

CHAPTER 4: DATA DICTIONARY/ METADATA

<i>Name of Variable</i>	<i>Description</i>	<i>Data Type</i>	<i>Length</i>	<i>Sample Data</i>
SME_LOAN_ID_NO	Loan registration number	Char	8	LP001002 LP001003 LP001005 LP001006 LP001008
GENDER	Applicant's gender; male or female	Varchar	6	Male Male Male Male Male
MARITAL_STATUS	Applicant's marital status; married or not married	Varchar	11	Not Married Married Married Married Not Married
FAMILY_MEMBERS	Number of applicant's family member	Varchar	2	0 2 0 3+ 2
QUALIFICATION	Applicant's education background; graduate or undergraduate	Varchar	14	Under Graduate Graduate Graduate Under Graduate Graduate
EMPLOYMENT	Applicant's employment status	Varchar	3	Yes No No Yes No
CANDIDATE_INCOME	Applicant's income	Num	8	3036 4006 12841 3200 2500

GUARANTEE_INCOME	Applicant's guarantee income	Num	8	2358 0 4196 1516 2504
LOAN_AMOUNT	Loan amount from application (thousands)	Num	8	141 267 95 158 168
LOAN_DURATION	Loan duration (months)	Num	8	360 360 120 360 240
LOAN_HISTORY	Applicant's loan history; yes or no	Num	8	1 1 0 1 0
LOAN_LOCATION	Location that applicant applied; city or town	Varchar	7	Town City City City Village
LOAN_APPROVAL_STATUS	Approval accept or reject? ;yes or no	Char	1	Y N N Y N

4.1 Uploads dataset

Screenshot(s)

The screenshot displays the SAS Studio interface. On the left, the 'Files (Home)' pane shows a folder named 'DAP_FT_MAR_2024_TP078601' which has been highlighted with a red box. Below this folder, several files are listed, including 'TESTING_DS.csv' and 'TRAINING_DS.csv', both of which are also highlighted with red boxes. On the right, the 'Log' window shows the execution log, which includes the following text:

```

1 /*****
2 Developer name: Mr.NGOO CHEN THONG
3 Job position: Data Scientist, APU SDN BHD
4 Program name: mysasprogram.sas
5 Description: Loan application status prediction - (1 - 2 lines)
6 Date first written: Fri,26-Apr-2024
7 Date last updated: Fri,26-Apr-2024
8 Folder name: MY_DAP_FT_MAR_2024_TP078601
9 Library name: LIB78601
10
11 *****/

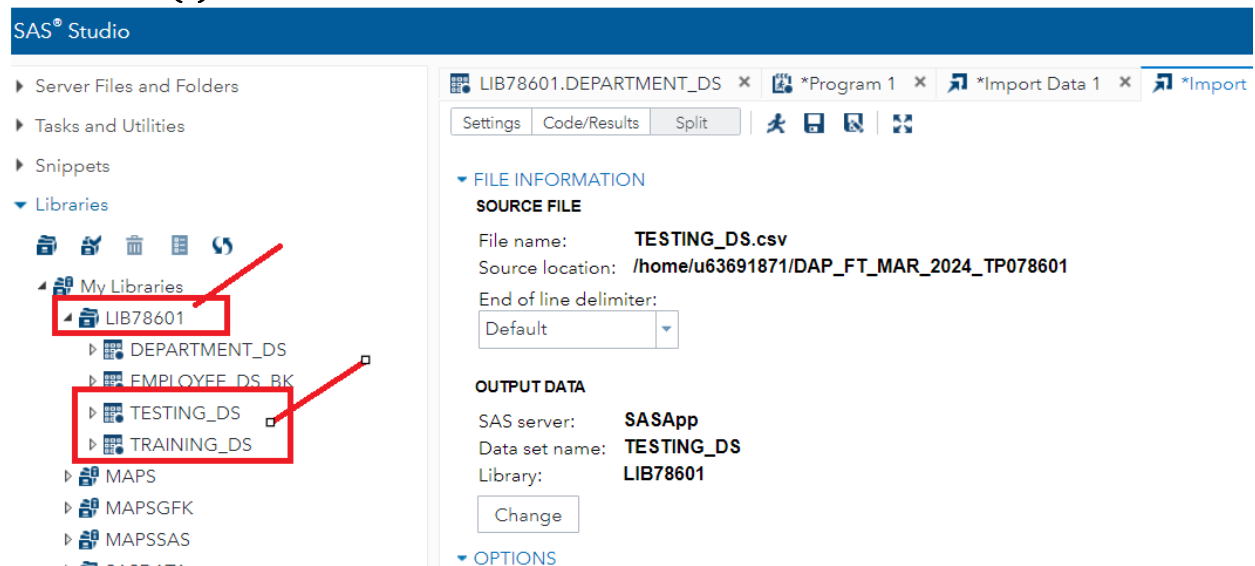
```

Description

The first thing we have to do is upload the datasets to SAS, the datasets includes training and testing dataset. As the screenshot shows, a folder named DAP_FT_MAR_TP078601 has been created for stored the relevant documents for this task.

4.2 Transfer the datasets from the Project folder to the newly created permanent SAS library – LIB78601

Screenshot(s)



Descriptions

A new library has been created with name LIB78601 to the SAS system, this library mainly purpose is to save the dataset used in this project. By saving these datasets we need to this library, we doesn't need to upload the datasets required for everytime we need it.

4.3 Display the structure of the dataset – LIB78601

```
1  /*****
2  Developer name: Mr.NGOO CHEN THONG
3  Job position: Data Scientist, APU SDN BHD
4  Program name: mydap_project_tp078601.sas
5  Description: Loan application status prediction - (1 - 2 lines)
6  Date first written: Fri,26-Apr-2024
7  Date last updated: Fri,26-Apr-2024
8  Folder name: MY_DAP_FT_MAR_2024_TP078601
9  Library name: LIB78601
10 *****/
11
12 /*SAS Codes to display the data dictionary of LIB78601.TRAINING_DS */
13 PROC SQL;
14 DESCRIBE TABLE LIB78601.TRAINING_DS;
15 RUN;
```

```
create table LIB78601.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.
);
```

Description

By display the structure of the dataset, we can easily to observe the properties of each attributes including the data type and format.

4.4 View the structure of dataset – LIB78601.TESTING_DS

SAS Code

```
PROC CONTENTS DATA = LIB78601.TRAINING_DS;
RUN;
```

Screenshot(s)

The CONTENTS Procedure			
Data Set Name	LIB78601.TRAINING_DS	Observations	614
Member Type	DATA	Variables	13
Engine	V9	Indexes	0
Created	04/26/2024 15:53:25	Observation Length	96
Last Modified	04/26/2024 15:53:25	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/u63691871/DAP_FT_MAR_2024_TP078601/training_ds.sas7bdat
Release Created	9.0401M7
Host Created	Linux
Inode Number	1757447410
Access Permission	rw-r--r--
Owner Name	u63691871
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.

Description

By using PROC CONTENTS from SAS, we can see that metadata about the dataset easily. This method helps us to visualize the properties of a dataset in convenient.

CHAPTER 6: ANALYSIS OF THE VARIABLES

6.1 Univariate Analysis of the variables found in the dataset LIB78601.TRAINING_DS

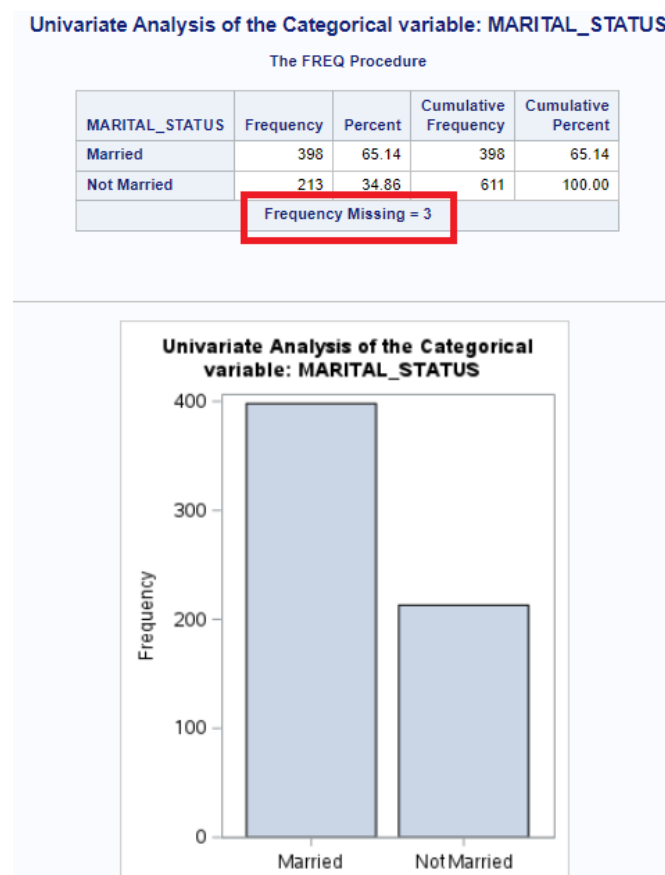
6.1.1 Univariate analysis of the categorical variables found in the dataset.

6.1.1.1 Univariate analysis of the categorical variables – MARITAL_STATUS.

SAS Code

```
22 TITLE ' Univariate Analysis of the Categorical variable: MARITAL_STATUS';
23 PROC FREQ DATA = LIB78601.TRAINING_DS;
24 TABLE marital_status;
25 RUN;
26 ODS GRAPHIC / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
27
28 PROC SGPLOT DATA = LIB78601.TRAINING_DS;
29 VBAR marital_status;
30 TITLE ' Univariate Analysis of the Categorical variable: MARITAL_STATUS';
31 RUN;
```

Screenshot(s)



Description

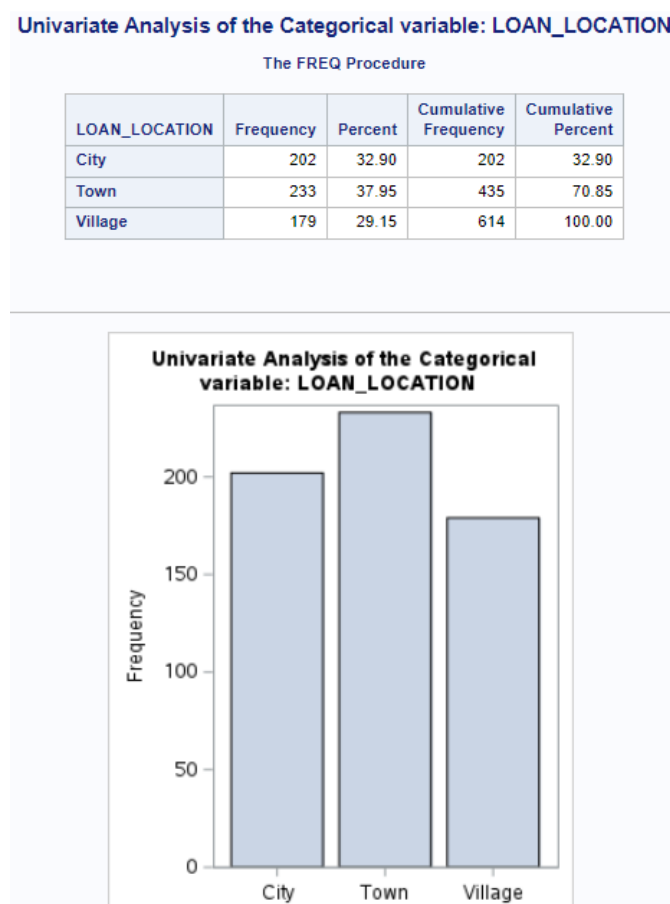
We can see that most of the loan applicants have married. Furthermore, there is total 3 missing values in the dataset.

6.1.1.2 Univariate analysis of the categorical variables – LOAN_LOCATION.

SAS Code

```
34 TITLE ' Univariate Analysis of the Categorical variable: LOAN_LOCATION';
35 PROC FREQ DATA = LIB78601.TRAINING_DS;
36 TABLE loan_location;
37 RUN;
38 ODS GRAPHIC / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
39
40 PROC SGPLOT DATA = LIB78601.TRAINING_DS;
41 VBAR loan_location;
42 TITLE ' Univariate Analysis of the Categorical variable: LOAN_LOCATION';
43 RUN;
```

Screenshot(s)



Description

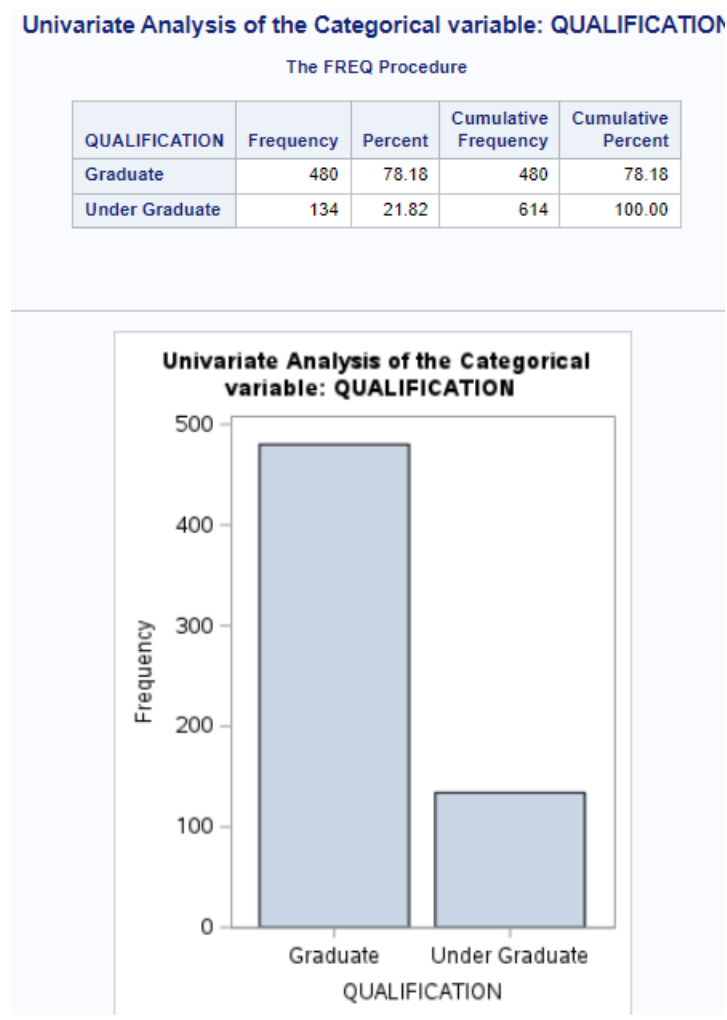
The distribution of loan location for applicants is nearly identical distribution, means that the applicants are coming from city, town and village, no skew will be found in the visualization. Furthermore, there is no missing value in this dataset.

6.1.1.3 Univariate analysis of the categorical variables – QUALIFICATION.

SAS Code

```
46 TITLE ' Univariate Analysis of the Categorical variable: QUALIFICATION';
47 PROC FREQ DATA = LIB78601.TRAINING_DS;
48 TABLE qualification;
49 RUN;
50 ODS GRAPHIC / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
51
52 PROC SGPLOT DATA = LIB78601.TRAINING_DS;
53 VBAR qualification;
54 TITLE ' Univariate Analysis of the Categorical variable: QUALIFICATION';
55 RUN;
```

Screenshot(s)



Description

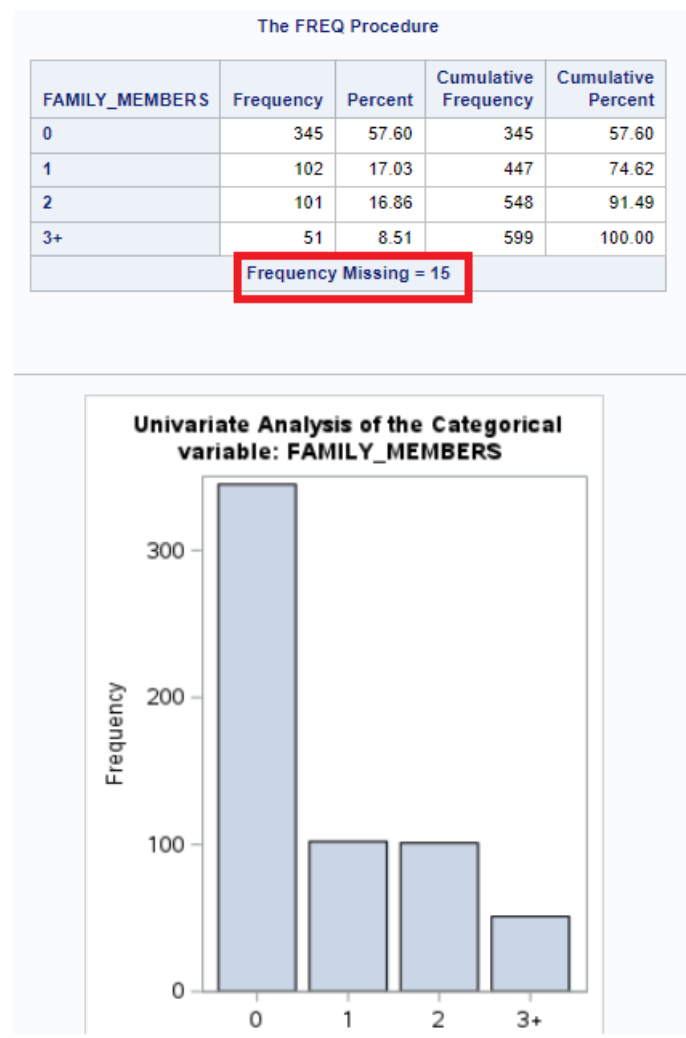
From the screenshot, we can see that most of the applicants have graduate in qualification, only around 21% of them have undergraduate for qualification. Furthermore, there is no missing value in this dataset.

6.1.1.4 Univariate analysis of the categorical variables – FAMILY_MEMBERS.

SAS Code

```
58 TITLE ' Univariate Analysis of the Categorical variable: FAMILY_MEMBERS';  
59 PROC FREQ DATA = LIB78601.TRAINING_DS;  
60 TABLE family_members;  
61 RUN;  
62 ODS GRAPHIC / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;  
63  
64 PROC SGPLOT DATA = LIB78601.TRAINING_DS;  
65 VBAR family_members;  
66 TITLE ' Univariate Analysis of the Categorical variable: FAMILY_MEMBERS';  
67 RUN;
```

Screenshot(s)



Description

Most of the applicants has 0 for the number of family members, there applicants with 1 member have almost same number with those have family numbers 2. Applicants with 3 or more family members only contribute 8.51% to this dataset. There are 15 missing values in this dataset.

6.1.2 Univariate analysis of the continuous variables found in the dataset – LIB78601.TRAINING_DS.

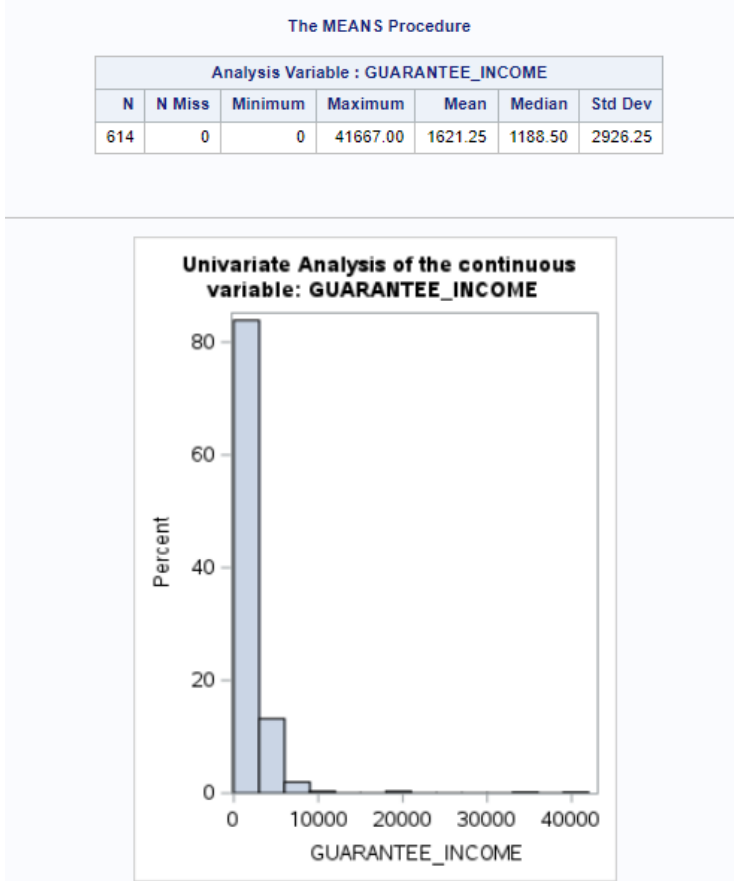
6.1.2.1 Univariate analysis of the continuous variables – GUARANTEE_INCOME.

SAS Code

```
93 TITLE ' Univariate Analysis of the continuous variable: GUARANTEE_INCOME';  
94 PROC MEANS DATA = LIB78601.TRAINING_DS N NMISS MIN MAX MEAN MEDIAN STD;  
95 VAR guarantee_income;  
96 RUN;  
97 ODS GRAPHIC / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;  
98 PROC SGPLOT DATA = LIB78601.TRAINING_DS;  
99 HISTOGRAM guarantee_income;  
100 TITLE ' Univariate Analysis of the continuous variable: GUARANTEE_INCOME';  
101 RUN;
```

Screenshot(s)

Univariate Analysis of the continuous variable: GUARANTEE_INCOME



Description

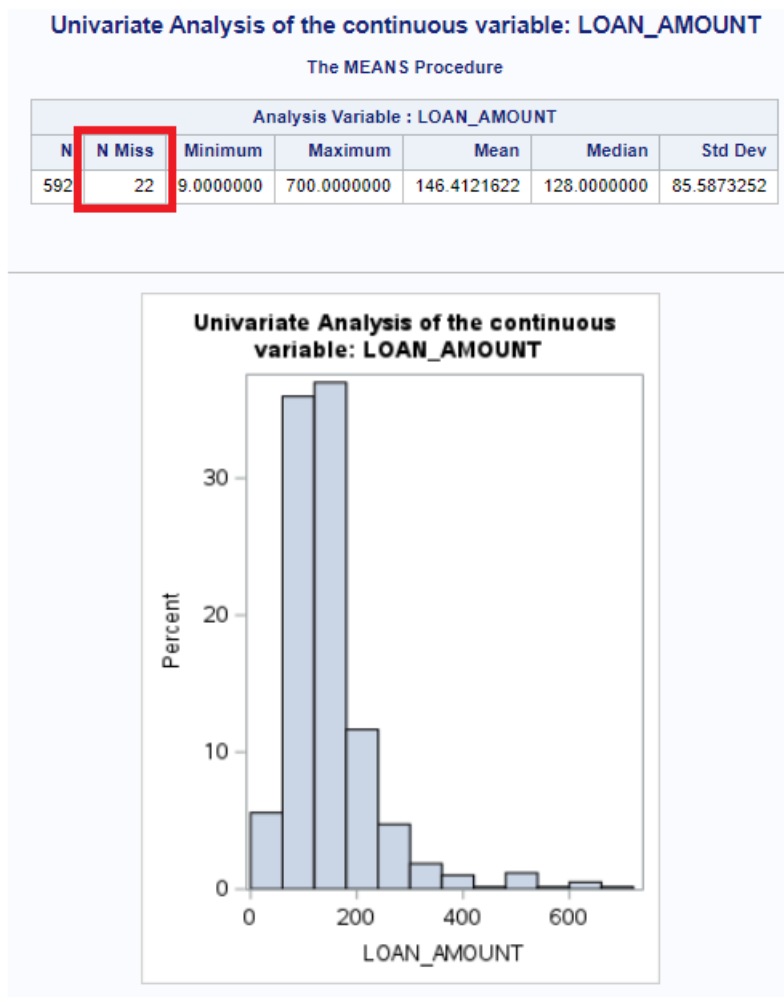
From the chart, we can easily see that the guaranteed income from the applicants is concentrated below 10,000. From this view, we can see that people with low income has higher chance to apply loan. Furthermore, there is no missing value in this dataset.

6.1.2.2 Univariate analysis of the continuous variables – LOAN_AMOUNT.

SAS Code

```
105 TITLE ' Univariate Analysis of the continuous variable: LOAN_AMOUNT';
106 PROC MEANS DATA = LIB78601.TRAINING_DS N NMISS MIN MAX MEAN MEDIAN STD;
107 VAR loan_amount;
108 RUN;
109 ODS GRAPHIC / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
110 PROC SGPLOT DATA = LIB78601.TRAINING_DS;
111 HISTOGRAM loan_amount;
112 TITLE ' Univariate Analysis of the continuous variable: LOAN_AMOUNT';
113 RUN;
```

Screenshot(s)



Description

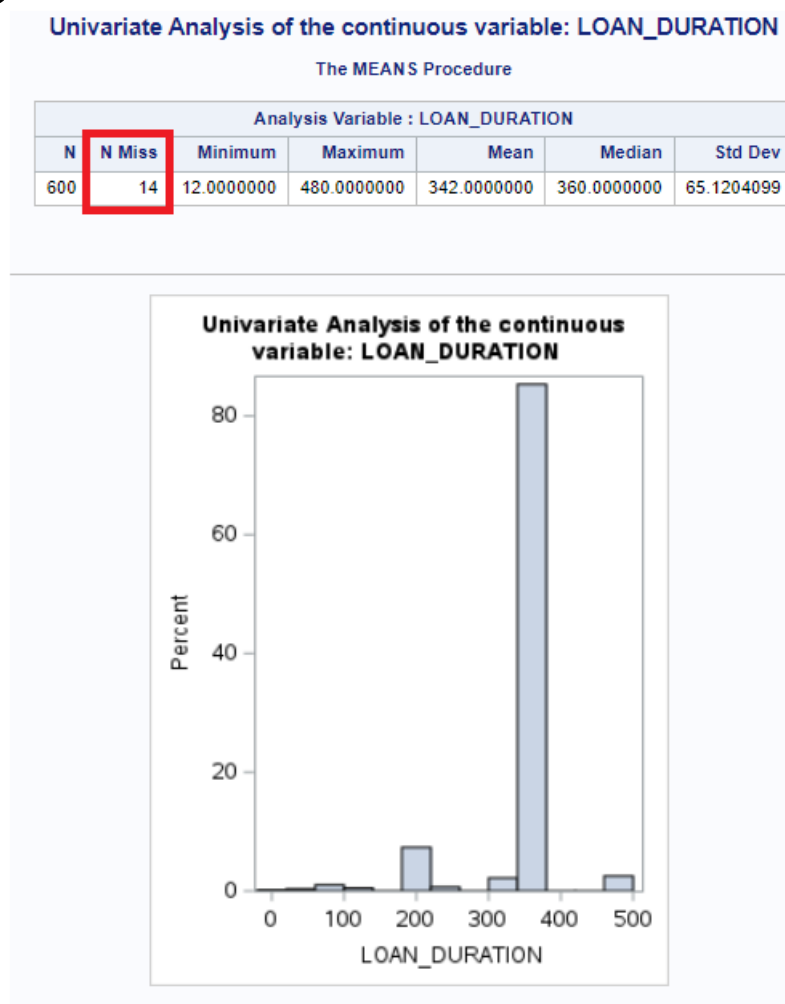
The distribution for loan amount has right skew distribution, where the mean and median are 146 and 128 respectively. The percentage of loan amount become smaller starts from 300. Furthermore, there are 22 missing values in this dataset.

6.1.2.3 Univariate analysis of the continuous variables – LOAN_DURATION.

SAS Code

```
117 TITLE ' Univariate Analysis of the continuous variable: LOAN_DURATION';
118 PROC MEANS DATA = LIB78601.TRAINING_DS N NMISS MIN MAX MEAN MEDIAN STD;
119 VAR loan_duration;
120 RUN;
121 ODS GRAPHIC / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
122 PROC SGPLOT DATA = LIB78601.TRAINING_DS;
123 HISTOGRAM loan_duration;
124 TITLE ' Univariate Analysis of the continuous variable: LOAN_DURATION';
125 RUN;
```

Screenshot(s)



Description

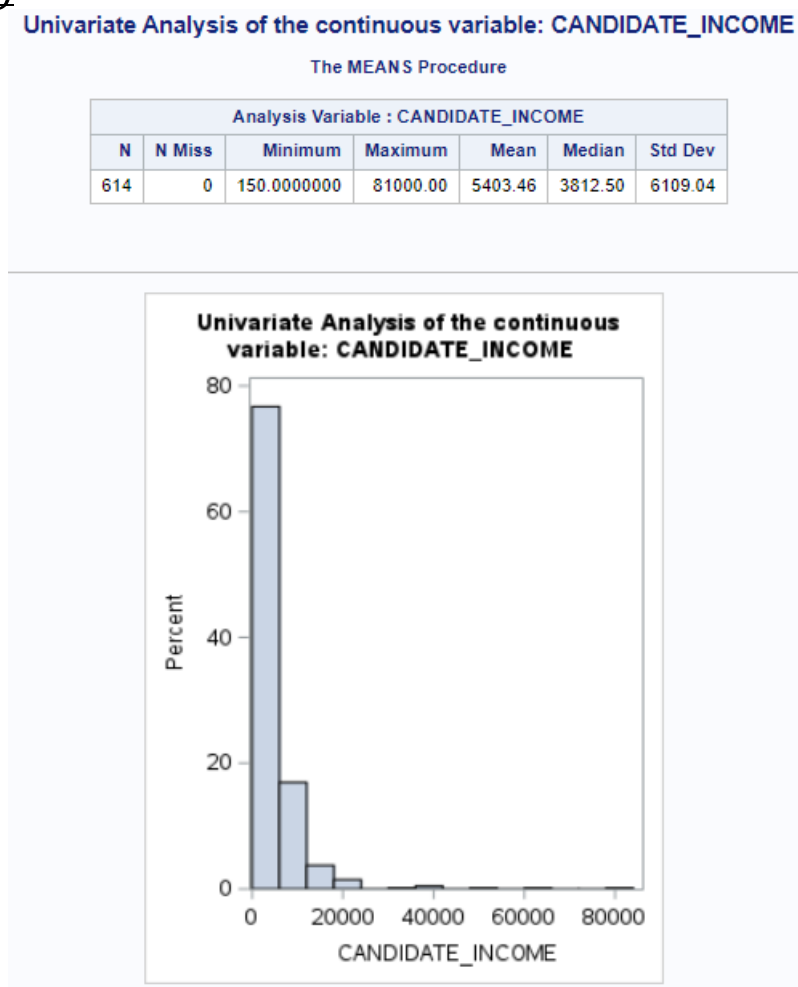
By looking the dataset and the visualization for loan duration attribute, we can see that most of the applicants apply for 360 months, which is 30 years. The minimum and maximum loan duration are 12 and 480 months respectively. Furthermore, there are 14 missing values in this dataset.

6.1.2.4 Univariate analysis of the continuous variables – CANDIDATE_INCOME.

SAS Code

```
129 TITLE ' Univariate Analysis of the continuous variable: CANDIDATE_INCOME';
130 PROC MEANS DATA = LIB78601.TRAINING_DS N NMISS MIN MAX MEAN MEDIAN STD;
131 VAR candidate_income;
132 RUN;
133 ODS GRAPHIC / RESET WIDTH = 3.0 IN HEIGHT = 4.0 IN IMAGEMAP;
134 PROC SGPLOT DATA = LIB78601.TRAINING_DS;
135 HISTOGRAM candidate_income;
136 TITLE ' Univariate Analysis of the continuous variable: CANDIDATE_INCOME';
137 RUN;
```

Screenshot(s)



Description

More than 90% of the applicants have income below 10,000, and the percentage become lower when income increase. Furthermore, there is no missing value in this dataset.

6.2 Bivariate analysis of the variables found in the dataset – LIB78601.TRAINING_DS.

6.2.1 Bivariate analysis of the continuous variables (Categorical vs Categorical).

6.2.1.1 Bivariate analysis of the continuous variables (Categorical - QUALIFICATION vs Categorical - EMPLOYMENT).

SAS Code

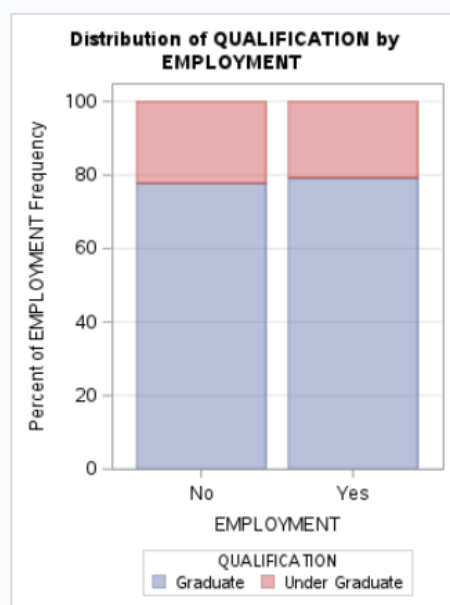
```
141 TITLE1 ' Bivariate analysis of the variables:';  
142 TITLE2 ' Categorical variable [QUALIFICATION] VS Categorical variable [EMPLOYMENT] ';  
143 PROC FREQ DATA = LIB78601.TRAINING_DS;  
144 TABLE qualification*employment/  
145 PLOTS = FREQPLOT(TWOWAY = STACKED SCALE = GROUPPCT);  
146 RUN;
```

Screenshot(s)

**Bivariate analysis of the variables:
Categorical variable [QUALIFICATION] VS Categorical variable [EMPLOYMENT]**

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of QUALIFICATION by EMPLOYMENT			
	QUALIFICATION	EMPLOYMENT		Total
		No	Yes	
	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			



Description

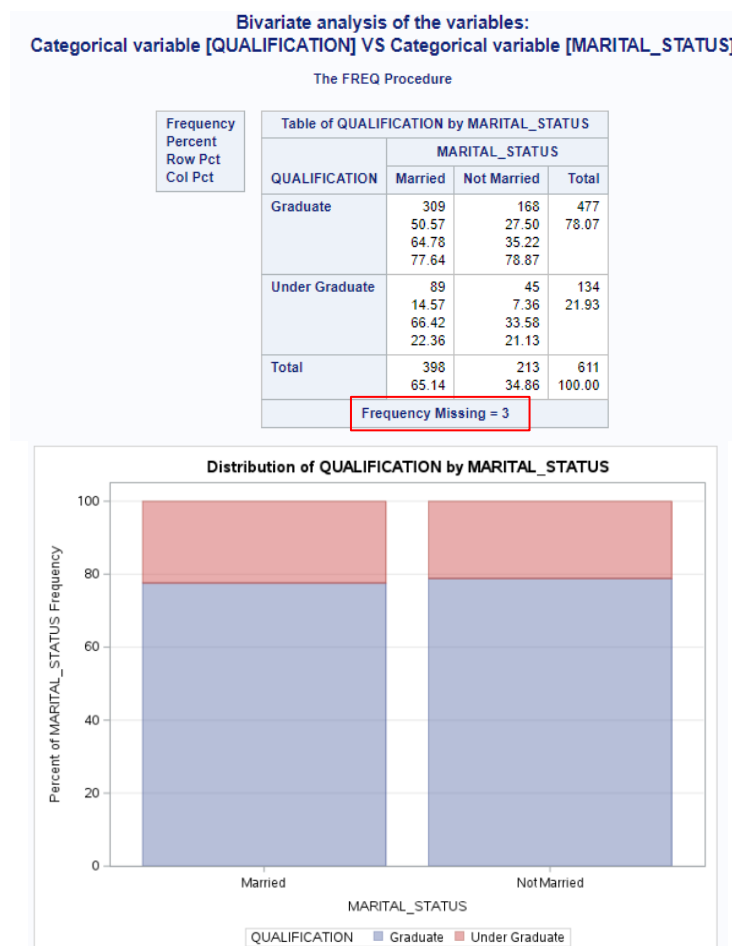
From the table, we can observe that most of the applicants have graduate qualification but didn't have employment. Another finding from this chart is, graduate and undergraduate qualification applicants have around 80% has no employment. Furthermore, there 32 missing values in this dataset.

6.2.1.2 Bivariate analysis of the continuous variables (Categorical - QUALIFICATION vs Categorical – MARITAL STATUS).

SAS Code

```
150 TITLE1 ' Bivariate analysis of the variables:';
151 TITLE2 ' Categorical variable [QUALIFICATION] VS Categorical variable [MARITAL_STATUS] ';
152 PROC FREQ DATA = LIB78601.TRAINING_DS;
153 TABLE qualification*marital_status/
154 PLOTS = FREQPLOT(TWOWAY = STACKED SCALE = GROUPPCT);
155 RUN;
```

Screenshot(s)



Description

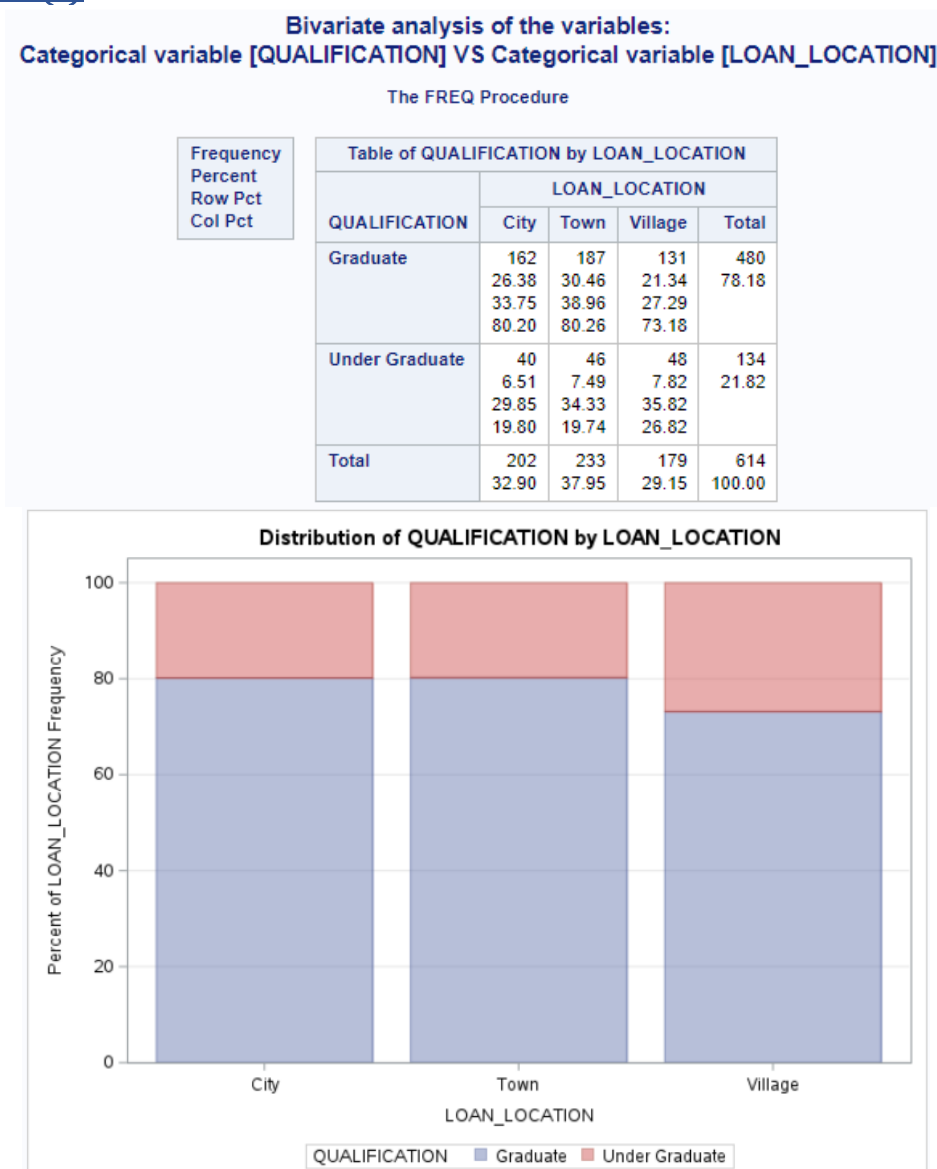
Most of the applicants have married for their marital status, this may be one of the reasons that why they trying to apply bank loan since they need more financial support to cover their daily needs. Furthermore, there are 3 missing values in this dataset.

6.2.1.3 Bivariate analysis of the continuous variables (Categorical - QUALIFICATION vs Categorical – LOAN_LOCATION).

SAS Code

```
159 TITLE1 ' Bivariate analysis of the variables: ';
160 TITLE2 ' Categorical variable [QUALIFICATION] VS Categorical variable [LOAN_LOCATION] ';
161 PROC FREQ DATA = LIB78601.TRAINING_DS;
162 TABLE qualification*loan_location/
163 PLOTS = FREQPLOT(TWOWAY = STACKED SCALE = GROUPPCT);
164 RUN;
```

Screenshot(s)



Description

As above, we have seen that the distribution of loan location is quite identically for city, village, and town. The ratio of graduate in village is minor lesser than others, where other two places have around 80% for graduate applicants. Furthermore, there is no missing value in this dataset.

6.2.2 Bivariate analysis of the continuous variables (Categorical vs Continuous).

6.2.2.1 Bivariate analysis of the continuous variables (Categorical – LOAN_LOCATION vs Continuous – LOAN_AMOUNT).

SAS Code

```
168 TITLE1 ' Bivariate analysis of the variables: ';
169 TITLE2 ' Categorical variable [LOAN_LOCATION] VS Continuous variable [LOAN_AMOUNT] ';
170 PROC MEANS DATA = LIB78601.TRAINING_DS;
171 CLASS loan_location; /* Categorical */
172 VAR loan_amount; /* Continuous */
173 RUN;
```

Screenshot(s)

Bivariate analysis of the variables:
Categorical variable [LOAN_LOCATION] VS Continuous variable [LOAN_AMOUNT]

The MEANS Procedure

Analysis Variable : LOAN_AMOUNT						
LOAN_LOCATION	N Obs	N	Mean	Std Dev	Minimum	Maximum
City	202	191	142.1989529	94.5471322	9.0000000	700.0000000
Town	233	228	145.5043860	81.6682608	25.0000000	600.0000000
Village	179	173	152.2601156	80.2332825	40.0000000	570.0000000

Description

For loan amount, applicants from village have the highest mean in these 3 loan locations, where the locations that have highest standard deviation will be city, this means that loan amount from city is wider in the distribution chart.

6.2.2.2 Bivariate analysis of the continuous variables (Categorical - QUALIFICATION vs Continuous – GUARANTEE_INCOME).

SAS Code

```
177 TITLE1 ' Bivariate analysis of the variables: ';
178 TITLE2 ' Categorical variable [QUALIFICATION] VS Continuous variable [GUARANTEE_INCOME] ';
179 PROC MEANS DATA = LIB78601.TRAINING_DS;
180 CLASS qualification; /* Categorical */
181 VAR guarantee_income; /* Continuous */
182 RUN;
```

Screenshot(s)

Bivariate analysis of the variables:
Categorical variable [QUALIFICATION] VS Continuous variable [GUARANTEE_INCOME]

The MEANS Procedure

Analysis Variable : GUARANTEE_INCOME						
QUALIFICATION	N Obs	N	Mean	Std Dev	Minimum	Maximum
Graduate	480	480	1717.47	3230.97	0	41667.00
Under Graduate	134	134	1276.54	1310.34	0	7101.00

Description

We can see that either graduate or undergraduate qualifications, there still have some applicants have 0 income, this means that some of them have totally no income or not employment. But for those who have income, the mean of income from graduate qualifications is higher than undergraduate.

6.2.2.3 Bivariate analysis of the continuous variables (Categorical – LOAN_DURATION vs Continuous – LOAN_AMOUNT).

SAS Code

```
187 TITLE1 ' Bivariate analysis of the variables:';
188 TITLE2 ' Categorical variable [LOAN_DURATION] VS Continuous variable [LOAN_AMOUNT] ';
189 PROC MEANS DATA = LIB78601.TRAINING_DS;
190 CLASS loan_duration; /* Categorical */
191 VAR loan_amount; /* Continuous */
192 RUN;
```

Screenshot(s)

Bivariate analysis of the variables:
Categorical variable [LOAN_DURATION] VS Continuous variable [LOAN_AMOUNT]

The MEANS Procedure

Analysis Variable : LOAN_AMOUNT						
LOAN_DURATION	N Obs	N	Mean	Std Dev	Minimum	Maximum
12	1	1	111.0000000	.	111.0000000	111.0000000
36	2	2	117.5000000	53.0330086	80.0000000	155.0000000
60	2	2	140.0000000	21.2132034	125.0000000	155.0000000
84	4	4	132.2500000	31.8786763	105.0000000	172.0000000
120	3	3	22.3333333	4.6188022	17.0000000	25.0000000
180	44	42	147.5238095	108.7678750	40.0000000	600.0000000
240	4	3	118.3333333	79.1096286	50.0000000	205.0000000
300	13	13	185.1538462	178.1169308	60.0000000	700.0000000
360	512	493	147.2454361	78.9102004	9.0000000	600.0000000
480	15	15	151.8000000	141.6576356	63.0000000	650.0000000

Description

The highest option chosen for loan duration will be 360 months, and the option with second highest chosen was 180 months, but the difference between them is large. The mean of highest loan amount from loan duration was 300 months with mean 185,000 and the standard deviation of loan duration with 120 months has only 4.61, this means that the distribution of this group of applicants is heavily concentrate.

6.3 Analysis of the variables found in the dataset

LIB78601.TESTING_DS

6.3.1 Univariate analysis of the variables found in the dataset –

LIB78601.TESTING_DS.

6.3.1.1 Univariate analysis of the categorical variables found in the dataset

using SAS MACRO.

Introduction to SAS MACRO

SAS MACRO is a feature in SAS programming that makes it possible to automate and customize repetitive tasks. Reusable code blocks can be formed by using the %macro and %mend statements to define macros. Their support for parameterization, conditional logic, looping, and the usage of macro variables makes SAS programming simpler as well as efficient. Macros are a useful tool for SAS developers and analysts because they improve code readability, expedite procedures, and allow code reuse.

SAS Code

```
196 /* Macro begins here */
197 OPTIONS MCOMPILENOTE=ALL;
198
199 %MACRO UVA_CATE_VARI(ptitle, pdataset, pcate_vari);
200 TITLE &ptitle;
201 PROC FREQ DATA = &pdataset;
202 TABLE &pcate_vari;
203 RUN;
204 %MEND UVA_CATE_VARI;
205 /* Macro end here */

208 /* Call the SAS Macro - UVA_CATE_VARI */
209 %UVA_CATE_VARI('Univariate Analysis of the Categorical Variable - FAMILY_MEMBERS', LIB78601.TESTING_DS, FAMILY_MEMBERS);
210 %UVA_CATE_VARI('Univariate Analysis of the Categorical Variable - GENDER', LIB78601.TESTING_DS, GENDER);
211 %UVA_CATE_VARI('Univariate Analysis of the Categorical Variable - LOAN_LOCATION', LIB78601.TESTING_DS, LOAN_LOCATION);
212 %UVA_CATE_VARI('Univariate Analysis of the Categorical Variable - QUALIFICATION', LIB78601.TESTING_DS, QUALIFICATION);
213 %UVA_CATE_VARI('Univariate Analysis of the Categorical Variable - MARITAL_STATUS', LIB78601.TESTING_DS, MARITAL_STATUS);
214 %UVA_CATE_VARI('Univariate Analysis of the Categorical Variable - EMPLOYMENT', LIB78601.TESTING_DS, EMPLOYMENT);
```

Screenshot(s)

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 - 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 - 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

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 - 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 - 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				

Description

We can observe the data of univariate analysis to categorical data using SAS Macro easily. The missing values for variables family members, gender, and employment are 10, 11, and 23 respectively. For family members, 56% of the applicants didn't have any family members. Furthermore, the distribution of male is greater than female. In addition, the number of graduate applicants is more than undergraduate. Around 63% of the applicants is married.

6.3.1.2 Univariate analysis of the continuous variables found in the dataset using SAS MACRO.

SAS Code

```
217 /* Macro begins here */
218 OPTIONS MCOMPILENOTE=ALL;
219
220 %MACRO UVA_CONT_VARI(ptitle, pdataset, pcont_vari);
221 TITLE &ptitle;
222 PROC MEANS DATA = &pdataset N NMISS MIN MAX MEAN MEDIAN STD;
223 VAR &pcont_vari;
224 RUN;
225 %MEND UVA_CONT_VARI;
226 /* Macro end here */

228 /* Call the SAS Macro - UVA_CONT_VARI */
229 %UVA_CONT_VARI('Univariate Analysis of the Continuous Variable - CANDIDATE_INCOME', LIB78601.TESTING_DS, CANDIDATE_INCOME);
230 %UVA_CONT_VARI('Univariate Analysis of the Continuous Variable - GUARANTEE_INCOME', LIB78601.TESTING_DS, GUARANTEE_INCOME);
231 %UVA_CONT_VARI('Univariate Analysis of the Continuous Variable - LOAN_AMOUNT', LIB78601.TESTING_DS, LOAN_AMOUNT);
232 %UVA_CONT_VARI('Univariate Analysis of the Continuous Variable - LOAN_DURATION', LIB78601.TESTING_DS, LOAN_DURATION);
233
```

Screenshot(s)

Univariate Analysis of the Continuous 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 Continuous 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 Continuous 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 Continuous 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

Description

We can observe the data of univariate analysis to continuous data using SAS Macro easily. The missing values for variables loan amount and loan duration are 5 and 6 respectively. There is total 367 of observations record in this dataset. For applicants' income, the mean and standard deviation value are 4,805.60 and 4,910.69 respectively. The maximum of loan amount in this dataset will be 550 thousand.

6.3.2 Bivariate analysis of the variables found in the dataset – LIB78601.TESTING_DS.

6.3.2.1 Bivariate analysis of the variables (Categorical vs Categorical) using the SAS MACRO.

SAS Code

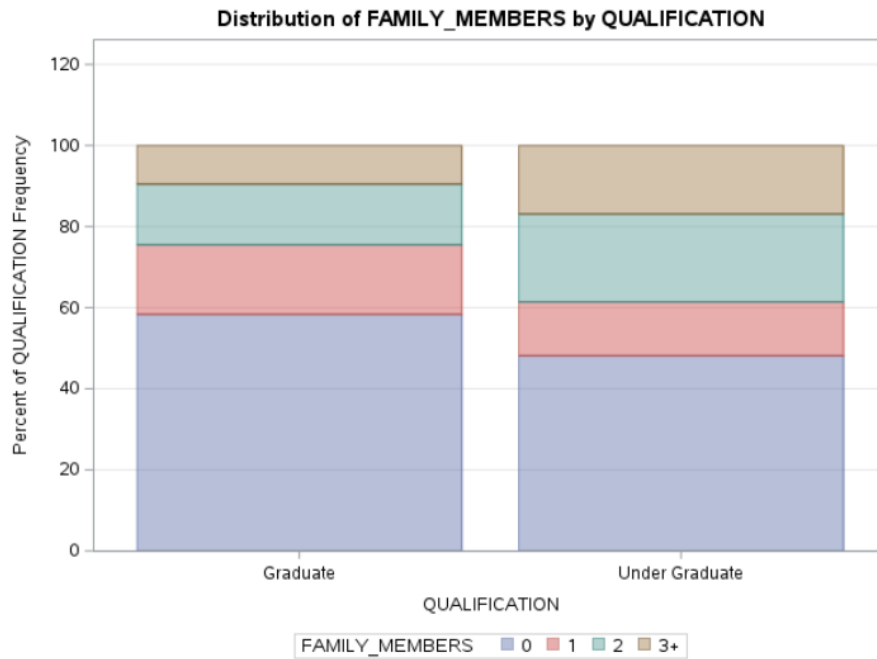
```
236 /* Macro begins here */
237 OPTIONS MCOMPILENOTE=ALL;
238
239 %MACRO BVA_CATE_CATE(ptitle1, ptitle2, pdataset, pcate_vari1, pcate_vari2);
240 TITLE &ptitle1;
241 TITLE2 &ptitle2;
242 PROC FREQ DATA = &pdataset;
243 TABLE &pcate_vari1 * &pcate_vari2/
244 PLOTS = FREQPLOT(TWOWAY = STACKED SCALE = GROUPPCT);
245 RUN;
246 %MEND BVA_CATE_CATE;
247 /* Macro end here */
248
249 /* Call the SAS Macro - bVA_CATE_CATE */
250 %BVA_CATE_CATE('Bivariate analysis of the variables:',
251 ' Categorical variable [FAMILY_MEMBERS] VS Categorical variable [QUALIFICATION] ',
252 LIB78601.TESTING_DS, FAMILY_MEMBERS, QUALIFICATION);
253
254 %BVA_CATE_CATE('Bivariate analysis of the variables:',
255 ' Categorical variable [MARITAL_STATUS] VS Categorical variable [LOAN_LOCATION] ',
256 LIB78601.TESTING_DS, MARITAL_STATUS, LOAN_LOCATION);
257
258 %BVA_CATE_CATE('Bivariate analysis of the variables:',
259 ' Categorical variable [GENDER] VS Categorical variable [EMPLOYMENT] ',
260 LIB78601.TESTING_DS, GENDER, EMPLOYMENT);
---
```

Screenshot(s)

Bivariate analysis of the variables:
Categorical variable [FAMILY_MEMBERS] VS Categorical variable [QUALIFICATION]

The FREQ Procedure

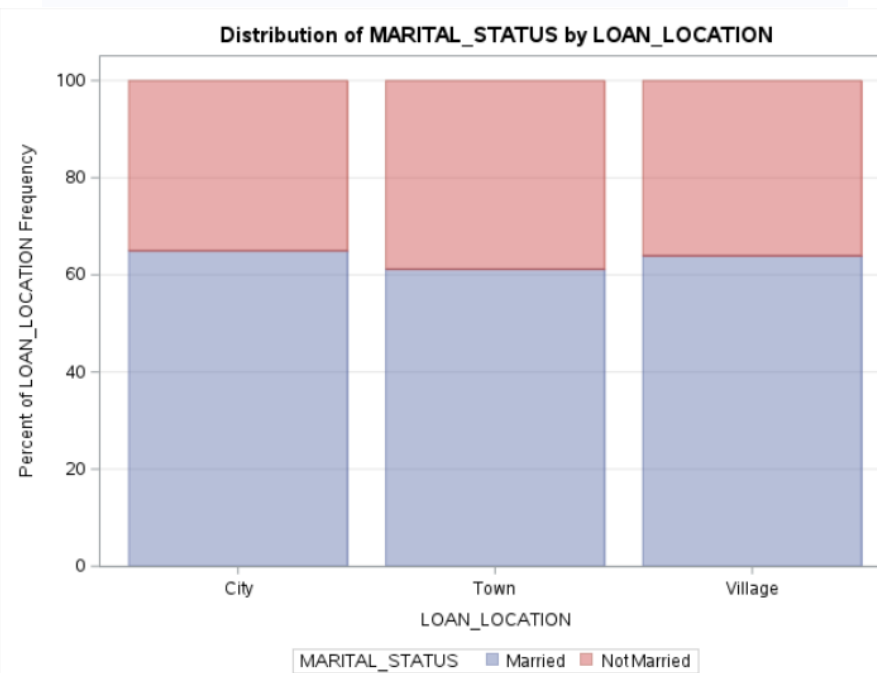
Frequency Percent Row Pct Col Pct	Table of FAMILY_MEMBERS by QUALIFICATION			
	FAMILY_MEMBERS	QUALIFICATION		
		Graduate	Under Graduate	Total
	0	160 44.82 80.00 58.39	40 11.20 20.00 48.19	200 56.02
	1	47 13.17 81.03 17.15	11 3.08 18.97 13.25	58 16.25
	2	41 11.48 69.49 14.96	18 5.04 30.51 21.69	59 16.53
	3+	26 7.28 65.00 9.49	14 3.92 35.00 16.87	40 11.20
	Total	274 76.75	83 23.25	357 100.00
	Frequency Missing = 10			



Bivariate analysis of the variables:
Categorical variable [MARITAL_STATUS] VS Categorical variable [LOAN_LOCATION]

The FREQ Procedure

	Table of MARITAL_STATUS by LOAN_LOCATION			
	LOAN_LOCATION			
MARITAL_STATUS	City	Town	Village	Total
Married	91	71	71	233
	24.80	19.35	19.35	63.49
	39.06	30.47	30.47	
	65.00	61.21	63.96	
Not Married	49	45	40	134
	13.35	12.26	10.90	36.51
	36.57	33.58	29.85	
	35.00	38.79	36.04	
Total	140	116	111	367
	38.15	31.61	30.25	100.00

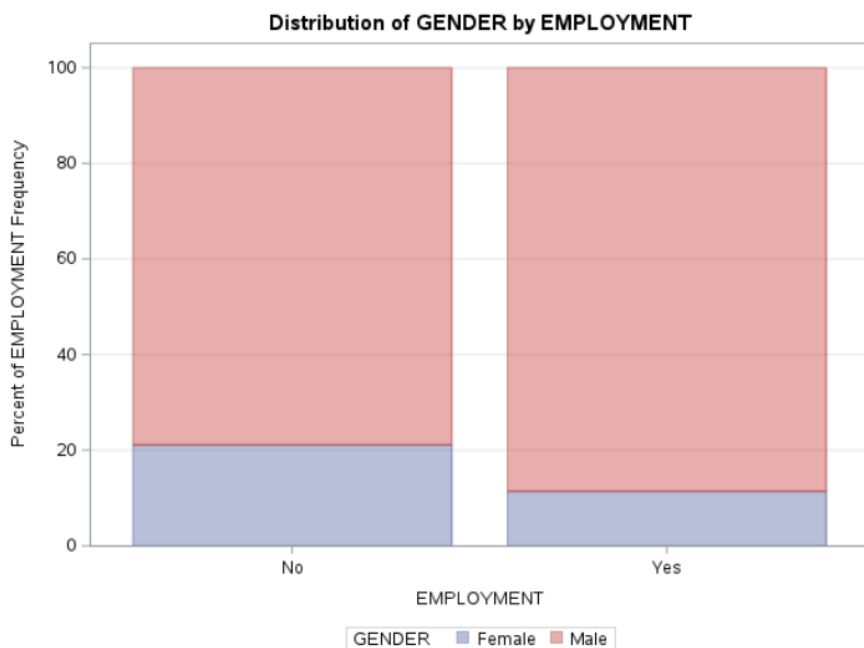


**Bivariate analysis of the variables:
Categorical variable [GENDER] VS Categorical variable [EMPLOYMENT]**

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



Description

We can observe the data of bivariate analysis to categorical data VS categorical data using SAS Macro easily. The missing values for first and third comparison are 10 and 34 respectively. From the first chart, we can observe that no matter applicants are married or not married, both groups of people have 0 family members. From the last chart in this output, we can see that most of the applicants are male, but male has higher employment percentage than female, vice versa.

6.3.2.2 Bivariate analysis of the variables (Categorical vs Continuous) using the SAS MACRO.

SAS Code

```
265 /* Macro begins here */
266 OPTIONS MCOMPILENOTE=ALL;
267
268 %MACRO BVA_CATE_CON(ptitle1, ptitle2, pdataset, pcate_vari1, pcate_vari2);
269 TITLE &ptitle1;
270 TITLE2 &ptitle2;
271 PROC MEANS DATA = &pdataset;
272 CLASS &pcate_vari1; /* Categorical */
273 VAR &pcate_vari2; /* Continuous */
274 RUN;
275 %MEND BVA_CATE_CON;
276 /* Macro end here */
277
278 /* Call the SAS Macro - BVA_CATE_CON */
279 %BVA_CATE_CON(' Bivariate analysis of the variables:',
280 ' Categorical variable [LOAN_LOCATION] VS Continuous variable [LOAN_AMOUNT] ',
281 LIB78601.TESTING_DS, LOAN_LOCATION, LOAN_AMOUNT);
282
283 %BVA_CATE_CON(' Bivariate analysis of the variables:',
284 ' Categorical variable [MARITAL_STATUS] VS Continuous variable [LOAN_AMOUNT] ',
285 LIB78601.TESTING_DS, MARITAL_STATUS, LOAN_AMOUNT);
286
287 %BVA_CATE_CON(' Bivariate analysis of the variables:',
288 ' Categorical variable [QUALIFICATION] VS Continuous variable [LOAN_AMOUNT] ',
289 LIB78601.TESTING_DS, QUALIFICATION, LOAN_AMOUNT);
```

Screenshot(s)

Bivariate analysis of the variables: Categorical variable [LOAN_LOCATION] VS Continuous variable [LOAN_AMOUNT]

The MEANS Procedure

Analysis Variable : LOAN_AMOUNT						
LOAN_LOCATION	N Obs	N	Mean	Std Dev	Minimum	Maximum
City	140	138	136.2246377	65.0807492	28.0000000	460.0000000
Town	116	114	134.0438596	61.8013361	35.0000000	550.0000000
Village	111	110	138.1818182	56.3947720	28.0000000	390.0000000

Bivariate analysis of the variables: Categorical variable [MARITAL_STATUS] VS Continuous variable [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

Bivariate analysis of the variables:
Categorical variable [QUALIFICATION] VS Continuous variable [LOAN_AMOUNT]
 The MEANS Procedure

Analysis Variable : LOAN_AMOUNT						
QUALIFICATION	N Obs	N	Mean	Std Dev	Minimum	Maximum
Graduate	283	279	141.3584229	66.1702665	28.0000000	550.0000000
Under Graduate	84	83	118.5662651	36.4628755	28.0000000	199.0000000

Description

Here is the bivariate analysis of categorical variable and continuous variable. From the loan location VS loan amount, we can see that the mean value of each category has no very huge difference, but the standard deviation for each category has difference. For marital status VS loan amount, the mean value and standard for both group of applicants have big difference. This issue also happened for qualification VS loan amount.

6.4 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TRAINING_DS.

6.4.1 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TRAINING_DS – [MARITAL STATUS]

Step1: Obtain the information.

SAS Code

```

295 /* Step1: Obtain the information of loan applicants who submitted their
296 loan application without marital status */
297
298 TITLE 'Obtain the information of loan applicants who submitted their';
299 TITLE2 'loan application without marital status';
300 FOTENOTE '-----END-----';
301 PROC SQL;
302 SELECT *
303 FROM LIB78601.TRAINING_DS e
304 WHERE (e.marital_status = '' OR e.marital_status IS MISSING);
305 QUIT;

```

Screenshot(s)

Obtain the information of loan applicants 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	180	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

-----END-----

Description

From the screenshot, we can easily see the full details of applicants that have missing values for marital status.

Step2: Count the number of missing values.

SAS Code

```
307 /* Step2: Count the number of loan applicants who submitted their
308 loan application without marital status */
309
310 TITLE 'Obtain the information of loan applicants who submitted their';
311 TITLE2 'loan application without marital status';
312 FOTENOTE '-----END-----';
313 PROC SQL;
314 SELECT COUNT(*) label = 'Number of Applicants'
315 FROM LIB78601.TRAINING_DS e
316 WHERE (e.marital_status = '' OR e.marital_status IS MISSING);
317 QUIT;
```

Screenshot(s)



Description

Clearly, we can see that there is total 3 missing values for this variable, we will treat this issue in the coming steps.

Step3: Statistics about missing values.

SAS Code

```
319 /* Step3: Find the statistics of applicants who submitted their
320 loan application without marital status */
321
322 TITLE 'Find the statistics of applicants who submitted their';
323 TITLE2 'loan application without marital status';
324 FOTENOTE '-----END-----';
325 PROC SQL;
326 SELECT e.marital_status AS MARITAL_STATUS,
327        COUNT(*) AS COUNTS
328 FROM LIB78601.TRAINING_DS e
329 WHERE (e.marital_status ne '' OR e.marital_status IS NOT MISSING)
330 GROUP BY e.marital_status;
331 QUIT;
```

Screenshot(s)

Find the statistics of applicants who submitted their loan application without marital status

MARITAL_STATUS	COUNTS
Married	398
Not Married	213

-----END-----

Description

We can observe that the number of each category for variable marital status, this information is useful for choosing treatment method for impute missing values.

Step4: Save the statistics in a dataset.

SAS Code

```
333 /* Step4: Save the statistics in a dataset */
334
335 PROC SQL;
336 CREATE TABLE LIB78601.TRAINING_STAT_DS AS
337 SELECT e.marital_status AS MARITAL_STATUS,
338        COUNT(*) AS COUNTS
339 FROM LIB78601.TRAINING_DS e
340 WHERE (e.marital_status ne ' ' OR e.marital_status IS NOT MISSING)
341 GROUP BY e.marital_status;
342 QUIT;
```

Screenshot(s)

Total rows: 2 Total columns: 2

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

Description

Save the output from last output as a table format, this will help for selecting specific value for impute missing value for specific variable in general.

Step4.1: Backup the existing dataset.

SAS Code

```
345 /* Step4.1: Create a backup dataset */
346 PROC SQL;
347 CREATE TABLE LIB78601.TRAINING_BK_DS AS
348 SELECT *
349 FROM LIB78601.TRAINING_STAT_DS;
350 QUIT;
```

Screenshot(s)

Total rows: 614 Total columns: 13

			MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	
1	LP001002	Male	Not Married	0	Graduate	No	
2	LP001003	Male	Married	1	Graduate	No	
3	LP001005	Male	Married	0	Graduate	Yes	
4	LP001006	Male	Married	0	Under Graduate	No	
5	LP001008	Male	Not Married	0	Graduate	No	
6	LP001011	Male	Married	2	Graduate	Yes	
7	LP001013	Male	Married	0	Under Graduate	No	
8	LP001014	Male	Married	3+	Graduate	No	
9	LP001018	Male	Married	2	Graduate	No	
10	LP001020	Male	Married	1	Graduate	No	
11	LP001024	Male	Married	2	Graduate	No	

Description

Create a table for duplicate the table before imputing the value for missing values cell. This action will guarantee that we will not corrupt the existing table.

Step5: Impute the missing values.

SAS Code

```
353 /* Step5: Impute the missing value found in the categorical variable MARITAL_STATUS */
354 PROC SQL;
355 UPDATE LIB78601.TRAINING_DS
356 SET marital_status = ( SELECT t1.MARITAL_STATUS AS MARITAL_STATUS
357                        FROM LIB78601.TRAINING_STAT_DS t1
358                        WHERE COUNTS = ( SELECT MAX(t2.COUNTS) AS HIGHEST_COUNT
359                                         FROM LIB78601.TRAINING_STAT_DS t2 ))
360                        /* sub-program to find highest count */
361 WHERE (marital_status = '' OR marital_status IS MISSING);
362 QUIT;
---
```

Screenshot(s)

NOTE: 3 rows were updated in LIB78601.TRAINING_DS.

Description

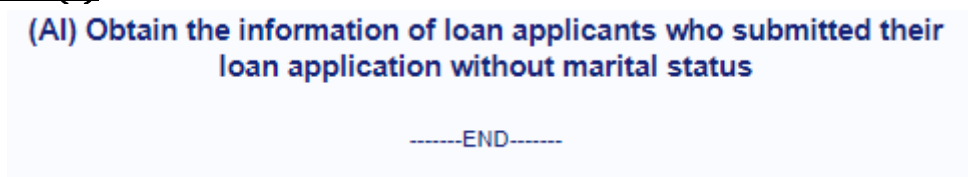
Impute the missing values by the mode of the observation from its variable. From the screenshot, we can see that 3 rows were updated.

Step6: Ensure that the missing values was imputed.

SAS Code

```
364 /* Step6: (AI) Obtain the information of loan applicants who submitted their
365 loan application without marital status */
366
367 TITLE '(AI) Obtain the information of loan applicants who submitted their';
368 TITLE2 'loan application without marital status';
369 FOTENOTE '-----END-----';
370 PROC SQL;
371 SELECT *
372 FROM LIB78601.TRAINING_DS e
373 WHERE (e.marital_status = '' OR e.marital_status IS MISSING);
374 QUIT;
```

Screenshot(s)



(AI) Obtain the information of loan applicants who submitted their
loan application without marital status

-----END-----

Description

We checked that there is not any missing value for this variable after impute the existing missing values.

6.4.2 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TRAINING_DS – [EMPLOYMENT]

Step1: Obtain the information.

SAS Code

```
383 /* Step1: Obtain the information of loan applicants who submitted their
384 loan application without employment */
385
386 TITLE 'Obtain the information of loan applicants who submitted their';
387 TITLE2 'loan application without employment';
388 FOTENOTE '-----END-----';
389 PROC SQL;
390 SELECT *
391 FROM LIB78601.TRAINING_DS e
392 WHERE (e.employment = '' OR e.employment IS MISSING);
393 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their 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	LOAN_LOCATION	LOAN_APPROVAL_STATUS
LP001027	Male	Married	2	Graduate		2500	1840	109	360	1	City	Y
LP001041	Male	Married	0	Graduate		2000	3500	115	.	1	City	Y
LP001052	Male	Married	1	Graduate		3717	2925	151	360	.	Town	N
LP001087	Female	Not Married	2	Graduate		3750	2083	120	360	1	Town	Y
LP001091	Male	Married	1	Graduate		4168	3359	201	360	.	City	N
LP001326	Male	Not Married	0	Graduate		6782	0	.	360	.	City	N
LP001370	Male	Not Married	0	Under Graduate		7333	0	120	360	1	Village	N
LP001387	Female	Married	0	Graduate		2029	2333	139	360	1	Town	Y
LP001398	Male	Not Married	0	Graduate		5050	0	118	360	1	Town	Y
LP001548	Male	Not Married	0	Graduate		2080	2083	120	360	1	Village	Y
LP001581	Male	Married	0	Under Graduate		1820	1759	95	360	1	Village	Y
LP001732	Male	Married	2	Graduate		5000	0	72	360	0	Town	N
LP001768	Male	Married	0	Graduate		3718	0	42	180	1	Village	Y
LP001786	Male	Married	0	Graduate		5745	0	255	360	.	City	N
LP001883	Female	Not Married	0	Graduate		3418	0	135	360	1	Village	N
LP001949	Male	Married	3+	Graduate		4416	1250	110	360	1	City	Y
LP002101	Male	Married	0	Graduate		6337	0	490	180	1	City	Y
LP002110	Male	Married	1	Graduate		5250	888	180	360	1	Village	Y
LP002128	Male	Married	2	Graduate		2583	2330	125	360	1	Village	Y
LP002209	Female	Not Married	0	Graduate		2784	1459	110	360	1	City	Y
LP002226	Male	Married	0	Graduate		3333	2500	128	360	1	Town	Y
LP002237	Male	Not Married	1	Graduate		3087	0	113	180	1	City	Y
LP002319	Male	Married	0	Graduate		6255	0	180	360	.	City	Y
LP002388	Male	Not Married	0	Graduate		12876	0	405	360	1	Town	Y
LP002435	Male	Married	0	Graduate		3539	1378	55	360	1	Village	N
LP002489	Female	Not Married	1	Under Graduate		5191	0	132	360	1	Town	Y
LP002502	Female	Married	2	Under Graduate		210	2917	98	360	1	Town	Y
LP002732	Male	Not Married	0	Under Graduate		2550	2042	125	360	1	Village	Y
LP002753	Female	Not Married	1	Graduate		3652	0	95	360	1	Town	Y
LP002888	Male	Not Married	0	Graduate		3182	2917	181	360	1	City	Y
LP002949	Female	Not Married	3+	Graduate		416	41657	360	180	.	City	N
LP002950	Male	Married	0	Under Graduate		2894	2792	155	360	1	Village	Y

-----END-----

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for employment.

Step2: Count the number of missing values.

SAS Code

```

395 /* Step2: Count the number of loan applicants who submitted their
396 loan application without employment */
397
398 TITLE 'Count the number of loan applicants who submitted their';
399 TITLE2 'loan application without employment';
400 FOTENOTE '-----END-----';
401 PROC SQL;
402 SELECT COUNT(*) label = 'Number of Applicants'
403 FROM LIB78601.TRAINING_DS e
404 WHERE (e.employment = '' OR e.employment IS MISSING);
405 QUIT;

```

Screenshot(s)

Count the number of loan applicants who submitted their loan application without employment

Number of Applicants
32

-----END-----

Description

Clearly, we can see that there is total 3 missing values for this variable, we will treat this issue in the coming steps.

Step3: Statistics about missing values.

SAS Code

```
407 /* Step3: Find the statistics of applicants who submitted their
408 loan application without employment */
409
410 TITLE 'Find the statistics of applicants who submitted their';
411 TITLE2 'loan application without employment';
412 FOTENOTE '-----END-----';
413 PROC SQL;
414 SELECT e.employment AS EMPLOYMENT,
415        COUNT(*) AS COUNTS
416 FROM LIB78601.TRAINING_DS e
417 WHERE (e.employment ne ' ' OR e.employment IS NOT MISSING)
418 GROUP BY e.employment;
419 QUIT;
```

Screenshot(s)

Find the statistics of applicants who submitted their loan application without employment

EMPLOYMENT	COUNTS
No	500
Yes	82

-----END-----

Description

We can observe that the number of each category for variable employment, this information is useful for choosing treatment method for impute missing values.

Step4: Save the statistics in a dataset.

SAS Code

```
421 /* Step4: Save the statistics in a dataset */
422
423 PROC SQL;
424 CREATE TABLE LIB78601.TRAINING_STAT_DS AS
425 SELECT e.employment AS EMPLOYMENT,
426        COUNT(*) AS COUNTS
427 FROM LIB78601.TRAINING_DS e
428 WHERE (e.employment ne ' ' OR e.employment IS NOT MISSING)
429 GROUP BY e.employment;
430 QUIT;
```

Screenshot(s)

Total rows: 2 Total columns: 2

Rows 1-2

	EMPLOYMENT	COUNTS
1	No	500
2	Yes	82

Description

Save the output from last output as a table format, this will help for selecting specific value for impute missing value for specific variable in general.

Step4.1: Backup the existing dataset.

SAS Code

```
433 /* Step4.1: Create a backup dataset */
434 PROC SQL;
435 CREATE TABLE LIB78601.TRAINING_BK_DS AS
436 SELECT *
437 FROM LIB78601.TRAINING_DS;
438 QUIT;
```

Screenshot(s)

Total rows: 614 Total columns: 13

Total rows: 614 Total columns: 13			MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	
1	LP001002	Male	Not Married	0	Graduate	No	
2	LP001003	Male	Married	1	Graduate	No	
3	LP001005	Male	Married	0	Graduate	Yes	
4	LP001006	Male	Married	0	Under Graduate	No	
5	LP001008	Male	Not Married	0	Graduate	No	
6	LP001011	Male	Married	2	Graduate	Yes	
7	LP001013	Male	Married	0	Under Graduate	No	
8	LP001014	Male	Married	3+	Graduate	No	
9	LP001018	Male	Married	2	Graduate	No	
10	LP001020	Male	Married	1	Graduate	No	
11	LP001024	Male	Married	2	Graduate	No	

Description

Create a table for duplicate the table before imputing the value for missing values cell. This action will guarantee that we will not corrupt the existing table.

Step5: Impute the missing values.

SAS Code

```
441 /* Step5: Impute the missing value found in the categorical variable EMPLOYMENT */
442 PROC SQL;
443 UPDATE LIB78601.TRAINING_DS
444 SET EMPLOYMENT = ( SELECT t1.EMPLOYMENT AS EMPLOYMENT
445                     FROM LIB78601.TRAINING_STAT_DS t1
446                     WHERE COUNTS = ( SELECT MAX(t2.COUNTS) AS HIGHEST_COUNT
447                                     FROM LIB78601.TRAINING_STAT_DS t2 ))
448                     /* sub-program to find highest count */
449 WHERE (employment = '' OR employment IS MISSING);
450 QUIT;
```

Screenshot(s)

NOTE: 32 rows were updated in LIB78601.TRAINING_DS.

Description

Impute the missing values by the mode of the observation from its variable. From the screenshot, we can see that 32 rows were updated.

Step6: Ensure that the missing values was imputed.

SAS Code

```
452 /* Step6: (AI) Obtain the information of loan applicants who submitted their
453 loan application without employment */
454
455 TITLE '(AI) Obtain the information of loan applicants who submitted their';
456 TITLE2 'loan application without employment';
457 FOTENOTE '-----END-----';
458 PROC SQL;
459 SELECT *
460 FROM LIB78601.TRAINING_DS e
461 WHERE (e.employment = '' OR e.employment IS MISSING);
462 QUIT;
```

Screenshot(s)

(AI) Obtain the information of loan applicants who submitted their
loan application without employment

-----END-----

Description

We checked that there is not any missing value for this variable after impute the existing missing values.

6.4.3 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TRAINING_DS – [GENDER]

Step1: Obtain the information.

SAS Code

```
468 /* Step1: Obtain the information of loan applicants who submitted their
469 loan application without gender */
470
471 TITLE 'Obtain the information of loan applicants who submitted their';
472 TITLE2 'loan application without gender';
473 FOTENOTE '-----END-----';
474 PROC SQL;
475 SELECT *
476 FROM LIB78601.TRAINING_DS e
477 WHERE (e.gender = '' OR e.gender IS MISSING);
478 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their 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	3385	1917	112	350	0	Village	N
LP001448		Married	3+	Graduate	No	23803	0	370	350	1	Village	Y
LP001585		Married	3+	Graduate	No	51793	0	700	300	1	City	Y
LP001644		Married	0	Graduate	Yes	874	5298	188	350	1	Village	Y
LP002024		Married	0	Graduate	No	2473	1843	159	350	1	Village	N
LP002103		Married	1	Graduate	Yes	9833	1833	182	180	1	City	Y
LP002478		Married	0	Graduate	Yes	2083	4083	180	350	.	Town	Y
LP002501		Married	0	Graduate	No	18692	0	110	350	1	Town	Y
LP002530		Married	2	Graduate	No	2873	1872	132	350	0	Town	N
LP002525		Not Married	0	Graduate	No	3583	0	98	350	1	City	N
LP002572		Married	0	Graduate	No	3087	2210	138	350	0	Town	N
LP002625		Not Married	0	Graduate	No	4750	0	84	350	1	Town	Y
LP002933		Not Married	3+	Graduate	Yes	9357	0	282	350	1	Town	Y

-----END-----

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for gender.

Step2: Count the number of missing values.

SAS Code

```
480 /* Step2: Count the number of loan applicants who submitted their
481 loan application without gender */
482
483 TITLE 'Count the number of loan applicants who submitted their';
484 TITLE2 'loan application without gender';
485 FOTENOTE '-----END-----';
486 PROC SQL;
487 SELECT COUNT(*) label = 'Number of Applicants'
488 FROM LIB78601.TRAINING_DS e
489 WHERE (e.gender = '' OR e.gender IS MISSING);
490 QUIT;
```

Screenshot(s)

Count the number of loan applicants who submitted their loan application without gender

Number of Applicants
13

-----END-----

Description

Clearly, we can see that there is total 3 missing values for this variable, we will treat this issue in the coming steps.

Step3: Statistics about missing values.

SAS Code

```
492 /* Step3: Find the statistics of applicants who submitted their
493 loan application without gender */
494
495 TITLE 'Find the statistics of applicants who submitted their';
496 TITLE2 'loan application without gender';
497 FOTENOTE '-----END-----';
498 PROC SQL;
499 SELECT e.gender AS GENDER,
500        COUNT(*) AS COUNTS
501 FROM LIB78601.TRAINING_DS e
502 WHERE (e.gender ne ' ' OR e.gender IS NOT MISSING)
503 GROUP BY e.gender;
504 QUIT;
```

Screenshot(s)

Find the statistics of applicants who submitted their loan application without gender

GENDER	COUNTS
Female	112
Male	489

-----END-----

Description

We can observe that the number of each category for variable gender, this information is useful for choosing treatment method for impute missing values.

Step4: Save the statistics in a dataset.

SAS Code

```
506 /* Step4: Save the statistics in a dataset */
507
508 PROC SQL;
509 CREATE TABLE LIB78601.TRAINING_STAT_DS AS
510 SELECT e.gender AS GENDER,
511        COUNT(*) AS COUNTS
512 FROM LIB78601.TRAINING_DS e
513 WHERE (e.gender ne ' ' OR e.gender IS NOT MISSING)
514 GROUP BY e.gender;
515 QUIT;
```

Screenshot(s)

Total rows: 2 Total columns: 2

	GEND...	COUNTS
1	Female	112
2	Male	489

Description

Save the output from last output as a table format, this will help for selecting specific value for impute missing value for specific variable in general.

Step4.1: Backup the existing dataset.

SAS Code

```
518 /* Step4.1: Create a backup dataset */
519 PROC SQL;
520 CREATE TABLE LIB78601.TRAINING_BK_DS AS
521 SELECT *
522 FROM LIB78601.TRAINING_DS;
523 QUIT;
```

Screenshot(s)

Total rows: 614 Total columns: 13

	Total rows: 614 Total columns: 13	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	Rows
1	LP001002	Male	Not Married	0	Graduate	No
2	LP001003	Male	Married	1	Graduate	No
3	LP001005	Male	Married	0	Graduate	Yes
4	LP001006	Male	Married	0	Under Graduate	No
5	LP001008	Male	Not Married	0	Graduate	No
6	LP001011	Male	Married	2	Graduate	Yes
7	LP001013	Male	Married	0	Under Graduate	No
8	LP001014	Male	Married	3+	Graduate	No
9	LP001018	Male	Married	2	Graduate	No
10	LP001020	Male	Married	1	Graduate	No
11	LP001024	Male	Married	2	Graduate	No

Description

Create a table for duplicate the table before imputing the value for missing values cell. This action will guarantee that we will not corrupt the existing table.

Step5: Impute the missing values.

SAS Code

```
526 /* Step5: Impute the missing value found in the categorical variable GENDER */
527 PROC SQL;
528 UPDATE LIB78601.TRAINING_DS
529 SET GENDER = ( SELECT t1.GENDER AS GENDER
530                FROM LIB78601.TRAINING_STAT_DS t1
531                WHERE COUNTS = ( SELECT MAX(t2.COUNTS) AS HIGHEST_COUNT
532                                FROM LIB78601.TRAINING_STAT_DS t2 ))
533                /* sub-program to find highest count */
534 WHERE (gender = '' OR gender IS MISSING);
535 QUIT;
```

Screenshot(s)

NOTE: 13 rows were updated in LIB78601.TRAINING_DS.

Description

Impute the missing values by the mode of the observation from its variable. From the screenshot, we can see that 13 rows were updated.

Step6: Ensure that the missing values was imputed.

SAS Code

```
538 /* Step6: (AI) Obtain the information of loan applicants who submitted their
539 loan application without gender */
540
541 TITLE '(AI) Obtain the information of loan applicants who submitted their';
542 TITLE2 'loan application without gender';
543 FOTENOTE '-----END-----';
544 PROC SQL;
545 SELECT *
546 FROM LIB78601.TRAINING_DS e
547 WHERE (e.gender = '' OR e.gender IS MISSING);
548 QUIT;
```

Screenshot(s)

(AI) Obtain the information of loan applicants who submitted their
loan application without gender

-----END-----

Description

We checked that there is not any missing value for this variable after imputing the existing missing values.

6.4.4 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TRAINING_DS – [FAMILY MEMBERS]

Step1: Obtain the information.

SAS Code

```
553 /* Step1: List the detaild of the loan applicants who sumitted their
554 applications without family members details*/
555
556 TITLE 'List the detaild of the loan applicants who sumitted their';
557 TITLE2 'applications without family members details';
558 FOTENOTE '-----END-----';
559 PROC SQL;
560 SELECT *
561 FROM LIB78601.TRAINING_DS e
562 WHERE (e.family_members = '' OR e.family_members IS MISSING);
563 QUIT;
```

Screenshot(s)

List the detaild of the loan applicants who sumitted their applications without family members details											
SME_LOAN_ID_NO	GENDER	MARITAL_STATUS	FAMILY_MEMBERS	QUALIFICATION	EMPLOYMENT	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY	LOAN_LOCATIC
LP001350	Male	Married		Graduate	No	13650	0	.	360	1	City
LP001357	Male	Married		Graduate	No	3816	754	160	360	1	City
LP001426	Male	Married		Graduate	No	5667	2667	180	360	1	Village
LP001754	Male	Married		Under Graduate	Yes	4735	0	138	360	1	City
LP001760	Male	Married		Graduate	No	4758	0	158	480	1	Town
LP001945	Female	Not Married		Graduate	No	5417	0	143	480	0	City
LP001972	Male	Married		Under Graduate	No	2875	1750	105	360	1	Town
LP002100	Male	Not Married		Graduate	No	2833	0	71	360	1	City
LP002106	Male	Married		Graduate	Yes	5503	4490	70	.	1	Town
LP002130	Male	Married		Under Graduate	No	3523	3230	152	360	0	Village
LP002144	Female	Not Married		Graduate	No	3813	0	116	180	1	City
LP002393	Female	Married		Graduate	No	10047	0	.	240	1	Town
LP002682	Male	Married		Under Graduate	No	3074	1800	123	360	0	Town
LP002847	Male	Married		Graduate	No	5116	1451	165	360	0	City
LP002943	Male	Not Married		Graduate	No	2987	0	88	360	0	Town

-----END-----

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for family members.

Step2: Count the number of missing values.

SAS Code

```
566 /* Step2: Count the number of loan applicants who submitted their
567 loan application without family members */
568
569 TITLE 'Count the number of loan applicants who submitted their';
570 TITLE2 'loan application without family members';
571 FOTENOTE '-----END-----';
572 PROC SQL;
573 SELECT COUNT(*) label = 'Number of Applicants'
574 FROM LIB78601.TRAINING_DS e
575 WHERE (e.family_members = '' OR e.family_members IS MISSING);
576 QUIT;
```

Screenshot(s)

Count the number of loan applicants who submitted their loan application without family members	
Number of Applicants	15
-----END-----	

Description

Clearly, we can see that there is total 15 missing values for this variable, we will treat this issue in the coming steps.

Step3: List the details of loan applicants with '3+' family members.

SAS Code

```
579 /* Step3: List the details of loan applicants with '3+' family members */
580
581 TITLE 'List the details of loan applicants with '3+' family members';
582 FOTENOTE '-----END-----';
583 PROC SQL;
584 SELECT e.family_members label = 'Family Members',
585        SUBSTR(e.family_members,1,1) label = 'The data found in the 1st position',
586        SUBSTR(e.family_members,2,1) label = 'The data found in the 2nd position'
587 FROM LIB78601.TRAINING_DS e
588 WHERE (e.family_members ne '' OR e.family_members IS NOT MISSING);
589 QUIT;
```

Screenshot(s)

List the details of loan applicants with 3+ family members		
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	
1	1	
2	2	
2	2	
2	2	
0	0	
2	2	
0	0	

Description

By observing the data, we found that some of the observations of family members column are recorded as '3+'. Before doing analysis, performing separation the symbol '+' to ensure that only numeric value remains is needed.

Step4: Backup the existing dataset.

SAS Code

```
592 /* Step4: Create a backup dataset */
593 PROC SQL;
594 CREATE TABLE LIB78601.TRAINING_BK_DS AS
595 SELECT *
596 FROM LIB78601.TRAINING_DS;
597 QUIT;
```

Screenshot(s)

Total rows: 614 Total columns: 13

Total rows: 614 Total columns: 13			MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	Rows
1	LP001002	Male	Not Married	0	Graduate	No	
2	LP001003	Male	Married	1	Graduate	No	
3	LP001005	Male	Married	0	Graduate	Yes	
4	LP001006	Male	Married	0	Under Graduate	No	
5	LP001008	Male	Not Married	0	Graduate	No	
6	LP001011	Male	Married	2	Graduate	Yes	
7	LP001013	Male	Married	0	Under Graduate	No	
8	LP001014	Male	Married	3+	Graduate	No	
9	LP001018	Male	Married	2	Graduate	No	
10	LP001020	Male	Married	1	Graduate	No	
11	LP001024	Male	Married	2	Graduate	No	

Description

Create a table for duplicate the table before imputing the value for missing values cell. This action will guarantee that we will not corrupt the existing table.

Step5: Remove the '+' found in the 'family_members' variable.

SAS Code

```
600 /* Step5: Remove the '+' found in the family_members variable */
601
602 TITLE 'Remove the '+' found in the family_members variable';
603 FOTENOTE '-----END-----';
604 PROC SQL;
605 UPDATE LIB78601.TRAINING_DS
606 SET family_members = SUBSTR(family_members,1,1)
607 WHERE SUBSTR(family_members,2,1) eq '+';
608 QUIT;
```

Screenshot(s)

NOTE: 51 rows were updated in LIB78601.TRAINING_DS.

Description

In step, we remove the symbol '+' from the dataset. By doing this step, the dataset will not lose any significant data since we used 3 to replace '3+'.

Step6: Show the statistics about family members variable.

SAS Code

```
611 /* Step6: Show the statistics about family members variable */
612
613 PROC SQL;
614 SELECT e.family_members AS FAMILY_MEMBERS,
615        COUNT(*) AS COUNTS
616 FROM LIB78601.TRAINING_DS e
617 WHERE (e.family_members ne ' ' OR e.family_members IS NOT MISSING)
618 GROUP BY e.family_members;
619 QUIT;
```

Screenshot(s)

FAMILY_MEMBERS	COUNTS
0	345
1	102
2	101
3	51

Description

We can observe that the number of each category for variable family members, this information is useful for choosing treatment method for impute missing values.

Step7: Save the statistics in a dataset.

SAS Code

```
622 /* Step7: Save the statistics in a dataset */
623
624 PROC SQL;
625 CREATE TABLE LIB78601.TRAINING_STAT_FM_DS AS
626 SELECT e.family_members AS FAMILY_MEMBERS,
627        COUNT(*) AS COUNTS
628 FROM LIB78601.TRAINING_DS e
629 WHERE (e.family_members ne ' ' OR e.family_members IS NOT MISSING)
630 GROUP BY e.family_members;
631 QUIT;
```

Screenshot(s)

Total rows: 4 Total columns: 2

	FAMILY_MEMB...	COUNTS
1	0	345
2	1	102
3	2	101
4	3	51

Description

Save the output from last output as a table format, this will help for selecting specific value for impute missing value for specific variable in general.

Step8: Impute the missing value found in the categorical variable FAMILY_MEMBERS.

SAS Code

```
634 /* Step8: Impute the missing value found in the categorical variable FAMILY_MEMBERS */
635 PROC SQL;
636 UPDATE LIB78601.TRAINING_DS
637 SET family_members = ( SELECT t1.FAMILY_MEMBERS AS FAMILY_MEMBERS
638                        FROM LIB78601.TRAINING_STAT_FM_DS t1
639                        WHERE COUNTS = ( SELECT MAX(t2.COUNTS) AS HIGHEST_COUNT
640                                       FROM LIB78601.TRAINING_STAT_FM_DS t2 ))
641                        /* sub-program to find highest count */
642 WHERE (family_members = '' OR family_members IS MISSING);
643 QUIT;
```

Screenshot(s)

NOTE: 15 rows were updated in LIB78601.TRAINING_DS.

Description

Impute the missing values by the mode of the observation from its variable. From the screenshot, we can see that 15 rows were updated.

Step9: (AI) List the details of the loan applicants who submitted their applications without family members details.

SAS Code

```
646 /* Step9: (AI) List the detaild of the loan applicants who sumitted their
647 applications without family members details*/
648
649 TITLE '(AI) List the detaild of the loan applicants who sumitted their';
650 TITLE2 'applications without family members details';
651 FOTENOTE '-----END-----';
652 PROC SQL;
653 SELECT *
654 FROM LIB78601.TRAINING_DS e
655 WHERE (e.family_members = '' OR e.family_members IS MISSING);
656 QUIT;
```

Screenshot(s)

(AI) List the detaild of the loan applicants who sumitted their
applications without family members details

-----END-----

Description

From the screenshot, we can easily to see that there is not any missing value after imputing. This step is to guarantee that the last step has impute all the missing values.

Step10: Step10: (AI) Count the number of loan applicants who submitted their loan application without family members.

SAS Code

```
659 /* Step10: (AI) Count the number of loan applicants who submitted their
660 loan application without family members */
661
662 TITLE '(AI) Count the number of loan applicants who submitted their';
663 TITLE2 'loan application without family members';
664 FOTENOTE '-----END-----';
665 PROC SQL;
666 SELECT COUNT(*) label = 'Number of Applicants'
667 FROM LIB78601.TRAINING_DS e
668 WHERE (e.family_members = '' OR e.family_members IS MISSING);
669 QUIT;
```

Screenshot(s)



Description

Clearly, we can see that there is total 0 missing value after imputing, this step is needed to double check that we have no making any mistake when imputing missing values.

6.5 Impute the CONTINUOUS missing values found in the dataset – LIB78601.TRAINING_DS.

6.5.1 Impute the CONTINUOUS missing values found in the dataset – LIB78601.TRAINING_DS – [LOAN_AMOUNT]

Step1: Obtain the information.

SAS Code

```
673 /* Step1: Obtain the information of loan applicants who submitted their
674 loan application without loan_amount */
675
676 TITLE 'Obtain the information of loan applicants who submitted their';
677 TITLE2 'loan application without loan_amount';
678 FOTENOTE '-----END-----';
679 PROC SQL;
680 SELECT *
681 FROM LIB78601.TRAINING_DS e
682 WHERE (e.loan_amount EQ . OR e.loan_amount IS MISSING);
683 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their loan application 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	4945	0	.	360	0	Village
LP001266	Male	Married	1	Graduate	Yes	2395	0	.	360	1	Town
LP001326	Male	Not Married	0	Graduate	No	6782	0	.	360	.	City
LP001350	Male	Married	0	Graduate	No	13650	0	.	360	1	City
LP001356	Male	Married	0	Graduate	No	4652	3583	.	360	1	Town
LP001392	Female	Not Married	1	Graduate	Yes	7451	0	.	360	1	Town
LP001449	Male	Not Married	0	Graduate	No	3865	1640	.	360	1	Village
LP001682	Male	Married	3	Under Graduate	No	3992	0	.	180	1	City
LP001922	Male	Married	0	Graduate	No	20667	0	.	360	1	Village
LP001990	Male	Not Married	0	Under Graduate	No	2000	0	.	360	1	City
LP002054	Male	Married	2	Under Graduate	No	3601	1590	.	360	1	Village
LP002113	Female	Not Married	3	Under Graduate	No	1830	0	.	360	0	City
LP002243	Male	Married	0	Under Graduate	No	3010	3136	.	360	0	City
LP002393	Female	Married	0	Graduate	No	10047	0	.	240	1	Town

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for loan amount.

Step2: Count the number of missing values.

SAS Code

```
685 /* Step2: Count the number of loan applicants who submitted their
686 loan application without loan_amount */
687
688 TITLE 'Count the number of loan applicants who submitted their';
689 TITLE2 'loan application without loan_amount';
690 FOTENOTE '-----END-----';
691 PROC SQL;
692 SELECT COUNT(*) label = 'Number of Applicants'
693 FROM LIB78601.TRAINING_DS e
694 WHERE (e.loan_amount EQ . OR e.loan_amount IS MISSING);
695 QUIT;
```

Screenshot(s)

Count the number of loan applicants who submitted their loan application without loan_amount	
Number of Applicants	22
-----END-----	

Description

Clearly, we can see that there is total 22 missing values for this variable, we will treat this issue in the coming steps.

Step3: Impute the missing value found in the CONTINUOUS variable - loan_amount.

SAS Code

```

704 /* Step3: Impute the missing value found in the
705 CONTINUOUS variable - loan_amount */
706
707 PROC STDIZE DATA = LIB78601.TRAINING_DS REONLY
708 METHOD = MEAN OUT = LIB78601.TRAINING_DS;
709 VAR loan_amount;
710 QUIT;

```

Screenshot(s)

Total rows: 614 Total columns: 13

..	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION
	5849	0	146.41216216	360
	4583	1508	128	360
	3000	0	66	360
	2583	2358	120	360
	6000	0	141	360
	5417	4196	267	360
	2333	1516	95	360
	3036	2504	158	360
	4006	1526	168	360

Description

In this step, we impute the missing value of variable loan amount by its mean value, this can guarantee that the missing data didn't affect the mean value of existing data.

Step4: (AI) Obtain the information.

SAS Code

```

712 /* Step4: (AI) Obtain the information of loan applicants who submitted their
713 loan application without loan_amount */
714
715 TITLE 'Obtain the information of loan applicants who submitted their';
716 TITLE2 'loan application without loan_amount';
717 FOTENOTE '-----END-----';
718 PROC SQL;
719 SELECT *
720 FROM LIB78601.TRAINING_DS e
721 WHERE (e.loan_amount EQ . OR e.loan_amount IS MISSING);
722 QUIT;

```

Screenshot(s)

**Obtain the information of loan applicants who submitted their
loan application without loan_amount**

-----END-----

Description

From the screenshot, we can easily see that there is not any missing value after imputing. This step is to guarantee that the last step has impute all the missing values.

Step5: (AI) Count the number of missing values.

SAS Code

```
724 /* Step5: (AI) Count the number of loan applicants who submitted their
725 loan application without loan_amount */
726
727 TITLE 'Count the number of loan applicants who submitted their';
728 TITLE2 'loan application without loan_amount';
729 FOTENOTE '-----END-----';
730 PROC SQL;
731 SELECT COUNT(*) label = 'Number of Applicants'
732 FROM LIB78601.TRAINING_DS e
733 WHERE (e.loan_amount EQ . OR e.loan_amount IS MISSING);
734 QUIT;
```

Screenshot(s)



Description

Clearly, we can see that there is total 0 missing value after imputing, this step is needed to double check that we have no making any mistake when imputing missing values.

6.5.2 Impute the CONTINUOUS missing values found in the dataset – LIB78601.TRAINING_DS – [LOAN_DURATION]

Step1: Obtain the information.

SAS Code

```
834 /* Step1: Obtain the information of loan applicants who submitted their
835 loan application without loan_duration */
836
837 TITLE 'Obtain the information of loan applicants who submitted their';
838 TITLE2 'loan application without loan_duration';
839 FOTENOTE '-----END-----';
840 PROC SQL;
841 SELECT *
842 FROM LIB78601.TRAINING_DS e
843 WHERE (e.loan_duration EQ . OR e.loan_duration IS MISSING);
844 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their loan application without loan_duration

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
LP001041	Male	Married	0	Graduate	No	2800	3500	115	-	1	City	Y
LP001109	Male	Married	0	Graduate	No	1828	1330	100	-	0	City	N
LP001136	Male	Married	0	Under Graduate	Yes	4695	0	95	-	1	City	Y
LP001137	Female	Not Married	0	Graduate	No	3410	0	88	-	1	City	Y
LP001250	Male	Married	3	Under Graduate	No	4755	0	95	-	0	Town	N
LP001361	Male	Married	0	Under Graduate	No	3572	4114	152	-	0	Village	N
LP001574	Male	Married	0	Graduate	No	3707	3185	182	-	1	Village	Y
LP001699	Female	Not Married	0	Under Graduate	No	1907	2365	120	-	1	City	Y
LP001749	Male	Married	0	Graduate	No	7578	1010	175	-	1	Town	Y
LP001770	Male	Not Married	0	Under Graduate	No	3189	2598	120	-	1	Village	Y
LP002106	Male	Married	0	Graduate	Yes	5503	4490	70	-	1	Town	Y
LP002188	Male	Not Married	0	Graduate	No	5124	0	124	-	0	Village	N
LP002357	Female	Not Married	0	Under Graduate	No	2720	0	80	-	0	City	N
LP002382	Male	Married	1	Graduate	No	7250	1987	110	-	0	City	N

-----END-----

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for loan duration.

Step2: Count the number of missing values.

SAS Code

```
846 /* Step2: Count the number of loan applicants who submitted their
847 loan application without loan_duration */
848
849 TITLE 'Count the number of loan applicants who submitted their';
850 TITLE2 'loan application without loan_duration';
851 FOTENOTE '-----END-----';
852 PROC SQL;
853 SELECT COUNT(*) label = 'Number of Applicants'
854 FROM LIB78601.TRAINING_DS e
855 WHERE (e.loan_duration EQ . OR e.loan_duration IS MISSING);
856 QUIT;
```

Screenshot(s)

Count the number of loan applicants who submitted their loan application without loan_duration

Number of Applicants
14

-----END-----

Description

Clearly, we can see that there is total 14 missing values for this variable, we will treat this issue in the coming steps.

Step3: Impute the missing value found in the CONTINUOUS variable - loan_duration.

SAS Code

```
865 /* Step3: Impute the missing value found in the
866 CONTINUOUS variable - loan_duration */
867
868 PROC STDIZE DATA = LIB78601.TRAINING_DS REONLY
869 METHOD = MEAN OUT = LIB78601.TRAINING_DS;
870 VAR loan_duration;
871 QUIT;
```

Screenshot(s)

2500	1840	109	360	1	City	Y
3073	8106	200	360	1	City	Y
1853	2840	114	360	1	Village	N
1299	1086	17	120	1	City	Y
4950	0	125	360	1	City	Y
3596	0	100	240	1	City	Y
3510	0	76	360	0	City	N
4887	0	133	360	1	Village	N
2600	3500	115	342	1	City	Y
7660	0	104	360	0	City	N
5955	5625	315	360	1	City	Y
2600	1911	116	360	0	Town	N

Description

In this step, we impute the missing value of variable loan duration by its mean value, this can guarantee that the missing data didn't affect the mean value of existing data.

Step4: (AI) Obtain the information.

SAS Code

```
873 /* Step4: (AI) Obtain the information of loan applicants who submitted their
874 loan application without loan_duration */
875
876 TITLE 'Obtain the information of loan applicants who submitted their';
877 TITLE2 'loan application without loan_duration';
878 FOTENOTE '-----END-----';
879 PROC SQL;
880 SELECT *
881 FROM LIB78601.TRAINING_DS e
882 WHERE (e.loan_duration EQ . OR e.loan_duration IS MISSING);
883 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their
loan application without loan_duration

-----END-----

Description

From the screenshot, we can easily to see that there is not any missing value after imputing. This step is to guarantee that the last step has impute all the missing values.

Step5: (AI) Count the number of missing values.

SAS Code

```
885 /* Step5: (AI) Count the number of loan applicants who submitted their
886 loan application without loan_duration */
887
888 TITLE 'Count the number of loan applicants who submitted their';
889 TITLE2 'loan application without loan_duration';
890 FOTENOTE '-----END-----';
891 PROC SQL;
892 SELECT COUNT(*) label = 'Number of Applicants'
893 FROM LIB78601.TRAINING_DS e
894 WHERE (e.loan_duration EQ . OR e.loan_duration IS MISSING);
895 QUIT;
```

Screenshot(s)



Description

Clearly, we can see that there is total 0 missing value after imputing, this step is needed to double check that we have no making any mistake when imputing missing values.

6.6 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TESTING_DS.

6.6.1 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TRAINING_DS – [LOAN_HISTORY]

Step1: Obtain the information.

SAS Code

```
902 /* Step1: Obtain the information of loan applicants who submitted their
903 loan application without loan_history */
904
905 TITLE 'Obtain the information of loan applicants who submitted their';
906 TITLE2 'loan application without loan_history';
907 FOTENOTE '-----END-----';
908 PROC SQL;
909 SELECT *
910 FROM LIB78601.TESTING_DS e
911 WHERE (e.loan_history eq . OR e.loan_history IS MISSING);
912 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their loan application without loan_history

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
LP001035	Male	Married	2	Graduate	No	2340	2548	100	360	.	City	
LP001083	Male	Not Married	3+	Graduate	No	4105	0	40	180	.	City	
LP001163	Male	Married	2	Graduate	No	4363	1250	140	360	.	City	
LP001174	Male	Married	0	Graduate	No	3772	833	57	360	.	Town	
LP001232	Male	Married	0	Graduate	No	4280	3600	165	.	.	City	
LP001475	Male	Married	0	Graduate	Yes	3188	2288	130	360	.	Village	
LP001527	Male	Married	3+	Graduate	No	6535	0	188	360	.	Town	
LP001558	Male	Not Married	0	Graduate	No	2243	2233	107	360	.	Town	
LP001601	Male	Not Married	3+	Graduate	No	4243	4123	157	360	.	Town	
LP001771	Female	Not Married	3+	Graduate	No	4083	0	103	360	.	Town	
LP001791	Male	Married	0	Graduate	Yes	32000	0	550	360	.	Town	
LP001921	Male	Not Married	1	Graduate	No	3180	2370	80	240	.	Village	
LP002009	Female	Not Married	0	Graduate	No	2018	0	65	360	.	Village	
LP002017	Male	Married	3+	Graduate	No	15312	0	187	360	.	City	
LP002048	Male	Married	0	Under Graduate	No	4483	0	135	360	.	Town	
LP002111	Male	Married		Graduate	No	3018	1300	100	360	.	City	
LP002212	Male	Married	0	Graduate	No	2165	2165	108	360	.	City	
LP002415	Female	Not Married	1	Graduate		1850	4583	81	360	.	Village	
LP002425	Male	Not Married	0	Graduate	No	3417	738	100	360	.	Village	
LP002441	Male	Not Married		Graduate	No	3579	3308	138	360	.	Town	
LP002555	Female	Not Married	0	Graduate	No	5530	0	135	360	.	City	

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for loan history.

Step2: Count the number of missing values.

SAS Code

```
914 /* Step2: Count the number of loan applicants who submitted their
915 loan application without loan_history */
916
917 TITLE 'Count the number of loan applicants who submitted their';
918 TITLE2 'loan application without loan_history';
919 FOTENOTE '-----END-----';
920 PROC SQL;
921 SELECT COUNT(*) label = 'Number of Applicants'
922 FROM LIB78601.TESTING_DS e
923 WHERE (e.loan_history eq . OR e.loan_history IS MISSING);
924 QUIT;
```

Screenshot(s)

Count the number of loan applicants who submitted their loan application without loan_history

Number of Applicants
29

-----END-----

Description

Clearly, we can see that there is total 29 missing values for this variable, we will treat this issue in the coming steps.

Step3: Statistics about missing values.

SAS Code

```
926 /* Step3: Find the statistics of applicants who submitted their
927 loan application without loan_history */
928
929 TITLE 'Find the statistics of applicants who submitted their';
930 TITLE2 'loan application without loan_history';
931 FOTENOTE '-----END-----';
932 PROC SQL;
933 SELECT e.loan_history AS loan_history,
934        COUNT(*) AS COUNTS
935 FROM LIB78601.TESTING_DS e
936 WHERE (e.loan_history ne . OR e.loan_history IS NOT MISSING)
937 GROUP BY e.loan_history;
938 QUIT;
```

Screenshot(s)

Find the statistics of applicants who submitted their
loan application without loan_history

loan_history	COUNTS
0	59
1	279

-----END-----

Description

We can observe that the number of each category for variable loan history, this information is useful for choosing treatment method for impute missing values.

Step4: Save the statistics in a dataset.

SAS Code

```
940 /* Step4: Save the statistics in a dataset */
941
942 PROC SQL;
943 CREATE TABLE LIB78601.TESTING_STAT_DS AS
944 SELECT e.loan_history AS loan_history,
945        COUNT(*) AS COUNTS
946 FROM LIB78601.TESTING_DS e
947 WHERE (e.loan_history ne . OR e.loan_history IS NOT MISSING)
948 GROUP BY e.loan_history;
949 QUIT;
```

Screenshot(s)

Total rows: 2 Total columns: 2

	loan_history	COUNTS
1	0	59
2	1	279

Description

Save the output from last output as a table format, this will help for selecting specific value for impute missing value for specific variable in general.

Step4.1: Backup the existing dataset.

SAS Code

```
952 /* Step4.1: Create a backup dataset */
953 PROC SQL;
954 CREATE TABLE LIB78601.TESTING_BK_DS AS
955 SELECT *
956 FROM LIB78601.TESTING_DS;
957 QUIT;
```

Screenshot(s)

Total rows: 367 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GI
1	LP001015	Male	Married	0	Graduate	No	5720	
2	LP001022	Male	Married	1	Graduate	No	3076	
3	LP001031	Male	Married	2	Graduate	No	5000	
4	LP001035	Male	Married	2	Graduate	No	2340	
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	
6	LP001054	Male	Married	0	Under Graduate	Yes	2165	
7	LP001055	Female	Not Married	1	Under Graduate	No	2226	
8	LP001056	Male	Married	2	Under Graduate	No	3881	
9	LP001059	Male	Married	2	Graduate		13633	
10	LP001067	Male	Not Married	0	Under Graduate	No	2400	
11	LP001078	Male	Not Married	0	Under Graduate	No	3091	
12	LP001082	Male	Married	1	Graduate		2185	
13	LP001083	Male	Not Married	3+	Graduate	No	4166	
14	LP001094	Male	Married	2	Graduate		12173	
15	LP001096	Female	Not Married	0	Graduate	No	4666	
16	LP001099	Male	Not Married	1	Graduate	No	5667	
17	LP001105	Male	Married	2	Graduate	No	4583	
18	LP001107	Male	Married	3+	Graduate	No	3786	

Description

Create a table for duplicate the table before imputing the value for missing values cell. This action will guarantee that we will not corrupt the existing table.

Step5: Impute the missing values.

SAS Code

```
960 /* Step5: Impute the missing value found in the categorical variable LOAN_HISTORY */
961 PROC SQL;
962 UPDATE LIB78601.TESTING_DS
963 SET loan_history = ( SELECT t1.loan_history AS loan_history
964                     FROM LIB78601.TESTING_STAT_DS t1
965                     WHERE COUNTS = ( SELECT MAX(t2.COUNTS) AS HIGHEST_COUNT
966                                     FROM LIB78601.TESTING_STAT_DS t2 ))
967 /* sub-program to find highest count */
968 WHERE (loan_history = . OR loan_history IS MISSING);
969 QUIT;
```

Screenshot(s)

NOTE: 29 rows were updated in LIB78601.TESTING_DS.

Description

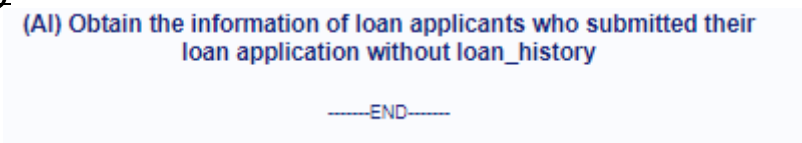
Impute the missing values by the mode of the observation from its variable. From the screenshot, we can see that 29 rows were updated.

Step6: Ensure that the missing values was imputed.

SAS Code

```
972 /* Step6: (AI) Obtain the information of loan applicants who submitted their
973 loan application without loan_history */
974
975 TITLE '(AI) Obtain the information of loan applicants who submitted their';
976 TITLE2 'loan application without loan_history';
977 FOTENOTE '-----END-----';
978 PROC SQL;
979 SELECT *
980 FROM LIB78601.TESTING_DS e
981 WHERE (e.loan_history eq . OR e.loan_history IS MISSING);
982 QUIT;
```

Screenshot(s)



(AI) Obtain the information of loan applicants who submitted their
loan application without loan_history

-----END-----

Description

We checked that there is not any missing value for this variable after imputing the existing missing values.

6.6.2 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TRAINING_DS – [GENDER]

Step1: Obtain the information.

SAS Code

```
987 /* Step1: Obtain the information of loan applicants who submitted their
988 loan application without gender */
989
990 TITLE 'Obtain the information of loan applicants who submitted their';
991 TITLE2 'loan application without gender';
992 FOTENOTE '-----END-----';
993 PROC SQL;
994 SELECT *
995 FROM LIB78601.TESTING_DS e
996 WHERE (e.gender eq '' OR e.gender IS MISSING);
997 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their 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
LP001128		Not Married	0	Graduate	No	3609	0	101	360	1	City	
LP001287		Married	3+	Under Graduate	No	3500	833	120	360	1	Town	
LP001593		Not Married	0	Graduate	No	1596	1790	119	360	0	City	
LP001799		Not Married		Graduate	No	3333	1250	110	360	1	Town	
LP002195		Not Married	1	Under Graduate	No	2038	4027	100	360	1	Village	
LP002398		Not Married	0	Graduate	Yes	2890	2688	138	360	1	City	
LP002395		Married	0	Graduate	No	3188	3145	150	180	0	Town	
LP002553		Not Married	0	Graduate	No	29167	0	185	360	1	Town	
LP002614		Not Married	0	Graduate	No	6478	0	108	360	1	Town	
LP002857		Married	1	Under Graduate	Yes	570	2125	88	360	1	Village	
LP002775		Not Married	0	Under Graduate	No	4788	0	125	360	1	Village	

-----END-----

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for gender.

Step2: Count the number of missing values.

SAS Code

```
999 /* Step2: Count the number of loan applicants who submitted their
1000 loan application without gender */
1001
1002 TITLE 'Count the number of loan applicants who submitted their';
1003 TITLE2 'loan application without gender';
1004 FOTENOTE '-----END-----';
1005 PROC SQL;
1006 SELECT COUNT(*) label = 'Number of Applicants'
1007 FROM LIB78601.TESTING_DS e
1008 WHERE (e.gender eq '' OR e.gender IS MISSING);
1009 QUIT;
```

Screenshot(s)

Count the number of loan applicants who submitted their loan application without gender

Number of Applicants
11

-----END-----

Description

Clearly, we can see that there is total 11 missing values for this variable, we will treat this issue in the coming steps.

Step3: Statistics about missing values.

SAS Code

```
1011 /* Step3: Find the statistics of applicants who submitted their
1012 loan application without gender */
1013
1014 TITLE 'Find the statistics of applicants who submitted their';
1015 TITLE2 'loan application without gender';
1016 FOTENOTE '-----END-----';
1017 PROC SQL;
1018 SELECT e.gender AS gender,
1019        COUNT(*) AS COUNTS
1020 FROM LIB78601.TESTING_DS e
1021 WHERE (e.gender ne '' OR e.gender IS NOT MISSING)
1022 GROUP BY e.gender;
1023 QUIT;
```

Screenshot(s)

Find the statistics of applicants who submitted their
loan application without gender

gender	COUNTS
Female	70
Male	286

-----END-----

Description

We can observe that the number of each category for variable gender, this information is useful for choosing treatment method for impute missing values.

Step4: Save the statistics in a dataset.

SAS Code

```
1025 /* Step4: Save the statistics in a dataset */
1026
1027 PROC SQL;
1028 CREATE TABLE LIB78601.TESTING_STAT_DS AS
1029 SELECT e.gender AS gender,
1030        COUNT(*) AS COUNTS
1031 FROM LIB78601.TESTING_DS e
1032 WHERE (e.gender ne '' OR e.gender IS NOT MISSING)
1033 GROUP BY e.gender;
1034 QUIT;
```


Screenshot(s)

Total rows: 2 Total columns: 2

	gender	COUNTS
1	Female	70
2	Male	286

Description

Save the output from last output as a table format, this will help for selecting specific value for impute missing value for specific variable in general.

Step4.1: Backup the existing dataset.

SAS Code

```
1037 /* Step4.1: Create a backup dataset */
1038 PROC SQL;
1039 CREATE TABLE LIB78601.TESTING_BK_DS AS
1040 SELECT *
1041 FROM LIB78601.TESTING_DS;
1042 QUIT;
```

Screenshot(s)

Total rows: 367 Total columns: 13

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GI
1	LP001015	Male	Married	0	Graduate	No	5720	
2	LP001022	Male	Married	1	Graduate	No	3076	
3	LP001031	Male	Married	2	Graduate	No	5000	
4	LP001035	Male	Married	2	Graduate	No	2340	
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	
6	LP001054	Male	Married	0	Under Graduate	Yes	2165	
7	LP001055	Female	Not Married	1	Under Graduate	No	2226	
8	LP001056	Male	Married	2	Under Graduate	No	3881	
9	LP001059	Male	Married	2	Graduate		13633	
10	LP001067	Male	Not Married	0	Under Graduate	No	2400	
11	LP001078	Male	Not Married	0	Under Graduate	No	3091	
12	LP001082	Male	Married	1	Graduate		2185	
13	LP001083	Male	Not Married	3+	Graduate	No	4166	
14	LP001094	Male	Married	2	Graduate		12173	
15	LP001096	Female	Not Married	0	Graduate	No	4666	
16	LP001099	Male	Not Married	1	Graduate	No	5667	
17	LP001105	Male	Married	2	Graduate	No	4583	
18	LP001107	Male	Married	3+	Graduate	No	3786	

Description

Create a table for duplicate the table before imputing the value for missing values cell. This action will guarantee that we will not corrupt the existing table.

Step5: Impute the missing values.

SAS Code

```
1045 /* Step5: Impute the missing value found in the categorical variable GENDER */
1046 PROC SQL;
1047 UPDATE LIB78601.TESTING_DS
1048 SET gender = ( SELECT t1.gender AS gender
1049                FROM LIB78601.TESTING_STAT_DS t1
1050                WHERE COUNTS = ( SELECT MAX(t2.COUNTS) AS HIGHEST_COUNT
1051                                FROM LIB78601.TESTING_STAT_DS t2 ))
1052                /* sub-program to find highest count */
1053 WHERE (gender eq '' OR gender IS MISSING);
1054 QUIT;
```

Screenshot(s)

NOTE: 11 rows were updated in LIB78601.TESTING_DS.

Description

Impute the missing values by the mode of the observation from its variable. From the screenshot, we can see that 11 rows were updated.

Step6: Ensure that the missing values was imputed.

SAS Code

```
1057 /* Step6: (AI) Obtain the information of loan applicants who submitted their
1058 loan application without gender */
1059
1060 TITLE '(AI) Obtain the information of loan applicants who submitted their';
1061 TITLE2 'loan application without gender';
1062 FOTENOTE '-----END-----';
1063 PROC SQL;
1064 SELECT *
1065 FROM LIB78601.TESTING_DS e
1066 WHERE (e.gender eq '' OR e.gender IS MISSING);
1067 QUIT;
```

Screenshot(s)

(AI) Obtain the information of loan applicants who submitted their
loan application without gender

-----END-----

Description

We checked that there is not any missing value for this variable after imputing the existing missing values.

6.6.3 Impute the CATEGORICAL missing values found in the dataset – LIB78601.TRAINING_DS – [EMPLOYMENT]

Step1: Obtain the information.

SAS Code

```
1072 /* Step1: Obtain the information of loan applicants who submitted their
1073 loan application without employment */
1074
1075 TITLE 'Obtain the information of loan applicants who submitted their';
1076 TITLE2 'loan application without employment';
1077 FOTENOTE '-----END-----';
1078 .....
1078 PROC SQL;
1079 SELECT *
1080 FROM LIB78601.TESTING_DS e
1081 WHERE (e.employment eq '' OR e.employment IS MISSING);
1082 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their 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	LOAN_LOCATION	LOAN_APPROVAL_STATUS
LP001099	Male	Married	2	Graduate		13033	0	280	240	1	City	
LP001082	Male	Married	1	Graduate		2185	1518	182	360	1	Town	
LP001094	Male	Married	2	Graduate		12173	0	168	360	0	Town	
LP001208	Male	Married	2	Graduate		7350	4029	185	180	1	City	
LP001375	Male	Married	1	Graduate		4083	1775	139	60	1	City	
LP001472	Female	Not Married	0	Graduate		5058	0	200	360	1	Village	
LP001789	Male	Married	3+	Under Graduate		6794	528	139	360	0	City	
LP001905	Male	Not Married	0	Graduate		2904	0	84	360	0	Town	
LP001950	Female	Married	3+	Graduate		1750	2935	94	360	0	Town	
LP001990	Male	Married	2	Graduate		4912	4614	160	360	1	Village	
LP002099	Male	Married	2	Under Graduate		3785	2912	180	360	0	Village	
LP002348	Male	Married	0	Graduate		2539	1704	125	360	0	Village	
LP002399	Male	Not Married	0	Graduate		2858	0	123	360	0	Village	
LP002415	Female	Not Married	1	Graduate		1850	4583	81	360	1	Village	
LP002542	Male	Married	0	Graduate		6500	0	144	360	1	City	
LP002551	Male	Married	3+	Under Graduate		3834	910	178	360	0	Town	

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for employment.

Step2: Count the number of missing values.

SAS Code

```
1084 /* Step2: Count the number of loan applicants who submitted their
1085 loan application without employment */
1086
1087 TITLE 'Count the number of loan applicants who submitted their';
1088 TITLE2 'loan application without employment';
1089 FOTENOTE '-----END-----';
1090 .....
1090 PROC SQL;
1091 SELECT COUNT(*) label = 'Number of Applicants'
1092 FROM LIB78601.TESTING_DS e
1093 WHERE (e.employment eq '' OR e.employment IS MISSING);
1094 QUIT;
```

Screenshot(s)

Count the number of loan applicants who submitted their loan application without employment	
Number of Applicants	23
-----END-----	

Description

Clearly, we can see that there is total 23 missing values for this variable, we will treat this issue in the coming steps.

Step3: Statistics about missing values.

SAS Code

```
1096 /* Step3: Find the statistics of applicants who submitted their
1097 loan application without employment */
1098
1099 TITLE 'Find the statistics of applicants who submitted their';
1100 TITLE2 'loan application without employment';
1101 FOTENOTE '-----END-----';
1102 PROC SQL;
1103 SELECT e.employment AS employment,
1104        COUNT(*) AS COUNTS
1105 FROM LIB78601.TESTING_DS e
1106 WHERE (e.employment ne '' OR e.employment IS NOT MISSING)
1107 GROUP BY e.employment;
1108 QUIT;
```

Screenshot(s)

Find the statistics of applicants who submitted their
loan application without employment

employment	COUNTS
No	307
Yes	37

-----END-----

Description

We can observe that the number of each category for variable employment, this information is useful for choosing treatment method for impute missing values.

Step4: Save the statistics in a dataset.

SAS Code

```
1110 /* Step4: Save the statistics in a dataset */
1111
1112 PROC SQL;
1113 CREATE TABLE LIB78601.TESTING_STAT_DS AS
1114 SELECT e.employment AS employment,
1115        COUNT(*) AS COUNTS
1116 FROM LIB78601.TESTING_DS e
1117 WHERE (e.employment ne '' OR e.employment IS NOT MISSING)
1118 GROUP BY e.employment;
1119 QUIT;
```

Screenshot(s)

Total rows: 2 Total columns: 2

Rows 1-2

	employment	COUNTS
1	No	307
2	Yes	37

Description

Save the output from last output as a table format, this will help for selecting specific value for impute missing value for specific variable in general.

Step4.1: Backup the existing dataset.

SAS Code

```
1122 /* Step4.1: Create a backup dataset */
1123 PROC SQL;
1124 CREATE TABLE LIB78601.TESTING_BK_DS AS
1125 SELECT *
1126 FROM LIB78601.TESTING_DS;
1127 QUIT;
```

Screenshot(s)

Total rows: 367 Total columns: 13

Rows 1-2

	SME_LOAN_ID...	GEND...	MARITAL_STA...	FAMILY_MEMB...	QUALIFICATION	EMPLOYM...	CANDIDATE_INCOME	GI
1	LP001015	Male	Married	0	Graduate	No	5720	
2	LP001022	Male	Married	1	Graduate	No	3076	
3	LP001031	Male	Married	2	Graduate	No	5000	
4	LP001035	Male	Married	2	Graduate	No	2340	
5	LP001051	Male	Not Married	0	Under Graduate	No	3276	
6	LP001054	Male	Married	0	Under Graduate	Yes	2165	
7	LP001055	Female	Not Married	1	Under Graduate	No	2226	
8	LP001056	Male	Married	2	Under Graduate	No	3881	
9	LP001059	Male	Married	2	Graduate		13633	
10	LP001067	Male	Not Married	0	Under Graduate	No	2400	
11	LP001078	Male	Not Married	0	Under Graduate	No	3091	
12	LP001082	Male	Married	1	Graduate		2185	
13	LP001083	Male	Not Married	3+	Graduate	No	4166	
14	LP001094	Male	Married	2	Graduate		12173	
15	LP001096	Female	Not Married	0	Graduate	No	4666	
16	LP001099	Male	Not Married	1	Graduate	No	5667	
17	LP001105	Male	Married	2	Graduate	No	4583	
18	LP001107	Male	Married	3+	Graduate	No	3786	

Description

Create a table for duplicate the table before imputing the value for missing values cell. This action will guarantee that we will not corrupt the existing table.

Step5: Impute the missing values.

SAS Code

```
1130 /* Step5: Impute the missing value found in the categorical variable EMPLOYMENT */
1131 PROC SQL;
1132 UPDATE LIB78601.TESTING_DS
1133 SET employment = ( SELECT t1.employment AS employment
1134                     FROM LIB78601.TESTING_STAT_DS t1
1135                     WHERE COUNTS = ( SELECT MAX(t2.COUNTS) AS HIGHEST_COUNT
1136                                     FROM LIB78601.TESTING_STAT_DS t2 ))
1137                     /* sub-program to find highest count */
1138 WHERE (employment eq '' OR employment IS MISSING);
1139 QUIT;
```

Screenshot(s)

NOTE: 23 rows were updated in LIB78601.TESTING_DS.

Description

Impute the missing values by the mode of the observation from its variable. From the screenshot, we can see that 23 rows were updated.

Step6: Ensure that the missing values was imputed.

SAS Code

```
1142 /* Step6: (AI) Obtain the information of loan applicants who submitted their
1143 loan application without employment */
1144
1145 TITLE '(AI) Obtain the information of loan applicants who submitted their';
1146 TITLE2 'loan application without employment';
1147 FOTENOTE '-----END-----';
1148 PROC SQL;
1149 SELECT *
1150 FROM LIB78601.TESTING_DS e
1151 WHERE (e.employment eq '' OR e.employment IS MISSING);
1152 QUIT;
```

Screenshot(s)

(AI) Obtain the information of loan applicants who submitted their
loan application without employment

-----END-----

Description

We checked that there is not any missing value for this variable after imputing the existing missing values.

6.7 Impute the CONTINUOUS missing values found in the dataset – LIB78601.TESTING_DS.

6.7.1 Impute the CONTINUOUS missing values found in the dataset – LIB78601.TESTING_DS – [LOAN_AMOUNT]

Step1: Obtain the information.

SAS Code

```
1279 /* Step1: Obtain the information of loan applicants who submitted their
1280 loan application without loan_amount */
1281
1282 TITLE 'Obtain the information of loan applicants who submitted their';
1283 TITLE2 'loan application without loan_amount';
1284 FOTENOTE '-----END-----';
1285 PROC SQL;
1286 SELECT *
1287 FROM LIB78601.TESTING_DS e
1288 WHERE (e.loan_amount EQ . OR e.loan_amount IS MISSING);
1289 QUIT;
```

Screenshot(s)

Obtain the information of loan applicants who submitted their loan application 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	LOAN_APPROVAL_STATUS
LP001415	Male	Married	1	Graduate	No	3413	4053	.	360	1	Town	
LP001542	Female	Married	0	Graduate	No	2282	0	.	480	0	Town	
LP002057	Male	Married	0	Under Graduate	No	13083	0	.	360	1	Village	
LP002380	Male	Married	0	Graduate	No	10000	0	.	360	1	City	
LP002593	Male	Married	1	Graduate	No	8333	4000	.	360	1	City	
-----END-----												

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for loan amount.

Step2: Count the number of missing values.

SAS Code

```
1291 /* Step2: Count the number of loan applicants who submitted their
1292 loan application without loan_amount */
1293
1294 TITLE 'Count the number of loan applicants who submitted their';
1295 TITLE2 'loan application without loan_amount';
1296 FOTENOTE '-----END-----';
1297 PROC SQL;
1298 SELECT COUNT(*) label = 'Number of Applicants'
1299 FROM LIB78601.TESTING_DS e
1300 WHERE (e.loan_amount EQ . OR e.loan_amount IS MISSING);
1301 QUIT;
```

Screenshot(s)



Description

Clearly, we can see that there is total 5 missing values for this variable, we will treat this issue in the coming steps.

Step3: Impute the missing value found in the CONTINUOUS variable - loan_amount.

SAS Code

```
1310 /* Step3: Impute the missing value found in the
1311 CONTINUOUS variable - loan_amount */
1312
1313 PROC STDIZE DATA = LIB78601.TESTING_DS REPNONLY
1314 METHOD = MEAN OUT = LIB78601.TESTING_DS;
1315 VAR loan_amount;
1316 QUIT;
```

Screenshot(s)

..	CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY
	5720	0	110	360	1
	3076	1500	126	360	1
	5000	1800	208	360	1
	2340	2546	100	360	1
	3276	0	78	360	1
	2165	3422	152	360	1
	2226	0	59	360	1
	3881	0	147	360	0
	13633	0	280	240	1
	2400	2400	123	360	1
	3091	0	90	360	1
	2185	1516	162	360	1
	4166	0	40	180	1
	12173	0	166	360	0

Description

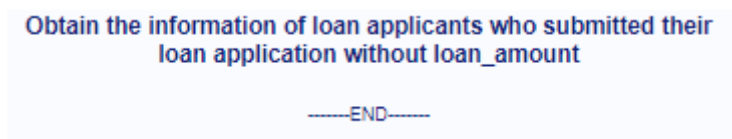
In this step, we impute the missing value of variable loan amount by its mean value, this can guarantee that the missing data didn't affect the mean value of existing data.

Step4: (AI) Obtain the information.

SAS Code

```
1318 /* Step4: (AI) Obtain the information of loan applicants who submitted their
1319 loan application without loan_amount */
1320
1321 TITLE 'Obtain the information of loan applicants who submitted their';
1322 TITLE2 'loan application without loan_amount';
1323 FOTENOTE '-----END-----';
1324 PROC SQL;
1325 SELECT *
1326 FROM LIB78601.TESTING_DS e
1327 WHERE (e.loan_amount EQ . OR e.loan_amount IS MISSING);
1328 QUIT;
```

Screenshot(s)



Description

From the screenshot, we can easily see that there is not any missing value after imputing. This step is to guarantee that the last step has impute all the missing values.

Step5: (AI) Count the number of missing values.

SAS Code

```
724 /* Step5: (AI) Count the number of loan applicants who submitted their
725 loan application without loan_amount */
726
727 TITLE 'Count the number of loan applicants who submitted their';
728 TITLE2 'loan application without loan_amount';
729 FOTENOTE '-----END-----';
730 PROC SQL;
731 SELECT COUNT(*) label = 'Number of Applicants'
732 FROM LIB78601.TRAINING_DS e
733 WHERE (e.loan_amount EQ . OR e.loan_amount IS MISSING);
734 QUIT;
```

Screenshot(s)



Description

Clearly, we can see that there is total 0 missing value after imputing, this step is needed to double check that we have no making any mistake when imputing missing values.

6.7.2 Impute the CONTINUOUS missing values found in the dataset – LIB78601.TRAINING_DS – [LOAN_DURATION]

Step1: Obtain the information.

SAS Code

```
1346 /* Step1: Obtain the information of loan applicants who submitted their
1347 loan application without loan_duration */
1348
1349 TITLE 'Obtain the information of loan applicants who submitted their';
1350 TITLE2 'loan application without loan_duration';
1351 FOTENOTE '-----END-----';
1352 PROC SQL;
1353 SELECT *
1354 FROM LIB78601.TESTING_DS e
1355 WHERE (e.loan_duration EQ . OR e.loan_duration IS MISSING);
1356 QUIT;
```

Screenshot(s)

loan application without loan_duration

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
LP001232	Male	Married	0	Graduate	No	4280	3900	185	.	1	City	
LP001288	Male	Not Married	0	Graduate	No	6792	3338	187	.	1	City	
LP001811	Male	Married	1	Graduate	No	1518	2900	80	.	0	Village	
LP001895	Male	Married	1	Under Graduate	No	3321	2088	70	.	1	Town	
LP002045	Male	Married	3	Graduate	No	10188	750	150	.	1	City	
LP002183	Male	Married	0	Under Graduate	No	3754	3719	118	.	1	Village	

-----END-----

Description

From the screenshot, we can easily to see the full details of applicants that have missing values for loan duration.

Step2: Count the number of missing values.

SAS Code

```
1358 /* Step2: Count the number of loan applicants who submitted their
1359 loan application without loan_duration */
1360
1361 TITLE 'Count the number of loan applicants who submitted their';
1362 TITLE2 'loan application without loan_duration';
1363 FOTENOTE '-----END-----';
1364 PROC SQL;
1365 SELECT COUNT(*) label = 'Number of Applicants'
1366 FROM LIB78601.TESTING_DS e
1367 WHERE (e.loan_duration EQ . OR e.loan_duration IS MISSING);
1368 QUIT;
```

Screenshot(s)

Count the number of loan applicants who submitted their
loan application without loan_duration

Number of Applicants
6

-----END-----

Description

Clearly, we can see that there is total 6 missing values for this variable, we will treat this issue in the coming steps.

Step3: Impute the missing value found in the CONTINUOUS variable - loan_duration.

SAS Code

```
1377 /* Step3: Impute the missing value found in the
1378 CONTINUOUS variable - loan_duration */
1379
1380 PROC STDIZE DATA = LIB78601.TESTING_DS REONLY
1381 METHOD = MEAN OUT = LIB78601.TESTING_DS;
1382 VAR loan_duration;
1383 QUIT;
```

Screenshot(s)

Total rows: 367 Total columns: 13

CANDIDATE_INCOME	GUARANTEE_INCOME	LOAN_AMOUNT	LOAN_DURATION	LOAN_HISTORY
2267	2792	90	360	1
5833	0	116	360	1
3643	1963	138	360	1
5629	818	100	360	1
3644	0	110	360	1
1750	2024	90	360	1
6500	2600	200	360	1
3666	0	84	360	1
4260	3900	185	342.53739612	1
4163	1475	162	360	1
2356	1902	108	360	1
6792	3338	187	342.53739612	1
8000	250	187	360	1

Description

In this step, we impute the missing value of variable loan amount by its mean value, this can guarantee that the missing data didn't affect the mean value of existing data.

Step4: (AI) Obtain the information.

SAS Code

```
1385 /* Step4: (AI) Obtain the information of loan applicants who submitted their
1386 loan application without loan_duration */
1387
1388 TITLE 'Obtain the information of loan applicants who submitted their';
1389 TITLE2 'loan application without loan_duration';
1390 FOTENOTE '-----END-----';
1391 PROC SQL;
1392 SELECT *
1393 FROM LIB78601.TESTING_DS e
1394 WHERE (e.loan_duration EQ . OR e.loan_duration IS MISSING);
1395 QUIT;
```

Screenshot(s)



Obtain the information of loan applicants who submitted their
loan application without loan_duration

-----END-----

Description

From the screenshot, we can easily see that there is not any missing value after imputing. This step is to guarantee that the last step has impute all the missing values.

Step5: (AI) Count the number of missing values.

SAS Code

```
1397 /* Step5: (AI) Count the number of loan applicants who submitted their
1398 loan application without loan_duration */
1399
1400 TITLE 'Count the number of loan applicants who submitted their';
1401 TITLE2 'loan application without loan_duration';
1402 FOTENOTE '-----END-----';
1403 PROC SQL;
1404 SELECT COUNT(*) label = 'Number of Applicants'
1405 FROM LIB78601.TESTING_DS e
1406 WHERE (e.loan_duration EQ . OR e.loan_duration IS MISSING);
1407 QUIT;
```

Screenshot(s)



Count the number of loan applicants who submitted their
loan application without loan_duration

Number of Applicants
0

-----END-----

Description

Clearly, we can see that there is total 0 missing value after imputing, this step is needed to double check that we have no making any mistake when imputing missing values.

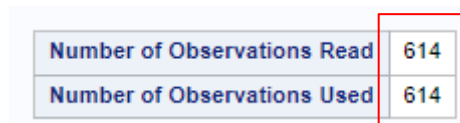
CHAPTER 7: CREATION OF MODEL USING LOGISTIC REGRESSION

7.1 Create Logistic Regression Model

SAS Code

```
1412 /* CREATION OF MODEL USING LOGISTIC REGRESSION */
1413
1414 PROC LOGISTIC DATA = LIB78601.TRAINING_DS OUTMODEL= LIB78601.TRAINING_DS_LR_MODEL;
1415 CLASS
1416     GENDER
1417     MARITAL_STATUS
1418     FAMILY_MEMBERS
1419     QUALIFICATION
1420     EMPLOYMENT
1421     LOAN_HISTORY
1422     LOAN_LOCATION
1423     ;
1424
1425 MODEL LOAN_APPROVAL_STATUS =
1426     GENDER
1427     MARITAL_STATUS
1428     FAMILY_MEMBERS
1429     QUALIFICATION
1430     EMPLOYMENT
1431     CANDIDATE_INCOME
1432     GUARANTEE_INCOME
1433     LOAN_AMOUNT
1434     LOAN_DURATION
1435     LOAN_HISTORY
1436     LOAN_LOCATION
1437     ;
1438 OUTPUT OUT = LIB78601.TRAINING_OUT_DS P = PPRED_PROB;
1439 /*****
1440 PPRED_PROB = predicted probability - variable to hold predicted probability
1441 OUT = the output will be stored in the dataset
1442 Akaike Information Criteria must (AIC) < SC (Schwarz Criterion)
1443 If Pr > ChiSq is <= 0.05, it means that that independent variable is an
1444 important variable and as is truly contributing to predict dependent variable
1445 *****/
1446 RUN;
```

Screenshot(s)



Number of Observations Read	614
Number of Observations Used	614

Description

Now, we can start to build the prediction model with logistic regression to predict the loan approval status by applicants' information. In the coding part, we have entered the variable used for building up the model. From the output, we can see that there are 614 observations in the training dataset and all of them are used in training the logistic regression model. The trained model will fit the observation through minimize the misclassification that suit the issue from loan approval.

Screenshot(s)

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

Description

From the output, we can observe that the convergence criterion is satisfied, this means that the model is trained well.

Screenshot(s)

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	764.891	587.154
SC	769.311	653.454
-2 Log L	762.891	557.154

Description

Clearly, we can see that the value for Akaike Information Criteria (AIC) was 764.891 and the value for Schwarz Criterion (SC) was 769.311. Since we have $AIC < SC$, this indicates that the model is good fit.

Screenshot(s)

Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
GENDER	1	0.0100	0.9204
MARITAL_STATUS	1	5.3173	0.0211
FAMILY_MEMBERS	3	4.3866	0.2226
QUALIFICATION	1	2.4952	0.1142
EMPLOYMENT	1	0.0060	0.9384
CANDIDATE_INCOME	1	0.2268	0.6339
GUARANTEE_INCOME	1	2.2688	0.1320
LOAN_AMOUNT	1	1.4294	0.2319
LOAN_DURATION	1	0.5322	0.4657
LOAN_HISTORY	1	87.4798	<.0001
LOAN_LOCATION	2	12.0908	0.0024

Description

From the output, we focus on the right column. We can see that the probability of variables marital_status, loan_history, and loan_location was 0.0211, <0.0001, and 0.0024 respectively, this indicates that these three variables have truly contributing to the logistic regression model for loan approval prediction.

7.2 Predicting the loan approval status using the LR model created.

SAS Code

```
1452 /*****  
1453  
1454 Predict the loan approval status using the model created  
1455  
1456 *****/  
1457  
1458 PROC LOGISTIC INMODEL = LIB78601.TRAINING_DS_LR_MODEL; /* The model that created*/  
1459 SCORE DATA = LIB78601.TESTING_DS /* Test dataset*/  
1460 OUT = LIB78601.TESTING_LAS_PRED_78601_DS; /* Location of output*/  
1461 QUIT;
```

Screenshot(s)

Table: LIB78601.TESTING_LAS_PRED_78601_DS | View: Column names | Filter: (none)

Total rows: 367 Total columns: 17

PROVAL_ST...	F_LOAN_APPROVAL_ST...	I_LOAN_APPROVAL_ST...	P_N	P_Y
		Y	0.1582296819	0.8417703181
		Y	0.2574444898	0.7425555102
		Y	0.1581934212	0.8418065788
		Y	0.1408937614	0.8591062386
		Y	0.3293748248	0.6706251752
		Y	0.2822197578	0.7177802422
		Y	0.2727031952	0.7272968048
		N	0.9369203432	0.0630796568
		Y	0.1311825475	0.8688174525
		Y	0.2360088939	0.7639911061

Description

From the output, we can see that the logistic regression has given the probability of no and yes for each observation, the probability was listed in the right column respectively. Other than that, the predicted loan approval status also listed in the list. From the list, Y represents 'Yes' and N represents 'No' for the loan approval status.

7.3 Report generation using the SAS ODL – output delivery/display system.

SAS Code

```
1466 TITLE 'Display the details of the loan approval status predicted';
1467 FOTENOTE '-----END-----';
1468
1469 PROC SQL;
1470 SELECT *
1471 FROM LIB78601.TESTING_LAS_PRED_78601_DS;
1472 QUIT;
```

Screenshot(s)

	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
1	110	360	1	City			Y	0.15823	0.84177
2	126	360	1	City			Y	0.257444	0.742556
3	208	360	1	City			Y	0.158193	0.841807
4	190	360	1	City			Y	0.140894	0.859106
5	78	360	1	City			Y	0.329375	0.670625
6	152	360	1	City			Y	0.28222	0.71778
7	59	360	1	Town			Y	0.272703	0.727297
8	147	360	0	Village			N	0.93692	0.06308
9	280	240	1	City			Y	0.131183	0.868817
10	123	360	1	Town			Y	0.236009	0.763991
11	90	360	1	City			Y	0.334946	0.665054
12	162	360	1	Town			Y	0.158905	0.841095
13	40	180	1	City			Y	0.187291	0.812709
14	166	360	0	Town			N	0.789355	0.210645
15	124	360	1	Town			Y	0.14597	0.85403
16	131	360	1	City			Y	0.359743	0.640257
17	200	360	1	City			Y	0.164788	0.835212
18	126	360	1	Town			Y	0.090288	0.909712

Description

In this section, we want to finalize the predictions output. First, we see that all observations have their output. If there are some missing values in the dataset, the predictions output will also be a missing value. From the output, we can check that there is no missing value from the output.

SAS Code

```
1476 /* Generate the report using SAS ODS - output delivery/display system
1477 Display the details of the loan approval status predicted */
1478
1479
1480 ODS HTML CLOSE;
1481 ODS PDF CLOSE;
1482 /* Determine the physical location of pdf*/
1483 ODS PDF FILE = "/home/u63691871/DAP_FT_MAR_2024_TP078601/LAS78601_report.pdf";
1484 OPTIONS NODATE;
1485 TITLE1 'Bank loan approval status predicted';
1486 TITLE2 'APU, TPM';
1487 PROC REPORT DATA = LIB78601.TESTING_LAS_PRED_78601_DS NOWINDOWS;
1488 BY SME_LOAN_ID_NO;
1489 DEFINE SME_LOAN_ID_NO / GROUP 'Loan ID';
1490 DEFINE GENDER / GROUP 'Gender Name';
1491 DEFINE MARITAL_STATUS / GROUP 'Marital Status';
1492 DEFINE FAMILY_MEMBERS / GROUP 'Family Members';
1493 DEFINE CANDIDATE_INCOME / GROUP 'Candidate Income';
1494 DEFINE GUARANTEE_INCOME / GROUP 'Co-Applicant Income';
1495 DEFINE LOAN_AMOUNT / GROUP 'Loan Amount';
1496 DEFINE LOAN_DURATION / GROUP 'Loan Duration';
1497 DEFINE LOAN_HISTORY / GROUP 'Loan History';
1498 DEFINE LOAN_LOCATION / GROUP 'Loan Location';
1499 FOOTNOTE '-----End of Report-----';
1500 RUN;
```

Screenshot(s)

V

Bank loan approval status predicted APU, TPM										
SME_LOAN_ID_NO=LP001015										
LOYMENT	Candidate Income	Co- Applicant Income	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
	5720	0	110	360	1	City			Y	0.1582297
-----End of Report-----										

Bank loan approval status predicted APU, TPM										
SME_LOAN_ID_NO=LP001022										
LOYMENT	Candidate Income	Co- Applicant Income	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
	3076	1500	126	360	1	City			Y	0.2574445
-----End of Report-----										

Description

Now, we want to generate the report for predicted output. From the screenshot, we can see that every single prediction will be written in a single form. This is easy to distribute the result for each applicant. The report has recorded the loan approval status is approve or disapprove.

CHAPTER 8: DATA VISUALIZATION

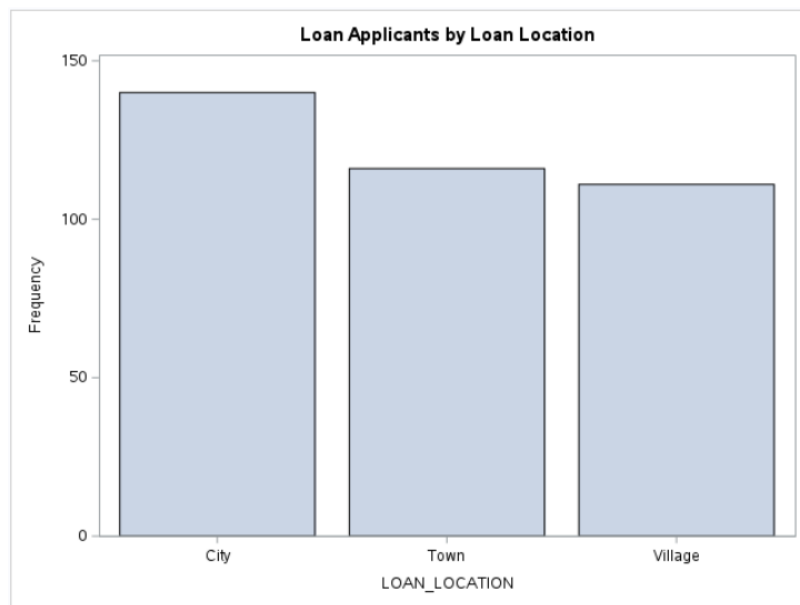
8.1 Introduction

In this section, the predicted outcomes from the loan approval prediction model are shown graphically. These data visualizations show important trends and patterns in the data, providing clear and meaningful visualizations of the model's performance. We seek to provide more comprehension of the outcomes of the model and their implications for loan process approval by using several of charts and graphs.

SAS Code

```
1514 /* SAS Simple Bar Chart*/  
1515  
1516 PROC SGPLOT DATA = LIB78601.TESTING_LAS_PRED_78601_DS;  
1517 VBAR loan_location;  
1518 TITLE 'Loan Applicants by Loan Location';  
1519 RUN;
```

Screenshot(s)



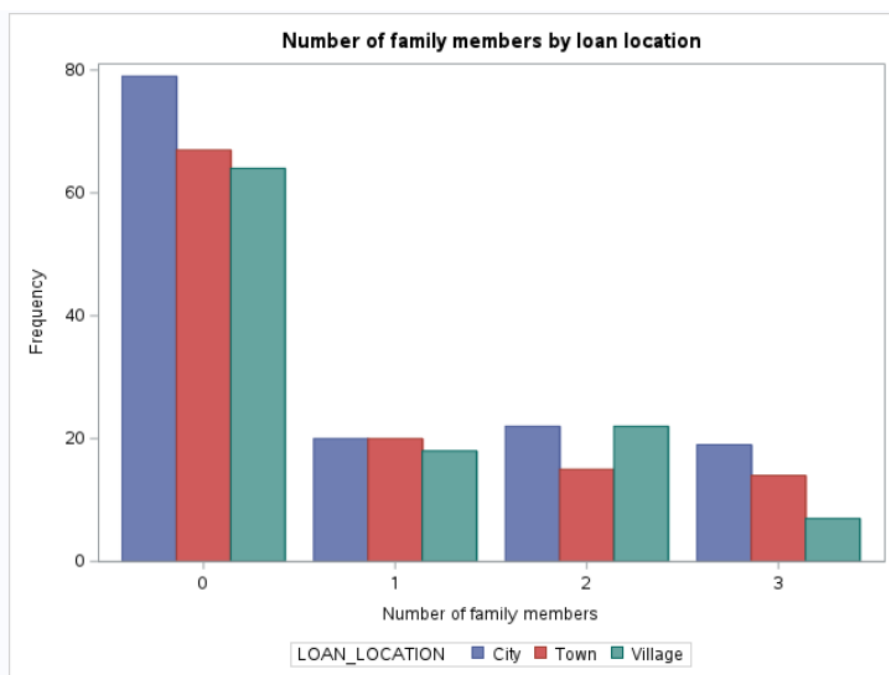
Description

It is obvious to see that most of the applicants are coming from city. From the distribution, we can see that the number of applicants from town and village are almost same, which is less than city by approximately 40.

SAS Code

```
1522 /* Bar Chart
1523     The groups were stacked one above the other */
1524
1525 TITLE 'Number of family members by loan location';
1526 PROC SGPLOT DATA = LIB78601.TESTING_LAS_PRED_78601_DS;
1527 VBAR family_members/group = loan_location groupdisplay = cluster;
1528 LABEL family_members = 'Number of family members';
1529 RUN;
```

Screenshot(s)



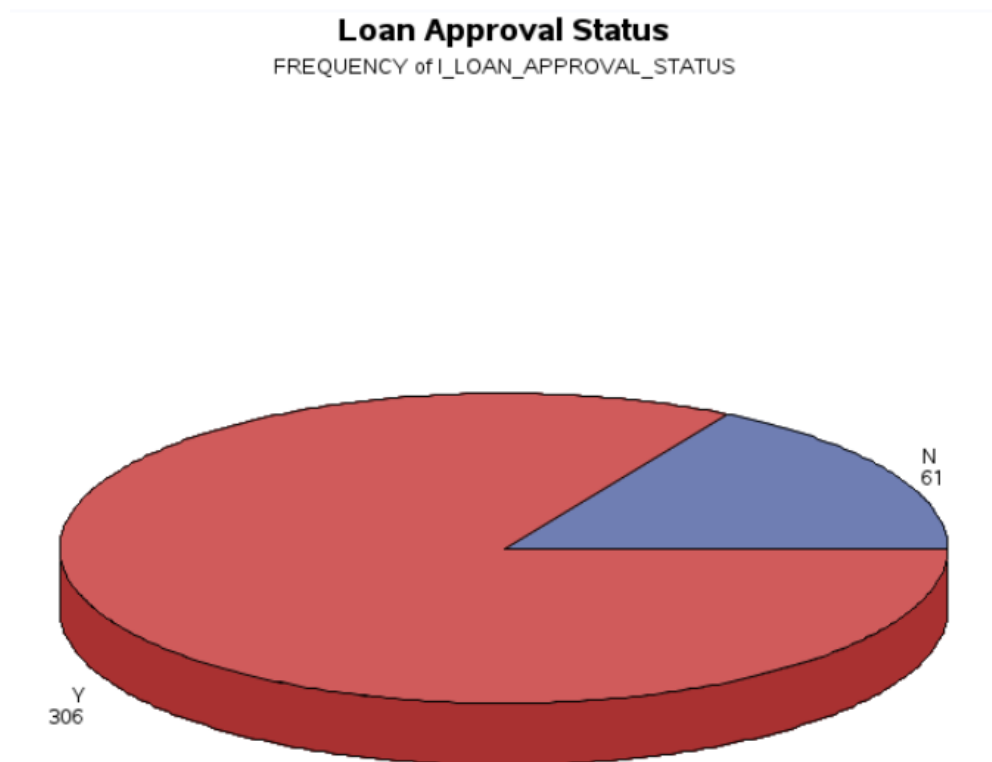
Description

This diagram shows the relationship between loan location and number of family members. From the output, we can observe that majority of the applicants didn't have any family members no matter where they come from.

SAS Code

```
1532 /* Pie Chart
1533    A pie-chart is a representation of values as slices of a circle with different colours */
1534
1535 TITLE 'Loan Approval Status';
1536 PROC GCHART DATA = LIB78601.TESTING_LAS_PRED_78601_DS;
1537     pid3d I_LOAN_APPROVAL_STATUS;
1538 RUN;
1539 QUIT;
```

Screenshot(s)



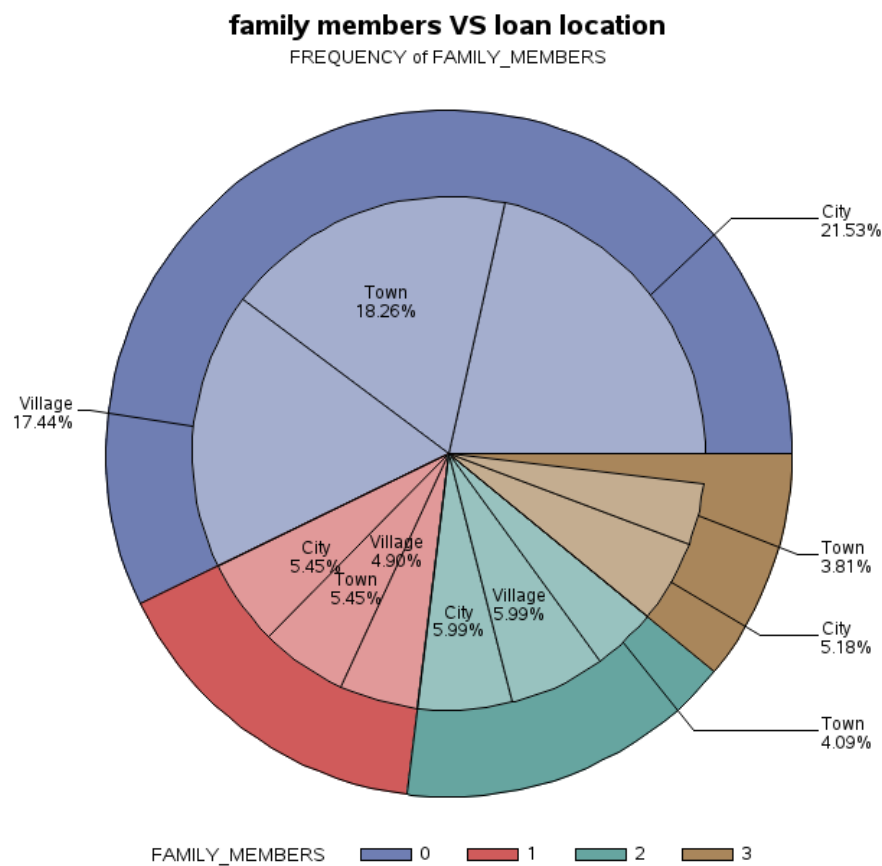
Description

This diagram is a 3-dimensional graph, the pie chart above shows the distribution of loan approval status for the testing set. From the label, we can easily to see that there are 306 applicants get approved for their loan application.

SAS Code

```
1541 GOPTIONS RESET = ALL BORDER;  
1542 TITLE 'family members VS loan location';  
1543 PROC GCHART DATA = LIB78601.TESTING_LAS_PRED_78601_DS;  
1544 pie family_members/detail = loan_location  
1545 detail_percent = best  
1546 detail_value = none  
1547 detail_slice = best  
1548 detail_threshold = 2  
1549 legend;  
1550 RUN;  
1551 QUIT;
```

Screenshot(s)



Description

This is a detailed pie chart for showing the relationship between family members and loan location. The colour represent the ratio of family members, and the inner pie chart shows the distribution of loan location for each family members category.