Bank loan Case study

Project description:

This case study aims to illustrate how EDA can be used in a practical corporate setting. We will discover the fundamentals of risk analytics in banking and financial services in this case study, as well as how data is used to lower the risk of losing money when lending to consumers, in addition to using the techniques you acquired in the EDA module.

Three datasets with information about bank loans are provided to us for our final project. Due to their weak or non-existent credit histories, banks that provide loans find it difficult to grant loans to individuals. Due to this, some customers take advantage of it by getting insolvent. Let's say we work for a consumer finance business that specializes in providing urban customers with different kinds of loans. To examine the patterns found in the data, we must use EDA. By doing this, it will be ensured that only those applicants who can repay the loan will be accepted.

Bank wants to understand the driving factors (or driver variables) behind loan default, i.e. the variables which are strong indicators of default. The bank can utilize this knowledge for its portfolio and risk assessment.

There are four possible actions the business could take in response to a loan application from a client:

1. Approved: The loan application has been accepted by the business.

2. Cancelled: The client canceled the application during the approval process. Either the customer changed their mind about the loan, or in other situations; a larger risk resulted in the customer receiving an unfavorable price.

3. Rejected: The business turned down the loan (because the client didn't comply with their conditions, etc.).

4. Unused Offer: The client has canceled the loan, albeit it is still in the approval process.

Datasets:

1. ‘application_data.csv' contains all the information of the client at the time of application.

The data is about whether an applicant has payment difficulties.

2. ‘previous_application.csv’ contains information about the applicant's previous loan data. It contains the data on whether the previous application had been **Approved, Cancelled, Refused or Unused** offer.

3. columns_description.csv’ is a data dictionary that elaborates the meaning of the variables.

Approach:

The current and prior applications are two big data sets used in this case study. Each contained many blank data fields as well as several unnecessary columns that were useless for risk evaluations. I cleaned up first. I cleaned the data, found some outliers, and eliminated them before using pivot tables and charts to perform univariate and bivariate analysis to examine this massive data set.

The technology stack used:

- MySQL Workbench 8.0 CE,
- Microsoft Excel 2010
- Python Programming language
- Google Collab

Comment:

- prev_ap_df (previous_application.csv')contains 37 features and 1670214 rows

(Out of which 15 features are float64, 6 features are integers and 16 features are object datatype)

- applications_df (application_data.csv) contains 121 features, 1 target variable, and 307511 rows

(Out of which 65 features are float64, 41 features are integers and 16 features are object datatype)

- SK_ID_CURR" is a unique identifier, which can be used to merge the relevant columns of 2 dataframes.

Cleaning the data

First, I Checked whether there are missing values available or not.

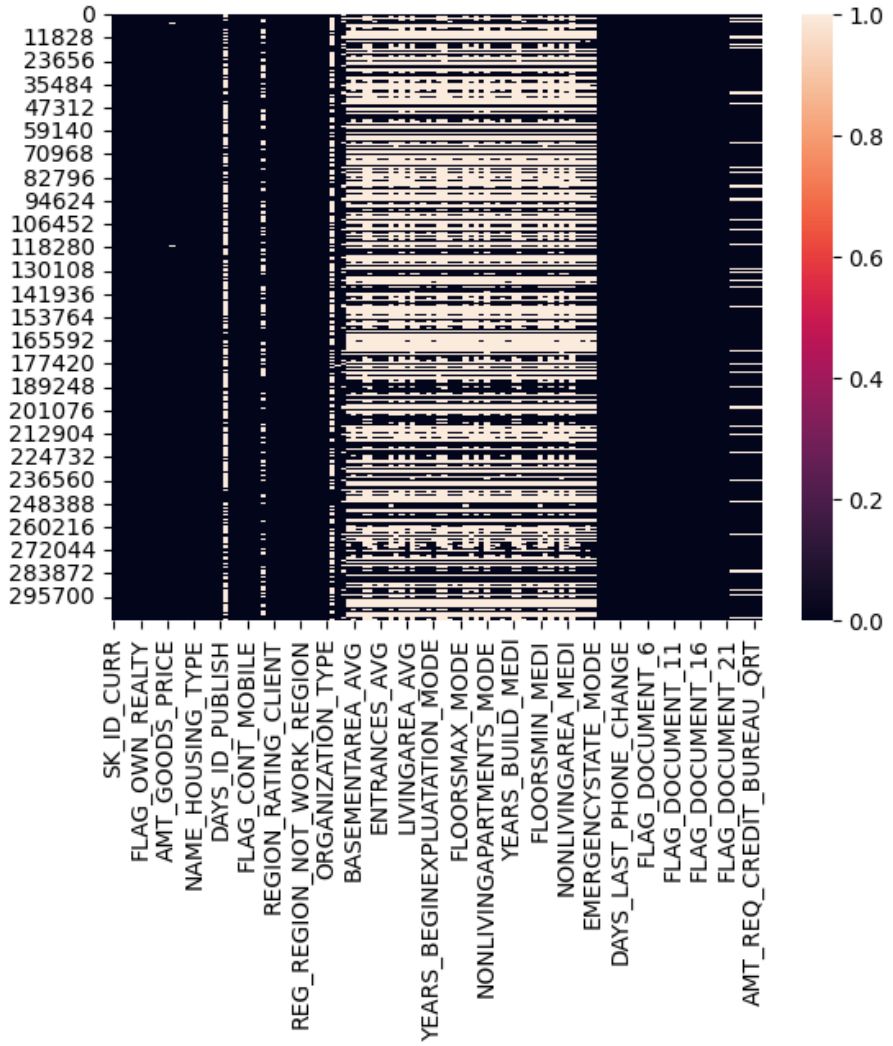
```
# Check the missing values are avalable or not

print ("Any missing value?",prev_ap_df.isnull().values.any())
```

```
Any missing value? True
```

Then I checked the percentage of missing values in the given dataset.

```
[ ] missingdata_prev = missingdata_percentage(prev_ap_df)

missingdata_prev.sort_values('percentage', ascending = False)
```

	category	percentage
5	RATE_INTEREST_PRIMARY	99.643698
6	RATE_INTEREST_PRIVILEGED	99.643698
2	AMT_DOWN_PAYMENT	53.636480
4	RATE_DOWN_PAYMENT	53.636480
7	NAME_TYPE_SUITE	49.119754
10	DAYS_FIRST_DRAWING	40.298129
11	DAYS_FIRST_DUE	40.298129
12	DAYS_LAST_DUE_1ST_VERSION	40.298129
13	DAYS_LAST_DUE	40.298129
14	DAYS_TERMINATION	40.298129
15	NFLAG_INSURED_ON_APPROVAL	40.298129
3	AMT_GOODS_PRICE	23.081773
0	AMT_ANNUITY	22.286665
8	CNT_PAYMENT	22.286366
9	PRODUCT_COMBINATION	0.020716
1	AMT_CREDIT	0.000060



Then I checked the info of the dataset given using the. Info command.

```
[ ]  prev_num_df = pd.DataFrame()

     for col in numeric_features:
       prev_num_df[col] = prev_ap_df[col]

     prev_num_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1669867 entries, 0 to 1670213
Data columns (total 19 columns):
 #   Column                     Non-Null Count    Dtype
---  ------                     --------------    -----
 0   SK_ID_PREV                 1669867 non-null  int64
 1   SK_ID_CURR                 1669867 non-null  int64
 2   AMT_ANNUITY                1297978 non-null  float64
 3   AMT_APPLICATION            1669867 non-null  float64
 4   AMT_CREDIT                 1669867 non-null  float64
 5   AMT_DOWN_PAYMENT           774370 non-null   float64
 6   AMT_GOODS_PRICE            1284699 non-null  float64
 7   HOUR_APPR_PROCESS_START    1669867 non-null  int64
 8   NFLAG_LAST_APPL_IN_DAY     1669867 non-null  int64
 9   RATE_DOWN_PAYMENT          774370 non-null   float64
 10  DAYS_DECISION              1669867 non-null  int64
 11  SELLERPLACE_AREA           1669867 non-null  int64
 12  CNT_PAYMENT                1297983 non-null  float64
 13  DAYS_FIRST_DRAWING         997149 non-null   float64
 14  DAYS_FIRST_DUE             997149 non-null   float64
 15  DAYS_LAST_DUE_1ST_VERSION  997149 non-null   float64
 16  DAYS_LAST_DUE              997149 non-null   float64
 17  DAYS_TERMINATION           997149 non-null   float64
 18  NFLAG_INSURED_ON_APPROVAL  997149 non-null   float64
dtypes: float64(13), int64(6)
memory usage: 254.8 MB
```

Comments:

There are 16 features in prev_app_of that have missing values.

- Permanently dropping the features (RATE_INTEREST_PRIMARY and RATE_INTEREST_PRIVILEGED) as 99% data is missing.
- Dropping rows containing missing values for the features (AMT_CREDIT and PRODUCT_COMBINATION) for very low % of missing data. Dropping entries would not cause impact the analysis as the percentage of missing values is very low (-2%).

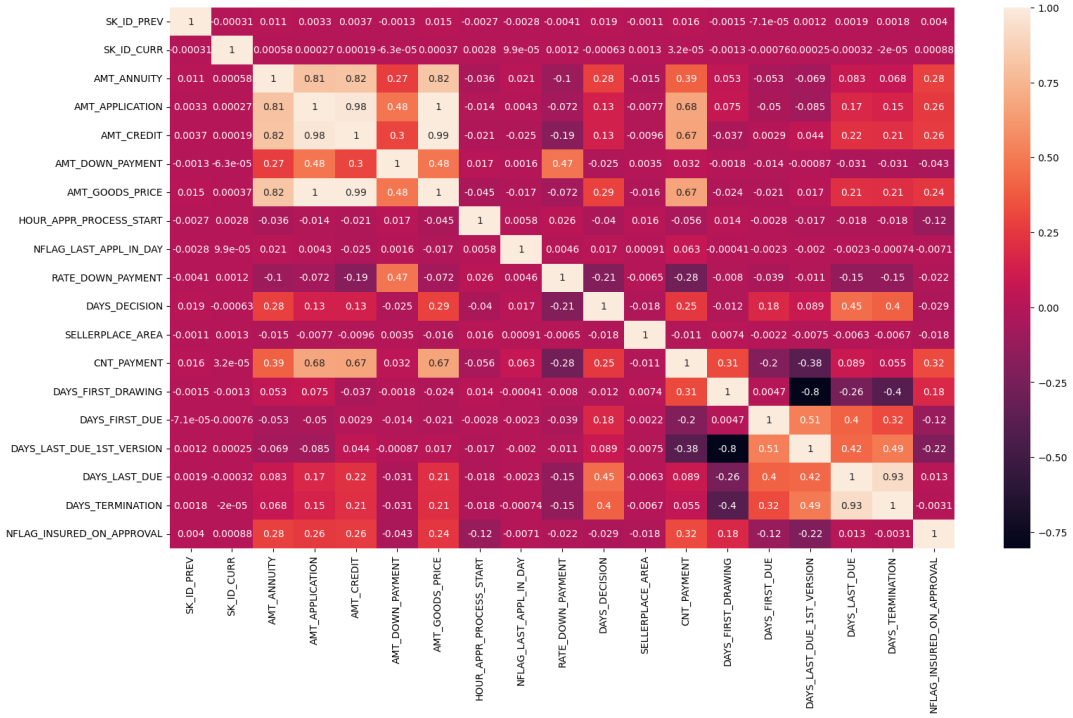
Then I performed correlation studies of the given columns.

Comment:

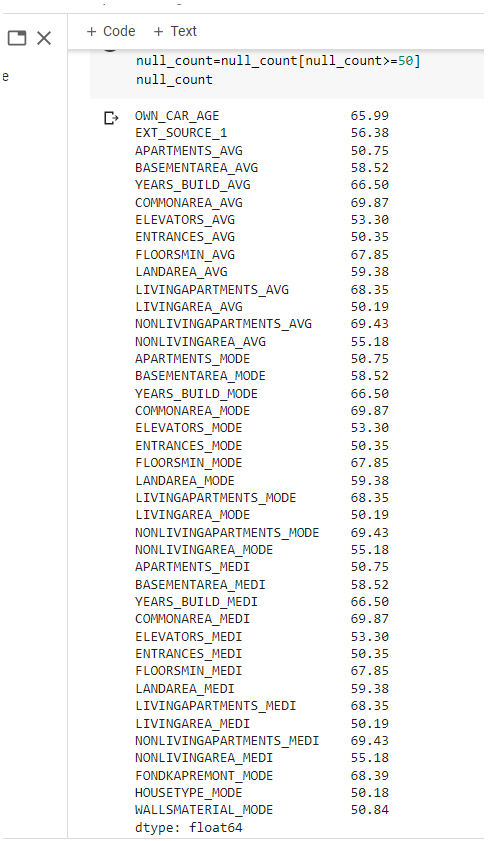
- DAYS_LAST_DUE and ‘DAYS_TERMINATION are highty correlated
- DAYS_FIRST_DRAWING and DAYS_LAST_DUE_tst_VERSION have high negative correlation
- 'AMT_ANNUITY', 'AMT_APPICATION, 'AMT_CREDIT', 'AMT_GOODS_PRICE' are highly correlated

The features can be removed before modeling ths data. 25 they would cause colinearty "DAYS_TERMINATION DAYS_LAST_DUE_1st_VERSION", "AMT_APPLICATION AMT_CREDIT","AMT_GOODS_PRICE" For EDA purpose we are not removing them.

- SK ID_PREV columns are not required for analysis.



Then I dropped columns having missing values more than 50%.



```
+ Code + Text

null_count=null_count[null_count>=50]
null_count
```

```
OWN_CAR_AGE                 65.99
EXT_SOURCE_1                56.38
APARTMENTS_AVG              50.75
BASEMENTAREA_AVG            58.52
YEARS_BUILD_AVG             66.50
COMMONAREA_AVG              69.87
ELEVATORS_AVG               53.30
ENTRANCES_AVG               50.35
FLOORSMIN_AVG               67.85
LANDAREA_AVG                59.38
LIVINGAPARTMENTS_AVG        68.35
LIVINGAREA_AVG              50.19
NONLIVINGAPARTMENTS_AVG     69.43
NONLIVINGAREA_AVG           55.18
APARTMENTS_MODE             50.75
BASEMENTAREA_MODE           58.52
YEARS_BUILD_MODE            66.50
COMMONAREA_MODE             69.87
ELEVATORS_MODE              53.30
ENTRANCES_MODE              50.35
FLOORSMIN_MODE              67.85
LANDAREA_MODE               59.38
LIVINGAPARTMENTS_MODE       68.35
LIVINGAREA_MODE             50.19
NONLIVINGAPARTMENTS_MODE    69.43
NONLIVINGAREA_MODE          55.18
APARTMENTS_MEDI             50.75
BASEMENTAREA_MEDI           58.52
YEARS_BUILD_MEDI            66.50
COMMONAREA_MEDI             69.87
ELEVATORS_MEDI              53.30
ENTRANCES_MEDI              50.35
FLOORSMIN_MEDI              67.85
LANDAREA_MEDI               59.38
LIVINGAPARTMENTS_MEDI       68.35
LIVINGAREA_MEDI             50.19
NONLIVINGAPARTMENTS_MEDI    69.43
NONLIVINGAREA_MEDI          55.18
FONDKAPREMONT_MODE          68.39
HOUSETYPE_MODE              50.18
WALLSMATERIAL_MODE          50.84
dtype: float64
```

Comments:

Notice that there are columns having almost 48%, 13%, 19% etc. missing values. When dealing with columns, you have two simple choices - either delete or retain the column. If you retain the column, you'll have to treat (i.e. delete or impute) the rows having missing values.

If you delete the missing rows, you lose data. If you impute, you introduce bias.

Apart from the number of missing values, the decision to delete or retain a variable depends on various other factors, such as:

The analysis task at hand, The usefulness of the variable (based on your understanding of the problem), The total size of available data (if you have enough, you can afford to throw away some of it) etc. Thus, for this exercise, let's remove the columns having more than missing values and which are not necessary for our analysis.

We Drop unnecessary columns from the dataset

Handling Outliers

Major approaches to the treat outliers:

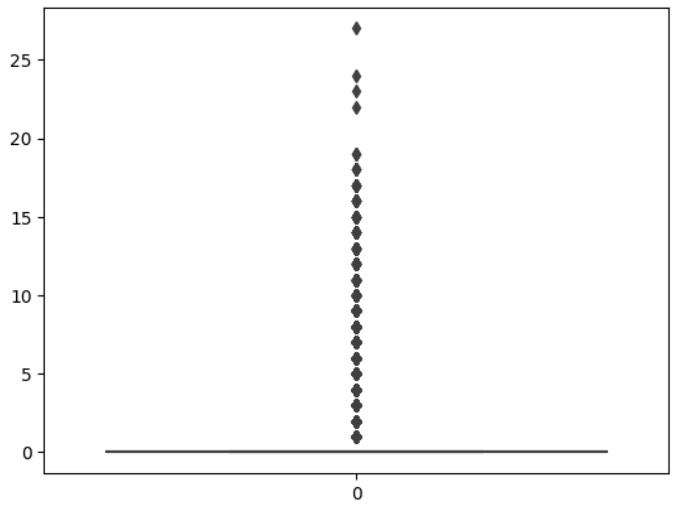
- **Imputation**
- **Deletion of outliers**
- **Binning of values**
- **Cap the outlier**

Imputation technique for with ~ 13% of missing values

The target columns for this analysis will be

- AMT_REQ_CREDIT_BUREAU_MON
- AMT_REQ_CREDIT_BUREAU_WEEK
- AMT_REQ_CREDIT_BUREAU_DAY
- AMT_REQ_CREDIT_BUREAU_HOUR
- AMT_REQ_CREDIT_BUREAU_QRT
- AMT_REQ_CREDIT_BUREAU_YEAR

Identifying imputation technique for AMT_REQ_CREDIT_BUREAU_MON




```
df['AMT_REQ_CREDIT_BUREAU_MON'].value_counts (no

0.0     83.548753
1.0     12.461653
2.0      2.024873
3.0      0.748519
4.0      0.404523
5.0      0.226323
6.0      0.128951
7.0      0.112033
9.0      0.077446
8.0      0.069551
10.0     0.049626
11.0     0.044738
12.0     0.028948
13.0     0.027068
14.0     0.015038
15.0     0.013158
16.0     0.008647
17.0     0.005263
18.0     0.002256
19.0     0.001128
24.0     0.000376
23.0     0.000376
27.0     0.000376
22.0     0.000376
Name: AMT_REQ_CREDIT_BUREAU_MON, dtype: float64
```

```
[ ] #Checking statistics of the column
    df['AMT_REQ_CREDIT_BUREAU_MON'].describe()

count    265992.000000
mean          0.267395
std           0.916002
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max          27.000000
Name: AMT_REQ_CREDIT_BUREAU_MON, dtype: float64
```

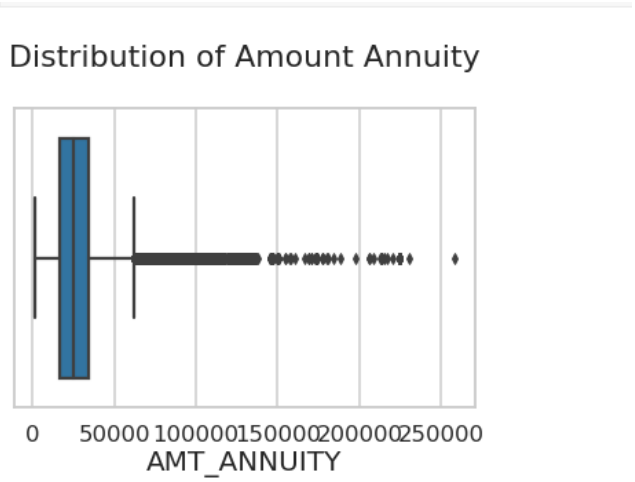
The statistical summary of the column shows there are or

Comments:

The statistical summary of the column shows there are only few records with value greater than 0. It explains the reason behind distorted boxplot.

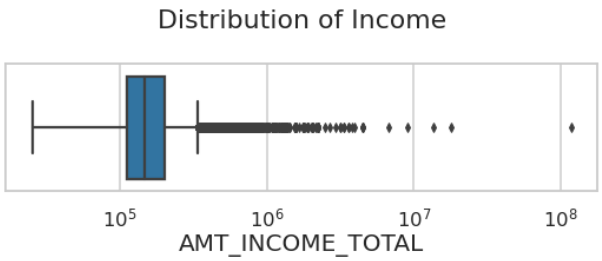
For column AMT_REQ_CREDIT_BUREAU_MON, we have two approaches either exclude missing values or impute the column with value 0 which is present in more than 83% of the rows. Hence, the recommended imputation technique is replacing null by the mode which is 0.

For the rest of the columns as well we followed the same technique to impute or restore values as they showed the same pattern as above.



As we take a look at the AMT_ANNUITY column we can see that there are outliers at 258025. But there is not much difference between the mean and median, We can impute the outliers with Median here.

AMT_INCOME variable



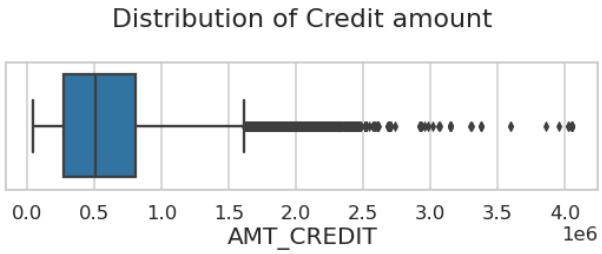

```
In [322]: df.AMT_INCOME_TOTAL.quantile([0.5, 0.7, 0.9,0.95,0.99])

Out[322]: 0.50    147150.0
          0.70    180000.0
          0.90    270000.0
          0.95    337500.0
          0.99    472500.0
          Name: AMT_INCOME_TOTAL, dtype: float64
```

Comment:

In the 'AMT_INCOME_TOTAL' column, We can see that there are outlier values at 1.17x10^8. Sometimes, it is beneficial to look into the quantiles instead of the box plot, mean, or median. Quantile may give you a fair idea about the outliers. If there is a huge difference between the maximum value and the 95th or 99th quantiles, then there are outliers in the data set.

Total income will vary from person to person. We can cap the outliers here

AMT_CREDIT variable

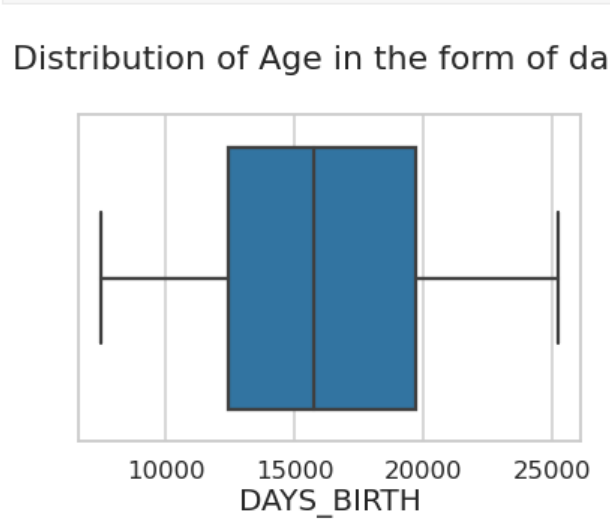



```
In [325]: df.AMT_CREDIT.quantile([0.5, 0.7, 0.9,0.95,0.99])

Out[325]:
0.50     513531.0
0.70     755190.0
0.90    1133748.0
0.95    1350000.0
0.99    1854000.0
Name: AMT_CREDIT, dtype: float64
```

Comment:

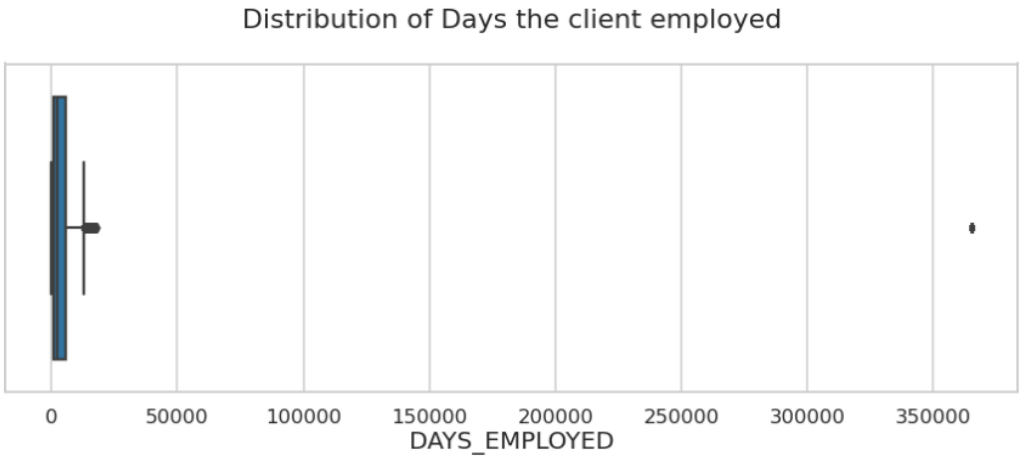
In this AMT_CREDIT column we can see the outliers after 99th quantile at 4.05x10^6 Amount credited also varies from person to person.

DAYS_BIRTH variable



DAYS_BIRTH column we can see from the box plot that there are no outliers. There is not much difference between mean and median. This means that all the applications received from the customers are of almost the same age.

DAYS_EMPLOYED variable



Comment:

DAYS_EMPLOYED column has outliers at 365243. The number of days the person was employed varies from person to person

Analysis

Checking the imbalance Percentage

```
0    91.927118
1     8.072882
Name: TARGET, dtype: float64
```

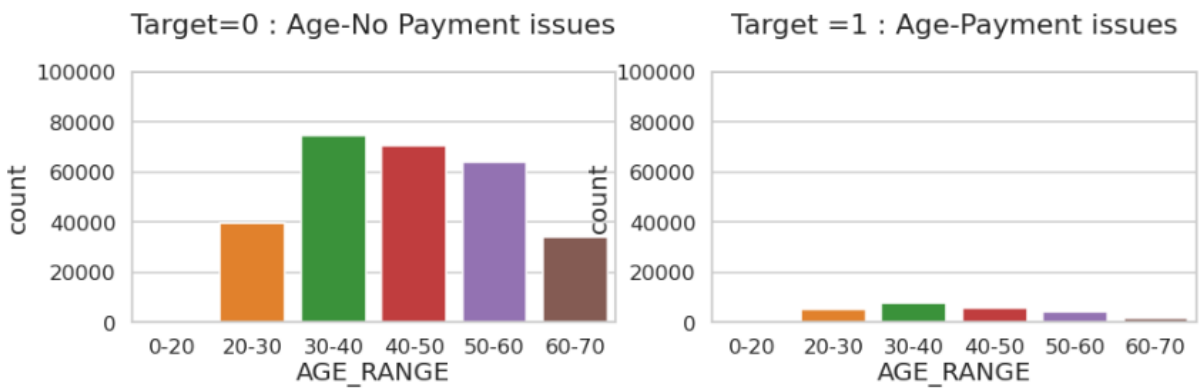
Comment:

So the TARGET column has 8.07% of 1's which means 8% of clients have payment difficulties and 91.92% are having no difficulties

Univariate Analysis for target =0 and target=1

Numeric variable

1. Age

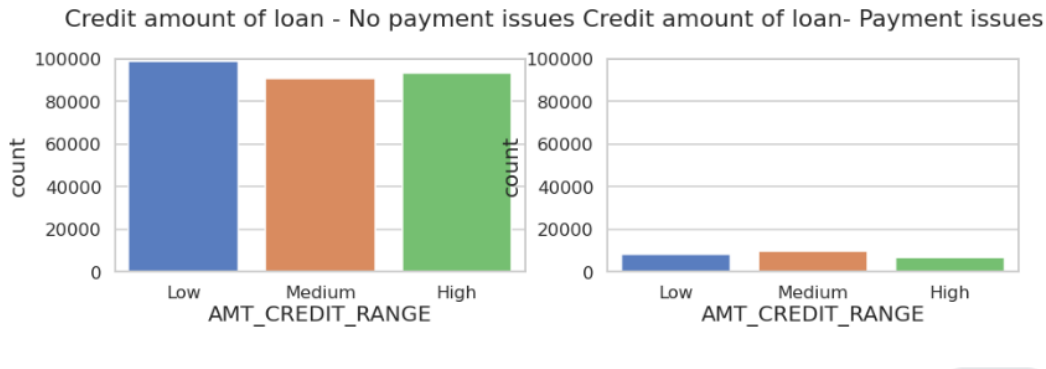


Comment:

We can observe that customers belonging to the age group 30-40 are able to make payment on time and can be considered while lending a loan!

Customers from 40 to 60 age are also can be considered.

2. Amount credit range

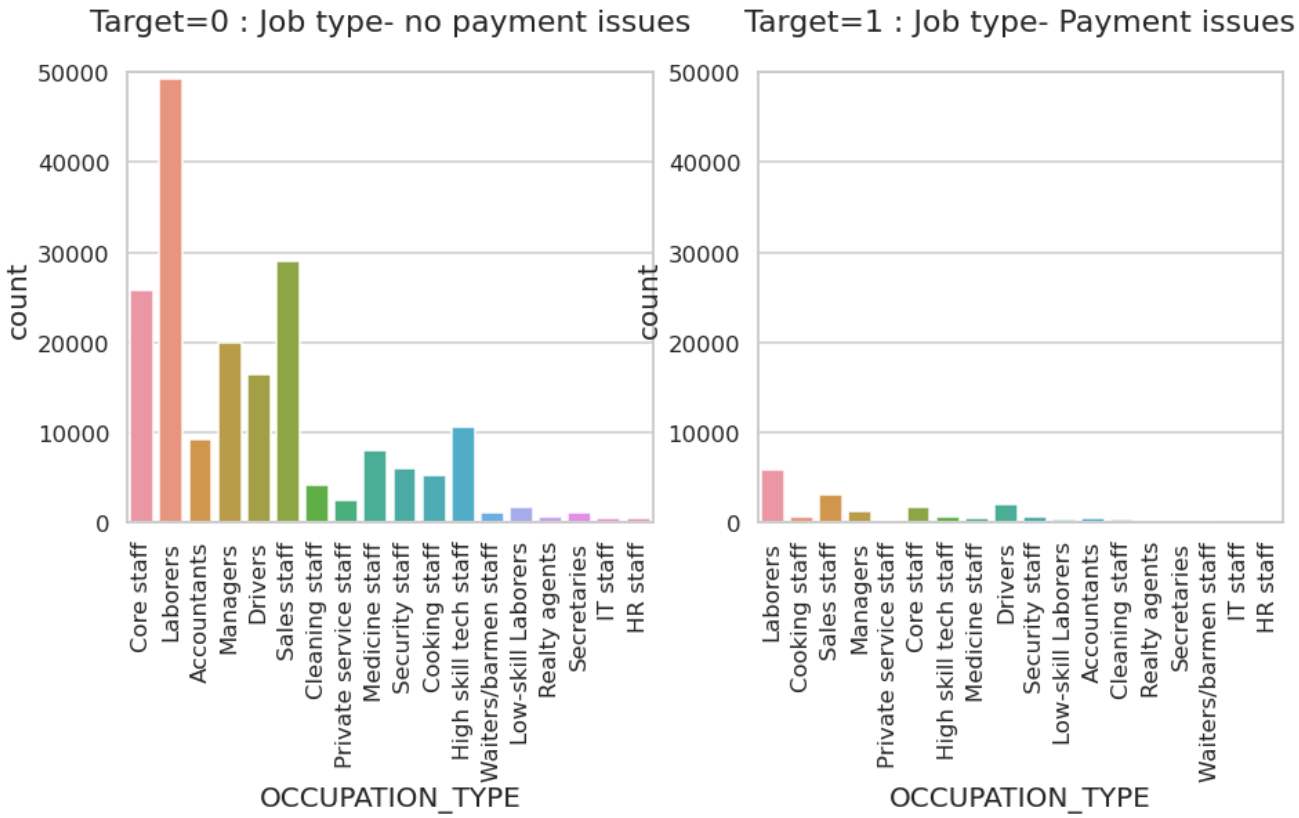


Comment:

Customers with less credit and most likely to make payments. Customers having medium and high credit can also be considered while lending the loan.

Categorical Variable

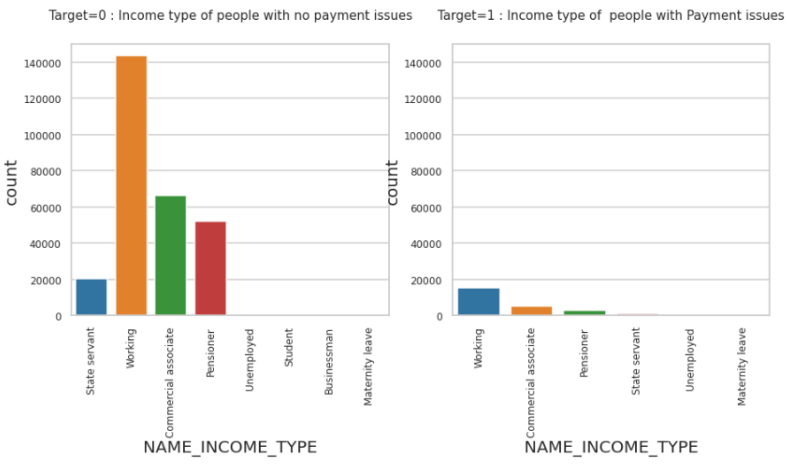
1. Occupation_type



Comment:

The plot clearly shows that labourers are most likely to make payments on time whereas HR staff are less likely to make payments on time

2. ***Name _Income _Type***



Comment:

The plot clearly shows that labourers are most likely to make payment on time whereas HR staff are less likely to make payment on time.

Analyse Categorical variables with respect to the Target variable

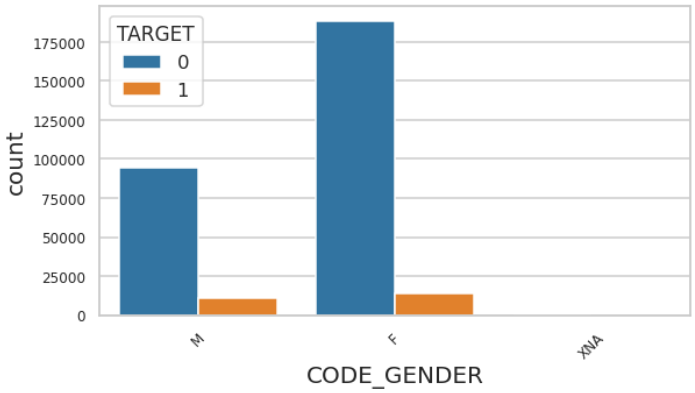
We tried to plot multiple categorical columns with respect to Target Column: Subplot

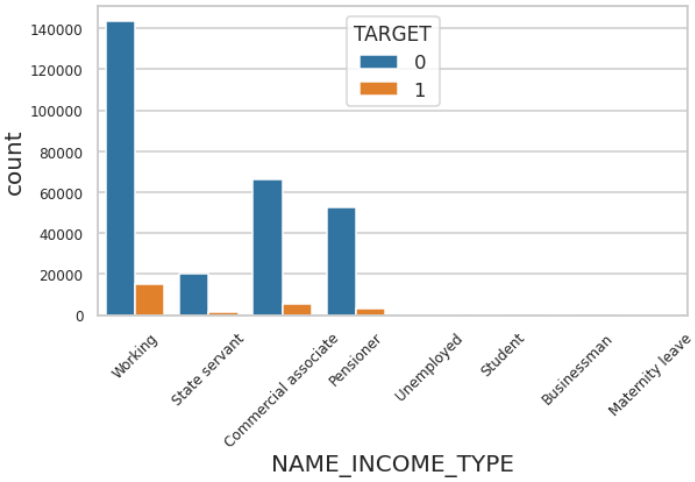
```
#Plot mutiple categorical columns with respect to Target column: Subplot
features = ['CODE_GENDER', 'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE','NAME_FAMILY_STATUS','NAME_HOUS
ING_TYPE','OCCUPATION_TYPE']
list(enumerate(features))
```

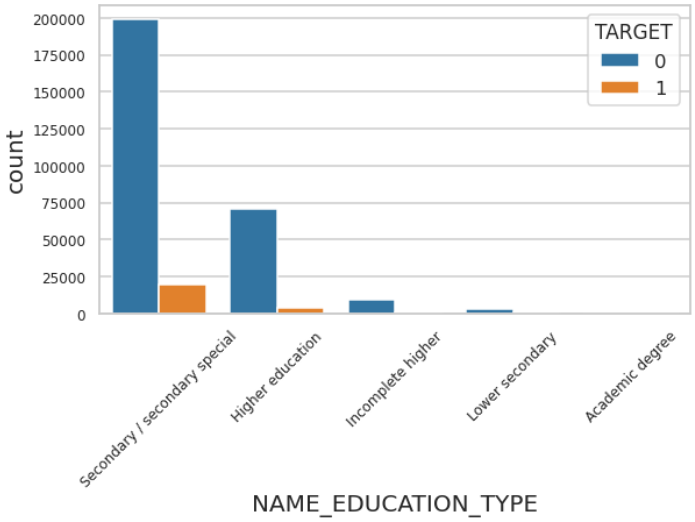
```
[(0, 'CODE_GENDER'),
 (1, 'NAME_INCOME_TYPE'),
 (2, 'NAME_EDUCATION_TYPE'),
 (3, 'NAME_FAMILY_STATUS'),
 (4, 'NAME_HOUSING_TYPE'),
 (5, 'OCCUPATION_TYPE')]
```

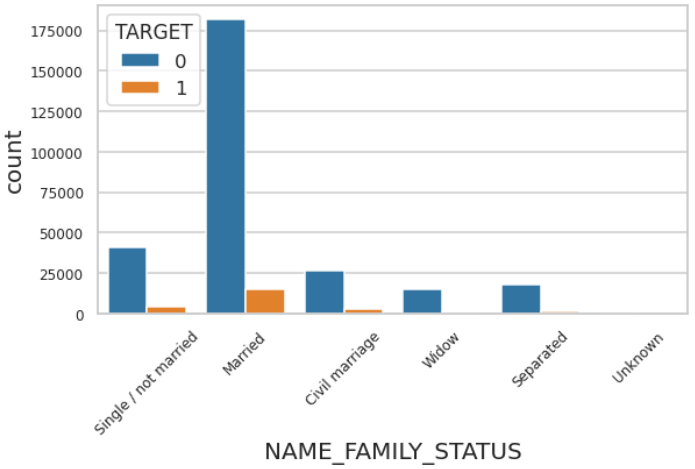
```
features = ['CODE_GENDER', 'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE','NAME_FAMILY_STATUS','NAME_HOUS
ING_TYPE','OCCUPATION_TYPE']
plt.figure(figsize = (20, 40))

plt.subplots_adjust(hspace=0.8)
for i in enumerate(features):
    plt.subplot(5, 2, i[0]+1)
    sns.countplot(x = i[1], hue = 'TARGET', data = df)
    plt.xticks(rotation = 45)
```

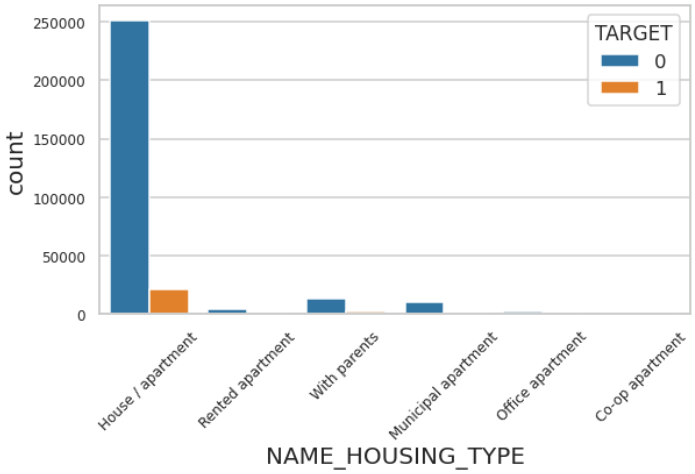
Comment:

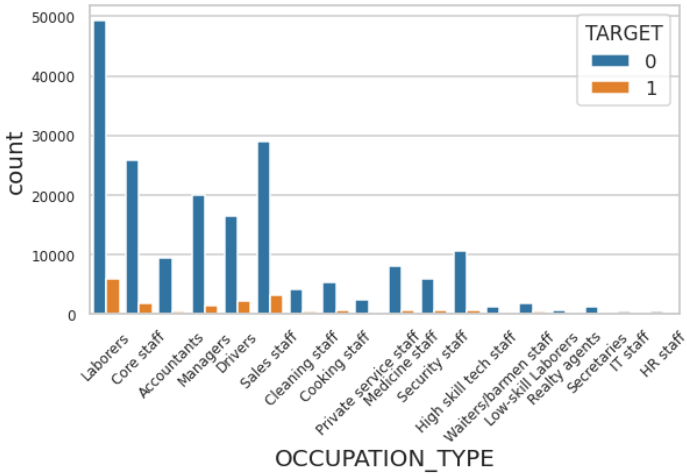

TARGET
0
1
count
175000
150000
125000
100000
75000
50000
25000
0
M
F
XNA
CODE_GENDER


TARGET
0
1
count
140000
120000
100000
80000
60000
40000
20000
0
Working
State servant
Commercial associate
Pensioner
Unemployed
Student
Businessman
Maternity leave
NAME_INCIDENT_TYPE


TARGET
0
1
count
200000
175000
150000
125000
100000
75000
50000
25000
0
Secondary / secondary special
Higher education
Incomplete higher
Lower secondary
Academic degree
NAME_EDUCATION_TYPE


TARGET
0
1
count
175000
150000
125000
100000
75000
50000
25000
0
Single / not married
Married
Civil marriage
Widow
Separated
Unknown
NAME_FAMILY_STATUS


TARGET
0
1
count
250000
200000
150000
100000
50000
0
House / apartment
Rented apartment
With parents
Municipal apartment
Office apartment
Co-op apartment
NAME_HOUSING_TYPE


TARGET
0
1
count
50000
40000
30000
20000
10000
0
Laborers
Core staff
Accountants
Managers
Drivers
Sales staff
Cleaning staff
Cooking staff
Private service staff
Medicine staff
Security staff
High skill tech staff
Waiters/barmen staff
Low-skill Laborers
Realty agents
Secretaries
IT staff
HR staff
OCCUPATION_TYPE

Comment:

From the above plot, we can see that,
Female customers pay loan amounts on time and banks can target more female customers for lending loans.
Working customers can be targeted to lend loans as they have a higher percentage of making payments on time.
Customers with secondary education are most likely to make payments when compared to customers with academic degrees.
Married customers have paid loan amounts on time when compared to widows.
Customers owning houses/apartments are most likely to make payments on time compared to those living in CO-OP apartments.
Laborers have a high repayment percentage. Hence banks can think of lending small amounts of loans to them.

Correlation Matrix

To get rid of the repeated correlation values between two variables we perform the following steps.

]:

	VAR1	VAR2	Correlation
398	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	1.00
148	AMT_GOODS_PRICE	AMT_CREDIT	0.98
423	DEF_60_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	0.87
124	AMT_ANNUITY	AMT_CREDIT	0.75
149	AMT_GOODS_PRICE	AMT_ANNUITY	0.75
224	DAYS_EMPLOYED	DAYS_BIRTH	0.58
399	OBS_60_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	0.34
374	DEF_30_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	0.33
248	DAYS_REGISTRATION	DAYS_BIRTH	0.29
422	DEF_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	0.26

We saw that for the Target_0 data frame, Social circle for 30 days and 60 days are most correlated, and Goods price and Loan amount credit are highly correlated. Then we have Goods price and amount annuity in 4th place

Comment:

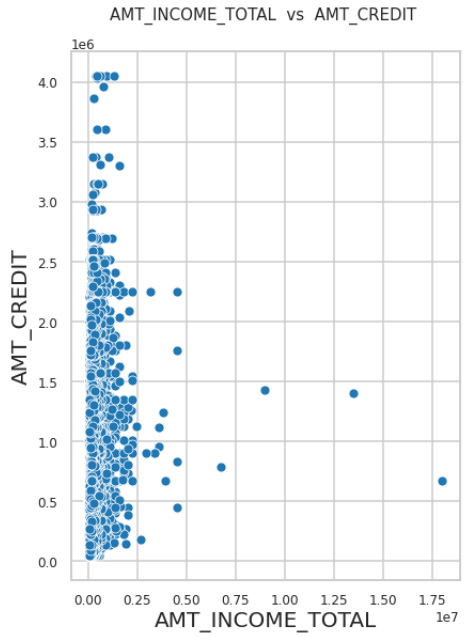
From the observation above we can say that for the target_1 data frame Goods price and loan credit amount are most correlated next to social circle observations for different days. So, the variables correlated in the target_0 data frame and target_1 data frame are same with slightly varying correlation values

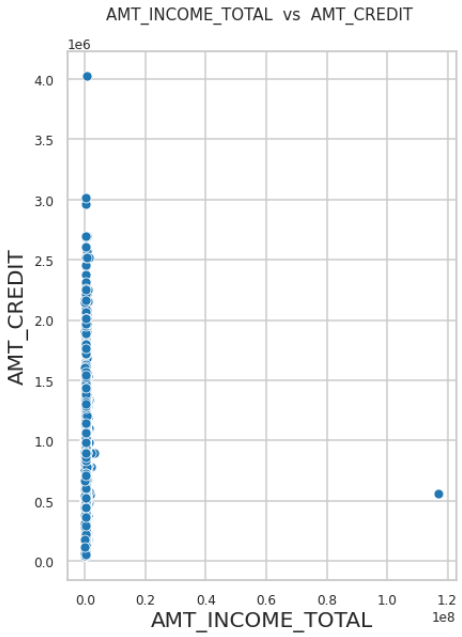
Bivariate Analysis for Target 0 and Target 1

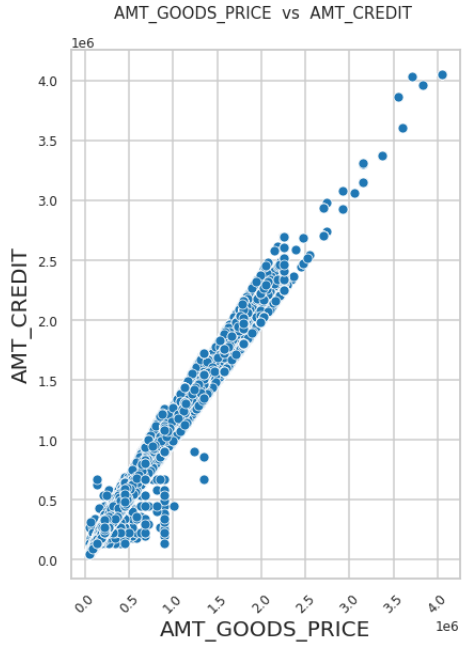
Numeric-Numeric Analysis

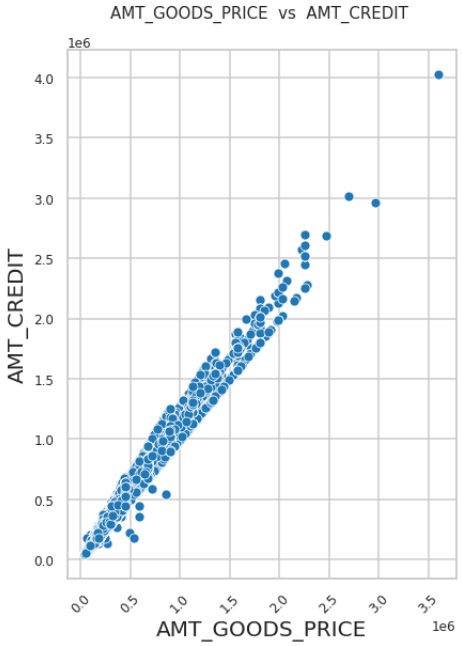
There are three ways to analyze the numeric-numeric data types simultaneously. Scatter plot: describes the pattern that how one variable is varying with another variable. Correlation matrix: to describe the linearity of two numeric variables. Pair plot: a group of scatter plots of all numeric variables in the data frame

Income vs Credit, Goods price vs Credit









Comment:

Those who have paid the loan amount on/within time are more likely to get higher credits than those who didn't pay/did late payments.

People who have higher goods prices and have made payments on time have higher credits than those with higher goods price but didn't pay loan.

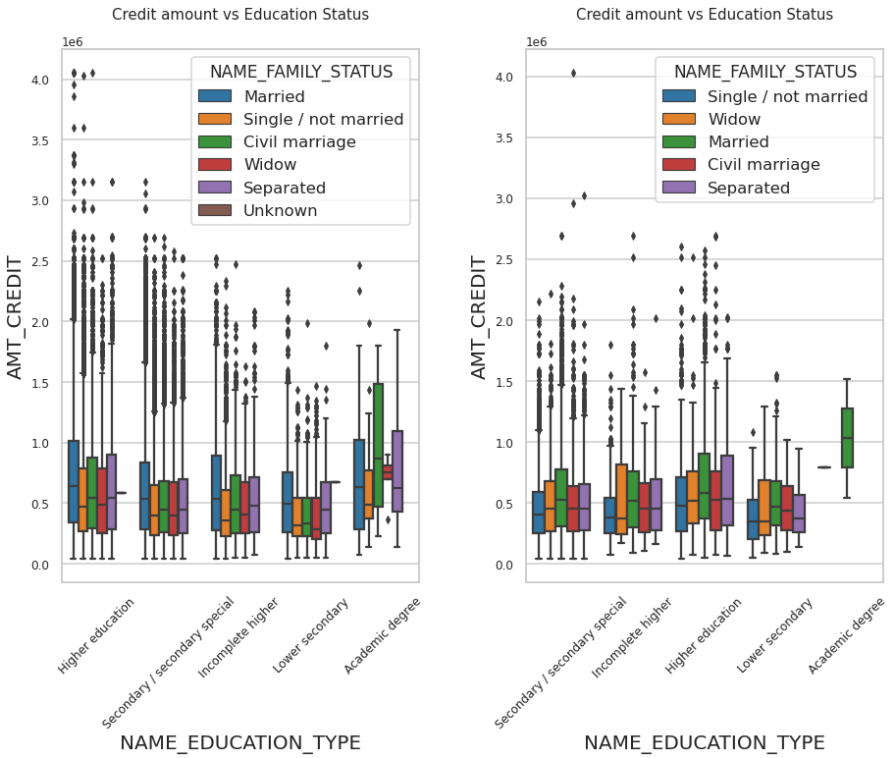
Numerical categorical analysis

Income range- Gender

Comment:

We can see that Females with low income don't have any payment issues.

Credit amount vs Education Status



Comment:

From the above plot, we can see that

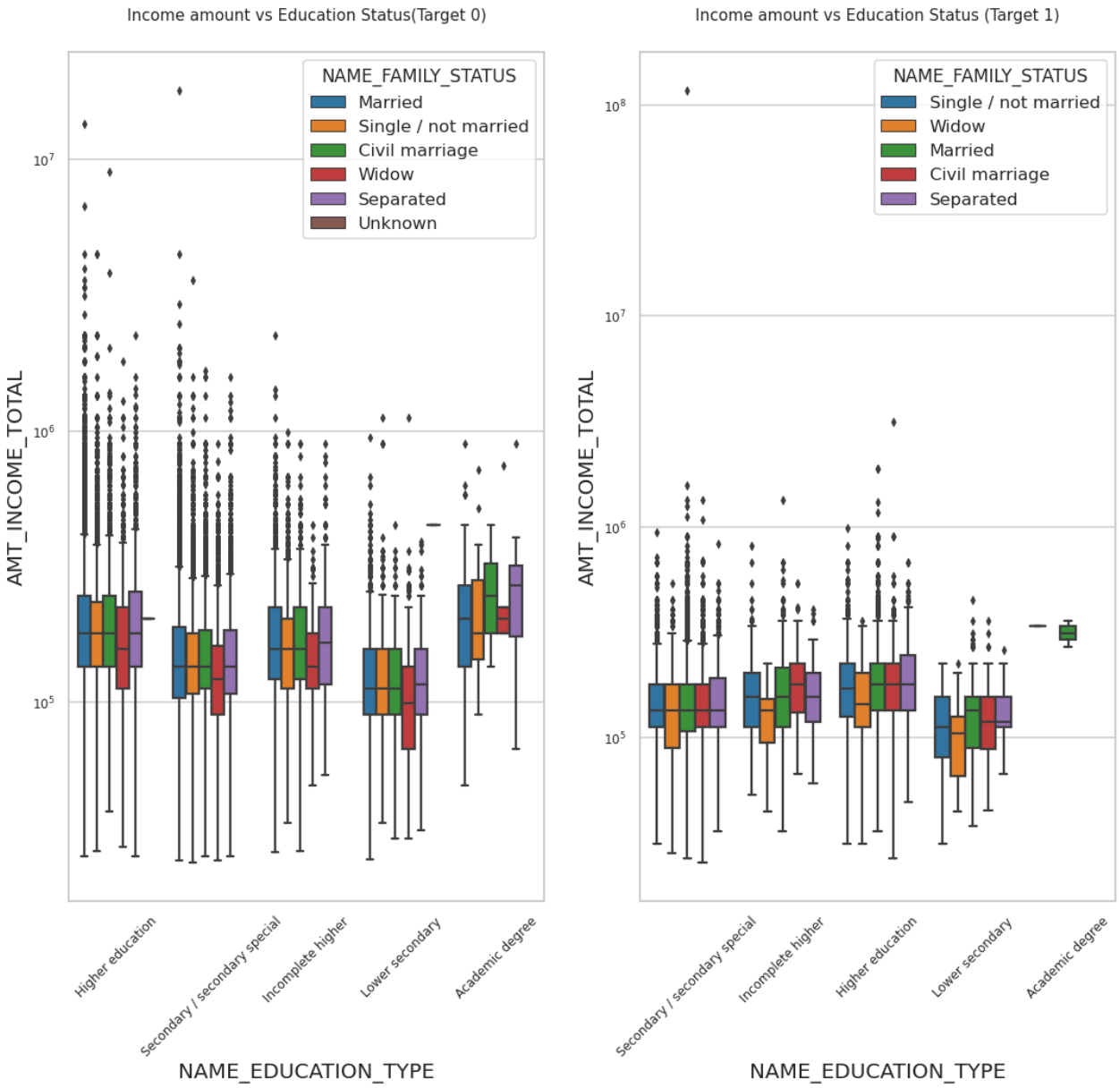
1. Some of the highly educated, married person is having credits higher than those who have done lower secondary education.

2. Those with higher education have higher credits and are more likely to make payments on time.

3. More outliers are seen in higher education.

4. people with secondary and secondary special education are less likely to make payments on time.

Income vs Education Status



Comment:

From the above plots,

1. we can see that Higher education has many outliers.

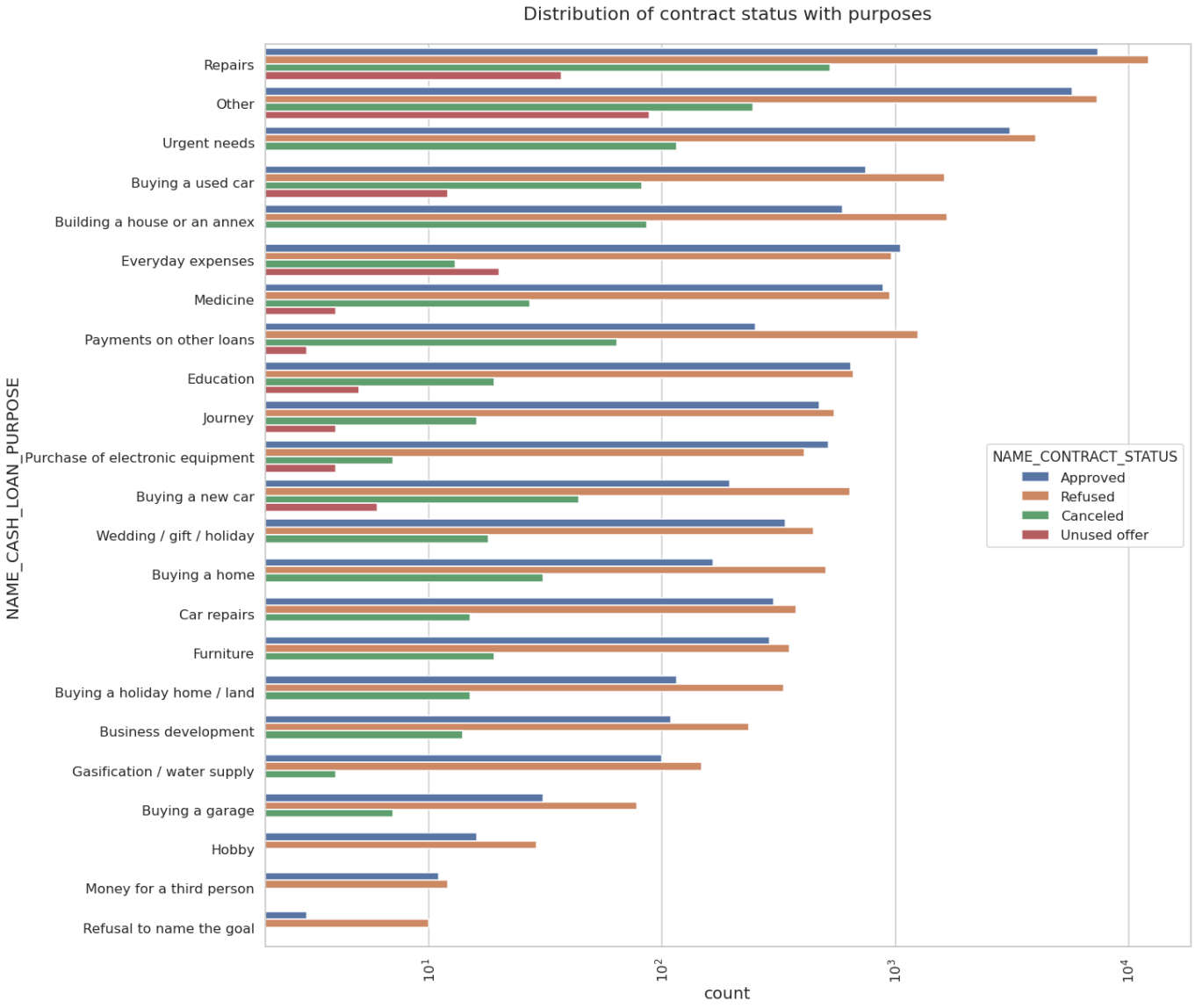
2. People with higher education have higher incomes and don't have difficulties in making loan payments.

3. People with higher education who have lesser income are unable to pay the loan.

Hence, we can conclude that people with Higher incomes are most likely to make payments.

Univariate Analysis:

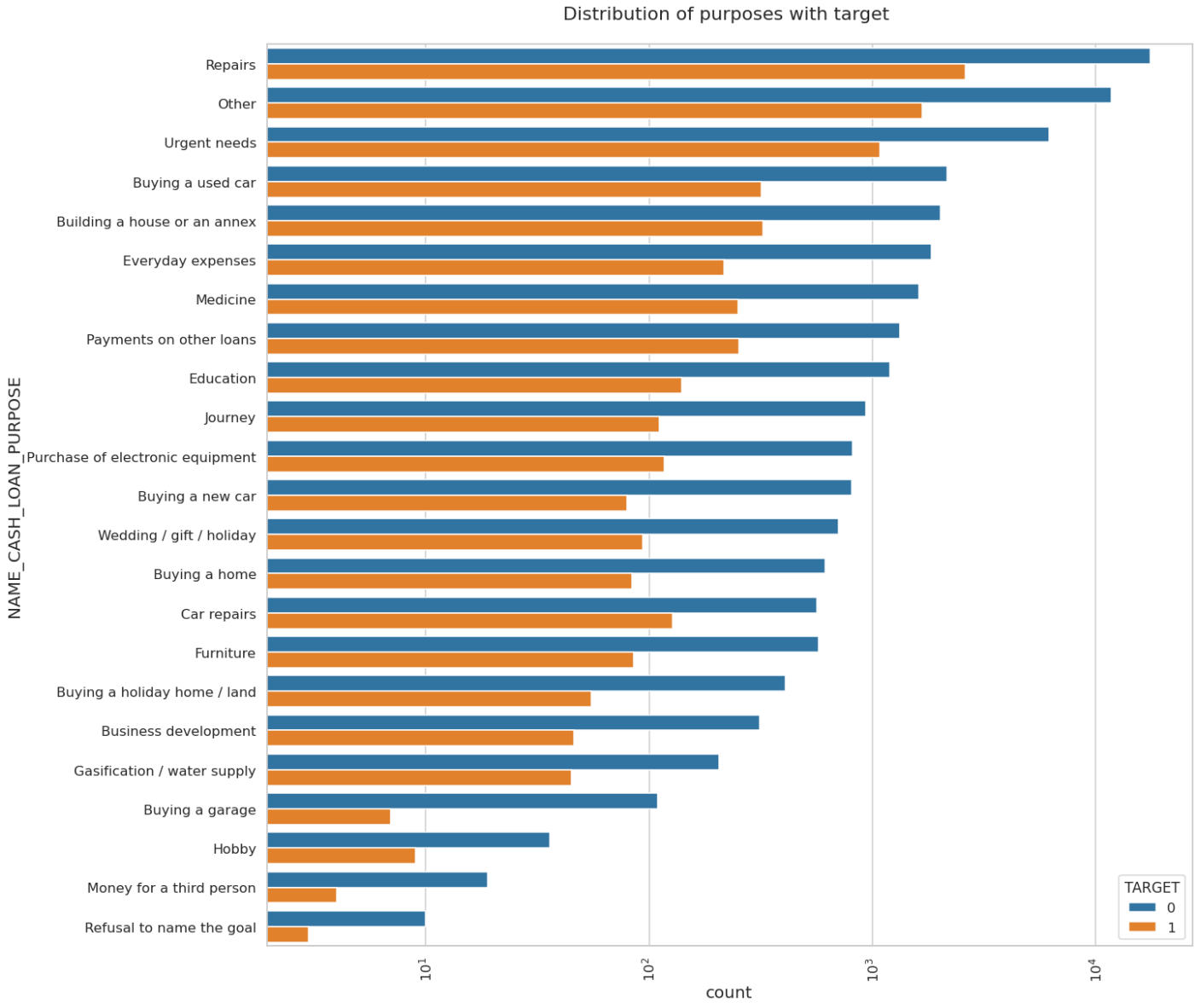
Distribution of contract status in a logarithmic scale



Comments:

Points to be concluded from the above plot:
Most rejections of loans came from the purpose of 'Repairs'. For education purposes, we have an equal number of approved and rejection Paying's other loans and buying a new car is having significant higher rejections than approves.

Distribution of contract status



Comments:

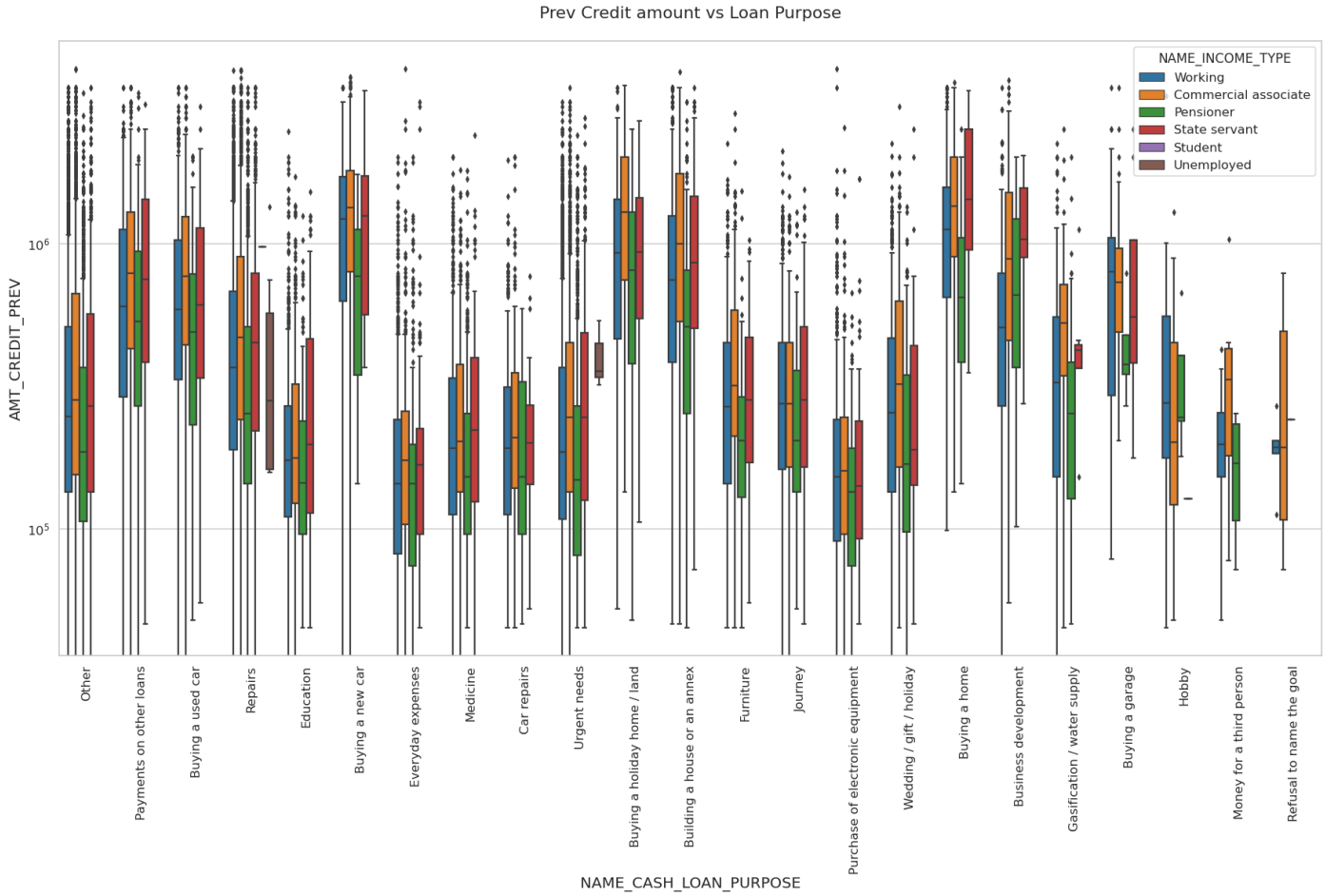
Few points we can conclude from the above plot:

Loan purposes with 'Repairs' are facing more difficulties in payment on time. There are few places where loan payment is significantly higher than facing difficulties. They are 'Buying a garage', 'Business development', 'Buying land', 'Buying a new car', and 'Education' Hence we can focus on these purposes for which the client is having minimal payment difficulties

Bivariate Analysis

Prev Credit amount vs Loan Purpose

Box plotting for Credit amount in logarithmic scale

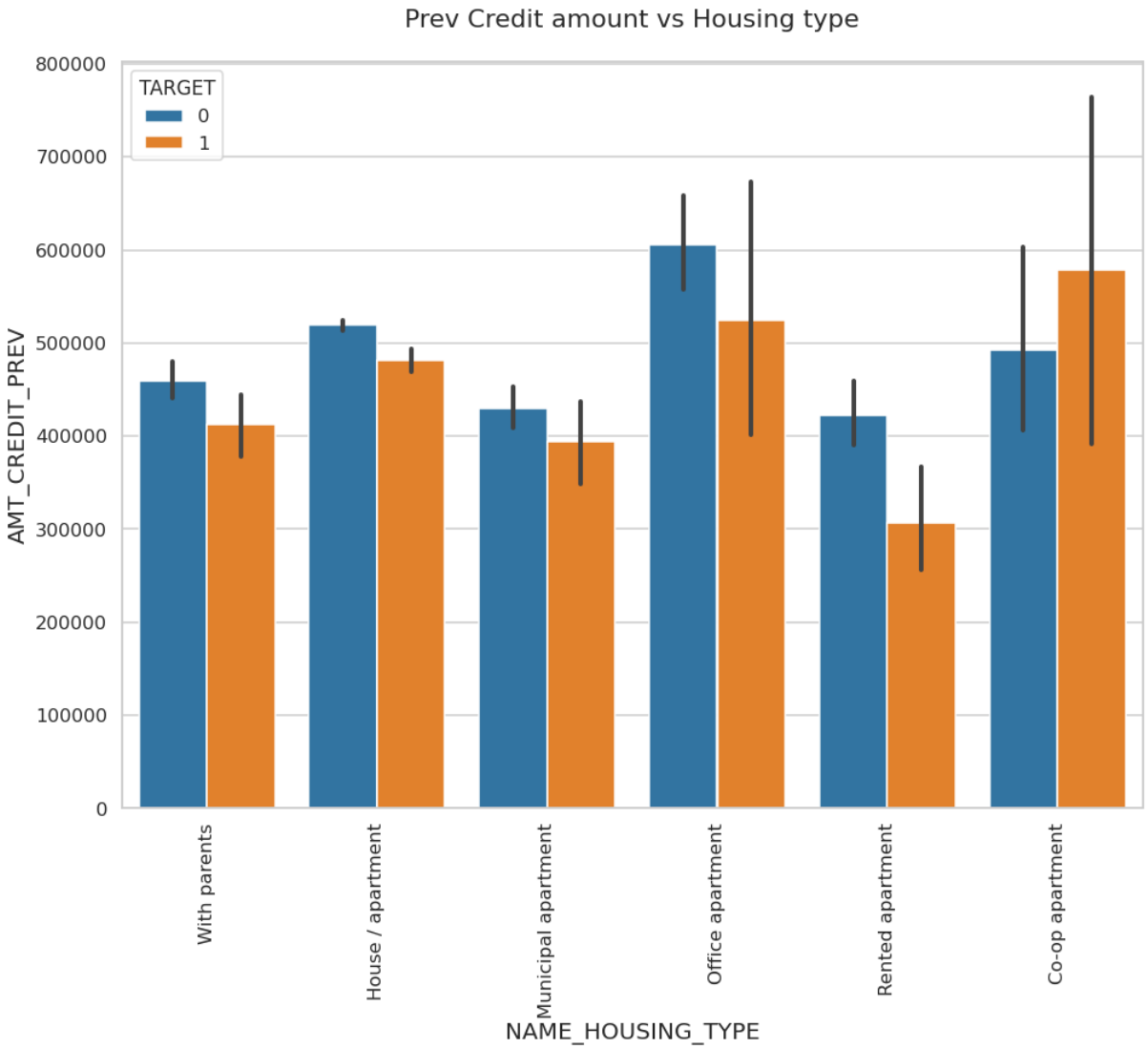


Comments:

From the above we can conclude some points- The credit amount for Loan purposes like 'Buying a home',' Buying land',' Buying a new car', and building a house' is higher. The

income type of state servants has a significant amount of credit applied Money for the third person or a Hobby is having fewer credits applied.

Box plotting for Credit amount prev vs Housing type in logarithmic scale



Comments:

Here for Housing type, office apartment is having higher credit of target 0 and co-op apartment is having higher credit of target 1. So, we can conclude that bank should avoid giving loans to the housing type of co-op apartment as they are having difficulties in payment. Bank can focus mostly on housing type with parents or House\apartment or municipal apartment for successful payments.

CONCLUSION

1. Banks should focus more on contract types 'Student',' pensioner', and 'Businessman' with housing 'types other than 'Co-op apartment' for successful payments.

2. Banks should focus less on income type 'Working' as they are having the most number of unsuccessful payments.

3. Also with loan purposes 'Repair' is having a higher number of unsuccessful payments on time.

4. Get as many clients from the housing type 'With parents' as they are having the least number of unsuccessful payments.

Results:

After running all the formulas in Microsoft Excel and plotting the charts we analyzed all the data sets of this Project.

In the making of this report, we used our Microsoft Excel knowledge and Python as a real-world examples.

Drive Link

https://drive.google.com/drive/folders/1ppGHWEgDT_QgoCvrFqdCa6_BPZLL7UNu?usp=sharing