Import Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
```

```
credit_card=pd.read_csv("C:/Users/Vyankatesh Pandit/Downloads/creditcard.csv",low_memory=False)    #Read CSV
```

```
credit_card.head()
```

	NPA Status	RevolvingUtilizationOfUnsecuredLines	age	Gender	Region	MonthlyIncome	Re
0	1.0	0.766127	45.0	Male	South	9120.0	
1	0.0	0.957151	40.0	Female	South	2600.0	
2	0.0	0.658180	38.0	Female	South	3042.0	
3	0.0	0.233810	30.0	Female	South	3300.0	
4	0.0	0.907239	49.0	Male	South	63588.0	

```
credit_card.isnull().sum()
```

```
NPA Status                                  2
RevolvingUtilizationOfUnsecuredLines        2
age                                         2
Gender                                      2
Region                                      2
MonthlyIncome                           29733
Rented_OwnHouse                             2
Occupation                                  2
Education                                   2
NumberOfTime30-59DaysPastDueNotWorse        2
DebtRatio                                   2
MonthlyIncome.1                         29733
NumberOfOpenCreditLinesAndLoans             2
NumberOfTimes90DaysLate                     2
NumberRealEstateLoansOrLines                2
NumberOfTime60-89DaysPastDueNotWorse        2
NumberOfDependents                       3924
Good_Bad                                    2
dtype: int64
```

```
credit_card.drop(columns=["MonthlyIncome.1"],inplace=True)
```

```
credit_card.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150002 entries, 0 to 150001
Data columns (total 17 columns):
 #   Column                                Non-Null Count   Dtype
---  ------                                --------------   -----
 0   NPA Status                            150000 non-null  float64
 1   RevolvingUtilizationOfUnsecuredLines  150000 non-null  float64
 2   age                                   150000 non-null  float64
 3   Gender                                150000 non-null  object
 4   Region                                150000 non-null  object
 5   MonthlyIncome                         120269 non-null  float64
 6   Rented_OwnHouse                       150000 non-null  object
 7   Occupation                            150000 non-null  object
 8   Education                             150000 non-null  object
 9   NumberOfTime30-59DaysPastDueNotWorse  150000 non-null  float64
 10  DebtRatio                             150000 non-null  float64
 11  NumberOfOpenCreditLinesAndLoans       150000 non-null  float64
 12  NumberOfTimes90DaysLate               150000 non-null  float64
 13  NumberRealEstateLoansOrLines          150000 non-null  float64
 14  NumberOfTime60-89DaysPastDueNotWorse  150000 non-null  float64
 15  NumberOfDependents                    146078 non-null  object
 16  Good_Bad                              150000 non-null  object
dtypes: float64(10), object(7)
memory usage: 19.5+ MB
```

```
credit_card.isnull().sum()
```

```
NPA Status                                 2
RevolvingUtilizationOfUnsecuredLines       2
age                                        2
Gender                                     2
Region                                     2
MonthlyIncome                          29733
Rented_OwnHouse                            2
Occupation                                 2
Education                                  2
NumberOfTime30-59DaysPastDueNotWorse       2
DebtRatio                                  2
NumberOfOpenCreditLinesAndLoans            2
NumberOfTimes90DaysLate                    2
NumberRealEstateLoansOrLines               2
NumberOfTime60-89DaysPastDueNotWorse       2
NumberOfDependents                      3924
Good_Bad                                   2
dtype: int64
```

NPA STATUS

```
credit_card['NPA Status'].mode()
credit_card['NPA Status'].fillna(credit_card['NPA Status'].mode()[0], inplace=True)
```

```
credit_card['RevolvingUtilizationOfUnsecuredLines'].median()
credit_card['RevolvingUtilizationOfUnsecuredLines'].fillna(credit_card['RevolvingUtilizationOfUnsecuredLines'].median(), inplace=True)
```

AGE

```
credit_card['age'].mean()
credit_card['age'].fillna(credit_card['age'].mean(), inplace=True)
```

Gender

```
credit_card['Gender'].mode()
credit_card['Gender'].fillna(credit_card['Gender'].mode()[0], inplace=True)
```

```
label_encoder = LabelEncoder()
credit_card['Gender_num'] = label_encoder.fit_transform(credit_card['Gender'])    #Convert text to Numerical Data
```

```
credit_card['Gender_num'].value_counts()
```

```
Gender_num
1    92306
0    57696
Name: count, dtype: int64
```

```
#credit_card['Gender'] = label_encoder.inverse_transform(credit_card['Gender_LabelEncoded'])
```

```
#credit_card['Gender'].value_counts()
```

Region

```
credit_card['Region'].mode()
credit_card['Region'].fillna(credit_card['Region'].mode()[0], inplace=True)
```

```
credit_card['Region_num']=label_encoder.fit_transform(credit_card['Region'])
```

```
credit_card['Region_num'].value_counts()
```

```
Region_num
0    43958
2    34099
4    27899
3    23495
```

```
1    20551
Name: count, dtype: int64
```

Monthly Income

```
credit_card['MonthlyIncome'].mean()
credit_card['MonthlyIncome'].fillna(credit_card['MonthlyIncome'].mean(), inplace=True)
```

Occupation

```
credit_card['Occupation'].value_counts()
credit_card['Occupation'].mode()
credit_card['Occupation'].fillna(credit_card['Occupation'].mode()[0], inplace=True)
```

```
credit_card['Occupation_num']=label_encoder.fit_transform(credit_card['Occupation'])
```

```
credit_card['Occupation_num'].value_counts()
```

```
Occupation_num
4    64118
0    41113
3    16274
1    15164
2    13333
Name: count, dtype: int64
```

Reneted House

```
credit_card['Rented_OwnHouse'].value_counts()
credit_card['Rented_OwnHouse'].mode()
credit_card['Rented_OwnHouse'].fillna(credit_card['Rented_OwnHouse'].mode()[0], inplace=True)
```

```
credit_card['Rented_OwnHouse_num']=label_encoder.fit_transform(credit_card['Rented_OwnHouse'])
```

```
credit_card['Rented_OwnHouse_num'].value_counts()
```

```
Rented_OwnHouse_num
0    85955
1    64047
Name: count, dtype: int64
```

Education

```
credit_card['Education'].value_counts()
credit_card['Education'].mode()
credit_card['Education'].fillna(credit_card['Education'].mode()[0], inplace=True)
```

```
credit_card['Education_num']=label_encoder.fit_transform(credit_card['Education'])
```

```
credit_card['Education_num'].value_counts()
```

```
Education_num
4    50922
0    39755
3    37214
1    15810
2     6301
Name: count, dtype: int64
```

```
credit_card['NumberOfTime30-59DaysPastDueNotWorse'].value_counts()
credit_card['NumberOfTime30-59DaysPastDueNotWorse'].mode()
credit_card['NumberOfTime30-59DaysPastDueNotWorse'].fillna(credit_card['NumberOfTime30-59DaysPastDueNotWorse'].mode()[0], inplace=True)
```

```
credit_card['DebtRatio'].mode()
credit_card['DebtRatio'].fillna(credit_card['DebtRatio'].mode()[0], inplace=True)
```

```
credit_card['NumberOfOpenCreditLinesAndLoans'].value_counts()
credit_card['NumberOfOpenCreditLinesAndLoans'].median()
credit_card['NumberOfOpenCreditLinesAndLoans'].fillna(credit_card['NumberOfOpenCreditLinesAndLoans'].median(), inplace=True)
```

```
credit_card['NumberOfTimes90DaysLate'].value_counts()
credit_card['NumberOfTimes90DaysLate'].median()
credit_card['NumberOfTimes90DaysLate'].fillna(credit_card['NumberOfTimes90DaysLate'].median(), inplace=True)
```

```
credit_card['NumberRealEstateLoansOrLines'].value_counts()
credit_card['NumberRealEstateLoansOrLines'].mode()
credit_card['NumberRealEstateLoansOrLines'].fillna(credit_card['NumberRealEstateLoansOrLines'].mode()[0], inplace=True)
```

```
credit_card['NumberOfTime60-89DaysPastDueNotWorse'].value_counts()
credit_card['NumberOfTime60-89DaysPastDueNotWorse'].mode()
credit_card['NumberOfTime60-89DaysPastDueNotWorse'].fillna(credit_card['NumberOfTime60-89DaysPastDueNotWorse'].mode()[0], inplace=True)
```

```
credit_card['NumberOfDependents'].value_counts()
credit_card['NumberOfDependents'].mode()
credit_card['NumberOfDependents'].fillna(credit_card['NumberOfDependents'].mode()[0], inplace=True)
```

```
condition = (credit_card['NumberOfDependents'] =='Good')
credit_card = credit_card.drop(credit_card[condition].index)
```

```
condition1 = (credit_card['NumberOfDependents'] =='Bad')
credit_card = credit_card.drop(credit_card[condition1].index)
```

```
credit_card['NumberOfDependents'].value_counts()
```

```
NumberOfDependents
0     90826
1     26316
2     19522
3      9483
4      2862
5       746
6       158
7        51
8        24
10        5
9         5
20        1
13        1
Name: count, dtype: int64
```

```
credit_card['Good_Bad'].value_counts()
credit_card['Good_Bad'].mode()
credit_card['Good_Bad'].fillna(credit_card['Good_Bad'].mode()[0], inplace=True)
```

```
credit_card['Good_Bad_num']=label_encoder.fit_transform(credit_card['Good_Bad'])
```

```
credit_card['Good_Bad_num'].value_counts()
```

```
Good_Bad_num
1    139974
0     10026
Name: count, dtype: int64
```

```
credit_card.isnull().sum()
```

```
NPA Status                              0
RevolvingUtilizationOfUnsecuredLines    0
age                                     0
Gender                                  0
Region                                  0
MonthlyIncome                           0
Rented_OwnHouse                         0
Occupation                              0
Education                               0
NumberOfTime30-59DaysPastDueNotWorse    0
DebtRatio                               0
NumberOfOpenCreditLinesAndLoans         0
NumberOfTimes90DaysLate                 0
```

```
NumberRealEstateLoansOrLines            0
NumberOfTime60-89DaysPastDueNotWorse    0
NumberOfDependents                      0
Good_Bad                                0
Gender_num                              0
Region_num                              0
Occupation_num                          0
Rented_OwnHouse_num                     0
Education_num                           0
Good_Bad_num                            0
dtype: int64
```

```
credit_card.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 150000 entries, 0 to 149999
Data columns (total 23 columns):
 #   Column                                Non-Null Count   Dtype  
---  ------                                --------------   -----  
 0   NPA Status                            150000 non-null  float64
 1   RevolvingUtilizationOfUnsecuredLines  150000 non-null  float64
 2   age                                   150000 non-null  float64
 3   Gender                                150000 non-null  object 
 4   Region                                150000 non-null  object 
 5   MonthlyIncome                         150000 non-null  float64
 6   Rented_OwnHouse                       150000 non-null  object 
 7   Occupation                            150000 non-null  object 
 8   Education                             150000 non-null  object 
 9   NumberOfTime30-59DaysPastDueNotWorse  150000 non-null  float64
 10  DebtRatio                             150000 non-null  float64
 11  NumberOfOpenCreditLinesAndLoans       150000 non-null  float64
 12  NumberOfTimes90DaysLate               150000 non-null  float64
 13  NumberRealEstateLoansOrLines          150000 non-null  float64
 14  NumberOfTime60-89DaysPastDueNotWorse  150000 non-null  float64
 15  NumberOfDependents                    150000 non-null  object 
 16  Good_Bad                              150000 non-null  object 
 17  Gender_num                            150000 non-null  int32  
 18  Region_num                            150000 non-null  int32  
 19  Occupation_num                        150000 non-null  int32  
 20  Rented_OwnHouse_num                   150000 non-null  int32  
 21  Education_num                         150000 non-null  int32  
 22  Good_Bad_num                          150000 non-null  int32  
dtypes: float64(10), int32(6), object(7)
memory usage: 24.0+ MB
```

```
credit_card.duplicated().sum()  #Check Duplicate values and drop them
```

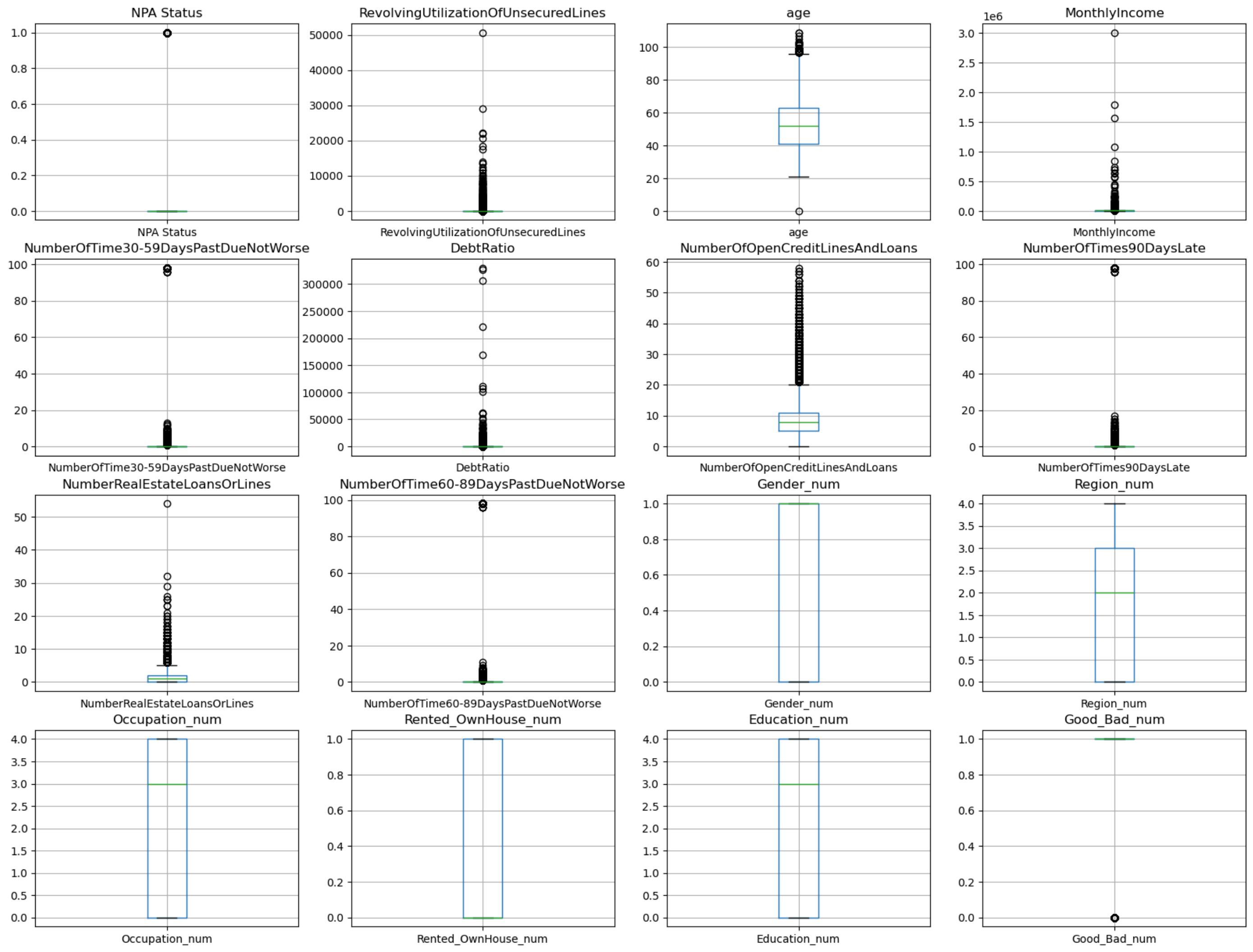
```
25
```

```
credit_card.drop_duplicates(inplace=True)
```

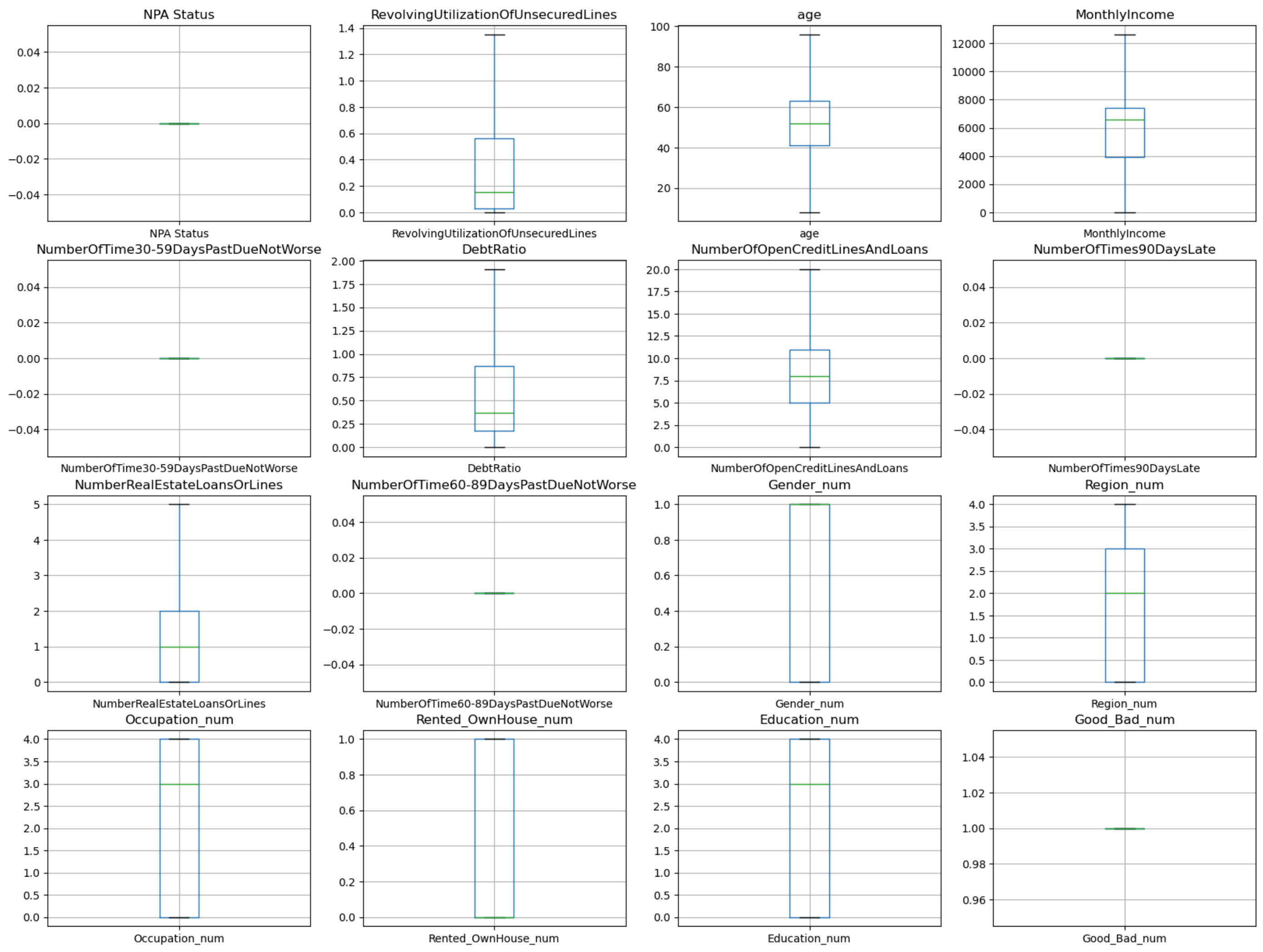
```
credit_card.shape
```

```
(149975, 23)
```

```
plt.figure(figsize=(20, 15))
for i, column in enumerate(credit_card.select_dtypes(include=['float64', 'int32']).columns, 1):
    plt.subplot(4, 4, i)
    credit_card.boxplot(column=[column])
    plt.title(column)
plt.show()
```



```python
numeric_columns = credit_card.select_dtypes(include=['float64','int32']).columns
for column in numeric_columns:
    Q1 = credit_card[column].quantile(0.25)
    Q3 = credit_card[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    credit_card[column] = credit_card[column].apply(lambda x: lower_bound if x < lower_bound else x)
    credit_card[column] = credit_card[column].apply(lambda x: upper_bound if x > upper_bound else x)
plt.figure(figsize=(20, 15))
for i, column in enumerate(numeric_columns, 1):
    plt.subplot(4, 4, i)
    credit_card.boxplot(column=[column])
    plt.title(column)
plt.show()
```



```
#Save cleaned data in new CSV FIle
credit_card.to_csv('C:/Users/Vyankatesh Pandit/Downloads/cleaned_creditcard_data.csv', index=False)

```

```
Start coding or generate with AI.
```

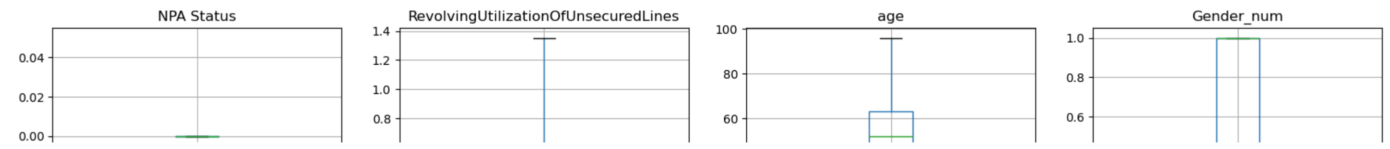
```
columns_to_keep = ['NPA Status', 'RevolvingUtilizationOfUnsecuredLines', 'age', 'Gender_num','Region_num', 'MonthlyIncome', 'Rented_OwnHc
                   'Education_num', 'NumberOfTime30-59DaysPastDueNotWorse', 'DebtRatio', 'NumberOfOpenCreditLinesAndLoans','NumberOfTimes
                     'NumberOfTime60-89DaysPastDueNotWorse','Good_Bad_num']
```

```
new_credit_card =credit_card[columns_to_keep]
```

```
new_credit_card.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 149975 entries, 0 to 149999
Data columns (total 16 columns):
 #   Column                                Non-Null Count   Dtype  
---  ------                                --------------   -----  
 0   NPA Status                            149975 non-null  float64
 1   RevolvingUtilizationOfUnsecuredLines  149975 non-null  float64
 2   age                                   149975 non-null  float64
 3   Gender_num                            149975 non-null  int64  
 4   Region_num                            149975 non-null  int64  
 5   MonthlyIncome                         149975 non-null  float64
 6   Rented_OwnHouse_num                   149975 non-null  int64  
 7   Occupation_num                        149975 non-null  int64  
 8   Education_num                         149975 non-null  int64  
 9   NumberOfTime30-59DaysPastDueNotWorse  149975 non-null  float64
 10  DebtRatio                             149975 non-null  float64
 11  NumberOfOpenCreditLinesAndLoans       149975 non-null  float64
 12  NumberOfTimes90DaysLate               149975 non-null  float64
 13  NumberRealEstateLoansOrLines          149975 non-null  float64
 14  NumberOfTime60-89DaysPastDueNotWorse  149975 non-null  float64
 15  Good_Bad_num                          149975 non-null  float64
dtypes: float64(11), int64(5)
memory usage: 19.5 MB
```

```
plt.figure(figsize=(20, 15))
for i, column in enumerate(new_credit_card.columns, 1):
    plt.subplot(4, 4, i)
    new_credit_card.boxplot(column=[column])
    plt.title(column)
plt.show()
```



```
new_credit_card['Gender'] = label_encoder.inverse_transform(new_credit_card['Gender_num'])
```

```
C:\Users\Vyankatesh Pandit\AppData\Local\Temp\ipykernel_2376\4133239179.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  new_credit_card['Gender'] = label_encoder.inverse_transform(new_credit_card['Gender_num'])
```

```
new_credit_card['Gender'].value_counts()
```

```
Gender
Good    92294
Bad     57681
Name: count, dtype: int64
```