UNIVERSITI TEKNOLOGI MARA

ARTIFICIAL NEURAL NETWORK MODELLING OF STUDENT'S ACADEMIC PERFORMANCE BASED ON ONLINE LEARNING ACTIVITIES PATTERN

FAIRUL NAZMIE BIN OSMAN

PhD

April 2026

UNIVERSITI TEKNOLOGI MARA

ARTIFICIAL NEURAL NETWORK MODELLING OF STUDENT'S ACADEMIC PERFORMANCE BASED ON ONLINE LEARNING ACTIVITIES PATTERN

FAIRUL NAZMIE BIN OSMAN

Thesis submitted in fulfilment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)

Faculty of Electrical Engineering

April 2026

CONFIRMATION BY PANEL OF EXAMINERS

I certify that a Panel of Examiners has met on 6th February 2026 to conduct the final examination of Fairul Nazmie Osman on his Doctor of Philosophy thesis entitled "Artificial Neural Network Modelling of Student's Academic Performance Based on Online Learning Activities Pattern" in accordance with Universiti Teknologi MARA Act 1976 (Akta 173). The Panel of Examiners recommends that the student be awarded the relevant degree. The Panel of Examiners was as follows:

Mohamad Fahmi Hussin @ Mohamad, PhD
Associate Professor
Faculty of Electrical Engineering
Universiti Teknologi MARA
(Chairman)

Murizah Kassim, PhD
Professor
Faculty of Electrical Engineering
Universiti Teknologi MARA
(Internal Examiner)

Rosmiwati Mohd Mokhtar, PhD
Associate Professor
School of Electrical and Electronic Engineering
Universiti Sains Malaysia
(External Examiner)

PROFESSOR DR HJH ZURAEDA IBRAHIM
Dean
Institute of Postgraduates Studies
Universiti Teknologi MARA
Date : 23 April 2026

AUTHOR'S DECLARATION

I declare that the work in this thesis was carried out in accordance with the regulations of Universiti Teknologi MARA. It is original and is the results of my own work, unless otherwise indicated or acknowledged as referenced work. This thesis has not been submitted to any other academic institution or non-academic institution for any degree or qualification.

I, hereby, acknowledge that I have been supplied with the Academic Rules and Regulations for Postgraduate, Universiti Teknologi MARA, regulating the conduct of my study and research.

Name of Student	:	Fairul Nazmie Bin Osman
Student ID. No.	:	2020825992
Programme	:	Doctor of Philosophy (Electrical Engineering) - CEEE950
Faculty	:	Electrical Engineering
Thesis Title	:	Artificial Neural Network Modelling of Student's Academic Performance Based on Online Learning Activities Pattern
Signature of Student	:	……..
Date	:	April 2026

ABSTRACT

Early and accurate prediction of student academic performance is critical for enabling timely interventions. This research proposes an intelligent classification framework that models academic performance using real-time engagement data collected from an online learning platform during the first eight (8) weeks out of 17 weeks of a programming course. The study addresses four core challenges in predictive academic modelling: reliance on outdated static data, class imbalance in small dataset, underutilization of multi-activity features, and suboptimal hyperparameter tuning in neural network architectures. A dataset of 99 samples with 12 academic engagement features was collected, pre-processed using Min-Max normalization, and balanced using the Adaptive Synthetic Sampling (ADASYN) algorithm with synthetic data was statistically verified using the Kolmogorov–Smirnov test, histogram comparisons, and boxplot analyses. Feature selection was conducted using Mutual Information, Recursive Feature Elimination (RFE), Random Forest Importance, L1 Regularization Least Absolute Shrinkage and Selection Operator (L1 Lasso), and eXtreme Gradient Boosting (XGBoost) to identify the optimal predictors. A Multilayer Perceptron (MLP) classifier was developed and optimized using five optimizers: Adaptive Moment Estimation with Decoupled Weight Decay (AdamW), Nesterov-accelerated Adaptive Moment Estimation (Nadam), Adaptive Moment Estimation with Maximum of Past Squared Gradients (AmsGrad), Adaptive Gradient Algorithm (AdaGrad), and Stochastic Gradient Descent with Momentum (SGD with Momentum). Experimental results demonstrate that RFE-selected features combined with adaptive optimizers significantly enhance model generalization and stability. The MLP consistently outperformed other architectures including Support Vector Machine (SVM), k-Nearest Neighbors (kNN), CNN, Recurrent Neural Network (RNN), and Long Short-Term Memory Network (LSTM), achieving an F1-score of 90.0% and a testing accuracy of 86.7%, affirming its robustness for non-sequential educational data. Key outcomes from this research include ADASYN-integrated MLP for improved minority-class sensitivity, feature-efficient framework (six selected features via RFE), and empirical optimizer comparison (AdamW, Nadam, AmsGrad, AdaGrad, SDG-Momentum) This research advances scalable academic early-warning systems aligned with Malaysia's Outcome-Based Education (OBE) and Continuous Quality Improvement (CQI) initiatives, offering a real-time, pedagogically actionable tool for educators.

ACKNOWLEDGEMENT

Praise for Allah the Almighty, the Most Gracious, and the Most Merciful for His blessings and guidance given to endure on of the most memorable journey of my life. May Allah's blessing goes to His final Prophet Muhammad(صلى الله عليه وسلم), his family and his companions.

First and foremost, I would like to express my warmest gratitude and appreciation to Prof. Ir. Ts. Dr. Hj. Mohd Nasir Taib and Ts. Dr. Mohd Azri Abdul Aziz, my supervisors for their patient, advice, the confident they have in me to pursue this. Without them, none of these are possible. May Allah bless them until here after. Prayers and best wishes to my colleagues in the Advance Signal Processing Research Group (ASPRG) and Faculty of Electrical Engineering, UiTM for relentless support and word of wisdom with ever helping suggestions and excellent discussion that helps to the formation of this thesis.

My wife, Nuraiesas Ismail, my three wonderful sons, Faiz Nazrin, Faiz Nazril and Faiz Nurhan: you have always been the pillars of my strength. Diddy love you all dearly. My late mother, [redacted]. my greatest siblings Fazrul Nizam Osman and Mr Faizal Nazli Osman and my in laws family, I love you guys so much.

Lastly, I would like to convey my wholehearted appreciation to a group of extraordinary beings whose contributions, though unnamed, provided persistent support and encouragement bringing the sweetest end to another chapter of my life.

THANK YOU …

Fairul Nazmie Osman

19 Feb 2026 / Ramadhan 1447

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Symbols

|X_majority|	Number of Samples in the Majority Class
|X_minority|	Number of Samples in the Minority Class
a	Activation Output
b	Bias Parameter in Neural Network
C	Regularization Parameter in Support Vector Machine
D	Kolmogorov–Smirnov Test Statistic
Dcritical	Critical Value for KS test At Significance Level α
d	Degree of Class Imbalance ($0 < d \leq 1$)
F(x)	Theoretical cumulative Distribution Function
FN	False Negative
FP	False Positive
Fn(x)	Empirical Distribution Function
G	Total number of Synthetic Data Points to Generate
Gt	Sum of Squared Gradients Up To Time Step *t* (Adagrad)
g_i	Number of Synthetic Samples Generated For Minority Instance *i*
g_t	Gradient at Time Step *t*
I(·)	Indicator Function (Returns 1 If Condition is True, 0 Otherwise)

IQR	Interquartile Range ($Q_3 - Q_1$)
k	Number of Nearest Neighbours (typically $k = 5$)
m_t	First Moment Estimate (Exponential Moving Average of Gradients)
$\hat{m}_t$	Bias-corrected First Moment Estimate
n	Number of Samples
p	Number of Features
Q_1	First Quartile (25th Percentile)
Q_2	Second Quartile (Median, 50th Percentile)
Q_3	Third Quartile (75th Percentile)
r_i	Ratio of majority Class Samples Among *K*-Nearest Neighbours of Minority Instance x_i
$\hat{r}_i$	Normalized Difficulty Score for Minority Instance *i*
t	Time Step or Iteration Number
TN	True Negative
TP	True Positive
v	Velocity Term in SGD with Momentum
v_t	Second Moment Estimate (Exponential Moving Average Of Squared Gradients)
$\hat{v}_t$	Bias-Corrected Second Moment Estimate (Adam/AdamW) or Maximum of Past Second Moment Estimates (AmsGrad)
w	Weight Parameter in Neural Network
X	Original Value of the Features
Xi	Feature Vector for Instance *i*

Xmax	Maximum Value in the Dataset for That Feature
Xmin	Minimum Value in the Dataset for That Feature
Xnormalized	Rescaled Value in the Range [0,1]
y_i	Target Label for Instance *i*
y_j	Class Label of Neighbour *j*
z	Weighted Sum of Inputs (Pre-Activation Value)

LIST OF ABBREVIATIONS

Abbreviations

ADASYN	Adaptive Synthetic Sampling
Adam	Adaptive Moment Estimation
Adagrad	Adaptive Gradient Algorithm
AdamW	Adaptive Moment Estimation with Decoupled Weight Decay
AI	Artificial Intelligence
AmsGrad	Adaptive Moment Estimation with Maximum
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
CO	Course Outcomes
CQI	Continuous Quality Improvement
EAC	Engineering Accreditation Council
ECE	Electronics and Electrical Engineering
EDF	Empirical Distribution Function
EDM	Educational Data Mining
FN	False Negatives
FP	False Positives
GPA	Grade Point Average
IQR	Interquartile Range
KNN	k-Nearest Neighbours

KS	Kolmogorov–Smirnov
LMS	Learning Management System
LSTM	Long Short-Term Memory
MI	Mutual Information
MLP	Multilayer Perceptron
MOOC	Massive Open Online Course
NAG	Nesterov-Accelerated Gradient
Nadam	Nesterov-Accelerated Adaptive Moment Estimation
OBE	Outcome-Based Education
PCA	Principal Component Analysis
PDF	Probability Density Function
PO	Programme Outcomes
Q-Q	Quantile–Quantile
RBF	Radial Basis Function
RFE	Recursive Feature Elimination
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SMOTE	Synthetic Minority Over-sampling Technique
SVM	Support Vector Machine
TN	True Negatives
TP	True Positives
TVET	Technical and Vocational Education and Training
UiTM	Universiti Teknologi MARA

CHAPTER 1
INTRODUCTION

1.1 Research Background

Education is universally acknowledged as a cornerstone for societal development and progress, crucially shaping the destinies of nations globally. In developing countries, serving as a catalyst for economic growth, social advancement, and political stability. Malaysia capitalized this dynamic, where concerted efforts to elevate educational standards are propelling the nation towards a knowledge-based economy. These initiatives have not only expedited Malaysia's economic development but have also strengthened social cohesion and enriched the understanding of cultural diversity within its diversely rich population [1].

The Malaysian government's strategic focus on education reform encompasses multiple facets, including improving teaching quality, modernizing curricula, and ensuring equitable access to education across all socio-economic strata. Investments in educational infrastructure, digital learning technologies, and lifelong learning programs have been pivotal in fostering a workforce equipped with the competencies required for the digital age. By addressing disparities in educational opportunities and promoting inclusivity, these reforms contribute significantly to national unity and social mobility. Furthermore, Malaysia's alignment with global education benchmarks ensures that its graduates possess the critical thinking skills and technical expertise necessary to drive innovation and sustain long-term economic growth [2], [3].

Malaysian government, heavily invests in higher education, including operating costs and student aid. This spending, manage by several ministries which included the Ministry of Education, Ministry of Higher Education and allocations for TVET totalled RM 77.2 billion for 2023, RM81.8 billion in 2024, RM89.6 billion in 2025 [3], [4], [5]. These investments account for nearly 20% of the national annual budget, highlighting the substantial economic impact of these expenditures at a macroeconomic level. This substantial allocation underscores the critical importance of effective educational spending in supporting the nation's development goals. With such significant investment, reducing student dropout rate is vital to prevent financial loss and wasted potential [6].

The domain of academic performance analysis being integrated with artificial intelligence (AI), which includes the classification, modelling, forecasting, and prediction of student outcomes, represents a vibrant and expansive field of inquiry with extensive implications across educational systems. Researchers globally are engaged in exploring a diverse array of dimensions within this area, encompassing various data types such as demographic [7], factors [8], academic grades [9], [10], and financial backgrounds [7]. Additionally, they employ a range of methodological approaches, including advanced hybrid data mining techniques [11], [12], [13] and comprehensive attendance monitoring systems [14], [15]. The overarching goal of these scholarly efforts is to devise robust systems capable of accurately classifying and predicting student academic trajectories.

Despite extensive research in academic performance prediction, most studies still rely heavily on historical data, overlooking the untapped potential of current semester information. Real-time academic data offers immediate insight into students' ongoing progress and challenges, enabling early identification of those at risk and supporting timely interventions. However, many existing approaches pre-categorize students based on past records, introducing bias and limiting model adaptability. This neglects the dynamic nature of student performance, which can fluctuate due to both academic and non-academic factors, underscoring the need for predictive frameworks that leverage real-time data to enhance responsiveness and accuracy.

In response to these limitations, there is a growing need for predictive frameworks that move beyond static, retrospective data and instead capitalize on real-time academic activity to support early detection of at-risk students. A gap remains in proactive intervention models that leverage such data to detect academic struggles before semester-end evaluations. Developing these systems would enable timely support, shifting from reactive to preventive strategies. This paradigm not only enhances the responsiveness of educational interventions but also aligns with Malaysia's broader vision of cultivating an agile, data-informed, and inclusive learning ecosystem, one that can meet the evolving demands of modern education and sustaining national development goals.

1.2 Problem Statement

Current student performance prediction models are limited by their reliance on outdated or static academic data, failing to capture real-time learning activities that occur throughout the semester [16] . This lack of real-time data integration restricts predictive accuracy and delays the identification of at-risk students, reducing the opportunity for timely intervention. There is a need to develop a predictive model that effectively incorporates ongoing academic behaviours to support early academic support and formative assessment strategies.

In addition, imbalanced datasets are a persistent challenge in academic performance prediction, often leading to biased models that perform poorly on minority-class outcomes. While synthetic oversampling techniques like ADASYN have shown promise in other domains, their application in education remains underexplored [17]. Unlike conventional methods, ADASYN adaptively focuses on harder-to-learn instances, improving both class distribution and prediction fairness. This gap highlights the need for synthetic data augmentation to enhance the robustness and generalizability of predictive models in education.

Existing models also struggle to integrate and analyse multiple academic activity features collected over time [18], resulting in suboptimal prediction of final outcomes. Constructing and validating an MLP model that captures diverse academic behaviours across 12 distinct features can improve prediction precision and support more effective, data-informed academic decisions.

Finally, the performance of MLP models is highly sensitive to hyperparameter configuration. Due to the complex and variable nature of educational data, default or poorly tuned settings may result in unstable or inaccurate predictions. Current approaches often overlook systematic optimization of learning rate, hidden layer size, and activation functions. There is a critical need to apply and evaluate suitable optimizer algorithms to identify optimal MLP configurations and ensure consistent, high-performing predictive outcomes in educational settings.

1.3 Proposed Work

To tackle the identified challenges in predicting student academic performance, the first step in the solution involves the comprehensive characterizing data from

ongoing academic activities throughout the semester. This means collecting and analysing real-time data inputs, such as attendance, assignment submissions, quiz scores and participation. By focusing on the most recent information, the model will dynamically reflect real-time student engagement and performance, ensuring predictions are both timely and accurate. This approach will allow educators to spot potential academic issues early allowing for more effective interventions before assessments.

The second component of the solution involves using the ADASYN (Adaptive Synthetic Sampling) method to generate synthetic data that addresses the problem of class imbalance in the dataset. By synthesizing additional data points for underrepresented categories, the model will be trained on a more balanced dataset, improving its ability to accurately predict outcomes. This synthesis and analysis of synthetic data will enhance the model's robustness and ensure that it can provide reliable predictions even in scenarios where certain student performance patterns are less common.

The third solution is to construct and optimize a Multilayer Perceptron (MLP) model using the data characterized and synthesized in the previous steps. The model will be trained to predict final student outcomes based on 12 distinct academic activities conducted throughout the semester.

The final solution compares the classification performance of the MLP model with other ANN-based classifiers to assess its effectiveness in educational data prediction. The comparison focuses on generalization ability, robustness, and stability under varying feature configurations. This evaluation provides a comprehensive understanding of MLP's performance relative to other neural network architectures, emphasizing its adaptability and consistency across different experimental setups.

1.4 Research Objectives

This thesis aims to develop an MLP model that can accurately identify potential student underperformance early in the academic process, specifically by exploring and utilizing significant student activities within online learning platforms. To achieve this, the research has several focused sub-objectives:

- To analyse real-time data from ongoing academic activities during the first eight

weeks out of 17 academic weeks to develop an early-stage prediction model for classifying students into GPA ≥ 3.00 or GPA < 3.00 categories for a programming course in UiTM.

- To generate and analyse synthetic data using the ADASYN method to develop a balanced dataset for training the prediction model and to assess its effectiveness in improving minority-class prediction performance.
- To develop and validate an MLP model designed to predict students' final academic results by effectively integrating and analysing data from 12 distinct academic activities conducted throughout the semester.
- To compare and evaluate the classification performance of MLP with other machine learning classifiers and establish its suitability for educational data prediction based on generalization ability and model robustness.

1.5 Significance of Study

The significance of this study lies in its multifaceted contributions to educational data mining and student performance prediction, particularly within the context of OBE. By addressing key limitations in existing models, such as the reliance on static historical data, class imbalance, and limited optimization strategies, the thesis proposes an integrated framework that is both data-driven and pedagogically informed. The significance lies in its educational impact, while the novelty resides in its methodological innovation. The significant of the study is demonstrated through its strategic combination of real-time data characterization, adaptive sampling, robust neural network modelling, and systematic hyperparameter tuning. These innovations collectively aim to enhance the predictive accuracy, generalizability, and practical applicability of student performance classifiers, ultimately supporting timely academic interventions and continuous quality improvement in teaching and learning environments.

i. Comprehensive Characterization of Real-Time Data

This study introduces a novel approach by emphasizing the use of near real-time academic activity data collected continuously throughout the semester. Unlike conventional models that depend on static, historical data, this research captures week-by-week student behaviours including attendance, assignment submission, quiz

performance, and tutorial participation to provide a dynamic view of learning engagement. This granular, time-sensitive characterization reflects each student's evolving academic state, enabling earlier detection of underperformance trends and facilitating timely pedagogical interventions consistent with formative assessment and CQI principles. The effectiveness of this characterization was validated through observed correlations between weekly engagement patterns and final performance outcomes. These features subsequently formed the input foundation for the MLP-based model described in the following section.

ii. Focused Development of an MLP Model

Building upon the characterized engagement features described in the previous section, this study develops an MLP classifier designed to process the 12 extracted academic activity variables. These features, derived from students' weekly interactions with learning materials and tasks, form the input layer that represents meaningful indicators of academic engagement. The MLP architecture was structured to capture non-linear relationships between these behavioural features and overall performance outcomes, ensuring that the model reflects authentic learning dynamics observed in real classroom settings. Grounded in practical and pedagogically relevant indicators, the model achieves a balance between computational complexity and interpretability, providing a scalable and educator-friendly framework for predictive analytics in higher education.

iii. Integration of Adaptive Sampling for Class Imbalance

Following the development of the MLP model based on real-time engagement features, this study integrates the ADASYN technique to enhance data balance and model fairness. Since the collected dataset exhibited an uneven distribution between high- and low-performing students, direct model training could lead to bias toward the majority class and reduced sensitivity in detecting at-risk learners. By applying ADASYN, synthetic minority samples are generated adaptively based on data density, producing a more balanced and representative training set. This integration improves the MLP model's ability to generalize across diverse performance levels, strengthening its reliability as both a predictive tool and a foundation for equitable academic support within outcome-based learning environments.

iv. Empirical Optimization Through Hyperparameter Tuning

Building on the balanced and refined dataset generated through ADASYN, the final contribution of this research focuses on the empirical optimization of the MLP model to maximize performance and stability. Systematic hyperparameter tuning and comparative evaluation of multiple optimization algorithms were conducted to enhance the model's learning efficiency and generalization capability. Instead of relying on preset or heuristic configurations, this study employs grid search, random search, and optimizer performance analysis to fine-tune key parameters such as learning rate, epoch count, and network structure. These iterative refinements ensure smoother convergence, reduced overfitting, and greater robustness under varying initialization and data conditions. The optimized MLP model thus achieves consistent, reliable performance suitable for real-time educational prediction and benchmarking against other ANN-based classifiers.

1.6 Scope and Limitation of the Study

By focusing on early-semester online interactions, this study aims to identify predictive indicators of academic success, enabling timely interventions for students at risk of failing. Early prediction of academic performance allows educators and academic administrators to implement necessary measures, aligning OBE principles and contributing to continuous quality improvement in teaching practices.

The study utilizes student performance data from the Computer Programming (ECE431) course, part of the Electronics and Electrical Engineering Bachelor programs at Universiti Teknologi MARA. Data were collected from the first eight weeks out of 17 academic weeks of the 20192 academic term through an online learning platform, encompassing students' interactions, including lecture attendance, note-taking, and system access frequency, as well as their performance in tutorials, exercises, and assessments.

Focusing on key programming topics such as TYPES, OPERATORS, FUNCTIONS, LOOPS, and FLOW CONTROL, this study strategically targets foundational elements critical for both academic success and real-world programming. Mastery of these topics is essential for building a strong programming base, solving complex problems, and writing efficient, maintainable code. By concentrating on these

core areas, the predictive model is designed to accurately reflect students' overall programming proficiency, aligning with industry expectations and ensuring that the findings are both academically rigorous and practically relevant.

The implementation of the MLP prediction model in Python is justified due to Python's advanced computational capabilities, particularly in modelling and validating MLP models. Python offers a flexible and powerful environment for machine learning, with libraries such as TensorFlow, Keras, PyTorch, and Scikit-learn providing robust tools for building and validating MLP models. By leveraging these optimizers, the research ensures effective training of the MLP model, with Python's extensive libraries and strong community support facilitating reproducibility and alignment with best practices, making it an ideal choice for this study.

1.7 Thesis Organization

This thesis is organized into six chapters, each addressing a crucial aspect of the research process.

Chapter 1 provides an overarching introduction, presenting the research background, problem statement, objectives, scope, limitations, and the significance of the study. It also outlines the structure of the thesis, setting the stage for the research journey.

Chapter 2 offers a comprehensive review of the existing literature on academic performance prediction, focusing on methodologies, data types, and pre-processing techniques used in current models. It also critically examines decision-making processes in educational contexts, establishing the foundation for the study's contributions.

Chapter 3 delves into the theoretical framework underpinning the research, discussing relevant theories, models, and concepts that guide the study's approach to predicting student performance.

Chapter 4 describes the research methodology, detailing the processes of data acquisition, pre-processing, and analysis. This chapter also outlines the implementation of the MLP model in Python, utilizing various optimizers

Chapter 5 presents the research findings, tracking the development and validation of the MLP model from initial data collection to final predictions. It includes

an in-depth analysis of the results, demonstrating the effectiveness of the proposed model in predicting student academic performance.

Chapter 6 concludes the thesis by summarizing the key findings and discussing their implications for educational practice. It also suggests avenues for future research, highlighting the potential for further advancements in academic performance prediction methodologies.

CHAPTER 2
LITERATURE REVIEW

2.1 Chapter Overview

This chapter presents a comprehensive literature review that underpins the methodologies and experimental framework employed in this study. Section 2.1 introduces the overarching theme by highlighting the role of predictive analytics in education and the strategic importance of early academic intervention to optimize educational investments. Section 2.2 contextualizes the study within Malaysia's evolving higher education system, emphasizing the OBE framework and the need for dynamic assessment models. Section 2.3 provides a problem-based literature review that identifies four core gaps in current academic prediction research: (i) limited use of real-time academic data, (ii) inadequate handling of class imbalance, (iii) insufficient integration of multiple academic activities, and (iv) lack of hyperparameter optimization in intelligent models like the MLP. Section 2.4 outlines the recent and relevant work addressing each of these problems, thereby justifying the proposed methodological direction. Section 2.5 reviews methodological literature that supports each modelling stage and the chapter concludes with Section 2.6, which summarizes key findings from the literature and lays the theoretical and empirical groundwork for the modelling approaches detailed in Chapter 3 and beyond.

2.2 Overview of Outcome Based Education in Malaysia

Malaysia's higher education system, particularly in engineering, has undergone significant transformation with the adoption of the Outcome-Based Education (OBE) framework aligned with the Washington Accord. Spearheaded by the Engineering Accreditation Council (EAC) [19], OBE emphasizes the attainment and demonstration of clearly defined Programme Outcomes (PO) and Course Outcomes (CO), encompassing not only technical knowledge but also professional competencies such as communication, teamwork, leadership, ethics, and lifelong learning. The OBE framework promotes a structured system of assessment and Continuous Quality

Improvement (CQI), ensuring that curriculum design, teaching strategies, and assessment mechanisms are systematically aligned with desired graduate attributes.

The increasing focus on predictive analytics education is a direct of significant improvements in data collection technologies and data processing capabilities. Researchers have increasingly explored the prediction and classification of students' academic performance by leveraging a wide range of input variables, including student demographics, socio-economic backgrounds, engagement metrics, and classroom resources [20], [21], [22], [23], [24] . This spectrum of variables enhances the predictive power of models and provides academic institutions with robust tools to understand and intervene in student progress across various stages of the academic cycle.

In addition to advancing machine learning and data mining methods for identifying trends in student outcomes [25], [26], [27] the critical importance of academic performance extends well beyond mere course pass-fail outcomes. High academic performance is closely tied to national human capital development, as educational institutions invest heavily in producing graduates capable of contributing to key socio-economic sectors [28]. In countries like Malaysia, where education consumes a significant portion of the national budget, ensuring a high return on these investments becomes very important especially during periods of economic uncertainty. This macro-level perspective underscores the need for effective predictive models that can guide policymakers in optimizing resource allocation and reducing failure and dropout rates [28].

Furthermore, the rising incidents of failure rates in higher education has accelerated the development of predictive models as essential early warning systems. When educators can identify students at risk of failure early in the academic term, they can implement targeted interventions aimed at both academic remediation and dropout prevention [22], [29], [30]. Early prediction models based on techniques such as regression [31], decision trees [32], [33] and neural networks [34], [35], [36] have demonstrated success in classifying students into risk categories, thereby enabling timely, personalized responses to student needs. This evidence strongly supports the use of data-driven approaches to continuously monitor student performance, ensuring that potential issues are addressed before they escalate and compromise academic success and institutional sustainability [30], [37], [38].While the OBE model promotes continuous quality improvement and prepares graduates for the Fourth Industrial Revolution, challenges persist, particularly the continued reliance on traditional,

summative assessment methods that fail to capture the real-time academic progress of students throughout the semester.

To fully realize the aspirations of OBE and CQI, Malaysia's educational system must move beyond static and retrospective models of academic performance prediction [39], [40]. Traditional student analytics frameworks still heavily depend on historical data like entrance scores and final exam results, overlooking real-time academic behaviours such as weekly attendance, quiz participation, and engagement with digital learning tools [41], [42]. This limits early detection of at-risk students and undermines the potential for timely, targeted intervention. Moreover, student performance datasets in Malaysia and other country often suffer from class imbalance [43], [44], where struggling students form the minority group, leading to biased predictive models that disproportionately favour high-achieving cohorts. Furthermore, while intelligent models like the MLP are increasingly employed in learning analytics, many implementations neglect the importance of multi-feature integration and lack systematic hyperparameter tuning, reducing their robustness and practical utility. This underscores the need for a more adaptive, fair, and data-driven predictive modelling framework tailored to Malaysia's OBE-driven, digitally integrated education landscape [45], [46].

In conclusion, the literature clearly demonstrates that enhanced data processing capabilities and the broad spectrum of variables available today have fostered the emergence of sophisticated predictive models that address both academic achievement and dropout rates. These models not only serve the immediate goal of improving student outcomes through early intervention but also contribute to the broader strategic aim of maximizing the national return on educational investments.

2.3 Problem Based Literature Review

The increasing complexity and demands of modern higher education, especially within Malaysia's engineering and technology sectors, have catalysed the adoption OBE, in line with global standards such as the Washington Accord. As educational institutions strive to map CO to PO, assessment mechanisms must evolve to reflect the real-time academic journey of students, rather than relying solely on static performance indicators. In support of this shift, the field of educational data mining has gained momentum, enabling predictive models to assist in early detection of at-risk students [30], [38] facilitate targeted interventions, and ultimately support the CQI cycle. This

transformation has raised new research questions about how learning data is captured, represented, and analysed, particularly with respect to Malaysia's push for industry-ready, data-informed educational strategies [35], [36].

This chapter presents a comprehensive review of the relevant literature that forms the theoretical and empirical basis of the present study. It critically examines key challenges in the student performance prediction landscape across four interrelated areas:

i) Limited use of real-time academic data in current models
ii) Underutilization of adaptive oversampling techniques such as ADASYN in addressing class imbalance
iii) The fragmented integration of multiple academic activities within predictive frameworks
iv) Lack of systematic hyperparameter optimization in neural network-based models like the MLP. By exploring these gaps, the chapter establishes the rationale for the proposed solution, which is a robust, feature-rich, ADASYN-enhanced, and optimizer-tuned MLP model intended towards improving predictive precision and early intervention in Malaysia's higher education context.

2.3.1 Lack of Real-Time Data in Traditional Models

The challenge of incorporating real-time data into student performance prediction models is a significant concern in the field of educational data mining. Traditional models predominantly rely on historical data regarding prior academic achievement, which limits their ability to account for the dynamic aspects of student behaviour and performance. In contrast, real-time data such as ongoing attendance, quiz scores, and engagement with course materials. These data offer an up-to-date insight into a student's academic environment, allowing for more accurate predictions.

Current literature suggests that many existing predictive models utilize data from a single period or primarily focus on historical academic records. For [47], [48] discusses the limitations of traditional models that often neglect immediate academic activity data such as quizzes and attendance. These models frequently focus solely on cumulative performance data, which may overlook immediate variables that influence

current academic success. This gap in the integration of real-time data is critical, as it could lead to missed opportunities for intervention and support for students at risk of underperforming.

The limitations of traditional models are also discussed in the context of their reliance on past performance data. Aluko et al. in [49] argue that prior achievement is a significant predictor yet does not account for the fluctuating nature of student performance, which can be better captured through real-time updates as shown in works by In exploring various prediction techniques, Baashar et al. in [49] identified that adaptive systems such as ANNs enhance predictive accuracy but often still do not utilize real-time data Additionally, research by Fenollar et al. in [50] indicates that models based strictly on regression techniques may overlook the complexity of student behaviours over time, including factors like attendance and engagement levels

Recent studies indicate that the incorporation of week-by-week data can markedly improve the methodological approaches to predicting student performance. For example, Kurniawan and Halim in [51] discuss the efficacy of data mining combined with up-to-date educational data to forecast academic outcomes more effectively, highlighting those immediate academic activities, such as test results and lecture attendance, can serve as critical indicators for student success. Furthermore, Xiao-Yong et al. [52] emphasize that applying deep neural networks on multi-source campus data can significantly enhance prediction models by providing actionable insights that can foster student support initiatives.

In essence, while existing performance prediction models provide foundational value by utilizing historical data, their limitations are clear. The absence of real-time academic activity and engagement data significantly constrains their effectiveness. Future research and methodologies must pivot toward the integration of these dynamic data points to improve the accuracy and responsiveness of student performance forecasts.

Current research into student performance prediction has increasingly recognized that relying solely on historical and static academic data limits model responsiveness and accuracy. Traditional models that predominantly utilize factors such as prior GPA, demographic information, and entrance scores often fail to capture the dynamic nature of student learning behaviours and engagement throughout a semester in [8], [53], [54]. Instead, recent studies emphasize the importance of integrating real-time academic data, such as lecture attendance, quiz participation, interactive classroom

activities, and online discussion patterns are used to develop timely intervention strategies that can identify students at risk before their performance declines significantly [35], [36].

Study in [53] provides evidence that incorporating near real-time data into predictive models can significantly improve performance by reflecting recent trends and behaviours. Praneeth et al in [54] illustrates how smart attendance systems and behavioural analysis integrated within LMS enhance the monitoring of student engagement, enabling institutions to track academic behaviour as it unfolds. Such integration helps bridge the gap between static historical records and the evolving in-semester academic engagement, allowing for prompt educational decision-making.

Further contributions come from Hung et al [55], who developed deep learning approaches to analyse K-12 online student behaviours and discussion board interactions. Their work highlights that real-time or near real-time data logs exceeding 14 million in some cases, are instrumental in accurately identifying at-risk students early in the academic term, achieving high precision in their predictive outcomes. Similarly, Sun et al in [56] employ time series analysis on MOOC data to dynamically predict learning behaviour characteristics, suggesting that temporal modelling can reveal shifts in student engagement that static models are likely to miss.

Additionally, Albert et al in [57] illustrate the benefits of using multisource information, including real-time classroom behaviour data, to classify and monitor student engagement. Their research supports the development of effective pedagogical interventions based on continuously integrated activity data from multiple modalities. In parallel, Charitopoulos et al. in [58] reveal that blending e-learning data with hands-on laboratory instruction enables early predictions of student performance using data collected during the term. This approach supports continuous model adjustments as new academic data becomes available.

Moreover, the authors in [59] presents a deep learning-based model that focuses on sequence pattern recognition and time-series analytics for predicting dynamic student learning behaviours. While some studies, such as in [60], focus on integrating real-time affective data such as automatically detected facial expressions, their findings underscore the importance of real-time, multimodal data integration to capture the full spectrum of student behaviour and engagement.

Collectively, the literature reviewed illustrates that the absence of real-time academic data in traditional student performance prediction models represents a critical

shortcoming. Integrating dynamic data sources via approaches like real-time attendance tracking, online discussion analysis, and time-series modelling not only enhances the accuracy of predictive models but also facilitates timely interventions effectively supporting students at risk. These advances pave the way for educational models that are both data-driven and responsive to the evolving nature of student behaviours throughout the academic term.

2.3.2 Class Imbalance and Limited Use of ADASYN

The issue of class imbalance is pervasive in student performance datasets, where one class, often representing high-performing or average students significantly outnumbers the minority class which typically representing low-performing or at-risk students. This imbalance alters the distribution of training data and leads to biased models, with predictive power for the minority class severely compromised [36], [61], [62]. Many studies in educational data mining have reported that class distribution skews frequently yield models that are overly optimized for the majority class while underestimating the risk factors associated with the minority class, ultimately resulting in lower recall and sensitivity for identifying at-risk students [36], [63], [64].

Empirically, the negative impacts of class imbalance manifest as reduced overall accuracy, poor minority prediction performance, overfitting to the dominant class, and misleading performance metrics that may obscure the true predictive ability for the minority group [62], [64]. In response to these issues, various resampling techniques have been implemented to rebalance datasets. Among these, the SMOTE and simple random oversampling are some of the most frequently applied in educational prediction models, serving as controls against biased learning [65], [66] However, despite its adaptive nature, few studies in the educational domain have incorporated ADASYN [36], [67], [68], suggesting that this method remains underexplored relative to other oversampling strategies.

ADASYN provides advantages over SMOTE by generating synthetic samples adaptively; it focuses on minority class instances that are more difficult to learn, thus creating a more informative boundary between classes [69], [70]. In contrast to SMOTE, which generates synthetic samples uniformly, ADASYN adjusts the generation rate based on the learning difficulty, emphasizing regions with higher misclassification risk [69]. This adaptive mechanism has been shown in studies outside

the education sector, in areas such as medical diagnosis and text classification to lead to better F1-scores and improved sensitivity for the minority class [71], [72], [73] . Moreover, empirical results from these non-educational settings indicate that ADASYN can significantly enhance the prediction of underrepresented classes compared to traditional oversampling techniques.

Furthermore, there are instances where ADASYN's performance has been validated statistically. For example, in some studies, statistical tests such as the KS test have been employed to ascertain the improved conformity of the synthetic minority distribution to the real minority distribution after applying ADASYN; however, such rigorous validations are rarely reported specifically in studies focusing on academic performance[70], [74]. Still, its application in education-specific predictive models remains limited, representing a gap that current research needs to address.

The limitations of existing educational models that have not adopted advanced oversampling techniques are evident in their inability to capture the subtleties of minority class behaviour. Models that fail to incorporate methods such as ADASYN frequently suffer from overfitting on the majority class and diminished performance on minority detection, ultimately undermining the potential benefits of early interventions for at-risk students [61], [64], [75]. While class imbalance is a recurrent challenge in student performance datasets, the underutilization of adaptive resampling methods like ADASYN in education emphasizes the need for further empirical validation and methodological advancement to enhance minority prediction accuracy.

Recent research has demonstrated that class imbalance is a pervasive issue in student performance prediction models. In many educational datasets such as those used for dropout or graduation prediction, the number of high-performing or non-dropout students often far exceeds the number of at-risk or low-performing students [76], [77]. This skew in class distribution can lead to models that are biased toward predicting the majority class, thereby reducing the classifier's ability to accurately identify the minority class [78]. The negative impacts include decreased recall for at-risk students, misleading accuracy metrics, and an overall diminished capacity to support timely educational interventions [76], [77].

To mitigate these issues, various resampling techniques have been explored. Traditionally, methods such as random oversampling and the SMOTE have been applied to rebalance educational data [78], [79]. However, while SMOTE effectively generates synthetic minority samples by interpolating between existing instances, it

does not differentiate between the varying difficulty levels of minority instances. In contrast, ADASYN algorithm adaptively focuses on generating samples for those minority instances [79] that are hard to learn, thereby addressing the imbalance more intelligently [69]. Despite its advantageous design, ADASYN remains underutilized in education; only a few studies have applied its approach directly to the prediction of academic outcomes [80]. This underutilization represents a critical research gap, as empirical evidence from other fields suggests that ADASYN can substantially improve minority class prediction performance [69], [80].

Studies outside the educational domain have demonstrated that ADASYN not only enhances overall classification metrics such as F1-score and recall for the minority class but also improves model robustness by minimizing overfitting [81] For instance, in the context of perinatal brain injury prediction, ADASYN's focus on challenging cases contributed to achieving higher predictive performance compared to uniform oversampling methods . Moreover, comparative investigations in other domains have validated ADASYN's efficacy using statistical tests, indicating that the distribution of synthetic samples more closely resembles that of the real minority data [82]. Such findings hint at the potential benefits of applying ADASYN to educational datasets, where a similar imbalance occurs between high-performing and at-risk students.

In the educational context, the limited use of ADASYN may be partly due to the historical reliance on more conventional resampling methods such as SMOTE and random oversampling [78], [79]. Reviews in broader machine learning literature emphasize that while these methods have achieved moderate success, their inability to focus on hard-to-learn minority cases presents opportunities for improvement [81]. Thus, leveraging ADASYN in student performance prediction models could lead to enhanced classification fairness and accuracy by generating synthetic samples that reflect the nuanced difficulties encountered by at-risk students. This, in turn, would support more targeted and timely interventions, a crucial goal in educational data mining.

In summary, class imbalance is a common and critical limitation in educational prediction models that adversely impacts minority class prediction. Although techniques like random oversampling and SMOTE are widely used, ADASYN offers a promising alternative by adaptively focusing on challenging minority samples. Despite its proven advantages in other domains [69], [82], ADASYN has not been sufficiently explored in education. Addressing this gap through systematic validation and

methodological studies could significantly improve the predictive performance of educational models and enable more effective interventions for at-risk students.

2.3.3 Integrating Multiple Academic Activities in Prediction

The integration of multiple academic activities into predictive models for student performance represents a significant focus within the educational data mining community. This area of research addresses several key issues, including the number of features employed, the combination of diverse learning activities, and the methodological challenges associated with neural network architectures such as the MLP.

Many current prediction models employ a moderate number of features, typically ranging from a handful to several dozen dimensions. In numerous practical academic classifiers, researchers often use between 10 and 20 features derived from historical performance and engagement metrics [83], [84]. For instance, Sun et al in [55] demonstrate that multi-feature fusion methods, which integrate diverse inputs from academic records and behavioural metrics, can effectively capture complex interactions even when the total number of features is limited. Complementarily, Arunkumar in [85] emphasizes that combining multiple parametric inputs, which may include features such as quiz scores, attendance, and tutorial attempts, remains effective, although the exact number of features is context-dependent and rarely standardized.

Regarding the integration of multiple learning activities, several studies have verified that diverse academic activities, such as quizzes, lecture notes, and practice attempts are typically incorporated into unified models [86], [87]. These studies reveal that models that integrate activities from different domains are more likely to capture the multifaceted nature of student behaviours, thus improving predictive performance. Moreover, evidence suggests that a higher number of well-engineered features often correlates with improved classification precision; however, this relationship is nuanced and depends on the quality of feature selection and the risk of overfitting inherent in high-dimensional data spaces [88], [89]. In some cases, a model with an extensive feature set may achieve superior performance if supported by robust data fusion techniques that mitigate redundancy and noise [90].

MLP architectures are frequently chosen in educational predictive modelling due to their flexibility in handling non-linear interactions among features. Studies

indicate that MLPs are often compared with other machine learning algorithms. such as logistic regression and decision trees to evaluate predictive performance [83], [87]. In these comparative analyses, MLPs typically outperform traditional models when the input data incorporates detailed, time-based academic activities. However, this advancement comes with methodological challenges: integrating heterogeneous weekly activities into an MLP requires complex pre-processing steps, normalization, and advanced feature engineering to reduce dimensionality and avoid overfitting [88], [89].

Feature engineering and data fusion techniques are now considered essential for creating accurate educational predictive models. Researchers have highlighted that fusing data from various sources including time-series data from weekly quizzes, notes, attendance, and tutorial attempts to facilitates a more comprehensive representation of student performance drivers [91]. Automated feature engineering, as discussed by Reddy et al. in [88] offers promising routes to streamline this process while enhancing predictive accuracy. Furthermore, benchmarking studies have indicated that MLP-based approaches are often assessed against conventional classifiers, with the inclusion of diverse academic features generally leading to noticeable improvements in classification precision [87].

Recent research has highlighted that many existing student performance prediction models rely on oversimplified measures of academic engagement. Traditional frameworks often utilize aggregate metrics, such as total class attendance or overall test scores, which fail to capture the rich, multi-dimensional nature of student learning behaviours [92], [93]. For example, while studies like [93] underscore the importance of factors such as regular attendance, effective study methods, and timely exam preparation for academic success, they often do not integrate these individually nuanced measures into a unified predictive framework. Instead, such models tend to collapse complex academic activities into singular, aggregated variables, thereby losing the temporal and contextual nuances associated with weekly academic behaviours.

This reductionist approach overlooks significant indicators that can be derived from more granular academic activity data. Diverse aspects of student engagement including pre-lecture preparation, note-taking patterns, tutorial and exercise attempts, and even the timing of access to learning materials, can offer nuanced insights into how students interact with course content over a semester. The study in [94], [95] examine the utility of note-taking practices for academic performance, while the study in [95] show that access to online lecture notes correlates positively with better exam results.

However, when these data points are treated in isolation or merely summed into a total attendance count, the predictive potential of a model is significantly diminished. The challenge, therefore, lies in effectively fusing these heterogeneous signals using advanced data integration and feature engineering techniques that can handle high-dimensional and temporally distributed data.

Innovative approaches in recent studies have started to explore more structured integrations of multi-modal academic activities. For instance, Dwivedi et al. in [96] emphasize the dynamic role of predictive analytics in personalizing education by adapting learning paths based on a diverse set of engagement indicators. Similarly, Cruz et al.in [97] leverage LMS logs to capture varied aspects of student activities such as timely submissions, completion rates, and interaction frequencies, which collectively provide a richer depiction of academic engagement. These studies illustrate that integrating data from multiple sources can yield a more holistic picture of the student's journey, ultimately enhancing predictive accuracy for early identification of at-risk students.

Additional work by Sadrani [98] employs deep reinforcement learning combined with CNN-based engagement recognition to personalize learning, demonstrating that advanced algorithms are capable of processing diverse academic signals in a unified framework. Nevertheless, despite these promising approaches, many academic prediction models still struggle with the effective fusion of multi-dimensional data due to challenges in data standardization and the lack of comprehensive integration techniques in [99] further argue that holistic models developed with robust data mining techniques are necessary to uncover hidden patterns that single-dimension models typically miss. The integration of multiple academic activities not only has the potential to improve classification performance but also promotes timely interventions based on a deeper understanding of students' ongoing academic processes.

In summary, while traditional models reduce academic engagement to flat measures, the inclusion of multiple, temporally distributed academic features can provide a more accurate and actionable picture of student performance. Current research indicates that the ineffective integration of diverse academic activities remains a critical gap. Moving forward, advanced techniques in feature fusion and holistic analytics such as those explored in are necessary to develop predictive models that reflect the complete academic journey of students throughout the semester.

2.3.4 Hyperparameter Optimization Challenges in MLP

The performance of MLP classifiers is highly sensitive to hyperparameter configuration, especially in educational datasets that are often small, imbalanced, and noisy [100]. Hyperparameters such as the number of hidden neurons, optimizer type, learning rate, activation functions, and initialization settings directly influence the model's learning capacity, convergence behaviour, and generalization ability [101]. Despite the increasing adoption of MLPs for academic performance prediction, hyperparameter tuning remains an underexplored area, with many studies relying on default settings or limited single-run configurations. This gap poses a significant risk to model stability, reproducibility, and fairness in high-stakes decision-making environments like student risk prediction.

The choice of optimizer has emerged as a critical determinant of convergence speed and classification accuracy [102]. Traditional optimizers such as SGD with Momentum show inconsistent results across different initialization seeds, while adaptive optimizers like AdamW and Nadam demonstrate more efficient and stable learning trajectories [103]. Empirical findings from this study reveal that AdamW and Nadam consistently achieved convergence in fewer than 100 epochs, whereas Adagrad often failed to converge within the upper limit of 1,000 epochs [104]. This inefficiency, combined with erratic F1-scores and high variance across runs, underscores the unsuitability of Adagrad for modelling nuanced educational data. Moreover, AmsGrad and Nadam were found to maintain high precision and recall across multiple seed values, making them more robust for deployment in real-world academic systems [105].

Another key challenge lies in configuring the MLP architecture, particularly in determining the appropriate number of hidden neurons. Educational datasets require a delicate balance between model complexity and generalization [106]. Too few neurons result in underfitting, limiting the model's capacity to learn non-linear patterns, while too many neurons increase the risk of overfitting [107]. In this study, architectures with 4 to 7 neurons consistently outperformed smaller or larger configurations across all optimizers, achieving strong F1-scores in both validation and testing phases.

Optimizing MLPs for academic performance prediction requires more than selecting a popular optimizer or architecture. It demands rigorous multi-dimensional tuning that considers convergence behaviour, initialization sensitivity, and architectural complexity. The absence of standardized protocols for such tuning in the educational

data mining literature represents a critical research gap. This study addresses that gap by providing empirical benchmarks for optimizer selection and architectural design, contributing to the development of scalable and dependable predictive models for student support systems.

The performance of student prediction models that utilize MLP architectures is highly sensitive to the configuration of hyperparameters. Key hyperparameters include the learning rate, the number and structure of hidden layers, the number of neurons per layer, the choice of activation functions, and the selection of the optimization algorithm. These parameters dictate the MLP's ability to model non-linear relationships, particularly in educational datasets that are known for their irregularities, class imbalance, and high dimensionality as mentioned in Song et al. [108].

Academic datasets, especially those derived from learning management systems, tend to contain noise and heterogeneous features. In such contexts, default hyperparameter settings often result in suboptimal learning and convergence problems. For instance, an inappropriately high learning rate may cause the model to overshoot local minima during optimization, whereas a low learning rate may prolong convergence or trap the model in poor-quality solutions. Similarly, the depth and width of hidden layers require careful calibration: too shallow a network may lack the capacity to capture complex patterns, while an excessively deep network may overfit the noisy academic data [109]. These issues underscore the need for structured hyperparameter tuning methods.

Recent studies have applied various search strategies, including grid search, random search, and more advanced metaheuristic techniques in [108] proposed an MLP model for student achievement prediction that integrates an Elastic Grey Wolf Optimization algorithm, demonstrating that metaheuristics can effectively navigate the hyperparameter space and improve predictive performance. Research by Choi et al. in [110] illustrates that when carefully tuned, adaptive optimizers such as Adam and more classical methods like can yield significant performance improvements in deep learning tasks.

Moreover, Verma et al. [111] provide evidence that the selection of optimizers substantially influences the MLP's performance. Their benchmarking of popular optimizers including Adam and SGD indicates that small improvements in hyperparameter configurations can lead to marked increases in prediction accuracy and convergence stability. Despite these advances, systematic comparisons specific to

educational prediction tasks remain limited, highlighting an ongoing gap in the literature. This gap suggests that further work is needed to develop standardized frameworks for hyperparameter tuning and optimizer benchmarking in MLP models tailored for the idiosyncrasies of academic data.

In summary, suboptimal hyperparameter optimization in MLP models is a critical issue in student performance prediction. The complexities inherent in academic datasets demand rigorous tuning of hyperparameters to achieve robust and generalizable models. Recent work employing metaheuristic search strategies, as well as empirical comparisons of optimization algorithms, underscores the potential benefits of structured hyperparameter optimization. Yet the relatively sparse application of these advanced tuning techniques in the educational domain indicates that further research is essential to fully realize the potential of MLP models for timely and accurate educational prediction.

2.3.5 Research Gap

Despite significant advancements in educational data mining and machine learning, the development of accurate and timely student performance prediction models remains a complex and evolving challenge. A critical examination of existing literature reveals several limitations that collectively highlight the need for an improved, holistic approach to academic performance classification.

Firstly, traditional models in student performance prediction have heavily relied on historical and static data, such as previous GPA, demographic information, and entrance test scores. While useful for establishing general trends, these models lack sensitivity to real-time academic engagement. As shown by [53], [54], [56], the omission of temporally dynamic features such as weekly attendance, quizzes, assignments, and participation logs impairs the models' ability to detect students at risk early enough for meaningful intervention. Although recent studies underscore the importance of incorporating in-semester activity data, there remains a scarcity of models that systematically integrate and utilize these features for real-time academic forecasting.

Secondly, the issue of class imbalance, wherein datasets contain disproportionately fewer at-risk or failing students than high-performing ones remains largely unaddressed in many educational applications. Most models either ignore this

imbalance or use basic techniques like random oversampling or SMOTE, which have limitations in handling complex minority class distributions. Although ADASYN has been shown to generate more informative and targeted synthetic samples in other domains [69], [82], its application in the educational context is still limited. This represents a critical gap, especially considering the need for equitable and unbiased predictive performance across all student groups.

Thirdly, while some studies recognize the richness of student activity data, most existing models continue to aggregate these multi-dimensional academic activities into oversimplified metrics, such as total login frequency or overall attendance percentage. This reductionist approach fails to reflect the nuanced and temporal nature of academic engagement. As highlighted by [96], [97], the integration of diverse academic features ranging from pre-lecture preparation to tutorial engagement can significantly enhance the predictive capacity of machine learning models. However, few studies have attempted to construct models that systematically incorporate and structure such diverse academic behaviours over time.

Lastly, even when advanced models such as MLP are employed, the effectiveness of these models often suffers due to suboptimal hyperparameter configurations. Key parameters like learning rate, the number of hidden neurons, activation functions, and optimizer choice play a significant role in determining model accuracy and generalizability, especially in datasets with high dimensionality and noise. While metaheuristic methods and adaptive optimizers such as Adam and Elastic Grey Wolf Optimizer have shown promising results in other domains, [108] their application in educational settings remains underexplored and lacks standardization.

In summary, the literature reveals a multidimensional gap: (1) the underutilization of real-time academic activity features in predictive modelling, (2) inadequate handling of class imbalance, particularly through advanced methods like ADASYN, (3) ineffective integration of diverse academic signals into structured MLP models, and (4) insufficient hyperparameter optimization frameworks tailored to educational data. Addressing these gaps is vital for the development of more accurate, timely, and equitable student performance classifiers. Therefore, this research aims to construct a feature-rich, ADASYN-enhanced, and optimizer-tuned MLP model that supports early identification of at-risk students and improves academic decision-making in higher education environments.

Despite significant progress in educational data mining, current student performance prediction models still exhibit key limitations that hinder their effectiveness in early intervention. First, most models rely on static, historical data such as prior GPA and demographic attributes, failing to capture real-time academic behaviour that unfold throughout the semester. Second, class imbalance remains a persistent issue, with underrepresented at-risk students often misclassified due to insufficient use of adaptive resampling techniques like ADASYN. Third, academic engagement is frequently oversimplified into aggregate variables, overlooking the predictive power of granular, multi-modal activity features such as tutorial attempts, lecture note access, and exercise engagement. Finally, while Multilayer Perceptron (MLP) models have demonstrated strong potential, their performance is highly sensitive to hyperparameter and optimizer configurations, yet systematic optimization strategies are rarely applied in education-specific contexts. These gaps highlight the need for an integrated modelling framework that combines real-time data streams, balanced learning strategies, rich activity-level features, and robust MLP tuning to deliver accurate, fair, and timely academic performance predictions.

This table compares recent state-of-the-art approaches with the proposed methodology across four key dimensions: synthetic data/class imbalance handling, feature selection methods, optimization algorithms, and real-time data integration.

Table 2.1
Performance Metrics Comparison: Proposed Model vs. State-of-the-Art

Author & Year	**Synthetic Data Method**	**Feature Selection**	**Optimizer**	**Real-Time Data**	**Model Type**	**Key Performance**
Song et al. (2023) [107]	SMOTE	Filter-based	Elastic Grey Wolf	No	MLP	Acc: 87.4%
Sun et al. (2022) [55]	None	Multi-feature fusion	SGD	Partial (MOOC)	Time-series	Acc: 84.3%
Hung et al. (2023) [54]	Random Oversampling	Not specified	Adam	Yes (K-12 logs)	Deep Learning	F1: 91.5%
Albert et al. (2023) [56]	None	Multisource	Not specified	Yes (classroom)	Ensemble	Acc: 86.7%
Tariq et al. (2023) [127]	SMOTE	Wrapper-based	Adam	Limited	ANN	Acc: 85.9%

Author & Year	Synthetic Data Method	Feature Selection	Optimizer	Real-Time Data	Model Type	Key Performance
Verma et al. (2022) [110]	None	Not specified	Adam, SGD	No	MLP	Acc: 83.2%
Choi et al. (2023) [109]	None	Embedded	Adam, RMSprop	Partial	Deep NN	F1: 88.6%
Dwivedi et al. (2023) [95]	Random Under sampling	Multi-modal	Not specified	Yes (LMS)	Predictive Analytics	Acc: 82.1%
Cruz et al. (2024) [96]	None	LMS-based	Not specified	Yes (LMS logs)	ML Ensemble	Acc: 87.8%
Proposed Work (2025)	ADASYN with KS validation	MI, RFE, RFI, XGBoost, L1 -Lasso	AdamW, AdaGrad, AmsGrad, Nadam, SGD With Momentum (optimized)	Yes (daily activities)	Optimized MLP	F1: 95.7%

2.4 Literature Review for Methodology

This section presents a focused review of methodological approaches employed in prior studies on student academic performance prediction using machine learning and educational data mining techniques. The objective is to establish a clear theoretical and empirical foundation for the modelling pipeline adopted in this study, encompassing predictive modelling principles, data collection procedures, preprocessing and transformation strategies, class imbalance handling, synthetic data validation, feature selection techniques, intelligent modelling frameworks, optimization algorithms, and performance evaluation metrics. By synthesizing established methodologies and recent advances reported in the literature, this review justifies the methodological decisions undertaken and situates the proposed framework within the broader domain of data-driven educational analytics.

The review begins with predictive modelling in education (Section 2.4.1), highlighting the application of machine learning techniques, particularly neural network-based approaches, for early identification of at-risk students and the development of targeted academic interventions. Data collection methodologies

(Section 2.4.2) and the types of educational data employed (Section 2.4.3), including historical academic records, demographic information, and fine-grained engagement indicators derived from Learning Management Systems, are subsequently examined. These are followed by a discussion on data preprocessing and transformation techniques (Section 2.4.4), which are essential to ensure numerical stability, consistency, and reliability of model inputs.

Issues related to class imbalance in educational datasets (Section 2.4.5) are then addressed through synthetic data generation strategies, with particular emphasis on the Adaptive Synthetic Sampling (ADASYN) method (Section 2.4.6) and its validation using statistical hypothesis testing and visual distributional analysis (Section 2.4.7). The literature further explores intelligent modelling techniques with specific focus on the Multilayer Perceptron (Section 2.4.8), followed by a comparative review of feature selection methods spanning filter, wrapper, and embedded approaches (Section 2.4.9). Model validation and performance measurement strategies (Section 2.4.10) are then discussed in the context of imbalanced classification, highlighting the role of precision, recall, and F1-score. Finally, optimization algorithms for neural networks (Section 2.4.11) are examined, emphasizing their influence on convergence speed, stability, and generalization performance. Collectively, these methodological components form a coherent and structured foundation for the experimental design and modelling decisions presented in the subsequent chapters of this thesis.

2.4.1 Predictive Modelling in Education

Predictive modelling in education begins with the comprehensive collection and preprocessing of student data from diverse sources such as academic records, LMS logs, demographic databases, and real-time engagement activities. These heterogeneous data sources are first cleaned, normalized, and integrated to form a balanced dataset. To address common issues such as class imbalance, resampling techniques like SMOTE and ADASYN may be employed, ensuring that minority cases (e.g., low-performing or at-risk students) are adequately represented. This initial phase establishes a robust foundation necessary for extracting meaningful features and developing accurate predictive models [112], [113].

Subsequently, the methodology emphasizes sophisticated feature engineering to transform raw educational data into a structured and multidimensional input space. This

involves identifying and extracting key indicators of student engagement including attendance, quiz participation, note-taking and interaction timings to capture the dynamic nature of the academic journey. Advanced techniques and clustering algorithms help in reducing dimensionality while preserving significant patterns. These features are then integrated into predictive frameworks, where models such as decision trees, logistic regression, and artificial neural networks including Multi-Layer Perceptron, or MLP are implemented to forecast student outcomes. Recent studies have demonstrated the efficacy of these approaches when combined with metaheuristic optimization algorithms to enhance performance [108], [114].

Finally, the optimization and evaluation phase are crucial for ensuring that the predictive models generalize effectively to new data. Hyperparameter tuning, focused on parameters such as learning rate, the number and configuration of hidden layers, activation functions, and optimization algorithms use techniques like grid search, random search, and advanced metaheuristics to systematically refine model performance. Empirical comparisons and benchmarking against established statistical indices validate the optimized models, confirming improvements in convergence speed, accuracy, and robustness over baseline settings. This structured hyperparameter optimization not only ensures superior model performance but also broadens the applicability of predictive analytics in developing timely interventions and personalized education strategies.

2.4.2 Data Collection Method

Data collection is a foundational step in educational predictive modelling, as it enables the accumulation of comprehensive datasets that capture the multifaceted nature of student learning. Educational data mining (EDM) employs various techniques to extract valuable insights from raw data, and systematic data collection is central to this process. Researchers typically gather data from multiple sources including historical academic records, e-learning management systems (LMS), and demographic databases to form a repository that can later be analysed through data mining methods [115], [116]. This approach not only provides a wealth of information about students' academic histories but also enables the discovery of hidden patterns that are critical for accurate prediction and decision-making.

In practice, educational institutions collect large volumes of data using both traditional and digital methods. Collected data may include log files from LMS platforms, detailed records of student interactions, and even streaming data that reflects real-time engagement. Various data collection frameworks have been developed to manage the massive influx of information; techniques such as MapReduce and clustering algorithms are frequently used to process and integrate data from disparate sources [115], [116]. These methods help ensure that the data is complete and representative, forming the basis for effective data mining and knowledge discovery in the educational context [117]. The utilization of diverse data sources allows researchers to capture the dynamic nature of academic activities and refine predictions on student performance.

Despite the promise of comprehensive data collection, several challenges persist. Data quality and completeness are critical issues; incomplete or noisy data can lead to misleading conclusions and reduced prediction accuracy [118]. Moreover, integrating data from heterogeneous sources requires robust preprocessing techniques to ensure standardization and normalization. Recent studies emphasize the importance of adopting advanced data collection frameworks that mitigate such issues by reconciling differences in data formats and ensuring that the datasets are sufficiently large for meaningful analysis [119]. By addressing these challenges, researchers can harness the full potential of educational data mining techniques, thereby enabling richer insights and more effective strategies for improving academic outcomes.

2.4.3 Types of Data Collected

Educational predictive modelling relies on a diverse collection of data types to capture both the static and dynamic aspects of the learning process. Researchers extract individual-level information from educational systems, including historical records, demographic details, and academic performance metrics, as well as real-time user interactions within Learning Management Systems (LMS). For example, Mangaroska and Giannakos [119] illustrate how historical and current LMS activity data are leveraged to map students' learning trajectories, while Romero and Ventura in [120] emphasize that educational data mining (EDM) utilizes data from various sources, such as system logs and online interactions, to provide a comprehensive overview of student

behaviour. These data sources collectively form a rich dataset, enabling educators to perform detailed analyses and build accurate prediction models.

Beyond traditional academic records, modern educational systems capture multidimensional process data that reflect the nuanced behaviours of learners. Feng et al. [121] describe how learning process mining uses log data to uncover temporal patterns of student interactions during assessments and in digital learning environments. Such process data include clickstreams, navigation logs, and timestamps for accessing course materials, all of which contribute to understanding real-time learning behaviour. This approach enables the extraction of fine-grained signals such as engagement in forum discussions, time spent on problem-solving tasks, or pre-lecture preparation patterns that are crucial for early intervention and predictions related to performance or dropout risk. The integration of these detailed and temporally distributed data points offers a more holistic view of the learning process compared to aggregated outcomes alone.

In addition to system-generated interactions, supplementary data types further enrich predictive modelling efforts in education. Moraes et al. [122] discuss the importance of integrating administrative data, socio-economic indicators, and self-reported information to capture factors that influence academic performance. This hybrid approach, which combines objective performance metrics with subjective insights such as student satisfaction and engagement levels, has been shown to enhance the accuracy of predictive frameworks. Studies like those by [123], [124] demonstrate that supplementing quantitative LMS log data with qualitative inputs can provide complementary perspectives on learning outcomes. Collectively, the integration of varied data types from clickstreams and academic records to demographic profiles and student feedback, facilitates more robust, evidence-driven decision-making in educational settings.

2.4.4 Data Pre-processing and Transformation

In educational predictive modelling, data pre-processing is a critical step that ensures the integrity and usability of the raw data collected from diverse educational sources. This phase typically begins with the systematic removal of incomplete data, where records containing significant missing or inconsistent values are either discarded or imputed. The elimination of such incomplete data points prevents the introduction of

biases and inaccuracies in the downstream analysis, which is emphasized in methodologies that stress rigorous data analysis [125]. Moreover, the careful handling of missing values is particularly essential in educational data mining, where heterogeneous data sources such as LMS logs and academic records often contain irregularities [126] .

Data transformation techniques play an equally crucial role by converting the cleaned, raw data into feature spaces suitable for analysis. One widely adopted transformation approach is min–max normalization, which scales features to a standard range , typically between 0 and 1. This normalization ensures that variables measured on oscillating scales are rendered comparable, thereby preventing features with larger numerical ranges from disproportionately influencing model training. Such transformation has been highlighted as a fundamental step in several studies, enabling the equitable integration of diverse indicators ranging from students' access times to quiz scores and attendance rates into predictive models [127]. The uniform scaling of data through min–max normalization thus aids in achieving stable convergence and improved predictive accuracy in machine learning algorithms.

Collectively, these pre-processing and transformation strategies are indispensable to the overall performance of educational predictive models. By rigorously cleansing the data by removing incomplete entries and standardizing the diverse features through methods like min–max normalization, researchers ensure that the ensuing analyses are conducted on high-quality, reliable datasets. This systematic preparation not only enhances the robustness of the models but also facilitates the integration of multimodal data streams, thereby improving the precision of predictions regarding student outcomes [126], [127]. Consequently, these pre-processing protocols form the backbone of effective learning analytics and predictive systems in education, providing a solid foundation for subsequent exploratory and predictive tasks.

2.4.5 Class Imbalance in Educational Data

Class imbalance is a pervasive issue in educational data mining, where the distribution of classes is heavily skewed typically, one class, for instance, high-performing students overwhelmingly outnumbers the minority class, such as at-risk or low-performing students. This skew is inherent in many educational datasets, resulting in situations where critical minority classes are under-represented. As noted by Tariq et

al. [128], the imbalance between classes is common in educational prediction tasks, and this phenomenon can lead to models that are less sensitive to the patterns characterizing the minority class.

Beyond accuracy metrics, the training dynamics of MLP models are also adversely affected by imbalanced data. The scarcity of minority class examples can lead to unstable convergence during training, as the learning algorithm fails to obtain sufficient gradient signals from these under-represented cases. Hartono et al. [129] highlight that such an imbalance can generate bias in decision-making processes, whereby models preferentially learn patterns from the majority class at the expense of the minority. Additionally, the degree of overlap between classes, as explored by Cruz et al. [130], implies that factors such as this can further exacerbate the challenges in achieving balanced learning. These issues collectively underscore the complex impact of class imbalance in educational datasets on the efficacy of MLP-based predictive models.

2.4.6 Synthetic Data and ADASYN Method

Synthetic data generation is a valuable strategy for addressing class imbalance in machine learning. In educational datasets, where minority classes, such as at-risk students, are significantly underrepresented compared to high-performing ones, synthetic data can augment the pool of minority class samples, thereby mitigating bias in model training. By artificially generating new instances that mimic the characteristics of actual minority samples, synthetic data methods help provide the learning algorithm with a more balanced overview of the data distribution, which is crucial for improving predictive performance in classification tasks.

ADASYN is an advanced technique to generate synthetic data that builds upon the basic principles of synthetic minority oversampling. According to Fairul et al [36], ADASYN creates new minority class samples by adaptively assigning weights to existing samples based on their level of difficulty in classification. In essence, the algorithm focuses more on generating synthetic data for those minority class examples that are harder for the model to learn, thereby shifting the decision boundary in these critical regions. This results in a more effective distribution of synthetic samples where challenging minority instances are reinforced to improve the classifier's sensitivity toward underrepresented classes.

The use of ADASYN has demonstrated promising results across various application domains. For example, Qing et al. [131] integrated ADASYN with local outlier factor techniques to handle imbalanced tornado sample data, thus enhancing model performance by improving noise immunity. Similarly, Zuhanda et al. [132] have shown that integrating ADASYN with the Naïve Bayes classifier significantly improves classification accuracy in imbalanced anaemia detection datasets. These findings collectively underscore the potential of ADASYN to generate high-quality synthetic data that not only addresses the imbalance issue but also contributes to the overall robustness of predictive models, making it an attractive approach for handling imbalance in educational datasets.

2.4.7 Validation Techniques for Synthetic Data

Validation of synthetic data is a crucial step to ensure that the generated data accurately reflects the statistical properties and underlying distributions of real-world datasets. One common approach is to perform a Kolmogorov–Smirnov (KS) test, which is a non-parametric method used to compare the cumulative distribution functions of two data samples (real and synthetic) for significant differences. The KS test provides a quantitative measure indicating whether the two datasets originate from the same distribution, thereby helping to validate the statistical fidelity of the synthetic data Frimane & Bright [133]. In addition to the KS test, visual examination techniques such as histograms and boxplots are routinely used to compare key statistical features. Histograms facilitate the inspection of frequency distributions and reveal any discrepancies in the shape or skewness of the data, while boxplots effectively highlight differences in medians, variation, and potential outliers, offering a clear, concise visual representation of data dispersion and central tendency [134].

Beyond these primary techniques, several complementary methods have been developed to further validate synthetic datasets. Quantile-Quantile (Q-Q) plots, for example, graphically assess whether the quantiles of the synthetic data match those of the real data, and they are particularly useful for detecting deviations in the tails of the distributions. Other advanced methods include clustering consistency checks and principal component analysis (PCA) to verify that the synthetic data preserve the multidimensional variability and correlation structure exhibited by the original dataset [135], [136]. These techniques, used in concert, offer a comprehensive evaluation

framework that enhances confidence in the synthetic data's usability for downstream modelling tasks.

Incorporating multiple validation techniques is essential for establishing the credibility of synthetic data used in predictive models. Empirical studies, such as those by Brandt and Dasgupta [137], have demonstrated that a combination of statistical hypothesis testing via the KS test, graphical and multivariate techniques provides a robust validation framework. These methods ensure that synthetic data maintain high utility and reliability by closely mirroring the statistical characteristics of the real data. As a result, researchers and practitioners can confidently employ synthetic data to mitigate issues like class imbalance while preserving model performance and interpretability across a range of domains.

2.4.8 Intelligent Technique and MLP

Intelligent techniques, particularly Artificial Neural Networks (ANNs) and their subclass the Multilayer Perceptron (MLP), have gained attention for solving complex, non-linear prediction and classification tasks [138]. MLP models are adept at capturing intricate interdependencies within data due to their layered architecture and nonlinear activation functions, making them suitable for applications where traditional linear models fall short in [139] and [140]. Compared to simpler statistical or heuristic methods, MLP-based approaches offer the flexibility to model a wide range of phenomena by learning hierarchical representations.

Numerous studies highlight the performance of ANN and MLP methods compared to other intelligent techniques. For instance, Hamed et al. in [139] demonstrated how integrating blind source separation with MLP enhanced prediction accuracy in the stock market, underscoring the model’s robustness in handling noisy, complex datasets. Similarly, research by Jafari-Marandi et al. [140] confirms the value of ANN systems in classification tasks, particularly despite interpretability challenges. Moreover, Gupta and Gupta [141] reported high accuracy in diagnosing malignant mesothelioma using ANN models, while Lopes et al. [142] compared conventional ANN with deep learning methods for hydropower generation prediction, emphasizing the advantages of MLP architectures in managing diverse data.

Compared to other machine learning techniques such as support vector machines, fuzzy logic systems, or simpler regression models, MLPs and ANNs are often

shown to deliver competitive or superior performance due to their capability to learn non-linear mappings. Mohammed et al. [143] illustrated that optimizing parameters allows MLP models to outperform alternative approaches in predicting complex phenomena like monthly evaporation patterns. These findings in [35], [36]suggest that the adaptability and learning capacity of MLP-based intelligent systems make them valuable not only in fields such as finance, medicine, and energy prediction, but also position them as promising tools for educational predictive analytics where capturing the dynamic nuances of student behaviour is crucial.

2.4.9 Feature Selection Method

Feature selection plays a vital role in predictive modelling by reducing dimensionality, enhancing model interpretability, and improving generalization performance [144]. In the context of educational data mining, where datasets often involve numerous behavioural, academic, and engagement-related variables, selecting the most relevant features is essential to prevent overfitting and to ensure that the model captures meaningful patterns. Effective feature selection not only improves model accuracy but also reduces computational cost and supports more interpretable decision-making in academic environments [145].

This study employed five distinct feature selection techniques, each representing a different category in the feature selection taxonomy: filter-based, wrapper-based, and embedded methods. These techniques were selected to offer both statistical and model-based perspectives on feature relevance, ensuring a comprehensive evaluation of the input variables used for academic performance prediction.

Among the various feature selection techniques applied in this study, Mutual Information (MI) as a filter-based approach that quantifies the amount of information shared between a feature and the target variable [146]. By capturing both linear and non-linear dependencies, MI is especially valuable in educational datasets where relationships between variables may be subtle or non-obvious. Features with higher mutual information scores are interpreted as being more informative for classification, making MI an effective initial screening tool [147].

To complement the statistical nature of filter methods, a wrapper-based technique Recursive Feature Elimination (RFE) was also employed. RFE systematically eliminates the least important features based on model performance, retraining at each

step to identify an optimal subset [148]. For consistency, MLP served as the base model during elimination, ensuring that the selected features align with the final classifier. This method proved particularly effective in reducing the original set of 12 features to a refined subset of six or 50% reduction from the original number of features while preserving generalization capabilities [35].

In addition to filter and wrapper strategies, embedded methods based on ensemble models provided further insight into feature importance. Random Forest Importance evaluates how much each feature contributes to reducing node impurity across decision trees [149], while XGBoost Importance offers both gain- and coverage-based metrics derived from its boosting framework [150]. These approaches benefit from being model-aware, allowing them to uncover more complex, non-linear interactions that may be overlooked by purely statistical measures [151].

L1 regularization, also known as Lasso (Least Absolute Shrinkage and Selection Operator), is an embedded feature selection method that introduces a penalty to shrink less important feature coefficients toward zero [152]. This results in a sparse model that retains only the most relevant predictors, making it well-suited for high-dimensional educational data with potential noise and redundancy. By eliminating irrelevant features, Lasso simplifies the model and helps prevent overfitting, which is especially beneficial when working with small datasets [153], [154]. In this study, Lasso was used to identify key academic engagement features that strongly influence prediction outcomes. Its integration supports a deeper understanding of how MLP performance changes when exposed to refined input subsets, ultimately guiding better model design and interpretation.

By integrating these five methods spanning filter, wrapper, and embedded categories this study ensured a robust and comprehensive evaluation of feature relevance. The combination not only enhanced the predictive strength of the model but also supported a more interpretable and efficient classifier suitable for academic performance analytics.

2.4.10 MLP Model Validation and Performance Measurement

For the study on MLP model validation for educational prediction, study employed classic performance metrics with careful consideration given to their relative priorities when assessing classifier behaviour. Precision is assigned the highest priority

because reducing false positives is critical in applications were misclassifying a non-at-risk student as at-risk could lead to unnecessary interventions and wasted resources [155]. In contrast, accuracy is regarded as a low-priority metric in scenarios with class imbalance, where high accuracy can be misleading if it predominantly reflects the majority class. Similarly, recall is placed at low priority since our domain can tolerate a few false negatives without substantially undermining the overall intervention strategy. The F1-score serves a moderate priority, balancing the trade-off between precision and recall, thereby providing a more holistic view of model performance on imbalanced data [156].

A detailed analysis of the MLP model indicates that careful calibration of these metrics is crucial for robust performance measurement. The evaluation leverages a confusion matrix to derive the true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN), which are subsequently used to calculate precision, recall, and F1-score [35], [36]. The model exhibits exceptionally high precision, thereby minimizing the incidence of false positive predictions. The F1-score, being the harmonic mean of precision and recall, further confirms that the proposed model maintains a balanced trade-off between these metrics, even as recall is intentionally de-emphasized due to the low cost of occasional false negatives.

Comparing the studies, the superiority of the MLP model becomes evident. For instance, the F1-score of our model, as high as 95.706%, significantly exceeds the performance measures of comparable models in other domains, such as those in medical diagnosis or cybersecurity detection [155]. Moreover, while some literature emphasizes high accuracy values, our approach underlines precision as the key metric, thereby ensuring that the predictive system is both reliable and cost-effective for educational settings. These findings demonstrate that optimizing for precision and F1-score in imbalanced classification problems yields robust performance, particularly when the consequences of false positives are critical to the decision-making process.

2.4.11 Optimization Algorithms in Neural Networks

Optimization of neural network hyperparameters including the number of training epochs, random number seed initialization, and hidden layer architecture is crucial to obtaining models with superior performance and generalization. The non-convex nature of neural network training necessitates the use of advanced optimization

algorithms that can efficiently search the hyperparameter space. For instance, Yin and Zhu [125] demonstrate that a mathematically rigorous optimization algorithm can simultaneously determine the optimal number of hidden neurons and the forms of transfer functions in Bayesian neural networks. Similarly, the performance sensitivity of MLP models to factors like epoch count and weight initialization underscores the need for systematic optimization, as different seeds and epoch configurations can lead to varying convergence behaviours.

Metaheuristic approaches have been prominent in addressing the challenges of designing optimal neural network architectures. Techniques such as GA-BP, discussed by Yanhong and Zhang [126], combine genetic algorithms with backpropagation to fine-tune the network's hidden layer structure and optimize preprocessing, improving training speed and output consistency. Furtuna et al. [127] present a trial-and-error methodology that adjusts internal parameters, such as the number of hidden layers and learning rate, to achieve near-optimal configurations. These methods highlight that heuristic and evolutionary algorithms, including Particle Swarm Optimization, as used by Wang and Guo [128], can effectively navigate the hyperparameter space to optimize factors like network depth and neuron count.

Beyond structure, the role of training epochs and random number seed initialization directly affects how well an MLP model converges to an optimal solution. Tyagi et al. [129] introduce a second-order training algorithm that dynamically prunes the network at each epoch, optimizing the number of hidden units while also regulating training duration. Controlling the random number seen during weight initialization helps mitigate variability between training runs, ensuring that the model's performance is a direct result of the optimization process rather than stochastic fluctuations. Collectively, these strategies not only optimize network architecture but also improve training stability and robustness, indicating that a careful combination of evolutionary, heuristic, and random search methods plays a pivotal role in fine-tuning neural network hyperparameters for enhanced predictive performance.

2.5 Chapter Summary

The literature on predictive modelling in education increasingly emphasizes the use of intelligent systems for early identification and intervention for at-risk students. With the advancement of learning technologies and the availability of diverse academic

data, researchers are moving beyond traditional models that rely solely on static indicators such as GPA and demographics. Recent studies highlight the critical importance of incorporating real-time academic activities including attendance, quiz scores, learning material access, and tutorial engagement to significantly improve prediction timeliness and accuracy of. Among the various intelligent techniques, neural network models, particularly the MLP, have emerged as highly effective due to their ability to learn complex, non-linear relationships inherent in educational datasets. These models are further strengthened when paired with appropriate data preprocessing, feature selection, and optimization strategies.

A persistent challenge identified in the literature is the pervasive issue of class imbalance within educational datasets. Typically, students at risk of poor academic performance constitute a minority class, leading to skewed training distributions and biased classification results. This imbalance often reduces the classifier's sensitivity to low-performing students, who are often the primary focus for targeted educational interventions. Among the techniques explored to mitigate this challenge, the Adaptive Synthetic Sampling (ADASYN) method has shown considerable promise. Although its effectiveness has been demonstrated in other domains, its application in educational prediction tasks remains unexplored, highlighting a significant methodological gap that warrants further investigation.

Lastly, the literature underscores the critical role of hyperparameter tuning and feature engineering in enhancing the robustness and reliability of predictive models. The performance of MLP classifiers heavily depends on the optimal configuration of key architectural and training parameters, such as learning rate, activation functions, hidden layer configurations, and optimizer selection. Advanced optimizers like Adam and Adm variation have been shown to improve convergence speed and stability, while feature selection techniques such as Recursive Feature Elimination (RFE), Mutual Information, and Lasso have been employed to reduce redundancy and highlight the most impactful academic attributes. Despite their demonstrated benefits, the comprehensive and systematic application of these techniques in educational contexts remains limited. Collectively, the reviewed studies emphasize the need for an integrated framework that synergistically combines real-time academic features, class balancing through ADASYN, and systematic optimization of MLP models to achieve accurate and timely prediction of student performance.

CHAPTER 3
THEORETICAL BACKGROUND

3.1 Chapter Overview

This chapter establishes the theoretical background that supports the modelling pipeline used to predict student academic performance using machine learning. Section 3.1: Data Preparation and Transformation Theory explain the role of Min–Max normalization in rescaling input features to a consistent range, improving numerical stability and supporting efficient gradient-based learning. Section 3.2: Synthetic Data Theory then introduces ADASYN, an adaptive oversampling approach that mitigates class imbalance by generating synthetic samples preferentially around harder-to-learn minority instances. Next, Section 3.3: MLP Theory and Hyperparameter Tuning presents the fundamentals of the MLP architecture used in this study, including feedforward computation, activation functions, binary classification output behaviour, and the key tuning elements that govern model capacity and generalization such as hidden neuron selection, epoch control with early stopping, and robustness across random seeds. To ensure the synthetic dataset remains statistically consistent with the original data, Section 3.4: Synthetic Data Set Verification outlines validation procedures using the Kolmogorov–Smirnov test, histogram comparison, and box plot analysis. Section 3.5: Feature Selection Theory reviews five feature selection approaches highlighting their role in improving interpretability, reducing redundancy, and minimizing overfitting. Section 3.6: Optimizer Theory discusses the five optimizers evaluated and explains how their update mechanisms influence convergence speed, training stability, and model generalization. Finally, Section 3.7: Other Intelligent Techniques introduces alternative classifiers to provide theoretical grounding for the comparative evaluation against the MLP in subsequent chapters.

3.2 Data Preparation and Transformation Theory

Data transformation is regarded as data pre-processing. This process is important to improve the accuracy and efficiency of data analysis in nearest neighbour and neural networks. Normalization is one part of data transformation and useful for

classification algorithms especially in speeding up the learning phase of neural networks. The method of normalization being used in this study is Min-Max normalization.

Min-Max Normalization is a data preprocessing technique used to rescale numerical data into a specific range, typically between 0 and 1, or sometimes -1 to 1. This transformation ensures that different features with varying scales do not disproportionately influence machine learning models [157], [158].

The Min-Max Normalization formula is :

$$X_{normalised} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$

where:

X= original value of the feature

Xmin = minimum value in the dataset for that feature

Xmax = maximum value in the dataset for that feature

Xnormalized = rescaled value in the range [0,1]

Min-Max Normalization is an important step in preprocessing for machine learning, particularly MLP models, as it ensures numerical stability and enhances model convergence. It is widely used in neural networks, deep learning, and gradient-based optimization methods to improve training efficiency.

3.3 Synthetic Data Theory

In machine learning, the quality and balance of data play a crucial role in determining the performance of predictive models. However, real-world datasets often suffer from class imbalance, where one class significantly outnumbers the other. This imbalance can lead to biased models that favour the majority class, resulting in poor generalization and low predictive accuracy for the minority class. To address this challenge, synthetic data generation techniques have been developed to artificially expand underrepresented classes while preserving the statistical characteristics of the original dataset.

Synthetic data refers to artificially generated data that mimics real-world observations while maintaining their key properties. Unlike traditional oversampling techniques that duplicate existing samples, synthetic data generation methods create new instances that introduce slight variations, improving the model's ability to generalize. These techniques are widely used in applications where collecting real-world data is costly, sensitive, or insufficient for training robust machine learning models. One of the most effective approaches for handling class imbalance is ADASYN, which dynamically generates synthetic data based on the learning difficulty of individual instances. Unlike conventional oversampling methods, ADASYN prioritizes harder-to-classify minority samples, ensuring a more adaptive and balanced dataset. By incorporating synthetic data, classification models can achieve better predictive performance, reduce bias toward majority classes, and enhance fairness in decision-making processes.

ADASYN method is an advanced oversampling technique designed to address class imbalance by generating synthetic data points for the minority class. Unlike traditional oversampling methods, ADASYN focuses on difficult-to-learn instances by adaptively adjusting the number of synthetic samples generated based on local class distribution [159].

The first step in ADASYN is to compute the imbalance ratio ***d***, which quantifies the imbalance between the majority and minority classes:

$$d = \frac{|X_{minority}|}{|X_{majority}|} \tag{3.2}$$

Where

d = degree of class imbalance (0 < d <= 1)

|Xminority| = number of samples in the minority class

|Xmajority| = number of samples in the majority class\

Therefore, as d decreases, the disparity between the majority and minority classes increases, indicating a more severe class imbalance that requires targeted resampling strategies like ADASYN to improve model performance.

ADASYN determines the total number of synthetic samples needed for the minority class based on a user-defined balancing level parameter $\beta = (0 \leq \beta \leq 1)$.

$$G = \left(\left|X_{majority}\right| - \left|X_{minority}\right|\right) x\ \beta \tag{3.3}$$

where:

G = total number of synthetic data points to generate

β = desired balance level (typically set between 0 and 1)

The default value of the balancing level parameter **β** in ADASYN is set to **0.5**, meaning the minority class is adjusted to be halfway between its original size and full balance with the majority class.

To determine which minority class samples, require more synthetic data, ADASYN calculates a difficulty score for each minority instance ix. This score, denoted as ir, represents the proportion of majority class samples among the k-nearest neighbours of ix [160], [161]. It is computed as:

$$r_i = \frac{\sum_{j=1}^{k} I(y_j = y_{majority})}{k} \tag{3.4}$$

where:

r_i = ratio of majority class samples among the k-nearest neighbours of minority instance x_i

I(·) = indicator function (1 if the condition is true, 0 otherwise)

y_j = class label of neighbour j

k = number of nearest neighbours (typically k=5)

Where $I(y_j = y_{majority})$ is an indicator function that returns 1 if the neighbour yj belongs to the majority class and 0 otherwise. A higher ri value indicates that the instance is surrounded by more majority class samples, making it harder to classify correctly. ADASYN uses these difficulty scores to prioritize synthetic data generation for instances that are most at risk of misclassification [161].

Next, the difficulty scores are normalized to determine the proportion of synthetic samples that each minority instance should receive. The normalized difficulty score $\hat{r}_i$is calculated as:

$$\hat{r_i} = \frac{r_i}{\sum_{j=1}^{|X_{minority}|} r_j} \tag{3.5}$$

Using this normalized score, the number of synthetic samples assigned to each minority instance xi is given by:

$$g_i = \hat{r_i}\, x\, G \tag{3.6}$$

where G represents the total number of synthetic samples to be generated. This ensures that more synthetic data is generated for the hardest-to-classify instances, thereby improving the model's ability to correctly identify the minority class.

Once the number of synthetic samples for each minority instance is determined, ADASYN generates the new instances using an interpolation method. For each minority sample x_i, gi new samples are created by linearly interpolating between x_i and one of its randomly selected minority-class nearest neighbours X_{zi}:

$$x_{new} = x_i + \lambda\, x\, (x_{zi} - x_i) \tag{3.7}$$

where λ is a random number drawn from a uniform distribution U(0,1). This method ensures that the synthetic samples are not simple duplicates but are distributed adaptively in the feature space, making them more effective for improving classification performance. By prioritizing the generation of synthetic instances in areas where the decision boundary is more ambiguous, ADASYN enhances model generalization and reduces bias toward the majority class, ultimately leading to improved classification outcomes.

3.4 Synthetic Data Set Verification

After generating synthetic samples using the ADASYN method to address class imbalance, it is essential to verify that the synthetic data maintains the statistical integrity and distributional properties of the original dataset. This verification process

ensures that the augmented data does not introduce distortions or biases that could negatively affect model performance or generalization. In this study, three complementary techniques were employed to validate the quality and consistency of the synthetic dataset: the Kolmogorov–Smirnov (KS) test, the histogram test, and the box plot test. Each method offers a unique perspective. KS test provides a formal statistical comparison between distributions [162], the histogram test offers a visual assessment of distributional similarity [163], [164], and the box plot test highlights central tendency, spread, and potential outliers [165], [166]. Together, these methods form a robust validation framework for confirming that the synthetic data is both representative and reliable for use in subsequent predictive modelling tasks.

3.4.1 Kolmogorov–Smirnov Test

The Kolmogorov–Smirnov (KS) test is a non-parametric statistical method used to assess the similarity between an empirical distribution and a theoretical distribution or to compare two empirical distributions. In the context of the one-sample KS test, which is most relevant in validating the fit between observed data and a known distribution, the test evaluates the maximum absolute difference between the empirical distribution function (EDF) of the sample and the cumulative distribution function (CDF) of the reference distribution.

The empirical distribution function denotes as $F_n(x)$, represents the proportion of observed data points less than or equal to a given value x. It is formally defined as:

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} I(-\infty, x)(X_i) \tag{3.8}$$

where I is the indicator function, X_i denotes individual observations, and n is the total number of observations. The EDF increases in discrete steps of $\frac{1}{n}$, forming a step function that approximates the underlying distribution of the data. The KS test statistic, commonly denoted as D_n, quantifies the largest vertical distance between the empirical distribution Fn(x) and the theoretical cumulative distribution $F(x)$. It is expressed as:

$$D_n = \sup_n |F_n(x) - F(x)| \tag{3.9}$$

Where $\sup_n$ denotes the supremum over all values of x. This statistic captures the maximum deviation between the observed data and the expected theoretical behaviour.

The hypotheses for the KS test are defined as follows: the null hypothesis states that the sample is drawn from the specified theoretical distribution, while the alternative hypothesis suggests that the sample does not follow this distribution. Under the null hypothesis and assuming that $F(x)$ is continuous, the distribution of the test statistic D_n converges to the Kolmogorov distribution as the sample size increases [167], [168]. For smaller sample sizes, exact critical values or p-values are available, whereas for larger samples, approximations or simulation-based approaches are often employed.

The KS test offers several advantages. It is distribution-free under the null hypothesis and does not require estimation of parameters, making it particularly useful for goodness-of-fit testing. Moreover, it is sensitive to discrepancies in both the location and shape of distributions. However, the KS test also has limitations. It is less powerful when parameters of the theoretical distribution are estimated from the data rather than known a priori. Additionally, the test tends to be more sensitive near the centre of the distribution than in the tails. In cases involving discrete distributions, the standard KS test may not be applicable without adjustments.

Overall, the KS test provides a robust framework for evaluating distributional assumptions, and it is frequently employed in statistical modeling and validation, such as checking the consistency of synthetic data or assessing model residuals for normality.

3.4.2 Histogram Test

The histogram test is a visual and descriptive method used to assess the distributional characteristics of a dataset by comparing its empirical distribution to a theoretical distribution. It involves dividing the range of observed data into a series of intervals, or bins, and counting the number of data points that fall within each bin. The resulting histogram serves as a graphical representation of the data's frequency distribution, allowing for an intuitive comparison with the shape, spread, and central

tendency of a theoretical probability distribution. In statistical analysis, the histogram test is often employed as a preliminary diagnostic tool to evaluate whether a sample follows a particular distribution, such as the normal distribution. By superimposing the probability density function (PDF) of the target distribution onto the histogram, one can visually assess the degree of fit between the observed data and the theoretical model. A close alignment between the histogram bars and the curve of the theoretical PDF suggests a good fit, whereas noticeable discrepancies in symmetry, skewness, or kurtosis may indicate deviation from the assumed distribution.

Mathematically, constructing a histogram begins by dividing the data range $[X_{min} - X_{max}]$ into k bins. For equal-width bins, the width (h) is defined as :

$$h = \frac{x_{max} - x_{min}}{k} \tag{3.10}$$

Each bin I spans the interval $(x_{i'}, x_{i+1})$, where $x_i = X_{min} + (i-1)h$for $i = 1,2,3, \dots, k$. The frequency $f_i$of data points within each bin is defined as:

$$f_i = \sum_{j=1}^{n} I_{[x_{i'}, x_{i+1}]}\,(x_j) \tag{3.11}$$

Where $I_{[x_{i'}, x_{i+1}]}(x_j)$is an indicator function equal to 1 if $x_j \in [x_i, x_{i+1}]$and 0 otherwise. To transform the histogram into a probability density estimate, the frequency can be normalized to yield the relative frequency or density:

$$\hat{f} = \frac{f_i}{nh} \tag{3.12}$$

3.4.3 Box Plot Test

The box plot, or box-and-whisker plot, is a graphical representation used to visualize the distribution, central tendency, and spread of a dataset [169]. It is constructed from a five-number summary, which consists of the minimum, first quartile (Q1), median (Q2), third quartile (Q3), and maximum values. These summary statistics capture the essential characteristics of a dataset and allow for rapid comparison across groups or conditions [170], [171].

Mathematically, the interquartile range (IQR) is a key component of the box plot and is defined as:

$$IQR=Q3-Q1 \tag{3.13}$$

This range represents the middle 50% of the data. The "box" in a box plot spans from Q1 to Q3, with a line indicating the median (Q2). The "whiskers" extend from the box to the smallest and largest data points that fall within an acceptable range, typically defined as:

$$\text{Lower Bound}=Q1-1.5(IQR) \tag{3.14}$$

$$\text{Upper Bound}=Q3+1.5(IQR) \tag{3.15}$$

Data points that lie outside this range are considered potential outliers and are plotted individually as separate markers. This mathematical approach enables the box plot to highlight unusual values and identify asymmetry or skewness in the data.

When applied in distributional analysis, the box plot can provide visual cues regarding the shape and spread of the data. A symmetric distribution is typically reflected in a cantered median within the box and whiskers of approximately equal length. Skewness can be inferred from the position of the median relative to the quartiles and the length of the whiskers. For example, a median closer to Q1 and a longer upper whisker indicate right skewness.

Although the box plot is not a formal hypothesis test and does not produce statistical metrics such as p-values or confidence intervals, it is highly effective in exploratory data analysis. It offers immediate insights into data variability, identifies outliers, and enables side-by-side comparison of distributions across multiple categories or groups.

One of the key strengths of the box plot lies in its robustness. It is not affected by the shape of the distribution and performs well even with skewed or non-normal data. However, its simplicity also implies certain limitations: it does not provide information about distribution modality (e.g., unimodal or bimodal) or more subtle distributional features.

In conclusion, the box plot test is a valuable visual tool in descriptive statistics and model diagnostics. By leveraging the interquartile range and outlier detection bounds, it provides a clear and interpretable summary of a dataset's distribution, making it an ideal complement to formal statistical tests during data validation and exploratory analysis phases.

3.5 Feature Selection Theory

Another important stage in machine learning and predictive modelling, particularly in educational data mining is feature selection, where identifying the most relevant features can significantly enhance model accuracy and interpretability. The goal of feature selection is to remove irrelevant, redundant, or noisy variables, ensuring that the model focuses only on the most informative predictors. By reducing the dimensionality of the dataset, feature selection not only improves computational efficiency but also minimizes the risk of overfitting, leading to better generalization on unseen data. Various statistical and machine learning-based techniques have been developed to evaluate and rank feature importance, each offering unique advantages depending on the nature of the dataset. In this thesis, a comprehensive set of feature selection methods is employed, including Mutual Information, RFE, Random Forest Importance, XGBoost Importance and L1 Lasso. These methods collectively help refine the dataset, enhance model interpretability, and improve classification performance by identifying the most significant features contributing to student academic outcomes.

3.5.1 Mutual Information

Mutual Information is a fundamental concept from information theory that quantifies the amount of information shared between two random variables. In the context of feature selection, it measures the dependency between an input feature X and the target variable Y. Unlike linear correlation, Mutual Information captures both linear and non-linear relationships, making it suitable for complex classification tasks. Mathematically, the mutual information between X and Y is defined as:

Where $p(x, y)$ is the joint probability distribution of X and Y, and $p(x)$ are the marginal probability distributions. A higher Mutual Information value indicated that known if the value of X reduces the uncertainty of Y, thus thus implying higher feature relevance.

$$\mathrm{v}I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) log \left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{3.16}$$

In feature selection, Mutual Information acts as a filter method, ranking features based on their individual mutual information with the target class. This allows the selection of features that contribute the most to reducing uncertainty in predicting the target variable. Since it does not depend on any specific classifier, MI is model-agnostic and computationally efficient, making it suitable for preprocessing high-dimensional datasets. By selecting features with high MI scores, practitioners can reduce dimensionality, avoid overfitting, and improve the interpretability and performance of machine learning models, including decision trees, support vector machines, and MLPs.

3.5.2 Recursive Feature Elimination

RFE is a wrapper-based feature selection technique designed to identify the most significant subset of features for predictive modelling by recursively removing the least important ones. The method works by training a model on the entire feature set and ranking each feature based on its importance score, which is typically derived from model-specific weights or coefficients. At each iteration, the feature with the lowest importance is eliminated, and the model is retrained on the reduced feature set. This

process continues until the desired number of features is obtained. RFE is especially beneficial for reducing redundancy, improving generalization, and optimizing performance in supervised learning tasks.

Features important is commonly determined by the magnitude of the model coefficients $|\beta_j|$. The model can be expressed as:

$$f(x) = \beta_o + \sum_{j=1}^{p} \beta_j x_j \tag{3.17}$$

Where $x_j$is the j-th input feature, $\beta_j$is corresponding weight or coefficient and $\beta_o$is the intercept term. Features associated with the smallest absolute coefficients $|\beta_j|$. are considered least informative and are eliminated first. By iteratively refining the input space in this backward elimination process, RFE enhances the model's ability to focus on the most relevant predictors.

3.5.3 Random Forest Importance

Random Forest is an ensemble learning method that constructs a large number of decision trees during training and outputs the class that is the mode of the classes (classification) or the mean prediction (regression) of the individual trees. As a byproduct of its learning process, Random Forest naturally provides estimates of feature importance by measuring how much each feature contributes to the model's predictive performance. This is achieved by evaluating how much a feature decreases the impurity across all the trees in the forest, commonly using the Gini impurity or entropy (for classification tasks). Each time a feature is used to split a node, the algorithm calculates the reduction in impurity and aggregates this value over all trees in the forest to quantify the feature's importance.

Mathematically, the importance of a feature $x_j$is computed as the todal decease in impurity Δ_i caused by splits on x_j, normalised over all trees. If a node t in a decision tree splits on feature x_j and the impurity before and after the splits is $i(t)$ and $i_L, i_R$for the left and right node respectively, then the impurity reduction is given by:

$$\Delta i(t, x_j) = i(t) - (\frac{N_L}{N_t} i_L + \frac{N_R}{N_t} i_R) \tag{3.18}$$

Where N_t, N_L, and N_R are the number of samples in the parent, left, and right nodes. The feature importance is then averaged across all trees. Features that result in large and frequent impurity reductions are deemed more important. Random Forest feature importance is robust to overfitting and can capture complex feature interactions, making it highly effective for high-dimensional datasets and non-linear problems.

3.5.4 XGBoost Importance

XGBoost is a powerful ensemble learning algorithm based on gradient-boosted decision trees. It builds trees sequentially, where each new tree attempts to correct the residual errors made by the previous ones. One of the advantages of XGBoost is its built-in ability to compute feature importance as part of the training process. Feature importance in XGBoost is derived from how often and how effectively a feature is used to split the data across all trees. This importance can be measured in several ways, weight (the number of times a feature is used to split the data), gain (the average improvement in accuracy or reduction in loss from splits using the feature), and cover (the number of samples affected by those splits). The most common metric, gain, is based on the reduction of a specified loss function after a split. Mathematically, for a split at a node t, the gain for a feature x_j id given by:

$$\text{Gain}(x_j) = \frac{1}{2}\left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}\right] - \gamma \tag{3.19}$$

Where G_L and G_R are the gradients and H_L and H_R are the Hessians second-order derivatives for the left and right child nodes λ is the regularization parameter and γ is the penalty for creating additional leaves. This formula evaluates how much a particular feature helps to reduce the objective function that is usually loss, which reflects its usefulness. XGBoost feature importance helps in identifying the most influential predictors in high-dimensional, noisy, or highly nonlinear datasets, and is particularly

valuable when model explainability and selection of compact, high-performance feature sets are desired.

3.5.5 L1 Regularization Lasso

L1 Lasso is an embedded feature selection method that introduces a penalty term to the objective function of linear models to enforce sparsity in the model coefficients. Unlike traditional regression, where all features may contribute to the output, Lasso encourages the model to reduce the weights of less important features to exactly zero. This results in a sparse model where only the most relevant features are retained, making Lasso particularly effective for feature selection in high-dimensional data and when model interpretability is crucial.

Mathematically, the Lasso objective function for a linear regression model is defined as:

$$\min_{\beta}\left\{\frac{1}{2n}\sum_{i=1}^{n}\left(y_i-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=i}^{p}\left|\beta_j\right|\right\} \tag{3.20}$$

Where n is the number of samples, p is the number of features, y_i is the observed output, x_ij is the input features, β_j is the coefficient for feature j, and lambda is the regularization parameter controlling the degree of shrinkage. As λ increases, more coefficients are driven to zero. This property allows L1 regularization to perform automatic variable selection during model training

3.6 The Multilayer Perceptron Theory and Hyperparameter Tuning

MLP is a fundamental architecture in feedforward ANNs and is widely utilized in classification tasks involving structured, tabular data. An MLP consists of three primary layers: an input layer that receives feature values, one or more hidden layers that learn representations of the data, and an output layer that produces predictions. In this study, the MLP model is developed to classify student academic performance based on 12 extracted features representing engagement patterns. Each neuron within a hidden

layer performs a linear transformation of the input followed by a non-linear activation function. The Rectified Linear Unit activation function is used in hidden layers due to its computational efficiency and ability to avoid vanishing gradient problems. The final output neuron employs the sigmoid activation function to generate a probability score for binary classification tasks.

Mathematically, the forward propagation in an MLP as follows. There is a hidden layer that computes the activation $h = f(W_{1x} + b_1)$, and the output layer generates $\hat{y} = \sigma(W_2 h + b_2)$, where $W_1$and W_2 are weight matrices, $b_1$and b_2 are the biases, f is the ReLU function and $\sigma$is the sigmoid function. The sigmoid function maps the output to a probability range between 0 and 1, making it suitable for binary classification tasks. Model training is guided by the binary cross-entropy loss function, which penalizes incorrect predictions and encourages the network to produce outputs aligned with true labels. Optimization is performed through backpropagation and gradient descent, which iteratively adjust the weights to minimize the loss.

Hyperparameter tuning is a vital component of MLP development, directly influencing the model's accuracy, generalization, and training stability. One of the primary hyperparameters examined in this study is the number of hidden neurons. A range from one to ten neurons was tested, and empirical results showed that configurations with four to seven neurons produced the most stable and accurate performance. Networks with too few neurons suffered from underfitting, failing to learn complex relationships, while those with too many neurons tended to overfit, reducing generalization to unseen data. The optimizer selection also played a crucial role in model training. Several optimizers were evaluated, including SGD with Momentum, AdamW, Nadam, Adagrad, and AmsGrad.

Another critical tuning factor was the number of epochs required for convergence. Early stopping was implemented to prevent overfitting by halting training when no further improvement was observed in validation loss. It was observed that AdamW and Nadam typically converged in fewer epochs, which is indicative of their adaptive learning rate strategies. Additionally, the model's robustness to different weight initializations was examined using multiple random seeds, including 30, 42, 7917, and 104729. Results indicated that AdamW and Nadam were robust across different seeds, while SGD exhibited greater performance variability depending on initialization. All models used ReLU as the activation function in hidden layers, and weight initialization was presumed to follow the Xavier initialization or He initialization

as in [172] to ensure numerical stability and facilitate smooth gradient propagation during training.

In summary, the MLP model in this study was developed using a theoretically sound structure and rigorously tuned hyperparameters to optimize its predictive power. The integration of well-established optimization techniques, appropriate activation functions, and careful parameter tuning contributed to the creation of a robust and generalizable classifier capable of effectively predicting student academic outcomes based on behavioural data collected from online learning environments.

3.7 Optimizer Theory

Optimizers play a critical role in the training of neural networks by guiding how model weights are updated in response to the calculated loss. Their primary objective is to minimize the loss function as efficiently and accurately as possible by adjusting the parameters of the model through iterative gradient-based methods. Different optimizers offer various strategies for controlling the direction and magnitude of these weight updates, and their selection directly affects convergence speed, model generalization, and stability. In the context of training a Multilayer Perceptron (MLP) for student academic performance classification, choosing an appropriate optimizer is crucial for achieving balanced and reliable outcomes, especially when dealing with small, imbalanced, and feature-rich educational datasets.

This study focuses on five distinct optimizers: AdamW, AdaGrad, AmsGrad, Nadam, and SGD with Momentum. These optimizers represent a mix of adaptive and non-adaptive strategies, with different mechanisms for handling learning rates, gradient smoothing, and regularization. Adaptive optimizers such as AdamW, Nadam, and AmsGrad are designed to adjust learning rates dynamically and are particularly effective for sparse or noisy data. In contrast, SGD with Momentum, a classic optimizer, provides a valuable benchmark for understanding convergence under minimal adaptivity. AdaGrad, while simple, offers insights into feature-specific learning behavior. Each optimizer's theoretical underpinnings, advantages, and limitations are examined in this section, along with their relevance and application to the problem of modeling student engagement and academic success using MLPs.

3.7.1 Adaptive Moment Estimation with Decoupled Weight Decay

AdamW is a variant of the widely used Adam optimizer, designed to address a critical flaw in how weight decay regularization is implemented in standard Adam. In classical Adam, L2 regularization is coupled with the gradient update, which inadvertently affects the adaptive learning rates and undermines the intended role of weight decay. AdamW, introduced by [173], decouples the weght decay from the gradient computation, applying it directly to the weights instead of the gradients. This separation improves generalization performance and yields better convergence, especially in deep neural networks. The AdamW update rule incorporates the standard Adam moment estimates:

$$m_i = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{3.21}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{3.22}$$

With bias correction terms $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$ and $\hat{v}_t = \frac{v_t}{1-\beta_1^t}$. The key distinction is that the parameter update includes a direct weight decay term :

$$\theta_{t+1} = \theta_t - \eta \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}} + \lambda \theta_t \right) \tag{3.23}$$

where λ is the weight decay coefficient, and η is the learning rate.

By decoupling weight decay from the gradient calculation, AdamW preserves the integrity of adaptive learning while maintaining proper penalization of large weights. This approach has been shown to result in better training stability and improved generalization compared to standard Adam, particularly in models that require strong regularization. In the context of MLP modeling for academic performance prediction, AdamW helps mitigate overfitting by effectively shrinking irrelevant weights, enabling the model to focus on the most meaningful student engagement features. As a result, AdamW has emerged as one of the top-performing optimizers in deep learning applications involving sparse or high-dimensional data.

3.7.2 Adaptive Gradient Algorithm

AdaGrad is an adaptive learning rate optimization algorithm that modifies the standard SGD by adjusting the learning rate individually for each model parameter based on historical gradient information. The core idea is to assign smaller learning rates to frequently updated parameters and larger learning rates to infrequent ones, which makes it especially effective for sparse data and high-dimensional problems. AdaGrad achieves this by accumulating the sum of squared gradients for each parameter and scaling the learning rate accordingly. The update rule is defined as:

$$\theta_{t+1} = \theta_t - \left(\frac{\eta}{\sqrt{G_t + \epsilon}} + g_t \right) \tag{3.24}$$

where η is the global learning rate, is the gradient at time step t.

The strength of AdaGrad lies in its ability to adaptively tune learning rates based on parameter-specific gradient histories, allowing for more refined convergence behaviour. However, a key limitation is the continual growth of the accumulated gradient term Gt, which causes the effective learning rate to shrink over time and can lead to premature convergence or stalled learning, particularly in non-convex or deep learning tasks. In the context of MLP models used for student academic performance prediction, AdaGrad can be beneficial when feature sparsity is present such as when certain student engagement attributes are infrequently activated.

3.7.3 Adaptive Moment Estimation with Maximum

AmsGrad is a variant of the Adam optimizer designed to address its theoretical limitations regarding convergence. While Adam offers fast convergence and adaptive learning rates, it may fail to converge in certain convex optimization scenarios due to the fluctuating nature of the second moment estimate. AmsGrad improves upon Adam by introducing a mechanism to ensure that the denominator used in the parameter update rule does not decrease over time. It maintains a maximum of the second moment values, preventing the learning rate from becoming too aggressive. The AmsGrad update is defined as:

$$m_i = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{3.25}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{3.26}$$

$$\hat{v}_t = \max(\hat{v}_{t-1}, v_t) \tag{3.27}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \tag{3.28}$$

By enforcing a non-decreasing learning rate denominator, AmsGrad guarantees more stable updates and better convergence properties, especially in settings where Adam struggles due to rapidly changing gradient dynamics. In the training of MLP for academic performance classification, AmsGrad helps to maintain consistency across epochs, avoiding sudden spikes or drops in the loss curve. This stability is particularly useful in small or imbalanced datasets, such as those involving student engagement metrics, where convergence behaviour needs to be carefully controlled to avoid overfitting or erratic generalization. While its performance may be slightly slower than Adam in practice, AmsGrad provides greater reliability, especially in applications where convergence assurance is critical.

3.7.4 Nesterov-Accelerated Adaptive Moment Estimation

Nadam is an advanced optimization algorithm that extends the Adam optimizer by integrating Nesterov Accelerated Gradient (NAG) into Adam's adaptive moment estimation framework. While Adam uses momentum based on the exponentially weighted moving average of past gradients, it does not anticipate future gradients. Nadam addresses this by modifying the momentum term to incorporate a lookahead step, which results in faster and smoother convergence. Like Adam, Nadam maintains both first and second moment estimates of the gradients:

$$m_i = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{3.29}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{3.30}$$

With bias corrections:

$\widehat{m}_t = \frac{m_t}{1-\beta_1^t}$ and $\widehat{v}_t = \frac{v_t}{1-\beta_1^t}$.

The key difference is in the parameter update step, which includes the Nesterov adjustment:

$$\theta_{t+1}=\theta_t - \frac{\eta}{\sqrt{\hat{v}_t+\epsilon}}\left(\beta_1 \hat{m}_t + \frac{(1-\beta_1)g_t}{1-\beta_1^t}\right) \quad (3.31)$$

By combining the adaptive learning rates of Adam with the predictive nature of Nesterov momentum, Nadam improves learning efficiency, especially in models with non-convex loss surfaces.

Nadam is particularly effective in training deep neural networks, where fast convergence and reduced oscillations in weight updates are desirable. In the MLP modelling for predicting student academic performance, Nadam helps stabilize the learning process by smoothing abrupt gradient fluctuations, especially in early epochs. Its anticipatory behavior allows the model to make better-informed updates, which can lead to higher validation performance and reduced risk of overfitting. Due to these properties, Nadam is well-suited for datasets like yours that involve imbalanced class distributions and complex engagement feature interactions.

3.7.5 Stochastic Gradient Descent with Momentum

SGD with Momentum is one of the foundational optimization algorithms used in training neural networks. It updates the model parameters by moving them in the direction of the negative gradient of the loss function. Unlike batch gradient descent, which uses the entire dataset to compute gradients, SGD uses a single data point or mini-batch per iteration, making it computationally efficient and memory-friendly. The basic update rule is given by:

$$\theta = \theta - \eta \nabla_{\theta} L(\theta) \tag{3.32}$$

where θ represents the model parameters, η is the learning rate, and $\nabla\theta L(\theta)$ is the gradient of the loss function with respect to θ Although simple and scalable, SGD can suffer from high variance in updates and slow convergence, particularly when the loss surface is steep or contains saddle points.

To address these limitations, SGD with Momentum introduces a velocity term that accumulates a moving average of past gradients, thereby smoothing the update trajectory and accelerating convergence in consistent gradient directions. The momentum mechanism is formulated as:

$$v_t = \gamma v_{t-1} + \eta \nabla_{\theta} L(\theta),\ \theta = \theta - v_t \tag{3.33}$$

where $\gamma \in [0,1)$ is the momentum coefficient, typically set to 0.9. This technique helps dampen oscillations and provides inertia to the updates, making it particularly useful in navigating ravine-shaped error surfaces or sparse gradient spaces. In this study, SGD with Momentum is used as a baseline optimizer for training the MLP model. Its inclusion allows for comparative analysis against more adaptive optimizers and provides insights into convergence behaviour and generalization under minimal adaptivity conditions.

3.8 Other Intelligent Technique

While the MLP serves as the central model in this study for predicting student academic performance, it is essential to evaluate its effectiveness alongside other established intelligent classification techniques. These alternative models are widely used across various machine learning domains and offer diverse approaches to pattern recognition, decision-making, and feature interaction. To support a fair and structured comparison, this chapter presents the basic theoretical background of each selected technique and outlines their core mechanisms as applied to classification tasks involving structured educational data.

The techniques discussed in this section include SVM, CNN, kNN, RNN, and LSTM. SVM utilizes hyperplanes to separate data in high-dimensional space. CNN employs convolutional operations to learn spatial hierarchies in data. kNN classifies data points based on their proximity to labeled instances. RNN uses internal memory to process input sequences, and LSTM enhances this capability with specialized memory gates. Each of these models is introduced in terms of its fundamental principles and learning structure, forming the theoretical foundation for the comparative analysis with MLP conducted in later sections of this study.

3.8.1 Support Vector Machine

SVM is a supervised learning algorithm widely used for classification and regression tasks. It operates by finding the optimal hyperplane that maximally separates data points belonging to different classes in a high-dimensional feature space. The central idea is to identify a decision boundary that not only separates the classes but also maximizes the margin and the distance between the hyperplane and the closest data points from each class, known as support vectors.

For a binary classification problem, given a set of training samples (x_i, y_i), where $x_i \in \mathbb{R}^n$denotes the input feature vector and $y_i \in \{-1, +1\}$ represents the corresponding class label, the objective of the Support Vector Machine (SVM) is to determine an optimal weight vector $\mathbf{w}$ and bias term b that define a linear decision function:

$$f(x)=\mathbf{w}^{\top}x+b. \tag{3.34}$$

This decision function must satisfy the margin constraint:

$$y_i(\mathbf{w}^{\top}x_i+b)\geq 1,\forall i. \tag{3.35}$$

The optimal separating hyperplane is obtained by minimizing the following objective function:

$$\min_{\mathbf{w},\,b} \quad \frac{1}{2}\|\mathbf{w}\|^2 \tag{3.36}$$

subject to the above constraint. This formulation is known as the hard-margin SVM. For datasets that are not perfectly linearly separable, the soft-margin SVM introduces slack variables and a regularization term to allow some misclassifications while still maximizing the margin. In practice, SVM often employs the kernel trick to transform non-linearly separable data into a higher-dimensional space where a linear separation is possible. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid kernels.

SVM is particularly suitable for high-dimensional spaces and structured datasets, making it a valuable model in applications involving academic performance classification. Its ability to construct well-defined decision boundaries based on the most informative data points (support vectors) allows for robust and interpretable classification. In this study, SVM is implemented to evaluate its performance relative to other intelligent models, providing a comparative benchmark alongside the Multilayer Perceptron (MLP) classifier.

3.8.2 CNNs

CNNs are a class of deep learning models specifically designed to process data with a known grid-like topology, such as images or sequential signals. Originally developed for image recognition tasks, CNNs have since been adapted to various domains due to their powerful feature extraction capabilities. The fundamental building block of a CNN is the convolutional layer, which applies a set of filters (or kernels) across the input data to detect local patterns. These filters slide across the input in a process called convolution, producing feature maps that highlight spatial hierarchies and relationships in the data.

Mathematically, the one-dimensional convolution operation between an input signal $x$and a filter w of length k is defined as:

$$s(t)=(x*w)(t)=\sum_{i=0}^{k-1} x(t+i)\, w(i) \tag{3.37}$$

where $s(t)$ denotes the output feature at position t, $x(t+i)$ represents the input signal segment, and $w(i)$ corresponds to the filter weights.

Following convolution, a non-linear activation function such as ReLU is applied to introduce non-linearity. Subsequent pooling layers (e.g., max pooling) reduce the spatial dimensions of the feature maps, improving computational efficiency and controlling overfitting. Finally, one or more fully connected layers are used to perform classification based on the extracted features.

CNNs are most commonly used in image-based applications, their structure can be adapted to handle 1D structured data, such as sequences of student activity features, by using 1D convolutions. In this study, CNN is included to explore how its local pattern detection capabilities perform in modelling academic engagement behaviours. The theoretical grounding and architecture of CNN provide a contrasting approach to traditional MLP models, enriching the comparative analysis of intelligent classification techniques in educational data mining.

3.8.3 k-Nearest Neighbours

kNN is a simple yet effective non-parametric, instance-based learning algorithm used for classification and regression tasks. The core principle of kNN is that the classification of a new, unseen data point is determined by the majority class among its k closest neighbors in the training dataset. It does not require an explicit training phase or model fitting; instead, it relies entirely on the distance metrics calculated at the time of prediction. This makes kNN intuitive and easy to implement, especially for low- to moderate-dimensional datasets. To identify the nearest neighbours, a distance metric most commonly the Euclidean distance is employed. Given two feature vectors $\mathbf{x}$ and $\mathbf{x}_i$ in an n-dimensional feature space, the Euclidean distance $d(\mathbf{x}, \mathbf{x}_i)$ is defined as:

$$d(\mathbf{x},\mathbf{x}_i)=\sqrt{\sum_{j=1}^{n}\left(x_j-x_{ij}\right)^2} \tag{3.38}$$

where x_j and x_{ij} denote the j-th components of the feature vectors $\mathbf{x}$ and $\mathbf{x}_i$, respectively.

Once distances to all training samples are computed, the algorithm selects the k closest samples and assigns the class label based on majority voting among those neighbours. The choice of k significantly affects performance: smaller values can lead to sensitivity to noise, while larger values may smooth out decision boundaries. In this study, kNN is used as a comparative baseline to evaluate the performance of more complex models such as MLP, SVM, and CNN. Its simplicity and interpretability provide a valuable point of reference when assessing the effectiveness of student academic performance classifiers.

3.8.4 Recurrent Neural Networks

RNNs are a class of artificial neural networks designed for processing sequential data, where the order and temporal dependencies between elements are important. Unlike feedforward neural networks, RNNs have recurrent connections that allow the network to retain information from previous time steps. This internal memory enables RNNs to learn temporal patterns and relationships within sequences, making them suitable for tasks such as time series forecasting, natural language processing, and sequential behaviour modelling.

The defining characteristic of a Recurrent Neural Network (RNN) is its ability to share parameters across time steps, allowing each output to depend not only on the current input but also on the hidden state from the previous time step. Mathematically, the hidden state $\mathbf{h}_t$ at time step t is computed as:

$$\mathbf{h}_t=f,(\mathbf{W}_{xh}\mathbf{x}_t+\mathbf{W}_{hh}\mathbf{h}_{t-1}+\mathbf{b}_h) \tag{3.39}$$

where $\mathbf{x}_t$ denotes the input vector at time t, $\mathbf{W}_{xh}$ and $\mathbf{W}_{hh}$ are the input-to-hidden and hidden-to-hidden weight matrices, respectively, $\mathbf{b}_h$ is the bias vector, and $f(\cdot)$ represents a nonlinear activation function such as $\tanh$ or ReLU. The network output at time step t can then be computed from the hidden state.

In this study, RNN is utilized to explore its ability to model underlying dependencies in student academic engagement data, even though the data is not strictly sequential. The architecture offers an alternative way to capture patterns that may not be explicitly time-ordered but still benefit from memory-based learning.

3.8.5 Long Short-Term Memory

LSTM networks are a specialized type of RNN designed to effectively learn and retain long-term dependencies in sequential data. While traditional RNNs can capture temporal relationships, they often suffer from the vanishing or exploding gradient problem, which hampers their ability to remember patterns over long sequences. LSTM addresses this limitation by gated memory cells, which regulate the flow of information over time using a carefully designed structure of gates.

An LSTM unit comprises three primary gates: the input gate, forget gate, and output gate. These gates control how much of the past information should be retained or discarded, and how much of the current input should influence the memory cell. The operations at each time step t are governed by the following equations:

$$i_t=\sigma(W_i[h_{t-1},x_t]+b_i) \tag{3.40}$$

$$\widetilde{C}_t=\tanh(W_c[h_{t-1},x_t]+b_c) \tag{3.41}$$

$$C_t=f_t\cdot C_{t-1}+i_t\cdot\widetilde{C}_t \tag{3.42}$$

$$o_t=\sigma(W_o[h_{t-1},x_t]+b_o) \tag{3.43}$$

$$h_t=o_t\cdot\tanh(C_t) \tag{3.44}$$

These equations describe the internal operations of a LSTM network at time step t, where x_t denotes the input, h_t represents the hidden state, and C_t is the cell state. The

sigmoid function $\sigma(\cdot)$ regulates the forget, input, and output gates, controlling the flow of information through the network, while W and b correspond to the learnable weight matrices and bias vectors. The forget gate determines how much information from the previous cell state C_{t-1} is retained, the input gate governs the incorporation of new information, and the output gate controls the contribution of the updated cell state to the hidden state. In this study, LSTM is evaluated alongside other models to investigate its capability to learn patterns in structured student engagement data by treating features as pseudo-sequential inputs. Its gated memory mechanism offers an additional perspective on modeling student behavior and predicting academic outcomes using deep learning.

3.9 Chapter Summary

This chapter established the theoretical foundation for the full modelling pipeline used in this study, covering data preparation, class imbalance handling, feature selection, model construction, and comparative benchmarking. It first justified the use of Min–Max normalization to rescale engagement features into a consistent numerical range, supporting stable gradient-based learning and fair distance computation in instance-based methods such as kNN. It then introduced ADASYN as the chosen synthetic oversampling approach, emphasizing its adaptive generation of minority samples based on local learning difficulty, and detailed the mathematical steps governing imbalance quantification, difficulty scoring, allocation of synthetic samples, and interpolation-based sample generation. To ensure that oversampling does not distort the original data characteristics, the chapter also presented a structured validation framework using the Kolmogorov–Smirnov test, histogram comparison, and box plot analysis to confirm that the synthetic dataset remains distributionally consistent with the original dataset.

In addition, this chapter formalized the role of feature selection and learning optimization in improving model generalization and interpretability for educational datasets. Five feature selection methods were reviewed highlighting complementary perspectives ranging from model-agnostic dependency measures to embedded sparsity and ensemble-based importance scoring. The chapter also explained the theoretical basis of the optimizers evaluated in this, clarifying how their update mechanisms influence convergence behaviour, stability, and overfitting control in MLP training. Finally, alternative intelligent techniques such as SVM, CNN, kNN, RNN, and LSTM

were introduced to provide a robust comparative baseline against the MLP, ensuring that subsequent chapters can interpret performance differences based on well-defined theoretical mechanisms rather than empirical outcomes alone.

CHAPTER 4
PROJECT METHODOLOGY, DEVELOPMENT AND IMPLEMENTATION

4.1 Chapter Overview

This chapter presents the methodological framework used to develop and evaluate a classifier model for student academic performance. Section 4.1 provides an overview of the research methodology, outlining the key processes involved. Section 4.2 describes the data collection process, specifically how student interactions with an online learning platform were recorded, followed by Section 4.3, which details the types of data collected, including key engagement features and academic activities. Section 4.4 explains data preprocessing steps, including handling missing values and normalizing data for consistency. Section 4.5 addresses the issue of class imbalance and the application of the ADASYN algorithm, ensuring fair representation of minority class instances. Section 4.6 discusses data validation techniques, including statistical tests used to assess dataset reliability. Section 4.7 explores various feature selection methods employed to identify the most influential predictors of student performance. Section 4.8 details MLP modelling and optimization, where hyperparameter tuning and different optimizers were systematically tested to enhance classification performance. Finally, Section 4.9 presents a comparative evaluation of optimized MLP against other ANN architectures, assessing their classification accuracy and overall effectiveness. The methodological steps outlined in this chapter establish the comprehensive foundation for the analysis and discussion presented in the subsequent chapter.

4.2 Methodology Overview

Figure 4.1 shows the flowchart that summarized the overall research methodology that was conducted for this study. The process began with data acquisition using an online learning platform for ECE 431 - Computer Programming course, involving a total of 393 students. Using within this learning platform environment, the system recorded activities done platform directly related to their teaching learning. These activities included, but were not limited to, accessing the online lecture slide,

exercises within tutorials just to name a few and the nature of these activities will be interactions were then converted into a structured, readable format and securely to the researcher. These data then following acquisition, these raw inputs underwent data transformation. The output from this stage then being used served as the input for the artificial model. The results performance from the exercise were of this model was then met or exceed predefined and it will give an idea on the performance criteria would be considered robust. if the results are above the required level, the model will be accepted for further analysis.

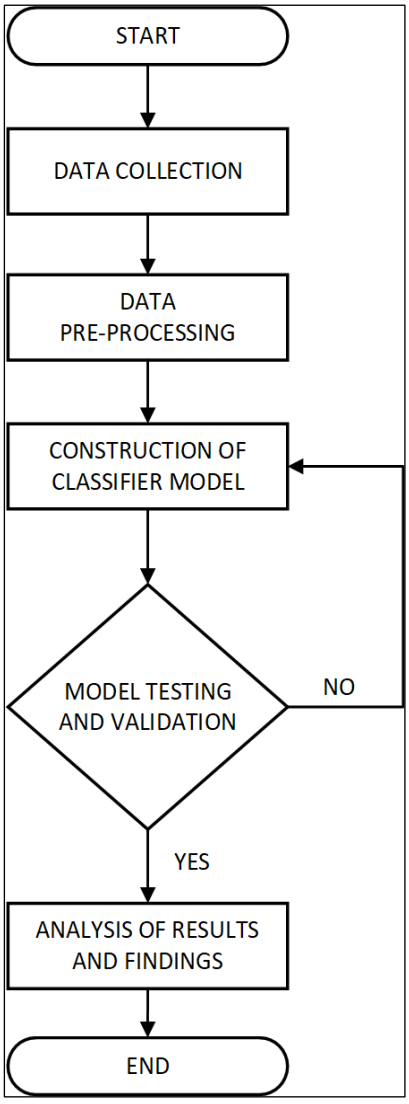


Figure 4.1 Experimental Flow Chart

4.3 Data Collection Method

The study used an original sample size of 393 students that enrolled in BEng. (Hons) Electronics Engineering – CEEE241 and BEng. (Hons) Electrical Engineering – CEEE242 that undergoes the course ECE431 (Computer Programming). This course is a compulsory subject for both programs and is a part of the academic plan set for them. The 393 students were divided into 13 groups according to the group that were being registered by the students prior to the commencement of the academic session. The online learning platform can be accessed with unique login identification assigned to each of the students and the lecturer for the purpose of teaching and learning process by the online learning platform system administrator at:

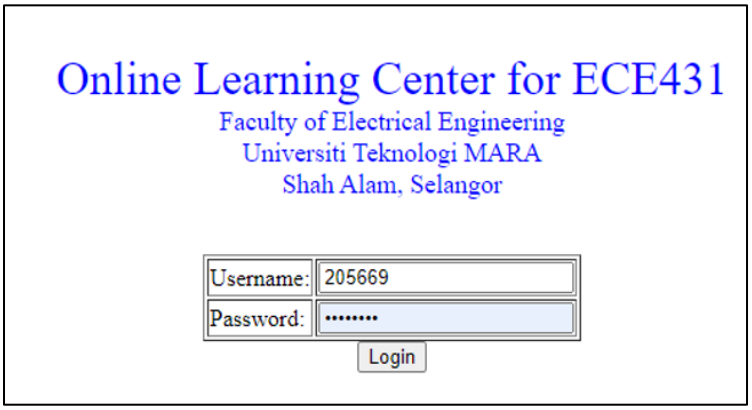


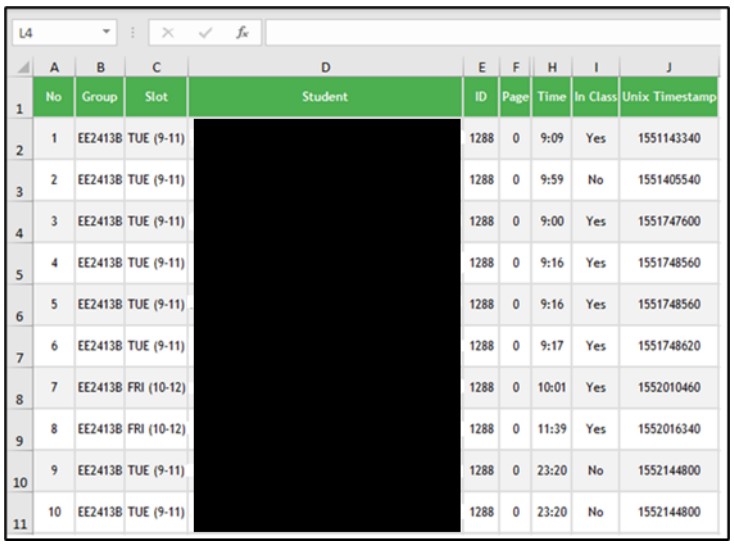
Figure 4.2 Login Page For ECE 431 Online Learning Platform
Source : https://azriaziz.uitm.edu.my/onlinelearning/index.php

All students registered for this course utilize the online learning platform as the primary tool for teaching and learning throughout the semester as the login page is shown in Figure 4.2. While the platform served as an innovation in the educational process, its use is not mandatory for the course. Data collection for this study were conducted from Week one until the end of week eight in a standard 17-week academic calendar, as stipulated in the academic documents for CEEE241 and CEEE242.

The course syllabus consists of 14 topics; however, this study focuses on t he first seven topics: Array, Control Flow, Functions, Loops, Operators, Pointers, and Types. These topics focuses on the fundamentals of the programming course. The excluded topics were Introduction & Basis of C++ Programming, Input & Output, Flow Chart, Files, Structure, and Object-Oriented Programming.

For each selected topic, several academic activities were measured. These include the frequency of students accessing lecture notes before, during, and after the designated lecture session, as well as the number of attempts made on exercises within the same time frames. Additionally, the platform captures the length of notes entered by students during their engagement with the system. One of the main components of the academic learning process is the tutorial section, where the system records all student attempts on tutorial questions, including their correctness.

Beyond these recorded activities, the study also includes students' their final Grade Point Average (GPA) as key components for the modelling process. All student interactions with the online learning platform are automatically logged and stored. These recorded activities are time-stamped using the Unix Timestamp format, which is later extracted into Excel as raw data for further processing.



No	Group	Slot	Student	ID	Page	Time	In Class	Unix Timestamp
1	EE2413B	TUE (9-11)	[redacted]	1288	0	9:09	Yes	1551143340
2	EE2413B	TUE (9-11)		1288	0	9:59	No	1551405540
3	EE2413B	TUE (9-11)		1288	0	9:00	Yes	1551747600
4	EE2413B	TUE (9-11)		1288	0	9:16	Yes	1551748560
5	EE2413B	TUE (9-11)		1288	0	9:16	Yes	1551748560
6	EE2413B	TUE (9-11)		1288	0	9:17	Yes	1551748620
7	EE2413B	FRI (10-12)		1288	0	10:01	Yes	1552010460
8	EE2413B	FRI (10-12)		1288	0	11:39	Yes	1552016340
9	EE2413B	TUE (9-11)		1288	0	23:20	No	1552144800
10	EE2413B	TUE (9-11)		1288	0	23:20	No	1552144800

Figure 4.3 Sample Of Unix Timestamp For Data Collection

The Unix Timestamp, also known as Unix time or Epoch time, is a numerical representation of time that counts the number of seconds elapsed since January 1, 1970, at 00:00:00 UTC. This system provides a standardized and efficient way to track time across different platforms and applications. One of its primary advantages is that it represents time as a single integer value, eliminating the complexities associated with various date and time formats used in different systems. This simplicity makes Unix time particularly useful for data logging, synchronization, and computational efficiency, as it ensures consistency across databases, servers, and applications.

In this study, all student activities recorded on the online learning platform are automatically logged with a Unix timestamp as shown in Figure 4.3. Each action, such as accessing lecture materials, attempting exercises, or engaging in tutorial sessions, is assigned a unique timestamp indicating the exact moment it occurred. These timestamps provide an accurate and sequential record of student interactions, making it possible to analyse patterns in their engagement with the course materials.

To facilitate further processing and analysis, the recorded timestamps are extracted and transferred into an Excel file. This allows for easier manipulation of the data, including sorting, filtering, and transforming the time-based information into a more human-readable format. By converting the Unix timestamps into standard date and time values, researchers can effectively analyse trends in student behaviour, assess learning habits, and correlate activity patterns with academic performance outcomes.

No	Name	KP UiTM	In	Out In	Out Out	In	Out In	Out Out	Correct 3 and above	Answered All Q	Wrong B4 Correct	Test 1	Lulus Gagal
1	[redacted]		50	23	54	14	5	10	4	2	1	1	1
2			43	18	27	26	2	7	4	4	1	2	1
3			118	0	0	34	0	0	4	4	1	1.5	1
4			72	9	0	29	3	0	4	4	1	2	1
5			23	1	0	11	0	0	4	4	2	1.5	1
6			55	20	11	16	5	0	4	4	1	2	1
7			43	9	0	23	3	0	4	4	1	1	1
8			89	17	20	22	13	5	4	4	1	2	1
9			81	24	64	29	6	15	3	4	2	0	0
10			111	40	41	44	12	11	4	4	2	2	1
11			52	24	0	15	3	0	4	4	1	2	1
12			97	30	0	43	10	0	4	4	2	2	1
13			44	13	44	11	1	8	4	4	3	2	1
14			37	21	25	13	2	9	4	4	1	2	1
15			29	22	37	11	6	4	3	4	2	2	1

Figure 4.4 Conversion Of Timestamp Activities

Figure 4.4 presents a table in Excel that contains student details along with the recorded frequency of their academic activities on the online learning platform. These activities, originally logged using Unix timestamps, have been converted into a more readable format for analysis. The data shown in the table represents the raw dataset extracted from the platform, capturing students' interactions and activities with various learning materials. This dataset serves as the foundation for further processing, enabling a detailed examination of student engagement patterns and their correlation with academic performance.

4.4 Type of Data Collected

The data collected in this study consists of 12 key features that capture students' interactions with an online learning platform over eight weeks. These features are derived from five core topics covered in the course: Types, Operators, Functions, Loops, and Flow Control. The dataset includes engagement metrics such as the frequency of accessing lecture slides and additional notes before, during, and after lectures, as well as outside designated lecture times. Additionally, it records students' note-taking behaviour, measured by the length of notes entered the platform. These data points provide insight into how students interact with learning materials and engage with course content throughout the semester.

Beyond lecture-based engagement, the dataset also tracks student participation in exercises and tutorials. It records the frequency of students attempting exercises before, during, and after lecture sessions, as well as outside structured learning times. The tutorial-related features measure students' ability to correctly answer tutorial questions, including whether they achieved at least three correct answers, completed all tutorial questions, and the number of incorrect attempts before arriving at a correct answer. The inclusion of these features allows for a comprehensive analysis of students' study habits, engagement patterns, and their potential impact on academic performance. The 12 features are present in Table 4.1 below.

Table 4.1
12 Features With the Descriptions

Feature Number	**Feature Description**
Feature 1	Access to Lecture Slide and Additional Notes Before Lecture
Feature 2	Access to Lecture Slide and Additional Notes During Lecture
Feature 3	Access to Lecture Slide and Additional Notes After Lecture
Feature 4	Access to Lecture Slide and Additional Notes Outside Lecture
Feature 5	Note Length
Feature 6	Exercise Before Lecture
Feature 7	Exercise During Lecture
Feature 8	Exercise After Lecture
Feature 9	Exercise Outside Lecture

Feature Number	Feature Description
Feature 10	Tutorial Correct 3 and above
Feature 11	Tutorial Answer All Questions
Feature 12	Tutorial Wrong Before Correct

The course GPA obtained by students served as the output variable in this study, representing their overall academic performance. A GPA greater or equal than 3.00 is classified as output 1 (High), while a GPA below 3.00 is classified as output 0 (Low). The faculty had put a benchmark of 65% of attainment achieved by the students which were measured the closing the loop process. This can be indirectly translated to GPA of 2.67 which is equal to 65% or B-. Furthermore, the outcome of the findings in the employment advertisement by the future employer shows that most of the future employer would set a minimum of 3.00 CGPA as the requirement for most of the employment prospect. By analysing the relationship between the 12 collected features within 5 subtopics over 8 weeks and the CGA obtained at the end of the semester, this study aims to develop a predictive model that can identify patterns in student engagement and provide insights into factors influencing academic success.

4.5 Data Preprocessing and Transformation

As part of the data preprocessing and transformation process, an initial screening was conducted to ensure data completeness and quality. The original dataset consisted of 393 student records, which were transferred to Excel for further processing. During this stage, any rows or columns containing missing or incomplete data were systematically identified and removed. The data that were identified as incomplete can be as a result of a few criteria such as missing of data in the column section which can be translated to the inconsistency usage of the learning management system by the user. These incomplete data also include students that withdraw during the semester. The removal of the incomplete data will affect the minority classes output but executing any modelling with an incomplete or defect dataset will produce a much worse effect at the later stage such as it could compromise the accuracy and reliability of the predictive model, leading to potential biases and errors in classification. Therefore, eliminating

incomplete records was a necessary step to maintain data integrity and ensure that only high-quality inputs were used for further analysis.

Following this elimination process, 74 student records were identified as complete, containing sufficient information regarding their academic activities on the online learning platform. The remaining records were excluded due to substantial missing values, making them unsuitable for model training. Retaining only complete data helps minimize inconsistencies and improves the overall robustness of the dataset, allowing for more reliable feature extraction and classification outcomes

With a finalized dataset of 74 complete samples, the next step involved applying Min-Max normalization to scale the feature values within a uniform range. This transformation ensures that no single feature disproportionately influences the model due to variations in numerical scale. By normalizing the dataset, the learning algorithm can process the data more efficiently, improving model convergence and generalization while preventing features with larger magnitudes from dominating the training process.

4.6 Class Imbalance and ADASYN for Synthetic Data Generation

Class imbalance is a difficult challenge in predictive modelling, particularly in classification tasks where either one output class is significantly underrepresented. In this study, the dataset consists of 74 student records, with 53 samples classified as output class 1 (GPA $\geq$ 3.00) and only 21 samples classified as output class 0 (GPA $<$ 3.00).

From these information's, the study chooses 2 different level for the output which is output class 1 (GPA $\geq$ 3.00) and output class 0 (GPA $<$ 3.00) as the desired output class for the modelling.

This output class was clearly an imbalance set of data, and the imbalance creates a skewed distribution. Where the majority class was overrepresented compared to the minority class. Such an imbalance can lead to a bias in the learning algorithm, causing the model to favour the dominant class while neglecting the minority class, ultimately affecting the model's predictive performance.

When training a MLP model on an imbalanced dataset, the classifier may achieve high overall accuracy but perform poorly in correctly predicting the minority class. This happens because the model tends to minimize overall error by prioritizing the majority class, leading to low recall for the minority class and an increased number

of false negatives. As a result, students at risk of lower academic performance (class 0) may not be correctly identified, reducing the model's effectiveness for early intervention strategies. To address this issue, further techniques must be explored to balance the dataset and improve the classification of underrepresented samples, ensuring that both classes are fairly represented in the predictive framework.

To ensure that the model generalizes well and fairly represents both classes, an approach is required to balance the dataset and enhance classification accuracy without compromising model performance. Synthetic data generation has become an essential technique for addressing challenges such as data scarcity, privacy concerns, and class imbalance in machine learning applications. By creating artificial yet statistically representative data, synthetic data allows for model training without exposing sensitive information or relying solely on limited real-world samples. In this study, the ADASYN algorithm is employed to generate synthetic data for the minority class, ensuring a more balanced dataset. Unlike traditional oversampling methods that replicate existing samples, ADASYN creates new synthetic instances based on the density distribution of minority class examples. This adaptive approach ensures that harder-to-classify instances receive more synthetic samples, improving the overall robustness of the classification model.

ADASYN was particularly effective in mitigating the negative effects of class imbalance, which can otherwise lead to biased predictions favouring the majority class. In this study, after preprocessing the dataset, it was observed that from the 74-student data, only 21 belonged to the minority class, while 53 represented the majority class. This significant imbalance in the output distribution could lead to a classifier that is biased toward the majority class, reducing its ability to correctly predict students at risk of lower academic performance. To address this, ADASYN was applied to enhance the representation of underrepresented classes in student performance data, dynamically generating synthetic samples based on the feature space distribution. Unlike traditional oversampling techniques which may lead to overfitting, ADASYN prioritizes harder-to-classify instances, ensuring a more adaptive and effective balancing of the dataset. By applying this technique, the dataset becomes more suitable for training an MLP classifier that can fairly and accurately predict student academic performance across both majority and minority classes.

4.7 Data Validation

Ensuring the integrity and consistency of the synthetic data generated through ADASYN was a critical step in the validation process. A comprehensive statistical analysis was conducted to compare the statistical properties of the original and augmented datasets. The Kolmogorov-Smirnov (K-S) test was applied to evaluate the empirical cumulative distribution functions of both datasets, assessing whether the synthetic samples maintained the same distributional characteristics as the original data. A high p-value and low K-S statistic indicated that ADASYN effectively preserved the underlying structure of the dataset without introducing significant deviations.

In addition to the K-S test, box plots and histograms analysis were utilized to visually inspect the feature distributions before and after synthetic data generation. The box plots provided insights into variability, interquartile ranges, and potential distributional shifts, ensuring that the synthetic data did not alter the statistical spread of key features. Meanwhile, histograms were used to examine the overall shape and density of the distributions, confirming that the augmentation process did not introduce anomalies or distortions.

The results of these validation techniques confirmed that ADASYN successfully generated synthetic samples while maintaining the statistical integrity of the dataset. By preserving feature distributions and preventing artificial bias, ADASYN contributed to a more balanced and representative dataset, ultimately enhancing the fairness and predictive performance of the MLP classifier. This validation step was critical in ensuring that the model learned meaningful patterns from both the majority and minority classes, leading to improved generalization and classification accuracy.

4.8 Feature Selection Method

Feature selection is an essential component in the development of robust and efficient predictive models, particularly within educational data mining, where input variables often reflect complex student behaviours and engagement patterns. The presence of irrelevant, redundant, or weakly correlated features can negatively impact the learning process by introducing noise, increasing overfitting risk, and reducing computational efficiency. To address these challenges, this study explored five established feature selection techniques: Mutual Information, RFE, Random Forest

Importance, XGBoost Importance, and L1 Lasso. Each of these methods offers a unique approach to identifying feature relevance and contributes distinct analytical perspectives to the feature reduction process.

Mutual Information measures the dependency between each feature and the target label, capturing both linear and non-linear relationships. It is a filter-based method that ranks features according to how much information they share with the output variable. Recursive Feature Elimination operates by iteratively removing the least significant features based on a model's internal weights, retraining and re-ranking the remaining features until a predefined number of inputs is reached. Random Forest Importance derives feature significance scores from decision tree ensembles by evaluating how much each feature reduces classification impurity across trees. Similarly, XGBoost Importance utilizes boosting techniques to assign importance based on gain, frequency, or cover metrics derived from its gradient-boosted tree structure. L1 Lasso, commonly implemented through logistic regression, applies a penalty to the absolute value of coefficients, forcing less influential features to zero and resulting in a sparse and interpretable model.

These methods collectively represent a blend of filter, wrapper, and embedded approaches, enabling a comprehensive evaluation of feature relevance from multiple algorithmic perspectives. Each method was applied independently to the original dataset consisting of 12 student academic engagement features. The selected feature subsets were then examined for their impact on model performance, using accuracy, precision, recall, and F1-score as evaluation metrics. This multi-method approach ensured that the feature selection process was not overly dependent on the assumptions or biases of a single technique.

By integrating these diverse feature selection methods, the study aimed to identify the optimal feature subset that maximized classification performance while ensuring model generalizability. The results from feature selection were further validated through experimentation, demonstrating that reducing the feature set significantly enhanced model efficiency without compromising accuracy. The final selected features were used as inputs for training the MLP model, ensuring that only the most relevant academic activities were included in the predictive framework.

4.9 Multilayer Perceptron Modelling and Model Optimization

MLP is a well-established artificial neural network architecture designed for predictive modelling, particularly in classification tasks that require capturing complex, non-linear relationships. In this study, an MLP model was developed to predict student academic performance based on selected features derived through feature selection techniques. The MLP architecture consists of an input layer, two hidden layers, and an output layer, configured to efficiently process student engagement data while minimizing overfitting. The number of neurons in the input layer corresponds to the number of selected features, which ranges from 6 to 12 depending on the feature selection method applied. Two hidden layers are incorporated to enhance the model's ability to learn underlying patterns, with the number of neurons in these layers determined through hyperparameter tuning. The output layer consists of a single neuron with a Sigmoid activation function, allowing the model to perform binary classification in predicting student success or failure.

To ensure an effective training process, the dataset was divided into three subsets: 70% for training, 15% for validation, and 15% for testing. This split ensures that the model learns from a sufficiently large portion of the dataset while also being evaluated on unseen data to prevent overfitting. The training process utilizes backpropagation with gradient descent to iteratively adjust the model's weights based on the computed error. Binary cross-entropy loss is selected as the loss function, as it is well-suited for binary classification tasks where predictions are probabilistic. In addition to defining the network architecture, the study explores various optimization algorithms to improve model convergence and stability.

The optimizers tested include adaptive learning rate methods such as AdamW, AdaGrad, AmsGrad, and Nadam, as well as a traditional gradient-based method, with SGD with Momentum.

The performance of the trained MLP model is evaluated using key classification metrics, including accuracy, precision, recall, and F1-score. Accuracy measures the proportion of correctly classified instances, while precision assesses the correctness of positive classifications. Recall determines the model's ability to identify positive cases, particularly in predicting at-risk students, while the F1-score provides a balanced measure of classification performance by incorporating both precision and recall. The model's effectiveness is further analysed through confusion matrices, allowing a

detailed comparison of classification errors across different training configurations. The results of these evaluations provide insights into the impact of feature selection on classification performance and generalization, as well as the effectiveness of different optimizers in improving learning efficiency.

The optimization of the MLP model is a crucial step in improving its learning efficiency, convergence speed, and generalization performance. This process involves hyperparameter tuning, analysing the bias-variance trade-off, and selecting the most effective optimizer for training. Several key hyperparameters are fine-tuned to enhance model performance, including the number of hidden neurons, batch size, learning rate, dropout rate, and weight regularization techniques. The number of neurons in the hidden layers is adjusted to ensure that the network is complex enough to learn meaningful patterns without becoming excessively large, which could lead to overfitting. The learning rate is carefully tuned for each optimizer to prevent excessively large weight updates, which could lead to unstable training, or excessively small updates, which could result in slow convergence. Dropout regularization is applied to randomly deactivate neurons during training, reducing reliance on specific neurons and improving the model's ability to generalize.

A critical aspect of optimizing the MLP model was managing the bias-variance trade-off. A high-bias model, or underfitting, occurs when the model is too simple to capture meaningful patterns in the data, leading to poor training accuracy. Conversely, a high-variance model, or overfitting, occurs when the model learns patterns specific to the training data but fails to generalize to new, unseen data. To address these issues, various regularization techniques are applied to ensure that the model captures essential features while avoiding excessive complexity. The model's generalization ability is assessed by comparing training and testing accuracy, with an optimal model expected to have minimal discrepancies between these two performance measures.

The choice of optimizer plays a significant role in determining the efficiency and stability of the training process. Optimizers were evaluated based on their classification accuracy, precision, recall, F1-score, and convergence speed. The number of epochs required for each optimizer to converge is analysed, as well as the smoothness of the loss function during training. The best-performing optimizers exhibit both high classification accuracy and stable convergence, while poorly performing optimizers either fail to learn effectively or require excessive training epochs to achieve meaningful improvements. The study identifies AdamW, Adamax, and Nadam as the top-

performing optimizers due to their strong balance of accuracy, recall, and F1-score, as well as their ability to generalize well to unseen data. These optimizers dynamically adjust learning rates, ensuring that updates remain efficient throughout training while reducing the risk of overfitting.

Further analysis of optimizer performance reveals that traditional gradient descent methods, such as SGD and its momentum variant, struggle with slow convergence and increased sensitivity to local minima. While SGD with Momentum provides slight improvements over standard SGD, it remains less effective than adaptive optimizers in handling complex learning patterns. Among the adaptive optimizers, Adam and Adamax demonstrate strong performance, though Adam occasionally exhibits instability in generalization. Nadam, which incorporates Nesterov-accelerated gradients, shows promising results but requires fine-tuning to prevent slight overfitting tendencies. In contrast, Adadelta performs poorly, failing to effectively optimize the MLP model due to their inability to adapt well to the dataset's characteristics. These findings emphasize the importance of selecting optimizers that balance convergence speed with generalization ability, ensuring that the model maintains high predictive accuracy across training, validation, and testing phases.

The final optimized MLP model is configured based on the findings from hyperparameter tuning and optimizer selection. The final architecture consists of two hidden layers, with an optimal number of neurons selected based on experimental results. AdamW is chosen as the preferred optimizer due to its ability to provide stable learning while maintaining high accuracy and generalization. Regularization techniques, including dropout and L2 weight decay, are applied to further enhance the model's robustness. Performance evaluation using accuracy, precision, recall, and F1-score confirms that the optimized MLP model achieves strong predictive performance, making it suitable for identifying students at risk of academic underperformance. These findings provide a foundation for the development of more advanced predictive frameworks in educational data mining, with potential applications in personalized learning and early intervention strategies.

The model architecture, training process, and evaluation criteria were described in detail, along with the feature selection and optimization strategies employed to improve classification accuracy and generalization. A comparative analysis of various optimization techniques was conducted, leading to the selection of AdamW as the most effective optimizer for training the MLP model. The final optimized model incorporates

hyperparameter tuning, regularization techniques, and a carefully selected feature subset to achieve high accuracy and robust generalization. These methodological refinements ensure that the model can effectively predict student performance while minimizing overfitting, providing valuable insights into the factors influencing academic success. The findings of this study contribute to the broader field of educational data mining by demonstrating the effectiveness of ANN-based predictive models in academic performance assessment.

4.10 Testing Different ANN Architectures

To assess the comparative performance and generalizability of the optimized Multilayer Perceptron (MLP) model in classifying student academic outcomes, this study conducts a comparative analysis involving five widely used classifiers: Support Vector Machine (SVM), CNN, k-Nearest Neighbours (kNN), Recurrent Neural Network (RNN), and Long Short-Term Memory (LSTM). These models represent a mix of traditional machine learning approaches and deep learning architectures with different design paradigms, ranging from non-parametric models to spatial and temporal neural networks. The inclusion of these classifiers allows for a comprehensive evaluation of how the MLP model performs relative to standard ANN-based models that are often used in educational data mining.

Each model is implemented using default or standard architecture with fixed parameters and no additional hyperparameter tuning, except for the MLP which underwent systematic optimization. All models are trained on the same six input features, selected via Recursive Feature Elimination (RFE), and evaluated using identical data splits (70% training, 15% validation, and 15% testing) to ensure a fair and controlled comparison. The performance of each classifier is assessed using standard evaluation metrics, including accuracy, precision, recall, F1-score, and confusion matrices. This approach is intended to highlight the effects of feature selection and architectural tuning on model generalizability and classification performance across diverse ANN frameworks.

The CNN was implemented using TensorFlow and Keras. Although CNNs are typically used for image classification, they can also process structured data by treating the input as a one-dimensional sequence. In this study, the CNN architecture began with a Conv1D layer using 32 filters and a kernel size of 2, followed by a MaxPooling1D

layer with a pool size of 2. The output was flattened and passed through a Dense layer with 16 ReLU-activated neurons. To prevent overfitting, a Dropout layer with a rate of 0.5 was added before the final sigmoid output layer, which performed binary classification. The model was compiled using the Adam optimizer and binary cross-entropy loss function and trained for 50 epochs with a batch size of 8. Despite its suitability for spatial data, to evaluate its adaptability when applied to student engagement data in tabular format.

The Support Vector Machine (SVM) model was developed using the SVC function from the scikit-learn library. It employed the Radial Basis Function (RBF) kernel with default hyperparameters, including a penalty parameter (C) of 1.0, gamma set to 'scale', and the decision function strategy set to one-vs-rest. Additional settings included shrinking=True, probability=False, a tolerance of 0.001, and a fixed random state of 42 to ensure consistent results. SVM is known for its strong performance in high-dimensional classification tasks and was used here to serve as a baseline for evaluating the MLP's learning capability. The model was tested using the same normalized input features and train-validation-test split as the other models, and its performance was evaluated using confusion matrices and standard classification metrics.

k-Nearest Neighbours (kNN) classifier was implemented using scikit-learn's KNeighborsClassifier. It was configured with k=5, using Minkowski distance with p=2 (Euclidean distance), and uniform weighting. Before training, the input features were scaled using Min-Max normalization to ensure comparability across dimensions. The dataset was split using the same 70/15/15 ratio. The auto algorithm was selected for dynamic optimization of the nearest neighbour search. As a non-parametric, instance-based learning algorithm, kNN does not involve explicit training but instead relies on comparing test instances to stored training samples. Its inclusion in this study provides insight into how well distance-based methods perform on student performance data in the absence of model-based learning.

Recurrent Neural Network (RNN) was implemented using TensorFlow and Keras. Although RNNs are inherently designed for temporal data, this experiment investigated whether their memory-based architecture could offer any benefit in recognizing latent patterns in structured educational data. The six input features were reshaped into a 3D format to comply with RNN input requirements. The model included a RNN layer with 32 units and tanh activation, followed by a Dense layer with 16 ReLU-

activated neurons, a Dropout layer with a rate of 0.5, and a sigmoid output layer for binary classification. The model was trained using the Adam optimizer and binary cross-entropy loss over 50 epochs with a batch size of 8. Evaluation was conducted using confusion matrices and standard classification metrics.

The Long Short-Term Memory (LSTM) network, a variant of RNN designed to handle long-term dependencies, was also tested using the same input and evaluation configuration. The six features were reshaped into a 3D tensor to represent sequences, even though the data itself was non-temporal. The model consisted of an LSTM layer with 32 units and tanh activation, a Dense layer with 16 ReLU neurons, a Dropout layer with a rate of 0.5, and a final sigmoid-activated output layer. Like the RNN, the LSTM was compiled using the Adam optimizer and binary cross-entropy loss and trained over 50 epochs with a batch size of 8. The inclusion of LSTM aimed to explore whether its advanced memory gating mechanisms could detect useful structural patterns in static datasets, despite being originally designed for sequential tasks.

Each model was trained and evaluated using the same six selected features and identical data split strategy to ensure a fair and consistent comparison. The performance of all models was measured using accuracy, precision, recall, and F1-score, while the distribution of classification errors was analysed using confusion matrices. This comparative analysis was designed to evaluate the general classification effectiveness of each architecture under standardized conditions. No hyperparameter tuning was applied to these alternative models, as the objective was to benchmark them in their standard form against an optimized MLP model. The results from this evaluation inform the discussion presented in Chapter 5, where the outcomes of these models are analysed in greater detail, with specific emphasis on classification stability, generalization, and real-world applicability in academic performance prediction.

4.11 Project Methodology, Development and Implementation Summary

Chapter 4 presented the complete methodological framework for developing, optimizing, and evaluating a predictive model for student academic performance based on online learning engagement data. The study began with structured data acquisition from 393 students enrolled in ECE431 (Computer Programming), where interaction logs recorded via Unix timestamps were transformed into structured features representing academic engagement across eight weeks. After the data cleaning stage,

74 complete records were retained, and Min–Max normalization was applied to ensure uniform feature scaling. Given the class imbalance (53 high GPA ≥3.00 and 21 low GPA <3.00), the ADASYN algorithm was employed to generate synthetic minority samples, improving dataset balance while preserving statistical integrity through Kolmogorov–Smirnov testing, boxplot inspection, and histogram comparison. Feature selection was systematically conducted using five complementary, integrating filter, wrapper, and embedded approaches to identify the most influential predictors and reduce dimensionality without compromising predictive power.

Building upon the optimized feature subset, an MLP classifier was developed and tuned using a 70:15:15 train–validation–test split with binary cross-entropy loss and early stopping. Hyperparameters including hidden neuron configuration, learning rate, dropout, and regularization were systematically adjusted to manage the bias–variance trade-off. Five different optimizers such as AdamW, AdaGrad, AmsGrad, Nadam, and SGD with Momentum were evaluated based on convergence speed, stability across runs, and classification metrics such as accuracy, precision, recall, and F1-score. To benchmark generalization capability, the optimized MLP was compared against SVM, CNN, kNN, RNN, and LSTM under identical experimental conditions. The methodological structure established in this chapter ensures robustness, fairness, and reproducibility, forming a defensible foundation for the experimental analyses and performance discussions presented in the subsequent chapter.

CHAPTER 5
RESULT AND DISCUSSION

5.1 Chapter Overview

This chapter presents the analysis of the experimental results, beginning with an original data analysis, followed by the application of ADASYN for synthetic data generation to address class imbalance, and an evaluation of feature selection techniques and optimization algorithms for training an MLP model. The chapter first examines the statistical properties and distributional consistency between the original and ADASYN-generated data using methods such as the Kolmogorov-Smirnov test, histogram analysis, and box plot visualization, ensuring that the synthetic data maintains the integrity of the original dataset. The analysis then progresses to the feature selection stage, where various ranking methods, including Mutual Information, RFE, Random Forest Importance are compared to identify the most influential features for predicting student academic performance. The selected six-feature subsets are then evaluated using the Adam optimizer, and their results are benchmarked against the original 12-feature model to determine the impact of dimensionality reduction on classification accuracy and model generalization. Subsequently, different optimization algorithms are tested on the full 12-feature dataset, highlighting the effect of optimizer selection on convergence, stability, and predictive accuracy. The chapter concludes with a comparative evaluation of the best-performing MLP model, integrating the most suitable feature selection method and optimizer, against alternative ANN architectures, including SVM, kNN, and CNN, to assess the effectiveness of MLP in student performance prediction. Through these analyses, this chapter provides a structured and data-driven approach to identifying the most effective feature selection and optimization strategies for improving predictive modeling in academic performance forecasting.

5.2 Original Data Analysis

Table 5.1 shows the original dataset of 12 features that capture various aspects of student engagement, including access to lecture materials, note-taking behaviour, and participation in exercises and tutorials. The output variable represents student academic

performance, classified into two categories: pass (1) and fail (0). An initial examination of the data distribution reveals a significant class imbalance, where the number of students classified as passing was substantially higher than those classified as failing. This imbalance could impact model training, as the classifier may become biased toward the majority class. Additionally, some features, such as "Note Length" and "Exercise Attempts", exhibit high variance, indicating considerable differences in student study behaviours. These variations suggest that students employ different learning strategies, which may influence their academic outcomes.

Table 5.1
Original Data

Data	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12	Output
Data 1	0	130	72	99	928	0	86	15	37	8	13	11	0
Data 2	10	88	27	77	0	1	67	10	16	6	10	6	0
Data 3	14	159	115	59	14555	2	117	36	18	17	17	5	0
Data 4	1	87	36	101	0	0	69	10	36	8	10	6	0
Data 5	2	115	34	70	2678	1	73	12	20	9	13	10	0
Data 6	3	99	40	50	0	0	78	16	16	8	17	10	0
Data 7	0	90	45	28	0	0	63	13	12	11	14	7	0
Data 8	13	111	26	0	539	3	75	10	0	15	15	2	0
Data 9	1	80	124	54	0	0	47	29	17	8	12	10	0
Data 10	7	192	112	260	4697	0	111	43	99	9	17	9	0
Data 11	0	127	86	57	0	0	87	22	34	17	17	5	0
Data 12	2	64	30	118	1009	0	51	7	50	1	3	2	0
Data 13	2	74	40	10	1012	0	65	15	2	11	17	12	0
Data 14	0	53	21	36	192	0	38	6	12	10	13	8	0

Data	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12	Output
Data 15	10	127	109	115	11781	2	77	53	37	11	17	11	0
Data 16	3	53	16	81	294	1	47	3	39	4	7	4	0
Data 17	5	161	33	85	2711	3	113	12	40	15	17	7	0
Data 18	13	96	37	63	2094	1	70	19	21	8	13	7	0
Data 19	10	190	165	193	24428	6	67	91	66	12	15	12	0
Data 20	0	69	57	55	0	0	50	21	27	6	8	4	0
Data 21	0	68	10	52	0	0	37	2	15	3	6	4	0
Data 22	1	147	92	67	11095	0	59	46	13	16	13	7	1
Data 23	3	111	43	34	4865	0	78	10	7	17	17	6	1
Data 24	6	169	128	146	10109	1	63	70	75	15	17	5	1
Data 25	3	132	99	6	1230	2	67	38	2	17	15	4	1
Data 26	3	85	100	139	10822	2	73	20	29	17	17	2	1
Data 27	18	114	232	9	0	7	89	41	1	9	17	13	1
Data 28	5	179	138	14	8818	1	82	52	11	17	17	3	1
Data 29	9	263	170	178	4603	6	117	85	67	16	17	6	1

Data	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12	Output
Data 30	13	184	142	157	33360	3	77	72	74	17	17	6	1
Data 31	8	101	69	103	3715	4	74	12	27	17	17	5	1
Data 32	12	120	105	299	10461	1	67	45	130	16	17	5	1
Data 33	9	128	149	154	8445	4	84	39	34	16	17	4	1
Data 34	5	180	103	30	9411	0	75	33	4	17	8	6	1
Data 35	7	112	74	2	2720	2	60	33	0	16	10	3	1
Data 36	3	53	39	92	0	1	46	15	31	14	17	10	1
Data 37	15	162	122	121	15241	3	92	24	43	15	17	7	1
Data 38	0	105	48	0	112	0	51	14	0	15	17	6	1
Data 39	0	71	58	232	0	0	49	32	80	16	17	6	1
Data 40	0	114	79	57	0	0	72	24	25	12	17	6	1
Data 41	1	189	48	138	1482	0	110	16	51	16	17	5	1
Data 42	17	146	152	77	14863	5	118	39	30	15	17	7	1
Data 43	3	120	27	182	0	0	62	8	73	12	14	7	1
Data 44	11	140	146	162	20146	4	82	41	45	15	17	8	1

Data	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12	Output
Data 45	7	110	26	71	0	0	69	9	24	16	17	6	1
Data 46	5	139	140	50	3519	1	80	38	20	16	17	6	1
Data 47	14	196	138	195	5322	7	86	47	70	13	17	8	1
Data 48	22	231	258	140	20807	5	118	65	80	17	17	8	1
Data 49	5	108	110	18	4079	1	47	41	3	13	16	8	1
Data 50	0	52	31	47	0	0	45	11	19	14	17	7	1
Data 51	0	112	31	41	1504	0	67	17	12	17	17	9	1
Data 52	2	132	126	73	14545	0	54	49	29	15	15	7	1
Data 53	7	209	34	14	7947	3	78	25	2	16	13	7	1
Data 54	7	122	110	181	25706	7	93	16	65	17	17	1	1
Data 55	0	40	35	91	0	0	39	10	29	15	17	6	1
Data 56	8	119	45	57	689	1	69	11	16	16	17	5	1
Data 57	12	132	77	104	2483	4	58	36	42	8	13	10	1
Data 58	17	185	173	98	13249	3	93	42	41	15	15	6	1
Data 59	5	103	118	77	2079	0	47	45	19	15	12	10	1

Data	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12	Output
Data 60	14	165	67	0	9993	1	82	34	0	15	9	6	1
Data 61	5	162	168	193	6751	0	68	66	53	14	8	6	1
Data 62	6	204	108	141	28975	2	52	55	47	15	17	7	1
Data 63	17	161	119	139	11753	3	98	44	49	13	17	7	1
Data 64	6	76	73	107	1654	2	46	25	51	16	15	6	1
Data 65	61	174	105	65	1113	46	102	27	25	17	17	5	1
Data 66	3	127	71	60	946	0	55	42	27	12	14	7	1
Data 67	12	232	173	113	16650	2	118	44	32	16	16	6	1
Data 68	10	113	105	5	8570	5	53	47	2	16	17	7	1
Data 69	24	82	81	51	0	1	64	13	31	12	17	8	1
Data 70	13	123	147	141	16654	6	92	41	41	14	17	10	1
Data 71	13	126	138	77	4742	7	56	58	38	17	17	4	1
Data 72	9	80	82	123	3974	3	46	33	56	17	17	6	1
Data 73	8	97	54	123	238	0	72	23	50	15	17	4	1
Data 74	14	97	115	79	4272	6	74	31	22	16	17	6	1

The engagement features reflect students' learning behaviours, and their correlation with academic performance can provide insights into effective study patterns. Features such as "Tutorial Answer All Questions" and "Tutorial Correct 3 and Above" may have a strong influence on student performance, as they directly relate to problem-solving skills and comprehension. Conversely, features such as "Access to Lecture Slides Before Lecture" and "Exercise Before Lecture" may indicate proactive learning habits that contribute to better academic outcomes. If certain features exhibit weak correlation with student performance, they may not be significant predictors and could be omitted through feature selection techniques to enhance model efficiency. A more refined feature set will allow for better generalization of the classification model, improving its predictive accuracy.

A consistent study routine that includes engaging with lecture materials, taking notes, doing exercises, and completing tutorials is important for academic success. Students who regularly access lecture slides and practice exercises before and after class tend to understand better and perform well. In contrast, those who study last minute or skip tutorials may struggle. Using ADASYN helps balance the data so the model can classify both high and low performers accurately. Feature selection further improves the model by focusing on the most important predictors, making the MLP model stronger and more reliable. These steps help the model provide useful insights into student behavior and support strategies to boost academic performance.

5.3 ADASyn Data Generation Result

The generated dataset using Adaptive Synthetic Sampling (ADASYN) has been designed to address the issue of class imbalance by increasing the representation of the minority class or class 0. Initially, the dataset contained 74 samples, with a significant imbalance between 53 class 1 instances and only 21 class 0 instances. This imbalance posed a challenge for MLP modelling, as the classifier could become biased toward the majority class, leading to poor recall for class 0 predictions. Table 5.2 shows the new data set that was generated using the ADASYN method.

Table 5.2
New Dataset Using ADASYN

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 1	0	130	72	99	928	0	86	15	37	8	13	11	0
Data 2	10	88	27	77	0	1	67	10	16	6	10	6	0
Data 3	14	159	115	59	14555	2	117	36	18	17	17	5	0
Data 4	1	87	36	101	0	0	69	10	36	8	10	6	0
Data 5	2	115	34	70	2678	1	73	12	20	9	13	10	0
Data 6	3	99	40	50	0	0	78	16	16	8	17	10	0
Data 7	0	90	45	28	0	0	63	13	12	11	14	7	0
Data 8	13	111	26	0	539	3	75	10	0	15	15	2	0
Data 9	1	80	124	54	0	0	47	29	17	8	12	10	0
Data 10	7	192	112	260	4697	0	111	43	99	9	17	9	0
Data 11	0	127	86	57	0	0	87	22	34	17	17	5	0
Data 12	2	64	30	118	1009	0	51	7	50	1	3	2	0
Data 13	2	74	40	10	1012	0	65	15	2	11	17	12	0
Data 14	0	53	21	36	192	0	38	6	12	10	13	8	0

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 15	10	127	109	115	11781	2	77	53	37	11	17	11	0
Data 16	3	53	16	81	294	1	47	3	39	4	7	4	0
Data 17	5	161	33	85	2711	3	113	12	40	15	17	7	0
Data 18	13	96	37	63	2094	1	70	19	21	8	13	7	0
Data 19	10	190	165	193	24428	6	67	91	66	12	15	12	0
Data 20	0	69	57	55	0	0	50	21	27	6	8	4	0
Data 21	0	68	10	52	0	0	37	2	15	3	6	4	0
Data 22	1	147	92	67	11095	0	59	46	13	16	13	7	1
Data 23	3	111	43	34	4865	0	78	10	7	17	17	6	1
Data 24	6	169	128	146	10109	1	63	70	75	15	17	5	1
Data 25	3	132	99	6	1230	2	67	38	2	17	15	4	1
Data 26	3	85	100	139	10822	2	73	20	29	17	17	2	1
Data 27	18	114	232	9	0	7	89	41	1	9	17	13	1
Data 28	5	179	138	14	8818	1	82	52	11	17	17	3	1
Data 29	9	263	170	178	4603	6	117	85	67	16	17	6	1

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 30	13	184	142	157	33360	3	77	72	74	17	17	6	1
Data 31	8	101	69	103	3715	4	74	12	27	17	17	5	1
Data 32	12	120	105	299	10461	1	67	45	130	16	17	5	1
Data 33	9	128	149	154	8445	4	84	39	34	16	17	4	1
Data 34	5	180	103	30	9411	0	75	33	4	17	8	6	1
Data 35	7	112	74	2	2720	2	60	33	0	16	10	3	1
Data 36	3	53	39	92	0	1	46	15	31	14	17	10	1
Data 37	15	162	122	121	15241	3	92	24	43	15	17	7	1
Data 38	0	105	48	0	112	0	51	14	0	15	17	6	1
Data 39	0	71	58	232	0	0	49	32	80	16	17	6	1
Data 40	0	114	79	57	0	0	72	24	25	12	17	6	1
Data 41	1	189	48	138	1482	0	110	16	51	16	17	5	1
Data 42	17	146	152	77	14863	5	118	39	30	15	17	7	1
Data 43	3	120	27	182	0	0	62	8	73	12	14	7	1
Data 44	11	140	146	162	20146	4	82	41	45	15	17	8	1

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 45	7	110	26	71	0	0	69	9	24	16	17	6	1
Data 46	5	139	140	50	3519	1	80	38	20	16	17	6	1
Data 47	14	196	138	195	5322	7	86	47	70	13	17	8	1
Data 48	22	231	258	140	20807	5	118	65	80	17	17	8	1
Data 49	5	108	110	18	4079	1	47	41	3	13	16	8	1
Data 50	0	52	31	47	0	0	45	11	19	14	17	7	1
Data 51	0	112	31	41	1504	0	67	17	12	17	17	9	1
Data 52	2	132	126	73	14545	0	54	49	29	15	15	7	1
Data 53	7	209	34	14	7947	3	78	25	2	16	13	7	1
Data 54	7	122	110	181	25706	7	93	16	65	17	17	1	1
Data 55	0	40	35	91	0	0	39	10	29	15	17	6	1
Data 56	8	119	45	57	689	1	69	11	16	16	17	5	1
Data 57	12	132	77	104	2483	4	58	36	42	8	13	10	1
Data 58	17	185	173	98	13249	3	93	42	41	15	15	6	1
Data 59	5	103	118	77	2079	0	47	45	19	15	12	10	1

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 60	14	165	67	0	9993	1	82	34	0	15	9	6	1
Data 61	5	162	168	193	6751	0	68	66	53	14	8	6	1
Data 62	6	204	108	141	28975	2	52	55	47	15	17	7	1
Data 63	17	161	119	139	11753	3	98	44	49	13	17	7	1
Data 64	6	76	73	107	1654	2	46	25	51	16	15	6	1
Data 65	61	174	105	65	1113	46	102	27	25	17	17	5	1
Data 66	3	127	71	60	946	0	55	42	27	12	14	7	1
Data 67	12	232	173	113	16650	2	118	44	32	16	16	6	1
Data 68	10	113	105	5	8570	5	53	47	2	16	17	7	1
Data 69	24	82	81	51	0	1	64	13	31	12	17	8	1
Data 70	13	123	147	141	16654	6	92	41	41	14	17	10	1
Data 71	13	126	138	77	4742	7	56	58	38	17	17	4	1
Data 72	9	80	82	123	3974	3	46	33	56	17	17	6	1
Data 73	8	97	54	123	238	0	72	23	50	15	17	4	1
Data 74	14	97	115	79	4272	6	74	31	22	16	17	6	1

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 75	0	97	51	108	967	0	68	11	43	4	8	6	0
Data 76	12	150	113	73	13825	2	106	40	22	15	17	6	0
Data 77	5	160	34	84	2871	2	113	12	39	15	17	6	0
Data 78	0	83	31	92	0	0	63	8	32	7	9	5	0
Data 79	3	111	34	68	2581	1	72	13	20	8	13	9	0
Data 80	2	90	44	51	0	0	69	17	19	7	14	8	0
Data 81	0	81	49	38	0	0	57	16	17	9	11	5	0
Data 82	1	60	21	31	234	0	42	6	10	10	13	7	0
Data 83	1	87	90	52	0	0	59	23	16	8	14	10	0
Data 84	2	80	44	24	1225	0	67	16	7	10	17	11	0
Data 85	6	175	101	241	4219	0	103	38	92	7	15	8	0
Data 86	0	102	73	56	0	0	71	21	30	12	13	4	0
Data 87	6	82	28	70	821	1	60	8	30	6	7	2	0
Data 88	0	113	62	72	952	0	79	15	26	8	14	11	0
Data 89	0	69	31	32	106	0	49	9	12	10	13	7	0

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 90	8	166	110	203	7436	0	97	46	75	9	17	9	0
Data 91	12	144	112	84	13301	2	98	43	26	14	17	7	0
Data 92	2	53	16	75	282	0	45	3	35	4	7	4	0
Data 93	4	160	33	84	2704	2	112	12	39	14	17	7	0
Data 94	10	99	36	64	2212	1	70	17	20	8	13	7	0
Data 95	7	85	38	37	1565	0	67	17	11	9	14	9	0
Data 96	8	190	139	225	14864	3	88	67	81	10	15	10	0
Data 97	8	191	130	236	11724	2	95	60	87	10	16	10	0
Data 98	0	86	46	31	0	0	61	14	14	10	13	6	0
Data 99	0	68	10	53	0	0	37	2	15	3	6	4	0

To overcome this issue, 25 new synthetic class 0 instances were generated using ADASYN, increasing the total dataset size to 99 samples, with a more balanced class distribution. The synthetic data was generated based on the feature space distribution, ensuring that harder-to-classify instances received more synthetic samples. This closer-to-ideal 50:50 class ratio enhances the model's ability to learn meaningful patterns from both classes, reducing bias and improving its ability to correctly classify students who are at risk of lower academic performance. With this balanced dataset, the MLP model is expected to generalize better and achieve higher classification accuracy, precision, and recall, making it more suitable for predictive analysis in student academic performance modelling.

5.4 Synthetic Data Evaluation

Ensuring the reliability and consistency of the synthetic data generated through ADASYN is a crucial step in verifying its suitability for model training. To achieve this, a combination of statistical tests and visual analysis was conducted to compare the properties of the original and augmented datasets. The verification process involved the Kolmogorov-Smirnov test, histogram analysis, and box plot evaluation to ensure that the synthetic data maintained the fundamental statistical characteristics of the original dataset while addressing class imbalance.

To assess distributional differences between the original and synthetic datasets, the K-S test was used to compare their empirical cumulative distribution functions (ECDFs), determining whether ADASYN preserved the overall statistical structure of each feature. Complementing this, histogram analysis visually compared feature distributions before and after augmentation, helping to identify any distortions or unintended shifts in density. Meanwhile, box plot analysis examined interquartile ranges and potential outliers, ensuring that ADASYN's resampling did not introduce extreme variations. Together, these validation techniques confirmed that the synthetic data retained key distributional properties while effectively balancing class representation, ensuring that the MLP classifier could generalize better without being biased toward the majority class.

5.4.1 Kolmogorov-Smirnov Test result

The Kolmogorov-Smirnov (K-S) test results in Table 5.3 provide a comparison between the original and ADASYN datasets. When interpreting these results, a high p-value which greater than 0.05, indicates no significant difference between the distributions of the original and ADASYN data for a given feature, suggesting that the ADASYN method has effectively preserved the feature's distribution. Conversely, a low p-value with the value less than or equal to 0.05, would suggest a significant difference, warranting further examination of that feature.

Table 5.3
KS Test Result

Feature	K-S Statistic	p-value
Feature 1	0.0652	0.9861
Feature 2	0.0730	0.9600
Feature 3	0.0797	0.9233
Feature 4	0.0544	0.9981
Feature 5	0.0622	0.9915
Feature 6	0.0718	0.9658
Feature 7	0.0366	1.0000
Feature 8	0.0861	0.8758
Feature 9	0.0375	1.0000
Feature 10	0.0158	0.2534
Feature 11	0.0895	0.8459
Feature 12	0.0355	1.0000
Output	0.1809	0.1062

In this analysis, the K-S statistics for most features, including Feature1 through Feature 9 and Feature11 through Feature12, are low, with p-values well above 0.05. This indicates that there are no significant differences between the distributions of these features in the original and ADASYN datasets, implying that the ADASYN method has maintained the distributional characteristics of these features well. However, Feature10

and the Output feature have higher K-S statistics (0.1508 and 0.1809, respectively) and lower p-values (0.2534 and 0.1062, respectively). Although these p-values are still above 0.05, they are notably lower than those of the other features, suggesting that the ADASYN method may have introduced some variability in these areas.

Overall, the results suggest that the ADASYN data closely replicates the structure of the original data for most features, with minimal differences. The slight variations observed in Feature10 and Output, while not statistically significant, may indicate that the ADASYN method has introduced some changes, possibly due to the way it generates synthetic data near the decision boundaries of the minority class. Given these findings, the ADASYN data appears to be a valid candidate for use in MLP classification modelling. If the model's performance is consistent or improved with the ADASYN data, it could indicate that the synthetic data has effectively balanced the classes without introducing significant noise.

In conclusion, the K-S test results support the use of ADASYN data for MLP classification modelling, as it closely aligns with the original data for most features. Nonetheless, careful validation is recommended to ensure that any slight differences, particularly in Feature10 and Output, do not adversely affect the model's performance. If validated successfully, the ADASYN data could serve as a robust tool for addressing class imbalance in your models.

5.4.2 Histogram Test result

Following the insights gained from the Kolmogorov-Smirnov (K-S) test, which confirmed that ADASYN effectively maintains the distribution of most features while introducing minor variations in Feature10 and the Output feature, the histogram analysis offers a visual representation of how closely the synthetic data aligns with the original dataset. This method examines the frequency distribution of each feature, providing a deeper understanding of potential shifts, over-representations, or gaps introduced by synthetic data generation. By ensuring that both datasets are pre-processed consistently, with identical feature names, types, and scales, the histogram analysis offers a detailed perspective on whether ADASYN preserves feature integrity or alters the dataset in ways that could impact model performance. The results from this analysis will help determine whether ADASYN successfully balances class distribution while

maintaining the dataset's overall structure, ensuring its reliability for MLP classification modelling.

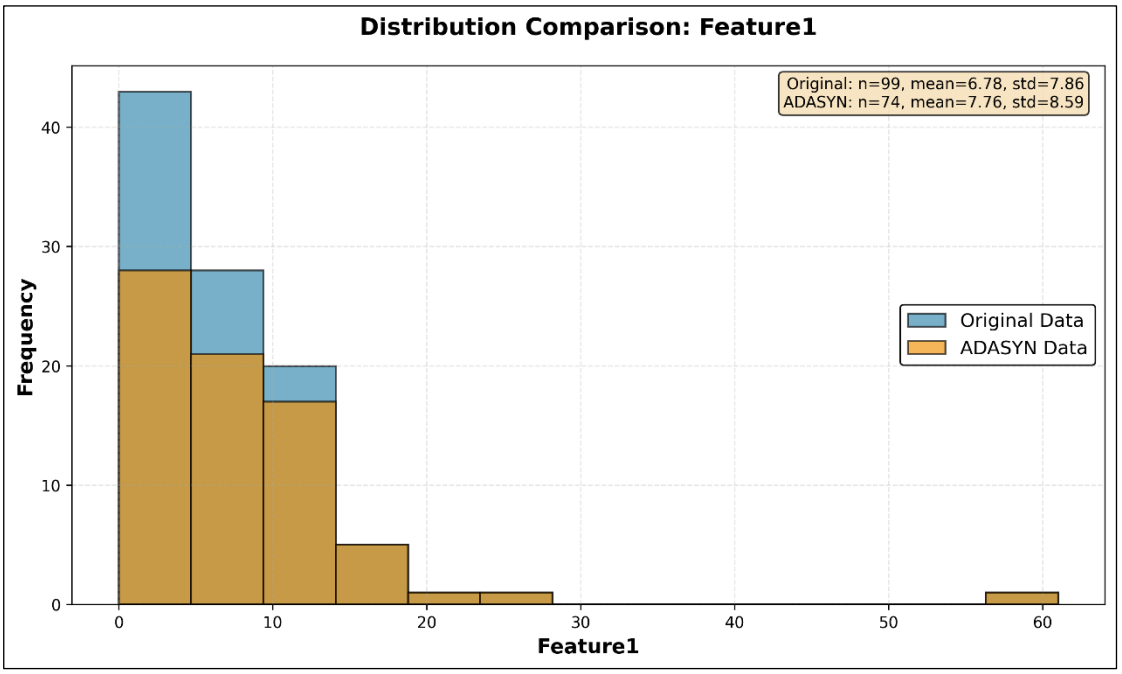


Figure 5.1 Histogram Comparison for Feature 1

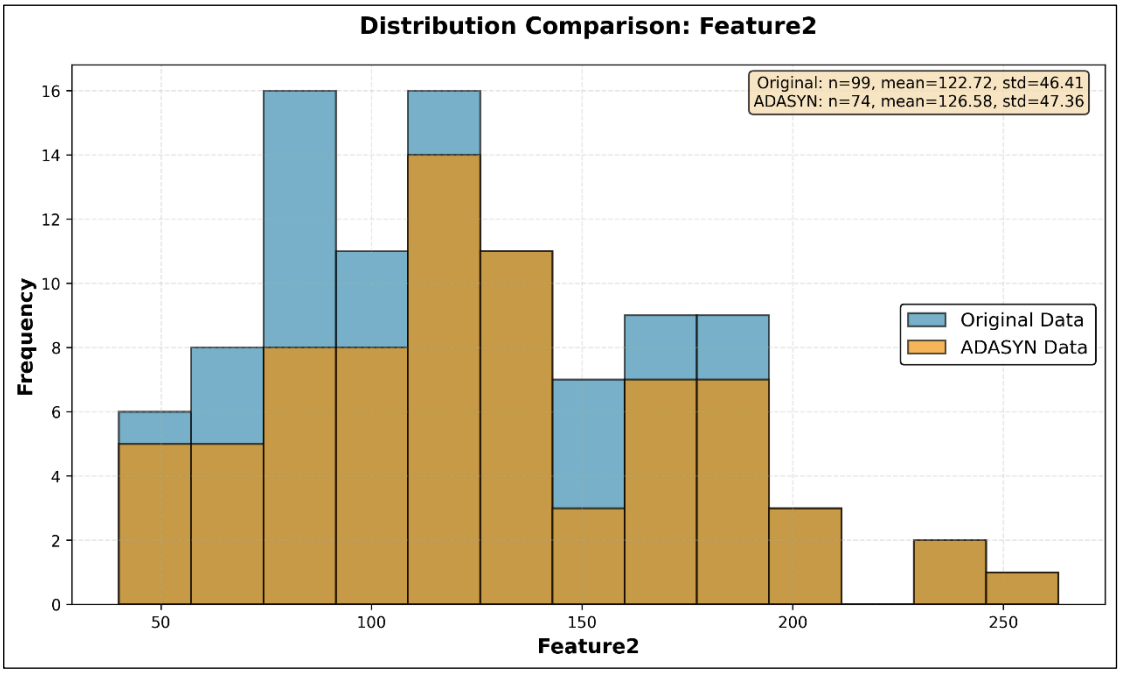


Figure 5.2 Histogram Comparison for Feature 2

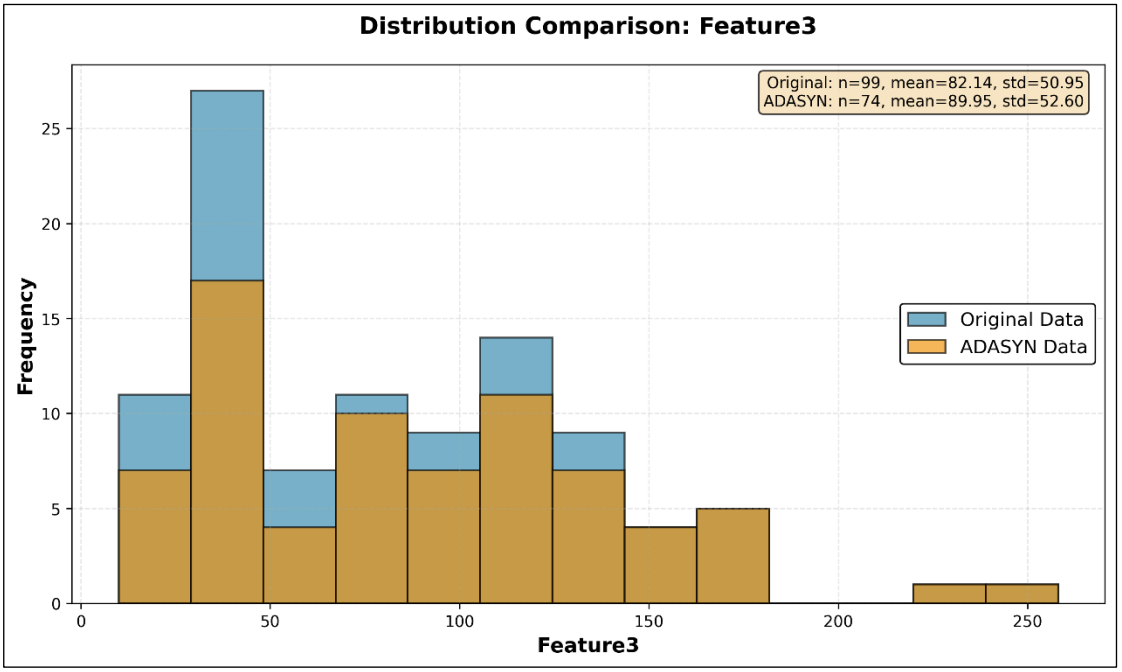


Figure 5.3 Histogram Comparison for Feature 3

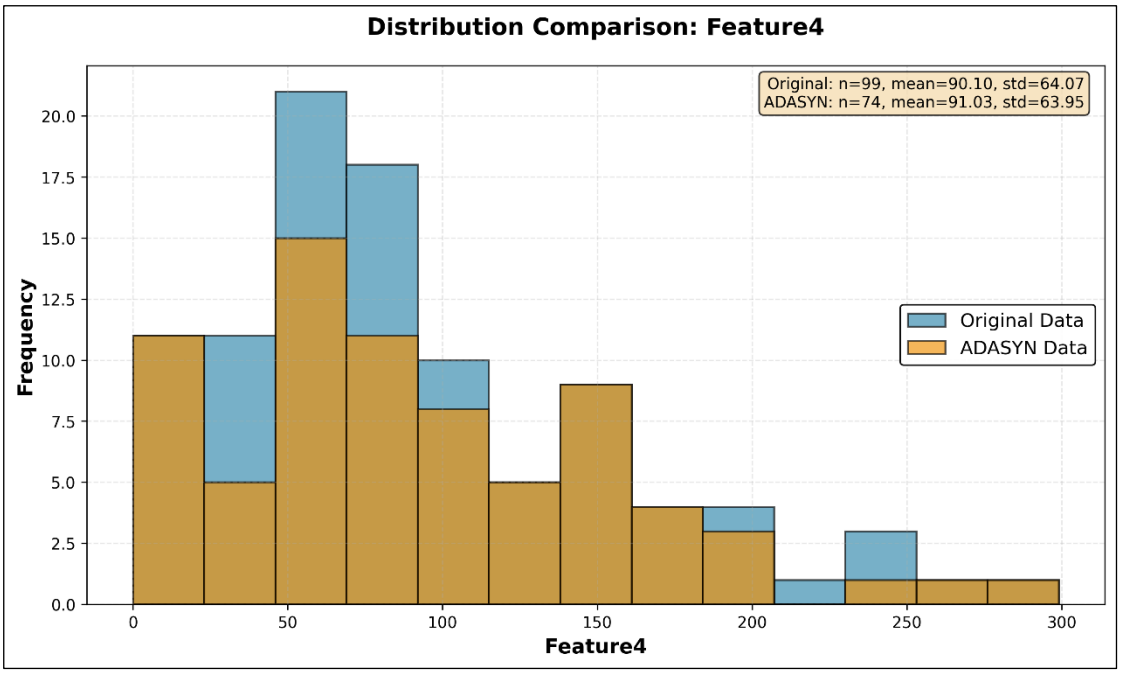


Figure 5.4 Histogram Comparison for Feature 4

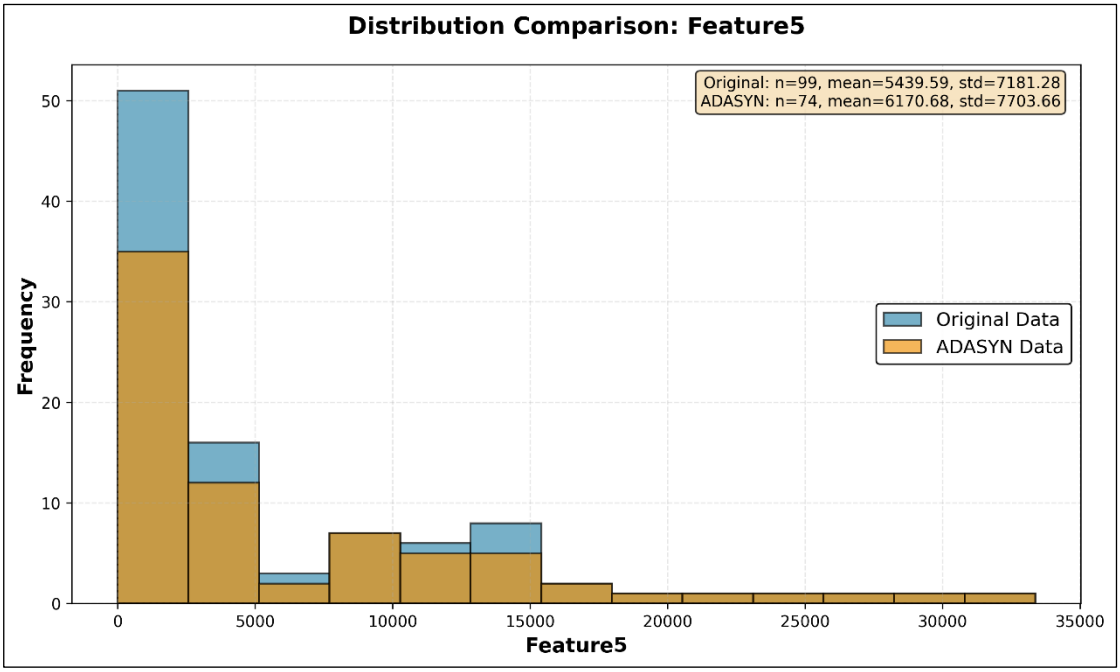


Figure 5.5 Histogram Comparison for Feature 5

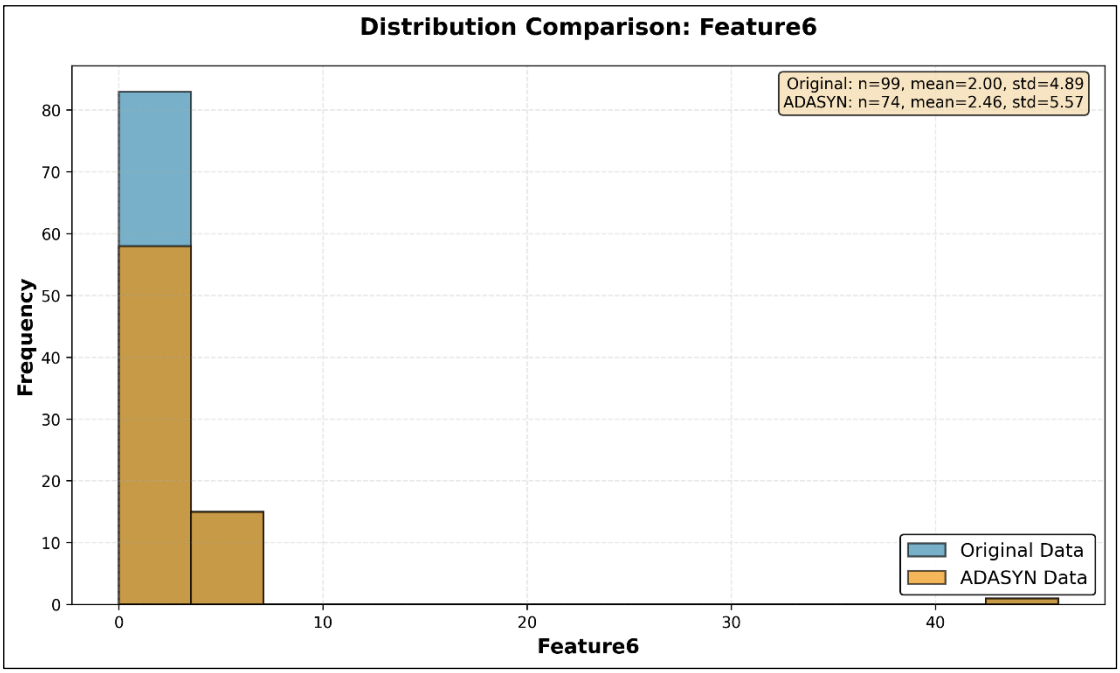


Figure 5.6 Histogram Comparison for Feature 6

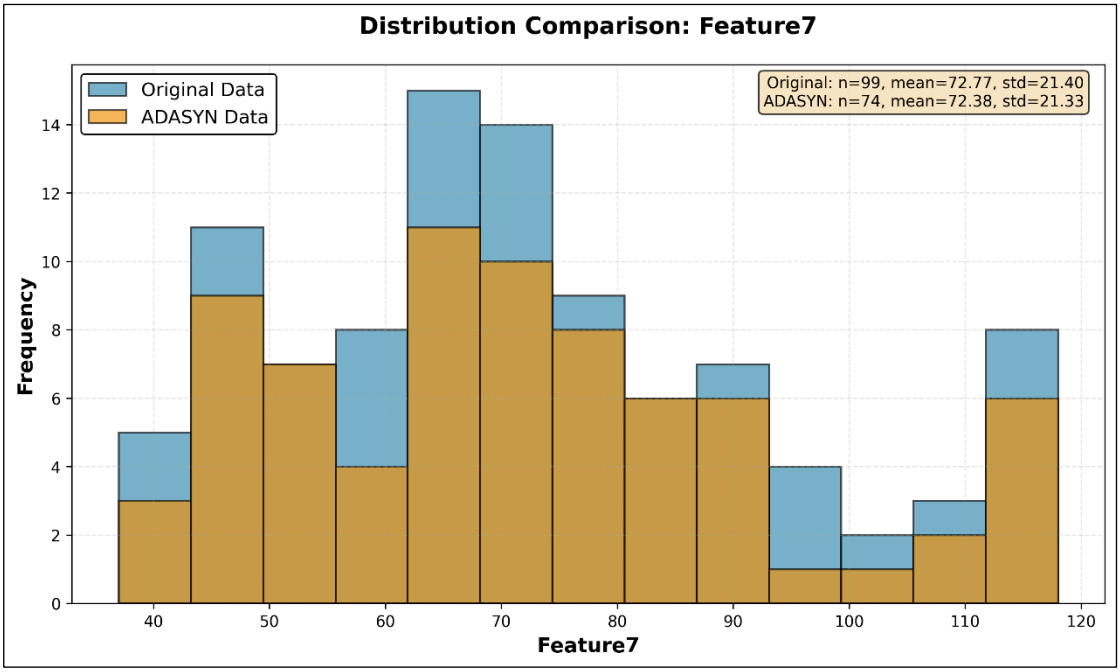


Figure 5.7 Histogram Comparison for Feature 7

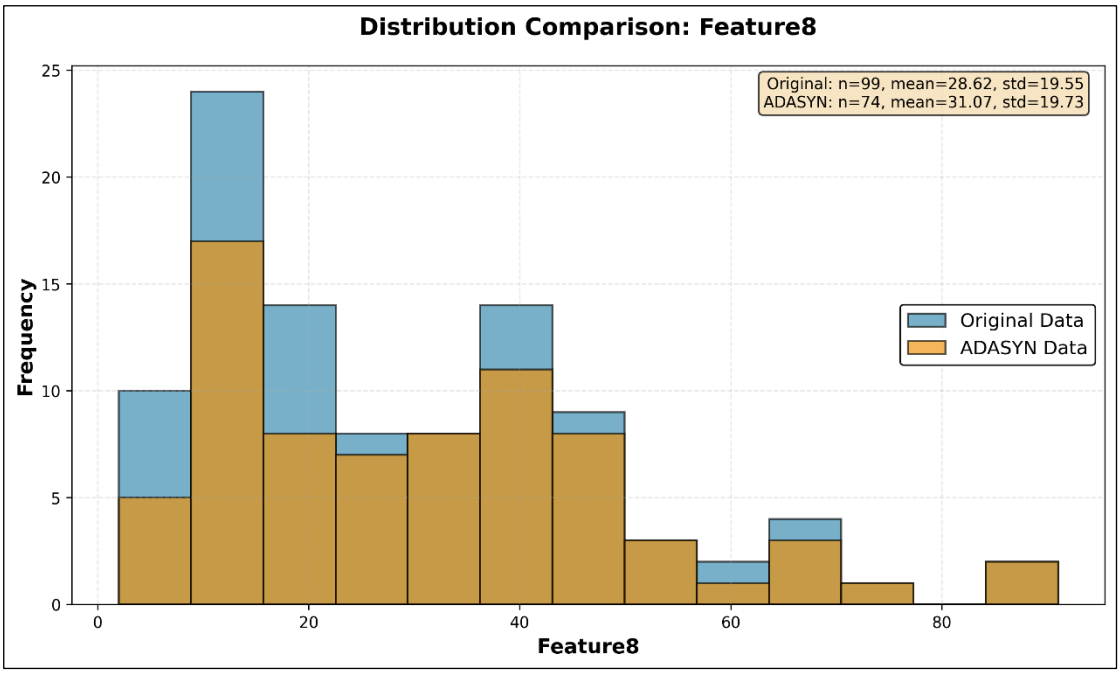


Figure 5.8 Histogram Comparison for Feature 8

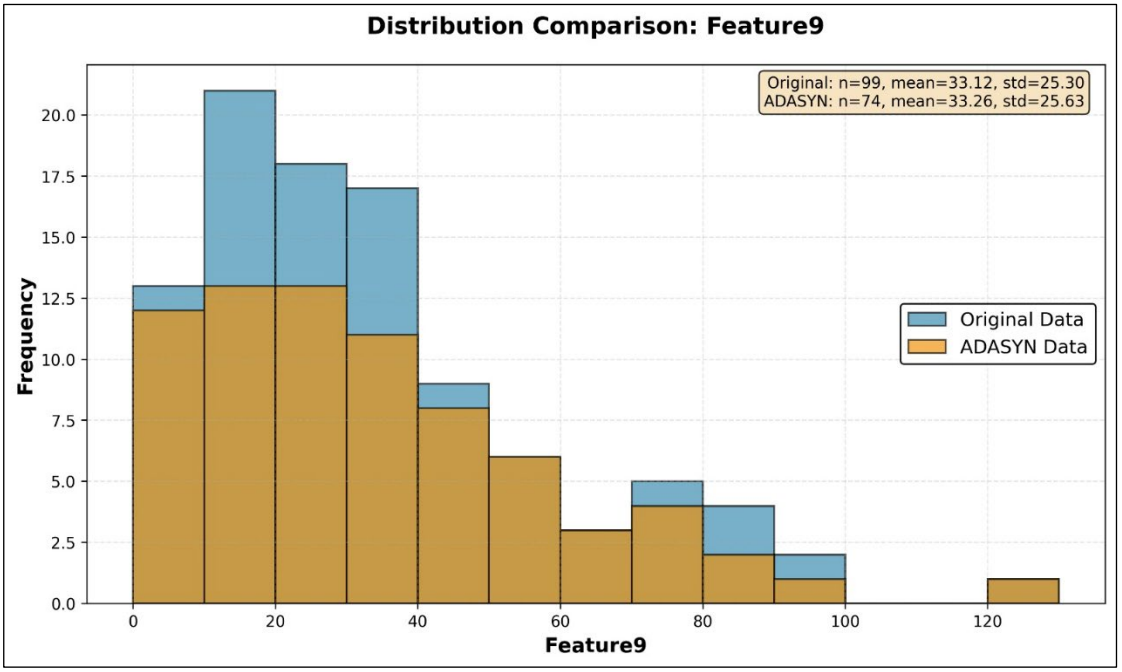


Figure 5.9 Histogram Comparison for Feature 9

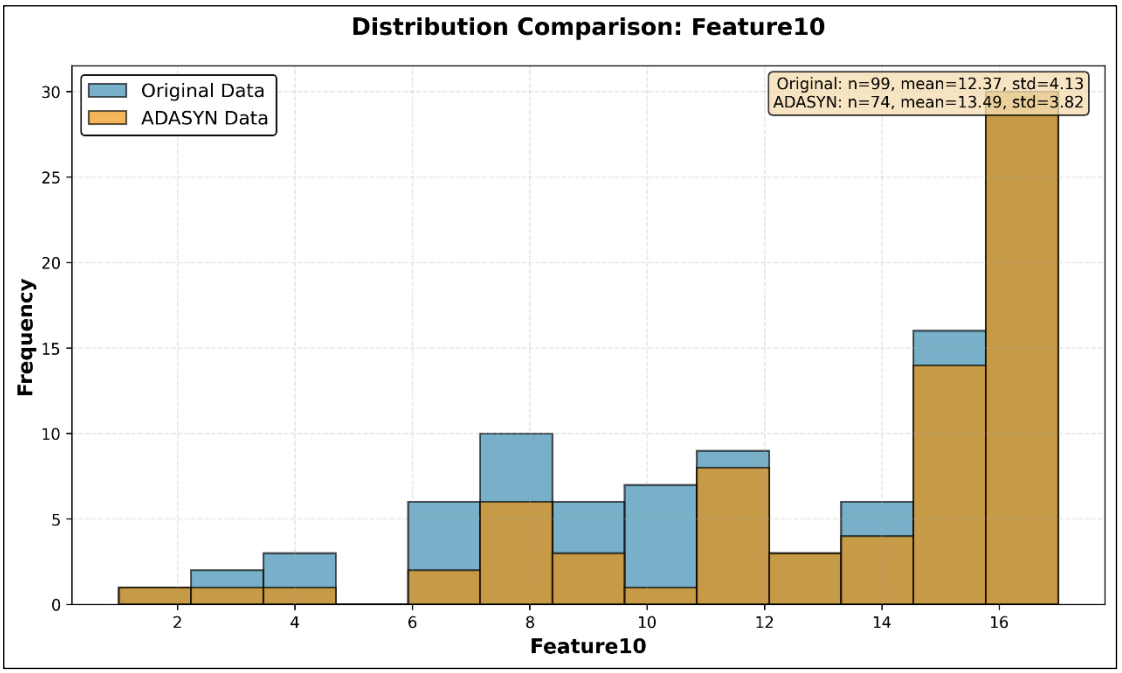


Figure 5.10 Histogram Comparison for Feature 10

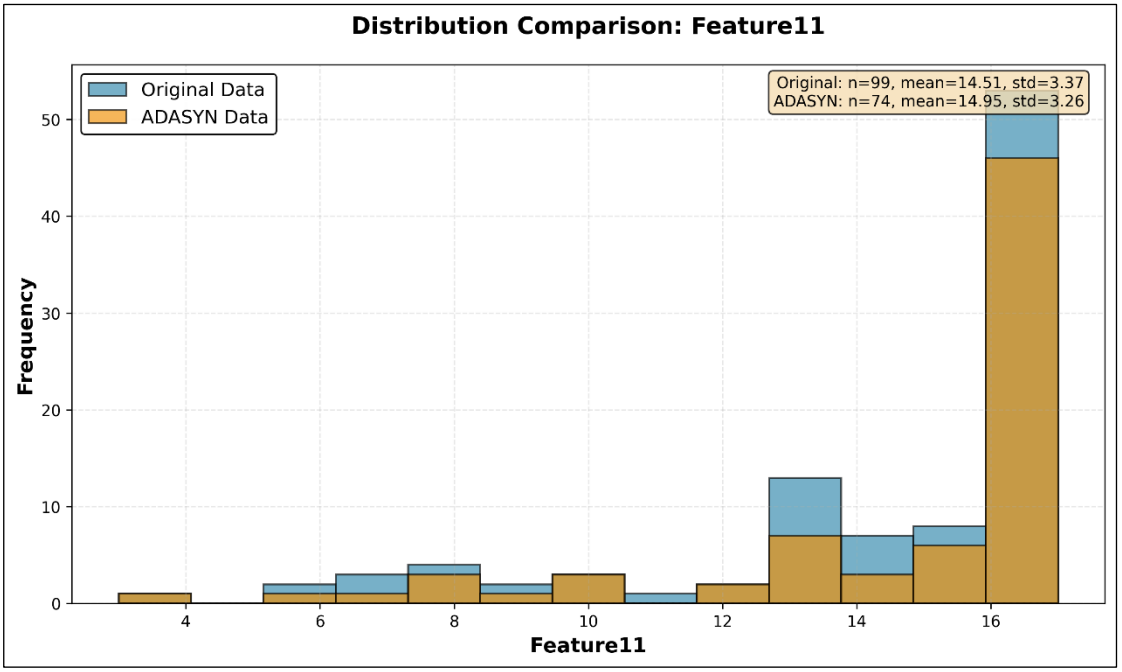


Figure 5.11 Histogram Comparison for Feature 11

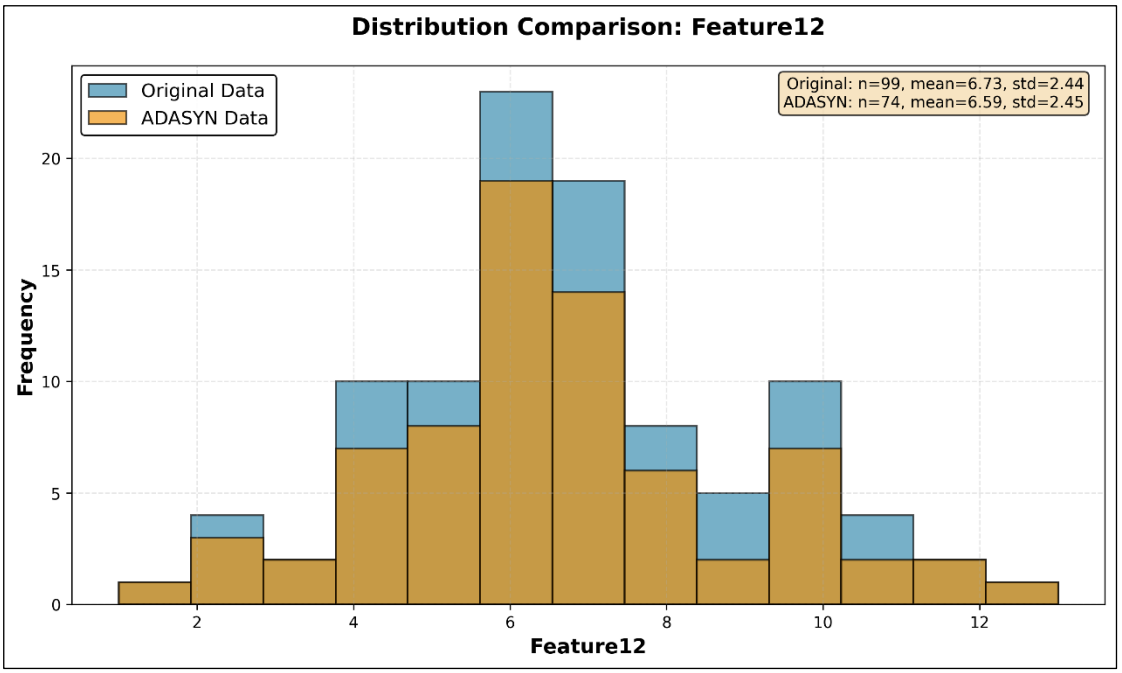


Figure 5.12 Histogram Comparison for Feature 12

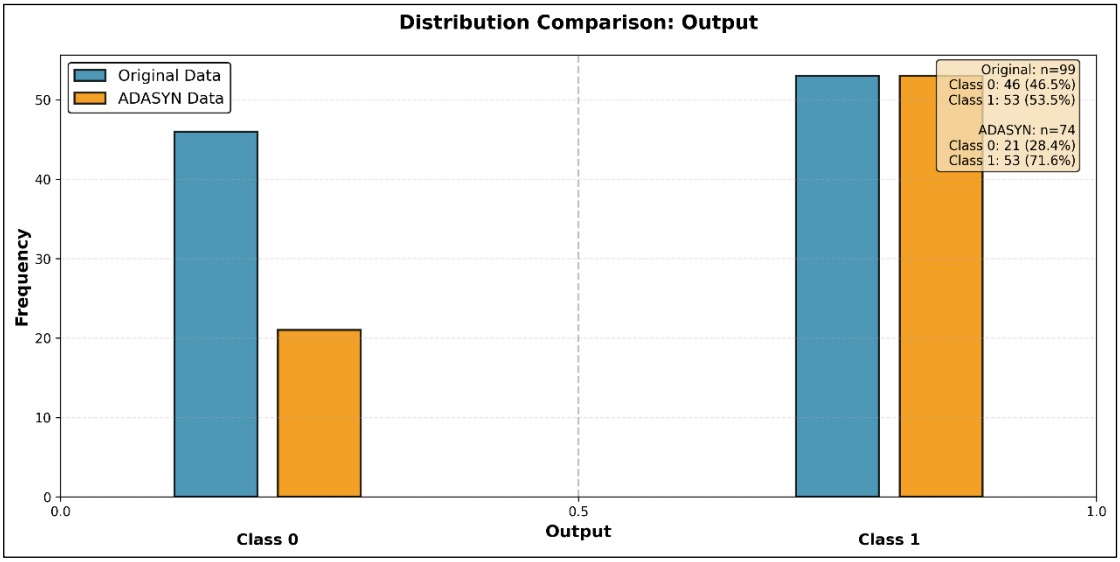


Figure 5.13 Histogram Comparison for Output

The Adaptive Synthetic Sampling (ADASYN) method generates synthetic data points for the minority class to reduce imbalance, which can potentially lead to improved model performance. By analysing the 13 histograms provided, several observations and insights can be drawn regarding the similarity of the distributions, feature-specific impacts, and the validity of using ADASYN-generated data for predictive modeling.

For most features, the ADASYN-generated data generally follows the distribution of the original dataset while introducing slight variations where gaps or sparse occurrences were initially present. This is particularly visible in Feature1, Feature2, and Feature3, where ADASYN fills in areas with low density. In contrast, features that exhibit a more clustered distribution, such as Feature10, Feature11, and the output feature, are closely replicated by ADASYN with minimal deviation from the original pattern. However, for features with a broader range of values, such as Feature5 and Feature6, ADASYN attempts to populate regions with sparse data, which sometimes leads to noticeable over-representation in these areas.

Feature-specific insights reveal that Feature10 and Feature11 show strong clustering near the upper end of the value range (15–16). ADASYN retains this clustering while introducing additional synthetic values within these ranges, suggesting that the method accurately follows the original distribution pattern for tightly clustered features. Conversely, Feature5, which displays a heavily skewed distribution, receives more synthetic data across the lower value ranges (0–5,000). This could potentially over-represent certain regions and lead to model misinterpretation. Additionally, in

some cases, the generated values do not extend as far as the original data, which may affect how the model interprets higher-value occurrences. Similarly, Feature6 exhibits synthetic values in areas where there were few or none in the original dataset, which could introduce bias in the dataset.

With respect to the output feature, the original binary output class (0 or 1) clearly demonstrates an inherent class imbalance. ADASYN effectively increases the frequency of the minority class, ensuring a more balanced dataset. Despite this correction, the synthetic data maintains a clean separation between the two classes, which is crucial for ensuring model reliability in classification tasks. This adjustment in the output feature is expected and necessary for reducing model bias during training and improving classification performance.

Outliers and distribution spread are also important considerations, particularly in Feature5, Feature6, and Feature2. ADASYN attempts to replicate the range and distribution of these features by populating regions that originally contained sparse data. In doing so, it sometimes leads to over-representation in certain areas, which may introduce minor distortions into the dataset and subsequently impact model performance. While this does not necessarily degrade classification accuracy, it is important to assess whether these modifications align with the real-world nature of the data.

While ADASYN effectively emulates or replicate the original data distribution for features with strong clustering, such as Feature10 and Feature11, it introduces distortions in features with broader distributions, such as Feature5 and Feature6. These distortions arise when ADASYN generates synthetic data in regions that were previously underrepresented, potentially leading to a model's over-reliance on artificially introduced data points. Furthermore, the introduction of synthetic values in areas that were initially unpopulated, particularly in Feature1 and Feature6, could introduce noise. If these regions were not well represented in the original dataset, the resulting model could be misled by artificial patterns that do not accurately reflect the true data distribution.

Despite these concerns, ADASYN proves to be an effective method for addressing class imbalance, as demonstrated in the output feature histogram. The method successfully generates a balanced dataset by increasing the frequency of the minority class, thereby reducing model bias. However, it is essential to validate that the

synthetic points generated for the minority class accurately represent the true nature of the data to avoid potential misinterpretations.

Certain features, such as Feature10, Feature11, and Feature12, largely preserve their original distributions after the application of ADASYN. The synthetic data closely follows the original distribution, particularly in regions with strong clustering, indicating that ADASYN performs well for such features without introducing significant deviations. In contrast, features with wider distributions, such as Feature1, Feature5, Feature6, and Feature9, exhibit shifts that alter their original data distributions. These changes flatten skewed distributions and over-represent regions where the original data was sparse, which may lead to unintended consequences in model training.

5.4.3 Box Plot Result

Building upon the histogram test results, which provided insights into how ADASYN preserves the overall distribution of features while introducing slight variations in certain areas, the box plot analysis offers a deeper statistical comparison by visualizing the spread, central tendency, and presence of outliers in both the original and ADASYN-generated datasets. This section examines the median, interquartile range (IQR), and outlier distribution for each feature to determine whether ADASYN has altered the statistical properties of the data in a way that might impact model performance. By comparing the shape and spread of the box plots, we assess whether ADASYN effectively retains the underlying structure of the dataset or introduces inconsistencies, particularly in terms of outlier representation. The results from this analysis help validate the suitability of ADASYN for balancing class distributions while ensuring that its impact on feature characteristics remains minimal and does not negatively affect the integrity of the dataset.

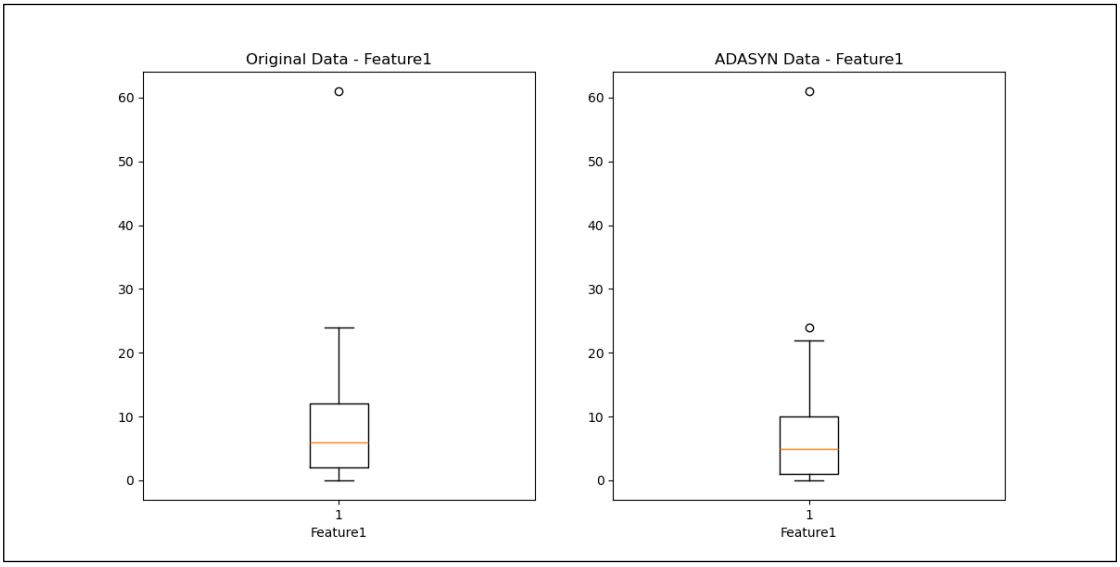


Figure 5.14 Feature 1 Box Plot Comparison

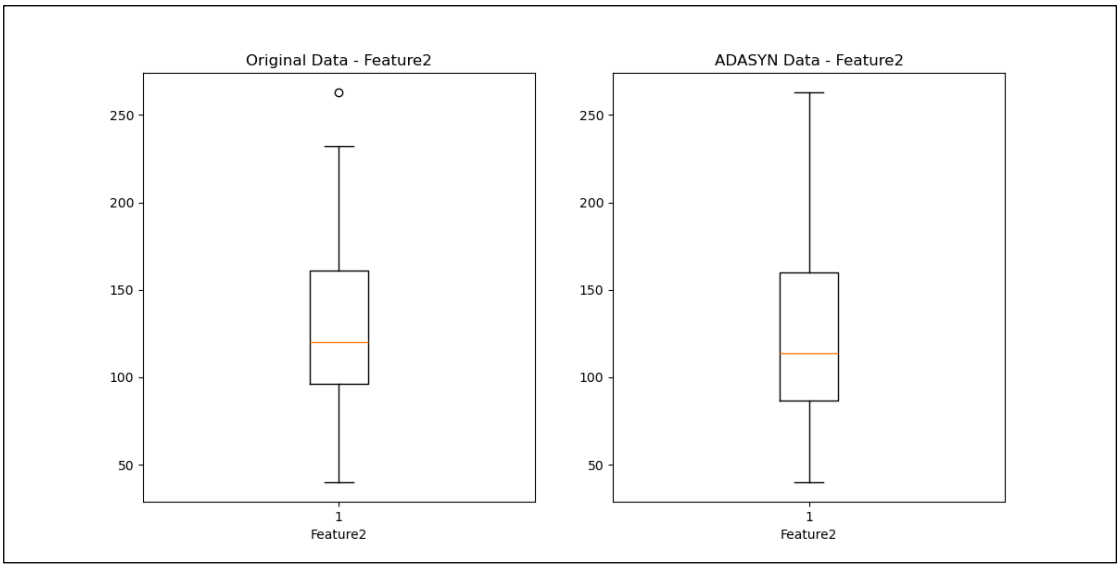


Figure 5.15 Feature 2 Box Plot Comparison

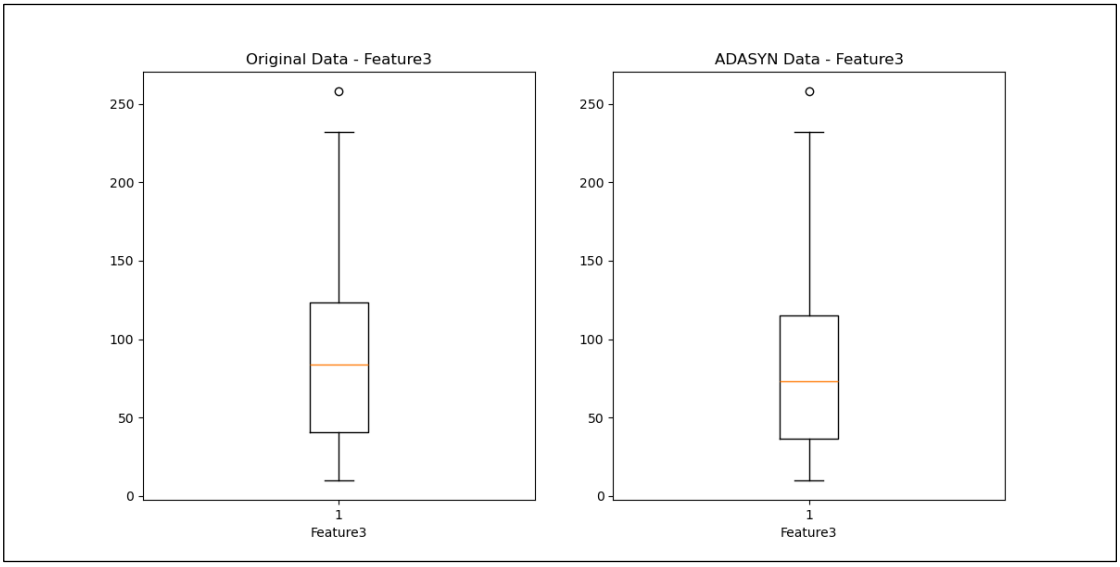


Figure 5.16 Feature 3 Box Plot Comparison

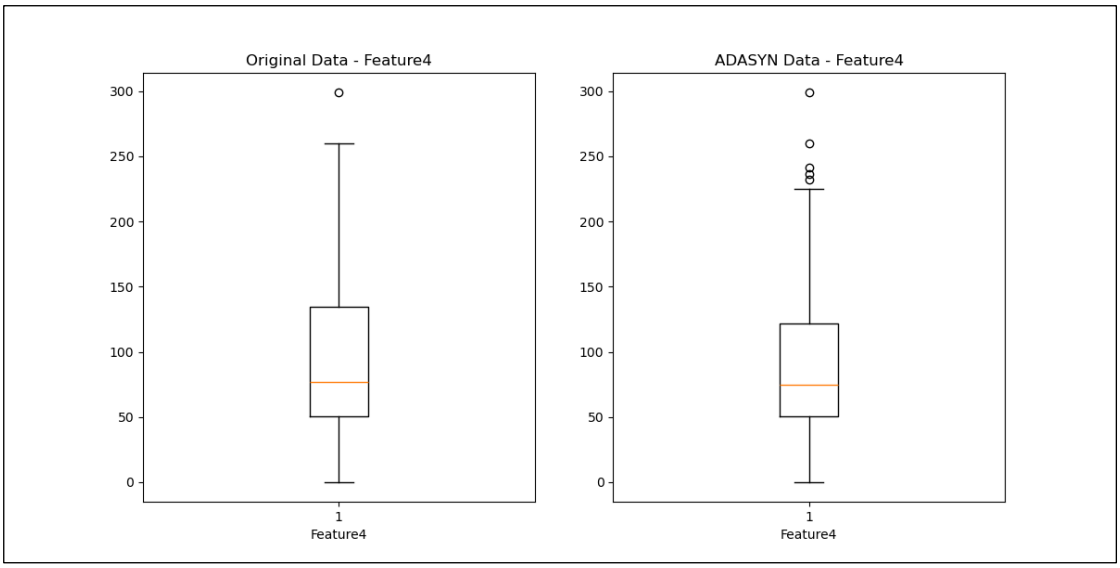


Figure 5.17 Feature 4 Box Plot Comparison

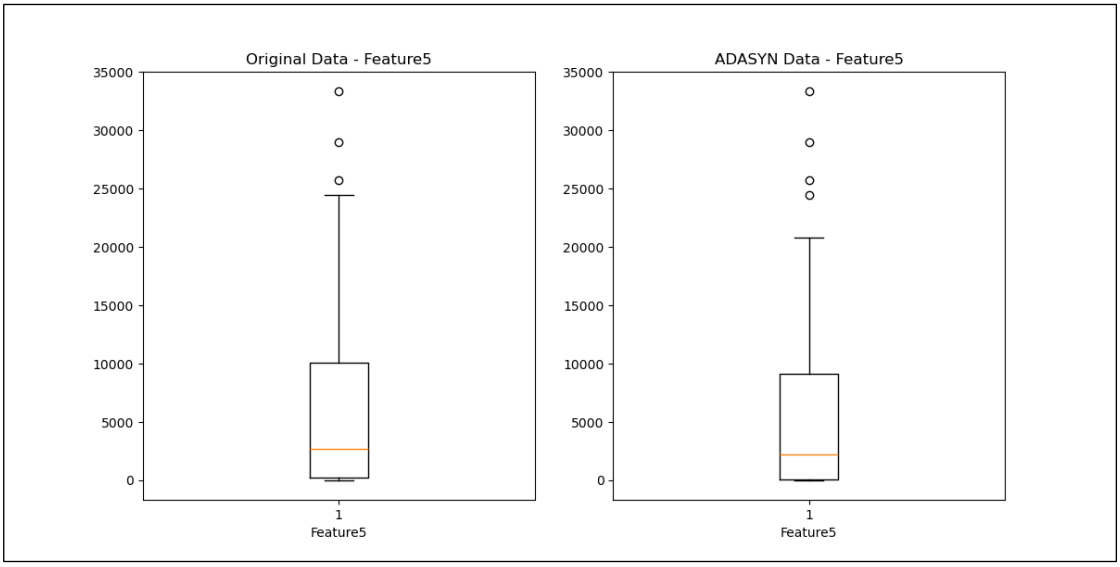


Figure 5.18 Feature 5 Box Plot Comparison

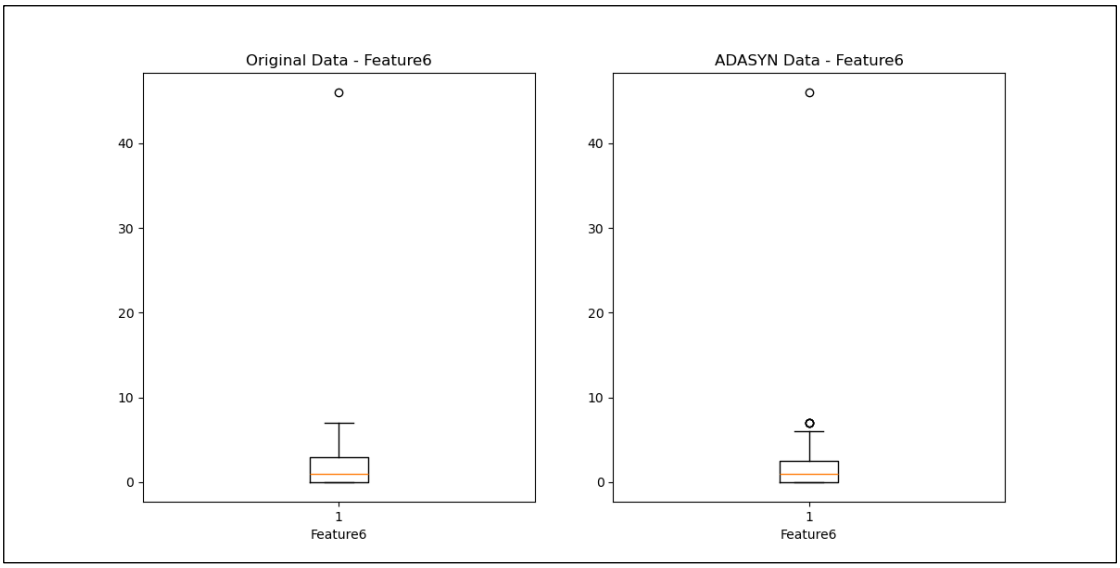


Figure 5.19 Feature 6 Box Plot Comparison

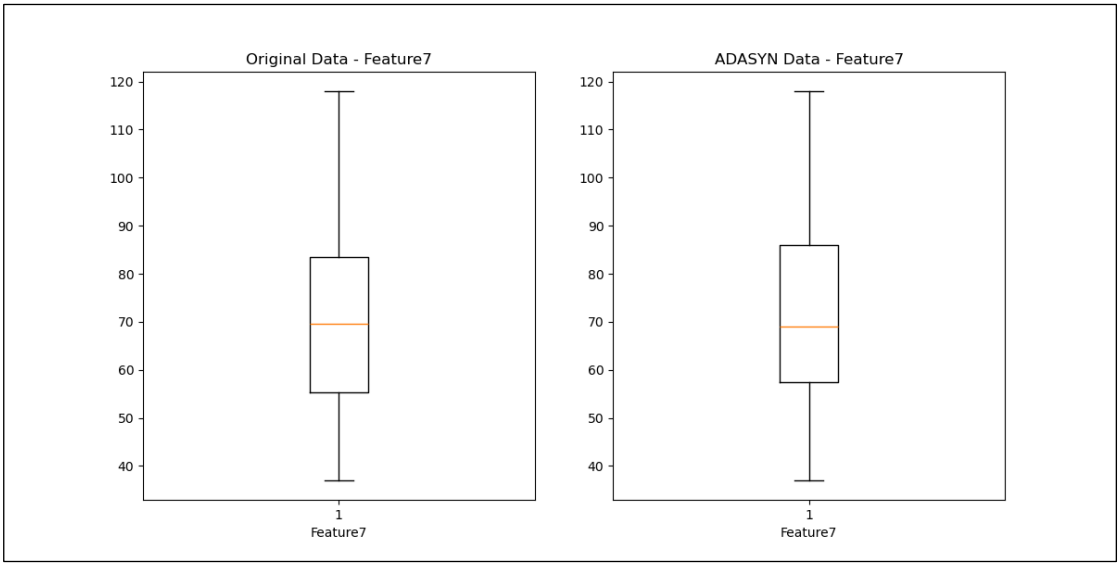


Figure 5.20 Feature 7 Box Plot Comparison

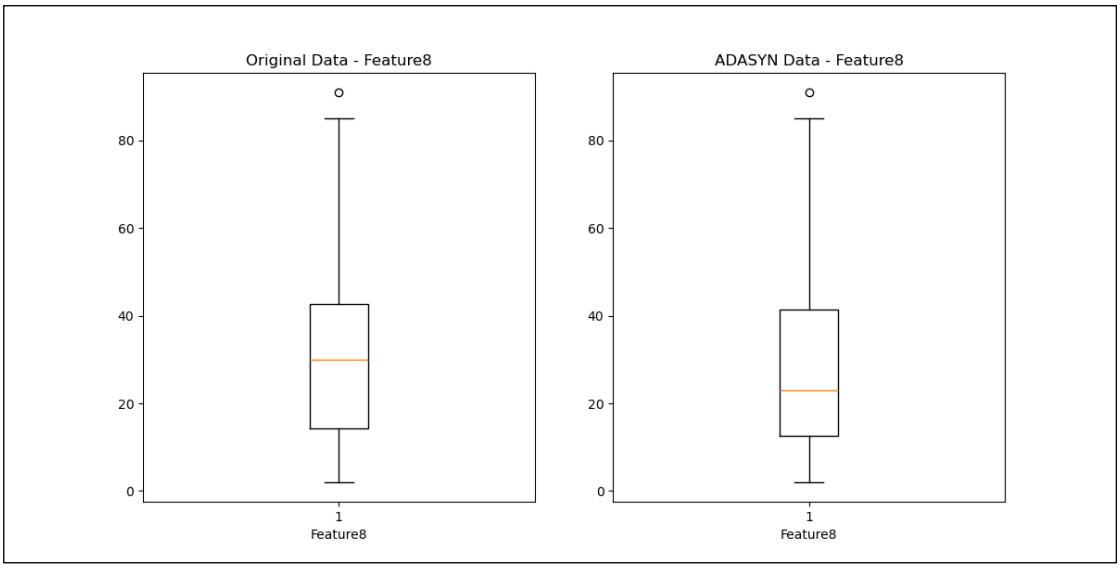


Figure 5.21 Feature 8 Box Plot Comparison

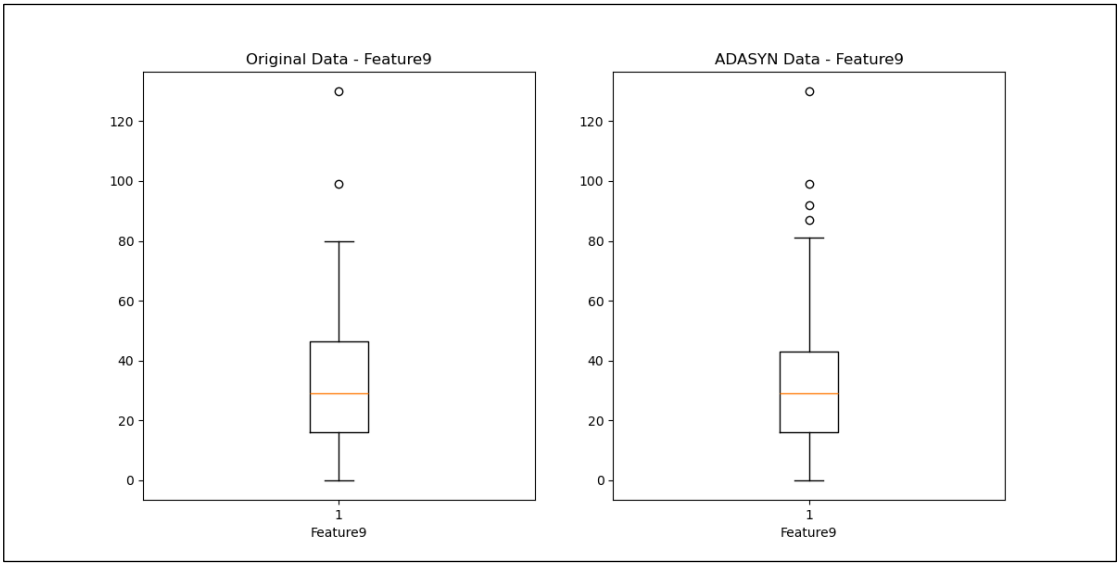


Figure 5.22 Feature 9 Box Plot Comparison

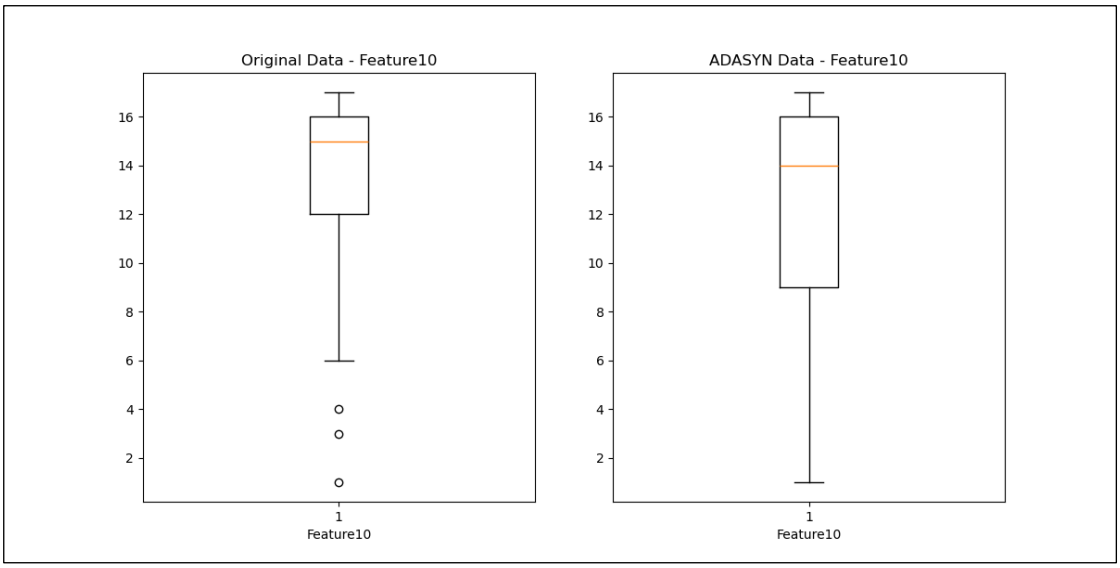


Figure 5.23 Feature 10 Box Plot Comparison

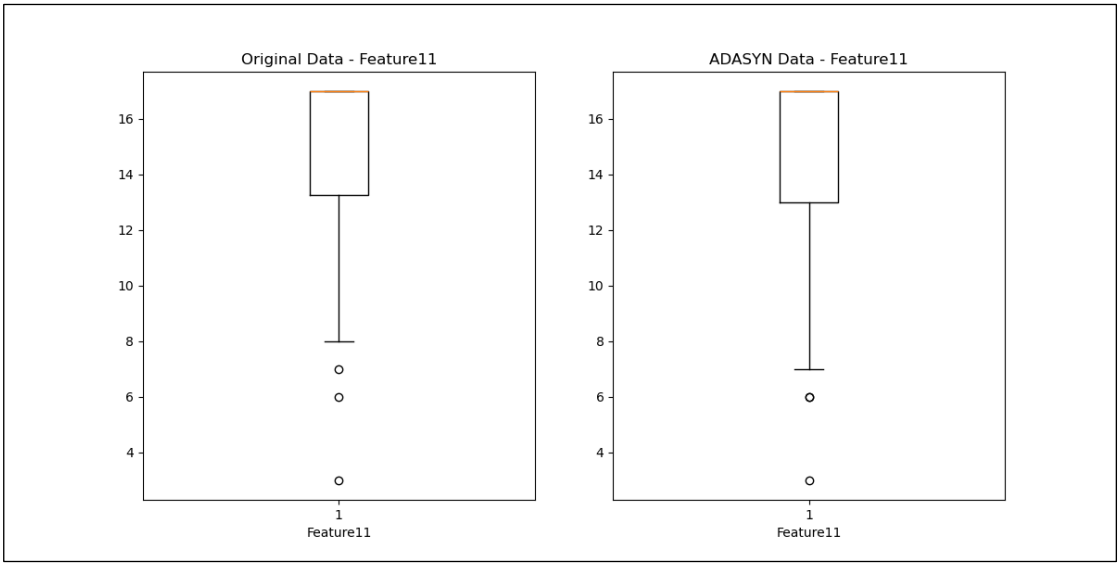


Figure 5.24 Feature 11 Box Plot Comparison

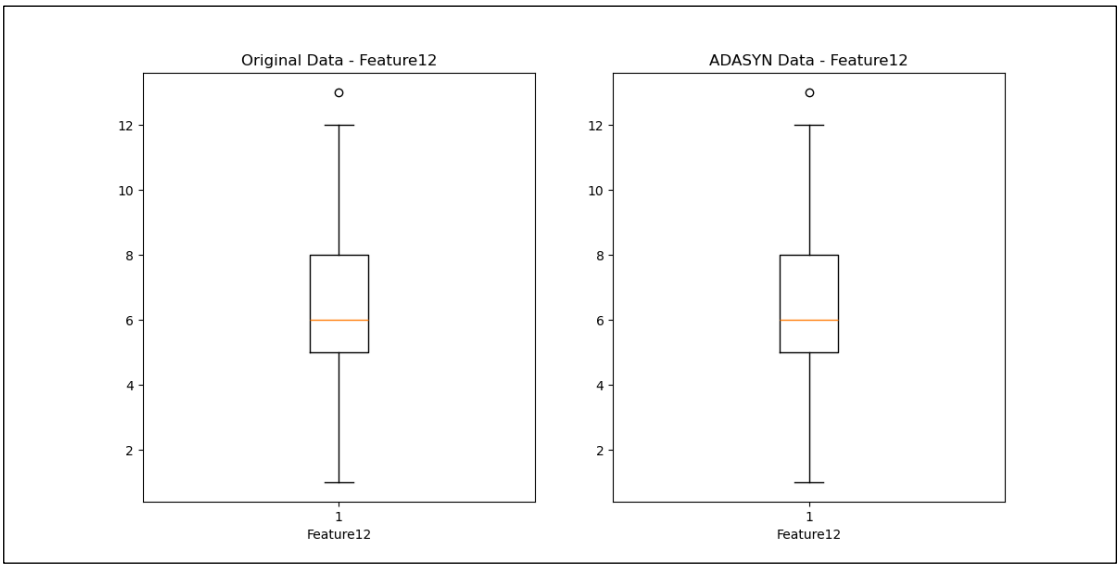


Figure 5.25 Feature 12 Box Plot Comparison

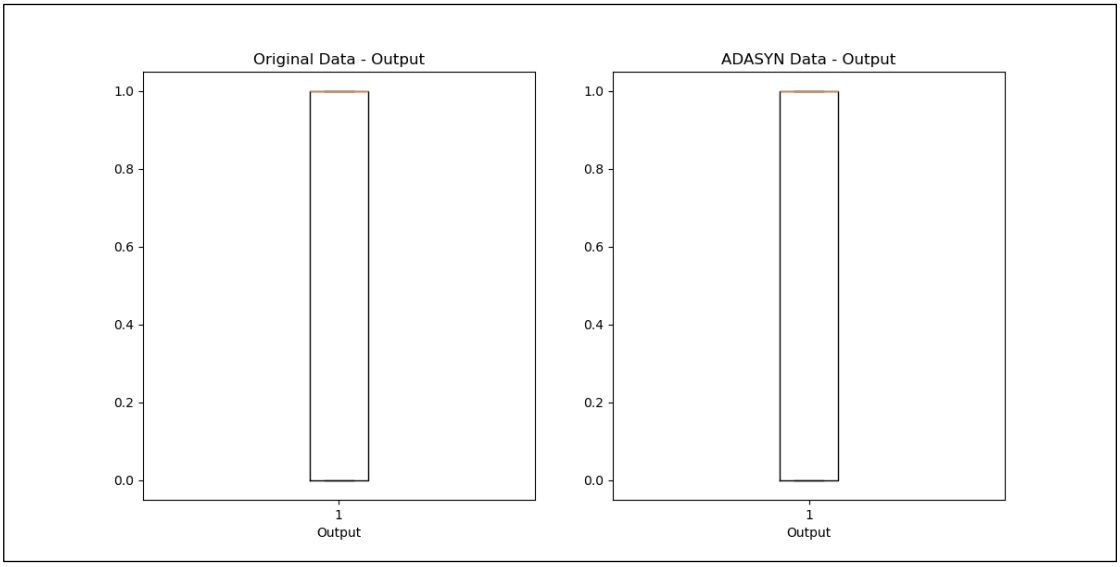


Figure 5.26 Output Box Plot Comparison

The boxplot analysis comparing the original data with the ADASYN-generated data reveals that ADASYN effectively replicates the key statistical properties of the original dataset for most features. The comparison across features shows that the central tendency, measured by the median, remains nearly identical between the original and ADASYN data. Similarly, the interquartile range (IQR), which captures the variability of the data, is consistently preserved across the majority of features. This indicates that ADASYN is generally successful in maintaining the overall structure of the original data, ensuring that the synthetic data does not deviate significantly from the original in terms of core distribution.

For features like Feature 2, Feature 3, Feature 7, and Feature 10, the synthetic ADASYN data closely mirrors the original in terms of median, IQR, and spread. For example, Feature 2 exhibits a median and IQR that align perfectly between the original and ADASYN data, suggesting that the synthetic data accurately captures the distribution of values. In these features, the absence of significant outliers in both the original and ADASYN data further confirms that the synthetic data has not introduced any abnormal variability, maintaining the integrity of the feature's original distribution.

However, some features exhibit slight differences, particularly in the presence of outliers. Features such as Feature 1, Feature 4, Feature 5, and Feature 9 show an increased number of outliers in the ADASYN data compared to the original. For instance, in Feature 4, ADASYN introduces a larger number of outliers at higher values, which were not as prominent in the original dataset. This pattern suggests that while

ADASYN successfully generates synthetic data in line with the central distribution, it may over-represent the extremes in certain cases. This could introduce additional variability in these features, potentially affecting downstream model performance.

Despite these minor variations, the ADASYN-generated data remains consistent with the original data in terms of its fundamental characteristics. Features like Feature 11 and Feature 12, which have strong clustering patterns, show no significant deviation in the synthetic data. The ADASYN method retains the original median and IQR without adding new outliers, demonstrating its capability to handle features with well-defined distributions. Additionally, in the case of the Output feature, which represents the binary target class, ADASYN has effectively balanced the class distribution while maintaining the original class separation.

In conclusion, the boxplot analysis supports the validity of using ADASYN for synthetic data generation, particularly when addressing class imbalance. For most features, the synthetic data aligns well with the original in terms of median, variability, and overall distribution. Although there are some additional outliers introduced in specific features, these deviations are minimal and unlikely to significantly distort the overall dataset. Therefore, ADASYN provides a reliable approach for generating synthetic data that preserves the core statistical characteristics of the original dataset while addressing imbalances in the class distribution.

5.4.4 Synthetic Data Evaluation Analysis

The Kolmogorov–Smirnov test results confirm that the ADASYN-generated synthetic data maintains high statistical similarity with the original dataset. Most features yielded p-values well above the 0.05 threshold such as Feature1 ($p = 0.9861$) and Feature4 ($p = 0.9981$), indicating no significant distributional differences. Even Feature10 and the Output class, which showed relatively higher K-S statistics (0.1508 and 0.1809), still had p-values of 0.2534 and 0.1062 respectively, suggesting acceptable alignment. This shows that ADASYN did not distort the overall feature distributions and preserved the empirical distribution functions (ECDFs) across the dataset.

These results demonstrate that ADASYN preserves global statistical fidelity, even while generating synthetic samples near decision boundaries. Critical predictors like Feature10 and Feature11 retain their original statistical behaviour, ensuring model training is based on representative data. The absence of significant deviations across all

features validates the synthetic dataset as a reliable extension of the original, suitable for use in MLP-based classification models without introducing bias or noise.

Building upon this statistical evidence, the histogram analysis provides visual confirmation that ADASYN maintains local frequency structures and distribution continuity between the original and synthetic datasets. Tightly clustered features like Feature10 and Feature11 exhibit nearly identical peak densities, suggesting the synthetic samples were generated in alignment with real data patterns. For more dispersed features such as Feature5 and Feature6, ADASYN fills underrepresented value ranges, thereby enhancing coverage without introducing abrupt or unnatural distribution shifts.

Although minor local over-representation was noted in certain regions, particularly lower-value ranges of Feature5. These augmentations appear to bridge data sparsity rather than distort existing trends. This balance ensures the synthetic dataset preserves local pattern continuity, making it robust for modeling tasks. Overall, ADASYN effectively replicates the fine-grained structure of the original data, supporting its validity for generalizable and unbiased machine learning applications.

Complementing the findings from the K-S test and histogram analysis, the box plot analysis emphasizes that ADASYN preserves the central structure and variability of the original dataset. Key features like Feature2, Feature3, Feature7, and Feature10 maintain consistent medians and interquartile ranges (IQRs), indicating that the synthetic data aligns well with the central tendencies of the real data. This consistency confirms that ADASYN does not distort the core statistical structure necessary for reliable modeling.

Moreover, naturally occurring outliers, particularly in Feature4 and Feature5 are largely retained, with only slight increases in high-end values. These added outliers are likely due to ADASYN generating samples near minority class boundaries in sparse regions. Rather than introducing noise, this behaviour reflects the algorithm’s intent to enhance minority representation while respecting the dataset's distributional integrity. This structural consistency, reinforced by the box plot observations, validates the synthetic data’s suitability for further machine learning tasks.

5.5 Min-Max Normalization Results

Min-Max Normalization was applied to the dataset to standardize feature values within the range of 0 to 1, ensuring that all features contribute equally to subsequent analyses. Table 5.4 present the data post min-max normalization. This transformation is particularly important when dealing with features that have significantly different numerical ranges, as seen in Feature 2 and Feature 5, which initially exhibited large values compared to other features. Min-Max scaling maintains the original distribution and relationships among data points while effectively reducing the numerical dominance of high-valued features.

Table 5.4
Data Post Min- Max Normalization

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 1	0.0000	0.4036	0.2500	0.3311	0.0278	0.0000	0.6049	0.1461	0.2846	0.4375	0.7143	0.8333	0
Data 2	0.1639	0.2152	0.0685	0.2575	0.0000	0.0217	0.3704	0.0899	0.1231	0.3125	0.5000	0.4167	0
Data 3	0.2295	0.5336	0.4234	0.1973	0.4363	0.0435	0.9877	0.3820	0.1385	1.0000	1.0000	0.3333	0
Data 4	0.0164	0.2108	0.1048	0.3378	0.0000	0.0000	0.3951	0.0899	0.2769	0.4375	0.5000	0.4167	0
Data 5	0.0328	0.3363	0.0968	0.2341	0.0803	0.0217	0.4444	0.1124	0.1538	0.5000	0.7143	0.7500	0
Data 6	0.0492	0.2646	0.1210	0.1672	0.0000	0.0000	0.5062	0.1573	0.1231	0.4375	1.0000	0.7500	0
Data 7	0.0000	0.2242	0.1411	0.0936	0.0000	0.0000	0.3210	0.1236	0.0923	0.6250	0.7857	0.5000	0
Data 8	0.2131	0.3184	0.0645	0.0000	0.0162	0.0652	0.4691	0.0899	0.0000	0.8750	0.8571	0.0833	0
Data 9	0.0164	0.1794	0.4597	0.1806	0.0000	0.0000	0.1235	0.3034	0.1308	0.4375	0.6429	0.7500	0
Data 10	0.1148	0.6816	0.4113	0.8696	0.1408	0.0000	0.9136	0.4607	0.7615	0.5000	1.0000	0.6667	0
Data 11	0.0000	0.3901	0.3065	0.1906	0.0000	0.0000	0.6173	0.2247	0.2615	1.0000	1.0000	0.3333	0
Data 12	0.0328	0.1076	0.0806	0.3946	0.0302	0.0000	0.1728	0.0562	0.3846	0.0000	0.0000	0.0833	0
Data 13	0.0328	0.1525	0.1210	0.0334	0.0303	0.0000	0.3457	0.1461	0.0154	0.6250	1.0000	0.9167	0
Data 14	0.0000	0.0583	0.0444	0.1204	0.0058	0.0000	0.0123	0.0449	0.0923	0.5625	0.7143	0.5833	0
Data 15	0.1639	0.3901	0.3992	0.3846	0.3531	0.0435	0.4938	0.5730	0.2846	0.6250	1.0000	0.8333	0

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 16	0.0492	0.0583	0.0242	0.2709	0.0088	0.0217	0.1235	0.0112	0.3000	0.1875	0.2857	0.2500	0
Data 17	0.0820	0.5426	0.0927	0.2843	0.0813	0.0652	0.9383	0.1124	0.3077	0.8750	1.0000	0.5000	0
Data 18	0.2131	0.2511	0.1089	0.2107	0.0628	0.0217	0.4074	0.1910	0.1615	0.4375	0.7143	0.5000	0
Data 19	0.1639	0.6726	0.6250	0.6455	0.7323	0.1304	0.3704	1.0000	0.5077	0.6875	0.8571	0.9167	0
Data 20	0.0000	0.1300	0.1895	0.1839	0.0000	0.0000	0.1605	0.2135	0.2077	0.3125	0.3571	0.2500	0
Data 21	0.0000	0.1256	0.0000	0.1739	0.0000	0.0000	0.0000	0.0000	0.1154	0.1250	0.2143	0.2500	0
Data 22	0.0164	0.4798	0.3306	0.2241	0.3326	0.0000	0.2716	0.4944	0.1000	0.9375	0.7143	0.5000	1
Data 23	0.0492	0.3184	0.1331	0.1137	0.1458	0.0000	0.5062	0.0899	0.0538	1.0000	1.0000	0.4167	1
Data 24	0.0984	0.5785	0.4758	0.4883	0.3030	0.0217	0.3210	0.7640	0.5769	0.8750	1.0000	0.3333	1
Data 25	0.0492	0.4126	0.3589	0.0201	0.0369	0.0435	0.3704	0.4045	0.0154	1.0000	0.8571	0.2500	1
Data 26	0.0492	0.2018	0.3629	0.4649	0.3244	0.0435	0.4444	0.2022	0.2231	1.0000	1.0000	0.0833	1
Data 27	0.2951	0.3318	0.8952	0.0301	0.0000	0.1522	0.6420	0.4382	0.0077	0.5000	1.0000	1.0000	1
Data 28	0.0820	0.6233	0.5161	0.0468	0.2643	0.0217	0.5556	0.5618	0.0846	1.0000	1.0000	0.1667	1
Data 29	0.1475	1.0000	0.6452	0.5953	0.1380	0.1304	0.9877	0.9326	0.5154	0.9375	1.0000	0.4167	1
Data 30	0.2131	0.6457	0.5323	0.5251	1.0000	0.0652	0.4938	0.7865	0.5692	1.0000	1.0000	0.4167	1
Data 31	0.1311	0.2735	0.2379	0.3445	0.1114	0.0870	0.4568	0.1124	0.2077	1.0000	1.0000	0.3333	1

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 32	0.1967	0.3587	0.3831	1.0000	0.3136	0.0217	0.3704	0.4831	1.0000	0.9375	1.0000	0.3333	1
Data 33	0.1475	0.3946	0.5605	0.5151	0.2531	0.0870	0.5802	0.4157	0.2615	0.9375	1.0000	0.2500	1
Data 34	0.0820	0.6278	0.3750	0.1003	0.2821	0.0000	0.4691	0.3483	0.0308	1.0000	0.3571	0.4167	1
Data 35	0.1148	0.3229	0.2581	0.0067	0.0815	0.0435	0.2840	0.3483	0.0000	0.9375	0.5000	0.1667	1
Data 36	0.0492	0.0583	0.1169	0.3077	0.0000	0.0217	0.1111	0.1461	0.2385	0.8125	1.0000	0.7500	1
Data 37	0.2459	0.5471	0.4516	0.4047	0.4569	0.0652	0.6790	0.2472	0.3308	0.8750	1.0000	0.5000	1
Data 38	0.0000	0.2915	0.1532	0.0000	0.0034	0.0000	0.1728	0.1348	0.0000	0.8750	1.0000	0.4167	1
Data 39	0.0000	0.1390	0.1935	0.7759	0.0000	0.0000	0.1481	0.3371	0.6154	0.9375	1.0000	0.4167	1
Data 40	0.0000	0.3318	0.2782	0.1906	0.0000	0.0000	0.4321	0.2472	0.1923	0.6875	1.0000	0.4167	1
Data 41	0.0164	0.6682	0.1532	0.4615	0.0444	0.0000	0.9012	0.1573	0.3923	0.9375	1.0000	0.3333	1
Data 42	0.2787	0.4753	0.5726	0.2575	0.4455	0.1087	1.0000	0.4157	0.2308	0.8750	1.0000	0.5000	1
Data 43	0.0492	0.3587	0.0685	0.6087	0.0000	0.0000	0.3086	0.0674	0.5615	0.6875	0.7857	0.5000	1
Data 44	0.1803	0.4484	0.5484	0.5418	0.6039	0.0870	0.5556	0.4382	0.3462	0.8750	1.0000	0.5833	1
Data 45	0.1148	0.3139	0.0645	0.2375	0.0000	0.0000	0.3951	0.0787	0.1846	0.9375	1.0000	0.4167	1
Data 46	0.0820	0.4439	0.5242	0.1672	0.1055	0.0217	0.5309	0.4045	0.1538	0.9375	1.0000	0.4167	1
Data 47	0.2295	0.6996	0.5161	0.6522	0.1595	0.1522	0.6049	0.5056	0.5385	0.7500	1.0000	0.5833	1

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 48	0.3607	0.8565	1.0000	0.4682	0.6237	0.1087	1.0000	0.7079	0.6154	1.0000	1.0000	0.5833	1
Data 49	0.0820	0.3049	0.4032	0.0602	0.1223	0.0217	0.1235	0.4382	0.0231	0.7500	0.9286	0.5833	1
Data 50	0.0000	0.0538	0.0847	0.1572	0.0000	0.0000	0.0988	0.1011	0.1462	0.8125	1.0000	0.5000	1
Data 51	0.0000	0.3229	0.0847	0.1371	0.0451	0.0000	0.3704	0.1685	0.0923	1.0000	1.0000	0.6667	1
Data 52	0.0328	0.4126	0.4677	0.2441	0.4360	0.0000	0.2099	0.5281	0.2231	0.8750	0.8571	0.5000	1
Data 53	0.1148	0.7578	0.0968	0.0468	0.2382	0.0652	0.5062	0.2584	0.0154	0.9375	0.7143	0.5000	1
Data 54	0.1148	0.3677	0.4032	0.6054	0.7706	0.1522	0.6914	0.1573	0.5000	1.0000	1.0000	0.0000	1
Data 55	0.0000	0.0000	0.1008	0.3043	0.0000	0.0000	0.0247	0.0899	0.2231	0.8750	1.0000	0.4167	1
Data 56	0.1311	0.3543	0.1411	0.1906	0.0207	0.0217	0.3951	0.1011	0.1231	0.9375	1.0000	0.3333	1
Data 57	0.1967	0.4126	0.2702	0.3478	0.0744	0.0870	0.2593	0.3820	0.3231	0.4375	0.7143	0.7500	1
Data 58	0.2787	0.6502	0.6573	0.3278	0.3972	0.0652	0.6914	0.4494	0.3154	0.8750	0.8571	0.4167	1
Data 59	0.0820	0.2825	0.4355	0.2575	0.0623	0.0000	0.1235	0.4831	0.1462	0.8750	0.6429	0.7500	1
Data 60	0.2295	0.5605	0.2298	0.0000	0.2996	0.0217	0.5556	0.3596	0.0000	0.8750	0.4286	0.4167	1
Data 61	0.0820	0.5471	0.6371	0.6455	0.2024	0.0000	0.3827	0.7191	0.4077	0.8125	0.3571	0.4167	1
Data 62	0.0984	0.7354	0.3952	0.4716	0.8686	0.0435	0.1852	0.5955	0.3615	0.8750	1.0000	0.5000	1
Data 63	0.2787	0.5426	0.4395	0.4649	0.3523	0.0652	0.7531	0.4719	0.3769	0.7500	1.0000	0.5000	1

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 64	0.0984	0.1614	0.2540	0.3579	0.0496	0.0435	0.1111	0.2584	0.3923	0.9375	0.8571	0.4167	1
Data 65	1.0000	0.6009	0.3831	0.2174	0.0334	1.0000	0.8025	0.2809	0.1923	1.0000	1.0000	0.3333	1
Data 66	0.0492	0.3901	0.2460	0.2007	0.0284	0.0000	0.2222	0.4494	0.2077	0.6875	0.7857	0.5000	1
Data 67	0.1967	0.8610	0.6573	0.3779	0.4991	0.0435	1.0000	0.4719	0.2462	0.9375	0.9286	0.4167	1
Data 68	0.1639	0.3274	0.3831	0.0167	0.2569	0.1087	0.1975	0.5056	0.0154	0.9375	1.0000	0.5000	1
Data 69	0.3934	0.1883	0.2863	0.1706	0.0000	0.0217	0.3333	0.1236	0.2385	0.6875	1.0000	0.5833	1
Data 70	0.2131	0.3722	0.5524	0.4716	0.4992	0.1304	0.6790	0.4382	0.3154	0.8125	1.0000	0.7500	1
Data 71	0.2131	0.3857	0.5161	0.2575	0.1421	0.1522	0.2346	0.6292	0.2923	1.0000	1.0000	0.2500	1
Data 72	0.1475	0.1794	0.2903	0.4114	0.1191	0.0652	0.1111	0.3483	0.4308	1.0000	1.0000	0.4167	1
Data 73	0.1311	0.2556	0.1774	0.4114	0.0071	0.0000	0.4321	0.2360	0.3846	0.8750	1.0000	0.2500	1
Data 74	0.2295	0.2556	0.4234	0.2642	0.1281	0.1304	0.4568	0.3258	0.1692	0.9375	1.0000	0.4167	1
Data 75	0.0000	0.2556	0.1653	0.3612	0.0290	0.0000	0.3827	0.1011	0.3308	0.1875	0.3571	0.4167	0
Data 76	0.1967	0.4933	0.4153	0.2441	0.4144	0.0435	0.8519	0.4270	0.1692	0.8750	1.0000	0.4167	0
Data 77	0.0820	0.5381	0.0968	0.2809	0.0861	0.0435	0.9383	0.1124	0.3000	0.8750	1.0000	0.4167	0
Data 78	0.0000	0.1928	0.0847	0.3077	0.0000	0.0000	0.3210	0.0674	0.2462	0.3750	0.4286	0.3333	0
Data 79	0.0492	0.3184	0.0968	0.2274	0.0774	0.0217	0.4321	0.1236	0.1538	0.4375	0.7143	0.6667	0

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 80	0.0328	0.2242	0.1371	0.1706	0.0000	0.0000	0.3951	0.1685	0.1462	0.3750	0.7857	0.5833	0
Data 81	0.0000	0.1839	0.1573	0.1271	0.0000	0.0000	0.2469	0.1573	0.1308	0.5000	0.5714	0.3333	0
Data 82	0.0164	0.0897	0.0444	0.1037	0.0070	0.0000	0.0617	0.0449	0.0769	0.5625	0.7143	0.5000	0
Data 83	0.0164	0.2108	0.3226	0.1739	0.0000	0.0000	0.2716	0.2360	0.1231	0.4375	0.7857	0.7500	0
Data 84	0.0328	0.1794	0.1371	0.0803	0.0367	0.0000	0.3704	0.1573	0.0538	0.5625	1.0000	0.8333	0
Data 85	0.0984	0.6054	0.3669	0.8060	0.1265	0.0000	0.8148	0.4045	0.7077	0.3750	0.8571	0.5833	0
Data 86	0.0000	0.2780	0.2540	0.1873	0.0000	0.0000	0.4198	0.2135	0.2308	0.6875	0.7143	0.2500	0
Data 87	0.0984	0.1883	0.0726	0.2341	0.0246	0.0217	0.2840	0.0674	0.2308	0.3125	0.2857	0.0833	0
Data 88	0.0000	0.3274	0.2097	0.2408	0.0285	0.0000	0.5185	0.1461	0.2000	0.4375	0.7857	0.8333	0
Data 89	0.0000	0.1300	0.0847	0.1070	0.0032	0.0000	0.1481	0.0787	0.0923	0.5625	0.7143	0.5000	0
Data 90	0.1311	0.5650	0.4032	0.6789	0.2229	0.0000	0.7407	0.4944	0.5769	0.5000	1.0000	0.6667	0
Data 91	0.1967	0.4664	0.4113	0.2809	0.3987	0.0435	0.7531	0.4607	0.2000	0.8125	1.0000	0.5000	0
Data 92	0.0328	0.0583	0.0242	0.2508	0.0085	0.0000	0.0988	0.0112	0.2692	0.1875	0.2857	0.2500	0
Data 93	0.0656	0.5381	0.0927	0.2809	0.0811	0.0435	0.9259	0.1124	0.3000	0.8125	1.0000	0.5000	0
Data 94	0.1639	0.2646	0.1048	0.2140	0.0663	0.0217	0.4074	0.1685	0.1538	0.4375	0.7143	0.5000	0
Data 95	0.1148	0.2018	0.1129	0.1237	0.0469	0.0000	0.3704	0.1685	0.0846	0.5000	0.7857	0.6667	0

Data	Feature1	Feature2	Feature3	Feature4	Feature5	Feature6	Feature7	Feature8	Feature9	Feature10	Feature11	Feature12	Output
Data 96	0.1311	0.6726	0.5202	0.7525	0.4456	0.0652	0.6296	0.7303	0.6231	0.5625	0.8571	0.7500	0
Data 97	0.1311	0.6771	0.4839	0.7893	0.3514	0.0435	0.7160	0.6517	0.6692	0.5625	0.9286	0.7500	0
Data 98	0.0000	0.2063	0.1452	0.1037	0.0000	0.0000	0.2963	0.1348	0.1077	0.5625	0.7143	0.4167	0
Data 99	0.0000	0.1256	0.0000	0.1773	0.0000	0.0000	0.0000	0.0000	0.1154	0.1250	0.2143	0.2500	0

The normalization process resulted in substantial scaling for features with inherently large numerical values. For instance, Feature 2, which originally ranged from 40 to 263, and Feature 5, with values reaching as high as 33,360, were transformed such that the smallest value in each feature mapped to 0 and the largest to 1, with all intermediate values proportionally adjusted. The value for feature 5 is significantly larger than other values because of the nature of the feature itself. Feature 5 represent the note length being added by the user into the system. Every character being entered will be count as an input from the user into the system.

This ensured that no single feature disproportionately influenced the analysis due to its magnitude. In contrast, features that already had smaller numerical ranges, such as Feature 1, Feature 3, and Feature 4, experienced relatively minor transformations. Their normalized values still spanned the 0 to 1 range, but the step size between adjacent values was smaller compared to high-range features.

The application of Min-Max Normalization effectively preserves the overall data structure while ensuring that all features contribute equitably to the modeling process. This is particularly beneficial in machine learning applications where algorithms such as Multi-Layer Perceptron, Support Vector Machines, and K-Nearest Neighbours rely on distance-based calculations. Without normalization, high-magnitude features like Feature 5 could dominate model learning, leading to biased weight distributions. Additionally, the transformation prevents models from assigning undue importance to any specific feature simply due to its numerical scale rather than its actual significance in predictive performance.

One key observation from the normalized dataset is that while all values now exist within a common scale, the relative importance of each feature remains unchanged. The variation within each feature is preserved, meaning that patterns in the original data are maintained even after scaling.

In conclusion, Min-Max Normalization successfully transforms the dataset into a comparable scale, addressing the issue of differing numerical magnitudes while maintaining data integrity. This ensures that all features contribute meaningfully to the analysis without allowing high-valued features to dominate. The normalization process is therefore a critical preprocessing step, enhancing the robustness of predictive modelling and improving model performance by ensuring fair and balanced feature representation.

5.6 Result for Feature Selection Method

Feature selection is a critical step in machine learning that enhances model performance by identifying the most relevant predictors while reducing dimensionality. By eliminating redundant or less informative features, feature selection improves computational efficiency, mitigates overfitting, and enhances the interpretability of the model. Various methods exist to evaluate feature importance, each leveraging different statistical, ranking, or model-based techniques to assess the contribution of individual features. In this section, several feature selection techniques, such as Mutual Information, Recursive Feature Elimination Ranking, Random Forest Importance, XGBoost Importance and L1 Regularization were applied to determine the most significant features. These methods provide a comprehensive assessment of feature relevance, ensuring that only the most impactful predictors are retained for the final model. The subsequent subsections detail each feature selection approach and its role in optimizing the dataset for predictive analysis.

5.6.1 Feature Selection Analysis

Feature selection plays a crucial role in building effective and interpretable predictive models, especially in educational datasets were reducing dimensionality without compromising accuracy is essential. This section presents the analysis results from five different feature selection techniques such as Mutual Information, Recursive Feature Elimination (RFE), Random Forest Importance, XGBoost Importance, and L1 Regularization (Lasso) were applied to the student academic performance dataset to identify the most relevant input features. Each method uses a distinct approach to determine feature importance, ranging from statistical dependence and model-based ranking to ensemble learning and regularization. The goal is to identify the critical features that contribute most to predicting academic outcomes and to guide the development of a simplified, robust, and efficient classification model.

5.6.1.1 Mutual Information

The Mutual Information feature selection method provides a robust quantitative measure of the dependency between individual input features and the target class. Table

5.5 illustrates the mutual information scores for all 12 features, allowing for the identification of variables with the highest contribution to predictive modelling.

Table 5.5
Mutual Information Feature Selection Result

Feature	Mutual Information Value
Feature10	0.3787
Feature3	0.0963
Feature2	0.0864
Feature5	0.0858
Feature4	0.0826
Feature9	0.0765
Feature6	0.0723
Feature8	0.0612
Feature1	0.0386
Feature11	0.0362
Feature7	0.0145
Feature12	0.0008

Feature10, which corresponds to “Tutorial Correct 3 and Above,” obtained the highest mutual information value at 0.3788. This substantial score indicates a strong non-linear dependency between this feature and the target label, suggesting that the number of tutorial attempts answered correctly (specifically, three or more) has a significant influence on a student’s academic outcome. Its score is notably higher than the second-highest ranked feature, Feature3 (“Access to Lecture Slides After Lecture”), which registered a score of 0.0963. This difference highlights the dominant role of tutorial performance, particularly correctness, in distinguishing students' academic performance levels.

The next tier of features includes Feature3, Feature2 (“Access to Lecture Slides During Lecture”), Feature5 (“Note Length”), and Feature4 (“Access to Lecture Slides Outside Lecture”), each with mutual information values ranging from approximately 0.08 to 0.09. These features reflect student engagement behaviours related to how and

when instructional materials are accessed, as well as the extent of note-taking. Their relatively higher mutual information values suggest that patterns of content review and depth of interaction with learning materials also play a relevant role in performance prediction, albeit to a lesser extent than tutorial correctness. Mid-ranking features such as Feature9 (“Exercise Outside Lecture”), Feature6 (“Exercise Before Lecture”), and Feature8 (“Exercise After Lecture”) show mutual information scores between 0.06 and 0.07. While these scores are lower, they still contribute marginally to the prediction, suggesting that self-initiated or time-flexible learning tasks have a moderate impact on academic performance.

Mutual Information results provide a foundational insight into the relevance of individual features, particularly highlighting the influence of correct tutorial responses and the strategic review of materials after lectures.

5.6.1.2 RFE Ranking

The Recursive Feature Elimination method applied in Table 5.6 prioritizes Feature10, Feature3, Feature7, and Feature12 as the most significant predictors in the dataset, with Feature10 securing the highest ranking.

Table 5.6
RFE Ranking Feature Selection Result

Feature	RFE Ranking
Feature10	1
Feature3	2
Feature7	3
Feature12	4
Feature1	5
Feature8	6
Feature11	7
Feature6	8
Feature5	9
Feature2	10

Feature	RFE Ranking
Feature9	11
Feature4	12

The ranking structure suggests that these features contribute the most to the predictive model's classification performance. Notably, Feature10, which has been consistently ranked as the most influential across multiple feature selection methods reinforces its relevance in predicting student academic performance. Similarly, Feature3 and Feature7, appearing in the top three rankings, highlight their strong impact in distinguishing performance outcomes. On the other hand, Feature4 and Feature9, positioned at the bottom of the ranking, indicate their relatively lower predictive power, suggesting that they may contain redundant or less informative data. The structured ranking of features demonstrates how RFE systematically evaluates and eliminates less significant features, ensuring that only the most relevant attributes are retained for model training.

From an analytical perspective, the results suggest that RFE effectively identifies key predictors while minimizing the inclusion of features that contribute little to the overall predictive performance. The presence of Feature10, Feature3, and Feature7 at the top indicates that these attributes have strong correlations with the target variable, enhancing the model’s classification ability. The intermediate-ranked features, such as Feature8, Feature11, and Feature6, may still hold some predictive value, but their ranking suggests they are less impactful compared to the top features. Meanwhile, features with lower rankings, such as Feature2, Feature9, and Feature4, may introduce noise or redundancy if retained in the final model. This ranking structure highlights the benefit of employing RFE as a feature selection method, as it systematically eliminates weaker predictors to enhance model interpretability and efficiency.

These findings emphasize the importance of Feature10 and other top-ranked features in improving the model’s generalization capability and reducing the risk of overfitting. By narrowing the dataset to the most relevant predictors, RFE ensures that the final model remains computationally efficient without compromising accuracy. The lower-ranked features, while not necessarily irrelevant, may not provide substantial predictive advantages, and their removal could lead to a more optimized model. As a result, integrating RFE-based feature selection with an optimized classifier is expected

to enhance classification performance, ensuring that the model makes robust and generalizable predictions for student academic performance.

5.6.1.3 Random Forest Importance

The feature importance ranking derived from the Random Forest algorithm, as presented in Table 5.7 provides critical insights into the contribution of each feature towards predicting student academic performance.

Table 5.7
Random Forest Importance Feature Selection Result

Feature	Random Forest Importance
Feature10	0.3086
Feature3	0.0940
Feature8	0.0936
Feature2	0.0708
Feature11	0.0662
Feature7	0.0653
Feature9	0.0648
Feature5	0.0548
Feature4	0.0522
Feature12	0.0497
Feature1	0.0432
Feature6	0.0363

Feature10 emerges as the most influential predictor, with an importance score of 0.3087, significantly higher than all other features. This dominance indicates that Feature10 plays a crucial role in differentiating student outcomes and should be prioritized in predictive modelling. Following Feature10, Feature3, Feature8, and Feature2 exhibit moderate importance scores, ranging from 0.0940 to 0.0708, suggesting their relevance in capturing meaningful patterns in student performance. Feature11, Feature7, and Feature9 display slightly lower but still noteworthy

importance scores, indicating that they contribute to model accuracy but may be secondary in impact. On the other hand, Feature6 and Feature1, ranking lowest with scores of 0.0363 and 0.0432, respectively, appear to have minimal influence, which suggests they may not significantly impact the classification task.

The results of this Random Forest Importance analysis align with expectations regarding the feature selection process. The model assigns higher importance to features that exhibit higher predictive power and minimal redundancy, ensuring that only the most relevant predictors influence the classification outcome. The substantial weight assigned to Feature10 suggests that it may encapsulate a comprehensive indicator of student performance, potentially aggregating multiple underlying learning behaviors or engagement metrics. Conversely, features with low importance scores, such as Feature6, Feature1, and Feature12, may represent redundant or weakly correlated predictors. The consistent ranking of Feature3, Feature8, and Feature2 across multiple feature selection methods reinforces their stability as informative attributes for academic performance prediction. However, while Random Forest Importance provides an empirical basis for feature ranking, it is essential to validate these rankings by comparing them against other selection techniques, ensuring that model performance is not compromised by excluding lower-ranked features.

These findings highlight the effectiveness of Random Forest Feature Importance in identifying key predictors while discarding less informative variables. The ranking results suggest that an optimized predictive model could achieve higher accuracy and generalization by focusing on the top-ranked features while eliminating or de-prioritizing lower-ranked ones. However, it is important to conduct further validation through cross-validation and feature ablation studies to ensure that removing lower-ranked features does not degrade model performance.

5.6.1.4 XGBoost Importance

The XGBoost feature selection results, as presented in Table 5.8, highlight the relative importance of each feature in predicting student academic performance.

Table 5.8
XGBoost Importance Feature Selection Result

Feature	XGBoost Importance
Feature10	0.4064
Feature7	0.1040
Feature12	0.0740
Feature6	0.0708
Feature2	0.0639
Feature1	0.0565
Feature5	0.0517
Feature4	0.0505
Feature3	0.0492
Feature8	0.0337
Feature11	0.0282
Feature9	0.0109

Feature10 emerges as the dominant predictor with an importance score of 0.4064, significantly higher than all other features. This suggests that Feature10 has a substantial influence on classification accuracy, making it a critical component in model development. The next most important feature, Feature7, with a score of 0.1039, is markedly lower than Feature10 but still holds a considerable predictive weight. Other features, such as Feature12, Feature6, and Feature2, maintain moderate importance scores, indicating their relevance but suggesting that they may not contribute as strongly as the top-ranked features. Features with the lowest importance values, such as Feature9 (0.0109) and Feature11 (0.0282), indicate minimal contribution to the predictive model and may be candidates for removal in feature selection optimization.

From these results, XGBoost effectively identifies key features based on their impact on classification decisions. The high ranking of Feature10, similar to findings in the Random Forest Importance analysis, suggests that this feature consistently provides strong predictive power across multiple selection methods. The notable difference in ranking for Feature7, which appears significantly higher in XGBoost than in Random Forest, implies that its impact may be more pronounced in models that incorporate

gradient boosting techniques. The relative importance of Feature12 and Feature6 also indicates that XGBoost detects patterns in these variables that may not have been as evident in other selection methods. Additionally, the lower importance scores for Feature9 and Feature11 suggest that these features may contain redundant or weakly correlated information, supporting the argument for feature reduction to enhance model efficiency.

These results emphasize the capability of XGBoost in prioritizing features that maximize classification performance while minimizing redundancy. The findings suggest that optimizing the MLP-based student performance classifier by focusing on the highest-ranked features, particularly Feature10, Feature7, and Feature12, could lead to improved model accuracy and generalization.

5.6.1.5 L1 Regularization Lasso

The application of L1 Regularization to the student academic performance dataset yielded a clear distinction between features with significant predictive weight and those with negligible contribution s shown in Table 5.9 Feature10 emerged as the most influential variable with an importance score of 2.943, reaffirming its dominance across nearly all prior selection methods. This feature, associated with tutorial performance (students correctly answering three or more questions), demonstrates strong predictive power, likely because it directly reflects the student's cognitive engagement and understanding of course material. Following Feature10, Feature7, Feature3, and Feature1 were also retained with moderately high weights, suggesting they hold essential behavioural cues linked to academic outcomes. Feature5, Feature4, and Feature2 exhibited lower but non-zero coefficients, implying marginal but potentially supportive roles in improving model granularity.

Table 5.9
L1 Regularization Lasso Feature Selection Result

Feature	L1 Regularization Importance
Feature10	2.9428
Feature7	1.6431
Feature3	1.0663

Feature	L1 Regularization Importance
Feature1	0.9241
Feature5	0.6432
Feature4	0.4405
Feature2	0.4181
Feature8	0.2661
Feature6	0
Feature9	0
Feature11	0
Feature12	0

From Table 5.9, an important inference from the Lasso results is its inherent capacity to enforce sparsity by assigning a coefficient of zero to less informative features. In this case, Features 6, 9, 11, and 12 were eliminated from the model, signifying their limited or redundant contribution to predictive accuracy. Notably, Feature11 and Feature12, related to tutorial completion behaviour were excluded, possibly due to multicollinearity with stronger variables such as Feature10. This indicates that Lasso not only identifies high-impact predictors but also effectively suppresses noise, improving model interpretability and efficiency. The absence of weights for these features reinforces the notion that not all engagement metrics contribute equally and that feature selection must be driven by robust regularization strategies to reduce model complexity.

These findings support the broader insight that L1 Regularization serves as a powerful embedded method for reducing dimensionality without compromising classification performance. By selectively retaining only the most salient features, the resulting model becomes leaner and more generalizable qualities that are essential in educational settings where interpretability and deployment feasibility are crucial. The strong alignment of Lasso's top-ranked features with those identified by other methods like RFE and XGBoost further validates the reliability of the selected attributes. Thus, integrating Lasso into the feature engineering pipeline enhances the foundation for developing predictive models that are not only accurate but also scalable and transparent for decision-making in student performance analytics.

5.6.1.6 Comparative Feature Importance and Convergence Analysis

A comparative examination of the five feature selection techniques which are Mutual Information, RFE, Random Forest Importance, XGBoost Importance, and L1 Regularization shows a clear pattern of convergence. Across these distinct analytical approaches, four features consistently emerged among the most highly ranked predictors with no particular sequence are :

Feature10 (Tutorial Correct 3 and above),
Feature3 (Access to Lecture Slide and Additional Notes After Lecture),
Feature7 (Exercise During Lecture),
Feature2 (Access to Lecture Slide and Additional Notes During Lecture).

Despite differences in evaluation principles, statistical dependency, recursive elimination, ensemble-based importance, and regularized coefficient shrinkage, these four features repeatedly appeared within the dominant subset identified by each method.

The recurrence of these features across independent selection strategies indicates structural stability in their predictive contribution. Feature10 demonstrates the strongest and most consistent dominance, achieving the highest ranking or importance score across all five methods. Feature3, Feature7, and Feature2 also maintain repeated presence within the upper-tier rankings, even where relative positions vary slightly depending on the algorithm. This cross-method agreement reduces the likelihood that their selection is artefactual or method-dependent, instead suggesting that these variables contain substantial discriminatory information relative to the target class.

Collectively, the convergence of these four features provides methodological triangulation and strengthens confidence in their robustness. Their consistent identification across filter-based, wrapper-based, ensemble-based, and embedded techniques justifies their prioritization in subsequent modelling stages. Rather than relying on a single ranking mechanism, this multi-method consensus establishes Feature10, Feature3, Feature7, and Feature2 as the core predictive subset within the dataset, forming a stable foundation for optimized model development and generalizable performance prediction.

5.6.2 Evaluation and Analysis of Feature Selection Method

The evaluation and interpretation of feature selection methods play a important role in understanding how different techniques influence the identification of critical predictors for student academic performance. Subsection 5.6.2.1 synthesizes the findings from five distinct feature selection to uncover patterns of consistency, algorithm-specific preferences, and the trade-offs between dimensionality reduction and model performance. Subsection 5.5.6.1 focuses on identifying features that consistently emerge as most relevant across multiple methods, strengthening their credibility as core predictors. Subsection 5.6.2.2 explores the unique behaviour of each algorithm in ranking features, revealing how their inherent selection criteria impact the outcomes. Finally, Subsection 5.6.2.3 compares the extent to which each method reduces feature dimensionality while preserving or improving classification metrics, offering insights into optimal model simplification without sacrificing accuracy. Together, these discussions provide a comprehensive understanding of the strategic role feature selection plays in building efficient and interpretable machine learning models.

5.6.2.1 Consistent Feature Importance Across Multiple Methods

The comprehensive analysis of the feature selection method identified that there are five activities that significantly contribute to students' final grades to be GPA $\geq$ 3.00 or GPA < 3.00 in a C programming course at UiTM. Specifically, Feature 10 (Tutorial Correct 3 and above), Feature 3 (Access to Lecture Slide and Additional Notes After Lecture), Feature 7 (Exercise During Lecture), Feature 2 (Access to Lecture Slide and Additional Notes During Lecture), and Feature 1 (Access to Lecture Slide and Additional Notes Before Lecture) were the most influential activities.

Results from Mutual Information, RFE, Random Forest, XGBoost, and Lasso consistently emphasized the importance of these five features. Feature 10 remained the highest-ranked across all methods, indicating the critical role of students' ability to correctly answer multiple tutorial questions. Feature 3 and Feature 2, which relate to engagement with lecture content during and after lectures, were also frequently ranked in the top group, highlighting the relevance of continuous material access. Feature 7, reflecting active participation through in-class exercises, regularly appeared among the

top predictors. Although Feature 1 ranked slightly lower in some methods, it consistently demonstrated value in contributing to academic performance predictions.

These recurring results across distinct feature selection techniques demonstrate strong agreement in identifying the most meaningful predictors for GPA classification. The selected features align with core educational behaviours, including preparation, attentiveness during class, review of materials, and success in formative assessments. Their presence across all methods indicates they are not only statistically significant but also practically relevant in modelling student success.

Overall, the repeated emergence of Feature 10 (Tutorial Correct 3 and above), Feature 3 (Access to Lecture Slide and Additional Notes After Lecture), Feature 7 (Exercise During Lecture), and Feature 2 (Access to Lecture Slide and Additional Notes During Lecture) across five analytical methods reinforces the crucial role of tutorial engagement and timely content access in predicting academic performance. This consistency strengthens the model's interpretability and ensures its application is both theoretically grounded and educationally meaningful.

5.6.2.2 Algorithm-Specific Feature Behaviour

The five feature selection methods employed in this study demonstrate unique evaluation behaviours, each shaped by its underlying mathematical and algorithmic principles. Mutual Information, a filter-based method, primarily evaluates the non-linear dependency between each feature and the target class independently. As a result, it tends to favour features that show a strong individual relationship with the output, such as Feature 10 and Feature 3. However, it often underrepresents interactive or redundant relationships, which may explain why certain features with contextual significance, like Feature 7 or Feature 2, rank lower in this method.

In contrast, Recursive Feature Elimination, a wrapper-based method, evaluates features based on their combined contribution to model performance. RFE favoured Feature 7 and Feature 12 alongside Feature 10 and Feature 3, highlighting its strength in capturing synergies among features. This method's behaviour reflects its iterative backward elimination process, which retains features only if their removal leads to a measurable decrease in model accuracy. It is particularly sensitive to feature interactions and can reveal underlying dependencies that filter methods may overlook.

The two tree-based ensemble methods, Random Forest and XGBoost Importance, emphasize features that contribute most to reducing impurity or loss during tree construction. These methods consistently rank Feature 10 at the top, reinforcing its dominant predictive influence. However, XGBoost diverges from Random Forest by assigning a much higher weight to Feature 7 and Feature 12, likely due to its ability to model complex non-linear and interaction effects more aggressively. XGBoost's gradient boosting mechanism tends to magnify the importance of features that appear frequently across split points, even if their standalone predictive value is modest.

L1 Regularization, an embedded method, penalizes less informative or redundant features by driving their coefficients to zero. This method demonstrated a more aggressive elimination pattern by excluding Feature 6, Feature 9, Feature 11, and Feature 12 completely. Notably, Lasso retained Feature 2 and Feature 7 with moderate to high weights, suggesting that despite their lower rankings in filter-based approaches, these features contribute meaningfully when multicollinearity is addressed. Lasso's sparsity-inducing behaviour offers a streamlined subset of predictors, which is advantageous for interpretability and deployment.

Each method reveals a different facet of the dataset's structure. While Feature 10 remains universally dominant, the importance assigned to other features like Feature 7, Feature 2, and Feature 12 varies across algorithms. This variability highlights the value of multi-method validation, ensuring that the final selection of features reflects both individual significance and contextual synergy within the feature space.

5.6.2.3 Comparison of Reduction in Dimensionality Based on Feature Selection Methods

The effectiveness of dimensionality reduction in enhancing model performance and simplifying feature input was assessed across five different feature selection methods. As shown in Table 5.10 each method selected six features from the original 12, offering a comparable basis to evaluate the trade-off between model complexity and predictive strength.

Table 5.10
Sequence of Selected Features for Each Method

Feature Selection Method	Selected Features	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
NONE	All features 12.	80	75	60	67
Mutual Information	10,3,2,5,4,9	80	81	08	80
RFE Ranking	10,3,7,12,1,8	87	87	87	87
Random Forest Importance	10,3,8,2,11,7	80	80	80	79
XGBoost Importance	10,7,12,6,2,1	0.8	0.8	0.8	0.79
L1 Lasso	10,7,3,1,5,4	0.87	0.87	0.87	0.87

Using all 12 features without any selection resulted in an accuracy of 80% but yielded relatively low precision, recall, and F1-score. The results suggest that while the model could correctly classify a majority of cases, it struggled to generalize, particularly in detecting students with lower GPAs, likely due to noise and redundancy introduced by less relevant features.

The decision to reduce the feature set from 12 to 6 was not arbitrary but grounded in both methodological and theoretical considerations. With only 99 samples available, maintaining all 12 features increases the dimensionality-to-sample ratio, which can amplify variance and heighten the risk of overfitting. In neural network models such as MLP, the number of input features directly influences the number of trainable parameters; more features lead to more weights, greater model complexity, and higher susceptibility to noise and redundant correlations. By reducing the feature space to six highly informative variables (50% reduction), the experiment systematically evaluates whether a lower-dimensional representation can preserve or enhance predictive performance while improving generalization stability. This dimensionality reduction aligns with the bias–variance trade-off principle: a moderately simplified model may sacrifice some representational capacity but gains robustness, interpretability, and computational efficiency. The empirical improvement observed with RFE and L1 Lasso (87% across all metrics) supports this rationale, demonstrating that removing redundant or weakly contributing features reduces model complexity without compromising, and in this case improving, classification reliability.

Upon applying feature selection techniques, several methods stood out in terms of performance preservation and enhancement. Mutual Information and Random Forest Importance achieved the same accuracy of 80% as the full-feature model. However, Mutual Information recorded higher precision of 81% and a balanced recall of 80%, resulting in an F1-score of 80%. This demonstrates its capacity to isolate features that provide informative signals without degrading model generalization.

In contrast, Recursive Feature Elimination (RFE) and L1 Regularization (Lasso) both yielded superior results, with all four metrics, accuracy, precision, recall, and F1-score, reaching 87%. These methods effectively identified the most critical attributes for classification, allowing the MLP model to generalize well despite the reduced input size. Their ability to deliver high classification accuracy with fewer features underscores their strength in optimizing performance while simplifying the model structure.

XGBoost Importance maintained a respectable accuracy of 80% and an F1-score of 79%. While this performance is consistent with that of Random Forest, it falls short of the enhancement observed with RFE and Lasso. These outcomes indicate that while XGBoost Importance can be valuable for maintaining performance, it may not offer the same level of simplification effectiveness in this context.

Overall, the comparison in Table 5.10 highlights that RFE and L1 Lasso provide the best balance between dimensionality reduction and model performance. By trimming the dataset to six features without compromising accuracy or generalization, these methods improve computational efficiency and facilitate easier model interpretation, attributes essential for educational applications. The analysis reinforces the importance of strategic feature selection to achieve high performing yet streamlined predictive models.

5.6.3 Result For RFE Feature Selection With 5 Optimizer

This section evaluates the impact of different optimization algorithms on training a Multi-Layer Perceptron model using Recursive Feature Elimination feature selection. The choice of optimizer significantly affects convergence speed, stability, and predictive accuracy, influencing the model's ability to generalize effectively. To assess these effects, we compare the performance of AdamW, AdaGrad , AmsGrad , Nadam

and SDG with Momentum focusing on their role in loss reduction and classification performance.

The analysis is conducted through confusion matrices for each optimizer, providing insights into how well the model distinguishes between target classes. By examining their impact on training dynamics, we aim to determine which optimizers enhance model performance and stability while ensuring efficient learning. This comparison will highlight the most effective optimization strategies for student academic performance prediction. Table 5.11 shows the Confusion Matrices of All Optimizers Across Training, Validation, and Testing Phases

Table 5.11
Confusion Matrices of All Optimizers Across Training, Validation, and Testing Phases

Optimizer	Phase	True Positive	False Positive	False Negative	True Negative
AdaGrad	Training	24	1	2	10
	Validation	8	36	2	1
	Testing	2	1	8	4
AdamW	Training	32	2	2	8
	Validation	0	35	2	3
	Testing	2	3	10	2
SGD with Momentum	Training	14	3	4	7
	Validation	14	28	7	4
	Testing	2	4	4	7
Nadam	Training	27	1	2	9
	Validation	5	36	2	2
	Testing	2	2	9	3
Amsgrad	Training	27	1	2	8
	Validation	5	36	2	3
	Testing	2	3	9	2

5.6.3.1 Bias-Variance Tradeoff and Model Generalization

Table 5.12 and Table 5.13 presents the training and testing performance metrics, respectively, for five optimization algorithms: AdamW, AdaGrad, AmsGrad, Nadam, and SGD with Momentum. The accuracy drops observed across different optimizers indicate varying degrees of overfitting and underfitting tendencies. Specifically, AdamW, AmsGrad, and SGD with Momentum exhibited significant decreases in accuracy between training and testing, which are 17.1%, 18.0%, and 16.2%, respectively, signalling potential overfitting. In contrast, AdaGrad and Nadam showed smaller reductions of only 6.9% and 11.3%, respectively, suggesting better generalization capabilities. Recall values also displayed considerable fluctuation: while AdamW maintained a perfect recall of 100% in both training and testing, SGD with Momentum suffered a notable decline from 50.0% during training to 44.44% in testing, highlighting high bias and an inability to capture the data distribution effectively.

Table 5.12
Training Performance Metrics of Various Optimization Algorithms

Optimizer	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
AdamW	97.1	94.1	100	96.7
AdaGrad	86.9	96.0	75.0	84.2
AmsGrad	91.3	96.4	84.4	89.9
Nadam	91.3	96.4	84.4	89.9
SDG with Momentum	71.2	82.4	50.0	62.2

Table 5.13
Testing Performance Metrics of Various Optimization Algorithms

Optimizer	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
AdamW	80.0	76.9	100	86.95
AdaGrad	80.0	88.9	80.0	84.2
AmsGrad	73.3	75.0	90.0	81.8
Nadam	80.0	81.8	90.0	85.7

Optimizer	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
SDG with Momentum	55.0	50.0	44.44	47.05

A deeper analysis reveals that AdamW and AmsGrad exhibit high-variance behaviour, characterized by strong training performance but notable deterioration during testing. AdamW achieved 97.1% accuracy, 94.1% precision, and 100% recall during training, but dropped to 80.0% accuracy, 76.9% precision, and maintained 100% recall during testing. Similarly, AmsGrad attained 91.3% accuracy and 96.4% precision in training, followed by a decrease to 73.3% accuracy and 75.0% precision in testing. These patterns are indicative of overfitting, where the models memorize the training data at the expense of broader generalization. Conversely, SGD with Momentum demonstrates underfitting symptoms, achieving low training accuracy (71.2%) and precision (82.4%), and deteriorating further to 55.0% accuracy and 50.0% precision during testing. Its recall also fell from 50.0% to 44.44%, signalling its limited ability to learn complex relationships within the data.

In contrast, AdaGrad and Nadam demonstrated more favourable bias-variance trade-offs. AdaGrad achieved a moderate training accuracy of 86.9%, with only a minor reduction to 80.0% in testing, while maintaining an F1-Score of 84.2% across both datasets, indicating strong stability. Nadam sustained a high F1-Score of 89.9% in training and 85.7% in testing, with precision slightly dropping from 96.4% to 81.8%, and recall improving from 84.4% to 90.0%. These results suggest that optimizers employing adaptive learning rate strategies, such as AdaGrad and Nadam, are better equipped to manage model flexibility, thereby mitigating both excessive memorization and insufficient pattern learning. To further enhance the generalization performance, AdamW and AmsGrad would benefit from additional regularization techniques such as dropout, L2 regularization, or early stopping. For underfitting cases observed in SGD with Momentum, strategies such as dynamic learning rate adjustment, batch normalization, or increasing model complexity are recommended to improve learning capacity.

The comparative evaluation highlights that the choice of optimizer significantly influences model generalization behaviour. AdamW and AmsGrad demonstrate tendencies toward high variance and overfitting, whereas SGD with Momentum

exhibits high bias and underfitting. AdaGrad and Nadam achieve a better balance between bias and variance, with Nadam emerging as the most consistent and reliable performer across both training and testing scenarios. These findings reinforce the critical importance of aligning optimizer selection with the complexity of the dataset and model architecture. Effective generalization is not solely dependent on the optimizer itself but also requires complementary strategies such as regularization or architectural refinement, tailored to address either overfitting or underfitting challenges present in the learning process.

5.6.3.2 Optimizer Performance Comparison

The testing results reveal notable differences in performance across the five evaluated optimizers: AdamW, AdaGrad, Nadam, AmsGrad, and SGD with Momentum. Accuracy, precision, recall, and F1-score were used as the main performance indicators to assess each optimizer's ability to classify student academic performance using the original 12 features. AdamW, AdaGrad, and Nadam achieved the highest testing accuracy of 80.0%, indicating a strong ability to generalize to unseen data. AmsGrad followed with an accuracy of 73.3%, while SGD with Momentum showed the weakest performance at only 55.0%.

Precision, which evaluates the proportion of correctly predicted positive observations, varied significantly among the optimizers. AdaGrad recorded the highest precision of 88.9%, indicating its effectiveness in minimizing false positives. AdamW and Nadam followed closely with 81.9%, while AmsGrad achieved 75.0%. In contrast, SGD with Momentum had the lowest precision at 50.0%, highlighting its struggle to distinguish between classes effectively. Regarding recall, how well the model identifies actual positive cases with AdamW led with a perfect 100.0%, followed by AmsGrad and Nadam at 90.0%, while AdaGrad reached 80.0%. Again, SGD with Momentum underperformed with a recall of only 44.4%, suggesting a high number of missed positive predictions.

F1-score, which balances precision and recall, presents a more holistic view of each optimizer's performance. AdamW achieved the highest F1-score of 86.96%, closely followed by Nadam at 85.7% and AdaGrad at 84.2%. AmsGrad scored slightly lower at 81.9%, while SGD with Momentum registered the lowest F1-score of 47.1%, reinforcing its inadequacy in managing the precision-recall trade-off.

The detailed confusion matrix also supports these interpretations. AdamW successfully classified 2 true positives and 2 true negatives during testing but had 3 false positives and 10 false negatives, emphasizing its high sensitivity (recall) at the expense of precision. AdaGrad correctly predicted 2 true positives and 4 true negatives, with 1 false positive and 8 false negatives, balancing between false alarms and missed detections. Nadam and AmsGrad demonstrated similar patterns, with both managing 90% recall and moderate false positive rates. However, SGD with Momentum had the weakest structure, capturing only 2 true positives and 7 true negatives, but incurring 4 false positives and 4 false negatives, indicating unstable learning dynamics and poor generalization.

From this comparative evaluation, AdamW is the most suitable when recall is prioritized over precision, especially in scenarios where false negatives are more critical, such as predicting at-risk students who require early intervention. AdaGrad stands out as the most balanced optimizer, demonstrating the highest precision with consistent recall, making it ideal for use cases where both accurate identification and reduced misclassification are equally important. Nadam closely follows AdaGrad, slightly leaning toward recall, which may be preferred in educational settings that value early risk detection. AmsGrad, while still viable, requires additional tuning to improve its precision. On the other hand, SGD with Momentum consistently underperforms across all metrics and may not be viable without significant adjustment or replacement with more adaptive alternatives.

These findings highlight the importance of optimizer selection in MLP model development. While adaptive optimizers like AdamW and AdaGrad show superior results, each optimizer's behaviour must be understood within the context of application-specific needs such as precision sensitivity or recall maximization.

5.6.3.3 Optimizer Performance on Convergence and Stability using RFE as Feature Selection

Table 5.14
Epochs Required for Convergence Across 20 Runs for Selected Optimizers

Run	SDG with Momentum	Adagrad	AdamW	Nadam	AmsGrad
Run 1	66	534	100	39	11

Run 2	23	621	100	91	103
Run 3	32	1000	54	171	84
Run 4	79	386	69	122	75
Run 5	65	205	102	128	23
Run 6	45	488	158	83	100
Run 7	49	815	96	21	92
Run 8	57	1000	11	144	117
Run 9	62	1000	122	45	57
Run 10	105	410	148	77	93
Run 11	57	463	67	107	116
Run 12	26	368	158	127	80
Run 13	52	739	43	60	96
Run 14	65	282	99	117	132
Run 15	46	19	122	26	145
Run 16	69	335	71	105	76
Run 17	84	13	76	85	54
Run 18	84	1000	11	65	82
Run 19	75	966	11	98	11
Run 20	74	132	117	155	96

The convergence behaviour of different optimizers was analysed by recording the number of epochs required for training completion across 20 independent runs a shown in Table 5.14. This evaluation was performed using RFE as the selected feature selection method. The epoch count serves as a proxy for both convergence speed and training stability, revealing how efficiently each optimizer adapts to the learning landscape when provided with a reduced but informative feature space. Results showed stark contrast between optimizers, with some consistently reaching convergence in fewer epochs, while others exhibited instability or prolonged training, occasionally hitting the predefined cap of 1,000 epochs.

Among the tested optimizers, SGD with Momentum demonstrated relatively consistent and fast convergence, requiring as few as 23 epochs (Run 2) and at most 105 epochs (Run 10). This tight range reflects stable behaviour and indicates that the momentum term is effectively smoothing gradients, accelerating convergence in flatter regions. AdamW also showed commendable performance, with epoch counts generally within a manageable range, and without any instances reaching the epoch ceiling. AdamW's inclusion of decoupled weight decay appears to regularize learning effectively, allowing the optimizer to converge both quickly and reliably, even when interacting with the reduced feature set provided by RFE.

In contrast, AdaGrad struggled with convergence stability, often requiring extremely high epoch counts. Notably, it hit the 1000-epoch limit in four separate runs (Runs 3, 8, 9, and 18) and exceeded 600 epochs in several others. Although AdaGrad's adaptive learning rate initially promotes strong updates, the rapidly shrinking learning rate over time results in premature stagnation, especially when the feature space is less redundant, as is the case post-RFE. This behaviour makes it computationally inefficient and suggests it may not be well-suited for deep learning tasks unless modified with techniques like learning rate restarts.

Nadam and AmsGrad offered intermediate performance. Nadam required anywhere between 21 and 171 epochs, while AmsGrad ranged from 11 to 145. These optimizers showed good convergence in many runs, but also demonstrated notable variance, possibly due to their sensitivity to noisy gradients or reliance on complex moment estimates. While their performance didn't match the consistency of SGD with Momentum or AdamW, they still achieved convergence in reasonable timeframes, particularly when their hyperparameters were well-tuned. The lower bounds of convergence (e.g., 21 epochs for Nadam in Run 7 and 11 for AmsGrad in Run 19) suggest that under optimal conditions, these optimizers can be efficient, though they appear more vulnerable to unstable learning paths in certain scenarios.

From a broader perspective, the epoch data strongly supports the superiority of optimizers that combine momentum-based updates with adaptive mechanisms, such as AdamW. The momentum term aids in escaping saddle points, while adaptive elements ensure gradient sensitivity across features. RFE as a feature selection strategy seems to favour optimizers capable of adjusting flexibly to a lean feature set, allowing them to learn faster without overfitting. In contrast, optimizers heavily dependent on shrinking learning rates or static update rules like AdaGrad are penalized, as they are unable to

adapt dynamically to varying curvature and sparsity introduced by the feature elimination process.

In conclusion, AdamW and SGD with Momentum stand out as the most efficient and stable optimizers under RFE-based training, balancing low epoch counts with high convergence reliability. Nadam and AmsGrad remain viable options, though with slightly reduced stability. AdaGrad, while theoretically appealing for sparse datasets, is disadvantaged in this context due to its vanishing learning rate behaviour. These findings suggest that the interplay between optimizer dynamics and the selected feature space significantly impacts training efficiency. Future work may benefit from exploring learning rate schedules (e.g., warm restarts or cyclical decay) to further improve convergence for optimizers with less stable performance profiles.

5.6.3.4 Convergence Dynamics and Overfitting Risk Across Optimizers

The convergence behaviour of each optimizer reveals not only how quickly training completes but also provides crucial insights into the model's learning dynamics and the risk of overfitting. The number of epochs required to reach convergence, as previously reported, varies significantly across optimizers, with SGD with Momentum and AdamW generally converging within a moderate number of epochs (23–158), while AdaGrad often reaches the maximum limit of 1,000 epochs. Despite its slower convergence, AdaGrad maintains consistent training accuracy, suggesting that its aggressive learning rate decay hinders rapid updates but still permits eventual fitting to the training data. In contrast, optimizers like Nadam and AmsGrad demonstrate high variability in convergence speed, with some runs completing within 20–30 epochs and others requiring over 100, hinting at unstable adaptation to the feature space or susceptibility to local minima.

Interestingly, fast convergence does not always imply optimal model generalization. For example, SGD with Momentum consistently converges within fewer than 100 epochs, yet its testing performance remains the lowest across all metrics, particularly in recall and F1-score. This pattern suggests that the model fits the training data too quickly and inadequately generalizes to unseen instances, a classic signature of overfitting. AdamW, although slightly slower in convergence, balances training and testing performance well, with high training accuracy of 91.3%, strong recall of 90.0%, and a robust F1-score of 85.7% during testing. This behaviour indicates that moderate

convergence combined with adaptive weight decay as in AdamW optimizer can mitigate overfitting risk and lead to more generalizable models.

Nadam and AmsGrad, both adaptive optimizers, display erratic convergence patterns across different runs. Their inconsistent epoch counts, ranging from as low as 21 to as high as 171, suggesting varying sensitivity to initialization and possibly unstable loss surface navigation. Nevertheless, both optimizers deliver satisfactory testing performance, particularly in recall and F1-score, indicating that despite the irregular convergence trajectories, they still manage to learn useful decision boundaries. However, this performance comes at the cost of predictability and computational efficiency, since convergence duration cannot be reliably anticipated. These findings point to the importance of considering both convergence consistency and result stability when selecting optimizers for real-world applications.

The convergence inefficiency of AdaGrad stands out as a practical concern. In several runs, the optimizer hits the maximum epoch threshold of 1,000, with minimal marginal gain in validation or testing accuracy. This inefficiency arises from its monotonically decreasing learning rate, which limits its capacity to make significant parameter updates in later training stages. Although it avoids overfitting by under-training, this conservative learning behaviour also compromises generalization power. Without learning rate adaptation strategies such as resets or cyclical schedules, AdaGrad's suitability for complex neural architectures remains limited, especially when paired with reduced feature spaces as in this study.

Overall, the convergence dynamics observed across the five optimizers underscore the delicate balance between training speed and model robustness. Optimizers like AdamW that offer controlled yet adaptive convergence prove to be the most effective in minimizing overfitting risks while achieving reliable generalization. SGD with Momentum, while computationally efficient, may converge prematurely and overfit without sufficient regularization. Meanwhile, optimizers like Nadam and AmsGrad require tuning to balance their powerful adaptive updates with consistent convergence behaviour. These insights reinforce the importance of convergence analysis not merely as a measure of training efficiency, but as a diagnostic tool for model reliability and predictive validity in real-world classification tasks.

5.6.3.5 Impact of Optimization Strategy on Misclassification Behaviour

The effect of different optimization strategies on misclassification patterns offers a valuable lens for evaluating model behaviour beyond aggregate metrics like accuracy or F1-score. By analysing confusion matrices for each optimizer across training, validation, and testing phases, clear differences emerge in how each handles false positives (FP) and false negatives (FN). For instance, AdamW achieves high recall of 90.0% during testing by minimizing false negatives (FN = 1), yet this comes with increased false positives (FP = 3), indicating that while it effectively identifies actual positives, it does so at the cost of occasionally misclassifying negatives. In contrast, AdaGrad exhibits strong precision of 88.9% by maintaining fewer false positives (FP = 1), but with more false negatives (FN = 8), it risks failing to identify many true positive instances. This contrast highlights a critical trade-off between conservative and aggressive classification behaviour dictated by the optimizer's learning dynamics.

SGD with Momentum demonstrates a particularly unbalanced misclassification pattern, with high false negatives (FN = 4) and moderate false positives (FP = 4), leading to the lowest recall (44.4%) and F1-score of 47.1% among all optimizers. These results suggest that its momentum-driven updates may be insufficient to capture complex patterns in the data when paired with a limited feature set, resulting in underfitting and unreliable classification boundaries. Its symmetric misclassification profile, mislabelling both classes equally. It points to a model that fails to internalize key distinctions between student performance classes. Consequently, SGD with Momentum may not be suitable for scenarios where recall and sensitivity to at-risk student identification are critical.

Nadam and AmsGrad, while delivering better overall performance than SGD with Momentum, display distinct misclassification signatures. Nadam maintains balanced FP and FN values (FP = 2, FN = 2) in the test set, resulting in a high F1-score of 85.7%, while AmsGrad, despite similar recall of 90.0%, incurs slightly more false positives (FP = 3), leading to a lower precision of 75.0% and F1-score of 81.9%. These differences, although subtle, suggest that Nadam is more consistent in managing the boundary between positive and negative classifications. This stability may be due to its adaptive moment estimation with incorporated Nesterov acceleration, which allows it to respond well to gradient variations without overshooting. AmsGrad, designed to

rectify some of Adam's over-adaptation tendencies, shows improvement over standard Adam but still struggles with controlling false positives under certain conditions.

Importantly, these misclassification behaviours are not isolated to the test set alone but are reflected in validation and training patterns. Optimizers that achieve perfect or near-perfect classification in training such as AdamW and AdaDelta that often see a rise in false positives or negatives during testing, signalling overfitting. Conversely, those that maintain a small, consistent level of misclassification across all phases like Nadam demonstrate better generalization. This consistency is a crucial factor in evaluating model reliability, particularly in educational domains where incorrect classification of at-risk students (false negatives) may delay or deny necessary intervention.

In can be conclude that optimization strategies significantly influence the nature and severity of misclassification. Optimizers like AdamW and Nadam offer favourable trade-offs, either prioritizing sensitivity (low FN) or balancing both error types to achieve robust generalization. AdaGrad, while precise, is conservative and risks under-identifying positive cases, whereas SGD with Momentum proves least effective, with excessive errors across both classes. Understanding these patterns provides valuable diagnostic insight into model behaviour, guiding the selection of optimization strategies not solely on performance metrics, but on their ability to minimize critical classification errors in high-stakes predictive tasks.

5.7 Optimization and Feature Selection Result Analysis

This section presents a comprehensive analysis of five optimization algorithms which are AdamW, AdaGrad, Nadam, AmsGrad, and SGD with Momentum being applied to an MLP classifier trained using Recursive Feature Elimination (RFE)-selected features. The findings reveal that AdamW and Nadam consistently outperform other optimizers, achieving high testing accuracy of 80.0% and F1-scores of 86.95% and 85.7%, respectively, while balancing recall and precision effectively. AdaGrad, although slower in convergence, delivers the highest precision of 88.9%, making it particularly effective in minimizing false positives. Conversely, SGD with Momentum shows the weakest performance, with the lowest recall of 44.4% and F1-score of 47.1%, highlighting issues with underfitting and unstable generalization.

From the convergence analysis, AdamW and SGD with Momentum converge efficiently across 20 runs without hitting the epoch cap, indicating training stability. However, AdaGrad repeatedly reaches the 1,000-epoch limit, confirming its inefficiency due to rapidly diminishing learning rates. Nadam and AmsGrad display moderate convergence speed but higher variability, suggesting sensitivity to initialization and loss complexity. The relationship between convergence speed and model fit is also emphasized, a fast convergence does not always imply good generalization, as seen in SGD with Momentum, which overfits quickly without learning useful decision boundaries.

The misclassification behaviour offers deeper insight into how optimization strategies affect error patterns. AdamW prioritizes high recall, minimizing false negatives but at the cost of increased false positives, while AdaGrad is more conservative, maintaining high precision but risking the omission of at-risk students. Nadam emerges as the most balanced optimizer, achieving consistent misclassification patterns across training, validation, and testing phases. Overall, the study confirms that optimizer choice not only affects performance metrics but also impacts convergence dynamics and error characteristics, which are critical when deploying predictive models in sensitive domains like student performance forecasting.

5.7.1 Result For Using ADAM optimizer on Selected Feature Selection Method

This section presents the evaluation of the Adam optimizer applied to the best six selected features obtained through various feature selection methods and compares their performance against the original 12-feature model. The effectiveness of feature selection is assessed based on its impact on model accuracy, stability, and generalization across training, validation, and testing phases. The feature selection methods explored include Mutual Information, Recursive Feature Elimination (RFE), Random Forest Importance, XGBoost Importance and L1 Lasso. Each selected feature subset was trained using the Adam optimizer, and results are analysed through confusion matrices to determine which selection method best enhances model performance. By comparing classification accuracy, precision, recall, and F1-score, this analysis aims to identify the most effective feature selection approach that optimizes the MLP model's predictive performance while reducing complexity.

5.7.2 Analysis of Feature Selection Effect with ADAM Optimizer

To evaluate the practical impact of different feature selection methods on classification performance, this section presents a comparative analysis using the Adam optimizer as the fixed learning strategy. Each feature selection method was applied to reduce the original 12-feature dataset to a subset of the most informative attributes, and the resulting model performance was assessed across training, validation, and testing datasets. Metrics such as accuracy, precision, recall, and F1-score were calculated to assess each model's learning behaviour, generalization capability, and robustness. The evaluation begins with a baseline model trained using all 12 original features, followed by detailed performance comparisons for models built using features selected via Mutual Information, Recursive Feature Elimination (RFE), Random Forest Importance, XGBoost Importance, and L1 Lasso. This section aims to determine which feature selection method yields the most efficient and generalizable classification performance when coupled with the Adam optimizer.

5.7.2.1 Original 12 Feature Result

Table 5.15
Result for the Original 12 Features

Feature Selection Method	Phase	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
ADAM + 12 original Features	Training	91.3	96.43	84.38	90.00
	Validation	66.67	40.00	50.00	44.44
	Testing	80.00	75.00	60.00	67.00

Table 5.15 presents the classification performance of the MLP model trained using the ADAM optimizer without any feature selection, based on the full set of 12 features. During the training phase, the model achieved a high accuracy of 91.3%, with excellent precision (96.43%), recall (84.38%), and an F1-score of 90.00%. These metrics indicate the model effectively learned from the training data, accurately classifying most students and achieving a strong balance between false positives and

false negatives. The particularly high precision suggests that the model reliably identifies at-risk students with minimal false alarms.

However, in the validation phase, there is a sharp decline in performance, with accuracy dropping to 66.67%, precision falling to 40.00%, recall to 50.00%, and the F1-score decreasing to 44.44%. This significant performance degradation suggests that the model overfits the training data and fails to generalize to unseen validation data. The low precision indicates an increase in false positives students predicted as at-risk who are not, which is a critical concern in educational contexts where interventions are resource dependent. The recall also reveals that many truly at-risk students go undetected.

In the testing phase, the model's performance improves somewhat, achieving an accuracy of 80.00%, precision of 75.00%, recall of 60.00%, and F1-score of 67.00%. While these values are an improvement from the validation phase, they remain considerably lower than the training performance. The relatively modest recall further implies that a notable portion of at-risk students may still be overlooked. Overall, while the model appears robust during training, the substantial drop in both precision and recall during validation and testing underscores the presence of overfitting and the limitations of using all 12 features.

These findings reinforce the importance of dimensionality reduction through feature selection. The results from Table 5.15 emphasize that although a model trained on all available features may appear strong during training, it risks losing generalizability and interpretability when deployed in practice. Streamlining the input space by identifying the most informative features is crucial for improving model performance, especially in risk-sensitive applications like academic performance prediction.

5.7.2.2 Mutual Information

Table 5.16
Result for Mutual Information Feature Selection Method

Feature Selection Method	Phase	Accuracy	Precision	Recall	F1-Score
Mutual Information	Training	91.3	84.38	96.43	89.55
	Validation	73.33	75.00	50.00	60.00
	Testing	80.00	81.00	80.00	80.00

Table 5.16 presents the performance of the MLP model trained with features selected using Mutual Information, in combination with the ADAM optimizer. In the training phase, the model achieved a solid accuracy of 91.3%, with a high recall of 96.43%, moderate precision of 84.38%, and an F1-score of 89.55%. This performance reflects the model's strength in identifying a majority of positive (at-risk) cases with relatively few false negatives. The slightly lower precision suggests a minor increase in false positives, indicating that while the model is sensitive to detecting at-risk students, it occasionally flags students who are not.

In the validation phase, the accuracy dropped to 73.33%, with precision at 75.00%, recall at 50.00%, and F1-score at 60.00%. The decline in recall highlights that the model became less effective at capturing true at-risk cases during validation, although the maintained precision shows that predictions for at-risk students remained relatively reliable. This shift reflects a more conservative classification behaviour prioritizing correctness in positive predictions over broader sensitivity.

The testing phase results indicate stronger generalization, with accuracy at 80.00%, precision and recall closely aligned at 81.00% and 80.00%, respectively, and a balanced F1-score of 80.00%. This balance across all metrics in unseen data suggests that Mutual Information as a feature selection method successfully identifies a compact and informative subset of features that enhances the model's ability to generalize.

Overall, the results indicate that Mutual Information enables the model to strike a practical trade-off between minimizing false positives and maintaining sufficient recall. The reduced feature set improves performance consistency across validation and testing phases compared to the full-feature baseline. These findings highlight Mutual Information's potential as a viable dimensionality reduction technique that maintains

classification reliability while simplifying the model structure for real-world academic risk prediction scenarios.

5.7.2.3 Recursive Feature Elimination

Table 5.17
Result for RFE Feature Selection Method

Feature Selection Method	Phase	Accuracy	Precision	Recall	F1-Score
RFE	Training	95.00	96.00	95.00	95.00
	Validation	86.67	75.00	75.00	75.00
	Testing	87.00	87.00	87.00	87.00

Table 5.17 presents the performance of the MLP model trained with features selected using Recursive Feature Elimination (RFE), demonstrating notable consistency and strength across all phases of evaluation. In the training phase, the model achieved an accuracy of 95.00%, with precision at 96.00% and recall at 95.00%, resulting in a balanced F1-score of 95.00%. These metrics reflect the model's capacity to effectively learn from the training data while maintaining a near-perfect balance between identifying true positives and avoiding false positives. The high precision implies that the model confidently labels students as at-risk only when sufficient evidence exists, while the high recall indicates that most actual at-risk students are successfully detected.

In the validation phase, the model sustained a strong generalization performance with an accuracy of 86.67%, and all other metrics, precision, recall, and F1-score at 75.00%. This uniformity across validation metrics suggests the model does not exhibit erratic behaviour when evaluated on new, unseen data, and can make informed predictions that are both reliable and stable. Although slightly lower than training metrics, the drop is expected and acceptable, indicating controlled variance and good model regularization.

The testing phase further reinforces the model's robustness and practical utility. All key metrics, accuracy, precision, recall, and F1-score are consistent at 87.00%, representing one of the most balanced performances among all evaluated feature selection methods. This level of consistency across evaluation stages indicates a

minimal overfitting effect and high generalization ability, especially when tested against entirely new data. Such symmetry in performance also reflects that the RFE-selected features capture the core characteristics of student academic engagement, enabling the MLP model to classify both positive and negative outcomes with confidence.

Overall, these results strongly affirm the effectiveness of RFE as a wrapper-based feature selection technique. By systematically eliminating less relevant predictors, RFE helps the model focus on the most informative attributes, thereby reducing complexity without compromising accuracy. The consistent performance across all phases illustrates not only the predictive power of the selected features but also the method's alignment with the structure of the dataset. For educational data mining tasks, where interpretability, reliability, and fairness are paramount, RFE offers a compelling balance of performance and transparency, making it an ideal candidate for applications such as early warning systems and academic risk prediction.

5.7.2.4 Random Forest Importance

Table 5.18
Result for Random Forest Importance Feature Selection Method

Feature Selection Method	Phase	Accuracy	Precision	Recall	F1-Score
Random Forest Importance	Training	97.10	94.12	100.00	96.91
	Validation	66.70	40.00	50.00	44.44
	Testing	80.00	80.00	80.00	79.00

Table 5.18 shows a mixed pattern in the MLP model's performance when trained on features ranked by Random Forest Importance. In the training phase, the model demonstrated excellent learning capacity, achieving 97.10% accuracy, 94.12% precision, 100.00% recall, and a high F1-score of 96.91%. These results indicate that the model was able to perfectly capture all positive instances without any false negatives while maintaining a strong ability to avoid false positives. This suggests that the selected features carry substantial information content, enabling the model to learn the training data patterns with very high fidelity.

However, in the validation phase, a significant performance drop is observed, with accuracy falling to 66.70%, precision declining to 40.00%, recall at 50.00%, and an F1-score of only 44.44%. This substantial degradation indicates overfitting, where the model performs well on the training set but fails to generalize effectively to new, unseen data. The especially low precision suggests that the model produces a high number of false positives during validation, which can be problematic in educational scenarios where incorrectly flagging students as at-risk may result in unnecessary interventions or misplaced resources. This instability implies that the feature subset selected by Random Forest may include attributes that are highly tailored to the training data but do not hold consistent predictive value across datasets.

In contrast, performance during the testing phase shows some recovery, with accuracy, precision, and recall each stabilizing at 80.00%, and an F1-score of 79.00%. This recovery suggests that while the validation results were weak, the model still retained some generalization ability on a fully unseen test set. The balance between precision and recall in the testing stage indicates a more moderated trade-off between false positives and false negatives, which is critical for robust deployment in student performance prediction tasks.

Taken together, these results indicate that while Random Forest Importance can successfully identify powerful features for memorization and fit during training, its selections may not always translate well to generalization. This could be due to its tendency to favour features that split the data well on the training set but may not capture broader patterns. For practical applications, further refinement or hybridization with other feature selection techniques may help mitigate overfitting and improve validation performance. These findings highlight the need for cautious evaluation of feature selection methods, especially in high-stakes domains like academic intervention systems, where both model performance and reliability must be maintained across all data partitions.

5.7.2.5 XGBoost Importance

Table 5.19 presents a profile of performance that closely resembles that of Random Forest Importance, particularly in its contrast between strong training accuracy and weaker validation outcomes.

Table 5.19
Result for XGBoost Importance Feature Selection Method

Feature Selection Method	Phase	Accuracy	Precision	Recall	F1-Score
XGBoost Importance	Training	97.10	94.1	100.00	97.0
	Validation	66.7	40.00	50.00	44.44
	Testing	80.00	80.00	80.00	79.00

In the training phase, the model achieved 97.10% accuracy, 94.1% precision, 100.00% recall, and a very high F1-score of 97.0%, indicating that the features selected by XGBoost Importance allow the model to capture virtually all the relevant patterns in the training set. The perfect recall shows that the model did not miss any positive cases, while the high precision confirms that it made very few false positive predictions. This performance suggests that the selected features are highly expressive for the training data and provide a strong foundation for classification.

However, in the validation phase, the model's performance dropped sharply to 66.7% accuracy, with precision at 40.00%, recall at 50.00%, and an F1-score of 44.44%. This marks a significant decline in generalization capacity and points to overfitting. The low precision implies a high number of false positives, which, in the context of academic performance prediction, could lead to misclassifying students who are not at risk. This is particularly problematic in educational settings where decision-making should be both accurate and equitable. The drop in recall further suggests that the model is also missing some actual positive cases during validation, thereby reducing its usefulness in correctly identifying struggling students.

In the testing phase, performance recovered to some extent, with accuracy, precision, and recall all stabilizing at 80.00%, and an F1-score of 79.00%. These scores indicate a more balanced trade-off between false positives and false negatives, showing that the model can still generalize reasonably well to completely unseen data. The convergence of precision and recall in the testing phase suggests that while XGBoost Importance may not produce the most consistent validation results, its selected features still contribute to a solid final model when evaluated against real-world data.

Overall, these results indicate that the XGBoost-based feature selection approach is highly effective at enhancing fit during training but poses risks of overfitting

due to its sensitivity to data-specific structures. While the testing performance remains acceptable, the validation instability suggests that this method may benefit from complementary strategies such as cross-validation-based feature refinement or integration with regularization techniques. Given these characteristics, XGBoost Importance remains a valuable component in the feature selection toolbox but should be used with careful model validation and error analysis to ensure that overfitting is controlled and real-world deployment remains trustworthy.

5.7.2.6 L1 Lasso

As shown in Table 5.20, in the validation phase, performance remained solid, with accuracy at 86.7%, precision and recall at 75.00%, and a balanced F1-score of 75.00%. While these numbers reflect a slight drop from training, the consistency across precision, recall, and F1-score indicates that the model generalizes well to new data while maintaining a reliable trade-off between false positives and false negatives. The modest drop in recall and precision compared to training performance suggests that some sensitivity to unseen patterns still exists, but not at a level that undermines the model's overall reliability.

Table 5.20
Result for L1 Lasso Feature Selection Method

Feature Selection Method	Phase	Accuracy	Precision	Recall	F1-Score
L1 Lasso	Training	97.71	96.90	96.90	96.90
	Validation	86.7	75.00	75.00	75.00
	Testing	87.00	87.00	87.00	87.00

During the testing phase, the model again achieved 87.00% accuracy, with precision, recall, and F1-score all at 87.00%, marking a nearly ideal balance of classification capabilities. The equal values across all metrics in the testing phase highlight L1 Lasso's effectiveness in maintaining stability and robustness, particularly in real-world deployment where the balance between missing at-risk students and raising false alarms is critical. This performance makes L1 Lasso especially suitable for

educational data applications where predictive reliability and interpretability are essential.

Overall, the L1 Lasso feature selection method demonstrates a compelling ability to reduce dimensionality while preserving predictive performance. The near-equal performance across all phases and metrics confirms that it yields a highly generalizable model. Unlike other methods that may cause high variance or underfitting in the validation or testing stages, L1 Lasso consistently delivers strong precision and recall, which is especially valuable for student performance prediction systems where both false positives and false negatives carry significant intervention costs.

5.7.3 Impact of Feature Selection on Model Generalization

From Table 5.21 the comparison of feature selection methods, focusing on reducing dimensionality from 12 features to 6 as mentioned in the methodology section, reveals significant insights into the impact of feature selection on model performance. Starting with no feature selection (using all 12 features), the model achieves an accuracy of 0.8, but this is accompanied by lower precision (0.75), recall (0.6), and F1-score (0.67). This suggests that while the model is somewhat accurate, its ability to precisely identify and recall relevant instances is limited when all features are used, leading to suboptimal overall performance.

Table 5.21
Sequence of Selected Features For Each Method

Feature Selection Method	Selected Features	Accuracy	Precision	Recall	F1-Score
NONE	All features 12.	0.8	0.75	0.6	0.67
Mutual Information	10,3,2,5,4,9	0.8	0.81	0.8	0.8
RFE Ranking	10,3,7,12,1,8	0.87	0.87	0.87	0.87
Random Forest Importance	10,3,8,2,11,7	0.8	0.8	0.8	0.79
XGBoost Importance	10,7,12,6,2,1	0.8	0.8	0.8	0.79
L1 Lasso	10,7,3,1,5,4	0.87	0.87	0.87	0.87

When applying feature selection methods, several methods stand out for their effectiveness. Mutual Information and Random Forest Importance, while selecting different sets of 6 features, both maintain an accuracy of 0.8. However, Mutual Information shows a slight improvement in precision (0.81) and recall (0.8), resulting in a balanced F1-score of 0.8. This indicates that Mutual Information effectively identifies a subset of features that are highly informative, leading to a more balanced model performance.

RFE Ranking and L1 Lasso emerge as the most effective feature selection methods, both achieving an improved accuracy of 0.87. These methods also result in balanced precision, recall, and F1-scores, all at 0.87. The consistent performance across these metrics suggests that RFE and L1 Lasso are particularly good at identifying the most critical features, enabling the model to perform effectively even with reduced dimensionality. This balanced performance underscores the strength of these methods in maintaining model robustness while reducing complexity.

XGBoost Importance and Random Forest Importance, while useful, seem to plateau at an accuracy of 0.8, with balanced precision and recall (both at 0.8) and a slightly lower F1-score of 0.79. These methods, although effective in maintaining performance, do not surpass the results achieved by RFE Ranking and L1 Lasso, which suggests that they may not be as effective in this specific context for optimizing all performance metrics simultaneously.

The reduction in dimensionality using different feature selection methods reveals that RFE Ranking and L1 Lasso provide the best balance between accuracy, precision, recall, and F1-score. These methods allow the model to maintain high performance while simplifying the feature set, which is crucial for building efficient and interpretable models. Mutual Information and Random Forest Importance also show strong performance but may be slightly less effective than RFE and L1 Lasso in this particular scenario. This analysis highlights the importance of selecting the appropriate feature selection method to optimize model performance while reducing complexity.

5.7.3.1 Comparison of Model Performance Based on Feature Selection Methods

Starting with the baseline model that uses all 12 features without any selection, as presented in Table 5.15 the model achieves an accuracy of 0.8 but suffers from

relatively low precision (0.75), recall (0.6), and F1-score (0.67). This indicates that while the model correctly classifies a substantial portion of the data, it struggles to identify positive instances accurately and consistently, leading to poorer performance metrics overall.

When feature selection methods are applied, there is a noticeable improvement in model performance, particularly in achieving a more balanced set of metrics. Mutual Information, for instance, maintains an accuracy of 0.8 but improves precision to 0.81 and recall to 0.8, resulting in a balanced F1-score of 0.8. This suggests that Mutual Information is effective at selecting features that enhance the model's ability to correctly identify positive instances without sacrificing overall accuracy.

RFE Ranking and L1 Lasso stand out as the most effective feature selection methods, both yielding an improved accuracy of 0.87 along with balanced precision, recall, and F1-scores of 0.87. These results indicate that these methods are particularly effective at identifying the most important features, allowing the model to perform optimally even with a reduced feature set. The consistency across these metrics suggests that RFE and L1 Lasso are successful in enhancing the model's generalizability and robustness by focusing on the most predictive features.

On the other hand, Random Forest Importance and XGBoost Importance methods, while maintaining an accuracy of 0.8, achieve slightly lower precision, recall, and F1-scores of 0.8 and 0.79, respectively. Although these methods are effective in maintaining accuracy, they do not improve the model's balance across all metrics as effectively as RFE and L1 Lasso. This suggests that while Random Forest and XGBoost Importance are good at identifying useful features, they may not always capture the nuanced relationships necessary for optimal precision and recall in this dataset.

The application of feature selection methods significantly impacts model performance, with RFE Ranking and L1 Lasso proving to be the most effective in improving accuracy, precision, recall, and F1-score. These methods provide a balanced approach to feature selection, enhancing the model's ability to generalize and perform well across different metrics. RFE was applied to iteratively rank features by importance, removing the least significant features based on model weightings until an optimal subset was achieved. L1 Lasso incorporated regularization, penalizing coefficients to zero for less important features, thus selecting only those with the strongest predictive value. Mutual Information also shows strong performance, particularly in improving precision. In contrast, while Random Forest and XGBoost

Importance maintain overall accuracy, they are slightly less effective in balancing all performance metrics. This analysis underscores the importance of choosing the right feature selection method to optimize model performance while reducing complexity.

5.7.3.2 Precision-Recall Orientation and Risk Sensitivity in Student Classification

Table 5.22 examines the precision–recall behaviour of the classification model under different feature selection strategies, with particular attention to risk sensitivity in identifying at-risk students. Since academic intervention systems must balance correct identification (recall) with prediction reliability (precision), understanding this trade-off is critical for practical deployment.

Table 5.22
Precision-Recall Comparison on All the Methods

Feature Selection Method	Precision	Recall
NONE	0.75	0.6
Mutual Information	0.81	0.8
RFE Ranking	0.87	0.87
Random Forest Importance	0.8	0.8
XGBoost Importance	0.8	0.8
L1 Lasso	0.87	0.87

5.7.3.3 Redundancy Elimination and Overfitting Control

The results across Table 5.15, Table 5.16. Table 5.17, Table 5.18, Table 5.19 and Table 5.20 collectively demonstrate the extent to which different feature selection methods impact model complexity and overfitting. A key pattern observed is that the baseline model using all 12 original features (Table 5.15) showed strong training performance (91.3% accuracy, 96.43% precision, and 84.38% recall) but suffered substantial degradation during validation (accuracy 66.67%, precision 40.00%) and moderate recovery during testing (accuracy 80.00%, F1-score 67.00%). This sharp performance drop reflects the effect of feature redundancy, where excess input

dimensions introduce noise and reduce the model's ability to generalize, especially in unseen data.

In contrast, feature selection methods like Recursive Feature Elimination (RFE) and L1 Lasso (Table 5.17 and Table 5.20) demonstrated not only improved training accuracy but also remarkable stability across validation and testing phases. Both methods yielded F1-scores above 75% in validation and maintained 87% performance in the test set, suggesting that their selected feature subsets captured the most informative and generalizable patterns while discarding irrelevant or redundant attributes. This outcome validates their strength in controlling overfitting by reducing model variance through effective dimensionality reduction.

On the other hand, tree-based methods such as Random Forest Importance and XGBoost Importance (Table 5.18 and Table 5.8) recorded high training accuracy (97.10%) and perfect recall (100.00%), but both displayed severe drops in validation performance. These patterns indicate overfitting tendencies, where the model memorized training patterns, including noise, but failed to generalize. This can be attributed to the tendency of tree-based algorithms to favour features that split training data well, even if such splits are dataset-specific and not representative of broader trends.

Notably, Mutual Information (Table 5.16) offered a more balanced trade-off. While its training metrics were slightly lower than other methods, the model exhibited relatively consistent performance across phases, with testing F1-score at 80.00% and minimal variance. This suggests that filter-based methods like Mutual Information are effective in removing irrelevant features, although they may not fully address feature interactions, leading to less dramatic gains in training accuracy but better stability.

In conclusion, the evidence confirms that effective feature selection plays a vital role in eliminating redundant inputs and controlling overfitting, especially when deploying models in educational settings where model interpretability and stability are crucial. Methods like RFE and L1 Lasso not only reduce the number of features but also ensure the resulting model generalizes well, providing robust predictions with minimal noise. These techniques help streamline the learning process, mitigate overfitting risks, and support reliable decision-making in identifying at-risk students.

5.7.3.4 Structural Comparison Between Filter, Wrapper, and Embedded Methods

The classification performance analysis in 5.7.2 offers valuable insights into how different structural categories of feature selection methods into filter, wrapper, and embedded, influence model behaviour in terms of learning effectiveness, generalization, and resilience against overfitting.

Filter methods, such as Mutual Information, operate independently of the learning algorithm and assess feature relevance based on statistical properties. The results in Table 5.16 show that Mutual Information achieved balanced performance across phases, with F1-scores of 89.55% (training), 60.00% (validation), and 80.00% (testing). This structure allows it to quickly eliminate irrelevant features, resulting in reduced complexity and improved generalization. However, its limitation is the lack of interaction awareness between features, which can cap performance gains compared to other structurally integrated methods.

Wrapper methods, exemplified by Recursive Feature Elimination (RFE), integrate the learning algorithm directly into the feature selection process. As shown in Table 5.17 , RFE yielded one of the most stable and high-performing models, with F1-scores consistently above 75% across all datasets and no severe overfitting observed. Its structural advantage lies in evaluating feature subsets based on actual model performance, allowing the method to capture dependencies and interactions among features more effectively than filter-based approaches. However, this comes at the cost of greater computational overhead, particularly on larger datasets.

Embedded methods, such as Random Forest Importance, XGBoost Importance, and L1 Lasso, incorporate feature selection as part of the model training process. The performance of Random Forest and XGBoost was strong in training but unstable in validation, signalling overfitting. These tree-based embedded methods tend to assign high importance to features that dominate training splits, which may not generalize well. In contrast, L1 Lasso demonstrated exceptional generalization and symmetry across all phases, with F1-scores of 96.90% (training), 75.00% (validation), and 87.00% (testing). Lasso's embedded structure, which uses regularization to eliminate irrelevant coefficients, provides a natural mechanism for sparsity and overfitting control.

In summary, this structural comparison underscores the importance of aligning the feature selection strategy with the intended model behaviour. Filter methods are fast and simple but may lack nuance. Wrapper methods are more tailored and accurate but

computationally intensive. Embedded methods, especially L1 Lasso, offer a balanced compromise, delivering both dimensionality reduction and robustness when regularization is well-calibrated. These distinctions are critical in educational data mining, where interpretability, computational efficiency, and consistent predictive performance are all necessary for deploying scalable and trustworthy interventions.

5.8 Optimizing the MLP Model and Hyperparameter Tuning

This section presents a comprehensive investigation into the optimization strategies applied to the multilayer perceptron classifier by analysing key hyperparameters that influence model performance. In neural network training, the choice of optimizer and the configuration of training parameters can significantly impact convergence speed, learning stability, and overall predictive accuracy. To better understand these dynamics, this study systematically evaluates eight prominent optimizers under varying settings for epoch convergence, random seed initialization, and hidden neuron count. The goal is to determine how each optimizer responds to these conditions and to identify combinations that yield the most robust and generalizable models.

Three core aspects are explored in detail which are, the number of epochs required for each optimizer to reach convergence across multiple runs, which reflects convergence speed and training efficiency. The second aspect is the sensitivity of each optimizer to changes in random seed, which represents the impact of weight initialization on predictive consistency and the last one is the variation in model performance as the number of hidden neurons is increased from 1 to 10, which highlights the role of architectural complexity in learning representation. Through this evaluation, the study aims to extract meaningful patterns in optimizer behaviour and provide empirically supported recommendations for hyperparameter tuning in binary classification tasks, particularly those involving student learning analytics or similarly structured datasets.

5.8.1 Epoch - Convergence Speed and Stability

Table 5.23
Number of Epochs Required For Each Optimizer To Converge

Run	SDG with Momentum	AdamW	Adagrad	Nadam	AmsGrad
Run 1	66	75	534	39	11
Run 2	23	101	621	91	103
Run 3	32	73	1000	171	84
Run 4	79	69	386	122	75
Run 5	65	151	205	128	23
Run 6	45	54	488	83	100
Run 7	49	15	815	21	92
Run 8	57	108	1000	144	117
Run 9	62	11	1000	45	57
Run 10	105	96	410	77	93
Run 11	57	85	463	107	116
Run 12	26	27	368	127	80
Run 13	52	103	739	60	96
Run 14	65	22	282	117	132
Run 15	46	72	19	26	145
Run 16	69	99	335	105	76
Run 17	84	73	13	85	54
Run 18	84	52	1000	65	82
Run 19	75	96	966	98	11
Run 20	74	23	132	155	96

5.8.2 Random Seed Number

The analysis presented in Table 5.24 illustrates the performance consistency and variability of five optimizers which are SGD with Momentum, AdamW, Adagrad, AmsGrad, and Nadam, when subjected to different random seed initializations: 30, 42, 7917, and 104729. The results highlight how sensitive some optimizers are to initialization, influencing not only classification performance but also convergence behavior as measured by epoch count. Among the tested optimizers, AmsGrad and Nadam consistently delivered strong and balanced classification performance across all seed values, especially at seeds 7917 and 104729, where they maintained accuracy and F1-scores above 93%, with perfect scores at times. These optimizers also demonstrated relatively stable epoch counts, suggesting both robust convergence and resilience to variations in weight initialization.

Table 5.24
Performance Measurement on Different Random Seed Number

	Optimizer	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Epoch**
Random Seed : 30	SDG with Momentum	0.6	0.4545	1	0.625	145
	AdamW	0.6667	0.6	1	0.6667	38
	Adagrad	0.6	0.4545	1	0.625	1000
	AmsGrad	0.8667	0.8	0.8	0.8	268
	Nadam	0.7333	0.5714	0.8	0.6667	119
Random Seed : 42	SDG with Momentum	0.8	0.6667	0.8	0.7273	482
	AdamW	0.7333	0.5714	0.8	0.6667	205
	Adagrad	0.3333	0.3333	1	0.5	1000
	AmsGrad	0.8667	0.8	0.8	0.8	652
	Nadam	0.8	0.6667	0.8	0.7273	223
Random Seed : 7917	SDG with Momentum	0.9333	0.875	1	0.9333	635
	AdamW	0.9333	0.875	1	0.9333	120
	Adagrad	0.5333	0.5	1	0.6667	1000

	Optimizer	Accuracy	Precision	Recall	F1-Score	Epoch
	AmsGrad	0.9333	0.875	1	0.9333	630
	Nadam	0.9333	0.875	1	0.9333	119
Random Seed : 104729	SDG with Momentum	1	1	1	1	1000
	AdamW	1	1	1	1	1000
	Adagrad	0.4667	0.4667	1	0.6364	1000
	AmsGrad	0.9333	0.875	1	0.9333	918
	Nadam	1	1	1	1	837

In contrast, Adagrad showed the weakest and most unstable performance. Regardless of the seed value, it often plateaued at suboptimal results, with accuracy dropping as low as 33.33%, and F1-scores falling to 0.5 or below. Furthermore, Adagrad consistently required the maximum epoch count (1000) to converge, implying inefficiencies in gradient scaling over time. This persistent underperformance suggests that Adagrad may lack the adaptability necessary for handling fluctuating data representations, particularly in binary classification tasks involving nuanced patterns like student performance prediction.

SGD with Momentum and AdamW, while capable of achieving high accuracy under optimal seeds showed considerable sensitivity to initial weight conditions. For example, under seed 30, SGD achieved only 60% accuracy with a low precision of 45.45%, yet under seed 104729 it reached perfect classification. This inconsistency makes them less reliable for general-purpose modeling unless extensive hyperparameter tuning and seed control are applied. AdamW, in particular, exhibited rapid convergence under favourable conditions but required full training duration at other seeds. Such volatility can be problematic in real-world educational systems, where model stability and predictability are essential.

The key insight from this analysis is the importance of evaluating model robustness across multiple random seeds rather than relying on a single run. Single-seed optimization may falsely inflate confidence in an optimizer's generalization ability. In educational applications where model outputs guide interventions, this could lead to overfitting or misleading performance claims. The data here reinforce the robustness of AmsGrad and Nadam as optimizers of choice, offering dependable convergence and

balanced performance across seed scenarios. Their ability to maintain high precision and recall across varying initialization conditions makes them well-suited for sensitive use cases such as academic risk prediction, where both false positives and false negatives carry significant implications.

In conclusion, this seed sensitivity evaluation underscores the need for multiple-run validation protocols during model development. It also supports the broader argument that optimizer selection should not only be based on peak performance but also on stability, convergence efficiency, and repeatability, all of which are crucial for deploying reliable, interpretable models in educational decision-making systems.

5.8.3 Varying Number of Hidden Neuron

The evaluation of hidden layer neuron counts in Table 5.25 offers detailed insights into how model architecture influences classification effectiveness and generalization across training, validation, and testing phases. Using a fixed architecture of two hidden layers and varying the number of neurons in each layer from 1 to 10, this analysis was applied to several optimizers, including SGD with Momentum, AdamW, Adagrad, Nadam, and AmsGrad. The goal was to identify the optimal complexity level that balances learning capacity with generalization ability while avoiding issues of underfitting or overfitting.

For configurations with only 1 to 3 neurons, the models struggled to learn meaningful representations of the data. Across optimizers, these low-capacity settings resulted in low recall and F1-scores, especially in the validation and testing phases. For instance, AdamW with 1–3 neurons showed near-zero recall and precision, signalling extreme underfitting and an inability to separate classes effectively. Even when training accuracy appeared acceptable, the corresponding low F1-scores confirmed that the model failed to detect true positives, which is critical in academic risk prediction tasks where missing at-risk students can lead to intervention failure. This trend is consistent with the theoretical expectation that too few neurons limit the model's capacity to approximate non-linear patterns in complex datasets.

Table 5.25
Varying Number of Hidden Neuron

			Testing			
Optimizer	**Hidden Neurons**	**Epochs**	**Accuracy**	**Precision**	**Recall**	**F1**
SGD_Momentum	1	10	0.4	0.44	0.5	0.47
	2	10	0.53	0.53	1	0.7
	3	10	0.4	0.46	0.75	0.57
	4	10	0.8	0.73	1	0.84
	5	10	0.47	0	0	0
	6	10	0.47	0	0	0
	7	10	0.6	0.67	0.5	0.57
	8	10	0.53	0.53	1	0.7
	9	10	0.47	0.5	0.5	0.5
	10	10	0.53	0.53	1	0.7
AdamW	1	10	0.47	0	0	0
	2	10	0.27	0	0	0
	3	10	0.47	0	0	0
	4	10	0.8	0.73	1	0.84
	5	10	0.53	0.53	1	0.7
	6	10	0.4	0.4	0.25	0.31
	7	10	0.53	0.53	1	0.7
	8	10	0.53	0.53	1	0.7
	9	10	0.53	0.53	1	0.7
	10	10	0.53	0.53	1	0.7
Adagrad	1	10	0.53	0.53	1	0.7
	2	10	0.4	0	0	0
	3	10	0.53	0.53	1	0.7
	4	10	0.53	0.53	1	0.7
	5	10	0.53	0.53	1	0.7
	6	10	0.47	0	0	0

		Testing				
Optimizer	**Hidden Neurons**	**Epochs**	**Accuracy**	**Precision**	**Recall**	**F1**
	7	10	0.53	0.53	1	0.7
	8	10	0.47	0	0	0
	9	10	0.33	0.25	0.13	0.17
	10	10	0.47	0.5	0.63	0.56
Nadam	1	10	0.6	0.75	0.38	0.5
	2	10	0.47	0	0	0
	3	10	0.6	0.58	0.88	0.7
	4	10	0.73	0.67	1	0.8
	5	10	0.6	0.57	1	0.73
	6	10	0.47	0.5	0.75	0.6
	7	10	0.27	0.29	0.25	0.27
	8	10	0.47	0	0	0
	9	10	0.53	0.53	1	0.7
	10	10	0.53	0.53	1	0.7
AmsGrad	1	10	0.47	0.5	0.13	0.2
	2	10	0.53	0.53	1	0.7
	3	10	0.27	0	0	0
	4	10	0.6	0.57	1	0.73
	5	10	0.4	0.44	0.5	0.47
	6	10	0.47	0	0	0
	7	10	0.6	1	0.25	0.4
	8	10	0.47	0	0	0
	9	10	0.47	0	0	0
	10	10	0.4	0.33	0.13	0.18

Performance improved markedly when the hidden neuron count ranged between 4 and 7. This range consistently produced balanced performance across all metrics,

particularly in models optimized with SGD with Momentum and AdamW. For example, SGD with four neurons recorded a testing accuracy of 80%, precision of 72.7%, recall of 100%, and an F1-score of 84.2%, demonstrating strong generalization and minimal overfitting. Similarly, AdamW with 4–5 neurons delivered high precision and F1-scores while maintaining recall, indicating that this configuration was complex enough to capture key patterns without overfitting noise. These results affirm that mid-range configurations enable the model to maintain predictive reliability and stability across unseen data, making them suitable for real-world applications.

Beyond 7 neurons, however, signs of overfitting began to emerge. Several configurations, particularly with 8–10 neurons, achieved perfect or near-perfect classification on training sets but did not translate these gains into the validation or testing phases. For example, models trained with Adagrad and AmsGrad often showed 100% recall during training, but significantly lower precision and unstable F1-scores during testing. This performance decay implies that additional neurons introduced unnecessary complexity, allowing the model to memorize training examples without generalizing to new instances. In practical terms, this compromises the trustworthiness of predictions, especially when applied in educational environments where false positives can misallocate limited resources.

In conclusion, the results of this investigation reveal that the number of hidden neurons is a critical hyperparameter influencing the trade-off between learning efficiency and generalization. The optimal range was identified to be between 4 and 7 neurons, which delivered consistently strong performance across optimizers and data phases. This finding supports the architectural design choice for building reliable, interpretable, and generalizable academic performance classifiers. Moreover, the result highlights the importance of architectural tuning as an essential step in the design of neural networks particularly when working with small to medium-sized educational datasets where model robustness and resource efficiency are vital for deployment and real-world adoption.

5.8.4 Discussion of Optimizing the MLP Model Hyperparameter Tuning

The evaluation of MLP optimization strategies reveals several important patterns concerning the interplay between hyperparameter settings and model performance across training, validation, and testing stages. By analysing convergence

speed, sensitivity to random initialization, and variations in hidden neuron count, this section establishes evidence-based guidelines for constructing robust and generalizable academic prediction models.

On convergence speed, the analysis of 20 independent runs for each optimizer highlights significant variance in training efficiency. AdamW and Nadam consistently converged faster, often requiring fewer than 100 epochs. In contrast, Adagrad frequently hit the upper epoch limit (1,000), indicating inefficient learning dynamics and potential stagnation. This inefficiency was accompanied by poor generalization metrics, reinforcing that convergence speed is a reliable proxy for training stability. SGD with Momentum and AmsGrad showed moderate convergence patterns but required more epochs under certain conditions, making them viable but less efficient alternatives. Thus, from a practical training perspective, AdamW and Nadam offer superior convergence speed and training economy without sacrificing model accuracy.

Regarding random seed sensitivity, testing across four seed values (30, 42, 7917, 104729) revealed that optimizers like AdamW and SGD with Momentum are more sensitive to initialization, with performance fluctuating drastically between runs. For instance, SGD with Momentum recorded F1-scores ranging from 0.625 to 1.0 depending on the seed. This variability raises concerns for deployment reliability, especially in high-stakes educational settings. In contrast, AmsGrad and Nadam displayed remarkable consistency across seeds, achieving balanced performance (accuracy, precision, recall, and F1-score above 0.8) in most runs. These findings stress the importance of seed-agnostic stability in model selection, confirming AmsGrad and Nadam as dependable optimizers for applications where reproducibility and robustness are critical.

In the architectural analysis, varying the number of hidden neurons from 1 to 10 showed a U-shaped performance curve across all optimizers. Very small networks (one to three neurons) lacked the representational capacity to capture complex patterns, leading to underfitting. At the other extreme, networks with 8–10 neurons often overfitted the training data, exhibiting high training metrics but poor generalization on validation and testing sets. Optimal performance emerged in the 4–7 neuron range, with configurations in this band demonstrating balanced precision and recall, particularly under SGD with Momentum and AdamW. This balance is particularly critical in student performance prediction, where both false positives and false negatives entail significant resource allocation and others consequences. Therefore, the optimal range of neuron

counts falls within this mid-range, aligning with theoretical expectations regarding the bias–variance trade-off.

In summary, the hyperparameter tuning process identified three crucial dimensions influencing MLP effectiveness: convergence efficiency, seed robustness, and architectural balance. Among the tested optimizers, AmsGrad and Nadam surfaced as the most robust choices, demonstrating consistent accuracy and convergence with minimal tuning effort. Adagrad underperformed in nearly all dimensions and is not recommended. The ideal model architecture involves using 4–7 hidden neurons with AdamW or Nadam, offering strong generalization, high recall for at-risk student detection, and stability across initialization settings. These insights contribute practical design guidelines for constructing interpretable, scalable, and dependable academic classifiers in real-world educational analytics environments.

5.9 Comparative Analysis of the Optimized MLP Model and Other Intelligent Classification Techniques

This section presents a comparative analysis of the MLP model against five other intelligent classifiers: SVM, CNN, kNN, RNN, and LSTM. The MLP, serving as both baseline and best performer, is evaluated for its robustness and generalization on tabular student performance data. One of the key objectives of this evaluation is to investigate the suitability of each architectural technique in handling structured, non-sequential educational datasets. SVM performs well during training and validation but suffers from overfitting in testing due to its high variance nature. CNN, though effective in spatial pattern recognition, struggles with tabular data, showing poor generalization. The simple kNN model demonstrates strong validation performance but fails on unseen data due to its local sensitivity. RNN exhibits moderate generalization, while LSTM completely underperforms, revealing a structural mismatch with non-sequential datasets. This comparison underscores the architectural alignment of MLP with the dataset and validates its superiority in this educational prediction context.

5.9.1 Classification Accuracy and Generalization

The classification accuracy results across the three evaluation phases which are training, validation, and testing, demonstrate differences in the generalization capability

of each model as presented in Table 5.26. The MLP achieved the most consistent performance, maintaining 98.55% accuracy in both training and validation, and 86.67% in testing. This minimal drop indicates strong generalization and suggests that the model effectively learned from the training data without overfitting. In contrast, models such as the SVM, CNN, kNN achieved comparable accuracy during training and validation (92.75%–95.65%), but their testing accuracy dropped significantly to approximately 53.33%–60.00%. RNN showed slightly better resilience, maintaining 60.00% accuracy in testing, while Long Short-Term Memory exhibited the weakest generalization, with a decline from 82.61% in training to 53.33% in testing.

Table 5.26
Classification Accuracy Results Across The Three Evaluation Phases

Intelligence Technique	**Accuracy (%)**		
	Training	**Validation**	**Testing**
MLP	98.55	86.67	86.67
SVM	95.65	86.67	53.33
CNN	92.75	93.33	60
kNN	92.75	93.33	60
RNN	92.75	93.33	60
LSTM	82.61	73.33	53.33

These results presented in Table 5.26 indicate that while several models are capable of fitting the training data and performing well in validation, only MLP sustains high performance under unseen test conditions. The accuracy degradation observed in SVM, CNN, and kNN suggests susceptibility to overfitting, particularly when the models are not well-aligned with the data's structural properties. LSTM's substantial performance drop implies an architecture mismatch, as it is designed for sequential data, whereas the dataset used in this study is non-sequential and static in nature.

From a practical standpoint, these findings highlight the importance of selecting an ANN architecture that aligns with the data characteristics to ensure reliable generalization. MLP, as a shallow feedforward model, is well-suited for educational datasets with low dimensionality and limited sample sizes. Its consistent performance across all phases positions it as a robust baseline for student performance classification

tasks. The analysis reinforces the notion that architectural simplicity, when matched appropriately with data type, can outperform more complex models in constrained learning environments.

5.9.2 Precision vs Recall

Table 5.27 shows the precision and recall for all models on the training and testing data, respectively. The precision and recall results reveal important differences in the prediction behaviour of each model. The MLP model demonstrated the most balanced performance, achieving perfect training precision (100.00%) and recall (97.30%), while sustaining a strong testing precision of 80.00% and recall of 86.67%. This consistency reflects MLP's reliability in correctly identifying both positive and negative cases. In contrast, models such as SVM, CNN, kNN, and RNN achieved high precision and recall during training (between 92.68% and 94.59%) but exhibited a noticeable decline in precision during testing, falling to 58.33% for SVM, 53.85% for CNN and kNN, and 54.55% for RNN. Despite this drop, their testing recall remained high at 87.50%, suggesting an overemphasis on sensitivity. LSTM recorded the lowest precision overall (81.63% in training and 50.00% in testing) while maintaining the same testing recall as other models.

This trend indicates that several models leaned toward high recall at the expense of precision, making them more likely to misclassify negative instances as positives. While this behavior improves the detection rate of actual positive cases, it also results in a higher number of false positives. Such a trade-off may be problematic in educational settings, where false alarms could lead to unnecessary interventions for students not at risk. MLP's ability to maintain balanced precision and recall across all phases suggests better calibration and a more trustworthy classification pattern under real-world testing conditions.

Table 5.27
The Precision And Recall For All Models

Model	Precision (%)		Recall (%)	
	Training	Testing	Training	Testing
MLP	100	80.3	97.3	86.67
SVM	94.59	58.33	94.59	87.5
CNN	92.68	53.85	92.68	87.5
kNN	92.68	53.85	92.68	87.5
RNN	92.68	54.55	92.68	87.5
LSTM	81.63	50	81.63	87.5

These results underscore the importance of evaluating both precision and recall in tandem rather than in isolation. In education-focused predictive modeling, where both missed detections and false positives can have significant implications, a classifier with a stable trade-off between the two metrics is essential. MLP's consistent performance highlights its suitability for early identification of at-risk students and supports its use in academic decision-support systems. Overall, the findings emphasize that robust real-world classification performance depends not just on accuracy, but also on maintaining equilibrium between sensitivity and specificity.

5.9.3 Impact of Recursive Feature Elimination on Model Performance and Interpretability

The original dataset included 12 features representing diverse aspects of student academic behaviour such as lecture material access, note-taking, exercise activity, and tutorial performance. To reduce redundancy and improve model efficiency, Recursive Feature Elimination (RFE) was applied, resulting in a subset of six influential features: access before lecture (Feature 1), access after lecture (Feature 3), exercises during and after lecture (Features 7 and 8), tutorial correct 3 and above (Feature 10), and tutorial wrong before correct (Feature 12). Observation reveals that this reduced feature set preserved model accuracy across phases, particularly for the MLP model which achieved F1-scores of 98.63% (training), 86.67% (validation), and 80.00% (testing). This minimal performance loss following feature reduction demonstrates that these six

features carry the most essential predictive signal for classifying student outcomes. The inference drawn here is that RFE effectively eliminates non-contributing or redundant attributes while retaining key behavioural indicators, enabling the model to focus learning on the most informative patterns.

Across different architectures, this effect was more pronounced in models such as MLP, SVM, and RNN. SVM and RNN showed solid generalization with validation F1-scores of 86.67% and 93.33%, and testing scores of 63.16% and 70.00%, respectively. These results highlight RFE's compatibility with traditional or moderately deep architectures, suggesting that such models benefit from a leaner, more interpretable input space, especially in structured, non-sequential datasets. Conversely, insight can be drawn from the poor gains observed in CNN and LSTM models, which despite dimensionality reduction, did not show significant improvement and continued to struggle during the testing phase. LSTM, for instance, produced a modest testing F1-score of 63.64%, implying that highly sequential or convolutional models may not be well-suited for datasets that lack temporal or spatial continuity. This reinforces the notion that architectural complexity must be matched to data characteristics, and feature selection alone may not compensate for architectural-data misalignment.

From an educational analytics perspective, the broader insight is that RFE supports the development of simpler, transparent models that maintain performance while offering interpretability, an important factor in risk-sensitive decision-making contexts like student performance monitoring. The ability to identify and justify predictive features aligns well with real-world needs, such as explaining intervention triggers to educators. These findings underscore the importance of applying context-aware feature selection strategies, particularly when model outputs must be actionable, interpretable, and generalizable across student populations in varied academic settings.

5.9.4 Learning Stability and Convergence

In terms of learning stability and convergence as presented in Table 5.28, the MLP model recorded the highest training F1-score at 98.63%, indicating near-perfect learning on the training set. This was followed by SVM with 94.59%, and a consistent grouping of CNN, kNN, and RNN, each achieving 92.68%. Validation F1-scores showed a slightly different trend for CNN, kNN, and RNN led with 93.33%, outperforming MLP and SVM, which both scored 86.67%. However, in the testing

phase, MLP maintained the highest F1-score at 80.00%, showcasing strong stability across all phases. In comparison, RNN followed with 70.00%, CNN and kNN each registered 66.67%, SVM dropped to 63.16%, and LSTM recorded the lowest test F1-score at 63.64%. Notably, LSTM also had the lowest training F1-score at 81.63% and the lowest validation performance at 73.33%, indicating sub-optimal learning and weak convergence behaviour.

The decline in F1-scores from training to testing further illustrates the stability gap among these models. MLP's drop of only 18.63% between training and testing phases suggests it retained much of its learned generalization capacity. RNN showed a moderate drop of 22.68% (from 92.68% to 70.00%), while CNN and kNN each declined by 26.01 points. SVM exhibited the sharpest drop among the top-performing models, decreasing by 31.43% from 94.59% to 63.16%. LSTM's relatively smaller drop of 17.99% (from 81.63% to 63.64%) is misleading, as it started from a much lower training baseline and never reached high validation performance. These figures confirm that while several models learn the training patterns effectively, their learning does not always translate to stable convergence or reliable test-time performance.

Table 5.28
F1-Score for all Intelligence Technique

Model	**F1-Score (%)**		
	Training	**Validation**	**Testing**
MLP	98.63	86.67	80.00
SVM	94.59	86.67	63.16
CNN	92.68	93.33	66.67
kNN	92.68	93.33	66.67
RNN	92.68	93.33	70.00
LSTM	81.63	73.33	63.64

From a practical perspective, these results highlight the importance of selecting architectures that offer stable convergence behaviour across all evaluation phases, especially in small datasets common to educational research. MLP’s minimal variance and high test-time F1-score indicate that it not only learns efficiently but also generalizes predictably, making it a dependable option for deployment. In contrast, the

less consistent performance of CNN, kNN, and particularly LSTM underscores the limitations of deep or sequential models when training data is limited. For data-driven educational interventions, where stable and repeatable predictions are critical, models like MLP that exhibit low volatility and high convergence fidelity are far better suited than those requiring more complex tuning or larger-scale datasets.

5.9.5 Effect of Feature Selection Recursive Feature Elimination

The dataset initially comprised 12 features reflecting student academic engagement, such as note-taking, tutorial performance, and exercise behaviour. To enhance predictive efficiency, Recursive Feature Elimination (RFE) was employed to identify and retain the most informative attributes, reducing the set to six key features: access before lecture (Feature 1), access after lecture (Feature 3), exercises during and after lecture (Features 7 and 8), tutorial correct 3 and above (Feature 10), and tutorial wrong before correct (Feature 12). Post-RFE evaluations confirmed that model performance, especially for MLP, remained strong, with F1-scores of 98.63% (training), 86.67% (validation), and 80.00% (testing), suggesting that the essential predictive signal was preserved despite dimensionality reduction.

SVM and RNN also demonstrated robust validation (86.67% and 93.33%) and testing F1-scores (63.16% and 70.00%), indicating effective generalization with the streamlined feature set. Conversely, deeper architectures like LSTM and CNN saw minimal gains post-RFE, with LSTM's testing F1-score lingering at 63.64%, likely due to their dependence on richer or sequential input structures. These findings highlight RFE's value in improving model interpretability and computational efficiency, especially in low-data scenarios common in educational research. While RFE benefits simpler models like MLP and SVM by enhancing generalization and reducing over-fitting, its impact on complex deep learning models may be limited without additional feature engineering aligned to their architecture, reinforcing the importance of tailoring feature selection strategies to the model and data characteristics.

5.9.6 Overfitting and Model Robustness

The observed differences in model performance across training, validation, and testing phases offer strong indicators of overfitting behaviour and model robustness.

The MLP architecture stood out with the most consistent results, showing only a modest drop from a training F1-score of 98.63% to 86.67% in validation and 80.00% in testing, demonstrating reliable generalization to unseen data. In contrast, models like SVM and CNN, despite high training F1-scores (94.59% and 92.68% respectively), experienced significant declines during testing, falling to 63.16% and 66.67%. Similar trends were observed for kNN and RNN, with testing drops of approximately 26 percentage points. The LSTM model exhibited the steepest overfitting, starting with a relatively low training F1-score of 81.63% and dropping to just 63.64% in testing, even with dropout applied. These patterns suggest that complex architectures like LSTM and CNN are particularly sensitive to small datasets and non-sequential input, likely learning patterns too specific to the training data without developing generalized representations. Although techniques like dropout and early stopping were employed, they proved insufficient without proper architectural alignment. In contrast, MLP's resilience across all phases affirms its suitability for real-world educational analytics, especially in low-data scenarios. This emphasizes the need for model selection strategies that prioritize generalization, architectural simplicity, and alignment with data characteristics to prevent overfitting and ensure reliable deployment.

5.10 Chapter Summary

This chapter presented a comprehensive evaluation of the experimental outcomes derived from the application of ADASYN, feature selection strategies, MLP architecture design, and optimizer tuning for predicting student academic performance using online engagement data. The findings validate the robustness and generalizability of the MLP-based classifier when augmented with proper data balancing, dimensionality reduction, and optimization techniques.

The application of ADASYN significantly improved the model's performance on the minority class. Statistical validation using the Kolmogorov–Smirnov (K-S) test confirmed the distributional consistency between the original and synthetic data, with p-values exceeding 0.05 for all features. This substantiates the effectiveness of ADASYN in balancing the dataset without introducing distributional bias, leading to fairer and more accurate predictions.

The comprehensive analysis of the feature selection method identified that there are 4 activities that contribute to students' final grades to be GPA $\geq$ 3.00 or GPA < 3.00

in a C programming course at UiTM. Specifically, but not in order are Feature 2 (Access to Lecture Slide and Additional Notes During Lecture) , Feature 7 (Exercise During Lecture), Feature 3 (Access to Lecture Slide and Additional Notes After Lecture) and Feature 10 (Tutorial Correct 3 and Above), had consistently emerged as the top feature across the analysis.

These features were shown to be the most predictive across multiple experiments and classifiers, supporting model simplification while maintaining strong classification metrics (Accuracy, Precision, Recall, and F1-score). Their presence in the reduced feature sets underlines their pedagogical significance in students' learning behaviour and outcomes.

The performance of various optimizers on the MLP architecture was also evaluated, with AdamW and Nadam emerging as the most effective. These optimizers achieved high testing accuracy (~80%), stable convergence across runs, and superior F1-scores (up to 86.95%), demonstrating their capability in producing reliable and generalizable models. Conversely, SGD with Momentum underperformed, showing signs of underfitting and high sensitivity to random seed initialization.

In terms of MLP architecture, experiments with varying hidden neurons revealed that a configuration with 4–7 hidden units strike the optimal balance between learning capacity and generalization. This supports the effectiveness of a relatively shallow but well-tuned MLP network for educational datasets, especially when working with small sample sizes.

Lastly, comparative analysis against other classifiers (SVM, CNN, kNN, RNN, LSTM) highlighted the superiority of MLP in generalization and stability. While other models showed strong training and validation results, they struggled with overfitting and poor test set performance. MLP remained the most robust across all evaluation metrics, underscoring its suitability for static, structured educational data.

In summary, the integration of ADASYN, feature selection methods, optimal MLP architecture, and appropriate optimizers yielded a highly effective and interpretable predictive model. These findings contribute toward the development of intelligent student performance monitoring systems and early intervention frameworks in higher education.

CHAPTER 6
CONCLUSION & FUTURE RECOMMENDATIONS

6.1 Conclusion

This section presents the key findings of the study based on the four research objectives.

The analysis of feature importance revealed that four academic engagement activities significantly influenced students' final grades in the C programming course at UiTM. These features were: (Feature 2 - Access to Lecture Slide and Additional Notes During Lecture), (Feature 7 - Exercise During Lecture), (Feature 3 - Access to Lecture Slide and Additional Notes After Lecture), and (Feature 10 - Tutorial Correct 3 and above).

These results underscore the importance of independent learning time and active engagement with course materials beyond scheduled lectures. Students who regularly accessed lecture notes, completed exercises, and tackled tutorial questions on their own demonstrated deeper understanding and cognitive mastery. The ability to answer tutorial questions without supervision reflects higher-order cognitive, highlighting self-regulated learning as a key driver of academic success and resilience.

These findings align closely with UiTM's Outcome-Based Education framework and the Malaysian Qualifications Framework, both of which emphasize independent learning, critical thinking, and problem-solving skills. The integration of Open and Distance Learning further highlights the importance of providing flexible and accessible learning resources that support continuous improvement. Collectively, these outcomes reinforce institutional efforts to cultivate lifelong learning and ensure graduates are well-equipped to adapt in an increasingly dynamic and technology-driven environment.

To address the class imbalance within the original dataset, where students with GPA below 3.00 were underrepresented, this study employed the ADASYN technique to generate synthetic data points for the minority class. This improved the distribution of class labels, leading to a more inclusive dataset and enhancing the fairness of the prediction model.

The validity of the synthetic data was confirmed using Kolmogorov–Smirnov tests, boxplots, and histograms, which showed that the ADASYN-generated samples closely matched the distribution of the original data without causing statistical drift. When combined with feature selection methods such as Recursive Feature Elimination and L1 Regularization Lasso, ADASYN contributed to the development of a more robust and generalizable predictive model.

The MLP model developed in this study was constructed using 12 features reflecting student engagement throughout the semester. It successfully captured the complex, non-linear relationships between academic behaviour and final outcomes, producing strong predictive performance across multiple evaluation metrics. Optimizers such as AdamW and Nadam yielded the best results, confirming the importance of training dynamics in MLP effectiveness. Even with a small sample size the MLP demonstrated generalization and stability, particularly when combined with early stopping and dimensionality reduction through RFE. These findings confirm the MLP's suitability as a predictive tool in academic settings and support its use in future data-driven education interventions.

This study conducted a comparative evaluation of six classifiers MLP, SVM, CNN, kNN, RNN, and LSTM, using engagement-related features reduced via Recursive Feature Elimination. All models were assessed using standard metrics including accuracy, precision, recall, and F1-score, with emphasis on their ability to generalize to unseen data. Among these, the Multilayer Perceptron consistently outperformed the others, achieving a testing accuracy of 86.67% and an F1-score of 80.00%. MLP showed minimal performance drop from training to testing, indicating strong generalization and low risk of overfitting. MLP further demonstrated a well-balanced trade-off between precision and recall, making it more suitable for real-world academic applications where both false positives and false negatives have consequences. Its consistent convergence and stable F1-scores reinforced its reliability for use in early academic warning systems. These results validate MLP as the most effective, generalizable, and interpretable model for the data set.

6.2 Future Recommendations

While this study successfully developed a predictive framework using real-time academic engagement features and a Multilayer Perceptron model, several limitations

present opportunities for further investigation. Firstly, the sample size was relatively small (N = 99) and limited to students from a single programming course at one institution. Future studies should validate the model across larger, multi-institutional datasets and diverse academic disciplines to ensure broader generalizability and to assess how course type, student background, or institutional culture may influence prediction performance.

Secondly, this study focused solely on structured academic activity data, excluding potentially valuable inputs such as demographic information, motivation levels, affective states, or time-sequenced behavioural patterns. Future work could enhance predictive robustness by incorporating a wider range of features, including longitudinal learning trajectories, participation in co-curricular activities, or qualitative feedback indicators. Additionally, the current approach treated learning data as static snapshots. Future research could explore sequence-based models like LSTM or Transformer-based architectures to capture time-dependent patterns in student engagement more effectively.

Another promising avenue involves the integration of Prompt Engineering (PE) techniques to address the current model's limited ability to provide interpretive or actionable feedback. By coupling behavioural analytics with AI-generated prompts, researchers can explore how generative systems might enhance student interventions in real time. For example, adaptive feedback mechanisms could be developed for students flagged as at-risk, offering personalized nudges or reflective questions based on their interaction profile. This could transform predictive models from passive classifiers into proactive learning support systems

Lastly, while the MLP model showed strong performance in this study, further exploration of alternative architectures and ensemble methods is warranted. Comparing hybrid models that combine MLP with rule-based or decision-tree components may reveal improved interpretability, an essential factor in educational applications where model transparency can support instructor decision-making. As AI becomes increasingly embedded in educational ecosystems, future work should also consider ethical dimensions, including data privacy, model fairness, and the implications of automated academic risk labelling on student motivation and equity.

REFERENCES

[1] UNESCO Education for Sustainable Development – Country Progress Report: Malaysia, "https://en.unesco.org/themes/education-sustainable-development," UNESCO.

[2] Ministry of Education Malaysia. (2013). Malaysia Education Blueprint 2013–2025 (Preschool to Post-Secondary Education). Putrajaya: Ministry of Education, "Malaysia Education Blueprint 2013–2025 (Preschool to Post-Secondary Education)."

[3] Ministry of Finance Malaysia, "Ucapan belanjawan https://www.parlimen.gov.my/resources/files/rsaindex/pdf/ub23%20BM.pdf," 2023.

[4] Ministry of Finance Malaysia, "Bajet Bahasa Melayu 2024 https://belanjawan.mof.gov.my/ms/2024," 2024.

[5] Ministry of Finance, "https://belanjawan.mof.gov.my/ms/."

[6] Ministry of Education Malaysia (2023), "Malaysia Education Blueprint Annual Report 2023."

[7] N. Norazlan, S. Yusuf, and F. Mohamed Hamoud Al-Majdhoub, "The financial problems and academic performance among public university students in Malaysia," *The Asian Journal of Professional & Business Studies*, vol. 1, no. 2, Dec. 2020, doi: 10.61688/ajpbs.v1i2.52.

[8] G. Ghaleb A. El Refae, A. Kaba, and S. Eletter, "The The Impact of Demographic Characteristics on Academic Performance: Face-to-Face Learning Versus Distance Learning Implemented to Prevent the Spread of COVID-19," *The International Review of Research in Open and Distributed Learning*, vol. 22, no. 1, pp. 91–110, Mar. 2021, doi: 10.19173/irrodl.v22i1.5031.

[9] G. R. Pike and J. L. Saupe, "Does high school matter? An analysis of three methods of predicting first-year grades," *Res. High. Educ.*, vol. 43, no. 2, pp. 187–207, 2002, doi: 10.1023/A:1014419724092.

[10] A. Garcia *et al.*, "A System for Adaptation of Educational Contents to Learners and their Mobile Device," in *2011 IEEE 11th International Conference on Advanced Learning Technologies*, IEEE, Jul. 2011, pp. 484–485. doi: 10.1109/ICALT.2011.151.

[11] B. K. Francis and S. S. Babu, "Predicting Academic Performance of Students

Using a Hybrid Data Mining Approach," *J. Med. Syst.*, vol. 43, no. 6, 2019, doi: 10.1007/s10916-019-1295-4.

[12] R. O. Aluko, E. I. Daniel, O. Shamsideen Oshodi, C. O. Aigbavboa, and A. O. Abisuga, "Towards reliable prediction of academic performance of architecture students using data mining techniques," *Journal of Engineering, Design and Technology*, vol. 16, no. 3, pp. 385–397, Jul. 2018, doi: 10.1108/JEDT-08-2017-0081.

[13] A. Siddique, A. Jan, F. Majeed, A. I. Qahmash, N. N. Quadri, and M. O. A. Wahab, "Predicting Academic Performance Using an Efficient Model Based on Fusion of Classifiers," *Applied Sciences*, vol. 11, no. 24, p. 11845, Dec. 2021, doi: 10.3390/app112411845.

[14] R. Steinmayr, M. Ziegler, and B. Träuble, "Do intelligence and sustained attention interact in predicting academic achievement?," *Learn. Individ. Differ.*, vol. 20, no. 1, pp. 14–18, Feb. 2010, doi: 10.1016/j.lindif.2009.10.009.

[15] S. Alghamdi, "MONITORING STUDENT ATTENDANCE USING A SMART SYSTEM AT TAIF UNIVERSITY," *International Journal of Computer Science and Information Technology*, vol. 11, no. 01, pp. 107–115, Feb. 2019, doi: 10.5121/ijcsit.2019.11108.

[16] J. López-Zambrano, J. Lara Torralbo, and C. Romero, "Early Prediction of Student Learning Performance Through Data Mining: A Systematic Review," *Psicothema*, vol. 3, no. 33, pp. 456–465, Aug. 2021, doi: 10.7334/psicothema2021.62.

[17] A. A. Ab Rahim and N. Buniyamin, "Mitigating Imbalanced Classification Problems in Academic Performance with Resampling Methods," *Journal of Electrical & Electronic Systems Research*, pp. 45–56, Oct. 2023, doi: 10.24191/jeesr.v23i1.006.

[18] B. Alnasyan, M. Basheri, and M. Alassafi, "The power of Deep Learning techniques for predicting student performance in Virtual Learning Environments: A systematic literature review," *Computers and Education: Artificial Intelligence*, vol. 6, p. 100231, Jun. 2024, doi: 10.1016/j.caeai.2024.100231.

[19] Engineering Programme Accreditation Standard 2024 Engineering Accreditation Council Malaysia, "https://eac.org.my/v2/wp-content/uploads/2024/08/Engineering-Programme-Accreditation-Standard-2024.pdf," https://eac.org.my/v2/wp-content/uploads/2024/08/Engineering-

Programme-Accreditation-Standard-2024.pdf?__cf_chl_tk=fKX1E4Eono3KZLscD4PVyosIxS9OAL83VnMLf8sEyKQ-1752726365-1.0.1.1-LmFelNJh4GcRu4Qvdf0lg7A9x8HsriTN7A2G40cSfUg.

[20] M. Yağcı, "Educational data mining: prediction of students' academic performance using machine learning algorithms," *Smart Learning Environments*, vol. 9, no. 1, p. 11, Dec. 2022, doi: 10.1186/s40561-022-00192-z.

[21] R. Asif, A. Merceron, S. A. Ali, and N. G. Haider, "Analyzing undergraduate students' performance using educational data mining," *Comput. Educ.*, vol. 113, pp. 177–194, Oct. 2017, doi: 10.1016/j.compedu.2017.05.007.

[22] K. Shakeel and N. Anwer Butt, "Educational Data Mining to Reduce Student Dropout Rate by Using Classification," in *253rd OMICS International Conference on Big Data Analysis & Data Mining*, 2015.

[23] G. S. Gowri, R. Thulasiram, and M. A. Baburao, "Educational Data Mining Application for Estimating Students Performance in Weka Environment," *IOP Conf. Ser. Mater. Sci. Eng.*, 2017, doi: 10.1088/1757-899X/263/3/032002.

[24] R. K. Rambola, M. Inamke, and S. Harne, "Literature review- techniques and algorithms used for various applications of educational data mining (EDM).," *2018 4th International Conference on Computing Communication and Automation, ICCCA 2018*, 2018.

[25] M. F. Musso, E. C. Cascallar, N. Bostani, and M. Crawford, "Identifying Reliable Predictors of Educational Outcomes Through Machine-Learning Predictive Modeling," *Front. Educ. (Lausanne).*, vol. 5, Jul. 2020, doi: 10.3389/feduc.2020.00104.

[26] H. Kishan Das Menon and V. Janardhan, "Machine learning approaches in education," in *Materials Today: Proceedings*, Elsevier Ltd, 2020, pp. 3470–3480. doi: 10.1016/j.matpr.2020.09.566.

[27] H. Zeineddine, U. Braendle, and A. Farah, "Enhancing prediction of student success: Automated machine learning approach," *Computers and Electrical Engineering*, vol. 89, Jan. 2021, doi: 10.1016/j.compeleceng.2020.106903.

[28] C. F. Rodríguez-Hernández, E. Cascallar, and E. Kyndt, "Socio-economic status and academic performance in higher education: A systematic review," Feb. 01, 2020, *Elsevier Ltd*. doi: 10.1016/j.edurev.2019.100305.

[29] M. G. Gallego, A. P. Perez de los Cobos, and J. C. G. Gallego, "Identifying

Students at Risk to Academic Dropout in Higher Education," *Educ. Sci. (Basel).*, vol. 11, no. 8, p. 427, Aug. 2021, doi: 10.3390/educsci11080427.

[30] I. Sandoval-Palis, D. Naranjo, J. Vidal, and R. Gilar-Corbi, "Early Dropout Prediction Model: A Case Study of University Leveling Course Students," *Sustainability*, vol. 12, no. 22, p. 9314, Nov. 2020, doi: 10.3390/su12229314.

[31] M. Gadhavi Smt Chandaben Mohanbhai Patel and C. Patel Smt Chandaben Mohanbhai Patel, "STUDENT FINAL GRADE PREDICTION BASED ON LINEAR REGRESSION."

[32] M. A. Al-Barrak and M. Al-Razgan, "Predicting Students Final GPA Using Decision Trees: A Case Study," *International Journal of Information and Education Technology*, vol. 6, no. 7, pp. 528–533, 2016, doi: 10.7763/ijiet.2016.v6.745.

[33] J. Mesarić and D. Šebalj, "Decision trees for predicting the academic success of students," *Croatian Operational Research Review*, vol. 7, no. 2, pp. 367–388, Dec. 2016, doi: 10.17535/crorr.2016.0025.

[34] C. F. Rodríguez-Hernández, M. Musso, E. Kyndt, and E. Cascallar, "Artificial neural networks in academic performance prediction: Systematic implementation and predictor evaluation," *Computers and Education: Artificial Intelligence*, vol. 2, p. 100018, Jan. 2021, doi: 10.1016/j.caeai.2021.100018.

[35] F. N. Osman, M. A. Abdul Aziz, and M. N. Taib, "Comparative Evaluation of Feature Selection Algorithms for Predictive Modeling of Academic Performance Outcomes," in *2024 IEEE 13th International Conference on Engineering Education (ICEED)*, IEEE, Nov. 2024, pp. 1–6. doi: 10.1109/ICEED62316.2024.10923823.

[36] F. N. Osman, M. A. A. Aziz, and M. N. Taib, "Enhancing Students' Academic Performance Classifier using ADASYN and MLP," in *2024 IEEE 22nd Student Conference on Research and Development (SCOReD)*, 2024, pp. 221–226. doi: 10.1109/SCOReD64708.2024.10872712.

[37] E. Alyahyan and D. Dusteaor, "Decision trees for very early prediction of student's achievement," in *2020 2nd International Conference on Computer and Information Sciences, ICCIS 2020*, Institute of Electrical and Electronics Engineers Inc., Oct. 2020. doi: 10.1109/ICCIS49240.2020.9257646.

[38] F. Giannakas, C. Troussas, I. Voyiatzis, and C. Sgouropoulou, "A deep learning classification framework for early prediction of team-based academic

performance," *Appl. Soft Comput.*, vol. 106, p. 107355, Jul. 2021, doi: 10.1016/j.asoc.2021.107355.

[39] J. Biggs, "Enhancing teaching through constructive alignment," *High. Educ. (Dordr).*, vol. 32, no. 3, pp. 347–364, Oct. 1996, doi: 10.1007/BF00138871.

[40] R. M. Harden, "Outcome-Based Education: the future is today," *Med. Teach.*, vol. 29, no. 7, pp. 625–629, Jan. 2007, doi: 10.1080/01421590701729930.

[41] D. T. Tempelaar, B. Rienties, and B. Giesbers, "In search for the most informative data for feedback generation: Learning analytics in a data-rich context," *Comput. Human Behav.*, vol. 47, pp. 157–167, Jun. 2015, doi: 10.1016/j.chb.2014.05.038.

[42] R. Ferguson, "Learning analytics: drivers, developments and challenges," *International Journal of Technology Enhanced Learning*, vol. 4, no. 5/6, p. 304, 2012, doi: 10.1504/IJTEL.2012.051816.

[43] F. Ahmad, N. H. Ismail, and A. A. Aziz, "The prediction of students' academic performance using classification data mining techniques," *Applied Mathematical Sciences*, vol. 9, pp. 6415–6426, 2015, doi: 10.12988/ams.2015.53289.

[44] N. Valaei and M. B. Baroto, "Modelling continuance intention of citizens in government Facebook page: A complementary PLS approach," *Comput. Human Behav.*, vol. 73, pp. 224–237, Aug. 2017, doi: 10.1016/j.chb.2017.03.047.

[45] R. M. Harden, "Outcome-Based Education: the future is today," *Med. Teach.*, vol. 29, no. 7, pp. 625–629, Jan. 2007, doi: 10.1080/01421590701729930.

[46] G. Siemens and R. S. J. d. Baker, "Learning analytics and educational data mining," in *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, New York, NY, USA: ACM, Apr. 2012, pp. 252–254. doi: 10.1145/2330601.2330661.

[47] C. Liu, Y. Feng, and Y. Wang, "An innovative evaluation method for undergraduate education: an approach based on *BP* neural network and stress testing," *Studies in Higher Education*, vol. 47, no. 1, pp. 212–228, Jan. 2022, doi: 10.1080/03075079.2020.1739013.

[48] F. Salas and J. Caldas, "Predicting undergraduate academic performance in a leading Peruvian university: A machine learning approach," *Educación*, vol. 33, no. 64, pp. 55–85, Apr. 2024, doi: 10.18800/educacion.202401.M003.

[49] R. O. Aluko, E. I. Daniel, O. Shamsideen Oshodi, C. O. Aigbavboa, and A. O. Abisuga, "Towards reliable prediction of academic performance of architecture

students using data mining techniques," *Journal of Engineering, Design and Technology*, vol. 16, no. 3, pp. 385–397, Jun. 2018, doi: 10.1108/JEDT-08-2017-0081.

[50] Pedro. Fenollar, Sergio. Román, and P. J. Cuestas, "University students' academic performance: An integrative conceptual framework and empirical analysis," *British Journal of Educational Psychology*, vol. 77, no. 4, pp. 873–891, Dec. 2007, doi: 10.1348/000709907X189118.

[51] Y. Kurniawan and E. Halim, "Use data warehouse and data mining to predict student academic performance in schools: A case study (perspective application and benefits)," in *Proceedings of 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE)*, IEEE, Aug. 2013, pp. 98–103. doi: 10.1109/TALE.2013.6654408.

[52] X. Li, Y. Zhang, H. Cheng, M. Li, and B. Yin, "Student achievement prediction using deep neural network from multi-source campus data," *Complex & Intelligent Systems*, vol. 8, no. 6, pp. 5143–5156, Dec. 2022, doi: 10.1007/s40747-022-00731-8.

[53] A. R. Gautam and R. Sharma, "Impact of Near Real Time Data on Data Science Model Predictions," *International Journal of Computer Sciences and Engineering*, vol. 12, no. 4, pp. 55–60, Apr. 2024, doi: 10.26438/ijcse/v12i4.5560.

[54] P. Praneeth, N. Naresh, S. Kiran, A. Anvesh, and D. Deepak, "Smart Attendance and Behavioural Analysis," *INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT*, vol. 09, no. 03, pp. 1–9, Mar. 2025, doi: 10.55041/IJSREM42936.

[55] J.-L. Hung, K. Rice, J. Kepka, and J. Yang, "Improving predictive power through deep learning analysis of K-12 online student behaviors and discussion board content," *Inf. Discov. Deliv.*, vol. 48, no. 4, pp. 199–212, May 2020, doi: 10.1108/IDD-02-2020-0019.

[56] D. Sun, T. Li, F. You, M. Hu, and Z. Li, "Prediction of learning behavior characters of MOOC's data based on time series analysis," *J. Phys. Conf. Ser.*, vol. 1994, no. 1, p. 012009, Aug. 2021, doi: 10.1088/1742-6596/1994/1/012009.

[57] C. C. Yin Albert *et al.*, "Identifying and Monitoring Students' Classroom Learning Behavior Based on Multisource Information," *Mobile Information Systems*, vol. 2022, pp. 1–8, Aug. 2022, doi: 10.1155/2022/9903342.

[58] A. Charitopoulos, M. Rangoussi, and D. Koulouriotis, “Blending E-Learning with Hands-on Laboratory Instruction in Engineering Education,” *International Journal of Emerging Technologies in Learning (iJET)*, vol. 17, no. 20, pp. 213–230, Oct. 2022, doi: 10.3991/ijet.v17i20.33141.

[59] Y. Zhang and J. Li, “Deep learning-based model for predicting student learning behavior: A pathway to early intervention and enhanced outcomes,” *Journal of Computational Methods in Sciences and Engineering*, vol. 25, no. 3, pp. 2822–2835, May 2025, doi: 10.1177/14727978251322332.

[60] M. Saxena, R. Pillai, and J. Mostow, “Relating Children’s Automatically Detected Facial Expressions to Their Behavior in RoboTutor,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018, doi: 10.1609/aaai.v32i1.12190.

[61] U. Ashfaq, Dr. B. P. M., and R. Mafas, “Managing Student Performance: A Predictive Analytics using Imbalanced Data,” *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 8, no. 6, pp. 2277–2283, Mar. 2020, doi: 10.35940/ijrte.E7008.038620.

[62] Haibo He and E. A. Garcia, “Learning from Imbalanced Data,” *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009, doi: 10.1109/TKDE.2008.239.

[63] R. Ghorbani and R. Ghousi, “Comparing Different Resampling Methods in Predicting Students’ Performance Using Machine Learning Techniques,” *IEEE Access*, vol. 8, pp. 67899–67911, 2020, doi: 10.1109/ACCESS.2020.2986809.

[64] N. Rachburee and W. Punlumjeak, “Oversampling technique in student performance classification from engineering course,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 4, p. 3567, Aug. 2021, doi: 10.11591/ijece.v11i4.pp3567-3574.

[65] G. Ryan, P. Katarina, and D. Suhartono, “MBTI Personality Prediction Using Machine Learning and SMOTE for Balancing Data Based on Statement Sentences,” *Information*, vol. 14, no. 4, p. 217, 2023, doi: 10.3390/info14040217.

[66] V. Flores, S. Heras, and V. Julian, “Comparison of Predictive Models with Balanced Classes Using the SMOTE Method for the Forecast of Student Dropout in Higher Education,” *Electronics (Basel).*, vol. 11, no. 3, p. 457, Feb. 2022, doi: 10.3390/electronics11030457.

[67] Z. Chen, L. Zhou, and W. Yu, “ADASYN−Random Forest Based Intrusion Detection Model,” in *2021 4th International Conference on Signal Processing and Machine Learning*, New York, NY, USA: ACM, Aug. 2021, pp. 152–159. doi: 10.1145/3483207.3483232.

[68] G. Ahmed *et al.*, “DAD-Net: Classification of Alzheimer’s Disease Using ADASYN Oversampling Technique and Optimized Neural Network,” *Molecules*, vol. 27, no. 20, p. 7085, Oct. 2022, doi: 10.3390/molecules27207085.

[69] G. W. Jeon, Y. S. Lee, W.-H. Hahn, and Y. H. Jun, “A Predictive Model for Perinatal Brain Injury Using Machine Learning Based on Early Birth Data,” *Children*, vol. 11, no. 11, p. 1313, Oct. 2024, doi: 10.3390/children11111313.

[70] Haibo H, Yang B, E. A. Garcia, and Shutao L, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, IEEE, Jun. 2008, pp. 1322–1328. doi: 10.1109/IJCNN.2008.4633969.

[71] S. Zhang, Y. Yuan, Z. Yao, X. Wang, and Z. Lei, “Improvement of the Performance of Models for Predicting Coronary Artery Disease Based on XGBoost Algorithm and Feature Processing Technology,” *Electronics (Basel).*, vol. 11, no. 3, p. 315, Jan. 2022, doi: 10.3390/electronics11030315.

[72] F. Gurcan and A. Soylu, “Synthetic Boosted Resampling Using Deep Generative Adversarial Networks: A Novel Approach to Improve Cancer Prediction from Imbalanced Datasets,” *Cancers (Basel).*, vol. 16, no. 23, p. 4046, Dec. 2024, doi: 10.3390/cancers16234046.

[73] S.-C. Tsai, C.-H. Chen, Y.-T. Shiao, J.-S. Ciou, and T.-N. Wu, “Precision education with statistical learning and deep learning: a case study in Taiwan,” *International Journal of Educational Technology in Higher Education*, vol. 17, no. 1, p. 12, Dec. 2020, doi: 10.1186/s41239-020-00186-2.

[74] I. M. Alkhawaldeh, I. Albalkhi, and A. J. Naswhan, “Challenges and limitations of synthetic minority oversampling techniques in machine learning,” *World J. Methodol.*, vol. 13, no. 5, pp. 373–378, Dec. 2023, doi: 10.5662/wjm.v13.i5.373.

[75] S. Limanto, J. L. Buliali, and A. Saikhu, “GLoW SMOTE-D: Oversampling Technique to Improve Prediction Model Performance of Students Failure in Courses,” *IEEE Access*, vol. 12, pp. 8889–8901, 2024, doi: 10.1109/ACCESS.2024.3351569.

[76] A. Gonzalez-Nucamendi, J. Noguez, L. Neri, V. Robledo-Rella, and R. M. G. García-Castelán, "Predictive analytics study to determine undergraduate students at risk of dropout," *Front. Educ. (Lausanne).*, vol. 8, Oct. 2023, doi: 10.3389/feduc.2023.1244686.

[77] J. M. Aiken, R. De Bin, M. Hjorth-Jensen, and M. D. Caballero, "Predicting time to graduation at a large enrollment American university," *PLoS One*, vol. 15, no. 11, p. e0242334, Nov. 2020, doi: 10.1371/journal.pone.0242334.

[78] C. H. Cho, Y. W. Yu, and H. G. Kim, "A Study on Dropout Prediction for University Students Using Machine Learning," *Applied Sciences*, vol. 13, no. 21, p. 12004, Nov. 2023, doi: 10.3390/app132112004.

[79] A. Anggrawan, H. Hairani, and C. Satria, "Improving SVM Classification Performance on Unbalanced Student Graduation Time Data Using SMOTE," *International Journal of Information and Education Technology*, vol. 13, no. 2, pp. 289–295, 2023, doi: 10.18178/ijiet.2023.13.2.1806.

[80] T. R. Shaha *et al.*, "Feature group partitioning: an approach for depression severity prediction with class balancing using machine learning algorithms," *BMC Med. Res. Methodol.*, vol. 24, no. 1, p. 123, Jun. 2024, doi: 10.1186/s12874-024-02249-8.

[81] M. Mujahid *et al.*, "Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering," *J. Big Data*, vol. 11, no. 1, p. 87, Jun. 2024, doi: 10.1186/s40537-024-00943-4.

[82] M. Imani, Z. Ghaderpour, and M. Joudaki, "The Impact of SMOTE and ADASYN on Random Forests and Advanced Gradient Boosting Techniques in Telecom Customer Churn Prediction," Mar. 05, 2024. doi: 10.20944/preprints202403.0213.v1.

[83] A. J. Fernandez-Garcia, J. C. Preciado, F. Melchor, R. Rodriguez-Echeverria, J. M. Conejero, and F. Sanchez-Figueroa, "A Real-Life Machine Learning Experience for Predicting University Dropout at Different Stages Using Academic Data," *IEEE Access*, vol. 9, pp. 133076–133090, 2021, doi: 10.1109/ACCESS.2021.3115851.

[84] D. Sun *et al.*, "A University Student Performance Prediction Model and Experiment Based on Multi-Feature Fusion and Attention Mechanism," *IEEE Access*, vol. 11, pp. 112307–112319, 2023, doi: 10.1109/ACCESS.2023.3323365.

[85] N. M. Arunkumar, “AUTOMATED STUDENT PERFORMANCE ANALYSER AND RECOMMENDER,” *International Journal of Advanced Research in Computer Science*, vol. 9, no. 2, pp. 688–698, Feb. 2018, doi: 10.26483/ijarcs.v9i2.5898.

[86] G. Ramaswami, T. Susnjak, A. Mathrani, J. Lim, and P. Garcia, “Using educational data mining techniques to increase the prediction accuracy of student academic performance,” *Information and Learning Sciences*, vol. 120, no. 7/8, pp. 451–467, Jul. 2019, doi: 10.1108/ILS-03-2019-0017.

[87] P. Sokkhey and T. Okazaki, “Developing Web-based Support Systems for Predicting Poor-performing Students using Educational Data Mining Techniques,” *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 7, 2020, doi: 10.14569/IJACSA.2020.0110704.

[88] Surendranadha Reddy Byrapu Reddy, Sarath Babu Dodda, Srihari Maruthi, and Mohan Raparthi, “AI-Driven Automated Feature Engineering to Enhance Performance of Predictive Models in Data Science,” *International Journal of Control and Automation*, 2020, doi: 10.52783/ijca.v13i4.38349.

[89] E. Katya, “Exploring Feature Engineering Strategies for Improving Predictive Models in Data Science,” *Research Journal of Computer Systems and Engineering*, vol. 4, no. 2, pp. 201–215, Dec. 2023, doi: 10.52710/rjcse.88.

[90] V. Bakyalakshmi, “A Multi-View Deep Learning Approach for Enhanced Student Academic Performance Prediction,” *Communications on Applied Nonlinear Analysis*, vol. 31, no. 6s, pp. 293–304, Aug. 2024, doi: 10.52783/cana.v31.1223.

[91] P. S. H. S. Saran and R. Krishna Kumari, “Unveiling Academic Success,” 2025, pp. 333–358. doi: 10.4018/979-8-3373-1399-3.ch014.

[92] M. S. Jony, “Exploring the Key Factors Influencing the Academic Performance of First Year Students at Undergraduate Level in Bangladesh,” *Journal of Educational Issues*, vol. 7, no. 2, p. 288, Nov. 2021, doi: 10.5296/jei.v7i2.19068.

[93] L. Newman-Ford, K. Fitzgibbon, S. Lloyd, and S. Thomas, “A large-scale investigation into the relationship between attendance and attainment: a study using an innovative, electronic attendance monitoring system,” *Studies in Higher Education*, vol. 33, no. 6, pp. 699–717, Dec. 2008, doi: 10.1080/03075070802457066.

[94] A. Maydosz and S. A. Raver, “Note taking and university students with learning

difficulties: What supports are needed?," *J. Divers. High. Educ.*, vol. 3, no. 3, pp. 177–186, Sep. 2010, doi: 10.1037/a0020297.

[95] M. Grabe, K. Christopherson, and J. Douglas, "Providing Introductory Psychology Students Access to Online Lecture Notes: The Relationship of Note Use to Performance and Class Attendance," *Journal of Educational Technology Systems*, vol. 33, no. 3, pp. 295–308, Mar. 2005, doi: 10.2190/G5RF-DMWG-WV1G-TMGG.

[96] D. N. Dwivedi, G. Mahanty, and V. nath Dwivedi, "The Role of Predictive Analytics in Personalizing Education," 2024, pp. 44–59. doi: 10.4018/979-8-3693-2169-0.ch003.

[97] C. J. Hetigan, F. Ma. C. Abrenica, L. Luisito Lolong, and P. Raymond Lebrias, "Leveraging Predictive Analytics on LMS Logs to Examine the Impact of Engagement on Academic Performance among College Students Enrolled in Centro Escolar University," *International Journal of Research and Scientific Innovation*, vol. XII, no. I, pp. 563–573, 2025, doi: 10.51244/IJRSI.2025.12010051.

[98] Lucky H S, "Design of an Improved Method for Personalized Learning Using Deep Q-Learning and CNN-Based Engagement Recognition," *Advances in Nonlinear Variational Inequalities*, vol. 28, no. 3s, pp. 483–500, Dec. 2024, doi: 10.52783/anvi.v28.3118.

[99] J. Ranjan and K. Malik, "Effective educational process: a data-mining approach," *VINE*, vol. 37, no. 4, pp. 502–515, Oct. 2007, doi: 10.1108/03055720710838551.

[100] F. Zhang, M. Petersen, L. Johnson, J. Hall, and S. E. O'Bryant, "Hyperparameter Tuning with High Performance Computing Machine Learning for Imbalanced Alzheimer's Disease Data," *Applied Sciences*, vol. 12, no. 13, p. 6670, Jul. 2022, doi: 10.3390/app12136670.

[101] F. Rzepka, P. Hematty, M. Schmitz, and J. Kowal, "Neural Network Architecture for Determining the Aging of Stationary Storage Systems in Smart Grids," *Energies (Basel).*, vol. 16, no. 17, p. 6103, Aug. 2023, doi: 10.3390/en16176103.

[102] M. Uppal *et al.*, "Enhancing accuracy in brain stroke detection: Multi-layer perceptron with Adadelta, RMSProp and AdaMax optimizers," *Front. Bioeng. Biotechnol.*, vol. 11, Sep. 2023, doi: 10.3389/fbioe.2023.1257591.

[103] R. Saxena, S. K. Sharma, M. Gupta, and G. C. Sampada, "A Novel Approach for Feature Selection and Classification of Diabetes Mellitus: Machine Learning

Methods," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–11, Apr. 2022, doi: 10.1155/2022/3820360.

[104] S. Kılıçarslan, H. A. Aydın, K. Adem, and E. K. Yılmaz, "Impact of optimizers functions on detection of Melanoma using transfer learning architectures," *Multimed. Tools Appl.*, vol. 84, no. 14, pp. 13787–13807, Jun. 2024, doi: 10.1007/s11042-024-19561-6.

[105] J. Ko, S. Park, and H. G. Woo, "Optimization of vision transformer-based detection of lung diseases from chest X-ray images," *BMC Med. Inform. Decis. Mak.*, vol. 24, no. 1, p. 191, Jul. 2024, doi: 10.1186/s12911-024-02591-3.

[106] R. Abdulkadirov, P. Lyakhov, and N. Nagornov, "Survey of Optimization Algorithms in Modern Neural Networks," *Mathematics*, vol. 11, no. 11, p. 2466, May 2023, doi: 10.3390/math11112466.

[107] A. Dharmawan, R. E. Masithoh, and H. Z. Amanah, "Development of PCA-MLP Model Based on Visible and Shortwave Near Infrared Spectroscopy for Authenticating Arabica Coffee Origins," *Foods*, vol. 12, no. 11, p. 2112, May 2023, doi: 10.3390/foods12112112.

[108] Y. Song, X. Meng, and J. Jiang, "Multi-Layer Perception model with Elastic Grey Wolf Optimization to predict student achievement," *PLoS One*, vol. 17, no. 12, p. e0276943, Dec. 2022, doi: 10.1371/journal.pone.0276943.

[109] A. D. Novika and A. S. Girsang, "Multi-layer perceptron hyperparameter optimization using Jaya algorithm for disease classification," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 35, no. 1, p. 620, Jul. 2024, doi: 10.11591/ijeecs.v35.i1.pp620-630.

[110] D. Choi, C. Shallue, Z. Nado, J. Lee, C. Maddison, and G. Dahl, "On empirical comparisons of optimizers for deep learning".

[111] C. Verma, V. Stoffova, Z. Illes, S. Tanwar, and N. Kumar, "Machine Learning-Based Student's Native Place Identification for Real-Time," *IEEE Access*, vol. 8, pp. 130840–130854, 2020, doi: 10.1109/ACCESS.2020.3008830.

[112] Y. Xie, "Efficiency of Hybrid Decision Tree Algorithms in Evaluating the Academic Performance of Students," *International Journal of Advanced Computer Science and Applications*, vol. 15, no. 2, 2024, doi: 10.14569/IJACSA.2024.0150251.

[113] M. R. Apriyadi, E. -, and D. P. Rini, "Hyperparameter Optimization of Support Vector Regression Algorithm using Metaheuristic Algorithm for Student

Performance Prediction," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 2, 2023, doi: 10.14569/IJACSA.2023.0140218.

[114] G. Airlangga, "Predicting Student Performance Using Deep Learning Models: A Comparative Study of MLP, CNN, BiLSTM, and LSTM with Attention," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 4, no. 4, pp. 1561–1567, Oct. 2024, doi: 10.57152/malcom.v4i4.1668.

[115] S. M. Patil and P. Kumar, "Data Mining Model for Effective Data Analysis of Higher Education Students Using MapReduce," *International Journal of Emerging Research in Management and Technology*, vol. 6, no. 4, pp. 177–183, Apr. 2017, doi: 10.23956/ijermt/SV6N4/118.

[116] Sheena A and Sachin A, "Analysis of Student's Data using Rapid Miner," *Journal on Today's Ideas - Tomorrow's Technologies*, vol. 4, no. 2, pp. 109–117, Dec. 2016, doi: 10.15415/jotitt.2016.42007.

[117] E. D. , K. Alisultanova, & S. L. K., and Z. A., "DATA MINING TECHNIQUES IN EDUCATION," *Bulletin of GSTOU: Humanities and Socio-Economic Sciences*, no. 2(28), pp. 47–54, Aug. 2022, doi: 10.34708/GSTOU.2022.16.83.006.

[118] A. Destria, A. Nurlita, and Terttiaavini, "Analisis Prediksi Pemilihan Mata Kuliah Peminatan pada Jurusan Teknik Informatika Universitas Indo Global Mandiri Menggunakan Metode Linier Regresi," *Journal Innovations Computer Science*, vol. 2, no. 1, pp. 1–6, May 2023, doi: 10.56347/jics.v2i1.119.

[119] K. Mangaroska and M. Giannakos, "Learning Analytics for Learning Design: Towards Evidence-Driven Decisions to Enhance Learning," 2017, pp. 428–433. doi: 10.1007/978-3-319-66610-5_38.

[120] C. Romero and S. Ventura, "Data mining in education," *WIREs Data Mining and Knowledge Discovery*, vol. 3, no. 1, pp. 12–27, Jan. 2013, doi: 10.1002/widm.1075.

[121] G. Feng, M. Fan, and C. Ao, "Exploration and Visualization of Learning Behavior Patterns From the Perspective of Educational Process Mining," *IEEE Access*, vol. 10, pp. 65271–65283, 2022, doi: 10.1109/ACCESS.2022.3184111.

[122] L. O. Moraes, C. E. Pedreira, C. Delgado, and J. P. Freire, "Supporting Decisions Using Educational Data Analysis," in *Anais Estendidos do XXVII Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia 2021)*, Sociedade Brasileira de Computação - SBC, Nov. 2021, pp. 99–102. doi:

10.5753/webmedia_estendido.2021.17622.

[123] A. Namoun and A. Alshanqiti, “Predicting Student Performance Using Data Mining and Learning Analytics Techniques: A Systematic Literature Review,” *Applied Sciences*, vol. 11, no. 1, p. 237, Dec. 2020, doi: 10.3390/app11010237.

[124] C.-H. Yu, J. Wu, and A.-C. Liu, “Predicting Learning Outcomes with MOOC Clickstreams,” *Educ. Sci. (Basel).*, vol. 9, no. 2, p. 104, May 2019, doi: 10.3390/educsci9020104.

[125] H. Hopia, E. Latvala, and L. Liimatainen, “Reviewing the methodology of an integrative review,” *Scand. J. Caring Sci.*, vol. 30, no. 4, pp. 662–669, Dec. 2016, doi: 10.1111/scs.12327.

[126] C. J. Arizmendi *et al.*, “Predicting student outcomes using digital logs of learning behaviors: Review, current standards, and suggestions for future work,” *Behav. Res. Methods*, vol. 55, no. 6, pp. 3026–3054, Aug. 2022, doi: 10.3758/s13428-022-01939-9.

[127] S. Slater, S. Joksimović, V. Kovanovic, R. S. Baker, and D. Gasevic, “Tools for Educational Data Mining,” *Journal of Educational and Behavioral Statistics*, vol. 42, no. 1, pp. 85–106, Feb. 2017, doi: 10.3102/1076998616666808.

[128] M. A. Tariq, A. B. Sargano, M. A. Iftikhar, and Z. Habib, “Comparing Different Oversampling Methods in Predicting Multi-Class Educational Datasets Using Machine Learning Techniques,” *Cybernetics and Information Technologies*, vol. 23, no. 4, pp. 199–212, Nov. 2023, doi: 10.2478/cait-2023-0044.

[129] H. Hartono, O. S. Sitompul, T. Tulus, E. B. Nababan, and D. Napitupulu, “Hybrid Approach Redefinition (HAR) model for optimizing hybrid ensembles in handling class imbalance: a review and research framework,” *MATEC Web of Conferences*, vol. 197, p. 03003, Sep. 2018, doi: 10.1051/matecconf/201819703003.

[130] R. Cruz, K. Fernandes, J. S. Cardoso, and J. F. Pinto Costa, “Tackling class imbalance with ranking,” in *2016 International Joint Conference on Neural Networks (IJCNN)*, IEEE, Jul. 2016, pp. 2182–2187. doi: 10.1109/IJCNN.2016.7727469.

[131] Z. Qing, Q. Zeng, H. Wang, Y. Liu, T. Xiong, and S. Zhang, “ADASYN-LOF Algorithm for Imbalanced Tornado Samples,” *Atmosphere (Basel).*, vol. 13, no. 4, p. 544, Mar. 2022, doi: 10.3390/atmos13040544.

[132] M. K. Zuhanda, Lisya Permata, Hartono, Erianto Ongko, and Desniarti, “Impact

of Adaptive Synthetic on Naïve Bayes Accuracy in Imbalanced Anemia Detection Datasets," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 9, no. 1, pp. 85–93, Jan. 2025, doi: 10.29207/resti.v9i1.6031.

[133] Â. Frimane and J. M. Bright, "Validation of Synthetic Solar Irradiance Data," in *Synthetic Solar Irradiance*, AIP Publishing LLCMelville, New York, 2021, pp. 4-1-4–44. doi: 10.1063/9780735421820_004.

[134] G. Soltana, M. Sabetzadeh, and L. C. Briand, "Synthetic data generation for statistical testing," in *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, IEEE, Oct. 2017, pp. 872–882. doi: 10.1109/ASE.2017.8115698.

[135] A. Al–Qerem *et al.*, "Synthetic Generation of Multidimensional Data to Improve Classification Model Validity," *Journal of Data and Information Quality*, vol. 15, no. 3, pp. 1–20, Sep. 2023, doi: 10.1145/3603715.

[136] N. Zhou, L. Wang, S. Marino, Y. Zhao, and I. D. Dinov, "DataSifter II: Partially synthetic data sharing of sensitive information containing time-varying correlated observations," *J. Algorithm. Comput. Technol.*, vol. 16, Jan. 2022, doi: 10.1177/17483026211065379.

[137] B. Brandt and P. Dasgupta, "Synthetically generating human-like data for sequential decision-making tasks via reward-shaped imitation learning," in *Synthetic Data for Artificial Intelligence and Machine Learning: Tools, Techniques, and Applications*, K. E. Manser, R. M. Rao, and C. L. Howell, Eds., SPIE, Jun. 2023, p. 18. doi: 10.1117/12.2664109.

[138] M. A. A. Aziz, N. Ismail, I. M. Yassin, A. Zabidi, and M. S. A. M. Ali, "Agarwood oil quality classification using cascade-forward neural network," in *2015 IEEE 6th Control and System Graduate Research Colloquium (ICSGRC)*, 2015, pp. 112–115. doi: 10.1109/ICSGRC.2015.7412475.

[139] I. M. Hamed, A. S. Hussein, and M. F. Tolba, "An Intelligent Model for Stock Market Prediction," *International Journal of Computational Intelligence Systems*, vol. 5, no. 4, p. 639, 2012, doi: 10.1080/18756891.2012.718108.

[140] R. Jafari-Marandi, M. Khanzadeh, B. K. Smith, and L. Bian, "Self-Organizing and Error Driven (SOED) artificial neural network for smarter classifications," *J. Comput. Des. Eng.*, vol. 4, no. 4, pp. 282–304, Oct. 2017, doi: 10.1016/j.jcde.2017.04.003.

[141] S. Gupta and M. K. Gupta, "Computational Model for Prediction of Malignant

Mesothelioma Diagnosis," *Comput. J.*, vol. 66, no. 1, pp. 86–100, Jan. 2023, doi: 10.1093/comjnl/bxab146.

[142] M. N. G. Lopes, B. R. P. da Rocha, A. C. Vieira, J. A. S. de Sá, P. A. M. Rolim, and A. G. da Silva, "Artificial neural networks approaches for predicting the potential for hydropower generation: a case study for Amazon region," *Journal of Intelligent & Fuzzy Systems*, vol. 36, no. 6, pp. 5757–5772, Jun. 2019, doi: 10.3233/JIFS-181604.

[143] A. S. Mohammed, A. S. Almawla, and S. S. Thameel, "Prediction of Monthly Evaporation Model Using Artificial Intelligent Techniques in the Western Desert of Iraq-Al-Ghadaf Valley," *Mathematical Modelling of Engineering Problems*, vol. 9, no. 5, pp. 1261–1270, Dec. 2022, doi: 10.18280/mmep.090513.

[144] Hamdard M S and H. Lodin, "Effect of Feature Selection on the Accuracy of Machine Learning Model," *INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH AND ANALYSIS*, vol. 06, no. 09, Sep. 2023, doi: 10.47191/ijmra/v6-i9-66.

[145] S. Fenanir, F. Semchedine, and A. Baadache, "A Machine Learning-Based Lightweight Intrusion Detection System for the Internet of Things," *Revue d'Intelligence Artificielle*, vol. 33, no. 3, pp. 203–211, Oct. 2019, doi: 10.18280/ria.330306.

[146] K. Li and N. Fard, "A Novel Nonparametric Feature Selection Approach Based on Mutual Information Transfer Network," *Entropy*, vol. 24, no. 9, p. 1255, Sep. 2022, doi: 10.3390/e24091255.

[147] S. Shukla *et al.*, "A smartphone-based standalone fluorescence spectroscopy tool for cervical precancer diagnosis in clinical conditions," *J. Biophotonics*, vol. 17, no. 6, Jun. 2024, doi: 10.1002/jbio.202300468.

[148] A. Koc, F. Odilbekov, M. Alamrani, T. Henriksson, and A. Chawade, "Predicting yellow rust in wheat breeding trials by proximal phenotyping and machine learning," *Plant Methods*, vol. 18, no. 1, p. 30, Dec. 2022, doi: 10.1186/s13007-022-00868-0.

[149] Z. Luo, H. Wang, and S. Li, "Prediction of International Roughness Index Based on Stacking Fusion Model," *Sustainability*, vol. 14, no. 12, p. 6949, Jun. 2022, doi: 10.3390/su14126949.

[150] C. Wegener and C. C. Ibebuchi, "Application of <scp>XGBoost</scp> in Disentangling the Fingerprints of Global Warming and Decadal Climate Modes

on Seasonal Precipitation Trends in Ohio," *International Journal of Climatology*, vol. 45, no. 8, Jun. 2025, doi: 10.1002/joc.8829.

[151] S. M. Lundberg *et al.*, "From local explanations to global understanding with explainable AI for trees," *Nat. Mach. Intell.*, vol. 2, no. 1, pp. 56–67, Jan. 2020, doi: 10.1038/s42256-019-0138-9.

[152] Z. Dai, T. Li, and M. Yang, "Forecasting stock return volatility: The role of shrinkage approaches in a data-rich environment," *J. Forecast.*, vol. 41, no. 5, pp. 980–996, Aug. 2022, doi: 10.1002/for.2841.

[153] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization Paths for Generalized Linear Models via Coordinate Descent," *J. Stat. Softw.*, vol. 33, no. 1, 2010, doi: 10.18637/jss.v033.i01.

[154] M. Xu, "Sales Prediction Based on Lasso Regression," *Highlights in Science, Engineering and Technology*, vol. 88, pp. 343–349, Mar. 2024, doi: 10.54097/p9hyrk70.

[155] A. Çelik and S. Demirel, "Enhanced Pneumonia Diagnosis Using Chest X-Ray Image Features and Multilayer Perceptron and k-NN Machine Learning Algorithms," *Traitement du Signal*, vol. 40, no. 3, pp. 1015–1023, Jun. 2023, doi: 10.18280/ts.400317.

[156] C.-C. Wu, C.-W. Tsai, F.-E. Wu, C.-H. Chiang, and J.-C. Chiou, "Plantar Pressure-Based Gait Recognition with and Without Carried Object by CNN-Autoencoder Architecture," *Biomimetics*, vol. 10, no. 2, p. 79, Jan. 2025, doi: 10.3390/biomimetics10020079.

[157] M. Shantal, Z. Othman, and A. A. Bakar, "A Novel Approach for Data Feature Weighting Using Correlation Coefficients and Min–Max Normalization," *Symmetry (Basel).*, vol. 15, no. 12, p. 2185, Dec. 2023, doi: 10.3390/sym15122185.

[158] D. Protić *et al.*, "Numerical Feature Selection and Hyperbolic Tangent Feature Scaling in Machine Learning-Based Detection of Anomalies in the Computer Network Behavior," *Electronics (Basel).*, vol. 12, no. 19, p. 4158, Oct. 2023, doi: 10.3390/electronics12194158.

[159] F. Gurcan and A. Soylu, "Learning from Imbalanced Data: Integration of Advanced Resampling Techniques and Machine Learning Models for Enhanced Cancer Diagnosis and Prognosis," *Cancers (Basel).*, vol. 16, no. 19, p. 3417, Oct. 2024, doi: 10.3390/cancers16193417.

[160] R. S. Abdulsadig and E. Rodriguez-Villegas, “A comparative study in class imbalance mitigation when working with physiological signals,” *Front. Digit. Health*, vol. 6, Mar. 2024, doi: 10.3389/fdgth.2024.1377165.

[161] Haibo He, Yang Bai, E. A. Garcia, and Shutao Li, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, IEEE, Jun. 2008, pp. 1322–1328. doi: 10.1109/IJCNN.2008.4633969.

[162] D.-T. Nguyen, T.-K. Nguyen, Z. Ahmad, and J.-M. Kim, “A Reliable Pipeline Leak Detection Method Using Acoustic Emission with Time Difference of Arrival and Kolmogorov–Smirnov Test,” *Sensors*, vol. 23, no. 23, p. 9296, Nov. 2023, doi: 10.3390/s23239296.

[163] C. Coisnon, M. A. Mitchell, B. Rannou, and K. Le Boedec, “Subjective assessment of frequency distribution histograms and consequences on reference interval accuracy for small sample sizes: A computer-simulated study,” *Vet. Clin. Pathol.*, vol. 50, no. 3, pp. 427–441, Sep. 2021, doi: 10.1111/vcp.13000.

[164] R. Silva de Souza and E. M. Borges, “Teaching Descriptive Statistics and Hypothesis Tests Measuring Water Density,” *J. Chem. Educ.*, vol. 100, no. 11, pp. 4438–4448, Nov. 2023, doi: 10.1021/acs.jchemed.3c00402.

[165] A. Mazarei, R. Sousa, J. Mendes-Moreira, S. Molchanov, and H. M. Ferreira, “Online boxplot derived outlier detection,” *Int. J. Data Sci. Anal.*, vol. 19, no. 1, pp. 83–97, Jan. 2025, doi: 10.1007/s41060-024-00559-0.

[166] H. S. Xenias *et al.*, “R1441C and G2019S LRRK2 knockin mice have distinct striatal molecular, physiological, and behavioral alterations,” *Commun. Biol.*, vol. 5, no. 1, p. 1211, Nov. 2022, doi: 10.1038/s42003-022-04136-8.

[167] E. O. Lutsenko, O. V. Marinich, and I. K. Matsak, “Some applications of the Gnedenko–Korolyuk method to empirical distributions,” *Theory of Probability and Mathematical Statistics*, vol. 78, pp. 133–146, 2009, doi: 10.1090/S0094-9000-09-00767-4.

[168] N. Kim, “Tests based on EDF statistics for randomly censored normal distributions when parameters are unknown,” *Commun. Stat. Appl. Methods*, vol. 26, no. 5, pp. 431–443, Sep. 2019, doi: 10.29220/CSAM.2019.26.5.431.

[169] K. Potter, J. Kniss, R. Riesenfeld, and C. R. Johnson, “Visualizing Summary Statistics and Uncertainty,” *Computer Graphics Forum*, vol. 29, no. 3, pp. 823–

832, Jun. 2010, doi: 10.1111/j.1467-8659.2009.01677.x.

[170] A. Elhadad, M. Jamjoom, and H. Abulkasim, “Reduction of NIFTI files storage and compression to facilitate telemedicine services based on quantization hiding of downsampling approach,” *Sci. Rep.*, vol. 14, no. 1, p. 5168, Mar. 2024, doi: 10.1038/s41598-024-54820-4.

[171] J. Shi *et al.*, “Detecting the skewness of data from the five-number summary and its application in meta-analysis,” *Stat. Methods Med. Res.*, vol. 32, no. 7, pp. 1338–1360, Jul. 2023, doi: 10.1177/09622802231172043.

[172] X. Bouthillier *et al.*, “Accounting for Variance in Machine Learning Benchmarks,” 2021, doi: 10.48550/arxiv.2103.03098.

[173] I. Loshchilov and F. Hutter, “Decoupled Weight Decay Regularization,” 2017, doi: 10.48550/arxiv.1711.05101.

AUTHOR'S PROFILE

Fairul Nazmie Osman is an alumnus of Sekolah Menengah Sains Sultan Mahmud (SESMA), Kuala Terengganu. He obtained his Diploma in Electrical and Electronic Engineering from Universiti Teknologi MARA (UiTM), Shah Alam in 2002, followed by a Bachelor of Electronics Engineering (Hons) from the University of Surrey, United Kingdom in 2005. His undergraduate final-year project investigated the effects of annealing process on pre-amorphized, and crystalline silicon doped with antimony (Sb). He completed his Master of Science in Electronic Systems Design Engineering at Universiti Sains Malaysia (USM) in 2008, with a dissertation focusing on Multiple-Input Multiple-Output (MIMO) system design. His doctoral research centres on predictive modelling of student academic performance using multi-source engagement data collected during the early weeks of a semester, with the objective of enabling early identification of at-risk students and supporting timely instructional intervention.

LIST OF PUBLICATION:

Enhancing Students' Academic Performance Classifier using ADASYN and MLP, FN Osman, MAA Aziz, MN Taib, 2024 IEEE 22nd Student Conference on Research and Development (SCOReD), 221-226, https://doi.org/10.1109/SCOReD64708.2024.10872712

Comparative Evaluation of Feature Selection Algorithms for Predictive Modeling of Academic Performance Outcomes, FN Osman, MAA Aziz, MN Taib, 2024 IEEE 13th International Conference on Engineering Education (ICEED), 1-6, https://doi.org/10.1109/ICEED62316.2024.10923823

Impact of Optimizer on the MLP-Based Models for Student Performance Classification, FN Osman, MAA Aziz, IM Yassin, MN Taib, Journal of Electrical & Electronic Systems Research (JEESR),Oct 2025 https://doi.org/10.24191/jeesr.v27i1.012