Data Analytics Project: Bank Loan Case Study

Overview:
1. Project Description
2. Approach
3. Tech-Stack Used
4. Data Summary/ Data Understanding
5. Data Cleaning
6. Outliers Detection 7. Data Imbalance
8. Univariate, Segmented Univariate and Bivariate analysis
9. Top 10 correlations
10. Final Conclusion
11. Result
12. Links to work files

1. Project Description:

The main aim of this project is to identify patterns that indicate if a customer will have difficulty paying their instalments. This information can be used to make decisions such as denying the loan, reducing the amount of loan, or lending at a higher interest rate to risky applicants.

This project results helps company understand the key factors behind loan default so it can make better decisions about loan approval.

2. Approach:

Use EDA to understand how customer attributes and loan attributes influence the likelihood of default.

Steps taken for EDA are

1) Research on risk analytics in banking & financial services
2) Understand the data description provided for columns
3) Clean the data, identify, and remove/impute missing values, identify outliers for better analysis
4) Perform univariate, segmented univariate and bivariate analysis to gain insights
5) Identify top correlations for different scenarios
6) Provide conclusion on the analysis

3. Tech-Stack Used:

Microsoft Excel 2019, Jupyter Notebook 6.4.8, python scripts.

4. Data summary:

Application_data.csv:

1) Data Understanding

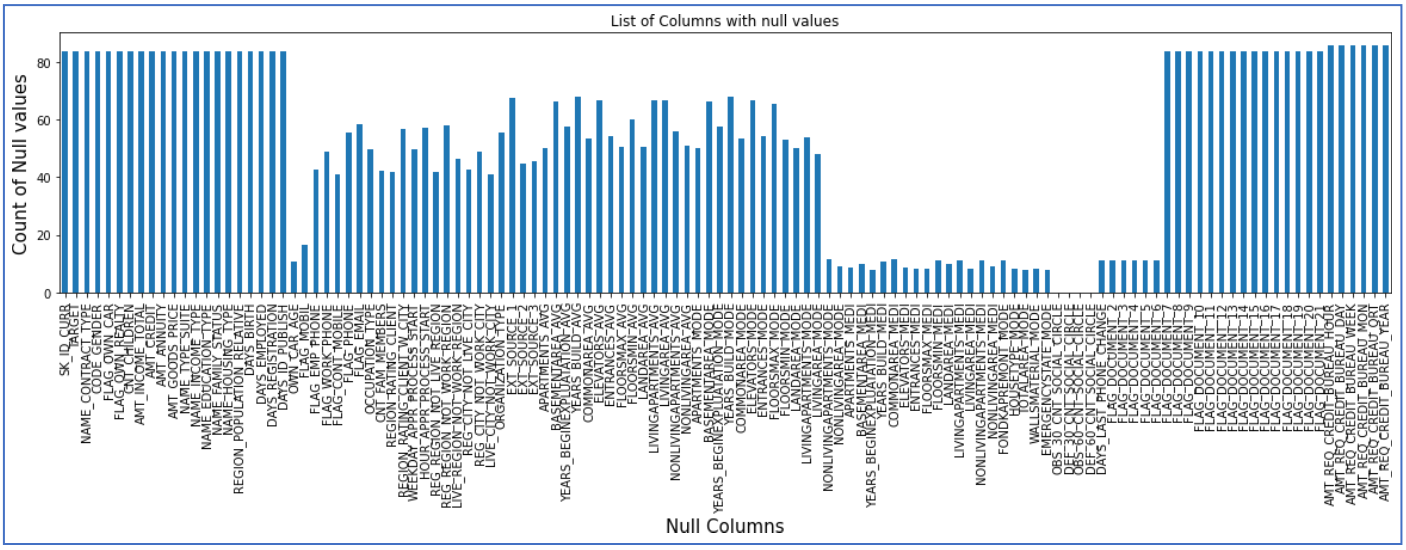
```
#metadata of df_current
print(df_current.shape)
df_current.describe()
```

```
(307511, 122)
```

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	REGION_POPULATION_RELATIVI
count	49999.000000	49999.000000	49999.000000	4.999900e+04	4.999900e+04	49998.000000	4.996100e+04	49999.00000
mean	129013.210584	0.080522	0.419848	1.707676e+05	5.997006e+05	27107.377355	5.390600e+05	0.02079
std	16690.512048	0.272102	0.724039	5.318191e+05	4.024154e+05	14562.944435	3.698533e+05	0.01376
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04	2052.000000	4.500000e+04	0.00053
25%	114570.500000	0.000000	0.000000	1.125000e+05	2.700000e+05	16456.500000	2.385000e+05	0.01000
50%	129076.000000	0.000000	0.000000	1.458000e+05	5.147775e+05	24939.000000	4.500000e+05	0.01885
75%	143438.500000	0.000000	1.000000	2.025000e+05	8.086500e+05	34596.000000	6.795000e+05	0.02866
max	157875.000000	1.000000	11.000000	1.170000e+08	4.050000e+06	258025.500000	4.050000e+06	0.07250

Here, the total number of rows is 307511, but the count of SK_ID_CURR (i.e client ID) is 49999, which shows that there is lot of missing data in the dataset.

5. Data cleaning:



From the above graph we see that all the columns in the dataset has null values.

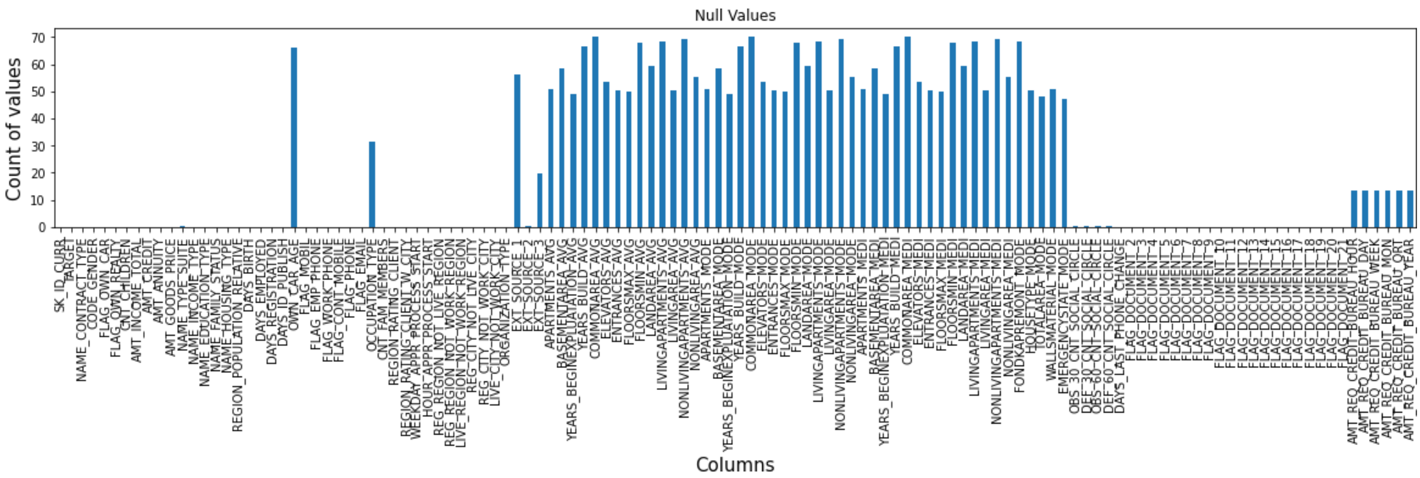
So, for the rows where client ID, TARGET column and all other columns related to client information are null, such rows shall be removed

After deleting above mentioned rows, we have 49999 rows and 122 columns

```
#drop rows where ID Column is null
df_current = df_current.dropna(axis=0, subset=['SK_ID_CURR'])
print(df_current.shape)
```

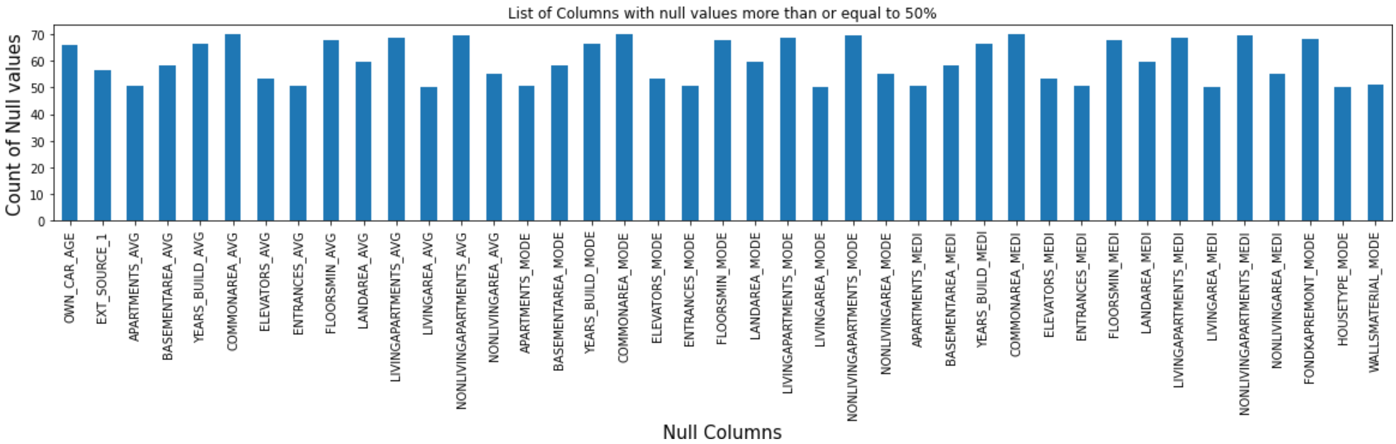
```
(49999, 122)
```

a) Identify the missing data from current application file and use appropriate method to deal with it.



From above graph, we see there are columns having more than 50% of data as null values, so we remove such columns

These are the columns that are removed:



b) Correcting datatypes:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 49999 entries, 0 to 49998
Data columns (total 81 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   SK_ID_CURR                   49999 non-null  float64
 1   TARGET                       49999 non-null  float64
 2   NAME_CONTRACT_TYPE           49999 non-null  object
 3   CODE_GENDER                  49999 non-null  object
 4   FLAG_OWN_CAR                 49999 non-null  object
 5   FLAG_OWN_REALTY              49999 non-null  object
 6   CNT_CHILDREN                 49999 non-null  float64
 7   AMT_INCOME_TOTAL             49999 non-null  float64
 8   AMT_CREDIT                   49999 non-null  float64
 9   AMT_ANNUITY                  49998 non-null  float64
 10  AMT_GOODS_PRICE              49961 non-null  float64
```

Integer columns like client ID, TARGET column has incorrect datatypes as float, many string columns have datatype as object, this is to be corrected

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY
0	100002	1	Cash loans	M	N	Y
1	100003	0	Cash loans	F	N	N
2	100004	0	Revolving loans	M	Y	Y
3	100006	0	Cash loans	F	N	Y
4	100007	0	Cash loans	M	N	Y

```
df_current.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 49999 entries, 0 to 49998
Data columns (total 81 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   SK_ID_CURR                    49999 non-null  Int64
 1   TARGET                        49999 non-null  Int64
 2   NAME_CONTRACT_TYPE            49999 non-null  string
 3   CODE_GENDER                   49999 non-null  string
 4   FLAG_OWN_CAR                  49999 non-null  string
 5   FLAG_OWN_REALTY               49999 non-null  string
```

As seen in above image, the datatypes are corrected

c) Selecting relevant columns:

There are 81 columns in application_data dataset after removing null values, not all columns are to be used for the analysis, so based on the **column_description.csv** file, we separate and store relevant columns, that are to be used for the analysis in different dataframe

```
#storing columns relevant for analysis based on the column_description.csv in seperate dataframe
relevant=['SK_ID_CURR','TARGET','NAME_CONTRACT_TYPE','CODE_GENDER','FLAG_OWN_CAR','FLAG_OWN_REALTY','CNT_CHILDREN',
          'AMT_INCOME_TOTAL','AMT_CREDIT','AMT_ANNUITY','AMT_GOODS_PRICE','NAME_INCOME_TYPE','NAME_EDUCATION_TYPE',
          'NAME_FAMILY_STATUS','NAME_HOUSING_TYPE','DAYS_BIRTH','DAYS_EMPLOYED','OCCUPATION_TYPE','CNT_FAM_MEMBERS',
          'ORGANIZATION_TYPE'
]

df_app=df_current[relevant]
df_app.head()
print(len(relevant))
print(len(df_current.columns))
```

```
20
81
```

20 relevant columns are selected as shown above.

d) Identify the missing data with less than 50% of null percentage.

```
#find the count percentage values in column having more than 0 null values
null_cols=(df_app.isnull().sum()/len(df_app.index))*100
null_cols[null_cols>0.0]
```

```
AMT_ANNUITY         0.002000
AMT_GOODS_PRICE     0.076002
OCCUPATION_TYPE    31.308626
CNT_FAM_MEMBERS     0.002000
dtype: float64
```

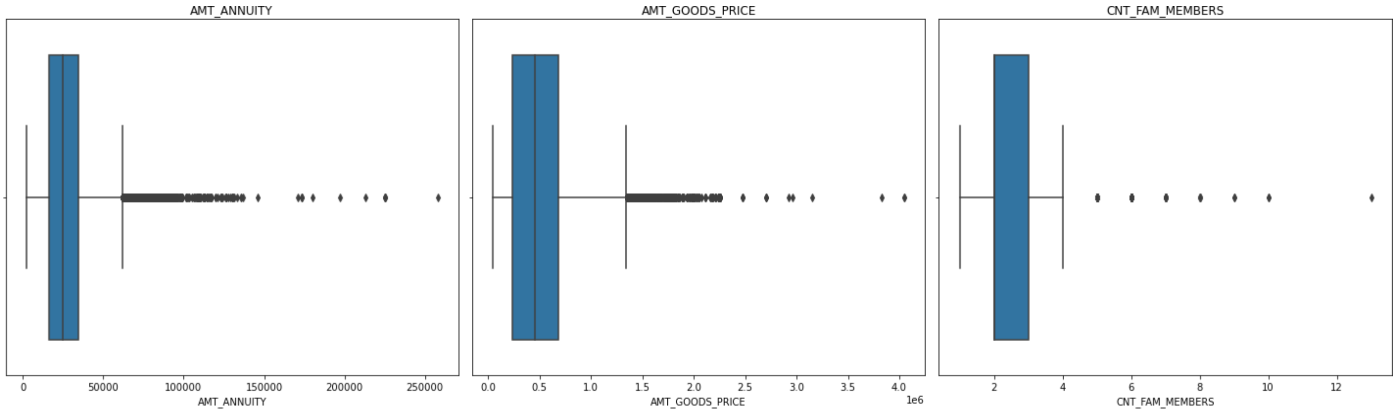
For above columns, we decide for imputation which method should be used

>Mean imputation is often used when the missing values are numerical and the distribution of the variable is approximately normal.

>Median imputation is preferred when there are huge outliers, as the median is less sensitive to outliers than the mean.

>Mode imputation is suitable for categorical variables or numerical variables with a small number of unique values.

For numerical columns, check for outliers, to decide between mean and median which method to be used:



As shown in the box plots above, all the numerical columns has huge outliers, so median value will be imputed for the missing data.

For columns OCCUPATION_TYPE, we first replace null with "UNKNOWN" and compare OCCUPATION_TYPE column with organization type and income type to find correlation if any.

```
#comparing occupation type with organization type and income type to find correlation if any:
df_app[['ORGANIZATION_TYPE','NAME_INCOME_TYPE','OCCUPATION_TYPE']].head(40)
```

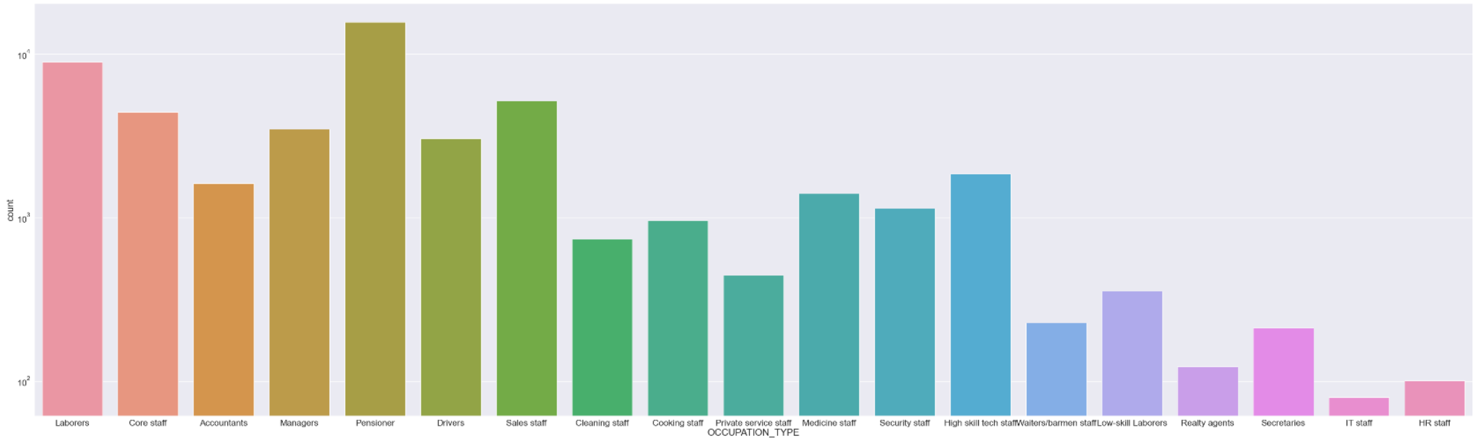
	ORGANIZATION_TYPE	NAME_INCOME_TYPE	OCCUPATION_TYPE
0	Business Entity Type 3	Working	Laborers
1	School	State servant	Core staff
2	Government	Working	Laborers
3	Business Entity Type 3	Working	Laborers
4	Religion	Working	Core staff
5	Other	State servant	Laborers
6	Business Entity Type 3	Commercial associate	Accountants
7	Other	State servant	Managers
8	XNA	Pensioner	UNKNOWN
9	Electricity	Working	Laborers
10	Medicine	Working	Core staff
11	XNA	Pensioner	UNKNOWN

The above table states for most unknown values in occupation type, income type is pensioner and organization type is xna, to analyse this further, we extract this table in csv file

From csv file, we found below ouput:

OCCUPATION_TYPE	UNKNOWN
Row Labels	**Count of ORGANIZATION_TYPE**
Pensioner	**8920**
XNA	8918
Business Entity Type 2	1
Industry: type 9	1
Working	**4119**
Commercial associate	**1972**
State servant	**635**
Unemployed	**6**
XNA	6
Maternity leave	**1**
Business Entity Type 1	1
Student	**1**
Business Entity Type 2	1
Grand Total	**15654**

Based on the output from csv file, we can substitute null values in occupation type with pensioner



In the dataset, highest number of clients are Pensioners, followed by Labourers and sales staff.

e) Converting negative days value into positive:

On inspecting data, in the days column, negative values are found, as days cannot be negative, convert them to positive values.

```
Negative values in DAYS_BIRTH column:  49999
Positive values in DAYS_BIRTH column:  0
Negative values in DAYS_EMPLOYED column:  41074
Positive values in DAYS_EMPLOYED column:  8924
```

```
#convert negative values to positive as days count can't be negative
df_app['DAYS_BIRTH']=abs(df_app['DAYS_BIRTH'])
df_app['DAYS_EMPLOYED']=abs(df_app['DAYS_EMPLOYED'])
```

f) Identify the missing data from previous application file and use appropriate method to deal with it.

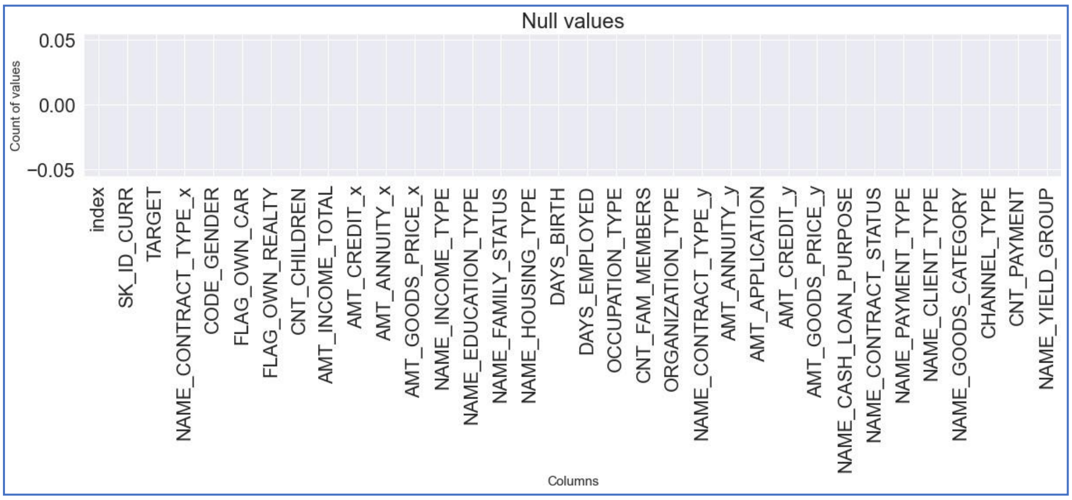
Following same steps followed above for current application dataset, previous_application dataset is also cleaned, more then 50% null columns are removed, relevant columns are seperated, datatypes are corrected and missing values are imputed with median, due to outliers:

After all the sanitisation, we have 2 datasets with 49999 rows and 20 columns in current application and 14 columns in previous application:

```
print(df_app.shape)
print(df_pre.shape)

(49999, 20)
(49999, 14)
```

Both the dataframes are merged for further analysis, and stored in dataframe dfmerg

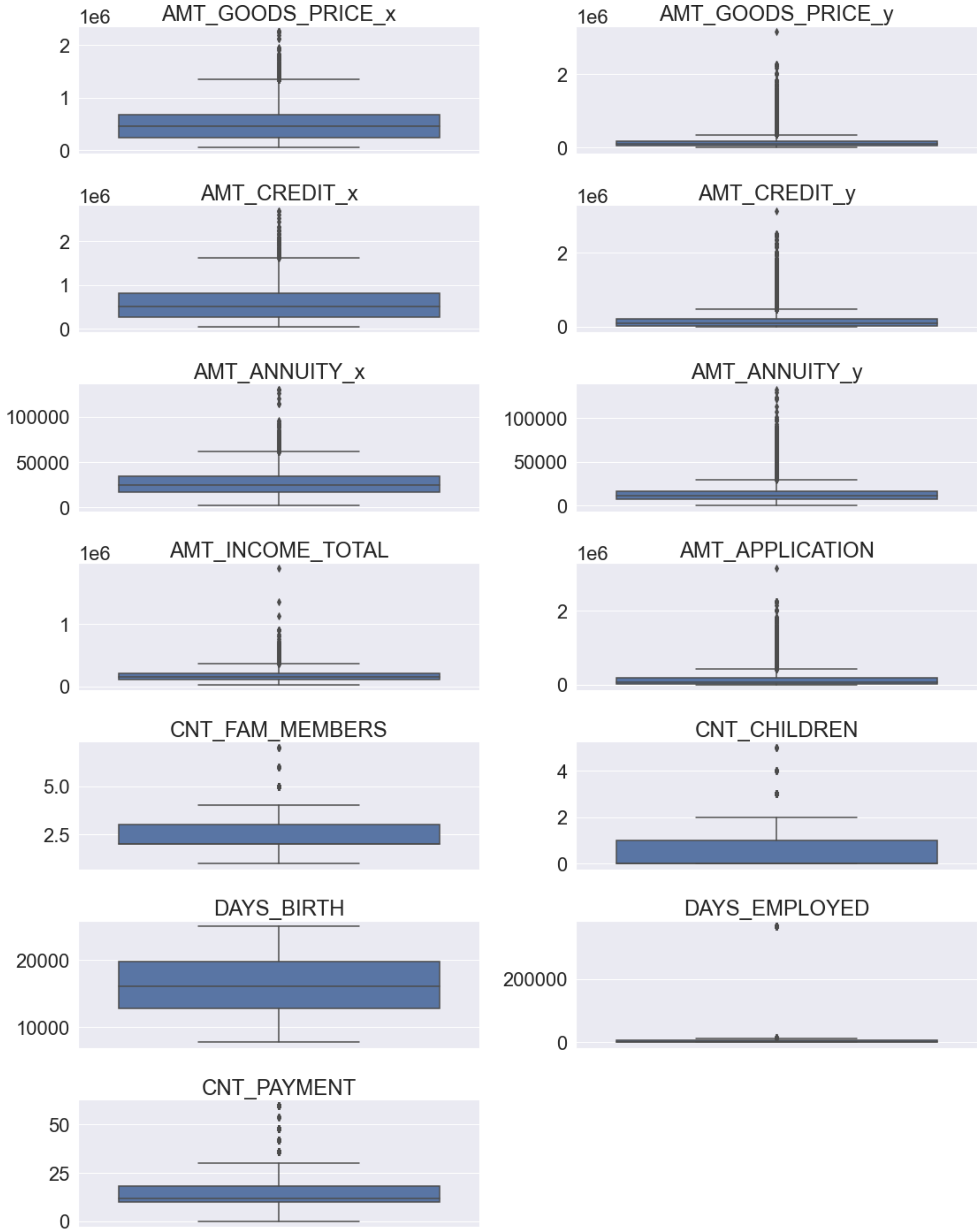
```
dfmerg = df_app.merge(df_pre, on=['SK_ID_CURR'], how='inner').reset_index()
dfmerg.shape

(6841, 34)
```



It has no null values, and will be used for further analysis.

6. Outliers Detection:

a) To identify outliers in numerical columns, we separate numerical columns and plot boxplots:



from above output we see:

1) DAYS_BIRTH has no outliers, rest all other columns have outliers

2) DAYS_EMPLOYED have huge outliers as we see the bar is very slim

3) For AMT_GOODS_PRICE, AMT_CREDIT & AMT_ANNUITY, as compared to previous application there are less outliers in current application data

b) binning columns having continuous variables for further analysis:

Using min,median,max and quantile function, we bin the columns with continuos variables.

c) Convert days columns to years:

For better analysis, convert days to years to identify outliers:

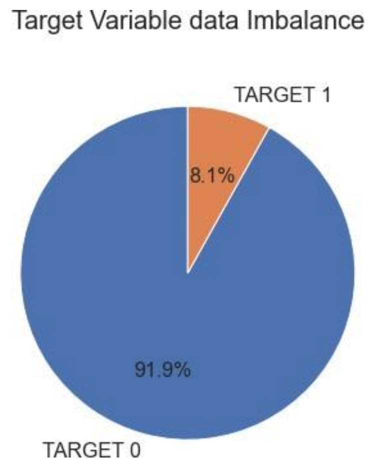
```
dfmerg['YEARS_EMPLOYED'].agg(['min', 'median', 'max'])

min          0.0
median       6.0
max       1000.0
Name: YEARS_EMPLOYED, dtype: float64
```

As seen in above output, YEARS_EMPLOYED is the number of years the client started employment before applying for loan, this column has an outlier showing 1000 years as a datapoint, which is not possible.

7. Analyze Data Imbalance

The dataset is based on te clients loan application and the analysis is to find the factors that lead clients to default, so we check TARGET variable for data imbalance:

```
Values:
 0     6287
1      554
Name: TARGET, dtype: Int64

Percentage:
 0     91.9
1      8.1
Name: TARGET, dtype: Float64
```



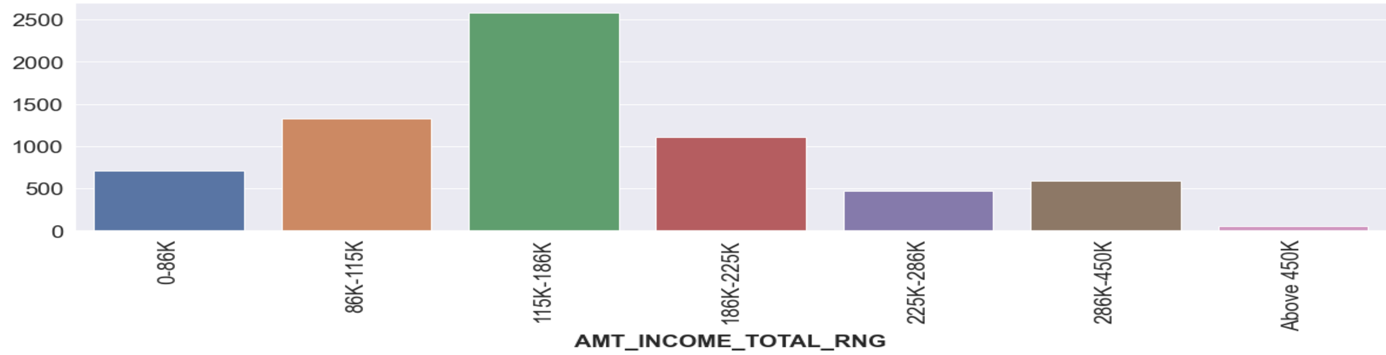
The above output states a huge imbalance in the dataset, as only 8.1% of data is about the clients with payment difficulties and 91.9% of data is about all other cases

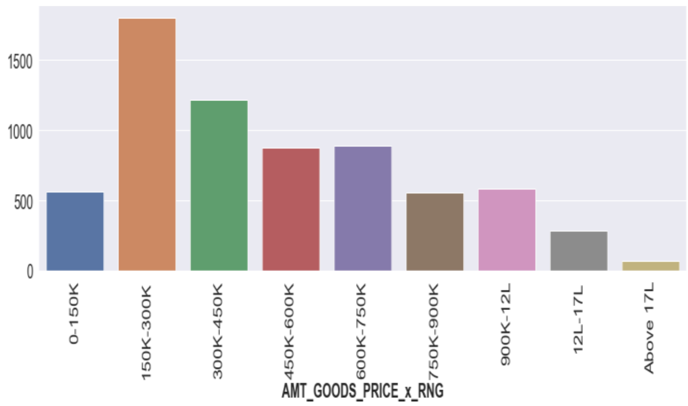
8. Univariate, Segmented Univariate and Bivariate Analysis

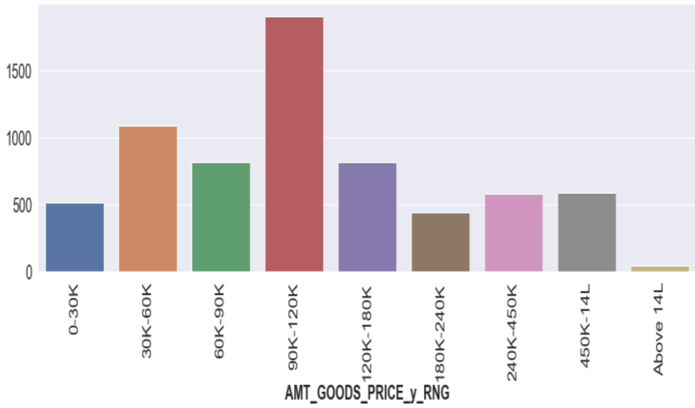
a) Univariate: to understand the distribution of individual variables

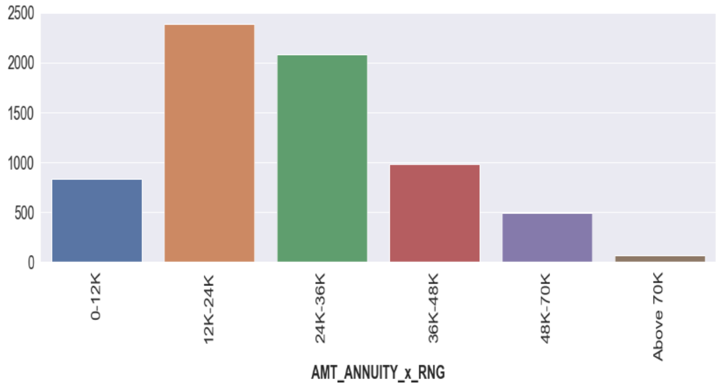
We plot individual variables into countplots to understand the distribution.

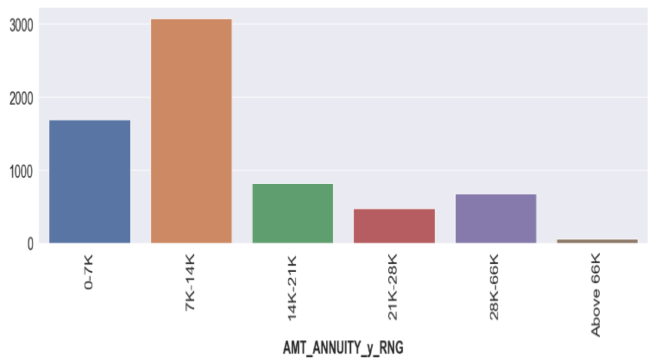
Most of the numerical columns are binned, so analysis can be performed on categorical columns

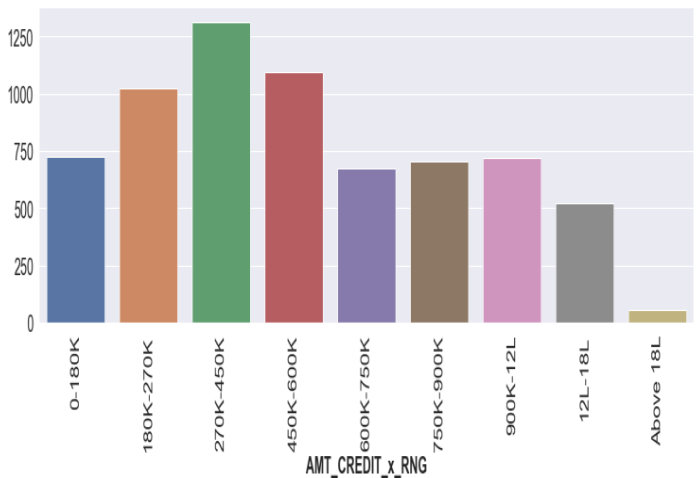


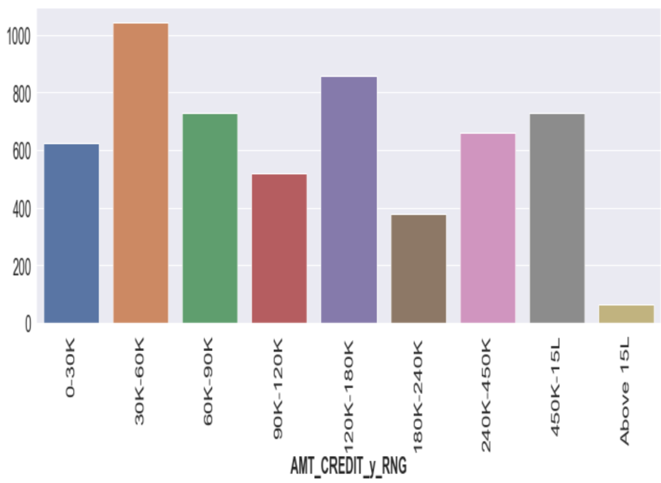


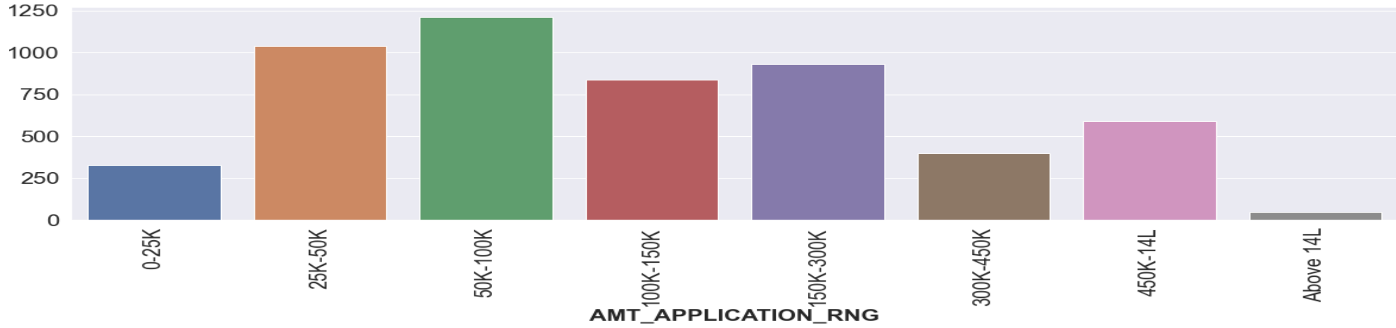


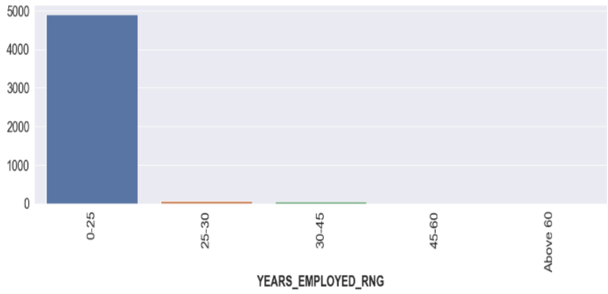


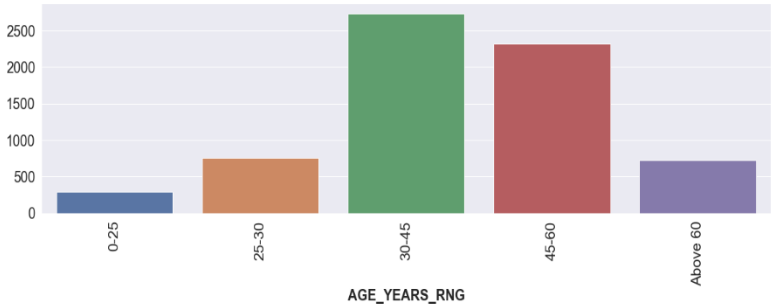


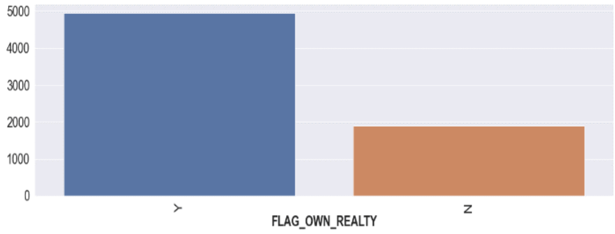


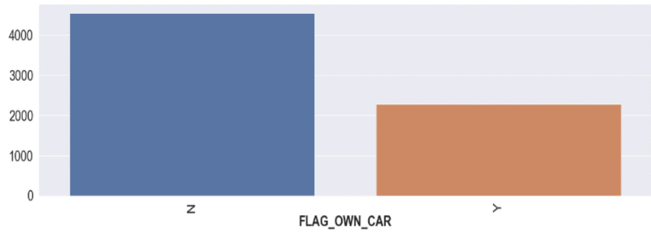


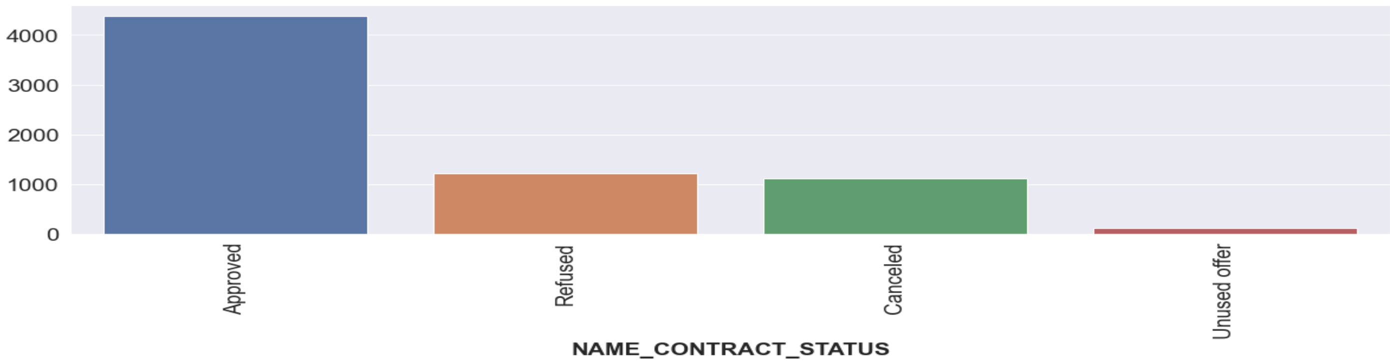

1250
1000
750
500
250
0
0-25K
25K-50K
50K-100K
100K-150K
150K-300K
300K-450K
450K-14L
Above 14L
AMT_APPLICATION_RNG

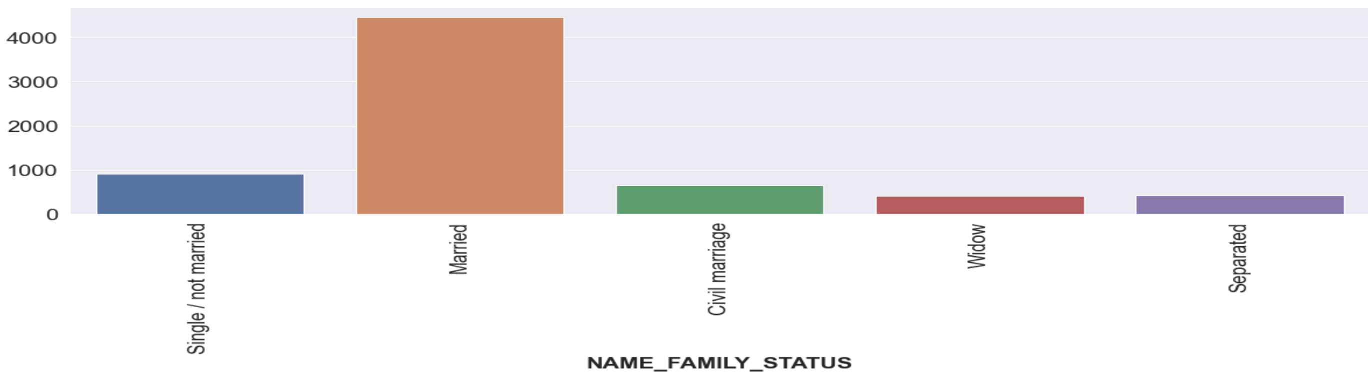

5000
4000
3000
2000
1000
0
0-25
25-30
30-45
45-60
Above 60
YEARS_EMPLOYED_RNG

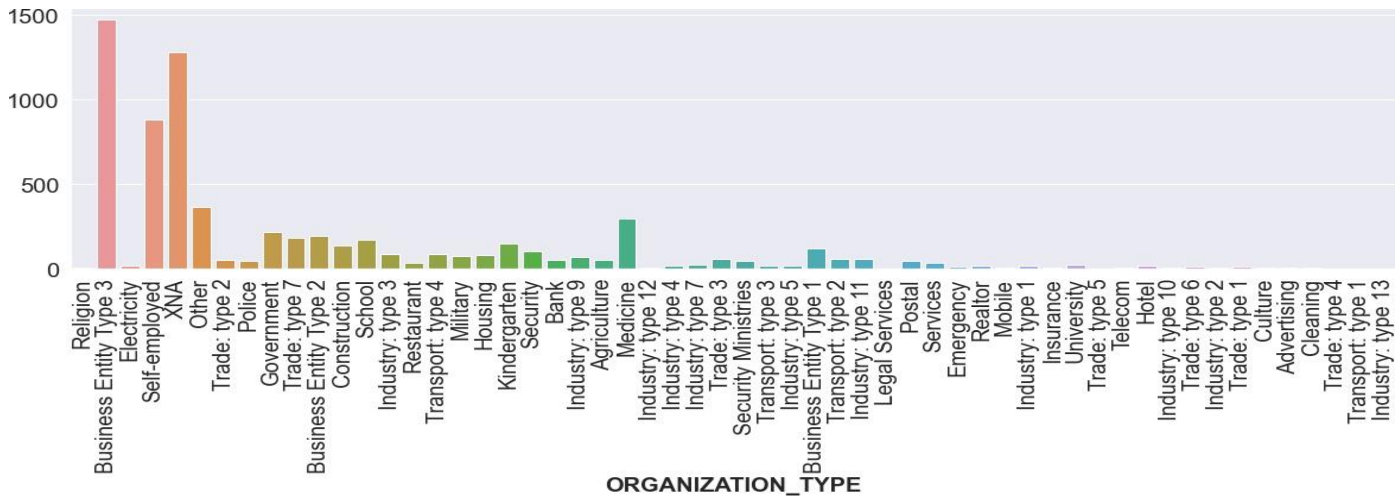

2500
2000
1500
1000
500
0
0-25
25-30
30-45
45-60
Above 60
AGE_YEARS_RNG

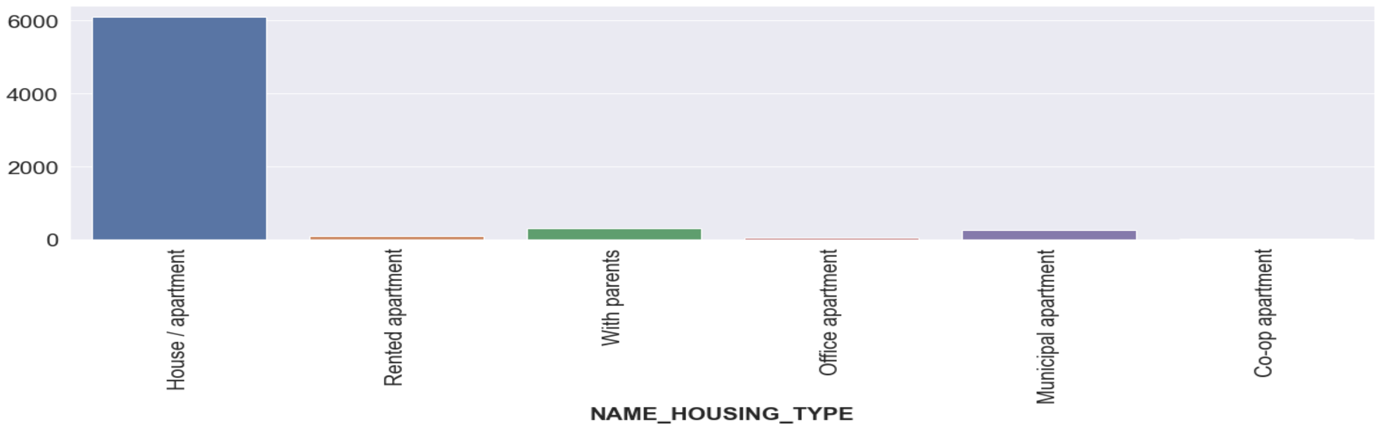

5000
4000
3000
2000
1000
0
Y
N
FLAG_OWN_REALTY

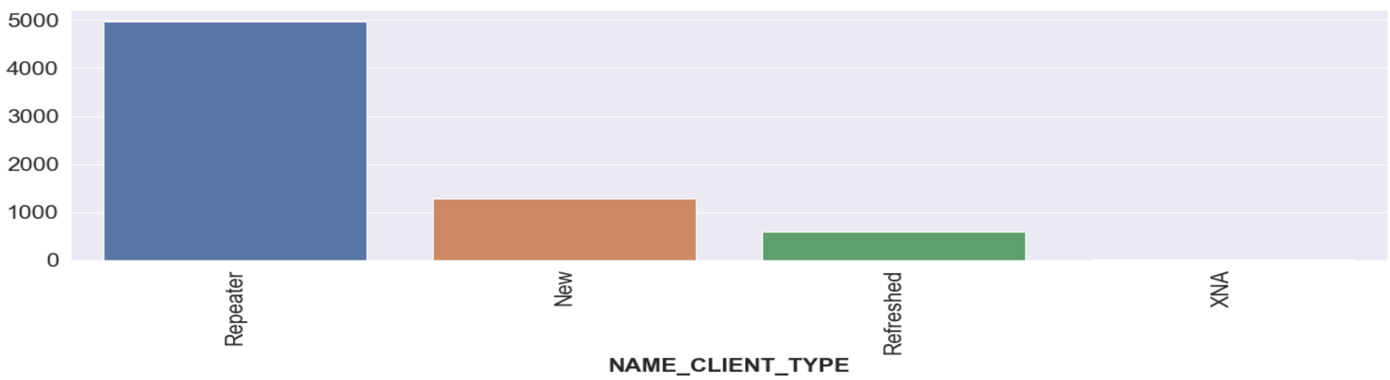

4000
3000
2000
1000
0
N
Y
FLAG_OWN_CAR

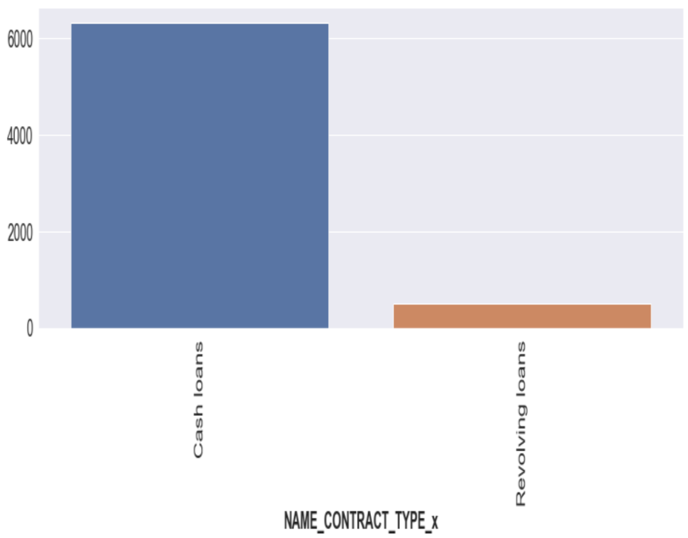

4000
3000
2000
1000
0
Approved
Refused
Canceled
Unused offer
NAME_CONTRACT_STATUS

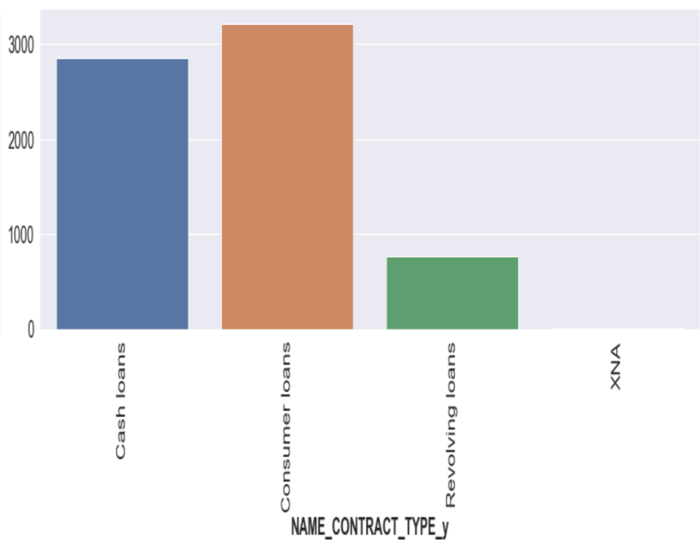

4000
3000
2000
1000
0
Single / not married
Married
Civil marriage
Widow
Separated
NAME_FAMILY_STATUS

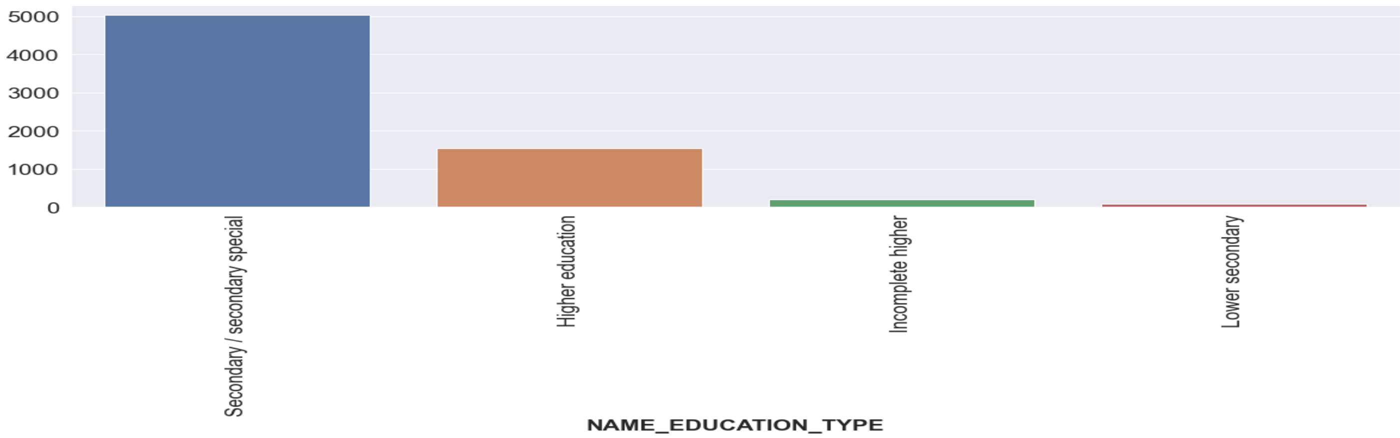

1500
1000
500
0
Religion
Business Entity Type 3
Electricity
Self-employed
XNA
Other
Trade: type 2
Police
Government
Trade: type 7
Business Entity Type 2
Construction
School
Industry: type 3
Restaurant
Transport: type 4
Military
Housing
Kindergarten
Security
Bank
Industry: type 9
Agriculture
Medicine
Industry: type 12
Industry: type 4
Industry: type 7
Trade: type 3
Security Ministries
Transport: type 3
Industry: type 5
Business Entity Type 1
Transport: type 2
Industry: type 11
Legal Services
Postal
Services
Emergency
Realtor
Mobile
Industry: type 1
Insurance
University
Trade: type 5
Telecom
Hotel
Industry: type 10
Trade: type 6
Industry: type 2
Trade: type 1
Culture
Advertising
Cleaning
Trade: type 4
Transport: type 1
Industry: type 13
ORGANIZATION_TYPE

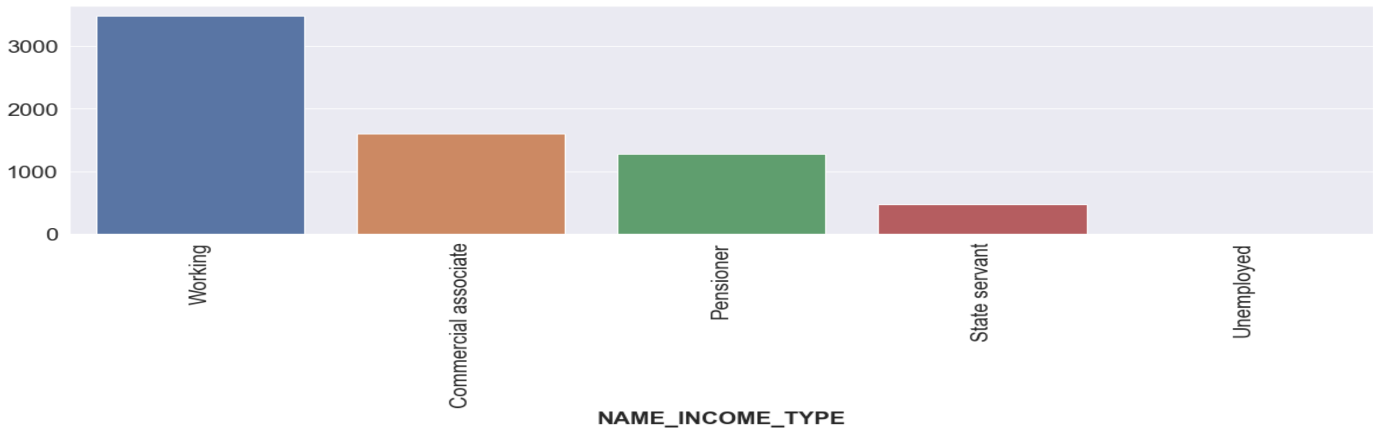

6000
4000
2000
0
House / apartment
Rented apartment
With parents
Office apartment
Municipal apartment
Co-op apartment
NAME_HOUSING_TYPE

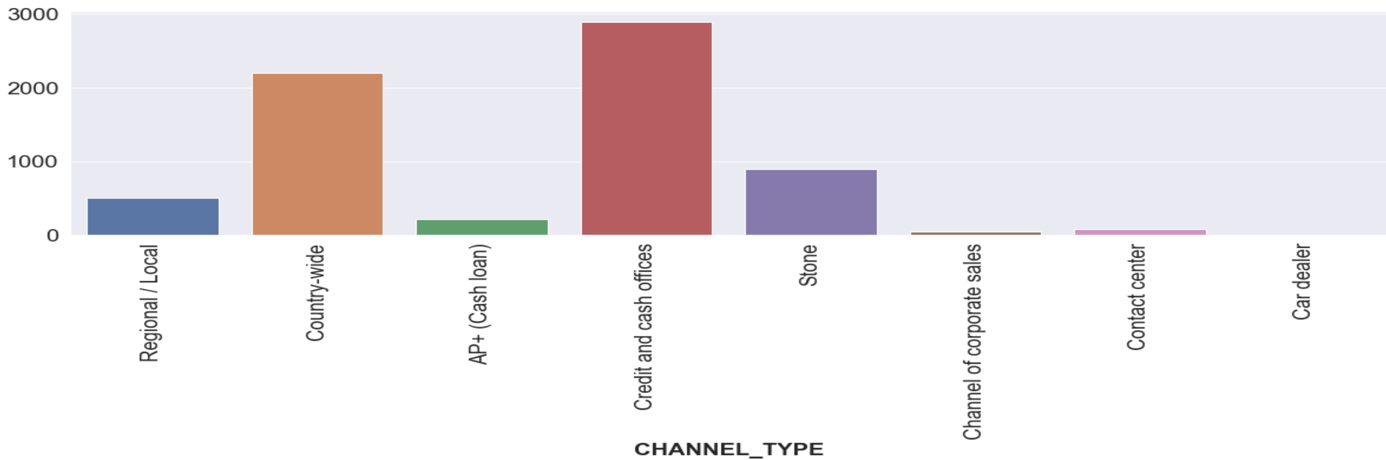

5000
4000
3000
2000
1000
0
Repeater
New
Refreshed
XNA
NAME_CLIENT_TYPE

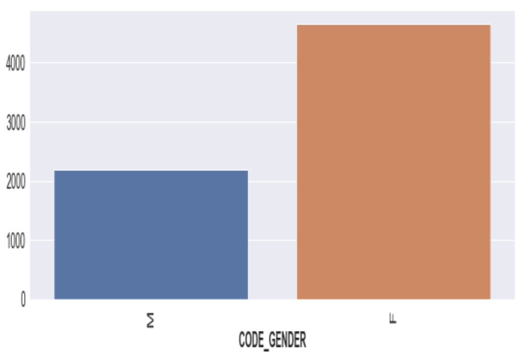

6000
4000
2000
0
Cash loans
Revolving loans
NAME_CONTRACT_TYPE_x

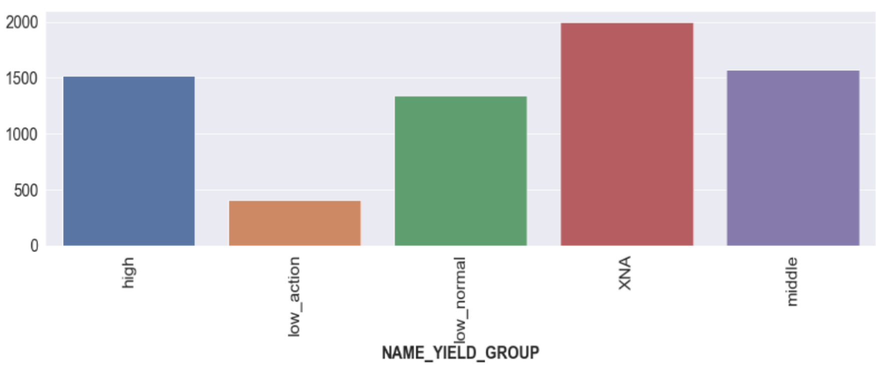

3000
2000
1000
0
Cash loans
Consumer loans
Revolving loans
XNA
NAME_CONTRACT_TYPE_y


5000
4000
3000
2000
1000
0
Secondary / secondary special
Higher education
Incomplete higher
Lower secondary
NAME_EDUCATION_TYPE


3000
2000
1000
0
Working
Commercial associate
Pensioner
State servant
Unemployed
NAME_INCOME_TYPE


3000
2000
1000
0
Regional / Local
Country-wide
AP+ (Cash loan)
Credit and cash offices
Stone
Channel of corporate sales
Contact center
Car dealer
CHANNEL_TYPE


4000
3000
2000
1000
0
M
F
CODE_GENDER


2000
1500
1000
500
0
high
low_action
low_normal
XNA
middle
NAME_YIELD_GROUP

Insights comparing current data with previous data with respect to clients:

>AMT_GOODS_PRICE range increased from 90K- 120K in previous application data to 150K- 300K in current application data

>AMT_ANNUITY range increased from 7K-14K to 12K-24K

>AMT_CREDIT range increased from 30K-60K 270K-450K

> Number of Cash loans increased as compared to previous_applications

Overall Insights states that most of the clients in the provided data have:

>Income range of 115K-186K, AMT_APPLICATION range is between 50K-100K, fall under 30-45 years of age group

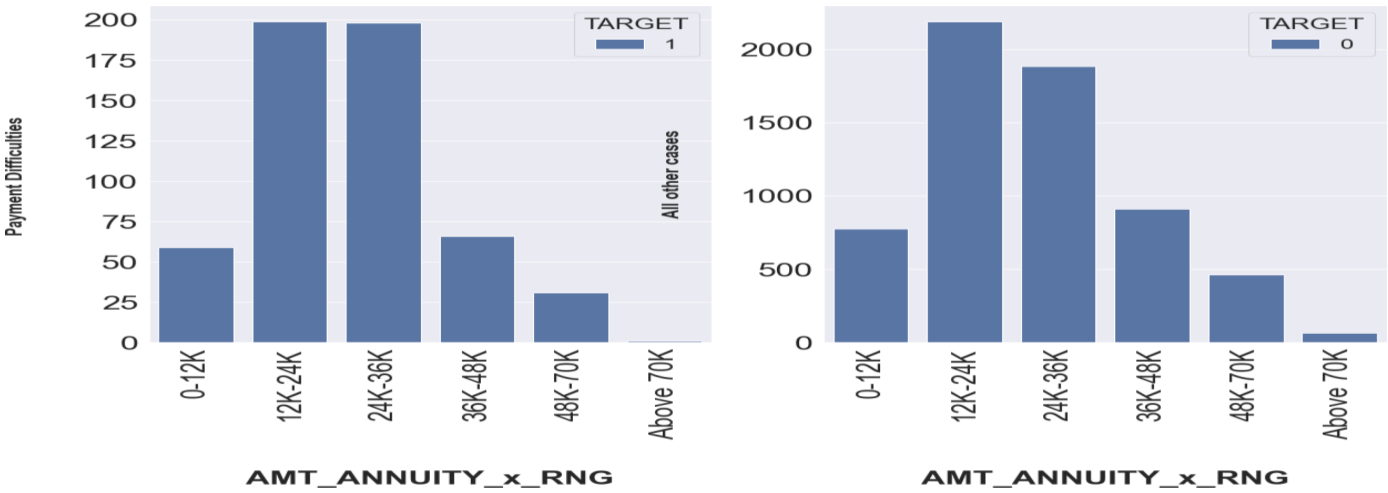
>Most clients own a property(i.e real estate or house) but don't own a car, most of the clients are married, own a house/apartment, are repeaters, have secondary/secondary special education and are from working class

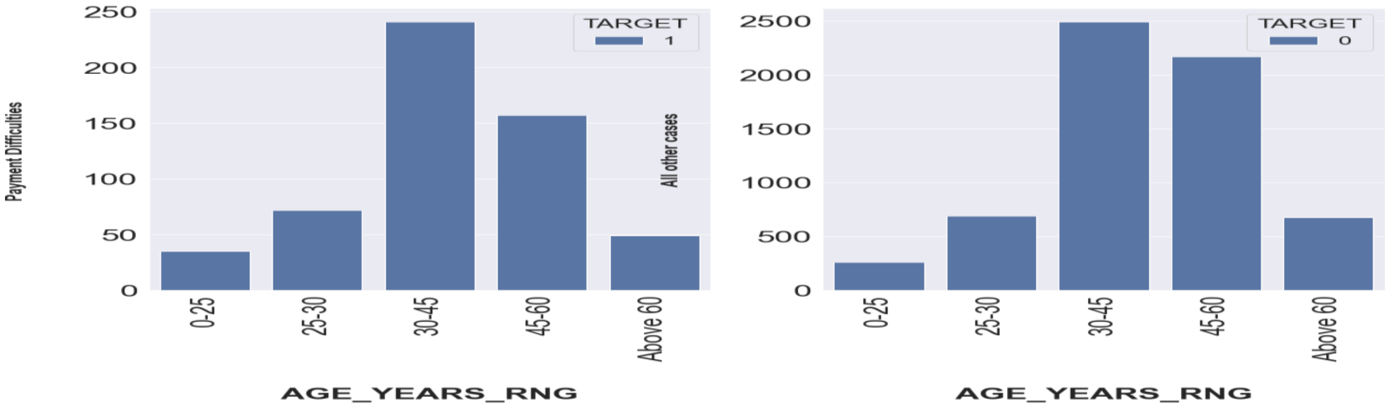
>The data is highly imbalanced for gender, as most of the data is for female clients, whose channel type is credit and cash offices, have middle yield group

>Most of the clients are Pensioners and the organization type for most clients is Business entity 3

b) Segmented Univariate: compare variable distributions for different scenarios

The scenario we consider for this analysis is TARGET variable, as the aim of this project is to identify factors of defaultors





Insights:

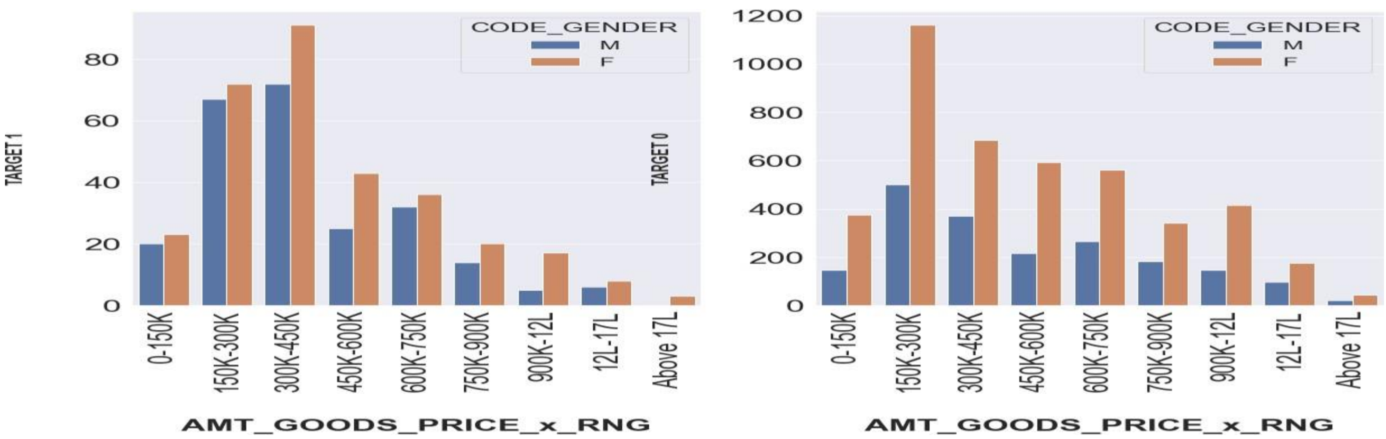
Defaultors or client with payment difficulties have:

> AMT_GOODS_PRICE range between 300K-450K

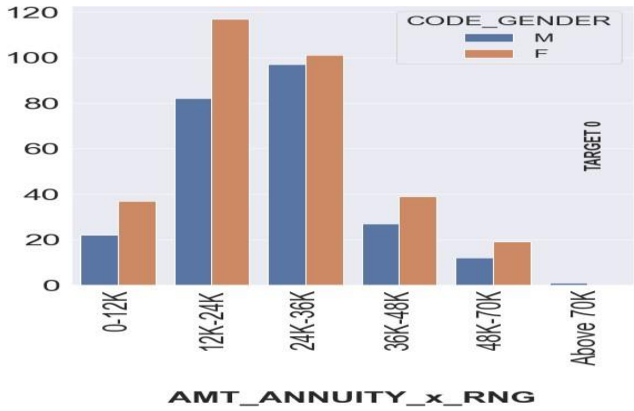
> AMT_ANNUITY range between 24K-36K

> Are of age between 30-45

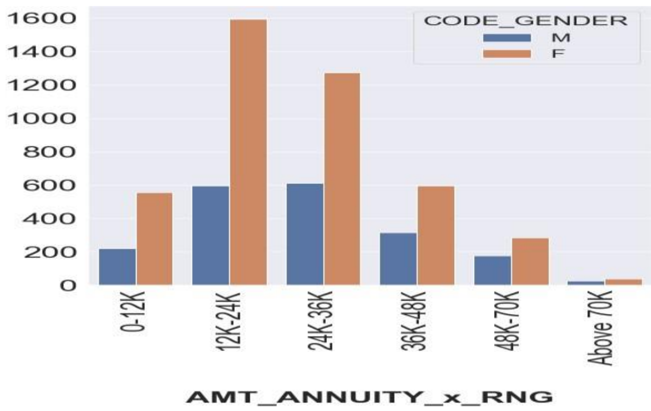
c) Bivariate Analysis: explore relationships between variables and the target variable



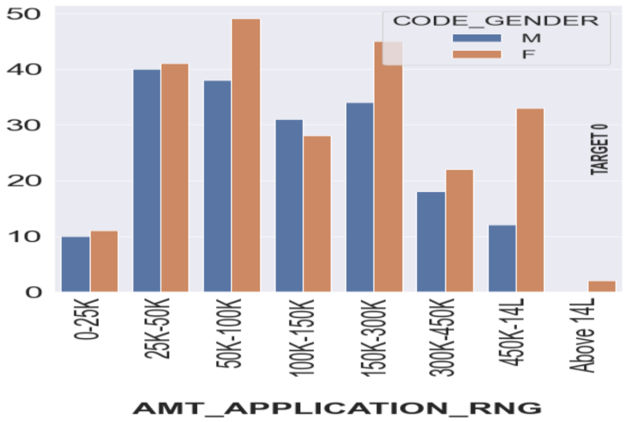
TARGET 1



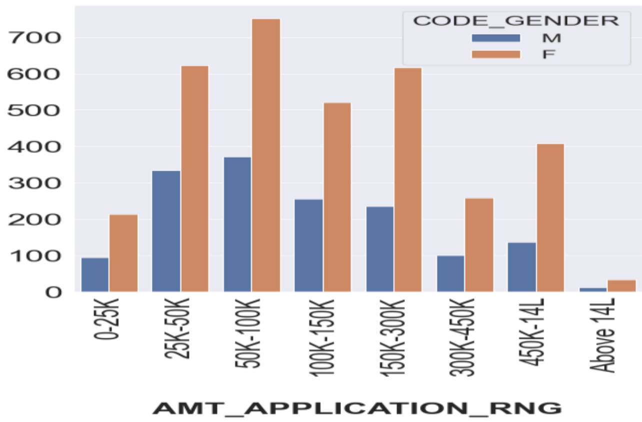
TARGET 0



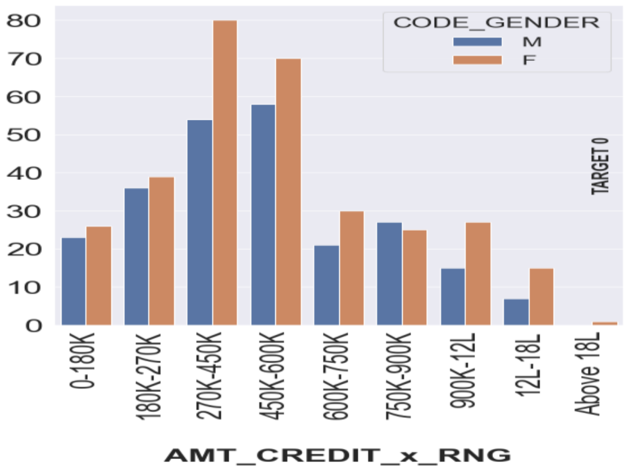
TARGET 1



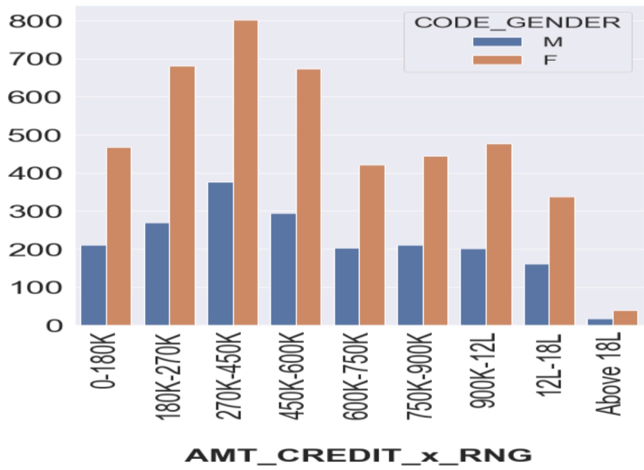
TARGET 0

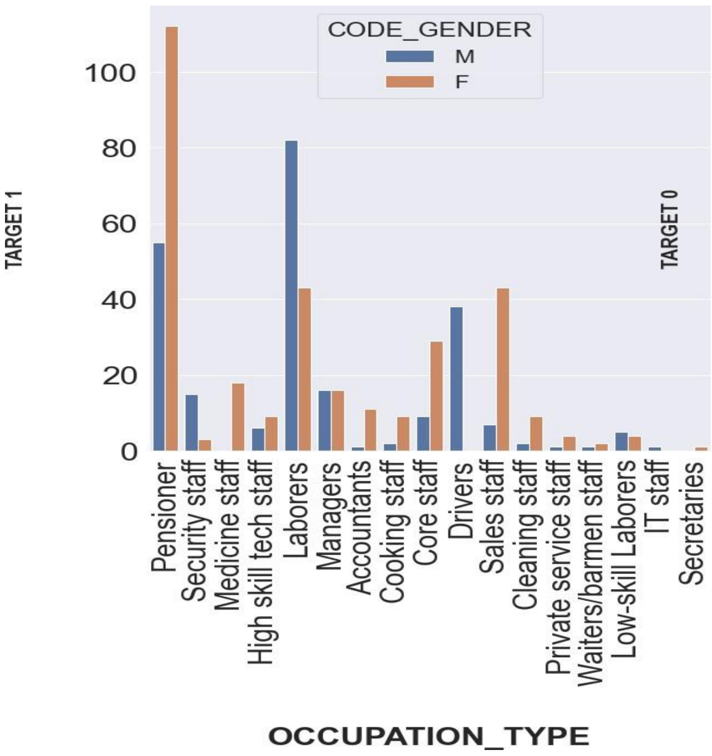


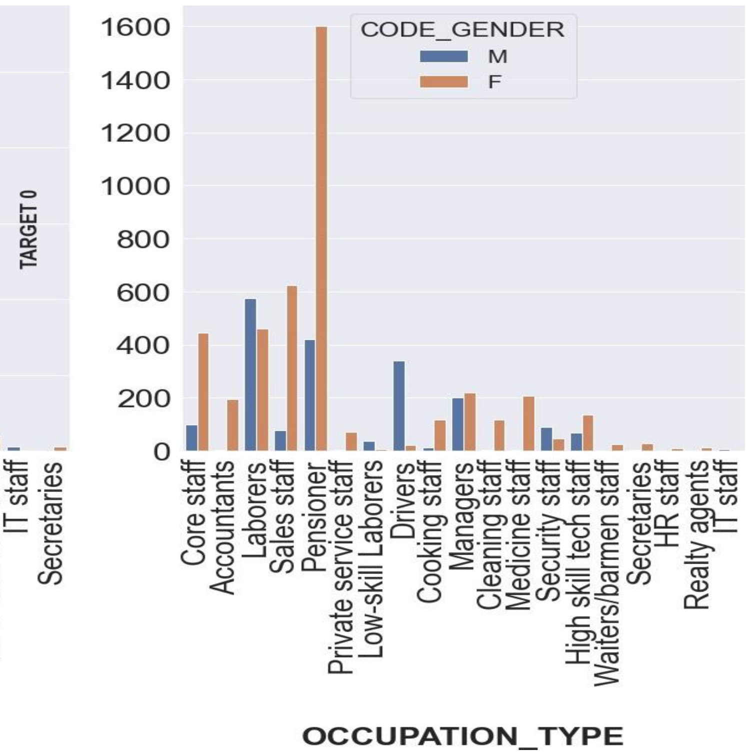
TARGET 1

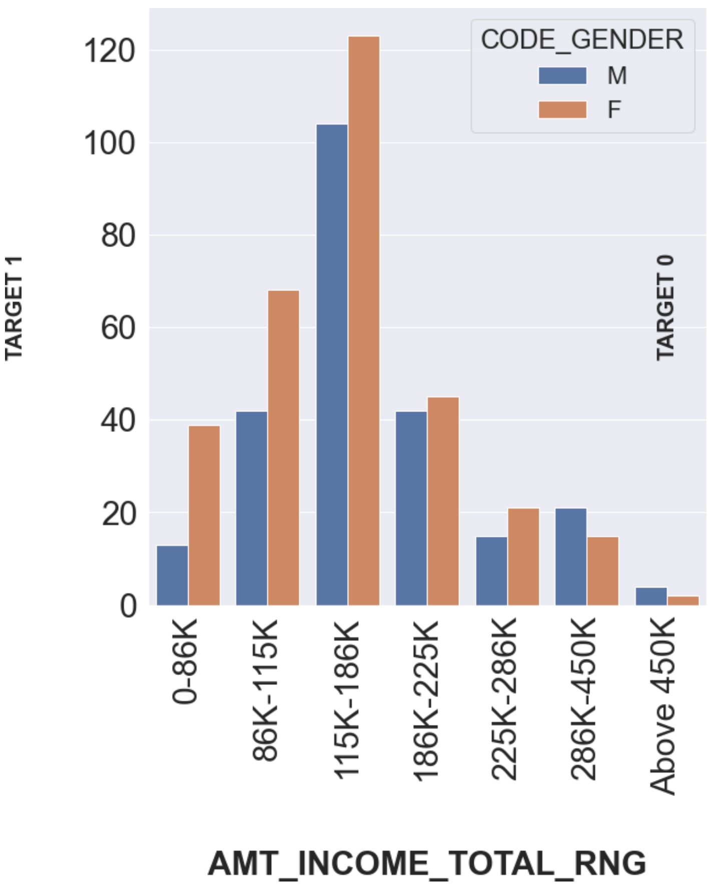


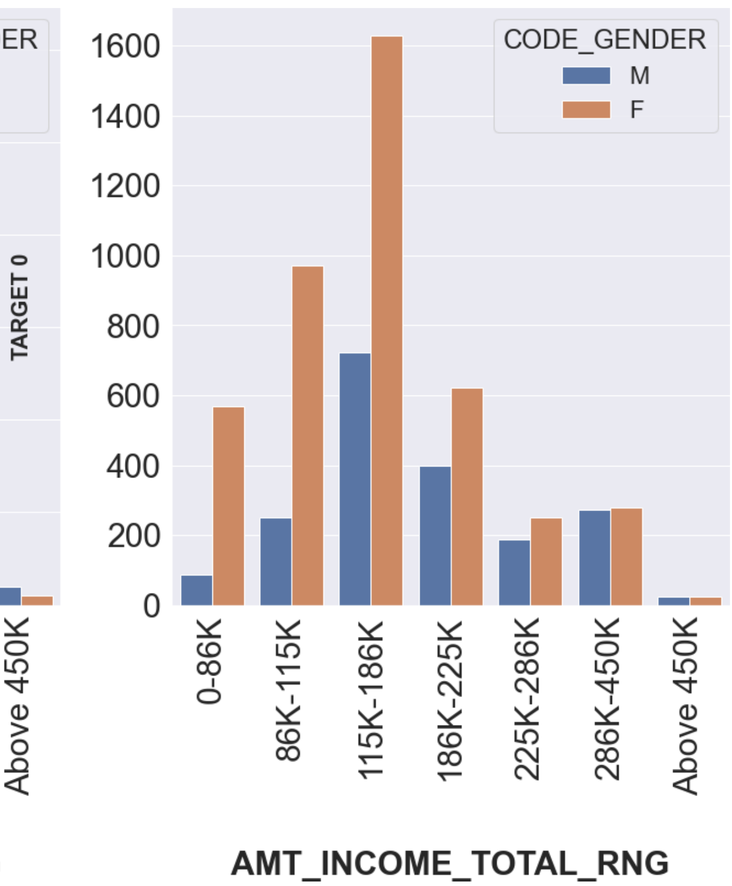
TARGET 0











Defaultors or client with payment difficulties insights:

>For AMT_GOODS_PRICE both male and female range between 300K-450K

>AMT_ANNUITY range for male clients is 24K-36K and for females is between 12K-24K

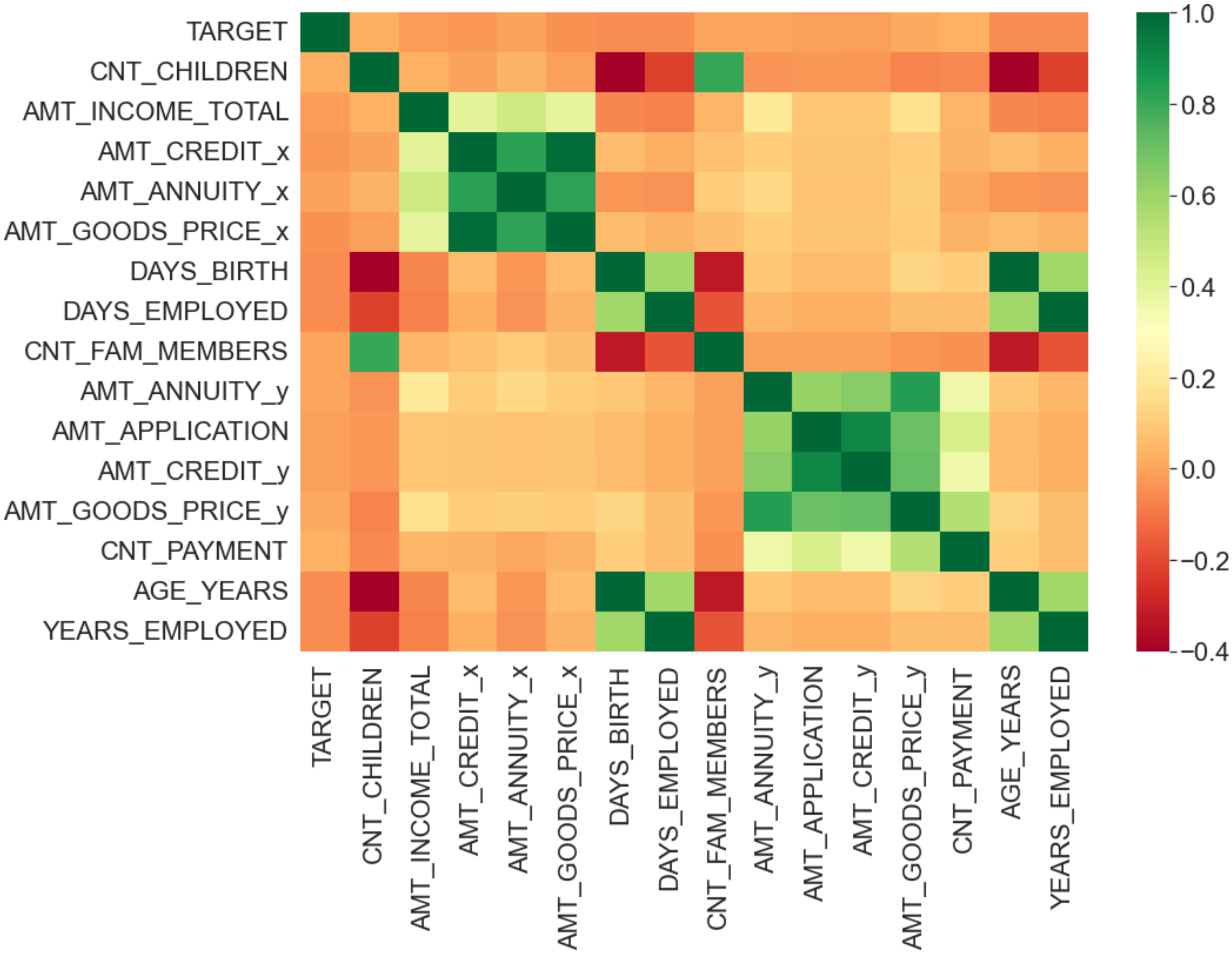
>AMT_APPLICATION range for male clients is between 25K- 50K and for females is between 50K-100K

>AMT_CREDIT range for female clients is between 270K-450K and for male clients is between 450K-600K

>Most of the defaulter males are laborer’s and females are pensioners

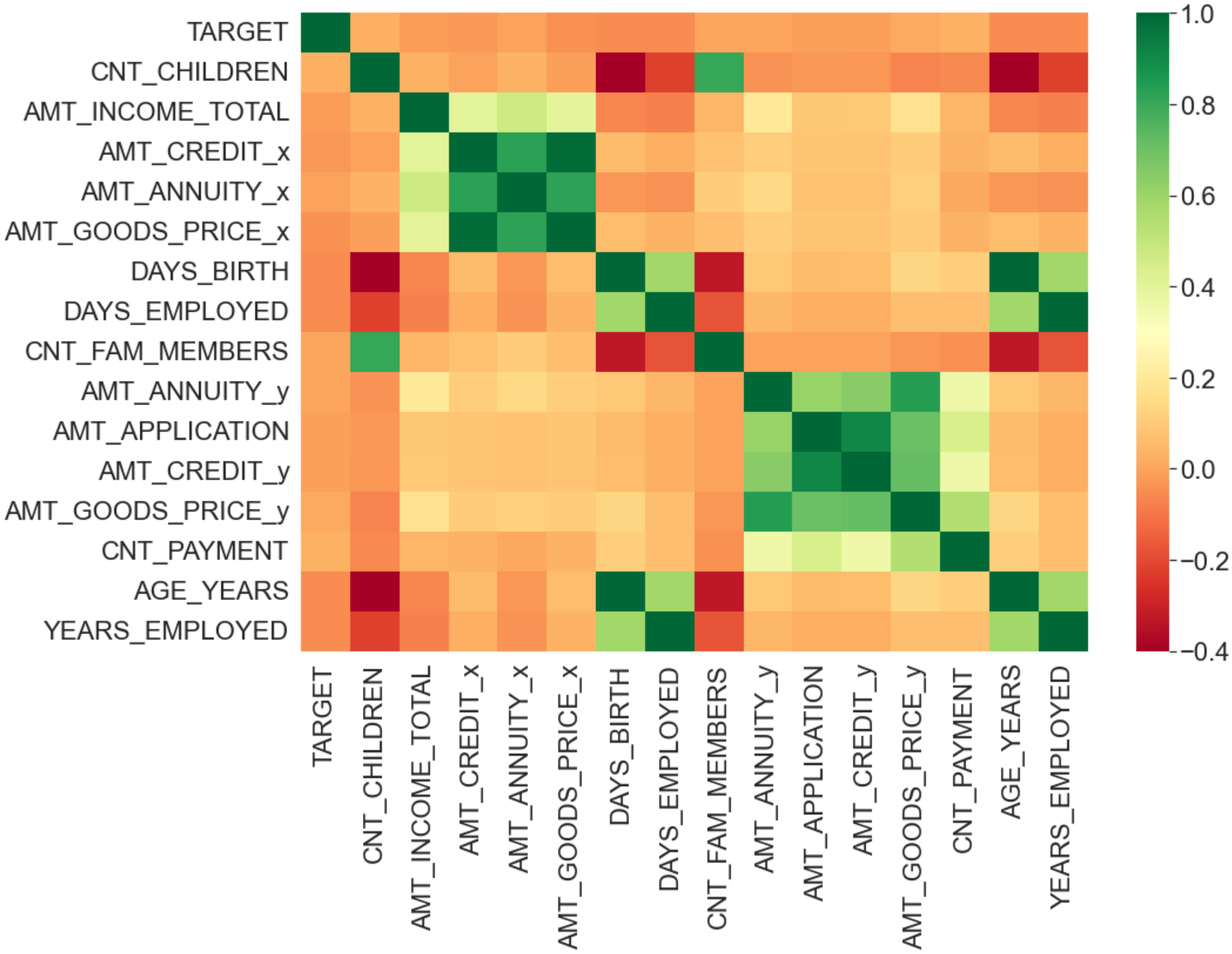
9. Identify Top Correlations for Different Scenarios:

Target0:



	VAR1	VAR2	CORRELATION	CORR_ABS
247	YEARS_EMPLOYED	DAYS_EMPLOYED	1.000000	1.000000
230	AGE_YEARS	DAYS_BIRTH	0.999702	0.999702
202	AMT_GOODS_PRICE_y	AMT_APPLICATION	0.989047	0.989047
83	AMT_GOODS_PRICE_x	AMT_CREDIT_x	0.986664	0.986664
186	AMT_CREDIT_y	AMT_APPLICATION	0.974410	0.974410
203	AMT_GOODS_PRICE_y	AMT_CREDIT_y	0.972064	0.972064
129	CNT_FAM_MEMBERS	CNT_CHILDREN	0.879420	0.879420
201	AMT_GOODS_PRICE_y	AMT_ANNUITY_y	0.828543	0.828543
185	AMT_CREDIT_y	AMT_ANNUITY_y	0.825461	0.825461
169	AMT_APPLICATION	AMT_ANNUITY_y	0.818092	0.818092

Target1:



	VAR1	VAR2	CORRELATION	CORR_ABS
247	YEARS_EMPLOYED	DAYS_EMPLOYED	1.000000	1.000000
230	AGE_YEARS	DAYS_BIRTH	0.999708	0.999702
202	AMT_GOODS_PRICE_y	AMT_APPLICATION	0.986720	0.989047
83	AMT_GOODS_PRICE_x	AMT_CREDIT_x	0.979949	0.986664
186	AMT_CREDIT_y	AMT_APPLICATION	0.973956	0.974410
203	AMT_GOODS_PRICE_y	AMT_CREDIT_y	0.966819	0.972064
129	CNT_FAM_MEMBERS	CNT_CHILDREN	0.885949	0.879420
201	AMT_GOODS_PRICE_y	AMT_ANNUITY_y	0.795789	0.828543
185	AMT_CREDIT_y	AMT_ANNUITY_y	0.789420	0.825461
169	AMT_APPLICATION	AMT_ANNUITY_y	0.780680	0.818092

Insights:

>For both Target0 & Target1, correlation between variables is same

>Years employed and Days employed has highest correlation

10. Final Conclusion:

>As the data is highly imbalanced, its difficult to identify the actual factors eading to default.

>Based on the assumptions and the analysis done on the given dataset,

->Clients of age between 30-45, with income range between 115k-186k, having AMT_GOODS_PRICE range between 300K-450K, AMT_CREDIT range for female clients between 270K-450K and for male clients between 450K-600K,

with AMT_APPLICATION range for male clients 25K- 50K and for females 50K-100K, AMT_ANNUITY range for male clients 24K-36K and for females between 12K-24K and those clients in case of females who are pensioners and males are labourers are likely to default

-> Banks should focus less on income type Working as they are having the greatest number of unsuccessful

payments.

-> Applicants living in House/Apartments has the highest number of loan application

11. Result:

This project helped gain the knowledge and hands-on experience on risk analytics also on statistics.
It also helped gain more training on using python for data analysis, using and plotting graphs and statistical functions

12. Links to work files:

Jupyter Notebook file:
https://github.com/dhanashrikandre/data_analytics_projects/blob/main/Bank_Loan_Analytics/src.ipynb

Occupation type.csv work file:
https://github.com/dhanashrikandre/data_analytics_projects/blob/main/Bank_Loan_Analytics/occupationtype.csv

Datasets:
Application_data: https://drive.google.com/file/d/15tzIOcnZM7f9xbJxkVENWQgpSK6RE8MO/view?usp=drive_link
Previous_application: https://drive.google.com/file/d/1JQDSzpTAJeiAZjPrj6IQ2Zh6ysHw_qoY/view?usp=drive_link
Column_description: https://drive.google.com/file/d/1fLaYi1pVp810TaHhwnNINC8uf0dicQtp/view?usp=drive_link