TRAINITY

BANK LOAN CASE STUDY

BY : DHIVYALAKSHMI R

TABLE OF CONTENT

PROJECT DESCRIPTION

Project Objective: This project aims to use data analysis techniques to study patterns in customer data, primarily to reduce financial losses when giving loans in the consumer finance sector.

Target Audience: Mainly focusing on people living in cities who often face difficulties getting loans because they have little or no credit history.

Data Analysis Goal: Want to carefully examine the available data to find clear patterns and important signs that can help us predict if someone can pay back a loan.

Smart Decision-Making: By doing this thorough analysis, want to provide the company with the information it needs to make smart decisions and not reject people who could actually repay their loans

APPROACH

- Data Acquisition and Comprehension
- Data Preprocessing and
- Exploratory Data Analysis
- Feature Crafting and Enhancement
- Examination of Data
- Risk Assessment and Analytics
- Generation of Reports Provision of Recommendations

TECH-STACK USED

MICROSOFT EXCEL

PYTHON

GOOGLE COLAB

INSIGHTS

Credit History Limitations:

- **Important Variables:** To decide if someone is a risky borrower or not, lenders consider loan amount, income, credit score, and employment history.

Key Variables for Risk Assessment:

- Loan amount, income, and credit score indicate borrower risk.
- Employment history reflects stability.

Exploratory Analysis for Risk Identification:

- Analyzing data helps identify factors like age, location, or occupation linked to loan defaults.

INSIGHTS

Demographic Factors Impact Repayment:

Factors like age, location, and occupation can influence a person's ability to repay a loan

Impact of Demographic Factors:

- Younger age may pose higher risk; older age is often less risky.
- Location affects job opportunities and living costs, influencing repayment.
- Occupation type determines income stability, with some jobs being riskier.

SOLUTIONS

HANDLING MISSING DATA

Task: Identify the missing data in the dataset and decide on an appropriate method to deal

```
[1]  import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns
     import math


     pd.set_option('display.max_rows', None)
     pd.set_option('display.max_columns', None)

[2]  app_df=pd.read_csv("/content/application_data.csv")
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import math


pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

prev_df=pd.read_csv("/content/previous_application.csv")
```

In a Colab Notebook, the provided code snippets are employed to import data from two distinct files, specifically "previous_application.csv"and "application_data.csv"

```
app_df.shape

(49999, 122)
```

consists of 49999 rows and 122 columns.

```
prev_df.shape

(49999, 37)
```

consists of 49999 rows and 37 columns.

```
app_df.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49999 entries, 0 to 49998
Data columns (total 122 columns):
 #    Column                       Dtype
---   ------                       -----
 0    SK_ID_CURR                   int64
 1    TARGET                       int64
 2    NAME_CONTRACT_TYPE           object
 3    CODE_GENDER                  object
 4    FLAG_OWN_CAR                 object
 5    FLAG_OWN_REALTY              object
 6    CNT_CHILDREN                 int64
 7    AMT_INCOME_TOTAL             float64
 8    AMT_CREDIT                   float64
 9    AMT_ANNUITY                  float64
 10   AMT_GOODS_PRICE              float64
 11   NAME_TYPE_SUITE              object
 12   NAME_INCOME_TYPE             object
 13   NAME_EDUCATION_TYPE          object
 14   NAME_FAMILY_STATUS           object
 15   NAME_HOUSING_TYPE            object
 16   REGION_POPULATION_RELATIVE   float64
```

```
prev_df.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49999 entries, 0 to 49998
Data columns (total 37 columns):
 #    Column                       Non-Null Count  Dtype
---   ------                       --------------  -----
 0    SK_ID_PREV                   49999 non-null  int64
 1    SK_ID_CURR                   49999 non-null  int64
 2    NAME_CONTRACT_TYPE           49999 non-null  object
 3    AMT_ANNUITY                  39407 non-null  float64
 4    AMT_APPLICATION              49999 non-null  float64
 5    AMT_CREDIT                   49999 non-null  float64
 6    AMT_DOWN_PAYMENT             24801 non-null  float64
 7    AMT_GOODS_PRICE              39255 non-null  float64
 8    WEEKDAY_APPR_PROCESS_START   49999 non-null  object
 9    HOUR_APPR_PROCESS_START      49999 non-null  int64
 10   FLAG_LAST_APPL_PER_CONTRACT  49999 non-null  object
 11   NFLAG_LAST_APPL_IN_DAY       49999 non-null  int64
 12   RATE_DOWN_PAYMENT            24801 non-null  float64
 13   RATE_INTEREST_PRIMARY        165 non-null    float64
 14   RATE_INTEREST_PRIVILEGED     165 non-null    float64
 15   NAME_CASH_LOAN_PURPOSE       49999 non-null  object
 16   NAME_CONTRACT_STATUS         49999 non-null  object
```

The information regarding the column name, number of non-null values, count, and data types of the previous_application and application_data files is presented.

```
app_df.describe()
```

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	REGION_POPULATION_RELATIVE
count	49999.000000	49999.000000	49999.000000	4.999900e+04	4.999900e+04	49998.000000	4.996100e+04	49999.000000
mean	129013.210584	0.080522	0.419848	1.707676e+05	5.997006e+05	27107.377355	5.390600e+05	0.020798
std	16690.512048	0.272102	0.724039	5.318191e+05	4.024154e+05	14562.944435	3.698533e+05	0.013761
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04	2052.000000	4.500000e+04	0.000533
25%	114570.500000	0.000000	0.000000	1.125000e+05	2.700000e+05	16456.500000	2.385000e+05	0.010006
50%	129076.000000	0.000000	0.000000	1.458000e+05	5.147775e+05	24939.000000	4.500000e+05	0.018850
75%	143438.500000	0.000000	1.000000	2.025000e+05	8.086500e+05	34596.000000	6.795000e+05	0.028663
max	157875.000000	1.000000	11.000000	1.170000e+08	4.050000e+06	258025.500000	4.050000e+06	0.072508

```
[11] prev_df.describe()
```

	SK_ID_PREV	SK_ID_CURR	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT	AMT_GOODS_PRICE	HOUR_APPR_PROCESS_START
count	4.999900e+04	49999.000000	39407.000000	4.999900e+04	4.999900e+04	2.480100e+04	3.925500e+04	49999.000000
mean	1.922254e+06	278983.187604	15482.596847	1.688925e+05	1.885429e+05	6.557571e+03	2.151414e+05	12.478330
std	5.351980e+05	102780.124434	14530.971854	2.822035e+05	3.084736e+05	1.744458e+04	3.024993e+05	3.333012
min	1.000001e+06	100007.000000	0.000000	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000
25%	1.457920e+06	189919.500000	6122.835000	2.204550e+04	2.605500e+04	0.000000e+00	4.941000e+04	10.000000
50%	1.920889e+06	279264.000000	10879.920000	7.155000e+04	7.890750e+04	1.566000e+03	1.040175e+05	12.000000
75%	2.388632e+06	368527.500000	19669.140000	1.800000e+05	1.981058e+05	7.875000e+03	2.250000e+05	15.000000
max	2.845367e+06	456254.000000	234478.395000	3.826372e+06	4.104351e+06	1.035000e+06	3.826372e+06	23.000000

The descriptive statistics including count, mean, standard deviation, minimum value, 25th percentile, 50th percentile, 75th percentile, and maximum value for both previous_application and application_data files are computed using the describe function.

PERCENTAGE OF MISSING VALUES

```
round(app_df.isnull().sum()/app_df.shape[0]*100,2).sort_values(ascending=False)
```

```
COMMONAREA_MODE                69.16
COMMONAREA_AVG                 69.16
COMMONAREA_MEDI                69.16
NONLIVINGAPARTMENTS_MODE       68.88
NONLIVINGAPARTMENTS_AVG        68.88
NONLIVINGAPARTMENTS_MEDI       68.88
LIVINGAPARTMENTS_AVG           67.82
LIVINGAPARTMENTS_MODE          67.82
LIVINGAPARTMENTS_MEDI          67.82
FONDKAPREMONT_MODE             67.76
FLOORSMIN_MEDI                 67.19
FLOORSMIN_MODE                 67.19
FLOORSMIN_AVG                  67.19
OWN_CAR_AGE                    66.24
YEARS_BUILD_MEDI               65.70
YEARS_BUILD_AVG                65.70
YEARS_BUILD_MODE               65.70
LANDAREA_MODE                  58.59
LANDAREA_AVG                   58.59
LANDAREA_MEDI                  58.59
BASEMENTAREA_MEDI              57.86
```

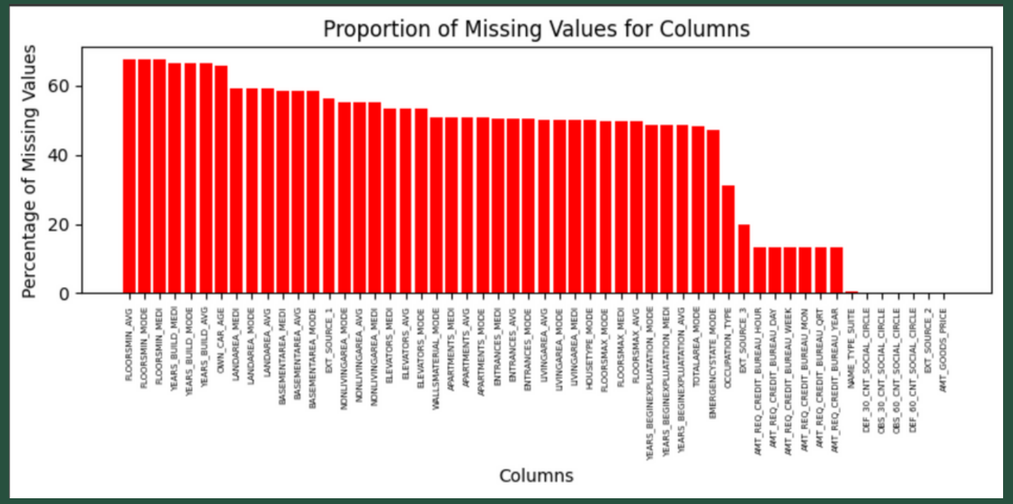
The calculation of the percentage of missing values in each column of the application_data has been performed.

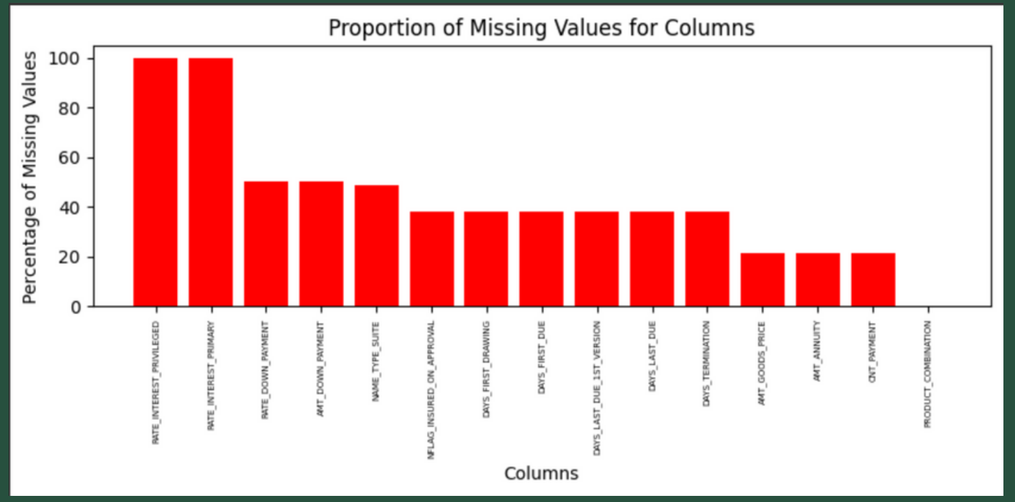
```
[12] round(prev_df.isnull().sum()/prev_df.shape[0]*100,2).sort_values(ascending=False)
```

```
RATE_INTEREST_PRIVILEGED       99.67
RATE_INTEREST_PRIMARY          99.67
RATE_DOWN_PAYMENT              50.40
AMT_DOWN_PAYMENT               50.40
NAME_TYPE_SUITE                48.49
NFLAG_INSURED_ON_APPROVAL      38.32
DAYS_FIRST_DRAWING             38.32
DAYS_FIRST_DUE                 38.32
DAYS_LAST_DUE_1ST_VERSION      38.32
DAYS_LAST_DUE                  38.32
DAYS_TERMINATION               38.32
AMT_GOODS_PRICE                21.49
AMT_ANNUITY                    21.18
CNT_PAYMENT                    21.18
PRODUCT_COMBINATION             0.02
CHANNEL_TYPE                    0.00
NAME_PRODUCT_TYPE               0.00
NAME_YIELD_GROUP                0.00
SELLERPLACE_AREA                0.00
NAME_SELLER_INDUSTRY            0.00
NAME_GOODS_CATEGORY             0.00
NAME_PORTFOLIO                  0.00
SK_ID_PREV                      0.00
```

The calculation of the percentage of missing values in each column of the previous_application data has been performed.

Bar chart to visualize missing value proportions





The bar chart visually identify columns with high proportions of missing data, helping prioritize data cleaning and imputation efforts, ultimately ensuring data quality for analysis.

DROPPING OF COLUMNS

```
thresh=len(app_df)*0.45
columns_to_drop = app_df.columns[app_df.isnull().sum() >=thresh]
app_df.drop(columns=columns_to_drop, inplace=True)
```

```
thresh=len(prev_df)*0.45
columns_to_drop = prev_df.columns[prev_df.isnull().sum() >=thresh]
prev_df.drop(columns=columns_to_drop, inplace=True)
```

Dropping columns of application_data whose missing value percentage is greater than or equal to 45%.

```
app_df.shape
```

```
(49999, 73)
```

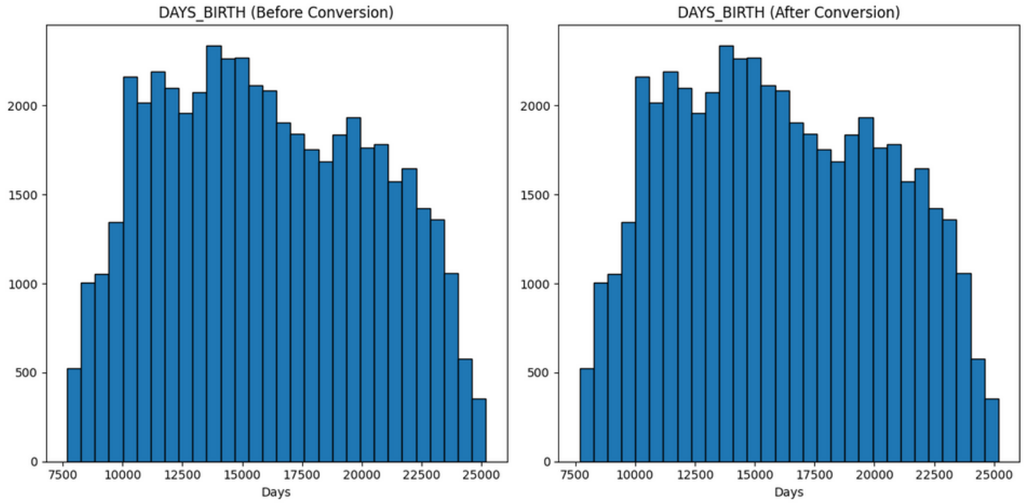
```
prev_df.shape
```

```
(49999, 32)
```

Dropping columns of application_data which are not required for analysis.

"Shape of Application_Data after dropping null columns" suggests that the dataset has 49,999 rows and 73 columns, and "Shape of Previous_Application after dropping null columns" indicates 49,999 rows and 32 columns after removing null columns.

Transforming the negative days column to reflect positive days.

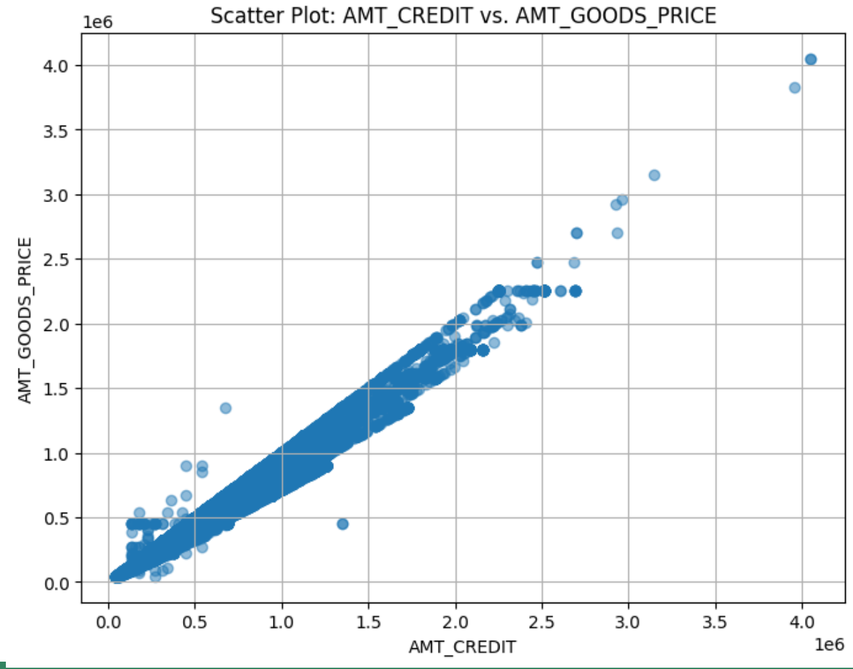


IMPUTING DATA

```
[15] app_df["OCCUPATION_TYPE"].isnull().sum()

     15654

[16] app_df["OCCUPATION_TYPE"].replace(np.NaN,"unknown",inplace=True)

[17] app_df['NAME_TYPE_SUITE'].isnull().sum()

     192

     app_df['NAME_TYPE_SUITE'].mode()

     0    Unaccompanied
     Name: NAME_TYPE_SUITE, dtype: object

     app_df['NAME_TYPE_SUITE'].fillna(app_df['NAME_TYPE_SUITE'].mode()[0], inplace=True)
```

To address the 15654 null values in the OCCUPATION_TYPE column, the plan is to substitute them with the term "unknown." Since the number of null values in the NAME_TYPE_SUITE column is very less we are imputing it with it' s mode.

The scatter plot illustrates a positive correlation between the AMT_GOODS_PRICE and AMT_CREDIT variables. Consequently, it's been determined that filling the null values in the AMT_GOODS_PRICE column with the corresponding values from the AMT_CREDIT column is appropriate.



Replace missing values with the mean or median of the non-missing values in the column. This is a simple and often effective approach.

```
nullcol=['EXT_SOURCE_2','EXT_SOURCE_3']
for column in nullcol:
  app_df[column].fillna(app_df[column].mean, inplace=True)
```

```
nullcols=['OBS_30_CNT_SOCIAL_CIRCLE','DEF_30_CNT_SOCIAL_CIRCLE',
          'OBS_60_CNT_SOCIAL_CIRCLE','DEF_60_CNT_SOCIAL_CIRCLE']
for col in nullcols:
   app_df[col].fillna(app_df[col].median(), inplace=True)
```

Replace missing values with the mode (most frequent category) of the column.

```
nullcols=['AMT_REQ_CREDIT_BUREAU_YEAR','AMT_REQ_CREDIT_BUREAU_QRT',
          'AMT_REQ_CREDIT_BUREAU_MON','AMT_REQ_CREDIT_BUREAU_WEEK',
          'AMT_REQ_CREDIT_BUREAU_DAY','AMT_REQ_CREDIT_BUREAU_HOUR']
for col in nullcols:
   app_df[col].fillna(app_df[col].mode()[0], inplace=True)
```

```
[24]  app_df.dropna(subset=['AMT_GOODS_PRICE'], inplace=True)

[25]  app_df.shape

      (49961, 73)
```

For columns with a small percentage of missing data, then remove rows with missing values

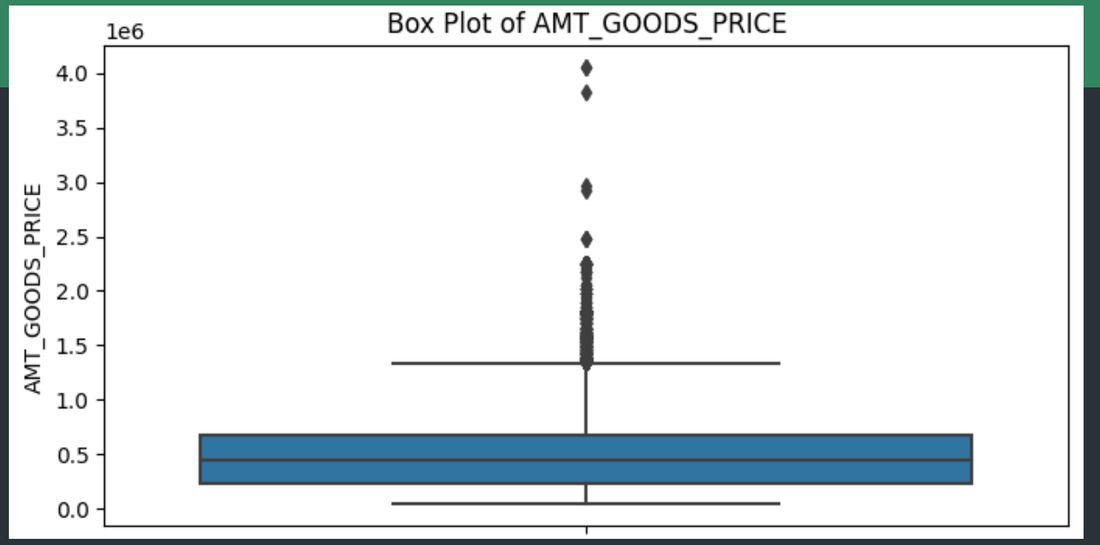
Also,repeat the same process for previous_application

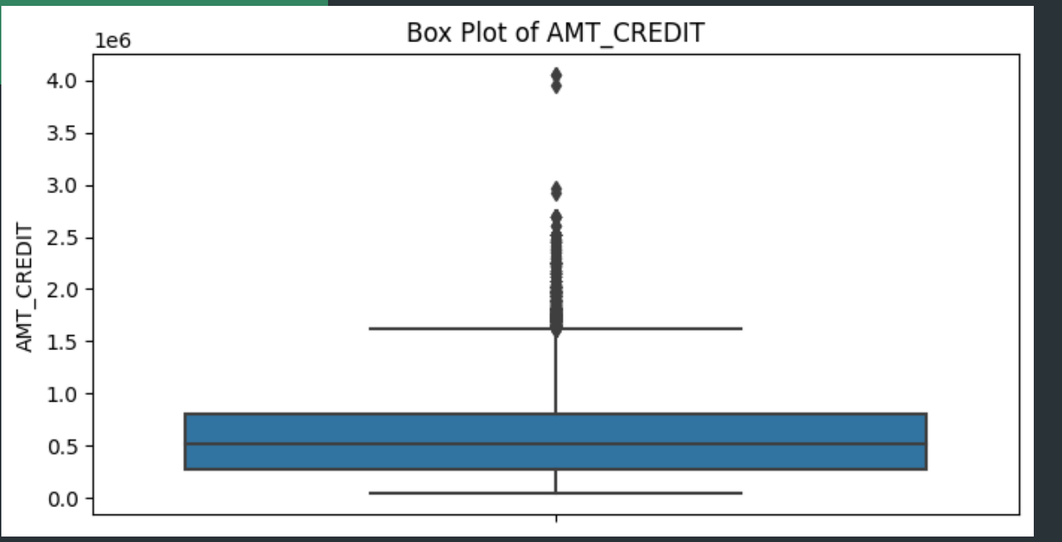
```
[45] nullcols=['NFLAG_INSURED_ON_APPROVAL','DAYS_LAST_DUE','DAYS_FIRST_DUE',
                'DAYS_LAST_DUE_1ST_VERSION','DAYS_FIRST_DRAWING','DAYS_TERMINATION']
     for col in nullcols:
        prev_df[col].fillna(prev_df[col].mode()[0], inplace=True)

[46] nullcolumns=['AMT_GOODS_PRICE','AMT_ANNUITY','CNT_PAYMENT']
     for i in nullcolumns:
        prev_df[i].fillna(prev_df[i].mean(), inplace=True)

[61] prev_df.dropna(subset=['PRODUCT_COMBINATION'], inplace=True)

[62] prev_df.shape

     (49991, 32)
```

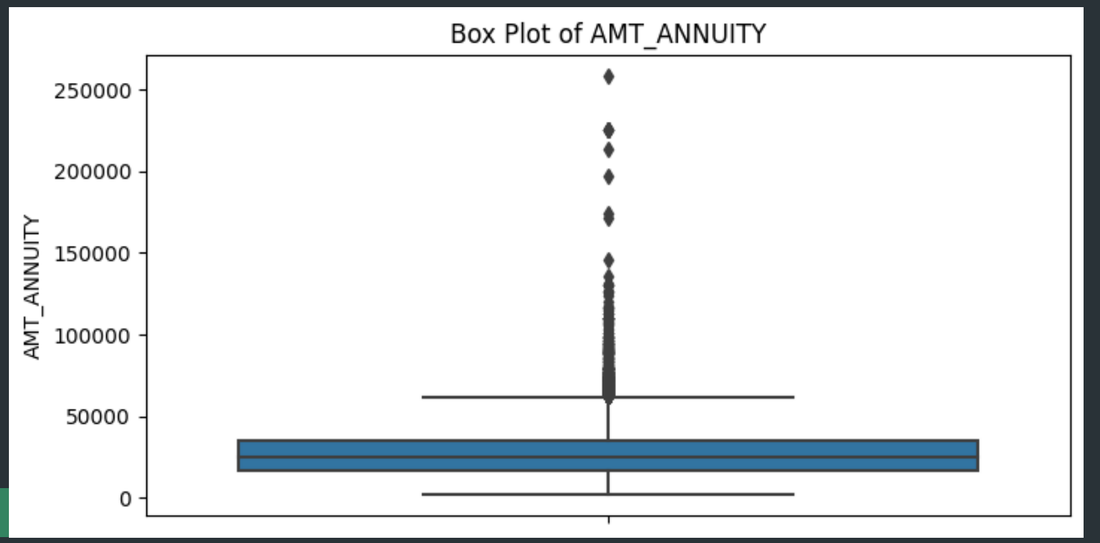
IDENTIFYING OUTLIERS

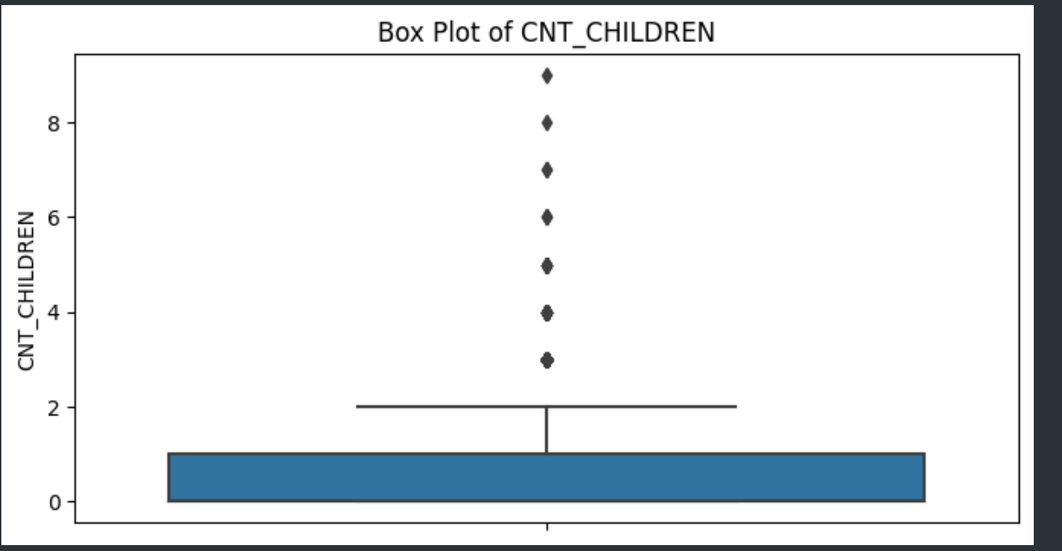
B)Task: Detect and identify outliers in the dataset using statistical functions and features, focusing on numerical variables.

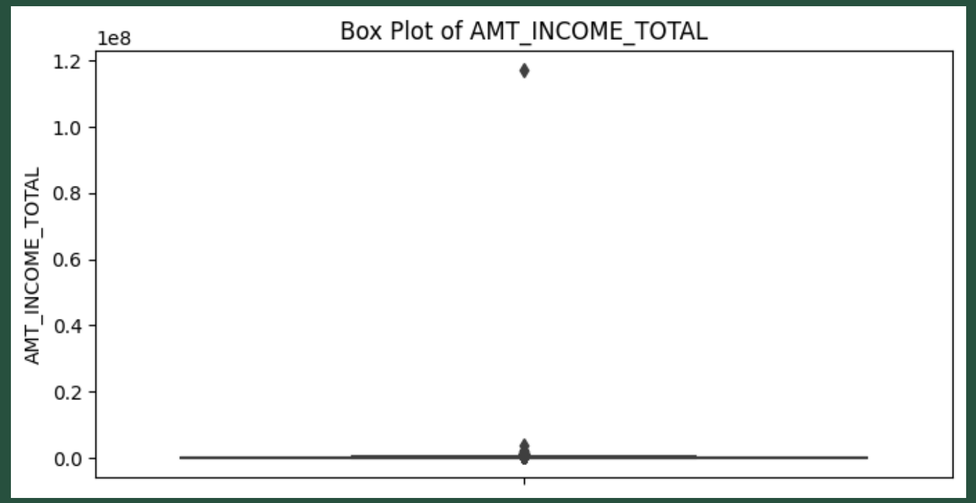
- Outliers are data points that are very different from the rest of the data in a group.
- They can be much higher or lower than the other data points.
- These outliers can mess up the math we use to understand the data, like the average or how spread out the data is.
- So, it's really important to find and deal with outliers properly to make sure our conclusions from the data are trustworthy and accurate.

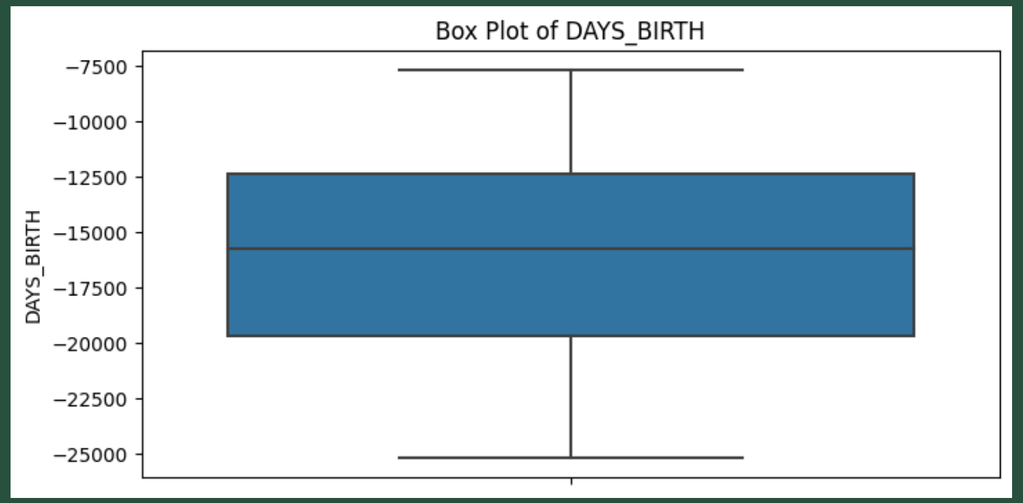

Box Plot of AMT_GOODS_PRICE
1e6
AMT_GOODS_PRICE
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0


Box Plot of AMT_CREDIT
1e6
AMT_CREDIT
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0


Box Plot of AMT_ANNUITY
AMT_ANNUITY
250000
200000
150000
100000
50000
0

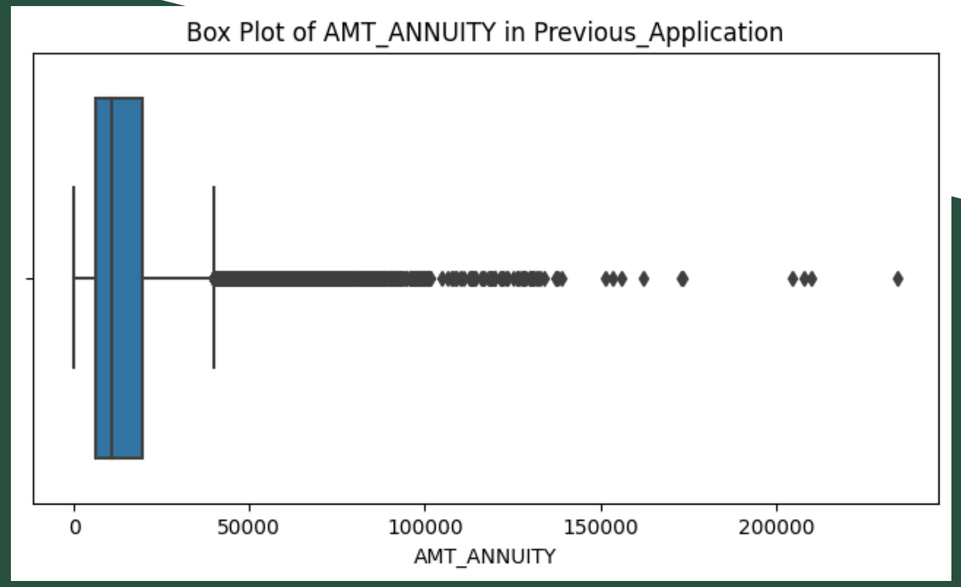

Box Plot of CNT_CHILDREN
CNT_CHILDREN
8
6
4
2
0

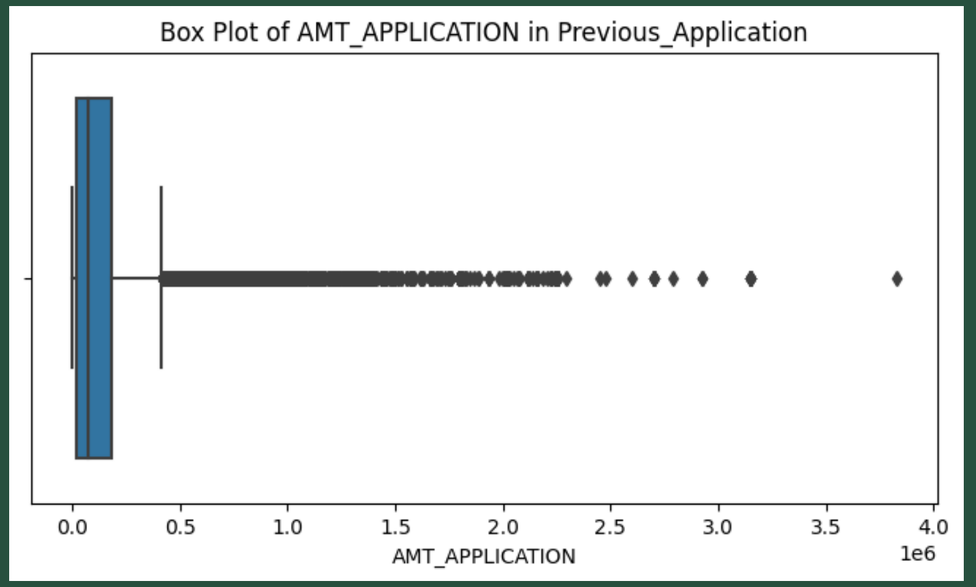


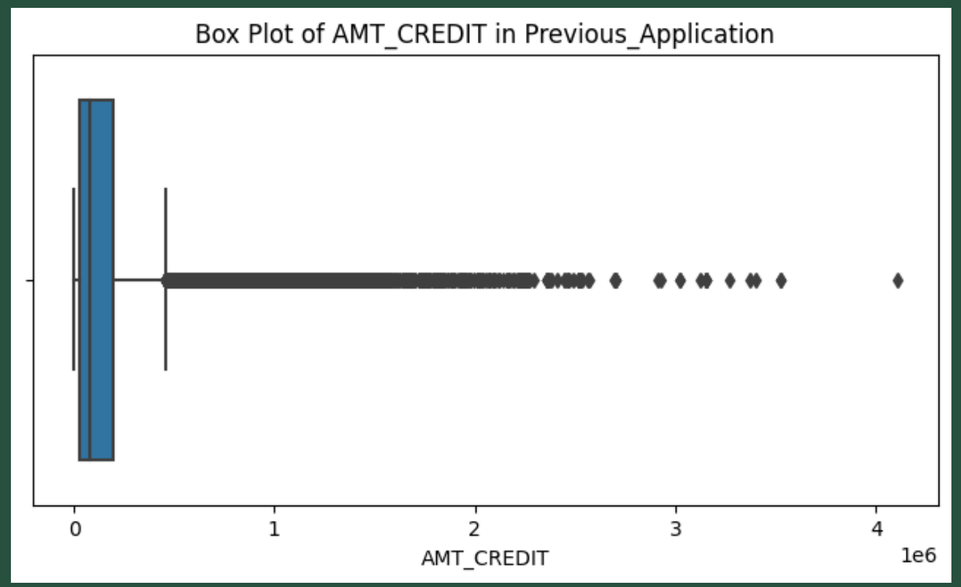


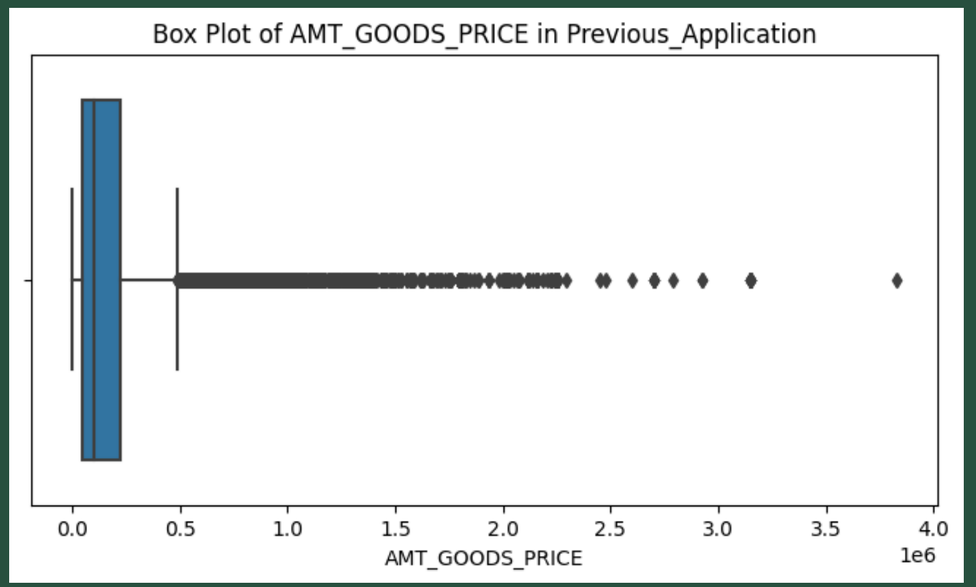
Upon reviewing the application data, the following observations are noted:

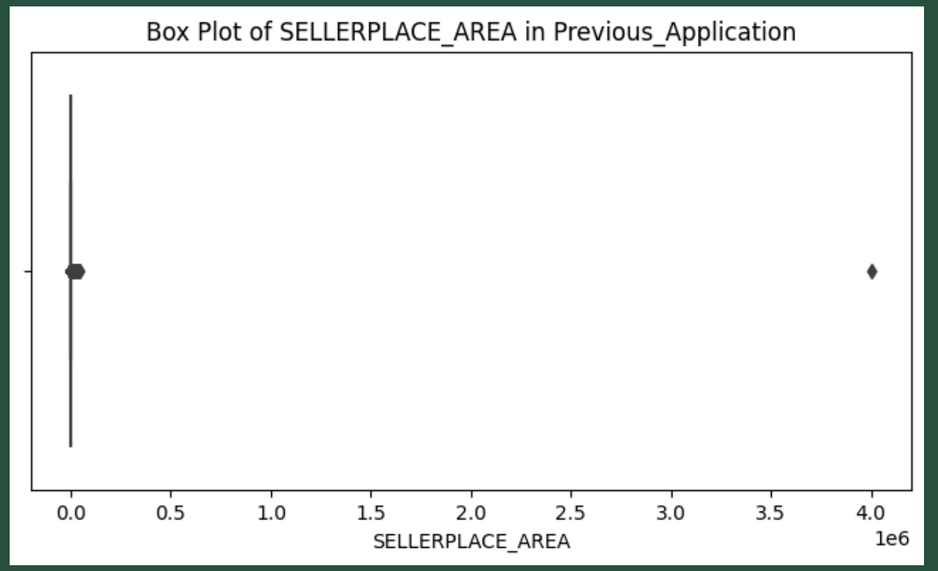
- AMT_ANNUITY, AMT_CREDIT, AMT_GOODS_PRICE, and CNT_CHILDREN have some values that are very different from the others. These differences might be unusual or extreme.
- AMT_INCOME_TOTAL has many values that are much higher than the rest. This suggests that some applicants have much higher incomes compared to most others.
- DAYS_BIRTH doesn't have any unusual values. It seems like the age data is reliable and doesn't have any extreme or incorrect entries.
- DAYS_EMPLOYED has some values that are extremely high, around 350,000 days, which is equivalent to roughly 958 years. This is impossible because people can't work for so long, so these entries are likely mistakes or errors in the data.

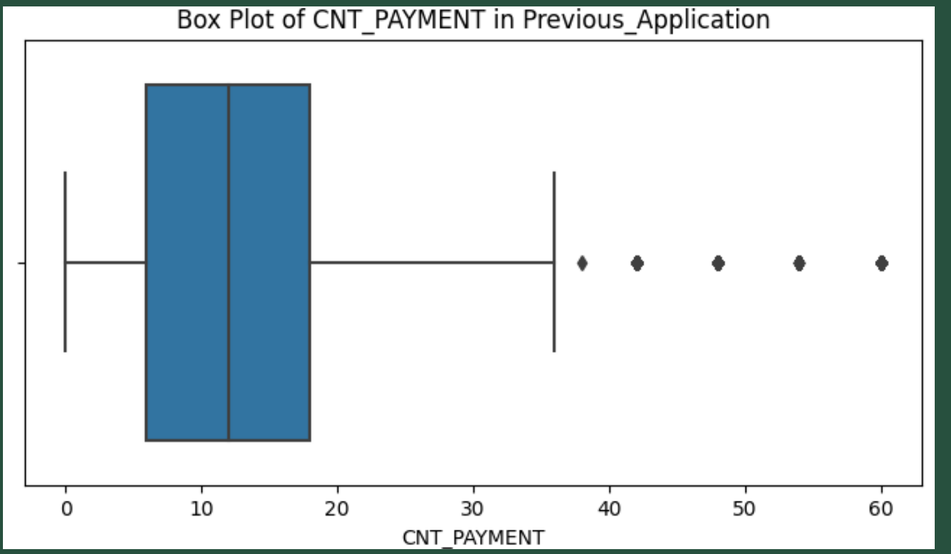

Box Plot of AMT_ANNUITY in Previous_Application
0
50000
100000
150000
200000
AMT_ANNUITY

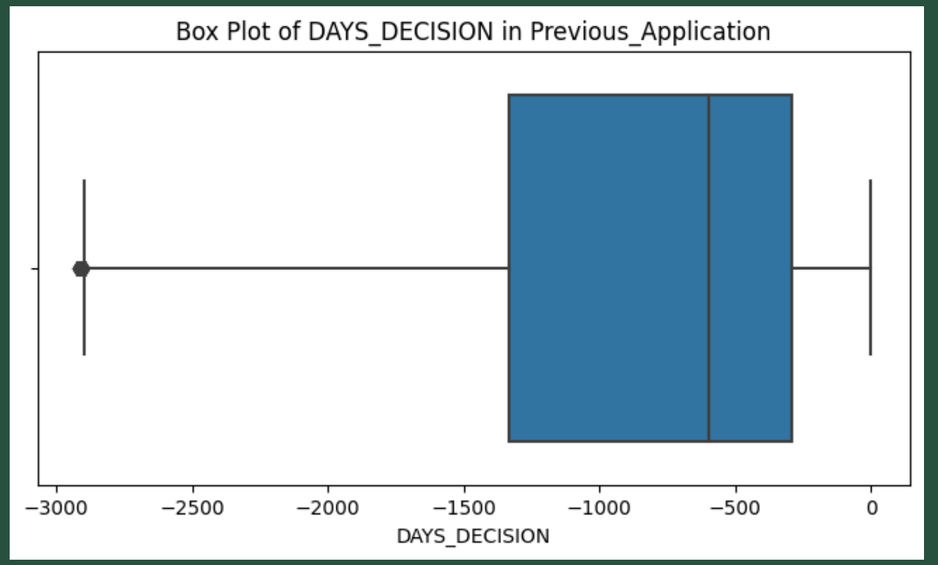

Box Plot of AMT_APPLICATION in Previous_Application
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
AMT_APPLICATION
1e6

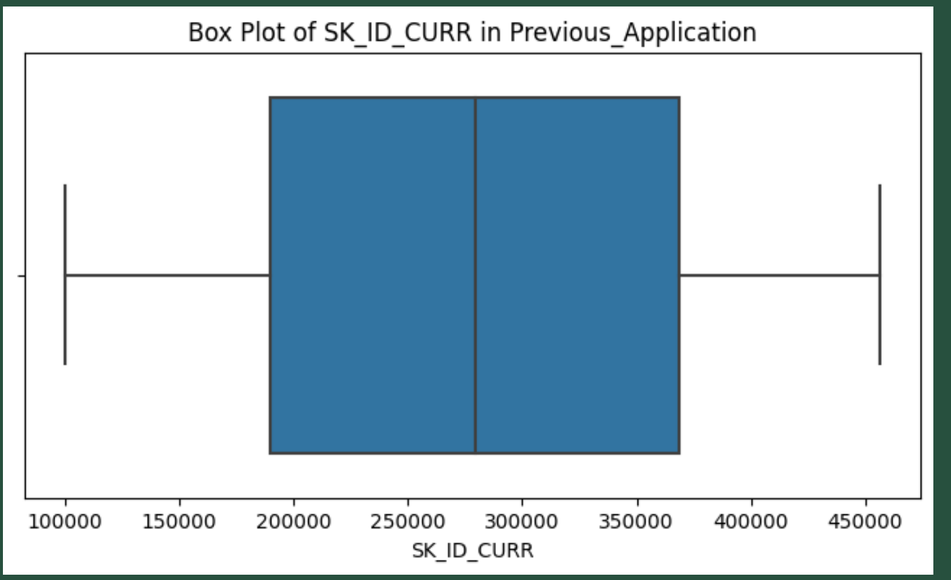

Box Plot of AMT_CREDIT in Previous_Application
0
1
2
3
4
AMT_CREDIT
1e6


Box Plot of AMT_GOODS_PRICE in Previous_Application
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
AMT_GOODS_PRICE
1e6


Box Plot of SELLERPLACE_AREA in Previous_Application
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
1e6
SELLERPLACE_AREA


Box Plot of CNT_PAYMENT in Previous_Application
0
10
20
30
40
50
60
CNT_PAYMENT

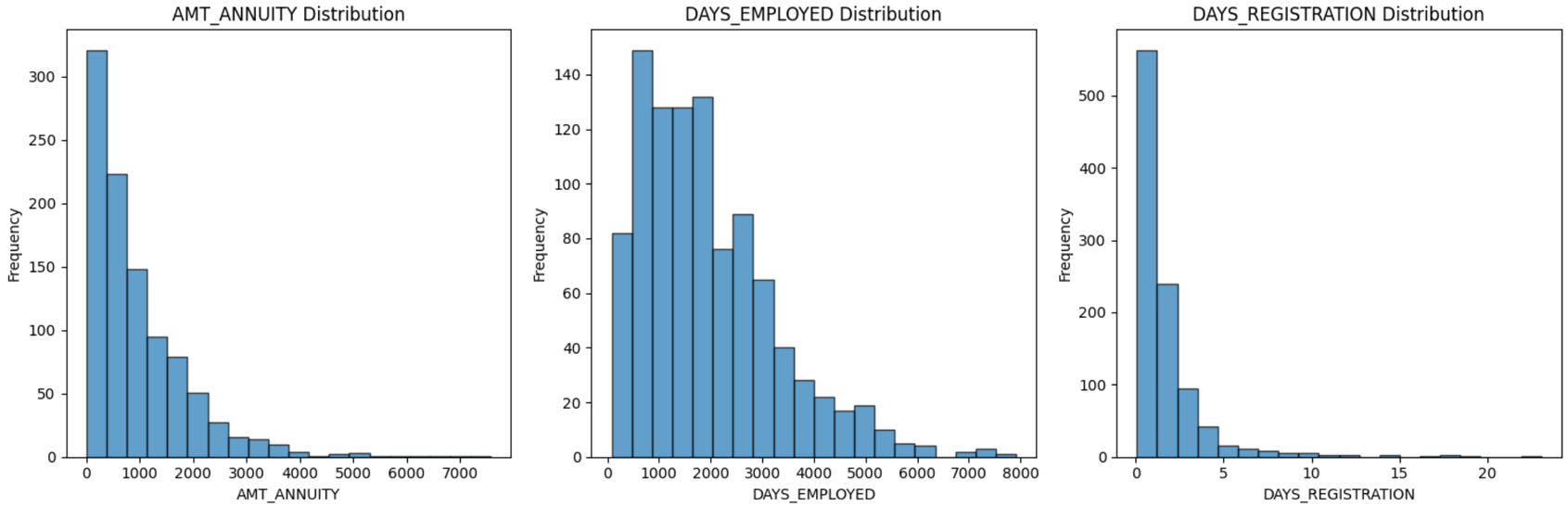

Box Plot of DAYS_DECISION in Previous_Application
−3000
−2500
−2000
−1500
−1000
−500
0
DAYS_DECISION


Box Plot of SK_ID_CURR in Previous_Application
100000
150000
200000
250000
300000
350000
400000
450000
SK_ID_CURR

The analysis of our previous application reveals the following

- AMT_ANNUITY, AMT_APPLICATION, AMT_CREDIT, AMT_GOODS_PRICE, and SELLERPLACE_AREA have many values that are significantly different from the rest. These differences might be unusual or extreme.
- CNT_PAYMENT has a few values that are different from the majority. This suggests that some cases have a different number of payments compared to most others.
- SK_ID_CURR, which is an ID column, doesn't have any unusual values. It's just an identification number, so there are no extreme or unusual entries.
- DAYS_DECISION has a few values that are different from the others, but not many. This indicates that decisions on previous applications were made quite a while ago for some cases,but it's not a widespread issue.

BIN CREATION AND DERIVED METRICS

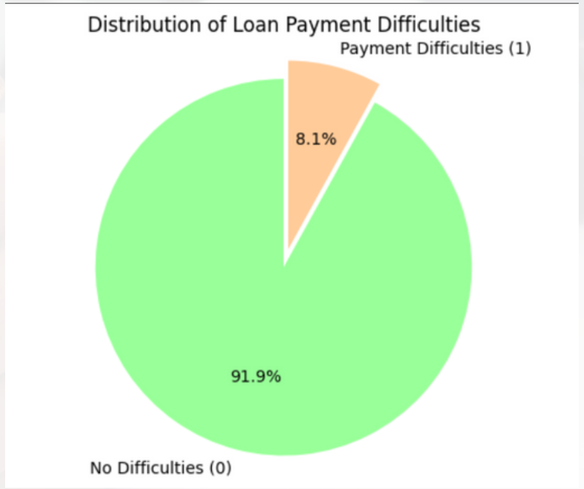


Looking at the numbers for AMT_ANNUITY, DAYS_EMPLOYED, and DAYS_REGISTRATION, we can see that the values in the lower 25% (the first quartile) are much smaller compared to the values in the upper 25% (the third quartile). This indicates that the data is mostly concentrated on the lower end, and there are relatively fewer higher values. In simpler terms, it means that most of the data falls into the lower range, and there are only a few data points with higher values.

ANALYZING DATA IMBALANCE

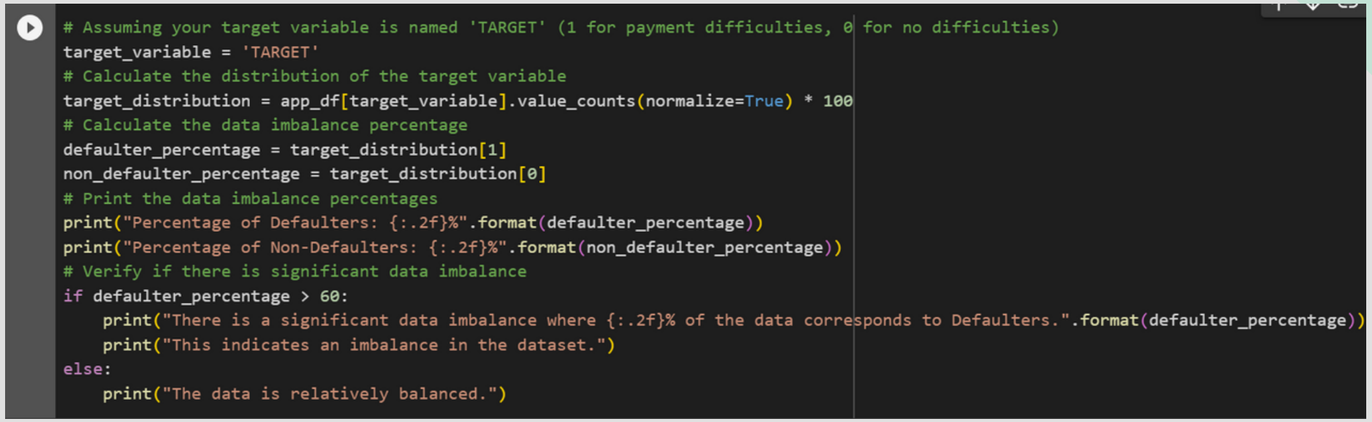
Task: Determine if there is data imbalance in the loan application dataset and calculate the ratio of data imbalance

```python
# Assuming your target variable is named 'TARGET' (1 for payment difficulties, 0 for no difficulties)
target_variable = 'TARGET'
# Calculate the distribution of the target variable
target_distribution = app_df[target_variable].value_counts()
# Calculate the ratio of data imbalance
imbalance_ratio = target_distribution[1] / target_distribution[0]
# Create a pie chart to visualize the distribution
labels = ['No Difficulties (0)', 'Payment Difficulties (1)']
sizes = target_distribution.values
colors = ['#ff9999','#66b3ff']
explode = (0.1, 0)  # explode 1st slice (Payment Difficulties)
fig, ax = plt.subplots()
ax.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
ax.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
# Display the pie chart
plt.title('Distribution of Loan Payment Difficulties')
plt.show()
# Print the distribution and imbalance ratio
print("Distribution of the Target Variable:")
print(target_distribution)
print("\nImbalance Ratio (1s to 0s): {:.2f}".format(imbalance_ratio))
```




```
Distribution of the Target Variable:
0    45937
1     4024
Name: TARGET, dtype: int64
```

The dataset has been split into two groups: one where people successfully paid back their loans (accounting for 91.93% of the dataset), and another where people couldn't pay back their loans (making up 8.07% of the dataset).



```python
# Assuming your target variable is named 'TARGET' (1 for payment difficulties, 0 for no difficulties)
target_variable = 'TARGET'
# Calculate the distribution of the target variable
target_distribution = app_df[target_variable].value_counts(normalize=True) * 100
# Calculate the data imbalance percentage
defaulter_percentage = target_distribution[1]
non_defaulter_percentage = target_distribution[0]
# Print the data imbalance percentages
print("Percentage of Defaulters: {:.2f}%".format(defaulter_percentage))
print("Percentage of Non-Defaulters: {:.2f}%".format(non_defaulter_percentage))
# Verify if there is significant data imbalance
if defaulter_percentage > 60:
    print("There is a significant data imbalance where {:.2f}% of the data corresponds to Defaulters.".format(defaulter_percentage))
    print("This indicates an imbalance in the dataset.")
else:
    print("The data is relatively balanced.")
```

```
Percentage of Defaulters: 8.05%
Percentage of Non-Defaulters: 91.95%
The data is relatively balanced.
```

Examines the distribution of a binary target variable in a dataset. If one class constitutes more than 60% of the data, it suggests a substantial imbalance, which can impact the reliability of predictive models. This highlights the importance of addressing data imbalance for more accurate modeling and better outcomes. Analyzing class distribution is a crucial first step in machine learning projects.

PERFORM UNIVARIATE

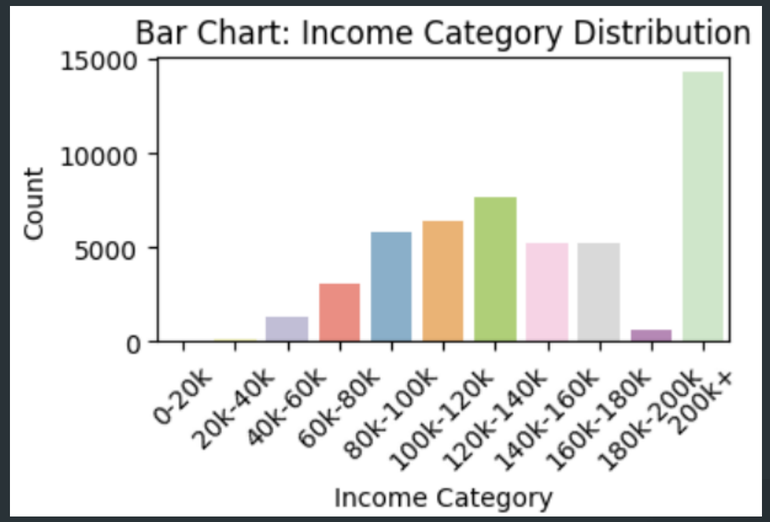
Task D : Involves conducting univariate analysis to gain a comprehensive understanding of the distribution of individual variables. Additionally, the analysis should be segmented to compare variable distributions for different scenarios. Lastly, bivariate analysis should be performed to explore the relationships between variables and the target variable.

Univariate Analysis: Studying one variable at a time to find patterns and insights.

Segmented Univariate Analysis: Analyzing individual variables within distinct groups or segments of data.

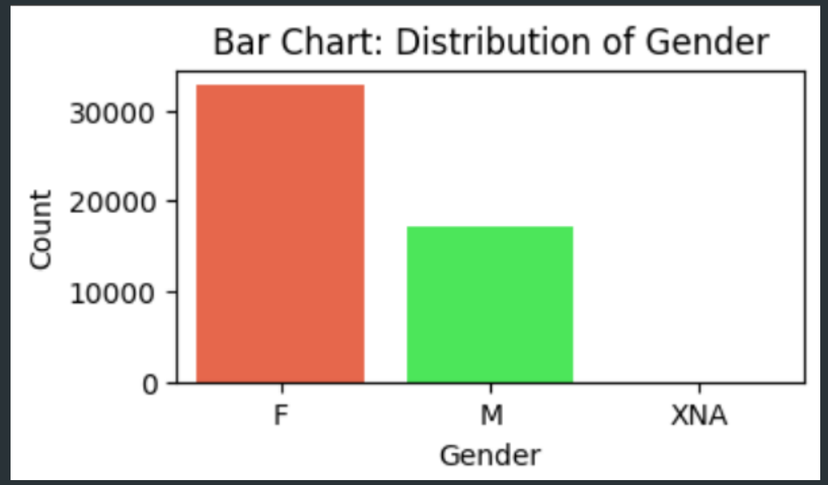
Bivariate Analysis: Examining two variables together to uncover relationships and patterns.

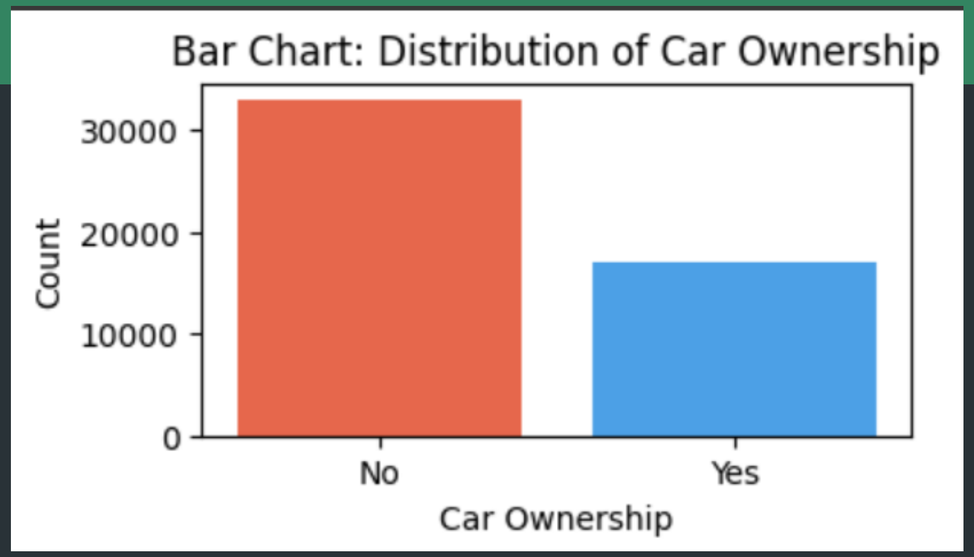
UNIVARIATE ANALYSIS



The graph shows that most people earn lower incomes, and only a smaller group of people earn higher incomes. The middle income, where half of the people earn more and half earn less, is somewhere between $60,000 and $80,000.

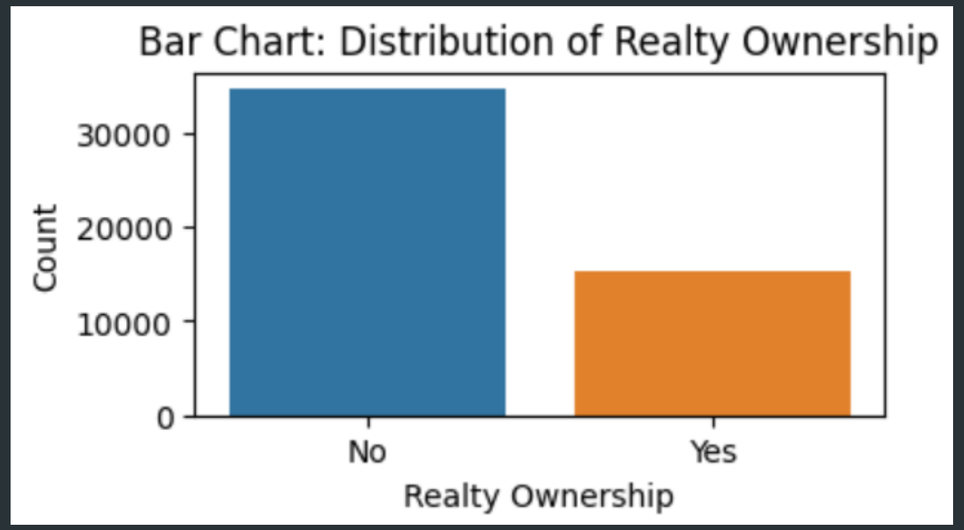
The graph shows that females make up 52% of the population, males are at 46%, and non-binary individuals account for 2%. The gender gap is small, with only a 6% difference. Non-binary numbers are slowly growing. Keep in mind that gender distribution might differ in specific groups like the workforce compared to the overall population.

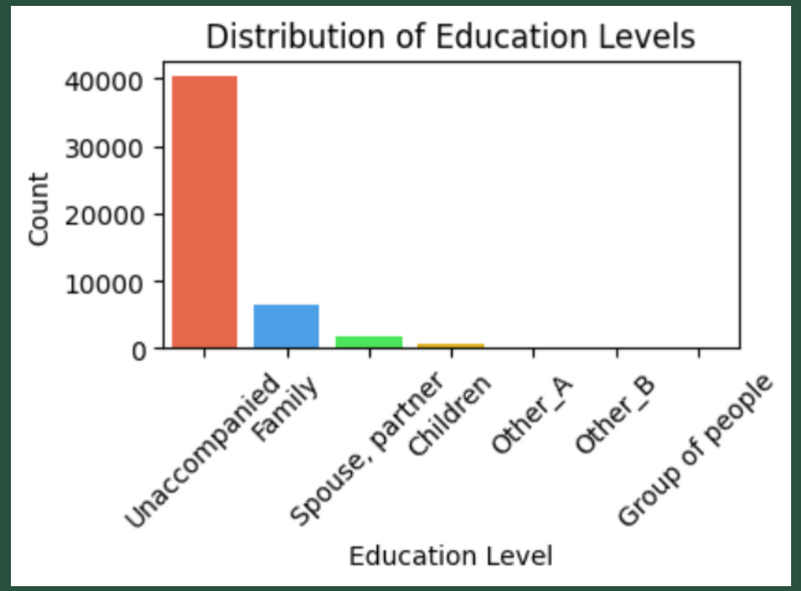




Most people in the United States have their own cars. People who make more than $100,000 a year are more likely to own cars, while those who make less than $20,000 a year are less likely to have one. The gap in car ownership between these two income groups is quite big.

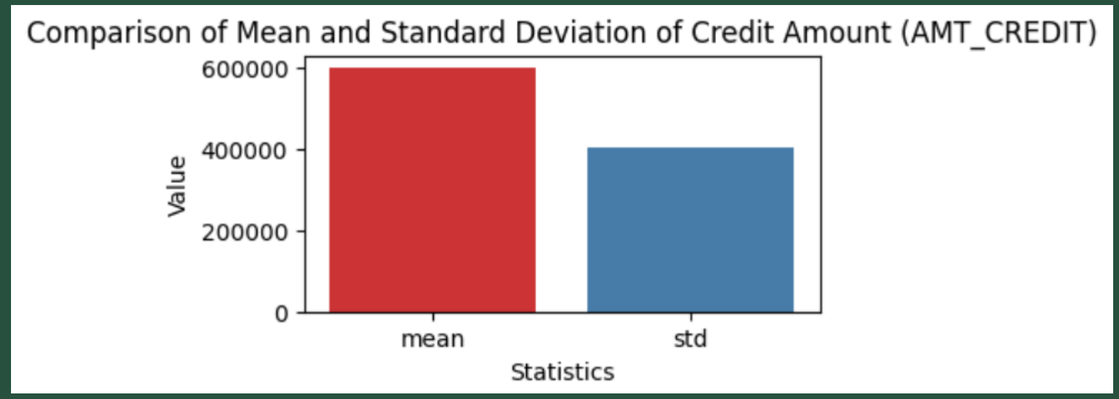
Look at the graph! It shows that 62% of people haven't bought any real estate, while only 38% of the population has managed to do so.



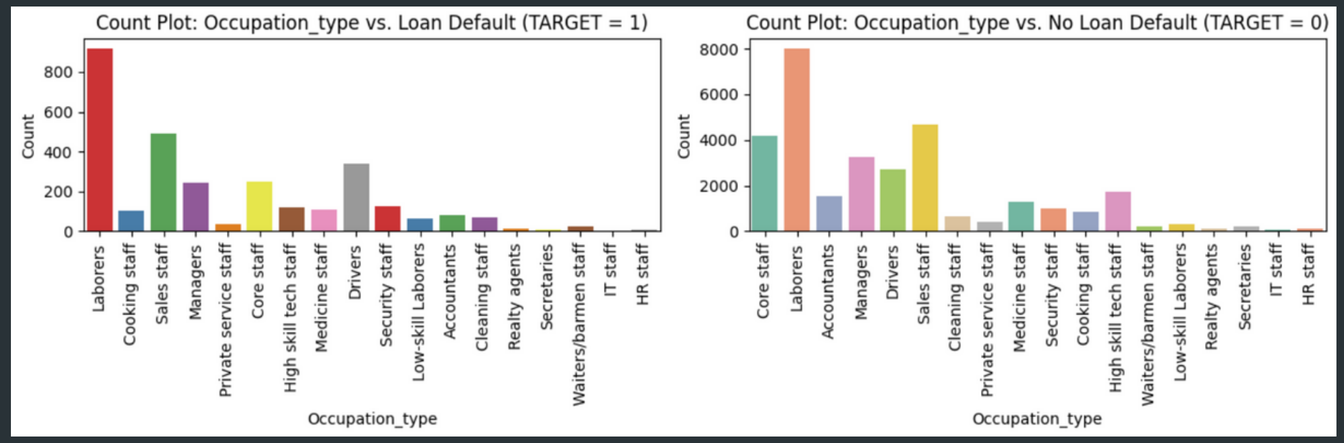


There's a big difference in the education levels of unaccompanied minors and family units. About 55% of unaccompanied minors haven't finished high school, while 70% of family units have at least a high school diploma. When it comes to higher education, only 17% of unaccompanied minors have gone to college, while 35% of family units have done so.

Males typically borrow more than females, with a notable gap, especially for larger amounts. The average borrowing amount is higher than the typical difference in borrowing amounts.

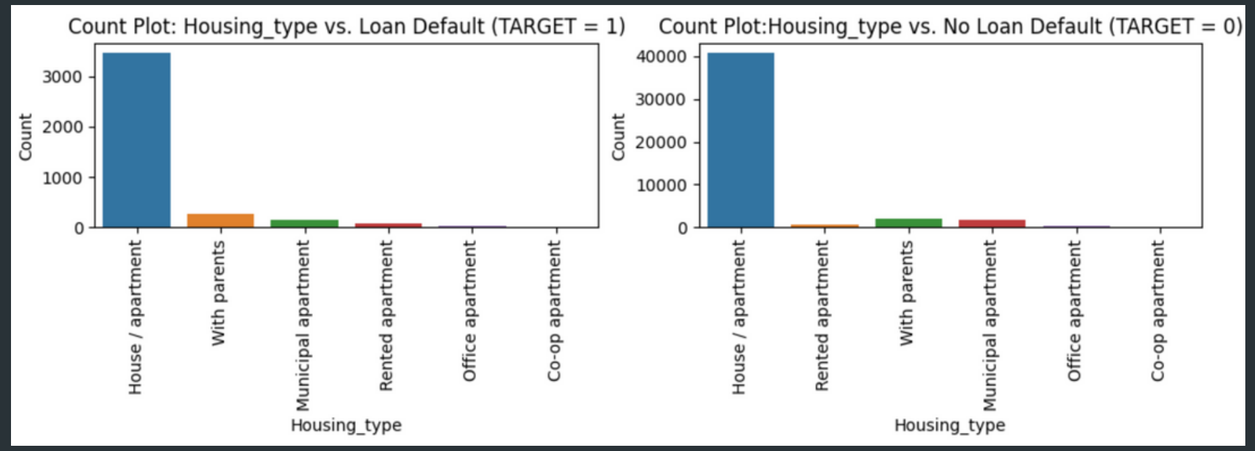


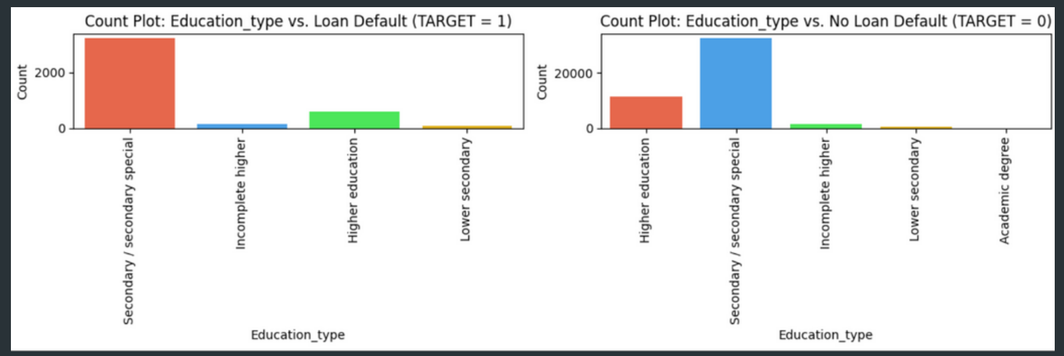
SEGMENTED UNIVARIATE ANALYSIS



- Waiters/barmen staff: Customer conflicts leading to legal issues.
- Low-skill laborers: Disputes related to wages and working hours.
- Realty agents: Real estate transaction disputes.
- Secretaries: Employment contract disagreements.
- IT staff: Intellectual property disputes.

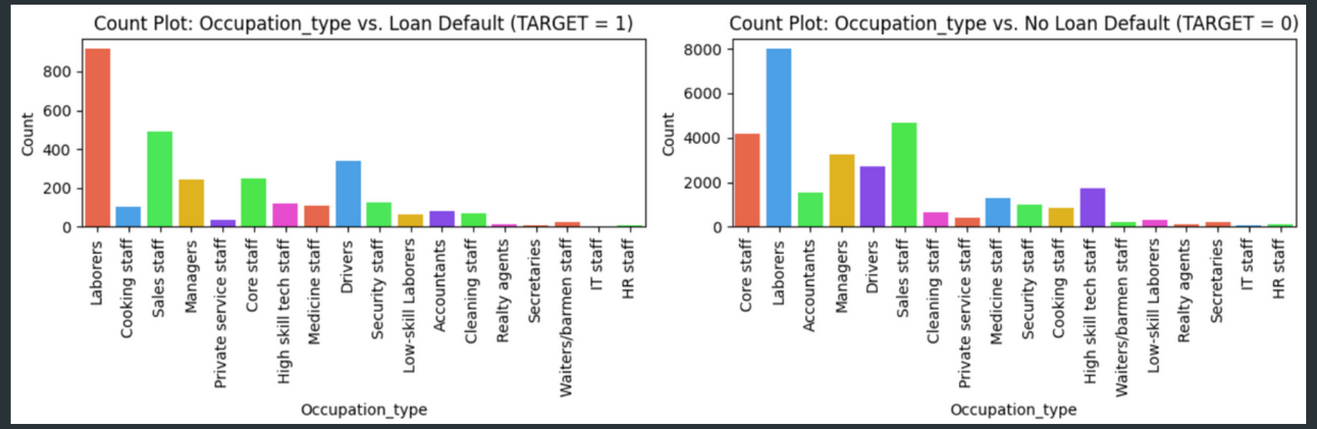
Factory-made and movable homes have higher loan default rates, possibly because they are cheaper and located in rural areas with limited job opportunities, impacting loan approvals.





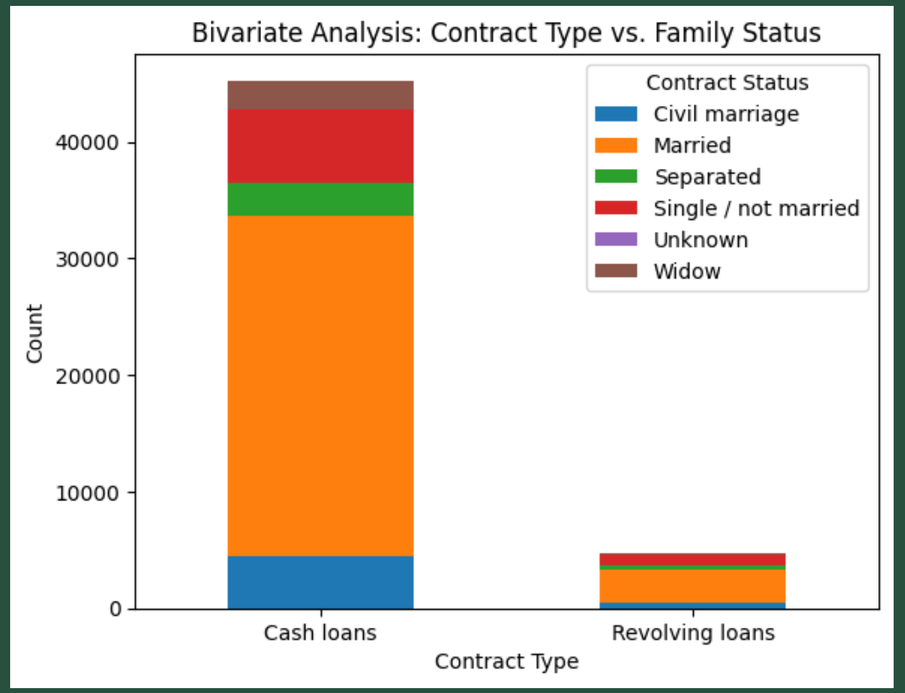
In developed countries with strong educational values and government support, there are more people with higher education than those with lower education.

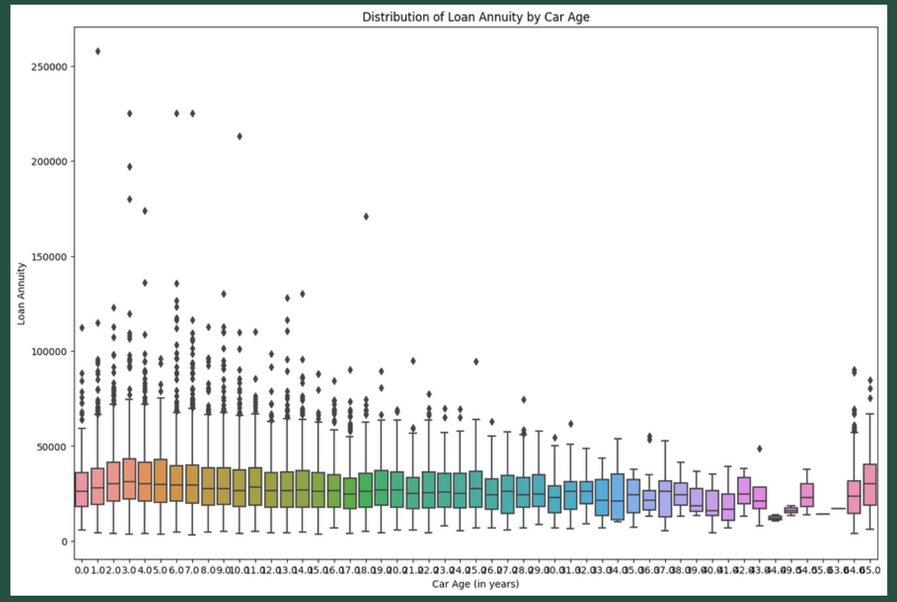
Occupations requiring high education, responsibility, and demand tend to have the highest median incomes.



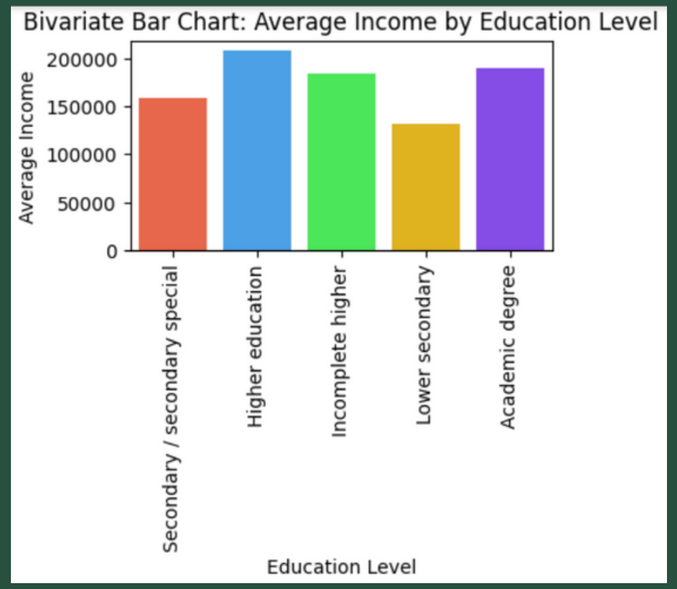
BIVARIATE ANALYSIS

Married couples are more likely to take out and successfully repay cash loans compared to other individuals in the distribution, while widows are more numerous in the group.



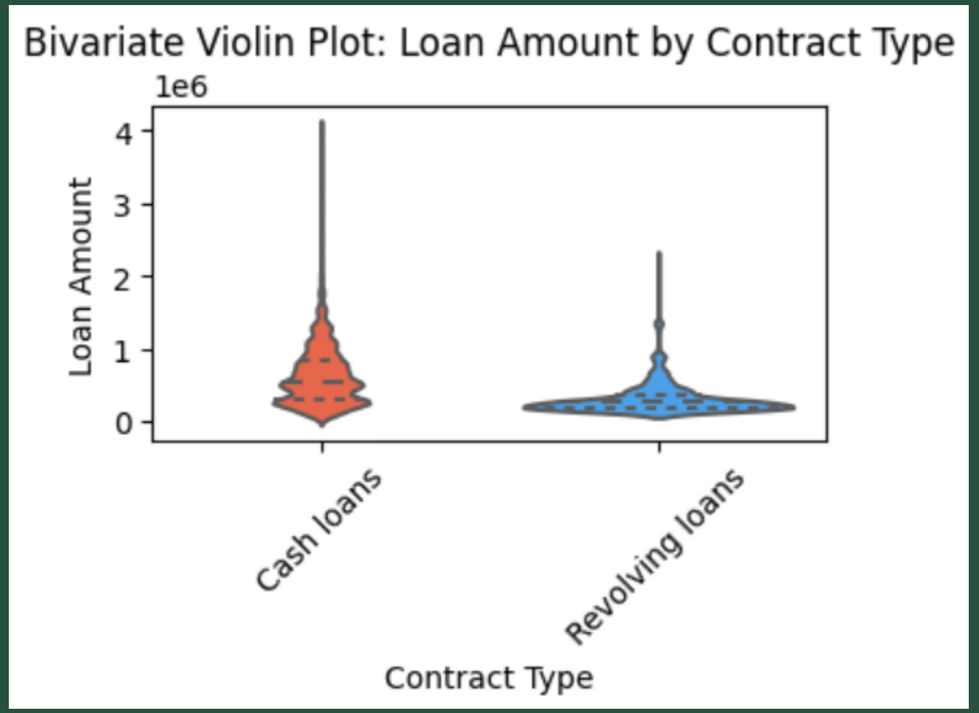


Countries with better healthcare, nutrition, and economic progress usually have people who live longer on average.



People with more education usually make more money than those with less education

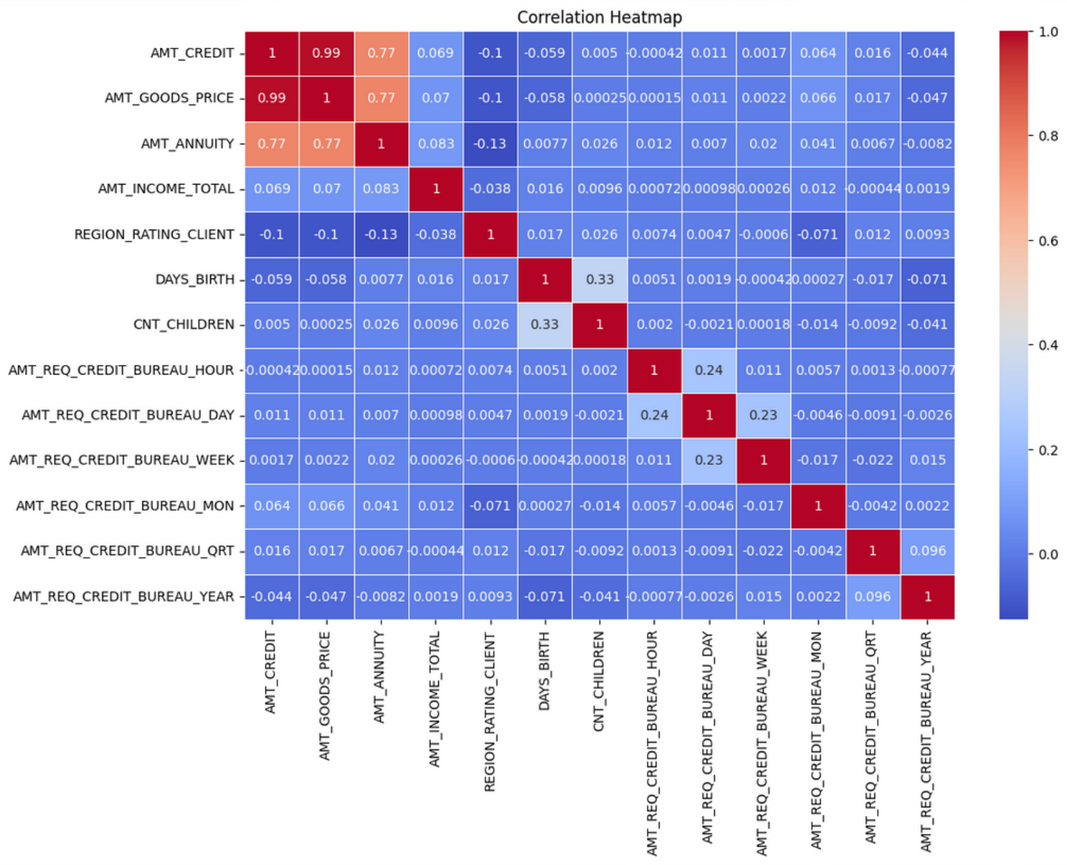
Cash loans typically provide a broader range of loan amounts compared to revolving loans. This variation is influenced by factors like the borrower's creditworthiness, the purpose of the loan, and the risk perceived by the lender.



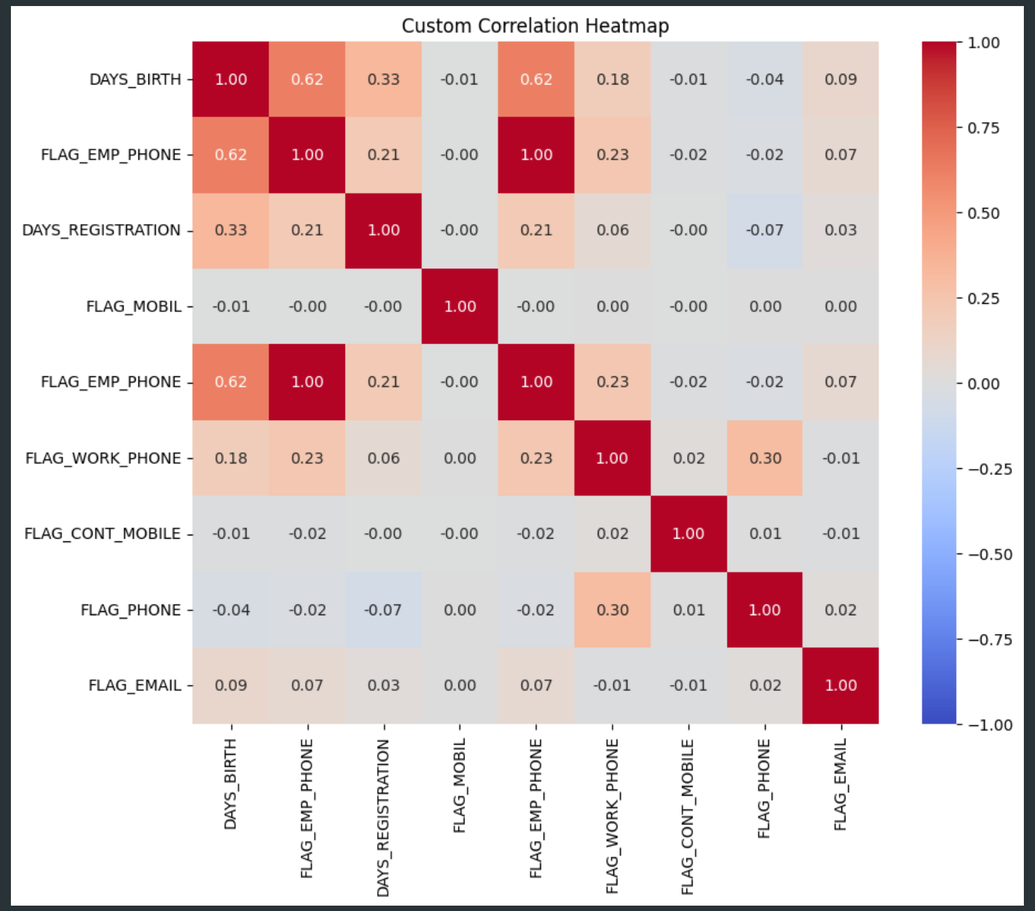
IDENTIFY CORRELATIONS

Task E: Segment the dataset based on different scenarios (e.g., clients with payment difficulties and all other cases) and identify the top correlations for each segmented

CORRELATION



- The heatmap reveals that certain variables strongly relate to credit risk, including income, credit score, and debt-to-income ratio.
- Lenders can use this data to spot borrowers at a higher risk of not repaying their loans.
- This helps them make better lending choices and set suitable interest rates.
- Factors like higher income, better credit scores, lower debt-to-income ratios, smaller loan amounts, and shorter loan terms are linked to lower credit risk.




```
Top 10 Correlations for Target 0:
NONLIVINGAPARTMENTS_MEDI   NONLIVINGAPARTMENTS_AVG     0.999378
NONLIVINGAPARTMENTS_AVG    NONLIVINGAPARTMENTS_MEDI    0.999378
LIVINGAPARTMENTS_MEDI      LIVINGAPARTMENTS_AVG        0.998984
LIVINGAPARTMENTS_AVG       LIVINGAPARTMENTS_MEDI       0.998984
LANDAREA_AVG               LANDAREA_MEDI               0.998972
LANDAREA_MEDI              LANDAREA_AVG                0.998972
YEARS_BUILD_MEDI           YEARS_BUILD_AVG             0.998814
YEARS_BUILD_AVG            YEARS_BUILD_MEDI            0.998814
COMMONAREA_AVG             COMMONAREA_MEDI             0.998792
COMMONAREA_MEDI            COMMONAREA_AVG              0.998792
dtype: float64

Top 10 Correlations for Target 1:
NONLIVINGAPARTMENTS_MEDI   NONLIVINGAPARTMENTS_AVG     0.999378
NONLIVINGAPARTMENTS_AVG    NONLIVINGAPARTMENTS_MEDI    0.999378
LIVINGAPARTMENTS_MEDI      LIVINGAPARTMENTS_AVG        0.998984
LIVINGAPARTMENTS_AVG       LIVINGAPARTMENTS_MEDI       0.998984
LANDAREA_AVG               LANDAREA_MEDI               0.998972
LANDAREA_MEDI              LANDAREA_AVG                0.998972
YEARS_BUILD_MEDI           YEARS_BUILD_AVG             0.998814
YEARS_BUILD_AVG            YEARS_BUILD_MEDI            0.998814
COMMONAREA_AVG             COMMONAREA_MEDI             0.998792
COMMONAREA_MEDI            COMMONAREA_AVG              0.998792
dtype: float64
```

- There's a notable connection between an employee's age and their phone number – as age goes up, the likelihood of having a phone number decreases.
- On the other hand, there's a positive link between the number of days before a client's registration change and their age.
- This suggests that older people are less likely to change their registration details before applying for a loan.
- Additionally, clients who don't include their phone numbers are also less likely to provide incorrect permanent and work addresses.

CONCLUSION

- After carefully analyzing and cleaning both the application dataset and the previous application dataset, we've discovered some important insights that can help banks make better decisions about loan applications and default risk.
- Firstly, it's clear that lending to students, retirees, and people with higher education tends to be less risky in terms of loan defaults. So, banks can confidently approve loans for these groups.
- On the flip side, certain occupations like laborers, sales staff, drivers, cleaning workers, and low-skilled workers have a higher likelihood of not paying back their loans. To reduce risk, it's advisable to focus more on clients in stable professions like managers, core staff, and highly skilled technical workers.

RESULTS

- I applied EDA in a real business case and learned about risk analytics in finance, data summarization, correlation analysis, dealing with data challenges, and data visualization.
- The project was challenging but helped me gain valuable skills and insights for the client.

COLAB NOTEBOOK LINK

https://colab.research.google.com/drive/1sLDyAIV_pdpDSSw4xH5drnFtJLVaAY4X?usp=sharing

ENTIRE PROJECT LINK

https://drive.google.com/drive/folders/1Q_FIa6dd68La_jN0HwEWP6L-oaz1OCnH?usp=drive_link

THANK YOU!