**PGPDSE FT Capstone Project**

**Oct 2023 - 2024**

**A Project Report on**

**“Loan Default Prediction”**

**Submitted in partial fulfilment for the award of the degree of**

**POST GRADUATE PROGRAM**

**IN**

## DATA SCIENCE AND ENGINEERING

### Submitted by

**3NJZ6RGL7W ANANNYA SINGH**

**VT8ZPRQXLM PRIYANSHU SHARMA**

**WX2EE50531 RAYA HARISH**

**FQXYZSGA75 SANNIDHI HARSHA VEENA**

**PVLWKE0CT4 VAISAKHI RAMESH**

**Under the guidance of**

**Mr. Animesh Tiwari**

**ABSTRACT**

Due to the ever-changing nature of the banking industry and its growing complexity, risk assessment has gained significant importance, especially in the financial sector. Consequently, financial institutions face intense rivalry, which leads to the loss of most loans. Researchers and banks have created credit scoring models to enhance the credit evaluation process with the goal of lowering credit risk and increasing credit quality. Because of the intricacy of the database, it is quite challenging for anyone to evaluate a customer's reliability. To address these problems, a framework that combines several features to determine risk assessment is required.

This project provides a succinct investigation of machine learning models for risk assessment strategy is examined, and suggested architecture is created with the intention of drastically lowering the intricacy of the information as well as to raise the classifications' accuracy in comparison to other current techniques.

## TABLE OF CONTENTS

[List of Figures IV & V](#_Toc46681)

List of Tables v

**Chapter 1** **01**

* 1. INTRODUCTION  **01**
  2. Problem Statement **01**
  3. Objectives **01**
  4. Methodology **02**

1.4.1 Business Understanding **02**

1.4.2 Data Understanding **03**

1.4.3 Data Preparation **03**

* 1. Software tools used **03**

**Chapter 2** **04**

1. **TOOL DESCRIPTION 04**

**Chapter 3** **07**

1. **DETAILED PROCEDURES 07**

**Chapter 4 17**

1. **VISUALIZATION 17**

**Chapter 5 22**

1. **FEATURE ENGINEERING 22**

**Chapter 6 23**

1. **FEATURE SELECTION 23**

**Chapter 7 25**

1. **MODEL BUILDING 25**

**Chapter 8 29**

1. **CHALLENGES FACED DURING MODEL BUILDING 29**

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| Fig. No. | Description of the figure | Page No. |
| 1.4 | CRISP-DM Methodology in a Nutshell | 2 |
| 4.1 | Employment type having the highest percentage of defaulters | 17 |
| 4.2 | Distribution of employment types among all customers | 18 |
| 4.3 | Age Group Showing the Highest Rate of Defaults | 19 |
| 4.4 | First Emi Bounce Correlates with Loan Default Likelihood | 19 |
| 4.5 | First Emi Bounces among Defaulters and Non-Defaulters | 20 |
| 4.6 | Employment Type vs. Age at Loan Origination by Default Status | 21 |
| 4.7 | Impact of Length of Repayment Period on the Likelihood of the Default | 21 |
| 7.1 | Base Model Evaluation | 25 |
| 7.1.1 | ROC-AUC Curve for Train and Test Data | 25 |
| 7.2 | Results after using SMOTE techniques | 26 |
| 7.3.1 | Results after using SMOTE-TOMEK techniques for Train Data | 27 |
| 7.3.1.1 | ROC-AUC Curve for Train Data | 27 |
| 7.3.2 | Results after using SMOTE-TOMEK techniques for Test Data | 28 |
| 7.3.2.1 | ROC-AUC Curve for Test Data | 28 |
| 7.4 | Results after using SMOTENN techniques for Train and Test Data | 29 |

# List of Tables

|  |  |  |
| --- | --- | --- |
| Table no. | Description of table | Page no. |
| 3.1.3 | Data Dictionary | 8 |
| 3.5.1 | Test Statistics and Decisions | 16 |
| 3.5.2 | Test Statistics and Decisions: | 16 |

**Chapter 1**

## INTRODUCTION

This project was developed to gain experience with machine learning using real-world data, and serves as a feasibility analysis of using ML with vehicle loan and customer information from TVS - a large Indian motorcycle company. The raw data is available here and is sourced from Kaggle.com. Our project addresses the concern of cross-selling personal loans to existing auto loan customers who are possible default risks, because personal loans are unsecured and therefore carry a relatively high cost for the provider if a default occurs. Using ML, a subset of the current vehicle loan customer base could be selected that only includes customers unlikely to default on a personal loan, thus reducing the risk to the personal loan provider (in this particular case, TVS Credit).

The application itself is hosted here on Heroku and would be used internally by loan officers at a company like TVS Credit to disqualify or at least re-assess loan terms for personal loans to customers flagged by this form as potential default risks. Any input field that corresponds to money is in Indian rupees, since the data was all in rupees from TVS.

### 1.2. Problem Statement

TVS Credit faces rising loan defaults, with an overall default rate increasing from 2.2% in FY2022 to 5.7% for sub-investment grade loans in FY2024. This trend results in financial losses for lenders, restricted credit access for future borrowers, and instability in the financial system. We developed a predictive model to identify loan defaulters and key risk factors, aiding financial institutions in making informed lending decisions and mitigating default risks.

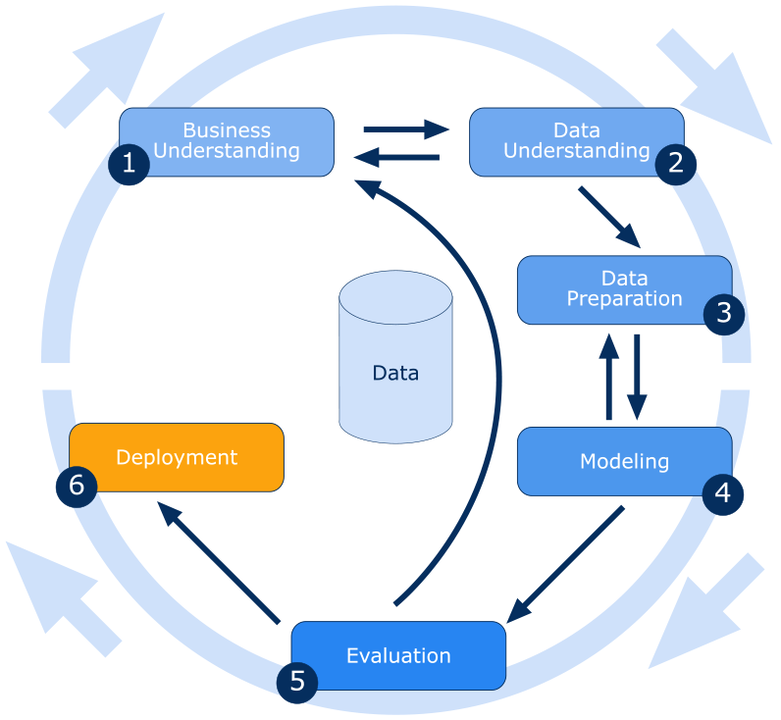
### 1.3. Objectives

The main objectives of this project are:

* Develop a robust risk assessment model to evaluate customer creditworthiness and predict loan default probability.
* Gain insight into the importance of loan default prediction in financial risk assessment and decision-making.
* Understand the application of machine learning algorithms for loan default prediction in Python.

### 1.4. Methodology

The Analysis of Different Classification Algorithms for Loan Default Prediction is illustrated in the flow chart as shown in the Figure.1**.4.**



**Figure 1.4: CRISP-DM Methodology in a Nutshell**

**1.4.1 Business Understanding**

Being a secured loan or an unsecured loan, the Personal Loan product necessitates evaluating the risk of each consumer by verifying their creditworthiness. To avoid loan defaults, this needs to be done. The aim is to utilize the dataset to construct a risk model that evaluates the likelihood of a client defaulting following a cross-sell of the Personal Loan. To determine the creditworthiness of customers and forecast the likelihood of loan default, create a reliable risk assessment model. Learn how crucial it is to predict loan default to assess and make decisions about financial risk. Recognize how to use machine learning techniques in Python to anticipate loan default.

**1.4.2 Data Understanding:**

In this project, we will use the lending club loan dataset that can be downloaded from ”[TVS\_Loan\_Default” (kaggle.com)](https://www.kaggle.com/datasets/sjleshrac/tvs-loan-default/code)”. It consists of 119528 records and 30 columns. The feature V32: Target variable (1: Defaulters / 0: Non-Defaulters) represents whether the customer has defaulted on their loan or not. Our objective is to build a machine learning-based solution to predict the loan default of customers in advance based on features such as V14: Employment type, V18: Number of loans, V5: Number of times bounced while repaying the loan, V25: Maximum amount sanctioned for any Two-wheeler loan and V21: Maximum amount sanctioned in the Live loan etc.

**1.4.3 Data Preparation:**

Our approach starts with a comprehensive data exploration phase where we dive deep into unique values, duplicates, and data types, meticulously separating categorical and numerical columns for clarity. We tackle missing values and outliers systematically, ensuring our data is clean and ready for analysis. Visualizations play a crucial role in understanding data distributions and relationships, guiding our decisions on feature engineering techniques like categorical encoding and scaling. We employ a diverse set of algorithms—Logistic Regression, Naive Bayes, KNN, Decision Trees, Random Forest, AdaBoost, XGBoost, and Gradient Boosting—allowing us to harness the power of ensemble methods for superior predictive accuracy. Throughout our process, we integrate insights from exploratory analysis to continually refine our pre-processing strategies and optimize our model selection, ensuring robust and reliable results for our predictive tasks.

**1.5 Software tools used**:

* Jupyter notebook
* Python

**Chapter 2**

**2. Tool Description**

* 1. **Jupyter Notebook:**

It is an open-source application that allows writing code, equations, visualizations and text (in markdown format) in one place, making the readability, documentation and execution of code very simple and easy. It is one of the best applications for data scientists. The standard Jupyter notebook document extension is. ipynb. It is a book record that is put away in the JSON design that contains the substance of the notebook. There might be numerous cells in a notebook and the substance of each can be python code, text or a video connection that has been changed over into strings of text and is accessible alongside the metadata of the notebook.

The kernel is a software that runs and translates the user’s code. There are exclusive kernels for exclusive languages that Jupyter Notebook makes use of however for Python it extends the I python kernel. A notebook kernel is a “computational engine” that executes the code contained in a Notebook document. The I python kernel, referenced in this guide, executes python code. Kernels for many other languages exist (official kernels).

The kernel executes the code withinside the mobileular and returns the output (if any) to the frontend interface. The state of a kernel relates to the complete record and now no longer simply individual cells. Anything carried out in a single cell could be available to be used within the subsequent cell as well.

There are 4 types of cells in a Jupyter notebook:

1. Code — This is the cell where we write our python code that will be computed by the I python kernel and the output is displayed under the cell.
2. Here is an example of a code cell
3. Markdown — This is where you add the documentation by putting text formatted using Markdown. The output is displayed in place of the cell when it is run.
4. Raw NB Convert — This is another tool to convert your Jupyter notebook into another file format like PDF, HTML, etc.
5. Heading — This is the same as writing a heading (Line starting with #) in Markdown. To add a headline to your notebook you can use this.

The latest product, Jupyter Lab incorporates Jupyter Notebook into an integrated, development type Editor that can run in the browser. It can be thought of Jupyter Lab as an advanced version of Jupyter Notebook. Jupyter Lab allows the user to run terminals, text editors and code consoles in your browser in addition to Notebooks.

**Various functions in the Kernel menu:**

1. Interrupt: Used to stop the execution or running of a particular cell. This command is useful when you have reached a desired result within a specific number of epochs or in case you made an error, and you realize this while running the code.
2. Restart: Useful to restart the Kernel of the Notebook.
3. Restart and Clear Output: Used to restart the Kernel of the Notebook and reset all the cells that were run previously.
4. Restart and run all: Used to restart the Kernel of the Notebook, and reset all the cells that were run previously, and finally re-run through all the cells of the Notebook.
5. Reconnect: Used to reconnect to a dead kernel which might occur at times due to a lack of memory.
6. Shutdown: Used to shut down the current working Kernel of the Notebook.
7. Change kernel: Allows you to switch Kernels.

Programming with python has been given a brand-new picture by these notebooks. Features provided by this notebook can be utilized and data science journey can be more enjoyable with those cells containing the step-by-step evaluation along with the documentation and visible insights.

**2.2 Python:**

The Python programming language is older than many of its popular counterparts including R, Java, and even JavaScript. The concept behind it was first implemented in the 1980s. The then developer, Guido van Rossum came up with Python as a hobby project during Christmas. His reason for creating it was to help him work on Amoeba operating system in handling and user-interfacing. A high-level programming language for data science and machine learning projects mainly due to the extensive libraries, community support and ease of use. Python language can be used on any modern computer operating system. It can be used for processing text, numbers, images, scientific data and just about anything else you might save on a computer. It is used daily in the operations of the Google search engine, the video-sharing website YouTube, NASA and the New York Stock Exchange.

Python is an interpreted language. This means that it is not converted to computer-readable code before the program is run but at runtime. In the past, this type of language was called a scripting language, intimating its use was for trivial tasks. However, programming languages such as Python have forced a change in that nomenclature. Increasingly, large applications are written almost exclusively in Python. It is a suitable language that bridges the gaps between business and developers. Subsequently, it takes less time to bring a Python program to market compared to other languages such as C#/Java. Additionally, there are a large number of python machine learning and analytical packages. A large number of communities and books are available to support Python developers. Nearly all types of applications, ranging from forecasting analytical to UI, can be implemented in Python.

Python works as follows:

* + A Python virtual machine is created where the packages (libraries) are installed. Think of a virtual machine as a container.
  + The python code is then written in .py files
  + C Python compiles the Python code to bytecode. This bytecode is for the Python virtual machine.
  + When you want to execute the bytecode then the code will be interpreted at runtime. The code will then be translated from the bytecode into the machine code. The bytecode is not dependent on the machine on which you are running the code. This makes Python machine-independent

Modules in python:

* + Python is shipped with over 200 standard modules.
  + A module is a component that groups similar functionality of your python solution.
  + Any python code file can be packaged as a module and then it can be imported.
  + Modules encourage componentized design in your solution.
  + They provide the concept of namespaces to help you share data and services.
  + Modules encourage code reusability and reduce variable name clashes.

Packages in python:

* + Package is a directory of modules.
  + If your Python solution offers a large set of functionalities that are grouped into module files, then you can create a package out of your modules to better distribute and manage your modules.
  + Packages enable us to organize our modules better which helps us in resolving issues and finding modules easier.
  + Third-party packages can be imported into your code such as pandas/sci-kit learn and tensor flow to name a few.
  + A package can contain a large number of modules.

**Chapter 3**

**3. Detailed Procedures**

**3.1 Data-Preparation:**

#### 3.1.1 Importing Required Libraries

#### To perform the analysis and modeling for the TVS Loan Default Prediction project, we first imported the necessary libraries. These libraries are essential for numerical operations, data manipulation, data visualization, and managing warnings in the code.

 NumPy: Used for numerical operations, this is foundational for data analysis.

 Pandas: Utilized for data manipulation and analysis, providing data structures like DataFrames.

 Matplotlib: A plotting library used for creating static, interactive, and animated visualizations.

 Seaborn: Built on top of Matplotlib, Seaborn is used for making statistical graphics.

 Warnings: The warnings library is used to manage and suppress warnings that might clutter the output during analysis.

**3.1.2 Importing the Dataset**

The dataset, stored in a CSV file named TVS.csv, was imported into a Pandas DataFrame for analysis. During the import, we specified custom missing value indicators to ensure that any irregular representations of missing values were correctly identified and handled.

* Reading the CSV File: The pd.read\_csv function is used to read the CSV file into a DataFrame. This function provides flexibility in handling various data formats and missing values.
* Custom Missing Value Indicators: Specifying na\_values ensures that custom indicators for missing values (e.g., '@', '#', '$', '?', ' ') are correctly interpreted as NaN by Pandas.
* Displaying the Data: The df.head() function is used to display the first five rows of the DataFrame, providing an initial glimpse into the dataset.

By setting up the environment with the necessary libraries and importing the dataset correctly, we laid a strong foundation for subsequent analysis and modelling.

**3.1.3 Data Dictionary**

This dictionary provides detailed information about the variables included in the TVS\_Loan\_dataset. Data consists of 119528 rows, 32 columns.

Name-TVS Loan Default.

Source-Kaggle.com

Purpose- To identify defaulters, which assists the bank in determining which loans are at risk and how to mitigate them.

**Table 3.1.3 Data Dictionary**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **S.NO** | **Variable Name** | **Description** | **Data Type** | **Missing Values**  **percentage** | **Outlier percentage** | | 1. | V1 | Customer ID | int64 | 0.000000 | 0.000000 | | 2. | V2 | If a customer has bounced in first EMI(1:Bounced, 0: Not bounced) | int64 | 0.000000 | 2.837829 | | 3. | V3 | Number of times bounced in recent 12 months | int64 | 0.000000 | 3.734690 | | 4. | V4 | Maximum MOB (Month of business with TVS Credit) | float64 | 28.846797 | 1.596279 | | 5. | V5 | Number of times bounced while repaying the loan | float64 | 28.846797 | 5.150258 | | 6. | V6 | EMI | float64 | 28.846797 | 0.633324 | | 7. | V7 | Loan Amount | float64 | 28.846797 | 0.078643 | | 8. | V8 | Tenure | float64 | 28.846797 | 3.175825 | | 9. | V9 | Dealer codes from where customer has purchased the Two wheeler | float64 | 28.846797 | 9.285690 | | 10. | V10 | Product code of Two wheeler (MC : Motorcycle , MO : Moped, SC : Scooter) | object | 28.846797 | 0.000000 | | 11. | V11 | No of advance EMI paid | float64 | 28.846797 | 5.365270 | | 12. | V12 | Rate of interest | float64 | 28.846797 | 2.110802 | | 13. | V13 | Gender (Male/Female) | object | 28.846797 | 0.000000 | | 14. | V14 | Employment type (HOUSEWIFE:housewife, SELF:Self-employed, SAL:Salaried, PENS:Pensioner, STUDENT Student) | object | 28.846797 | 0.000000 | | 15. | V15 | Resident type of customer | object | 29.613982 | 0.000000 | | 16. | V16 | Date of birth |  |  |  | | 17. | V17 | Age at which customer has taken the loan | float64 | 28.846797 | 0.000000 | | 18. | V18 | Number of loans | int64 | 0.000000 | 10.820059 | | 19. | V19 | Number of secured loans | int64 | 0.000000 | 13.196071 | | 20. | V20 | Number of unsecured loans | int64 | 0.000000 | 11.158892 | | 21. | V21 | Maximum amount sanctioned in the Live loans | float64 | 69.357807 | 2.929857 | | 22. | V22 | Number of new loans in last 3 months | int64 | 0.000000 | 0.000000 | | 23. | V23 | Total sanctioned amount in the secured Loans which are Live | float64 | 83.869052 | 1.598789 | | 24. | V24 | Total sanctioned amount in the unsecured Loans which are Live | float64 | 84.080717 | 1.401345 | | 25. | V25 | Maximum amount sanctioned for any Two wheeler loan | float64 | 12.600395 | 0.544642 | | 26. | V26 | Time since last Personal loan taken (in months) | float64 | 88.763302 | 1.451543 | | 27. | V27 | Time since first consumer durables loan taken (in months) | float64 | 82.905261 | 1.583729 | | 28. | V28 | Number of times 30 days past due in last 6 months | int64 | 0.000000 | 19.792015 | | 29. | V29 | Number of times 60 days past due in last 6 months | int64 | 0.000000 | 15.967807 | | 30. | V30 | Number of times 90 days past due in last 3 months | int64 | 0.000000 | 10.680343 | | 31. | V31 | Tier;(Customer’s geographical location) | object | 0.000000 | 0.000000 | | 32. | V32 | Target variable ( 1: Defaulters / 0: Non-Defaulters) | int64 | 0.000000 | 2.186935 | |

The data used for this project was got from Kaggle platform. The dataset used in this study contains 119528 records and 32 variables, Among the variables,26 are numerical and 6 are categorical variables. The target variable in modelling is "Default", which is binary with values 0 and 1. A value of 0 represents a client who has paid off their loan, while a value of 1 indicates a loan default.

**3.2 Exploratory Data Analysis (EDA):**

Exploratory Data Analysis (EDA) is a crucial step in understanding the dataset, identifying patterns, detecting anomalies, and uncovering insights. In this section, we explore the data types, handle missing values, identify outliers, and prepare the data for modelling.

**3.2.1 Exploring Data Types and Conversions**

First, we examined the data types of all columns in the dataset to understand the nature of each feature.

The dataset contains a mix of integer, float, and object data types. We identified the following categories:

- Numerical Columns: V4-V9, V12, V17-V21, V23-V30

- Categorical Columns: V2, V3, V10, V11, V13-V15, V31, V32

- Constant Column: V22 (only one unique value)

We converted the relevant object columns to categorical data types for better understanding and analysis.

**3.2.2 Exploring Unique Values**

We analysed the unique values in both numerical and categorical columns.

Key observations:

- Columns like V2, V3, V10, V11, V13, V14, V15, V31, and V32 are categorical.

- Column V22 is constant and doesn't provide useful information.

- High cardinality in V16 (dates) suggests it should be dropped.

**3.2.3 Exploring Duplicate Values**

We checked for duplicate rows in the dataset to ensure data quality. There were no duplicate rows in the dataset, ensuring data integrity.

**3.2.4 Exploring and Separating Categorical and Numerical Columns**

We separated the numerical and categorical columns for focused analysis.

**3.2.5 Exploring Missing Values and Their Proportions**

We analysed the missing values in each column and calculated their proportions to understand the extent of missing data.

Key findings:

- Columns V21, V23, V24, V26, and V27 have high missing values.

- Columns V4-V16 and V25 have moderate missing values.

- Columns V1-V3, V18-V20, V22, V28-V32 have no missing values.

**3.2.6 Exploring Outliers and Their Proportions**

We detected outliers in the numerical columns using the Interquartile Range (IQR) method.

Key findings:

- Columns V4-V9, V12, V18-V21, V23-V30 have significant outliers.

**3.2.7 Exploring Magnitudes of Data**

We examined the magnitudes of the numerical data to identify the need for scaling. Columns V6, V7, V9, V21, V24, and V25 require scaling due to their large magnitudes.

**3.2.8 Exploring Cardinality of Categorical Variables**

We assessed the cardinality of categorical variables to understand their distinct values.

Key observations:

- V10: 5 unique values

- V13: 2 unique values

- V14: 5 unique values

- V15: 3 unique values

- V31: 4 unique values

These insights guided our subsequent data cleaning and pre-processing steps, setting the stage for effective modelling.

#### 3.3 Data Cleaning:

The initial phase of data cleaning involved removing unnecessary columns, converting data types, treating missing values, and addressing multicollinearity. The cleaned dataset is now ready for further analysis and modeling.The output of segmentation is used as input for feature extraction in feature extraction. Following the input, the GLCM algorithm approaches will be used. It will detect properties like contrast, dissimilarity, homogeneity, energy, correlation, and entropy in the form of decimals using the GLCM approach. And this output will be used as a classification input.

**3.3.1 Dropping Unnecessary Columns**

The initial dataset contained an unnecessary column (`V1`) which was dropped to streamline the data.

**3.3.2 Type Conversion**

Appropriate data types were assigned to several columns for better data handling and analysis.

**3.3.3 Missing Values Treatment**

Columns with more than 30% missing values were dropped. Rows with more than 28% missing data were also removed to improve data quality.

**3.3.4 Missing Values Imputation**

Missing values were imputed to handle the remaining missing data efficiently.

**3.3.5 Dropping Unwanted Columns**

The column `V22` was dropped as part of the data cleaning process.

**3.3.6 Variance Inflation Factor (VIF) Calculation**

VIF was calculated to detect multicollinearity among the numeric features. High VIFs indicate high multicollinearity, which can adversely affect model performance.

VIF Analysis

- High VIFs (> 50): Features `V4`, `V5`, `V6`, `V7`, `V8`, and `V29` indicate severe multicollinearity.

- Moderate VIFs (15-25): Features `V12`, `V15`, `V17`, `V18`, `V19`, `V28` indicate moderate multicollinearity.

- Low VIFs (< 5):\*\* Features `V2`, `V3`, `V9`, `V11`, `V20`, `V32` indicate low multicollinearity.

Given the high multicollinearity detected, alternative modelling techniques like ridge regression or LASSO regression, which are less sensitive to multicollinearity, should be considered.

**3.3.7 Saving Cleaned Data**

The cleaned data was saved to a CSV file for further analysis.

**3.4 Data Cleaning and Preparation:**

This report outlines the steps undertaken to clean and prepare the data for analysis, focusing on handling outliers and applying feature scaling. The processes include:

1. Reading and initial inspection of the dataset

2. Identifying and treating outliers

3. Encoding categorical variables

4. Applying feature scaling

**3.4.1 Reading and Initial Inspection of the Dataset**

The dataset was read into a pandas DataFrame `df1` and inspected to understand the data types of each column:

The columns were then converted to appropriate data types, with several columns converted to the 'category' type:

**3.4.2 Identifying and Treating Outliers**

Outliers in the numerical columns were identified using the Interquartile Range (IQR) method. The outlier information was displayed, highlighting columns with a high proportion of outliers:

Outliers were treated by capping values below the 5th percentile and above the 95th percentile.

**3.4.3 Encoding Categorical Variables**

Categorical variables were one-hot encoded. The cleaned data was saved:

**3.4.4 Feature Scaling**

Two types of scaling were applied to the data:

1. Robust Scaling without Outlier Treatment

2. Standard Scaling with Outlier Treatment

**3.5 Hypothesis Testing:**

This presents the findings from hypothesis testing conducted on a dataset to identify significant features related to the target variable V32. The dataset contains both numerical and categorical variables. The goal is to determine which features have statistically significant differences between the classes of the target variable. The dataset, loaded from Data\_without\_Scaling\_&\_without\_Outliers\_Treatment.csv, includes various features denoted as V2 to V32. The target variable is V32, and its distribution is highly imbalanced with 83194 instances of class 0 and 1854 instances of class 1.

**3.5.1 Hypothesis Testing for Numerical Variables**

### Methodology:

### For numerical variables, the Mann-Whitney U test was employed due to the non-normal distribution of the data and the violation of homogeneity of variances assumption. The Shapiro-Wilk test confirmed that the features did not follow a normal distribution for both classes.

### Results:

### The Mann-Whitney U test was applied to the following numerical features:

* V6
* V7
* V9
* V17
* V25

#### Table 3.5.1 Test Statistics and Decisions:

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **U Statistic** | **p-value** | **Decision** |
| V6 | 1116746.5 | 0.7279 | Not Significant |
| V7 | 1049871.0 | 0.0015 | Significant |
| V9 | 1033009.5 | 0.0001 | Significant |
| V17 | 1255361.5 | 3.837×10−83.837 \times 10^{-8}3.837×10−8 | Significant |
| V25 | 1041724.0 | 0.0004 | Significant |

### Interpretation:

### The features V7V7V7, V9V9V9, V17V17V17, and V25V25V25 showed significant differences between the classes of V32V32V32, indicating they are important variables for distinguishing between the classes. V6V6V6 did not show a significant difference.

**3.5.2 Hypothesis Testing for Categorical Variables**

### Methodology:

### The Chi-Square test of independence was used for categorical variables to examine the relationship between each categorical feature and the target variable V32.

### Results:

### The Chi-Square test was applied to the following categorical features:

* V10
* V13
* V14
* V15
* V31

#### Table 3.5.2 Test Statistics and Decisions:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Chi-Square Statistic** | **p-value** | **Degrees of Freedom** | **Decision** |
| V10 | 83.1758 | <0.0001 | 4 | Significant |
| V13 | 19.3307 | <0.0001 | 1 | Significant |
| V14 | 41.5957 | <0.0001 | 4 | Significant |
| V15 | 4.7139 | 0.0947 | 2 | Not Significant |
| V31 | 209.2468 | <0.0001 | 3 | Significant |

### Interpretation:

### The features V10, V13, V14, and V31 were found to have significant associations with the target variable V32, suggesting they are influential in determining the classes of V32. V15 did not show a significant association.

**Chapter 4**

**4. Visualization**

Visualization plays a crucial role in enhancing the comprehensibility and impact of a report. By transforming raw data into graphical representations, visualizations make complex information more accessible and engaging. They help in quickly conveying key insights, identifying trends, and highlighting relationships within the data.

**4.1 Employment type having the highest percentage of defaulters:**

**Figure 4.1: Employment type having the highest percentage of defaulters**

Self-employed individuals have the highest default rate (2.38%).Pensioners and housewives, with lower default rates. Default rates by group: SELF (2.38%), SAL (1.80%), HOUSEWIFE (1.02%), PENS (0.72%), and STUDENT (1.34%).

**4.2 Distribution of employment types among all customers:**

**Figure 4.2: Distribution of employment types among all customers**

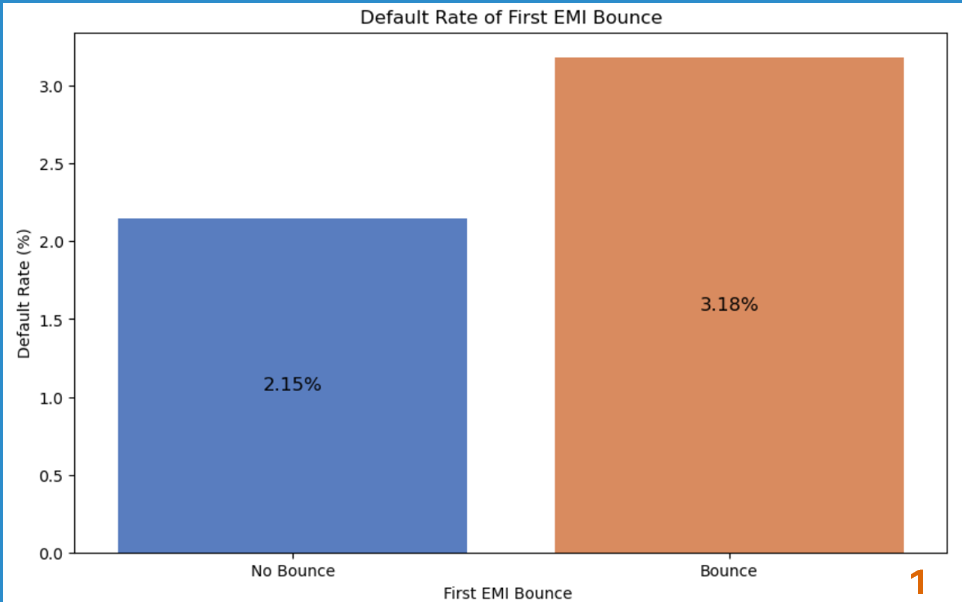
Majority of customers are self-employed (SELF), followed by salaried individuals (SAL). Households, students, and pensioners make up a smaller portion of the customer base.

**4.3 Age Group Showing the Highest Rate of Defaults:**

**Figure 4.3: Age Group Showing the Highest Rate of Defaults**

Default rates decrease with age, highest rates in younger individuals (18-25), lowest in older adults (55-75). Financial institutions should adjust risk assessments and credit policies based on age, offering tailored products to younger borrowers and investigating factors behind their higher default rates.

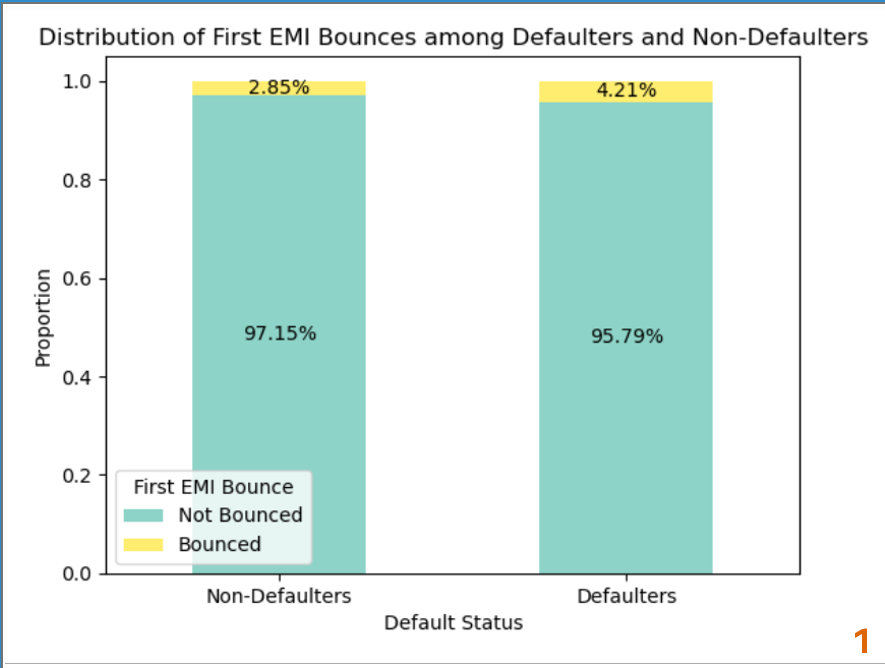
**4.4 First Emi Bounce Correlates with Loan Default Likelihood:**



**Figure 4.4: First Emi Bounce Correlates with Loan Default Likelihood**

Early payment issues are a strong indicator of future default. Customers who bounce their first EMI have a higher default rate (3.18%) than those who do not (2.15%). Only 2.88% fall into this high-risk category, but they pose a significant default risk.

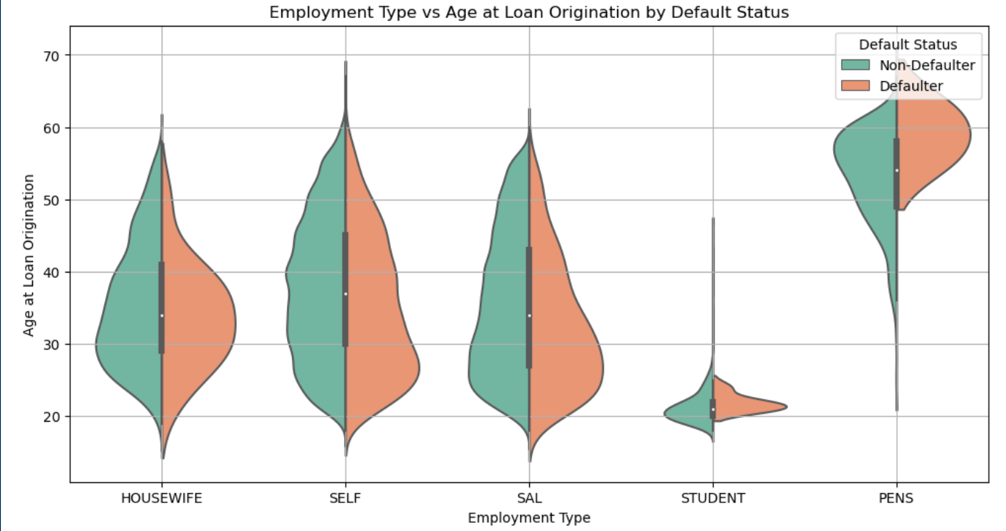
**4.5 First Emi Bounces among Defaulters and Non-Defaulters:**



**Figure 4.5: First Emi Bounces among Defaulters and Non-Defaulters**

The bounce rate for the first EMI is significantly higher for defaulters (4.207%) compared to non-defaulters (2.85%)

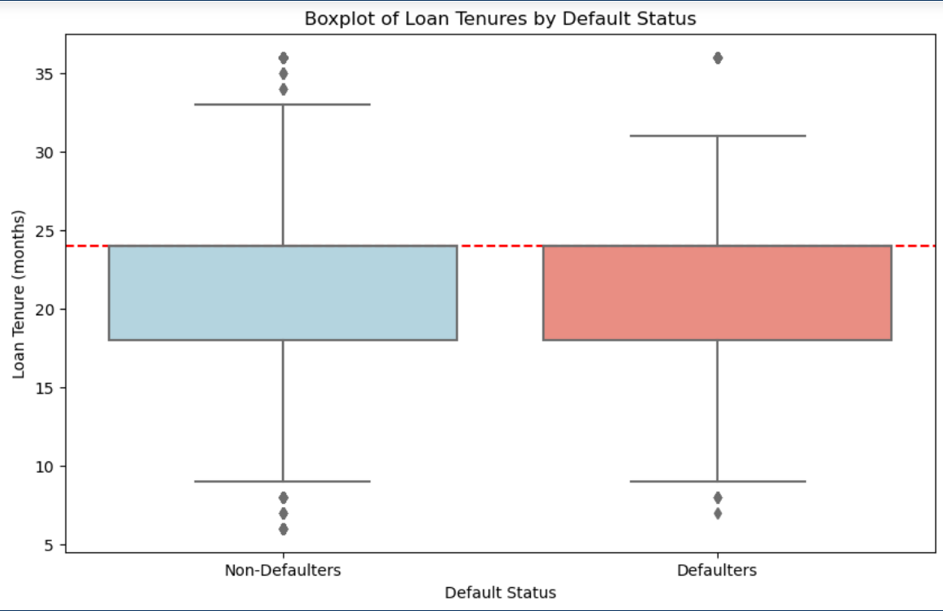
**4.6 Employment Type vs. Age at Loan Origination by Default Status:**



**Figure 4.6: Employment Type vs. Age at Loan Origination by Default Status**

Higher default risk in younger and self-employed individuals. Age and employment type are key in risk assessment.

**4.7 Impact of Length of Repayment Period on the Likelihood of the Default:**



**Figure 4.7: Impact of Length of Repayment Period on the Likelihood of the Default**

Loan tenure alone might not be a strong predictor of default status in this dataset.

**Chapter 5**

**5. Feature Engineering**

Feature engineering is the process of using domain knowledge to create new features from raw data that enhance the performance of machine learning models. It involves transforming and combining existing features to produce more informative ones, which can significantly improve the accuracy and robustness of predictive models. Effective feature engineering helps in capturing underlying patterns and relationships in the data that may not be evident in the raw features.

Here as a part od feature engineering we are dropping columns-'V5','V6', 'V8', 'V7', 'V12', 'V29', 'V28', 'V30' and replacing those with features-Total Repayment, Total Interest Paid, Total Past Due, Repayment Risk Score, Loan Burden, Default Risk Score, Loan\_Affordability\_Index. The conversions are as follows-

* Total Repayment=EMI×Tenure\_Months
* Total Interest Paid=[Loan Amount×Tenure​×Rate of Interest]/100
* Total Past Due=Times 30+ days Past due(6 mo) + Times 60+ Days Past Due (6 Mo.)​ + Times 90+ Days Past Due (3 Mo.)
* Repayment Risk Score=(Times 30+ days Past due(6 mo) \* 2) + (Times 60+ Days Past Due (6 Mo.) \* 1.5) + (Times 90+ Days Past Due (3 Mo.) \* 3) + (Times bounced repaying loan \*2)
* Loan Burden=(Loan Amount/Tenure)​ + EMI
* Default Risk Score=[Times 30+ days Past due(6 mo) + Times 60+ Days Past Due (6 Mo.) + Times 90+ Days Past Due (3 Mo.) + Times bounced repaying loan ]/4
* Loan\_Affordability\_Index =[EMI / Rate of Interest]

**Chapter 6**

**6. Feature Selection**

Feature selection is a crucial process in data analysis and machine learning that involves identifying the most relevant variables or features from a dataset. By selecting only the most important features, we can enhance the performance of predictive models, reduce computational cost, and improve the interpretability of the results. Effective feature selection leads to more accurate, efficient, and generalizable models.

**6.1 Using VIF:**

Variance Inflation Factor (VIF) is a statistical measure used to detect multicollinearity in a dataset. Multicollinearity occurs when two or more predictor variables are highly correlated, leading to redundancy and instability in regression models. High multicollinearity can inflate the standard errors of the coefficients, making it difficult to assess the true relationship between predictors and the target variable. VIF quantifies how much the variance of a regression coefficient is inflated due to collinearity with other predictors.

* VIF = 1: No correlation between the predictor and other variables.
* 1 < VIF < 5: Moderate correlation that is usually acceptable.
* VIF ≥ 5: High correlation indicating multicollinearity and requiring further investigation.
* VIF > 10: Severe multicollinearity that necessitates corrective action.

Using this VIF measure dropped columns having VIF >10 and those columns include- 'Default\_Risk\_Score', 'Total\_Repayment', 'RepaymentRiskScore', 'V25', 'V18', 'V4'.

**6.2 Using Lasso CV Method:**

LassoCV (Least Absolute Shrinkage and Selection Operator with Cross-Validation) is a powerful technique for feature selection in machine learning. It is an extension of the Lasso regression method, which not only helps in regularizing the model to prevent overfitting but also performs feature selection by shrinking less important feature coefficients to zero. LassoCV enhances this process by incorporating cross-validation to automatically find the optimal value of the regularization parameter, ensuring robust and reliable feature selection.

Uses Lasso CV regularization technique for feature selection and while predicting Target on selected features using Logistic Regression provides 97.7923% accuracy.Total features – 20. 'V2', 'V3', 'V9', 'V11', 'V17', 'V19', 'V20', 'Total\_Interest\_Paid', 'Total\_Past\_Due', ‘LoanBurden', 'V10\_MO', 'V10\_SC', 'V10\_TL', 'V13\_MALE', 'V14\_SELF', 'V14\_STUDENT', 'V15\_RENT', 'V31\_TIER 2', 'V31\_TIER 3', 'V31\_TIER 4'.

**6.3 Using RFECV Method:**

Recursive Feature Elimination with Cross-Validation (RFECV) is an advanced feature selection technique that helps identify the most relevant features for a predictive model. RFECV recursively removes the least important features and uses cross-validation to evaluate the performance of the model at each step. This method ensures that the selected features contribute maximally to the model's accuracy while minimizing redundancy and overfitting.

Uses Recursive feature elimination technique for feature selection and while predicting Target on selected features using Logistic Regression provides 97.7963% accuracy. Total features: 17- 'V2', 'V3', 'V9', 'V11', 'V17', 'V19', 'V20', 'Total\_Interest\_Paid', 'Total\_Past\_Due', 'V10\_MO', 'V10\_TL', 'V13\_MALE', 'V14\_SELF', 'V14\_STUDENT', 'V31\_TIER 2', 'V31\_TIER 3', 'V31\_TIER 4'.

**6.4 Using Tree Method:**

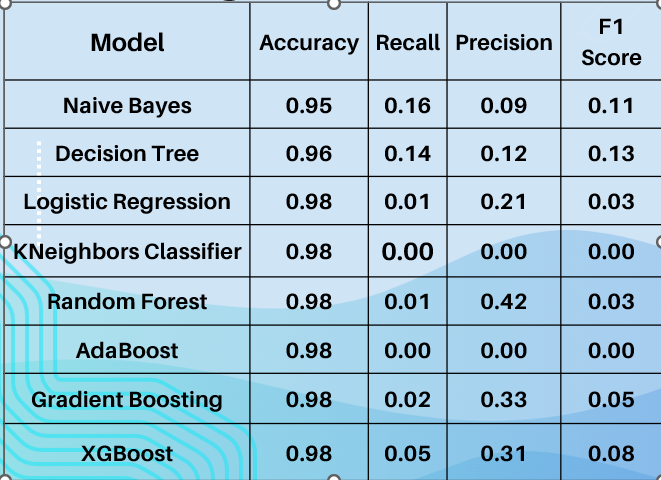
Tree-based feature selection is a powerful method used to identify the most relevant feature s for predictive modeling. Decision trees, random forests, and gradient boosting machines inherently perform feature selection by evaluating the importance of each feature during the model training process. This approach leverages the structure of tree-based algorithms to rank features based on their contribution to the model's predictive performance, allowing for the identification of key variables that significantly impact the target variable.

Uses Extra tree Classifier method for feature selection and while predicting Target on selected features using Logistic Regression provides 97.7843% accuracy. Total features:9- V3', 'V9', 'V17', 'V19', 'V20', 'Total\_Interest\_Paid', 'Total\_Past\_Due', 'LoanBurden', 'Loan\_Affordability\_Index'.

**Chapter 7**

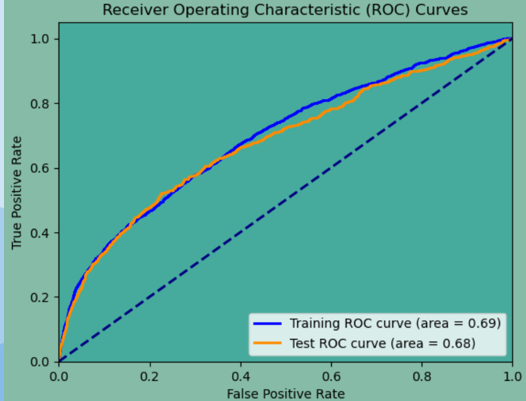
**7. Model Building**

**7.1 Base Model Evaluation**:

****

**Figure 7.1: Base Model Evaluation**

**7.1.1 ROC-AUC Curve for Train and Test Data:**

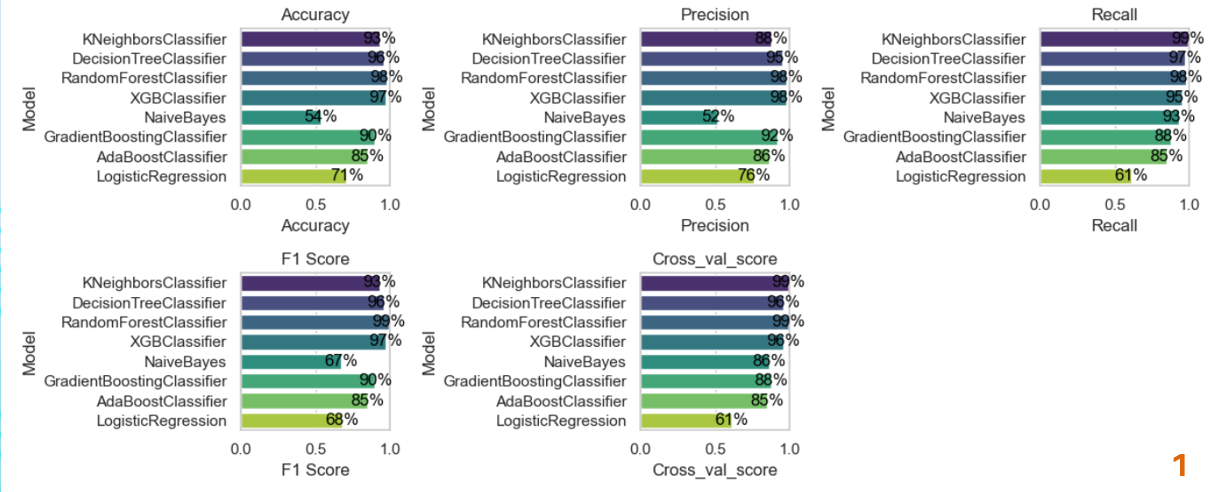


**Figure 7.1.1: ROC-AUC Curve for Train and Test Data**

**7.2 Model Using SMOTE Technique:**

Synthetic Minority Over-sampling Technique (SMOTE) is a widely used method to address class imbalance in datasets. Class imbalance occurs when the number of instances in one class significantly outnumbers those in another class, which can lead to biased model performance. SMOTE helps to balance the dataset by generating synthetic samples for the minority class, thereby improving the model’s ability to learn and predict the minority class effectively.

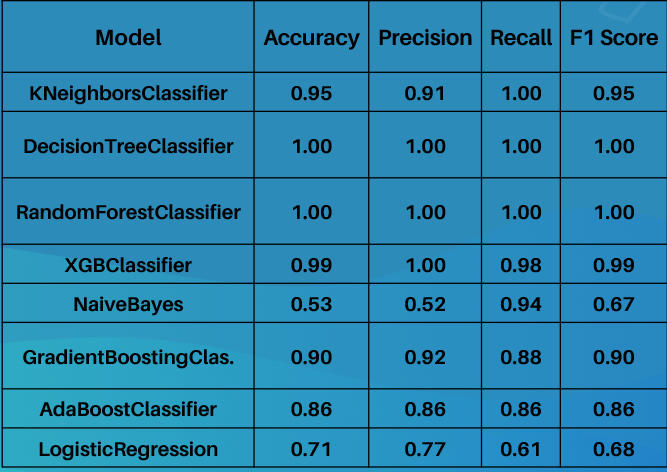
Results after using smote techniques for different algorithms like-KNeighbors classifier, DecisionTreeClassifier, RandomForestClassifier, XGBClassifier, NaiveBayes, GradientBoostingClassifier, AdaBoostClassifier, LogisticRegression-



**Figure 7.2: Results after using SMOTE techniques**

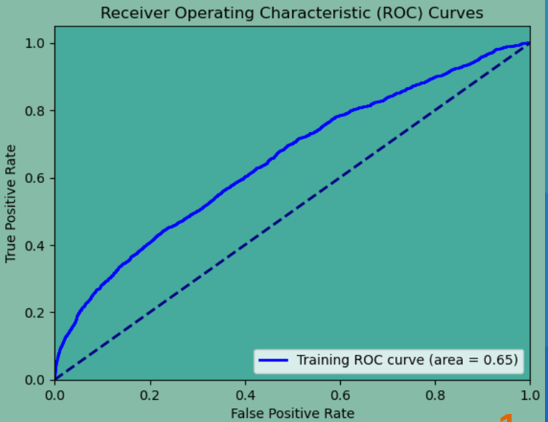
**7.3 Model Evaluation Using SMOTE-TOMEK:**

**7.3.1 For Train data:**



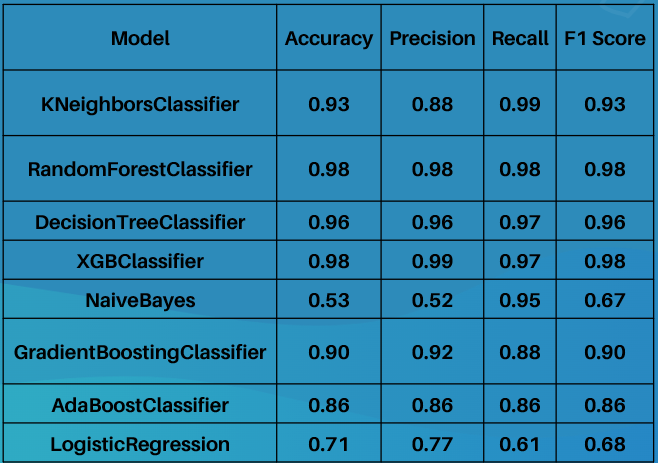
**Figure 7.3.1: Results after using SMOTE-TOMEK techniques for Train Data**

**7.3.1.1 ROC-AUC Curve:**



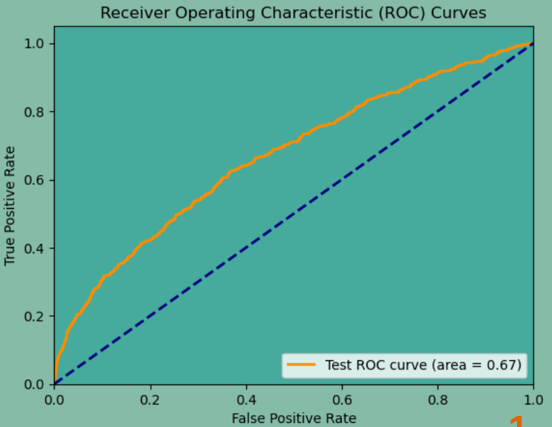
**Figure 7.3.1.1: ROC-AUC Curve for Train Data**

**7.3.2 For Test data:**



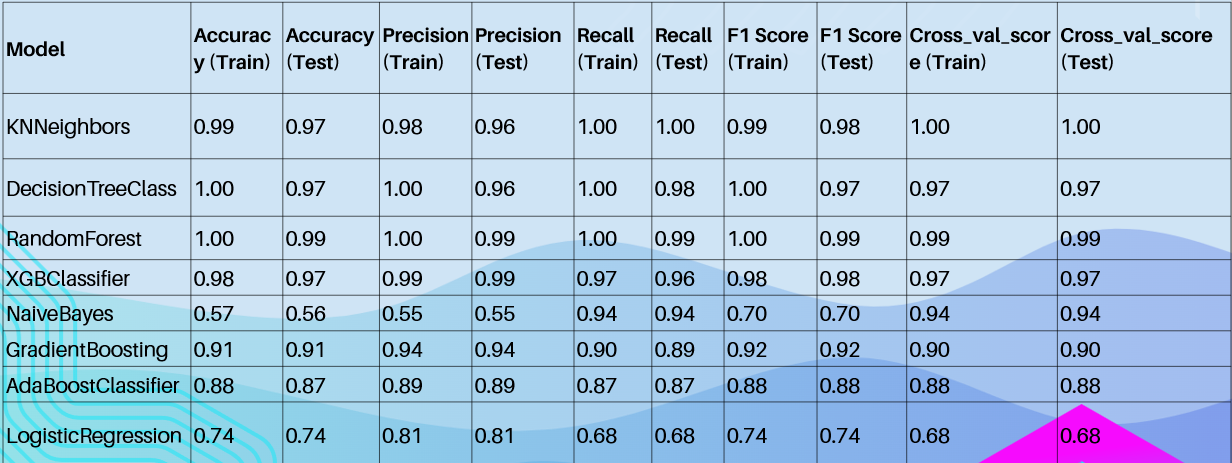
**Figure 7.3.2: Results after using SMOTE-TOMEK techniques for Test Data**

**7.3.2.1 ROC-AUC Curve:**



**Figure 7.3.2.1: ROC-AUC Curve for Test Data**

**7.4 Model Evaluation Using SMOTENN:**



**Figure 7.4: Results after using SMOTENN techniques for Train and Test Data**

**Chapter 8**

**8. Challenges Faced During Model Building:**

1. Missing Values: Managing missing data without introducing bias or losing valuable information.
2. Imbalanced Data: Handling unequal class representation that can lead to biased models. Techniques like SMOTE, SMOTE\_TOMEK, SMOTENN are used to balance the data.
3. Feature Selection: Choosing the most relevant features to prevent overfitting and improve model performance.
4. Feature Scaling: Standardizing or normalizing data to ensure all features contribute equally to the model.
5. Overfitting vs. Underfitting: Striking a balance between model complexity to avoid overfitting (too complex) or underfitting (too simple).
6. Hyperparameter Tuning: Finding the best parameters for models to boost performance.
7. Cross-Validation: Ensuring the model generalizes well to unseen data using techniques like k-fold cross-validation.
8. Evaluation Metrics: Using appropriate metrics (accuracy, precision, recall, F1 score) for a comprehensive evaluation of model performance.
9. Training Time: Managing the time required to train models, especially ensemble methods like RandomForest and XGBClassifier, which can be computationally intensive.
10. Memory Usage: Ensuring the model can be processed within available memory, particularly with large datasets.
11. Model Interpretability: Balancing model complexity and interpretability. Some models, like Decision Trees, are easier to explain, while others, like XGBClassifier, are more complex and harder to interpret for stakeholders.

### REFERENCES

[1]. Aiyar, C., & Tiberghien, Y. (2005). “Formal versus informal finance: Evidence from India”. Journal of Development Economics, 78(1), 83-106.

[2]. Demirguc-Kunt, A., & Klapper, L. (2012). “Financial inclusion in developing countries”. The World Bank.

[3]. Singh, A., et al. (2017). “Machine learning for credit risk assessment”. International Journal of Computer Applications, 169(9), 16-22

[4]. Huang, C., et al. (2018). “Deep learning for credit risk assessment”. In Proceedings of the 2018 ACM International Conference on Computing Advances with Financial Applications (pp. 610-617).

[5]. Lessmann, K., et al. (2015). “Loan default prediction with social network features”.

[6]. World Bank. (2021). Global Findex Database. <https://datacatalog.worldbank.org/dataset/global-financial-inclusion-global-findex-database>

[7]. Loan Portfolio Risk Assessment: A Guide for Lenders (defisolutions.com)

[8]. Top 10 Binary Classification Algorithms [a Beginner’s Guide] | by Alex Ortner | Towards Data Science

[9]. (PDF) Determining key factors influencing SMEs’ performance: A systematic literature review and experts’ verification (researchgate.net)

[10]. Decision Tree Ensemble Model. A decision tree ensemble model is a… | by Pankaj Pandey | Medium

[11]. Fusion of linear and non-linear dimensionality reduction techniques for feature reduction in LSTM-based Intrusion Detection System - ScienceDirect

[12].[www.braced.org/contentAsset/raw-data/b4a59a03-6fed-4928-a991eccb588f05a5/attachmentFile](http://www.braced.org/contentAsset/raw-data/b4a59a03-6fed-4928-a991eccb588f05a5/attachmentFile)

[13]. The Rise of Personal Loans and the Need for Risk Assessment

(<https://www.experian.com/corporate/credit-risk>): This article from Experian discusses the surge in personal loans and emphasizes the growing need for robust risk assessment models to mitigate defaults.

[14]. Machine Learning in Lending: How AI is Transforming Loan Approvals (<https://www.forbes.com/sites/forbesfinancecouncil/2021/12/17/how-ai-and-ml-are-changing-finance-in-2022/>): This Forbes article explores how Machine Learning is revolutionizing the lending industry by enabling faster, more accurate loan approvals and risk assessments.

[15]. Why Credit Scores Aren't Enough: A Look at Alternative Data in Lending (<https://www.americanbanker.com/creditunions/news/lenders-find-more-uses-for-alternative-credit-data>): This American Banker piece dives into the limitations of traditional credit scores and explores the value of alternative data sources for a more comprehensive borrower evaluation.

[16]. Consumer Financial Protection Bureau (CFPB) - Personal Loans (<https://www.consumerfinance.gov/>): This resource from the CFPB provides an overview of personal loans and consumer rights. It emphasizes the importance of responsible lending practices, which your project can contribute to by helping TVS Credit assess risk more accurately.

[17]. Can Machine Learning Help Reduce Loan Defaults in Developing Economies? (<https://www.sciencedirect.com/science/article/abs/pii/S0020025522012543>): This World Bank article explores the potential of Machine Learning to reduce loan defaults in developing economies. While the focus is broader, it highlights the global relevance of projects aimed at improving loan risk assessment.

[18]. Dos and Don'ts of Personal Loan Cross-Selling (<https://www.greenpath.com/>): This blog post from Greenpath Financial Wellness offers insights into effective cross-selling strategies for personal loans. Understanding these strategies can provide context for how your project's risk assessment model can be integrated into TVS Credit's cross-selling process.