A·P·U
ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

INDIVIDUAL ASSIGNMENT

TECHNOLOGY PARK MALAYSIA

CT051-3-M-DM

DATA MANAGEMENT

APDMF2112DSBA(DE)(PR)

Assignment Part 1

Feature Engineering

Lee Kean Lim
TP065778

Assoc. Prof. Dr. Raja Rejeswari

ABSTRACT

Banks receive loan applications daily and data generated from such activities are voluminous and velocious. In addition, the data comes in different data types. Therefore, there is a need to standardize the data to allow machine learning models to efficiently interpret and produce higher value insights. This study investigates the data pre-processing method specifically in feature engineering for a loan dataset from Kaggle. The feature engineering would comprise of feature encoding and feature scaling which transform the data into a numeric format and within a standardized range. Based on literature, one-hot encoding and min-max scaler was chosen as the feature encoding and feature scaling technique in this study. Application of the techniques are applied to the dataset to prepare the dataset for use in a predictive model.

TABLE OF CONTENTS

ABSTRACT i

TABLE OF CONTENTS ii

LIST OF TABLES iii

LIST OF FIGURES iv

SECTION 1: INTRODUCTION 1

1.1 INTRODUCTION 1

1.2 AIM & OBJECTIVES 2

1.2.1 Aim 2

1.2.2 Objectives 2

SECTION 2: RELATED WORKS 3

2.1 LITERATURE SURVEY MATRIX 3

2.2 DISCUSSION 5

SECTION 3: METADATA 6

3.1 DATASET IMPORT 6

3.2 DATASET EXPORT 8

SECTION 4: FEATURE ENGINEERING 12

4.1 FEATURE ENCODING 12

4.2 FEATURE SCALING 22

SECTION 5: CONCLUSION 27

REFERENCES 28

LIST OF TABLES

Table 2.1: List of data transformation methods used in related works 3

LIST OF FIGURES

Figure 3.1: File upload ... 6

Figure 3.2: Dataset import ... 7

Figure 3.3: Import successful ... 7

Figure 3.4: Dataset table attributes and variables ... 8

Figure 3.5: Dataset export ... 9

Figure 3.6: Dataset export prompt ... 10

Figure 3.7: Saved file ... 11

Figure 4.1: Recoding “Gender” into numeric ... 13

Figure 4.2: Recoding “Married” into numeric ... 14

Figure 4.3: Recoding “Education” into numeric ... 15

Figure 4.4: Recoding “Self_Employed” into numeric ... 16

Figure 4.5: Recoding “Loan_Status” into numeric ... 17

Figure 4.6: One-hot encoding for “Property_Area” ... 18

Figure 4.7: Recoding “Dependents” into characters ... 19

Figure 4.8: One-hot encoding for “Dependents_Recode” ... 20

Figure 4.9: Encoded dataset overview ... 21

Figure 4.10: Min-max scaler for “ApplicantIncome” ... 22

Figure 4.11: Min-max scaler for “CoapplicantIncome” ... 23

Figure 4.12: Min-max scaler for “LoanAmount” ... 24

Figure 4.13: Min-max scaler for “Loan_Amount_Term” ... 25

Figure 4.14: Scaled dataset overview ... 26

SECTION 1

INTRODUCTION

1.1 INTRODUCTION

The automation of the loan approval process can significantly improve the work efficiency and provide a cost-effective solution to the banking sector. High volume and velocity of loan application data is received by the banks daily. As people are applying loans for different purpose namely for businesses or personal matter. In addition, these data arrive in a variety of format such as texts and numbers which require processing to fixate a standard data format to enable the predictive models to understand and utilize the data well.

Feature engineering is one of the data pre-processing methods which involve the transformation of data values into a format that allows predictive models to receive and interpret data more efficiently and output a better prediction. There are many data transformation technique under feature engineering. This study focusses on the utilization of feature encoding and feature scaling which are subsets of feature engineering on a loan dataset from Kaggle.

Feature encoding involves the transformation of text labels into a numeric format to allow predictive models to interpret the data as not many predictive models can work directly with text labels. In addition, feature scaling transforms the data values into a specified range to reduce the bias of predictive models towards large values which can be an issue particularly in distance-based predictive models.

This study is structured as followed. Section 2 discusses the literature survey, section 3 outlines the metadata, section 4 discusses the feature engineering, and section 5 concludes the study.

1.2 AIM & OBJECTIVES

1.2.1 Aim

To prepare a dataset suitable for use of machine learning predictive models by performing feature encoding and feature scaling.

1.2.2 Objectives

1. To identify the appropriate feature encoding method to be applied to the dataset.
2. To identify the appropriate feature scaling method to be applied to the dataset.
3. To perform feature encoding and feature scaling onto the dataset.

SECTION 2

RELATED WORKS

2.1 LITERATURE SURVEY MATRIX

This section outlines the feature engineering and data transformation methods used in related works. The domain in the related works would cover the credit risk assessment for credit approval. Table 2.1 shows the work conducted by researchers in developing a prediction model to assist the loan approval process. However, only feature engineering and data transformation process will be mentioned in the table in line with the objectives of this study. Following the table, will be a discussion on the findings based on the literature survey.

Table 2.1: List of data transformation methods used in related works

Reference	Feature Engineering / Data Transformation	Comments
Gupta *et al.* (2021)	- Feature scaling using various methods to identify the best transformation, achieving features closest to normal distribution	- Did not mention specifically which method of normalization was performed - Did not perform feature engineering in this study - Did not mention feature encoding
Munoz *et al.* (2021)	- Engineered feature to capture historical loan application behavior based on sum of past loan attributes - Feature dimensionality reduction using principal component analysis - One-hot encoding applied for categorical variables	- Did not mention feature scaling
Lappas and Yannacopoulos (2021)	- Standardization of dataset using z-score method	- Did not mention feature encoding as variables are all in numeric - Did not perform feature engineering in this study
Tripathi *et al.* (2020)	- Data discretization for numerical variables to balance the range using Boolean Reasoning Algorithm	- Did not perform feature engineering in this study - Did not mention feature encoding

Bao *et al.* (2019)	- Feature normalization performed using min-max scaler - Feature selection is performed to reduce number of variables used in model	- Did not perform feature engineering in this study - Did not mention feature encoding - Did not mention the method for feature selection
Wang *et al.* (2022)	- One-hot encoding applied for categorical variables	- Did not perform feature engineering in this study - Did not mention feature scaling
H. Zhang *et al.* (2021)	- Feature normalization performed using min-max scaler	- Did not perform feature engineering in this study - Did not mention feature encoding
W. Zhang *et al.* (2022)	- One-hot encoding applied for categorical variables - Feature normalization performed using z-score - Feature engineering performed based on dataset from two different sources	- Did not mention what features are engineered
Xia *et al.* (2020)	- Feature normalization performed using min-max scaler	- Did not perform feature engineering in this study - Did not mention feature encoding
Chen *et al.* (2019)	- One-hot encoding applied for categorical variables - Feature binarization applied for features with extremely uneven distribution	- Did not perform feature engineering in this study
Ashwini S. Kadam *et al.* (2021)	- One-hot encoding applied for categorical variables	- Did not perform feature engineering in this study - Did not mention feature scaling
L. Udaya Bhanu and Narayana (2021)	- Feature normalization performed using min-max scaler	- Did not perform feature engineering in this study - Did not mention feature encoding

2.2 DISCUSSION

Based on Table 2.1, it is observed that data transformation is widely applied before utilization of the data by the predictive models. This is to ensure the predictive models can operate with the data and provide better convergence speed and predictive accuracy. Typical data transformation performed by the researchers are data encoding, feature scaling, and feature creation.

In the data encoding process, typical method used by the researchers is one-hot encoding. It is performed on categorical variables to convert the labels into a numeric format. However, not all researchers outlined their method of dealing with categorical variables. As the need of data encoding would depend on the predictive algorithms used. Some predictive algorithms can work directly with labels thus does not require the conversion of labels into numeric.

Feature scaling is observed to perform by many researchers. It is a method used to convert and confine the range of the feature values. Advantage of such process would allow some predictive algorithms to gain accuracy and reduce model training time. Example, distance-based algorithms are sensitive to extreme values thus scaling features within a specified range would ensure the algorithms are treating the features with equal importance. Based on Table 2.1, the widely used feature scaling technique is the Min-Max scaler followed by the Z-score normalization. Selection of feature scaling technique is dependent on the predictive task. In general, if outliers are present, utilize Z-score normalization as it is more robust to outliers. Else, using the Min-Max scaler is sufficient.

Feature creation can provide significant improvement to the performance of the predictive model by combining features to better represent the underlying problem. However, engineering new features would require extensive experience in the specific domain to identify and combine features that would better represent the dataset thus it is not commonly used without knowledge from subject matter experts. As seen in the table, only one researcher conducted feature engineering.

SECTION 3

METADATA

This section outlines the dataset importing sequence and dataset exporting sequence in SAS® Studio. The initial file type of the dataset is a comma separated values file.

3.1 DATASET IMPORT

This section outlines the sequence of uploading a comma separated values file into SAS® Studio.

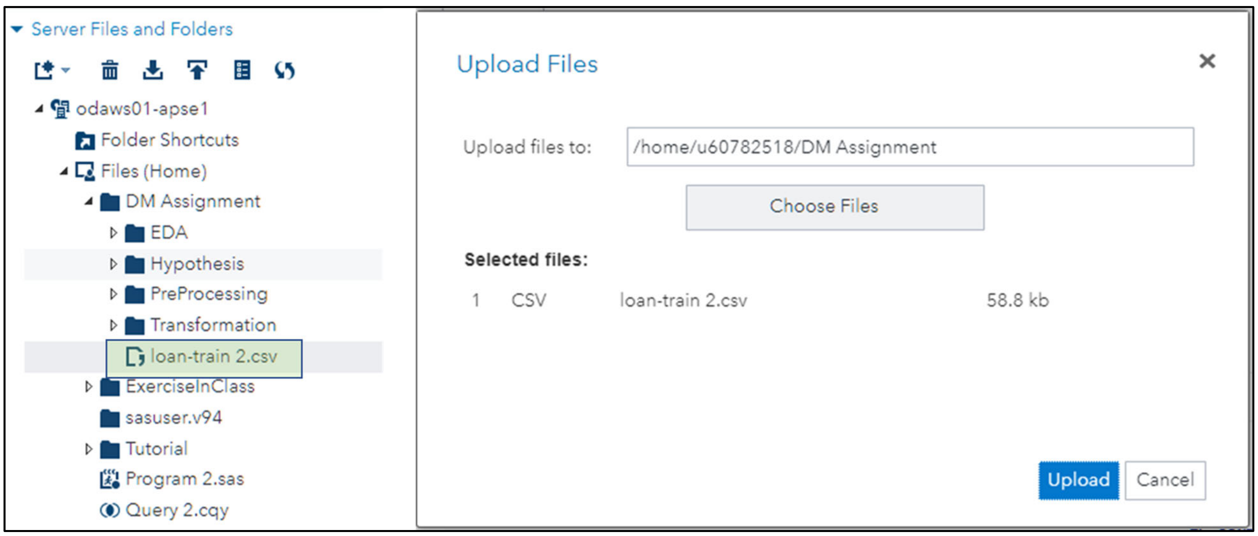

Figure 3.1: File upload

Figure 3.1 shows the file uploading interface in SAS® Studio. The comma separated values file named "loan-train 2.csv" is uploaded into the folder "DM Assignment" which contains the loan dataset in the row and column format. Once the file is uploaded, the data will be to be loaded into SAS® Studio prior to any data processing to begin. The "loan-train 2.csv" file is selected and run to import the dataset into SAS® Studio for processing as shown in Figure 3.2.

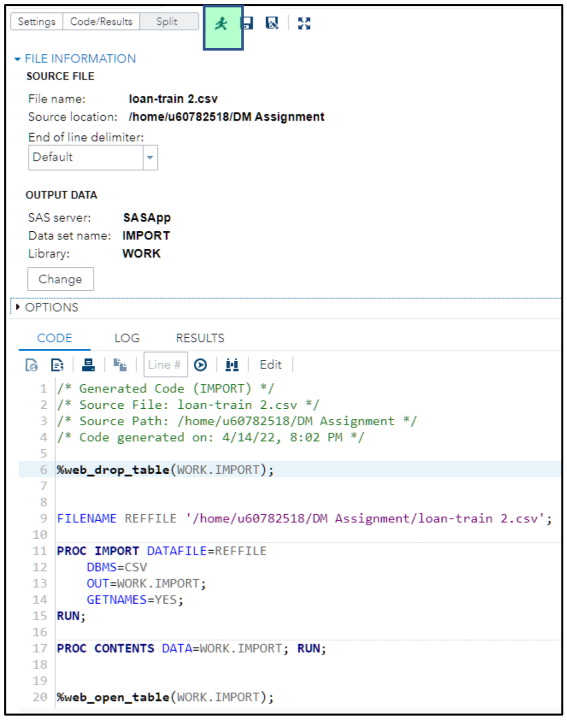

Figure 3.2: Dataset import

Once the dataset is imported successfully, navigate to the libraries pane and under My Libraries, the dataset will be appear as "IMPORT" inside the WORK folder as shown in Figure 3.3.

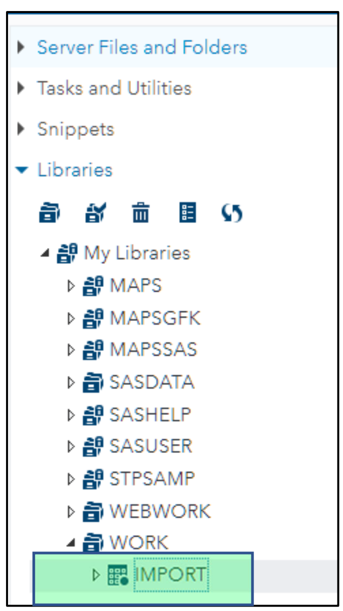

Figure 3.3: Import successful

Data Set Name	WORK.IMPORT	Observations	981
Member Type	DATA	Variables	13
Engine	V9	Indexes	0
Created	04/14/2022 20:03:52	Observation Length	88
Last Modified	04/14/2022 20:03:52	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)		

Alphabetic List of Variables and Attributes					
#	Variable	Type	Len	Format	Informat
7	ApplicantIncome	Num	8	BEST12.	BEST32.
8	CoapplicantIncome	Num	8	BEST12.	BEST32.
11	Credit_History	Num	8	BEST12.	BEST32.
4	Dependents	Char	2	$2.	$2.
5	Education	Char	12	$12.	$12.
2	Gender	Char	4	$4.	$4.
9	LoanAmount	Num	8	BEST12.	BEST32.
10	Loan_Amount_Term	Num	8	BEST12.	BEST32.
1	Loan_ID	Char	8	$8.	$8.
13	Loan_Status	Char	1	$1.	$1.
3	Married	Char	3	$3.	$3.
12	Property_Area	Char	9	$9.	$9.
6	Self_Employed	Char	3	$3.	$3.

```
ods noproctitle;
ods select attributes variables;

proc datasets;
    contents data=WORK.IMPORT order=collate;
quit;
```

Figure 3.4: Dataset table attributes and variables

The dataset will be ready to undergo any processing once it has successfully imported into the libraries. Figure 3.4 shows the attributes and variables of the dataset uploaded. The raw dataset contains 981 observations and 13 variables. A mixed of character and numeric data type is observed in the dataset. The variable table display the variables, data type, length, format, and informat of the data.

3.2 DATASET EXPORT

This section outlines the sequence of exporting a comma separated values file from SAS® Studio.

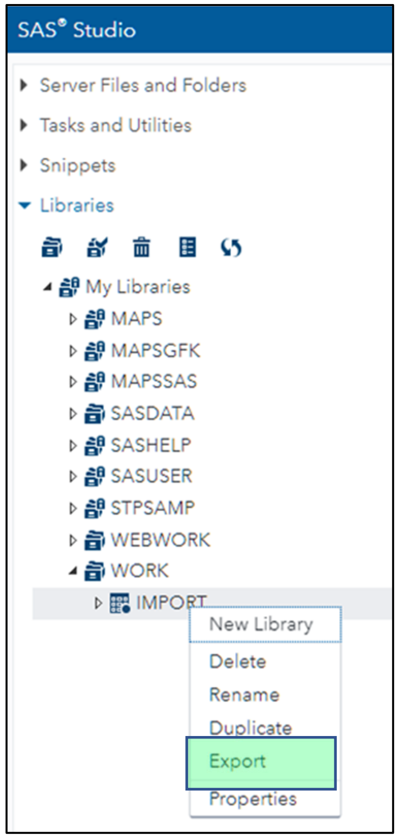

Figure 3.5: Dataset export

Assuming the data analytic work is completed, and the working file is to be downloaded from SAS® Studio. Navigate to the work file under libraries and export the file as shown in Figure 3.5.

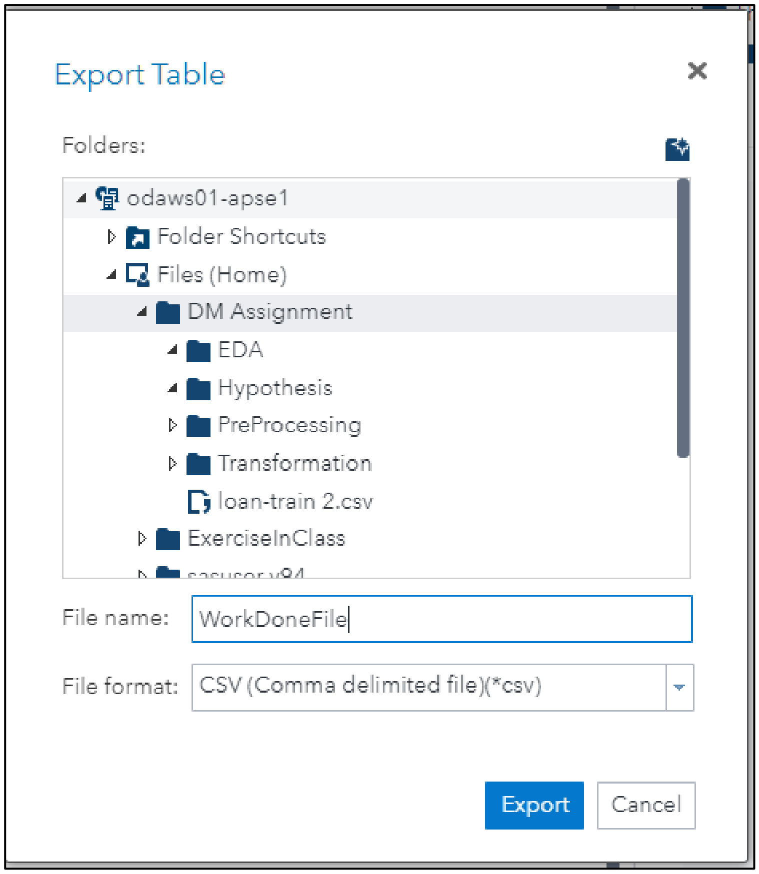

Figure 3.6: Dataset export prompt

A prompt would arise from clicking the file export and would require user to select location to store the file as shown in Figure 3.6. In this scenario, the file is named “WorkDoneFile” and will be saved as comma separated values file format. In addition, the file will be saved in “DM Assignment” directory.

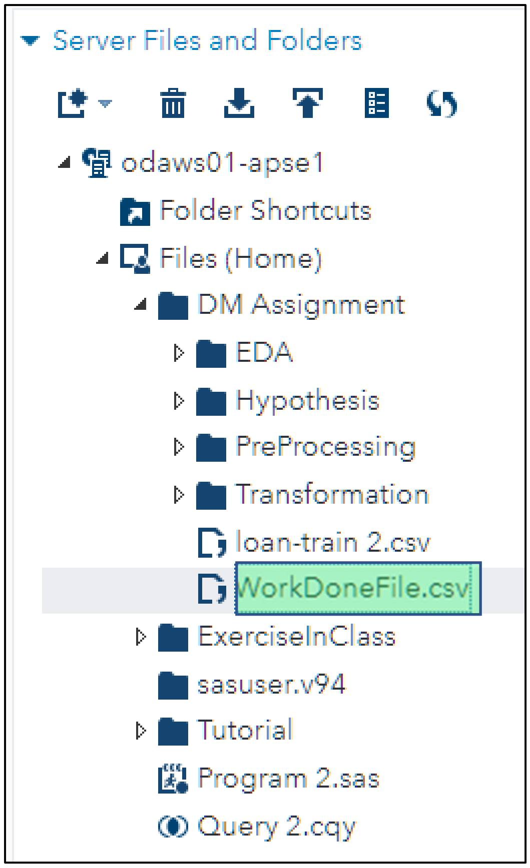

Figure 3.7: Saved file

Figure 3.7 shows the directory of the saved file. The saved file would appear under “DM Assignment” directory and would be ready for download and export to elsewhere.

SECTION 4

FEATURE ENGINEERING

This section discusses the feature engineering performed in the dataset which include feature encoding and feature scaling. Feature engineering is the modification of the format of the data to allow machine learning models to better understand the data and provide better predictive results. The cleaned dataset which performed missing value imputation and outliers removal will be used in this study for feature engineering.

4.1 FEATURE ENCODING

Feature encoding is applied to categorical variables to convert character labels into a numeric format. This is performed to enable predictive models to better understand the data, as not all predictive models are capable of receiving labels as direct input. This would improve the predictive performance of the models as well. Based on literature, the commonly used method of feature encoding is the one-hot encoding. One-hot encoding creates additional features based on the distinct classes in each categorical variable. It is typically applied to a categorical variable with three or more classes. For categorical variables with only two classes, recoding the class from text labels into binary value is sufficient.

	Gender_Recode	Gender
53	0	Male
54	0	Male
55	1	Fema
56	0	Male
57	0	Male
58	1	Fema

```
data WORK.IMPORT;
    length Gender_Recode $ 4;
    set WORK.IMPORT;

    select (Gender);
        when ('Male') Gender_Recode='0';
        when ('Fema') Gender_Recode='1';
        otherwise Gender_Recode=Gender;
    end;
run;
```

Figure 4.1: Recoding “Gender” into numeric

Figure 4.1 shows a snapshot of the table after recoding the “Gender” from character labels into numeric labels. In addition, the code snippet for performing the recoding is attached below the table. A new column is created with the name “Gender_Recode” for the encoded feature. The original feature contains two classes namely “Male” and “Fema”. The “Male” class is converted to “0” while the “Fema” class is converted to “1”.

	Married_Recode	Married
39	0	No
40	0	No
41	0	No
42	1	Yes
43	0	No
44	1	Yes

```
data WORK.IMPORT;
    length Married_Recode $ 3;
    set WORK.IMPORT;

    select (Married);
        when ('No') Married_Recode='0';
        when ('Yes') Married_Recode='1';
        otherwise Married_Recode=Married;
    end;
run;
```

Figure 4.2: Recoding “Married” into numeric

Figure 4.2 shows a snapshot of the table after recoding the “Married” from character labels into numeric labels. In addition, the code snippet for performing the recoding is attached below the table. A new column is created with the name “Married_Recode” for the encoded feature. The original feature contains two classes namely “No” and “Yes”. The “No” class is converted to “0” while the “Yes” class is converted to “1”.

	Education_Recode	Education
53	0	Not Graduate
54	1	Graduate
55	0	Not Graduate
56	1	Graduate
57	1	Graduate
58	0	Not Graduate

```
data WORK.IMPORT;
    length Education_Recode $ 12;
    set WORK.IMPORT;

    select (Education);
        when ('Not Graduate') Education_Recode='0';
        when ('Graduate') Education_Recode='1';
        otherwise Education_Recode=Education;
    end;
run;
```

Figure 4.3: Recoding “Education” into numeric

Figure 4.3 shows a snapshot of the table after recoding the “Education” from character labels into numeric labels. In addition, the code snippet for performing the recoding is attached below the table. A new column is created with the name “Education_Recode” for the encoded feature. The original feature contains two classes namely “Not Graduate” and “Graduate”. The “Not Graduate” class is converted to “0” while the “Graduate” class is converted to “1”.

	Self_Employed_Recode	Self_Employed
112	1	Yes
113	0	No
114	1	Yes
115	0	No
116	0	No
117	1	Yes

```
data WORK.IMPORT;
    length Self_Employed_Recode $ 3;
    set WORK.IMPORT;

    select (Self_Employed);
        when ('No') Self_Employed_Recode='0';
        when ('Yes') Self_Employed_Recode='1';
        otherwise Self_Employed_Recode=Self_Employed;
    end;
run;
```

Figure 4.4: Recoding "Self_Employed" into numeric

Figure 4.4 shows a snapshot of the table after recoding the "Self_Employed" from character labels into numeric labels. In addition, the code snippet for performing the recoding is attached below the table. A new column is created with the name "Self_Employed_Recode" for the encoded feature. The original feature contains two classes namely "No" and "Yes". The "No" class is converted to "0" while the "Yes" class is converted to "1".

	Loan_Status_Recode	Loan_Status
18	1	Y
19	0	N
20	0	N
21	1	Y
22	0	N
23	0	N

```
data WORK.IMPORT;
    length Loan_Status_Recode $ 1;
    set WORK.IMPORT;

    select (Loan_Status);
        when ('N') Loan_Status_Recode='0';
        when ('Y') Loan_Status_Recode='1';
        otherwise Loan_Status_Recode=Loan_Status;
    end;
run;
```

Figure 4.5: Recoding "Loan_Status" into numeric

Figure 4.5 shows a snapshot of the table after recoding the "Loan_Status" from character labels into numeric labels. In addition, the code snippet for performing the recoding is attached below the table. A new column is created with the name "Loan_Status_Recode" for the encoded feature. The original feature contains two classes namely "N" and "Y". The "N" class is converted to "0" while the "Y" class is converted to "1".

	Property_Area	Rural	Semiurban	Urban
28	Semiurban	0	1	0
29	Rural	1	0	0
30	Semiurban	0	1	0
31	Semiurban	0	1	0
32	Urban	0	0	1
33	Urban	0	0	1

```
%macro hot_encoding(data,var);
   proc sql noprint;
     select distinct &var
       into:val1-
     from &data;
 select count(distinct(&var))   into:len from &data;
 quit;
 data import;
     set &data;
   %do i=1 %to &len;
       if &var="&&&val&i" then %sysfunc(compress(&&&val&i,'$ - /'))=1 ;
       else  %sysfunc(compress(&&&val&i,'$ - /'))=0;
   %end;
   run;
 %mend;

 %hot_encoding(import, Property_Area)
```

Figure 4.6: One-hot encoding for “Property_Area”

Figure 4.6 shows a snapshot of the table after applying one-hot encoding to “Property_Area”. In addition, the code snippet for applying one-hot encoding is attached below the table. Three new columns are created from “Property_Area” namely “Rural”, “Semiurban”, and “Urban”. Each new column represents a single class from “Property_Area” with binary value where “0” indicates no and “1” indicates yes.

	Dependents	Dependents_Recode
7	3+	Dependents_3_or_more
8	0	Dependents_0
9	2	Dependents_2
10	1	Dependents_1
11	2	Dependents_2
12	2	Dependents_2

```
data WORK.IMPORT;
    length Dependents_Recode $ 20;
    set WORK.IMPORT;

    select (Dependents);
        when ('0') Dependents_Recode='Dependents_0';
        when ('1') Dependents_Recode='Dependents_1';
        when ('2') Dependents_Recode='Dependents_2';
        when ('3+') Dependents_Recode='Dependents_3_or_more';
        otherwise Dependents_Recode=Dependents;
    end;
run;
```

Figure 4.7: Recoding “Dependents” into characters

Figure 4.7 shows a snapshot of the table after recoding the “Dependents”. In addition, the code snippet for performing the recoding is attached below the table. A new column is created with the name “Dependents_Recode” for the encoded feature. The original feature contains four classes namely “0”, “1”, “2”, and “3+”. The “0” class is converted to “Dependents_0”, the “1” class is converted to “Dependents_1”, the “2” class is converted to “Dependents_2”, and the “3” class is converted to “Dependents_3_or_more”. The recoding of “Dependents” is performed to facilitate the one-hot encoding process in the following step.

	Dependents_Recode	Dependents_0	Dependents_1	Dependents_2	Dependents_3_or_more
42	Dependents_0	1	0	0	0
43	Dependents_1	0	1	0	0
44	Dependents_2	0	0	1	0
45	Dependents_0	1	0	0	0
46	Dependents_3_or_more	0	0	0	1
47	Dependents_0	1	0	0	0

```
%macro hot_encoding(data,var);
   proc sql noprint;
     select distinct &var
       into:val1-
     from &data;
 select count(distinct(&var))   into:len from &data;
 quit;
 data import;
     set &data;
   %do i=1 %to &len;
       if &var="&&&val&i" then %sysfunc(compress(&&&val&i,'$ - / +'))=1 ;
       else  %sysfunc(compress(&&&val&i,'$ - / +'))=0;
   %end;
   run;
 %mend;

 %hot_encoding(import, Dependents_Recode)
```

Figure 4.8: One-hot encoding for “Dependents_Recode”

Figure 4.8 shows a snapshot of the table after applying one-hot encoding to “Dependents_Recode”. In addition, the code snippet for applying one-hot encoding is attached below the table. Four new columns are created from “Dependents_Recode” namely “Dependents_0”, “Dependents_1”, “Dependents_2”, and “Dependents_3_or_more”. Each new column represents a single class from “Dependents_Recode” with binary value where “0” indicates no and “1” indicates yes.

Obs	Self_Employed_Recode	Married_Recode	Loan_Status_Recode	Gender_Recode	Education_Recode	ApplicantIncome
1	0	0	1	0	1	5849
2	0	1	0	0	1	4583
3	1	1	1	0	1	3000
4	0	1	1	0	0	2583
5	0	0	1	0	1	6000
6	0	1	1	0	0	2333

Obs	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Dependents_0	Dependents_1
1	0	142.5	360	1	1	0
2	1508	128	360	1	0	1
3	0	66	360	1	1	0
4	2358	120	360	1	1	0
5	0	141	360	1	1	0
6	1516	95	360	1	1	0

Obs	Dependents_2	Dependents_3_or_more	Rural	Semiurban	Urban
1	0	0	0	0	1
2	0	0	1	0	0
3	0	0	0	0	1
4	0	0	0	0	1
5	0	0	0	0	1
6	0	0	0	0	1

```
PROC SQL;
CREATE TABLE WORK.query AS
SELECT Self_Employed_Recode , Married_Recode , Loan_Status_Recode ,
Gender_Recode , Education_Recode , ApplicantIncome , CoapplicantIncome ,
LoanAmount , Loan_Amount_Term , Credit_History , Dependents_0 ,
Dependents_1 , Dependents_2 , Dependents_3_or_more , Rural , Semiurban ,
Urban FROM WORK.IMPORT;
RUN;
QUIT;

PROC DATASETS NOLIST NODETAILS;
CONTENTS DATA=WORK.query OUT=WORK.details;
RUN;

PROC PRINT DATA=WORK.details;
RUN;
```

Figure 4.9: Encoded dataset overview

Figure 4.9 shows the dataset from the results of performing feature encoding. In addition, the code snippet for generating the table view is attached below the table. Excluding the ID column, the total number of variables have increased to 17 variables from the initial 11 variables. All variables are now in the numeric format which can be easily managed by the predictive models.

4.2 FEATURE SCALING

Feature scaling is applied to numerical variables to normalize the value within a confined range. This is performed to enable predictive models to converge in a quicker manner. In addition, scaling the variables would reduce the bias in predictive models towards variables with larger values in magnitude. Based on literature, the commonly used method of feature scaling is min-max scaler. The min-max scaler transforms the variable values into a range between zero and one.

	ApplicantIncome	MM_ApplicantIncome
1	5849	0.6270368782
2	4583	0.4913164666
3	3000	0.3216123499
4	2583	0.2769082333
5	6000	0.6432246998
6	2333	0.2501072041

```
%macro Min_MaxScaler(dataset, variable);

proc sql noprint;
select min(&variable) into: min
from &dataset;

select max(&variable) into: max
from &dataset;
quit;

data import;
set &dataset;
MM_&variable = (&variable - &min)/(&max-&min);
run;

%mend;

%Min_MaxScaler(import, ApplicantIncome)
```

Figure 4.10: Min-max scaler for "ApplicantIncome"

Figure 4.10 shows a snapshot of the table after feature scaling for "ApplicantIncome". In addition, the code snippet for performing feature scaling is attached below the table. A new column is created with the name "MM_ApplicantIncome" for the scaled variable. The new value for the variable is confined between zero and one.

	CoapplicantIncome	MM_CoapplicantIncome
5	0	0
6	1516	0.26591826
7	2504	0.4392211893
8	0	0
9	1526	0.2676723382
10	1500	0.2631117348

```
%macro Min_MaxScaler(dataset, variable);

proc sql noprint;
select min(&variable) into: min
from &dataset;

select max(&variable) into: max
from &dataset;
quit;

data import;
set &dataset;
MM_&variable = (&variable - &min)/(&max-&min);
run;

%mend;

%Min_MaxScaler(import, CoapplicantIncome)
```

Figure 4.11: Min-max scaler for "CoapplicantIncome"

Figure 4.11 shows a snapshot of the table after feature scaling for "CoapplicantIncome". In addition, the code snippet for performing feature scaling is attached below the table. A new column is created with the name "MM_CoapplicantIncome" for the scaled variable. The new value for the variable is confined between zero and one.

	LoanAmount	MM_LoanAmount
1	142.5	0.5480349345
2	128	0.4847161572
3	66	0.2139737991
4	120	0.4497816594
5	141	0.5414847162
6	95	0.3406113537

```
%macro Min_MaxScaler(dataset, variable);

proc sql noprint;
select min(&variable) into: min
from &dataset;

select max(&variable) into: max
from &dataset;
quit;

data import;
set &dataset;
MM_&variable = (&variable - &min)/(&max-&min);
run;

%mend;

%Min_MaxScaler(import, LoanAmount)
```

Figure 4.12: Min-max scaler for “LoanAmount”

Figure 4.12 shows a snapshot of the table after feature scaling for “LoanAmount”. In addition, the code snippet for performing feature scaling is attached below the table. A new column is created with the name “MM_LoanAmount” for the scaled variable. The new value for the variable is confined between zero and one.

	Loan_Amount_Term	MM_Loan_Amount_Term
1	6	0
2	12	0.0126582278
3	36	0.0632911392
4	36	0.0632911392
5	36	0.0632911392
6	60	0.1139240506

```
%macro Min_MaxScaler(dataset, variable);

proc sql noprint;
select min(&variable) into: min
from &dataset;

select max(&variable) into: max
from &dataset;
quit;

data import;
set &dataset;
MM_&variable = (&variable - &min)/(&max-&min);
run;

%mend;

%Min_MaxScaler(import, Loan_Amount_Term)
```

Figure 4.13: Min-max scaler for “Loan_Amount_Term”

Figure 4.13 shows a snapshot of the table after feature scaling for “Loan_Amount_Term”. In addition, the code snippet for performing feature scaling is attached below the table. A new column is created with the name “MM_Loan_Amount_Term” for the scaled variable. The new value for the variable is confined between zero and one.

Obs	Self_Employed_Recode	Married_Recode	Loan_Status_Recode	Gender_Recode	Education_Recode	Credit_History
1	0	0	1	0	1	1
2	0	1	0	0	1	1
3	1	1	1	0	1	1
4	0	1	1	0	0	1
5	0	0	1	0	1	1
6	0	1	1	0	0	1

Obs	Dependents_0	Dependents_1	Dependents_2	Dependents_3_or_more	Rural	Semiurban
1	1	0	0	0	0	0
2	0	1	0	0	1	0
3	1	0	0	0	0	0
4	1	0	0	0	0	0
5	1	0	0	0	0	0
6	1	0	0	0	0	0

Obs	Urban	MM_ApplicantIncome	MM_CoapplicantIncome	MM_LoanAmount	MM_Loan_Amount_Term
1	1	0.62704	0.00000	0.54803	0.74684
2	0	0.49132	0.26451	0.48472	0.74684
3	1	0.32161	0.00000	0.21397	0.74684
4	1	0.27691	0.41361	0.44978	0.74684
5	1	0.64322	0.00000	0.54148	0.74684
6	1	0.25011	0.26592	0.34061	0.74684

```
PROC SQL;
CREATE TABLE WORK.query AS
SELECT Self_Employed_Recode , Married_Recode , Loan_Status_Recode ,
 Gender_Recode , Education_Recode , Credit_History , Dependents_0 ,
 Dependents_1 , Dependents_2 , Dependents_3_or_more , Rural ,
 Semiurban , Urban , MM_ApplicantIncome , MM_CoapplicantIncome ,
 MM_LoanAmount , MM_Loan_Amount_Term FROM WORK.IMPORT;
RUN;
QUIT;

PROC DATASETS NOLIST NODETAILS;
CONTENTS DATA=WORK.query OUT=WORK.details;
RUN;

PROC PRINT DATA=WORK.details;
RUN;
```

Figure 4.14: Scaled dataset overview

Figure 4.14 shows the dataset from the results of performing feature scaling. In addition, the code snippet for generating the table view is attached below the table. It is observed that the value range of numerical variables are transformed into a range between zero and one.

The dataset undergone feature encoding and feature scaling will now be ready for use in the predictive models.

SECTION 5

CONCLUSION

Feature engineering remains a crucial process in the data pre-processing stage to transform the data into usable format for the predictive models to excel. Two methods of feature engineering are discussed in this study namely feature encoding and feature scaling which are important data transformation step to enable predictive models to understand and produce a quicker and more accurate result. Specifically, one-hot encoding and min-max scaler is performed in this study. These methods are the commonly used methods in feature engineering for loan dataset based on literature. Further research can explore feature creations that would significantly improve the predictive results if performed with the opinions of subject matter experts. In which, feature creations require the in depth understanding of the domain to be able to compute new and insightful features based on the existing available data.

REFERENCES

Ashwini S. Kadam, Shraddha R. Nikam, Ankita A. Aher, Gayatri V. Shelke, & Chandgude, A. S. (2021). Prediction for Loan Approval using Machine Learning Algorithm. *International Research Journal of Engineering and Technology (IRJET), 8*(4), 4089-4092.

Bao, W., Lianju, N., & Yue, K. (2019). Integration of unsupervised and supervised machine learning algorithms for credit risk assessment. *Expert Systems with Applications, 128*, 301-315. doi:https://doi.org/10.1016/j.eswa.2019.02.033

Chen, X., Liu, Z., Zhong, M., Liu, X., & Song, P. (2019). *A Deep Learning Approach Using DeepGBM for Credit Assessment*. Paper presented at the Proceedings of the 2019 International Conference on Robotics, Intelligent Control and Artificial Intelligence, Shanghai, China. https://doi-org.ezproxy.apu.edu.my/10.1145/3366194.3366333

Dumitrescu, E., Hué, S., Hurlin, C., & Tokpavi, S. (2022). Machine learning for credit scoring: Improving logistic regression with non-linear decision-tree effects. *European Journal of Operational Research, 297*(3), 1178-1192. doi:https://doi.org/10.1016/j.ejor.2021.06.053

Gupta, K., Chakrabarti, B., Ansari, A. A., Rautaray, S. S., & Pandey, M. (2021). *Loanification-Loan Approval Classification using Machine Learning Algorithms.* Paper presented at the Proceedings of the International Conference on Innovative Computing & Communication (ICICC) 2021.

L. Udaya Bhanu, & Narayana, D. S. (2021). Customer Loan Prediction Using Supervised Learning Technique. *International Journal of Scientific and Research Publications, 11*(6). doi: 10.29322/IJSRP.11.06.2021.p11453

Lappas, P. Z., & Yannacopoulos, A. N. (2021). A machine learning approach combining expert knowledge with genetic algorithms in feature selection for credit risk assessment. *Applied Soft Computing, 107*, 107391. doi:https://doi.org/10.1016/j.asoc.2021.107391

Munoz, J., Rezaei, A. A., Jalili, M., & Tafakori, L. (2021). Deep learning based bi-level approach for proactive loan prospecting. *Expert Systems with Applications, 185*, 115607. doi:https://doi.org/10.1016/j.eswa.2021.115607

Tripathi, D., Edla, D. R., Kuppili, V., & Bablani, A. (2020). Evolutionary Extreme Learning Machine with novel activation function for credit scoring. *Engineering Applications of Artificial Intelligence, 96*, 103980. doi:https://doi.org/10.1016/j.engappai.2020.103980

Wang, T., Liu, R., & Qi, G. (2022). Multi-classification assessment of bank personal credit risk based on multi-source information fusion. *Expert Systems with Applications, 191*, 116236. doi:https://doi.org/10.1016/j.eswa.2021.116236

Xia, Y., Zhao, J., He, L., Li, Y., & Niu, M. (2020). A novel tree-based dynamic heterogeneous ensemble method for credit scoring. *Expert Systems with Applications, 159*, 113615. doi:https://doi.org/10.1016/j.eswa.2020.113615

Zhang, H., Shi, Y., Yang, X., & Zhou, R. (2021). A firefly algorithm modified support vector machine for the credit risk assessment of supply chain finance. *Research in International Business and Finance, 58*, 101482. doi:https://doi.org/10.1016/j.ribaf.2021.101482

Zhang, W., Yan, S., Li, J., Tian, X., & Yoshida, T. (2022). Credit risk prediction of SMEs in supply chain finance by fusing demographic and behavioral data. *Transportation Research Part E: Logistics and Transportation Review, 158*, 102611. doi:https://doi.org/10.1016/j.tre.2022.102611