BIG DATA PROJECT

DR. MOHAMED ABBASSY

ENG. HOSSAM ELSOKARY

ENG. SALLY

OUR TEAM

Ali Mohamed Sayed Ahmed Ali	20221449583
Alaa Ebrahem Mohamed Ali	20221379220
Menna Allah Mohamed Abdelhamid Shweel	20221400420
Doaa Khamis Kamal Ewaidat	20221400409
Marwa Mohamed Atya	20221370533
Manar Mohamed Younis Ahmed	20221370636
Mayar Mohamed Abdelfattah	20221381232

What is SAS OnDemand

SAS OnDemand for Academics provides an online delivery model for teaching and learning statistical analysis, data mining and forecasting. Whether you are an independent learner, a student, or an instructor, you can access SAS software via the cloud free of charge.

1. DATA COLLECTION

- The dataset contains information on loan applications and approval decisions. It includes both numerical and categorical variables describing applicants’ financial profiles, loan details, and demographic information.

OVERVIEW DATASET

- **The dataset** contains 1015 rows and 14 columns.
- **Key Attributes:**
  - Loan_ID: Unique identifier for each loan.
  - Gender: Gender of the applicant (Male/Female).
  - Married: Marital status of the applicant (Yes/No).
  - Dependents: Number of dependents the applicant has (e.g., 0, 1, 2, 3+).
  - Education: Education level of the applicant (Graduate/Not Graduate).
  - Self_Employed: Employment type of the applicant (Yes/No).
  - ApplicantIncome: Income of the loan applicant.
  - CoapplicantIncome: Income of the co-applicant, if any.
  - LoanAmount: Amount of loan requested.
  - Loan_Amount_Term: Loan repayment term in months.
  - Credit_History: Whether the applicant has a credit history (1: Yes, 0: No).
  - Property_Area: Urban, Semi-Urban, or Rural areas.
  - Loan_Status: Loan approval status (Y/N) (Target).
  - Age

2. Data preprocessing

- Data preprocessing is a crucial step in any data analysis or machine learning workflow. It involves transforming raw data into a clean, structured, and usable format to ensure high-quality results.

The main steps include:

1. **Data Cleaning:**
    - Handling missing values (imputation).
    - Addressing outliers to reduce their impact on models.
2. **Data Transformation:**
    - Encoding categorical variables (label encoding).
    - Scaling or normalizing numerical data for consistency.

CODING IMPORT

- The PROC IMPORT statement reads the CSV file loan_train.csv into the SAS dataset loan_data. The dataset is structured with multiple variables.

```
PROC IMPORT DATAFILE="/home/u63508066/final_proj/loan_train.csv"
    out=loan_data
    DBMS=csv
    REPLACE;
RUN;
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIn
1	LP001002	Male	No	0	Graduate	No	5849	
2	LP001003	Male	Yes	1	Graduate	No	4583	
3	LP001005	Male	Yes	0	Graduate	Yes	3000	
4	LP001006	Male	Yes	0	Not Graduate	No	2583	
5	LP001008	Male	No	0	Graduate	No	6000	

CODING CLEAN FIRST WAY

- Replace Missing Values with Mode using macro function

```
%macro replace_missing_with_mode(data=, column=, out_data=);
    Step 1: Calculate the mode for the specified column
    proc sql;
        create table &column._Freq as
        select &column, count(*) as freq
        from &data
        where not missing(&column) /* Exclude missing values
        group by &column
        order by freq desc; /* Sort by descending frequency */
    quit;
    proc sql noprint;
        select &column
        into :mode_value
        from &column._Freq
        having freq = max(freq); /* Select the &column with the highest frequency
    quit;
    Debugging: Verify the resolved mode value for the specified column
    %put Mode Value for &column.: &mode_value.;

    Step 2: Replace missing values with the mode for the specified column
    data &out_data;
        set &data;
        if missing(&column) then &column = "&mode_value.";
    run;
    Step 3: Verify the results
    proc freq data=&out_data;
        tables &column / missing;
    run;
%mend replace_missing_with_mode;
```

- A temporary table (&column._Freq) is created that counts the frequency of each non-missing value in the specified column.
- Using PROC SQL NOPRINT, the most frequent value (mode) is extracted into a macro variable &mode_value.
- A new dataset &out_data is created, where missing values in the target column are replaced with the mode value.
- The macro assumes the input column has non-missing values to calculate the mode.

CODING CLEAN FIRST WAY

- This SAS macro is designed to replace missing numeric values in a specified column of a dataset with the mean (average) value of that column.

```
/* 1-replace_missing_with_mean */
%macro replace_missing_and_save(data=, numeric_column=, out_data=);
    Step 1: Calculate the mean for the specified numeric column
    proc sql noprint;
        select mean(&numeric_column) into :mean_value
        from &data
        where not missing(&numeric_column); /* Exclude missing values */
    quit;

    %put Mean Value of &numeric_column: &mean_value.;

    Step 2: Create a new dataset with missing values replaced
    data &out_data;
        set &data;
        if missing(&numeric_column) then &numeric_column = &mean_value; /* Replace mis
    run;

    Step 3: Verify the results
    proc means data=&out_data nmiss mean;
        var &numeric_column;
    run;

    View the first 10 rows of the new dataset
    proc print data=&out_data (obs=10);
        title "First 10 Observations of Dataset &out_data";
    run;
%mend replace_missing_and_save;

Example Usage
%replace_missing_and_save(data=loan_data, numeric_column=LoanAmount, out_data=loan_dat
```

- The PROC SQL step calculates the mean of the specified numeric column, excluding missing values (where not missing), and saves it into the macro variable &mean_value.
- A new dataset (&out_data) is created, where missing values in the target column are replaced with the calculated mean.
- PROC MEANS is used to calculate and display the number of missing values (NMISS), mean, and other statistics for the numeric column after replacement.

CODING CLEAN SECOND WAY

- The dataset handles missing categorical variables by imputing them with their respective modes, ensuring consistency in categorical data.

```
proc freq  data=loan_data;
/* tables Loan_Amount_Term,Credit_History,Property_Area*/
run;
/* Impute missing values in loan_data to create imputed_data */
data imputed_data;
    set loan_data;
    Gender = COALESCEC(Gender, "Male");
    Married = COALESCEC(Married, "Yes");
    Dependents = COALESCEC(Dependents, "0");
    Self_Employed = COALESCEC(Self_Employed, "No");
    Education = COALESCEC(Education, "Graduate");
    Loan_Amount_Term = COALESCE(Loan_Amount_Term, 360);
    Credit_History = COALESCE(Credit_History, 1);
    Property_Area = COALESCEC(Property_Area, "Semiurban");
```

the Final Dataset imputed_data

Obs	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status	Age
1	LP001002	Male	No	0	Graduate	No	5849	0	145.2771709	360	1	Urban	Y	39
2	LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	1	Rural	N	33
3	LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	1	Urban	Y	41
4	LP001006	Male	Yes	0	Not Graduate	No	2583	2358	120	360	1	Urban	Y	50
5	LP001008	Male	No	0	Graduate	No	6000	0	141	360	1	Urban	Y	32
6	LP001011	Male	Yes	2	Graduate	Yes	5417	4196	267	360	1	Urban	Y	32
7	LP001013	Male	Yes	0	Not Graduate	No	2333	1516	95	360	1	Urban	Y	50
8	LP001014	Male	Yes	3+	Graduate	No	3036	2504	158	360	0	Semiurban	N	42
9	LP001018	Male	Yes	2	Graduate	No	4006	1526	168	360	1	Urban	Y	30
10	LP001020	Male	Yes	1	Graduate	No	12841	10968	349	360	1	Semiurban	N	40

CODING CLEAN SECOND WAY

- The dataset handles missing numerical values by replaced with precomputed mean values.

```
/* Compute summary statistics for numerical variables */
proc means data=imputed_data mean;
    var LoanAmount ApplicantIncome CoapplicantIncome;
run;

/* Additional imputations for numerical variables */
data imputed_data;
    set imputed_data; /* Use the already imputed dataset */
    LoanAmount = COALESCE(LoanAmount, 145.2771709);
    ApplicantIncome = COALESCE(ApplicantIncome, 5324.75);
    CoapplicantIncome = COALESCE(CoapplicantIncome, 1549.75);
run;
```

the Final Dataset imputed_data

Obs	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status	Age
1	LP001002	Male	No	0	Graduate	No	5849	0	145.2771709	360	1	Urban	Y	39
2	LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	1	Rural	N	33
3	LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	1	Urban	Y	41
4	LP001006	Male	Yes	0	Not Graduate	No	2583	2358	120	360	1	Urban	Y	50
5	LP001008	Male	No	0	Graduate	No	6000	0	141	360	1	Urban	Y	32
6	LP001011	Male	Yes	2	Graduate	Yes	5417	4196	267	360	1	Urban	Y	32
7	LP001013	Male	Yes	0	Not Graduate	No	2333	1516	95	360	1	Urban	Y	50
8	LP001014	Male	Yes	3+	Graduate	No	3036	2504	158	360	0	Semiurban	N	42
9	LP001018	Male	Yes	2	Graduate	No	4006	1526	168	360	1	Urban	Y	30
10	LP001020	Male	Yes	1	Graduate	No	12841	10968	349	360	1	Semiurban	N	40

HANDLING OUTLIERS

- Outliers are removed based on the Interquartile Range (IQR) rule. Observations that fall outside 1.5 times the IQR from the quartiles are excluded.
- Lower Threshold = Q1 – 1.5 × IQR , Upper Threshold = Q3 + 1.5 × IQR

The PROC MEANS procedure computes the first quartile (Q1) and third quartile (Q3) for the numerical variables:

- LoanAmount, ApplicantIncome, CoapplicantIncome

```
PROC MEANS DATA=imputed_data NOPRINT Q1 Q3;
    VAR LoanAmount ApplicantIncome CoapplicantIncome;
    OUTPUT OUT=iqr_params
        Q1=Q1_LoanAmount Q1_ApplicantIncome Q1_CoapplicantIncome
        Q3=Q3_LoanAmount Q3_ApplicantIncome Q3_CoapplicantIncome;
RUN;
```

The DATA step removes rows that contain outliers based on the IQR thresholds:

1. Compute IQR and thresholds:
2. Filter rows where all numerical values fall within the calculated thresholds.

```
DATA no_outliers;
    SET imputed_data;
    IF _N_ = 1 THEN SET iqr_params; /* Bring in Q1 and Q3 values */

    /* Calculate IQR and thresholds */
    IQR_LoanAmount = Q3_LoanAmount - Q1_LoanAmount;
    LoanAmount_Low = Q1_LoanAmount - 1.5 * IQR_LoanAmount;
    LoanAmount_High = Q3_LoanAmount + 1.5 * IQR_LoanAmount;

    IQR_ApplicantIncome = Q3_ApplicantIncome - Q1_ApplicantIncome;
    ApplicantIncome_Low = Q1_ApplicantIncome - 1.5 * IQR_ApplicantIncome;
    ApplicantIncome_High = Q3_ApplicantIncome + 1.5 * IQR_ApplicantIncome;

    IQR_CoapplicantIncome = Q3_CoapplicantIncome - Q1_CoapplicantIncome;
    CoapplicantIncome_Low = Q1_CoapplicantIncome - 1.5 * IQR_CoapplicantIncome;
    CoapplicantIncome_High = Q3_CoapplicantIncome + 1.5 * IQR_CoapplicantIncome;

    /* Retain rows without outliers */
    IF LoanAmount >= LoanAmount_Low AND LoanAmount <= LoanAmount_High AND
       ApplicantIncome >= ApplicantIncome_Low AND ApplicantIncome <= ApplicantIncome_High AND
       CoapplicantIncome >= CoapplicantIncome_Low AND CoapplicantIncome <= CoapplicantIncome_High;
RUN;
```

HANDLING OUTLIERS

- Outliers are removed based on the Interquartile Range (IQR) rule. Observations that fall outside 1.5 times the IQR from the quartiles are excluded.
- Lower Threshold = Q1 – 1.5 × IQR , Upper Threshold = Q3 + 1.5 × IQR

- PROC MEANS confirms that all remaining data falls within the expected ranges by displaying the minimum and maximum values for the variables.

```
PROC MEANS DATA=no_outliers MIN MAX;
    VAR LoanAmount ApplicantIncome CoapplicantIncome;
RUN;

proc print data=no_outliers ();
    title "the Final Dataset no_outliers";
run;
```

The MEANS Procedure

Variable	Minimum	Maximum
LoanAmount	8.2429319	260.0000000
ApplicantIncome	147.7648000	10139.00
CoapplicantIncome	0	5625.00

the Final Dataset no_outliers

Obs	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_
1	LP001002	Male	No	0	Graduate	No	5849	0	145.2771709	360	
2	LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	
3	LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	
4	LP001006	Male	Yes	0	Not Graduate	No	2583	2358	120	360	
5	LP001008	Male	No	0	Graduate	No	6000	0	141	360	

3. Some analysis

- **A SAS procedure used to sort data in descending order.**
  - Sorting by Loan Amount makes it easier to identify the highest loans quickly.

```
/* 3-Sort the dataset by LoanAmount in descending order */
PROC SORT DATA=imputed_data;
    BY DESCENDING LoanAmount;
RUN;
```

the SORTed Dataset

Obs	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term
1	LP001585	Male	Yes	3+	Graduate	No	51624.988883	0	718.85397826	300
2	LP001469	Male	No	0	Graduate	Yes	21646.31143	0	710.72645998	480
3	LP001585	Male	Yes	3+	Graduate	No	51763	0	700	300
4	LP001469	Male	No	0	Graduate	Yes	21554.108255	0	675.23957456	480
5	LP001469	Male	No	0	Graduate	Yes	20166	0	650	480

- **Calculate total and average LoanAmount by Loan_Status**
  - Provides a breakdown of total and average loan amounts for approved and disapproved loans.

```
PROC SQL;
    CREATE TABLE Loan_totals AS
    SELECT Loan_Status,
           SUM(LoanAmount) AS LoanAmount_Sum,
           MEAN(LoanAmount) AS LoanAmount_Average
    FROM imputed_data
    GROUP BY Loan_Status;
QUIT;
```

Total rows: 2 Total columns: 3 Rows 1-2

	Loan_Status	LoanAmount_Sum	LoanAmount_Average
1	N	46322.432005	148.46933335
2	Y	100988.61925	143.85843197

- **categorized_Aged**
  - The categorized age groups help in performing more granular analysis, such as age-specific trends in loan approval

```
DATA categorized_Aged;
    SET imputed_data;
    IF Age < 18 THEN Age_Category = 'Child';
    ELSE IF Age < 30 THEN Age_Category = 'Young_Adult';
    ELSE IF Age < 50 THEN Age_Category = 'Adult';
    ELSE IF Age < 70 THEN Age_Category = 'Old';
RUN;
```

_History	Property_Area	Loan_Status	Age	Age_Category
1	Urban	Y	26	Young
1	Urban	Y	20	Young
1	Urban	Y	26	Young

- **categorized_Aged**
  - The categorized age groups help in performing more granular analysis, such as age-specific trends in loan approval

```
/* 4-Create a subset of LoanAmount */
DATA high_value_Loan_Amount;
    SET imputed_data;
    WHERE LoanAmount > 600;
RUN;
```

Total rows: 5 Total columns: 14

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount
1	LP001585	Male	Yes	3+	Graduate	No	51624.988883	0	718.85397826
2	LP001469	Male	No	0	Graduate	Yes	21646.31143	0	710.72645998
3	LP001585	Male	Yes	3+	Graduate	No	51763	0	700
4	LP001469	Male	No	0	Graduate	Yes	21554.108255	0	675.23957456
5	LP001469	Male	No	0	Graduate	Yes	20166	0	650

SCALING NUMERICAL DATA.

- Min-Max Scaling ensures that all scaled variables fall within the range of 0 to 1, making them comparable and suitable for downstream analyses or machine learning models.
- ensure the correctness of scaling by checking minimum, maximum, and mean values,

```
/* Perform Min-Max Scaling */
DATA scaled_data;
    SET no_outliers;
    /* Scaling formula: (value - min) / (max - min) */
    IF _N_ = 1 THEN SET scaling_params; /* Bring in min-max values */
    Scaled_LoanAmount = (LoanAmount - min_LoanAmount) / (max_LoanAmount - min_LoanAmount);
    Scaled_ApplicantIncome = (ApplicantIncome - min_ApplicantIncome) / (max_ApplicantIncome - min_ApplicantIncome);
    Scaled_CoapplicantIncome = (CoapplicantIncome - min_CoapplicantIncome) / (max_CoapplicantIncome - min_Coapplica
RUN;

/* Verify the scaled dataset */
PROC MEANS DATA=scaled_data MIN MAX MEAN;
    VAR Scaled_LoanAmount Scaled_ApplicantIncome Scaled_CoapplicantIncome;
RUN;

/* View the first 10 rows of the scaled dataset */
PROC PRINT DATA=scaled_data (OBS=10);
    TITLE "First 10 Observations of Scaled Dataset";
RUN;
```

The MEANS Procedure

Variable	Minimum	Maximum	Mean
Scaled_LoanAmount	0	1.0000000	0.4642031
Scaled_ApplicantIncome	0	1.0000000	0.3877737
Scaled_CoapplicantIncome	0	1.0000000	0.2294882

ENCODING CATEGORICAL VARIABLES.

- transforms categorical variables into numeric values, allowing them to be used effectively in machine learning algorithms
  - Gender: Encoded as 1 for "Male" and 0 for "Female".
  - Married: Encoded as 1 for "Yes" and 0 for "No".
  - Loan_Status: Encoded as 1 for "Y" (Loan Approved) and 0 for "N" (Loan Rejected).
  - Self_Employed: Encoded as 1 for "Yes" and 0 for "No".
  - Education: Encoded as 1 for "Graduate" and 0 for "Not Graduate".
  - Property_Area: Encoded as 1 for "Urban", 2 for "Semiurban", and 3 for "Rural".

```
DATA label_encoded_data;
    SET scaled_data;
    /* Example: Encoding Gender */
    IF Gender = "Male" THEN Gender_Encoded = 1;
    ELSE IF Gender = "Female" THEN Gender_Encoded = 0;

    /* Encoding Married */
    IF Married = "Yes" THEN Married_Encoded = 1;
    ELSE IF Married = "No" THEN Married_Encoded = 0;

    /* Encoding Loan_Status */
    IF Loan_Status = "Y" THEN Loan_Status_Encoded = 1;
    ELSE IF Loan_Status = "N" THEN Loan_Status_Encoded = 0;

    /* Encoding Self_Employed */
    IF Self_Employed = "Yes" THEN Self_Employed_Encoded = 1;
    ELSE IF Self_Employed = "No" THEN Self_Employed_Encoded = 0;

    /* Encoding Education */
    IF Education = "Graduate" THEN Education_Encoded = 1;
    ELSE IF Education = "Not Graduate" THEN Education_Encoded = 0;

    /* Similarly, encode other categorical variables */
    IF Property_Area = "Urban" THEN Property_Area_Encoded = 1;
    ELSE IF Property_Area = "Semiurban" THEN Property_Area_Encoded = 2;
    ELSE IF Property_Area = "Rural" THEN Property_Area_Encoded = 3;
RUN;
```

Gender_Encoded	Married_Encoded	Loan_Status_Encoded	Self_Employed_Encoded	Education_Encoded	Property_Area_Encoded
1	1	1	1	1	2
1	1	1	0	1	2
1	1	0	0	1	2
1	1	0	0	1	1

4. Machine Learning

- Classification Model
  - Logistic Regression
  - Decision Tree

- Regression Model
  - Random Forest
  - Linear Regression

SPLIT THE DATA

This code is used to split a dataset into training and testing subsets, which is essential for building and evaluating predictive models.

1.Random Sampling (PROC SURVEYSELECT):

Selects 80% of the data for training using a random seed (seed=42) for consistent results.
Adds a selected column indicating if a record is for training (1) or testing (0).
Data Separation (DATA Step):

2.Divides the sampled data:
Training records (selected=1) go to loan_train_data.
Testing records (selected=0) go to loan_test_data.

```
/*  Split the data into training and testing datasets */
proc surveyselect data=label_encoded_data out=loan_train samprate=0.8 seed=42 outall;
run;
data loan_train_data loan_test_data;
    set loan_train;
    if selected = 1 then output loan_train_data;
    else output loan_test_data;
```

The SURVEYSELECT Procedure

Selection Method	Simple Random Sampling

Input Data Set	LABEL_ENCODED_DATA
Random Number Seed	42
Sampling Rate	0.8
Sample Size	703
Selection Probability	0.800683
Sampling Weight	0
Output Data Set	LOAN_TRAIN

LOGISTIC REGRESSION

```
proc logistic data=loan_train_data outmodel=logistic_model;
    class Gender_Encoded Married_Encoded Education_Encoded
          Self_Employed_Encoded Property_Area_Encoded / param=ref;
    model Loan_Status_Encoded(event='1') = Scaled_LoanAmount Scaled_ApplicantIncome
          Scaled_CoapplicantIncome Gender_Encoded Married_Encoded Education_Encoded
          Self_Employed_Encoded Property_Area_Encoded;
run;
/* Scoring the test data */
proc logistic inmodel=logistic_model;
    score data= loan_test_data out=logistic_predictions;
run;
```

This code performs **logistic regression** for predicting loan status:

1. Logistic Regression Model (`PROC LOGISTIC`):

Encoded categorical variables (e.g., Gender_Encoded, Married_Encoded) are included with reference parameterization (param=ref).

Scaled numerical features (e.g., Scaled_LoanAmount, Scaled_ApplicantIncome) are used as predictors.

The model predicts Loan_Status_Encoded (event='1') and saves the trained model as logistic_model.

2. Scoring the Test Data:

The trained model (logistic_model) is applied to the test dataset (loan_test_data) to generate predictions.

Outputs predictions to logistic_predictions for further evaluation.

The LOGISTIC Procedure

Model Information	
Data Set	WORK.LOAN_TRAIN_DATA
Response Variable	Loan_Status_Encoded
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	703
Number of Observations Used	703

Response Profile		
Ordered Value	Loan_Status_Encoded	Total Frequency
1	0	209
2	1	494

Probability modeled is Loan_Status_Encoded=1.

Class Level Information			
Class	Value	Design Variables	
Gender_Encoded	0	1	
	1	0	
Married_Encoded	0	1	
	1	0	
Education_Encoded	0	1	
	1	0	
Self_Employed_Encoded	0	1	
	1	0	
Property_Area_Encoded	1	1	0
	2	0	1
	3	0	0

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

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	857.631	837.827
SC	862.186	883.380
-2 Log L	855.631	817.827

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	37.8041	9	<.0001
Score	37.4733	9	<.0001
Wald	35.4169	9	<.0001

Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
Scaled_LoanAmount	1	4.6683	0.0307
Scaled_ApplicantInco	1	2.2786	0.1312
Scaled_CoapplicantIn	1	1.3728	0.2413
Gender_Encoded	1	0.1826	0.6691
Married_Encoded	1	2.0737	0.1499
Education_Encoded	1	12.8122	0.0003
Self_Employed_Encode	1	2.9333	0.0868
Property_Area_Encode	2	14.8906	0.0006

Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	0.6100	0.4429	1.8974	0.1684
Scaled_LoanAmount		1	-1.3355	0.6181	4.6683	0.0307
Scaled_ApplicantInco		1	0.8982	0.5950	2.2786	0.1312
Scaled_CoapplicantIn		1	0.4928	0.4206	1.3728	0.2413
Gender_Encoded	0	1	-0.1023	0.2394	0.1826	0.6691
Married_Encoded	0	1	-0.2778	0.1929	2.0737	0.1499
Education_Encoded	0	1	-0.6962	0.1945	12.8122	0.0003
Self_Employed_Encode	0	1	0.4596	0.2683	2.9333	0.0868
Property_Area_Encode	1	1	0.1661	0.2067	0.6456	0.4217
Property_Area_Encode	2	1	0.7901	0.2130	13.7550	0.0002

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
Scaled_LoanAmount	0.263	0.078	0.883
Scaled_ApplicantInco	2.455	0.765	7.881
Scaled_CoapplicantIn	1.637	0.718	3.732
Gender_Encoded 0 vs 1	0.903	0.565	1.443
Married_Encoded 0 vs 1	0.757	0.519	1.106
Education_Encoded 0 vs 1	0.498	0.340	0.730
Self_Employed_Encode 0 vs 1	1.583	0.936	2.679
Property_Area_Encode 1 vs 3	1.181	0.787	1.770
Property_Area_Encode 2 vs 3	2.204	1.451	3.346

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	64.2	Somers' D	0.284
Percent Discordant	35.8	Gamma	0.284
Percent Tied	0.0	Tau-a	0.119
Pairs	103246	c	0.642

Obs	Selected	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status	Age	_TYPE_	_
1	0	LP001786	Male	Yes	0	Graduate	No	5449.5581322	0	257.22044908	360	1	Urban	N	20	0	
2	0	LP002983	Male	Yes	1	Graduate	No	8072	240	253	360	1	Urban	Y	34	0	
3	0	LP002529	Male	Yes	2	Graduate	No	6728.4970911	1869.3050528	252.03882715	300	1	Semiurban	Y	40	0	
4	0	LP001713	Male	Yes	1	Graduate	Yes	7787	0	240	360	1	Urban	Y	44	0	
5	0	LP001519	Female	No	0	Graduate	No	10052.805322	1603.2971355	239.49464223	360	1	Rural	N	40	0	
6	0	LP002170	Male	Yes	2	Graduate	No	5000	3667	236	360	1	Semiurban	Y	50	0	
7	0	LP001531	Male	No	0	Graduate	No	8995.3436736	0	231.28809829	360	1	Urban	N	28	0	

EVALUATE THE MODEL (LOGISTIC REGRESSION)

This code assesses the logistic regression model's performance by comparing actual
and predicted outcomes:
PROC FREQ: Generates a contingency table showing the relationship between the actual loan status (Loan_Status_Encode
and the predicted loan status (I_Loan_Status_Encoded) in the test results (logistic_predictions)

```
/* Evaluate model performance */
proc freq data=logistic_predictions;
    tables Loan_Status_Encoded*I_Loan_Status_Encoded / norow nocol nopercent;
run;
```

The FREQ Procedure

Frequency

Table of Loan_Status_Encoded by I_Loan_Status_Encoded			
	I_Loan_Status_Encoded(Into: Loan_Status_Encoded)		
Loan_Status_Encoded	0	1	Total
0	3	54	57
1	5	113	118
Total	8	167	175

DECISION TREE

This code builds a **decision tree** model for classifying loan statuses:

1. Model Creation (`PROC HPSPLIT`):

The target variable (Loan_Status_Encoded) is defined as categorical using class.

Predictors include scaled numerical features (e.g., Scaled_LoanAmount) and encoded categorical variables (e.g., Gender_Encoded).

2. Tree Growth:

grow entropy: Builds the tree using the entropy criterion to optimize splits for classification.

prune costcomplexity: Prunes the tree to balance complexity and performance.

```
 Decision Tree for Classification */
roc hpsplit data=loan_train_data;
   class Loan_Status_Encoded /* Define the target variable as categorical */
         Gender_Encoded Married_Encoded Education_Encoded
         Self_Employed_Encoded Property_Area_Encoded;
   model Loan_Status_Encoded = Scaled_LoanAmount Scaled_ApplicantIncome
         Scaled_CoapplicantIncome Gender_Encoded Married_Encoded Education_Encoded
         Self_Employed_Encoded Property_Area_Encoded;
   grow entropy; /* Use ENTROPY for classification */
   prune costcomplexity;
   code file='/home/u64078764/excel files/decision_tree_code.sas';
n;
```

The output

The HPSPLIT Procedure

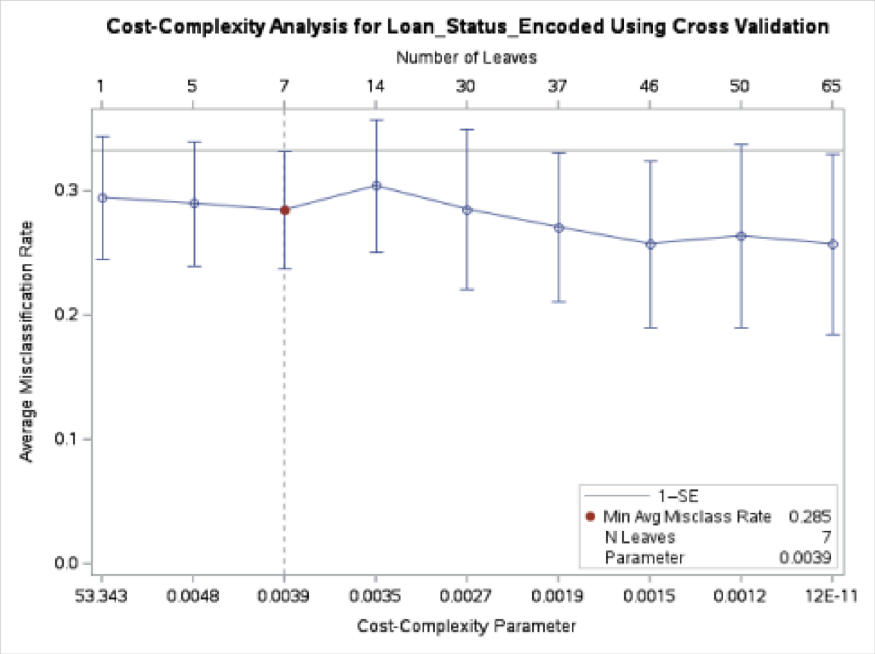
Performance Information	
Execution Mode	Single-Machine
Number of Threads	2

Data Access Information			
Data	Engine	Role	Path
WORK.LOAN_TRAIN_DATA	V9	Input	On Client

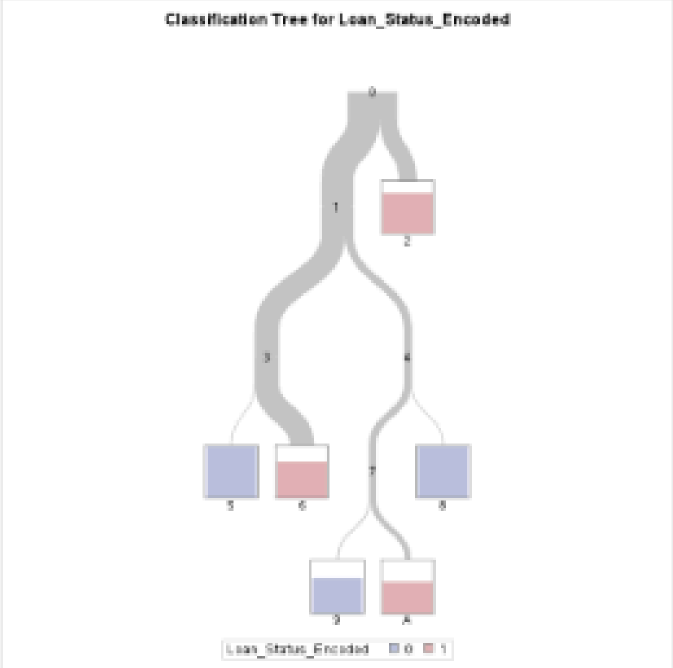
Model Information	
Split Criterion Used	Entropy
Pruning Method	Cost-Complexity
Subtree Evaluation Criterion	Cost-Complexity
Number of Branches	2
Maximum Tree Depth Requested	10
Maximum Tree Depth Achieved	10
Tree Depth	4
Number of Leaves Before Pruning	71
Number of Leaves After Pruning	6
Model Event Level	0

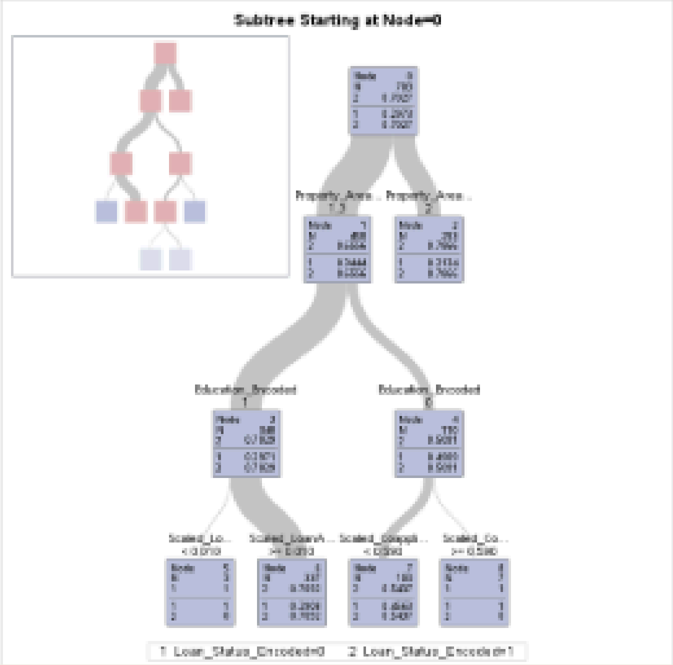
Number of Observations Read	703
Number of Observations Used	703

The HPSPLIT Procedure



The HPSPLIT Procedure

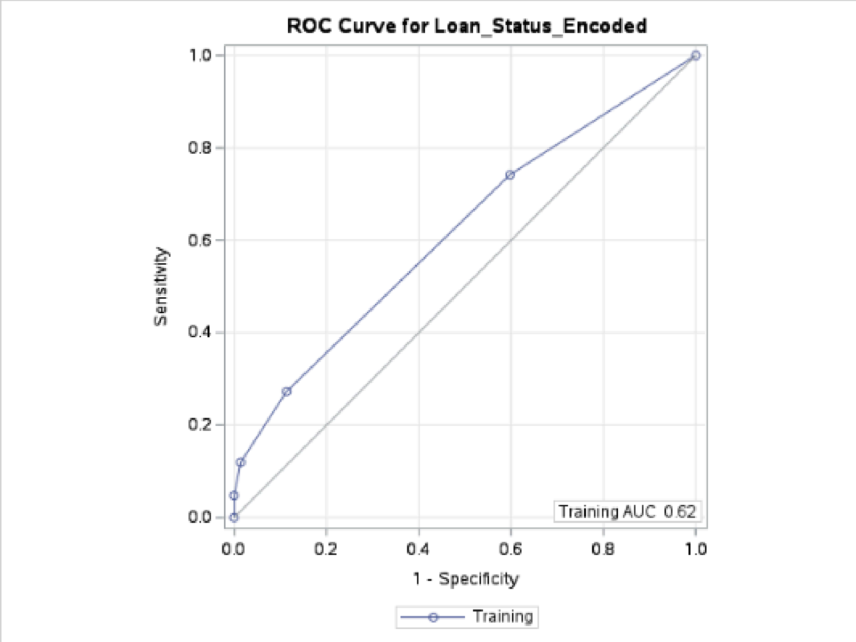




The HP SPLIT Procedure

Model-Based Confusion Matrix			
	Predicted		
Actual	0	1	Error Rate
0	25	184	0.8804
1	7	487	0.0142

Model-Based Fit Statistics for Selected Tree								
N Leaves	ASE	Mis-class	Sensitivity	Specificity	Entropy	Gini	RSS	AUC
6	0.1936	0.2717	0.1196	0.9858	0.8259	0.3872	272.2	0.6168



Variable Importance			
	Training		
Variable	Relative	Importance	Count
Education_Encoded	1.0000	2.4993	1
Scaled_LoanAmount	0.9666	2.4159	2
Property_Area_Encoded	0.9434	2.3577	1
Scaled_CoapplicantIncome	0.7876	1.9685	1

Apply the model on test data

```
data decision_tree_predictions;
    set loan_test_data;
    %include '/home/u64078764/excel files/decision_tree_code.sas';
run;
```

Obs	Selected	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_St
1	0	LP001786	Male	Yes	0	Graduate	No	5449.5581322	0	257.22044908	360	1	Urban	N
2	0	LP002983	Male	Yes	1	Graduate	No	8072	240	253	360	1	Urban	Y
3	0	LP002529	Male	Yes	2	Graduate	No	6726.4970911	1869.3050528	252.03882715	300	1	Semiurban	Y
4	0	LP001713	Male	Yes	1	Graduate	Yes	7787	0	240	360	1	Urban	Y
5	0	LP001519	Female	No	0	Graduate	No	10052.805322	1603.2971355	239.49464223	360	1	Rural	N
6	0	LP002170	Male	Yes	2	Graduate	No	5000	3667	236	360	1	Semiurban	Y
7	0	LP001531	Male	No	0	Graduate	No	8995.3436736	0	231.28809629	360	1	Urban	N
8	0	LP002328	Male	Yes	0	Not Graduate	No	6096	0	218	360	0	Rural	N

THE FIRST FEW ROWS TO CONFIRM PREDICTIONS EXIST

```
proc print data=decision_tree_predictions(obs=10);
run;
```

Obs	Selected	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status	Age	_TYPE_	_FREQ_	Q1_LoanAmount	Q1_ApplicantIncome	Q1_CoapplicantIncome
1	0	LP001786	Male	Yes	0	Graduate	No	5449.5581322	0	257.22044908	360	1	Urban	N	20	0	1014	100	2787	0
2	0	LP002983	Male	Yes	1	Graduate	No	8072	240	253	360	1	Urban	Y	34	0	1014	100	2787	0
3	0	LP002529	Male	Yes	2	Graduate	No	6726.4970911	1869.3050528	252.03882715	300	1	Semiurban	Y	40	0	1014	100	2787	0
4	0	LP001713	Male	Yes	1	Graduate	Yes	7787	0	240	360	1	Urban	Y	44	0	1014	100	2787	0
5	0	LP001519	Female	No	0	Graduate	No	10052.805322	1603.2971355	239.49464223	360	1	Rural	N	40	0	1014	100	2787	0
6	0	LP002170	Male	Yes	2	Graduate	No	5000	3667	236	360	1	Semiurban	Y	50	0	1014	100	2787	0
7	0	LP001531	Male	No	0	Graduate	No	8995.3436736	0	231.28809629	360	1	Urban	N	28	0	1014	100	2787	0
8	0	LP002328	Male	Yes	0	Not Graduate	No	6096	0	218	360	0	Rural	N	20	0	1014	100	2787	0
9	0	LP002160	Male	Yes	3+	Graduate	No	5613.4578442	3136.4496477	217.32453325	360	1	Semiurban	Y	26	0	1014	100	2787	0
10	0	LP001379	Male	Yes	2	Graduate	No	3800	3600	216	360	0	Urban	N	37	0	1014	100	2787	0

Q3_LoanAmount	Q3_ApplicantIncome	Q3_CoapplicantIncome	IQR_LoanAmount	LoanAmount_Low	LoanAmount_High	IQR_ApplicantIncome	ApplicantIncome_Low	ApplicantIncome_High	IQR_CoapplicantIncome	CoapplicantIncome_Low	CoapplicantIncome_High	min_LoanAmount
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.24293

min_LoanAmount	max_LoanAmount	min_ApplicantIncome	max_ApplicantIncome	min_CoapplicantIncome	max_CoapplicantIncome	Scaled_LoanAmount	Scaled_ApplicantIncome	Scaled_CoapplicantIncome	Gender_Encoded	Married_Encoded	Loan_Status_Encoded	Self_Employed_Encoded
8.24293	260	147.765	10139	0	5625	0.98896	0.53064	0.00000	1	1	0	0
8.24293	260	147.765	10139	0	5625	0.97220	0.79312	0.04267	1	1	1	0
8.24293	260	147.765	10139	0	5625	0.96838	0.65845	0.33232	1	1	1	0
8.24293	260	147.765	10139	0	5625	0.92056	0.76459	0.00000	1	1	1	1
8.24293	260	147.765	10139	0	5625	0.91855	0.99137	0.28503	0	0	0	0
8.24293	260	147.765	10139	0	5625	0.90467	0.48565	0.65191	1	1	1	0
8.24293	260	147.765	10139	0	5625	0.88595	0.88553	0.00000	1	0	0	0
8.24293	260	147.765	10139	0	5625	0.83317	0.59535	0.00000	1	1	0	0
8.24293	260	147.765	10139	0	5625	0.83049	0.54705	0.55759	1	1	1	0
8.24293	260	147.765	10139	0	5625	0.82523	0.36554	0.64000	1	1	0	0

1	1	6	2		0.29080	0.70920
1	1	6	2		0.29080	0.70920
1	2	2	0		0.21344	0.78656
1	1	6	2		0.29080	0.70920
1	3	6	2		0.29080	0.70920
1	2	2	0		0.21344	0.78656
1	1	6	2		0.29080	0.70920
0	3	10	5		0.39506	0.60494
1	2	2	0		0.21344	0.78656
1	1	6	2		0.29080	0.70920

EVALUATE THE MODEL (DECISION TREE)

```
proc freq data=decision_tree_predictions;
    tables Loan_Status_Encoded*P_Loan_Status_Encoded0*P_Loan_Status_Encoded1 / norow nocol nopercent;
run;
```

evaluates the performance of the decision tree model by analyzing predicted probabilities and actual loan statuses:

`PROC FREQ`:

Creates a multi-dimensional table comparing:

Actual loan status (Loan_Status_Encoded).

Predicted probability of class 0 (P_Loan_Status_Encoded0).

Predicted probability of class 1 (P_Loan_Status_Encoded1)

The FREQ Procedure

Frequency

Table 1 of P_Loan_Status_Encoded0 by P_Loan_Status_Encoded1

Controlling for Loan_Status_Encoded=0

	P_Loan_Status_Encoded1(Predicted: Loan_Status_Encoded=1)					
P_Loan_Status_Encoded0(Predicted: Loan_Status_Encoded=0)	0	0.31818182	0.60493827	0.70919881	0.78656126	Total
0.21343874	0	0	0	0	13	13
0.29080119	0	0	0	29	0	29
0.39506173	0	0	10	0	0	10
0.68181818	0	4	0	0	0	4
1	1	0	0	0	0	1
Total	1	4	10	29	13	57

Frequency

Table 2 of P_Loan_Status_Encoded0 by P_Loan_Status_Encoded1

Controlling for Loan_Status_Encoded=1

	P_Loan_Status_Encoded1(Predicted: Loan_Status_Encoded=1)					
P_Loan_Status_Encoded0(Predicted: Loan_Status_Encoded=0)	0	0.31818182	0.60493827	0.70919881	0.78656126	Total
0.21343874	0	0	0	0	52	52
0.29080119	0	0	0	51	0	51
0.39506173	0	0	14	0	0	14
0.68181818	0	1	0	0	0	1
1	0	0	0	0	0	0
Total	0	1	14	51	52	118

RANDOM FOREST

implements a **Random Forest** model to predict loan approval:

1. Model Training (`PROC HPFOREST`):
 Specifies Loan_Status_Encoded as the binary target variable (level=binary).

2. Model Configuration:
 maxtrees=100: Limits the forest to 100 decision trees for classification.

3. Input Variables:
Uses numerical (e.g., ApplicantIncome, LoanAmount) and encoded categorical variables (e.g., Gender_Encoded) as predictors.

```
proc hpforest data=loan_test_data maxtrees=100 seed=42;
    target Loan_Status_Encoded / level=binary;
    input ApplicantIncome CoapplicantIncome LoanAmount Loan_Amount_Term Credit_History Age
          Gender_Encoded Married_Encoded Education_Encoded Self_Employed_Encoded Property_Area_Encoded;
    score out=loan_predictions;
run;
```

The HPFOREST Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	2

Data Access Information			
Data	Engine	Role	Path
WORK.LOAN_TEST_DATA	V9	Input	On Client
WORK.LOAN_PREDICTIONS	V9	Output	On Client

Model Information		
Parameter	Value	
Variables to Try	3	(Default)
Maximum Trees	100	
Actual Trees	100	
Inbag Fraction	0.6	(Default)
Prune Fraction	0	(Default)
Prune Threshold	0.1	(Default)
Leaf Fraction	0.00001	(Default)
Leaf Size Setting	1	(Default)
Leaf Size Used	1	
Category Bins	30	(Default)
Interval Bins	100	
Minimum Category Size	5	(Default)
Node Size	100000	(Default)
Maximum Depth	20	(Default)
Alpha	1	(Default)
Exhaustive	5000	(Default)
Rows of Sequence to Skip	5	(Default)
Split Criterion	.	Gini
Preselection Method	.	BinnedSearch
Missing Value Handling	.	Valid value

Number of Observations	
Type	N
Number of Observations Read	175
Number of Observations Used	175

Baseline Fit Statistics	
Statistic	Value
Average Square Error	0.220
Misclassification Rate	0.326
Log Loss	0.631

Fit Statistics							
Number of Trees	Number of Leaves	Average Square Error (Train)	Average Square Error (OOB)	Misclassification Rate (Train)	Misclassification Rate (OOB)	Log Loss (Train)	Log Loss (OOB)
1	37	0.1273	0.294	0.14857	0.329	2.553	6.315
2	66	0.0788	0.236	0.11429	0.310	0.697	4.599
3	98	0.0702	0.260	0.09714	0.331	0.449	4.990
4	133	0.0584	0.231	0.05714	0.322	0.200	3.496
5	170	0.0519	0.207	0.05714	0.244	0.183	3.190

Loss Reduction Variable Importance					
Variable	Number of Rules	Gini	OOB Gini	Margin	OOB Margin
Credit_History	201	0.111438	0.11635	0.222877	0.22608
Married_Encoded	144	0.020346	0.00069	0.040692	0.02590
Loan_Amount_Term	97	0.009455	-0.00221	0.018910	0.00457
Self_Employed_Encoded	93	0.006052	-0.00826	0.012103	-0.00139
Education_Encoded	185	0.010730	-0.00940	0.021461	0.00287
CoapplicantIncome	250	0.033109	-0.01165	0.066218	0.02284
Gender_Encoded	145	0.009959	-0.01472	0.019918	-0.00507
Property_Area_Encoded	327	0.027837	-0.01716	0.055674	0.01112
ApplicantIncome	482	0.065999	-0.03704	0.131997	0.02954
LoanAmount	468	0.057163	-0.05202	0.114327	-0.00023
Age	590	0.059272	-0.05873	0.118545	0.00077

EVALUATE THE MODEL (RANDOM FOREST)

```
/* Step 5: Evaluate the Model */
/* Print Predictions */
proc print data=loan_predictions;
    var  Loan_Status_Encoded I_Loan_Status_Encoded P_Loan_Status_Encoded1; /* Actual vs predicted */
run;
```

Display Predictions (`PROC PRINT`):
 Compares actual loan statuses (Loan_Status_Encoded), predicted classes (I_Loan_Status_Encoded), and predicted probabilities (P_Loan_Status_Encoded1).

Obs	Loan_Status_Encoded	I_Loan_Status_Encoded	P_Loan_Status_Encoded1
1	0	0	0.34547
2	1	1	0.88326
3	1	1	0.93556
4	1	1	0.82252
5	0	0	0.25333
6	1	1	0.88133

EVALUATE THE MODEL (RANDOM FOREST)

```
/* Step 6: Calculate Residuals */
data loan_predictions_with_residuals;
    set loan_predictions;
    residual = Loan_Status_Encoded - P_Loan_Status_Encoded1;
run;
```

Compute Residuals:
Adds a residual column to a new dataset, capturing the difference between actual loan status and predicted probabilities.

Obs	Loan_Status_Encoded	P_Loan_Status_Encoded0	P_Loan_Status_Encoded1	F_Loan_Status_Encoded	I_Loan_Status_Encoded	_WARN_	residual
1	0	0.65453	0.34547	0	0		-0.34547
2	1	0.11674	0.88326	1	1		0.11674
3	1	0.06444	0.93556	1	1		0.06444
4	1	0.17748	0.82252	1	1		0.17748
5	0	0.74667	0.25333	0	0		-0.25333
6	1	0.11867	0.88133	1	1		0.11867

EVALUATE THE MODEL (RANDOM FOREST)

```
/* Step 7: Model Performance Metrics */
proc means data=loan_predictions_with_residuals n mean std min max;
    var residual;
run;
```

Residual Analysis (`PROC MEANS`):

Calculates statistical metrics (mean, standard deviation, min, max) for residuals to summarize prediction errors.

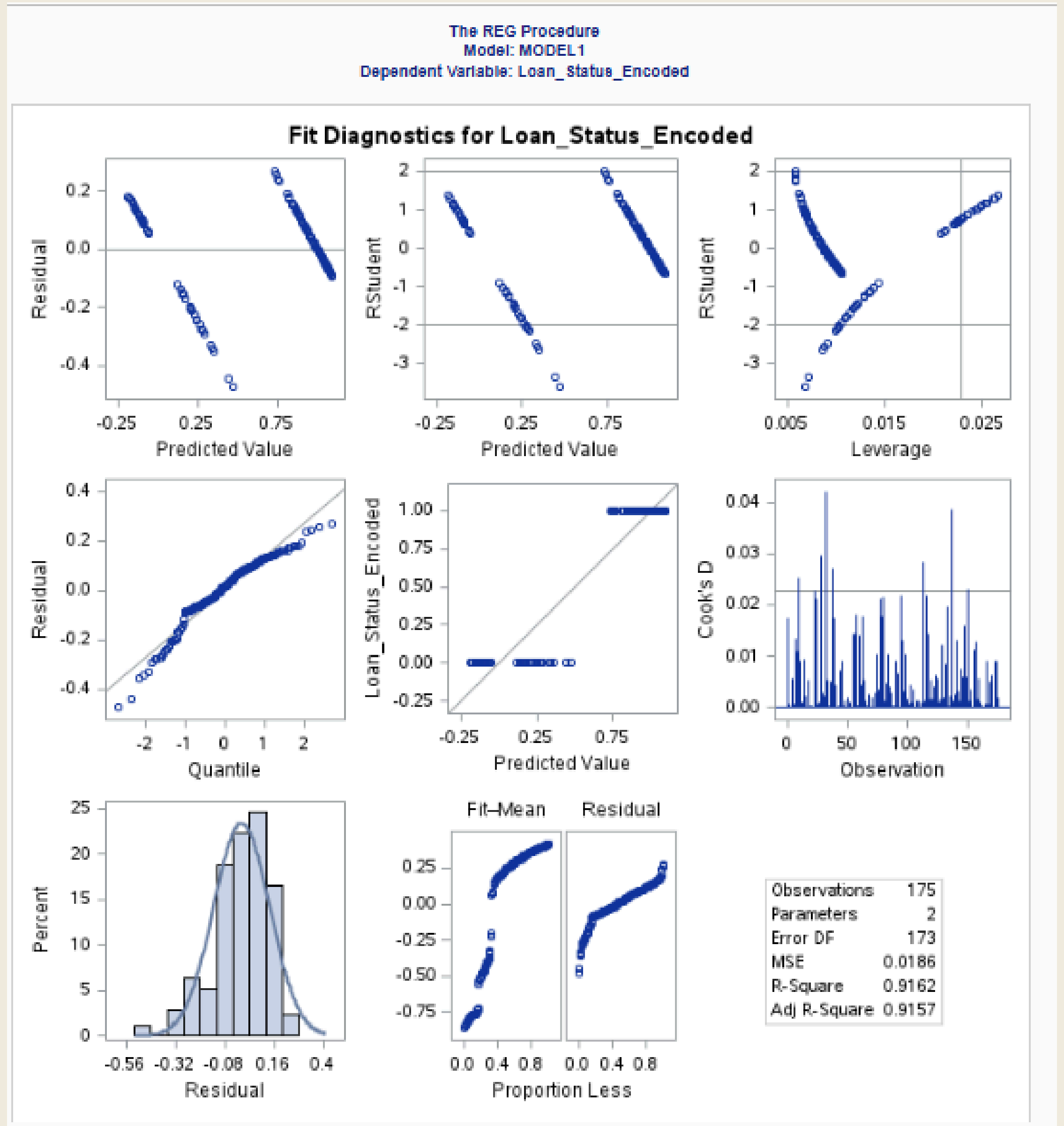
The MEANS Procedure

Analysis Variable : residual				
N	Mean	Std Dev	Minimum	Maximum
175	-0.0047738	0.1712230	-0.5259874	0.2773117

```
/* Step 8: Calculate R-Squared (R²) */
proc reg data=loan_predictions_with_residuals;
    model Loan_Status_Encoded = P_Loan_Status_Encoded1; /* Regression to calculate R² */
run;
```

Determine R-Squared (`PROC REG`):

Runs a regression of actual loan statuses on predicted probabilities to calculate R^2, indicating how well the model explains variability in the target.



The REG Procedure
Model: MODEL1
Dependent Variable: Loan_Status_Encoded

Number of Observations Read	175
Number of Observations Used	175

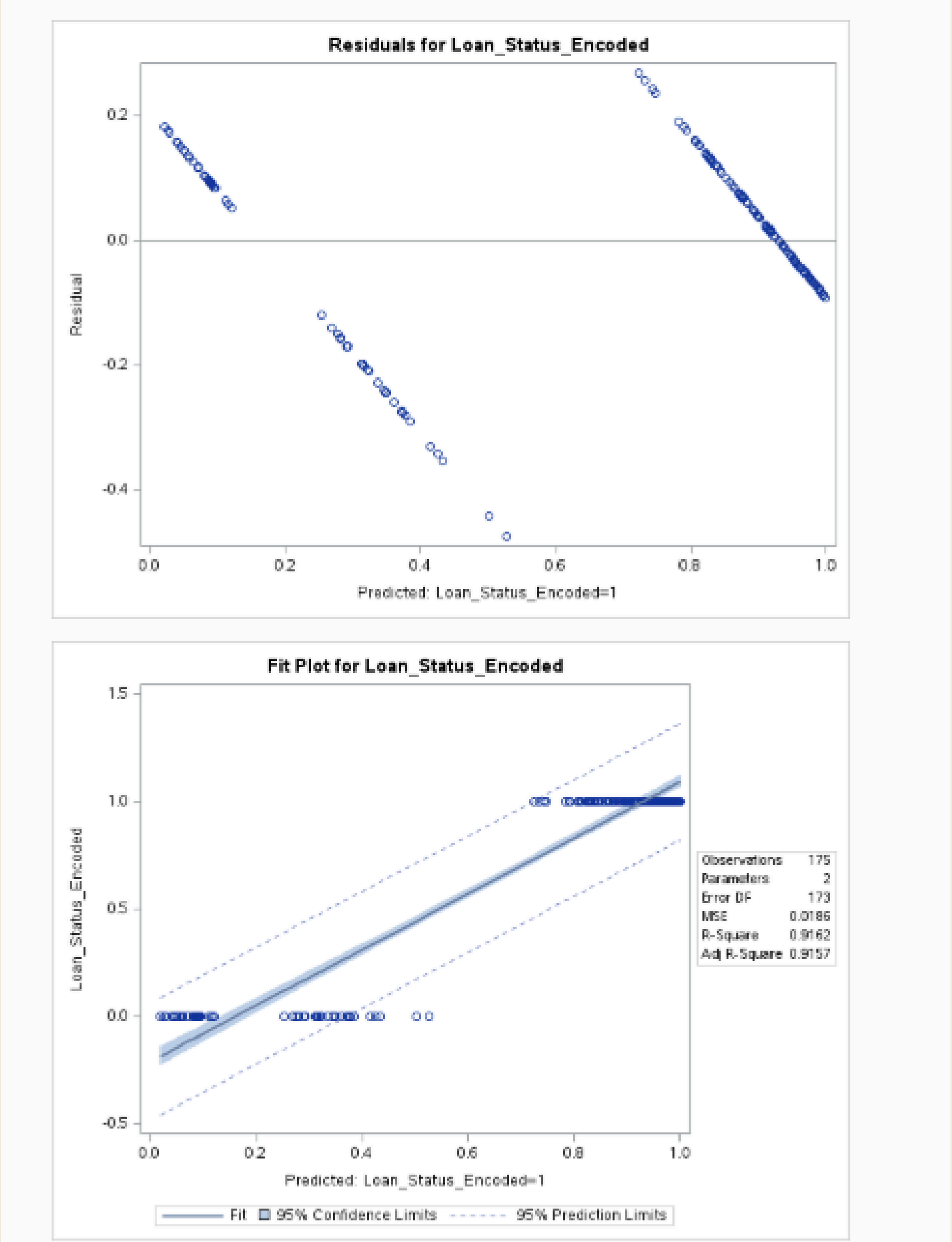
Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	35.21295	35.21295	1891.09	<.0001
Error	173	3.22134	0.01862		
Corrected Total	174	38.43429			

Root MSE	0.13646	R-Square	0.9162
Dependent Mean	0.67429	Adj R-Sq	0.9157
Coeff Var	20.23724		

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	Intercept	1	-0.20882	0.02278	-9.17	<.0001
P_Loan_Status_Encoded1	Predicted: Loan_Status_Encoded=1	1	1.30048	0.02991	43.49	<.0001



LINEAR REGRESSION

```
proc reg data=loan_train_data;

    model LoanAmount = Scaled_ApplicantIncome Scaled_CoapplicantIncome
          Gender_Encoded Married_Encoded Education_Encoded
          Self_Employed_Encoded Property_Area_Encoded;
           Property_Area_Encoded;
    output out=predicted_regression_results p=predicted_value r=residual;

run;
```

1.Train the Model (`PROC REG`):
 Builds a linear regression model with LoanAmount as the dependent variable.
 Predictors include scaled numerical features (e.g., Scaled_ApplicantIncome) and encoded categorical variables (e.g., Gender_Encoded).

2. Generate Outputs:
  Creates a new dataset (predicted_regression_results) with:
  Predicted loan amounts (predicted_value).
  Residuals (residual), representing the error between actual and predicted values

The REG Procedure
Model: MODEL1
Dependent Variable: LoanAmount

Number of Observations Read	703
Number of Observations Used	703

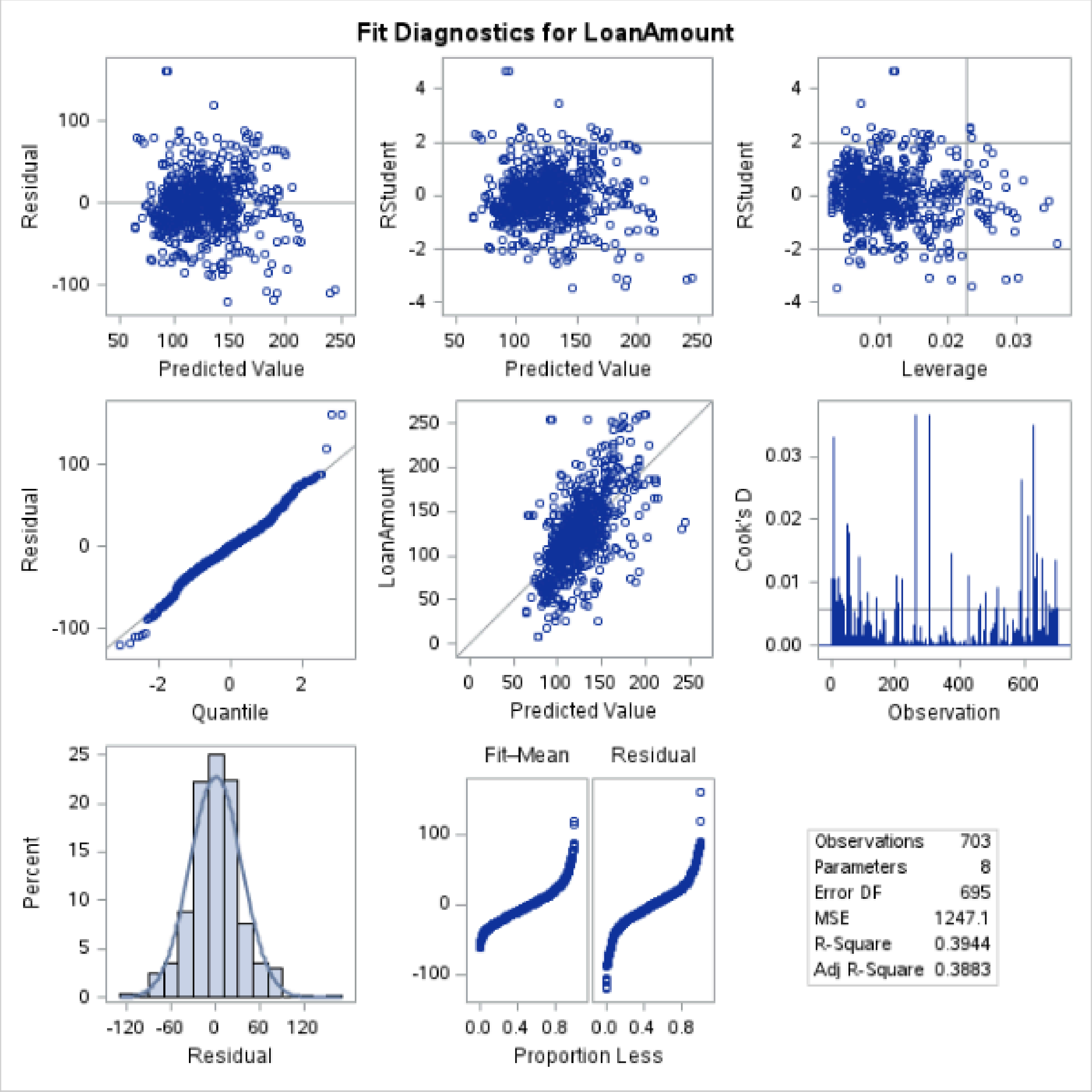
Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	564493	80642	64.66	<.0001
Error	695	866730	1247.09390		
Corrected Total	702	1431224			

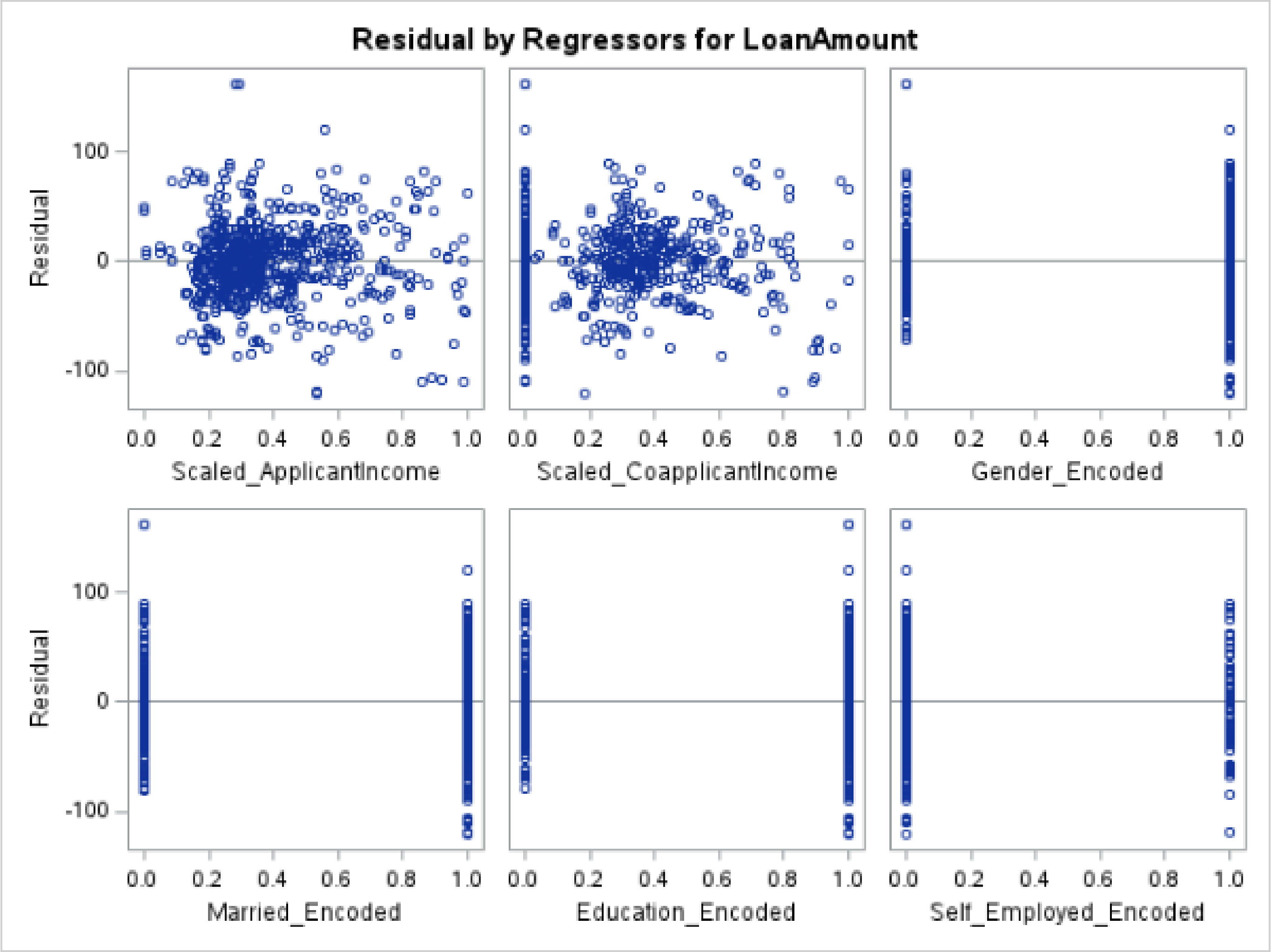
Root MSE	35.31422	R-Square	0.3944
Dependent Mean	125.69318	Adj R-Sq	0.3883
Coeff Var	28.09557		

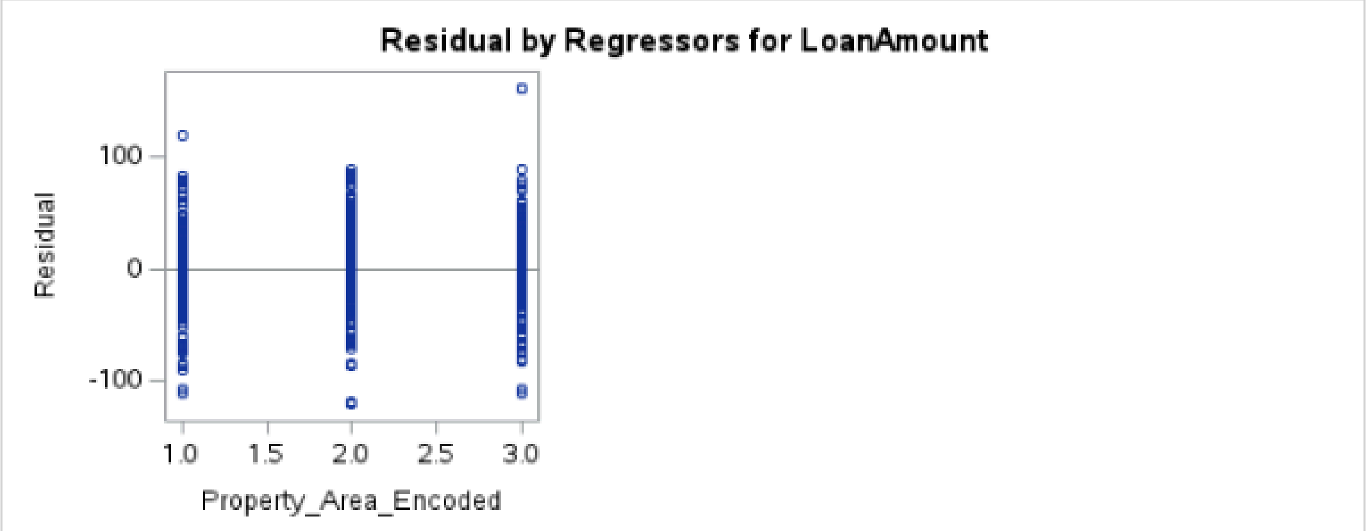
Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	40.20588	5.75158	6.99	<.0001
Scaled_ApplicantIncome	1	132.35861	7.66301	17.27	<.0001
Scaled_CoapplicantIncome	1	65.94327	5.92748	11.13	<.0001
Gender_Encoded	1	3.52416	3.74228	0.94	0.3467
Married_Encoded	1	9.20310	3.02496	3.04	0.0024
Education_Encoded	1	4.29328	3.20250	1.34	0.1805
Self_Employed_Encoded	1	1.23246	4.38229	0.28	0.7786
Property_Area_Encoded	1	3.38925	1.69601	2.00	0.0461

Output

The REG Procedure
Model: MODEL1
Dependent Variable: LoanAmount







EVALUATE MODEL (LINEAR REGRESSION)

```
proc means data=predicted_regression_results;
    var residual;
run;
```

Residual Statistics (`PROC MEANS`):
Computes metrics for residuals (e.g., mean, variance) to understand the error distribution and model fit.

The MEANS Procedure

Analysis Variable : residual Residual				
N	Mean	Std Dev	Minimum	Maximum
703	-5.87019E-14	35.1377075	-120.5829823	161.4961481

EVALUATE MODEL (LINEAR REGRESSION)

```
proc print data=predicted_regression_results;
    var LoanAmount predicted_value residual;
run;
```

(`PROC PRINT`):
Lists actual loan amounts (LoanAmount), predicted values (predicted_value), and residuals to visualize prediction accuracy

Obs	LoanAmount	predicted_value	residual
1	260	197.596	62.404
2	259	199.671	59.329
3	258	191.771	66.229
4	258	175.223	82.777
5	255	188.704	66.296
6	255	134.778	120.222

KNN MODEL

```
proc fastclus data=loan_train_data out=clustered_data maxclusters=2;
    var ApplicantIncome LoanAmount Gender_Encoded Married_Encoded;
run;
```

Clustering (`PROC FASTCLUS`):

Divides the loan_train_data into clusters, where maxclusters=2 sets two clusters.

Uses features such as ApplicantIncome, LoanAmount and encoded categorical variables (Gender_Encoded, Married_Encoded) to group similar records.

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=2 Maxiter=1

Initial Seeds				
Cluster	ApplicantIncome	LoanAmount	Gender_Encoded	Married_Encoded
1	10139.00000	260.00000	1.00000	1.00000
2	147.76480	138.11305	1.00000	1.00000

Criterion Based on Final Seeds =	532.8

Cluster Summary						
Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	157	715.4	3189.7		2	3738.1
2	546	468.4	3024.8		1	3738.1

Statistics for Variables				
Variable	Total STD	Within STD	R-Square	RSQ/(1-RSQ)
ApplicantIncome	1887	1066	0.681435	2.139074
LoanAmount	45.15285	40.92658	0.179608	0.218929
Gender_Encoded	0.38618	0.38442	0.010515	0.010627
Married_Encoded	0.48153	0.48182	0.000238	0.000238
OVER-ALL	943.84589	533.34172	0.681147	2.136246

Pseudo F Statistic =	1497.51

Approximate Expected Over-All R-Squared =	0.75029

Cubic Clustering Criterion =	-6.458

WARNING: The two values above are invalid for correlated variables.

Cluster Means				
Cluster	ApplicantIncome	LoanAmount	Gender_Encoded	Married_Encoded
1	6892.472063	161.353502	0.891720	0.649682
2	3154.647033	115.439200	0.796703	0.631868

Cluster Standard Deviations				
Cluster	ApplicantIncome	LoanAmount	Gender_Encoded	Married_Encoded
1	1430.055044	45.402664	0.311728	0.478596
2	935.932966	39.552212	0.402820	0.482740

EVALUATE MODEL (KNN MODEL)

Step 1
Cluster Summary Statistics

```
proc means data=clustered_data n mean std min max;
    class cluster; /* Cluster assignment variable from PROC FASTCLUS */
    var ApplicantIncome LoanAmount Gender_Encoded Married_Encoded;
run;
```

Cluster Summary (`PROC MEANS`):
Calculates summary statistics for different clusters created by the KNN model (cluster).
Computes mean, standard deviation, minimum, and maximum for features like ApplicantIncome, LoanAmount, and encoded categorical variables (Gender_Encoded, Married_Encoded).

This helps analyze and compare how variables differ across clusters, offering insights into group characteristics

The MEANS Procedure

Cluster	N Obs	Variable	N	Mean	Std Dev	Minimum	Maximum
1	157	ApplicantIncome	157	6892.47	1430.06	5105.17	10139.00
		LoanAmount	157	161.3535017	45.4026644	26.0000000	260.0000000
		Gender_Encoded	157	0.8917197	0.3117284	0	1.0000000
		Married_Encoded	157	0.6496815	0.4785963	0	1.0000000
2	546	ApplicantIncome	546	3154.65	935.9329661	147.7648000	5050.00
		LoanAmount	546	115.4392003	39.5522123	8.2429319	255.0000000
		Gender_Encoded	546	0.7967033	0.4028205	0	1.0000000
		Married_Encoded	546	0.6318681	0.4827397	0	1.0000000

EVALUATE MODEL (KNN MODEL)

Step 2:
Analyze the Distribution of Clusters

```
proc freq data=clustered_data;
    tables cluster / nocum nopercent; /* Check the number of observations in each cluster */
run;
```

Analyzes the distribution of clusters created by a KNN model:

Cluster Distribution (`PROC FREQ`):
Calculates the frequency of observations in each cluster (cluster).
Uses nocum and nopercent options to display only the count of observations per cluster without cumulative or percentage values.

The FREQ Procedure

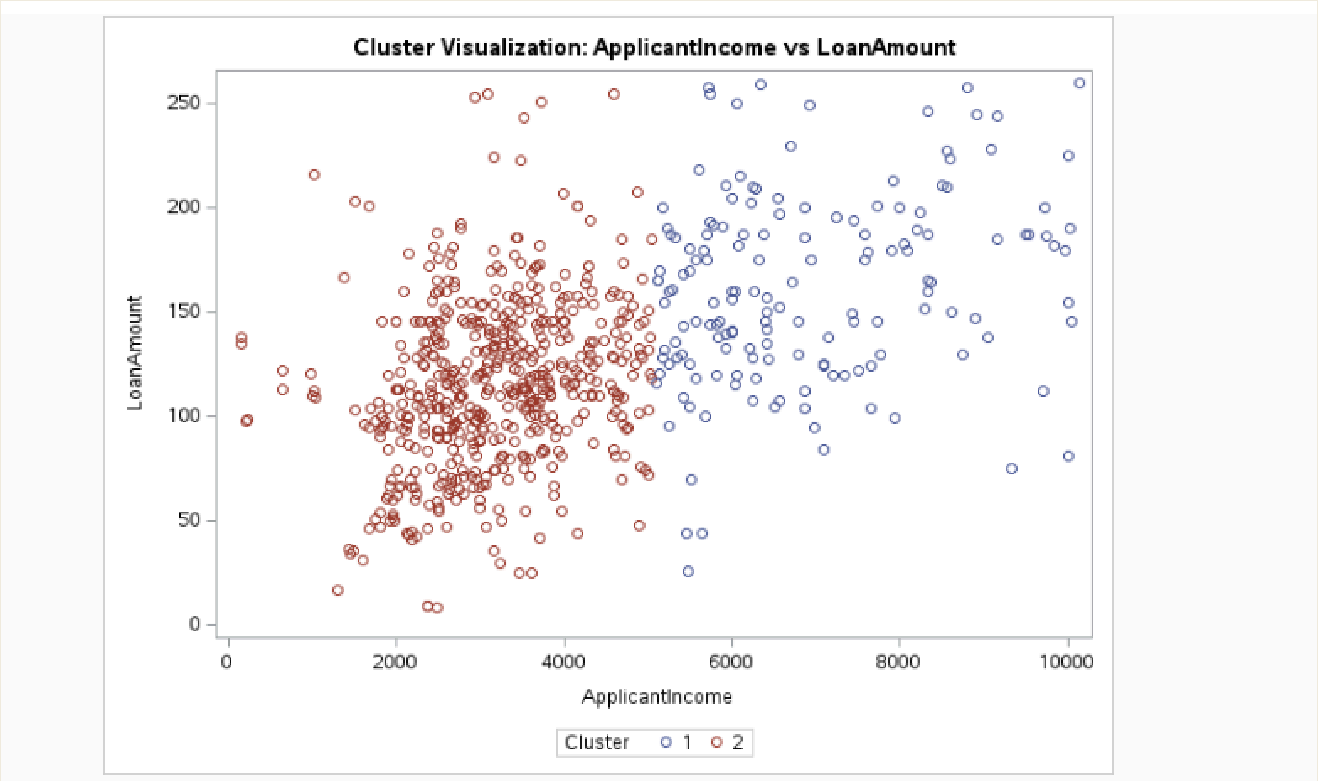
Cluster	
CLUSTER	Frequency
1	157
2	548

EVALUATE MODEL (KNN MODEL)

Step 3
Visualize the Clusters

```
proc sgplot data=clustered_data;
    scatter x=ApplicantIncome y=LoanAmount / group=cluster; /* Scatter plot for visual separation */
    title "Cluster Visualization: ApplicantIncome vs LoanAmount";
run;
```

1. Cluster Visualization (`PROC SGPLOT`):
Creates a scatter plot with ApplicantIncome on the x-axis and LoanAmount on the y-axis, colored by cluster (cluster).
Helps visually separate and compare clusters, showing how variables are distributed within different groups, helping to understand how observations are grouped based on their attributes.
This visualization aids in analyzing the relationship between key features and cluster separation.

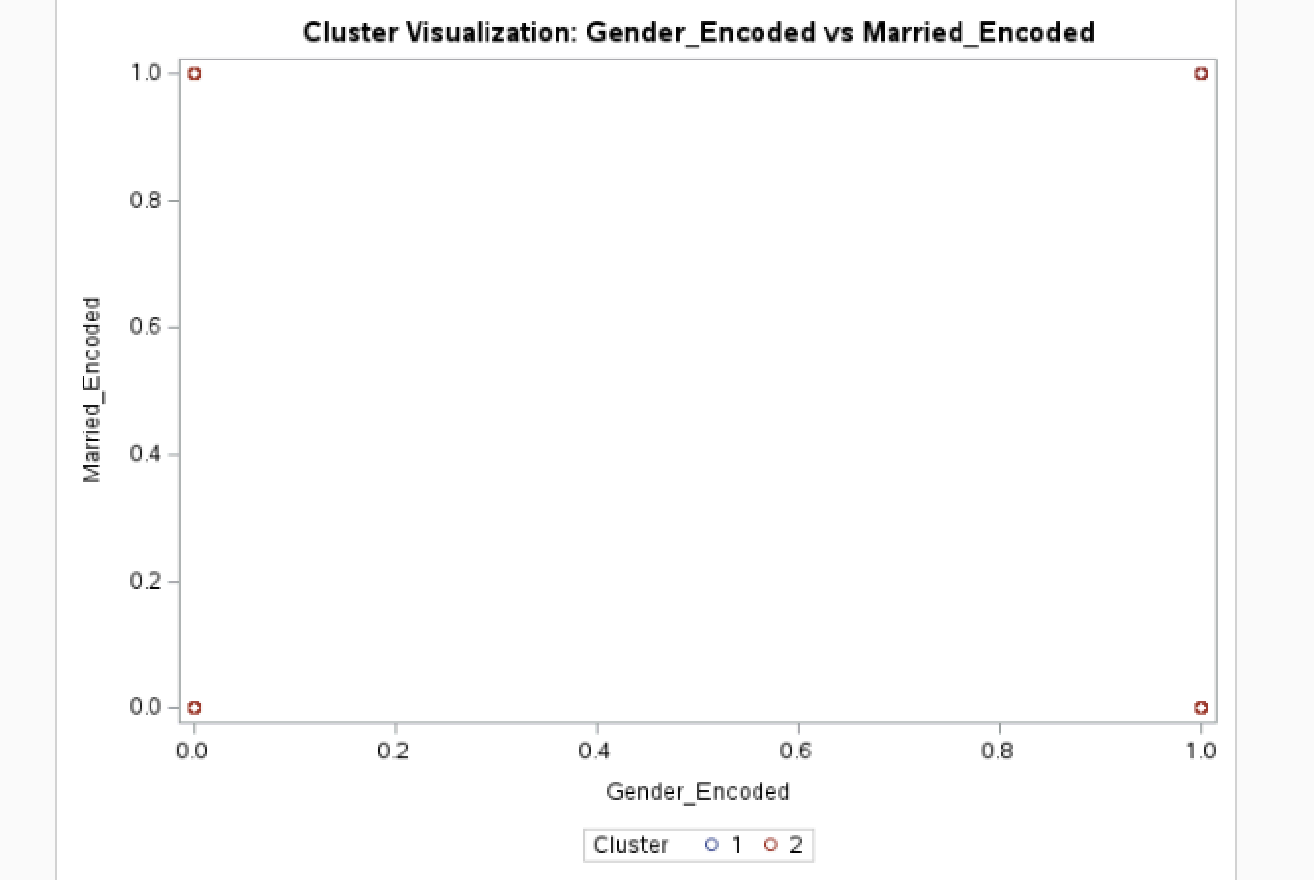


EVALUATE MODEL (KNN MODEL)

Step 3
Visualize the Clusters

2. Creates a scatter plot with Gender_Encoded on the x-axis and Married_Encoded on the y-axis, grouped by clusters (cluster)

```
proc sgplot data=clustered_data;
    scatter x=Gender_Encoded y=Married_Encoded / group=cluster; /* Scatter plot for categorical variables */
    title "Cluster Visualization: Gender_Encoded vs Married_Encoded";
run;
```



EVALUATE MODEL (KNN MODEL)

Step 4

Evaluate Cluster Separation with FASTCLUS

Cluster Separation (`PROC FASTCLUS`):
Re-applies clustering on the data (clustered_data) with maxclusters=2.
Uses variables like ApplicantIncome, LoanAmount,
and encoded categorical variables (Gender_Encoded,
Married_Encoded) to define clusters.

```
/* Step 4: Evaluate Cluster Separation with FASTCLUS */
proc fastclus data=clustered_data maxclusters=2 out=clustered_data_out;
    var ApplicantIncome LoanAmount Gender_Encoded Married_Encoded; /* Input variables */
run;
```

Initial Seeds

Cluster	ApplicantIncome	LoanAmount	Gender_Encoded	Married_Encoded
1	10139.00000	260.00000	1.00000	1.00000
2	147.76480	138.11305	1.00000	1.00000

Criterion Based on Final Seeds =	532.8

Cluster Summary

Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	157	715.4	3189.7		2	3738.1
2	546	468.4	3024.8		1	3738.1

Statistics for Variables

Variable	Total STD	Within STD	R-Square	RSQ/(1-RSQ)
ApplicantIncome	1887	1066	0.681435	2.139074
LoanAmount	45.15285	40.92658	0.179608	0.218929
Gender_Encoded	0.38618	0.38442	0.010515	0.010627
Married_Encoded	0.48153	0.48182	0.000238	0.000238
OVER-ALL	943.84589	533.34172	0.681147	2.136246

Pseudo F Statistic =	1497.51

Approximate Expected Over-All R-Squared =	0.75029

Cubic Clustering Criterion =	-6.458

WARNING: The two values above are invalid for correlated variables.

Cluster Means

Cluster	ApplicantIncome	LoanAmount	Gender_Encoded	Married_Encoded
1	6892.472063	161.353502	0.891720	0.649682
2	3154.647033	115.439200	0.796703	0.631868

Cluster Standard Deviations

Cluster	ApplicantIncome	LoanAmount	Gender_Encoded	Married_Encoded
1	1430.055044	45.402664	0.311728	0.478596
2	935.932966	39.552212	0.402820	0.482740

EVALUATE MODEL (KNN MODEL)

Cluster Summary (`PROC MEANS`):
Computes statistical measures (mean, standard deviation, minimum, maximum) for each cluster to assess intra-cluster characteristics.

```
/* Cluster Summary to check intra-cluster means and distances */
proc means data=clustered_data_out n mean std min max;
    class cluster; /* Cluster variable */
    var ApplicantIncome LoanAmount Gender_Encoded Married_Encoded;
run;
```

The MEANS Procedure

Cluster	N Obs	Variable	N	Mean	Std Dev	Minimum	Maximum
1	157	ApplicantIncome	157	6892.47	1430.06	5105.17	10139.00
		LoanAmount	157	161.3535017	45.4026644	26.0000000	260.0000000
		Gender_Encoded	157	0.8917197	0.3117284	0	1.0000000
		Married_Encoded	157	0.6496815	0.4785963	0	1.0000000
2	546	ApplicantIncome	546	3154.65	935.9329661	147.7648000	5050.00
		LoanAmount	546	115.4392003	39.5522123	8.2429319	255.0000000
		Gender_Encoded	546	0.7967033	0.4028205	0	1.0000000
		Married_Encoded	546	0.6318681	0.4827397	0	1.0000000

EVALUATE MODEL (KNN MODEL)

Cluster Distribution (`PROC FREQ`):
Counts the number of observations in each cluster to evaluate cluster sizes.
This process validates the clustering model's separation and provides insights into the characteristics and distribution of clusters

```
/* Check the number of observations per cluster */
proc freq data=clustered_data_out;
    tables cluster / nocum nopercent;
run;
```

The FREQ Procedure

Cluster	
CLUSTER	Frequency
1	157
2	546

Export After Changes

```
PROC EXPORT DATA=label_encoded_data
    OUTFILE="/home/u64078764/big data/loan_train_after_changes.xlsx"
    DBMS=xlsx
    REPLACE;
RUN;
```

DATA AFTER CHANGES

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Terr	Credit_History	Property_Area	Loan_Status	Age	_TYPE_	_FREQ_	Q1_LoanAmoun	Q1_ApplicantIncome	Q1_CoapplicantIncome
2	LP002795	Male	Yes	3+	Graduate	Yes	10139	0	260	360	1	Semiurban	Y	30	0	1014	100	2787	0
3	LP002137	Male	Yes	0	Graduate	No	6333	4583	259	360	1	Semiurban	Y	27	0	1014	100	2787	0
4	LP001225	Male	Yes	0	Graduate	No	5726	4595	258	360	1	Semiurban	N	48	0	1014	100	2787	0
5	LP002832	Male	Yes	2	Graduate	No	8799	0	258	360	0	Urban	N	47	0	1014	100	2787	0
6	LP001786	Male	Yes	0	Graduate	No	5449.558132	0	257.2204491	360	1	Urban	N	20	0	1014	100	2787	0
7	LP001552	Male	Yes	0	Graduate	No	4583	5625	255	360	1	Semiurban	Y	41	0	1014	100	2787	0
8	LP001786	Male	Yes	0	Graduate	No	5746	0	255	360	1	Urban	N	20	0	1014	100	2787	0
9	LP001846	Female	No	3+	Graduate	No	3083	0	255	360	1	Rural	Y	30	0	1014	100	2787	0
10	LP001846	Female	No	3+	Graduate	No	2940.561333	0	253.1605568	360	1	Rural	Y	30	0	1014	100	2787	0

Q3_LoanAmoun	Q3_ApplicantIncome	Q3_CoapplicantIncome	IQR_LoanAmoun	LoanAmount_Lov	LoanAmount_High	IQR_ApplicantIncome	ApplicantIncome_Low	ApplicantIncome_High	IQR_CoapplicantIncome	CoapplicantIncome_Low	CoapplicantIncome_High	min_LoanAmoun	max_LoanAmour
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260
164	5780	2250	64	4	260	2993	-1702.5	10269.5	2250	-3375	5625	8.242931852	260

	nin_ApplicantIncome	max_ApplicantIncome	min_CoapplicantIncome	max_CoapplicantIncome	Scaled_LoanAmount	Scaled_ApplicantIncome	Scaled_CoapplicantIncome	Gender_Encoded	Married_Encoded	Loan_Status_Encoded	Self_Employed_Encoded	Education_Encoded	Property_Area_Encoded
2	147.7648	10139	0	5625	1	1	0	1	1	1	1	1	2
3	147.7648	10139	0	5625	0.996027917	0.619066119	0.814755556	1	1	1	0	1	2
4	147.7648	10139	0	5625	0.992055834	0.55831287	0.816888889	1	1	0	0	1	2
5	147.7648	10139	0	5625	0.992055834	0.865882449	0	1	1	0	0	1	1
6	147.7648	10139	0	5625	0.988959393	0.530644432	0	1	1	0	0	1	1
7	147.7648	10139	0	5625	0.980139584	0.443912601	1	1	1	1	0	1	2
8	147.7648	10139	0	5625	0.980139584	0.560314625	0	1	1	0	0	1	1
9	147.7648	10139	0	5625	0.980139584	0.293781013	0	0	0	1	0	1	3
10	147.7648	10139	0	5625	0.972833163	0.279524651	0	0	0	1	0	1	3

THANK YOU