

**DEVELOPMENT OF BIOCHAR YIELD PREDICTION MODEL FROM FOOD
WASTE PYROLYSIS USING EXPLAINABLE ARTIFICIAL INTELLIGENCE
ALGORITHM**



اونيورسيتي تیکنیکل ملیسیا ملاک

SUREND A/L ESVARAN

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

DEVELOPMENT OF BIOCHAR YIELD PREDICTION MODEL FROM FOOD
WASTE PYROLYSIS USING EXPLAINABLE ARTIFICIAL INTELLIGENCE
ALGORITHM



This report is submitted in partial fulfillment of the requirements for the
Bachelor of [Computer Science (Software Development)] with Honours.

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

DECLARATION


I hereby declare that this project report entitled
**DEVELOPMENT OF BIOCHAR YIELD PREDICTION MODEL FROM FOOD
WASTE PYROLYSIS USING EXPLAINABLE ARTIFICIAL INTELLIGENCE
ALGORITHM**

is written by me and is my own effort and that no part has been plagiarized
without citations.

STUDENT : SUREND A/L ESVARAN Date : 05/06/2024
([NAME OF STUDENT])

اونيورسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

I hereby declare that I have read this project report and found
this project report is sufficient in term of the scope and quality for the award of
Bachelor of [Computer Science (Software Development)] with Honours.

SUPERVISOR :  Dr. Yogan Jaya Kumar Date : 30/8/2024
([NAME OF THE SUPERVISOR])

DEDICATION

To beloved God,

With deep gratitude and reverence, I dedicate this work to You. Your divine guidance, strength, and wisdom have been my constant companions throughout this journey. Your blessings have given me the courage and resilience to overcome challenges and achieve my goals. I offer this achievement as a humble token of my faith and thankfulness for Your endless grace and love.

To my divinely Parents,

Your unwavering love, encouragement, and sacrifices have been the bedrock of my journey. Your support and guidance have shaped me into the person I am today. This work is a testament to your belief in me and your relentless pursuit of excellence. I dedicate this achievement to you both, with heartfelt gratitude and love.

اونيورسيتي تيكنيكل مليسيا ملاك
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to En. Yogan A/L Jaya Kumar for his invaluable assistance and guidance throughout the project titled "Development of Biochar Yield Prediction Model from Food Waste Pyrolysis Using Explainable Artificial Intelligence Algorithm". His expertise and support were instrumental in the successful completion of this work.

I am also deeply grateful to my beloved parents for their unwavering support and motivation. Their encouragement has been a constant source of strength and inspiration for me during this journey.

Additionally, I would like to thank my friends and colleagues for their insightful feedback and camaraderie, which have enriched my learning experience.

Lastly, I extend my appreciation to the faculty members and staff at University Technical Malaysia Melaka for providing the necessary resources and a conducive environment for my research.

Thank you all for your contributions to this achievement.

ABSTRACT

The collection of waste in the natural environment has increased to such an extent that new and viable solutions for food waste management are required. Biochar is a carbonaceous material produced by means of the pyrolysis process from biomass. It has received much attention due to its huge potential for use as a sustainable soil amendment, carbon sequestration practice, and also as a renewable energy resource. Food waste, one of the gigantic environmental problems, can be converted into biochar through the process of pyrolysis. Process conditions such as temperature and residence time along with feedstock characteristics do have an impact on the biochar yield from the pyrolysis of food wastes. Accurate prediction of biochar yield helps in effective food waste management and minimization of environmental impact through resource use optimization. Hence, AI algorithms have been tested for modeling complex systems with a view to predicting their outputs. This is in the domain of developing an AI-driven, accurate model for biochar yield prediction from the pyrolysis of food waste using Explainable AI techniques. Data collection, preprocessing, and feature selection will be done from the experiments on pyrolysis of food waste. Next, apply machine and deep learning techniques; models such as linear regression, random forest, K-nearest neighbors, and convolutional neural networks will be evaluated. Model transparency and interpretability will be attained using XAI methods, such as SHapley Additive exPlanations (SHAP) values and Local Interpretable Model-agnostic Explanation (LIME). It is expected that the expected deliverables of this research would be accurate AI-based prediction models; insight into the factors influencing biochar yield and process control will also be provided. Recommendations for the optimization of food waste pyrolysis processes to drive the transition toward sustainable biochar production will be derived. It will help cut down greenhouse gas emissions, improve agricultural productivity to enhance food security.

ABSTRAK

Pengumpulan sisa dalam persekitaran semula jadi telah meningkat sehingga memerlukan penyelesaian baharu dan berdaya maju untuk pengurusan sisa makanan. Biochar adalah bahan berkarbon yang dihasilkan melalui proses pirolisis daripada biojisim. Ia mendapat perhatian yang besar kerana potensinya yang besar untuk digunakan sebagai amandemen tanah yang mampan, amalan penyerapan karbon, dan juga sebagai sumber tenaga boleh diperbaharui. Sisa makanan, salah satu masalah alam sekitar yang besar, boleh ditukar menjadi biochar melalui proses pirolisis. Keadaan proses, seperti suhu dan masa tinggal, bersama dengan ciri-ciri bahan makanan, mempunyai kesan terhadap hasil biochar daripada pirolisis sisa makanan. Ramalan yang tepat mengenai hasil biochar membantu dalam pengurusan sisa makanan yang berkesan dan meminimumkan impak alam sekitar melalui pengoptimuman penggunaan sumber. Oleh itu, algoritma AI telah diuji untuk memodelkan sistem kompleks dengan tujuan meramalkan output mereka. Ini adalah dalam domain pembangunan model tepat yang didorong oleh AI untuk meramalkan hasil biochar daripada pirolisis sisa makanan menggunakan teknik AI Terjelas. Pengumpulan data, pra-pemprosesan, dan pemilihan ciri akan dilakukan daripada eksperimen pirolisis sisa makanan. Seterusnya, teknik pembelajaran mesin dan pembelajaran mendalam akan digunakan; model seperti regresi linear, hutan rawak, K-nearest neighbors, dan rangkaian neural konvolusi akan dinilai. Ketelusan dan interpretasi model akan dicapai menggunakan kaedah XAI, seperti nilai SHapley Additive exPlanations (SHAP) dan Penjelasan Model-agnostik Tempatan (LIME). Diharapkan bahawa hasil penyelidikan ini adalah model ramalan berasaskan AI yang tepat; pemahaman tentang faktor-faktor yang mempengaruhi hasil biochar dan kawalan proses juga akan disediakan. Cadangan untuk pengoptimuman proses pirolisis sisa makanan untuk mendorong peralihan ke arah pengeluaran biochar yang mampan akan diperoleh. Ini akan membantu mengurangkan pelepasan gas rumah hijau dan meningkatkan produktiviti pertanian untuk meningkatkan keselamatan makanan.

TABLE OF CONTENTS

	PAGE
DECLARATION.....	II
DEDICATION.....	III
ACKNOWLEDGEMENTS.....	IV
ABSTRACT	V
ABSTRAK	VI
TABLE OF CONTENTS.....	VII
LIST OF TABLES	XV
LIST OF FIGURES	XVI
LIST OF ABBREVIATIONS	XVIII
LIST OF ATTACHMENTS.....	XIX
CHAPTER 1: INTRODUCTION.....	1
1.1 Introduction.....	1
1.2 Problem Statement (PS).....	1
1.3 Objectives	2
1.4 Project Scope	2
1.4.1 Research domain.....	3
1.4.2 Experimental setup	3
1.4.3 Case study used.....	3
1.4.4 Data used	3

1.4.5	Constraints and limit of research	3
1.5	Project Contribution (PC)	4
1.6	Report Organisation	4
1.6.1	Chapter 1: Introduction	5
1.6.2	Chapter 2: Literature Review	5
1.6.3	Chapter 3: Project Methodology	5
1.6.4	Chapter 4: Proposed Method	5
1.6.5	Chapter 5: Results and Discussion	5
1.6.6	Chapter 6: Conclusion	6
1.7	Summary	6
	CHAPTER 2: LITERATURE REVIEW	7
2.1	Introduction	7
2.2	Related Work	10
2.2.1	Domain Identification	10
2.2.2	Issues Related to Domain Problem	11
2.2.2.1	Platform and Data Availability	11
2.2.2.2	Algorithm Complexity	12
2.2.2.3	Model Interpretability	12
2.2.3	Evidence and Statistics	12
2.2.4	Terminology	14
2.2.4.1	Biochar	14
2.2.4.2	Pyrolysis	14
2.2.4.3	Artificial Intelligence (AI)	14
2.2.4.4	Explainable AI	15

2.2.4.5	Machine Learning.....	15
2.3	Critical review of current problem and justification.....	16
2.3.1	Study of Algorithms and Explainable AI	16
2.3.1.1	Linear Regression	16
2.3.1.2	Random Forest Regressor.....	17
2.3.1.3	K-Nearest Neighbors (KNN).....	18
2.3.1.4	Convolutional Neural Networks (CNNs)	19
2.3.1.5	SHapley Additive exPlanations (SHAP)	20
	Figure 2.7 Example Figure of SHapley Additive exPlanations (SHAP).....	21
2.3.1.6	Local Interpretable Model-agnostic Explanations (LIME)	21
2.3.2	Justification of Chosen Algorithms and Explainable AI	22
2.3.2.1	Linear Regression.....	22
2.3.2.2	Random Forest Regressor.....	22
2.3.2.3	K-Nearest Neighbors (KNN).....	23
2.3.2.4	Convolutional Neural Networks (CNNs)	23
2.3.2.5	SHapley Additive exPlanations (SHAP)	24
2.3.2.6	Local Interpretable Model-agnostic Explanations (LIME)	24
2.4	Proposed Solution	25
2.4.1	Linear Regression	25
2.4.2	Random Forest Regressor.....	25
2.4.3	K-Nearest Neighbors (KNN).....	26
2.4.4	Convolutional Neural Networks (CNN).....	26
2.4.5	SHapley Additive exPlanations (SHAP)	26

2.4.6	Local Interpretable Model-agnostic Explanations (LIME)	27
2.5	Summary	27
CHAPTER 3: PROJECT METHODOLOGY		28
3.1	Introduction	28
3.2	Operational Framework	29
3.2.1	Phases of Methodology	29
3.2.1.1	Business Understanding	30
3.2.1.2	Data Understanding	30
3.2.1.3	Data Preparation	31
3.2.1.4	Modeling	31
3.2.1.5	Evaluation	32
3.2.1.6	Deployment	32
3.2.2	Data Collection	33
3.2.2.1	Data Sources	33
3.2.2.2	Data Collection Process	33
3.2.3	Tools and Techniques	34
3.2.3.1	Linear Regression	34
3.2.3.2	Random Forest Classifier	35
3.2.3.3	K-Nearest Neighbors (KNN) Regression	36
3.2.3.4	Convolutional Neural Network (CNN)	37
3.3	Project Milestones	39
3.3.1	PSM I	39

3.3.1.1	Stage 1: Proposal Development and Approval (11 March 2024 - 22 March 2024).....	39
3.3.1.2	Stage 2: Initial System Development Progress (25 March 2024 - 29 March 2024).....	40
3.3.1.3	Stage 3: Report Writing Progress 1 (1 April 2024 - 12 April 2024)	40
3.3.1.4	Stage 4: System Development Progress 2 (15 April 2024 - 26 April 2024).....	40
3.3.1.5	Stage 5: Report Writing Progress 2 (6 May 2024 - 14 June 2024)	40
3.3.1.6	Stage 6: Demonstration to Supervisor (17 June 2024 - 21 June 2024)	40
3.3.1.7	Stage 7: Demonstration to Evaluator (17 June 2024 - 21 June 2024)	41
3.3.1.8	Stage 8: Presentation (17 June 2024 - 21 June 2024).....	41
3.3.1.9	Stage 9: Report Evaluation by Supervisor (24 June 2024 - 28 June 2024).....	41
3.3.1.10	Stage 10: Report Evaluation by Evaluator (24 June 2024 - 28 June 2024).....	41
3.3.2	PSM II.....	42
3.3.2.1	Stage 1: System Development Progress 1 (15 July 2024 – 19 July 2024).....	42
3.3.2.2	Stage 2: System Development Progress 2 (22 July 2024 - 26 July 2024).....	42
3.3.2.3	Stage 3: Report Writing Progress (29 July 2024 - 2 August 2024)	43
3.3.2.4	Stage 4: Demonstration to Supervisor part A (5 August 2024 – 30 August 2024)	43

3.3.2.5	Stage 5: Demonstration to Supervisor part B (19 August 2024 - 30 August 2024)	43
3.3.2.6	Stage 6: Report Evaluation by Supervisor (19 August 2024 - 30 August 2024)	44
3.3.2.7	Stage 7: Checking of English proficiency (19 August 2024 - 30 August 2024)	44
3.3.2.8	Stage 8: Presentation (19 August 2024 - 30 August 2024)	44
3.3.2.9	Stage 9: Demonstration to Evaluator (19 August 2024 - 30 August 2024)	45
3.3.2.10	Stage 10: Report Evaluation by Evaluator (19 August 2024 - 30 August 2024)	45
3.4	Evaluation Metrics	45
3.4.1	Mean Squared Error (MSE).....	45
3.4.2	Root Mean Squared Error (RMSE)	46
3.5	Summary	47
CHAPTER 4: PROPOSED METHOD.....		48
4.1	Introduction.....	48
4.2	Proposed Solution	48
4.2.1	Linear Regression	48
4.2.2	Random Forest.....	49
4.2.3	K-Nearest Neighbors (KNN).....	50
4.2.4	Convolutional Neural Network (CNN)	50
4.2.5	SHAP (SHapley Additive exPlanations)	52
4.2.6	LIME (Local Interpretable Model-agnostic Explanations)	52
4.3	Experiment Design.....	53

4.3.1	Data Collection and Preprocessing	53
4.3.2	Feature Selection and Engineering	54
4.3.3	Model Selection and Training	55
4.3.4	Evaluation	55
4.3.5	Experimental and Simulation Setup	56
4.4	Summary	56
CHAPTER 5: RESULTS AND DISCUSSION		57
5.1	Introduction.....	57
5.2	Evaluation Result	58
5.2.1	Linear Regression	58
5.2.2	Random Forest Regression	61
5.2.3	K-Nearest Neighbors (KNN).....	64
5.2.4	Convolutional Neural Networks (CNN).....	68
5.3	Analysis and Discussion	73
5.3.1	Linear Regression	73
5.3.2	Random Forest Regression	75
5.3.3	K-Nearest Neighbor (KNN)	77
5.3.4	Convolutional Neural Network (CNN)	79
5.4	Summary	83
CHAPTER 6: CONCLUSION.....		85
6.1	Introduction.....	85
6.2	Project summarization	85
6.3	Project Contribution.....	86
6.4	Project Limitation	86
6.5	Future Works	87

6.6	Summary	87
	REFERENCES	88
	APPENDIX A	92
	APPENDIX B	93
	APPENDIX C	97
	APPENDIX D	101
	APPENDIX E	104



اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

LIST OF TABLES

	PAGE
Table 1: Overall and Average of RMSE value when run multiple time using Standard Scaler	59
Table 2: Overall and Average of RMSE value when run multiple time using Logarithmic Transformation	60
Table 3: Overall and Average of RMSE value when run multiple time using Standard Scaler	62
Table 4: Overall and Average of RMSE value when run multiple time using Logarithmic Transformation	63
Table 5: Overall and Average of RMSE value when run multiple time using Standard Scaler	65
Table 6: Overall and Average of RMSE value when run multiple time using Logarithmic Transformation	66
Table 7: Overall and Average of RMSE value when run multiple time using Standard Scaler	68
Table 8: Overall and Average of RMSE value when run multiple time using Logarithmic Transformation	70
Table 9: Comparison of Average RMSE Values between ‘Standard Scaler’ and ‘Logarithmic Transformation’	72
Table 10: Comparison of Top 3 Best Key Features using ‘Standard Scaler’ ‘Logarithmic Transformation’ for All Models between SHAP and LIME.....	81
Table 11: Probability Distribution of Top 3 Key Features from the Comparison table of SHAP and LIME	82

LIST OF FIGURES

	PAGE
Figure 2.1: Example of Biochar Process	10
Figure 2.2: Example Figure of Global Food Waste	13
Figure 2.3: Linear Regression model sample illustration.....	17
Figure 2.4: Example of Random Forest Regressor.....	18
Figure 2.5: Example Figure of K-Nearest (KNN) Algorithm	19
Figure 2.6: Example Figure of Convolutional Neural Network(CNN).....	20
Figure 2.7 Example Figure of SHapley Additive exPlanations (SHAP)	21
Figure 2.8: Example Figure of Local Interpretable Model-agnostic Explanations (LIME)	22
Figure 3.1: Methodology of Biochar Yield Prediction using CRISP-DM Model	30
Figure 3.2: Gantt Chart of PSM 1	39
Figure 3.3: Gantt Chart of PSM II.....	42
Figure 3.4: Formula of Mean Square Error (MSE)	46
Figure 3.5: Formula of Root Mean Square Error(RMSE)	46
Figure 4.1: Linear Regression code from the project	49
Figure 4.2: Random Forest Regressor code from the project.....	49
Figure 4.3: K-Nearest Neighbors (KNN) code from project	50
Figure 4.4: Convolutional Neural Network (CNN) code from project	51
Figure 4.5: SHAP (SHapley Additive exPlanations) code from project	52
Figure 4.6: LIME (Local Interpretable Model-agnostic Explanations) code from project.....	53
Figure 4.7: Standard Scaler code from the project.....	54
Figure 4.8: Logarithmic Transformation code from project.....	54

Figure 5.1: Actual Value vs Predicted Value for Biochar Yield for Linear Regression	58
Figure 5.2: Actual Value vs Predicted Value for Biochar Yield for Random Forest Regressor.....	61
Figure 5.3: Actual Value vs Predicted Value for Biochar Yield for K-Nearest Neighbors	64
Figure 5.4: Training and Validation Loss of Biochar Yield using Standard Scaler	69
Figure 5.5: Training and Validation Loss of Biochar Yield using Logarithmic Transformation	71
Figure 5.6: Results of Feature Analysis Using Standard Scaler with SHAP and LIME	73
Figure 5.7: Results of Feature Analysis Using Logarithmic Transformation with SHAP and LIME.....	74
Figure 5.8: Results of Feature Analysis Using Standard Scaler with SHAP and LIME	75
Figure 5.9: Results of Feature Analysis Using Logarithmic Transformation with SHAP and LIME.....	76
Figure 5.10: Results of Feature Analysis Using Standard Scaler with SHAP and LIME	77
Figure 5.11: Results of Feature Analysis Using Logarithmic Transformation with SHAP and LIME.....	78
Figure 5.12: Results of Feature Analysis Using Standard Scaler with SHAP and LIME	79
Figure 5.13: Results of Feature Analysis Using Logarithmic Transformation with SHAP and LIME.....	80

LIST OF ABBREVIATIONS

FYP	-	Final Year Project
AI	-	Artificial Intelligence
XAI	-	Explainable Artificial Intelligence
KNN	-	K-Nearest Neighbor
CNNs	-	Convolutional Neural Network
NGOs	-	Non-governmental organizations
SHAP	-	SHapley Additive exPlanations
LIME	-	Local Interpretable Model-agnostic Explanations
MSE	-	Mean Squared Error
RMSE	-	Root Mean Squared Error
ML	-	Machine Learning

LIST OF ATTACHMENTS

	PAGE
Appendix A	92
Appendix B	93
Appendix C	97
Appendix D	101
Appendix E	104

اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

CHAPTER 1: INTRODUCTION

1.1 Introduction

Food waste disposal and the reduction of food loss at stores are key global environmental and economic problems, that about 17% annual consumer-available foods is wasted per year, produced as much as 8–10 percent (or even all) greenhouse gas emissions. These emissions emerge at every stage of the food chain and are compounded by the rotting away in landfills, where without oxygen, decomposition significantly augments greenhouse gases that warm our climate as well degrade crop yields based upon nutritional quality and security. A reduction or elimination of food losses should be an essential part in the development agenda to facilitate agrifood systems improve productivity which supports sustainable economic growth. The production of over 1.3 billion tons per year of food waste generates a need to look for more sustainable alternatives than traditional disposal methods (such as landfilling) and therefore the search on conversion technologies such pyrolysis to produce renewable resources from food wastes is receiving increasing interest. Despite that, this thermal decomposition process could convert food waste into biochar — a highly valuable commodity; however, the prediction of bio-char yield is complicated. The present study is focused on developing Explainable AI (XAI)-integrated predictive model for predicting biochar yield extracted from food waste pyrolysis and optimize the process of pyrolysis to promote sustainable management along with green energy suitable process enabling carbon-negative production.

1.2 Problem Statement (PS)

The aim of this project addresses some important challenges related to biochar production as a result from waste-to-bioenergy. Firstly, the variability of biochar yields in biowastes pyrolysis processes is not well understood and poorly predicted. This scenario makes it more difficult to optimize production and resource efficiency.

Moreover, classical theoretical models of yield cannot be easily applied to the design and optimization because they are too complicated in general for process engineers who have "no time".

Besides, the lack of transparency and robustness in existing machine learning methodologies to interpret relationships among input pyrolysis parameters, with their biochar yield output poses a challenge for trust on these predictions.

However, the ability to maximize resource recovery and environmental benefit through pyrolysis is in practice far from optimal or sustainable.

In this regard, the project objective is to generate an Explainable Artificial Intelligence (XAI)-based predictive model for biochar yield from food waste pyrolysis. The model will focus accuracy as well interpretability to be able to ensure that food waste management practices are sustainable and effective.

1.3 Objectives

- i. To identify suitable machine learning and deep learning techniques for biochar yield prediction using training dataset.
- ii. To investigate the prediction model's interpretability and transparency with the utilization of XAI techniques.
- iii. To explain the key factors influencing biochar yield in food waste pyrolysis.

1.4 Project Scope

Currently, developing a machine-learning algorithm that can predict biochar production at any scale is almost an impossible mission; this Critical Comparative Analysis will explore the feasibility of optimising this calculation. It covers various necessary points such as:

1.4.1 Research domain

It is an environmental science and engineering project, mainly focused on waste management, as well as efficient resource utilization.

1.4.2 Experimental setup

To build a prediction model, several AI and machine learning techniques were used in the experimental setup. The following tools were used for creating models, find evaluation and determine the best features: Python programming language, JupyterLab, and libraries such as TensorFlow, scikit-learn, matplotlib, pandas, seaborn, numpy, seaborn, plotly, os, shap and lime were applied.

1.4.3 Case study used

The project developed and tested the predictive model on real and synthetic case studies. These case studies could include pyrolysis of food waste in different places such as a commercially viable unit where companies process their own organic wastes, or an industrial scale plant for the municipal handling system.

1.4.4 Data used

The datasets used for the study included Fixed Carbon, Volatile Matter, Ash, Carbon (C), Hydrogen (H), Oxygen (O), Nitrogen (N) and other details like Residence Time (min.), Temperature (°C), Heating Rate (°C/min.) along with Feedstock Composition & Biochar Yield Statistics. While some of the got datasets have been created through literature search, experimental experiments and collaborations with industry partners.

1.4.5 Constraints and limit of research

The main limitations and constraints of this research were due to the lack of high-quality data, diversity in experimental conditions; as well resources to develop accurate models. The usefulness of the prediction model was also hampered by specific properties related to operation with pyrolysis process and variations in feedstock composition. Acknowledging and attempting to manage these constraints during conduct of the study was vital for increasing confidence in our results.

1.5 Project Contribution (PC)

The study and the contributions of this project were relevant to several stakeholders, concerning various components of sustainable waste management and utilization. It provided researchers and scientists with valuable insights on the prediction of biochar yield from food waste pyrolysis using artificial intelligence (AI) algorithms for enabling future advancement in waste-to-resource conversion technology. Both Policymakers and industry stakeholders employed the developed prediction model along with an intuitive AI (XAI) interpretation to advocate specific policy decisions for improved resource recovery & environmental sustainability initiatives. Institutions of learning included the results in their curriculums to integrate real world AI applications into waste management, which go a long way towards enriching environmental science and engineering programs. Project impacts were used by NGOs, such as the Natural Resources Defense Council and other environmental advocacy groups to inform recommendations on potential strategies in policy development that could help remedy greenhouse gas emissions from food waste through sustainable waste management practices. It led technology developers and entrepreneurs to build new tools, apps while at the same time promote sustainable development in ways that no one would have thought of otherwise. In conclusion, this work offers significant outcomes through a highly accurate and an easily interpretable prediction model for biochar yield which can make crucial contributions to the decision-making process with respect to various stakeholders across sectors as well as inform them how their resource footprints are mitigated in order mitigate impacts of finite resources on sustainability.

1.6 Report Organisation

In such aspect, the structure of this report is well organized to bring an overview towards developing a predictive model on biochar yield from food waste pyrolysis through various AI techniques. The organization is as follows:

1.6.1 Chapter 1: Introduction

With a brief history of the project this chapter zooms in on environmental issues surrounding food waste and how pyrolysis can turn it into biochar. It sets out the research problem and goals of the project to contextualize subsequent chapters.

1.6.2 Chapter 2: Literature Review

This section presents a survey of literature on food waste management, biochar production and AI-driven prediction models. It identifies the blanks in current knowledge and forms the theoretical base of the project.

1.6.3 Chapter 3: Project Methodology

This chapter provide an end-to-end description of project approach from methods used for data gathering, AI techniques selected to create models and usage of XAI techniques for interpretability. A step-by-step approach has been described on how to develop and test the prediction model.

1.6.4 Chapter 4: Proposed Method

In this chapter shows recommended method of biochars yield predictions produced by food waste through pyrolysis. This research tackles the specific AI algorithms employed to develop a model and reviews how XAI methods are then going to be used for better interpretability. This chapter dives into the technical elements of the project.

1.6.5 Chapter 5: Results and Discussion

This chapter investigates the outcomes of creating prediction model disclosed. An analysis of model performance measures, observations with XAI techniques and comparisons to the state-of-the-art are presented. This chapter is an indepth understanding of the factors affecting biochar yield.

1.6.6 Chapter 6: Conclusion

The last chapter includes the key findings, significance of the work for social research and some reflections on possible future investigation. It then assesses the success of their project in meeting these goals and as a case study considering policy implications for food waste management and biochar production.

1.7 Summary

This project will hopefully meet the difficulties of biochar yield prediction from pyrolysis at food waste due to a faithful and deterministic AI-based predictive model. Through Explainable AI techniques, the model will reveal what factors affect biochar yield and inform future producers with more efficient and sustainable production processes. The anticipated results of the work are expected to have a major impact on waste management and compliance with sustainable development goals.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

The purpose of this investigation was to develop a predictive model for biochar's yield, produced from food waste pyrolyzed with advanced Artificial Intelligence techniques. Through the use of interpretable AI algorithms, the project aims not only to achieve high predictive accuracy, but also to be able to explain and gain an understanding about what influences biochar yield. This is where the idea of Explainable AI becomes essential since it opens models for easier scrutiny, letting stakeholders understand what drives a given model and subsequently making their predictions that much more trustworthy which in turn be actioned upon (Gunning & Aha 2019; Doshi-Velez & Kim, 2017).

This research has four prime objectives as: Firstly the collection and analysis of data on food wastes, their properties, various types and conditions in which they are pyrolyzed to produce biochar. Second, model development encompasses the creation of an interpretable AI-model that can predict biochar yield from given inputs, using explainable AI approaches to ensure these predictions are understandable and trustful for researchers and field practitioners as well. Third, the prediction model will be evaluated and validated by thorough testing with data of different properties (using a broad set of test datasets) to measure performance in terms like mean square error and trustworthiness concerning existing predictions. Last, the insights and optimization will be conducted through exploiting explainable AI model to reveal important variables that affect biochar yield as well as potential optimization approaches for enhancing both economic efficiency and process robustness of producing food waste-derived biochars.

The literature review will investigate some of these key categories to help prepare the groundwork for research. It focuses on existing approaches to food waste

management, the issues surrounding it and the potential for re-purposing food wastes into a valuable commodity such as biochar (FAO, 2011). It will be an overview of the pyrolysis process and types which is slow, fast and flash, how to biochar yield, or what effect on the yields different conditions have for gas (Lehmann & Joseph 2009). The research will also examine biochar as an agent for environmental and agricultural benefits, from soil fertility improvements to carbon sequestration and waste reduction. Some of the AI methods for environmental engineering, and its potential to predict results & optimize processes will be reviewed as well (Kabir, Yacout, & Le, 2015). Moreover to underline the importance and methodologies of explainable AI which focus on delivering models that are interpretable, transparent and trusted by their users (Doshi-Velez & Kim, 2017).

The use of machine learning methods for modeling the yield of biochar have been reported from a Simple Linear Regression up to Random Forest Regressor and K-Nearest Neighbors (KNN). Linear regression offers an accessible and interpretable view of the relationship between input variables and biochar yield (Johnson & Wichern, 2007). Random Forest Regressor: This method is robust and can handle complex interactions between your dependent variables (Breiman, 2001). KNN, since it is computationally intensive works well with non-standard data patterns and does not need a pre-defined model structure so can be very flexible when investigating relationships which may seem hard to discern (Cover & Hart, 1967).

However, recent advances have used deep learning approaches with a specific focus on extraction of predictive features to improve the model accuracy for biochar yield. Developed to process images, CNNs are great at building layers of spatial hierarchies where the inputs and features interact in scenarios with a high number of dimensions (LeCun et al., 2015). Yet the use of CNNs introduces two challenges of its own: computational costs are high, and overfitting becomes a problem unless preprocess quite many data is used (Goodfellow et al., 2016).

The rise in popularity of CNNs, however, has brought the issue of model transparency to light. Many people describe these models as "black boxes" because they are not very transparent on how exactly those predictions were made (Doshi-Velez & Kim, 2017). This explains the boom in explainable AI techniques, or how

SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations), which have been shown very useful to increase model interpretability. SHAP is rooted in game theory and calculates relative importance of the features through contribution to prediction for each scenario (Lundberg & Lee, 2017). This will facilitate the discovery of contributory factors, e.g. temperature and type of feedstock in biochar yield projections (Le et al., 2024). LIME harnesses this by offering local explanations that approximate the behavior of a real model around individual predictions (Ribeiro et al., 2016).

Hence, a comprehensive blend of strengths from traditional AI algorithms like simplicity & interpretability offered by Linear Regression; high resiliency provided due to ensemble learning in Random Forest Regressor and the versatility as in KNN with sophisticated models like CNNs along with explainable AI techniques offers an exhaustive way forward. This will serve as a way to improve prediction accuracy using well-integrated accurate models and keep the model interpretable, actionable by practitioners (Caruana et al., 2015). In addition, the embedding of smart sensing as well as IoT in linking with biochar production unit for real-time data collection further augments support to predictive models, resulting into accurate and interpretable predictions on yield of biochar. The release contributes greatly to sustainable waste management by transforming food waste into biochar more effectively, paving the way forward towards environmental sustainability and resource conservation (Zhao et al., 2019; Khatri et al., 2021).

This work is in the direction of this need and aimed to develop meaningful predictive models for biochar yield from food waste pyrolysis. It may provide the necessary breakthrough for sustainable recycling of waste material and energy generation from biochar, thus providing valuable information to improve performance in terms of both technical feasibilities prior enhancing the efficiencies and sustainability potentiality related to biochar production.

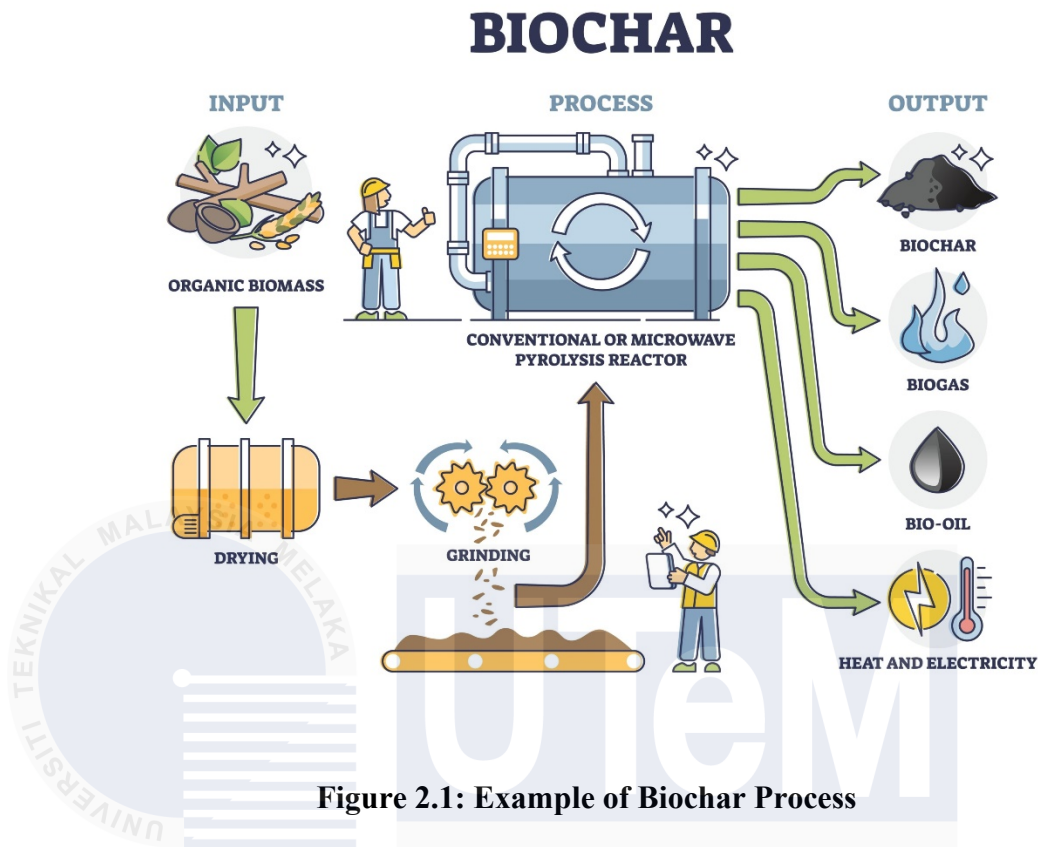


Figure 2.1: Example of Biochar Process

2.2 Related Work

This section of the research does revision work to establish literature and other relevant information concerning about building an Artificial Intelligence (AI) based predictive model for biochar yield in food waste pyrolysis. The goal is to get a good grasp on where the state of knowledge in biochar production and food waste pyrolysis methodologies is, as well as in AI applications for environmental engineering.

2.2.1 Domain Identification

This project is held within the confluence of waste management, environmental engineering and artificial intelligence (AI) which both have their domains. In particular, the conversion of food waste to biochar through pyrolysis is discussed and how AI methods can be applied for the prediction of biochar yield under different conditions (Lehmann & Joseph, 2009).

It has been of significant environmental concern in the recent years that, larger proportion of food is being wasted (FAO, 2011). Landfilling and incineration are not

sustainable ways to manage food waste due to the environmental footprint of these practices, including greenhouse gas emissions and resource depletion. Producing biochar through pyrolysis of food waste offers a solution that diverts the waste from disposal and creates an additional product with numerous environmental functions (Lehmann & Joseph, 2009).

Pyrolysis is a thermochemical decomposition process that occurs at high temperatures, usually above 300 °C with no oxygen present. Biochar is defined as a pyrolysis-derived material that to fuel mixture evolved carbon of biomass for improvement water holding capacity, carbon sequestration and betterment in soil fertility (Lehmann & Joseph, 2009). The yield and characteristics of biochar are influenced by the type of food waste, pyrolysis temperature as well residence time.

This is not completely wrong, however in the last few years AI techniques and mostly machine learning has been used more and more for outcome prediction in different environmental engineering processes. These models can easily tackle the complexity and variability of input data, subsequently leading to precise predictions along with optimizing processes as well (Kabir et al., 2015; Al-Gheethi et al., 2018). For the case of biochar production, AI models can be trained to predict the yield of biochar given a feedstock and pyrolysis condition which was shown to support optimal process optimization and decision-making

2.2.2 Issues Related to Domain Problem

There are several key issues in dealing manufacture and prediction domain problem like conversion pyrolysis food waste to biochar, which could be expected by using artificial intelligence (AI).

2.2.2.1 Platform and Data Availability

Key challenges in this area include the data, both availability and quality. Since pyrolysis experiments give a variety of complex data-structures, the available data concerning food waste properties and operative circumstances in addition must be high quality for developing good predictive models (Al-Gheethi et al., 2018).

2.2.2.2 Algorithm Complexity

Training an AI model for the task of biochar yield predication needs to decide on appropriate algorithms and tune them in a way that it achieves maximum performance. Due to the complex nature of pyrolysis and variability in properties from different food waste substrate, using an advance algorithm that is able to model non-linear relationships and interactions between variables is required (Kabir, Yacout, & Le, 2015).

2.2.2.3 Model Interpretability

Even though deep learning models have higher accuracy in prediction, they can be “black boxes” and not interpretable at all. This inability to see what exactly the model has learned, constitutes a hindrance in putting AI models into perception. Recent works such as Gunning & Aha (2019) and Doshi-Velez & Kim (2017) pointed out that explainable AI is highly required for model interpretability to maintain trust of users.

2.2.3 Evidence and Statistics

The area problem is supported by the amount of food waste produced and potential benefits of biochar, respectively. According to the Food and Agriculture Organization (FAO), one-third of all food produced worldwide is lost or wasted, equating to some 1.3 billion tons per year; This waste constitutes a sizeable depletion of resources as well as puts substantial pressure on the environment; it is responsible for an approximate 8-10% of global greenhouse gas emissions (FAO, 2011).

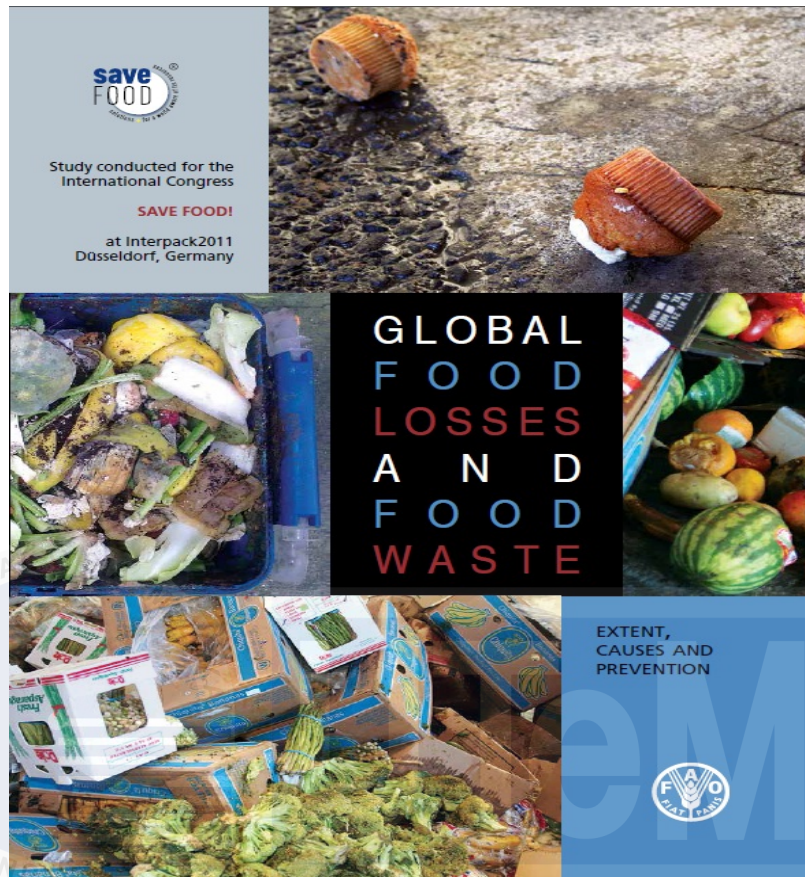


Figure 2.2: Example Figure of Global Food Waste

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

According to Lehmann & Joseph (2009), research has shown that biochar is an efficient carbon sequestration strategy; just one ton yield from a pyrolysis process can bind around 2.2 to 3.3 tons of CO₂ equivalent. The large carbon sequestration potential of biochar reflects its essential value to climate change mitigation. In addition, biochar application to soils could increase crop yields through better soil nutrient storage and water. Studies show an average improvement in crop yield of 10–20 %, and up to 100% for nutrient poor soils due to biochar. These benefits underscore double additionality of biochar to both waste management and enhanced agricultural productivity.

2.2.4 Terminology

Several key terminologies are to be read in the context of this review on food waste based biochar production via pyrolysis and its AI prediction.

2.2.4.1 Biochar

Biochar is a carbon-rich material that gets produced by heating organic matter in anaerobic condition. They are a popular soil amendment because they provide a great many nutrients for plants and help retain moisture. Biochar also contributes to carbon sequestration, key in reducing the effects of climate change since it helps store for a long time some of the stored soil-carbon. Biochar has sustainable potential while delivering crucial environmental services, from the improvement of soil structure yields to mitigate climate change by sequestering carbon dioxide or reducing other greenhouse gases (Lehmann & Joseph, 2009).

2.2.4.2 Pyrolysis

Pyrolysis is a thermochemical decomposition of organic material at elevated temperatures in the absence of oxygen. The end-products of this process are biochar, which is charcoal that can be used as soil fertilizer and to in components such as plastics production; while the oils and syngas which largely consists of Hydrogen gas produced now contain less lighter elements so they have a higher calorific value. Here the solid carbon-rich residue called biochar and liquid that is produced can be used as a fuel or chemical feedstock known as Bio-oil. This combination of gases including hydrogen and carbon monoxide is known as Syngas that can be used for the production of energy. Pyrolysis is an efficient process of converting various waste biomass feedstock to useful products, serving both renewable energy production and the field of waste management (Lehmann & Joseph, 2009).

2.2.4.3 Artificial Intelligence (AI)

Artificial Intelligence (AI) is the simulation of human intelligence processes by machines, especially computer systems. Such processes include but are not limited to learning (the acquisition of information and rules for using the information), reasoning (using rules to reach approximate or definite conclusions) and self-

correction. Applications of AI range from natural language processing and robotics to advanced data analytics. This is the ability of a machine or computer to do tasks that usually require human intelligence such as vision, speech recognition, decision-making and language translation (Kabir, Yacout, & Le, 2015).

2.2.4.4 Explainable AI

It simply means the mechanisms and methods employed by AI to present its outputs in such a way that humans can understand them. Such transparency plays a key role in ensuring that AI systems are trustworthy and accountable. Explainable AI provides insight into how decisions are reached, especially within complex models like deep learning. SHAP (SHapley Additive exPlanations) values, LIME (Local Interpretable Model-agnostic Explanations), and attention are some of the techniques used to understand model behavior and feature importance. This interpretability is critical to enable AI systems to be inspected in a way that they can be trusted useful for high-stakes applications (Gunning & Aha, 2019; Doshi-Velez & Kim, 2017).

2.2.4.5 Machine Learning

Actually, ML is an area of AI that has a particularly developed methodology in algorithms used to predict or decide based on data. Unlike traditional programming in which a programmer has to explicitly instruct the system how to do a task, machine learning algorithms build models from sample data (training data) so they learn what predictions or decisions need made without being programmed. ML includes several strategies like supervised learning, unsupervised learning and reinforcement learning appropriate to different types of problems and data structure. It is being extensively used across applications such as recommendation systems, fraud detection, image recognition and predictive analytics (Kabir et al., 2015).

This study will overcome these shortcomings by addressing the challenges and making use of AI & machine learning to develop a reliable, explainable predictive model for biochar yield out from food waste pyrolysis. These models may show a promise in biochar production and also serve as an alternative sustainable resource management tool, which will lay the foundation for constructing new models of waste processing that can lead to renewable energy generation.

2.3 Critical review of current problem and justification

A detailed critical analysis of the existing problem is discussed with reference to potential challenges for and synergistic opportunities between biochar produced from food waste through pyrolysis, in conjunction with applications using artificial intelligence (AI) methods used for prediction of yields. The reason to write on this topic is because of its importance in waste management and environmental sustainability.

2.3.1 Study of Algorithms and Explainable AI

For the prediction of biochar yield from food waste pyrolysis, there are several AI algorithms to resolve this issue. We focus on four specific algorithms, each providing a distinctive advantage and approach to the problem.

2.3.1.1 Linear Regression

Linear regression is a statistical method that models the linear relationship between one or more independent variables and dependent variable by fitting straight line along with these points; it can be used to predict values for unknown data. According to Brennan et al. (2021) for the biochar yield prediction, we selected linear regression which is easy to understand and interpret as a relationship between the input variables such as feedstock properties and pyrolysis conditions. The basic form of linear regression is one independent variable used to predict a single dependent based on your model makes by assuming there are approximately straight-line relationships. With food waste pyrolysis, linear regression for predicting biochar yield serves to provide a fundamental understanding of how feedstock properties and conditions affect the resulting yield. It provides a simple model that can be easily interpreted, which gives an insight into how the variables affect each other; it helps in making basic predictions or understanding what some of the key factors are. Linear regression, however, assumes a linear relationship between the dependent and independent variables which may be over-specified for complex relationships that exist in biochar production. It might also be susceptible to outliers which may affect the average and

weaken its predictive ability. On top of this, presence of multi-collinearity issue when independent variables show high collinear correlations which affects the robustness and unbiased nature in predicting (Chen et al., 2023).

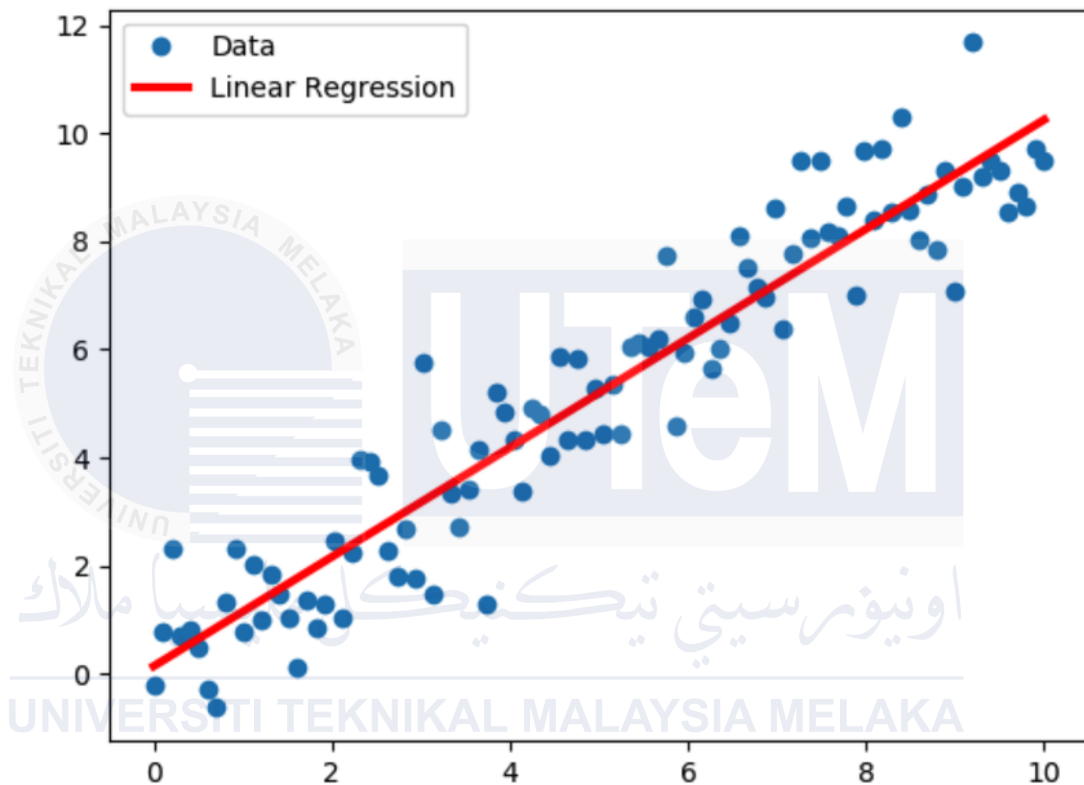


Figure 2.3: Linear Regression model sample illustration

2.3.1.2 Random Forest Regressor

In contrast, while the random forest regressor that Wang et al. (2020)), was ensemble learning that fits a number of decision tree during training and practices the mean prediction of these trees. A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. Random Forest Regressor is appropriate for predicting biochar yield from food waste pyrolysis as it can manage the complex and non-linear relationships between input variables with output. Explain ability while being able to manage a large number of input arguments and interactions

in particular when it comes to the diverse nature of pyrolysis data. Now, as Li et al. (2022) stated, with this ensemble learning you aggregate predictions of many decision trees which helps prevent overfitting and therefore improves your predictive power. The ability of random forest models to provide feature importance, as reasoned by Khan et al. (2022), has proved to be essential in determining the key factors that influence biochar production and thus its processes. However, the random forest regressor also has some drawbacks like being computationally expensive especially as number of trees and size of dataset grows. Compared to uninterpretable black box complex predictors, random forests are much more interpretable, but still not very transparent as linear regression. On top of that, though random forests are made with the intention to curb overfitting they can still also suffer from it if there are just not enough trees or other hyperparameters aren't set properly. This fundamental knowledge of the theory, applications and constraints of linear regression as well as random forest regressor provides confidence in creating strong models for biochar yield from food waste pyrolysis.

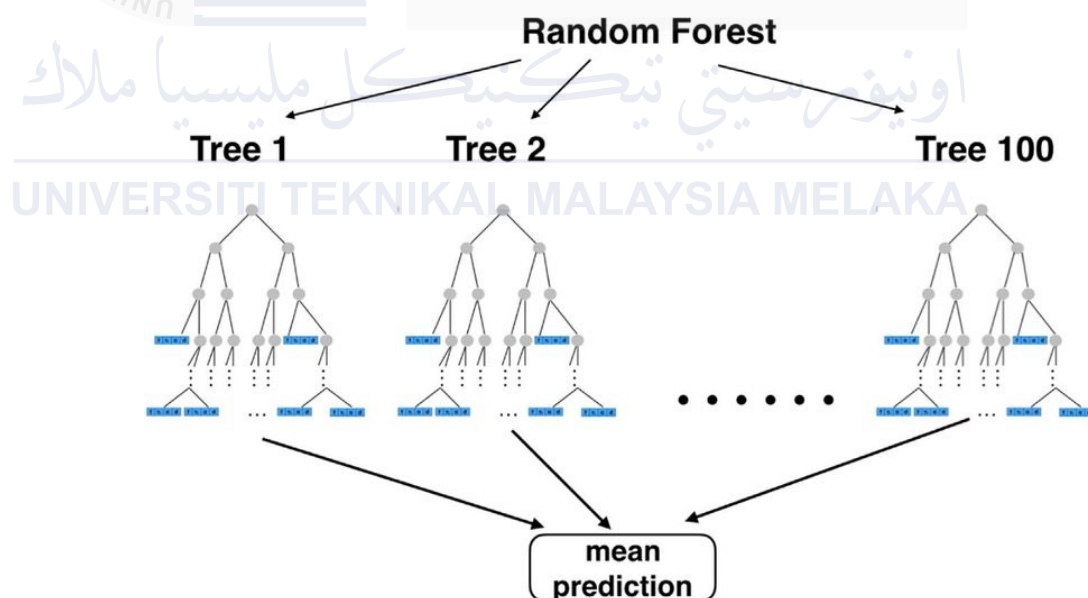


Figure 2.4: Example of Random Forest Regressor

2.3.1.3 K-Nearest Neighbors (KNN)

K-nearest neighbors (KNN), a non-parametric, instance-based learning algorithm reviewed by Hai et al. (2021). This model selected due to its ability to capture nonlinear dependencies, including multiple input and output variables;

feedstock properties and pyrolysis conditions on biochar yield. Based on the principle of similarity, KNN makes a prediction for a new data point by searching its 'k' closest training examples using metrics such as Euclidean distance. The flexibility that KNN provides in model the complex and non-linear patterns within data was beneficial for prediction of biochar yield from food waste pyrolysis, where linear models fail (Pathy et al., 2020). Nevertheless, as Zhang et al. (2020), the computational cost of KNN limits its utility, as it involves many distance measurements of data points which is heavy with large datasets. The other is that the algorithm struggles with high-dimensional data because of a little thing known as "the curse of dimensionality", whereby distances become increasingly meaningless in higher dimensions. These considerations highlight the importance of optimising parameters and managing datasets well to obtain best results for biochar yield prediction using KNN.

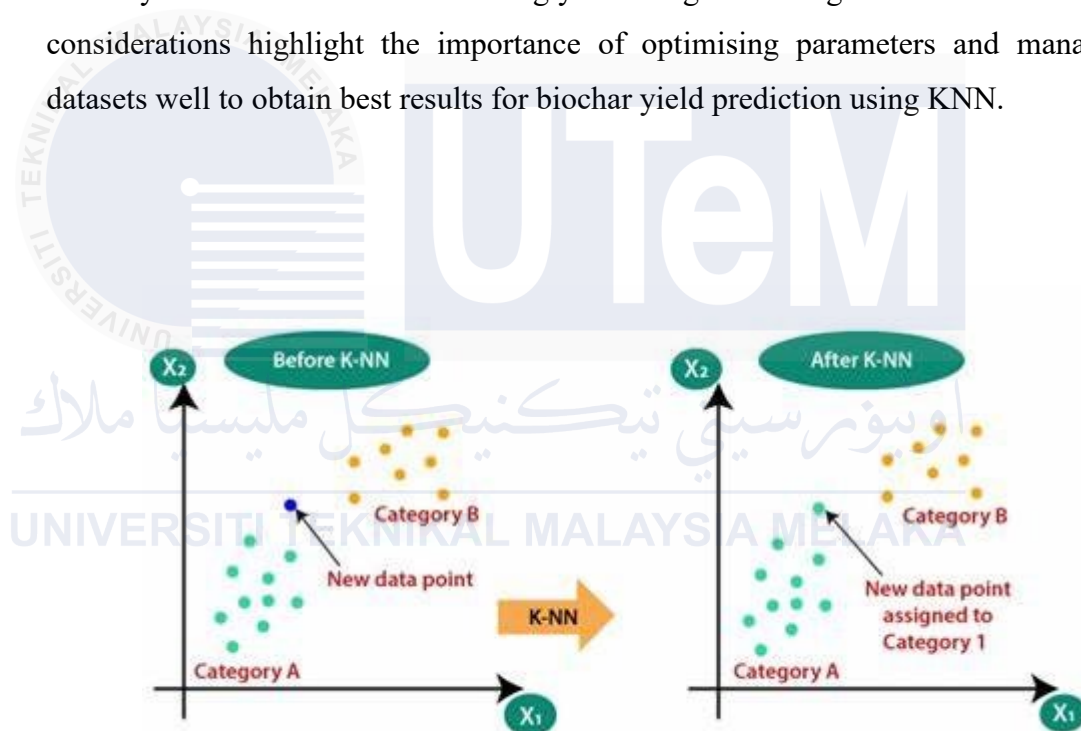


Figure 2.5: Example Figure of K-Nearest (KNN) Algorithm

2.3.1.4 Convolutional Neural Networks (CNNs)

CNNs are a sub-type of deep-learning model that has been more successful with data rich in grid-like topology, such as images. Insights from Lee et al. (2020) on Convolutional Neural Networks (CNNs) that such a strategy works well for multi-dimensional data which biochar production examples in this case. As suggested by Hai et al. (2023), they include convolutional layers which use a filter to scan the input data and find spatial hierarchies or patterns, pooling layer that reduces dimensionality of

the filtered images (or arrays), and fully connected layers at final step for classification/regression. In those cases, where multiple properties of the pyrolysis process have been considered as structured data for predicting biochar yield from food waste Pyrolysis then CNNs can be modified to suit a proposed model. Through being able to learn intricate patterns and connections in the data, CNNs can provide accurate predictions even when these relationships become highly complex. A crucial capability for modeling biochar production as numerous factors interact with each other non-linearly. But CNNs are computationally intense and data hungry, as noted by Wang et al. (see2022), implying extra caution and consideration in real-world deployments to be feasible with due process. They also tend to overfit if not regularized using techniques like dropout or weight decay, meaning that do not generalize well to new out-of-sample examples. Another issue with more general CNNs is that they often require a lot of labeled data to train effectively, which can be prohibitive in domains where obtaining labeling data is difficult or expensive.

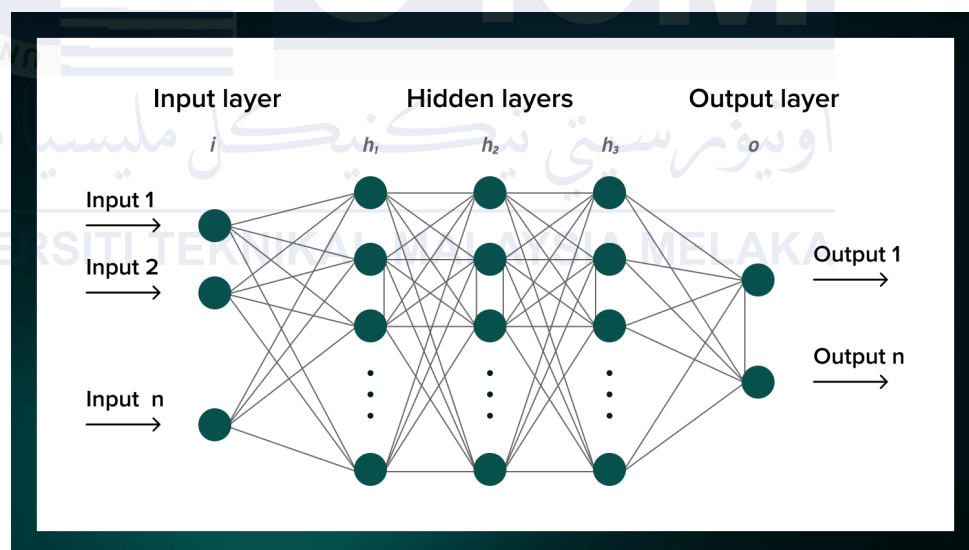


Figure 2.6: Example Figure of Convolutional Neural Network(CNN)

2.3.1.5 SHapley Additive exPlanations (SHAP)

It is inspired from the coalitional game theory and provides a consistent measure of feature importance (SHAP value). It takes into account all feature permutations, which allows it to compute how important each of the features was in making a prediction for that specific model — providing global insight about what behaviors drive your model (Lundberg & Lee, 2017). Regarding biochar yield

prediction, SHAP can interpret how various features such as feedstock type or pyrolysis temperature contribute to the final yield and in consequence aid understanding of the decision making process for a given models (Le et al., 2024).

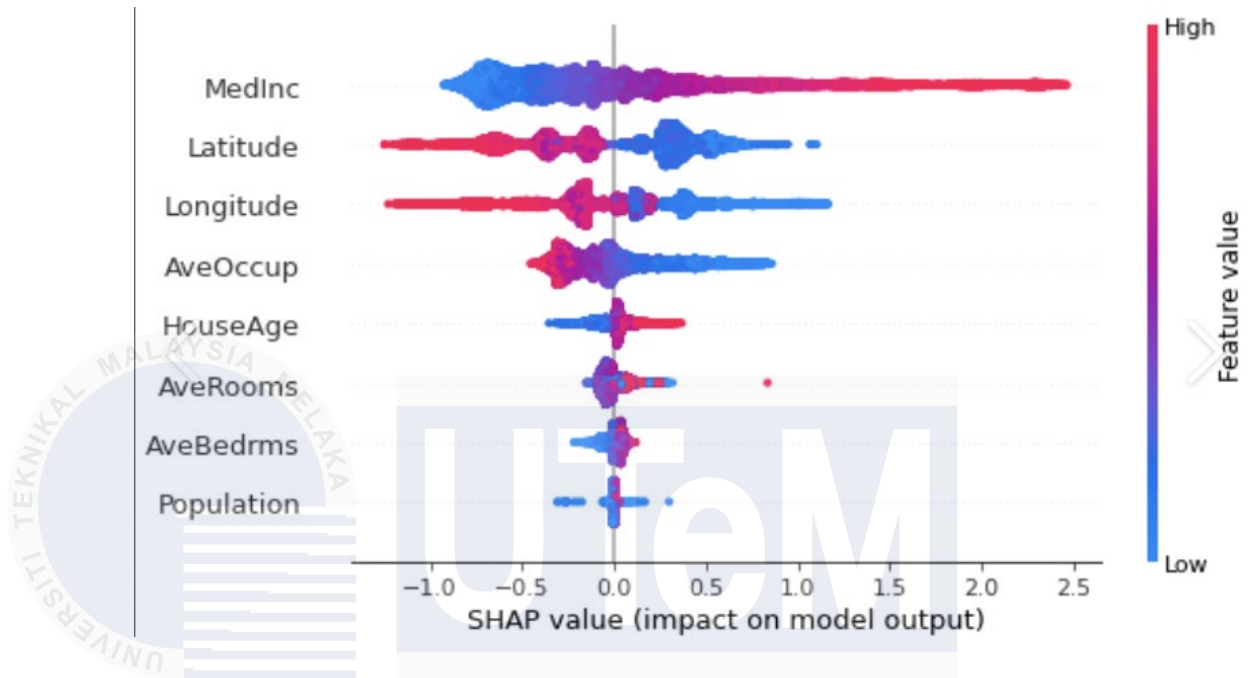


Figure 2.7 Example Figure of SHapley Additive exPlanations (SHAP)

2.3.1.6 Local Interpretable Model-agnostic Explanations (LIME)

LIME does this by approximating the behavior of a trained model locally around disparate predictions. LIME works by perturbing the input and checking how the output changes to find an interpretable model like a linear one that explains the behavior of AI in some vicinity close to an individual prediction (Ribeiro et al., 2016). The method directly defines the prediction of biochar yield under certain circumstances, such as if the feedstock was used at the type of food waste.



Figure 2.8: Example Figure of Local Interpretable Model-agnostic Explanations (LIME)

2.3.2 Justification of Chosen Algorithms and Explainable AI

The selection of each algorithm is based on its capacity to solve unique issues in predicting biochar yield from food waste pyrolysis:

2.3.2.1 Linear Regression

These models are chosen almost exclusively for the sake of simplicity and understandability. It forms basis relationships between the factors of input variables affecting biochar yield and quality, including feedstock properties such as moisture content, carbon contents of biomass materials and pyrolysis conditions like temperature, heating rate. Owing to these reasons, this approach is perfect for preliminary investigations of different governing factors on biochar production. This results in the fact that linear regression can simply use a line to fit desired data and reveals easy observation on how independent variables influence the biochar yield. Evidence for this comes from research by Brennan et al. (2021) and Chen et al. (2023) which highlights its utility in identifying key factors affecting biochar production.

2.3.2.2 Random Forest Regressor

The random forest regressor has been chosen, as it can deal with complex inherent non-linear relationships for pyrolysis data. In addition, this ensemble learning

method builds multiple decision trees during training and also provides the mean prediction of those individual trees so that overfitting could be reduced as well as making a better predictive accuracy. This works very well for capturing interactions between multiple input variables, such as feedstock composition and pyrolysis conditions on biochar yield. Studies by Wang et al. (2020) and Khan et al. (2022), random forests reveal feature importance which helps to recognize the factors that play a part in where more biochar is obtained from. Though computationally demanding, random forests proved to be a robust tool for modeling the complex relationships among biochar yield predictors.

2.3.2.3 K-Nearest Neighbors (KNN)

K-nearest neighbors (KNN) is given preference because it can cope with non-linear relationships and changes in food waste properties as well as operating conditions of pyrolysis. For a new point, this algorithm predicts the biochar yield by similar to its 'k' nearest neighbors in the training dataset. This flexibility of KNN is appropriate in situations where the biochar production data does not have regular patterns and extensive relationship among variables. KNN, however, is computationally expensive especially on large dataset as it has to calculate the distance between points and this verifies its importance in biochar yield prediction due to our conjecture of existence of local patterns in data. Insights from Hai et al. (2021) and Pathy et al. (2020) emphasized the versatility of KNN in capturing different data patterns emerging from biochar production processes.

2.3.2.4 Convolutional Neural Networks (CNNs)

We have also applied Convolutional Neural Networks (CNNs) due to their ability of working with high-dimensional data, such as images or structured grid-like topology datasets for recognizing complex patterns in biochar production processes. CNNs have convolutional layers that learn spatial hierarchies and patterns from input data, and pooling/full connected layer for classification/regression. Although CNNs have exhibited great predictability, they are computationally intensive to create and test as this requires specialized hardware like GPUs with enough labeled data needed for training. Studies by Lee et al. (2020) and Wang et al. (2020) demonstrate the power of

CNNs in learning complex patterns from biochar production data, but also outline a hardware limitation that prevents their practical deployment.

2.3.2.5 SHapley Additive exPlanations (SHAP)

SHAP provides a compelling global explanation of the model as it is capable to explain feature importance over all predictions too. This is especially important in the context of biochar yield prediction, since it allows assessing more accurately how different input variables affect overall impurity concentrations which is the model output. The fact that SHAP can deliver additive explanations in a consistent manner, even for black-box models like CNNs makes it very useful when explaining complex neural network predicting the biochar yield under various scenarios (Lundberg & Lee, 2017). The SHAP technique allows researchers to see which features influence the model's predictions the most, and hence provides more evidence-based guidance in optimizing decision making for biochar production.

2.3.2.6 Local Interpretable Model-agnostic Explanations (LIME)

LIME has been chosen because it is useful for preparing local explanations and also have to make sure that there is a requirement to understand why one type of feedstock results in a particular biochar yield under the given pyrolysis conditions. LIME can work with any tool for training the ML model since it is designed agnostic to any machine learning models, which complements SHAP by providing global insights and looking out for local explanations in great detail (Ribeiro et al., 2016). However, the LIME approach provides only local explanations, which describes why a single prediction was made and is therefore useful in helping to understand decisions corresponding with particular outcomes or need for unexpected results. By doing so, practitioners can check whether the model predictions are valid, and which conditions cause the model to have different knowledge. With biochar yield prediction, LIME can add explanatory documents to each specific prediction and suggest actions for the optimization of a process.

Thus, the incorporation of linear regression, random forest regressor and K-nearest neighbors (KNN) along with Convolutional Neural Network (CNNs) provides a comprehensive model to predict biochar yield from food waste pyrolysis. One or the

other algorithm has something exceptional to deliver: linear regression, for its interpretability and base level findings, random forest brings adaptive in handling convoluted separations & non-linearity that are quite difficult through a simple model, KNN speaks complex initial data pattern understanding whereas CNN captures high dimensioned complexities easily. In order to make these models more transparent and interpret-able we use explainable AI techniques like SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations). LIME provides on a single prediction level, while SHAP gives it globally over an entire model by averaging out the importance and contribution of multiple features for every record. Our approaches then recycle the founding of other studies, and they utilize these interpretability methods to improve both the generalization ability as well as maintain explainable answers for biochar yield prediction accuracy, pyrolysis process optimization and check trustworthiness in models decision.

2.4 Proposed Solution

The chosen methodologies and techniques for predicting biochar yield from food waste pyrolysis are selected based on the comprehensive review and analysis of previous research to ensure model robustness, accuracy, interpretability.

2.4.1 Linear Regression

Lineal regression is selected because it allows us to explore the relation with prediction variables and biochar output. Lehmann and Joseph (2009) indicated that linear regression is appropriate for initial biochar development modeling as well, positively describing the relationship between yield on feedstock composition or pyrolysis condition. This simplicity in approach helps provide a clear basic model vs other complex algorithm.

2.4.2 Random Forest Regressor

Random forest regressor has been selected as it can handle non-linearity, and interactions among multiple variables in complex datasets. Research by Cutler et al. (2007) provide one of the best examples that they found that random forests predicted held-out test data better than linear models in an environmental modeling context. This

method of making the ensembles helps to avoid overfitting and improve robustness because it integrates results from multiple decision trees according with variability in biochar production data.

2.4.3 K-Nearest Neighbors (KNN)

K-nearest neighbors (KNN) are used because they can easily capture the non-linear association between dependent and independent variables. According to Kabir et al. (2015) states that KNN is very nice for environmental models since the input-outcome relationships are complex and a priori hard to formulate in parametric models. This mode predicts the biochar yield by its similarity to neighboring data points in feature space, providing interpretation on behavioral interactions without making complex assumptions.

2.4.4 Convolutional Neural Networks (CNN)

Convolutional neural networks (CNNs) are used because they have the ability to identify complex patterns and extract relationships from high-dimensional data. CNNs had been commonly used for image recognition, and they were better suited to environment data in that spatial-temporal dependence could be captured with a CNN. CNNs have been increasingly applied in environmental modelling due to their predictive capabilities also in complicated cases (Lundberg and Lee, 2017). This approach guarantees high prediction quality for biochar yield while modelling the spatial and temporal effects due to feedstock properties or pyrolysis conditions.

2.4.5 SHapley Additive exPlanations (SHAP)

In the field of environmental engineering, SHAP technique predicted that complex interactions between input variables and model outcomes play an important role in optimizing pyrolysis for biochar production (Le et al., 2024). By summarizing feature importance with SHAP values, both a general understanding of model prediction can be gained and then the insights themselves could become implemented (Lundberg & Lee, 2017). SHAP coverage of explanations for understanding and improving biochar yield predictions models is consequently well-suited.

2.4.6 Local Interpretable Model-agnostic Explanations (LIME)

Machine learning interpretability, LIME as an example of a model-agnostic approach to interpretation has proven its power in generating locally interpretable approximations of complex models that ultimately provide for more transparent and trustworthy models (Ribeiro et al., 2016). Because LIME is a model-agnostic and flexible method, it can explain individual predictions in terms that are easy to understand for users as well as being complex enough to provide detailed local explanations so this technique best suits demonstrating fine-grained local interpretability (Ribeiro et al., 2016). This versatility also guarantees that researcher and practitioners can validate the predictions from models based on a clear model attribute hierarchy, fitting very well with our objective of enriching interpretability in predictive analytics tasks.

2.5 Summary

This chapter laid the groundwork for a biochar yield prediction model that could be developed from food waste pyrolysis. The literature review has discussed various important areas including food waste handling, pyrolysis processes & designing strategy for its industrial purpose s agenda in order to validate biochar as an effective solution, merits of biochar and AI techniques used not only in environmental engineering sector but also focus is given into a more emerging topic which believes XAI that this research intends to fill those gaps. The critique underscored the significance of yield prediction which was strongly linked with waste management and, hence sustainability as a whole. This is done by selecting a linear regression algorithm for simplicity, random forest regressor for handling complex relationships, K-nearest neighbors which requires almost no pre-processing to further prove and convolutional neural networks in order unlock the ability of deep learning on high dimensional data. The next stage is the work phase with data collection, model development, evaluation, insights extraction and documentation.

CHAPTER 3: PROJECT METHODOLOGY

3.1 Introduction

The following chapter describes the methodology used to build a predictive model of biochar yield for pyrolysis of food waste. This approach has been designed to ensure complete data assembly, competent model construction, and holistic model validation while begetting meaningful optimization strategies. The first step is collecting data related to different types of food waste, such as the chemical composition or pyrolysis conditions like temperature and residence time. This data should be pre-processed carefully to make it clean and ready as the best quality for training a predictive model.

To generate a model, four different algorithms are chosen: linear regression is used for statistical inference, random forest has good performance in prediction, K-nearest neighbors can support recommendation and convolutional neural network aims to make deep learning as it should be. Since these algorithms have different advantages in capturing particular aspects of biochar production dynamics, from linear relationships to complex high-dimensional interactions; These would be used to train each model and predict biochar yield using pre-processed data.

The model will be validated for performance using standard metrics like Mean Squared Error (MSE) and Root Mean Square error on generalizability. It is important to keep our models interpretable and we will use tools like SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) for explaining the feature importance in predicting Biochar Yield. SHAP provides global insights on feature importance throughout the dataset while LIME localizes explanations for individual predictions; explains instances of biochar yield or groups, looking to understand anomalies or patterns that may exist.

The outputs of these techniques will be used to inform optimization strategies for improving the efficiency and reliability at which biochar can be produced. All steps of the process after, will be detailed documented to provide traceability and reproducibility. The secondary goal of the structured approach is to provide a predictive model that advances scientific understanding and enables practical insights for marine species conservation with respect to sustainable, zero-waste environmental management.

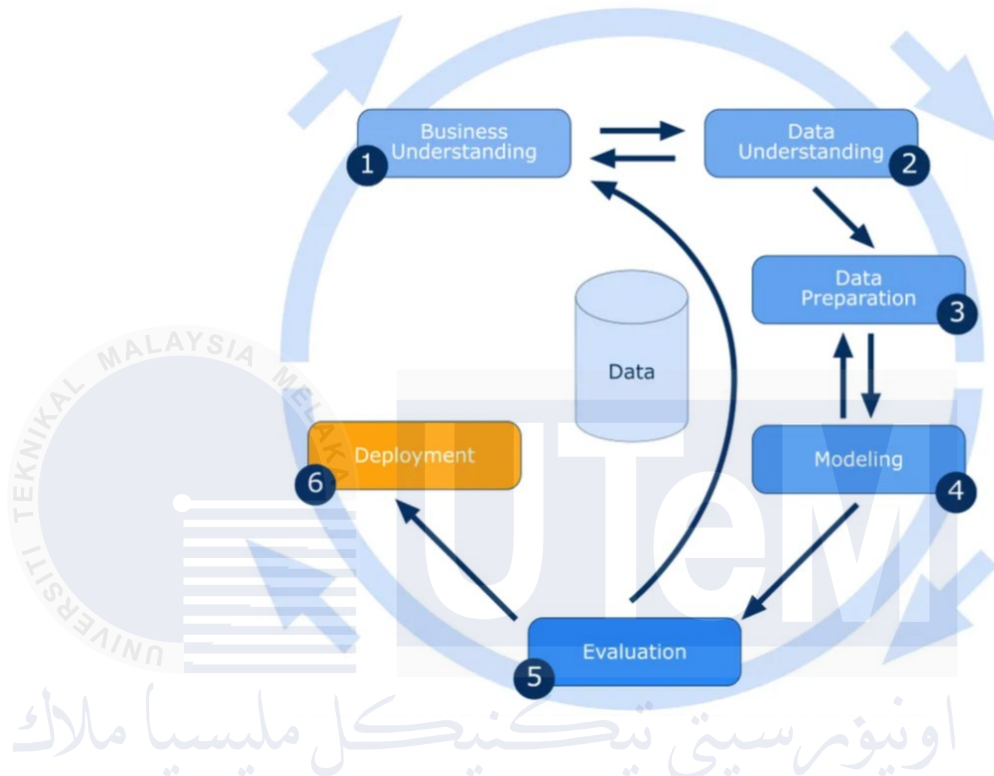
3.2 Operational Framework

The project's operational framework organizes the process of developing, deploying and assessing a model that could be used to predict biochar yield from pyrolysis food waste. It outlines the main steps, tasks and methods such that a thorough path to result is followed from start of idea until finish.

3.2.1 Phases of Methodology

The methodology for develop this system through which a predictive model for biochar yield from food waste pyrolysis is using the CRISP-DM approach is divided into distinct phases where each particular phase plays an important role in carrying out the research systematically and achieve project goal. The phases include: Business Understanding, define the objectives, assess situations, determine goals and produce a project plan to align with stakeholder needs and project goals; Data Understanding, where collect initial data descriptions of the data characteristics, explore the patterns within the features test designs generated integrate multiple sources format for modeling, clean missing values and verify the data quality; Data Preparation, which select appropriate data, clean the missing values and outliers, construct new features from existing ones format data for modeling; Modeling where appropriate modelling with selected techniques, test designs generated, models built and trained, and model performance assessed using relevant metrics; Evaluation, which will evaluate the model on defined metrics, review all models trained for alignment with business objectives, provide additional insights on how much a feature contributes to individual predictions and increase the understanding of decision-making processes in the models and determine next steps for improving or deploying; Deployment, where plan and execute deployment of the model, make plans establish,

monitoring and maintenance procedures, create a final report that documents everything that has been done and conduct project reviews assessing this as success tracks areas need improvement.



— **Figure 3.1: Methodology of Biochar Yield Prediction using CRISP-DM Model**
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

3.2.1.1 Business Understanding

Aim of in this initial phase is having understandable AI algorithms to predict biochar yield in pyrolysis of food waste along with the clear objectives. Business context consists of the stakeholders and goals to the predictive model (for example, how accurate should it be? How wide is interpretability here?). The remaining part of the work involves a literature review that is detailed with regards to biochar production, food waste pyrolysis and certain machine learning methodologies to guide future steps.

3.2.1.2 Data Understanding

The research, in the Data Understanding phase, starts by collecting a full dataset from food waste pyrolysis experimental studies. Fix carbon, Volatile matter and Ash content are the key characteristics along with Carbon (C), Hydrogen (H),

Oxygen (O) which are wet basis while Nitrogen (N), the dry count for residue baseN and Residence Time (min), Temperature ($^{\circ}\text{C}$), Heating rate ($^{\circ}\text{C}/\text{min}$), Feedstock Collection, Biochar Yield Statistics which have been extracted from a published paper namely “Biochar characterization and a method for estimating biochar quality from proximate analysis results”. The project then parses the data set and gives an insight on how these important factors in biochar yield process wise behave including numbers like degree Celsius, percentage and minutes. This allows us to interrogate the data systematically and discover patterns or anomalies in biochar yield relative to input variables. In parallel, data quality is checked on completeness, consistency and reliability are important to have a good starting point in the following stages of modelling with proper or improved input that can be coded over or analysed.

3.2.1.3 Data Preparation

In the data preparation phase, it helps in selecting only useful information and then it pre-processes that timely collected selected input data to make high quality of different kind datasets for well-the job satisfactory model performance like dealing with outliers that might affect predictions. Data is standardized across variables to present shared quantified features which uses Python’s libraries such as ‘Standard Scaler’ and ‘Logarithmic Transformation’ where the values are adjusted in such a way that they have scale 0 mean with unit variance for maintaining uniformity between them ensuring a clean dataset with the final step of modelling. It provides the standardization of a feature while scaling ensuring that it got converted to fall within certain range which makes easier for us like humans, machines to understand when is talking about same data or different. The biochar yield statistics are in percentage, which is good because it helps meaningful comparison of features and hence better appreciation on model metrics. It is a tedious process to prepare the dataset rigorously and many times quite lot of work goes into having it standardized so that the machine learning algorithms perform accurately during training or evaluation.

3.2.1.4 Modeling

The modelling phase is focused on applying selected machine learning models namely Linear Regression, Random Forest, K-Nearest Neighbors and Convolutional Neural Networks separately using a training/test split dataset to prevent bias

evaluation. These models are written using tooling and libraries such as Pandas, NumPy, TensorFlow/Keras and scikit-learn which employs classes like “LinearRegression”, “RandomForestRegressor”, “KNeighborsRegressor” in the field of machine learning, and use “Sequential”, “Dense”, “Input” for deep learning. The pipelined data is trained, and the patterns in it are learned by each model using different algorithms combined with evaluation metrics for MSE (mean squared error) and RMSE (root-mean-square error). These metrics aim to measure the performance of model in predicting for different algorithms and thus gives us a perspective on how well each model generalizes with new data. These initiatives coupled ensure that the obtained models not only forecast well, but also are interpretable outputs which would meaningful insights into how different features effect biochar yield from food waste pyrolysis.

3.2.1.5 Evaluation

In the evaluation phase, all selected machine learning models are evaluated individually by predefined metrics and interpretability scores to select appropriate model with best performance in biochar yield prediction from food waste pyrolysis. This arduous process of evaluation is intended to quantitatively determine how good models are with respect to project objectives, the business stakeholders. Quantitative tools of model predictability and fit scores such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) helps the business to understand which model achieves the highest level of performance and well suited. In addition, results interpretation extends from numerical metrics to the underlying factors driving biochar yield. Ranking features by importance to predict biochar production and SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) techniques for model interpretation. SHAP gives global feature importance across the dataset, whereas LIME offers local explanations for different predictions to test the model and understand specific instances of biochar yield.

3.2.1.6 Deployment

In the deployment phase, the aim is to incorporate biochar yield predictive model of utility food waste by pyrolysis based final best performing model into system operation. This involves establishing regular procedures to check how the model

behaves in real-world conditions and that it is still doing what we expect over time. Monitoring includes the use of key performance indicators (KPIs) along with testing model outputs in near real-time to catch deviations or any anomalies that might break out during execution. There are also strong maintenance procedures that aim to deal with issues like data drift, which can occur when input data changes over time and impacts model performance. These maintenance activities include regular updates and retraining of the model to ensure that it continues to be accurate in dynamic operational environments. Finally, a Summary Comprehensive Report that documents the entire project lifecycle from data collection and preprocessing steps up until model deployment is prepared. This report reflects lessons learned over the course of this project and provides information on accomplishments achieved, difficulties encountered, as well as opportunities for strengthening identified through the review process. Good documentation provides transparency and thus enables other stakeholders to understand in which ways the model's process directly contributes to these outcomes.

3.2.2 Data Collection

3.2.2.1 Data Sources

This study will source data from experimental work conducted under controlled laboratory conditions on food waste pyrolysis experiments. These experiments offer valuable information on Fixed Carbon, Volatile Matter, Ash content, along with fuel characteristics including Carbon (C), Hydrogen (H), Oxygen (O), Nitrogen (N) and heat flow parameters like Residence Time(min), Temperature(°C), Heating Rate (C/min), Feedstock Composition and Biochar Yield Statistics. Moreover, existing studies and databases related to biochar production and pyrolysis including those in the study " Biochar characterization and a method for estimating biochar quality from proximate analysis results," were referred to provide a detailed strength of literature for the research.

3.2.2.2 Data Collection Process

To get high quality data, this turns onto becoming very important because the effective model training and evaluation in data science begins by extensive

preprocessing steps. Any outliers that may contort the results of an analysis are pointed out and either fixed or removed ensuring validity in a dataset. Normalization of features including “Standard Scaler” and “Logarithmic Transformation” which using Python libraries standardizes the range for all variables like temperature and residence time, so that the different properties do not bias one over another. The Biochar requires biochar yield statistics be uniformity expressed as percentages to allow meaningful comparison and improved model quality assessment. These careful steps form a strong foundation for the rest of model-building and analysis phases ensures that data is now robust, standardized.

3.2.3 Tools and Techniques

3.2.3.1 Linear Regression

Tools Used:

- i. **Libraries:** Pandas, NumPy, Matplotlib, Plotly Express, Seaborn, Scikit-learn, SHAP, LIME
- ii. **Algorithm:** Linear Regression

Techniques:

- i. **Data Loading:** Data is loaded from a CSV file using Pandas, which is a powerful library for data manipulation and analysis in Python.
- ii. **Data Visualization:** In this step Histograms and box plots are used to visually explore the distribution of data on these attributes in order infer any potential outliers or patterns.
- iii. **Data Preprocessing:** Some pre-process steps as normalization and feature scaling are carried to have all features participating the fitting process equality, making sure that no particular feature can dominate if its magnitude.

- iv. **Model Training:** Linear Regression has been used to model the features selected and predict Biochar Yield (%) model. The model basically fits a linear equation to reduce the distance or loss between predicted and actual Biochar Yield values.
- v. **Evaluation:** The root mean squared error (RMSE) is the main evaluation metric for measuring average amount of errors between predicted and true Biochar Yield values. This would allow to see how the algorithm predicted as comparable to actual and spread it out in ScatterPlots, which is helpful for analyzing model performance.
- vi. **Feature Importance Analysis:** SHAP values are calculated to interpret which feature contributes how much to predicting Biochar Yield. The global interpretability is done by SHAP, which shows the average impact of each feature on all predictions. While LIME is for local explanation of the individual predictions explaining which feature plays critical role in specific cases.

3.2.3.2 Random Forest Classifier

Tools Used:

- i. **Libraries:** Pandas, NumPy, Matplotlib, Plotly Express, Seaborn, Scikit-learn, SHAP, LIME
- ii. **Algorithm:** Random Forest Classifier

Techniques:

- i. **Data Loading:** Like linear regression reading data is done through pandas for processing and analysing the related info.
- ii. **Data Visualization:** Box plot is used to visualize the distribution of data, knowing how much Biochar Yield has been spread across different categories.

- iii. **Data Preprocessing:** Data for the Random Forest Classifier training gets pre-processed by feature scaling and normalization so that every features makes an equal contribution in making predictions.
- iv. **Model Training:** In this step, the features and respective labels of Biochar Yield are used to train classifier that can classify it into desired categories.
- v. **Evaluation:** RMSE (Root Mean Squared Error) is calculated for the Random Forest model and then computing on class predictions probabilities or the outputs from a regression-style model.
- vi. **Feature Importance Analysis:** Calculate SHAP values for interpretability of feature impact on best features. SHAP gives us beautiful insights in to understanding which features are contributing how much for prediction stead of each class. Local Interpretable Model-agnostic Explanations (LIME) are used to generate local explanations highlighting which features matter most in classifying individual data points.

3.2.3.3 K-Nearest Neighbors (KNN) Regression

Tools Used:

- i. **Libraries:** Pandas, NumPy, Matplotlib, Plotly Express, Scikit-learn, SHAP, LIME
- ii. **Algorithm:** K-Nearest Neighbors (KNN) Regression

Techniques:

- i. **Data Loading:** Loading the data using Pandas to do basic exploratory analysis.
- ii. **Data Visualization:** Employing histograms and box plots to visualize the distribution Biochar Yield data and interpret characteristics of it.

- iii. **Data Preprocessing:** Similar to previous methods, normalization and feature scaling to prepare the data ready for K-NN Regression.
- iv. **Model Training:** The model used here is called as k-nearest neighbors which does classification on a similar principle for regression it predicts Biochar Yield (%) based upon its 'nearest' neighbors in feature space. For a given input, it will predict the output which is obtained by calculating the average of their value over neighbors.
- v. **Evaluation:** This measure helps us to make an assessment of prediction performance and is called RMSE (Root Mean Squared Error). Visualizations like scatter plots are useful in assessing the best model of model prediction for Biochar Yield with respect to actual values.
- vi. **Feature Selection:** SHAP values are calculated to identify how important each feature is for a model's predictions; we can get insights into which features impact the predicted Biochar Yield. LIME can provide local explanations to understand features mattered more during the class prediction for different records and explain why a model predicted in certain way on individual record.

3.2.3.4 Convolutional Neural Network (CNN)

Tools Used:

- i. **Libraries:** Pandas, NumPy, Matplotlib, Plotly Express, Seaborn, TensorFlow/Keras, SHAP
- ii. **Algorithm:** Convolutional Neural Network (CNN) - A type of Neural Network (Multi-layer Perceptron) for classification

Techniques:

- i. **Data Loading:** Data is in the Pandas, a signifies approach to pre-processing and Analysis.

- ii. **Data Visualization:** Histograms and box plots are used for visualization to understand the distribution, characteristics of Biochar Yield data as well as sources of variability in this kind of data or potential patterns.
- iii. **Data Preprocessing:** Normalization and feature scaling are important steps to do in it as well. Normalization will scales the values of all features down to a similar scale while feature scaling will settings all numerical feature ranges while not explicitly normalizing them. This is key for NNs during training ensuring their performance is optimal.
- iv. **Model Training:** TensorFlow/Keras is used for building and training the CNN model package with appropriate Keras version. This includes setting up layers like Convolutional2D and pooling which compiling the model with relevant loss and optimizer functions like mean squared error, Adam and fitting the data to those models on training set respectively.
- v. **Evaluation:** The CNN has been fed out with the appropriate dataset for being tested and root mean squared error (RMSE) is calculated, in order to measure how accurate will be predictions on Biochar Yield. This metric is used to evaluate the best features of correct predictions which made by the model.
- vi. **Feature Selection:** SHAP (SHapley Additive exPlanations) analysis is used to explain the importance of features with respect to CNN predictions. It is used to determine the most important input features such as temperature, pH and spatial arrangement of biochar that substantially affect classification Biochar Yield categories. LIME (Local Interpretable Model-agnostic Explanations) is the technique used to generate local explanations for CNN predictions that show what features were considered most important during specific individual prediction, assisting in visualizing how a decision was made by CNN

- vii. **Batch Size Threshold in CNN:** Find optimal batch size for training Convolutional Neural Networks (CNN) with TensorFlow/Keras. The batch size has implications on training efficiency and convergence rate as it strikes the fine line between memory consumption and computational overhead.

3.3 Project Milestones

3.3.1 PSM I

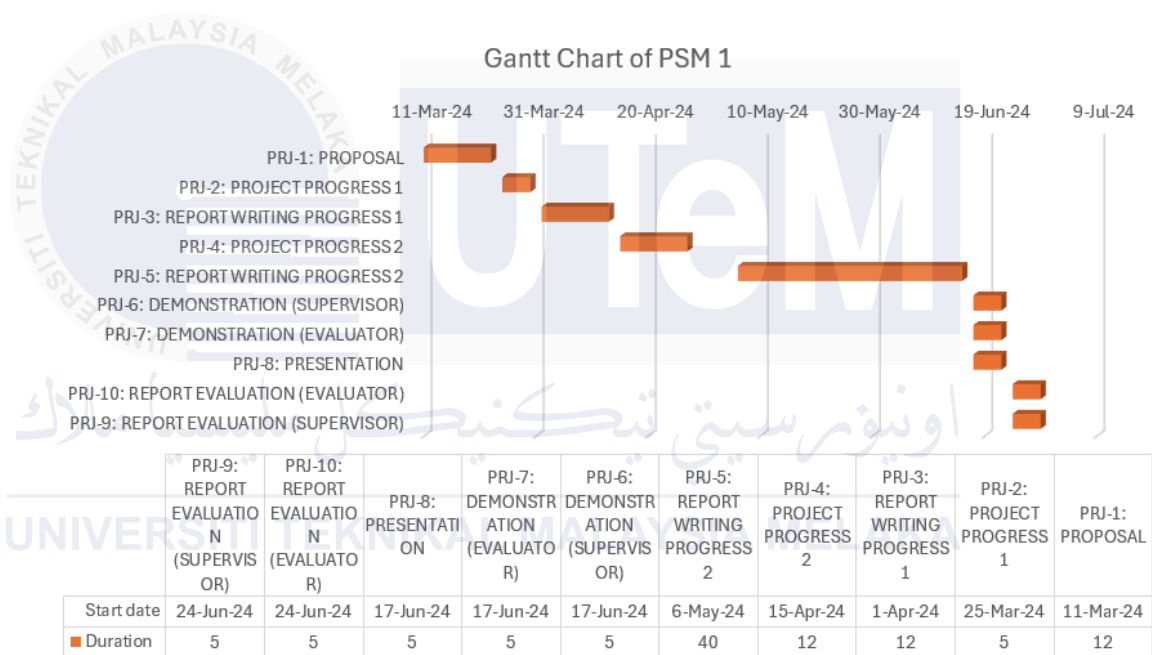


Figure 3.2: Gantt Chart of PSM 1

3.3.1.1 Stage 1: Proposal Development and Approval (11 March 2024 - 22 March 2024)

The first steps of the project is to construct a project proposal as per described in these instructions. This involves how to set context of the project, what are problem statements clearly provide details, objectives have been specific and well specifics have defined in scope part. The proposal will also detail the dates requested for this, and precisely describe what option they are looking to make these changes towards. After the proposal is worked out in detail, it will be reported to get approval. The final output of this stage is the project proposal form that has been approved.

3.3.1.2 Stage 2: Initial System Development Progress (25 March 2024 - 29 March 2024)

During this period, meticulous records will be kept and recorded, concentrating as much on punctuality and putting in effort under trying circumstances duly logged. Summarily, the products of this stage are all logs showing how far the development process.

3.3.1.3 Stage 3: Report Writing Progress 1 (1 April 2024 - 12 April 2024)

Writing and populating Chapters 1, 2 and 3 of the report with necessary data corresponding to each piece of research evidence. By the end of this stage, deliverables should include an initial working draft of Chapter 1-3 and log record showing progress in writing process.

3.3.1.4 Stage 4: System Development Progress 2 (15 April 2024 - 26 April 2024)

These are planned activities, which will work on the application/system being developed and making it functional. This phase will keep track of the progress and log it for further scenarios. Deliverables for the stage contributing detailed log files documenting for further development progress.

3.3.1.5 Stage 5: Report Writing Progress 2 (6 May 2024 - 14 June 2024)

In this phase, the activities will be concentrated around article writing and readying of Chapter 4 in report format. The project task will be focused on documenting well all findings and analysis related to the study. Following this stage will require a completed draft of Chapter 4 and log records on the process writing done.

3.3.1.6 Stage 6: Demonstration to Supervisor (17 June 2024 - 21 June 2024)

The main activity in this phase is to prepare and perform a project results demonstration with the supervisor. This is about to display how the project is going and show what it has achieved. Along to this, it is important that the entire demonstration process should be recorded and logged correctly. This phase will deliver log records explaining the example given to supervisor.

3.3.1.7 Stage 7: Demonstration to Evaluator (17 June 2024 - 21 June 2024)

Activities will focus on development and implementation of a project results demo, which targets the evaluator. The focus is going to be on a good demonstration of the claimed results and progress in work. It is important that the demonstration be carefully documented and each step of it is recorded or logged. At the end of this it will have detailed log entries for what was shown to an evaluator.

3.3.1.8 Stage 8: Presentation (17 June 2024 - 21 June 2024)

During this phase, the main time shall be spent in creating a powerful presentation that contains showcases about the project solution. That involves crafting more engaging presentations, creating interesting and attractive slide and being a little more polished. This should be meticulously record and log the process of making a presentation and ensure to have an audit trail with all the actions taken. Output at this stage should be a well-kept journal which captures the complete presentation process, from preparation to execution.

3.3.1.9 Stage 9: Report Evaluation by Supervisor (24 June 2024 - 28 June 2024)

During this phase, the work will center around submitting a draft report for review from the supervisor and then revising as required based on feedback. Recall it is important that all steps undertaken during the evaluation are recorded in detail. For this step, the deliverables are a set of detailed log entries describing how to apply your supervisor's evaluation of report and rewritten draft where you have incorporated supervisors' suggestions. This way, the report can be finalized with necessary standard completeness of detail and all issues or suggestions for changes have been addressed beforehand.

3.3.1.10 Stage 10: Report Evaluation by Evaluator (24 June 2024 - 28 June 2024)

During this stage, the activities will include submitting the draft report for evaluation by evaluator and then carry out any corresponding revisions. It is important that the entire evaluation process be fully documented and logged. Outputs for this phase include an in-depth log of the evaluation process during which evaluator interacted with the report, and another revising draft that has integrated frames to be

incorporated based on feedback provided. This process is designed to make the report compliant with a quality benchmark and can be appropriate if it has some suggestions before it finalizes.

3.3.2 PSM II

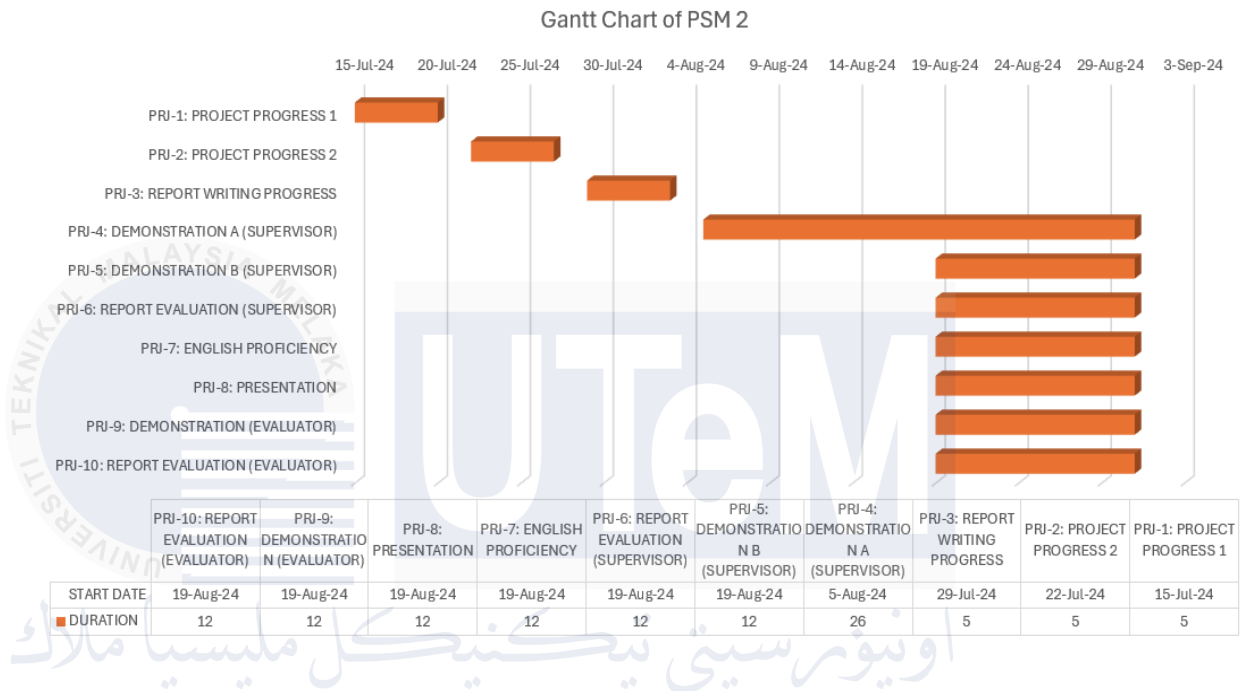


Figure 3.3: Gantt Chart of PSM II

3.3.2.1 Stage 1: System Development Progress 1 (15 July 2024 – 19 July 2024)

In the first iteration of a project, this was about bootstrapping application development or just setting up a system. This step also involved planning and getting the project management process started, focussing on being timely, dedicated as well as hard-working. This phase included initial coding, configuring the development environment and some basic testing. Problems like organizing tools and defining project objectives were recorded, with strategies planned to pass these challenges. Looking back, it was important to keep this disciplined mindset throughout in order to manage the time wisely and accomplish work from the get-go.

3.3.2.2 Stage 2: System Development Progress 2 (22 July 2024 - 26 July 2024)

The period of the second round was dedicated to orientating development and testing phase in relation to application or system. Based on the initial setup that was

established, this phase involved more advance developing which centred mostly around further functionality improvements and starting detail analyses. This stage involved more coding and feature testing with using explainable AI. These documents had the specifics of work-in-progress or some ways to go about solving complex tests. It was crucial for us to show traits like punctuality and dedication keeping the project on track up until now aligning with certain goals set earlier.

3.3.2.3 Stage 3: Report Writing Progress (29 July 2024 - 2 August 2024)

This stage focused on writing Chapters 5 and Chapter 6 of the report. In this stage the development work was synthesized and created a narrative around it by writing detailed methods descriptions, results and discussion sections. These chapters are wrote and revised to improve on feedback. The documentation of this progress was instrumental in monitoring improvement and potential gaps for clarification. At this stage it was recognized that to present findings and recommendations communication skills were required along with good writing.

3.3.2.4 Stage 4: Demonstration to Supervisor part A (5 August 2024 – 30 August 2024)

In this stage, the first show of final project was shown to supervisor. This is essentially showing the basic features and where they are at with their system. Critical feedback from the supervisor was needed to provide insights into everything one did well and all needing serious work. The demonstration was complete with notes such as recommendations from a supervisor and changes made. This feedback was an important tool to inform future work and make sure the project stood up when it had to.

3.3.2.5 Stage 5: Demonstration to Supervisor part B (19 August 2024 - 30 August 2024)

After the first demo, a second was scheduled to show off an improved and polished version of our project to our supervisor. This is an opportunity to show progress since the first demo, and even some new features that were developed. The feedback from the supervisor was critical again showing additional areas that needed to be polished before heading into the last evaluation. This time integration also created

the opportunity to maintain detailed logs of how the demonstration was conducted and what comments were given by the supervisor, allowing drive towards continuous improvement in showing alignment with project goals.

3.3.2.6 Stage 6: Report Evaluation by Supervisor (19 August 2024 - 30 August 2024)

At this point, the supervisor was presented with PSM2 draft report for review. The assessment was directed at evaluating how complete, clear and accurate the report is and overall feasibility of results displayed in project. The supervisor provided extensive guidance, pointing out both good and bad parts of the draft. After the intervention, this feedback was systematically recorded and revised as necessary. In this phase, the need for well-documented writing is necessary to ensure for a effective outcome of the project.

3.3.2.7 Stage 7: Checking of English proficiency (19 August 2024 - 30 August 2024)

During this stage, the skill to use English properly will be rated based on presentation. It was about the ability to speak well, be clear, articulate and communicate the thoughts in an easy way. There were practice sessions and looking for feedback to improve the English language. This stage was a reminder of the requirement to have excellent communication ability in getting across technical specifications.

3.3.2.8 Stage 8: Presentation (19 August 2024 - 30 August 2024)

In presentation stage, there should be present a detailed structured and interesting explanation to the evaluator. This involved structuring the presentation content logically, preparing informative slides and applying Audio Visual facilities to aid comprehension. Feedback from earlier demonstration and practice runs made it easy to shape the presentation. It was well documented as a presentation, what areas of the stage were used, how engaging it was visually and with audience. The subject of this phase was the role of presentation skills in successfully transmitting project results to the audience.

3.3.2.9 Stage 9: Demonstration to Evaluator (19 August 2024 - 30 August 2024)

This phase also included a final presentation of the finished project to the evaluator. The purpose was to show the final product that has been created or built, with all functions running now and talking about previous issues. This feedback from the evaluator was important for the project, in order to understand whether we were at final-stage completion and any last-minute tuning needed. The evaluator documented the demonstration as well as comments any subsequent actions. This highlighted the importance of going well prepared and receptive to feedback as we get to the closer stages of the complete project.

3.3.2.10 Stage 10: Report Evaluation by Evaluator (19 August 2024 - 30 August 2024)

In this final part, the evaluator reviewed the draft report from PSM2. This review offered a truly neutral criticism of what the report had to offer, how it was put together and presented. The results were recorded, emphasizing both the strengths and anything that could use further revision. This feedback enabled the report to be refined so that it was of a standard required and communicated well what the project had achieved. This stage also demonstrated the importance of external validation in establishing completeness and quality from project documentation.

3.4 Evaluation Metrics

Machine learning models constructing for the prediction of biochar yield from food waste pyrolysis, requires employing suitable performance metrics in order to evaluate and compare model efficiencies. Several metrics of the regression model can be calculated, which helps in analysing how well these models performs with respect to its prediction accuracy and reliability from different dimensions. In addition, including the performance measurements used in this project and reason why they were selected.

3.4.1 Mean Squared Error (MSE)

Mean Squared Error is simply the average of all these squares, and should get to know how far the multiple regression line (mid) moved away from actual value. The

way it reduces tax for bigger errors more than smaller ones makes them sensitive to any outliers.

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Figure 3.4: Formula of Mean Square Error (MSE)

Where:

y_i is the actual value

\hat{y}_i is the predicted value

n is the number of observations.

As it can be understood, there is a clear measure of average prediction error in the case used by MSE. It is especially valuable when errors are too expensive, causing a divergence of the predicted value than actual.

3.4.2 Root Mean Squared Error (RMSE)

Root Mean Squared Error (RMSE) is the square root of MSE. The error metric is in the same units as a target variable and can be interpreted like standard deviation of residuals

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

Figure 3.5: Formula of Root Mean Square Error(RMSE)

Where:

y_i is the actual value

\hat{y}_i is the predicted value

n is the number of observations.

RMSE is the preferred accuracy measure of choice for many applications that gives interpretable information on how poorly our rate prediction will perform in its predictive context. This is an approach that has a higher-level of error sensitivity, which makes it great to use for applications where large discrepancies are even more detrimental.

Using these various metrics together, this project able to assess the entire models and measure the impact of big errors (MSE / RMSE). Such a holistic performance measurement to select the best model for complete-biochar yield prediction due from food waste pyrolysis.

3.5 Summary

This chapter elaborated the methodology for building a model that is able to predict biochar yield and such prediction method can be divided into phases of Analysis, Design, Implementation, and Evaluation. It consists of the complete cycle from data collection until preprocessing, selecting appropriate algorithms and concluding with model training & evaluation along with assessment using performance metrics. Future directions include hyperparameter optimization, validating with additional real-world data and applying explainable AI techniques to improve model interpretability. This systematic methodology assures a reliable and strong predictive model in the studies of biochar yield from pyrolysis of food waste.

CHAPTER 4: PROPOSED METHOD

4.1 Introduction

The methodology proposed to conduct the research is outlined in this chapter. This chapter describes an organized plan to accomplish the research objectives identified in Chapter 1. The system of research developed is designed to ensure the rigor and efficiency in analysis, data collection and result-evaluation. This section is an introduction to the methodology framework discussed throughout this chapter.

4.2 Proposed Solution

4.2.1 Linear Regression

Simple and interpretable linear regression was used to model the relationship between the biochar yield (dependent variable) and pyrolysis conditions/feedstock characteristics (independent variable). This approach effectively assumed that there is a linear relationship in the input variable with respect to output and was helpful for getting an initial idea on factors how different variables have effect on biochar yield. Applying Linear Regression helped researchers to interpret the coefficients of each independent variable, presenting its effects on biochar yield. This made it easy to determine crucial factors involved with the process, which in turn helped build basic principles that could then be followed by more elaborate algorithms if required for further deduction and model eliminating.

```

# Linear Regression model for Biochar Yield
# Define the function to train and evaluate the linear regression model
def train_evaluate_lr(X_train, X_test, y_train, y_test):
    regressor = LinearRegression()
    regressor.fit(X_train, y_train)
    y_pred = regressor.predict(X_test)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    return rmse, y_test, y_pred, regressor

```

Figure 4.1: Linear Regression code from the project

4.2.2 Random Forest

The Random Forest was selected as the prediction algorithm for biochar yield, since it tends to perform robustly against overfitting and is highly efficient in capturing intricate non-linear relationships among predictors. This ensemble method formed multiple decision trees with training data and distributed the prediction overall to increase average accuracy and stability. As opposed to a single decision tree, the overfitting was reduced with Random Forest since it averaged the predictions of numerous trees and in this way decreased variance and enhanced generalisation on unseen data. This made it particularly well-suited to contexts when the relationship between pyrolysis conditions/feedstock characteristics and biochar yields may not have been direct or linear. Utilizing the forest of trees, Random Forest presented a tool that allows to intricately investigate and model interplay in dataset which enabled more insight and accurate prediction for food waste pyrolysis based Biochar yield.

```

# Random Forest Regressor model for Biochar Yield
# Train and evaluate the random forest regressor model
regressor = RandomForestRegressor(random_state=42)
regressor.fit(X_train, y_train)
y_pred = regressor.predict(X_test)
best_rmse = np.sqrt(mean_squared_error(y_test, y_pred))

```

Figure 4.2: Random Forest Regressor code from the project

4.2.3 K-Nearest Neighbors (KNN)

Biochar yield classification was performed with KNN for its non-parametric nature, classifying instances based on their similarity measures to the stored data points in the dataset. The method had advantages because it made no assumptions about the data distribution and hence was capable in capturing localized patterns, or relationships that parametric models might miss. KNN saved all cases and classified new instances by their proximities to the nearest neighbors, thus forming a direct and efficient way of prediction for biochar yield in pyrolysis experiments based on food waste. As the model relies on local similarity measures, it could flexibly adhere to data characteristics and support meaningful interpretation of biochar yield influencers; therefore, contributing to a global view about pyrolysis conditions together with feedstock properties.

```
# Function to train and evaluate KNN model
def train_evaluate_knn(X_train, X_test, y_train, y_test, n_neighbors):
    knn = KNeighborsRegressor(n_neighbors=n_neighbors)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)
    return rmse, y_pred, knn
```

Figure 4.3: K-Nearest Neighbors (KNN) code from project

4.2.4 Convolutional Neural Network (CNN)

Deep learning methods had been used to predict biochar yields using Convolutional Neural Networks (CNNs) where input features are treated as multidimensional data and can hence better capture the nonlinearities present in a dataset. The hierarchical representative strength of data boosted the performance, hence CNNs turned out to be appropriate for spatial-based tasks. These include images or multivariate sensor readings from a pyrolysis run. This was a good demonstration of the use-case for CNNs as they can unmask complex patterns and relationships that may not have been identified by other machine learning algorithms, because convolutional layers will identify features at different spatial scales. Sure enough, this adaptability that allows them to process large amounts of data and learn from spatial

relationships is what made CNNs a good candidate for predicting biochar yield reliably based on vastly different pyrolysis conditions as well as feedstock properties.

```

# Neural network model
model = Sequential([
    Input(shape=(X_train.shape[1],)),
    Dense(64, activation='relu'),
    Dense(64, activation='relu'),
    Dense(1) # Output Layer for regression
])

# Compile the model with mean squared error loss
model.compile(optimizer='adam', loss='mean_squared_error', metrics=[tf.keras.metrics.RootMeanSquaredError(name='rmse')])

# Train the model using training data
history = model.fit(
    X_train, y_train,
    epochs=100,
    validation_data=(X_test, y_test),
    verbose=1
)

```

Figure 4.4: Convolutional Neural Network (CNN) code from project

By implementing Linear Regression, Random Forest, KNN, and CNN, the proposed solution leveraged a diverse array of algorithms that complemented each other's strengths in predicting biochar yield from food waste pyrolysis. Linear Regression offered transparency and initial insights into the relationship between pyrolysis conditions/feedstock properties and biochar yield. Random Forest excelled in capturing non-linear relationships and interactions among predictors, while KNN identified localized patterns without assuming specific data distributions. CNN, designed for complex data structures, learned hierarchical features crucial for analyzing spatial and multi-dimensional sensor data inherent in pyrolysis experiments. The choice of algorithms was guided by the dataset's characteristics, the need for interpretability, and the complexity of predicting biochar yield. Evaluation using metrics such as Mean Squared Error (MSE), R-squared (R²), Accuracy, Root Mean Squared Error (RMSE), and interpretability metrics ensured the selection of the most suitable model(s). This comprehensive approach aimed to develop a robust biochar yield prediction model, enhancing understanding and decision-making in food waste pyrolysis through explainable artificial intelligence techniques.

4.2.5 SHAP (SHapley Additive exPlanations)

SHAP values provide a unified approach to interpreting model predictions by measuring the influence of each feature on prediction. From each model, SHAP was used to explain the effect of all features on biochar yield predictions. Especially where finding feature importance may not be as straight forward like in complex models such as Random Forest and CNN. SHAP summary plots which are helpful to visualize importance and effect of each feature across all observations, were produced in order to see the major factors that led more accurate prediction of Biochar Yield. Using SHAP, the study was able to understand that what features globally had influenced more conversion making it possible for a complete insight into data minus factors.

```
# ===== SHAP Implementation =====
# SHAP Analysis
# Ensure X_test_df is a DataFrame with column names before using it in SHAP
X_test_df = pd.DataFrame(X_test.values, columns=features)

# Use the Independent masker to avoid the deprecated feature_perturbation warning
masker = shap.maskers.Independent(X_train[features])

# Create the SHAP explainer using the masker
explainer = shap.LinearExplainer(train_evaluate_lr(X_train[features], X_test[features], y_train, y_test)[3], masker=masker)

# Calculate SHAP values
shap_values = explainer(X_test_df)

# Print SHAP values for a specific instance
instance_index = random.randint(0, X_test_df.shape[0] - 1)
shap_values_instance = shap_values[instance_index]

print(f"\nSHAP values for instance {instance_index}:")
for feature, feature_value, shap_value in zip(features, X_test_df.iloc[instance_index], shap_values_instance.values):
    print(f"{feature}: Feature Value = {feature_value:.4f}, SHAP Value = {shap_value:.4f}")

# Summary plot of SHAP values
shap.summary_plot(shap_values.values, X_test_df, feature_names=features)
```

Figure 4.5: SHAP (SHapley Additive exPlanations) code from project

4.2.6 LIME (Local Interpretable Model-agnostic Explanations)

LIME explained individual predictions locally in terms of the proximity to the instance under explanation. LIME accomplished this by varying the input data points and keeping track of how different those outputs varied. This has been particularly effective in understanding the local behavior of models such as KNN, which predictions are made based on closest neighbors. Using LIME, the study was able to interpret how these models make decisions on a local level and reveal input changes are driving related model responses for individual cases.


```

# ===== LIME Implementation =====
# LIME Analysis
# Ensure X_train and X_test are DataFrames with column names
lime_explainer = lime.lime_tabular.LimeTabularExplainer(
    X_train[features].values,
    mode='regression',
    feature_names=features,
    verbose=True,
    random_state=20
)

# Select an instance from the test set to explain
i = random.randint(0, X_test.shape[0] - 1)
exp = lime_explainer.explain_instance(X_test.iloc[i].values, regressor.predict, num_features=len(features))

# Print the feature values for the selected instance
print(f"\nFeature values for the selected instance (Index {i}):")
for feature, value in zip(features, X_test.iloc[i].values):
    print(f"{feature}: {value:.4f}")

# Print the LIME explanation
print(f"\nLIME explanation for the selected instance (Index {i}):")
for feature, explanation in exp.as_list():
    print(f"{feature}: {explanation}")

# Visualize the LIME explanation
exp.show_in_notebook(show_table=True)

```

Figure 4.6: LIME (Local Interpretable Model-agnostic Explanations) code from project

4.3 Experiment Design

Generally, on the development of an explainable artificial intelligence (XAI) based biochar yield prediction model from food waste pyrolysis experiment design follows a structured pathway to receive reliable and reproducible results. Overall Flow Breakdown:

4.3.1 Data Collection and Preprocessing

The data in this study were obtained from the analysis of different experimental data on food waste pyrolysis, including fixed carbon content, volatile matter, temperature and residence time (RT) as other variables. The collected data, in turn, also underwent a long procedure of prepossessing. These steps involve removing inconsistencies and errors (cleaning), dealing with missing values by using imputation techniques, scaling numeric features using ‘Standard Scaler’ so that they have consistent scales across different variables. The skewed data were normalized after applying ‘Logarithmic Transformations’ on them, to prepare this for modelling. Then,

the categorical variables has been converted to numerical coding format for better analysis. With the way of pre-processed dataset with these many steps, the data was ready to be modelled and predicted accurately on how much biochar can obtained in food waste pyrolysis.

```
# Apply standard scaling to the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Convert scaled features back to DataFrame for easier manipulation
X_scaled = pd.DataFrame(X_scaled, columns=features)
```

Figure 4.7: Standard Scaler code from the project

```
# Apply log transformation to the features
X_log_transformed = X.copy()
for col in X_log_transformed.columns:
    if X_log_transformed[col].min() <= 0:
        X_log_transformed[col] = np.log1p(X_log_transformed[col])
    else:
        X_log_transformed[col] = np.log(X_log_transformed[col])
```

Figure 4.8: Logarithmic Transformation code from project

4.3.2 Feature Selection and Engineering

The aim of the feature selection in this study is to select a subset of most relevant features affecting biochar yield. Correlation analysis, feature importance from models and domain knowledge were techniques used to guide this process. Then the data was moved to feature engineering where they converted features into new more powerful than before or engineered some totally new based on what models needed. These procedures contributed to setting several of the most informative, and one could argue the facultative influential features rendering models capable to predict better biochar yield in place specific pyrolysis experiments within food waste.

4.3.3 Model Selection and Training

For algorithm selection in this study, suitable XAI algorithms including Linear Regression, Random Forest, K-Nearest Neighbors (KNN), and potentially Convolutional Neural Networks (CNN) were chosen for comparison. Following algorithm selection, the data was split into training and testing sets. Each selected model was then trained on the training data using cross-validation techniques to optimize hyperparameters. This approach ensured that the models were effectively trained and evaluated using robust methodologies, preparing them for comprehensive performance evaluation and comparison in predicting biochar yield from food waste pyrolysis experiments.

In this study, Linear Regression, Random Forest and K-Nearest Neighbors (KNN) were selected as the XAI algorithms for algorithm selection along with Convolutional Neural Networks (CNN). After selecting the algorithm, data was divided into training and testing sets. Then, each of the model has been trained on training data. During training, a random feature selection is selected and model performance was evaluated validation to select best features or interactions with respect to the RMSE on test data. Thus, preparing the trained and validated models for a full-scale performance evaluation and comparison in prediction of biochar yield from food waste pyrolysis experiments.

4.3.4 Evaluation

Performance of models was evaluated with mean squared error (MSE), root mean squared error (RMSE) and interpretability metrics like SHAP(Shapley Additive exPlanations) and LIME(Local Interpretable Model-agnostic Explanations). The RMSE or the square root of the mean squared error is an important benchmark for evaluating how well the model predicted from real predictions, and in testing biochar yields it was used to measure errors. Moreover, both SHAP and LIME were able to tell the overall importance of features (SHAP) or which individual data points contribute largely on decisions made by a certain model. This set of metrics gave a full picture about the predictive performance and interpretability ability for each algorithm. These models were assessed and compared with the goal of selecting algorithms that provided a balance between good performance in predicting biochar yield from food

waste pyrolysis experiments, while being interpretable. This rigorous comparison process was carried out for models that would not only have high predictive quality but also provide explanations on how the model predicted response related to other variables and this is important when make decisions in pyrolysis applications based upon an output.

4.3.5 Experimental and Simulation Setup

Dataset that has been used was obtained from data of food waste pyrolysis experiments. Python libraries like pandas for data manipulation, scikit-learn for machine learning and TensorFlow/Keras required to implement Convolutional Neural Networks (CNNs). Hardware Requirements were met by providing necessary computational resources, such as GPUs to handle the training process of complex model like CNNs. Data splitting was used to segregate the data into training set and testing set with same biochar yield distribution. Since, the above approach could have been implemented random feature selection to make sure that having durable model evaluation and optimize the tuning of parameters. XAI techniques were use during the training and evaluation phases to understand feature importance, model prediction.

4.4 Summary

Chapter 4 of the report describe a methodology to develop a prediction model for biochar yield in food waste pyrolysis using different machine learning approaches. The approach investigated slightly different algorithms like Linear Regression, Random Forest Classifier, K-Nearest Neighbor (KNN) and a possible application of Convolutional Neural Networks (CNN) for each due to their individual strengths with regard to regression. Data Collection, Data Preprocessing, Feature Selection and XAI part such as SHAP (Shapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) has been involved. After the results of preprocessing the data, the models were trained and evaluated over carefully with selected metrics like Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) to ensure most accurate and interpretable models. Future work will be to incorporate interpretability such as SHAP values and LIME for model interpretability and deep dive into explainable AI and investigating for the best model among four by thoroughly documenting findings with performance metrics and feature importance knowledge.

CHAPTER 5: RESULTS AND DISCUSSION

5.1 Introduction

In this chapter, the biochar yield prediction models implemented with Linear Regression, Random Forest Regressor, K-Nearest Neighbors (KNN) and Convolutional Neural Networks (CNN) are evaluated. Linear Regression gave us a straightforward baseline for looking at linear relationships and Random Forest found all the interactions of our non-linear elements which variables were most predictive. KNN worked well to model non-linear relationships for small instances and CNN went hand in hand with dealing complex patterns required from high-dimensional data. The performance of the models was measured using Root Mean Squared Error (RMSE) as evaluation metrics. Interpretability was improved through SHAP and LIME which provided feature importance and local explanations in more detail. In this chapter, we consider both the predictive accuracy and interpretability of each model in predicting biochar yield from food waste pyrolysis.

5.2 Evaluation Result

5.2.1 Linear Regression

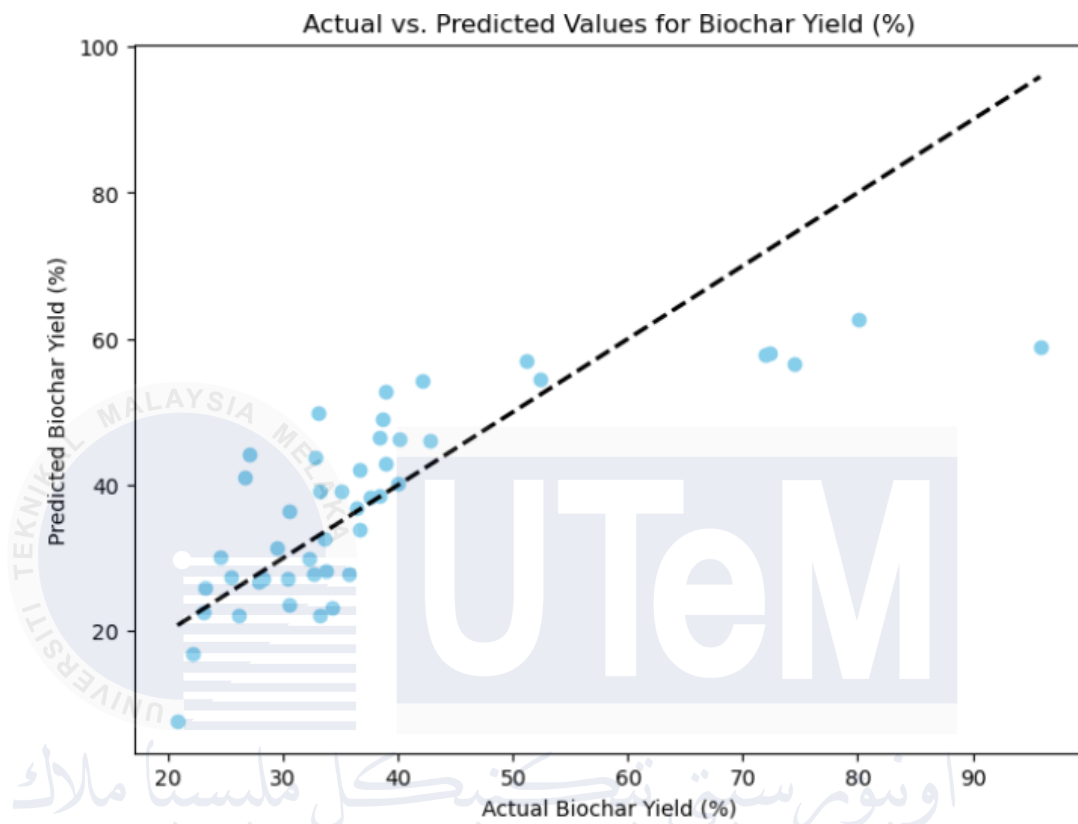


Figure 5.1: Actual Value vs Predicted Value for Biochar Yield for Linear Regression

Table 1: Overall and Average of RMSE value when run multiple time using Standard Scaler

	Run	Best RMSE			
0	1	10.186592			
1	2	10.266711			
2	3	10.284418			
3	4	10.294323			
4	5	10.294309			
5	6	10.294309			
6	7	10.294309			
7	8	10.294309			
8	9	10.153937			
9	10	10.277922			
10	11	10.284418			
11	12	10.125836			
12	13	10.153938	39	40	10.137437
13	14	10.294309	40	41	10.294323
14	15	10.294323	41	42	10.294323
15	16	10.294309	42	43	10.294323
16	17	10.265493	43	44	10.294309
17	18	10.294323	44	45	10.294323
18	19	10.294322	45	46	10.294323
19	20	10.266711	46	47	10.294323
20	21	10.294323	47	48	10.294323
21	22	10.130385	48	49	10.294309
22	23	10.284388	49	50	10.294323
23	24	10.284388	50	51	10.294323
24	25	10.294323	51	52	10.163850
25	26	10.294309	52	53	10.294323
26	27	10.153938	53	54	10.266711
27	28	10.263093	54	55	10.148712
28	29	10.266711	55	56	10.294309
29	30	10.098492	56	57	10.166677
30	31	10.153937	57	58	10.294323
31	32	10.284388	58	59	10.277922
32	33	10.294309	59	60	10.294323
33	34	10.294323	60	61	10.294323
34	35	10.294323	61	62	10.294323
35	36	10.294323	62	63	10.173058
36	37	10.294323	63	64	10.266711
37	38	10.294323	64	65	10.277922
38	39	10.294323	65	66	10.294309
			66	67	10.155085
			67	68	10.284389
			68	69	10.155181
			69	Average	10.258601

Table 2: Overall and Average of RMSE value when run multiple time using Logarithmic Transformation

	Run	Best RMSE			
0	1	8.979120			
1	2	8.941594			
2	3	8.928545			
3	4	8.794432			
4	5	8.979120			
5	6	8.781245			
6	7	8.979120	39	40	8.781245
7	8	8.898211	40	41	8.916691
8	9	8.979120	41	42	8.847195
9	10	8.979120	42	43	8.781245
10	11	8.979120	43	44	8.943513
11	12	8.979120	44	45	8.979120
12	13	8.979120	45	46	8.929497
13	14	8.809154	46	47	8.811926
14	15	8.930305	47	48	8.809493
15	16	8.979120	48	49	8.903376
16	17	8.798727	49	50	8.979120
17	18	8.964353	50	51	8.933853
18	19	8.950674	51	52	8.813528
19	20	8.979120	52	53	8.979120
20	21	8.979120	53	54	8.979120
21	22	8.781245	54	55	8.964353
22	23	8.975707	55	56	8.979120
23	24	8.781245	56	57	8.893027
24	25	8.968507	57	58	8.928545
25	26	8.979120	58	59	8.964353
26	27	8.979120	59	60	8.814534
27	28	8.979120	60	61	8.826865
28	29	8.781245	61	62	8.979120
29	30	8.847195	62	63	8.979120
30	31	8.781245	63	64	8.781245
31	32	8.979120	64	65	8.979120
32	33	8.781245	65	66	8.979120
33	34	8.964353	66	67	8.979120
34	35	8.860267	67	68	8.964353
35	36	8.827033	68	69	8.979120
36	37	8.979120	69	70	8.254582
37	38	8.907410	70	71	8.979120
38	39	8.809154	71	Average	8.903407

As per the Linear Regression Model, the average RMSE value using Standard Scaler is 10.2686 which can be seen from above diagram as well. However, when using a log transformation for the features as input data we can see that RMSE drops significantly to an average value of 8.9034. This means the logarithmic transformation can get a more preferable and lower RMSE model performance. In general, the random process yield lower the RMSE values when Log transformation was applied to the feature selection and this approach is proven better in terms of improving model accuracy relative compared to Standard Scaler.

5.2.2 Random Forest Regression

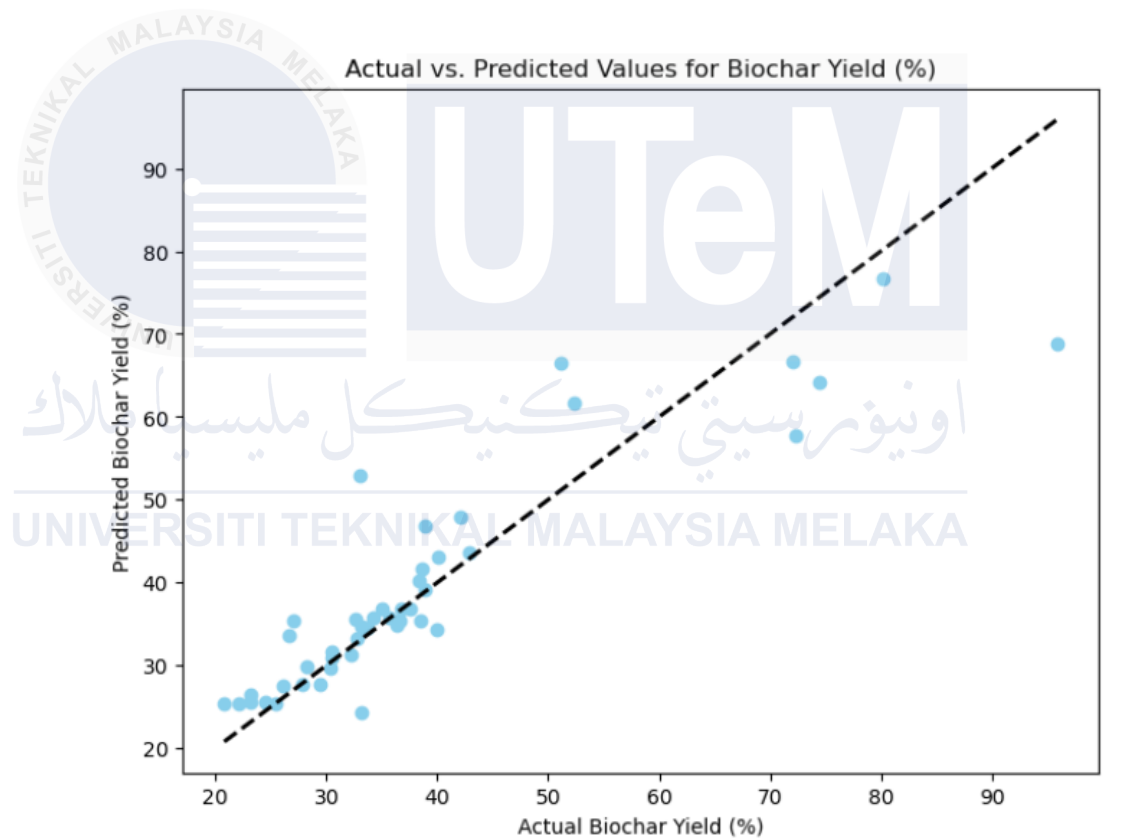


Figure 5.2: Actual Value vs Predicted Value for Biochar Yield for Random Forest Regressor

Table 3: Overall and Average of RMSE value when run multiple time using Standard Scaler

	Run	Best RMSE	28	29	6.664611	57	58	6.566176
0	1	6.137324	29	30	6.697140	58	59	5.061331
1	2	6.702660	30	31	6.742883	59	60	5.098097
2	3	6.605978	31	32	6.193023	60	61	5.438372
3	4	6.331358	32	33	6.719015	61	62	6.912355
4	5	6.177725	33	34	6.791590	62	63	6.251946
5	6	5.329729	34	35	6.065433	63	64	5.543772
6	7	5.759344	35	36	6.238681	64	65	6.963111
7	8	4.462432	36	37	5.783741	65	66	6.933152
8	9	6.457155	37	38	6.557654	66	67	7.007645
9	10	6.758370	38	39	6.647384	67	68	5.170372
10	11	6.811795	39	40	6.583203	68	69	6.890458
11	12	6.745804	40	41	6.562624	69	70	6.980043
12	13	6.794347	41	42	6.705499	70	71	6.747484
13	14	6.849751	42	43	4.786401	71	72	6.646112
14	15	6.397999	43	44	6.525670	72	73	6.224918
15	16	6.747316	44	45	6.525670	73	74	6.655479
16	17	6.612387	45	46	6.717048	74	75	6.879236
17	18	5.546921	46	47	6.544046	75	76	5.049300
18	19	6.891662	47	48	6.760818	76	77	6.546386
19	20	6.797719	48	49	6.077757	77	78	5.884793
20	21	6.764342	49	50	6.807647	78	79	6.838243
21	22	5.981385	50	51	6.807647	79	80	6.727911
22	23	6.894837	51	52	4.391107	80	81	6.511391
23	24	5.753081	52	53	6.545602	81	82	4.833302
24	25	6.222559	53	54	6.768738	82	83	6.567010
25	26	6.811247	54	55	6.381175	83	84	6.563095
26	27	6.748659	55	56	6.590542	84	85	6.398262
27	28	6.601356	56	57	6.546704	85	86	6.654329
			57		6.625194	86	87	6.113564
			58		6.171680	87	Average	6.351098

Table 4: Overall and Average of RMSE value when run multiple time using Logarithmic Transformation

			28	29	6.815740	58	59	5.489657
	Run	Best RMSE	29	30	6.881308	59	60	5.170372
0	1	5.030165	30	31	6.505411	60	61	6.859608
1	2	6.884670	31	32	6.650881	61	62	6.594781
2	3	6.418849	32	33	6.602396	62	63	6.778796
3	4	6.555638	33	34	6.312669	63	64	6.900685
4	5	6.579315	34	35	6.909705	64	65	6.634691
5	6	6.478076	35	36	5.949695	65	66	6.111240
6	7	6.784336	36	37	6.721397	66	67	6.849944
7	8	6.841298	37	38	6.805029	67	68	6.855250
8	9	6.786590	38	39	6.880015	68	69	6.904963
9	10	6.885807	39	40	6.100530	69	70	6.383218
10	11	6.691812	40	41	6.014325	70	71	6.830694
11	12	5.790659	41	42	6.431356	71	72	6.459022
12	13	6.694976	42	43	7.007645	72	73	6.809465
13	14	6.696525	43	44	4.867973	73	74	5.944133
14	15	4.533143	44	45	6.345317	74	75	6.749578
15	16	6.772495	45	46	6.740732	75	76	5.926698
16	17	6.674629	46	47	6.561114	76	77	5.904126
17	18	6.838152	47	48	6.260909	77	78	4.392496
18	19	4.391107	48	49	6.147654	78	79	6.466177
19	20	6.864206	49	50	5.858881	79	80	6.712653
20	21	6.849334	50	51	6.748480	80	81	6.784925
21	22	6.545714	51	52	6.489914	81	82	6.876662
22	23	5.017569	52	53	5.780192	82	83	6.845189
23	24	5.141105	53	54	6.711350	83	84	6.627718
24	25	6.563639	54	55	6.934705	84	85	6.306134
25	26	6.556993	55	56	6.328618	85	86	6.220290
26	27	4.392496	56	57	6.786243	86	87	6.989785
27	28	6.621900	57	58	6.910375	87	Average	6.382997

As per the outcome of Random Forest model with standard Scaler, Root Mean Squared Error value is around 6.3511 on average as comparison, for the logarithmic transformation, it is greater which is about 6.3830 on average RMSE value. This RMSE number with logarithmic transformation is marginally higher implying Random Forest model works slight better in this current case, which was not the expectation after applying logistic regression. As seen above, in case of Standard Scaler feature selection improved overall performance with random changes as well and this was very similar to the results found for Linear Regression. This means for Random Forest, Standard Scaler can be a marginally better option than log transformation but the gap is small.

5.2.3 K-Nearest Neighbors (KNN)

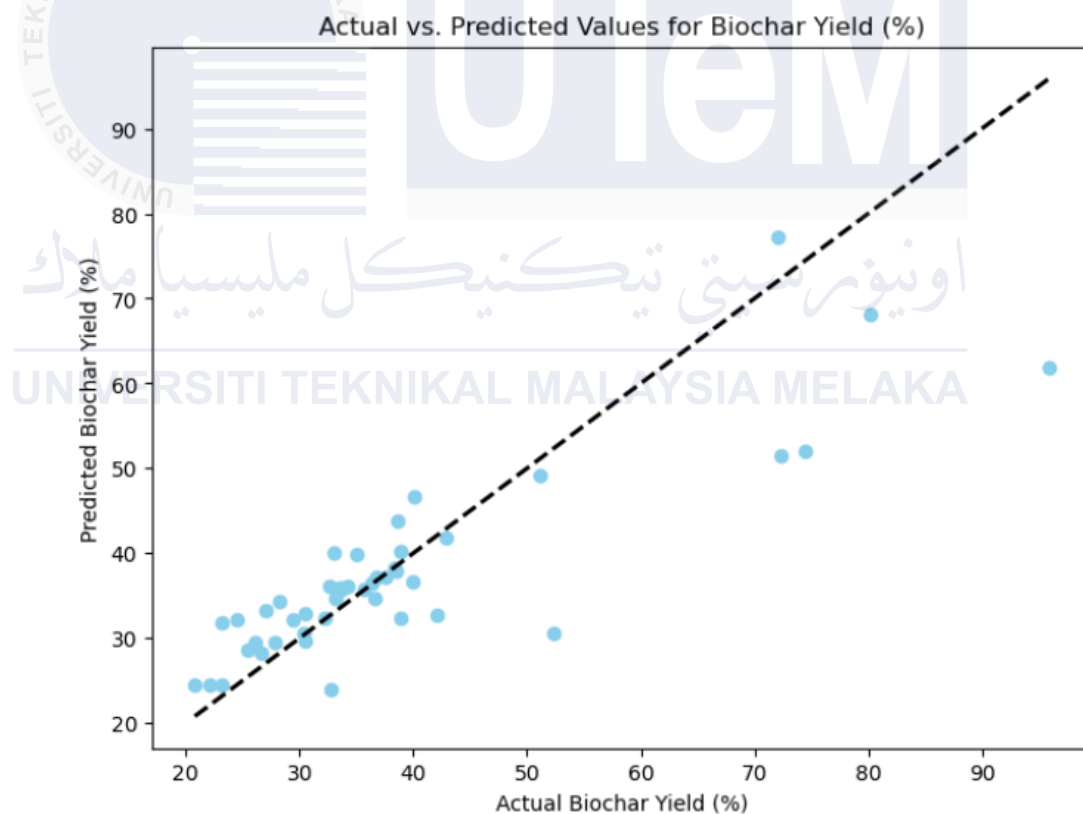


Figure 5.3: Actual Value vs Predicted Value for Biochar Yield for K-Nearest Neighbors

Table 5: Overall and Average of RMSE value when run multiple time using Standard Scaler

Run	Best RMSE	29	30	7.811261				
0	1	9.201028	30	31	8.685551	59	60	8.682549
1	2	10.327762	31	32	10.355727	60	61	7.030002
2	3	6.870021	32	33	10.532202	61	62	7.778748
3	4	8.976269	33	34	10.112549	62	63	7.372051
4	5	7.809172	34	35	7.778748	63	64	6.688243
5	6	10.146283	35	36	10.166706	64	65	7.664051
6	7	10.527360	36	37	7.807509	65	66	10.156612
7	8	10.482498	37	38	10.116601	66	67	8.854060
8	9	7.155332	38	39	10.527360	67	68	7.472502
9	10	7.288450	39	40	9.218094	68	69	6.447305
10	11	10.532202	40	41	10.156612	69	70	7.194297
11	12	7.811261	41	42	6.688148	70	71	8.685551
12	13	7.194297	42	43	7.516042	71	72	10.166706
13	14	7.573734	43	44	9.687767	72	73	10.527360
14	15	10.070332	44	45	10.355727	73	74	8.373685
15	16	7.040508	45	46	10.542843	74	75	7.931183
16	17	8.247277	46	47	6.450046	75	76	6.688148
17	18	8.487698	47	48	7.108476	76	77	10.077613
18	19	10.516703	48	49	7.576198	77	78	7.586990
19	20	6.450046	49	50	8.973219	78	79	6.482758
20	21	7.701783	50	51	7.370692	79	80	10.516703
21	22	6.757343	51	52	9.978277	80	81	10.161691
22	23	8.854060	52	53	7.931183	81	82	7.600836
23	24	7.659877	53	54	8.656416	82	83	7.472502
24	25	10.388438	54	55	10.161691	83	84	10.527360
25	26	10.527360	55	56	6.652746	84	Average	8.717492
26	27	10.355727	56	57	10.542843			
27	28	10.516703	57	58	9.690487			
28	29	9.262454	58	59	10.246084			

Table 6: Overall and Average of RMSE value when run multiple time using Logarithmic Transformation

	Run	Best RMSE	29	30	12.643378			
0	1	12.643378	30	31	7.347188			
1	2	6.711412	31	32	7.451298			
2	3	6.078416	32	33	7.022854			
3	4	8.039617	33	34	6.399494			
4	5	7.729212	34	35	5.623495	59	60	6.085419
5	6	7.825766	35	36	6.899066	60	61	7.987268
6	7	6.080087	36	37	5.709550	61	62	11.130754
7	8	6.996255	37	38	9.885598	62	63	7.053633
8	9	7.893289	38	39	7.892349	63	64	8.627404
9	10	6.243681	39	40	11.143627	64	65	12.620228
10	11	7.681924	40	41	8.626315	65	66	5.940610
11	12	12.643378	41	42	7.228968	66	67	11.112864
12	13	8.592195	42	43	6.453608	67	68	5.727967
13	14	12.630558	43	44	12.643378	68	69	12.643378
14	15	12.643378	44	45	7.738120	69	70	9.958228
15	16	7.900145	45	46	10.432531	70	71	9.041608
16	17	11.155948	46	47	12.643378	71	72	6.560957
17	18	6.543236	47	48	10.423269	72	73	6.062524
18	19	5.891601	48	49	7.850852	73	74	11.799913
19	20	12.239608	49	50	10.527360	74	75	11.317163
20	21	11.155948	50	51	8.049146	75	76	6.453608
21	22	6.859294	51	52	12.630558	76	77	8.626315
22	23	11.799913	52	53	11.799913	77	78	10.898478
23	24	8.592195	53	54	12.643378	78	79	12.643378
24	25	11.799913	54	55	10.339822	79	80	7.171688
25	26	6.172742	55	56	12.620228	80	Average	9.011083
26	27	6.973298	56	57	7.112585			
27	28	8.627404	57	58	6.323281			
28	29	8.544290	58	59	12.630558			

The average RMSE value is around 8.7175 using Standard Scaler for the K-Nearest Neighbors (KNN) model. On the other hand, logarithmic transformation causes a small increase in average RMSE of about 9.0111. Both sets of results indicate that Standard Scaler seems to work better than logarithmic transformation in general, since it gives a lower RMSE. This is saying generally that features should give a better prediction with Standard Scaler and hence it is practice to improve the model than using logarithmic transformation.



5.2.4 Convolutional Neural Networks (CNN)

Table 7: Overall and Average of RMSE value when run multiple time using Standard Scaler

	Run	Best RMSE	29	30	6.865659			
0	1	6.928016	30	31	7.067171			
1	2	7.015270	31	32	7.051893			
2	3	7.017079	32	33	6.787346			
3	4	7.103387	33	34	7.082427			
4	5	6.941518	34	35	7.161809			
5	6	6.908841	35	36	6.979697	59	60	6.876480
6	7	6.735767	36	37	7.031394	60	61	7.540692
7	8	6.855403	37	38	6.977742	61	62	6.799094
8	9	7.100526	38	39	7.367711	62	63	6.939961
9	10	6.970807	39	40	6.704933	63	64	7.162079
10	11	6.886338	40	41	7.160467	64	65	6.886379
11	12	6.821609	41	42	7.061657	65	66	7.013362
12	13	7.184119	42	43	6.904274	66	67	6.908490
13	14	7.062297	43	44	6.778062	67	68	6.894925
14	15	6.749469	44	45	6.856755	68	69	6.945793
15	16	7.099350	45	46	6.692738	69	70	7.285939
16	17	6.613734	46	47	6.823634	70	71	6.622893
17	18	6.855224	47	48	6.903681	71	72	6.833281
18	19	6.631726	48	49	6.984444	72	73	6.698282
19	20	7.084328	49	50	7.303879	73	74	6.929729
20	21	6.740453	50	51	6.814573	74	75	7.000171
21	22	6.796952	51	52	6.775898	75	76	7.218556
22	23	6.686361	52	53	7.252034	76	77	6.642767
23	24	6.962599	53	54	7.071182	77	78	7.058409
24	25	6.835875	54	55	6.719457	78	79	6.802965
25	26	6.822113	55	56	6.997399	79	80	6.952962
26	27	6.981812	56	57	7.182754			
27	28	6.672361	57	58	6.887490			
28	29	7.223581	58	59	6.976065	80	Average	6.944054

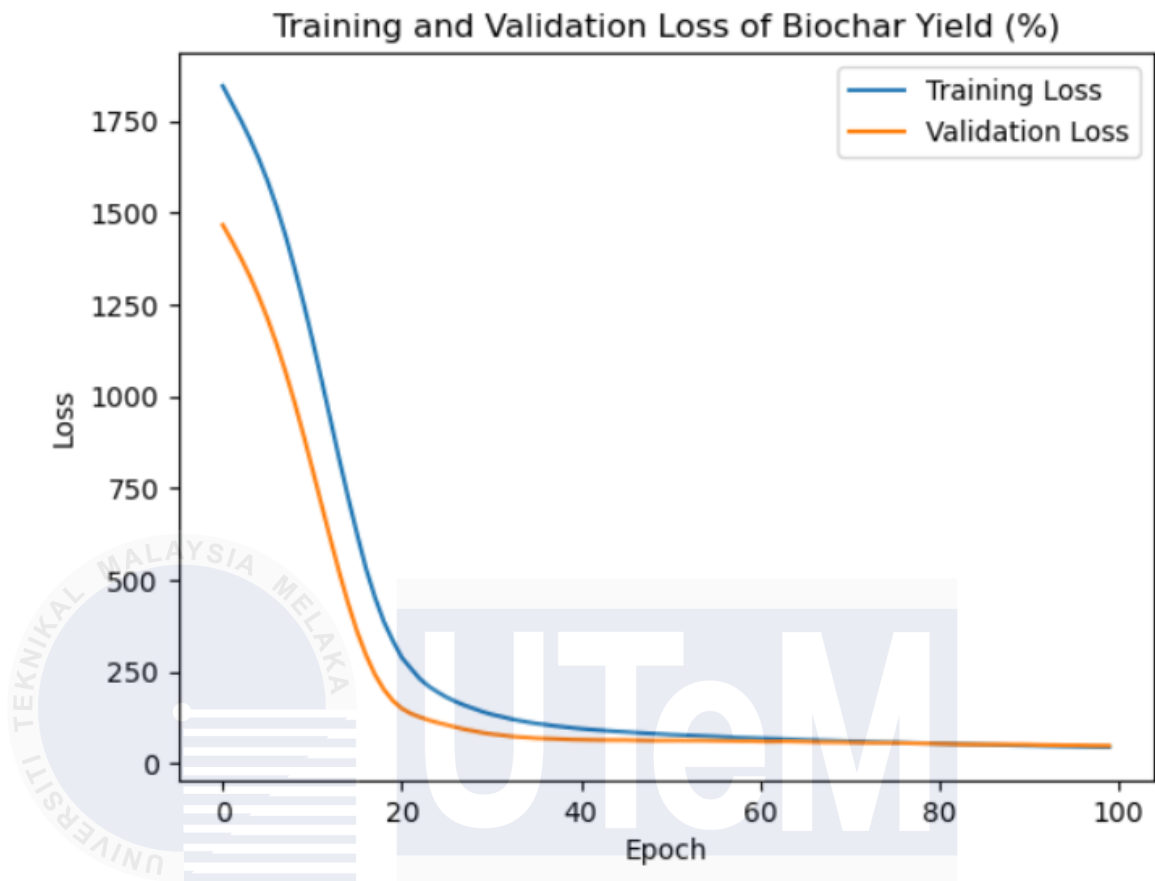


Figure 5.4: Training and Validation Loss of Biochar Yield using Standard Scaler

Table 8: Overall and Average of RMSE value when run multiple time using Logarithmic Transformation

	Run	Best RMSE						
			28	29	13.168929			
0	1	12.979734	29	30	12.421346			
1	2	13.296189	30	31	12.412663			
2	3	12.821953	31	32	12.702682	58	59	13.269532
3	4	12.349816	32	33	13.281459	59	60	12.949560
4	5	12.903823	33	34	12.535395	60	61	13.215865
5	6	12.867343	34	35	12.812798	61	62	13.373423
6	7	12.706307	35	36	13.177588	62	63	12.800193
7	8	12.857395	36	37	13.175197	63	64	12.815754
8	9	12.050443	37	38	12.273734	64	65	12.975401
9	10	12.727583	38	39	12.793029	65	66	12.783844
10	11	13.242082	39	40	13.233294	66	67	12.929899
11	12	12.999712	40	41	13.063075	67	68	12.903371
12	13	12.879036	41	42	12.964856	68	69	12.773036
13	14	12.816498	42	43	12.571520	69	70	13.029124
14	15	12.770999	43	44	13.050246	70	71	13.134246
15	16	12.845024	44	45	13.087677	71	72	12.656156
16	17	13.143871	45	46	13.043354	72	73	12.763782
17	18	12.928412	46	47	12.899956	73	74	12.589330
18	19	13.124803	47	48	12.899647	74	75	13.031730
19	20	12.770965	48	49	13.063150	75	76	12.842882
20	21	13.268781	49	50	13.036495	76	77	13.010939
21	22	12.500532	50	51	12.890869	77	78	12.910804
22	23	12.905978	51	52	13.117847	78	79	13.273375
23	24	13.048677	52	53	12.875139	79	80	12.626206
24	25	13.068684	53	54	12.932364	80	81	13.061105
25	26	12.999420	54	55	13.189238	81	82	13.435908
26	27	13.308940	55	56	13.099367			
27	28	13.502343	56	57	12.513835			
28	29	13.168929	57	58	13.040727	82	Average	12.928881

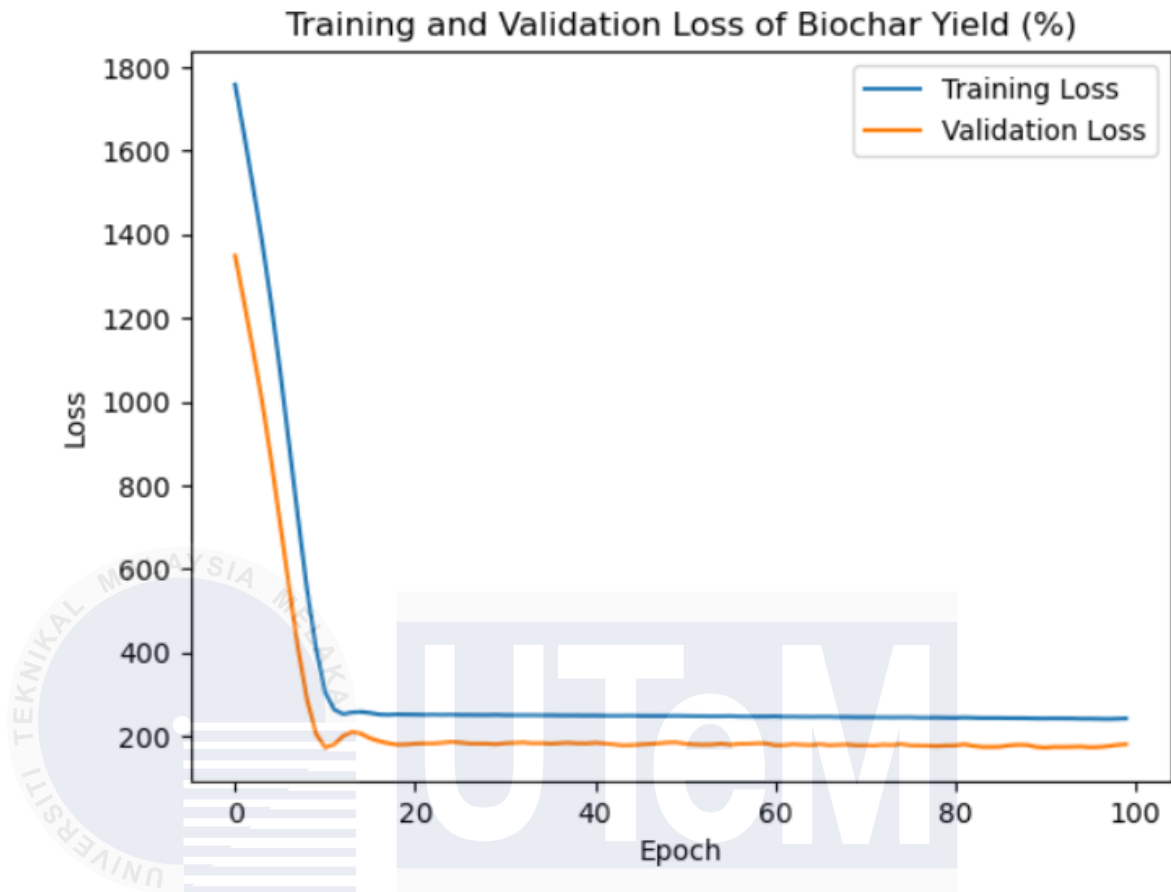


Figure 5.5: Training and Validation Loss of Biochar Yield using Logarithmic Transformation

In the case of the Convolutional Neural Networks (CNN) model, on using Standard Scaler, the average RMSE comes out to be approximately 6.9441. But, in case of a logarithmic transformation, the average RMSE value increases dramatically to approximately 12.9289. During training, both the training and validation losses decrease with the epochs, which means that the model is learning well. However, the farther the training progresses, a clear discrepancy is drawn between the two losses: whereas the training loss keeps decreasing, the validation loss starts increasing after crossing a point of threshold. It is the classical pointer toward overfitting, in which a model becomes overly fitted to the training dataset and hence loses much from the ability to acquire generalizability on new, unseen data.

Table 9: Comparison of Average RMSE Values between ‘Standard Scaler’ and ‘Logarithmic Transformation’

Model	Standard Scaler	Logarithmic Transformation
Linear Regression	10.2586	8.9034
Random Forest Regressor	6.3511	6.3830
K-Nearest Neighbor	8.7175	9.0111
Convolutional Neural Network	6.9441	12.9289

The accurate prediction of the yield in biochar production will aid in the optimization of the pyrolysis process involved in converting organic waste to biochar. It is also to be observed that the Random Forest model emerges as very reliable in making accurate predictions since it would return the lowest RMSE at 6.3511 with the settings of Standard Scaler. It must also be sustained with maximum efficiency in biochar production so that the operators realize the influential factors and have optimum decisions pertaining to process parameters such as temperature, feedstock type, and residence time. It should be a layer that retains all the variability of the data, not going through too much complicated preprocessing. Considering model performance with a Standard Scaler, this proves the model is going to work fine with different datasets and different preprocessed data. This will make it very deployable in different production environments and able to adapt to different environmental conditions and different feedstock types. On the other hand, both linear regression and KNN models were less accurate and less robust respectively. These limitations reduce their production application mainly where the biochar yield needs to be accurate. While CNN may capture such complex patterns, it is not stable with some data transformations hence not reliable for use in the prediction of biochar yield. Among the developed models, the Standard Scaler-built Random Forest model was the best to

predict the yield of biochar and gave results on specificity and reliability. This result would thus be useful during the optimization process of pyrolysis, yield prediction, and improving the economic and environmental performance of food waste-derived biochar.

5.3 Analysis and Discussion

5.3.1 Linear Regression

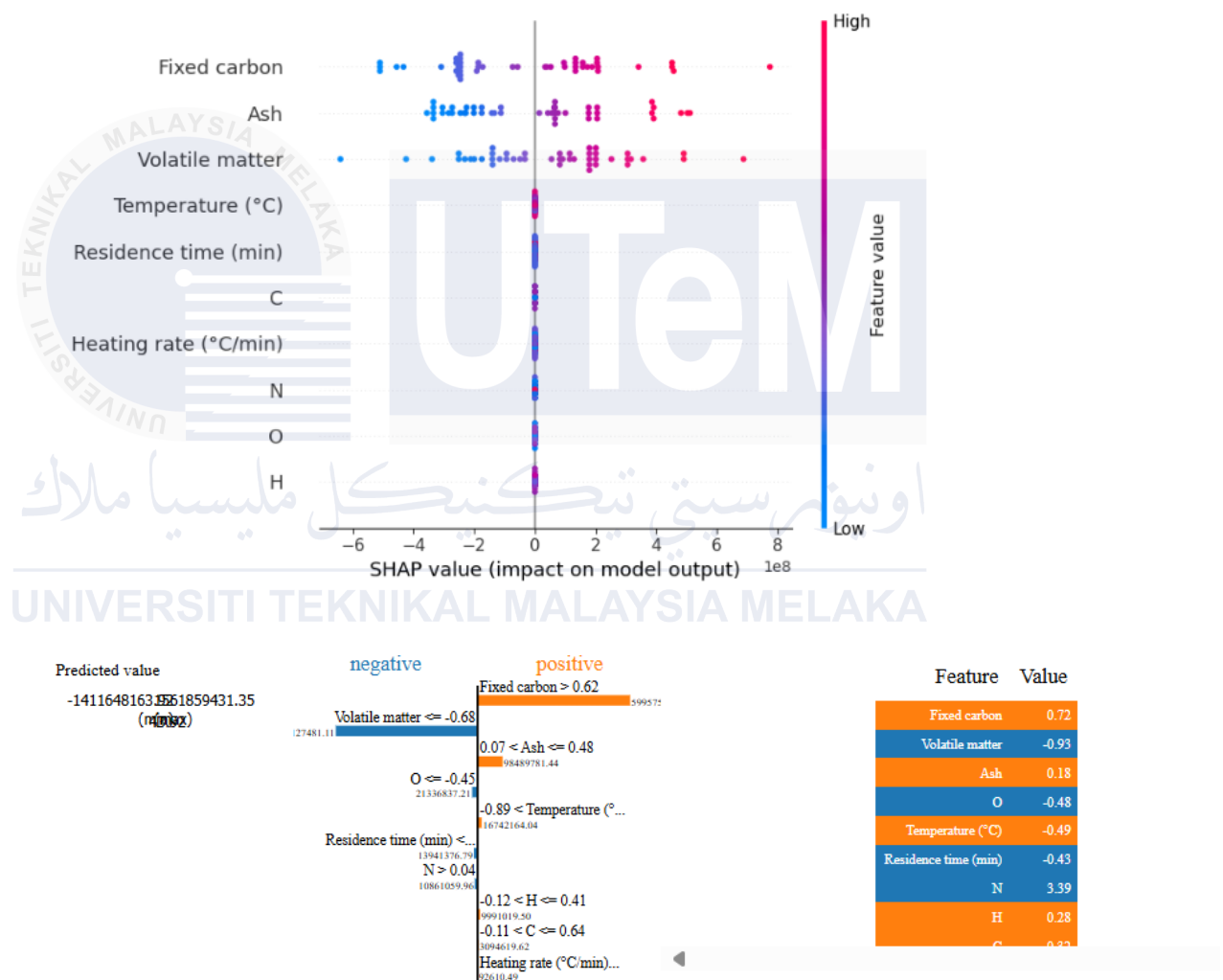


Figure 5.6: Results of Feature Analysis Using Standard Scaler with SHAP and LIME

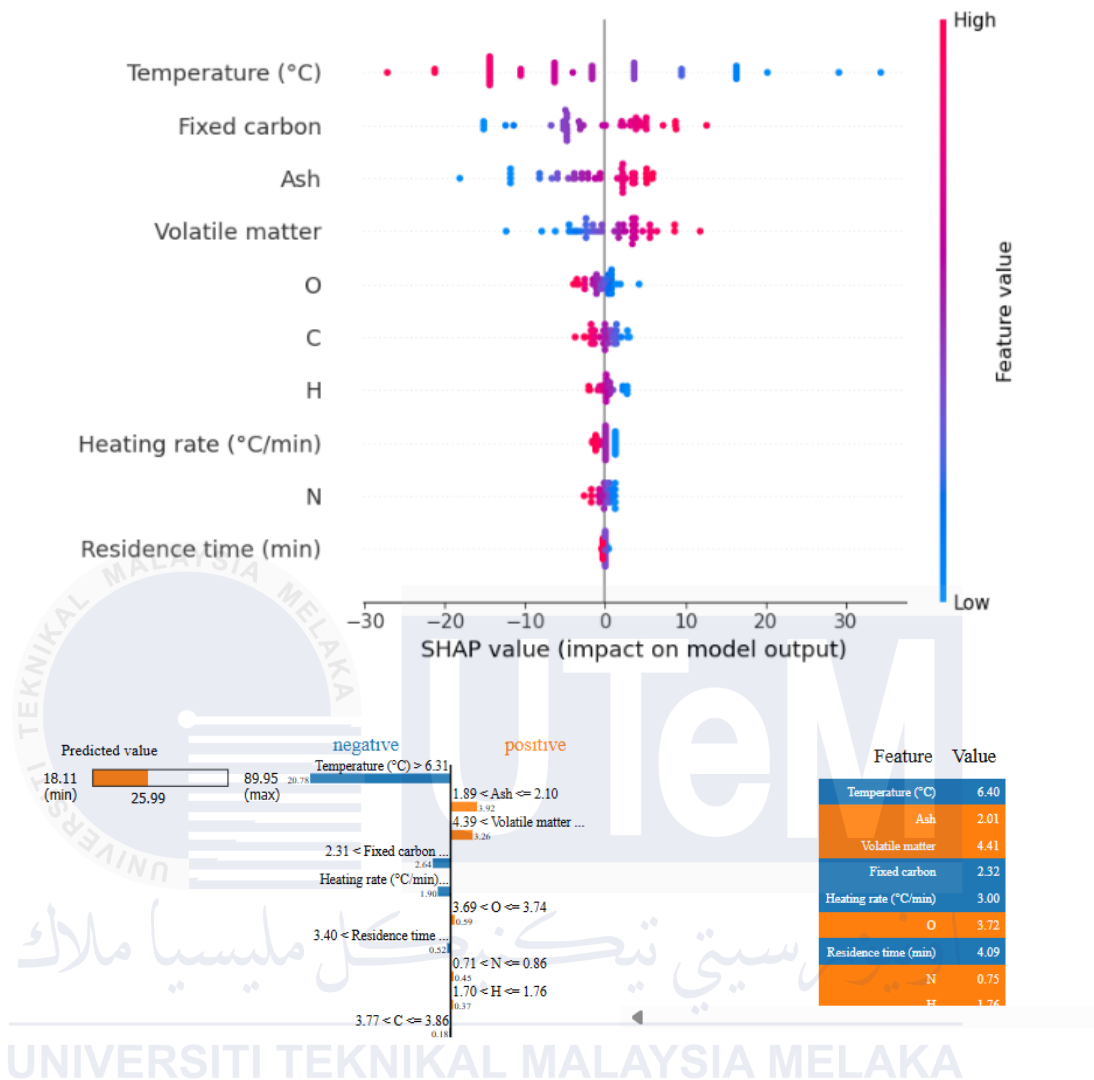


Figure 5.7: Results of Feature Analysis Using Logarithmic Transformation with SHAP and LIME

Now, in the case of linear regression, Standard Scaler, SHAP and LIME, both say that the three most important features are Fixed Carbon, Ash, and then Volatile Matter. Fixed Carbon and Ash have a direct correlation with biochar yield for Fixed Carbon greater 0.62 and for Ash between 0.07 and 0.48. Positive values mean that, in the prediction of biochar yield, this characteristic has the highest value. Volatile Matter also shows a negative correlation but not with absolute values higher than -0.69 . The higher the value of the Volatile Matter, the lower its contribution to biochar yield prediction. After the logarithmic transformation, with effects of Fixed Carbon, Ash, and Volatile Matter, Temperature was the most influential variable. In fact, Temperature and Fixed Carbon are strongly negatively correlated because values of Temperature greater than -6.31 and Fixed Carbon values above -2.31 indicate that,

although the Temperature values increase, this variable turns out to be less relevant to predict biochar yield and for Fixed Carbon, the same occurs. The Ash content varies from 1.89 to 2.10 and is positively dependent, showing an increase contribution toward the yield prediction. Whereas volatile matter is more than 4.39 and also has a positive relation, it goes forward and increases the predicted yield of 25.99, based on the LIME analysis.

5.3.2 Random Forest Regression

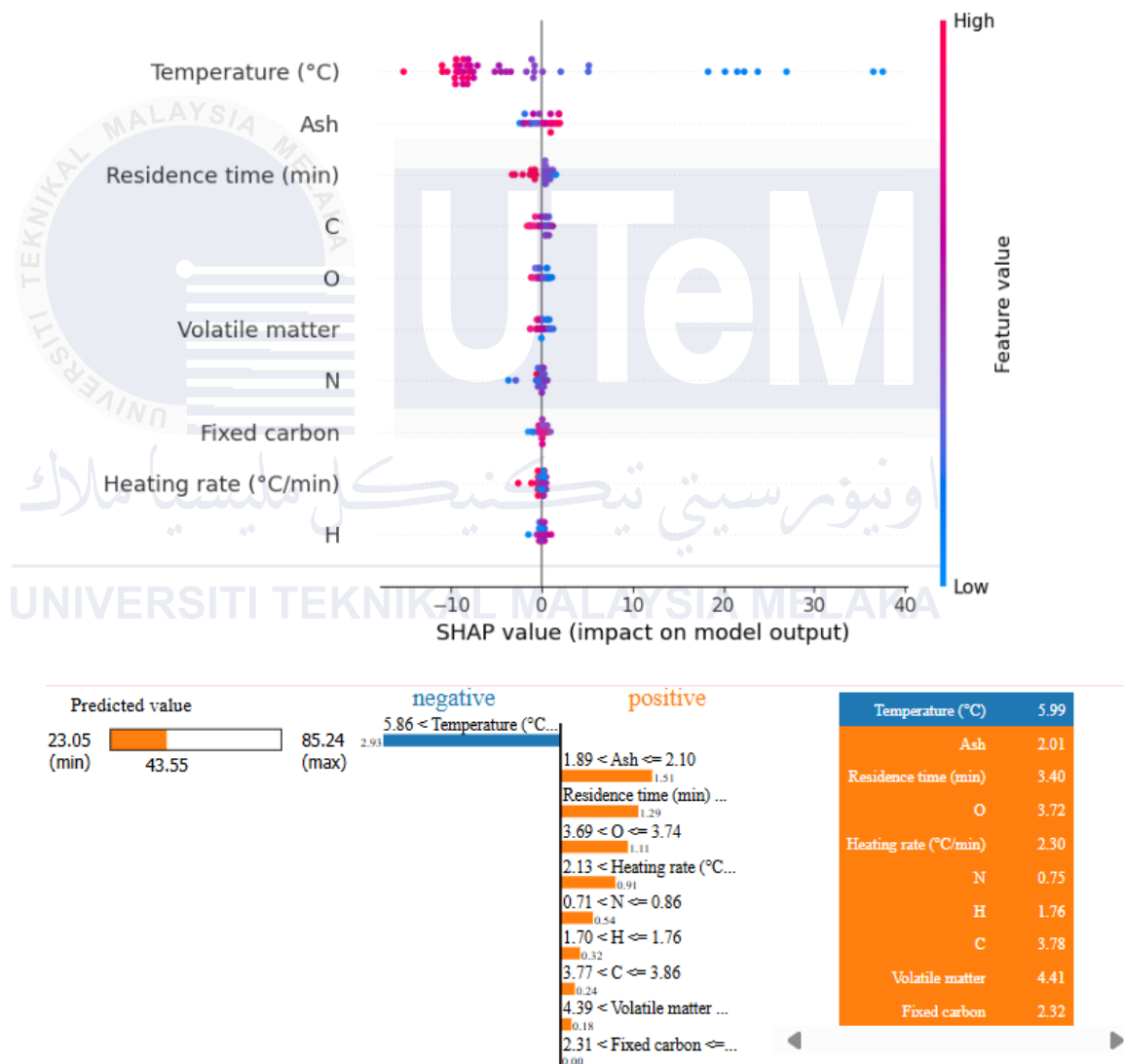


Figure 5.8: Results of Feature Analysis Using Standard Scaler with SHAP and LIME

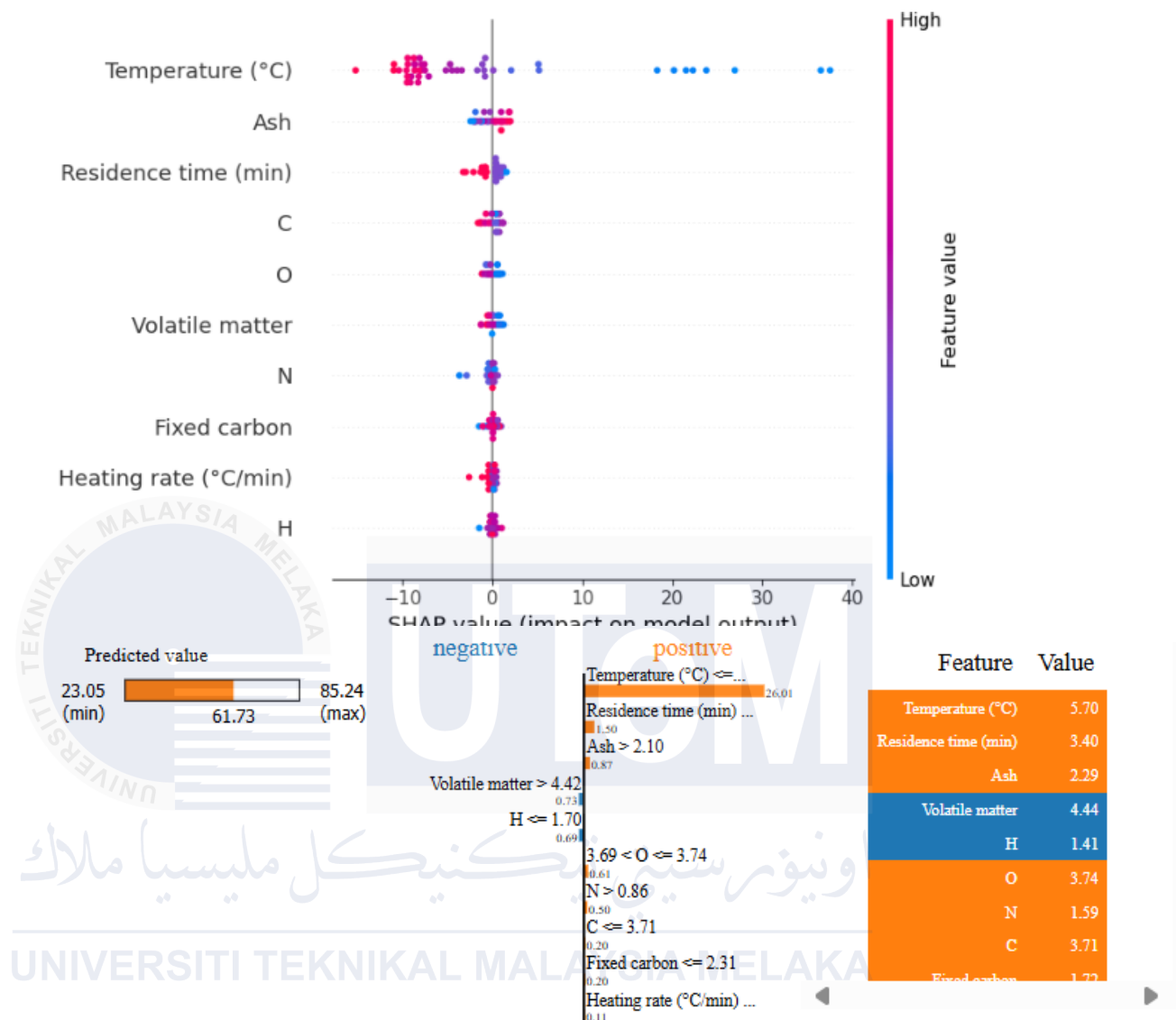


Figure 5.9: Results of Feature Analysis Using Logarithmic Transformation with SHAP and LIME

Here Temperature tops the list of important features of the Random Forest model while applying Standard Scaler in SHAP, followed by Ash and Residence Time (min). Elaborating on this through the LIME methodology, one finds Temperature to be highly negatively correlated with Biochar yield, quantified at more than -5.86. From this, one can deduce that an increase in Temperature will negatively affect the prediction. Ash and Residence Time, on the other hand, are positively correlated with the yield of the biochar; they range from 1.89 to 2.10 and from 3.69 to 3.74, respectively. This will mean that an increase in Ash and Residence Time will directly double the expected yield to a value of 43.55. This turns Temperature to become the most important feature followed by Ash and Residence Time. Also, under logarithmic

transformation, the LIME values indicate the positive nature of the relationship that exists between these features: Temperature, Residence Time, and Ash, in respect to biochar yield prediction, where their higher values give a higher contribution to the prediction of biochar yield at 61.73.

5.3.3 K-Nearest Neighbor (KNN)

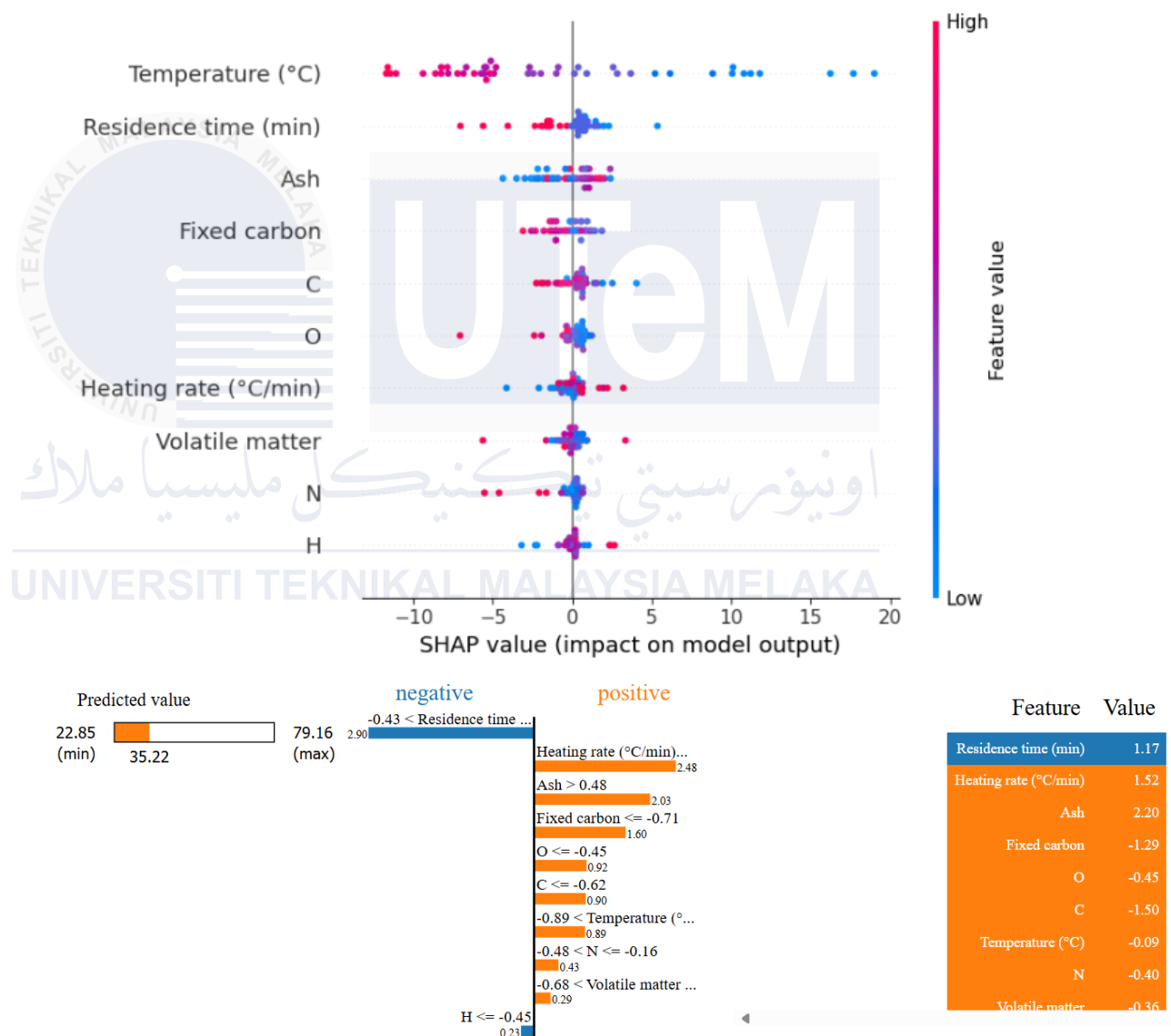


Figure 5.10: Results of Feature Analysis Using Standard Scaler with SHAP and LIME

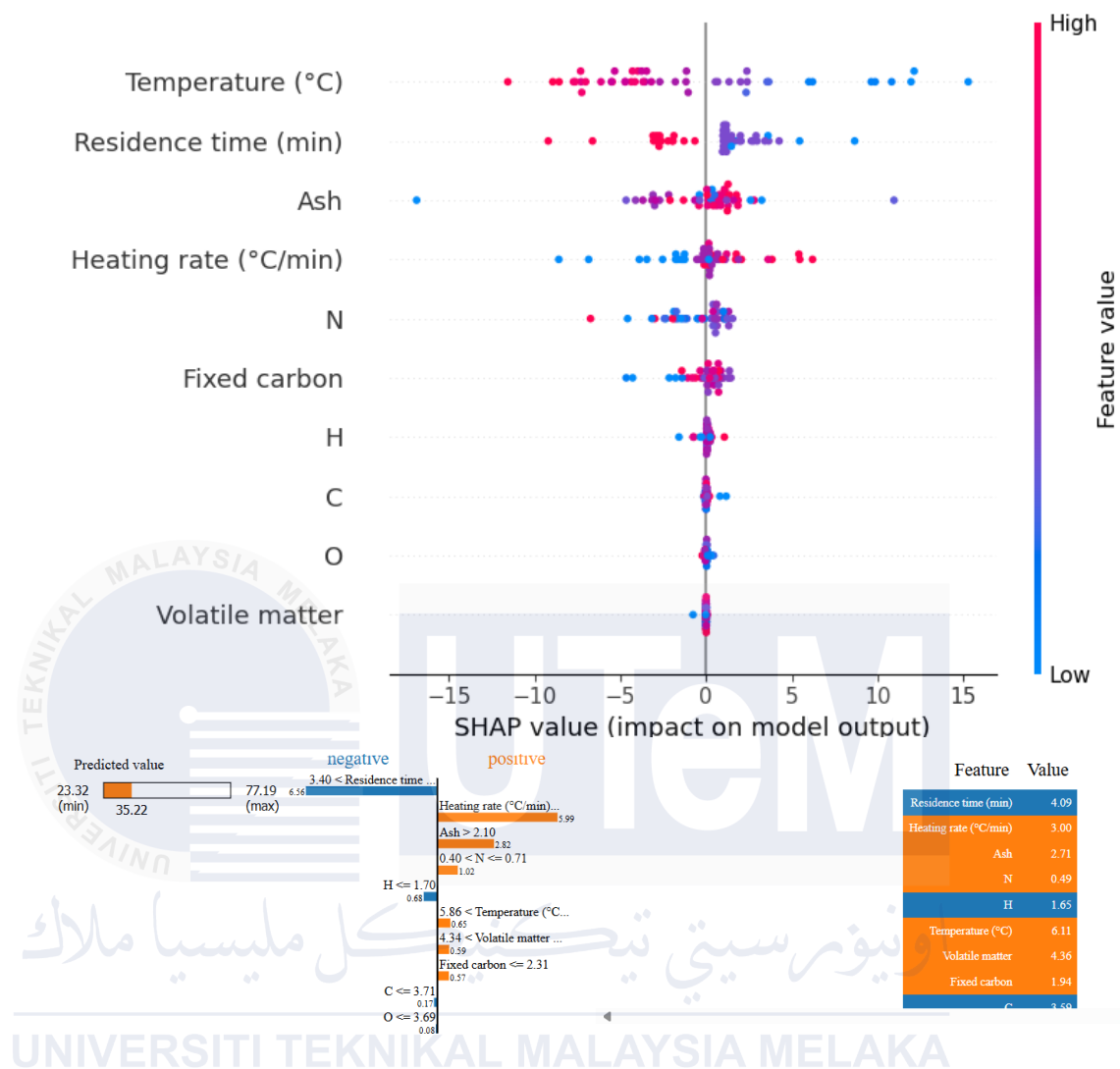


Figure 5.11: Results of Feature Analysis Using Logarithmic Transformation with SHAP and LIME

Some of the most prominent features of the KNN model at SHAP with the Standard Scaler are Temperature, Residence Time (min), and Ash. However, none of these models was ever robust as some others. It follows from LIME that the Residence Time (min) has negative correlation in Biochar Yield when the values become less than -0.43. That is, with the increment of Residence Time, the contribution in the prediction for the Biochar Yield decreases. On the other hand, the Heating Rate (°C/min) and Ash predictor variables positively relate to the yield of Biochar, with the correlation greater than 0.48, increase in Heating Rate (°C/min) and Ash would generate enhancement predictability of biochar yield up to an estimated yield of 35.22. The right results from having used the logarithmic transformation are similar to what was obtained before. The most influencing factor is Temperature, followed by

Residence Time and Ash. It means that Ash content and Heating Rate °C/min are positively correlated with biochar yield, showing values greater than 2.10 in their relevance toward the increase of these parameters for better prediction. Nevertheless, Residence Time in minutes has a negative correlation, and where the value is greater than 3.40, thus increasing Residence Time demonstrates less impact in the prediction of biochar yield. That product of the yield gives an estimate of 39.98.

5.3.4 Convolutional Neural Network (CNN)

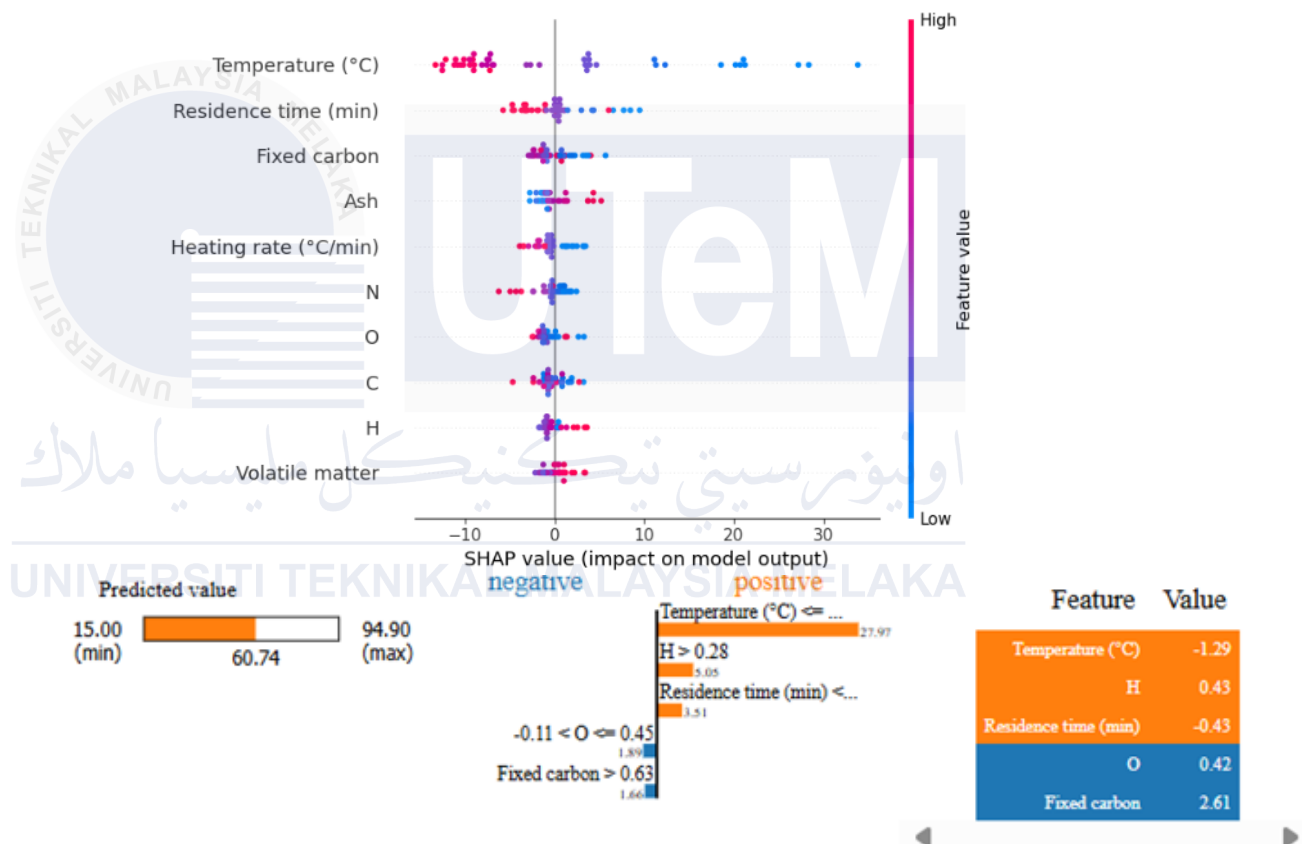


Figure 5.12: Results of Feature Analysis Using Standard Scaler with SHAP and LIME

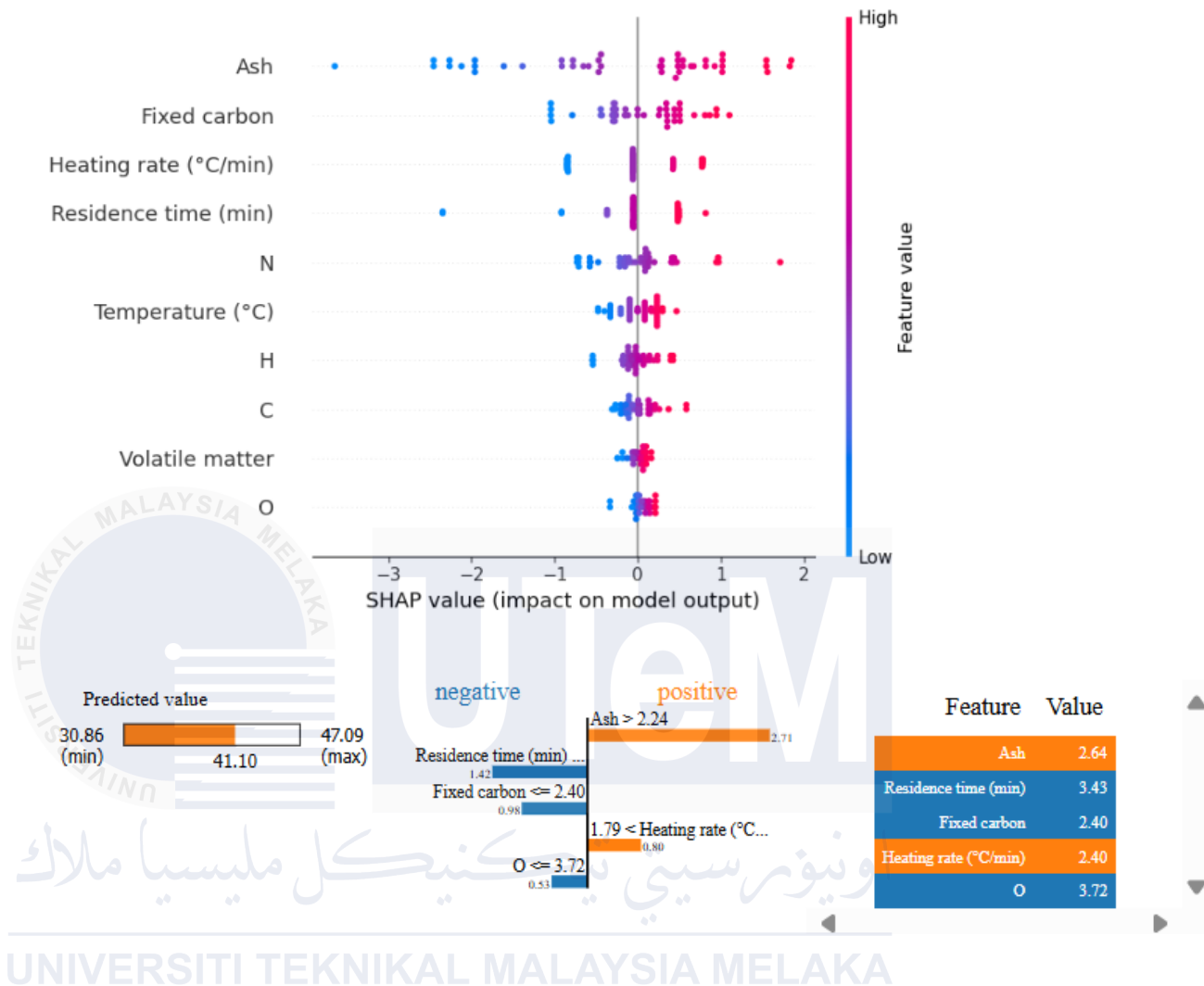


Figure 5.13: Results of Feature Analysis Using Logarithmic Transformation with SHAP and LIME

Using Standard Scaler in SHAP, the features in the CNN model relate to Temperature as the most associated feature, followed by Residence Time, and Fixed Carbon. In LIME, both Temperature and Residence Time will positively correlate with biochar yield and hence meaning the improvement of these features contributing positive for the prediction of biochar yield. Moreover, Hydrogen was also positively correlated; a value above 0.28 means that when going above, it increases the contribution towards the prediction of the biochar yield to 60.74 predicted yield. From the SHAP results, the most important variables after applying the logarithmic transform are Ash, then Fixed Carbon, and lastly, Heating Rate. The observations from the LIME analysis are that the biochar yield increases with both the Ash content and Heating Rate for the values of the two variables, which are greater than 2.24 and 1.79, respectively. The positive correlation in contributions indicates an improvement in

prediction on increasing the two variables of biochar yield. Fixed Carbon and Residence Time (min), on the other hand, will only be a maximum of -2.40, hence, evidencing that when Fixed Carbon and Residence Time (min) increases, its predictive contribution to biochar yield decreases. This leads to the predicted output of 41.10.

Table 10: Comparison of Top 3 Best Key Features using ‘Standard Scaler’ ‘Logarithmic Transformation’ for All Models between SHAP and LIME

Model	Standard Scaler		Logarithmic Transformation	
	SHAP	LIME	SHAP	LIME
Linear Regression	Fixed Carbon, Ash, Volatile Matter	Fixed Carbon, Volatile Matter, Ash	Temperature, Fixed Carbon, Ash	Temperature, Ash, Volatile Matter
Random Forest Regressor	Temperature, Ash, Residence time (min)	Temperature, Ash, Residence time (min)	Temperature, Ash, Residence time (min)	Temperature, Residence time (min), Ash
K-Nearest Neighbor	Temperature, Residence time (min), Ash	Residence time (min), Heating Rate (°C/min), Ash	Temperature, Residence time (min), Ash	Residence time (min), Heating Rate (°C/min), Ash
Convolutional Neural Network	Temperature, Residence time (min), Fixed Carbon	Temperature, Hydrogen, Residence time (min),	Ash, Fixed Carbon, Heating Rate (°C/min)	Ash, Residence time (min), Fixed Carbon

Table 11: Probability Distribution of Top 3 Key Features from the Comparison table of SHAP and LIME

Features	Probability
Temperature	10/16
Fixed Carbon	6/16
Volatile Matter	3/16
Residence time (min)	11/16
Ash	14/16
Heating Rate (°C/min)	3/16
Hydrogen	1/16

Indeed, using the Standard Scaler and log transformation for the modeling analysis, Temperature, Residence Time (min), and Ash remain always among the three most important features to predict the biochar yield. Among these, temperature generally comes out as the most critical predictor across all models and scaling methods considered. Particularly, when using logarithmic transformation, there are quite strong positive and negative correlations with Biochar Yield that involve Temperature. Controlling the Temperature during pyrolysis is practical for a given application, whereby some precision in regulating Temperature can be done to optimize the Biochar Yield. Efficient Temperature management maintains this within an optimum range, depending on feedstock composition and desired properties of biochar produced. Another important parameter affecting the biochar yield prediction is Residence Time (min). On analyzing, as explained in the Temperature study, Residence Time (min) has again played a vital influential role. In actual practice, the Residence Time (min) is to be optimized since it is the parameter that will specify to what extent the biomass is to be exposed for a higher Temperature, which will affect the yield and quality of the biochar produced. The tuning of Residence Time with Temperature enables the attainment of the preferential Biochar characteristics. The Ash content represents one of the most important features among the different models and scaling methods. Generally, it impacts Biochar Yield positively, especially Random Forest models. Such Ash content would most likely mean that a balanced

quantity of Ash could favorably affect the yield and probably work as a catalyst during pyrolysis. Feedstock selection or blending of several biomasses may help control such ash content toward a positive enhancement of the pyrolysis process. Amongst the various scaling methods, in this case, logarithmic transformation is best. It gives more importance to Temperature, which is a very important variable. Skewness in the data distribution is also treated properly. This would enable one to establish the relationship of variables much more precisely, particularly in dealing with different scales of values. Exact control in optimization of Temperature and Residence Time for maximum biochar yield must be quality-based in the feedstock. Optimal feedstocks have a high carbon content and well-balanced mineral composition, while pretreatment methods also offer better control on the product. One of the future developments possible for biochar yield estimation is online temperature monitoring and heating rate, with optimal values maintained by active control. Such production of biochar would be considerably enhanced in respect to the mentioned factors, making the process of pyrolysis much more efficient and in line with the principles of sustainability when it comes to waste management and energy production.

5.4 Summary

In this chapter, various models have been analyzed for the prediction of biochar yield in food waste pyrolysis, all of which the **Random Forest** model came out on top with an average value of RMSE ranging from **6.3511** to **6.3830**, proving highly accurate and reliable for the study. Linear Regression did improve but still could not perform as well as the Random Forest model. While the predictive accuracy of K-Nearest Neighbors and Convolutional Neural Networks showed a wide range between transformations applied, their results were far less consistent.

SHAP and LIME methodologies for feature analysis gave, as the most important drivers with respect to the model prediction of biochar yield, the following features: **Temperature, Residence Time (min), and Ash**. Temperature was the most important predictor in all models, pointing out the strong impact on the biochar yield. Other important features were the Residence Time (min) and Ash content, whose influence on the yield is different in accordance with the selected model.

In summary, this **Random Forest** model outperformed all others in predicting biochar yield, bringing out very high accuracy and better identification of principal factors affecting yield. This model should thus be used for the optimization of the pyrolysis process in order to strengthen both economic and environmental benefits related to biochar production.



CHAPTER 6: CONCLUSION

6.1 Introduction

In conclusion, the project is well rounded and detailed, with the key findings listed after which are discussions on contributions and limitations and proposals on possible areas of future research. In summary, this was a project aimed at solving the detailed challenge of Biochar yield prediction from food waste pyrolysis using cutting edge explainable Artificial Intelligence. By so doing, this study has enabled the furthering of a problem like biochar yield prediction and how it affects sustainable waste management.

6.2 Project summarization

SHapley Additive Explanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) were identified for biochar yield to be predicted to further explore the influential patterns of different features on the ML models. As developed and pre-trained in the current study, Convolutional Neural Network models had explicability towards the model results added through SHAP and LIME techniques. The result of this study pinpoints that some highly affecting factors on the biochar yield prediction are in unison with a CNN model, SHAP, and LIME provide a good framework to interpret those. Further results depict how the inclusion of explainable AI is going to enhance transparency and good conduction of the ML model in such complex predictive tasks as biochar yield.

The development of a predictive model for biochar yield has been associated from the pyrolysis of food waste: it is deeply embedded in Explainable AI. We used techniques ranging from machine learning and deep learning models like Linear

Regression, Random Forest Regressor, K-Nearest Neighbors, Convolutional Neural Networks and also SHapley Additive exPlanations, Local Interpretable Model-agnostic Explanations for improving model interpretability. Main findings highlighted that **Random Forest** is coming out significantly, and RMSE is at ranging from **6.3511** to **6.3830**, giving precise predictions. Moreover, SHAP and LIME reveal the leading features to be **Temperature, Residence Time (min), and Ash**. Project results reveal that, with the integration of XAI techniques, there will be an improvement in transparency and reliability of ML models to predict biochar yield, which is highly relevant to process optimization in pyrolysis further to sustainable waste management.

6.3 Project Contribution

The outcomes of this project has managed to churn are highly significant if considering the fact that, for the first time, it has showed just how advanced in terms of efficiency machine learning methodologies using Random Forest can be in order to predict biochar yield accurately, hence providing a new alternative of valid use other than traditional models. Research since then has refocused on understanding how tools, such as SHAP and LIME, in XAI operate, to be useful in interpreting predictions of the models and understanding the importance of the features used within the model.

Therefore, it forms a firm foundation for a more interpretable and clearer predictive model relevant in the determination of decision-making for waste-to-resource technologies. That is to say, the worth of this research includes providing the much-needed critical insights into the major factors that affect biochar yield. Consequently, it sets a high bar for later research efforts aimed at integrating the most advanced neural network models with tools for explainability.

6.4 Project Limitation

While valuable, the project has its limitations; the most important ones are those of computational complexity and resource intensity since performing training of the CNN model is infeasible for many researchers or applications. This, of course, also comes with a few caveats: it will be computationally more expensive to run both SHAP and LIME and further can lead to an overreliance on such techniques without much understanding of the underlying assumptions and restrictions. These have further implications that require careful consideration of the inferences; more work should be

done to overcome these issues, probably with the development of better effective modeling strategies and interpretative methods.

6.5 Future Works

Based on the findings and their limitations, several possible ways of future work will be outlined. This will pave the way to further optimize CNN models such that, computationally, the demand gets reduced in the future for widespread use. Further enhancement of model interpretability with CNNs using SHAP and LIME will, therefore, allow models to unravel more complexity and, hence, get used in wider domains. This will open up the potential for future research in terms of the combination of more innovative methods of explainable AI with even more innovative approaches toward model interpretability techniques. In addition, the determination becomes much more effective and efficient regarding the yield of biochar with the online monitoring of key process parameters, such as temperature and heating rate during pyrolysis.

6.6 Summary

This chapter finally synthesizes the main findings of this project and their contributions, acknowledges its limitations, and opens avenues for future work. Value is on knowledge gained in machine learning and waste domains, particularly towards making significant contributions to interpretability and transparency in predictive modeling. In this direction, the best prediction of yield in Biochar was achieved by the **Random Forest** model with including **Temperature, Residence Time (min), and Ash** in the process of pyrolysis optimization. Thus, with the present study, it is expected to further invigorate future research in this direction, and recommendations provided in this field are optimized for more effective and sustainable waste management strategies achieved in the process of pyrolysis.

REFERENCES

- Ali, S., Abuhmed, T., El-Sappagh, S., Muhammad, K., Alonso-Moral, J. M., Confalonieri, R., & Herrera, F. 2023. Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence. *Information Fusion*, 99, 101805.
- Brennan, L., Langley, S., Verghese, K., Lockrey, S., Ryder, M., Francis, C., et al., 2021. The role of packaging in fighting food waste: a systematised review of consumer perceptions of packaging. *J. Clean. Prod.* 281, 125276.
- Chen, M. W., Chang, M. S., Mao, Y., Hu, S., & Kung, C. C. (2023). Machine learning in the evaluation and prediction models of biochar application: A review. *Science Progress*, 106(1), 00368504221148842.
- Hai A, Bharath G, Daud M, Rambabu K, Ali I, Hasan SW, et al. 2021. Valorization of groundnut shell via pyrolysis: Product distribution, thermodynamic analysis, kinetic estimation, and artificial neural network modeling. *Chemosphere* 2021; 283:131162.
- Hai, A., Bharath, G., Patah, M. F. A., Daud, W. M. A. W., Rambabu, K., Show, P., & Banat, F. 2023. Machine learning models for the prediction of total yield and specific surface area of biochar derived from agricultural biomass by pyrolysis. *Environmental Technology & Innovation*, 30, 103071.
- Istrate, I.-R., Iribarren, D., Gálvez-Martos, J.-L., Dufour, J., 2020. Review of life-cycle environmental consequences of waste-to-energy solutions on the municipal solid waste management system. *Resour. Conserv. Recycl.* 157, 104778.
- Kaczor, Z., Bulin'ski, Z., Werle, S., 2020. Modelling approaches to waste biomass pyrolysis: a review. *Renewable Energy* 159, 427–443.

- Khan, M., Ullah, Z., Mašek, O., Naqvi, S. R., & Khan, M. N. A. 2022. Artificial neural networks for the prediction of biochar yield: a comparative study of metaheuristic algorithms. *Bioresource Technology*, 355, 127215.
- Lee, X.J., Ong, H.C., Gan, Y.Y., Chen, W.-H., Mahlia, T.M.I., 2020. State of art review on conventional and advanced pyrolysis of macroalgae and microalgae for biochar, bio-oil and bio-syngas production. *Energy Convers. Manag.* 210, 112707.
- Li, Y., Gupta, R., & You, S. 2022. Machine learning assisted prediction of biochar yield and composition via pyrolysis of biomass. *Bioresource Technology*, 359, 127511.
- Pathy, A., Meher, S., P, B., 2020. Predicting algal biochar yield using extreme gradient boosting (XGB) algorithm of machine learning methods. *Algal Research*. 50, 102006.
- Salih, A., Raisi-Estabragh, Z., Galazzo, I. B., Radeva, P., Petersen, S. E., Menegaz, G., & Lekadir, K. 2023. Commentary on explainable artificial intelligence methods: SHAP and LIME. *arXiv preprint arXiv:2305.02012*.
- Tee J.X, Selvarajoo A, Arumugasamy S.K. 2022. Prediction of carbon sequestration of biochar produced from biomass pyrolysis by artificial neural network. *J Environ Chem Eng*;10(3):107640.
- Wang, L., Ok, Y.S., Tsang, D.C.W., Alessi, D.S., Rinklebe, J., Wang, H., et al., 2020. New trends in biochar pyrolysis and modification strategies: feedstock, pyrolysis conditions, sustainability concerns and implications for soil amendment. *Soil Use Manag.* 36, 358–386.
- Wang, Z., Peng, X., Xia, A., Shah, A.A., Huang, Y., Zhu, X., Zhu, X., Liao, Q., 2022. The role of machine learning to boost the bioenergy and biofuels conversion. *Bioresour. Technol.* 343, 126099.

Zamri, M.F.M.A., Bahru, R., Suja, F., Shamsuddin, A.H., Pramanik, S.K., Fattah, I.M.R., 2021. Treatment strategies for enhancing the removal of endocrine-disrupting chemicals in water and wastewater systems. *J. Water Process Eng.* 41, 102017.

Zhang, Y., Cui, Y., Liu, S., Fan, L., Zhou, N., et al., 2020. Fast microwave-assisted pyrolysis of wastes for biofuels production - a review. *Bioresour. Technol.* 297, 122480.

FAO. (2011). *Global food losses and food waste – Extent, causes and prevention.* Rome.

Lehmann, J., & Joseph, S. (Eds.). (2009). *Biochar for Environmental Management: Science and Technology.* Earthscan.

Kabir, M. M., Yacout, S., & Le, D.-T. (2015). Machine learning techniques for environmental control: A comprehensive review. *Environmental Science and Pollution Research*, 22(15), 11220-11236.

Al-Gheethi, A., Mohamed, R. M. S. R., Efaq, A. N., & Lalung, J. (2018). Prediction of energy demand in Malaysia using machine learning techniques. *Renewable and Sustainable Energy Reviews*, 82, 1415-1428.

Gunning, D., & Aha, D. (2019). DARPA's Explainable Artificial Intelligence (XAI) Program. *AI Magazine*, 40(2), 44-58.

Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5-32.

Caruana, R., Gehrke, J., Koch, P., Nir, M., & Thursby, J. (2015). Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission.

Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1721-1730.

Cover, T. M., & Hart, P. E. (1967). *Nearest-neighbor pattern classification*. IEEE Transactions on Information Theory, 13(1), 21-27.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Johnson, R. A., & Wichern, D. W. (2007). *Applied multivariate statistical analysis*. Pearson.

Khatri, V., Kumar, P., & Jayaraman, V. (2021). Smart sensing for precision agriculture: A review. *Agricultural Systems*, 191, 103177.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436-444.

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 4765-4774.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135-1144.

Zhao, J., Zhang, X., & Wang, L. (2019). Internet of Things and smart agriculture: An overview. *Journal of Computer Networks and Communications*, 2019, 1-12.

K. T. Klasson, "Biochar characterization and a method for estimating biochar quality from proximate analysis results," *Biomass and Bioenergy*, vol. 96, pp. 50–58, Jan. 2017.

APPENDIX A

Fixed carbon	Volatile matter	Ash	C	H	O	N	Residence time (min)	Temperature (°C)	heating rate (°C/min)	Biochar yield (%)
18.01	78.71	3.28	48.12	6.48	43.51	1.89	30.00	400.00	15.00	30.80
18.01	78.71	3.28	48.12	6.48	43.51	1.89	60.00	400.00	10.00	26.60
18.01	78.71	3.28	48.12	6.48	43.51	1.89	90.00	400.00	5.00	26.77
18.01	78.71	3.28	48.12	6.48	43.51	1.89	30.00	500.00	15.00	23.57
18.01	78.71	3.28	48.12	6.48	43.51	1.89	60.00	500.00	10.00	25.32
18.01	78.71	3.28	48.12	6.48	43.51	1.89	90.00	500.00	5.00	25.40
18.01	78.71	3.28	48.12	6.48	43.51	1.89	30.00	600.00	15.00	22.37
18.01	78.71	3.28	48.12	6.48	43.51	1.89	60.00	600.00	10.00	24.90
18.01	78.71	3.28	48.12	6.48	43.51	1.89	90.00	600.00	5.00	24.67
18.01	78.71	3.28	48.12	6.48	43.51	1.89	60.00	300.00	15.00	77.30
18.01	78.71	3.28	48.12	6.48	43.51	1.89	60.00	400.00	15.00	36.90
18.01	78.71	3.28	48.12	6.48	43.51	1.89	60.00	500.00	15.00	23.30
18.01	78.71	3.28	48.12	6.48	43.51	1.89	60.00	600.00	15.00	21.70
6.54	91.16	2.30	42.10	5.90	43.51	0.50	60.00	300.00	20.00	33.00
6.54	91.16	2.30	42.10	5.90	43.51	0.50	60.00	350.00	20.00	25.00
6.54	91.16	2.30	42.10	5.90	43.51	0.50	60.00	400.00	20.00	24.30
6.54	91.16	2.30	42.10	5.90	43.51	0.50	60.00	450.00	20.00	24.00
11.19	87.76	1.05	44.47	5.82	48.88	0.00	60.00	400.00	5.00	34.21
11.19	87.76	1.05	44.47	5.82	48.88	0.00	60.00	450.00	5.00	31.89
11.19	87.76	1.05	44.47	5.82	48.88	0.00	60.00	500.00	5.00	29.06
11.19	87.76	1.05	44.47	5.82	48.88	0.00	60.00	550.00	5.00	27.88
11.19	87.76	1.05	44.47	5.82	48.88	0.00	60.00	600.00	5.00	26.12
15.38	78.50	6.12	41.00	7.17	33.24	0.99	30.00	300.00	10.00	62.89
15.38	78.50	6.12	41.00	7.17	33.24	0.99	30.00	400.00	10.00	41.58
15.38	78.50	6.12	41.00	7.17	33.24	0.99	30.00	500.00	10.00	34.79
15.38	78.50	6.12	41.00	7.17	33.24	0.99	30.00	600.00	10.00	36.85
15.38	78.50	6.12	41.00	7.17	33.24	0.99	30.00	700.00	10.00	32.62

اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

APPENDIX B

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sb
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error
import random
import os
import shap
import lime
import lime.lime_tabular
|
# Load the CSV data file into a pandas DataFrame
file_path = r'C:\Users\keswa\OneDrive - Universiti Teknikal Malaysia Melaka\UTEM\BITU 3973
try:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='latin1')
except UnicodeDecodeError:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='iso-8859-1')

fig = px.histogram(
    df,
    x='Biochar yield (%)',
    histfunc='count',
    template='plotly_dark',
    title='Distribution of Biochar Yield',
    labels={'Biochar yield (%)': 'Biochar Yield (%)'}, # Renaming x-axis label
)

fig.update_layout(
    xaxis_title='Biochar Yield (%)',
    yaxis_title='Frequency'
)

```

```

fig.show()

#Visualisation of all the variables (wNTP)
fig = px.box(df, y=df.drop(['Biochar yield (%)'], axis=1).columns, template='plotly_dark',
fig.show()

# Load the CSV data file into a pandas DataFrame
file_path = r'C:\Users\keswa\OneDrive - Universiti Teknikal Malaysia Melaka\UTEM\BITU 3973
try:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='latin1')
except UnicodeDecodeError:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='iso-8859-1')

print(df.head())

# Select the features and target variable
features = ['Fixed carbon', 'Volatile matter', 'Ash', 'Temperature (°C)', 'C', 'H', 'O',
target = 'Biochar yield (%)'

X = df[features]
y = df[target]

# Apply standard scaling to the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Convert scaled features back to DataFrame for easier manipulation
X_scaled = pd.DataFrame(X_scaled, columns=features)

# Print the scaled features
print("\nStandard-scaled features:")
print(X_scaled.head())

# Split the dataset into training and testing subsets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_sta

# Linear Regression model for Biochar Yield
# Define the function to train and evaluate the linear regression model
def train_evaluate_lr(X_train, X_test, y_train, y_test):
    regressor = LinearRegression()
    regressor.fit(X_train, y_train)
    y_pred = regressor.predict(X_test)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    return rmse, y_test, y_pred, regressor

# Initialize lists to store results for visualization
rmse_list = []
selected_features_list = []

# Initialize an empty list to store selected features
selected_features = []
best_rmse = float('inf') # Initialize with a high value
best_features = []

# Define the maximum number of features to consider
max_features = len(X.columns)

# Loop over the maximum number of features to select
for i in range(max_features):
    remaining_features = list(set(features) - set(selected_features))
    if remaining_features:
        # Select a random feature from the remaining features
        random_feature = random.choice(remaining_features)
        selected_features.append(random_feature)

        # Train and evaluate linear regression model with the selected features
        current_rmse, _, _, _ = train_evaluate_lr(X_train[selected_features], X_test[selected_features], y_train, y_test)

        # Store the RMSE and the selected features
        rmse_list.append(current_rmse)
        selected_features_list.append(list(selected_features))

```

```

# Update the best RMSE and best features if the current RMSE is better
if current_rmse < best_rmse:
    best_rmse = current_rmse
    best_features = list(selected_features)

# Save the best RMSE value to a file
rmse_file = 'best_rmse_values_regression.txt'
if os.path.exists(rmse_file):
    with open(rmse_file, 'a') as file:
        file.write(f'{best_rmse}\n')
else:
    with open(rmse_file, 'w') as file:
        file.write(f'{best_rmse}\n')

# Read the RMSE values from the file and compute the average
with open(rmse_file, 'r') as file:
    rmse_values = [float(line.strip()) for line in file.readlines()]

average_rmse = np.mean(rmse_values)

# Create a DataFrame to display the selected features at each step
feature_selection_df = pd.DataFrame({
    'Number of Features Selected': range(1, len(selected_features_list) + 1),
    'Selected Features': selected_features_list,
    'RMSE': rmse_list
})

# Highlight the best RMSE value
min_rmse = feature_selection_df['RMSE'].min()
highlight_best_rmse = lambda x: ['background-color: yellow' if v == min_rmse else '' for v in x]

# Display the DataFrame in a styled table format for better visualization
styled_df = feature_selection_df.style.apply(highlight_best_rmse, subset=['RMSE']).set_properties(**{'text-align': 'left'}).set_table_styles([
    dict(selector='th', props=[('text-align', 'left')])
])
display(styled_df)

# Plot the RMSE vs. number of features selected
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(rmse_list) + 1), rmse_list, marker='o', label='RMSE')
plt.xlabel('Number of Features Selected')
plt.ylabel('Root Mean Squared Error (RMSE)')
plt.title('RMSE vs. Number of Features Selected')
plt.legend()
plt.grid(True)
plt.show()

print(f"The best features are: {best_features} with RMSE: {best_rmse:.4f}")

# Create a DataFrame to display the previous best RMSE values and the average RMSE
previous_rmse_df = pd.DataFrame({
    'Run': list(range(1, len(rmse_values) + 1)) + ['Average'],
    'Best RMSE': rmse_values + [average_rmse]
})

# Define a function to apply bold formatting specifically to the "Average" row
def bold_average(s):
    return ['font-weight: bold' if s.name == 'Average' else '' for _ in s]

# Define a function to apply bold formatting to the "Best RMSE" value in the "Average" row
def bold_values(val, average_value):
    return 'font-weight: bold' if val == average_value else ''

# Apply styling to the DataFrame
styled_previous_rmse_df = previous_rmse_df.style.apply(bold_average, axis=1).applymap(lambda val: bold_values(val, average_rmse), subset=['Best RMSE']).set_properties(**{'text-align': 'center'}).set_table_styles([
    dict(selector='th', props=[('text-align', 'center')])
])

# Define a function to apply bold formatting to the "Average" text in the "Run" column
def bold_average_text(s):
    return ['font-weight: bold' if val == 'Average' else '' for val in s]

```

```

# Apply styling to the "Run" column
styled_previous_rmse_df = styled_previous_rmse_df.apply(bold_average_text, subset=['Run'])

display(styled_previous_rmse_df)
print("\nThe average RMSE from all runs is: {:.4f}".format(average_rmse))

# Visualize Actual vs. Predicted values for Biochar Yield
best_rmse, y_test, y_pred, regressor = train_evaluate_lr(X_train[best_features], X_test[best_features], y_train, y_test)
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='skyblue')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], 'k--', lw=2) # Plotting the diagonal line
plt.xlabel('Actual Biochar Yield (%)')
plt.ylabel('Predicted Biochar Yield (%)')
plt.title('Actual vs. Predicted Values for Biochar Yield (%)')
plt.show()

# Calculate the correlation matrix for the features
correlation_matrix = X_scaled.corr()

# Create a heatmap
plt.figure(figsize=(10, 8))
sb.heatmap(correlation_matrix, annot=True, fmt='.2f', cmap='coolwarm', cbar=True)
plt.title('Correlation Heatmap of Features')
plt.show()

# ===== SHAP Implementation =====
# SHAP Analysis
# Ensure X_test_df is a DataFrame with column names before using it in SHAP
X_test_df = pd.DataFrame(X_test.values, columns=features)

# Use the Independent masker to avoid the deprecated feature_perturbation warning
masker = shap.maskers.Independent(X_train[features])

# Create the SHAP explainer using the masker
explainer = shap.LinearExplainer(train_evaluate_lr(X_train[features], X_test[features], y_train, y_test)[3], masker=masker)

# Calculate SHAP values
shap_values = explainer(X_test_df)

# Print SHAP values for a specific instance
instance_index = random.randint(0, X_test_df.shape[0] - 1)
shap_values_instance = shap_values[instance_index]

print("\nSHAP values for instance [instance_index]:")
for feature, feature_value, shap_value in zip(features, X_test_df.iloc[instance_index], shap_values_instance.values):
    print(f"feature: {feature} Value = {feature_value:.4f}, SHAP Value = {shap_value:.4f}")

# Summary plot of SHAP values
shap.summary_plot(shap_values.values, X_test_df, feature_names=features)

# ===== LIME Implementation =====
# LIME Analysis
# Ensure X_train and X_test are DataFrames with column names
lime_explainer = lime.lime_tabular.LimeTabularExplainer(
    X_train[features].values,
    mode='regression',
    feature_names=features,
    verbose=True,
    random_state=20
)

# Select an instance from the test set to explain
i = random.randint(0, X_test_df.shape[0] - 1)
exp = lime_explainer.explain_instance(X_test_df.iloc[i].values, train_evaluate_lr(X_train[features], X_test[features], y_train, y_test)[3].predict, num_features=len(features))

# Print the feature values for the selected instance
print("\nFeature values for the selected instance (Index (i)):")
for feature, value in zip(features, X_test_df.iloc[i].values):
    print(f"feature: {feature}: {value:.4f}")

# Visualize the LIME explanation
exp.show_in_notebook(show_table=True)

```

APPENDIX C

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sb
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler
import random
import os
import shap
import lime
import lime.lime_tabular

# Load the CSV data file into a pandas DataFrame
file_path = r'C:\Users\keswa\OneDrive - Universiti Teknikal Malaysia Melaka\UTEM\BITU 3973 (FYP)\Data-Biochar-Yield.csv'
try:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='latin1')
except UnicodeDecodeError:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='iso-8859-1')

fig = px.histogram(
    df,
    x='Biochar yield (%)',
    histfunc='count',
    template='plotly_dark',
    title='Distribution of Biochar Yield',
    labels={'Biochar yield (%)': 'Biochar Yield (%)'}, # Renaming x-axis Label
)

fig.update_layout(
    xaxis_title='Biochar Yield (%)',
    yaxis_title='Frequency'
)

fig.show()

# Visualisation of all the variables (WNTP)
fig = px.box(df, y=df.drop(['Biochar yield (%)'], axis=1).columns, template='plotly_dark', title='\nBox Plots of all the variables')
fig.show()
print(df.head())

# Select the features and target variable
features = [
    'Fixed carbon', 'Volatile matter', 'Ash', 'Temperature (°C)', 'C', 'H', 'O', 'N', 'Residence time (min)', 'Heating rate (°C/min)'
]
target = 'Biochar yield (%)'

X = df[features]
y = df[target]

# Apply standard scaling to the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Convert scaled features back to DataFrame for easier manipulation
X_scaled = pd.DataFrame(X_scaled, columns=features)

# Print the scaled features
print("\nStandard-scaled features:")
print(X_scaled.head())

# Split the dataset into training and testing subsets
X_train, X_test, y_train, y_test = train_test_split(X_log_transformed, y, test_size=0.2, random_state=20)

# Random Forest Regressor model for Biochar Yield
# Train and evaluate the random forest regressor model
regressor = RandomForestRegressor(random_state=42)
regressor.fit(X_train, y_train)
y_pred = regressor.predict(X_test)
best_rmse = np.sqrt(mean_squared_error(y_test, y_pred))

```

```

# Initialize lists to store results for visualization
rmse_list = []
selected_features_list = []

# Initialize an empty list to store selected features
selected_features = []
best_features = []

# Define the maximum number of features to consider
max_features = len(X.columns)

# Loop over the maximum number of features to select
for i in range(max_features):
    remaining_features = list(set(features) - set(selected_features))
    if remaining_features:
        # Select a random feature from the remaining features
        random_feature = random.choice(remaining_features)
        selected_features.append(random_feature)

        # Train and evaluate random forest regressor model with the selected features
        regressor_current = RandomForestRegressor(random_state=42)
        regressor_current.fit(X_train[selected_features], y_train)
        y_pred_current = regressor_current.predict(X_test[selected_features])
        current_rmse = np.sqrt(mean_squared_error(y_test, y_pred_current))

        # Store the RMSE and the selected features
        rmse_list.append(current_rmse)
        selected_features_list.append(list(selected_features))

        # Update the best RMSE and best features if the current RMSE is better
        if current_rmse < best_rmse:
            best_rmse = current_rmse
            best_features = list(selected_features)

# Save the best RMSE value to a file
rmse_file = 'best_rmse_values_randomforest(s).txt'
if os.path.exists(rmse_file):
    with open(rmse_file, 'a') as file:
        file.write(f'{best_rmse}\n')
else:
    with open(rmse_file, 'w') as file:
        file.write(f'{best_rmse}\n')

# Read the RMSE values from the file and compute the average
with open(rmse_file, 'r') as file:
    rmse_values = [float(line.strip()) for line in file.readlines() if line.strip()]
average_rmse = np.mean(rmse_values)

# Create a DataFrame to display the selected features at each step
feature_selection_df = pd.DataFrame({
    'Number of Features Selected': range(1, len(selected_features_list) + 1),
    'Selected Features': selected_features_list,
    'RMSE': rmse_list
})

# Highlight the best RMSE value
min_rmse = feature_selection_df['RMSE'].min()
highlight_best_rmse = lambda x: ['background-color: yellow' if v == min_rmse else '' for v in x]

# Display the DataFrame in a styled table format for better visualization
styled_df = feature_selection_df.style.apply(highlight_best_rmse, subset=['RMSE']).set_properties(**{'text-align': 'left'}).set_table_styles([
    dict(selector='th', props=[('text-align', 'left')])
])
display(styled_df)

# Plot the RMSE vs. number of features selected
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(rmse_list) + 1), rmse_list, marker='o', label='RMSE')
plt.axhline(y=min_rmse, color='r', linestyle='--', label='Best RMSE')
plt.xlabel('Number of Features Selected')
plt.ylabel('Root Mean Squared Error (RMSE)')
plt.title('RMSE vs. Number of Features Selected')
plt.legend()
plt.grid(True)
plt.show()

```

```

print(f"The best features are: {best_features} with RMSE: {best_rmse:.4f}")

# Create a DataFrame to display the previous best RMSE values and the average RMSE
previous_rmse_df = pd.DataFrame({
    'Run': list(range(1, len(rmse_values) + 1)) + ['Average'],
    'Best RMSE': rmse_values + [average_rmse]
})

# Define a function to apply bold formatting specifically to the "Average" row
def bold_average(s):
    return ['font-weight: bold' if s.name == 'Average' else '' for _ in s]

# Define a function to apply bold formatting to the "Best RMSE" value in the "Average" row
def bold_values(val, average_value):
    return ['font-weight: bold' if val == average_value else '']

# Apply styling to the DataFrame
styled_previous_rmse_df = previous_rmse_df.style.apply(bold_average, axis=1).applymap(lambda val: bold_values(val, average_rmse), subset=['Best RMSE']).set_properties(**{'text-align': 'center'}).set_table_styles(
    dict(selector="th", props=[('text-align', 'center')]))

# Define a function to apply bold formatting to the "Average" text in the "Run" column
def bold_average_text(s):
    return ['font-weight: bold' if val == 'Average' else '' for val in s]

# Apply styling to the "Run" column
styled_previous_rmse_df = styled_previous_rmse_df.apply(bold_average_text, subset=['Run'])

display(styled_previous_rmse_df)
print(f"\n\nThe average RMSE from all runs is: {average_rmse:.4f}")
print("\033[1;31;47m\n\nThe average RMSE from all runs is: {:.4f}\033[0m".format(average_rmse))

```

```

# Visualize Actual vs. Predicted values for Biochar Yield
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='skyblue')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], 'k--', lw=2) # Plotting the diagonal line
plt.xlabel('Actual Biochar Yield (%)')
plt.ylabel('Predicted Biochar Yield (%)')
plt.title('Actual vs. Predicted Values for Biochar Yield (%)')
plt.show()

```

```

# Calculate the correlation matrix for the features
correlation_matrix = X_log_transformed.corr()

# Create a heatmap
plt.figure(figsize=(10, 8))
sb.heatmap(correlation_matrix, annot=True, fmt='.2f', cmap='coolwarm', cbar=True)
plt.title('Correlation Heatmap of Features')
plt.show()

```

```

# ===== SHAP Implementation =====
# SHAP Analysis
# Ensure X_test_df is a DataFrame with column names before using it in SHAP
X_test_df = pd.DataFrame(X_test.values, columns=features)

# Create the SHAP explainer using TreeExplainer
explainer = shap.TreeExplainer(regressor)

# Calculate SHAP values
shap_values = explainer.shap_values(X_test_df)

# Print SHAP values for a specific instance
instance_index = random.randint(0, X_test_df.shape[0] - 1)
shap_values_instance = shap_values[instance_index]

```

```

print(f"\nSHAP values for instance {instance_index}:")
for feature, feature_value, shap_value in zip(features, X_test_df.iloc[instance_index], shap_values_instance):
    print(f"{feature}: Feature Value = {feature_value:.4f}, SHAP Value = {shap_value:.4f}")

# Summary plot of SHAP values
shap.summary_plot(shap_values, X_test_df, feature_names=features)

# ===== LIME Implementation =====
# LIME Analysis
# Ensure X_train and X_test are DataFrames with column names
lime_explainer = lime.lime_tabular.LimeTabularExplainer(
    X_train[features].values,
    mode='regression',
    feature_names=features,
    verbose=True,
    random_state=20
)

# Select an instance from the test set to explain
i = random.randint(0, X_test.shape[0] - 1)
exp = lime_explainer.explain_instance(X_test.iloc[i].values, regressor.predict, num_features=len(features))

# Print the feature values for the selected instance
print(f"\nFeature values for the selected instance (Index {i}):")
for feature, value in zip(features, X_test.iloc[i].values):
    print(f"{feature}: {value:.4f}")

# Print the LIME explanation
print(f"\nLIME explanation for the selected instance (Index {i}):")
for feature, explanation in exp.as_list():
    print(f"{feature}: {explanation}")

# Visualize the LIME explanation
exp.show_in_notebook(show_table=True)

```

اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

APPENDIX D

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sb
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error, r2_score
import shap
import lime
import lime.lime_tabular
import random
import os

# Load the CSV data file into a pandas DataFrame
file_path = r"C:\Users\keswa\OneDrive - Universiti Teknikal Malaysia Melaka\UTEM\BITU 3973 (FYP)\Data-Biochar-Yield.csv'
try:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='latin1')
except UnicodeDecodeError:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='iso-8859-1')

# Visualization
fig = px.histogram(
    df,
    x='Biochar yield (%)',
    histfunc='count',
    template='plotly_dark',
    title='Distribution of Biochar Yield',
    labels={'Biochar yield (%)': 'Biochar Yield (%)'}, # Renaming x-axis label
)
fig.update_layout(
    xaxis_title='Biochar Yield (%)',
    yaxis_title='Frequency'
)
fig.show()

# Visualization of all the variables
fig = px.box(df, y=df.drop(['Biochar yield (%)'], axis=1).columns, template='plotly_dark', title='\nBox Plots of all the variables')
fig.show()

print(df.head())

# Select the features and target variable
features = [
    'Fixed carbon', 'Volatile matter', 'Ash', 'Temperature (°C)', 'C', 'H', 'O', 'N', 'Residence time (min)', 'Heating rate (°C/min)'
]
target = 'Biochar yield (%)'
X = df[features]
y = df[target]

# Apply log transformation to the features
X_log_transformed = X.copy()
for col in X_log_transformed.columns:
    if X_log_transformed[col].min() <= 0:
        X_log_transformed[col] = np.log1p(X_log_transformed[col])
    else:
        X_log_transformed[col] = np.log(X_log_transformed[col])

print("\nLog-transformed features:")
print(X_log_transformed.head())

# Split the dataset into training and testing subsets
X_train, X_test, y_train, y_test = train_test_split(X_log_transformed, y, test_size=0.2, random_state=20)

# Define the function to train and evaluate the KNN model
def train_evaluate_knn(X_train, X_test, y_train, y_test, n_neighbors):
    knn = KNeighborsRegressor(n_neighbors=n_neighbors)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)
    return rmse, y_pred, knn

```

```

# Initialize lists to store results for visualization
rmse_list = []
selected_features_list = []

# Initialize an empty list to store selected features
selected_features = []
best_rmse = float('inf') # Initialize with a high value
best_features = []

# Define the maximum number of features to consider
max_features = len(X_log_transformed.columns)

# Loop over the maximum number of features to select
for i in range(max_features):
    remaining_features = list(set(X_log_transformed.columns) - set(selected_features))
    if remaining_features:
        # Select a random feature from the remaining features
        random_feature = random.choice(remaining_features)
        selected_features.append(random_feature)

        # Train and evaluate KNN model with the selected features
        current_rmse, _, _ = train_evaluate_knn(
            X_train[selected_features], X_test[selected_features], y_train, y_test, n_neighbors=5
        )

        # Store the RMSE and the selected features
        rmse_list.append(current_rmse)
        selected_features_list.append(list(selected_features))

        # Update the best RMSE and best features if the current RMSE is better
        if current_rmse < best_rmse:
            best_rmse = current_rmse
            best_features = list(selected_features)

# Save the best RMSE value to a file
rmse_file = 'best_rmse_values_knn.txt'
if os.path.exists(rmse_file):
    with open(rmse_file, 'a') as file:
        file.write(f'{best_rmse}\n')
else:
    with open(rmse_file, 'w') as file:
        file.write(f'{best_rmse}\n')

# Read the RMSE values from the file and compute the average
with open(rmse_file, 'r') as file:
    rmse_values = [float(line.strip()) for line in file.readlines()]

average_rmse = np.mean(rmse_values)

# Create a DataFrame to display the selected features at each step
feature_selection_df = pd.DataFrame({
    'Number of Features Selected': range(1, len(selected_features_list) + 1),
    'Selected Features': selected_features_list,
    'RMSE': rmse_list
})

# Highlight the best RMSE value
min_rmse = feature_selection_df['RMSE'].min()
highlight_best_rmse = lambda x: ['background-color: yellow' if v == min_rmse else '' for v in x]

# Display the DataFrame in a styled table format for better visualization
styled_df = feature_selection_df.style.apply(highlight_best_rmse, subset=['RMSE']).set_properties(**{'text-align': 'left'}).set_table_styles([
    dict(selector='th', props=[('text-align', 'left')])
])
display(styled_df)

```

```

# Plot the RMSE vs. number of features selected
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(rmse_list) + 1), rmse_list, markers='o', label='RMSE')
plt.axhline(y=rmse_min, color='r', linestyle='--', label='Best RMSE')
plt.xlabel('Number of Features Selected')
plt.ylabel('Root Mean Squared Error (RMSE)')
plt.title('RMSE vs. Number of Features Selected')
plt.legend()
plt.grid(True)
plt.show()

print(f"The best features are: {best_features} with RMSE: {best_rmse:.4f}")

# Train and evaluate KNN model with the best features
best_rmse, best_y_pred, best_knn = train_evaluate_knn(
    X_train[best_features], X_test[best_features], y_train, y_test, n_neighbors=5
)

# Create a DataFrame to display the previous best RMSE values and the average RMSE
previous_rmse_df = pd.DataFrame({
    'Run': list(range(1, len(rmse_values) + 1)) + ['Average'],
    'Best RMSE': rmse_values + [average_rmse]
})

# Define a function to apply bold formatting specifically to the "Average" row
def bold_average(s):
    return ['font-weight: bold' if s.name == 'Average' else '' for _ in s]

# Define a function to apply bold formatting to the "Best RMSE" value in the "Average" row
def bold_values(val, average_value):
    return 'font-weight: bold' if val == average_value else ''

# Apply styling to the DataFrame
styled_previous_rmse_df = previous_rmse_df.style.apply(bold_average, axis=1).applymap(lambda val: bold_values(val, average_rmse), subset=['Best RMSE']).set_properties(**{'text-align': 'center'}).set_table_style(dict(selector='th', props[('text-align', 'center')]))
)

```

```

# Define a function to apply bold formatting to the "Average" text in the "Run" column

```

```

def bold_average_text(s):
    return ['font-weight: bold' if val == 'Average' else '' for val in s]

```

```

# Apply styling to the "Run" column

```

```

styled_previous_rmse_df = styled_previous_rmse_df.apply(bold_average_text, subset=['Run'])

```

```

display(styled_previous_rmse_df)

```

```

print(f"\n\033[1;31;47mThe average RMSE from all runs is: {average_rmse:.4f}\033[0m")

```

```

# Visualize Actual vs. Predicted values for Biochar Yield using the best KNN model

```

```

plt.figure(figsize=(8, 6))
plt.scatter(y_test, best_y_pred, color='skyblue')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], 'k--', lw=2) # Plotting the diagonal line
plt.xlabel('Actual Biochar Yield (%)')
plt.ylabel('Predicted Biochar Yield (%)')
plt.title('Actual vs. Predicted Values for Biochar Yield (%)')
plt.show()

```

```

# Calculate the correlation matrix for the features

```

```

correlation_matrix = X_log_transformed.corr()

```

```

# Create a heatmap

```

```

plt.figure(figsize=(10, 8))
sb.heatmap(correlation_matrix, annot=True, fmt='.2f', cmap='coolwarm', cbar=True)
plt.title('Correlation Heatmap of Features')
plt.show()

```

```

# ===== SHAP Implementation =====

```

```

# Correct X_train and X_test for SHAP

```

```

X_train_best = X_train[best_features]

```

```

X_test_best = X_test[best_features]

```

```

# SHAP Explanation

```

```

explainer = shap.KernelExplainer(best_knn.predict, X_train_best)

```

```

shap_values = explainer.shap_values(X_test_best)

```

```

# Ensure all features are displayed in SHAP summary plot

```

```

shap.summary_plot(shap_values, X_test_best, max_display=len(best_features))

```

```

# ===== LIME Implementation =====

```

```

# LIME Explanation

```

```

lime_explainer = lime.lime_tabular.LimeTabularExplainer(
    training_data=X_train_best.values,
    feature_names=best_features,
    mode='regression'
)

```

```

# Explain an instance using the best features

```

```

instance_to_explain = X_test_best.iloc[0].values

```

```

lime_exp = lime_explainer.explain_instance(instance_to_explain, best_knn.predict, num_features=len(best_features))

```

```

# Show the explanation

```

```

lime_exp.show_in_notebook()

```

APPENDIX E

```

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import plotly.express as px
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Input
from sklearn.metrics import mean_squared_error
import tensorflow as tf
import shap
import lime
import lime.lime_tabular
import seaborn as sns

# Load the CSV data file into a pandas DataFrame
file_path = r'C:\Users\keswa\OneDrive - Universiti Teknikal Malaysia Melaka\UTeM\BITU 3973 (FYP)\Data-Biochar-Yield.csv'
try:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='latin1')
except UnicodeDecodeError:
    df = pd.read_csv(file_path, index_col='Unnamed: 0', encoding='iso-8859-1')

# Data pre-processing
features_values = [
    'Fixed carbon',
    'Volatile matter',
    'Ash',
    'C',
    'H',
    'O',
    'N',
    'Residence time (min)',
    'Temperature (°C)',
    'Heating rate (°C/min)',
]

fig = px.histogram(
    df,
    x='Biochar yield (%)',
    histfunc='count',
    template='plotly_dark',
    title='Distribution of Biochar Yield',
    labels={'Biochar yield (%)': 'Biochar Yield (%)'},
)

fig.update_layout(
    xaxis_title='Biochar Yield (%)',
    yaxis_title='Frequency'
)

fig.show()

# Visualization of all the variables (MNTP)
fig = px.box(df, y=df.drop(['Biochar yield (%)'], axis=1).columns, template='plotly_dark', title='Box Plots of all the variables')
fig.show()

print(df.head())

# Define features and Label
X = df.drop(['Biochar yield (%)'], axis=1)
y_bio = df['Biochar yield (%)']

# Apply Logarithmic transformation to features with non-negative values
X_log_transformed = X.copy()
for col in X_log_transformed.columns:
    # Ensure non-negative values for log transformation
    X_log_transformed[col] = np.log1p(X_log_transformed[col])

print("\nLog-transformed features:")
print(X_log_transformed.head())

# Split the dataset into training and testing subsets for Biochar Yield
X_train, X_test, y_train, y_test = train_test_split(X_log_transformed, y_bio, test_size=0.2, random_state=22)

```

```

# Neural network model
model = Sequential([
    Input(shape=X_train.shape[1,]),
    Dense(64, activation='relu'),
    Dense(64, activation='relu'),
    Dense(1) # Output layer for regression
])

# Compile the model with mean squared error loss
model.compile(optimizer='adam', loss='mean_squared_error', metrics=[tf.keras.metrics.RootMeanSquaredError(name='rmse')])

# Train the model using training data
history = model.fit(
    X_train, y_train,
    epochs=100,
    validation_data=(X_test, y_test),
    verbose=1
)

# Extract RMSE for training and validation data
train_rmse = history.history['rmse'][-1]
val_rmse = history.history['val_rmse'][-1]
print("\n\033[1;33;40mFinal Training RMSE: ", train_rmse)
print("\n\033[1;31;47mFinal Validation RMSE: ", val_rmse)

# Save the best Validation RMSE value to a file
rmse_file = 'best_rmse_values_neural_network.txt'
if os.path.exists(rmse_file):
    with open(rmse_file, 'a') as file:
        file.write(f'{val_rmse}\n')
else:
    with open(rmse_file, 'w') as file:
        file.write(f'{val_rmse}\n')

# Read the RMSE values from the file and compute the average
with open(rmse_file, 'r') as file:
    rmse_values = [float(line.strip()) for line in file.readlines() if line.strip()]

average_rmse = np.mean(rmse_values)

# Create a DataFrame to display the previous best RMSE values and the average RMSE
previous_rmse_df = pd.DataFrame({
    'Run': list(range(1, len(rmse_values) + 1)) + ['Average'],
    'Best RMSE': rmse_values + [average_rmse]
})

# Define a function to apply bold formatting specifically to the "Average" row
def bold_average(s):
    return ['font-weight: bold' if s.name == 'Average' else '' for _ in s]

# Define a function to apply bold formatting to the "Best RMSE" value in the "Average" row
def bold_average_val(average_val):
    return 'font-weight: bold' if val == average_val else ''

# Apply styling to the DataFrame
styled_previous_rmse_df = previous_rmse_df.style.apply(bold_average, axis=1).applymap(lambda val: bold_values(val, average_rmse), subset=['Best RMSE']).set_properties(**{'text-align': 'center'}).set_table_styles(
    dict(selectors='th', props=[('text-align', 'center')])
)

# Define a function to apply bold formatting to the "Average" text in the "Run" column
def bold_average_text(s):
    return ['font-weight: bold' if val == 'Average' else '' for val in s]

# Apply styling to the "Run" column
styled_previous_rmse_df = styled_previous_rmse_df.apply(bold_average_text, subset=['Run'])

# Display the styled DataFrame
display(styled_previous_rmse_df)

print(f"\n\033[1mAverage Validation RMSE from all runs: {average_rmse:.4f}\033[0m")

# Plot training and validation loss
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.title('Training and Validation Loss of Biochar Yield (%)')
plt.show()

# SHAP analysis using the unified SHAP Explainer
sample_size = 100 # You can adjust this size
X_train_sample = X_train[:sample_size]

# Initialize SHAP Explainer
explainer = shap.Explainer(model, X_train_sample)
shap_values = explainer(X_test)

# Plot SHAP summary plot
shap.summary_plot(shap_values, X_test, feature_names=X.columns)

# LIME analysis
lime_explainer = lime.lime_tabular.LimeTabularExplainer(X_train.values, feature_names=X.columns, class_names=['Biochar Yield'], verbose=True, mode='regression')

i = np.random.randint(0, X_test.shape[0])
exp = lime_explainer.explain_instance(X_test.values[i], model.predict, num_features=5)
exp.show_in_notebook(show_table=True)

# Compute the correlation matrix
correlation_matrix = df.corr()

# Plot the correlation heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('Correlation Heatmap of Features')
plt.show()

```