III. EDA-Data Cleaning

AIM:

- Handling missing values: detection, filling, and dropping
- Removing duplicates and unnecessary data
- Data type conversion and ensuring consistency
- Normalize data (e.g., standardization, min-max scaling).

PROCEDURE:

1. **Import required libraries and load the dataset using pandas.**
2. **Display initial data info and check for missing values.**
3. **Handle missing values and remove duplicates in the dataset.**
4. **Convert data types, standardize categorical values, and scale numeric columns.**
5. **Show the cleaned data summary to verify the results.**

PROGRAM AND OUTPUT:

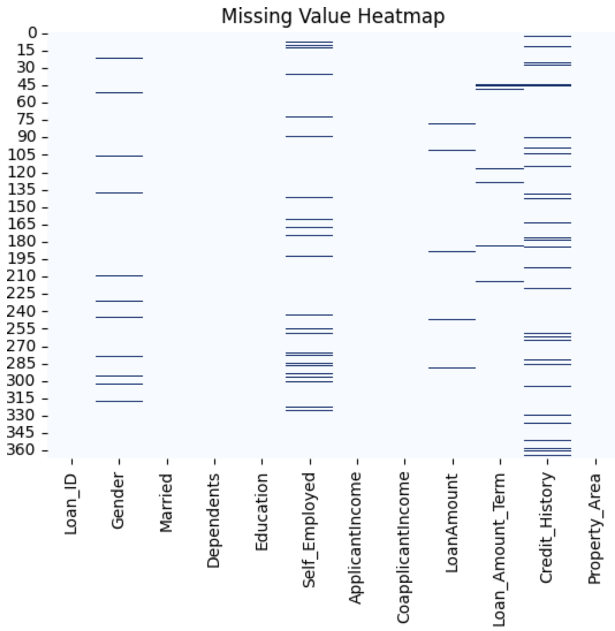
```
# Importing required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler, StandardScaler
# Load the dataset
df = pd.read_csv('/content/test_Y3wMUE5_7gLdaTN.csv')
# Display basic information
print("Initial Data Overview:")
print(df.info())
```

```
Initial Data Overview:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 367 entries, 0 to 366
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            367 non-null    object
 1   Gender             356 non-null    object
 2   Married            367 non-null    object
 3   Dependents         357 non-null    object
 4   Education          367 non-null    object
 5   Self_Employed      344 non-null    object
 6   ApplicantIncome    367 non-null    int64
 7   CoapplicantIncome  367 non-null    int64
 8   LoanAmount         362 non-null    float64
 9   Loan_Amount_Term   361 non-null    float64
 10  Credit_History     338 non-null    float64
 11  Property_Area      367 non-null    object
dtypes: float64(3), int64(2), object(7)
memory usage: 34.5+ KB
None
```

```
# 1. Handling Missing Values

print("\nMissing Values in Each Column:\n", df.isnull().sum())
sns.heatmap(df.isnull(), cbar=False, cmap="Blues")
plt.title("Missing Value Heatmap")
plt.show()
for col in ['Gender', 'Married', 'Dependents', 'Self_Employed']:
    df[col].fillna(df[col].mode()[0])
df['LoanAmount'].fillna(df['LoanAmount'].median())
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0])
df['Credit_History'].fillna(df['Credit_History'].mode()[0])
```

```
Missing Values in Each Column:
 Loan_ID              0
Gender               11
Married               0
Dependents            0
Education             0
Self_Employed        23
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount            5
Loan_Amount_Term      6
Credit_History       29
Property_Area         0
dtype: int64
```



	Credit_History
0	0.459858
1	0.459858
2	0.459858
3	0.459858
4	0.459858
...	...
362	0.459858
363	0.459858
364	0.459858
365	0.459858
366	0.459858

367 rows × 1 columns

dtype: float64

```
# 2. Removing Duplicates
initial_rows = df.shape[0]
df.drop_duplicates(inplace=True)
print(f"\nRemoved {initial_rows - df.shape[0]} duplicate rows.")
```

```
Removed 0 duplicate rows.
```

```
# 3. Data Type Conversion
# Convert 'Dependents' to numeric (replace '3+' with 3)
df['Dependents'] = df['Dependents'].replace('3+', 3).fillna(0).astype(int)
# 4. Ensuring Categorical Consistency
for col in ['Gender', 'Married', 'Education', 'Self_Employed', 'Property_Area']:
    df[col] = df[col].str.strip().str.capitalize()
# 5. Normalization
min_max_scaler = MinMaxScaler()
scale_cols = ['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount']
```

```
df[scale_cols] = min_max_scaler.fit_transform(df[scale_cols])
scaler = StandardScaler()
df[['Credit_History']] = scaler.fit_transform(df[['Credit_History']])
# 6. Final Overview
print("\nCleaned Data Summary:")
print(df.info())
print(df.head())
```

```
Cleaned Data Summary:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 367 entries, 0 to 366
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            367 non-null    object
 1   Gender             356 non-null    object
 2   Married            367 non-null    object
 3   Dependents         367 non-null    int64
 4   Education          367 non-null    object
 5   Self_Employed      344 non-null    object
 6   ApplicantIncome    367 non-null    float64
 7   CoapplicantIncome  367 non-null    float64
 8   LoanAmount         362 non-null    float64
 9   Loan_Amount_Term   361 non-null    float64
 10  Credit_History     338 non-null    float64
 11  Property_Area      367 non-null    object
dtypes: float64(5), int64(1), object(6)
memory usage: 34.5+ KB
None
    Loan_ID Gender Married  Dependents     Education Self_Employed  \
0  LP001015   Male     Yes           0      Graduate            No
1  LP001022   Male     Yes           1      Graduate            No
2  LP001031   Male     Yes           2      Graduate            No
3  LP001035   Male     Yes           2      Graduate            No
4  LP001051   Male      No           0  Not graduate            No

   ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0         0.078865           0.000000    0.157088             360.0
1         0.042411           0.062500    0.187739             360.0
2         0.068938           0.075000    0.344828             360.0
3         0.032263           0.106083    0.137931             360.0
4         0.045168           0.000000    0.095785             360.0

   Credit_History Property_Area
0        0.459858         Urban
1        0.459858         Urban
2        0.459858         Urban
3             NaN         Urban
4        0.459858         Urban
```

RESULT:

Thus, the program was written and executed successfully.