). Aqueduct Projected Water Stress Country Rankings. *World Resources Insititute*.

Makariou, D., Barrieu, P., & Chen, Y. (2021). A random forest based approach for predicting spreads in the primary catastrophe bond market. *Insurance: Mathematics and Economics*. https://doi.org/10.1016/J.INSMATHECO.2021.07.003

Marras, S., & Mandli, K. T. (2021). Modeling and simulation of tsunami impact: A short review of recent advances and future challenges. *Geosciences (Switzerland)*, *11*(1), 1–19. https://doi.org/10.3390/geosciences11010005

Masindi, V. (2020). Dataset on physicochemical and microbial properties of raw water in four drinking water treatment plants based in South Africa. *Data in Brief*, *31*, 105822. https://doi.org/10.1016/J.DIB.2020.105822

Mijwil, M. M. (2018). Artificial Neural Networks Advantages and Disadvantages. Accessed

online: https://www.researchgate.net/publication/323665827

Mirauda, D., Capece, N., & Erra, U. (2020). Sustainable water management: Virtual reality training for open-channel flow monitoring. *Sustainability (Switzerland)*, *12*(3). https://doi.org/10.3390/su12030757

Modaresi, F., & Araghinejad, S. (2014). A comparative assessment of support vector machines, probabilistic neural networks, and K-nearest neighbor algorithms for water quality classification. *Water Resources Management*, *28*(12), 4095–4111. https://doi.org/10.1007/s11269-014-0730-z

Moguerza, J. M., & Muñoz, A. (2006). Support vector machines with applications. *Statistical Science*, *21*(3), 322–336. https://doi.org/10.1214/088342306000000493

Mohammed, H., Hameed, I. A., & Seidu, R. (2017). Random forest tree for predicting fecal indicator organisms in drinking water supply. *Proceedings of 4th International Conference on Behavioral, Economic, and Socio-Cultural Computing, BESC 2017*, *2018-Janua*, 1–6. https://doi.org/10.1109/BESC.2017.8256398

Moore, D. ., Notz, W, I., & Finger, M. . (2013). The Basic Practice of Statistics (6th ed.). *W.H. Freeman and Company*, 138

Na, S. H., Kim, M. J., Kim, J. T., Jeong, S., Lee, S., Chung, J., & Kim, E. J. (2021). Microplastic removal in conventional drinking water treatment processes: Performance, mechanism, and potential risk. *Water Research*, *202*, 117417. https://doi.org/10.1016/J.WATRES.2021.117417

Newhart, K. B., Holloway, R. W., Hering, A. S., & Cath, T. Y. (2019). Data-driven performance analyses of wastewater treatment plants: A review. *Water Research*, *157*, 498–513. https://doi.org/10.1016/j.watres.2019.03.030

Nourani, V., & Behfar, N. (2021). Multi-station runoff-sediment modeling using seasonal LSTM models. *Journal of Hydrology*, *601*, 126672. https://doi.org/10.1016/J.JHYDROL.2021.126672

Oduah, U. I., Kevin, I. F., Oluwole, D. O., & Izunobi, J. U. (2021). Towards a high-precision contactless fingerprint scanner for biometric authentication. *Array*, *11*, 100083.

https://doi.org/10.1016/J.ARRAY.2021.100083

Olyaie, E., Zare Abyaneh, H., & Danandeh Mehr, A. (2017). A comparative analysis among computational intelligence techniques for dissolved oxygen prediction in Delaware River. *Geoscience Frontiers*, *8*(3), 517–527. https://doi.org/10.1016/j.gsf.2016.04.007

Oostwal, E., Straat, M., & Biehl, M. (2021). Hidden unit specialization in layered neural networks: ReLU vs. sigmoidal activation. *Physica A: Statistical Mechanics and Its Applications*, *564*, 125517. https://doi.org/10.1016/J.PHYSA.2020.125517

Patro, S. G. K., & Sahu, K. K. (2015). Normalization: A Preprocessing Stage. *Iarjset*, 20–22. https://doi.org/10.17148/iarjset.2015.2305

Pisa, I., Santín, I., López Vicario, J., Morell, A., & Vilanova, R. (2018). A recurrent neural network for wastewater treatment plant effuents' prediction. in *Proc.Actas 39th Jornadas Autom,*. Badajoz, Spain *September*, 621–628. https://doi.org/10.17979/spudc.9788497497565.0621

Preeti, P., Bala, R., & Singh, R. (2019). Financial and Non-Stationary Time Series Forecasting using LSTM Recurrent Neural Network for Short and Long Horizon. *2019 10th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2019*. https://doi.org/10.1109/ICCCNT45670.2019.8944624

Prajwala, T. R. (2015). A Comparative Study on Decision Tree and Random Forest Using R Tool. *Ijarcce*, *4*(1), 196–199. https://doi.org/10.17148/ijarcce.2015.4142

Raghavendra, S., & Deka, P. C. (2014). Support vector machine applications in the field of hydrology: A review. *Applied Soft Computing Journal*, *19*, 372–386. https://doi.org/10.1016/j.asoc.2014.02.002

Rakhra, M., Soniya, P., Tanwar, D., Singh, P., Bordoloi, D., Agarwal, P., Takkar, S., Jairath, K., & Verma, N. (2021). Crop Price Prediction Using Random Forest and Decision Tree Regression:-A Review. *Materials Today: Proceedings*. https://doi.org/10.1016/J.MATPR.2021.03.261

Ramli, N. A., Fairuz, M., & Hamid, A. (2018). Data based modeling of a wastewater treatment plant by using machine learning methods. *Journal of Engineering Technology*, *6*(1), 14–21.

Rao, R., & Scherer, R. (2010). Statistical Pattern Recognition and Machine Learning in Brain–Computer Interfaces. *Statistical Signal Processing for Neuroscience and Neurotechnology*, 335–367. https://doi.org/10.1016/B978-0-12-375027-3.00010-7

Rodriguez, C. (1980). On detection of outliers and their effect in supervised classification. *Department of Mathematics, November 2004*, 1–14.

Rosin, P. L., & Fierens, F. (1998). The effects of data filtering on neural network learning. *Neurocomputing*, *20*(1–3), 155–162. https://doi.org/10.1016/S0925-2312(98)00008-3

Salami, E. S., & Ehteshami, M. (2015). Simulation, evaluation and prediction modeling of river water quality properties (case study: Ireland Rivers). *International Journal of Environmental Science and Technology*, *12*(10), 3235–3242. https://doi.org/10.1007/s13762-015-0800-7

Sameen, M. I., Pradhan, B., & Lee, S. (2019). Self-Learning Random Forests Model for Mapping Groundwater Yield in Data-Scarce Areas. *Natural Resources Research*, *28*(3), 757–775. https://doi.org/10.1007/s11053-018-9416-1

Satapathy, S. C., Govardhan, A., Raju, K. S., & Mandal, J. K. (2015). Emerging ICT for Bridging the Future - Proceedings of the 49th Annual Convention of the Computer Society of India (CSI) Volume 2. *Advances in Intelligent Systems and Computing*, *338*(July), I–IV. https://doi.org/10.1007/978-3-319-13731-5

Saud, J. L., & Abass, K. Z. (2018). A Comparison between Multi-Layer Perceptron and Radial Basis Function Networks in Detecting Humans Based on Object Shape. *Ibn AL- Haitham Journal For Pure and Applied Science*, *31*(2), 210. https://doi.org/10.30526/31.2.1950

Sethi, A. (2020). *Support Vector Regression Tutorial for Machine Learning*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/ (accessed 30 September 2021)

Shafiee, M. E., Barker, Z., & Rasekh, A. (2018). Enhancing water system models by integrating big data. *Sustainable Cities and Society*, *37*, 485–491. https://doi.org/10.1016/J.SCS.2017.11.042

Sharma, H., Rawal, N., & Mathew, B. B. (2015). The Characteristics , Toxicity and Effects of Cadmium. *International Journal of Nanotechnology and Nanoscience*, *3*(January), 1–9.

Shaukat Dar, K., & Zaheer, S. (2015). Association Rule Mining: An Application Perspective. *International Journal of Computer Science and Innovation*, *1*.

Sikka, M. (2020). *Balancing the Regularization Effect of Data Augmentation*. Towards Data Science. https://towardsdatascience.com/balancing-the-regularization-effect-of-data-augmentation-eb551be48374 (accessed 3 November 2020)

Singh, J., & Dhiman, G. (2021). A survey on machine-learning approaches : Theory and their concepts. *Materials Today: Proceedings*, https://doi.org/10.1016/j.matpr.2021.05.335

Hopp, D. (2021). Economic Nowcasting with Long Short-Term Memory Artificial Neural Networks (LSTM *)*. *UNCTAD*, *62*.

Su, D., Kong, H., Qiao, Y., & Sukkarieh, S. (2021). Data augmentation for deep learning based semantic segmentation and crop-weed classification in agricultural robotics. *Computers and Electronics in Agriculture*, *190*, 106418. https://doi.org/10.1016/J.COMPAG.2021.106418

Sun, B., Zhou, Y., Wang, J., & Zhang, W. (2021). A new PC-PSO algorithm for Bayesian network structure learning with structure priors. *Expert Systems with Applications*, *184*, 115237. https://doi.org/10.1016/J.ESWA.2021.115237

Kumar, T., Steinbach., Karpatne. (2021). Model Overfitting. *Introduction to Data Mining* (2nd ed), 15. https://www-users.cs.umn.edu/~kumar001/dmbook/slides/chap3_overfitting.pdf

Thompson, K., & Kadiyala, R. (2014). Leveraging big data to improve water system operations. *Procedia Engineering*, *89*, 467–472. https://doi.org/10.1016/j.proeng.2014.11.235

Tuan Hoang, A., Nižetić, S., Chyuan Ong, H., Tarelko, W., Viet Pham, V., Hieu Le, T., Quang Chau, M., & Phuong Nguyen, X. (2021). A review on application of artificial neural network (ANN) for performance and emission characteristics of diesel engine fueled with biodiesel-based fuels. *Sustainable Energy Technologies and Assessments*, *47*, 101416. https://doi.org/10.1016/J.SETA.2021.101416

Turton, A. R., & Meissner, R. (2015). Towards a Set of Guidelines for Best Practices : An Assessment of Forced Removals for the Lesotho Highlands Water Project in Light of the World Commission on Dams. *ICAES*. 1–25.

Van Houdt, G., Mosquera, C., & Nápoles, G. (2020). A review on the long short-term memory model. *Artificial Intelligence Review*, *53*(8), 5929–5955. https://doi.org/10.1007/s10462-020-09838-1

Wang, H., Zhou, J., Wang, Y., Wei, J., Liu, W., Yu, C., & Li, Z. (2019). Optimization algorithms of neural networks for traditional time-domain equalizer in optical communications. *Applied Sciences (Switzerland)*, *9*(18). https://doi.org/10.3390/app9183907

Wang, Q., & Yang, Z. (2016). Industrial water pollution, water environment treatment, and health risks in China. *Environmental Pollution*, *218*(July), 358–365. https://doi.org/10.1016/j.envpol.2016.07.011

Wu, Q., & Zhou, D. X. (2005). SVM Soft Margin Classifiers: Linear Programming versus Quadratic Programming. *Neural Computation*, 17(5), 1160 - 1187

Yadav, A., Jha, C. K., & Sharan, A. (2020). Optimizing LSTM for time series prediction in Indian stock market. *Procedia Computer Science*, *167*(2019), 2091–2100. https://doi.org/10.1016/j.procs.2020.03.257

Yahya, A. S. A., Ahmed, A. N., Othman, F. B., Ibrahim, R. K., Afan, H. A., El-Shafie, A., Fai, C. M., Hossain, M. S., Ehteram, M., & Elshafie, A. (2019). Water quality prediction model based support vector machine model for ungauged river catchment under dual scenarios. *Water (Switzerland)*, *11*(6). https://doi.org/10.3390/w11061231

Yarnold, P. R., Bryant, F. B., & Soltysik, R. C. (2013). Optimal Data Analysis Maximizing the Accuracy of Multiple Regression Models using UniODA: Regression Away From the Mean. *Llc*, *2*(1), 19–25.

Yoon, N., Kim, J., Lim, J. L., Abbas, A., Jeong, K., & Cho, K. H. (2021). Dual-stage attention-based LSTM for simulating performance of brackish water treatment plant. *Desalination*, *512*(February), 115107. https://doi.org/10.1016/j.desal.2021.115107

You, W., Simalatsar, A., & De Micheli, G. (2013). Parameterized SVM for personalized drug concentration prediction. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 5789–5792. https://doi.org/10.1109/EMBC.2013.6610867

Zaremba, L. S., & Smoleński, W. H. (2000). Optimal portfolio choice under a liability constraint. *Annals of Operations Research*, *97*(1–4), 131–141. https://doi.org/10.1023/A