HOME LOAN APPROVAL

EXPLORATORY DATA ANALYSIS

- DATA EXPLORATION
- PATTERNS, TRENDS
- POTENTIAL INSIGHTS

MONU GUPTA monugupta8758@gmail.com 9310393905

Table of content

About the Loan

Loans are financial instruments provided by banks, financial institutions, or lenders, allowing individuals or organizations to borrow money with an agreement to repay it over a specific term, usually with interest. In the context of loan application datasets, each record represents a unique borrower's profile and the details of the loan they seek.

This project focuses on analyzing the factors that influence loan approval decisions using a dataset containing features like Applicant Income, Coapplicant Income, Loan Amount, Loan Term, Credit History, Education, Property Area, and more. By studying these variables, we can understand the pattern behind approved and rejected loans.

The main goal of this study is to build a data-driven model that can predict whether a loan application will be approved or not, based on the applicant's financial and personal information. This helps financial institutions make faster and more reliable decisions while ensuring fairness and efficiency in the loan approval process.

Objective

Purpose of the Dataset:

The primary aim of this dataset is to analyze borrower characteristics and assess their impact on loan approval outcomes. Understanding these factors is crucial for financial institutions as they strive to make informed lending decisions and enhance their risk assessment processes.

Analyze Borrower Profiles: Identify demographic trends and financial behaviors among applicants.

Evaluate Approval Criteria: Investigate how various features, such as income, credit history, and marital status, influence the likelihood of loan approval.

Support Predictive Modeling: Provide a foundation for building predictive models to forecast loan approvals and repayment behavior

TOOLS

python™

NumPy

pandas

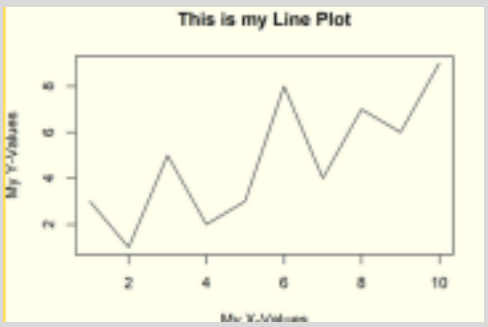
seaborn

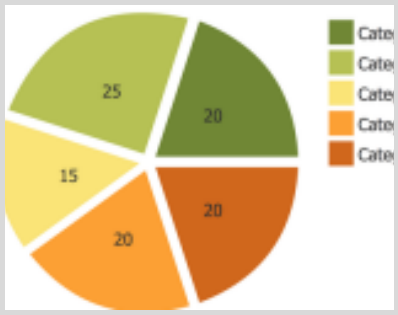
matplotlib

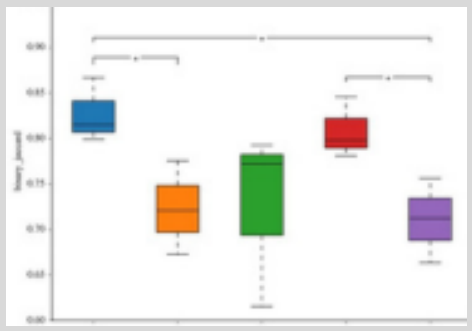
plotly

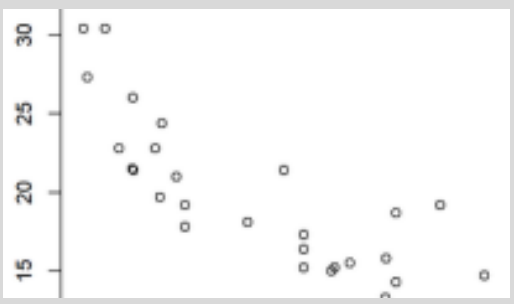
LOAN ANALYSIS

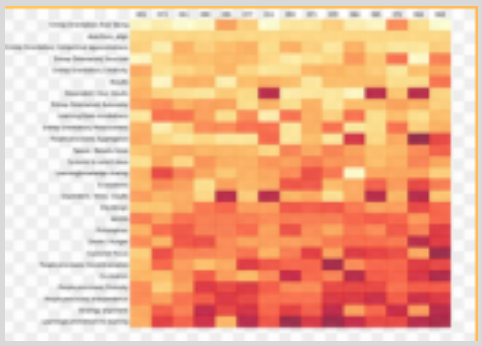
GRAPHS

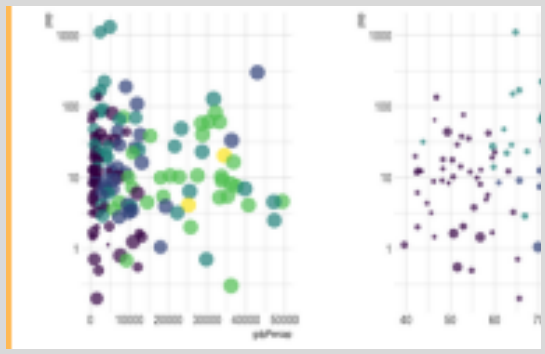












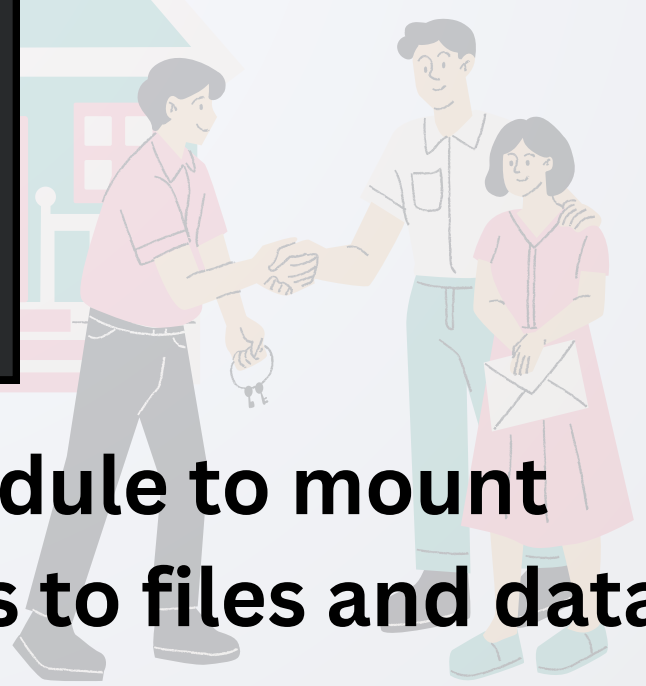
Data Overview

- Loading and preeparing loan application data for analysis in google Colab

```
[ ] from google.colab import drive
    drive.mount('/content/drive')

    Mounted at /content/drive
```

- In this analysis,we import Pandas for data manipulation ,Numpy for numerical operations ,Matplotlib for creating visualization and Seaborn and plotly for enhanced statistical graphic.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```



- We import the Google drive module to mount Google `Drive`, enabiling access to files and dataset in the for our analysis.

Data Overview

```
x = "/content/drive/MyDrive/Loan Approval EDA/loan_sanction_test.csv"

df= pd.read_csv(x)
```

- We use pandas to Read the CSV file containing big vehicle insurance from Google Drive, loading it into a DATAFRAME Name ‘df’ for analysis.

overview of dataset

```
df
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area
0	LP001015	Male	Yes	0	Graduate	No	5720	0	110.0	360.0	1.0	Urban
1	LP001022	Male	Yes	1	Graduate	No	3076	1500	126.0	360.0	1.0	Urban
2	LP001031	Male	Yes	2	Graduate	No	5000	1800	208.0	360.0	1.0	Urban
3	LP001035	Male	Yes	2	Graduate	No	2340	2546	100.0	360.0	NaN	Urban
4	LP001051	Male	No	0	Not Graduate	No	3276	0	78.0	360.0	1.0	Urban
...	...	...	...	...	...	...	...	...	...	...	...	...
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	1777	113.0	360.0	1.0	Urban
363	LP002975	Male	Yes	0	Graduate	No	4158	709	115.0	360.0	1.0	Urban
364	LP002980	Male	No	0	Graduate	No	3250	1993	126.0	360.0	NaN	Semiurban
365	LP002986	Male	Yes	0	Graduate	No	5000	2393	158.0	360.0	1.0	Rural
366	LP002989	Male	No	0	Graduate	Yes	9200	0	98.0	180.0	1.0	Rural

367 rows × 12 columns

The dataset comprises of 367 rows and 12 columns

Understanding the columns

- **First Few rows**

```
df.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area
0	LP001015	Male	Yes	0	Graduate	No	5720	0	110.0	360.0	1.0	Urban
1	LP001022	Male	Yes	1	Graduate	No	3076	1500	126.0	360.0	1.0	Urban
2	LP001031	Male	Yes	2	Graduate	No	5000	1800	208.0	360.0	1.0	Urban
3	LP001035	Male	Yes	2	Graduate	No	2340	2546	100.0	360.0	NaN	Urban
4	LP001051	Male	No	0	Not Graduate	No	3276	0	78.0	360.0	1.0	Urban

Description of columns

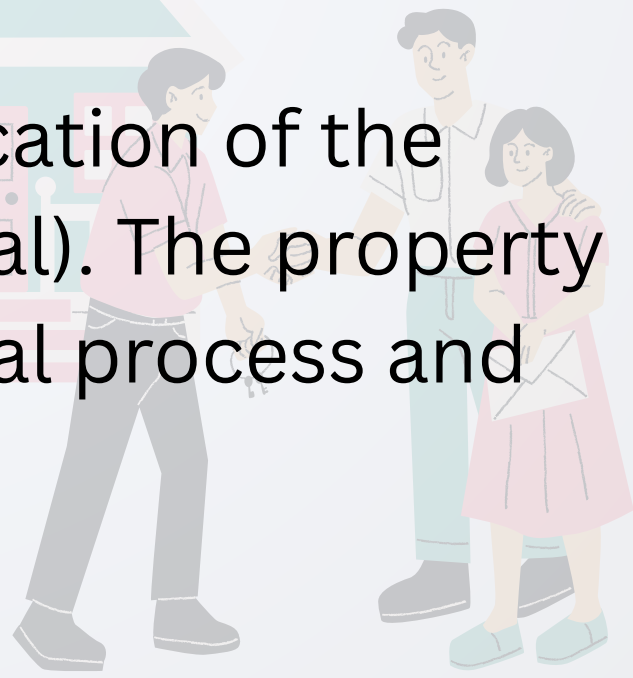
- **Loan_ID:** Each loan application has a unique identifier (e.g., LP001015). This helps in tracking and referencing individual applications.
- **Gender:** This column indicates the gender of the applicant. In our sample, all applicants are male.
- **Married:** Shows the marital status of the applicant. Here, we see both married (Yes) and unmarried (No) applicants.his column indicates the gender of the applicant. In our sample, all applicants are male.

Understanding the columns

- **Dependents:** Indicates the number of dependents the applicant has. This affects financial responsibilities and can influence loan approval decisions.
- **Education:** This field captures the educational background of the applicant. We see both graduates and non-graduates in the dataset.
- **Self_Employed:** This indicates whether the applicant is selfemployed (Yes/No). Employment status can impact the income stability and repayment capability.
- **ApplicantIncome:** This represents the monthly income of the applicant. For instance, the first applicant has an income of 5720.
- **CoapplicantIncome:** This shows the monthly income of a coapplicant, if applicable. A co-applicant can enhance the financial profile of the application.

Understanding the columns

- **LoanAmount:** This is the amount of loan requested by the applicant. For example, the first applicant is requesting a loan of 110.0.
- **Loan_Amount_Term:** Indicates the duration of the loan in months, with 360 months being common for home loans.
- **Credit_History:** A binary indicator that shows the creditworthiness of the applicant. A value of 1.0 indicates good credit, while NaN indicates missing data or unknown credit history.
- **Property_Area:** Describes the location of the property (Urban/Semiurban/Rural). The property area can impact the loan approval process and interest rates.



Description

```
df.info()
```

```
<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
```

Explanation of Dataset Information

The given loan dataset contains 367 records and 12 columns, representing applicant and loan-related details that can be used to predict loan approval decisions. There are no missing values, and the dataset includes both categorical (7) and numerical (5) variables. The categorical features describe demographic and employment details, while the numerical features represent income, loan, and credit information.

.

Description

Total Entries: The dataset contains **367** entries (rows), each representing a unique loan application.

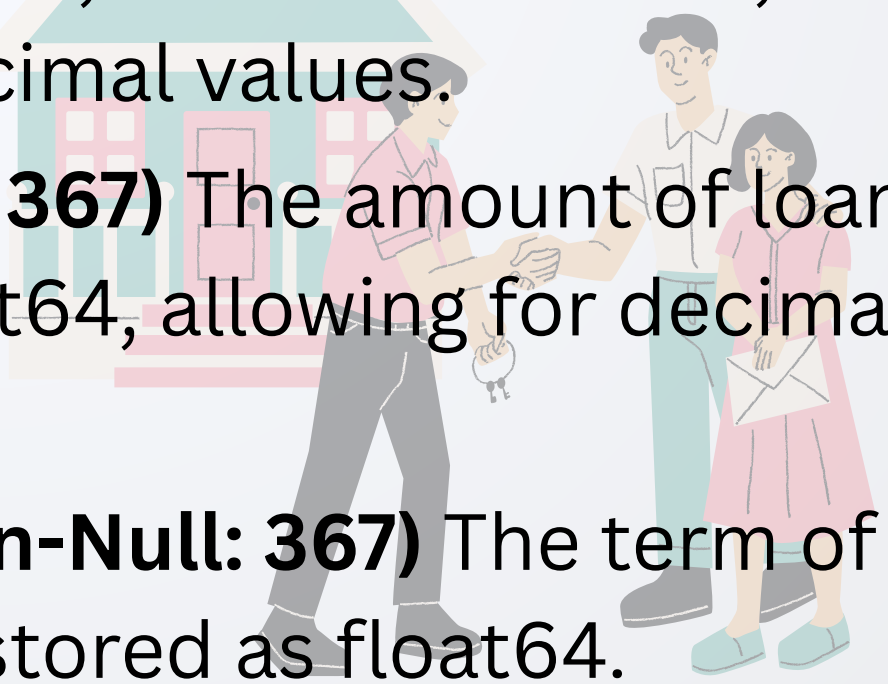
RangeIndex: The index ranges from 0 to 366, indicating the sequential nature of the entries.

The dataset consists of 12 columns, each representing a different feature of the loan applications:

- **Loan_ID: (Non-Null: 367)** This column contains unique identifiers for each loan application. It is of type object, indicating it contains string values.
- **Gender: (Non-Null: 367)** Indicates the gender of the applicants. It is also of type object.
- **Married: (Non-Null: 367)** Shows the marital status of the applicants. This is recorded as object.
- **Dependents: (Non-Null: 367)** Indicates the number of dependents (0, 1, 2, or 3+). This is also categorized as object, which may require conversion for numerical analysis.

Description

- **Education: (Non-Null: 367)** Details the educational qualifications of the applicants, recorded as object.
- **Self_Employed: (Non-Null: 367)** Shows whether the applicant is self-employed. This is stored as object.
- **ApplicantIncome: (Non-Null: 367)** Represents the monthly income of the applicant, recorded as int64, indicating it contains integer values.
- **CoapplicantIncome: (Non-Null: 367)** The monthly income of the co-applicant, stored as float64, indicating it can have decimal values.
- **LoanAmount: (Non-Null: 367)** The amount of loan requested, stored as float64, allowing for decimal values.
- **Loan_Amount_Term: (Non-Null: 367)** The term of the loan in months, also stored as float64.



Description

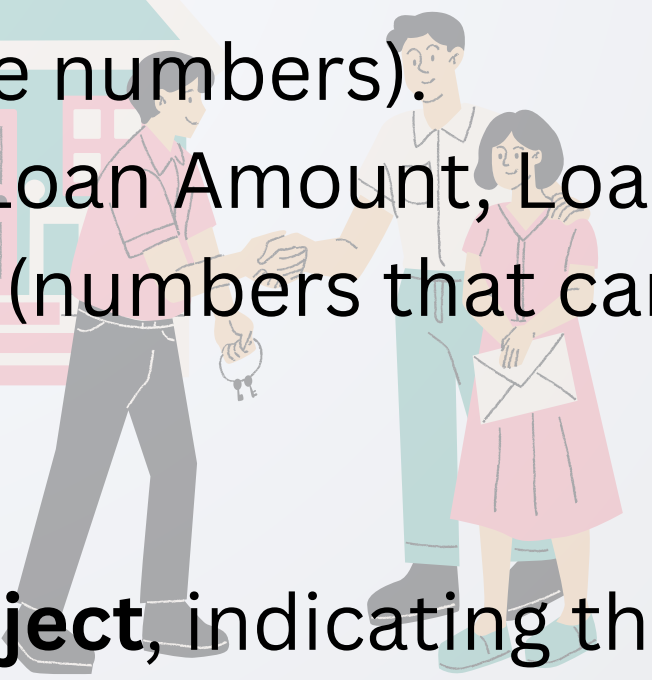
- **Credit_History (Non-Null: 366):** A crucial indicator of the applicant's credit history. There is one missing value here (NaN), indicating that one applicant's credit history is unknown.
- **Gender: (Non-Null: 367)** Indicates the gender of the applicants. It is also of type object.
- **Property_Area (Non-Null: 367):** Describes the area classification of the property (Urban/Semiurban/Rural), recorded as object.

Numeric Types:

int64 for Applicant Income (whole numbers).
float64 for Coapplicant Income, Loan Amount, Loan Amount Term, and Credit History (numbers that can include decimals).

Object Types:

Most columns are classified as **object**, indicating they contain categorical data (like strings for gender, marital status, etc.).



Data Cleaning & Pre-Processing

- **Shape of the Loan Application Dataset**

```
df.shape
```

```
(367, 12)
```

Rows (367): The dataset contains 367 loan application records, indicating the total number of applications included in this dataset. Each row represents a unique loan application.

Columns (12): There are 12 features (or columns) that provide various attributes about each loan application. These columns capture demographic and financial information relevant to the loan approval process.

- **Checking duplicates values if exists**

```
df.duplicated().sum()
```

```
np.int64(0)
```

Data Cleaning & Pre-Processing

- **Checking for missing Values**

By values

```
df.isnull().sum()
```

	0
Loan_ID	0
Gender	11
Married	0
Dependents	10
Education	0
Self_Employed	23
ApplicantIncome	0
CoapplicantIncome	0
LoanAmount	5
Loan_Amount_Term	6
Credit_History	29
Property_Area	0

By Percentage

```
df.isnull().sum()/df.shape[0]*100
```

	0
Loan_ID	0.000000
Gender	2.997275
Married	0.000000
Dependents	2.724796
Education	0.000000
Self_Employed	6.267030
ApplicantIncome	0.000000
CoapplicantIncome	0.000000
LoanAmount	1.362398
Loan_Amount_Term	1.634877
Credit_History	7.901907
Property_Area	0.000000

Data Cleaning & Pre-Processing

FILLING MISSING VALUES ACCORDING TO TYPES:

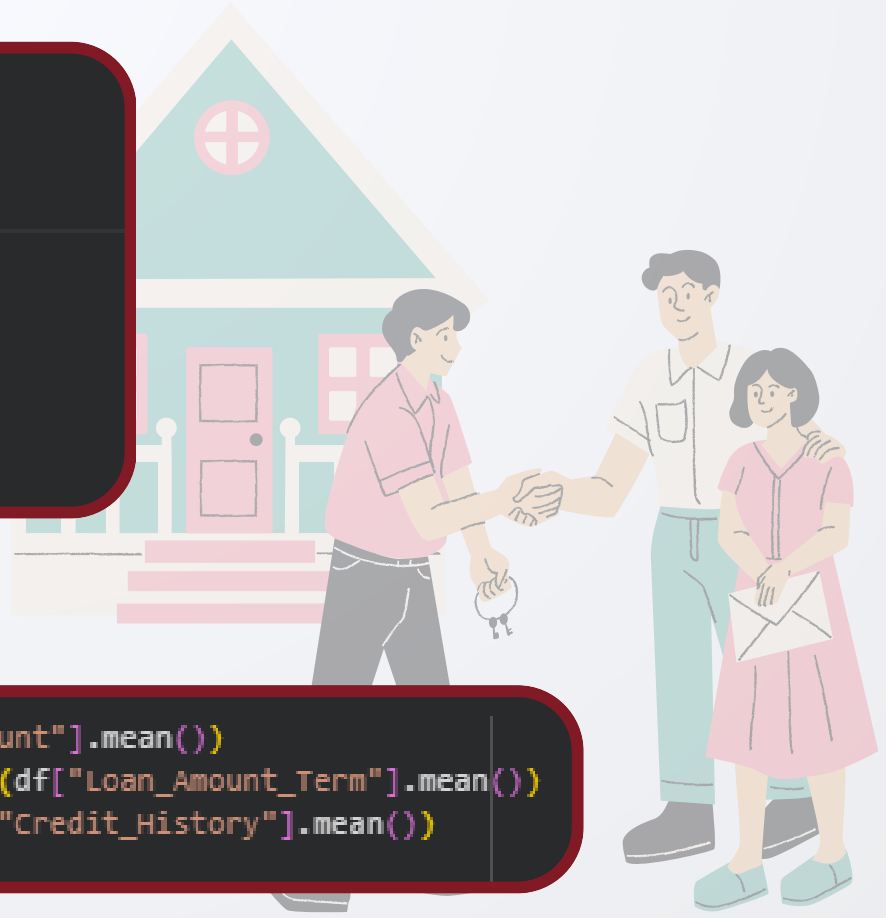
- **categorical columns:**

```
cat_columns = ['Gender','Dependents','Self_Employed']

def fill_missing_with_mode(df, cat_cols):
    for col in cat_cols:
        mode_value = df[col].mode()[0]
        df[col].fillna(mode_value, inplace=True)
    return df

df = fill_missing_with_mode(df, cat_columns)
```

No missing values left

```
print(df[cat_columns].isnull().sum())
```

```
Gender           0
Dependents       0
Self_Employed    0
dtype: int64
```



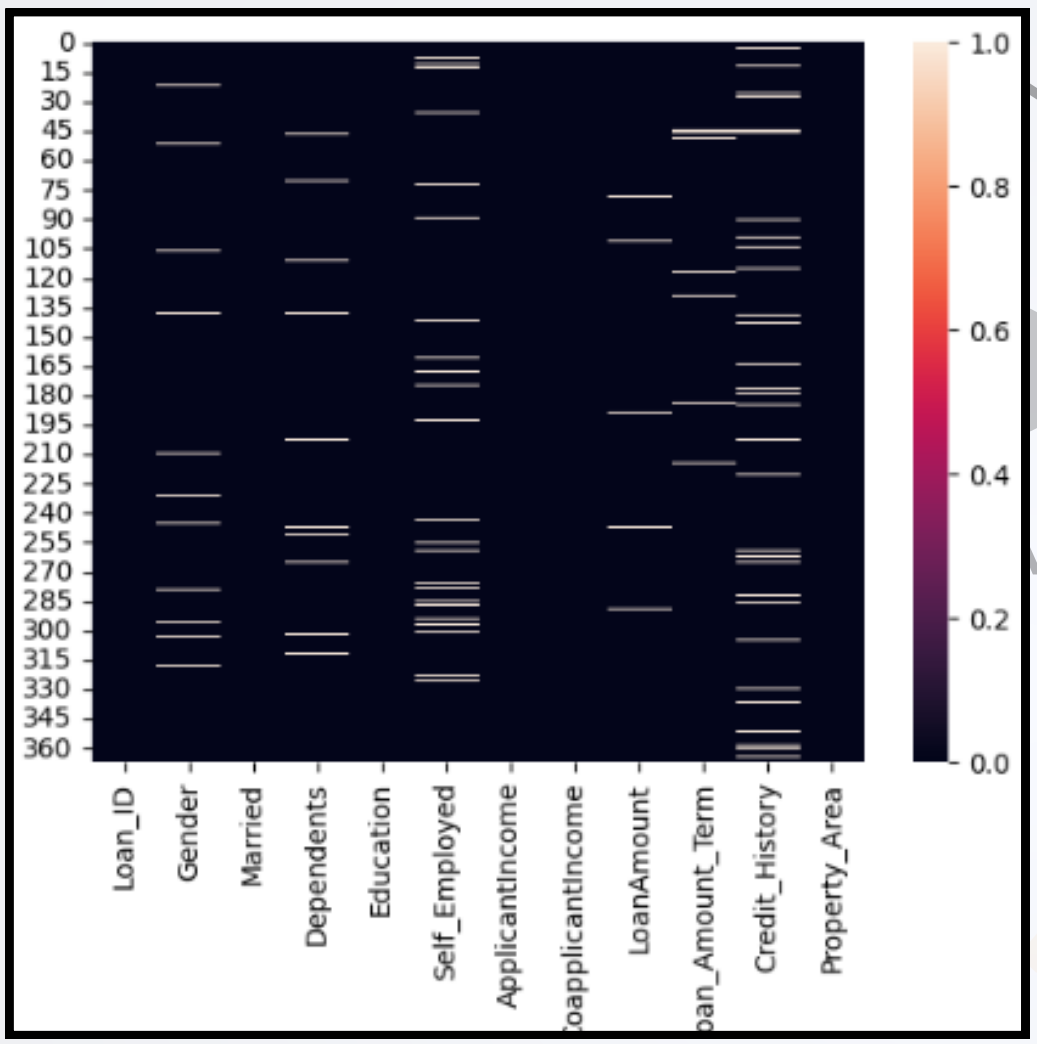
- **Numerical columns:**

```
df["LoanAmount"]=df["LoanAmount"].fillna(df["LoanAmount"].mean())
df["Loan_Amount_Term"]=df["Loan_Amount_Term"].fillna(df["Loan_Amount_Term"].mean())
df["Credit_History"]=df["Credit_History"].fillna(df["Credit_History"].mean())
```

Data Cleaning & Pre-Processing

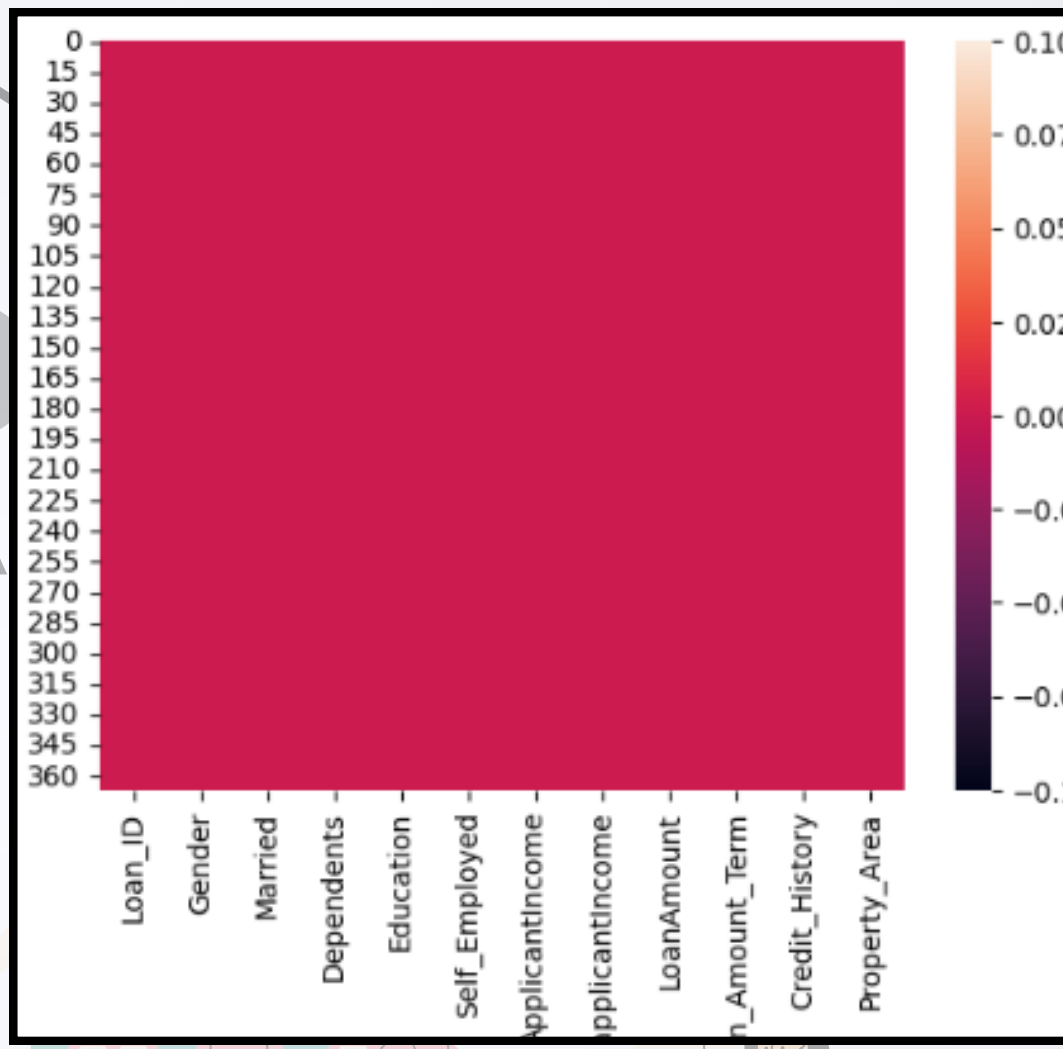
Before cleaning

```
sns.heatmap(df.isnull())
```



After cleaning

```
sns.heatmap(df.isnull())
```



- **Summarize basic statistics for the numeric columns.**

```
df.describe()
```

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	367.000000	367.000000	367.000000	367.000000	367.000000
mean	4805.599455	1569.577657	136.132597	342.537396	0.825444
std	4910.685399	2334.232099	60.946040	64.620366	0.364778
min	0.000000	0.000000	28.000000	6.000000	0.000000
25%	2864.000000	0.000000	101.000000	360.000000	1.000000
50%	3786.000000	1025.000000	126.000000	360.000000	1.000000
75%	5060.000000	2430.500000	157.500000	360.000000	1.000000
max	72529.000000	24000.000000	550.000000	480.000000	1.000000

Data Visualization

UNIVARIENT ANALYSIS

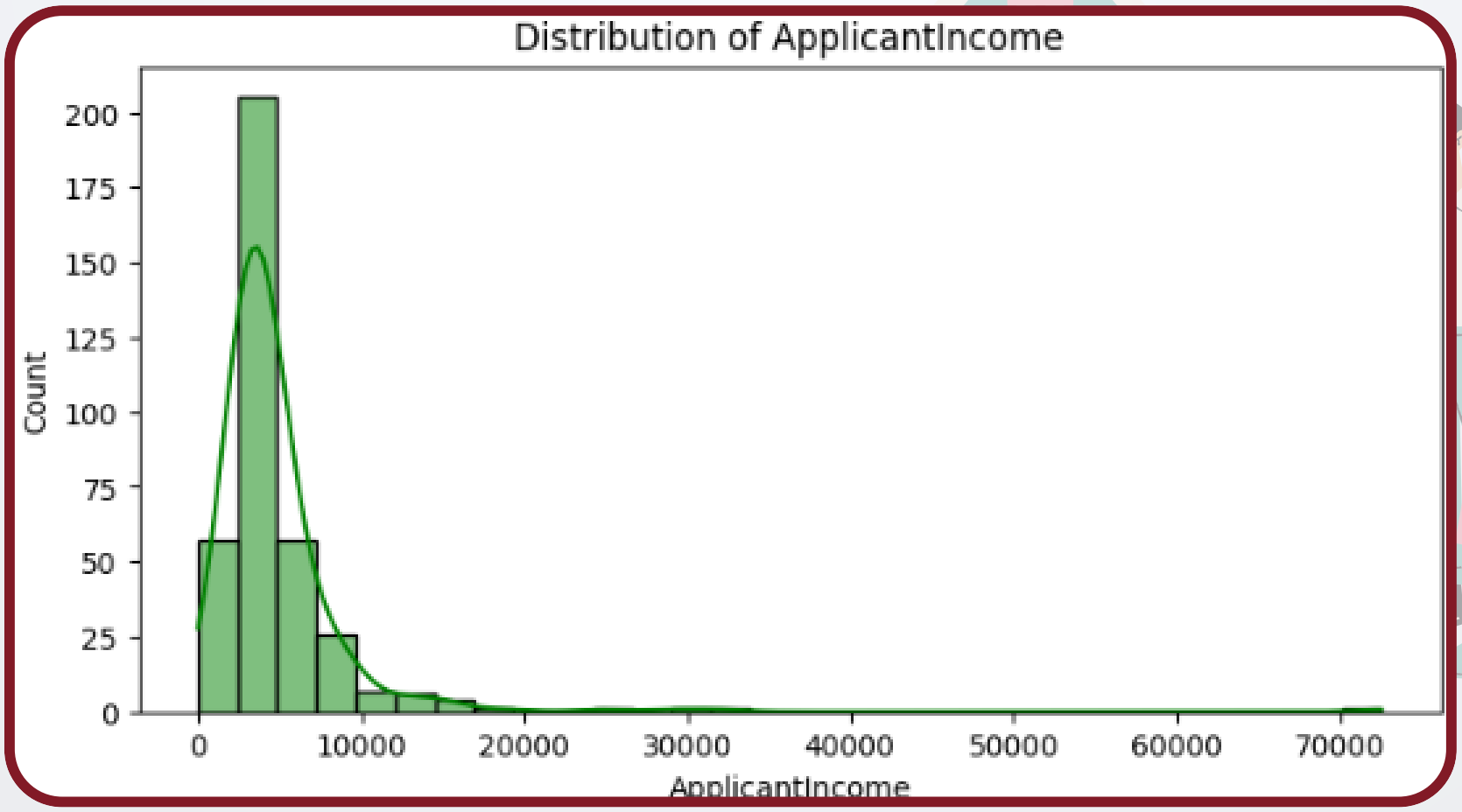
- **Histograms:** Plot the frequency distribution of key numeric variables.

```
num_cols = df.select_dtypes(include=['int64', 'float64']).drop(columns=['Credit_History']).columns
num_cols
```

```
Index(['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',
       'Loan_Amount_Term'],
      dtype='object')
```

```
for col in num_cols:
    plt.figure(figsize=(8, 4))
    sns.histplot(df[col], kde=True, bins=30, color='green')
    plt.title(f'Distribution of {col}')
    plt.show()
```

Applicant Income

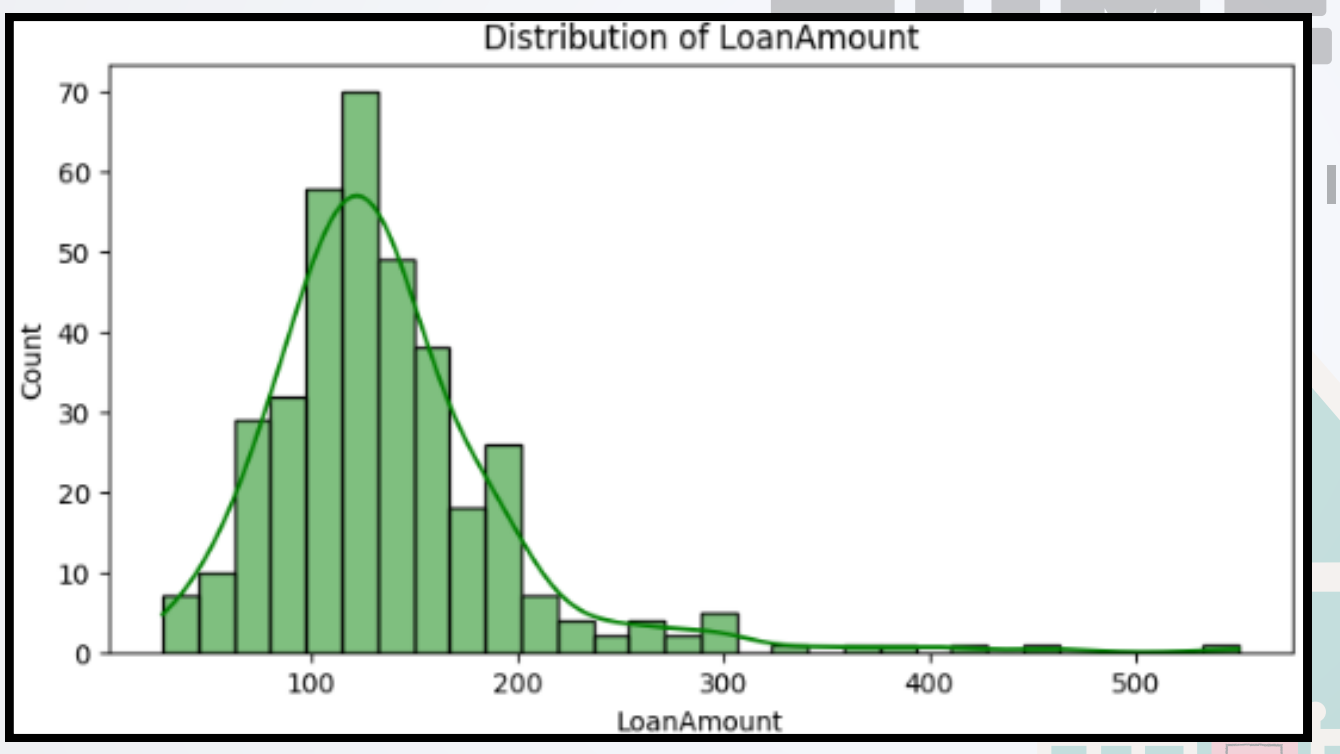


Data Visualization

Insights

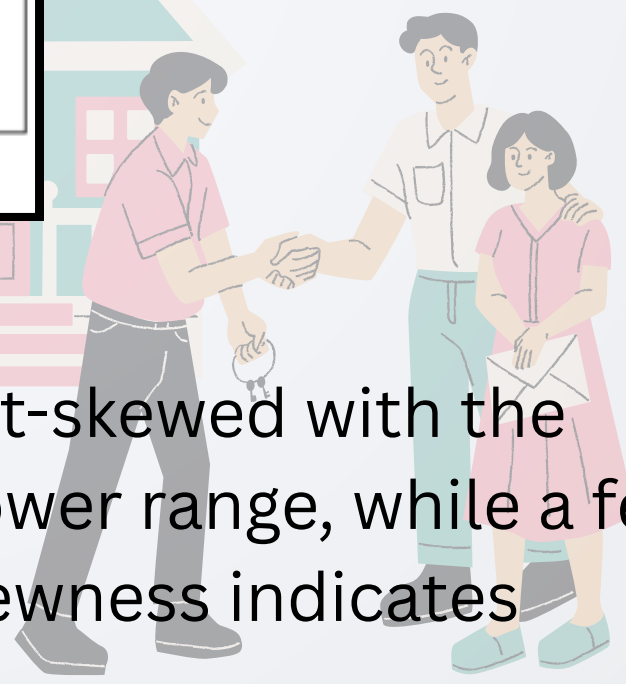
The income of applicants is right-skewed, indicating most applicants earn lower to moderate incomes, but a small group earns significantly higher amounts. This skewness suggests income varies widely among applicants, and there may be outliers with high incomes.

Loan Amount



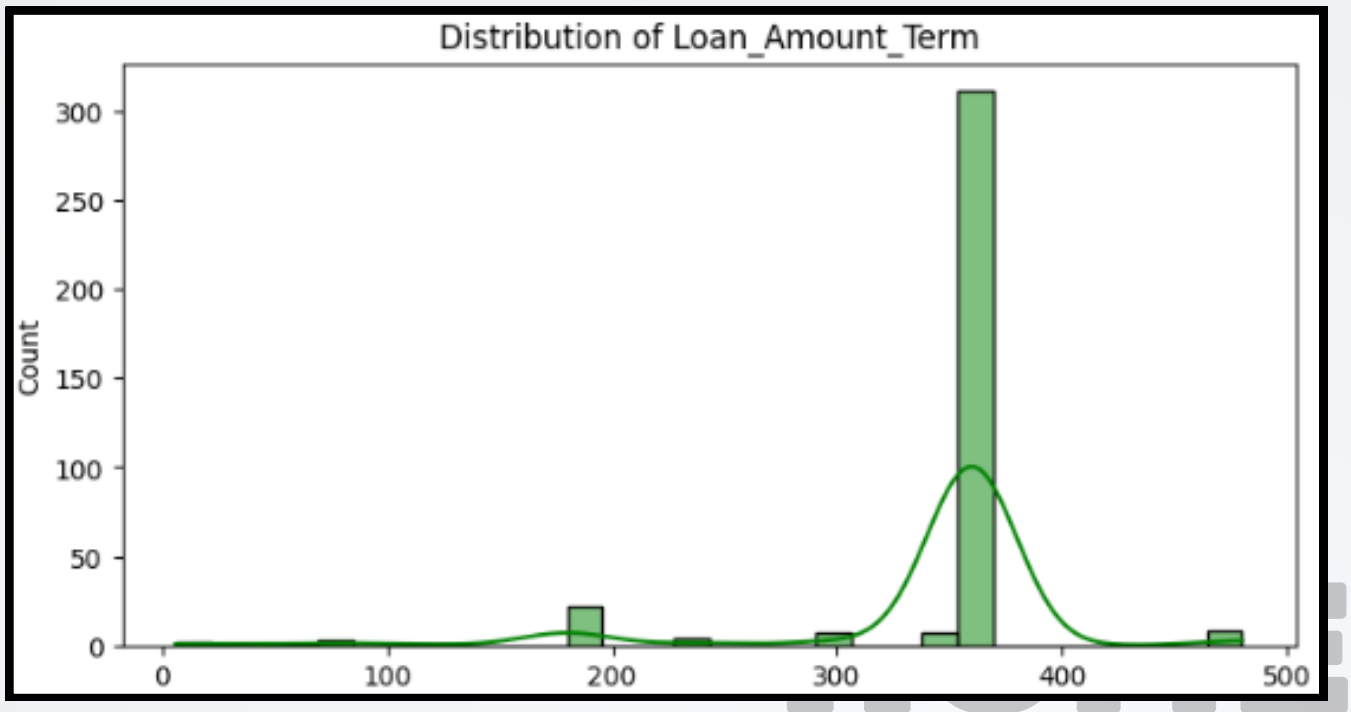
Insights

The loan amount applied for is also right-skewed with the majority of loans concentrated in the lower range, while a few applicants request larger loans. This skewness indicates varying loan needs and risk profiles.



Data Visualization

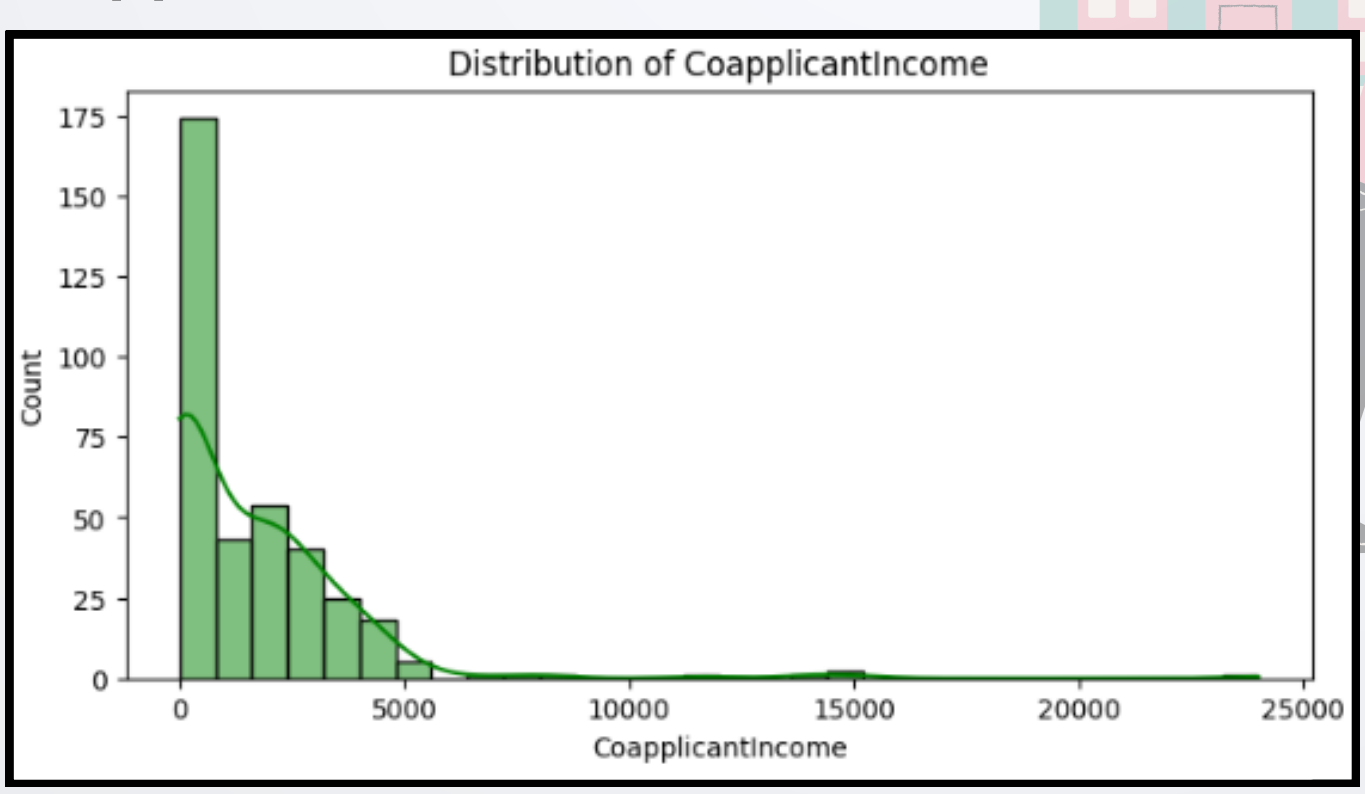
Loan Amount Term



Insights

The loan terms mostly cluster around typical durations such as 360 months (30 years), with fewer loans with shorter or longer terms. This reflects standard repayment plans in lending practices.

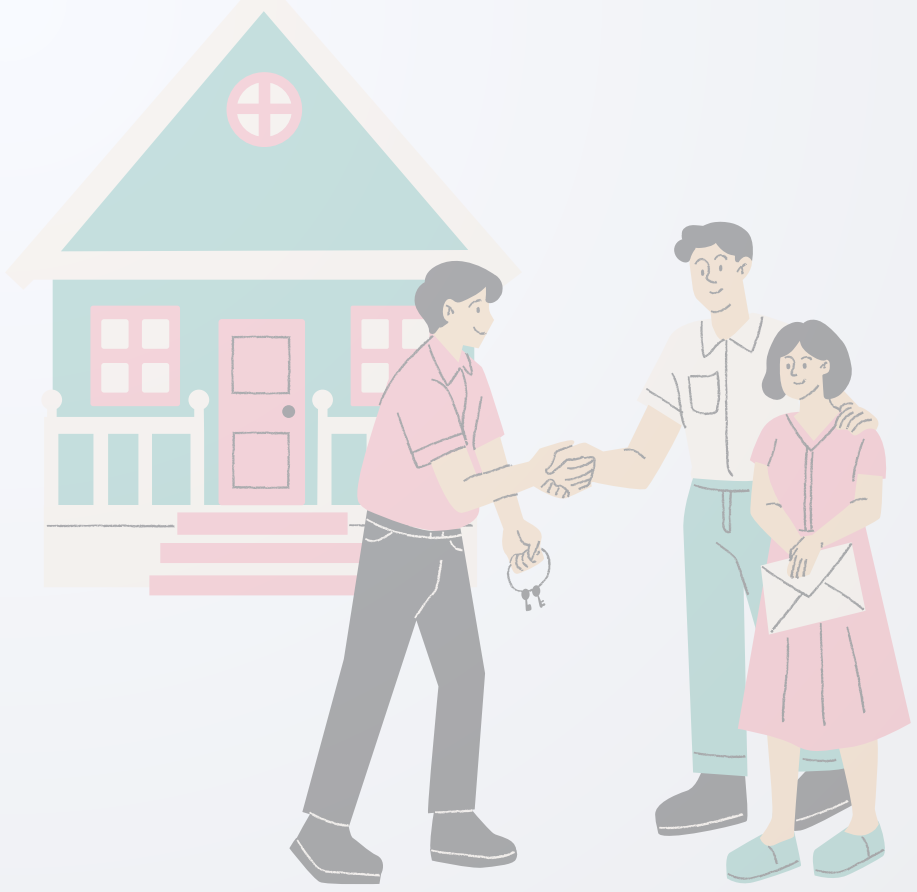
Coapplicant Income



Data Visualization

Insights

The coapplicant income is heavily concentrated near zero, showing most applicants do not have a coapplicant or the coapplicant has little to no income. The distribution may show a sharp drop after zero, with fewer cases of high coapplicant incomes.

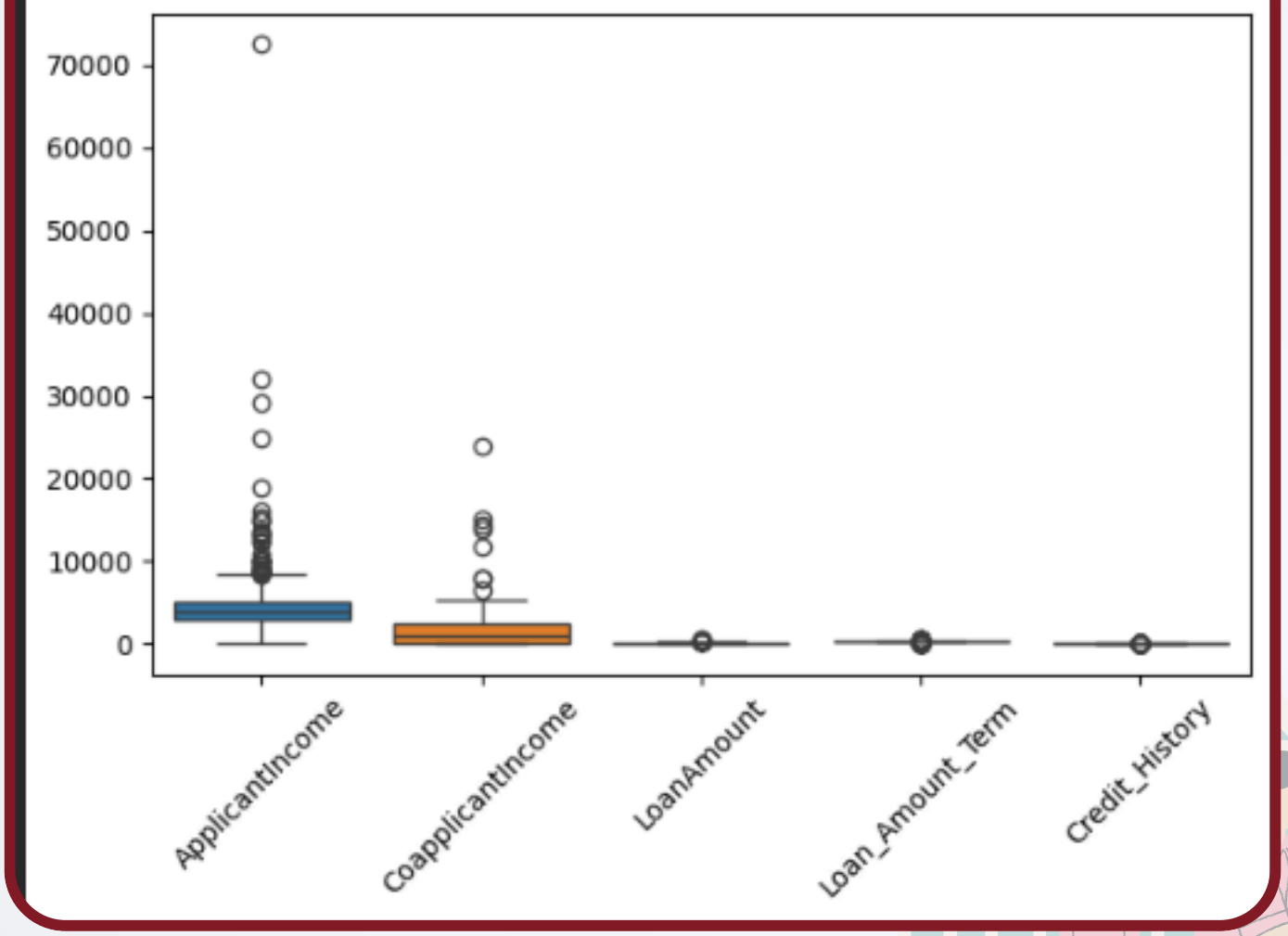


Data Visualization

- **Box Plots:** Identify potential outliers and visualize the spread of data.

Overview of potential outliers

```
sns.boxplot(data=df)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Checking Outlier according to columns one by one and see further results after caping them.

Caping: Capping outliers (also known as Winsorization) is a data-cleaning technique used to handle extreme values without removing any data points altogether. Instead of deleting outliers, this method replaces extreme values with more reasonable limits—typically using the percentiles or the Interquartile Range (IQR) method

Data Visualization

ApplicantIncome

```
q1=df['ApplicantIncome'].quantile(0.25)
q3=df['ApplicantIncome'].quantile(0.75)

iqr=q3-q1

lower_bound= q1 -1.5 * iqr
upper_bound= q3 +1.5 * iqr
print(f"the IQR of ApplicantIncome is : {iqr}")
print(f"the lower bound of ApplicantIncome is : {lower_bound}")
print(f"the upper bound of ApplicantIncome is : {upper_bound}")

outliers=df[(df['ApplicantIncome'] <lower_bound) | (df['ApplicantIncome']<upper_bound)]
print(f"Number of outliers in ApplicantIncome : {len(outliers)}")
```

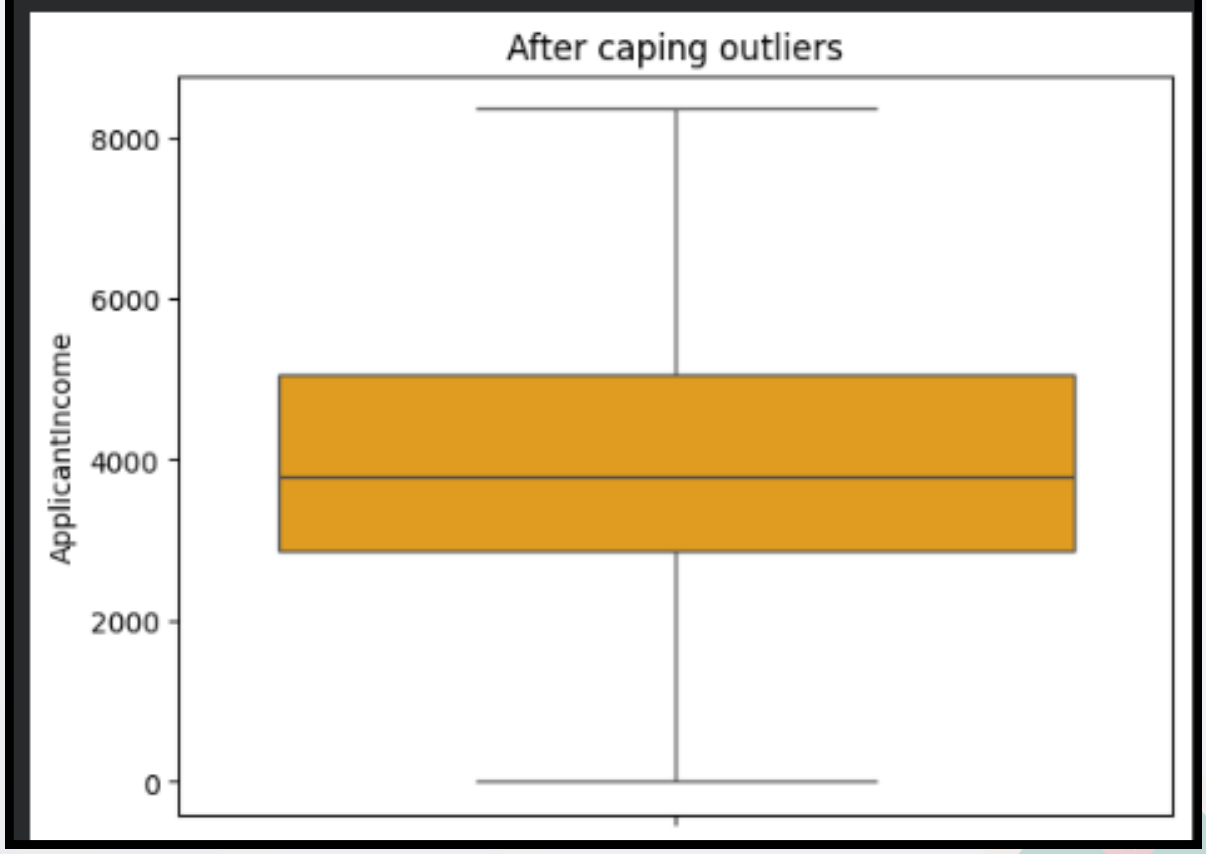
```
the IQR of ApplicantIncome is : 2196.0
the lower bound of ApplicantIncome is : -430.0
the upper bound of ApplicantIncome is : 8354.0
Number of outliers in ApplicantIncome : 335
```

Caping


```
df["ApplicantIncome"]=np.where(df["ApplicantIncome"]<=lower_bound,lower_bound,df["ApplicantIncome"])
df["ApplicantIncome"]=np.where(df["ApplicantIncome"]>=upper_bound,upper_bound,df["ApplicantIncome"])
```

Boxplot after caping

```
sns.boxplot(df["ApplicantIncome"],color="orange")
plt.title("After caping outliers")
plt.show()
```



Data Visualization

Insights

ApplicantIncome showed a wide variability with several extreme high-income outliers. After applying the IQR method, these outliers were capped at the calculated upper and lower bounds to minimize their influence. This step standardized income distribution and maintained the integrity of the data, ensuring more robust analysis in subsequent modeling.

CoapplicantIncome

```
q1=df['CoapplicantIncome'].quantile(0.25)
q3=df['CoapplicantIncome'].quantile(0.75)

iqr=q3-q1

lower_bound= q1 -1.5 * iqr
upper_bound= q3 +1.5 * iqr
print(f"the IQR of ApplicantIncome is : {iqr}")
print(f"the lower bound of ApplicantIncome is : {lower_bound}")
print(f"the upper bound of ApplicantIncome is : {upper_bound}")

outliers=df[(df['CoapplicantIncome'] <lower_bound) | (df['CoapplicantIncome']<upper_bound)]
print(f"Number of outliers in CoapplicantIncome : {len(outliers)}")
```

```
the IQR of ApplicantIncome is : 2430.5
the lower bound of ApplicantIncome is : -3645.75
the upper bound of ApplicantIncome is : 6076.25
Number of outliers in CoapplicantIncome : 359
```

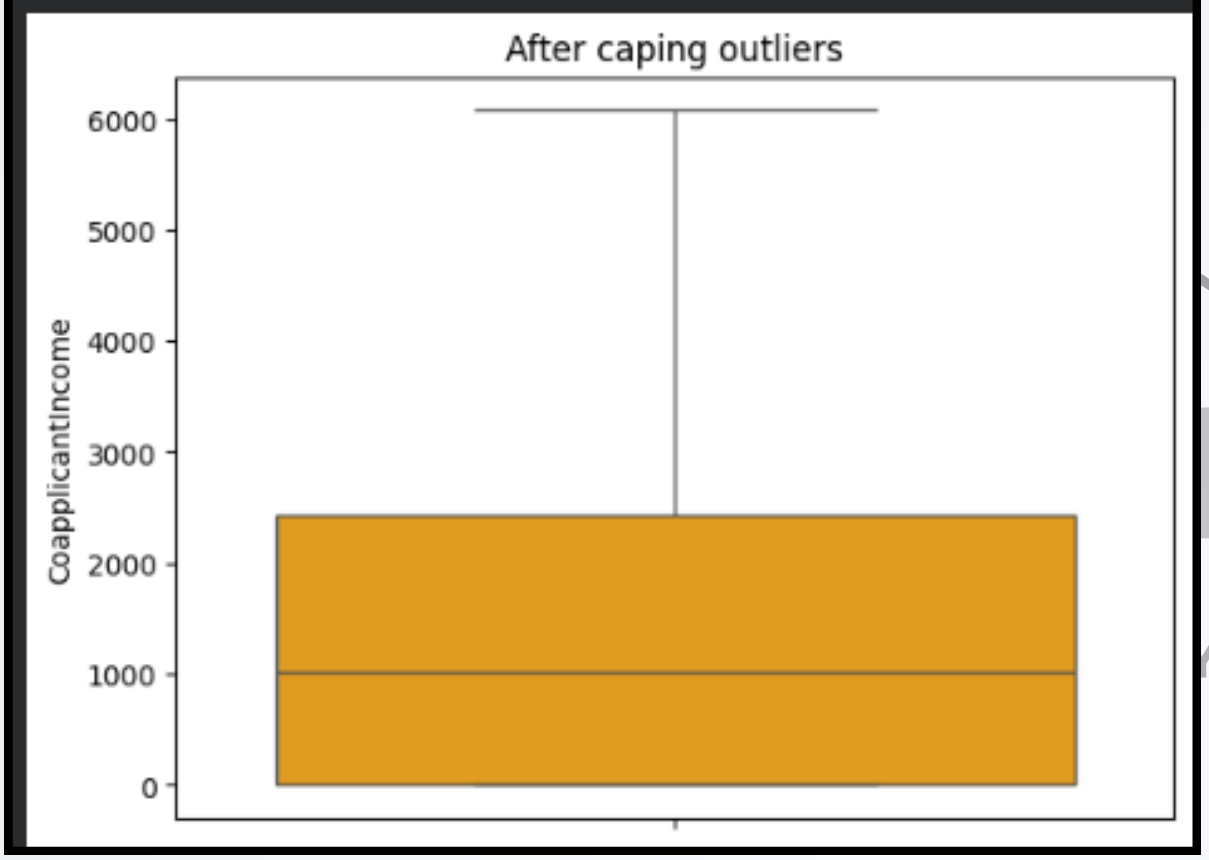
Caping

```
df["CoapplicantIncome"]=np.where(df["CoapplicantIncome"]<=lower_bound,lower_bound,df["CoapplicantIncome"])
df["CoapplicantIncome"]=np.where(df["CoapplicantIncome"]>=upper_bound,upper_bound,df["CoapplicantIncome"])
```

Data Visualization

Boxplot after caping

```
sns.boxplot(df["CoapplicantIncome"],color="orange")
plt.title("After caping outliers")
plt.show()
```



Insights

Capping the outliers in CoapplicantIncome helped in stabilizing the data distribution and prevented the extreme income values from skewing the analysis. This ensures that future statistical summaries and visualizations reflect a more accurate representation of the typical coapplicant income range.

Data Visualization

Loan_amount

```
q1=df['LoanAmount'].quantile(0.25)
q3=df['LoanAmount'].quantile(0.75)

iqr=q3-q1

lower_bound= q1 -1.5 * iqr
upper_bound= q3 +1.5 * iqr
print(f"the IQR of ApplicantIncome is : {iqr}")
print(f"the lower bound of ApplicantIncome is : {lower_bound}")
print(f"the upper bound of ApplicantIncome is : {upper_bound}")

outliers=df[(df['LoanAmount'] <lower_bound) | (df['LoanAmount']<upper_bound)]
print(f"Number of outliers in LoanAmount : {len(outliers)}")
```

```
the IQR of ApplicantIncome is : 56.5
the lower bound of ApplicantIncome is : 16.25
the upper bound of ApplicantIncome is : 242.25
Number of outliers in LoanAmount : 349
```

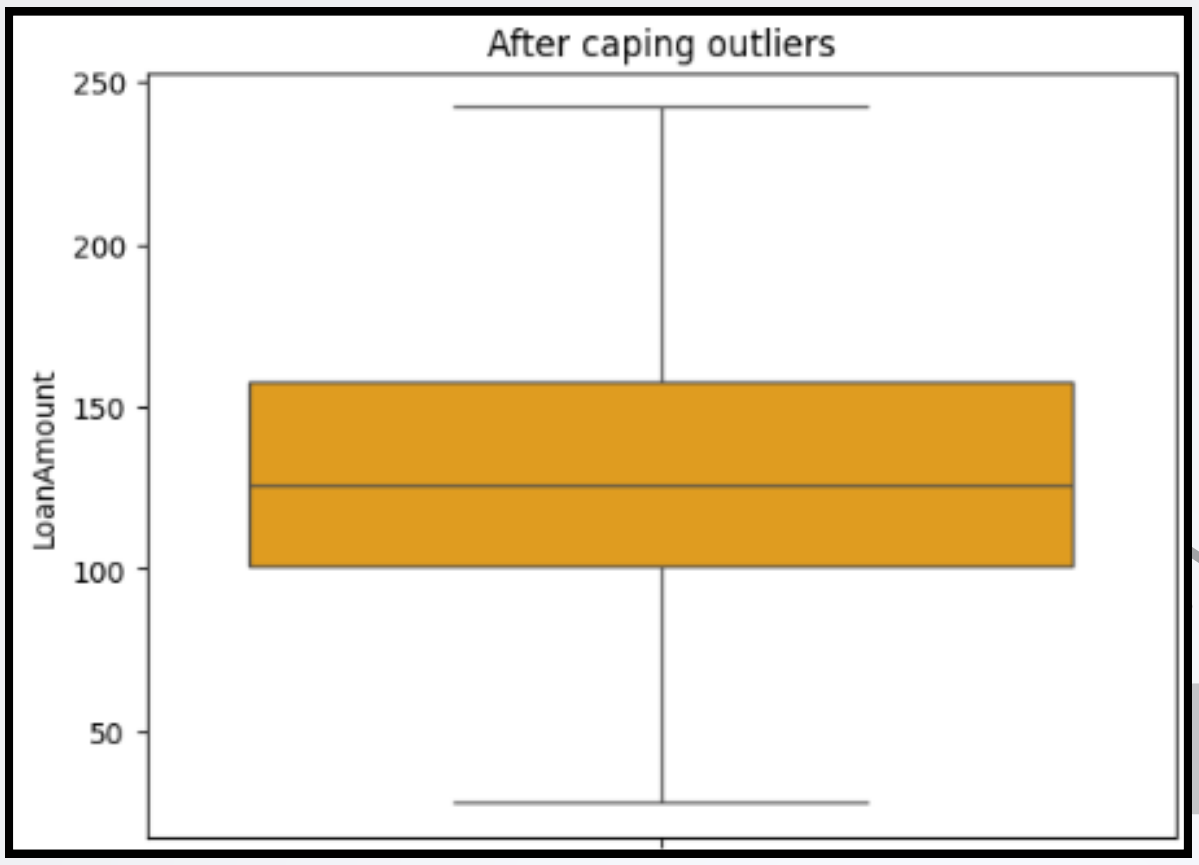
caping

```
df["LoanAmount"]=np.where(df["LoanAmount"]<=lower_bound,lower_bound,df["LoanAmount"])
df["LoanAmount"]=np.where(df["LoanAmount"]>=upper_bound,upper_bound,df["LoanAmount"])
```

Boxplot after caping

```
sns.boxplot(df["LoanAmount"],color="orange")
plt.title("After caping outliers")
plt.show()
```

Data Visualization



LOAN ANALYSIS

Insights

Capping the extreme values in LoanAmount helped reduce the influence of unusually high or low loan amounts on the analysis. This ensures that the average and median loan values better represent the typical loan size in the dataset, leading to more reliable insights and visualizations.

Data Visualization

Loan_Amount_term

```
q1=df['Loan_Amount_Term'].quantile(0.25)
q3=df['Loan_Amount_Term'].quantile(0.75)

iqr=q3-q1

lower_bound= q1 -1.5 * iqr
upper_bound= q3 +1.5 * iqr
print(f"the IQR of ApplicantIncome is : {iqr}")
print(f"the lower bound of ApplicantIncome is : {lower_bound}")
print(f"the upper bound of ApplicantIncome is : {upper_bound}")

outliers=df[(df['Loan_Amount_Term'] <lower_bound) | (df['Loan_Amount_Term']<upper_bound)]
print(f"Number of outliers in Loan_Amount_Term : {len(outliers)}")
```

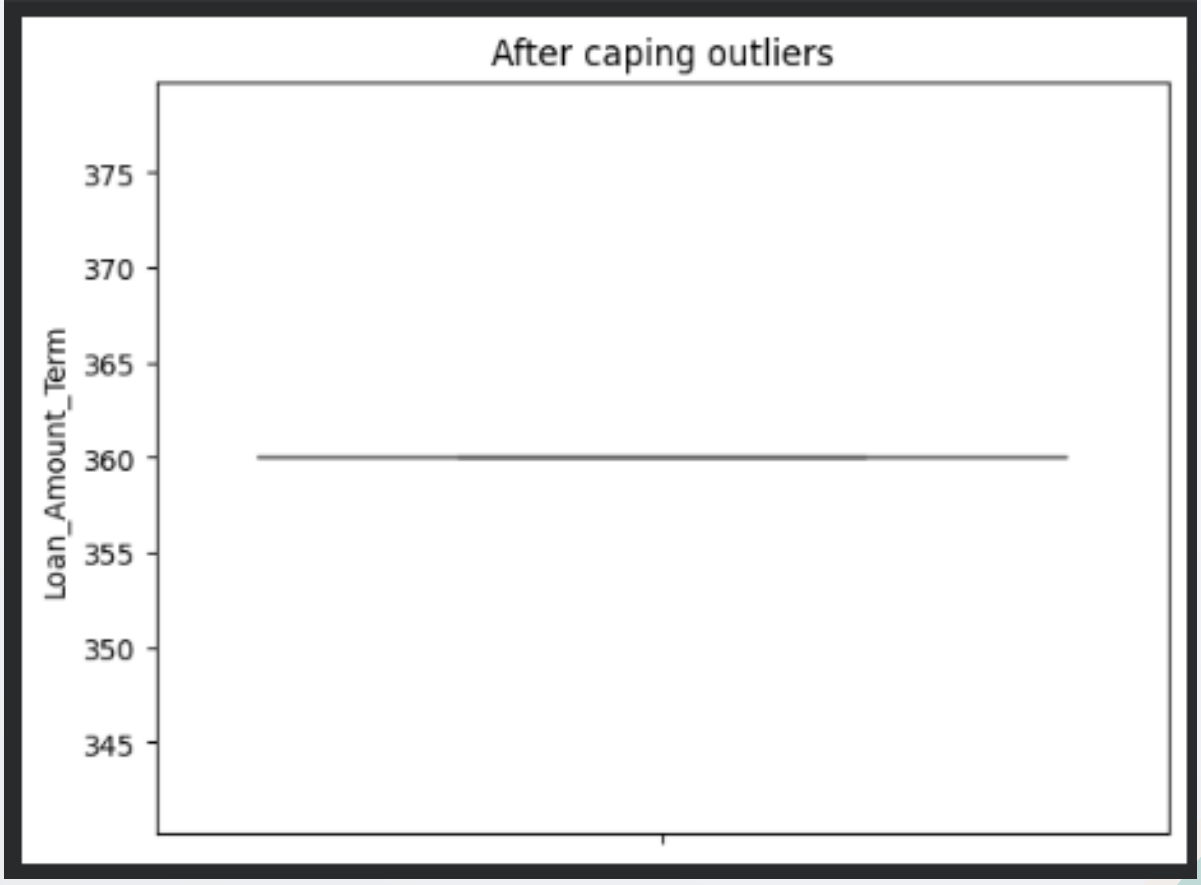
```
the IQR of ApplicantIncome is : 0.0
the lower bound of ApplicantIncome is : 360.0
the upper bound of ApplicantIncome is : 360.0
Number of outliers in Loan_Amount_Term : 48
```

caping

```
Loan_Amount_Term"]=np.where(df["Loan_Amount_Term"]<=lower_bound,lower_bound,df["Loan_Amount_Ter
Loan_Amount_Term"]=np.where(df["Loan_Amount_Term"]>=upper_bound,upper_bound,df["Loan_Amount_Ter
```


Boxplot after caping

```
sns.boxplot(df["Loan_Amount_Term"],color="orange")
plt.title("After caping outliers")
plt.show()
```



Data Visualization

Insights

After treating outliers in Loan_Amount_Term, the term durations became more evenly distributed. This helps ensure that further analysis (like comparing loan term vs loan amount or approval rate) reflects a more realistic view of loan durations offered to applicants.

- **Bar Charts:** Visualize the frequency distribution of categorical variables.

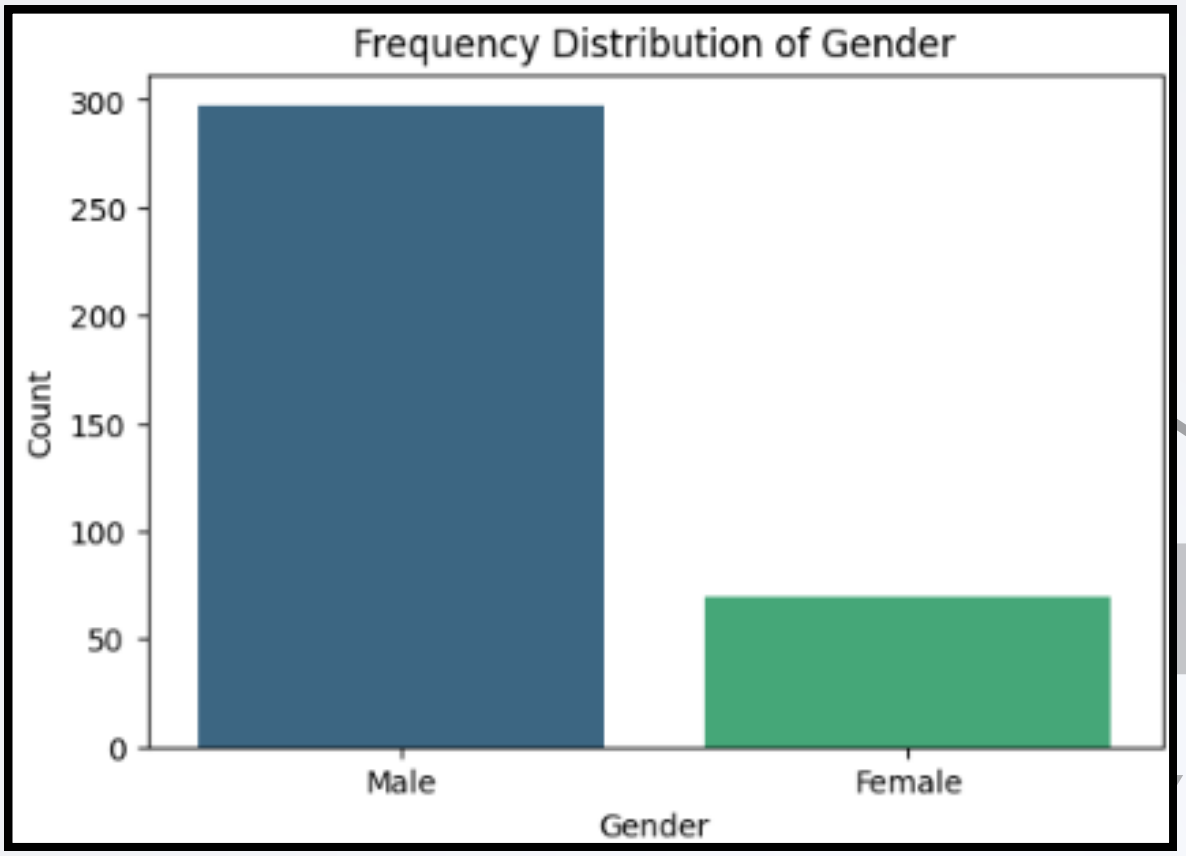
```
temp_df = df.select_dtypes(include=["object"]).columns.drop("Loan_ID")
temp_df
```

```
Index(['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed',
       'Property_Area'],
      dtype='object')
```

```
for col in temp_df:
    plt.figure(figsize=(6, 4))
    sns.countplot(x=df[col], palette='viridis')
    plt.title(f'Frequency Distribution of {col}')
    plt.xlabel(col)
    plt.ylabel('Count')
    plt.show()
```

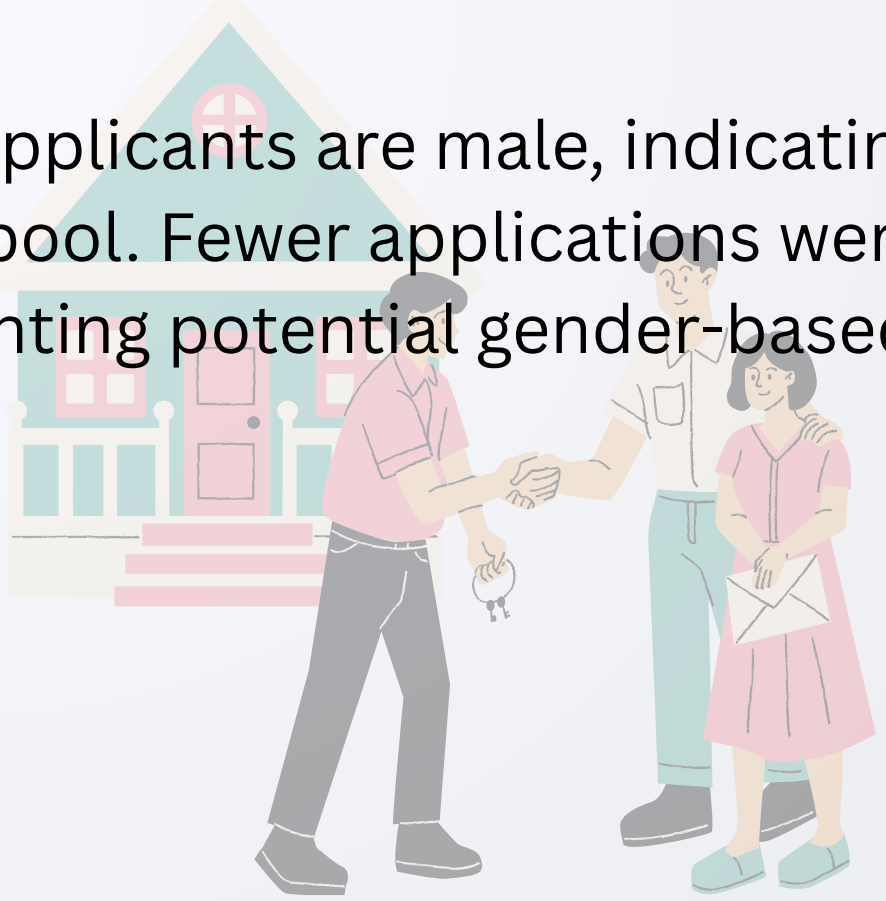
Data Visualization

Gender



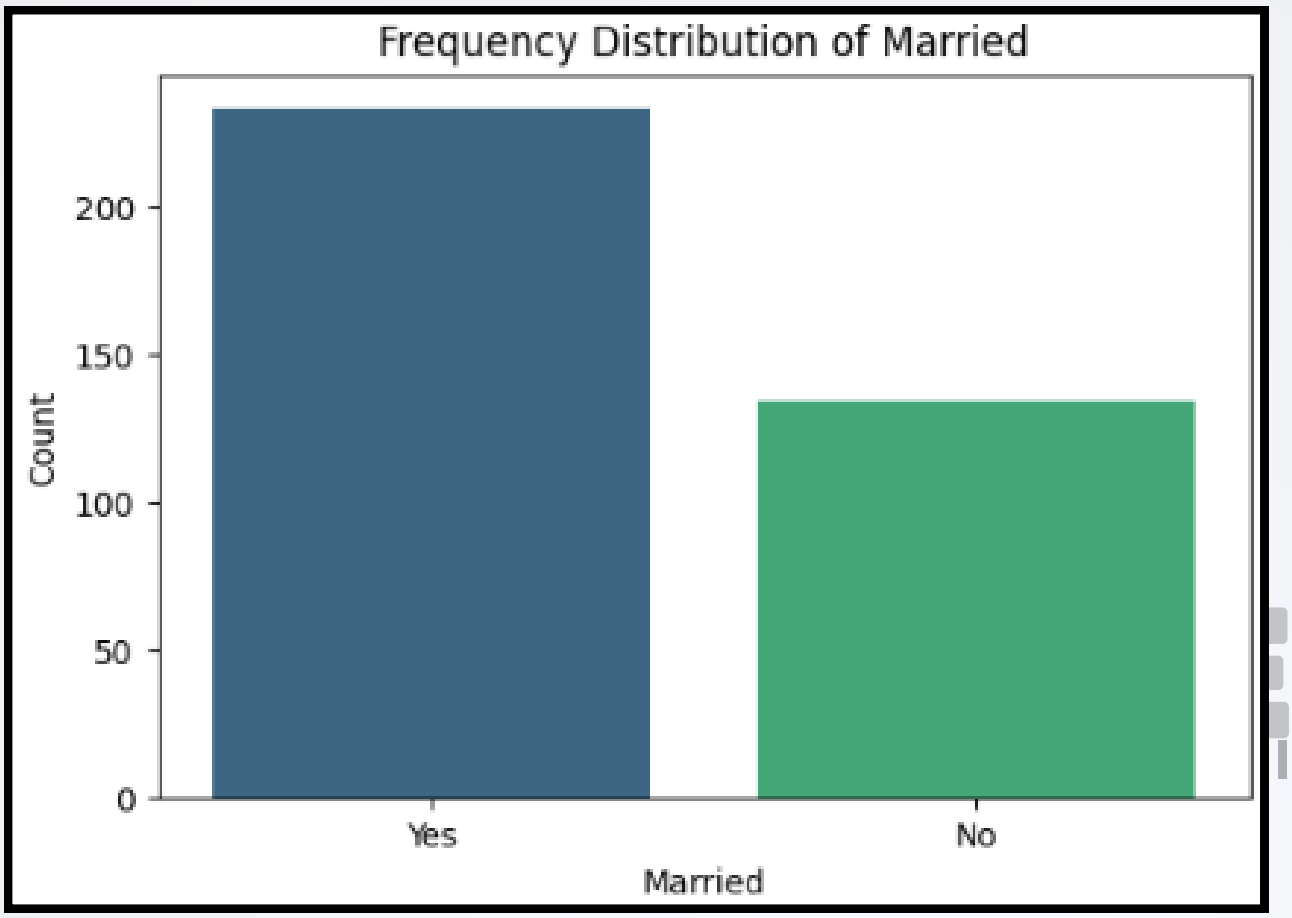
Insights

Gender: The majority of loan applicants are male, indicating a gender skew in the applicant pool. Fewer applications were submitted by females, highlighting potential gender-based trends in seeking loans.



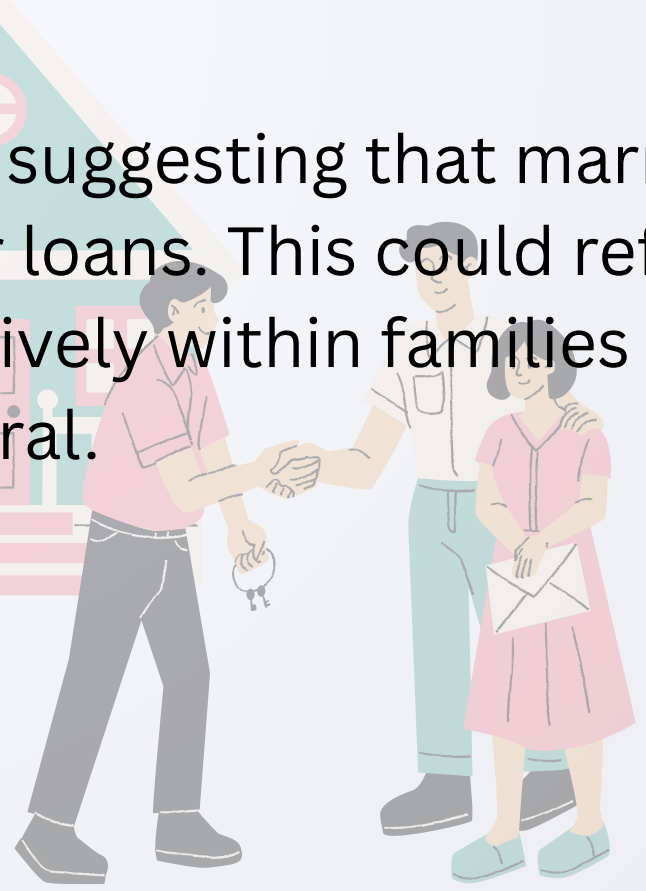
Married

Data Visualization



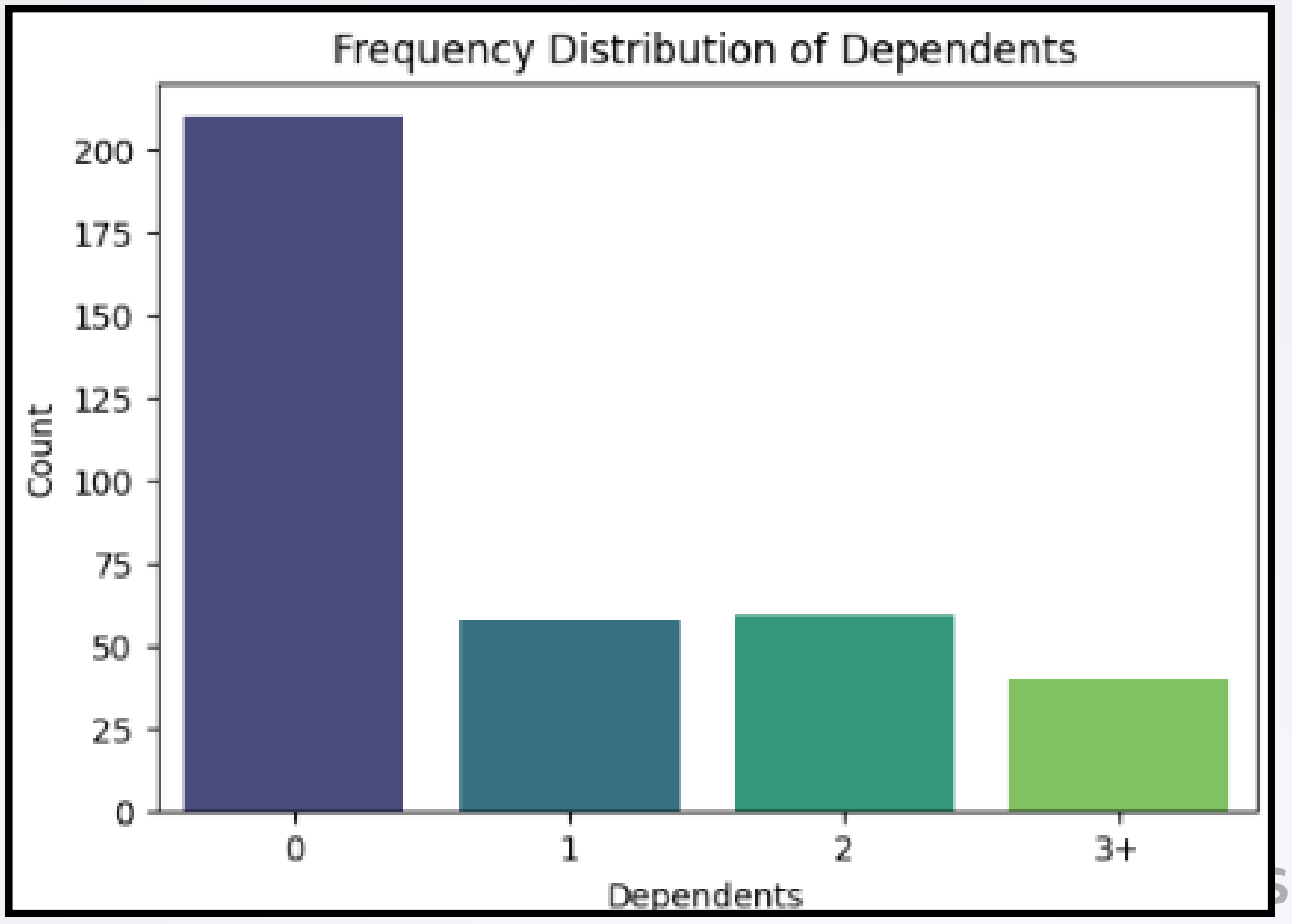
Insights

Married: Most applicants are married, suggesting that married individuals are more likely to apply for loans. This could reflect financial decisions being made collectively within families or the impact of joint income and collateral.



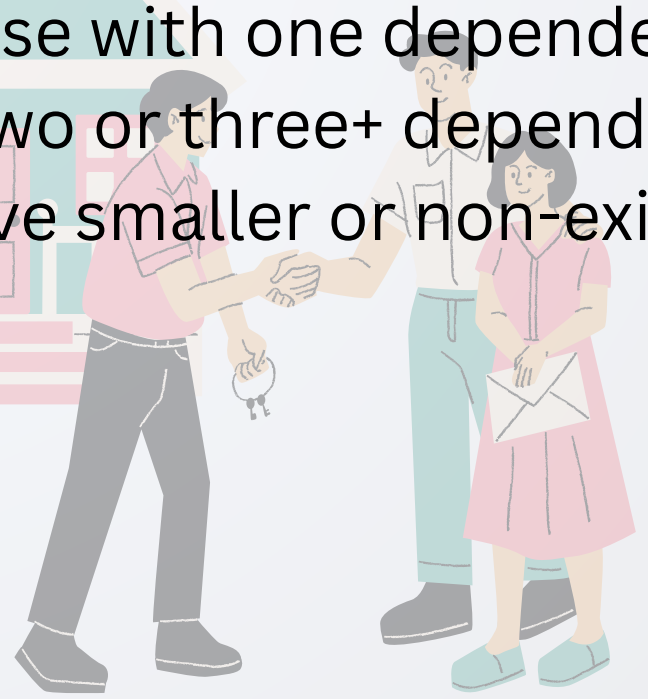
Dependents

Data Visualization



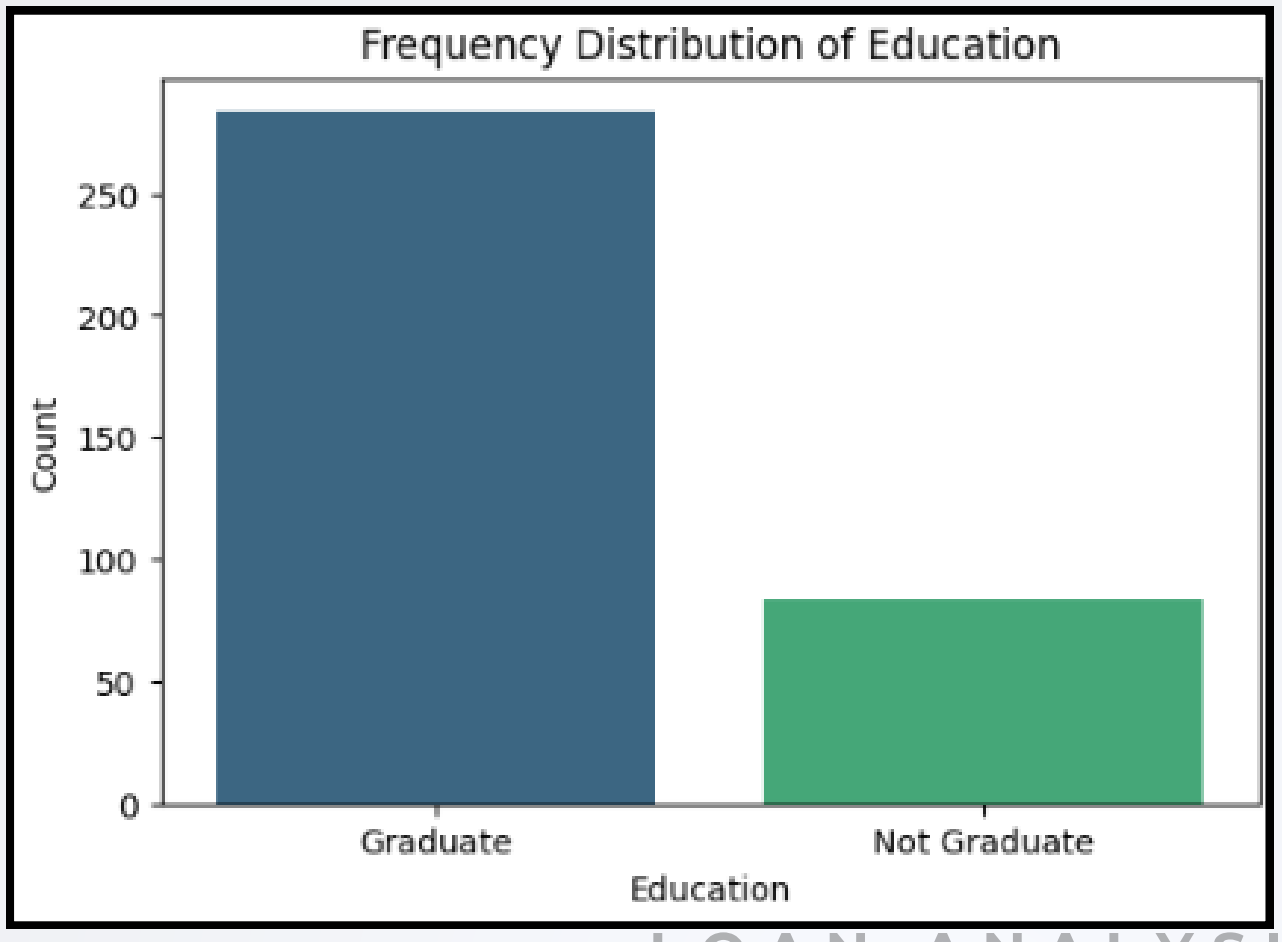
Insights

Dependents: The highest frequency is observed for applicants with zero dependents, followed by those with one dependent. There are fewer applicants reporting two or three+ dependents, indicating loan applicants typically have smaller or non-existent dependent families.



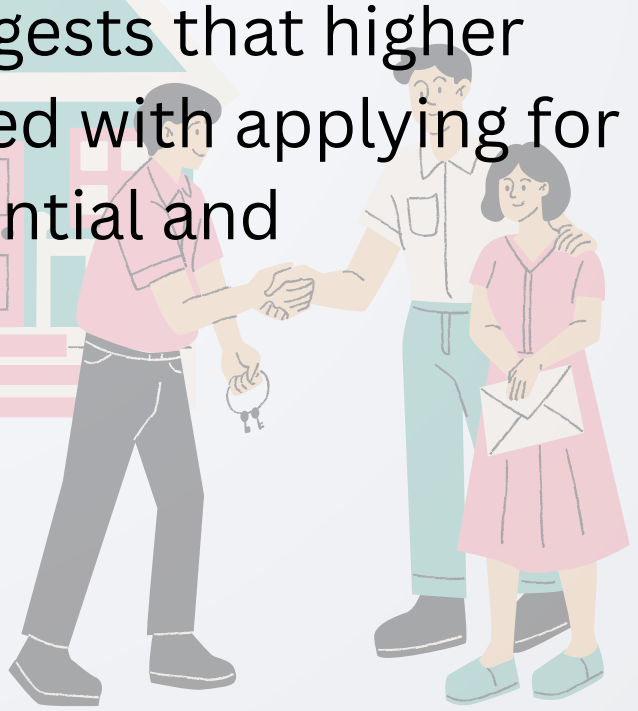
Education

Data Visualization



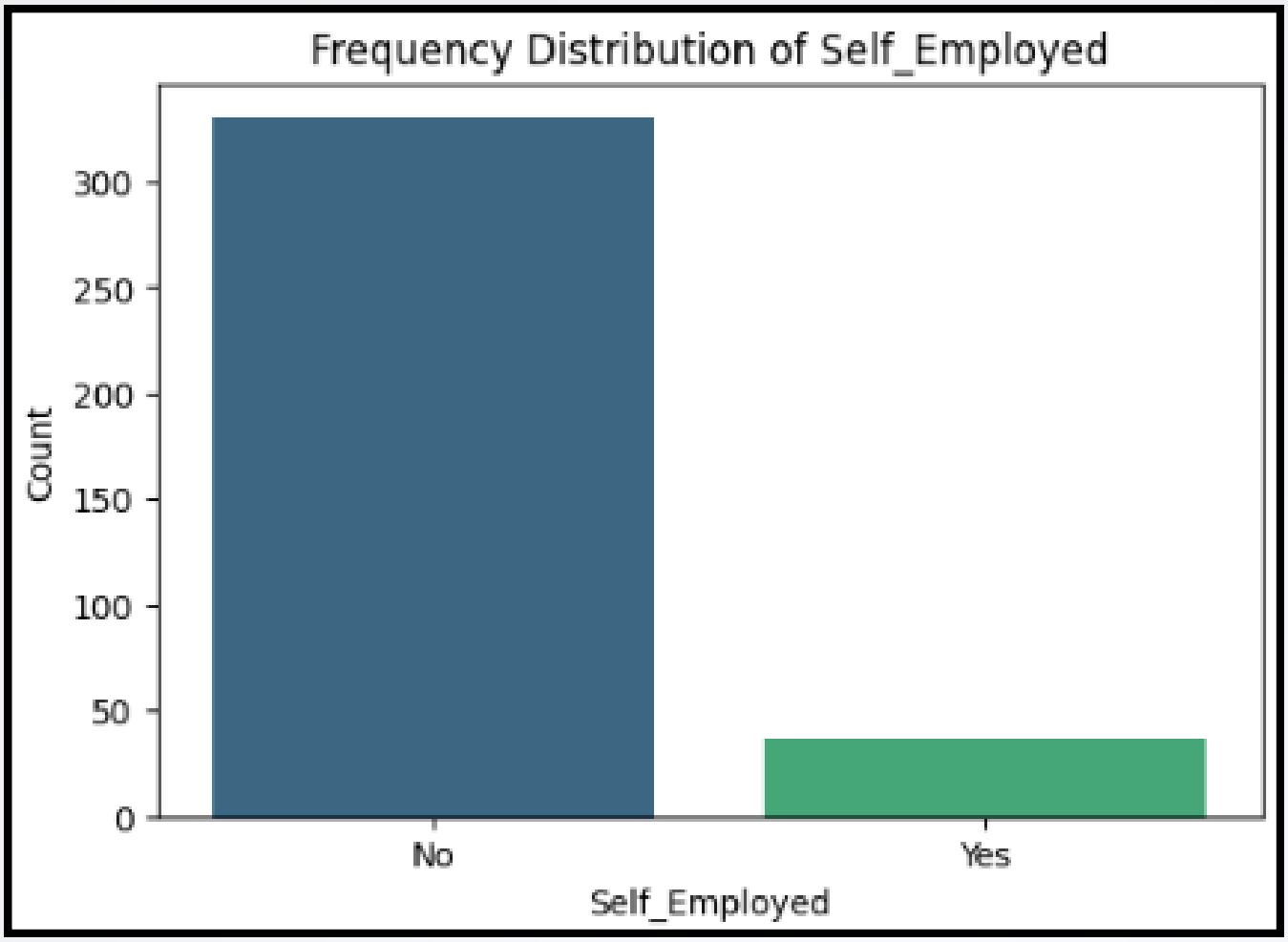
Insights

Education: A larger proportion of applicants are graduates as compared to non-graduates. This suggests that higher education may be positively associated with applying for loans possibly linked to better earning potential and creditworthiness.



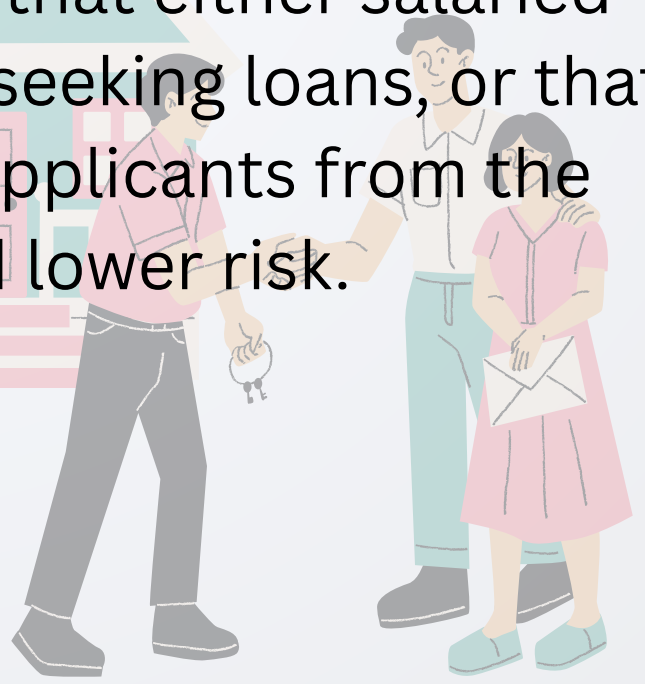
Self_Employed

Data Visualization



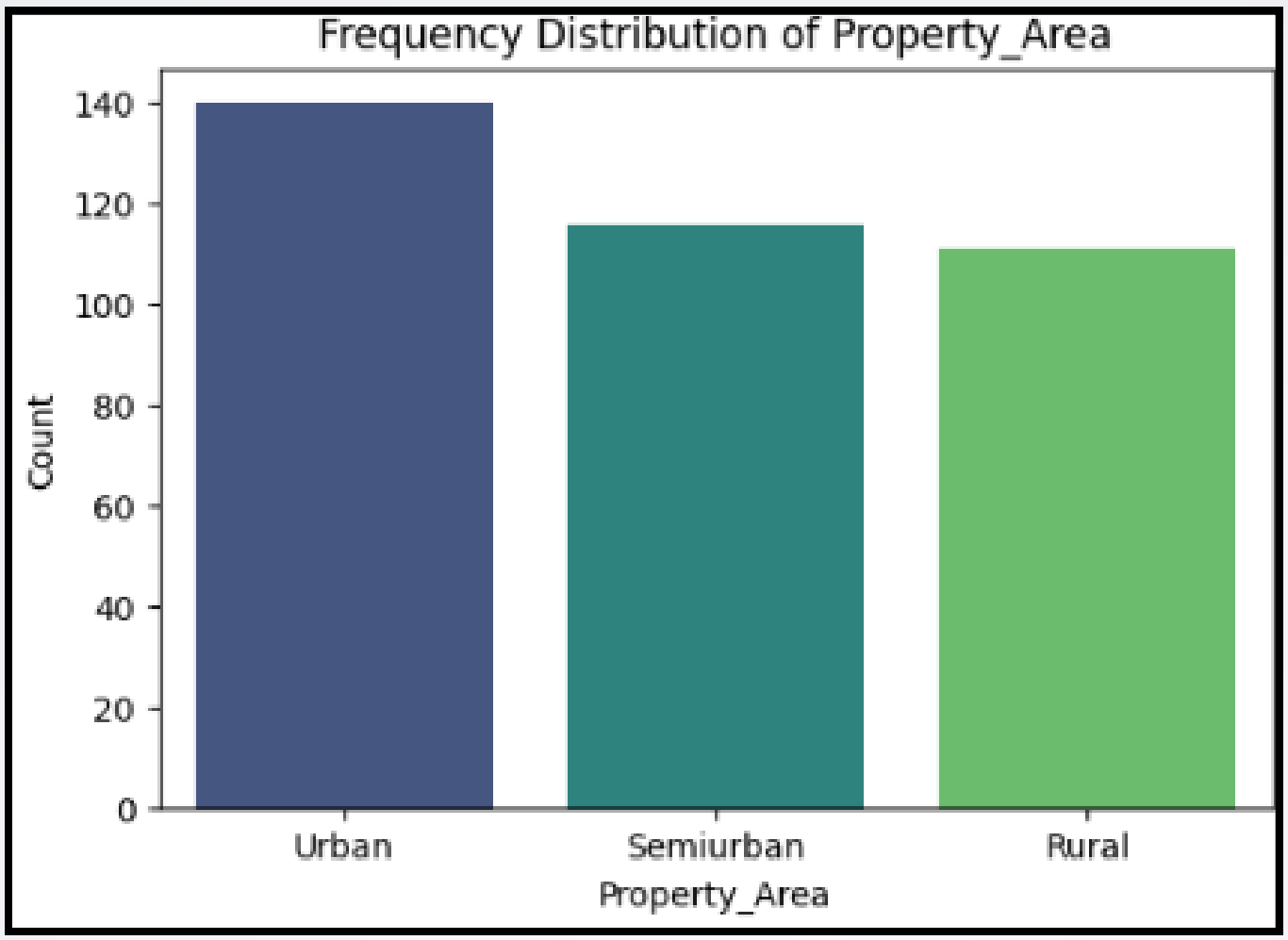
Insights

Self_Employed: Salaried applicants outnumber self-employed ones. This may indicate that either salaried individuals are more comfortable seeking loans, or that lending institutions attract more applicants from the salaried segment due to perceived lower risk.



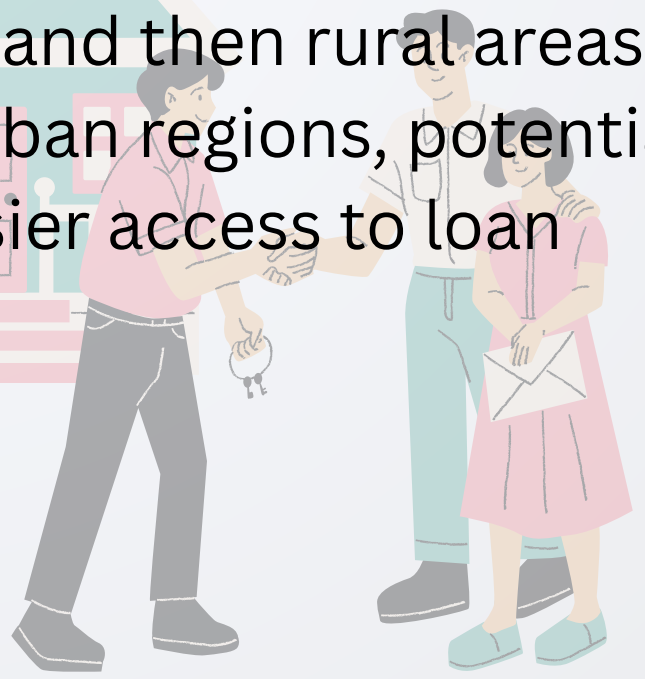
Property_Area

Data Visualization



Insights

Property_Area: Semiurban areas record the highest number of applications, followed by urban and then rural areas. This shows more loan activity in semiurban regions, potentially due to growth opportunities or easier access to loan services.

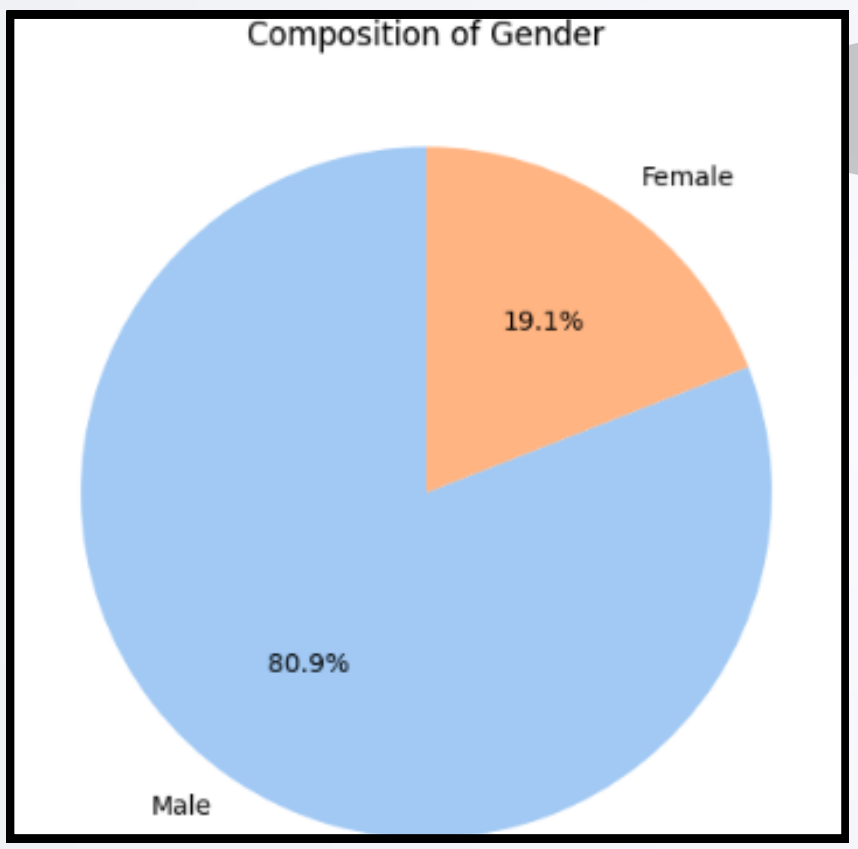


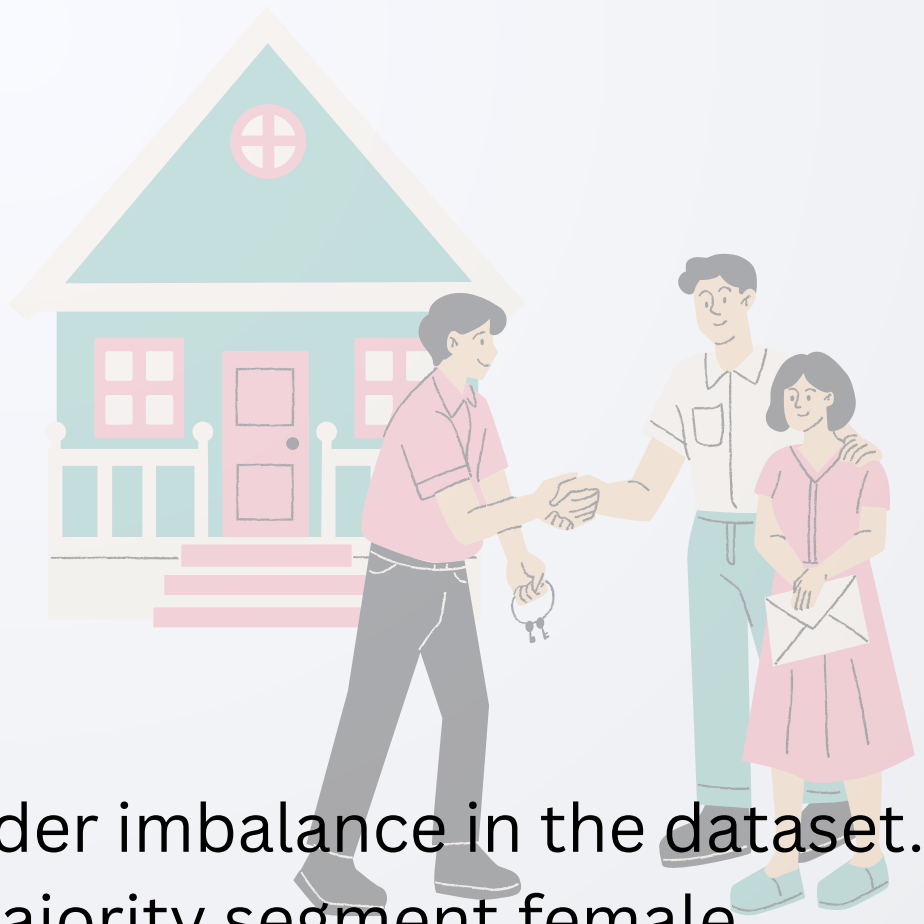
Data Visualization

- **Pie Charts:** Represent the composition of categorical variables.

```
cat_cols = ['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'Property_Area']

for col in cat_cols:
    plt.figure(figsize=(6, 6))
    colors = sns.color_palette("pastel", n_colors=df[col].nunique())
    df[col].value_counts().plot.pie(autopct='%1.1f%%', startangle=90, colors=colors)
    plt.title(f'Composition of {col}')
    plt.ylabel('')
    plt.show()
```

Gender





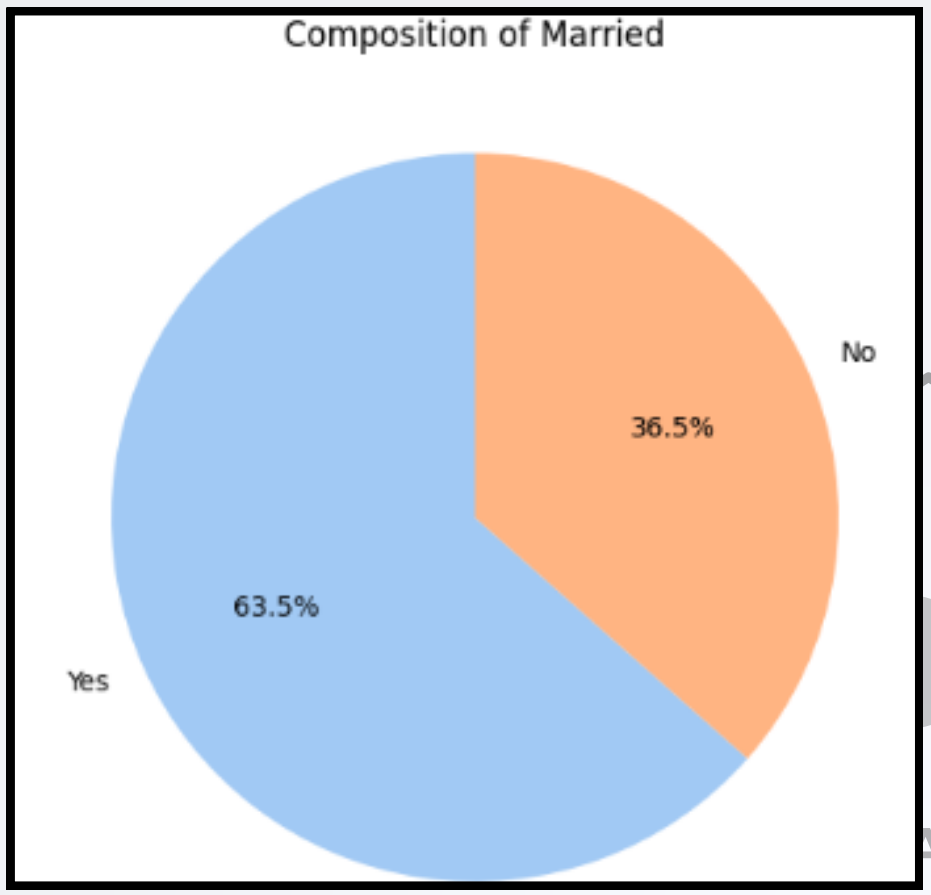
Insights

- **Male applicants:** 80.9%
- **Female applicants:** 19.1%
- Indicates a significant gender imbalance in the dataset.

Male applicants form the majority segment,female applicants represent a significantly smaller portion, indicating a substantial gender gap among borrowers.

Data Visualization

Married



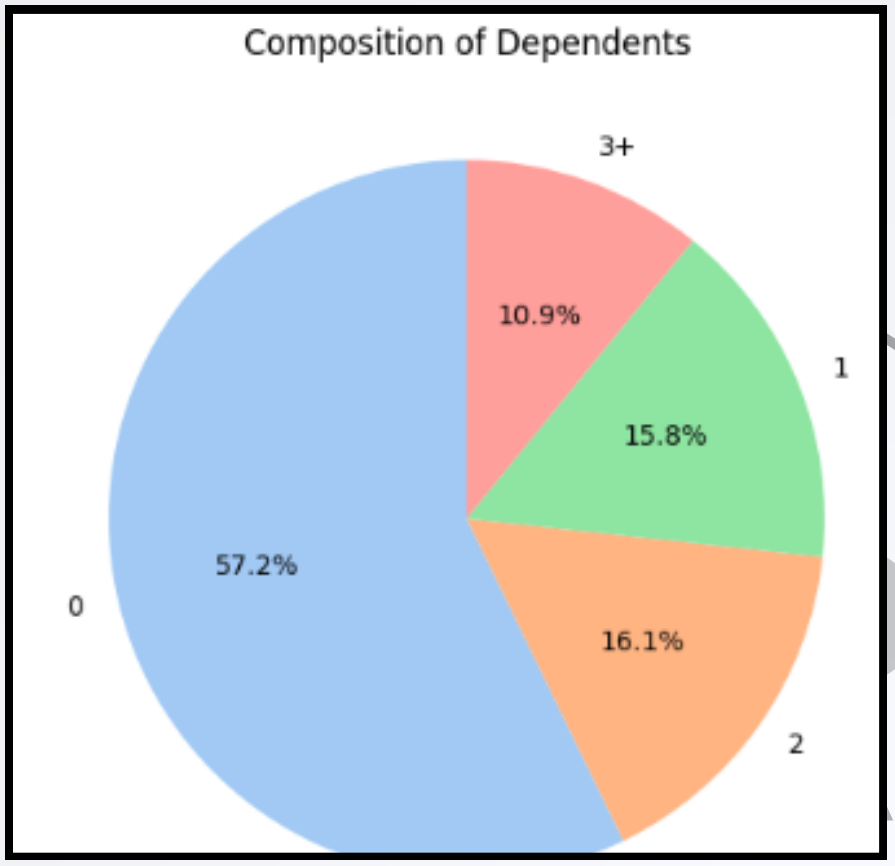
Insights

- **Married applicants:** 63.5%
- **Unmarried applicants:** 36.5%
- Indicates that the majority of loan applicants are married.
- Marital status may be a relevant factor in loan eligibility or approval trends.

Married individuals constitute most loan seekers, with a large majority slice. Single applicants form a minority, suggesting that married individuals may perceive higher need or eligibility for loans, possibly due to combined income or family responsibilities.

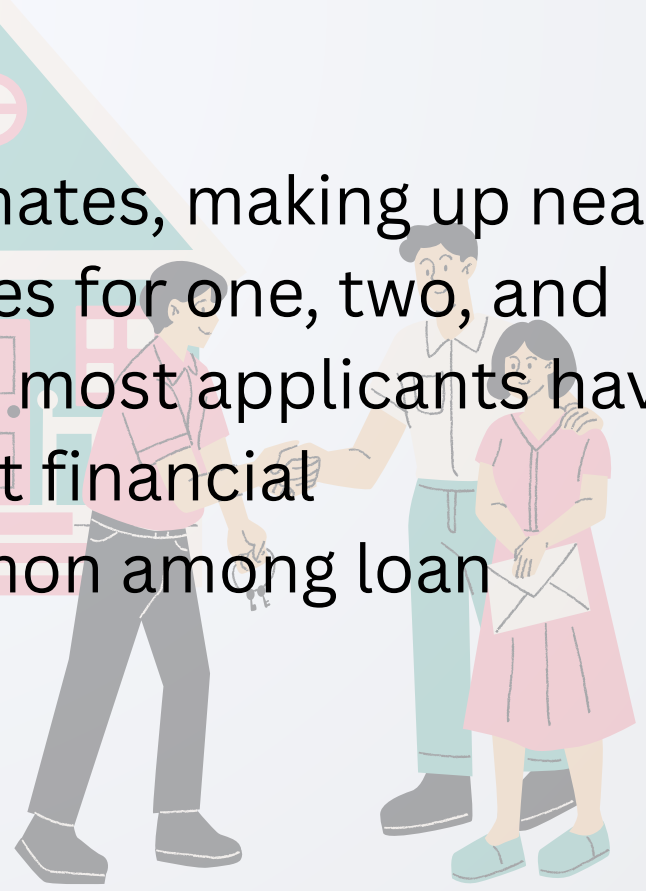
Data Visualization

Dependents



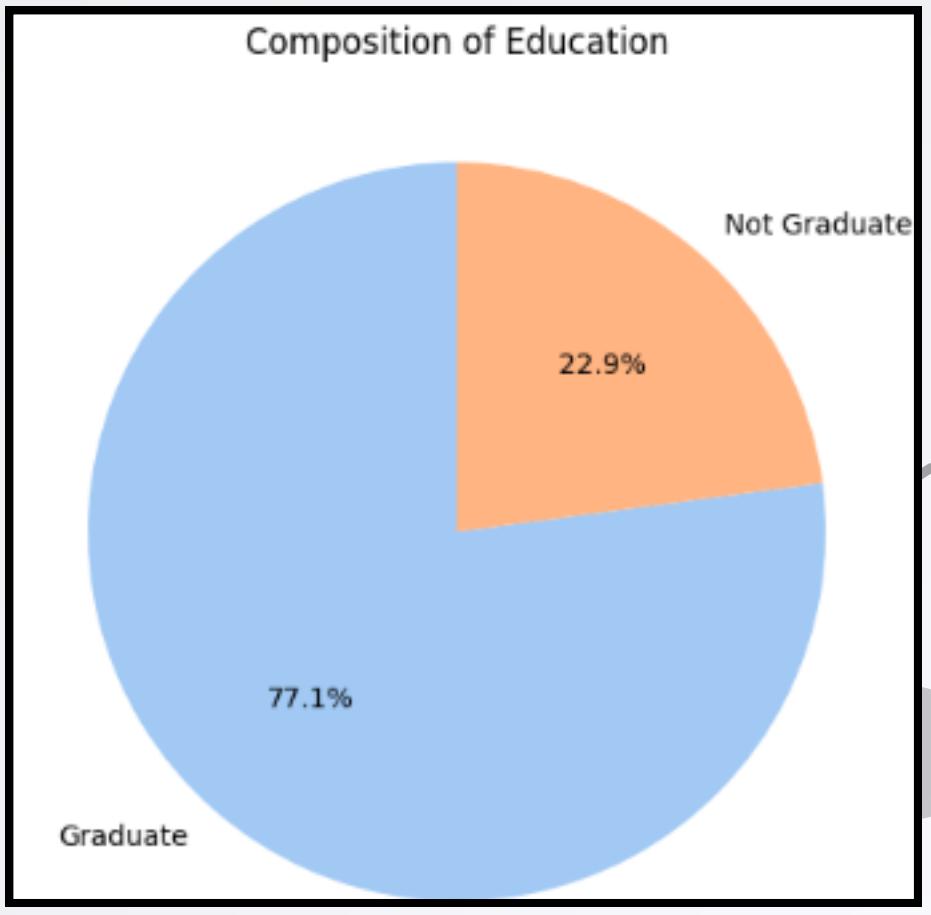
Insights

The zero dependents category dominates, making up nearly half or more of applicants. Small slices for one, two, and three or more dependents show that most applicants have few or no dependents, indicating that financial responsibility for others is less common among loan seekers.



Education

Data Visualization



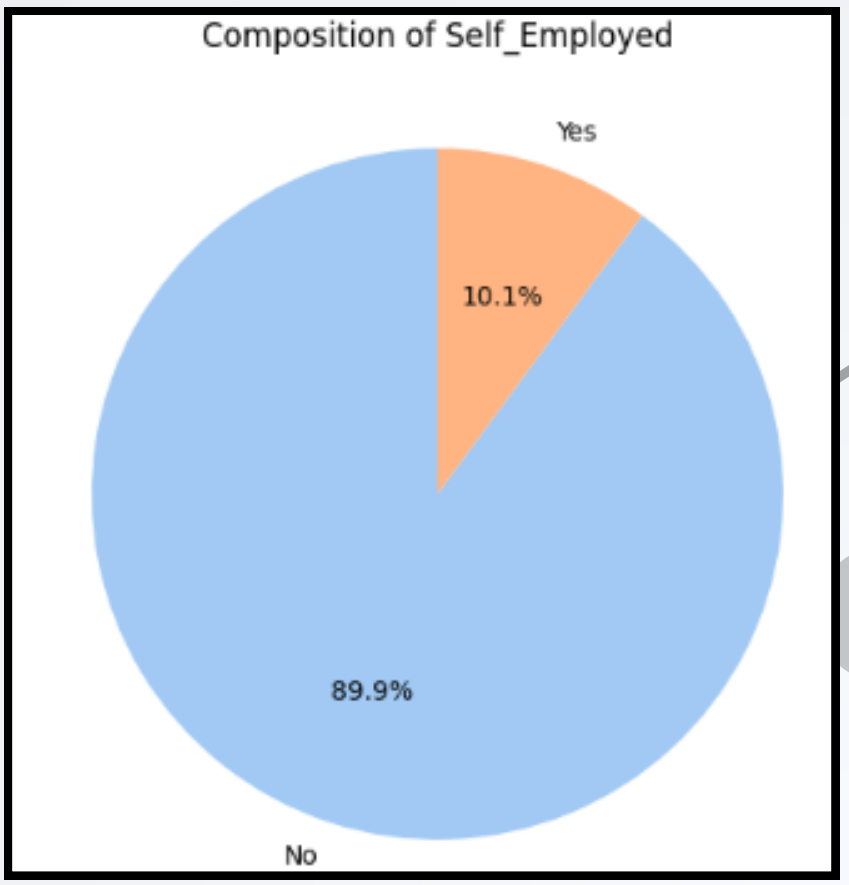
Insights

- **Graduates:** 77.1%
- **Not Graduates:** 22.9%
- Majority of applicants have graduation-level education.
- Education may influence loan approval, creditworthiness, or income stability.

Graduates account for the largest share of applications, suggesting higher educational attainment among borrowers. Non-graduates, while present, form a notably smaller slice, possibly hinting that educational background is linked to credit access or confidence in financial management.

Data Visualization

Self_Employed



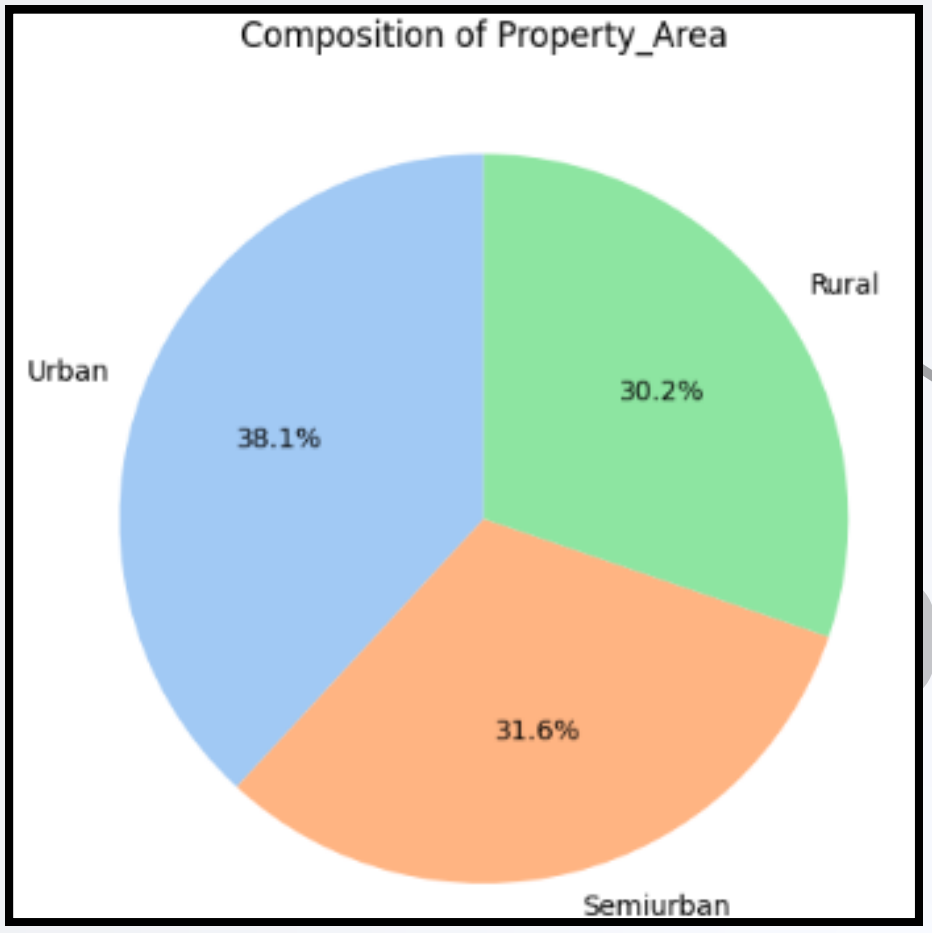
Insights

- **Not Self-Employed:** 89.9%
- **Self-Employed:** 10.1%
- Majority of applicants are salaried or employed by others.

Salaried applicants comprise the majority, while self-employed individuals represent a much smaller percentage. This could reflect a preference by lending institutions for the perceived stability of salaried incomes, or lower participation by self-employed borrowers.

Data Visualization

Property_Area



Insights

•**Urban applicants:** 38.1% (highest proportion)
- Semiurban applicants: 31.6%
-Rural applicants: 30.2%
The dataset is fairly balanced across all areas, with a slight dominance ofurban regions. - Suggests loans are accessed by people from diverse geographic locations, supporting a wide-ranging customer base.
The relatively smaller slice for rural areas may indicate limited access or lower demand for loans, with semiurban regions being key markets for financial products.

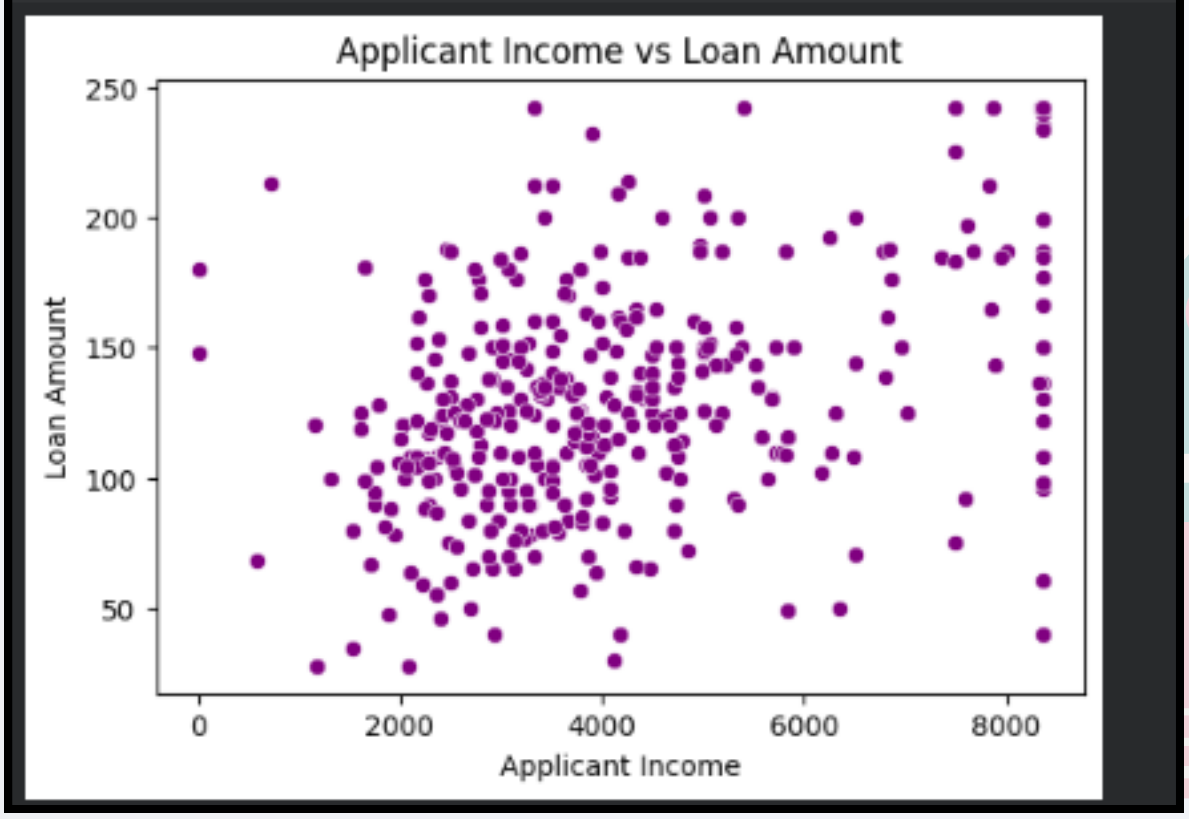
Data Visualization

BIVARIENT ANALYSIS

- **Create scatter plots to explore relationships between pairs of numeric variables.**

Applicant Income vs Loan Amount

```
plt.figure(figsize=(6, 4))
sns.scatterplot(data=df, x='ApplicantIncome', y='LoanAmount',
color='purple')
plt.title('Applicant Income vs Loan Amount')
plt.xlabel('Applicant Income')
plt.ylabel('Loan Amount')
plt.show()
```



Insights

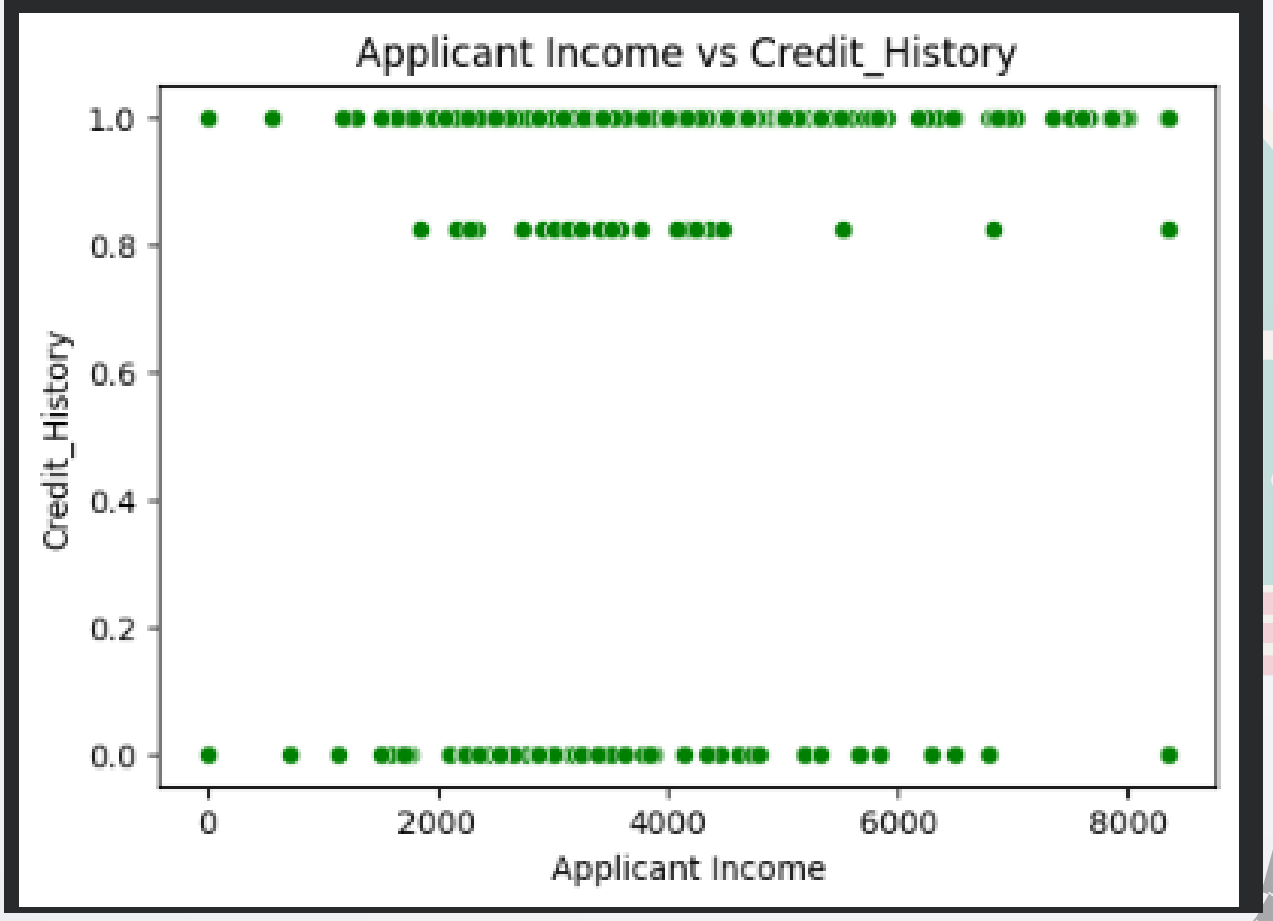
1.Applicants with higher incomes generally apply for larger loan amounts, indicating income is a strong influencing factor in loan size.

Data Visualization

2.For middle-range incomes, loan amounts vary more widely, suggesting additional factors
(like coapplicant income or credit history) may influence loan decisions.

Scatter Plot in Applicant Income vs Credit_History

```
plt.figure(figsize=(6, 4))
sns.scatterplot(data=df, x='ApplicantIncome', y='Credit_History',
color='green')
plt.title('Applicant Income vs Credit_History')
plt.xlabel('Applicant Income')
plt.ylabel('Credit_History')
plt.show()
```



Data Visualization

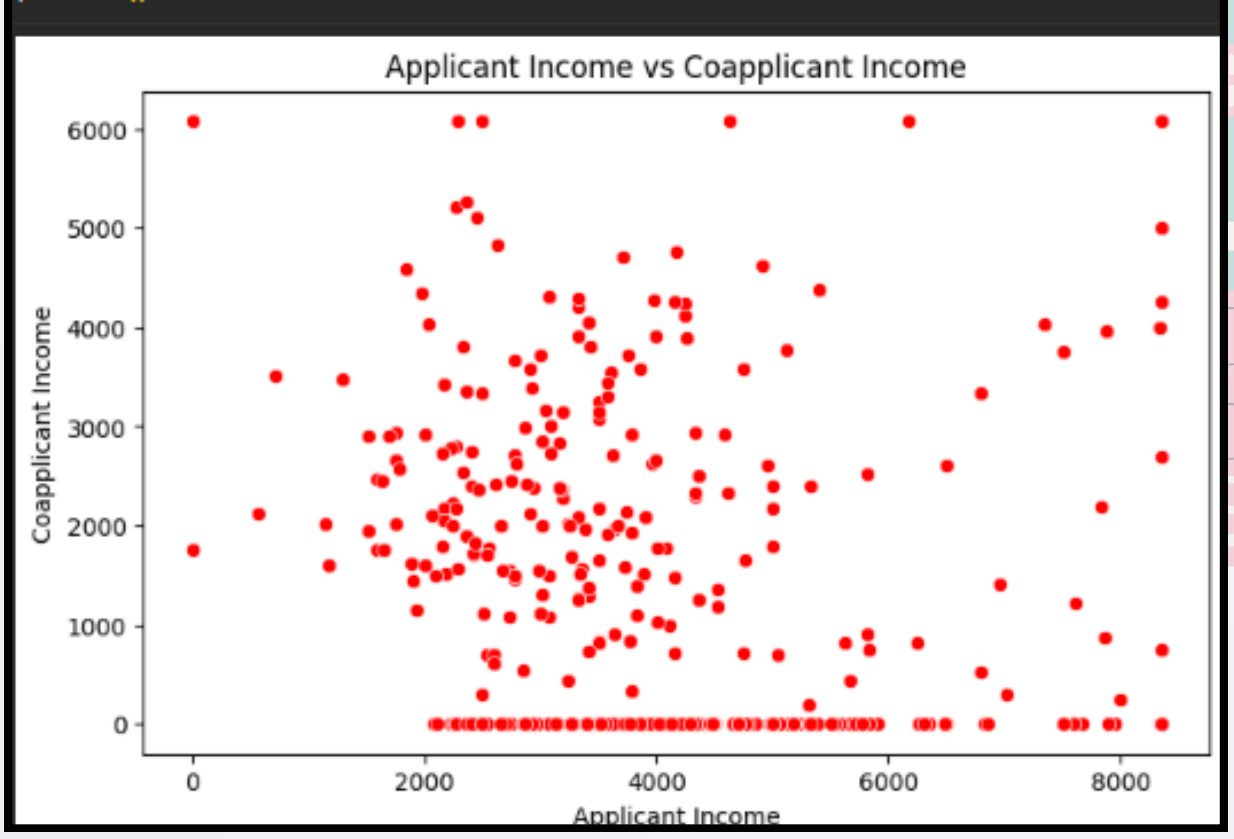
Insights

1.Applicant income does not show a clear relationship with credit history values (0.0 or 1.0). Both low- and high-income applicants appear in both credit history categories.

2.Good or bad credit history is not limited to a particular income group, suggesting it depends more on financial behavior than earning capacity.

Scatter Plot in ApplicantIncome vs CoapplicantIncome

```
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df, x='ApplicantIncome',
y='CoapplicantIncome',color="red")
plt.title('Applicant Income vs Coapplicant Income')
plt.xlabel('Applicant Income')
plt.ylabel('Coapplicant Income')
plt.show()
```



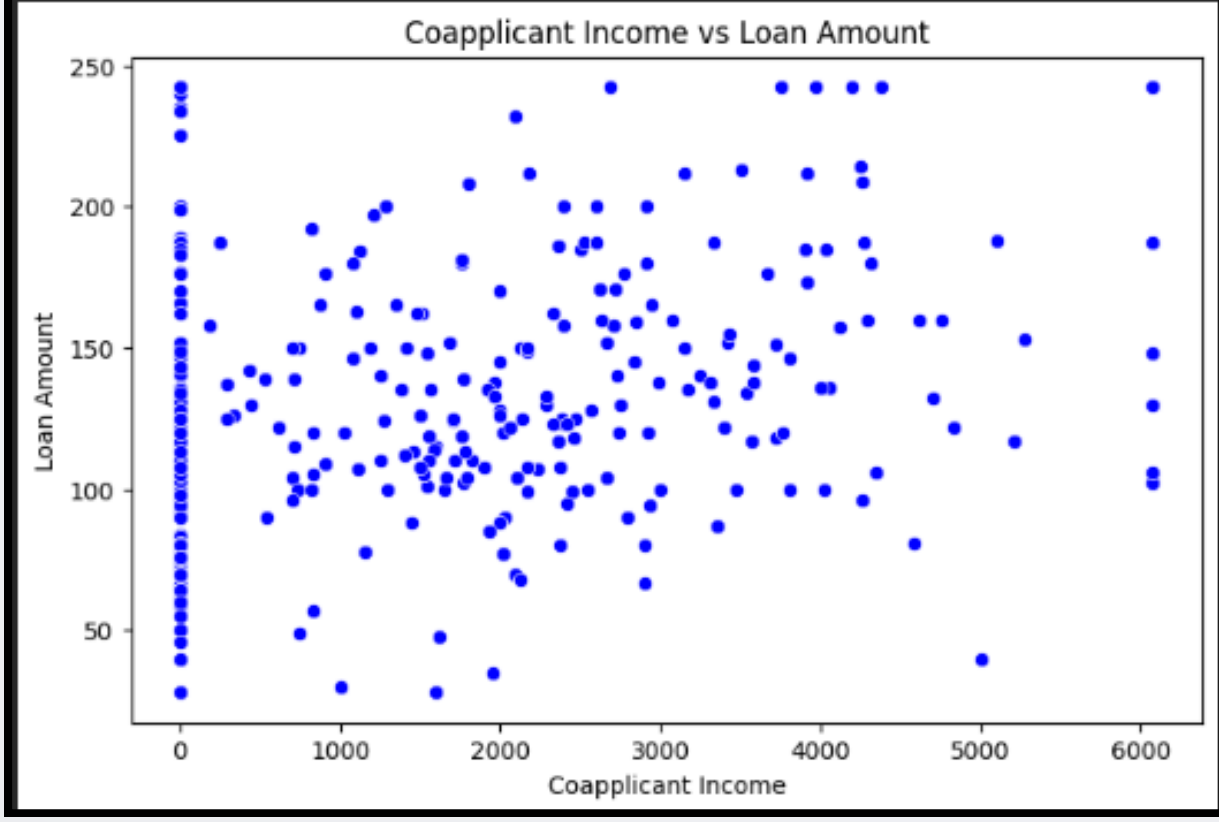
Data Visualization

Insights

1.A large cluster appears where coapplicant income is zero, indicating many applicants apply individually without coapplicants.
2.There are fewer cases where both applicant and coapplicant have high incomes, suggesting dual high-income applications are less common.
3.Applicant income tends to be the primary source, with coapplicant income acting as a supplementary factor for a smaller group

CoapplicantIncome vs LoanAmount

```
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df, x='CoapplicantIncome',
y='LoanAmount',color='blue')
plt.title('Coapplicant Income vs Loan Amount')
plt.xlabel('Coapplicant Income')
plt.ylabel('Loan Amount')
plt.show()
```



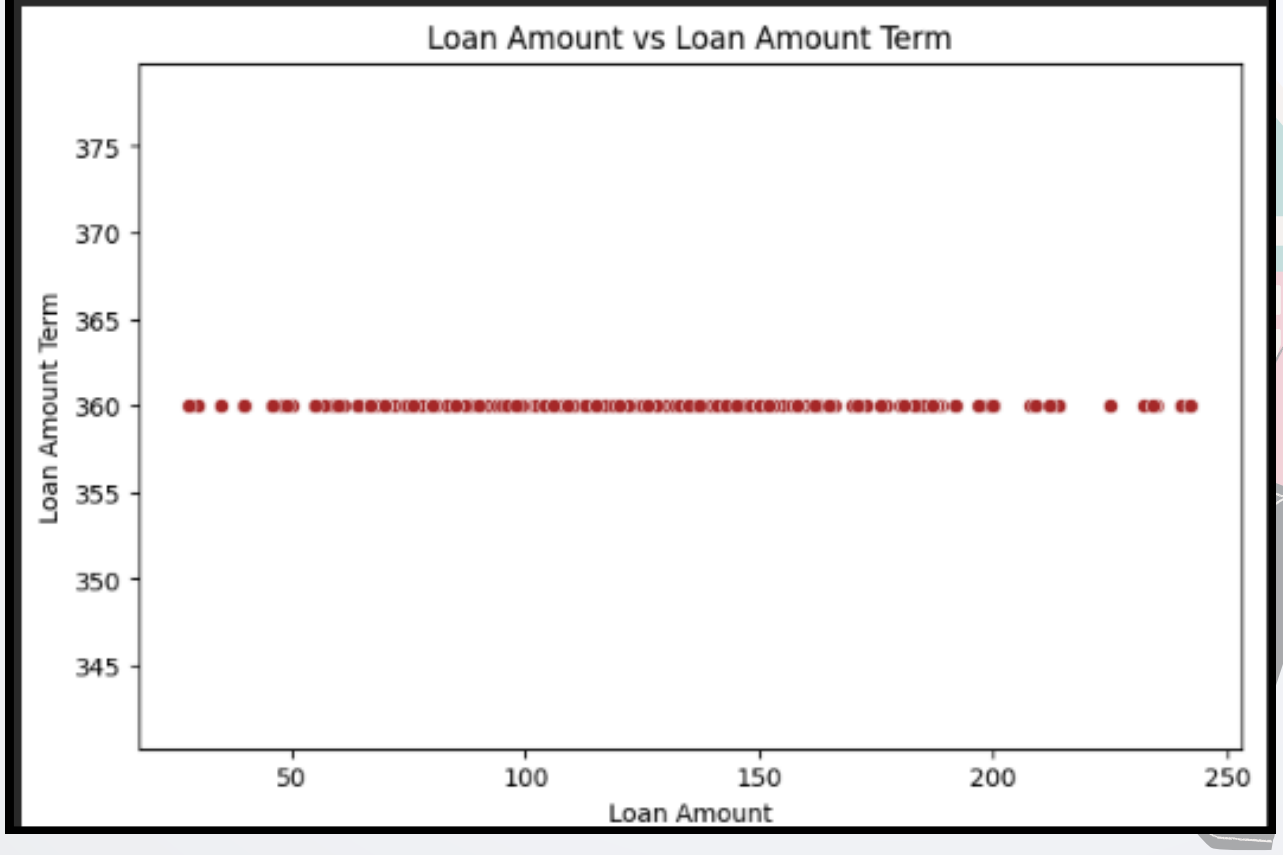
Data Visualization

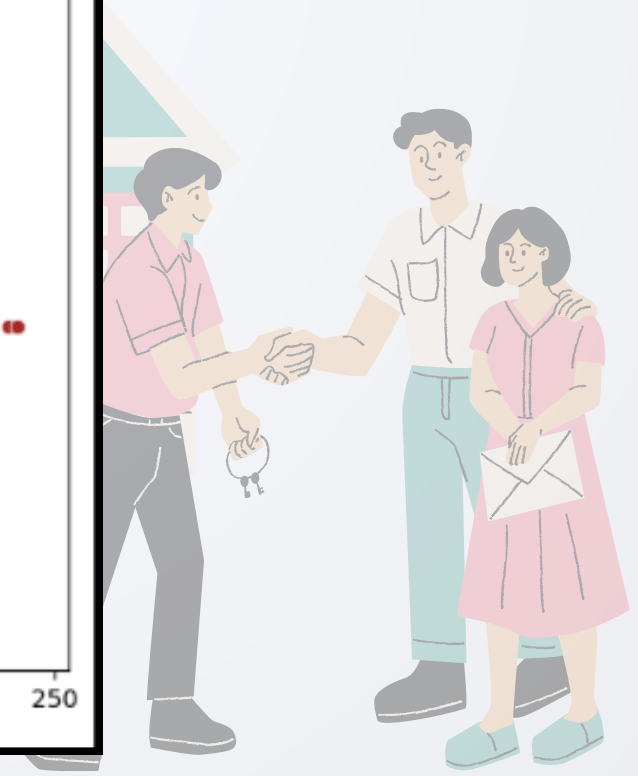
Insights

1.Loan amounts show no strong or consistent increase with higher coapplicant incomes. Many applicants with zero or low coapplicant income still receive varied loan amounts.
2.The plot suggests that applicant income, rather than coapplicant income, is the primary factor influencing loan amount decisions.

LoanAmount vs Loan_Amount_Term

```
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df, x='LoanAmount',
y='Loan_Amount_Term',color="brown")
plt.title('Loan Amount vs Loan Amount Term')
plt.xlabel('Loan Amount')
plt.ylabel('Loan Amount Term')
plt.show()
```





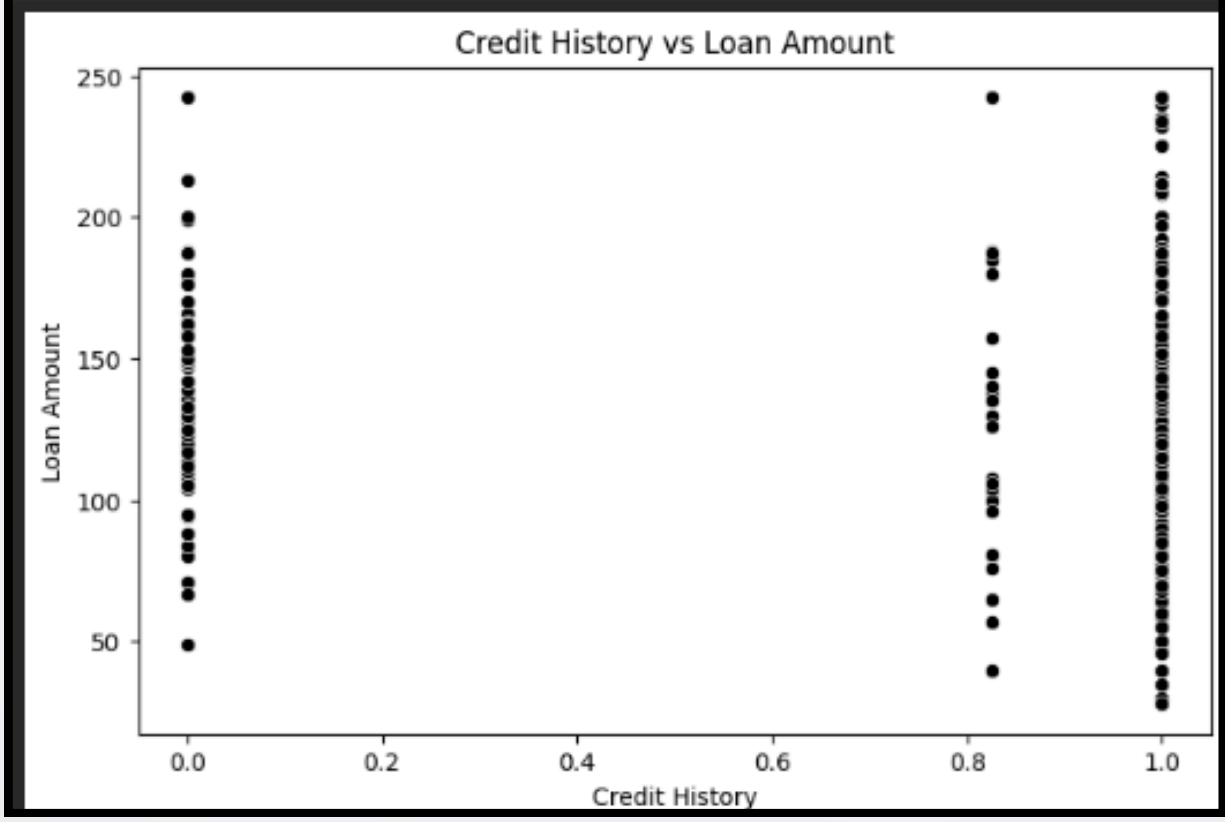
Data Visualization

Insights

1.Most loan terms cluster around specific values (e.g., 360 months), regardless of loan amount. This indicates financial institutions use standard loan term options rather than customizing terms per loan amount.
2.Both small and large loan amounts are often assigned the same loan term, showing that loan term decisions may be policy-driven rather than amount-driven.
3.Only a small number of records show variation from the standard loan term, suggesting limited flexibility in term options for applicants.

Credit_History vs LoanAmount

```
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df, x='Credit_History', y='LoanAmount', color='black')
plt.title('Credit History vs Loan Amount')
plt.xlabel('Credit History')
plt.ylabel('Loan Amount')
plt.show()
```



Data Visualization

Insights

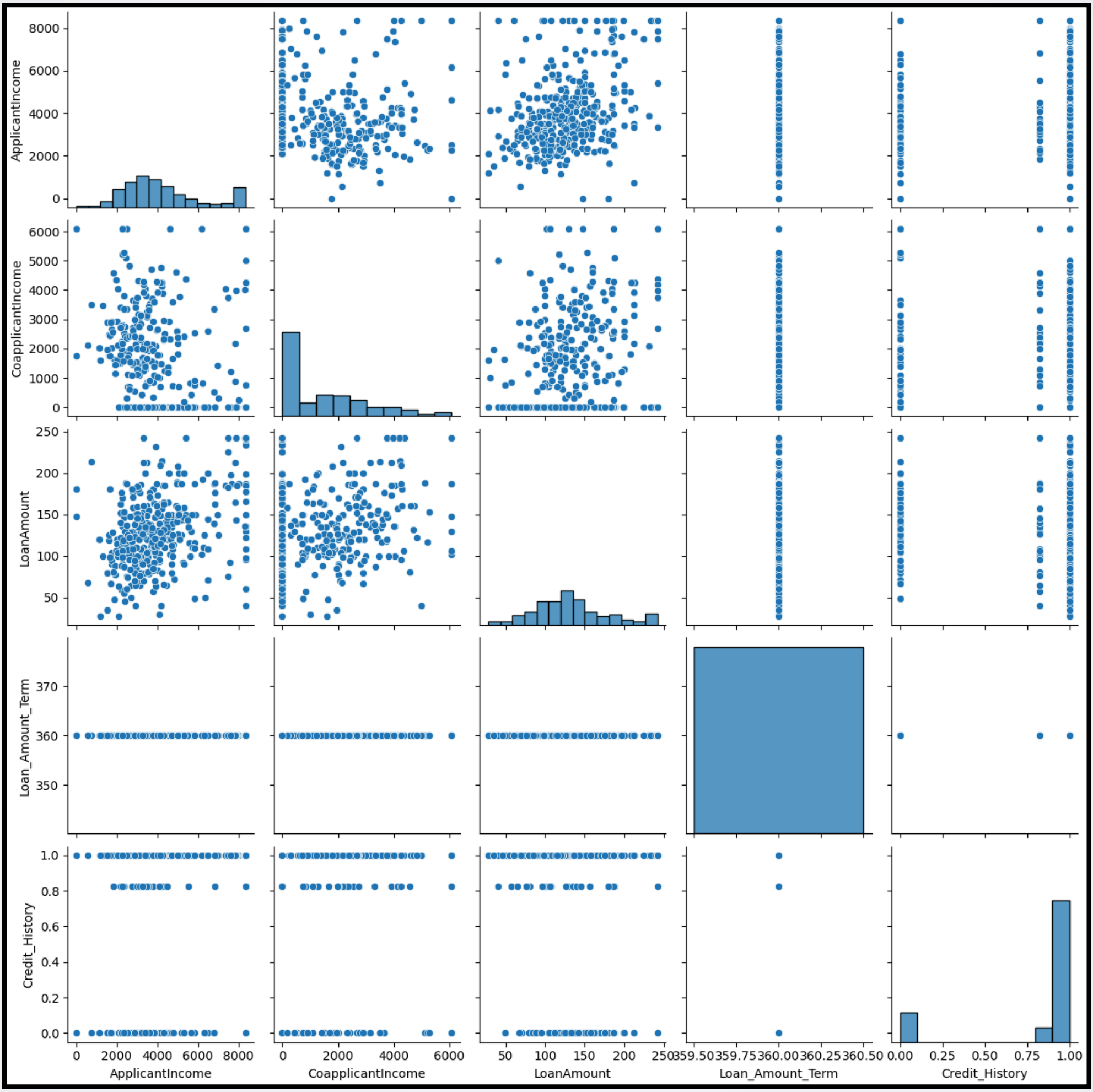
1.Applicants with a credit history value of 1.0 generally receive higher loan amounts compared to those with no or poor credit history (0.0).
2.Loan amounts for applicants with credit history value 0.0 are mostly clustered at lower values, showing risk-averse lending behavior by financial institutions.
3.The scatter plot shows two distinct vertical bands, indicating that credit history is a decisive factor in loan sanction amounts.

HOME LOAN ANALYSIS

- **Use pair plots (scatter matrix) to visualize interactions between multiple numeric variables simultaneously.**

```
df_1=df.select_dtypes(include=["int64","float64"])
sns.pairplot(df_1)
```

Data Visualization

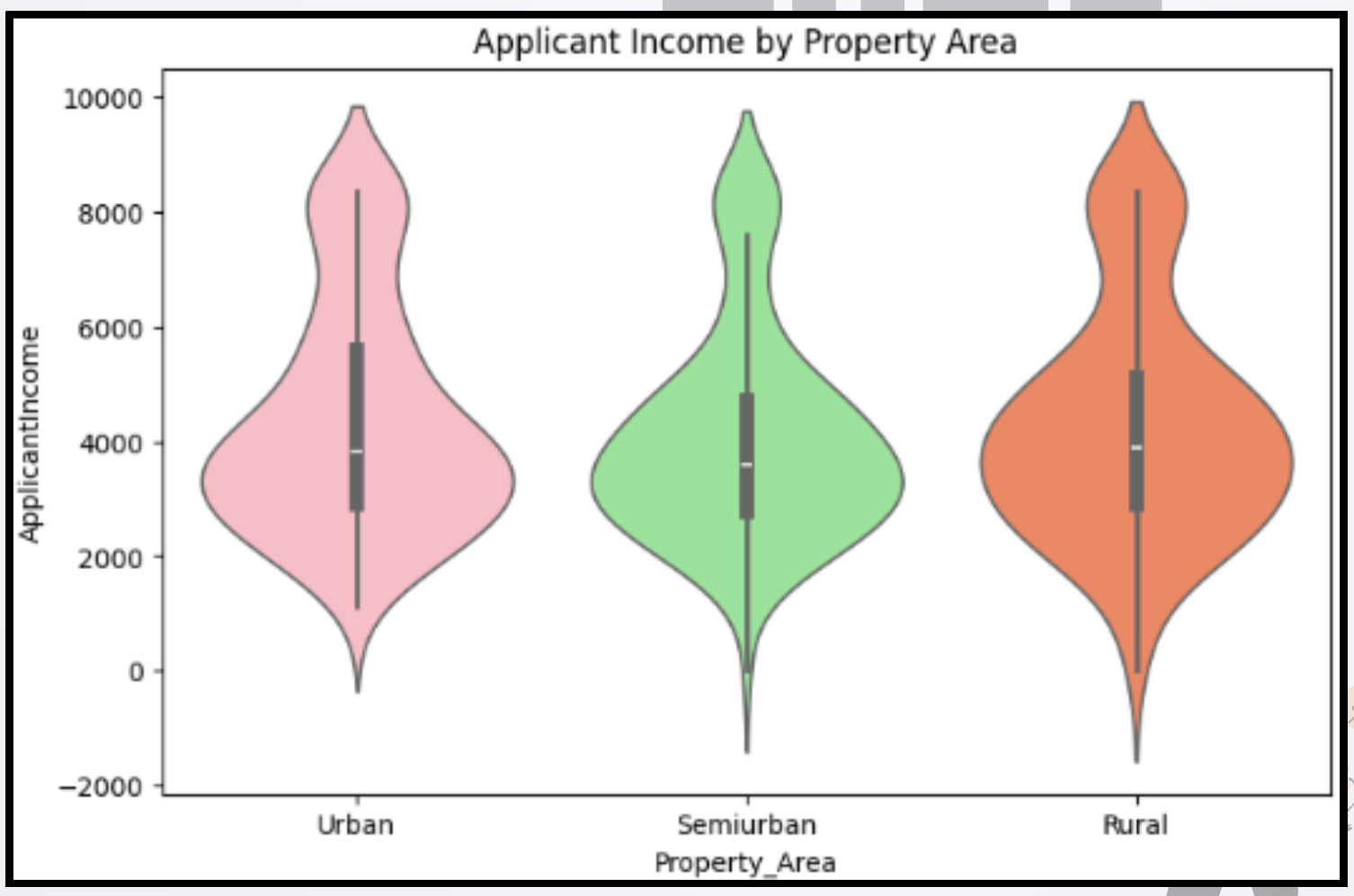


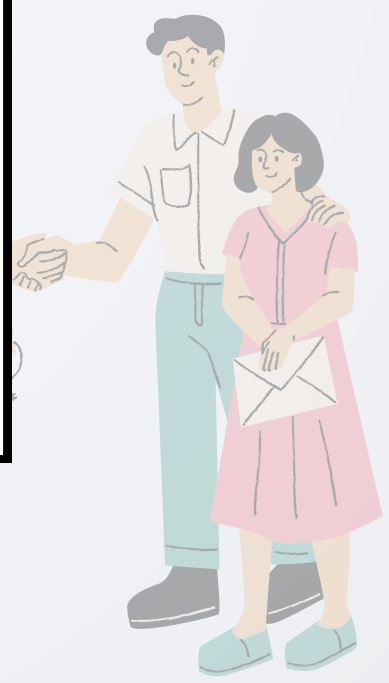
Data Visualization

- **Investigate the relationship between categorical and numeric variables using boxplots or violin plots.**

Violinplot of Applicant Income by Property Area

```
plt.figure(figsize=(8, 5))
sns.violinplot(x='Property_Area', y='ApplicantIncome', data=df,
palette=["lightpink","lightgreen","coral"])
plt.title('Applicant Income by Property Area')
plt.show()
```





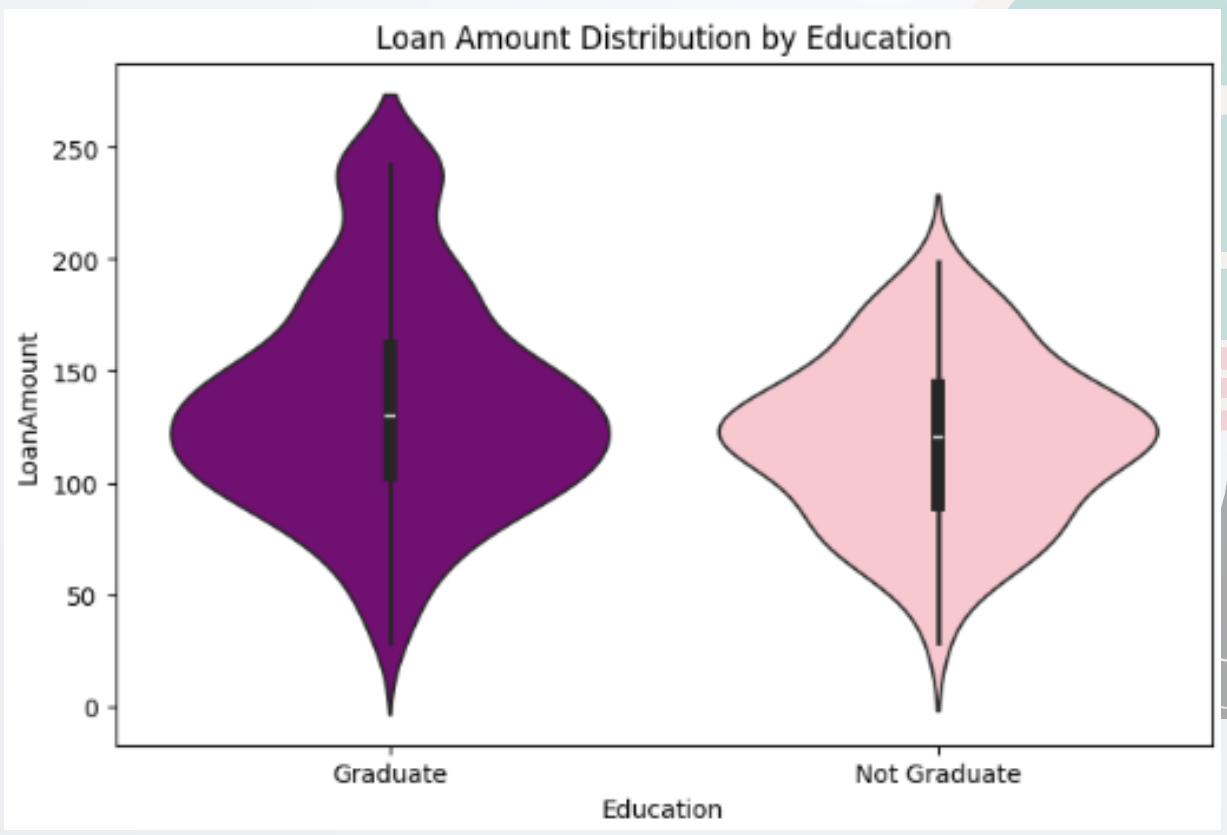
Insights:

Data Visualization

1.Urban property areas show a broader distribution of applicant incomes, with more high□income outliers compared to rural and semiurban areas.
2.Applicants from rural areas generally have lower median incomes, with fewer high□income applicants observed.
3.Semiurban areas show a more concentrated distribution around the median, suggesting a more uniform income profile.

Violinplot of Loan Amount Distribution by Education

```
plt.figure(figsize=(8, 5))
sns.violinplot(x='Education', y='LoanAmount', data=df,
palette=["#800080","#FFC0CB"]) # Using hex codes for purple and pink
plt.title('Loan Amount Distribution by Education')
plt.show()
```



Data Visualization

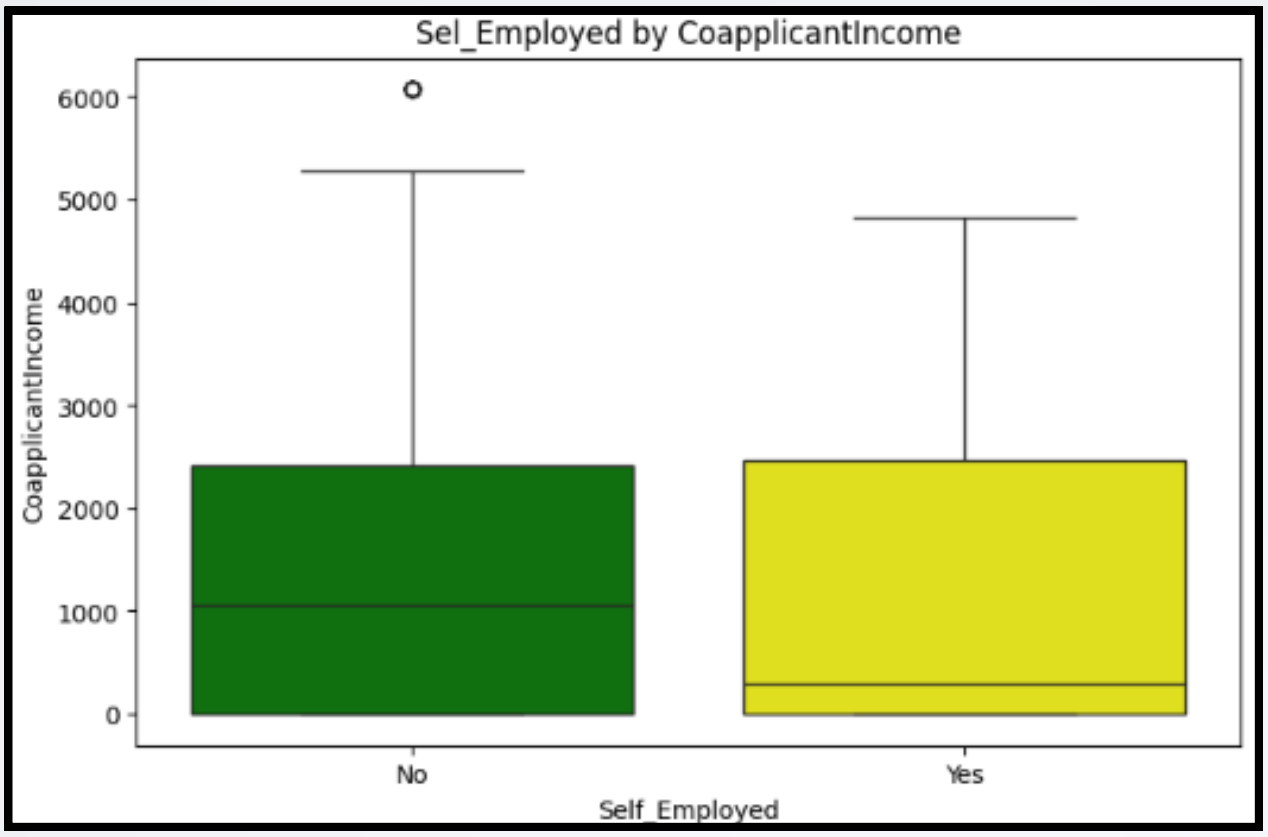
Insights:

1.Both graduates and non-graduates show similar median loan amounts, indicating education level may not drastically affect the typical loan size.
2.Graduates have a broader range of loan amounts, including higher-value loans, suggesting more diverse borrowing needs or financial capacity.
3.Non-graduate applicants mostly take smaller, more consistent loan amounts, indicating more conservative borrowing behavior.

Boxplot of Self_Employed by CoapplicantIncome

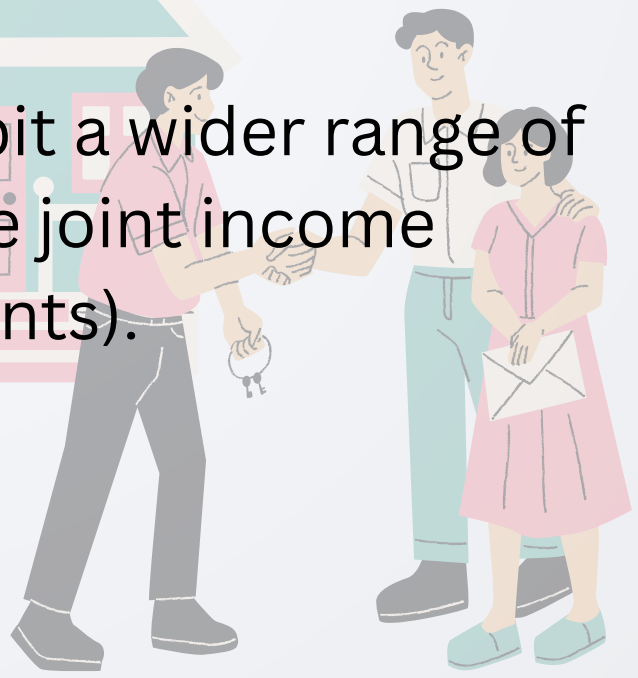
```
plt.figure(figsize=(8, 5))
sns.boxplot(x='Self_Employed', y='CoapplicantIncome', data=df,
palette=["green","yellow"])
plt.title('Sel_Employed by CoapplicantIncome')
plt.xlabel('Self_Employed')
plt.ylabel('CoapplicantIncome')
plt.show()
```

Data Visualization



Insights:

1.Self-employed applicants generally show lower or zero coapplicant incomes compared to non-self-employed applicants.
2.Non-self-employed applicants exhibit a wider range of coapplicant incomes, suggesting more joint income opportunities (e.g., salaried coapplicants).

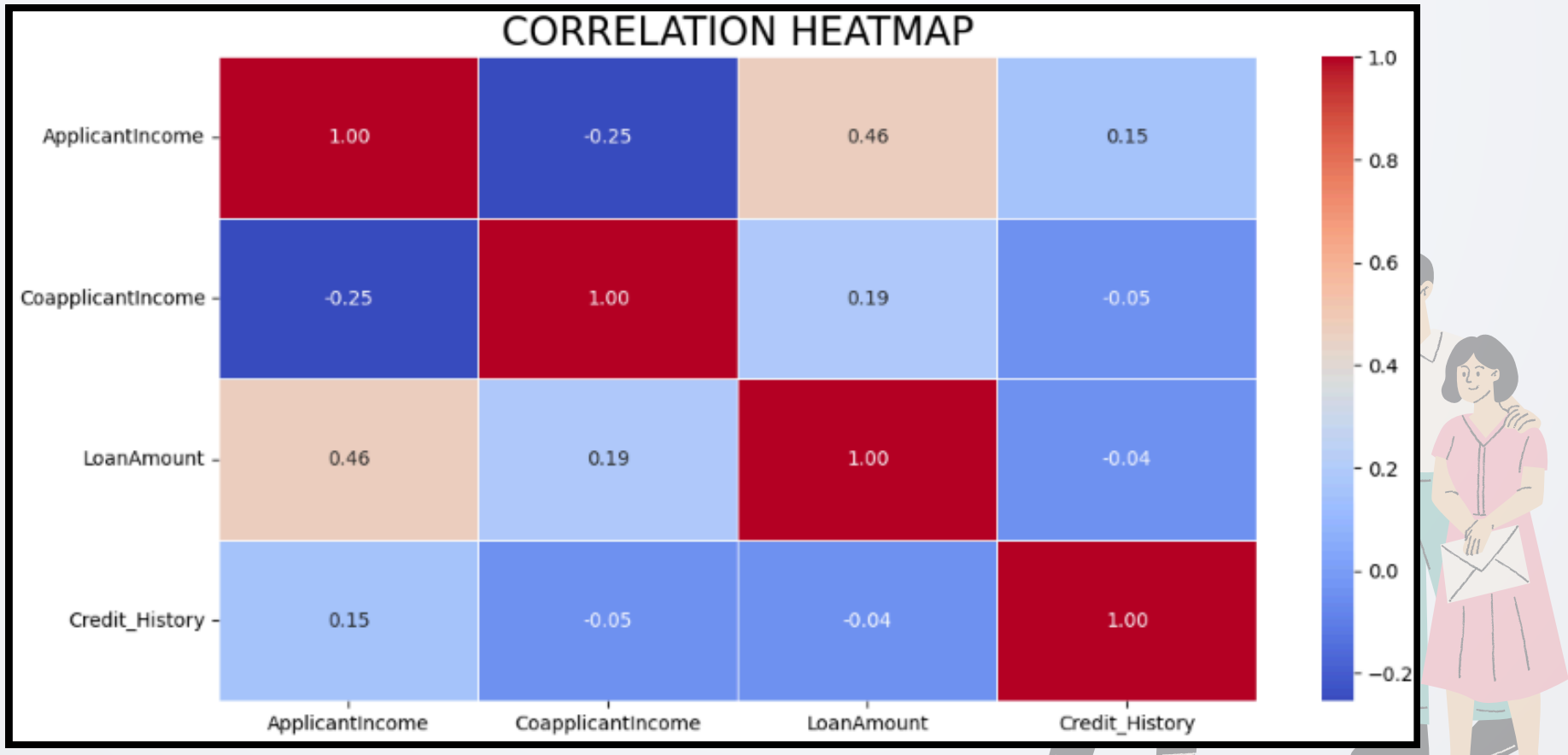


Data Visualization

MULTIVARIENT ANALYSIS

- Perform a correlation analysis to identify relationships between numeric variables. Visualize correlations using a heatmap.

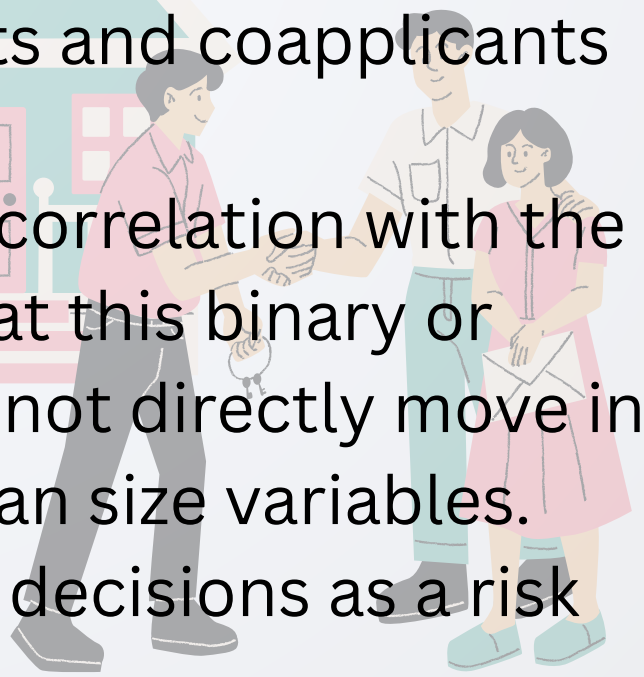
```
num_cols = ['ApplicantIncome', 'CoapplicantIncome', 'LoanAmount','Credit_History']
corr = df[num_cols].corr()
plt.figure(figsize=(12,6))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('CORRELATION HEATMAP',fontsize=20)
plt.show()
```



Data Visualization

Insights

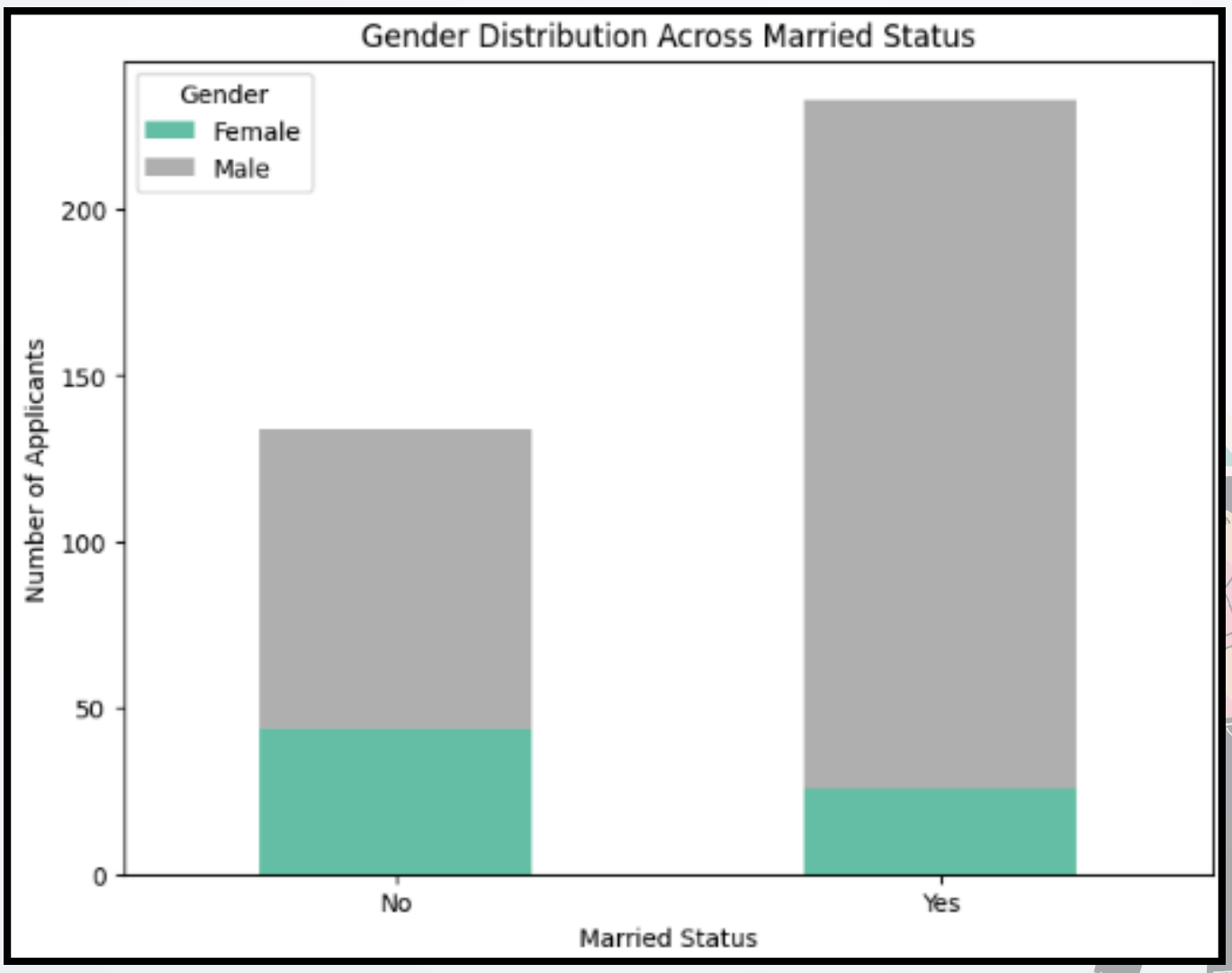
- ApplicantIncome and LoanAmount have a positive correlation, indicating that higher applicant incomes tend to be associated with larger loan amounts. This aligns with the intuitive understanding that borrowers with higher incomes request larger loans, possibly due to higher repayment capacity.
- CoapplicantIncome shows a moderate positive correlation with LoanAmount, suggesting that when there is a coapplicant with higher income, the total loan amount requested tends to be bigger. This highlights the combined earning potential's role in loan size determination.
- The correlation between ApplicantIncome and CoapplicantIncome is very low or near zero, which suggests that the incomes of primary applicants and coapplicants vary independently in this dataset.
- Credit_History has weak or negligible correlation with the other numeric variables, reflecting that this binary or categorical feature (often 0 or 1) may not directly move in tandem with continuous income or loan size variables. However, it remains vital for approval decisions as a risk indicator.

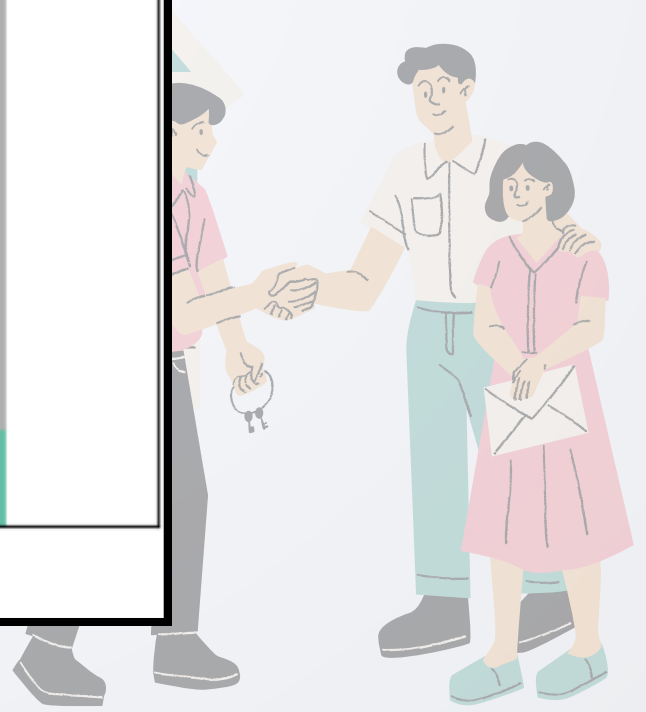


Data Visualization

- **Create a stacked bar chart to show the distribution of categorical variables across multiple categories.**

```
ct = pd.crosstab(df['Married'], df['Gender'])

ct.plot(kind='bar', stacked=True, figsize=(8,6), colormap='Set2')

plt.title('Gender Distribution Across Married Status')
plt.xlabel('Married Status')
plt.ylabel('Number of Applicants')
plt.xticks(rotation=0)
plt.legend(title='Gender')
plt.show()
```





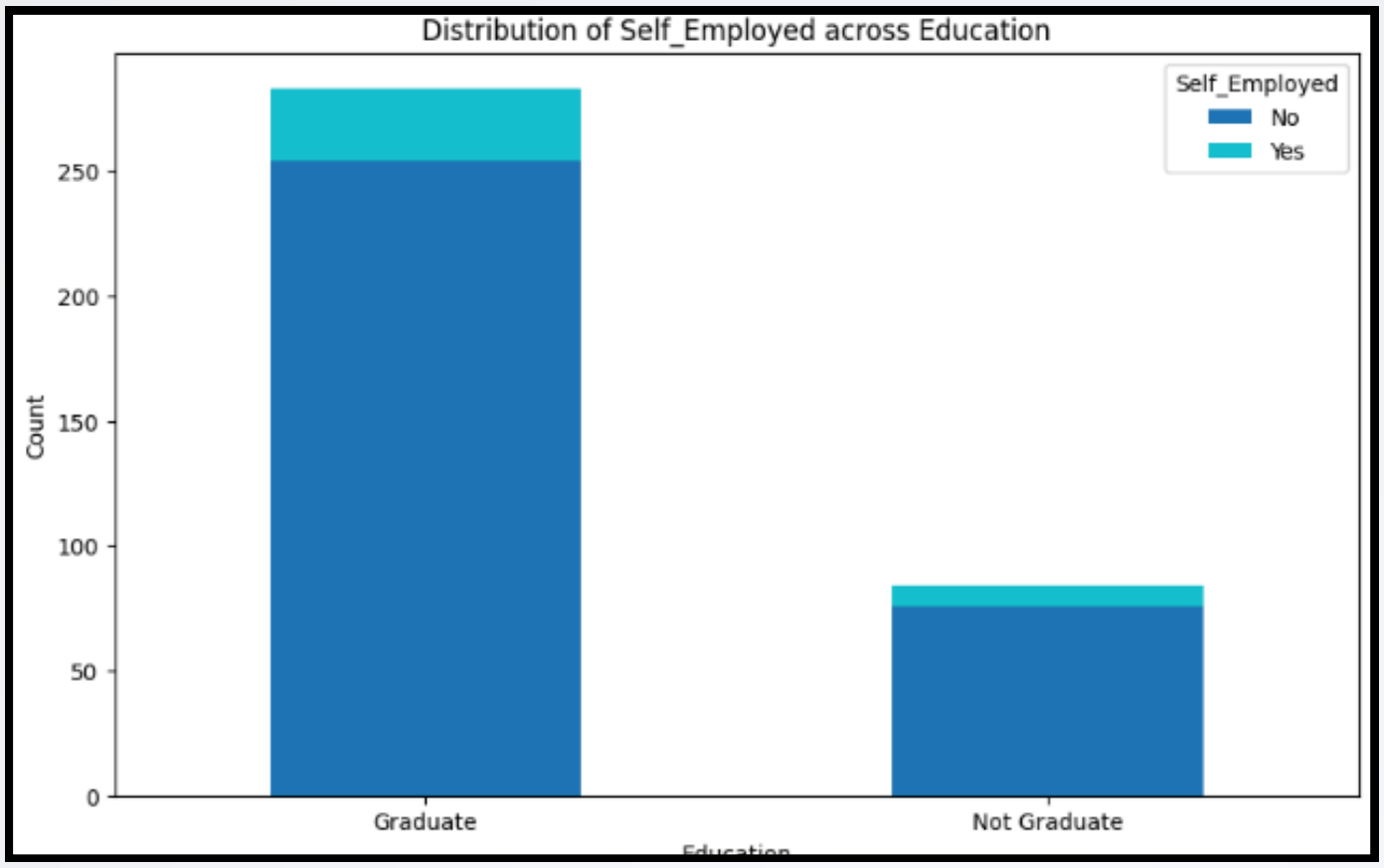
Data Visualization

Insights

- The stacked bar chart illustrating Gender distribution across Married status reveals distinct demographic patterns in the loan applicant pool. It shows that a significantly larger proportion of married applicants are male, indicating that males are more likely to be married loan applicants. Conversely, among unmarried applicants, while males still dominate, the proportion of female applicants is comparatively higher than in the married group.
- Such insights are valuable for designing targeted lending products and tailoring credit risk assessments that consider borrower demographics effectively.

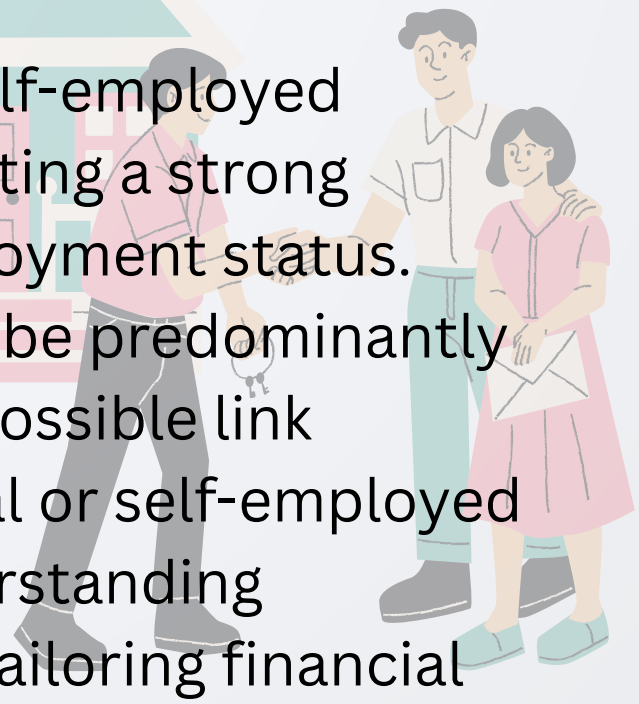
```
pd.crosstab(df['Education'], df['Self_Employed']).plot(
    kind='bar', stacked=True, figsize=(10,6), colormap='tab10'
)
plt.title('Distribution of Self_Employed across Education')
plt.xlabel('Education')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.legend(title='Self_Employed')
plt.show()
```

Data Visualization



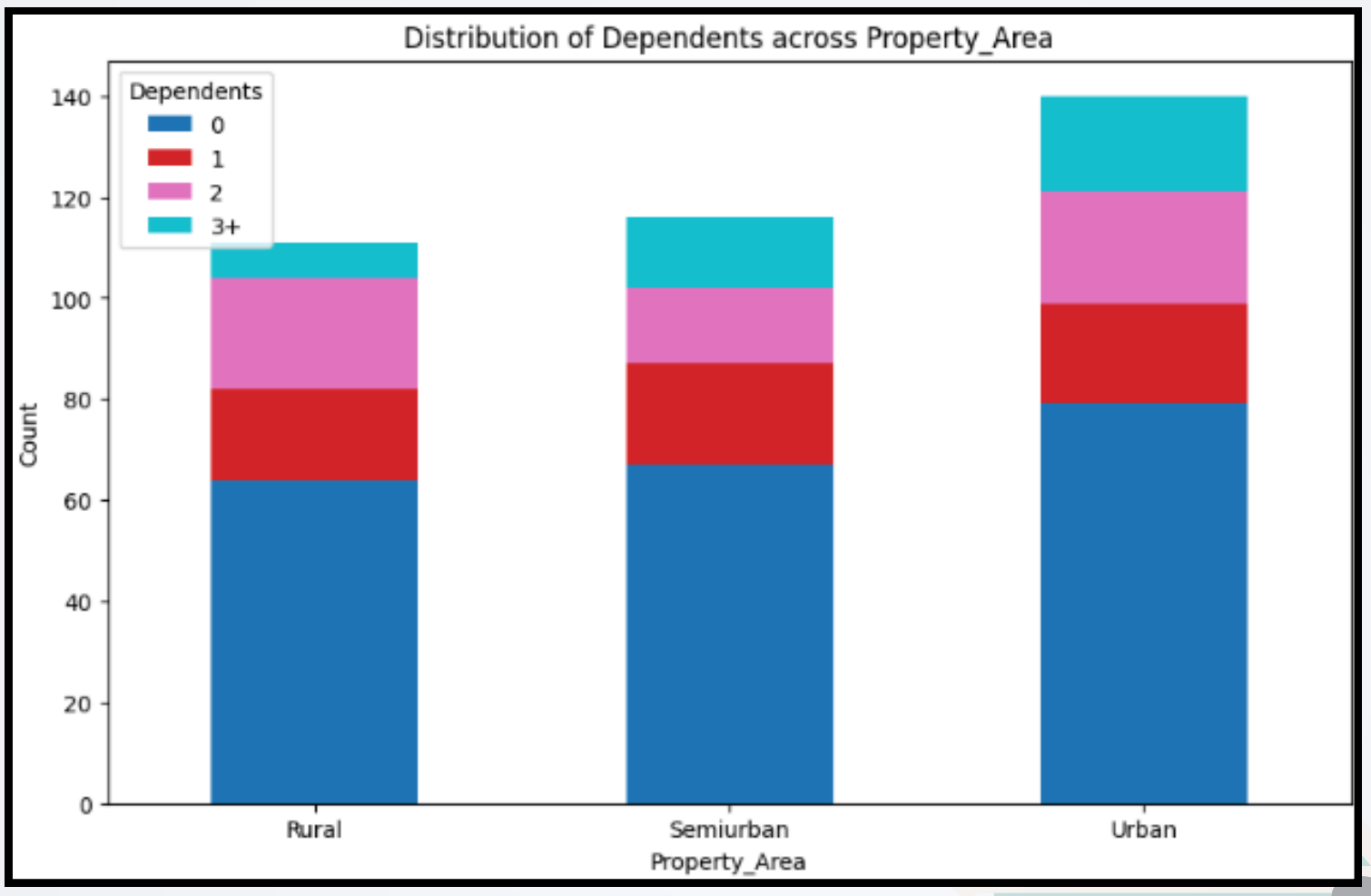
Insights

- The stacked bar chart depicting the distribution of Self_Employed applicants across different education levels indicates that individuals with higher education are more likely to be self-employed.
- The chart shows a visibly larger segment of self-employed individuals within the educated group, suggesting a strong association between education and self-employment status.
- Conversely, less educated applicants tend to be predominantly salaried or not self-employed, highlighting a possible link between higher education and entrepreneurial or self-employed pursuits. This insight could be useful for understanding employment patterns among borrowers and tailoring financial products or risk assessments based on education and employment status.



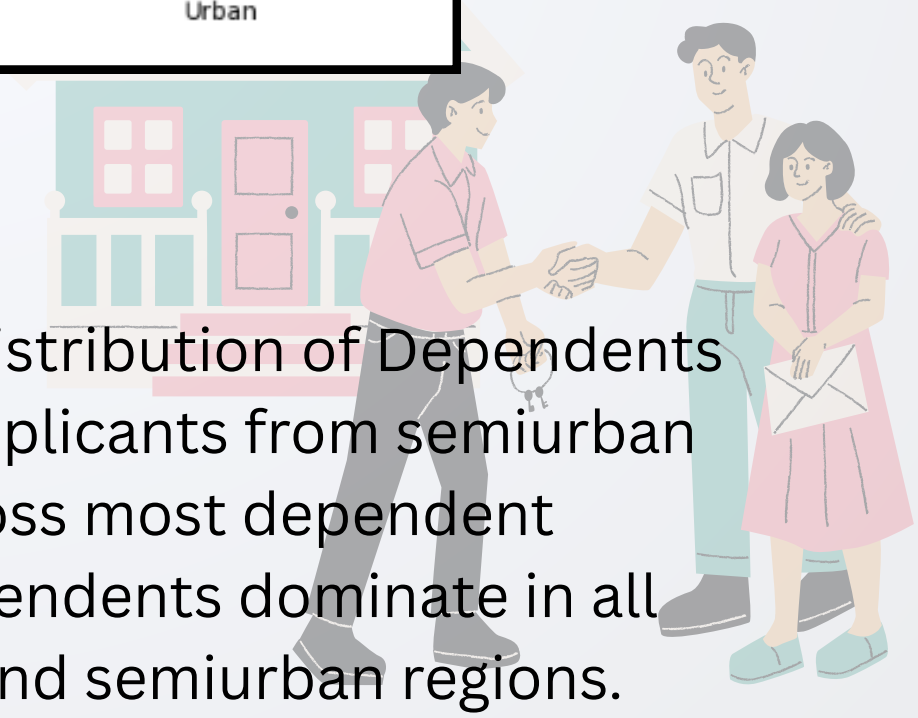
Data Visualization

```
pd.crosstab(df['Property_Area'], df['Dependents']).plot(
    kind='bar', stacked=True, figsize=(10,6), colormap='tab10'
)
plt.title('Distribution of Dependents across Property_Area')
plt.xlabel('Property_Area')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.legend(title='Dependents')
plt.show()
```



Insights

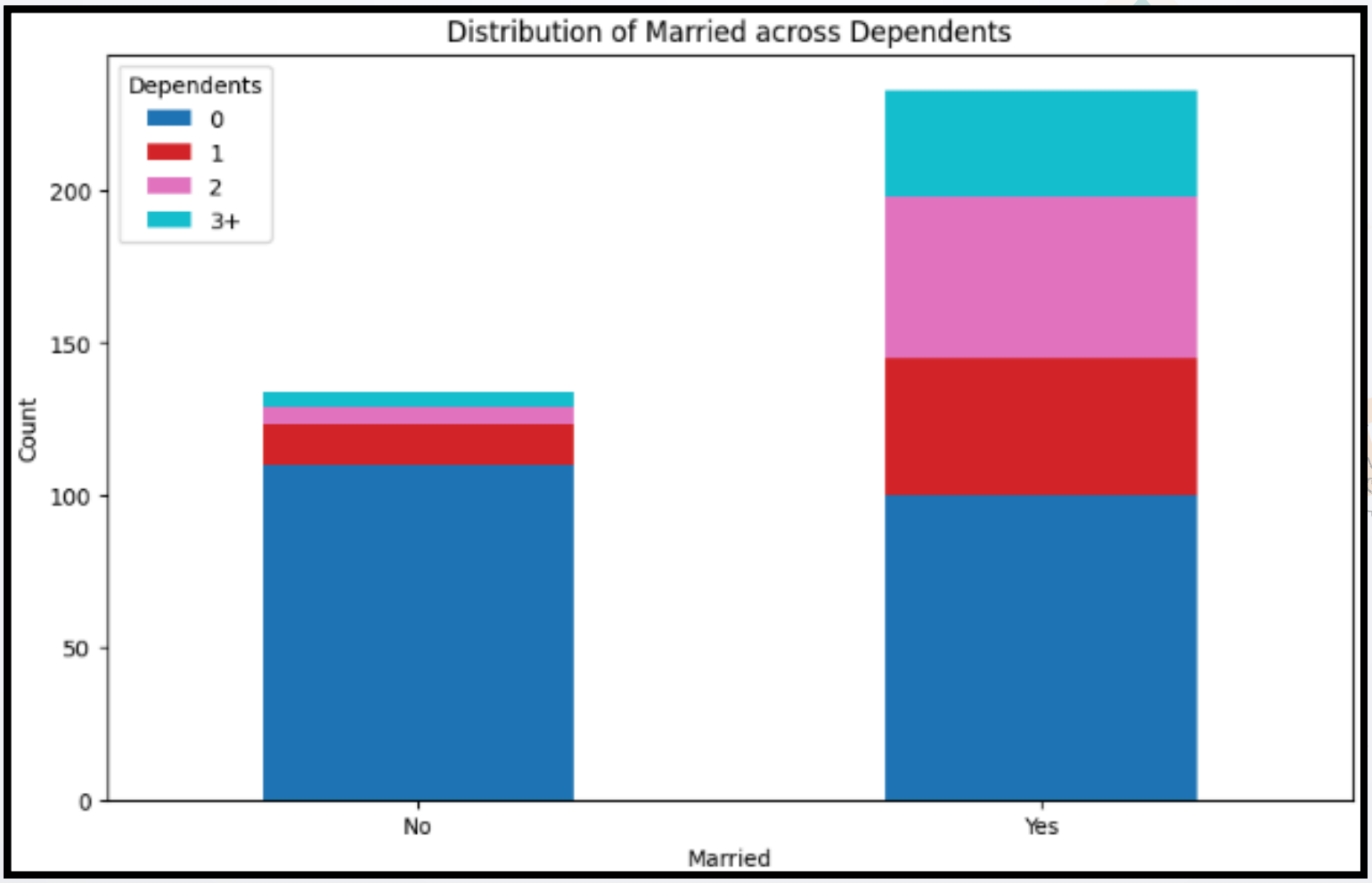
- The stacked bar chart showing the distribution of Dependents across Property_Area reveals that applicants from semiurban areas make up the largest group across most dependent categories. Applicants with zero dependents dominate in all property areas, especially in urban and semiurban regions.



Data Visualization

- Rural areas show a slightly higher proportion of applicants with one or more dependents, suggesting family size may vary with geographic location.
- This insight can assist in tailoring financial products and support services according to regional demographic characteristics

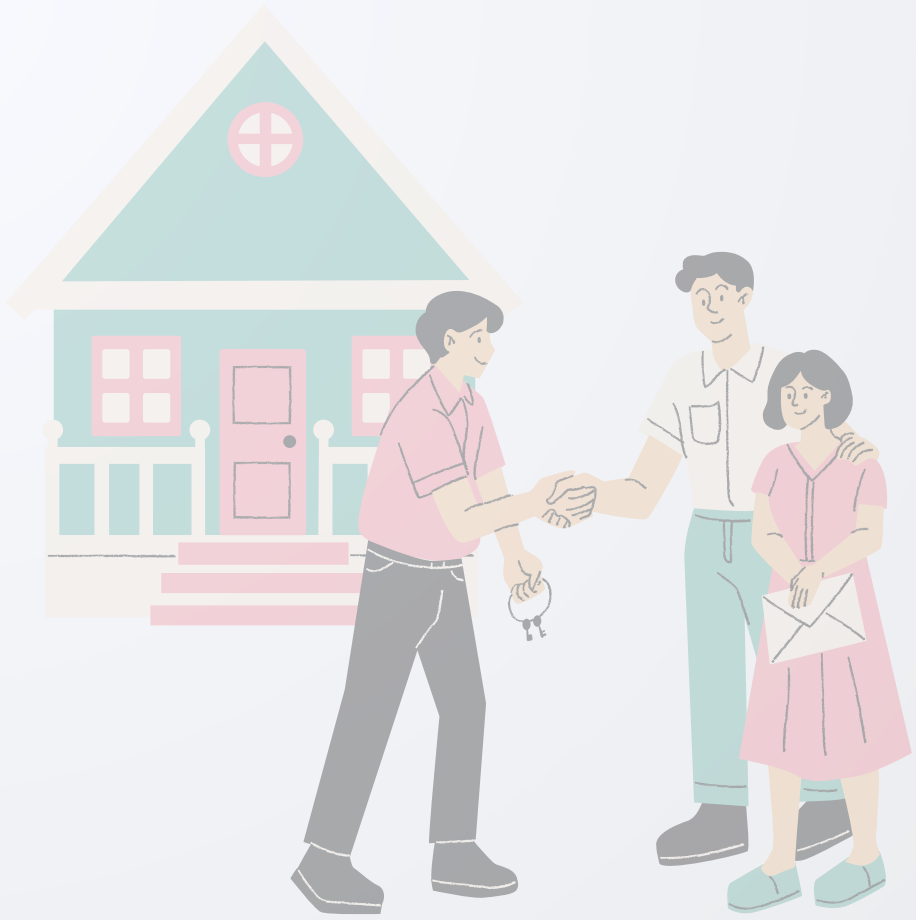
```
pd.crosstab(df['Married'], df['Dependents']).plot(
    kind='bar', stacked=True, figsize=(10,6), colormap='tab10'
)
plt.title('Distribution of Married across Dependents')
plt.xlabel('Married')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.legend(title='Dependents')
plt.show()
```



Data Visualization

Insights

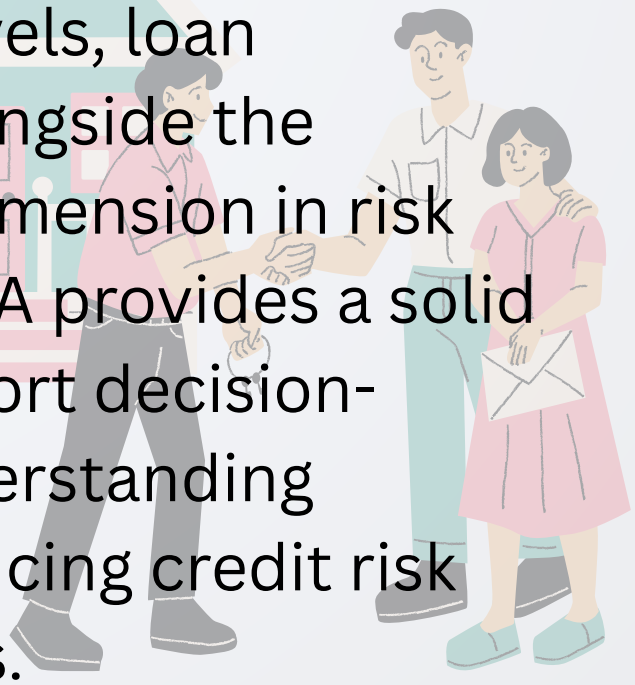
- The stacked bar chart illustrating the distribution of marital status across different numbers of dependents shows that married applicants overwhelmingly dominate across all dependent counts.
- Most married applicants have zero or one dependent, with fewer reporting two or more dependents.
- In contrast, unmarried applicants make up a small proportion across all dependent categories, highlighting that larger family responsibilities are mostly associated with married individuals.
- This insight helps in understanding family dynamics and their potential impact on loan applications and repayment capacity.



Conclusion

The exploratory data analysis of the loan application dataset reveals valuable insights into the profile and behavior of borrowers. The dataset comprises a diverse set of applicant characteristics, including income, employment status, education, and family details, which all play crucial roles in loan approval decisions. Analysis of numeric variables, such as ApplicantIncome and LoanAmount, shows skewed distributions with notable outliers, prompting necessary data treatment through capping to improve modeling accuracy.

Categorical variables like Gender, Marital Status, and Property Area display discernible group patterns that highlight lending demographics and potential market segments. Relationships uncovered through correlation and scatter analyses emphasize the interplay between income levels, loan amounts, and coapplicant contributions, alongside the importance of credit history as a separate dimension in risk assessment. Overall, this comprehensive EDA provides a solid foundation to build predictive models, support decision-making, and tailor financial products by understanding borrower profiles, mitigating bias, and enhancing credit risk evaluations within the loan approval process.



Thanks for reading

for coding part.....