Loan Defaulter Segmentation

An EDA

Importing required libraries

In [1]:

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

pd.options.display.max_columns = None
pd.options.display.max_rows = None
```

Data import and basic exploration

In [2]:

```
app = pd.read_csv("application_data.csv")
prev_app = pd.read_csv("previous_application.csv")
```

In [3]:

```
app.head()
```

Out[3]:

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

In [4]:

```
prev_app.head()
```

Out[4]:

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

Feature Selection

In [5]:

```
app.columns
```

Out[5]:

```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
       'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL',
       'AMT_CREDIT', 'AMT_ANNUITY',
       ...
       'FLAG_DOCUMENT_18', 'FLAG_DOCUMENT_19', 'FLAG_DOCUMENT_20',
       'FLAG_DOCUMENT_21', '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', length=122)
```

In [6]:

```
app.shape
```

Out[6]:

```
(307511, 122)
```

In [7]:

```
# missing info

msng_info = pd.DataFrame(app.isnull().sum().sort_values()).reset_index()
msng_info.rename(columns = {"index" : "col_name", 0 : "null_count"}, inplace = True)
msng_info.head()
```

Out[7]:

	col_name	null_count
0	SK_ID_CURR	0
1	HOUR_APPR_PROCESS_START	0
2	REG_REGION_NOT_WORK_REGION	0
3	LIVE_REGION_NOT_WORK_REGION	0
4	REG_CITY_NOT_LIVE_CITY	0

In [8]:

```
# calculating percentage of missing data

msng_info["msng_pct"] = (msng_info["null_count"] / app.shape[0])*100
msng_info.tail()
```

Out[8]:

	col_name	null_count	msng_pct
117	NONLIVINGAPARTMENTS_MEDI	213514	69.432963
118	NONLIVINGAPARTMENTS_MODE	213514	69.432963
119	COMMONAREA_MODE	214865	69.872297
120	COMMONAREA_AVG	214865	69.872297
121	COMMONAREA_MEDI	214865	69.872297

In [9]:

```
msng_info.to_excel("missing_infor.xlsx", index = False)
```

In [10]:

```
msng_col = msng_info[msng_info["msng_pct"] > 40]["col_name"].to_list()
len(msng_col)
```

Out[10]:

```
49
```

In [11]:

```
# dropping 49 columns which have more that 40% of the data missing

app_msng_rmvd = app.drop(labels = msng_col, axis = 1)
```

In [12]:

```
app_msng_rmvd.shape
```

Out[12]:

```
(307511, 73)
```

In [13]:

```
app_msng_rmvd.head()
```

Out[13]:

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

In [14]:

```
# analysing the flag columns

flag_col = []

for col in app_msng_rmvd.columns :

    if col.startswith("FLAG_") :

        flag_col.append(col)

len(flag_col)
```

Out[14]:

```
28
```

In [15]:

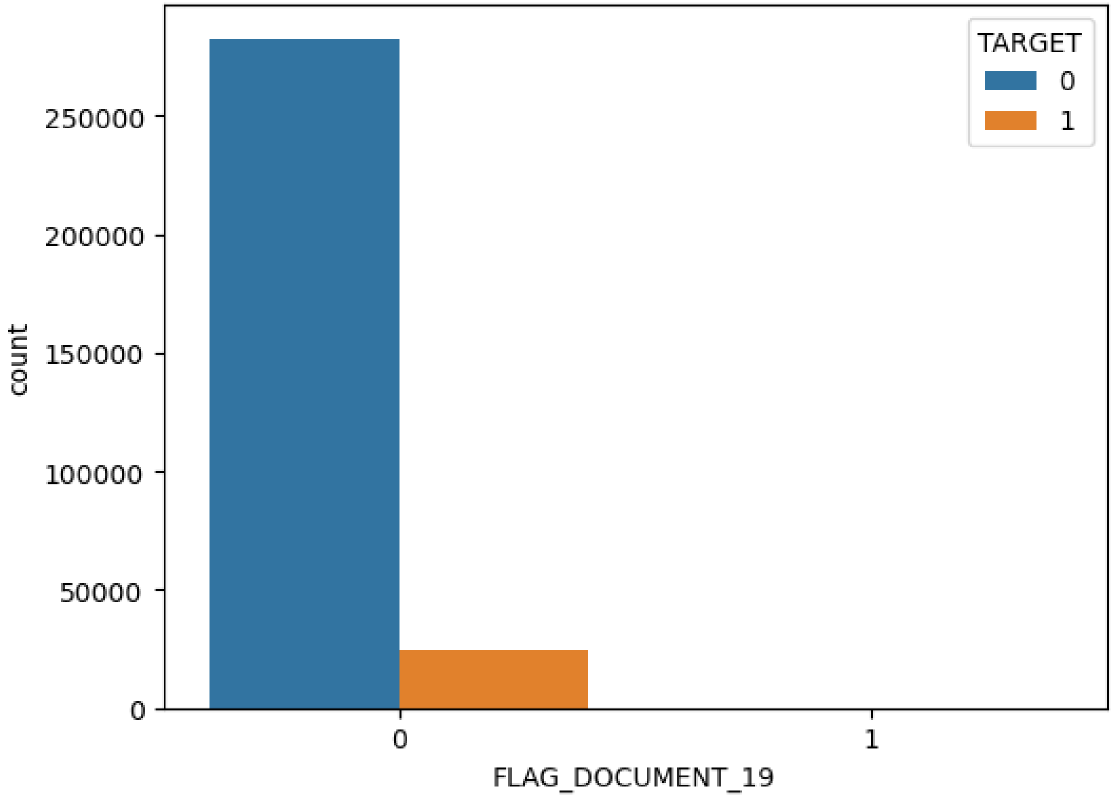
```
# seperating the flag columns in flag_tgt_colm

flag_tgt_colm = app_msng_rmvd[flag_col+["TARGET"]]
flag_tgt_colm.head()
```

Out[15]:

	FLAG_OWN_CAR	FLAG_OWN_REALTY	FLAG_MOBIL	FLAG_EMP_PHONE	FLAG_WORK_PHONE	FLAG_C
0	N	Y	1	1	0	
1	N	N	1	1	0	
2	Y	Y	1	1	1	
3	N	Y	1	1	0	
4	N	Y	1	1	0	

In [16]:

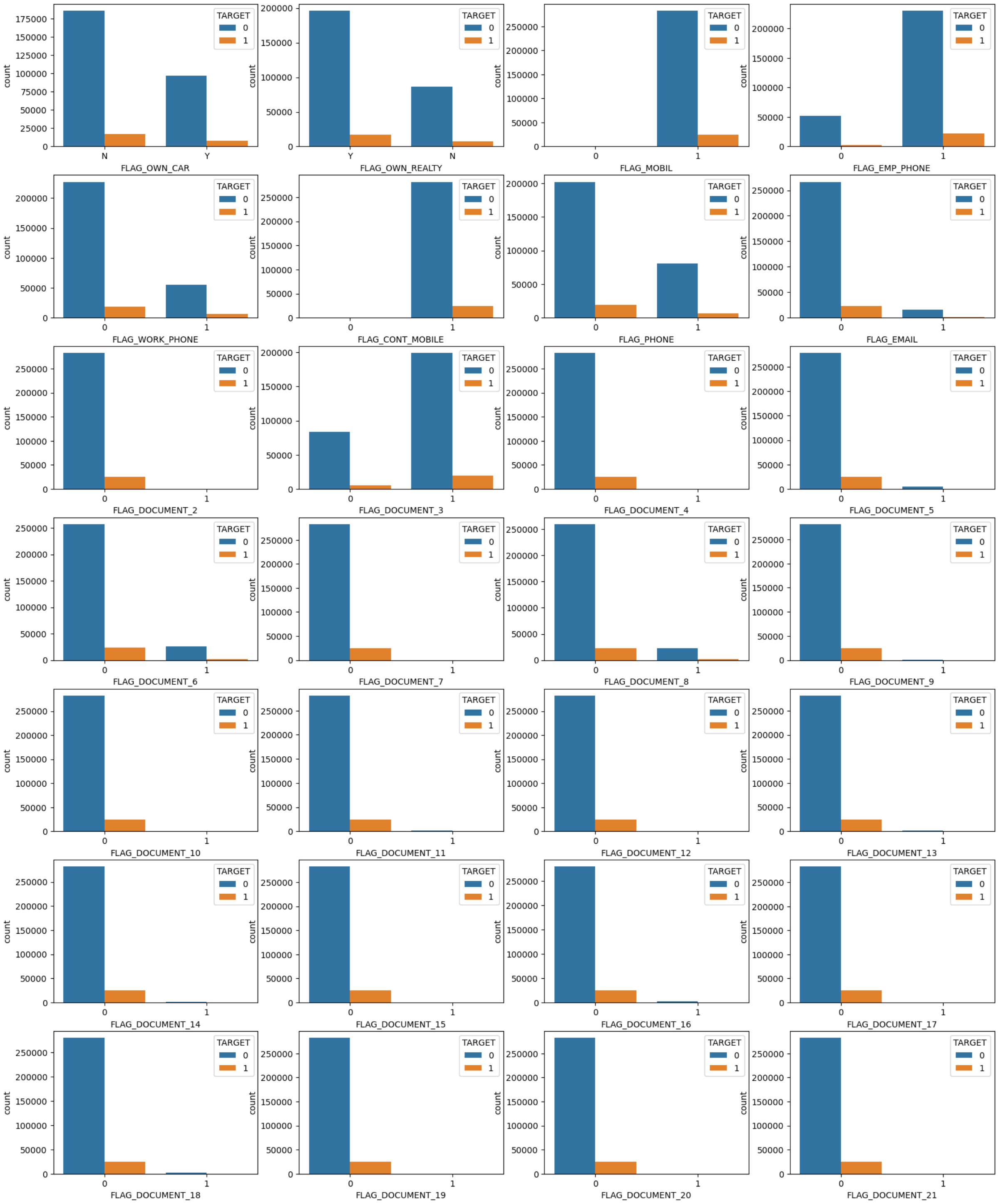
```
# Target : 1 means Defaulter, 0 means ok
# FLAG_DOCUMNET_19 : 0 means not submitted, 1 means submitted

sns.countplot(data = flag_tgt_colm, x = "FLAG_DOCUMENT_19", hue = "TARGET")
plt.show()
```



Where ever the documnets are not being submitted, we can see that flag column is not needed for our loan defaulter analysis

In [17]:

```
# We now need to check the relation of these flag columns with the TARGET column.
# This is to know which flag columns are necessary for our analysis and which not.

plt.figure(figsize = (20,25))

for index, col in enumerate(flag_col) :

    plt.subplot(7, 4, index+1)
    sns.countplot(data = flag_tgt_colm, x = col, hue = "TARGET")
```



In [18]:

```
# viewing the FLAG columns which might have correlation with our TARGET column

flg_corr = ['FLAG_OWN_CAR',
 'FLAG_OWN_REALTY',
 'FLAG_MOBIL',
 'FLAG_EMP_PHONE',
 'FLAG_WORK_PHONE',
 'FLAG_CONT_MOBILE',
 'FLAG_PHONE',
 'FLAG_EMAIL',
 'TARGET']
flag_corr_df = app_msng_rmvd[flg_corr]
corr_df = round(flag_corr_df.corr(numeric_only = True), 2)
```

In [19]:

```
flag_corr_df.groupby(["FLAG_OWN_CAR"]).size()
```

Out[19]:

```
FLAG_OWN_CAR
N    202924
Y    104587
dtype: int64
```

In [20]:

```
flag_corr_df.groupby(["FLAG_OWN_REALTY"]).size()
```

Out[20]:

```
FLAG_OWN_REALTY
N     94199
Y    213312
dtype: int64
```

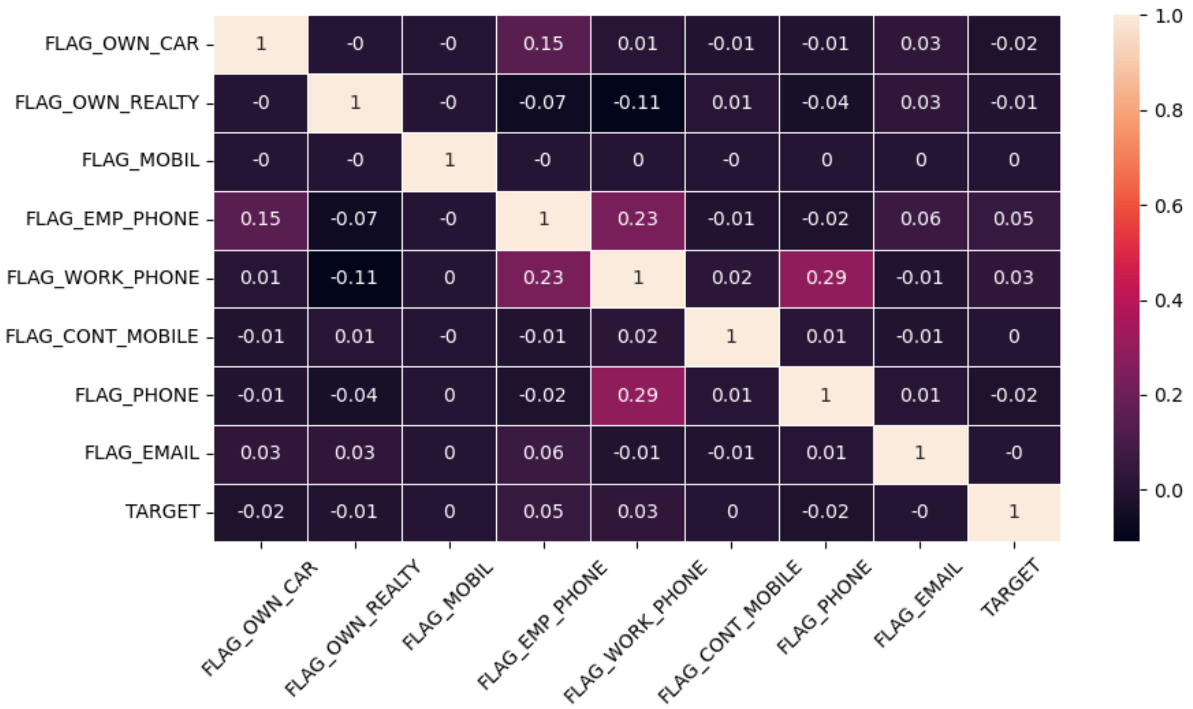
In [21]:

```
# Replacing 'Y' with 1 and 'N' with 0 in the 'FLAG_OWN_CAR' column
flag_corr_df.loc[:, "FLAG_OWN_CAR"] = flag_corr_df["FLAG_OWN_CAR"].replace({'Y': 1, 'N':

# Replacing 'Y' with 1 and 'N' with 0 in the 'FLAG_OWN_REALTY' column
flag_corr_df.loc[:, "FLAG_OWN_REALTY"] = flag_corr_df["FLAG_OWN_REALTY"].replace({'Y': 1
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_10932\2612263016.py:2: SettingWithCopyWarnin
g:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  flag_corr_df.loc[:, "FLAG_OWN_CAR"] = flag_corr_df["FLAG_OWN_CAR"].replace({'Y': 1,
'N': 0})
C:\Users\Admin\AppData\Local\Temp\ipykernel_10932\2612263016.py:2: DeprecationWarning: I
n a future version, `df.iloc[:, i] = newvals` will attempt to set the values inplace ins
tead of always setting a new array. To retain the old behavior, use either `df[df.column
s[i]] = newvals` or, if columns are non-unique, `df.isetitem(i, newvals)`
  flag_corr_df.loc[:, "FLAG_OWN_CAR"] = flag_corr_df["FLAG_OWN_CAR"].replace({'Y': 1,
'N': 0})
C:\Users\Admin\AppData\Local\Temp\ipykernel_10932\2612263016.py:5: SettingWithCopyWarnin
g:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  flag_corr_df.loc[:, "FLAG_OWN_REALTY"] = flag_corr_df["FLAG_OWN_REALTY"].replace({'Y':
1, 'N': 0})
C:\Users\Admin\AppData\Local\Temp\ipykernel_10932\2612263016.py:5: DeprecationWarning: I
n a future version, `df.iloc[:, i] = newvals` will attempt to set the values inplace ins
tead of always setting a new array. To retain the old behavior, use either `df[df.column
s[i]] = newvals` or, if columns are non-unique, `df.isetitem(i, newvals)`
  flag_corr_df.loc[:, "FLAG_OWN_REALTY"] = flag_corr_df["FLAG_OWN_REALTY"].replace({'Y':
1, 'N': 0})
```

In [22]:

```
# plotting the correlation
corr_df = round(flag_corr_df.corr(numeric_only = True), 2)

plt.figure(figsize = (10,5))
sns.heatmap(corr_df, cmap = "rocket", annot = True, linewidths = 0.5)
plt.xticks(rotation = 45)
plt.show()
```



None of these flag columns have a strong correlation with the TARGET column, so we need to drop them off.

```
In [23]: app_flag_rmvd = app_msng_rmvd.drop(labels = flag_col, axis = 1)
         app_flag_rmvd.shape
```

```
Out[23]: (307511, 45)
```

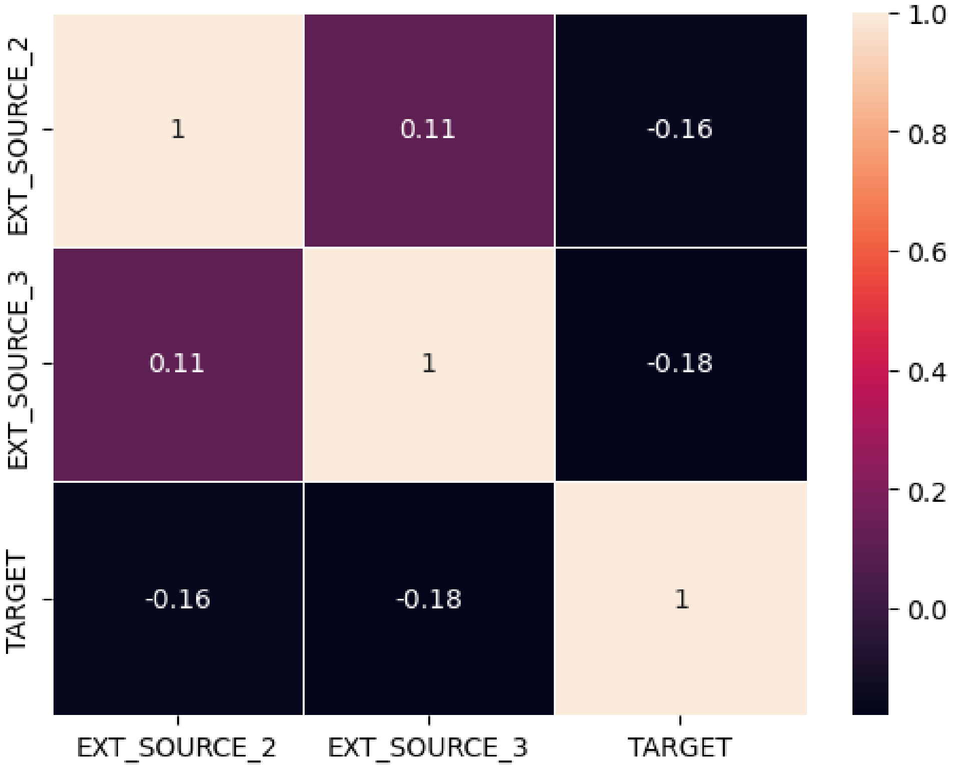
```
In [24]: app_flag_rmvd.head()
```

Out[24]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	CNT_CHILDREN	AMT_INCOME_TOTAL
0	100002	1	Cash loans	M	0	202500.0
1	100003	0	Cash loans	F	0	270000.0
2	100004	0	Revolving loans	M	0	67500.0
3	100006	0	Cash loans	F	0	135000.0
4	100007	0	Cash loans	M	0	121500.0

```
In [25]: sns.heatmap(data = round(app_flag_rmvd[["EXT_SOURCE_2","EXT_SOURCE_3","TARGET"]].corr(),
                     cmap = "rocket", linewidths = 0.5, annot = True)
```

```
Out[25]: <Axes: >
```



In [26]:
```
# as these columns named EXT_SOURCE_2 and EXT_SOURCE_3 don't have a good correlation wit

app_score_col_rmvd = app_flag_rmvd.drop(["EXT_SOURCE_2","EXT_SOURCE_3"], axis = 1)
app_score_col_rmvd.shape
```

Out[26]:
```
(307511, 43)
```

Feature Engineering

In [27]:
```
((app_score_col_rmvd.isnull().sum() / app_score_col_rmvd.shape[0])*100).sort_values(asce
```

Out[27]:

```
OCCUPATION_TYPE                31.345545
AMT_REQ_CREDIT_BUREAU_YEAR     13.501631
AMT_REQ_CREDIT_BUREAU_QRT      13.501631
AMT_REQ_CREDIT_BUREAU_MON      13.501631
AMT_REQ_CREDIT_BUREAU_WEEK     13.501631
AMT_REQ_CREDIT_BUREAU_DAY      13.501631
AMT_REQ_CREDIT_BUREAU_HOUR     13.501631
NAME_TYPE_SUITE                 0.420148
DEF_60_CNT_SOCIAL_CIRCLE        0.332021
OBS_30_CNT_SOCIAL_CIRCLE        0.332021
DEF_30_CNT_SOCIAL_CIRCLE        0.332021
OBS_60_CNT_SOCIAL_CIRCLE        0.332021
AMT_GOODS_PRICE                 0.090403
AMT_ANNUITY                     0.003902
CNT_FAM_MEMBERS                 0.000650
DAYS_LAST_PHONE_CHANGE          0.000325
HOUR_APPR_PROCESS_START         0.000000
ORGANIZATION_TYPE               0.000000
LIVE_CITY_NOT_WORK_CITY         0.000000
REG_CITY_NOT_WORK_CITY          0.000000
REG_CITY_NOT_LIVE_CITY          0.000000
LIVE_REGION_NOT_WORK_REGION     0.000000
REG_REGION_NOT_WORK_REGION      0.000000
REG_REGION_NOT_LIVE_REGION      0.000000
SK_ID_CURR                      0.000000
WEEKDAY_APPR_PROCESS_START      0.000000
REGION_RATING_CLIENT_W_CITY     0.000000
NAME_CONTRACT_TYPE              0.000000
CODE_GENDER                     0.000000
CNT_CHILDREN                    0.000000
AMT_INCOME_TOTAL                0.000000
AMT_CREDIT                      0.000000
NAME_INCOME_TYPE                0.000000
NAME_EDUCATION_TYPE             0.000000
NAME_FAMILY_STATUS              0.000000
NAME_HOUSING_TYPE               0.000000
REGION_POPULATION_RELATIVE      0.000000
DAYS_BIRTH                      0.000000
DAYS_EMPLOYED                   0.000000
DAYS_REGISTRATION               0.000000
DAYS_ID_PUBLISH                 0.000000
TARGET                          0.000000
REGION_RATING_CLIENT            0.000000
dtype: float64
```

Missing Imputation

In [28]:

```
app_score_col_rmvd.groupby("CNT_FAM_MEMBERS").size()
```

Out[28]:

```
CNT_FAM_MEMBERS
1.0      67847
2.0     158357
3.0      52601
4.0      24697
5.0       3478
6.0        408
7.0         81
8.0         20
9.0          6
10.0         3
11.0         1
12.0         2
13.0         1
14.0         2
15.0         1
16.0         2
20.0         2
dtype: int64
```

In [29]:

```
app_score_col_rmvd["CNT_FAM_MEMBERS"].mode()
```

Out[29]:

```
0    2.0
Name: CNT_FAM_MEMBERS, dtype: float64
```

In [30]:

```
# replacing the missing values in CNT_FAM_MEMBERS with it's mode value for better analys

mode_value = app_score_col_rmvd["CNT_FAM_MEMBERS"].mode()[0]
app_score_col_rmvd["CNT_FAM_MEMBERS"] = app_score_col_rmvd["CNT_FAM_MEMBERS"].fillna(flo
```

In [31]:

```
app_score_col_rmvd.groupby("OCCUPATION_TYPE").size().sort_values(ascending = False)
```

Out[31]:

```
OCCUPATION_TYPE
Laborers                 55186
Sales staff              32102
Core staff               27570
Managers                 21371
Drivers                  18603
High skill tech staff    11380
Accountants               9813
Medicine staff            8537
Security staff            6721
Cooking staff             5946
Cleaning staff            4653
Private service staff     2652
Low-skill Laborers        2093
Waiters/barmen staff      1348
Secretaries               1305
Realty agents              751
HR staff                   563
IT staff                   526
dtype: int64
```

In [32]:

```
app_score_col_rmvd["OCCUPATION_TYPE"].mode()[0]
```

Out[32]:

```
'Laborers'
```

In [33]:

```
# replacing the missing values in OCCUPATION_TYPE with it's mode value that is 'Laborers

app_score_col_rmvd["OCCUPATION_TYPE"] = app_score_col_rmvd["OCCUPATION_TYPE"].fillna
((app_score_col_rmvd["OCCUPATION_TYPE"].mode()[0]))

app_score_col_rmvd["OCCUPATION_TYPE"].isnull().sum()
```

Out[33]:

```
0
```

In [34]:
```
app_score_col_rmvd.groupby("NAME_TYPE_SUITE").size().sort_values(ascending = False)
```

Out[34]:
```
NAME_TYPE_SUITE
Unaccompanied      248526
Family              40149
Spouse, partner     11370
Children             3267
Other_B              1770
Other_A               866
Group of people       271
dtype: int64
```

In [35]:
```
app_score_col_rmvd["NAME_TYPE_SUITE"].mode()[0]
```

Out[35]:
```
'Unaccompanied'
```

In [36]:
```
# replacing the missing values in NAME_TYPE_SUITE with it's mode value that is 'Unaccomp

app_score_col_rmvd["NAME_TYPE_SUITE"] = app_score_col_rmvd["NAME_TYPE_SUITE"].fillna
((app_score_col_rmvd["NAME_TYPE_SUITE"].mode()[0]))

app_score_col_rmvd["NAME_TYPE_SUITE"].isnull().sum()
```

Out[36]:
```
0
```

In [37]:
```
# Replacing the missing values in "AMT_ANNUITY" with its mean value
app_score_col_rmvd["AMT_ANNUITY"] = app_score_col_rmvd["AMT_ANNUITY"].fillna(app_score_c

# Verifying if there are any remaining missing values in the "AMT_ANNUITY" column
print(app_score_col_rmvd["AMT_ANNUITY"].isnull().sum())
```

```
0
```

In [38]:
```
app_score_col_rmvd["AMT_REQ_CREDIT_BUREAU_HOUR"].describe()
```

Out[38]:
```
count    265992.000000
mean          0.006402
std           0.083849
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max           4.000000
Name: AMT_REQ_CREDIT_BUREAU_HOUR, dtype: float64
```

In [39]:
```
app_score_col_rmvd["AMT_REQ_CREDIT_BUREAU_HOUR"].value_counts()
```

Out[39]:
```
0.0    264366
1.0      1560
2.0        56
3.0         9
4.0         1
Name: AMT_REQ_CREDIT_BUREAU_HOUR, dtype: int64
```

In [40]:
```
amt_req_col = []

for col in app_score_col_rmvd.columns:
    if col.startswith("AMT_REQ_CREDIT_BUREAU"):
        amt_req_col.append(col)

amt_req_col
```

Out[40]:

```
['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 [41]:

```
# replacing the null values in AMT_REQ_CREDIT_BUREAU_HOUR, AMT_REQ_CREDIT_BUREAU_DAY, AM
# AMT_REQ_CREDIT_BUREAU_MON, AMT_REQ_CREDIT_BUREAU_QRT, AMT_REQ_CREDIT_BUREAU_YEAR colum

for col in amt_req_col:
    app_score_col_rmvd[col] = app_score_col_rmvd[col].fillna((app_score_col_rmvd[col].me
```

In [42]:

```
app_score_col_rmvd[col].isnull().sum()
```

Out[42]:

```
0
```

In [43]:

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

Out[43]:

```
278
```

In [44]:

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

Out[44]:

```
count    3.072330e+05
mean     5.383962e+05
std      3.694465e+05
min      4.050000e+04
25%      2.385000e+05
50%      4.500000e+05
75%      6.795000e+05
max      4.050000e+06
Name: AMT_GOODS_PRICE, dtype: float64
```

In [45]:

```
app_score_col_rmvd["AMT_GOODS_PRICE"].agg(['min','max','median'])
```

Out[45]:

```
min         40500.0
max       4050000.0
median     450000.0
Name: AMT_GOODS_PRICE, dtype: float64
```

In [46]:

```
app_score_col_rmvd["AMT_GOODS_PRICE"].mean()
```

Out[46]:

```
538396.2074288895
```

In [47]:

```
app_score_col_rmvd["AMT_GOODS_PRICE"].dtype
```

Out[47]:

```
dtype('float64')
```

In [48]:

```
app_score_col_rmvd["AMT_GOODS_PRICE"] = app_score_col_rmvd["AMT_GOODS_PRICE"].fillna(app
```

In [49]:

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

Out[49]:

```
0
```

In [50]:

```
days_col = []

for col in app_score_col_rmvd.columns:
    if col.startswith("DAYS"):
        days_col.append(col)

days_col
```

Out[50]:
```
['DAYS_BIRTH',
 'DAYS_EMPLOYED',
 'DAYS_REGISTRATION',
 'DAYS_ID_PUBLISH',
 'DAYS_LAST_PHONE_CHANGE']
```

In [51]:
```
for col in days_col:
    app_score_col_rmvd[col] = abs(app_score_col_rmvd[col])
```

In [52]:
```
app_score_col_rmvd.head(1)
```

Out[52]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	CNT_CHILDREN	AMT_INCOME_TOTAL
0	100002	1	Cash loans	M	0	202500.0

Outlier Detection & Treatment

In [53]:
```
app_score_col_rmvd.nunique().sort_values()
```

Out[53]:

```
OCCUPATION_TYPE                     1
NAME_TYPE_SUITE                     1
LIVE_REGION_NOT_WORK_REGION         2
TARGET                              2
NAME_CONTRACT_TYPE                  2
REG_REGION_NOT_LIVE_REGION          2
REG_CITY_NOT_LIVE_CITY              2
REG_CITY_NOT_WORK_CITY              2
LIVE_CITY_NOT_WORK_CITY             2
REG_REGION_NOT_WORK_REGION          2
REGION_RATING_CLIENT_W_CITY         3
REGION_RATING_CLIENT                3
CODE_GENDER                         3
AMT_REQ_CREDIT_BUREAU_HOUR          5
NAME_EDUCATION_TYPE                 5
NAME_FAMILY_STATUS                  6
NAME_HOUSING_TYPE                   6
WEEKDAY_APPR_PROCESS_START          7
NAME_INCOME_TYPE                    8
DEF_60_CNT_SOCIAL_CIRCLE            9
AMT_REQ_CREDIT_BUREAU_DAY           9
AMT_REQ_CREDIT_BUREAU_WEEK          9
DEF_30_CNT_SOCIAL_CIRCLE           10
AMT_REQ_CREDIT_BUREAU_QRT          11
CNT_CHILDREN                       15
CNT_FAM_MEMBERS                    17
AMT_REQ_CREDIT_BUREAU_MON          24
HOUR_APPR_PROCESS_START            24
AMT_REQ_CREDIT_BUREAU_YEAR         25
OBS_30_CNT_SOCIAL_CIRCLE           33
OBS_60_CNT_SOCIAL_CIRCLE           33
ORGANIZATION_TYPE                  58
REGION_POPULATION_RELATIVE         81
AMT_GOODS_PRICE                  1002
AMT_INCOME_TOTAL                 2548
DAYS_LAST_PHONE_CHANGE           3773
AMT_CREDIT                       5603
DAYS_ID_PUBLISH                  6168
DAYS_EMPLOYED                   12574
AMT_ANNUITY                     13673
DAYS_REGISTRATION               15688
DAYS_BIRTH                      17460
SK_ID_CURR                     307511
dtype: int64
```

In [54]:

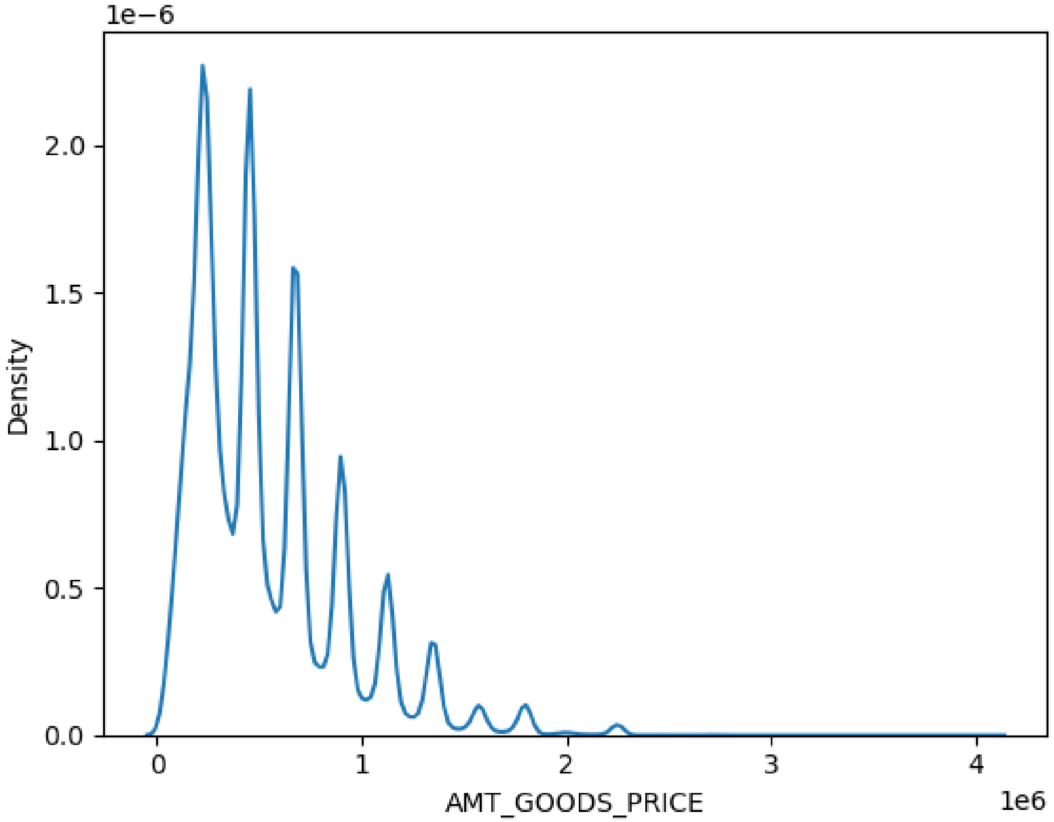
```
app_score_col_rmvd["AMT_GOODS_PRICE"].agg(['min','max','median'])
```

Out[54]:

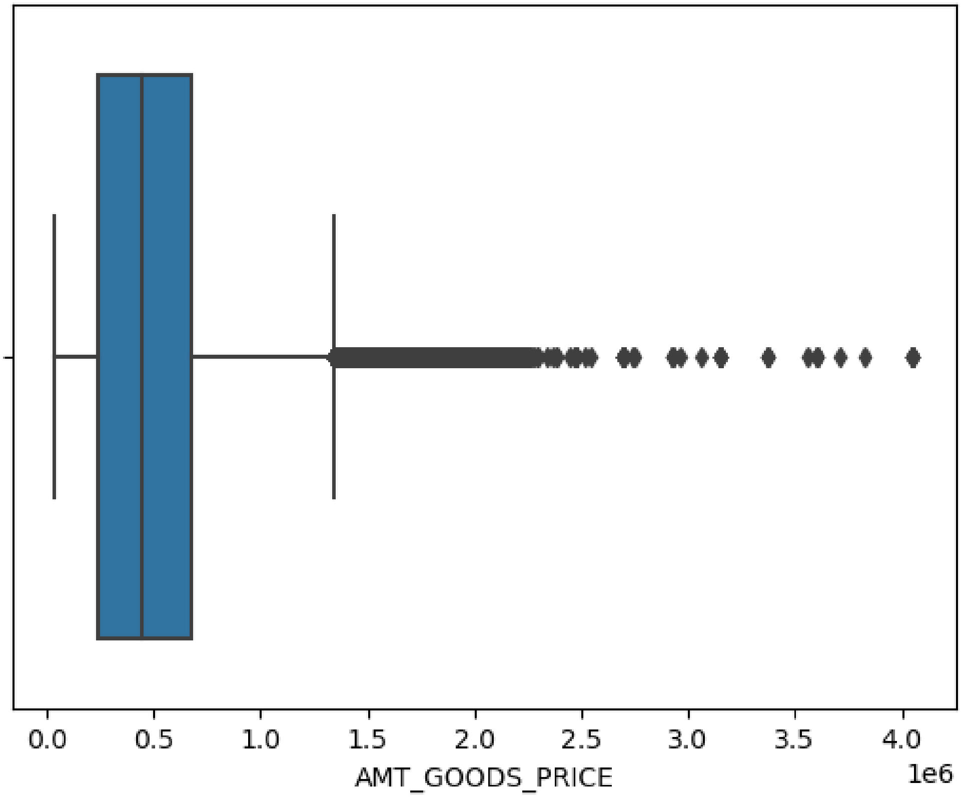
```
min         40500.0
max       4050000.0
median     450000.0
Name: AMT_GOODS_PRICE, dtype: float64
```

In [55]:

```
sns.kdeplot(x = "AMT_GOODS_PRICE", data = app_score_col_rmvd)
plt.show()
```



```
In [56]: sns.boxplot(x = "AMT_GOODS_PRICE", data = app_score_col_rmvd)
         plt.show()
```



```
In [57]: app_score_col_rmvd["AMT_GOODS_PRICE"].quantile([0.1, 0.2, 0.3,0.4, 0.5, 0.6, 0.7, 0.8, 0
```

Out[57]:

```
0.10     180000.0
0.20     225000.0
0.30     270000.0
0.40     378000.0
0.50     450000.0
0.60     522000.0
0.70     675000.0
0.80     814500.0
0.90    1093500.0
0.99    1800000.0
Name: AMT_GOODS_PRICE, dtype: float64
```

Binning

In [58]:

```
bins = [0, 100000, 200000, 300000, 400000, 500000, 600000, 700000, 800000, 900000, 40500
ranges = ['0-100K','100k-200K','200K-300K','300K-400K','400K-500K','500K-600K',
          '600K-700K','700K-800K','800K-900K','Above 900K']

app_score_col_rmvd['AMT_GOODS_PRICE_RANGE'] = pd.cut(app_score_col_rmvd['AMT_GOODS_PRICE
```

In [59]:

```
app_score_col_rmvd.groupby(['AMT_GOODS_PRICE_RANGE']).size()
```

Out[59]:

```
AMT_GOODS_PRICE_RANGE
0-100K         8709
100k-200K     32956
200K-300K     62761
300K-400K     21219
400K-500K     57251
500K-600K     13117
600K-700K     40024
700K-800K      8110
800K-900K     21484
Above 900K    41880
dtype: int64
```

In [60]:

```
app_score_col_rmvd['AMT_INCOME_TOTAL'].quantile([0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8,
```

Out[60]:

```
0.10     81000.0
0.20     99000.0
0.30    112500.0
0.40    135000.0
0.50    147150.0
0.60    162000.0
0.70    180000.0
0.80    225000.0
0.90    270000.0
0.99    472500.0
Name: AMT_INCOME_TOTAL, dtype: float64
```

In [61]:

```
app_score_col_rmvd['AMT_INCOME_TOTAL'].max()
```

Out[61]:

```
117000000.0
```

In [62]:

```
bins = [0, 100000, 150000, 200000, 250000, 300000, 350000, 400000, 117000000]
ranges = ['0-100K', '100K-150K', '150K-200K', '200K-250K', '250K-300K', '300K-350K', '35

app_score_col_rmvd['AMT_INCOME_TOTAL_RANGE'] = pd.cut(app_score_col_rmvd['AMT_INCOME_TOT
```

In [63]:

```
app_score_col_rmvd.groupby(['AMT_INCOME_TOTAL_RANGE']).size()
```

Out[63]:

```
AMT_INCOME_TOTAL_RANGE
0-100K        63698
100K-150K     91591
150K-200K     64307
200K-250K     48137
250K-300K     17039
300K-350K      8874
350K-400K      5802
Above 400K     8063
dtype: int64
```

In [64]:

```
app_score_col_rmvd['AMT_CREDIT'].quantile([0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9,
```

Out[64]:

```
0.10     180000.0
0.20     254700.0
0.30     306306.0
0.40     432000.0
0.50     513531.0
0.60     604152.0
0.70     755190.0
0.80     900000.0
0.90    1133748.0
0.99    1854000.0
Name: AMT_CREDIT, dtype: float64
```

In [65]:

```
app_score_col_rmvd['AMT_CREDIT'].max()
```

Out[65]:

```
4050000.0
```

In [66]:

```
bins = [0, 200000, 400000, 600000, 800000, 900000, 1000000, 2000000, 3000000, 4050000]
ranges = ['0-200K', '200K-400K', '400K-600K', '600K-800K', '800K-900K', '900K-1M', '1M-2

app_score_col_rmvd['AMT_CREDIT_RANGE'] = pd.cut(app_score_col_rmvd['AMT_CREDIT'], bins,
```

In [67]:

```
app_score_col_rmvd.groupby(['AMT_CREDIT_RANGE']).size()
```

Out[67]:

```
AMT_CREDIT_RANGE
0-200K       36144
200K-400K    81151
400K-600K    66270
600K-800K    43242
800K-900K    21792
900K-1M       8927
1M-2M        47956
2M-3M         1997
Above 3M        32
dtype: int64
```

In [68]:

```
app_score_col_rmvd['AMT_ANNUITY'].quantile([0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9,
```

Out[68]:

```
0.10    11074.5
0.20    14701.5
0.30    18189.0
0.40    21870.0
0.50    24903.0
0.60    28062.0
0.70    32004.0
0.80    37516.5
0.90    45954.0
0.99    70006.5
Name: AMT_ANNUITY, dtype: float64
```

In [69]:

```
app_score_col_rmvd['AMT_ANNUITY'].max()
```

Out[69]:

```
258025.5
```

In [70]:

```
bins = [0, 25000, 50000, 100000, 150000, 200000, 258025.5]
ranges = ['0-25K', '25K-50K', '50K-100K', '100K-150K', '150K-200K', 'Above 200K']

app_score_col_rmvd['AMT_ANNUITY_RANGE'] = pd.cut(app_score_col_rmvd['AMT_ANNUITY'], bins
```

In [71]:

```
app_score_col_rmvd.groupby(['AMT_ANNUITY_RANGE']).size()
```

Out[71]:

```
AMT_ANNUITY_RANGE
0-25K         154867
25K-50K       131347
50K-100K       20792
100K-150K        437
150K-200K         32
Above 200K        36
dtype: int64
```

In [72]:

```
app_score_col_rmvd['DAYS_EMPLOYED'].agg(['min', 'max', 'median'])
```

Out[72]:

```
min            0.0
max       365243.0
median      2219.0
Name: DAYS_EMPLOYED, dtype: float64
```

In [73]:

```
app_score_col_rmvd['DAYS_EMPLOYED'].quantile([0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.
```

Out[73]:

```
0.10       392.0
0.20       749.0
0.30      1132.0
0.40      1597.0
0.50      2219.0
0.60      3032.0
0.70      4435.0
0.80      9188.0
0.90    365243.0
0.99    365243.0
Name: DAYS_EMPLOYED, dtype: float64
```

In [74]:

```
app_score_col_rmvd.loc[app_score_col_rmvd['DAYS_EMPLOYED'] < app_score_col_rmvd['DAYS_EM
```

Out[74]:

```
17912
```

In [75]:

```
app_score_col_rmvd['DAYS_EMPLOYED'].max()
```

Out[75]:

```
365243
```

In [76]:

```
bins = [0, 1825, 3650, 5475, 7300, 9125, 10950, 12775, 14600, 16425, 18250, 23691, 36524
ranges = ['0-5Y', '5Y-10Y', '10Y-15Y', '15Y-20Y', '20Y-25Y', '25Y-30Y', '30Y-35Y', '35Y-
          '40Y-45Y', '45Y-50Y', '50Y-65Y', 'Above 65Y']

app_score_col_rmvd['DAYS_EMPLOYED_RANGE'] = pd.cut(app_score_col_rmvd['DAYS_EMPLOYED'],
```

In [77]:

```
app_score_col_rmvd.groupby(['DAYS_EMPLOYED_RANGE']).size()
```

Out[77]:

```
DAYS_EMPLOYED_RANGE
0-5Y          136309
5Y-10Y         64872
10Y-15Y        27549
15Y-20Y        10849
20Y-25Y         6243
25Y-30Y         3308
30Y-35Y         1939
35Y-40Y          832
40Y-45Y          210
45Y-50Y           24
50Y-65Y            0
Above 65Y      55374
dtype: int64
```

In [78]:

```
app_score_col_rmvd['DAYS_BIRTH'].quantile([0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.81,
```

Out[78]:

```
0.10    10284.0
0.20    11694.0
0.30    13140.0
0.40    14416.0
0.50    15750.0
0.60    17220.0
0.70    18885.0
0.80    20474.0
0.81    20641.0
0.85    21316.0
0.90    22181.0
0.95    23204.0
0.99    24419.0
Name: DAYS_BIRTH, dtype: float64
```

In [79]:

```
app_score_col_rmvd['DAYS_BIRTH'].min()
```

Out[79]:

```
7489
```

In [80]:

```
bins = [0, 7300, 10950, 14600, 18250, 21900, 25229]
ranges = ['20Y', '20Y-30Y', '30Y-40Y', '40Y-50Y', '50Y-60Y', 'Above 60Y']

app_score_col_rmvd['DAYS_BIRTH_RANGE'] = pd.cut(app_score_col_rmvd['DAYS_BIRTH'], bins,
```

In [81]:

```
app_score_col_rmvd.groupby(['DAYS_BIRTH_RANGE']).size()
```

Out[81]:

```
DAYS_BIRTH_RANGE
20Y              0
20Y-30Y      45021
30Y-40Y      82308
40Y-50Y      76541
50Y-60Y      68062
Above 60Y    35579
dtype: int64
```

Data Analysis

In [82]:

```
app_score_col_rmvd.dtypes
```

Out[82]:

```
SK_ID_CURR                        int64
TARGET                            int64
NAME_CONTRACT_TYPE               object
CODE_GENDER                      object
CNT_CHILDREN                      int64
AMT_INCOME_TOTAL                float64
AMT_CREDIT                      float64
AMT_ANNUITY                     float64
AMT_GOODS_PRICE                 float64
NAME_TYPE_SUITE                  object
NAME_INCOME_TYPE                 object
NAME_EDUCATION_TYPE              object
NAME_FAMILY_STATUS               object
NAME_HOUSING_TYPE                object
REGION_POPULATION_RELATIVE      float64
DAYS_BIRTH                        int64
DAYS_EMPLOYED                     int64
DAYS_REGISTRATION               float64
DAYS_ID_PUBLISH                   int64
OCCUPATION_TYPE                  object
CNT_FAM_MEMBERS                 float64
REGION_RATING_CLIENT              int64
REGION_RATING_CLIENT_W_CITY       int64
WEEKDAY_APPR_PROCESS_START       object
HOUR_APPR_PROCESS_START           int64
REG_REGION_NOT_LIVE_REGION        int64
REG_REGION_NOT_WORK_REGION        int64
LIVE_REGION_NOT_WORK_REGION       int64
REG_CITY_NOT_LIVE_CITY            int64
REG_CITY_NOT_WORK_CITY            int64
LIVE_CITY_NOT_WORK_CITY           int64
ORGANIZATION_TYPE                object
OBS_30_CNT_SOCIAL_CIRCLE        float64
DEF_30_CNT_SOCIAL_CIRCLE        float64
OBS_60_CNT_SOCIAL_CIRCLE        float64
DEF_60_CNT_SOCIAL_CIRCLE        float64
DAYS_LAST_PHONE_CHANGE          float64
AMT_REQ_CREDIT_BUREAU_HOUR      float64
AMT_REQ_CREDIT_BUREAU_DAY       float64
AMT_REQ_CREDIT_BUREAU_WEEK      float64
AMT_REQ_CREDIT_BUREAU_MON       float64
AMT_REQ_CREDIT_BUREAU_QRT       float64
AMT_REQ_CREDIT_BUREAU_YEAR      float64
AMT_GOODS_PRICE_RANGE          category
AMT_INCOME_TOTAL_RANGE         category
AMT_CREDIT_RANGE               category
AMT_ANNUITY_RANGE              category
DAYS_EMPLOYED_RANGE            category
DAYS_BIRTH_RANGE               category
dtype: object
```

In [83]:

```
app_score_col_rmvd.dtypes.value_counts()
```

Out[83]:

```
float64     18
int64       15
object      10
category     1
category     1
category     1
category     1
category     1
category     1
dtype: int64
```

In [84]:

```
obj_var = app_score_col_rmvd.select_dtypes(include = ["object"]).columns
obj_var
```

Out[84]:

```
Index(['NAME_CONTRACT_TYPE', 'CODE_GENDER', 'NAME_TYPE_SUITE',
       'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS',
       'NAME_HOUSING_TYPE', 'OCCUPATION_TYPE', 'WEEKDAY_APPR_PROCESS_START',
       'ORGANIZATION_TYPE'],
      dtype='object')
```

In [85]:

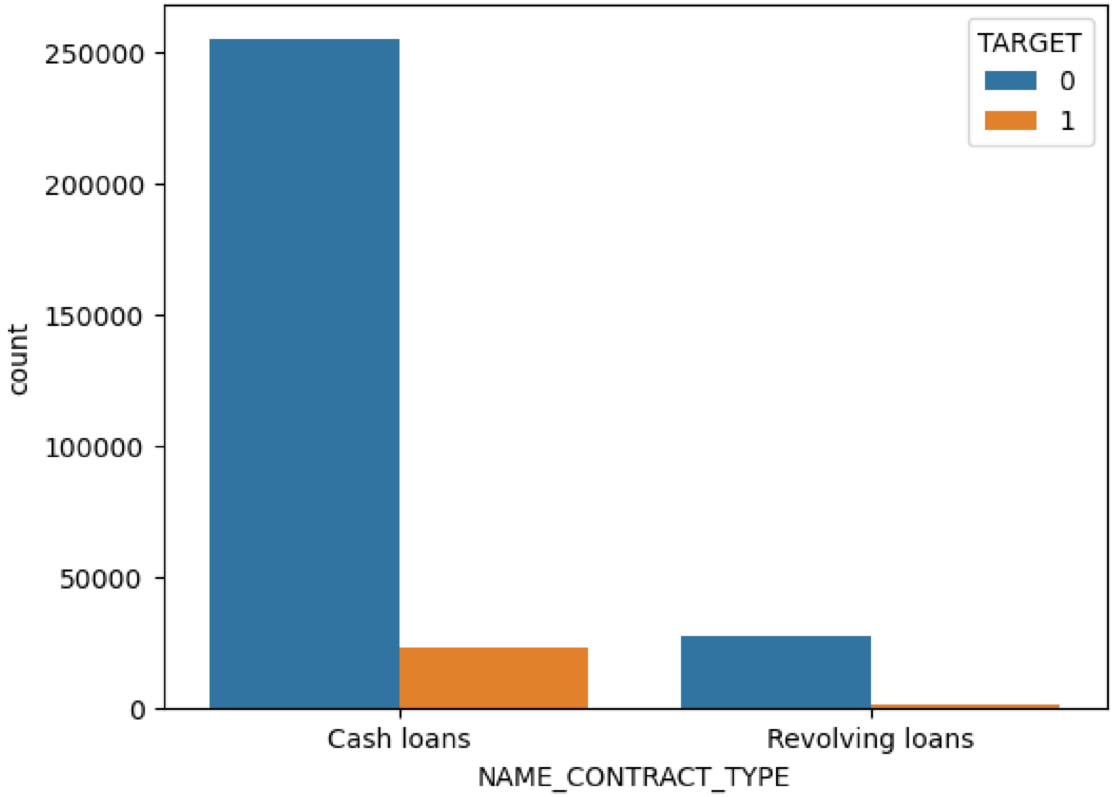
```
app_score_col_rmvd.groupby(["NAME_CONTRACT_TYPE"]).size()
```

Out[85]:

```
NAME_CONTRACT_TYPE
Cash loans         278232
Revolving loans     29279
dtype: int64
```

In [86]:

```
sns.countplot(x = "NAME_CONTRACT_TYPE", data = app_score_col_rmvd, hue = "TARGET")
plt.show()
```



In [87]:

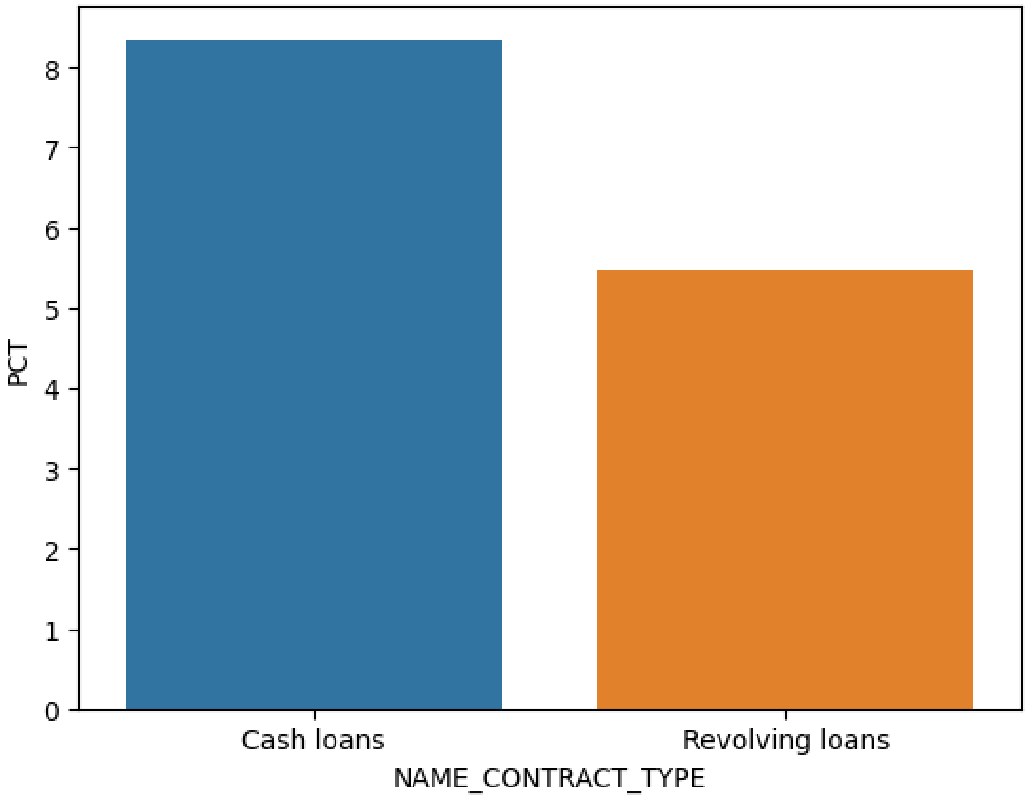
```
data_pct = app_score_col_rmvd[["NAME_CONTRACT_TYPE","TARGET"]].groupby(["NAME_CONTRACT_T
data_pct["PCT"] = data_pct["TARGET"] * 100
data_pct
```

Out[87]:

	NAME_CONTRACT_TYPE	TARGET	PCT
0	Cash loans	0.083459	8.345913
1	Revolving loans	0.054783	5.478329

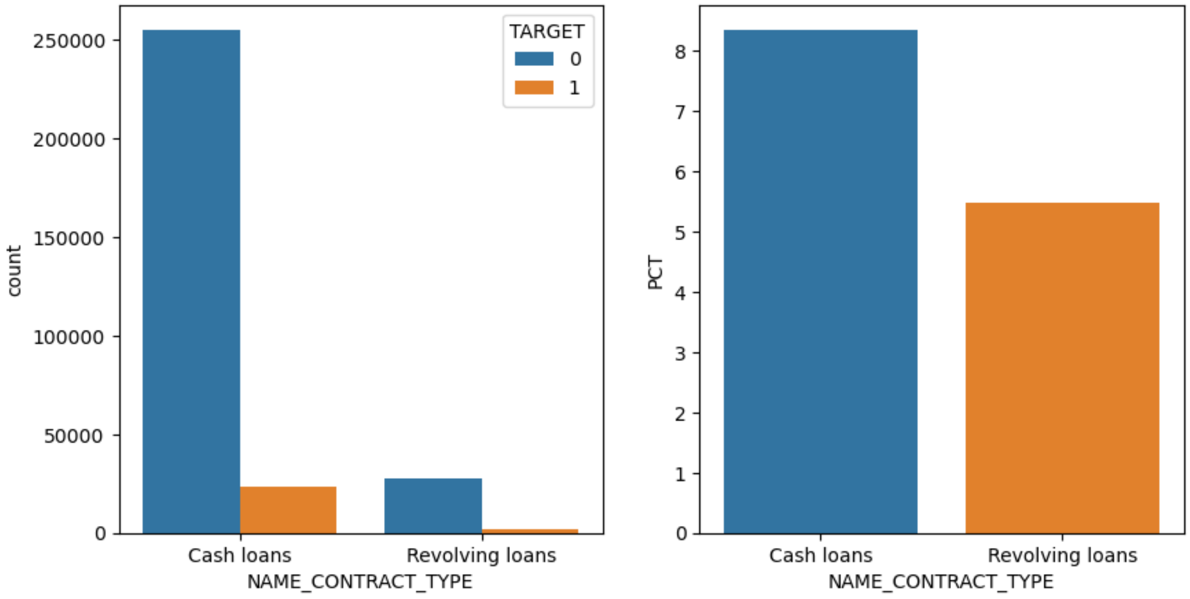
In [88]:

```
sns.barplot(x = "NAME_CONTRACT_TYPE", y = "PCT", data = data_pct)
plt.show()
```



In [89]:

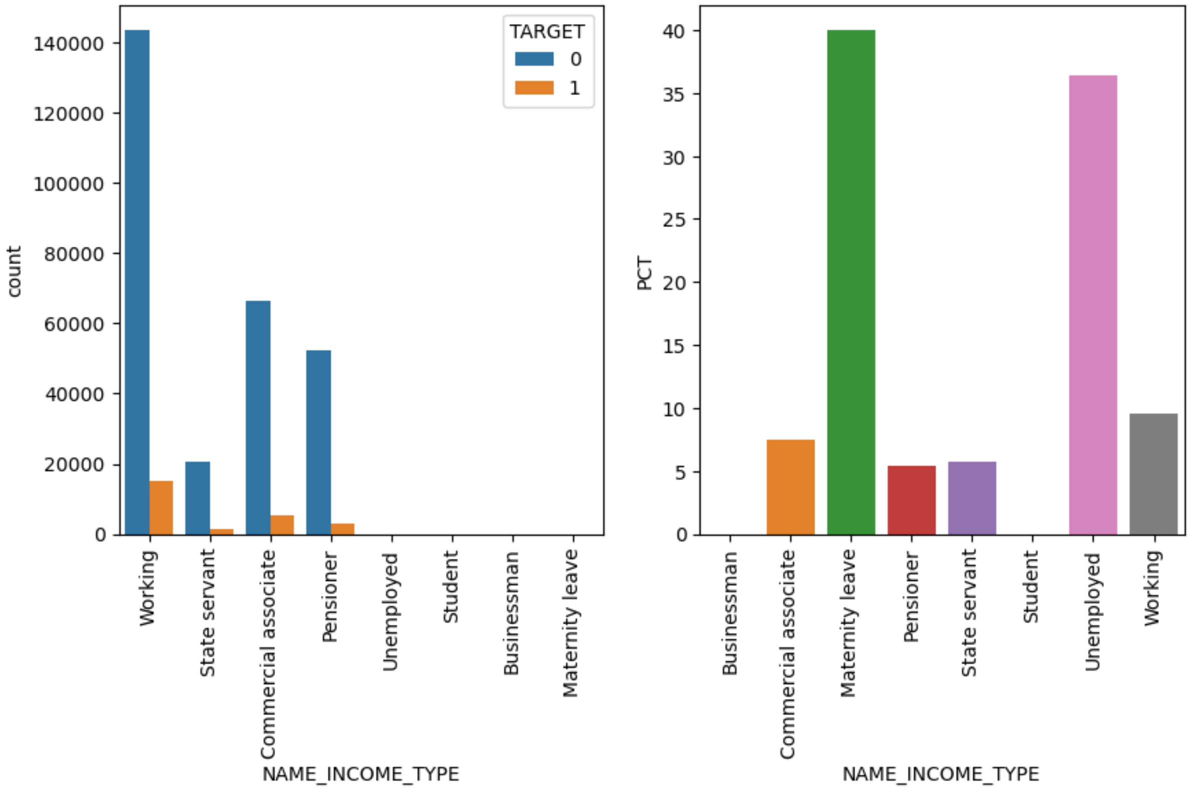
```
plt.figure(figsize = (10,5))

plt.subplot(1,2,1)
sns.countplot(x = "NAME_CONTRACT_TYPE", data = app_score_col_rmvd, hue = "TARGET")

plt.subplot(1,2,2)
sns.barplot(x = "NAME_CONTRACT_TYPE", y = "PCT", data = data_pct)

plt.show()
```



In [90]:

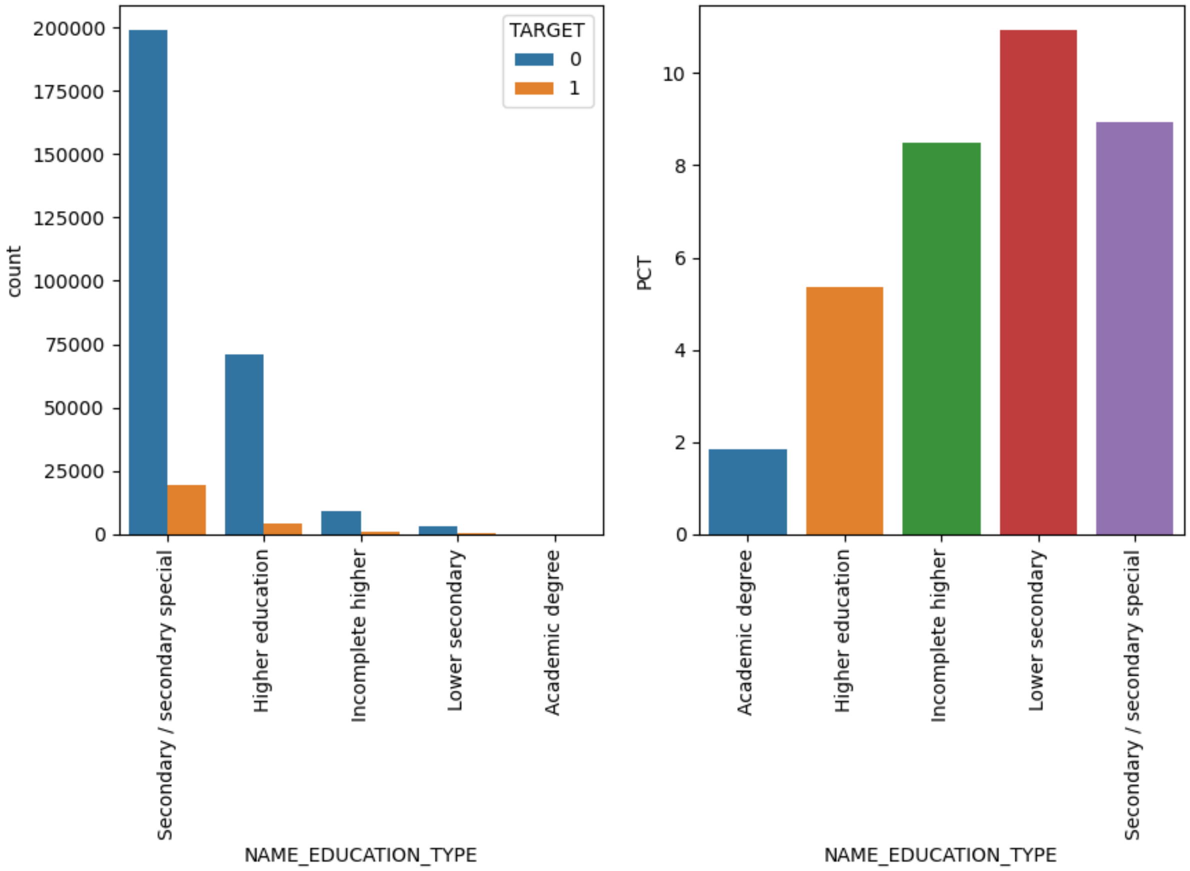
```
data_pct = app_score_col_rmvd[["NAME_INCOME_TYPE","TARGET"]].groupby(["NAME_INCOME_TYPE"
data_pct["PCT"] = data_pct["TARGET"] * 100

plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
sns.countplot(x = "NAME_INCOME_TYPE", data = app_score_col_rmvd, hue = "TARGET")
plt.xticks(rotation = 90)
plt.subplot(1,2,2)
sns.barplot(x = "NAME_INCOME_TYPE", y = "PCT", data = data_pct)
```

```
plt.xticks(rotation = 90)
plt.show()
```



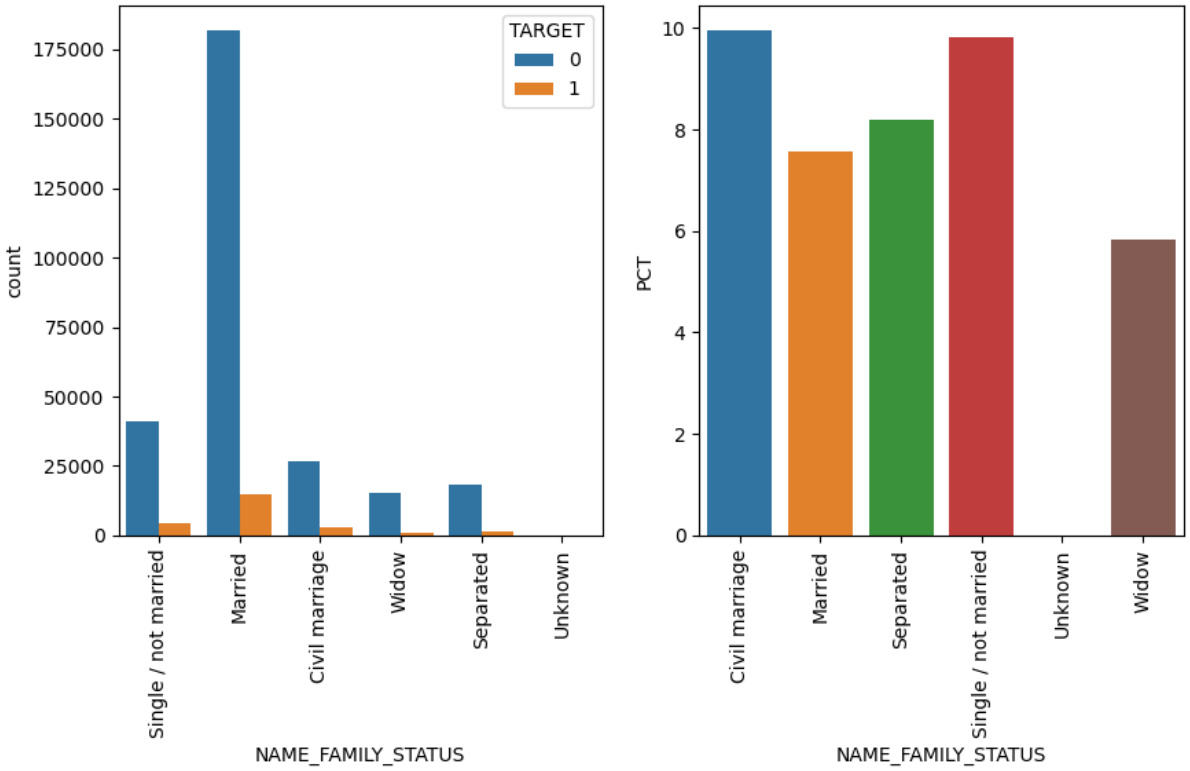
In [91]:

```
data_pct = app_score_col_rmvd[["NAME_EDUCATION_TYPE","TARGET"]].groupby(["NAME_EDUCATION
data_pct["PCT"] = data_pct["TARGET"] * 100

plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
sns.countplot(x = "NAME_EDUCATION_TYPE", data = app_score_col_rmvd, hue = "TARGET")
plt.xticks(rotation = 90)
plt.subplot(1,2,2)
sns.barplot(x = "NAME_EDUCATION_TYPE", y = "PCT", data = data_pct)
plt.xticks(rotation = 90)
plt.show()
```



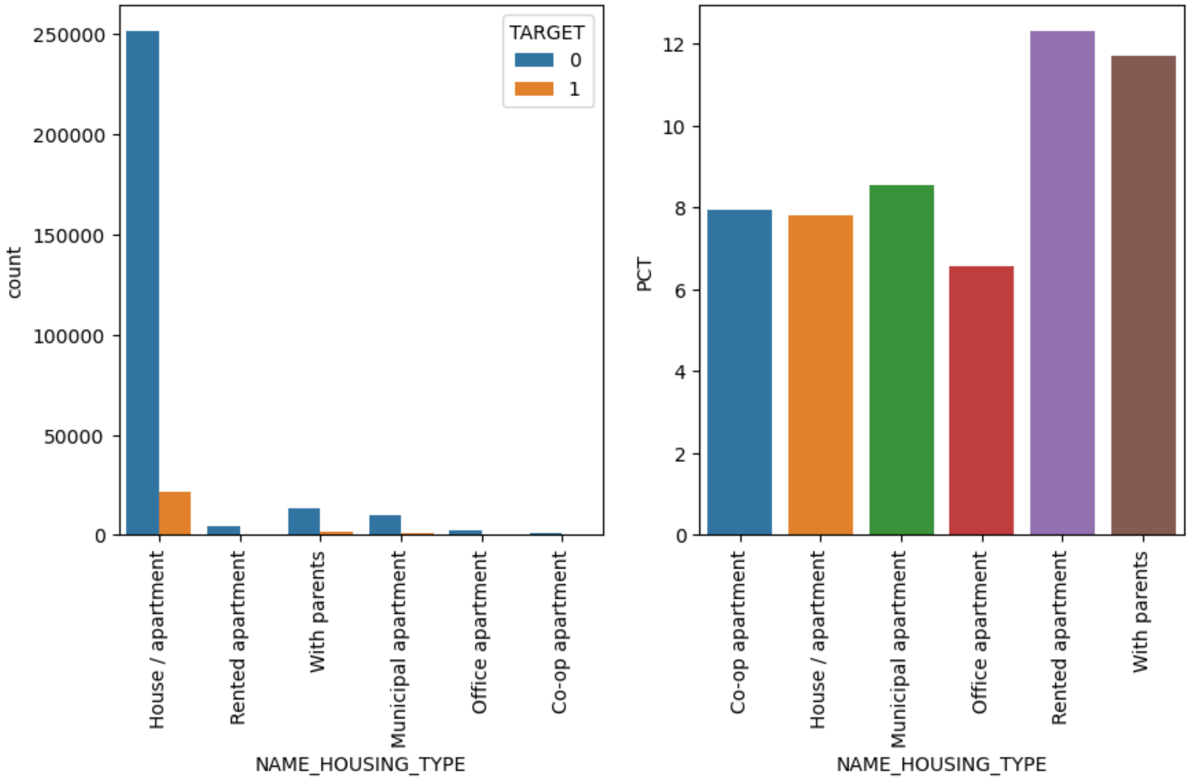
In [92]:

```
data_pct = app_score_col_rmvd[["NAME_FAMILY_STATUS","TARGET"]].groupby(["NAME_FAMILY_STA
data_pct["PCT"] = data_pct["TARGET"] * 100

plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
sns.countplot(x = "NAME_FAMILY_STATUS", data = app_score_col_rmvd, hue = "TARGET")
plt.xticks(rotation = 90)
plt.subplot(1,2,2)
sns.barplot(x = "NAME_FAMILY_STATUS", y = "PCT", data = data_pct)
plt.xticks(rotation = 90)
plt.show()
```



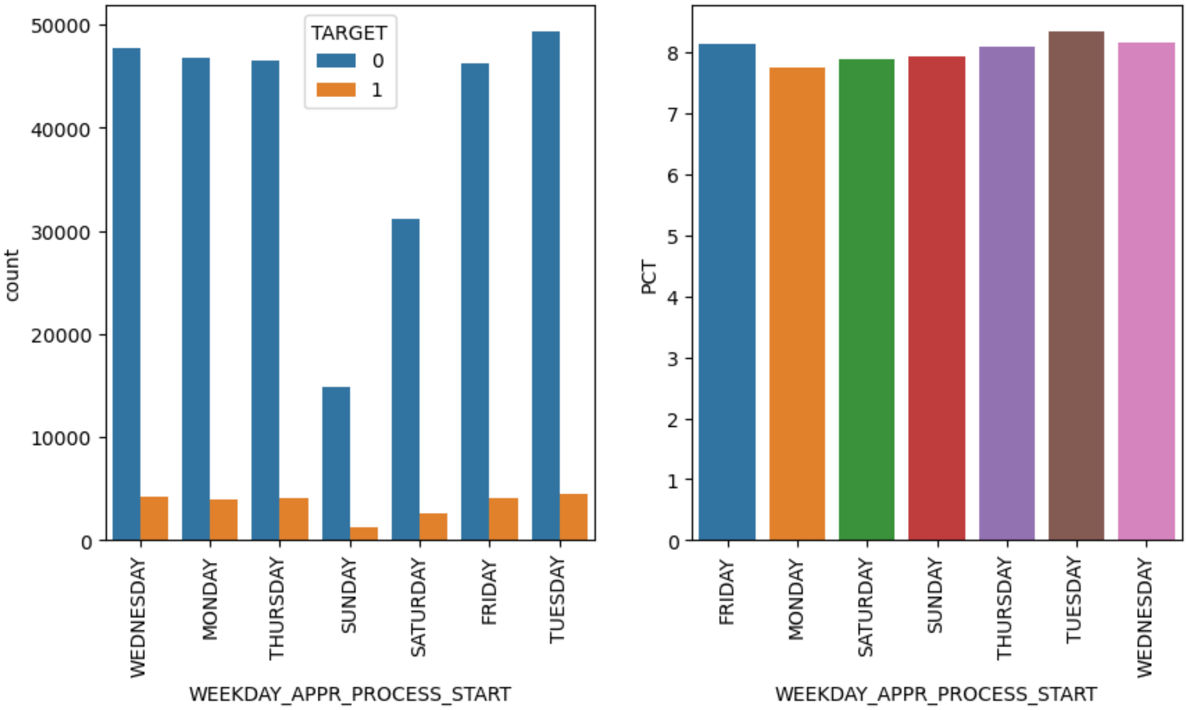
In [93]:

```
data_pct = app_score_col_rmvd[["NAME_HOUSING_TYPE","TARGET"]].groupby(["NAME_HOUSING_TYP
data_pct["PCT"] = data_pct["TARGET"] * 100

plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
sns.countplot(x = "NAME_HOUSING_TYPE", data = app_score_col_rmvd, hue = "TARGET")
plt.xticks(rotation = 90)
plt.subplot(1,2,2)
sns.barplot(x = "NAME_HOUSING_TYPE", y = "PCT", data = data_pct)
plt.xticks(rotation = 90)
plt.show()
```



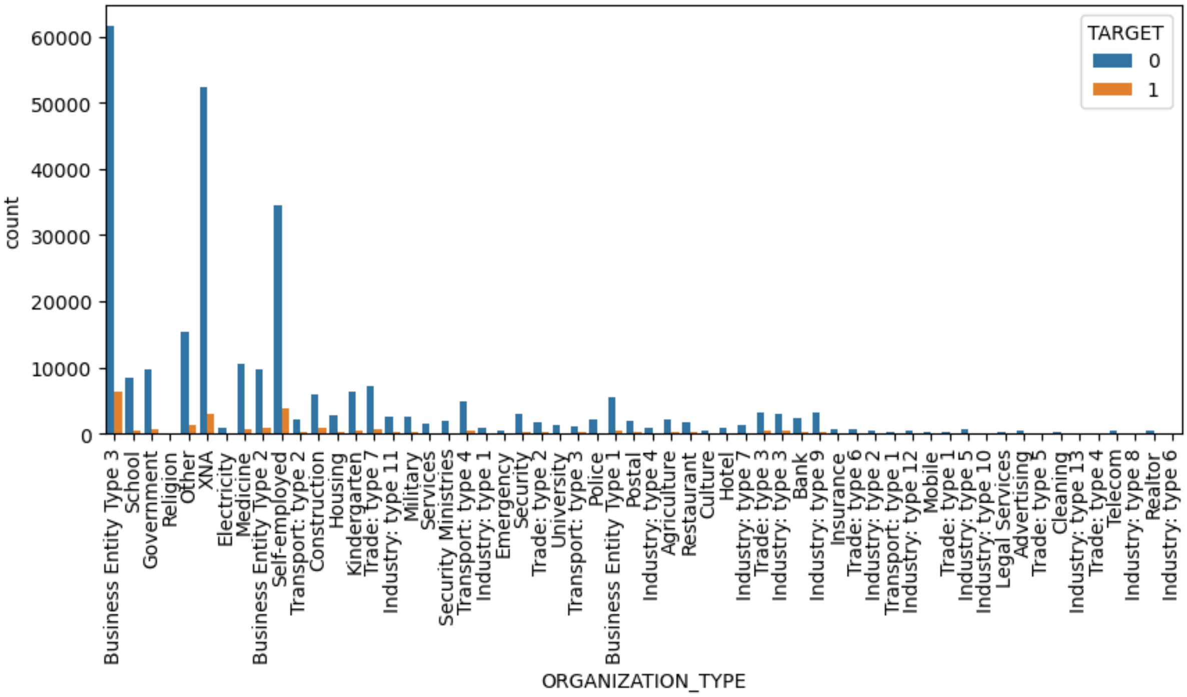
In [94]:

```
data_pct = app_score_col_rmvd[["WEEKDAY_APPR_PROCESS_START","TARGET"]].groupby(["WEEKDAY
data_pct["PCT"] = data_pct["TARGET"] * 100

plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
sns.countplot(x = "WEEKDAY_APPR_PROCESS_START", data = app_score_col_rmvd, hue = "TARGET
plt.xticks(rotation = 90)
plt.subplot(1,2,2)
sns.barplot(x = "WEEKDAY_APPR_PROCESS_START", y = "PCT", data = data_pct)
plt.xticks(rotation = 90)
plt.show()
```



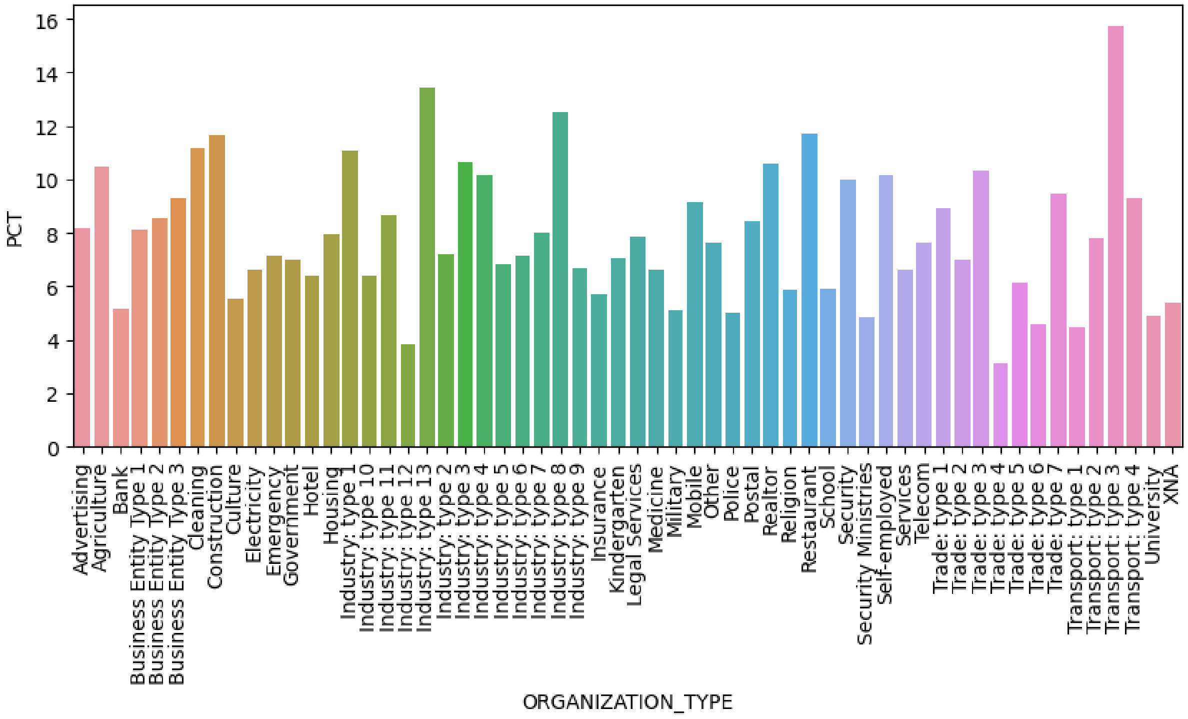
In [95]:

```
data_pct = app_score_col_rmvd[["ORGANIZATION_TYPE","TARGET"]].groupby(["ORGANIZATION_TYP
data_pct["PCT"] = data_pct["TARGET"] * 100

plt.figure(figsize = (10,4))
sns.countplot(x = "ORGANIZATION_TYPE", data = app_score_col_rmvd, hue = "TARGET")
plt.xticks(rotation = 90)
plt.show()
```



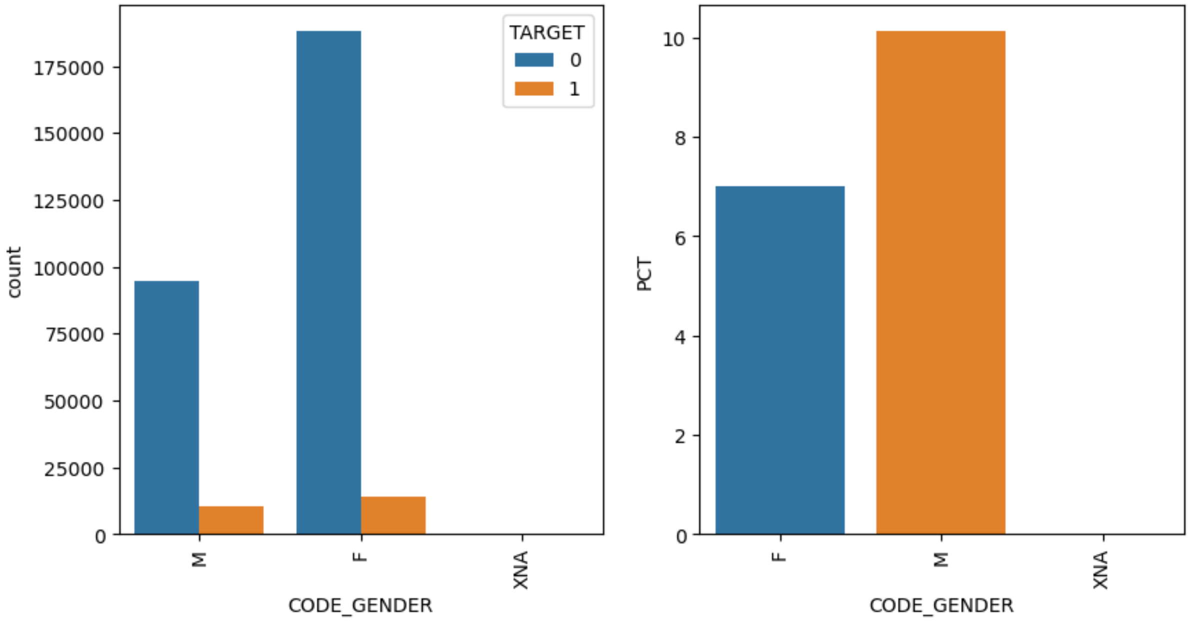
In [96]:

```
plt.figure(figsize = (10,4))
sns.barplot(x = "ORGANIZATION_TYPE", y = "PCT", data = data_pct)
plt.xticks(rotation = 90)
plt.show()
```



In [97]:

```
data_pct = app_score_col_rmvd[["CODE_GENDER","TARGET"]].groupby(["CODE_GENDER"], as_inde
data_pct["PCT"] = data_pct["TARGET"] * 100

plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
sns.countplot(x = "CODE_GENDER", data = app_score_col_rmvd, hue = "TARGET")
plt.xticks(rotation = 90)
plt.subplot(1,2,2)
sns.barplot(x = "CODE_GENDER", y = "PCT", data = data_pct)
plt.xticks(rotation = 90)
plt.show()
```



Univariate analysis

In [98]:

```
app_score_col_rmvd.dtypes.value_counts()
```

Out[98]:

```
float64     18
int64       15
object      10
category     1
category     1
category     1
category     1
category     1
category     1
dtype: int64
```

In [99]:

```
num_var = app_score_col_rmvd.select_dtypes(include = ['float64','int64']).columns
num_cat_var = app_score_col_rmvd.select_dtypes(include = ['float64','int64','category'])
```

In [100…

```
len(num_var)
```

Out[100]:

```
33
```

In [101…

```
len(num_cat_var)
```

Out[101]:

```
39
```

In [102…

```
num_data = app_score_col_rmvd[num_var]
defaulters = num_data[num_data['TARGET'] == 1]
repayers = num_data[num_data['TARGET'] == 0]
```

In [103…

```
defaulters.head()
```

Out[103]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GO
0	100002	1	0	202500.0	406597.5	24700.5	
26	100031	1	0	112500.0	979992.0	27076.5	
40	100047	1	0	202500.0	1193580.0	35028.0	
42	100049	1	0	135000.0	288873.0	16258.5	
81	100096	1	0	81000.0	252000.0	14593.5	

In [104…

```
repayers.head()
```

Out[104]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOC
1	100003	0	0	270000.0	1293502.5	35698.5	
2	100004	0	0	67500.0	135000.0	6750.0	
3	100006	0	0	135000.0	312682.5	29686.5	
4	100007	0	0	121500.0	513000.0	21865.5	
5	100008	0	0	99000.0	490495.5	27517.5	

In [105…

```
defaulters[['SK_ID_CURR', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL']].corr()
```

Out[105]:

	SK_ID_CURR	CNT_CHILDREN	AMT_INCOME_TOTAL
SK_ID_CURR	1.000000	-0.005144	-0.010165
CNT_CHILDREN	-0.005144	1.000000	0.004796
AMT_INCOME_TOTAL	-0.010165	0.004796	1.000000

In [106…

```
defaulter_corr = defaulters.corr()
defaulter_corr_unstck = defaulter_corr.where(np.triu(np.ones(defaulter_corr.shape), k =
                            (bool)).unstack().reset_index().rename(columns={'level_0':'v

defaulter_corr_unstck['corr'] = abs(defaulter_corr_unstck['corr'])
defaulter_corr_unstck.dropna(subset=['corr']).sort_values(by = ['corr'], ascending = Fal
```

Out[106]:

	var1	var2	corr
814	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	0.998269
202	AMT_GOODS_PRICE	AMT_CREDIT	0.982783
475	REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	0.956637
398	CNT_FAM_MEMBERS	CNT_CHILDREN	0.885484
848	DEF_60_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	0.868994
611	LIVE_REGION_NOT_WORK_REGION	REG_REGION_NOT_WORK_REGION	0.847885
713	LIVE_CITY_NOT_WORK_CITY	REG_CITY_NOT_WORK_CITY	0.778540
203	AMT_GOODS_PRICE	AMT_ANNUITY	0.752295
169	AMT_ANNUITY	AMT_CREDIT	0.752195
305	DAYS_EMPLOYED	DAYS_BIRTH	0.582185

In [107…

```
repayers_corr = repayers.corr()
repayers_corr_unstck = repayers_corr.where(np.triu(np.ones(repayers_corr.shape), k = 1).
                          (bool)).unstack().reset_index().rename(columns={'level_0':'va

repayers_corr_unstck['corr'] = abs(repayers_corr_unstck['corr'])
repayers_corr_unstck.dropna(subset=['corr']).sort_values(by = ['corr'], ascending = Fals
```

Out[107]:

	var1	var2	corr
814	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	0.998508
202	AMT_GOODS_PRICE	AMT_CREDIT	0.987022
475	REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	0.950149
398	CNT_FAM_MEMBERS	CNT_CHILDREN	0.878571
611	LIVE_REGION_NOT_WORK_REGION	REG_REGION_NOT_WORK_REGION	0.861861
848	DEF_60_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	0.859332
713	LIVE_CITY_NOT_WORK_CITY	REG_CITY_NOT_WORK_CITY	0.830381
203	AMT_GOODS_PRICE	AMT_ANNUITY	0.776421
169	AMT_ANNUITY	AMT_CREDIT	0.771297
305	DAYS_EMPLOYED	DAYS_BIRTH	0.626114

In [108…

```
num_data.head()
```

Out[108]:

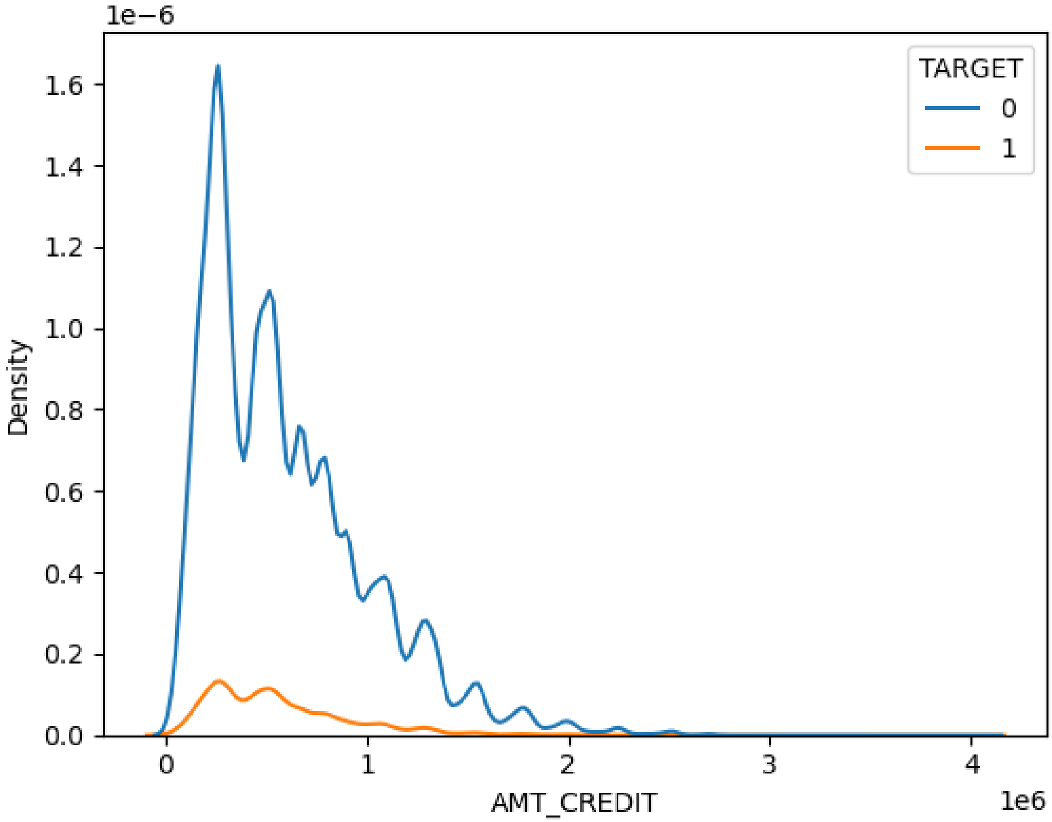
	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOC
0	100002	1	0	202500.0	406597.5	24700.5	
1	100003	0	0	270000.0	1293502.5	35698.5	
2	100004	0	0	67500.0	135000.0	6750.0	
3	100006	0	0	135000.0	312682.5	29686.5	
4	100007	0	0	121500.0	513000.0	21865.5	

In [109…

```
amt_var = ['AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AMT_ANNUITY', 'AMT_GOODS_PRICE']
```

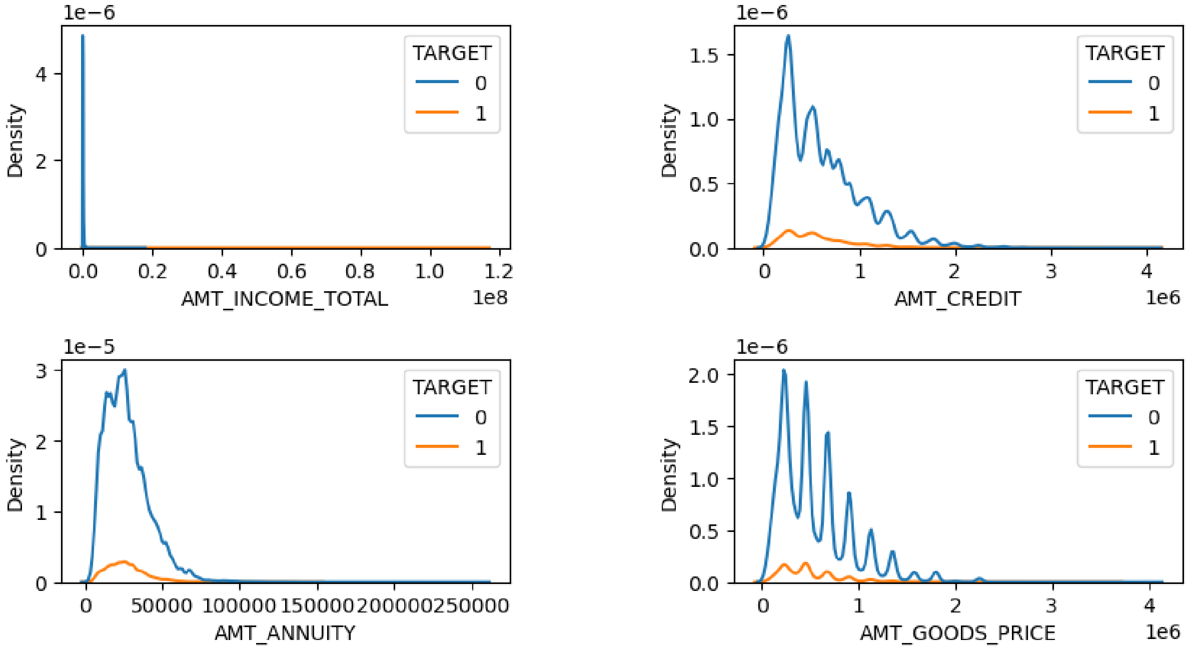
In [110…

```
sns.kdeplot(x = 'AMT_CREDIT', data = num_data, hue = 'TARGET')
plt.show()
```



In [111…

```
plt.figure(figsize = (10,5))

for i, col in enumerate(amt_var):
    plt.subplot(2, 2, i+1)
    sns.kdeplot(x = col, data = num_data, hue = 'TARGET')
    plt.subplots_adjust(wspace = 0.5, hspace = 0.5)
```



Bivariate analysis

In [112…

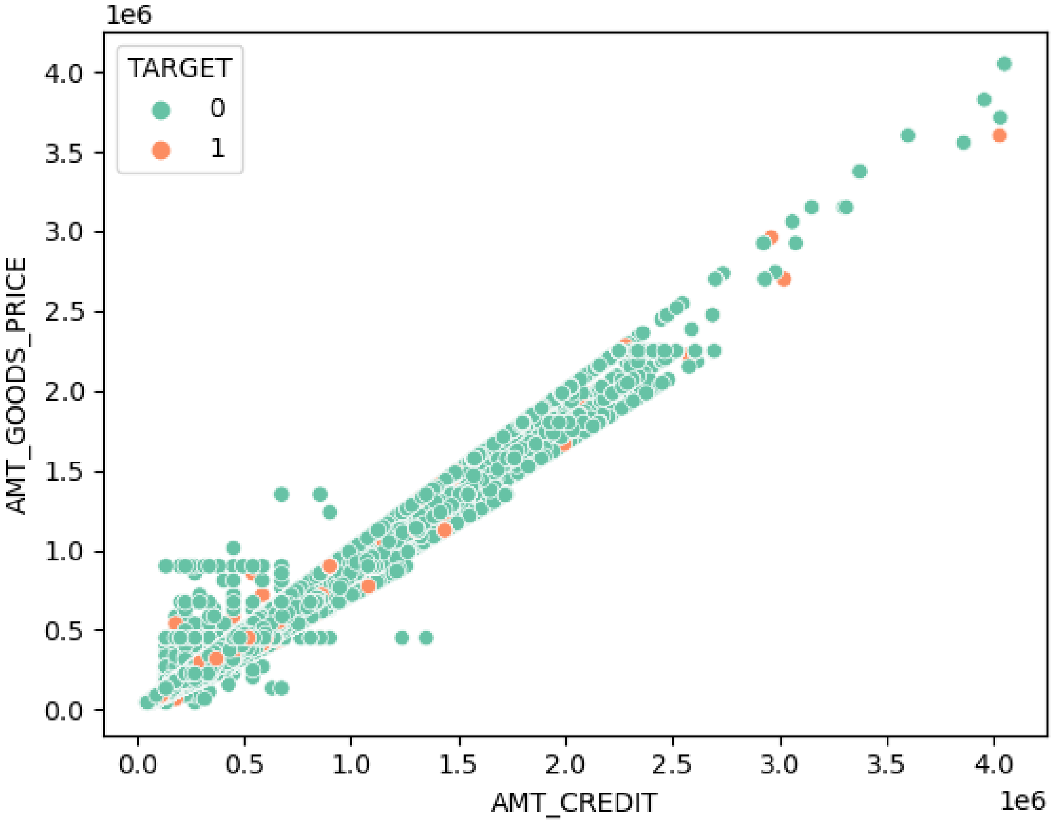
```
num_data.head()
```

Out[112]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOC
0	100002	1	0	202500.0	406597.5	24700.5	
1	100003	0	0	270000.0	1293502.5	35698.5	
2	100004	0	0	67500.0	135000.0	6750.0	
3	100006	0	0	135000.0	312682.5	29686.5	
4	100007	0	0	121500.0	513000.0	21865.5	

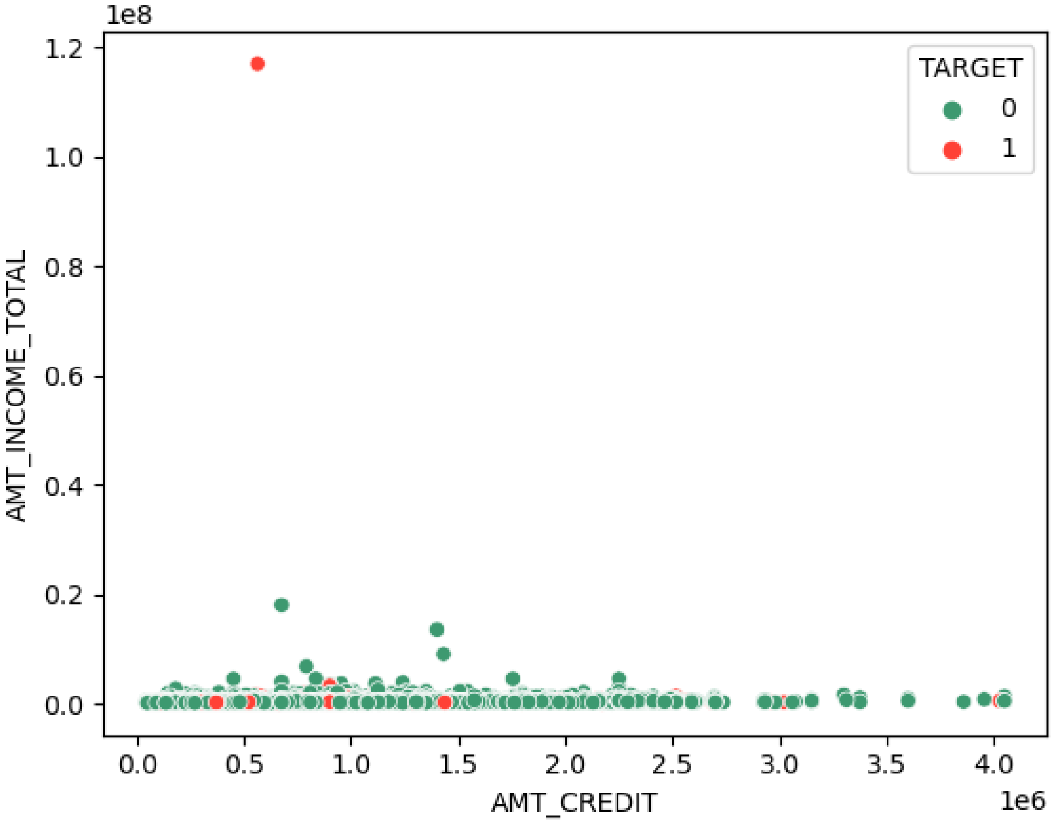
In [113…

```
sns.scatterplot(x = 'AMT_CREDIT', y = 'AMT_GOODS_PRICE', data = num_data, hue = 'TARGET'
plt.show()
```



In [114…

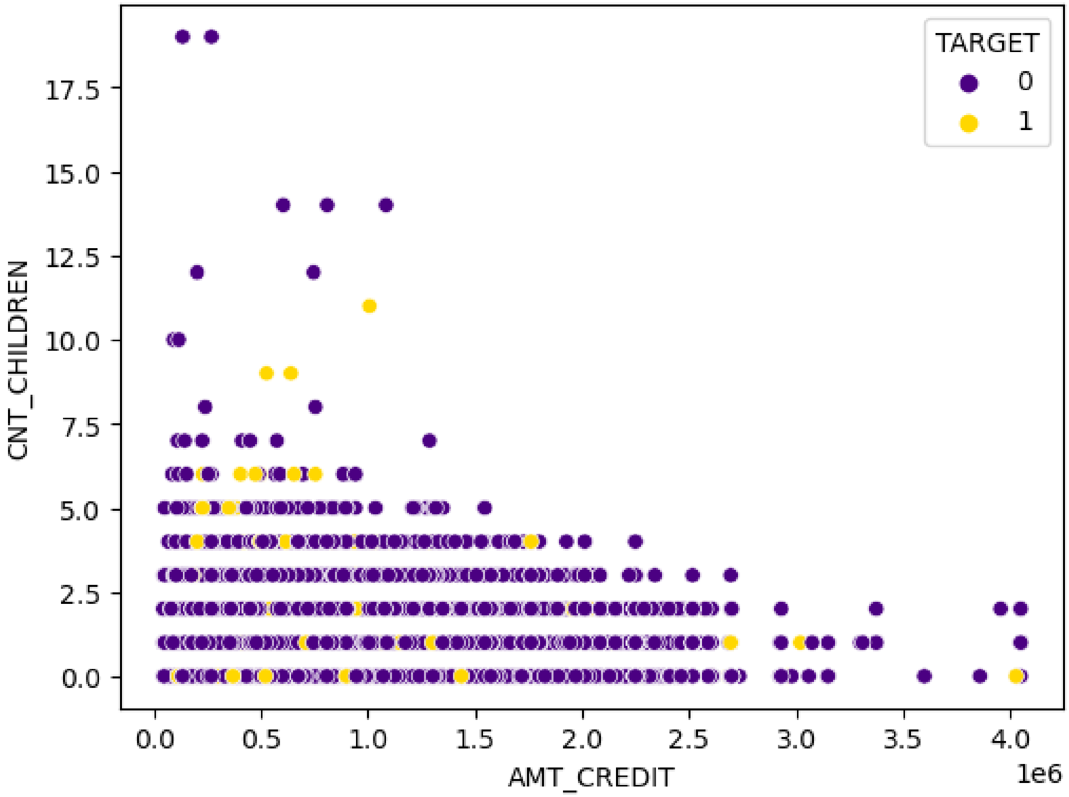
```
color1 = "#3D9970"
color2 = "#FF4136"
color_palette = [color1, color2]
sns.scatterplot(x = 'AMT_CREDIT', y = 'AMT_INCOME_TOTAL', data = num_data, hue = 'TARGET
plt.show()
```



In [115…

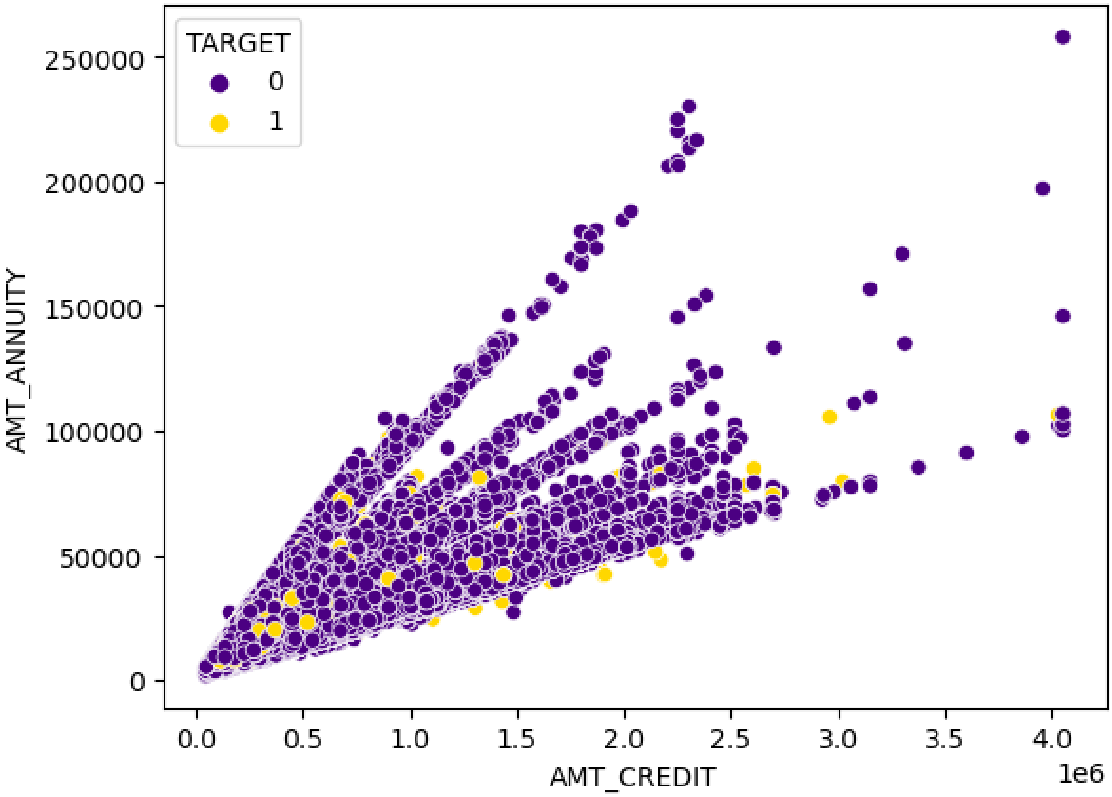
```
color1 = "#4B0082"
color2 = "#FFD700"
color_palette = [color1, color2]
```

```
sns.scatterplot(x = 'AMT_CREDIT', y = 'CNT_CHILDREN', data = num_data, hue = 'TARGET', p
plt.show()
```



In [116…

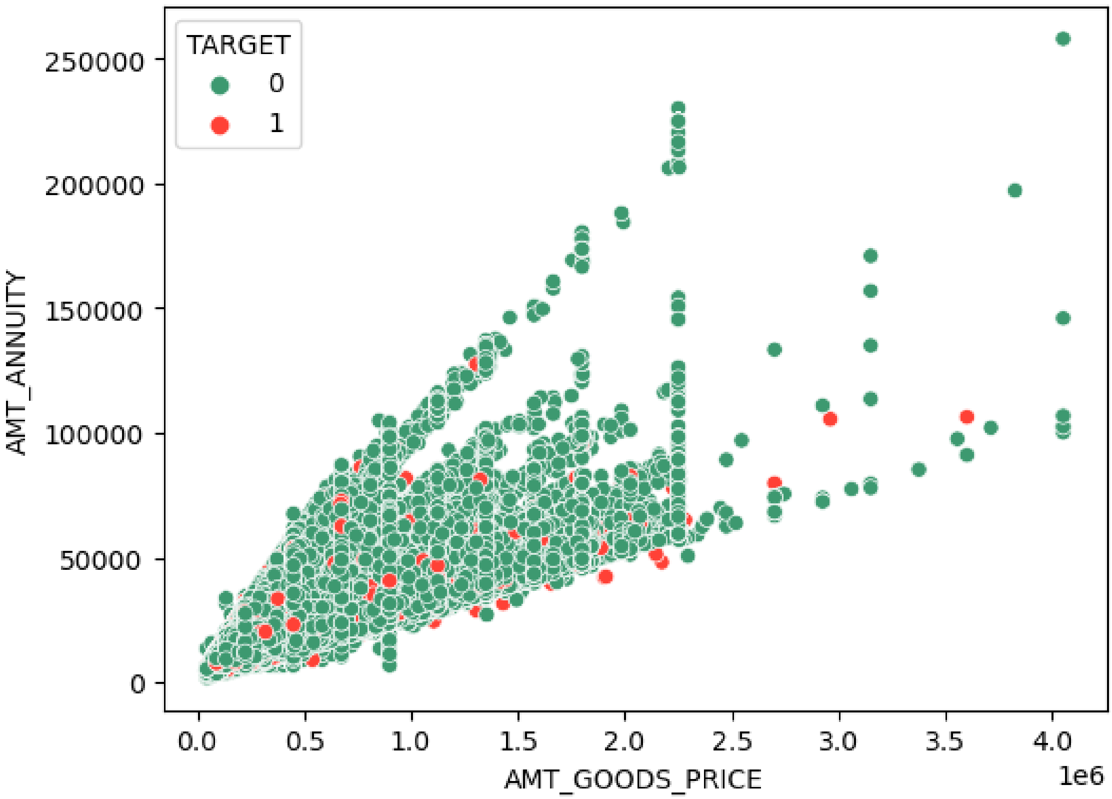
```
color1 = "#4B0082"
color2 = "#FFD700"
color_palette = [color1, color2]
sns.scatterplot(x = 'AMT_CREDIT', y = 'AMT_ANNUITY', data = num_data, hue = 'TARGET', pa
plt.show()
```



In [117…

```
color1 = "#3D9970"
color2 = "#FF4136"
color_palette = [color1, color2]
```

```
sns.scatterplot(x = 'AMT_GOODS_PRICE', y = 'AMT_ANNUITY', data = num_data, hue = 'TARGET
plt.show()
```



Exploring the previous_application dataset

```
In [118…
null_count = pd.DataFrame(prev_app.isnull().sum().sort_values(ascending=False
                       )/prev_app.shape[0]*100).reset_index().rename(columns={'index':'v
```

```
In [119…
var_msng_ge_40 = list(null_count[null_count['count_pct'] >= 40]['var'])
var_msng_ge_40
```

```
Out[119]:
['RATE_INTEREST_PRIVILEGED',
 'RATE_INTEREST_PRIMARY',
 'AMT_DOWN_PAYMENT',
 'RATE_DOWN_PAYMENT',
 'NAME_TYPE_SUITE',
 'NFLAG_INSURED_ON_APPROVAL',
 'DAYS_TERMINATION',
 'DAYS_LAST_DUE',
 'DAYS_LAST_DUE_1ST_VERSION',
 'DAYS_FIRST_DUE',
 'DAYS_FIRST_DRAWING']
```

```
In [120…
nva_cols = var_msng_ge_40+['WEEKDAY_APPR_PROCESS_START', 'HOUR_APPR_PROCESS_START',
                           'FLAG_LAST_APPL_PER_CONTRACT', 'NFLAG_LAST_APPL_IN_DAY']
nva_cols
```

Out[120]:

```
['RATE_INTEREST_PRIVILEGED',
 'RATE_INTEREST_PRIMARY',
 'AMT_DOWN_PAYMENT',
 'RATE_DOWN_PAYMENT',
 'NAME_TYPE_SUITE',
 'NFLAG_INSURED_ON_APPROVAL',
 'DAYS_TERMINATION',
 'DAYS_LAST_DUE',
 'DAYS_LAST_DUE_1ST_VERSION',
 'DAYS_FIRST_DUE',
 'DAYS_FIRST_DRAWING',
 'WEEKDAY_APPR_PROCESS_START',
 'HOUR_APPR_PROCESS_START',
 'FLAG_LAST_APPL_PER_CONTRACT',
 'NFLAG_LAST_APPL_IN_DAY']
```

In [121…

```
len(nva_cols)
```

Out[121]:

```
15
```

In [122…

```
len(prev_app.columns)
```

Out[122]:

```
37
```

In [123…

```
prev_app_nva_col_rmvd = prev_app.drop(labels = nva_cols ,axis = 1)

len(prev_app_nva_col_rmvd.columns)
```

Out[123]:

```
22
```

In [124…

```
prev_app_nva_col_rmvd.columns
```

Out[124]:

```
Index(['SK_ID_PREV', 'SK_ID_CURR', 'NAME_CONTRACT_TYPE', 'AMT_ANNUITY',
       'AMT_APPLICATION', 'AMT_CREDIT', 'AMT_GOODS_PRICE',
       'NAME_CASH_LOAN_PURPOSE', 'NAME_CONTRACT_STATUS', 'DAYS_DECISION',
       'NAME_PAYMENT_TYPE', 'CODE_REJECT_REASON', 'NAME_CLIENT_TYPE',
       'NAME_GOODS_CATEGORY', 'NAME_PORTFOLIO', 'NAME_PRODUCT_TYPE',
       'CHANNEL_TYPE', 'SELLERPLACE_AREA', 'NAME_SELLER_INDUSTRY',
       'CNT_PAYMENT', 'NAME_YIELD_GROUP', 'PRODUCT_COMBINATION'],
      dtype='object')
```

In [125…

```
prev_app_nva_col_rmvd.head()
```

Out[125]:

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

In [126…

```
prev_app_nva_col_rmvd.isnull().sum().sort_values(ascending = False) / prev_app_nva_col_r
```

```
Out[126]: AMT_GOODS_PRICE           23.081773
          AMT_ANNUITY               22.286665
          CNT_PAYMENT               22.286366
          PRODUCT_COMBINATION        0.020716
          AMT_CREDIT                 0.000060
          NAME_GOODS_CATEGORY        0.000000
          NAME_YIELD_GROUP           0.000000
          NAME_SELLER_INDUSTRY       0.000000
          SELLERPLACE_AREA           0.000000
          CHANNEL_TYPE               0.000000
          NAME_PRODUCT_TYPE          0.000000
          NAME_PORTFOLIO             0.000000
          SK_ID_PREV                 0.000000
          NAME_CLIENT_TYPE           0.000000
          SK_ID_CURR                 0.000000
          NAME_PAYMENT_TYPE          0.000000
          DAYS_DECISION              0.000000
          NAME_CONTRACT_STATUS       0.000000
          NAME_CASH_LOAN_PURPOSE     0.000000
          AMT_APPLICATION            0.000000
          NAME_CONTRACT_TYPE         0.000000
          CODE_REJECT_REASON         0.000000
          dtype: float64
```

```
In [127…  prev_app_nva_col_rmvd['AMT_GOODS_PRICE'].agg(func=['mean', 'median'])
```

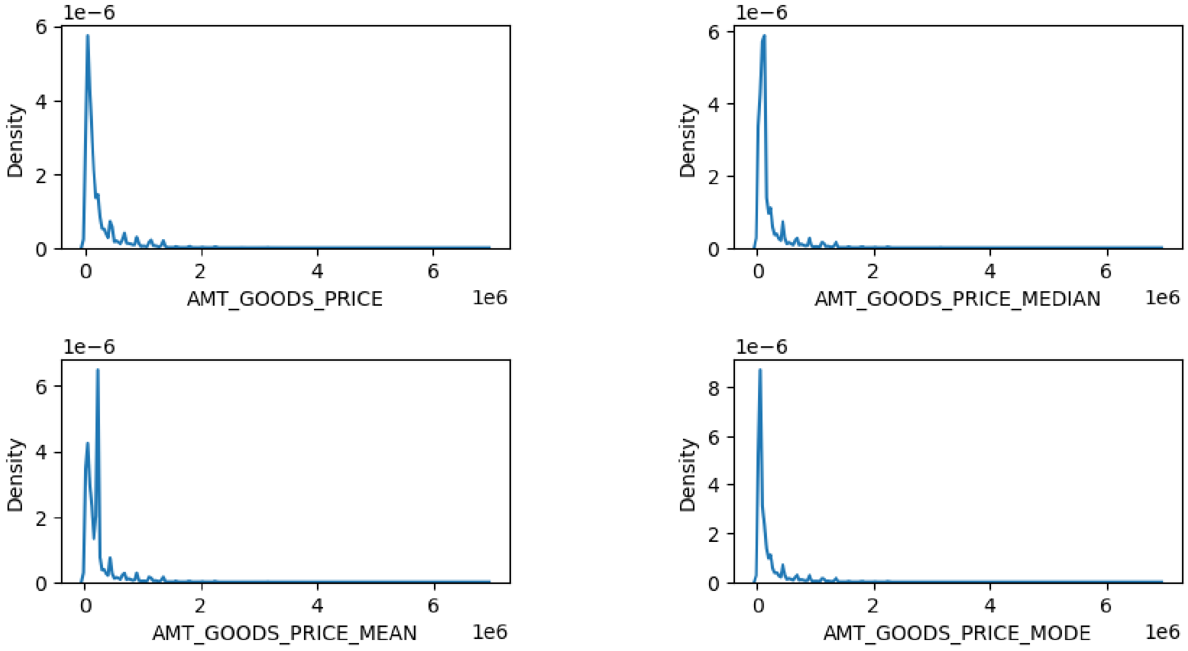
```
Out[127]: mean      227847.279283
          median    112320.000000
          Name: AMT_GOODS_PRICE, dtype: float64
```

```
In [128…  prev_app_nva_col_rmvd['AMT_GOODS_PRICE_MEDIAN'] = prev_app_nva_col_rmvd['AMT_GOODS_PRICE
              prev_app_nva_col_rmvd['AMT_GOODS_PRICE'].median())
```

```
In [129…  prev_app_nva_col_rmvd['AMT_GOODS_PRICE_MEAN'] = prev_app_nva_col_rmvd['AMT_GOODS_PRICE']
              prev_app_nva_col_rmvd['AMT_GOODS_PRICE'].mean())
```

```
In [130…  prev_app_nva_col_rmvd['AMT_GOODS_PRICE_MODE'] = prev_app_nva_col_rmvd['AMT_GOODS_PRICE']
              prev_app_nva_col_rmvd['AMT_GOODS_PRICE'].mode()[0])
```

```
In [131…  gp_cols = ['AMT_GOODS_PRICE','AMT_GOODS_PRICE_MEDIAN','AMT_GOODS_PRICE_MEAN','AMT_GOODS_

          plt.figure(figsize=(10,5))

          for i, col in enumerate(gp_cols):
              plt.subplot(2,2,i+1)
              sns.kdeplot(data=prev_app_nva_col_rmvd,x=col)
              plt.subplots_adjust(wspace=0.5,hspace=0.5)
```



In [132…

```
prev_app_nva_col_rmvd['AMT_GOODS_PRICE'] = prev_app_nva_col_rmvd['AMT_GOODS_PRICE'].fill
    prev_app_nva_col_rmvd['AMT_GOODS_PRICE'].median())
```

In [133…

```
prev_app_nva_col_rmvd['AMT_GOODS_PRICE'].isnull().sum()
```

Out[133]:

```
0
```

In [134…

```
prev_app_nva_col_rmvd['AMT_ANNUITY'].agg(func=['mean', 'median', 'max'])
```

Out[134]:

```
mean       15955.120659
median     11250.000000
max       418058.145000
Name: AMT_ANNUITY, dtype: float64
```

In [135…

```
prev_app_nva_col_rmvd['AMT_ANNUITY'] = prev_app_nva_col_rmvd['AMT_ANNUITY'].fillna(
    prev_app_nva_col_rmvd['AMT_ANNUITY'].median())
```

In [136…

```
prev_app_nva_col_rmvd['PRODUCT_COMBINATION'].head()
```

Out[136]:

```
0    POS mobile with interest
1            Cash X-Sell: low
2           Cash X-Sell: high
3         Cash X-Sell: middle
4           Cash Street: high
Name: PRODUCT_COMBINATION, dtype: object
```

In [137…

```
prev_app_nva_col_rmvd['PRODUCT_COMBINATION'] = prev_app_nva_col_rmvd['PRODUCT_COMBINATIC
    prev_app_nva_col_rmvd['PRODUCT_COMBINATION'].mode()[0])
```

In [138…

```
prev_app_nva_col_rmvd['CNT_PAYMENT'].agg(func=['mean','median','max'])
```

Out[138]:

```
mean      16.054082
median    12.000000
max       84.000000
Name: CNT_PAYMENT, dtype: float64
```

In [139…

```
prev_app_nva_col_rmvd[prev_app_nva_col_rmvd['CNT_PAYMENT'].isnull()
                     ].groupby(['NAME_CONTRACT_STATUS']).size().sort_values(ascending =
```

Out[139]:

```
NAME_CONTRACT_STATUS
Canceled        305805
Refused          40897
Unused offer     25524
Approved             4
dtype: int64
```

In [140…

```
prev_app_nva_col_rmvd['CNT_PAYMENT'] = prev_app_nva_col_rmvd['CNT_PAYMENT'].fillna(0)
```

In [141…

```
prev_app_nva_col_rmvd.isnull().sum().sort_values(ascending = False)
```

Out[141]:

```
AMT_CREDIT              1
SK_ID_PREV              0
NAME_GOODS_CATEGORY     0
AMT_GOODS_PRICE_MEAN    0
AMT_GOODS_PRICE_MEDIAN  0
PRODUCT_COMBINATION     0
NAME_YIELD_GROUP        0
CNT_PAYMENT             0
NAME_SELLER_INDUSTRY    0
SELLERPLACE_AREA        0
CHANNEL_TYPE            0
NAME_PRODUCT_TYPE       0
NAME_PORTFOLIO          0
NAME_CLIENT_TYPE        0
SK_ID_CURR              0
CODE_REJECT_REASON      0
NAME_PAYMENT_TYPE       0
DAYS_DECISION           0
NAME_CONTRACT_STATUS    0
NAME_CASH_LOAN_PURPOSE  0
AMT_GOODS_PRICE         0
AMT_APPLICATION         0
AMT_ANNUITY             0
NAME_CONTRACT_TYPE      0
AMT_GOODS_PRICE_MODE    0
dtype: int64
```

In [142…

```
prev_app_nva_col_rmvd = prev_app_nva_col_rmvd.drop(labels=
              ['AMT_GOODS_PRICE_MEDIAN','AMT_GOODS_PRICE_MEAN','AMT_GOODS_PRICE_MODE'],
```

In [143…

```
prev_app_nva_col_rmvd.isnull().sum().sort_values(ascending = False)
```

Out[143]:

```
AMT_CREDIT              1
SK_ID_PREV              0
NAME_CLIENT_TYPE        0
NAME_YIELD_GROUP        0
CNT_PAYMENT             0
NAME_SELLER_INDUSTRY    0
SELLERPLACE_AREA        0
CHANNEL_TYPE            0
NAME_PRODUCT_TYPE       0
NAME_PORTFOLIO          0
NAME_GOODS_CATEGORY     0
CODE_REJECT_REASON      0
SK_ID_CURR              0
NAME_PAYMENT_TYPE       0
DAYS_DECISION           0
NAME_CONTRACT_STATUS    0
NAME_CASH_LOAN_PURPOSE  0
AMT_GOODS_PRICE         0
AMT_APPLICATION         0
AMT_ANNUITY             0
NAME_CONTRACT_TYPE      0
PRODUCT_COMBINATION     0
dtype: int64
```

In [144…

```
len(prev_app_nva_col_rmvd.columns)
```

Out[144]:

```
22
```

In [145…

```
prev_app_nva_col_rmvd.head()
```

Out[145]:

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

Merging two datasets

In [146…

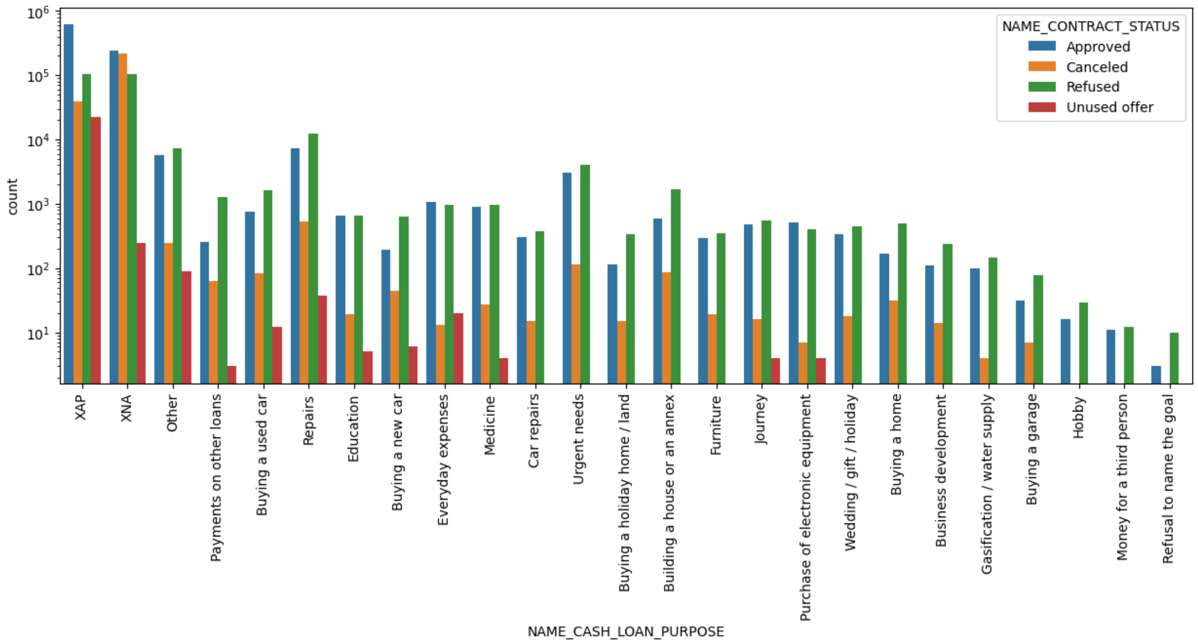
```
merged_df = pd.merge(app_score_col_rmvd, prev_app_nva_col_rmvd, how = 'inner', on = 'SK_
merged_df.head()
```

Out[146]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	CNT_CHILDREN	AMT_INCOME_TOTA
0	100002	1	Cash loans	M	0	202500.
1	100003	0	Cash loans	F	0	270000.
2	100003	0	Cash loans	F	0	270000.
3	100003	0	Cash loans	F	0	270000.
4	100004	0	Revolving loans	M	0	67500.

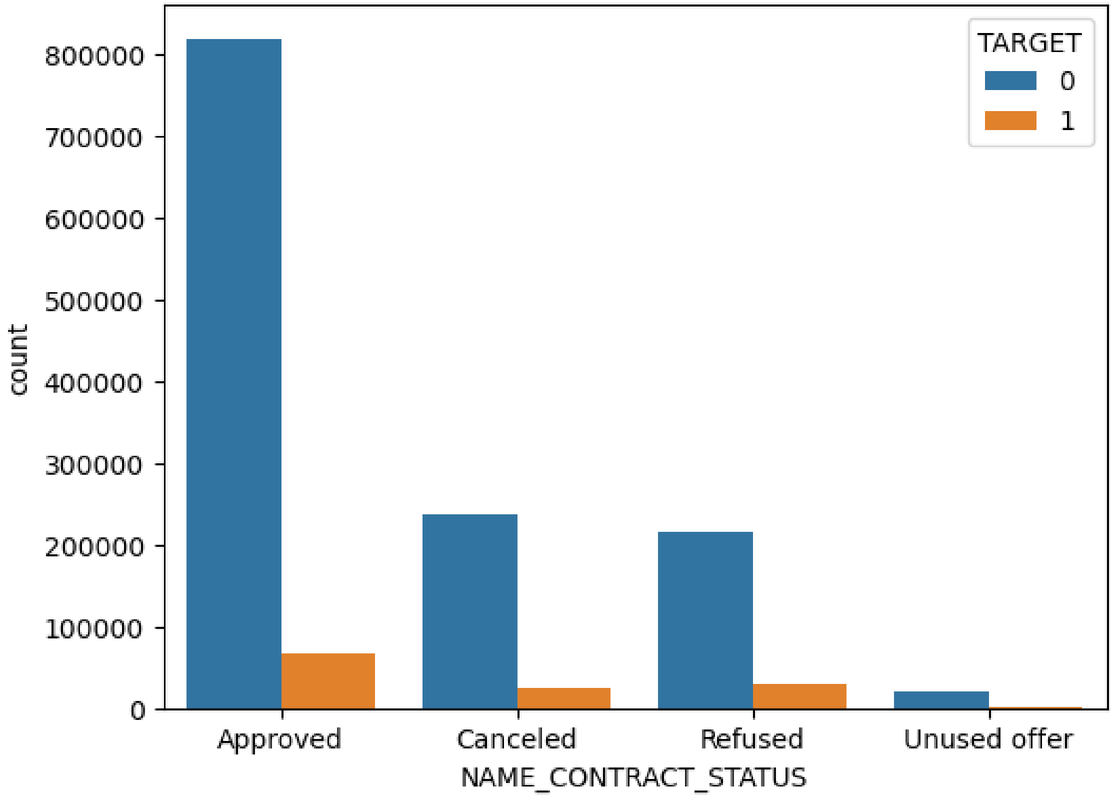
In [147…

```
plt.figure(figsize = (15,5))

sns.countplot(x = 'NAME_CASH_LOAN_PURPOSE', data = merged_df, hue = 'NAME_CONTRACT_STATU
plt.xticks(rotation = 90)
plt.yscale('log')
```



In [148…

```
sns.countplot(x = 'NAME_CONTRACT_STATUS', data = merged_df, hue = 'TARGET')
plt.show()
```



In [149…

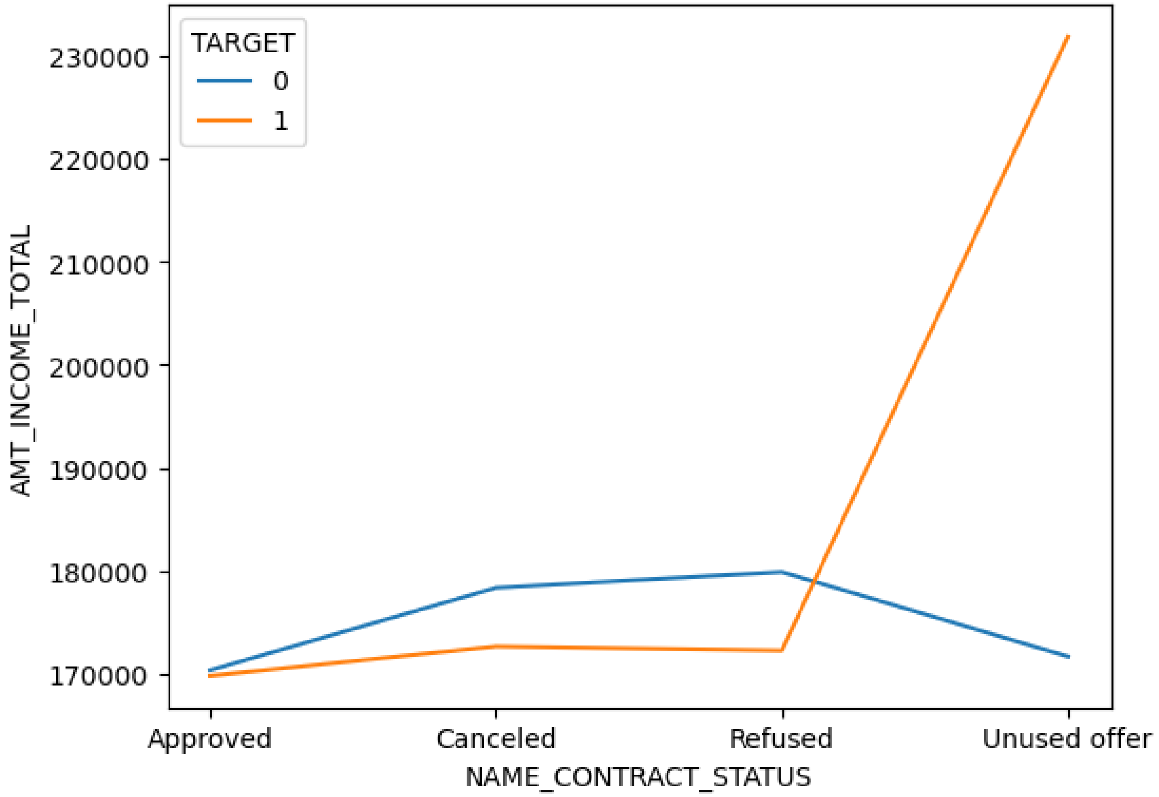
```
merged_agg = merged_df.groupby(['NAME_CONTRACT_STATUS', 'TARGET']).size().reset_index().
sum_df  = merged_agg.groupby(['NAME_CONTRACT_STATUS'])['counts'].sum().reset_index()

merged_agg_2 = pd.merge(merged_agg, sum_df, how = 'left', on = 'NAME_CONTRACT_STATUS')
merged_agg_2['pct'] = round(merged_agg_2['counts_x'] / merged_agg_2['counts_y']*100, 2)
merged_agg_2
```

Out[149]:

	NAME_CONTRACT_STATUS	TARGET	counts_x	counts_y	pct
0	Approved	0	818856	886099	92.41
1	Approved	1	67243	886099	7.59
2	Canceled	0	235641	259441	90.83
3	Canceled	1	23800	259441	9.17
4	Refused	0	215952	245390	88.00
5	Refused	1	29438	245390	12.00
6	Unused offer	0	20892	22771	91.75
7	Unused offer	1	1879	22771	8.25

In [150…

```
sns.lineplot(x = 'NAME_CONTRACT_STATUS', y = 'AMT_INCOME_TOTAL', data = merged_df, error
plt.show()
```



In [151…

```
len(merged_df.columns)
```

Out[151]:

```
70
```