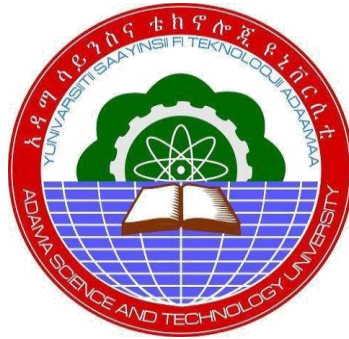


Airtime and Package Credit Risk Prediction using Deep Learning: The Case of Ethio telecom



Birhanu Gudisa Tolosa

A Thesis Submitted to the Department of Computer Science and
Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of
Master's in

Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

September, 2023

Adama, Ethiopia

Airtime and Package Credit Risk Prediction using Deep Learning: The Case of Ethio telecom

Birhanu Gudisa Tolosa

Advisor: Mesfin Abebe (PhD)

A Thesis Submitted to the Department of Computer Science and
Engineering

School of Electrical Engineering and Computing

Office of Graduate Studies

Presented in Partial Fulfillment of the Requirement for the Degree of
Master's in
Computer Science and Engineering

Adama Science and Technology University

September, 2023

Adama, Ethiopia

APPROVAL PAGE

I, the advisors of the thesis entitled “**Airtime and Package Credit Risk Prediction using Deep Learning: The Case of Ethio telecom**” developed by Birhanu Gudisa Tolosa hereby certify that the recommendations and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Dr. Mesfin Abebe

Major Advisor

Signature

Date

We, the undersigned, members of the Board of Examiners of the thesis by Birhanu Gudisa Tolosa have read and evaluated the thesis entitled “Airtime and Package Credit Risk Prediction using Deep Learning: The Case of Ethio telecom” and examined the candidate during open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

Chairperson

Signature

Date

Internal Examiner

Signature

Date

External Examiner

Signature

Date

Finally, approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

Department Head

Signature

Date

School Dean

Signature

Date

Office of Post Graduate Studies, Dean

Signature

Date

DECLARATION

I declare that this Master's thesis entitled "**Airtime and Package Credit Risk Prediction using Deep Learning: The Case of Ethio telecom**" is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate from any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation

Birhanu Gudisa Tolosa

Name of Student

Signature

Date

RECOMMENDATION OF ADVISORS/SUPERVISORS

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled “**Airtime and Package Credit Risk Prediction using Deep Learning: The Case of Ethio telecom**” prepared under my/our guidance by Birhanu Gudisa Tolosa submitted in partial fulfillment of the requirements for the degree of Masters of Science in Computer Science and Engineering. Therefore, I/we recommend the submission of a revised version of the thesis to the department following the applicable procedures.

Dr. Mesfin Abebe

Major Advisor/Supervisor

Signature

Date

Co-Advisor/Co-Supervisor

Signature

Date

ACKNOWLEDGMENT

I acknowledge that all my accomplishments are a testament to God's grace. At every turn, I've witnessed His divine intervention, lighting my way and fortifying me during challenging times. Thank you, Jesus, for being my unwavering anchor.

I express my profound gratitude to my mentor, Dr. Mesfin Abebe, for his steadfast direction, priceless wisdom, and consistent support during every phase of this research endeavor. His deep knowledge and guidance have been instrumental in charting the trajectory of this study.

I'd like to convey my sincere appreciation to Adama Science and Technology University (ASTU) for their collaborative spirit and support by providing me with vital facilities, resources, and tools necessary for this research. Their invaluable assistance was pivotal in the successful execution of the experiments and analyses.

In conclusion, I owe an immeasurable debt of gratitude to my parents, who planted in me an insatiable curiosity and thirst for knowledge. Their unwavering support has been my anchor, and their sacrifices have molded me into the individual I stand as today. This research journey would not have reached its zenith without the love, patience, and unwavering backing of my family. I am profoundly blessed to belong to such an extraordinary family unit, and their enduring influence in my life is a gift I will cherish endlessly.

TABLE OF CONTENTS

ACKNOWLEDGMENT	v
LIST OF TABLES	ix
LIST OF FIGURES	x
LIST OF ABBREVIATIONS AND ACRONYMS	xii
ABSTRACT	ii
CHAPTER ONE.....	1
1. INTRODUCTION.....	1
1.1 Background of the Study	1
1.2 Motivation for the Study.....	3
1.3 Statement of the Problem.....	4
1.4 Research Questions.....	5
1.5 Object of the Study	5
1.5.1 General Objective	5
1.5.2 Specific Objectives	5
1.6 Scope and Limitations of the Study.....	6
1.6.1 Scope of the study	6
1.6.2 Limitations of the Study	6
1.7 Significant of the Study	6
1.8 Organization of the Study	7
CHAPTER TWO.....	9
2. LITERATURE REVIEW AND RELATED WORK.....	9
2.1 Conceptual Background.....	9
2.2 Mobile Airtime and Package Credit	9
2.2.1 Mobile Credit Service Risks.....	10
2.3 Machine Learning.....	11
2.3.1 Logistic Regression	12
2.3.2 Support Vector Machine.....	12
2.3.3 Random Forest.....	13
2.3.4 Decision Tree.....	13
2.4 Deep Learning.....	13
2.5 Types of Deep Learning Algorithms	14
2.5.1 Convolutional Neural Network	14

2.5.2	Multi-layer Perceptron.....	14
2.5.3	Wide and Deep Learning.....	16
2.5.4	TabNet	17
2.6	Activation Functions.....	19
2.6.1	Sigmoid function	19
2.6.2	ReLU.....	20
2.7	Related Work	21
	Summary of Related Work.....	27
CHAPTER THREE		28
3.	RESEARCH METHODOLOGY	28
3.1	Chapter Overview	28
3.2	Overall Methodology of the Research	28
3.3	Data Collection	30
3.3.1	Data Understanding	30
3.4	Data Preparation	35
3.4.1	Data Cleaning	36
3.4.2	Data Construction.....	36
3.4.3	Data Integration	37
3.5	Development Tools.....	39
3.5.1	Hardware Tools	39
3.5.2	Software Tools.....	40
3.6	Evaluation Metrics	40
3.6.1	Confusion Matrix.....	41
3.6.2	Accuracy.....	41
3.6.3	Precision	41
3.6.4	F-score	42
CHAPTER FOUR		43
4.	PROPOSED SOLUTION	43
4.1	Chapter Overview	43
4.2	Proposed Model Architecture	43
4.3	Dataset	44
4.4	Data Preprocessing	44
4.4.1	Data Cleaning	44

4.4.2	Attribute Selection.....	45
4.4.3	Data Transformation.....	46
4.4.4	Data Integration	47
4.5	Proposed Model	47
4.6	Environmental Setup.....	47
4.7	Implantation of the proposed model	48
4.7.1	Initiating Experiment.....	48
4.7.2	Exploratory Data Analysis	49
4.7.3	Cleaning Dataset.....	51
4.7.4	Machine Learning Model implementation	57
4.7.5	Deep Learning Model Implementation.....	58
4.8	Parameter Tuning.....	63
CHAPTER FIVE		65
5.	RESULT AND DISCUSSION.....	65
5.1	Result of Machine Learning Models	65
5.2	Result of Deep Learning Algorithms.....	68
5.3	Feature Importance	76
5.4	Prototype	76
5.5	Research Question Discussion.....	78
CHAPTER SIX		79
6.	CONCLUSION AND FUTURE WORK.....	79
6.1	Conclusion	79
6.2	Future Work.....	80
References		81
Appendix A - Additional Results and Illustrations.....		i
Appendix B – Telegram bot.....		iii

LIST OF TABLES

Table 2.1 Summaries of related work.....	27
Table 3.1 Selected attributes.....	38
Table 3.2 Software tools.....	40
Table 3.3 Confusion Matrix.....	41
Table 4.1: Summaries of hyper parameters	64
Table 5.1 Performance Measures of Machine Learning Algorithms	66
Table 5.2 Performance Measures of Deep Learning Algorithms	69

LIST OF FIGURES

Figure 2.1 MLP architecture.....	15
Figure 2.2 Architecture of wide and deep learning model (Cheng et al., 2016)	17
Figure 2.3 TabNet encoder architecture (left) and TabNet decoder architecture (right) (Arik & Pfister, 2020)	18
Figure 2.4 Sigmoid function (Sharma et al., 2020)	20
Figure 2.5 ReLu activation function (Sharma et al., 2020)	21
Figure 3.1 Research methodology	29
Figure 4.1 Proposed Solution of our research	43
Figure 4.2 Libraries and frameworks required for our research.....	48
Figure 4.3 Loading Dataset	49
Figure 4.4 Exploratory Data Analysis for airtime and credit risk prediction	50
Figure 4.5 Sample datasets of customer recharge before preprocessing.....	51
Figure 4.6 Cleaning Recharge	51
Figure 4.7 Sample datasets of customer recharge after preprocessing.....	52
Figure 4.8 Sample datasets of customer profile before preprocessing	52
Figure 4.9 Customer Profile Data Preprocessing	53
Figure 4.10 Sample Customer Profile data after preprocessing	54
Figure 4.11 Sample Customer Data usage before preprocessing	54
Figure 4.12 Data usage preprocessing.....	55
Figure 4.13 Data usage after preprocessing.....	55
Figure 4.14 Loan data before preprocessing	56
Figure 4.15 Preprocessing loan data.....	56
Figure 4.16 Implementation of Logistic Regression	57
Figure 4.17 Implementation of Rando Decision Tree	58
Figure 4.18 Implementation of Support Vector Machine.....	58
Figure 4.19 MLP Model Definition.....	59
Figure 4.20 TabNet model implementation.....	60
Figure 4.21 Wide and Deep Network Implementation	61
Figure 4.22 CNN model implementation	63
Figure 5.1 Comparison of Machine Learning Algorithms using 10-fold cross validation.....	67

Figure 5.2 Confusion Matrix for Machine Learning algorithms: LR (top left), SVM (top-right) and DT bottom.....	68
Figure 5.3 Confusion Matrix for MLP (left) and TabNet (right)	71
Figure 5.4 Confusion Matrix for wide and deep model (left) and CNN (right).....	71
Figure 5.5 ROC curve for MLP	72
Figure 5.6 ROC curve for CNN model.....	73
Figure 5.7 ROC curve for Wide and Deep Model.....	73
Figure 5.8 MLP model accuracy and model loss	75
Figure 5.9 CNN accuracy and model loss	75
Figure 5.10 Feature importance in airtime and credit risk prediction	76
Figure 5.11 Sample prototype using telegram bot.....	77

LIST OF ABBREVIATIONS AND ACRONYMS

ACS	Airtime Credit Service
AUROC	Area Under the Receiver Operating Characteristic
CNN	Convolutional Neural Network
CV	Cross Validation
DT	Decision Tree
KNN	K-nearest neighbors
LR	Logistic Regression
LSTM	Long Short-Term Memory
ML	Machine Learning
MNOs	Mobile Network Operators
MLP	Multilayer-Perceptron
NN	Neural Network
PAS	Package Credit Service
PLC	Private Licensed Company
ROC	Receiver Operating Characteristic
SMS	Short Message Service
RF	Random Forest
USSD	Unstructured Supplementary Service Data
VAS	Value-Added Services

ABSTRACT

Mobile credit service is a type of service that allows prepaid mobile subscribers to use telecom services at any time and from any location after running out of balance. When there isn't a nearby top-up option or when there isn't any more money to top up, customers can use the different services that Ethio telecom provides and help the company (provider) by becoming an additional source of revenue. However, providing credit service comes with its own set of risks for the company because many subscribers do not repay their credit or they are defaulters. In this research we developed a model for airtime and credit risk prediction. To predict airtime and credit risk, deep learning algorithms are used namely Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), TabNet, and Wide and Deep network with two validation techniques 10-fold cross-validation and separate train test. To facilitate this, the dataset is collected from Ethio telecom. After dataset is collected, relevant features were chosen and preprocessing operations such as data cleaning, integrating, and aggregating were completed. The experimental results demonstrate that MLP (multilayer perceptron) classifier with a 10-fold cross validation technique obtains a higher classification accuracy of 99.00%, Wide and deep network ranked second with the accuracy of 98.22%, TabNet ranked third with accuracy of 97.54%, and CNN ranked forth with accuracy of 97.27%. We also applied machine learning algorithms namely Logistic Regression (LR), Decision Tree (DT) and Support Vector Machine (SVM). Prediction airtime and package credit has numerous advantages for telecom companies, including raising awareness to mitigate risk, preventing financial loss, and increasing revenue assurance.

Keywords: *Airtime credit, Package credit, Risk prediction, Deep Learning, Machine Learning, MLP, TabNet, CNN, Wide and Deep Network*

CHAPTER ONE

1. INTRODUCTION

1.1 Background of the Study

Communication technology has become essential and has aided in creating a society in which anyone, no matter how far away, can be reached in a matter of seconds. Due to corporate requirements and technological developments, the number of mobile subscribers is steadily increasing in the telecommunications industry, which requires a high level of multi-service delivery. One of the services included in mobile airtime credit services (Garba Muhammad et al., 2019). And other services include package credit service.

Mobile credit services like Airtime Credit Service (ACS) and Package Credit Service (PAS), which are quickly rising in developing countries, are becoming essential. As a result, several mobile network operators (MNOs) operating in emerging markets are investing in new services like low-interest package loans and short-term airtime. Mobile ACS is a short-term airtime loan that provides a solution that enables prepaid mobile customers to top up mobile airtime on a credit basis, anytime and anywhere and Mobile PCS is a short-timer package loan that provides a solution that enables prepaid mobile customers to easily buy mobile voice or data package on a credit basis, anytime and anywhere. When your airtime balance is low, when there isn't a nearby top-up option, or when you don't have any more money to top-up, both ACS and PCS will be of assistance. They contribute to lowering consumer annoyance and revenue loss for telecom providers.

The airtime loan service is implemented following two approaches. In the first approach, mobile network operators (MNOs), provide clients with airtime loans and face the risk of non-performing loans. Safaricom, through its Okoa Jahazi service, is an example of a corporation that employs such an approach. When a subscriber runs out of airtime, they can borrow money equal to the amount they had topped up in the previous seven days, with the expectation of repaying it within five days. The second approach entails collaboration between the MNO and a third-party lender, in which the MNO offers access to clients and the mobile network while the risk is shifted to the third party. ComzAfrica is an example of a lender that relies on such

collaboration. The third-party lender pays the MNO the credit amount before the customer repays it under this process. As a result, in the event of a default, the third-party loses the entire sum (Garba Muhammad et al., 2019).

The time period in which the loan has to be paid, however, varies based on the rules set by MNO. For example, some MNOs set the period to be five days, while others set it to be 72 hrs. If the loan is not settled in the given time, the subscriber who took the loan will be considered as default. To early identify subscribers who default and minimize the loss incurred due to them, MNOs implement credit scoring algorithms (Tarekegn & Ababa, 2019). The credit scoring algorithms allow MNOs to estimate the subscriber's ability to pay the loan in the given time.

By leveraging a contract with Ethio telecom, Hikma electronic PLC and Credok communication technology partners launched an airtime credit service in Ethiopia on August 31, 2018 (Brhane et al., 2021). In Ethiopia, there are three parties involved in the provision of the airtime credit service. These are the subscriber, a VAS (Value Added Service) provider, and the mobile service operator. These partners each have a unique role to play in the airtime credit service. The mobile operator is the one who provides the platform that connects the other two parties in the value chain. The subscriber is the end-user interested in the credit service. The VAS provider is the middleman who provides the credit and assumes the risk of loss if the subscriber defaults in the repayment of the credit.

The airtime loan service can be accessed via USSD (Unstructured Supplementary Service Data) by dialing *810# and following the on-screen instructions to get the desired service, or via SMS by sending one of the keywords (A, L, or C) to 810. Ethio telecom uses credit scoring to determine a user's eligibility before activating the service and granting them a loan. The following criteria are listed in the credit scoring method: The user must have a current prepaid service number, be a member of the Ethio telecom network for at least three months, have fully paid for any prior airtime credits, have no barred, blacklisted, or suspended service numbers, and have a minimum top-up of 15 Birr in airtime per month recharge history with a 10% service fee.

1.2 Motivation for the Study

On a daily basis, Ethio telecom, a leading telecommunications provider in Ethiopia, handles a multitude of inquiries related to mobile credit services from its extensive customer base. A significant number of these customers, for various reasons, are unable to meet their payment obligations in a timely manner. This results in the accumulation of bad debt which, if not addressed promptly and effectively, poses a substantial financial risk to the company. Such instances of unpaid debt can impact the company's overall financial health, deteriorating its profit margins and potentially undermining its competitive standing in the telecom industry.

Moreover, these defaults also necessitate increased efforts and resources in chasing debts and managing collections, thereby leading to additional operational challenges. The amplification of this problem might even harm the company's reputation, ultimately affecting customer trust and satisfaction levels. Given these potentially damaging ramifications, it is of utmost importance for Ethio telecom to employ robust strategies to mitigate such risks. The prime motivation of this study, therefore, is to safeguard Ethio telecom's financial and operational interests from the adverse impacts of bad debt.

In essence, this research aims to devise a predictive model using the power of deep learning algorithms. These models are anticipated to enable the company to identify prospective defaulters among customers who apply for credit service. By predicting which customers are more likely to default, Ethio telecom can take preemptive measures, either by denying credit service or by implementing risk-adjusted pricing for high-risk customers. This model would also help to optimize Ethio telecom's credit risk management processes, by enabling more targeted and effective interventions for managing credit and reducing defaults. In the long run, this could lead to significant cost savings, improved customer service, and ultimately, better financial stability for the company.

By developing such a model, we aim to bridge the gap between advanced machine learning techniques and practical financial risk management in the telecom sector, thus driving Ethio telecom's mission of delivering efficient and innovative services to its customers. This study is driven by the understanding that improved risk management strategies could play a critical role in the sustainable growth and profitability of Ethio telecom, as well as the broader telecom sector. Every day, several customers contact Ethio telecom to inquire about their mobile

credit services. Some of these clients fail to pay on time, which causes bad debt or losses for the company. Managing these kinds of risk will help lenders to protect their assets from potential loss. The motivation for this study is to protect Ethio telecom from bad debt by identifying whether or not a customer who requests credit service would default.

1.3 Statement of the Problem

Credit risk, as defined by (Bhatore et al., 2020), is a prominent challenge in the financial world. It's the looming threat of a financial setback that emerges when a borrower fails to fulfill their repayment obligations. While this risk is most commonly associated with banks and lending institutions, it has found its way into diverse sectors, including telecommunications. Any institution, whether focused on individuals or larger businesses, needs an effective strategy to manage and mitigate credit risk. The very sustainability and profitability of these organizations hinge on the efficient management of this risk.

The telecommunications sector, over the years, has ventured into the domain of offering credit-based services to its customers, further complicating its operational landscape. A significant offering among these services is the credit service provisioned by Mobile Network Operators (MNOs). Tailored for the convenience of prepaid mobile users, this facility allows them to recharge their mobile balances anytime, anywhere. However, such a service, despite its advantages, introduces new challenges. One major concern has been the consistent and timely repayment of these short-term credit facilities. The gravity of this issue is highlighted by data from the Ethio telecom Airtime Credit Service Report. By April 2019, the outstanding debt in this category had surged to a staggering Birr 291 million (Tarekegn & Ababa, 2019).

To address the rising concern of outstanding debts in the telecommunications sector, various strategies have been deployed. Notably, the use of predictive algorithms has gained traction among researchers and industry practitioners. Such algorithms function by analyzing a customer's historical data, encompassing their usage patterns, repayment history, and other relevant behaviors. The objective is clear: to preemptively identify and categorize those users who are more likely to default on their credit repayments.

This gap in the existing research methodologies presents an opportunity, one that this study seeks to capitalize on. Recognizing the intricate nature of both airtime and package credit services, our research aims to develop a predictive model that encompasses the entirety of the

credit services spectrum offered by MNOs. Such a holistic approach ensures that we're not just mitigating risks in one segment but across the board, offering a comprehensive solution to the credit risk challenge in the telecommunications sector.

In conclusion, the telecommunications sector, with its unique challenges and dynamic operational landscape, demands innovative solutions to manage credit risks. Through this study, we aspire to provide MNOs with a tool that not only anticipates potential defaulters but also strategizes credit offerings based on a more detailed, data-driven understanding of their customer base. Ultimately, the goal is to foster a symbiotic relationship where customers benefit from flexible credit offerings, and MNOs safeguard their financial interests.

1.4 Research Questions

This study, attempt to address the following research questions:

RQ1: What are the key features required for mobile airtime and credit risk prediction?

RQ2: Which deep learning algorithm can improve the accuracy of mobile airtime and package credit risk prediction?

1.5 Object of the Study

1.5.1 General Objective

The general objective of this study is to develop a model that can predict mobile airtime and package risk using deep learning techniques.

1.5.2 Specific Objectives

The following specific objectives are addressed to achieve the general objective.

- To collect a comprehensive dataset encompassing customer profile information, airtime and data usage details, loan data, and repayment data from Ethio telecom.
- To extract features necessary for predicting package and airtime credit risk.
- To research and choose deep learning methods that are appropriate for predicting package and airtime risk.
- Preprocess the raw data through handling missing values, removing duplicates, normalization, etc. to prepare the data for model building.
- To evaluate the models.

- Analyze feature importance to understand key drivers of credit risk.

1.6 Scope and Limitations of the Study

1.6.1 Scope of the study

The scope of the study within the given time and resources includes: -

- This study will focus on collecting and analyzing relevant data to develop the model.
- Predict risks of mobile airtime and package credit.
- Use deep learning algorithm approaches and machine learning to predict the risks.
- It include researching current trends in mobile airtime credit and package-related services as well as existing predictive models.

1.6.2 Limitations of the Study

The study's primary limitation is its exclusive reliance on datasets from Ethio telecom. While Ethio telecom is a significant player in the telecommunications landscape, there are other service providers in the region and globally. By focusing solely on Ethio telecom data, the research may miss nuances, trends, or perspectives that other datasets might offer.

1.7 Significant of the Study

The critical importance of managing financial risk, especially in businesses that extend credit or lending services, cannot be understated. The potential impact of customer defaults is not limited to immediate financial loss but can also ripple through the business, affecting its operational efficacy and strategic objectives.

One of the key aspects of effective risk management is the ability to accurately identify potential defaulters even before credit is extended. This step can significantly reduce the possibility of bad debt, thereby safeguarding the financial stability of the business. In the context of a telecom company like Ethio telecom, this can help ensure uninterrupted and quality service to its customers, a factor that significantly impacts the company's market reputation and competitive standing.

This study carries substantial importance because it seeks to establish a deep learning-based predictive model that can be used to assess the credit risk posed by customers. The use of such

advanced predictive algorithms will enhance the accuracy and speed of these risk evaluations, enabling the company to make informed decisions on credit extensions rapidly.

The insights derived from this research can be a valuable tool for companies like Ethio telecom. The ability to identify potential defaulters can guide their strategies when extending credit services. These strategies might involve precise credit approval processes, customized credit terms, or even differential pricing for high-risk customers. By doing so, companies not only reduce their exposure to bad debts but also foster a more financially responsible customer base.

Moreover, the predictive model developed in this study has potential implications beyond Ethio telecom. Other businesses offering credit or lending services, in the telecom sector and beyond, can adapt and implement this model, thereby benefiting from improved risk management.

In essence, the significance of this study lies in its potential to revolutionize credit risk management practices. By leveraging the power of deep learning, it aims to help businesses minimize bad debts, improve operational efficiency, and maintain financial health, ultimately contributing to their sustainable growth.

1.8 Organization of the Study

The organization of this study is divided into six chapters, which are outlined below:

Chapter One – Introduction

This foundational chapter introduces the study by providing essential context. It covers the research's background, delves into the motivation driving the investigation, outlines the problem statement, formulates the research questions to be addressed, defines the study's scope and limitations, and outlines the objectives guiding the research.

Chapter Two – Literature Review and Related Works

In this chapter, a comprehensive literature review is presented, specifically focusing on credit risk prediction. It synthesizes existing knowledge and related works in the field, providing a solid understanding of the current state of research and developments.

Chapter Three – Research Methodology

This chapter elaborates on the methodology employed in the study. It encompasses various aspects, such as data collection and preparation techniques. Additionally, it discusses the chosen design tools, frameworks, and evaluation methods that constitute the study's research approach.

Chapter Four – Proposed Solution

Within this section, an intricate explanation of the proposed solution to address the research problem is provided. The chapter includes not only a detailed conceptualization of the solution but also offers practical insights through code snippets. Furthermore, it outlines the required environment setup for the solution's implementation.

Chapter Five – Results and Discussion

The outcomes of conducted experiments are presented and discussed in this chapter. It serves as a platform to showcase the results obtained from applying the proposed solution. The discussion interprets these results within the context of the research questions, shedding light on their implications.

Chapter Six – Conclusion and Future Works

The final chapter concludes the study by summarizing its key findings and results. Building on these insights, the chapter offers well-grounded recommendations that stem from the study's outcomes. Moreover, it identifies promising avenues for future research, paving the way for further exploration in related domains.

CHAPTER TWO

2. LITERATURE REVIEW AND RELATED WORK

2.1 Conceptual Background

Nowadays, corporations, including banks, telcos, and other businesses engaged in transactions, typically keep large volumes of data. Numerous formats, such as multimedia, geographic, numerical, and hypertext publishing, are accessible for those Data. These data could be stored in several of databases or repositories. What are the best ways to use such data? Machine Learning enables computers to imitate and adapt human-like behavior. Using machine learning, each interaction, each action performed, becomes something the system can learn and use as experience for the next time (Alzubi et al., 2018). It will address the question of how to build computers that improve themselves automatically through experience (Jordan et al., 2015). It is all about making computers modify their actions to improve the actions to attain more accuracy, where accuracy is measured in terms of the number of times the chosen actions result in correct ones.

2.2 Mobile Airtime and Package Credit

Airtime in developing countries is quickly becoming a basic commodity among the rapidly growing middle class. Airtime and package credit are integral components of modern telecommunication services that play a pivotal role in facilitating seamless communication for individuals and businesses alike. It acts as a fundamental currency in the world of mobile communication, enabling users to stay connected with friends, family, and colleagues, regardless of geographical boundaries. With the widespread adoption of smartphones and the increasing reliance on mobile communication, airtime has become a lifeline for staying connected and informed.

Many prepaid users are finding it difficult to communicate or load data packages since they don't have enough airtime. This presents an opportunity for several mobile network operators (MNOs) operating in emerging markets to provide their subscribers with short-term airtime loans at a reasonable interest rate of about 10%. Their average revenue per user could rise as a result of this new offering. Once the subscriber's account is credited, the loan is paid back (Dushimimana et al., 2020). Making phone calls, sending SMS messages, and purchasing

bundles for browsing are all made simpler for the consumer by mobile credit services. When phones have airtime credit loaded, they offer a variety of advantages, including enhancing connections, enabling clear communication, and acting as a marketing tool. Mobile packages enable customers to access a variety of mobile services, including phone, data, SMS, and mobile + data bundles, for a lower cost.

Both ACS and PCS will be useful when your airtime balance is low, when there isn't a nearby top up facility, or when you don't have any more money to top-up. They help telecom operators retain customers and reduce customer annoyance.

After the customer uses airtime/package credit service they pay a service charge/fee. A service charge is a fee charged for a particular service often in addition to a standard or basic fee. It is also called a service fee. It is an amount of money added to the basic price of something to pay for the cost of dealing with the customer. It may also be referred to as a specific price for services associated with the main good or service being purchased. In other terms, a service charge is a cost added on top of any regular charges for a service (Garba Muhammad et al., 2019). In the case of Ethiotelcom, a 10% service fee will be charged.

2.2.1 Mobile Credit Service Risks

The instability of lenders' earnings is referred to as credit risk. Due to the borrower's failure to make loan payments or otherwise fulfill a contractual commitment, a financial institution has recorded financial losses that are the cause of this volatility (Tarekegn & Ababa, 2019). Credit risk will occur for telecommunications companies with airtime and package credit services. When subscribers are unable to promptly return the airtime credit they have taken, there is a risk to their airtime or package credit. Therefore, if unpaid, it will result in extra expenses for the telecom firm. As a result, to reduce their losses, financial firms like telecommunications need to examine credit risk.

Credit scoring is a tool used by financial institutions to facilitate the decision-making process of whether to accept or reject a loan. The objective of credit scoring is to help credit providers quantify and manage the financial risk involved in providing credit so that they can make better lending decisions quickly and more objectively. By considering several features of the consumer, credit scoring establishes the probability that a customer will be able to repay a loan. The features include historical and financial information about the consumer, such as loan

history, the number of transactions, and the customer's source of income. The benefits of credit scoring include the ability to make faster credit judgments, lower the cost of credit analysis, and keep track of an existing account selection (Dushimimana et al., 2020).

2.3 Machine Learning

Machine learning stands as a prominent field within Artificial Intelligence, playing a crucial role in digitalization solutions that have garnered significant focus in the digital realm (Susmita Ray, 2019). To make machines handle data more effectively, machine learning (ML) is applied. After viewing the data, we occasionally find it difficult to understand the precise information. We use machine learning in that situation. With the abundance of datasets available, the demand for machine learning is on the rise.

Machine learning is used in many industries to retrieve relevant information. Machine learning is used to get knowledge from data (Mahesh, 2018). Various machine learning techniques exist, including supervised learning, unsupervised learning, semi-supervised learning, Reinforcement learning, neural network, and deep learning.

Supervised Learning: It is the machine learning task of studying sample input-output pairs to learn a function that converts input to output. It deduces a function from labeled training data made up of a collection of practice samples (Mahesh, 2018). The algorithm acquires the mapping function that connects the input to the output (Sah, 2020). Regression and classification are the two main categories for supervised learning. A machine learning function called classification uses labeled data to train and divide new data into two or more groups. Machine learning algorithms like neural networks, decision trees, random forests, and gradient boosting machines take in feature vectors to produce predictions. These models undergo supervised training, where we feed them with paired feature vectors and their corresponding expected outputs (Heaton, 2017). Classification algorithms include Logistic Regression (LR), Decision Tree (DT), Naive Bayes, Support Vector Machine (SVM), multilayer-perceptron (MPLs), and Neural Network. The regression function is used to find correlations between two or more characteristics and then utilize those correlations to classify the target value. The most used regression technique is linear regression.

Unsupervised Learning: These are called unsupervised learning because unlike supervised learning there are no correct answers and there is no teacher. Algorithms are left to their own

devices to discover and present an interesting structure in the data. It is applied when the data is available only in the form of an input and there is no corresponding output variable. Such algorithms model the underlying patterns in the data in order to learn more about its characteristics (Sah, 2020). The algorithms for unsupervised learning extract features from the data. When new data is introduced, it recognizes the class of the data using the previously learned features. It is mostly utilized for feature reduction and clustering (Mahesh, 2018). Examples of unsupervised learning algorithms include K-means clustering, k-nearest neighbors (KNN), and Principal component analysis.

Semi-Supervised Learning: Semi-supervised machine learning merges both supervised and unsupervised machine learning techniques. This approach is particularly beneficial in situations where there's an abundance of unlabeled data, but obtaining labeled data is challenging (Mahesh, 2018). Semi-supervised learning occupies a position that bridges the gap between the methodologies of supervised and unsupervised learning. This approach entails training algorithms through a fusion of both labeled and unlabeled data points, enabling the model to grasp patterns and relationships more comprehensively (Sah, 2020).

Reinforcement learning: Given an observation of the world, the algorithm learns a policy of how to act. Every action has an effect on the environment, and that impact provides feedback to the learning algorithm (Nasteski, 2017). Reinforcement learning is utilized when the objective involves a series of decisions aimed at achieving a final reward. In this learning approach, an artificial agent receives rewards or penalties based on its actions. The primary objective is to optimize the cumulative reward over time (Sah, 2020).

2.3.1 Logistic Regression

Logistic Regression is an approach that combines weights and features linearly, followed by a non-linear transformation. This method is utilized for predictive analysis, estimating the probability of a specific outcome given observable characteristics. Its primary application is in solving classification problems, providing a binary outcome. It calculates the probability of an event occurrence (represented as 0 or 1) based on input variable values (Susmita Ray, 2019).

2.3.2 Support Vector Machine

Support Vector Machines (SVMs) are algorithms designed to categorize data points within a dataset. Instead of aiming to estimate a function, SVMs focus on classifying data by effectively

separating it using its features. SVMs have the capacity to address both classification and regression tasks (Susmita Ray, 2019).

2.3.3 Random Forest

Random forest (RF) is an ensemble learning method that consists of a large number of decision trees. The predictions of the many decision trees are combined to provide the final prediction after each tree has been trained on a random portion of the training data. Random forests consist of multiple tree models where each tree's construction is influenced by a random vector, sampled uniformly and independently for every tree in the ensemble (Breiman, 2001). It can be used for both classification and regression problems (Liaw & Wiener, 2002).

2.3.4 Decision Tree

A decision tree resembles a flowchart with its hierarchical structure, where rectangular internal nodes represent decisions and oval-shaped leaf nodes indicate outcomes. Its popularity stems from its intuitive design and simpler implementation compared to other classification techniques. Decision trees cater to both classification and regression tasks in supervised learning. The transparent nature of decision trees allows for clear interpretations, making them a favored choice among many data scientists and analysts (Priyam et al., 2013). Their adaptability and versatility further enhance their position in various application domains, from finance to healthcare.

2.4 Deep Learning

Nowadays, deep learning is a current and stimulating field of machine learning. Deep learning is the most effective, supervised, time and cost-efficient machine learning approach. It is considered to be the best choice for discovering complex architecture in high-dimensional data by employing backpropagation algorithms and Deep learning has been increasingly successful thanks to expanding computer power and the accessibility of big, labeled data sets (Borisov et al., 2021). Deep learning is a prominent subset of machine learning that focuses on hierarchically learning multiple levels of representations from data, allowing for the extraction of intricate patterns and features (Liu et al., 2015). Deep learning is a type of machine learning that excels at handling unstructured data. Deep learning approaches outperform traditional machine learning techniques. It enables computational models to learn features from data at several levels sequentially (Amitha Mathew et al., 2021). Deep learning is widely used in

business, science, and government because it performs incredibly well in a wide range of applications. These applications include adaptive testing, biological image classification, computer vision, cancer detection, natural language processing, speech recognition, and many more (Dargan et al., 2020). Additionally, it has a wide range of uses for heterogeneous data, including financial applications (such as risk analysis and creditworthiness estimation) and medical diagnostics based on patient history. A rise in predictive performance and robustness may have significant advantages in all of these applications, benefiting both customers and the businesses that offer such services (Borisov et al., 2021). Deep learning discovers intricate structures in big data sets by employing the backpropagation algorithm to determine how a machine should adjust its internal parameters that are used to compute the representation in each layer from the representation in the previous layer (Lecun et al., 2015).

2.5 Types of Deep Learning Algorithms

2.5.1 Convolutional Neural Network

Convolutional Neural Networks (CNNs), which were originally created for image and visual data processing, have found a substantial role in text analysis. Although their fundamental strength in image processing stems from their capacity to capture spatial hierarchies and local patterns, they have cleverly repurposed this skill for textual input. CNNs can detect local contextual patterns and complicated connections within sequences in the context of the text. This skill has enabled them to accomplish outstanding results in a variety of text-related jobs. CNNs have proven their adaptability and efficacy in applications ranging from sentiment analysis, where understanding the complexity and tone of statements is critical, to document classification, which requires recognizing thematic patterns, and even spam detection, where subtle cues can determine the legitimacy of a message. Deep convolutional neural networks (CNNs) are the cutting-edge solution for large-scale visual recognition (Xie & Yuille, n.d.). The Convolutional Neural Network (CNN) is a paramount architecture in the realm of deep learning. It's a type of feedforward neural network equipped with convolutional layers, adept at feature extraction from data (Li et al., 2021).

2.5.2 Multi-layer Perceptron

The multilayer perceptron has several classification and regression applications in many domains, including pattern recognition, voice recognition, and classification difficulties

(Ramchoun et al., 2016). A multilayer perceptron consists of a complex arrangement of interconnected neurons or nodes, which are illustrated in the provided diagram. These nodes are responsible for establishing a nonlinear relationship between an input vector and an output vector. The core components of this structure involve weights and output signals. The output signals are determined by the summation of inputs received by each node, and this summation is adjusted by a straightforward nonlinear transfer function, often referred to as an activation function.

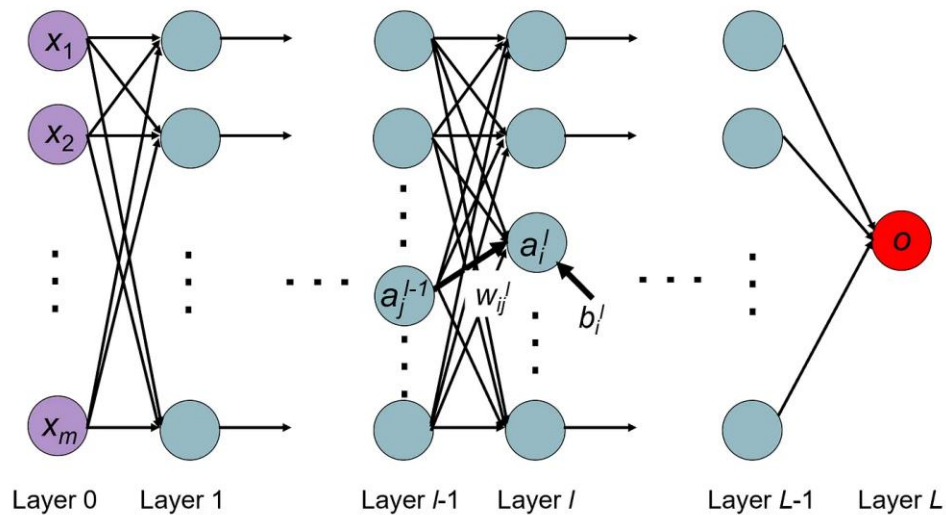


Figure 2.1 MLP architecture¹

The multilayer perceptron is a computational model composed of multiple layers of nodes. Each node within a layer is connected to nodes in the preceding and succeeding layers. These connections are characterized by adjustable weights that represent the significance of the relationship between nodes. When an input vector is fed into the network, it undergoes a series of transformations as it passes through the interconnected nodes.

The critical aspect of these transformations is the activation function, which introduces nonlinearity into the model. This function takes the weighted sum of inputs at a node and processes it to produce an output. The purpose of this nonlinearity is to enable the multilayer perceptron to capture complex relationships within the data it processes. Commonly used

¹ <https://towardsdatascience.com/multi-layer-perceptrons-8d76972afa2b>

activation functions include the sigmoid function, hyperbolic tangent (tanh) function, and rectified linear unit (ReLU) function.

2.5.3 Wide and Deep Learning

Wide & Deep Learning constitutes a technique in which neural networks are trained to simultaneously grasp insights from both basic, elemental features (referred to as the "wide" model) and intricate interactions among features (termed the "deep" model). While its original design was tailored for enhancing recommender systems, this approach has proven adaptable for tasks involving classification and prediction. The central objective of the wide and deep model is to synergistically harness the strengths of memorization and generalization (Cheng et al., 2016).

The memorization aspect pertains to the network's capacity to learn and internalize patterns of frequent co-occurrences among items or features. By discerning correlations present within historical data, the model becomes adept at predicting outcomes based on previously observed associations. On the other hand, generalization operates on the principle of correlation transitivity. This involves extrapolating from known relationships to explore novel combinations of features that may not have been encountered frequently, or even at all, in the past (Cheng et al., 2016).

The wide component of this approach is responsible for accommodating memorization. It involves a broad set of features, often encompassing categorical and sparse attributes. This wide model captures straightforward relationships that have been consistently observed in the data. Conversely, the deep component delves into the complexities of high-level feature interactions. This aspect is achieved through multiple layers of interconnected neurons that transform the input data into progressively abstract representations. The deep model excels at discovering intricate patterns and subtle associations within the data.

The amalgamation of these two components leads to a holistic learning process. By blending memorization and generalization, the wide and deep learning model can make accurate predictions based on both well-established patterns and innovative feature combinations.

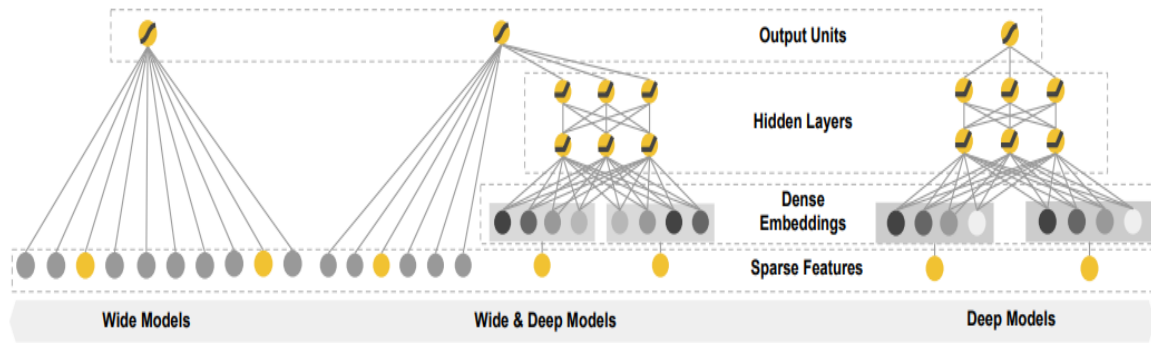


Figure 2.2 Architecture of wide and deep learning model (Cheng et al., 2016)

2.5.4 TabNet

TabNet is a state-of-the-art deep learning model specifically tailored for tabular data. One of its standout features is the utilization of sequential attention mechanisms, which enable the model to judiciously select features during each decision-making step. This not only fosters enhanced interpretability but also promotes more efficient learning by channeling the learning capacity towards the most pertinent features (Arik & Pfister, 2020).

TabNet's architectural design emphasizes sparse, data-driven feature selection. This dynamism allows the model to comprehend and rank the significance of various features, establishing a foundational understanding of feature importance. Moreover, TabNet's decision-making process unfolds sequentially. Each phase in this sequence zeroes in on a specific subset of the selected features, cumulatively building up to the final decision. This process is bolstered by the inclusion of non-linear transformations, which significantly amplify TabNet's learning capability. The model's multi-step approach, paired with its ability to capture higher-dimensional representations, equips it with characteristics akin to ensemble methods, without actually employing them (Arik & Pfister, 2020).

Diving deeper into its structural composition, TabNet's encoder is a complex assembly of intertwined components. The Feature Transformer refines input features, prepping them for subsequent decision-making. Meanwhile, the Attentive Transformer's role is pivotal in setting the stage for the next decision-making step by determining which features should be at the forefront. One of the standout elements in this architecture is Feature Masking. With each decision step, a mask is crafted, spotlighting the features the model deems paramount. This mask, when analyzed over multiple steps, offers a panoramic view of global feature importance.

Additionally, the Split Block represents a critical point in the architecture. It takes the mantle of partitioning the processed representation; one fragment is channeled to the attentive transformer, setting the tone for the subsequent decision step, while the other converges into the cumulative output. Complementing the encoder is the decoder, which boasts a dedicated feature transformer block at every step, further refining the representations. (Arik & Pfister, 2020).

The figure below shows the TabNet encoder and decoder architecture.

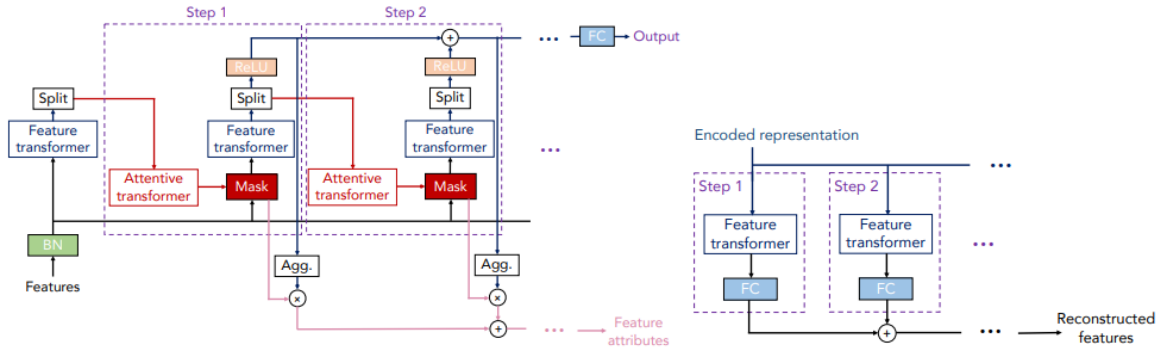


Figure 2.3 TabNet encoder architecture (left) and TabNet decoder architecture (right) (Arik & Pfister, 2020)

TabNet, has consistently showcased superior performance over several traditional and contemporary algorithms, such as DT, MLP, XGBoost, and RF. In the Forest Cover Type Dataset, which involves classifying forest cover types based on cartographic variables, TabNet's adeptness at capturing intricate patterns and relationships between these variables has enabled it to surpass other models in accuracy and interpretability. Similarly, in the Poker Hand Induction Dataset, where the challenge lies in classifying poker hands using the raw suit and rank attributes of the cards, TabNet's deep learning structure excels by efficiently processing and recognizing patterns in such high-cardinality categorical features. This prowess extends to the Sarcos Dataset as well, which focuses on regressing the inverse dynamics of an anthropomorphic robot arm. Given the complexity of this task, TabNet's capability to model non-linear relationships proves to be a significant advantage over traditional algorithms. This superior performance can be attributed to TabNet's architecture, which blends the interpretability of decision trees with the representational power of neural networks.

2.6 Activation Functions

An activation function in an artificial neural network serves as a computational mechanism that decides whether a given node should be activated by evaluating its output against a predetermined threshold. This function transforms the aggregated weighted sum of its inputs into an output signal, which is subsequently utilized by the ensuing layer for predictive computations. By introducing nonlinearity into the network, activation functions enable the system to capture intricate relationships between input and output variables. The prediction accuracy of a neural network is influenced not only by the number of layers it incorporates but, more critically, by the choice of activation function employed. This function plays a pivotal role in shaping the network's ability to learn complex patterns and make accurate predictions (Sharma et al., 2020). A neural network without activation functions essentially operates as a linear regression model, often resulting in limited performance and computational capabilities. The inclusion of activation functions elevates a neural network beyond mere linear computations, enabling it to tackle more complex tasks. This enhanced functionality is crucial for accurately modeling and interpreting intricate types of data, such as images, videos, audio, speech, and text (Sharma et al., 2020). The most common activation functions are ReLU, softmax, linear, step, and sigmoid functions.

2.6.1 Sigmoid function

It is a mathematical function that transforms numbers from 0 to 1 and has a distinctive "S"-shaped curve or sigmoid curve.

Sigmoid function defined as:

$$f(x) = \frac{1}{1 + e^{-x}}$$

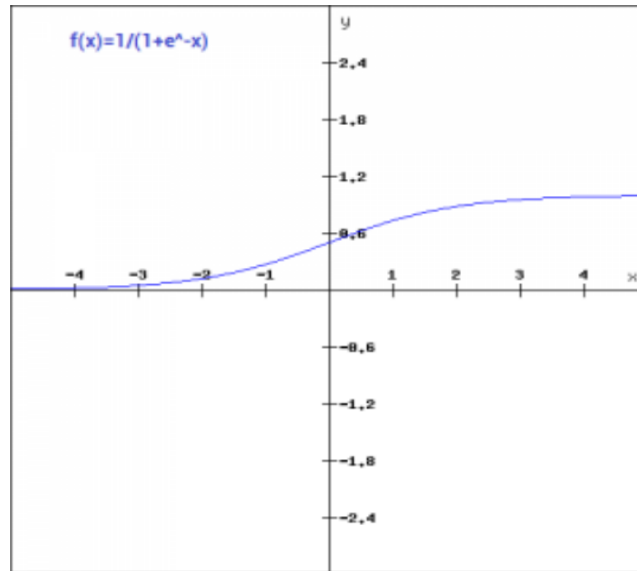


Figure 2.4 Sigmoid function (Sharma et al., 2020)

2.6.2 ReLu

The Rectified Linear Unit (ReLU) is a commonly employed non-linear activation function in neural networks. One of its key benefits is its selective activation of neurons, meaning not all neurons are triggered simultaneously. Specifically, a neuron remains inactive or is "silenced" only if the outcome of its linear transformation is zero or less. Mathematically, the ReLU function is defined as:

$$f(x) = \max(0, x)$$

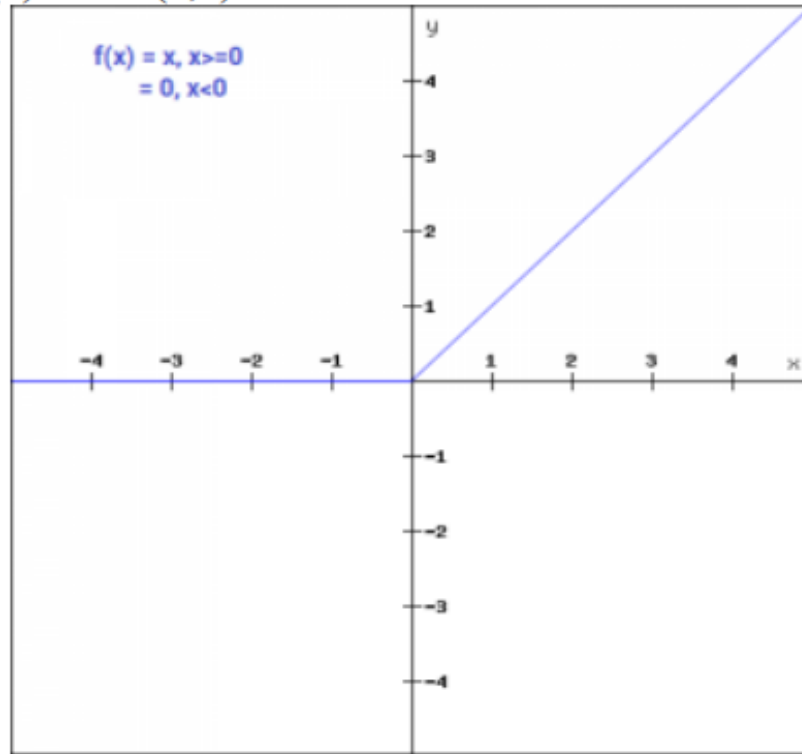


Figure 2.5 ReLu activation function (Sharma et al., 2020)

2.7 Related Work

(Tarekegn & Ababa, 2019) applied data mining approaches and compared four classification algorithms to forecast airtime credit risk for Ethio telecom. They used WEKA data mining tools for the investigation of the experiment. They obtained information from Ethio telecom. They used 86,024 instances and 11 attributes. They used accuracy, precision, recall, F-measure, and ROC area evaluation metrics. The algorithms are Naive Bayes, Multilayer Perceptron, J48 Decision Tree, and Logistic Regression. J48 decision tree outperformed the other classifiers with an accuracy of 98.5632%, precision, recall, and F-measure of 0.986. The primary limitation of this paper is the limited size of the datasets analyzed.

To determine if credit scoring is necessary, (Dushimimana et al., 2020) compare a variety of models and cross-validation methods. Logistic regression, a decision tree, and an ensemble method known as random forest are three categorization models that are assessed and contrasted. They create suitable cross-validation methods to prevent overfitting. This study aims to show that accurate credit scoring models for airtime lending may be produced without knowing the demographics of the customers. The study makes use of Comzafrica records for

46 thousand consumers and 3 million loans overall. They only make use of seven features from the dataset. They omitted client information like age and gender. The categorization models are evaluated using a confusion matrix. They present three examples of cross-validation. With no consideration of any variables, CV1 randomly divides the loan into 70:30. The CV2 similarly divided the data according to the consumer into 70:30. (i.e., a particular customer can either be included in the training set or testing set but not both). To balance the dataset, CV3 includes defaulting and non-defaulting consumers in a 50:50 ratio. The dataset is also separated into training and testing segments at a 70:30 ratio. CV1 and CV2 have high accuracy but low specificity. CV3 has lower performance than others but it has good specificity. For CV3 Random Forest beat Logistic Regression and decision tree with an accuracy of 82.3% and specificity of 84.7%. The issue with this research is that they fail to explicitly identify the performance evaluation of the algorithms and instead generalizes the scoring technique based on certain specific customer information.

(Brhane et al., 2021) has proposed a presents a machine learning approach to predict the credit risk of mobile airtime loan applicants based on their customer profile and loan information. The paper aims to address the problem of high default rates and low profitability in the mobile airtime loan industry, which is a popular service in developing countries. The paper proposes a novel framework that combines feature engineering, feature selection, and classification techniques to build a predictive model that can classify loan applicants into low-risk or high-risk groups. The paper evaluates the performance of the proposed model using a real-world dataset from an Ethiopian mobile network operator and compares it with several baseline models. The paper claims that the proposed model achieves higher accuracy, precision, recall, and F1-score than the baseline models, and also reduces the false positive rate and the false negative rate. The paper also discusses the implications of the results for mobile airtime loan providers and suggests some directions for future research. However, the paper also acknowledges some limitations of the study, such as the imbalance of the dataset. The paper also discusses the implications of the results for mobile airtime loan providers and suggests some directions for future research.

(Ereiz, 2019) use machine learning algorithms to predict default loans. The ideal prediction model was found using OptiML. BigML's automatic optimization feature, OptiML, enables

users to choose the most effective supervised learning algorithm for the supplied data. Both classification and regression problems can be resolved using it. By utilizing Bayesian parameter optimization, it operates by automatically developing and accessing several models with various configurations (decision trees, ensembles, logistic regressions, and deep networks). The dataset contains various client information, clients' households, and business. The total dataset contains 150, 091 rows and 29 attributes. 75% of the dataset was used for training and 25% for testing. Accuracy, Precision and Recall evaluation matrices are used. Machine learning models are compared to the credit scoring model used by microcredit organizations, which is based on a linear calculation of a limited set of attributes. The accuracy of the scoring model is 37.9% and the machine learning model (neural network) has an accuracy of 82.1% recall for the scoring model is 1.80% and for ML is 52.6% which indicates ML models are much better than the scoring model. However, this strategy does not work for telecom firms because their airtime/package credit service must be provided in real-time. Customers typically request the service when they wish to use a telecom company's service but realize they have no balance.

By using a minimal number of factors and a constrained dataset that solely considers a customer's history of airtime recharges, the authors of this study want to show how a trustworthy credit rating system may be created (Shema, 2019). They try to avoid more extensive and possibly sensitive datasets such as phone, and SMS logs. The dataset is collected from MNO and a lender (third-party) the dataset from MNO contains 150 million with 3 million unique mobile subscriber and dataset from lender contains 40 million with 1.7 million unique customers. They combined both dataset for processing. They selected 125,000 customers that are not borrow under the current scoring system to compare with their proposed model. They employed Random Forest classifier. They create three random forest model for three different features (recharge dataset, loan dataset and combined recharge and loan features). They use sensitivity, precision and Area under the Receiver Operating Characteristic (AUROC) curve as their evaluation metrics. Sensitivity, precision, and AUROC is 62.62%, 73.99%, and 0.79 for Recharge dataset, 24.19%, 81.63%, and 0.67 for loan dataset, 80.77%, 76.22%, and 0.80 for combined dataset. The problem with this research is that it only chooses consumers' past histories of airtime recharge, which is a relatively narrow set of data for more accurate prediction. They didn't compare several machine learning models and instead chose just one.

(Kumar Gupta & Goyal, 2018) The paper proposes a novel artificial neural network (ANN) model for predicting the credit risk of loan applicants based on their personal and financial information. The paper claims that the proposed model outperforms existing methods such as logistic regression, decision tree, and support vector machine in terms of accuracy, precision, recall, and F1-score. The paper makes several contributions to the field of credit risk prediction using ANN models. It provides a comprehensive literature review that covers the recent developments and challenges in this domain. Furthermore, they proposed a novel ANN model that incorporates various features such as age, gender, income, education, marital status, occupation, loan amount, loan duration, credit history, and purpose of loan. It makes use of various data preprocessing techniques such as normalization, outlier detection, missing value imputation, and feature selection to improve the quality and performance of the model. It also evaluates the proposed model on a real-world dataset of 614 loan applicants from an Indian bank and compares it with other methods using various metrics such as accuracy, precision, recall, and F1-score. The gap identified in this paper is that it does not provide a clear explanation of how the ANN model was trained and optimized, such as the choice of activation functions, learning rate, and number of hidden layers and neurons. The paper finds that both models have similar results in terms of accuracy, precision, recall, and F1-score, with values ranging from 0.81 to 0.85. However, the ANN model has a lower error rate than the LR model, with MAE of 0.19, RMSE of 0.44, and MAPE of 19.03%, compared to MAE of 0.21, RMSE of 0.46, and MAPE of 21.01% for the LR model. The paper also shows that the ANN model has a higher area under the ROC curve (AUC) than the LR model, with 0.86 versus 0.83. The paper concludes that both models are efficient and reliable for predicting credit risk, but the ANN model has a slight edge over the LR model due to its lower error rate and higher AUC.

In the research (Bastani et al., 2019) , the authors delved into the burgeoning realm of P2P lending platforms and highlighted the paramount importance of accurately predicting loan defaults to ensure the sustainability and robust growth of these platforms. Traditional methodologies often fall short in capturing the intricate non-linear relationships and interactivity between various features, which often results in suboptimal predictions. To bridge this gap, the study introduced a novel approach leveraging a Wide & Deep Learning model. This model ingeniously marries the strengths of both linear models (the "wide" component) and deep neural

networks (the "deep" component). While the wide component is adept at memorizing specific feature interactions, the deep component excels in generalizing patterns across the data. The dataset harnessed in this study is sourced from a P2P platform, encompassing comprehensive loan details, borrower information, and the eventual outcome of the loan—whether it defaulted or not. After a meticulous phase of feature engineering, where features were classified for wide or deep categories, the authors proceeded with the model training. The wide and deep elements were seamlessly integrated and co-trained. When evaluated against metrics such as AUC-ROC, accuracy, precision, recall, and F1-score, the WDL model showcased superior performance, overshadowing several baseline models, including Logistic Regression and Random Forest. The conclusions drawn from this research underscore the efficacy of the Wide & Deep Learning model in predicting loan defaults, especially in the domain of P2P lending platforms. By adeptly capturing both granular feature interactions and overarching data patterns, the model augments prediction accuracy. The dataset's unique nature, rooted in P2P lending, might not be directly applicable or transferable to predicting defaulters in a telecom setting. Hence, while the methodology offers promising avenues, care must be taken in ensuring its adaptability and relevance to the new dataset and domain.

In the study (McDonnell et al., 2023), researchers explore the potential of TabNet, a deep learning model tailored for tabular data, within the insurance domain. Historically, while traditional statistical models have dominated insurance risk assessments, there's a growing intrigue towards harnessing advanced machine learning models. However, the challenge lies in striking a balance between robust predictions and transparent decision-making, especially crucial in the insurance sector. The research leverages a synthetic telematics dataset, modeled on genuine data from a Canadian insurer and enriched using techniques like Synthetic Minority Oversampling Tech (SMOTE). This dataset, consisting of 100,000 samples and 52 variables, spans traditional data (e.g., car age, insured age), telematic data (e.g., miles driven, harsh acceleration), and response data. A novel response column, ClaimYN, is introduced to differentiate between risky and non-risky drivers based on claim numbers and amounts. TabNet's performance on this dataset is noteworthy. The model not only surpasses traditional insurance models in prediction accuracy but also shines in terms of interpretability. It provides invaluable insights into influential features and the underlying rationale of its predictions. In conclusion, TabNet emerges as a promising tool for the insurance industry, offering a

harmonious blend of sophisticated predictions and clear insights, positioning it as a potential game-changer in insurance analytics. However, the datasets or applications are not applicable for airtime and credit risk prediction.

Summary of Related Work

Table 2.1 Summaries of related work

Related Work	Dataset used	Methodology	Evaluation metrics	Limitation
(Tarekegn & Ababa, 2019)	Ethiotelecom customer dataset. 86024 x 11	DT, LR, NB, MLP	Accuracy, precision, recall, F-measure, ROC area	The study utilizes a small dataset, which might limit the generalizability of its findings and raise concerns about potential biases.
(Dushimimana et al., 2020)	ComzAfrica dataset 46,000 rows	LR, DT, RF	accuracy, specificity	They fail to explicitly identify the performance evaluation of the algorithms and instead generalize the scoring technique based on certain specific customer information.
(Bastani et al., 2019)	LendingClub, a P2P lending platform.	Wide & Deep Learning model	Accuracy, Precision, Recall, AUC-ROC	The dataset, specific to P2P lending, may not readily apply to telecom default predictions, requiring caution when adapting the methodology to a telecom context.
(Shema, 2019)	MNO and lender from central Africa	RF	sensitivity, precision, AUROC	It only chooses consumers' past histories of airtime recharge, which is a relatively narrow set of data for more accurate prediction. They didn't compare several machine learning models and instead chose just one.
(Brhane et al., 2021)	Ethiotelecom customer profile and loan information. 90,000 rows	DT, LR, RF, MLP	Accuracy, F measure	They utilize used imbalance of the dataset.
(Kumar Gupta & Goyal, 2018)	Indian bank 614 rows	LR, ANN	accuracy, precision, recall, and F1-score	The study was exclusively focused on applications for bank loans, and the number of datasets used was quite limited

CHAPTER THREE

3. RESEARCH METHODOLOGY

3.1 Chapter Overview

This chapter aims to provide a comprehensive overview of the research methodology employed in this study. It covers various aspects, including data collection techniques such as gathering and preprocessing, as well as the detailed explanation of model evaluation metrics.

3.2 Overall Methodology of the Research

The primary objective of this study is to develop a predictive model using deep learning techniques that can accurately assess the likelihood of payment for Ethio telecom airtime and package credit. We used the Research methodology to propose the solution, which deals with data collection, data analysis, algorithm selection techniques, and interpretations to achieve the objectives and answer the research questions. It is the science of studying how research is carried out systematically.

We employed the procedures and approaches mentioned below to meet the research objectives and develop our research:

- **Literature Review:**
 - Objective: Acquire foundational knowledge and identify gaps in the existing literature.
 - Process: Reviewed a broad spectrum of sources, such as academic articles, journals, and related online resources.
- **Problem Identification:**
 - Objective: Define the core challenge to address.
 - Process: Leveraged insights from the literature review to pinpoint the specific problems in the current predictive models.
- **Data Collection:**
 - Objective: Accumulate relevant data.
 - Process: Utilized both qualitative and quantitative methods, ensuring the data's depth and breadth catered to the intricacies of the problem.
- **Data Preprocessing:**

- Objective: Make data model-ready.
 - Process: Cleaned anomalies, managed missing values, standardized scales, and encoded categories to ensure a clean dataset.
- **Exploratory Data Analysis:**
 - Objective: Understand the underlying structure of the data
 - Process: Visualized distributions, analyzed correlations and gain a comprehensive understanding of the dataset.
- **Model Design and Testing:**
 - Objective: Construct the predictive frameworks and assess their performance.
 - Process: Chose a suitable deep learning architecture, initialized specific parameters, and iteratively refined based on test outcomes.
- **Evaluation:**
 - Objective: Evaluating the constructed models and choose the best one.
 - Process: Training the proposed models

Figure 3.1 show the overall processing starting from literature review to evaluation phase.

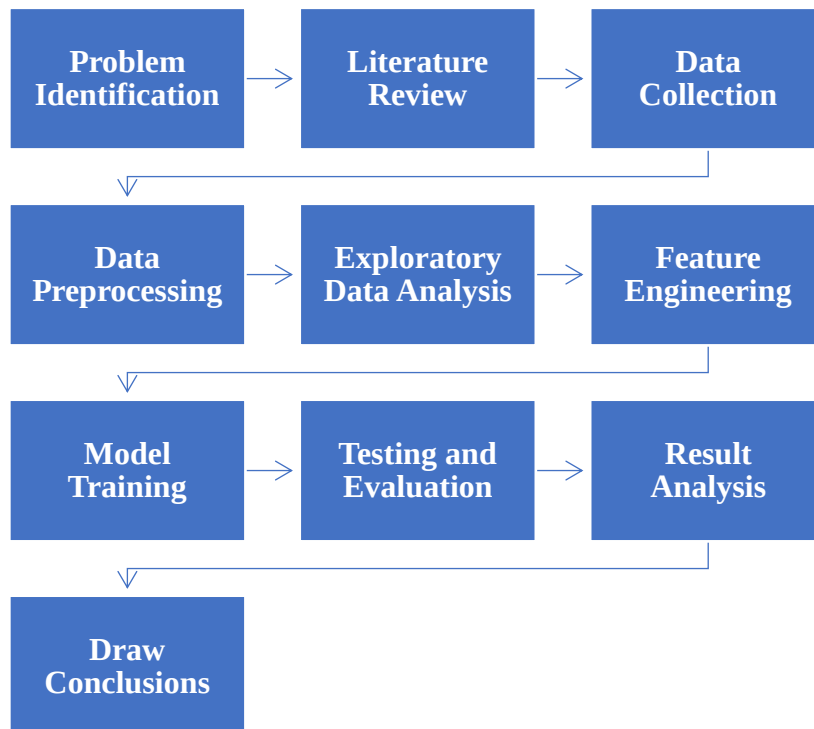


Figure 3.1 Research methodology

3.3 Data Collection

In order for deep learning (machine learning) models to effectively learn and perform, high-quality data is essential. The primary objective of this research is to gather the necessary data and prepare it for further processing. The data for this study is collected from the prepaid mobile customer's database of Ethio telecom. This means that the researchers aim to collect relevant and accurate data from a reliable source, in this case, the prepaid mobile customer's database of Ethio telecom, to ensure that their deep learning models can effectively learn from the data and produce accurate results. Data preparation is also an important step in this process, as it involves cleaning, organizing, and formatting the data to make it suitable for analysis. By gathering high-quality data and preparing it properly, the researchers aim to develop a robust and accurate predictive model using deep learning techniques.

3.3.1 Data Understanding

The data collected for this study comprises the following sets:

- Customer Loan Information Data
- Mobile Customer Profile data
- Customer Recharge information data
- Customer Usage Data

The above datasets have many features in the Ethio telecom database, however the expert recommends that they select a lower number of attributes for this thesis.

3.3.1.1 Customer Loan Information Data

This dataset provides a comprehensive overview of subscribers who have availed advance loans within the context of the telecommunications industry. It contains a number of important details that provide insight into the loan management procedure and the credit behavior of these subscribers.

The data set includes crucial information such as loan amount received by each subscriber is recorded, providing insights into the extent of credit availed by individuals. Moreover, the dataset captures the repaid loan amount, which highlights the portion of the loan that has been successfully paid back by the subscribers. This information is crucial for assessing the repayment capabilities and financial responsibility of the borrowers. Additionally, the remaining

loan amount attribute indicates the outstanding balance that subscribers still owe, thereby facilitating an understanding of the extent of their financial obligations.

To further understand the financial dynamics associated with these loans, the dataset includes the poundage fee, which refers to the additional charge or fee levied on the loan amount. This fee is often associated with administrative costs or interest charges and contributes to the overall loan repayment burden faced by subscribers. By analyzing this loan data, it becomes possible to categorize subscribers based on their loan repayment behavior. In the case of Ethio telecom, a specific criterion is applied to classify subscribers as having a good loan score or a bad loan score. Subscribers who repay their loans within seven days of receiving the credit are considered to have a good loan score. This indicates their promptness and reliability in meeting their financial obligations. Conversely, customers who do not adhere to this repayment timeframe are categorized as having a bad loan score.

By leveraging the information provided in this dataset, telecom service providers can gain valuable insights into the creditworthiness and risk profiles of their subscribers. This knowledge can inform decision-making processes related to credit extension, risk mitigation, and overall loan management strategies. Furthermore, it serves as a foundation for predicting credit risk in airtime and package transactions using deep learning techniques, as proposed in the thesis.

Ethio telecom offers both package and airtime credit services. For database storage, the company translates both airtime and package values into their monetary equivalents, consolidating them under the ‘Customer Loan Information Data’

The loan information dataset comprises several attributes that provide valuable insights into the loan repayment behavior of customers, facilitating the identification of customers with different loan scores. The selected attributes, along with their descriptions, are as follows:

- **Service Number (Phone Number):** This attribute serves as a unique identifier for each customer, allowing for individual customer tracking and analysis.

- **Operation Type:** This attribute indicates the means of the latest payment made by the customer, such as recharge, transfer, or other relevant methods. It provides insights into the customer's preferred mode of repayment.
- **Initial Loan Amount:** This attribute denotes the initial amount of the loan taken by the subscriber. It represents the advance credit provided to the customer.
- **Loan Amount (Loan Balance):** The loan balance attribute reflects the remaining unpaid loan amount. It provides a measure of the customer's outstanding financial obligation.
- **Loan Poundage:** This attribute represents the remaining unpaid poundage amount associated with the loan. It indicates the additional fee or commission charged on top of the original loan amount.
- **Repay Amount:** This attribute captures the amount repaid by the customer from the initial loan. It demonstrates the customer's repayment behavior and progress in meeting their financial obligations.
- **Repay Poundage:** The repay poundage attribute signifies the amount repaid by the customer from the initial loan poundage. It reflects the customer's adherence to repaying the associated commission or fee.
- **Entry Date:** This attribute records the date of the customer's latest repayment. It allows for tracking the repayment patterns and frequency of customers over time.

3.3.1.2 Mobile Customer Profile Data

The Mobile Customer Profile data contains the following attributes: From the Mobile Customer Profile data, it contains the following attributes:

- **Service Number (Phone Number):** This attribute serves as an identification number used to access the mobile service. It uniquely identifies each customer's account.
- **Customer Type:** This attribute identifies the type of customer. For the purpose of this study, only the "Individual" customer type is selected. It helps distinguish individual customers from other types, such as corporate or institutional customers.
- **Network Business Type:** This attribute indicates the type of network used by the customer. It includes options such as GSM (2G), WCDMA (3G), and LTE (4G) networks. It provides insights into the technology preferences of customers.

- **Status:** The status attribute reflects the current status of the customer's account. For this study, only the "Active" status is selected, indicating that the customer's account is currently active and in use.
- **Age:** This attribute represents the age of the customer at the time of subscription. It provides information about the age distribution of customers and their demographics.
- **Gender:** The gender attribute indicates the gender of individual customers. It includes options such as "Male" or "Female." Understanding the gender distribution among customers can help in analyzing gender-specific preferences or patterns.
- **Education:** This attribute provides information about the educational attainment of the customer. It includes categories such as "Illiterate," "Secondary School," "Primary School," "Diploma/Certificate," "Bachelor's Degree," and "Master's Degree and above." It offers insights into the educational background of customers and its potential impact on their preferences and behaviors.
- **Religion:** The religion attribute represents the religion of the customer. It provides information about the religious diversity within the customer base.

In this study, the selected attributes from the Mobile Customer Profile data will be utilized to analyze their relationship with the payment likelihood of airtime and package credit provided by Ethio telecom. Deep learning techniques will be employed to uncover patterns and predict the credit risk associated with these transactions. By examining the selected attributes, the study aims to gain a comprehensive understanding of how customer demographics, network preferences, age, gender, education, and religion can influence payment behaviors in the context of airtime and package credit.

3.3.1.3 Customer Usage information data

The dataset provides insights into the usage patterns of customers, specifically related to voice calls, SMS, and internet usage. The dataset includes the following attributes:

- **Service Number:** This attribute serves as an identification number used to access the service. It uniquely identifies each customer's account.
- **SGMT Type:** The SGMT (Subscription) Type attribute indicates the subscription type of the customer, distinguishing between prepaid, postpaid, or hybrid plans.

- **Customer Type:** This attribute identifies the customer type as either an individual or enterprise customer.
- **Network Type:** The Network Type attribute indicates the type of network used by the customer, such as 2G, 3G, or 4G.
- **Activate Date:** The Activate Date attribute represents the date when a user's service number becomes active or is activated for use on the mobile network. It marks the starting point of the customer's connection to the Ethio telecom network.
- **Status:** The Status attribute represents the current status of the customer, indicating whether the account is active, suspended, terminated, or in any other state.
- **Voice Usage:** This attribute refers to the amount of time a user spends making voice calls or engaging in voice communication over the mobile network.
- **SMS Usage:** The SMS Usage attribute indicates the number of text messages sent or received by a user over the mobile network.
- **Data (Internet) Usage:** This attribute represents the amount of data consumed by a user when accessing the internet or using online services over the mobile network.

These selected attributes provide valuable insights into customer behavior, preferences, and usage patterns related to voice calls, SMS, and internet usage. Analyzing this data can contribute to understanding customer preferences and usage trends, which can further assist in predicting the payment likelihood for Ethio telecom airtime and package credit using deep learning techniques. By studying the selected attributes, the research can uncover patterns and correlations that may exist between customer usage behavior and payment behaviors, aiding in credit risk prediction and management for Ethio telecom.

3.3.1.4 Recharge information data

The Recharge Information Data consists of information regarding customer recharge activities for airtimes. It includes the following attributes:

- **Service Number:** This attribute serves as an identification number used to access the service. It uniquely identifies each customer's account.
- **SGMT Type:** The SGMT (Service or Subscription) Type attribute distinguishes between different service types, such as prepaid, postpaid, or hybrid plans.

- **Recharge Amount:** This attribute represents the value added to the customer's account balance through a recharge transaction. The recharge amount enables users to continue using mobile services, including making calls, sending text messages, and accessing the internet.
- **Date of Recharge:** The Date of Recharge attribute indicates the specific date and time when a prepaid mobile account or SIM card is topped up or refilled with a recharge amount.

These attributes provide valuable insights into customer recharge behavior, including the frequency and amounts of recharges made. Analyzing this data can help researchers and service providers understand customer usage patterns, recharge preferences, and their potential impact on payment likelihood for Ethio telecom airtime and package credit.

By studying the selected attributes within the Recharge Information Data, researchers can identify trends in customer recharge behavior, such as peak recharge periods, preferred recharge amounts, or patterns associated with payment behaviors. This analysis can contribute to predicting payment likelihood for airtime and package credit, aiding in credit risk assessment and management for Ethio telecom.

In the datasets provided by Ethio telecom, the service numbers have been specifically altered to uphold customer privacy. This unique modification involves multiplying the numbers by a certain constant. While this change ensures the confidentiality of users associated with each service number, the overall analytical value of the dataset remains intact.

3.4 Data Preparation

Data must be arranged into an appropriate format before it can be analyzed. The process of modifying and arranging data prior to analysis is known as data preparation. Data preparation is often an iterative process of transforming raw data, which is frequently unstructured and haphazard, into a more structured and useful format that is ready for further analysis (Abdallah et al., 2017).

The data preprocessing can often have a significant impact on generalization performance of Machine learning and Deep learning algorithms. Data preparation in research process involves organizing, cleaning, transforming, and formatting raw data to make it suitable for analysis (S. B. Kotsiantis et al., 2006). It ensures that the data is accurate, complete, consistent, and in a

format that can be effectively utilized by statistical or analytical techniques. It includes the following steps data cleaning, data integration, data transformation, variable selection and data formatting.

3.4.1 Data Cleaning

Data cleaning is a critical step in the data preparation process that ensures the quality and reliability of the dataset for analysis. It involves identifying and addressing errors, inconsistencies, missing values, and anomalies that could affect the validity of the research findings. By rectifying or mitigating these issues, data cleaning enhances the accuracy and integrity of the data.

Specific attributes with limited data availability, such as religion and bonus offering, are excluded from the analysis due to their low coverage in the dataset. Similarly, attributes like loan penalty, loan penalty left, loan, and balance type, which have consistent values across all data instances, are removed as they may not contribute meaningful insights to the analysis. By performing these data cleaning tasks, you ensure that your dataset is free from inconsistencies, errors, and missing values, creating a reliable foundation for your analysis on airtime and package credit risk prediction using deep learning techniques. Additionally, any rows in the dataset containing NaN (missing) values are also eliminated.

3.4.2 Data Construction

In addition to the existing attributes, there are several additional attributes that can be constructed from the dataset, which can provide vital information and enhance the dataset for analysis. These attributes include:

- **Network Age:** Calculated from the activation date, this attribute represents the duration of a customer's connection to the Ethio telecom network. It provides insights into the length of time a customer has been using the service, which can be indicative of their loyalty or tenure.
- **Total Usages:** By aggregating the usage data for voice calls, SMS, and data, this attribute captures the overall usage behavior of a customer. It reflects the cumulative amount of time spent on voice calls, the number of text messages sent or received, and the total

data consumed over the specified period. This information helps understand the customer's level of engagement with the mobile services.

- **Total Recharge:** This attribute represents the cumulative value of all recharges made by a customer. It sums up the recharge amounts over a specific period, providing insights into the customer's recharge behavior and the overall investment they make into their account balance.
- **Recharge Frequency:** this attribute counts the number of airtime recharge by a customer.
- **Total Loan Amount:** By summing up the loan amounts taken by a customer, this attribute provides an overview of the total credit extended to them. It reflects the customer's utilization of advance loan facilities and can indicate their financial needs or repayment behavior.
- **Number of Loans (Frequency of Loan):** This attribute counts the number of loans taken by a customer. It represents the frequency of loan usage and can be an indicator of their reliance on credit facilities or financial stability.
- **Total Consumption:** This attributes represent the total consumption by a user. It sums up SMS usage, data usage and voice usage by a customer.

By incorporating these additional attributes into the dataset, researchers can gain a deeper understanding of customer behavior, usage patterns, and financial interactions. This enriched dataset enables more comprehensive analysis and can contribute to the prediction of payment likelihood for Ethio telecom airtime and package credit using deep learning techniques.

3.4.3 Data Integration

In order to have a unified dataset for analysis, the individual datasets should be merged together based on the common attribute of "Service Number" or "Phone Number." By matching the service numbers across the datasets, the relevant information from each dataset can be combined into a single consolidated dataset.

The merging process involves combining the attributes from each dataset into a unified structure, ensuring that the corresponding information for each service number is aligned

correctly. This enables the creation of a comprehensive dataset that incorporates customer loan information, mobile customer profiles, recharge information, and usage data.

By merging the datasets, researchers can gain a holistic view of the customers, their loan history, profile characteristics, recharge behavior, and usage patterns. This integrated dataset provides a more comprehensive and detailed understanding of the factors that influence payment likelihood for Ethio telecom airtime and package credit. It is important to ensure data integrity and accuracy during the merging process, verifying that the merged dataset aligns correctly and contains the complete and relevant information from each individual dataset.

From the provided datasets, specific attributes were meticulously selected based on insights and expertise in customer credit services from Ethio telecom. This expert deep understanding of the nuances and intricacies of the domain ensured that the chosen attributes were the most pertinent and relevant for analysis. This optimized the outcomes and enhanced the reliability of the results. Such a targeted and informed approach highlights the significance of domain-specific knowledge in data-driven projects, ensuring that the analysis remains both contextual and meaningful.

Table 3.1 Selected attributes

No	Feature	Description
1	Service Number	Identification number used to access the service. It uniquely identifies each customer's account.
2	Age	Represents the age of the customer at the time of subscription.
3	Gender	Indicates the gender of individual customers.
4	Education	Provides information about the educational attainment of the customer.
5	Total loan amount	Total loan amounts taken by a customer.
6	Total repay amount	Total loan repayment amounts by a customer.
7	Loan frequency	Counts the number of loans taken by a customer.

8	Operation type	Indicates the means of the latest payment made by the customer, such as recharge, transfer, or other relevant methods.
9	Active Age	Represents the duration of a customer's connection to the Ethio telecom network.
10	SMS Usage	Indicates the number of text messages sent or received by a user over the mobile network.
11	Data usage	Represents the amount of data consumed by a user when accessing the internet or using online services over the mobile network.
12	Voice usage	Amount of time a user spends making voice calls or engaging in voice communication over the mobile network.
13	Total recharge amount	Represents the cumulative value of all recharges made by a customer.
14	Total consumption	Represent the total consumption by a user. It sums up SMS usage, data usage and voice usage by a customer.
15	Network type	The Network Type attribute indicates the type of network used by the customer, such as 2G, 3G, or 4G.
16	Recharge Frequency	Counts the number of airtime recharge by a customer.

3.5 Development Tools

3.5.1 Hardware Tools

For the research conducted in the thesis, a computer system equipped with an 11th Gen Intel(R) Core(TM) i7-1165G7 processor was used. This processor runs at a base frequency of 2.80GHz and can boost up to 2.70GHz. The system was complemented with 8 GB of installed RAM and operates on a 64-bit Windows 11 Home operating system, specifically designed for x64-based processors. Additionally, for storage purposes, a flash card was employed to ensure the safekeeping and quick access to data.

3.5.2 Software Tools

The term "software tools" refers to a group of computer programs used to develop, support, debug, or maintain other applications and programs.

Table 3.2 Software tools

<u>NO</u>	Software tools	Description
1	Python	Python is a high level and general-purpose programming language that has an abundance of libraries and frameworks that facilitate coding and save development time.
2	Anaconda	Anaconda is an enterprise data science platform that distributes python for machine learning packages
3	Jupyter Notebook	Jupyter Notebook is an open-source web-based application for creating and sharing live code and explanatory text documents.
4	TensorFlow	TensorFlow: an open-source library for fast numerical computing
5	Keras	Keras: a user-friendly deep learning API written in python running on the top of TensorFlow.
6	Scikit-learn	Scikit-learn: is an open-source library in python that contains machine learning algorithms.
7	Colab	Colab: is an open environment for Jupiter Notebook running on the cloud. The environment provides both CPU and GPU runtime.

3.6 Evaluation Metrics

Model quality is assessed using evaluation metrics, which also ensure that the model is operating correctly and at its best. Several evaluation metrics can be utilized to provide both qualitative and quantitative assessments of the model's performance. We can evaluate a given deep-learning model using a variety of evaluation metrics. Confusion matrix, classification accuracy, F-measure, Recall, and ROC curve are examples of common evaluation matrices.

3.6.1 Confusion Matrix

Because the model becomes confused between the two classes, the confusion matrix is one approach of quantifying the performance of supervised machine/deep learning algorithms. It is a 2X2 matrix with classification labels True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN).

Table 3.3 Confusion Matrix

		Predicted	
		Non-defaulter	Defaulter
Actual	Non-defaulter	TN	FP
	Defaulter	FN	TP

- **True Positive (TP):** Number of instances correctly classified as defaulter.
- **True Negative (TN):** Number of instances correctly classified as non-defaulter.
- **False Positive (FP):** Number of non-defaulter instance incorrectly classified as defaulter.
- **False Negative (FN):** Number of defaulter instance incorrectly classified as non-defaulter.

3.6.2 Accuracy

Accuracy is a classification metric that assesses the general accuracy of a model's estimates. The number of correctly identified cases (including True Positives and True Negatives) is divided by the total number of occurrences in the dataset to calculate accuracy. The accuracy metric determines the ratio of accurate predictions to all instances considered (M & M.N, 2015).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

3.6.3 Precision

Precision is a classification metric that measures the proportion of correctly predicted positive cases (True Positives) out of all the predicted positive cases. Essentially, it evaluates the

reliability of a model's positive predictions. Precision is used to determine the proportion of accurately predicted positive patterns in a positive class (M & M.N, 2015).

$$Precision = \frac{TP}{TP + FP}$$

3.6.4 F-score

The F1-score serves as a comprehensive classification metric that harmonizes precision and recall within a singular value, offering an equitable assessment of a model's performance, especially in scenarios characterized by an uneven distribution between positive and negative instances.

$$F - measure = \frac{2 * precision * recall}{precision + recall}$$

CHAPTER FOUR

4. PROPOSED SOLUTION

4.1 Chapter Overview

This chapter provides a comprehensive overview of the model architecture, the tools utilized, and implementation process, highlighting the key elements necessary for credit risk prediction.

4.2 Proposed Model Architecture

We have proposed a deep learning/deep neural network, which allows computational models composed of multiple processing layers to learn data representations with several levels of abstraction. DNN is capable of solving complex problems that necessitate the discovery of hidden patterns in data and/or a thorough understanding of intricate relationships among a significant number of interdependent variables. DNN is capable of solving complex problems that necessitate the discovery of hidden patterns in data and/or a thorough understanding of intricate relationships among a significant number of interdependent variables (Shrestha & Mahmood, 2019). *Figure 4.1* below depicts the proposed solution for our research.

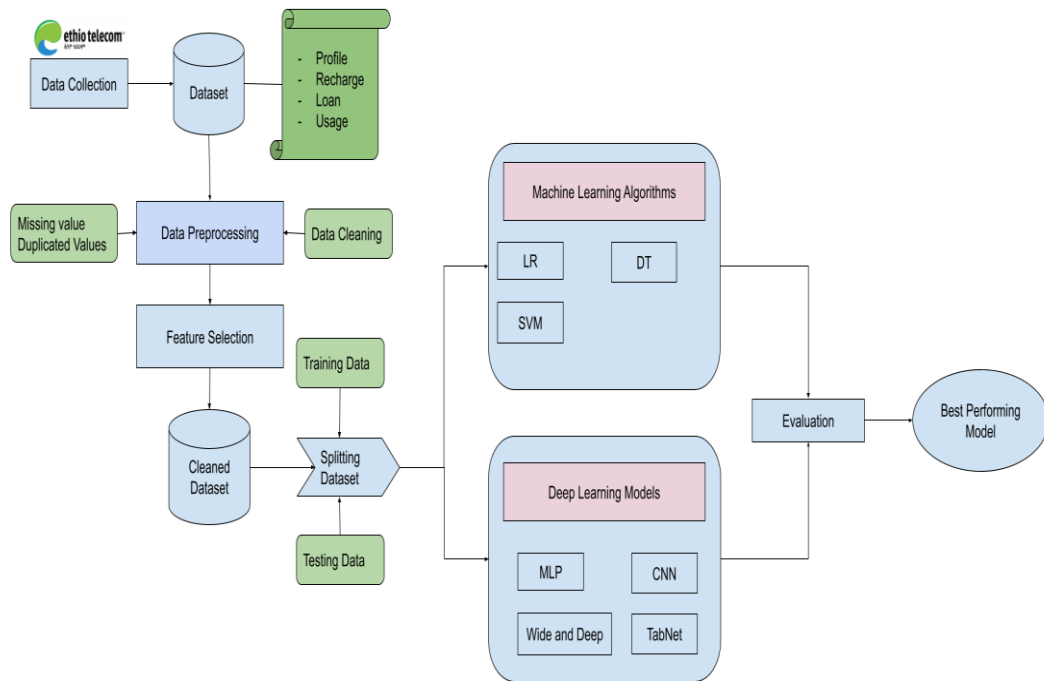


Figure 4.1 Proposed Solution of our research

4.3 Dataset

For the purpose of airtime and package credit risk prediction, the dataset was collected from Ethio telecom. The dataset consists of four separate datasets, each containing over 10 million instances before preprocessing. The dataset was collected across three months in 2023, in May, June, and July. These datasets provide valuable information for analyzing and predicting credit risk in relation to airtime and package credit.

4.4 Data Preprocessing

Before start training the proposed models, it is essential to preprocess each dataset individually to ensure data quality, consistency, and compatibility. The preprocessing steps outlined below will be applied to all datasets to maintain uniformity and facilitate integration.

4.4.1 Data Cleaning

Data cleansing serves as the pivotal first step within the data preprocessing stage. It is a meticulous process that necessitates managing anomalies like missing values, extreme outliers, and unvarying attributes.

In the context of values labeled as NaN (Not a Number), they are removed from the dataset to avert potential inconsistencies they may engender. Outliers, which are defined as values that exhibit a marked deviation from the bulk of the data, are pinpointed and excised to inhibit their potential skewing effect on the analysis.

Moreover, attributes that display consistent value across all instances are discarded. The rationale behind this is that they fail to imbue the analysis with any meaningful or differentiated information.

The datasets also contain erroneous entries such as a customer's age exceeding 150 years or falling below 10 years, and network age attributes surpassing the customer's age. These are clear instances of data inaccuracies that necessitate removal to maintain the integrity and validity of the dataset.

Data Cleaning Algorithm

Input: Unclean dataset

Output: Clean dataset

Start:

1. while(not the end of dataset):
 2. Read row:
 - If row has NaN value:
 - Remove NaN Value
 - If rows has an age of less than 18:
 - Remove row
 - If row has an age of greater than 100:
 - Remove row
 - If row has an age greater than Network Active age:
 - Remove row
3. Return Clean dataset

End

4.4.2 Attribute Selection

Once the data cleaning phase is done, we move on to attribute selection. The data we have comes from four different sets provided by Ethio telecom: Customer Profile, Recharge data, Data Usage and Loan. These datasets together include around 35 different attributes, but not all of these are helpful for predicting credit risk. So, we have to pick out the attributes that are most useful for our analysis and predictions with the help of expert from Ethio telecom.

In this step, we look at each attribute's value and how much it relates to our research goals. If an attribute doesn't give much useful information or doesn't help much with predictions, we leave it out. This helps us make the data easier to handle and can make our model work better.

By choosing only the most useful and relevant attributes, we make our dataset more targeted and easier to understand. This can help improve the efficiency of our analysis and prediction work. This careful selection makes sure we're focusing on the attributes that have the most impact, which can lead to more precise and dependable predictions about airtime and credit risk.

4.4.3 Data Transformation

Data transformation is a process in data preprocessing that involves converting or modifying the original dataset to make it more suitable and meaningful for analysis or modeling tasks. It aims to improve the quality, consistency, and interpretability of the data. It involves the following:

- **Feature Engineering:** Creating new features or attributes from existing ones to capture additional information or improve model performance. It involves the creative process of designing and constructing new features that can better represent the underlying patterns and relationships in the data. For instance, one can create a new feature called "Active age" by calculating the duration between the activation date and the current date. Other feature constructed are Loan frequency and recharge frequency.
- **Normalization:** It refers to the process of adjusting values within a dataset to a common scale, without distorting the differences in the range of values. It involves scaling numerical features so that they have a mean of 0 and a standard deviation of 1.
- **Aggregation:** it is a process that consolidates multiple attributes or values to create a singular, synthesized attribute. For instance, consider the attribute TOTAL_ETB, which might represent a customer's total expenditure in ETB. This is derived by summing up amounts from several other attributes such as OFF_PEAK_AMT_ETB, PEAK_AMT_ETB, and SMS_LOCAL_FEE_ETB. Similarly, the TOTAL_USAGE attribute encapsulates the entirety of a customer's service usage time, by aggregating values from OFF_PEAK_USAGE_MINUTE, PEAK_USAGE_MINUTE, and INTER_USAG_MINUT.

4.4.4 Data Integration

Data integration is the process of combining data from multiple sources or datasets into a unified and consistent format. It involves merging or joining datasets based on common attributes or keys to create a comprehensive and integrated dataset that contains information from all the individual sources. By combining the datasets based on the service number, we can establish a seamless connection between the loan information dataset, mobile customer profile data, customer usage information data, and recharge information data. This integration allows us to analyze the relationships and patterns among various attributes, uncover hidden insights, and develop predictive models that can aid in assessing airtime and package credit risk for Ethio telecom.

4.5 Proposed Model

We have proposed four Deep learning/deep neural network models namely MLP, TabNet, Wide and Deep Network, and CNN to predict airtime and credit risk prediction and we employed Machine learning algorithms namely Logistic Regression (LR), Support Vector Machine (SVM) , and Decision Tree (DT).

4.6 Environmental Setup

Python: chosen as the programming language to implement the proposed method.

Anaconda: Anaconda is an enterprise data science platform that distributes python for machine learning packages.

Pandas: Pandas are a powerful library for data manipulation and analysis.

NumPy: NumPy is a fundamental library for numerical computing in Python.

Scikit-learn is a machine-learning library that provides a comprehensive set of tools for data mining, classification, regression, clustering, dimensionality reduction, and model selection. It offers a uniform interface for implementing and analyzing various machine learning and deep learning algorithms.

TensorFlow: TensorFlow is a prominent open-source deep learning package created by Google. It offers a complete ecosystem for developing and training machine learning models, particularly deep neural networks.

Keras: Keras is a high-level neural network API that can run on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (CNTK). It provides a user-friendly and straightforward interface for constructing and training deep learning models.

4.7 Implantation of the proposed model

In this section, we discuss our journey towards leveraging deep learning models for predicting airtime and credit risk within Ethio telecom. Initially, we establish a foundation using traditional machine learning algorithms such as Logistic Regression (LR), Support Vector Machine (SVM), and Decision Tree (DT). These algorithms serve as our baseline, offering a fundamental understanding and comparative performance metrics. Building upon this foundation, we transition to more advanced neural network architectures, including the Multi-layer Perceptron (MLP), TabNet, Wide and Deep Network, and the Convolutional Neural Network (CNN).

4.7.1 Initiating Experiment

The experiment began by importing the necessary libraries and frameworks required for the implementation of airtime and credit risk prediction using deep learning. Key libraries such as pandas, numpy, and scikit-learn were imported to handle data processing and manipulation tasks effectively. The Keras and TensorFlow libraries were employed for building and training deep learning models.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from joblib import dump, load

# Scikit-Learn imports
from sklearn.model_selection import train_test_split, KFold
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import (
    classification_report, accuracy_score, confusion_matrix,
    precision_score, recall_score, f1_score, roc_curve, auc, roc_auc_score
)
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import LinearSVC

# Tensorflow imports
from tensorflow.keras import layers, models, Model, Input
from tensorflow.keras.layers import (
    Dense, Dropout, concatenate,
    Input, Dense, concatenate, Embedding, Flatten, add
)
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.optimizers import Adam

# Other imports
from pytorch_tabnet.tab_model import TabNetClassifier
```

Figure 4.2 Libraries and frameworks required for our research

To load the dataset, the pandas' library was used to read CSV files representing the combined dataset for airtime and package credit risk prediction which include both default and non-defaulters. The CSV files were loaded into pandas data frames.

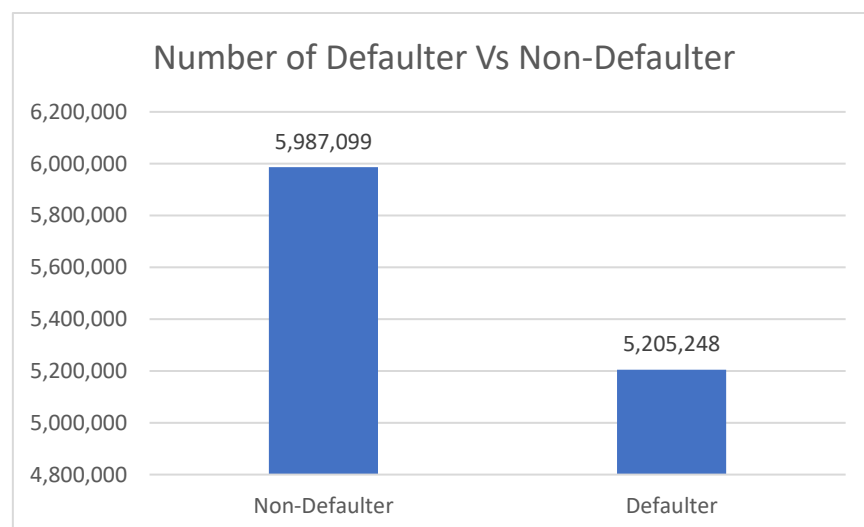
```
recharge = pd.read_csv('data/recharge.csv')
loan = pd.read_csv('data/loan.csv')
profile = pd.read_csv('data/data_profile.csv')
data_usage = pd.read_csv('data/data_profile.csv')
```

Figure 4.3 Loading Dataset

4.7.2 Exploratory Data Analysis

Exploratory data analysis helps uncover underlying patterns, insights, and trends in the dataset. It serves as the initial step to inform the subsequent modeling phase. In our dataset, the total loan amount and the total repay amount present compelling figures. A cursory analysis reveals a sizable gap between these two metrics, implying a considerable level of risk and potential default. When we segregate these figures between 'defaulters' and 'non-defaulters,' the disparity becomes even more telling. The total repay amount by defaulters is significantly less compared to non-defaulters. This could be indicative of a systemic issue or a segment of the customer base that is inherently more risky.

The figure 4.4 show data exploratory analysis for airtime and credit risk prediction.



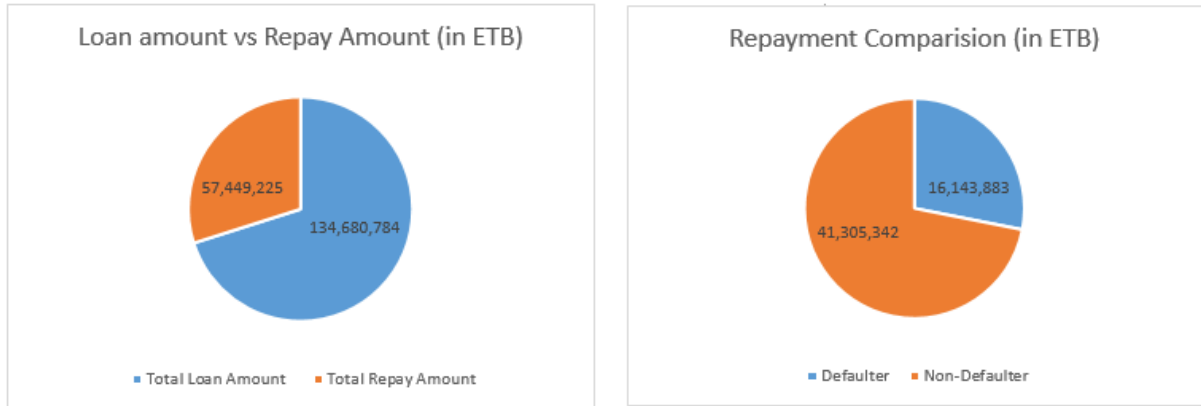


Figure 4.4 Exploratory Data Analysis for airline and credit risk prediction

Given a dataset of over 10 million rows, we chose to focus on a subset of 200,000 unique records, evenly split between defaulters (100,000) and non-defaulters (100,000). One of the primary motivations behind this decision was the inherent processing limitations that come into play when handling data of such magnitude. Engaging with vast datasets demands substantial computational power. Even with advanced systems, processing times can become prolonged, potentially hindering timely data analysis and decision-making. This is especially true when intricate computations or analytics are involved. Moreover, there's the undeniable challenge of storage constraints. The sheer volume of data means that significant storage space is required. Storing, retrieving, and backing up such large datasets can become a logistical challenge, more so if there are restrictions on storage capacity or if swift data retrieval is essential for the project's success. By narrowing our focus to a manageable subset, we are better positioned to ensure efficient processing, as well as effective storage management, laying the groundwork for a more streamlined and productive analytical journey.

4.7.3 Cleaning Dataset

4.7.3.1 Cleaning Customer Recharge

Customer recharge contain four features namely SERV_NO, SGMT_TYPE, DATE_OF_RECHARGE, and RECHARGE_AMOUNT. The figure below shows sample datasets of customer recharge before cleaning:

SERV_NO	SGMT_TYPE	DATE_OF_RECHARGE	RECHARGE_AMOUNT
1142301564	0	2023-05-01 07:19:38	5
1142300304	0	2023-05-01 07:19:38	10
1142300304	0	2023-05-01 07:19:38	10
1066012039	0	2023-05-01 07:19:38	5
1066012039	0	2023-05-01 07:19:38	5
...	...	...	...
1052111700	0	2023-05-01 07:19:38	10
1075947661	0	2023-05-01 07:19:38	25
1075941778	0	2023-05-01 07:19:38	10
1075943780	0	2023-05-01 07:19:38	15
1075943780	0	2023-05-01 07:19:38	15

Figure 4.5 Sample datasets of customer recharge before preprocessing

To clean the customer recharge dataset, one begins by removing duplicate rows to ensure the uniqueness of each record. The column SGMT_TYPE is then dropped as it might not hold significant relevance for the upcoming analysis. During the cleaning process, it's imperative to handle missing data; hence, rows containing NaN values are also dropped. The DATE_OF_RECHARGE feature, potentially seen as redundant for the purpose of the subsequent study, is discarded. Lastly, the dataset is further enhanced by adding two new columns. One named 'Frequency', which indicates the regularity of a customer's recharge habits, and the other, 'TOTAL_RECHARGE', summing up the overall amount a customer has recharged over time.

```
df['TOTAL_RECHARGE'] = df.groupby(['SERV_NO'])['RECHARGE_AMOUNT'].transform('sum')
df['frequency'] = df['SERV_NO'].map(df['SERV_NO'].value_counts())
```

Figure 4.6 Cleaning Recharge

The figure below shows recharge information after cleaned:

SERV_NO	TOTAL_RECHARGE	frequency
1142301564	25.0	4.0
1142300304	25.0	2.0
1066012039	75.0	10.0
1066012026	350.0	18.0
1066007403	10.0	2.0
...	...	...
1037320508	160.0	15.0
1037320508	160.0	15.0
1096770601	140.0	10.0
1142349454	115.0	9.0
1142350030	240.0	11.0

Figure 4.7 Sample datasets of customer recharge after preprocessing

4.7.3.2 Cleaning Customer Profile

Customer profile dataset contain several columns, the figure below shows the profile data before preprocessing:

SERV_NO	ACTIVATE_DATE	CITY_NAME	CUST_LEVEL_NAME	CATEGORY_NAME	SUB_CATEGORY_NAME	NET_BUSI_TYPE_NAME	OFFER_NAME	SGMT_NAME	STATUS_NAME	CUST_AGE	GENDER_NAME	EDUCATION_NAME
133539466	22/10/2020	Bole Sub City	NaN	Residential	Residential	LTE	4G LTE Data (Prepaid)	Prepaid	Active	42.0	Male	Illiterate
133540623	07/01/2021	Lideta sub city	NaN	Residential	Residential	LTE	4G LTE Data (Prepaid)	Prepaid	Active	52.0	Male	Bachelor's Degree
133605186	19/10/2020	Arada sub city	NaN	Residential	Residential	WCDMA	3G DATA(Prepaid)	Prepaid	Active	44.0	Male	Bachelor's Degree
134189243	21/04/2021	Yeka sub city	NaN	Residential	Residential	GSM	NaN	Postpaid	Active	43.0	Male	Illiterate
136046656	27/02/2010	NaN	NaN	Residential	Residential	GSM	NaN	Prepaid	Active	25.0	NaN	NaN
--	--	--	--	--	--	--	--	--	--	--	--	--
1035001670	30/11/2018	North shewa zone	NaN	Residential	Residential	WCDMA	WCDMA 3G (Prepaid)	Prepaid	Active	30.0	Male	Secondary school
1035001672	06/12/2018	North shewa zone	NaN	Residential	Residential	WCDMA	WCDMA 3G (Prepaid)	Prepaid	Active	27.0	Male	Illiterate
1035001673	04/03/2021	Bale Zone	NaN	Residential	Residential	WCDMA	WCDMA 3G (Prepaid)	Prepaid	Active	27.0	Male	Secondary school
1035001673	04/03/2021	Bale Zone	NaN	Residential	Residential	WCDMA	WCDMA 3G (Prepaid)	Prepaid	Active	31.0	Male	Illiterate
1035001678	03/12/2018	North shewa zone	NaN	Residential	Residential	WCDMA	WCDMA 3G (Prepaid)	Prepaid	Active	27.0	Male	Secondary school

Figure 4.8 Sample datasets of customer profile before preprocessing

Cleaning the customer profile dataset is a multifaceted task. Initially, the dataset undergoes a dimensionality reduction by dropping several columns. Features such as CITY_NAME, CUST_LEVEL_NAME, CATEGORY_NAME, SUB_CATEGORY_NAME, OFFER_NAME, SGMT_NAME, and STATUS_NAME are discarded, albeit with specific criteria applied beforehand. For instance, only entries tagged 'Residential' in the CATEGORY_NAME, 'prepaid' in the SGMT_NAME, and 'active status' in STATUS_NAME are retained before the respective columns are removed. The cleaning process also emphasizes data integrity. Rows where the NETWORK_AGE exceeds the CUST_AGE are dropped. Additionally, if a customer's age is not at least 18 years greater than their Network age, it's deemed inappropriate and thus discarded. The dataset is also refined to cater to a specific age bracket, dropping rows containing customers aged below 18 or exceeding 70. Duplicate entries are eliminated, ensuring the dataset's uniqueness. The dataset's depth is further enriched by introducing the 'NETWORK_AGE' feature, which is derived from the ACTIVATE_DATE. Lastly, any row marred with NaN values is excluded, ensuring the dataset's consistency and completeness. The figure below shows code snippets for preprocessing

```
: profile.drop_duplicates(inplace=True)
profile = profile.drop("CUST_LEVEL_NAME", axis=1)
profile = profile.drop("CITY_NAME", axis=1)
profile = profile.drop("OFFER_NAME", axis=1)
profile = profile[profile['STATUS_NAME'] == 'Active']
profile = profile.drop("STATUS_NAME", axis=1)
profile = profile[profile['CATEGORY_NAME'] == 'Residential']
profile = profile.drop("CATEGORY_NAME", axis=1)
profile = profile.drop('SUB_CATEGORY_NAME', axis=1)
profile = profile.drop('CITY_NAME', axis=1)
profile = profile.drop('CUST_LEVEL_NAME', axis=1)
profile = profile.drop('OFFER_NAME', axis=1)
profile = profile[profile['SGMT_NAME'] == 'Prepaid']
profile = profile.drop('SGMT_NAME', axis=1)

from datetime import datetime
current_year = datetime.today().year
profile['ACTIVATE_DATE'] = pd.to_datetime(profile['ACTIVATE_DATE'], format='%d/%m/%Y')
profile['NETWORK_AGE'] = current_year - profile['ACTIVATE_DATE'].dt.year
```

Figure 4.9 Customer Profile Data Preprocessing

The figure below shows customer profile data after preprocessed

SERV_NO	NET_BUSI_TYPE_NAME	CUST_AGE	GENDER_NAME	EDUCATION_NAME	NETWORK_AGE
133539466	LTE	42.0	Male	Illiterate	3.0
133540623	LTE	52.0	Male	Bachelor's Degree	2.0
133605186	WCDMA	44.0	Male	Bachelor's Degree	3.0
1035000001	LTE	35.0	Male	Bachelor's Degree	5.0
1035000005	LTE	27.0	Male	Bachelor's Degree	5.0
***	***	***	***	***	***
1146545100	WCDMA	23.0	Female	Illiterate	4.0
1146545102	WCDMA	43.0	Male	Secondary school	0.0
1146545104	WCDMA	33.0	Male	Illiterate	4.0
1146545107	LTE	36.0	Male	Secondary school	4.0
1146545108	LTE	38.0	Female	Secondary school	4.0

Figure 4.10 Sample Customer Profile data after preprocessing

The data usage dataset, encompassing many 21 features, requires a series of cleaning steps to make it more manageable and insightful. The figure below shows sample dataset before preprocessing:

SERV_NO	NETWORK_TYPE	ACTIVATE_DATE	STATUS_NAME	OFFER_NAME	OFF_PEAK_USG_MINUTE	OFF_PEAK_AMT_ETB	PEAK_USG_MINUTE	PEAK_AMT_ETB	INTER_USG_MINUTE	INTER_AMT_ETB	SMS_LOCAL_USAGE	SMS_LOCAL_FEE_ETB	SMS_INTER_USAGE	SMS_INTER_FEE_ETB	SMS_PRIMA_USAGE	SMS_PRIMA_FEE_ETB	DATA_USAGE_MB	DU
113442501	WCDMA	2.022100e+13	Active	WCDMA_3G (Prepaid_SIM_Starter_Bundle	0	0.000000	0	0.000000	0	0.0	0	0.000000	0	0	0	0.0	0	
113443355	WCDMA	2.022080e+13	Active	WCDMA_3G (Prepaid_SIM_Starter_Bundle	1676	87974.762609	10809	524543.470261	0	0.0	5	5215.043478	0	0	0	0.0	0	
113445407	WCDMA	2.022072e+13	Active	WCDMA_3G (Prepaid_SIM_Starter_Bundle	1431	85524.547826	12942	516286.956522	0	0.0	0	0.000000	0	0	0	0.0	0	
113445855	WCDMA	2.023021e+13	Active	WCDMA_3G (Prepaid_SIM_Starter_Bundle	573	29180.000000	18997	255325.217391	0	0.0	5	1042.608696	0	0	0	0.0	0	
113447700	WCDMA	2.023012e+13	Active	WCDMA_3G (Prepaid_SIM_Starter_Bundle	1755	59766.080957	11162	420154.762609	0	0.0	2	0.000000	0	0	0	0.0	564288	
--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	
145116548	WCDMA	2.019123e+13	Active	WCDMA_3G (Prepaid)	64	3264.347826	1273	87145.217391	0	0.0	0	0.000000	0	0	0	0.0	0	
145116623	WCDMA	2.020011e+13	Active	WCDMA_3G (Prepaid)	641	37838.260870	3315	217074.762609	0	0.0	0	0.000000	0	0	0	0.0	0	
145116730	WCDMA	2.020010e+13	Active	WCDMA_3G (Prepaid)	532	24366.080957	2857	81422.608696	0	0.0	3	1042.608696	0	0	0	0.0	204800	
145117040	WCDMA	2.020013e+13	Active	WCDMA_3G (Prepaid)	254	0.000000	17056	49062.608696	0	0.0	21	1042.608696	0	0	0	0.0	6451200	
145117114	WCDMA	2.020010e+13	Active	WCDMA_3G (Prepaid)	0	0.000000	0	0.000000	0	0.0	0	0.000000	0	0	0	0.0	0	

Figure 4.11 Sample Customer Data usage before preprocessing

Initially, specific columns are selectively pruned. SGMT_NAME is trimmed down to only include entries of 'prepaid customers' and similarly, the STATUS_NAME is refined to encapsulate only 'active users'. Once these selections are made, the columns, along with OFFER_NAME and DATA_MONTH, are entirely removed from the dataset. Subsequent to these column-specific operations, any rows afflicted with NaN values are discarded to maintain data accuracy. Lastly, to consolidate the dataset, identical data usage instances, like SMS usage

for a particular customer, are aggregated. After summing these usages together, redundant rows are eliminated, leaving behind a concise, aggregated record for each customer.

The figure below shows preprocessing steps of data usage

```
data_usage = data_usage[data_usage['SGMT_NAME'] == 'Prepaid']
data_usage = data_usage.drop("SGMT_NAME", axis=1)
data_usage = data_usage[data_usage['STATUS_NAME'] == 'Active']
data_usage = data_usage[data_usage['OFFER_NAME'] != 'WCDMA 3G (Postpaid)']
data_usage = data_usage[data_usage['OFFER_NAME'] != '5G Mobile (prepaid)']
data_usage = data_usage.drop("OFFER_NAME", axis=1)
data_usage.dropna(inplace=True)
data_usage['TOTAL_ETB'] = data_usage['OFF_PEAK_AMT_ETB'] + data_usage['PEAK_AMT_ETB'] + data_usage['INTER_AMT_ETB']
+ data_usage['SMS_LOCAL_FEE_ETB'] + data_usage['SMS_INTER_FEE_ETB'] + data_usage['SMS_PRPM_FEE_ETB'] + data_usage['DATA_REVENUE_ETB']

data_usage['TOTAL_MINUTE'] = data_usage['OFF_PEAK_USG_MINUTE'] + data_usage['PEAK_USG_MINUTE'] + data_usage['INTER_USG_MINUTE']
data_usage = data_usage.drop('OFF_PEAK_USG_MINUTE', axis=1)
data_usage = data_usage.drop('OFF_PEAK_AMT_ETB', axis=1)
data_usage = data_usage.drop('PEAK_USG_MINUTE', axis=1)
data_usage = data_usage.drop('PEAK_AMT_ETB', axis=1)
data_usage = data_usage.drop('INTER_USG_MINUTE', axis=1)
data_usage = data_usage.drop('INTER_AMT_ETB', axis=1)
data_usage = data_usage.drop('SMS_LOCAL_FEE_ETB', axis=1)
data_usage = data_usage.drop('SMS_INTER_FEE_ETB', axis=1)
data_usage = data_usage.drop('SMS_PRPM_FEE_ETB', axis=1)
data_usage = data_usage.drop('DATA_REVENUE_ETB', axis=1)
data_usage['TOTAL_SMS'] = data_usage['SMS_LOCAL_USAGE'] + data_usage['SMS_INTER_USAGE'] + data_usage['SMS_PRPM_USAGE']
data_usage = data_usage.drop('SMS_LOCAL_USAGE', axis=1)
data_usage = data_usage.drop('SMS_INTER_USAGE', axis=1)
data_usage = data_usage.drop('SMS_PRPM_USAGE', axis=1)
data_usage = data_usage[data_usage['DATA_USAGE_MB'] + data_usage['TOTAL_MINUTE'] + data_usage['TOTAL_ETB'] + data_usage['TOTAL_SMS'] != 0]
data_usage = data_usage[data_usage['DATA_USAGE_MB'] + data_usage['TOTAL_MINUTE'] + data_usage['TOTAL_ETB'] + data_usage['TOTAL_SMS'] != 0]
data_usage['TOTAL_ETB'] = data_usage['TOTAL_ETB'] / 10000
data_usage['DATA_USAGE_MB'] = data_usage['DATA_USAGE_MB'] / 1024/1024
```

Figure 4.12 Data usage preprocessing

The figure below shows data usage after preprocessed:

SERV_NO	NETWORK_TYPE	ACTIVATE_DATE	DATA_USAGE_MB	TOTAL_MINUTE	TOTAL_ETB	TOTAL_SMS	ACTIVE_YEAR_LENGTH
I113443355	WCDMA	20220802	0	12585	62	5	1
I113445407	WCDMA	20220719	0	14373	101	0	1
I113445855	WCDMA	20230206	0	14570	29	5	0
I113447700	WCDMA	20230123	5	12917	49	2	0
I113454003	WCDMA	20220706	0	1876	13	1	1
...	...	...	...	...	...	...	...
I145116439	WCDMA	20200109	0	2426	15	0	3
I145116548	WCDMA	20191230	0	1337	10	0	4
I145116623	WCDMA	20200109	0	3956	26	0	3
I145116730	WCDMA	20200104	0	3389	11	3	3
I145117040	WCDMA	20200130	6	17320	7	21	3

Figure 4.13 Data usage after preprocessing

4.6.3.4 Cleaning Loan dataset

In the quest to preprocess the loan dataset, as visualized in *Figure 4.14*, some strategic cleaning actions are undertaken. Initially, some columns that might be seen as redundant or superfluous, like `OPER_DATE`, are removed from the dataset. This is followed by the addition of value through feature engineering, where a new feature, 'loan frequency', is conceived and integrated. This new column likely encapsulates the regularity of loan applications or disbursements for each entity. The final touch in this cleaning exercise is the removal of duplicate rows, ensuring that each entry in the dataset remains unique and free from repetitions. Through these steps, the loan dataset is primed for further analysis or modeling activities.

SERV_NO	OPER_DATE	OPER_TYPE	INIT_LOAN_AMT	INIT_LOAN_POUNDA	LOAN_AMT	LOAN_POUNDA	REPAY_AMT	REPAY_POUNDA	ENTRY_DATE
1073076322	20230401043546	R	15.0	1.5	0.0	0.0	15.0	1.5	20230401043546
1075611773	20230401115711	R	15.0	1.5	0.0	0.0	5.0	1.5	20230401115711
1075611773	20230401121305	L	15.0	1.5	15.0	1.5	0.0	0.0	20230401121305
1075611773	20230401115557	R	15.0	1.5	5.0	1.5	10.0	0.0	20230401115557
1121475566	20230401093413	R	15.0	1.5	5.0	1.5	10.0	0.0	20230401093413
...	...	...	...	...	...	...	...	...	...
1054492677	20230430041552	R	10.0	1.0	0.0	0.0	10.0	1.0	20230430041552
1054492677	20230430152709	L	15.0	1.5	15.0	1.5	0.0	0.0	20230430152709
1047872942	20230430190355	L	50.0	5.0	50.0	5.0	0.0	0.0	20230430190355
1118697538	20230430193311	T	5.0	0.5	4.0	0.5	1.0	0.0	20230430193311
1046544913	20230430071855	L	25.0	2.5	25.0	2.5	0.0	0.0	20230430071855

Figure 4.14 Loan data before preprocessing

```
loan = loan.drop("OPER_DATE", axis=1)
loan['LOAN_FREQ'] = loan.groupby('SERV_NO')['SERV_NO'].transform('count')
loan['ENTRY_DATE'] = loan['ENTRY_DATE'].astype(str).str[:8].astype(int)
loan.drop_duplicates(inplace=True)
```

Figure 4.15 Preprocessing loan data

Following the separate analysis of our datasets, the integration of these disparate data sources is a critical step in our methodology. This combination is critical because it serves as the cornerstone for our deep learning algorithms, allowing us to generate meaningful forecasts about airtime and package credit risk.

To accomplish this, common attributes that act as bridges across these databases must be identified. The "Service Number" (`SERV_NO`) attribute serves as the pivot in our context, simply connecting our datasets.

4.7.4 Machine Learning Model implementation

4.7.4.1 Logistic Regression

We implemented the "Logistic Regression" method on our data using the "LogisticRegression" tool from the scikit-learn library. After importing the necessary class, we created a new Logistic Regression model. We then trained this model using our training data, teaching it with past examples. Once it was trained, we tested its predictions on both our training and testing datasets. Finally, we saved these predictions to evaluate the model's performance later.

```
X = df.drop(['class', 'SERV_NO'], axis=1)
y = df['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

lr = LogisticRegression(max_iter=1000)
lr.fit(X_train, y_train)

lr_predictions = lr.predict(X_test)

# Calculating metrics
accuracy = accuracy_score(y_test, lr_predictions)
precision = precision_score(y_test, lr_predictions)
recall = recall_score(y_test, lr_predictions)
f1 = f1_score(y_test, lr_predictions)

# Printing results
print("Metrics for Logistic Regression:")
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1 Score: {f1:.4f}")
```

Figure 4.16 Implementation of Logistic Regression

4.7.4.2 Decision Tree

We started our experiment with the Decision Tree by making a new DecisionTreeClassifier model. After creating this model, we trained it using our training data, which includes the features (X_train) and the labels (y_train). The training began when we used the 'fit' method.

After that, we tested the model on both the training and testing data to see how accurate it was and how well it remembered the information.

```
from sklearn.tree import DecisionTreeClassifier

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

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

dt = DecisionTreeClassifier()
dt.fit(X_train, y_train)

dt_predictions = dt.predict(X_test)

# print("Accuracy for Decision Tree:", accuracy_score(y_test, dt_predictions))

print("Accuracy:", accuracy_score(y_test, dt_predictions))
print("Confusion Matrix:\n", confusion_matrix(y_test, dt_predictions))
print("Precision:", precision_score(y_test, dt_predictions))
print("Recall:", recall_score(y_test, dt_predictions))
print("F-score:", f1_score(y_test, dt_predictions))
```

Figure 4.17 Implementation of Rando Decision Tree

4.7.4.3 Support Vector Machine

We started testing the Support Vector Machine (SVM) by making a new LinearSVC model. After setting it up, we trained the model using our training data, which has the features (X_train) and the labels (y_train).

```
X = df.drop(['class', 'SERV_NO'], axis=1)
y = df['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Training the SVM
svm_model = LinearSVC(random_state=42)
svm_model.fit(X_train, y_train)

# Predictions
y_pred = svm_model.predict(X_test)

# Evaluation
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Precision:", precision_score(y_test, y_pred))
print("Recall:", recall_score(y_test, y_pred))
print("F-score:", f1_score(y_test, y_pred))
```

Figure 4.18 Implementation of Support Vector Machine

4.7.5 Deep Learning Model Implementation

4.7.5.1 MLP Model Implementation

After important libraries are imported and preprocessed data is loaded into Jupyter Notebook, we have defined the required parameter and trained our MLP model. The model consists of an input layer and three hidden layers followed by an output layer. Immediately after the input

layer, a dropout layer with a rate of 0.5 is introduced to reduce the risk of overfitting. The subsequent hidden layers consist of 64 and 32 neurons, respectively, with each layer utilizing the ReLU activation function. After each of these hidden layers, a dropout layer with a rate of 0.5 is added. The output layer is made up of a single neuron with a sigmoid activation function, making it appropriate for binary classification tasks. The model is compiled using the Adam optimizer, with the binary crossentropy loss function, indicating its use for a binary classification problem. Accuracy is chosen as the metric to evaluate the model's performance. For training, the model uses a batch size of 32 and is trained for 50 epochs. Additionally, 20% of the training data is set aside for validation to monitor the model's performance on unseen data during training. The verbose parameter set to 1 ensures that detailed logs of the training process are displayed.

The following Figure shows the implementation of the MLP model from defining parameters to training the model.

```
X = df.drop(['class', 'SERV_NO'], axis=1) # Drop target variable and unique identifier
y = df['class']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = Sequential()
model.add(Dense(128, input_dim=X_train.shape[1], activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(32, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

model.fit(X_train, y_train, epochs=50, batch_size=32, validation_split=0.2, verbose=1)
```

Figure 4.19 MLP Model Definition

4.7.5.2 TabNet Implementation

After dividing the dataset into training and validation sets, with 20% of the data reserved for validation. Following this, a TabNet classifier, a deep learning model known for its efficacy in tabular data tasks, is defined. The model is trained on the training data, with both training and validation sets provided for evaluation during training. The Area Under the Curve (AUC) is chosen as the evaluation metric, offering insights into the model's discriminative power. The training process is set to a maximum of 1000 epochs, but an early stopping mechanism is incorporated with a patience of 20 epochs. This means that if the model's validation performance

doesn't improve for 20 consecutive epochs, training will cease, preventing potential overfitting and saving computational resources. The following figures shows TabNet implementations.

```
: # Splitting data into train and validation sets
train, valid = train_test_split(df, test_size=0.2, random_state=42)

# Separating the target variable
target = 'class'
train_X = train.drop(columns=[target])
train_y = train[target].values
valid_X = valid.drop(columns=[target])
valid_y = valid[target].values

# Define TabNet model
model = TabNetClassifier()

# Train the model
model.fit(
    train_X.values, train_y,
    eval_set=[(train_X.values, train_y), (valid_X.values, valid_y)],
    eval_name=['train', 'valid'],
    eval_metric=['auc'],
    max_epochs=1000, patience=20)
```

Figure 4.20 TabNet model implementation

4.7.5.3 Wide and Deep Network Implementation

After splitting the dataset into training and testing subsets, the code transforms categorical columns such as 'OPER_TYPE', 'NETWORK_TYPE', 'GENDER_NAME', and 'EDUCATION_NAME'. Leveraging the LabelEncoder, each of these categorical columns is transformed; it's fitted on the training data and then applied to both the training and test datasets. The architecture of the Wide & Deep model requires separate inputs for its wide and deep components. The wide component, adept at memorization, primarily uses categorical data. Thus, datasets like `wide_input_train` and `wide_input_test` are extracted from the encoded categorical columns. In contrast, the deep component, designed to unravel intricate patterns and interactions, relies on numerical data. For this, the code identifies all relevant numerical columns, excluding specific ones like 'SERV_NO', 'class', and the aforementioned categorical columns. This differentiation leads to the creation of the `deep_input_train` and `deep_input_test` datasets, representing the numerical inputs for the model's training and testing phases.

The wide component is elegantly designed, consisting of an input layer directly connected to its output through a dense layer, using a sigmoid activation function tailored for binary classification tasks. The deep component starts with an input layer, followed by three dense layers with 128, 64, and 32 neurons, respectively. After each dense layer (except the final one), a dropout layer with a rate of 0.5 is added, serving to reduce the risk of overfitting. All these

dense layers in the deep component employ the ReLU activation function. To harness the strengths of both components, their outputs are amalgamated using the concatenate function. This merged data is then directed to a final dense layer, which also features a sigmoid activation, to produce the model's ultimate output.

Before training, the categorical columns are label-encoded. The wide component takes in categorical features, while the deep component processes numerical features. After finalizing the architecture, the model is compiled using the Adam optimizer with the binary cross-entropy loss, indicative of its application to a binary classification problem. Accuracy remains the chosen metric for evaluation.

The model is trained using the prepared datasets with a batch size of 32 over 30 epochs. Validation data is also provided to monitor the model's performance on unseen data during training. Upon completion of training, a message indicates the successful training of the Wide & Deep Network.

```
from tensorflow.keras.layers import Dropout

# Split the dataset into train and test sets
train, test = train_test_split(df, test_size=0.2, random_state=42)
y_train = train['class']
y_test = test['class']

categorical_cols = ['OPER_TYPE', 'NETWORK_TYPE', 'GENDER_NAME', 'EDUCATION_NAME']
encoders = {}
for col in categorical_cols:
    le = LabelEncoder()
    train[col] = le.fit_transform(train[col])
    test[col] = le.transform(test[col])
    encoders[col] = le

wide_input_train = train[categorical_cols]
wide_input_test = test[categorical_cols]

numerical_cols = df.drop(columns=['SERV_NO', 'class'] + categorical_cols).columns.tolist()
deep_input_train = train[numerical_cols]
deep_input_test = test[numerical_cols]

# Define model structure
wide_input = Input(shape=(wide_input_train.shape[1]), name='wide_input')
wide_output = Dense(1, activation='sigmoid')(wide_input)

deep_input = Input(shape=(deep_input_train.shape[1]), name='deep_input')
deep_layer = Dense(128, activation='relu')(deep_input)
deep_layer = Dropout(0.5)(deep_layer) # Adding dropout layer with 50% dropout rate
deep_layer = Dense(64, activation='relu')(deep_layer)
deep_layer = Dropout(0.5)(deep_layer)
deep_layer = Dense(32, activation='relu')(deep_layer)
deep_layer = Dropout(0.5)(deep_layer)
deep_output = Dense(1, activation='sigmoid')(deep_layer)

combined = concatenate([wide_output, deep_output])
final_output = Dense(1, activation='sigmoid')(combined)

wide_deep_model = Model(inputs=[wide_input, deep_input], outputs=final_output)
wide_deep_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train
history = wide_deep_model.fit([wide_input_train, deep_input_train], y_train, validation_data=([wide_input_test, deep_input_test], y_test),
    epochs=30, batch_size=32)
print("Wide & Deep Network trained.")
```

Figure 4.21 Wide and Deep Network Implementation

4.7.5.4 CNN Model Implementation

The model adopts a sequential structure, meaning layers are organized linearly. It starts with a 2D convolutional layer that employs 128 filters, each of dimensions 3x3. This layer, featuring the ReLU activation function, processes the input data of shape (4, 4, 1). After the convolution

operation, a max-pooling layer with a pool size of 2x2 is integrated. This layer's primary role is to diminish the spatial dimensions, facilitating computational efficiency and warding off overfitting. Subsequent to the pooling, a dropout layer with a rate of 0.5 is introduced, further aiding in the prevention of overfitting.

The data then undergoes a flattening operation through a flattening layer, which converts the 2D matrix into a 1D vector, priming it for fully connected layers (dense layers). Following the flattening, the model includes a dense layer with 64 neurons using the ReLU activation function, succeeded by another dropout layer with a rate of 0.5. This setup is replicated with a subsequent dense layer with 32 neurons, again paired with the ReLU activation, and yet another dropout layer at the same rate. These layers are dedicated to discerning intricate patterns and relationships from the previously processed data.

Concluding the architecture, there's a final dense layer with a single neuron, employing a sigmoid activation function. Given the properties of the sigmoid function, this configuration suggests that the model is tailored for a binary classification task, determining one out of two potential outcomes. For the training phase, the 'adam' optimizer is harnessed to iteratively refine the weights. The choice of the 'binary_crossentropy' loss function reinforces the model's alignment with binary classification objectives. During training, the model's performance is evaluated using the 'accuracy' metric.

Finally, the training process is executed on the dataset `X_train` and its corresponding labels `y_train` for 50 epochs. The training data is split, with 10% reserved for validation purposes, ensuring the model's generalization capabilities. The model is trained using batches of 32 samples at a time, a common practice to both ensure stable convergence and to expedite the training process.

The following figure show CNN model implementations.

```

X = df.drop(['SERV_NO', 'class'], axis=1)
y = df['class']
num_features = X.shape[1]
X_resaped = np.reshape(X.values, (X.shape[0], 4, 4, 1))

X_train, X_test, y_train, y_test = train_test_split(X_resaped, y, test_size=0.2, random_state=42)
model = models.Sequential()
model.add(layers.Conv2D(128, (3, 3), activation='relu', input_shape=(4, 4, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Dropout(0.5)) # Dropout Layer after MaxPooling
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dropout(0.5)) # Dropout Layer after first Dense Layer
model.add(layers.Dense(32, activation='relu'))
model.add(layers.Dropout(0.5)) # Dropout Layer after second Dense Layer
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])

history = model.fit(X_train, y_train, epochs=50, batch_size=32, validation_split=0.1)

```

Figure 4.22 CNN model implementation

4.8 Parameter Tuning

Hyperparameter tuning is a systematic process that seeks the ideal combination of hyperparameters to optimize a model's performance. By carefully adjusting these settings, we aim to fine-tune our models to achieve peak efficiency. The primary motivation behind this approach is to strike a balance between a model's intricacy and its ability to generalize across various datasets. The overarching goal is twofold: to prevent overfitting and underfitting. Overfitting arises when a model gets excessively tailored to its training data, reducing its capability to perform well on new, unseen data. Conversely, underfitting occurs when the model remains too simplistic, neglecting to capture crucial patterns in the data. By refining hyperparameters, we ensure that the model can understand the data's complexities while retaining its broad applicability to diverse datasets. The following Table 4.1 summarizes hyper parameters used for all experiments.

We adopted a manual methodology for hyperparameter selection across our proposed architectures, encompassing CNN, MLP, and the wide and deep network. Through meticulous experimentation, we delved deeply into the nuances of these hyperparameters, detailed as follows:

- **Batch Size:** This refers to the number of training samples processed in a single iteration. It plays a pivotal role in influencing both the convergence rate and overall training efficiency of the model.

- *Epochs*: This parameter signifies the number of complete passes through the entire dataset. It's pivotal in weight assignment, and its optimal value is essential to circumvent the pitfalls of both underfitting and overfitting.
- *Activation Function*: A foundational aspect of neural networks, it dictates the computation of a neuron's or layer's output.
- *Optimizer*: Central to refining a model, the optimizer tweaks the weight parameters with an aim to reduce the loss function, ensuring the model continually improves its predictive accuracy.
- *Loss Function*: Serving as the yardstick of model performance during training, the loss function quantifies the deviation between predicted outcomes and actual values, guiding the model towards optimization.
- *Metrics*: While these don't influence the training process directly, metrics provide an invaluable lens to evaluate a model's real-world performance, ensuring its predictions align with expected outcomes.

Table 4.1: Summaries of hyper parameters

Parameters	Models			
	MLP	TabNet	Wide and Deep Network	CNN
Batch size	32	1024 (default)	32	32
Epochs	50	Max of 1000 (early stopping)	30	50
Activation Function	ReLu, sigmoid	Relu	Sigmoid, relu	Relu, sigmoid
Optimizer	adam	adam	adam	Adam
Loss Function	binary_crossentropy	binary_crossentropy	binary_crossentropy	binary_crossentropy
Metrics	Accuracy	AUC	Accuracy	accuracy

CHAPTER FIVE

5. RESULT AND DISCUSSION

In this chapter, we present the outcomes of the research experiments utilizing both 10-fold cross-validation and a separate test data validation approach. The assessment report offers a thorough analysis of the model's performance through the examination of crucial metrics, including accuracy, precision, recall, and F1-score. Furthermore, this chapter encompasses visual representations of the confusion matrix, enhancing comprehension of the model's classification skill.

5.1 Result of Machine Learning Models

Before transitioning to the complexities of deep learning algorithms tailored for predicting airline and credit risk, we first rooted our research in the foundational techniques provided by traditional machine learning algorithms. Among the variety of machine learning techniques available, we settled on three predominant models: Logistic Regression (LR), Decision Tree (DT), and Support Vector Machine (SVM).

Logistic Regression stands out in its effective for binary classification tasks. Its strength shines particularly when dealing with datasets, such as ours, that encompass a blend of numerical and categorical attributes. It possesses an innate capability to evaluate and assign weightage to features, estimating their relative significance to the predictive outcome. Decision Trees naturally cater to both numerical and categorical data. Their structure a composite of nodes denoting features and branches indicating decision rules provides an intuitive mapping of data relationships and interactions. Support Vector Machine is renowned for its efficacy in higher-dimensional spaces. Given the multifaceted nature of our dataset, SVM's approach of formulating hyperplanes to segregate distinct classes becomes invaluable.

A systematic evaluation of their performance was conducted to understand the efficacy of these algorithms in the context of the aforementioned tasks. In doing so, two predominant validation techniques were chosen to ensure that the evaluation process was robust and free from biases: 10-fold cross-validation and a separate test data approach, which allocated 80% of the data for training and the remaining 20% for testing purposes. The choice of these two validation techniques was intentional. The 10-fold cross-validation method, commonly employed in

machine learning, partitions the dataset into ten subsets. A model is trained on nine of these subsets, while the tenth is used for validation. This process is repeated ten times, rotating the validation set each time. This method provides a more holistic evaluation of the model's performance, reducing the risk of overfitting and ensuring the results are less susceptible to the variances of a specific data split.

On the other hand, the separate test data approach divides the dataset into training and testing segments, typically allowing a larger chunk for training (in our case, 80%). Such a technique offers insights into how the model would perform on entirely unseen data, simulating a real-world scenario where predictions are made on new data points.

The meticulous preparation of our dataset was paramount to the study. We partitioned our data with a significant 80% reserved for training, allowing the models to familiarize themselves with underlying patterns and intricacies. Conversely, the remaining 20% was used for testing, granting us the opportunity to evaluate each model's capability to generalize and make predictions on unseen data.

Beyond the standard train-test split, we implemented a 10-fold cross-validation. This enhanced methodology segmented the training data into ten parts or "folds." In each iteration, nine parts were used for training while the tenth part was used for validation. By rotating the validation fold in each iteration, we achieved a comprehensive understanding of each model's stability and consistent performance across different data segments.

Generally the following table summarize the performance of machine learning algorithms by different performance measure metrics such as Accuracy, Precision, recall and F-1 score.

Table 5.1 Performance Measures of Machine Learning Algorithms

Algorithm	Performance measure metrics							
	10-fold cross-validation				Train test split data (80 training, 20 test)			
	accuracy	precision	recall	F1 score	accuracy	precision	recall	F1 score
LR	96.59%	97.56%	95.58%	96.56%	96.67%	97.66%	95.61%	96.62%
SVM	92.61%	91.88%	93.46%	92.66%	92.12%	91.15%	93.25%	92.19%
DT	97.33%	97.30%	97.36%	97.33%	97.39%	97.48%	97.28%	97.38%

From the above result detailed in Table 5.1, Decision Tree (DT) model emerged superior with an impressive accuracy rate of 97.33%. It stood as a testament to both the algorithm's robustness and the quality of the dataset. Following closely was the Logistic Regression (LR) with an accuracy of 96.59%. Its performance was a testament to its continued relevance and efficiency in binary classification tasks. The Support Vector Machine (SVM), while potent and often favored for high-dimensional datasets, yielded an accuracy of 92.61% in this study.

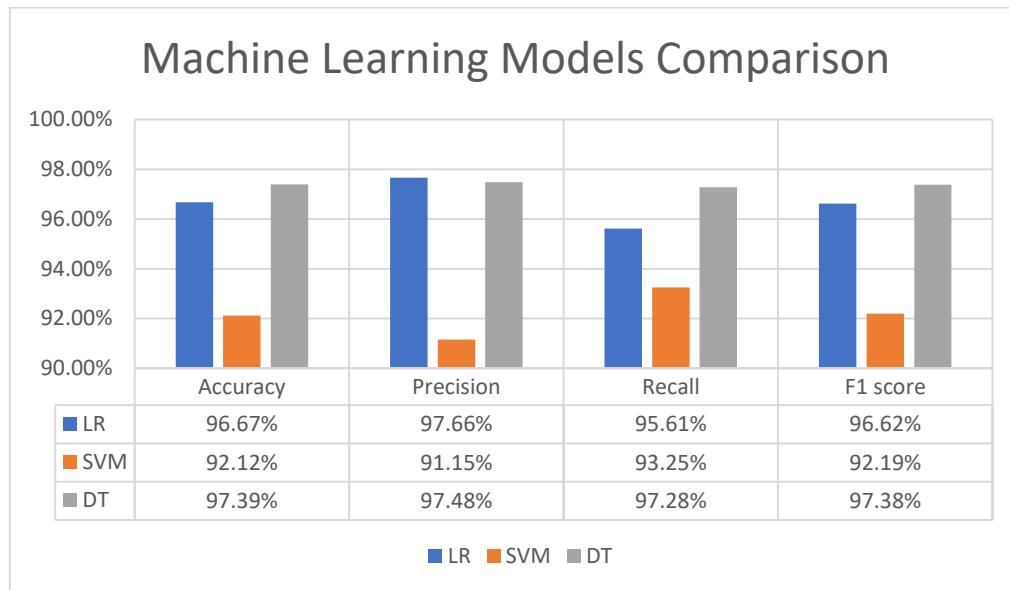
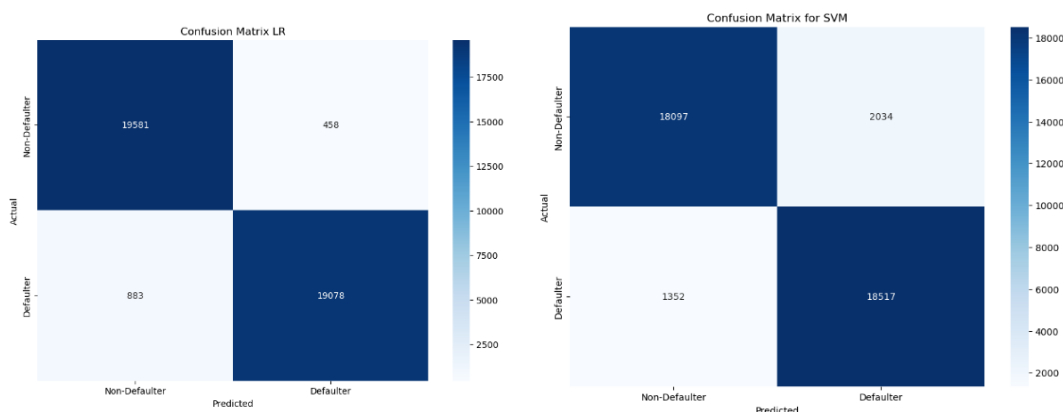


Figure 5.1 Comparison of Machine Learning Algorithms using 10-fold cross validation

The following figure shows the confusion metrics of machine learning models.



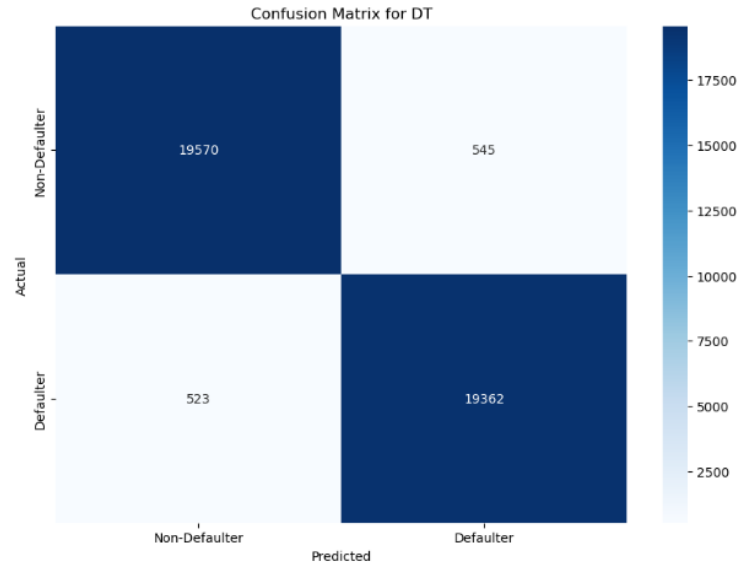


Figure 5.2 Confusion Matrix for Machine Learning algorithms: LR (top left), SVM (top-right) and DT bottom

As we pivot towards the exploration of deep learning algorithms, these preliminary models and their benchmarks will serve as a cornerstone. The outstanding performance of the Decision Tree, especially, offers a glimpse into the potential successes we can achieve with deeper and more complex models. The journey ahead will seek to ascertain whether deep learning can indeed surpass these initial benchmarks and set new standards in airtime and credit risk prediction.

5.2 Result of Deep Learning Algorithms

In our quest to optimize the prediction of airtime and package selection, we ventured into the domain of deep learning, harnessing the power of its advanced computational capabilities. We selected four cutting-edge deep learning algorithms for our study: the Multi-layer Perceptron (MLP), TabNet, Wide and Deep network, and the Convolutional Neural Network (CNN).

Each of these algorithms brings its unique strengths to the table. The Multi-layer Perceptron (MLP), with its interconnected layers and nodes, is a classic yet potent deep learning tool. The TabNet, on the other hand, is a newer and more sophisticated algorithm, capable of capturing intricate patterns with its decision-based approach. The Wide and Deep network is renowned for its dual architecture, effectively balancing memorization and generalization. Meanwhile, the CNN, typically lauded for image classification tasks, has shown promise in various other domains due to its ability to recognize spatial hierarchies in data.

Our comparative analysis yielded some intriguing results. The Multi-Layer Perceptron (MLP) classifier emerged as a distinct frontrunner, boasting an impressive accuracy of 99.01%. This performance starkly contrasted with the other models: TabNet registered an accuracy of 97.60%, the Wide and Deep Learning model achieved 98.34%, and CNN model achieved accuracy of 97.39%.

To ensure the robustness of our findings, we also implemented the 10-fold cross-validation technique, a method that further tests the consistency and reliability of each model across varied data segments. The results mirrored our initial findings. The MLP steadfastly retained its apex position, recording an accuracy of 99.00%. TabNet, Wide and Deep Learning, and CNN hovered closely, with scores of 97.54%, 98.22%, and 97.27% respectively.

Generally the following table summarize the performance of algorithms by different performance measure metrics such as Accuracy, Precision, recall and F-1 score.

Table 5.2 Performance Measures of Deep Learning Algorithms

Algorithm	Performance measure metrics							
	10-fold cross-validation				Train test split data (80 training, 20 test)			
	accuracy	precision	recall	F1 score	accuracy	precision	recall	F1 score
MLP	99.00%	98.77%	98.61%	98.54%	99.01%	98.87%	99.14%	99.00%
TabNet	97.54%	97.33%	97.78%	97.53%	97.60%	97.36%	97.84%	97.60%
Wide and Deep Learning	98.22%	98.40%	98.00%	98.0%	98.34%	98.82 %	97.82%	98.32%
CNN	97.27%	97.54%	96.98%	97.26%	97.39%	97.77%	96.96%	97.36%

From the result detailed in Table 5.2, the Multilayer Perceptron (MLP) model clearly stands out as the top performer across all evaluation criteria and for both validation techniques. In the realm of 10-fold cross-validation, the MLP shines with an unmatched accuracy of 99%. Its nearest competitors, TabNet and Wide and Deep Learning, exhibit close performance at 97.54% and 98.22% respectively. The CNN model also perform well with an accuracy of 97.27%.

In the train-test split evaluation (comprising 80% training and 20% testing data), the MLP reaffirms its superior performance. It not only showcases an impressive accuracy of 99.01% but also excels in precision (98.87%), recall (99.14%), and F1 score (99.00%). TabNet, while

admirable in its performance, solidifies its second-place position across the board, illustrating its consistency. The Wide and Deep Learning model ranks third and CNN, presenting noteworthy results.

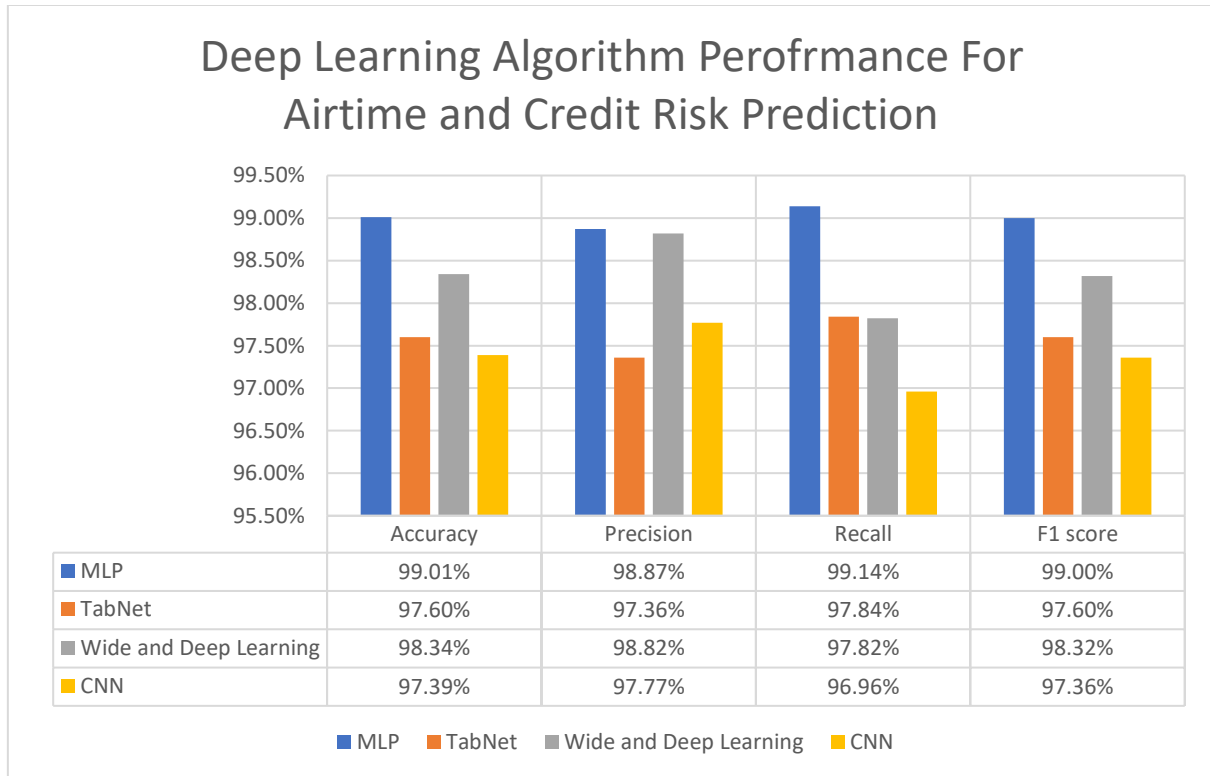


Figure 5.2 Comparison of Deep Learning Algorithms

In addition to the performance measures described above, we used a confusion matrix to evaluate the performance of our classification algorithms. The confusion matrix depicts our model's performance by examining True Positives (TP) are positive instances that are correctly classified as positive, True Negatives (TN) are negative instances that are correctly classified as negative, False Positives (FP) are negative instances that are incorrectly classified as positive, and False Negatives (FN) are positive instances that are incorrectly classified as negative. The following figure shows the confusion metrics of our model.

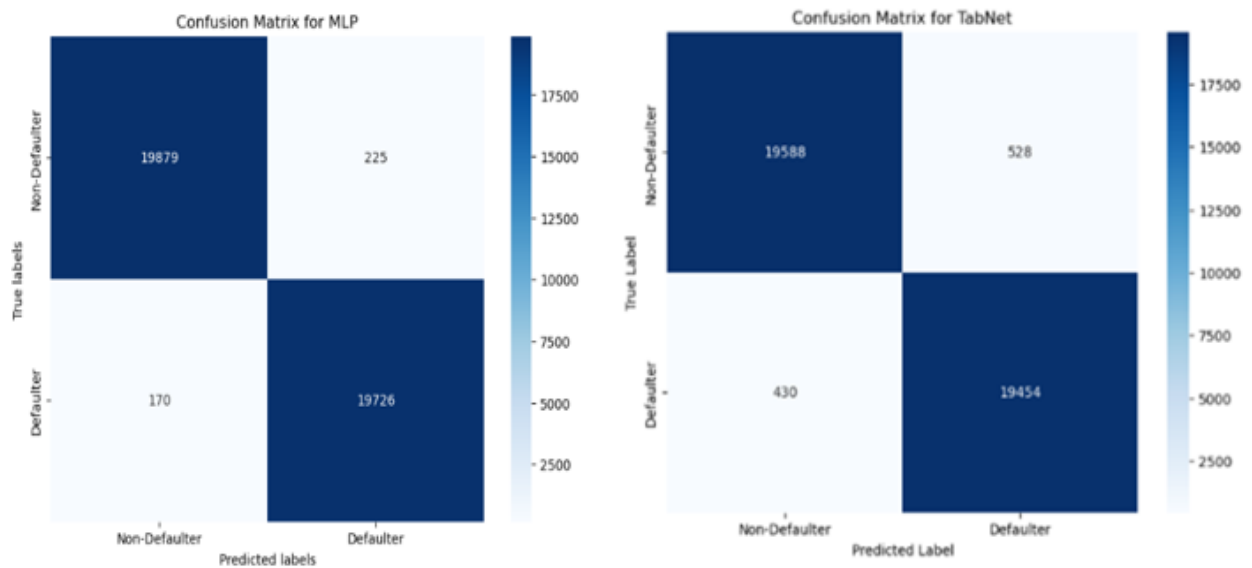


Figure 5.3 Confusion Matrix for MLP (left) and TabNet (right)

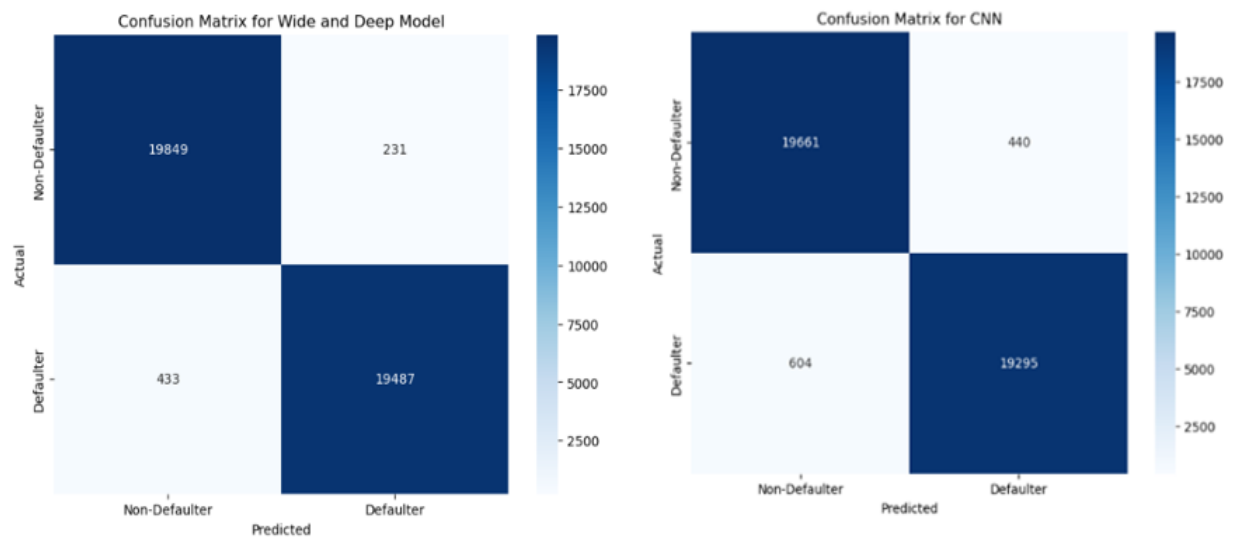


Figure 5.4 Confusion Matrix for wide and deep model (left) and CNN (right)

The ROC curve, or Receiver Operating Characteristic curve, is another essential metric for assessing the performance of classification models. Beyond the traditional measures like accuracy, precision, recall, and the F1 score, the ROC curve offers a visual representation of a model's true positive rate (TPR) against its false positive rate (FPR) at different threshold levels.

Essentially, it provides insights into the trade-offs between correctly classifying positive instances and falsely predicting negative instances as positive. A significant aspect of the ROC curve is the area under the curve (AUC). A model with perfect prediction capability will have an AUC of 1, while a model that predicts randomly will have an AUC of 0.5. Evaluating the ROC curve, especially using techniques like test split validation, allows for a comprehensive understanding of a model's performance across various decision thresholds.

The following figures shows the ROC curve of proposed models:

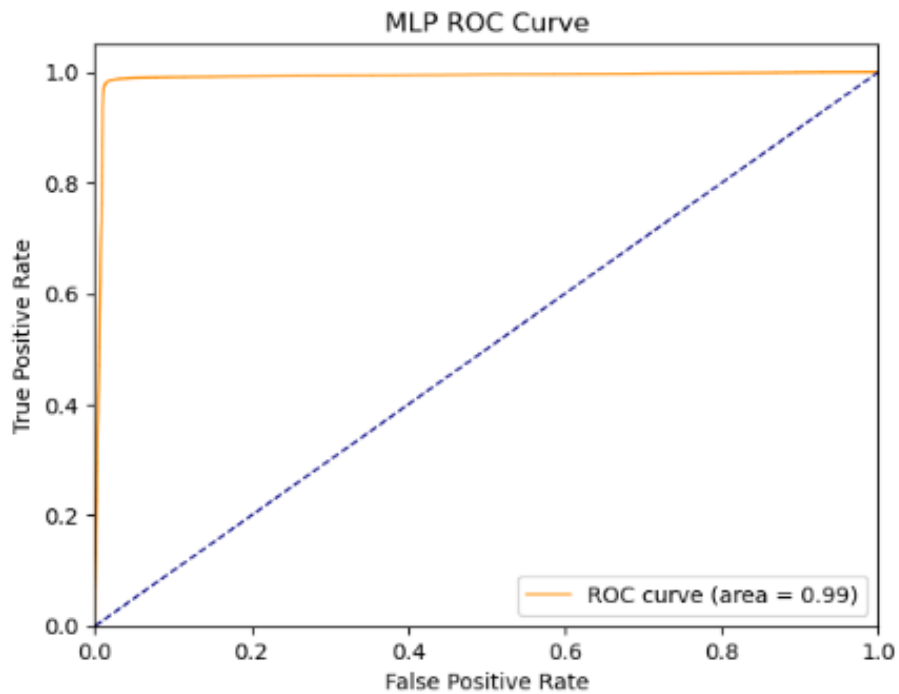


Figure 5.5 ROC curve for MLP

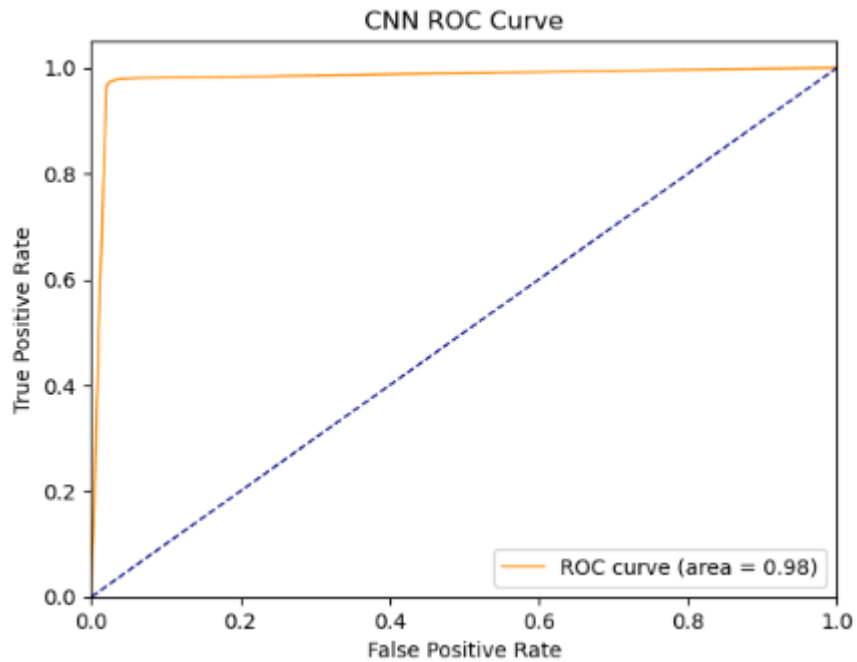


Figure 5.6 ROC curve for CNN model

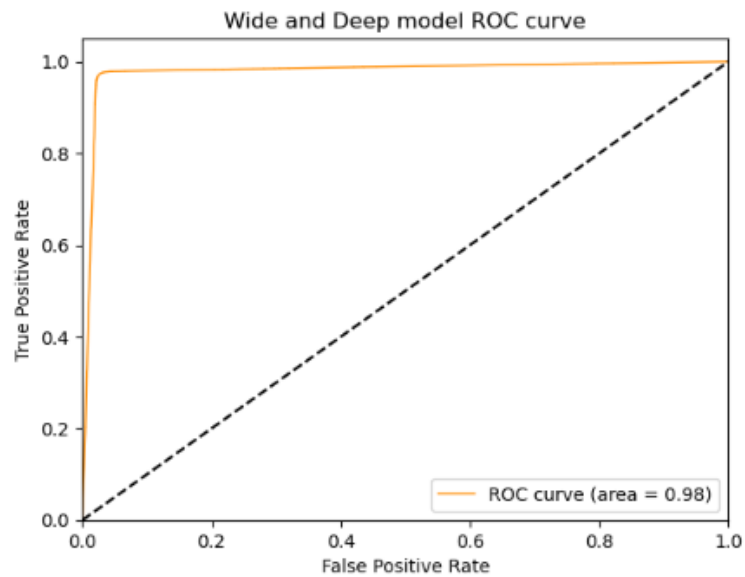


Figure 5.7 ROC curve for Wide and Deep Model

The Receiver Operating Characteristic (ROC) curve offers a compelling representation of the performance of binary classification models. As illustrated in the preceding diagrams, the ROC curve maps the true positive rate (TPR) against the false positive rate (FPR). The horizontal axis, or the FPR, denotes the rate at which a model erroneously categorizes negative instances as positive—a metric we aim to minimize. Conversely, the vertical axis, representing the TPR, portrays the model's proficiency in accurately identifying positive cases, a value we strive to

maximize. A perfect classifier's ROC curve would touch the top-left vertex of the graph, symbolizing a high true positive rate coupled with a negligible false positive rate. The area beneath the ROC curve, commonly referred to as AUC (Area Under the Curve), serves as an aggregate measure of a classifier's efficacy. A larger AUC value signifies superior performance. Observing the ROC curves for the various classification algorithms, we note their proximity to the top-left corner, with the MLP, CNN, and Wide and Deep learning models achieving impressive AUC scores of 99%, 98%, and 98%, respectively. To sum it up, the high AUC-ROC values and the nature of the ROC curves indicate that these models not only exhibit exceptional discriminative power but also proficiently differentiate between defaulters and non-defaulters. This underscores the models' capability to effectively categorize individuals based on their likelihood to default.

In the realm of deep learning, visualizing the training process can offer invaluable insights into the behavior and performance of a model. Two of the most pivotal visual aids in this domain are the accuracy curve and the loss curve. The accuracy curve plots the model's accuracy on the training and validation datasets over successive epochs. For both Multi-Layer Perceptrons (MLP) and Convolutional Neural Networks (CNN), this curve provides a clear picture of how well the model is learning and generalizing to unseen data. A steadily increasing training accuracy indicates effective learning, while the behavior of the validation accuracy can reveal if the model is overfitting (where the training accuracy continues to improve, but the validation accuracy starts to decrease) or underfitting (where both training and validation accuracies are low).

On the other hand, the loss curve illustrates how the loss function's value (or the model's error) evolves during training. For deep learning models like MLP and CNN, the loss function is a mathematical measure of the difference between the predicted output and the actual output. As the model learns, the aim is to minimize this difference. By plotting the loss values for training and validation sets over epochs, we can ascertain the convergence of the model. A divergence between the training and validation loss, much like the accuracy curve, can signal potential overfitting.

The figure below show the model accuracy and loss graph for MLP and CNN:

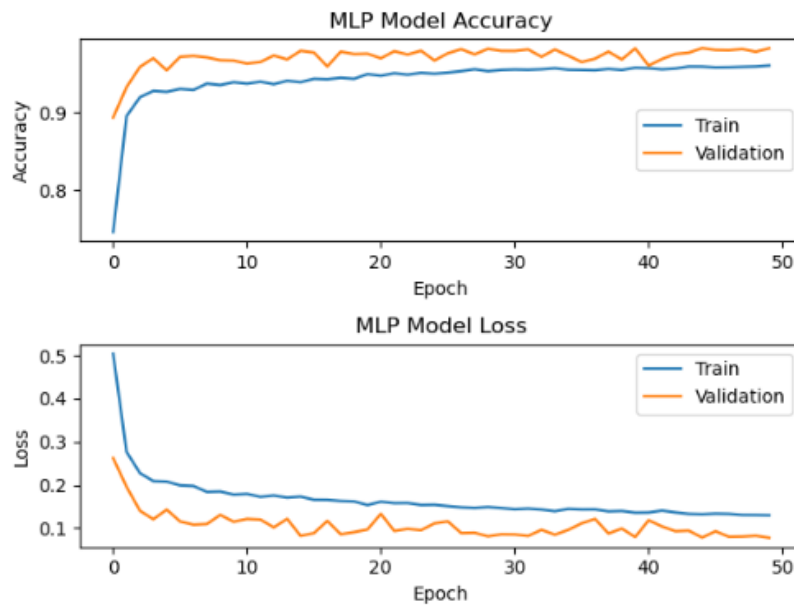


Figure 5.8 MLP model accuracy and model loss

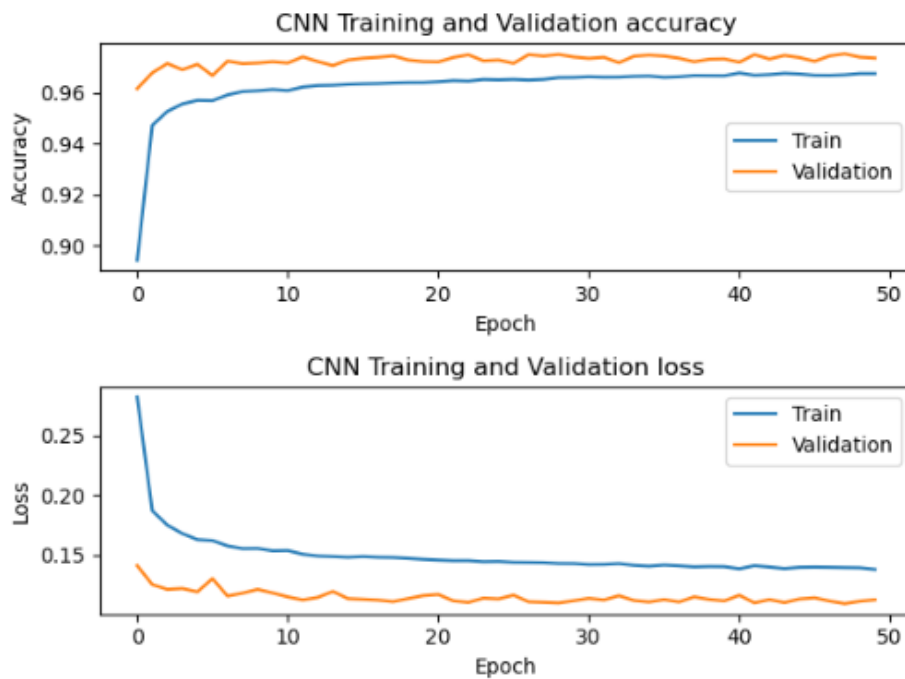


Figure 5.9 CNN accuracy and model loss

5.3 Feature Importance

In the quest to discern the most impactful predictors within our comprehensive dataset, we leveraged the Random Forest classifier, renowned for its capability to compute feature significance (Breiman, 2001). This model was chosen for its interpretability and robust nature in ranking features based on their contribution to predictive accuracy. Our results spotlighted TOTAL_LOAN_AMT as the predominant feature, accounting for approximately 28.4% of the model's predictive power. Following closely, TOTAL_REPAY_AMT and LOAN_FREQ also emerged as significant contributors, with importance scores of 26.3% and 19.6% respectively. Further down the list, features such as TOTAL_ETB, DATA_USAGE_MB, and TOTAL_RECHARGE held their ground with noticeable influence. Conversely, aspects like NETWORK_TYPE and GENDER_NAME manifested lesser importance in the broader scheme. These findings, while resonating with some domain expectations, furnish critical insights that could guide strategic decisions or interventions.

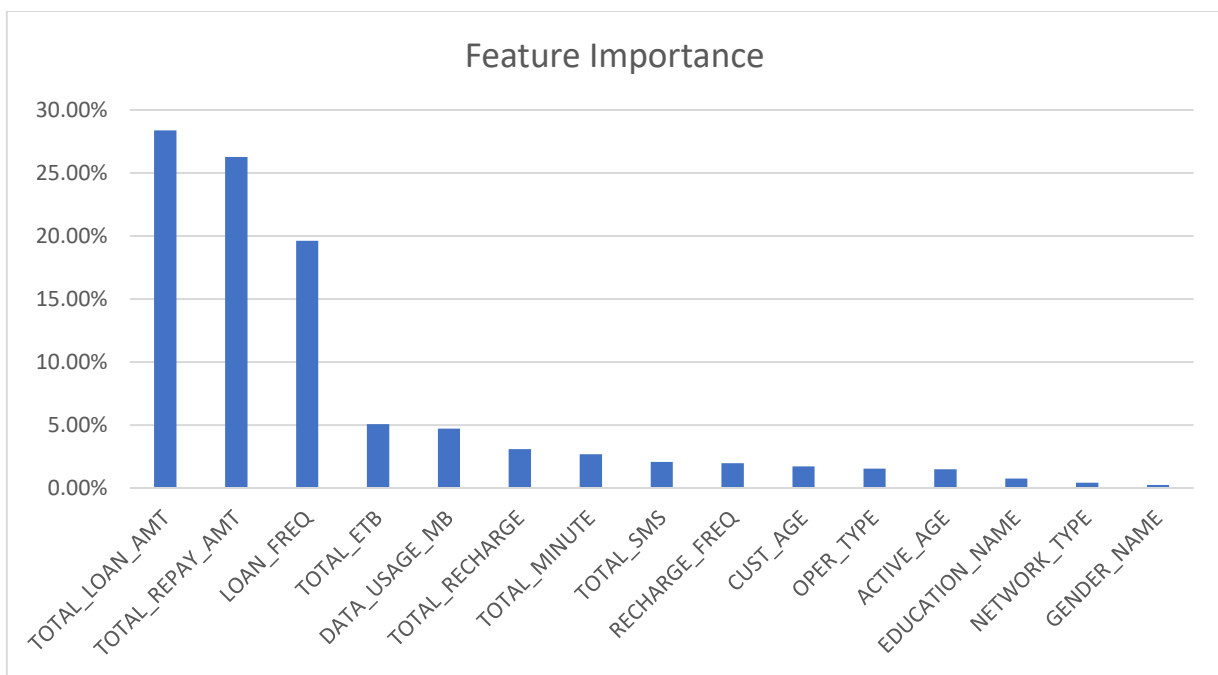


Figure 5.10 Feature importance in airtime and credit risk prediction

5.4 Prototype

In the context of our research, we introduced a novel approach by leveraging the Telegram Bot API for the development of our model prototype. The core of our system is a predictive model

that has been trained and subsequently saved in the .h5 format. This format is particularly chosen because of its compatibility with frameworks like Keras. To facilitate the deployment of this model, we also utilized joblib alongside Keras.

The primary advantage of our design is the integration of the model with a Telegram bot. This bot serves as an intermediary between the user's Telegram application and our saved model. By functioning within the familiar ecosystem of Telegram, our bot offers an intuitive and user-friendly interface. When users wish to assess the creditworthiness or predict the potential of a customer being a defaulter, they simply interact with our Telegram bot. The process is straightforward: the user inputs the service number associated with a particular customer. Our model, working behind the scenes, then processes this information and predicts whether the customer is likely to default on their payments or not. The bot promptly communicates this prediction back to the user.

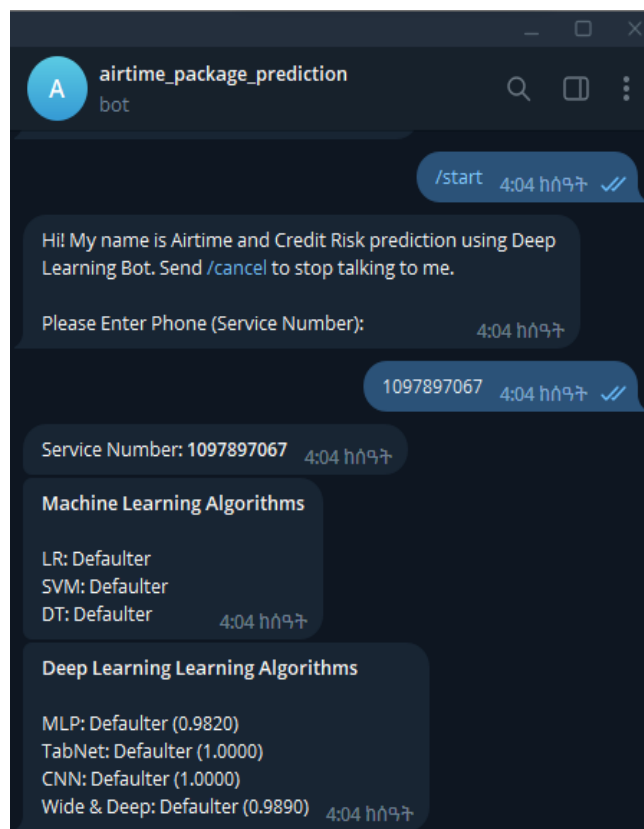


Figure 5.11 Sample prototype using telegram bot

5.5 Research Question Discussion

This study answers to each of the research questions raised in the first chapter. This research provides the following answers to each question:

RQ1: What are the key features required for mobile airtime and credit risk prediction?

Given the multitude of attributes present in our dataset, it was imperative to discern the most influential predictors that maximized the model's forecasting capabilities. To accomplish this objective, valued for its embedded feature importance evaluation. The results exposed TOTAL_LOAN_AMT, TOTAL_REPAY_AMT, LOAN_FREQ, TOTAL_ETB, DATA_USAGE_MB, and TOTAL_RECHARGE as the predominant forces in shaping predictions. Equipped with these insights, we can optimize data collection processes and refine predictive models by emphasizing the high-impact features.

RQ2: Which deep learning algorithm can improve the accuracy of mobile airtime and package credit risk prediction?

We trained and evaluated a repertoire of sophisticated deep learning models, including Multi-Layer Perceptron, TabNet, Wide and Deep Network, and Convolutional Neural Networks on our dataset. Through rigorous comparison using both 10-fold cross-validation and train-test split validation, the Multi-Layer Perceptron emphatically outperformed its peers. With an accuracy approaching 99%, the MLP exhibited unparalleled capability in discriminating between defaulters and non-defaulters. The evidence overwhelmingly supports the Multi-Layer Perceptron as the optimal deep learning architecture for elevating prediction accuracy in this domain.

CHAPTER SIX

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

In today's rapidly progressing technological age, convenience has become a prime concern for consumers, especially in the telecom sector. Mobile credit services have become indispensable by offering a solution for prepaid mobile subscribers to continue using telecom services when they run out of balance. This not only enables consistent communication for the subscribers but also ensures that telecom operators, such as Ethio telecom, maintain a steady flow of revenue even during subscriber 'downtimes'. The introduction of such services solidifies the fact that Ethio telecom seeks to ensure that their customers have uninterrupted access to their services, regardless of their financial or geographical circumstances.

However, the provision of these services is not without its set of challenges. By venturing into the territory of credit services, Ethio telecom exposes itself to potential risks. A significant proportion of subscribers might delay or, worse, avoid repayment entirely. Such defaults pose a real threat to the company's revenue stream and can offset the gains realized from this service.

The research undertaken to understand and potentially mitigate these risks was exhaustive. Before any substantive analysis could be undertaken, it was essential to mold the raw data into an analyzable format. Preprocessing, thus, played a pivotal role. The data underwent rigorous cleaning to eliminate inconsistencies, redundancies, and outliers. Feature engineering helped in the identification and creation of variables that could provide richer insights. Integrating data from various sources, ensuring that the final dataset was both comprehensive and cogent, further reinforced the robustness of our methodology.

The crux of the research focused on leveraging the power of deep learning to predict airtime and credit risks. Deep learning, with its layered architecture and ability to draw patterns from vast datasets, was deemed ideal for this task. Four specific algorithms - Multi-Layer Perceptron (MLP), Wide and Deep network, TabNet, and Convolutional Neural Network (CNN) were employed.

To ensure that the findings of the research were both reliable and valid, a rigorous evaluation methodology was employed. The 10-fold cross-validation provided multiple layers of

validation, reducing the chance of overfitting and enhancing the generalizability of the models. Moreover, the 80-20 train-test split reinforced the models' performance, providing a real-world testing scenario.

The results from the deep learning models were revealing. MLP, with its accuracy rate of 99.01%, emerged as the most potent tool for airtime and credit risk prediction. While TabNet and Wide and Deep Learning, exhibit close performance at 97.60% and 98.34% respectively. The CNN model also perform well with an accuracy of 97.39%. The superior performance of MLP highlights its suitability for tasks that involve complex datasets with intricate relationships, as seen in the telecom sector. Ethio telecom, equipped with these insights, can now adopt a more strategic approach towards its mobile credit service. Predictive analytics can be employed to identify potential defaulters in advance, allowing for preventive measures.

6.2 Future Work

In the future, the researcher intends to enhance the models by incorporating financial datasets from services like Mela Microcredit and Enderas microcredit into the existing telecom data, offering a more comprehensive training set for algorithms. Moreover, while models like MLP, TabNet, and Wide and Deep network have been effective, there's potential in exploring other deep learning algorithms like Transformer-based models or RNNs. The goal is also to pivot towards real-time prediction systems that flag potential default indicators instantly. Given the convergence of the telecommunication and financial sectors, there's also an opportunity to collaborate across industries, such as e-commerce or transportation, to unearth new behavioral insights related to credit risk.

*This research project was funded by Adama Science and Technology University under the grant number: **ASTU/SM-R/851/23***

Adama, Ethiopia

References

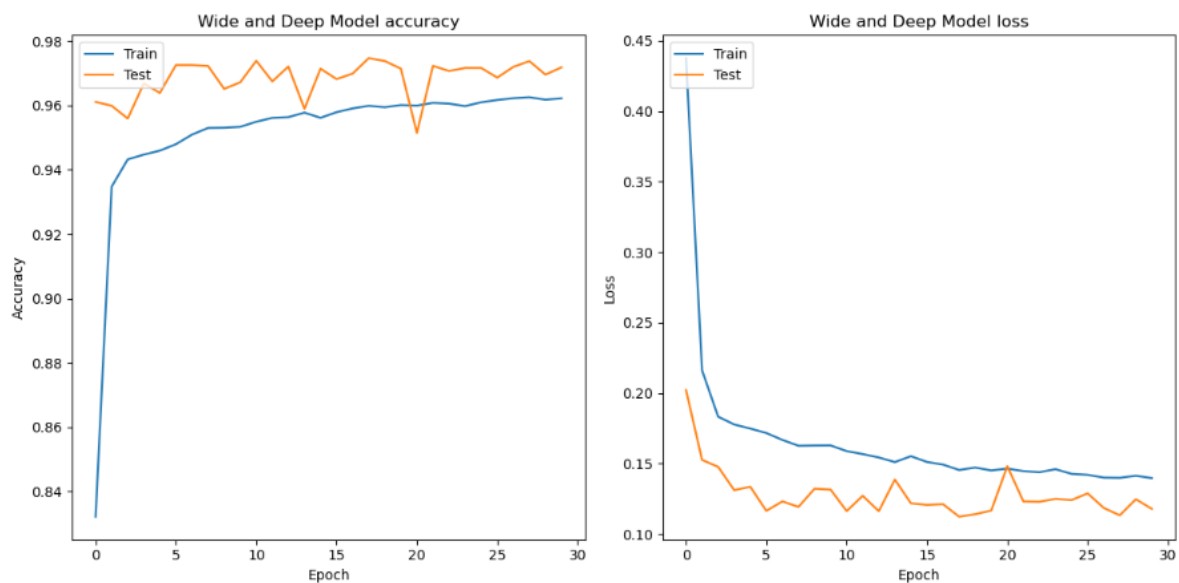
- Abdallah, Z. S., Du, L., & Webb, G. I. (2017). Data Preparation. In *Encyclopedia of Machine Learning and Data Mining* (pp. 318–327). Springer US. https://doi.org/10.1007/978-1-4899-7687-1_62
- Alzubi, J., Nayyar, A., & Kumar, A. (2018). Machine Learning from Theory to Algorithms: An Overview. *Journal of Physics: Conference Series*, 1142(1). <https://doi.org/10.1088/1742-6596/1142/1/012012>
- Amitha Mathew, P. Amudha, & S. Sivakumari. (2021). *Deep Learning Techniques An Overview* (pp. 599–609). <http://www.springer.com/series/11156>
- Arik, S. O., & Pfister, T. (2020). *TabNet: Attentive Interpretable Tabular Learning*. <http://arxiv.org/abs/1908.07442>
- Bastani, K., Asgari, E., & Namavari, H. (2019). Wide and deep learning for peer-to-peer lending. *Expert Systems with Applications*, 134, 209–224. <https://doi.org/10.1016/j.eswa.2019.05.042>
- Bhatore, S., Mohan, L., & Reddy, Y. R. (2020). Machine learning techniques for credit risk evaluation: a systematic literature review. *Journal of Banking and Financial Technology*, 4(1), 111–138. <https://doi.org/10.1007/s42786-020-00020-3>
- Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., & Kasneci, G. (2021). *Deep Neural Networks and Tabular Data: A Survey*. <http://arxiv.org/abs/2110.01889>
- Breiman, L. (2001). *Random Forests* (Vol. 45).
- Brhane, S., Advisor, B., & Lemma, S. (2021). Machine Learning Based Mobile Airtime Credit Risk Prediction using Customer Profile and Loan Information. *Addis Ababa Institute of Technology School of Electrical and Computer Engineering Telecommunication Engineering*.
- Cheng, H. T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., Anil, R., Haque, Z., Hong, L., Jain, V., Liu, X., & Shah, H. (2016). Wide & deep learning for recommender systems. *ACM International Conference Proceeding Series*, 15-September-2016, 7–10. <https://doi.org/10.1145/2988450.2988454>
- Dargan, S., Kumar, M., Ayyagari, M. R., & Kumar, G. (2020). A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning. *Archives of Computational Methods in Engineering*, 27(4), 1071–1092. <https://doi.org/10.1007/s11831-019-09344-w>
- Dushimimana, B., Wambui, Y., Lubega, T., & McSharry, P. E. (2020). Use of Machine Learning Techniques to Create a Credit Score Model for Airtime Loans. *Journal of Risk and Financial Management*, 13(8), 180. <https://doi.org/10.3390/jrfm13080180>

- Ereiz, Z. (2019, November 1). Predicting Default Loans Using Machine Learning (OptiML). *27th Telecommunications Forum, TELFOR 2019*. <https://doi.org/10.1109/TELFOR48224.2019.8971110>
- Garba Muhammad, K., Sa, S., & Muhammad, K. G. (2019). *Airtime Credit Loan and Service Charge/Fee by Telecommunications Service Providers in Nigeria: Islamic Law Perspective*. 21, 8–14. <https://doi.org/10.9790/487X-2108040814>
- Heaton, J. (2017). *An Empirical Analysis of Feature Engineering for Predictive Modeling*. <https://doi.org/10.1109/SECON.2016.7506650>
- Jordan, M. I. & Mitchell, & T.M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245), 253–255. <https://doi.org/10.1126/science.aac4520>
- Kumar Gupta, D., & Goyal, S. (2018). Credit Risk Prediction Using Artificial Neural Network Algorithm. *International Journal of Modern Education and Computer Science*, 10(5), 9–16. <https://doi.org/10.5815/ijmecs.2018.05.02>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. In *Nature* (Vol. 521, Issue 7553, pp. 436–444). Nature Publishing Group. <https://doi.org/10.1038/nature14539>
- Liaw, A., & Wiener, M. (2002). *Classification and Regression by randomForest* (Vol. 2, Issue 3). <http://www.stat.berkeley.edu/>
- Liu, T., Fang, S., Zhao, Y., Wang, P., & Zhang, J. (2015). *Implementation of Training Convolutional Neural Networks*.
- Li, Z., Liu, F., Y., W., P. S., & Zhou, J. (2021). *Survey of Convolutional Neural Networks Analysis, Applications, and Prospects*.
- Mahesh, B. (2018). Machine Learning Algorithms-A Review Machine Learning Algorithms-A Review View project Self Flowing Generator View project Batta Mahesh Independent Researcher Machine Learning Algorithms-A Review. *International Journal of Science and Research*. <https://doi.org/10.21275/ART20203995>
- McDonnell, K., Murphy, F., Sheehan, B., Masello, L., & Castignani, G. (2023). Deep learning in insurance: Accuracy and model interpretability using TabNet. *Expert Systems with Applications*, 217. <https://doi.org/10.1016/j.eswa.2023.119543>
- M, H., & M.N, S. (2015). A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2), 01–11. <https://doi.org/10.5121/ijdkp.2015.5201>
- Nasteski, V. (2017). An overview of the supervised machine learning methods. *HORIZONS.B*, 4, 51–62. <https://doi.org/10.20544/horizons.b.04.1.17.p05>

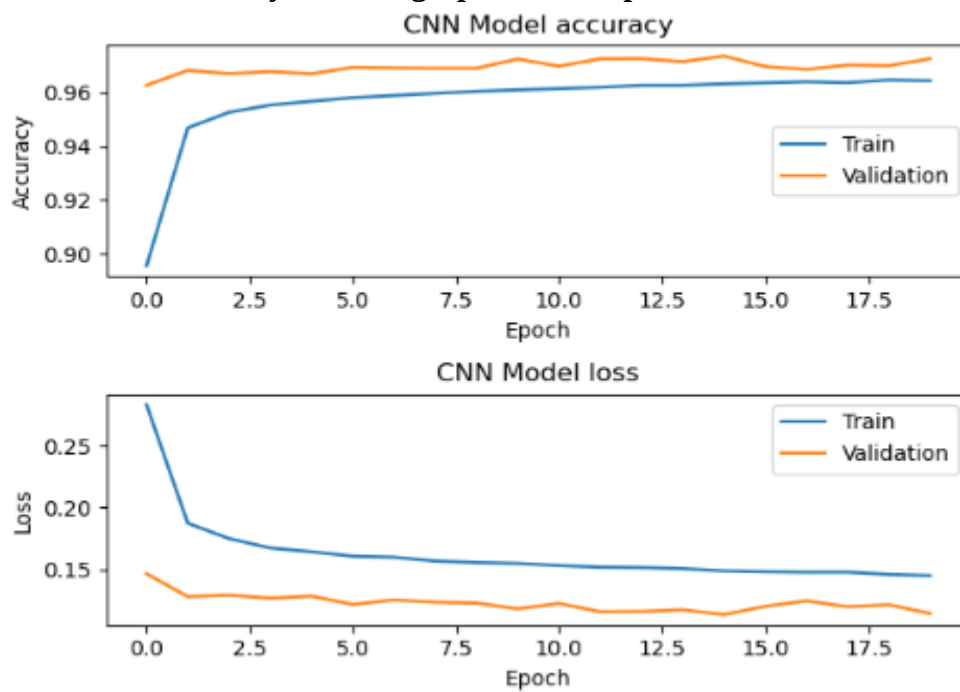
- Priyam, A., Gupta, R., Rathee, A., & Srivastava, S. (2013). *Comparative Analysis of Decision Tree Classification Algorithms* (Vol. 3, Issue 2). <http://inpressco.com/category/ijcet>
- Ramchoun, H., Amine, M., Idrissi, J., Ghanou, Y., & Ettaouil, M. (2016). Multilayer Perceptron: Architecture Optimization and Training. *International Journal of Interactive Multimedia and Artificial Intelligence*, 4(1), 26. <https://doi.org/10.9781/ijimai.2016.415>
- Sah, S. (2020). *Machine Learning: A Review of Learning Types*. <https://doi.org/10.20944/preprints202007.0230.v1>
- S. B. Kotsiantis, D. Kanellopoulos, & P. E. Pintelas. (2006). Data Preprocessing for Supervised Learning. *INTERNATIONAL JOURNAL OF COMPUTER SCIENCE VOLUME 1 NUMBER 1 2006 ISSN 1306-4428*.
- Sharma, S., Sharma, S., & Athaiya, A. (2020). Activation Functions In Neural Networks. In *International Journal of Engineering Applied Sciences and Technology* (Vol. 4). <http://www.ijeast.com>
- Shema, A. (2019, January 4). Effective credit scoring using limited mobile phone data. *ACM International Conference Proceeding Series*. <https://doi.org/10.1145/3287098.3287116>
- Shrestha, A., & Mahmood, A. (2019). Review of deep learning algorithms and architectures. In *IEEE Access* (Vol. 7, pp. 53040–53065). Institute of Electrical and Electronics Engineers Inc. <https://doi.org/10.1109/ACCESS.2019.2912200>
- Susmita Ray. (2019). A Quick Review of Machine Learning Algorithms. *International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (Com-IT-Con), India*.
- Tarekegn, O., & Ababa, A. (2019). *Application of Data Mining Technique for Predicting Airtime Credit Risk: The Case of Ethio Telecom*.
- Xie, L., & Yuille, A. (n.d.). *Genetic CNN*.

Appendix A - Additional Results and Illustrations

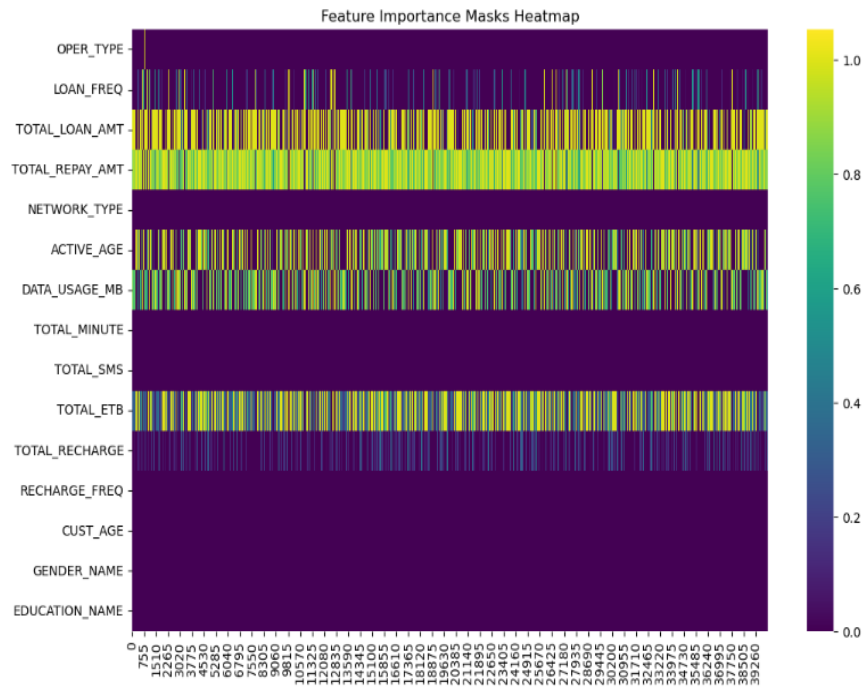
a) Wide and Deep Learning



b) CNN model accuracy and loss graph with 20 epochs



c) TabNet Masks Heat map



d) Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train a random forest classifier
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

# Get feature importances from the trained random forest model
rf_feature_importances = rf.feature_importances_

# Create a dictionary of feature names and their importance scores
feature_importance_dict = dict(zip(X.columns, rf_feature_importances))

# Sort the dictionary by importance score
sorted_feature_importance = sorted(feature_importance_dict.items(), key=lambda x: x[1], reverse=True)

sorted_feature_importance

[('TOTAL_LOAN_AMT', 0.28392615239238544),
 ('TOTAL_REPAY_AMT', 0.2627905098577073),
 ('LOAN_FREQ', 0.19623670296590068),
 ('TOTAL_ETB', 0.0506704522552659),
 ('DATA_USAGE_MB', 0.04699147575490752),
 ('TOTAL_RECHARGE', 0.03085867768363023),
 ('TOTAL_MINUTE', 0.026751857860053442),
 ('TOTAL_SMS', 0.020602321737614716),
 ('RECHARGE_FREQ', 0.01964357738632111),
 ('CUST_AGE', 0.017199957195831162),
 ('OPER_TYPE', 0.015271880508339558),
 ('ACTIVE_AGE', 0.014761727862908379),
 ('EDUCATION_NAME', 0.007538842168829137),
 ('NETWORK_TYPE', 0.004258510289603969),
 ('GENDER_NAME', 0.0024973540807015607)]
```

Appendix B – Telegram bot

For demonstration purpose we build the telegram bot. Here the sample code for telegram bot

```
import logging
from telegram import Update
from telegram.ext import (
    Application,
    CommandHandler,
    ContextTypes,
    ConversationHandler,
    MessageHandler,
    filters,
)
from joblib import load
from pytorch_tabnet.tab_model import TabNetClassifier
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np
import pandas as pd

# Enable logging
logging.basicConfig(
    format='%(asctime)s - %(name)s - %(levelname)s - %(message)s",
    level=logging.INFO
)
logger = logging.getLogger(__name__)

# Define conversation state constant
SERV_NO = 1

features = [ 'OPER_TYPE', 'LOAN_FREQ', 'TOTAL_LOAN_AMT', 'TOTAL_REPAY_AMT',
             'NETWORK_TYPE', 'ACTIVE_AGE', 'DATA_USAGE_MB', 'TOTAL_MINUTE',
             'TOTAL_SMS', 'TOTAL_ETB', 'TOTAL_RECHARGE', 'RECHARGE_FREQ',
             'CUST_AGE', 'GENDER_NAME', 'EDUCATION_NAME'
           ]

df = pd.read_csv('validation.csv')
lr_model = load('models/logistic_regression_model.joblib')

dt_model = load('models/decision_tree_model.joblib')
dt_features = features

svm_model = load('models/svm_model.joblib')
svm_features = features

mlp_model = load_model("models/mlp.keras")
tabnet_model = TabNetClassifier()

tabnet_model.load_model('models/tabnet-model.zip')

cnn_model = load_model("models/cnn.h5")
wide_deep_model = load_model("models/wide-and-deep")

def find_serv_no(col):
    a = df.loc[df['SERV_NO'] == serv_no, col]

    # Check if 'a' has at least one row
    if len(a) > 0:
        return a # Return the first row as a Series
    else:
        raise ValueError(f"serv_no {serv_no} not found in DataFrame.")

def wide_deep_model_prediction(serv_no):
    categorical_cols = ['OPER_TYPE', 'NETWORK_TYPE', 'GENDER_NAME', 'EDUCATION_NAME']
    numerical_cols = ['LOAN_FREQ', 'TOTAL_LOAN_AMT', 'TOTAL_REPAY_AMT', 'ACTIVE_AGE',
                     'DATA_USAGE_MB', 'TOTAL_MINUTE', 'TOTAL_SMS', 'TOTAL_ETB',
                     'TOTAL_RECHARGE', 'RECHARGE_FREQ', 'CUST_AGE']

    encoders = {}
    for col in categorical_cols:
        le = LabelEncoder()
        df[col] = le.fit_transform(df[col])
        encoders[col] = le
    new_df = find_serv_no, features
    # Apply the transformations to new_df
    for col, encoder in encoders.items():
        new_df.loc[:, col] = encoder.transform(new_df[col].astype(str))

    # Create Wide and Deep inputs
    wide_input_new = new_df[categorical_cols].values
    deep_input_new = new_df[numerical_cols].values

    # Use the loaded model to make predictions
    predictions = wide_deep_model.predict([wide_input_new, deep_input_new])

    # Convert predictions to class labels if you are working on a classification problem
    predicted_labels = (predictions > 0.5).astype(int)
    threshold = 0.5
    if predicted_labels[0] >= threshold:
        predicted_class = "Defaulter"
```

```

else:
    predicted_class = "Non-defaulter"
    return f"{predicted_class} ({predictions[0][0]:.4f})"

def cnn_model_prediction(serv_no):
    row = find_(serv_no, cnn_features)

    reshaped = np.reshape(row.values, (row.shape[0], 4, 4, 1))

    cnn_predictions = cnn_model.predict(reshaped)
    cnn_prediction_rounded = np.round(cnn_predictions)
    threshold = 0.5
    if cnn_prediction_rounded[0][0] >= threshold:
        predicted_class = "Defaulter"
    else:
        predicted_class = "Non-defaulter"

    return f"{predicted_class} ({cnn_prediction_rounded[0][0]:.4f})"
    # return cnn_predictions

def tabnet_model_prediction(serv_no):
    tabnet_ = find_(serv_no, dt_features).iloc[0].values.reshape(1, -1)
    tabnet_prediction = tabnet_model.predict(tabnet_)[0]
    threshold = 0.5
    if tabnet_prediction >= threshold:
        predicted_class = "Defaulter"
    else:
        predicted_class = "Non-defaulter"

    return f"{predicted_class} ({tabnet_prediction:.4f})"
    # return tabnet_prediction

def mlp_model_prediction(serv_no):
    mlp = find_(serv_no, mlp_features)
    mlp_prediction = mlp_model.predict(mlp)
    threshold = 0.5
    if mlp_prediction[0][0] >= threshold:
        predicted_class = "Defaulter"
    else:
        predicted_class = "Non-defaulter"

    return f"{predicted_class} ({mlp_prediction[0][0]:.4f})"
    # return mlp_prediction

# Start handler
async def start(update: Update, context: ContextTypes.DEFAULT_TYPE) -> int:
    await update.message.reply_text(
        "Hi! My name is Airtime and Credit Risk prediction using Deep Learning Bot. "
        "Send /cancel to stop talking to me. \n\n"
        "Please Enter Phone (Service Number):"
    )
    return SERV_NO

# Service number handler
async def service_no(update: Update, context: ContextTypes.DEFAULT_TYPE) -> int:
    serv_no = update.message.text
    try:
        serv_no = int(serv_no)

        # ML model:
        lr = lr_prediction(serv_no)
        svm = svm_prediction(serv_no)
        dt = dt_prediction(serv_no)

        # Deep Learning
        mlp = mlp_model_prediction(serv_no)
        tabnet = tabnet_model_prediction(serv_no)
        cnn = cnn_model_prediction(serv_no)
        wide_deep = wide_deep_model_prediction(serv_no)

        await update.message.reply_html(
            f"""
            Service Number: <b>{serv_no}</b>
            """
        )
        await update.message.reply_html(
            f"""
            <b>Machine Learning Algorithms</b>
            """
        )
    except ValueError:
        await update.message.reply_text(
            "That doesn't look like a valid service number or not found in the data set. Please enter a number: ""
        )
    return SERV_NO

# Cancel handler
async def cancel(update: Update, context: ContextTypes.DEFAULT_TYPE) -> int:
    user = update.message.from_user
    logger.info("User %s canceled the conversation.", user.first_name)
    await update.message.reply_text("Bye!")
    return ConversationHandler.END

# Main function
def main() -> None:
    TOKEN = "6477580901:AAH762rAe7p8WpgrJFYGGGQbn8I8-0y0Mc"
    application = Application.builder().token(TOKEN).build()

    conv_handler = ConversationHandler(
        entry_points=[CommandHandler("start", start)],
        states={
            SERV_NO: [MessageHandler(filters.TEXT, service_no)],
        },
        fallbacks=[CommandHandler("cancel", cancel)],
    )

    application.add_handler(conv_handler)
    application.run_polling()

if __name__ == "__main__":
    main()

```

