Loan_Data_Exploration_Part2

September 7, 2022

1 Effects of Borrower Characteristics on Loan Repayment

1.1 Investigation Overview

In this investigation, I wanted to look at the characteristics of Loan Borrowers that could be used to predict their loan repayment behaviour. The main focus was on: > - Home ownership > - Monthly Income > - Credit Score > - Borrower State > - Employment Status

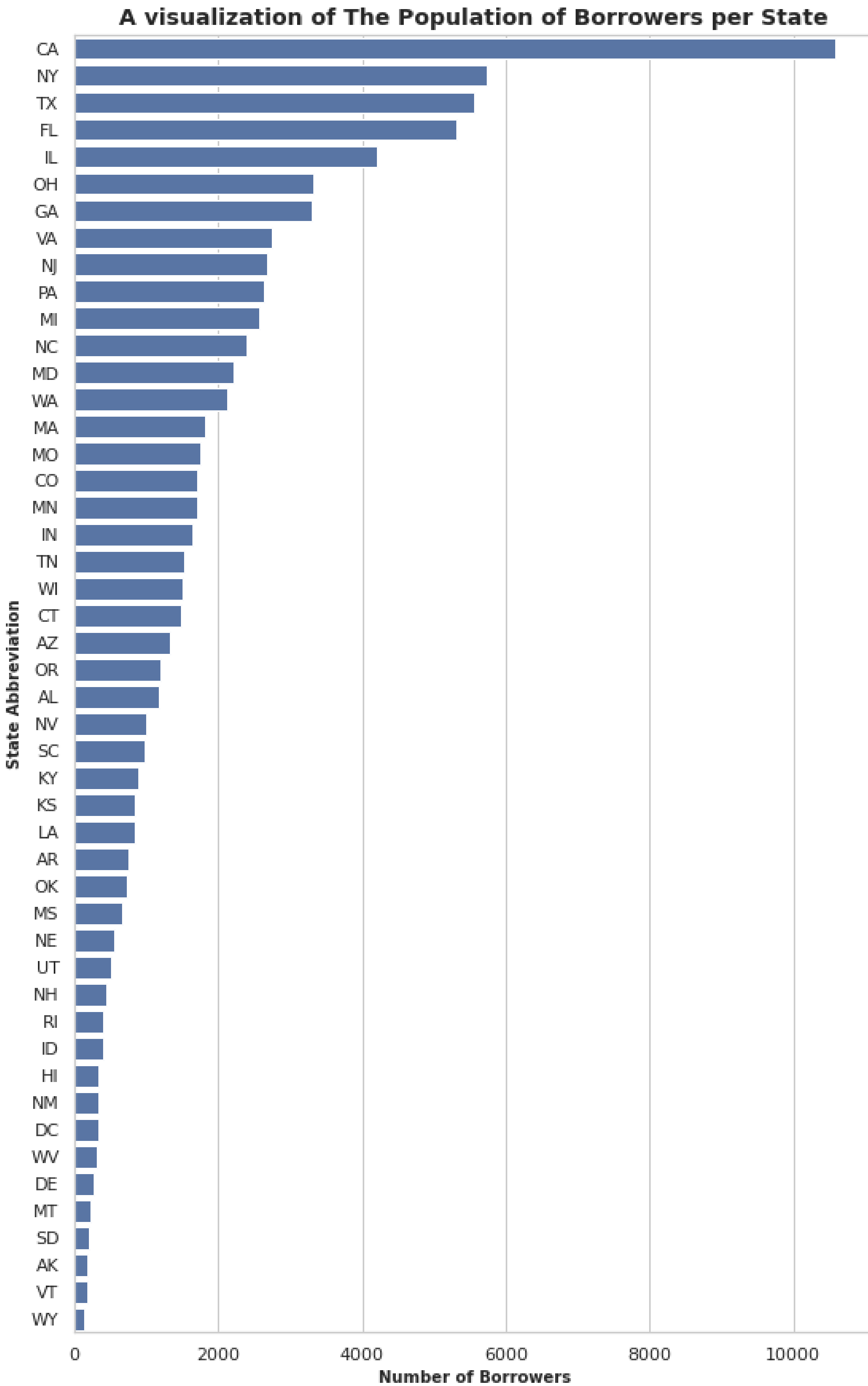
1.2 Dataset Overview

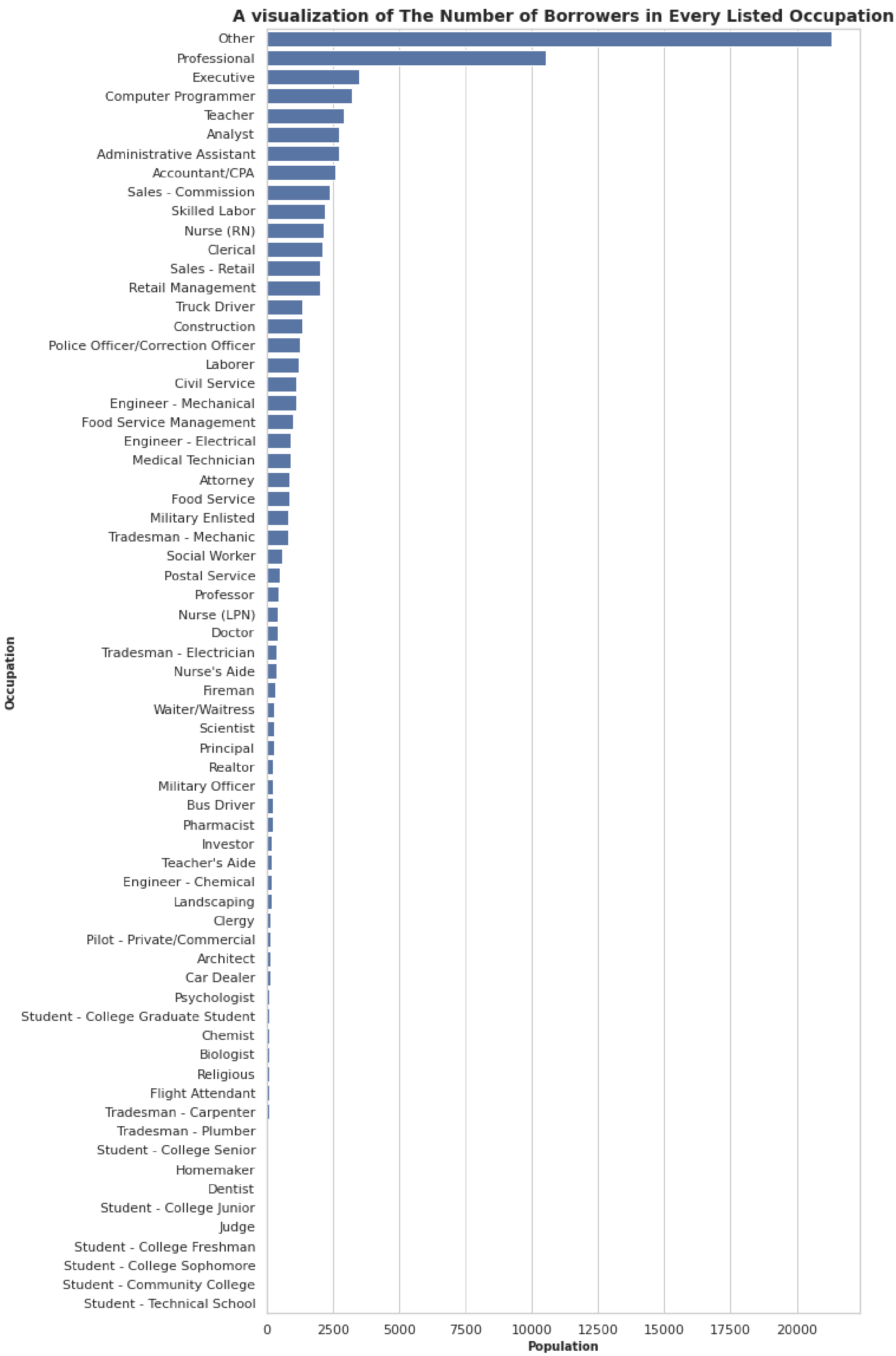
The cleaned data consisted of Loan borrower information of 83,507 loan borrowers, with each entry having 15 attributes. The attributes include the above listed borrower traits of interest among others

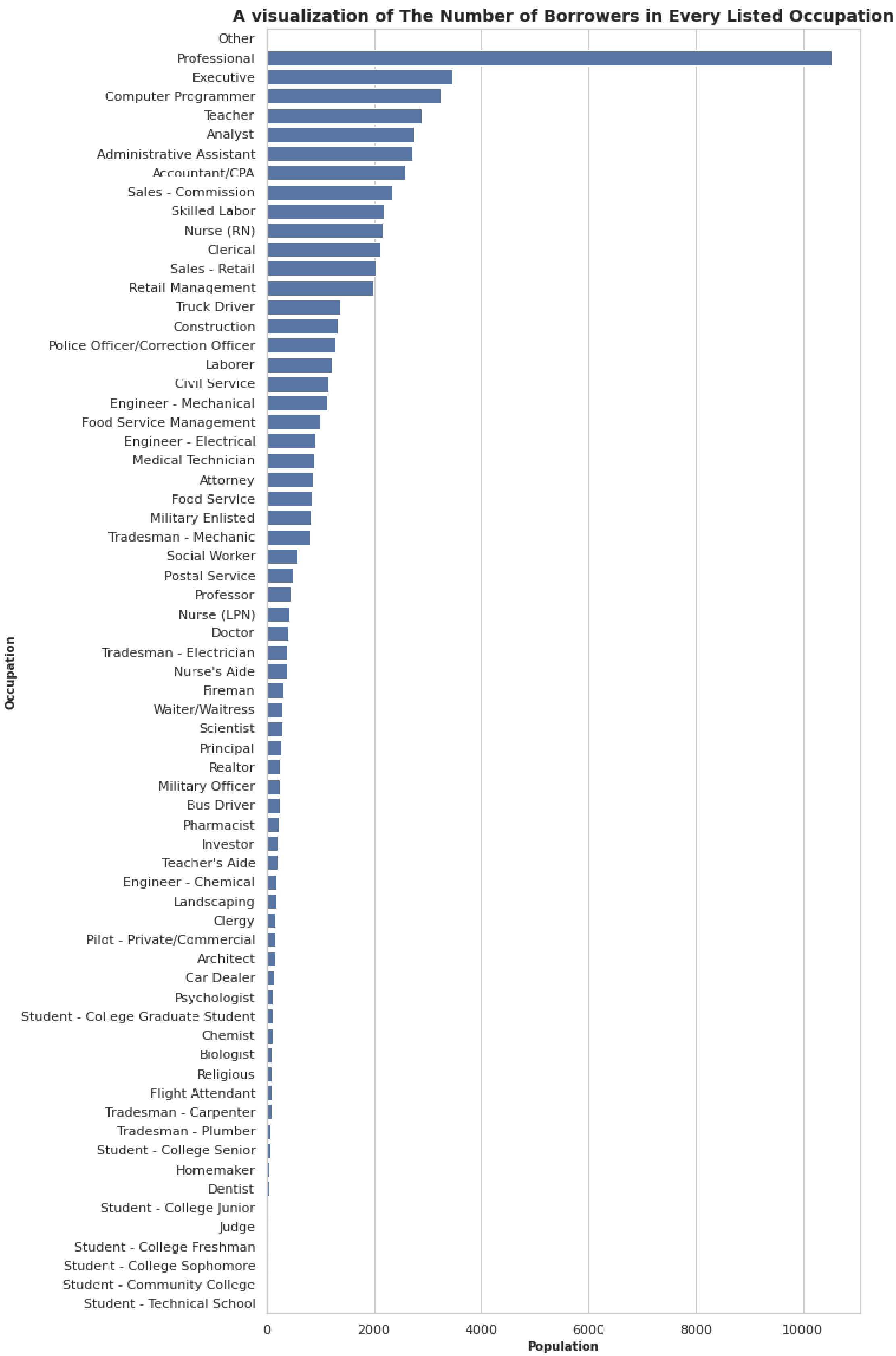
[1]:
```
# import all packages and set plots to be embedded inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import import_ipynb

from Loan_Data_Exploration_Part1 import regPlots, splitString # Imports␣
 ↪regPlots function defined in the part1 notebook

%matplotlib inline



# suppress warnings from final output
import warnings
warnings.simplefilter("ignore")
```

```
importing Jupyter notebook from Loan_Data_Exploration_Part1.ipynb
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 113937 entries, 0 to 113936
Data columns (total 16 columns):
 #   Column                      Non-Null Count   Dtype
```

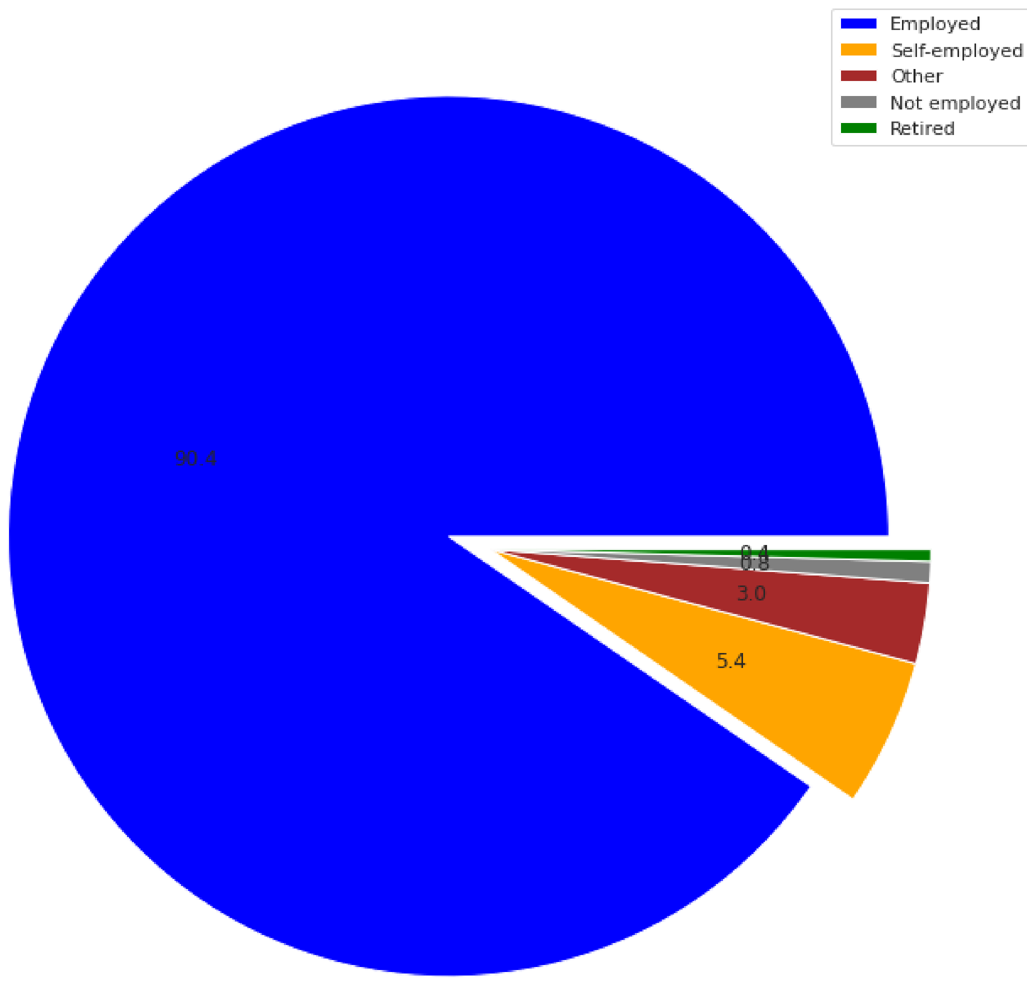
```
---  ------                    --------------   -----
 0   Term                      113937 non-null  int64
 1   ProsperScore              84853 non-null   float64
 2   BorrowerState             108422 non-null  object
 3   Occupation                110349 non-null  object
 4   EmploymentStatus          111682 non-null  object
 5   EmploymentStatusDuration  106312 non-null  float64
 6   IsBorrowerHomeowner       113937 non-null  bool
 7   CreditScoreRangeLower     113346 non-null  float64
 8   CreditScoreRangeUpper     113346 non-null  float64
 9   DelinquenciesLast7Years   112947 non-null  float64
 10  StatedMonthlyIncome       113937 non-null  float64
 11  LoanNumber                113937 non-null  int64
 12  LoanOriginalAmount        113937 non-null  int64
 13  MonthlyLoanPayment        113937 non-null  float64
 14  LP_CustomerPayments       113937 non-null  float64
 15  LP_InterestandFees        113937 non-null  float64
dtypes: bool(1), float64(9), int64(3), object(3)
memory usage: 13.1+ MB
<class 'pandas.core.frame.DataFrame'>
Int64Index: 83507 entries, 1 to 113936
Data columns (total 16 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Term                      83507 non-null  int64
 1   ProsperScore              83507 non-null  float64
 2   BorrowerState             83507 non-null  object
 3   Occupation                83507 non-null  object
 4   EmploymentStatus          83507 non-null  object
 5   EmploymentStatusDuration  83507 non-null  float64
 6   IsBorrowerHomeowner       83507 non-null  bool
 7   CreditScoreRangeLower     83507 non-null  float64
 8   CreditScoreRangeUpper     83507 non-null  float64
 9   DelinquenciesLast7Years   83507 non-null  float64
 10  StatedMonthlyIncome       83507 non-null  float64
 11  LoanNumber                83507 non-null  int64
 12  LoanOriginalAmount        83507 non-null  int64
 13  MonthlyLoanPayment        83507 non-null  float64
 14  LP_CustomerPayments       83507 non-null  float64
 15  LP_InterestandFees        83507 non-null  float64
dtypes: bool(1), float64(9), int64(3), object(3)
memory usage: 10.3+ MB
```

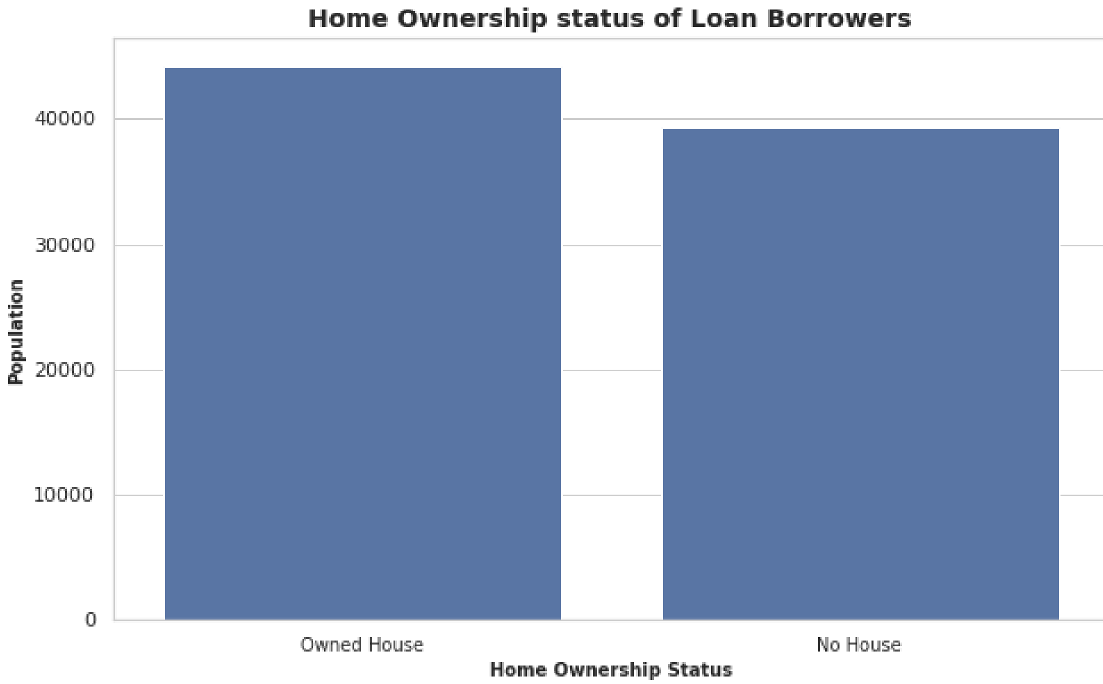

A visualization of The Population of Borrowers per State
State Abbreviation
CA
NY
TX
FL
IL
OH
GA
VA
NJ
PA
MI
NC
MD
WA
MA
MO
CO
MN
IN
TN
WI
CT
AZ
OR
AL
NV
SC
KY
KS
LA
AR
OK
MS
NE
UT
NH
RI
ID
HI
NM
DC
WV
DE
MT
SD
AK
VT
WY
0
2000
4000
6000
8000
10000
Number of Borrowers

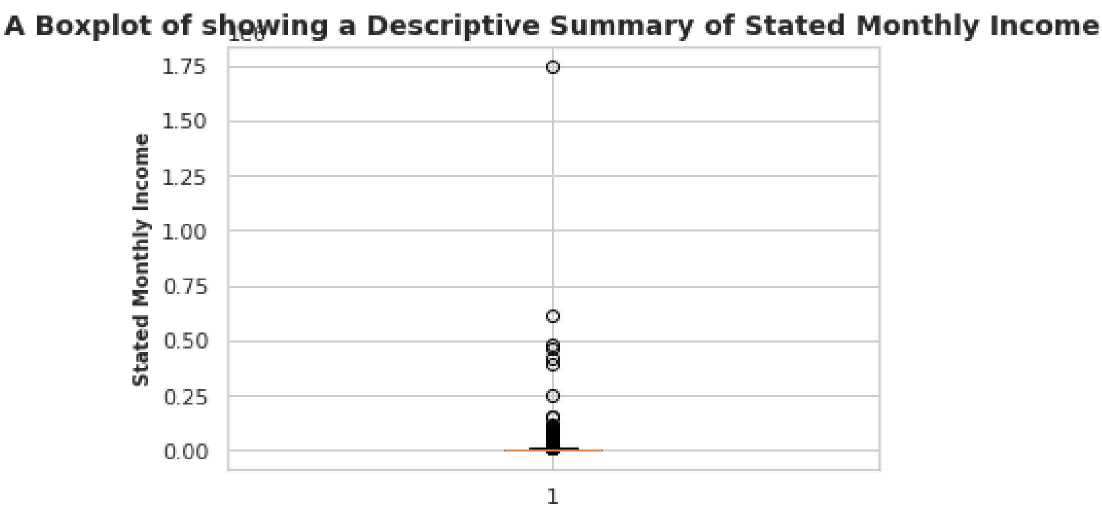

A visualization of The Number of Borrowers in Every Listed Occupation
Occupation
Other
Professional
Executive
Computer Programmer
Teacher
Analyst
Administrative Assistant
Accountant/CPA
Sales - Commission
Skilled Labor
Nurse (RN)
Clerical
Sales - Retail
Retail Management
Truck Driver
Construction
Police Officer/Correction Officer
Laborer
Civil Service
Engineer - Mechanical
Food Service Management
Engineer - Electrical
Medical Technician
Attorney
Food Service
Military Enlisted
Tradesman - Mechanic
Social Worker
Postal Service
Professor
Nurse (LPN)
Doctor
Tradesman - Electrician
Nurse's Aide
Fireman
Waiter/Waitress
Scientist
Principal
Realtor
Military Officer
Bus Driver
Pharmacist
Investor
Teacher's Aide
Engineer - Chemical
Landscaping
Clergy
Pilot - Private/Commercial
Architect
Car Dealer
Psychologist
Student - College Graduate Student
Chemist
Biologist
Religious
Flight Attendant
Tradesman - Carpenter
Tradesman - Plumber
Student - College Senior
Homemaker
Dentist
Student - College Junior
Judge
Student - College Freshman
Student - College Sophomore
Student - Community College
Student - Technical School
0
2500
5000
7500
10000
12500
15000
17500
20000
Population

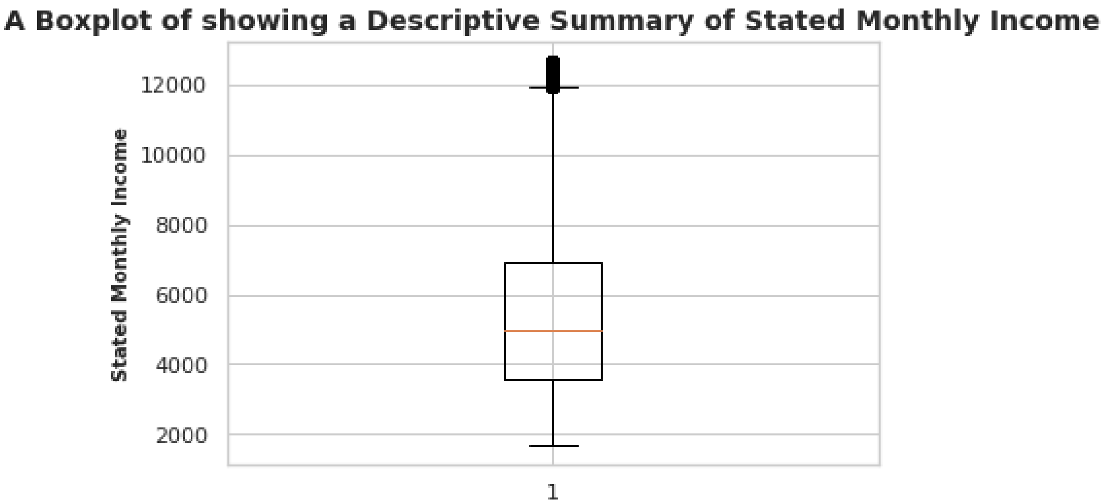

A visualization of The Number of Borrowers in Every Listed Occupation
Occupation
Other
Professional
Executive
Computer Programmer
Teacher
Analyst
Administrative Assistant
Accountant/CPA
Sales - Commission
Skilled Labor
Nurse (RN)
Clerical
Sales - Retail
Retail Management
Truck Driver
Construction
Police Officer/Correction Officer
Laborer
Civil Service
Engineer - Mechanical
Food Service Management
Engineer - Electrical
Medical Technician
Attorney
Food Service
Military Enlisted
Tradesman - Mechanic
Social Worker
Postal Service
Professor
Nurse (LPN)
Doctor
Tradesman - Electrician
Nurse's Aide
Fireman
Waiter/Waitress
Scientist
Principal
Realtor
Military Officer
Bus Driver
Pharmacist
Investor
Teacher's Aide
Engineer - Chemical
Landscaping
Clergy
Pilot - Private/Commercial
Architect
Car Dealer
Psychologist
Student - College Graduate Student
Chemist
Biologist
Religious
Flight Attendant
Tradesman - Carpenter
Tradesman - Plumber
Student - College Senior
Homemaker
Dentist
Student - College Junior
Judge
Student - College Freshman
Student - College Sophomore
Student - Community College
Student - Technical School
0
2000
4000
6000
8000
10000
Population

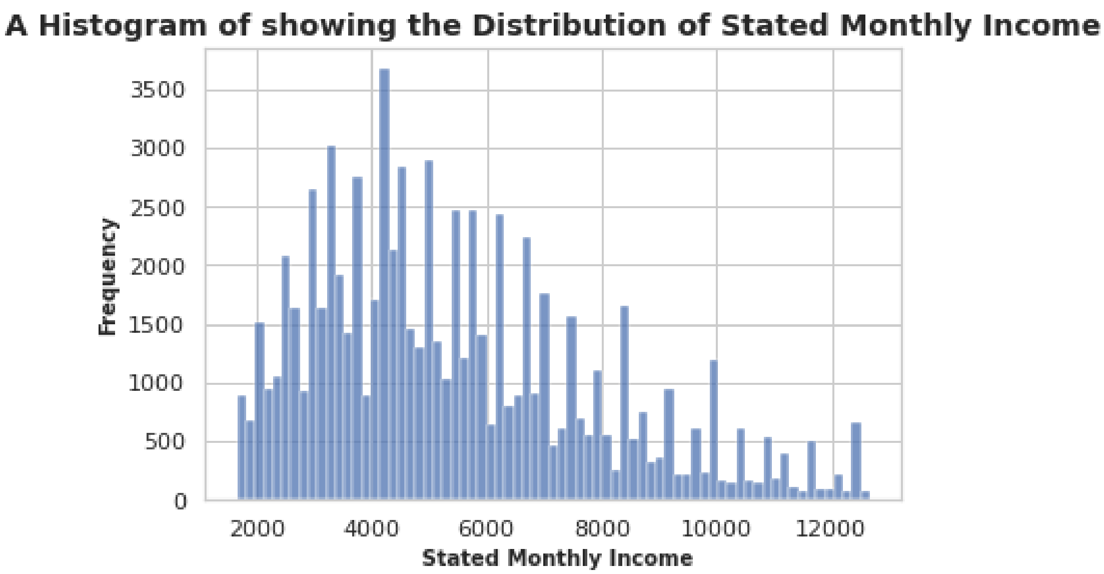
A Pie chart representing the Employment Status of Loan Borrowers

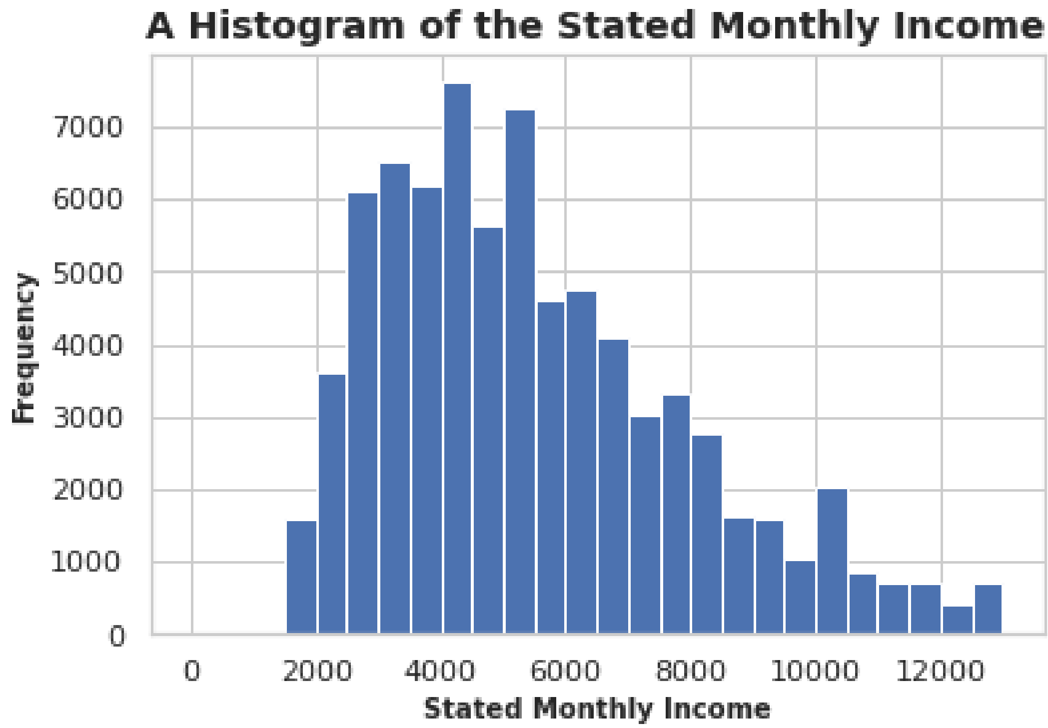


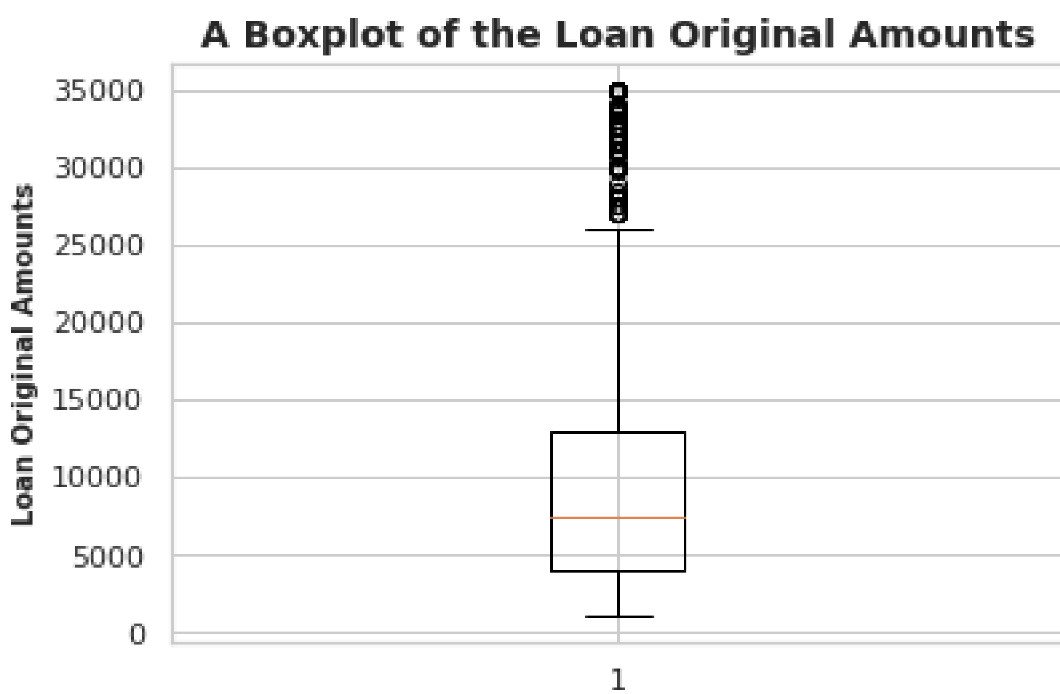

Home Ownership status of Loan Borrowers
Population
40000
30000
20000
10000
0
Owned House
No House
Home Ownership Status

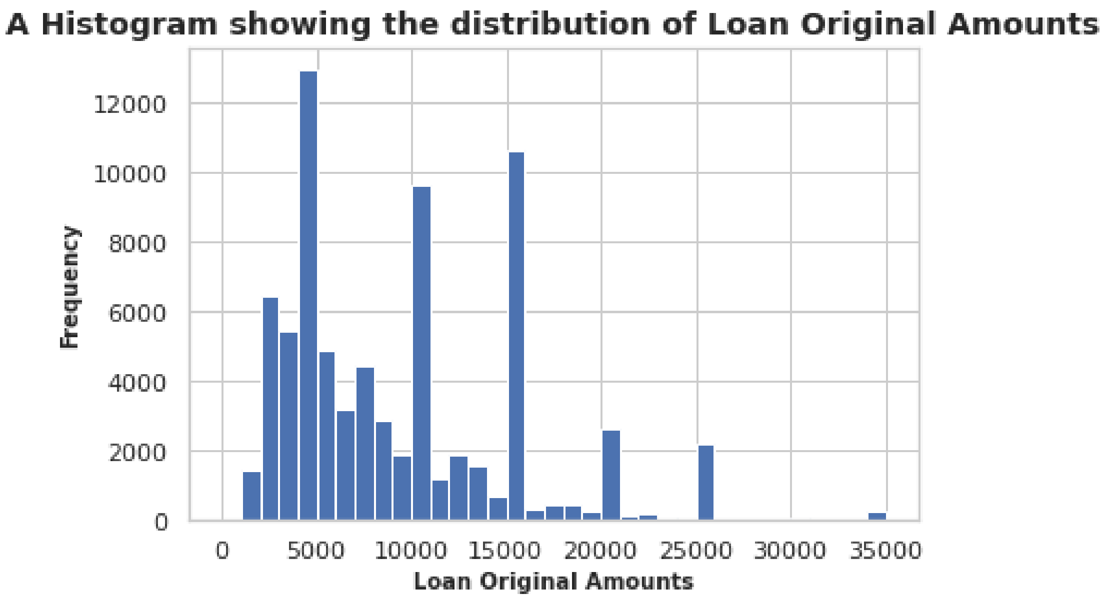

A Boxplot of showing a Descriptive Summary of Stated Monthly Income
1e6
Stated Monthly Income
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
1

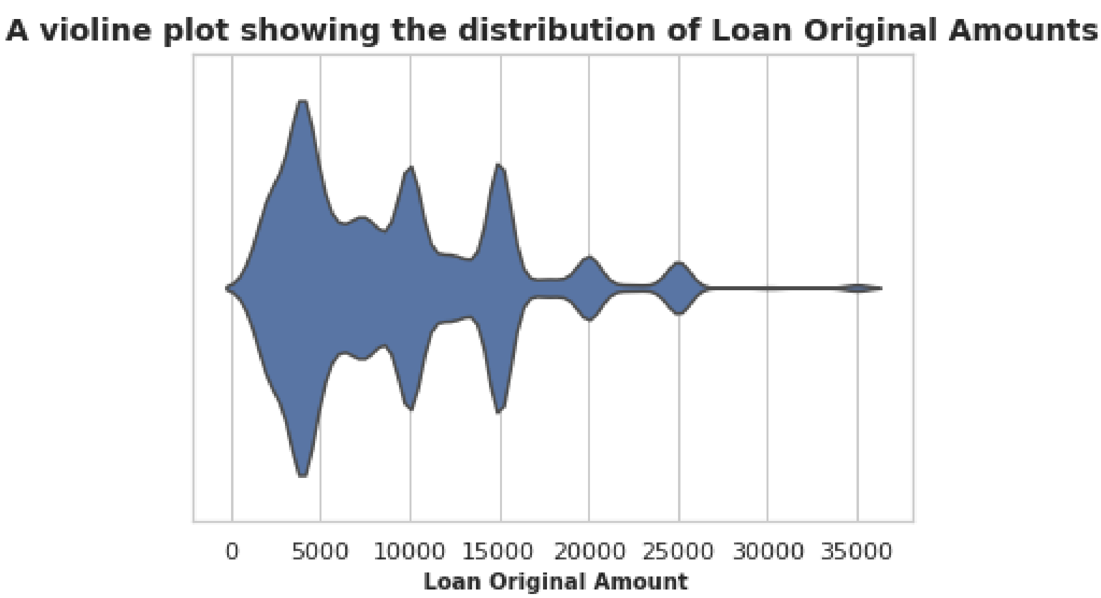

A Boxplot of showing a Descriptive Summary of Stated Monthly Income
Stated Monthly Income
12000
10000
8000
6000
4000
2000
1


A Histogram of showing the Distribution of Stated Monthly Income
Frequency
3500
3000
2500
2000
1500
1000
500
0
2000
4000
6000
8000
10000
12000
Stated Monthly Income


A Histogram of the Stated Monthly Income
Frequency
7000
6000
5000
4000
3000
2000
1000
0
0
2000
4000
6000
8000
10000
12000
Stated Monthly Income


A Boxplot of the Loan Original Amounts
Loan Original Amounts
35000
30000
25000
20000
15000
10000
5000
0
1





[2]:
```
# load in the dataset into a pandas dataframe
clean_loan_df = pd.read_csv('Datasets/clean_loan_data.csv')
```

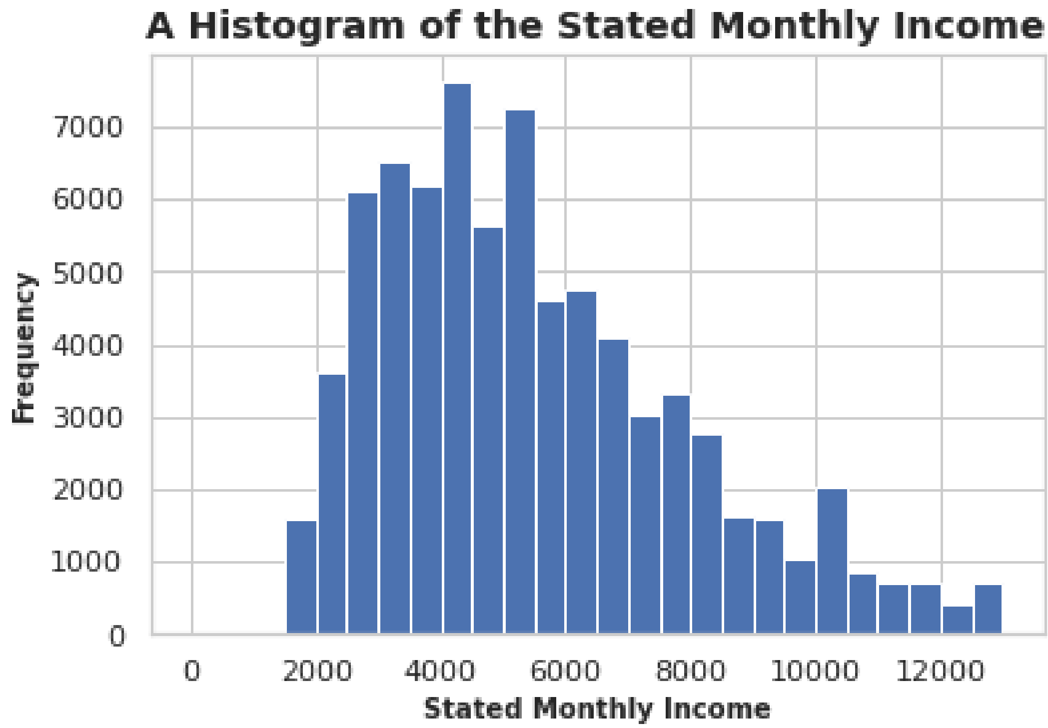
[3]:
```
# data wrangling, removing records with outliers in the stated monthly income␣
↪column

# Calculate the lower quartile and the upper quartile values.
q75, q25 = np.percentile(clean_loan_df.loc[:,'StatedMonthlyIncome'],[75,25])
```

```
# Calculate the Interquartile Range
intr_qr = q75-q25

# Calculate the Minimum and maximum possible values for the stated monthly␣
↪income entries.
maxim = q75+(1.5*intr_qr)
minim = q75-(1.5*intr_qr)

# Replace the outliers with np.nan
clean_loan_df.loc[clean_loan_df['StatedMonthlyIncome'] <␣
↪minim,'StatedMonthlyIncome'] = np.nan
clean_loan_df.loc[clean_loan_df['StatedMonthlyIncome'] >␣
↪maxim,'StatedMonthlyIncome'] = np.nan

# Drop the records with null entries
clean_loan_df.dropna(inplace=True)
```

1.3 Distribution of Stated Monthly Income

`Stated Monthly Income` refers to the amount that a borrower indicated as the amount they receive as monthly payment from their occupation or employment opportunity.

[4]:
```
# A Histogram of Stated Monthly Income
sns.set_theme(style="whitegrid")

bins = np.arange(0, clean_loan_df.StatedMonthlyIncome.max()+500, 500)
plt.hist(data = clean_loan_df, x = 'StatedMonthlyIncome', bins=bins);
plt.title('A Histogram of the Stated Monthly Income', size = 14, weight='bold')
plt.xlabel('Stated Monthly Income', size = 10, weight='bold')
plt.ylabel('Frequency', size = 10, weight='bold');
```



1.4 Observation:

1.4.1 Most of the records are concentrated around between 2000 and 6000, which makes the distribution of the stated monthly incomes right tailed.

1.5 Distribution of Borrowers in every state

[5]:
```
# Using the value_counts() function to view the number of borrowers per state.
borrowers_by_state = clean_loan_df.BorrowerState.value_counts()

# First convert the Series to a pandas dataframe.
borrowers_by_state = borrowers_by_state.to_frame(name='borrower_population').
 ↪reset_index()

# Rename the column with the name index to "State"
borrowers_by_state.rename(columns = {'index':'State'}, inplace=True)

# Using the newly created dataframe, generate a heatmap indicating the␣
 ↪population of borrowers in every state.
fig = px.choropleth(borrowers_by_state,
                   locations='State',
                    locationmode='USA-states',
                   scope = 'usa',
```

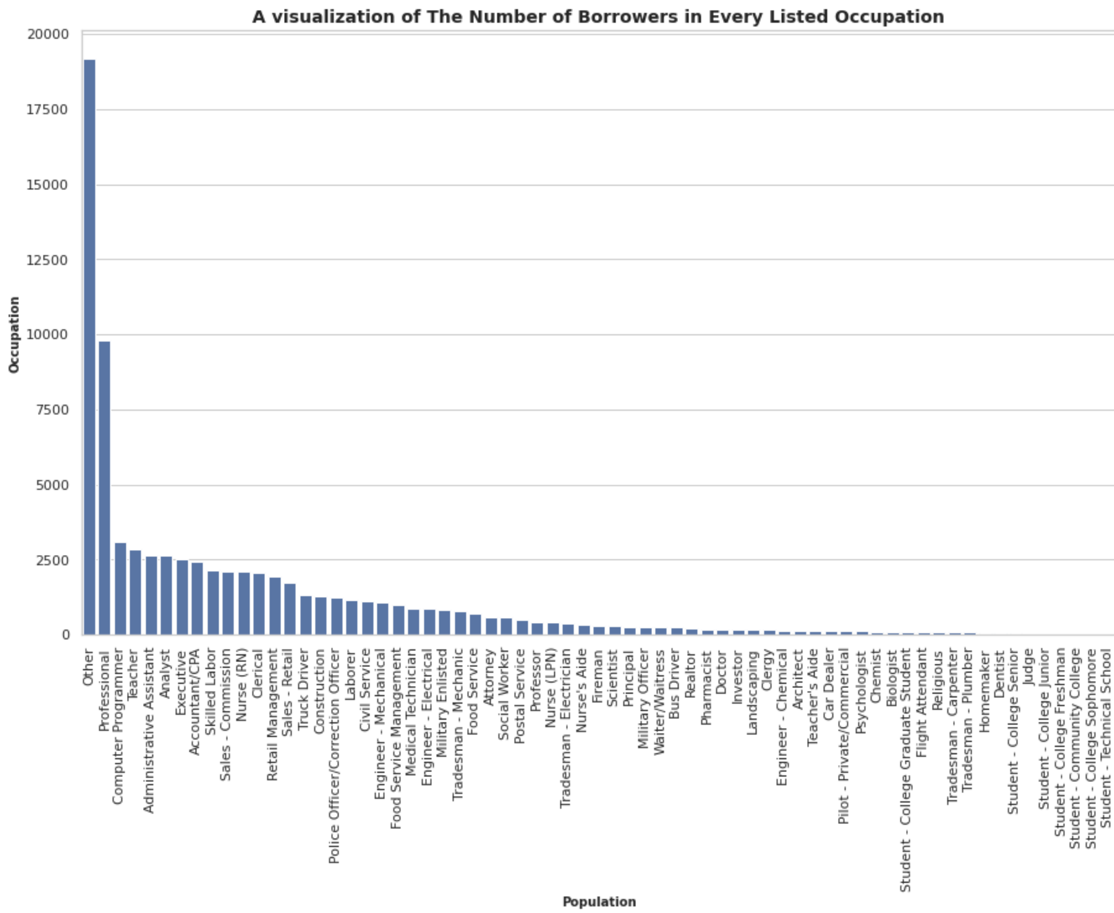
```
                   color='borrower_population',
                   color_continuous_scale=px.colors.sequential.Inferno_r)
fig.update_layout(title_text = 'Borrower population by State',
                  title_font_size = 22,
                  title_font_color = 'black',
                  title_x = 0.5)

fig.show();
```

1.6 Observation:

1.6.1 From a user's perspective, it is evident that California has the highest population of borrowers. In this visualization, the user can clearly see the location of every State, which makes the data even more engaging.

1.7 Number of Borrowers grouped by Occupation

[6]:
```
# Set the theme of the visualization
sns.set_theme(style="whitegrid")

# Set the size of the visualization
f, ax = plt.subplots(figsize=(15,9))

# Set the color of the visualization
base_color = sns.color_palette()[0]

# Define the order in which the bars will appear in the visualization
occupations_order = clean_loan_df.Occupation.value_counts().index

# Visualize the bar graph.
sns.countplot(data=clean_loan_df, x = 'Occupation', color = base_color,␣
 ↪order=occupations_order);
plt.xticks(rotation=90)

# Set the labels and plot title
plt.title('A visualization of The Number of Borrowers in Every Listed␣
 ↪Occupation', size = 14, weight='bold')
plt.xlabel('Population', size = 10, weight='bold')
plt.ylabel('Occupation', size = 10, weight='bold');
```



1.8 Observation:

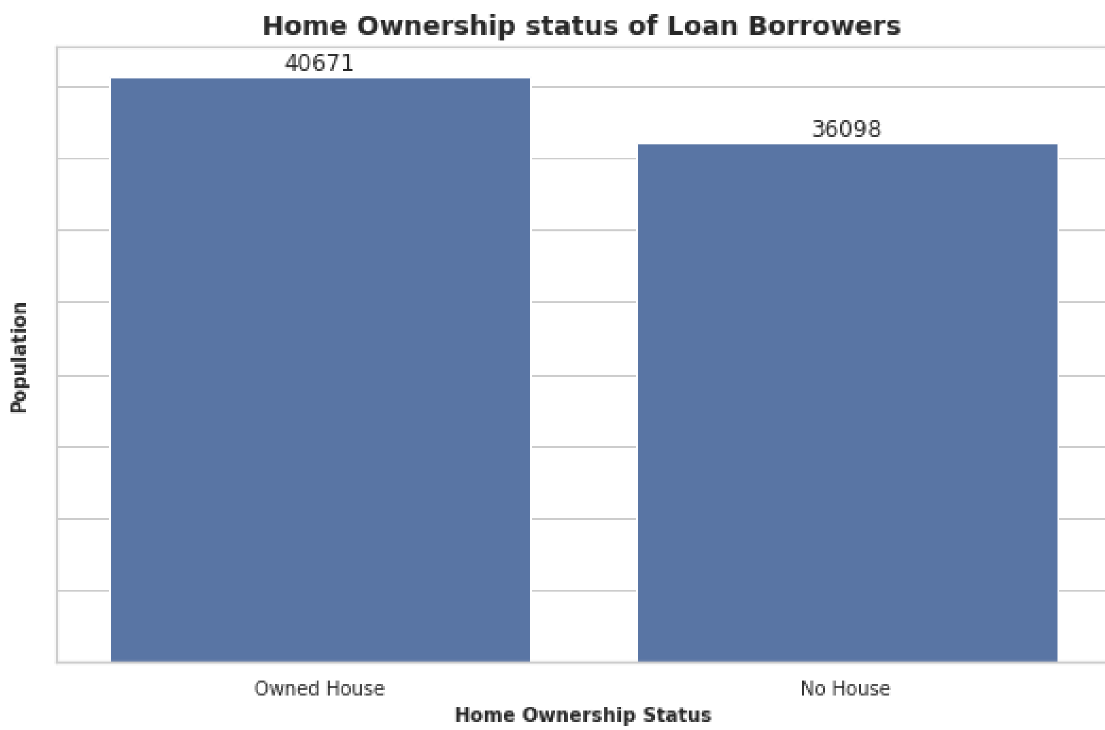
1.8.1 In the population of Borrowers, the number of individuals who listed their occupation as `other` was disproportionately high.

» Further research should be conducted to check whether selecting `other` as an occupation was an escape strategy of avoiding to indicate that they were unemployed.

1.9 Loan Borrowers grouped by Home Ownership

[7]:
```
# Set the theme of the visualization
sns.set_theme(style="whitegrid")

# Set the size of the visualization
f, ax = plt.subplots(figsize=(10,6))

# Set the color of the visualization
base_color = sns.color_palette()[0]
```

```
# Define the order in which the bars will appear in the visualization
arr_order = clean_loan_df.IsBorrowerHomeowner.value_counts().index

# Visualize the bar graph.
sns.countplot(data=clean_loan_df, x = 'IsBorrowerHomeowner', color =␣
↪base_color, order=arr_order);
ax.bar_label(ax.containers[0])

ax.set_xticklabels(['Owned House', 'No House'], size=10)

# Set the labels and plot title
plt.title('Home Ownership status of Loan Borrowers', size = 14, weight='bold')
plt.xlabel('Home Ownership Status', size = 10, weight='bold')
plt.ylabel('Population', size = 10, weight='bold')
ax.set(yticklabels=[]);
```

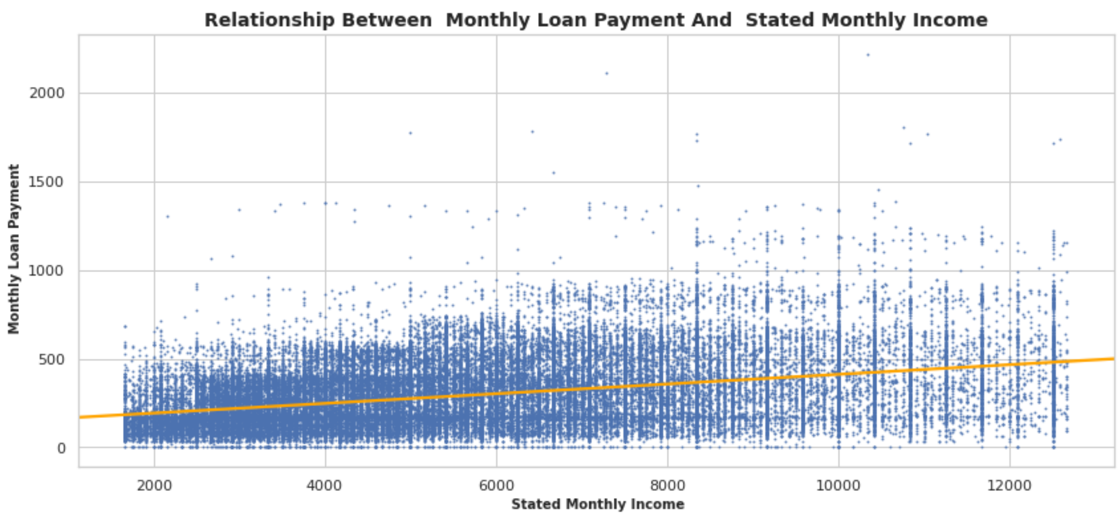


1.10 Observation

1.10.1 A slightly higher number of borrowers owned houses. Precisely, in the cleaned data, 40,671 borrowers owned homes while 36,098 borrowers were not home owners.

1.11 `Stated Monthly Salary` Versus `Montly Loan Payment`

[8]:
```
# A scatter plot that explores the relationship between these two variables.
regPlots(clean_loan_df, 'StatedMonthlyIncome', 'MonthlyLoanPayment', 0, 0)
```

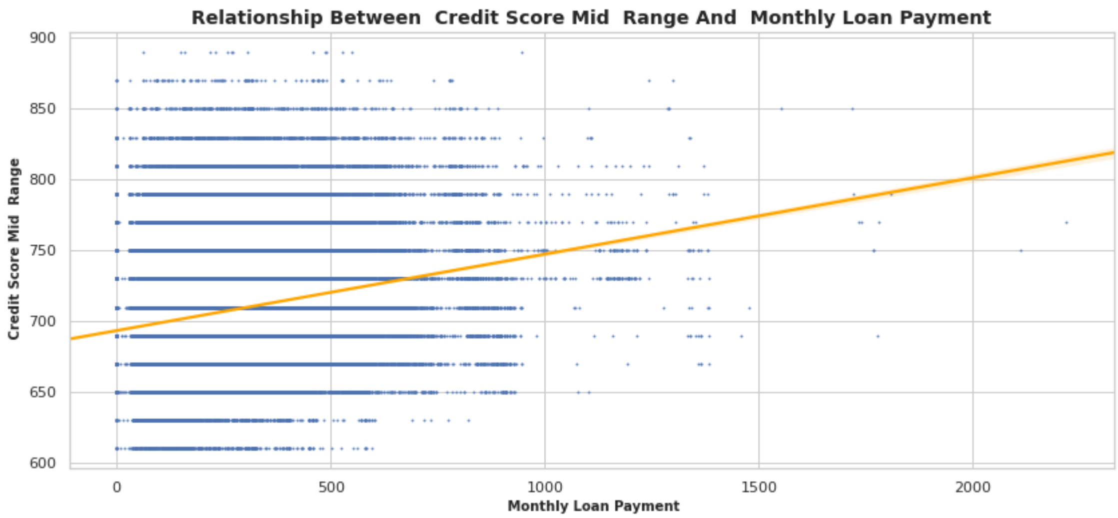


1.12 Comment:

1.12.1 The scatter plot above indicates a positive relationship between the `stated monthly income` and `credit score midrange`. This implies that `borrowers with a higher stated monthly income tend to have a higher credit score mid_range`

1.13 `Credit Score Mid_range` versus `Monthly Loan Payment`

[9]:
```
# A scatter plot of Monthly Loan Payment against Credit Score Midrange
regPlots(clean_loan_df, 'MonthlyLoanPayment', 'CreditScoreMid_range', 0, 0)
```



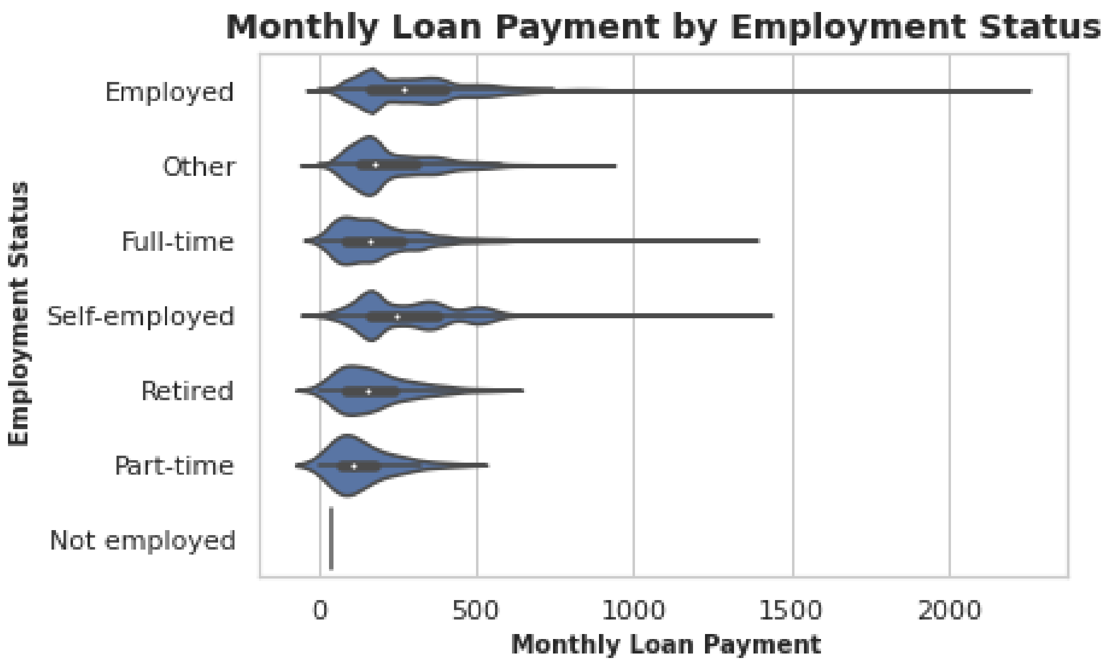
1.14 Comment:

1.14.1 There exist a positive correlation between monthly loan payment and credit score. This means that borrowers with a higher credit score tend have a higher monthly loan payment amount.

1.15 Monthly Loan Payment by Employment Status

[10]:
```
color = sns.color_palette()[0]
sns.violinplot(data = clean_loan_df, y = 'EmploymentStatus', x =␣
↪'MonthlyLoanPayment', color=color)
plt.title('Monthly Loan Payment by Employment Status', size = 14,␣
↪weight='bold');
plt.xlabel(splitString('MonthlyLoanPayment'), size = 10, weight='bold')
plt.ylabel(splitString('EmploymentStatus'), size = 10, weight='bold')
```

[10]:
```
Text(0, 0.5, ' Employment Status')
```



1.16 Comment:

1.16.1 The Monthly Loan Payment of Employed borrowers varies more as compared to the monthly loan payment of individuals who had other employment status.

1.16.2 Borrowers who were not employed had a monthly loan payment which was almost zore on average.

[12]:
```
!jupyter nbconvert Loan_Data_Exploration_Part2.ipynb --to slides --post serve␣
↪--no-input --no-prompt
```

```
[NbConvertApp] Converting notebook Loan_Data_Exploration_Part2.ipynb to slides
[NbConvertApp] Writing 726541 bytes to Loan_Data_Exploration_Part2.slides.html
[NbConvertApp] Redirecting reveal.js requests to
https://cdnjs.cloudflare.com/ajax/libs/reveal.js/3.5.0
Serving your slides at
http://127.0.0.1:8000/Loan_Data_Exploration_Part2.slides.html
Use Control-C to stop this server
WARNING:tornado.access:404 GET /custom.css (127.0.0.1) 0.74ms
WARNING:tornado.access:404 GET /plotly.js (127.0.0.1) 0.52ms
^C

Interrupted
```

[ ]: