Undergraduate Academic Performance Prediction While Maintaining Both Accuracy And Interpretability

P.M. Pathiranagama

Undergraduate Academic Performance Prediction While Maintaining Both Accuracy And Interpretability

P.M. Pathiranagama

Index No: 19001118

Supervisor : Dr. H.K.T.C. Halloluwa

Co-supervisor : Mr. R.M.U.A. Rathnayake

May 2024

Submitted in partial fulfillment of the requirements of the

B.Sc in Computer Science Final Year Project (SCS4224)

Declaration

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

Candidate Name: P.M. Pathiranagama

27/09/2024

Signature of the candidate

Date

This is to certify that this dissertation is based on the work of Mr. P.M. Pathiranagama under my supervision. The thesis has been prepared according to the format stipulated and is of an acceptable standard.

Principle Supervisor's Name: Dr. H.K.T.C. Halloluwa

27/09/2024

Signature of the Supervisor

Date

This is to certify that this dissertation is based on the work of Mr. P.M. Pathiranagama under my supervision. The thesis has been prepared according to the format stipulated and is of an acceptable standard.

Co-Supervisor's Name: Mr. R.M.U.A. Rathnayake

27/09/2024

Signature of the Co-Supervisor

Date

Abstract

Student dropout in higher education is one of the significant problems encountered by educational institutions and students globally. This theses focuses on identifying contributing factors and improving prediction accuracy using various machine learning techniques. The research uses Logistic Regression, Decision Trees, Random Forest Trees, Support Vector Machines (SVM), Naive Bayes, and Boosting Classifiers like XG Boost Classifier, Gradient Boost Classifier, CatBoost Classifier, and AdaBoost Classifier to examine both academic and non-academic factors. To enhance the analysis, the research incorporates techniques like correlated feature management and hyperparameter tuning, alongside data sampling methodologies such as SMOTE, SVM-SMOTE, and ADASYN with ADASYN emerging as the best sampling technique.

After the initial stages the research found that the CatBoost classifier, enhanced by ADASYN sampling, significantly improved prediction accuracy with a testing F1-Score of 0.8603, suggesting a robust model for educational institutions to early identify at-risk students. Then, the second phase of this research was focused on the interpretation of the prediction. There we considered LIME,SHAP and Explainable Boosting Machine as the interpretable and explainable models. This thesis identifies non-academic factors such as socio-economic background and personal resilience as significant predictors of student dropout rates, beyond academic performance alone from the explanations of the two XAI models.

This study conducted comprehensive experiments encompassing machine learning and explainable artificial intelligence methodologies, aiming to optimize accuracy and interoperability in the obtained results and found out how LIME and SHAP can be applied for interpretability according to the context.

Preface

This document has been produced for the partial fulfillment of the requirements of the B.Sc. in Computer Science (Hons) Final Year Project in Computer Science (SCS4224).

This study examines the intersection of Machine Learning (ML) and Explainable Artificial Intelligence (XAI), with a specific emphasis on the prediction of undergraduate dropouts and their academic performance. It aims to identify students who are at risk of dropping out or underperforming early on, using effective machine learning techniques and XAI techniques.

A Portugal university undergraduate student dataset was used in this study which had an inherent class imbalance problem. We used SMOTE, SVM-SMOTE, and ADASYN sampling techniques to overcome from the class imbalance problem. The predictive power of several machine learning models is evaluated in this study including 5 general models and 4 boosting models. In addition, the research employs XAI approaches such as LIME, SHAP and EBM to demystify the models' decision-making processes, improving the transparency and interpretability of the results.

In Chapter 1, it presents the introduction and background of the study with an overview of the dissertation. In Chapter 2, it explore a wider range of theories, techniques, and related work in this field. In Chapter 3, it explains how we designed our approach, and in Chapter 4, it will provide all the details about how we implemented it. All the results and how each and every evaluation is done is described in Chapter 5. Then, in Chapter 6, we will discuss the conclusions, limitaions and future work of the research.

This dissertation represents the original work that I, along with my supervisor and co-supervisor, have conducted and hereby claim everything else mentioned in this dissertation without a specific reference to any third-party work as our own.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my supervisor Dr. H.K.T.C. Halloluwa for the tremendous support and guidance given to me in continuing this research throughout this year and also my co-supervisor Mr. R.M.U.A. for the constant support and assistance given to me in carrying out this research.

Further I would like to thank to all my friends who helped me and supported me in numerous ways to carry out this research successfully. Finally many thanks to my loving parents and dear brother for always being my pillars of strength, guiding me in the right direction, and shaping me into the person I am today.

This thesis is dedicated to all of my family members, colleagues, school teachers, lecturers and staff members of the University of Colombo School of Computing, and to everyone who has helped me even if it was just by a single word.

Acronyms

ADASYN Adaptive Synthetic Sampling

DL Deep Learning

EBM Explainable Boosting Machine

ML Machine Learning

SAP Student Academic Performance

SMOTE Synthetic Minority Oversampling Technique

SVM Support Vector Machine

XAI Explainable Artificial Intelligence

MOOC Massive Open Online Courses

Contents

Declaration **i**

Abstract **ii**

Preface **iii**

ACKNOWLEDGMENTS **iv**

Acronyms **v**

List of Figures **xv**

1 Introduction **1**

1.1 Introduction 1

1.2 Motivation 3

1.3 Problem Statement 4

1.4 Research Aim Questions and Objectives 5

1.4.1 Research Aim 5

1.4.2 Research Questions 5

1.4.3 Research Objectives 6

1.5 Significance of The Project 7

1.6 Scope Including Delimitations 8

1.6.1 In Scope 8

1.6.2 Out of Scope 8

1.7 Chapter Summary 9

2 Literature Review **10**

2.1 General Machine Learning Techniques to Predict Student Dropouts 10

2.1.1 Class Imbalance Problem 12

2.1.2 Physical Learning Environment 12

2.1.3 E-Learning Environments 14
2.1.4 Video Interaction Prediction 14
2.2 Implication of XAI for the interpretation model results 15
2.2.1 What is XAI and why it is important 16
2.2.2 How XAI can be used to address the problem 16
2.3 Evaluating XAI models 18
2.4 Research Gap 18
2.5 Chapter Summary 20

3 Design and Approach 21

3.1 Research Design and Approach 21
3.1.1 Data Collection 21
3.1.2 Data Exploration And Analysis 21
3.1.3 Data Preprocessing 26
3.1.4 Modifying Categories of The Target Feature 27
3.1.5 Handling Correlations 29
3.1.6 Handling Class Imbalance Problem 31
3.2 Chapter Summary 32

4 Implementation of the models 33

4.1 Architecture Design 33
4.2 Implenting Explainable Boosting Machine (EBM) 35
4.3 Experiments 37
4.3.1 Classification Models 37
4.3.2 Explainable and Interpretable Models 38
4.3.3 Model Training 39
4.3.4 Implementation of LIME 41
4.3.5 Implementation of SHAP 42
4.3.6 What are SHAPLEY Values 42

4.3.7 Insights from Implementing LIME and SHAP 43
4.4 Chapter Summary 44

5 Results And Evaluation 45

5.1 Results of Data Sampling 45
5.1.1 SMOTE 45
5.1.2 SVM-SMOTE 48
5.1.3 ADASYN 49
5.2 Hyper Parameter Tuning 50
5.3 Results and Evaluation of Machine Learining Models 51
5.4 XAI and Interpretation Results 56
5.5 Interpret The Decision Tree 58
5.6 Interpretation of the Explainable Boosting Machine (EBM) 59
5.7 Evaluation of XAI Models 60
5.7.1 Results of The Evaluation of LIME - User Study 1 61
5.7.2 Results of The Evaluation of SHAP - User Study 1 65
5.7.3 Results of The General Evaluation - User Study 1 68
5.7.4 Results of The Evaluation of LIME - User Study 2 75
5.7.5 Results of The Evaluation of SHAP - User Study 2 79
5.7.6 Results of The General Evaluation - User Study 2 83
5.7.7 Overall Summary of The XAI Evaluation 87
5.8 Chapter Summary 89

6 Conclusions 90

6.1 Introduction 90
6.2 Conclusions about Research Questions 90
6.3 Conclusions about Research Problem 92
6.4 Limitations 93
6.5 Implications for further research 93

7 Appendix A: Results of Experiments **99**

7.1 Results of The Machine Learning Models 99

7.2 Results of The XAI and Interpretable Models 109

8 Appendix B: Code Listing **111**

List of Figures

2.2.1 XAI vs Machine Learning 16
3.1.1 No of students vs nationality 22
3.1.2 No of students vs international 22
3.1.3 Age at enrollment 23
3.1.4 Distribution of Debtors 23
3.1.5 Educational special needs 23
3.1.6 Martial status 24
3.1.7 Father's occupation 24
3.1.8 Mother's occupation 25
3.1.9 Previous education qualifications 25
3.1.10 Scholarship holder 25
3.1.11 Tuition fees up to date 26
3.1.12 Target variable class distribution 28
3.1.13 Target variable class distribution 29
3.1.14 Correlation Heatmap 31
4.2.1 Architecure Graph 36
4.3.1 Training, Testing and Validation Split 39
4.3.2 SHAP Waterfall plot 43
5.1.1 Logistic Regression - SMOTE ROC 46
5.4.1 LIME local explanation 56
5.4.2 SHAP local explanation 57
5.4.3 SHAP global explanation 57
5.4.4 SHAP global bar chart explanation 58
5.5.1 Pruned Decision Tree Interpretation 59
5.6.1 EBM Interpretation 60
5.7.1 Participated Evaluators 61

5.7.2 Awareness on LIME 62
5.7.3 Understandability of LIME 62
5.7.4 Satisfaction of LIME 63
5.7.5 Sufficiency of LIME 63
5.7.6 Trustworthiness of LIME 64
5.7.7 Awareness on SHAP 65
5.7.8 Understandability of SHAP 66
5.7.9 Satisfaction of SHAP 66
5.7.10 Sufficiency of SHAP 67
5.7.11 Trustworthiness of SHAP 67
5.7.12 Preference between the two models 69
5.7.13 Reasons For The Preferences between the two models 69
5.7.14 Preferred model 70
5.7.15 Experience in identifying or consulting at-risk students 71
5.7.16 Difficulties in identifying at-risk students 72
5.7.17 General problems in identifying at-risk students 73
5.7.18 Usage of data observation-based techniques in identifying at-risk students 73
5.7.19 Types of data observation-based techniques used in identifying at-risk students 74
5.7.20 Overall understandability of the models 74
5.7.21 Feedbacks 75
5.7.22 Awareness of LIME 75
5.7.23 How did you became aware of LIME 76
5.7.24 Understandability of LIME 76
5.7.25 Explanation on Understandability of LIME 77
5.7.26 Satisfaction of LIME 77
5.7.27 Sufficiency of LIME 78

5.7.28 Trustworthiness of LIME 78
5.7.29 Awareness on SHAP 80
5.7.30 How did you become aware of SHAP 80
5.7.31 Understandability of SHAP 81
5.7.32 Satisfaction of SHAP 81
5.7.33 Sufficiency of SHAP 82
5.7.34 Trustworthiness of SHAP 82
5.7.35 Preference between the two models 84
5.7.36 Reasons For The Preferences between the two models 84
5.7.37 Preferred model 85
5.7.38 Impact on the decision making process 85
5.7.39 Overall understandability of XAI techniques 86
5.7.40 Experience in using a student dropout prediction system 86
5.7.41 Impact on the decision making process 87
5.7.42 How useful to identify the factors 87
7.1.1 SMOTE Logistic Regression Results 99
7.1.2 SMOTE Decision Tree Results 99
7.1.3 SMOTE Random Forest Results 99
7.1.4 SMOTE SVM Results 100
7.1.5 SMOTE Naive Bayes Results 100
7.1.6 SMOTE Gradient Boost Results 100
7.1.7 SMOTE CatBoost Results 100
7.1.8 SMOTE AdaBoost Results 100
7.1.9 SVM-SMOTE Logistic Regression Results 101
7.1.10 SVM-SMOTE Decision Tree Results 101
7.1.11 SVM-SMOTE Random Forest Results 101
7.1.12 SVM-SMOTE SVM Results 101
7.1.13 SVM-SMOTE Naive Bayes Results 102

7.1.14 SVM-SMOTE Gradient Boost Results 102

7.1.15 SMOTE CatBoost Results 102

7.1.16 SVM-SMOTE AdaBoost Results 102

7.1.17 SVM-SMOTE Logistic Regression Results 103

7.1.18 SVM-SMOTE Decision Tree Results 103

7.1.19 SVM-SMOTE Random Forest Results 103

7.1.20 SVM-SMOTE SVM Results 103

7.1.21 SVM-SMOTE Naive Bayes Results 104

7.1.22 SVM-SMOTE Gradient Boost Results 104

7.1.23 SVM-SMOTE CatBoost Results 104

7.1.24 SVM-SMOTE AdaBoost Results 104

7.1.25 ADASYN Logistic Regression Results 105

7.1.26 ADASYN Decision Tree Results 105

7.1.27 ADASYN Random Forest Results 105

7.1.28 ADASYN SVM Results 105

7.1.29 ADASYN Naive Bayes Results 105

7.1.30 ADASYN Gradient Boost Results 106

7.1.31 ADASYN CatBoost Results 106

7.1.32 ADASYN AdaBoost Results 106

7.1.33 HPT Logistic Regression Results 106

7.1.34 HPT Decision Tree Results 106

7.1.35 HPT Random Forest Results 107

7.1.36 HPT SVM Results 107

7.1.37 HPT Naive Bayes Results 107

7.1.38 HPT Gradient Boost Results 107

7.1.39 HPT CatBoost Results 107

7.1.40 HPT AdaBoost Results 108

7.2.1 Pruned Decision Tree Interpretation 109

7.2.2 EBM Interpretation . . . 109
7.2.3 Preferred model . . . 110
7.2.4 Preference between the two models . . . 110
8.0.1 Pre-processing-Part 1 . . . 111
8.0.2 Pre-processing-Part 2 . . . 112
8.0.3 Pre-processing-Part 3 . . . 113
8.0.4 Pre-processing-Part 4 . . . 113
8.0.5 Pre-processing-Part 5 . . . 114
8.0.6 SVM-SMOTE sampling . . . 115
8.0.7 ADASYN sampling . . . 115
8.0.8 Logistic Regression code . . . 116
8.0.9 Random Forest code . . . 117
8.0.10 Naive Bayes code . . . 118
8.0.11 SVM code . . . 119
8.0.12 Decision Trees code . . . 120
8.0.13 Gradient Boost code . . . 121
8.0.14 CatBoost code . . . 122
8.0.15 XG Boost code . . . 123
8.0.16 ADABoost code . . . 124
8.0.17 HPT Logistic Regression code . . . 125
8.0.18 HPT Decision Tree code . . . 125
8.0.19 HPT Random Forest code . . . 126
8.0.20 HPT Naive Bayes code . . . 127
8.0.21 HPT CatBoost code . . . 128
8.0.22 HPT Gradient Boost code . . . 129
8.0.23 HPT ADABoost code . . . 130

List of Tables

5.1.1 Using SMOTE - Standard Models 48

5.1.2 Using SMOTE - Boosting Machines 48

5.1.3 Using SVM-SMOTE - Standard Models 49

5.1.4 Using SVM-SMOTE - Boosting Machines 49

5.1.5 Using ADASYN - Standard Models 49

5.1.6 Using ADASYN - Boosting Machines 50

1 Introduction

1.1 Introduction

Student dropout in higher education is one of the significant problems encountered by educational institutions and students globally. Therefore, it is an important field to explore student dropouts in higher education institutes as it has a significant impact on both educational institutions and student success. Academic achievement is an important predictor of identifying students at risk of dropping out. Other than academic achievements there are numerous other factors that may affect and contribute to the dropouts of students. According to a statistical study that was done in 2022 (Hanson 2022), it has found, 32.9% of United States undergraduates do not complete their degree program. These statistics implicate why it is important to explore student dropouts in higher education.

This research focuses on predicting how well students will do in their studies while making sure the research methods we use are both accurate and interpretable. With the help of machine learning and XAI, we can now look at a lot of data to try and figure out what helps students succeed and what doesn't not. This is about finding the right balance between making predictions that we can trust and making sure we know why the predictions say what they do. Understanding and predicting how well students will do in their courses is a big challenge. It involves looking at many things, like how well they have done previously and what kind of background they come from. These things can tell us a lot about how likely they are to do well in the future. Identifying these contributing factors and how much they contribute to student dropout is an important area that lacks research. Many research studies have focused on either academic factors or non-academic factors when trying to predict student dropouts. In the study (Balbin 2019), they have explained how the non-academic features such as family income, size of the family, parent's involvement and support affects to the academic performance of a student. In our study we examine how various factors, both related to academics and non-academics, can impact students' performance.

The performance of a student is measured through various practices in a university such as examinations, assignments, quizzes, and vivas (Pillay et al. 2017). The sole goal of students, higher educational institutes, and universities is to improve the student's performance and reduce dropouts. For a long time, people have been trying to figure out the best way to predict student success. This has led to many different methods being tried, from basic statistics to more complex computer models. Unlike previous studies that predominantly focused on academic predictors, this research integrates both academic and non-academic predictors. In our study we use advanced machine learning algorithms, including CatBoost (Prokhorenkova et al. 2018) and XGBoost (Chen and Guestrin 2016), together with unique data sampling techniques like ADASYN (He et al. 2008). The objective is to improve the precision of predicting student outcomes and provide useful insights. The scope of our study includes nine discrete machine learning models: Logistic Regression, Decision Trees, Random Forests, Support Vector Machines, Naive Bayes, Extreme Gradient Boosting, Gradient Boosting Machines, CatBoost, and AdaBoost. These models are employed to enhance the accuracy of our predictions. Nevertheless, our goal extends beyond only attaining high levels of accuracy in predictions. It is equally important to guarantee the transparency and understanding of the prediction process.

This is precisely the juncture at which explainable artificial intelligence (XAI) becomes invaluable. Through the application of XAI methodologies, we can obtain transparent explanations for the predictions generated by machine learning models. Our research focuses on two interpretable models, the Explainable Boosting Machine and Decision Trees, alongside two XAI techniques, LIME (Ribeiro, Singh, and Guestrin 2016) and SHAP (Lundberg and S.-I. Lee 2017). These explanations enable stakeholders to discern the key features that significantly influence the model's predictions. The institute professionals, lecturers and students can get an idea of what is the amount of risk that a particular student is having for getting dropout and what are the reasons for the prediction. In that case all the parties can take necessary precautions to avoid the situation. That is the importance of these combined solutions rather than a machine learning prediction.

1.2 Motivation

In today's complex educational landscape, student success is crucial for both individuals and institutions. However, many face challenges that hinder their academic journey. Early identification and support are essential to improve their educational trajectory and ensure success becomes the norm, not a privilege. Despite the common goal of improving student performance and reducing dropout rates, many students are unable to achieve these objectives due to a variety of academic and non-academic problems. This reality emphasizes the crucial importance of early identification and support for at-risk students, which has the potential to significantly modify many individuals' educational paths

The motivation behind this research stems from the idea that comprehending the complex elements that lead to student dropouts is just as important as making the prediction. Through the use of XAI for interpretability and machine learning models for prediction, this project seeks to both predict academic underperformance and provide insights into the underlying causes of these predictions.

Therefore, the motivation behind this research is to identify student dropouts in higher education at an early stage and discern the underlying reasons for these predictions.

1.3 Problem Statement

The problem of students dropping out of college is a major issue that affects education systems and students all over the world. Addressing the underlying causes of student dropouts in higher education is imperative because they have huge effects on their academic performance and the general effectiveness of schools. Academic achievements of students are a vital predictive factor for identifying students at risk of dropping out. Other than the academic features there can be numerous non-academic elements that also play a substantial role in contributing to student dropouts.

In higher education student performance is assessed through various methods including examinations, assignments, quizzes and vivas. The ultimate objective for educational institutions and students alike is to enhance student performance and reduce dropout rates. However, numerous students are unable to attain this objective due to excess of academic and non-academic reasons. Understanding the contributing factors and their impact on student dropout rates is an area that requires more in-depth research. Existing studies often focus primarily on academic or non-academic factors when predicting student dropouts neglecting a comprehensive analysis of both categories. Hence, this research aims to bridge this gap by exploring both academic and non-academic factors and employing various general classification machine learning techniques to enhance predictive accuracy.

Additionally, an important aspect that should be considered after predictive analysis is deciphering the output from the nine machine learning models that were employed in this investigation. Although performance is often assessed using metrics like accuracy and F1-Score, it can be difficult to determine how credible these findings are for educators and students who want to understand the rationale behind these predictions. By using XAI techniques it will be able to bridge this gap. Using XAI methodologies it allows stakeholders to determine which features have the most influence on the dropout prediction. This knowledge enables educators, students, and institutional experts to assess a student's likelihood of dropping out and to understand the factors that contribute to these projections, allowing them to take preventative action.

1.4 Research Aim Questions and Objectives

1.4.1 Research Aim

The aim of this research is to predict student academic performance and detect underperforming students early by analyzing the contributing features leading to their underperformance. This proactive approach aims to empower both students and academic staff to intervene and enhance the academic performance of identified students by taking necessary precautions according to the corresponding features that increase their chance of dropping out.

1.4.2 Research Questions

In order to address the requirements of finding a method to predict undergraduate academic performance while maintaining both accuracy and interpretability, as elaborated above, we intend to answer the following research questions through the proposed study.

- RQ1. What is the most suitable machine learning technique for predicting students' performance?
  A range of machine learning algorithms exists for making predictions, each with its own set of pros and cons. To accurately forecast student performance in this environment, it is important to identify the most suited machine learning technique.

- RQ2. How to adapt Explainable Artificial Intelligence (XAI) and Interpretation techniques to identify contributing features for each prediction?

  There can be several reasons that may affect student dropouts. It is important to identify those at risk students who have a higher chance of getting dropout while identifying the contributing factors for the result. By using XAI and interpretation techniques we have to investigate how to get an interpretation of the features affected by the result

- RQ3. What is the most effective XAI/Interpretable technique for student dropout prediction models in terms of interpretability?

  This seeks to identify the most effective XAI or the interpretable technique that offers superior interpretability for student dropout prediction models. This task involves comparing various XAI and interpretable model approaches to find one that best explanations for the predictions made by these models.

1.4.3 Research Objectives

In order to achieve our research aim, the following objectives will be achieved during the study.

- RO1: To apply and evaluate different machine learning techniques to predict undergraduate student's performance

  Focuses on employing and assessing a range of machine learning algorithms to forecast the academic performance of undergraduate students. This involves selecting appropriate models, training them with educational data, and then evaluating their predictive accuracy and reliability. The aim is to identify which technique(s) can most effectively predict student outcomes, thereby enabling early interventions to enhance student success.

- RO2: To investigate and adapt different Explainable Artificial Intelligence (XAI) techniques to evaluate the interpretability of the different features on the output for dropout prediction

  This explores the application and adaptation of various Explainable Artificial Intelligence (XAI) methods to understand how different features influence the prediction of student dropouts. It involves examining which factors are most predictive of dropout risks and how these factors are weighted within the models.

- RO3: To compare and analyze the results of different XAI models and evaluate the results to find the best suitable model that suits student dropout predictions.

  This objective aims to conduct a comparative analysis of different XAI models used in predicting student dropout focusing on evaluating their effectiveness and interpretability. By analyzing various XAI approaches this objective seeks to determine the most suitable model that provides clear, understandable, trustworthy insights into the dropout prediction process. This involves assessing the models' ability to accurately explain the reasoning behind their predictions, thus facilitating more informed decision-making in educational settings.

1.5 Significance of The Project

This study aims to make several significant contributions to the domain of higher education and predictive analysis with interpretability and explainability. First, by using XAI to predict the at-risk students while maintaining interpretability and transparency, it bridges the gap between accuracy and interpretability. It will enable the stakeholder to understand how the model predicted the answer and what factors contributed to the prediction.

This study will contribute to the development and finding of more effective methods for predicting academic performance and identifying at-risk students. Throughout the study different models will be used for the predictions and their performance will be analyzed. Also, this study will give insight into how predictive modeling and XAI can be utilized to enhance the student's performance and give a better outcome at the end.The outcomes of this research are intended to inform the development of targeted intervention programs at educational institutions. By implementing the predictive models developed through this study, schools can proactively identify at-risk students at an early stage, enabling timely and tailored support strategies that significantly reduce dropout rates.

1.6 Scope Including Delimitations

1.6.1 In Scope

Under this research, the following topics will be covered.

- Build predictive models to predict the academic performance of students
- Comparison of machine learning models with their performance
- Application of XAI to get interpretations for the dropout predictions to identify contributing factors for the dropouts.
- Evaluate and compare the results of the XAI models using user evaluations.

1.6.2 Out of Scope

The following will not be covered under the scope of this research project.

- Predicting the student performance and dropout rates of university students not related to the considered dataset

- Prediction of students' performance regarding other features that are not available in the dataset and other universities

1.7 Chapter Summary

This chapter describes the foundation for the dissertation by providing the introduction and the background in sections 1.1, 1.2 and 1.3. In section 1.4, it describes the research aim, questions and objectives. In section 1.5 it is the description on the significance of the project and the scope of the project is described in section 1.6. Moving forward the dissertation will provide the detailed implementations and steps described in this chapter.

2 Literature Review

Academic performance is one of the key factors that determine the success of a higher education student. Although different universities follow different strategies commonly universities measure the performance of a student by holding exams, assignments, presentations and other related mechanisms. From all of these measurements, not all the students will be able to score and perform well. There is a considerable number of students who fail to approach the required performance at the end. There might be several reasons that led them to that result. But if we can identify those students earlier the students them self and also the academic staff can help to get them out of that disaster.

2.1 General Machine Learning Techniques to Predict Student Dropouts

There are several solutions that have been designed to address this issue using machine learning and deep learning techniques (Lykourentzou et al. 2009).(Marwaha and Singla 2020). In the paper 'Using learning analytics to predict at-risk students in online graduate public affairs and administration education' they have analyzed the characteristics and behaviors of at-risk students in online education (Bainbridge et al. 2015). For that they have used an online Masters degree dataset. They have made a predictive model to predict the at-risk students and have identified the contributing features for it.

The paper titled 'A model to predict student failure in the first year of the undergraduate medical curriculum' by Baars, Gerard JA and Stijnen, Theo and Splinter, Ted AW which was published in 2017, is a comprehensive study focused on predicting academic failure among first-year medical students at Erasmus Medical School (Baars, Stijnen, and Splinter 2017). The goal of this study is to provide early assistance for first-year students who may not finish their academic work within the allocated two years. The methodology encompasses a comprehensive analysis of five consecutive cohorts, totaling 1819 students.

The dataset contains both pre-admission factors (age, gender, pre-university GPA, selection techniques) and post-admission variables (number of credits earned, exam participation, and success rates). Notably, the study discovered that students who had completed every exam by the fourth or sixth month (known as ”optimals”) had a 99 percent probability of completing the first-year curriculum successfully. Among the ”non-optimals” at the 6-month mark, the model could predict failure with a specificity of 66.7% and a sensitivity of 84.5%, based on the critical indicator of not passing any exams during the fourth and sixth months.

This research emphasizes the usefulness of logistic regression models in identifying students at risk early in their academic career, allowing for customized counseling measures. However, the study admits the need to include other criteria to improve prediction precision or permit early intervention. This study not only adds to the scholarly literature on student success and retention, but it also provides practical insights for educational institutions looking to reduce dropout rates through educated, data-driven methods.

The paper titled 'An Empirical Study for Student Academic Performance Prediction Using Machine Learning Techniques' which is published in 2020 is another study on predicting the final grade of Vietnamese students within the early phase of an educational program utilizing several models built using different machine learning techniques within the education domain (Ha et al. 2020). Initially, a student data dataset was created. It consists of demographic information, personal characteristics, information prior to administering to university and the first and second year academic performance data. Selecting the most suitable ML technique is done here by comparing the several performance measures including accuracy, precision and recall. These aforementioned models take 42 input variables and generate only one output variable. It represents the class (A,B,C,D,F) achieved by students.

ML techniques used to build the models or learners can be categorized as Rule Based, Neural Based and Statistical Based. OneR, PART, J48, Random Tree, Random Forest used to build Rule based learners. MLP (multilayer Perceptron) was taken to develop neural-based learners. Naïve Bayes and support vector machine techniques used for statistical-based learners. Considering both accuracy comparison and the classification performance comparison done within this research, both MLP and Naïve Baye perform similarly better than other techniques. Since this is an imbalanced dataset, class A and F have a small number of observations and both these techniques fail in these classes.

2.1.1 Class Imbalance Problem

In most of the student performance and dropout data related databases, there is an inherent class imbalance problem as most of the time the number of students who are dropping out is lesser than the number of students who have graduated. This is an important context that we should keep it attention. In the paper (Brandt and Lanzén 2021), they have clearly discussed how sampling techniques like SMOTE and ADASYN can handle this issue by sampling the data.

2.1.2 Physical Learning Environment

(Baars, Stijnen, and Splinter 2017) offers a comprehensive and insightful exploration into the critical issue of student dropout within higher education, particularly focusing on the context of medical education. They have effectively highlighted the global significance of student failure emphasizing its impact on both financial resources and the loss of potential talent. They support their assertions by citing various studies that delve into the multifaceted motives and factors contributing to student dropout, encompassing academic, personal, social, and environmental dimensions. According to this study in medical education relatively it has a low dropout rate compared to other fields of study. However, they draw attention to the notable trend that many unsuccessful medical students face challenges particularly in the first year of the curriculum.

They have mentioned about reviews of prior studies that have attempted to predict student failure within medical education using pre- and post-admission variables. While acknowledging that certain variables such as pre-university education GPA and selection methods, display some predictive value for student achievement, the authors critically examine the limitations and discrepancies inherent in these studies. They highlight issues including the lack of specificity and sensitivity in the predictive variables, variations in defining and measuring student failure, and the diverse curricula and contexts across different medical schools. From the research we can identify how the non-academic features can affect for the student dropouts. This sets a promising stage for further advancements in predicting and preventing student dropout in higher education.

The paper (Aulck et al. 2016) explores deeply into the process of predicting student dropout rates in the context of a major public university in the United States using machine learning techniques. Using a specially chosen dataset from the University of Washington covering undergraduate students enrolled from 1998 to 2006, the study offers a broad overview of the academic and demographic details that may indicate an individual's tendency to drop out of school. This dataset provides a basis for the research's predictive modeling efforts since it is rich in demographic information and academic achievement factors.

The application of three different machine learning models random forests, k-nearest neighbors (KNN), and regularized logistic regression is a key component of the study's methodology. The study carefully describes the technological foundations of these models, such as the creative feature selection process and the thoughtful imputation of missing data points, ensuring a strong analytical context. Regularized logistic regression outperforms other approaches in terms of prediction accuracy and the Area Under the Curve (AUC) for the ROC curves, according to the detailed research, indicating that it is effective in navigating the complex network of factors that affect student retention. According to the study, GPA in important courses like chemistry, English, mathematics, and psychology is a strong indicator of how persistent a student would be. The results suggest a promising direction to further research and suggest at the untapped potential of different machine learning techniques to further improve prediction accuracy.

2.1.3 E-Learning Environments

In the realm of e-learning education systems, various studies have utilized attributes associated with interaction within the learning environment. (Kuzilek et al. 2015) utilized these attributes, achieving an accuracy of 93.4%. Similarly, (Chui et al. 2020) employed similar attributes, along with module presentation-related ones, and attained an accuracy range between 92.2% and 93.8% when predicting at-risk students. Focusing on interaction attributes within study courses, (You 2016) used metrics like interaction time with resources, student engagement with problems and submissions, and study habits to achieve an AUC between 0.62 and 0.83. Other studies incorporated metrics such as the volume of emails sent and evaluations made. Also they obtained significant predictive outcomes and emphasized the importance of quiz marks as a predictor. (Kuzilek et al. 2015) used neural networks to forecast students likely to submit their assignments punctually, utilizing data from student and peer activity, both combined and separate from course information, finding networks with higher predictive power.

2.1.4 Video Interaction Prediction

(Mbouzao, Desmarais, and Shrier 2020) has researched about a new aspect in dropout prediction using video recognition technology. As they have mentioned in the paper there are prior studies that employed data analysis techniques to forecast student outcomes in MOOCs. These studies used various features such as video interactions, engagement levels, online social networking, and online activity. Factors influencing student behavior, like video characteristics, course structure and student profiles are also explored. In the proposed solution they have introduced, three predictive metrics based on students' interactions with MOOC videos: attendance rate (AR), utilization rate (UR) and watch index (WI). They elaborate on how these metrics are calculated from student video-watching patterns. hey recognized failure patterns in 60% of students likely to drop out or fail based on their initial week interaction with Massive Open Online Courses (MOOC) videos, successfully identifying 78% of students who thrived. Although academic related features contribute for a predicting system of at-risk students, there might be several other considerable facts that led them to be unsuccessful in their journey. In the research idea its a potential gap to address. Especially in a country like Sri Lanka there can be several other facts such as economic facts, social facts, geographical facts and etc that might affect to the academic performance of a student. And in most of the studies (Mduma, Kalegele, and Machuve 2019) they have built the machine learning model considering different features.

2.2 Implication of XAI for the interpretation model results

Machine learning models serve as powerful tools for identifying and predicting students at risk of dropping out. But beyond mere prediction, understanding the factors influencing these outcomes is crucial. This is where Explainable Artificial Intelligence (XAI) comes into play, bridging the gap by shedding light on the determinants that the machine learning models rely upon to make their predictions.

2.2.1 What is XAI and why it is important

Explainable Artificial Intelligence (XAI) can be described as a set of processes and methods that will allow the user of the machine learning model to trust the results and output of the machine(IBM 2023).

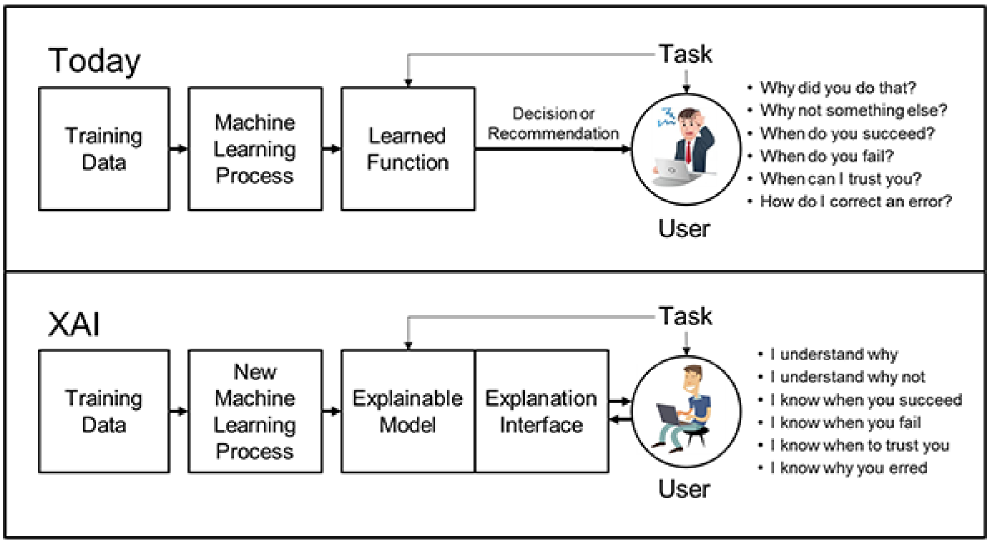


Figure 2.2.1: XAI vs Machine Learning

As depicted in Figure 2.2.1, normal deep learning models and machine learning models, it lacks interpretability which means the stakeholders will not be able to determine how the model will make their predictions. XAI comes into the picture to address this gap which will allow the stakeholders to understand how a model arrives at its predictions. It allows the user to get a clear understanding of the machine learning outcomes which is really important in the future decision making process.

2.2.2 How XAI can be used to address the problem

The problem that is going to be addressed in this research is predicting the academic performance of students and identifying the underperforming students earlier while maintaining both accuracy and interpretability. In the process of identifying the student it is important to know the reasons that led them to under-perform. XAI comes into the picture in this context (Turri 2022). Using XAI we will be able to identify those students while identifying the features and reasons that are involved in the final prediction.

In this paper (Mahboob, Asif, and Haider 2023) it has described the features that they have considered as marks of assignments, mid-term, lab exams, semester marks, total, grade, grade point (G.P.), quality point (Q.P.), grade point average (G.P.A.), and credit hours data of multiple courses. And they have come up with prediction results using general machine learning models. Concidering the features that they have used in the research if we can identify exactly what are the features that contributes to the result that will be more beneficial for the student as well as for the institute.

In (Jayasundara, Indika, and Herath 2022) reviewing the current landscape of Explainable Artificial Intelligence (XAI) in education, the paper highlights a notable gap in the literature pertaining to interpretable models for assessing student performance. To address this gap, the paper introduces an innovative Explainable Boosting Machine (EBM) tailored for achieving high accuracy and interpretability, particularly in the context of multi-class classification problems within the educational domain. They have used an Indian dataset for the project. The dataset includes multiple classes in the features and they have used multi-class classification models for the predictions.

Through comparative analysis with other transparent models like linear models, decision trees, and decision rules utilizing a student performance dataset sourced from India, the paper showcases how EBM outperforms in terms of providing both performance and interpretability. It demonstrates that EBM not only delivers accurate predictions but also offers comprehensive global and local explanations consistent with feature correlations and selection outcomes. Moreover, the paper hints at future research directions, underscoring the continuing significance of developing and refining such interpretative models for the education domain.

Likewise, if we can apply XAI to interpret and identify the reasons for the prediction it will be a great opportunity for the student to identify their cons as well as for the academic of higher educational institutes to enhance the outcome of their courses. It is another potential gap that can be identified from the studies.

2.3 Evaluating XAI models

Evaluating interpretations of the models is another challenge. As there are no accepted quantitative evaluation mechanisms for evaluating XAI outputs, the most common and accepted way of evaluating the outputs is by user studies related to the domain. In the paper titles 'Explainable AI methods for credit card fraud detection: Evaluation of LIME and SHAP through a User Study' by Ji, Yingchao which is published in 2021 (Ji 2021), describes how a XAI research can be evaluated using a user study.

The XAI methods selected for this thesis were SHAP and LIME because of their better explainability compared with others. (Hamelers 2021)

For evaluating XAI (Hoffman et al. 2018) has proposed a new method which has been widely used by several other papers. It has mentioned about 4 metrics that needs to be considered when evaluating an XAI output which are Satisfaction, Understandability, Trustworthiness and Sufficiency.

2.4 Research Gap

While existing studies have made significant progress in leveraging various machine learning and deep learning techniques to predict student dropouts within higher education, there remains a notable research gap concerning the comprehensive inclusion of non-academic factors influencing student performance and dropout. The current focus largely centers on academic performance metrics and their correlation with dropout prediction. However, these studies tend to over-look critical non-academic factors that may significantly contribute to student underperformance and subsequent dropout.

Factors such as economic, social, geographical and other non-academic elements might serve as crucial determinants affecting a student's academic journey, especially. These elements might influence a student's capacity to thrive within a higher education system. While the academic features certainly contribute to an effective predictive system for identifying at-risk students, neglecting the impact of these non-academic variables represents a research gap within the current literature.

Additionally, while some studies have attempted to identify critical features influencing academic performance, the interpretability of machine learning models remains limited. Incorporating XAI methods to not only predict student performance but also interpret the underlying reasons behind those predictions is an area that could significantly enhance the efficacy of dropout prediction models. Enhancing the interpretability of these models using XAI methods to pinpoint the specific features contributing to the outcomes could immensely benefit both students and educational institutions. Addressing this research gap by integrating non-academic factors and employing XAI to interpret predictions could lead to a more comprehensive understanding of student dropout, thereby fostering enhanced support systems and strategies for both students and educational institutions.

Another main gap that can be identified is the accuracy values of the outputs of the research. Most of them has comparatively low values because of the class imbalance problem that inherently causes in student dropout datasets. Generally in a year the number of students who are graduating in a university is higher than the students who are dropping out. Because of this problem in most of the datasets there is an inherent class imbalance problem within it. We need to properly explore the dataset and handle the problem for a better prediction.

These research gaps present a promising avenue for future studies, aiming to create more accurate and comprehensive predictive models for student dropout while providing actionable insights into the contributing factors behind these predictions

2.5 Chapter Summary

In this chapter it describes the related works to this study. It describes about the significance of exploring the problem of higher education student performance prediction (Chapter 2). Then it presents how machine learning can be used to address this problem, how the previous studies has address the problem and how the solutions are proposed in different environments (section 2.1). Furthermore, it presents how XAI can be used to apply in this problem and the previous studies in the domain (Section 2.2.2). Additionally, this chapter highlights the limitations in the current solution and the research gaps that seeks to this study in section 2.4.

3 Design and Approach

3.1 Research Design and Approach

This study will be focused on the applicability of machine learning models and XAI methodologies to build more accurate and interpretable solutions to predict at-risk students in higher education by utilizing the existing theoretical concepts and foundations. As it uses the existing theoretical concepts this research will fall under the category of deductive research according to Saunders' Research Onion Framework.

3.1.1 Data Collection

The dataset that was selected for the research is obtained from various disjoint databases that pertain to students who were enrolled in undergraduate programs at the Polytechnic Institute of Portalegre in Portugal (Martins et al. 2021). This data covers the records of students who attended the institute between the academic years 2008/09 and 2018/2019. It includes students from a wide range of undergraduate majors including agronomy, design, education, nursing, journalism, management, social service, and technology-related fields. It has both academic related features and non-academic related features of the students. Therefore it will enable us to identify how the different features contribute to students in their academic performance.

3.1.2 Data Exploration And Analysis

Understanding the dataset is a crucial part as it forms the foundation for the predictive models. As mentioned earlier, this dataset comprises data which are collected from a wide variety of students who follow different degree programs in a university. It has information on the demographic, socioeconomic, and educational factors of each student with the academic status of that student. It is essential to understand the significance of these attributes before diving into the process of training the models. The explorations and observations that were conducted are explained below.

The relationships were explored of the variables concerning the target variable. Among these relationships, it was observed that more than 99% of the student's nationality is Portuguese as it is visible in Figure 3.1.1. More than 99% students were local students and not international as you can see in Figure 3.1.2. Other than these two, there were several additional observations encountered during the data preprocessing phase that proved to be valuable.

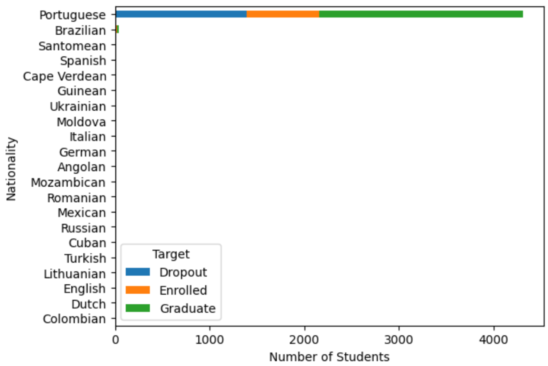


Figure 3.1.1: No of students vs nationality

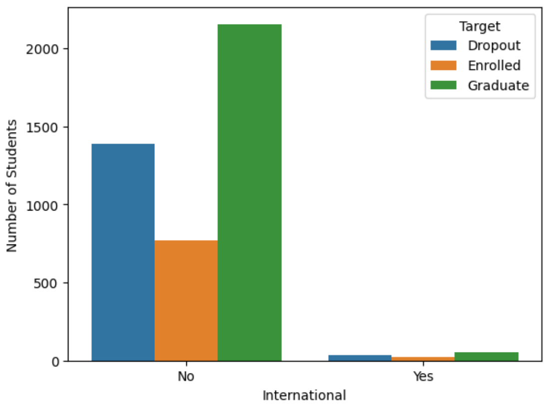


Figure 3.1.2: No of students vs international

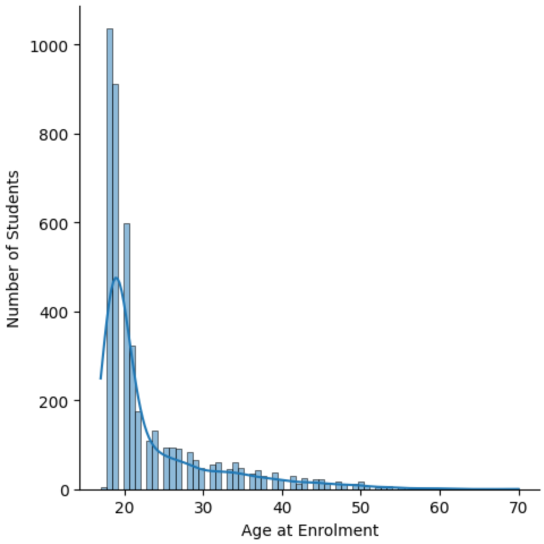


Figure 3.1.3: Age at enrollment

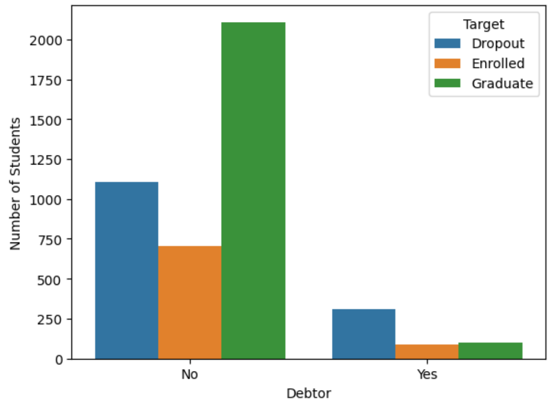


Figure 3.1.4: Distribution of Debtors

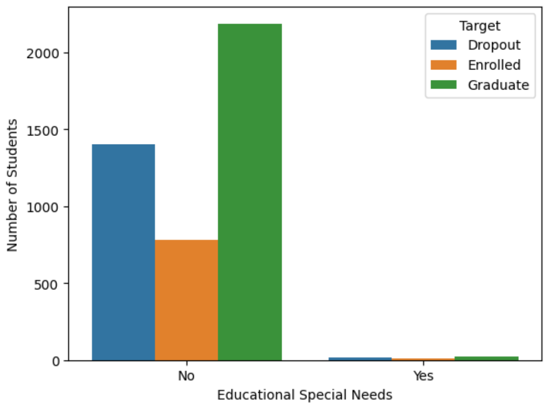


Figure 3.1.5: Educational special needs

From the martial status distribution, it depicts the martial status of a student when a particular student is enrolling in the course (Figure 3.1.6). It has a normal distribution that we can expect from a university student. The inclusion of educational special needs in Figure 3.1.5 emphasizes an important part of student support. The existence of special educational needs may need more resources and accommodations, while their absence may result in greater dropout rates. From Figure 3.1.4, it represents whether the student is a debtor or not. In most of the research, they have identified how crucially the financial background of a student is affecting to his/her studies (Yukseltruk and Inan 2006; Latif, Choudhary, and

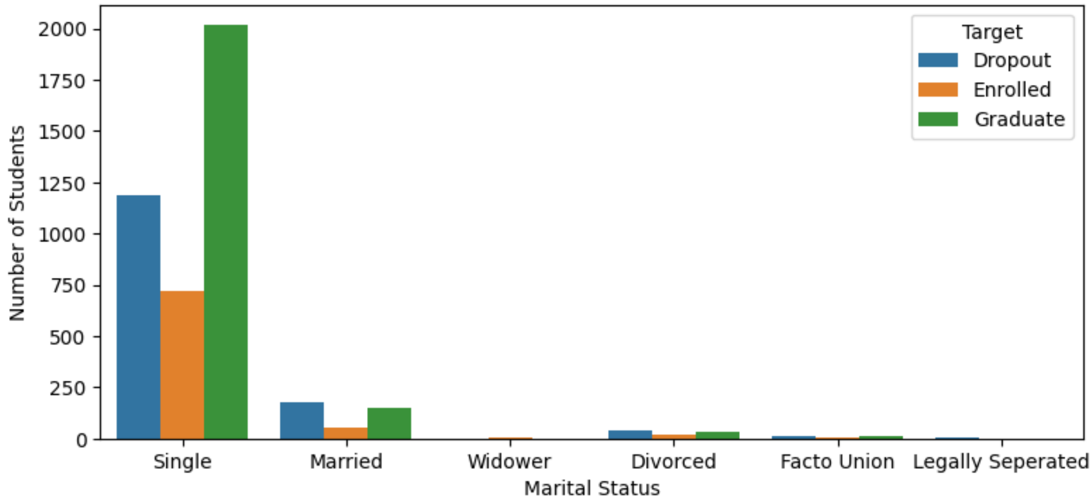


Figure 3.1.6: Martial status

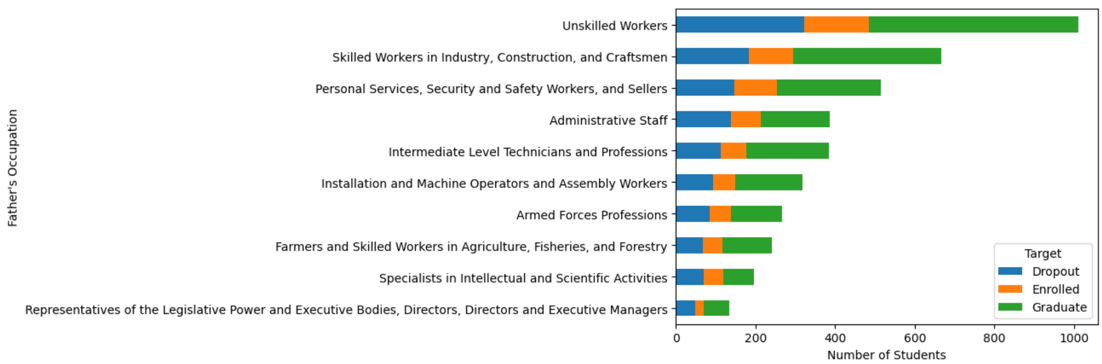


Figure 3.1.7: Father's occupation

Hammayun 2015).

From Figure 3.1.3 it represents the age distribution of the students when they join for the course. The age at enrollment shows a skewed distribution suggesting that a significant fraction of students are typically aged, falling within the average age range of undergraduates (Reyes 2023). The observed skewness in the student population may indicate that older students represent a minority, which might lead to distinct problems and distinct support requirements that could impact their tenacity in pursuing higher education.

From the Figure 3.1.7 and Figure 3.1.8 it represent what are the occupations that the mother and father are doing for a particular student.

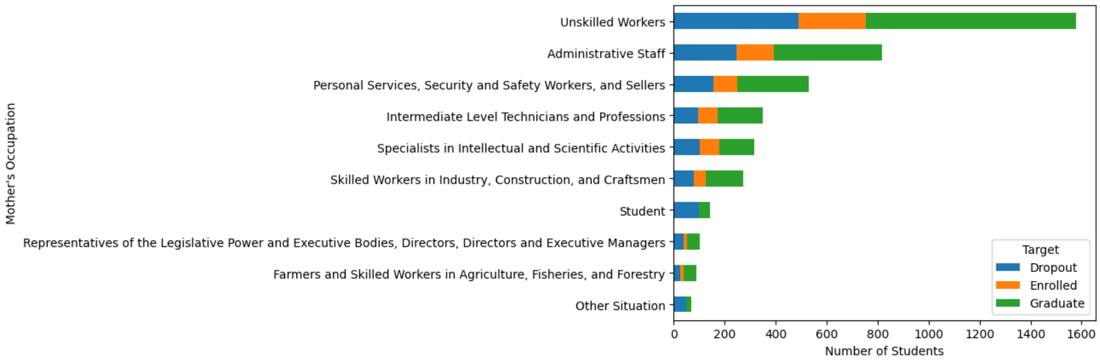


Figure 3.1.8: Mother's occupation

From these graphs, it is observable that there is some kind of a similarity between these two features.

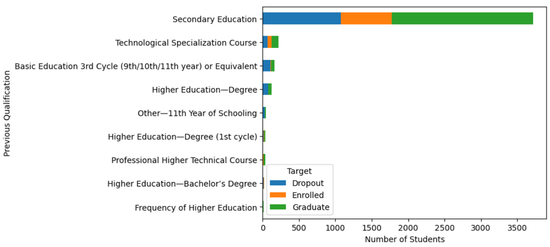


Figure 3.1.9: Previous education qualifications

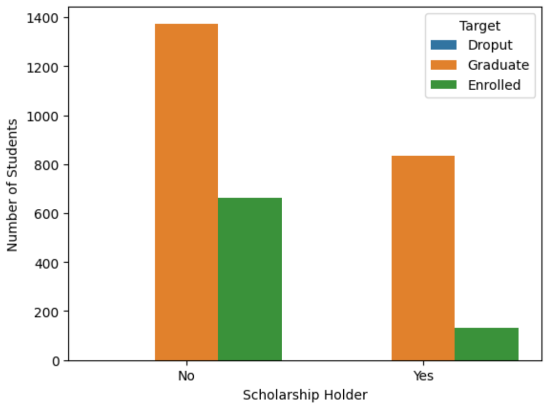


Figure 3.1.10: Scholarship holder

The previous educational qualifications of a student are another factor that will be really useful when predicting the student's academic performance in the current course. Figure 3.1.9 depicts the distribution of the student's data in their previous educational qualifications.

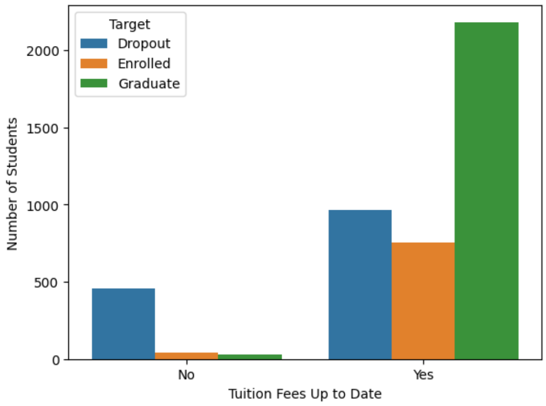


Figure 3.1.11: Tuition fees up to date

3.1.3 Data Preprocessing

During the initial analysis of the dataset, it was observed that there is a significant class imbalance, which needs to be addressed with utmost care and attention. The presence of this imbalance, which is marked by an unequal proportion of the target classes has the potential to greatly impact the effectiveness of predictive models frequently leading to a bias in favor of the dominant class. In order to address this problem here in this study we implemented data sampling techniques in a meticulous manner to create a more balanced distribution of students in the class.

Considering the characteristics of the dataset and the number of instances available, we have decided to opt for data over-sampling as our preferred strategy. The proposed approach aims to improve the representation of the minority class by generating supplementary samples which helps to balance its prevalence with that of the majority class. Utilizing such a strategy proves to be highly advantageous in scenarios where the minority class is significantly underrepresented and the dataset lacks the necessary size to allow for the elimination of majority class instances as is the case with under-sampling.

The Synthetic Minority Over-sampling Technique (SMOTE) is a method that is used to artificially generate new instances of the minority class by interpolating between existing instances and their nearest neighbors (Pradipta et al. 2021).

SVM-SMOTE is an advanced variant of SMOTE that makes use of Support Vector Machines (SVM) to identify instances that are located in close proximity to the decision boundary. The text artfully combines new examples in these domains enhancing the representation of the minority class (Bordia 2024).

Adaptive Synthetic Sampling, commonly referred to as ADASYN, is a technique that aims to address the issue of imbalanced class distribution. It achieves this by generating synthetic samples in close proximity to the minority instances that have been misclassified (Nian 2019). The key idea behind ADASYN is to adapt to the varying densities of the class distribution, thereby improving the performance of classification models. By proportionally increasing the amount of synthetic data in regions where the classifier's performance is not optimal, it effectively enhances the decision boundaries.

The incorporation of these sampling methods into the preprocessing pipeline was done with the intention of creating a more evenly distributed dataset. From using these techniques the goal was to improve the ability of the learning algorithm to accurately identify and classify instances of the class that is not well-represented. Simultaneously, the variable 'Target', which specifies the result for each student underwent label encoding.

The variable named 'Target' which serves as the target variable in this dataset was encoded using label encoding for model training purposes.

3.1.4 Modifying Categories of The Target Feature

In the feature named 'target' which is the target variable of the dataset, there were three classes which were named 'Dropout', 'Graduate', and 'Enrolled'.

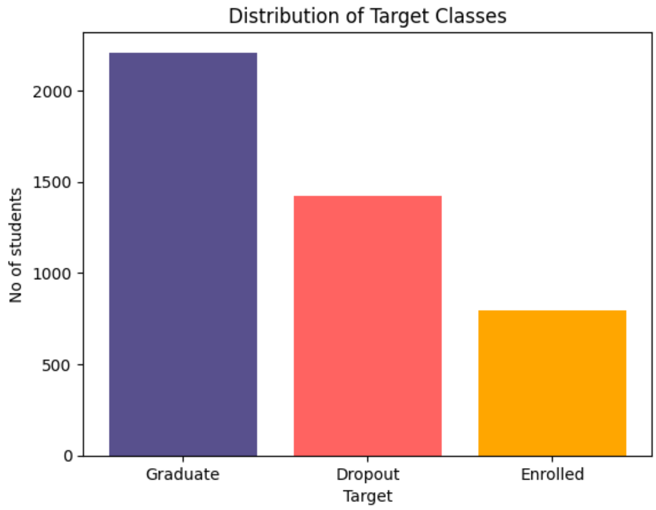


Figure 3.1.12: Target variable class distribution

The distribution of the students among the three classes is shown in Figure 3.1.12. The 'Graduate' class represents the students who obtained the degree in due time. The 'Dropout' class represents the student who has failed to obtain the degree in the due time. The primary objective of this study is to predict student dropouts while simultaneously identifying the key factors that contribute to this outcome. For this reason, the 'Enrolled' class was omitted from consideration and focused on predicting 'Graduate' and 'Dropout' statuses among students.

But still, the record distribution has a clear imbalance in the categories in the target variable. As shown in Figure 3.1.13 the two categories 'Graduate' and 'Dropout' has 60.9%, and 39.1% records respectively.

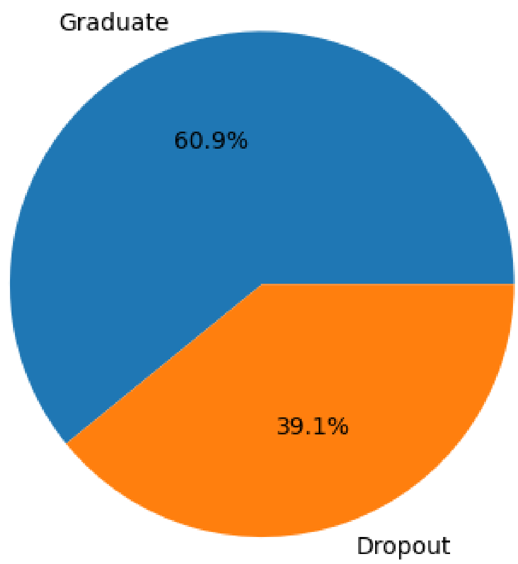


Figure 3.1.13: Target variable class distribution

The class imbalance issue was addressed by adopting over-sampling techniques as a pivotal approach. A detailed explanation of it is provided in the upcoming sections.

3.1.5 Handling Correlations

The correlation heatmap, shown in Figure 3.1.14, is an important tool used to analyze the relationships between variables in the dataset. This examination is important for understanding how the variables are related and how they affect each other's behavior. Some attributes showed a strong correlation, so we needed to carefully evaluate them to make sure our analyses were accurate.

A notable relationship was found between the variables 'Nationality' and 'International'. There seems to be a strong correlation here, which could mean that these features provide similar information about a student's background. In order to improve the model's performance, the 'International' attribute was carefully removed from the dataset. The purpose of this exclusion was to remove any unnecessary influence that could affect the accuracy of the model's predictions and to make sure that each variable had a distinct impact on the outcome.

In the same way, the heatmap showed a significant correlation between the occupations of the father and mother. 'Father's Occupation' was deliberately excluded during the preprocessing stage.

The strong correlation between these variables suggests that they may have a tendency to overshadow each other's predictive value. Variance Inflation Factor (VIF) with a threshold of 7 (Zaki et al. 2023) and based on the domain knowledge the following features were removed: 'International', 'Curricular units 2nd sem (approved)', 'Curricular units 2nd sem (grade), 'Curricular units 1st sem (credited)', 'Curricular units 1st sem (approved)', 'Curricular units 1st sem (grade)', 'Curricular units 1st sem (without evaluations)', 'Curricular units 1st sem (enrolled)', 'Curricular units 1st sem (evaluations)', 'Father's occupation' for a successful feature selection process.

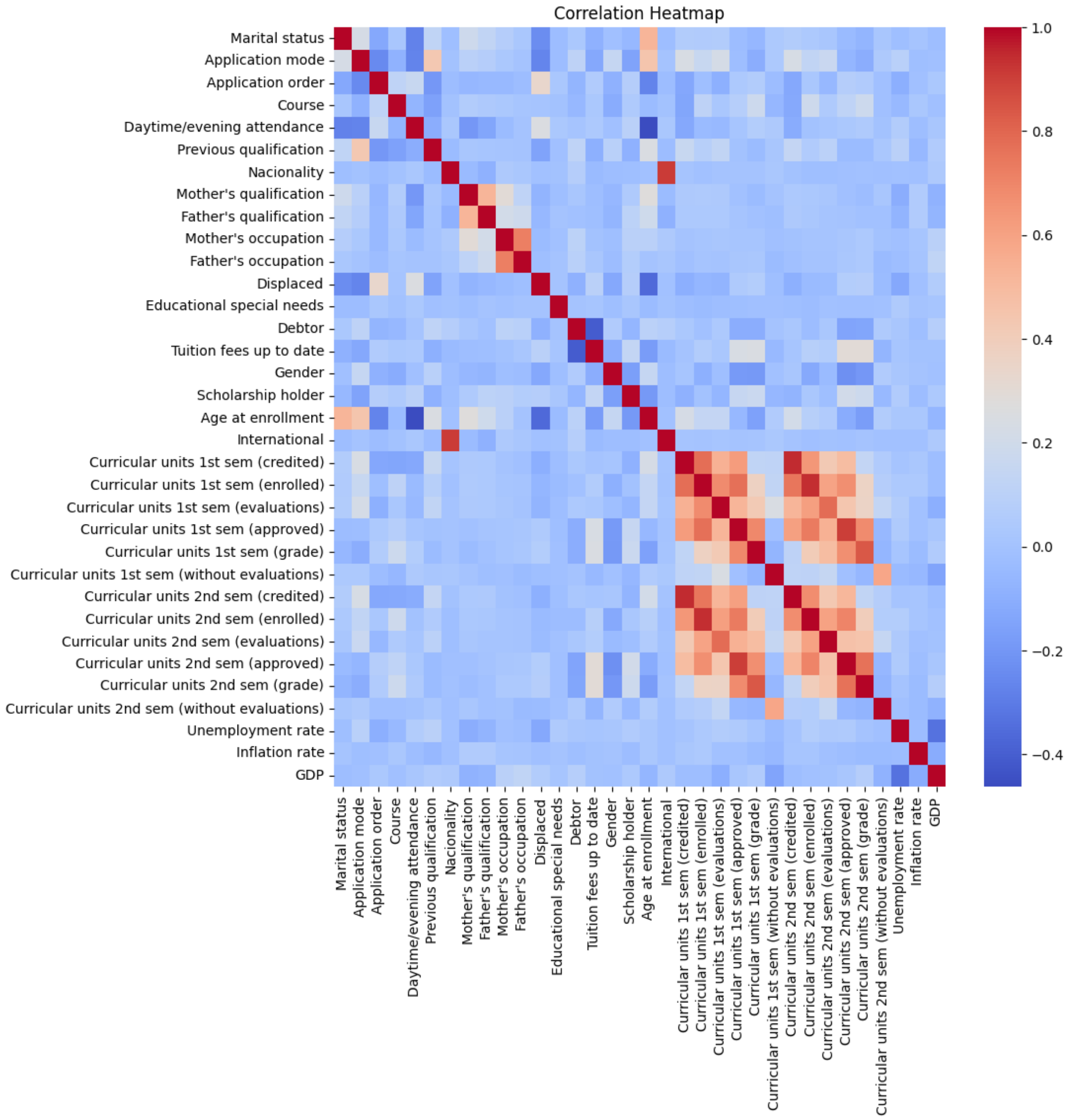


Figure 3.1.14: Correlation Heatmap

3.1.6 Handling Class Imbalance Problem

The class imbalance problem is one of the main issues in this dataset. Typically class imbalance problems can be handled using over-sampling and under-sampling techniques. Considering the dataset and the data distribution the study was done using over-sampling techniques in order to increase the number of records in the minor class. Moreover, in order to address the issue of class imbalance in the dataset, three distinct data augmentation techniques. Specifically, SMOTE, ADASYNC, and SVM SMOTE were used as the augmentation methods. The primary objective of these experiments was to augment the data within the minority class.

The Synthetic Minority Over-sampling Technique which is referred to as SMOTE is a data augmentation method that synthesizes new instances for the minority class (Chawla et al. 2002). SMOTE works by generating synthetic examples in feature space, interpolating between neighboring minority class instances. This helps balance the class distribution.

The next method, ADASYNC or Adaptive Synthetic Sampling is another data augmentation approach that was employed. ADASYNC generates synthetic samples for the minority class, placing more emphasis on challenging instances to improve the model's performance (He et al. 2008).

3.2 Chapter Summary

In this chapter, it presents the design and approach of the study. Section 3.1 describes the design and section 3.1.2 describes data collection and exploration details. Furthermore, section 3.1.3, represents the data pre-processing part. Section 3.1.5 and 3.1.6 represent the correlation handling part and the approach to handle the class imbalance problem in the dataset respectively.

4 Implementation of the models

4.1 Architecture Design

During the research, a detailed architecture was carefully designed to tackle the predictive task. In the Figure 4.2.1 it represents the proposed architecture. The process started with the initial dataset and went through a thorough preprocessing stage. This stage was important for getting the data ready for the next modeling steps. It made sure that the input data was clean, correctly formatted, and suitable for machine learning algorithms.

After preprocessing, the dataset was used as the starting point for the data augmentation phase. The augmentation played a crucial role in tackling the class imbalance problem, which was a major challenge identified at the beginning of the research. Three different augmentation techniques were used: SMOTE, SVM-SMOTE, and ADASYN.Each method was applied to generate synthetic instances within the minority class, thereby enhancing the dataset's balance. These augmentation techniques were selected because they are highly effective in enhancing the dataset with synthetic samples that closely resemble the feature space of the minority class.

Machine learning classifiers were trained and evaluated using augmented datasets. The classifiers included various algorithms such as Logistic Regression, Decision Trees, Random Forests, Support Vector Machines, Naive Bayes, Extreme Gradient Boosting, Gradient Boosting Machines, CatBoost, and AdaBoost. This varied selection was designed to thoroughly evaluate which model could most effectively handle the complicated nature of the augmented data.

By testing these models on the testing augmented datasets, we were able to determine the most effective data augmentation technique. This step was crucial in figuring out which synthetic sampling technique resulted in the most notable improvement in performance for the models.

After developing the best data augmentation technique, we proceeded to fine-tune the hyperparameters of the classifiers. The tuning process was done with great attention to detail, with the goal of improving the performance of each algorithm by optimizing its parameters. The objective was to optimize the models for the unique attributes of the expanded dataset in order to achieve the highest possible level of predictive accuracy.

After completing the hyperparameter tuning process, we identified the machine learning model that performed the best. It was chosen based on the F1 score evaluation metrics. This model represented the culmination of the research's model-building endeavors, encapsulating the most effective data preprocessing, augmentation, and optimization techniques.

For the last part of the research, we used interpretability methods on the model that performed the best. The first XAI technique is LIME which was used to gain a better understanding of how the model makes decisions on a specific level. At the same time, we used SHapley Additive exPlanations (SHAP) to get a broad view of which features were important and how each feature affected the model's predictions. Explainable Boosting Machine is another new model that was incorporated into the research that has the capability of interpreting the outputs generated.

Finally, the outputs from the Explainable Boosting Machine (EBM) model, the Decision Tree model, and the two XAI techniques, LIME and SHAP, are all regarded as interpretable outputs. As previously discussed, the outcomes interpreted through XAI methods were subjected to a user study for evaluation. Comprehensive details on these results, along with additional insights, are provided in the forthcoming sections.

4.2 Implenting Explainable Boosting Machine (EBM)

We incorporated Explainable Boosting Machine which is a machine learning model with the capability of providing interpretations. In the paper (Jayasundara, Indika, and Herath 2022), they have used this model to get their machine learning predictions and interpretations. We used the base EBM model in this study and the results of the model and interpretation will be discussed in the upcoming section.

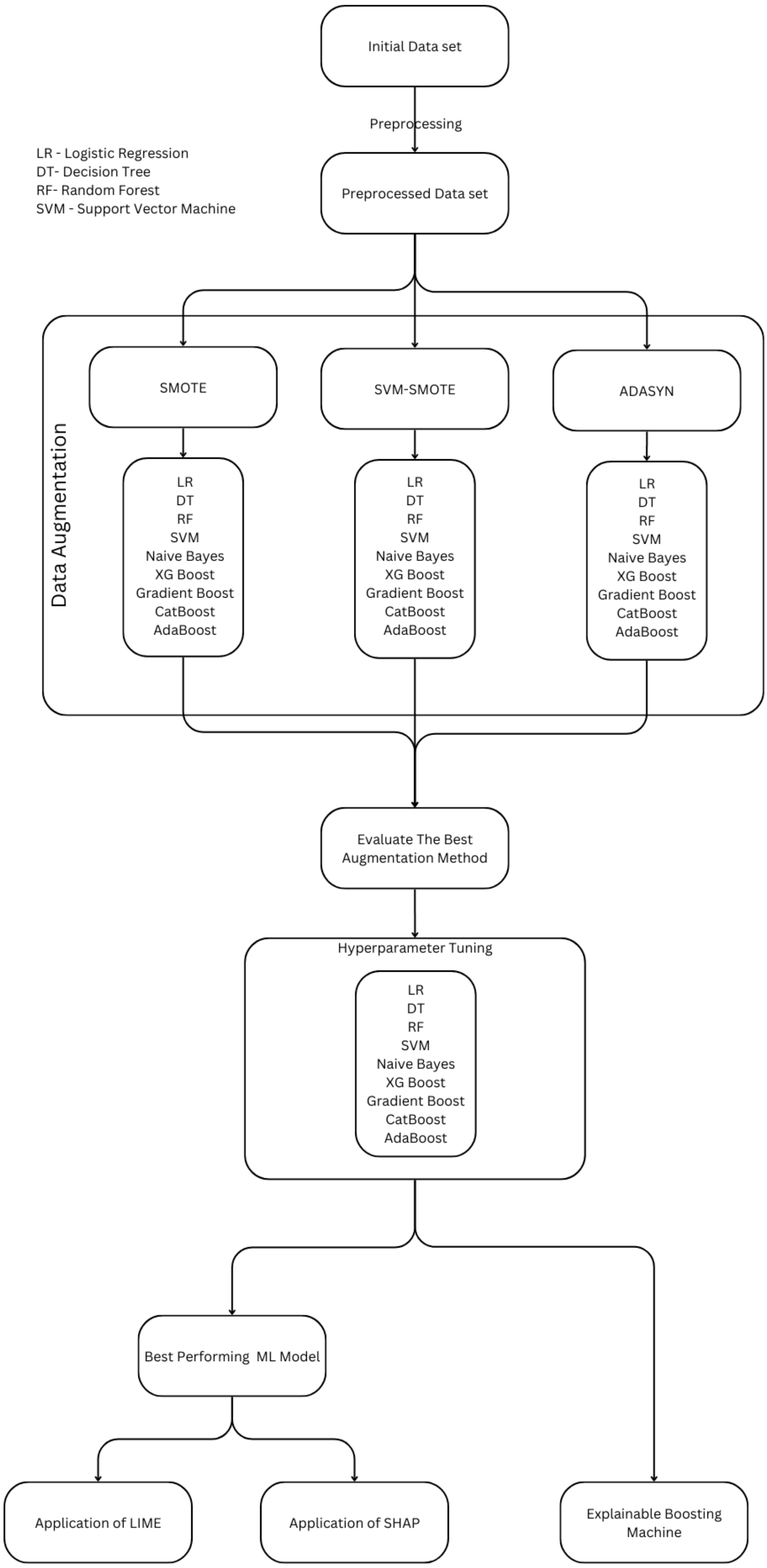


Figure 4.2.1: Architecure Graph

4.3 Experiments

In this research, to facilitate a comprehensive comparison, I selected a total of nine models, each with a different combination of data augmentation techniques. This rigorous evaluation allowed us to assess the effectiveness of the augmentation methods and their impact on model performance. The results and comparisons are elaborated upon in the subsequent sections of this report.

4.3.1 Classification Models

The research was focused on binary classification due to the presence of two classes in the target variable within the pre-processed dataset. There are widely used and well-established machine learning algorithms that are utilized for predictive tasks which are applicable for this study to tackle this problem (Adnan et al. 2021; Marwaha and Singla 2020).

As part of the study, we used a range of basic machine learning models. These models laid the groundwork for the initial classification efforts. Later on the research expanded to include boosting algorithms for better exploration and results (Al-Shabandar et al. 2019).

- Logistic Regression
- Decision Trees
- Random Forest Classifier
- Support Vector Machine(SVM)
- Naive Bayes Classifier
- XGBoost Classifier
- Gradient Boosting Classifier

- CatBoost Classifier
- AdaBoost Classifier

The reason for adopting these boosting techniques is that they have been proven to be effective in handling imbalanced datasets, which is relevant to the challenges of this research. Additionally, they can improve performance in tasks that involve classification (Tanha et al. 2020). In the next section, we will provide more details on the training methods used for the models and how they were evaluated.

This progression from basic classifiers to more advanced boosting methods highlights the systematic and analytical approach of the research, maintaining a consistent and scholarly tone throughout the study.

4.3.2 Explainable and Interpretable Models

After completing the classifier models, the research delved into the important field of exploring Explainable Artificial Intelligence (XAI). XAI helps to enhance our understanding of how complex models make decisions, which is crucial for ensuring transparency and trust in machine learning applications.

The LIME method was used to understand the complexities of model predictions. LIME provides valuable insights by approximating the predictions of any classifier in a way that is easy to understand and accurate with a focus on individual predictions.

In addition, the study used SHapley Additive exPlanations (SHAP) values to assess the influence of each feature on the prediction outcome. SHAP values offer a comprehensive way to assess the importance of features giving insights into both the overall and specific explanations for the model's behavior.

Incorporated into the system is the Explainable Boosting Machine (EBM) a model that combines the predictive capabilities of machine learning with the interpretability capabilities. EBMs offer valuable insights into the contributions of different features to the prediction.

The implementation of these XAI techniques will help to achieve reliable predictive results and effectively communicate its results to stakeholders. The thesis provides a thorough analysis of how these XAI methodologies have been implemented and their impact on the interpretability of the model outcomes in the upcoming sections.

4.3.3 Model Training

After handling the correlated features, finding out the optimal feature set and completing the pre-processing steps, the next step was to build the basic classification models for the prediction. This process can be unfolded into several stages. Initially, the models were checked and experimented with by feeding the data set without doing any data augmentations or any other class imbalance handling techniques. Initially, after the pre-processing stage dataset was splitter into three parts, training, testing and validation (Figure: 4.3.1).

Splitting Data Into Training and Testing

```
# Splitting the data into training+validation (80%) and testing (20%)
X_temp, X_test, y_temp, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Splitting training+validation into actual training (60%) and validation (20%)
X_train, X_val, y_train, y_val = train_test_split(X_temp, y_temp, test_size=0.25, random_state=42)
```

Figure 4.3.1: Training, Testing and Validation Split

Then SMOTE, ADASYNC and SVM SMOTE were applied to the dataset and rebuilt the models and made performance comparisons (S. Lee and Chung 2019). Based on observations and comparisons, One of the main important things that needed to be clarified is whether the model has been overfitted or not. To address this concern the data was split into training, testing and validation sets, and trained the models using the training set. All the models were trained under three datasets. Dataset augmented by SMOTE, dataset augmented by ADASYN and dataset augmented by SVM-SMOTE. All the nine models each were trained using the augmented dataset which was altogether 27 models. After the model training according to the results, the best model was selected for augmentation. The results of this experiment and the best model which was selected are mentioned in the next section.

After the selection of the model the next task was to fine tune the model parameters to get the best outcome. There are various methods that we can use for hyperparameter tuning such as Grid Search, Random Search, Bayesian Optimization, Gradient-based optimization and etc (Hossain and Timmer 2021). Among them, Grid Search is considered a highly effective method for determining the optimal parameters for a model. One major benefit of Grid Search is its transparency and thorough approach. Grid Search methodically explores a specified subset of hyperparameters, ensuring that all combinations within the grid are thoroughly examined. This thorough search is especially useful when the hyperparameter space is not too large and when we have a good grasp of the parameters that are likely to have the most impact on model performance (Navaz 2022). This approach offers a structured method for tuning parameters, guaranteeing that we discover the optimal combination of settings for our validation set.

However, it is important to note that methods such as Random Search and Bayesian Optimisation can be more efficient when dealing with large hyperparameter spaces. These methods intelligently explore and exploit the search space, but they don not guarantee the evaluation of all possible parameter combinations. Occasionally, the global optimum may be overlooked, particularly when dealing with a performance landscape that is highly irregular. Optimization methods based on gradients are highly effective when dealing with continuous and differentiable objective functions. However, they may not be the best choice for all hyperparameter tuning tasks, particularly those that involve discrete parameters or non-differentiable relationships. Evolutionary algorithms and advanced techniques, such as Hyperband, show great potential in tackling intricate and high-dimensional problems. However, they demand meticulous parameter tuning and can be computationally demanding to execute.

Given the requirements of our problem and the manageable size of the parameter space, Grid Search is the best option for a comprehensive yet efficient search. It strikes a good balance between being thorough and straightforward, offering a dependable way to carefully adjust the model parameters and achieve optimal performance.

From grid search, we were able to find out the optimal parameters for each machine learning model. This systematic and thorough approach involved examining all the different combinations of parameters for each model, which was a very detailed and time-consuming task. After carefully going through everything, CatBoost came out as the best performer. Because the dataset is not balanced, we focused on the F1 score as our main way of evaluating the model. The F1 score is important because it gives us a fair measure of both precision and recall.

We moved on from studying machine learning to exploring techniques for Explainable Artificial Intelligence (XAI). We specifically worked on using two popular XAI models, LIME and SHAP, to understand the CatBoost model. This model was found to be the most accurate in our comparison. We added these XAI models to help us better understand how our CatBoost model makes decisions and improve the transparency of its predictions.

4.3.4 Implementation of LIME

LIME was used to come up with local explanations that gave information about individual prediction cases. When we chose which cases to have LIME explain.

Instances with high, medium, and low model confidence levels were chosen so that we could see how the model acts at different levels of sure. To figure out what the model might be focused on in each situation, we used examples of both right and wrong predictions. This wide range of samples helped us figure out how accurate our model was in certain areas by showing us how different traits affected certain predictions. Out of the experimented outcomes one local interpretation was used for the evaluation of the LIME model in the user evaluation.

4.3.5 Implementation of SHAP

On the other hand, SHAP was used to give both local and global answers. In this case, SHAP values showed how each feature contributed to certain predictions. By adding up the SHAP values from all the cases, we were able to see how the features affected the model's estimates as a whole.

4.3.6 What are SHAPLEY Values

Shapley values are a concept of the cooperative game theory field. The objective of Shapley values is to measure each player's contribution to the game. The concept behind the calculation of Shapley values is fundamentally based on the game theory where 'n' players are participating in the game with an aim to achieve the reward 'a', and this reward is intended to be fairly distributed at each one of the 'n' players according to the individual contribution, such as Shapley Value.

In short, the Shapley value is the measure of the average marginal contribution of a feature for an instance among all possible bunchs in the sample. Let's understand this in detail.

Here E[F(x)] gives the average value and waterfall plot gives an explanation for a single instance. Here the model predicted value is f(x)=0.06. [0-dropout — 1-Graduate]. The values show how each value has contributed to the output. Being not a scholarship holder has been affected by -0.58 SHAP value. Tuition fees up to date have affected +0.48 SHAP value for the prediction and so on.

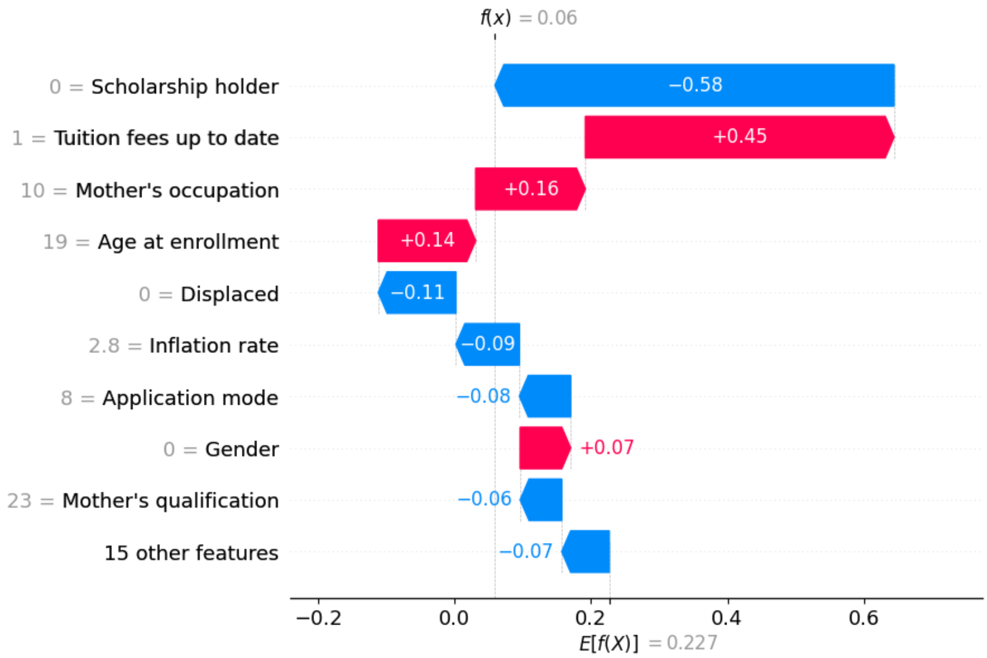


Figure 4.3.2: SHAP Waterfall plot

Our plan for using SHAP included the following:

Finding the SHAP values for a group of cases that are similarly varied as the ones in the LIME project. To get a better idea of the feature contributions for local explanations, we plotted the SHAP values of each prediction. To find the most significant features in general for global explanations, we added up all the SHAP values from the whole dataset. The selected instances were evaluated using SHAP. A single instance was used to get the local explanation for the user study and various global explanations were taken considering the entire dataset for the evaluation using the user study.

4.3.7 Insights from Implementing LIME and SHAP

The use of LIME and SHAP gave us a deeper understanding of our model in many ways.

Local Explanations: LIME and SHAP helped us understand why certain predictions were made, which let us trust the model and evaluate it closely case by case. Global Explanations: The overall SHAP insights showed the bigger trends and feature importance, which let us check that the model matched what we knew about the topic and what we expected. Trust and Transparency in the Model: We made our model more trustworthy and clear by knowing both the local and global reasons behind its predictions. This is very important for models that are used in sensitive and important areas, so that people who have a stake in the outcome can use it to make smart decisions. Adding LIME and SHAP to our study not only made our machine learning model easier to understand, but it also showed how important it is for AI-driven decision-making processes to be clear and easy to understand. This thorough way of explaining models has built a strong base for future study and uses, encouraging a culture of responsibility and knowledge in the use of AI technologies.

4.4 Chapter Summary

From this chapter, it represents the implementation of different models associated with the study. Section 4.1 describes the architecture design for the research study. In the other sub-sections it describes how each model and method were implemented such as the nine machine learning models, the implementation of the Explainable Boosting Machine and the implementation of the two XAI models.

5 Results And Evaluation

5.1 Results of Data Sampling

As the dataset was inherently imbalanced it needs to be experimented with using over-sampling techniques for the minority class. For that, the techniques SMOTE, SVM SMOTE, and ADASYN was used as minority class data augmentation. Following are the results that were obtained after applying SMOTE, SVM-SMOTE and ADASYN data augmentation methods.

5.1.1 SMOTE

After the pre-processing, we applied the SMOTE data augmentation technique first. First, we initialized the model and resampled the data. Then the dataset was split into training, testing and validation sets. After the resampling process we trained all nine machine learning models using that dataset. First we trained the Logistic Regression Model. For logistic regression we got Training Accuracy: 0.7907, Training F1 Score: 0.7979, Training ROC AUC Score: 0.8712, Validation Accuracy: 0.7534, Validation F1 Score: 0.8031, Validation ROC AUC Score: 0.8141. The ROC curve for the Logistic Regression model using SMOTE is shown in Figure 5.1.1.

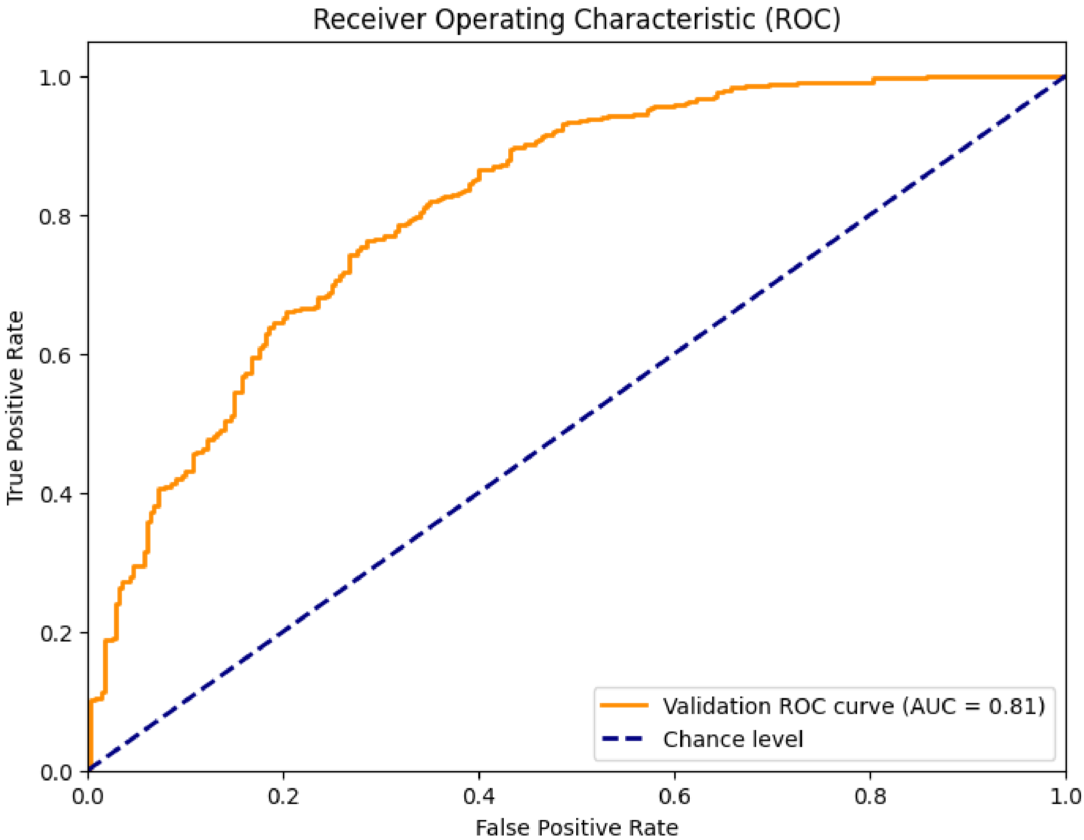


Figure 5.1.1: Logistic Regression - SMOTE ROC

Next we implemented the Decision tree model. In this stage with SMOTE we got the following accuracy,ROC and F1-Score values for the Decision Tree. Training Accuracy: 0.7797, Training F1 Score: 0.7777, Training ROC AUC Score: 0.8579, Validation Accuracy: 0.7617, Validation F1 Score: 0.7995, Validation ROC AUC Score: 0.8244.

Next model is Random Forest Classifier. From the basic model, we got the evaluation values as Random Forest Training Accuracy: 0.8337, Random Forest Training F1 Score: 0.8424, Random Forest Training ROC AUC Score: 0.9132, Random Forest Validation Accuracy: 0.7920, Random Forest Validation F1 Score: 0.8360 and Random Forest Validation ROC AUC Score: 0.8574.

Next for the Support Vector Machine, we got the values as the following. SVM Training Accuracy: 0.7390, SVM Training F1 Score: 0.7536, SVM Training ROC AUC Score: 0.8319, SVM Validation Accuracy: 0.7080, SVM Validation F1 Score: 0.7644 and SVM Validation ROC AUC Score: 0.7798.

Naive Bayes Training Accuracy: 0.7481, Naive Bayes Training F1 Score: 0.7706, Naive Bayes Training ROC AUC Score: 0.8432, Naive Bayes Validation Accuracy: 0.7383, Naive Bayes Validation F1 Score: 0.8013 and Naive Bayes Validation ROC AUC Score: 0.7885 are the evaluation values that we got for the Naive Bayes model with the SMOTE augmented dataset. After the general machine learning models we incorporate some boosting models into our study. Firstly from the XGBoost model we got the following outputs. XGBoost Training Accuracy: 0.8311, XGBoost Training F1 Score: 0.8383, XGBoost Training ROC AUC Score: 0.9037, XGBoost Validation Accuracy: 0.8030, XGBoost Validation F1 Score: 0.8427 and XGBoost Validation ROC AUC Score: 0.8693.

From the boosting model, Gradient Boost we got Gradient Boosting Training Accuracy: 0.7957, Gradient Boosting Training F1 Score: 0.8120, Gradient Boosting Training ROC AUC Score: 0.8730, Gradient Boosting Validation Accuracy: 0.7879, Gradient Boosting Validation F1 Score: 0.8358 and Gradient Boosting Validation ROC AUC Score: 0.8327.

As depicted in Table 5.1.2, from the basic CatBoost model we got the results as CatBoost Training Accuracy: 0.8390, CatBoost Training F1 Score: 0.8450, CatBoost Training ROC AUC Score: 0.9156, CatBoost Validation Accuracy: 0.8196, CatBoost Validation F1 Score: 0.8475 and CatBoost Validation ROC AUC Score: 0.8756.

Finally from the ADABoost model, we got the output values as AdaBoost Training Accuracy: 0.8250, AdaBoost Training F1 Score: 0.8331, AdaBoost Training ROC AUC Score: 0.9051, AdaBoost Validation Accuracy: 0.8099, AdaBoost Validation F1 Score: 0.8410 and AdaBoost Validation ROC AUC Score: 0.8663.

Algorithm	Training Accuracy	Validation Accuracy	Training F1-Score	Validation F1-Score
Logistic Regression	0.7907	0.7534	0.7979	0.8031
Decision Tree Classifier	0.7797	0.7617	0.7777	0.7995
Random Forest Classifier	0.8337	0.7920	0.8424	0.8360
SVM	0.7390	0.7080	0.7536	0.7644
Naive Bayes	0.7481	0.7383	0.7706	0.8013

Table 5.1.1: Using SMOTE - Standard Models

Algorithm	Training Accuracy	Validation Accuracy	Training F1-Score	Validation F1-Score
XG Boost Classifier	0.8311	0.8030	0.8383	0.8427
Gradient Boost Classifier	0.7957	0.7879	0.8120	0.8358
CatBoost Classifier	0.8390	0.8196	0.8450	0.8475
AdaBoost Classifier	0.8250	0.8099	0.8331	0.8410

Table 5.1.2: Using SMOTE - Boosting Machines

These are the results from SMOTE. There we can identify how the models are behaving. We can identify Random Forest Classifier is overfitting the results. However with SMOTE, the Random Forest Classifier demonstrated the highest accuracy and F1-Score among the standard machine learning models while the CatBoost Classifier exhibited the best performance among the boosting classifiers.

5.1.2 SVM-SMOTE

The next augmentation method we incorporated is SVM-SMOTE. Following are the outputs from the nine models incorporating SVM-SMOTE.

Algorithm	Training Accuracy	Validation Accuracy	Training F1-Score	Validation F1-Score
Logistic Regression	0.7861	0.7507	0.7932	0.8004
Decision Tree Classifier	0.7801	0.7617	0.7789	0.7991
Random Forest Classifier	0.8265	0.8030	0.8327	0.8434
SVM	0.7565	0.7163	0.7614	0.7675
Naive Bayes	0.7333	0.7328	0.7612	0.7979

Table 5.1.3: Using SVM-SMOTE - Standard Models

Algorithm	Training Accuracy	Validation Accuracy	Training F1-Score	Validation F1-Score
XG Boost Classifier	0.8250	0.7934	0.8331	0.8370
Gradient Boost Classifier	0.7907	0.7948	0.8072	0.8403
CatBoost Classifier	0.8390	0.8209	0.8448	0.8475
AdaBoost Classifier	0.8238	0.8127	0.8297	0.8409

Table 5.1.4: Using SVM-SMOTE - Boosting Machines

With the experiment using SVM-SMOTE, the Random Forest Classifier showcased the highest accuracy and F1-Score among the standard machine learning models, while the CatBoost Classifier displayed the best performance among the boosting classifiers.

5.1.3 ADASYN

Following table represents the evaluation values for the models using the ADASYN augmentation.

Algorithm	Training Accuracy	Validation Accuracy	Training F1-Score	Validation F1-Score
Logistic Regression	0.7779	0.7521	0.7871	0.8018
Decision Tree Classifier	0.7519	0.7727	0.7899	0.8328
Random Forest Classifier	0.8216	0.7961	0.8309	0.8374
SVM	0.7335	0.7052	0.7465	0.7579
Naive Bayes	0.7335	0.7369	0.7616	0.8000

Table 5.1.5: Using ADASYN - Standard Models

With the results it is observable that with ADASYN there is a slightly noticeable performance improvement in both standard and boosting models compared

Algorithm	Training Accuracy	Validation Accuracy	Training F1-Score	Validation F1-Score
XG Boost Classifier	0.8174	0.7934	0.8214	0.8315
Gradient Boost Classifier	0.7508	0.7727	0.7902	0.8332
CatBoost Classifier	0.8397	0.8354	0.8578	0.8594
AdaBoost Classifier	0.8185	0.8099	0.8280	0.8487

Table 5.1.6: Using ADASYN - Boosting Machines

to SVM and SVM-SMOTE.

After an exhaustive exploration of the results, it is observable that there are slight improvements in accuracy values and F1-Scores with the use of these sampling techniques. Out of the three algorithms, it is observable that ADASYN has slightly noticeable outcomes compared to the other two models in terms of overall model performance. As a result, before training the model ADASYN was chosen as the sampling algorithm and used for the remainder of the work in this study.

5.2 Hyper Parameter Tuning

In our pursuit of developing an optimal machine learning model,hyperparameter tuning became an important step. We used Grid Search as our main tool to look through all of the hyperparameter spaces for each model we were studying as part of this process. This part goes into more detail about our approach, why we chose it, and how it has affected the overall results of our study.

Grid Search, a method known for being thorough, was the first step in our hyperparameter tuning journey. We started an exhaustive search to find the combinations that improve model performance by creating a full grid of hyperparameter values for each model. Logistic Regression, Decision Trees, Random Forest, SVM, Naive Bayes, XGBoost, Gradient Boosting, CatBoost, and AdaBoost were some of the models that were looked at in this step.

A specific set of hyperparameters was carefully chosen for each model based on how they might affect the results of the model. A lot of research was done on parameters like C, kernel, and gamma for SVM and n_estimators, max_depth, and learning_rate for Random Forest and XGBoost ensemble methods.

With the help of GridSearchCV from scikit-learn, we carefully looked at every possible mix of hyperparameters in our specified grid. Not only did this make sure that the whole search area was covered, but it also added cross-validation to check how well the model was doing and prevent it from becoming overfitting. The accuracy and the F1 score were the main metrics we used to judge, with the F1-score being the main metrics because of the imbalance nature of the dataset.

5.3 Results and Evaluation of Machine Learining Models

With hyperparameter tuning, machine learning models had the following results.

logistic Regression best parameters

- Best Parameters : {C: 10, max_iter: 98, penalty: l2, solver: lbfgs}
- Training Accuracy: 0.7779,
- Training F1 Score: 0.7871,
- Training ROC AUC Score: 0.8615,
- Testing Accuracy: 0.7521,
- Testing F1 Score: 0.8018,
- Testing ROC AUC Score: 0.8154.

Decision trees best parameters

- Best Parameters : {'criterion': 'gini', 'max_depth': 10, 'min_samples_leaf': 3, 'min_samples_split': 40}
- Decision Tree Training Accuracy: 0.8637

- Training F1 Score: 0.8665
- Training ROC AUC Score: 0.9399
- Decision Tree Testing Accuracy: 0.7782
- Testing F1 Score: 0.8197
- Testing ROC AUC Score: 0.8302.

Random Forest best parameters

- Best Parameters: {max_depth': 15, 'min_samples_leaf': 5, 'min_samples_split': 10, 'n_estimators': 100}
- Training Accuracy: 0.9269,
- Training F1 Score: 0.9286,
- Training ROC AUC Score: 0.9754,
- Testing Accuracy: 0.8113,
- Testing F1 Score: 0.8509 and
- Testing ROC AUC Score: 0.8788.

Support Vector Machine best parameters

- Best Parameters: {'C': 50, 'kernel': 'rbf'}
- Training Accuracy: 0.8714
- Training F1 Score: 0.8767
- Training ROC AUC Score: 0.9380

- Testing Accuracy: 0.7851
- Testing F1 Score: 0.8301
- Testing ROC AUC Score: 0.8507

Naive Bayes best best parameters

- Best Parameters: {'var_smoothing': 0.001}
- Training Accuracy: 0.7446
- Training F1 Score: 0.7644
- Training ROC AUC Score: 0.8136
- Testing Accuracy: 0.7452
- Testing F1 Score: 0.8026
- Testing ROC AUC Score: 0.7692

XGBoost best parameters

- Best Parameters: {'learning_rate': 0.5, 'max_depth': 3, 'n_estimators': 5}
- Training Accuracy: 0.8170
- Training F1 Score: 0.8252
- Training ROC AUC Score: 0.8972
- Testing Accuracy: 0.8154
- Testing F1 Score: 0.8508

- Testing ROC AUC Score: 0.8651

Gradient Boost best parameters

- Best Parameters: {'learning_rate': 0.2, 'max_depth': 3, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 10, 'subsample': 1.0}
- Training Accuracy: 0.8120
- Training F1 Score: 0.8256
- Training ROC AUC Score: 0.9037
- Testing Accuracy: 0.8127
- Testing F1 Score: 0.8531
- Testing ROC AUC Score: 0.8631

CatBoost best parameters

- Best Parameters: {'depth': 3, 'iterations': 40, 'l2_leaf_reg': 1, 'learning_rate': 0.1, 'loss_function': 'Logloss'}
- Training Accuracy: 0.8319
- Training F1 Score: 0.8378
- Training ROC AUC Score: 0.9141
- Testing Accuracy: 0.8264
- Testing F1 Score: 0.8603
- Testing ROC AUC Score: 0.8773

ADABoost best parameters

- Best Parameters: {'base_estimator__max_depth': 2, 'learning_rate': 0.1, 'n_estimators': 60}
- Training Accuracy: 0.8396
- Training F1 Score: 0.8464
- Training ROC AUC Score: 0.9231
- Testing Accuracy: 0.8058
- Testing F1 Score: 0.8442
- Testing ROC AUC Score: 0.8739

During the hyperparameter tuning stage, we learned a lot about how different models behave and how sensitive they are to their settings. This not only helped us choose the best model but also helped us learn more about how the models worked.

CatBoost, along with the ADASYN technique for dealing with varying data, has become the best-performing model in our study after careful hyperparameter tuning and careful application of different data-augmentation methods. This result shows how important it is to not only pick the right machine learning model but also use the right preprocessing methods to deal with the problems that come with datasets. With this result, we used the ADASYN and CatBoost models for our XAI Evaluation output generation.

5.4 XAI and Interpretation Results

We wanted to explain how well our model, CatBoost, could predict the future, so we started an exploration with two Explainable Artificial Intelligence (XAI) techniques, LIME and SHAP. These methods gave us a way to look at the model's choices and understand them in a very specific way. By using LIME, we learned more about local prediction-specific explanations, which showed how specific features affect certain results. In addition, SHAP gave us both local and global views on how important each feature was, which helped us see how the model was thinking across the whole dataset.

The following results for the local explanations are made for the same data instance.

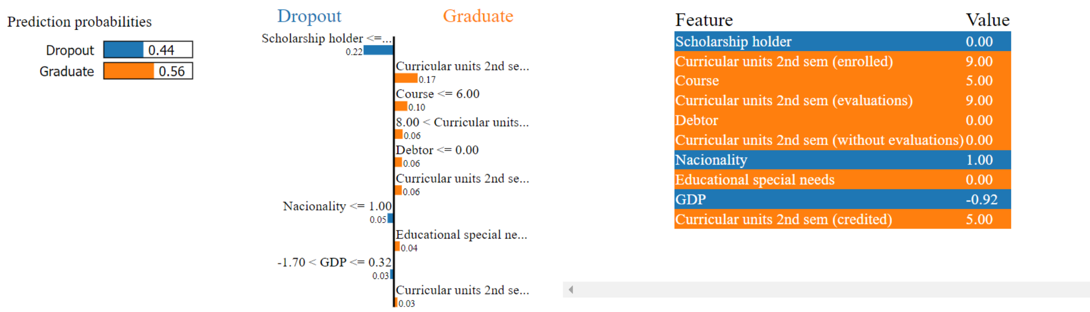


Figure 5.4.1: LIME local explanation

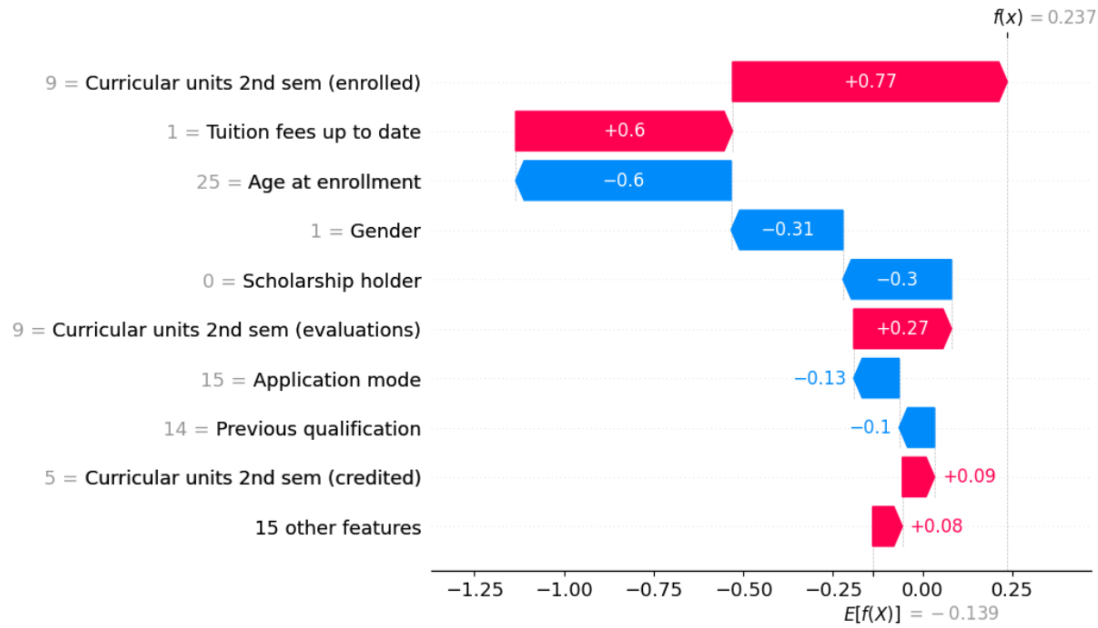


Figure 5.4.2: SHAP local explanation

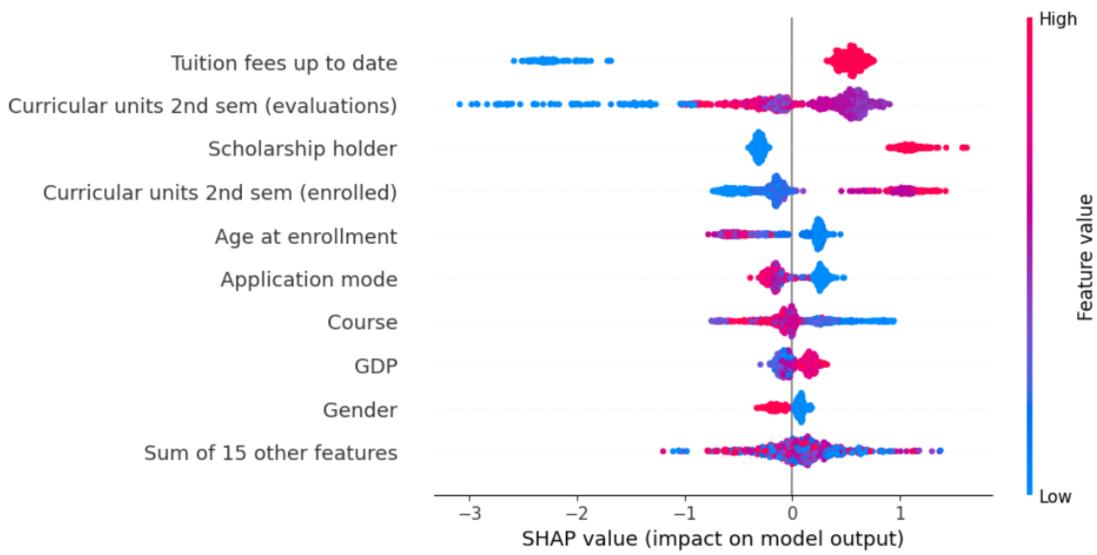


Figure 5.4.3: SHAP global explanation

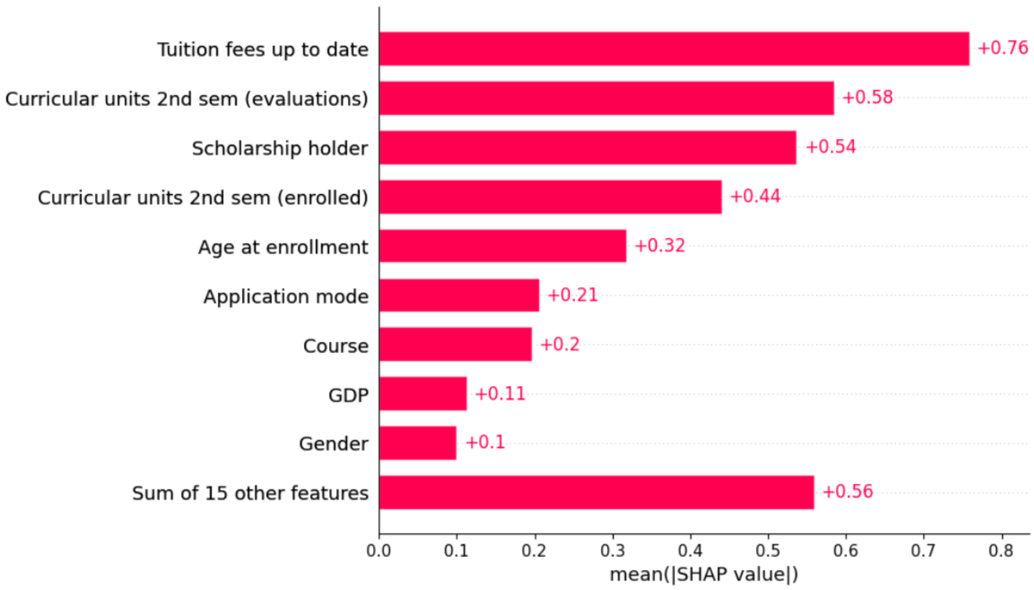


Figure 5.4.4: SHAP global bar chart explanation

Adding these XAI techniques to the CatBoost model not only made us trust it more by comparing its predictions to our subject knowledge and gut feelings, but it also opened the door for a more in-depth conversation about model accountability and transparency. By making it clear how the model works on the inside, we can now confidently involve stakeholders, giving them answers that are both deep and easy to understand. This work on XAI has been a major turning point in our study, bridging the gap between complicated machine learning algorithms and intelligence that can be used.

5.5 Interpret The Decision Tree

We interpret the decision tree using Graphviz. The initial interpretation was really complex and we performed pruning to get a better interpretation as show in Figure 5.5.1.

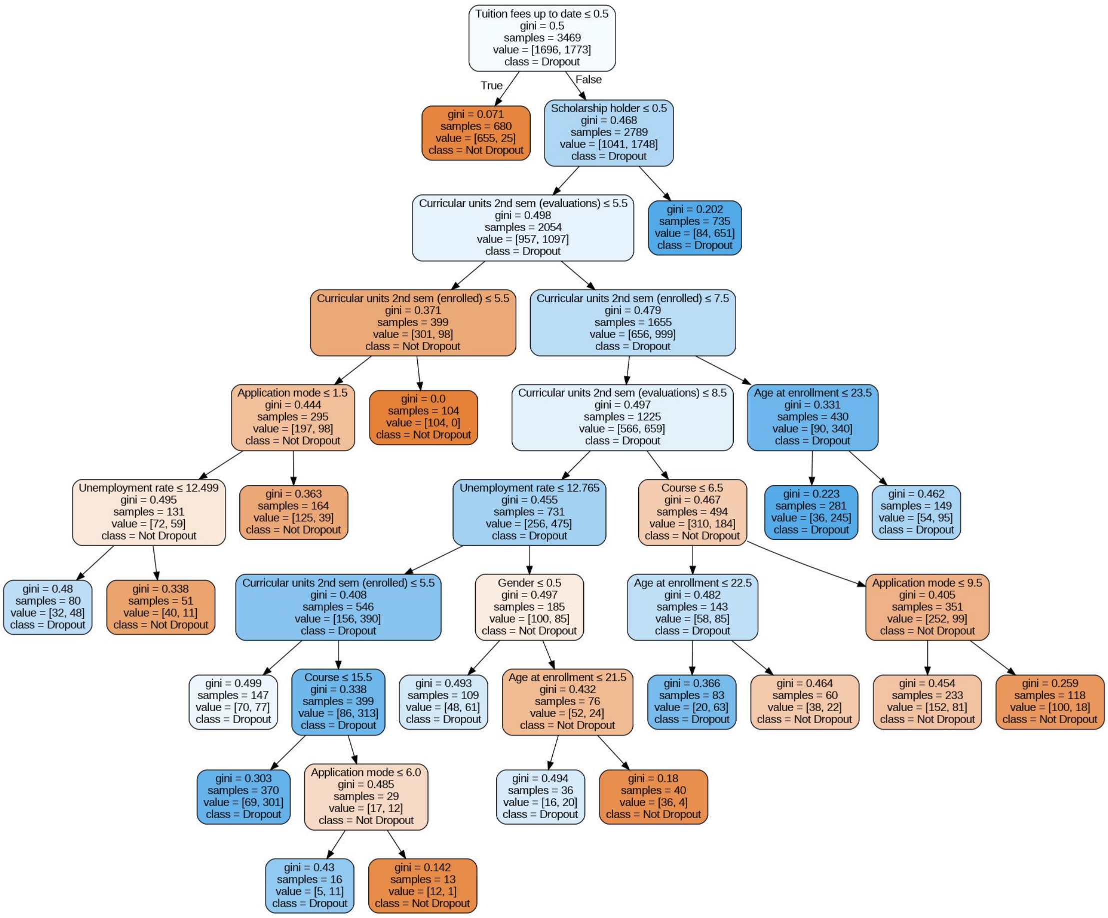


Figure 5.5.1: Pruned Decision Tree Interpretation

But the interpretation was still too complicated to get an clear idea about the model's behavior. And decision tree interprets the rules for the tree but not the features that are contributed to the result. Contributed factors are not directly highlighted.

5.6 Interpretation of the Explainable Boosting Machine (EBM)

The interpretation of the EBM model is shown in Figure 5.6.1. They it has provided a clear outcome of global explanation for the model's prediction. It provides an output somewhat similar to LIME's local output and SHAP's bar chart. However the issue with this EBM is its accuracy. Its accuracy was in the

60%s, which is comparably low. In other research studies also which has used this EBM model the accuracy was really low (Jayasundara, Indika, and Herath 2022). Because of this reason, EBM's interpretation was not considered in the user study for evaluation.

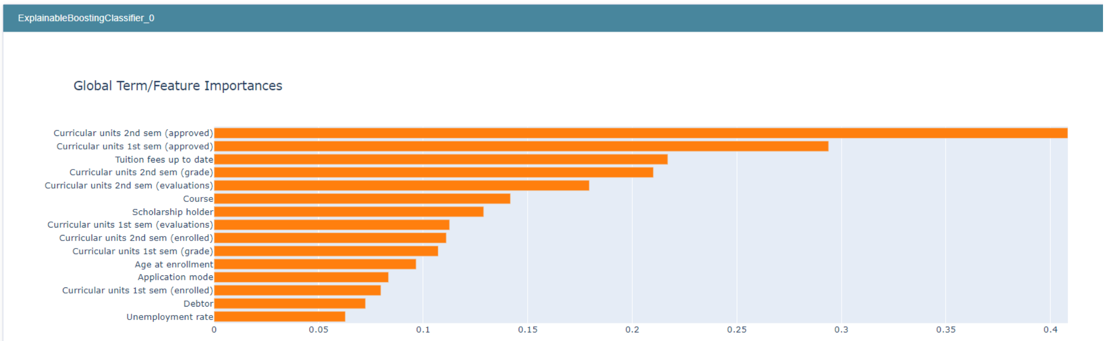


Figure 5.6.1: EBM Interpretation

5.7 Evaluation of XAI Models

To evaluate the efficacy and impact of our Explainable Artificial Intelligence (XAI) models, we conducted a user study, drawing participants from a diverse range of backgrounds. This study aimed to assess the interpretability and transparency provided by LIME and SHAP explanations as applied to our CatBoost model's predictions.

Participants were presented with model predictions alongside the explanations generated by these XAI techniques. Through a series of qualitative assessments and quantitative measures, we gauged the participants' ability to understand the model's decision-making process, their level of trust in the predictions, and the overall clarity of the explanations provided. This user-centric evaluation approach offered invaluable insights into the practical applicability of XAI methods, enabling us to refine our approach to model explanation and to ensure that our AI solutions are both accessible and trustworthy to end-users. The feedback and results from this user study have been instrumental in validating the effectiveness of our XAI implementations, marking a significant step towards achieving transparent and interpretable AI-driven decisions.

The questionnaire was based on two categories, one for the lecturers and academic staff to evaluate the dropout prediction explanations by LIME and SHAP (User Study 1). The second part was done by the students (User study 2). For this evaluation evaluators from various backgrounds were selected. Lectures from UCSC, student counselors from UCSC, student counselors and lectures and student counseling lectors from the University of Colombo Faculty of Science were involved in the user study. As students, UCSC students from various academic years participated in the study.

5.7.1 Results of The Evaluation of LIME - User Study 1

Following is the summary of the user study from the participation of university lecturers and counselors. It was chosen that professors and counselors would be part of this study because they have a lot of experience with situations where students drop out and the problems associated with them. Their professional viewpoints were invaluable in evaluating the XAI explanations of our models.

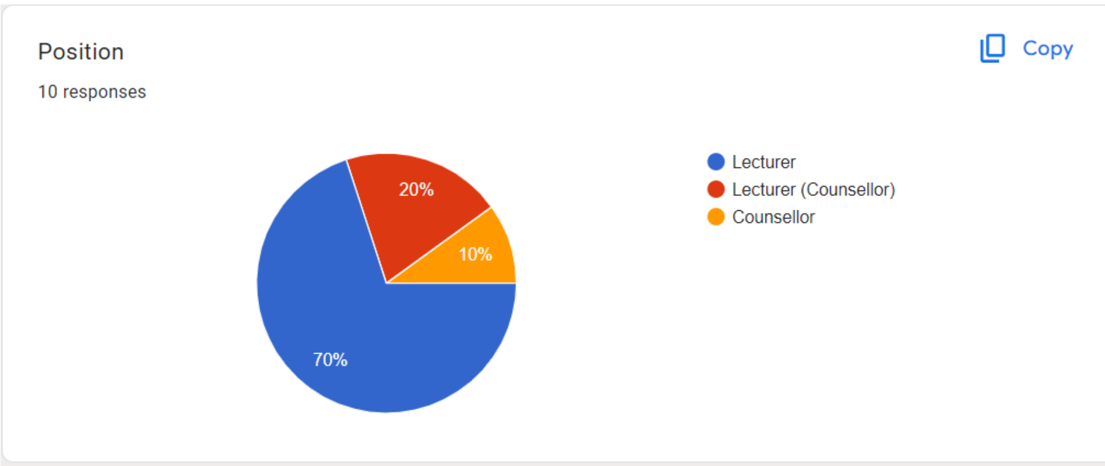


Figure 5.7.1: Participated Evaluators

From Figure 5.7.1 it represents the categories of the participated evaluators.

Next questions were based on LIME and SHAP. Figure 5.7.2 show the awareness of the participants on the model LIME.

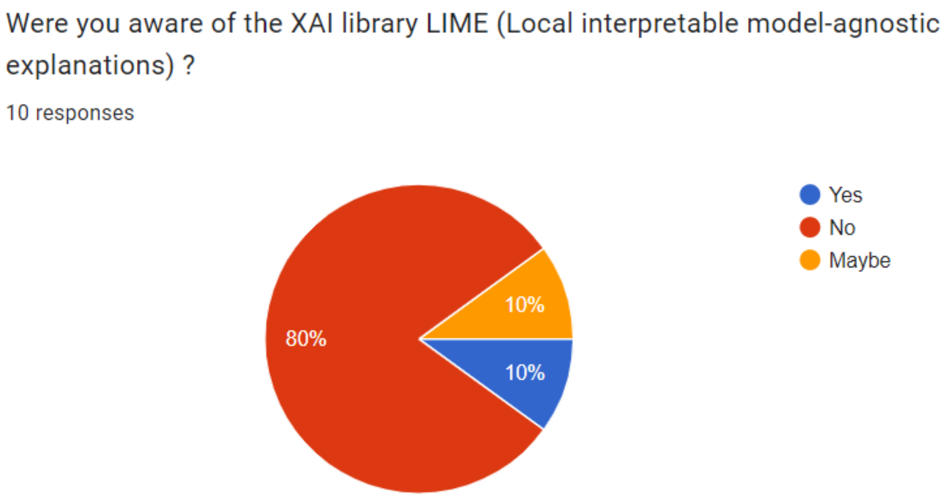


Figure 5.7.2: Awareness on LIME

Figure 5.7.3 depicts the rates that was given for the understandability of the LIME model.

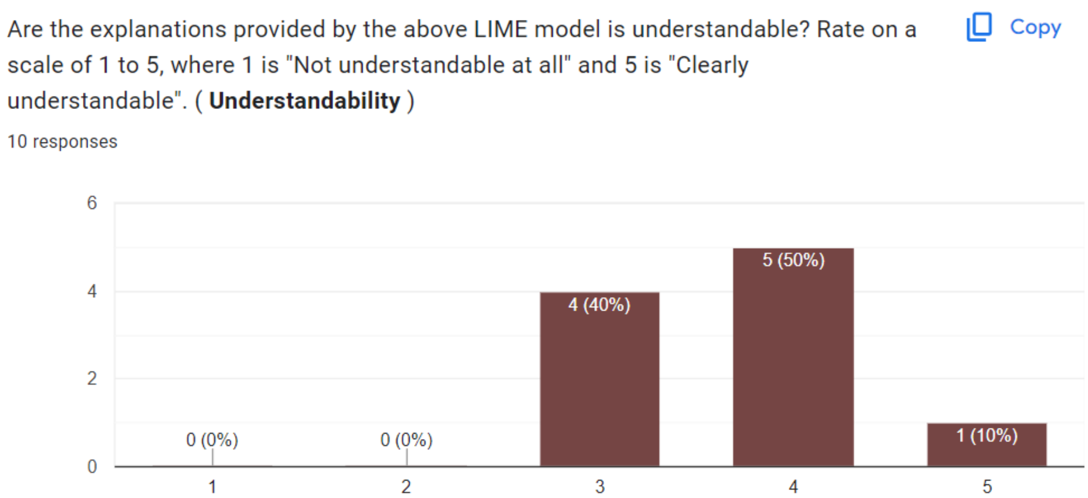


Figure 5.7.3: Understandability of LIME

Figure 5.7.4 depicts the rates that was given for the satisfaction of the LIME model.

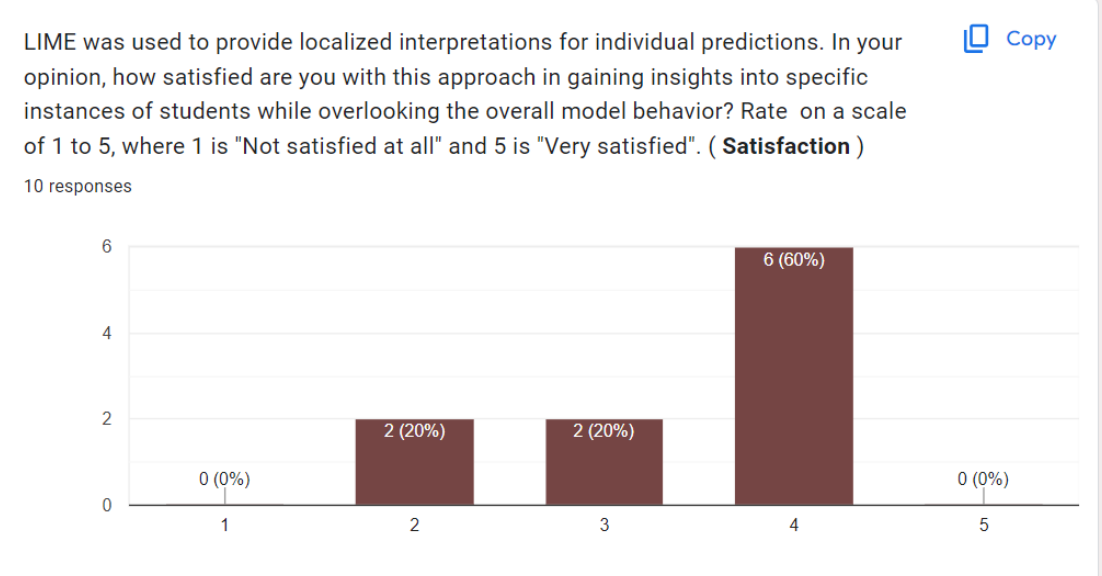


Figure 5.7.4: Satisfaction of LIME

Figure 5.7.5 depicts the rates that was given for the sufficiency of the LIME model.

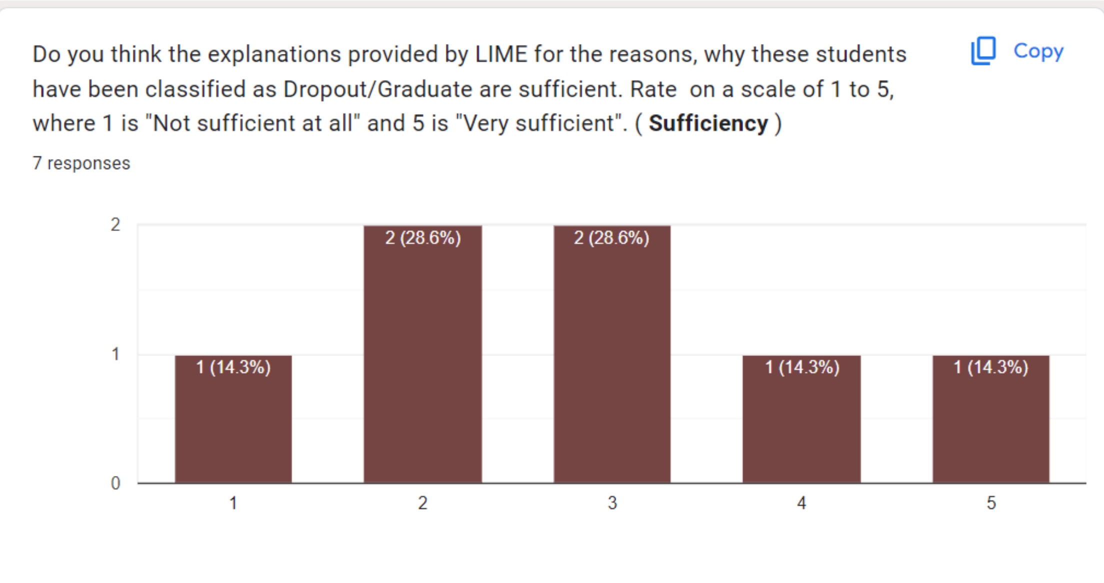


Figure 5.7.5: Sufficiency of LIME

Figure 5.7.6 depicts the rates that was given for the satisfaction of the LIME model.

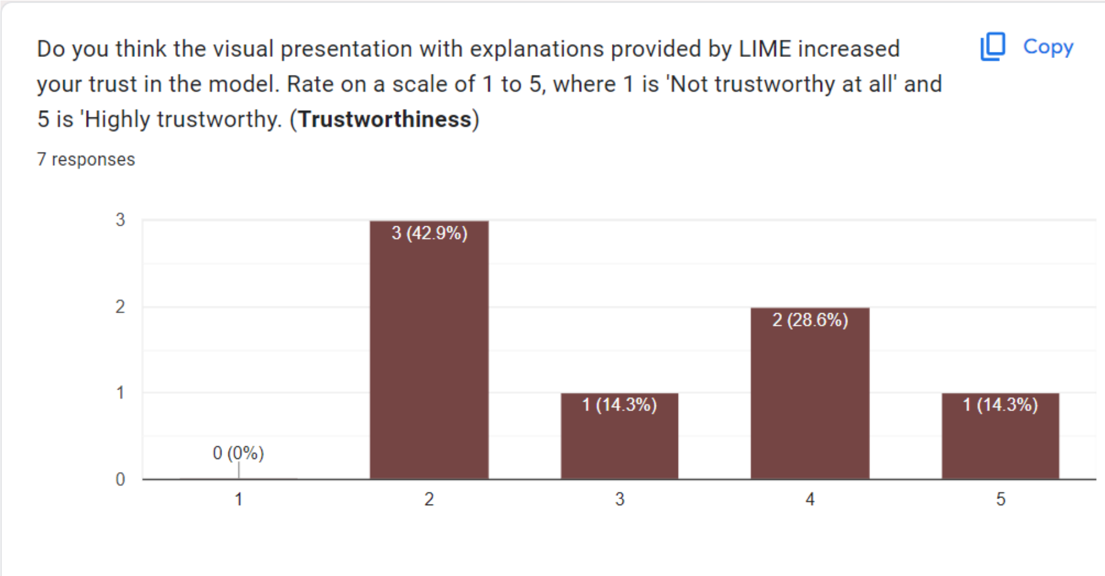


Figure 5.7.6: Trustworthiness of LIME

Based on the feedback that was given for LIME, it was clear that most of the participants were mostly satisfied with LIME's answers, with a large majority giving them a score of 4 out of 5. This shows that there is a lot of agreement with how LIME localized interpretations for each forecast.

However, when considering the sufficiency of the explanations regarding the reasons for students' dropout or graduation classifications, the responses were more varied. This suggests that the details that affect these results need to be looked at in more depth or using different methods.

People who trusted the model had a wider range of views, with answers that showed a healthy mix of doubt and belief in the explanations, which made the model more trustworthy. This feedback shows that we need to improve how the explanations are given and maybe make the explanations easier or harder depending on the user's level of knowledge.

Most of the people who answered thought the explanations were easy to understand; in fact, half of them gave them the highest grade. This makes it look like LIME's way of explaining each prediction is going in the right direction. However, users' opinions showed some confusion and disbelief in certain feature contributions.

5.7.2 Results of The Evaluation of SHAP - User Study 1

After getting evaluated the LIME model questionnaire was based on the SHAP model. Following are the questions and the respective responses for each question based on the SHAP model.

Figure 5.7.7 shows the awareness of the participants on the model SHAP.

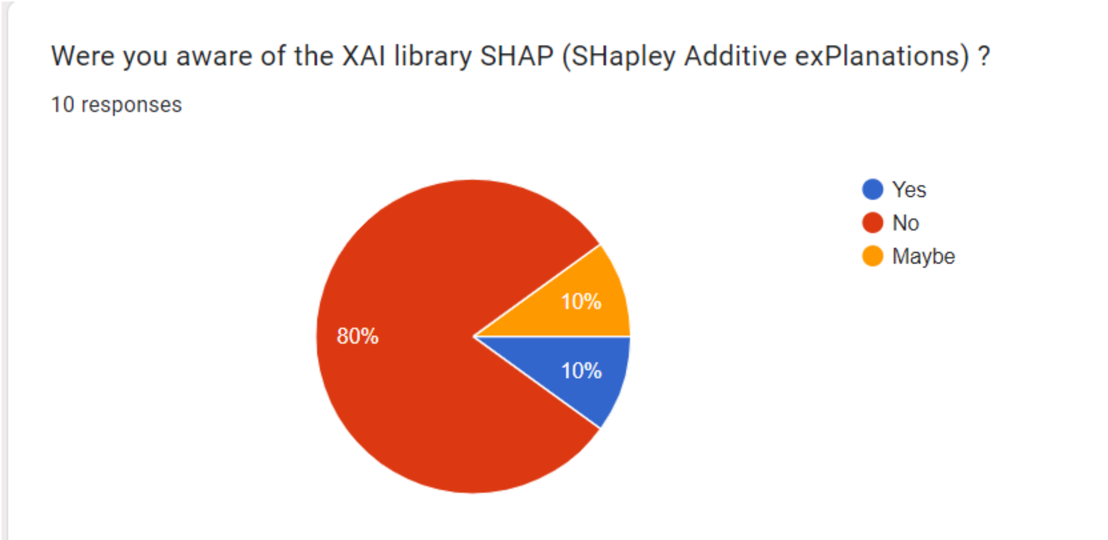


Figure 5.7.7: Awareness on SHAP

Figure 5.7.8 depicts the rates that was given for the understandability of the SHAP model.

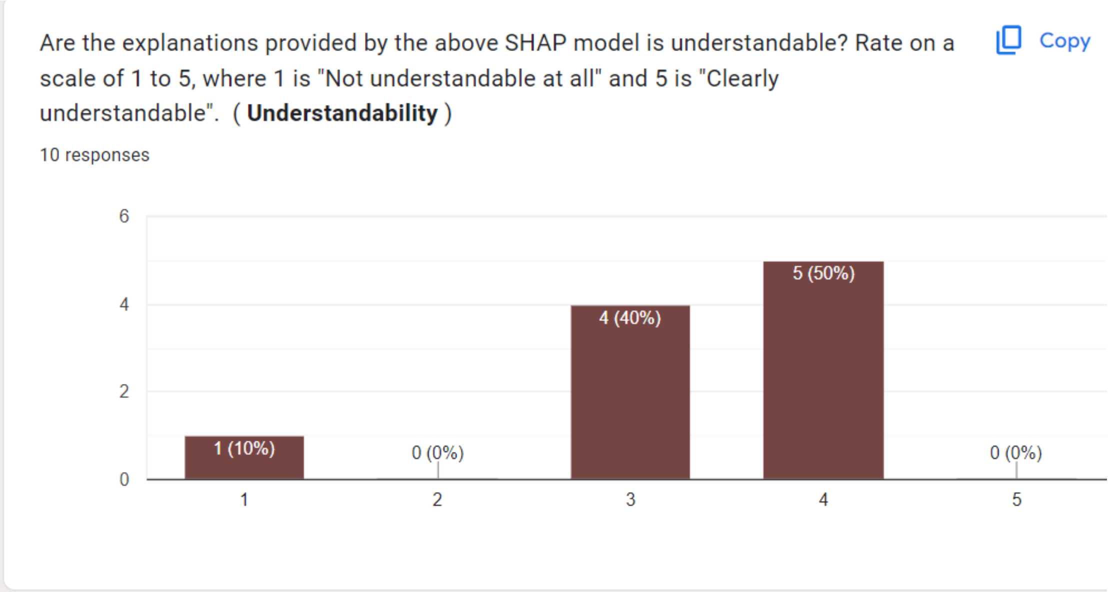


Figure 5.7.8: Understandability of SHAP

Figure 5.7.4 depicts the rates that was given for the satisfaction of the SHAP model.

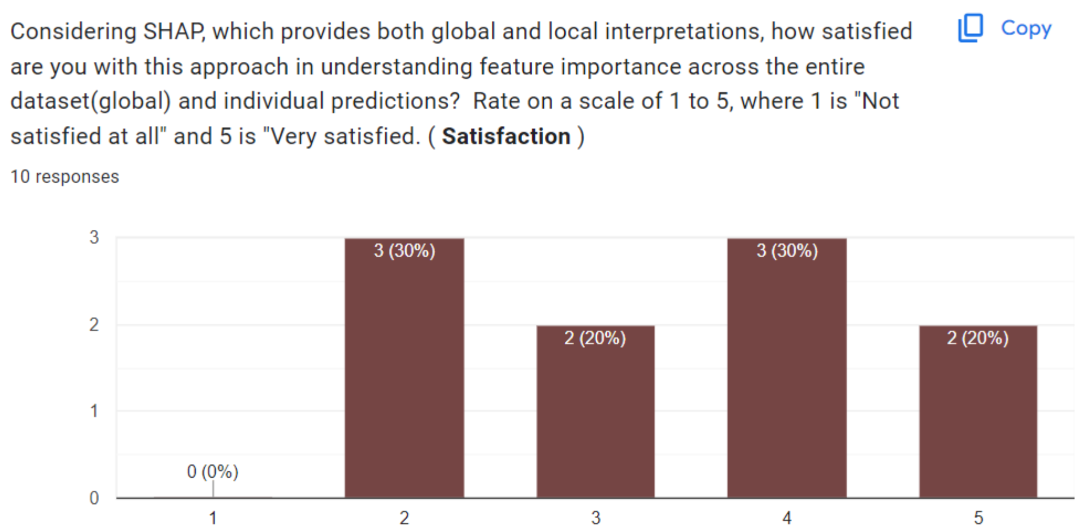


Figure 5.7.9: Satisfaction of SHAP

Figure 5.7.10 depicts the rates that was given for the sufficiency of the SHAP model.

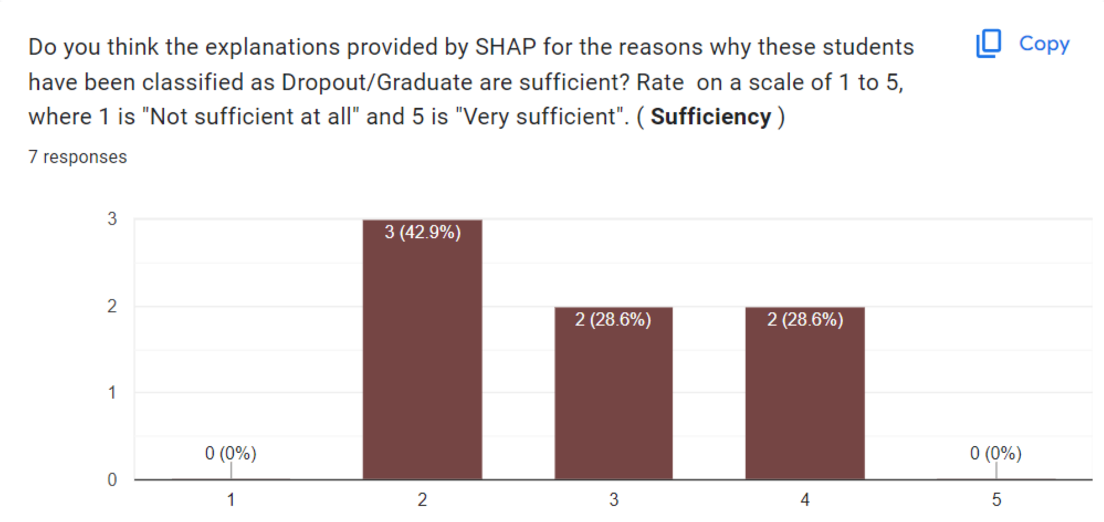


Figure 5.7.10: Sufficiency of SHAP

Figure 5.7.11 depicts the rates that was given for the satisfaction of the SHAP model.

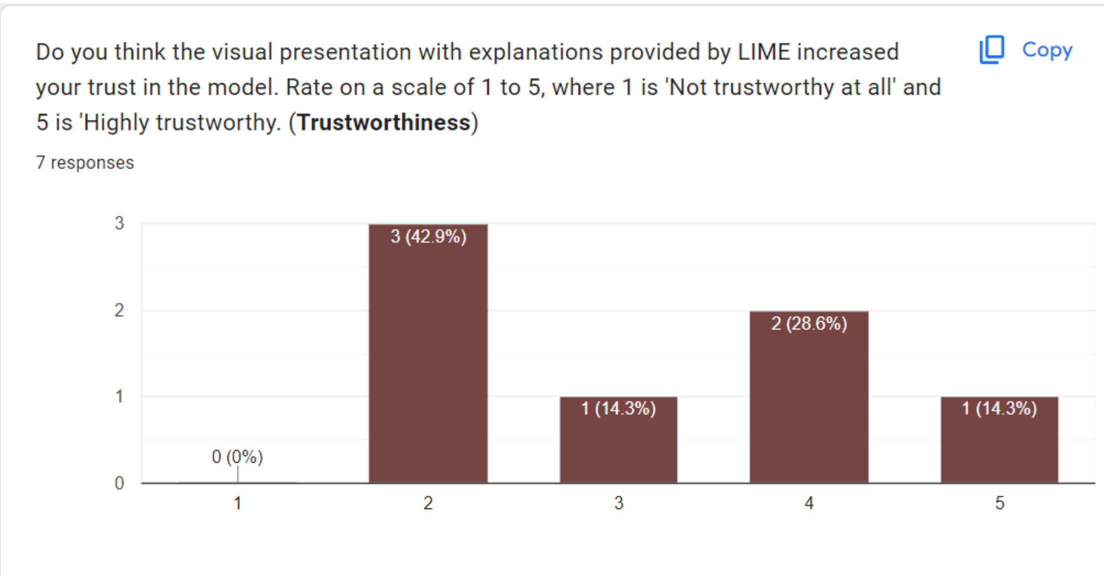


Figure 5.7.11: Trustworthiness of SHAP

The study showed that many of the participants did not know about SHAP before, which means there is a chance to spread the word and educate more people about XAI tools. However, once people got to know it, they had a wide range of opinions about how useful SHAP is for understanding model predictions. Many students were happy with SHAP's interpretive power, as shown by the satisfaction scores.

Some people had different opinions on whether the explanations for classifications were enough. They emphasized the importance of being careful when looking at SHAP's output because student dropouts and graduations are complex and have many factors to consider. Trust in the model, as SHAP's visual explanations have made a lot of people more confident in the model's trustworthiness.

Most tellingly, when asked about the understandability of SHAP's explanations, responses skewed positively, showcasing a majority finding SHAP's output to be clear and easy to understand. This supports our goal of making advanced machine learning models like CatBoost more accessible and interpretable to end-users.

5.7.3 Results of The General Evaluation - User Study 1

After the specific evaluation of LIME and SHAP, the next phrase of the questionnaire was on general evaluation. In that chapter, we evaluated the general interpretability of XAI using LIME and SHAP.

Figure 5.7.12 depicts whether there any preferences by the evaluator for either of the models LIME or SHAP.

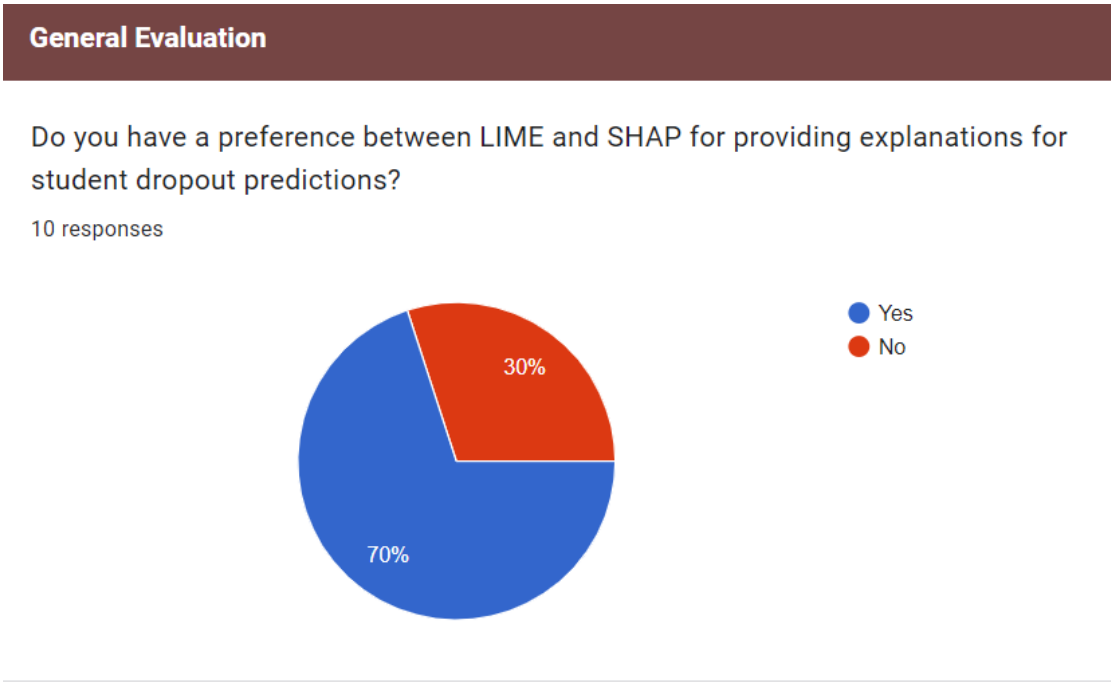
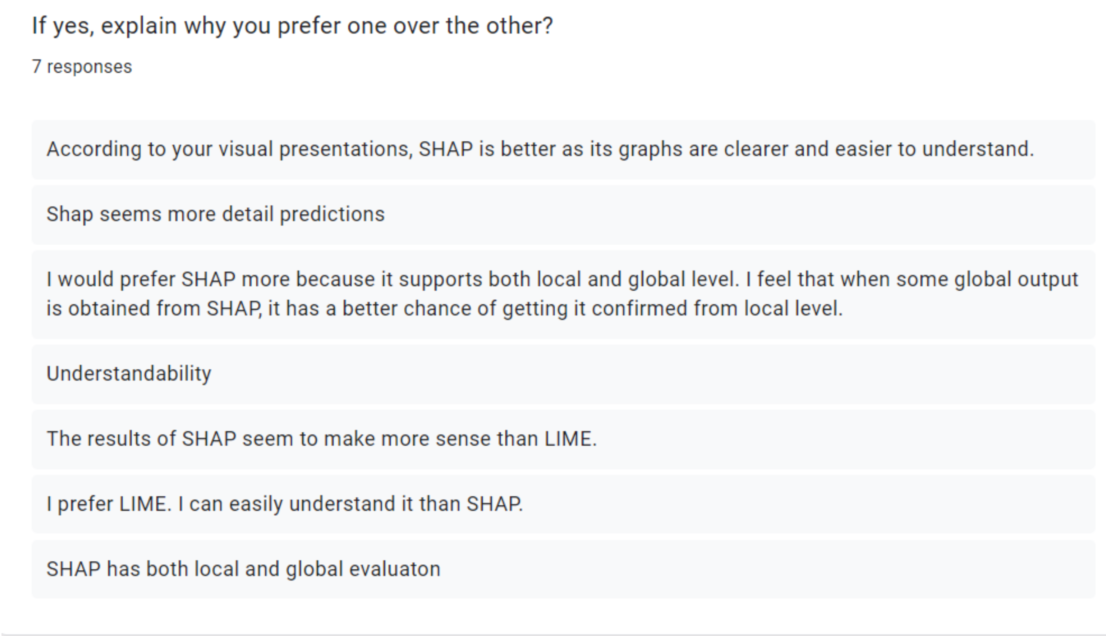


Figure 5.7.12: Preference between the two models

Figure 5.7.13 depicts the reasons for any preferences. Most of them have highlighted understandability as a reason for the preference.


If yes, explain why you prefer one over the other?

7 responses

According to your visual presentations, SHAP is better as its graphs are clearer and easier to understand.

Shap seems more detail predictions

I would prefer SHAP more because it supports both local and global level. I feel that when some global output is obtained from SHAP, it has a better chance of getting it confirmed from local level.

Understandability

The results of SHAP seem to make more sense than LIME.

I prefer LIME. I can easily understand it than SHAP.

SHAP has both local and global evaluaton

Figure 5.7.13: Reasons For The Preferences between the two models

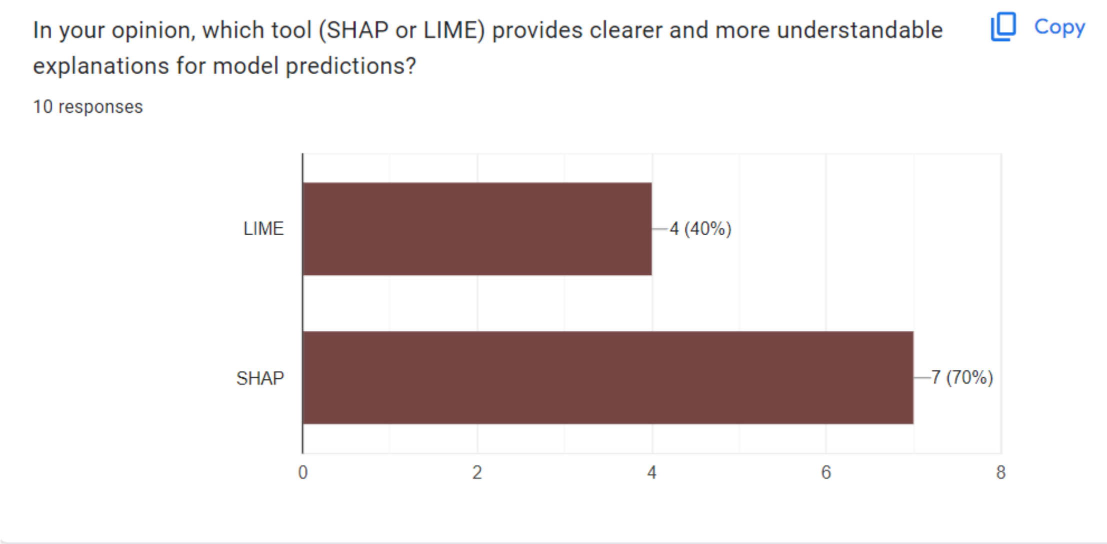


Figure 5.7.14: Preferred model

Here in Figure 5.7.14, it shows the results of how the responders have picked their preferred model. The majority of the people have selected SHAP as their preferred model.

When collecting the responses from a user study it is important to know whether they have the expertise in the field that we are going to take them as evaluators. So in the questionnaire, we checked their experience on identifying or consulting at-risk students. Figure 5.7.15 depicts the outcome which is more positive.

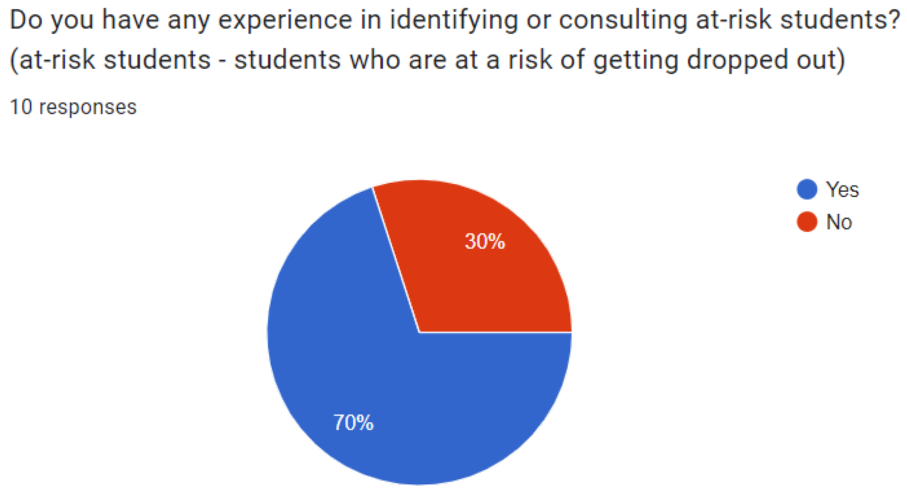


Figure 5.7.15: Experience in identifying or consulting at-risk students

As depicted in Figure 5.7.16, lecturers and counselors have pointed out that it can be really tough to identify and help students who are at risk. It can be difficult to identify the specific reasons that indicate a student is at risk of leaving school. However, the real challenge often comes from personal issues that affect students' attendance and ability to complete assignments. In addition, educators face the challenges of having many students in their classes and the time-consuming task of constantly assessing individuals who may be at risk. It is really hard to communicate with each other, which makes it even more difficult to help students who are struggling academically. We need better ways to quickly identify and support these students. It is really important to have tools and systems that can help identify students who may be struggling so that we can provide them with the right support to improve their education. From these responses we can clearly get an idea of how important is to have a good system to identify at-risk students.

If yes, have you ever face any difficulties when identifying or consulting at-risk students? Briefly explain please.

6 responses

identifying the factors is a challenging task

Not difficult to identify them. But very difficult to make them attend lectures and complete assignments, mainly due to their personal issues.

Yes. It was difficult to verify which students are getting absent for group projects I supervised and group meetings because they are going to dropout or because they really do not care about the projects.

1.Large student count is a difficulty (N= 100 in IS degree and N=230 (approx) in CS degree. 2. Identification if at-risk students in the early stage is difficult.
3. To identify at-risk students, we need to conduct test/assignments/ assessments with students in several time periods (ex: after 3 week). It needs more time and human effort.

difficult to identify features due to communication issues

There is no efficient way of contacting who are going to dropout

Figure 5.7.16: Difficulties in identifying at-risk students

It is important to identify the at-risk students early. And it is much more important to identify the factors for that student to be at that stage. In the general section, we asked a question on the difficulties that the lecturers/counselors might have faced when identifying the factors that affect students to dropout (Figure 5.7.17).

Have you face any difficulties when identifying the factors that contribute/affect students to get dropped out? Explain Briefly.

5 responses

We can identify them easily, but we cannot identify the factors that contribute to/affect them unless they speak or write about their issues to us.

Yes. When a certain students about to drop out was identified from a group project that I supervise, it was difficult to understand the reasons why that happened.

Yes. There are lots parameters. Economic status, living area, family background, mental status, motivation level they all impact to that. Its hard to investigate all these factors.

yes, Sometimes it's related with their psychological issues so it's difficult to identify factors as they may provide false infrmation

No. When I speak with they clearly the reason of dropping out

Figure 5.7.17: General problems in identifying at-risk students

In Figure 5.7.18, it depicts whether the evaluators have used any data observation based techniques in identifying at-risk students. From FIgure 5.7.19 it shows the data observation based techniques currently used by the evaluators.

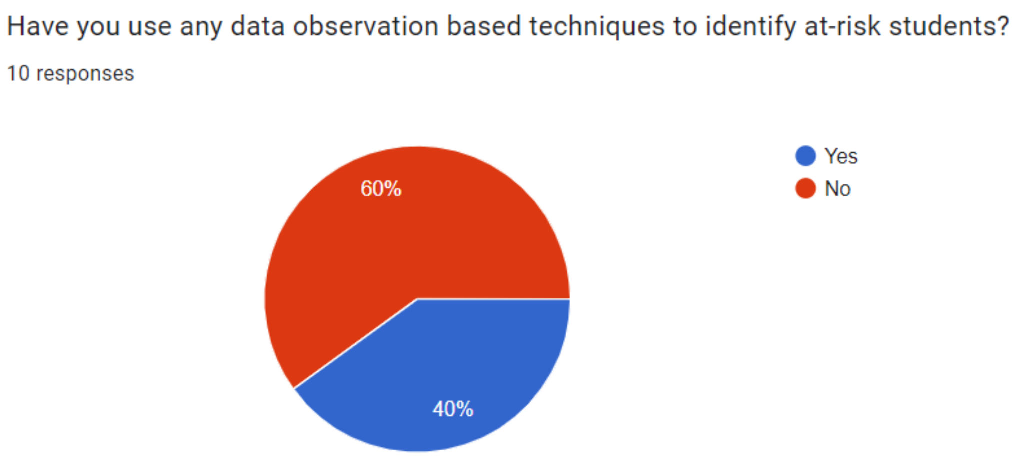
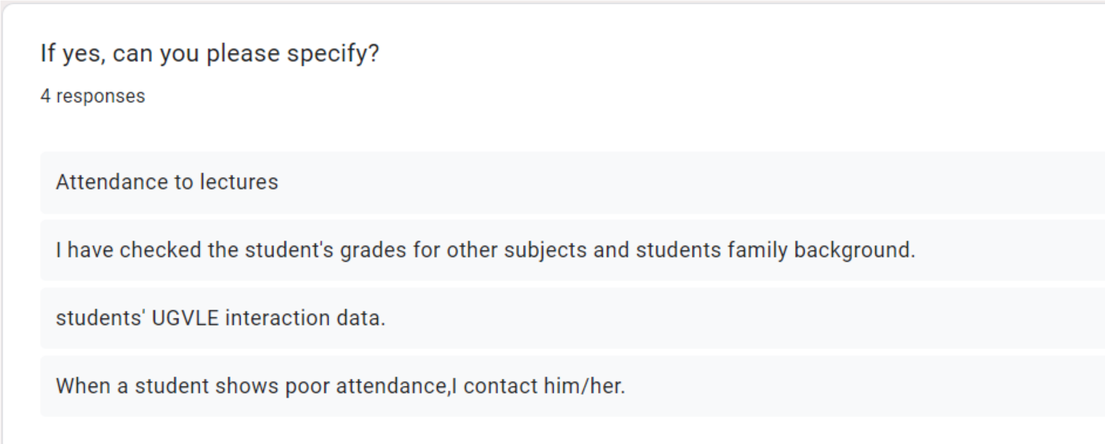


Figure 5.7.18: Usage of data observation-based techniques in identifying at-risk students



If yes, can you please specify?

4 responses

Attendance to lectures

I have checked the student's grades for other subjects and students family background.

students' UGVLE interaction data.

When a student shows poor attendance,I contact him/her.

Figure 5.7.19: Types of data observation-based techniques used in identifying at-risk students

Then we evaluate the overall understandability of the XAI models and their interpretation of the student dropout predictions (Figure 5.7.20).

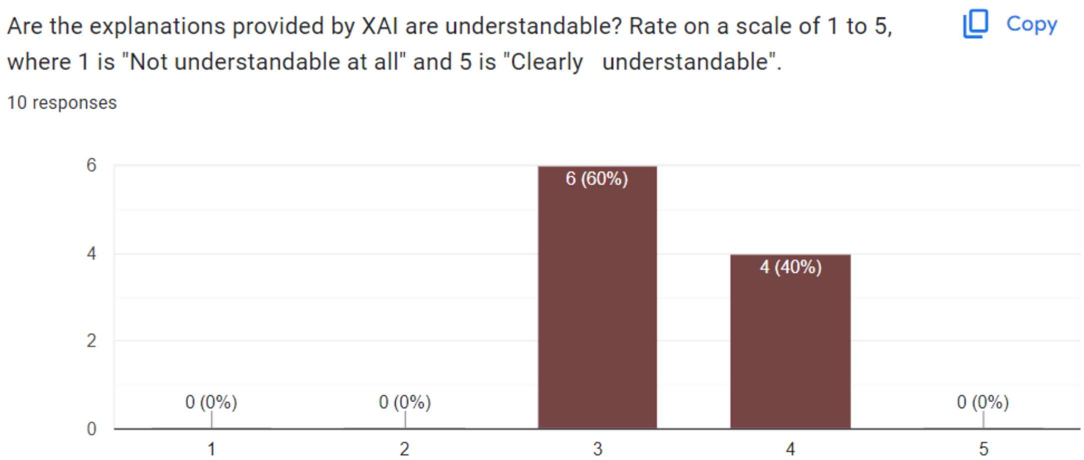


Figure 5.7.20: Overall understandability of the models

Other than evaluating the output in the questionnaire we collected the ideas from the evaluator for any additional feature or improvements that they would like to suggest. They are represented in the Figure 5.7.21.

Are there any additional features or improvements you would like to suggest for enhancing student dropout predictions?

4 responses

no

If you can add features like considering student's physical and mental health issues, family crisis issues and student's level of motivation to do the course it would be good.

You can include a video explanation

better if the model can provide more descriptive factors.

Figure 5.7.21: Feedbacks

5.7.4 Results of The Evaluation of LIME - User Study 2

When it comes to the evaluation of the interpretability of a student dropout prediction system undergraduate students are another major party that we should consider. As undergraduates, students themselves have experiences in filtering out

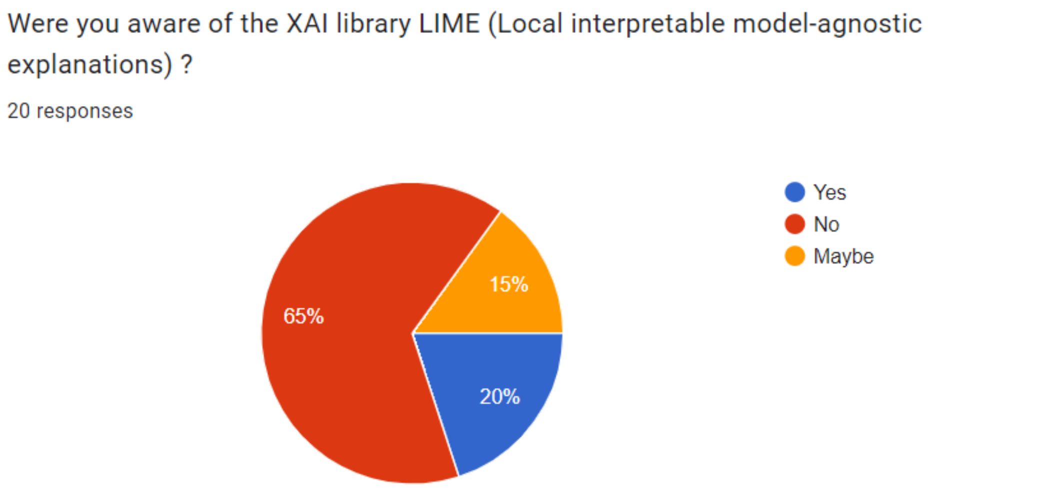


Figure 5.7.22: Awareness of LIME

Figure 5.7.22 shows the response from students on their awareness of the XAI models.

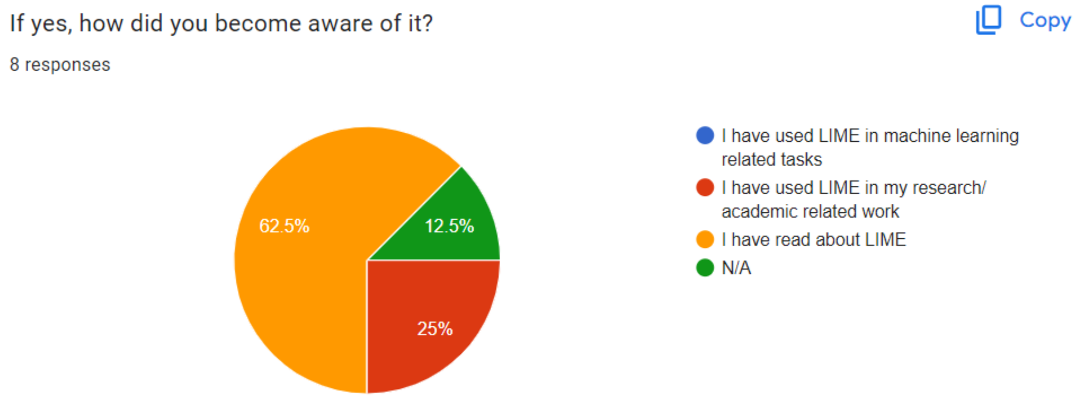


Figure 5.7.23: How did you became aware of LIME

Figure 5.7.22 shows the reasons of on how students become aware of the LIME models. Figure 5.7.24 depicts the rates that was given for the understandability of the LIME model. From Figure 5.7.25 it represents the students' response on the specific comments on the understandability of the LIME model.

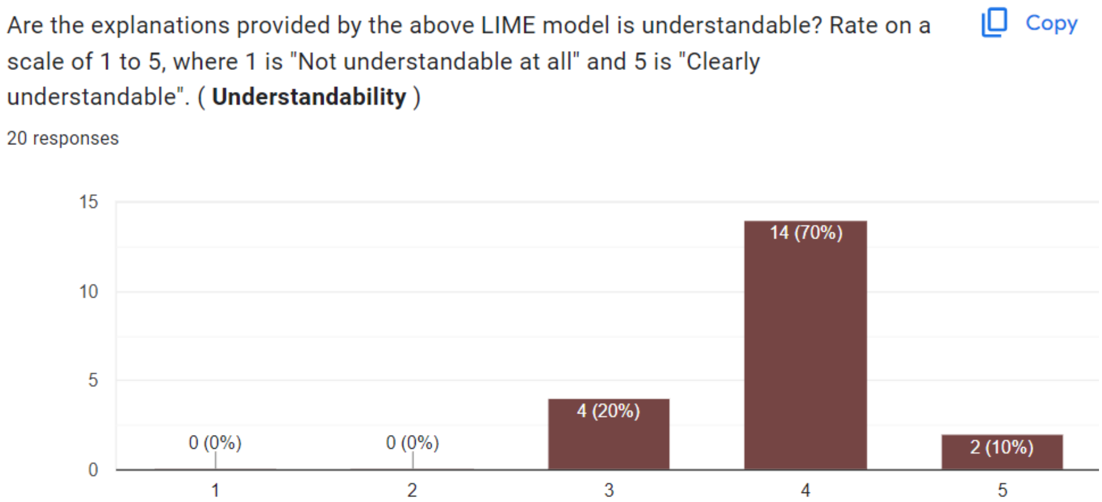


Figure 5.7.24: Understandability of LIME

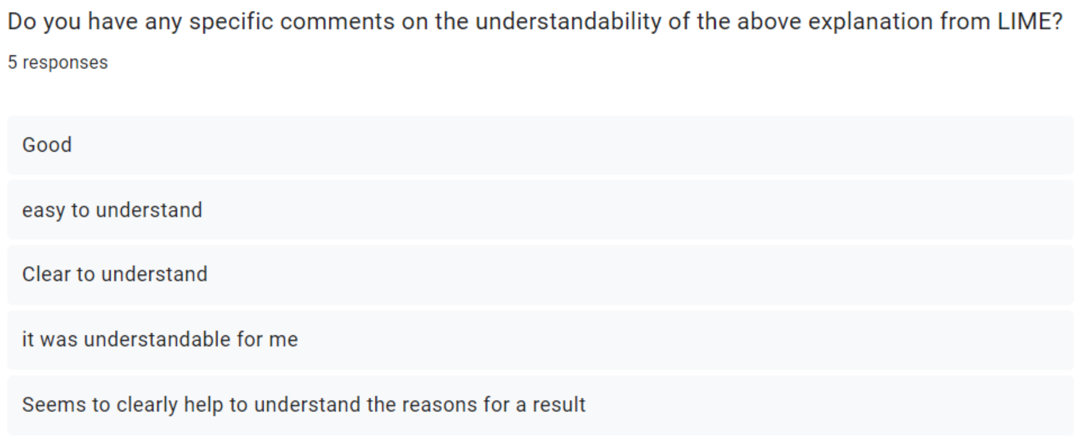


Figure 5.7.25: Explanation on Understandability of LIME

From Figure 5.7.26 it depicts the rates that was given for the satisfaction of the LIME model.

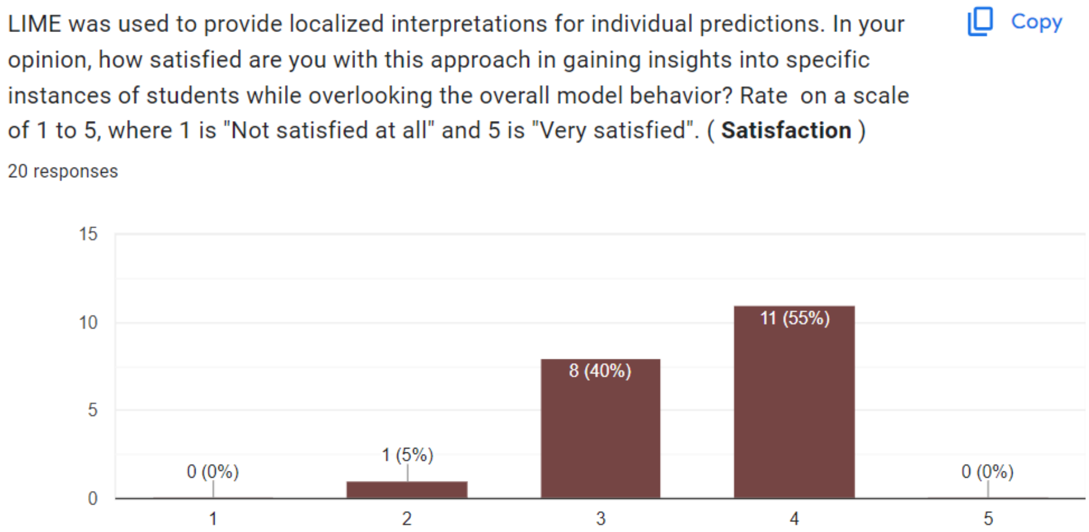


Figure 5.7.26: Satisfaction of LIME

Figure 5.7.27 depicts the rates that was given for the sufficiency of the LIME model.

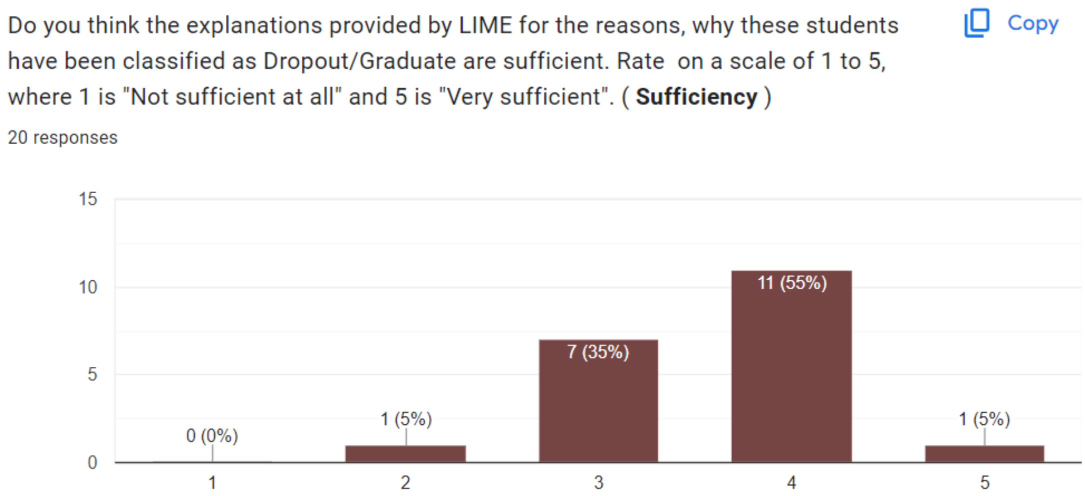


Figure 5.7.27: Sufficiency of LIME

Figure 5.7.28 depicts the rates that was given for the satisfaction of the LIME model.

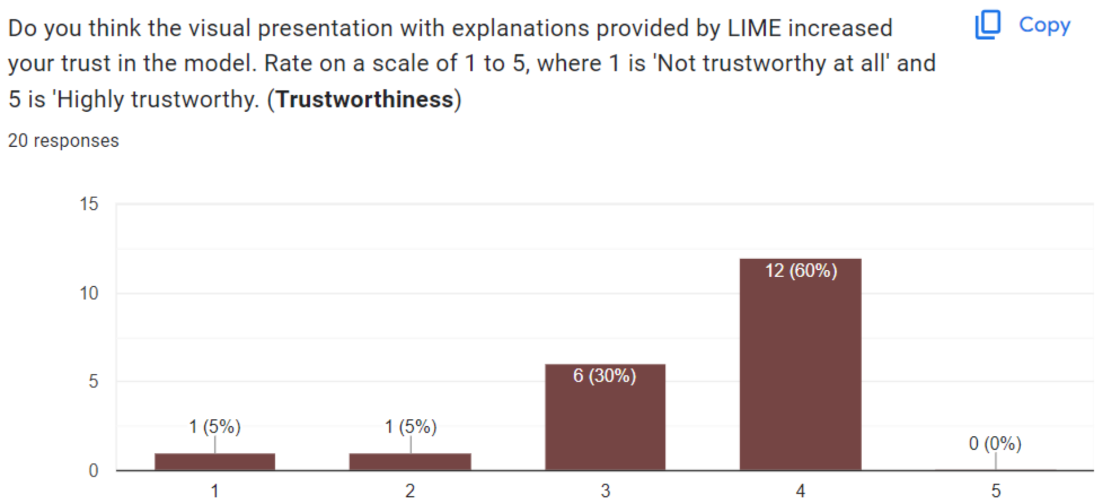


Figure 5.7.28: Trustworthiness of LIME

As mentioned in our second user study, we examined how LIME contributes to improving the interpretability of models through evaluation with a group of undergraduate students. Based on the feedback we received, it seems like more and more students are becoming aware of and understanding the LIME methodology.

This is a positive indication that XAI tools are becoming more widely used in educational settings.

Most students seemed to grasp the explanations provided by LIME, indicating that the tool effectively conveyed the reasoning behind the predictions made by the model. Many university students found LIME to be very helpful. They appreciated how clear the explanations were for each prediction, and they believed that LIME could be a valuable tool in education for making AI decisions easier to understand.

But when it came to whether LIME's explanations were enough to justify the model's classifications of student outcomes, the responses differed. This suggests that there is a need for more detailed or contextually nuanced explanations. People had different opinions about the model's trustworthiness, but overall, the visual aids provided by LIME suggested that it was becoming more trustworthy.

The students' feedback further supported the findings, praising how easy it was to understand and how clear the explanations were. This helped them better understand the reasons behind the model's results.

5.7.5 Results of The Evaluation of SHAP - User Study 2

As in the previous study, after getting evaluated the LIME model questionnaire was based on the SHAP model. Following are the questions and the respective responses for each question based on the SHAP model in the user study 2.

Figure 5.7.29 shows the awareness of the participants on the model SHAP.

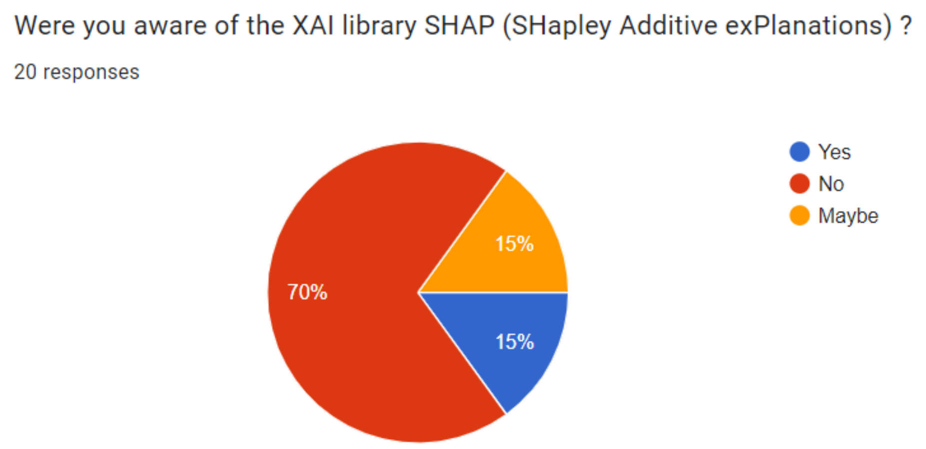


Figure 5.7.29: Awareness on SHAP

Figure 5.7.22 shows the response from students on their awareness of the XAI models.

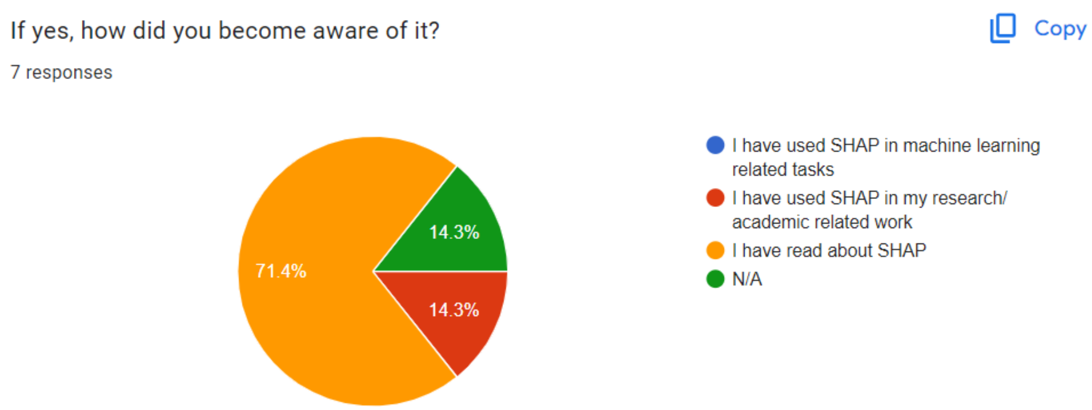


Figure 5.7.30: How did you become aware of SHAP

Figure 5.7.31 depicts the rates that were given for the understandability of the SHAP model.

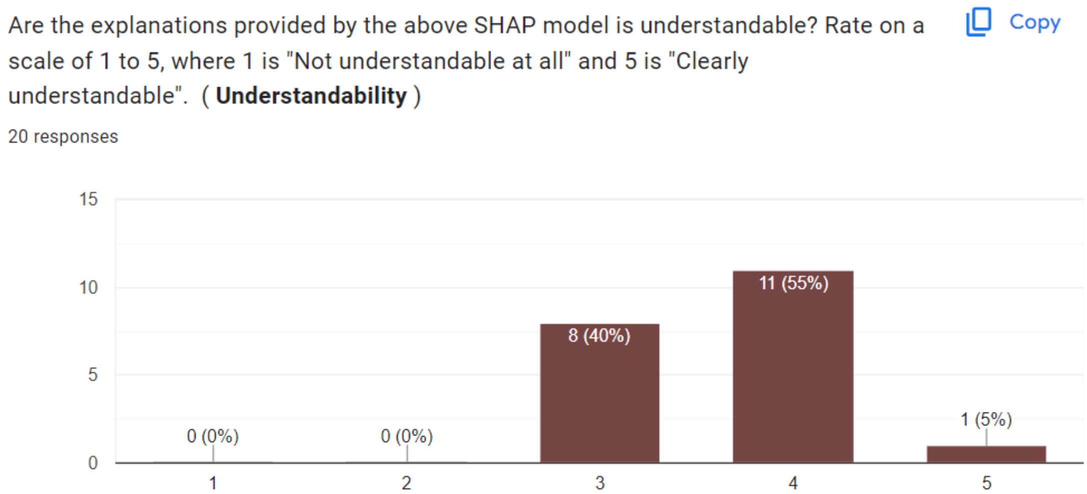


Figure 5.7.31: Understandability of SHAP

Figure 5.7.26 depicts the rates that was given for the satisfaction of the SHAP model.

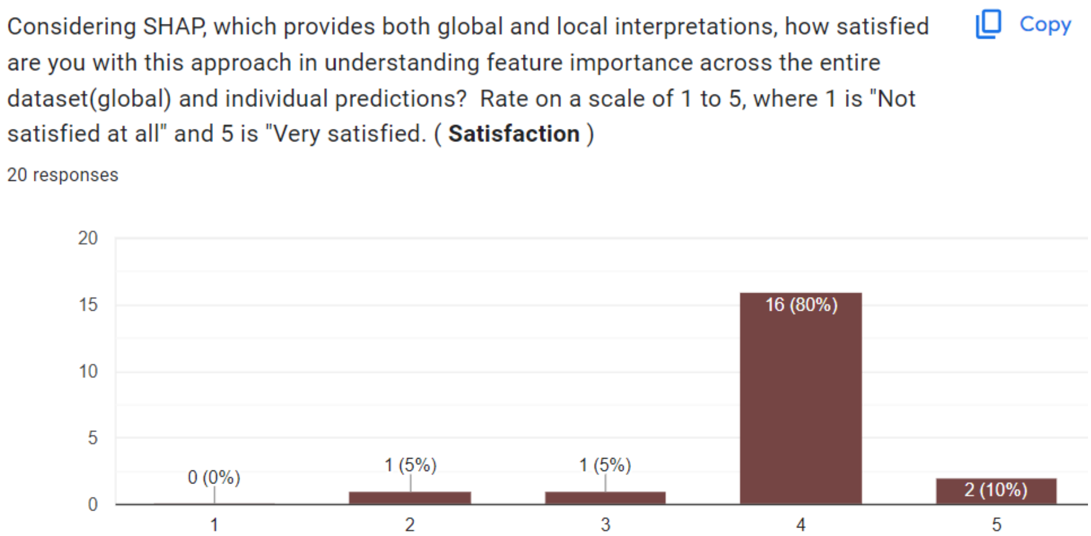


Figure 5.7.32: Satisfaction of SHAP

Figure 5.7.33 depicts the rates that was given for the sufficiency of the SHAP model.

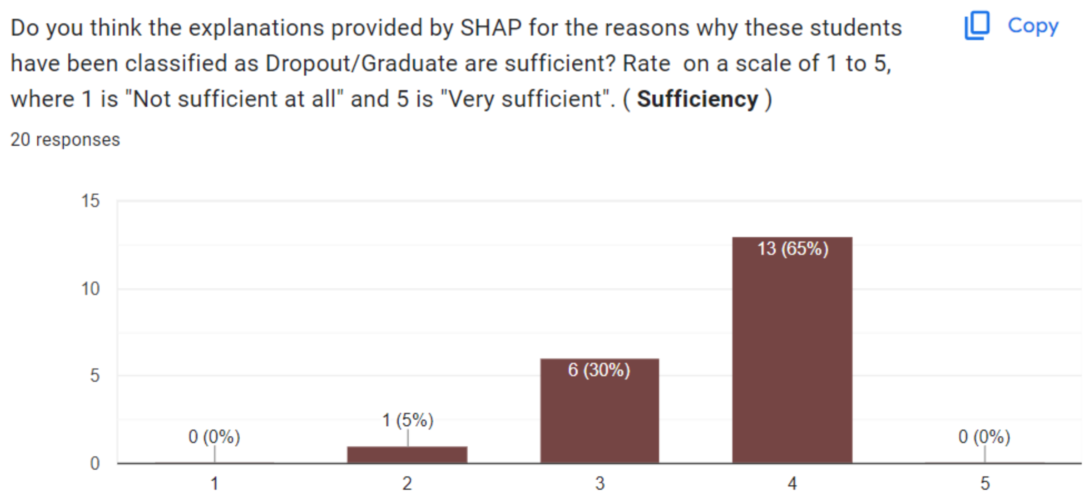


Figure 5.7.33: Sufficiency of SHAP

Figure 5.7.34 depicts the rates that was given for the satisfaction of the SHAP model.

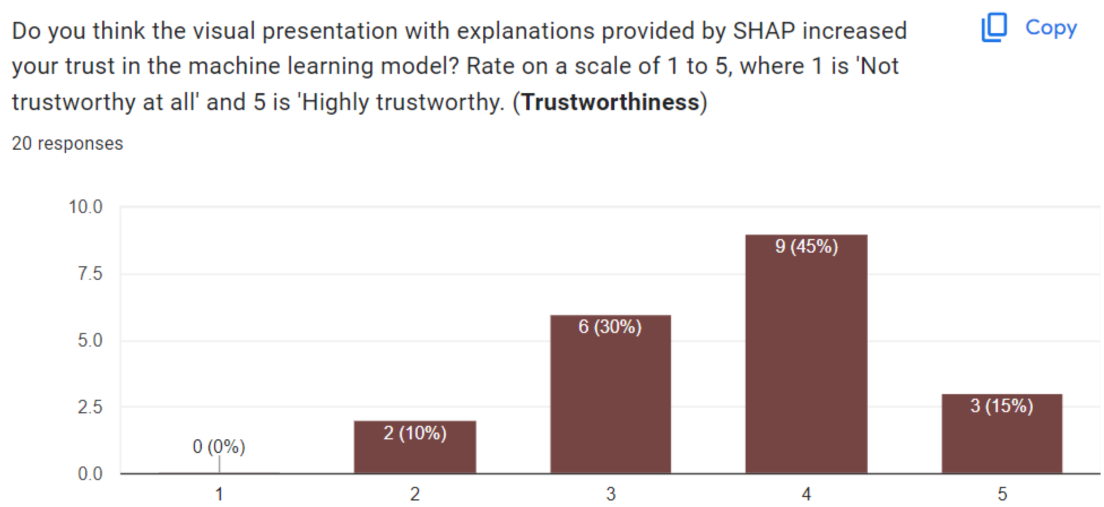


Figure 5.7.34: Trustworthiness of SHAP

Based on the study conducted with university students, it is evident that most of the participants were already familiar with the SHAP library. Many university students have become familiar with SHAP, as it has gained recognition and been integrated into their coursework and research projects.

People really liked how well SHAP explained things on a global and local level. It showed that SHAP was really good at giving detailed insights. Most students were satisfied with SHAP because it helped them understand individual predictions and how different features affect the overall model.

Many university students found the explanations for dropout or graduation classifications in SHAP to be sufficient, but some felt that more detailed explanations could be provided. According to the data university students reported that the visual presentation with explanations from SHAP increased their trust in the machine learning model. This indicates that SHAP's interpretative visualizations were successful in making the model appear more trustworthy.

However, people had different opinions about how easy it was to understand SHAP's explanations. Some people found SHAP to be clear and helpful in understanding the model's output. However, others found it slightly less clear than LIME and a bit more challenging to understand. In general, the students had a positive perception of SHAP. They found its explanations to be detailed and easy to understand, which can help them better understand complex predictive models.

5.7.6 Results of The General Evaluation - User Study 2

In study 2 also, after the specific evaluation of LIME and SHAP, the next phase of the questionnaire was on general evaluation. In that section, we evaluated the general interpretability of XAI using LIME and SHAP.

Figure 5.7.35 depicts whether there are any preferences by the evaluator for either of the models LIME or SHAP.

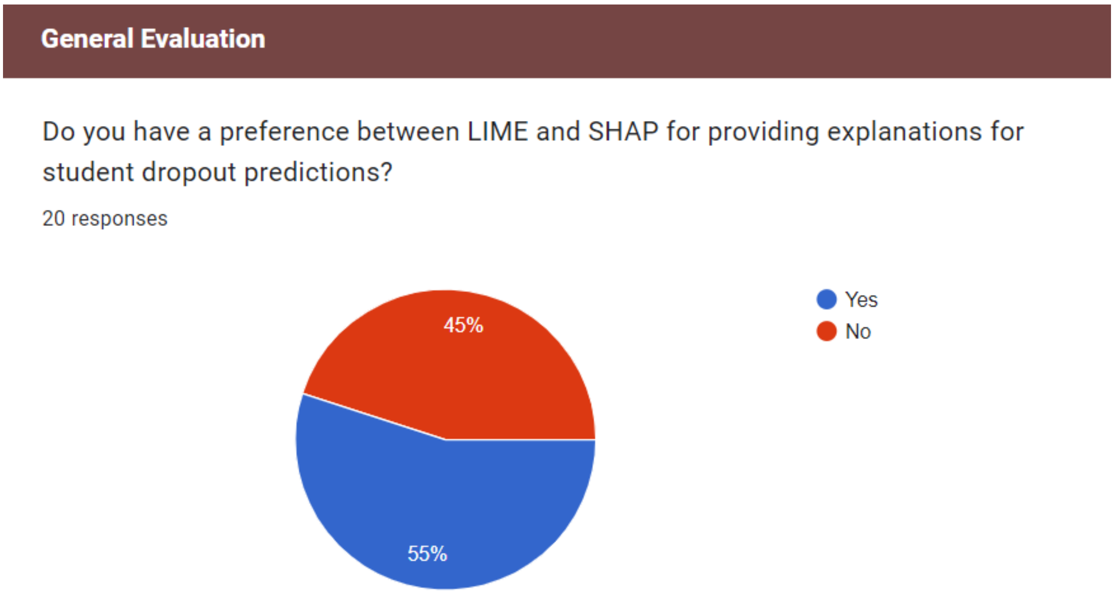
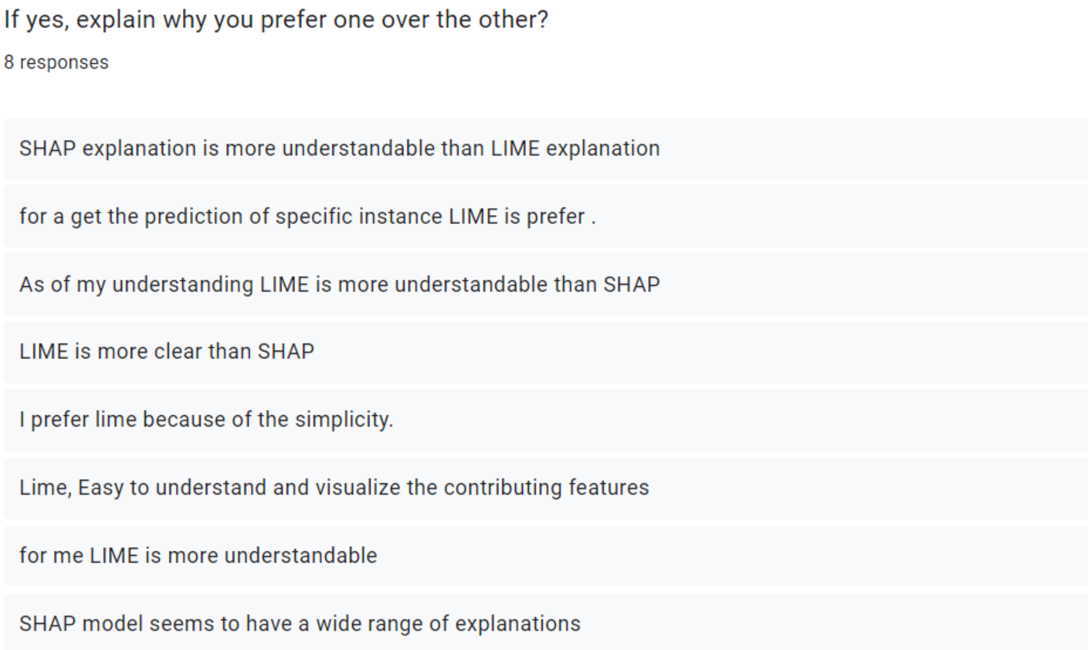


Figure 5.7.35: Preference between the two models

Figure 5.7.36 depicts the reasons for any preferences. Most of them have highlighted understandability as a reason for the preference.


If yes, explain why you prefer one over the other?

8 responses

SHAP explanation is more understandable than LIME explanation

for a get the prediction of specific instance LIME is prefer .

As of my understanding LIME is more understandable than SHAP

LIME is more clear than SHAP

I prefer lime because of the simplicity.

Lime, Easy to understand and visualize the contributing features

for me LIME is more understandable

SHAP model seems to have a wide range of explanations

Figure 5.7.36: Reasons For The Preferences between the two models

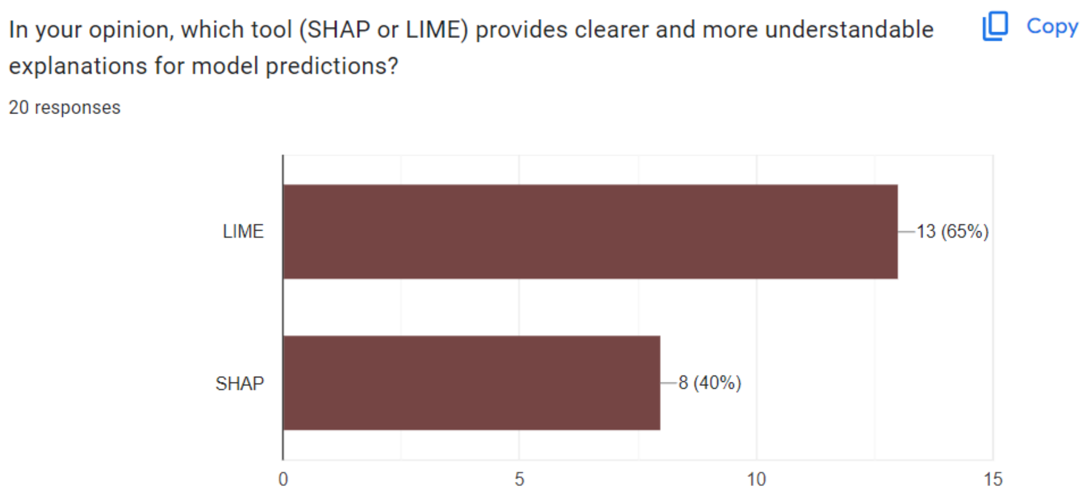


Figure 5.7.37: Preferred model

Here in Figure 5.7.37, it shows the results of how the responders have picked their preferred model. The majority of the people have selected LIME as their preferred model in this study.

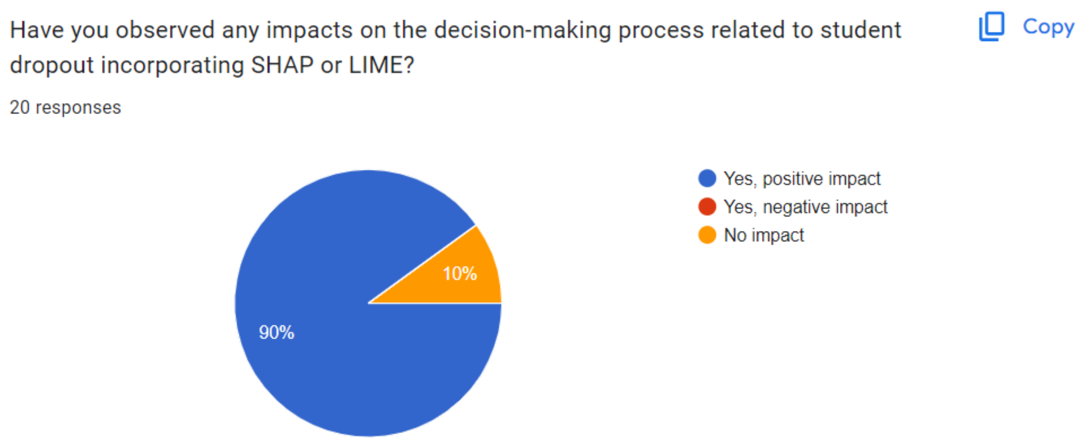


Figure 5.7.38: Impact on the decision making process

In Figure 5.7.41, the highlighted responses from students underscore the impact of integrating LIME and SHAP into the decision-making process regarding student dropout issues. This yields an important and intriguing outcome, demonstrating the capability of employing XAI and interpretative techniques to enhance the well being and improve the quality of higher education.

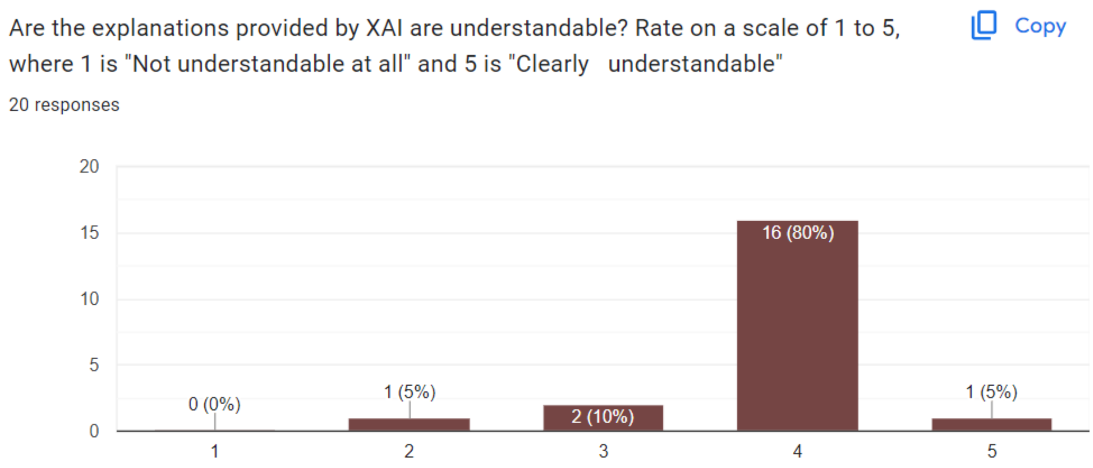


Figure 5.7.39: Overall understandability of XAI techniques

Figure 5.7.39, highlights the overall understandability of XAI models with the students. And in the Figure 5.7.40 it shows that no one have experience in using a dropout prediction system.

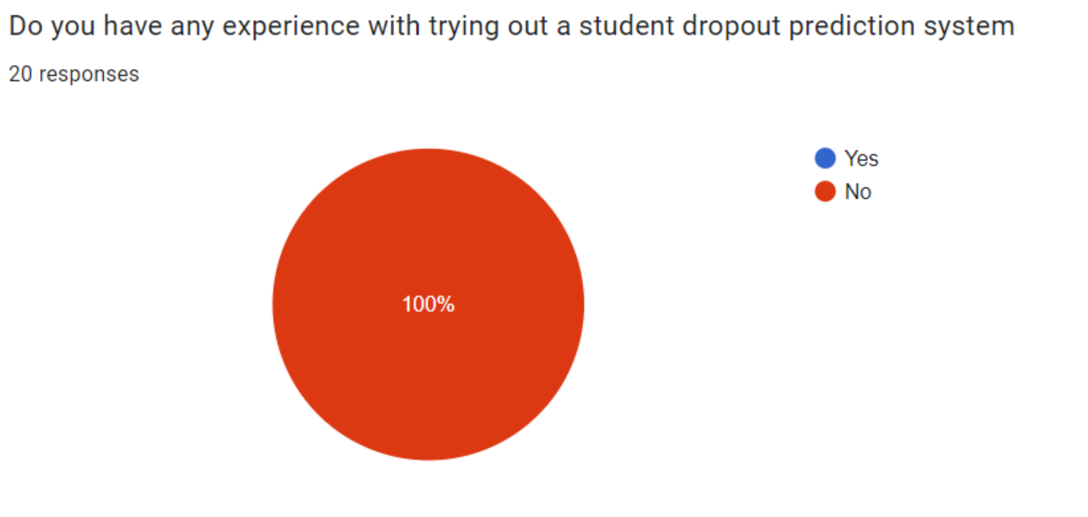


Figure 5.7.40: Experience in using a student dropout prediction system

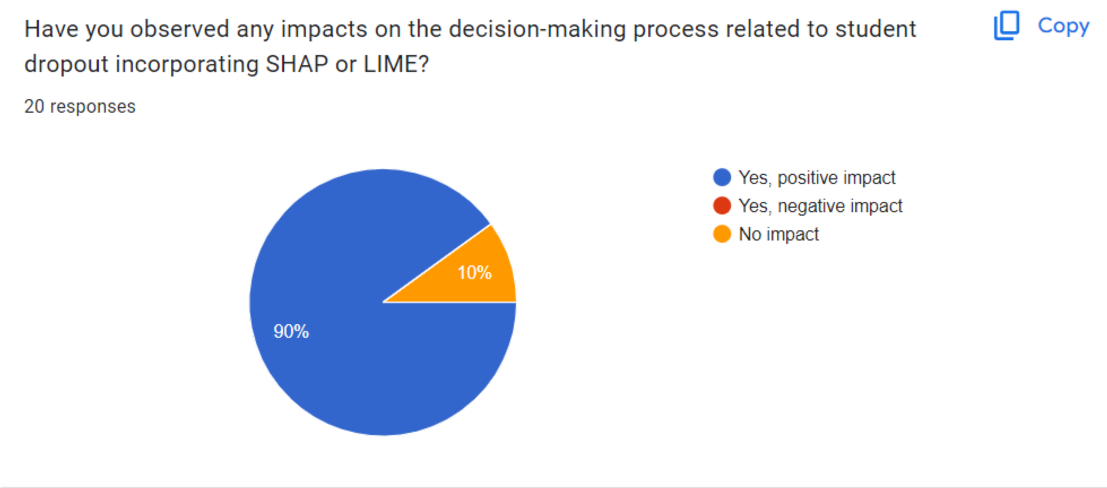


Figure 5.7.41: Impact on the decision making process

As depicted in Figure 5.7.42, most of the students has mentioned that XAI can be found really useful in identifying contributing factors for academic performance as undergraduate students.

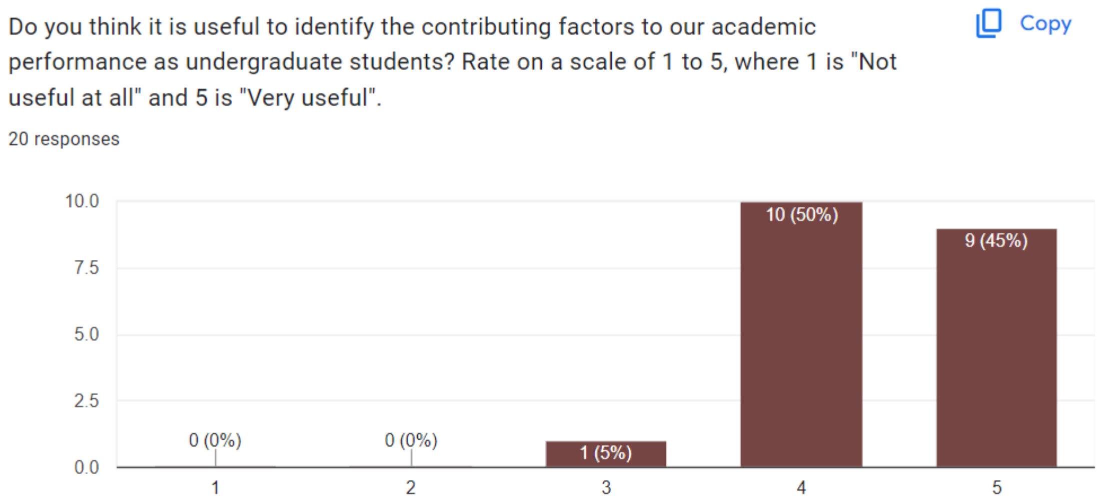


Figure 5.7.42: How useful to identify the factors

5.7.7 Overall Summary of The XAI Evaluation

The findings from both user studies highlight how the most appropriate XAI model for understanding machine learning decisions in the education field depends on the specific context. In the initial study, which included lecturers and counselors who have dealt with student dropout cases, SHAP was preferred because it offers a clear understanding of the data at both a broad and specific level. The model's clear explanations helped experienced educators understand and trust its predictions, which is important for their decision-making.

On the other hand, the second study, which involved a group of students, showed that they preferred LIME. The students preferred LIME's localized explanations because they were easier to understand. This shows that LIME is particularly useful in situations where users need simpler and more direct interpretations. This indicates that LIME's method of providing explanations is effective for audiences who prefer clear and straightforward explanations in order to understand the model's results.

These results show that deciding which XAI tool is better, LIME or SHAP, is not a clear-cut choice. It depends on what the users want and how well they understand XAI concepts. For people like lecturers and counselors who need detailed information to make important decisions, SHAP provides a thorough framework. And it has the capability of interpreting both globally and locally. For university students who prefer a simpler language, LIME offers a more user-friendly option.

Overall, SHAP is known for its in-depth and informative explanations that are ideal for academic experts, while LIME offers easy-to-understand explanations that are better suited for a wider range of people looking for simple interpretations. Choosing the best XAI model depends on who will be using it and how it will be used in the context of higher education.

5.8 Chapter Summary

In this chapter it represents all the results and evaluation details of the study. It has complete comprehensive results and analysis of all the experiments and outcomes of the study. In section 5.1, it represents the results from the data sampling techniques. Section 5.2 represents the results of the hyperparameter tuning process, section 5.3, results of the machine learning models, section 5.4, results of the interpretation models and in section 5.7, results of the XAI model evaluation is represented.

6 Conclusions

6.1 Introduction

This chapter presents the conclusions of the study on undergraduate academic performance prediction while maintaining both accuracy and interpretability. From section 6.2 it will discuss the conclusions related to the research questions, while in section 6.3, we present the conclusions regarding the research problem. In section 6.4, we will discuss the limitations of this study. Then, in section 6.5, we will explore the implications for future research.

6.2 Conclusions about Research Questions

The main aim of our study was to to predict student academic performance while detecting the contributing features leading to their under-performance. The first research question was to find out the most suitable machine learning technique for predicting students' performance. In the preprocessing stage as mentioned in the section 3.1.3, we found that there is a class imbalance problem in the dataset. In addressing the research question regarding the effectiveness of various machine learning models, as depicted in Chapter 5, this thesis demonstrates that models incorporating ADASYN sampling techniques enhance predictive accuracy compared with SMOTE and SVM-SMOTE techniques.

After finding the best sampling technique we did hyperparameter tuning for all mine machine learning and boosting models to find out the most optimal parameters and matrices. According to the results in the sub-section 5.3, we identified the CatBoost model as the best performing model under this data imbalanced environment.

The second research question was how to adapt Explainable Artificial Intelligence (XAI) and Interpretation techniques to identify contributing features for each prediction. We experimented with two XAI models LIME and SHAP. Other than these two models we considered the decision tree interpretation and Explainable Boosting Machine(EBM) interpretation. The interpretation of the decision tree is too complex to understand with the increased number of features. The EBM showed less accuracy compared with the other nine models. Because of those reasons, we did not consider those two interpretation mechanisms for the user study. As discussed in section 4.3, we implemented and experimented with both LIME and SHAP models for our best accurate model.

The third research question was to find out the most effective XAI technique for student dropout prediction models in terms of interpretability. As discussed and concluded in the previous section (section 5.7), we find out SHAP provides in-depth, enlightening explanations with both local and global explanations, while LIME provides straightforward and simple local explanations for a wider audience. From the results of the user study which was done with the participation of lecturers and counselors, the majority identified SHAP as the most preferred model. The reasons for their choice were the clear graphs and visualization of SHAP, SHAP has a more detailed representation and its local and global explanations feature. From the second user study which was done with the participation of university students, their choice was LIME. The undestandability of LIME is higher than SHAP, for a local explanation LIME gives a better explanation and its simplicity was the reasons for their choice.

From the results of the study, it was observable that both LIME and SHAP models were able to give a good interpretable and accurate interpretation compared to the decision tree's interpretation and EBM's interpretation. Moreover from these results it was observable that rather than coming to a conclusion of selecting one single XAI model as the best explainable model, choosing the optimal XAI model depends on who will use it and what is the context it will use in higher education.

6.3 Conclusions about Research Problem

This study was focused on finding a method for undergraduate academic performance prediction while maintaining both accuracy and interpretability. In order to achieve this objective, nine different machine learning and boosting models were experimented with. Other than these models Explainable Boosting Machine(EBM) which is considered as an interpretable machine learning model was also included into the study. Out of the models CatBoost performed the best with a training F1-Score of 0.8378 and testing F1-Score of 0.8603. These results were obtained after a comprehensive parameter tuning process for all the models using grid search. Then the best model was used to get the interpretation from the XAI models, LIME and SHAP. Other than the interpretation of these models decision trees interpretation and EBM interpretation was also experimented in the study. But EBM interpretation was not considered for the interpretation evaluation because of the low accuracy values obtained compared with the other models. Interpretation of the decision tree was not considered because of the high complexity of the outcome. From the user studies and results which was done considering the LIME and SHAP models it indicated that the selection of an optimal XAI (Explainable Artificial Intelligence) model does not rest upon determining a single 'best' model. Instead, the choice of the most suitable XAI model hinges on the intended user and the specific context within which it will be employed in higher education.

In conclusion, from this study we were able to find out how different data sampling techniques will affect an outcome and how we can apply different XAI techniques to get interpretations according to the context. Thus, this study contributed to both machine learning and interpretable machine learning domains by the findings of the study.

6.4 Limitations

The study was done using a dataset of a Portugal university. Therefore the trained model will not be the optimal solution model to predict the dropout of students of the other universities.

As the dataset contained features related to that country and the university, the models were trained on only those features. Thus, the explanations was provided considering only the features that was available in the dataset.

6.5 Implications for further research

Our model is effective and practical, making it a good starting point for future research. One can further explore how the model can be adapted by using different sets of data. This method will not only improve how it works, but also test how well it can handle different situations.

In addition, studying interpretive models like Explainable Boosting Machines (EBM) can be a useful way to make progress in XAI. Improvements in the accuracy of EBM could make it easier to use as a tool for interpreting information, which would help make complex models more transparent for decision-making.

In the future, researchers could try using different machine learning algorithms and explore various methods of explainable artificial intelligence. Our goal is to improve the performance and understanding of the models. This comprehensive approach has the potential to create more detailed and user-focused XAI applications that meet the changing needs of different people in the artificial intelligence and education field.

References

Adnan, Muhammad et al. (2021). "Predicting at-risk students at different percentages of course length for early intervention using machine learning models". In: *Ieee Access* 9, pp. 7519–7539.

Aulck, Lovenoor et al. (2016). "Predicting student dropout in higher education". In: *arXiv preprint arXiv:1606.06364*.

Baars, Gerard JA, Theo Stijnen, and Ted AW Splinter (2017). "A model to predict student failure in the first year of the undergraduate medical curriculum". In: *Health Professions Education* 3.1, pp. 5–14.

Bainbridge, Jay et al. (2015). "Using learning analytics to predict at-risk students in online graduate public affairs and administration education". In: *Journal of Public Affairs Education* 21.2, pp. 247–262.

Balbin, Arnel (2019). *Influence of non-academic factors on the academic performance among education students.* URL: https://journalcra.com/article/influence-non-academic-factors-academic-performance-among-education-students.

Bordia, Ansh (Mar. 2024). *Handling imbalanced data by oversampling with smote and its variants.* URL: https://medium.com/analytics-vidhya/handling-imbalanced-data-by-oversampling-with-smote-and-its-variants-23a4bf188eaf.

Brandt, Jakob and Emil Lanzén (2021). "A comparative review of SMOTE and ADASYN in imbalanced data classification". In.

Chawla, Nitesh V et al. (2002). "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16, pp. 321–357.

Chen, Tianqi and Carlos Guestrin (2016). "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794.

Chui, Kwok Tai et al. (2020). "Predicting at-risk university students in a virtual learning environment via a machine learning algorithm". In: *Computers in Human behavior* 107, p. 105584.

Ha, Dinh Thi et al. (2020). "An empirical study for student academic performance prediction using machine learning techniques". In: *International Journal of Computer Science and Information Security (IJCSIS)* 18.3, pp. 75–82.

Hamelers, LH (2021). "Detecting and explaining potential financial fraud cases in invoice data with Machine Learning". MA thesis. University of Twente.

Hanson, Melanie (June 2022). *College Dropout Rates*. URL: https://educationdata.org/college-dropout-rates.

He, Haibo et al. (2008). "ADASYN: Adaptive synthetic sampling approach for imbalanced learning". In: *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. Ieee, pp. 1322–1328.

Hoffman, Robert R et al. (2018). "Metrics for explainable AI: Challenges and prospects". In: *arXiv preprint arXiv:1812.04608*.

Hossain, Md Riyad and Douglas Timmer (2021). "Machine learning model optimization with hyper parameter tuning approach". In: *Global Journal of Computer Science and Technology* 21.D2, pp. 7–13.

IBM (2023). URL: https://www.ibm.com/watson/explainable-ai.

Jayasundara, Shyaman, Amila Indika, and Damayanthi Herath (2022). "Interpretable Student Performance Prediction Using Explainable Boosting Machine for Multi-Class Classification". In: *2022 2nd International Conference on Advanced Research in Computing (ICARC)*. IEEE, pp. 391–396.

Ji, Yingchao (2021). *Explainable AI methods for credit card fraud detection: Evaluation of LIME and SHAP through a User Study*.

Kuzilek, Jakub et al. (2015). "OU Analyse: analysing at-risk students at The Open University". In: *Learning Analytics Review*, pp. 1–16.

Latif, A, AI Choudhary, and AA Hammayun (2015). "Economic effects of student dropouts: A comparative study". In: *Journal of global economics* 3.2, pp. 1–4.

Lee, Sunbok and Jae Young Chung (2019). "The machine learning-based dropout early warning system for improving the performance of dropout prediction". In: *Applied Sciences* 9.15, p. 3093.

Lundberg, Scott M and Su-In Lee (2017). "A unified approach to interpreting model predictions". In: *Advances in neural information processing systems* 30.

Lykourentzou, Ioanna et al. (2009). "Dropout prediction in e-learning courses through the combination of machine learning techniques". In: *Computers & Education* 53.3, pp. 950–965.

Mahboob, Khalid, Raheela Asif, and Najmi Ghani Haider (2023). "Quality enhancement at higher education institutions by early identifying students at risk using data mining". In: *Mehran University Research Journal of Engineering and Technology* 42.1, pp. 120–136.

Martins, Mónica V et al. (2021). "Early Prediction of Student's Performance in Higher Education: A Case Study". In: *Trends and Applications in Information Systems and Technologies: Volume 1 9*. Springer, pp. 166–175.

Marwaha, Anu and Anshu Singla (2020). "A study of factors to predict at-risk students based on machine learning techniques". In: *Intelligent Communication, Control and Devices: Proceedings of ICICCD 2018*. Springer, pp. 133–141.

Mbouzao, Boniface, Michel C Desmarais, and Ian Shrier (2020). "Early prediction of success in MOOC from video interaction features". In: *International Conference on Artificial Intelligence in Education*. Springer, pp. 191–196.

Mduma, Neema, Khamisi Kalegele, and Dina Machuve (2019). "A survey of machine learning approaches and techniques for student dropout prediction". In.

Navaz, Juan (Feb. 2022). *What is hyperparameter tuning?* URL: `https://www.anyscale.com/blog/what-is-hyperparameter-tuning`.

Nian, Rui (Dec. 2019). *An introduction to ADASYN (with code!)* URL: https://medium.com/@ruinian/an-introduction-to-adasyn-with-code-1383a5ece7aa.

Pillay, Hitendra et al. (2017). *Innovative strategies for accelerated human resource development in South Asia: Teacher professional development-Special focus on Bangladesh, Nepal, and Sri Lanka*. Asian Development Bank.

Pradipta, Gede Angga et al. (2021). "SMOTE for handling imbalanced data problem: A review". In: *2021 sixth international conference on informatics and computing (ICIC)*. IEEE, pp. 1–8.

Prokhorenkova, Liudmila et al. (2018). "CatBoost: unbiased boosting with categorical features". In: *Advances in neural information processing systems* 31.

Reyes, Arnoldo (Aug. 2023). *What is the average age of college students?* URL: https://www.collegeranker.com/what-is-the-average-age-of-college-students/.

Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin (2016). "" Why should i trust you?" Explaining the predictions of any classifier". In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144.

Al-Shabandar, Raghad et al. (2019). "Detecting at-risk students with early interventions using machine learning techniques". In: *IEEE Access* 7, pp. 149464–149478.

Tanha, Jafar et al. (2020). "Boosting methods for multi-class imbalanced data classification: an experimental review". In: *Journal of Big Data* 7, pp. 1–47.

Turri, Violet (Jan. 2022). *What is explainable ai?* URL: https://insights.sei.cmu.edu/blog/what-is-explainable-ai/.

You, Ji Won (2016). "Identifying significant indicators using LMS data to predict course achievement in online learning". In: *The Internet and Higher Education* 29, pp. 23–30.

Yukseltruk, Erman and Fethi Ahmet Inan (2006). “Examining the factors affecting student dropout in an online learning environment”. In: *ASHE-ERIC Higher Education Report (ERIC No. ED 494 345).*

Zaki, Amira et al. (2023). “5G and Beyond: Channel Classification Enhancement Using VIF-Driven Preprocessing and Machine Learning”. In: *Electronics* 12.16, p. 3496.

7 Appendix A: Results of Experiments

An appendix of all the results of the study will be included in this section.

Results presented in the upcoming figures follows the notations defined below.

7.1 Results of The Machine Learning Models

Following are the results from SMOTE sampling

```
SMOTE LR Results
Training Accuracy: 0.7907
Training F1 Score: 0.7979
Training ROC AUC Score: 0.8712
Validation Accuracy: 0.7534
Validation F1 Score: 0.8031
Validation ROC AUC Score: 0.8141
```

Figure 7.1.1: SMOTE Logistic Regression Results

```
SMOTE DT Results
Training Accuracy: 0.7797
Training F1 Score: 0.7777
Training ROC AUC Score: 0.8579
Validation Accuracy: 0.7617
Validation F1 Score: 0.7995
Validation ROC AUC Score: 0.8244
```

Figure 7.1.2: SMOTE Decision Tree Results

```
Random Forest Training Accuracy: 0.8337
Random Forest Training F1 Score: 0.8424
Random Forest Training ROC AUC Score: 0.9132
Random Forest Validation Accuracy: 0.7920
Random Forest Validation F1 Score: 0.8360
Random Forest Validation ROC AUC Score: 0.8574
```

Figure 7.1.3: SMOTE Random Forest Results

```
SVM Training Accuracy: 0.7390
SVM Training F1 Score: 0.7536
SVM Training ROC AUC Score: 0.8319
SVM Validation Accuracy: 0.7080
SVM Validation F1 Score: 0.7644
SVM Validation ROC AUC Score: 0.7798
```

Figure 7.1.4: SMOTE SVM Results

```
Naive Bayes Training Accuracy: 0.7481
Naive Bayes Training F1 Score: 0.7706
Naive Bayes Training ROC AUC Score: 0.8432
Naive Bayes Validation Accuracy: 0.7383
Naive Bayes Validation F1 Score: 0.8013
Naive Bayes Validation ROC AUC Score: 0.7885
```

Figure 7.1.5: SMOTE Naive Bayes Results

```
Gradient Boosting Training Accuracy: 0.7957
Gradient Boosting Training F1 Score: 0.8120
Gradient Boosting Training ROC AUC Score: 0.8730
Gradient Boosting Validation Accuracy: 0.7879
Gradient Boosting Validation F1 Score: 0.8358
Gradient Boosting Validation ROC AUC Score: 0.8327
```

Figure 7.1.6: SMOTE Gradient Boost Results

```
CatBoost Training Accuracy: 0.8390
CatBoost Training F1 Score: 0.8450
CatBoost Training ROC AUC Score: 0.9156
CatBoost Validation Accuracy: 0.8196
CatBoost Validation F1 Score: 0.8475
CatBoost Validation ROC AUC Score: 0.8756
```

Figure 7.1.7: SMOTE CatBoost Results

```
AdaBoost Training Accuracy: 0.8250
AdaBoost Training F1 Score: 0.8331
AdaBoost Training ROC AUC Score: 0.9051
AdaBoost Validation Accuracy: 0.8099
AdaBoost Validation F1 Score: 0.8410
AdaBoost Validation ROC AUC Score: 0.8663
```

Figure 7.1.8: SMOTE AdaBoost Results

Following are the results from SVM-SMOTE sampling

```
SVM-SMOTE LR Results
Training Accuracy: 0.7861
Training F1 Score: 0.7932
Training ROC AUC Score: 0.8691
Validation Accuracy: 0.7507
Validation F1 Score: 0.8004
Validation ROC AUC Score: 0.8186
```

Figure 7.1.9: SVM-SMOTE Logistic Regression Results

```
SVM-SMOTE DT Results
Training Accuracy: 0.7801
Training F1 Score: 0.7789
Training ROC AUC Score: 0.8555
Validation Accuracy: 0.7617
Validation F1 Score: 0.7991
Validation ROC AUC Score: 0.8287
```

Figure 7.1.10: SVM-SMOTE Decision Tree Results

```
Random Forest Training Accuracy: 0.8265
Random Forest Training F1 Score: 0.8327
Random Forest Training ROC AUC Score: 0.9158
Random Forest Validation Accuracy: 0.8030
Random Forest Validation F1 Score: 0.8434
Random Forest Validation ROC AUC Score: 0.8599
```

Figure 7.1.11: SVM-SMOTE Random Forest Results

```
SVM Training Accuracy: 0.7565
SVM Training F1 Score: 0.7614
SVM Training ROC AUC Score: 0.8343
SVM Validation Accuracy: 0.7163
SVM Validation F1 Score: 0.7675
SVM Validation ROC AUC Score: 0.7753
```

Figure 7.1.12: SVM-SMOTE SVM Results

```
Naive Bayes Training Accuracy: 0.7333
Naive Bayes Training F1 Score: 0.7612
Naive Bayes Training ROC AUC Score: 0.8371
Naive Bayes Validation Accuracy: 0.7328
Naive Bayes Validation F1 Score: 0.7979
Naive Bayes Validation ROC AUC Score: 0.7936
```

Figure 7.1.13: SVM-SMOTE Naive Bayes Results

```
Gradient Boosting Training Accuracy: 0.7907
Gradient Boosting Training F1 Score: 0.8072
Gradient Boosting Training ROC AUC Score: 0.8683
Gradient Boosting Validation Accuracy: 0.7948
Gradient Boosting Validation F1 Score: 0.8403
Gradient Boosting Validation ROC AUC Score: 0.8307
```

Figure 7.1.14: SVM-SMOTE Gradient Boost Results

```
CatBoost Training Accuracy: 0.8390
CatBoost Training F1 Score: 0.8448
CatBoost Training ROC AUC Score: 0.9176
CatBoost Validation Accuracy: 0.8209
CatBoost Validation F1 Score: 0.8475
CatBoost Validation ROC AUC Score: 0.8763
```

Figure 7.1.15: SMOTE CatBoost Results

```
AdaBoost Training Accuracy: 0.8238
AdaBoost Training F1 Score: 0.8297
AdaBoost Training ROC AUC Score: 0.9023
AdaBoost Validation Accuracy: 0.8127
AdaBoost Validation F1 Score: 0.8409
AdaBoost Validation ROC AUC Score: 0.8693
```

Figure 7.1.16: SVM-SMOTE AdaBoost Results

Following are the results from SVM-SMOTE sampling

```
SVM-SMOTE LR Results
Training Accuracy: 0.7861
Training F1 Score: 0.7932
Training ROC AUC Score: 0.8691
Validation Accuracy: 0.7507
Validation F1 Score: 0.8004
Validation ROC AUC Score: 0.8186
```

Figure 7.1.17: SVM-SMOTE Logistic Regression Results

```
SVM-SMOTE DT Results
Training Accuracy: 0.7801
Training F1 Score: 0.7789
Training ROC AUC Score: 0.8555
Validation Accuracy: 0.7617
Validation F1 Score: 0.7991
Validation ROC AUC Score: 0.8287
```

Figure 7.1.18: SVM-SMOTE Decision Tree Results

```
Random Forest Training Accuracy: 0.8265
Random Forest Training F1 Score: 0.8327
Random Forest Training ROC AUC Score: 0.9158
Random Forest Validation Accuracy: 0.8030
Random Forest Validation F1 Score: 0.8434
Random Forest Validation ROC AUC Score: 0.8599
```

Figure 7.1.19: SVM-SMOTE Random Forest Results

```
SVM Training Accuracy: 0.7565
SVM Training F1 Score: 0.7614
SVM Training ROC AUC Score: 0.8343
SVM Validation Accuracy: 0.7163
SVM Validation F1 Score: 0.7675
SVM Validation ROC AUC Score: 0.7753
```

Figure 7.1.20: SVM-SMOTE SVM Results

```
Naive Bayes Training Accuracy: 0.7333
Naive Bayes Training F1 Score: 0.7612
Naive Bayes Training ROC AUC Score: 0.8371
Naive Bayes Validation Accuracy: 0.7328
Naive Bayes Validation F1 Score: 0.7979
Naive Bayes Validation ROC AUC Score: 0.7936
```

Figure 7.1.21: SVM-SMOTE Naive Bayes Results

```
Gradient Boosting Training Accuracy: 0.7907
Gradient Boosting Training F1 Score: 0.8072
Gradient Boosting Training ROC AUC Score: 0.8683
Gradient Boosting Validation Accuracy: 0.7948
Gradient Boosting Validation F1 Score: 0.8403
Gradient Boosting Validation ROC AUC Score: 0.8307
```

Figure 7.1.22: SVM-SMOTE Gradient Boost Results

```
CatBoost Training Accuracy: 0.8390
CatBoost Training F1 Score: 0.8448
CatBoost Training ROC AUC Score: 0.9176
CatBoost Validation Accuracy: 0.8209
CatBoost Validation F1 Score: 0.8475
CatBoost Validation ROC AUC Score: 0.8763
```

Figure 7.1.23: SVM-SMOTE CatBoost Results

```
AdaBoost Training Accuracy: 0.8238
AdaBoost Training F1 Score: 0.8297
AdaBoost Training ROC AUC Score: 0.9023
AdaBoost Validation Accuracy: 0.8127
AdaBoost Validation F1 Score: 0.8409
AdaBoost Validation ROC AUC Score: 0.8693
```

Figure 7.1.24: SVM-SMOTE AdaBoost Results

Following are the results from ADASYN sampling

```
ADASYN LR Results
Training Accuracy: 0.7779
Training F1 Score: 0.7871
Training ROC AUC Score: 0.8615
Validation Accuracy: 0.7521
Validation F1 Score: 0.8018
Validation ROC AUC Score: 0.8155
```

Figure 7.1.25: ADASYN Logistic Regression Results

```
ADASYN DT Results
Training Accuracy: 0.7519
Training F1 Score: 0.7899
Training ROC AUC Score: 0.8503
Validation Accuracy: 0.7727
Validation F1 Score: 0.8328
Validation ROC AUC Score: 0.8151
```

Figure 7.1.26: ADASYN Decision Tree Results

```
Random Forest Training Accuracy: 0.8216
Random Forest Training F1 Score: 0.8309
Random Forest Training ROC AUC Score: 0.9113
Random Forest Validation Accuracy: 0.7961
Random Forest Validation F1 Score: 0.8374
Random Forest Validation ROC AUC Score: 0.8595
```

Figure 7.1.27: ADASYN Random Forest Results

```
SVM Training Accuracy: 0.7335
SVM Training F1 Score: 0.7465
SVM Training ROC AUC Score: 0.8149
SVM Validation Accuracy: 0.7052
SVM Validation F1 Score: 0.7579
SVM Validation ROC AUC Score: 0.7794
```

Figure 7.1.28: ADASYN SVM Results

```
Naive Bayes Training Accuracy: 0.7335
Naive Bayes Training F1 Score: 0.7616
Naive Bayes Training ROC AUC Score: 0.8349
Naive Bayes Validation Accuracy: 0.7369
Naive Bayes Validation F1 Score: 0.8000
Naive Bayes Validation ROC AUC Score: 0.7922
```

Figure 7.1.29: ADASYN Naive Bayes Results

```
Gradient Boosting Training Accuracy: 0.7508
Gradient Boosting Training F1 Score: 0.7902
Gradient Boosting Training ROC AUC Score: 0.8622
Gradient Boosting Validation Accuracy: 0.7727
Gradient Boosting Validation F1 Score: 0.8332
Gradient Boosting Validation ROC AUC Score: 0.8284
```

Figure 7.1.30: ADASYN Gradient Boost Results

```
CatBoost Training Accuracy: 0.8397
CatBoost Training F1 Score: 0.8578
CatBoost Training ROC AUC Score: 0.9047
CatBoost Validation Accuracy: 0.8354
CatBoost Validation F1 Score: 0.8594
CatBoost Validation ROC AUC Score: 0.8724
```

Figure 7.1.31: ADASYN CatBoost Results

```
AdaBoost Training Accuracy: 0.8185
AdaBoost Training F1 Score: 0.8280
AdaBoost Training ROC AUC Score: 0.8961
AdaBoost Validation Accuracy: 0.8099
AdaBoost Validation F1 Score: 0.8487
AdaBoost Validation ROC AUC Score: 0.8640
```

Figure 7.1.32: ADASYN AdaBoost Results

Following are the results from hyperparameter tuning

```
Best Parameters: {'C': 10, 'max_iter': 98, 'penalty': 'l2', 'solver': 'lbfgs'}
Training Accuracy: 0.7779
Training F1 Score: 0.7871
Training ROC AUC Score: 0.8615
Validation Accuracy: 0.7521
Validation F1 Score: 0.8018
Validation ROC AUC Score: 0.8154
```

Figure 7.1.33: HPT Logistic Regression Results

```
Decision Trees HPT

Best Parameters: {'criterion': 'gini', 'max_depth': 10, 'min_samples_leaf': 3, 'min_samples_split': 40}
Decision Tree Training Accuracy: 0.8637, Training F1 Score: 0.8665, Training ROC AUC Score: 0.9399
Decision Tree Validation Accuracy: 0.7782, Validation F1 Score: 0.8197, Validation ROC AUC Score: 0.8302
```

Figure 7.1.34: HPT Decision Tree Results

```
Best Parameters: {'max_depth': 15, 'min_samples_leaf': 5, 'min_samples_split': 10, 'n_estimators': 100}
Training Accuracy: 0.9269
Training F1 Score: 0.9286
Training ROC AUC Score: 0.9754
Validation Accuracy: 0.8113
Validation F1 Score: 0.8509
Validation ROC AUC Score: 0.8788
```

Figure 7.1.35: HPT Random Forest Results

```
Support Vector Machine Hyper Parameter Tuning

Best Parameters: {'C': 50, 'kernel': 'rbf'}
Training Accuracy: 0.8714
Training F1 Score: 0.8767
Training ROC AUC Score: 0.9380
Validation Accuracy: 0.7851
Validation F1 Score: 0.8301
Validation ROC AUC Score: 0.8507
```

Figure 7.1.36: HPT SVM Results

```
Naive Bayers Hyper Parameter Tuning

Best Parameters: {'var_smoothing': 0.001}
Training Accuracy: 0.7446
Training F1 Score: 0.7644
Training ROC AUC Score: 0.8136
Validation Accuracy: 0.7452
Validation F1 Score: 0.8026
Validation ROC AUC Score: 0.7692
```

Figure 7.1.37: HPT Naive Bayes Results

```
Gradient Boost Hyper Parameter Tuning

Best Parameters: {'learning_rate': 0.2, 'max_depth': 3, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 10, 'subsample': 1.0}
Training Accuracy: 0.8120
Training F1 Score: 0.8256
Training ROC AUC Score: 0.9037
Validation Accuracy: 0.8127
Validation F1 Score: 0.8531
Validation ROC AUC Score: 0.8631
```

Figure 7.1.38: HPT Gradient Boost Results

```
CatBoost Training Accuracy Hyper Parameter Tuning

Best Parameters: {'depth': 3, 'iterations': 40, 'l2_leaf_reg': 1, 'learning_rate': 0.1, 'loss_function': 'Logloss'}
Training Accuracy: 0.8319
Training F1 Score: 0.8378
Training ROC AUC Score: 0.9141
Validation Accuracy: 0.8264
Validation F1 Score: 0.8603
Validation ROC AUC Score: 0.8773
```

Figure 7.1.39: HPT CatBoost Results

```
AdaBoost Classifier Hyper Parameter Tuning

Best Parameters: {`base_estimator__max depth`: 2,`learning_rate:0.1,`n estimators`: 60}
Training Accuracy: 0.8396
Training F1 Score: 0.8464
Training ROC AUC Score: 0.9231
Validation Accuracy: 0.8058
Validation F1 Score: 0.8442
Validation ROC AUC Score: 0.8739
```

Figure 7.1.40: HPT AdaBoost Results

7.2 Results of The XAI and Interpretable Models

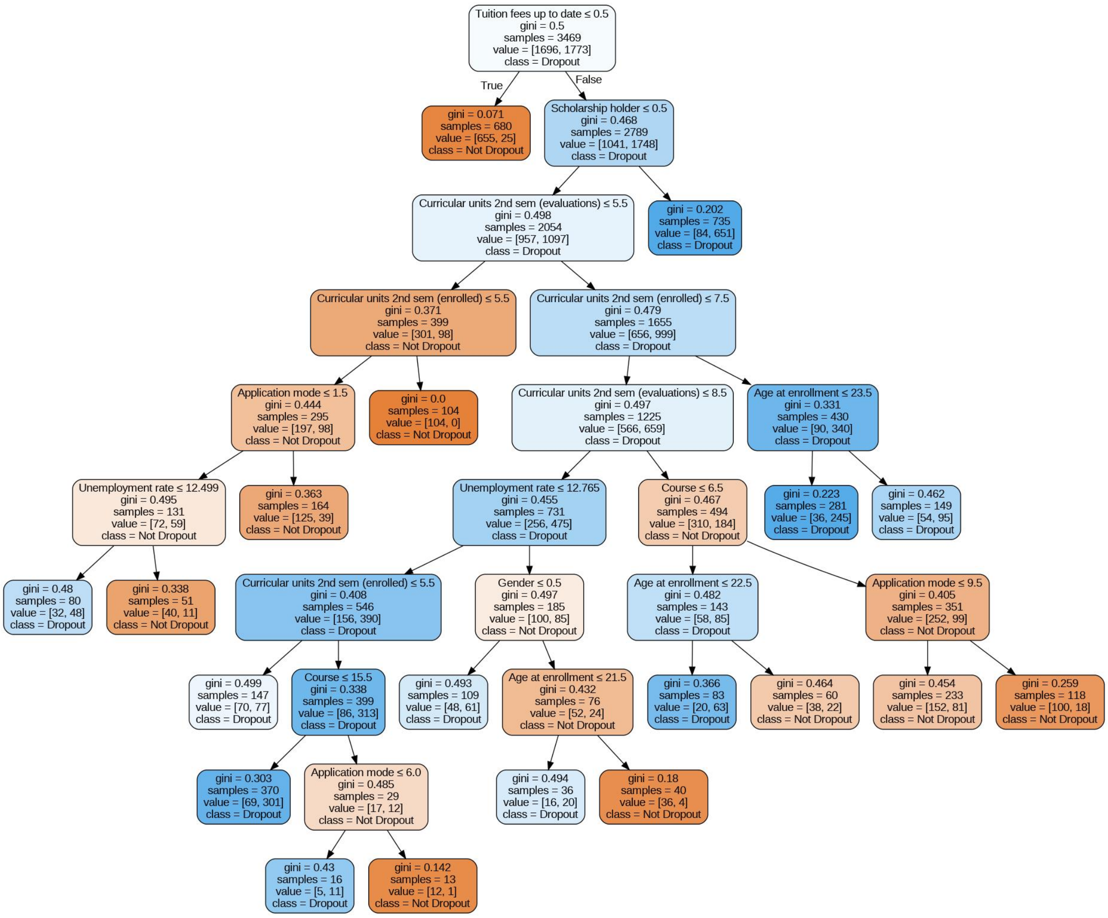


Figure 7.2.1: Pruned Decision Tree Interpretation

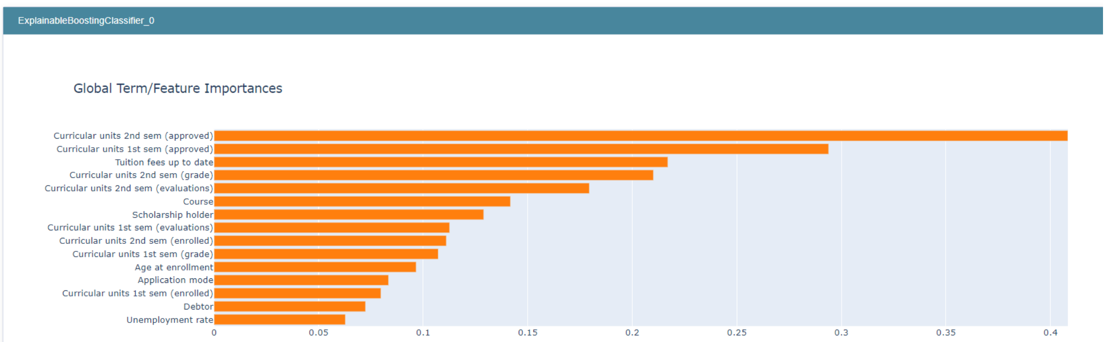


Figure 7.2.2: EBM Interpretation

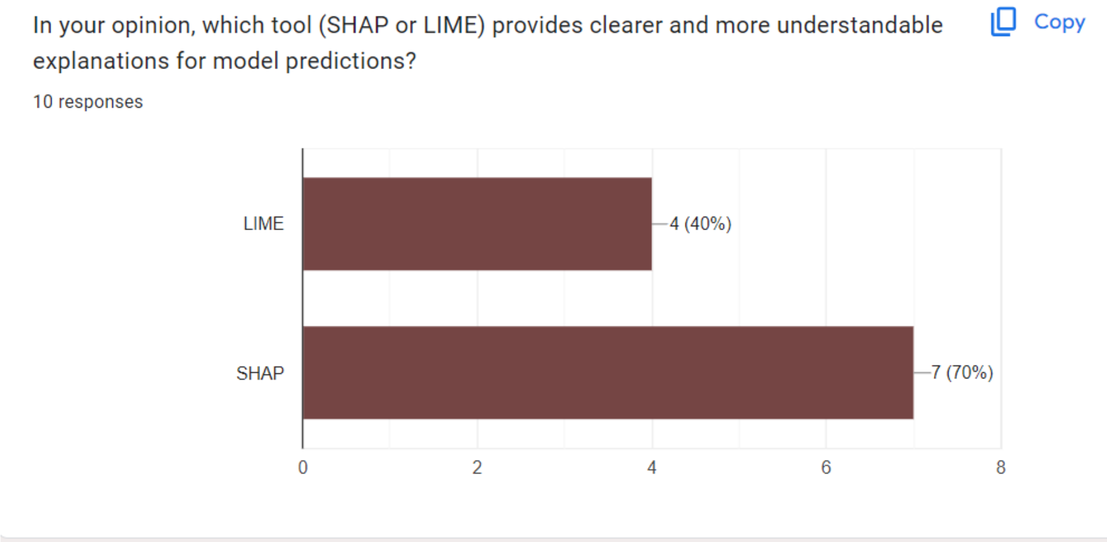


Figure 7.2.3: Preferred model

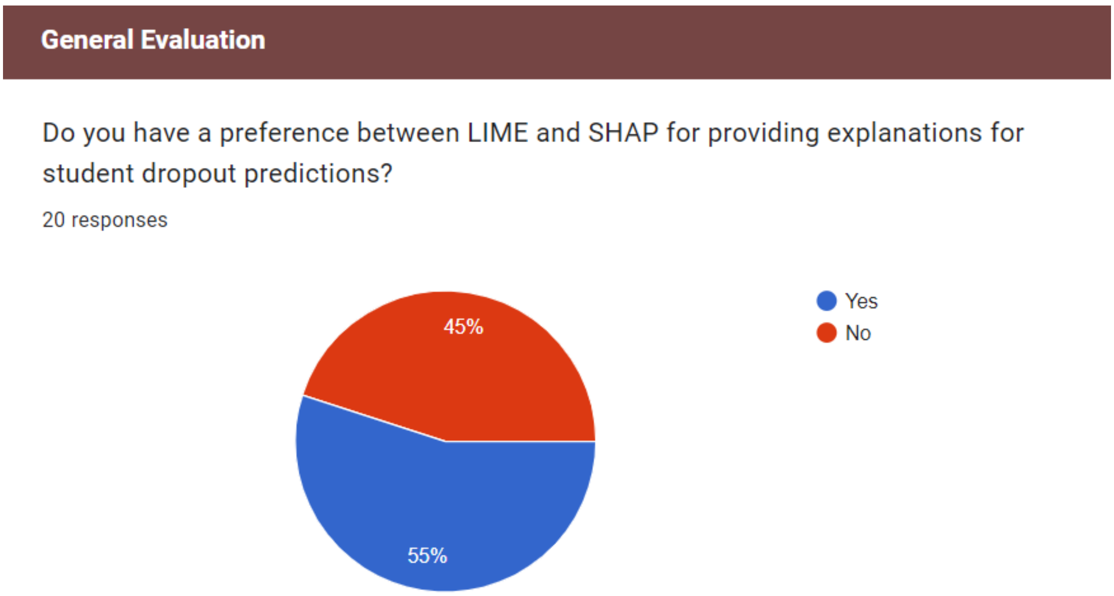


Figure 7.2.4: Preference between the two models

8 Appendix B: Code Listing

An appendix of all the codes used in the study will be included in this section.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt # data visualization
import plotly.graph_objects as go
import plotly.express as px
import seaborn as sns # data visualization

import xgboost as xgb
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split,learning_curve
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, roc_curve, auc

df=pd.read_csv("drive/MyDrive/4th_Year_Research/Implementation/Dataset/dataset.csv")
df.shape

df.head()

# ds.isnull().sum()
df.isnull().sum().values.any()

df.duplicated().values.any()

student_target = df['Target'].value_counts()
student_target

plt.pie(student_target, labels=student_target.index, autopct='%1.1f%%')
plt.show()

colors = ['#58508d', '#ff6361', '#ffa600']
plt.bar(student_target.index, student_target.values,color=colors)

# Add labels and title
plt.xlabel('Target')
plt.ylabel('No of students')
plt.title('Distribution of Target Classes')

sns.countplot(data=df, x='Gender', hue='Target', hue_order=['Enrolled', 'Graduate', 'Dropout'])

plt.xticks(ticks=[0,1], labels=['Female','Male'])
plt.ylabel('Number of Students')
plt.show()

plt.figure(figsize=(9,4))
sns.countplot(data=df, x='Marital status', hue='Target', hue_order=['Dropout', 'Enrolled', 'Graduate'])
plt.xticks(ticks=[0,1,2,3,4,5], labels=['Single','Married','Widower','Divorced','Facto Union','Legally Seperated'])
plt.xlabel('Marital Status')
plt.ylabel('Number of Students')
plt.show()

student_nationality = df.groupby(['Nacionality', 'Target']).size().reset_index().pivot(columns='Target', index='Nacionality', values=0)

```

Figure 8.0.1: Pre-processing-Part 1

```
student_nationality = student_nationality.rename(index={1:'Portuguese', 2:'German', 3:'Spanish', 4:'Italian', 5:'Dutch', 6:'English',
7:'Lithuanian', 8:'Angolan', 9:'Cape Verdean', 10:'Guinean', 11:'Mozambican', 12:'Santomean', 13:'Turkish', 14:'Brazilian', 15:'Romanian',
16:'Moldova', 17:'Mexican', 18:'Ukrainian', 19:'Russian', 20:'Cuban', 21:'Colombian'})

student_nationality_total = student_nationality.sum(axis=1)
student_nationality_sorted = student_nationality_total.sort_values(ascending=True)
student_nationality.loc[student_nationality_sorted.index].plot(kind='barh', stacked=True)

plt.xlabel('Number of Students')
plt.ylabel('Nationality')
plt.show()

student_course = df.groupby(['Course', 'Target']).size().reset_index().pivot(columns='Target', index='Course', values=0)

student_course = student_course.rename(index={1:'Biofuel Production Technologies',2:'Animation and Multimedia Design',3:'Social Service (Evening Attendance)',
                                              4:'Agronomy',5:'Communication Design',6:'Veterinary Nursing',7:'Informatics Engineering',8:'Equiniculture',
                                              9:'Management',10:'Social Service',11:'Tourism',12:'Nursing',13:'Oral Hygiene',
                                              14:'Advertising and Marketing Management',15:'Journalism and Communication',16:'Basic Education',
                                              17:'Management (Evening Attendance)'})

student_course_total = student_course.sum(axis=1)
student_course_sorted = student_course_total.sort_values(ascending=True)
student_course.loc[student_course_sorted.index].plot(kind='barh', stacked=True)

plt.xlabel('Number of Students')
plt.show()

sns.countplot(data=df, x='Displaced', hue='Target', hue_order=['Dropout', 'Enrolled', 'Graduate'])

plt.xticks(ticks=[0,1], labels=['No','Yes'])
plt.ylabel('Number of Students')
plt.show()

sns.countplot(data=df, x='International', hue='Target', hue_order=['Dropout', 'Enrolled', 'Graduate'])

plt.xticks(ticks=[0,1], labels=['No','Yes'])
plt.ylabel('Number of Students')
plt.show()

student_data=df

student_prequal = student_data.groupby(['Previous qualification', 'Target']).size().reset_index().pivot(columns='Target', index='Previous qualification', values=0)

student_prequal = student_prequal.rename(index={1:'Secondary Education',2:'Higher Education–Bachelor’s Degree',3:'Higher Education–Degree',
                                                4:'Higher Education–Master’s Degree',5:'Higher Education–Doctorate',6:'Frequency of Higher Education',
                                                7:'12th Year of Schooling–Not Completed',8:'11th Year of Schooling–Not Completed',9:'Other–11th Year of Schooling',
                                                10:'10th Year of Schooling',11:'10th Year of Schooling–Not Completed',
                                                12:'Basic Education 3rd Cycle (9th/10th/11th year) or Equivalent',
                                                13:'Basic Education 2nd Cycle (6th/7th/8th year) or Equivalent',
                                                14:'Technological Specialization Course',15:'Higher Education–Degree (1st cycle)',
                                                16:'Professional Higher Technical Course',17:'Higher Education–Master’s Degree (2nd Cycle)'})

student_prequal_total = student_prequal.sum(axis=1)
student_prequal_sorted = student_prequal_total.sort_values(ascending=True)
student_prequal_top = student_prequal_sorted[8:]
student_prequal.loc[student_prequal_top.index].plot(kind='barh', stacked=True)
```

Figure 8.0.2: Pre-processing-Part 2

```python
plt.xlabel('Count')
plt.xlabel('Number of Students')
plt.ylabel('Previous Qualification')
plt.show()

sns.displot(data=student_data, x='Age at enrollment', kde=True)
student_data['Age at enrollment'].describe()

plt.xlabel('Age at Enrolment')
plt.ylabel('Number of Students')
plt.show()

student_foccupation = student_data.groupby(["Father's occupation", 'Target']).size().reset_index().pivot(columns='Target', index="Father's occupation", values=0)

student_foccupation = student_foccupation.rename(index={1:'Student',2:'Representatives of the Legislative Power and Executive Bodies, Directors, Directors and Executive Managers',
                                    3:'Specialists in Intellectual and Scientific Activities',4:'Intermediate Level Technicians and Professions',
                                    5:'Administrative Staff',6:'Personal Services, Security and Safety Workers, and Sellers',
                                    7:'Farmers and Skilled Workers in Agriculture, Fisheries, and Forestry',
                                    8:'Skilled Workers in Industry, Construction, and Craftsmen',9:'Installation and Machine Operators and Assembly Workers',
                                    10:'Unskilled Workers',11:'Armed Forces Professions',12:'Other Situation',13:'(blank)',14:'Armed Forces Officers',
                                    15:'Armed Forces Sergeants',16:'Other Armed Forces personnel',17:'Directors of Administrative and Commercial Services',
                                    18:'Hotel, Catering, Trade, and Other Services Directors',
                                    19:'Specialists in the Physical Sciences, Mathematics, Engineering, and Related Techniques',
                                    20:'Health Professionals',21:'Teachers',
                                    22:'Specialists in Finance, Accounting, Administrative Organization, and Public and Commercial relations',
                                    23:'Intermediate Level Science and Engineering Technicians and Professions',
                                    24:'Technicians and Professionals of Intermediate Level of Health',
                                    25:'Intermediate Level Technicians from Legal, Social, Sports, Cultural, and Similar Services',
                                    26:'Information and Communication Technology Technicians',
                                    27:'Office Workers, Secretaries in General, and Data Processing Operators',
                                    28:'Data, Accounting, Statistical, Financial Services, and Registry-related Operators',
                                    29:'Other Administrative Support Staff',30:'Personal Service Workers',31:'Sellers',
                                    32:'Personal Care workers and The Like',33:'Protection and Security Services Personnel',
                                    34:'Market-oriented Farmers and Skilled Agricultural and Animal Production Workers',
                                    35:'Farmers, Livestock Keepers, Fishermen, Hunters and Gatherers, and Subsistence',
                                    36:'Skilled Construction Workers and The Like, except Electricians',
                                    37:'Skilled Workers in Metallurgy, Metalworking, and Similar',38:'Skilled workers in Electricity and Electronics',
                                    39:'Workers in Food Processing, Woodworking, and Clothing and Other industries and Crafts',
                                    40:'Fixed Plant and Machine Operators',41:'Assembly Workers',42:'Vehicle Drivers and Mobile Equipment Operators',
                                    43:'Unskilled Workers in Agriculture, Animal Production, and Fisheries and Forestry',
                                    44:'Unskilled Workers in Extractive Industry, Construction, Manufacturing, and Transport',
                                    45:'Meal Preparation Assistants',46:'Street Vendors (except food) and Street Service Providers'})

student_foccupation_total = student_foccupation.sum(axis=1)
student_foccupation_sorted = student_foccupation_total.sort_values(ascending=True)
student_foccupation_top10 = student_foccupation_sorted[36:]
student_foccupation.loc[student_foccupation_top10.index].plot(kind='barh', stacked=True)

plt.xlabel('Number of Students')
plt.ylabel("Father's Occupation")
plt.show()

student_moccupation = student_data.groupby(["Mother's occupation", 'Target']).size().reset_index().pivot(columns='Target', index="Mother's occupation", values=0)

student_moccupation = student_moccupation.rename(index={1:'Student',2:'Representatives of the Legislative Power and Executive Bodies, Directors, Directors and Executive Managers',
                                    3:'Specialists in Intellectual and Scientific Activities',4:'Intermediate Level Technicians and Professions',
                                    5:'Administrative Staff',6:'Personal Services, Security and Safety Workers, and Sellers',
```

Figure 8.0.3: Pre-processing-Part 3

```python
student_moccupation = student_moccupation.rename(index={1:'Student',2:'Representatives of the Legislative Power and Executive Bodies, Directors, Directors and Executive Managers',
                                    3:'Specialists in Intellectual and Scientific Activities',4:'Intermediate Level Technicians and Professions',
                                    5:'Administrative Staff',6:'Personal Services, Security and Safety Workers, and Sellers',
                                    7:'Farmers and Skilled Workers in Agriculture, Fisheries, and Forestry',
                                    8:'Skilled Workers in Industry, Construction, and Craftsmen',9:'Installation and Machine Operators and Assembly Workers',
                                    10:'Unskilled Workers',11:'Armed Forces Professions',12:'Other Situation',13:'(blank)',14:'Armed Forces Officers',
                                    15:'Armed Forces Sergeants',16:'Other Armed Forces personnel',17:'Directors of Administrative and Commercial Services',
                                    18:'Hotel, Catering, Trade, and Other Services Directors',1
                                    9:'Specialists in the Physical Sciences, Mathematics, Engineering, and Related Techniques',20:'Health Professionals',
                                    21:'Teachers',22:'Specialists in Finance, Accounting, Administrative Organization, and Public and Commercial relations',
                                    23:'Intermediate Level Science and Engineering Technicians and Professions',
                                    24:'Technicians and Professionals of Intermediate Level of Health',
                                    25:'Intermediate Level Technicians from Legal, Social, Sports, Cultural, and Similar Services',
                                    26:'Information and Communication Technology Technicians',27:'Office Workers, Secretaries in General, and Data Processing Operators',
                                    28:'Data, Accounting, Statistical, Financial Services, and Registry-related Operators',29:'Other Administrative Support Staff',
                                    30:'Personal Service Workers',31:'Sellers',32:'Personal Care workers and The Like',33:'Protection and Security Services Personnel',
                                    34:'Market-oriented Farmers and Skilled Agricultural and Animal Production Workers',
                                    35:'Farmers, Livestock Keepers, Fishermen, Hunters and Gatherers, and Subsistence',
                                    36:'Skilled Construction Workers and The Like, except Electricians',37:'Skilled Workers in Metallurgy, Metalworking, and Similar',
                                    38:'Skilled workers in Electricity and Electronics',39:'Workers in Food Processing, Woodworking, and Clothing and Other industries and Crafts',
                                    40:'Fixed Plant and Machine Operators',41:'Assembly Workers',42:'Vehicle Drivers and Mobile Equipment Operators',
                                    43:'Unskilled Workers in Agriculture, Animal Production, and Fisheries and Forestry',
                                    44:'Unskilled Workers in Extractive Industry, Construction, Manufacturing, and Transport',45:'Meal Preparation Assistants',
                                    46:'Street Vendors (except food) and Street Service Providers'})

student_moccupation_total = student_moccupation.sum(axis=1)
student_moccupation_sorted = student_moccupation_total.sort_values(ascending=True)
student_moccupation_top10 = student_moccupation_sorted[22:]
student_moccupation.loc[student_moccupation_top10.index].plot(kind='barh', stacked=True)

plt.xlabel('Number of Students')
plt.ylabel("Mother's Occupation")
plt.show()

sns.countplot(data=student_data, x='Educational special needs', hue='Target', hue_order=['Dropout', 'Enrolled', 'Graduate'])

plt.xticks(ticks=[0,1], labels=['No','Yes'])
plt.xlabel('Educational Special Needs')
plt.ylabel('Number of Students')
plt.show()

sns.countplot(data=student_data, x="Debtor", hue='Target', hue_order=['Dropout', 'Enrolled', 'Graduate'])

plt.xticks(ticks=[0,1], labels=['No','Yes'])
plt.xlabel('Debtor')
plt.ylabel('Number of Students')
plt.show()

sns.countplot(data=student_data, x="Tuition fees up to date", hue='Target', hue_order=['Dropout', 'Enrolled', 'Graduate'])

plt.xticks(ticks=[0,1], labels=['No','Yes'])
plt.xlabel('Tuition Fees Up to Date')
plt.ylabel('Number of Students')
plt.show()
```

Figure 8.0.4: Pre-processing-Part 4

```python
df=pd.read_csv("drive/MyDrive/4th_Year_Research/Implementation/Dataset/Pre-Processed-Dataset.csv")
df.shape

X = df.drop('Target', axis=1)
y = df['Target']

X.shape

X.info()

y.head()

# Splitting the data into training+validation (80%) and testing (20%)
X_temp, X_test, y_temp, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Splitting training+validation into actual training (60%) and validation (20%)
X_train, X_val, y_train, y_val = train_test_split(X_temp, y_temp, test_size=0.25, random_state=42)

print(X.shape, X_train.shape, X_test.shape)
```

Figure 8.0.5: Pre-processing-Part 5

```python

"""# SMOTE"""

# Initialize SMOTE
smote = SMOTE(random_state=42)

# Apply SVM-SMOTE to the training data only
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

# X_train = X_train_resampled
# Y_train = Y_train_resampled

count_of_ones = (X_train_resampled == 1).sum()
count_of_zeros = (y_train_resampled == 0).sum()
print(f"Number of rows with Target = 0: {count_of_zeros}")
print(f"Number of rows with Target = 1: {count_of_ones }")

print(X_train_resampled.shape, y_train_resampled.shape)

```

SMOTE sampling

```python
"""# SVM_SMOTE"""

# Initialize SVM-SMOTE
svm_smote = SVMSMOTE(sampling_strategy='auto', random_state=42, n_jobs=-1)

# Apply SVM-SMOTE to the training data only
X_train_resampled, y_train_resampled = svm_smote.fit_resample(X_train, y_train)

# X_train = X_train_resampled
# Y_train = Y_train_resampled

count_of_ones = (X_train_resampled == 1).sum()
count_of_zeros = (y_train_resampled == 0).sum()
print(f"Number of rows with Target = 0: {count_of_zeros}")
print(f"Number of rows with Target = 1: {count_of_ones }")

print(X_train_resampled.shape, y_train_resampled.shape)

```

Figure 8.0.6: SVM-SMOTE sampling

```python
"""# ADASYN"""

# svm_smote = SVMSMOTE(sampling_strategy='auto', random_state=42, n_jobs=-1)
from imblearn.over_sampling import ADASYN

adasyn = ADASYN(sampling_strategy='auto', random_state=42, n_neighbors=5)

X_train_resampled, y_train_resampled = adasyn.fit_resample(X_train, y_train)

count_of_ones = (X_train_resampled == 1).sum()
count_of_zeros = (y_train_resampled == 0).sum()
print(f"Number of rows with Target = 0: {count_of_zeros}")
print(f"Number of rows with Target = 1: {count_of_ones }")

print(X_train_resampled.shape, y_train_resampled.shape)

```

Figure 8.0.7: ADASYN sampling

```
"""# **Model Training**

# Logistic Regression
"""

# Train Logistic Regression model
clf = LogisticRegression(max_iter=1000, random_state=42)
clf.fit(X_train_resampled, y_train_resampled)

# Predict on training set
y_train_pred = clf.predict(X_train_resampled)
y_train_proba = clf.predict_proba(X_train_resampled)[:, 1]

# Predict on validation set
y_val_pred = clf.predict(X_val)
y_val_proba = clf.predict_proba(X_val)[:, 1]

# Calculate accuracy and F1 score for training set
accuracy_train = accuracy_score(y_train_resampled, y_train_pred)
f1_train = f1_score(y_train_resampled, y_train_pred)

# Calculate accuracy and F1 score for validation set
accuracy_val = accuracy_score(y_val, y_val_pred)
f1_val = f1_score(y_val, y_val_pred)

# Calculate ROC AUC score for training set
roc_auc_train = roc_auc_score(y_train_resampled, y_train_proba)

# Calculate ROC AUC score for validation set
roc_auc_val = roc_auc_score(y_val, y_val_proba)

# Print the metrics for both the training and validation sets
print(f"SMOTE LR Results")
print(f"Training Accuracy: {accuracy_train:.4f}")
print(f"Training F1 Score: {f1_train:.4f}")
print(f"Training ROC AUC Score: {roc_auc_train:.4f}")
print(f"Validation Accuracy: {accuracy_val:.4f}")
print(f"Validation F1 Score: {f1_val:.4f}")
print(f"Validation ROC AUC Score: {roc_auc_val:.4f}")

import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, roc_auc_score

# Plot ROC curve for the validation set
fpr, tpr, thresholds = roc_curve(y_val, y_val_proba)

# Plotting the ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'Validation ROC curve (AUC = {roc_auc_val:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Chance level')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC)')
plt.legend(loc="lower right")
plt.show()
```

Figure 8.0.8: Logistic Regression code

```
"""# Random Forest Trees"""

from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

clf_rf = RandomForestClassifier(max_depth=5, random_state=0)
clf_rf.fit(X_train_resampled, y_train_resampled)

y_train_pred_rf = clf_rf.predict(X_train_resampled)
y_train_proba_rf = clf_rf.predict_proba(X_train_resampled)[:, 1]

y_val_pred_rf = clf_rf.predict(X_val)
y_val_proba_rf = clf_rf.predict_proba(X_val)[:, 1]

accuracy_train_rf = accuracy_score(y_train_resampled, y_train_pred_rf)
f1_train_rf = f1_score(y_train_resampled, y_train_pred_rf)

accuracy_val_rf = accuracy_score(y_val, y_val_pred_rf)
f1_val_rf = f1_score(y_val, y_val_pred_rf)

roc_auc_train_rf = roc_auc_score(y_train_resampled, y_train_proba_rf)

roc_auc_val_rf = roc_auc_score(y_val, y_val_proba_rf)

print(f"SMOTE LR Results")
print(f"Random Forest Training Accuracy: {accuracy_train_rf:.4f}")
print(f"Random Forest Training F1 Score: {f1_train_rf:.4f}")
print(f"Random Forest Training ROC AUC Score: {roc_auc_train_rf:.4f}")
print(f"Random Forest Validation Accuracy: {accuracy_val_rf:.4f}")
print(f"Random Forest Validation F1 Score: {f1_val_rf:.4f}")
print(f"Random Forest Validation ROC AUC Score: {roc_auc_val_rf:.4f}")

fpr_rf, tpr_rf, thresholds_rf = roc_curve(y_val, y_val_proba_rf)

plt.figure(figsize=(8, 6))
plt.plot(fpr_rf, tpr_rf, color='darkorange', lw=2, label=f'Validation ROC curve (AUC = {roc_auc_val_rf:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Chance level')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) - Random Forest')
plt.legend(loc="lower right")
plt.show()
```

Figure 8.0.9: Random Forest code

```
"""# Naive Bayers"""

from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

clf_nb = GaussianNB()
clf_nb.fit(X_train_resampled, y_train_resampled)

y_train_pred_nb = clf_nb.predict(X_train_resampled)
y_train_proba_nb = clf_nb.predict_proba(X_train_resampled)[:, 1]

y_val_pred_nb = clf_nb.predict(X_val)
y_val_proba_nb = clf_nb.predict_proba(X_val)[:, 1]

accuracy_train_nb = accuracy_score(y_train_resampled, y_train_pred_nb)
f1_train_nb = f1_score(y_train_resampled, y_train_pred_nb)

accuracy_val_nb = accuracy_score(y_val, y_val_pred_nb)
f1_val_nb = f1_score(y_val, y_val_pred_nb)

roc_auc_train_nb = roc_auc_score(y_train_resampled, y_train_proba_nb)

roc_auc_val_nb = roc_auc_score(y_val, y_val_proba_nb)

print(f"Naive Bayes Training Accuracy: {accuracy_train_nb:.4f}")
print(f"Naive Bayes Training F1 Score: {f1_train_nb:.4f}")
print(f"Naive Bayes Training ROC AUC Score: {roc_auc_train_nb:.4f}")
print(f"Naive Bayes Validation Accuracy: {accuracy_val_nb:.4f}")
print(f"Naive Bayes Validation F1 Score: {f1_val_nb:.4f}")
print(f"Naive Bayes Validation ROC AUC Score: {roc_auc_val_nb:.4f}")

fpr_nb, tpr_nb, thresholds_nb = roc_curve(y_val, y_val_proba_nb)

plt.figure(figsize=(8, 6))
plt.plot(fpr_nb, tpr_nb, color='darkorange', lw=2, label=f'Validation ROC curve (AUC = {roc_auc_val_nb:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Chance level')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) - Naive Bayes')
plt.legend(loc="lower right")
plt.show()
```

Figure 8.0.10: Naive Bayes code

```python
"""# Support Vector Machine (SVM)"""

from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

clf_svm = SVC(probability=True, random_state=42)
clf_svm.fit(X_train_resampled, y_train_resampled)

y_train_pred_svm = clf_svm.predict(X_train_resampled)
y_train_proba_svm = clf_svm.predict_proba(X_train_resampled)[:, 1]

y_val_pred_svm = clf_svm.predict(X_val)
y_val_proba_svm = clf_svm.predict_proba(X_val)[:, 1]

accuracy_train_svm = accuracy_score(y_train_resampled, y_train_pred_svm)
f1_train_svm = f1_score(y_train_resampled, y_train_pred_svm)

accuracy_val_svm = accuracy_score(y_val, y_val_pred_svm)
f1_val_svm = f1_score(y_val, y_val_pred_svm)

roc_auc_train_svm = roc_auc_score(y_train_resampled, y_train_proba_svm)

roc_auc_val_svm = roc_auc_score(y_val, y_val_proba_svm)

print(f"SVM Training Accuracy: {accuracy_train_svm:.4f}")
print(f"SVM Training F1 Score: {f1_train_svm:.4f}")
print(f"SVM Training ROC AUC Score: {roc_auc_train_svm:.4f}")
print(f"SVM Validation Accuracy: {accuracy_val_svm:.4f}")
print(f"SVM Validation F1 Score: {f1_val_svm:.4f}")
print(f"SVM Validation ROC AUC Score: {roc_auc_val_svm:.4f}")

fpr_svm, tpr_svm, thresholds_svm = roc_curve(y_val, y_val_proba_svm)

plt.figure(figsize=(8, 6))
plt.plot(fpr_svm, tpr_svm, color='darkorange', lw=2, label=f'Validation ROC curve (AUC = {roc_auc_val_svm:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Chance level')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) - SVM')
plt.legend(loc="lower right")
plt.show()
```

Figure 8.0.11: SVM code

```python
"""# Decision Trees"""

from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

dt = DecisionTreeClassifier(random_state=0,max_depth=4,min_samples_split=2)
dt.fit(X_train_resampled, y_train_resampled)

y_train_pred = dt.predict(X_train_resampled)
y_train_proba = dt.predict_proba(X_train_resampled)[:, 1]

y_val_pred = dt.predict(X_val)
y_val_proba = dt.predict_proba(X_val)[:, 1]

accuracy_train = accuracy_score(y_train_resampled, y_train_pred)
f1_train = f1_score(y_train_resampled, y_train_pred)

accuracy_val = accuracy_score(y_val, y_val_pred)
f1_val = f1_score(y_val, y_val_pred)

roc_auc_train = roc_auc_score(y_train_resampled, y_train_proba)

roc_auc_val = roc_auc_score(y_val, y_val_proba)

print(f"SMOTE DT Results")
print(f"Training Accuracy: {accuracy_train:.4f}")
print(f"Training F1 Score: {f1_train:.4f}")
print(f"Training ROC AUC Score: {roc_auc_train:.4f}")
print(f"Validation Accuracy: {accuracy_val:.4f}")
print(f"Validation F1 Score: {f1_val:.4f}")
print(f"Validation ROC AUC Score: {roc_auc_val:.4f}")

import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, roc_auc_score

fpr, tpr, thresholds = roc_curve(y_val, y_val_proba)

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'Validation ROC curve (AUC = {roc_auc_val:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Chance level')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC)')
plt.legend(loc="lower right")
plt.show()
```

Figure 8.0.12: Decision Trees code

```python
"""HPT GB"""

from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score

gb = GradientBoostingClassifier(random_state=42)

param_grid_gb = {
    'n_estimators': [3,4,5,10],
    'learning_rate': [0.01,0.1, 0.2,0.5],
    'max_depth': [1,2,3,4,5,10],
    'min_samples_split': [2,3,4],
    'min_samples_leaf': [1,2,3],
    'subsample': [0.8, 1.0],
}

grid_search_gb = GridSearchCV(estimator=gb, param_grid=param_grid_gb, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

grid_search_gb.fit(X_train_resampled, y_train_resampled)

print(f"Gradient Boost Hyper Parameter Tuning\n")

print("Best Parameters:", grid_search_gb.best_params_)

y_train_pred_gs_gb = grid_search_gb.best_estimator_.predict(X_train_resampled)
y_val_pred_gs_gb = grid_search_gb.best_estimator_.predict(X_val)

accuracy_train_gs_gb = accuracy_score(y_train_resampled, y_train_pred_gs_gb)
f1_train_gs_gb = f1_score(y_train_resampled, y_train_pred_gs_gb)

accuracy_val_gs_gb = accuracy_score(y_val, y_val_pred_gs_gb)
f1_val_gs_gb = f1_score(y_val, y_val_pred_gs_gb)

roc_auc_train_gs_gb = roc_auc_score(y_train_resampled, grid_search_gb.best_estimator_.predict_proba(X_train_resampled)[:, 1])

roc_auc_val_gs_gb = roc_auc_score(y_val, grid_search_gb.best_estimator_.predict_proba(X_val)[:, 1])

print(f"Training Accuracy: {accuracy_train_gs_gb:.4f}")
print(f"Training F1 Score: {f1_train_gs_gb:.4f}")
print(f"Training ROC AUC Score: {roc_auc_train_gs_gb:.4f}")
print(f"Validation Accuracy: {accuracy_val_gs_gb:.4f}")
print(f"Validation F1 Score: {f1_val_gs_gb:.4f}")
print(f"Validation ROC AUC Score: {roc_auc_val_gs_gb:.4f}")
```

Figure 8.0.13: Gradient Boost code

```
"""# CatBoost"""

pip install catboost

from catboost import CatBoostClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

clf_cb = CatBoostClassifier(iterations=20,
                            depth=4,
                            learning_rate=0.1,
                            loss_function='Logloss',
                            random_seed=42)
clf_cb.fit(X_train_resampled, y_train_resampled)

y_train_pred_cb = clf_cb.predict(X_train_resampled)
y_train_proba_cb = clf_cb.predict_proba(X_train_resampled)[:, 1]

y_val_pred_cb = clf_cb.predict(X_val)
y_val_proba_cb = clf_cb.predict_proba(X_val)[:, 1]

accuracy_train_cb = accuracy_score(y_train_resampled, y_train_pred_cb)
f1_train_cb = f1_score(y_train_resampled, y_train_pred_cb)

accuracy_val_cb = accuracy_score(y_val, y_val_pred_cb)
f1_val_cb = f1_score(y_val, y_val_pred_cb)

roc_auc_train_cb = roc_auc_score(y_train_resampled, y_train_proba_cb)

roc_auc_val_cb = roc_auc_score(y_val, y_val_proba_cb)

print(f"CatBoost Training Accuracy: {accuracy_train_cb:.4f}")
print(f"CatBoost Training F1 Score: {f1_train_cb:.4f}")
print(f"CatBoost Training ROC AUC Score: {roc_auc_train_cb:.4f}")
print(f"CatBoost Validation Accuracy: {accuracy_val_cb:.4f}")
print(f"CatBoost Validation F1 Score: {f1_val_cb:.4f}")
print(f"CatBoost Validation ROC AUC Score: {roc_auc_val_cb:.4f}")


fpr_cb, tpr_cb, thresholds_cb = roc_curve(y_val, y_val_proba_cb)

plt.figure(figsize=(8, 6))
plt.plot(fpr_cb, tpr_cb, color='darkorange', lw=2, label=f'Validation ROC curve (AUC = {roc_auc_val_cb:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Chance level')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) - CatBoost')
plt.legend(loc="lower right")
plt.show()
```

Figure 8.0.14: CatBoost code

```python
"""# XGBoost"""

from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt


clf_xgb = xgb.XGBClassifier(
    n_estimators=5,
    max_depth=2,
    learning_rate=1,
    objective='binary:logistic',
    random_state=42
)
clf_xgb.fit(X_train_resampled, y_train_resampled)

y_train_pred_xgb = clf_xgb.predict(X_train_resampled)
y_train_proba_xgb = clf_xgb.predict_proba(X_train_resampled)[:, 1]

y_val_pred_xgb = clf_xgb.predict(X_val)
y_val_proba_xgb = clf_xgb.predict_proba(X_val)[:, 1]

accuracy_train_xgb = accuracy_score(y_train_resampled, y_train_pred_xgb)
f1_train_xgb = f1_score(y_train_resampled, y_train_pred_xgb)

accuracy_val_xgb = accuracy_score(y_val, y_val_pred_xgb)
f1_val_xgb = f1_score(y_val, y_val_pred_xgb)

roc_auc_train_xgb = roc_auc_score(y_train_resampled, y_train_proba_xgb)

roc_auc_val_xgb = roc_auc_score(y_val, y_val_proba_xgb)

print(f"XGBoost Training Accuracy: {accuracy_train_xgb:.4f}")
print(f"XGBoost Training F1 Score: {f1_train_xgb:.4f}")
print(f"XGBoost Training ROC AUC Score: {roc_auc_train_xgb:.4f}")
print(f"XGBoost Validation Accuracy: {accuracy_val_xgb:.4f}")
print(f"XGBoost Validation F1 Score: {f1_val_xgb:.4f}")
print(f"XGBoost Validation ROC AUC Score: {roc_auc_val_xgb:.4f}")

fpr_xgb, tpr_xgb, thresholds_xgb = roc_curve(y_val, y_val_proba_xgb)

plt.figure(figsize=(8, 6))
plt.plot(fpr_xgb, tpr_xgb, color='darkorange', lw=2, label=f'Validation ROC curve (AUC = {roc_auc_val_xgb:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Chance level')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) - XGBoost')
plt.legend(loc="lower right")
plt.show()
```

Figure 8.0.15: XG Boost code

```python
"""# AdaBoost Classifier"""

from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

clf_ab = AdaBoostClassifier(n_estimators=100,
                            learning_rate=0.1)
clf_ab.fit(X_train_resampled, y_train_resampled)

y_train_pred_ab = clf_ab.predict(X_train_resampled)
y_train_proba_ab = clf_ab.predict_proba(X_train_resampled)[:, 1]

y_val_pred_ab = clf_ab.predict(X_val)
y_val_proba_ab = clf_ab.predict_proba(X_val)[:, 1]

accuracy_train_ab = accuracy_score(y_train_resampled, y_train_pred_ab)
f1_train_ab = f1_score(y_train_resampled, y_train_pred_ab)

accuracy_val_ab = accuracy_score(y_val, y_val_pred_ab)
f1_val_ab = f1_score(y_val, y_val_pred_ab)

roc_auc_train_ab = roc_auc_score(y_train_resampled, y_train_proba_ab)

roc_auc_val_ab = roc_auc_score(y_val, y_val_proba_ab)

print(f"AdaBoost Training Accuracy: {accuracy_train_ab:.4f}")
print(f"AdaBoost Training F1 Score: {f1_train_ab:.4f}")
print(f"AdaBoost Training ROC AUC Score: {roc_auc_train_ab:.4f}")
print(f"AdaBoost Validation Accuracy: {accuracy_val_ab:.4f}")
print(f"AdaBoost Validation F1 Score: {f1_val_ab:.4f}")
print(f"AdaBoost Validation ROC AUC Score: {roc_auc_val_ab:.4f}")

fpr_ab, tpr_ab, thresholds_ab = roc_curve(y_val, y_val_proba_ab)

plt.figure(figsize=(8, 6))
plt.plot(fpr_ab, tpr_ab, color='darkorange', lw=2, label=f'Validation ROC curve (AUC = {roc_auc_val_ab:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--', label='Chance level')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) - AdaBoost')
plt.legend(loc="lower right")
plt.show()
```

Figure 8.0.16: ADABoost code

```python
param_grid = {
    'C': [0.001, 0.01, 0.1, 1, 10, 100],
    'penalty': ['l2'],
    'solver': ['liblinear', 'lbfgs'],  # 'lbfgs' works well with 'l2'
    'max_iter' :[10,50,100,150,200]
}
    # 'max_iter' :[50,100,150,300,500]  -> 100
    # 'max_iter' :[90,100,110] -> 100
    # 'max_iter' :[98,99,100,101,102,103] -> 100

# Initialize the GridSearchCV object
grid_search = GridSearchCV(estimator=lr, param_grid=param_grid, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

grid_search.fit(X_train_resampled, y_train_resampled)

print(f"Logistic Regression\n")
print("Best Parameters:", grid_search.best_params_)

y_train_pred_gs = grid_search.best_estimator_.predict(X_train_resampled)
y_val_pred_gs = grid_search.best_estimator_.predict(X_val)

accuracy_train_gs = accuracy_score(y_train_resampled, y_train_pred_gs)
f1_train_gs = f1_score(y_train_resampled, y_train_pred_gs)

accuracy_val_gs = accuracy_score(y_val, y_val_pred_gs)
f1_val_gs = f1_score(y_val, y_val_pred_gs)

# *******************
y_train_pred = clf.predict(X_train_resampled)
y_train_proba = clf.predict_proba(X_train_resampled)[:, 1]

y_val_pred = clf.predict(X_val)
y_val_proba = clf.predict_proba(X_val)[:, 1]

accuracy_train = accuracy_score(y_train_resampled, y_train_pred)
f1_train = f1_score(y_train_resampled, y_train_pred)

accuracy_val = accuracy_score(y_val, y_val_pred)
f1_val = f1_score(y_val, y_val_pred)
```

Figure 8.0.17: HPT Logistic Regression code

```python
"""HyperPT DT"""

from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score

dt = DecisionTreeClassifier(random_state=42)

param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [5,10,20,40,60,80],
    'min_samples_split': [5,10,20,30,40,50],
    'min_samples_leaf': [1, 2,3]
}

grid_search_dt = GridSearchCV(estimator=dt, param_grid=param_grid, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

grid_search_dt.fit(X_train_resampled, y_train_resampled)
print(f"Decision Trees HPT\n")
print("Best Parameters:", grid_search_dt.best_params_)
y_train_pred_gs_dt = grid_search_dt.best_estimator_.predict(X_train_resampled)
y_val_pred_gs_dt = grid_search_dt.best_estimator_.predict(X_val)

accuracy_train_gs_dt = accuracy_score(y_train_resampled, y_train_pred_gs_dt)
f1_train_gs_dt = f1_score(y_train_resampled, y_train_pred_gs_dt)

accuracy_val_gs_dt = accuracy_score(y_val, y_val_pred_gs_dt)
f1_val_gs_dt = f1_score(y_val, y_val_pred_gs_dt)

roc_auc_train_gs_dt = roc_auc_score(y_train_resampled, grid_search_dt.best_estimator_.predict_proba(X_train_resampled)[:, 1])

roc_auc_val_gs_dt = roc_auc_score(y_val, grid_search_dt.best_estimator_.predict_proba(X_val)[:, 1])

print(f"Decision Tree Training Accuracy: {accuracy_train_gs_dt:.4f}, Training F1 Score: {f1_train_gs_dt:.4f}, Training ROC AUC Score: {roc_auc_train_gs_dt:.4f}")
print(f"Decision Tree Validation Accuracy: {accuracy_val_gs_dt:.4f}, Validation F1 Score: {f1_val_gs_dt:.4f}, Validation ROC AUC Score: {roc_auc_val_gs_dt:.4f}")
```

Figure 8.0.18: HPT Decision Tree code

```python
"""hyper RF"""

from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score

rf = RandomForestClassifier(random_state=42)

param_grid_rf = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10,15,20],
    'min_samples_split': [10,20],
    'min_samples_leaf': [5,10]
}
grid_search_rf = GridSearchCV(estimator=rf, param_grid=param_grid_rf, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

grid_search_rf.fit(X_train_resampled, y_train_resampled)

print(f"Random Forest Classifier HPT\n")

print("Best Parameters:", grid_search_rf.best_params_)

y_train_pred_gs_rf = grid_search_rf.best_estimator_.predict(X_train_resampled)
y_val_pred_gs_rf = grid_search_rf.best_estimator_.predict(X_val)

accuracy_train_gs_rf = accuracy_score(y_train_resampled, y_train_pred_gs_rf)
f1_train_gs_rf = f1_score(y_train_resampled, y_train_pred_gs_rf)

accuracy_val_gs_rf = accuracy_score(y_val, y_val_pred_gs_rf)
f1_val_gs_rf = f1_score(y_val, y_val_pred_gs_rf)

roc_auc_train_gs_rf = roc_auc_score(y_train_resampled, grid_search_rf.best_estimator_.predict_proba(X_train_resampled)[:, 1])

roc_auc_val_gs_rf = roc_auc_score(y_val, grid_search_rf.best_estimator_.predict_proba(X_val)[:, 1])

print(f"Training Accuracy: {accuracy_train_gs_rf:.4f}")
print(f"Training F1 Score: {f1_train_gs_rf:.4f}")
print(f"Training ROC AUC Score: {roc_auc_train_gs_rf:.4f}")
print(f"Validation Accuracy: {accuracy_val_gs_rf:.4f}")
print(f"Validation F1 Score: {f1_val_gs_rf:.4f}")
print(f"Validation ROC AUC Score: {roc_auc_val_gs_rf:.4f}")
```

Figure 8.0.19: HPT Random Forest code

```python
"""HPT NB"""

from sklearn.model_selection import GridSearchCV
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score

gnb = GaussianNB()

param_grid_gnb = {
    'var_smoothing': [1e-09, 1e-08, 1e-07, 1e-06, 1e-05, 1e-04, 1e-03, 1e-02, 1e-01]
}

grid_search_gnb = GridSearchCV(estimator=gnb, param_grid=param_grid_gnb, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

grid_search_gnb.fit(X_train_resampled, y_train_resampled)

print(f"Naive Bayers Hyper Parameter Tuning\n")

print("Best Parameters:", grid_search_gnb.best_params_)

y_train_pred_gs_gnb = grid_search_gnb.best_estimator_.predict(X_train_resampled)
y_val_pred_gs_gnb = grid_search_gnb.bes (variable) y_train_resampled: Any

accuracy_train_gs_gnb = accuracy_score(y_train_resampled, y_train_pred_gs_gnb)
f1_train_gs_gnb = f1_score(y_train_resampled, y_train_pred_gs_gnb)

accuracy_val_gs_gnb = accuracy_score(y_val, y_val_pred_gs_gnb)
f1_val_gs_gnb = f1_score(y_val, y_val_pred_gs_gnb)

roc_auc_train_gs_gnb = roc_auc_score(y_train_resampled, grid_search_gnb.best_estimator_.predict_proba(X_train_resampled)[:, 1])
roc_auc_val_gs_gnb = roc_auc_score(y_val, grid_search_gnb.best_estimator_.predict_proba(X_val)[:, 1])

print(f"Training Accuracy: {accuracy_train_gs_gnb:.4f}")
print(f"Training F1 Score: {f1_train_gs_gnb:.4f}")
print(f"Training ROC AUC Score: {roc_auc_train_gs_gnb:.4f}")
print(f"Validation Accuracy: {accuracy_val_gs_gnb:.4f}")
print(f"Validation F1 Score: {f1_val_gs_gnb:.4f}")
print(f"Validation ROC AUC Score: {roc_auc_val_gs_gnb:.4f}")
```

Figure 8.0.20: HPT Naive Bayes code

```
hyper_parameter_tuning_final_adasyn.py > ...
"""HPT CB"""

from sklearn.model_selection import GridSearchCV
from catboost import CatBoostClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score

cb = CatBoostClassifier(random_state=42, verbose=0, auto_class_weights='Balanced')

param_grid_cb = {
    'iterations': [10,20,30,40,50,100,200,300],
    'learning_rate': [0.01,0.1,0.2,0.3],
    'depth': [2,3,4,5,10],
    'l2_leaf_reg': [1,2,3],
    'loss_function':['Logloss']
}

grid_search_cb = GridSearchCV(estimator=cb, param_grid=param_grid_cb, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

grid_search_cb.fit(X_train_resampled, y_train_resampled)

print(f"CatBoost Training Accuracy Hyper Parameter Tuning\n")

print("Best Parameters:", grid_search_cb.best_params_)

y_train_pred_gs_cb = grid_search_cb.best_estimator_.predict(X_train_resampled)
y_val_pred_gs_cb = grid_search_cb.best_estimator_.predict(X_val)

accuracy_train_gs_cb = accuracy_score(y_train_resampled, y_train_pred_gs_cb)
f1_train_gs_cb = f1_score(y_train_resampled, y_train_pred_gs_cb)

accuracy_val_gs_cb = accuracy_score(y_val, y_val_pred_gs_cb)
f1_val_gs_cb = f1_score(y_val, y_val_pred_gs_cb)

roc_auc_train_gs_cb = roc_auc_score(y_train_resampled, grid_search_cb.best_estimator_.predict_proba(X_train_resampled)[:, 1])

roc_auc_val_gs_cb = roc_auc_score(y_val, grid_search_cb.best_estimator_.predict_proba(X_val)[:, 1])

print(f"Training Accuracy: {accuracy_train_gs_cb:.4f}")
print(f"Training F1 Score: {f1_train_gs_cb:.4f}")
print(f"Training ROC AUC Score: {roc_auc_train_gs_cb:.4f}")
print(f"Validation Accuracy: {accuracy_val_gs_cb:.4f}")
print(f"Validation F1 Score: {f1_val_gs_cb:.4f}")
print(f"Validation ROC AUC Score: {roc_auc_val_gs_cb:.4f}")
```

Figure 8.0.21: HPT CatBoost code

```python
"""HPT GB"""

from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score

gb = GradientBoostingClassifier(random_state=42)

param_grid_gb = {
    'n_estimators': [3,4,5,10],
    'learning_rate': [0.01,0.1, 0.2,0.5],
    'max_depth': [1,2,3,4,5,10],
    'min_samples_split': [2,3,4],
    'min_samples_leaf': [1,2,3],
    'subsample': [0.8, 1.0],
}

grid_search_gb = GridSearchCV(estimator=gb, param_grid=param_grid_gb,
                              cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

grid_search_gb.fit(X_train_resampled, y_train_resampled)

print(f"Gradient Boost Hyper Parameter Tuning\n")

print("Best Parameters:", grid_search_gb.best_params_)

y_train_pred_gs_gb = grid_search_gb.best_estimator_.predict(X_train_resampled)
y_val_pred_gs_gb = grid_search_gb.best_estimator_.predict(X_val)

accuracy_train_gs_gb = accuracy_score(y_train_resampled, y_train_pred_gs_gb)
f1_train_gs_gb = f1_score(y_train_resampled, y_train_pred_gs_gb)

accuracy_val_gs_gb = accuracy_score(y_val, y_val_pred_gs_gb)
f1_val_gs_gb = f1_score(y_val, y_val_pred_gs_gb)

roc_auc_train_gs_gb = roc_auc_score(y_train_resampled,
        grid_search_gb.best_estimator_.predict_proba(X_train_resampled)[:, 1])

roc_auc_val_gs_gb = roc_auc_score(y_val,
                    grid_search_gb.best_estimator_.predict_proba(X_val)[:, 1])

print(f"Training Accuracy: {accuracy_train_gs_gb:.4f}")
print(f"Training F1 Score: {f1_train_gs_gb:.4f}")
print(f"Training ROC AUC Score: {roc_auc_train_gs_gb:.4f}")
print(f"Validation Accuracy: {accuracy_val_gs_gb:.4f}")
print(f"Validation F1 Score: {f1_val_gs_gb:.4f}")
print(f"Validation ROC AUC Score: {roc_auc_val_gs_gb:.4f}")
```

Figure 8.0.22: HPT Gradient Boost code

```python
"""HPT ADA"""

from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score

base_estimator = DecisionTreeClassifier(max_depth=1)
adb = AdaBoostClassifier(base_estimator=base_estimator, random_state=42)

param_grid_adb = {
    'n_estimators': [50, 60,80, 100, 200],
    'learning_rate': [0.01, 0.1, 1],
    'base_estimator__max_depth': [1, 2, 3],
}

grid_search_adb = GridSearchCV(estimator=adb, param_grid=param_grid_adb, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)

grid_search_adb.fit(X_train_resampled, y_train_resampled)

print(f"AdaBoost Classifier Hyper Parameter Tuning\n")

print("Best Parameters:", grid_search_adb.best_params_)

y_train_pred_gs_adb = grid_search_adb.best_estimator_.predict(X_train_resampled)
y_val_pred_gs_adb = grid_search_adb.best_estimator_.predict(X_val)

accuracy_train_gs_adb = accuracy_score(y_train_resampled, y_train_pred_gs_adb)
f1_train_gs_adb = f1_score(y_train_resampled, y_train_pred_gs_adb)

accuracy_val_gs_adb = accuracy_score(y_val, y_val_pred_gs_adb)
f1_val_gs_adb = f1_score(y_val, y_val_pred_gs_adb)

roc_auc_train_gs_adb = roc_auc_score(y_train_resampled, grid_search_adb.best_estimator_.predict_proba(X_train_resampled)[:, 1])

roc_auc_val_gs_adb = roc_auc_score(y_val, grid_search_adb.best_estimator_.predict_proba(X_val)[:, 1])

print(f"Training Accuracy: {accuracy_train_gs_adb:.4f}")
print(f"Training F1 Score: {f1_train_gs_adb:.4f}")
print(f"Training ROC AUC Score: {roc_auc_train_gs_adb:.4f}")
print(f"Validation Accuracy: {accuracy_val_gs_adb:.4f}")
print(f"Validation F1 Score: {f1_val_gs_adb:.4f}")
print(f"Validation ROC AUC Score: {roc_auc_val_gs_adb:.4f}")
```

Figure 8.0.23: HPT ADABoost code