EC 9560 DATA MINING
LAB 02

PRIYATHARSINI S.

2020/E/122

SEMESTER 07

09 OCT 2024

Data Visulaization

```
1 data.head()
```

	id	person_age	person_income	person_home_ownership	person_emp_length	loan_intent	loan_grade	loan_amnt	loan_int_rate	loan_percent_income
0	0	37	35000	RENT	0.0	EDUCATION	B	6000	11.49	0.17
1	1	22	56000	OWN	6.0	MEDICAL	C	4000	13.35	0.07
2	2	29	28800	OWN	8.0	PERSONAL	A	6000	8.90	0.21
3	3	30	70000	RENT	14.0	VENTURE	B	12000	11.11	0.17
4	4	22	60000	RENT	2.0	MEDICAL	A	6000	6.92	0.10

cb_person_default_on_file	cb_person_cred_hist_length	loan_status
N	14	0
N	2	0
N	10	0
N	5	0
N	3	0

Describe feature data types

```
[10] 1 data.dtypes
```

	0
id	int64
person_age	int64
person_income	int64
person_home_ownership	object
person_emp_length	float64
loan_intent	object
loan_grade	object
loan_amnt	int64
loan_int_rate	float64
loan_percent_income	float64
cb_person_default_on_file	object
cb_person_cred_hist_length	int64
loan_status	int64

dtype: object

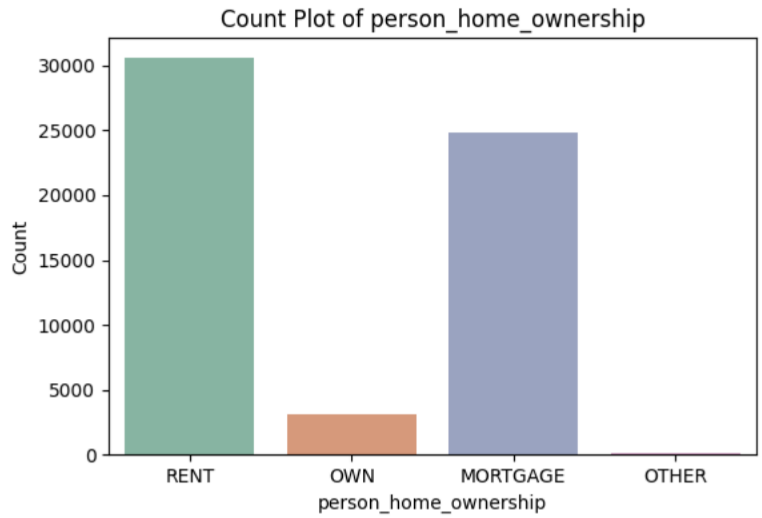
Encoding the feature using Label encoding

```python
[12]

from sklearn.preprocessing import LabelEncoder



# Create DataFrame
df = pd.DataFrame(data)

# Initialize the LabelEncoder
label_encoder = LabelEncoder()

# Columns to be label encoded
categorical_columns = [
    'person_home_ownership',
    'loan_intent',
    'loan_grade',
    'cb_person_default_on_file'
]

# Convert person_emp_length to int
df['person_emp_length'] = pd.to_numeric(df['person_emp_length'],
errors='coerce').fillna(0).astype(int)

# Apply label encoding to categorical columns
for column in categorical_columns:
    df[column] = label_encoder.fit_transform(df[column])

# Display the updated DataFrame
print(df.head())
```

	id	person_age	person_income	person_home_ownership	person_emp_length	loan_intent	loan_grade	loan_amnt	loan_int_rate	loan_percent_income	cb_person_default_on_file	cb_person_cred_hist_length	loan_status
0	0	37	35000	3	0	1	1	6000	11.49	0.17	0	14	0
1	1	22	56000	2	6	3	2	4000	13.35	0.07	0	2	0
2	2	29	28800	2	8	4	0	6000	8.90	0.21	0	10	0
3	3	30	70000	3	14	5	1	12000	11.11	0.17	0	5	0
4	4	22	60000	3	2	3	0	6000	6.92	0.10	0	3	0

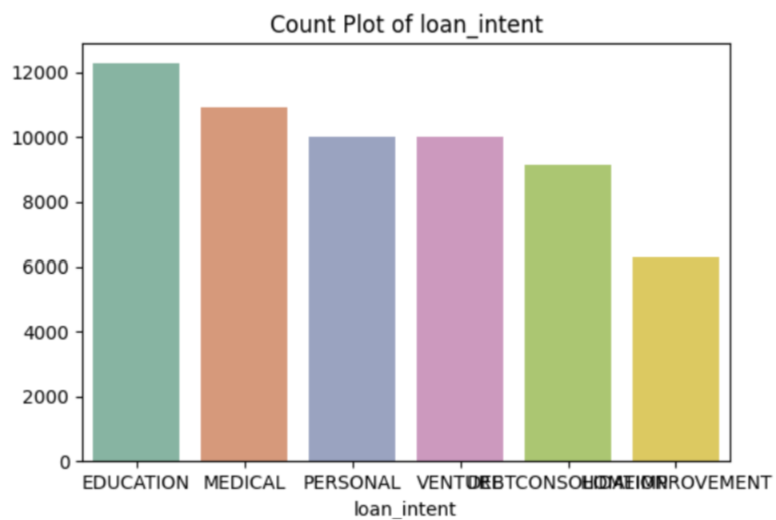
Visualize each feature of the dataset

```python
# List of object-type columns to plot
object_columns = [
    'person_home_ownership',
    'loan_intent',
    'loan_grade',
    'cb_person_default_on_file'
]

# Create count plots for each object column
for i, column in enumerate(object_columns):
    plt.subplot(2, 2, i + 1)  # Adjust subplot layout as nee
    sns.countplot(data=df, x=column, palette='Set2')
    plt.title(f'Count Plot of {column}')
    plt.xlabel(column)
    plt.ylabel('Count')

# Adjust layout
plt.tight_layout()
plt.show()
```



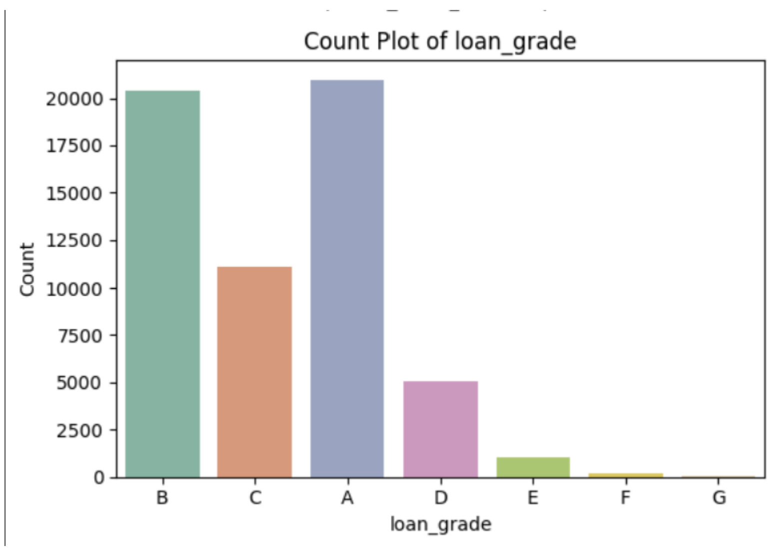
✓ **Count Plot of Person Home Ownership**:

- The majority of individuals in the dataset are renters, with **30,000** people in this category.
- Homeowners make up a significant portion as well, while individuals with a mortgage are less numerous than renters but still represent a noticeable fraction.
- The category labeled "Other" has a minimal count, indicating that most individuals fall into the "Rent," "Own," or "Mortgage" categories.



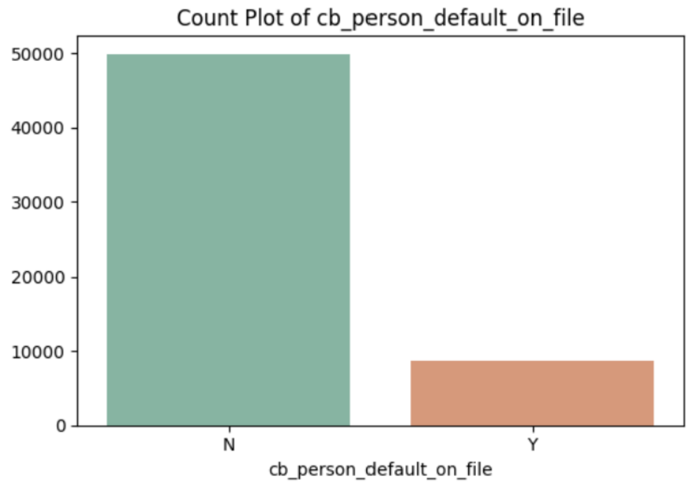
✓ **Count Plot of Loan Intent**:

- This plot shows the distribution of the reasons individuals apply for loans. The most common loan intents are **EDUCATION**, **MEDICAL**, and **PERSONAL**, each with around **10,000 to 12,000** applications.
- Other categories such as **DEBT CONSOLIDATION** and **HOME IMPROVEMENT** have lower counts compared to education and medical intents.
- The loan intent for **VENTURE** has a moderate count, while **OTHER** categories have relatively low numbers.



✓ **Count Plot of Loan Grade**:

- The distribution of loan grades shows that grades **B** and **C** are the most common, with **over 20,000** for grade B and **around 15,000** for grade C.
- Grade **A** has a lower count, while grades **D**, **E**, **F**, and **G** are significantly less common, indicating a steeper decline in count as the loan grade decreases.

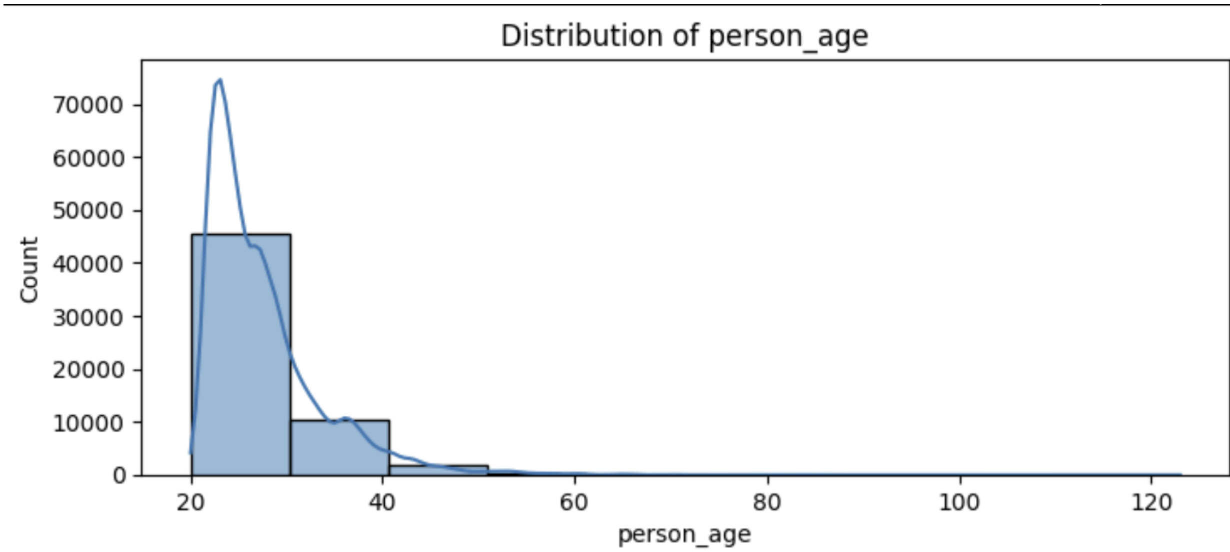


✓ **Count Plot of Credit Bureau Person Default on File**:

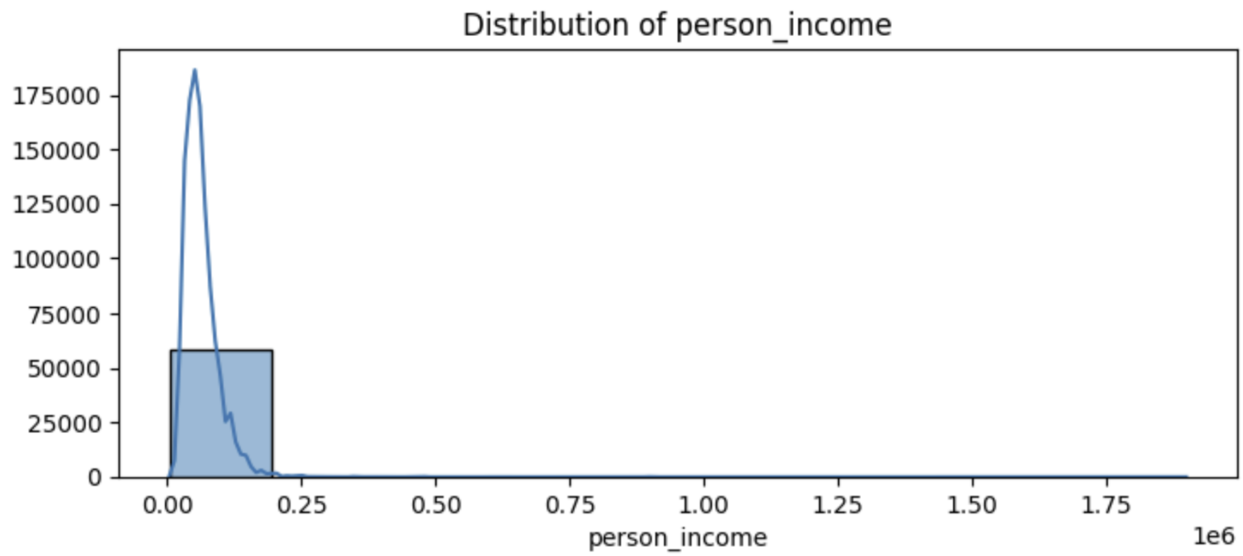
- The majority of individuals in the dataset do not have a default on their credit file (**N**), with **over 50,000** instances.
- Only a small fraction of individuals has a default on file (**Y**), indicating that defaults are relatively uncommon in this dataset.

[51]

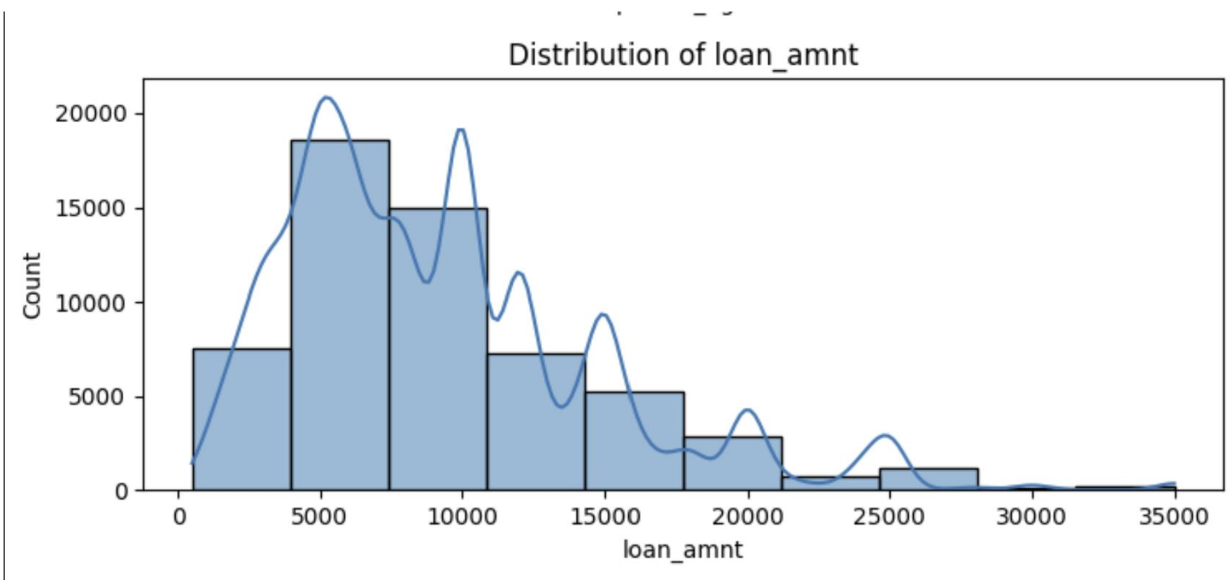
```python
# List of integer columns to plot
int_columns = [
    'person_age',
    'person_income',
    'loan_amnt',
    'cb_person_cred_hist_length',
    'person_emp_length',
    'loan_int_rate'
]

# Set the plot size
plt.figure(figsize=(15, 10))

# Create count plots (histograms) for each integer column
for i, int_column in enumerate(int_columns):
    plt.subplot(3, 2, i + 1)  # Adjust subplot layout as needed
    sns.histplot(df[int_column], bins=10, kde=True, palette='Set2')  # Using histplot for count visualization
    plt.title(f'Distribution of {int_column}')
    plt.xlabel(int_column)
    plt.ylabel('Count')

# Adjust layout
plt.tight_layout()
plt.show()
```



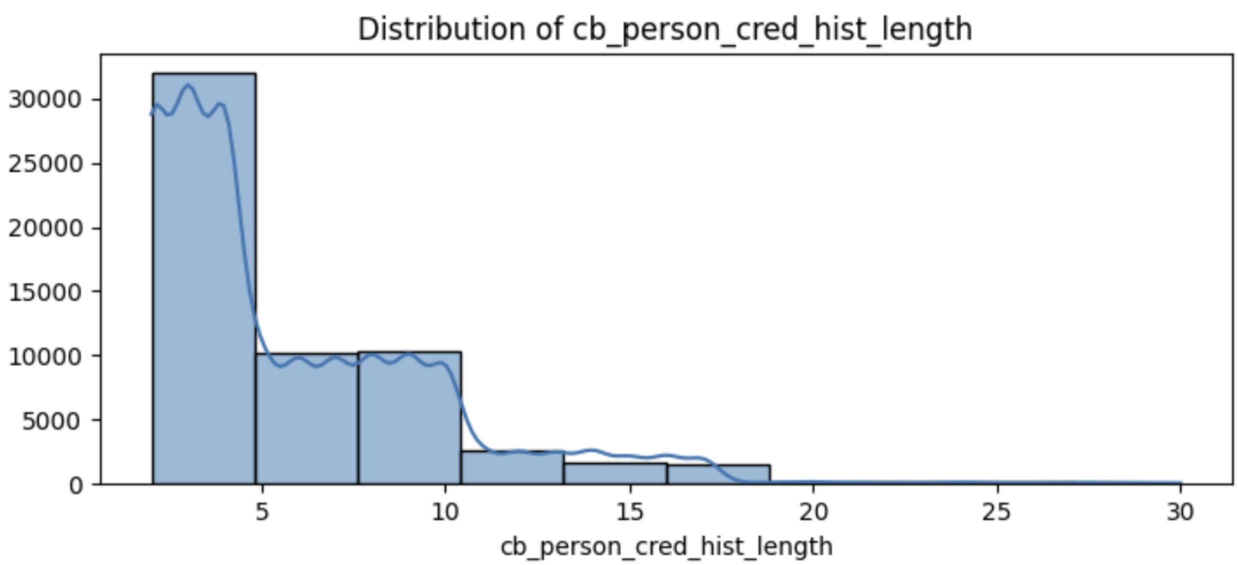
- The distribution is right-skewed, with a higher concentration of individuals in the younger age range (20-40 years), indicating that the dataset may have more younger individuals compared to older ones.



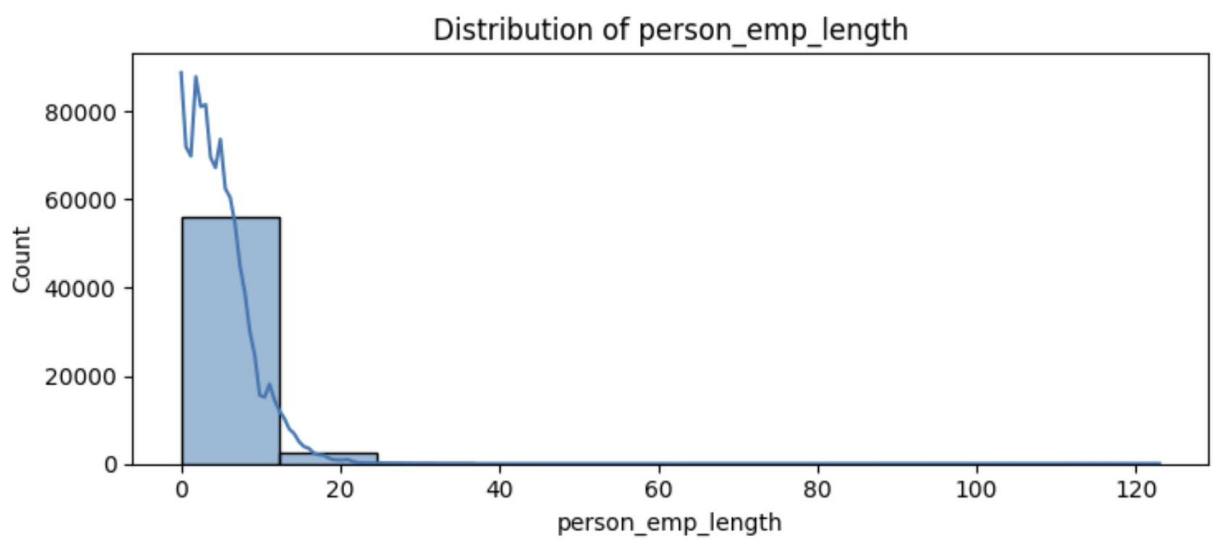
- This plot also shows a right-skewed distribution, with most individuals earning lower incomes. The significant peak around zero suggests that many individuals may have reported incomes near the lower end of the scale, with few high-income outliers.



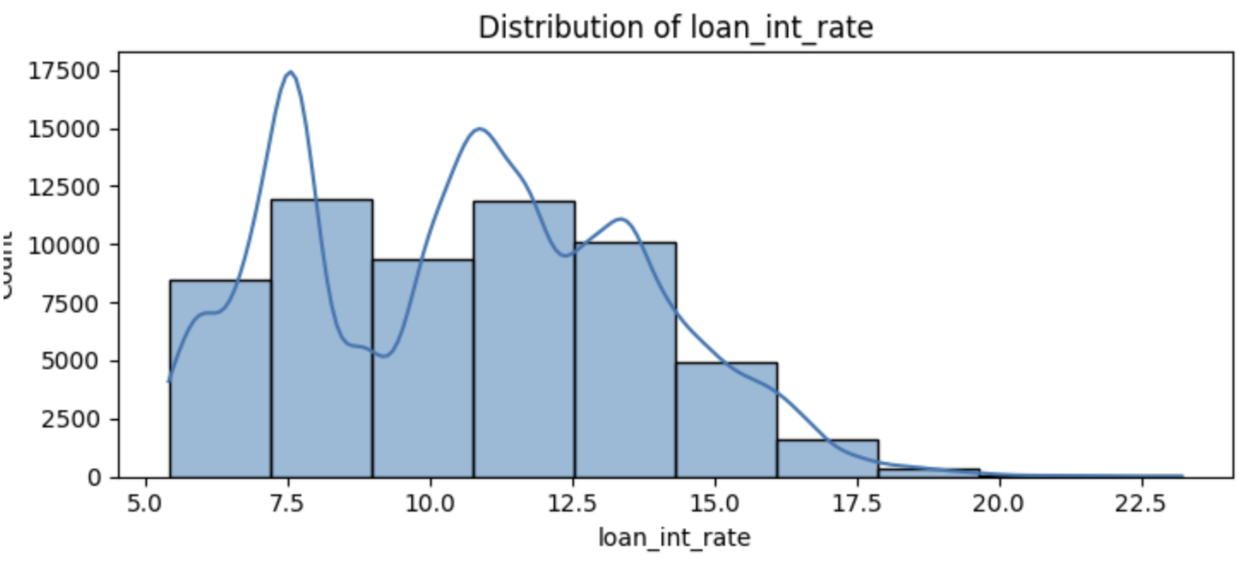
- The distribution indicates that smaller loan amounts are more common, with a gradual decrease in frequency as the loan amount increases. This trend suggests that borrowers typically request lower loans, with fewer individuals taking out larger loans.



- Credit History Length the distribution appears to be heavily right-skewed, indicating that most individuals have a shorter credit history, with fewer individuals having longer credit histories.



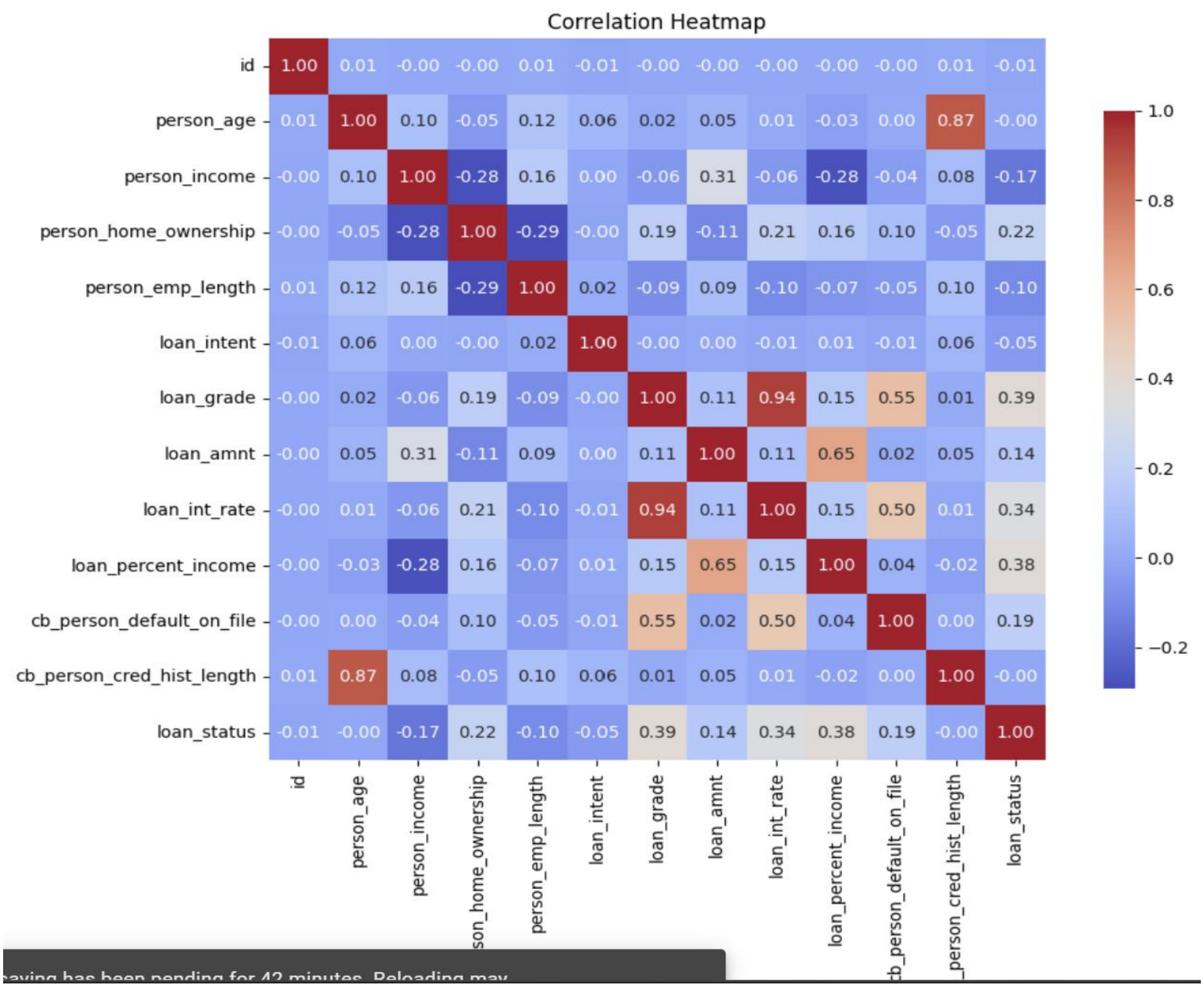
- Person Employment Length this distribution shows a significant concentration of individuals with shorter employment lengths, particularly those with less than 20 years of experience. The plot suggests that most individuals in the dataset may be relatively new to the workforce.



- Loan Interest Rate the distribution of interest rates shows a slight right-skew, with most rates concentrated around the lower end of the scale. This indicates that many loans are issued at lower interest rates, with fewer loans at higher rates.

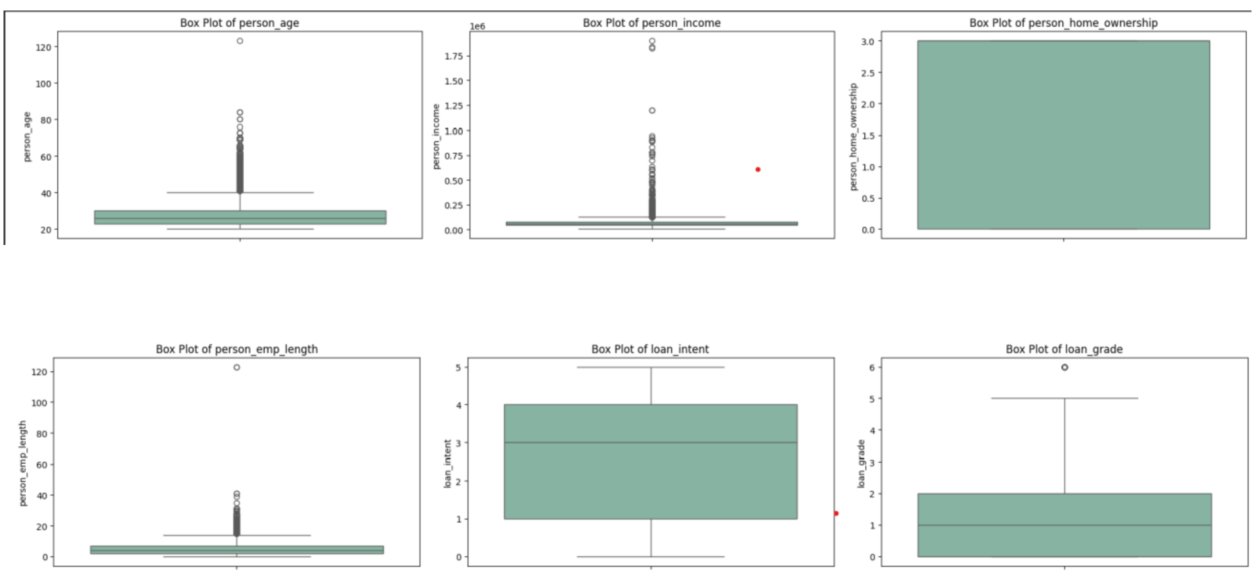
Visualize the Correlation between features and target using correlations coefficient matrix

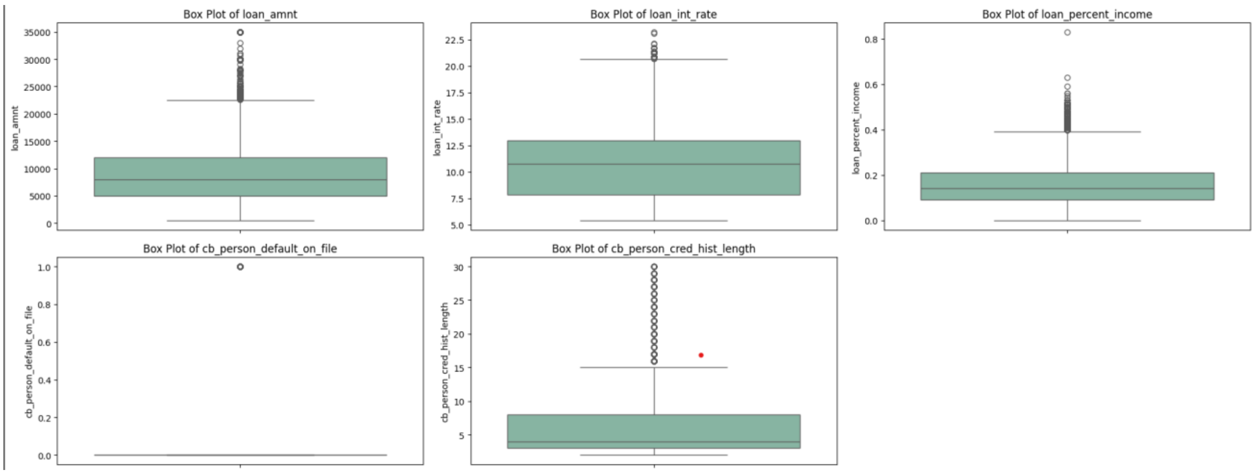
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate the correlation matrix
correlation_matrix = df.corr()

# Set up the matplotlib figure
plt.figure(figsize=(12, 8))

# Generate a heatmap
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap='coolwarm',
square=True, cbar_kws={"shrink": .8})

# Set title and labels
plt.title('Correlation Heatmap')
plt.show()
```



Checking Outliers

```python
# Select numeric columns, excluding the first (id) and last (loan_status)
numeric_columns = df.select_dtypes(include=['int64', 'float64']).columns.tolist()[1:-1]  # Exclude first and last

# Set the plot size
plt.figure(figsize=(20, 15))

# Calculate the number of rows and columns for subplots
n = len(numeric_columns)
ncols = 3  # Set the number of columns
nrows = (n + ncols - 1) // ncols  # Calculate the number of rows needed

# Create box plots for each selected numeric column
for i, column in enumerate(numeric_columns):
    plt.subplot(nrows, ncols, i + 1)  # Adjust subplot layout dynamically
    sns.boxplot(data=df, y=column, palette='Set2')
    plt.title(f'Box Plot of {column}')
    plt.ylabel(column)

# Adjust layout
plt.tight_layout()
plt.show()
```





- **Person Age**: The distribution appears relatively normal with some outliers at higher ages.
- **Person Income**: The income distribution shows significant outliers, indicating a few individuals with very high incomes.
- **Person Employment Length**: There are some outliers present, suggesting variability in employment history length.
- **Loan Amount**: This feature displays a wider range of values with several outliers, particularly on the higher end.
- **Loan Intent**: The box plot seems to indicate a concentration of loans around certain intents without significant outliers.
- **Loan Grade**: This feature has a narrow range, suggesting limited variation among loan grades.
- **Loan Interest Rate**: The interest rates show a more balanced distribution, with a few higher outliers.
- **Loan Percent Income**: Most values cluster around lower percentages, with a few outliers indicating higher loan percentages compared to income.
- **Credit History Length**: The distribution indicates a range of credit history lengths, with outliers present, suggesting variability in individuals' credit histories.

Summery

```
Summery = df.describe()
Summery
```

	id	person_age	person_income	person_home_ownership	person_emp_length	loan_intent	loan_grade	loan_amnt	loan_int_rate	loan_percent_income
count	58645.000000	58645.000000	5.864500e+04	58645.000000	58645.000000	58645.000000	58645.000000	58645.000000	58645.000000	58645.000000
mean	29322.000000	27.550857	6.404617e+04	1.673578	4.701015	2.519430	1.066638	9217.556518	10.677874	0.159238
std	16929.497605	6.033216	3.793111e+04	1.452534	3.959784	1.722896	1.046181	5563.807384	3.034697	0.091692
min	0.000000	20.000000	4.200000e+03	0.000000	0.000000	0.000000	0.000000	500.000000	5.420000	0.000000
25%	14661.000000	23.000000	4.200000e+04	0.000000	2.000000	1.000000	0.000000	5000.000000	7.880000	0.090000
50%	29322.000000	26.000000	5.800000e+04	3.000000	4.000000	3.000000	1.000000	8000.000000	10.750000	0.140000
75%	43983.000000	30.000000	7.560000e+04	3.000000	7.000000	4.000000	2.000000	12000.000000	12.990000	0.210000
max	58644.000000	123.000000	1.900000e+06	3.000000	123.000000	5.000000	6.000000	35000.000000	23.220000	0.830000

cb_person_default_on_file	cb_person_cred_hist_length	loan_status
58645.000000	58645.000000	58645.000000
0.148384	5.813556	0.142382
0.355484	4.029196	0.349445
0.000000	2.000000	0.000000
0.000000	3.000000	0.000000
0.000000	4.000000	0.000000
0.000000	8.000000	0.000000
1.000000	30.000000	1.000000