

Table of Contents

1	Introduction.....	4
2	Background Study.....	5
3	Assumption and Justification	6
4	Data Dictionary/ Metadata.....	7
4.1	Uploading the Dataset	8
4.2	Transfer Dataset to Permanent SAS Library	8
4.3	Data Dictionary	9
4.4	Metadata	10
4.4.1	Structure of the Dataset – LIB77702.TRAINING_DS	10
5	Literature Review.....	11
5.1	Enhancing Loan Processing Efficiency	11
5.2	AI in Predicting Loan Defaults.....	12
5.3	Managing Bad Debts with AI.....	13
5.4	Challenges and Future Directions.....	14
6	Analysis of the Variables – LIB77702.TRAINING_DS	16
6.1	Univariate Analysis of the Variable in the Dataset LIB77702.TRAINING_DS	16
6.1.1	Analysis of the Categorical Variables in the Dataset	16
6.1.2	Analysis of the Continuous Variable in the dataset	21
6.2	Bivariate Analysis of Variables in the Dataset LIB77702.TRAINING_DS	25
6.2.1	Bivariate Analysis of the Variable (Categorical vs Categorical)	25
6.2.2	Bivariate Analysis of the Variable (Categorical vs Continuous)	28
7	Analysis of Variables – LIB77702.TESTING_DS	30
7.1	Introduction to SAS MACRO	30
7.2	Univariate Analysis of the Variables using SAS MACRO – LIB77702.TESTING_DS	30
7.2.1	Analysis of the Categorical Variables in the Dataset	30
7.2.2	Analysis of the Continuous Variables in the Dataset	32
7.3	Bivariate Analysis of the variable Using SAS MACRO – LIB7702.TESTING_DS	34
7.3.1	Bivariate Analysis of the variable (Categorical vs Categorical)	34
7.3.2	Bivariate Analysis of the variable (Categorical vs Continuous)	37
8	Imputation of Missing Values.....	40
8.1	Imputing the Missing Values in Categorical Variables – LIB7702.TRAINING_DS	40
8.1.1	Handling Missing Values in MARITAL_STATUS.....	40
8.1.2	Handling Missing Values in FAMILY_MEMBERS	43
8.1.3	Handling Missing Values in EMPLOYMENT	48
8.1.4	Handling Missing Values in GENDER.....	51
8.1.5	Handling Missing Values in LOAN_HISTORY	55

8.2	Imputing the missing values in Continuous variables – LIB7702.TRAINING_DS	58
8.2.1	Handling Missing Values in LOAN_AMOUNT	58
8.2.2	Handling Missing Values in LOAN_DURATION	61
8.3	Imputing the missing values in Categorical variables – LIB7702.TESTING_DS	63
8.3.1	Handling Missing Values in FAMILY_MEMBERS	63
8.3.2	Handling Missing Values in EMPLOYMENT	67
8.3.3	Handling Missing Values in GENDER.....	71
8.3.4	Handling Missing Values in LOAN_HISTORY.....	74
8.4	Imputing the missing values in Continuous variables – LIB7702.TESTING_DS	78
8.4.1	Handling Missing Values in LOAN_AMOUNT	78
8.4.2	Handling Missing Values in LOAN_DURATION	80
9	Logistic Regression.....	83
9.1	Creation of the Model using Logistic Regression Algorithm.....	83
9.2	Predicting the Loan Approval Status Using Logistic Regression.....	85
10	Data Visualization and Report Generation	86
10.1	Data Visualization	86
10.1.1	Simple Bar Chart	86
10.1.2	Stacked Bar Chart	86
10.1.3	Pie chart	87
10.1.4	Stacked Pie Chart.....	88
10.2	Report Generation using SAS ODS.....	89
10.2.1	Display Details of Predicted Loan Approvals Status.....	89
10.2.2	Generated Report from SAS ODS	89
11	Conclusion	91
12	References.....	92

1 Introduction

In the current economic environment, using sophisticated analytical software in banking has transformed how financial institutions manage risk and improve their services. A company situated in New York called Lasiandra Finance Inc. (LFI) which is a leading private mortgage lender is one of the pioneers of this approach. The company uses complex data analysis programs to facilitate fast loan approval processes. This paper focuses on examining the way logistic regression models and data management methods are employed at LFI to support small and medium-sized enterprises (SMEs) decision-making as well as operational efficiency.

There are two main goals for this research study. Firstly, it seeks to determine if logistic regression can be used to predict whether or not loans will be approved based on certain borrower characteristics such as marriage status, income level and employment among others. Secondly, it addresses issues related to missing information by imputing the same so that any model used becomes more robust and accurate.

This task heavily relies on SAS language for automating different data manipulation routines required especially when dealing with large volumes common within bank loan dataset. Furthermore, through univariate analysis alone followed by bivariate analyses also being conducted; specific insights with regards to what each individual variable contributes towards influencing decisions on approving loans shall be brought out.

By incorporating sophisticated types of analysis into their systems, LFI hopes not just to make its operations more efficient but also reliable in serving clients. Consequently, they anticipate that there will be faster processing speeds, decreased cases of defaulting on credit facilities granted while at the same time enhancing overall customer satisfaction levels. Moreover, by strategically embracing such technologies it proves their willingness towards staying ahead of competition within an industry that is highly dynamic thus becoming a role model for other organizations operating in similar sectors.

2 Background Study

Lasiandra Finance Inc. (LFI) is an established and recognized private financing company in New York, USA, with a financial portfolio for support SMEs with custom financial solutions. LFI is committed to many economic solutions in business and business loans, financial consulting, and strategic investment solutions supporting business development and growth.

A key strength of LFI is the innovative use of advanced analytics with machine-learning algorithms that automate the loan approval process. This infusion of technology into LFI would provide loaning services that are even more efficient and precise to decrease the time and effort businesses would have to invest to secure funding. By processing vast amounts of financial data, LFI can make accurate decisions relating to credit, which, in turn, are instrumental in managing the financial risk and lifting the performance of the loans. The operations of LFI are marked with a strong customer orientation. It designs an individual financial solution for each unique need of the business customer, which helps specifically serve challenges and opportunities in their business. The personalized service model allows customers properly to overcome financial challenges and effectively seize opportunities.

In addition to this, the address at 230 Park Avenue, 3rd Floor, New York, NY, is one of the prime addresses of the United States, if not the whole world, of a leading global financial centre. This puts LFI in a position to tap into many resources and networks that the financial industry has to offer. This strategic location further cements LFI's ability to be in touch with a wide range of client interests and to establish firm connections with the industry at large. All this underlines the company's commitment to innovation and delivery of what the customer needs. From several company projects and initiatives details of which can be found on the website and in the repository at GitHub. LFI reinforces its commitment to transparency and continued improvement of its financial services.

In a nutshell, Lasiandra Finance Inc. is a visioned financial institution, incorporating technological innovation within the understanding of client needs towards delivering remarkable financial solutions. Strategically and with an attitude of excellence, Lasiandra Finance Inc. is a vital partner to SMEs desiring success in sustainable growth.

3 Assumption and Justification

Several critical assumptions and justifications resulted in the use of SAS for the above assignment regarding loan prediction data analysis. It proved its aptness in handling complicated data-related tasks. Firstly, it is assumed, that both categorical and continuous variables will dominate the dataset, and consequently, a tool that can apply this wide range of various statistical analyses has to be used. As a propagative statistical software, SAS enables the conduction of all sorts of EDA by performing univariate, bivariate, and multivariate analyses. The robust set of procedures such as PROC FREQ, PROC MEANS, and PROC UNIVARIATE provide elaborate summaries of even the most trivial details for the distribution and relationships of data, along with as much visualization as possible. The use of SAS also implies that large datasets must be manipulated efficiently. SAS has been optimized to handle and analyse vast data, and its speed in doing so is still very high, a factor that may turn out quite valuable for loan prediction given the datasets used are generally massive and hence complex, thus needing much power computationally to run through them properly.

Another critical assumption is the requirement to have advanced data manipulation and transformation. As SAS offers the ultimate data manipulation regarding DATA step programming and various functions, sophisticated data cleaning, transformation, and feature engineering could be carried out. Such flexibility in data preprocessing becomes a must for preparing data for predictive modelling so that all relevant factors get comprehensively captured and analysed. Also, comprehensive support for different types of statistical testing and hypothesis validation justifies using SAS in loan prediction. Here, in the case of loan prediction, understanding the relations between variables such as marital status, employment status, and Loan amounts is crucial. For that, reliable statistical tests should be available. SAS provides the broadest range of tests and diagnosis tools to check for assumptions, data anomalies, and general robustness of analytical results.

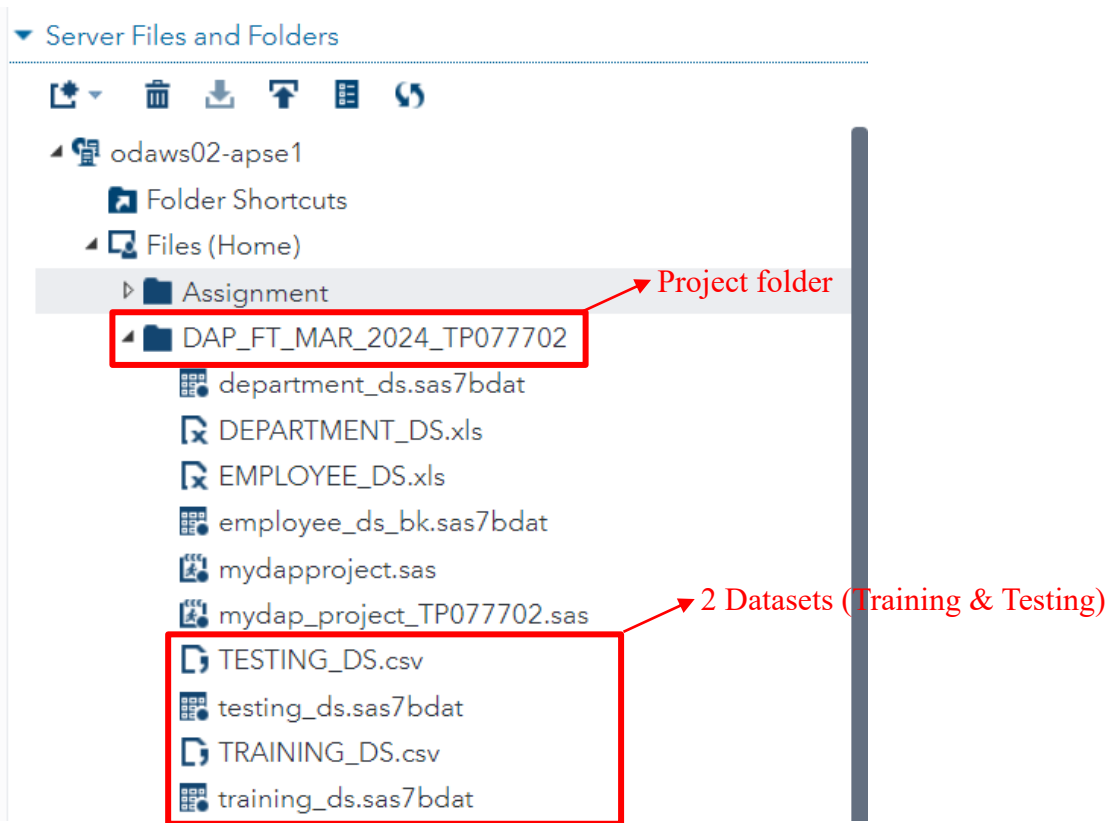
Further, SAS facilitates knowledge distribution by producing reports and visualizations deep in insight. The outputs are marvellous and, thus, best arranged in a way that they explain complex statistical outcomes in such a manner that the user has little bother to acquire insights³³⁷ and inform the resolution semblance. Specifically, due to its highly developed statistical powers, effectiveness with large scales of data, advanced powers of data manipulation, totality in statistical testing support, and the capability of producing detailed and interpretable reports, SAS is used in the above loan prediction data analysis.

4 Data Dictionary/ Metadata

Name of Variable	Description	Data Type	Length	Sample Data
SME_LOAN_ID_NO	Loan application number	Char	8	LP001002 LP001003 LP001005
GENDER	Gender of the applicant	Char	6	Female; Male
MARITAL_STATUS	Marital Status of the applicant	Char	11	Married; Not Married
FAMILY_MEMBERS	Number of family members of the applicant	Char	2	1, 2, 3+
QUALIFICATION	Education Qualification of the applicant	Char	14	Graduate; Undergraduate
EMPLOYMENT	Employment Status of the applicant	Char	3	Yes; No
CANDIDATE_INCOME	Income of the applicant	Numeric	5	5849, 4583, 3000
GUARANTEE_INCOME	Income of Joint Applicant	Numeric	5	1508, 2358, 4196
LOAN_AMOUNT	Loan amount	Numeric	5	128, 66, 120
LOAN_DURATION	Duration of Loan Tenure	Numeric	3	71, 360
LOAN_HISTORY	Loan History of the applicant	Numeric	1	0; 1
LOAN_LOCATION	Location of the application	Char	7	City, Village, Town
LOAN_APPROVAL_STATUS	Approval Status of Loan Application	Char	1	Y; N (Y=Yes,N=No)

4.1 Uploading the Dataset

4.1.1.1 Screenshot

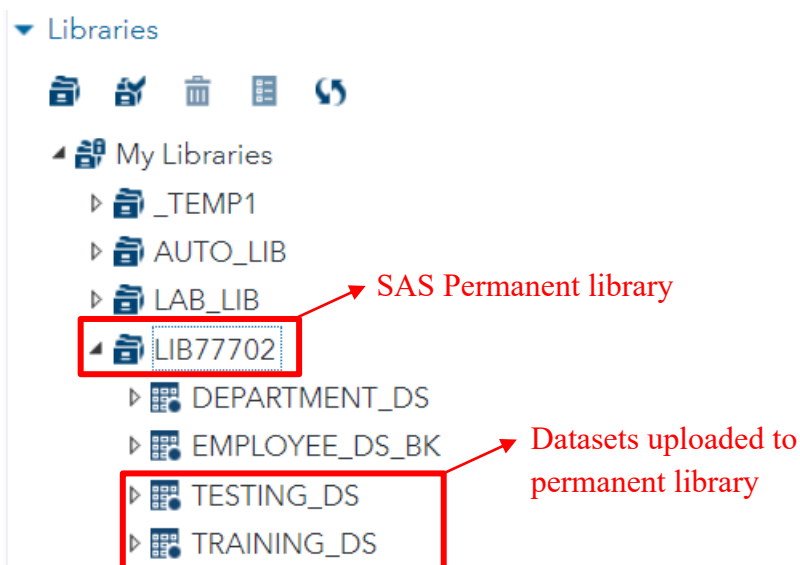


4.1.1.2 Description

The two dataset TRAINING_DS and TESTING_DS is stored in the permanent project folder DAP_FT_MAR_2024_TP077702.

4.2 Transfer Dataset to Permanent SAS Library

4.2.1.1 Screenshot

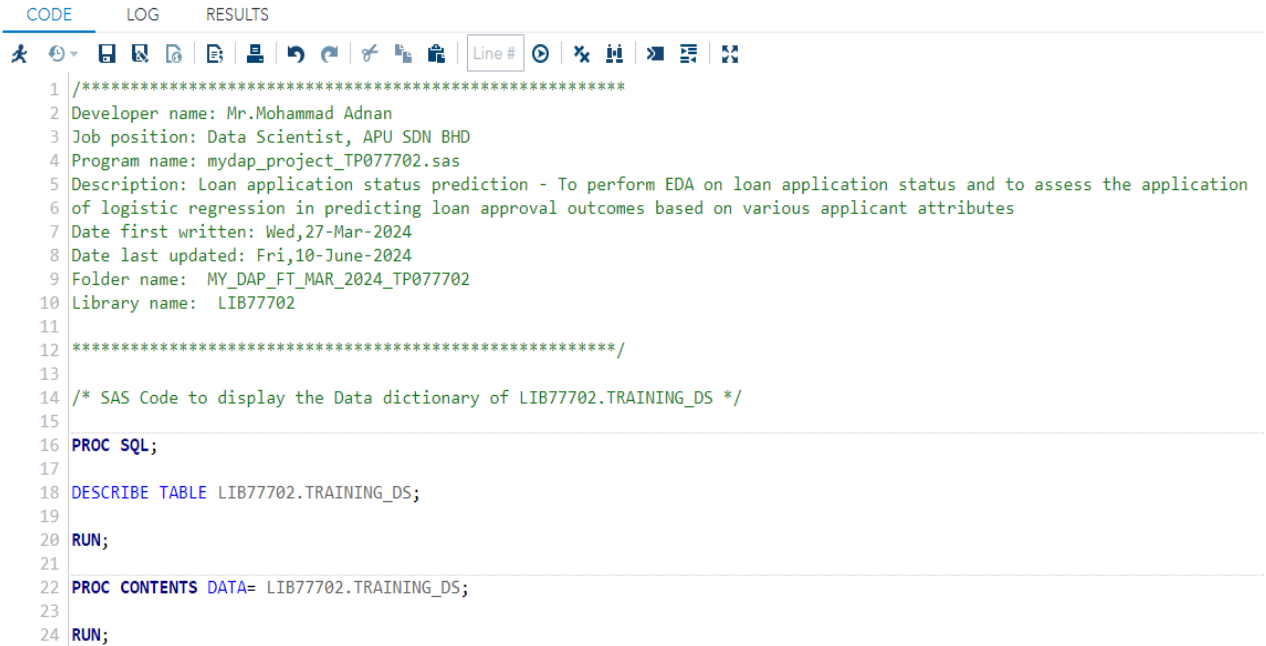


4.2.1.2 Description

The dataset TRAINING_DS and TESTING_DS are then uploaded to the SAS permanent library LIB77702 which can be accessed easily.

4.3 Data Dictionary

4.3.1.1 Code Snippet



```

CODE    LOG    RESULTS
1  /*****
2  Developer name: Mr.Mohammad Adnan
3  Job position: Data Scientist, APU SDN BHD
4  Program name: mydap_project_TP077702.sas
5  Description: Loan application status prediction - To perform EDA on loan application status and to assess the application
6  of logistic regression in predicting loan approval outcomes based on various applicant attributes
7  Date first written: Wed,27-Mar-2024
8  Date last updated: Fri,10-June-2024
9  Folder name: MY_DAP_FT_MAR_2024_TP077702
10 Library name: LIB77702
11
12 *****/
13
14 /* SAS Code to display the Data dictionary of LIB77702.TRAINING_DS */
15
16 PROC SQL;
17
18 DESCRIBE TABLE LIB77702.TRAINING_DS;
19
20 RUN;
21
22 PROC CONTENTS DATA= LIB77702.TRAINING_DS;
23
24 RUN;

```

4.3.1.2 Output

```

create table LIB77702.TRAINING_DS( bufsize=131072 )
(
  SME_LOAN_ID_NO char(8) format=$8. informat=$8.,
  GENDER char(6) format=$6. informat=$6.,
  MARITAL_STATUS char(11) format=$11. informat=$11.,
  FAMILY_MEMBERS char(2) format=$2. informat=$2.,
  QUALIFICATION char(14) format=$14. informat=$14.,
  EMPLOYMENT char(3) format=$3. informat=$3.,
  CANDIDATE_INCOME num format=BEST12. informat=BEST32.,
  GUARANTEE_INCOME num format=BEST12. informat=BEST32.,
  LOAN_AMOUNT num format=BEST12. informat=BEST32.,
  LOAN_DURATION num format=BEST12. informat=BEST32.,
  LOAN_HISTORY num format=BEST12. informat=BEST32.,
  LOAN_LOCATION char(7) format=$7. informat=$7.,
  LOAN_APPROVAL_STATUS char(1) format=$1. informat=$1.
);

```

4.3.1.3 Description

Above output using DESCRIBE TABLE from PROC SQL to get brief overview of the data types and size in the dataset.

4.4 Metadata

4.4.1 Structure of the Dataset – LIB77702.TRAINING_DS

4.4.1.1 Code Snippet

```
21 PROC CONTENTS DATA= LIB77702.TRAINING_DS;
22
23 RUN;
```

4.4.1.2 Output

The CONTENTS Procedure			
Data Set Name	LIB77702.TRAINING_DS	Observations	614
Member Type	DATA	Variables	13
Engine	V9	Indexes	0
Created	04/26/2024 15:53:14	Observation Length	96
Last Modified	04/26/2024 15:53:14	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64		
Encoding	utf-8 Unicode (UTF-8)		

Engine/Host Dependent Information	
Data Set Page Size	131072
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	1363
Obs in First Data Page	614
Number of Data Set Repairs	0
Filename	/home/u63691876/DAP_FT_MAR_2024_TP077702/training_ds.sas7bdat
Release Created	9.0401M7
Host Created	Linux
Inode Number	8589936855
Access Permission	rw-r--r--
Owner Name	u63691876
File Size	256KB
File Size (bytes)	262144

Alphabetic List of Variables and Attributes					
#	Variable	Type	Len	Format	Informat
7	CANDIDATE_INCOME	Num	8	BEST12.	BEST32.
6	EMPLOYMENT	Char	3	\$3.	\$3.
4	FAMILY_MEMBERS	Char	2	\$2.	\$2.
2	GENDER	Char	6	\$6.	\$6.
8	GUARANTEE_INCOME	Num	8	BEST12.	BEST32.
9	LOAN_AMOUNT	Num	8	BEST12.	BEST32.
13	LOAN_APPROVAL_STATUS	Char	1	\$1.	\$1.
10	LOAN_DURATION	Num	8	BEST12.	BEST32.
11	LOAN_HISTORY	Num	8	BEST12.	BEST32.
12	LOAN_LOCATION	Char	7	\$7.	\$7.
3	MARITAL_STATUS	Char	11	\$11.	\$11.
5	QUALIFICATION	Char	14	\$14.	\$14.
1	SME_LOAN_ID_NO	Char	8	\$8.	\$8.

4.4.1.3 Description

The output above provides a detailed overview of the dataset by identifying the file location, as well as its type, length, format, and other details.

5 Literature Review

The banking sector, in processing loans, assessing the credit risk of prospective borrowers, and managing bad debt, has vividly seen the effect of AI technologies. Advanced document recognition, automated underwriting, and deep learning models have transformed the way we process loans, speeding up loan approval time and reducing errors. The emergence of machine learning algorithms has also improved the ability to predict credit defaults, assisting financial risk management by banks. AI has played a major role in bad debt management by identifying likely defaulters and predicting bankruptcies, effectively mitigating and controlling the risk of non-performing loans. This progress comes with its fair share of challenges, including privacy concerns, bias, and the requirement for new and robust regulatory frameworks. Finally, collaboration with industry experts and researchers, as well as bias-mitigation strategies, will be essential for the ethical and responsible use of AI in banking (Mehndiratta, Arora, & Bathla, 2023).

5.1 Enhancing Loan Processing Efficiency

AI technologies have indeed revolutionized the process of loan processing through the automation of operations, therefore increasing efficiencies and significantly reducing errors. One of the significant developments in this area is intelligent document recognition systems. It consists of the automation of verification of financial statements, performed by different techniques—image segmentation, classification, and the optical character recognition process. As a result, an absolute minimum amount of work remains to be done manually, and the processing time for loan issuance is accelerated dramatically (Hsu, 2020). In the end, banks become more efficient in their operations with fewer errors, eventually reducing their reliance on manual entry of data, which accrues to the good in banks and their customers.

Models developed through AI have also changed the whole process of underwriting credit. It allows a bank to analyse vast sets of data to decide the credit possible for the applicant faster and more accurately than under low-tech regimes (Jain & Verma, 2022). Automation in decision-making speeds up the application process and guarantees that only creditworthy applicants are approved, thus reducing the risk of financial loss to the bank. The implication of AI in banking is beyond the mere integration with processing efficiency; it enhances the service quality and customer experience. As put forward by Rana et al., (2023) AI applications have notably improved customer service and operational efficiency. For example, chatbots can

handle periodic customer inquiries, thus freeing up human agents to confront more rigorous issues. This results in quick response times and increased customer satisfaction.

For example, real-time deep learning models have taken the loan approval process to the next level. These use intense neural networks to allow for the analysis of data in real-time, hence making better more reliable, and timely decisions about applicants for loans. In the process, the models can categorize loan applicants as good and bad risks with high precision, recall, and accuracy, thereby shielding only the creditworthy applicants, hence overall improving the loan approval process (Abakarim, Lahby, & Attioui, 2018). In short, AI technologies are driving changes in loan processing via task automation, enhanced accuracy, and better customer service. These advances substantially ease operations and establish a more reliable and efficient banking experience for the people involved.

5.2 AI in Predicting Loan Defaults

The application of machine learning models in the area of prediction of loan default has exhibited phenomenal effectiveness in identifying patterns and trends that might have been red lights for potential defaulters. These models are basically driven by data in their quest to realize patterns and trends that have emerged to show whether loan defaults can occur. For instance, Dong (2022) investigates the capability of a machine-learning model, LightGBM, to predict loan defaults effectively after processing vast volumes of lender information, resulting in a high Area Under the Curve value as a strong indicator of its prediction strength. Other comparison studies have cited the superiority of ensemble learners like XGBoost and AdaBoost in better prediction of loan defaults (Shaheen & Fakharany, 2018) (Odegua, 2020). These algorithms involve advanced technologies in combining multiple models to enhance accuracy, and their performance has been better than the individual models, having high recall, precision, and F1 scores. AdaBoost came on top of all other models based on the findings by Lai (2020) on a real-world dataset from an international bank comparing the performances between AdaBoost, XGBoost, random forest, k-nearest neighbours, and multilayer perceptron's. The best performance was obtained by AdaBoost, with perfect accuracy in predicting loan defaults.

Besides, it has proven to be an efficient technique to enlarge the model's performance when carried out at a more comprehensive preprocessing and feature selection level (Al-qerem, Alnaymat, & Alhasan, 2020). With the novelty in every possible preprocessing technique and feature extraction algorithm, the augmentation of data quality and choice of relevant features can accordingly enlarge the accuracy and general performance of the prediction models. It has

also been implemented along with Synthetic Minority Over-sampling Technique (SMOTE) to correct imbalanced training data, increasing prediction accuracy (Shingi, 2020). Utilizing federated learning, model weights can be shared among organizations without leaks of data, and combined with SMOTE, that is going to increase the teaching of a model from an imbalanced dataset and lead to further accurate predictions.

In summary, AI-driven machine learning models play a critical role in accurately predicting loan defaults, thereby enhancing risk management in banking. By leveraging advanced algorithms, data engineering techniques, and federated learning, banks can improve their ability to predict loan defaults and manage financial risks effectively.

5.3 Managing Bad Debts with AI

Effective management of bad debts is critical to maintaining the financial stability of banks. AI technologies have shown potential in this regard. Predictive models such as Random Forest and data mining algorithms have identified potential defaulters appropriately and helped the banks become proactive (Widiyono & Isa, 2020). Such models help banks properly predict non-performing loans (NPL) to decrease bad debts and improve the financial stability of any bank. Other than that, AI facilitates advanced risk assessment models whereby it has improved the accuracy of bankruptcy assessment, such as logistic and neural network techniques for improved bankruptcy assessment (Chen, 2022). These are models examining data from customer attributes and transaction behaviour to predict the financial default behaviour, hence significantly improving the bankruptcy prediction accuracies and assisting in risk management for the banking industries.

The development of autonomous decision-making AI on credits, based on biometric data and credit history, has significantly increased loan risk management (Pavlikov, 2020). This intelligent engine can, in split seconds, crunch through heaps of data points that include biometric information in making informed decisions related to loans, which will reduce risks of defaults and increase the general process of loan management.

In this line, further transparency and reliability of credit scoring systems, including these XAI models, have increased regulatory compliance and conferred greater trust from the users (Lange et al., 2022). More importantly, the models using the combination of Light GBM with SHAP for model interpretation increase the explainability of predictions, making decisions transparent. The most crucial explanation variable was credit balance volatility, followed by

remaining credit percentage and customer relationship lifetime, all indicative of the potential for XAI models to provide insights into the fundamentals driving loan default predictions.

Further, weighted training models using decision tree algorithms are demonstrated to be effective in predicting loan defaults (Pang & Yuan, 2018). These models, being in use with the help of weighted calculating algorithms and combining models with decision trees like C5.0, CART, and CHAID, have enabled high comparison accuracy and warning rates, which confirms effectiveness in revealing potential defaulters which in result helps us in managing bad debts. In short, AI technologies play a crucial role in managing bad debts by improving prediction accuracy, enhancing risk assessment models, and providing transparent and reliable credit scoring systems.

5.4 Challenges and Future Directions

Although there are so many benefits of applying AI in banking, there are still several challenges in this area. The related issues of data privacy and security must be dealt with by the introduction of AI solutions in financial institutions through customer data protection and responsible use to be trusted to be used in a manner consistent with the obligatory requirements on the subject (Rana, et al., 2023). Nonetheless, biases of AI models can still lead to discriminatory outputs, and loan processing and credit scoring remain a significant frontier where the challenge remains paramount (Lange et al., 2022). The dispatch of such a policy is the key to building trustful and fair AI systems, ensured through algorithmic fairness techniques and bias mitigation, where no customer is treated differently from the other.

Other concerns include the need to develop and pursue appropriate regulatory frameworks that will control the application of AI in banking (Mehndiratta, Arora, & Bathla, The use of Artificial Intelligence in the Banking Industry, 2023). It is essential to regularly monitor and update such frameworks to keep pace with technological advances and changing regulatory environments in promoting the ethical use of AI in banking and compliance with financial services regulations. Therefore, future directions will require looking at new directions in AI, including deep reinforcement learning and GAN for improving accuracy in prediction and enabling better decision-making. Adding command-and-control ability may improve decision-making, leading to security and transparency in bank operations, hence more trust and reliability.

Continuous improvement in AI models through advanced data engineering and feature selection techniques would make them superior in performance and reliability. Transfer learning and meta-learning techniques can be used to improve generalization and accommodate new data distributions so that models remain effective in a changing environment. Further, developing the interpretability and explainability of AI models using Explainable AI techniques is very important to get regulatory approval and trust from users, especially in processes such as credit scoring and loan approval (Lange et al., 2022). LIME Technology, like SHAP will help in understanding the model's reasoning and thereby inculcate transparency and accountability.

Collaborations among researchers, financial institutions, and regulatory bodies can yield innovation in banking with AI that is both ethical and value based. Such interdisciplinary teams, made up of data scientists, domain experts, and legal professionals, can thereby translate research to practice most effectively, always considering regulation concerns. Besides, a mitigation strategy of biases in the AI model would be good practice to ensure fairness and equal treatment to all the customers. Adversarial debiasing, calibrated equalization, and many others are factoring techniques that can be effective in bias mitigation (González-Sendino, Serrano, & Bajo, 2024). Algorithm fairness is fostered in loan processing and credit scoring.

To summarize AI technologies have revolutionized the banking industry, driving effectiveness, sound risk management, and efficiencies in processing loans, managing credit risks, and mitigating bad debts. However, the journey to more AI deployments should not forget the requirements to overcome issues covering data privacy, algorithmic bias, and a regulatory fit for the sector. This will involve innovative scientific research in concert with financial institutions and regulatory authorities to support ethical standards and fairness. All these can be achieved with advanced data engineering, developed AI techniques for explanations, and strategies for bias mitigation for continuous model reliability, transparency, and accountability. In this way, the financial sector will be able to unlock the full potential of AI without affecting consumer trust by prioritizing responsible AI practices, fostering interdisciplinary collaborations, and embracing interpretable and unbiased models. The future will see AI being smartly integrated into bank operations in a balanced way: technological advancement and ethical considerations for a robust, transparent, and just financial ecosystem.

6 Analysis of the Variables – LIB77702.TRAINING_DS

6.1 Univariate Analysis of the Variable in the Dataset LIB77702.TRAINING_DS

6.1.1 Analysis of the Categorical Variables in the Dataset

6.1.1.1 Univariate analysis of the Categorical variables – MARITAL_STATUS

6.1.1.1.1 Code Snippets

```

25 TITLE 'Figure no - Univariate Analysis of the categorical variable: MARITAL_STATUS';
26
27 PROC FREQ DATA = LIB77702.TRAINING_DS;
28
29 TABLE marital_status;
30
31 RUN;
32
33 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
34
35 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
36
37 VBAR marital_status;
38
39 TITLE 'Univariate Analysis of the categorical';
40
41 RUN;

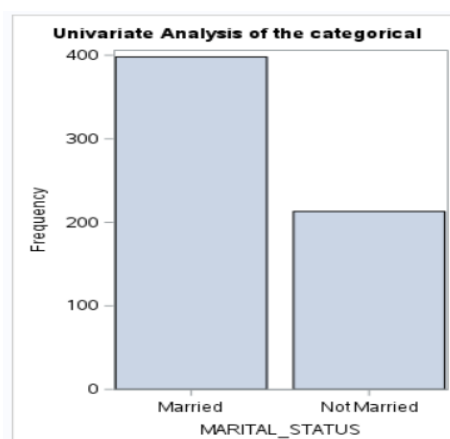
```

6.1.1.1.2 Output

Figure no - Univariate Analysis of the categorical variable: MARITAL_STATUS

The FREQ Procedure

MARITAL_STATUS	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Married	398	65.14	398	65.14
Not Married	213	34.86	611	100.00
Frequency Missing = 3				



6.1.1.1.3 Description

Both the table and graph show that there are 398 applicants who are married and 213 applicants unmarried. This averages around 65.14% of the applicants are married, and 34.86% are not married. There are presences 3 missing values in the marital status variable.

6.1.1.2 Univariate analysis of the categorical variables – FAMILY_MEMBERS

6.1.1.2.1 Code Snippet

```

43 TITLE 'Figure no - Univariate Analysis of the categoriical variable: FAMILY_MEMBERS';
44
45 PROC FREQ DATA = LIB77702.TRAINING_DS;
46
47 TABLE family_members;
48
49 RUN;
50
51 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
52
53 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
54
55 VBAR family_members;
56
57 TITLE 'Univariate Analysis of the categorical';
58
59 RUN;

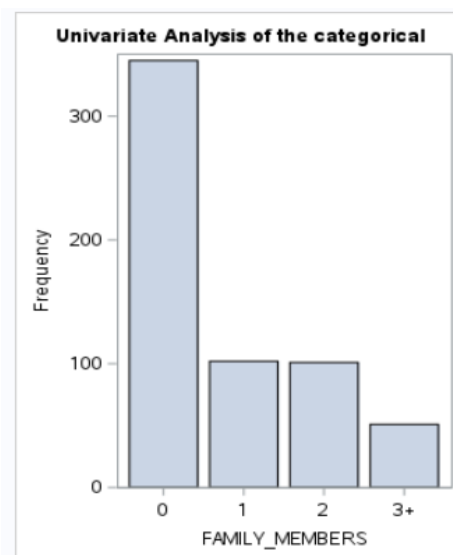
```

6.1.1.2.2 Output

Figure no - Univariate Analysis of the categoriical variable: FAMILY_MEMBERS

The FREQ Procedure

FAMILY_MEMBERS	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	345	57.60	345	57.60
1	102	17.03	447	74.62
2	101	16.86	548	91.49
3+	51	8.51	599	100.00
Frequency Missing = 15				



6.1.1.2.3 Description

The table and graph show the distribution of a categorical variable: family members. Most households have no family members (57.6%) followed by households with 1 (17.0%) and 2 (16.9%) family members and 3+ (8.5%) family member. There are also 15 missing values.

6.1.1.3 Univariate analysis of the categorical variables – LOAN_LOCATION

6.1.1.3.1 Code Snippet

```

61 TITLE 'Figure no - Univariate Analysis of the categoriical variable: LOAN_LOCATION';
62
63 PROC FREQ DATA = LIB77702.TRAINING_DS;
64
65 TABLE loan_location;
66
67 RUN;
68
69 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
70
71 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
72
73 VBAR loan_location;
74
75 TITLE 'Univariate Analysis of the categorical';
76
77 RUN;

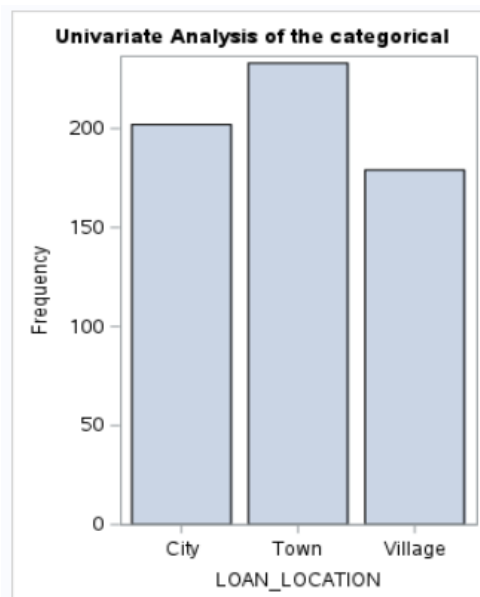
```

6.1.1.3.2 Output

Figure no - Univariate Analysis of the categoriical variable: LOAN_LOCATION

The FREQ Procedure

LOAN_LOCATION	Frequency	Percent	Cumulative Frequency	Cumulative Percent
City	202	32.90	202	32.90
Town	233	37.95	435	70.85
Village	179	29.15	614	100.00



6.1.1.3.3 Description

The table and graph show the frequency of loan repayments for each type of loan location. The frequency of loan repayments for each type of loan varies depending on the location. Town has the highest frequency (37.95%) followed by City (32.90%) and Village (29.15%).

6.1.1.4 Univariate analysis of the categorical variables – GENDER

6.1.1.4.1 Code Snippet

```

79 TITLE 'Figure no - Univariate Analysis of the categoriical variable: GENDER';
80
81 PROC FREQ DATA = LIB77702.TRAINING_DS;
82
83 TABLE gender;
84
85 RUN;
86
87 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
88
89 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
90
91 VBAR gender;
92
93 TITLE 'Univariate Analysis of the categorical';
94
95 RUN;

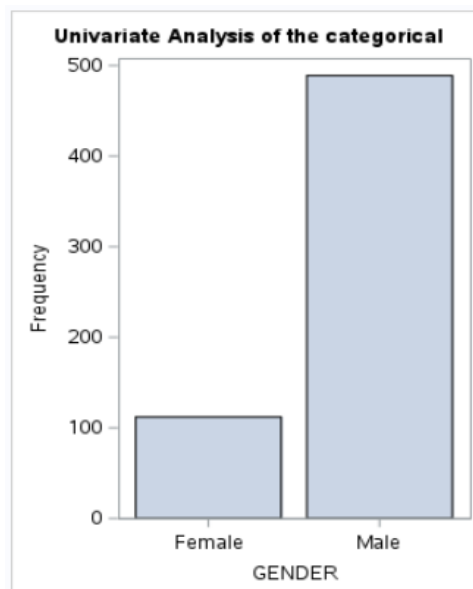
```

6.1.1.4.2 Output

Figure no - Univariate Analysis of the categoriical variable: GENDER

The FREQ Procedure

GENDER	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Female	112	18.64	112	18.64
Male	489	81.36	601	100.00
Frequency Missing = 13				



6.1.1.4.3 Description

The table and graph show the frequency distribution for each gender in a categorical variable. There are 112 females (18.64%) and 489 males (81.36%) with data recorded, and 13 entries missing gender data. Overall, the table shows a higher frequency of males in the dataset.

6.1.1.5 Univariate analysis of the categorical variables – Qualification

6.1.1.5.1 Code Snippet

```

97 TITLE 'Figure no - Univariate Analysis of the categorical variable: Qualification';
98
99 PROC FREQ DATA = LIB77702.TRAINING_DS;
100
101 TABLE qualification;
102
103 RUN;
104
105 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
106
107 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
108
109 VBAR qualification;
110
111 TITLE 'Univariate Analysis of the categorical';
112
113 RUN;

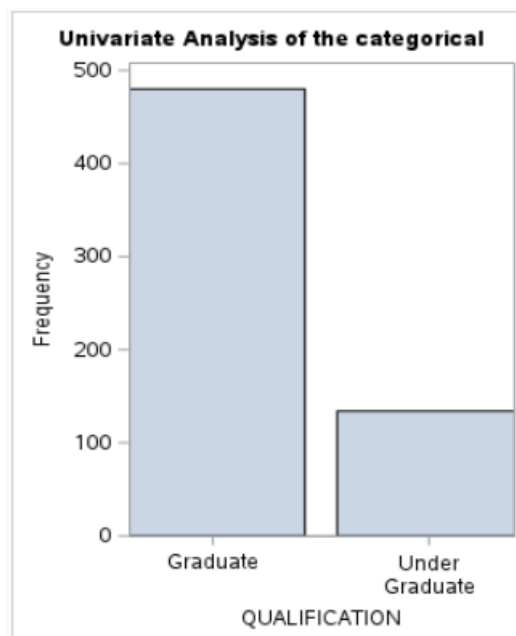
```

6.1.1.5.2 Output

Figure no - Univariate Analysis of the categorical variable: Qualification

The FREQ Procedure

QUALIFICATION	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Graduate	480	78.18	480	78.18
Under Graduate	134	21.82	614	100.00



6.1.1.5.3 Description

The table and graph show the frequency of graduates and undergraduates. There are 480 graduates (78.18%) and 134 undergraduates (21.82%). This suggests a higher frequency of graduates in the data set.

6.1.2 Analysis of the Continuous Variable in the dataset

6.1.2.1 Univariate Analysis of the Continuous Variable – LOAN_AMOUNT

6.1.2.1.1 Code Snippet

```

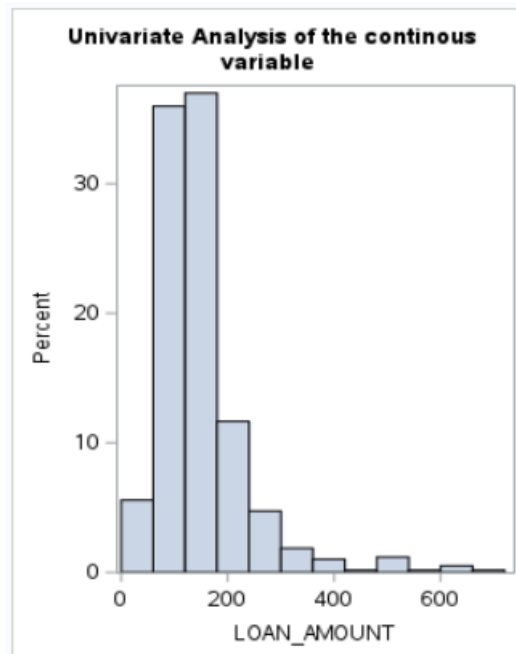
133 TITLE 'Figure no - Univariate Analysis of the continous/numeric variable: LOAN_DURATION';
134
135 PROC MEANS DATA = LIB77702.TRAINING_DS N NMISS MIN MAX MEAN MEDIAN STD;
136
137 VAR loan_duration;
138
139 RUN;
140
141 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
142
143 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
144
145 HISTOGRAM loan_duration;
146
147 TITLE 'Univariate Analysis of the categorical';
148
149 RUN;

```

6.1.2.1.2 Output

Figure no - Univariate Analysis of the continous/numeric variable: LOAN_AMOUNT

The MEANS Procedure						
Analysis Variable : LOAN_AMOUNT						
N	N Miss	Minimum	Maximum	Mean	Median	Std Dev
592	22	9.0000000	700.0000000	146.4121622	128.0000000	85.5873252



6.1.2.1.3 Description

The univariate analysis shows that the average loan amount is \$146.41 with a standard deviation of \$85.58. There are 592 loans included in the analysis, with a minimum loan amount of \$9 and a maximum loan amount of \$700. There are also 22 missing values.

6.1.2.2 Univariate Analysis of the continuous variable – LOAN_DURATION

6.1.2.2.1 Code Snippet

```

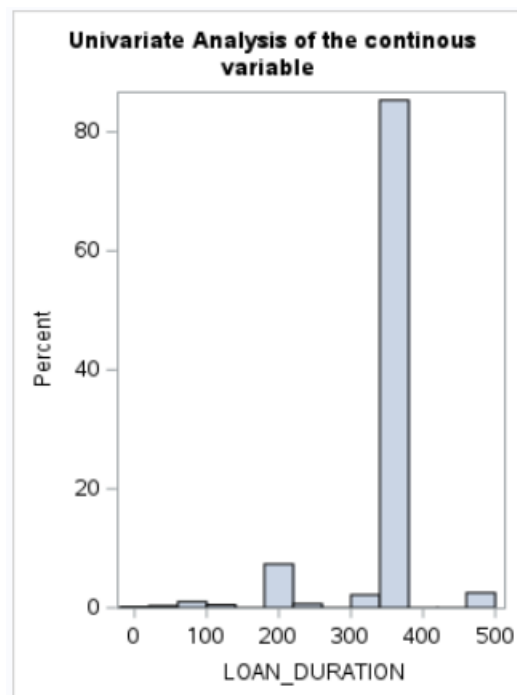
133 TITLE 'Figure no - Univariate Analysis of the continous/numeric variable: LOAN_DURATION';
134
135 PROC MEANS DATA = LIB77702.TRAINING_DS N NMISS MIN MAX MEAN MEDIAN STD;
136
137 VAR loan_duration;
138
139 RUN;
140
141 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
142
143 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
144
145 HISTOGRAM loan_duration;
146
147 TITLE 'Univariate Analysis of the continous variable';
148
149 RUN;

```

6.1.2.2.2 Output

Figure no - Univariate Analysis of the continous/numeric variable: LOAN_DURATION

The MEANS Procedure						
Analysis Variable : LOAN_DURATION						
N	N Miss	Minimum	Maximum	Mean	Median	Std Dev
600	14	12.0000000	480.0000000	342.0000000	360.0000000	65.1204099



6.1.2.2.3 Description

The univariate analysis shows values for loan duration. The univariate analysis shows an average loan duration of 342 days, with a median of 360 days. The loan duration ranges from 12 days to 480 days. There are 600 loans included in the analysis, with 14 missing values.

6.1.2.3 Univariate Analysis of the continuous variable – CANDIDATE_INCOME

6.1.2.3.1 Code Snippet

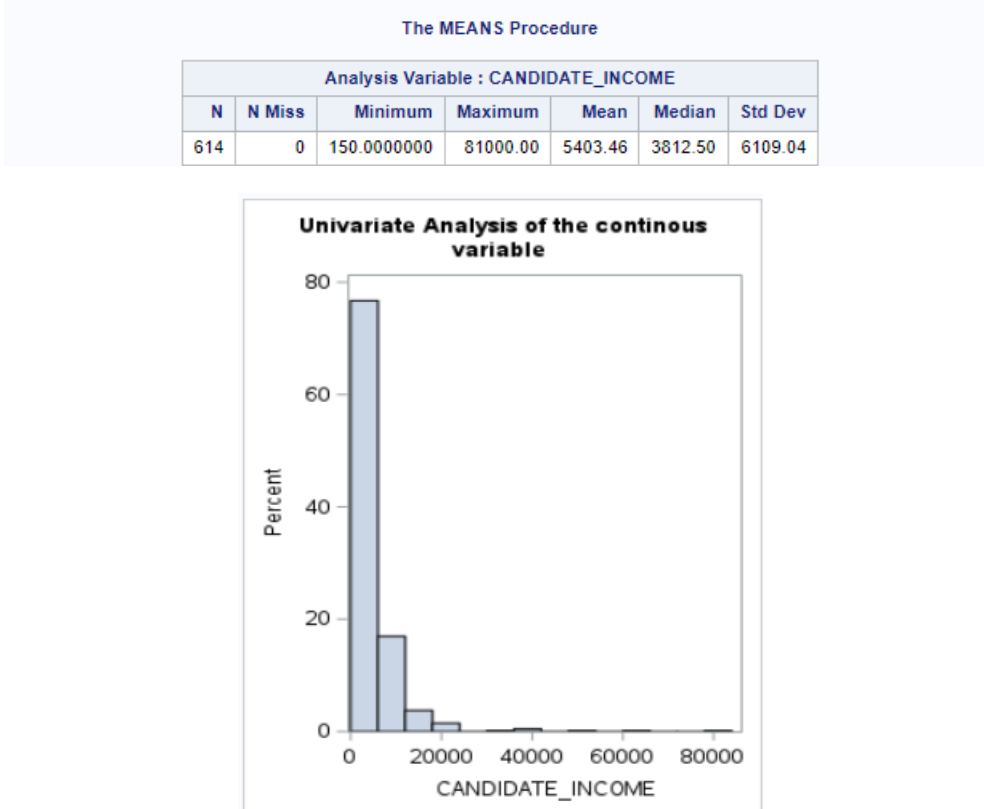
```

151 TITLE 'Figure no - Univariate Analysis of the continous/numeric variable: CANDIDATE_INCOME';
152
153 PROC MEANS DATA = LIB77702.TRAINING_DS N NMISS MIN MAX MEAN MEDIAN STD;
154
155 VAR candidate_income;
156
157 RUN;
158
159 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
160
161 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
162
163 HISTOGRAM candidate_income;
164
165 TITLE 'Univariate Analysis of the continous variable';
166
167 RUN;

```

6.1.2.3.2 Output

Figure no - Univariate Analysis of the continous/numeric variable: CANDIDATE_INCOME



6.1.2.3.3 Description

The univariate analysis shows a summary of a candidate's income. The average income is \$5,403.46 with a standard deviation of \$6,109.04. The minimum income is \$150, and the maximum income is \$81,000.

6.1.2.4 Univariate Analysis of the continuous variable – GUARENTEE_INCOME

6.1.2.4.1 Code Snippet

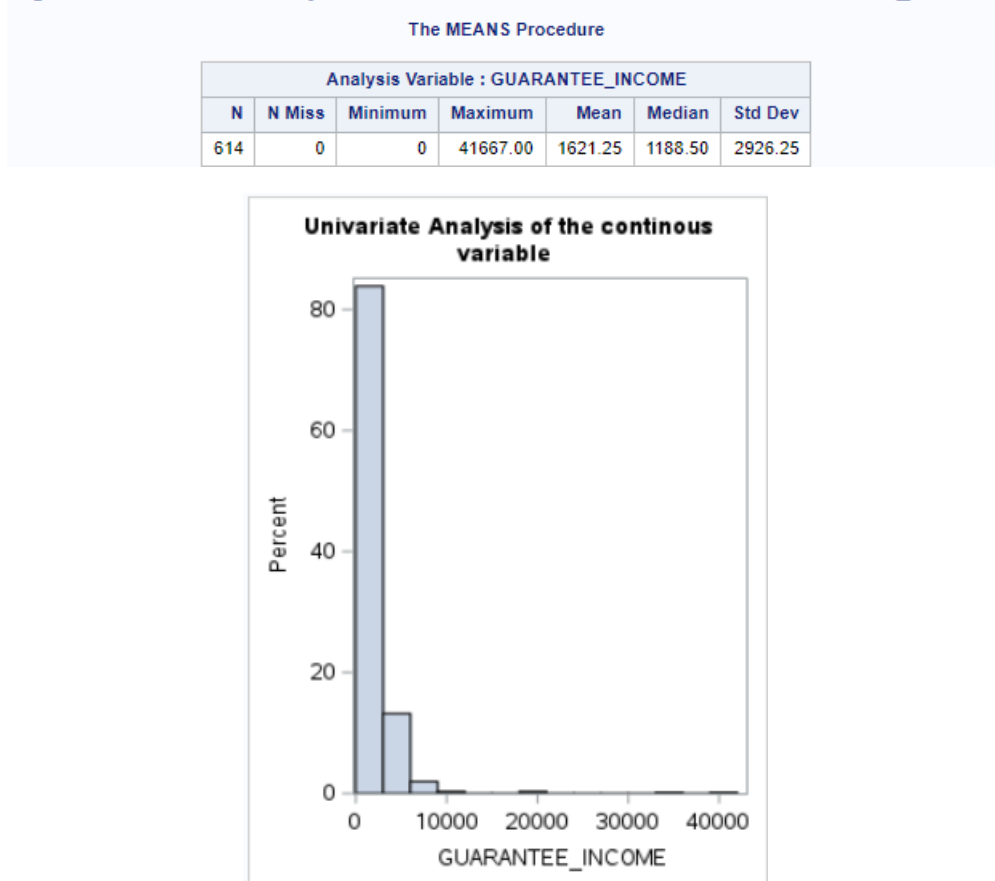
```

169 TITLE 'Figure no - Univariate Analysis of the continous/numeric variable: GUARENTEE_INCOME';
170
171 PROC MEANS DATA = LIB77702.TRAINING_DS N NMISS MIN MAX MEAN MEDIAN STD;
172
173 VAR guarantee_income;
174
175 RUN;
176
177 ODS GRAPHICS / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
178
179 PROC SGPLOT DATA = LIB77702.TRAINING_DS;
180
181 HISTOGRAM guarantee_income;
182
183 TITLE 'Univariate Analysis of the continous variable';
184
185 RUN;

```

6.1.2.4.2 Output

Figure no - Univariate Analysis of the continous/numeric variable: GUARENTEE_INCOME



6.1.2.4.3 Description

The univariate analysis shows a summary of annual variable guaranteed income. The average annual income is \$1,621.25, with a standard deviation of \$2,926.25. The minimum annual income is \$0, and the maximum annual income is \$41,667.

6.2 Bivariate Analysis of Variables in the Dataset LIB77702.TRAINING_DS

6.2.1 Bivariate Analysis of the Variable (Categorical vs Categorical)

6.2.1.1 Bivariate Analysis of the variable (GENDER vs MARITAL_STATUS)

6.2.1.1.1 Code Snippet

```

200 TITLE1 'Bivariate analysis of the variables: ';
201 TITLE2 'Categorical variable[Gender] vs Categorical variable[MARITAL_STATUS]';
202 FOOTNOTE '-----END-----';
203
204 PROC FREQ DATA = LIB77702.TRAINING_DS;
205
206 TABLE gender * marital_status/
207 PLOTS = FREQPLOT( TWOWAY = STACKED SCALE = GROUPPCT );
208
209 RUN;

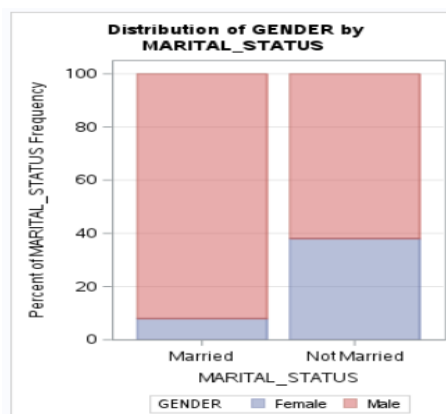
```

6.2.1.1.2 Output

Bivariate analysis of the variables:
Categorical variable[Gender] vs Categorical variable[MARITAL_STATUS]

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of GENDER by MARITAL_STATUS			
	GENDER	MARITAL_STATUS		
		Married	Not Married	Total
	Female	31	80	111
		5.18	13.38	18.56
		27.93	72.07	
		7.99	38.10	
Male	357	130	487	
	59.70	21.74	81.44	
	73.31	26.69		
	92.01	61.90		
Total	388	210	598	
	64.88	35.12	100.00	
	Frequency Missing = 16			



6.2.1.1.3 Description

The bivariate analysis shows a cross-tabulation of gender and marital status. Among females, 27.93% are married and 72.07% are not. Among males, 73.31% are married and 26.69% are not. Overall, 64.88% of the applicants are married, while 35.12% are not. There are 16 missing values.

6.2.1.2 Bivariate Analysis of the variable (GENDER vs LOAN_APPROVAL_STATUS)

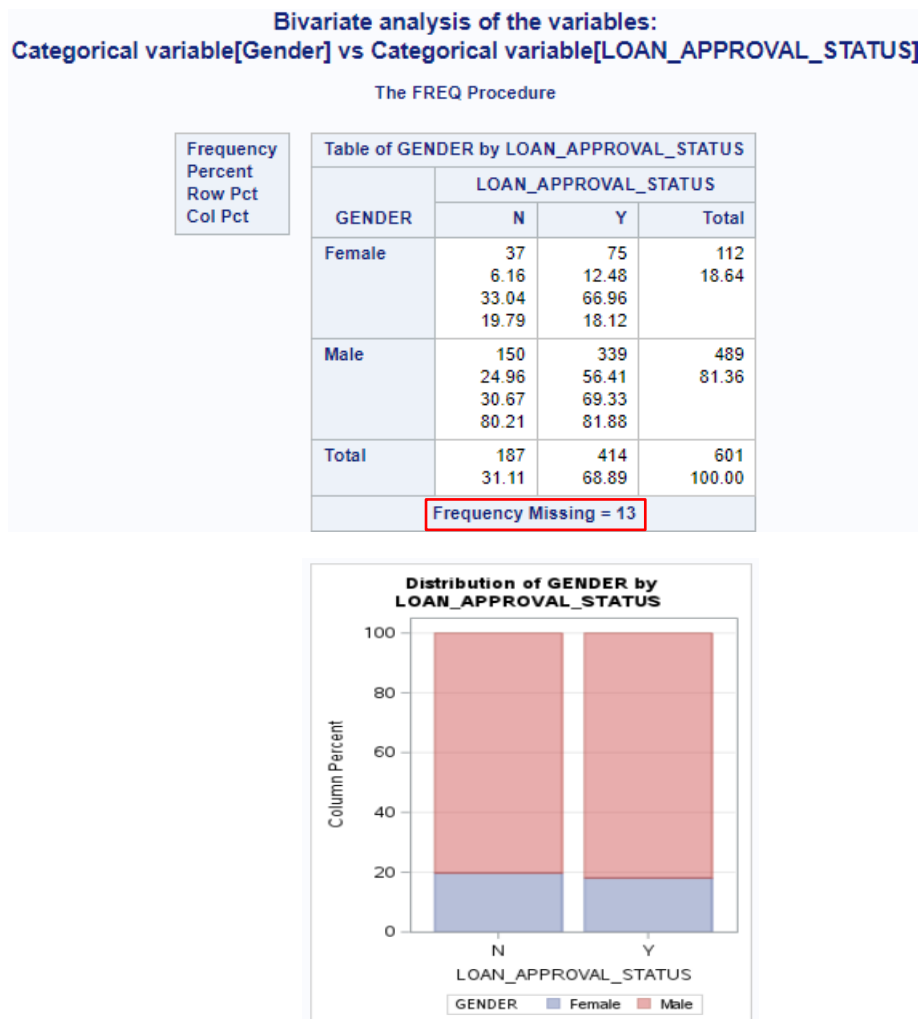
6.2.1.2.1 Code Snippet

```

187 /* SAS Codes to display the bivariate analysis between variables (CATEGORICAL VS CATEGORICAL) */
188
189 TITLE1 'Bivariate analysis of the variables: ';
190 TITLE2 'Categorical variable[Gender] vs Categorical variable[LOAN_APPROVAL_STATUS]';
191 FOOTNOTE '-----END-----';
192
193 PROC FREQ DATA = LIB77702.TRAINING_DS;
194
195 TABLE gender * loan_approval_status/
196 PLOTS = FREQPLOT( TWOWAY = STACKED SCALE = GROUPPCT );
197
198 RUN;

```

6.2.1.2.2 Output



6.2.1.2.3 Description

The bivariate analysis shows a cross-tabulation of gender and loan approval status. Among females, 33.04% of the loan is not approved and 66.96% of the loan is approved. Among males, 30.67% of the loan is not approved and 69.33% are approved. Overall, 31.11% of the applicant's loan are not approved, while 68.89% are approved. There are 13 missing values.

6.2.1.3 Bivariate Analysis of the variable (QUALIFICATION vs EMPLOYMENT)

6.2.1.3.1 Code Snippet

```

211 TITLE1 'Bivariate analysis of the variables: ';
212 TITLE2 'Categorical variable[Qualification] vs Categorical variable[Employment]';
213 FOOTNOTE '-----END-----';
214
215 PROC FREQ DATA = LIB77702.TRAINING_DS;
216
217 TABLE qualification * employment/
218 PLOTS = FREQPLOT( TWOWAY = STACKED SCALE = GROUPPCT );
219
220 RUN;

```

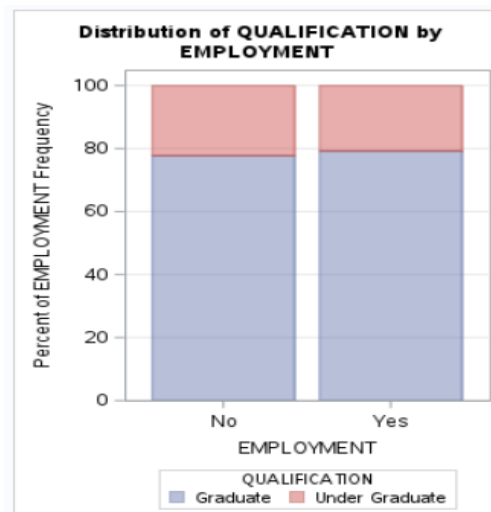
6.2.1.3.2 Output

Bivariate analysis of the variables:
Categorical variable[Qualification] vs Categorical variable[Employment]

The FREQ Procedure

QUALIFICATION	EMPLOYMENT		
	No	Yes	Total
Graduate	389 66.84 85.68 77.80	65 11.17 14.32 79.27	454 78.01
Under Graduate	111 19.07 86.72 22.20	17 2.92 13.28 20.73	128 21.99
Total	500 85.91	82 14.09	582 100.00

Frequency Missing = 32



6.2.1.3.3 Description

The bivariate analysis shows the relationship between qualification and employment status. Among graduates, 85.68% are unemployed and 14.32% are employed. Among undergraduates, 86.72% are unemployed and 13.28% are employed. Overall, 85.91% of applicants are unemployed, and 14.09% are employed. There are 32 missing values.

6.2.2 Bivariate Analysis of the Variable (Categorical vs Continuous)

6.2.2.1 Bivariate Analysis of the variable (GENDER vs LOAN_AMOUNT)

6.2.2.1.1 Code Snippet

```

224 TITLE1 'Bivariate analysis of the variables: ';
225 TITLE2 'Categorical variable[GENDER] vs Continuous variable[LOAN_AMOUNT]';
226 FOOTNOTE '-----END-----';
227
228 PROC MEANS DATA = LIB77702.TRAINING_DS;
229
230 CLASS gender; /* It is a categorical variable */
231 VAR loan_amount; /* It is a numerical variable */
232
233 RUN;

```

6.2.2.1.2 Output

Bivariate analysis of the variables: Categorical variable[GENDER] vs Continuous variable[LOAN_AMOUNT]						
The MEANS Procedure						
Analysis Variable : LOAN_AMOUNT						
GENDER	N Obs	N	Mean	Std Dev	Minimum	Maximum
Female	112	109	126.6972477	79.2864596	9.0000000	600.0000000
Male	489	470	149.2659574	82.8108508	17.0000000	650.0000000

6.2.2.1.3 Description

The bivariate analysis between gender and loan amount reveals that the average loan amount for females is approximately \$126.70 with a standard deviation of \$79.29, ranging from \$9 to \$600. For males, the average is around \$149.27 with a standard deviation of \$82.81, ranging from \$17 to \$650.

6.2.2.2 Bivariate Analysis of the variable (GENDER vs CANDIDATE_INCOME)

6.2.2.2.1 Code Snippet

```

235 TITLE1 'Bivariate analysis of the variables: ';
236 TITLE2 'Categorical variable[GENDER] vs Continuous variable[CANDIDATE_INCOME]';
237 FOOTNOTE '-----END-----';
238
239 PROC MEANS DATA = LIB77702.TRAINING_DS;
240
241 CLASS gender; /* It is a categorical variable */
242 VAR candidate_income; /* It is a numerical variable */
243
244 RUN;

```

6.2.2.2.2 Output

Bivariate analysis of the variables: Categorical variable[GENDER] vs Continuous variable[CANDIDATE_INCOME]						
The MEANS Procedure						
Analysis Variable : CANDIDATE_INCOME						
GENDER	N Obs	N	Mean	Std Dev	Minimum	Maximum
Female	112	112	4643.47	3585.38	210.0000000	19484.00
Male	489	489	5446.46	6185.79	150.0000000	81000.00

6.2.2.2.3 Description

The bivariate analysis between gender and candidate income reveals that the average income for females is approximately \$4643.47 with a standard deviation of \$3585.38, ranging from \$210 to \$19484. For males, the average is around \$5446.46 with a standard deviation of \$6185.79, ranging from \$150 to \$81000.

6.2.2.3 Bivariate Analysis of the variable (GENDER vs GUARANTEE_INCOME)

6.2.2.3.1 Code Snippet

```

246 TITLE1 'Bivariate analysis of the variables: ';
247 TITLE2 'Categorical variable[GENDER] vs Continuous variable[GUARANTEE_INCOME]';
248 FOOTNOTE '-----END-----';
249
250 PROC MEANS DATA = LIB77702.TRAINING_DS;
251
252 CLASS gender; /* It is a categorical variable */
253 VAR guarantee_income; /* It is a numerical variable */
254
255 RUN;

```

6.2.2.3.2 Output

Bivariate analysis of the variables:
Categorical variable[GENDER] vs Continuous variable[GUARANTEE_INCOME]
 The MEANS Procedure

Analysis Variable : GUARANTEE_INCOME						
GENDER	N Obs	N	Mean	Std Dev	Minimum	Maximum
Female	112	112	1108.01	4094.60	0	41667.00
Male	489	489	1742.93	2606.51	0	33837.00

6.2.2.3.3 Description

The bivariate analysis between gender and candidate income reveals that the average income for females is approximately \$1108.01 with a standard deviation of \$4094.60, ranging from \$210 to \$19484. For males, the average is around \$1742.93 with a standard deviation of \$2606.51, ranging from \$0 to \$33837.

6.2.2.4 Bivariate Analysis of the variable (LOAN_APPROVAL_STATUS vs LOAN_AMOUNT)

6.2.2.4.1 Code Snippet

```

257 TITLE1 'Bivariate analysis of the variables: ';
258 TITLE2 'Categorical variable[LOAN_APPROVAL_STATUS] vs Continuous variable[LOAN_AMOUNT]';
259 FOOTNOTE '-----END-----';
260
261 PROC MEANS DATA = LIB77702.TRAINING_DS;
262
263 CLASS loan_approval_status; /* It is a categorical variable */
264 VAR loan_amount; /* It is a numerical variable */
265
266 RUN;

```

6.2.2.4.2 Output

Bivariate analysis of the variables:
Categorical variable[LOAN_APPROVAL_STATUS] vs Continuous variable[LOAN_AMOUNT]
 The MEANS Procedure

Analysis Variable : LOAN_AMOUNT						
LOAN_APPROVAL_STATUS	N Obs	N	Mean	Std Dev	Minimum	Maximum
N	192	181	151.2209945	85.8627833	9.0000000	570.0000000
Y	422	411	144.2944039	85.4846068	17.0000000	700.0000000

6.2.2.4.3 Description

The bivariate analysis between loan approval status and loan amount shows that for approved loans (Y), the average loan amount is approximately 144.29 with a standard deviation of 85.48, ranging from 17 to 700. For unapproved loans (N), the average is higher at 151.22 with a standard deviation of 85.86, ranging from 9 to 570.

7 Analysis of Variables – LIB77702.TESTING_DS

7.1 Introduction to SAS MACRO

The SAS Macro facility is a powerful feature in the SAS programming language. Using the macro, one can automate procedures and clean up code, making it more flexible. Macros take that ability to write fully dynamic programs that drive themselves using data conditions without applying manual interference to the extreme. At the most superficial level, an SAS macro can be defined as a collection of instructions that can be reused as many times as required within any SAS program-opening route to get rid of redundancy and be effective. Macros especially find applications in large-scale data processing and analysis where similar operations need to be applied over several datasets or variables. They dynamically create code according to the values that the macro variables take, hence making the programs more flexible concerning any changes. For example, one can quickly write one macro that cleans up whatever dataset one desires, as opposed to needing to individually write the codes for cleaning each data set. This thereby not only conserves time but also precludes the possibility of committing errors by code duplication. Some features that are associated with the macro language in SAS include the macro variables, macro functions, and macro programs, which allow more compact and maintainable codes (Debikaghosh, 2024).

7.2 Univariate Analysis of the Variables using SAS MACRO – LIB77702.TESTING_DS

7.2.1 Analysis of the Categorical Variables in the Dataset

7.2.1.1 Code Snippet

```

268 /* Macro begins here */
269 OPTIONS MCOMPILENOTE=ALL;
270
271 %MACRO UVA_CAT_VAR(ptitle,pdataset,pcat_var);
272
273 TITLE &ptitle;
274
275 PROC FREQ DATA = &pdataset;
276
277 TABLE &pcat_var;
278
279 RUN;
280
281 %MEND UVA_CAT_VAR;
282 /* Macro ends here */
283
284 /* (TESTING SET) SAS Codes to display the univariate analysis of categorical variables using MACRO */
285
286 /* Call the SAS Macro */
287 %UVA_CAT_VAR('Univariate Analysis of the Categorical Variable - MARITAL_STATUS', LIB77702.TESTING_DS, MARITAL_STATUS);
288
289 %UVA_CAT_VAR('Univariate Analysis of the Categorical Variable - QUALIFICATION', LIB77702.TESTING_DS, QUALIFICATION);
290
291 %UVA_CAT_VAR('Univariate Analysis of the Categorical Variable - LOAN_HISTORY', LIB77702.TESTING_DS, LOAN_HISTORY);
292
293 %UVA_CAT_VAR('Univariate Analysis of the Categorical Variable - GENDER', LIB77702.TESTING_DS, GENDER);
294
295 %UVA_CAT_VAR('Univariate Analysis of the Categorical Variable - FAMILY_MEMBERS', LIB77702.TESTING_DS, FAMILY_MEMBERS);
296
297 %UVA_CAT_VAR('Univariate Analysis of the Categorical Variable - EMPLOYMENT', LIB77702.TESTING_DS, EMPLOYMENT);
298
299 %UVA_CAT_VAR('Univariate Analysis of the Categorical Variable - LOAN_LOCATION', LIB77702.TESTING_DS, LOAN_LOCATION);

```

7.2.1.2 Output

Univariate Analysis of the Categorical Variable - MARITAL_STATUS

The FREQ Procedure

MARITAL_STATUS	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Married	233	63.49	233	63.49
Not Married	134	36.51	367	100.00

Univariate Analysis of the Categorical Variable - QUALIFICATION

The FREQ Procedure

QUALIFICATION	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Graduate	283	77.11	283	77.11
Under Graduate	84	22.89	367	100.00

Univariate Analysis of the Categorical Variable - LOAN_HISTORY

The FREQ Procedure

LOAN_HISTORY	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	59	17.46	59	17.46
1	279	82.54	338	100.00
Frequency Missing = 29				

Univariate Analysis of the Categorical Variable - GENDER

The FREQ Procedure

GENDER	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Female	70	19.66	70	19.66
Male	286	80.34	356	100.00
Frequency Missing = 11				

Univariate Analysis of the Categorical Variable - FAMILY_MEMBERS

The FREQ Procedure

FAMILY_MEMBERS	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	200	56.02	200	56.02
1	58	16.25	258	72.27
2	59	16.53	317	88.80
3+	40	11.20	357	100.00
Frequency Missing = 10				

Univariate Analysis of the Categorical Variable - EMPLOYMENT

The FREQ Procedure

EMPLOYMENT	Frequency	Percent	Cumulative Frequency	Cumulative Percent
No	307	89.24	307	89.24
Yes	37	10.76	344	100.00
Frequency Missing = 23				

Univariate Analysis of the Categorical Variable - LOAN_LOCATION

The FREQ Procedure

LOAN_LOCATION	Frequency	Percent	Cumulative Frequency	Cumulative Percent
City	140	38.15	140	38.15
Town	116	31.61	256	69.75
Village	111	30.25	367	100.00

7.2.1.3 Description

The univariate analysis of categorical variables in testing set reveals several key insights. A majority of the applicants, 63.49%, are married, while 36.51% are not. Regarding qualifications, 77.11% are graduates, and 22.89% are undergraduates. In terms of loan history, 82.54% have a loan history, while 17.46% do not, with 29 missing values. The gender distribution shows that 80.34% of the applicants are male, and 19.66% are female, with 11 missing values. These summaries indicate that the dataset predominantly consists of married graduates, primarily male, who generally have a loan history. The analysis of family member shows that 56.02% of applicants have no family members, 16.25% have one, 16.53% have two, and 11.20% have three or more, with 10 missing values. In employment, 89.24% are unemployed, and 10.76% are employed, with 23 missing values. For loan location, 38.15% are in cities, 31.61% in towns, and 30.25% in villages.

7.2.2 Analysis of the Continuous Variables in the Dataset

7.2.2.1 Code Snippet

```

320 /* Macro begins here */
321 OPTIONS MCOMPILENOTE=ALL;
322
323 %MACRO UVA_NUM_VAR(ptitle,pdataset,pnum_var);
324
325 TITLE &ptitle;
326
327 PROC MEANS DATA = &pdataset N NMISS MIN MAX MEAN MEDIAN STD;
328
329 VAR &pnum_var;
330
331 RUN;
332
333 %MEND UVA_NUM_VAR;
334 /* Macro ends here */

336 /* (TESTING SET) SAS Codes to display the univariate analysis of numerical variables using MACRO */
337
338 /* Call the SAS Macro */
339 %UVA_NUM_VAR('Univariate Analysis of the Numerical Variable - CANDIDATE_INCOME', LIB77702.TESTING_DS, CANDIDATE_INCOME);
340 %UVA_NUM_VAR('Univariate Analysis of the Numerical Variable - GUARANTEE_INCOME', LIB77702.TESTING_DS, GUARANTEE_INCOME);
341 %UVA_NUM_VAR('Univariate Analysis of the Numerical Variable - LOAN_AMOUNT', LIB77702.TESTING_DS, LOAN_AMOUNT);
342 %UVA_NUM_VAR('Univariate Analysis of the Numerical Variable - LOAN_DURATION', LIB77702.TESTING_DS, LOAN_DURATION);

```

7.2.2.2 Output

Univariate Analysis of the Numerical Variable - CANDIDATE_INCOME

The MEANS Procedure

Analysis Variable : CANDIDATE_INCOME						
N	N Miss	Minimum	Maximum	Mean	Median	Std Dev
367	0	0	72529.00	4805.60	3786.00	4910.69

Univariate Analysis of the Numerical Variable - GUARANTEE_INCOME

The MEANS Procedure

Analysis Variable : GUARANTEE_INCOME						
N	N Miss	Minimum	Maximum	Mean	Median	Std Dev
367	0	0	24000.00	1569.58	1025.00	2334.23

Univariate Analysis of the Numerical Variable - LOAN_AMOUNT

The MEANS Procedure

Analysis Variable : LOAN_AMOUNT						
N	N Miss	Minimum	Maximum	Mean	Median	Std Dev
362	5	28.0000000	550.0000000	136.1325967	125.0000000	61.3666524

Univariate Analysis of the Numerical Variable - LOAN_DURATION

The MEANS Procedure

Analysis Variable : LOAN_DURATION						
N	N Miss	Minimum	Maximum	Mean	Median	Std Dev
361	6	6.0000000	480.0000000	342.5373961	360.0000000	65.1566434

7.2.2.3 Description

The univariate analysis of the testing set provides a detailed summary of various variables. For candidate's income, the average is \$4,805.60 with a standard deviation of \$4,910.69, ranging from \$0 to \$72,529. In terms of guarantee income, the average is \$1,569.58 with a standard deviation of \$2,334.23, and it ranges from \$0 to \$24,000. The loan amount analysis shows an average of \$136.13 with a standard deviation of \$61.36. This analysis includes 362 values, with amounts ranging from \$28 to \$550, and 5 missing values. For loan duration, the average is 342.53 days, with a median of 360 days, ranging from 6 to 480 days. This analysis includes 361 values, with 6 missing values.

7.3 Bivariate Analysis of the variable Using SAS MACRO – LIB7702.TESTING_DS

7.3.1 Bivariate Analysis of the variable (Categorical vs Categorical)

7.3.1.1 Bivariate Analysis of the variable (MARITAL_STATUS vs EMPLOYMENT)

7.3.1.1.1 Code Snippet

```

354 /* Macro begins here */
355 OPTIONS MCOMPILENOTE=ALL;
356 %MACRO BVA_CAT_CAT(ptitle1,ptitle2,pdataset,pcat_var1,pcat_var2);
357
358 TITLE1 &ptitle1;
359 TITLE2 &ptitle2;
360
361 PROC FREQ DATA = &pdataset;
362
363 TABLE &pcat_var1 * &pcat_var2/
364 PLOTS = FREQPLOT( TWOWAY = STACKED SCALE = GROUPPCT );
365
366 RUN;
367 %MEND BVA_CAT_CAT;
368 /* Macro ends here */
370 /* Call the SAS Macro */
371 %BVA_CAT_CAT('Bivariate Analysis of the Variables:',
372 'MARITAL_STATUS vs EMPLOYMENT',
373 LIB7702.TESTING_DS,
374 MARITAL_STATUS,EMPLOYMENT);

```

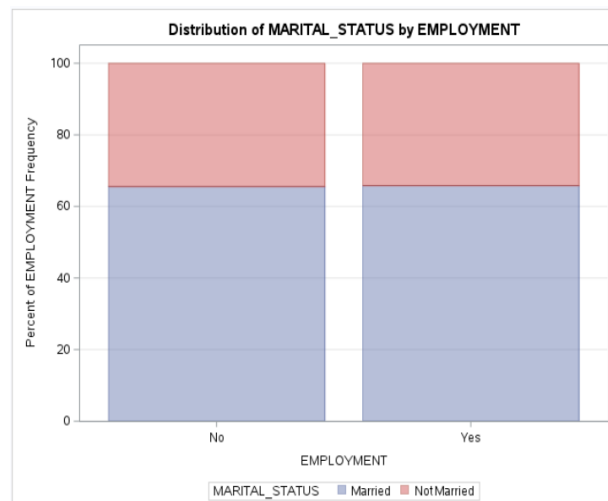
7.3.1.1.2 Output

**Bivariate Analysis of the Variables:
MARITAL_STATUS vs EMPLOYMENT**

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of MARITAL_STATUS by EMPLOYMENT		
	EMPLOYMENT		
MARITAL_STATUS	No	Yes	Total
Married	326 56.30 85.79 65.59	54 9.33 14.21 65.85	380 65.63
Not Married	171 29.53 85.93 34.41	28 4.84 14.07 34.15	199 34.37
Total	497 85.84	82 14.16	579 100.00

Frequency Missing = 35



7.3.1.1.3 Description

The bivariate analysis of marital status versus employment shows that among the 579 individuals, 65.63% are married, with 14.21% of them employed. Among the 34.37% who are not married, 14.07% are employed. Overall, 14.16% of the total population is employed, with 35 missing values.

7.3.1.2 Bivariate Analysis of the variable (GENDER vs EMPLOYEMENT)

7.3.1.2.1 Code Snippet

```

354 /* Macro begins here */
355 OPTIONS MCOMPILENOTE=ALL;
356 %MACRO BVA_CAT_CAT(ptitle1,ptitle2,pdataset,pcat_var1,pcat_var2);
357
358 TITLE1 &ptitle1;
359 TITLE2 &ptitle2;
360
361 PROC FREQ DATA = &pdataset;
362
363 TABLE &pcat_var1 * &pcat_var2/
364 PLOTS = FREQPLOT( TWOWAY = STACKED SCALE = GROUPPCT );
365
366 RUN;
367 %MEND BVA_CAT_CAT;
368 /* Macro ends here */
376 /* Call the SAS Macro */
377 %BVA_CAT_CAT('Bivariate Analysis of the Variables:',
378 'GENDER vs QUALIFICATION',
379 LIB77702.TRAINING_DS,
380 GENDER,QUALIFICATION);

```

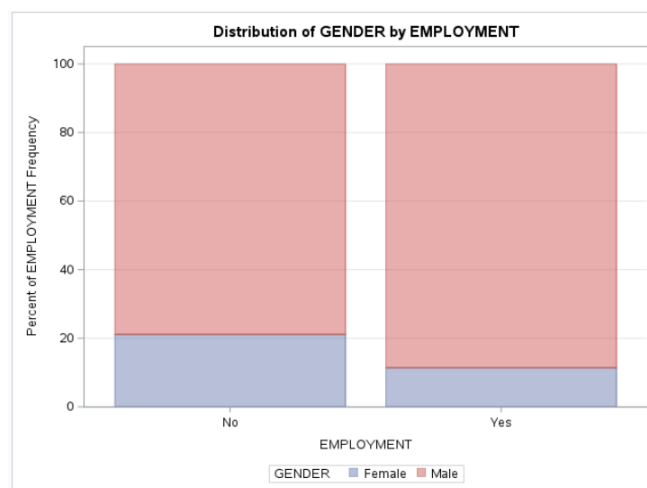
7.3.1.2.2 Output

**Bivariate Analysis of the Variables:
GENDER vs EMPLOYMENT**

The FREQ Procedure

GENDER	EMPLOYMENT		
	No	Yes	Total
Female	63 18.92 94.03 21.14	4 1.20 5.97 11.43	67 20.12
Male	235 70.57 88.35 78.86	31 9.31 11.65 88.57	266 79.88
Total	298 89.49	35 10.51	333 100.00

Frequency Missing = 34



7.3.1.2.3 Description

The bivariate analysis of gender versus employment reveals that among 333 individuals, 20.12% are female, with 5.97% employed and 94.03% are unemployed. Males comprise 79.88% of the data, with 11.65% employed and 88.35% are unemployed. Overall, 10.51% of the total population is employed and 89.49% are unemployed, with 34 missing values.

7.3.1.3 Bivariate Analysis of the variable (QUALIFICATION vs EMPLOYMENT)

7.3.1.3.1 Code Snippet

```

354 /* Macro begins here */
355 OPTIONS MCOMPILENOTE=ALL;
356 %MACRO BVA_CAT_CAT(p1title1,p1title2,pdataset,pcat_var1,pcat_var2);
357
358 TITLE1 &p1title1;
359 TITLE2 &p1title2;
360
361 PROC FREQ DATA = &pdataset;
362
363 TABLE &pcat_var1 * &pcat_var2/
364 PLOTS = FREQPLOT( TWOWAY = STACKED SCALE = GROUPPCT );
365
366 RUN;
367 %MEND BVA_CAT_CAT;
368 /* Macro ends here */
382 /* Call the SAS Macro */
383 %BVA_CAT_CAT('Bivariate Analysis of the Variables:',
384 'QUALIFICATION vs EMPLOYMENT',
385 LIB77702.TESTING_DS,
386 QUALIFICATION,EMPLOYMENT);

```

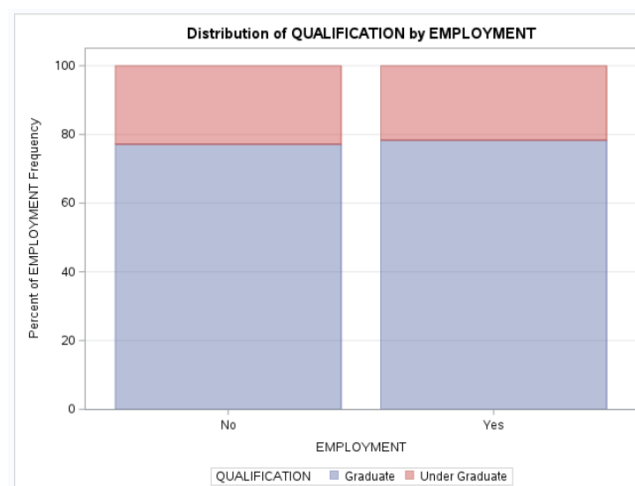
7.3.1.3.2 Output

**Bivariate Analysis of the Variables:
QUALIFICATION vs EMPLOYMENT**

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of QUALIFICATION by EMPLOYMENT		
	EMPLOYMENT		
QUALIFICATION	No	Yes	Total
Graduate	237 68.90 89.10 77.20	29 8.43 10.90 78.38	266 77.33
Under Graduate	70 20.35 89.74 22.80	8 2.33 10.26 21.62	78 22.67
Total	307 89.24	37 10.76	344 100.00

Frequency Missing = 23



7.3.1.3.3 Description

The bivariate analysis of qualification versus employment reveals that among 344 individuals, 77.33% are graduate, with 10.9% employed and 89.1% are unemployed. Undergraduates comprise 22.67% of the data, with 10.26% employed and 89.74% are unemployed. Overall, 10.76% of the total population is employed and 89.24% are unemployed, with 23 missing values.

7.3.2 Bivariate Analysis of the variable (Categorical vs Continuous)

7.3.2.1 Bivariate Analysis of the variable (MARITAL_STATUS vs LOAN_AMOUNT)

7.3.2.1.1 Code Snippet

```

390 /* SAS Codes to display the bivariate analysis of variables (CATEGORICAL VS CONTINOUS) using MACRO */
391
392 /* Macro begins here */
393 OPTIONS MCOMPILENOTE=ALL;
394 %MACRO BVA_CAT_NUM(ptitle1,ptitle2,pdataset,pcat_var,pnum_var);
395
396 TITLE1 &ptitle1;
397 TITLE2 &ptitle2;
398
399 PROC MEANS DATA = &pdataset;
400
401 CLASS &pcat_var; /* It is a categorical variable */
402 VAR &pnum_var; /* It is a continous variable */
403
404 RUN;
405
406 %MEND BVA_CAT_NUM;
407 /* Macro ends here */

```

```

409 /* Call the SAS Macro */
410 %BVA_CAT_NUM('Bivariate Analysis of the variables(Categorical vs Continous)',
411 'MARITAL_STATUS VS LOAN_AMOUNT',
412 LIB77702.TESTING_DS,
413 MARITAL_STATUS,
414 LOAN_AMOUNT);

```

7.3.2.1.2 Output

Bivariate Analysis of the variables(Categorical vs Continous)						
MARITAL_STATUS VS LOAN_AMOUNT						
The MEANS Procedure						
Analysis Variable : LOAN_AMOUNT						
MARITAL_STATUS	N Obs	N	Mean	Std Dev	Minimum	Maximum
Married	233	228	144.6754386	67.7425153	28.0000000	550.0000000
Not Married	134	134	121.5970149	45.2903946	28.0000000	300.0000000

7.3.2.1.3 Description

The bivariate analysis between marital status and loan amount reveals that the average loan amount for married is approximately \$144.7 with a standard deviation of \$67.74, ranging from \$28 to \$550. For Not married, the average is around \$121.6 with a standard deviation of \$45.3, ranging from \$28 to \$300.

7.3.2.2 Bivariate Analysis of the variable (GENDER vs CANDIDATE_INCOME)

7.3.2.2.1 Code Snippet

```

390 /* SAS Codes to display the bivariate analysis of variables (CATEGORICAL VS CONTINUOUS) using MACRO */
391
392 /* Macro begins here */
393 OPTIONS MCOMPILERNOTE=ALL;
394 %MACRO BVA_CAT_NUM(ptitle1,ptitle2,pdataset,pcat_var,pnum_var);
395
396 TITLE1 &ptitle1;
397 TITLE2 &ptitle2;
398
399 PROC MEANS DATA = &pdataset;
400
401 CLASS &pcat_var; /* It is a categorical variable */
402 VAR &pnum_var; /* It is a continous variable */
403
404 RUN;
405
406 %MEND BVA_CAT_NUM;
407 /* Macro ends here */
416 /* Call the SAS Macro */
417 %BVA_CAT_NUM('Bivariate Analysis of the variables(Categorical vs Continous)',
418 'GENDER VS CANDIDATE_INCOME',
419 LIB77702.TESTING_DS,
420 GENDER,
421 CANDIDATE_INCOME);

```

7.3.2.2.2 Output

Bivariate Analysis of the variables(Categorical vs Continous)
GENDER VS CANDIDATE_INCOME

The MEANS Procedure

Analysis Variable : CANDIDATE_INCOME						
GENDER	N Obs	N	Mean	Std Dev	Minimum	Maximum
Female	70	70	4163.60	2644.87	0	14987.00
Male	286	286	4932.86	5186.56	0	72529.00

7.3.2.2.3 Description

The bivariate analysis between gender and candidate income reveals that the average income for females is approximately \$4163.60 with a standard deviation of \$2644.87, ranging from \$0 to \$14987. For males, the average is around \$4932 with a standard deviation of \$72529, ranging from \$0 to \$72529.

7.3.2.3 Bivariate Analysis of the variable (GENDER vs LOAN_AMOUNT)

7.3.2.3.1 Code Snippet

```

390 /* SAS Codes to display the bivariate analysis of variables (CATEGORICAL VS CONTINUOUS) using MACRO */
391
392 /* Macro begins here */
393 OPTIONS MCOMPILENOTE=ALL;
394 %MACRO BVA_CAT_NUM(ptitle1,ptitle2,pdataset,pcat_var,pnum_var);
395
396 TITLE1 &ptitle1;
397 TITLE2 &ptitle2;
398
399 PROC MEANS DATA = &pdataset;
400
401 CLASS &pcat_var; /* It is a categorical variable */
402 VAR &pnum_var; /* It is a continuous variable */
403
404 RUN;
405
406 %MEND BVA_CAT_NUM;
407 /* Macro ends here */

```

```

423 /* Call the SAS Macro */
424 %BVA_CAT_NUM('Bivariate Analysis of the variables(Categorical vs Continous)',
425 'GENDER VS LOAN_AMOUNT',
426 LIB77702.TESTING_DS,
427 GENDER,
428 LOAN_AMOUNT);

```

7.3.2.3.2 Output

Bivariate Analysis of the variables(Categorical vs Continous)						
GENDER VS LOAN_AMOUNT						
The MEANS Procedure						
Analysis Variable : LOAN_AMOUNT						
GENDER	N Obs	N	Mean	Std Dev	Minimum	Maximum
Female	70	69	126.7971014	61.1698647	28.0000000	460.0000000
Male	286	282	139.0319149	62.1310953	28.0000000	550.0000000

7.3.2.3.3 Description

The bivariate analysis between gender and loan amount reveals that the average loan amount for females is approximately \$126.80 with a standard deviation of \$61.2, ranging from \$28 to \$460. For males, the average is around \$139.03 with a standard deviation of \$62.1, ranging from \$28 to \$550.

8 Imputation of Missing Values

8.1 Imputing the Missing Values in Categorical Variables – LIB7702.TRAINING_DS

8.1.1 Handling Missing Values in MARITAL_STATUS

8.1.1.1 Find Missing Value – MARITAL_STATUS

8.1.1.1.1 Code Snippet

```

430 /* STEP 1: Find the details of the loan applicant who submitted their loan
431 application without marital status */
432
433 TITLE1 'Find the details of the loan applicant who submitted their';
434 TITLE2 'Loan application without marital status';
435 PROC SQL;
436
437 SELECT *
438 FROM LIB7702.TRAINING_DS e
439 WHERE ( e.marital_status eq '' OR e.marital_status IS MISSING );
440
441 QUIT;

```

8.1.1.1.2 Output

Find the details of the loan applicant who submitted their Loan application without marital status												
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY	LOAN_LOCATION	LOAN_APPROVAL_STATUS
LP001357	Male			Graduate	No	3818	754	160	360	1	City	Y
LP001780	Male			Graduate	No	4768	0	158	480	1	Town	Y
LP002393	Female			Graduate	No	10047	0	.	240	1	Town	Y

8.1.1.1.3 Description

From the output above, we observe that the training set contains missing values for marital status; these values will be imputed later.

8.1.1.2 Counting Number of Missing Values – MARITAL_STATUS

8.1.1.2.1 Code Snippet

```

443 /* STEP 2: Count the number of the loan applicant who submitted their loan
444 application without marital status */
445
446 TITLE1 'Count the no.of loan applicant who submitted thier';
447 TITLE2 'Loan application without marital status';
448 PROC SQL;
449
450 SELECT COUNT(*) Label = 'Number of Loan Applicants'
451 FROM LIB7702.TRAINING_DS e
452 WHERE ( e.marital_status eq '' OR e.marital_status IS MISSING );
453
454 QUIT;

```

8.1.1.2.2 Output

Count the no.of loan applicant who submitted thier Loan application without marital status	
Number of Loan Applicants	3

8.1.1.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their marital status, which is 3 loan applicants.

8.1.1.3 Finding the Statistics – MARITAL_STATUS

8.1.1.3.1 Code Snippet

```

456 /* STEP 3: Find the statistics for marital status */
457
458 TITLE1 'Find the statistics for marital status';
459
460 PROC SQL;
461
462 SELECT e.marital_status AS MARITAL_STATUS,
463        COUNT(*) AS COUNTS
464 FROM LIB77702.TRAINING_DS e
465 WHERE ( e.marital_status NE '' OR e.marital_status IS NOT MISSING )
466 GROUP BY e.marital_status;
467
468 QUIT;

```

8.1.1.3.2 Output

Find the statistics for marital status

MARITAL_STATUS	COUNTS
Married	398
Not Married	213

8.1.1.3.3 Description

The statistics for marital status show that out of the total applicants, 398 are married and 213 are unmarried. This indicates that most of the applicants are married, while rest are not married.

8.1.1.4 Saving the Statistics – MARITAL_STATUS

8.1.1.4.1 Code Snippet

```



470 /* STEP 4: Save the statistics for marital status in a dataset */
471
472 TITLE1 'Save the statistics for marital status in a dataset';
473
474 PROC SQL;
475
476 CREATE TABLE LIB77702.TRAINING_STATS_DS AS
477 SELECT e.marital_status AS MARITAL_STATUS,
478        COUNT(*) AS COUNTS
479 FROM LIB77702.TRAINING_DS e
480 WHERE ( e.marital_status NE '' OR e.marital_status IS NOT MISSING )
481 GROUP BY e.marital_status;
482
483 QUIT;

```

8.1.1.4.2 Output

Table: | View:

Columns ⓘ Total rows: 2 Total columns: 2

<input checked="" type="checkbox"/>	Select all
<input checked="" type="checkbox"/>	 MARITAL_STATUS
<input checked="" type="checkbox"/>	 COUNTS

MARITAL_STAT...	COUNTS
1 Married	398
2 Not Married	213

8.1.1.4.3 Description

The above output shows that the statistics performed earlier is saved successfully in TRAINING_STATS_DS.

8.1.1.5 Backup of the Dataset – MARITAL_STATUS

8.1.1.5.1 Code Snippet

```

485 /* STEP 4.1: Make a backup of the dataset */
486
487 PROC SQL;
488
489 CREATE TABLE LIB77702.TRAINING_BACKUP_DS AS
490 SELECT *
491 FROM LIB77702.TRAINING_DS;
492
493 QUIT;

```

8.1.1.5.2 Output

Table: View: Filter: (none)

Total rows: 614 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001002	Male	Not Married	0	Graduate	No	5849	0
2	LP001003	Male	Married	1	Graduate	No	4583	1508
3	LP001005	Male	Married	0	Graduate	Yes	3000	0
4	LP001006	Male	Married	0	Under Graduate	No	2583	2358
5	LP001008	Male	Not Married	0	Graduate	No	6000	0
6	LP001011	Male	Married	2	Graduate	Yes	5417	4196
7	LP001013	Male	Married	0	Under Graduate	No	2333	1516
8	LP001014	Male	Married	3+	Graduate	No	3036	2504
9	LP001018	Male	Married	2	Graduate	No	4006	1526
10	LP001020	Male	Married	1	Graduate	No	12841	10968

8.1.1.5.3 Description

The newly formed backup table LIB77702.TRAINING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.1.1.6 Imputing the missing values– MARITAL_STATUS

8.1.1.6.1 Code Snippet

```

495 /* STEP 5: Impute the missing values in the categorical variable (MARITAL_STATUS) */
496
497 PROC SQL;
498
499 UPDATE LIB77702.TRAINING_DS
500 SET marital_status = ( SELECT marital_status AS marital_status
501                       FROM LIB77702.TRAINING_STATS_DS t2
502                       WHERE t2.counts eq ( SELECT MAX(t1.counts) AS HIGHEST_COUNT
503                                           FROM LIB77702.TRAINING_STATS_DS t1 ) )
504                       /* Above is sub-program to find the highest count */
505 WHERE ( marital_status eq '' OR marital_status IS MISSING );
506
507 QUIT;

```

8.1.1.6.2 Output

NOTE: 3 rows were updated in LIB77702.TRAINING_DS.

8.1.1.6.3 Description

The above output shows that the 3 missing values were successfully imputed and updated in marital status.

8.1.1.7 After imputing the missing values – MARITAL_STATUS

8.1.1.7.1 Code Snippet

```

509 /* STEP 6: (AFTER IMPUTATION) Find missing values for marital status */
510
511 TITLE1 '(AFTER IMPUTATION) Find missing values for marital status';
512 TITLE2 'Loan application without marital status';
513 FOOTNOTE '-----End-----';
514
515 PROC SQL;
516
517 SELECT *
518 FROM LIB77702.TRAINING_DS e
519 WHERE ( e.marital_status eq '' OR e.marital_status IS MISSING );
520
521 QUIT;

```

8.1.1.7.2 Output

```

(AFTER IMPUTATION) Find missing values for marital status
Loan application without marital status

-----End-----

```

8.1.1.7.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.1.2 Handling Missing Values in FAMILY_MEMBERS

8.1.2.1 Find missing values – FAMILY_MEMBERS

8.1.2.1.1 Code Snippet

```

527 /* STEP 1: List the details of the loan applicants who submitted the applications
528 without family member details */
529 PROC SQL;
530
531 SELECT *
532 FROM LIB77702.TRAINING_DS e
533 WHERE ( e.family_members eq '' OR e.family_members IS NULL );
534
535 QUIT;

```

8.1.2.1.2 Output

SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY	LOAN_LOCATION	LOAN_APPROVAL_STATUS
LP001350	Male	Married		Graduate	No	13550	0	-	350	1	City	Y
LP001357	Male	Married		Graduate	No	3515	754	150	350	1	City	Y
LP001425	Male	Married		Graduate	No	5557	2557	150	350	1	Village	Y
LP001754	Male	Married		Under Graduate	Yes	4735	0	135	350	1	City	N
LP001750	Male	Married		Graduate	No	4755	0	155	450	1	Town	Y
LP001945	Female	Not Married		Graduate	No	5417	0	143	450	0	City	N
LP001972	Male	Married		Under Graduate	No	2575	1750	105	350	1	Town	Y
LP002100	Male	Not Married		Graduate	No	2533	0	71	350	1	City	Y
LP002105	Male	Married		Graduate	Yes	5503	4450	70	-	1	Town	Y
LP002130	Male	Married		Under Graduate	No	3523	3230	152	350	0	Village	N
LP002144	Female	Not Married		Graduate	No	3513	0	115	150	1	City	Y
LP002353	Female	Married		Graduate	No	10047	0	-	240	1	Town	Y
LP002552	Male	Married		Under Graduate	No	3074	1500	123	350	0	Town	N
LP002547	Male	Married		Graduate	No	5115	1451	105	350	0	City	N
LP002543	Male	Not Married		Graduate	No	2557	0	55	350	0	Town	N

8.1.2.1.3 Description

From the output above, we observe that the training set contains missing values for family members; these values will be imputed later.

8.1.2.2 Counting the Number of Missing values – FAMILY_MEMBERS

8.1.2.2.1 Code Snippet

```

537 /* STEP 2: Count the number of the loan applicants who submitted the applications
538 without family member details */
539 PROC SQL;
540
541 SELECT COUNT (*) Label = 'Number of Loan Applicants'
542 FROM LIB77702.TRAINING_DS e
543 WHERE ( e.family_members eq '' OR e.family_members IS NULL );
544
545 QUIT;

```

8.1.2.2.2 Output

Number of Loan Applicants
15

8.1.2.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their family members, which is 15 loan applicants.

8.1.2.3 List Details of Family with '3+' Members

8.1.2.3.1 Code Snippet

```

547 /* STEP 3: List the details of the loan applicants with '3+' family members who submitted the applications
548 without family member details */
549 PROC SQL;
550
551 SELECT e.family_members Label = 'Family Members',
552        SUBSTR(e.family_members,1,1) Label = 'The data found in the 1st position',
553        SUBSTR(e.family_members,2,1) Label = 'The data found in the 2nd position'
554 FROM LIB77702.TRAINING_DS e
555 WHERE ( e.family_members ne '' OR e.family_members IS NOT MISSING );
556
557 QUIT;

```

8.1.2.3.2 Output

Family Members	The data found in the 1st position	The data found in the 2nd position
0	0	
1	1	
0	0	
0	0	
0	0	
2	2	
0	0	
3+	3	+
2	2	

8.1.2.3.3 Description

From the above output we can see that applicant with 3+ family member is split into two positions to remove '+' sign in the later stage.

8.1.2.4 Backup of Dataset – FAMILY_MEMBERS

8.1.2.4.1 Code Snippet

```

559 /* STEP 4: Make a backup of the dataset */
560
561 PROC SQL;
562
563 CREATE TABLE LIB77702.TRAINING_BACKUP_DS AS
564 SELECT *
565 FROM LIB77702.TRAINING_DS;
566
567 QUIT;

```

8.1.2.4.2 Output

Table: TEMP0.TRAINING_BACKUP_DS | View: Column names | Filter: (none)

Total rows: 614 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001002	Male	Not Married	0	Graduate	No	5849	0
2	LP001003	Male	Married	1	Graduate	No	4583	1508
3	LP001005	Male	Married	0	Graduate	Yes	3000	0
4	LP001006	Male	Married	0	Under Graduate	No	2583	2358
5	LP001008	Male	Not Married	0	Graduate	No	6000	0
6	LP001011	Male	Married	2	Graduate	Yes	5417	4196
7	LP001013	Male	Married	0	Under Graduate	No	2333	1516
8	LP001014	Male	Married	3+	Graduate	No	3036	2504
9	LP001018	Male	Married	2	Graduate	No	4006	1526
10	LP001020	Male	Married	1	Graduate	No	12841	10968

8.1.2.4.3 Description

The newly formed backup table LIB77702.TRAINING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.1.2.5 Removing '+' Found in Family Member

8.1.2.5.1 Code Snippet

```

569 /* STEP 5: Remove the '+' found in the family_members variable */
570 PROC SQL;
571
572 UPDATE LIB77702.TRAINING_DS
573 SET family_members = SUBSTR(family_members,1,1)
574 WHERE SUBSTR(family_members,2,1) eq '+';
575
576 QUIT;

```

8.1.2.5.2 Output

NOTE: 51 rows were updated in LIB77702.TRAINING_DS.

8.1.2.5.3 Description

From the above output we can see that the '+' sign has been removed successfully and the rows have been updated in family members.

8.1.2.6 Finding the Statistics – FAMILY_MEMBERS

8.1.2.6.1 Code Snippet

```

578 /* STEP 6: Display the statistics in a dataset */
579
580 PROC SQL;
581
582 SELECT e.family_members AS family_members,
583 COUNT (*) AS COUNTS
584 FROM LIB77702.TRAINING_DS e
585 WHERE ( e.family_members ne '' or e.family_members IS NOT MISSING )
586 GROUP BY e.family_members;
587
588 QUIT;

```

8.1.2.6.2 Output

family_members	COUNTS
0	345
1	102
2	101
3	51

8.1.2.6.3 Description

The statistics for family members show that out of the total applicants, 345 are having 0 family members, 102 having 1 member, 101 having 2 members and 51 having 3 members. This indicates that most of the applicants are having no family members.

8.1.2.7 Save the Statistics – FAMILY_MEMBERS

8.1.2.7.1 Code Snippet

```

590 /* STEP 7: Save the statistics in a dataset */
591
592 PROC SQL;
593
594 CREATE TABLE LIB77702.TRAINING_STAT_FM AS
595 SELECT e.family_members AS family_members,
596 COUNT (*) AS counts
597 FROM LIB77702.TRAINING_DS e
598 WHERE ( e.family_members ne '' or e.family_members IS NOT MISSING )
599 GROUP BY e.family_members;
600
601 QUIT;

```

8.1.2.7.2 Output

Table: **_TEMP0.TRAINING_STAT_FM** | View: **Column names** |

Columns

- ☒ Select all
- ☒ family_members
- ☒ counts

Total rows: 4 Total columns: 2

	family_members	counts
1	0	345
2	1	102
3	2	101
4	3	51

8.1.2.7.3 Description

The above output shows that the statistics performed earlier is saved successfully in TRAINING_STATS_DS.

8.1.2.8 Imputing the Missing Value – FAMILY_MEMBERS

8.1.2.8.1 Code Snippet

```

603 /* STEP 8: Impute the missing values found in the Categorical Variable - FAMILY_MEMBERS */
604
605 PROC SQL;
606
607 UPDATE LIB77702.TRAINING_DS
608 SET family_members = ( SELECT to.family_members AS family_members
609                        FROM LIB77702.TRAINING_STAT_FM to
610                        WHERE to.counts eq ( SELECT MAX (ti.counts) AS highest_count
611                                           FROM LIB77702.TRAINING_STAT_FM ti ) )
612                        /* Above is a sub-program to find the highest count */
613 WHERE ( family_members eq '' OR family_members IS NULL );
614
615 QUIT;

```

8.1.2.8.2 Output

NOTE: 15 rows were updated in LIB77702.TRAINING_DS.

8.1.2.8.3 Description

The above output shows that the 15 missing values rows were successfully imputed and updated in family members.

8.1.2.9 After Imputing Missing Value – FAMILY_MEMBERS

8.1.2.9.1 Code Snippet

```

627 /* STEP 10: (After Imputation) Count the number of the loan applicants who submitted the
628 applications without family member details */
629 PROC SQL;
630
631 SELECT COUNT (*) Label = 'Number of Loan Applicants'
632 FROM LIB77702.TRAINING_DS e
633 WHERE ( e.family_members eq '' OR e.family_members IS NULL );
634
635 QUIT;

```

8.1.2.9.2 Output

Number of Loan Applicants
0

8.1.2.9.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.1.3 Handling Missing Values in EMPLOYMENT

8.1.3.1 Find Missing Value – EMPLOYMENT

8.1.3.1.1 Code Snippet

```
640 /* STEP 1: Find missing values for employment */
641
642 TITLE1 'Find missing values for employment';
643 TITLE2 'Loan application without employment';
644 PROC SQL;
645
646 SELECT *
647 FROM LIB77702.TRAINING_DS e
648 WHERE ( e.employment eq '' OR e.employment IS MISSING );
649
650 QUIT;
```

8.1.3.1.2 Output

Find missing values for employment Loan application without employment										
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY
LP001027	Male	Married	2	Graduate		2500	1840	109	360	1
LP001041	Male	Married	0	Graduate		2600	3500	115	.	1
LP001052	Male	Married	1	Graduate		3717	2925	151	360	.
LP001087	Female	Not Married	2	Graduate		3750	2083	120	360	1
LP001091	Male	Married	1	Graduate		4166	3369	201	360	.
LP001326	Male	Not Married	0	Graduate		6782	0	.	360	.

8.1.3.1.3 Description

From the output above, we observe that the training set contains missing values for employment; these values will be imputed later.

8.1.3.2 Counting Number of Missing Values – EMPLOYMENT

8.1.3.2.1 Code Snippet

```
652 /* STEP 2: Count the number of missing values for employment */
653
654 TITLE1 'Count missing values for employment';
655 TITLE2 'Loan application without employment';
656 PROC SQL;
657
658 SELECT COUNT(*) Label = 'Number of Loan Applicants'
659 FROM LIB77702.TRAINING_DS e
660 WHERE ( e.employment eq '' OR e.employment IS MISSING );
661
662 QUIT;
```

8.1.3.2.2 Output

Count missing values for employment Loan application without employment	
Number of Loan Applicants	32

8.1.3.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their employment status, which is 32 loan applicants.

8.1.3.3 Finding the Statistics – EMPLOYMENT

8.1.3.3.1 Code Snippet

```

664 /* STEP 3: Find the statistics for employment */
665
666 TITLE1 'Find the statistics for employment';
667
668 PROC SQL;
669
670 SELECT e.employment AS EMPLOYMENT,
671        COUNT(*) AS COUNTS
672 FROM LIB77702.TRAINING_DS e
673 WHERE ( e.employment NE '' OR e.employment IS NOT MISSING )
674 GROUP BY e.employment;
675
676 QUIT;

```

8.1.3.3.2 Output

Find the statistics for employment

EMPLOYMENT	COUNTS
No	500
Yes	82

8.1.3.3.3 Description

The statistics for employment show that out of the total applicants, 500 are employed and 82 are not employed. This indicates that most of the applicants are employed, while rest are not employed.

8.1.3.4 Saving the Statistics – EMPLOYMENT




8.1.3.4.1 Code Snippet



```

678 /* STEP 4: Save the statistics for employment in a dataset */
679
680 TITLE1 'Save the statistics for employment in a dataset';
681
682 PROC SQL;
683
684 CREATE TABLE LIB77702.TRAINING_STATS_DS AS
685 SELECT e.employment AS EMPLOYMENT,
686        COUNT(*) AS COUNTS
687 FROM LIB77702.TRAINING_DS e
688 WHERE ( e.employment NE '' OR e.employment IS NOT MISSING )
689 GROUP BY e.employment;
690
691 QUIT;

```

8.1.3.4.2 Output

Table: _TEMP0.TRAINING_STATS_DS | View: Column names |    |

Columns		Total rows: 2 Total columns: 2	
<input checked="" type="checkbox"/>	Select all	EMPLOYM...	COUNTS
<input checked="" type="checkbox"/>	 EMPLOYMENT	1 No	500
<input checked="" type="checkbox"/>	 COUNTS	2 Yes	82

8.1.3.4.3 Description

The above output shows that the statistics performed earlier is saved successfully in TRAINING_STATS_DS.

8.1.3.5 Backup of the Dataset – EMPLOYMENT

8.1.3.5.1 Code Snippet

```

693 /* STEP 4.1: Make a backup of the dataset */
694
695 PROC SQL;
696
697 CREATE TABLE LIB77702.TRAINING_BACKUP_DS AS
698 SELECT *
699 FROM LIB77702.TRAINING_DS;
700
701 QUIT;

```

8.1.3.5.2 Output

Table: View: Filter: (none)

Total rows: 614 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001002	Male	Not Married	0	Graduate	No	5849	0
2	LP001003	Male	Married	1	Graduate	No	4583	1508
3	LP001005	Male	Married	0	Graduate	Yes	3000	0
4	LP001006	Male	Married	0	Under Graduate	No	2583	2358
5	LP001008	Male	Not Married	0	Graduate	No	6000	0
6	LP001011	Male	Married	2	Graduate	Yes	5417	4196
7	LP001013	Male	Married	0	Under Graduate	No	2333	1516
8	LP001014	Male	Married	3	Graduate	No	3036	2504
9	LP001018	Male	Married	2	Graduate	No	4006	1526
10	LP001020	Male	Married	1	Graduate	No	12841	10968

8.1.3.5.3 Description

The newly formed backup table LIB77702.TRAINING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.1.3.6 Imputing the missing values– EMPLOYMENT

8.1.3.6.1 Code Snippet

```

703 /* STEP 5: Impute the missing values in the categorical variable (EMPLOYMENT) */
704
705 PROC SQL;
706
707 UPDATE LIB77702.TRAINING_DS
708 SET employment = ( SELECT employment AS employment
709                     FROM LIB77702.TRAINING_STATS_DS t2
710                     WHERE t2.counts eq ( SELECT MAX(t1.counts) AS HIGHEST_COUNT
711                                           FROM LIB77702.TRAINING_STATS_DS t1 ) )
712                     /* Above is sub-program to find the highest count */
713 WHERE ( employment eq '' OR employment IS MISSING );
714
715 QUIT;

```

8.1.3.6.2 Output

NOTE: 32 rows were updated in LIB77702.TRAINING_DS.

8.1.3.6.3 Description

The above output shows that the 32 missing values rows were successfully imputed and updated in employment.

8.1.3.7 After imputing the missing values – EMPLOYMENT

8.1.3.7.1 Code Snippet

```

717 /* STEP 6: (AFTER IMPUTATION) Find missing values for employment */
718
719 TITLE1 '(AFTER IMPUTATION) Find missing values for employment';
720 TITLE2 'Loan application without employment';
721 FOOTNOTE '-----End-----';
722
723 PROC SQL;
724
725 SELECT *
726 FROM LIB77702.TRAINING_DS e
727 WHERE ( e.employment eq '' OR e.employment IS MISSING );
728
729 QUIT;

```

8.1.3.7.2 Output

(AFTER IMPUTATION) Find missing values for employment
Loan application without employment

-----End-----

8.1.3.7.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.1.4 Handling Missing Values in GENDER

8.1.4.1 Find Missing Value – GENDER

8.1.4.1.1 Code Snippet

```

733 /* STEP 1: Find missing values for gender */
734
735 TITLE1 'Find missing values for gender';
736 TITLE2 'Loan application without gender';
737 PROC SQL;
738
739 SELECT *
740 FROM LIB77702.TRAINING_DS e
741 WHERE ( e.gender eq '' OR e.gender IS MISSING );
742
743 QUIT;

```

8.1.4.1.2 Output

Find missing values for gender Loan application without gender												
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY	LOAN_LOCATION	LOAN_APPROVAL_STATUS
LP001050		Married	2	Under Graduate	No	3365	1917	112	360	0	Village	N
LP001448		Married	3	Graduate	No	23803	0	370	360	1	Village	Y
LP001585		Married	3	Graduate	No	51763	0	700	300	1	City	Y
LP001644		Married	0	Graduate	Yes	674	5266	168	360	1	Village	Y
LP002024		Married	0	Graduate	No	2473	1843	159	360	1	Village	N
LP002103		Married	1	Graduate	Yes	9833	1833	182	180	1	City	Y
LP002478		Married	0	Graduate	Yes	2083	4083	160	360	.	Town	Y

8.1.4.1.3 Description

From the output above, we observe that the training set contains missing values for gender; these values will be imputed later.

8.1.4.2 Counting Number of Missing Values – GENDER

8.1.4.2.1 Code Snippet

```

745 /* STEP 2: Count the number of missing values for gender */
746
747 TITLE1 'Count missing values for gender';
748 TITLE2 'Loan application without gender';
749 PROC SQL;
750
751 SELECT COUNT(*) Label = 'Number of Loan Applicants'
752 FROM LIB77702.TRAINING_DS e
753 WHERE ( e.gender eq '' OR e.gender IS MISSING );
754
755 QUIT;

```

8.1.4.2.2 Output

Count missing values for gender
Loan application without gender

Number of Loan Applicants
13

8.1.4.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their gender, which is 13 loan applicants.

8.1.4.3 Finding the Statistics – GENDER

8.1.4.3.1 Code Snippet

```

757 /* STEP 3: Find the statistics for gender */
758
759 TITLE1 'Find the statistics for gender';
760
761 PROC SQL;
762
763 SELECT e.gender AS GENDER,
764        COUNT(*) AS COUNTS
765 FROM LIB77702.TRAINING_DS e
766 WHERE ( e.gender NE '' OR e.gender IS NOT MISSING )
767 GROUP BY e.gender;
768
769 QUIT;

```

8.1.4.3.2 Output

Find the statistics for gender

GENDER	COUNTS
Female	112
Male	489

8.1.4.3.3 Description

The statistics for gender show that out of the total applicants, 112 are female and 489 are male. This indicates that most of the applicants are male.

8.1.4.4 Saving the Statistics – GENDER

8.1.4.4.1 Code Snippet

```

771 /* STEP 4: Save the statistics for employment in a dataset */
772
773 TITLE1 'Save the statistics for employment in a dataset';
774
775 PROC SQL;
776
777 CREATE TABLE LIB77702.TRAINING_STATS_DS AS
778 SELECT e.gender AS GENDER,
779        COUNT(*) AS COUNTS
780 FROM LIB77702.TRAINING_DS e
781 WHERE ( e.gender NE '' OR e.gender IS NOT MISSING )
782 GROUP BY e.gender;
783
784 QUIT;

```

8.1.4.4.2 Output

Table: _TEMP0.TRAINING_STATS_DS | View: Column names

Columns: ☒ Select all, ☒ GENDER, ☒ COUNTS

Total rows: 2 Total columns: 2

	GENDER	COUNTS
1	Female	112
2	Male	489

8.1.4.4.3 Description

The above output shows that the statistics performed earlier is saved successfully in TRAINING_STATS_DS.

8.1.4.5 Backup of the Dataset – GENDER

8.1.4.5.1 Code Snippet

```

786 /* STEP 4.1: Make a backup of the dataset */
787
788 PROC SQL;
789
790 CREATE TABLE LIB77702.TRAINING_BACKUP_DS AS
791 SELECT *
792 FROM LIB77702.TRAINING_DS;
793
794 QUIT;

```

8.1.4.5.2 Output

Table: _TEMP0.TRAINING_BACKUP_DS | View: Column names | Filter: (none)

Total rows: 614 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001002	Male	Not Married	0	Graduate	No	5849	0
2	LP001003	Male	Married	1	Graduate	No	4583	1508
3	LP001005	Male	Married	0	Graduate	Yes	3000	0
4	LP001006	Male	Married	0	Under Graduate	No	2583	2358
5	LP001008	Male	Not Married	0	Graduate	No	6000	0
6	LP001011	Male	Married	2	Graduate	Yes	5417	4196
7	LP001013	Male	Married	0	Under Graduate	No	2333	1516
8	LP001014	Male	Married	3	Graduate	No	3036	2504
9	LP001018	Male	Married	2	Graduate	No	4006	1526
10	LP001020	Male	Married	1	Graduate	No	12841	10968

8.1.4.5.3 Description

The newly formed backup table LIB77702.TRAINING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.1.4.6 Imputing the missing values– GENDER

8.1.4.6.1 Code Snippet

```
796 /* STEP 5: Impute the missing values in the categorical variable (GENDER) */
797
798 PROC SQL;
799
800 UPDATE LIB77702.TRAINING_DS
801 SET gender = ( SELECT gender AS gender
802                FROM LIB77702.TRAINING_STATS_DS t2
803                WHERE t2.counts eq ( SELECT MAX(t1.counts) AS HIGHEST_COUNT
804                                   FROM LIB77702.TRAINING_STATS_DS t1 ) )
805                /* Above is sub-program to find the highest count */
806 WHERE ( gender eq '' OR gender IS MISSING );
807
808 QUIT;
```

8.1.4.6.2 Output

NOTE: 13 rows were updated in LIB77702.TRAINING_DS.

8.1.4.6.3 Description

The above output shows that the 13 missing values rows were successfully imputed and updated in gender.

8.1.4.7 After imputing the missing values – GENDER

8.1.4.7.1 Code Snippet

```
810 /* STEP 6: (AFTER IMPUTATION) Find missing values for gender */
811
812 TITLE1 '(AFTER IMPUTATION) Find missing values for gender';
813 TITLE2 'Loan application without gender';
814 FOOTNOTE '-----End-----';
815
816 PROC SQL;
817
818 SELECT *
819 FROM LIB77702.TRAINING_DS e
820 WHERE ( e.gender eq '' OR e.gender IS MISSING );
821
822 QUIT;
```

8.1.4.7.2 Output

```
(AFTER IMPUTATION) Find missing values for gender
Loan application without gender

-----End-----
```

8.1.4.7.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.1.5 Handling Missing Values in LOAN_HISTORY

8.1.5.1 Find Missing Value – LOAN_HISTORY

8.1.5.1.1 Code Snippet

```

889 /* STEP 1:List the details of the loan applicants who submitted the applications without loan_history */
890 PROC SQL;
891
892 SELECT *
893 FROM LIB77702.TRAINING_DS e
894 WHERE ( e.loan_history eq . OR e.loan_history IS MISSING );
895
896 QUIT;

```

8.1.5.1.2 Output

SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY
LP001034	Male	Not Married	1	Under Graduate	No	3500	0	100	240	.
LP001052	Male	Married	1	Graduate	No	3717	2925	151	360	.
LP001091	Male	Married	1	Graduate	No	4186	3369	201	360	.
LP001123	Male	Married	0	Graduate	No	2400	0	75	360	.
LP001264	Male	Married	3	Under Graduate	Yes	3333	2186	130	360	.
LP001273	Male	Married	0	Graduate	No	6000	2250	265	360	.
LP001280	Male	Married	2	Under Graduate	No	3333	2000	99	360	.
LP001326	Male	Not Married	0	Graduate	No	6782	0	.	360	.

8.1.5.1.3 Description

From the output above, we observe that the training set contains missing values for loan history; these values will be imputed later.

8.1.5.2 Counting Number of Missing Values – LOAN_HISTORY

8.1.5.2.1 Code Snippet

```

898 /* STEP 2:Count the number of the loan applicants who submitted the applications without loan_history */
899 PROC SQL;
900
901 SELECT COUNT (*) Label = 'Number of Loan Applicants'
902 FROM LIB77702.TRAINING_DS e
903 WHERE ( e.loan_history eq . OR e.loan_history IS MISSING );
904
905 QUIT;

```

8.1.5.2.2 Output

Number of Loan Applicants
50

8.1.5.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their loan history, which is 50 loan applicants.

8.1.5.3 Finding the Statistics – LOAN_HISTORY

8.1.5.3.1 Code Snippet

```

907 /* STEP 3: Find the statistics for loan_history */
908
909 TITLE1 'Find the statistics for loan_history';
910
911 PROC SQL;
912
913 SELECT e.loan_history AS LOAN_HISTORY,
914        COUNT(*) AS COUNTS
915 FROM LIB77702.TRAINING_DS e
916 WHERE ( e.loan_history NE . OR e.loan_history IS NOT MISSING )
917 GROUP BY e.loan_history;
918
919 QUIT;

```

8.1.5.3.2 Output

Find the statistics for loan_history

LOAN_HISTORY	COUNTS
0	89
1	475

8.1.5.3.3 Description

The statistics for loan history show that out of the total applicants, 89 had no past loans and 475 had loans in the past. This indicates that most of the applicants had loans in the past.

8.1.5.4 Saving the Statistics – LOAN_HISTORY

8.1.5.4.1 Code Snippet

```

921 /* STEP 4: Save the statistics for loan_history in a dataset */
922
923 PROC SQL;
924
925 CREATE TABLE LIB77702.TRAINING_STATS_DS AS
926 SELECT e.loan_history AS LOAN_HISTORY,
927        COUNT(*) AS COUNTS
928 FROM LIB77702.TRAINING_DS e
929 WHERE ( e.loan_history NE . OR e.loan_history IS NOT MISSING )
930 GROUP BY e.loan_history;
931
932 QUIT;

```

8.1.5.4.2 Output

Table: | View: | | Filter: (none)

Columns Total rows: 2 Total columns: 2

<input checked="" type="checkbox"/>	Select all			
<input checked="" type="checkbox"/>	LOAN_HISTORY			
<input checked="" type="checkbox"/>	COUNTS			

	LOAN_HISTORY	COUNTS
1	0	89
2	1	475

8.1.5.4.3 Description

The above output shows that the statistics performed earlier is saved successfully in TRAINING_STATS_DS.

8.1.5.5 Backup of the Dataset – LOAN_HISTORY

8.1.5.5.1 Code Snippet

```

934 /* STEP 4.1: Make a backup of the dataset */
935
936 PROC SQL;
937
938 CREATE TABLE LIB77702.TRAINING_BACKUP_DS AS
939 SELECT *
940 FROM LIB77702.TRAINING_DS;
941
942 QUIT;

```

8.1.5.5.2 Output

Table: **_TEMP0.TRAINING_BACKUP_DS** | View: **Column names** | Filter: (none)

Total rows: 614 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001002	Male	Not Married	0	Graduate	No	5849	0
2	LP001003	Male	Married	1	Graduate	No	4583	1508
3	LP001005	Male	Married	0	Graduate	Yes	3000	0
4	LP001006	Male	Married	0	Under Graduate	No	2583	2358
5	LP001008	Male	Not Married	0	Graduate	No	6000	0
6	LP001011	Male	Married	2	Graduate	Yes	5417	4196
7	LP001013	Male	Married	0	Under Graduate	No	2333	1516
8	LP001014	Male	Married	3	Graduate	No	3036	2504
9	LP001018	Male	Married	2	Graduate	No	4006	1526
10	LP001020	Male	Married	1	Graduate	No	12841	10968

8.1.5.5.3 Description

The newly formed backup table LIB77702.TRAINING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.1.5.6 Imputing the missing values– LOAN_HISTORY

8.1.5.6.1 Code Snippet

```

944 /* STEP 5: Impute the missing values in the categorical variable (LOAN_HISTORY) */
945
946 PROC SQL;
947
948 UPDATE LIB77702.TRAINING_DS
949 SET loan_history = ( SELECT loan_history AS loan_history
950                     FROM LIB77702.TRAINING_STATS_DS t2
951                     WHERE t2.counts eq ( SELECT MAX(t1.counts) AS HIGHEST_COUNT
952                                         FROM LIB77702.TRAINING_STATS_DS t1 ) )
953 /* Above is sub-program to find the highest count */
954 WHERE ( loan_history eq . OR loan_history IS MISSING );
955
956 QUIT;

```

8.1.5.6.2 Output

NOTE: 50 rows were updated in LIB77702.TRAINING_DS.

8.1.5.6.3 Description

The above output shows that the 50 missing values rows were successfully imputed and updated in loan history.

8.1.5.7 After imputing the missing values – LOAN_HISTORY

8.1.5.7.1 Code Snippet

```

958 /* STEP 6: (AFTER IMPUTATION) Find missing values for loan_history */
959
960 TITLE1 '(AFTER IMPUTATION) Find missing values for loan_history';
961 TITLE2 'Loan application without loan_history';
962 FOOTNOTE '-----End-----';
963
964 PROC SQL;
965
966 SELECT *
967 FROM LIB77702.TRAINING_DS e
968 WHERE ( e.loan_history eq . OR e.loan_history IS MISSING );
969
970 QUIT;

```

8.1.5.7.2 Output

(AFTER IMPUTATION) Find missing values for loan_history
Loan application without loan_history

-----End-----

8.1.5.7.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.2 Imputing the missing values in Continuous variables – LIB7702.TRAINING_DS

8.2.1 Handling Missing Values in LOAN_AMOUNT

8.2.1.1 Find Missing Value – LOAN_AMOUNT

8.2.1.1.1 Code Snippet

```

974 /* STEP 1: List the details of the loan applicants who submitted the applications without loan_amount */
975 TITLE 'STEP 1: List the details of the loan applicants who submitted the applications without loan_amount';
976 FOOTNOTE '-----END-----';
977
978 PROC SQL;
979
980 SELECT *
981 FROM LIB77702.TRAINING_DS t
982 WHERE ( t.loan_amount eq . OR t.loan_amount IS MISSING );
983
984 QUIT;

```

8.2.1.1.2 Output

STEP 1: List the details of the loan applicants who submitted the applications without loan_amount											
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY	LOAN_LOCATION
LP001002	Male	Not Married	0	Graduate	No	5849	0	.	360	1	City
LP001106	Male	Married	0	Graduate	No	2275	2067	.	360	1	City
LP001213	Male	Married	1	Graduate	No	4645	0	.	360	0	Village
LP001266	Male	Married	1	Graduate	Yes	2395	0	.	360	1	Town
LP001326	Male	Not Married	0	Graduate	No	6792	0	.	360	1	City
LP001350	Male	Married	0	Graduate	No	13650	0	.	360	1	City
LP001356	Male	Married	0	Graduate	No	4652	3583	.	360	1	Town

8.2.1.1.3 Description

From the output above, we observe that the training set contains missing values for loan amount; these values will be imputed later.

8.2.1.2 Counting Number of Missing Values – LOAN_AMOUNT

8.2.1.2.1 Code Snippet

```

986 /* STEP 2: Count the number of the loan applicants who submitted the applications without loan_amount */
987 PROC SQL;
988
989 SELECT COUNT (*) Label = 'Number of Loan Applicants'
990 FROM LIB77702.TRAINING_DS t
991 WHERE ( t.loan_amount eq . OR t.loan_amount IS MISSING );
992
993 QUIT;

```

8.2.1.2.2 Output

Number of Loan Applicants
22

8.2.1.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their loan amount, which is 22 loan applicants.

8.2.1.3 Backup of the Dataset – LOAN_AMOUNT

8.2.1.3.1 Code Snippet

```

995 /* Create a Back-Up */
996 PROC SQL;
997
998 CREATE TABLE LIB77702.TRAINING_BACKUP_DS AS
999 SELECT *
1000 FROM LIB77702.TRAINING_DS;
1001
1002 QUIT;

```

8.2.1.3.2 Output

Table: View: Filter: (none)

Total rows: 614 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001002	Male	Not Married	0	Graduate	No	5849	0
2	LP001003	Male	Married	1	Graduate	No	4583	1508
3	LP001005	Male	Married	0	Graduate	Yes	3000	0
4	LP001006	Male	Married	0	Under Graduate	No	2583	2358
5	LP001008	Male	Not Married	0	Graduate	No	6000	0
6	LP001011	Male	Married	2	Graduate	Yes	5417	4196
7	LP001013	Male	Married	0	Under Graduate	No	2333	1516
8	LP001014	Male	Married	3	Graduate	No	3036	2504
9	LP001018	Male	Married	2	Graduate	No	4006	1526
10	LP001020	Male	Married	1	Graduate	No	12841	10968

8.2.1.3.3 Description

The newly formed backup table LIB77702.TRAINING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.2.1.4 Imputing the missing values– LOAN_AMOUNT

8.2.1.4.1 Code Snippet

```

1004 /* STEP 3: Impute the missing values found in the Continuous variable - LOAN_AMOUNT */
1005
1006 PROC STDIZE DATA = LIB77702.TRAINING_DS REONLY
1007
1008 METHOD = MEAN OUT = LIB77702.TRAINING_DS;
1009 var loan_amount;
1010
1011 QUIT;

```

8.2.1.4.2 Output

Table: _TEMP0.TRAINING_DS View: Column names Filter: (none)

Total rows: 614 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT
1	LP001002	Male	Not Married	0	Graduate	No	5849	0	146.41216216
2	LP001003	Male	Married	1	Graduate	No	4583	1508	128
3	LP001005	Male	Married	0	Graduate	Yes	3000	0	66
4	LP001006	Male	Married	0	Under Graduate	No	2583	2358	120
5	LP001008	Male	Not Married	0	Graduate	No	6000	0	141

8.2.1.4.3 Description

The above output shows that the missing value of the loan amount was successfully imputed using the loan amount's average value.

8.2.1.5 After imputing the missing values – LOAN_AMOUNT

8.2.1.5.1 Code Snippet

```

1024 /* STEP 5: (After Imputation) Count the number of the loan applicants who submitted the applications without loan_amount */
1025 FOOTNOTE '-----END-----';
1026
1027 PROC SQL;
1028
1029 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1030 FROM LIB77702.TRAINING_DS t
1031 WHERE ( t.loan_amount eq . OR t.loan_amount IS MISSING );
1032
1033 QUIT;

```

8.2.1.5.2 Output

Number of Loan Applicants
0

8.2.1.5.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.2.2 Handling Missing Values in LOAN_DURATION

8.2.2.1 Find Missing Value – LOAN_DURATION

8.2.2.1.1 Code Snippet

```

1037 /* STEP 1: List the details of the loan applicants who submitted the applications without loan_duration */
1038 TITLE 'STEP 1: List the details of the loan applicants who submitted the applications without loan_duration';
1039 FOOTNOTE '-----END-----';
1040
1041 PROC SQL;
1042
1043 SELECT *
1044 FROM LIB77702.TRAINING_DS t
1045 WHERE ( t.loan_duration eq . OR t.loan_duration IS MISSING );
1046
1047 QUIT;

```

8.2.2.1.2 Output

STEP 1: List the details of the loan applicants who submitted the applications without loan_duration										
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY
LP001041	Male	Married	0	Graduate	No	2800	3500	115	.	1
LP001109	Male	Married	0	Graduate	No	1828	1330	100	.	0
LP001136	Male	Married	0	Under Graduate	Yes	4695	0	96	.	1
LP001137	Female	Not Married	0	Graduate	No	3410	0	88	.	1
LP001250	Male	Married	3	Under Graduate	No	4755	0	95	.	0

8.2.2.1.3 Description

From the output above, we observe that the training set contains missing values for loan duration; these values will be imputed later.

8.2.2.2 Counting Number of Missing Values – LOAN_DURATION

8.2.2.2.1 Code Snippet

```

1049 /* STEP 2: Count the number of the loan applicants who submitted the applications without loan_duration */
1050 PROC SQL;
1051
1052 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1053 FROM LIB77702.TRAINING_DS t
1054 WHERE ( t.loan_duration eq . OR t.loan_duration IS MISSING );
1055
1056 QUIT;

```

8.2.2.2.2 Output

Number of Loan Applicants
14

8.2.2.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their loan duration, which is 14 loan applicants.

8.2.2.3 Backup of the Dataset – LOAN_DURATION

8.2.2.3.1 Code Snippet

```

1058 /* Create a Back-Up */
1059 PROC SQL;
1060
1061 CREATE TABLE LIB77702.TRAINING_BACKUP_DS AS
1062 SELECT *
1063 FROM LIB77702.TRAINING_DS;
1064
1065 QUIT;

```

8.2.2.3.2 Output

Table: View: Filter: (none)

Total rows: 614 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001002	Male	Not Married	0	Graduate	No	5849	0
2	LP001003	Male	Married	1	Graduate	No	4583	1508
3	LP001005	Male	Married	0	Graduate	Yes	3000	0
4	LP001006	Male	Married	0	Under Graduate	No	2583	2358
5	LP001008	Male	Not Married	0	Graduate	No	6000	0
6	LP001011	Male	Married	2	Graduate	Yes	5417	4196
7	LP001013	Male	Married	0	Under Graduate	No	2333	1516
8	LP001014	Male	Married	3	Graduate	No	3036	2504
9	LP001018	Male	Married	2	Graduate	No	4006	1526
10	LP001020	Male	Married	1	Graduate	No	12841	10968

8.2.2.3.3 Description

The newly formed backup table LIB77702.TRAINING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.2.2.4 Imputing the missing values– LOAN_DURATION

8.2.2.4.1 Code Snippet

```

1067 /* STEP 3: Impute the missing values found in the Continous varibale - loan_duration */
1068
1069 PROC STDIZE DATA = LIB77702.TRAINING_DS REPNLY
1070
1071 METHOD = MEAN OUT = LIB77702.TRAINING_DS;
1072 var loan_duration;
1073
1074 QUIT;

```

8.2.2.4.2 Output

Table: View: Filter: (none)

Total rows: 614 Total columns: 13

	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION
0	Under Graduate	Yes		2609	3449	165	180
1	Graduate	No		4945	0	146,412,162,16	360
0	Graduate	No		4166	0	116	360
0	Graduate	No		5726	4595	258	360
0	Under Graduate	No		3200	2254	126	180
1	Graduate	No		10750	0	312	360
3	Under Graduate	Yes		7100	0	125	60

8.2.2.4.3 Description

The above output shows that the missing value of the loan duration was correctly imputed using the loan duration's average value.

8.2.2.5 After imputing the missing values – LOAN_DURATION

8.2.2.5.1 Code Snippet

```
1087 /* STEP 4.1: (After Imputation) Count the number of the loan applicants who submitted the applications without loan_duration */
1088 FOOTNOTE '-----END-----';
1089
1090 PROC SQL;
1091
1092 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1093 FROM LIB77702.TRAINING_DS t
1094 WHERE ( t.loan_duration eq . OR t.loan_duration IS MISSING );
1095
1096 QUIT;
```

8.2.2.5.2 Output

Number of Loan Applicants
0

8.2.2.5.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.3 Imputing the missing values in Categorical variables – LIB7702.TESTING_DS

8.3.1 Handling Missing Values in FAMILY_MEMBERS

8.3.1.1 Find missing values – FAMILY_MEMBERS

8.3.1.1.1 Code Snippet

```
1484 /* STEP 1: List the details of the loan applicants who submitted the applications without family member details */
1485 PROC SQL;
1486
1487 SELECT *
1488 FROM LIB77702.TESTING_DS e
1489 WHERE ( e.family_members eq '' OR e.family_members IS NULL );
1490
1491 QUIT;
```

8.3.1.1.2 Output

SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY	LOAN_LOCATION
LP001237	Male	Married		Under Graduate	No	4183	1475	182	360	1	City
LP001396	Female	Not Married		Graduate	No	3250	0	95	360	1	Town
LP001587	Male	Married		Graduate	No	4082	0	93	360	1	Town
LP001789		Not Married		Graduate	No	3333	1250	110	360	1	Town
LP002111	Male	Married		Graduate	No	3016	1300	100	360	.	City
LP002380	Male	Married		Graduate	No	10000	0	.	360	1	City

8.3.1.1.3 Description

From the output above, we observe that the testing set contains missing values for family members; these values will be imputed later.

8.3.1.2 Counting the Number of Missing values – FAMILY_MEMBERS

8.3.1.2.1 Code Snippet

```
1493 /* STEP 2: Count the number of the loan applicants who submitted the applications without family member details */
1494 PROC SQL;
1495
1496 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1497 FROM LIB77702.TESTING_DS e
1498 WHERE ( e.family_members eq '' OR e.family_members IS NULL );
1499
1500 QUIT;
```

8.3.1.2.2 Output

Number of Loan Applicants
10

8.3.1.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their family members, which is 10 loan applicants.

8.3.1.3 List Details of Family with ‘3+’ Members

8.3.1.3.1 Code Snippet

```
1502 /* STEP 3: List the details of the loan applicants with '3+' family members who submitted
1503 the applications without family member details */
1504 PROC SQL;
1505
1506 SELECT e.family_members Label = 'Family Members',
1507        SUBSTR(e.family_members,1,1) Label = 'The data found in the 1st position',
1508        SUBSTR(e.family_members,2,1) Label = 'The data found in the 2nd position'
1509 FROM LIB77702.TESTING_DS e
1510 WHERE ( e.family_members ne '' OR e.family_members IS NOT MISSING );
1511
1512 QUIT;
```

8.3.1.3.2 Output

Family Members	The data found in the 1st position	The data found in the 2nd position
0	0	
1	1	
0	0	
0	0	
0	0	
2	2	
0	0	
3+	3	+
2	2	

8.3.1.3.3 Description

From the above output we can see that applicant with 3+ family member is split into two positions to remove ‘+’ sign in the later stage.

8.3.1.4 Backup of Dataset – FAMILY_MEMBERS

8.3.1.4.1 Code Snippet

```

1514 /* STEP 4: Make a backup of the dataset */
1515
1516 PROC SQL;
1517
1518 CREATE TABLE LIB77702.TESTING_BACKUP_DS AS
1519 SELECT *
1520 FROM LIB77702.TESTING_DS;
1521
1522 QUIT;

```

8.3.1.4.2 Output

Table: | View: | | Filter: (none)

Total rows: 367 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001015	Male	Married	0	Graduate	No	5720	0
2	LP001022	Male	Married	1	Graduate	No	3076	1500
3	LP001031	Male	Married	2	Graduate	No	5000	1800
4	LP001035	Male	Married	2	Graduate	No	2340	2546
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	0
6	LP001054	Male	Married	0	Under Graduate	Yes	2165	3422

8.3.1.4.3 Description

The newly formed backup table LIB77702.TESTING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.3.1.5 Removing '+' Found in Family Member

8.3.1.5.1 Code Snippet

```

1524 /* STEP 5: Remove the '+' found in the family_members variable */
1525 PROC SQL;
1526
1527 UPDATE LIB77702.TESTING_DS
1528 SET family_members = SUBSTR(family_members,1,1)
1529 WHERE SUBSTR(family_members,2,1) eq '+';
1530
1531 QUIT;

```

8.3.1.5.2 Output

NOTE: 40 rows were updated in LIB77702.TESTING_DS.

8.3.1.5.3 Description

From the above output we can see that the '+' sign has been removed successfully and the 40 rows have been updated in family members.

8.3.1.6 Finding the Statistics – FAMILY_MEMBERS

8.3.1.6.1 Code Snippet

```

1533 /* STEP 6: Display the statistics in a dataset */
1534
1535 PROC SQL;
1536
1537 SELECT e.family_members AS family_members,
1538 COUNT (*) AS COUNTS
1539 FROM LIB77702.TESTING_DS e
1540 WHERE ( e.family_members ne '' or e.family_members IS NOT MISSING )
1541 GROUP BY e.family_members;
1542
1543 QUIT;

```

8.3.1.6.2 Output

family_members	COUNTS
0	200
1	58
2	59
3	40

8.3.1.6.3 Description

The statistics for family members show that out of the total applicants, 200 are having 0 family members, 58 having 1 member, 59 having 2 members and 40 having 3 members. This indicates that most of the applicants are having no family members.

8.3.1.7 Saving the Statistics – FAMILY_MEMBERS

8.3.1.7.1 Code Snippet

```

1545 /* STEP 7: Save the statistics in a dataset */
1546
1547 PROC SQL;
1548
1549 CREATE TABLE LIB77702.TESTING_STAT_FM AS
1550 SELECT e.family_members AS family_members,
1551 COUNT (*) AS counts
1552 FROM LIB77702.TESTING_DS e
1553 WHERE ( e.family_members ne '' or e.family_members IS NOT MISSING )
1554 GROUP BY e.family_members;
1555
1556 QUIT;

```

8.3.1.7.2 Output

Table: LIB77702.TESTING_STAT_FM | View: Column names | Filter

Columns Total rows: 4 Total columns: 2

<input checked="" type="checkbox"/>	Select all		
<input checked="" type="checkbox"/>	family_members	family_members	counts
<input checked="" type="checkbox"/>	counts		

1	0	200
2	1	58
3	2	59
4	3	40

8.3.1.7.3 Description

The above output shows that the statistics performed earlier is saved successfully in TESTING_STATS_DS.

8.3.1.8 Imputing the Missing Value – FAMILY_MEMBERS

8.3.1.8.1 Code Snippet

```

1558 /* STEP 8: Impute the missing values found in the Categorical Variable - FAMILY_MEMBERS */
1559
1560 PROC SQL;
1561
1562 UPDATE LIB77702.TESTING_DS
1563 SET family_members = ( SELECT to.family_members AS family_members
1564                        FROM LIB77702.TESTING_STAT_FM to
1565                        WHERE to.counts eq ( SELECT MAX (ti.counts) AS highest_count
1566                                           FROM LIB77702.TESTING_STAT_FM ti ) )
1567 /* Above is a sub-program to find the highest count */
1568 WHERE ( family_members eq '' OR family_members IS NULL );
1569
1570 QUIT;

```

8.3.1.8.2 Output

NOTE: 10 rows were updated in LIB77702.TESTING_DS.

8.3.1.8.3 Description

The above output shows that the 10 missing values rows were successfully imputed and updated in family members.

8.3.1.9 After Imputing Missing Value – FAMILY_MEMBERS

8.3.1.9.1 Code Snippet

```

1581 /* STEP 9.1: (After Imputation) Count the number of the loan applicants who submitted
1582 the applications without family member details */
1583 PROC SQL;
1584
1585 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1586 FROM LIB77702.TESTING_DS e
1587 WHERE ( e.family_members eq '' OR e.family_members IS NULL );
1588
1589 QUIT;

```

8.3.1.9.2 Output

Number of Loan Applicants
0

8.3.1.9.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.3.2 Handling Missing Values in EMPLOYMENT

8.3.2.1 Find Missing Value – EMPLOYMENT

8.3.2.1.1 Code Snippet

```

1219 /* STEP 1: Find missing values for employment */
1220
1221 TITLE1 'Find missing values for employment';
1222 TITLE2 'Loan application without employment';
1223 PROC SQL;
1224
1225 SELECT *
1226 FROM LIB77702.TESTING_DS e
1227 WHERE ( e.employment eq '' OR e.employment IS MISSING );
1228
1229 QUIT;

```

8.3.2.1.2 Output

Find missing values for employment Loan application without employment								
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT
LP001059	Male	Married	2	Graduate		13633	0	280
LP001082	Male	Married	1	Graduate		2185	1516	162
LP001094	Male	Married	2	Graduate		12173	0	166
LP001208	Male	Married	2	Graduate		7350	4029	185
LP001375	Male	Married	1	Graduate		4083	1775	139
LP001472	Female	Not Married	0	Graduate		5058	0	200

8.3.2.1.3 Description

From the output above, we observe that the testing set contains missing values for employment; these values will be imputed later.

8.3.2.2 Counting Number of Missing Values – EMPLOYMENT

8.3.2.2.1 Code Snippet

```

1231 /* STEP 2: Count the number of missing values for employment */
1232
1233 TITLE1 'Find missing values for employment';
1234 TITLE2 'Loan application without employment';
1235 PROC SQL;
1236
1237 SELECT COUNT(*) Label = 'Number of Loan Applicants'
1238 FROM LIB77702.TESTING_DS e
1239 WHERE ( e.employment eq '' OR e.employment IS MISSING );
1240
1241 QUIT;

```

8.3.2.2.2 Output

Count missing values for employment Loan application without employment	
Number of Loan Applicants	23

8.3.2.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their employment status, which is 23 loan applicants.

8.3.2.3 Finding the Statistics – EMPLOYMENT

8.3.2.3.1 Code Snippet

```

1243 /* STEP 3: Find the statistics for employment */
1244
1245 TITLE1 'Find the statistics for employment';
1246
1247 PROC SQL;
1248
1249 SELECT e.employment AS EMPLOYMENT,
1250        COUNT(*) AS COUNTS
1251 FROM LIB77702.TESTING_DS e
1252 WHERE ( e.employment NE '' OR e.employment IS NOT MISSING )
1253 GROUP BY e.employment;
1254
1255 QUIT;

```

8.3.2.3.2 Output

Find the statistics for employment

EMPLOYMENT	COUNTS
No	307
Yes	37

8.3.2.3.3 Description

The statistics for employment show that out of the total applicants, 307 are employed and 37 are not employed. This indicates that most of the applicants are employed, while rest are not employed.

8.3.2.4 Saving the Statistics – EMPLOYMENT

8.3.2.4.1 Code Snippet

```

1257 /* STEP 4: Save the statistics for employment in a dataset */
1258
1259 TITLE1 'Save the statistics for employment in a dataset';
1260
1261 PROC SQL;
1262
1263 CREATE TABLE LIB77702.TESTING_STATS_DS AS
1264 SELECT e.employment AS EMPLOYMENT,
1265        COUNT(*) AS COUNTS
1266 FROM LIB77702.TESTING_DS e
1267 WHERE ( e.employment NE '' OR e.employment IS NOT MISSING )
1268 GROUP BY e.employment;
1269
1270 QUIT;

```






8.3.2.4.2 Output

Table:

_TEMP0.TESTING_STATS_DS


 | View:


Column names

 |     

Columns

☒ Select all

☒  EMPLOYMENT

☒  COUNTS

Total rows: 2

Total columns: 2

	EMPLOYM...	COUNTS
1	No	307
2	Yes	37

8.3.2.4.3 Description

The above output shows that the statistics performed earlier is saved successfully in TESTING_STATS_DS.

8.3.2.5 Backup of the Dataset – EMPLOYMENT

8.3.2.5.1 Code Snippet

```

1272 /* STEP 4.1: Make a backup of the dataset */
1273
1274 PROC SQL;
1275
1276 CREATE TABLE LIB77702.TESTING_BACKUP_DS AS
1277 SELECT *
1278 FROM LIB77702.TESTING_DS;
1279
1280 QUIT;

```

8.3.2.5.2 Output

Table: View: Filter: (none)

Total rows: 367 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001015	Male	Married	0	Graduate	No	5720	0
2	LP001022	Male	Married	1	Graduate	No	3076	1500
3	LP001031	Male	Married	2	Graduate	No	5000	1800
4	LP001035	Male	Married	2	Graduate	No	2340	2546
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	0
6	LP001054	Male	Married	0	Under Graduate	Yes	2165	3422

8.3.2.5.3 Description

The newly formed backup table LIB77702.TESTING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.3.2.6 Imputing the missing values– EMPLOYMENT

8.3.2.6.1 Code Snippet

```

1282 /* STEP 5: Impute the missing values in the categorical variable (EMPLOYMENT) */
1283
1284 PROC SQL;
1285
1286 UPDATE LIB77702.TESTING_DS
1287 SET employment = ( SELECT employment AS employment
1288                     FROM LIB77702.TESTING_STATS_DS t2
1289                     WHERE t2.counts eq ( SELECT MAX(t1.counts) AS HIGHEST_COUNT
1290                                         FROM LIB77702.TESTING_STATS_DS t1 ) )
1291                     /* Above is sub-program to find the highest count */
1292 WHERE ( employment eq '' OR employment IS MISSING );
1293
1294 QUIT;

```

8.3.2.6.2 Output

NOTE: 23 rows were updated in LIB77702.TESTING_DS.

8.3.2.6.3 Description

The above output shows that the 23 missing values rows were successfully imputed and updated in employment.

8.3.2.7 After imputing the missing values – EMPLOYMENT

8.3.2.7.1 Code Snippet

```

1296 /* STEP 6: (AFTER IMPUTATION) Find missing values for employment */
1297
1298 TITLE1 '(AFTER IMPUTATION) Find missing values for employment';
1299 TITLE2 'Loan application without employment';
1300 FOOTNOTE '-----End-----';
1301
1302 PROC SQL;
1303
1304 SELECT *
1305 FROM LIB77702.TESTING_DS e
1306 WHERE ( e.employment eq '' OR e.employment IS MISSING );
1307
1308 QUIT;

```

8.3.2.7.2 Output

(AFTER IMPUTATION) Find missing values for employment
Loan application without employment

-----End-----

8.3.2.7.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.3.3 Handling Missing Values in GENDER

8.3.3.1 Find Missing Value – GENDER

8.3.3.1.1 Code Snippet

```
1312 /* STEP 1: Find missing values for gender */
1313
1314 TITLE1 'Find missing values for gender';
1315 TITLE2 'Loan application without gender';
1316 PROC SQL;
1317
1318 SELECT *
1319 FROM LIB77702.TESTING_DS e
1320 WHERE ( e.gender eq '' OR e.gender IS MISSING );
1321
1322 QUIT;
```

8.3.3.1.2 Output

Find missing values for gender Loan application without gender												
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY	LOAN_LOCATION	LOAN_APPROVAL_STATUS
LP001050		Married	2	Under Graduate	No	3365	1917	112	360	0	Village	N
LP001448		Married	3	Graduate	No	23803	0	370	360	1	Village	Y
LP001565		Married	3	Graduate	No	51763	0	700	360	1	City	Y
LP001644		Married	0	Graduate	Yes	674	5266	168	360	1	Village	Y
LP002024		Married	0	Graduate	No	2473	1843	156	360	1	Village	N
LP002103		Married	1	Graduate	Yes	9833	1833	182	180	1	City	Y
LP002478		Married	0	Graduate	Yes	2083	4083	160	360	.	Town	Y

8.3.3.1.3 Description

From the output above, we observe that the testing set contains missing values for gender; these values will be imputed later.

8.3.3.2 Counting Number of Missing Values – GENDER

8.3.3.2.1 Code Snippet

```
1324 /* STEP 2: Count the number of missing values for gender */
1325
1326 TITLE1 'Count missing values for gender';
1327 TITLE2 'Loan application without gender';
1328 PROC SQL;
1329
1330 SELECT COUNT(*) Label = 'Number of Loan Applicants'
1331 FROM LIB77702.TESTING_DS e
1332 WHERE ( e.gender eq '' OR e.gender IS MISSING );
1333
1334 QUIT;
```

8.3.3.2.2 Output

Count missing values for gender
Loan application without gender

Number of Loan Applicants	
	11

8.3.3.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their gender, which is 11 loan applicants.

8.3.3.3 Finding the Statistics – GENDER

8.3.3.3.1 Code Snippet

```

1336 /* STEP 3: Find the statistics for gender */
1337
1338 TITLE1 'Find the statistics for gender';
1339
1340 PROC SQL;
1341
1342 SELECT e.gender AS GENDER,
1343        COUNT(*) AS COUNTS
1344 FROM LIB77702.TESTING_DS e
1345 WHERE ( e.gender NE '' OR e.gender IS NOT MISSING )
1346 GROUP BY e.gender;
1347
1348 QUIT;

```

8.3.3.3.2 Output

Find the statistics for gender

GENDER	COUNTS
Female	70
Male	286

8.3.3.3.3 Description

The statistics for marital status show that out of the total applicants, 70 are female and 286 are male. This indicates that most of the applicants are male.

8.3.3.4 Saving the Statistics – GENDER





8.3.3.4.1 Code Snippet


```

1350 /* STEP 4: Save the statistics for employment in a dataset */
1351
1352 TITLE1 'Save the statistics for employment in a dataset';
1353
1354 PROC SQL;
1355
1356 CREATE TABLE LIB77702.TESTING_STATS_DS AS
1357 SELECT e.gender AS GENDER,
1358        COUNT(*) AS COUNTS
1359 FROM LIB77702.TESTING_DS e
1360 WHERE ( e.gender NE '' OR e.gender IS NOT MISSING )
1361 GROUP BY e.gender;
1362
1363 QUIT;

```

8.3.3.4.2 Output

Table: **_TEMP0.TESTING_STATS_DS** | View: **Column names** |    

Columns  Total rows: 2 Total columns: 2

	GEND...	COUNTS
1	Female	70
2	Male	286

8.3.3.4.3 Description

The above output shows that the statistics performed earlier is saved successfully in TESTING_STATS_DS.

8.3.3.5 Backup of the Dataset – GENDER





8.3.3.5.1 Code Snippet


```

1365 /* STEP 4.1: Make a backup of the dataset */
1366
1367 PROC SQL;
1368
1369 CREATE TABLE LIB77702.TESTING_BACKUP_DS AS
1370 SELECT *
1371 FROM LIB77702.TESTING_DS;
1372
1373 QUIT;

```

8.3.3.5.2 Output

Table: **_TEMP0.TESTING_BACKUP_DS** | View: **Column names** |     | Filter: (none)

 Total rows: 367 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001015	Male	Married	0	Graduate	No	5720	0
2	LP001022	Male	Married	1	Graduate	No	3076	1500
3	LP001031	Male	Married	2	Graduate	No	5000	1800
4	LP001035	Male	Married	2	Graduate	No	2340	2546
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	0
6	LP001054	Male	Married	0	Under Graduate	Yes	2165	3422

8.3.3.5.3 Description

The newly formed backup table LIB77702.TESTING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.3.3.6 Imputing the missing values– GENDER

8.3.3.6.1 Code Snippet

```

1375 /* STEP 5: Impute the missing values in the categorical variable (GENDER) */
1376
1377 PROC SQL;
1378
1379 UPDATE LIB77702.TESTING_DS
1380 SET gender = ( SELECT gender AS gender
1381                FROM LIB77702.TESTING_STATS_DS t2
1382                WHERE t2.counts eq ( SELECT MAX(t1.counts) AS HIGHEST_COUNT
1383                                     FROM LIB77702.TESTING_STATS_DS t1 ) )
1384 /* Above is sub-program to find the highest count */
1385 WHERE ( gender eq '' OR gender IS MISSING );
1386
1387 QUIT;

```


8.3.3.6.2 Output

NOTE: 11 rows were updated in LIB77702.TESTING_DS.

8.3.3.6.3 Description

The above output shows that the 11 missing values rows were successfully imputed and updated in gender.

8.3.3.7 After imputing the missing values – GENDER

8.3.3.7.1 Code Snippet

```
1389 /* STEP 6: (AFTER IMPUTATION) Find missing values for gender */
1390
1391 TITLE1 '(AFTER IMPUTATION) Find missing values for gender';
1392 TITLE2 'Loan application without gender';
1393 FOOTNOTE '-----End-----';
1394
1395 PROC SQL;
1396
1397 SELECT *
1398 FROM LIB77702.TESTING_DS e
1399 WHERE ( e.gender eq '' OR e.gender IS MISSING );
1400
1401 QUIT;
```

8.3.3.7.2 Output

```
(AFTER IMPUTATION) Find missing values for gender
Loan application without gender

-----End-----
```

8.3.3.7.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.3.4 Handling Missing Values in LOAN_HISTORY

8.3.4.1 Find Missing Value – LOAN_HISTORY

8.3.4.1.1 Code Snippet

```
1403 /* STEP 1:List the details of the loan applicants who submitted the applications without loan_history */
1404 PROC SQL;
1405
1406 SELECT *
1407 FROM LIB77702.TESTING_DS e
1408 WHERE ( e.loan_history eq . OR e.loan_history IS MISSING );
1409
1410 QUIT;
```

8.3.4.1.2 Output

SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY
LP001034	Male	Not Married	1	Under Graduate	No	3595	0	100	240	.
LP001052	Male	Married	1	Graduate	No	3717	2925	151	360	.
LP001091	Male	Married	1	Graduate	No	4165	3369	201	360	.
LP001123	Male	Married	0	Graduate	No	2400	0	75	360	.
LP001264	Male	Married	3	Under Graduate	Yes	3333	2186	130	360	.
LP001273	Male	Married	0	Graduate	No	6000	2250	265	360	.
LP001280	Male	Married	2	Under Graduate	No	3333	2000	99	360	.
LP001326	Male	Not Married	0	Graduate	No	6782	0	.	360	.

8.3.4.1.3 Description

From the output above, we observe that the testing set contains missing values for loan history; these values will be imputed later.

8.3.4.2 Counting Number of Missing Values – LOAN_HISTORY

8.3.4.2.1 Code Snippet

```

1412 /* STEP 2:Count the number of the loan applicants who submitted the applications without loan_history */
1413 PROC SQL;
1414
1415 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1416 FROM LIB77702.TESTING_DS e
1417 WHERE ( e.loan_history eq . OR e.loan_history IS MISSING );
1418
1419 QUIT;

```

8.3.4.2.2 Output

Number of Loan Applicants	
	29

8.3.4.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their loan history, which is 29 loan applicants.

8.3.4.3 Finding the Statistics – LOAN_HISTORY

8.3.4.3.1 Code Snippet

```

1421 /* STEP 3: Find the statistics for loan_history */
1422
1423 TITLE1 'Find the statistics for loan_history';
1424
1425 PROC SQL;
1426
1427 SELECT e.loan_history AS LOAN_HISTORY,
1428        COUNT(*) AS COUNTS
1429 FROM LIB77702.TESTING_DS e
1430 WHERE ( e.loan_history NE . OR e.loan_history IS NOT MISSING )
1431 GROUP BY e.loan_history;
1432
1433 QUIT;

```

8.3.4.3.2 Output

Find the statistics for loan_history		
LOAN_HISTORY	COUNTS	
0	59	
1	279	

8.3.4.3.3 Description

The statistics for loan history show that out of the total applicants, 59 had no past loan and 279 had past loan. This indicates that most of the applicants are married, while rest are not married.

8.3.4.4 Saving the Statistics – LOAN_HISTORY

8.3.4.4.1 Code Snippet

```

1435 /* STEP 4: Save the statistics for loan_history in a dataset */
1436
1437 PROC SQL;
1438
1439 CREATE TABLE LIB77702.TESTING_STATS_DS AS
1440 SELECT e.loan_history AS LOAN_HISTORY,
1441        COUNT(*) AS COUNTS
1442 FROM LIB77702.TESTING_DS e
1443 WHERE ( e.loan_history NE . OR e.loan_history IS NOT MISSING )
1444 GROUP BY e.loan_history;
1445
1446 QUIT;

```

8.3.4.4.2 Output

Table: **_TEMP0.TESTING_STATS_DS** | View: **Column names** | Filter: (none)

Columns Total rows: 2 Total columns: 2

	LOAN_HISTORY	COUNTS
1	0	59
2	1	279

8.3.4.4.3 Description

The above output shows that the statistics performed earlier is saved successfully in TESTING_STATS_DS.

8.3.4.5 Backup of the Dataset – LOAN_HISTORY

8.3.4.5.1 Code Snippet

```

1448 /* STEP 4.1: Make a backup of the dataset */
1449
1450 PROC SQL;
1451
1452 CREATE TABLE LIB77702.TESTING_BACKUP_DS AS
1453 SELECT *
1454 FROM LIB77702.TESTING_DS;
1455
1456 QUIT;

```

8.3.4.5.2 Output

Table: **_TEMP0.TESTING_BACKUP_DS** | View: **Column names** | Filter: (none)

Total rows: 367 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001015	Male	Married	0	Graduate	No	5720	0
2	LP001022	Male	Married	1	Graduate	No	3076	1500
3	LP001031	Male	Married	2	Graduate	No	5000	1800
4	LP001035	Male	Married	2	Graduate	No	2340	2546
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	0
6	LP001054	Male	Married	0	Under Graduate	Yes	2165	3422

8.3.4.5.3 Description

The newly formed backup table LIB77702.TESTING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.3.4.6 Imputing the missing values– LOAN_HISTORY

8.3.4.6.1 Code Snippet

```

1458 /* STEP 5: Impute the missing values in the categorical variable (LOAN_HISTORY) */
1459
1460 PROC SQL;
1461
1462 UPDATE LIB77702.TESTING_DS
1463 SET loan_history = ( SELECT loan_history AS loan_history
1464                     FROM LIB77702.TESTING_STATS_DS t2
1465                     WHERE t2.counts eq ( SELECT MAX(t1.counts) AS HIGHEST_COUNT
1466                                         FROM LIB77702.TESTING_STATS_DS t1 ) )
1467                     /* Above is sub-program to find the highest count */
1468 WHERE ( loan_history eq . OR loan_history IS MISSING );
1469
1470 QUIT;

```

8.3.4.6.2 Output

NOTE: 29 rows were updated in LIB77702.TESTING_DS.

8.3.4.6.3 Description

The above output shows that the 29 missing values rows were successfully imputed and updated in loan history.

8.3.4.7 After imputing the missing values – LOAN_HISTORY

8.3.4.7.1 Code Snippet

```

1472 /* STEP 6: (AFTER IMPUTATION) Find missing values for loan_history */
1473
1474 TITLE1 '(AFTER IMPUTATION) Find missing values for loan_history';
1475 TITLE2 'Loan application without loan_history';
1476 FOOTNOTE '-----End-----';
1477
1478 PROC SQL;
1479
1480 SELECT *
1481 FROM LIB77702.TESTING_DS e
1482 WHERE ( e.loan_history eq . OR e.loan_history IS MISSING );
1483
1484 QUIT;

```

8.3.4.7.2 Output

```

(AFTER IMPUTATION) Find missing values for loan_history
Loan application without loan_history

-----End-----

```

8.3.4.7.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.4 Imputing the missing values in Continuous variables – LIB7702.TESTING_DS

8.4.1 Handling Missing Values in LOAN_AMOUNT

8.4.1.1 Find Missing Value – LOAN_AMOUNT

8.4.1.1.1 Code Snippet

```

1597 /* STEP 1: List the details of the loan applicants who submitted the applications without loan_amount */
1598 TITLE 'STEP 1: List the details of the loan applicants who submitted the applications without loan_amount';
1599 FOOTNOTE '-----END-----';
1600
1601 PROC SQL;
1602
1603 SELECT *
1604 FROM LIB7702.TESTING_DS t
1605 WHERE ( t.loan_amount eq . OR t.loan_amount IS MISSING );
1606
1607 QUIT;

```

8.4.1.1.2 Output

STEP 1: List the details of the loan applicants who submitted the applications without loan_amount									
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION
LP001415	Male	Married	1	Graduate	No	3413	4053	.	360
LP001542	Female	Married	0	Graduate	No	2282	0	.	480
LP002057	Male	Married	0	Under Graduate	No	13083	0	.	360
LP002360	Male	Married	0	Graduate	No	10000	0	.	360
LP002593	Male	Married	1	Graduate	No	8333	4000	.	360

8.4.1.1.3 Description

From the output above, we observe that the testing set contains missing values for loan amount; these values will be imputed later.

8.4.1.2 Counting Number of Missing Values – LOAN_AMOUNT

8.4.1.2.1 Code Snippet

```

1609 /* STEP 2: Count the number of the loan applicants who submitted the applications without loan_amount */
1610 PROC SQL;
1611
1612 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1613 FROM LIB7702.TESTING_DS t
1614 WHERE ( t.loan_amount eq . OR t.loan_amount IS MISSING );
1615
1616 QUIT;

```

8.4.1.2.2 Output

Number of Loan Applicants
5

8.4.1.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their loan amount, which is 5 loan applicants.

8.4.1.3 Backup of the Dataset – LOAN_AMOUNT

8.4.1.3.1 Code Snippet

```

1618 /* Create a Back-Up */
1619 PROC SQL;
1620
1621 CREATE TABLE LIB77702.TESTING_BACKUP_DS AS
1622 SELECT *
1623 FROM LIB77702.TESTING_DS;
1624
1625 QUIT;

```

8.4.1.3.2 Output

Table: _TEMP0.TESTING_BACKUP_DS View: Column names Filter: (none)

Total rows: 367 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001015	Male	Married	0	Graduate	No	5720	0
2	LP001022	Male	Married	1	Graduate	No	3076	1500
3	LP001031	Male	Married	2	Graduate	No	5000	1800
4	LP001035	Male	Married	2	Graduate	No	2340	2546
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	0

8.4.1.3.3 Description

The newly formed backup table LIB77702.TESTING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.4.1.4 Imputing the missing values– LOAN_AMOUNT

8.4.1.4.1 Code Snippet

```

1627 /* STEP 3: Impute the missing values found in the Continuous varibale - LOAN_AMOUNT */
1628
1629 PROC STDIZE DATA = LIB77702.TESTING_DS REONLY
1630
1631 METHOD = MEAN OUT = LIB77702.TESTING_DS;
1632 var loan_amount;
1633
1634 QUIT;

```

8.4.1.4.2 Output

Table: _TEMP0.TESTING_DS View: Column names Filter: (none)

Total rows: 367 Total columns: 13

MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT
Not Married	0	Graduate	Yes	6356	0	50
Married	1	Graduate	No	3413	4053	136.13259669
Married	0	Graduate	No	7950	0	185
Married	3	Graduate	No	3829	1103	163
Married	3	Graduate	No	72529	0	360
Married	2	Under Graduate	No	4136	0	149
Married	0	Graduate	No	8449	0	257

8.4.1.4.3 Description

The above output shows that the missing value of the loan amount was correctly imputed using the loan amount's average value.

8.4.1.5 After imputing the missing values – LOAN_AMOUNT

8.4.1.5.1 Code Snippet

```

1647 /* STEP 5: (After Imputation) Count the number of the loan applicants who submitted the applications without loan_amount */
1648 FOOTNOTE '-----END-----';
1649
1650 PROC SQL;
1651
1652 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1653 FROM LIB77702.TESTING_DS t
1654 WHERE ( t.loan_amount eq . OR t.loan_amount IS MISSING );
1655
1656 QUIT;

```

8.4.1.5.2 Output

Number of Loan Applicants
0

8.4.1.5.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

8.4.2 Handling Missing Values in LOAN_DURATION

8.4.2.1 Find Missing Value – LOAN_DURATION

8.4.2.1.1 Code Snippet

```

1658 /* STEP 1: List the details of the loan applicants who submitted the applications without loan_duration */
1659 TITLE 'STEP 1: List the details of the loan applicants who submitted the applications without loan_duration';
1660 FOOTNOTE '-----END-----';
1661
1662 PROC SQL;
1663
1664 SELECT *
1665 FROM LIB77702.TESTING_DS t
1666 WHERE ( t.loan_duration eq . OR t.loan_duration IS MISSING );
1667
1668 QUIT;

```

8.4.2.1.2 Output

STEP 1: List the details of the loan applicants who submitted the applications without loan_duration									
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION
LP001232	Male	Married	0	Graduate	No	4280	3900	185	.
LP001268	Male	Not Married	0	Graduate	No	6792	3338	187	.
LP001611	Male	Married	1	Graduate	No	1516	2900	80	.
LP001695	Male	Married	1	Under Graduate	No	3321	2088	70	.
LP002045	Male	Married	3	Graduate	No	10166	750	150	.
LP002183	Male	Married	0	Under Graduate	No	3754	3719	118	.

8.4.2.1.3 Description

From the output above, we observe that the testing set contains missing values for loan duration; these values will be imputed later.

8.4.2.2 Counting Number of Missing Values – LOAN_DURATION

8.4.2.2.1 Code Snippet

```

1670 /* STEP 2: Count the number of the loan applicants who submitted the applications without loan_duration */
1671 PROC SQL;
1672
1673 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1674 FROM LIB77702.TESTING_DS t
1675 WHERE ( t.loan_duration eq . OR t.loan_duration IS MISSING );
1676
1677 QUIT;

```

8.4.2.2.2 Output

Number of Loan Applicants
6

8.4.2.2.3 Description

The above output displays the number of loan applicants who submitted without disclosing their loan duration, which is 6 loan applicants.

8.4.2.3 Backup of the Dataset – LOAN_DURATION

8.4.2.3.1 Code Snippet

```
1679 /* Create a Back-Up */
1680 PROC SQL;
1681
1682 CREATE TABLE LIB77702.TESTING_BACKUP_DS AS
1683 SELECT *
1684 FROM LIB77702.TESTING_DS;
1685
1686 QUIT;
```

8.4.2.3.2 Output

Table: _TEMP0.TESTING_BACKUP_DS View: Column names Filter: (none)

Total rows: 367 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME
1	LP001015	Male	Married	0	Graduate	No	5720	0
2	LP001022	Male	Married	1	Graduate	No	3076	1500
3	LP001031	Male	Married	2	Graduate	No	5000	1800
4	LP001035	Male	Married	2	Graduate	No	2340	2546
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	0

8.4.2.3.3 Description

The newly formed backup table LIB77702.TESTING_BACKUP_DS's content is displayed in the output. The screenshot ensures that the data may be retrieved or referenced as needed without changing the primary dataset by verifying that the backup was correctly made and contains detailed records that are equal to those in the original dataset.

8.4.2.4 Imputing the missing values– LOAN_DURATION

8.4.2.4.1 Code Snippet

```
1688 /* STEP 3: Impute the missing values found in the Continuous variable - loan_duration */
1689
1690 PROC STDIZE DATA = LIB77702.TESTING_DS REONLY
1691
1692 METHOD = MEAN OUT = LIB77702.TESTING_DS;
1693 var loan_duration;
1694
1695 QUIT;
```


8.4.2.4.2 Output

Table: **_TEMP0.TESTING_DS** | View: **Column names** | Filter: (none)

Total rows: 367 Total columns: 13

EMPLOYM...	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY
No	6500	2600	200	360	1
No	3666	0	84	360	1
No	4260	3900	185	342.53739612	1
No	4163	1475	162	360	1
No	2356	1902	108	360	1
No	6792	3338	187	342.53739612	1
Yes	8000	250	187	360	1

8.4.2.4.3 Description

The above output shows that the statistics performed earlier is saved successfully in TESTING_STATS_DS.

8.4.2.5 After imputing the missing values – LOAN_DURATION

8.4.2.5.1 Code Snippet

```

1708 /* STEP 5: (After Imputation) Count the number of the loan applicants who submitted the applications without loan_duration */
1709 FOOTNOTE '-----END-----';
1710
1711 PROC SQL;
1712
1713 SELECT COUNT (*) Label = 'Number of Loan Applicants'
1714 FROM LIB77702.TESTING_DS t
1715 WHERE ( t.loan_duration eq . OR t.loan_duration IS MISSING );
1716
1717 QUIT;

```

8.4.2.5.2 Output

Number of Loan Applicants
0

8.4.2.5.3 Description

The above output shows that the missing values are absent after the imputation, indicating that the imputation was successfully implemented.

9 Logistic Regression

9.1 Creation of the Model using Logistic Regression Algorithm

9.1.1.1 Code Snippet

```

1721 /* Creation of model using logistic regression algorithm */
1722
1723 PROC LOGISTIC DATA=LIB77702.TRAINING_DS OUTMODEL = LIB77702.TRAINING_DS_LR_MODEL;
1724 CLASS
1725     GENDER
1726     MARITAL_STATUS
1727     FAMILY_MEMBERS
1728     QUALIFICATION
1729     EMPLOYMENT
1730     LOAN_HISTORY
1731     LOAN_LOCATION
1732 ;
1733 MODEL LOAN_APPROVAL_STATUS =
1734     GENDER
1735     MARITAL_STATUS
1736     FAMILY_MEMBERS
1737     QUALIFICATION
1738     EMPLOYMENT
1739     LOAN_HISTORY
1740     LOAN_LOCATION
1741     CANDIDATE_INCOME
1742     GUARANTEE_INCOME
1743     LOAN_AMOUNT
1744     LOAN_HISTORY
1745     LOAN_DURATION
1746 ;
1747
1748 OUTPUT OUT = LIB77702.TRAINING_OUT_DS P = PRED_PROB;
1749 /*PRED_PROB -> Predicted probability - variable to hold predicted probability
1750 OUT -> the output will be sotred in the dataset
1751 Akaike Information Criteria ( AIC ) < SC ( Schwarz Criterion ) */
1752 /*If Pr > ChiSq is <= 0.05, it means that independent variable is an important variable and it is
1753 truly contributing to predict the dependent variable */
1754 RUN;

```

9.1.1.2 Output

Number of Observations Read	614
Number of Observations Used	614

9.1.1.3 Description

The training dataset is well imputed and contains no missing data, as evidenced by the matched number of observations utilised and read (614 observations total). Additionally, 422 applications were expected to be accepted and 192 to be rejected.

9.1.1.4 Output

Model Convergence Status	
Convergence criterion (GCONV=1E-8)	satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	784.891	558.929
SC	789.311	659.649
-2 Log L	782.891	556.929

9.1.1.5 Description

The model convergence status indicates that the convergence criterion is satisfied. Comparing model fit statistics, the AIC and SC values decrease significantly from 784.891 and 789.311 (intercept only) to 558.929 and 659.646 (with covariates), suggesting a better fit with the covariates.

9.1.1.6 Output

Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
GENDER	1	0.0080	0.9287
MARITAL_STATUS	1	5.2980	0.0213
FAMILY_MEMBERS	3	4.4219	0.2194
QUALIFICATION	1	2.5251	0.1120
EMPLOYMENT	2	0.2338	0.8897
LOAN_HISTORY	1	87.2544	<.0001
LOAN_LOCATION	2	12.1746	0.0023
CANDIDATE_INCOME	1	0.2744	0.6004
GUARANTEE_INCOME	1	2.3102	0.1285
LOAN_AMOUNT	1	1.3742	0.2411
LOAN_DURATION	1	0.5372	0.4636

9.1.1.7 Description

When $Pr > ChiSq$ is less than 0.05, it indicates that the independent variable plays a significant role in the prediction of the dependent variable. In this instance, the prediction depends on LOAN_LOCATION, MARITAL_STATUS, and LOAN_HISTORY.

9.2 Predicting the Loan Approval Status Using Logistic Regression

9.2.1.1 Code Snippet

```

1756 /*****
1757 Predict the loan approval status using the model created
1758 *****/
1759 *****/
1760 PROC LOGISTIC INMODEL = LIB77702.TRAINING_DS_LR_MODEL; /* The model that was created */
1761
1762 SCORE DATA = LIB77702.TESTING_DS /*Test ds*/
1763 OUT = LIB77702.TESTING_LAS_PREDICTED_78617_DS; /*Location of output */
1765
1766 QUIT;

```

9.2.1.2 SAS Output

Table: _TEMPO0.TESTING_LAS_PREDICTED_78617_DS | View: Column names | Filter: (none)

Total rows: 367 Total columns: 17

LOAN_HISTORY	LOAN_LOCAT...	LOAN_APPROVAL_ST...	F_LOAN_APPROVAL_ST...	I_LOAN_APPROVAL_ST...	P_N	P_Y
1	City			Y	0.1564815609	0.8435184391
1	City			Y	0.2552196713	0.7447803287
1	City			Y	0.155149763	0.844850237
1	City			Y	0.1390595676	0.8609404324
1	City			Y	0.3278233819	0.6721766181
1	City			Y	0.2998617389	0.7001382611
1	Town			Y	0.2710530155	0.7289469845
0	Village			N	0.9356196705	0.0643803295
1	City			Y	0.1270251286	0.8729748714
1	Town			Y	0.2338503628	0.7661496372
1	City			Y	0.3333114689	0.6666885311
1	Town			Y	0.1566127878	0.8433872122
1	City			Y	0.1844055119	0.8155944881
0	Town			N	0.7829947785	0.2170052215

9.2.1.3 Description

The models predict whether the status is accepted (Y) or not approved (N) based on the P_N and P_Y, as seen in the output above. Predicted is what the letter P stands for. In the event that the P_Y exceeds the P_N, the loan approval status will be Y (approved), or vice versa.

10 Data Visualization and Report Generation

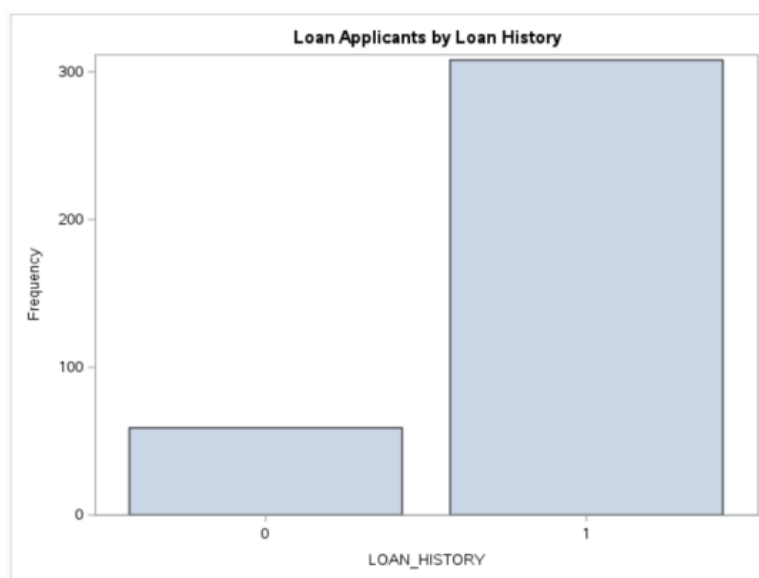
10.1 Data Visualization

10.1.1 Simple Bar Chart

10.1.1.1 Code Snippet

```
1808 /* SAS Simple Bar Chart */
1809 PROC SGPLOT DATA= LIB77994.TESTING_LAS_PREDICTED_TP77994_DS;
1810 VBAR loan_location;
1811 TITLE 'Loan Applicants by Loan Location';
1812 RUN;
```

10.1.1.2 Output



10.1.1.3 Description

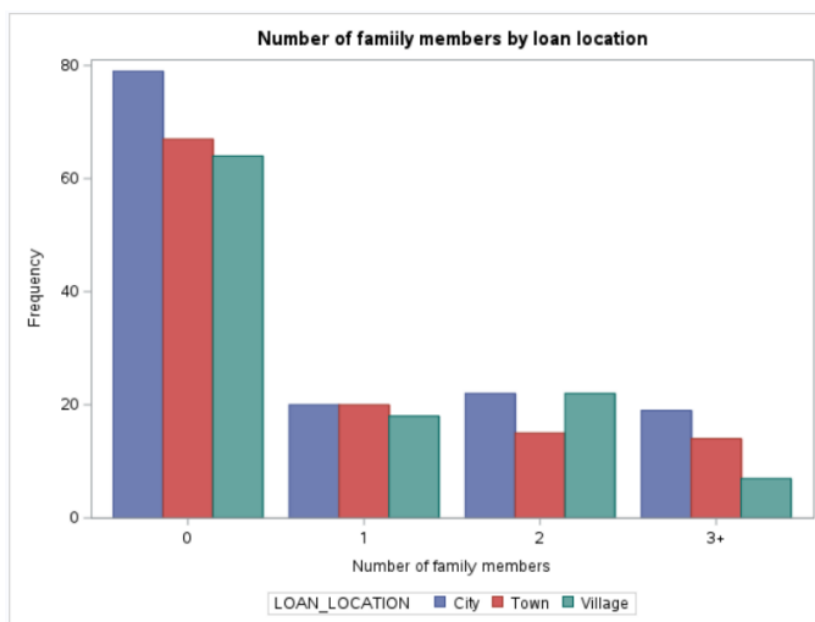
The output above is a simple bar chart that displays the loan history from the dataset. It can be seen from the bar chart that the majority of applicants had already taken out a loan before reapplying.

10.1.2 Stacked Bar Chart

10.1.2.1 Code Snippet

```
1814 /*Stacked Bar Chart
1815 The groups were stacked one above the other*/
1816 TITLE 'Number of family members by loan location';
1817 PROC SGPLOT data= LIB77702.TESTING_LAS_PREDICTED_TP77702_DS;
1818 vbar family_members /group = loan_location groupdisplay = cluster;
1819 Label family_members = 'Number of family members';
1820 RUN;
```

10.1.2.2 Output



10.1.2.3 Description

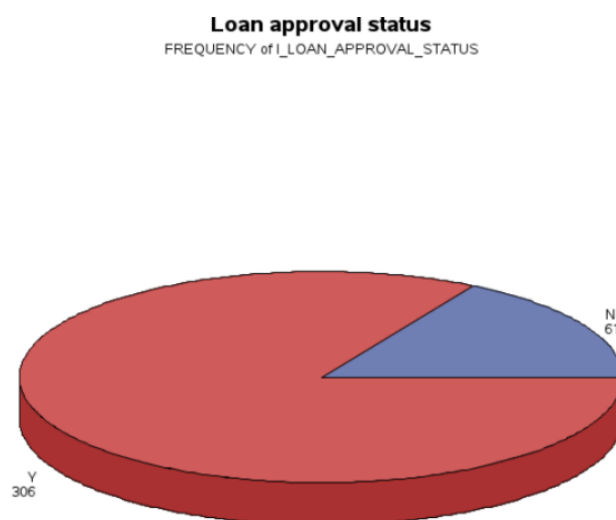
The output shown above is a stacked bar chart from the dataset that shows the number of family members by loan location. According to the stacked bar chart, the majority of applicants from cities, towns, and villages are alone.

10.1.3 Pie chart

10.1.3.1 Code Snippet

```
1822 /*Pie Chart*/
1823 TITLE 'Loan approval status';
1824 PROC GCHART data= LIB77702.TESTING_LAS_PREDICTED_TP77702_DS;
1825 pie3d I_LOAN_APPROVAL_STATUS;
1826 RUN;
1827 QUIT;
```

10.1.3.2 Output



10.1.3.3 Description

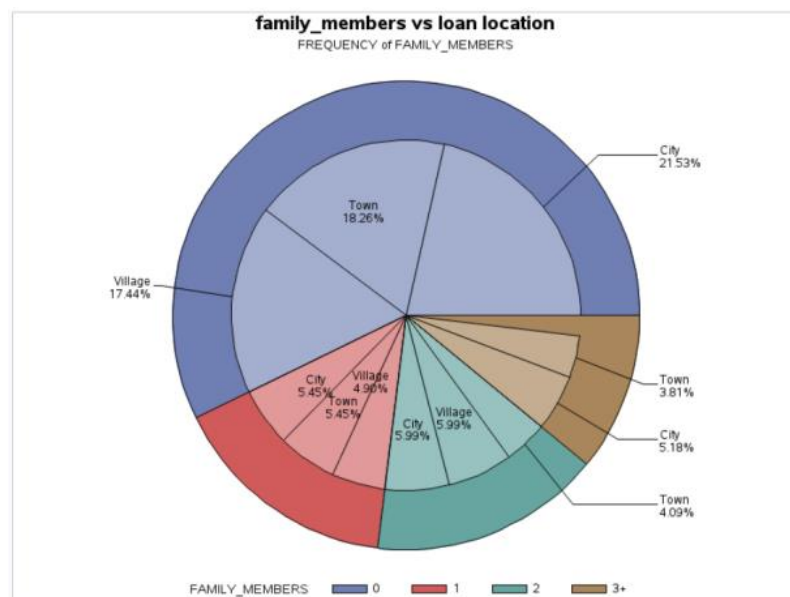
A pie chart representing the loan approval status in the dataset is shown in the output above. As can be seen from the pie chart, 356 of the total loans that were reviewed were approved, or yes, while 61 of the loans were not approved, or no.

10.1.4 Stacked Pie Chart

10.1.4.1 Code Snippet

```
1829 /*Pie Chart*/
1830 TITLE 'Loan approval status';
1831 PROC GCHART data= LIB77702.TESTING_LAS_PREDICTED_TP77702_DS;
1832 pie3d I_LOAN_APPROVAL_STATUS;
1833 RUN;
1834 QUIT;
1835
1836 GOPTIONS RESET=ALL BORDER;
1837 TITLE 'family members vs loan location';
1838 PROC GCHART DATA=LIB77702.TESTING_LAS_PREDICTED_TP77702_DS;
1839 pie family_members / detail=loan_location
1840 detail_percent=best
1841 detail_value=none
1842 detail_slice=best
1843 detail_threshold=2
1844 legend;
1845 RUN;
1846 QUIT;
```

10.1.4.2 Output



10.1.4.3 Description

Another pie chart that shows how family members are distributed among loan sites is produced using the above output. Whereas towns and villages are home to a variety of family sizes, the majority of loan applicants from cities are alone.

10.2 Report Generation using SAS ODS

10.2.1 Display Details of Predicted Loan Approvals Status

10.2.1.1 Code Snippet

```

1768 TITLE 'Display the details of the predicted loan approval status';
1769 FOOTNOTE '-----END-----';
1770
1771 /* Display the details of the predicted loan approval status */
1772
1773 PROC SQL;
1774
1775 SELECT *
1776 FROM LIB77702.TESTING_LAS_PREDICTED_78617_DS;
1777
1778 QUIT;

```

10.2.1.2 Output

Into: LOAN_APPROVAL_STATUS	Predicted Probability: LOAN_APPROVAL_STATUS=N	Predicted Probability: LOAN_APPROVAL_STATUS=Y
Y	0.156482	0.843518
Y	0.25522	0.74478
Y	0.15515	0.84485
Y	0.13906	0.86094
Y	0.327823	0.672177
Y	0.299882	0.700138
Y	0.271053	0.728947
N	0.93562	0.06438
Y	0.127025	0.872975
Y	0.23385	0.76615
Y	0.333311	0.666689

10.2.1.3 Description

The above output shows the details about the predicted probability of loan approval status of approved and not approved loans.

10.2.2 Generated Report from SAS ODS

10.2.2.1 Code Snippet

```

1780 /* Generate report using SAS ODS -Output Delivery/Display System */
1781
1782 ODS HTML CLOSE;
1783 ODS PDF CLOSE;
1784 /* Determine where the PDF is stored */
1785 ODS PDF FILE = "/home/u63686860/DAP_FT_MAR_2024_TP078617/LAS_REPORT_078617.pdf";
1786 OPTIONS NODATE;
1787 TITLE1 'Predicted Bank Loan Approval Status';
1788 TITLE2 'ASIA PACIFIC UNIVERSITY';
1789 PROC REPORT DATA = LIB77702.TESTING_LAS_PREDICTED_78617_DS NOWINDOWS;
1790 BY SME_LOAN_ID_NO;
1791 DEFINE SME_LOAN_ID_NO / GROUP 'LOAN ID';
1792 DEFINE GENDER / GROUP 'GENDER';
1793 DEFINE MARITAL_STATUS / GROUP 'MARITAL STATUS';
1794 DEFINE FAMILY_MEMBERS / GROUP 'FAMILY MEMBERS';
1795 DEFINE CANDIDATE_INCOME / GROUP 'MONTHLY INCOME';
1796 DEFINE GUARANTEE_INCOME / GROUP 'CO-APPLICANT INCOME';
1797 DEFINE LOAN_AMOUNT / GROUP 'LOAN AMOUNT';
1798 DEFINE LOAN_DURATION / GROUP 'LOAN DURATION';
1799 DEFINE LOAN_LOCATION / GROUP 'LOAN LOCATION';
1800 DEFINE LOAN_HISTORY / GROUP 'LOAN HISTORY';
1801 FOOTNOTE '-----END-----';
1802 RUN;

```


10.2.2.2 Output

Predicted Bank Loan Approval Status ASIA PACIFIC UNIVERSITY								
SME_LOAN_ID_NO=LP001055								
LOAN AMOUNT	LOAN DURATION	LOAN HISTORY	LOAN LOCATION	LOAN_APPROVAL_STATUS	From: LOAN_APPROVAL_STATUS	Into: LOAN_APPROVAL_STATUS	Predicted Probability: LOAN_APPROVAL_STATUS=N	Predicted Probability: LOAN_APPROVAL_STATUS=Y
56	360	1	Town			Y	0.271053	0.728947
-----END-----								
Predicted Bank Loan Approval Status ASIA PACIFIC UNIVERSITY								
SME_LOAN_ID_NO=LP001056								
LOAN AMOUNT	LOAN DURATION	LOAN HISTORY	LOAN LOCATION	LOAN_APPROVAL_STATUS	From: LOAN_APPROVAL_STATUS	Into: LOAN_APPROVAL_STATUS	Predicted Probability: LOAN_APPROVAL_STATUS=N	Predicted Probability: LOAN_APPROVAL_STATUS=Y
147	360	0	Village			N	0.9356197	0.0643803
-----END-----								
Predicted Bank Loan Approval Status ASIA PACIFIC UNIVERSITY								
SME_LOAN_ID_NO=LP001059								
LOAN AMOUNT	LOAN DURATION	LOAN HISTORY	LOAN LOCATION	LOAN_APPROVAL_STATUS	From: LOAN_APPROVAL_STATUS	Into: LOAN_APPROVAL_STATUS	Predicted Probability: LOAN_APPROVAL_STATUS=N	Predicted Probability: LOAN_APPROVAL_STATUS=Y
280	240	1	City			Y	0.1270251	0.8729749
-----END-----								

Predicted Bank Loan Approval Status ASIA PACIFIC UNIVERSITY

SME_LOAN_ID_NO=LP001015

LOAN ID	GENDER	MARITAL STATUS	FAMILY MEMBERS	QUALIFICATION	EMPLOYMENT	MONTHLY INCOME	CO-APPLICANT INCOME	LOAN AMOUNT
LP001015	Male	Married	0	Graduate	No	5720	0	110

LOAN DURATION	LOAN HISTORY	LOAN LOCATION	LOAN_APPROVAL_STATUS	From: LOAN_APPROVAL_STATUS
360	1	City		

Into: LOAN_APPROVAL_STATUS	Predicted Probability: LOAN_APPROVAL_STATUS=N	Predicted Probability: LOAN_APPROVAL_STATUS=Y
Y	0.1564816	0.8435184

10.2.2.3 Description

Using PROC REPORT, the output produced a report that shows loan approval forecasts depending on several variables, including income, marital status, and gender. The result displays each loan case together with the expected chances of acceptance and rejection. Generated reports are need in order to evaluate the predictions made by the models, confirm the accuracy, and effectively convey the findings to decision-makers involved in loan approval procedures.

11 Conclusion

In conclusion, by exploring the loan approval process in Lasiandra Finance Inc., it is evident that there has been a huge step towards the use of AI and machine learning in financial services. In this research, we have learned that logistic regression can predict loans awarded and understood more the importance proper data management such as filling in missing values in ensuring trustworthy predictions. The study further states that advanced data analysis greatly increases accuracy and speed while at the same time reducing risks involved with financial transactions. Moreover, SAS Macro application has made it possible for us to handle larger sets of information easily without slowing down our processing power.

Integrating complex analytical tools and techniques into banking systems represents not just an upgrade but a survival strategy within today's fast-changing world of money matters. It is therefore not shocking that such a move will serve to improve operations at Lasiandra Finance Inc apart from acting as a yardstick for creativity in provision of money-related services as well.

12 References

- Abakarim, Y., Lahby, M., & Attiou, A. (2018). Towards An Efficient Real-time Approach To Loan Credit Approval Using Deep Learning. *Deep Learning. 2018 9th International Symposium on Signal, Image, Video and Communications (ISIVC)*, 306-313.
doi:<https://doi.org/10.1109/ISIVC.2018.8709173>
- Al-qerem, A., Alnaymat, G., & Alhasan, M. (2020). Model Improvement Through Comprehensive Preprocessing For Loan Default Prediction. *International Journal of Scientific & Technology Research*, 1314-1318.
- Chen, H. (2022). Prediction and Analysis of Financial Default Loan Behavior Based on Machine Learning Model. *Computational Intelligence and Neuroscience*.
doi:<https://doi.org/10.1155/2022/7907210>
- Debikaghosh. (2024, February). *An Introduction to SAS Macro: Simplify Your Code and Boost Your Productivity*. Retrieved from Medium:
<https://medium.com/@debikaghosh91184/an-introduction-to-sas-macro-simplify-your-code-and-boost-your-productivity-536733f5dbb5>
- González-Sendino, R., Serrano, E., & Bajo, J. (2024). Mitigating bias in artificial intelligence: Fair data generation via causal models for transparent and explainable decision-making. *Future Generation Computer Systems*, 384-401.
doi:<https://doi.org/10.1016/j.future.2024.02.023>
- Hsu, W.-P. (2020, August 1). Intelligent Document Recognition on Financial Process Automation. doi:<https://doi.org/10.1109/VLSI-DAT49148.2020.9196318>.
- Jain, A., & Verma, D. (2022, november). Making Credit Underwriting Process More Accurate using ML. *International Conference on Advances in Computing, Communication and Materials (ICACCM)*, 1-4.
doi:<https://doi.org/10.1109/ICACCM56405.2022.10009117>
- Mehndiratta, N., Arora, G., & Bathla, R. (2023, May). The use of Artificial Intelligence in the Banking Industry. *International Conference on Recent Advances in Electrical, Electronics & Digital Healthcare Technologies (REEDCON)*, 588-591.
doi:<https://doi.org/10.1109/REEDCON57544.2023.10150681>.
- Mehndiratta, N., Arora, G., & Bathla, R. (2023). The use of Artificial Intelligence in the Banking Industry. *International Conference on Recent Advances in Electrical, Electronics & Digital Healthcare Technologies (REEDCON)*, 588-891.
doi:<https://doi.org/10.1109/REEDCON57544.2023.10150681>.
- Odegua, R. (2020, January 18). Predicting Bank Loan Default with Extreme Gradient Boosting. doi:ArXiv, abs/2002.02011.

- Pang, S., & Yuan, J. (2018). WT Model & Applications in Loan Platform Customer Default Prediction Based on Decision Tree Algorithms. 359-371.
doi:https://doi.org/10.1007/978-3-319-95930-6_33.
- Pavlikov, S. (2020). ON THE PROSPECTS OF NEUROCYBERNETIC (STRONG) ARTIFICIAL INTELLIGENCE IN BANKING. 33-38.
doi:<https://doi.org/10.18572/1812-3945-2020-5-33-38>.
- Rana, A., Bisht, D. S., Pandey, S., Singh, R., Chhabra, G., & Joshi, K. (2023, April). Artificial Intelligence Indulgence in Banking and Financial Sectors. *IEEE International Conference on Contemporary Computing and Communications (InC4)*, 1-5.
doi:<https://doi.org/10.1109/InC457730.2023.10263088>
- Shaheen, S. K., & Fakharany, E. (2018, October). Predictive analytics for loan default in banking sector using machine learning techniques. *International Conference on Computer Theory and Applications (ICCTA)*, 66-71.
doi:<https://doi.org/10.1109/ICCTA45985.2018.9499147>
- Shingi, G. (2020). A federated learning based approach for loan defaults prediction. *International Conference on Data Mining Workshops (ICDMW)*, 362-368.
doi:<https://doi.org/10.1109/ICDMW51313.2020.00057>
- Widiyono, Y. C., & Isa, S. M. (2020). Utilization of Data Mining to Predict Non-Performing Loan. *Advances in Science, Technology and Engineering Systems Journal*, 252-256.
doi:<https://doi.org/10.25046/aj050431>