JAGDISH SHETH SCHOOL OF MANAGEMENT
AACSB Accredited, Formerly IFIM Business School
AACSB ACCREDITED

REQUEST FOR PROBLEM

INDIVIDUAL WHITE PAPER

TOPIC - **To develop a model which can predict the probability of default while giving loans to customers.**

SUBMITTED BY :

Name – Anshika Panwar
Registration no.- 2022SEPVPGP0007
Course – MBA (2022-2024)

TABLE OF CONTENT

BACKGROUND

1.1 Industry and Company Overview

Industry Overview

Banking and Financial services are a broadly defined sector covering all forms of money management and exchange. This refers to companies, for example, banks, insurance institutions, investment firms, and payment processors, which supply finance products and services to individuals and businesses.

As new technologies and innovations are developing, the financial services industry is constantly evolving. There has been a growing trend towards digitalization, as more and more people are using their smartphones and computers to manage their finances in recent years. As a result, the emergence of new financial services providers, such as fintech startups.

The financial services industry is an important part of the global economy. It provides businesses with the financial resources they need to grow and invest, as well as helps people save for retirement and other financial goals. The sector also produces millions of jobs worldwide, which is a major source of employment.

Market Size:

The global financial services software market size was valued at $118.65 billion in 2021 and is projected to reach $282.71 billion by 2031, growing at a CAGR of 9.2% from 2022 to 2031.

There are three general types of financial services:

- Personal,
- Consumer, and
- Corporate.

These three categories encompass the major players and influencers for companies and organizations trying to climb the ladder of the industry.

Personal Finance:

Personal finance simply means managing your money, such as how much you earn, saving and spending, and thinking about bills and future plans. This is important at every stage of life, from buying your first car to planning for retirement.

When people choose a bank or other money-related company, they often want a company that provides help with their personal finances, such as a financial advisor. As money management is increasingly done online, people want banks to allow them to control their accounts from the Internet and mobile apps.

Younger people who are comfortable with technology like banks have tools for managing personal finances.

Some popular companies in this area are:

- Chime
- N26
- VARO
- CLEO

Consumer Finance:

From investing in real estate to paying for college tuition, consumer credit helps people buy products and services in installments over a set period of time. The consumer financial services market is made up of major players including credit card services, mortgage lenders, and personal and student loan services.

Some popular consumer finance services include:

- American Express
- Ally Financial
- LendingTree

Corporate Finance:

Corporate finance is a general term to describe a company's financial activities, such as funding, capital structure, actions to increase the value of the company, and resource allocation tools. force.

Jobs in the corporate finance industry include accountants, analysts, treasurers, and investor relations specialists, all of whom work to maximize a company's value.

Three key sources of funding in corporate finance include:

- Private Equity
- Venture capital
- Angel investor

Company Overview

HSBC is one of the world’s largest banking and financial services organizations. Through our three global businesses—Wealth and Personal Banking, Commercial Banking, and Global Banking & Markets—we provide services to almost 39 million customers worldwide. Our network spans 62 nations and territories throughout North America, Latin America, Europe, Asia, the Middle East, and Africa.

Purpose of HSBC:

Opening up a world of opportunity – explains why we exist. We’re here to use our unique expertise, capabilities, breadth, and perspectives to open up new kinds of opportunities for our customers. We’re bringing together the people, ideas, and capital that nurture progress and growth, helping to create a better world – for our customers, our people, our investors, our communities, and the planet we all share.

History of HSBC:

HSBC was born from one simple idea – a local bank serving international needs. In March 1865, HSBC opened its doors for business in Hong Kong, helping to finance trade between Europe and Asia.
For more than 150 years, we have provided our customers with support. We currently work with almost 39 million individuals, families, and businesses throughout 62 nations and territories.

The previous 150 years of experience have shaped the way HSBC is today. Looking back at our history helps to illustrate why we value strong capital, careful cost management, and developing lasting partnerships with clients.

The bank has survived change in all its manifestations, including revolutions, economic crises, and the introduction of new technology. HSBC is able to take on the problems of the twenty-first century thanks to the consequent corporate identity.

HSBC's business segments are:

- Wealth and Personal Banking: Individuals and families can access banking and financial services from this section.
- Commercial Banking: This sector offers financial and banking services to companies.
- Global Banking and Markets: This business unit offers institutional clients investment banking, securities trading, and other financial services.

HSBC's business model is based on providing a wide range of financial services to a global customer base. The bank generates revenue from a variety of sources, including:

- Interest income from loans and deposits
- Fees from financial services such as investment banking and insurance
- Trading income from securities and commodities

Here are some of the key facts about HSBC:

- Founded in 1865 as The Hongkong and Shanghai Banking Corporation
- Headquartered in London, England
- Operates in over 60 countries and territories
- Employs over 230,000 people
- Total assets of $2.95 trillion (as of December 2021)
- The market capitalization of $166 billion (as of March 8, 2023)
- Listed on the London Stock Exchange, the Hong Kong Stock Exchange, and the New York Stock Exchange

1.2 Statement of Purpose (SOP)

The creation and implementation of an enhanced underwriting model and a framework for acquisition risk management is the problem's Statement of Purpose. With this, HSBC hopes to reduce the steadily rising default rates on its portfolio of personal loans. The project aims to improve lending decisions, decrease defaults, maximize loan affordability, and maintain the bank's financial stability and image as a responsible lender by precisely determining the affordability and reliability of borrowers.

2. PROBLEM FRAMING

2.1 Broader Description of the Problem

The problems faced by HSBC are:

1. The first problem is that HSBC's personal loan portfolio is becoming riskier. This indicates that the bank is making loans to clients who are more prone to miss payments on their debts. This can be because of the borrower's income, employment situation, or credit history, among other things.
2. The second problem is that the risk analytics team needs to identify the key factors that are contributing to the increasing default rate. They can create a more effective underwriting model with the use of this. The borrower's income, employment position, credit history, debt-to-income ratio, and housing costs are some of the important variables they may take into account.
3. The third issue is that the group must create an underwriting model that can determine whether or not to approve a loan for an applicant. A borrower failing on a loan should be precisely predicted by the model, which should be built around the important variables they have discovered.
4. The fourth problem is that the team needs to create a framework to assess the effectiveness of the underwriting model. This framework ought to track how the model has affected the acquisition framework in terms of things like the volume of loans approved and the default rate.

HSBC faces a pressing issue within its personal loans portfolio, particularly concerning its retail unsecured term loans, a key lending product. These term loans have a defined term and must be repaid by the borrower in Equated Monthly Instalments (EMIs) by the due dates. Borrowers are classed as delinquent after failing to make an EMI payment, and defaulters after missing four consecutive payments.

In this portfolio, HSBC has noticed a worrying tendency of rising default rates over time. In response, the Risk Analytics team has been given two main tasks: first, to pinpoint the underlying factors behind the portfolio's increasing riskiness; and second, to improve the bank's acquisition risk management procedures. The team is tasked with creating an underwriting model catered to HSBC's particular requirements in order to accomplish these aims.

The default rate on HSBC's portfolio of personal loans is rising. The risk analytics team has been tasked with pinpointing the main causes of the portfolio's rising riskiness. By creating an underwriting model that can evaluate the dependability and affordability of borrowers, they also want to improve their acquisition risk management.

The envisioned underwriting model is expected to serve a dual purpose.

- Firstly, it should evaluate the affordability of the loan product for each applicant. This entails assessing an individual's financial capacity to comfortably manage EMI obligations.
- Secondly, the model should gauge the reliability of the borrower concerning a predefined set of origination criteria. This reliability assessment may incorporate various factors, including the applicant's credit history and, notably, external data sources like credit bureau information.

2.2 Visualization of the Situation (situation analysis)

HSBC, a well-known financial institution, is currently dealing with a problematic issue in its portfolio of personal loans, especially with regard to its retail unsecured term loans. Equated Monthly Instalment (EMI) repayments and set tenures define these term loans. An EMI missed results in delinquency, and after four consecutive missed EMIs, the borrower is regarded as defaulting.

Challenges:
Over recent years, HSBC has experienced a worrying trend of increasing default rates within this portfolio. This situation necessitates a thorough analysis and strategic response from the Risk Analytics team to mitigate risks and enhance the bank’s lending practices.

Cause:
A number of elements, such as the following, may be causing the rising default rate:

- Low income: If borrowers with low incomes encounter an unforeseen financial hardship, such as a job loss or medical emergency, they may be more prone to default on their debts.
- Unemployment: Because they do not have the income to make their payments, borrowers who are unemployed are more likely to fail on their debts.
- Poor credit history: Because they may have a history of making late payments or defaulting on prior loans, borrowers with a low credit history are more likely to go into default on their loans.
- High debt-to-income ratio: Borrowers with a high debt-to-income ratio are more likely to default on their loans, as they may not have enough disposable income to make their payments.
- High housing costs: Borrowers who have high housing costs may be more likely to default on their loans, as they may not have enough money left over to make their loan payments.

Solution:
HSBC can solve these issues by creating an underwriting model that would enable them to more accurately evaluate the riskiness of potential borrowers. The model must be based on the crucial elements that have been discovered and be able to precisely forecast the probability of a borrower defaulting on a loan.

Visualization:

The following visualization can be used to illustrate the situation:

- The graph can be used to demonstrates that throughout the previous few years, the default rate has been progressively rising. To maintain its financial stability, HSBC needs to address this trend.
- The graph can be used by the risk analytics team to pinpoint the years when the default rate started to rise and any potential contributing variables. In order to lower the default rate in the future, an underwriting model can be created using this information.

Impact:

The portfolio of personal loans held by HSBC has been significantly impacted by the rising default rate. In addition to suffering costs associated with default and collection, the bank is losing money on loans that are not being repaid. Furthermore, the rising default rate harms HSBC's reputation and makes it harder to draw in new borrowers.

2.3 Articulation of the Situation (with respect to the problem)

The default rate on the portfolio of retail unsecured term loans held by HSBC Bank is rising. The risk analytics team has been tasked with pinpointing the main causes of the portfolio's rising riskiness. They also seek to enhance the way they manage acquisition risk. The risk analytics team is to create an underwriting model for them as a result. The borrower's affordability for the product and reliability for a certain set of originations should be evaluated using the underwriting model.

There are several elements that could be causing the rising default rate, such as:

- **Economic conditions:** The ongoing conflict in Ukraine, rising inflation, and supply-chain interruptions have all been a hindrance to the world economy. As a result, there is now greater unemployment and lower salaries, which makes it more challenging for borrowers to pay back their loans.
- **Lending practises have changed:** In recent years, HSBC Bank has relaxed its lending requirements. Due to this, banks are now providing money to customers who are more likely to default.
- **Fraud:** The loan sector has seen a rise in fraud. Due to this, some borrowers have taken out loans that they are unable to repay.

All of these elements must be carefully taken into account by the risk analytics team when creating the underwriting model. The model should be able to accurately forecast the likelihood that a borrower would default on their loan and should be based on the best data currently available.

For HSBC Bank, the underwriting model will be a useful tool. It will assist the bank in better risk management and lower the portfolio's default rate for retail unsecured term loans. The model will also assist the bank in lowering acquisition risk and luring in new clients who are less likely to default.

The risk analytics team can use the following specific actions to explain the scenario in relation to the issue:

- **Gather data:** The first stage is to compile information on the current lending portfolio. The debtors' income, work situation, and credit history should all be included in this data. It should also provide details on the loans, such as their sum, rates of interest, and terms.
- **Identify the key factors**: Finding the important elements that are causing the default rate to rise is the next stage. Analysing the data and searching for patterns can help with this.
- **Develop the underwriting model**: The risk analytics team can create the underwriting model after identifying the important factors. The model should be able to accurately forecast the likelihood that a borrower would default on their loan and should be based on the best data currently available.
- **Test the model**: Prior to being used to decide which loans to make, the model should be evaluated on a sample of data. This will support ensuring the model's accuracy and dependability.
- **Launch the model**: The model can be used to make loan decisions after it has been evaluated and tested. To make sure the model is still accurate and trustworthy, it should be reviewed continuously.

By following these steps, the risk analytics team can articulate the situation with respect to the problem and develop an underwriting model that can help HSBC Bank to reduce the default rate on its retail unsecured term loans portfolio.

2.4 Problem definition

The personal loans portfolio of HSBC Bank presents a significant problem, particularly with regard to its retail unsecured term loans. These loans, which are characterised by fixed terms and EMI repayments, have seen a concerning trend of rising default rates. In addition to putting the bank's financial viability in danger, this circumstance also makes its lending policies and risk management plans a source of concern. HSBC has started a large effort with two main goals to address this difficulty.

Its first goal is to pinpoint the fundamental causes of the personal loans portfolio's increasing riskiness. For the purpose of creating targeted risk mitigation strategies, understanding these aspects is essential.

Second, HSBC is creating a sophisticated underwriting model to improve its acquisition risk management. According to internal bank data, this model will analyse the loan applicants' affordability for the product and their dependability according to external data sources such credit bureau reports. The three primary parts of the project are the creation of an effect assessment framework, underwriting model development, and portfolio analysis. By making these efforts, HSBC hopes to provide its esteemed clients with ethical financial services while improving loan decisions, reducing risks, and ensuring the bank's long-term financial health.

2.5 Project objectives

The project's objectives are created to help HSBC Bank successfully address the rising default rates in its portfolio of personal loans and improve risk management in the following ways:

1. **Identify Risk Factors:**
   The first step is to pinpoint the main causes of the personal loans portfolio's increasing riskiness. This entails a thorough examination of the current lending portfolio, looking at crucial indicators such as the overall loan amount, portfolio volume, total number of defaults, and historical trends in default rates. The bank can create focused initiatives to efficiently minimize these risks by recognizing the underlying causes of the rising default rates.
2. **Strengthen Acquisition Risk Management:**
   HSBC wants to improve its acquisition risk management procedures in order to make sure that loans are only given to deserving candidates. The bank is concentrating on creating an advanced underwriting model to do this.

   Using this model, loan applicants will be evaluated on two key criteria:
   - **Affordability Assessment:** Using internal bank data, the underwriting algorithm will assess a borrower's capacity to pay back the loan product. With the use of this study, lenders may be sure that borrowers can comfortably handle their EMIs.
   - **Evaluation of Reliability:** The model will take into account external data sources, in particular information from credit bureaus, to assess the dependability of applicants. The applicant's creditworthiness and loan repayment history will be shown by this external data.
3. **Establish an Impact Assessment Framework:**
   An impact evaluation mechanism will be designed to guarantee the underwriting model's efficacy and alignment with the bank's business objectives.

   This framework will assess the model's effectiveness in numerous crucial areas, including:
   - **Business Impacts**: The underwriting model's effects on decision-making and risk management in the bank's lending practices will be assessed in terms of their effects on the business.
   - **Operational Efficiency:** The framework will evaluate how effective the model is at streamlining loan origination procedures, increasing efficiency, and lowering operational expenses.
   - **Customer experience**: It's essential to make sure that creditworthy applicants get quick approvals and have a great experience all the way through the loan application process. The framework will gauge how much the model has improved customer satisfaction.

PROJECT EXECUTION

BASEL PREREQUISITES

What is BASEL?

The Basel Accords, a collection of global banking norms and guidelines created by the Basel Committee on Banking Supervision (BCBS), are referred to as Basel in the context of credit risk. These agreements provide standards for banks to evaluate and manage a variety of risks, including credit risk, in order to ensure the stability and soundness of the global financial system.

Conceptual Framework for Personal Loan PD (Probability of Default) Development Model

Introduction:

The development of a Probability of Default (PD) model for personal loans is essential for risk assessment and regulatory compliance, particularly under the BASEL framework. This conceptual framework outlines the key components and considerations for building a robust PD model for personal loans

BASEL OVERVIEW:

1. **BASEL Portfolios:**

   Bank lending books are categorized into two main portfolios by BASEL: Retail and Commercial.

- **Retail category:** This portfolio includes loans to multiple debtors, where each debtor has a risk exposure of less than 1 million euros. It primarily serves households for their basic credit needs.
- **Commercial category:** Loans exceeding the €1 million limit set by the retail category are considered part of the commercial category. Commercial loans typically involve fewer accounts but greater risk.
- **Commercial Portfolio Types:** Commercial portfolios are divided into two main categories: small and medium enterprises (SMEs) and large companies.

Analysis:

Understanding portfolio breakdown is important for banks, as **BASEL** regulations treat retail and commercial lending differently in terms of risk assessment and capital requirements. This knowledge helps banks classify loans and allocate capital accordingly.

2. **Retail Sub-Portfolios:**

The retail portfolio has three sub-portfolios: Qualifying Revolving Retail Exposure (QRRE), Mortgage, and Other Retail Portfolio.

- **QRRE:** QRRE includes unsecured, revolving loans with no definite maturity date.
- **Mortgage:** Consists of secured loans with a set, usually lengthy repayment duration.
- **Other Retail:** This category includes secured and unsecured term loans, typically with shorter terms than mortgages.

Analysis:

Classifying retail loans into subcategories emphasizes that different types of retail loans have distinct risk profiles. This knowledge is essential for risk modeling and capital allocation purposes.

3. **Data Requirements:** Models should be developed using a minimum of 5 years of historical data, preferably including economic downturn periods.

- If a bank lacks 5 years of data, it can start with available data and update the model as more data becomes available or consider standardized approaches.
- Longer data vintages reflecting representative loss behavior can be used if available, but their representativeness should be tested, e.g., by examining bad rates over time.
- Changes in data or the business environment during the modeling period must be controlled for and analyzed, with impacts on the distribution documented.
- For major data architecture changes, check if key statistical moments of variables remain unchanged, possibly using descriptive analysis.

4. **Data Quality:** For stakeholders to use variables in a clear and unambiguous manner, data must adhere to strict standards and be well-documented with a data dictionary.To make sure that variables behave in a representative manner, we basically execute data quality checks, such as source-to-target mapping, first occurrence-last occurrence analysis, missing observation analysis, univariate analysis, and frequency distribution analysis.

5. **Model Development and Validation:**

- Develop PD models specific to each retail sub-portfolio (QRRE, Mortgage, Other Retail). Then, we will use appropriate modelling techniques such as logistic regression, machine learning, or other statistical methods.
- Validate models using out-of-sample data and assess their accuracy, discrimination, and calibration.
- Ensure model robustness by assessing the impact of changes in variables or economic conditions on PD predictions.

6. **Documentation and Reporting:**

- Document all aspects of model development, including data sources, modeling techniques, assumptions, and validation results.
- Regulatory Compliance:
- Ensure that the PD models we are developing align with BASEL requirements and any other relevant regulatory standards.
- Stay updated with regulatory changes and adapt the PD models accordingly to maintain compliance.

<u>Checking for the BASEL Criteria in the Loan_Stats dataset which is provided to us:</u>

1. **BASEL Portfolio:**

- Total Number of Observations -42535
- Total Exposure -USD 460,296,150
- Average Exposure - USD 10,821.58
- The criteria for a retail portfolio are satisfied as there is a large number of observations, and the exposure per obligor is less than USD 1 million.

Analysis:

These statistics indicate that the dataset aligns with the characteristics of a retail portfolio as defined under BASEL criteria. It consists of a substantial number of observations with exposures per obligor falling below USD 1 million, which is typical for a retail portfolio.

2. **Retail Sub-Portfolio:** A fixed amount of credit has been lent out to borrowers for 36 or 60 months, making them term loans. However, these loans are not secured, classifying them as personal loans within the 'Other-Retail' product category.

   Analysis: The loans in the dataset have specific terms (36 or 60 months), indicating they are term loans. Since these loans are not secured, they fit the definition of personal loans, which falls under the 'Other-Retail' product category as per BASEL criteria.

3. **Data Requirements:** The dataset covers the period from June 2007 to September 2016, encompassing the downturn period of 2008-2010. This provides a 10-year data history that includes a significant economic downturn.

   Analysis: The dataset meets the first data requirement for BASEL, which is to have at least 5 years of data, preferably including a downturn period. In this case, it covers a 10-year period with a downturn included.

4. **Data Documentation:** The dataset Loan_Stats is confirmed to have a documented data dictionary.
   Analysis: Having a documented data dictionary is a fundamental requirement for BASEL compliance. It ensures that variables are well-defined and documented, enabling unambiguous usage among different stakeholders.

Analysis of Key Observations on the AIRB Feeder Models

provides key observations regarding the Probability of Default (PD) and Exposure at Default (EAD) in the context of AIRB (Advanced Internal Ratings-Based) feeder models, which are important for risk modeling and capital calculation under Basel regulations. Here's an analysis based on the information provided:

1. **Probability of Default (PD):**
   Definition: The variable 'Y' takes a value of 1 if the borrower defaults in the next 12 months and 0 otherwise.
   Observations:
   Delinquency Variables: Delinquency variables or their variations play a crucial role in modeling the probability of default. These variables are not only used to create the default tag but also serve as critical explanatory factors for developing PD models.
   Behavior on Revolving Products: Analyzing the behavior of borrowers who have access to revolving credit products is essential. It provides insights into their creditworthiness and repayment capabilities.
   For instance, if a borrower has an overdraft (OD) facility and takes a loan, the bank needs to be vigilant. In times of financial stress, the borrower can use the OD to repay the loan while becoming delinquent on the OD.

Analysis: The PD modeling process is highly dependent on delinquency variables, which are indicative of a borrower's credit behavior and potential default risk.

Understanding borrower behavior on revolving credit products is critical for assessing their creditworthiness. Borrowers with access to such products may exhibit different repayment patterns during financial stress.

2. **Exposure at Default (EAD):**

   Definition: EAD represents the most conservative estimate of the bank's exposure if an account were to go into default in the next 12 months. It is calculated as the net outstanding balance plus any other outstanding fees, charges, penalties, etc.

   Observations:

   Credit Conversion Factor (CCF): For revolving products, the Credit Conversion Factor (CCF) becomes a vital factor in calculating EAD. EAD for revolving products is determined by adding the outstanding balance on the revolving product to the product of CCF and headroom (the difference between the total limit and balance). In contrast, for non-revolving products, there is no role for the CCF in EAD calculation.

   Analysis: EAD provides a conservative estimate of the bank's potential exposure in the event of a default. It includes not only the outstanding balance but also additional fees and charges.

   The use of CCF is emphasized for revolving products, where it helps account for potential credit usage in times of default risk. Non-revolving products do not require CCF consideration in EAD calculations.

IDENTIFICATION OF DATA SOURCE AND DATA COLLECTION

1. **mths_since_last_record:**
   - **Variable Type**: Application Variable
   - **Variable Source:** Bureau Variable
   - **Comments:** This variable has 100% missing observations.
   - **Analysis:** It is primarily sourced from credit bureau data and records the number of months since the last public record was obtained. However, it appears that there is no data available for this variable in the dataset.

2. **mths_since_last_major_derog:**
    - **Variable Type:** Date Variable
    - **Variable Source:** Bureau Variable
    - **Comments:** Useful for identifying bad flags. Effectively used in PD and (or) LGD model.
    - **Analysis:** Similar to the previous variable, this variable also has 100% missing observations. It tracks the months since the last occurrence of 90+ days past due or worse rating, making it relevant for assessing borrower creditworthiness.

3. **annual_inc_joint:**
    - **Variable Type:** Application Variable
    - **Variable Source:** Bank Internal Variable
    - **Comments:** This variable has 100% missing observations.
    - **Analysis:** It represents the total income of the applicant and the co-applicant. Despite its high importance in assessing the joint income of applicants, there is no available data for this variable.

4. **dti_joint:**
    - **Variable Type:** Application Variable
    - **Variable Source:** Bank Internal Variable
    - **Comments:** This variable has 100% missing observations.
    - **Analysis:** It records the total debts paid by co-applicants on other debt obligations relative to their reported income. Similar to 'annual_inc_joint,' there is no data available for this variable.

5. **verification_status_joint:**
    - **Variable Type:** Operational Variable
    - **Variable Source:** Bank Internal Variable
    - **Comments:** This variable has 100% missing observations.
    - **Analysis:** While it is an operational variable related to verification status for joint applicants, there is no data recorded for this variable in the dataset.

6. **tot_coll_amt:**
    - **Variable Type:** LGD Model Variable
    - **Variable Source:** Bank Internal Variable

- **Comments:** This variable has 100% missing observations.
- **Analysis:** It represents the total collection amount outstanding. Despite its relevance to LGD modeling, there is no data available for this variable.

7. **tot_cur_bal:**
    - **Variable Type**: PD Model Variable
    - **Variable Source:** Bureau Variable
    - **Comments:** This variable has 100% missing observations.
    - **Analysis**: It accounts for the total current balances from all accounts, which is important for PD modeling. However, there is no data available for this variable.

8. **open_acc_6m:**
    - **Variable Type:** PD Model Variable
    - **Variable Source:** Bank Internal Variable
    - **Comments:** It is derived from the variable "number of open accounts." Need to check which variable is more appropriate.
    - **Analysis:** This variable records the number of open accounts in the last 6 months and is important for assessing recent credit behavior. However, there is no available data to analyze in the dataset.

9. **open_il_6m:**
    - **Variable Type:** PD Model Variable
    - **Variable Source:** Bank Internal Variable
    - **Comments:** Need to check which installment variable to be used.
    - **Analysis:** This variable represents the number of currently active installments within the last 6 months. However, it is also missing entirely from the dataset.

10. **open_il_12m:**
    - **Variable Type:** PD Model Variable
    - **Variable Source:** Bank Internal Variable
    - **Comments:** Need to check which installment variable to be used.
    - **Analysis:** Similar to the previous variable, this one records the number of installment accounts opened in the past 12 months. Unfortunately, it is entirely missing in the dataset.

11. open_il_24m:

- **Variable Type:** PD Model Variable
- **Variable Source:** Bank Internal Variable
- **Comments:** Need to check which installment variable to be used.
- **Analysis:** This variable tracks the number of installment accounts opened in the past 24 months. However, like the others, it also has 100% missing observations.

12. mths_since_rcnt_il:

- **Variable Type:** PD Model Variable
- **Variable Source:** Bank Internal Variable
- **Comments:** Need to check which installment variable to be used.
- **Analysis:** This variable records the month when the most recent installment account was opened. Despite its potential relevance, there is no data available for analysis.

13. total_bal_il:

- **Variable Type:** PD Model Variable
- **Variable Source:** Bureau Variable
- **Comments:** This variable has 100% missing observations.
- **Analysis:** It represents the total current balances from all accounts related to installment loans.

14. il_util:

- **Variable Type:** PD Model Variable
- **Variable Source:** Bureau Variable
- **Comments:** This variable has 100% missing observations.
- **Analysis:** It calculates the total current balance from all accounts relative to the credit limit on all installment accounts, providing insight into credit utilization.

15. open_rv_12m:

- **Variable Type:** PD Model Variable
- **Variable Source:** Bureau Variable
- **Comments**: This variable has 100% missing observations.
- **Analysis:** It records the number of times credit has been revolved in the last 12 months.

16. open_rv_24m:

- **Variable Type:** PD Model Variable
- **Variable Source:** Bureau Variable
- **Comments:** This variable has 100% missing observations.
- **Analysis:** Similar to the previous variable, it tracks the number of times credit has been revolved, but it is entirely missing from the dataset.

17. max_bal_bc:

- **Variable Type:** EAD Model Variable
- **Variable Source:** Bureau Variable
- **Comments:** This variable has 100% missing observations.
- **Analysis:** It calculates the maximum current balance among all revolving accounts, which is essential for EAD modeling.

18. all_util:

- **Variable Type:** PD and (or) EAD Model Variable
- **Variable Source:** Bureau Variable
- **Comments:** This variable has 100% missing observations.
- **Analysis:** It computes the total balance relative to the total limit for all products, offering insights into overall credit utilization.

19. total_rev_hi_lim:

- **Comments**: This variable has 100% missing observations.
- **Analysis:** There is no description provided for this variable, and it is missing entirely from the dataset.

20. inq_fi:

- **Variable Type:** Application Variable
- **Variable Source:** Bureau Variable
- **Comments:** The number of inquiries made by an applicant is analyzed, usually at the acquisition stage.
- **Analysis:** It records the number of personal finance inquiries and is relevant for assessing borrower behavior. However, no data is available for analysis.

The analysis shows that many critical variables necessary for modeling and assessing credit risk are missing entirely from the dataset. This could significantly impact the ability to build accurate predictive models and risk assessments. To proceed, efforts should be made to collect or obtain data for these missing variables to ensure robust modeling and analysis.

MISSING OBSERVATION ANALYSIS

The table you've provided appears to show the percentage of missing values for various variables across different issues or time periods. Here's an interpretation of the information:

Variables and Missing Value Percentage:

- Each row in the table represents a different variable or feature.
- The columns labeled "Issue_Date" seem to indicate different time periods or data issues.
- The columns with percentages represent the proportion of missing values for each variable during those specific periods or issues.

"Keep/Drop" Column:

- The "Keep" designation in the "Keep/Drop" column suggests that all variables should be retained for analysis, as indicated by "100%" in all columns.
- This means that there are no variables with missing values exceeding the threshold for dropping.

Variable Type:

- The "Variable Type" column indicates that all the listed variables are categorical in nature.

Analysis Type:

- The "Analysis Type" column specifies that a "Frequency Distribution Analysis" will be conducted on these categorical variables.
- This analysis likely involves examining the distribution of categories within each categorical variable to understand their frequencies and proportions.

Interpretation:

- Based on the provided table, it appears that there are no missing values (all "100%" or complete data) for any of the listed categorical variables across different issues or time periods.
- This suggests that data completeness is not an issue for these variables and that they can be used in subsequent analyses without any major concerns related to missing data.

The table indicates that the listed categorical variables have complete data with no missing values across different issues or time periods, making them suitable for further analysis.

UNIVARIATE ANALYSIS

Univariate analysis in credit risk assessment is a statistical method used to analyze individual variables or factors that may affect a borrower's creditworthiness. In this context, credit risk assessment refers to the process of evaluating the likelihood that a borrower will default on their loan or credit obligation. Univariate analysis focuses on examining one variable at a time to understand its impact on credit risk.

In a symmetrically distributed variable, data points tend to be evenly distributed around the mean, creating a balanced and smooth distribution curve.

Symmetric Behavior:

- Symmetric behavior refers to a variable's data distribution without abnormal fluctuations or extreme deviations.
- Abnormal fluctuations are deviations from the typical pattern that result in unnatural spikes or dips in the data trend.

X = Mean +- 1*std.dev:

- If a variable follows a symmetric distribution, approximately 66.67% of the observations of that variable will lie within this interval.
- This interval represents one standard deviation away from the mean in both directions.

X = Mean +- 2*std.dev:

- For a symmetrically distributed variable, around 95.5% of the observations will fall within this interval.
- This interval represents two standard deviations away from the mean in both directions.

X= Mean+-3*Std.dev:

In a symmetric distribution, approximately 99.97% of the observations will be within this interval.
This interval signifies three standard deviations away from the mean in both directions.

A key step in evaluating a variable's univariate summary is to look for symmetric behaviour. Deviations outside of these specified intervals may be an indication of aberrant fluctuations or outliers in the data since symmetrically distributed variables typically follow a normal distribution pattern. By doing so, we can discover any odd trends or extreme data points and gain a better understanding of the variables' regular behaviour.

Z SCORE ANALYSIS:

Z = ABS(X - Mean)/Std_Dev	Scales
Z < 1	The value lies within +- 1 Standard Deviation
1 < Z < 2	The value lies within +- 1 and +- 2 Standard Deviation
2 < = Z < 3	The value lies within +-2 and +-3 Standard Deviation
Z > = 3	The value lies outside +-3 Standard Deeviation

The Z-score is a statistical measure used to standardize data points and assess how far they deviate from the mean in terms of standard deviations.

Analysis of the above table:

1. **Z < 1:**

- When the Z-score is less than 1, it indicates that the data point is within ±1 standard deviation from the mean.
- This means that approximately 68% of the data falls within this range for a normally distributed variable.
- Data points in this range are considered to be close to the mean and are not considered outliers.

2. **1 < Z < 2:**

- If the Z-score falls between 1 and 2, it implies that the data point is between ±1 and ±2 standard deviations from the mean.
- Roughly 27% of the data is expected to lie within this range in a normal distribution.
- While these data points are somewhat further from the mean, they are still relatively common and not considered extreme outliers.

3. **2 <= Z < 3:**

- When the Z-score is between 2 and 3, it means the data point falls within ±2 and ±3 standard deviations from the mean.
- Only about 4% of the data is expected to fall within this range for a normally distributed variable.
- Data points in this range are less common and may be considered moderately distant from the mean.

4. **Z >= 3:**

- If the Z-score is greater than or equal to 3, it suggests that the data point lies outside ±3 standard deviations from the mean.
- Less than 0.3% of the data is expected to be in this range in a normal distribution.
- Data points with Z-scores in this range are typically considered outliers and may be of particular interest for further investigation.

ANALYSIS OF VARIABLE NAME AND DESCRIPTIVE MEASURES:

These statistics are meant to help the risk analytics team identify key factors contributing to an increase in default rates and improve acquisition risk management.

1. **chargeoff_within_12_mths:**

- This variable appears to represent the number of accounts with charge-offs within 12 months.
- However, all the descriptive statistics (mean, standard deviation, min, max, percentiles) are zero for all months (201106 to 201112). This suggests that there might be a data issue or that charge-offs are extremely rare or absent during these months.

2. **collection_recovery_fee:**

- This variable seems to represent the collection recovery fees for accounts.
- Similar to the previous variable, all descriptive statistics are zero for all months, indicating an issue with the data or the absence of collection recovery fees.

3. **collections_12_mths_ex_med:**

- This variable appears to represent the number of collections within 12 months, excluding medical collections.
- As with the previous two variables, all descriptive statistics are zero for all months, suggesting a potential data issue or a lack of collections.

4. **delinq_amnt:**

- This variable seems to represent the delinquent amount for accounts.
- Once again, all descriptive statistics are zero for all months, which may indicate an issue with the data or that delinquent amounts are negligible during these months.

5. installment:

- This variable represents the installment amounts for accounts.
- Descriptive statistics show variability across months, including mean, standard deviation, min, max, and percentiles. This suggests that installment amounts vary over time.

6. int_rate:

- This variable represents the interest rates for accounts.
- Descriptive statistics show variability in interest rates across months, with different means, standard deviations, and percentiles.

7. last_pymnt_amnt:

- This variable appears to represent the last payment amounts for accounts.
- Descriptive statistics indicate variation in last payment amounts across months.

8. Mon_on_books:

- This variable represents the months on the books for accounts.
- The mean and standard deviation vary slightly across months, indicating some changes in the average months on the books.

9. out_prncp and out_prncp_inv:

- These variables represent outstanding principal amounts and outstanding principal amounts (investor).
- Descriptive statistics show variability across months, indicating changes in outstanding principal amounts.

10. recoveries:

- This variable represents recovery amounts.
- Similar to earlier variables, all descriptive statistics are zero for all months, suggesting a data issue or a lack of recovery amounts.

11. revol_bal and revol_util:

- These variables represent revolving balance and revolving credit utilization, respectively.
- Descriptive statistics indicate variability in revolving balances and credit utilization across months

ANALYSIS OF FREQUENCY SUMMARY RESULT:

The frequency summary result offers a picture of several categorical variables over various months. Here is the detailed analysis of it:

1. **Grade and Subgrade Analysis:**
    - Grades A, B, C, D, E, F, and G represent different risk levels or quality of loans.
    - Grade A has a mean of 7.00, indicating a relatively low frequency.
    - Grade B has the highest mean at 31.40, suggesting it's the most common grade.
    - The frequency generally decreases as you move from grade B to grade G.
    - Subgrades show a similar pattern as grades.

2. **Initial List Status and Payment Plan Analysis:**
    - These variables have binary values (Yes/No).
    - Both have a similar mean of 78.00, indicating a roughly equal distribution between the two categories.
    - The standard deviation is high for both, suggesting some variation in these variables across months.

3. **Pymnt Plan Analysis:**
    - This variable also has a binary value (Yes/No).
    - Similar to initial list status, it has a mean of 78.00 and relatively high standard deviation, indicating variation across months.

Overall Analysis of frequency summary result:

- Grades B and C seem to be the most common loan grades, while grades A and G are less common.
- Initial list status and payment plan appear to have relatively even distributions.
- Subgrades follow a similar pattern to grades.

pgp0007-anshika-panwar-rfp-draft-3

September 30, 2023

[93]:
```
# RFP draft 3
```

[94]:
```
# Name- Anshika Panwar
# Registration no.- 2022SEPVPGP0007
```

[37]:
```
import pandas as pd
```

[38]:
```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix
```

[39]:
```
# Read the CSV file into a DataFrame
data = pd.read_csv(r"D:\MBA\TERM-4 & 5 (CARRER TRACK)\RFP\master data.csv")
```

[40]:
```
data.head()
```

[40]:
```
        id  member_id  loan_amnt  funded_amnt  funded_amnt_inv        term  \
0  1077501    1296599       5000         5000           4975.0   36 months
1  1077430    1314167       2500         2500           2500.0   60 months
2  1077175    1313524       2400         2400           2400.0   36 months
3  1076863    1277178      10000        10000          10000.0   36 months
4  1075358    1311748       3000         3000           3000.0   60 months

   installment  annual_inc home_ownership  loan_status  …  total_pymnt_inv  \
0       162.87     24000.0           RENT   Fully Paid  …          5833.84
1        59.83     30000.0           RENT  Charged Off  …          1008.71
2        84.33     12252.0           RENT   Fully Paid  …          3005.67
3       339.31     49200.0           RENT   Fully Paid  …         12231.89
4        67.79     80000.0           RENT      Current  …          3784.49

   total_pymnt.1  total_pymnt_inv.1  total_rec_prncp  total_rec_int  \
0    5863.155187            5833.84          5000.00         863.16
1    1008.710000            1008.71           456.46         435.17
2    3005.666844            3005.67          2400.00         605.67
```

```
3   12231.890000           12231.89         10000.00        2214.92
4    3784.490000            3784.49          2729.22        1055.27

   total_rec_late_fee  recoveries  collection_recovery_fee  last_pymnt_d  \
0                0.00        0.00                     0.00        15-Jan
1                0.00      117.08                     1.11        13-Apr
2                0.00        0.00                     0.00        14-Jun
3               16.97        0.00                     0.00        15-Jan
4                0.00        0.00                     0.00        16-Sep

   last_pymnt_amnt
0           171.62
1           119.66
2           649.91
3           357.48
4            67.79

[5 rows x 26 columns]
```

[41]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 42535 entries, 0 to 42534
Data columns (total 26 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   id                       42535 non-null  int64
 1   member_id                42535 non-null  int64
 2   loan_amnt                42535 non-null  int64
 3   funded_amnt              42535 non-null  int64
 4   funded_amnt_inv          42535 non-null  float64
 5   term                     42535 non-null  object
 6   installment              42535 non-null  float64
 7   annual_inc               42531 non-null  float64
 8   home_ownership           42535 non-null  object
 9   loan_status              42535 non-null  object
 10  out_prncp                42535 non-null  float64
 11  out_prncp_inv            42535 non-null  float64
 12  dti                      42535 non-null  float64
 13  delinq_2yrs              42506 non-null  float64
 14  revol_bal                42535 non-null  int64
 15  total_pymnt              42535 non-null  float64
 16  total_pymnt_inv          42535 non-null  float64
 17  total_pymnt.1            42535 non-null  float64
 18  total_pymnt_inv.1        42535 non-null  float64
 19  total_rec_prncp          42535 non-null  float64
 20  total_rec_int            42535 non-null  float64
 21  total_rec_late_fee       42535 non-null  float64
 22  recoveries               42535 non-null  float64
 23  collection_recovery_fee  42535 non-null  float64
 24  last_pymnt_d             42452 non-null  object
 25  last_pymnt_amnt          42535 non-null  float64
dtypes: float64(17), int64(5), object(4)
memory usage: 8.4+ MB
```

[42]:

```
# Now we will Check the missing values
data.isnull().sum()
```

[42]:

```
id                          0
member_id                   0
loan_amnt                   0
funded_amnt                 0
funded_amnt_inv             0
term                        0
installment                 0
annual_inc                  4
home_ownership              0
loan_status                 0
out_prncp                   0
out_prncp_inv               0
dti                         0
delinq_2yrs                29
revol_bal                   0
total_pymnt                 0
total_pymnt_inv             0
total_pymnt.1               0
total_pymnt_inv.1           0
total_rec_prncp             0
total_rec_int               0
total_rec_late_fee          0
recoveries                  0
collection_recovery_fee     0
last_pymnt_d               83
last_pymnt_amnt             0
dtype: int64
```

[43]:

```
# Drop rows with missing values in the 'last_pymnt_d' column
data = data.dropna(subset=['last_pymnt_d'])

# After dropping the rows, you can check the missing values again if needed
missing_values = data.isnull().sum()
print(missing_values)
```

```
id                          0
member_id                   0
```

```
loan_amnt                  0
funded_amnt                0
funded_amnt_inv            0
term                       0
installment                0
annual_inc                 4
home_ownership             0
loan_status                0
out_prncp                  0
out_prncp_inv              0
dti                        0
delinq_2yrs               29
revol_bal                  0
total_pymnt                0
total_pymnt_inv            0
total_pymnt.1              0
total_pymnt_inv.1          0
total_rec_prncp            0
total_rec_int              0
total_rec_late_fee         0
recoveries                 0
collection_recovery_fee    0
last_pymnt_d               0
last_pymnt_amnt            0
dtype: int64
```

[44]:

```
# Define a function to map loan statuses to 0 or 1
def map_loan_status(status):
    if status == 'current':
        return 0
    else:
        return 1

# Apply the function to the 'loan_status' column and create a new column␣
 ↪'status_binary'
data['status_binary'] = data['delinq_2yrs'].apply(map_loan_status)
```

[45]:

```
data.head()
```

[45]:

```
        id  member_id  loan_amnt  funded_amnt  funded_amnt_inv        term  \
0  1077501    1296599       5000         5000           4975.0   36 months
1  1077430    1314167       2500         2500           2500.0   60 months
2  1077175    1313524       2400         2400           2400.0   36 months
3  1076863    1277178      10000        10000          10000.0   36 months
4  1075358    1311748       3000         3000           3000.0   60 months

   installment  annual_inc home_ownership  loan_status  …  total_pymnt.1  \
0       162.87     24000.0           RENT   Fully Paid  …    5863.155187
1        59.83     30000.0           RENT  Charged Off  …    1008.710000
2        84.33     12252.0           RENT   Fully Paid  …    3005.666844
3       339.31     49200.0           RENT   Fully Paid  …   12231.890000
4        67.79     80000.0           RENT      Current  …    3784.490000

   total_pymnt_inv.1  total_rec_prncp  total_rec_int  total_rec_late_fee  \
0            5833.84          5000.00         863.16                0.00
1            1008.71           456.46         435.17                0.00
2            3005.67          2400.00         605.67                0.00
3           12231.89         10000.00        2214.92               16.97
4            3784.49          2729.22        1055.27                0.00

   recoveries  collection_recovery_fee  last_pymnt_d  last_pymnt_amnt  \
0        0.00                     0.00        15-Jan           171.62
1      117.08                     1.11        13-Apr           119.66
2        0.00                     0.00        14-Jun           649.91
3        0.00                     0.00        15-Jan           357.48
4        0.00                     0.00        16-Sep            67.79

   status_binary
0              1
1              1
2              1
3              1
4              1

[5 rows x 27 columns]
```

[46]:

```
# Drop irrelevant columns
data = data.drop(['id', 'member_id', 'last_pymnt_d'], axis=1)

# Convert 'term' column to numerical values (e.g., 36 months to 36)
data['term'] = data['term'].str.extract('(\d+)').astype(int)

# Encode categorical variables using Label Encoding
label_encoders = {}
categorical_columns = ['home_ownership', 'loan_status']
for col in categorical_columns:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col])
    label_encoders[col] = le
```

[47]:

```
missing_values = data.isnull().sum()
print(missing_values)
# Example: Impute missing values with the mean
data['delinq_2yrs'].fillna(data['delinq_2yrs'].mean(), inplace=True)
```

```
data.dropna(axis=0, inplace=True)
data['delinq_2yrs'].fillna('missing', inplace=True)
```

```
loan_amnt                   0
funded_amnt                 0
funded_amnt_inv             0
term                        0
installment                 0
annual_inc                  4
home_ownership              0
loan_status                 0
out_prncp                   0
out_prncp_inv               0
dti                         0
delinq_2yrs                29
revol_bal                   0
total_pymnt                 0
total_pymnt_inv             0
total_pymnt.1               0
total_pymnt_inv.1           0
total_rec_prncp             0
total_rec_int               0
total_rec_late_fee          0
recoveries                  0
collection_recovery_fee     0
last_pymnt_amnt             0
status_binary               0
dtype: int64
```

[48]:

```
# Fill missing values in numerical columns with the median
numerical_columns = ['loan_amnt', 'funded_amnt', 'funded_amnt_inv',␣
 ↪'installment', 'annual_inc', 'out_prncp', 'out_prncp_inv', 'dti',␣
 ↪'delinq_2yrs', 'revol_bal', 'total_pymnt', 'total_pymnt_inv',␣
 ↪'total_rec_prncp', 'total_rec_int', 'total_rec_late_fee', 'recoveries',␣
 ↪'collection_recovery_fee', 'last_pymnt_amnt']
data[numerical_columns] = data[numerical_columns].
 ↪fillna(data[numerical_columns].median())

# Fill missing values in categorical columns with the mode
categorical_columns = ['home_ownership',]
data[categorical_columns] = data[categorical_columns].
 ↪fillna(data[categorical_columns].mode().iloc[0])

# Ensure there are no more missing values
missing_values = data.isnull().sum()
print("Missing Values:\n", missing_values)
```

```
Missing Values:
 loan_amnt                  0
funded_amnt                0
funded_amnt_inv            0
term                       0
installment                0
annual_inc                 0
home_ownership             0
loan_status                0
out_prncp                  0
out_prncp_inv              0
dti                        0
delinq_2yrs                0
revol_bal                  0
total_pymnt                0
total_pymnt_inv            0
total_pymnt.1              0
total_pymnt_inv.1          0
total_rec_prncp            0
total_rec_int              0
total_rec_late_fee         0
recoveries                 0
collection_recovery_fee    0
last_pymnt_amnt            0
status_binary              0
dtype: int64
```

[49]:

```
# Encode categorical variables using Label Encoding
label_encoders = {}
for col in categorical_columns:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col])
    label_encoders[col] = le
```

[50]:

```
# Run Random forest classifier
# Split data into features and target variable
X = data.drop('loan_status', axis=1)
y = data['loan_status']
```

[51]:

```
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
  ↪random_state=42)
```

[52]:

```
# Standardize numerical features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[53]:

```python
# Initialize and train a Random Forest Classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)
```

[53]:

```
RandomForestClassifier(random_state=42)
```

[54]:

```python
# Predict loan status on the test set
y_pred = clf.predict(X_test)
```

[55]:

```python
# Calculate accuracy and other classification metrics
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", classification_rep)
```

```
Accuracy: 0.9365135453474676
Confusion Matrix:
 [[1064    0   21    2   24    0    0    0]
 [   0  109    0    0    1    0    0    1]
 [ 141    0   30    2    2    0    0    0]
 [   0    0    0   66  314    0    0    0]
 [   1    0    0   25 6682    0    0    0]
 [   0    3    0    0    0    0    0    0]
 [   0    1    0    0    0    0    0    0]
 [   0    1    0    0    0    0    0    0]]
Classification Report:
               precision    recall  f1-score   support

           0       0.88      0.96      0.92      1111
           1       0.96      0.98      0.97       111
           3       0.59      0.17      0.27       175
           4       0.69      0.17      0.28       380
           5       0.95      1.00      0.97      6708
           6       0.00      0.00      0.00         3
           7       0.00      0.00      0.00         1
           8       0.00      0.00      0.00         1

    accuracy                           0.94      8490
   macro avg       0.51      0.41      0.43      8490
weighted avg       0.92      0.94      0.92      8490


C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
```

```
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[56]:

```
# Create a new loan application with the same number of features
new_loan_application = np.array([5000, 5000, 4975, 36, 162.87, 24000, 0, 27.65,␣
  ↪0, 13648, 5863.155187, 5833.84, 5863.155187, 5833.84, 5000, 863.16, 0, 0, 0,␣
  ↪171.62, 0, 0])
new_loan_application = np.append(new_loan_application, 0)

# Standardize the new data
new_loan_application = scaler.transform([new_loan_application])

# Calculate probability of default for the new loan application
probability_of_default = clf.predict_proba(new_loan_application)[0][1]

print("Probability of Default:", probability_of_default)
```

```
Probability of Default: 0.09
```

```
C:\Users\user\anaconda3\Lib\site-packages\sklearn\base.py:439: UserWarning: X
does not have valid feature names, but StandardScaler was fitted with feature
names
  warnings.warn(
```

[57]:

```
# For logistic regression
data.head()
```

[57]:

```
   loan_amnt  funded_amnt  funded_amnt_inv  term  installment  annual_inc  \
0       5000         5000           4975.0    36       162.87     24000.0
1       2500         2500           2500.0    60        59.83     30000.0
2       2400         2400           2400.0    36        84.33     12252.0
3      10000        10000          10000.0    36       339.31     49200.0
4       3000         3000           3000.0    60        67.79     80000.0

   home_ownership  loan_status  out_prncp  out_prncp_inv  …  \
0               4            5       0.00           0.00  …
1               4            0       0.00           0.00  …
2               4            5       0.00           0.00  …
```

```
3               4            5    0.00           0.00  …
4               4            1  270.78         270.78  …

   total_pymnt_inv  total_pymnt.1  total_pymnt_inv.1  total_rec_prncp  \
0          5833.84    5863.155187            5833.84          5000.00
1          1008.71    1008.710000            1008.71           456.46
2          3005.67    3005.666844            3005.67          2400.00
3         12231.89   12231.890000           12231.89         10000.00
4          3784.49    3784.490000            3784.49          2729.22

   total_rec_int  total_rec_late_fee  recoveries  collection_recovery_fee  \
0         863.16                0.00        0.00                     0.00
1         435.17                0.00      117.08                     1.11
2         605.67                0.00        0.00                     0.00
3        2214.92               16.97        0.00                     0.00
4        1055.27                0.00        0.00                     0.00

   last_pymnt_amnt  status_binary
0           171.62              1
1           119.66              1
2           649.91              1
3           357.48              1
4            67.79              1

[5 rows x 24 columns]
```

[58]:

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
```

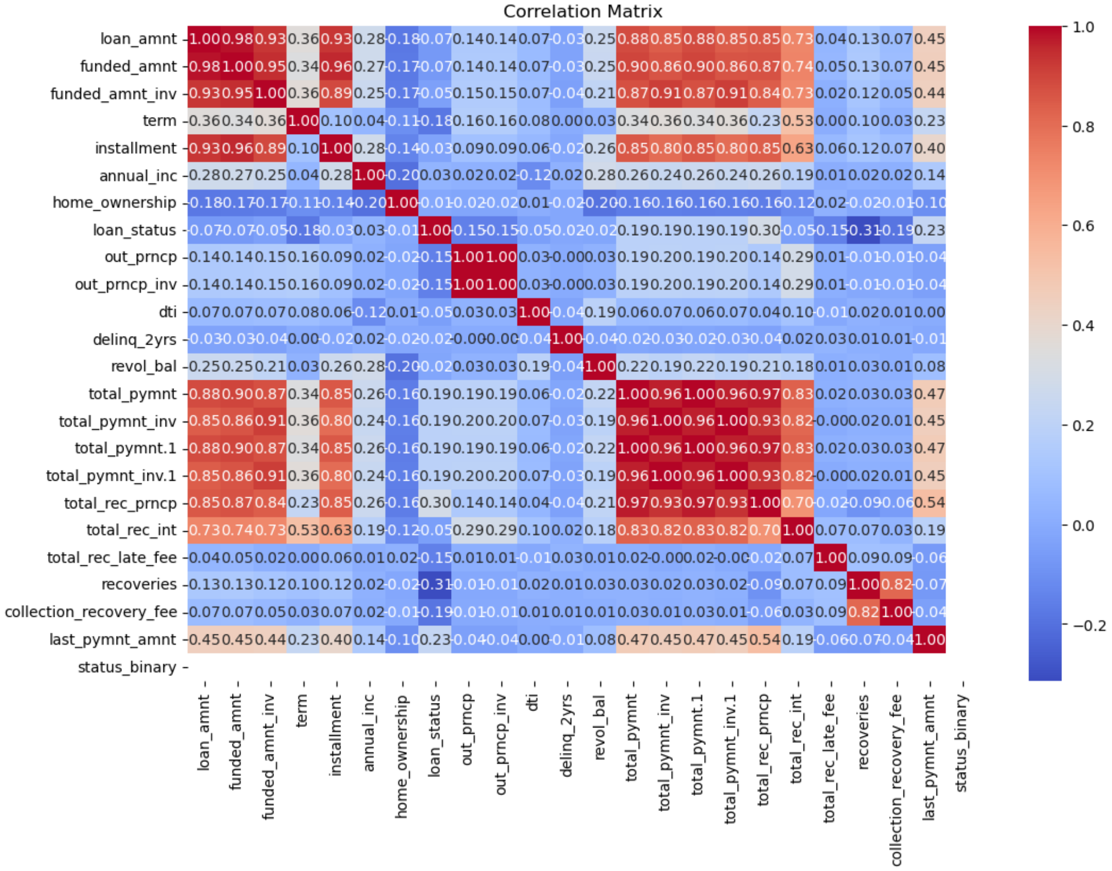
[59]:

```
# We will find out the correlation between variables and plot it

# Calculate the correlation matrix
correlation_matrix = data.corr()
```

[60]:

```
# Create a heatmap to visualize the correlation matrix
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Matrix")
plt.show()
```



[61]:
```
from sklearn.linear_model import LogisticRegression
```

[66]:
```
# Split data into features and target variable
X = data.drop('loan_status', axis=1)
y = data['loan_status']
```

[67]:
```
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[69]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix
```

[70]:
```
# Create a logistic regression model
model = LogisticRegression()
```

[71]:
```
# Train the model on the training data
model.fit(X_train, y_train)
```

```
C:\Users\user\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:458:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

```
[71]: LogisticRegression()
```

```
[72]: # Make predictions on the testing data
      y_pred = clf.predict(X_test)

      # Calculate accuracy
      accuracy = accuracy_score(y_test, y_pred)

      # Print a classification report
      class_report = classification_report(y_test, y_pred)

      # Print a Confusion matrix
      conf_matrix = confusion_matrix(y_test, y_pred)

      print(f"Accuracy: {accuracy}")
      print(f"Confusion Matrix:\n{conf_matrix}")
      print(f"Classification Report:\n{class_report}")
```

```
Accuracy: 0.9365135453474676
Confusion Matrix:
[[1064    0   21    2   24    0    0    0]
 [   0  109    0    0    1    0    0    1]
 [ 141    0   30    2    2    0    0    0]
 [   0    0    0   66  314    0    0    0]
 [   1    0    0   25 6682    0    0    0]
 [   0    3    0    0    0    0    0    0]
 [   0    1    0    0    0    0    0    0]
 [   0    1    0    0    0    0    0    0]]
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.96      0.92      1111
           1       0.96      0.98      0.97       111
           3       0.59      0.17      0.27       175
           4       0.69      0.17      0.28       380
           5       0.95      1.00      0.97      6708
           6       0.00      0.00      0.00         3
           7       0.00      0.00      0.00         1
           8       0.00      0.00      0.00         1

    accuracy                           0.94      8490
   macro avg       0.51      0.41      0.43      8490
weighted avg       0.92      0.94      0.92      8490
```

```
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[74]:

```python
# Get the probability of default for each sample in the testing data

y_prob = clf.predict_proba(X_test)
probability_of_default = y_prob[:, 1]
print("Probability of Default:", probability_of_default)
```

```
Probability of Default: [0. 0. 0. … 0. 0. 0.]
```

[75]:

```python
# Now we will run the decision tree model
data.head()
```

[75]:

```
   loan_amnt  funded_amnt  funded_amnt_inv  term  installment  annual_inc  \
0       5000         5000           4975.0    36       162.87     24000.0
1       2500         2500           2500.0    60        59.83     30000.0
2       2400         2400           2400.0    36        84.33     12252.0
3      10000        10000          10000.0    36       339.31     49200.0
4       3000         3000           3000.0    60        67.79     80000.0

   home_ownership  loan_status  out_prncp  out_prncp_inv  …  \
0               4            5       0.00           0.00  …
1               4            0       0.00           0.00  …
2               4            5       0.00           0.00  …
3               4            5       0.00           0.00  …
4               4            1     270.78         270.78  …
```

```
   total_pymnt_inv  total_pymnt.1  total_pymnt_inv.1  total_rec_prncp  \
0          5833.84    5863.155187            5833.84          5000.00
1          1008.71    1008.710000            1008.71           456.46
2          3005.67    3005.666844            3005.67          2400.00
3         12231.89   12231.890000           12231.89         10000.00
4          3784.49    3784.490000            3784.49          2729.22

   total_rec_int  total_rec_late_fee  recoveries  collection_recovery_fee  \
0         863.16                0.00        0.00                     0.00
1         435.17                0.00      117.08                     1.11
2         605.67                0.00        0.00                     0.00
3        2214.92               16.97        0.00                     0.00
4        1055.27                0.00        0.00                     0.00

   last_pymnt_amnt  status_binary
0           171.62              1
1           119.66              1
2           649.91              1
3           357.48              1
4            67.79              1

[5 rows x 24 columns]
```

```
[76]: import pandas as pd
      from sklearn.tree import DecisionTreeClassifier
      from sklearn.model_selection import train_test_split
      from sklearn.metrics import accuracy_score, classification_report,␣
       ↪confusion_matrix
```

```
[77]: # Split data into features and target variable
      X = data.drop('loan_status', axis=1)
      y = data['loan_status']

      # Split data into training and testing sets
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
       ↪random_state=42)
```

```
[78]: # Create the decision tree classifier
      Decision_Tree = DecisionTreeClassifier(random_state=42)
```

```
[79]: # Train the classifier on the training data
      Decision_Tree.fit(X_train, y_train)
```

```
[79]: DecisionTreeClassifier(random_state=42)
```

[83]:

```python
# Make predictions on the testing data
y_pred = Decision_Tree.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)

# Print a classification report
class_report = classification_report(y_test, y_pred)

# Print a Confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print(f"Confusion Matrix:\n{conf_matrix}")
print(f"Classification Report:\n{class_report}")
```

```
Accuracy: 0.8958775029446407
Confusion Matrix:
[[ 979    0    0  106    2   24    0    0    0]
 [   0  100    2    0    0    1    5    1    2]
 [   0    0    0    0    0    0    0    0    0]
 [ 115    0    0   55    4    1    0    0    0]
 [   1    0    0    3  105  271    0    0    0]
 [  15    0    0    5  321 6367    0    0    0]
 [   0    2    0    0    0    0    0    0    1]
 [   0    1    0    0    0    0    0    0    0]
 [   0    1    0    0    0    0    0    0    0]]
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.88      0.88      1111
           1       0.96      0.90      0.93       111
           2       0.00      0.00      0.00         0
           3       0.33      0.31      0.32       175
           4       0.24      0.28      0.26       380
           5       0.96      0.95      0.95      6708
           6       0.00      0.00      0.00         3
           7       0.00      0.00      0.00         1
           8       0.00      0.00      0.00         1

    accuracy                           0.90      8490
   macro avg       0.37      0.37      0.37      8490
weighted avg       0.90      0.90      0.90      8490


C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
```

```
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\user\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1344: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[ ]:

```

```

RESULTS AND FINDINGS

I have solved it in Python and here is the finding from it.

I have run three models: Random Forest, Logistic Regression, and Decision tree for the given loan data set.

SOLUTIONS AND INTERPRETATIONS

RANDOM FOREST

STEP:1: FIND THE CONTINUOUS VARIABLES FROM THE GIVEN DATA

1. First, import the data CSV file into Python.
2. Check, the missing values using data.isnull().sum()
3. There were missing values in annual_inc, delinq_2yrs, and last_pymnt_d variables.
4. We will drop the column last_pymnt_d and fill the other two variables by mean.
5. Now we will map_loan_status by binary numbers.
6. And convert home_ownership and loan_status from string to numeric using label_encoders.
7. Now, the data is ready to use for model.

INTERPRETATION OF RANDOM FOREST OUTPUT:

1. **Accuracy:** The accuracy of the model is approximately 93.65%, which is a measure of how many of the total predictions made by the model are correct. In other words, the model correctly classifies 93.65% of the total instances in your test dataset.

2. **Confusion Matrix:**
   - The confusion matrix is a table that shows how many true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions the model made for each class.
   - The confusion matrix is not in a square shape, indicating that your model is likely performing multiclass classification with more than two classes. Each row and column of the confusion matrix corresponds to a specific class.
   - For example, the element in row 1, column 1 (1064) represents the true positives for class 0, and the element in row 2, column 2 (109) represents the true positives for class 1.

3. **Classification Report:**

   The classification report provides several evaluation metrics for each class, as well as overall metrics. For each class (0, 1, 3, 4, 5, 6, 7, 8), we have:

   i) **Precision:** This measures the accuracy of positive predictions. For example, for class 0, the precision is 0.88, which means that 88% of the instances predicted as class 0 were correct.

   ii) **Recall:** This measures the model's ability to identify all relevant instances of a class. For example, for class 0, the recall is 0.96, indicating that 96% of the actual class 0 instances were correctly identified.

   iii) **F1-score:** The F1-score is the harmonic mean of precision and recall. It provides a balanced measure of a model's performance. For class 0, the F1-score is 0.92.

   iv) **Support:** The number of instances in each class in your test dataset.

4. **Accuracy, Macro Avg, Weighted Avg:** These are summary statistics:

   i) **Accuracy:** Already discussed, it's the overall accuracy of the model.

   ii) **Macro Avg:** The macro-average is the average of the metrics (precision, recall, F1-score) calculated for each class independently. It gives equal weight to each class, regardless of its size.

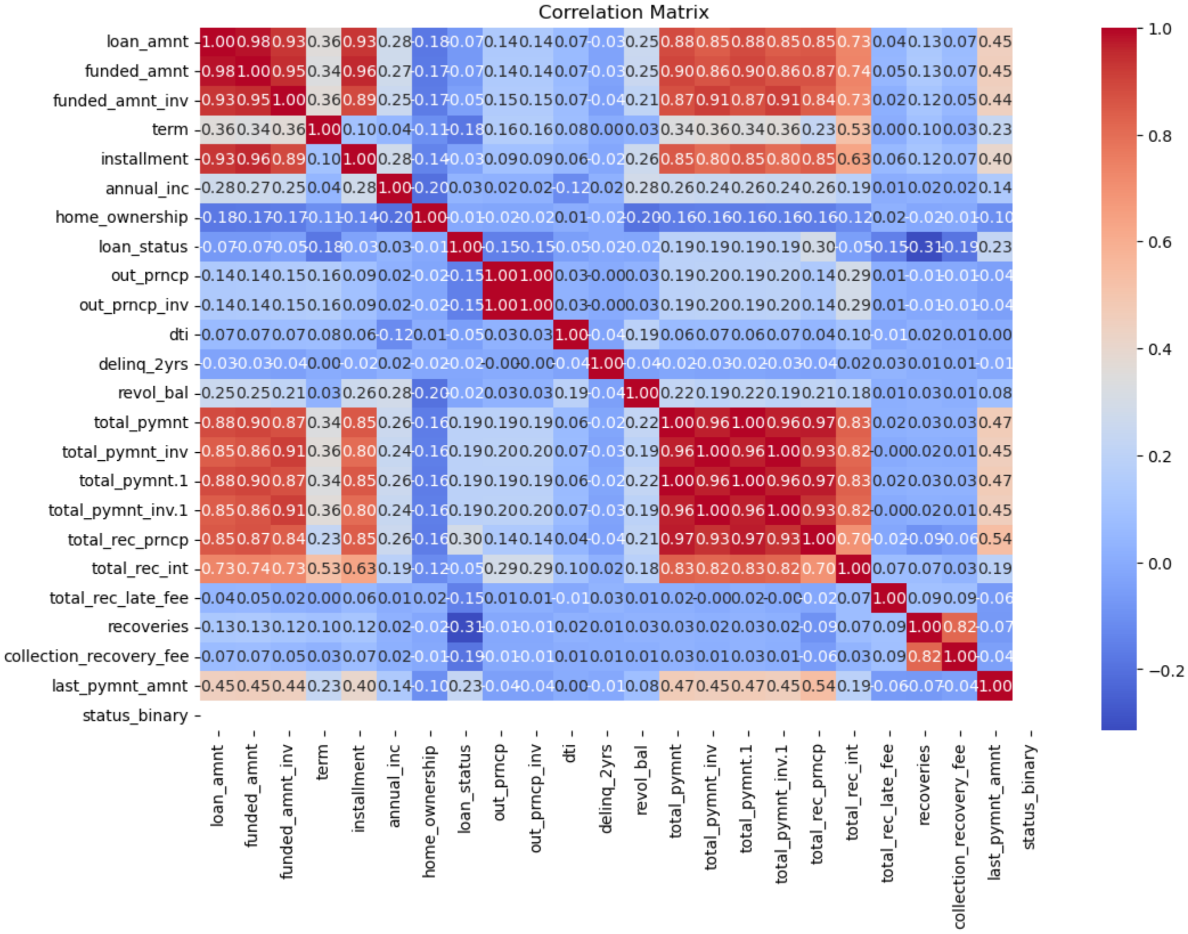
   iii) **Weighted Avg:** The weighted average is the same as the macro-average, but it takes into account the number of instances in each class. This means that classes with more instances have a greater impact on the average.

5. The Probability of Default is 0.09

In summary, the model seems to have high accuracy, but the classification report highlights that the performance varies significantly across different classes. For some classes, like class 5, the model performs very well, with high precision, recall, and F1-score. However, for other classes, like class 3, the model's performance is relatively poor, with lower precision, recall, and F1-score. This suggests that the model might struggle to correctly classify instances in certain classes and may benefit from further tuning or additional data.

LOGISTIC REGRESSION

CORRELATION MATRIX:



INTERPRETATION OF RANDOM FOREST OUTPUT:

1. **Accuracy:** Accuracy is a measure of how many of the total predictions made by the model were correct. In this case, the model's accuracy is approximately 91.91%, which suggests that it correctly predicted the class labels for about 91.91% of the total instances.

2. **Confusion Matrix:**
   - The confusion matrix is a table that shows how many true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) predictions the model made for each class.
   - Each cell in the matrix represents the number of instances that fall into the intersection of a predicted class and an actual class. For example, the cell at (0,0) indicates that 1042 instances were correctly classified as class 0, while the cell at (1,1) indicates that 49 instances were correctly classified as class 1.

3. **Classification Report:**

The classification report provides several evaluation metrics for each class, as well as overall metrics.

- **Precision:** Precision measures the accuracy of positive predictions. A high precision means that when the model predicts a positive class, it is likely to be correct. For example, class 5 has a high precision of 0.94, indicating that the model's positive predictions for class 5 are usually correct.
- **Recall:** Recall (or sensitivity) measures the ability of the model to correctly identify all instances of a class. A high recall means that the model is good at identifying the true positive instances of a class.
- **F1-score:** The F1-score is the harmonic mean of precision and recall and provides a balance between the two metrics. It is a useful measure when dealing with imbalanced datasets.
- **Support**: Support is the number of instances in each class.

4. **Accuracy, Macro Avg, Weighted Avg:** These are summary statistics:

- **Accuracy:** Overall accuracy of the model.
- **Macro Avg:** The macro-average calculates the average of the metrics for each class, treating all classes equally.
- .**Weighted Avg:** The weighted average calculates the average of the metrics for each class, with each class's contribution weighted by its support.

In summary, the model appears to perform well in classifying class 0 and class 5, as indicated by their high precision, recall, and F1-scores. However, it performs poorly for some other classes, especially class 3 and class 4, which have lower precision, recall, and F1-scores. The weighted average F1-score of 0.90 suggests that the model is reasonably good, but its performance varies across different classes.

DECISION TREE

INTERPRETATION OF RANDOM FOREST OUTPUT:

1. Accuracy: The accuracy of the model is approximately 89.59%, which is a measure of how many of the total predictions made by the model are correct. In other words, the model correctly classifies 89.59% of the total instances in your test dataset.

2. **Confusion Matrix:**

The confusion matrix is a table that shows how many true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) predictions the model made for each class.

- The confusion matrix is not in a square shape, indicating that your model is likely performing multiclass classification with more than two classes. Each row and column of the confusion matrix corresponds to a specific class.
- For example, the element at row 1, column 1 (979) represents the true positives for class 0, and the element at row 2, column 2 (100) represents the true positives for class 1.

3. Classification Report:

The classification report provides several evaluation metrics for each class, as well as overall metrics.

For each class (0, 1, 2, 3, 4, 5, 6, 7, 8), you have:

- **Precision**: This measures the accuracy of positive predictions. For example, for class 0, the precision is 0.88, which means that 88% of the instances predicted as class 0 were correct.
- **Recall:** This measures the model's ability to identify all relevant instances of a class. For example, for class 0, the recall is 0.88, indicating that 88% of the actual class 0 instances were correctly identified.
- **F1-score:** The F1-score is the harmonic mean of precision and recall. It provides a balanced measure of a model's performance. For class 0, the F1-score is 0.88.
- **Support:** The number of instances in each class in your test dataset.

4. **Accuracy, Macro Avg, Weighted Avg:** These are summary statistics:

- **Accuracy:** Already discussed, it's the overall accuracy of the model.
- **Macro Avg:** The macro-average is the average of the metrics (precision, recall, F1-score) calculated for each class independently. It gives equal weight to each class, regardless of its size.
- **Weighted Avg:** The weighted average is the same as the macro-average, but it takes into account the number of instances in each class. This means that classes with more instances have a greater impact on the average.

In summary, the model appears to have decent accuracy, but the classification report highlights that the performance varies significantly across different classes. For some classes, like class 5, the model performs very well, with high precision, recall, and F1 score. However, for other classes, like class 4 and class 3, the model's performance is relatively poor, with lower precision, recall, and F1-score. It's important to note that some classes, like class 2, have no support (zero instances), which makes it impossible for the model to make predictions for those classes.

RECOMMENDATIONS:

- **Feature Engineering:** For all three models, consider exploring feature engineering techniques to improve performance. Feature selection, scaling, and creating new features can enhance model accuracy.
- **Hyperparameter Tuning**: Experiment with hyperparameter tuning for each model to optimize their performance. Adjust parameters such as the number of estimators (for Random Forest), regularization strength (for Logistic Regression), and maximum depth (for Decision Tree).
- **Class Imbalance**: Address class imbalance if applicable. Use techniques like oversampling, undersampling, or class-weighted approaches to handle imbalanced datasets.
- **Ensemble Methods:** Explore ensemble methods such as Random Forest and Gradient Boosting to improve predictive accuracy.
- **Cross-Validation:** Implement cross-validation techniques like k-fold cross-validation to assess model generalization and reduce overfitting.
- **Feature Importance:** Analyze feature importance scores provided by Random Forest and Decision Tree models to gain insights into influential features.
- **Model Interpretability**: Consider using techniques like SHAP (SHapley Additive explanations) or LIME (Local Interpretable Model-agnostic Explanations) to interpret complex models like Random Forest.
- **Data Quality:** Ensure data quality by addressing missing values and outliers. Clean, high-quality data is crucial for model performance.

LIMITATIONS:

- **Class Imbalance:** Class imbalance can impact model performance, leading to biased predictions, especially for Logistic Regression and Decision Tree.
- **Overfitting:** All three models are susceptible to overfitting, especially when not properly tuned or when the dataset is noisy.
- **Interpretability:** Logistic Regression is more interpretable than Random Forest and Decision Tree. Complex models like Random Forest may lack interpretability, making it challenging to understand the reasoning behind predictions.
- **Computation Time:** Random Forest can be computationally intensive with a large number of trees, potentially limiting its use in real-time applications.
- **Extrapolation:** Decision Tree and Random Forest models may not perform well when extrapolating outside the range of training data.

- **Data Quality:** The quality of input data is critical, and noisy data or inaccuracies can adversely affect model performance for all three models.
- **Parameter Tuning:** Tuning hyperparameters can be time-consuming and may require multiple iterations to find the optimal settings for each model.
- **Noisy Data:** Noisy data and outliers can adversely affect model performance, especially for Decision Tree and Random Forest.
- **Limited Interpretability:** While Logistic Regression is more interpretable, Decision Tree and Random Forest may provide less intuitive insights into model decisions.