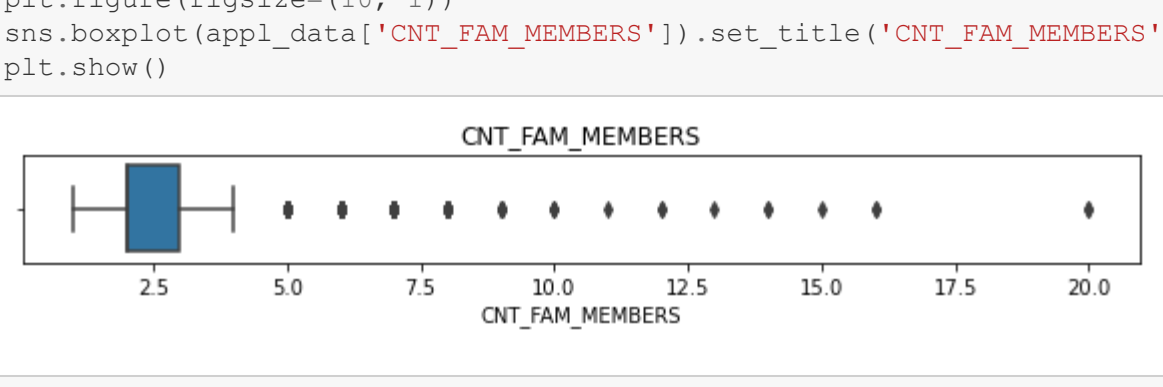
```
In [35]: plt.figure(figsize=(10, 1))
sns.boxplot(appl_data['CNT_FAM_MEMBERS']).set_title('CNT_FAM_MEMBERS')
plt.show()
```




```
In [36]: #We can observe that the number of outliers are low
```

```
In [37]: appl_data.columns[appl_data.isna().any()]
Out[37]: Index(['AMT_ANNUITY', 'AMT_GOODS_PRICE', 'EXT_SOURCE_2', 'EXT_SOURCE_3',
       'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF_30_CNT_SOCIAL_CIRCLE',
       'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCLE',
       'DAYS_LAST_PHONE_CHANGE', 'AMT_REQ_CREDIT_BUREAU_HOUR',
       'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK',
       'AMT_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT',
       'AMT_REQ_CREDIT_BUREAU_YEAR'],
      dtype='object')
```

Column: 'AMT_ANNUITY'

```
In [38]: no_rows = appl_data['AMT_ANNUITY'].isna().sum()
no_rows
Out[38]: 12
```

We can note that the number of values here is very less.
Annuity being a major contributing factor based the business context from **column_description.csv** data, imputing this value could bias the result. Considering we have large number of rows in comparision with this, we can **drop** these rows

```
In [39]: appl_data = appl_data.loc[~appl_data['AMT_ANNUITY'].isna()]
```

```
In [40]: no_rows_dropped = no_rows
no_rows_dropped
Out[40]: 12
```

```
In [41]: #imputed. so no null values anymore.
appl_data['AMT_ANNUITY'].isna().sum()
Out[41]: 0
```

Column: AMT_GOODS_PRICE

is dependant on loan type being consumer goods. The loan type column relevant to this is 'NAME_CONTRACT_TYPE'. If the NAME_CONTRACT_TYPE is Consumer Loan, then this the price of the consumer item being purchased by the client using this loan.

```
In [42]: appl_data['NAME_CONTRACT_TYPE'].value_counts()
Out[42]: Cash loans         278220
Revolving loans     29279
Name: NAME_CONTRACT_TYPE, dtype: int64
```

We observe that we have no records with us which has the relevant NAME_CONTRACT_TYPE
This implies this value **cannot be imputed**

```
In [43]: appl_data['AMT_GOODS_PRICE'].isna().sum()
Out[43]: 278
```

```
In [44]: appl_data.loc[appl_data['AMT_GOODS_PRICE'].isna()]['NAME_CONTRACT_TYPE'].value_counts()
Out[44]: Revolving loans    278
Name: NAME_CONTRACT_TYPE, dtype: int64
```

We observe that all the records with AMT_GOODS_PRICE as NaN are ***Revolving loans***
and hence they **do not have an AMT_GOODS_PRICE** associated with it.
Thus, these values can be safely ***imputed with '0'*** based on the business context.

```
In [45]: appl_data.loc[appl_data['AMT_GOODS_PRICE'].isna(), 'AMT_GOODS_PRICE'] = 0.0
```

```
In [46]: appl_data['AMT_GOODS_PRICE'].isna().sum()
#imputed. Thus no more NaNs
Out[46]: 0
```

```
In [47]: appl_data.columns[appl_data.isna().any()]
Out[47]: Index(['EXT_SOURCE_2', 'EXT_SOURCE_3', 'OBS_30_CNT_SOCIAL_CIRCLE',
       'DEF_30_CNT_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCLE',
       'DEF_60_CNT_SOCIAL_CIRCLE', 'DAYS_LAST_PHONE_CHANGE',
       'AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY',
       'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUREAU_MON',
       'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR'],
      dtype='object')
```

Columns: EXT_SOURCE_2

Based on the information from the **column_description**,
EXT_SOURCE_2 is a "Normalized Score" that are sourced from an external data source. Considering this, EXT_SOURCE_2 could potentially be significant for business and hence imputing them might not be advicable.

```
In [48]: no_rows = appl_data['EXT_SOURCE_2'].isna().sum()
no_rows
Out[48]: 660
```

```
In [49]: appl_data.shape
Out[49]: (307499, 53)
```

```
In [50]: #no_rows_dropped has the number of rows dropped already.
```

```
In [51]: no_rows + no_rows_dropped
Out[51]: 672
```

Consider, the no_rows_dropped, together with the no_rows with EXT_SOURCE_2 is NaN,
is **significantly less than the total no. of records**,
we can **safely drop** this values to avoid skewing the result.

```
In [52]: appl_data = appl_data.loc[~appl_data['EXT_SOURCE_2'].isna()]
```

```
In [53]: appl_data['EXT_SOURCE_2'].isna().sum()
#rows are dropped. hence, no more NaNs
Out[53]: 0
```

```
In [54]: no_rows_dropped = no_rows_dropped + no_rows
no_rows_dropped
Out[54]: 672
```

Columns: EXT_SOURCE_3

Based on the information from the **column_description**,
EXT_SOURCE_3 is a "Normalized Score" that are sourced from an external data source. Considering this, EXT_SOURCE_3 could potentially be significant for business
and hence imputing them might not be advisable.

We have to analyse the data before decision

```
In [55]: no_rows = appl_data['EXT_SOURCE_3'].isna().sum()
no_rows
Out[55]: 60734
```

```
In [56]: round(no_rows/appl_data.shape[0], 3)
Out[56]: 0.198
```

This is a considerable percentage.
Being a 'Score' from an external source, with no further business information,
we ***must consider dropping*** these rows.

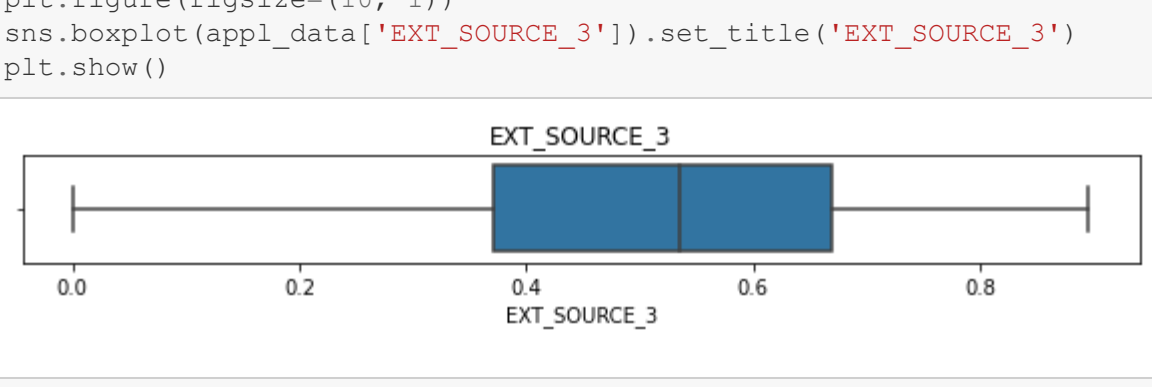
```
In [57]: mean = appl_data['EXT_SOURCE_3'].mean()
mean
Out[57]: 0.5107773158925729
```

```
In [58]: mode = appl_data['EXT_SOURCE_3'].mode()
mode
Out[58]: 0    0.7463
dtype: float64
```

```
In [59]: median = round(appl_data['EXT_SOURCE_3'].median(), 6)
median
Out[59]: 0.535276
```

```
In [60]: es3_data =  appl_data.loc[~appl_data['EXT_SOURCE_3'].isna()]
```

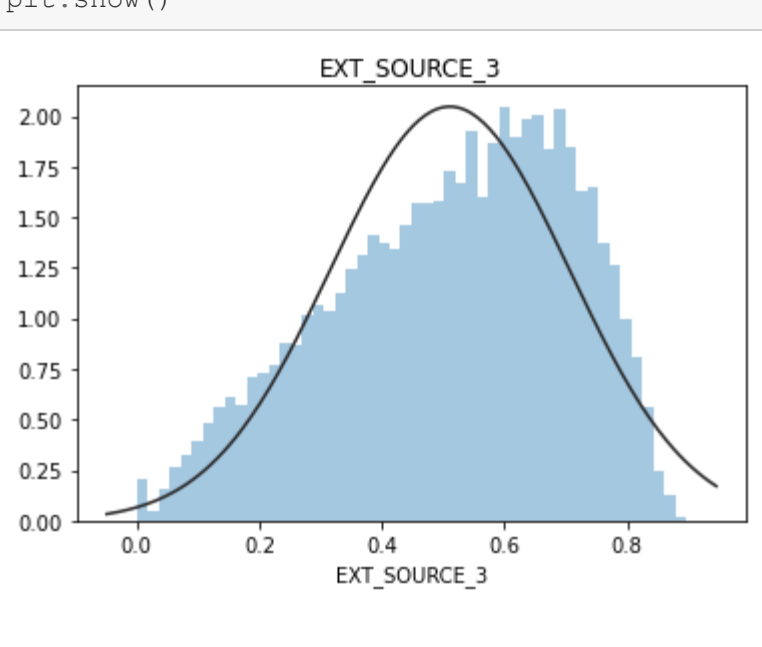
```
In [61]: plt.figure(figsize=(10, 1))
sns.boxplot(appl_data['EXT_SOURCE_3']).set_title('EXT_SOURCE_3')
plt.show()
```




```
In [62]: es3_data['EXT_SOURCE_3'].describe()
Out[62]: count    246105.000000
mean          0.510777
std           0.194854
min           0.000527
25%           0.370650
50%           0.535276
75%           0.669057
max           0.896010
Name: EXT_SOURCE_3, dtype: float64
```

The above implies the available score are fairly distributed without any outliers.
We can also observe the percentage of missing values is higher than the standard deviation of the rest of the values.

```
In [63]: from scipy.stats import norm

sns.distplot(appl_data['EXT_SOURCE_3'], fit=norm, kde=False).set_title('EXT_SOURCE_3')
plt.show()
```



Considering these results, we will refrain from imputing this value
When involving this variable for analysis, depending on the co-relation,
we can analyse ***without the missing value rows for the related variables*** as well.

```
In [64]: appl_data.columns[appl_data.isna().any()]
Out[64]: Index(['EXT_SOURCE_3', 'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF_30_CNT_SOCIAL_CIRCLE',
       'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCLE',
       'AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY',
       'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUREAU_MON',
       'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR'],
      dtype='object')
```

Columns: Observation of Defaulters and Overdue Payments in the region

- OBS_30_CNT_SOCIAL_CIRCLE
- DEF_30_CNT_SOCIAL_CIRCLE
- OBS_60_CNT_SOCIAL_CIRCLE
- DEF_60_CNT_SOCIAL_CIRCLE

```
In [65]: no_rows = appl_data['OBS_30_CNT_SOCIAL_CIRCLE'].isna().sum()
no_rows
Out[65]: 1017
```

```
In [66]: SK_ID_CURRs = appl_data.loc[appl_data['OBS_30_CNT_SOCIAL_CIRCLE'].isna()]['SK_ID_CURR']
SK_ID_CURRs
Out[66]: 68        100080
394       100457
397       100460
457       100527
1042      101209
           ...
305526    453980
305623    454093
305641    454116
305642    454117
307402    456135
Name: SK_ID_CURR, Length: 1017, dtype: int64
```

```
In [67]: appl_data['DEF_30_CNT_SOCIAL_CIRCLE'].isna().sum()
Out[67]: 1017
```

```
In [68]: appl_data.loc[appl_data['DEF_30_CNT_SOCIAL_CIRCLE'].isna()]['SK_ID_CURR'].isin(SK_ID_CURRs).sum()
Out[68]: 1017
```

```
In [69]: appl_data['OBS_60_CNT_SOCIAL_CIRCLE'].isna().sum()
Out[69]: 1017
```

```
In [70]: appl_data.loc[appl_data['OBS_60_CNT_SOCIAL_CIRCLE'].isna()]['SK_ID_CURR'].isin(SK_ID_CURRs).sum()
Out[70]: 1017
```

```
In [71]: appl_data['DEF_60_CNT_SOCIAL_CIRCLE'].isna().sum()
Out[71]: 1017
```

```
In [72]: appl_data.loc[appl_data['DEF_60_CNT_SOCIAL_CIRCLE'].isna()]['SK_ID_CURR'].isin(SK_ID_CURRs).sum()
Out[72]: 1017
```

```
In [73]: appl_data.shape
Out[73]: (306839, 53)
```

```
In [74]: no_rows_dropped + no_rows
Out[74]: 1689
```

```
In [75]: round((no_rows_dropped + no_rows)/appl_data.shape[0],3)
Out[75]: 0.006
```

- We observe that the records are ***missing for the same set of applicants***
- Considering the total number of records dropped sofar
  and the number of rows in these columns with missing values,

  The number of dropped records is significantly low and ***can be dropped***.

```
In [76]: appl_data = appl_data.loc[~appl_data['SK_ID_CURR'].isin(SK_ID_CURRs)]
```

```
In [77]: no_rows_dropped = no_rows_dropped + no_rows
no_rows_dropped
Out[77]: 1689
```

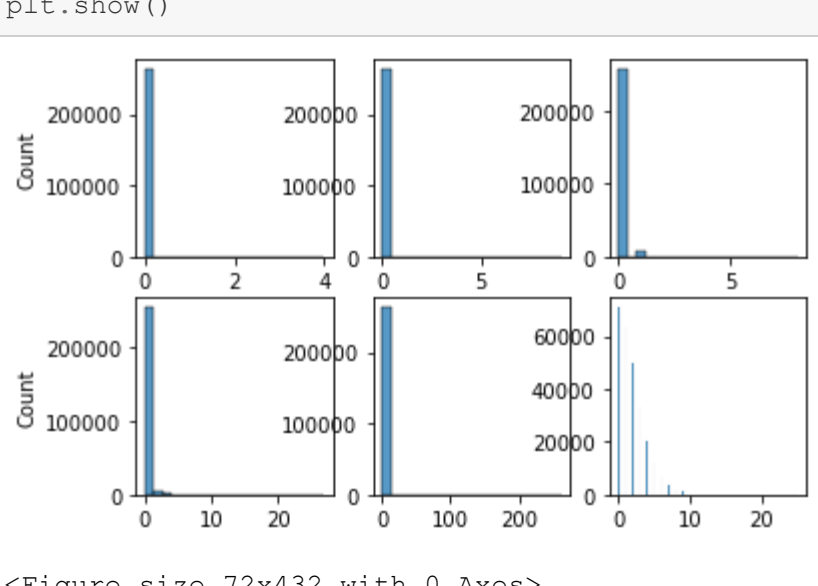
```
In [78]: appl_data['OBS_30_CNT_SOCIAL_CIRCLE'].isna().sum()
#the records are dropped. so no more missing values
Out[78]: 0
```

```
In [79]: appl_data.columns[appl_data.isna().any()]
Out[79]: Index(['EXT_SOURCE_3', 'AMT_REQ_CREDIT_BUREAU_HOUR',
       'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK',
       'AMT_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT',
       'AMT_REQ_CREDIT_BUREAU_YEAR'],
      dtype='object')
```

Columns: Number of enquiries to Credit Bureau about the client

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

```
In [80]: f, axes = plt.subplots(2, 3)
plt.figure(figsize=(1, 6))
s = sns.histplot(data=appl_data, x='AMT_REQ_CREDIT_BUREAU_HOUR', ax=axes[0][0])
s.set(xlabel=None)

s = sns.histplot(data=appl_data, x='AMT_REQ_CREDIT_BUREAU_DAY', ax=axes[0][1])
s.set(xlabel=None)
s.set(ylabel=None)

s = sns.histplot(data=appl_data, x='AMT_REQ_CREDIT_BUREAU_WEEK', ax=axes[0][2])
s.set(xlabel=None)
s.set(ylabel=None)

s = sns.histplot(data=appl_data, x='AMT_REQ_CREDIT_BUREAU_MON', ax=axes[1][0])
s.set(xlabel=None)

s = sns.histplot(data=appl_data, x='AMT_REQ_CREDIT_BUREAU_QRT', ax=axes[1][1])
s.set(xlabel=None)
s.set(ylabel=None)

s = sns.histplot(data=appl_data, x='AMT_REQ_CREDIT_BUREAU_YEAR', ax=axes[1][2])
s.set(xlabel=None)
s.set(ylabel=None)

plt.show()
```




```
<Figure size 72x432 with 0 Axes>
```

We observe that all these columns are mostly indicate that '0' requests are made to the credit bureau. Rest of the values are very less significant. The result is biased for the business reason that these values are this making any form or imputation unreliable.

Hence we can considering dropping these columns.

```
In [81]: drop_cols = appl_data.filter(regex='AMT_REQ_CREDIT_BUREAU', axis=1).columns
```

```
In [82]: #Dropping the above columns
appl_data.drop(drop_cols,axis=1,inplace=True)
```

```
In [83]: appl_data.columns[appl_data.isna().any()]
Out[83]: Index(['EXT_SOURCE_3'], dtype='object')
```

2.4 Data Preparation for Numerical Data Analysis

```
In [84]: appl_data.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 305822 entries, 0 to 307510
Data columns (total 47 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   SK_ID_CURR                   305822 non-null  int64
 1   TARGET                       305822 non-null  int64
 2   NAME_CONTRACT_TYPE           305822 non-null  object
 3   CODE_GENDER                  305822 non-null  object
 4   FLAG_OWN_CAR                 305822 non-null  object
 5   FLAG_OWN_REALTY              305822 non-null  object
 6   CNT_CHILDREN                 305822 non-null  int64
 7   AMT_INCOME_TOTAL             305822 non-null  float64
 8   AMT_CREDIT                   305822 non-null  float64
 9   AMT_ANNUITY                  305822 non-null  float64
 10  AMT_GOODS_PRICE              305822 non-null  float64
 11  NAME_TYPE_SUITE              305822 non-null  object
 12  NAME_INCOME_TYPE             305822 non-null  object
 13  NAME_EDUCATION_TYPE          305822 non-null  object
 14  NAME_FAMILY_STATUS           305822 non-null  object
 15  NAME_HOUSING_TYPE            305822 non-null  object
 16  REGION_POPULATION_RELATIVE   305822 non-null  float64
 17  DAYS_BIRTH                   305822 non-null  int64
 18  DAYS_EMPLOYED                305822 non-null  int64
 19  DAYS_REGISTRATION            305822 non-null  float64
 20  DAYS_ID_PUBLISH              305822 non-null  int64
 21  FLAG_MOBIL                   305822 non-null  int64
 22  FLAG_EMP_PHONE               305822 non-null  int64
 23  FLAG_WORK_PHONE              305822 non-null  int64
 24  FLAG_CONT_MOBILE             305822 non-null  int64
 25  FLAG_PHONE                   305822 non-null  int64
 26  FLAG_EMAIL                   305822 non-null  int64
 27  OCCUPATION_TYPE              305822 non-null  object
 28  CNT_FAM_MEMBERS              305822 non-null  float64
 29  REGION_RATING_CLIENT         305822 non-null  int64
 30  REGION_RATING_CLIENT_W_CITY  305822 non-null  int64
 31  WEEKDAY_APPR_PROCESS_START   305822 non-null  object
 32  HOUR_APPR_PROCESS_START      305822 non-null  int64
 33  REG_REGION_NOT_LIVE_REGION   305822 non-null  int64
 34  REG_REGION_NOT_WORK_REGION   305822 non-null  int64
 35  LIVE_REGION_NOT_WORK_REGION  305822 non-null  int64
 36  REG_CITY_NOT_LIVE_CITY       305822 non-null  int64
 37  REG_CITY_NOT_WORK_CITY       305822 non-null  int64
 38  LIVE_CITY_NOT_WORK_CITY      305822 non-null  int64
 39  ORGANIZATION_TYPE            305822 non-null  object
 40  EXT_SOURCE_2                 305822 non-null  float64
 41  EXT_SOURCE_3                 245358 non-null  float64
 42  OBS_30_CNT_SOCIAL_CIRCLE     305822 non-null  float64
 43  DEF_30_CNT_SOCIAL_CIRCLE     305822 non-null  float64
 44  OBS_60_CNT_SOCIAL_CIRCLE     305822 non-null  float64
 45  DEF_60_CNT_SOCIAL_CIRCLE     305822 non-null  float64
 46  DAYS_LAST_PHONE_CHANGE       305822 non-null  float64
dtypes: float64(14), int64(21), object(12)
memory usage: 112.0+ MB
```

```
In [85]: days_cols = appl_data.filter(regex='DAYS_', axis=1).columns
days_cols
Out[85]: Index(['DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH',
       'DAYS_LAST_PHONE_CHANGE'],
      dtype='object')
```

```
In [86]: appl_data[[days_cols[0]]]
```

Out[86]:

	DAYS_BIRTH
0	-9461
1	-16765
2	-19046
3	-19005
4	-19932
...	...
307506	-9327
307507	-20775
307508	-14966
307509	-11961
307510	-16856

305822 rows × 1 columns

```
In [87]: #DAYS_BIRTH is the Client's age in days at the time of application
#convert DAYS_BIRTH to AGE
appl_data['AGE'] = -round(appl_data['DAYS_BIRTH']/365,0)
appl_data['AGE']
Out[87]: 0         26.0
1         46.0
2         52.0
3         52.0
4         55.0
          ...
307506    26.0
307507    57.0
307508    41.0
307509    33.0
307510    46.0
Name: AGE, Length: 305822, dtype: float64
```

```
In [88]: #DAYS_EMPLOYED
#How many days before the application the person started current employment
#convert DAYS_EMPLOYED to YEARS_EMPLOYED
appl_data['YEARS_EMPLOYED'] = abs(round(appl_data['DAYS_EMPLOYED']/365, 0))

#DAYS_REGISTRATION
#How many days before the application did client change his registration
#convert DAYS_REGISTRATION to YEARS_REGISTRATION
appl_data['YEARS_REGISTRATION'] = abs(round(appl_data['DAYS_REGISTRATION']/365, 0))

#DAYS_ID_PUBLISH
#How many days before the application did client change the identity document with which he applied for
 the loan
#convert DAYS_ID_PUBLISH to YEARS_ID_PUBLISH
appl_data['YEARS_ID_PUBLISH'] = abs(round(appl_data['DAYS_ID_PUBLISH']/365, 0))

#DAYS_ID_PUBLISH
#How many days before the application did client change the identity document with which he applied for
 the loan
#convert DAYS_LAST_PHONE_CHANGE to YEARS_LAST_PHONE_CHANGE
appl_data['YEARS_LAST_PHONE_CHANGE'] = abs(round(appl_data['DAYS_LAST_PHONE_CHANGE']/365, 0))
```

```
In [89]: appl_data[['AGE', 'YEARS_EMPLOYED', 'YEARS_REGISTRATION', 'YEARS_ID_PUBLISH', 'YEARS_LAST_PHONE_CHANGE'
]]
```

Out[89]:

	AGE	YEARS_EMPLOYED	YEARS_REGISTRATION	YEARS_ID_PUBLISH	YEARS_LAST_PHONE_CHANGE
0	26.0	2.0	10.0	6.0	3.0
1	46.0	3.0	3.0	1.0	2.0
2	52.0	1.0	12.0	7.0	2.0
3	52.0	8.0	27.0	7.0	2.0
4	55.0	8.0	12.0	9.0	3.0
...	...	...	...	...	...
307506	26.0	1.0	23.0	5.0	1.0
307507	57.0	1001.0	12.0	11.0	0.0
307508	41.0	22.0	18.0	14.0	5.0
307509	33.0	13.0	7.0	3.0	1.0
307510	46.0	3.0	14.0	1.0	2.0

305822 rows × 5 columns

```
In [90]: ## Drop all the days colns as we have converted them to years for computation
appl_data.drop(days_cols,axis=1,inplace=True)
```

```
In [91]: appl_data[['AGE', 'YEARS_EMPLOYED', 'YEARS_REGISTRATION', 'YEARS_ID_PUBLISH', 'YEARS_LAST_PHONE_CHANGE'
]]
```

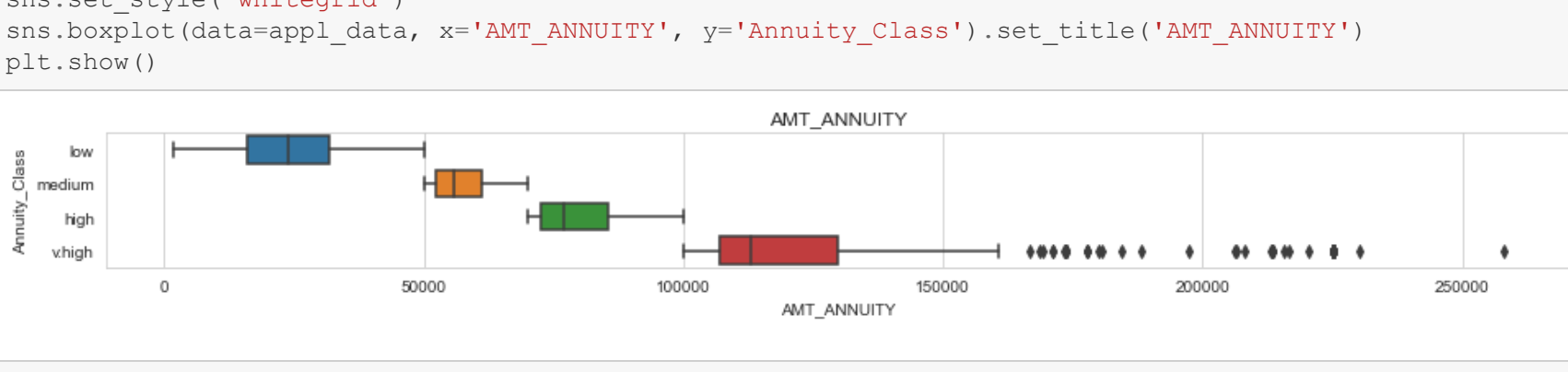
Out[91]:

	AGE	YEARS_EMPLOYED	YEARS_REGISTRATION	YEARS_ID_PUBLISH	YEARS_LAST_PHONE_CHANGE
0	26.0	2.0	10.0	6.0	3.0
1	46.0	3.0	3.0	1.0	2.0
2	52.0	1.0	12.0	7.0	2.0
3	52.0	8.0	27.0	7.0	2.0
4	55.0	8.0	12.0	9.0	3.0
...	...	...	...	...	...
307506	26.0	1.0	23.0	5.0	1.0
307507	57.0	1001.0	12.0	11.0	0.0
307508	41.0	22.0	18.0	14.0	5.0
307509	33.0	13.0	7.0	3.0	1.0
307510	46.0	3.0	14.0	1.0	2.0

305822 rows × 5 columns

2.5 Binning required columns

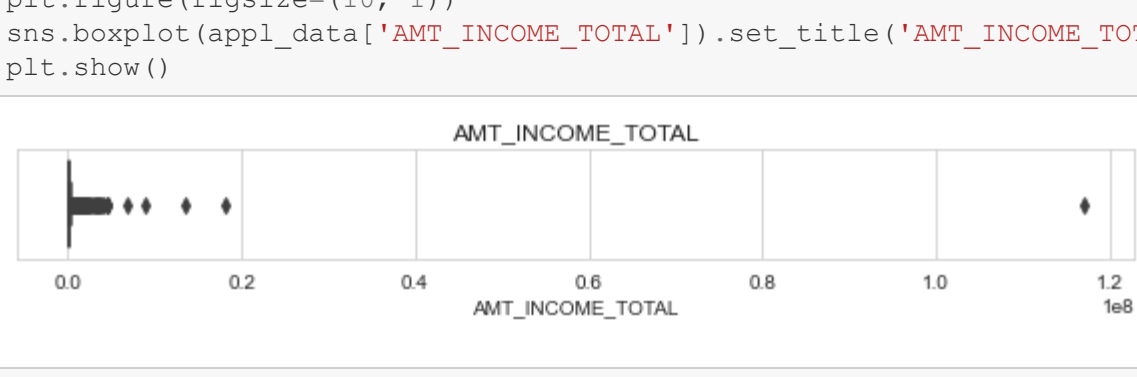
```
In [92]: ## Binning for Annuity and AMT_INCOME_TOTAL
appl_data['Annuity_Class'] = pd.cut(appl_data['AMT_ANNUITY'],\
                                    bins=[0,50000,70000,100000,260000],\
                                    labels=['low','medium','high','v.high'])

plt.figure(figsize=(16, 1.5))
sns.set_style("whitegrid")
sns.boxplot(data=appl_data, x='AMT_ANNUITY', y='Annuity_Class').set_title('AMT_ANNUITY')
plt.show()
```




```
In [93]: appl_data['Annuity_Class'].value_counts()
Out[93]: low       284653
medium     18096
high        2571
v.high       502
Name: Annuity_Class, dtype: int64
```

```
In [94]: appl_data.shape
Out[94]: (305822, 48)
```

```
In [95]: appl_data['AMT_INCOME_TOTAL'].describe()
plt.figure(figsize=(10, 1))
sns.boxplot(appl_data['AMT_INCOME_TOTAL']).set_title('AMT_INCOME_TOTAL')
plt.show()
```




```
In [96]: ##The above record is having a very high income value and is biasing the other records.
##Dropping this outlier
```

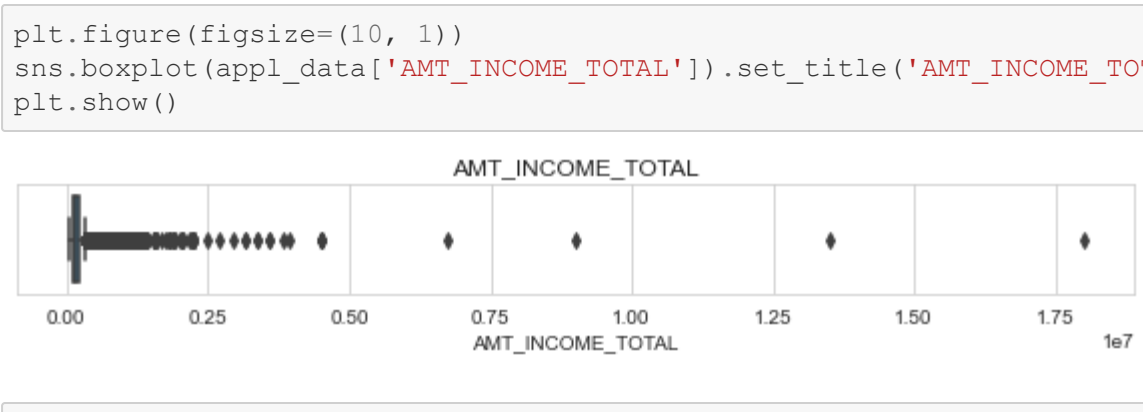
```
In [97]: ## Removing the outlier
appl_data = appl_data.sort_values(by=['AMT_INCOME_TOTAL'],ascending=False)[1:]
```

```
In [98]: mean = appl_data['AMT_INCOME_TOTAL'].mean()
mean
Out[98]: 168297.0670391667
```

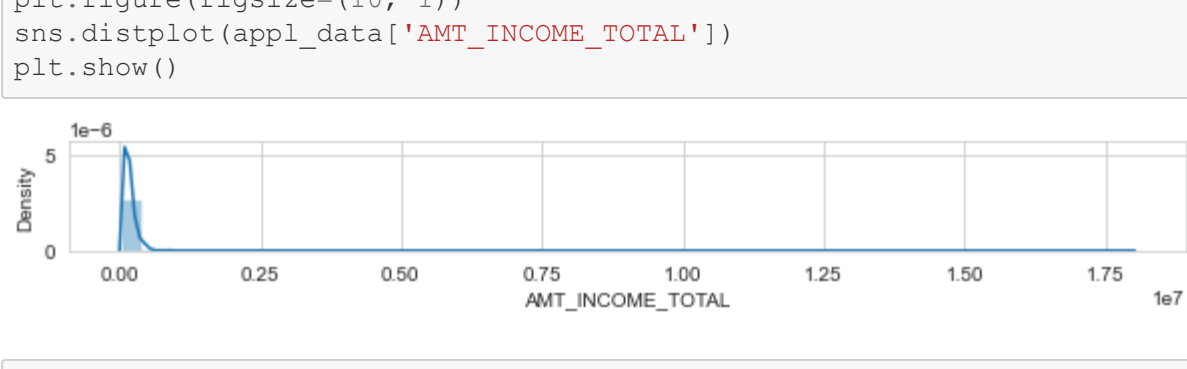
```
In [99]: appl_data.loc[appl_data['AMT_INCOME_TOTAL'] > mean].shape
Out[99]: (119728, 48)
```

```
In [100]: appl_data['AMT_INCOME_TOTAL'].max()
Out[100]: 18000090.0
```

```
In [101]: plt.figure(figsize=(10, 1))
sns.boxplot(appl_data['AMT_INCOME_TOTAL']).set_title('AMT_INCOME_TOTAL')
plt.show()
```




```
In [102]: plt.figure(figsize=(10, 1))
sns.distplot(appl_data['AMT_INCOME_TOTAL'])
plt.show()
```




```
In [103]: appl_data['Income_Class'] = pd.qcut(appl_data['AMT_INCOME_TOTAL'],\
                                     q=10, precision=2)
```

```
In [104]: appl_income_grps = appl_data.groupby(by='Income_Class')['AMT_INCOME_TOTAL'].describe()
appl_income_grps
```

Out[104]:

Income_Class	count	mean	std	min	25%	50%	75%	max
(25649.99, 81000.0]	33211.0	66246.127214	12096.275507	25650.0	58500.00	67500.0	76500.0	81000.00
(81000.0, 99000.0]	30101.0	91088.792065	3401.030365	81180.0	90000.00	90000.0	90000.0	99000.00
(99000.0, 112500.0]	36694.0	111332.451613	2844.519397	99031.5	112500.00	112500.0	112500.0	112500.00
(112500.0, 135000.0]	48598.0	131847.429542	5660.457554	112549.5	130500.00	135000.0	135000.0	135000.00
(135000.0, 146771.91]	4307.0	142941.286222	2050.488281	135256.5	143696.25	144000.0	144000.0	146771.91
(146771.91, 162000.0]	30991.0	157136.798700	2411.856114	146812.5	157500.00	157500.0	157500.0	162000.00
(162000.0, 180000.0]	30560.0	178058.020524	4199.414450	162279.0	180000.00	180000.0	180000.0	180000.00
(180000.0, 225000.0]	44577.0	212716.093524	12450.582865	180090.0	202500.00	216000.0	225000.0	225000.00
(225000.0, 270000.0]	19845.0	259479.343959	12473.345567	225112.5	247500.00	270000.0	270000.0	270000.00
(270000.0, 18000090.0]	26937.0	388889.574948	219676.271830	270013.5	315000.00	355500.0	405000.0	18000090.00

```
In [105]: income_levels_iv = appl_income_grps.index
list(income_levels_iv)
Out[105]: [Interval(25649.99, 81000.0, closed='right'),
 Interval(81000.0, 99000.0, closed='right'),
 Interval(99000.0, 112500.0, closed='right'),
 Interval(112500.0, 135000.0, closed='right'),
 Interval(135000.0, 146771.91, closed='right'),
 Interval(146771.91, 162000.0, closed='right'),
 Interval(162000.0, 180000.0, closed='right'),
 Interval(180000.0, 225000.0, closed='right'),
 Interval(225000.0, 270000.0, closed='right'),
 Interval(270000.0, 18000090.0, closed='right')]
```

```
In [106]: appl_data['NAME_INCOME_TYPE'].value_counts()
Out[106]: Working                 157982
Commercial associate     71154
Pensioner                55038
State servant            21595
Unemployed                  19
Student                     18
Businessman                 10
Maternity leave              5
Name: NAME_INCOME_TYPE, dtype: int64
```

```
In [107]: appl_data['INCOME_TYPE']=''
appl_data['INCOME_TYPE']=np.where(appl_data['NAME_INCOME_TYPE'].isin(['Maternity leave','Businessman',
'Student','Unemployed','Pensioner']),'Other',appl_data['NAME_INCOME_TYPE'])
```

```
In [108]: appl_data['INCOME_TYPE']
Out[108]: 203693    Commercial associate
246858    Commercial associate
77768                  Working
131127                 Working
287463                 Working
                  ...
240137                   Other
246104                   Other
186643                   Other
20727                    Other
1678                   Working
Name: INCOME_TYPE, Length: 305821, dtype: object
```

```
In [109]: #appl_data.info()
```

2.6 Imbalance Ratio

- We observe the imbalance ratio in the target variable and if it is high, then we shall split the datasets for brevity.

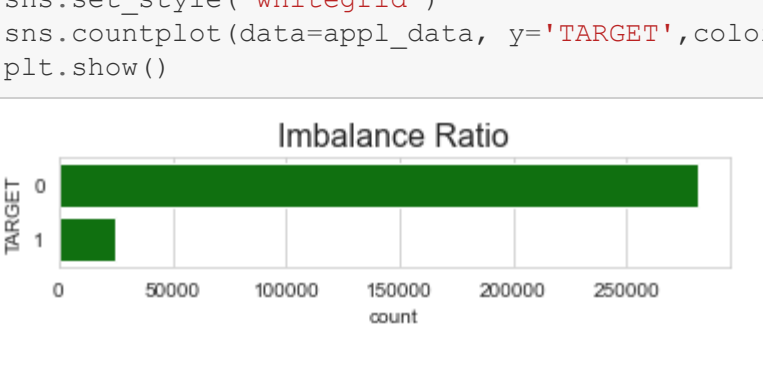
```
In [110]: total_appl = appl_data['TARGET'].value_counts().sum()
normal_appl = appl_data['TARGET'].value_counts()[0]
defaulter_appl = appl_data['TARGET'].value_counts()[1]

normal_percentage = normal_appl/total_appl
print('Non-Defaulter Applicants:', normal_appl, 'Percentage', round(normal_percentage, 2))

defaulter_percentage = defaulter_appl/total_appl
print('Defaulters:', defaulter_appl, 'Percentage', round(defaulter_percentage, 2))
Non-Defaulter Applicants: 281085 Percentage 0.92
Defaulters: 24736 Percentage 0.08
```

Note:

- Only ***8%*** of the applicants are ***Defaulters(TARGET=1)***.
- Others (***92%***) are ***Non-Defaulters(TARGET=0)***
- We will now split them into two different sets and analyze accordingly.

```
In [111]: # Target 0 refers to people with payment difficulties
target0 = appl_data.loc[appl_data['TARGET'] == 0]

# Target 1 without any payment issues
target1 = appl_data.loc[appl_data['TARGET'] == 1]
```

```
In [112]: plt.figure(figsize=[6,1])
sns.set_style('whitegrid')
sns.countplot(data=appl_data, y='TARGET',color='green').set_title('Imbalance Ratio', fontsize=16)
plt.show()
```



2.7 Univariate Analysis

Here, we look at variables directly influencing the target.
We use the term **Target** for the target variable.
We split the data into two datasets based on the Imbalance Ratio between the two types of applicants(Clients*)

- **Target 1:** `Defaulter Applicants` -- (Clients*) `with` payment difficulties: he/she had late payment more than X days on at least one of the first Y installments of the loan in our sample.
- **Target 0:** `Non-Defaulter Applicants` (All other cases*) `with no` difficulties paying due on time

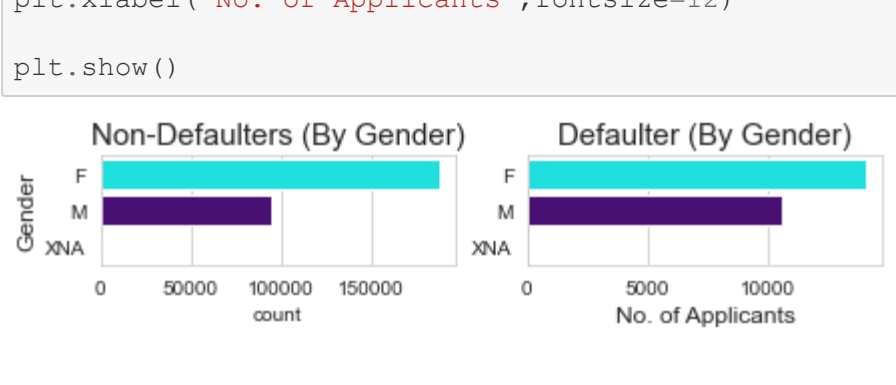
**As described in the columns_description.csv document*

2.7.1 Reusable function for univariate analysis

```
In [113]: def __Univariate(x):
              plt.figure(figsize=[12,6])
              sns.set_style('whitegrid')
              sns.countplot(data=appl_data,x=x,hue='TARGET')
              plt.xticks(rotation=90)
              plt.title(f'Distribution of {x} ',fontsize=25)
              plt.yscale('log')
              plt.xlabel(x,fontsize=20)
              plt.show()
```

2.7.2 Gender vs Target

```
In [114]: plt.figure(figsize=[7,1])
          sns.set_style('whitegrid')
          plt.ylabel('Gender', fontsize=10)

          _gender_order = target0['CODE_GENDER'].value_counts().index

          plt.subplot(1,2,1)
          sns.countplot(data=target0, y='CODE_GENDER', order=_gender_order, palette=['cyan', 'Indigo'])\
                  .set_title('Non-Defaulters (By Gender)', fontsize=15)
          plt.ylabel('Gender', fontsize=12)

          plt.subplot(1,2,2)
          sns.countplot(data=target1, y='CODE_GENDER', order=_gender_order, palette=['cyan', 'Indigo'])\
                  .set_title('Defaulter (By Gender)', fontsize=15)
          plt.ylabel('', fontsize=12)
          #plt.xscale('log')

          plt.xlabel('No. of Applicants',fontsize=12)

          plt.show()
```



Observations

We generally notice that

- `Female` applicants are higher than the **Male** applicants and directly influences the numbers
- `Gender has No Impact on the Target`, other than the generic imbalance across the Defaulters and the Non-Defaulters.
- We will `Ignore Gender` for all further analysis

2.7.3 Income vs Target

```
In [115]: plt.figure(figsize=[15,10])
          sns.set_style('whitegrid')

          plt.subplot(2,1,1)
          sns.countplot(data=target0, y='Income_Class', palette=['cyan', 'Indigo'])\
                  .set_title('Non-Defaulter Income Levels', fontsize=16)
          plt.ylabel('Income Levels', fontsize=12)

          plt.subplot(2,1,2)
          sns.countplot(data=target1, y='Income_Class', palette=['cyan', 'Indigo'])\
                  .set_title('Defaulter Income Levels', fontsize=16)
          plt.ylabel('Income Levels', fontsize=12)


          plt.xlabel('No. of Applicants',fontsize=12)

          plt.show()
```

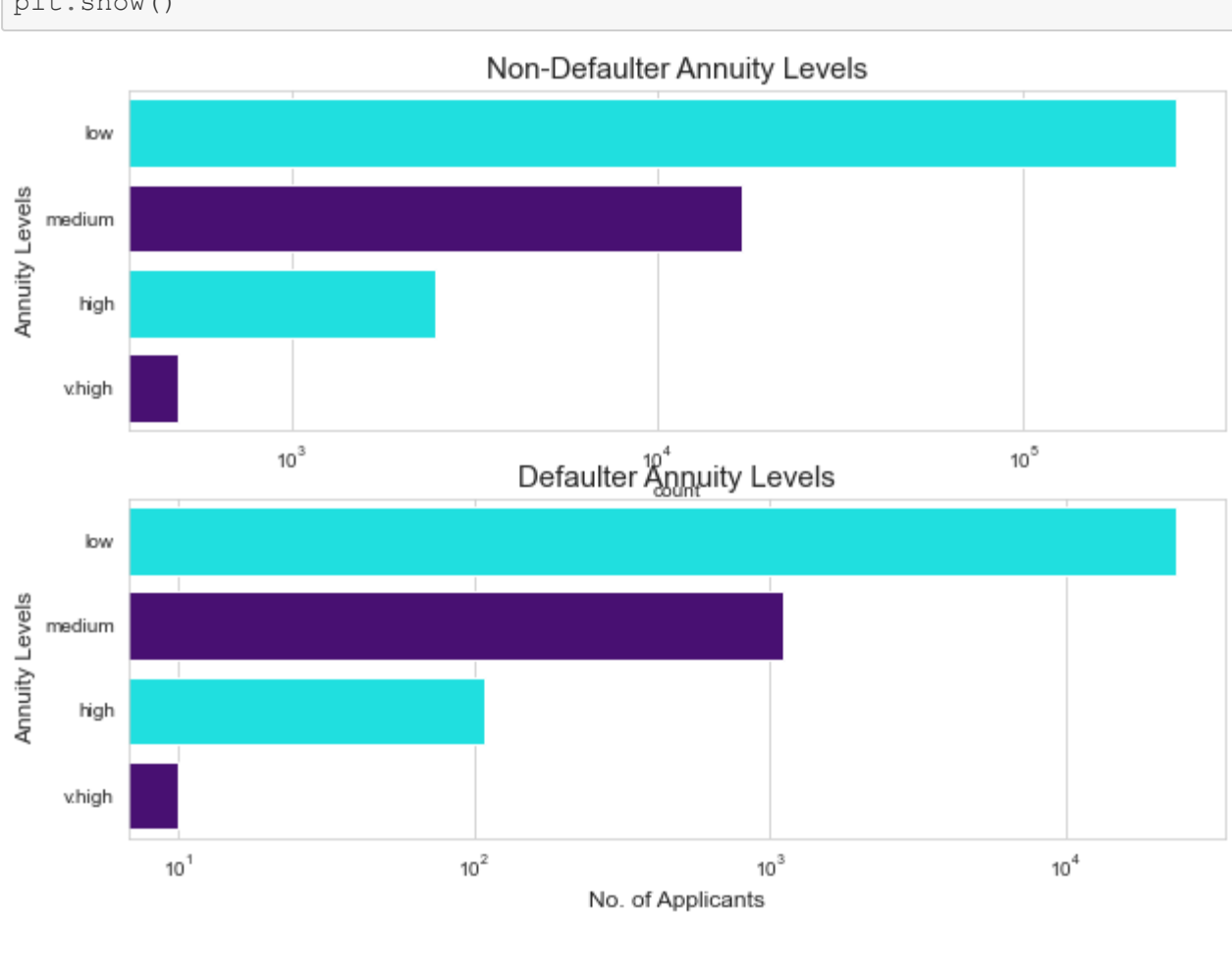

```
In [116]: income_levels_iv[4]
Out[116]: Interval(135000.0, 146771.91, closed='right')

In [117]: appl_data['Income_Class'].value_counts()
Out[117]: (112500.0, 135000.0]      48598
          (180000.0, 225000.0]      44577
          (99000.0, 112500.0]       36694
          (25649.99, 81000.0]       33211
          (146771.91, 162000.0]     30991
          (162000.0, 180000.0]      30560
          (81000.0, 99000.0]        30101
          (270000.0, 18000090.0]    26937
          (225000.0, 270000.0]      19845
          (135000.0, 146771.91]      4307
          Name: Income_Class, dtype: int64
```

Observations

- The (135000.0, 146771.91) income level applicants seem to be `**Very Less(Only 4307)**` in number.
- `**Regardless** Income Levels` different applicants face payment difficulties.
- No. of defaulters and non-defaulters seems to be proportionally `same`
- Considering the above observations, it seems like along with the income,
  there are other factors in play when it comes to the Target.
  We might require bi-variate or multi-variate analysis

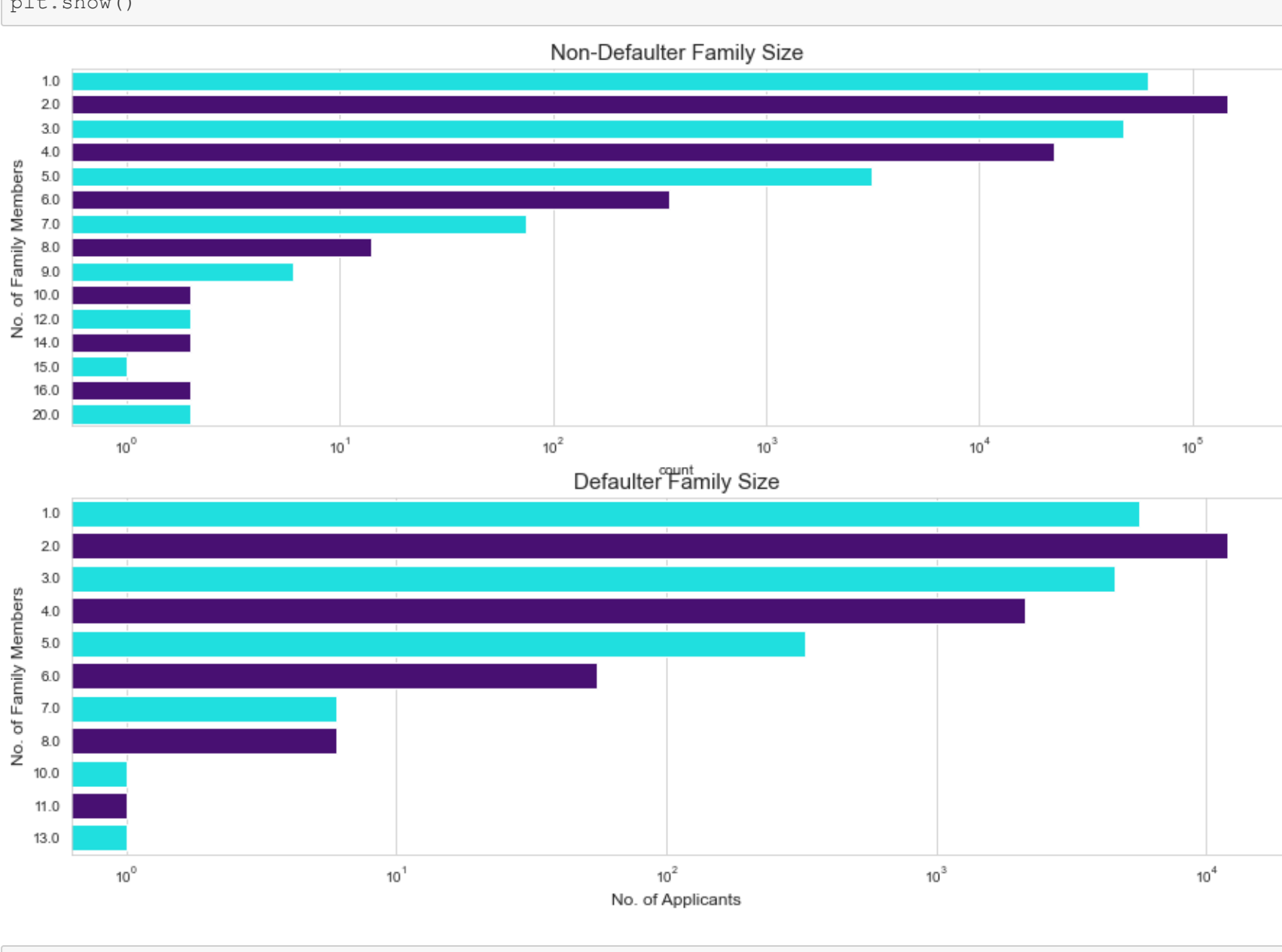
2.7.4 Annuity vs Target

```
In [118]: plt.figure(figsize=[10,7])
          sns.set_style('whitegrid')

          plt.subplot(2,1,1)
          sns.countplot(data=target0, y='Annuity_Class', palette=['cyan', 'Indigo'])\
                  .set_title('Non-Defaulter Annuity Levels', fontsize=15)
          plt.ylabel('Annuity Levels', fontsize=12)
          plt.xscale('log')

          plt.subplot(2,1,2)
          sns.countplot(data=target1, y='Annuity_Class', palette=['cyan', 'Indigo'])\
                  .set_title('Defaulter Annuity Levels', fontsize=15)
          plt.ylabel('Annuity Levels', fontsize=12)

          plt.xscale('log')
          plt.xlabel('No. of Applicants',fontsize=12)

          plt.show()
```



2.7.5 Family Size vs Target

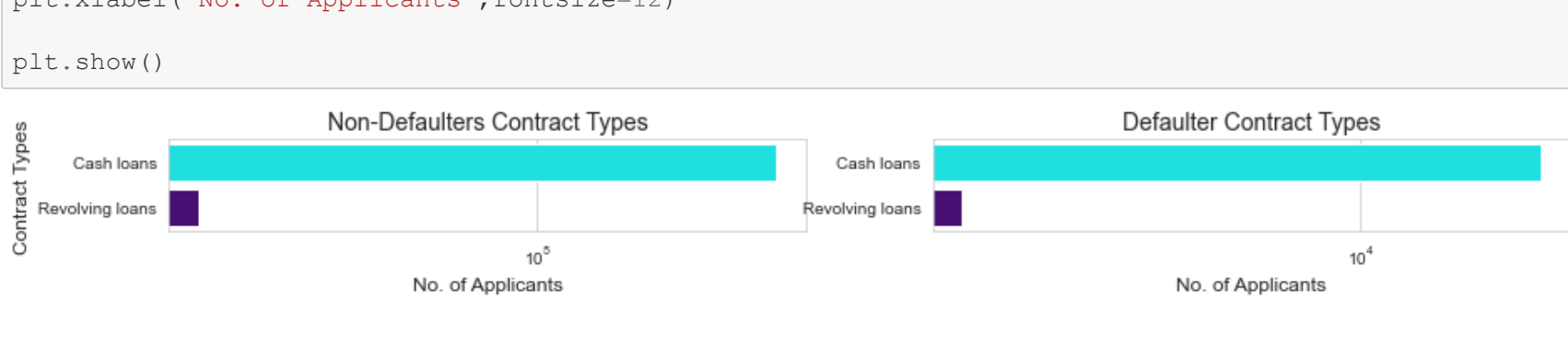
```
In [119]: plt.figure(figsize=[15,10])
          sns.set_style('whitegrid')

          plt.subplot(2,1,1)
          sns.countplot(data=target0, y='CNT_FAM_MEMBERS', palette=['cyan', 'Indigo'])\
                  .set_title('Non-Defaulter Family Size', fontsize=15)
          plt.ylabel('No. of Family Members', fontsize=12)
          plt.xscale('log')

          plt.subplot(2,1,2)
          sns.countplot(data=target1, y='CNT_FAM_MEMBERS', palette=['cyan', 'Indigo'])\
                  .set_title('Defaulter Family Size', fontsize=15)
          plt.ylabel('No. of Family Members', fontsize=12)
          plt.xscale('log')

          plt.xlabel('No. of Applicants',fontsize=12)

          plt.show()
```




```
In [120]: target0['CNT_FAM_MEMBERS'].value_counts()
Out[120]: 2.0     145548
          1.0      61760
          3.0      47741
          4.0      22443
          5.0       3135
          6.0        352
          7.0         75
          8.0         14
          9.0          6
          12.0         2
          20.0         2
          10.0         2
          16.0         2
          14.0         2
          15.0         1
          Name: CNT_FAM_MEMBERS, dtype: int64

In [121]: target1['CNT_FAM_MEMBERS'].value_counts()
Out[121]: 2.0     11971
          1.0      5653
          3.0      4587
          4.0      2129
          5.0       326
          6.0        55
          7.0         6
          8.0         6
          11.0        1
          10.0        1
          13.0        1
          Name: CNT_FAM_MEMBERS, dtype: int64
```

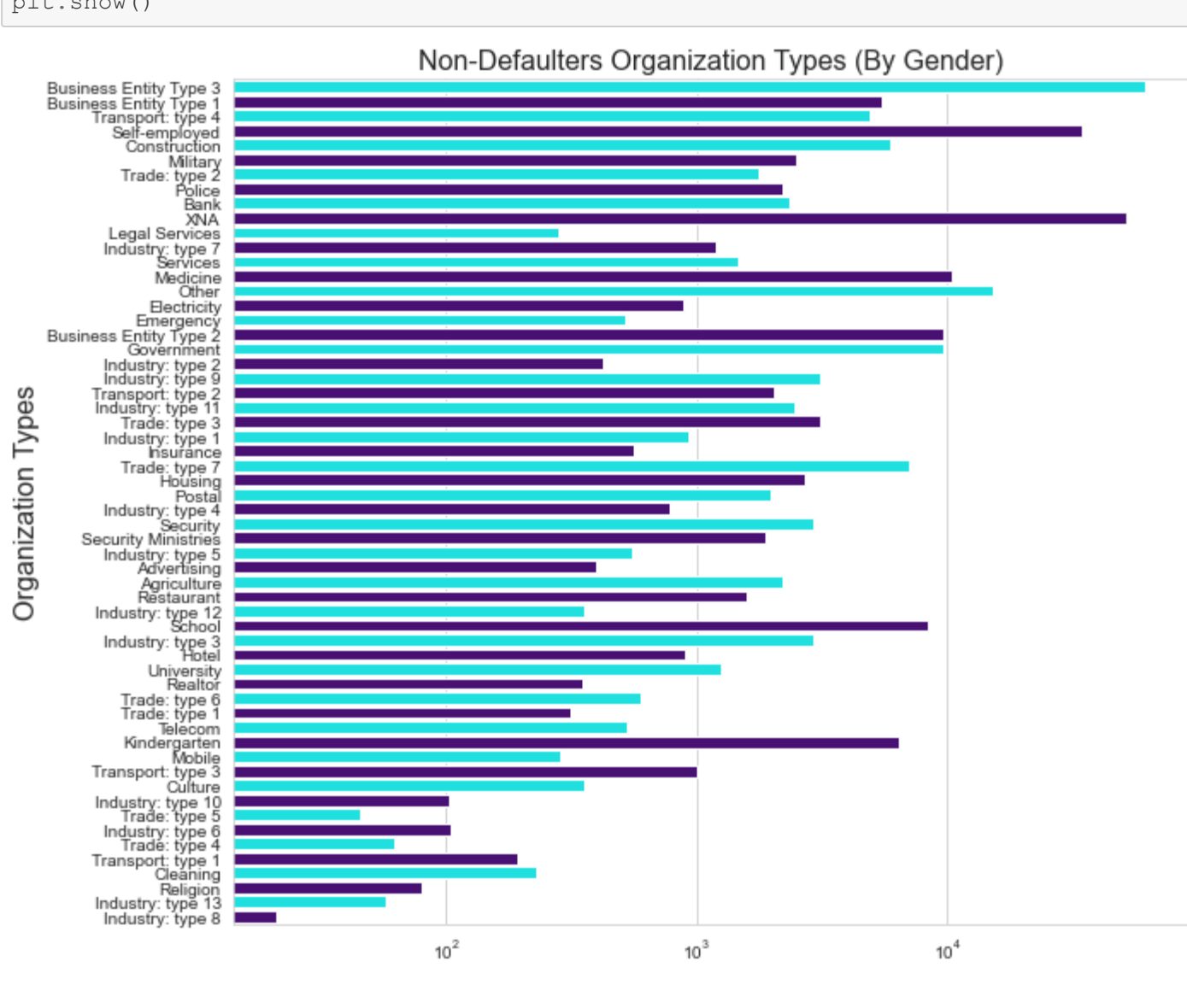
2.7.6 Loan Type vs Target

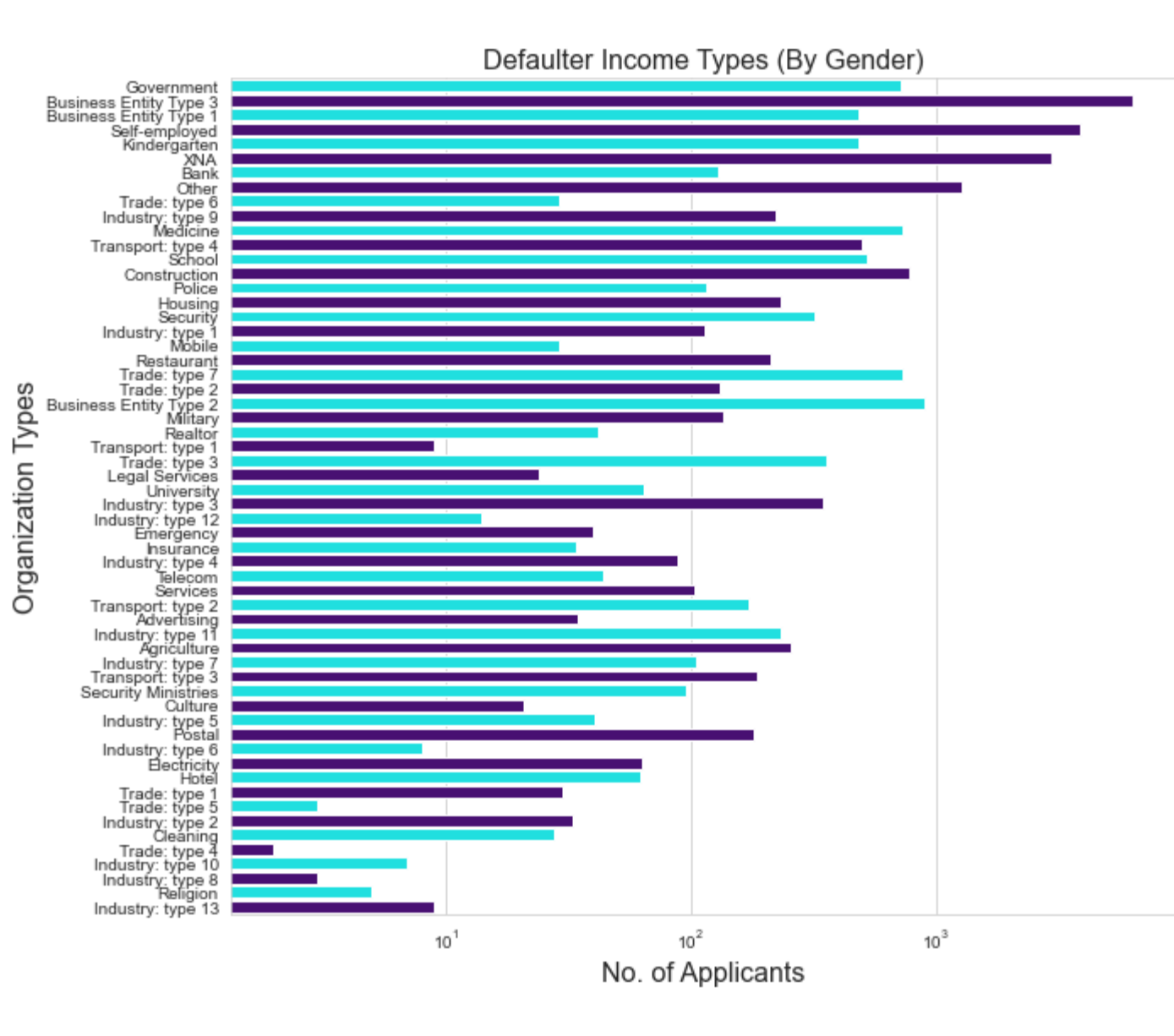
```
In [122]: plt.figure(figsize=[15,1])

          sns.set_style('whitegrid')

          plt.subplot(1,2,1)
          sns.countplot(data=target0,y='NAME_CONTRACT_TYPE', palette=['cyan','Indigo'])\
              .set_title('Non-Defaulters Contract Types', fontsize=14)
          plt.ylabel('Contract Types', fontsize=12)
          plt.xlabel('No. of Applicants',fontsize=12)
          plt.xscale('log')

          plt.subplot(1,2,2)
          sns.countplot(data=target1,y='NAME_CONTRACT_TYPE', palette=['cyan','Indigo'])\
              .set_title('Defaulter Contract Types', fontsize=14)

          plt.ylabel('', fontsize=12)
          plt.xscale('log')
          plt.xlabel('No. of Applicants',fontsize=12)

          plt.show()
```



2.7.7 Organization Type vs Target

```
In [123]: plt.figure(figsize=[10,20])
          sns.set_style('whitegrid')

          plt.subplot(2,1,1)
          sns.countplot(data=target0,y='ORGANIZATION_TYPE', palette=['cyan','Indigo'])\
              .set_title('Non-Defaulters Organization Types (By Gender)', fontsize=16)
          plt.ylabel('Organization Types', fontsize=16)
          plt.xlabel(' ',fontsize=12)
          plt.xscale('log')

          plt.subplot(2,1,2)
          sns.countplot(data=target1,y='ORGANIZATION_TYPE', palette=['cyan','Indigo'])\
              .set_title('Defaulter Income Types (By Gender)', fontsize=16)

          plt.ylabel('Organization Types', fontsize=16)
          plt.xscale('log')

          plt.xlabel('No. of Applicants',fontsize=16)

          plt.show()
```



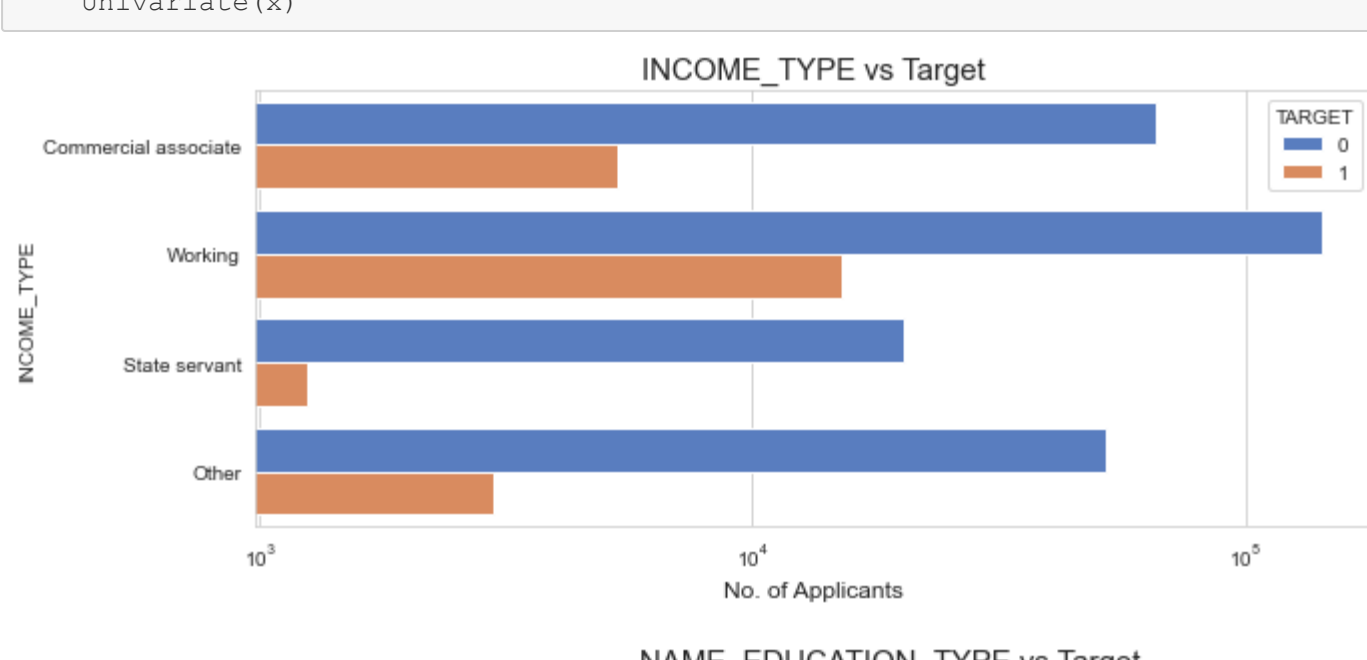


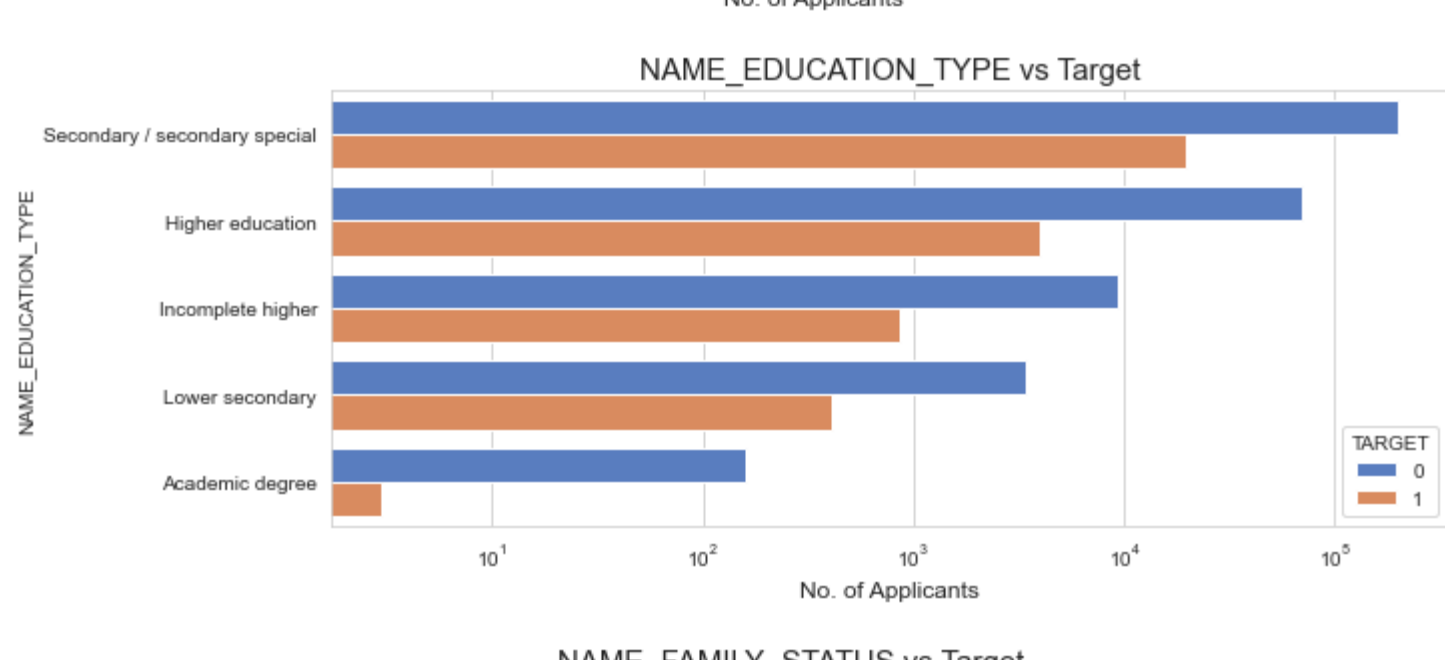
2.7.8 Observations so far:

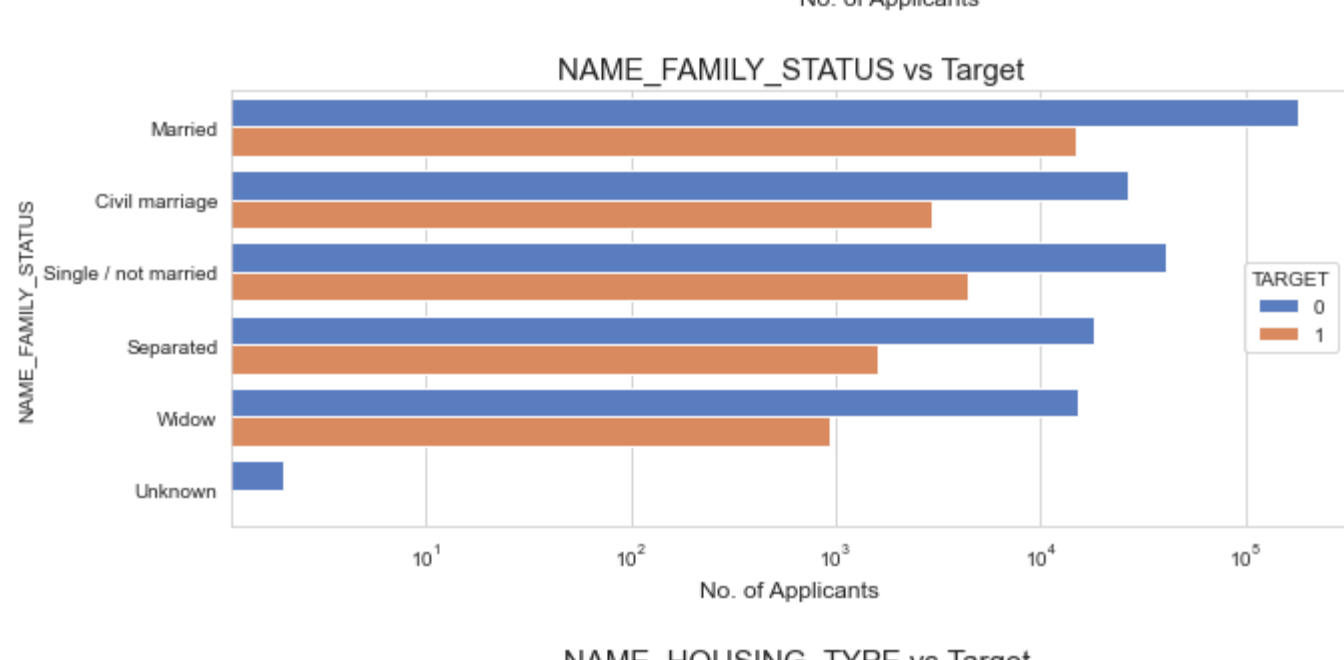
- Interestingy, All important social variables so far seem not to individually effecting on the Target.
- Let's run them for rest of the columns.
- Considering the above observations, it seems like along with the income, there are other factors in play when it comes to the Target. We might require bi-variate or multi-variate analysis

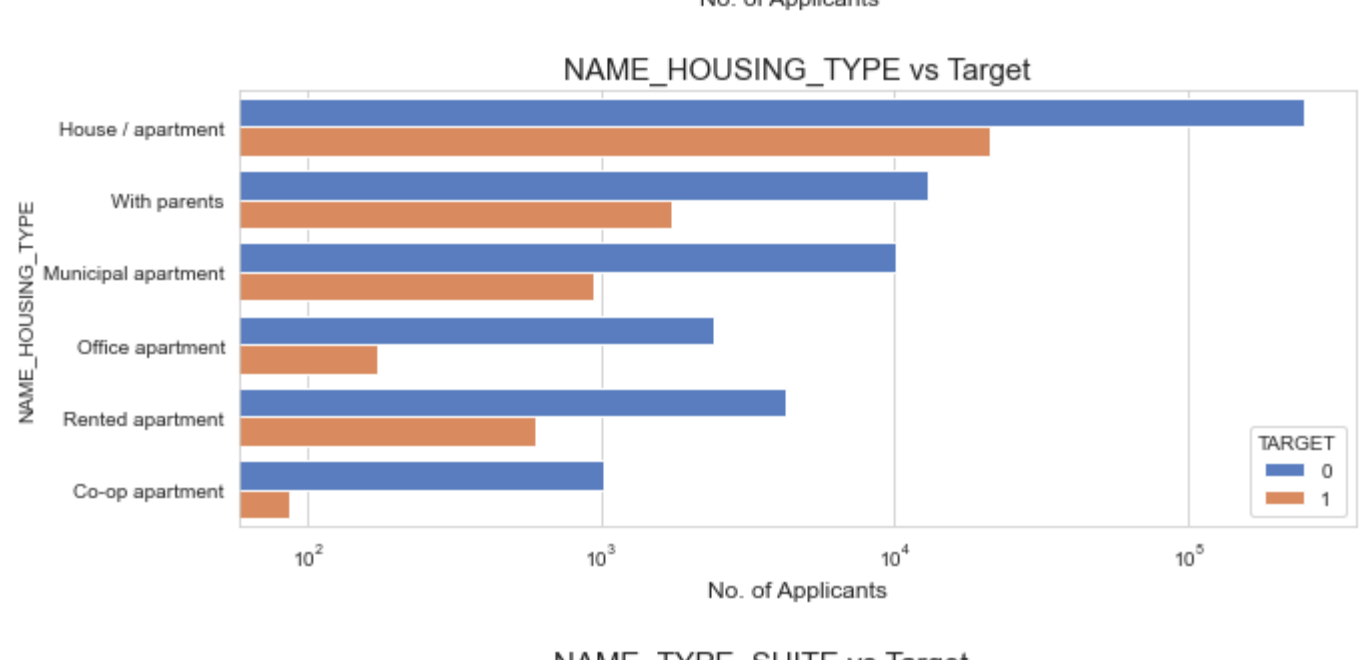
2.7.9 Running Univariate Analysis for all to assert the above

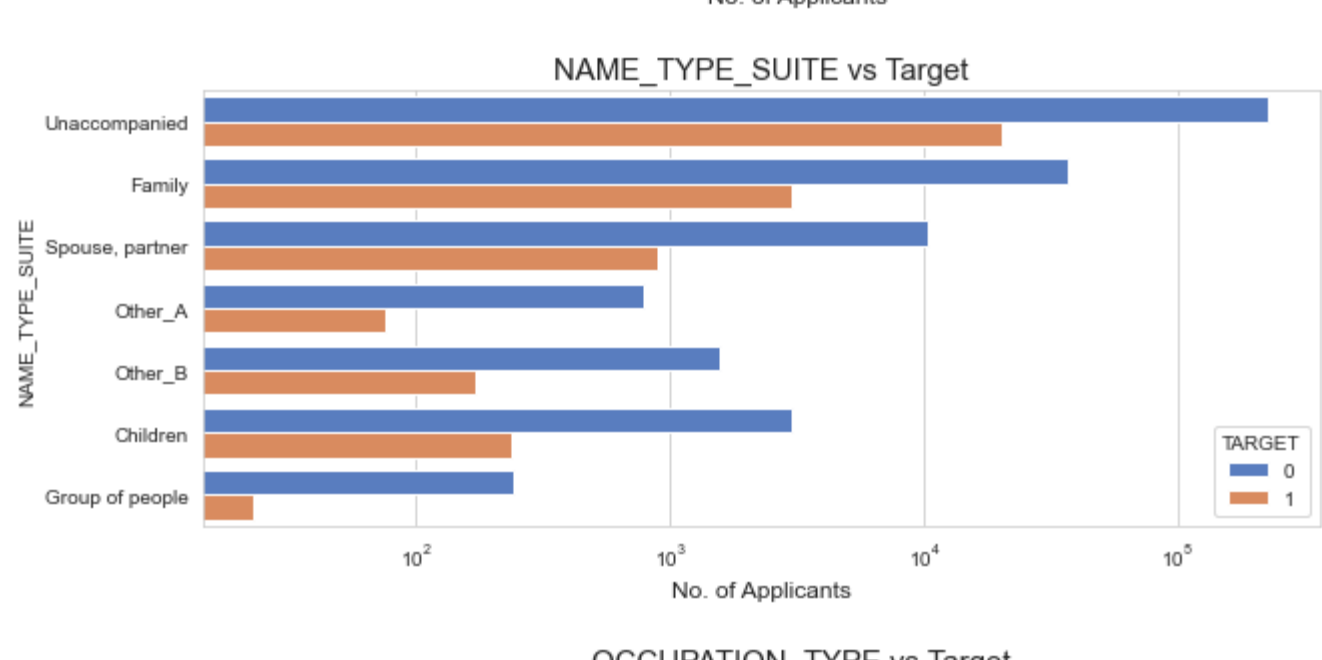
```
In [124]: def Univariate(x):
              plt.figure(figsize=[10,4])
              sns.set_style('whitegrid')

              sns.countplot(data=appl_data, y=x, hue='TARGET', palette='muted')

              plt.title(f'{x} vs Target', fontsize=15)
              plt.xlabel('No. of Applicants', fontsize=12)

              plt.xscale('log')
              plt.show()

In [125]: colns=['INCOME_TYPE','NAME_EDUCATION_TYPE','NAME_FAMILY_STATUS','NAME_HOUSING_TYPE','NAME_TYPE_SUITE',
          'OCCUPATION_TYPE']

In [126]: for x in colns:
              Univariate(x)
```

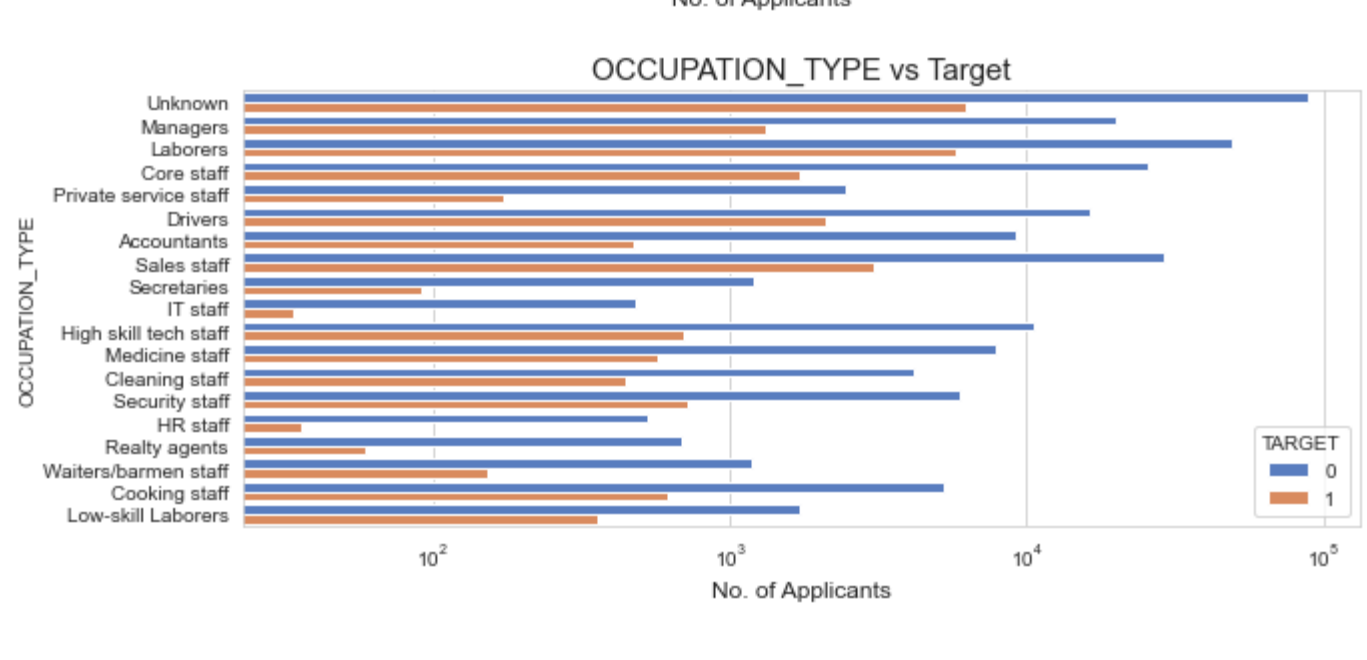












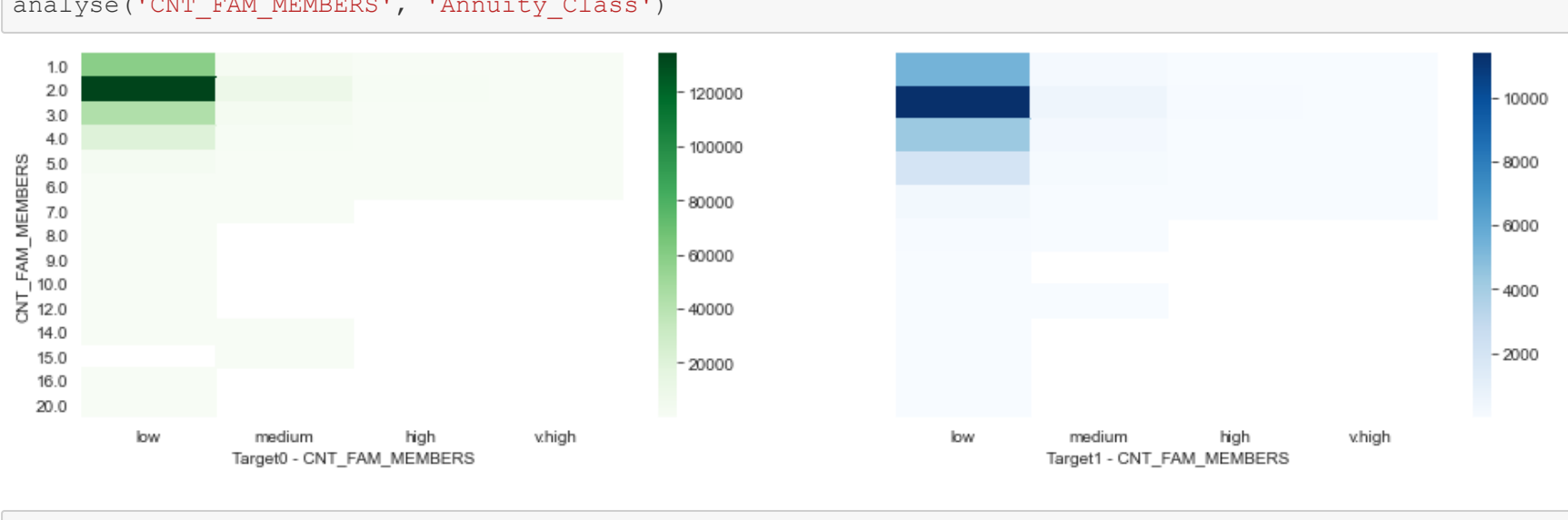
2.7.10 Observations for Univariate Analysis:

- All important social variables are `not individually effecting` on the Target.
- There are other factors in play when it comes to the Target.
- We require `bi-variate or multi-variate analysis`

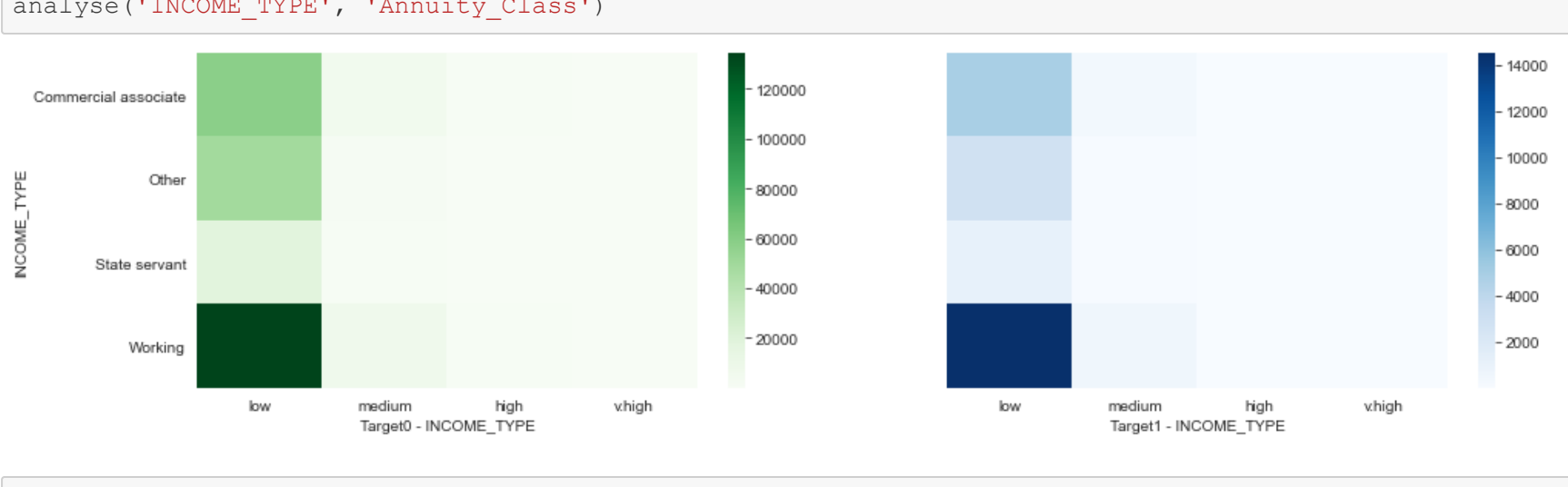
2.8 Multivariate Analysis

2.8.1 Bi-Variate Analysis: Heatmap for different categories for both the targets

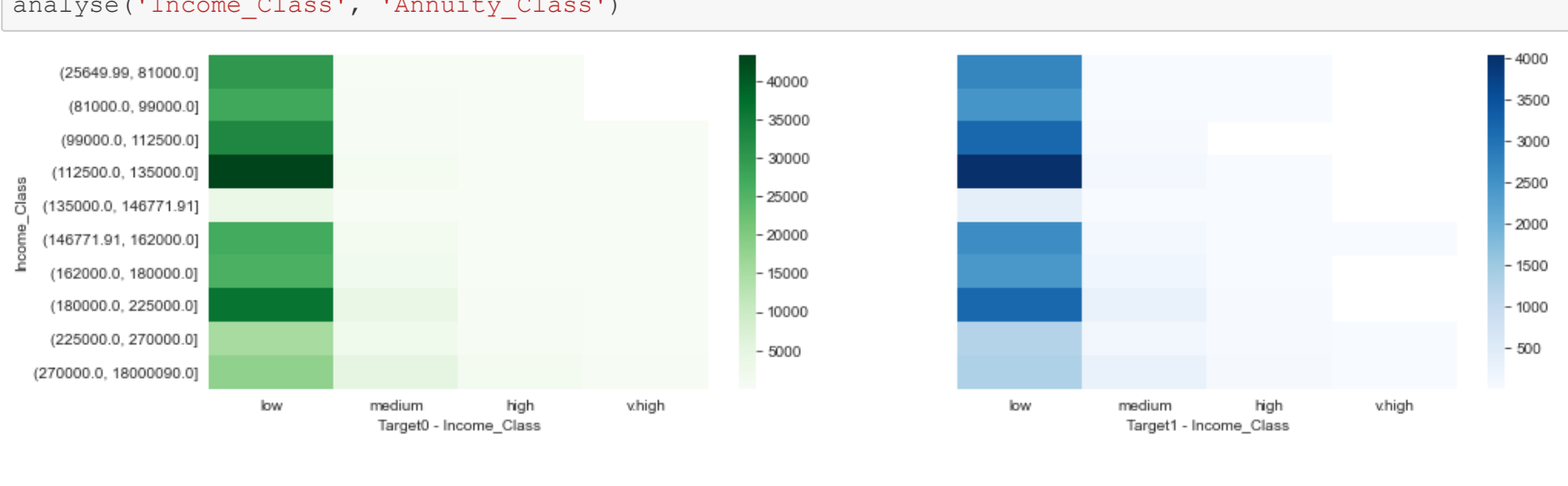
```
In [127]: def analyse(x,y):
              corr0=pd.pivot_table(data=target0,values='TARGET',index=[x],columns=[y],aggfunc='count')
              corr1=pd.pivot_table(data=target1,values='TARGET',index=[x],columns=[y],aggfunc='count')

              plt.figure(figsize=(16,4))
              plt.subplot(1,2,1)

              sns.heatmap(corr0, cmap='Greens')
              plt.xlabel(f'Target0 - {x}')
              ax = plt.subplot(1,2,2)

              sns.heatmap(corr1,cmap='Blues')
              plt.xlabel(f'Target1 - {x}')
              plt.ylabel('')
              ax.set_yticklabels([])

              plt.show()

In [128]: analyse('CNT_FAM_MEMBERS', 'Annuity_Class')
```




```
In [129]: analyse('INCOME_TYPE', 'Annuity_Class')
```




```
In [130]: analyse('Income_Class', 'Annuity_Class')
```



Observations for Multivariate Analysis:

- No significance variations were observed for specific variables.
- This calls for multi-variate analysis
- Cross-referencing with Previous Application data could potentially yield more results

2.8.2 Multivariate Analysis: Correlation

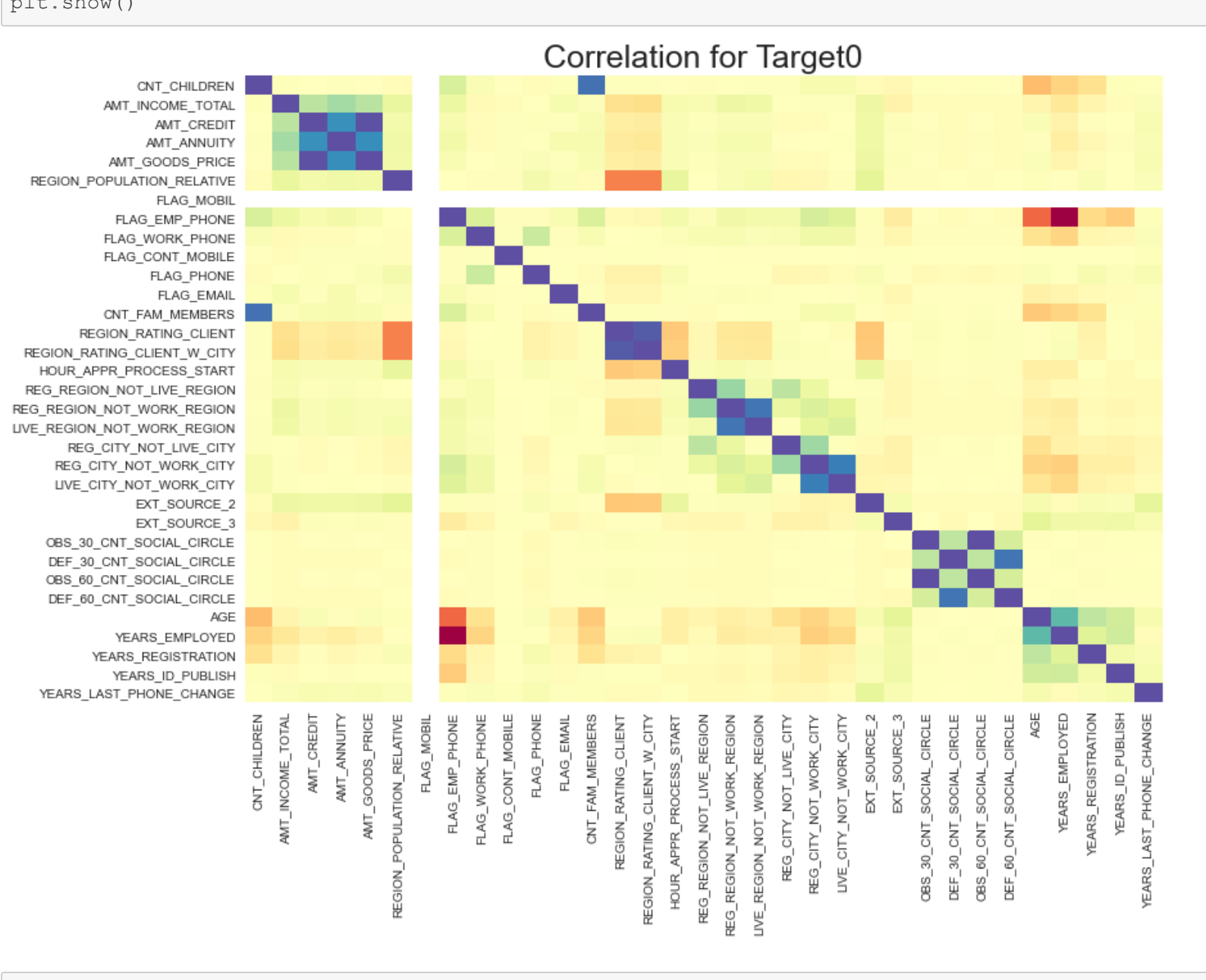
```
In [131]: corr0=target0.iloc[0:,2:].corr()
          heatmap_corr0 = corr0.copy()
          corr0
```

Out[131]:

	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	REGION_P
CNT_CHILDREN	1.000000	0.027992	0.003286	0.021290	-0.000469	
AMT_INCOME_TOTAL	0.027992	1.000000	0.342454	0.418268	0.348698	
AMT_CREDIT	0.003286	0.342454	1.000000	0.770579	0.986965	
AMT_ANNUITY	0.021290	0.418268	0.770579	1.000000	0.775859	
AMT_GOODS_PRICE	-0.000469	0.348698	0.986965	0.775859	1.000000	
REGION_POPULATION_RELATIVE	-0.024171	0.166220	0.100946	0.121195	0.104015	
FLAG_MOBIL	NaN	NaN	NaN	NaN	NaN	
FLAG_EMP_PHONE	0.244188	0.141555	0.071760	0.106431	0.069835	
FLAG_WORK_PHONE	0.055332	-0.031791	-0.017799	-0.023008	0.002785	
FLAG_CONT_MOBILE	-0.001356	-0.018535	0.023486	0.021946	0.020566	
FLAG_PHONE	-0.030056	0.003262	0.026145	0.011680	0.040461	
FLAG_EMAIL	0.023569	0.084291	0.016893	0.071412	0.017007	
CNT_FAM_MEMBERS	0.878663	0.034615	0.064690	0.076095	0.062849	
REGION_RATING_CLIENT	0.022762	-0.185675	-0.103611	-0.132253	-0.104381	
REGION_RATING_CLIENT_W_CITY	0.021781	-0.199703	-0.112540	-0.145348	-0.112652	
HOUR_APPR_PROCESS_START	-0.005048	0.076220	0.053721	0.053534	0.062511	
REG_REGION_NOT_LIVE_REGION	-0.012406	0.068456	0.024573	0.041898	0.026722	
REG_REGION_NOT_WORK_REGION	0.010630	0.137093	0.053998	0.080882	0.055060	
LIVE_REGION_NOT_WORK_REGION	0.017119	0.127537	0.054602	0.075786	0.054560	
REG_CITY_NOT_LIVE_CITY	0.021562	0.010914	-0.025049	-0.005708	-0.024934	
REG_CITY_NOT_WORK_CITY	0.072138	0.018519	-0.015552	0.001765	-0.016670	
LIVE_CITY_NOT_WORK_CITY	0.071027	0.021577	0.002720	0.010887	0.001428	
EXT_SOURCE_2	-0.015585	0.139360	0.129341	0.126729	0.135676	
EXT_SOURCE_3	-0.041566	-0.073621	0.035939	0.027746	0.038875	
OBS_30_CNT_SOCIAL_CIRCLE	0.014528	-0.027929	-0.001014	-0.013176	-0.000700	
DEF_30_CNT_SOCIAL_CIRCLE	-0.002158	-0.027620	-0.019927	-0.022593	-0.021032	
OBS_60_CNT_SOCIAL_CIRCLE	0.014203	-0.027793	-0.000992	-0.012981	-0.000717	
DEF_60_CNT_SOCIAL_CIRCLE	-0.002126	-0.027619	-0.022320	-0.023447	-0.023216	
AGE	-0.337254	-0.063089	0.047306	-0.012606	0.045068	
YEARS_EMPLOYED	-0.245092	-0.141203	-0.070606	-0.105764	-0.068675	
YEARS_REGISTRATION	-0.185701	-0.065049	-0.013241	-0.039315	-0.015546	
YEARS_ID_PUBLISH	0.028661	-0.023179	0.001063	-0.014466	0.003439	
YEARS_LAST_PHONE_CHANGE	0.008425	0.042493	0.068400	0.062069	0.070325	

33 rows × 33 columns

```
In [132]: plt.figure(figsize=[16,9])
          sns.heatmap(heatmap_corr0, cmap='Spectral')
          plt.title("Correlation for Target0",fontsize=25)
          plt.show()
```



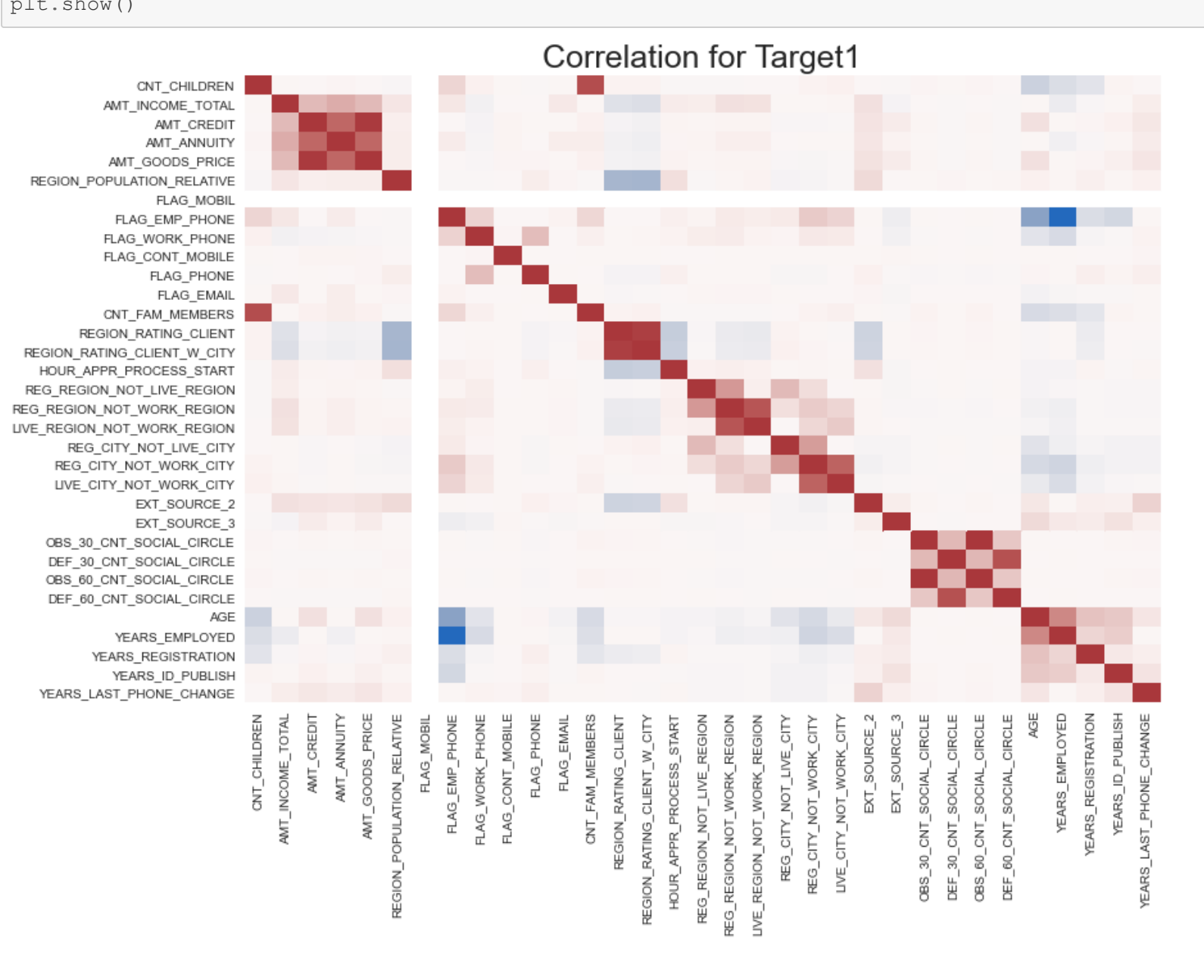

```
In [133]: corr1=target1.iloc[0:,2:].corr()
          heatmap_corr1 = corr1.copy()
          corr1
```

Out[133]:

	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	REGION_P
CNT_CHILDREN	1.000000	0.002735	-0.002009	0.030826	-0.008419	
AMT_INCOME_TOTAL	0.002735	1.000000	0.324956	0.397928	0.327434	
AMT_CREDIT	-0.002009	0.324956	1.000000	0.751707	0.982846	
AMT_ANNUITY	0.030826	0.397928	0.751707	1.000000	0.752511	
AMT_GOODS_PRICE	-0.008419	0.327434	0.982846	0.752511	1.000000	
REGION_POPULATION_RELATIVE	-0.032140	0.117307	0.069764	0.072334	0.076864	
FLAG_MOBIL	NaN	NaN	NaN	NaN	NaN	
FLAG_EMP_PHONE	0.192442	0.107672	-0.001221	0.081315	-0.005860	
FLAG_WORK_PHONE	0.051446	-0.074123	-0.060478	-0.050918	-0.035903	
FLAG_CONT_MOBILE	0.003596	-0.009768	0.031903	0.035833	0.028688	
FLAG_PHONE	-0.024280	-0.008116	0.007914	-0.014653	0.025731	
FLAG_EMAIL	0.014160	0.097457	0.010016	0.078180	0.013872	
CNT_FAM_MEMBERS	0.885495	0.010453	0.050958	0.075375	0.047018	
REGION_RATING_CLIENT	0.041276	-0.167664	-0.059293	-0.074052	-0.066285	
REGION_RATING_CLIENT_W_CITY	0.043741	-0.180752	-0.071507	-0.089605	-0.077143	
HOUR_APPR_PROCESS_START	-0.023846	0.081275	0.031536	0.031004	0.044038	
REG_REGION_NOT_LIVE_REGION	-0.024031	0.071840	0.019550	0.034846	0.022568	
REG_REGION_NOT_WORK_REGION	-0.020414	0.137324	0.033232	0.066662	0.036572	
LIVE_REGION_NOT_WORK_REGION	-0.011732	0.126213	0.033644	0.064363	0.035848	
REG_CITY_NOT_LIVE_CITY	-0.000639	0.000537	-0.033239	-0.005847	-0.033944	
REG_CITY_NOT_WORK_CITY	0.046257	0.009430	-0.038317	0.001438	-0.039609	
LIVE_CITY_NOT_WORK_CITY	0.053372	0.017115	-0.017052	0.009300	-0.017516	
EXT_SOURCE_2	-0.012235	0.136898	0.121132	0.116656	0.130772	
EXT_SOURCE_3	-0.020244	-0.061981	0.078405	0.042357	0.079518	
OBS_30_CNT_SOCIAL_CIRCLE	0.025718	-0.006558	0.019218	0.004447	0.019872	
DEF_30_CNT_SOCIAL_CIRCLE	0.001647	-0.024705	-0.026280	-0.022717	-0.022243	
OBS_60_CNT_SOCIAL_CIRCLE	0.025091	-0.005862	0.019606	0.005479	0.020437	
DEF_60_CNT_SOCIAL_CIRCLE	-0.004703	-0.024381	-0.031081	-0.027798	-0.026598	
AGE	-0.258924	0.002472	0.135758	0.014412	0.135844	
YEARS_EMPLOYED	-0.192836	-0.107321	0.002492	-0.080725	0.007230	
YEARS_REGISTRATION	-0.149065	-0.038559	0.026374	-0.033585	0.026028	
YEARS_ID_PUBLISH	0.033540	0.003478	0.052156	0.016262	0.055779	
YEARS_LAST_PHONE_CHANGE	-0.012451	0.076472	0.108875	0.079614	0.115704	

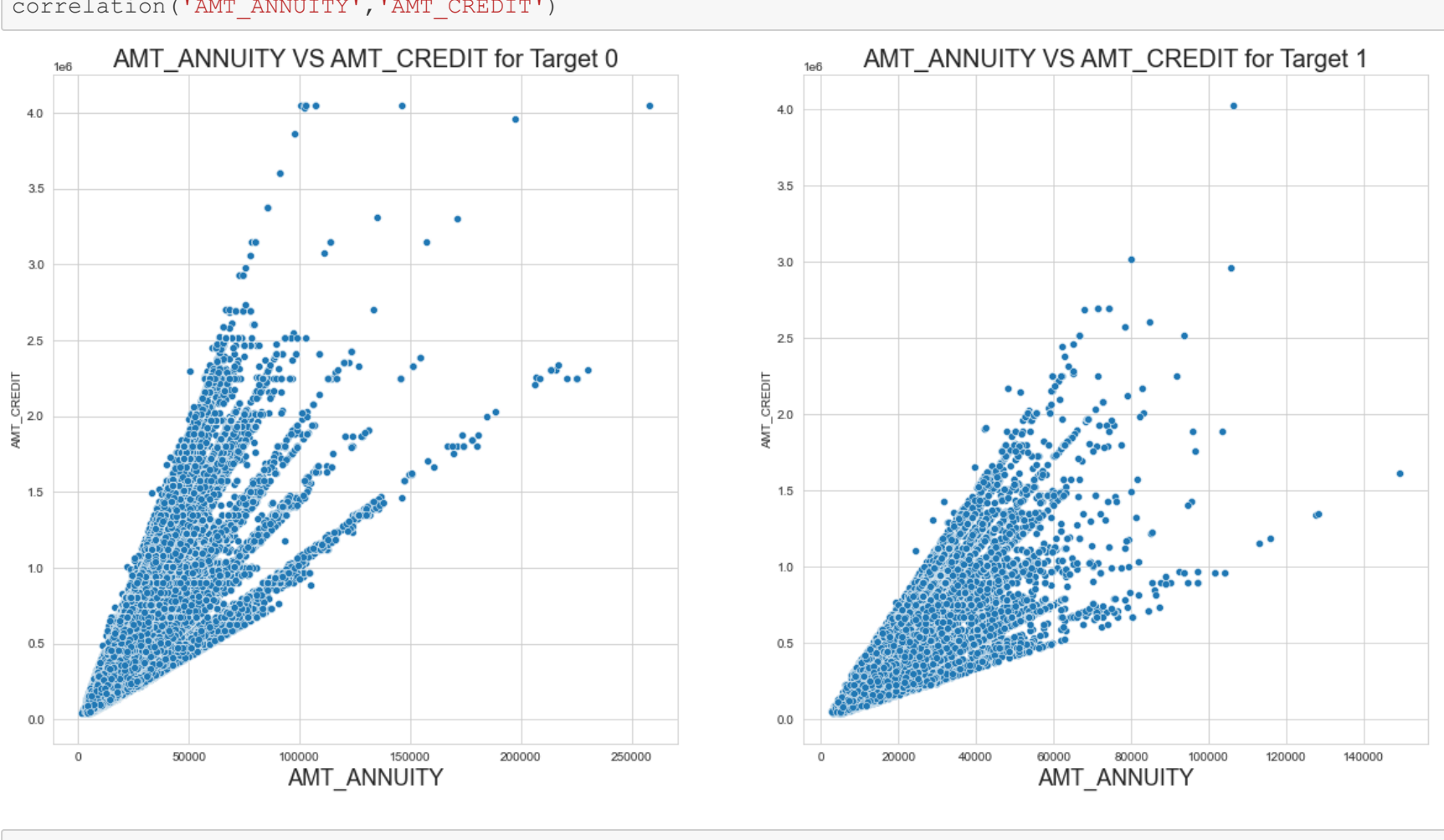
33 rows × 33 columns

```
In [134]: plt.figure(figsize=[16,9])
          sns.heatmap(heatmap_corr1, cmap='vlag')
          plt.title("Correlation for Target1",fontsize=25)
          plt.show()
```

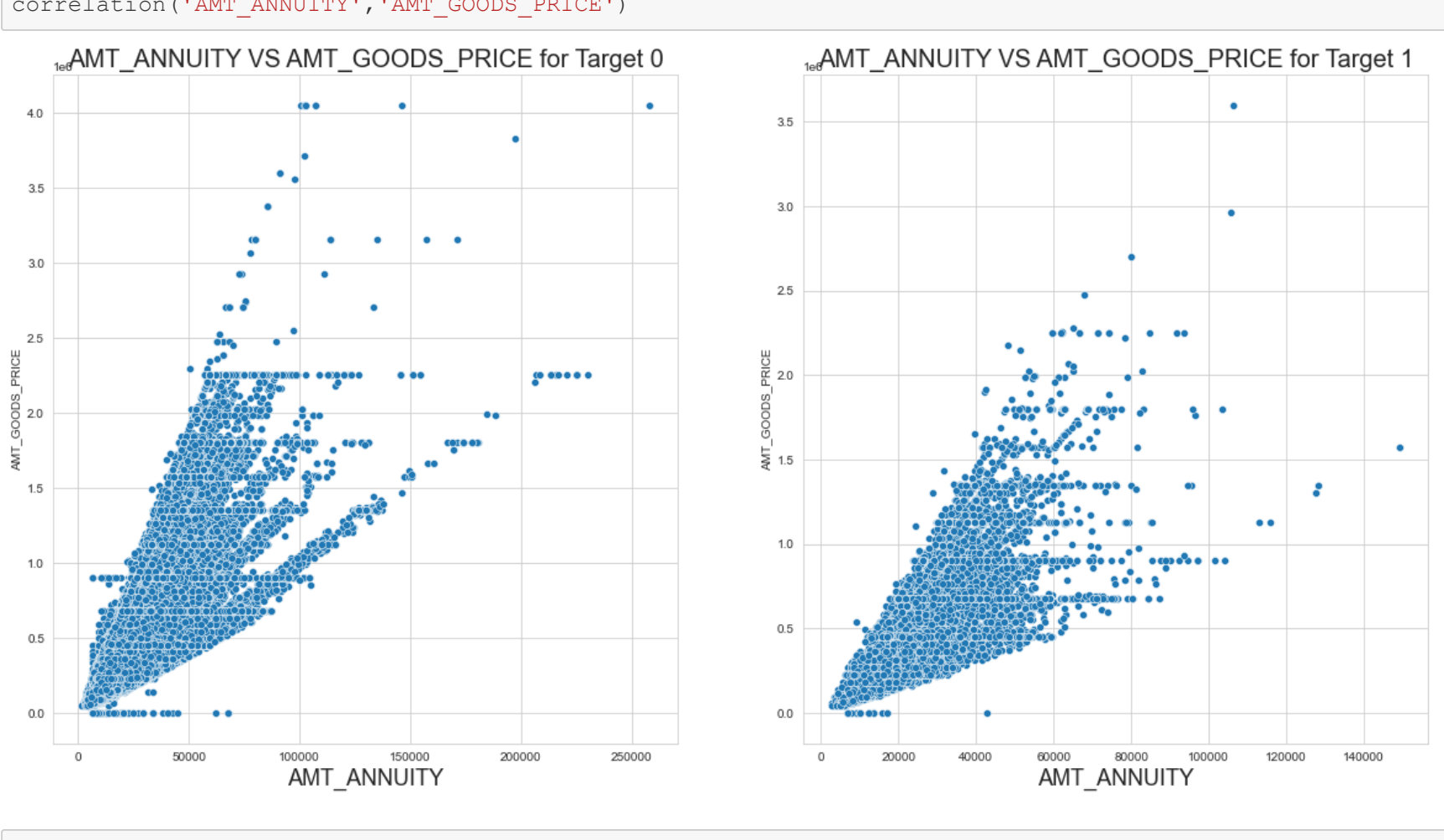


In [135]:
```python
## correlation between columns by observations
def correlation(x,y):
    plt.figure(figsize=[20,10])
    ax = plt.subplot(1,2,1)
    sns.scatterplot(target0[x],target0[y])
    plt.title(f'{x} VS {y} for Target 0',fontsize=20)

    plt.xlabel(f'{x}')
    ax.xaxis.get_label().set_fontsize(18)
    plt.ylabel(f'{y}')

    ax = plt.subplot(1,2,2)
    sns.scatterplot(target1[x],target1[y])
    plt.title(f'{x} VS {y} for Target 1',fontsize=20)

    plt.xlabel(f'{x}')
    ax.xaxis.get_label().set_fontsize(18)
    plt.ylabel(f'{y}')
    plt.show()
```

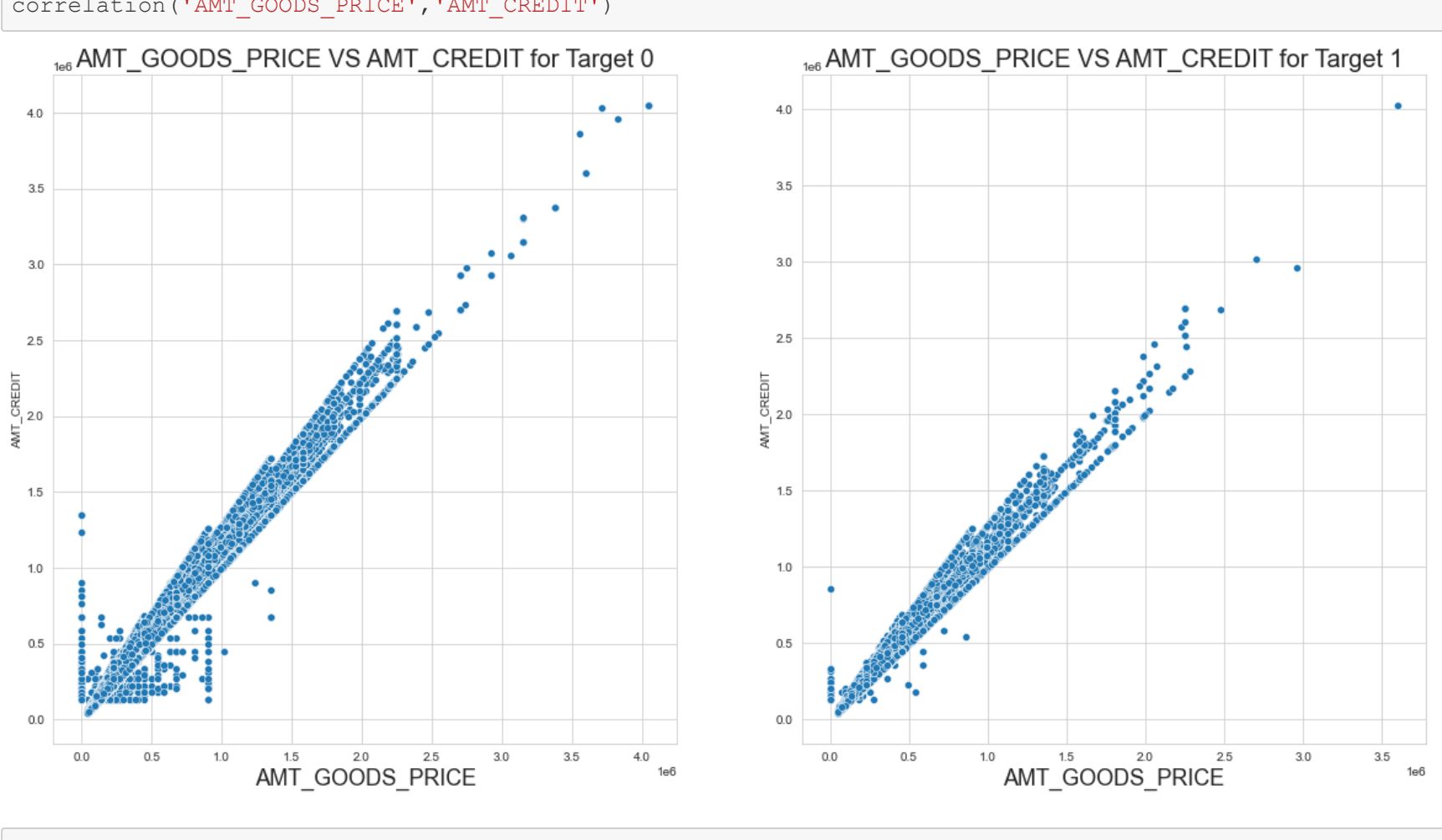
In [136]:
```python
correlation('AMT_ANNUITY','AMT_CREDIT')
```



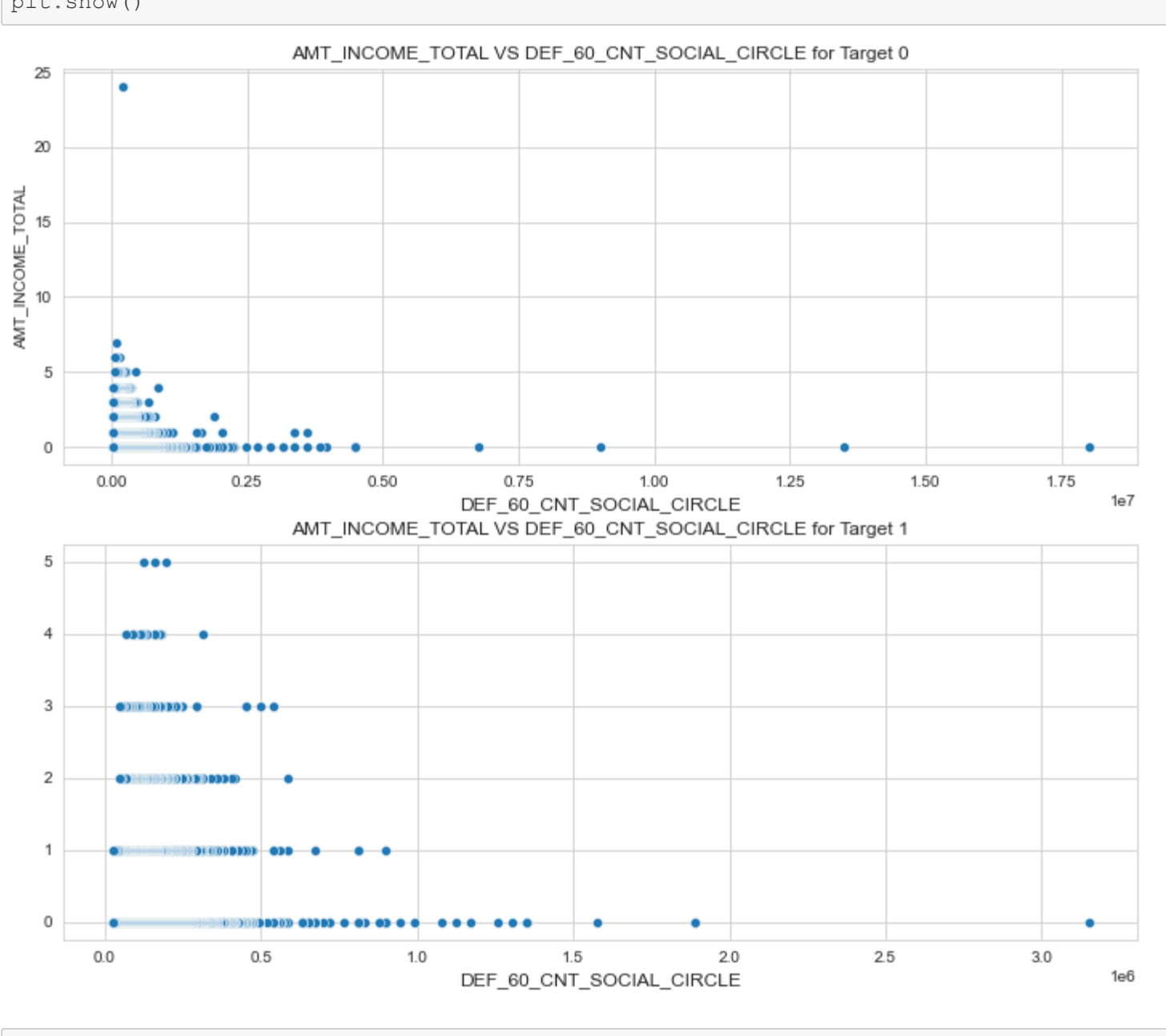
In [137]:
```python
correlation('AMT_ANNUITY','AMT_GOODS_PRICE')
```



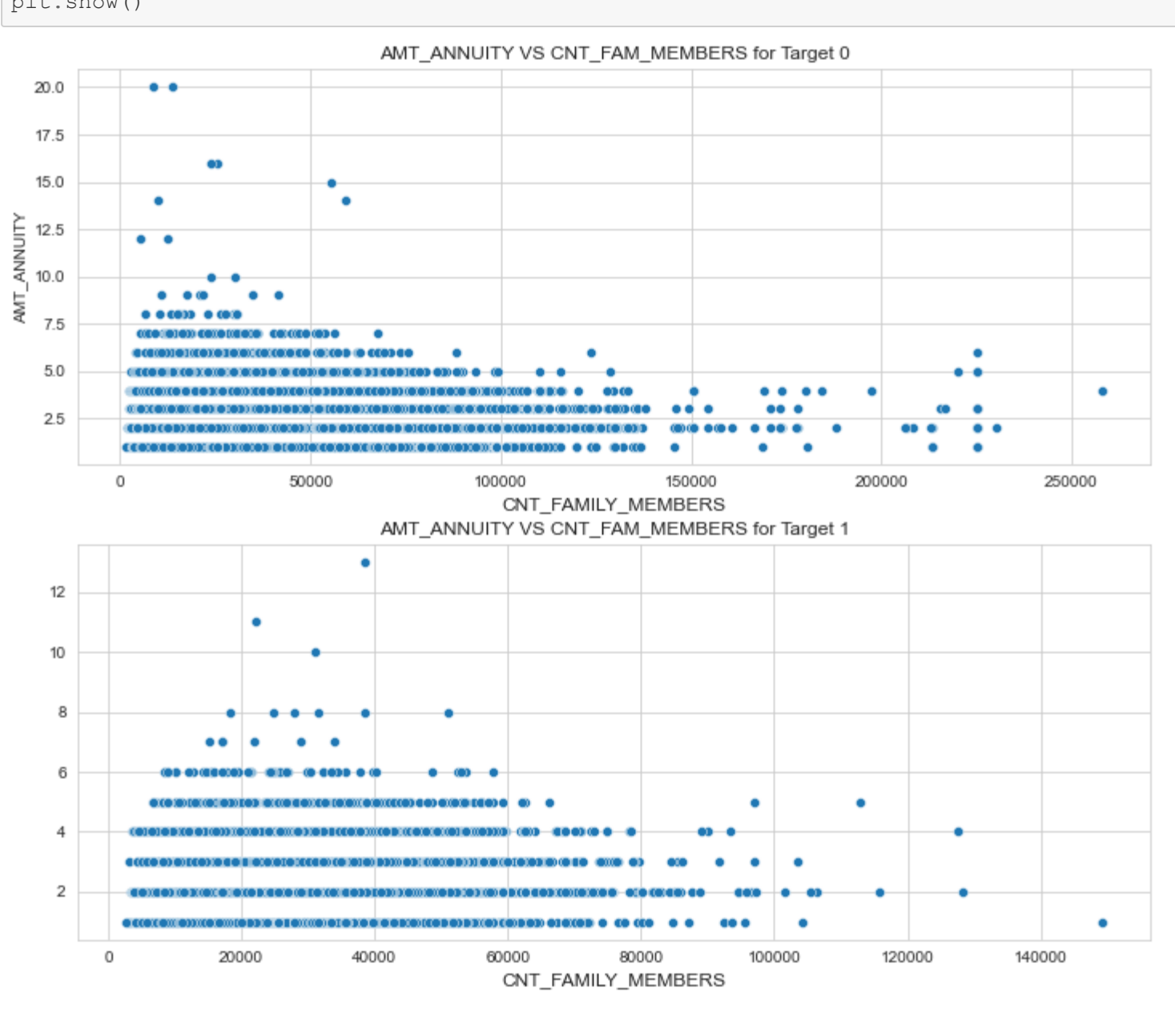
In [138]:
```python
correlation('AMT_GOODS_PRICE','AMT_CREDIT')
```



In [139]:
```python
plt.figure(figsize=[12,10])
ax = plt.subplot(2,1,1)
sns.scatterplot(target0.AMT_INCOME_TOTAL,target0.DEF_60_CNT_SOCIAL_CIRCLE)
plt.title('AMT_INCOME_TOTAL VS DEF_60_CNT_SOCIAL_CIRCLE for Target 0',fontsize=12)

plt.xlabel('DEF_60_CNT_SOCIAL_CIRCLE')
ax.xaxis.get_label().set_fontsize(12)
plt.ylabel('AMT_INCOME_TOTAL')

ax = plt.subplot(2,1,2)
sns.scatterplot(target1.AMT_INCOME_TOTAL,target1.DEF_60_CNT_SOCIAL_CIRCLE)
plt.title('AMT_INCOME_TOTAL VS DEF_60_CNT_SOCIAL_CIRCLE for Target 1',fontsize=12)

plt.xlabel('DEF_60_CNT_SOCIAL_CIRCLE')
ax.xaxis.get_label().set_fontsize(12)
plt.ylabel('')
plt.show()
```



In [140]:
```python
plt.figure(figsize=[12,10])
ax = plt.subplot(2,1,1)
sns.scatterplot(target0.AMT_ANNUITY,target0.CNT_FAM_MEMBERS)
plt.title('AMT_ANNUITY VS CNT_FAM_MEMBERS for Target 0',fontsize=12)

plt.xlabel('CNT_FAMILY_MEMBERS')
ax.xaxis.get_label().set_fontsize(12)
plt.ylabel('AMT_ANNUITY')

ax = plt.subplot(2,1,2)
sns.scatterplot(target1.AMT_ANNUITY,target1.CNT_FAM_MEMBERS)
plt.title('AMT_ANNUITY VS CNT_FAM_MEMBERS for Target 1',fontsize=12)

plt.xlabel('CNT_FAMILY_MEMBERS')
ax.xaxis.get_label().set_fontsize(12)
plt.ylabel('')
plt.show()
```



Finding top10 correlations for target0 and target1

In [141]:
```python
target_corr0=corr0=corr0.abs().unstack().sort_values().dropna()
target_corr0=target_corr0[target_corr0!=1.0]
target_corr0
```

Out[141]:
```
FLAG_PHONE                 EXT_SOURCE_3                  0.000002
EXT_SOURCE_3               FLAG_PHONE                    0.000002
YEARS_LAST_PHONE_CHANGE    DEF_60_CNT_SOCIAL_CIRCLE      0.000032
DEF_60_CNT_SOCIAL_CIRCLE   YEARS_LAST_PHONE_CHANGE       0.000032
REG_CITY_NOT_WORK_CITY     DEF_30_CNT_SOCIAL_CIRCLE      0.000386
                                                           ...
AMT_CREDIT                 AMT_GOODS_PRICE               0.986965
OBS_60_CNT_SOCIAL_CIRCLE   OBS_30_CNT_SOCIAL_CIRCLE      0.998511
OBS_30_CNT_SOCIAL_CIRCLE   OBS_60_CNT_SOCIAL_CIRCLE      0.998511
FLAG_EMP_PHONE             YEARS_EMPLOYED                0.999755
YEARS_EMPLOYED             FLAG_EMP_PHONE                0.999755
Length: 992, dtype: float64
```

In [142]:
```python
# Top 10 correlation for target0
## As observed we can see our estimated variables are under top10 correlated variables
target_corr0.tail(10)
```

Out[142]:
```
CNT_FAM_MEMBERS             CNT_CHILDREN                  0.878663
CNT_CHILDREN                CNT_FAM_MEMBERS               0.878663
REGION_RATING_CLIENT        REGION_RATING_CLIENT_W_CITY   0.949976
REGION_RATING_CLIENT_W_CITY REGION_RATING_CLIENT          0.949976
AMT_GOODS_PRICE             AMT_CREDIT                    0.986965
AMT_CREDIT                  AMT_GOODS_PRICE               0.986965
OBS_60_CNT_SOCIAL_CIRCLE    OBS_30_CNT_SOCIAL_CIRCLE      0.998511
OBS_30_CNT_SOCIAL_CIRCLE    OBS_60_CNT_SOCIAL_CIRCLE      0.998511
FLAG_EMP_PHONE              YEARS_EMPLOYED                0.999755
YEARS_EMPLOYED              FLAG_EMP_PHONE                0.999755
dtype: float64
```

In [143]:
```python
target_corr1=corr1=corr1.abs().unstack().sort_values().dropna()
target_corr1=target_corr1[target_corr1!=1.0]
target_corr1
```

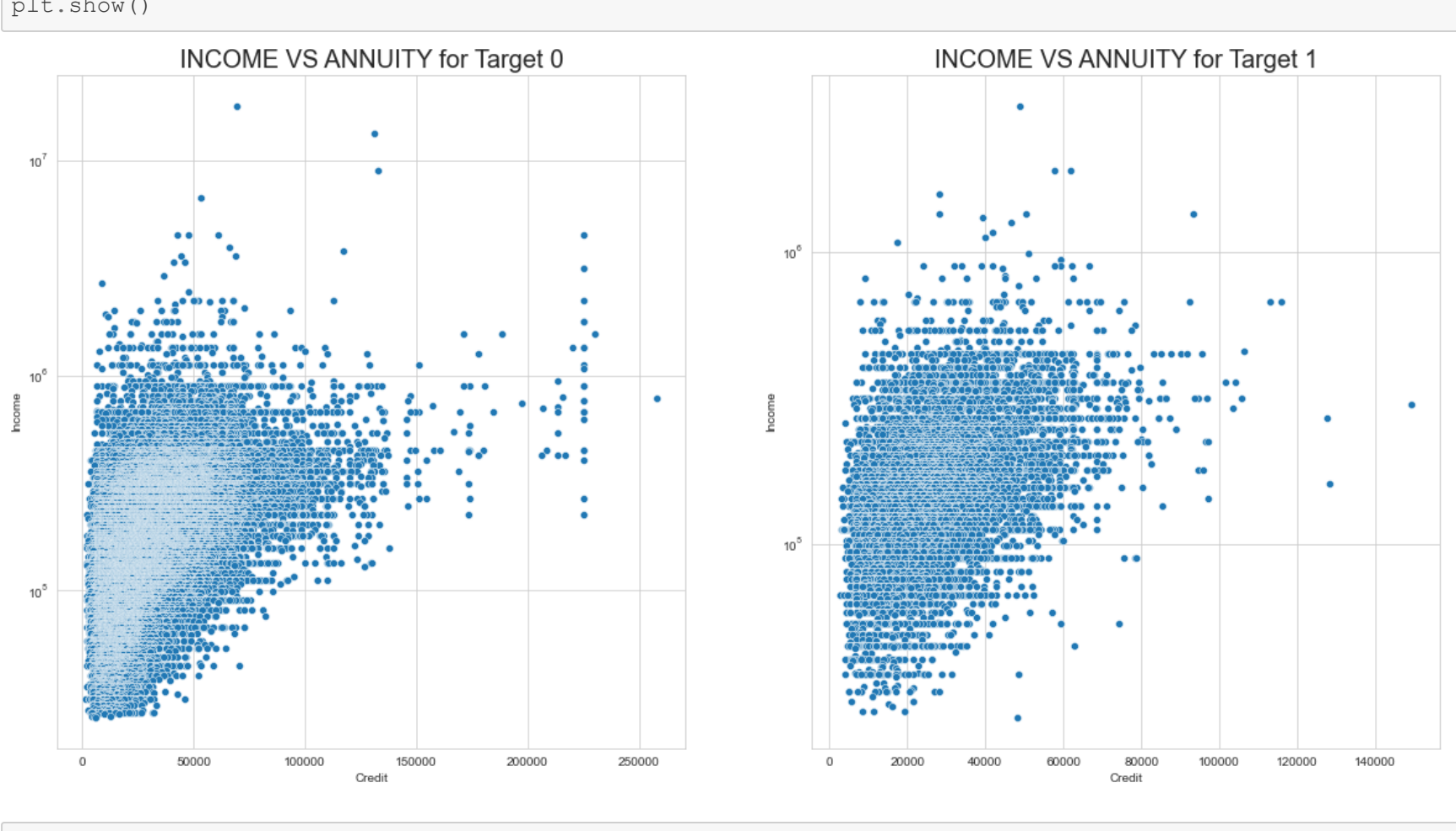
Out[143]:
```
REGION_RATING_CLIENT_W_CITY YEARS_ID_PUBLISH              0.000105
YEARS_ID_PUBLISH            REGION_RATING_CLIENT_W_CITY   0.000105
DEF_30_CNT_SOCIAL_CIRCLE    YEARS_LAST_PHONE_CHANGE       0.000213
YEARS_LAST_PHONE_CHANGE     DEF_30_CNT_SOCIAL_CIRCLE      0.000213
FLAG_EMP_PHONE              EXT_SOURCE_2                  0.000331
                                                            ...
AMT_GOODS_PRICE             AMT_CREDIT                    0.982846
OBS_30_CNT_SOCIAL_CIRCLE    OBS_60_CNT_SOCIAL_CIRCLE      0.998272
OBS_60_CNT_SOCIAL_CIRCLE    OBS_30_CNT_SOCIAL_CIRCLE      0.998272
YEARS_EMPLOYED              FLAG_EMP_PHONE                0.999704
FLAG_EMP_PHONE              YEARS_EMPLOYED                0.999704
Length: 992, dtype: float64
```

In [144]:
```python
# Top 10 correlation for target1
## As observed we can see our estimated variables are under top10 correlated variables
target_corr1.tail(10)
```

Out[144]:
```
CNT_CHILDREN                CNT_FAM_MEMBERS               0.885495
CNT_FAM_MEMBERS             CNT_CHILDREN                  0.885495
REGION_RATING_CLIENT_W_CITY REGION_RATING_CLIENT          0.956560
REGION_RATING_CLIENT        REGION_RATING_CLIENT_W_CITY   0.956560
AMT_CREDIT                  AMT_GOODS_PRICE               0.982846
AMT_GOODS_PRICE             AMT_CREDIT                    0.982846
OBS_30_CNT_SOCIAL_CIRCLE    OBS_60_CNT_SOCIAL_CIRCLE      0.998272
OBS_60_CNT_SOCIAL_CIRCLE    OBS_30_CNT_SOCIAL_CIRCLE      0.998272
YEARS_EMPLOYED              FLAG_EMP_PHONE                0.999704
FLAG_EMP_PHONE              YEARS_EMPLOYED                0.999704
dtype: float64
```

Bivariate Analysis

In [145]:
```python
plt.figure(figsize=[20,10])
plt.subplot(1,2,1)
sns.scatterplot(target0.AMT_ANNUITY,target0.AMT_INCOME_TOTAL)
plt.title('INCOME VS ANNUITY for Target 0',fontsize=20)
plt.yscale('log')
plt.xlabel('Credit')
plt.ylabel('Income')

plt.subplot(1,2,2)
sns.scatterplot(target1.AMT_ANNUITY,target1.AMT_INCOME_TOTAL)
plt.title('INCOME VS ANNUITY for Target 1',fontsize=20)
plt.yscale('log')
plt.xlabel('Credit')
plt.ylabel('Income')
plt.show()
```



In [146]:
```python
plt.figure(figsize=[20,10])
ax = plt.subplot(1,2,1)
sns.scatterplot(target0.AMT_CREDIT,target0.AMT_GOODS_PRICE)
plt.title('GOODS PRICE VS CREDIT for Target 0',fontsize=20)

plt.xlabel('Credit')
ax.xaxis.get_label().set_fontsize(18)
plt.ylabel('Goods Price')

ax = plt.subplot(1,2,2)
sns.scatterplot(target1.AMT_CREDIT,target1.AMT_GOODS_PRICE)
plt.title('GOODS_PRICE VS CREDIT for Target 1',fontsize=20)

plt.xlabel('Credit')
ax.xaxis.get_label().set_fontsize(18)
plt.ylabel('Goods Price')
plt.show()
```

Goods_price and Credit are highly correlated

Previous application data

In [147]:
```python
prev_data.head()
```

Out[147]:

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

5 rows × 37 columns

In [148]:
```python
pd.set_option('display.max_rows', prev_data.shape[1])
prev_data.describe().T
```

Out[148]:

	count	mean	std	min	25%	50%	75%	
SK_ID_PREV	1670214.0	1.923089e+06	532597.958696	1.000001e+06	1.461857e+06	1.923110e+06	2.384280e+06	2
SK_ID_CURR	1670214.0	2.783572e+05	102814.823849	1.000010e+05	1.893290e+05	2.787145e+05	3.675140e+05	
AMT_ANNUITY	1297979.0	1.595512e+04	14782.137335	0.000000e+00	6.321780e+03	1.125000e+04	2.065842e+04	
AMT_APPLICATION	1670214.0	1.752339e+05	292779.762387	0.000000e+00	1.872000e+04	7.104600e+04	1.803600e+05	6
AMT_CREDIT	1670213.0	1.961140e+05	318574.616546	0.000000e+00	2.416050e+04	8.054100e+04	2.164185e+05	6
AMT_DOWN_PAYMENT	774370.0	6.697402e+03	20921.495410	-9.000000e-01	0.000000e+00	1.638000e+03	7.740000e+03	3
AMT_GOODS_PRICE	1284699.0	2.278473e+05	315396.557937	0.000000e+00	5.084100e+04	1.123200e+05	2.340000e+05	6
HOUR_APPR_PROCESS_START	1670214.0	1.248418e+01	3.334028	0.000000e+00	1.000000e+01	1.200000e+01	1.500000e+01	
NFLAG_LAST_APPL_IN_DAY	1670214.0	9.964675e-01	0.059330	0.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	
RATE_DOWN_PAYMENT	774370.0	7.963682e-02	0.107823	-1.497876e-05	0.000000e+00	5.160508e-02	1.089091e-01	
RATE_INTEREST_PRIMARY	5951.0	1.883569e-01	0.087671	3.478125e-02	1.607163e-01	1.891222e-01	1.933299e-01	
RATE_INTEREST_PRIVILEGED	5951.0	7.735025e-01	0.100879	3.731501e-01	7.156448e-01	8.350951e-01	8.525370e-01	
DAYS_DECISION	1670214.0	-8.806797e+02	779.099667	-2.922000e+03	-1.300000e+03	-5.810000e+02	-2.800000e+02	
SELLERPLACE_AREA	1670214.0	3.139511e+02	7127.443459	-1.000000e+00	-1.000000e+00	3.000000e+00	8.200000e+01	4
CNT_PAYMENT	1297984.0	1.605408e+01	14.567288	0.000000e+00	6.000000e+00	1.200000e+01	2.400000e+01	
DAYS_FIRST_DRAWING	997149.0	3.422099e+05	88916.115834	-2.922000e+03	3.652430e+05	3.652430e+05	3.652430e+05	
DAYS_FIRST_DUE	997149.0	1.382627e+04	72444.869708	-2.892000e+03	-1.628000e+03	-8.310000e+02	-4.110000e+02	
DAYS_LAST_DUE_1ST_VERSION	997149.0	3.376777e+04	106857.034789	-2.801000e+03	-1.242000e+03	-3.610000e+02	1.290000e+02	
DAYS_LAST_DUE	997149.0	7.658240e+04	149647.415123	-2.889000e+03	-1.314000e+03	-5.370000e+02	-7.400000e+01	
DAYS_TERMINATION	997149.0	8.199234e+04	153303.516729	-2.874000e+03	-1.270000e+03	-4.990000e+02	-4.400000e+01	
NFLAG_INSURED_ON_APPROVAL	997149.0	3.325702e-01	0.471134	0.000000e+00	0.000000e+00	0.000000e+00	1.000000e+00	

In [149]:
```python
perc_missing = round(prev_data.isna().sum() / prev_data.shape[0], 2)

#number of columns with no missing values
perc_missing[perc_missing == 0].size
```

Out[149]: 23

In [150]:
```python
#number of columns with missing values
perc_missing[perc_missing != 0].size
```

Out[150]: 14

In [151]:
```python
perc_missing[perc_missing>0.5]
```

Out[151]:
```
AMT_DOWN_PAYMENT            0.54
RATE_DOWN_PAYMENT           0.54
RATE_INTEREST_PRIMARY       1.00
RATE_INTEREST_PRIVILEGED    1.00
dtype: float64
```

In [152]:
```python
# As there are morethan 50% missing values in AMT_DOWN_PAYMENT,RATE_DOWN_PAYMENT
# We cannot use these colns for analysis
```

In [ ]:

In [153]:
```python
prev_data['SK_ID_CURR'].value_counts()
```

Out[153]:
```
187868    77
265681    73
173680    72
242412    68
206783    67
          ..
382489     1
426056     1
454726     1
380442     1
124145     1
Name: SK_ID_CURR, Length: 338857, dtype: int64
```

In [154]:
```python
## Few appl numbers are repeated --> Assuming them as fraud
```

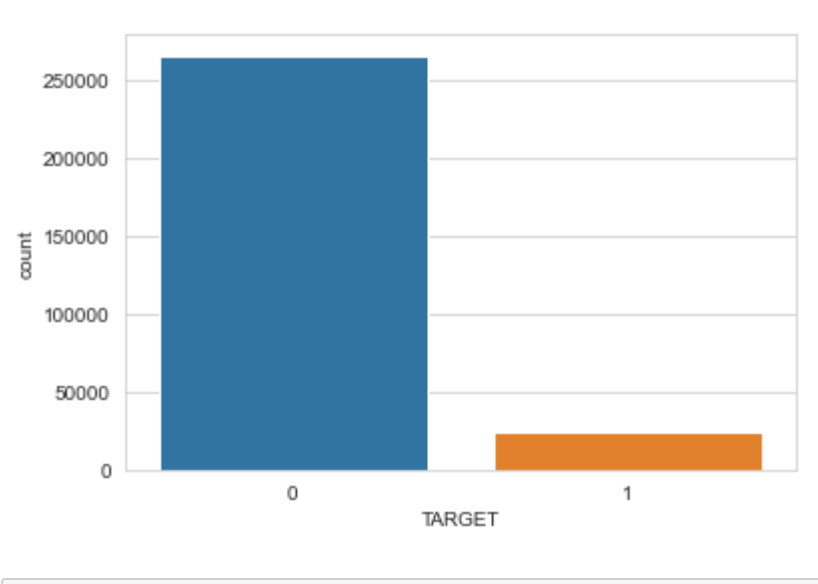
In [155]:
```python
fraud_id=(prev_data['SK_ID_CURR'].value_counts()!=1).index
```

In [156]:
```python
valid_appl=appl_data[~appl_data['SK_ID_CURR'].isin (fraud_id)]
```

In [157]:
```python
fraud_appl=appl_data[appl_data['SK_ID_CURR'].isin(fraud_id)]
```

In [158]:
```python
sns.countplot(fraud_appl['TARGET'])
```

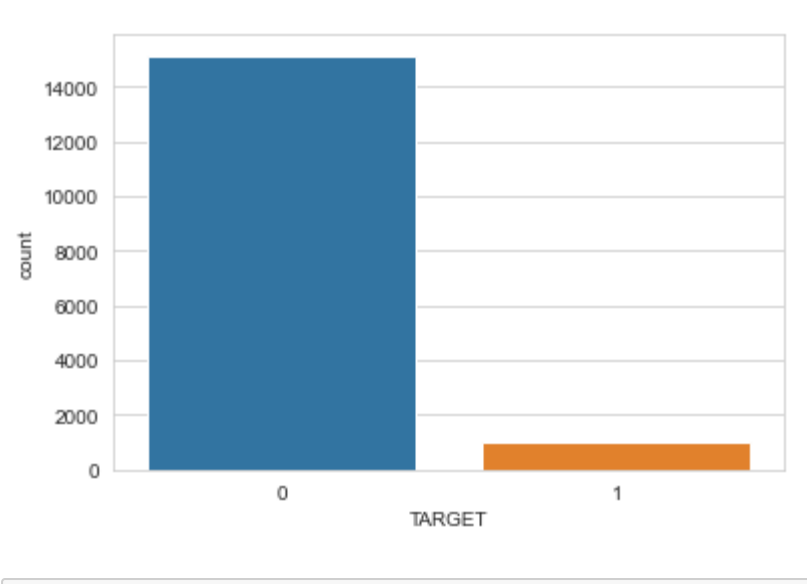
Out[158]: <AxesSubplot:xlabel='TARGET', ylabel='count'>



In [159]:
```python
# We can observe that many fraud applications are treated as defaulter
```

In [160]:
```python
sns.countplot(valid_appl['TARGET'])
```

Out[160]: <AxesSubplot:xlabel='TARGET', ylabel='count'>



In [161]:
```python
prev_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 37 columns):
 #   Column                       Non-Null Count    Dtype
---  ------                       --------------    -----
 0   SK_ID_PREV                   1670214 non-null  int64
 1   SK_ID_CURR                   1670214 non-null  int64
 2   NAME_CONTRACT_TYPE           1670214 non-null  object
 3   AMT_ANNUITY                  1297979 non-null  float64
 4   AMT_APPLICATION              1670214 non-null  float64
 5   AMT_CREDIT                   1670213 non-null  float64
 6   AMT_DOWN_PAYMENT             774370 non-null   float64
 7   AMT_GOODS_PRICE              1284699 non-null  float64
 8   WEEKDAY_APPR_PROCESS_START   1670214 non-null  object
 9   HOUR_APPR_PROCESS_START      1670214 non-null  int64
 10  FLAG_LAST_APPL_PER_CONTRACT  1670214 non-null  object
 11  NFLAG_LAST_APPL_IN_DAY       1670214 non-null  int64
 12  RATE_DOWN_PAYMENT            774370 non-null   float64
 13  RATE_INTEREST_PRIMARY        5951 non-null     float64
 14  RATE_INTEREST_PRIVILEGED     5951 non-null     float64
 15  NAME_CASH_LOAN_PURPOSE       1670214 non-null  object
 16  NAME_CONTRACT_STATUS         1670214 non-null  object
 17  DAYS_DECISION                1670214 non-null  int64
 18  NAME_PAYMENT_TYPE            1670214 non-null  object
 19  CODE_REJECT_REASON           1670214 non-null  object
 20  NAME_TYPE_SUITE              849809 non-null   object
 21  NAME_CLIENT_TYPE             1670214 non-null  object
 22  NAME_GOODS_CATEGORY          1670214 non-null  object
 23  NAME_PORTFOLIO               1670214 non-null  object
 24  NAME_PRODUCT_TYPE            1670214 non-null  object
 25  CHANNEL_TYPE                 1670214 non-null  object
 26  SELLERPLACE_AREA             1670214 non-null  int64
 27  NAME_SELLER_INDUSTRY         1670214 non-null  object
 28  CNT_PAYMENT                  1297984 non-null  float64
 29  NAME_YIELD_GROUP             1670214 non-null  object
 30  PRODUCT_COMBINATION          1669868 non-null  object
 31  DAYS_FIRST_DRAWING           997149 non-null   float64
 32  DAYS_FIRST_DUE               997149 non-null   float64
 33  DAYS_LAST_DUE_1ST_VERSION    997149 non-null   float64
 34  DAYS_LAST_DUE                997149 non-null   float64
 35  DAYS_TERMINATION             997149 non-null   float64
 36  NFLAG_INSURED_ON_APPROVAL    997149 non-null   float64
dtypes: float64(15), int64(6), object(16)
memory usage: 471.5+ MB
```

In [162]:
```python
join_data = appl_data.join(prev_data, on='SK_ID_CURR', how='inner', lsuffix='_curr', rsuffix='_prev')
```

In [163]:
```python
join_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 305821 entries, 203693 to 1678
Data columns (total 88 columns):
 #   Column                            Non-Null Count   Dtype
---  ------                            --------------   -----
 0   SK_ID_CURR                        305821 non-null  int64
 1   SK_ID_CURR_curr                   305821 non-null  int64
 2   TARGET                            305821 non-null  int64
 3   NAME_CONTRACT_TYPE_curr           305821 non-null  object
 4   CODE_GENDER                       305821 non-null  object
 5   FLAG_OWN_CAR                      305821 non-null  object
 6   FLAG_OWN_REALTY                   305821 non-null  object
 7   CNT_CHILDREN                      305821 non-null  int64
 8   AMT_INCOME_TOTAL                  305821 non-null  float64
 9   AMT_CREDIT_curr                   305821 non-null  float64
 10  AMT_ANNUITY_curr                  305821 non-null  float64
 11  AMT_GOODS_PRICE_curr              305821 non-null  float64
 12  NAME_TYPE_SUITE_curr              305821 non-null  object
 13  NAME_INCOME_TYPE                  305821 non-null  object
 14  NAME_EDUCATION_TYPE               305821 non-null  object
 15  NAME_FAMILY_STATUS                305821 non-null  object
 16  NAME_HOUSING_TYPE                 305821 non-null  object
 17  REGION_POPULATION_RELATIVE        305821 non-null  float64
 18  FLAG_MOBIL                        305821 non-null  int64
 19  FLAG_EMP_PHONE                    305821 non-null  int64
 20  FLAG_WORK_PHONE                   305821 non-null  int64
 21  FLAG_CONT_MOBILE                  305821 non-null  int64
 22  FLAG_PHONE                        305821 non-null  int64
 23  FLAG_EMAIL                        305821 non-null  int64
 24  OCCUPATION_TYPE                   305821 non-null  object
 25  CNT_FAM_MEMBERS                   305821 non-null  float64
 26  REGION_RATING_CLIENT              305821 non-null  int64
 27  REGION_RATING_CLIENT_W_CITY       305821 non-null  int64
 28  WEEKDAY_APPR_PROCESS_START_curr   305821 non-null  object
 29  HOUR_APPR_PROCESS_START_curr      305821 non-null  int64
 30  REG_REGION_NOT_LIVE_REGION        305821 non-null  int64
 31  REG_REGION_NOT_WORK_REGION        305821 non-null  int64
 32  LIVE_REGION_NOT_WORK_REGION       305821 non-null  int64
 33  REG_CITY_NOT_LIVE_CITY            305821 non-null  int64
 34  REG_CITY_NOT_WORK_CITY            305821 non-null  int64
 35  LIVE_CITY_NOT_WORK_CITY           305821 non-null  int64
 36  ORGANIZATION_TYPE                 305821 non-null  object
 37  EXT_SOURCE_2                      305821 non-null  float64
 38  EXT_SOURCE_3                      245357 non-null  float64
 39  OBS_30_CNT_SOCIAL_CIRCLE          305821 non-null  float64
 40  DEF_30_CNT_SOCIAL_CIRCLE          305821 non-null  float64
 41  OBS_60_CNT_SOCIAL_CIRCLE          305821 non-null  float64
 42  DEF_60_CNT_SOCIAL_CIRCLE          305821 non-null  float64
 43  AGE                               305821 non-null  float64
 44  YEARS_EMPLOYED                    305821 non-null  float64
 45  YEARS_REGISTRATION                305821 non-null  float64
 46  YEARS_ID_PUBLISH                  305821 non-null  float64
 47  YEARS_LAST_PHONE_CHANGE           305821 non-null  float64
 48  Annuity_Class                     305821 non-null  category
 49  Income_Class                      305821 non-null  category
 50  INCOME_TYPE                       305821 non-null  object
 51  SK_ID_PREV                        305821 non-null  int64
 52  SK_ID_CURR_prev                   305821 non-null  int64
 53  NAME_CONTRACT_TYPE_prev           305821 non-null  object
 54  AMT_ANNUITY_prev                  237562 non-null  float64
 55  AMT_APPLICATION                   305821 non-null  float64
 56  AMT_CREDIT_prev                   305821 non-null  float64
 57  AMT_DOWN_PAYMENT                  141986 non-null  float64
 58  AMT_GOODS_PRICE_prev              235178 non-null  float64
 59  WEEKDAY_APPR_PROCESS_START_prev   305821 non-null  object
 60  HOUR_APPR_PROCESS_START_prev      305821 non-null  int64
 61  FLAG_LAST_APPL_PER_CONTRACT       305821 non-null  object
 62  NFLAG_LAST_APPL_IN_DAY            305821 non-null  int64
 63  RATE_DOWN_PAYMENT                 141986 non-null  float64
 64  RATE_INTEREST_PRIMARY             1090 non-null    float64
 65  RATE_INTEREST_PRIVILEGED          1090 non-null    float64
 66  NAME_CASH_LOAN_PURPOSE            305821 non-null  object
 67  NAME_CONTRACT_STATUS              305821 non-null  object
 68  DAYS_DECISION                     305821 non-null  int64
 69  NAME_PAYMENT_TYPE                 305821 non-null  object
 70  CODE_REJECT_REASON                305821 non-null  object
 71  NAME_TYPE_SUITE_prev              155283 non-null  object
 72  NAME_CLIENT_TYPE                  305821 non-null  object
 73  NAME_GOODS_CATEGORY               305821 non-null  object
 74  NAME_PORTFOLIO                    305821 non-null  object
 75  NAME_PRODUCT_TYPE                 305821 non-null  object
 76  CHANNEL_TYPE                      305821 non-null  object
 77  SELLERPLACE_AREA                  305821 non-null  int64
 78  NAME_SELLER_INDUSTRY              305821 non-null  object
 79  CNT_PAYMENT                       237563 non-null  float64
 80  NAME_YIELD_GROUP                  305821 non-null  object
 81  PRODUCT_COMBINATION               305757 non-null  object
 82  DAYS_FIRST_DRAWING                182215 non-null  float64
 83  DAYS_FIRST_DUE                    182215 non-null  float64
 84  DAYS_LAST_DUE_1ST_VERSION         182215 non-null  float64
 85  DAYS_LAST_DUE                     182215 non-null  float64
 86  DAYS_TERMINATION                  182215 non-null  float64
 87  NFLAG_INSURED_ON_APPROVAL         182215 non-null  float64
dtypes: category(2), float64(32), int64(25), object(29)
memory usage: 203.6+ MB
```

In [164]:
```python
join_data.loc[join_data['TARGET'] > 0]
```

Out[164]:

	SK_ID_CURR	SK_ID_CURR_curr	TARGET	NAME_CONTRACT_TYPE_curr	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY
248159	387126	387126	1	Cash loans	F	Y	Y
248970	388040	388040	1	Cash loans	M	Y	Y
41725	148308	148308	1	Cash loans	M	Y	Y
167656	294352	294352	1	Cash loans	F	N	Y
196106	327395	327395	1	Cash loans	F	Y	N
...	...	...	...	...	...	...	...
86070	199880	199880	1	Cash loans	F	Y	N
111197	229012	229012	1	Cash loans	F	N	N
76778	189037	189037	1	Cash loans	F	N	N
18548	121642	121642	1	Cash loans	F	N	N
20727	124157	124157	1	Cash loans	F	Y	Y

24736 rows × 88 columns

In [165]:
```python
fil = join_data.loc[join_data['TARGET'] > 0]['SK_ID_CURR'].value_counts()
fil.sum()
```

Out[165]: 24736

In [166]:
```python
## We can observe that few applications which are repeated in previous data are not present in applicat
ion data
join_data['SK_ID_CURR_prev'].value_counts()
```

Out[166]:
```
265681    21
110899    16
206783    16
173680    15
206862    15
          ..
345643     1
368176     1
366129     1
362035     1
278507     1
Name: SK_ID_CURR_prev, Length: 180724, dtype: int64
```

In [167]:
```python
prev_data['SK_ID_CURR'].value_counts()
```

Out[167]:
```
187868    77
265681    73
173680    72
242412    68
206783    67
          ..
382489     1
426056     1
454726     1
380442     1
124145     1
Name: SK_ID_CURR, Length: 338857, dtype: int64
```

In [168]:
```python
join_data[['SK_ID_CURR', 'TARGET', 'CODE_REJECT_REASON']]
```

Out[168]:

	SK_ID_CURR	TARGET	CODE_REJECT_REASON
203693	336147	0	HC
246858	385674	0	LIMIT
77768	190160	0	XAP
131127	252084	0	XAP
287463	432980	0	XAP
...	...	...	...
240137	378118	0	XAP
246104	384810	0	XAP
186643	316377	0	XAP
20727	124157	1	XAP
1678	101965	0	HC

305821 rows × 3 columns

In [169]:
```python
join_data[['SK_ID_CURR', 'TARGET', 'CODE_REJECT_REASON']].loc[join_data['TARGET'] > 0]
```

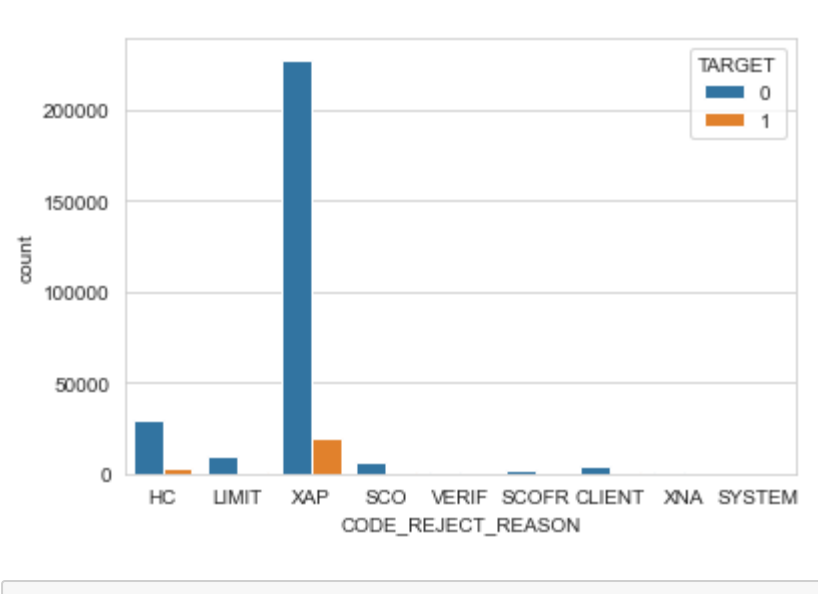
Out[169]:

	SK_ID_CURR	TARGET	CODE_REJECT_REASON
248159	387126	1	XAP
248970	388040	1	XAP
41725	148308	1	SCO
167656	294352	1	XAP
196106	327395	1	XAP
...	...	...	...
86070	199880	1	XAP
111197	229012	1	HC
76778	189037	1	XAP
18548	121642	1	XAP
20727	124157	1	XAP

24736 rows × 3 columns

In [170]:
```python
sns.countplot(data=join_data[['TARGET','CODE_REJECT_REASON']],x='CODE_REJECT_REASON',hue='TARGET')
```

Out[170]: <AxesSubplot:xlabel='CODE_REJECT_REASON', ylabel='count'>



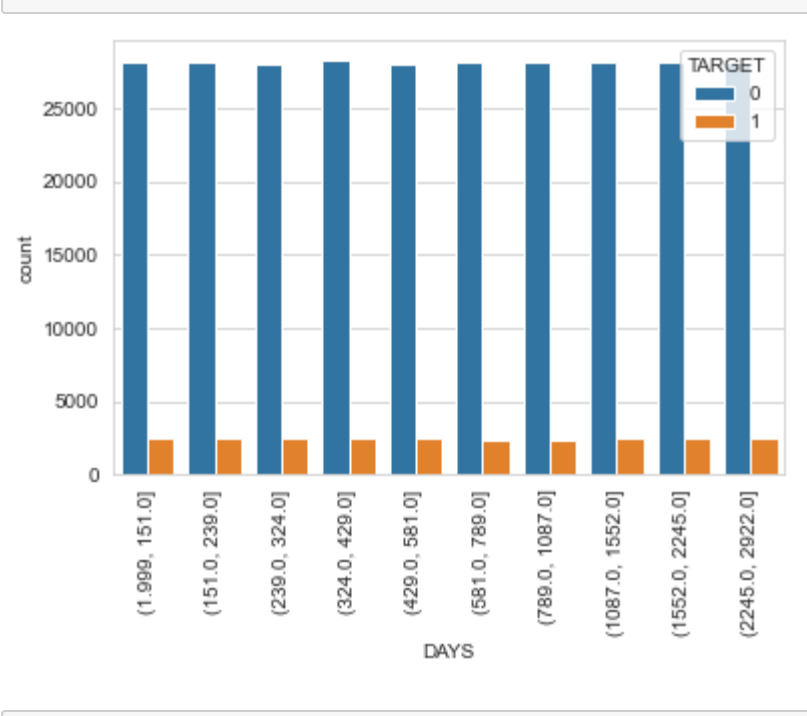
In [171]:
```python
join_data['DAYS_DECISION']
```

Out[171]:
```
203693    -220
246858    -656
77768     -792
131127   -1070
287463   -1197
           ...
240137    -734
246104    -270
186643    -642
20727     -579
1678      -324
Name: DAYS_DECISION, Length: 305821, dtype: int64
```

In [172]:
```python
join_data['DAYS_DECISION']=abs(join_data['DAYS_DECISION'])
```

In [173]:
```python
join_data['DAYS']=pd.qcut(join_data['DAYS_DECISION'],q=10)
```

In [174]:
```
sns.countplot(data=join_data[['TARGET','DAYS']],x='DAYS',hue='TARGET')
plt.xticks(rotation=90)
plt.show()
```



In [175]:
```
## Days to process the defaulters application is more compared to non defaulters
```

In [176]:
```
join_data[['SK_ID_CURR','DAYS_DECISION']].loc[join_data['TARGET']==1].sort_values(by='DAYS_DECISION')
```

Out[176]:

	SK_ID_CURR	DAYS_DECISION
229124	365383	2
35106	140667	2
87434	201476	2
85711	199463	2
170065	297103	2
...	...	...
291088	437210	2921
199478	331259	2922
293786	440369	2922
193764	324690	2922
223434	358790	2922

24736 rows × 2 columns

In [177]:
```
join_data[['SK_ID_CURR','DAYS_DECISION']].loc[join_data['TARGET']==0].sort_values(by='DAYS_DECISION')
```

Out[177]:

	SK_ID_CURR	DAYS_DECISION
22839	126576	2
292252	438560	2
278341	422476	2
219630	354440	2
87937	202087	2
...	...	...
34160	139586	2922
135141	256734	2922
115640	234100	2922
184510	313869	2922
91958	206788	2922

281085 rows × 2 columns

Notes on Analysis

- Pre-liminary analysis indicate that: Some applicants are having multiple previous applications There are more relevant business information like CODE_REJECT_REASON which means a direct impact on the Target variable. Previous Applications dataset has more than double the number of records across both the Target segments.