DEVELOPMENT OF CHLOROPHYLL-A SOFT SENSOR USING MACHINE LEARNING AND IOT

Thesis · August 2022

DOI: 10.31219/osf.io/v9cy2

2 authors:

Palok Biswas
Delft University of Technology
2 PUBLICATIONS **1** CITATION
SEE PROFILE

Zati Hakim Azizul Hasan
University of Malaya
30 PUBLICATIONS **158** CITATIONS
SEE PROFILE

DEVELOPMENT OF CHLOROPHYLL-A SOFT SENSOR USING MACHINE LEARNING AND IOT

PALOK BISWAS

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITI MALAYA
KUALA LUMPUR

2022

DEVELOPMENT OF CHLOROPHYLL-A SOFT SENSOR USING MACHINE LEARNING AND IOT

PALOK BISWAS

DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE (APPLIED COMPUTING)

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITI MALAYA
KUALA LUMPUR

2022

UNIVERSITI MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: PALOK BISWAS [redacted]

Matric No: WOA180016 / 17198713/1

Name of Degree: Master of Computer Science (Applied Computing)

Title of Dissertation: Development of Chlorophyll-a Soft Sensor Using Machine Learning and IoT

Field of Study: Machine Learning

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This Work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the Universiti Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature [redacted] Date: 06/08/2022

Subscribed and solemnly declared before,

Witness's Signature Date: 7.8.2022

Name: DR ZATI HAKIM AZIZUL HASAN

Designation: SENIOR LECTURER
DEPT OF ARTIFICIAL INTELLIGENCE
FACULTY OF COMPUTER SCIENCE &
INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
50603 KUALA LUMPUR

DEVELOPMENT OF CHLOROPHYLL-A SOFT SENSOR USING MACHINE LEARNING AND IOT

ABSTRACT

Eutrophication is a precursor to harmful algal blooms and has become a global concern for its impacts on freshwater supply and biodiversity. About 60% of 90 Malaysian lakes are reportedly eutrophic, encouraged by the warm tropical climate and nutrient inflows during heavy rain. Measuring Chlorophyll-a (Chl-a) concentration level using in-situ sondes has helped the water community monitor lakes' health. However, the expensive Chl-a sonde is not always affordable, hindering the continuous real-time data collection required for algae profiling. Interestingly, temperature, dissolved oxygen, electrical conductivity, turbidity, pH, and other water quality (WQ) parameters can estimate Chl-a concentration level. Soft sensor development is gaining traction through machine learning (ML) in deriving Chl-a concentration levels from WQ parameters. ML models can map the complex nonlinear relationship between WQ parameters and Chl-a. Popular ML models include artificial neural network (ANN), Random Forest (RF), Support Vector Regression (SVR), and Recurrent Neural Networks (RNN). Researchers have successfully used seven or more WQ parameters to input their Chl-a ML model. However, their prediction is usually based on historical WQ data, using past years to predict the current algae situation. Lastly, none of the researchers considered the LightGBM model, a new and improved classification and regression tree (CART) based decision tree model. This study explores the development of a Chl-a soft sensor that can estimate Chl-a concentration level using four WQ parameters, namely, temperature, dissolved oxygen, electrical conductivity, and turbidity. Since WQ parameters have nonlinear relationships with Chl-a, CART models are considered. CART is also preferred because of its explainability, which aids decision-making. This study also proposes an IoT-Cloud

application development with online ML inferencing towards continuous and real-time monitoring of Chl-a concentration levels. The WQ data were collected for experimental purposes between February and October 2019 at Tasik Aman in Petaling Jaya. Aside from the four WQ parameters, the Chl-a sonde is also used for training labels and ground truth. The data was sent to the cloud application from the dataset file using the MQTT protocol. The cloud's deployment service containerizes the ML model and hosts it as a web service endpoint. The endpoint can be called for real-time inferencing. In terms of results, the LightGBM is the best performing model achieving a coefficient of determination, R^2, of 98.9%. Random forest is second-best at 97.9% R^2. Interestingly, ANN is the worst performing at 80% R^2. The LightGBM prediction plot matches all trends and spikes compared to ground truth, whereas the ANN prediction plot has a lot of anomalous fluctuations in the prediction.

Keywords: Eutrophication, water quality parameters, soft sensor, machine learning, internet of things

PEMBANGUNAN SENSOR LEMBUT CHLOROPHYLL-A DENGAN PEMBELAJARAN MESIN AND IOT

ABSTRAK

Eutrofikasi adalah penyebab kepada pertumbuhan alga yang berbahaya dan telah menjadi kebimbangan global kerana impaknya kepada bekalan air tawar dan biodiversiti. Kira-kira 60% daripada 90 tasik Malaysia dilaporkan eutropik, kesan iklim tropika dan aliran masuk nutrien semasa hujan lebat. Mengukur tahap kepekatan Klorofil-a (Chl-a) menggunakan sensor secara in-situ telah membantu komuniti air memantau kesihatan tasik. Walaubagaimanapun, sensor Chl-a yang mahal tidak selalu mampu milik, Ini menghalang pengumpulan data masa nyata berterusan yang diperlukan untuk pemprofilan alga. Menariknya, data suhu, oksigen terlarut, kekonduksian elektrik, kekeruhan, pH dan parameter-parameter kualiti air (WQ) lain boleh menganggarkan tahap kepekatan Chl-a. Pembangunan sensor lembut melalui pemodelan induktif pembelajaran mesin (ML) dalam memperoleh tahap kepekatan Chl-a daripada parameter WQ kini mendapat daya tarikan. Pemodelan induktif boleh memetakan hubungan kompleks tak linear antara parameter-parameter WQ dan Chl-a. Model ML yang popular termasuk rangkaian saraf tiruan (ANN), hutan rawak, regresi vektor sokongan dan rangkaian saraf berulang. Kebanyakan penyelidik telah berjaya dengan tujuh atau lebih parameter-parameter WQ sebagai input kepada model ML Chl-a mereka. Ramalan mereka biasanya berdasarkan data lama WQ, bermakna mereka menggunakan data pada tahun-tahun lepas untuk meramalkan situasi semasa alga. Akhir sekali, tiada penyelidik mempertimbangkan model LightGBM, suatu model ML berasaskan pokok keputusan yang telah ditambahbaik. Kajian ini meneroka pembangunan sensor lembut Chl-a yang boleh menganggar tahap kepekatan Chl-a menggunakan empat parameter WQ iaitu suhu, oksigen terlarut, kekonduksian elektrik dan kekeruhan. Memandangkan parameter WQ mempunyai hubungan tak linear dengan Chl-a, klasifikasi dan pepohon regresi atau

model CART telah dipertimbangkan. Ia juga diutamakan kerana kebolehjelasannya yang membantu membuat keputusan. Kajian ini juga mencadangkan pembangunan aplikasi IoT-Cloud dengan inferens ML dalam usaha ke arah pemantauan tahap kepekatan Chl-a yang berterusan dalam masa nyata. Untuk tujuan percubaan, data WQ telah dikumpul antara Februari hingga Oktober 2019 di Tasik Aman, Petaling Jaya. Selain daripada empat parameter WQ, sensor Chl-a juga digunakan untuk mengukur *ground truth*. Data telah dihantar ke aplikasi awan daripada fail set data melalui protokol MQTT. Servis awan mempakejkan model ML yang dihoskan sebagai titik akhir perkhidmatan web. Titik akhir ini boleh dipanggil untuk inferens masa nyata. Dari segi keputusan, model LightGBM menunjukkan prestasi terbaik dengan pekali penentuan, R^2, sebanyak 98.9%. Model hutan rawak adalah kedua terbaik dengan 97.9% R^2. Menariknya, ANN mencapai prestasi terburuk dengan 80% R^2. Perbandingan dengan *ground truth* menunjukkan corak LightGBM berpadanan dengan semua pancang dan lonjakan manakala ANN menunjukkan banyak ralat dalam corak ramalan.

Kata kunci: Eutrofikasi, parameter-parameter kualiti air, sensor lembut, pembelajaran mesin, internet pelbagai benda

ACKNOWLEDGEMENTS

It was through pure serendipity that I stumbled into this project and got to work on problems I am passionate about—applying technology to help accelerate the achievement of the SDGs. It was only possible because of Dr Zati Hakim Azizul Hasan, who introduced me to this exciting problem in the water quality domain and helped me navigate its challenges. I am truly grateful to her for her unwavering support, countless encouragements, and meticulous review of my work throughout this journey.

I am indebted to Johnathan Daniel Maxey for introducing me to the professional approaches to water quality monitoring and for sharing his wealth of experience in Malaysian lakewater sampling. I am thankful to his company, ADS Environmental Services Sdn. Bhd. for permitting me to use the Tasik Aman lake dataset for scientific research. I would also like to thank MBPJ and the head of MBPJ's Environmental Department, Mr. Mohd Zaim bin Mohd Nor, for allowing me to conduct research on Tasik Aman lake.

I am thankful to UM Water Warriors for introducing me to the citizen science aspects of water sampling. I was inspired to develop soft sensors after attending their workshop and understanding the current predicaments of water sampling.

Lastly, I am thankful to Microsoft for their AI for Earth Grant, which allowed me to utilize Microsoft Azure's resources to develop this soft sensor. This work is also fully funded by the Ministry of Higher Education Malaysia through the Universiti Malaya's Impact Oriented Interdisciplinary Research Grant with grant number IIRG006A-2019.

Aug 2022,
Palok Biswas

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence
AMSA	Automatic Model Selection Algorithm
ANN	Artificial Neural Network
API	Application Programming Interface
ASV	Autonomous Surface Vehicle
AWS	Amazon Web Services
BOD	Biological Oxygen Demand
CART	Classification and Regression Tree
CB	Cubist Regression Trees
CDN	Content Delivery Network
CDOM	Coloured Dissolved Organic Matter
Chl-a	Chlorophyll-a
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DNN	Deep Neural Networks
DO	Dissolved Oxygen
DOE	Department of Environment Malaysia
DOE-WQI	Malaysian Department of Environment Water Quality Index
DT	Decision Tree
EFB	Exclusive Feature Bundling
ELR	Extreme Learning Machine Regression
ET	Extremely randomized Trees
GBDT	Gradient Boosting Decision Tree
GOSS	Gradient-Based One-Side Sampling
GPR	Gaussian Process Regression
HAB	Harmful Algal Bloom
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
ISE	Ion-selective electrode
JSON	JavaScript Object Notation
KNN	K-Nearest Neighbour
KRR	Kernel Ridge Regression

LightGBM	Light Gradient Boosting Machine
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MDN	Mixed Density Network
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
MVR	Multivariate Regression
NDWQS	National Drinking Water Quality Standard
NLWQS	National Lake Water Quality Standard
NRMSE	Normalized Root Mean Square
PLSR	Partial Least Square Regression
R2	Coefficient of Determination
RF	Random Forest
RF	Random Forest
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
SARIMA	Seasonal Auto Regressive Integrated Moving Average
SD	Secchi Depth
SDG	Sustainable Development Goal
SS	Suspended Solid
SSR	Sum of the Square of Residuals
SST	Total Sum of Squares
SVR	Support Vector Regression
TDS	Total Dissolved Solid
TN	Total Nitrogen
TP	Total Phosphorus
TSI	Trophic State Index
TSS	Total Suspended Solid
UAV	Unmanned Aerial Vehicle
USEPA	United States Environmental Protection Agency
WHO	World Health Organization
WQ	Water Quality
WQI	Water Quality Index
WSN	Wireless Sensor Networks
XGBoost	Extreme Gradient Boosting

CHAPTER 1: INTRODUCTION

1.1 Background

The concentration of nutrients (i.e. nitrogen and phosphorus) contributes to excessive phytoplankton and algae growth in the water body such as lakes (Ashraf et al., 2010). When the nutrient concentration reaches unhealthy levels, the lake is considered eutrophic. Eutrophication is a precursor to harmful algal blooms (HAB). HABs degrade water quality (WQ), reflected by the water's colour, smell, and taste, reducing transparency and dissolved oxygen (Nieto et al., 2019). Eutrophication is a global concern in the 21st century due to the importance of freshwater bodies for drinking water supply and biodiversity (Huo et al., 2013; Jimeno-Sáez et al., 2020; Mamun et al., 2020). About 60% of Malaysian lakes suffer from eutrophication (NAHRIM, 2009). More concerning, Sharip et al. (2014) further assessed 15 Malaysian lakes and found that all of them had become eutrophic. Malaysian lakes are vulnerable to algal blooms because of warmer temperatures, plentiful sunshine, and nutrient inflow in the lake during the rainy season.

The core WQ parameter in determining algae growth is the Chlorophyll-a (Chl-a). The Chl-a is the primary photosynthetic (green) pigment in all plants, algae, bacteria, cyanobacteria, and phototrophs (Keller et al., 2018; García-Nieto et al., 2020). The Chl-a absorbs wavelengths between 400-450 nm and 650-700 nm in the electromagnetic spectrum. Its molecular formula is $C_{55}H_{72}MgN_4O_5$ (Wurtsbaugh et al., 2019). The Chl-a concentration in lakes can be determined from water samples (Zeng & Li, 2015). Alternately, the water industry has manufactured special sensors or sondes based on the electromagnetic spectrum to measure Chl-a concentration values. The sondes are handheld and can be dipped at the lake to measure Chl-a concentration data in situ. The Chl-a sondes are often expensive, easily ranging between MYR10,000 and MYR20,000 starting price per sonde.

Profiling lakes for eutrophication usually centres around continuous monitoring of the Chl-a concentration levels (Su et al., 2015; Tian et al., 2017; Yi et al., 2018). Furthermore, shorter intervals are recommended for accuracy (Behmel et al., 2016; Hafeez et al., 2019). The requirements are due to environmental conditions' uncertain and dynamic nature (Castrillo & García, 2020). For example, the varying temperatures, carbon dioxide levels in the atmosphere, solar irradiance, rainfall distribution, and other meteorological parameters. Hydrological attributes, including inflow and outflow disrupting water levels and nutrient concentration, also contribute to the uncertain day-to-day conditions at the lake. Furthermore, contrasting daytime and nighttime conditions also heavily affect Chl-a concentration levels.

The unpredictable environment variables make traditional WQ monitoring methods such as grab sampling, which takes buckets of lake water to the lab, for Chl-a concentration level measurement less valuable. Grab sampling is tedious and time-consuming for continuous and real-time Chl-a concentration level monitoring, particularly at large lakes. The Chl-a sondes are sometimes better alternatives as they can be left at the lake, and sampling intervals can be preprogrammed. However, deploying the sondes is not cost-effective and can be vulnerable to vandalism. The sondes also require physical calibration once in a while. Therefore, it is common to see gaps of a few weeks or months in the middle of data collected using sondes.

Figure 1.1 shows WQ data collected via industry-grade sondes. The user dips the WQ sonde underwater and monitors the reading through a display unit. Additionally, the sonde usually has data storage capacity, with an additional fee. Data is recorded in a MicroSD card and manually transferred from sonde to another platform for processing. The sonde

is handheld for quick WQ reading. Alternatively, the sonde can be harnessed and left for more extended WQ readings.

Figure 1.1: The EXO WQ sonde with display for in-situ sampling

Interestingly, the Chl-a concentration is correlated with WQ parameters such as dissolved oxygen (DO), water temperature, electrical conductivity, and turbidity (Huo et al., 2013; Sharip et al., 2014; Mamun et al., 2018). Aside from these WQ parameters, Chl-a concentration is also highly correlated with the month of the year (Huo et al., 2013). The DO, temperature, conductivity, and turbidity sondes are less costly than the Chl-a sondes. They are common in various WQ lake monitoring activities, making them more accessible. Huo et al. (2013) and Jimeno-Sáez et al. (2020) pointed out the correlation between Chl-a and the DO, temperature, conductivity, turbidity, nutrients, and month. Castrillo & García (2020) found that nutrients (e.g., phosphorus and nitrogen) correlate with the DO, temperature, conductivity, and turbidity.

1.2 Motivation

For parameters that are difficult to measure in-situ or require an expensive sensor, it is possible to predict the parameter based on other available WQ parameters by training a machine learning (ML) model. The ML prediction is purely software-based, and the approach is called a soft sensor (Castrillo & García, 2020). The ML prediction is generally

of two types: deductive and inductive. Deductive models require an in-depth understanding of the complex biological, chemical and physical processes and are often time-consuming. On the other hand, the inductive solely focuses on the statistical correlation of features and patterns of the measured data to form a comprehensive understanding of the system. Algorithms like Artificial Neural Network (ANN) are an excellent example of inductive modelling that can model complexities of natural processes and predict parameters like the Chl-a (Tian et al., 2017). Figure 1.2 shows the approaches in soft sensor development.

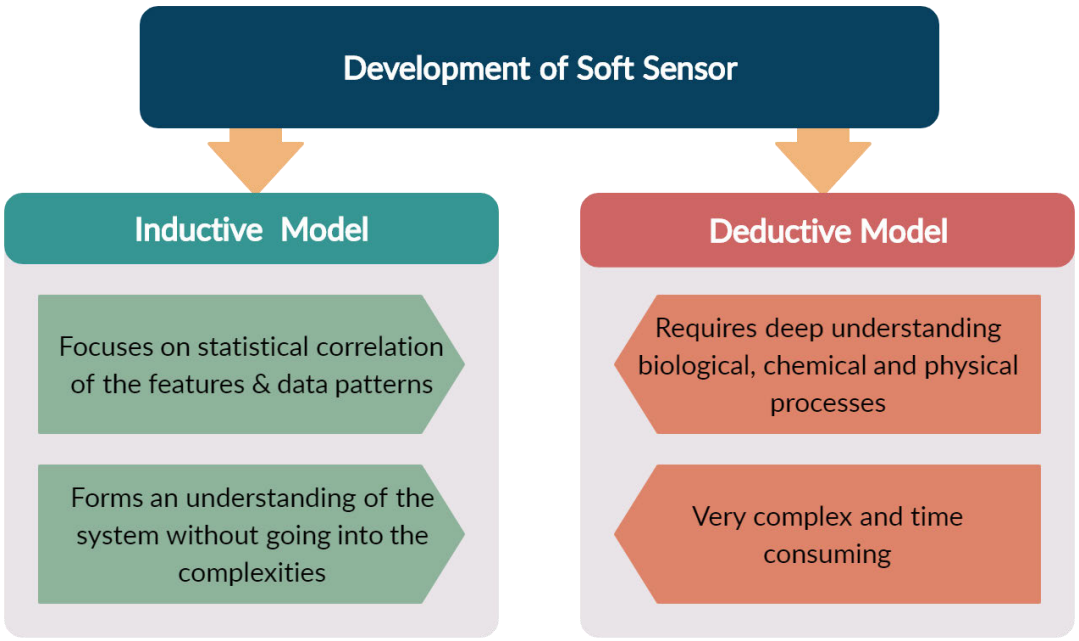

Figure 1.2: Soft sensor development approaches

Using ML, soft sensor development via inductive modelling can predict Chl-a concentration levels from WQ parameters. The WQ parameters include DO, temperature, conductivity and turbidity, total dissolved solids (TDS), pH, and nutrients such as phosphorus (TP) and nitrogen (TN). The ML models can address the nonlinear relationship between WQ parameters. For example, Huo et al. (2013) apply ANN, and their results improve when more WQ parameters are used. Tian et al. (2017) also propose ANN but find the temperature, total suspended solids (TSS), TP, and NP influence the Chl-a measurement.

The Deep Belief Network proposed by Zhang et al. (2016) takes in meteorological, hydrological and biochemical processes to obtain Chl-a concentration levels at the lake. The support vector regression (SVR) is tested by Jimeno-Sáez et al. (2020) and Mamun et al. (2020) to predict Chl-a concentration using different WQ parameters with reasonable accuracy. WQ researchers concluded the effectiveness of ML models in estimating Chl-a concentration by exploiting the correlation between Chl-a and WQ parameters. ML models demonstrate a significant performance boost when more than seven WQ parameters are used in training the ML models for Chl-a value prediction.

Sharip et al. (2014) reported that Malaysia follows the National Lake Water Quality Standards (NLWQS) and the National Drinking Water Quality Standards (NDWQS). These standards evaluate WQ parameters against the Water Quality Index (WQI) to test safe drinking water and assess the aquatic ecosystem. Researchers estimating the WQI standards also attempted ML-based models in their prediction. Examples of the ML models include the k-nearest neighbour (kNN), extreme learning machine, deep neural network, gradient boosting, polynomial regression, and SVR (Camejo et al., 2013; Yan et al., 2017; Shafi et al., 2018; Ahmed et al., 2019; Li et al., 2019). It is interesting to note that the deep learning techniques outperformed the kNN and SVR.

ML techniques are widely adopted in predicting WQ parameters because of their robust performance. ML algorithms such as the random forest (RF), extremely randomized trees (ET), and the Gaussian process regression have been proposed (Keller et al., 2018; Ruescas et al., 2018; Hafeez et al., 2019; Shehhi & Kaya; 2020). However, due to the lack of continuous WQ parameters data, these ML models were mainly trained on a small dataset collected over a limited period. The prediction accuracy in some ML models was average and required many input parameters. Moreover, the predictions were not made

with real-time data, and the models were usually trained and tested with data generated using expensive sensors by government bodies or private institutions.

The trend in developing ML-based soft sensors for WQ parameter measurement is encouraging. Soft sensors can act as an alternative and remove the dependency on expensive physical Chl-a sensors. Extending the soft sensor to an IoT cloud application enables real-time inferencing of Chl-a concentration based on selected WQ parameters data. One can upload compiled datasets of WQ parameters to the IoT cloud architecture for a remote Chl-a concentration level estimation. One can also pump live data to the IoT cloud architecture for continuous and real-time Chl-a concentration monitoring from the sondes. An IoT cloud with an online Chl-a concentration soft sensor is practical and exciting. Real-time lake profiling aids water conservation management and relevant authorities in quickly curbing the effects of eutrophication and HABs.

1.3 Problem Statement

There are strong correlations between Chl-a concentration and four basic WQ parameters: DO, temperature, conductivity and turbidity. Existing methods rely on historical, time-series data to predict Chl-a concentration levels. Historical WQ parameters can miss current environmental factors like temperature and humidity variation, rain distribution, and sudden pollution inflow at any given time. Therefore, live WQ parameters must be considered in estimating Chl-a concentration levels accurately. However, live collection of Chl-a data is scarce due to expensive Chl-a sondes. A Chl-a soft sensor can support live monitoring of Chl-a concentration level at the lake.

This study proposes the development of a Chl-a soft sensor using an ML model that can estimate Chl-a concentration levels. While most soft sensor approaches train their model with more than seven WQ parameters, this study explores the ML performance when only four WQ parameters are used. The aim is to reduce the constraint of WQ data collection when fewer WQ parameters are required. The ML model is trained to learn the correlation between Chl-a and the WQ parameters like DO, temperature, conductivity, and turbidity. For completion, this study also explores an IoT cloud application development to test the ML inferencing on the four WQ parameters. An IoT cloud application with ML inferencing can support the water community to monitor low-cost continuous Chl-a concentration levels.

1.4 Objectives of the Study

Three objectives have been identified to address the problem statement of the study:

1. To develop an ML model for Chl-a concentration measurement based on four basic WQ parameters (DO, temperature, conductivity, and turbidity)
2. To develop an IoT Cloud application with online ML inferencing for continuous and real-time monitoring
3. To compare the performance of the proposed developed models against the literature

1.5 Research Mapping

Table 1.1 shows the mapping of research questions, objectives, methods, and outcomes for the study. The first indicates the bulk of literature search on WQ parameters and measurements, supporting the study. The literature search directs what and where the progress is in the lake water conservation domain. The review helped identify gaps in tackling challenges for the water community, specifically, how applied computing can

contribute. Since this study focuses on applied computing contribution, the research objectives must produce outcomes related to computational development and assessment. Two computational outcomes are proposed, including developing an ML soft sensor for Chl-a and an IoT cloud application to test the ML inferencing. For completion, the study includes evaluating the computational outcomes to analyze their performance.

Table 1.1: Mapping between research questions, objectives, methodology and the study outcomes

Research Questions	Research Objectives	Methodology	Outcomes
What are the various methods for measuring Chl-a concentration in lake water?	To study different monitoring methods for efficient lake water sampling and the core parameters required to measure Chl-a concentration	a) Literature Search b) Systematic Literature Review	a) Findings on existing Chl-a concentration measurement approaches b) Findings on core parameters and their correlations in measuring Chl-a concentration
What are the core WQ parameters and their correlations in measuring Chl-a concentration?			
How can Chl-a concentration be measured using four WQ parameters?	To develop an ML model for Chl-a concentration measurement based on four WQ parameters	Model Development	A soft sensor for Chl-a concentration measurement based on four WQ parameters
How can Chl-a concentration monitoring be continuous and real-time?	To develop an IoT-cloud application with online ML inferencing	System Development	An IoT-cloud application with online ML inferencing
How does the performance of the proposed solutions differ from existing methods?	To compare the proposed solutions against existing measurement methods	Comparative Evaluation	Comparative analysis between proposed solutions against existing solutions

1.6 Research Scopes

The study proposed two computational outcomes. The first is an ML model that learns the correlation between four WQ parameters for Chl-a concentration measurement. The second is an IoT cloud application that tests the IoT architecture for real-time Chl-a inferencing. The development of the Chl-a soft sensor with ML and IoT contributes an end-to-end software solution toward Chl-a concentration estimation. It must be noted that the hardware requirements enabling the four WQ parameters (DO, temperature, conductivity and turbidity) data collection is outside the study's scope.

The scope of the study includes:

1. Only seven or fewer WQ parameters are to be used
2. The ML model must achieve at least 85% accuracy
3. The data collection has a minimum of 10-min intervals for a minimum of one month
4. The data collection must be done at a Malaysian lake with a climate of 25-30 degrees Celsius
5. The IoT architecture must support data publishing a minimum of once every second
6. The IoT architecture must support automatic scaling to accommodate data volume and traffic
7. The inference latency must be below 500ms
8. The IoT architecture must support bidirectional data communication

1.7 Significance or Impact of the Study

This research aims to improve WQ monitoring of water bodies, particularly the Chl-a concentration levels at lakes in Malaysia. Additionally, this study addresses the mitigation measures under the Environmental Quality Act (1974) and the National Water Resources Policy (2012). This study relates to the United Nation's Sustainable Development Goal 6 (SDG 6) on clean water and sanitation. Locally, the UN's SDG 6 aspects are reflected in the *Rancangan Malaysia ke-12*, especially on integrated water resource management.

1.8 Dissertation Organization

This dissertation is divided into five chapters. Chapter 1 introduces the research topic, including the background and motivation, problem statement, research questions, objectives and scopes. Chapter 2 provides a critical overview of the literature for topics of interest in this dissertation. Learning from other researchers on training the ML models to capture the correlation between WQ parameters and Chl-a concentration is central to the chapter. The methodology adopted in this dissertation is presented in Chapter 3. Included in the description are the proposed research framework and the overall process flow of the research. Chapter 4 presents the results and comprehensive analysis of the performance of the proposed method, while Chapter 5 forwards the final remark of the work done, concluding the dissertation.

CHAPTER 2: LITERATURE REVIEW

2.1 Overview

This chapter reviews the literature concerning eutrophication, HABs, WQ sampling procedures, and the WQ parameters correlation mapping. The later sections review popular ML models in the WQ domain and the various ML techniques to improve predictions. Before the chapter summary, a section reviewing IoT interests among WQ researchers is also included.

2.2 Eutrophication

Eutrophication is best described as the enrichment of nutrients, namely nitrogen and phosphorus, in the water body, causing abundant phytoplankton and plant growth (Ashraf et al., 2010). Excessive algal growth can also be attributed to higher temperatures, increased levels of carbon dioxide in the atmosphere, and solar irradiance – the ideal condition for photosynthesis. Eutrophication is prevalent in inland water bodies such as lakes and is essential to managers and scientists. This is because lakes are the primary source of freshwater supply and are at the centre of biodiversity. Other important usages of lakes include irrigation for agriculture, hydroelectricity, and recreational activities (Keller et al., 2018; García-Nieto et al., 2020).

Due to its vast impact on ecosystems, the economy and human health, eutrophication has become one of the most critical global concerns in the 21st century (Huo et al., 2013; Jimeno-Sáez et al., 2020; Mamun et al., 2020). Eutrophication disrupts aquatic ecosystems and substantially impacts agriculture, fisheries, recreational use of water bodies, and drinking water supply. Eutrophication degrades WQ and can be identified through water's colour, smell, and taste while reducing transparency and dissolved oxygen (Nieto et al., 2019).

In recent years, eutrophication has become more common in water bodies due to anthropogenic activities, also called cultural eutrophication. Scientists discovered a direct relationship between the discharge of nutrient-rich pollutants to water bodies at an increased rate of eutrophication (Chrislock et al., 2013). This dramatic rise of eutrophication due to rapid urbanization, land use, farming, and industrialization degrades surface and underground WQ (Gafri 2018). Agricultural practices and extensive land clearance significantly increased soil erosion. This further exacerbated lake WQ since lakes usually collect pollutants, nutrients, and sediments embedded in the eroded soils (Sharip et al., 2014).

Nutrient enrichment in water bodies from pollutants such as sewage, agricultural run-off, and industrial by-products (also known as effluents) can come from point and nonpoint sources. Point sources of pollution are characterized by pollutants entering the water body from any single identifiable source, such as a pipe, drainage channels or outlets from industries or wastewater treatment plants. Non-point sources are challenging to identify as pollution enters the water body from many sources, such as urban or agricultural run-offs. Urban run-offs usually occur when rainwater collects on impervious surfaces like roads, parking lots, and rooftops and mixes with pollutants, eventually ending in the water bodies. Stormwater can also mix with the fertilizers from farmlands and carry the nutrient-rich water to different water bodies like rivers, lakes, estuaries, and coastal waters (Koparan et al., 2018).

The trophic state is vital for effective waterbody management (Jimeno-Sáez et al., 2020; Mamun et al., 2020). The level of eutrophication in a lake or reservoir can be classified into trophic states determined by the algal biomass in the waterbody at a given time and place. The trophic state index (TSI) developed by Carlson (1977), also referred to as

Carlson's index is widely used to classify eutrophication levels into trophic classes such as Oligotrophic, Mesotrophic, Eutrophic, and Hypereutrophic. These trophic classes indicate the biomass of algae present in the water, and each class is defined as follows:

- Oligotrophic refers to clear water with little to no algae or biological productivity (considered good WQ).
- Mesotrophic refers to moderate algal biomass and biological productivity (considered fair WQ)
- Eutrophic and Hypereutrophic indicate abundant algal biomass and high biological productivity (considered poor WQ).

The TSI is a range from 0 to 100 and is calculated using three WQ parameters: Chlorophyll-a (Chl-a) concentration in micrograms per litre (µg/L), Total Phosphorus (TP) in µg/L and Secchi Depth (SD) measured in meters (m). Different values of TSI correspond to different trophic classes, as shown in Table 2.1.

Table 2.1: Trophic state classification based on Chl-a concentration (NAHRIM, 2015)

TSI	Chl-a (µg/L)	TP (µg/L)	SD (m)	Trophic Class
<30-40	0-2.6	0-12	>8-4	Oligotrophic
40-50	2.6-20	12-24	4-2	Mesotrophic
50-70	20-56	24-96	2-0.5	Eutrophic
70-100+	56-155+	96-384+	0.5-<0.25	Hypereutrophic

Besides TSI, other metrics are used to classify ecosystems based on biological productivity and algal biomass. The United States Environmental Protection Agency (US EPA) suggested limiting the annual average concentration to a maximum of 10 µg/L of Chlorophyll-a. If this limit is exceeded, the waterbody is deemed eutrophic. Another indicator based on Forsberg and Ryding's annual average total nitrogen concentration

classifies an eutrophic state if the concentration falls from 0.6 to 1.5 mg/L. However, these indicators are not universal and must be applied carefully considering geography and time (Yi et al., 2018). The TSI, for example, is developed for temperate lakes, and special care must be taken if the indicator is applied to lakes in tropical regions (Sharip et al., 2014).

2.3 Algal Blooms and Harmful Algal Blooms

The explosive growth of phytoplankton biomass due to eutrophication is called an algal bloom. Recently, algal blooms have become a primary global concern because of their adverse effect on the environment, society, and economy (Zhang et al., 2016; Choi et al., 2019; Jimeno-Sáez et al., 2020). Additionally, the frequency of algal blooms is projected yearly because of climate change (Lee et al., 2016; Shin et al., 2020). An exponential increase in phytoplankton concentration and macrophytes (aquatic plants) blooms, caused by eutrophication from excessive nutrient inflows, affects different water bodies, including rivers, lakes, reservoirs, estuaries, coastal waters, and oceans.

Algal growth, both unicellular microalgae (e.g., cyanobacteria) and multicellular macroalgae (e.g., seaweed), is similar to terrestrial plant growth. They grow in warm temperatures and require plenty of sunlight and nutrients. Hence, the algal population has explosive growth in the summer seasons (Lee et al., 2016; Yi et al., 2018; Du et al., 2018). Some algal blooms are harmless, but others can be lethal to aquatic life, animals, and humans (Li et al., 2018). This increase in the algae population causes discolouration of the waterbody, where the water turns green, brown, or bluish (Koparan et al., 2018). Satellites can detect this discolouration, as seen in Figure 2.1.

Figure 2.1: Satellite image of toxic algal bloom in Lake Erie ("Lake Erie Abloom", 2017)

Lakes are especially susceptible to eutrophication (Tian et al., 2019). Algal blooms that produce toxins and cause hypoxia, usually blue-green algae cyanobacteria (which produces harmful cyanotoxins), are termed harmful algal blooms (HABs) (García-Nieto et al., 2020). The toxins produced by algal blooms can kill fish, poison domestic and wild animals, disrupt water supply to households, industries, and farming, and even affect human health (Ashraf et al., 2010; Yi et al., 2018). Using HAB-infested water for recreational activities can also be hazardous. The World Health Organization (WHO) classifies HAB as a microbial hazard significantly impacting human health (Sharip & Suratman, 2017).

HABs are usually a direct consequence of human activities: increased pollution of inorganic nutrients such as nitrogen favours the growth of harmful cyanobacteria over the harmless phytoplankton-like diatoms (Yajima & Derot, 2018). Blooms block sunlight penetration. The photosynthesis from algal blooms and decomposition of dead algae,

therefore, depletes dissolved oxygen in the water to a dangerously low-level, causing hypoxia (partial lack of oxygen) or, worse, anoxia (total lack of oxygen) (Hafeez et al., 2019). Toxins coupled with hypoxia result in the death of aquatic organisms and severely disturb the ecosystem's equilibrium.

Besides affecting the health of aquatic organisms, animals, and humans, HABs can threaten water security and devastate the economy (Cho & Park. 2019). If HABs occur in lakes and reservoirs used for drinking water, it degrades the WQ and disrupts the drinking water supply to cities (Tian et al., 2017; García-Nieto et al., 2020). HABs from anthropogenic activities and environmental degradation result in substantial economic losses worldwide by affecting fisheries and recreational activities (Du et al., 2018). It is estimated that the U.S. spends USD 2.2 billion yearly while China spends almost 8% of their GDP to fight algal blooms caused by eutrophication (Chrislock et al., 2013; Zhang et al., 2016).

Sharip et al. (2014) and Tian et al. (2019) call for sustainable management of lakes to prevent eutrophication in the early stages to avoid potential HABs. A very high cost is involved in restoring a lake infested with HABs and macrophytes. Also, if eutrophication causing nutrient inflows are not controlled in the early stages, lakes can enter a non-reversible state called hysteresis. Several measures can be taken to either prevent the occurrence of HABs or treat the waterbody after it has been affected by severe algal blooms.

Preventive measures using prediction systems can lower the risks and provide early warnings a few weeks before potential algal blooms (Shin et al., 2020). This technology is adopted in countries like the USA, Canada, and South Korea to prevent algal blooms.

Post-treatment strategies are required for aquatic bodies already suffering from algal blooms. Several established processes that can be undertaken include algal fences, algae removal boats, and activated carbon treatments (Lee et al., 2016). Lakes with irreversible damages like hysteresis may require dredging and aeration coupled with long-term rehabilitation measures (Sharip et al., 2014).

2.4 Eutrophication and Harmful Algal Blooms in Malaysia

Malaysian lakes have significant socio-economic importance and play an essential role in freshwater supply, agriculture, fisheries, biodiversity preservation, hydroelectricity, and recreational activities. Over 98% of Malaysian freshwater comes from lakes, and 11% of hydroelectricity is generated from lakes and reservoirs (NAHRIM, 2014). Lakes provide habitat for various plants and animal species and serve as a natural flood mitigation system that collects water from urban catchments. However, Malaysian lakes are also particularly vulnerable to algal blooms because of the warmer temperature, plentiful sunshine, and nutrient inflow in the lake during heavy rainfall. (Sharip et al., 2014).

Eutrophication and algal blooms are prevalent in Malaysian inland aquatic bodies, hindering the proper functioning of the ecosystem. More than 60% of Malaysian lakes suffer from eutrophication because of abundant nutrients in the water, a by-product of anthropogenic activities (NAHRIM, 2009). Sharip et al. (2014) assessed the trophic state of Malaysian lakes and found all 15 lakes studied to be eutrophic. Algal and macrophyte blooms in urban lakes affected water usage, aesthetic aspects, and recreational use of the lakes in Malaysia (Sharip & Suratman, 2017).

Water pollution from nutrients and sediments deteriorates WQ, which recently became one of Malaysia's most significant water resource management problems. Pollution from point and nonpoint sources is most common in urban areas with high population density or where land clearance and logging activities are present (Hafeez et al., 2019). Agriculture consumes 70% of the freshwater supply in Malaysia, and the industry is the largest polluter (from fertilizers and pesticides) of rivers and lakes of Malaysia. Besides agriculture, treated and untreated sewage from the cities and industrial effluents contribute to the accelerated growth of HABs. According to Malaysia's Department of Environment (DOE), around 36.6% of Malaysian rivers were identified as 'slightly polluted', and another 5.2% were reported to be 'polluted' (Sakai et al., 2018).

The WQ parameters used to measure this pollution of rivers are Biological Oxygen Demand (BOD), Suspended Solids (SS), and Ammoniacal Nitrogen (NH_3-N) (Gafri, 2018). Assessment of water quality in Malaysia is generally carried out by using two nationally accepted standards, i.e., the National Lake Water Quality Standards (NLWQS) and the National Drinking Water Quality Standards (NDWQS) (Sharip et al., 2014). NLWQS assesses the WQ inland aquatic bodies by measuring physicochemical and nutrient parameters and mainly focuses on preserving the ecosystem and public health.

The NDWQS assesses water and river water safety (NAHRIM, 2014). NLWQS gives lake managers an idea of the suitability of the water for preserving aquatic biodiversity and suitability for recreational purposes (NAHRIM, 2015). Carlson's TSI is used alongside NLWQS to classify the trophic states of the lakes. The Malaysian Department of Environment Water Quality Index (DOE-WQI) is also used to assess lake WQ, but DOE-WQI is mainly developed for flowing waters (also known as lotic waters) as opposed to the mostly still waters (also termed as lentic waters) found in lakes.

2.5 Water Quality Monitoring

Water quality (WQ) monitoring is collecting water samples, measuring, and analysing their physicochemical, nutrient, and biological properties (Barzegar et al., 2020). Human and environmental health depends on good WQ; thus, accurate measurement becomes essential (Tuna et al., 2013). However, WQ is deteriorating rapidly due to economic development and climate change pollution. As a result, algal blooms are becoming more frequent worldwide. Water pollution also leads to transboundary water conflicts and hinders the sustainable management of water resources (Su et al., 2015).

Proactive measures in HABs prevention include regular WQ monitoring to curb its adverse effects (Tian et al., 2017). Periodic monitoring can help environmental managers to control lake eutrophication. It can be an early indication of algal blooms and help prevent HABs (Sharip & Yusop, 2007; Yi et al., 2018). Continuous WQ monitoring is also required for environmental risk assessment and understanding the impact of pollution and contamination (Hafeez et al., 2019). Data and insights from WQ assessments are vital to relevant stakeholders in understanding the lakes' status and making informed and timely decisions toward sustainable conservation (NAHRIM, 2014; Behmel et al., 2016; Castrillo & García, 2020).

2.5.1 Traditional Sampling

Spatially separated and high-frequency water sampling is crucial in managing freshwater resources and maintaining public health (Ore et al., 2015). Indicators for eutrophication such as nitrogen, phosphorus and phytoplankton concentration are traditionally measured using grab sampling (Castrillo & García, 2020). Usually, trained personnel must use boats for lake sampling and hire safety officers for the fieldwork (Podnar et al., 2010; Esakki et al., 2018). Also, large volumes of water samples, which might require pre-treatment,

must be collected and transported back to the laboratory for analysis (Zeng & Li, 2015; Azizul, 2019). One drawback to this approach is that the WQ sample collected from one location might not represent a large waterbody because of the spatial variability (Koparan et al., 2018; Keller et al., 2018). Due to this drawback, traditional sampling methods fail to provide holistic measurements of a particular region (Hafeez et al., 2019).

Although accurate, traditional laboratory testing methods to determine WQ are also not ideal for early contamination detection and may lead to a slow response. The sample must be transported to the laboratory for manual analysis, which is time-consuming (Barzegar et al., 2020). Also, transporting water samples might change the physicochemical properties of the water sample (Zeng & Li, 2015). This time delay between sample collection and relaying information to stakeholders also increases the response time during an emergency, such as natural disasters (Saab et al., 2017; Tian et al., 2017; Yang et al., 2018). Hence, timely, informed decisions cannot be made with traditional water sampling methods (Siyang & Kerdcharoen, 2016; Zhu et al., 2018).

Besides being slow, conventional water sampling is inefficient and infrequent because of the time and manpower constraints (Zhang et al., 2016; Esakki et al., 2018; Hafeez et al., 2019; Barzegar et al., 2020). In contrast, periodic, high-frequency monitoring is required for the early detection and prevention of eutrophication (Saab et al., 2017). Moreover, infrequent measurement of nutrients like nitrogen and phosphorus from eutrophic water bodies may underestimate the severity of the problem. They can also risk further degradation of the waterbody (Tian et al., 2017; Castrillo & García, 2020). The logistics of field water sampling also come with considerable disadvantages that prevent high-frequency sampling, which include:

- Multiple trained personnel to carry out water sampling, and the frequency of sampling depends on the availability of the trained staff (Tian et al., 2017).
- Safety concerns in sending a team of people to carry out the sampling. For example, sampling cyanobacteria-infested water puts the sampler's health at risk (Podnar et al., 2010; Koparan et al., 2018).

2.5.2 In Situ Sampling

In situ is a Latin phrase that means 'on site'. In-situ sampling measures the WQ of the waterbody without isolating the water sample from the waterbody. Compared to conventional laboratory testing, in situ measurements are fast, simple, and allow sampling at multiple locations and depths (Zeng & Li, 2015). Due to in situ sampling's convenience and accuracy, it has become a popular choice for WQ measurement in the 21st century (Leeuw et al., 2013). In situ water parameter sensing sondes are famous for measuring the physicochemical properties of the water. It is often used alongside manual water sampling for bacteriological analysis in the laboratory (Al-Badaii et al., 2013).

Commercially available in-situ WQ sensors are of four types:

1. Ion-selective electrode (ISE). This sensor is designed to measure the concentration of a specific dissolved ion. The sensor contains an ion-selective membrane and two electrodes that measure the activity of the ion through the potential difference between the electrodes (Barker, 2020).
2. Wet Chemical. This process usually measures the sample by mixing specific reagents, giving the sample a particular colour based on the concentration of the chemical being analysed. The intensity of the colour is then measured electronically through a spectrophotometer in the sensor via absorbance (Barker, 2020).

3. Optical. The sensor emits a light that makes the sensing element in the sensor luminesce. When the sample is introduced, it changes this luminescence which is electronically detected and is proportional to the sample being measured (Antylia Scientific Blog, 2015).
4. Spectral. Spectral encompasses both multi- and hyperspectral sensors, and they measure spectral bands to assess the chemical composition of the water sample. An essential difference between multi and hyper-spectral sensors is that multi-spectral deals with 3-10 bands, whereas hyper-spectral deals with thousands of bands (GISGeography, 2021).

WQ sensors for in situ measurements are often expensive (Leeuw et al., 2013; Keller et al., 2018; Ahmed et al., 2019). Out of these four options, only ISE sensors are cost-effective. However, ISE is more susceptible to sensor drifting and inaccuracies. Wet chemical, optical, and spectral sensors are accurate and precise but expensive and require expensive routine maintenance (Castrillo & García, 2020). These in situ sensor sondes, especially the portable ones, are often not Internet of Things (IoT) enabled and cannot be used for real-time WQ monitoring (Tuna et al., 2013). The IoT-enabled sondes that are commercially available are very expensive and require significant initial investments (Esakki et al., 2018). Despite the high cost, Wiranto et al. (2015) state a high demand for online sensors that can be used for environmental telemonitoring to detect pollutant inflows.

2.5.3 Online Real-Time Monitoring

High-frequency real-time monitoring of WQ parameters is crucial for detecting the onset of algal blooms or contamination in the early stages (Dunbabin & Grinham, 2010; Saab et al., 2017). Remote monitoring also has the advantage of monitoring the water for a

long time to understand the long-term variations of the parameters (Koparan et al., 2018). Online monitoring is already changing how we understand our environment by producing big data which can be analysed to uncover new insights (Zhang et al., 2016).

With rapid urbanization, population growth, industrialization, and climate change in recent years, real-time monitoring and analysis have become even more crucial (Gafri et al., 2018). Continuous real-time monitoring, through IoT-enabled sensors, can streamline decision-making for water conservation managers and policymakers (Detweiler et al., 2015; Yang et al., 2018). Environmental authorities in charge of curbing residential and industrial pollutants can also highly benefit from the real-time data informing them about the current pollution levels (Wiranto et al., 2015). Despite the advantages of real-time water monitoring, its adoption is slow because commercially available IoT-enabled multiparameter sondes are often quite expensive (Joslyn & Lipor, 2018).

2.5.4 In-Situ Monitoring Strategies

Many researchers have attempted to optimize WQ monitoring programs by employing various data collection strategies with in-situ sensors, namely static (fixed-site monitoring) and dynamic (mobile vehicle equipped with multiparameter sondes). The static strategy was implemented by Tuna et al. (2013) to monitor the reservoir WQ storing drinking water. The WQ was monitored in real-time using a wireless sensor network (WSN) that could collect large data samples necessary to assess drinking WQ.

Li et al. (2017) deployed sensor nodes in a water body to collect data remotely and monitor WQ. With this setup, they could generate high-resolution long-term data that will be useful for developing environmental models to understand the long-term dynamics of the water body. Saab et al. (2017) also used a distributed sensor network to monitor water

quality in real-time to prevent degradation in the early stages. They concluded that these in situ sensors were as effective as laboratory measurements. However, all these studies used expensive sensors to monitor WQ online and require a high initial investment.

The dynamic strategy involves attaching in situ sensors (either online or offline) to a mobile vehicle to generate high-resolution spatiotemporal data spanning large spatial regions over a long period (Detweiler et al., 2015). Such implementations were observed in studies by Yang et al. (2018) and Koparan et al. (2019), where an Unmanned Aerial Vehicle (UAV) was equipped with a multiparameter sonde for in situ samplings. In addition to quick and real-time sampling across multiple regions, it reduces operational costs and solves safety issues by sampling inaccessible or hazardous regions without needing trained professionals. Dunbabin & Grinham (2010) developed an Autonomous Surface Vehicle (ASV) to collect high-resolution spatiotemporal water quality in real-time for early algal bloom detection. In another study by Podnar et al. (2010), autonomous robot sensor boats were developed with multiparameter sondes. After heavy rainfall, the sensors analyze algal growth from fertilizer pollutants entering the lake.

These studies show that IoT in situ sensors overcome several disadvantages of traditional sampling. It can provide real-time, high-resolution spatiotemporal data, eliminating the need for trained practitioners or transporting water samples to the laboratory. IoT sensors also automate data collection and structuring, thereby eliminating human errors. However, none of these studies examined the potential of low-cost multiparameter sensors in WQ monitoring. Inexpensive IoT sensors can potentially increase their adaptation in developing nations and open new horizons for citizen scientists.

2.5.5 Community-based Water Quality Monitoring

Lakes are an integral part of the community with traditional and historical influence. Raising awareness of the issues faced by the lake and the environment at large can propel the community to protect it. Examples of sustainable management of lakes can be seen in many countries where the success can be directly attributed to community involvement (Sharip et al., 2014). Citizen science, as a result, plays a vital role in monitoring, managing water bodies and raising awareness about the local issues that need attention. A recent study by Aronoff et al. (2021) has shown the effectiveness of community science where motivated people from local communities carry out cost-effective and successful WQ monitoring. Sakai et al. (2018) have analysed six WQ parameters at the Universiti Malaya's Varsiti Lake, measured and reported by local citizen scientists calling themselves the 'UM Water Warriors'. They carry out WQ sampling using low-cost WQ tools.

The low-cost methods used by the UM Water Warriors and citizen scientists alike are sometimes unreliable because of the inaccuracies involved in their methods. The tools used can provide a baseline understanding of the water quality but are inaccurate. Also, the manual methods used to measure WQ parameters such as Secchi Depth and colour indicators for certain chemicals in the water are susceptible to human errors. These issues could be resolved by having low-cost IoT in situ sensors that automatically measure and store data. It can also encourage people to participate in water conservation and take collective action to sustain water resources.

2.6 Water Quality Parameters

Good WQ monitoring programs can ensure the proper functioning of ecosystems, preserve biodiversity, and protect public health (Castrillo & García, 2020). In situ physicochemical properties for inland waters that can determine water quality deterioration are DO, temperature, conductivity, turbidity and pH (Koparan et al., 2018). These WQ parameters are also part of NLWQS, the Water Framework Directive, and the United States Environmental Protection Agency (Tuna et al., 2013; Sharip & Suratman, 2017). Biological parameters, like cyanobacteria, can also identify HABs (Sharip et al., 2014). Physicochemical and biological variables are the main drivers for eutrophication (Nieto et al., 2019). However, measuring biological parameters usually require laboratory testing, which is time-consuming and labour-intensive (Choi et al., 2019).

Secchi Depth (SD) measures the water's transparency and obtains the TSI of the water body. SD is one of the critical indicators for eutrophication (Huo et al., 2013) and is also crucial for water body management strategies (Mamun et al., 2020). Huo et al. (2013) found that SD influences algal biomass growth by determining sunlight penetration. Li et al. (2018) also found SD to be one of the most influential factors affecting algae growth. Sharip et al. (2014) reported that SD negatively correlated with conductivity, a WQ parameter that measures the concentration of mobile ions in the water through conductance. High turbidity decreases SD, indicating either high algal biomass or sediments in the water.

Interestingly, SD can be inferred from turbidity as there is a significant correlation between the two parameters. Turbidity measures the cloudiness of the water and is an essential parameter for water body management (Joslyn & Lipor, 2018). Turbid water affects aquatic fauna because it limits sunlight penetration in the water affecting algal

growth (Keller et al., 2018). Total Suspended Solids (TSS) is another crucial WQ parameter required in the sustainable management of water bodies (Silveira et al., 2020) which is also correlated with turbidity (Sharip et al., 2014).

Besides turbidity, DO is another critical indicator for eutrophication (Huo et al., 2013; Joslyn & Lipor, 2018). DO monitors lakes because it can detect oxygen production via photosynthesis by algal blooms, oxygen consumption by aquatic organisms, and chemical oxidation in the water (Barzegar et al., 2020). Monitoring DO production and depletion can reflect nutrient contents that characterize lakes (Huo et al., 2013). DO correlate with pH as an increase in the oxygen levels in the water increases the concentration of hydroxide ions, raising the pH (Sharip et al., 2014).

Dissolved nutrient concentrations, mainly Total Phosphorus (TP) and Total Nitrogen (TN), are also indicators of eutrophication (Huo et al., 2013). They can also assess the lakes' trophic state (NAHRIM, 2015). Although Yajima & Derot (2018) found both TP and TN to be the main drivers of algal biomass proliferation, other authors emphasised that TP concentration is the leading nutrient indicator for eutrophication and algal biomass (Sharip & Yusop, 2007; Mamun et al., 2018; Shin et al., 2020).

2.6.1 Chlorophyll-a

Chlorophyll-a (or Chl-a) is one of the primary pigments of the chlorophyll pigment family (e.g., pigments a, b, c, d, e). Organisms that carry out photosynthesis, like phytoplankton and cyanobacteria, usually have Chl-a pigments (Keller et al., 2018; García-Nieto et al., 2020). Chl-a is the primary indicator of the phytoplankton biomass in freshwater and saltwater (Zeng & Li, 2015). The European Commission Water Framework Directive standardises Chl-a to measure phytoplankton abundance (García-Nieto et al., 2020). Chl-

a is widely used to forecast algal blooms (Su et al., 2015; Lee et al., 2016; Tian et al., 2017; Yi et al., 2018; Li et al., 2018; Yajima & Derot, 2018; Du et al., 2018; Hafeez et al., 2019; Nieto et al., 2019; Choi et al., 2019; Jimeno-Sáez et al., 2020; Shin et al., 2020).

Du et al. (2018) reported Chl-a to have solid seasonal properties where the concentration increases during summer and autumn and decreases in winter. This finding is consistent with algae growth, as algal blooms (or HAB) are more common during warmer seasons (Lee et al., 2016; Yi et al., 2018). An increase in the concentration of Chl-a can indicate HAB, and it can be an alternative indicator of cyanotoxins from cyanobacteria blooms. Hence, Chl-a concentration can also serve as an early warning for HABs and aid in managing the affected water body (García-Nieto et al., 2020).

Eutrophication, a precursor to HAB, provides an ideal growth environment for algae. Chl-a can determine eutrophication, specifically the presence of nutrients such as phosphorus or nitrogen in lakes, reservoirs, and oceans (Keller et al., 2018; Mamun et al., 2020; García-Nieto et al., 2020). Chl-a concentration is one of the parameters used by the US EPA for TSI classification (Mamun et al., 2020). Barzegar et al. (2020) also underscore the importance of Chl-a concentration in monitoring and managing lakes.

Being a pigment, Chl-a has specific optical properties that can be measured by fluorometers, spectral sensors or satellites capable of identifying Chl-a by measuring the spectrum of the reflected light (García-Nieto et al., 2020). In situ monitoring is done with fluorometers that measure the Chl-a molecule's fluorescence. A red light is re-emitted by Chl-a when it is excited by a light source (Zeng & Li, 2015). Satellite images can be used to determine Chl-a over a large water area and are commonly used in remote ocean monitoring (Zeng & Li, 2015).

2.6.2 Relationship of HABs with WQ and Meteorological Parameters

WQ parameters are interdependent and interrelated, increasing the complexity in eutrophication prediction (Huo et al., 2013). In addition, Chl-a exhibits highly nonlinear relationships with other WQ parameters (Jimeno-Sáez et al., 2020; García-Nieto et al., 2020). Examples from the literature to illustrate the complex relationships between WQ parameters are given below:

- DO strongly correlates positively with Chl-a, pH and temperature (Sharip et al., 2014). The fact is also supported by Huo et al. (2013), where the authors found the relationship between Chl-a growth with DO and WT.
- SD negatively correlates with turbidity, TSS and conductivity (Sharip et al., 2014). However, conductivity and Chl-a are highly correlated (Lee et al., 2016).
- Chl-a is also significantly correlated with temperature, DO, pH, TN and TP (Huo et al., 2013). Mamun et al. (2018) also supported the correlation between TP and Chl-a.
- Sharip et al. (2014) pointed out the positive correlation between TP and TSS and the negative correlation between TP with temperature and SD.

Besides the WQ indicators, meteorological and hydrological parameters also influence the growth of algae (Zhang et al., 2016; Tian et al., 2017; Yi et al., 2018; Shin et al., 2020). Meteorology studies the earth's atmosphere and is mainly concerned with the weather. Air temperature, solar irradiance, and rainfall are considered meteorological variables. On the other hand, hydrology is concerned with the movement and distribution of water resources, and some relevant variables are water level, inflow, and outflow. An increase in nutrients in water and rainfall has a direct relationship and was pointed out by Mamun et al. (2018). Chl-a concentration is reported to strongly correlate with the month of the year (Huo et al., 2013).

Due to the complex relationships described above, it is possible to use state-of-the-art Machine Learning and Artificial Intelligence techniques to understand this relationship. With the advent of the IoT, data from different sources such as the environment and water are becoming increasingly available, and it paves the way to new opportunities in estimating and forecasting eutrophication and algal blooms.

2.7 Machine Learning Approaches to WQ Monitoring and Management

Due to algal blooms' complex and nonlinear relationship with physicochemical, biological, and meteorological variables, algal bloom prediction has been interesting for many researchers (Zhang et al., 2016). With the recent developments of machine learning (ML) and artificial intelligence (AI), researchers have applied various ML techniques to grasp this complex nonlinear relationship of algal blooms with the other variables (Huo et al., 2013). ML algorithms have been proven to decipher this complex relationship with acceptable accuracy. As a result, ML has become a promising tool for algal blooms forecasting and has been used successfully for the past two decades (Huo et al., 2013; Tian et al., 2017; Mamun et al., 2020). ML makes it possible to understand the complex relationship holistically, extracting patterns based on correlations useful for water resource management (Tian et al., 2017).

Forecasting using ML enables real-time WQ assessment using IoT-enabled sensors (Jimeno-Sáez et al., 2020). As pointed out by Barzegar et al. (2020), this alternative method of WQ monitoring and forecasting using AI has distinct advantages over traditional methods, which are:

- Monitoring and forecasting with AI is fast and cost-effective
- Eliminates the need for trained professionals for regular water sampling

- Simplifies the complex nonlinear relationships of algal blooms and makes it easy to understand
- AI techniques can estimate water quality parameters for difficult-to-access water bodies

2.7.1 Machine Learning Prediction and Forecasting of Chlorophyll-a

This section surveyed different AI and ML techniques used for Chl-a prediction, and the findings are summarized in Table 2.2. In the reviewed studies, future Chl-a concentration was either estimated via the correlation between WQ parameters or forecasted based on the time-series Chl-a concentration data.

In Chl-a prediction literature, the number of data samples used to train the ML models is usually minimal due to the lack of WQ data available. Only Barzegar et al. (2020) and Du et al. (2018) used an extensive dataset to train their models, as seen in Table 2.2. Barzegar et al. (2020) used a hybrid deep learning model which combines Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) to predict Chla-a and DO but achieved a relatively low R squared (R^2) value compared to others. The R^2 score, also called the coefficient of determination, is a metric to evaluate the performance of a regression-based ML model. R^2 score measures how well the predictor variables explain the variances in the predicted value. R^2 is also simply referred to as the ‘goodness-of-fit’ because it measures how well the predictions approximate the real values.

Du et al. (2018) used 8000 Chl-a concentration data points collected hourly and carried out time-series forecasting based on the trends with an R^2 value of 0.91. Also, most WQ data used in this study are monthly data, with only a few studies developing their models on high-frequency data (i.e. every minute or hourly). Real-time alerts for HABs with models trained on monthly data are not feasible.

Lee et al. (2016) and Nieto et al. (2019) achieved a high R^2 of over 90% on a minimal sample size, indicating a possible overfit. Almost all the papers reviewed in this section carried out Chl-a concentration prediction based on historical Chl-a values. For example, Su et al. (2015) trained their model from 2000 to 2004 and tested it from 2005 to 2010. Similarly, Yajima & Derot (2018) trained their model on data collected from 1999 to 2010, whereas they carried out their study in 2018. Choi et al. (2019) achieved an R^2 value of over 0.90 and used over 2000 data samples to train the model, but the focus was more on data imbalance than model performance.

The 'WQ Parameters' column of Table 2.2 show researchers using seven or more input parameters for Chl-a prediction. The maximum was 33 WQ input parameters by Yajima & Derot (2018), and the least was seven by Keller et al. (2018). The number of WQ parameters can influence the prediction performance. It is generally thought that more can improve the prediction performance, but the results show it can also degrade when the input parameters are redundant. In the case of the Support Vector Regression (SVR) model, the feature selection step is a must before training, which is cumbersome.

The studies show that the dataset used in training the ML models was also based on historical data that are primarily monthly intervals. The prediction was not demonstrated with online monitoring using IoT sensors. Also, none of the authors built a soft sensor for Chl-a. The only study that developed a soft sensor is Castrillo & Garcia (2020), but the soft sensor is for nutrients, not Chl-a.

The 'ML Algorithms in Table 2.2 lists the model suggested to be the best performing model for each study, while the 'ML Compared Against' column lists the models for comparative analysis. It can be stated from Table 2.2, irrespective of the water bodies, that there is no single ML model that performs consistently across different water bodies. The pattern is consistent with findings that a waterbody's physicochemical properties vary not only from different waterbody types but also depend on the location, climate, and many other environmental factors (Sharip et al., 2014; Koparan et al., 2018; Keller et al., 2018).

Hence, researchers have no consensus on the best performing ML model to predict Chl-a. Also, different researchers use different performance evaluation metrics to assess their models and the metrics are not standardized in WQ literature. The varying research methodologies and technical assessments make comparing ML models between different works challenging. Table 2.2. summarizes them.

Table 2.2: Literature review summary for Chl-a prediction using ML in different water bodies

Water Body	References	ML Algorithm	ML Compared Against	WQ Parameters	Data Frequency	Total Data	Results
Lake	Huo et al. (2013)	BP-ANN	RBF-ANN	10	Monthly	100	R^2=0.70
	Lee et al. (2016)	NGA (GA-ANN)	-	15	Monthly	60	R^2=0.91
	Li et al. (2018)	RF	SVR	7	Monthly	391	R^2=0.82
	Nieto et al. (2019)	SVM-ABC	MLP & M5	15	Monthly	244	R^2=0.92
	Choi et al. (2019)	CNN	-	8	Daily	2190	R^2=0.92
	Barzegar et al. (2020)	CNN-LSTM	SVR, DT	6	15 min	35000	R^2=0.76
Reservoir	Su et al. (2015)	GA-SVR	-	7	Monthly	100	R^2=0.82
	Tian et al. (2017)	ANN (optimized)	ANN (traditional)	7	10 min	1152	R^2=0.91
	Yajima & Derot (2018)	RF	-	33 & 25	Monthly	1000	R^2=0.61
	Tian et al. (2019)	FNN with TL	RNN, LSTM	7	5 min	1440	RMSE=0.3647
	Mamun et al. (2020)	SVR	MLR, ANN	10	Monthly	319	R^2=0.80
	García-Nieto et al. (2020)	LBFGSB-GPR	MLP, SVR & RF	20	Monthly	268	R^2=0.86
River	Keller et al. (2018)	ANN, SVM, ET	RF, AdaBoost, etc.	5	0.5-5 min	-	R^2=91.4
	Yi et al. (2018)	ELM	ANN	18	Weekly	200	R^2=0.47
	Cho & Park (2019)	Merged LSTM	LSTM & MLP	12	Daily	981	RMSE=0.0459
	Shin et al. (2020)	RNN	SVR, RF, XGBoost, LSTM	11	Daily	922	RMSE=2.6453
Coastal Water	Zhang et al. (2016)	DBN	BP-NN	11	Bi-Weekly	1000	RMSE=0.0475
	Du et al. (2018)	WNARNet	ANN, ARIMA, etc.	-	Hourly	8000	R^2=0.91
	Jimeno-Sáez et al. (2020)	SVR	MLNN	9	Daily	126	R^2=0.68

2.7.2 WQ Soft Sensor Development

Soft sensors are virtual sensors used to estimate the value of a target variable (usually difficult to measure) using other predictor variables that are easy to measure with high spatiotemporal resolution (Liu et al., 2016; Castrillo & García, 2020). The predictor variables are proxies or surrogates and directly correlate with the target variable, which the ML models can utilize to build a soft sensor (Castrillo & García, 2020). Developing a soft sensor for Chl-a estimation is just since Chl-a measurement usually requires expensive in situ fluorometric sensors (Leeuw et al., 2013), hyperspectral sensors, satellite imaging or laboratory measurements.

Predictive models of WQ parameters are generally of two types: deductive and inductive. Deductive models require an in-depth understanding of the complex biological, chemical, and physical processes and are often time-consuming. On the other hand, the inductive solely focuses on the statistical correlation of features and patterns of the measured data to form a comprehensive understanding of the system. Algorithms like ANN are good examples of inductive modelling and have been used to predict parameters like Chl-a (Tian et al., 2017).

Castrillo & García (2020) demonstrated the effectiveness of nutrient soft sensors where phosphorus and nitrogen-based nutrients were predicted using in situ WQ parameters like DO, temperature, conductivity, turbidity, pH, and chlorophyll-a. The trend in using ML to provide fast and reliable predictions without needing a physical sensor or laboratory tests has been ongoing for the last two decades (Tian et al., 2017). Castrillo & García (2020) remarked that researchers are gaining trust for online soft sensors and acknowledge their convenience over expensive, time-consuming laboratory testing. The high cost of in situ Chl-a sensors discourages its use, particularly in developing countries.

Measuring the Chl-a with laboratory or satellite methods means that the collected data will not have a high spatiotemporal resolution. These problems result in the scarcity of WQ data related to Chl-a. Scarce WQ data collection is a concern in the literature (Podnar et al., 2010; Keller et al., 2018; Du et al., 2018; Shin et al., 2020).

2.7.3 WQ Monitoring via Satellite Remote Sensing

Remote sensing technology uses sensors, typically mounted on satellites, to measure the physical characteristics of the earth's surface. Remote sensing is quite popular in WQ monitoring as it allows researchers to determine algal blooms in lakes, rivers, and oceans. Recently, much research has been done on ML to remote sensing data.

Hafeez et al. (2019) tested ANN, SVR, RF, Multivariate Regression (MVR) and Cubist Regression Trees (CB) on datasets from two different sources. The first dataset was obtained from the spectral reflectance sensor of the Landsat 5, 7 and 8 satellites, while the second was obtained from in situ Multispectral Radiometer. Hafeez et al. (2019) concluded that ANN outperformed other ML models on the coastal water dataset with a coefficient of determination, R^2 of 0.79 and Root Mean Squared Error (RMSE) of 0.27. ANN has a good prediction accuracy for Chl-a using Sentinel-2 satellite's spectral images lakes with R^2=0.77 and RMSE=2.859 (Silveira et al., 2020).

RF and K-Nearest Neighbours (KNN) were also strong contenders compared to ANN, as Silveira et al. (2020) reported. Shehhi & Kaya (2020) also used ANN to predict Chl-a in coastal waters using satellite data and compared the performance of ANN against MVR and Seasonal AutoRegressive Integrated Moving Average (SARIMA). However, the results obtained by Shehhi & Kaya (2020) are one of the lowest within the literature reviewed in this section.

Pahlevan et al. (2020) retrieved Chl-a concentration from coastal and inland waters using Sentinel 2 and 3 satellite's reflectance data coupled with in situ hyperspectral radiometric data. The authors suggested a new model called Mixed Density Network (MDN), which outperformed existing algorithms in the field of remote sensings such as the polynomial coefficients of blue-green algae ratio (termed OC) and Blend (an algorithm combing the two algorithms OCx and red-NIR 2-Band ratio). The MDN model, however, had high RMSE for Chl-a retrievals. Blix & Eltoft (2018) proposed an algorithm selection framework Automatic Model Selection Algorithm (AMSA), that selects the top-performing model based on the dataset. The AMSA model selected Gaussian Process Regression (GPR) to be the best in predicting Chl-a with R^2 of 0.8973 and Normalized Root Mean Square (NRMSE) of 0.1497. The SVR was also one of the top performers, but the Partial Least Square Regression (PLSR) did not perform well.

However, Martinez et al. (2020) claimed SVR to be the best model for reconstructing oceanic Chl-a on the surface, but the R^2 achieved for the model was 0.7921. Ruescas et al. (2018) used reflectance data from the Sentinel-3 satellite to retrieve Chl-a, Coloured Dissolved Organic Matter (CDOM) and TSS in coastal and inland waters. The authors compared the performance of RF, MVR, GPR, SVR and Kernel Ridge Regression (KRR) models and found RF to have the highest accuracy in retrieving Chl-a with R^2 of 0.75 and RMSE of 12.656. Ruescas et al. (2018) & Martinez et al. (2020) achieved R^2 values that are low compared to other works reviewed in this section. Peterson et al. (2019) achieved R^2 of 0.8972 and RMSE of 4.5948 for rivers and lakes by fusing satellite data and in situ water quality measurement. The authors built an ensemble model using PLSR, GPR, SVR, and Extreme Learning Machine Regression (ELR) models and found this combination effective in predicting Chl-a.

Syariz et al. (2020) proposed a novel model named 'WaterNet' based on CNN to retrieve Chl-a concentration in lakes using Sentinel-2 satellite data combined with in situ measurements. WaterNet could predict Chl-a from the dataset with acceptable accuracy and had an RMSE of 1.369. In a different paper, Yussof et al. (2020) tried to predict HABs in West Malaysian coastal waters from satellite data using LSTM. They concluded that LSTM performed better than CNN, although the model's performance was inferior, the lowest of all the articles reviewed in this section. Table 2.3 lists the application of remote sensing and ML modelling to retrieve Chl-a concentration levels in water bodies.

Table 2.3: Remote Sensing for Chlorophyll-a retrieval using Machine Learning

References	Key Algorithms	Compared Against	Water Body Type	Results
Blix & Eltoft (2018)	GPR	SVR, PLSR	Ocean	R^2 = 0.8973 NRMSE=0.1497
Ruescas et al. (2018)	RF	GPR, MVR, KRR, SVR	Coastal & Lake Water	R^2= 0.75 RMSE=12.656
Hafeez et al. (2019)	ANN	SVR, RF, CB, MVR	Coastal Water	R^2=0.79 RMSE=0.27
Peterson et al. (2019)	DLF Ensemble	-	Lake & River	R^2= 0.8972 RMSE=4.5948
Silveira et al. (2020)	ANN	RF, SVR, etc	Lake & Dam	R^2=0.9 RMSE=0.07
Martinez et al. (2020)	SVR	-	Ocean	R^2=0.7921
Pahlevan et al. (2020)	MDN	Blend, OC	Coastal & Lake Water	RMSE=30.31 MAE=1.275
Syariz et al. (2020)	CNN	ANN	Lake	RMSE=1.369
Shehhi & Kaya (2020)	ANN	SARIMA, MVR	Coastal Water	R^2=0.77 RMSE=2.859
Yussof et al. (2020)	LSTM	CNN	Coastal Water	R^2=0.1145 RMSE=3.402142

Satellite data can help determine the Chl-a concentration remotely. The images can help researchers determine the spatial variability of algae over the entire water body, especially if the water body is large and challenging to reach. It can also detect algal bloom hotspots, point sources of pollution and guide scientists to the exact location of the water body that needs to be sampled (Hafeez et al., 2019)

The main limitation of satellite remote sensing is the need to couple with in situ measurements from multi or hyperspectral sensors to calibrate and validate the satellite data (Koparan et al., 2018; Hafeez et al., 2019; Syariz et al., 2020). Also, satellite data can only measure specific WQ parameters that are optically active such as Chl-a, TSS, Coloured Dissolved Organic Matter (CDOM) and turbidity (Hafeez et al., 2019). Although it can measure Chl-a over a large water body, it cannot detect small changes in its concentration (Zeng & Li, 2015). Additionally, satellites cannot monitor WQ at different depths, which is crucial for lakes (Hafeez et al., 2019). Clouds obstructing lands cause satellite image issues, as Yussof et al. (2021) pointed out. Hence, satellite data can be used to monitor WQ only on cloud-free days, and the image should not contain any sunglints (Hafeez et al., 2019).

In summary, the accuracy of satellite data is as good as its atmospheric correction methods. If the atmospheric correction is not accurate, it will impact the accuracy of the satellite data. Also, atmospheric correction requires many preprocessing steps to be carried out on the satellite data, which is usually computationally demanding (Hafeez et al., 2019).

2.8 Machine Learning Techniques and Performance Evaluation Metrics

ML is becoming increasingly popular as an effective tool for modelling the nonlinearity of Chl-a and predicting algal blooms (Mamun et al., 2020). Section 2.7 shows different ML applications on different water bodies worldwide with different results. Researchers have no consensus on a single best model for WQ parameter prediction, such as Chl-a. The lack of consistency is because Chl-a's prediction depends on geography, waterbody type, climate, and many other variables.

Decision trees, also known as Classification and Regression Trees (CART) algorithms, are supervised ML models widely used for classification and regression problems. One key advantage of CART models is their ability to model nonlinear relationships. Such ability is why CART models are trending in modelling WQ parameters dynamics. Castrillo & García (2020) demonstrated the feasibility of RF in developing a soft sensor for nutrients and proved that RF could model the complex nonlinear relationship between WQ parameters. RF has a superior ability to extract relationships from limited data compared to other ML techniques. The authors also highlighted the applicability of RF in continuous online WQ monitoring using low-cost IoT sensors. Furthermore, RF is immune to outliers and has greater interpretability, making it easy to understand complex relationships and aid decision-making (Yajima & Derot, 2018; Castrillo & García, 2020).

Li et al. (2018) also showed the superiority of RF models (part of CART) over SVR in predicting Chl-a. They concluded that RF has better generalization ability than SVR. The authors also highlighted the benefits of RF, such as resistance to overfitting, working well with limited dataset and input parameters, low computational complexity, and the inherent ability to rank the importance of input variables.

On the other hand, SVR requires complex feature selection using different techniques before training (Jimeno-Sáez et al., 2020). Feature selection adds to the pre-processing steps, increasing the time to develop and train models and the computational complexity. This claim is also supported by Su et al. (2015), who mentioned that SVR is sensitive to input features. Therefore, feature selection is a mandatory step before training SVRs.

Another CART model is called the Extreme Gradient Boosting (XGBoost). The XGBoost and SVR outperformed the LSTM networks in predicting Chl-a concentration when optimum input variables were not selected (Shin et al., 2020). The LSTM, a Recurrent Neural Networks (RNN) model, is better only when crucial input variables are selected (Shin et al., 2020). LSTM's key advantage is its ability to capture the dataset's temporal property, making it appropriate for time series forecasting. However, as Shin et al. (2020) reported, LSTMs are prone to overfitting and require considerable data for training.

Like LSTMs, the ANN also require large datasets to model nonlinear parameters. ANN is popular for Chl-a prediction mainly because of its ability to capture the nonlinear relation of Chl-a with WQ parameters reasonably. Hafeez et al. (2019) showed ANN outperformed RF and SVR in retrieving Chl-a concentration value from reflectance and satellite data and stated ANNs ability to grasp nonlinear Chl-a dynamics. Keller et al. (2018) also found ANN's superior ability to retrieve Chl-a concentration values from hyperspectral data. However, big datasets are scarce in the WQ domain, making training ANNs difficult (Shin et al., 2020).

Besides data scarcity, ANNs have some significant disadvantages. Mamun et al. (2020) pointed out that ANN, as a black-box algorithm, is usually difficult to tune the hyperparameters. The inner mechanisms of the ANN algorithm are poorly understood and thus cannot interpret the outcomes, which can stall management decision-making. Nevertheless, the black-box nature of ANN is still valuable in modelling problems without domain-specific knowledge, e.g. geological information (Lee et al., 2016). Jimeno-Sáez et al. (2020) also mentioned that the training of ANN requires significant computational power and Yajima & Derot (2018) pointed out that ANNs are prone to overfitting.

The following section reviews relevant ML models, including ANN, RF, XGBoost, and a new CART model named Light Gradient Boosting Machine (LightGBM) at greater depth. The literature review regarded these models highly for nonlinear parameters predictions supporting Chl-a soft sensor development.

2.8.1 Random Forest (RF)

Random Forest (RF), also known as Random Decision Forests, invented by Ho (1995), improves the generalization ability of Decision Tree (DT) models. RF algorithm is an ensemble learning technique combining multiple DT models to perform the classification or regression task. This ensemble technique solves the overfitting problem that plagues conventional DT models by averaging the outputs of many DT models (Castrillo & García, 2020).

As the name suggests, RF is comprised of multiple DTs. Each tree is trained on a subsample of data and is called a base learner. The subsample is selected at random with replacement from the original dataset. The outputs from the multiple DTs are then aggregated to derive the output, as depicted in Figure 2.2. This process of randomly selecting a subset of data, training multiple trees, and aggregating their outputs is called Bagging, short for Bootstrap Aggregating (see Figure 2.3). Different aggregation methods such as majority voting or averaging are chosen based on whether the ML task is classification or regression.

Trees in RF are split based on one predictor at any given point, analogous to human decision-making. RF is also not sensitive to outliers and noise in the data, and scaling the input dataset is not mandatory. These attributes of RF make it very suitable for handling the nonlinear and dynamic nature of water quality data (Castrillo & García, 2020). Training individual trees by bagging can lead to correlated decision trees, impairing performance. RF solves this problem using feature randomization, which trains each tree based on a subset of predictors. Feature randomization creates independent DT models and improves performance (Shin et al., 2020).

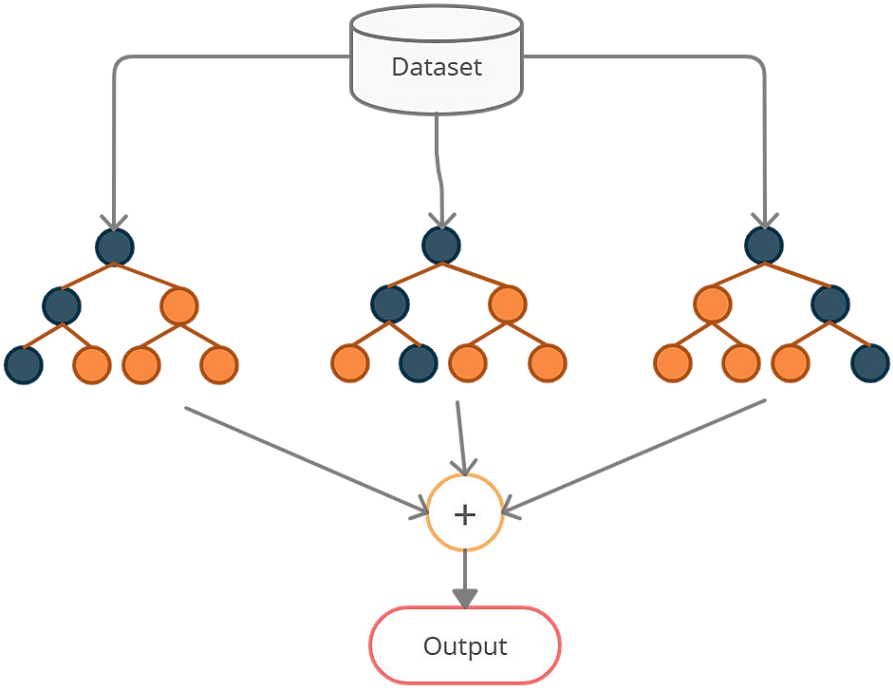

Figure 2.2: Random Forest algorithm showing different trees are trained on a bootstrapped data sample, and the individual DTs are aggregated for the output

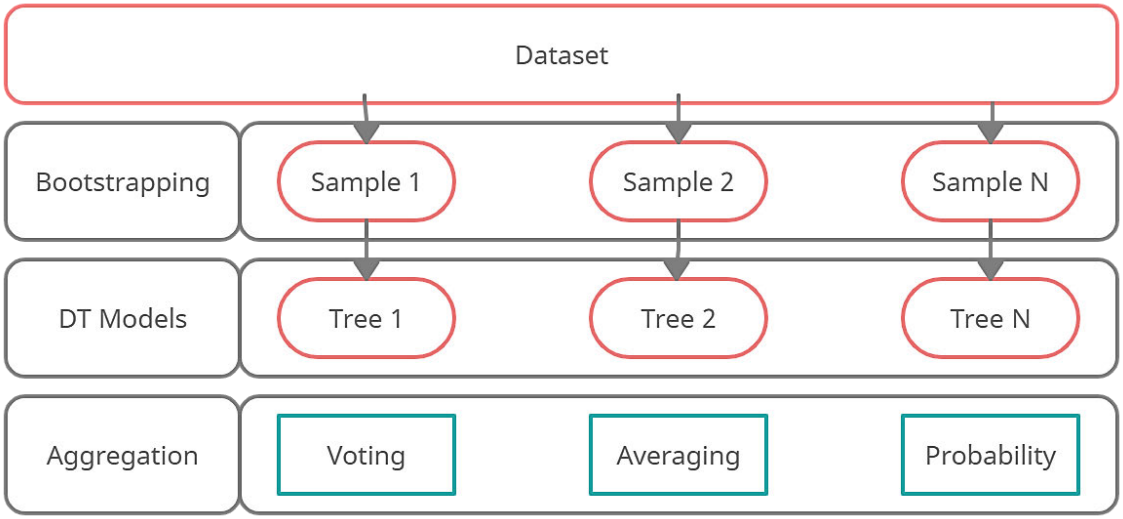

Figure 2.3: Diagram depicting the entire Bootstrap Aggregating (Bagging) process

Key Hyperparameters of RF that determine the performance are:

1. **n_estimators:** determines the total number of trees in the RF model.
2. **max_features:** determines the total number of features used in each node of the DT to make a split

3. **max_depth:** limits the tree's growth by determining the maximum number of levels (or depth) the tree can grow.
4. **Min_samples_split:** determines the minimum number of data samples required in a node before splitting
5. **Min_samples_leaf:** this determines the minimum number of samples required in a leaf node after a split.

2.8.2 Extreme Gradient Boosting (XGBoost)

Like RF, the XGBoost is also an ensemble learning technique proposed by Chen & Guestrin (2016) and is one of the most popular algorithms in ML competitions due to its performance and high scalability. XGBoost belongs to ML algorithms' Gradient Boosting Decision Tree (GBDT) family. The XGBoost is considered an advanced implementation of the GBDT algorithm. Unlike RF, the base learners in XGBoost are trained sequentially using the gradient boosting technique, also called boosting. In boosting the base, learners are trained sequentially where the current tree learns from the mistakes of the previous trees. The loss function is optimized based on the previous model's residuals (i.e. misclassifications).

Equation (2) describes the squared error loss function (without the regularization term) for the regression task, where L is the loss function, y_i is the actual value and p_i the predicted value. Total loss is found by summing the squared differences between actual and predicted values.

$$\sum_{i=1}^{n} L(y_i, p_i) = \frac{1}{2}(y_i - p_i)^2 \qquad (2)$$

The outputs of different models are then aggregated to compute the final output, as seen in Figure 2.4 (Shin et al., 2020).

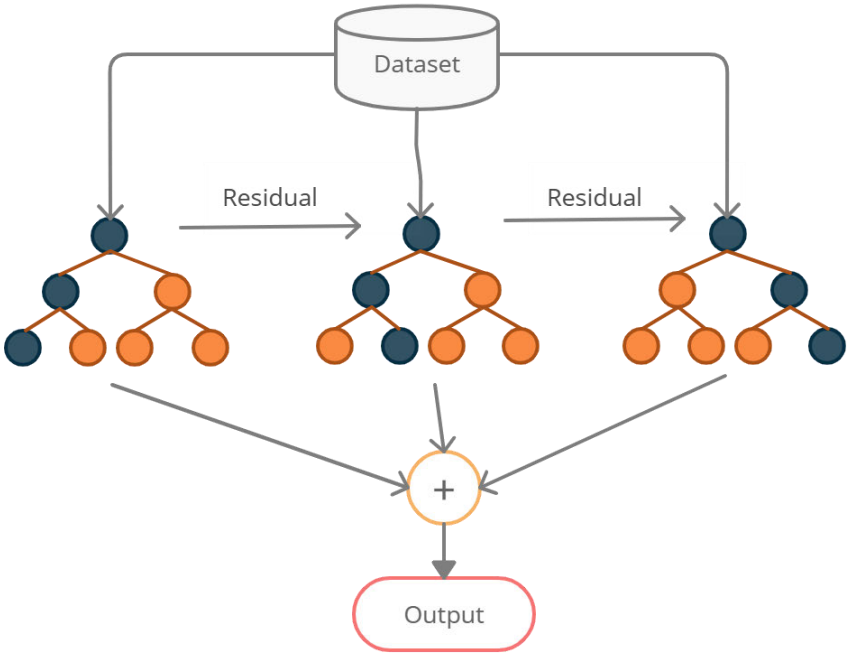

Figure 2.4: Working mechanism of XGBoost Algorithm

The built-in features that make XGBoost a robust algorithm include:

1. Regularization. This feature prevents overfitting by penalizing complex models using L1 and L2 regularization.
2. Handling sparse and missing data. XGBoost can handle sparse data by identifying different sparsity patterns in the dataset and has a built-in feature to deal with missing values.
3. Cross-validation. XGBoost has a built-in feature to compute the cross-validation at every step to determine the optimum number of boosted trees in every iteration.
4. Parallelization and efficient memory usage. XGBoost's architecture allows parallelization and efficient memory use, making training fast.

Key hyperparameters of XGBoost include:

1. **Min_child_weight:** determines the minimum sum of weights needed in a child leaf and controls overfitting.

2. **Max_depth:** limits the tree's growth by determining the maximum number of levels (or depth) the tree can grow.
3. **Eta:** is the learning rate and determines the decay rate of the weights on every step.
4. **Alpha:** is the L1 regularization term applied on leaf weights
5. **Lambda:** is the L2 regularization term applied on leaf weights
6. **Gamma:** a regularization parameter that determines the minimum value of loss that should be exceeded for a split to take place.

2.8.3 Light Gradient Boosting Machine (LightGBM)

LightGBM, like XGBoost, also belongs to the GBDT family of ML techniques. The technique was invented by Ke et al. (2017) at Microsoft. LightGBM inherits most of XGBoost's advantages, including efficient memory usage and parallelization, sparse data handling, and regularization, but is much faster and more efficient than XGBoost. LightGBM uses a highly optimized histogram-based approach to determine the splits, leading to faster training and low memory usage. LightGBM excels with large datasets with many features, while XGBoost takes a longer training time. The architectural difference in how DTs are split in LightGBM and XGBoost determines their performances. In XGBoost, the tree grows level-wise; in LightGBM, the tree is split leaf-wise (see Figure 2.5). Leaf-wise growth results in more significant loss reduction than level-wise because leaf-wise only select leaf to grow, resulting in a maximum reduction in loss.

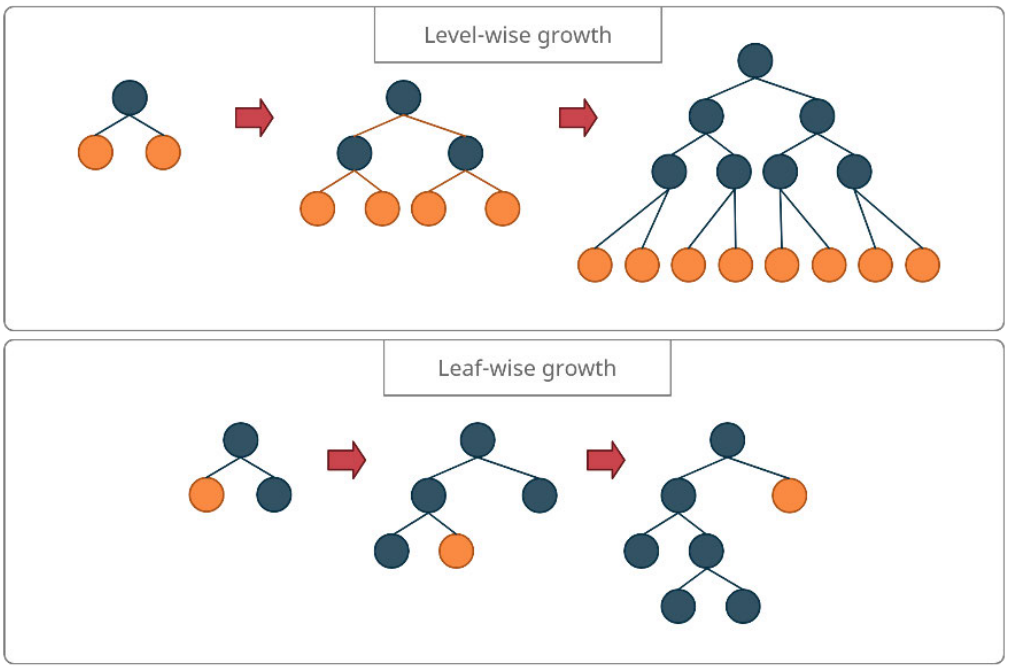

Figure 2.5: Difference between Level-wise growth and Leaf-wise growth

Besides the difference in the construction of new trees in LightGBM, two other novel techniques, namely Gradient-Based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB), give LightGBM its high speed and accuracy (Ke et al., 2017). GOSS is a sampling technique that keeps data instances contributing more to the information gain. For example, data instances with small gradients indicate they are well-trained with minor training errors and hence do not contribute much to the information gain. GOSS randomly excludes these data instances with small gradients but keeps data instances with large gradients (i.e. significant training errors). GOSS can produce more accurate information, gain estimates, and improve accuracy.

EFB is a dimensionality reduction technique that bundles mutually exclusive features to reduce dimensionality without losing important information. Since LightGBM is a histogram-based algorithm, EFB's dimensionality reduction reduces the computational complexity, increasing speed without compromising accuracy. This feature also shields LightGBM from the curse of dimensionality.

Some core hyperparameters for LightGBM include:

1. **Max_depth**: controls overfitting by limiting the depth to which the tree can grow.
2. **Num_leaves**: Determines the number of leaves allowed in a single tree.
3. **Max_bin**: determines the maximum number of histogram bins to contain the features.
4. **Min_data_in_leaf**: determines the minimum data samples needed in a leaf for a split.

2.8.4 Machine Learning Model Performance Evaluation Metrics

Performance metrics such as coefficient of determination (R^2), Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are widespread in Chl-a prediction literature. Authors including Hafeez et al. (2019), Jimeno-Sáez et al. (2020), Castrillo & García (2020) and Mamun et al. (2020) carried out a comparative analysis between different ML models using the performance metrics to find the best performing model.

R^2 is a numerical value between 0 to 1 that determines how well the ML model predicts the actual outcomes. Specifically, R^2 represents to what extent the predictors can explain the variances in the predicted value. Zero (0) means no predictive power, while one (1) means the model fits perfectly and correctly predicts all outcomes. The sum of the square of residuals (SSR) and the total sum of squares (SST) must be calculated to find R^2. SSR indicates the deviations of the predicted values from the actual values. Equation (3) describes SSR where N is the total number of samples, y_j is the actual value and $\hat{y}_j$ the predicted value. Smaller SSR means the model has a good fit.

$$SSR = \sum_{j=1}^{N} \left(y_j - \hat{y}_j\right)^2 \quad (3)$$

SST indicates the squared differences between the actual value and its mean. Equation (4) describes the SST formula where N is the total number of samples, y_j is the actual value and $\overline{y}$ the mean of the actual values.

$$SST = \sum_{j=1}^{N} (y_j - \overline{y})^2 \tag{4}$$

Finally, the R^2 can be calculated using the formula in Equation (5):

$$R^2 = 1 - \frac{SSR}{SST} \tag{5}$$

MAE is a single numeric value that measures the difference between prediction and the actual value. A low MAE value means the prediction has very little error and is close to the actual value. However, MAE is blind to the direction of error, i.e., it cannot detect over or underprediction. MAE is represented in Equation (6), where N is the total number of samples, y_j is the actual value and $\hat{y}_j$ the predicted value.

$$MAE = \frac{1}{N} \sum_{j=1}^{N} |y_j - \hat{y}_j| \tag{6}$$

RMSE is also a numeric value measuring the error between prediction and actual value, but the errors are squared. The squaring of errors penalizes significant errors more than the minor errors. Hence this metric is sometimes used to train the ML model. RMSE, like MAE, is also blind to the direction of the error. Equation (7) describes the RMSE formula where N is the total number of samples, y_j is the actual value and $\hat{y}_j$ the predicted value.

$$RMSE = \sqrt{\left(\frac{1}{N} \sum_{j=1}^{N} (y_j - \hat{y}_j)^2 \right)} \tag{7}$$

2.9 Internet of Things and Data Protocols

The Internet of Things or IoT refers to everyday objects connected to the internet. IoT allows connected things to communicate to execute tasks efficiently and intelligently. These everyday objects or things are usually connected to the internet via microcontrollers or microprocessors responsible for sending and receiving data through various communication protocols. IoT shapes the technological landscape by generating big data and making intelligence more ubiquitous and accessible (Xia et al., 2012). US National Intelligence Council included IoT technology in their "Disruptive Civil Technologies" list and concluded that by 2025, IoT would be incorporated into our everyday objects like furniture and packages (Atzori et al., 2010).

IoT shapes technologies via identification, tracking, wireless sensors, and distributed edge intelligence, which significantly impacts transportation, healthcare, smart homes, and intelligent industries (Atzori et al., 2010). The byproduct of IoT is big data. The WQ domain is an area that can hugely benefit from big data. Edge computing and sensors that enable big data for WQ monitoring can impact water management. China has already deployed Wireless Sensor Networks (WSNs) and robots equipped with WQ monitoring sensors for effective water conservation management (Liu et al., 2019).

IoT is not a single technology but rather an orchestration of different technologies to carry out a task. The fundamental building blocks of IoT are hardware, communication infrastructure and cloud computing.

Hardware is the 'thing' connected to the internet and is typically an embedded system with integrated sensors and actuators. The embedded system can be microcontroller-based or microprocessor-based (depending on the type of application). Microcontrollers

are more commonly used for sensors, and it converts the electrical signals from the sensor to the parameter being measured using calibration values. The microcontroller is also responsible for sending the data to the cloud using standard protocols like Hypertext Transfer Protocol (HTTP) or Message Queuing Telemetry Transport (MQTT). Microcontrollers are programmed to carry out tasks, and the software written for microcontrollers is known as firmware. The firmware is uploaded to the microcontroller's memory, where the CPU reads and executes the instructions. The firmware can be written in many different languages, including but not limited to Assembly, C, C++, and Python. C and C++ are widely popular in firmware programming.

An IoT device can connect to other IoT Devices (thing-to-thing), gateways (middleman between thing & cloud), or directly to cloud computers. The IoT communication infrastructure begins by choosing connectivity options based on the type of application and the location. Many internet options include Ethernet Cable (Wired Connection), WiFi, and cellular networks (2G/3G/4G, NB-IoT, CAT-M1, and others). Devices can be indirectly connected to the internet via a gateway, and popular options for connecting to the gateway include Bluetooth (for shorter distances), LoRa and SigFox (for longer distances).

Cloud computing is the on-demand availability of computing resources over the internet. Cloud computing is a collection of powerful servers configured to deliver services ranging from computing instances, managed databases, AI-inferencing engines, load balancing, automatic scaling of resources, and many others. A typical cloud computing architecture for a web application is displayed in Figure 2.6, where T3s and M5s are computing instances, P and S are relational database services, Elastic Load Balancing is load balancers, and CloudFront is the content delivery network (CDN).

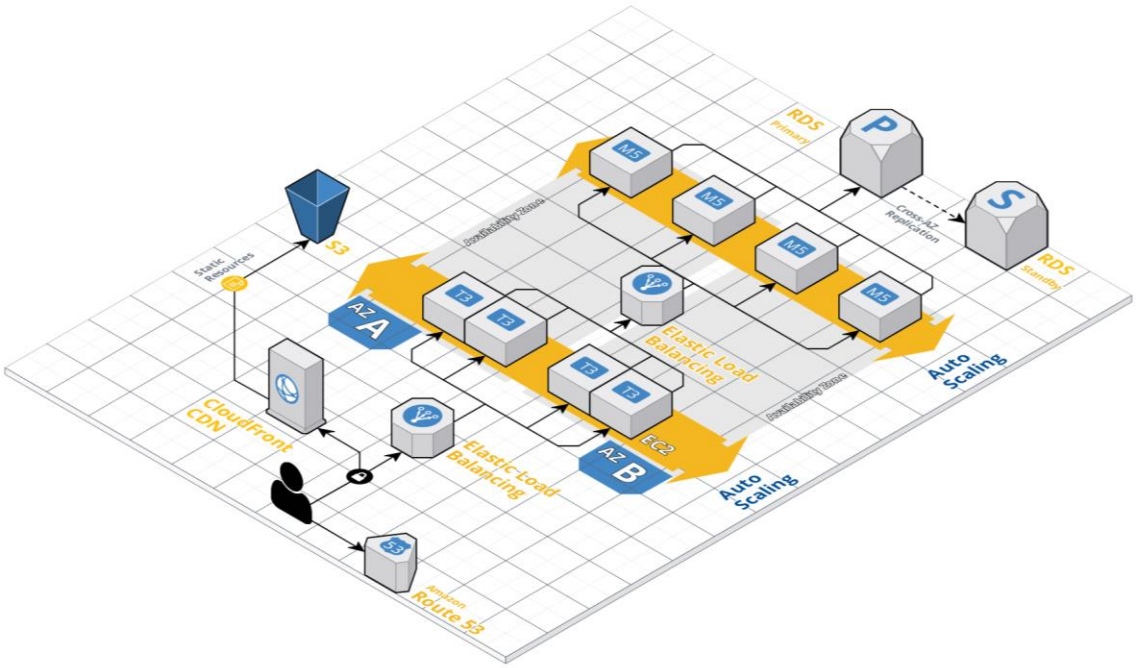

Figure 2.6: Cloud Computing Architecture for a typical web application

IoT devices use specific protocols to communicate with the cloud's IoT service. The most common IoT protocol is MQTT. MQTT is a lightweight protocol explicitly designed to overcome challenges faced by WSNs in a traditional protocol like HTTP. MQTT is designed for resource- and bandwidth-constrained applications and vastly simplifies the integration of distributed WSNs (Hunkeler et al., 2008). MQTT is a data-centric protocol with publish-subscribe architecture essential to streamline the generation of vast amounts of data from distributed sensors.

The MQTT protocol allows bidirectional communication between the hardware device (the thing) and the cloud. Bidirectional communication allows device-to-cloud, device-to-device, and cloud-to-cloud communication to be carried out seamlessly. The publish-subscribe architecture decouples the publisher (e.g. the hardware device) and subscriber (the cloud) by placing a middleman between the communication known as the broker, as seen in Figure 2.7.

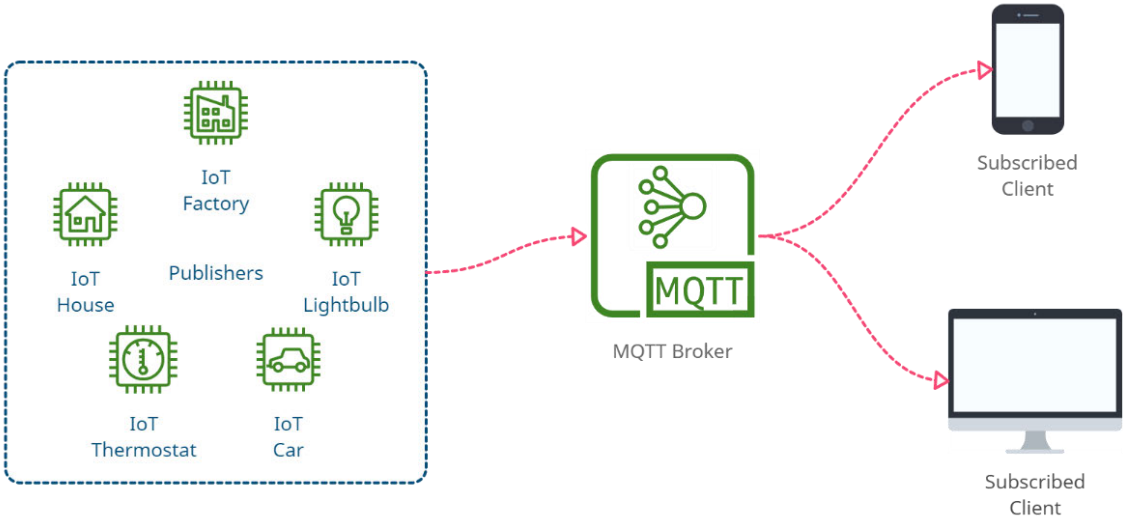

Figure 2.7: IoT devices communicating with the subscribed clients via the MQTT broker

A device can be either a publisher or a subscriber. A device needs to publish to a topic to send data, while a device needs to subscribe to a topic to receive data. A topic can have any name, e.g., “temperature”. If the subscriber subscribes to the “temperature” topic, it will receive data when the publisher publishes the “temperature” topic. This entire process is summarized in Figure 2.8.

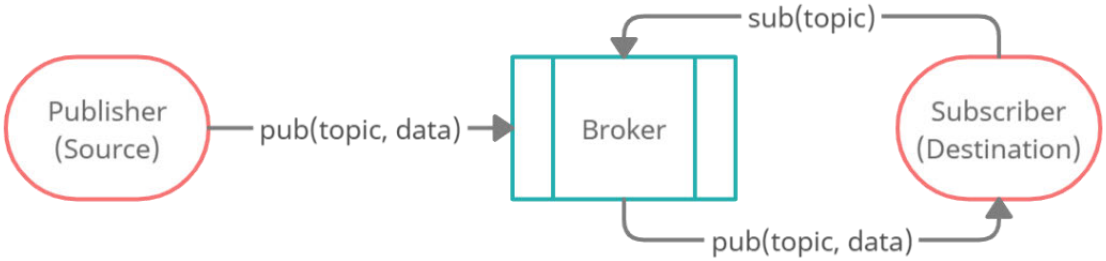

Figure 2.8: Publish-subscribe architecture of MQTT protocol

Considering the example of a device publishing temperature data while the web-application dashboard is the subscriber, the web application subscribes to the same topic the device publishes. Consider the device publishing the topic “readings” in this example. The dashboard in the Cloud that displays the data receives the data by subscribing to the same topic, “readings”, as shown in Figure 2.9.

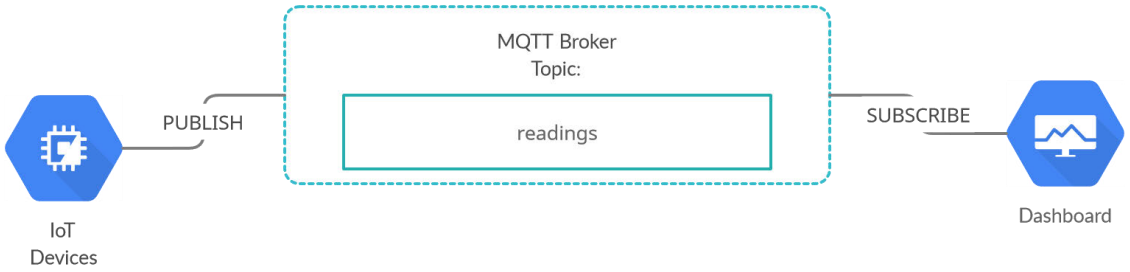

Figure 2.9: IoT devices publishing data on “readings” topic. The dashboard web application is subscribed to the “readings” topic to receive the data.

Similarly, in the case of the Cloud communicating with IoT devices through an IoT service called IoT Core, the cloud must publish this data to a topic. In this example, the cloud is publishing the topic “control”. The IoT device receives the command data by subscribing to the same topic, “control”, as shown in Figure 2.10.

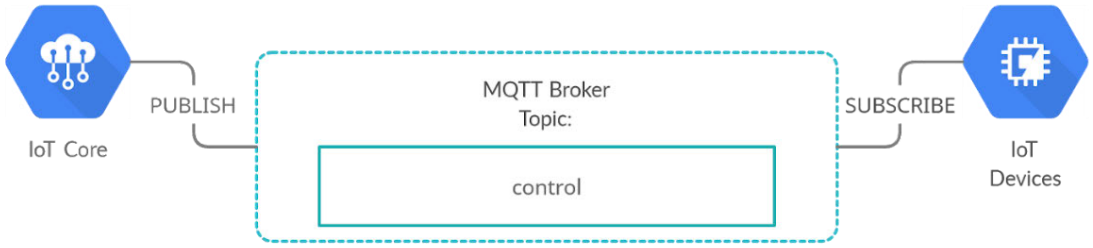

Figure 2.2: Cloud sending data to IoT devices

HTTP is the most popular protocol for exchanging data on the web in HTML documents and is the backbone of the modern-day Web. Despite its popularity, HTTP has several limitations if used as the primary protocol for WSNs. The first limitation is that HTTP is a synchronous protocol where the client requests and waits for the server to respond. In high-traffic applications like IoT, HTTP becomes unreliable, especially when a vast amount of data is exchanged in an unreliable network. IoT device data are generally generated and transported asynchronously, making MQTT, an asynchronous protocol, more reliable.

The second limitation is the design of the HTTP protocol. HTTP is a one-way and one-to-one communication protocol, so the cloud or the server cannot send data to the devices unless the devices request it. In IoT applications, it is common for the cloud to send control commands after running analytics on the IoT data, e.g., turning on the alarm in the event of a fire. Responding to such an event in real-time with HTTP is inefficient and cumbersome to implement.

2.10 Concluding remark

The literature covers the global concern on the prevalent eutrophication, consequently HABs that can disrupt the ecosystem of water bodies. Computer scientists propose several solutions to support the water community in WQ monitoring. The most recent contribution is the development of Chl-a soft sensors to predict Chl-a concentration levels based on ML inductive modelling. There is also an exploration of IoT-enabled extensions to WQ-related soft sensors. However, there is yet work that considers Chl-a soft sensor.

This chapter has reviewed articles between 2013 and 2020 and describes various ML models for inductive modelling. Table 2.4 shows a summary of some related works. Three gaps have been identified in the literature. The first is that the ML models are trained with a minimum of seven WQ parameters. Second, their WQ parameter dataset is not tested with online inferencing architecture, even though they argue that real-time Chl-a monitoring is the way forward for Chl-a soft sensor. Lastly, although DT models are favoured, none of the researchers tested LightGBM in the Chl-a concentration estimation, a new and improved DT model. Table 2.4 summarizes the gaps in the Chl-a soft sensor literature.

Table 2.4: Chlorophyll-a concentration prediction via inductive Machine Learning

References	ML Algo	WQ	Aim	Gaps
Huo et al. (2013)	ANN	10	TN, SD, DO & Chl-a Estimation in lakes	a) Requires seven or more WQ parameters as inputs to the ML model b) Offline inductive ML used in Chl-a concentration estimation/prediction. c) LightGBM model not explored in any of the studies
Lee et al. (2016)		15	Future Chl-a concentration prediction in lakes	
Tian et al. (2017)		7	Prediction of change in Chl-a dynamics in reservoirs	
Li et al. (2018)	RF	7	Chl-a estimation in lakes	
Yajima & Derot (2018)		33	Future Chl-a concentration prediction in lakes	
Castrillo & Garcia (2020)		20	Nutrient parameters estimation in rivers	
Su et al. (2015)	SVR	7	Future Chl-a concentration prediction in lakes	
Nieto et al. (2019)		15	Predicts algal blooms in lakes by estimating TP & Chl-a	
Mamun et al. (2020)		10	Chl-a & SD estimation in reservoirs	
Jimeno-Sáez et al. (2020)		9	Chl-a estimation in Coastal Waters	

2.11 Chapter Summary

The insights learned from this chapter shapes the methodology proposed in this study. The following chapter describes the methodology that addresses the gaps in the Chl-a soft sensor literature.

CHAPTER 3: METHODOLOGY

3.1 Overview

This chapter is divided into data acquisition, data preprocessing, ML model development, and IoT application development. The data acquisition describes the relevant WQ parameters and how the data is measured in the water body. The data preprocessing explain the strategies used to handle missing data and outlier and the transformation required before using the data to train ML models. The section describes how different ML learning models are developed and trained and their relevant performance metrics. Finally, the IoT application development discusses the cloud computing architectures and services used to realize ML real-time inferencing using IoT data.

3.2 Data Acquisition

The literature review revealed different correlations between Chl-a and other WQ parameters. Huo et al. (2013) and Jimeno-Sáez et al. (2020) found that Chl-a can be predicted from the DO, temperature, turbidity, nutrients, and month. In another study, Castrillo & García (2020) found that nutrients are correlated with DO, temperature, conductivity and turbidity. Figure 3.1 shows a correlation map for these findings. Following Figure 3.1, the data acquisition is designed to collect four WQ parameters, i.e., DO, temperature, conductivity and turbidity. All four WQ parameters can determine the nutrient level at the lake. The nutrient level and the four WQ parameters can combine to determine the Chl-a concentration. The Chl-a parameter is also required for ground truth.

Additionally, timestamped recordings of WQ data are acquired since the algal growth rate varies depending on the time of the day. The data was collected from 20th February 2019 to 9th October 2019 from Tasik Taman Aman lake in Section 22, Petaling Jaya, Selangor, Malaysia. The satellite view of the lake is shown in Figure 3.2.

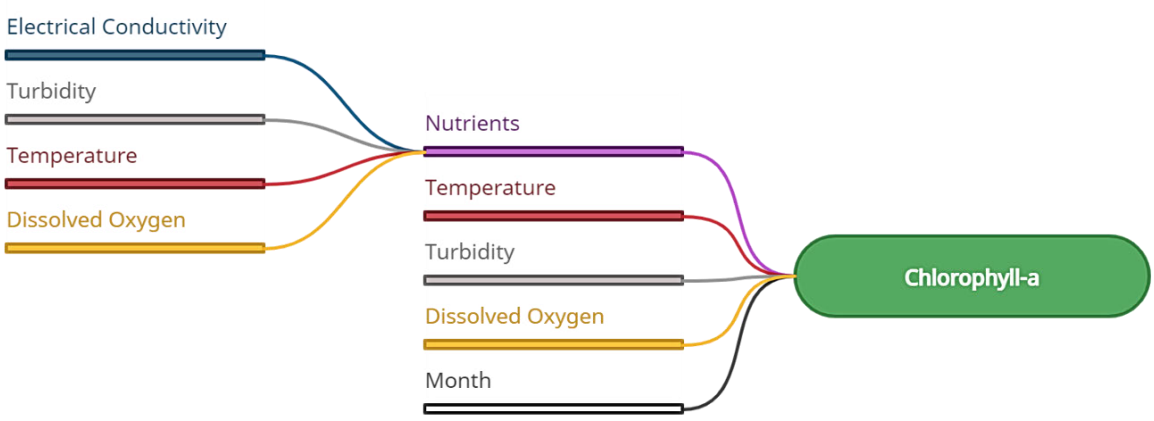

Figure 3.1: Correlation map of Chlorophyll-a to WQ parameters and time

Figure 3.2: Google earth view of Tasik Taman Aman lake

A gap occurred between April 05th, 2019 and September 12th, 2019, as no data collection was done during that timeframe. A multiparameter WQ sonde was used for data acquisition. The In-Situ company's Aqua Troll 600 Multiparameter Sonde collects DO, temperature, conductivity, turbidity, TDS and Chl-a every 10-minute within the monitoring period. A platform at the lake was used to deploy the multiparameter sonde. Figure 3.3. shows the platform setup and multiparameter sonde used.

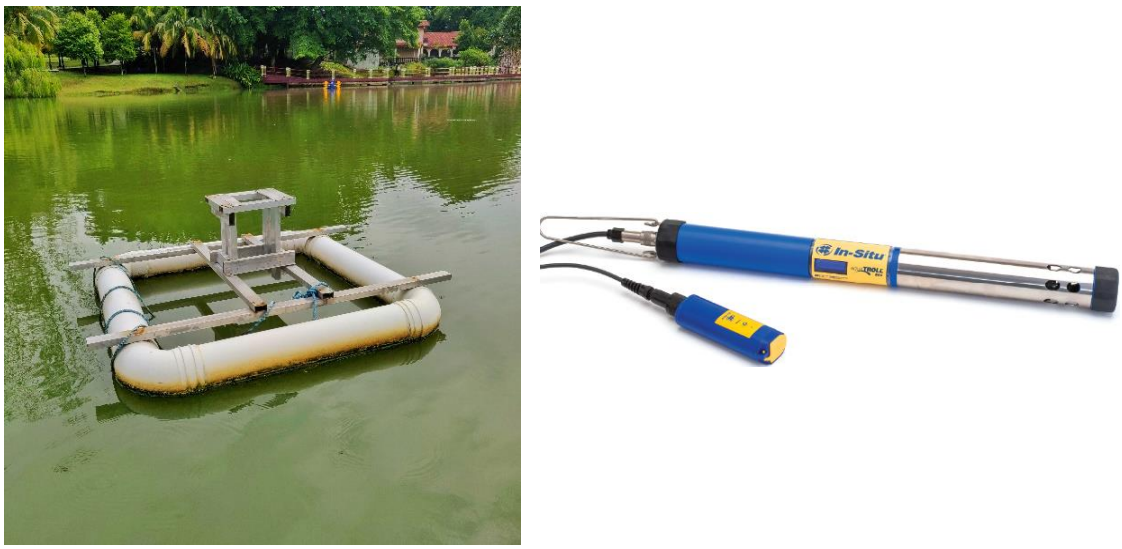

Figure 3.3: Platform for sonde deployment at Tasik Taman Aman lake on the left and In-Situ Aqua Troll 600 Multiparameter Sonde on the right

3.3 Data Preprocessing

The dataset collected was first plotted to check the distribution and correlation between parameters using a pair plot from the seaborn library of Python. Figure 3.4 shows that the conductivity (labelled as 'cond') is highly correlated with TDS (labelled as 'tds'). The reason is that the TDS value is derived from the conductivity sensor. Therefore, the TDS parameter was dropped to avoid redundancy.

Table 3.1 has the descriptive statistics to understand the range of values for each feature and their mean, standard deviation (Std. Dev.), variance, skewness, and kurtosis. Understanding the dataset and its descriptive statistics helps determine the preprocessing steps required before developing and training an ML model. The dataset obtained from the sonde requires preprocessing before training, evaluating ML models, and making predictions. The steps in data preprocessing are handling the missing values, outliers removal, data transformation, and finally, data splitting into training and testing datasets. Figure 3.5 shows the date preprocessing flow adopted in this study.

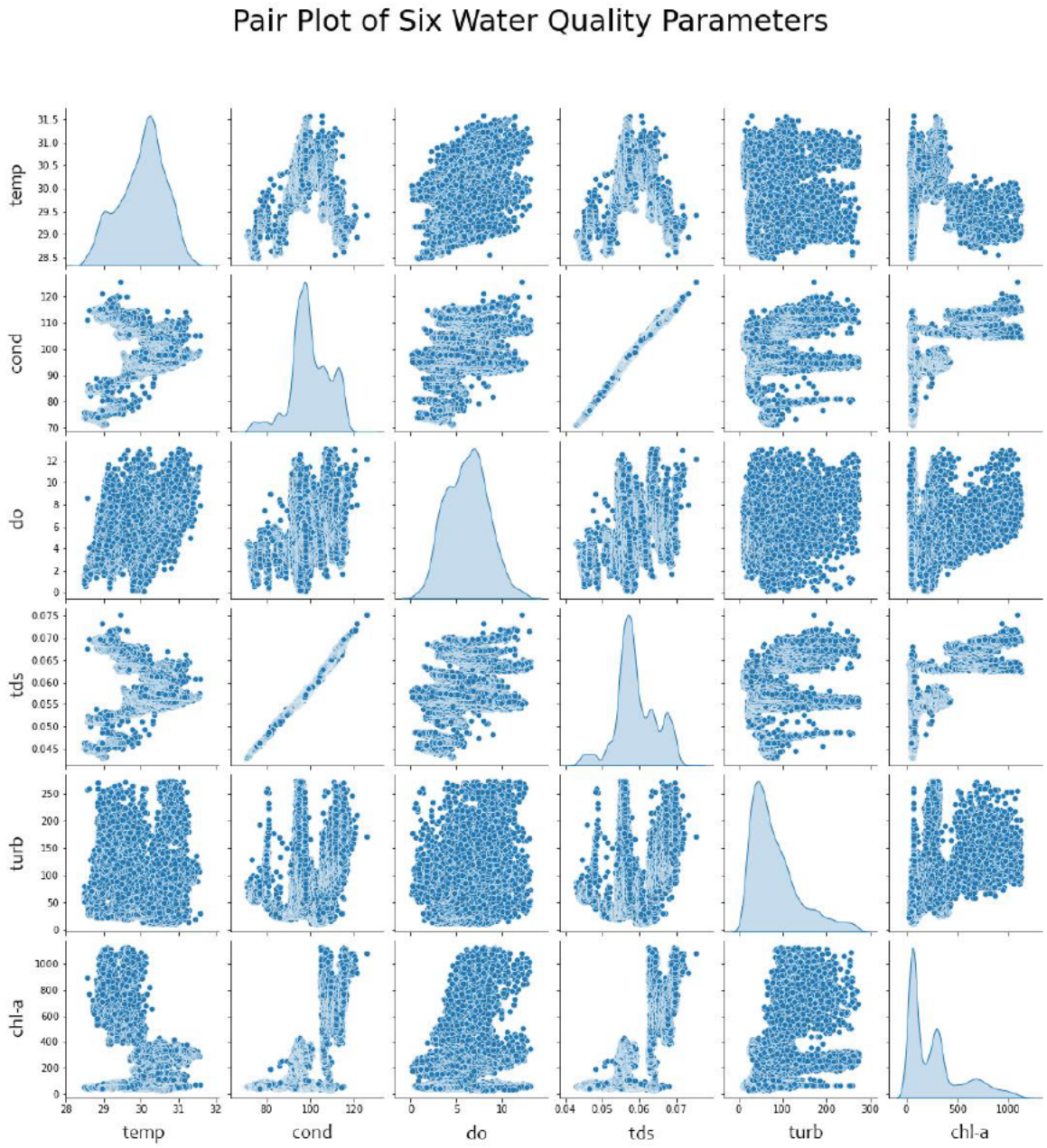

Figure 3.4: Pair plot showing the correlation between different WQ parameters and the curve along the diagonal showing the distribution of the data

Table 3.1: Descriptive statistics of the WQ dataset (from February to October 2019)

Parameter	Min	Max	Mean	Std. Dev.	Variance	Skewness	Kurtosis
Conductivity	71.09	125.58	99.56	8.89	79.05	-0.34	0.38
DO	0.12	13.12	6.14	2.28	5.19	0.12	-0.37
Turbidity	9.63	272.19	82.24	54.43	2962.35	1.28	1.27
Chl-a	24.96	1126.18	276.4	264.21	69804.75	1.22	0.56
Temperature	28.47	31.6	30.03	0.61	0.38	-0.32	-0.52

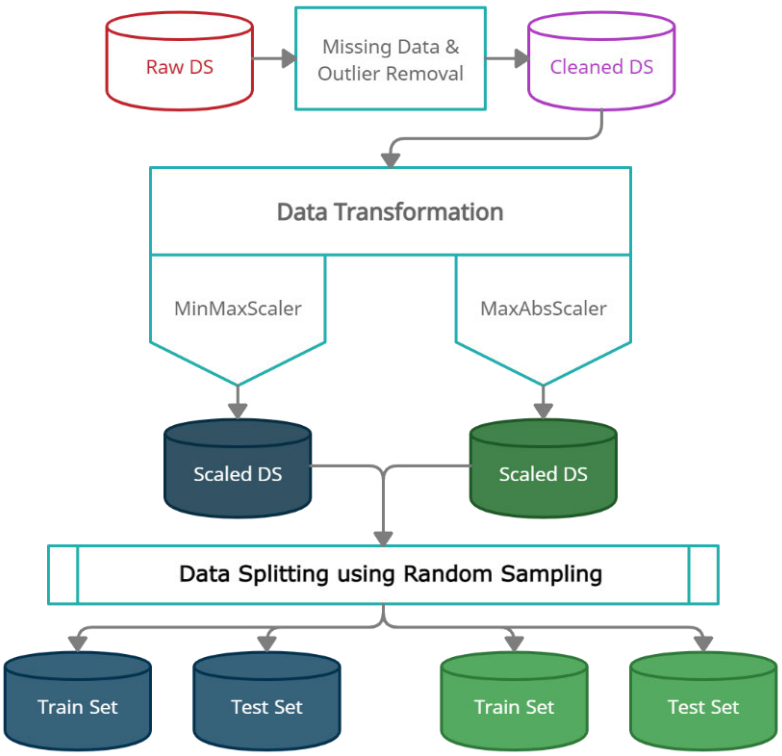

Figure 3.5: Data Preprocessing Overview

3.3.1 Handling Missing Values

The Tasik Taman Aman Lake dataset had missing values for different parameters at different points in time. The number of missing values per parameter is shown in Figure 3.6. Most missing values are from the DO sensor measuring dissolved oxygen concentration. Conductivity, turbidity, and Chl-a sensors also had missing values throughout the dataset, but the temperature sensor did not have any missing data. Since the missing values are small compared to the entire dataset, the rows with missing values are dropped.

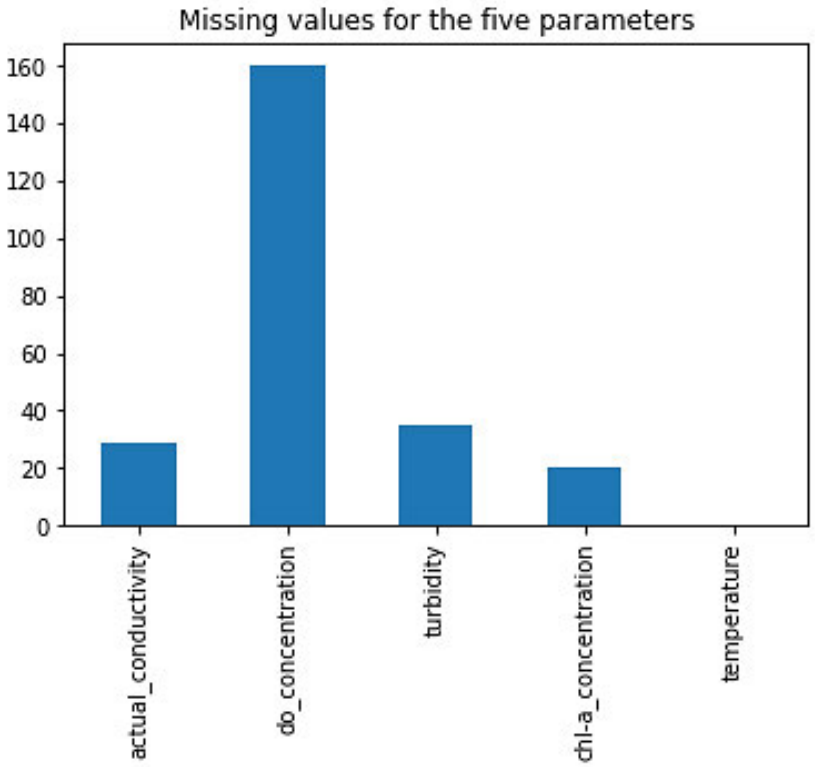

Figure 3.6: Bar chart representing the proportion of missing data per WQ variable

3.3.2 Outlier Removal

The dataset also consisted of some outliers due to systemic errors. The Z-score determines how much a data point deviates from the standard deviation. A simple rule of thumb is to check whether the data point lies within three standard deviations from the mean of the dataset. This rule of thumb is derived from the three-sigma rule, which states that 99.7% of the data falls within three standard deviations of the mean. If the z-score is less than -3 or more than +3, it indicates the data point lies beyond the accepted bound and is considered an outlier. Data points with a Z-score less than -3 or more than +3 were dropped in this study. The formula to calculate the Z-score is given in Equation (8), where Z is the standard score, x is the data point, μ is the mean of the dataset, and σ is the standard deviation of the dataset.

$$Z = \frac{(x - \mu)}{\sigma} \tag{8}$$

3.3.3 Data Transformation

This study used two data transformation techniques: Maximum Absolute Scaling and Min-Max Scaling. The data transformation was implemented using the Scikit-learn library in Python.

The Maximum Absolute Scaler (MaxAbsScaler) scales each feature in the dataset based on its maximum absolute value and sets the maximum value for each feature to 1. Every data point in a single feature transforms based on the maximum value. MaxAbsScaler scales the data to fall between -1 to +1 by dividing every data sample by its maximum absolute value. MaxAbsScaler also preserves the sparsity in the data because it does not alter the original distribution of the data. Since MaxAbsScaler scales data based on the maximum absolute value, it is vulnerable to the presence of outliers, and hence outliers should be removed before scaling the data with MaxAbsScaler. The MaxAbsScaler scales the data samples using the formula shown in Equation (9) where x_{scaled}^{i} is the scaled i^{th} data point, x^{i} is the original i^{th} data point and $max(|x|)$ the maximum absolute value of feature x.

$$x_{scaled}^{i} = \frac{x^{i}}{max(|x|)} \tag{9}$$

The Min-Max Scaler, also known as normalization, scales the data sample such that it lies in the range between 0 and +1. A data sample in a feature is scaled by subtracting the minimum value of that particular feature from the data sample. It is then divided by the difference between that feature's maximum and minimum value, as shown in Equation (10). Like MaxAbsScaler, Min-Max Scaler does not change the original distribution of the dataset and mainly works well for datasets that are not normally distributed.

MinMax Scaler is also susceptible to outliers like MaxAbsScaler. In Equation (10), x_{scaled}^{i} is the scaled i^{th} data point, x^{i} is the original i^{th} data point, x_{min} is the minimum value of feature x and x_{max} is the maximum value of feature x.

$$x_{scaled}^{i} = \frac{x^{i} - x_{min}}{x_{max} - x_{min}} \tag{10}$$

3.3.4 Data Splitting

The WQ dataset was sampled randomly using the pandas DataFrame function '*sample*' to obtain a representative and unbiased testing dataset. This function split the dataset into 80% for training and 20% for testing. A histogram was plotted for all four WQ parameters, DO, temperature, conductivity, turbidity, and Chl-a, to ensure the random split keeps the training and testing data distribution the same. Figure 3.7 shows the testing dataset (orange distribution) compared to the training dataset (blue distribution). The height of the orange testing distribution is smaller because the distribution represents only 20% of the data.

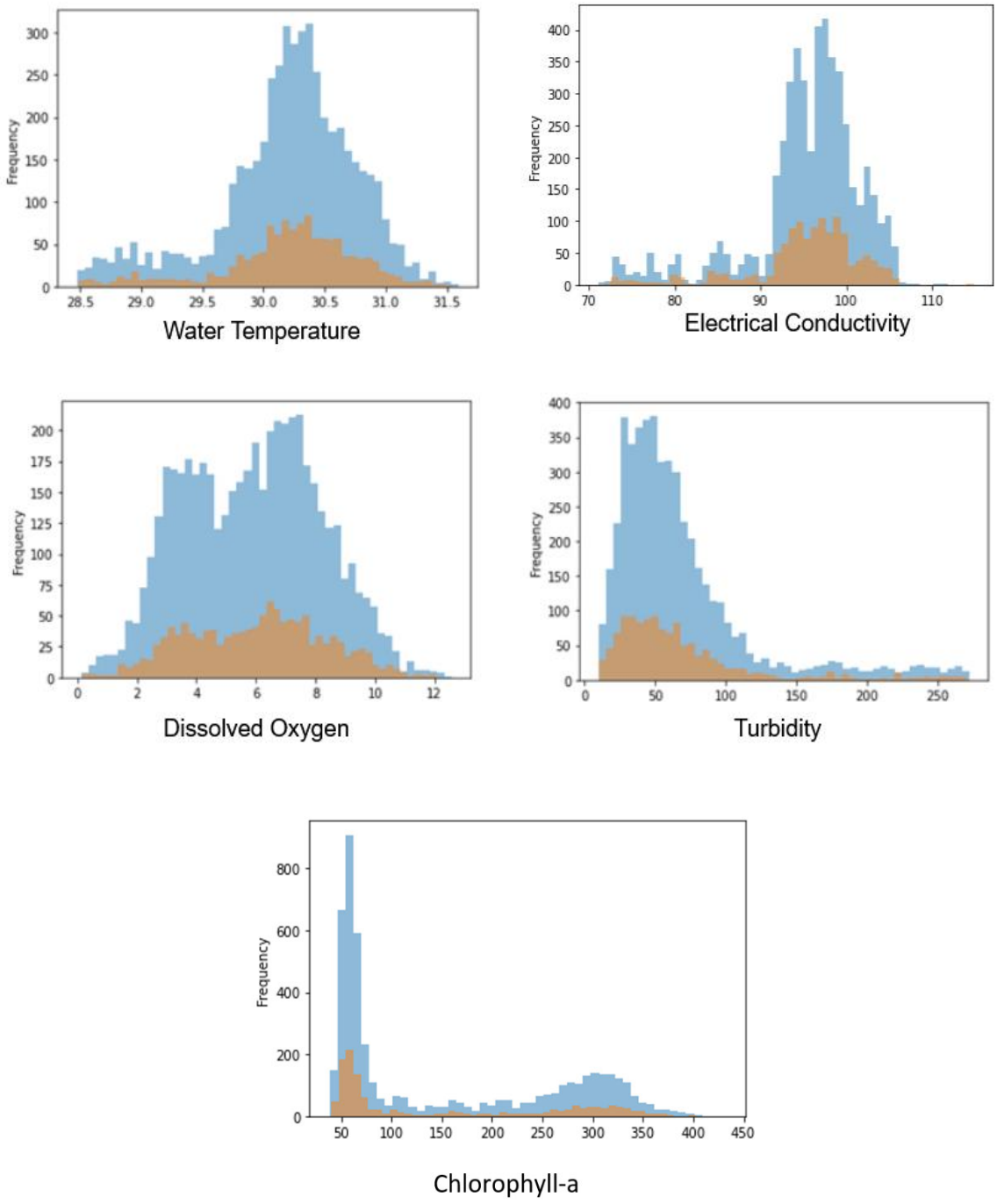

Figure 3.7: Distribution of training set (blue coloured distribution) and testing set (orange coloured distribution) for DO, temperature, conductivity, turbidity and Chlorophyll-a.

3.3.5 Dataset Summary

After removing the missing values and outliers, the total number of data samples was reduced from 8887 to 8516. After splitting the dataset into training and testing datasets following the 80:20 ratio, the training dataset contains 6813 rows, and the testing dataset contains 1703 rows. Table 3.2 shows the data before and after preprocessing.

Table 3.2: WQ data samples before and after preprocessing

Dataset	Sample size (rows)
Raw dataset	8887
After Dropping Missing Values	8727
After Dropping Outliers	8516
Final Cleaned Dataset	8516
Training Dataset (80%)	6813
Testing Dataset (20%)	1703

3.4 ML Model Development, Validation, and Evaluation

The literature review proposed ANN, RF, and XGBoost as favourable ML model candidates for Chl-a soft sensor development. RF and XGBoost belong to the CART family of ML models. However, a new and improved CART model called LightGBM has not been applied in WQ literature. Therefore, LightGBM is included in the ML model development to see its performance. These ML models were trained using the training dataset achieved by splitting the original dataset following Table 3.2.

This study adopted a train-validation-test methodology, and 10-fold cross-validation was used during training to check for overfitting. The 10-fold cross-validation splits the training data set into ten training and validation folds (Figure 3.8). The training and validation step is iterated several times to avoid biases from the split and ensure more consistent results throughout the training phase. The best model was selected based on the results from the cross-validation step. The best model was finally evaluated on an unseen test dataset to assess the overall performance. Figure 3.9 depict the process flow.

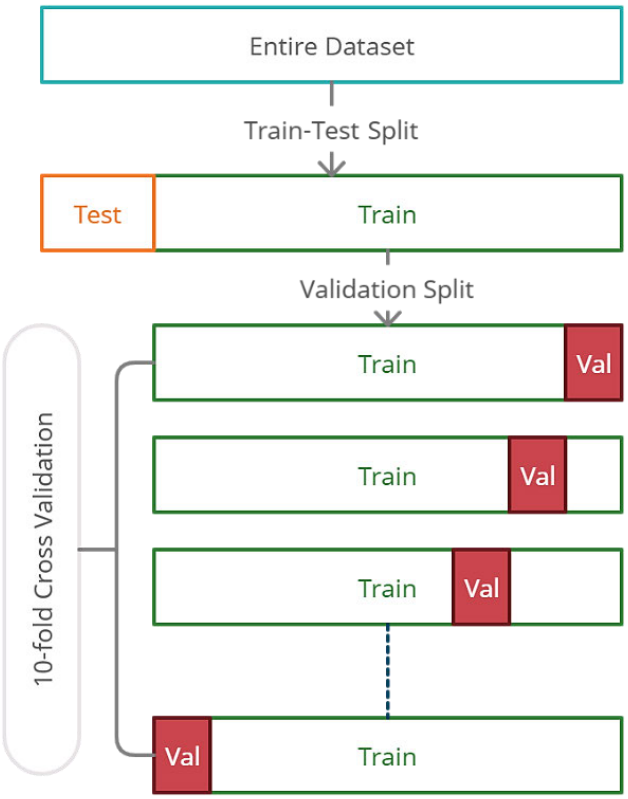

Figure 3.8: The 10-fold cross-validation flow

3.4.1 CART Models Development

The CART models follow the same development procedure, including RF, XGBoost, and LightGBM. Thus, the development of all three CART models is described in this section. Before training the CART models on the training dataset, additional feature engineering was carried out on the timestamp column of the dataset. Every row of data contains the date and time information of when the sample is collected. Ten features were extracted from the timestamp information using the *DateTime* library in Python. The timestamp for each data point was converted into the following features:

1. Year
2. Month
3. Day
4. Day of Week

5. Day of Year
6. Quarter of Year
7. Week of Month
8. Hour
9. Minute
10. Second

The ten DateTime features and four WQ parameters were used as inputs to the ML model to produce the Chl-a concentration output, as shown in Figure 3.10.

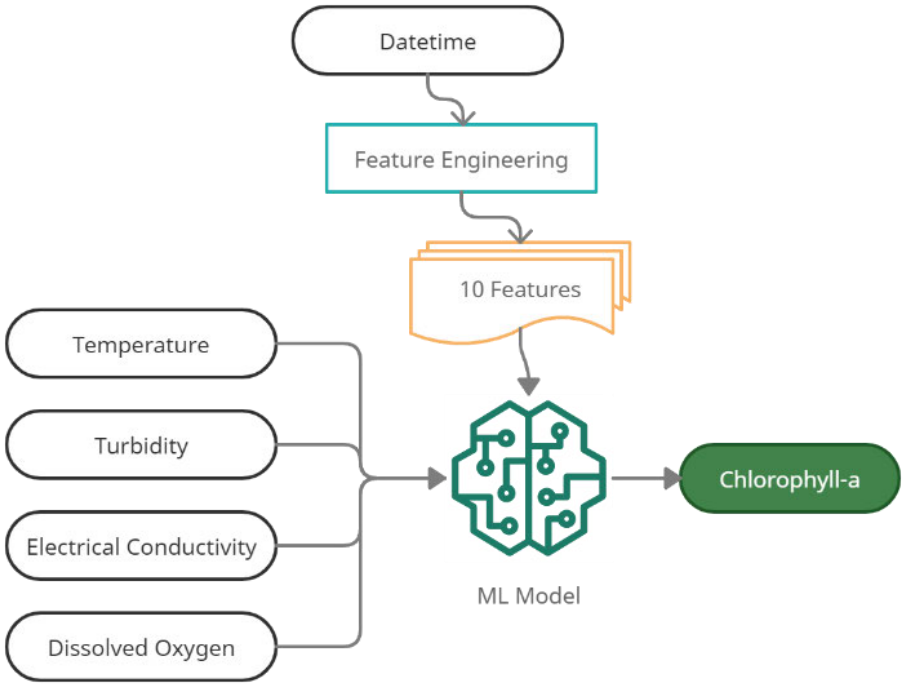

Figure 3.9: Four WQ inputs and ten DateTime features were fed as inputs to the CART model to produce the Chl-a concentration output

Only RF is available through the popular python library Scikit-Learn where RF can be imported. XGBoost and LightGBM require separate installation to be used in the Python Script. After importing the models, RF, XGBoost and LightGBM are instantiated using *RandomForestRegressor*, *XGBRegressor* and *LGBMRegressor*, respectively.

RMSE was selected as the primary metric to be optimized during training. The 10-fold cross-validation was set using the *RepeatedKFold* function, and the cross-validation was set up using the *cross_val_score* function from the Scikit-Learn library. After this step, the model fits the training data with default hyperparameters.

3.4.2 Hyperparameter Optimization for CART Models

Automated Machine Learning (AutoML) is a library used to accelerate ML model development by automating the time-consuming iterative tasks such as training and optimization in the development process. AutoML carries out parameter sweep to find an optimum configuration of hyperparameters to improve the performance and accuracy of ML models. AutoML is available as a service in Microsoft Azure's cloud computing platform, where AutoML can take advantage of the high computational resources available. Due to the simple architecture of CART models, Azure's AutoML service supports RF, XGBoost, LightGBM and several other CART models.

The first step in configuring the AutoML is to specify the type of ML problem, i.e., regression, in this study. The second step is to provide the pre-processed training dataset while specifying the input features (Datetime features, DO, temperature, conductivity and turbidity) and output predictions column (Chl-a concentration value). The third step is to specify the primary metric.

The primary metric is the metric that will be optimized during the training. In this study, *normalized_root_mean_squared_error* was selected as the primary metric for the AutoML experiment. Before starting the AutoML engine, the final step is to specify the exit criteria to stop the experiment. A timeout of 1 hour was selected for this study for the entire experimentation phase. During this phase, AutoML iterates through different

hyperparameter settings based on the primary metric specified and comes up with the best hyperparameter combination for the respective ML model to obtain the best performance.

3.5 IoT Application Development

Soft-sensor development requires combining IoT technology with ML to enable real-time inferencing from the IoT sensor's online data stream. Commercially available cloud computing platforms provide IoT and ML inferencing services that can be configured to develop an IoT-based soft-sensor application. With the services provided by the cloud computing platform, it is possible to develop a serverless solution that eliminates the need to provision and manage servers. Serverless applications also save costs since payment is based on usage only.

3.5.1 IoT Cloud Service Configuration

Amazon Web Services (AWS) is a commercial cloud platform that offers many services, including IoT. The IoT-related services from AWS are the most cost-effective among other competitors while being robust and highly scalable. Four critical AWS services were explored in this study to create the soft-sensor IoT application: the IoT Core, IoT Analytics, DynamoDB and Lambda Function.

IoT Core is a serverless, managed cloud service that acts as a hub for IoT devices. IoT Core can support billions of devices and process trillions of messages from the IoT devices, making it ideal for WSN applications. IoT Core is also responsible for managing the connection and security of the IoT devices and acts as an MQTT broker to exchange messages using MQTT topics. IoT Core can also process messages and forward the MQTT data using a built-in option IoT rule. IoT rule can forward the data in the MQTT messages based on certain conditions configured in the IoT Core settings. The IoT rule

was set up in this study to send data to DynamoDB for storage and forward it to the Lambda function for further processing.

IoT analytics service stores the incoming sensor data from the IoT device in long-term storage. IoT analytics service is used for offline data analytics and experimentation on the long-term IoT data. This service is used to find trends in data and test ML models before deploying the ML model in a real-time inferencing instance. DynamoDB is a managed NoSQL database service engineered to process data from high-traffic sources such as IoT devices or sensors. DynamoDB is also highly scalable and can scale up based on the demand. DynamoDB also offers high-speed read and write functionalities, making it ideal for real-time soft-sensor applications.

Lambda function is a serverless, event-driven computing instance service used to execute a code or a script on demand. The Lambda function can run a code when configured while automatically managing the computing resources and environment. In this study, the Lambda function was used to forward incoming IoT data to the ML model by calling the API of the ML instance hosting the ML model. The Lambda function is triggered by the IoT rule of the IoT core described earlier. Lambda also stores the ML model's predicted value in the DynamoDB table.

3.5.2 Deploying ML for Real-Time Inferencing

Microsoft Azure Cloud Computing platform provides compute instances as a service where ML models can be hosted for real-time inferencing. The advantage of Azure is that it streamlines the development of the ML model using the AutoML deployment. When the model is deployed in the compute instance, the service automatically hosts the model as a web service endpoint. The model can then be accessed for real-time inferencing by

calling the API endpoint. LightGBM was the best performing model in this study, deployed in Azure's ML inferencing instance. The real-time inferencing was done by calling the API endpoint, as shown in Figure 3.11.

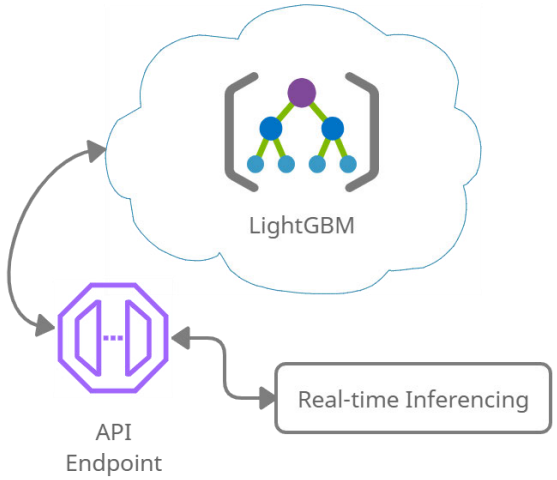

Figure 3.10: Realtime inferencing of hosted LightGBM model using the API endpoint

In summary, the Azure ML inferencing service carries out the following tasks automatically:

- The ML model is containerized, encapsulating all the model's dependencies.
- Load-balancing for the inferencing requests via HTTP
- Security of inferencing requests
- Achieves autoscaling by managing the infrastructure underneath and scaling up or down based on the need.

3.5.3 Soft-Sensor Architecture Overview

A serverless, event-driven IoT architecture was developed by combining the AWS and ML inferencing service from Azure to realize the Chl-a soft sensor. Data from IoT sensors or a WQ dataset file can be pumped into the IoT Core using the MQTT protocol. In this study, the functionality of the architecture was tested using WQ dataset files only. Testing through data from IoT sensors is outside of this study's scope. The data from the dataset

file was pushed to the IoT Core using an MQTT Client Python script. Figure 3.12 shows the Chl-a soft sensor data flow architecture proposed in this study. The architecture pushes the real-time data to the deployed ML model for real-time inferencing, while the MQTT protocols manage bidirectional data flow. The input data and the output from the ML model are stored in the database. This data can be viewed and exported by the end-user.

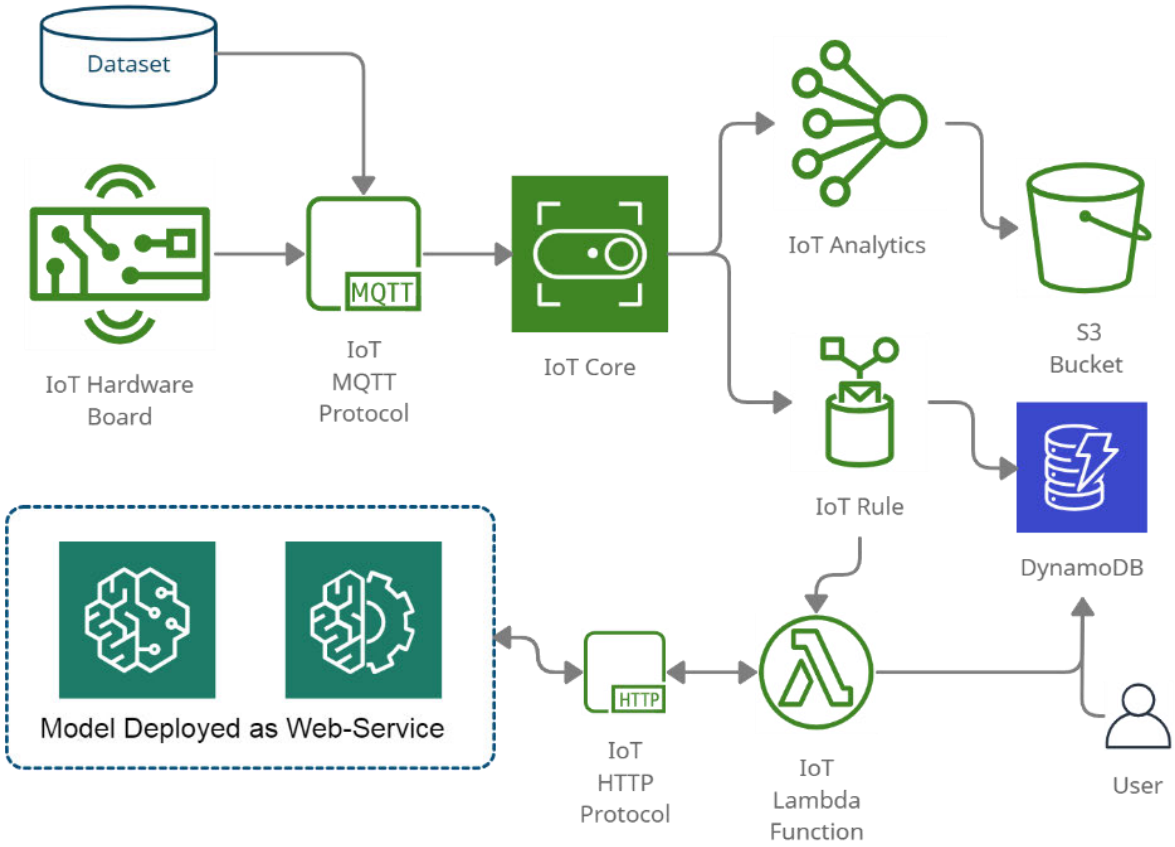

Figure 3.11: Serverless and event-driven IoT and ML inferencing architecture proposed for the Chl-a soft-sensor development in this study

3.6 Chapter Summary

This chapter discusses the methodology adopted in developing the Chl-a soft sensor. Two objectives of the study have been addressed in this chapter. The first objective is to develop an ML model for Chl-a concentration measurement based on four WQ parameters. This chapter addresses the first objective in sections 3.2 to 3.4. On the other hand, the second objective focuses on developing an IoT-cloud application with online ML inferencing, addressed in section 3.5. The following Chapter 4 presents the results of the research outcomes. Included in Chapter 4 is the analysis of the performance of the proposed solution against existing measurement methods. The analysis addresses objective three of this study concerning evaluation comparison for the work done.

CHAPTER 4: RESULTS AND DISCUSSION

4.1 Overview

This chapter presents the work results, beginning with evaluating the four WQ parameters correlated to the Chl-a values. The chapter continues by examining important CART model features. The ML model results are discussed in detail, highlighting this study's best and worst-performing models, with or without hyperparameters tuning. The chapter evaluates the performance of ML models experimented with within this study with ML models favoured in the WQ literature. For completion, the online inferencing data architecture results are also discussed in this chapter.

4.2 Dataset Overview

Figure 4.1 presents the dataset's time series plot of six parameters collected by the WQ sonde, DO, temperature, conductivity, turbidity, TDS and Chl-a. The conductivity (‘cond’ plot in Figure 4.1) matches the TDS (‘tds’) as they are derived from the same sensor, conductivity. Thus, it is redundant to have both in the WQ dataset consideration. Only the conductivity is kept for the correlation mapping.

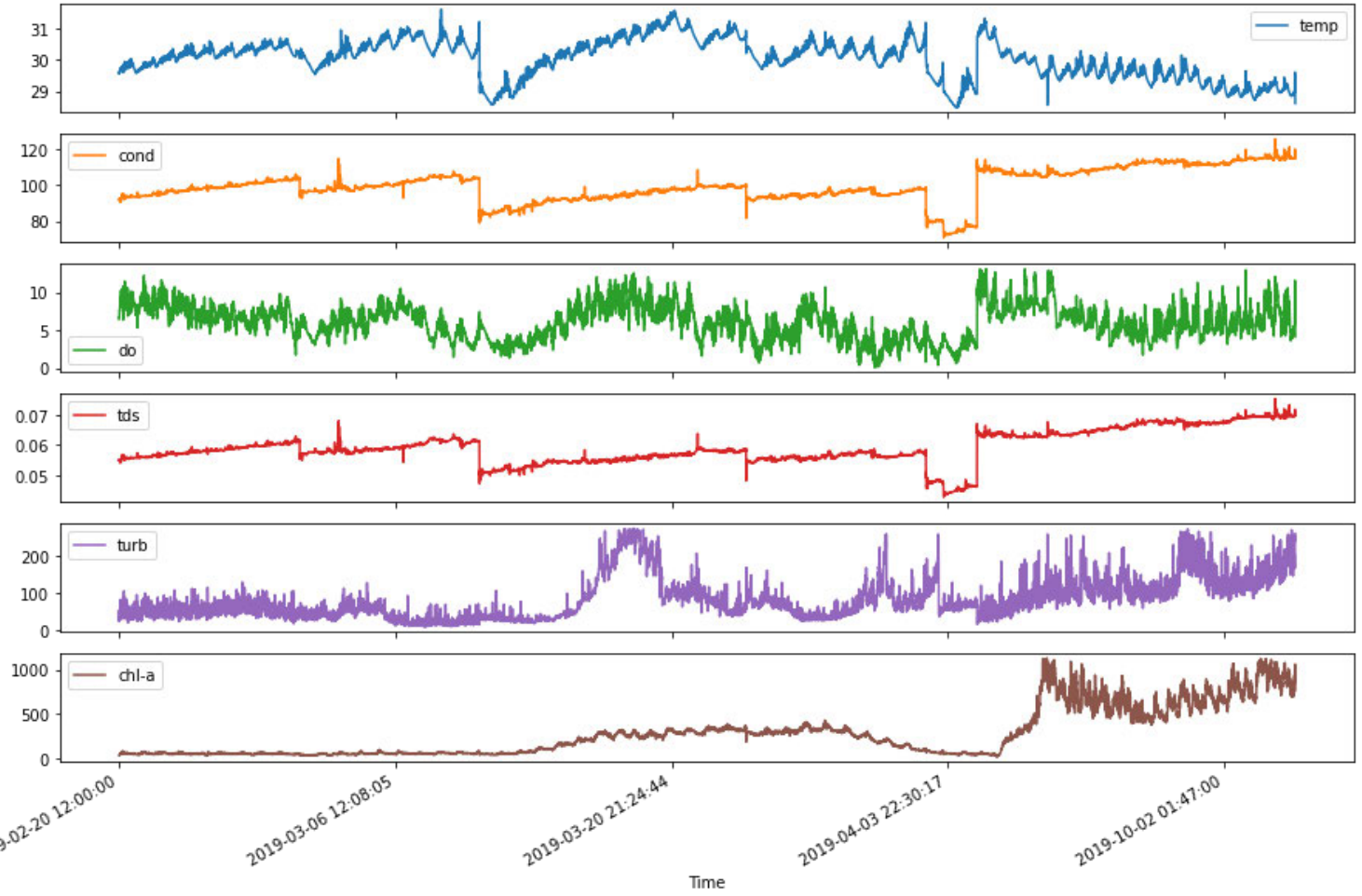

Figure 4.1: Timeseries plot of six WQ parameters from February to October 2019. From the top, temperature (‘temp’), conductivity (‘cond’), dissolved oxygen (‘do’), total dissolved solids (‘tds’), turbidity (‘turb’) and the Chlorophyll-a (‘Chl-a’) are shown

Figure 4.2 shows a closeup of the plot for Chl-a, the parameter of interest that the ML models predict. The Chl-a data collected from February to April (2019) is plotted in green, and the Chl-a data from September to October (2019) is plotted in brown. From February to April (2019), the Chl-a concentration rises and plateaus a little before dropping. However, there is a noticeable spike in the Chl-a concentration values after the break-in data collection. The Chl-a concentration stays relatively high compared to the concentration values between February to April (2019).

Due to the data gap, the distribution of Chl-a concentration data is skewed, as seen in Figure 3.4 and Figure 3.7, respectively. Turbidity also follows the same distribution as Chl-a concentration, although no discernible correlation can be seen between these two parameters in Figure 3.4. Turbidity is a proxy for Chl-a and suspended solids (SS) in the WQ domain. Hence, a rise in turbidity can indicate a rise in Chl-a, SS, or both. Therefore, the distribution of the data for Chl-a and turbidity is similar.

Since the distribution for the WQ parameters is not Gaussian, conventional data standardization techniques were not applied. Data normalization techniques such as Min-Max Scaler and Maximum Absolute Scaler were used because they work well with data that are not normally distributed (i.e. lacks Gaussian distribution). The raw dataset also had significant outliers, likely due to random errors in the sensors. These outliers can be easily visualized with a box plot for every parameter, as seen in Figure 4.3.

The diamond-shaped object in Figure 4.3 represents data points that are the outlier. Figure 4.3 shows that data from all WQ parameters consists of outliers. Therefore, a Z-score was used to drop outliers before training the ML model.

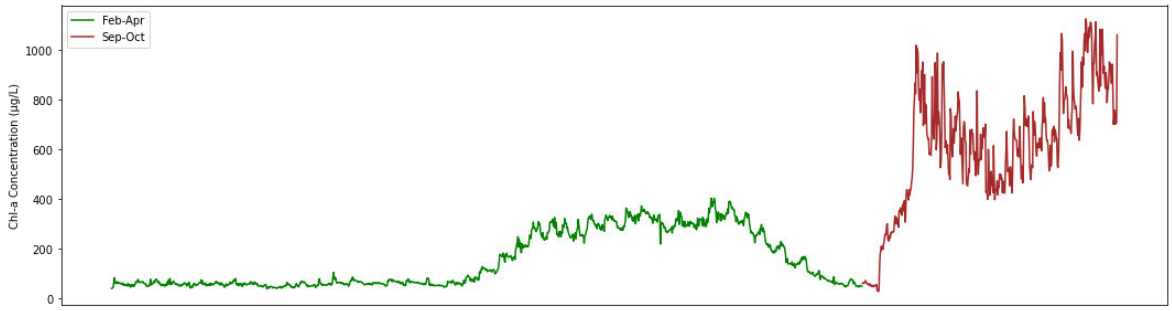

Figure 4.2: Timeseries plot of Chl-a concentration. Data from February to April (2019) is plotted in green, whereas data from September to October (2019) is plotted in brown

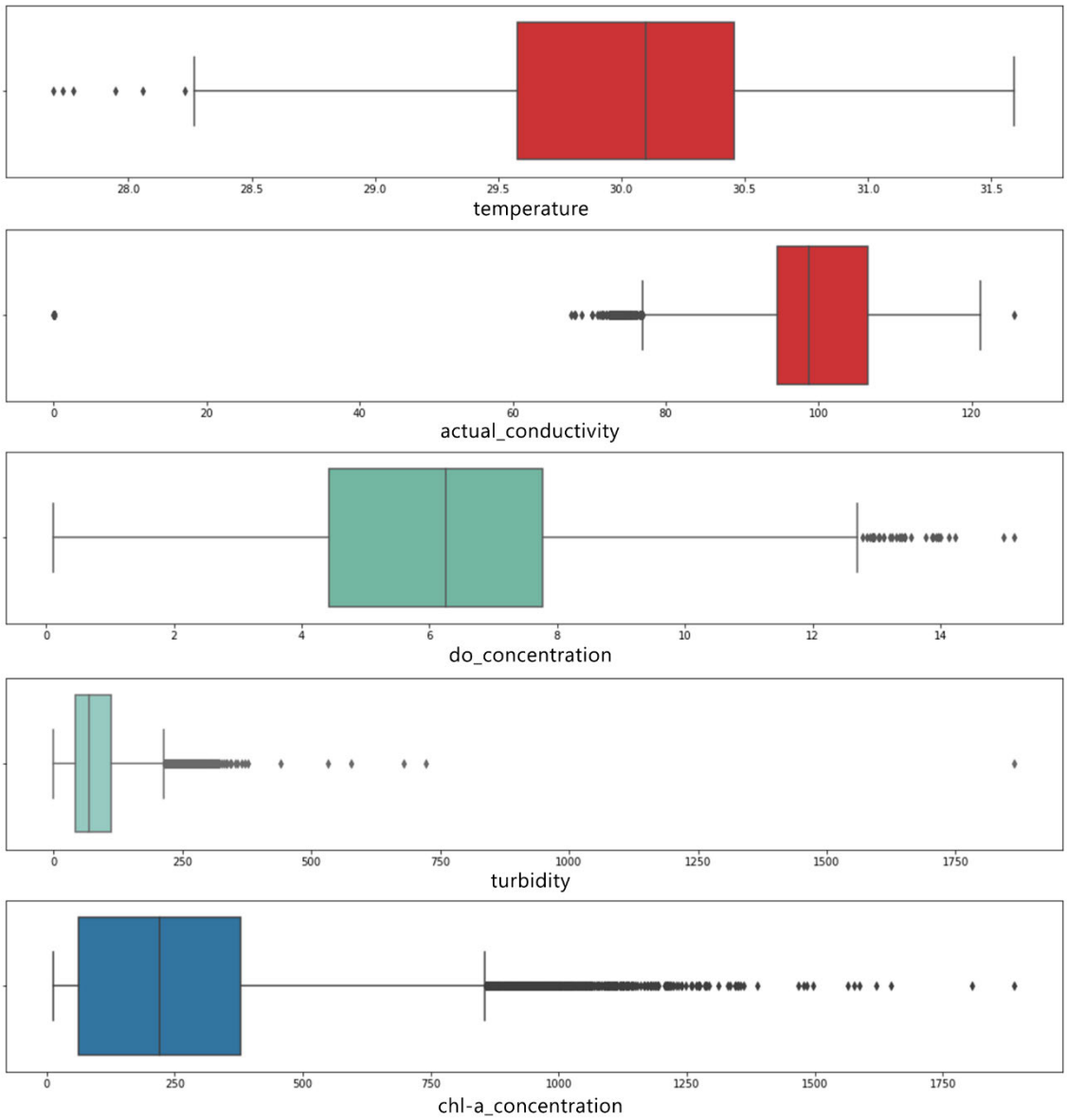

Figure 4.3: Box-plot of five WQ parameters. From the top, temperature (‘temperature’), conductivity (‘actual_conductivity’), dissolved oxygen (‘do_concentration’), turbidity (‘turbidity’) and Chlorophyll-a (‘Chl-a_concentration’) are shown

4.3 Input Feature Importance for CART Models

CART models have an inherent capability of ranking input features based on their importance in predicting the output. Since LightGBM, RF, and XGBoost are all CART models, they can rank the features. The ranking is done by assigning a numerical score to the input feature based on how well the feature can predict the output.

Feature importance plays a crucial role in understanding and deriving insights from the data. Feature importance can also be used for dimensionality reduction or removing less important features to improve efficiency and accuracy. Removing features can also reduce costs in practical applications. For instance, in the WQ domain, each feature corresponds to a physical sensor and eliminating a feature means eliminating the need for a WQ sensor. A sensorless approach can reduce the cost when the sensor is expensive. Some WQ sensors are also labour intensive with maintenance.

Figure 4.4 shows the importance of the LightGBM model's input features. Figure 4.4(a) shows the importance of the DateTime feature to the four WQ parameters, whereas Figure 4.4(b) shows the feature breakdown of the DateTime along with the four WQ parameters.

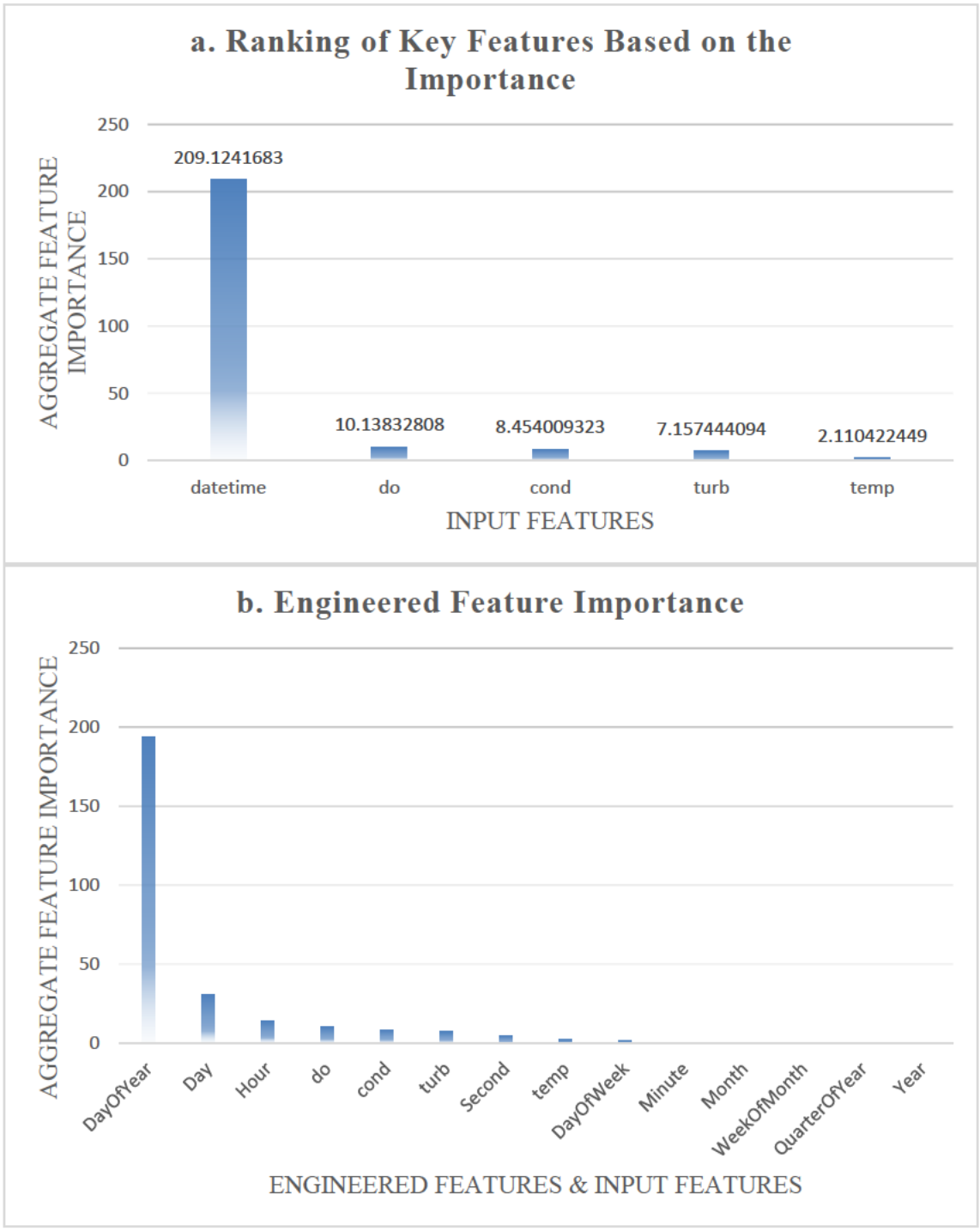

Figure 4.4: Bar chart depicting the importance of input features for the LightGBM model. Figure 4.4(a) shows the overall importance of the DateTime feature, while Figure 4.4(b) shows the importance of individual DateTime features

Datetime was an essential input feature in predicting Chl-a concentration, with 'DayOfYear' being the most critical feature. The value of 'DayOfYear' is '1' on January 1st and '365' on December 31st. The 'DayOfYear' feature is most influential because it corresponds to the season and determines the amount of rainfall on any given day of the year. Since rainfall is a crucial driver of algal growth, rainy seasons will potentially increase Chl-a concentration.

The high increase in Chl-a concentration in September and October in Figure 4.2 coincides with Malaysia's monsoon season. According to the Malaysian Meteorological Department, the southwest monsoon is from May to September, and the northeast monsoon is from November to March. The spike in Chl-a value in September is likely due to the southwest monsoon from May until September.

'Day' and 'Hour' are the second and third most important predictors of Chl-a, as shown in Figure 4.4(b). 'Day' represents the day of the month (e.g., the 20th of February is 20), and the hour is in the 24-hour format. 'Day' is perhaps also used to determine the season, whereas 'Hour' is used to determine the sunlight. With the 'Hour' information, the amount of daylight can be easily approximated. The amount of daylight corresponds to solar irradiance, which is crucial for the algal to carry out photosynthesis and grow.

The fourth most important feature is DO, apparently the most important WQ parameter for Chl-a estimation. DO concentration tend to go up during the day because photosynthesis by algae increases the amount of oxygen dissolved in the water. However, DO concentration falls at night because there is a lack of photosynthetic activities, and algae and other aquatic animals use the DO. This behaviour can be observed through the frequent fluctuations in DO concentration values shown in Figure 4.1.

The fifth most crucial input feature also the second most important WQ feature is the conductivity (i.e., ‘cond’ in Figure 4.4). Conductivity depends on dissolved ions and may correspond to dissolved nutrient ions such as phosphate and nitrate ions. Nutrients are essential for algal growth, and their presence can be inferred by conductivity. Conductivity can, however, measure other dissolved ions that have no relationship with algal growth. Turbidity (‘turb’ in Figure 4.4) is ranked after conductivity and measures the opaqueness of water. The presence of algae affects water clarity, which can be measured using the turbidity parameter. Turbidity is not a strong predictor because water sediments also affect water clarity.

The DateTime ‘Second’ feature follows the ranking after turbidity and is probably used with other important DateTime features to extract meaningful seasonal or day/time information. Temperature is ranked next (labelled in Figure 4.4 as ‘temp’) and is the least important among the four WQ parameters. This finding contradicts the literature review where several authors pointed out the importance of temperature (Lee et al., 2016; Yi et al., 2018; Du et al., 2018).

However, those studies were conducted in temperate or subtropical zones where temperatures vary due to the seasons. In a tropical country such as Malaysia, the temperature stays constant on average, and the temperature is warmer. Since temperature is constant throughout the year, no noticeable difference in algae growth can be seen. Hence, the temperature parameter turned out to be the least important and may be dropped in future studies.

The last and least important feature is ‘DayOfWeek’, probably redundant to other DateTime information and can be dropped without significant loss in model accuracy. Lastly, the model never used the ‘Minute’, ‘Month’, ‘WeekOfMonth’, ‘QuarterOfYear’ and ‘Year’ features and can be dropped in future studies without performance issues.

4.4 ML Model Results

This section is divided into four subsections. The first subsection discusses the results of ML models, the CART models used in this study and the ANN without hyperparameter optimization. The hyperparameters were left in the default setting or manually tuned for good performance. The second subsection discusses the results of the CART models after hyperparameter optimization. The third subsection shows experiment results when an important feature is unavailable. The last subsection describes the ML model comparison.

4.4.1 ML Model Results with Default Hyperparameters

The LightGBM model was trained with the default hyperparameters, and the input features were transformed using MaxAbsScaler. Table 4.1 shows that LightGBM achieved R^2 of 0.995 on the training dataset but R^2 of 0.922 on the testing dataset. The difference between the training and testing R^2 suggests overfitting and is further substantiated by high MAE and RMSE values in the test dataset. Since default hyperparameters are used, the model is not optimized, resulting in slight overfitting. However, the overfitting is less than ANN. The LightGBM outperformed the ANN by a significant margin on the test dataset. Figure 4.5 shows that LightGBM accurately predicted the Chl-a concentration trends on the test dataset and captured all the spikes with reasonable accuracy. Figure 4.5 also shows a big jump in the Chl-a concentration value during September towards the plot's end. Interestingly, the LightGBM predicted the jump slightly earlier than the actual spike.

Table 4.1: LightGBM model results without hyperparameter optimization

	LightGBM			
	Preprocessing	**R^2**	**MAE**	**RMSE**
Train	MaxAbsScaler	0.995355	10.561939	18.155827
Test		0.922348	28.101163	71.028762

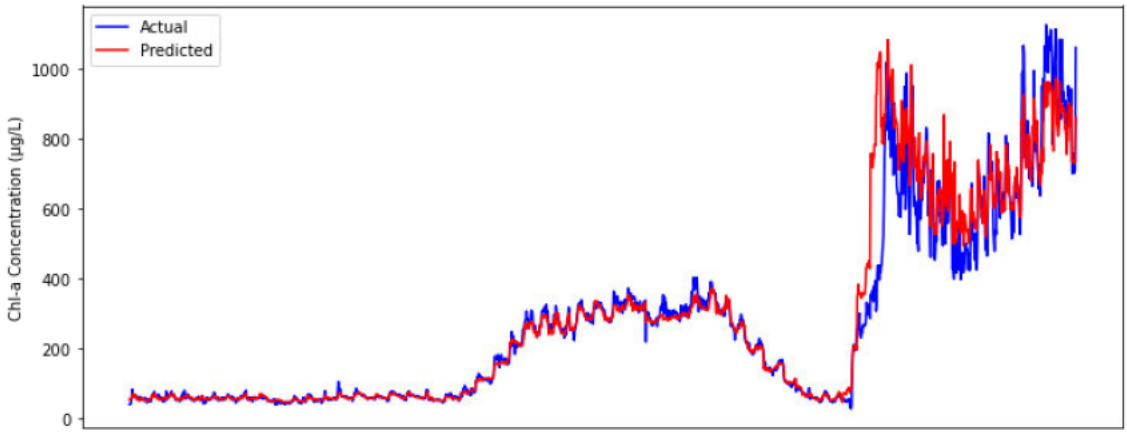

Figure 4.5: Time series plot of the actual test data (blue colour) and the predicted Chl-a concentration value by LightGBM (red colour) without hyperparameter optimization

When the RF was trained with default hyperparameters, the model achieved an R^2 of 0.998 on the training dataset and an R^2 of 0.928 on the testing dataset (see Table 4.2). RF's input features were transformed using MinMaxScaler instead of MaxAbsScaler because RF performs better with MinMaxScaler than MaxAbsScaler (Ahsan et al., 2021). RF's performance is slightly better than LightGBM in this case, where none of the models' hyperparameters was fine-tuned. RF also achieved lower MAE and RMSE values than LightGBM, proving that RF is more accurate than LightGBM with default hyperparameters. RF also predicted the trends of the Chl-a concentration in the test dataset with good accuracy and captured the small spikes better than LightGBM. However, Figure 4.6 shows the RF predicting a significant spike earlier than when the actual one occurs, with consistent patterns.

Table 4.2: RF model results without hyperparameter optimization

	RF			
	Preprocessing	**R^2**	**MAE**	**RMSE**
Trai	MinMaxScaler	0.998456	4.892936	10.469093
Test		0.928315	26.520661	68.245324

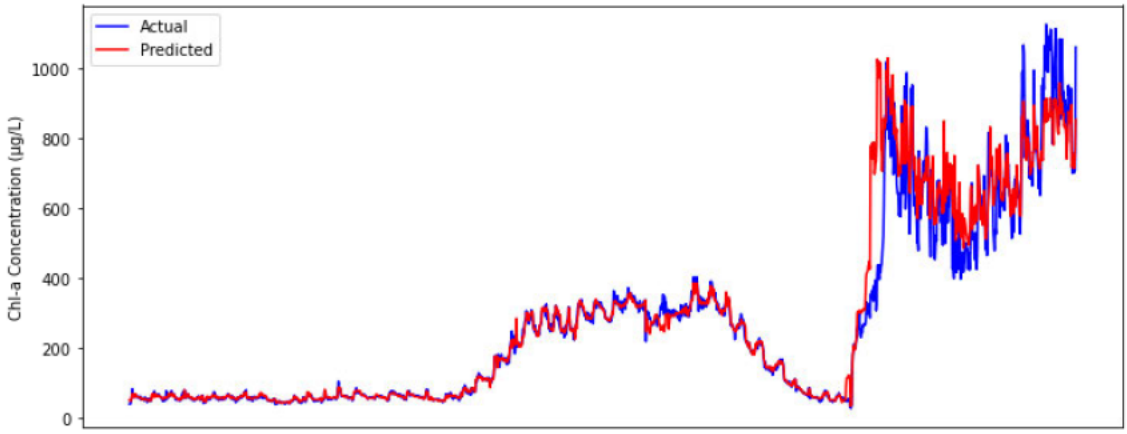

Figure 4.6: Time series plot of the actual test data (blue colour) and the predicted Chl-a concentration value by RF (red colour) without hyperparameter optimization

XGBoost is also trained with default hyperparameter and performs the worst among the CART models. The training dataset obtained an R^2 of 0.982 and an R^2 of 0.912 on the testing dataset (see Table 4.3). Like LightGBM, XGBoost also had better results when input features were scaled using MaxAbsScaler. XGBoost also had higher MAE and RMSE than the other CART models suggesting that XGBoost is the least accurate CART model (compared to LightGBM and RF).

The higher prediction errors are reflected in the time series plot of Chl-a concentration in Figure 4.7. Although XGBoost predicted the overall trend of the plot accurately, it failed to estimate the subtle fluctuations of Chl-a concentration. However, XGBoost still outperformed the ANN model by a significant margin.

Table 4.3: XGBoost model results without hyperparameter optimization

	XGBoost			
	Preprocessing	**R^2**	**MAE**	**RMSE**
Train	MaxAbsScaler	0.981824	20.162734	35.915016
Test		0.91185	32.513675	75.67769

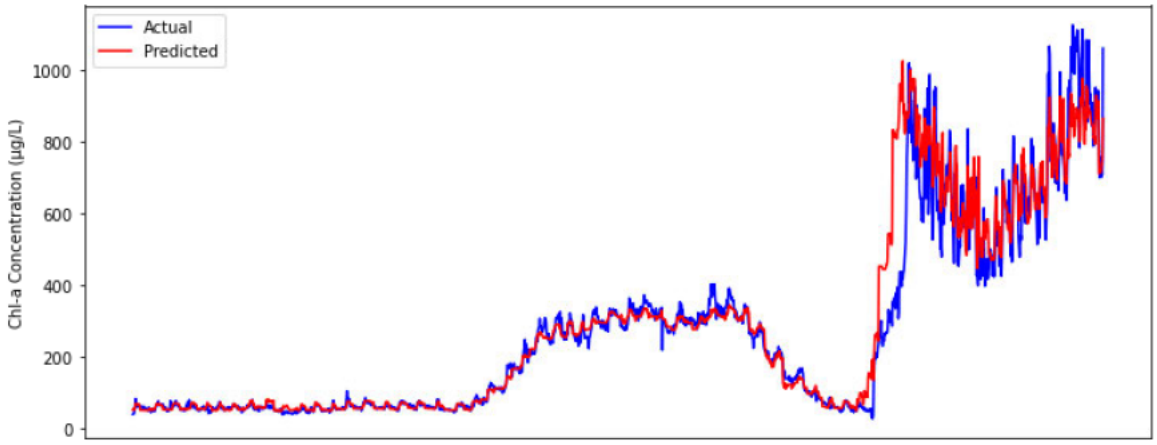

Figure 4.7: Time series plot of the actual test data (blue colour) and the predicted Chl-a concentration value by XGBoost (red colour) without hyperparameter optimization

4.4.2 ML Model Results with Hyperparameters Tuning

The results of the CART models selected in this study were encouraging, even with default hyperparameters. The study explores hyperparameter tuning to see if the CART model performance in predicting Chl-a would improve. However, the constraint of local machine computing resources does not allow automating the time-consuming, iterative tasks of manual hyperparameter tuning. External computing resources were considered. The Azure Automated ML (or AutoML) offers a cloud-based service that can be programmed with Python to tune the CART models hyperparameters proposed in this study.

The LightGBM model was preprocessed using the MaxAbsScaler. The hyperparameter optimized LightGBM model achieved a very high R^2 of 0.989 on the test dataset, and it is the highest R^2 achieved in this study. Table 4.4 shows the results. The optimized

LightGBM model also has a similar R^2 on the training set, suggesting no overfit. The MAE and RMSE obtained in this study are also the lowest among all the other models suggesting LightGBM is the most accurate model compared to other models. The MAE and RMSE in both train and test sets are close, suggesting the absence of overfitting during training.

LightGBM performed better in the unseen test set, and MAE and RMSE are lower than the training set's MAE and RMSE. The LightGBM closely mirrors the actual Chl-a concentration trend on the testing set in Figure 4.8, capturing the overall trend and the small spikes along with the plot. Although LightGBM's prediction occasionally misses either very high or low spikes in the plot, overall, LightGBM's plot mirrors the actual plot better than other models in this study.

Table 4.4: LightGBM model results with hyperparameter optimization

	LightGBM			
	Preprocessing	**R^2**	**MAE**	**RMSE**
Train	MaxAbsScaler	0.98906	14.412	27.774
Test		0.989095	13.898911	26.618131

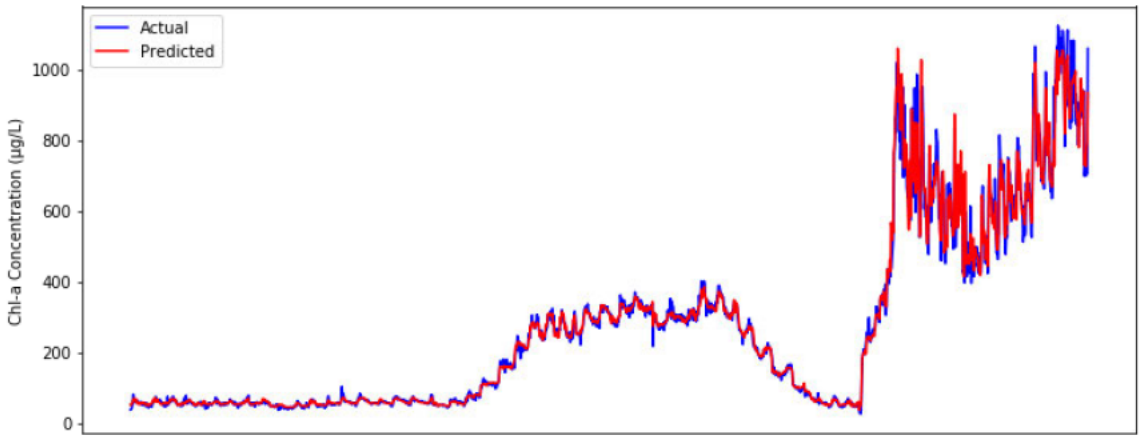

Figure 4.8: Time series plot of the actual test data (blue colour) and the predicted Chl-a concentration value by LightGBM (red colour) with hyperparameter optimization

An RF model trained with input features scaled with MinMaxScaler achieved better performance when the hyperparameters were fine-tuned. Table 4.5 shows that the RF achieved an R^2 of 0.9796 on the test dataset and an R^2 of 0.9795 on the training dataset. The R^2, MAE and RMSE values were close, suggesting little to no overfitting in the model's training. RF is the second-best performing model after LightGBM, and RF also predicted the trends and fluctuations of the Chl-a concentration's time series plot with reasonable accuracy (see Figure 4.9). The RF cannot capture large spikes like LightGBM, so RF has higher MAE and RMSE than LightGBM.

Table 4.5: RF model results with hyperparameter optimization

	RF			
	Preprocessing	**R^2**	**MAE**	**RMSE**
Train	MinMaxScaler	0.97947	18.755	38.066
Test		0.979604	17.491467	36.402656

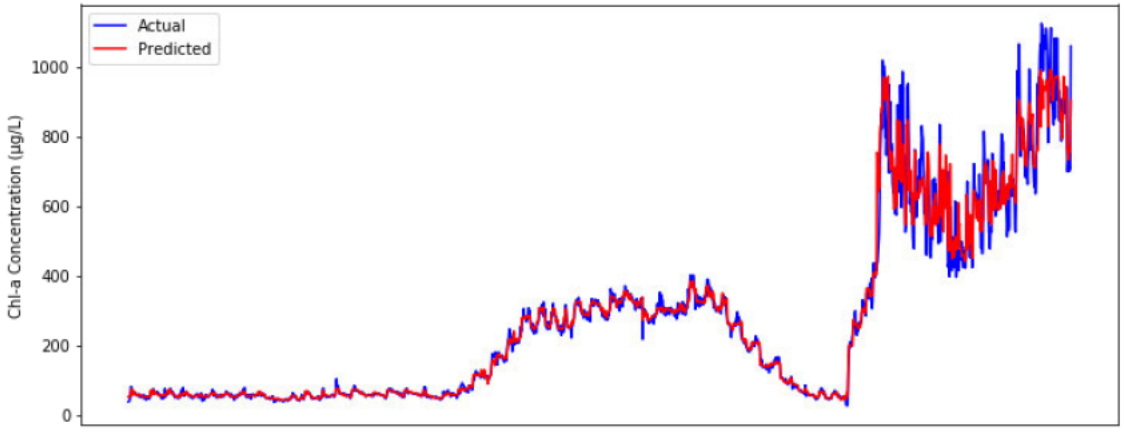

Figure 4.9: Time series plot of the actual test data (blue colour) and the predicted Chl-a concentration value by RF (red colour) with hyperparameter optimization

Input features of XGBoost were also transformed using MaxAbsScaler for better performance. After hyperparameter optimization, XGBoost had an R^2 of 0.9776 on the testing set, as shown in Table 4.6. The R^2, MAE, and RMSE for both train and test datasets are very close and hence no noticeable overfitting during the training. XGBoost is the third-best performing model in this study. The XGBoost's predicted Chl-a concentration values could be seen matching closely with the actual Chl-a concentration values of the testing dataset (see Figure 4.10). Like the unoptimized XGBoost model, optimized XGBoost cannot capture the plot's small fluctuations and spikes but the overall trend with reasonable accuracy.

Table 4.6: XGBoost model results with hyperparameter optimization

	XGBoost			
	Preprocessing	**R^2**	**MAE**	**RMSE**
Train	MaxAbsScaler	0.97843	21.51	39.013
Test		0.97762	20.841458	38.131637

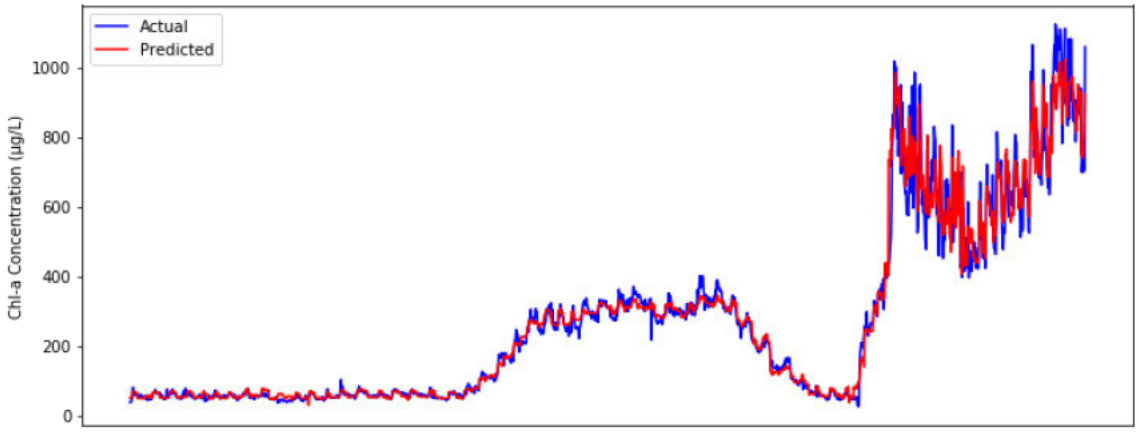

Figure 4.10: Time series plot of the actual test data (blue colour) and the predicted Chl-a concentration value by XGBoost (red colour) with hyperparameter optimization

4.4.3 Hyperparameter Optimized CART Models without DateTime Feature

Figure 4.4 shows that the DateTime is the most critical feature for the CART models training, followed by the four WQ parameters of DO, conductivity, turbidity and temperature. The DateTime feature and engineered features are not always part of WQ-related work. In this study, the DateTime feature is considered following recommendations in Huo et al. (2013) and Jimeno-Sáez et al. (2020). All three optimized CART models were trained on the same training dataset without the DateTime feature to understand its effect on the models' performance. The train and test results of all the CART models are tabulated in Table 4.7.

Table 4.7: Results of optimized CART models on Train and Test dataset

	Model	Preprocessing	R^2	MAE	RMSE
Train	LightGBM	MaxAbsScaler	0.95317	33.885	57.341
Test			0.95151	33.459	56.130
Train	RF	MinMaxScaler	0.93551	41.365	67.358
Test			0.93656	40.310	64.203
Train	XGBoost	MaxAbsScaler	0.9158	52.898	77.092
Test			0.9227	50.132	70.853

Table 4.7 shows that the best-performing model without the DateTime feature is the LightGBM. The LightGBM model achieved an R^2 of 0.95317 on the training dataset and an R^2 of 0.95151 on the test dataset. The second-best performing model is RF, with an R^2 of 0.93551 and 0.93656 on the training and test dataset. The XGBoost had an R^2 of 0.9158 and 0.9227 on the training and test dataset, making it the least performing model. All

three CART models' train R^2 and test R^2 are close, implying little to no overfit. The R^2 for all three models are over 0.90 suggesting a good fit. However, the MAE and RMSE are almost double compared to the hyperparameter optimized models with the DateTime Feature.

Since the LightGBM model is the best performing model without the DateTime feature, the feature importance values were extracted and plotted on a bar chart shown in Figure 4.11. Interestingly, DO is no longer an essential WQ feature. The most critical WQ feature without DateTime was conductivity ('cond'). Temperature ('temp') is the second most important WQ feature when DateTime is unavailable, contrasting its last position for WQ feature ranking when DateTime was available. Next is turbidity ('turb'). While DO was the most critical WQ feature when DateTime was available, DO is now the least essential WQ feature without DateTime.

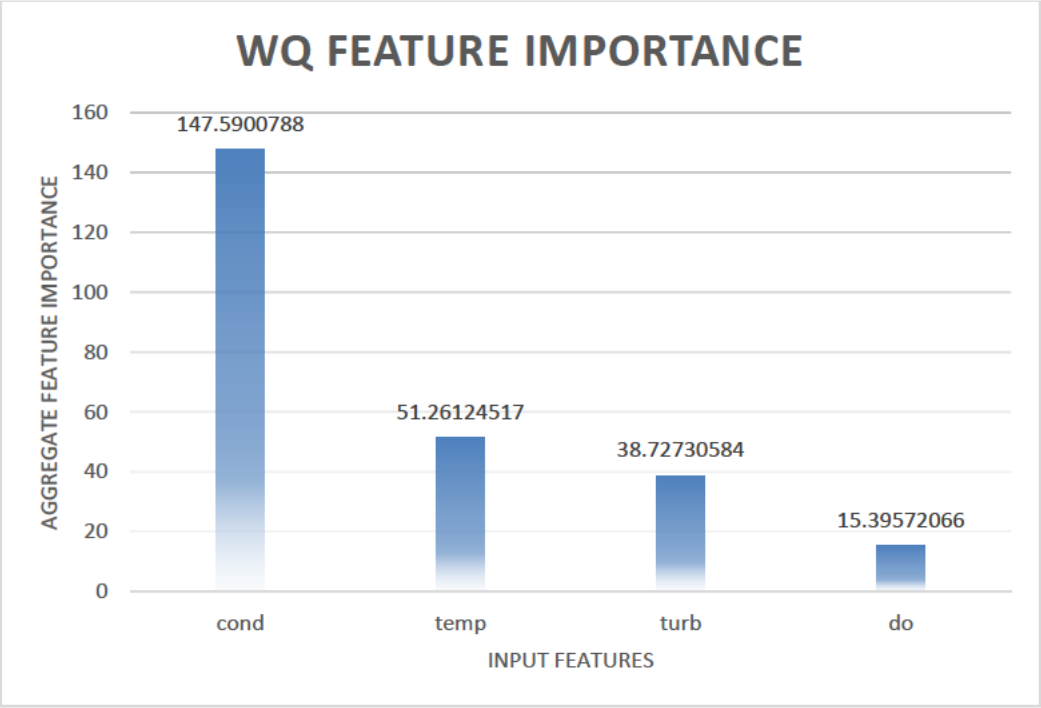

Figure 4.11: Bar chart depicting the importance of input features for the hyperparameter optimized LightGBM model without DateTime Feature

The model performance, ranking-wise, is consistent whether DateTime is introduced or not—the LightGBM leading the ranking, followed by RF and XGBoost, so long their hyperparameters are fine-tuned. However, the same cannot be said for the WQ parameters input feature importance. There is a significant difference in the contribution of DO, temperature, conductivity and turbidity towards Chl-a concentration estimation when DateTime is available or not.

4.4.4 Comparison of ML Models with Models in Literature

Models developed in this study (both unoptimized and optimized) are compared with the state-of-the-art models suggested in the literature. Specifically, studies that predicted Chl-a in lakes were selected because the physicochemical and spatiotemporal properties vary in different water bodies. There are no agreed standard performance metrics for evaluating and comparing ML models in the WQ literature. As a result, different studies reported different performance metrics to compare their models. The standard metric that is almost always reported is R^2. Therefore, the R^2 is used to compare the models in the literature with the models developed in this study.

Table 4.8 lists the ML models from this study. The models from the literature are arranged in descending order following their R^2 performance on the test set. The R^2 value is rounded to two decimal places to match the literature's reported results for better representation. The hyperparameter optimized CART models of LightGBM, RF, and XGBoost occupy the first, second, and third positions. Interestingly, the unoptimized RF secured the fifth position, followed by the unoptimized LightGBM, which shares the sixth rank with SVR (Nieto et al., 2019) and CNN (Choi et al., 2019).

Table 4.8: Comparison of all the models developed in this study with other studies from the literature based on test set coefficient of determination

Rank	References	Model	Test Set R^2
1	*This study*	LightGBM[2]	0.99
2		RF[2]	0.98
2		XGBoost[2]	0.98
3		LightGBM[3]	0.95
4		RF[3]	0.94
5		RF[1]	0.93
6		XGBoost[3]	0.92
6		LightGBM[1]	0.92
6	*Nieto et al. (2019)*	SVR	0.92
6	*Choi et al. (2019)*	CNN	0.92
7	*This study*	XGBoost[1]	0.91
7	*Lee et al. (2016)*	ANN	0.91
8	*Li et al. (2018)*	RF	0.82
9	*This study*	ANN	0.80
10	*Barzegar et al. (2020)*	CNN-LSTM	0.76
11	*Huo et al. (2013)*	ANN	0.70

1 Unoptimized model with DateTime
2 Hyperparameter optimized model with DateTime
3 Hyperparameter optimized model without DateTime

The unoptimized XGBoost shares the seventh rank with the ANN developed by Lee et al. (2016). RF developed by Li et al. (2018) secures the eighth position, the lowest-performing CART model in the comparison. Interestingly, ANN developed in this study outperforms deep learning models like the hybrid CNN-LSTM developed by Barzegar et al. (2020) and ANN developed by Huo et al. (2013).

Although Nieto et al. (2019) achieved a high R^2 of 0.92, the authors trained the SVR model on a monthly dataset containing only 244 data samples (see Table 2.2), whereas the models in this study were trained on 6813 data samples and tested on another 1703 data samples. Since Nieto et al. (2019) trained their SVR model on a limited dataset, there is a high possibility of overfitting because of a lack of training data. Also, the SVR model requires 15 different WQ parameters to predict Chl-a concentration, whereas only four WQ parameters are required in this study.

Similarly, the CNN model by Choi et al. (2019) was trained on a limited dataset of around 2190 data samples using 8 WQ input parameters to predict Chl-a concentration. Also, the paper by Choi et al. (2019) lacked a detailed analysis of the model developed. Hence there is a possibility of overfitting in the models. Lee et al. (2016) and Li et al. (2018) also trained their models on 60 and 391 data samples (see Table 2.2). Their results could be susceptible to overfitting and a possible lack of generalization. Also, the ANN model by Lee et al. (2016) requires 15 WQ input parameters, and Li et al. (2018) require at least 7 WQ parameters for the Chl-a prediction.

Only Barzegar et al. (2020) used an extensive dataset (more than 35000 data samples) to train and test their CNN-LSTM model. However, their R^2 on the test dataset was not very high, as shown in Table 4.8. The CNN-LSTM model by Barzegar et al. (2020) requires at least 6 WQ parameters to predict Chl-a concentration. Finally, the least performing model is the ANN developed by Huo et al. (2013). The model was trained on only 100 data samples. Since the dataset is very small, the model has a high chance of overfitting the training data. Also, the ANN model by Huo et al. (2013) requires 10 WQ input parameters for predicting Chl-a concentration. In contrast, ANN developed in this study has a higher R^2 value than the ANN developed by Huo et al. (2013) and requires fewer parameters for Chl-a prediction.

In a nutshell, unoptimized RF and LighGBM outperformed the state-of-the-art models in the literature. Performance of the CART models (LightGBM, RF, and XGBoost) were significantly boosted by hyperparameter optimization, as hypothesized. Overall, the results show the suitability of CART models for developing the Chl-a soft sensor, especially for lacustrine water in Malaysia.

4.5 ML Real-Time Inferencing Benchmarks

The online ML inferencing was done by deploying the hyperparameter optimized LightGBM model to the cloud as a web service. The LightGBM was hosted on the Azure cloud, and inferencing was done using the endpoint configured by Azure. The LightGBM inferencing was carried out by feeding the model with data points from the test dataset. A popular API platform called Postman was used to pack the data from the test set into a JSON data packet and make an HTTP POST request to the inferencing endpoint.

Figure 4.12 shows several columns on the right highlighting the input data rows with DateTime, temperature (labelled as 'temp), conductivity ('cond'), DO ('do'), and turbidity ('turb') parameters. These WQ parameters are defined in the body of the API request; see the script on the left. The Chl-a concentration value in the dataset, 60.483 μg/L, is depicted in the far-right column in Figure 4.13. After making the HTTP request, the model in the cloud carried out inferencing and sent a response with the predicted Chl-a concentration value of 58.187 μg/L. See the bottom script on the left in Figure 4.12.

The difference between the actual Chl-a and predicted Chl-a concentrations is 2.296 μg/L. From table 4.4, it can be seen that the RMSE for the hyperparameter optimized LightGBM model is around 26.6 μg/L. RMSE estimates how far off the predicted value is from the actual data point, i.e. RMSE estimates the standard deviation of the predicted value from the actual value. The error of 2.296 μg/L is smaller than the RMSE of 26.6 μg/L. Therefore, the result is acceptable. In Table 3.1, the minimum and maximum values of Chl-a concentration are 24.96 μg/L and 1126.18 μg/L, respectively, and the standard deviation is 264.21 μg/L. Given the wide range of Chl-a concentration values and the relatively large standard deviation, a 2.296 μg/L error is acceptable.

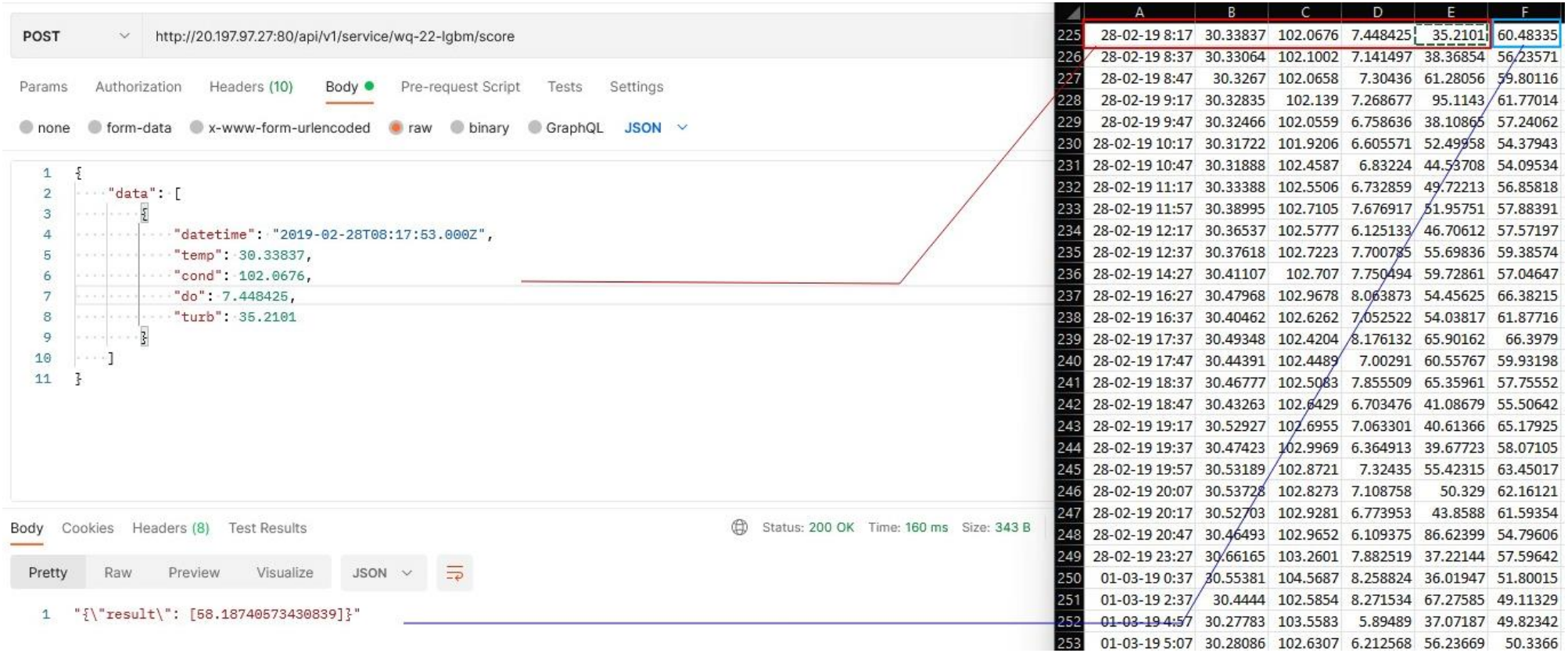

Figure 4.12: HTTP POST request using Postman API platform. The rectangle in red denotes the input data, and the rectangle in blue is the actual Chl-a concentration value

Postman API platform was used for testing because it provides information about the request's response time and the size of both the response and the request. The HTTP request during the test took 162.55 milliseconds (ms) in total (see Table 4.9). This response time is not fixed and varies between 100 and ms-200 ms, but that speed is justified for real-time inferencing. The HTTP POST request is 537 bytes, as seen in Table 4.11, while the HTTP response is 343 bytes, as shown in Table 4.10. The response size is smaller because the response only contains Chl-a concentration values; hence, the request's body size is 35 bytes (see Table 4.10). However, the HTTP request body size is 315, containing all 4 WQ input data and datetime information.

Table 4.9: HTTP request-response time in ms

Event	Time (ms)
Prepare	11.34
Socket Initialization	0.34
Transfer Start	148.68
Download	2.13
Process	0.07
Total	**162.55**

Table 4.10: HTTP response size in bytes

HTTP Response	Size (Bytes)
Headers	308
Body	35
Total Size	**343**

Table 4.11: HTTP request size in bytes

HTTP Request	Size (Bytes)
Body	222
Headers	315
Total Size	**537**

4.6 Discussion

This chapter presents the results of CART-based ML models tested with the WQ dataset from Tasik Aman. Ranking the input feature importance for the CART models, i.e., RF, LightGBM and XGBoost, shows DateTime playing the most significant contributing factor in the correlation between Chl-a and the four WQ parameters, DO, temperature, conductivity, and turbidity.

Nine sets of experiments were done on the WQ dataset. The first three experiments focus on the manual development of RF, XGBoost, and LightGBM with default hyperparameters on a local machine. The LightGBM's R^2 margin between the training dataset (0.995) and testing dataset (0.9223) suggests overfitting and is justified with high MAE and RMSE in the testing dataset. The R^2 margin between the RF training dataset (0.998) and the testing dataset (0.928) suggests a slight overfit. However, the RF's MAE and RMSE are lower than the LightGBM, suggesting RF performs better predicting the Chl-a concentration values. The XGBoost shows the worst performance among the manually developed CART models. All CART models show better prediction potential than the widely favoured ANN for the WQ correlation study. Their results are represented graphically in Figure 4.5, Figure 4.6 and Figure 4.7.

Motivated by these results, three other experiments are conducted where the hyperparameters of RF, LightGBM and XGBoost are fine-tuned with Azure's AutoML. LightGBM's training and testing R^2 values have a fine margin, suggesting no overfitting. The same pattern is recorded with optimized RF. Notably, the LightGBM's R^2 value on the testing dataset is significantly impressive.

All three CART models have similar MAE and RMSE on the train and test datasets showing no noticeable overfitting during training. LightGBM also has the lowest MAE and RMSE values, overtaking RF in Chl-a prediction capabilities. Figure 4.8, Figure 4.9 and Figure 4.10 show the LightGBM model performance graphically compared to the optimized RF and XGBoost. The Chl-a values between actual and predicted overlap very closely, not missing any large spikes or small fluctuations in the LightGBM model. However, the RF missed the large spikes, while the XGBoost missed the smaller fluctuations. The RF beats the XGBoost performance by a slight difference.

The contributing factor to the CART models' performance thus far is the DateTime feature. The final three experiments focused on understanding the significance of the DateTime feature to the CART models' input feature importance. In particular, what would happen to the WQ parameters ranking when the DateTime feature is removed. The models selected for these experiments are the CART models with optimized parameters, seeing their performances are leading thus far.

Table 4.7 summarizes the results of the optimized CART models when the DateTime feature is unavailable. The results show that the CART models are ranked similarly to when DateTime is available. The LightGBM performance is superior to RF, and RF is superior to XGBoost in Chl-a prediction capabilities. However, without DateTime, the WQ parameters contribute differently toward the Chl-a prediction. The order after DateTime was DO, conductivity, turbidity and temperature. When DateTime is unavailable, conductivity takes first position, followed by temperature, turbidity and DO. The dataset is reviewed again to learn possible reasoning for such behaviour.

The WQ dataset was collected between February and October 2019, with a gap from April to September 2019. That leaves the 2019 dataset in two chunks, from February to April and September to October. A closer inspection of the weather information shows that both chunks were collected during the peak monsoon season of the year. Peak monsoon in Malaysia means heavy rainfall, and according to the literature, it can frequently disrupt the consistency of the WQ parameter values.

Weather changes may have contributed to the importance of the WQ parameter when DateTime is available. The breakdown of 10-minute intervals is specific in determining when is the day, night, most sunlight, most rain, most heat, and most moisture occur for the day or week or month. The WQ parameters contributing to these weather values are mapped to specific minute intervals. However, when DateTime is removed, the ML model relies solely on WQ correlation to Chl-a—in hindsight, predicting when the model has temporal information is more accessible.

The WQ correlation map in Figure 3.1 shows that nutrients contribute to Chl-a concentration. On the other hand, nutrients can be derived from conductivity, turbidity, temperature, and DO by exploiting the intercorrelations (Castrillo & García, 2020). The conductivity measures the concentration of dissolved ions in an aqueous solution. The rise in conductivity value can signify higher nutrient content at the lake. Most algal blooms result from excess nutrients; thus, conductivity value is easily a solid indicator for Chl-a concentration. Therefore, when the DateTime feature is unavailable, it is not surprising that conductivity leads the input feature importance for the CART models.

Warmer temperatures prevent water from mixing, allowing algae to grow thicker and faster. Warmer water is more accessible for tiny organisms to move through and allows algae to float to the surface faster. Furthermore, algal blooms absorb sunlight, warming water and promoting more blooms. More light can penetrate the water column when turbidity is low, allowing optimal algal growth conditions. The turbidity correlates positively with temperature for this reason; in retrospect, they feed each other. Warmer temperature leads to algal growth, which means turbid water. Therefore, temperature and turbidity determine Chl-a concentration when DateTime is unavailable.

Stratification is the division of the water column into layers with different densities. The layers can be caused by different salinity, temperature, nutrient, and DO combinations. For example, the bottom layer of lakes is usually filled with sediments full of decaying materials. The decay releases nutrients deep underneath the water surface, which depletes DO. Low DO harms fish that lives at the bottom. Closer to the lake surface, more algae or nutrients mean more food available, increasing bacteria count. Bacteria need oxygen in the water to survive, so they also use DO.

Heavy rainfall during monsoon seasons attracts high volume inflows into lakes and reservoirs. This phenomenon disturbs stratification and drastically increases the DO level at the bottom column. Therefore, the DO level can shoot up when there is heavy rainfall. The Malaysian monsoon promises plenty of rainfalls on average. Compared to the coastal areas, fewer downpours are observed in the midland, such as Petaling Jaya, where Tasik Aman is. The WQ parameter dataset collected for the study shows that DO level is affected by the rise of nutrients, temperature, and turbidity values. Therefore, DO is expected to take less precedence than temperature and turbidity when DateTime is unavailable.

The CART models lose prediction accuracy when the DateTime feature is removed, likely due to losing the temporal aspect for the WQ parameter correlations. Table 4.8 shows how the models rank compared to other works in the domain. CART models optimized with DateTime lead the Chl-a prediction capability, followed by optimized models without DateTime. The unoptimized models fare similarly to models used in other works such as SVR, CNN, and LSTM, albeit slightly better. ANN has been used widely in WQ-related work; however, it performed poorly on the dataset in this study.

The IoT-cloud application developed with ML inferencing shows an architecture integrating services from two cloud giants; AWS's IoT-Core and Microsoft's Azure. The IoT-cloud application completes the Chl-a soft sensor developed as a real-time inferencing solution. The main challenge in the development is selecting the suitable protocol for bidirectional data communication. Data can go through the IoT application (human-machine interface) when collecting WQ parameters. The results from the ML inferencing can be conveyed back to the human-machine interface for human decision-making. The MQTT protocol selected successfully demonstrates bidirectional data communication in the experiment.

Another challenge in developing the Chl-a soft sensor is managing the communication between AWS and Azure. Here, the HTTP POST protocols successfully send data packets from the IoT-Core to the ML inferencing endpoint of Azure. The inferred Chl-a concentration value is displayed along with the WQ parameters values used for the inferencing. There is a time delay of 100-200ms before the inferred outcome is displayed, but this delay is negligible to human eyes and should feel real-time.

4.7 Chapter Summary

This chapter discusses the experiment results of three CART models, LightGBM, RF and XGBoosts, on the WQ parameters dataset collected at Tasik Aman. In doing so, this chapter addresses the third objective of the study concerning a comparative evaluation of the proposed model against the performance of other models used in the Chl-a literature. Table 4.8 summarizes how the CART models selected in this study performed against other models in existing works. The next chapter concludes this dissertation.

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 Conclusion

This dissertation begins with a question – can ML be used to develop a Chl-a soft sensor using a minimal number of WQ parameters? The answer is yes, and in this study, a Chl-a soft sensor has been developed with a LightGBM model using four WQ parameters, DO, temperature, conductivity, and turbidity. This study achieves the Chl-a soft sensor with three fewer WQ parameters than other WQ studies between 2013 and 2020. Most of these works proposed seven or more WQ parameters for predicting Chl-a concentration levels.

The four WQ parameters in determining Chl-a will not be possible without the contribution from other researchers. In their observations, Huo et al. (2013) and Jimeno-Sáez et al. (2020) found that Chl-a can be predicted from the DO, temperature, turbidity, nutrients and month. Meanwhile, Castrillo & García (2020) found that nutrients are correlated with DO, temperature, conductivity and turbidity. This dissertation supports these findings and suggests a Chl-a correlation mapping using only four WQ parameters, as depicted in Figure 3.1.

The notion of developing Chl-a soft sensors for the water community is the desire to find a more sustainable approach to improving WQ monitoring. Presently, WQ data collection is feasible via multiparameter sondes. Almost all of the work published in this domain collected WQ data using this type of sonde. The sonde often performs well with industry-grade calibration and minimal maintenance cost. The sonde technology has evolved with cloud-storage features, making collecting and recording WQ data efficient.

The sondes can be left at the lake for an extended period to get a continuous supply of WQ parameters data. Sondes with connectivity features can push data to the cloud for data logging. When connectivity is unavailable, most sondes have internal storage for data logging. To copy logged data to a preferred storage point, one has to retrieve the storage card from the sonde. However, analytics on these collected data is usually a separate solution. Researchers often tabulate and review the data manually to learn about the characteristics of a lake. More importantly, eliciting contributing factors or ranking the WQ parameters is tedious and time-consuming, especially with the nonlinear nature of the WQ parameters. Managing vandalism is also a challenge in WQ monitoring, as the multiparameter sondes are expensive and susceptible to theft.

Learning lake algal bloom behaviour requires frequent WQ data sampling; as frequent as 10-minute intervals have been suggested. Environmental factors can change suddenly, so sampling should continue throughout the day and night. The lake ecosystem is alive and ever-changing when factoring in dynamic external factors. Hence, the water community must consider months and years of continuous WQ data collection to learn the lake's behaviour. Data scarcity has been why most works struggle to find recent datasets and consider past data to predict Chl-a concentration.

Table 2.2 summarizes the Chl-a prediction work using ML in different water bodies. Only Barzegar et al. (2020) have access to 35,000 data points from six WQ parameters. The rest are 8,000 data points or lower, some even below 100 data points to perform Chl-a prediction. This dissertation obtained about 8887 data points; after cleaning, the number is reduced to 8516. It is motivating that ML can help the water community with Chl-a prediction, but careful dataset planning and retrieving sufficient data volume can increase the ML predicting capacity.

The Chl-a soft sensor developed in this dissertation explored the CART models, particularly LightGBM, RF and XGBoost. For experiment purposes, the CART models have been designed in three ways. The CART models were first developed on a local machine with manual adjustment of the hyperparameters. The results were comparable to recent works in the area, as Table 4.9 shown. Next, the CART models' hyperparameters are automatically adjusted for optimal performance. The optimized CART models show better performance and dominate Table 4.9.

Further analysis shows that the DateTime feature is the most significant factor in predicting Chl-a on the CART models, optimized or not. When DateTime is available, the WQ parameters are ordered from DO, conductivity, turbidity, and temperature on the input feature importance. When DateTime is removed, the WQ precedence changed to conductivity, temperature and turbidity, and DO. The DateTime feature was engineered to extract different temporal information from the WQ parameter data. When DateTime is unavailable, the model had to rely on a more generic WQ correlation. The model could miss sudden spikes and tremors in the Chl-a pattern due to dynamic weather during the monsoon season where the dataset was taken.

Even without DateTime, the optimized CART models perform well, and their Chl-a prediction capabilities are beaten only by the performance of the optimized CART models with DateTime. These experiments show that input feature importance can influence the ML model performance. There is much more to learn about WQ parameter monitoring before deducing if four WQ parameters will always be sufficient to predict Chl-a concentration. The work done in this dissertation is promising; however, more study must be done to see the model performance in various lake situations.

For example, the dataset used in this dissertation focused more on the monsoon seasons. It would be interesting to run the model on a continuous dataset for the entire year. It is also necessary to collect data from other lakes in various areas in Malaysia to learn their characteristics. Monsoon or not, Malaysia has a consistent temperature, about 25-30 degrees Celsius, throughout the year. However, rainfall patterns are not always the same throughout the country, influencing stratification and, eventually, DO level. Nutrients influence the speed of algal growth, which can fluctuate between day and night. Chl-a estimates how much algae is present at the sampling point. Chl-a concentration measured from the water surface also does not reflect algae concentration at deeper levels.

Encouragingly, the Chl-a soft sensor developed in this dissertation solves a critical gap in the literature. The soft sensor shows that live inferencing can be done given the four WQ parameters. The IoT-cloud architecture can take in WQ data from physical sensors or tabulated datasets and pump them to the ML backend for Chl-a inferencing. The Chl-a concentration level is published for logging. Online logging means researchers are no longer limited to past data for analytics. ML model trained on past data may be helpful to predict Chl-a concentration level then. However, the prediction can get less accurate if the training model is not updated. Lessons from the literature include training based on the 2000 to 2004 dataset to predict 2005 to 2010 Chl-a (Su et al., 2015). Another is training based on the 1999 to 2010 dataset, which was carried out in 2018 (Yajima & Derot, 2018). Both works show an R^2 of 0.82 and 0.61, respectively.

In conclusion, this dissertation shows that researchers can use four common WQ parameters to estimate Chl-a concentrations accurately. Additionally, the proposed IoT-cloud architecture can allow the ML selection model to self-update when new batches of WQ parameters are available. The CART models proposed lack recognition in WQ-

related work, and this dissertation shows that they are promising in understanding the nonlinear relationship of WQ parameters. Reducing the number of WQ parameters is significant as it reduces the cost of data collection. The LightGBM proposed also has several technical advantages. The model has a faster training speed and consumes lower memory. Furthermore, the LightGBM is capable of handling large-scale data. The LightGBM is an excellent choice for online inferencing architecture from a computational standpoint.

5.2 Future work

After addressing the gaps, objectives and scopes from this dissertation, realization hits that there are more things to look forward to supporting the water community further. For example, this dissertation leaves out WQ sensors from its scopes. The four WQ parameters of DO, temperature, conductivity, and turbidity can be purchased separately. Their costs are much lower, possibly over a 90% reduction than purchasing a multiparameter sonde. These physical sensors can be combined using a microcontroller and configured for cloud communication. Theoretically, if these physical sensors are considered, Chl-a WQ monitoring costs can be significantly reduced.

With its AutoML feature, Microsoft Azure is a powerful cloud service for training and optimizing CART models' performance. The Azure services are great options to optimize models' hyperparameters without the need for extra computational resources on local machines. The cloud service is not cheap. However, one does not require Azure services after a model is trained and fine-tuned. The model can be used on the IoT-cloud application developed in this dissertation without Azure. When the data logging reaches a particular stage where the model training is due for an update, Azure can be considered

again. The model performance is dependent on the frequency of updates and dataset quality.

Figure 5.1 shows a possible architecture for future work. The physical sensors upload live WQ parameters via the MQTT protocol to the backend for database logging. At the same time, the Chl-a soft sensor developed in this study can infer Chl-a concentration using the four WQ parameters. The Chl-a concentration value can be published on a web application for the user. These live WQ data acquisition and inferencing are valuable for improving WQ monitoring.

This dissertation explores the correlation between Chl-a and four WQ parameters. These WQ parameters represent the water composition and are the most recommended in lake conservation. However, for prediction purposes, weather forecasts can be influential. It may help if the current architecture also considers weather sensors. Like DateTime helped the model's accuracy, engineered features such as weather information may improve model performance. Figure 5.2 shows a possible extension to the IoT architecture, with IoT weather sensors complementing the WQ parameters.

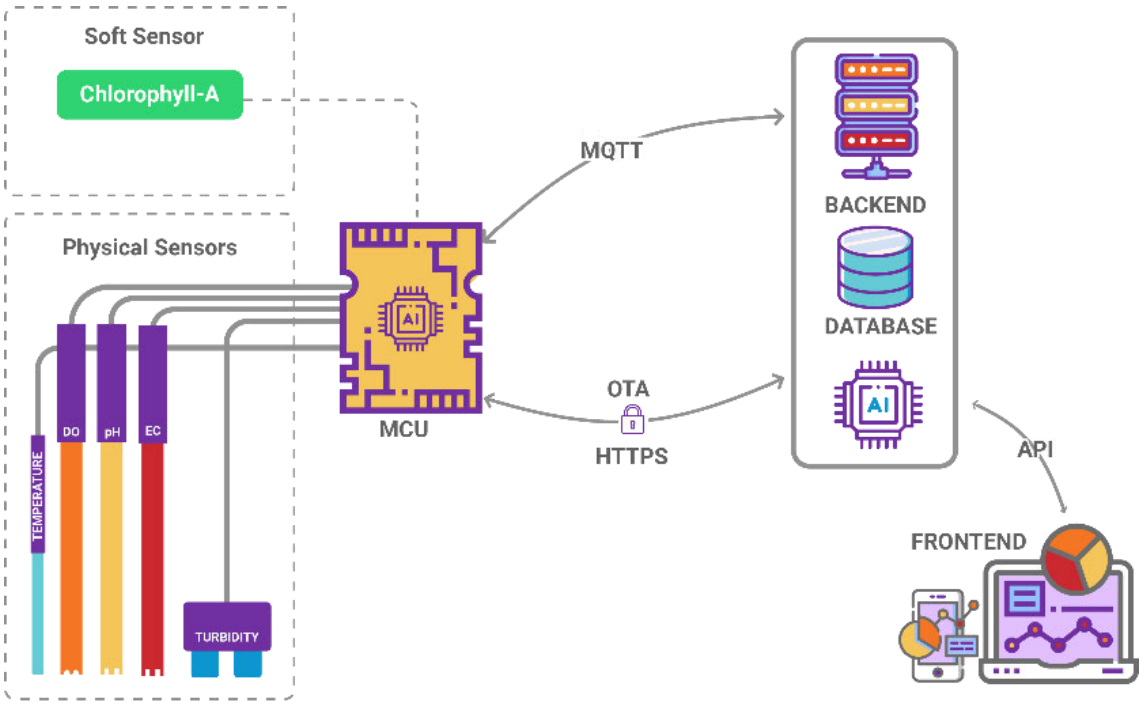

Figure 5.1: Proposed Chl-a soft sensor architecture with physical WQ parameter sensors

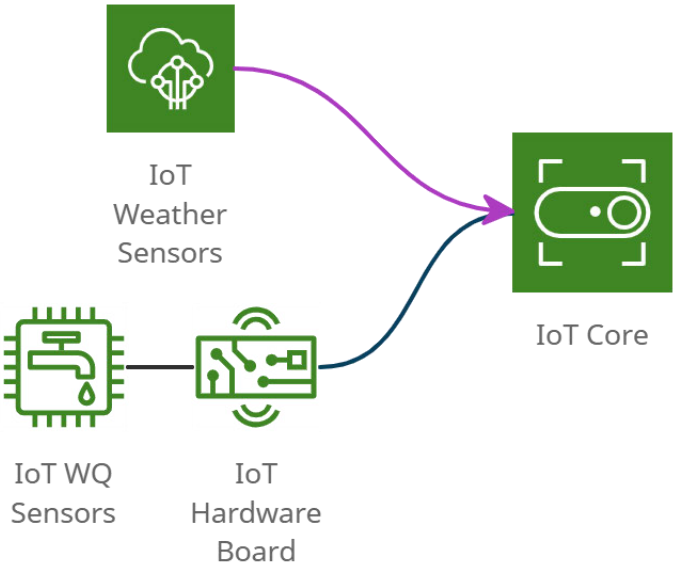

Figure 5.2: Proposed Chl-a soft sensor architecture with IoT weather sensors

Integrating weather sensors alongside the WQ parameters for continuous data collection can further support understanding the nonlinear relationship of the WQ parameters with Chl-a at the lakes. A less tedious and time-savvy approach to data logging and analytics via the Chl-a soft sensor can encourage citizen scientists to perform place-based analysis of their local lakes. A cumulative and centralized effort is required to support researchers in understanding eutrophication and algal bloom behaviour at the lakes.

REFERENCES

Ahmed, A. N., Othman, F. B., Afan, H. A., Ibrahim, R. K., Fai, C. M., Hossain, M. S., ... & Elshafie, A. (2019). Machine learning methods for better water quality prediction. *Journal of Hydrology*, *578*, 124084.

Ahmed, U., Mumtaz, R., Anwar, H., Shah, A. A., Irfan, R., & García-Nieto, J. (2019). Efficient water quality prediction using supervised machine learning. *Water*, *11*(11), 2210.

Ahsan, M. M., Mahmud, M. P., Saha, P. K., Gupta, K. D., & Siddique, Z. (2021). Effect of data scaling methods on machine learning algorithms and model performance. *Technologies*, *9*(3), 52.

Al-Badaii, F., Shuhaimi-Othman, M., & Gasim, M. B. (2013). Water quality assessment of the Semenyih river, Selangor, Malaysia. *Journal of chemistry*, *2013*.

Antylia Scientific Blog Team. (2015, February 16). *How Optical Dissolved Oxygen Meters Work*. Cole-Palmer Blog. https://www.coleparmer.com/blog/2015/02/16/how-optical-dissolved-oxygen-meters-work/

Aronoff, R., Dussuet, A., Erismann, R., Erismann, S., Patiny, L., & Vivar-Rios, C. (2021). Participatory research to monitor lake water pollution. *Ecological Solutions and Evidence*, *2*(3), e12094.

Ashraf, M. A., Maah, M. J., & Yusoff, I. (2010). Water quality characterization of varsity lake, University of Malaya, Kuala Lumpur, Malaysia. *E-Journal of Chemistry*, *7*(S1), S245-S254.

Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. *Computer networks*, *54*(15), 2787-2805.

Azizul, Z. H. (2019). Internal discussion with SPAN and NAHRIM on challenges of lake sampling in Malaysia. Unpublished manuscript.

Barker, B. (2020, October 12). *ISE Sensor vs Wet Chemistry Analyzer for Ammonia/Ammonium.* Xylem. https://www.xylem.com/siteassets/brand/wtw/resources/technical-brochure/wtw-ise-sensor-vs-wet-chemistry-analyzer.pdf

Behmel, S., Damour, M., Ludwig, R., & Rodriguez, M. J. (2016). Water quality monitoring strategies—A review and future perspectives. *Science of the Total Environment*, *571*, 1312-1329.

Blix, K., & Eltoft, T. (2018). Machine learning automatic model selection algorithm for oceanic chlorophyll-a content retrieval. *Remote Sensing*, *10*(5), 775.

Carlson, R. E. (1977). A trophic state index for lakes 1. *Limnology and oceanography*, *22*(2), 361-369.

Castrillo, M., & García, Á. L. (2020). Estimation of high frequency nutrient concentrations from water quality surrogates using machine learning methods. *Water research*, *172*, 115490.

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794).

Chislock, M. F., Doster, E., Zitomer, R. A., & Wilson, A. E. (2013). Eutrophication: causes, consequences, and controls in aquatic ecosystems. *Nature Education Knowledge*, *4*(4), 10.

Cho, H., & Park, H. (2019, October). Merged-LSTM and multistep prediction of daily chlorophyll-a concentration for algal bloom forecast. In IOP Conference Series: Earth and Environmental Science (Vol. 351, No. 1, p. 012020). IOP Publishing.

Choi, J. H., Kim, J., Won, J., & Min, O. (2019, February). Modelling chlorophyll-a concentration using deep neural networks considering extreme data imbalance

and skewness. In 2019 21st International Conference on Advanced Communication Technology (ICACT) (pp. 631-634). IEEE.

Detweiler, C., Ore, J. P., Anthony, D., Elbaum, S., Burgin, A., & Lorenz, A. (2015). Environmental reviews and case studies: bringing unmanned aerial systems closer to the environment. *Environmental Practice*, *17*(3), 188-200.

Du, Z., Qin, M., Zhang, F., & Liu, R. (2018). Multistep-ahead forecasting of chlorophyll a using a wavelet nonlinear autoregressive network. *Knowledge-Based Systems*, *160*, 61-70.

Dunbabin, M., & Grinham, A. (2010, May). Experimental evaluation of an autonomous surface vehicle for water quality and greenhouse gas emission monitoring. In *2010 IEEE International Conference on Robotics and Automation* (pp. 5268-5274). IEEE.

Lake Erie Abloom. Earthobservatory.nasa.gov. (2017). Retrieved 17 January 2022, from https://earthobservatory.nasa.gov/images/91038/lake-erie-abloom.

Esakki, B., Ganesan, S., Mathiyazhagan, S., Ramasubramanian, K., Gnanasekaran, B., Son, B., ... & Choi, J. S. (2018). Design of amphibious vehicle for unmanned mission in water quality monitoring using internet of things. *Sensors*, *18*(10), 3318.

Gafri, H. G. F. (2018). A Study on Water Quality Status of Varsity Lake and Pantai River, Anak Air Batu River in UM Kuala Lumpur, Malaysia and Classify it based on (WQI) Malaysia. *EQA-International Journal of Environmental Quality*, *29*, 51-65.

García-Nieto, P. J., García-Gonzalo, E., Fernández, J. R. A., & Muñiz, C. D. (2020). A New Predictive Model for Evaluating Chlorophyll-a Concentration in Tanes Reservoir by Using a Gaussian Process Regression. *Water Resources Management*, *34*(15), 4921-4941.

GISGeography. (2021, October 29). *Multispectral vs Hyperspectral Imagery Explained*. GIS Geography. https://gisgeography.com/multispectral-vs-hyperspectral-imagery-explained/

Hafeez, S., Wong, M. S., Ho, H. C., Nazeer, M., Nichol, J., Abbas, S., ... & Pun, L. (2019). Comparison of machine learning algorithms for retrieval of water quality indicators in case-II waters: a case study of Hong Kong. *Remote sensing*, *11*(6), 617.

Ho, T. K. (1995, August). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition* (Vol. 1, pp. 278-282). IEEE.

Hunkeler, U., Truong, H. L., & Stanford-Clark, A. (2008, January). MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks. In 2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08) (pp. 791-798). IEEE.

Huo, S., He, Z., Su, J., Xi, B., & Zhu, C. (2013). Using artificial neural network models for eutrophication prediction. *Procedia Environmental Sciences*, *18*, 310-316.

Joslyn, K., & Lipor, J. (2018, December). A Supervised Learning Approach to Water Quality Parameter Prediction and Fault Detection. In *2018 IEEE International Conference on Big Data (Big Data)* (pp. 2511-2514). IEEE.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*, 3146-3154.

Keller, S., Maier, P. M., Riese, F. M., Norra, S., Holbach, A., Börsig, N., ... & Hinz, S. (2018). Hyperspectral data and machine learning for estimating CDOM, chlorophyll a, diatoms, green algae and turbidity. *International journal of environmental research and public health*, *15*(9), 1881.

Koparan, C., Koc, A. B., Privette, C. V., & Sawyer, C. B. (2018). In situ water quality measurements using an unmanned aerial vehicle (UAV) system. *Water*, *10*(3), 264.

Koparan, C., Koc, A. B., Privette, C. V., & Sawyer, C. B. (2019). Autonomous in situ measurements of noncontaminant water quality indicators and sample collection with a UAV. *Water*, *11*(3), 604.

Lee, G., Bae, J., Lee, S., Jang, M., & Park, H. (2016). Monthly chlorophyll-a prediction using neuro-genetic algorithm for water quality management in Lakes. Desalination and Water Treatment, 57(55), 26783-26791.

Leeuw, T., Boss, E. S., & Wright, D. L. (2013). In situ measurements of phytoplankton fluorescence using low cost electronics. *Sensors*, *13*(6), 7872-7883.

Li, J., Abdulmohsin, H. A., Hasan, S. S., Kaiming, L., Al-Khateeb, B., Ghareb, M. I., & Mohammed, M. N. (2019). Hybrid soft computing approach for determining water quality indicator: Euphrates River. *Neural Computing and Applications*, *31*(3), 827-837.

Li, T., Xia, M., Chen, J., Zhao, Y., & De Silva, C. (2017). Automated water quality survey and evaluation using an IoT platform with mobile sensor nodes. *Sensors*, 17(8), 1735.

Li, X., Sha, J., & Wang, Z. L. (2018). Application of feature selection and regression models for chlorophyll-a prediction in a shallow lake. Environmental Science and Pollution Research, 25(20), 19488-19498.

Li, X., Sha, J., & Wang, Z. L. (2017). Chlorophyll-a prediction of lakes with different water quality patterns in China based on hybrid neural networks. *Water*, *9*(7), 524.

Liu, J., Jang, S. S., & Wong, D. S. H. (2016). Developing a soft sensor with online variable selection for industrial multi-mode processes. In *Computer Aided Chemical Engineering* (Vol. 38, pp. 398-403). Elsevier.

Liu, P., Wang, J., Sangaiah, A. K., Xie, Y., & Yin, X. (2019). Analysis and prediction of water quality using LSTM deep neural networks in IoT environment. Sustainability, 11(7), 2058.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533-536.

Liu, Y., Liang, Y., Liu, S., Rosenblum, D. S., & Zheng, Y. (2016). Predicting urban water quality with ubiquitous data. *arXiv preprint arXiv:1610.09462*.

Mamun, M., Kim, J. J., Alam, M. A., & An, K. G. (2020). Prediction of algal chlorophyll-a and water clarity in monsoon-region reservoir using machine learning approaches. *Water*, *12*(1), 30.

Mamun, M., Lee, S. J., & An, K. G. (2018). Temporal and spatial variation of nutrients, suspended solids, and chlorophyll in Yeongsan watershed. *Journal of Asia-Pacific Biodiversity*, *11*(2), 206-216.

Martinez, E., Gorgues, T., Lengaigne, M., Fontana, C., Sauzède, R., Menkes, C., ... & Fablet, R. (2020). Reconstructing global chlorophyll-a variations using a non-linear statistical approach. *Frontiers in Marine Science*, *7*, 464.

NAHRIM (2014) Blueprint for Lake Research and Development in Malaysia. National Hydraulic Research Institute of Malaysia, Seri Kembangan, Malaysia.

NAHRIM. (2009). Desk Study on the Status of Eutrophication of Lakes in Malaysia. *National Hydraulic Research Institute Malaysia*.

National Hydraulic Research Institute of Malaysia (NAHRIM). (2015). National Lake Water Quality Criteria and Standards 2015. Seri Kembangan: NAHRIM.

Nieto, P. G., García-Gonzalo, E., Fernández, J. A., & Muñiz, C. D. (2019). Water eutrophication assessment relied on various machine learning techniques: A case study in the Englishmen Lake (Northern Spain). *Ecological Modelling*, *404*, 91-102.

Ore, J. P., Elbaum, S., Burgin, A., & Detweiler, C. (2015). Autonomous aerial water sampling. *Journal of Field Robotics*, *32*(8), 1095-1113.

Pahlevan, N., Smith, B., Schalles, J., Binding, C., Cao, Z., Ma, R., ... & Stumpf, R. (2020). Seamless retrievals of chlorophyll-a from Sentinel-2 (MSI) and Sentinel-3 (OLCI) in inland and coastal waters: A machine-learning approach. *Remote Sensing of Environment*, *240*, 111604.

Peterson, K. T., Sagan, V., Sidike, P., Hasenmueller, E. A., Sloan, J. J., & Knouft, J. H. (2019). Machine learning-based ensemble prediction of water-quality variables using feature-level and decision-level fusion with proximal remote sensing. *Photogrammetric Engineering & Remote Sensing*, *85*(4), 269-280.

Podnar, G., Dolan, J. M., Low, K. H., & Elfes, A. (2010, March). Telesupervised remote surface water quality sensing. In *2010 IEEE Aerospace Conference* (pp. 1-9). IEEE.

Ruescas, A. B., Mateo-Garcia, G., Camps-Valls, G., & Hieronymi, M. (2018, July). Retrieval of case 2 water quality parameters with machine learning. In *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium* (pp. 124-127). IEEE.

Saab, C., Shahrour, I., & Chehade, F. H. (2017, September). Smart technology for water quality control: Feedback about use of water quality sensors. In *2017 Sensors Networks Smart and Emerging Technologies (SENSET)* (pp. 1-4). IEEE.

Sakai, N., Mohamad, Z. F., Nasaruddin, A., Abd Kadir, S. N., Salleh, M. S. A. M., & Sulaiman, A. H. (2018). Eco-Heart Index as a tool for community-based water quality monitoring and assessment. *Ecological Indicators, 91*, 38-46.

Sharip, Z., & Suratman, S. (2017). Formulating specific water quality criteria for lakes: A Malaysian perspective. *Water Quality; Tutu, H., Ed.; IntechOpen: Rijeka, Croatia*, 293-313.

Sharip, Z., & Yusop, Z. (2007, August). National overview: the status of lakes eutrophication in Malaysia. In *Colloquium on Lakes and Reservoir Management: Status and Issues* (pp. 2-3). Putrajaya: NAHRIM.

Sharip, Z., Zaki, A. T., Shapai, M. A., Suratman, S., & Shaaban, A. J. (2014). Lakes of Malaysia: Water quality, eutrophication and management. *Lakes & Reservoirs: Research & Management*, *19*(2), 130-141.

Shehhi, M. R. A., & Kaya, A. (2020). Time series and machine learning to forecast the water quality from satellite data. *arXiv preprint arXiv:2003.11923*.

Shin, Y., Kim, T., Hong, S., Lee, S., Lee, E., Hong, S., ... & Heo, T. Y. (2020). Prediction of chlorophyll-a concentrations in the Nakdong River using machine learning methods. *Water*, *12*(6), 1822.

Silveira Kupssinskü, L., Thomassim Guimarães, T., Menezes de Souza, E., C Zanotta, D., Roberto Veronez, M., Gonzaga, L., & Mauad, F. F. (2020). A method for chlorophyll-a and suspended solids prediction through remote sensing and machine learning. *Sensors*, *20*(7), 2125.

Siyang, S., & Kerdcharoen, T. (2016, June). Development of unmanned surface vehicle for smart water quality inspector. In *2016 13th International conference on electrical engineering/electronics, computer, telecommunications and information technology (ECTI-CON)* (pp. 1-5). IEEE.

Su, J., Wang, X., Zhao, S., Chen, B., Li, C., & Yang, Z. (2015). A structurally simplified hybrid model of genetic algorithm and support vector machine for prediction of chlorophyll a in reservoirs. *Water*, *7*(4), 1610-1627.

Syariz, M. A., Lin, C. H., Nguyen, M. V., Jaelani, L. M., & Blanco, A. C. (2020). WaterNet: A convolutional neural network for chlorophyll-a concentration retrieval. *Remote Sensing*, *12*(12), 1966.

Tian, W., Liao, Z., & Wang, X. (2019). Transfer learning for neural network model in chlorophyll-a dynamics prediction. *Environmental Science and Pollution Research*, *26*(29), 29857-29871.

Tian, W., Liao, Z., & Zhang, J. (2017). An optimization of artificial neural network model for predicting chlorophyll dynamics. *Ecological Modelling*, *364*, 42-52.

Tuna, G., Arkoc, O., & Gulez, K. (2013). Continuous monitoring of water quality using portable and low-cost approaches. *International Journal of Distributed Sensor Networks*, *9*(6), 249598.

Wiranto, G., Mambu, G. A., Hermida, I. D. P., & Widodo, S. (2015, August). Design of online data measurement and automatic sampling system for continuous water quality monitoring. In *2015 IEEE International Conference on Mechatronics and Automation (ICMA)* (pp. 2331-2335). IEEE.

Wurtsbaugh, W. A., Paerl, H. W., & Dodds, W. K. (2019). Nutrients, eutrophication and harmful algal blooms along the freshwater to marine continuum. *Wiley Interdisciplinary Reviews: Water*, *6*(5), e1373

Xia, F., Yang, L. T., Wang, L., & Vinel, A. (2012). Internet of things. *International journal of communication systems*, *25*(9), 1101.

Yang, T. H., Hsiung, S. H., Kuo, C. H., Tsai, Y. D., Peng, K. C., Hsieh, Y. C., ... & Kuo, C. (2018, April). Development of unmanned surface vehicle for water quality

monitoring and measurement. In *2018 IEEE International Conference on Applied System Invention (ICASI)* (pp. 566-569). IEEE.

Yajima, H., & Derot, J. (2018). Application of the Random Forest model for chlorophyll-a forecasts in fresh and brackish water bodies in Japan, using multivariate long-term databases. Journal of Hydroinformatics, 20(1), 206-220.

Yi, H. S., Park, S., An, K. G., & Kwak, K. C. (2018). Algal bloom prediction using extreme learning machine models at artificial weirs in the Nakdong River, Korea. *International journal of environmental research and public health*, *15*(10), 2078.

Yussof, F. N., Maan, N., & Md Reba, M. N. (2021). LSTM Networks to Improve the Prediction of Harmful Algal Blooms in the West Coast of Sabah. *International Journal of Environmental Research and Public Health*, *18*(14), 7650.

Zeng, L., & Li, D. (2015). Development of in situ sensors for chlorophyll concentration measurement. *Journal of Sensors*, *2015*.

Zhang, F., Wang, Y., Cao, M., Sun, X., Du, Z., Liu, R., & Ye, X. (2016). Deep-learning-based approach for prediction of algal blooms. *Sustainability*, *8*(10), 1060.

Zhu, C., Liu, X., Chen, H., & Tian, X. (2018). Automatic cruise system for water quality monitoring. *International Journal of Agricultural and Biological Engineering*, *11*(4), 244-250.

APPENDIX A

PYTHON SOURCE CODE

This study contains ten python programs that encompass preprocessing and feature engineering of the data, developing, training, and testing the ML models, setting up AutoML, uploading data via MQTT, and execution script for Lambda function in AWS.

A.1 Dropping Missing Values and Outlier

Python script that reads excel file and converts datetime to pandas datetime format and removes missing values and outliers.

```
#importing all libraries

import pandas as pd
import datetime
import matplotlib.pyplot as plt
#import seaborn as sns
from scipy import stats
import numpy as np

#Reading excel file and converting date time to pandas datetime format

df1 = pd.read_excel('dataset/lake_dataset_raw.xlsx', converters={'Date
    Time': pd.to_datetime})
df1 = df1.rename(columns={'Date Time' : 'datetime', 'Actual
    Conductivity (µS/cm) (624571)' : 'actual_conductivity','Specific
    Conductivity (µS/cm) (624571)' : 'specific_conductivity','Total
    Dissolved Solids (ppt)624571': 'total_dissolved_solids','RDO
    Concentration (mg/L) (543925)': 'do_concentration', 'RDO Saturation
    (%Sat) (543925)': 'do_saturation',       'Oxygen Partial Pressure
    (Torr) (543925)':'oxygen_partial_pressure', 'Turbidity (NTU)
    (607780)': 'turbidity','Chlorophyll-a Fluorescence (RFU) (622895)':
    'Chl-a_fluorescence', 'Chlorophyll-a Concentration (µg/L)
    (622895)': 'Chl-a_concentration','Temperature (°C) (606143)':
    'temperature'})

#dropping non-useful columns
df1 = df1.drop(columns= ['specific_conductivity', 'do_saturation',
    'oxygen_partial_pressure', 'Chl-a_fluorescence'])

#setting datetime as the index and sorting the dataframe
df1 = df1.set_index('datetime').sort_index()

#Missing value info
missing_summary= df1.isna().sum()

#Plotting Missing values plot
missing_summary.plot(kind='bar')
plt.title('Missing values for the five parameters')
plt.show()
```

```
#Dropping Missing Values
df2 = df1.dropna()

#calculating z-score and removing outliers
abs_z_scores = np.abs(stats.zscore(df2))
filtered_entries= (abs_z_scores<3).all(axis=1)
new_df = df2[filtered_entries]

#Saving cleaned dataset as a csv file
new_df.to_csv('wq_dataset.csv')
```

A.2 ANN Development, Training & Testing

Python script that splits the clean dataset into training and testing datasets and develops and trains the ANN model. ANN model is then evaluated with the test dataset and performance metrics were printed. The predicted Chl-a values were also plotted along with the actual Chl-a values from the test set. Finally, the trained model was saved.

```
#importing all libraries

import tensorflow as tf
import keras
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, r2_score

from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping
from keras.regularizers import l2
from math import sqrt

#Importing the Dataset

Dataset = pd.read_csv('wq_dataset.csv', header=0, index_col=0)
#Renaming Columns
dataset = dataset.rename(columns={'actual_conductivity':'cond',
'total_dissolved_solids':'tds', 'do_concentration':'do',
'turbidity':'turb', 'Chl-a_concentration':'Chl-a',
'temperature':'temp'})
#Rearranging Columns
dataset = dataset[['temp', 'cond', 'do', 'tds', 'turb', 'Chl-a']]
#dropping TDS
dataset = dataset.drop('tds', axis=1)
#Sorting the dataset based on the datetime
dataset.sort_index()
dataset.head()

#Splitting data into Train and Test datasets (80-20 split)

#Selecting entire dataset
dataset_trimmed = dataset[:]
#Training Set
```

```
train_dataset_rand = dataset_trimmed.sample(frac=0.8) #80% split
#Testing Set
test_dataset_rand = dataset_trimmed.drop(train_dataset_rand.index)
#Train Labels
train_labels_rand = train_dataset_rand.pop('Chl-a')
#Test Labels
test_labels_rand = test_dataset_rand.pop('Chl-a')

#Data Preprocessing with MinMaxScaler

train_rand_val = train_dataset_rand.values
test_rand_val = test_dataset_rand.values
scaler = MinMaxScaler(feature_range=(-1,1))
scaled_train_rand = scaler.fit_transform(train_rand_val)
scaled_test_rand = scaler.fit_transform(test_rand_val)


#Building the model

model = Sequential()
#input layer
model.add(Dense(32, kernel_initializer= 'normal',
input_dim=len(train_dataset_rand.keys()), activation="tanh"))
#hidden layers
model.add(Dense(64, kernel_initializer= 'normal',
activation="sigmoid"))
model.add(Dense(8, kernel_initializer= 'normal', activation="tanh"))
#Output layer with a single neuron with a linear activation function.
model.add(Dense(1, activation="linear"))

# The model is initialized with the Adam optimizer and then it is
compiled.
model.compile(loss='mae', optimizer= 'adam', metrics=['mae', 'mse'])

#Checking model summary before training
model.summary()

#Early stopping condition based on monitoring validation loss
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss',
patience=200)

#Trainig the model on the training dataset
history = model.fit(scaled_train_rand, train_labels_rand,
validation_data=(scaled_test_rand, test_labels_rand), epochs=10000,
batch_size=128, verbose=2, shuffle=True, callbacks=[early_stop])

#Code to monitor the training process
hist = pd.DataFrame(history.history)
hist['epoch'] = history.epoch
hist.tail()

#Function to plot the training loss
def plot_history(history):
    hist = pd.DataFrame(history.history)
    hist['epoch'] = history.epoch

    plt.figure()
    plt.xlabel('Epoch')
    plt.ylabel('Mean Abs Error [Chl]')
    plt.plot(hist['epoch'], hist['mae'],
            label='Train_Error')
    plt.plot(hist['epoch'], hist['val_mae'],
            label='Val Error')
    plt.legend()
```

```
    # plt.ylim([0,20])

     plt.figure()
     plt.xlabel('Epoch')
     plt.ylabel('Mean Square Error [$Chl^2$]')
     plt.plot(hist['epoch'], hist['mse'],
              label='Train_Error')
     plt.plot(hist['epoch'], hist['val_mse'],
              label='Val Error')
     plt.legend()
     #plt.ylim([0,100])

plot_history(history)

#Printing out performance results

loss, mae, mse = model.evaluate(scaled_test_rand, test_labels_rand,
verbose=0)
print("Testing set Mean Abs Error: {:5.2f} Chlorophyll".format(mae))
yhat = model.predict(scaled_test_rand)
yhat_trn = model.predict(scaled_train_rand)
# calculate RMSE
rmse = sqrt(mean_squared_error(test_labels_rand, yhat))
print('Test RMSE: %.3f' % rmse)
rmse = sqrt(mean_squared_error(train_labels_rand, yhat_trn))
print('Train RMSE: %.3f' % rmse)
print('Test R2 Score: ', r2_score(test_labels_rand, yhat))
print('Train R2 Score: ', r2_score(train_labels_rand, yhat_trn))

#Plotting the timeseries of test data and predicted values

yhat = pd.DataFrame(model.predict(scaled_test_rand))
y_actual = pd.DataFrame(test_labels_rand)
plt.figure(figsize=(12,5))
plt.ylabel('Chl-a Concentration (µg/L)')
plt.plot(y_actual, color='blue', label = 'Actual')
plt.plot(yhat, color='red', label = 'Predicted')
ax = plt.gca()
ax.axes.xaxis.set_visible(False)
plt.xlabel('February to March')
plt.legend()
plt.show()

#Save Model for later use
model.save('ann_wq')
```

A.3 Date-Time Feature Engineering

Python script that converts the pandas datetime into datetime features for the CART models.

```
#importing all libraries

import pandas as pd
import numpy as np
import datetime as dt

#Importing the Dataset
```

```
dataset = pd.read_csv('wq_dataset.csv', header=0,
    parse_dates=['datetime'])

#Transforming Pandas Datetime into distinct features

dataset['year'] = dataset['datetime'].dt.year
dataset['month'] = dataset['datetime'].dt.month
dataset['day'] = dataset['datetime'].dt.day
dataset['day_of_week'] = dataset['datetime'].dt.dayofweek
dataset['day_of_year'] = dataset['datetime'].dt.dayofyear
dataset['quarter'] = dataset['datetime'].dt.quarter
dataset['hour'] = dataset['datetime'].dt.hour
dataset['minute'] = dataset['datetime'].dt.minute
dataset['second'] = dataset['datetime'].dt.second

#Adding the new features to the dataframe
dataset = dataset[['datetime', 'year', 'month', 'day', 'day_of_week',
    'day_of_year', 'quarter', 'hour', 'minute', 'second', 'temp',
    'cond', 'do', 'turb', 'Chl-a']]

#Dropping the datetime column
dataset = dataset.drop(['datetime'], axis=1)

#Exporting the feature engineered dataset
dataset.to_csv("feature_engineered_wq_dataset.csv")
```

A.4 LightGBM Development, Training & Testing without AutoML

Python script for the development, training and testing of LightGBM model without hyperparameter tuning. Note: LightGBM requires separate installation before importing.

```
#importing all libraries

import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from scipy.stats import norm
import lightgbm as lgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score,
    mean_absolute_error
from sklearn.model_selection import cross_val_score, cross_val_predict
from sklearn import preprocessing
from sklearn.preprocessing import MaxAbsScaler
from lightgbm import LGBMRegressor
from sklearn.model_selection import RepeatedKFold
from numpy import mean
from numpy import std

#importing datasets
train_dataset = pd.read_csv('feature_engineered_train_dataset.csv',
    header=0)
#Dropping unnamed column
train_dataset = train_dataset.drop(labels= 'Unnamed: 0' , axis=1)

#Preparing Input Features and Prediction Labels
```

```
train_X = train_dataset.iloc[:,0:13]
train_Y = train_dataset.iloc[:, 13]
test_X = test_dataset.iloc[:,0:13]
test_Y = test_dataset.iloc[:, 13]

#Preprocessing with MaxAbsScaler
MAScaler = MaxAbsScaler()
train_X_MaxAbs = MAScaler.fit_transform(train_X)
test_X_MaxAbs = MAScaler.fit_transform(test_X)

#Model Training
model = LGBMRegressor()
#Setting up 10-fold Cross-Validation
cv = RepeatedKFold(n_splits=10, n_repeats=3)
n_scores = cross_val_score(model, train_X_MaxAbs, train_Y,
    scoring='neg_root_mean_squared_error', cv=cv)

#Initiate training
model.fit(train_X_MaxAbs, train_Y)

#Assessing the performance metrics
yhat_train = model.predict(train_X_MaxAbs)
yhat_test = model.predict(test_X_MaxAbs)

#Print Performance Metrics for Training set
print('Train R2: %f' % (r2_score(train_Y, yhat_train)))
print('Train MAE: %f' % (mean_absolute_error(train_Y, yhat_train)))
print('Train RMSE: %f' % (mean_squared_error(train_Y, yhat_train,
    squared=False)))

#Print Performance Metrics for Testing set
print('Test R2: %f' % (r2_score(test_Y, yhat_test)))
print('Test MAE: %f' % (mean_absolute_error(test_Y, yhat_test)))
print('Test RMSE: %f' % (mean_squared_error(test_Y, yhat_test,
    squared=False)))

#Plotting the predicted vs actual on test set
plt.figure(figsize=(12,5))
plt.ylabel('Chl-a Concentration (µg/L)')
plt.plot(test_Y, color='blue', label = 'Actual')
plt.plot(yhat_test, color='red', label = 'Predicted')
ax = plt.gca()
ax.axes.xaxis.set_visible(False)
plt.legend()
plt.show()
```

A.5 XGBoost Development, Training & Testing without AutoML

Python script for the development, training and testing of XGBoost model without hyperparameter tuning. Note: XGBoost requires separate installation before importing.

```
#importing all libraries
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from scipy.stats import norm
import lightgbm as lgb
from sklearn.model_selection import train_test_split
```

```
from sklearn.metrics import mean_squared_error, r2_score,
    mean_absolute_error
from sklearn.model_selection import cross_val_score, cross_val_predict
from sklearn import preprocessing
from sklearn.preprocessing import MaxAbsScaler
from xgboost import XGBRegressor
from sklearn.model_selection import RepeatedKFold
from numpy import mean
from numpy import std

#importing datasets
train_dataset = pd.read_csv('feature_engineered_train_dataset.csv',
    header=0)
#Dropping unnamed column
train_dataset = train_dataset.drop(labels= 'Unnamed: 0' , axis=1)

#Preparing Input Features and Prediction Labels
train_X = train_dataset.iloc[:,0:13]
train_Y = train_dataset.iloc[:, 13]
test_X = test_dataset.iloc[:,0:13]
test_Y = test_dataset.iloc[:, 13]

#Preprocessing with MaxAbsScaler
MAScaler = MaxAbsScaler()
train_X_MaxAbs = MAScaler.fit_transform(train_X)
test_X_MaxAbs = MAScaler.fit_transform(test_X)

#Model Training
model = XGBRegressor(tree_method= 'auto')

#Setting up 10-fold Cross-Validation
cv = RepeatedKFold(n_splits=10, n_repeats=3)
n_scores = cross_val_score(model, train_X_MaxAbs, train_Y,
    scoring='neg_root_mean_squared_error', cv=cv)

#Initiate training
model.fit(train_X_MaxAbs, train_Y)


#Assessing the performance metrics
yhat_train = model.predict(train_X_MaxAbs)
yhat_test = model.predict(test_X_MaxAbs)
#Print Performance Metrics for Training set
print('Train R2: %f' % (r2_score(train_Y, yhat_train)))
print('Train MAE: %f' % (mean_absolute_error(train_Y, yhat_train)))
print('Train RMSE: %f' % (mean_squared_error(train_Y, yhat_train,
    squared=False)))

#Print Performance Metrics for Testing set
print('Test R2: %f' % (r2_score(test_Y, yhat_test)))
print('Test MAE: %f' % (mean_absolute_error(test_Y, yhat_test)))
print('Test RMSE: %f' % (mean_squared_error(test_Y, yhat_test,
    squared=False)))

#Plotting the predicted vs actual on test set
plt.figure(figsize=(12,5))
plt.ylabel('Chl-a Concentration (µg/L)')
plt.plot(test_Y, color='blue', label = 'Actual')
plt.plot(yhat_test, color='red', label = 'Predicted')
ax = plt.gca()
ax.axes.xaxis.set_visible(False)
plt.legend()
plt.show()
```

A.6 RF Development, Training & Testing without AutoML

Python script for the development, training and testing of RF model without hyperparameter tuning.

```
#importing all libraries
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from scipy.stats import norm
import lightgbm as lgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score,
    mean_absolute_error
from sklearn.model_selection import cross_val_score, cross_val_predict
from sklearn import preprocessing
from sklearn.preprocessing import MinMaxScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import RepeatedKFold
from numpy import mean
from numpy import std

#importing datasets
train_dataset = pd.read_csv('feature_engineered_train_dataset.csv',
    header=0)
#Dropping unnamed column
train_dataset = train_dataset.drop(labels= 'Unnamed: 0' , axis=1)

#Preparing Input Features and Prediction Labels
train_X = train_dataset.iloc[:,0:13]
train_Y = train_dataset.iloc[:, 13]
test_X = test_dataset.iloc[:,0:13]
test_Y = test_dataset.iloc[:, 13]

#Preprocessing with MinMaxScaler
MMScaler = MinMaxScaler(feature_range=(-1,1))
train_X_MinMax = MMScaler.fit_transform(train_X)
test_X_MinMax = MMScaler.fit_transform(test_X)

#Model Training
model = RandomForestRegressor()
cv = RepeatedKFold(n_splits=10, n_repeats=3)
n_scores = cross_val_score(model, train_X_MinMax, train_Y,
scoring='neg_root_mean_squared_error', cv=cv)
#Initiate training
model.fit(train_X_MinMax, train_Y)

#Assessing the performance metrics
yhat_train = model.predict(train_X_MinMax)
yhat_test = model.predict(test_X_MinMax)

#Print Performance Metrics for Training set
print('Train R2: %f' % (r2_score(train_Y, yhat_train)))
print('Train MAE: %f' % (mean_absolute_error(train_Y, yhat_train)))
print('Train RMSE: %f' % (mean_squared_error(train_Y, yhat_train,
    squared=False)))

#Print Performance Metrics for Testing set
print('Test R2: %f' % (r2_score(test_Y, yhat_test)))
```

```
print('Test MAE: %f' % (mean_absolute_error(test_Y, yhat_test)))
print('Test RMSE: %f' % (mean_squared_error(test_Y, yhat_test,
    squared=False)))

#Plotting the predicted vs actual on test set
plt.figure(figsize=(12,5))
plt.ylabel('Chl-a Concentration (µg/L)')
plt.plot(test_Y, color='blue', label = 'Actual')
plt.plot(yhat_test, color='red', label = 'Predicted')
ax = plt.gca()
ax.axes.xaxis.set_visible(False)
plt.legend()
plt.show()
```

A.7 AutoML Setup for CART Model Hyperparameter Tuning

Python script for the setting up AutoML experiment to tune hyperparameters of the CART models. Note: Azure subscription is required for this step. With the azure account, workspace and resource group needs to be created before executing this code in Azure Notebooks. Dataset needs to be uploaded on the Datasets section of the Azure Machine Learning Studio.

```
#importing all libraries
import pandas as pd
from azureml.core import Workspace, Dataset
from datetime import datetime
import logging
from azureml.train.automl import AutoMLConfig
from azureml.core.experiment import Experiment
from azureml.widgets import RunDetails

#Setting up Azure Workspace
subscription_id = 'id' #enter subscription id of Azure account
resource_group = 'group_name' #enter resource group name
workspace_name = 'workspace_name' #enter workspace name
workspace = Workspace(subscription_id, resource_group, workspace_name)

#Configure Workspace
from azureml.core.workspace import Workspace
ws = Workspace.from_config()

#Importing Dataset
dataset = Dataset.get_by_name(workspace, name='wq-all-train-data')
df = dataset.to_pandas_dataframe()
df = df.reset_index()
df = df.drop(labels='index', axis=1)

#Define training settings
automl_settings = {
    "iteration_timeout_minutes": 30,
    "experiment_timeout_hours": 1,
    "enable_early_stopping": True,
    "primary_metric": 'normalized_root_mean_squared_error',
    "featurization": 'auto',
    "verbosity": logging.INFO,
    "n_cross_validations": 10
}
```

```
#Configuring AutoML settings
automl_config = AutoMLConfig(task='regression',

debug_log='wq_22_automated_ml_errors.log',
                             training_data=df,
                             label_column_name="Chl-a",
                             **automl_settings)

#Train the automatic regression model
experiment = Experiment(ws, "wq-22-jupy-train")
local_run = experiment.submit(automl_config, show_output=True)

#Explore the results
RunDetails(local_run).show()

#Retrieve the best model
best_run, fitted_model = local_run.get_output()
print(best_run)
print(fitted_model)
```

A.8 Testing CART Models after Hyperparameter Optimization

Hyperparameter optimized CART models can be downloaded from Azure in .pkl or pickle format. Pickle exports the trained ML model so that it can be used later. This Python script shows how to use .pkl ML models to carry out inferencing on unseen test data.

```
#importing all libraries
import pickle
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, r2_score,
    mean_absolute_error

#Importing the test Dataset
dataset_full_test = pd.read_csv('azure_wq_test_entire_dataset.csv')
dataset_full_test = dataset_full_test[['datetime','temp', 'cond',
    'do', 'turb', 'Chl-a']]
dataset_full_test.sort_index()
print(dataset_full_test.shape)

#Preparing Input & Output Features
X = dataset_full_test[:]
Y = X.pop('Chl-a')

#Loading LightGBM Model
filename = 'lgbm_model.pkl'
entire_ds_model = pickle.load(open(filename, 'rb'))

#Testing LightGBM model on test dataset
result = entire_ds_model.score(X,Y)
predicted_Y = pd.DataFrame(entire_ds_model.predict(X))
print(result)
print('R2: %f' % (r2_score(Y, predicted_Y)))
print('MAE: %f' % (mean_absolute_error(Y, predicted_Y)))
print('MAE: %f' % (mean_squared_error(Y, predicted_Y, squared=False)))
```

```
#Plotting the predicted vs actual on test set
plt.figure(figsize=(12,5))
plt.ylabel('Chl-a Concentration (µg/L)')
plt.plot(Y, color='blue', label = 'Actual')
plt.plot(predicted_Y, color='red', label = 'Predicted')
ax = plt.gca()
ax.axes.xaxis.set_visible(False)
plt.legend()
plt.show()

#Loading RF model
filename = 'rf_model.pkl'
entire_ds_model = pickle.load(open(filename, 'rb'))

#Testing RF model on test dataset
result = entire_ds_model.score(X,Y)
predicted_Y = pd.DataFrame(entire_ds_model.predict(X))
print(result)
print('R2: %f' % (r2_score(Y, predicted_Y)))
print('MAE: %f' % (mean_absolute_error(Y, predicted_Y)))
print('MAE: %f' % (mean_squared_error(Y, predicted_Y, squared=False)))

#Plotting the predicted vs actual on test set
plt.figure(figsize=(12,5))
plt.ylabel('Chl-a Concentration (µg/L)')
plt.plot(Y, color='blue', label = 'Actual')
plt.plot(predicted_Y, color='red', label = 'Predicted')
ax = plt.gca()
ax.axes.xaxis.set_visible(False)
plt.legend()

#Loading XGBoost model
filename = 'xgboost_model.pkl'
entire_ds_model = pickle.load(open(filename, 'rb'))

#Testing XGBoost model on test dataset
result = entire_ds_model.score(X,Y)
predicted_Y = pd.DataFrame(entire_ds_model.predict(X))
print(result)
print('R2: %f' % (r2_score(Y, predicted_Y)))
print('MAE: %f' % (mean_absolute_error(Y, predicted_Y)))
print('MAE: %f' % (mean_squared_error(Y, predicted_Y, squared=False)))

#Plotting the predicted vs actual on test set
plt.figure(figsize=(12,5))
plt.ylabel('Chl-a Concentration (µg/L)')
plt.plot(Y, color='blue', label = 'Actual')
plt.plot(predicted_Y, color='red', label = 'Predicted')
ax = plt.gca()
ax.axes.xaxis.set_visible(False)
plt.legend()
```

A.9 Sending Data from Dataset File to IoT Core

Python Script sends each row of the dataset to AWS IoT Core using MQTT Protocol. Note: AWS account is required to carry out this step. A 'thing' needs to be created in AWS IoT Core, and the certificate file, private key file, and Amazon Root CA file needs to be downloaded from AWS IoT Core for this step. The endpoint address information also needs to be collected from AWS IoT Core.

```
#importing all libraries
from AWSIoTPythonSDK.MQTTLib import AWSIoTMQTTClient
import logging
import time
import argparse
import json
from colorama import Fore, Back, Style

#IoT Credentials Setup
ENDPOINT = "xxxx.amazonaws.com" #specify endpoint address
CLIENT_ID = "test_client_1"     #specify thing name
PATH_TO_CERT = "certificates/certificate.pem.crt" #specify local file
address
PATH_TO_KEY = "certificates/private.pem.key"      #specify local file
address
PATH_TO_ROOT = "certificates/AmazonRootCA1.pem"   #specify local file
address
PUB_TOPIC = "test_topic"        #specify MQTT Topic to publish to
PORT = 8883                     #port for MQTTS

# Puback callback. Notificaiton for Successful Publish
def customPubackCallback(mid):
    print("Received PUBACK packet id: ")
    print(mid)
    print("++++++++++++++\n\n")

# Configure logging
logger = logging.getLogger("AWSIoTPythonSDK.core")
logger.setLevel(logging.DEBUG)
streamHandler = logging.StreamHandler()
formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s
- %(message)s')
streamHandler.setFormatter(formatter)
logger.addHandler(streamHandler)

# Init AWSIoTMQTTClient
myAWSIoTMQTTClient = None
myAWSIoTMQTTClient = AWSIoTMQTTClient(CLIENT_ID)
myAWSIoTMQTTClient.configureEndpoint(ENDPOINT, PORT)
myAWSIoTMQTTClient.configureCredentials(PATH_TO_ROOT, PATH_TO_KEY,
PATH_TO_CERT)

# AWSIoTMQTTClient connection configuration
myAWSIoTMQTTClient.configureAutoReconnectBackoffTime(1, 32, 20)
myAWSIoTMQTTClient.configureOfflinePublishQueueing(-1)
myAWSIoTMQTTClient.configureDrainingFrequency(2)
myAWSIoTMQTTClient.configureConnectDisconnectTimeout(10)
myAWSIoTMQTTClient.configureMQTTOperationTimeout(5)
myAWSIoTMQTTClient.onMessage = customOnMessage
```

```
# Connect and subscribe to AWS IoT
myAWSIoTMQTTClient.connect()
time.sleep(2) #delay to establish connection

#Importing the test Dataset
dataset = pd.read_csv('wq_dataset.csv')

for index, row in dataset.iterrows():
    data = { "datetime" : row['datetime'],
         "temp" : row['temp'],
         "cond" : row['cond'],
         "do":  row['do'],
         "turb": row['turb']

    }

    #Conversion to JSON done by dumps() function
    msg = json.dumps(data)
    #Publishing the data row to the MQTT Topic
    myAWSIoTMQTTClient.publishAsync(PUB_TOPIC, msg, 1,
    ackCallback=customPubackCallback)
```

A.10 Sending Data from IoT Core to AzureML using Lambda Function

Python Script for Lambda function that parses MQTT message and call the web service endpoint of Azure that hosts the ML model for real time inferencing. This step requires AzureML subscription. No SSL and API keys/authentication tokens were used for this experiment. Security layers are recommended for production deployment.

```
#importing all libraries
import urllib.request
import json
import os
import ssl

def allowSelfSignedHttps(allowed):
    # bypass the server certificate verification on client side
    if allowed and not os.environ.get('PYTHONHTTPSVERIFY', '') and
getattr(ssl, '_create_unverified_context', None):
        ssl._create_default_https_context =
ssl._create_unverified_context

allowSelfSignedHttps(True) # this line is needed if you use self-
signed certificate in your scoring service.

# Request data goes here
data = {
    "data":
    [
        {
            #incoming json data goes here
        },
    ],
}

#Conversion to JSON done by dumps() function
body = str.encode(json.dumps(data))
```

```
url = 'http://xxxx' #Specify the endpoint for ML inferencing
api_key = '' # Replace this with the API key for the web service
headers = {'Content-Type':'application/json', 'Authorization':('Bearer
'+ api_key)}

#HTTP request
req = urllib.request.Request(url, body, headers)
try:
    response = urllib.request.urlopen(req)
    result = response.read()
    print(result)
except urllib.error.HTTPError as error:
    print("The request failed with status code: " + str(error.code))

    # Print the headers - they include the requert ID and the
timestamp, which are useful for debugging the failure
    print(error.info())
    print(json.loads(error.read().decode("utf8", 'ignore')))
```