BHARTIYA VIDYA BHAVANS

SARDAR PATEL INSTITUTE OF TECHNOLOGY

BIG DATA ANALYTICS lab

Mini Project Phase 1

GROUP MEMBERS :

Shreya Shetty (2019140059)

Shruti Shetty (2019140060)

AIM : To prepare Exploratory Data Analysis Report on chosen Dataset

DATASET : Credit Case Study

DATASET LINK : **https://www.kaggle.com/code/venkatasubramanian/credit-eda-case-study-analysis/data**

DESCRIPTION :

This dataset has 3 files as explained below:

1. 'application_data.csv' contains all the information of the client at the time of application. The data is about whether a client has payment difficulties.
2. 'previous_application.csv' contains information about the client's previous loan data. It contains the data whether the previous application had been Approved, Cancelled, Refused or Unused offer.

PROBLEM STATEMENT :

Apply EDA on the Credit Dataset and develop a basic understanding of risk analytics in banking and financial services and understand how data is used to minimise the risk of losing money while lending to customers. Identify patterns which indicate if a client has difficulty paying their instalments which may be used for taking actions such as denying the loan, reducing the amount of loan, lending (to risky applicants) at a higher interest rate, etc. This will ensure that the consumers capable of repaying the loan are not rejected. Identification of such applicants using EDA is useful for portfolio and risk assessment.

PREVIOUS_APPLICATION.CSV:

- Shape : Shows number of rows and columns

```
previousApplication.shape
```

```
(1421211, 37)
```

- Head : Shows first few records

```
previousApplication.head()
```

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT	AMT_GOODS_PRICE	WEEKDAY_APPR_PROCESS_START	HOUR_APPR_PROCESS_
0	2030495	271877	Consumer loans	1730.430	17145.0	17145.0	0.0	17145.0	SATURDAY	
1	2802425	108129	Cash loans	25188.615	607500.0	679671.0	NaN	607500.0	THURSDAY	
2	2523466	122040	Cash loans	15060.735	112500.0	136444.5	NaN	112500.0	TUESDAY	
3	2819243	176158	Cash loans	47041.335	450000.0	470790.0	NaN	450000.0	MONDAY	
4	1784265	202054	Cash loans	31924.395	337500.0	404055.0	NaN	337500.0	THURSDAY	

5 rows × 37 columns

- Information : Shows information of data including datatypes and whether there are null values

```
previousApplication.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1421211 entries, 0 to 1421210
Data columns (total 37 columns):
 #   Column                       Non-Null Count    Dtype
---  ------                       --------------    -----
 0   SK_ID_PREV                   1421211 non-null  int64
 1   SK_ID_CURR                   1421211 non-null  int64
 2   NAME_CONTRACT_TYPE           1421211 non-null  object
 3   AMT_ANNUITY                  1105616 non-null  float64
 4   AMT_APPLICATION              1421211 non-null  float64
 5   AMT_CREDIT                   1421210 non-null  float64
 6   AMT_DOWN_PAYMENT             663145 non-null   float64
 7   AMT_GOODS_PRICE              1094755 non-null  float64
 8   WEEKDAY_APPR_PROCESS_START   1421211 non-null  object
 9   HOUR_APPR_PROCESS_START      1421211 non-null  int64
 10  FLAG_LAST_APPL_PER_CONTRACT  1421211 non-null  object
 11  NFLAG_LAST_APPL_IN_DAY       1421211 non-null  int64
 12  RATE_DOWN_PAYMENT            663145 non-null   float64
 13  RATE_INTEREST_PRIMARY        5114 non-null     float64
 14  RATE_INTEREST_PRIVILEGED     5114 non-null     float64
 15  NAME_CASH_LOAN_PURPOSE       1421211 non-null  object
 16  NAME_CONTRACT_STATUS         1421211 non-null  object
 17  DAYS_DECISION                1421211 non-null  int64
 18  NAME_PAYMENT_TYPE            1421211 non-null  object
 19  CODE_REJECT_REASON           1421211 non-null  object
 20  NAME_TYPE_SUITE              723810 non-null   object
 21  NAME_CLIENT_TYPE             1421211 non-null  object
 22  NAME_GOODS_CATEGORY          1421210 non-null  object
 23  NAME_PORTFOLIO               1421210 non-null  object
 24  NAME_PRODUCT_TYPE            1421210 non-null  object
 25  CHANNEL_TYPE                 1421210 non-null  object
 26  SELLERPLACE_AREA             1421210 non-null  float64
 27  NAME_SELLER_INDUSTRY         1421210 non-null  object
 28  CNT_PAYMENT                  1105619 non-null  float64
 29  NAME_YIELD_GROUP             1421210 non-null  object
 30  PRODUCT_COMBINATION          1420917 non-null  object
 31  DAYS_FIRST_DRAWING           850901 non-null   float64
 32  DAYS_FIRST_DUE               850901 non-null   float64
 33  DAYS_LAST_DUE_1ST_VERSION    850901 non-null   float64
 34  DAYS_LAST_DUE                850901 non-null   float64
 35  DAYS_TERMINATION             850901 non-null   float64
 36  NFLAG_INSURED_ON_APPROVAL    850901 non-null   float64
dtypes: float64(16), int64(5), object(16)
memory usage: 401.2+ MB
```

- isnull().sum().: returns the number of missing values in the data set.

```
previousApplication.isnull().sum()
```

```
SK_ID_PREV                          0
SK_ID_CURR                          0
NAME_CONTRACT_TYPE                  0
AMT_ANNUITY                    315595
AMT_APPLICATION                     0
AMT_CREDIT                          1
AMT_DOWN_PAYMENT               758066
AMT_GOODS_PRICE                326456
WEEKDAY_APPR_PROCESS_START          0
HOUR_APPR_PROCESS_START             0
FLAG_LAST_APPL_PER_CONTRACT         0
NFLAG_LAST_APPL_IN_DAY              0
RATE_DOWN_PAYMENT              758066
RATE_INTEREST_PRIMARY         1416097
RATE_INTEREST_PRIVILEGED      1416097
NAME_CASH_LOAN_PURPOSE              0
NAME_CONTRACT_STATUS                0
DAYS_DECISION                       0
NAME_PAYMENT_TYPE                   0
CODE_REJECT_REASON                  0
NAME_TYPE_SUITE                697401
NAME_CLIENT_TYPE                    0
NAME_GOODS_CATEGORY                 1
NAME_PORTFOLIO                      1
NAME_PRODUCT_TYPE                   1
CHANNEL_TYPE                        1
SELLERPLACE_AREA                    1
NAME_SELLER_INDUSTRY                1
CNT_PAYMENT                    315592
NAME_YIELD_GROUP                    1
PRODUCT_COMBINATION               294
DAYS_FIRST_DRAWING             570310
DAYS_FIRST_DUE                 570310
DAYS_LAST_DUE_1ST_VERSION      570310
DAYS_LAST_DUE                  570310
DAYS_TERMINATION               570310
NFLAG_INSURED_ON_APPROVAL      570310
dtype: int64
```

- Finding percentage of missing values

```
#percentage of missing values
(previousApplication.isnull().sum()/len(previousApplication.index))*100
```

```
SK_ID_PREV                      0.000000
SK_ID_CURR                      0.000000
NAME_CONTRACT_TYPE              0.000000
AMT_ANNUITY                    22.206062
AMT_APPLICATION                 0.000000
AMT_CREDIT                      0.000070
AMT_DOWN_PAYMENT               53.339441
AMT_GOODS_PRICE                22.970270
WEEKDAY_APPR_PROCESS_START      0.000000
HOUR_APPR_PROCESS_START         0.000000
FLAG_LAST_APPL_PER_CONTRACT     0.000000
NFLAG_LAST_APPL_IN_DAY          0.000000
RATE_DOWN_PAYMENT              53.339441
RATE_INTEREST_PRIMARY          99.640166
RATE_INTEREST_PRIVILEGED       99.640166
NAME_CASH_LOAN_PURPOSE          0.000000
NAME_CONTRACT_STATUS            0.000000
DAYS_DECISION                   0.000000
NAME_PAYMENT_TYPE               0.000000
CODE_REJECT_REASON              0.000000
NAME_TYPE_SUITE                49.070898
NAME_CLIENT_TYPE                0.000000
NAME_GOODS_CATEGORY             0.000070
NAME_PORTFOLIO                  0.000070
NAME_PRODUCT_TYPE               0.000070
CHANNEL_TYPE                    0.000070
SELLERPLACE_AREA                0.000070
NAME_SELLER_INDUSTRY            0.000070
CNT_PAYMENT                    22.205851
NAME_YIELD_GROUP                0.000070
PRODUCT_COMBINATION             0.020687
DAYS_FIRST_DRAWING             40.128454
DAYS_FIRST_DUE                 40.128454
DAYS_LAST_DUE_1ST_VERSION      40.128454
DAYS_LAST_DUE                  40.128454
DAYS_TERMINATION               40.128454
NFLAG_INSURED_ON_APPROVAL      40.128454
dtype: float64
```

Inference: Too many columns have missing data. Hence, we can Impute values in columns having % of missing values < 50 or Delete columns having more than 50% missing values as they cannot help in analysis.

DATA CLEANING OF PREVIOUS_APPLICATION.CSV :

- Deleting columns with missing values more than 50% and then displaying the percentage of missing values from remaining columns

```
#Deleting columns with missing values > 50%
previousApplication = previousApplication.drop(["AMT_DOWN_PAYMENT", "RATE_INTEREST_PRIMARY", "RATE_DOWN_PAYMENT", "RATE_INTEREST_PRIVILEGED"], axis=1)
```

```
#finding the percentage of missing values
(previousApplication.isnull().sum()/len(previousApplication.index))*100
```

```
SK_ID_PREV                     0.000000
SK_ID_CURR                     0.000000
NAME_CONTRACT_TYPE             0.000000
AMT_ANNUITY                   22.286062
AMT_APPLICATION                0.000000
AMT_CREDIT                     0.000070
AMT_GOODS_PRICE               22.970270
WEEKDAY_APPR_PROCESS_START     0.000000
HOUR_APPR_PROCESS_START        0.000000
FLAG_LAST_APPL_PER_CONTRACT    0.000000
NFLAG_LAST_APPL_IN_DAY         0.000000
NAME_CASH_LOAN_PURPOSE         0.000000
NAME_CONTRACT_STATUS           0.000000
DAYS_DECISION                  0.000000
NAME_PAYMENT_TYPE              0.000000
CODE_REJECT_REASON             0.000000
NAME_TYPE_SUITE               49.070898
NAME_CLIENT_TYPE               0.000000
NAME_GOODS_CATEGORY            0.000070
NAME_PORTFOLIO                 0.000070
NAME_PRODUCT_TYPE              0.000070
CHANNEL_TYPE                   0.000070
SELLERPLACE_AREA               0.000070
NAME_SELLER_INDUSTRY           0.000070
CNT_PAYMENT                   22.285851
NAME_YIELD_GROUP               0.000070
PRODUCT_COMBINATION            0.020687
DAYS_FIRST_DRAWING            40.128454
DAYS_FIRST_DUE                40.128454
DAYS_LAST_DUE_1ST_VERSION     40.128454
DAYS_LAST_DUE                 40.128454
DAYS_TERMINATION              40.128454
NFLAG_INSURED_ON_APPROVAL     40.128454
dtype: float64
```

- TREATING MISSING VALUES (Column wise):
  Firstly, we find the amount of null values that are present. Then we display mean, max, min, count etc. in decimal form using 'describe'. After this a histogram is plotted to see data visually. Using median, we impute missing values as mean would skew the data. After imputation we plot a histogram, we check the spread of the data and for outliers and verify there are no null values.

  - COLUMN: [AMT_ANNUITY]

```
#find the amount of null values
previousApplication["AMT_ANNUITY"].isnull().sum()
```

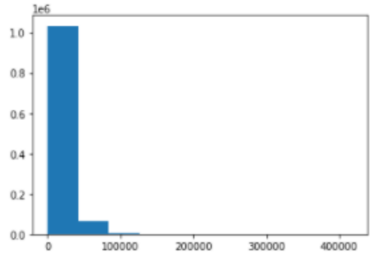
```
315595
```

ıble-click (or enter) to edit

```
pd.options.display.float_format = "{:.2f}".format #to display in decimal form
previousApplication["AMT_ANNUITY"].describe()
```

```
count   1105616.00
mean      15895.26
std       14747.04
min           0.00
25%        6300.00
50%       11250.00
75%       20532.10
max      418058.15
Name: AMT_ANNUITY, dtype: float64
```

```
#plotting a histogram to see data visually
plt.hist(previousApplication["AMT_ANNUITY"])
plt.show()
```




```
#We can see most values lie between min and 50%.
#Using median to impute missing values as mean would skew the data

previousApplication["AMT_ANNUITY"].median(skipna = True)
```

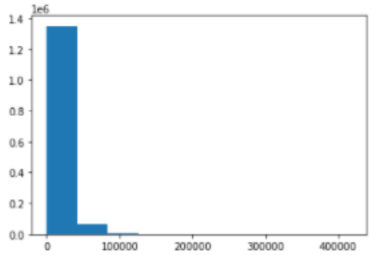
```
11250.0
```

```
previousApplication["AMT_ANNUITY"] = previousApplication["AMT_ANNUITY"].fillna(previousApplication["AMT_ANNUITY"].median(skipna = True))
```

```
pd.options.display.float_format = "{:.2f}".format #to display in decimal form
previousApplication["AMT_ANNUITY"].describe()
```

```
count   1421211.00
mean      14863.73
std       13149.53
min           0.00
25%        7499.81
50%       11250.00
75%       16751.38
max      418058.15
Name: AMT_ANNUITY, dtype: float64
```

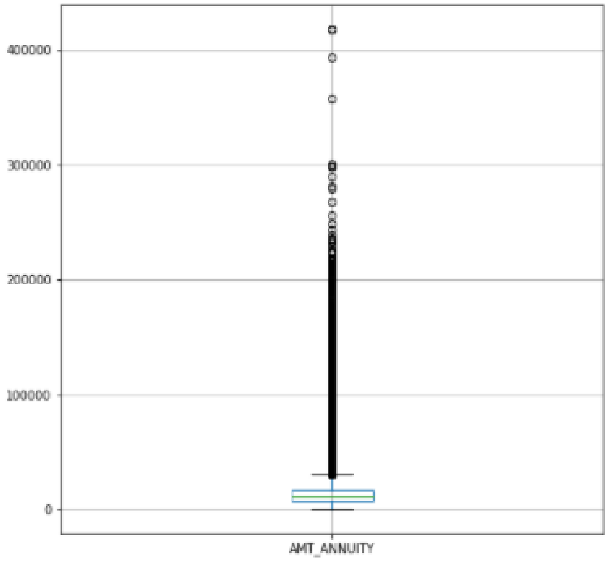
```
#After imputation
plt.hist(previousApplication["AMT_ANNUITY"])
plt.show()
```



Y-axis changed from 1.2 to 1.6 after imputing null values with median

```
[17] #Checking the spread of the data and also for outliers

     previousApplication.boxplot(["AMT_ANNUITY"], figsize=[8,8])
     plt.show()
```



1. Data points seem to be scattered uniformly
2. No need to remove/treat outliers (as also seen in the displot, values are mostly continuous)

```
previousApplication.isnull().sum() #verifying there are no null values in AMT_ANNUITY after imputation
```

```
SK_ID_PREV                          0
SK_ID_CURR                          0
NAME_CONTRACT_TYPE                  0
AMT_ANNUITY                         0
AMT_APPLICATION                     0
AMT_CREDIT                          1
AMT_GOODS_PRICE                326456
WEEKDAY_APPR_PROCESS_START          0
HOUR_APPR_PROCESS_START             0
FLAG_LAST_APPL_PER_CONTRACT         0
NFLAG_LAST_APPL_IN_DAY              0
NAME_CASH_LOAN_PURPOSE              0
NAME_CONTRACT_STATUS                0
DAYS_DECISION                       0
NAME_PAYMENT_TYPE                   0
CODE_REJECT_REASON                  0
NAME_TYPE_SUITE                697401
NAME_CLIENT_TYPE                    0
NAME_GOODS_CATEGORY                 1
NAME_PORTFOLIO                      1
NAME_PRODUCT_TYPE                   1
CHANNEL_TYPE                        1
SELLERPLACE_AREA                    1
NAME_SELLER_INDUSTRY                1
CNT_PAYMENT                    315592
NAME_YIELD_GROUP                    1
PRODUCT_COMBINATION               294
DAYS_FIRST_DRAWING             570310
DAYS_FIRST_DUE                 570310
DAYS_LAST_DUE_1ST_VERSION      570310
DAYS_LAST_DUE                  570310
DAYS_TERMINATION               570310
NFLAG_INSURED_ON_APPROVAL      570310
dtype: int64
```

- Amt_Goods_price

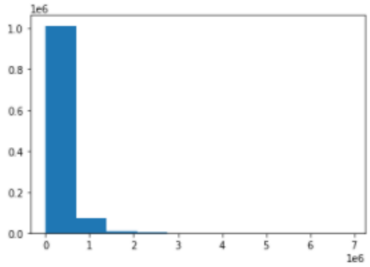
```
previousApplication["AMT_GOODS_PRICE"].describe()
```

```
count   1094755.00
mean     226475.79
std      314156.01
min           0.00
25%       50575.50
50%      111375.00
75%      229500.00
max     6905160.00
Name: AMT_GOODS_PRICE, dtype: float64
```

```
previousApplication["AMT_GOODS_PRICE"].isnull().sum()
```

```
326456
```

```
plt.hist(previousApplication["AMT_GOODS_PRICE"])
plt.show()
```



```
previousApplication["AMT_GOODS_PRICE"].head()
```

```
0    17145.00
1   607500.00
2   112500.00
3   450000.00
4   337500.00
Name: AMT_GOODS_PRICE, dtype: float64
```

```
## IMPUTE WITH MISSING VALUE
```

```
previousApplication["AMT_GOODS_PRICE"].isnull().sum()
```

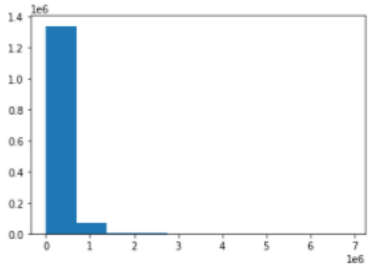
```
326456
```

```
previousApplication["AMT_GOODS_PRICE"] = previousApplication["AMT_GOODS_PRICE"].fillna(previousApplication["AMT_GOODS_PRICE"].isnull().sum())
```

```
previousApplication["AMT_GOODS_PRICE"].isnull().sum()
```

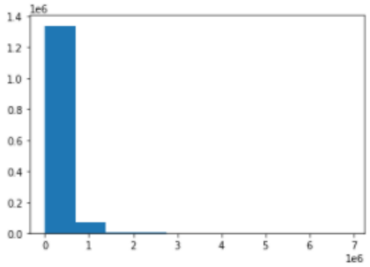
```
0
```

```
plt.hist(previousApplication["AMT_GOODS_PRICE"])
plt.show()
```

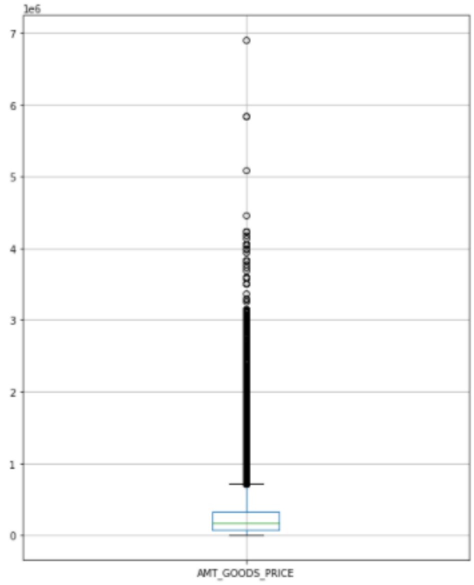


```
#after imputation

plt.hist(previousApplication["AMT_GOODS_PRICE"])
plt.show()
```




```
previousApplication.boxplot(["AMT_GOODS_PRICE"], figsize = [8, 10])
plt.show()
```



- COLUMN: [NAME_TYPE_SUITE]

```
previousApplication["NAME_TYPE_SUITE"].describe()
```

```
count           723810
unique               7
top       Unaccompanied
freq            432426
Name: NAME_TYPE_SUITE, dtype: object
```

```
previousApplication["NAME_TYPE_SUITE"].value_counts()
```

```
Unaccompanied      432426
Family             182139
Spouse, partner     57430
Children            27085
Other_B             15023
Other_A              7793
Group of people      1914
Name: NAME_TYPE_SUITE, dtype: int64
```

IE_TYPE_SUITE is a categorical column. Impute missing values with the value 'missing'.

```
previousApplication["NAME_TYPE_SUITE"] = previousApplication["NAME_TYPE_SUITE"].fillna("Missing")
```

```
previousApplication["NAME_TYPE_SUITE"].value_counts()
```

```
Missing            697401
Unaccompanied      432426
Family             182139
Spouse, partner     57430
Children            27085
Other_B             15023
Other_A              7793
Group of people      1914
Name: NAME_TYPE_SUITE, dtype: int64
```

- COLUMN: [CNT_PAYMENT]

```
previousApplication["CNT_PAYMENT"].describe() ##Term of previous credit at application of the previous application (WHAT DOES THIS EVEN MEAN)
```

```
count   1105619.00
mean         16.00
std          14.52
min           0.00
25%           6.00
50%          12.00
75%          24.00
max          84.00
Name: CNT_PAYMENT, dtype: float64
```

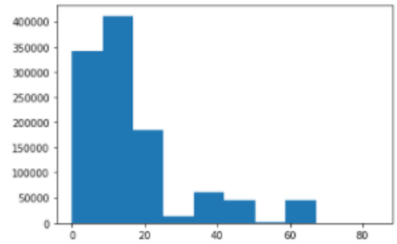
```
previousApplication["CNT_PAYMENT"]
```

```
0          12.00
1          36.00
2          12.00
3          12.00
4          24.00
            ...
1421206     6.00
1421207    24.00
1421208    12.00
1421209    12.00
1421210      NaN
Name: CNT_PAYMENT, Length: 1421211, dtype: float64
```

```
previousApplication["CNT_PAYMENT"].isnull().sum()
```

```
315592
```

```
plt.hist(previousApplication["CNT_PAYMENT"])
plt.show()
```

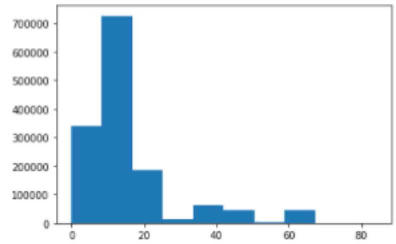



```
##Impute missing values with the mean
previousApplication["CNT_PAYMENT"].mean()
```

```
16.00480907075584
```

```
previousApplication["CNT_PAYMENT"] = previousApplication["CNT_PAYMENT"].fillna(previousApplication["CNT_PAYMENT"].mean())
```

```
#After imputation
plt.hist(previousApplication["CNT_PAYMENT"])
plt.show()
```




```
#Verifying
previousApplication["CNT_PAYMENT"].isnull().sum()
```

```
0
```

- Similarly perform the operation for COLUMN: [DAYS_FIRST_DRAWING], COLUMN: [DAYS_FIRST_DUE], COLUMN: [DAYS_LAST_DUE_1ST_VERSION], COLUMN: [DAYS_LAST_DUE], COLUMN: [DAYS_TERMINATION], NFLAG_INSURED_ON_APPROVAL, COLUMN: [Product_Combination], COLUMN: [AMT_CREDIT]

APPLICATION_DATA.CSV:

Information : Shows information of data including datatypes and whether there are null values

```
applicationData.info(verbose = True, null_counts = True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 122 columns):
 #    Column                        Non-Null Count   Dtype
---   ------                        --------------   -----
 0    SK_ID_CURR                    307511 non-null  int64
 1    TARGET                        307511 non-null  int64
 2    NAME_CONTRACT_TYPE            307511 non-null  object
 3    CODE_GENDER                   307511 non-null  object
 4    FLAG_OWN_CAR                  307511 non-null  object
 5    FLAG_OWN_REALTY               307511 non-null  object
 6    CNT_CHILDREN                  307511 non-null  int64
 7    AMT_INCOME_TOTAL              307511 non-null  float64
 8    AMT_CREDIT                    307511 non-null  float64
 9    AMT_ANNUITY                   307499 non-null  float64
 10   AMT_GOODS_PRICE               307233 non-null  float64
 11   NAME_TYPE_SUITE               306219 non-null  object
 12   NAME_INCOME_TYPE              307511 non-null  object
 13   NAME_EDUCATION_TYPE           307511 non-null  object
 14   NAME_FAMILY_STATUS            307511 non-null  object
 15   NAME_HOUSING_TYPE             307511 non-null  object
 16   REGION_POPULATION_RELATIVE    307511 non-null  float64
 17   DAYS_BIRTH                    307511 non-null  int64
 18   DAYS_EMPLOYED                 307511 non-null  int64
 19   DAYS_REGISTRATION             307511 non-null  float64
 20   DAYS_ID_PUBLISH               307511 non-null  int64
 21   OWN_CAR_AGE                   104582 non-null  float64
 22   FLAG_MOBIL                    307511 non-null  int64
 23   FLAG_EMP_PHONE                307511 non-null  int64
 24   FLAG_WORK_PHONE               307511 non-null  int64
 25   FLAG_CONT_MOBILE              307511 non-null  int64
 26   FLAG_PHONE                    307511 non-null  int64
 27   FLAG_EMAIL                    307511 non-null  int64
 28   OCCUPATION_TYPE               211120 non-null  object
 29   CNT_FAM_MEMBERS               307509 non-null  float64
 30   REGION_RATING_CLIENT          307511 non-null  int64
 31   REGION_RATING_CLIENT_W_CITY   307511 non-null  int64
 32   WEEKDAY_APPR_PROCESS_START    307511 non-null  object
 33   HOUR_APPR_PROCESS_START       307511 non-null  int64
 34   REG_REGION_NOT_LIVE_REGION    307511 non-null  int64
 35   REG_REGION_NOT_WORK_REGION    307511 non-null  int64
 36   LIVE_REGION_NOT_WORK_REGION   307511 non-null  int64
 37   REG_CITY_NOT_LIVE_CITY        307511 non-null  int64
 38   REG_CITY_NOT_WORK_CITY        307511 non-null  int64
```

Displaying all the columns with the percentage of missing values

```
pd.set_option("display.max_rows", None, "display.max_columns", None) #TO DISPLAY ALL COLUMNS
(applicationData.isnull().sum()/len(applicationData.index))*100
```

```
FLOORSMAX_MODE                 49.76
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_BEGINEXPLUATATION_MEDI   48.78
YEARS_BUILD_MEDI               66.50
COMMONAREA_MEDI                69.87
ELEVATORS_MEDI                 53.30
ENTRANCES_MEDI                 50.35
FLOORSMAX_MEDI                 49.76
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
TOTALAREA_MODE                 48.27
WALLSMATERIAL_MODE             50.84
EMERGENCYSTATE_MODE            47.40
OBS_30_CNT_SOCIAL_CIRCLE        0.33
DEF_30_CNT_SOCIAL_CIRCLE        0.33
OBS_60_CNT_SOCIAL_CIRCLE        0.33
DEF_60_CNT_SOCIAL_CIRCLE        0.33
DAYS_LAST_PHONE_CHANGE          0.00
FLAG_DOCUMENT_2                 0.00
FLAG_DOCUMENT_3                 0.00
FLAG_DOCUMENT_4                 0.00
FLAG_DOCUMENT_5                 0.00
FLAG_DOCUMENT_6                 0.00
FLAG_DOCUMENT_7                 0.00
FLAG_DOCUMENT_8                 0.00
FLAG_DOCUMENT_9                 0.00
FLAG_DOCUMENT_10                0.00
FLAG_DOCUMENT_11                0.00
FLAG_DOCUMENT_12                0.00
FLAG_DOCUMENT_13                0.00
FLAG_DOCUMENT_14                0.00
FLAG_DOCUMENT_15                0.00
FLAG_DOCUMENT_16                0.00
FLAG_DOCUMENT_17                0.00
FLAG_DOCUMENT_18                0.00
FLAG_DOCUMENT_19                0.00
FLAG_DOCUMENT_20                0.00
FLAG_DOCUMENT_21                0.00
```

DATA CLEANING OF APPLICATION_DATA.CSV :

Remove columns with > 50% missing data

```
columnsToDelete = ["OWN_CAR_AGE", "EXT_SOURCE_1", "APARTMENTS_AVG", "BASEMENTAREA_AVG", "YEARS_BUILD_AVG", "COMMONAREA_AVG", "ELEVATORS_AVG", "ENTRANCES_AVG", "FLOORSMIN_AVG", "LANDAREA_AVG", "LIVINGAPARTMENTS_AVG", "LIVINGAREA_AVG", "NONLIVINGAPARTMENTS_AVG", "NONLIVINGAR
```

```
applicationData.drop(columnsToDelete, axis = 1, inplace = True)
```

```
applicationData.shape
```

```
(307511, 81)
```

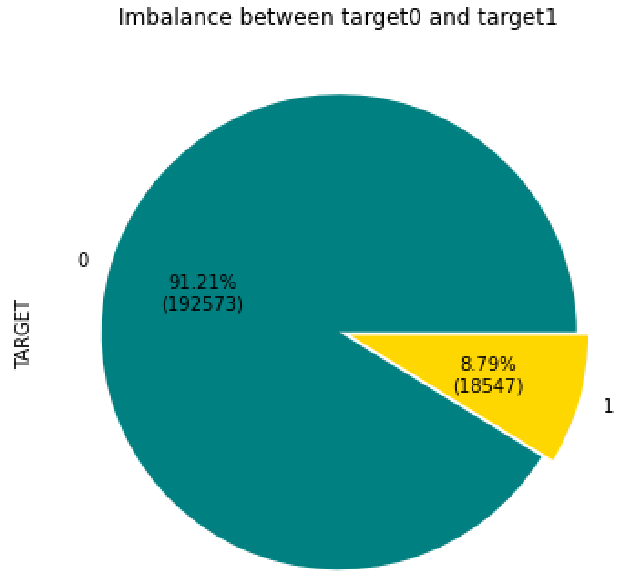
Verifying that all columns with >50% missing values are removed

```
#VERIFYING THAT ALL COLUMNS WITH >50% MISSING VALUES ARE REMOVED
(applicationData.isnull().sum()/len(applicationData.index))*100
```

```
FLAG_CONT_MOBILE               0.00
FLAG_PHONE                     0.00
FLAG_EMAIL                     0.00
OCCUPATION_TYPE               31.35
CNT_FAM_MEMBERS                0.00
REGION_RATING_CLIENT           0.00
REGION_RATING_CLIENT_W_CITY    0.00
WEEKDAY_APPR_PROCESS_START     0.00
HOUR_APPR_PROCESS_START        0.00
REG_REGION_NOT_LIVE_REGION     0.00
REG_REGION_NOT_WORK_REGION     0.00
LIVE_REGION_NOT_WORK_REGION    0.00
REG_CITY_NOT_LIVE_CITY         0.00
REG_CITY_NOT_WORK_CITY         0.00
LIVE_CITY_NOT_WORK_CITY        0.00
ORGANIZATION_TYPE              0.00
EXT_SOURCE_2                   0.21
EXT_SOURCE_3                  19.83
YEARS_BEGINEXPLUATATION_AVG   48.78
FLOORSMAX_AVG                 49.76
YEARS_BEGINEXPLUATATION_MODE  48.78
FLOORSMAX_MODE                49.76
YEARS_BEGINEXPLUATATION_MEDI  48.78
FLOORSMAX_MEDI                49.76
TOTALAREA_MODE                48.27
EMERGENCYSTATE_MODE           47.40
OBS_30_CNT_SOCIAL_CIRCLE       0.33
DEF_30_CNT_SOCIAL_CIRCLE       0.33
OBS_60_CNT_SOCIAL_CIRCLE       0.33
DEF_60_CNT_SOCIAL_CIRCLE       0.33
DAYS_LAST_PHONE_CHANGE         0.00
FLAG_DOCUMENT_2                0.00
FLAG_DOCUMENT_3                0.00
FLAG_DOCUMENT_4                0.00
FLAG_DOCUMENT_5                0.00
FLAG_DOCUMENT_6                0.00
FLAG_DOCUMENT_7                0.00
FLAG_DOCUMENT_8                0.00
FLAG_DOCUMENT_9                0.00
FLAG_DOCUMENT_10               0.00
FLAG_DOCUMENT_11               0.00
FLAG_DOCUMENT_12               0.00
FLAG_DOCUMENT_13               0.00
FLAG_DOCUMENT_14               0.00
FLAG_DOCUMENT_15               0.00
FLAG_DOCUMENT_16               0.00
FLAG_DOCUMENT_17               0.00
FLAG_DOCUMENT_18               0.00
FLAG_DOCUMENT_19               0.00
FLAG_DOCUMENT_20               0.00
FLAG_DOCUMENT_21               0.00
AMT_REQ_CREDIT_BUREAU_HOUR    13.50
AMT_REQ_CREDIT_BUREAU_DAY     13.50
AMT_REQ_CREDIT_BUREAU_WEEK    13.50
```

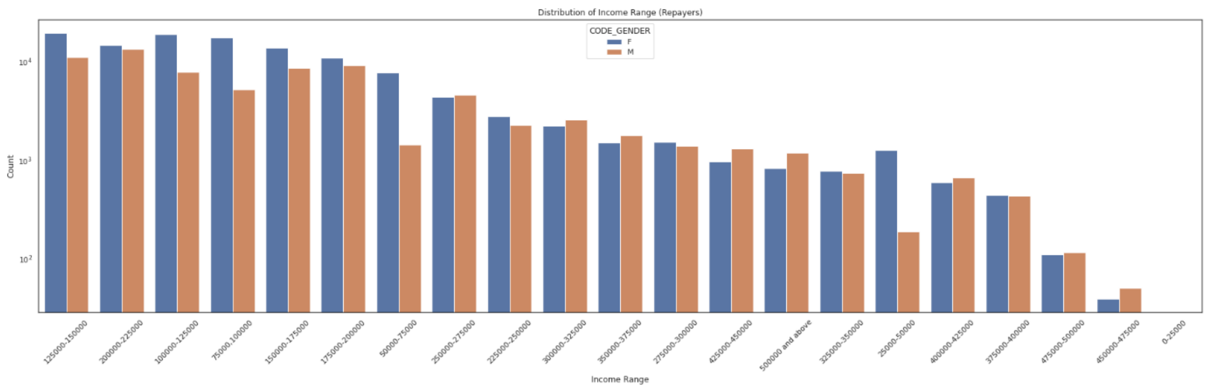
We carry out the same steps as incase of previous_application.csv for cleaning of application_data.csv

DATASET ANALYSIS :



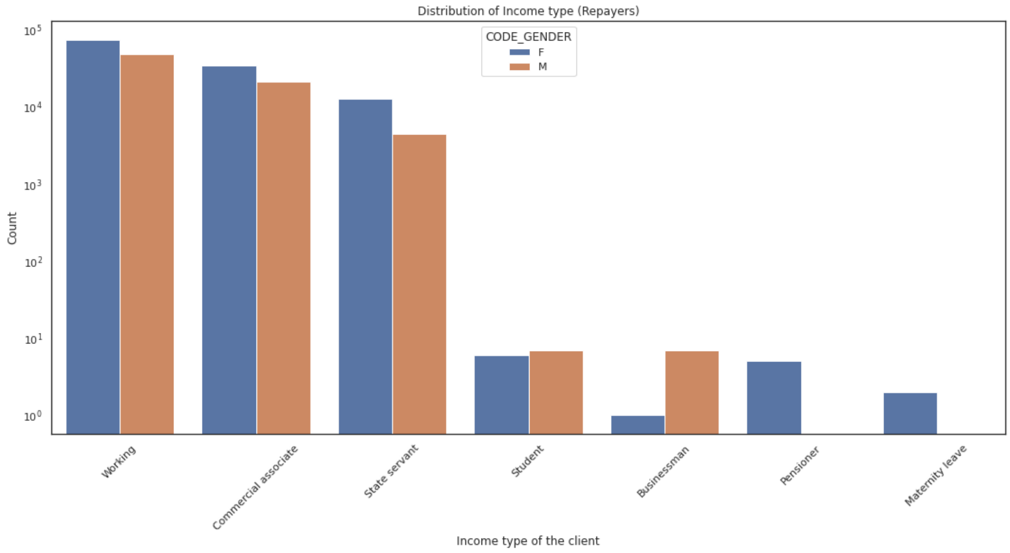
Inference:

8.78% of clients are clients with payment difficulties. 91.21% of clients fall under the 'all other cases' category.



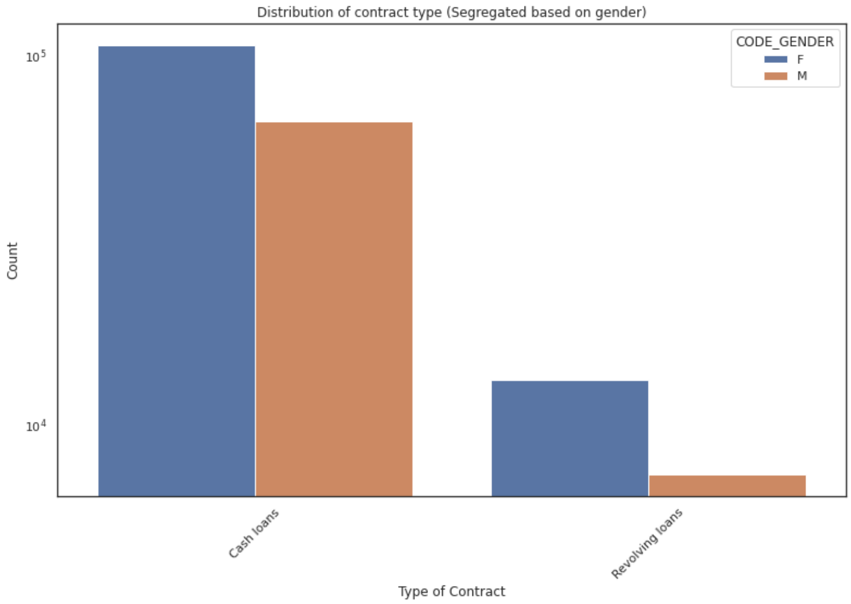
Inferences:

1. In majority of the cases, female counts are higher than male.
2. Income range from 100000 to 200000 is having more number of credits.
3. In the slots 250000-275000 and 375000-400000, the count for both males and females are almost the same



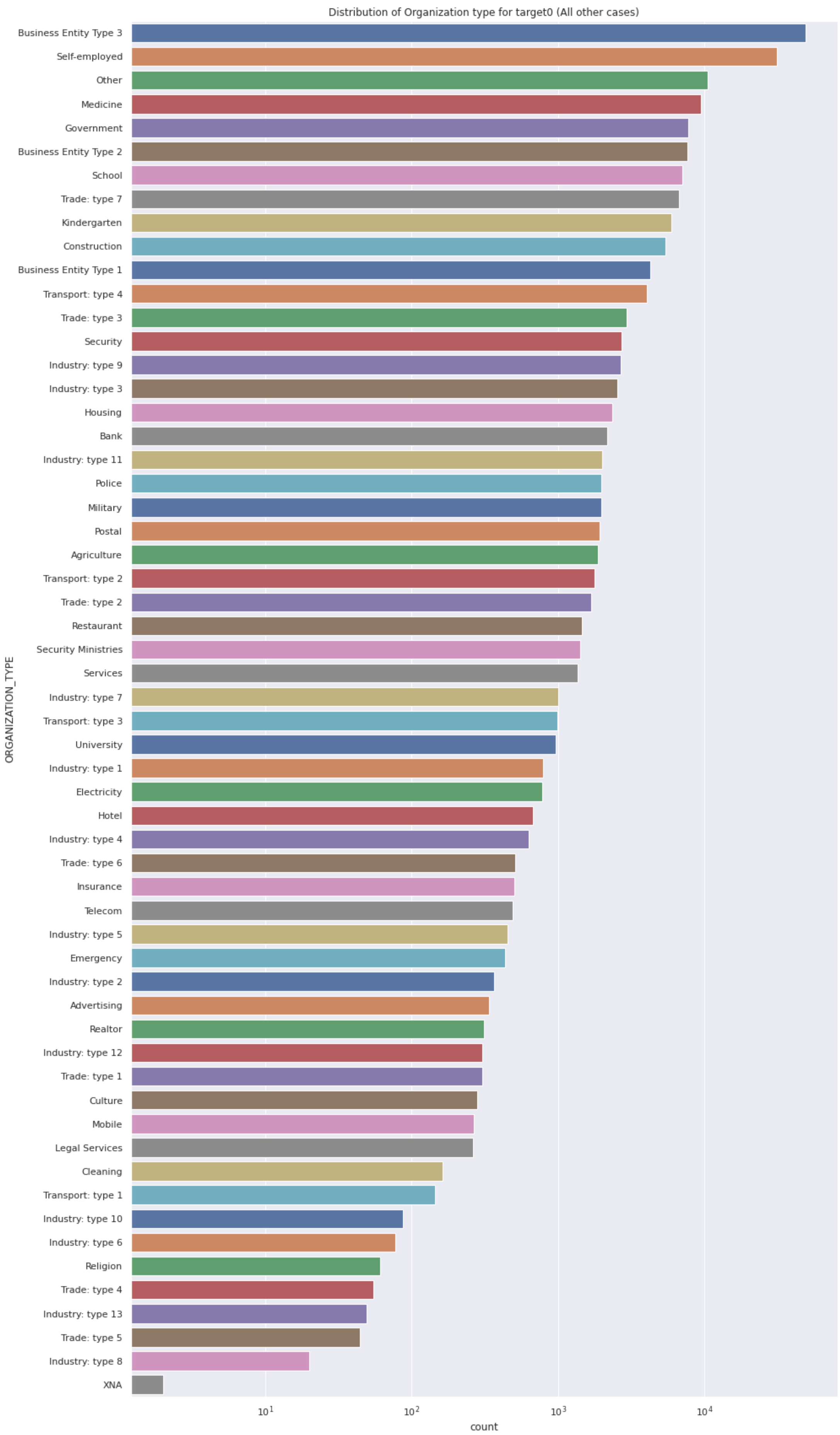
Inference:

1. Working people make up most of the clients.
2. Gender distribution amongst students is almost the same.
3. There is a stark decerease in the amount of clients if they are not working, commerical associate or state servants.
4. There are more female clients in the top 3 categories of income type.



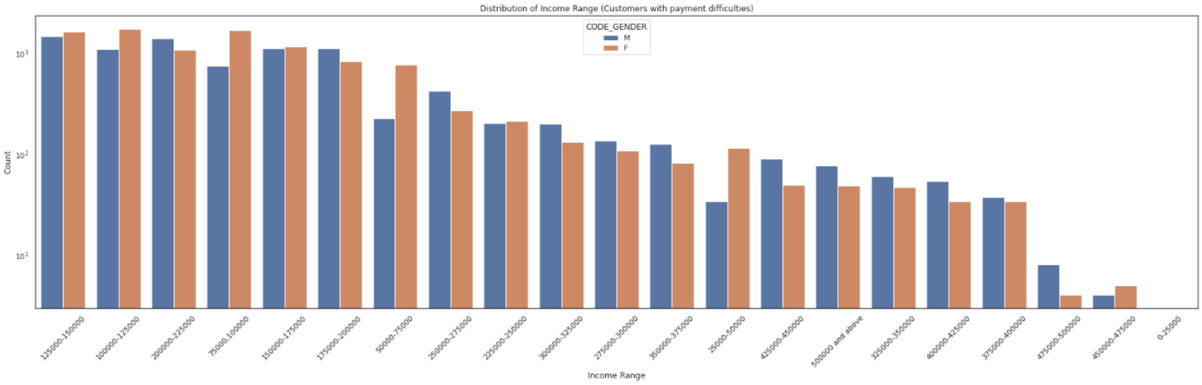
Inference:

Cash loans clearly has more clients per credit compared to revolving loans. In both cases, there are female clients than male clients.


Distribution of Organization type for target0 (All other cases)
ORGANIZATION_TYPE
Business Entity Type 3
Self-employed
Other
Medicine
Government
Business Entity Type 2
School
Trade: type 7
Kindergarten
Construction
Business Entity Type 1
Transport: type 4
Trade: type 3
Security
Industry: type 9
Industry: type 3
Housing
Bank
Industry: type 11
Police
Military
Postal
Agriculture
Transport: type 2
Trade: type 2
Restaurant
Security Ministries
Services
Industry: type 7
Transport: type 3
University
Industry: type 1
Electricity
Hotel
Industry: type 4
Trade: type 6
Insurance
Telecom
Industry: type 5
Emergency
Industry: type 2
Advertising
Realtor
Industry: type 12
Trade: type 1
Culture
Mobile
Legal Services
Cleaning
Transport: type 1
Industry: type 10
Industry: type 6
Religion
Trade: type 4
Industry: type 13
Trade: type 5
Industry: type 8
XNA
10^1
10^2
10^3
10^4
count

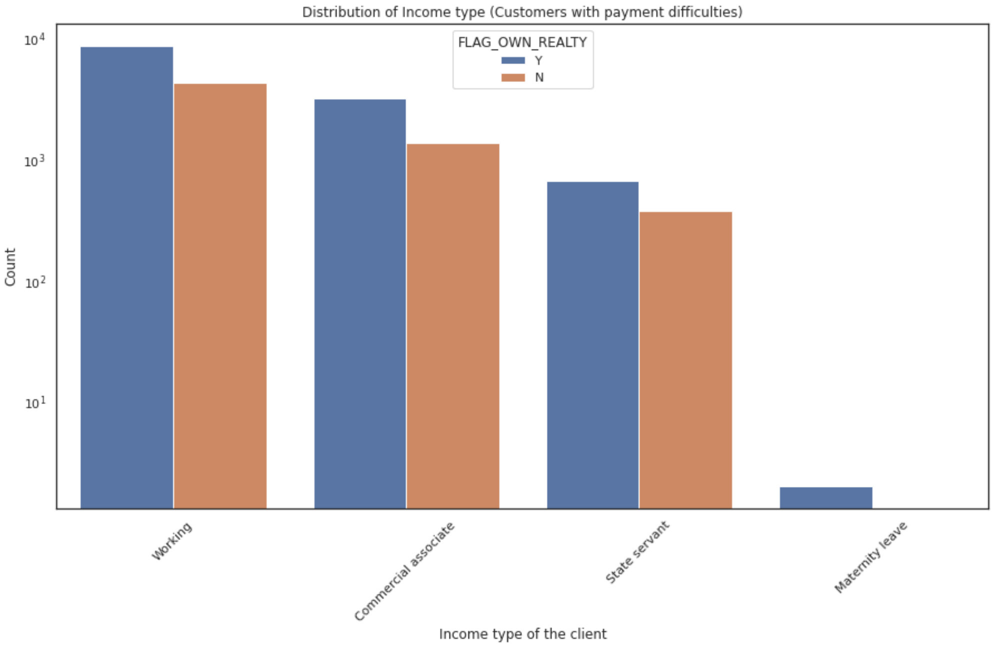
Inference:

1. Clients which have applied for credits are from most of the organization type ‘Business entity Type 3’ , ‘Self employed’, ‘Other’ , ‘Medicine’, ‘Government’ and 'Business entity type2'.
2. Fewer clients are from Industries type 8, type 5, Industry: type 13, Trade: type4, Religion, Industry type 6 and type 10.



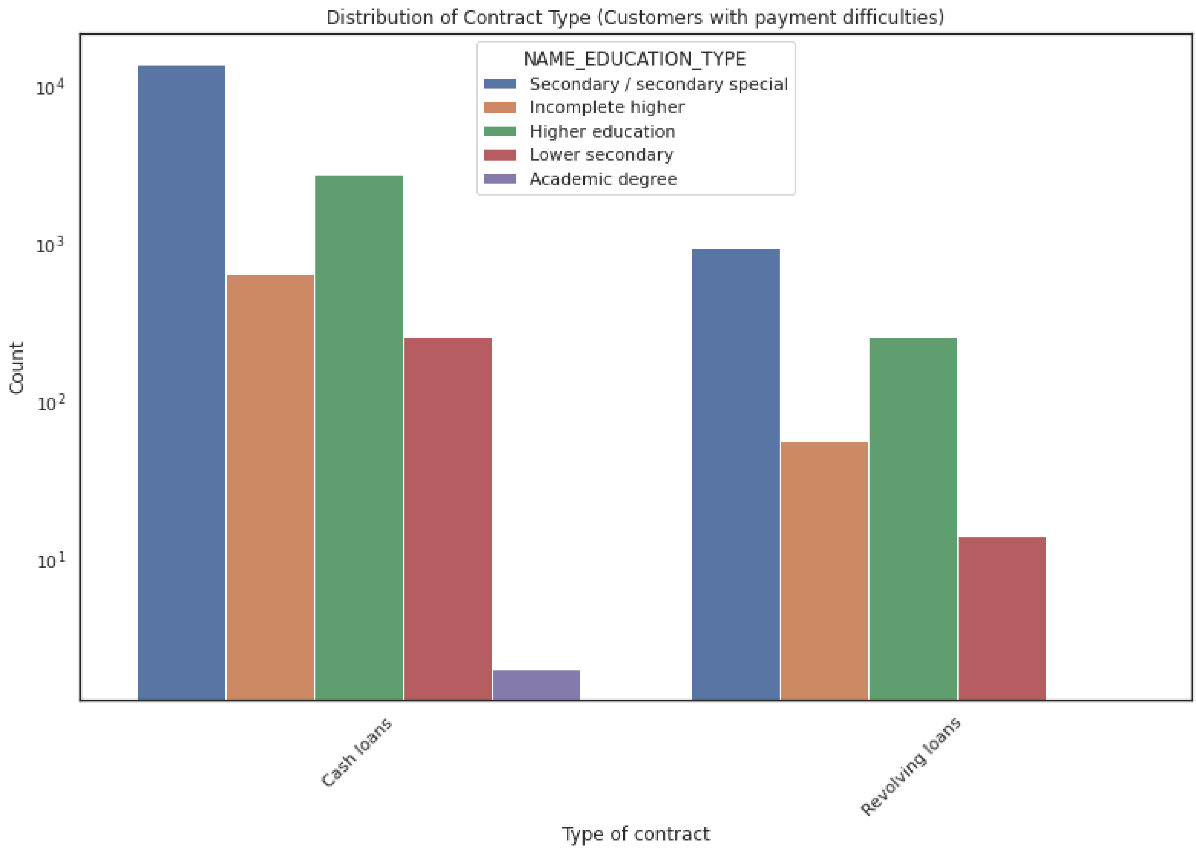
Inference:

1. Income range from 100000 to 200000 is having the highest number of credits.
2. Very less count for income range 400000 and above.
3. On average, there are more number of male clients where the number of credits are less.



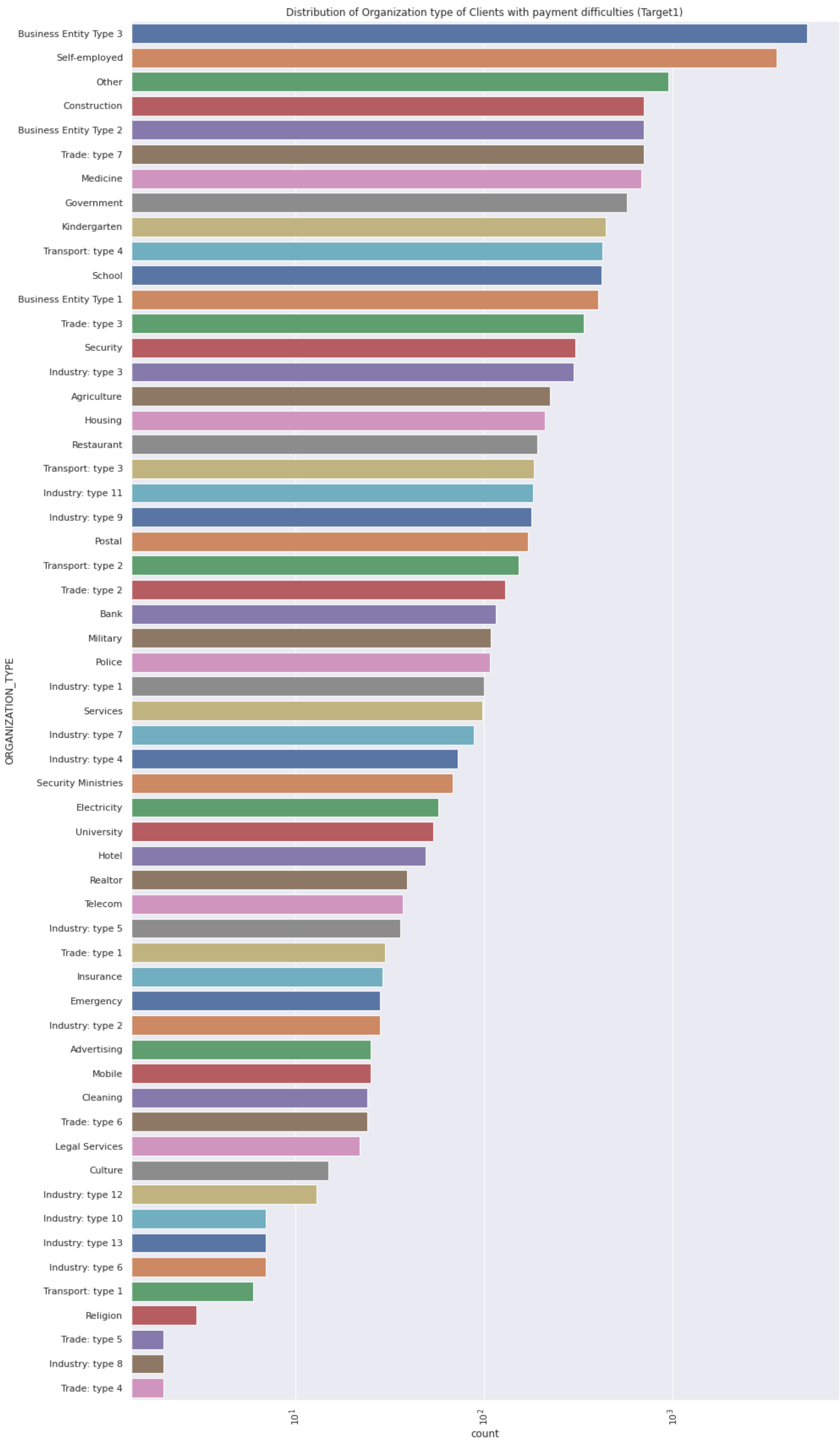
Inferences:

1. Working customers, obviously, have a higher count.
2. As we can see, most customers do have their own property (house or a flat) but a large number of customers can be stated as otherwise.



Inferences:

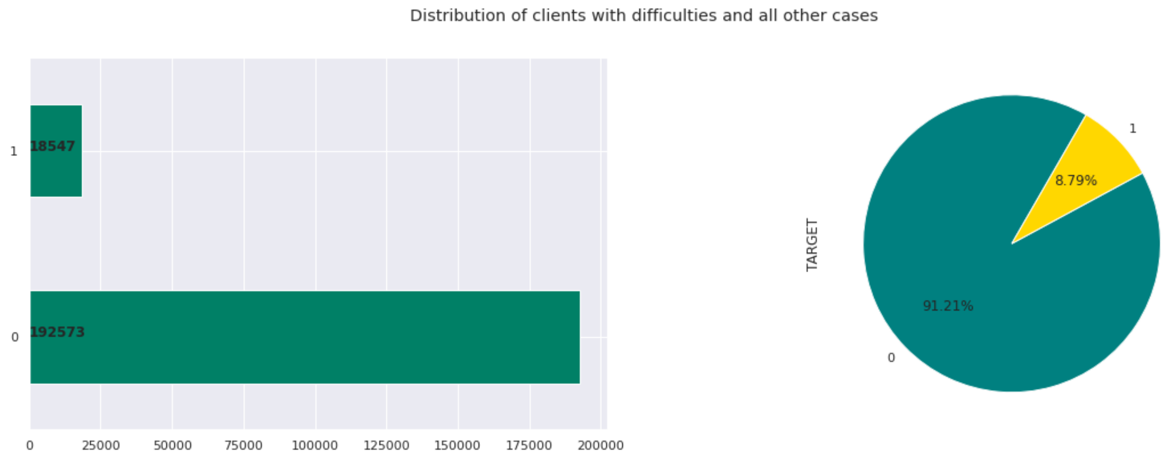
1. Cash loans, as we can see, are preferred by clients of all education backgrounds with an overwhelming majority.
2. People with only an academic degree do not prefer revolving loans at all.



Distribution of Organization type of Clients with payment difficulties (Target1)
Business Entity Type 3
Self-employed
Other
Construction
Business Entity Type 2
Trade: type 7
Medicine
Government
Kindergarten
Transport: type 4
School
Business Entity Type 1
Trade: type 3
Security
Industry: type 3
Agriculture
Housing
Restaurant
Transport: type 3
Industry: type 11
Industry: type 9
Postal
Transport: type 2
Trade: type 2
Bank
Military
Police
Industry: type 1
Services
Industry: type 7
Industry: type 4
Security Ministries
Electricity
University
Hotel
Realtor
Telecom
Industry: type 5
Trade: type 1
Insurance
Emergency
Industry: type 2
Advertising
Mobile
Cleaning
Trade: type 6
Legal Services
Culture
Industry: type 12
Industry: type 10
Industry: type 13
Industry: type 6
Transport: type 1
Religion
Trade: type 5
Industry: type 8
Trade: type 4
ORGANIZATION_TYPE
10^1
10^2
10^3
count

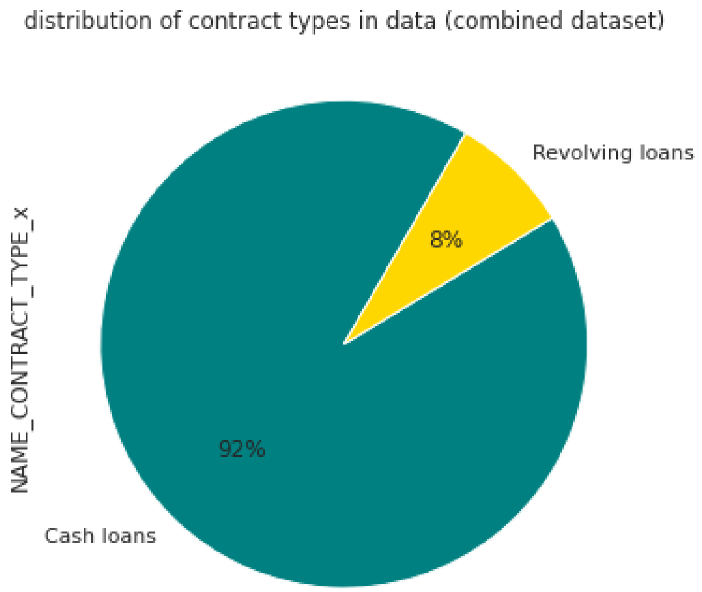
Inferences:

1. As compared to the clients with NO payment difficulties, clients WITH payment difficulties have the 'construction' business type in the top 5 count replacing the 'medicine' business type.
2. Most of the business types are the same as clients with NO payment difficulties, except we have the business type 'Transport: type1' in the case of clients WITH payment difficulties which wasn't present before.



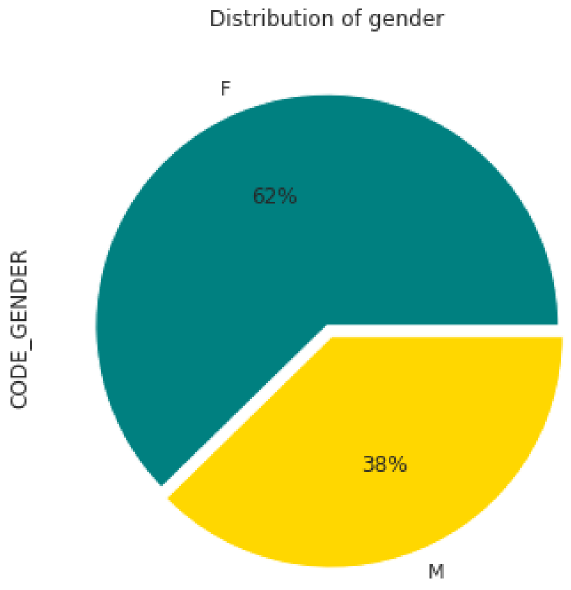
Inference:

8.79% (18547) out of total client population (192573) have difficulties in repaying loans.



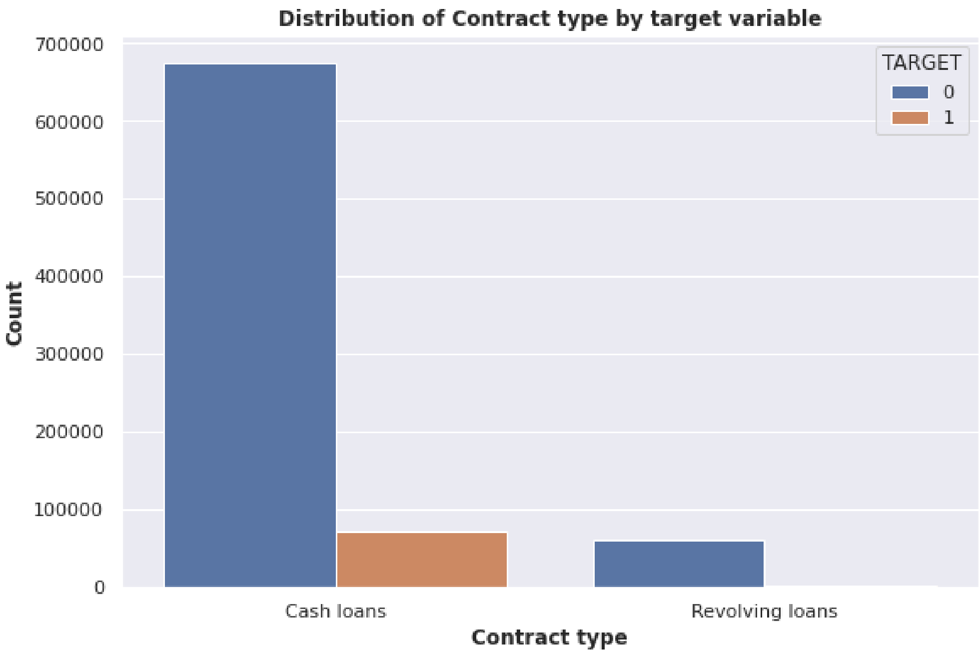
Inference:

The percentage of revolving loans and cash loans are 8% & 92%.



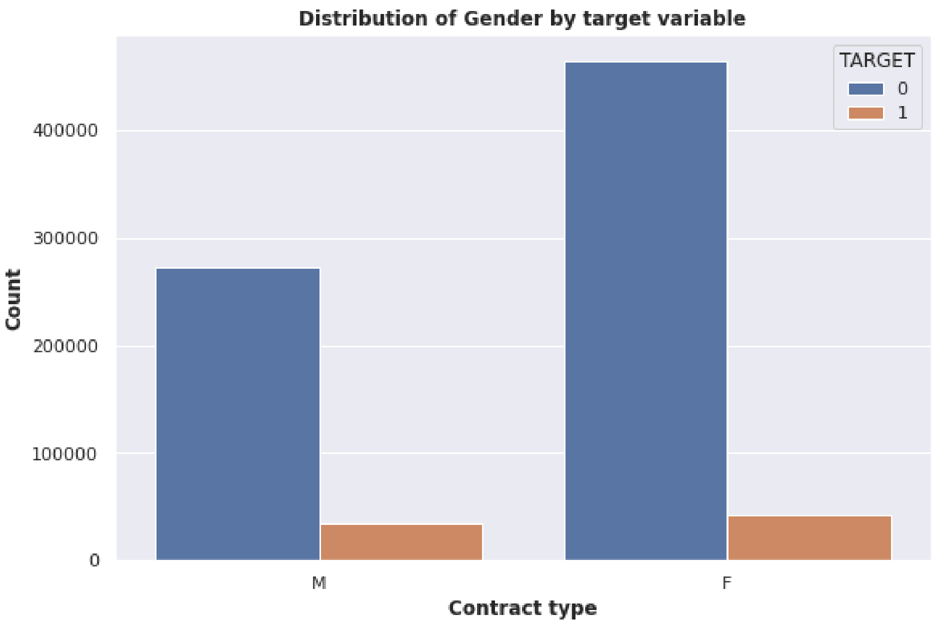
Inference:

In the applicationData file, we saw females had 61% and males had 39% but now in the combined dataset we see:- Females: 62% Males: 38%



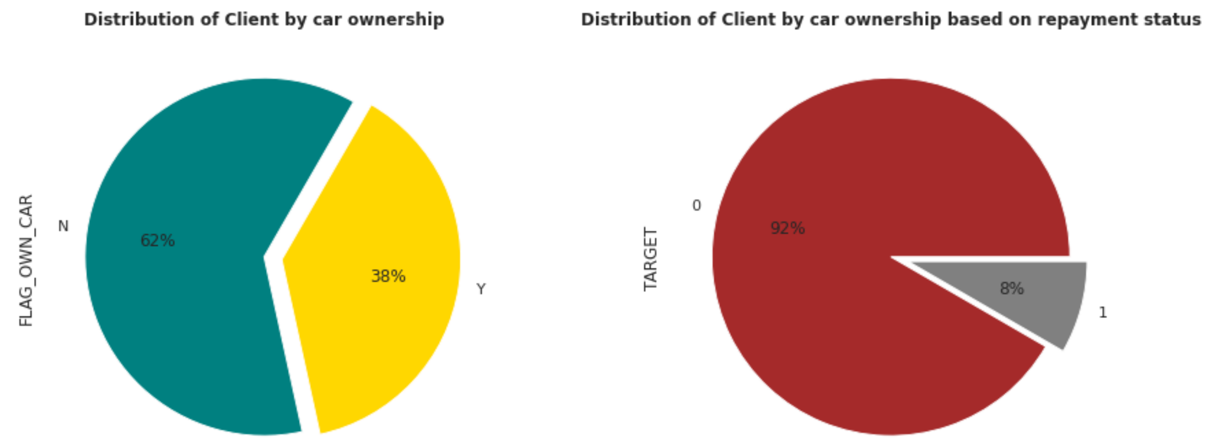
Inference:

Both set of clients (Target 0 and target 1) prefer cash loans over revolving loans with overwhelming numbers



Inference:

1. Clearly, female clients are the best repayers of their loan (almost double the amount of males).
2. Amount of defaulters in both genders are almost equally distributed.

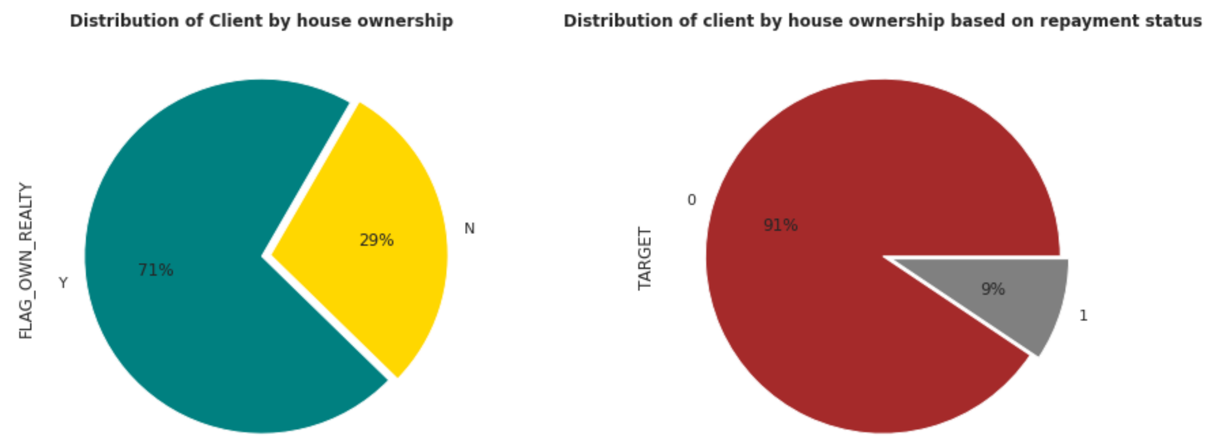


Inference:

1st pie plot : Only 38% of clients own a car .

2nd pie plot : Only

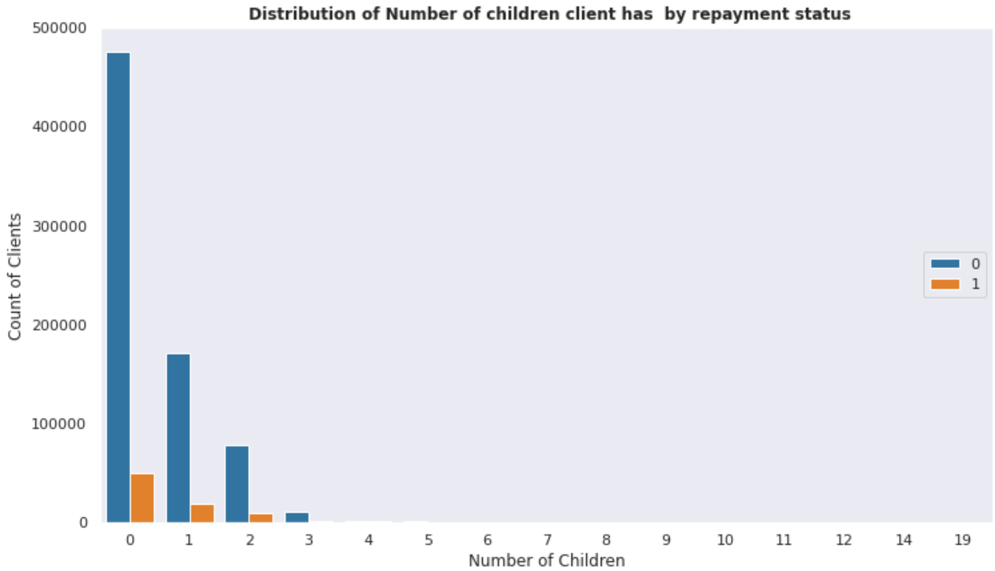
8% of clients who own a car have difficulty in payments

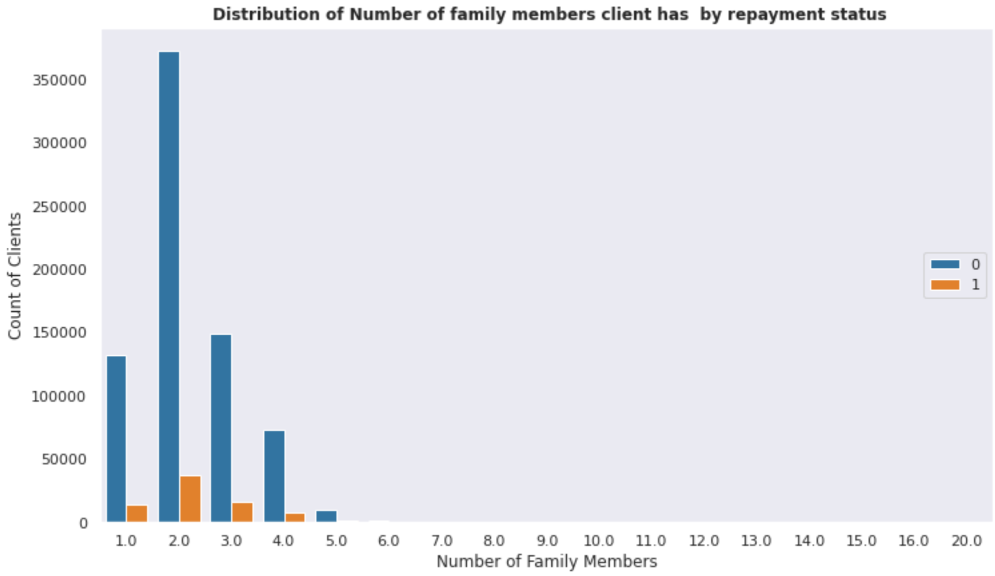


Inference:

SUBPLOT 1 : 71% of clients own a house or a flat.

SUBPLOT 2 : Out of all the clients who own a house, 9% of clients have difficulty in making payments.





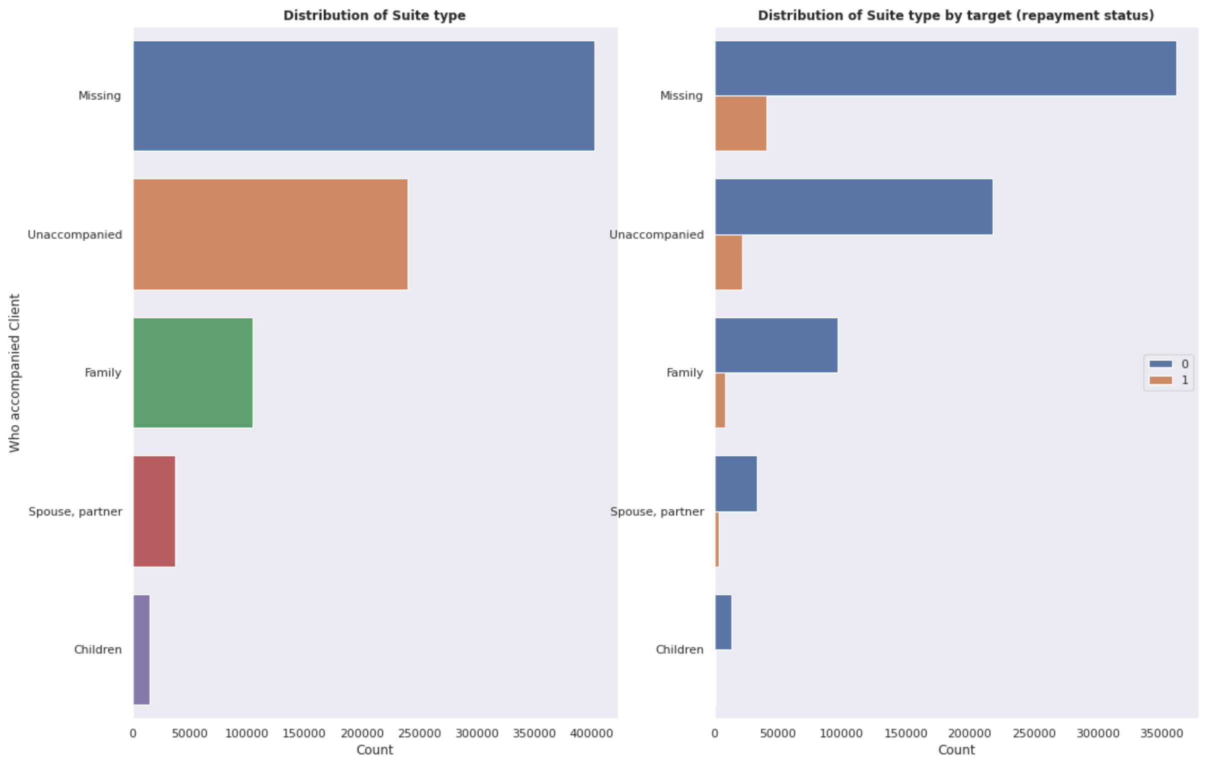
Inference:

Subplot1:

1. The majority as per both cases of repayment status, have zero children.
2. Clients with more than 2 children do not have difficulty in making payments.
3. Clients with 0 children have the majority in terms of having difficulty in making payments.

Subplot2:

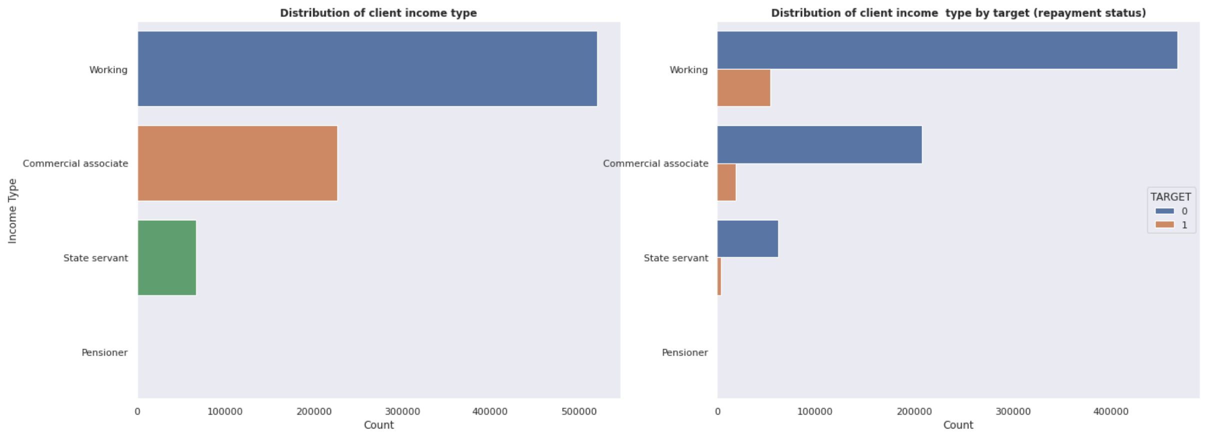
1. Clients with 2 family members living together are in high numbers as per both cases of repayment status
2. Also, from point 1, the majority of clients having difficulty in payments have 2 family members



Inferences:

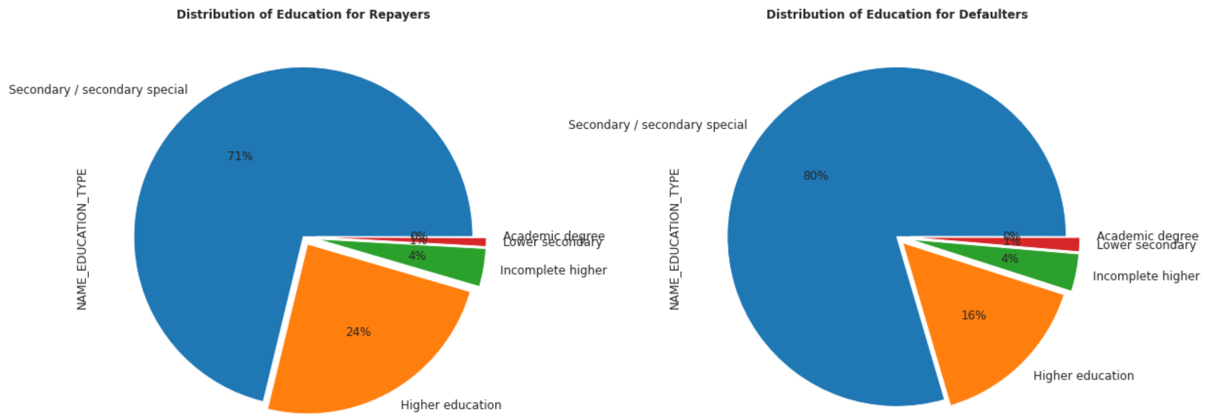
Note: Missing data was labelled as 'missing' during the data cleaning process so we can ignore it.

1. Majority of the clients are (in both cases of repayment status) unaccompanied (without anyone to help/guide them)
2. Least amount of clients are in the company of their children.



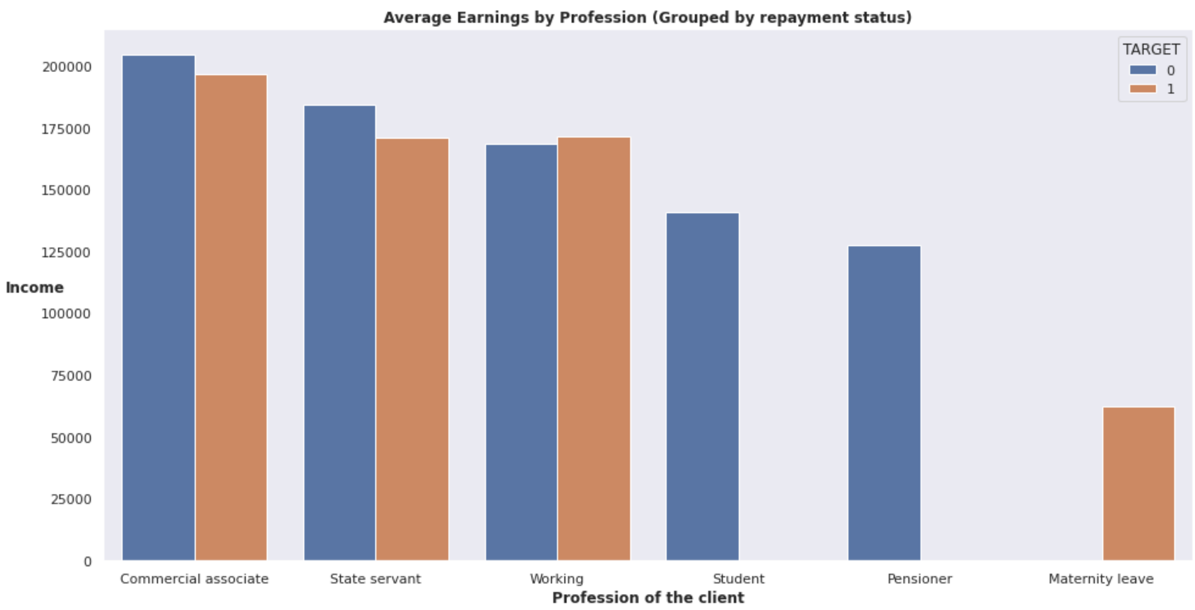
Inference:

1. Most clients as per both cases of repayment status, are working.
2. Conversely, the least amount of clients are pensioners (retired clients)



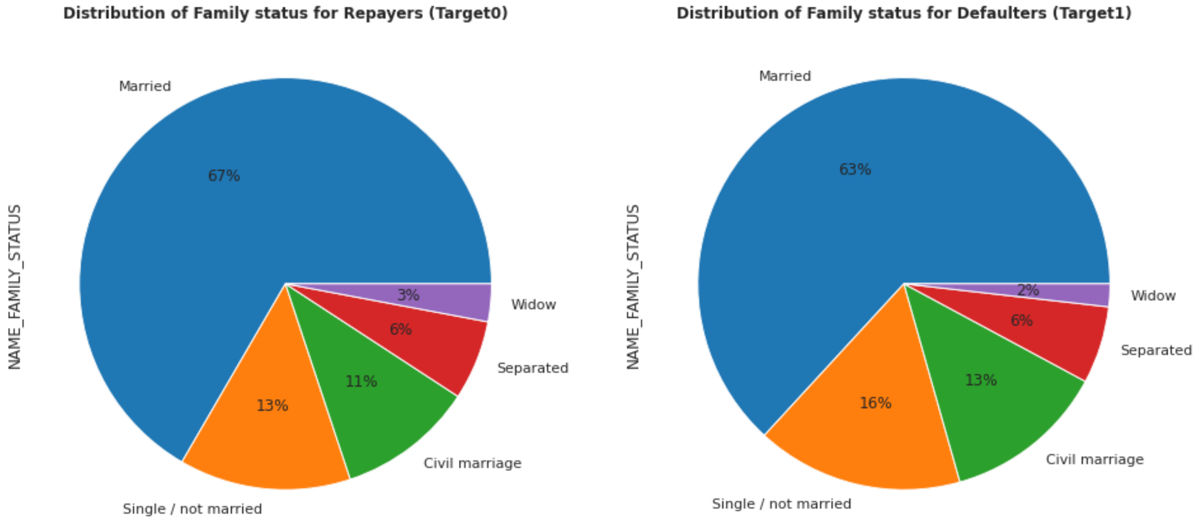
Inference:

1. Clients who default are proportionally 9% higher compared to clients who do not default (for clients with education as secondary).
2. In the higher education category, clients who default are 8% fewer.
3. In both cases of repayment status, lower secondary and academic degree categories are the minority.



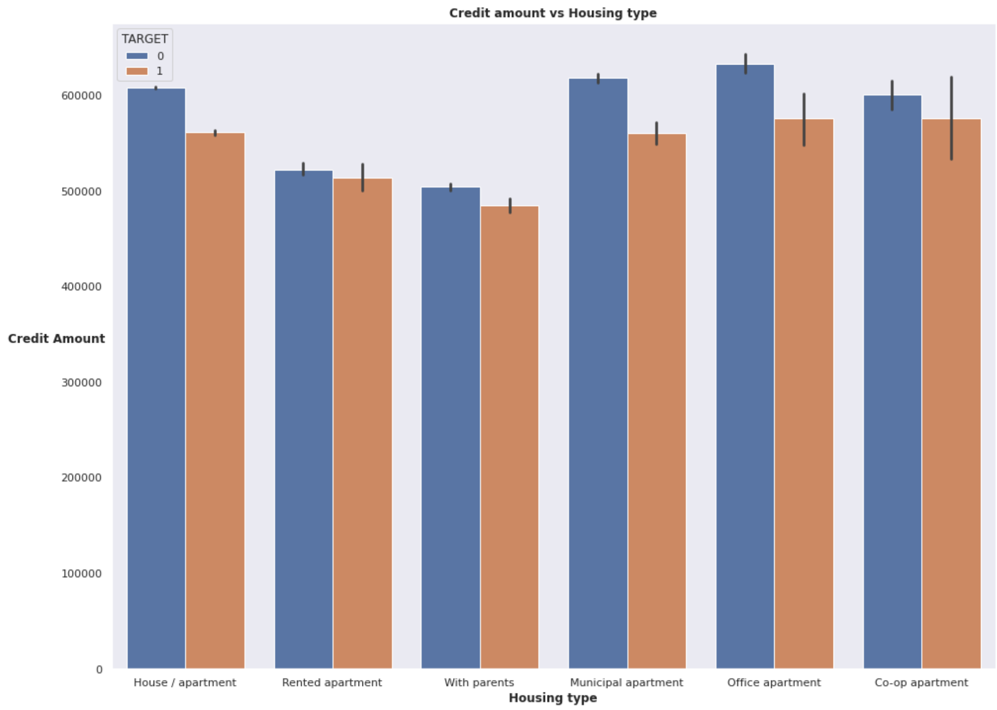
Inference:

1. In both cases of repayment status, commerical associate clients are the highest earners.
2. Clients who are on maternity leave (therefore, female clients) have difficulty in making payments
3. Pensioners and students do not have any difficulties in repayments.
4. There are almost an equal number of clients under the working category who repay and default.



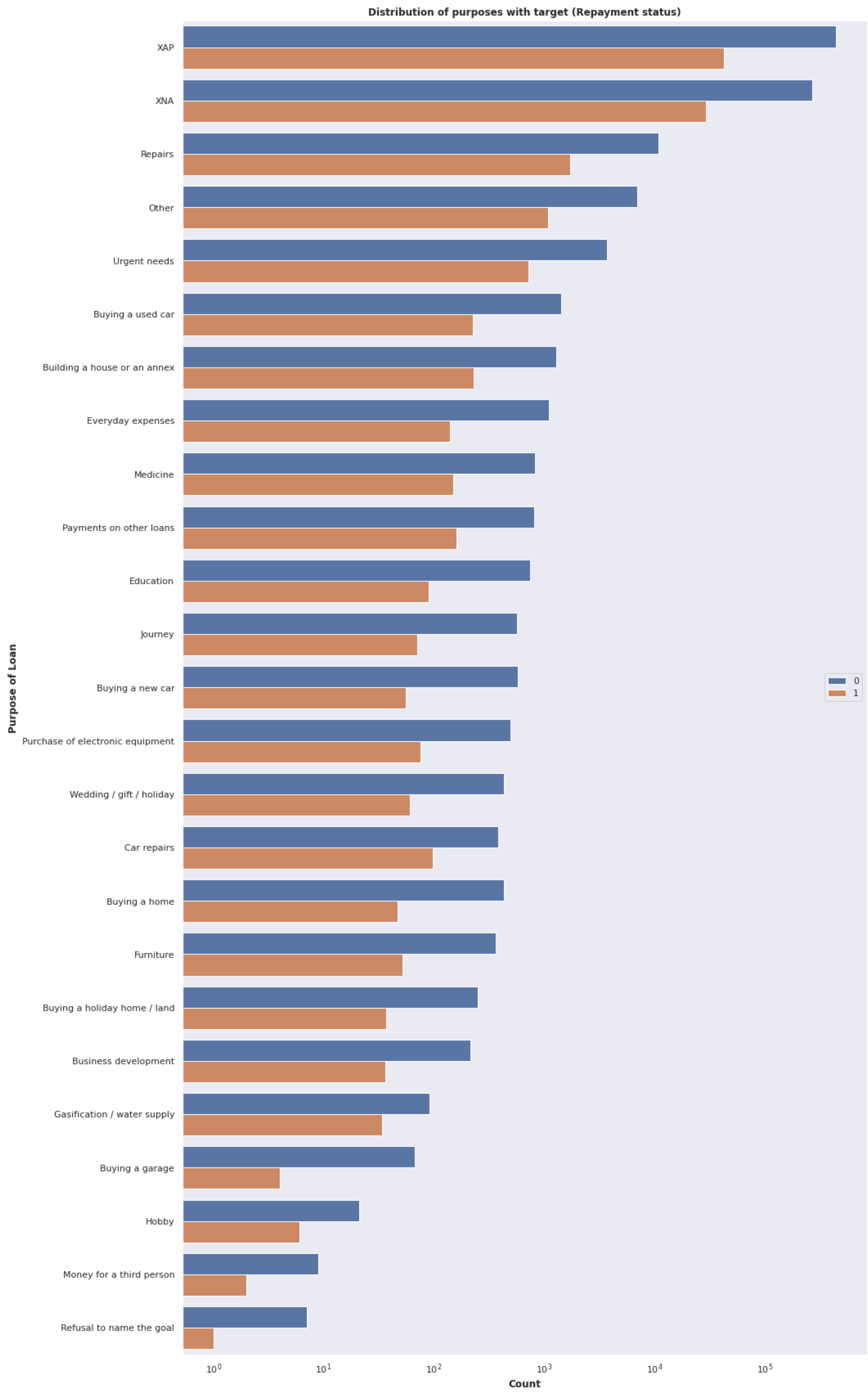
Inference:

1. There's a difference of -4% in married clients who have difficulty in making payments.
2. Family status for both cases of repayment status have an almost evenly distributed family status (family members living with the client)



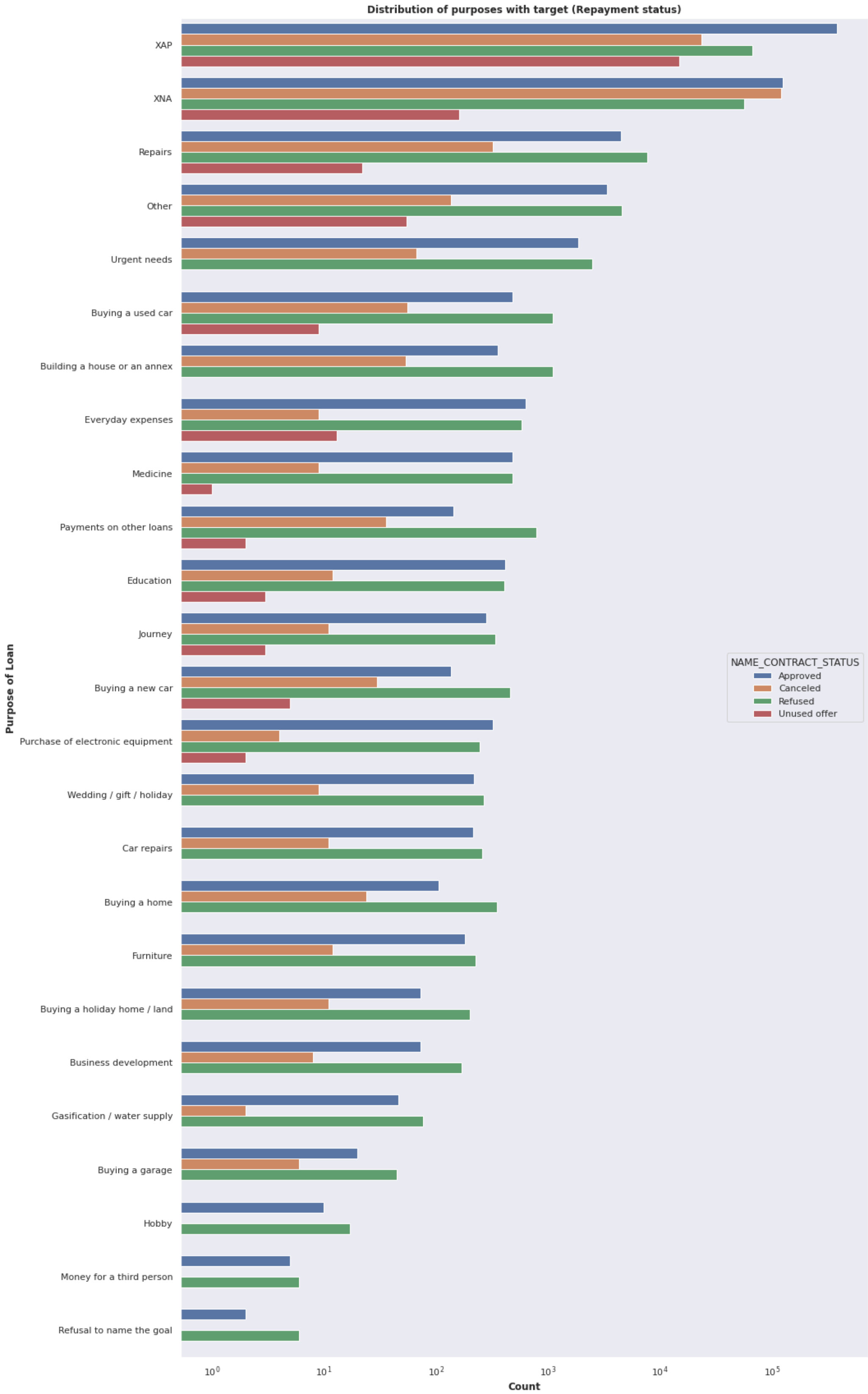
Inference:

1. Clients with office, co-op, municipal aparments have the highest repayers.
2. Clients living with parents or in a parents' aparment have the least amount of repayers and defaulters.



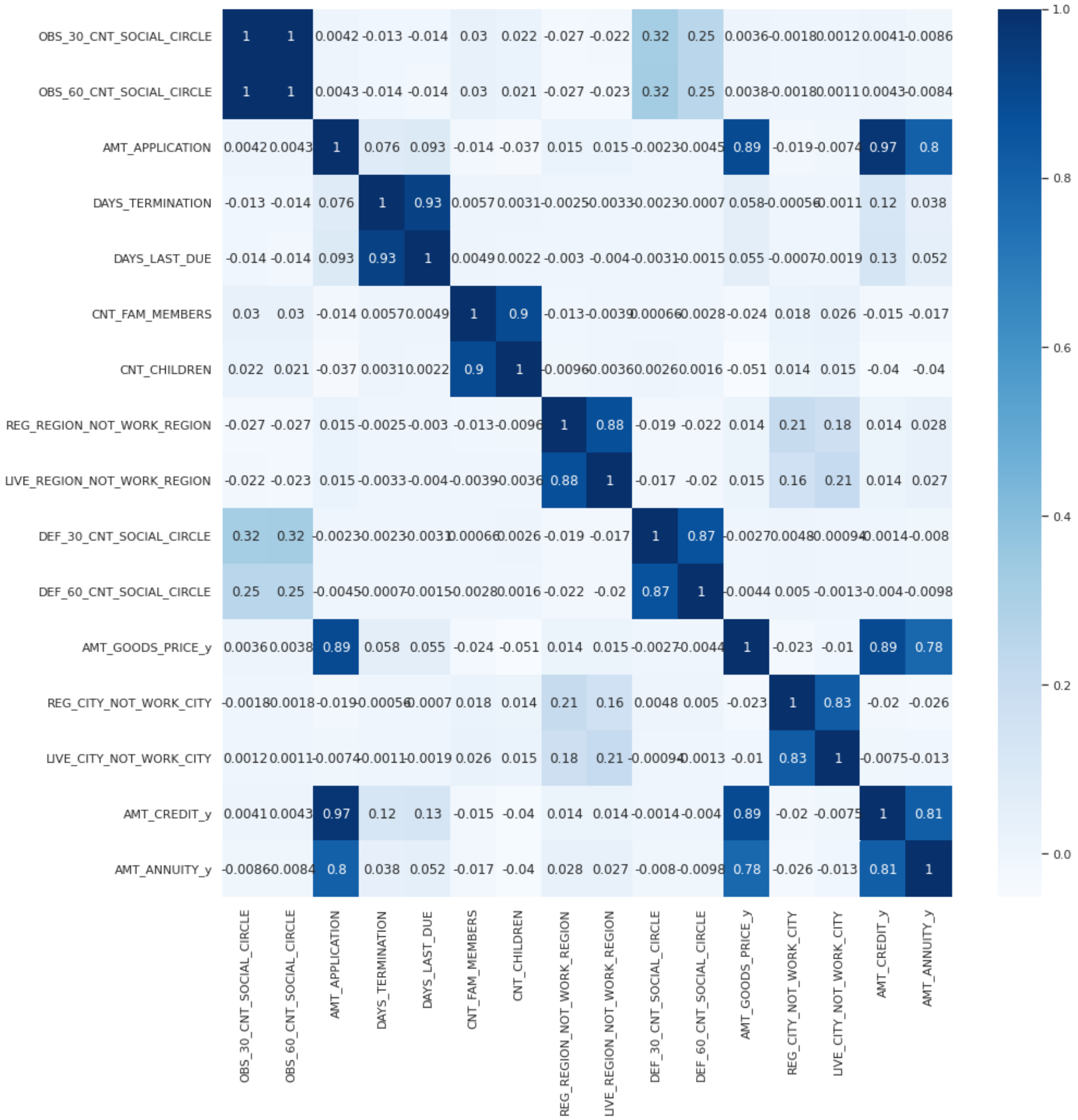
Inferences:

1. Repair purposes are on top with most defaulters and repayers.
2. Proportion wise, there are high amount of repayers when the client refuses to name the purpose of the loan. Although such clients are rare.



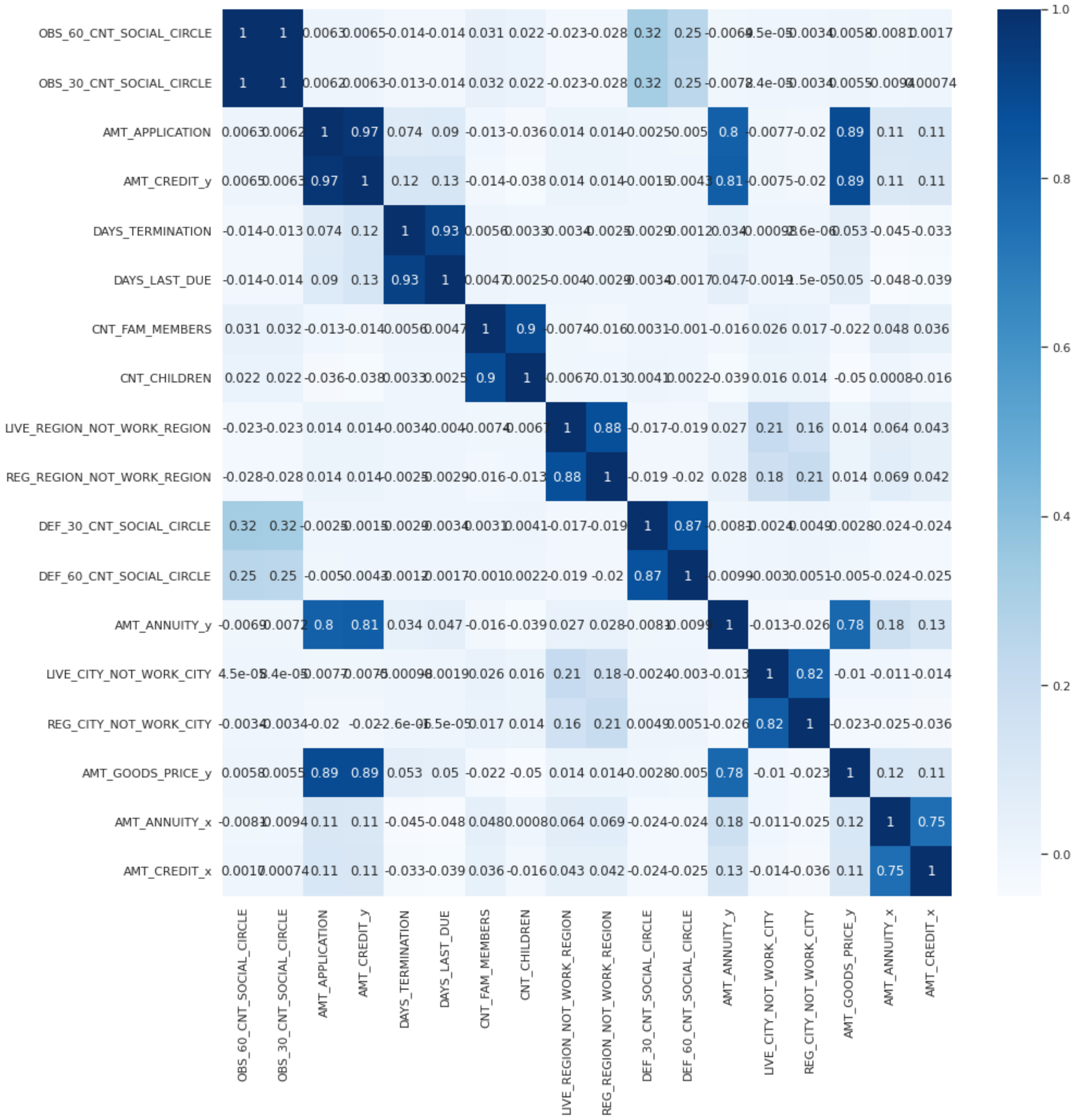
Inferences:

1. Most rejection of loans is when the purpose of the client is based on Repairs.
2. For education purposes we have equal number of approvals and refusals.



Inferences:

1. AMT_GOODS_PRICE and AMT_APPLICATION have a high correlation, which means the more credit the client asked for previously is proportional to the goods price that the client asked for previously.
2. AMT_ANNUITY and AMT_APPLICATION also have a high correlation, which means the higher the loan annuity issued, the higher the goods price that the client asked for previously.
3. If the client's contact address does not match the work address, then there's a high chance that the client's permanent address also does not match the work address.
4. First due of the previous application is highly correlated with Relative to the expected termination of the previous application
5. CNT_CHILDREN and CNT_FAM_MEMBERS are highly correlated which means a client with children is higly likely to have family members as well.



HEAT MAP:

- In comparison to the repayer heatmap, AMT_GOODS_PRICE and AMT_APPLICATION have a high correlation here as well, which means the more credit the client asked for previously is proportional to the goods price that the client asked for previously.
- In comparison to the repayer heatmap, AMT_ANNUITY and AMT_APPLICATION also have a high correlation, which means the higher the loan annuity issued, the higher the goods price that the client asked for previously.
- In comparison to the repayer heatmap, If the client's contact address does not match the work address, then there's a high chance that the client's permanent address also does not match the work address.
- Higher the goods price, higher the credit by the client

- First due of the previous application is highly correlated with Relative to the expected termination of the previous application (same as with the repayer heatmap)
- CNT_CHILDREN and CNT_FAM_MEMBERS are highly correlated which means a client with children is higly likely to have family members as well (same as with the repayer heatmap)

CONCLUSION:

1. Clients who are Students, Pensioners and Commercial Associates with a housing type such as office/co-op/municipal apartments NEED TO BE TARGETED by the bank for successful repayments. These clients have the highest amount of repayment history.
2. Female clients on maternity leave should NOT be targeted as they have no record of repayments (therefore they are highly likely to default and targeting them would lead to a loss)
3. While clients living with parents have the least amount of repayers, they also have the least amount of defaulters. So, in cases where the risk is less, such clients can be TARGETED.
4. Clients who are working need to be targeted LESS by the bank as they have the highest amount of defaulters.
5. Clients should NOT be targeted based on their education type alone as the data is very inconclusive.
6. Banks SHOULD target clients who own a car.
7. There are NO repayers/negligible repayers when the contract type is of revolving loan.
8. Banks SHOULD target more people with no children.
9. 'Repairs' purpose of loan is the one with the most defaulters and repayers. Therefore, clients with very low risk SHOULD be given loans for such purpose to yield high profits.
10. Banks SHOULD also target female clients as they are the highest repayers (almost as double as males) amongst both the genders.