```
import warnings
warnings.filterwarnings('ignore')
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
pd.set_option('display.max_columns', None)
```

```
#READ APPLICATION csv

app_data = pd.read_csv('application_data.csv')
app_data.head()
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_A
0	100002	1	Cash loans	M	N	Y	0	202500.0	406597.5	2
1	100003	0	Cash loans	F	N	N	0	270000.0	1293502.5	3
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	135000.0	
3	100006	0	Cash loans	F	N	Y	0	135000.0	312682.5	2
4	100007	0	Cash loans	M	N	Y	0	121500.0	513000.0	2

```
# Data Inspection on Application dataset
#get info and shape on the dataset

app_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3864 entries, 0 to 3863
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(85), int64(21), object(16)
memory usage: 3.6+ MB
```

```
#DATA QUALITY CHECK
# check for percentage null values in application dataset

pd.set_option('display.max_rows', None)
app_data.isnull().mean()*100
```

```
OBS_60_CNT_SOCIAL_CIRCLE         0.517598
DEF_60_CNT_SOCIAL_CIRCLE         0.517598
DAYS_LAST_PHONE_CHANGE           0.025880
FLAG_DOCUMENT_2                  0.025880
FLAG_DOCUMENT_3                  0.025880
FLAG_DOCUMENT_4                  0.025880
FLAG_DOCUMENT_5                  0.025880
FLAG_DOCUMENT_6                  0.025880
FLAG_DOCUMENT_7                  0.025880
FLAG_DOCUMENT_8                  0.025880
FLAG_DOCUMENT_9                  0.025880
FLAG_DOCUMENT_10                 0.025880
FLAG_DOCUMENT_11                 0.025880
FLAG_DOCUMENT_12                 0.025880
FLAG_DOCUMENT_13                 0.025880
FLAG_DOCUMENT_14                 0.025880
FLAG_DOCUMENT_15                 0.025880
FLAG_DOCUMENT_16                 0.025880
FLAG_DOCUMENT_17                 0.025880
FLAG_DOCUMENT_18                 0.025880
FLAG_DOCUMENT_19                 0.025880
FLAG_DOCUMENT_20                 0.025880
FLAG_DOCUMENT_21                 0.025880
AMT_REQ_CREDIT_BUREAU_HOUR      13.819876
AMT_REQ_CREDIT_BUREAU_DAY       13.819876
AMT_REQ_CREDIT_BUREAU_WEEK      13.819876
AMT_REQ_CREDIT_BUREAU_MON       13.819876
AMT_REQ_CREDIT_BUREAU_QRT       13.819876
AMT_REQ_CREDIT_BUREAU_YEAR      13.819876
dtype: float64
```

```
# CONCLUSION COLUMNS WITH NULL VALUES MORE THAN 47% MAY GIVE WRONG INSIGHTS.HENCE WILL DROP THEM
```

```
# DROPPING COLUMNS WITH MISSING VALUES GREATER THAN 47%

percentage = 47
threshold = int(((100-percentage)/100)*app_data.shape[0] + 1)
app_df=app_data.dropna(axis=1,thresh=threshold)
app_df.head()
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_A
0	100002	1	Cash loans	M	N	Y	0	202500.0	406597.5	2
1	100003	0	Cash loans	F	N	N	0	270000.0	1293502.5	3
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	135000.0	
3	100006	0	Cash loans	F	N	Y	0	135000.0	312682.5	2
4	100007	0	Cash loans	M	N	Y	0	121500.0	513000.0	2

```
app_df.shape
```

```
(3864, 74)
```

```
app_df.isnull().mean()*100
```

```
REG_REGION_NOT_LIVE_REGION      0.000000
REG_REGION_NOT_WORK_REGION      0.000000
LIVE_REGION_NOT_WORK_REGION     0.000000
REG_CITY_NOT_LIVE_CITY          0.000000
REG_CITY_NOT_WORK_CITY          0.000000
LIVE_CITY_NOT_WORK_CITY         0.000000
ORGANIZATION_TYPE               0.000000
EXT_SOURCE_2                    0.336439
EXT_SOURCE_3                   20.082816
EMERGENCYSTATE_MODE            46.972050
OBS_30_CNT_SOCIAL_CIRCLE        0.517598
DEF_30_CNT_SOCIAL_CIRCLE        0.517598
OBS_60_CNT_SOCIAL_CIRCLE        0.517598
DEF_60_CNT_SOCIAL_CIRCLE        0.517598
DAYS_LAST_PHONE_CHANGE          0.025880
FLAG_DOCUMENT_2                 0.025880
FLAG_DOCUMENT_3                 0.025880
FLAG_DOCUMENT_4                 0.025880
FLAG_DOCUMENT_5                 0.025880
FLAG_DOCUMENT_6                 0.025880
FLAG_DOCUMENT_7                 0.025880
FLAG_DOCUMENT_8                 0.025880
FLAG_DOCUMENT_9                 0.025880
FLAG_DOCUMENT_10                0.025880
FLAG_DOCUMENT_11                0.025880
FLAG_DOCUMENT_12                0.025880
FLAG_DOCUMENT_13                0.025880
FLAG_DOCUMENT_14                0.025880
FLAG_DOCUMENT_15                0.025880
FLAG_DOCUMENT_16                0.025880
FLAG_DOCUMENT_17                0.025880
FLAG_DOCUMENT_18                0.025880
FLAG_DOCUMENT_19                0.025880
FLAG_DOCUMENT_20                0.025880
FLAG_DOCUMENT_21                0.025880
AMT_REQ_CREDIT_BUREAU_HOUR     13.819876
AMT_REQ_CREDIT_BUREAU_DAY      13.819876
AMT_REQ_CREDIT_BUREAU_WEEK     13.819876
AMT_REQ_CREDIT_BUREAU_MON      13.819876
AMT_REQ_CREDIT_BUREAU_QRT      13.819876
AMT_REQ_CREDIT_BUREAU_YEAR     13.819876
dtype: float64
```

```
1 #IMPUTE MISSING VALUES
2 # check the missing values in application datset before imputing
3
4 app_df.info()
```

```
 18  DAYS_EMPLOYED               3864 non-null   int64
 19  DAYS_REGISTRATION           3864 non-null   float64
 20  DAYS_ID_PUBLISH             3864 non-null   int64
 21  FLAG_MOBIL                  3864 non-null   int64
 22  FLAG_EMP_PHONE              3864 non-null   int64
 23  FLAG_WORK_PHONE             3864 non-null   int64
 24  FLAG_CONT_MOBILE            3864 non-null   int64
 25  FLAG_PHONE                  3864 non-null   int64
 26  FLAG_EMAIL                  3864 non-null   int64
 27  OCCUPATION_TYPE             2665 non-null   object
 28  CNT_FAM_MEMBERS             3864 non-null   float64
 29  REGION_RATING_CLIENT        3864 non-null   int64
```

```
 53  FLAG_DOCUMENT_7              3863 non-null   float64
 54  FLAG_DOCUMENT_8              3863 non-null   float64
 55  FLAG_DOCUMENT_9              3863 non-null   float64
 56  FLAG_DOCUMENT_10             3863 non-null   float64
 57  FLAG_DOCUMENT_11             3863 non-null   float64
 58  FLAG_DOCUMENT_12             3863 non-null   float64
 59  FLAG_DOCUMENT_13             3863 non-null   float64
 60  FLAG_DOCUMENT_14             3863 non-null   float64
 61  FLAG_DOCUMENT_15             3863 non-null   float64
 62  FLAG_DOCUMENT_16             3863 non-null   float64
 63  FLAG_DOCUMENT_17             3863 non-null   float64
 64  FLAG_DOCUMENT_18             3863 non-null   float64
 65  FLAG_DOCUMENT_19             3863 non-null   float64
 66  FLAG_DOCUMENT_20             3863 non-null   float64
 67  FLAG_DOCUMENT_21             3863 non-null   float64
 68  AMT_REQ_CREDIT_BUREAU_HOUR   3330 non-null   float64
 69  AMT_REQ_CREDIT_BUREAU_DAY    3330 non-null   float64
 70  AMT_REQ_CREDIT_BUREAU_WEEK   3330 non-null   float64
 71  AMT_REQ_CREDIT_BUREAU_MON    3330 non-null   float64
 72  AMT_REQ_CREDIT_BUREAU_QRT    3330 non-null   float64
 73  AMT_REQ_CREDIT_BUREAU_YEAR   3330 non-null   float64
dtypes: float64(40), int64(21), object(13)
memory usage: 2.2+ MB
```

```
#OCCUPATION TYPE

app_df.OCCUPATION_TYPE.isnull().mean()*100
```

```
31.030020703933747
```

```
app_df.OCCUPATION_TYPE.value_counts(normalize=True)*100
```

```
OCCUPATION_TYPE
Laborers                 25.628518
Sales staff              16.210131
Core staff               12.908068
Drivers                   9.343340
Managers                  9.230769
High skill tech staff     4.990619
Accountants               4.840525
Medicine staff            4.240150
Security staff            3.227017
Cooking staff             2.889306
Cleaning staff            1.726079
Private service staff     1.350844
Low-skill Laborers        1.350844
Secretaries               0.712946
Waiters/barmen staff      0.412758
Realty agents             0.375235
HR staff                  0.337711
IT staff                  0.225141
Name: proportion, dtype: float64
```

```
app_df.OCCUPATION_TYPE.fillna('Other',inplace=True)
```

```
app_df.OCCUPATION_TYPE.isnull().mean()*100

```

```
0.0
```

```
app_df.OCCUPATION_TYPE.value_counts(normalize=True)*100
```

```
OCCUPATION_TYPE
Other                    31.030021
Laborers                 17.675983
Sales staff              11.180124
Core staff                8.902692
Drivers                   6.444099
Managers                  6.366460
High skill tech staff     3.442029
Accountants               3.338509
Medicine staff            2.924431
Security staff            2.225673
Cooking staff             1.992754
Cleaning staff            1.190476
Low-skill Laborers        0.931677
Private service staff     0.931677
Secretaries               0.491718
Waiters/barmen staff      0.284679
```

```
Realty agents             0.258799
HR staff                  0.232919
IT staff                  0.155280
Name: proportion, dtype: float64
```

```
#EXT_SOURCE_3 COLUMN MISSING VALUES 19%

app_df.EXT_SOURCE_3.isnull().mean()*100
```

```
20.082815734989648
```

```
app_df.EXT_SOURCE_3.value_counts(normalize=True)*100
```

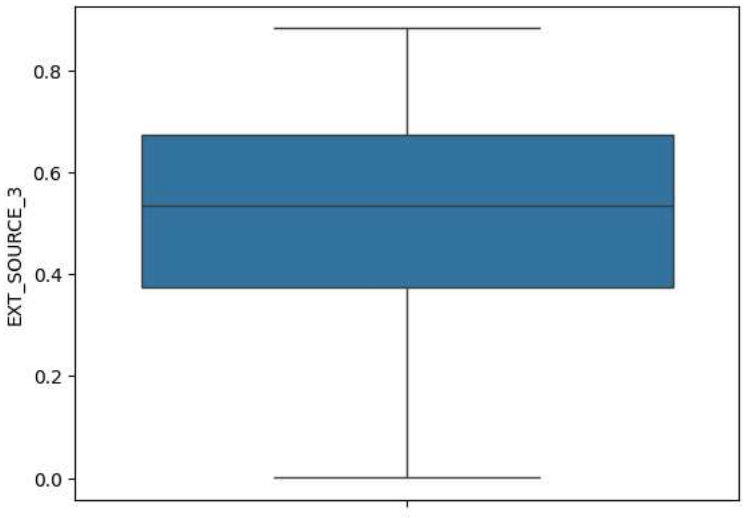
```
0.098859    0.032383
0.204423    0.032383
0.124519    0.032383
0.052036    0.032383
0.157595    0.032383
0.058826    0.032383
0.222581    0.032383
0.119878    0.032383
0.167408    0.032383
0.096948    0.032383
0.072497    0.032383
0.126898    0.032383
0.164414    0.032383
0.093225    0.032383
0.286652    0.032383
0.126101    0.032383
0.079060    0.032383
0.090221    0.032383
0.115387    0.032383
0.051682    0.032383
0.800451    0.032383
0.196334    0.032383
0.742182    0.032383
0.788681    0.032383
0.156640    0.032383
0.096319    0.032383
0.062103    0.032383
0.111756    0.032383
0.091412    0.032383
0.859924    0.032383
0.076474    0.032383
0.172495    0.032383
0.054957    0.032383
0.168416    0.032383
0.362277    0.032383
0.095693    0.032383
0.117614    0.032383
0.111042    0.032383
0.808394    0.032383
0.068251    0.032383
0.861653    0.032383
0.127699    0.032383
0.140243    0.032383
0.122955    0.032383
0.083364    0.032383
0.100152    0.032383
0.216403    0.032383
0.248536    0.032383
0.137654    0.032383
0.073965    0.032383
0.080650    0.032383
0.227613    0.032383
0.820383    0.032383
0.141115    0.032383
0.123735    0.032383
0.071533    0.032383
0.163426    0.032383
Name: proportion, dtype: float64
```

```
app_df.EXT_SOURCE_3.describe()
```

```
count    3088.000000
mean        0.512437
std         0.195588
min         0.000527
25%         0.374021
50%         0.535276
75%         0.672243
```

```
max         0.882530
Name: EXT_SOURCE_3, dtype: float64
```

```
sns.boxplot(app_df.EXT_SOURCE_3)
plt.show()
```




```
#conclusion  since its a numerical column with no outliers and there is not much difference between mean and median .hence we can impute
```

```
app_df.EXT_SOURCE_3.fillna(app_df.EXT_SOURCE_3.mean(),inplace=True)
```

```
app_df.EXT_SOURCE_3.isnull().mean()*100
```

```
0.0
```

```
app_df.EXT_SOURCE_3.value_counts(normalize=True)*100
```

```
0.068251     0.025880
0.861653     0.025880
0.127699     0.025880
0.140243     0.025880
0.122955     0.025880
0.083364     0.025880
0.100152     0.025880
0.216403     0.025880
0.248536     0.025880
0.137654     0.025880
0.073965     0.025880
0.080650     0.025880
0.227613     0.025880
0.820383     0.025880
0.141115     0.025880
0.123735     0.025880
0.071533     0.025880
0.163426     0.025880
Name: proportion, dtype: float64
```

```
null_cols = list(app_df.columns[app_df.isnull().any()])
len(null_cols)
```

```
35
```

```
app_df.isnull().mean()*100
```

```
DAYS_BIRTH                     0.000000
DAYS_EMPLOYED                  0.000000
DAYS_REGISTRATION              0.000000
DAYS_ID_PUBLISH                0.000000
FLAG_MOBIL                     0.000000
FLAG_EMP_PHONE                 0.000000
FLAG_WORK_PHONE                0.000000
FLAG_CONT_MOBILE               0.000000
FLAG_PHONE                     0.000000
FLAG_EMAIL                     0.000000
OCCUPATION_TYPE                0.000000
CNT_FAM_MEMBERS                0.000000
REGION_RATING_CLIENT           0.000000
REGION_RATING_CLIENT_W_CITY    0.000000
WEEKDAY_APPR_PROCESS_START     0.000000
HOUR_APPR_PROCESS_START        0.000000
REG_REGION_NOT_LIVE_REGION     0.000000
REG_REGION_NOT_WORK_REGION     0.000000
LIVE_REGION_NOT_WORK_REGION    0.000000
REG_CITY_NOT_LIVE_CITY         0.000000
REG_CITY_NOT_WORK_CITY         0.000000
LIVE_CITY_NOT_WORK_CITY        0.000000
ORGANIZATION_TYPE              0.000000
EXT_SOURCE_2                   0.336439
EXT_SOURCE_3                   0.000000
EMERGENCYSTATE_MODE           46.972050
OBS_30_CNT_SOCIAL_CIRCLE       0.517598
DEF_30_CNT_SOCIAL_CIRCLE       0.517598
OBS_60_CNT_SOCIAL_CIRCLE       0.517598
DEF_60_CNT_SOCIAL_CIRCLE       0.517598
DAYS_LAST_PHONE_CHANGE         0.025880
FLAG_DOCUMENT_2                0.025880
FLAG_DOCUMENT_3                0.025880
FLAG_DOCUMENT_4                0.025880
FLAG_DOCUMENT_5                0.025880
FLAG_DOCUMENT_6                0.025880
FLAG_DOCUMENT_7                0.025880
FLAG_DOCUMENT_8                0.025880
```

```
AMT_REQ_CREDIT_BUREAU_QRT      13.819876
AMT_REQ_CREDIT_BUREAU_YEAR     13.819876
dtype: float64
```

```
#HANDLING MISSING VALUES IN COLUMNS WITH 13.5%NULL VALUES

app_df.AMT_REQ_CREDIT_BUREAU_HOUR.value_counts(normalize=True)*100
```

```
AMT_REQ_CREDIT_BUREAU_HOUR
0.0    99.069069
1.0     0.900901
2.0     0.030030
Name: proportion, dtype: float64
```

```
app_df.AMT_REQ_CREDIT_BUREAU_DAY.value_counts(normalize=True)*100
```

```
AMT_REQ_CREDIT_BUREAU_DAY
0.0    99.339339
1.0     0.570571
3.0     0.030030
2.0     0.030030
4.0     0.030030
Name: proportion, dtype: float64
```

```
#CONCLUSION  we could see that 99% of values in the columns AMT_REQ_BUREAU_HOUR,AMT_REQ_CREDIT_BUREAU_DAY,AMT_REQ_CREDIT_BUREAU_WEEK,AMT_
```

```
Cols = ['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']  # Create a list of column names

for col in Cols:
    app_df[col].fillna(app_df[col].mode()[0], inplace=True)  # Fill missing values with mode for each column
```

```
app_df.isnull().mean()*100
```

```
DAYS_BIRTH                     0.000000
DAYS_EMPLOYED                  0.000000
DAYS_REGISTRATION              0.000000
DAYS_ID_PUBLISH                0.000000
FLAG_MOBIL                     0.000000
FLAG_EMP_PHONE                 0.000000
FLAG_WORK_PHONE                0.000000
FLAG_CONT_MOBILE               0.000000
FLAG_PHONE                     0.000000
FLAG_EMAIL                     0.000000
OCCUPATION_TYPE                0.000000
CNT_FAM_MEMBERS                0.000000
REGION_RATING_CLIENT           0.000000
REGION_RATING_CLIENT_W_CITY    0.000000
WEEKDAY_APPR_PROCESS_START     0.000000
HOUR_APPR_PROCESS_START        0.000000
REG_REGION_NOT_LIVE_REGION     0.000000
REG_REGION_NOT_WORK_REGION     0.000000
LIVE_REGION_NOT_WORK_REGION    0.000000
REG_CITY_NOT_LIVE_CITY         0.000000
REG_CITY_NOT_WORK_CITY         0.000000
LIVE_CITY_NOT_WORK_CITY        0.000000
ORGANIZATION_TYPE              0.000000
EXT_SOURCE_2                   0.336439
EXT_SOURCE_3                   0.000000
EMERGENCYSTATE_MODE           46.972050
OBS_30_CNT_SOCIAL_CIRCLE       0.517598
DEF_30_CNT_SOCIAL_CIRCLE       0.517598
```

```
FLAG_DOCUMENT_16                0.025880
FLAG_DOCUMENT_17                0.025880
FLAG_DOCUMENT_18                0.025880
FLAG_DOCUMENT_19                0.025880
FLAG_DOCUMENT_20                0.025880
FLAG_DOCUMENT_21                0.025880
AMT_REQ_CREDIT_BUREAU_HOUR      0.000000
AMT_REQ_CREDIT_BUREAU_DAY       0.000000
AMT_REQ_CREDIT_BUREAU_WEEK      0.000000
AMT_REQ_CREDIT_BUREAU_MON       0.000000
AMT_REQ_CREDIT_BUREAU_QRT       0.000000
AMT_REQ_CREDIT_BUREAU_YEAR      0.000000
dtype: float64
```

```
#HANDLING MISSING VALUES LESS THAN 1%

null_cols=list(app_df.columns[app_df.isnull().any()])
len(null_cols)
```

```
29
```

```
app_df.NAME_TYPE_SUITE.value_counts(normalize=True)*100
```

```
NAME_TYPE_SUITE
Unaccompanied      81.282451
Family             13.395639
Spouse, partner     3.530633
Children            0.830737
Other_B             0.700935
Other_A             0.207684
Group of people     0.051921
Name: proportion, dtype: float64
```

```
app_df.EXT_SOURCE_2.value_counts(normalize=True)*100
```

```
0.497561    0.025967
0.201584    0.025967
0.583926    0.025967
0.505288    0.025967
0.430618    0.025967
0.547036    0.025967
0.757328    0.025967
0.628532    0.025967
0.064858    0.025967
0.601408    0.025967
0.515845    0.025967
0.256737    0.025967
0.627200    0.025967
```

```
app_df.NAME_TYPE_SUITE.value_counts(normalize=True)*100
```

```
NAME_TYPE_SUITE
Unaccompanied      81.282451
Family             13.395639
Spouse, partner     3.530633
Children            0.830737
Other_B             0.700935
Other_A             0.207684
Group of people     0.051921
Name: proportion, dtype: float64
```

```
app_df.OBS_30_CNT_SOCIAL_CIRCLE.value_counts(normalize=True)*100
```

```
OBS_30_CNT_SOCIAL_CIRCLE
0.0     53.954214
1.0     15.322581
2.0      9.703434
3.0      6.659729
4.0      4.968783
5.0      3.121748
6.0      2.289282
7.0      1.118626
9.0      0.832466
8.0      0.780437
10.0     0.494277
11.0     0.338189
13.0     0.130073
12.0     0.104058
14.0     0.104058
22.0     0.052029
16.0     0.026015
Name: proportion, dtype: float64
```

```
#CONCLUSION  FOR CATEGORICAL COLUMNS IMPUTE THE MISSING VALUES WITH MODE
#            FOR NUMERICAL COLUMNS IMPUTING THE MISSING VALUES WITH MEDIAN
```

```
app_df.NAME_TYPE_SUITE.fillna(app_df.NAME_TYPE_SUITE.mode()[0],inplace=True)
```

```
app_df.CNT_FAM_MEMBERS.fillna(app_df.CNT_FAM_MEMBERS.mode(),inplace=True)
```

```
#IMPUTING NUMERICAL COLUMNS

app_df.EXT_SOURCE_2.fillna(app_df.EXT_SOURCE_2.median(),inplace=True)
app_df.AMT_GOODS_PRICE.fillna(app_df.AMT_GOODS_PRICE.median(),inplace=True)
app_df.AMT_ANNUITY.fillna(app_df.AMT_ANNUITY.median(),inplace=True)
app_df.DEF_60_CNT_SOCIAL_CIRCLE.fillna(app_df.DEF_60_CNT_SOCIAL_CIRCLE.median(),inplace=True)
app_df.DEF_30_CNT_SOCIAL_CIRCLE.fillna(app_df.DEF_30_CNT_SOCIAL_CIRCLE.median(),inplace=True)
app_df.OBS_30_CNT_SOCIAL_CIRCLE.fillna(app_df.OBS_30_CNT_SOCIAL_CIRCLE.median(),inplace=True)
app_df.OBS_60_CNT_SOCIAL_CIRCLE.fillna(app_df.OBS_60_CNT_SOCIAL_CIRCLE.median(),inplace=True)
app_df.DAYS_LAST_PHONE_CHANGE.fillna(app_df.DAYS_LAST_PHONE_CHANGE.median(),inplace=True)

```

```
null_cols=list(app_df.columns[app_df.isnull().any()])
len(null_cols)
```

```
21
```

```
app_df.isnull().mean()*100
```

```
DAYS_REGISTRATION              0.00000
DAYS_ID_PUBLISH                0.00000
FLAG_MOBIL                     0.00000
FLAG_EMP_PHONE                 0.00000
FLAG_WORK_PHONE                0.00000
FLAG_CONT_MOBILE               0.00000
FLAG_PHONE                     0.00000
FLAG_EMAIL                     0.00000
OCCUPATION_TYPE                0.00000
CNT_FAM_MEMBERS                0.00000
REGION_RATING_CLIENT           0.00000
REGION_RATING_CLIENT_W_CITY    0.00000
WEEKDAY_APPR_PROCESS_START     0.00000
HOUR_APPR_PROCESS_START        0.00000
REG_REGION_NOT_LIVE_REGION     0.00000
REG_REGION_NOT_WORK_REGION     0.00000
LIVE_REGION_NOT_WORK_REGION    0.00000
REG_CITY_NOT_LIVE_CITY         0.00000
REG_CITY_NOT_WORK_CITY         0.00000
LIVE_CITY_NOT_WORK_CITY        0.00000
ORGANIZATION_TYPE              0.00000
EXT_SOURCE_2                   0.00000
EXT_SOURCE_3                   0.00000
EMERGENCYSTATE_MODE           46.97205
OBS_30_CNT_SOCIAL_CIRCLE       0.00000
DEF_30_CNT_SOCIAL_CIRCLE       0.00000
OBS_60_CNT_SOCIAL_CIRCLE       0.00000
DEF_60_CNT_SOCIAL_CIRCLE       0.00000
DAYS_LAST_PHONE_CHANGE         0.00000
FLAG_DOCUMENT_2                0.02588
FLAG_DOCUMENT_3                0.02588
FLAG_DOCUMENT_4                0.02588
FLAG_DOCUMENT_5                0.02588
FLAG_DOCUMENT_6                0.02588
FLAG_DOCUMENT_7                0.02588
FLAG_DOCUMENT_8                0.02588
FLAG_DOCUMENT_9                0.02588
FLAG_DOCUMENT_10               0.02588
FLAG_DOCUMENT_11               0.02588
FLAG_DOCUMENT_12               0.02588
FLAG_DOCUMENT_13               0.02588
FLAG_DOCUMENT_14               0.02588
FLAG_DOCUMENT_15               0.02588
FLAG_DOCUMENT_16               0.02588
FLAG_DOCUMENT_17               0.02588
FLAG_DOCUMENT_18               0.02588
FLAG_DOCUMENT_19               0.02588
FLAG_DOCUMENT_20               0.02588
FLAG_DOCUMENT_21               0.02588
AMT_REQ_CREDIT_BUREAU_HOUR     0.00000
AMT_REQ_CREDIT_BUREAU_DAY      0.00000
AMT_REQ_CREDIT_BUREAU_WEEK     0.00000
AMT_REQ_CREDIT_BUREAU_MON      0.00000
AMT_REQ_CREDIT_BUREAU_QRT      0.00000
AMT_REQ_CREDIT_BUREAU_YEAR     0.00000
dtype: float64
```

```
# CONVERT NEGATIVE TO POSITIVE IN DAYS VARIABLES SO THAT MEDIAN IS NOT DEFINED

app_df.DAYS_BIRTH=app_df.DAYS_BIRTH.apply(lambda x: abs(x))
app_df.DAYS_EMPLOYED=app_df.DAYS_EMPLOYED.apply(lambda x: abs(x))
app_df.DAYS_REGISTRATION=app_df.DAYS_REGISTRATION.apply(lambda x: abs(x))
app_df.DAYS_ID_PUBLISH=app_df.DAYS_ID_PUBLISH.apply(lambda x: abs(x))
app_df.DAYS_LAST_PHONE_CHANGE=app_df.DAYS_LAST_PHONE_CHANGE.apply(lambda x: abs(x))
```

```
# BINNING OF CONTINUOUS VARIABLES

### STANDARDIZING DAYS COLUMNS IN YEARS FOR EASY BINNING

app_df['YEARS_BIRTH'] = app_df.DAYS_BIRTH.apply(lambda x: int(x//365))
app_df['YEARS_EMPLOYED'] = app_df.DAYS_EMPLOYED.apply(lambda x: int(x//365))
app_df['YEARS_REGISTRATION'] = app_df.DAYS_REGISTRATION.apply(lambda x: int(x//365))
app_df['YEARS_ID_PUBLISH'] = app_df.DAYS_ID_PUBLISH.apply(lambda x: int(x//365))
app_df['YEARS_LAST_PHONE_CHANGE'] = app_df.DAYS_LAST_PHONE_CHANGE.apply(lambda x: int(x//365))
```

```
# BINNING AMT_CREDIT COLUMN

app_df.AMT_CREDIT.value_counts(normalize=True)*100
```

```
701721.0     0.025880
553581.0     0.025880
1084383.0    0.025880
579942.0     0.025880
305955.0     0.025880
81504.0      0.025880
1024636.5    0.025880
854901.0     0.025880
525735.0     0.025880
1479541.5    0.025880
505665.0     0.025880
1170000.0    0.025880
760131.0     0.025880
1774039.5    0.025880
102384.0     0.025880
621000.0     0.025880
389484.0     0.025880
913180.5     0.025880
703728.0     0.025880
416052.0     0.025880
420718.5     0.025880
479974.5     0.025880
940500.0     0.025880
1635795.0    0.025880
1213227.0    0.025880
1057266.0    0.025880
358213.5     0.025880
859500.0     0.025880
1540588.5    0.025880
2159842.5    0.025880
1124478.0    0.025880
805072.5     0.025880
297040.5     0.025880
1965226.5    0.025880
344043.0     0.025880
943425.0     0.025880
1100709.0    0.025880
657702.0     0.025880
1963494.0    0.025880
142200.0     0.025880
381096.0     0.025880
776583.0     0.025880
1093500.0    0.025880
657000.0     0.025880
445086.0     0.025880
604413.0     0.025880
543037.5     0.025880
730017.0     0.025880
779688.0     0.025880
745119.0     0.025880
130320.0     0.025880
477175.5     0.025880
814500.0     0.025880
1260702.0    0.025880
237204.0     0.025880
133528.5     0.025880
614574.0     0.025880
Name: proportion, dtype: float64
```

```
app_df.AMT_CREDIT.describe()
```

```
count    3.864000e+03
mean     6.017847e+05
std      4.033991e+05
min      4.500000e+04
25%      2.700000e+05
50%      5.095012e+05
75%      8.086500e+05
max      2.517300e+06
Name: AMT_CREDIT, dtype: float64
```

```
app_df['AMT_CREDIT_CATEGORY'] = pd.cut(app_df.AMT_CREDIT, [0,200000,400000,600000,800000,1000000], labels=['Very Low Credit', 'Low Credi
```

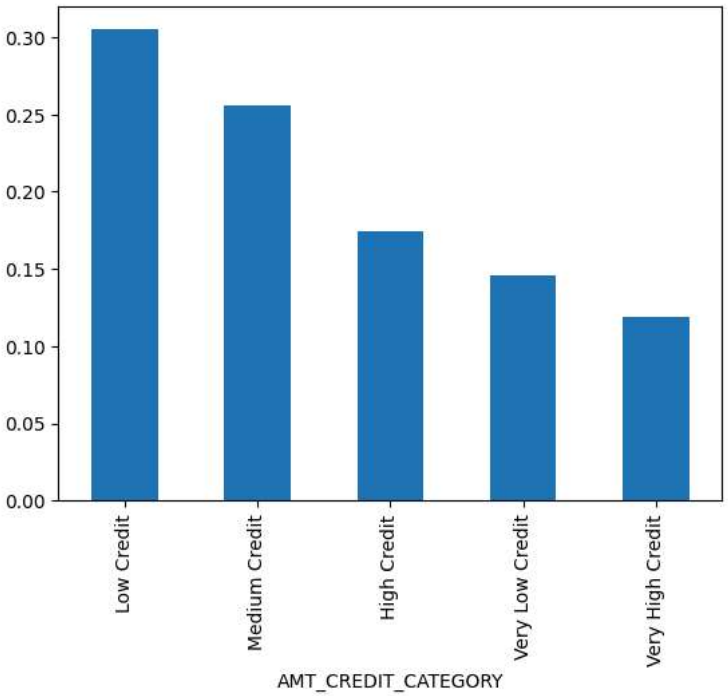
```
app_df.AMT_CREDIT_CATEGORY.value_counts(normalize=True)*100
```

```
AMT_CREDIT_CATEGORY
Low Credit          30.495356
Medium Credit       25.572755
High Credit         17.430341
Very Low Credit     14.613003
```

```
Very High Credit    11.888545
Name: proportion, dtype: float64
```

```
app_df.AMT_CREDIT_CATEGORY.value_counts(normalize=True).plot.bar()
```

```
<Axes: xlabel='AMT_CREDIT_CATEGORY'>
```




```
# conclusion  -the credit amount of the loan for amount low (2L to 4L) or very high (above 8L)
```

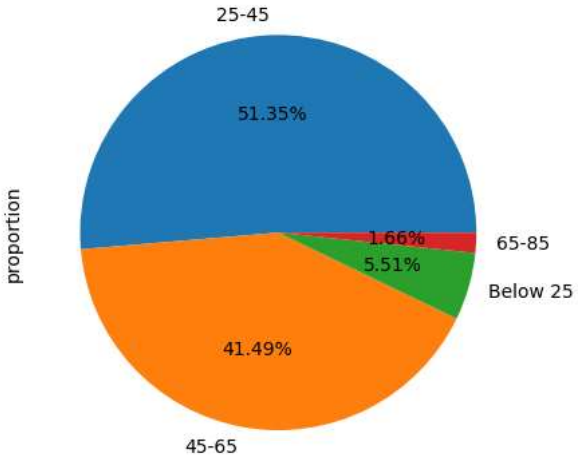
```
#binning years_birth column

app_df['AGE_CATEGORY'] = pd.cut(app_df.YEARS_BIRTH, [0,25,45,65,85], labels=['Below 25', '25-45', '45-65', '65-85'])
```

```
app_df.AGE_CATEGORY.value_counts(normalize=True)*100
```

```
AGE_CATEGORY
25-45       51.345756
45-65       41.485507
Below 25     5.512422
65-85        1.656315
Name: proportion, dtype: float64
```

```
app_df['AGE_CATEGORY'].value_counts(normalize=True).plot.pie(autopct='%1.2f%%')
plt.show()
```



```
#conclusion - most of the applicants are between 25-45 age grp

```

```
# DATA IMBALANCE CHECK

app_df.head()
```

AMT_REQ_CREDIT_BUREAU_QRT	AMT_REQ_CREDIT_BUREAU_YEAR	YEARS_BIRTH	YEARS_EMPLOYED	YEARS_REGISTRATION	YEARS_ID_PUBLISH	YEARS_LAST_PHO
0.0	1.0	25	1	9	5	
0.0	0.0	45	3	3	0	
0.0	0.0	52	0	11	6	
0.0	0.0	52	8	26	6	
0.0	0.0	54	8	11	9	

```
# Diving application dataset with target variables as 0 and 1

tar_0 = app_df[app_df.TARGET==0]
tar_1 = app_df[app_df.TARGET==1]
```

```
app_df.TARGET.value_counts(normalize=True)*100
```

```
TARGET
0    92.158385
1     7.841615
Name: proportion, dtype: float64
```

```
# conclusion - 1 out of 9/10 applicants are defaults
```

```
#UNIVARIATE ANALYSIS

cat_cols = list(app_df.columns[app_df.dtypes==object])
num_cols = list(app_df.columns[app_df.dtypes==np.int64]) + list(app_df.columns[app_df.dtypes==np.float64])
```

```
cat_cols
```

```
['NAME_CONTRACT_TYPE',
 'CODE_GENDER',
 'FLAG_OWN_CAR',
 'FLAG_OWN_REALTY',
 'NAME_TYPE_SUITE',
 'NAME_INCOME_TYPE',
 'NAME_EDUCATION_TYPE',
 'NAME_FAMILY_STATUS',
 'NAME_HOUSING_TYPE',
 'OCCUPATION_TYPE',
 'WEEKDAY_APPR_PROCESS_START',
 'ORGANIZATION_TYPE',
 'EMERGENCYSTATE_MODE']
```

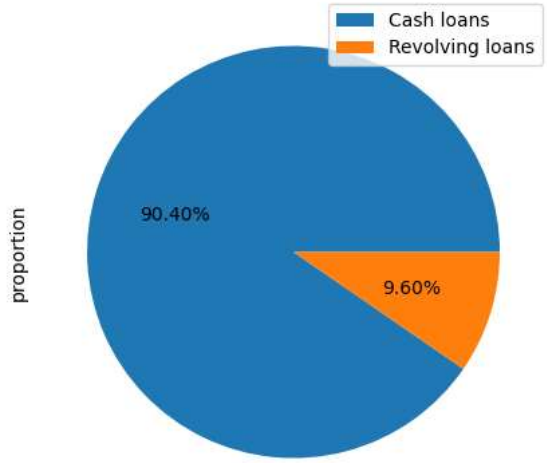
```
num_cols
```

```
 'FLAG_WORK_PHONE',
 'FLAG_CONT_MOBILE',
 'FLAG_PHONE',
 'FLAG_EMAIL',
 'REGION_RATING_CLIENT',
```

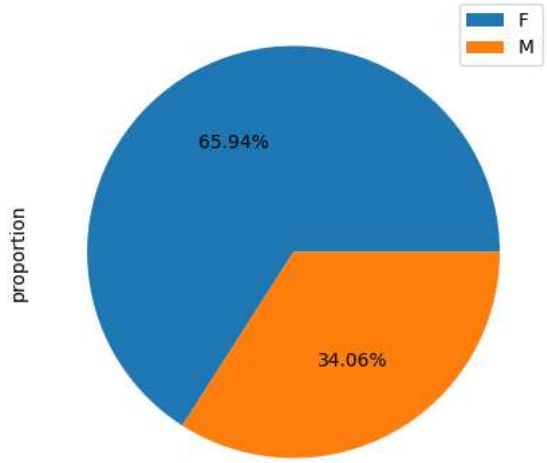
```
 'YEARS_BIRTH',
'YEARS_EMPLOYED',
'YEARS_REGISTRATION',
'YEARS_ID_PUBLISH',
'YEARS_LAST_PHONE_CHANGE',
'AMT_INCOME_TOTAL',
'AMT_CREDIT',
'AMT_ANNUITY',
'AMT_GOODS_PRICE',
'REGION_POPULATION_RELATIVE',
'DAYS_REGISTRATION',
'CNT_FAM_MEMBERS',
'EXT_SOURCE_2',
'EXT_SOURCE_3',
'OBS_30_CNT_SOCIAL_CIRCLE',
'DEF_30_CNT_SOCIAL_CIRCLE',
'OBS_60_CNT_SOCIAL_CIRCLE',
'DEF_60_CNT_SOCIAL_CIRCLE',
'DAYS_LAST_PHONE_CHANGE',
'FLAG_DOCUMENT_2',
'FLAG_DOCUMENT_3',
'FLAG_DOCUMENT_4',
'FLAG_DOCUMENT_5',
'FLAG_DOCUMENT_6',
'FLAG_DOCUMENT_7',
'FLAG_DOCUMENT_8',
'FLAG_DOCUMENT_9',
'FLAG_DOCUMENT_10',
'FLAG_DOCUMENT_11',
'FLAG_DOCUMENT_12',
'FLAG_DOCUMENT_13',
'FLAG_DOCUMENT_14',
'FLAG_DOCUMENT_15',
'FLAG_DOCUMENT_16',
'FLAG_DOCUMENT_17',
'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']
```

```
for col in cat_cols:
    print(app_df[col].value_counts(normalize=True))
    plt.figure(figsize=(5,5))
    app_df[col].value_counts(normalize=True).plot.pie(labeldistance=None,autopct='%1.2f%%')
    plt.legend()
    plt.show()
```

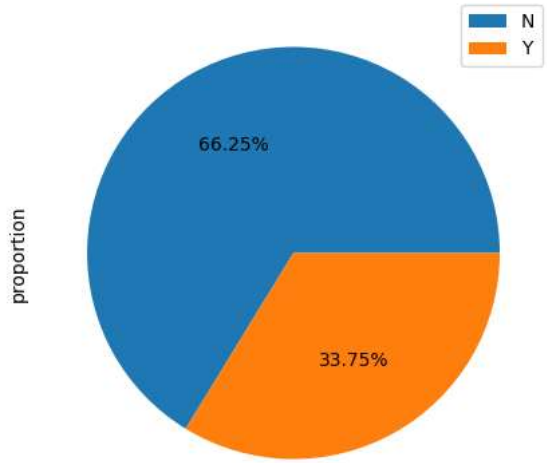
```
NAME_CONTRACT_TYPE
Cash loans         0.903986
Revolving loans    0.096014
Name: proportion, dtype: float64
```




```
CODE_GENDER
F    0.65942
M    0.34058
Name: proportion, dtype: float64
```

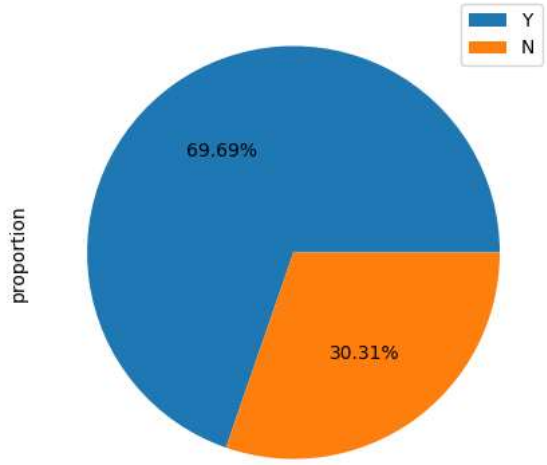



```
FLAG_OWN_CAR
N    0.662526
Y    0.337474
Name: proportion, dtype: float64
```

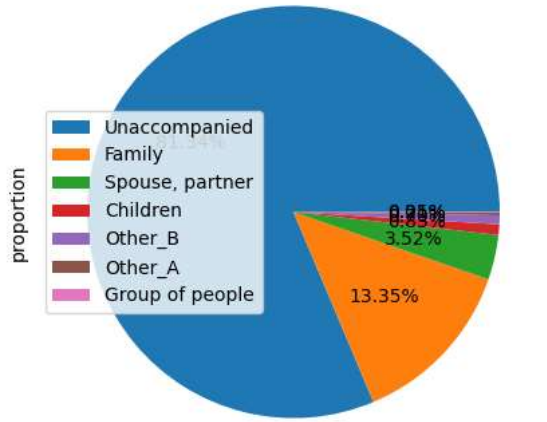



```
FLAG_OWN_REALTY
Y    0.696946
```

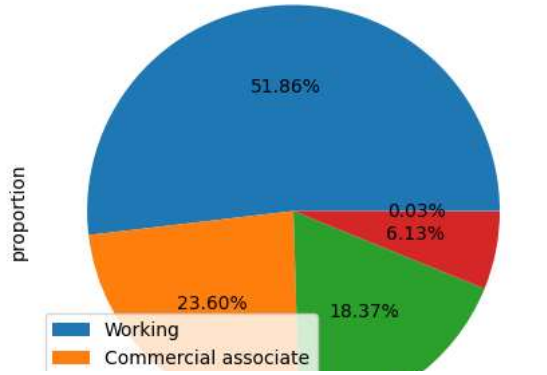
```
Y    0.696946
N    0.303054
Name: proportion, dtype: float64
```



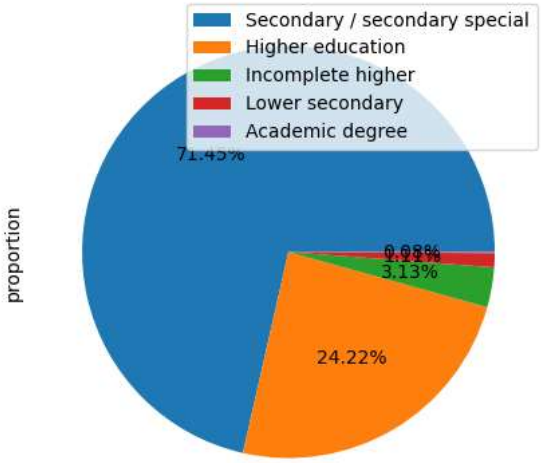

```
NAME_TYPE_SUITE
Unaccompanied      0.813406
Family             0.133540
Spouse, partner    0.035197
Children           0.008282
Other_B            0.006988
Other_A            0.002070
Group of people    0.000518
Name: proportion, dtype: float64
```



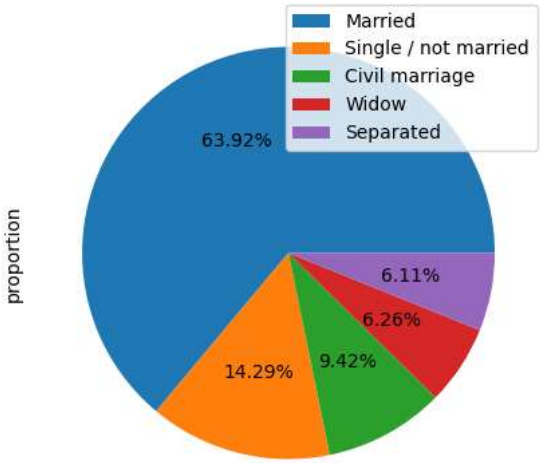

```
NAME_INCOME_TYPE
Working                 0.518634
Commercial associate    0.236025
Pensioner               0.183747
State servant           0.061335
Unemployed              0.000259
Name: proportion, dtype: float64
```



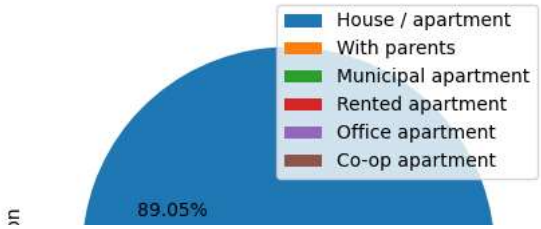

```
NAME_EDUCATION_TYPE
Secondary / secondary special    0.714545
Higher education                 0.242236
Incomplete higher                0.031315
Lower secondary                  0.011128
Academic degree                  0.000776
Name: proportion, dtype: float64
```

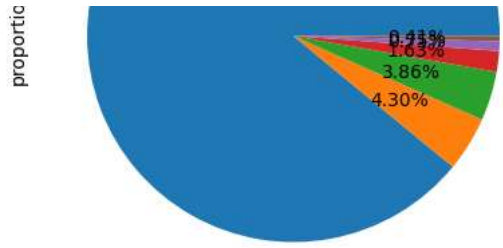



```
NAME_FAMILY_STATUS
Married                 0.639234
Single / not married    0.142857
Civil marriage          0.094203
Widow                   0.062629
Separated               0.061077
Name: proportion, dtype: float64
```

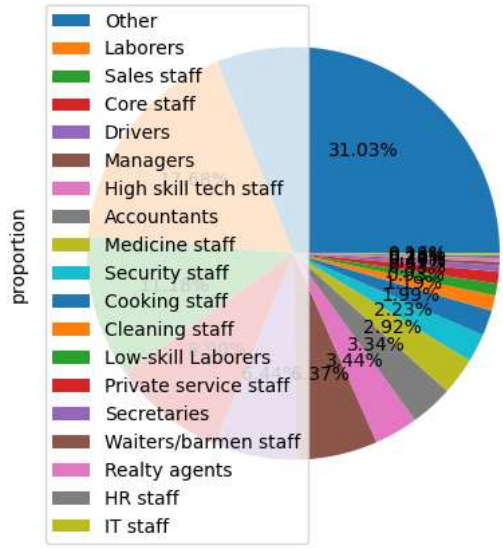



```
NAME_HOUSING_TYPE
House / apartment      0.890528
With parents           0.042961
Municipal apartment    0.038561
Rented apartment       0.016304
Office apartment       0.007505
Co-op apartment        0.004141
Name: proportion, dtype: float64
```

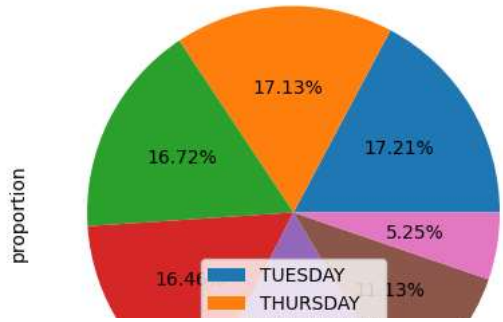




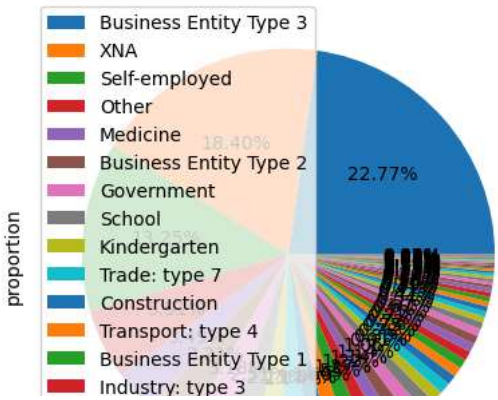

```
OCCUPATION_TYPE
Other                    0.310300
Laborers                 0.176760
Sales staff              0.111801
Core staff               0.089027
Drivers                  0.064441
Managers                 0.063665
High skill tech staff    0.034420
Accountants              0.033385
Medicine staff           0.029244
Security staff           0.022257
Cooking staff            0.019928
Cleaning staff           0.011905
Low-skill Laborers       0.009317
Private service staff    0.009317
Secretaries              0.004917
Waiters/barmen staff     0.002847
Realty agents            0.002588
HR staff                 0.002329
IT staff                 0.001553
Name: proportion, dtype: float64
```



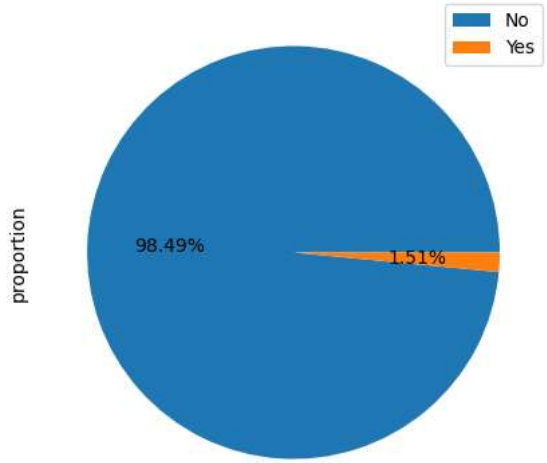

```
WEEKDAY_APPR_PROCESS_START
TUESDAY      0.172101
THURSDAY     0.171325
WEDNESDAY    0.167184
MONDAY       0.164596
FRIDAY       0.160973
SATURDAY     0.111284
SUNDAY       0.052536
Name: proportion, dtype: float64
```




```
ORGANIZATION_TYPE
Business Entity Type 3    0.227743
XNA                       0.184006
Self-employed             0.132505
Other                     0.055124
Medicine                  0.034938
Business Entity Type 2    0.032091
Government                0.031832
School                    0.025104
Kindergarten              0.021480
Trade: type 7             0.020704
Construction              0.018375
Transport: type 4         0.016046
Business Entity Type 1    0.015528
Industry: type 3          0.012681
Trade: type 3             0.012164
Housing                   0.011387
Industry: type 9          0.011387
Industry: type 11         0.010611
Services                  0.009058
Transport: type 2         0.008799
Postal                    0.007764
Military                  0.007505
Security                  0.007246
Bank                      0.007246
Police                    0.006988
Agriculture               0.006729
Security Ministries       0.005694
Trade: type 2             0.005435
Transport: type 3         0.004658
Restaurant                0.004141
Industry: type 7          0.004141
Hotel                     0.003882
University                0.003623
Industry: type 1          0.003623
Electricity               0.003364
Industry: type 12         0.002588
Emergency                 0.002329
Industry: type 4          0.002329
Trade: type 6             0.002329
Industry: type 5          0.002070
Industry: type 2          0.001812
Insurance                 0.001553
Mobile                    0.001294
Telecom                   0.001294
Legal Services            0.001035
Industry: type 13         0.001035
Transport: type 1         0.000776
Trade: type 1             0.000776
Culture                   0.000776
Industry: type 10         0.000776
Advertising               0.000776
Cleaning                  0.000776
Religion                  0.000518
Trade: type 5             0.000518
Realtor                   0.000518
Trade: type 4             0.000259
Industry: type 8          0.000259
Name: proportion, dtype: float64
```




```
EMERGENCYSTATE_MODE
No     0.984871
Yes    0.015129
Name: proportion, dtype: float64
```



```
# conclusion >> insights on below columns
#1.NAME_CONTRACT_TYPE -more application have cash loans than receiving loans
#2.CODE_GENDER - number of female applicants are twice the male applicants
#3.FLAG_OWN_CAR-most (70%)of the applicants do not own car
#4.FLAG_OWN_REALTY-most (70%)of the applicants do not own a house
#5.NAME_TYPE_SUITE-most(81%) of the applicants are unaccomplished
#6.NAME_INCOME_TYPE-most (51%) of the applicants are working
#7.NAME_EDUCATION_TYPE-most (71%) of the applicants has completed secondary education
#8.NAME_FAMILY_STATUS-most (63%) of the applicants are married
#9.NAME_HOUSING_TYPE-most (88%) of the applicants own a house
#10.OCCUPATION_TYPE-most (31%) of the applicants have other occupation type
#11.WEEKDAY_APPR_PROCESS_START- MOST of the applicant have applied the loan on tuesday
#12.ORGANIZATION_TYPE-most of the oreganization type of employees are business entity type 3
```

```
# PLOT ON NUMERICAL COLUMNS
# CATEGORIZING COLUMNS WITH AND WITHOUT FLAGS

num_cols_withoutflag=[]
num_cols_withflag=[]

for col in num_cols:
    if col.startswith('FLAG'):
        num_cols_withflag.append(col)

    else:
        num_cols_withoutflag.append(col)
```

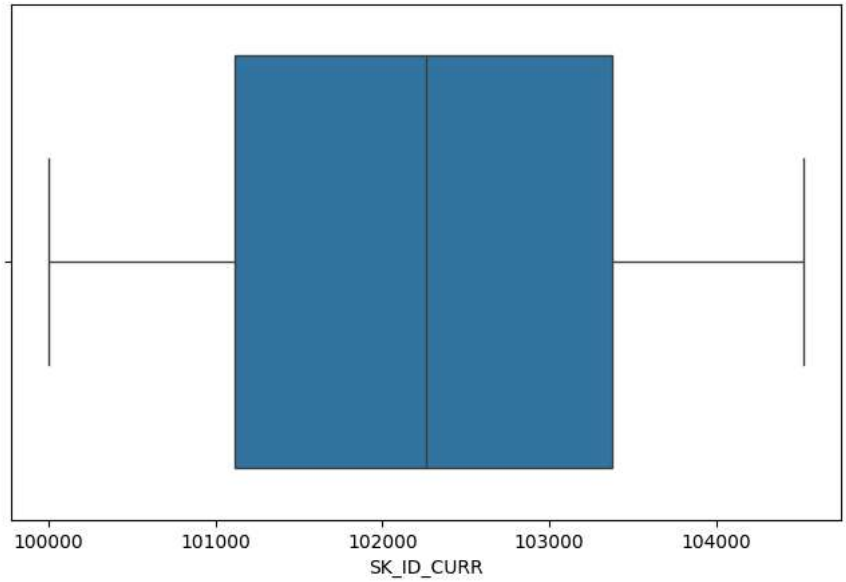
Double-click (or enter) to edit

```
num_cols_withoutflag
```

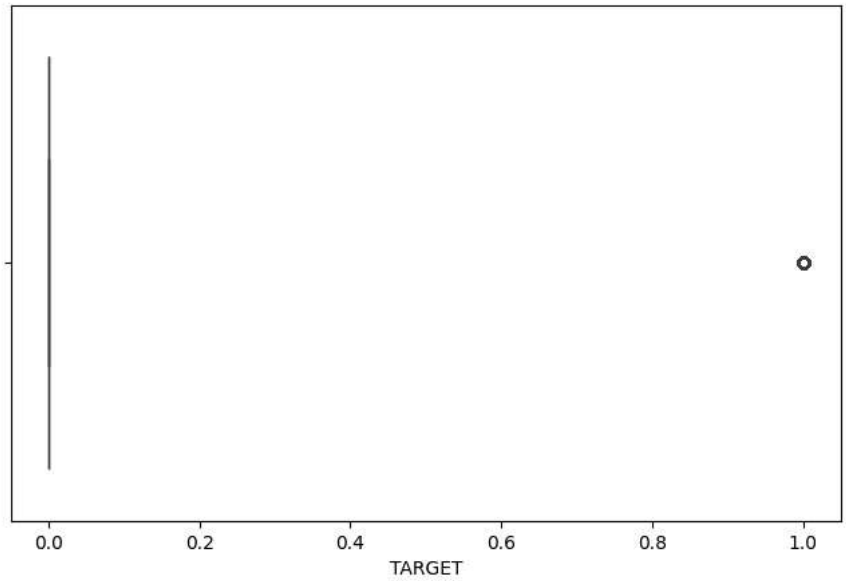
```
['SK_ID_CURR',
 'TARGET',
 'CNT_CHILDREN',
 'DAYS_BIRTH',
 'DAYS_EMPLOYED',
 'DAYS_ID_PUBLISH',
 'REGION_RATING_CLIENT',
 'REGION_RATING_CLIENT_W_CITY',
 'HOUR_APPR_PROCESS_START',
 'REG_REGION_NOT_LIVE_REGION',
 'REG_REGION_NOT_WORK_REGION',
 'LIVE_REGION_NOT_WORK_REGION',
 'REG_CITY_NOT_LIVE_CITY',
 'REG_CITY_NOT_WORK_CITY',
 'LIVE_CITY_NOT_WORK_CITY',
 'YEARS_BIRTH',
 'YEARS_EMPLOYED',
 'YEARS_REGISTRATION',
 'YEARS_ID_PUBLISH',
 'YEARS_LAST_PHONE_CHANGE',
 'AMT_INCOME_TOTAL',
 'AMT_CREDIT',
 'AMT_ANNUITY',
 'AMT_GOODS_PRICE',
 'REGION_POPULATION_RELATIVE',
 'DAYS_REGISTRATION',
 'CNT_FAM_MEMBERS',
 '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']
```

```
for col in num_cols_withoutflag:
    print(app_df[col].describe())
    plt.figure(figsize=(8,5))
    sns.boxplot(data=app_df,x=col)
    plt.show()
    print('----------------------------')
```

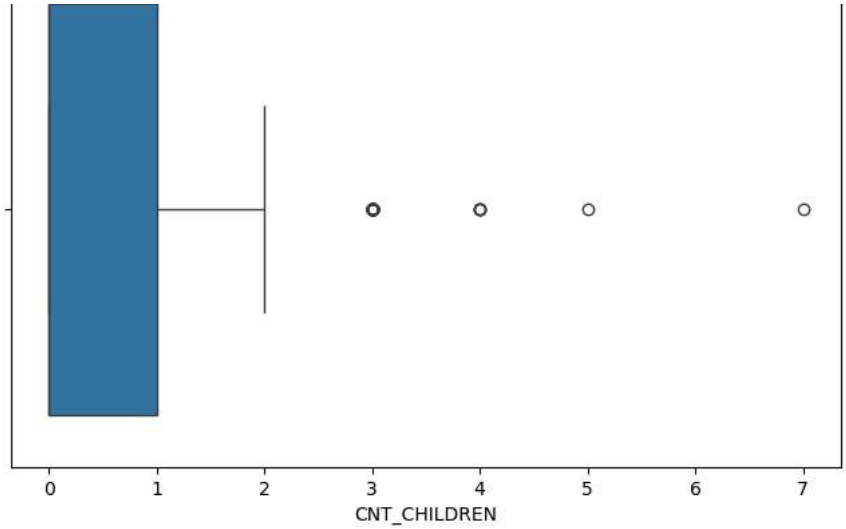
```
count      3864.000000
mean     102256.062888
std        1304.554619
min      100002.000000
25%      101115.750000
50%      102264.500000
75%      103379.250000
max      104519.000000
Name: SK_ID_CURR, dtype: float64
```



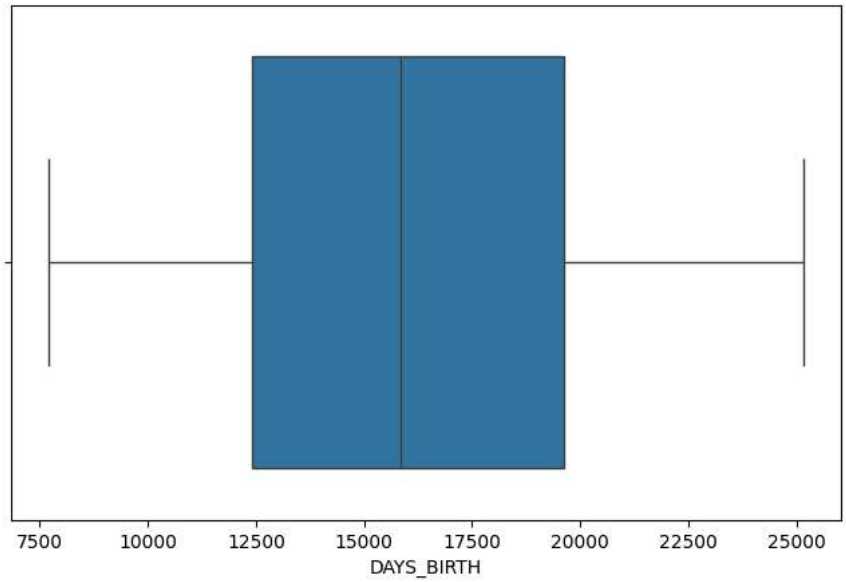

```
----------------------------
count    3864.000000
mean        0.078416
std         0.268860
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: TARGET, dtype: float64
```



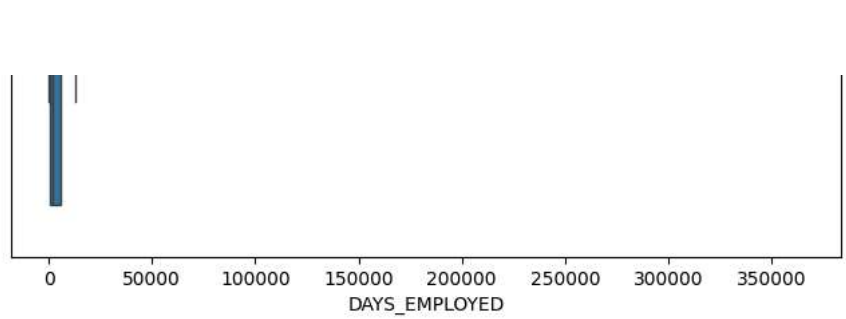

```
----------------------------
count    3864.000000
mean        0.402950
std         0.713685
min         0.000000
25%         0.000000
50%         0.000000
75%         1.000000
max         7.000000
Name: CNT_CHILDREN, dtype: float64
```



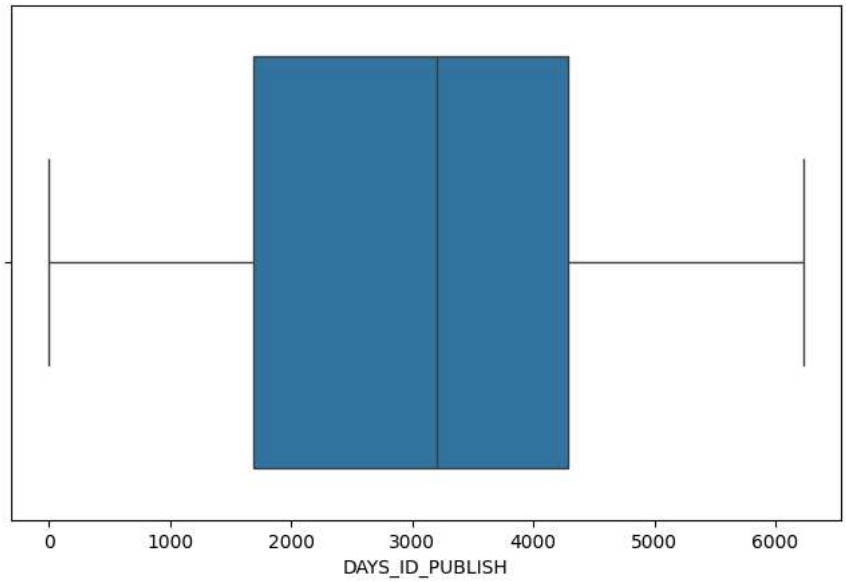

```
----------------------------
count     3864.000000
mean     16048.218427
std       4340.195881
min       7705.000000
25%      12418.000000
50%      15861.000000
75%      19620.750000
max      25160.000000
Name: DAYS_BIRTH, dtype: float64
```



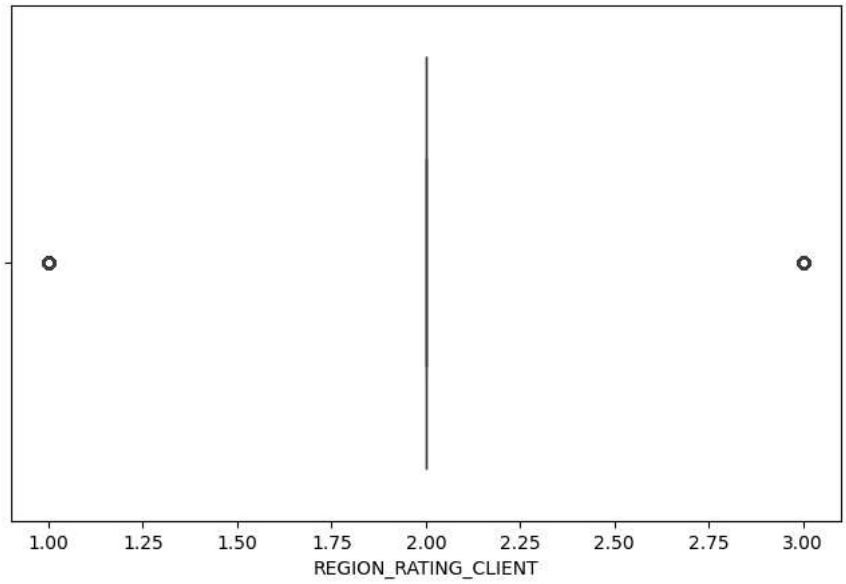

```
----------------------------
count      3864.000000
mean      69125.606625
std      140650.217003
min          17.000000
25%         912.750000
50%        2196.500000
75%        5820.250000
max      365243.000000
Name: DAYS_EMPLOYED, dtype: float64
```




```
----------------------------
count    3864.000000
mean     2969.757246
std      1511.374557
min         0.000000
25%      1693.750000
50%      3199.000000
75%      4288.250000
max      6228.000000
Name: DAYS_ID_PUBLISH, dtype: float64
```

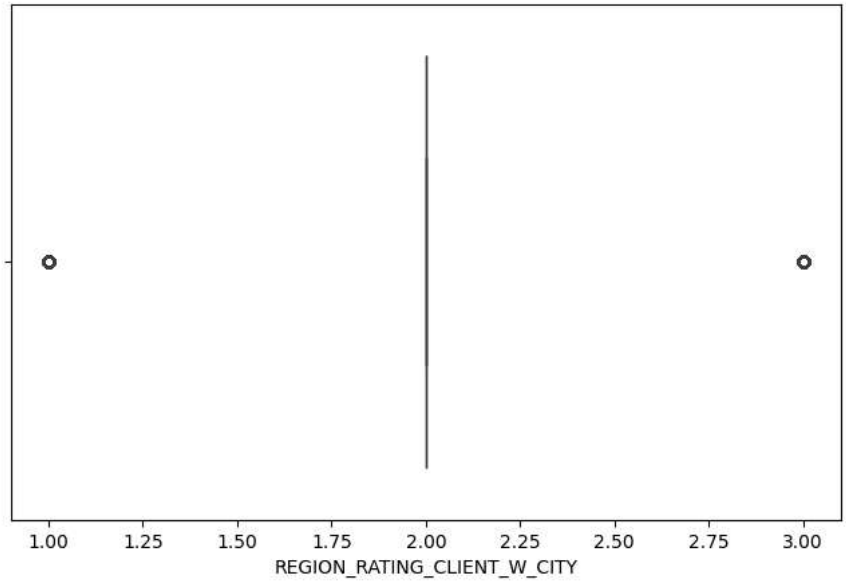



```
----------------------------
count    3864.000000
mean        2.046843
std         0.508413
min         1.000000
25%         2.000000
50%         2.000000
75%         2.000000
max         3.000000
Name: REGION_RATING_CLIENT, dtype: float64
```

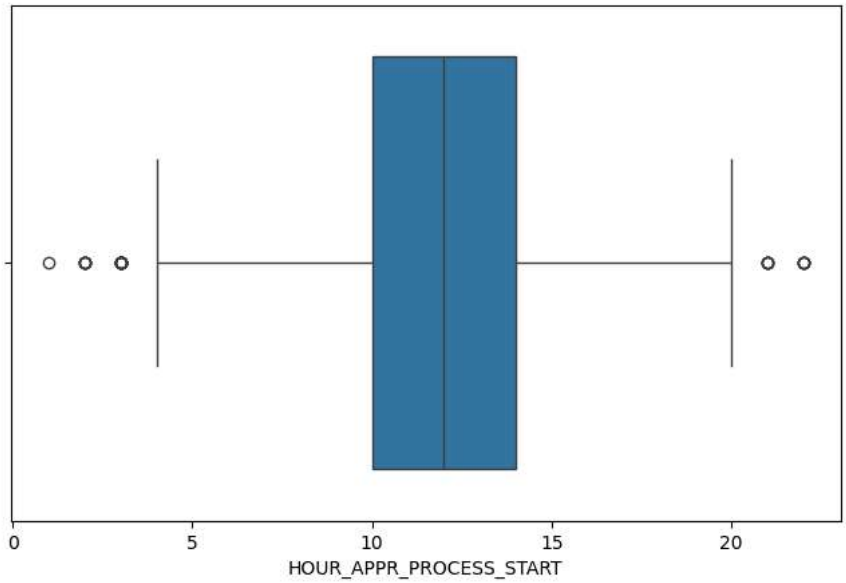



```
----------------------------
count    3864.000000
```

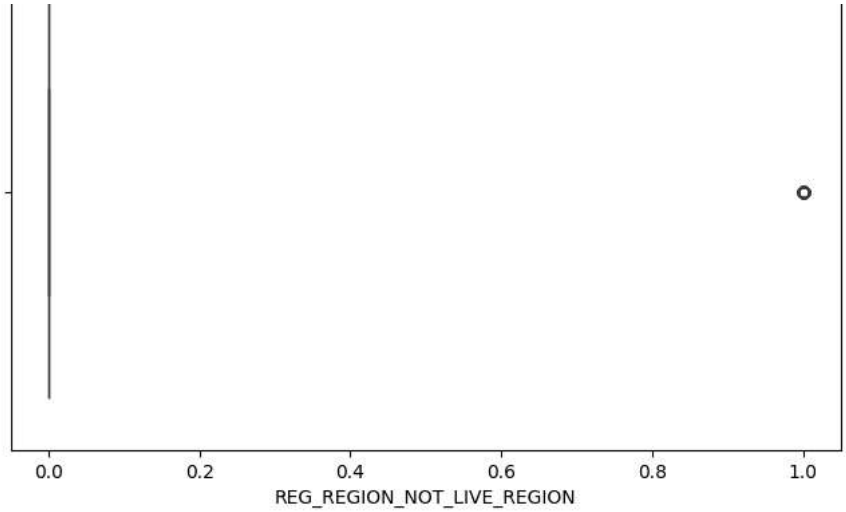
```
count    3864.000000
mean        2.026656
std         0.503227
min         1.000000
25%         2.000000
50%         2.000000
75%         2.000000
max         3.000000
Name: REGION_RATING_CLIENT_W_CITY, dtype: float64
```



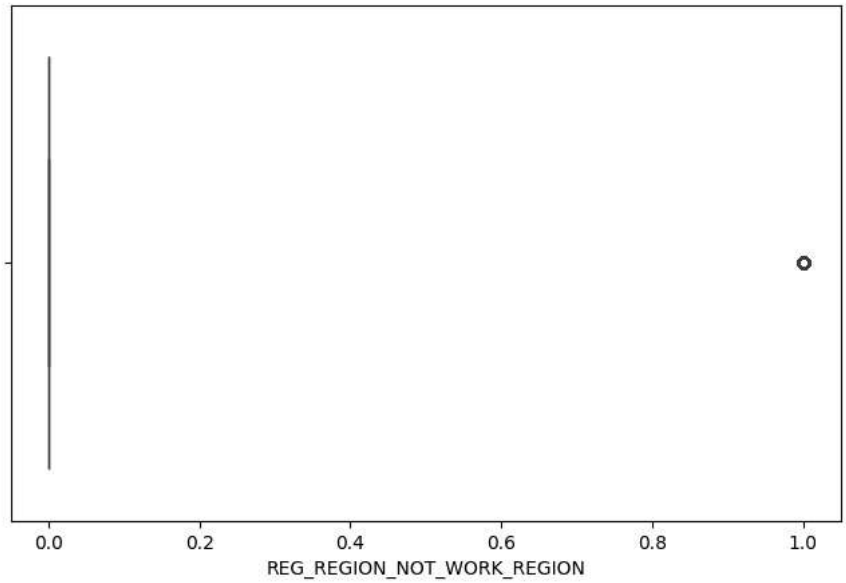

```
----------------------------
count    3864.000000
mean       12.081263
std         3.240649
min         1.000000
25%        10.000000
50%        12.000000
75%        14.000000
max        22.000000
Name: HOUR_APPR_PROCESS_START, dtype: float64
```



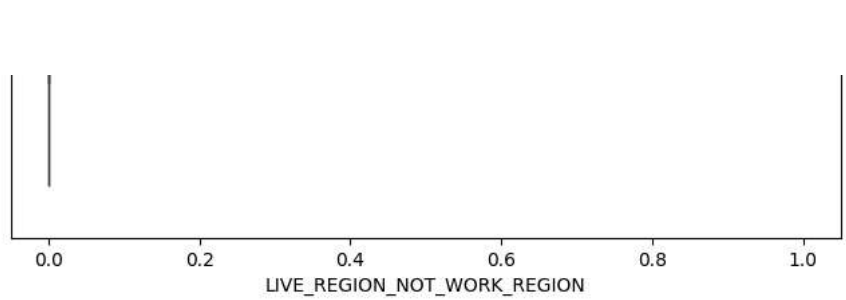

```
----------------------------
count    3864.000000
mean        0.017857
std         0.132449
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: REG_REGION_NOT_LIVE_REGION, dtype: float64
```



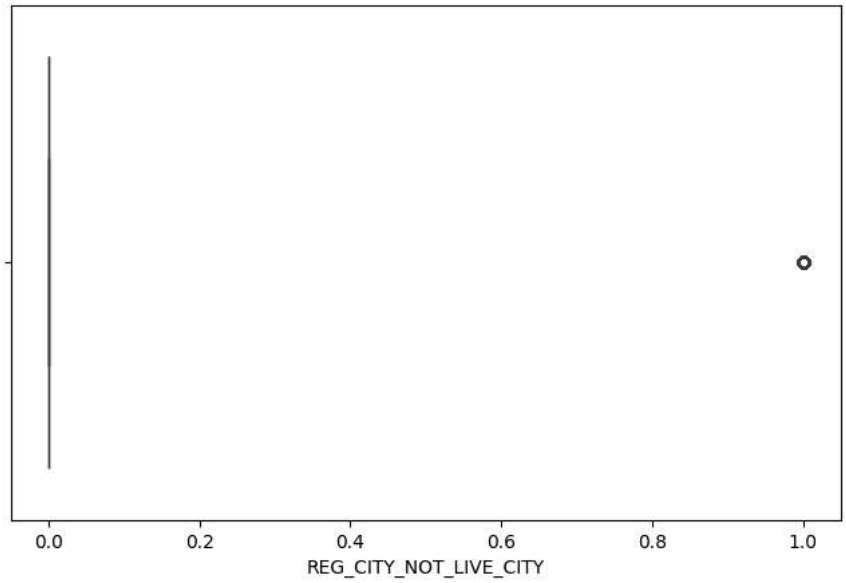

```
----------------------------
count    3864.000000
mean        0.049431
std         0.216793
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: REG_REGION_NOT_WORK_REGION, dtype: float64
```



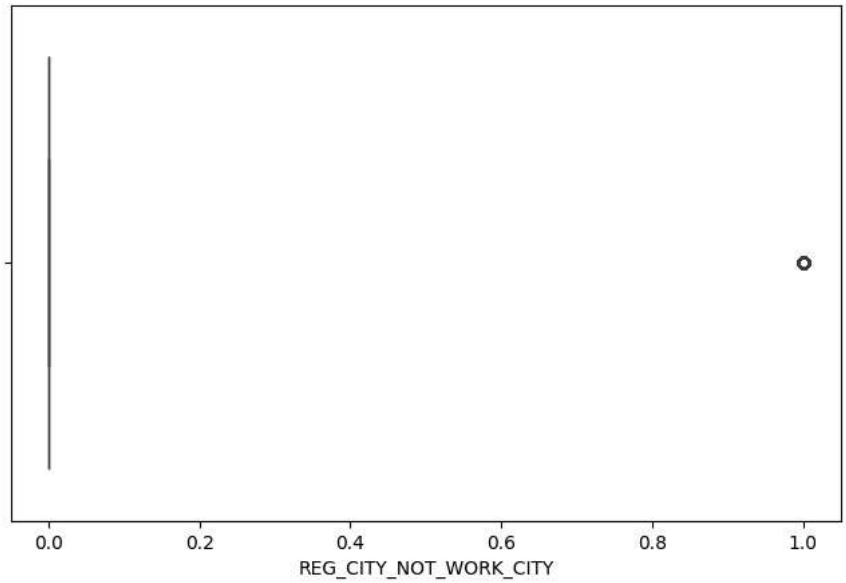

```
----------------------------
count    3864.000000
mean        0.038043
std         0.191326
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: LIVE_REGION_NOT_WORK_REGION, dtype: float64
```




```
----------------------------
count    3864.000000
mean        0.080487
std         0.272080
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: REG_CITY_NOT_LIVE_CITY, dtype: float64
```

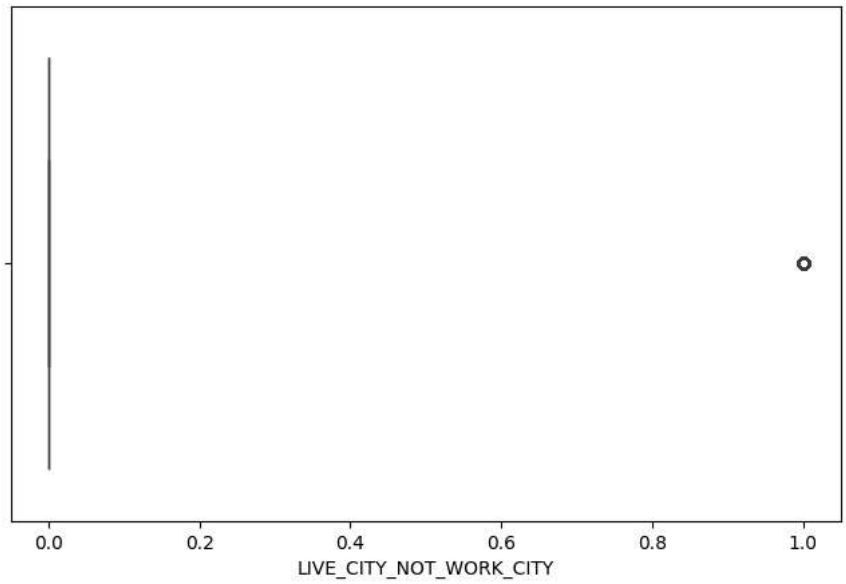



```
----------------------------
count    3864.000000
mean        0.239389
std         0.426766
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: REG_CITY_NOT_WORK_CITY, dtype: float64
```

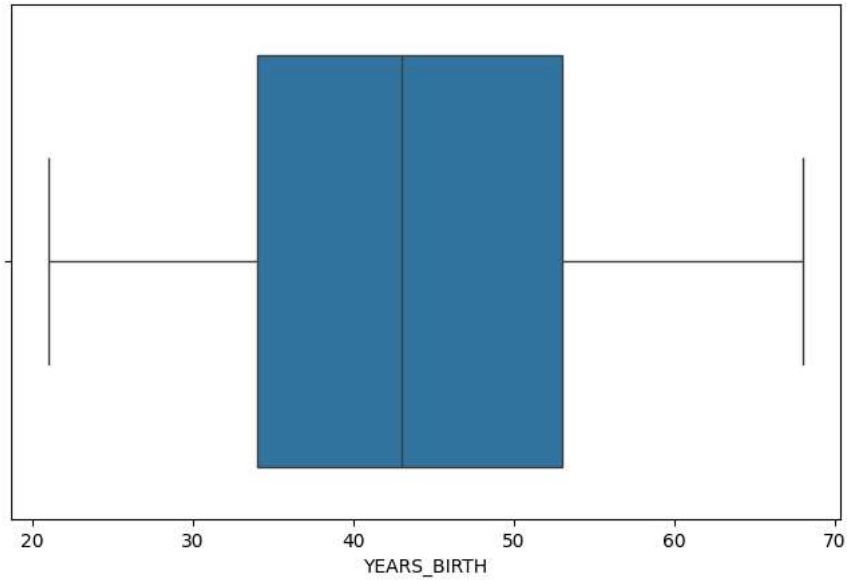



```
----------------------------
count    3864.000000
```

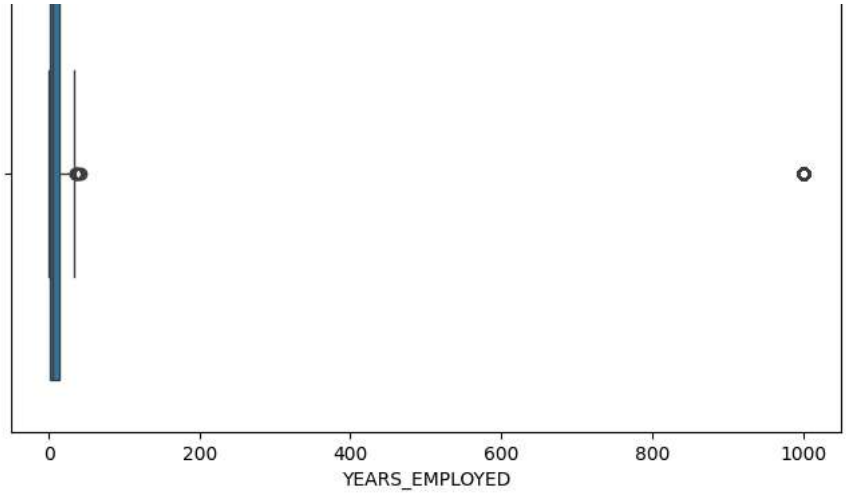
```
mean        0.183489
std         0.387117
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: LIVE_CITY_NOT_WORK_CITY, dtype: float64
```



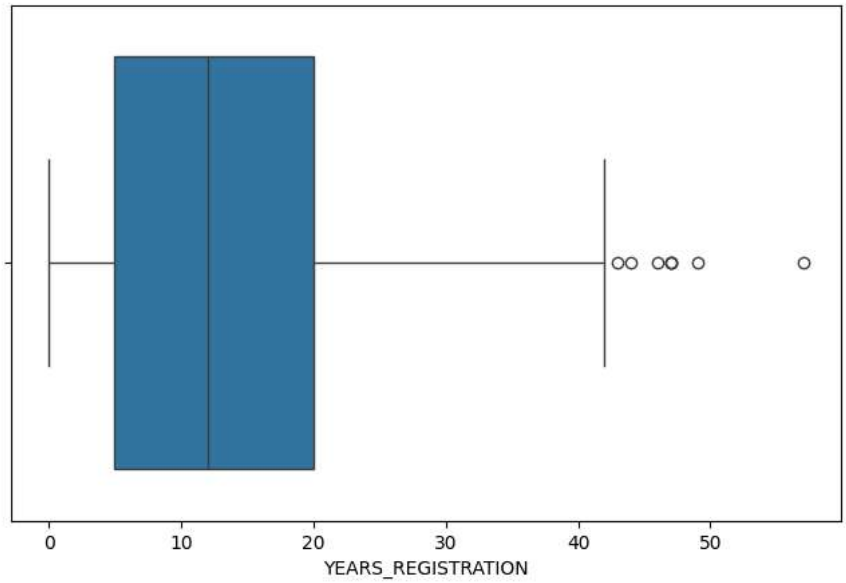

```
----------------------------
count    3864.000000
mean       43.474638
std        11.893713
min        21.000000
25%        34.000000
50%        43.000000
75%        53.000000
max        68.000000
Name: YEARS_BIRTH, dtype: float64
```



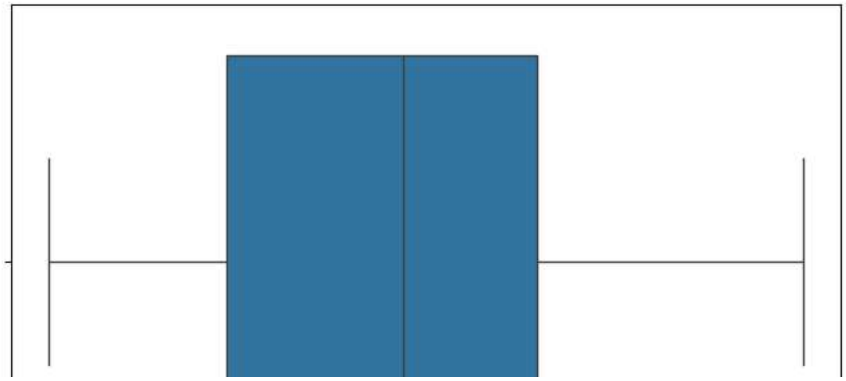

```
----------------------------
count    3864.000000
mean      188.862578
std       385.275231
min         0.000000
25%         2.000000
50%         6.000000
75%        15.000000
max      1000.000000
Name: YEARS_EMPLOYED, dtype: float64
```

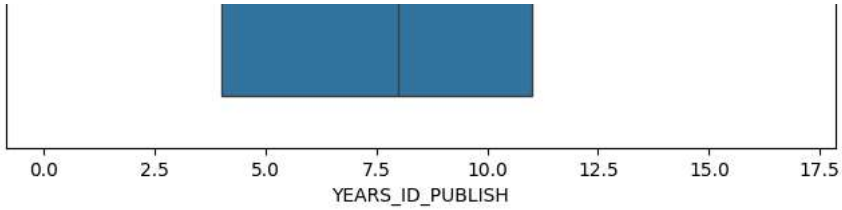



```
----------------------------
count    3864.000000
mean       13.127329
std         9.782535
min         0.000000
25%         5.000000
50%        12.000000
75%        20.000000
max        57.000000
Name: YEARS_REGISTRATION, dtype: float64
```

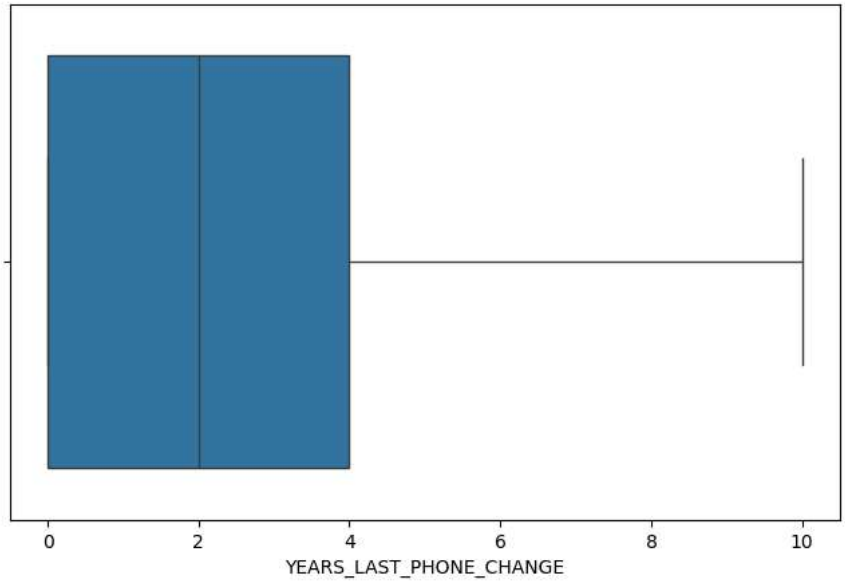



```
----------------------------
count    3864.000000
mean        7.640528
std         4.140732
min         0.000000
25%         4.000000
50%         8.000000
75%        11.000000
max        17.000000
Name: YEARS_ID_PUBLISH, dtype: float64
```

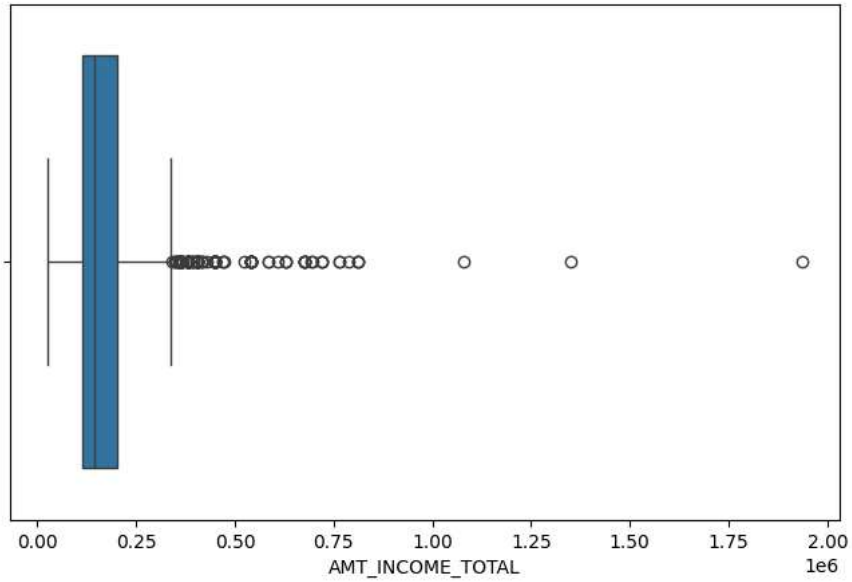





```
----------------------------
count    3864.000000
mean        2.236284
std         2.183723
min         0.000000
25%         0.000000
50%         2.000000
75%         4.000000
max        10.000000
Name: YEARS_LAST_PHONE_CHANGE, dtype: float64
```

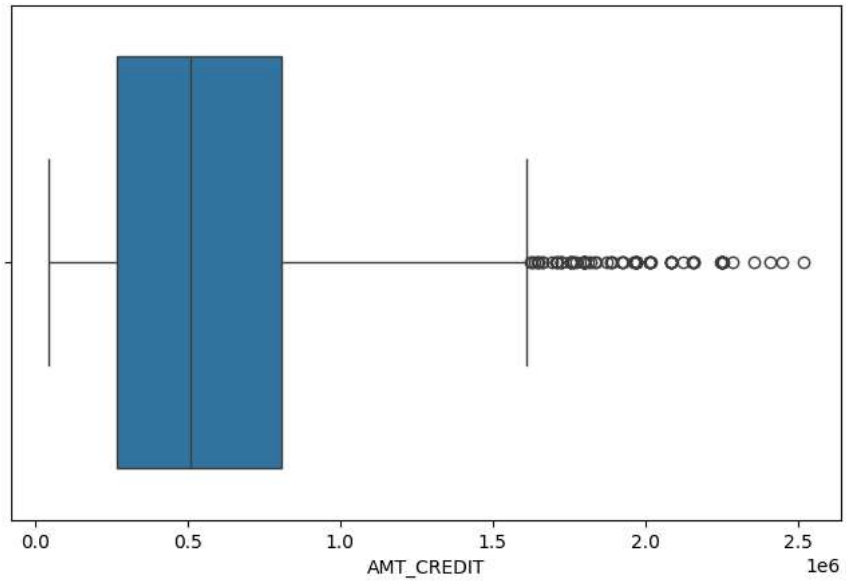



```
----------------------------
count    3.864000e+03
mean     1.685222e+05
std      9.938526e+04
min      2.565000e+04
25%      1.125000e+05
50%      1.440000e+05
75%      2.025000e+05
max      1.935000e+06
Name: AMT_INCOME_TOTAL, dtype: float64
```

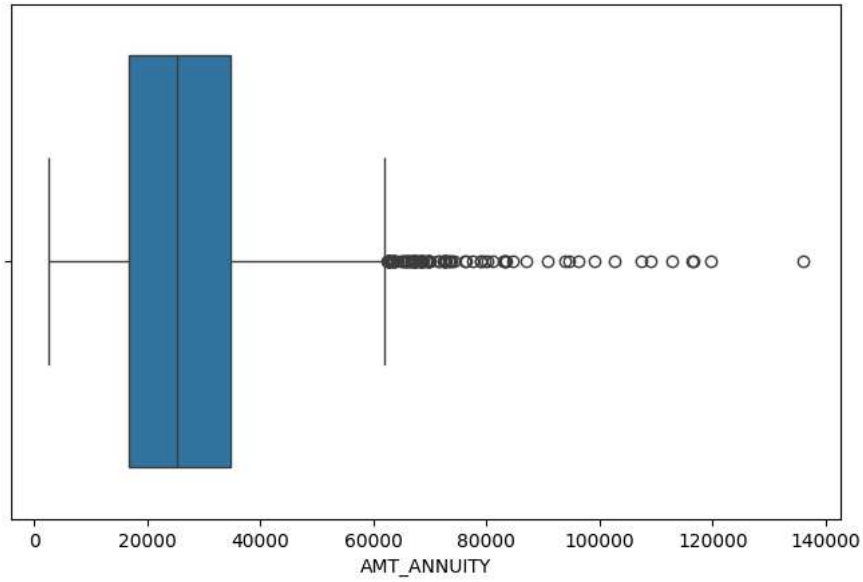



```
----------------------------
count    3.864000e+03
mean     6.017847e+05
```

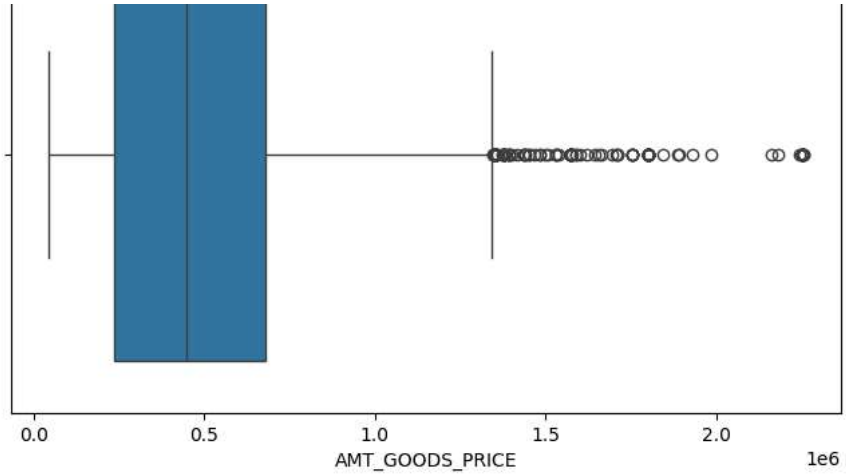
```
std      4.033991e+05
min      4.500000e+04
25%      2.700000e+05
50%      5.095012e+05
75%      8.086500e+05
max      2.517300e+06
Name: AMT_CREDIT, dtype: float64
```



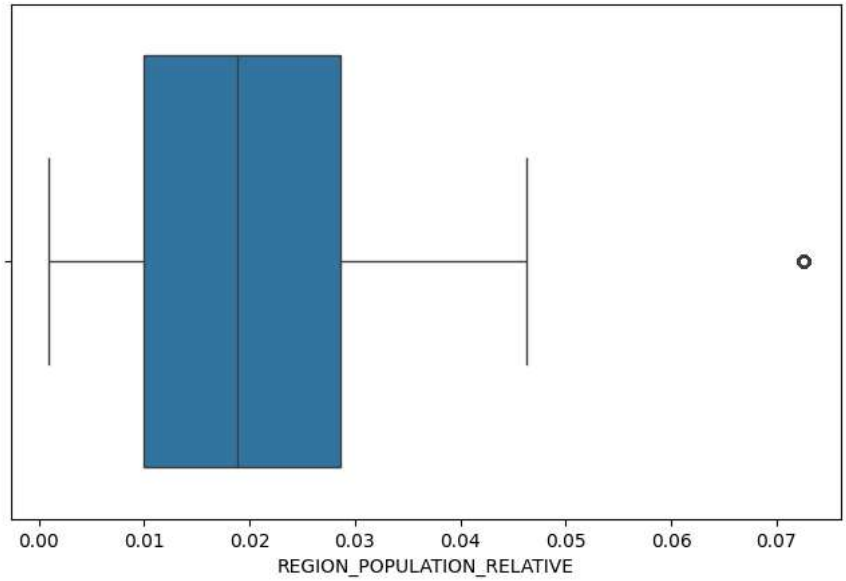

```
----------------------------
count      3864.000000
mean      27239.107919
std       14522.447899
min        2596.500000
25%       16713.000000
50%       25195.500000
75%       34845.750000
max      135936.000000
Name: AMT_ANNUITY, dtype: float64
```



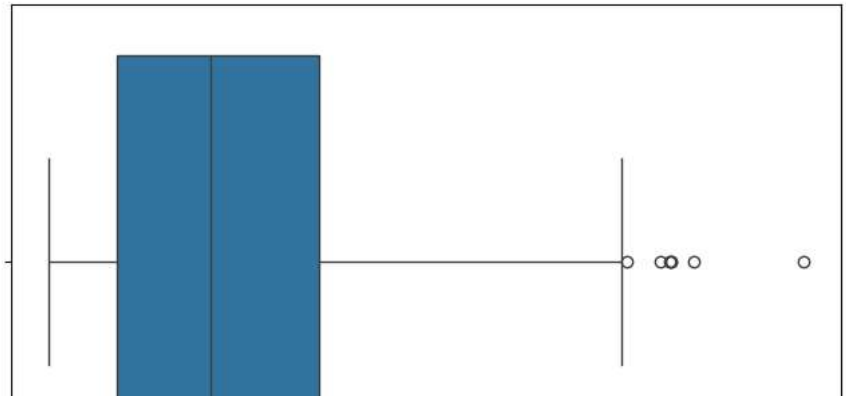

```
----------------------------
count    3.864000e+03
mean     5.422328e+05
std      3.711752e+05
min      4.500000e+04
25%      2.385000e+05
50%      4.500000e+05
75%      6.795000e+05
max      2.254500e+06
Name: AMT_GOODS_PRICE, dtype: float64
```

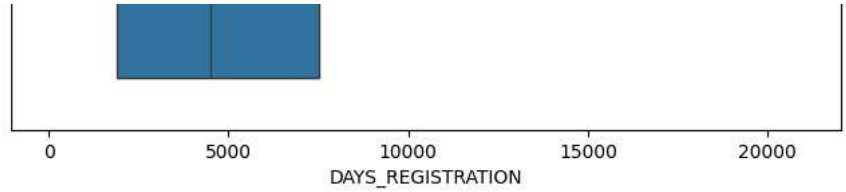



```
----------------------------
count    3864.000000
mean        0.021044
std         0.014162
min         0.000938
25%         0.010006
50%         0.018850
75%         0.028663
max         0.072508
Name: REGION_POPULATION_RELATIVE, dtype: float64
```

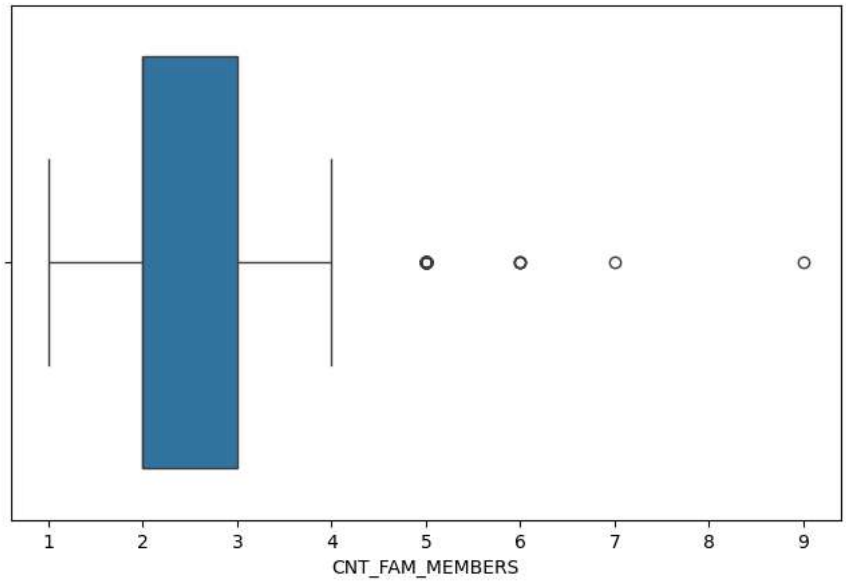



```
----------------------------
count     3864.000000
mean      4971.123447
std       3573.020833
min          0.000000
25%       1899.500000
50%       4493.500000
75%       7520.000000
max      20981.000000
Name: DAYS_REGISTRATION, dtype: float64
```

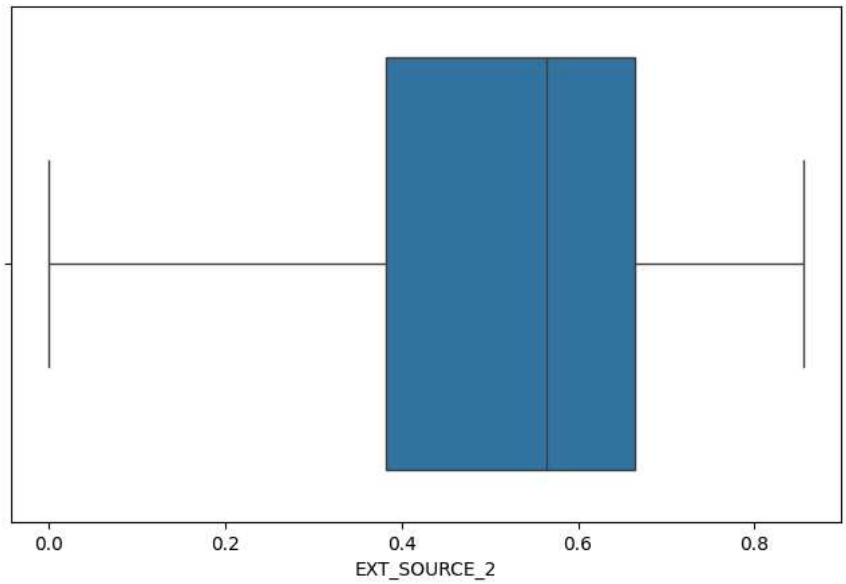





```
----------------------------
count    3864.000000
mean        2.136387
std         0.904318
min         1.000000
25%         2.000000
50%         2.000000
75%         3.000000
max         9.000000
Name: CNT_FAM_MEMBERS, dtype: float64
```

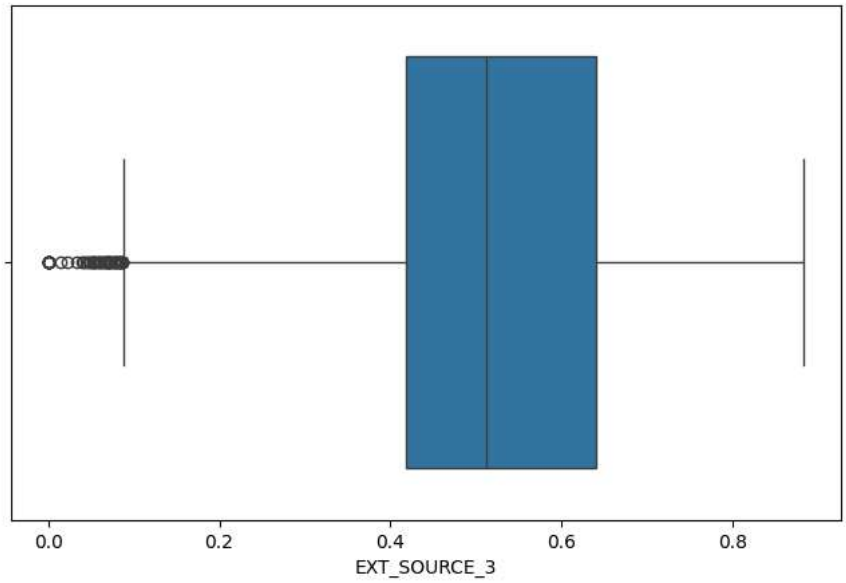



```
----------------------------
count    3864.000000
mean        0.512490
std         0.192023
min         0.000256
25%         0.382666
50%         0.564215
75%         0.664483
max         0.855000
Name: EXT_SOURCE_2, dtype: float64
```

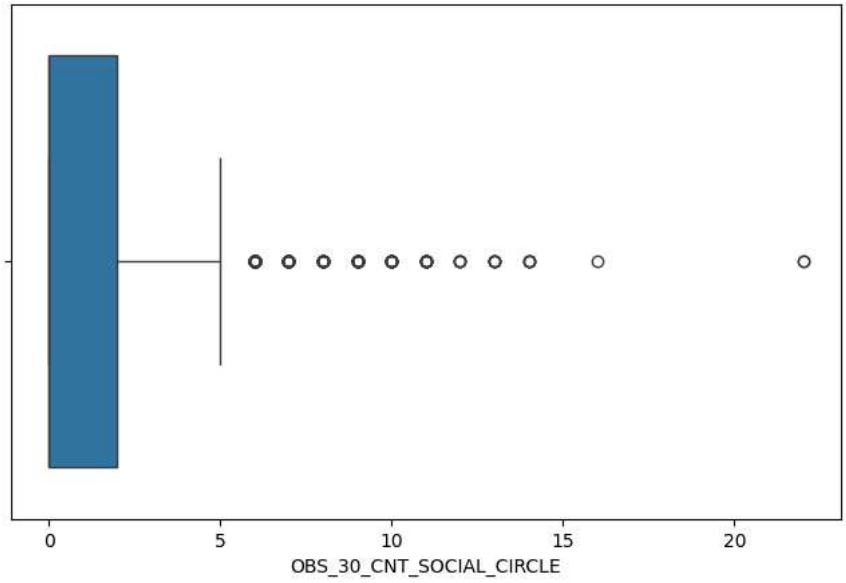



```
----------------------------
count    3864.000000
mean        0.512437
std         0.174843
```

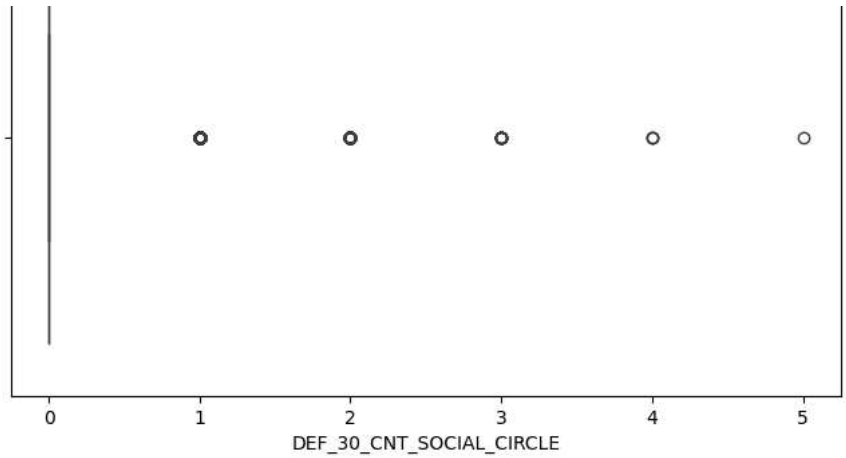
```
min         0.000527
25%         0.418854
50%         0.512437
75%         0.639708
max         0.882530
Name: EXT_SOURCE_3, dtype: float64
```



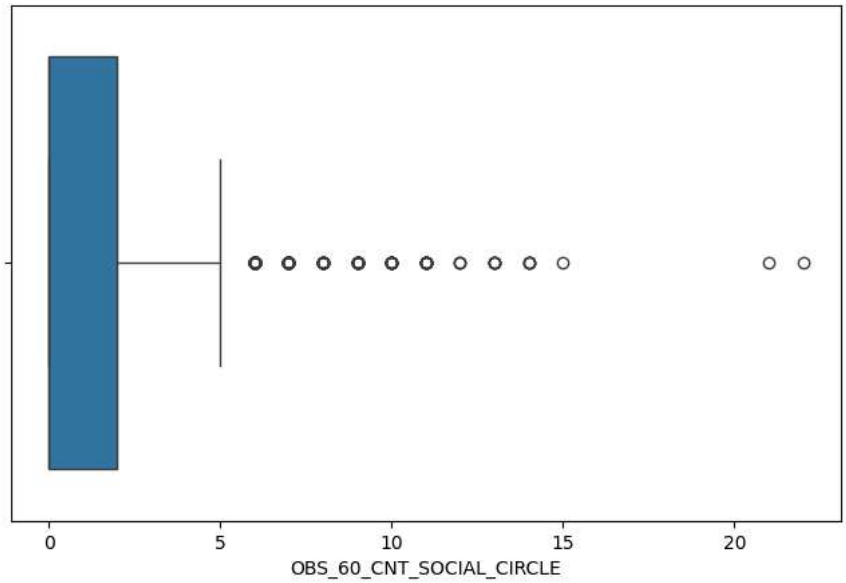

```
----------------------------
count    3864.000000
mean        1.393892
std         2.253797
min         0.000000
25%         0.000000
50%         0.000000
75%         2.000000
max        22.000000
Name: OBS_30_CNT_SOCIAL_CIRCLE, dtype: float64
```



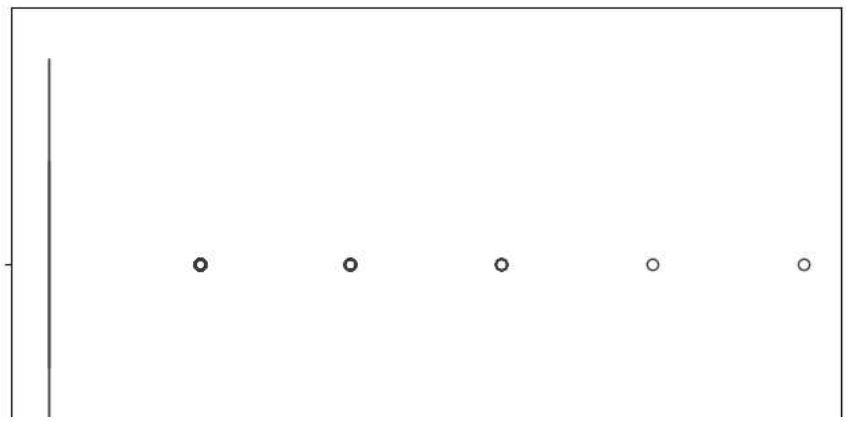

```
----------------------------
count    3864.000000
mean        0.144669
std         0.447078
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         5.000000
Name: DEF_30_CNT_SOCIAL_CIRCLE, dtype: float64
```

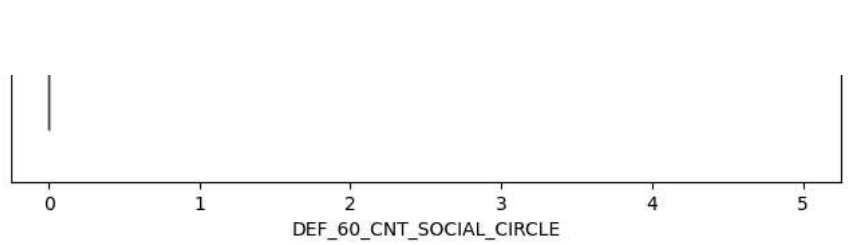



```
----------------------------
count    3864.000000
mean        1.379658
std         2.232537
min         0.000000
25%         0.000000
50%         0.000000
75%         2.000000
max        22.000000
Name: OBS_60_CNT_SOCIAL_CIRCLE, dtype: float64
```

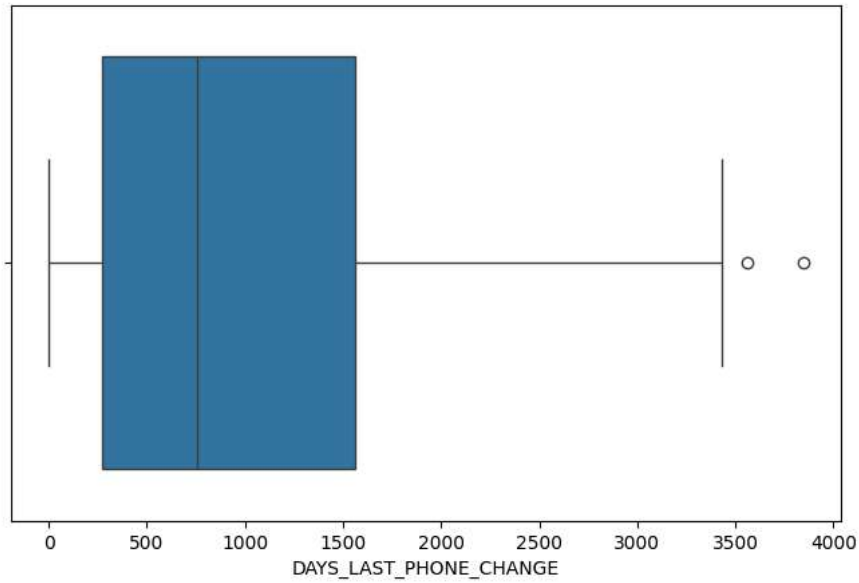



```
----------------------------
count    3864.000000
mean        0.102226
std         0.372015
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         5.000000
Name: DEF_60_CNT_SOCIAL_CIRCLE, dtype: float64
```

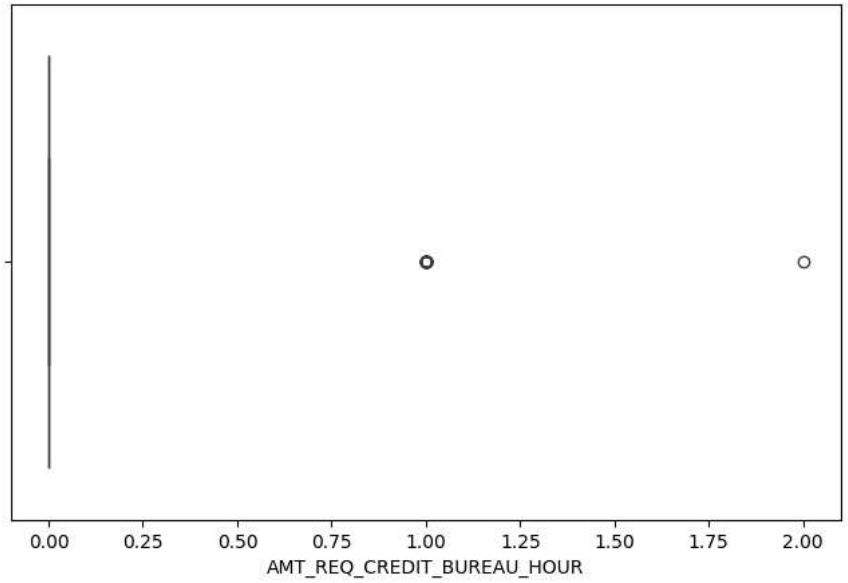





```
----------------------------
count    3864.000000
mean      966.462215
std       823.195739
min         0.000000
25%       272.000000
50%       758.000000
75%      1565.000000
max      3845.000000
Name: DAYS_LAST_PHONE_CHANGE, dtype: float64
```

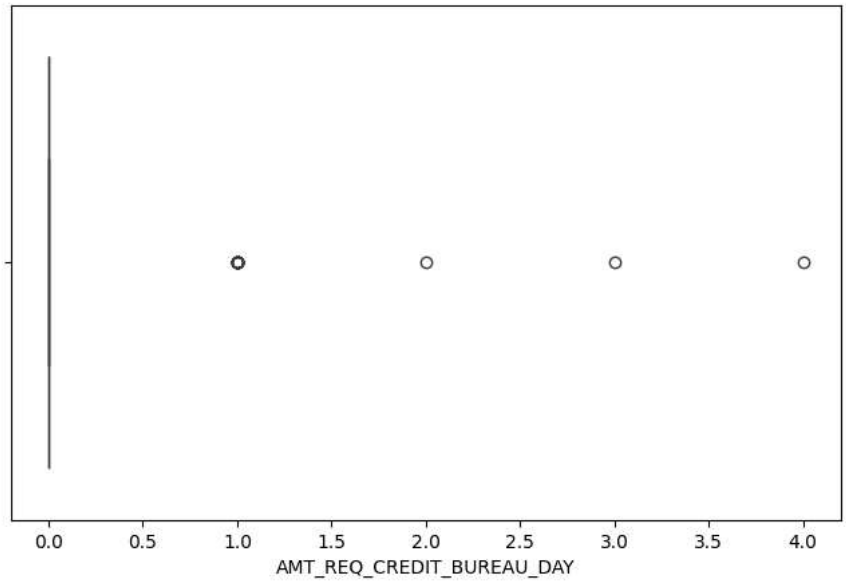



```
----------------------------
count    3864.000000
mean        0.008282
std         0.093450
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         2.000000
Name: AMT_REQ_CREDIT_BUREAU_HOUR, dtype: float64
```

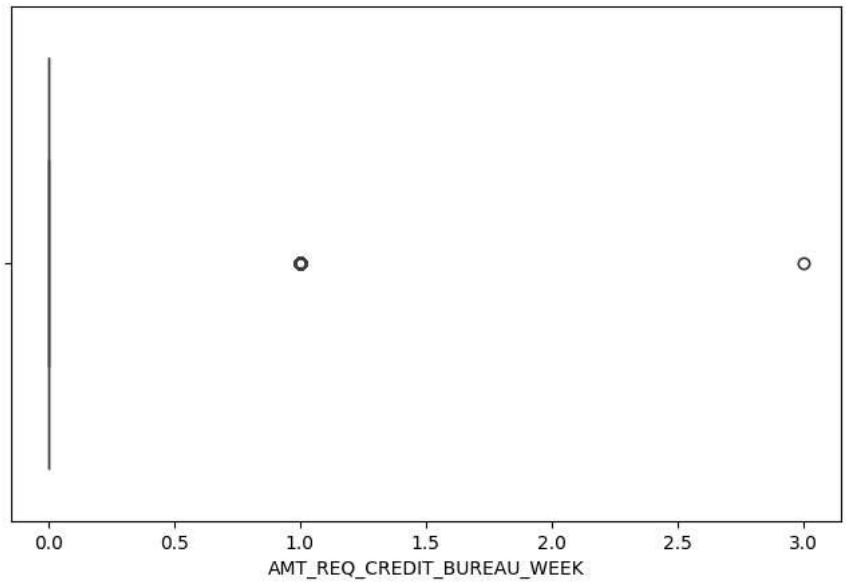



```
----------------------------
count    3864.000000
mean        0.007246
std         0.111234
min         0.000000
```

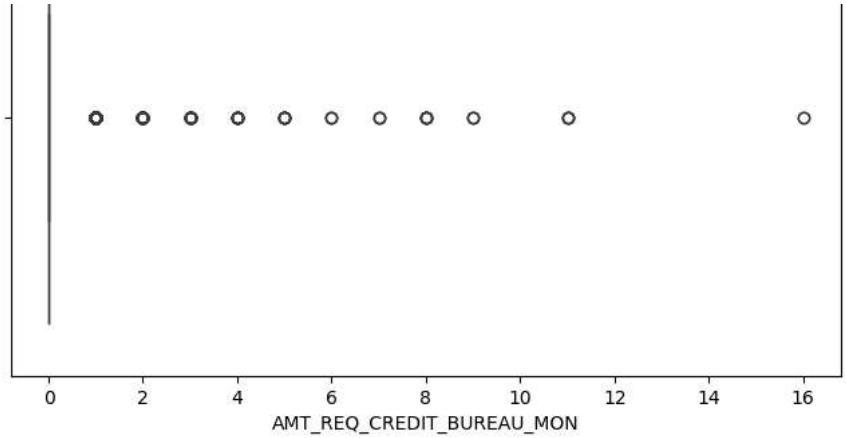
```
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         4.000000
Name: AMT_REQ_CREDIT_BUREAU_DAY, dtype: float64
```



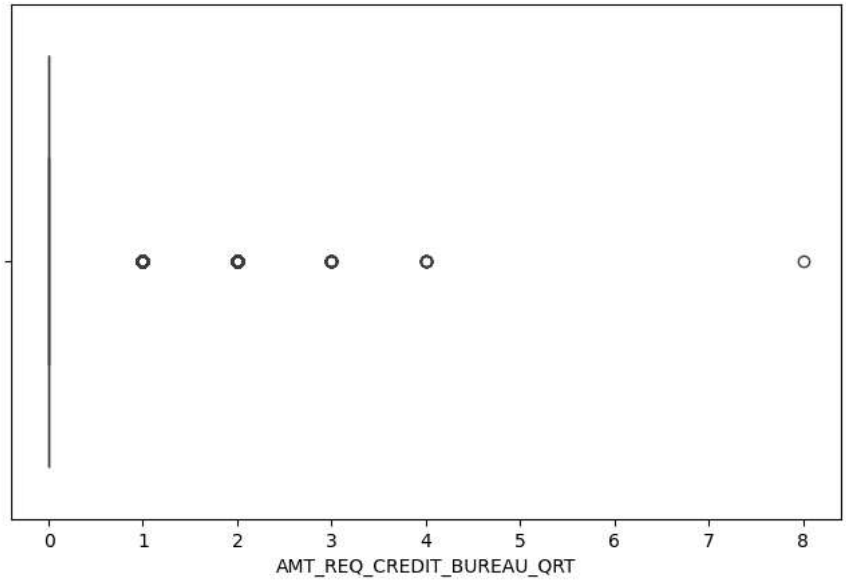

```
----------------------------
count    3864.000000
mean        0.030797
std         0.181556
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         3.000000
Name: AMT_REQ_CREDIT_BUREAU_WEEK, dtype: float64
```



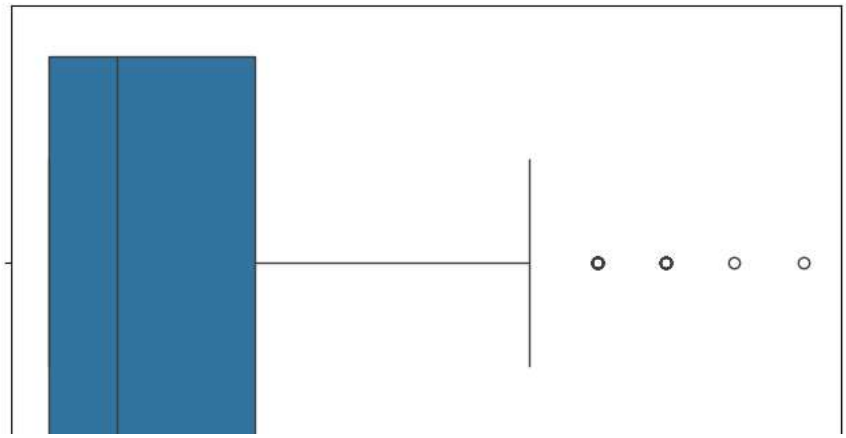

```
----------------------------
count    3864.000000
mean        0.230072
std         0.874831
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max        16.000000
Name: AMT_REQ_CREDIT_BUREAU_MON, dtype: float64
```

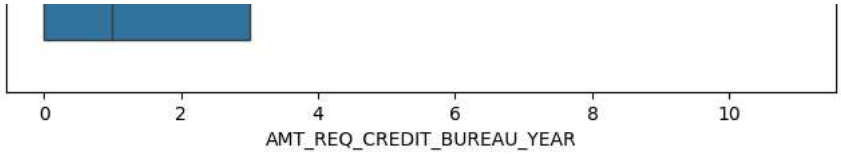



```
----------------------------
count    3864.000000
mean        0.236025
std         0.585604
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         8.000000
Name: AMT_REQ_CREDIT_BUREAU_QRT, dtype: float64
```




```
----------------------------
count    3864.000000
mean        1.616977
std         1.831720
min         0.000000
25%         0.000000
50%         1.000000
75%         3.000000
max        11.000000
Name: AMT_REQ_CREDIT_BUREAU_YEAR, dtype: float64
```





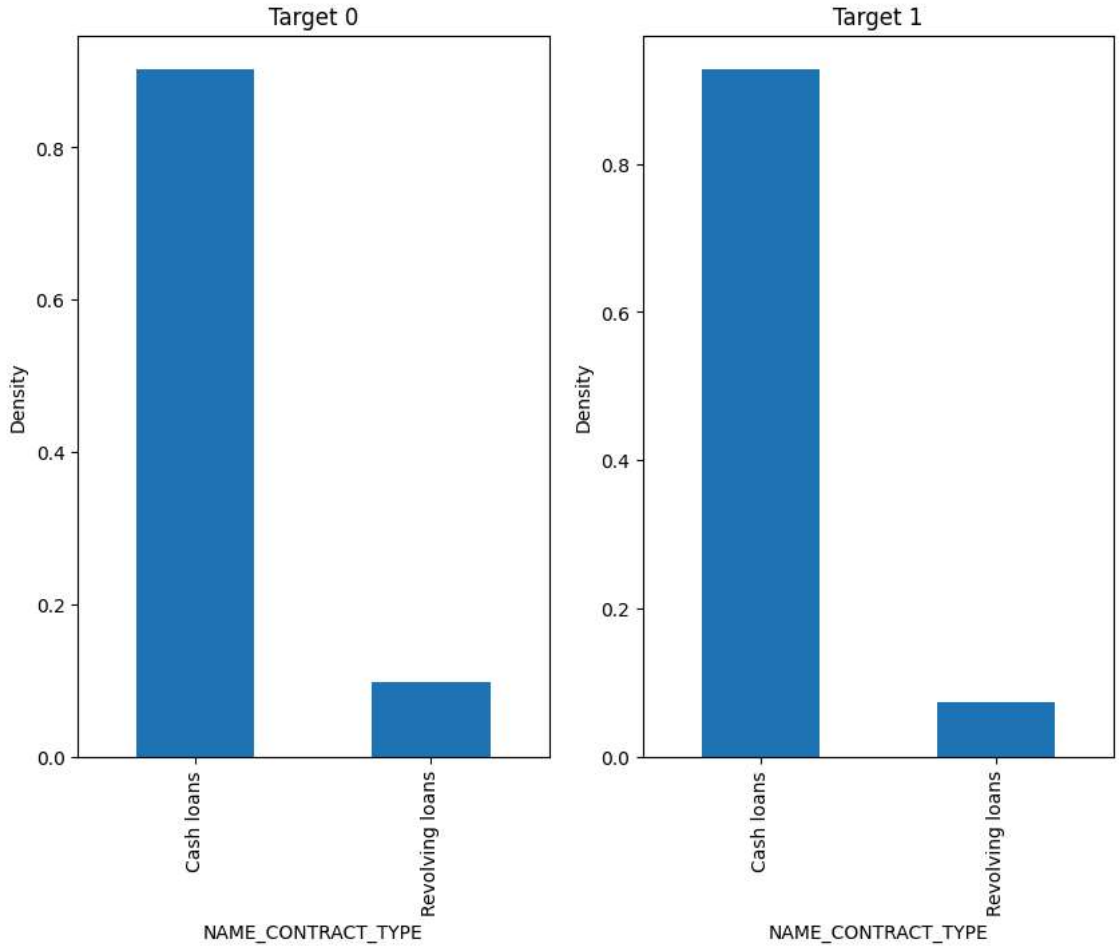
```
# conclusion
#1.AMT_INCOME_TOTAL column has a few outliers and there is a huge difference between the 99th percentile and the max value also we could
#2.AMT_CREDIT column has a few outliers and there is a huge difference between the 99th percentile and the max value also we could see h
#3.AMT_ANNUITY column has a few outliers and there is a huge difference between the 99th percentile and the max value also we could see
#4.AMT_GOODS_PRICE column has a few outliers and there is a huge difference between the 99th percentile and the max value also we could
#5.REGION_POPULATION_RELATIVE column has a one outliers and there is not much difference between the mean and median

```

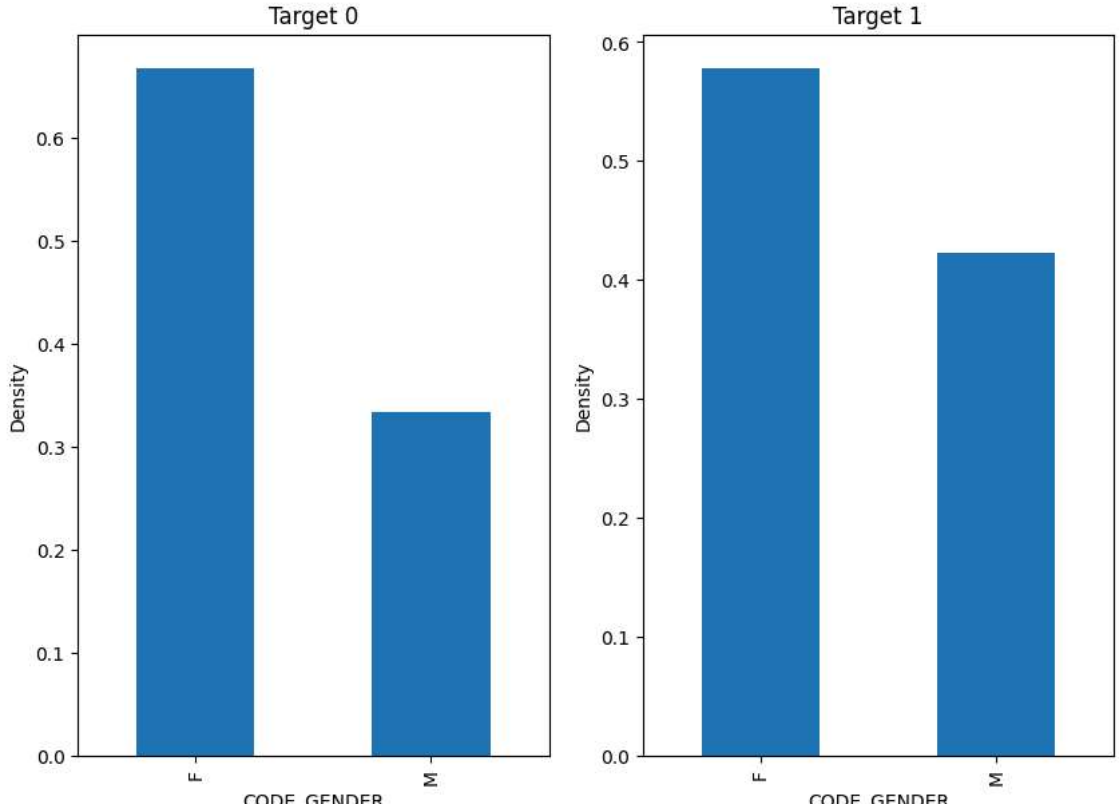
```
# UNivariate analysis on columns with target 0 and 1

for col in cat_cols:
    print(f"Plot on {col} for target 0 and 1")
    plt.figure(figsize=(10,7))
    plt.subplot(1,2,1)
    tar_0[col].value_counts(normalize=True).plot.bar()
    plt.title('Target 0')
    plt.xlabel(col)
    plt.ylabel('Density')
    plt.subplot(1,2,2)
    tar_1[col].value_counts(normalize=True).plot.bar()
    plt.title('Target 1')
    plt.xlabel(col)
    plt.ylabel('Density')
    plt.show()
    print("\n\n----------------------------------------\n\n")
```

Plot on NAME_CONTRACT_TYPE for target 0 and 1

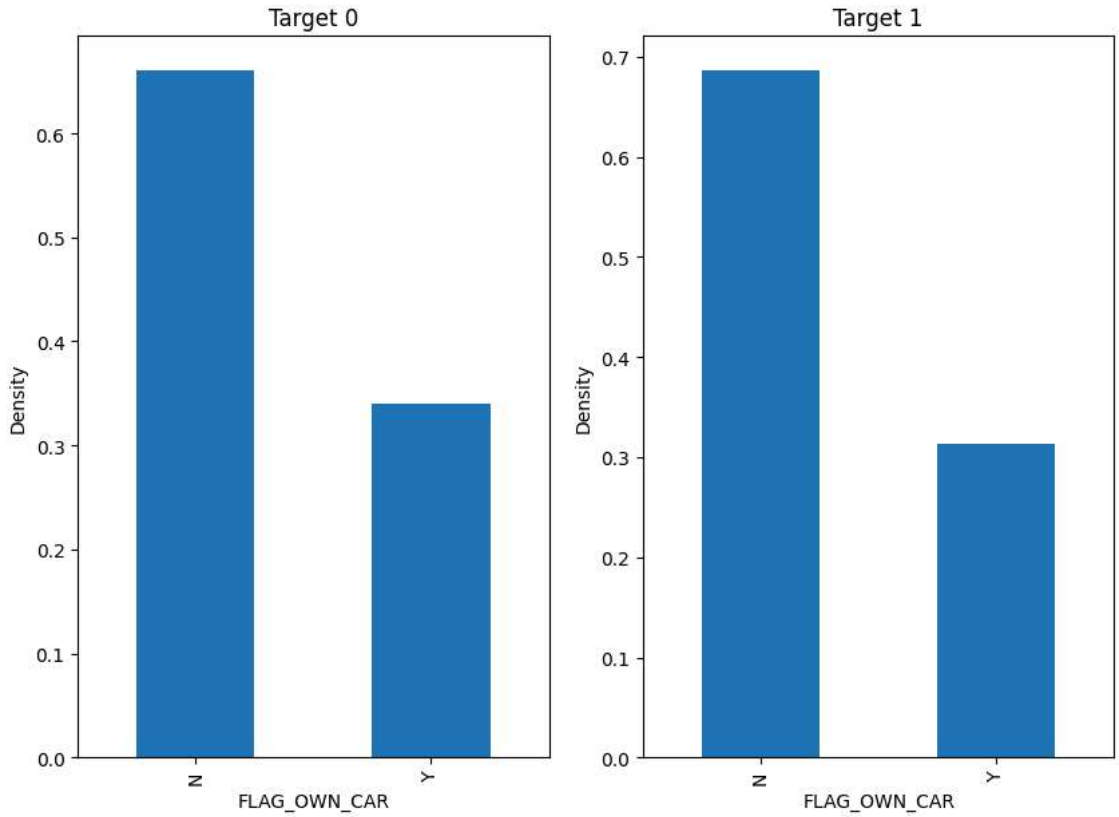


Plot on CODE_GENDER for target 0 and 1

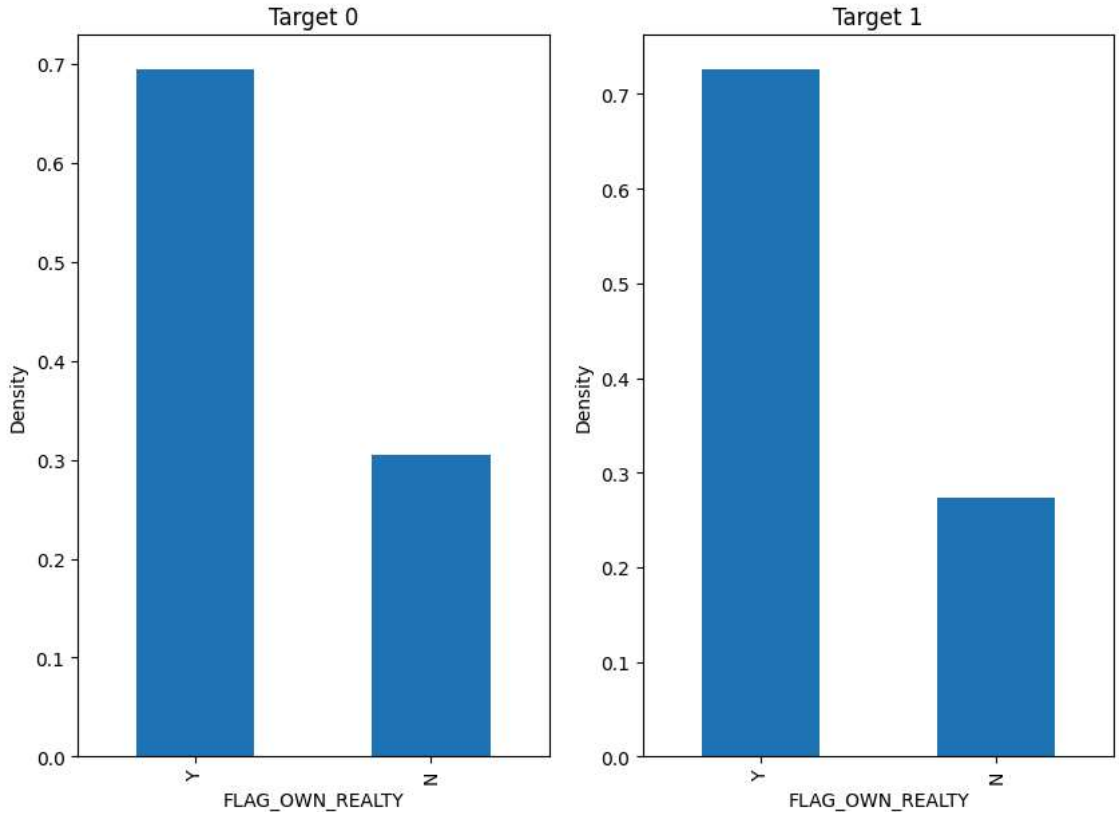


CODE_GENDER CODE_GENDER

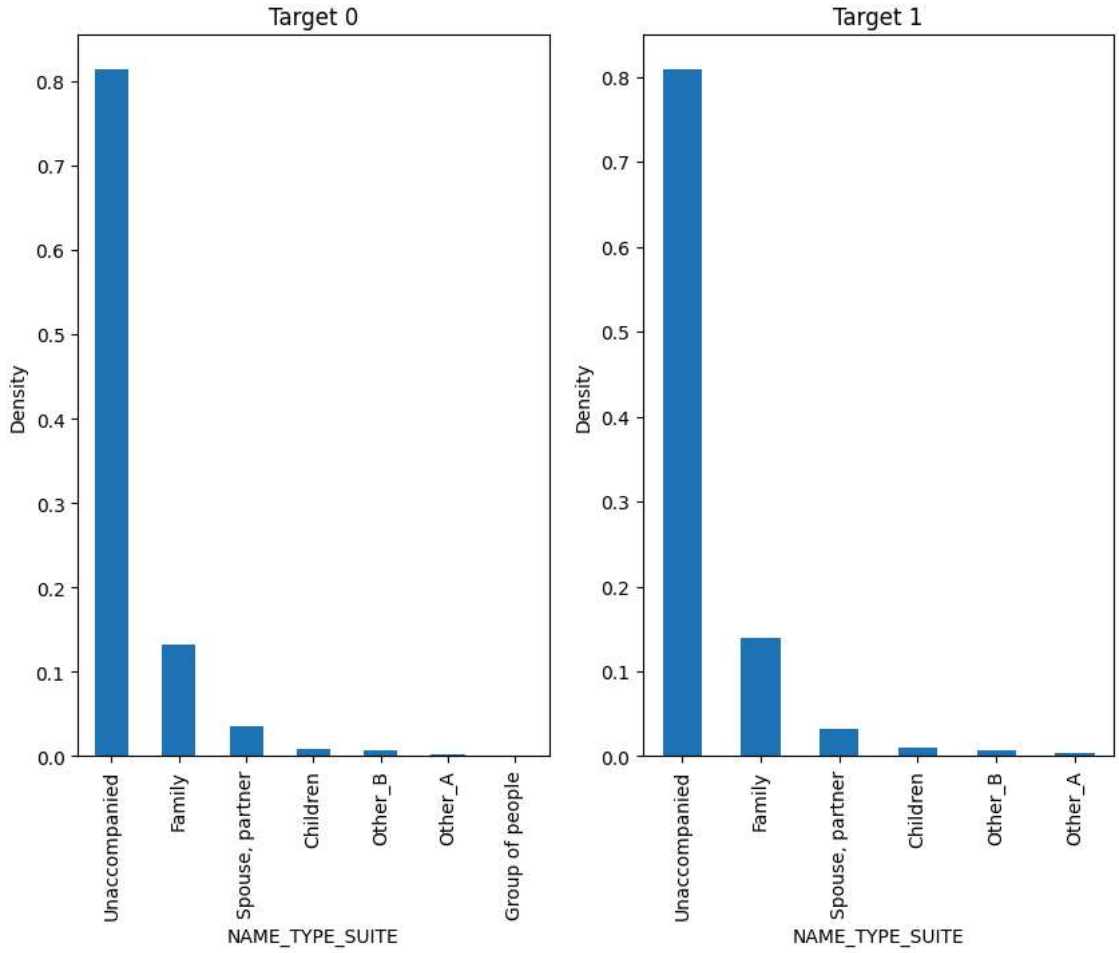
Plot on FLAG_OWN_CAR for target 0 and 1



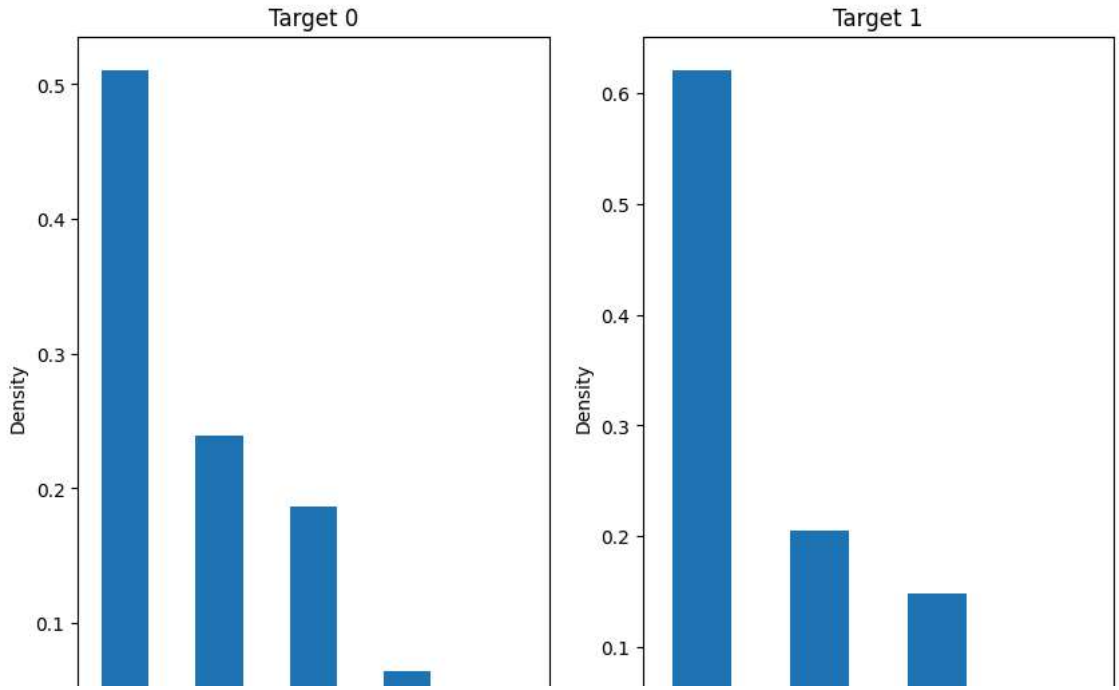
Plot on FLAG_OWN_REALTY for target 0 and 1

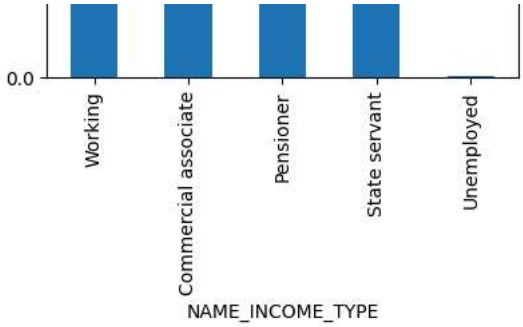


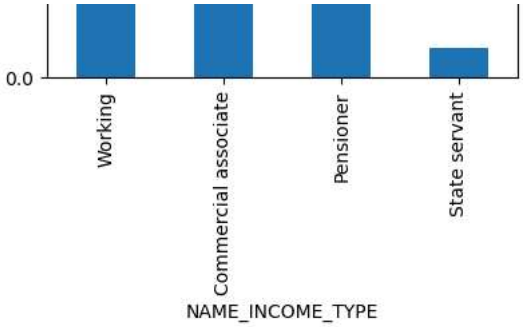
Plot on NAME_TYPE_SUITE for target 0 and 1



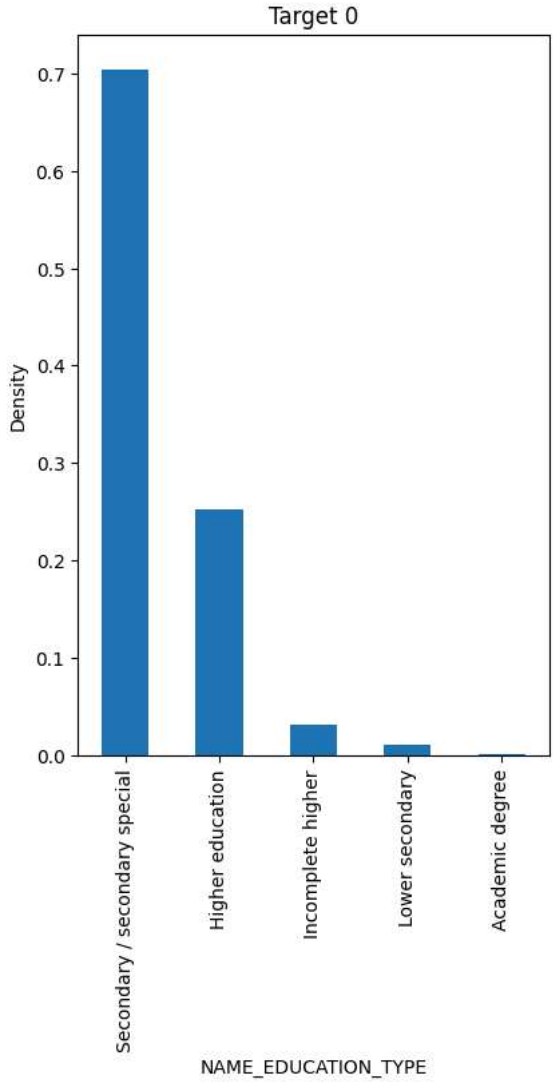
Plot on NAME_INCOME_TYPE for target 0 and 1

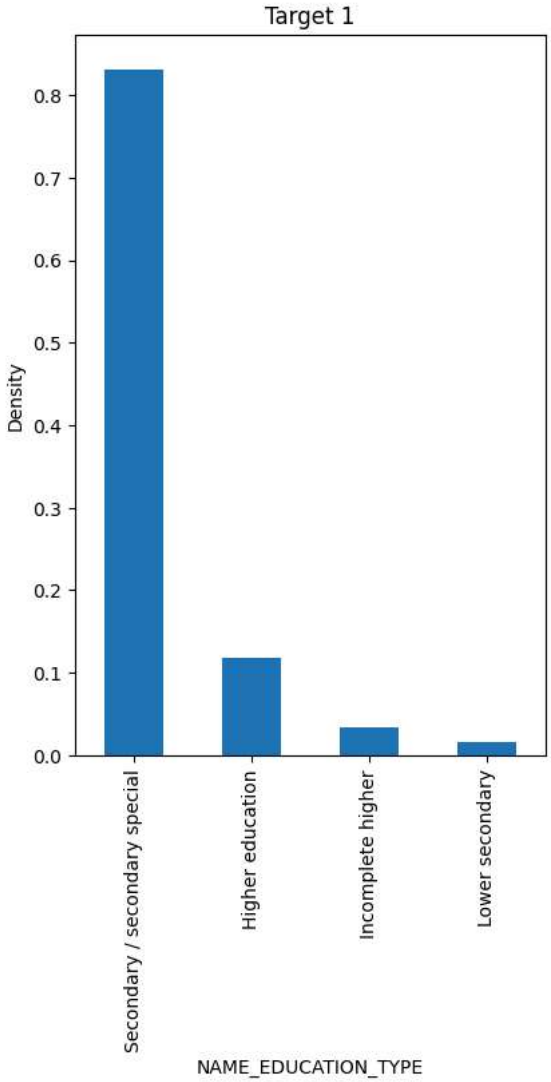




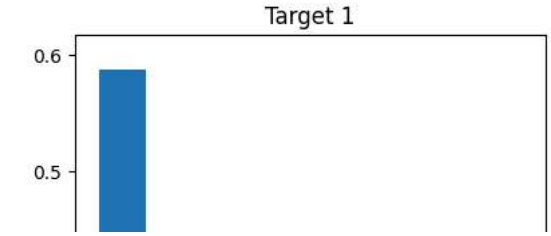


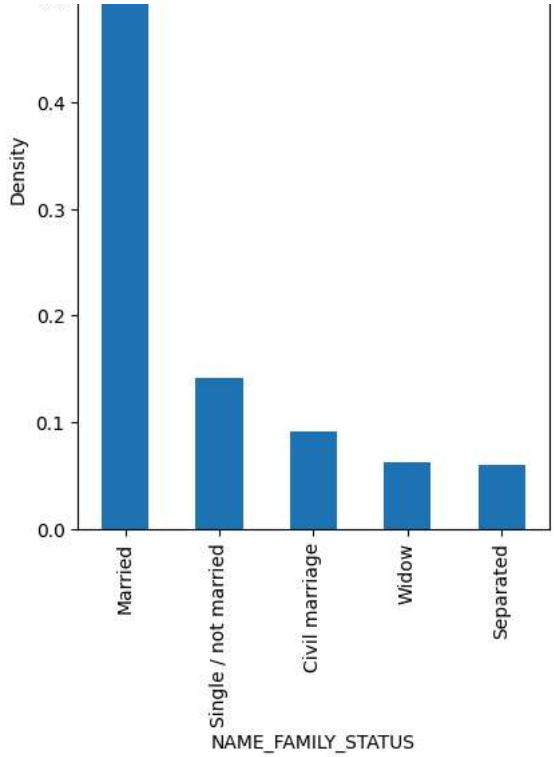
Plot on NAME_EDUCATION_TYPE for target 0 and 1

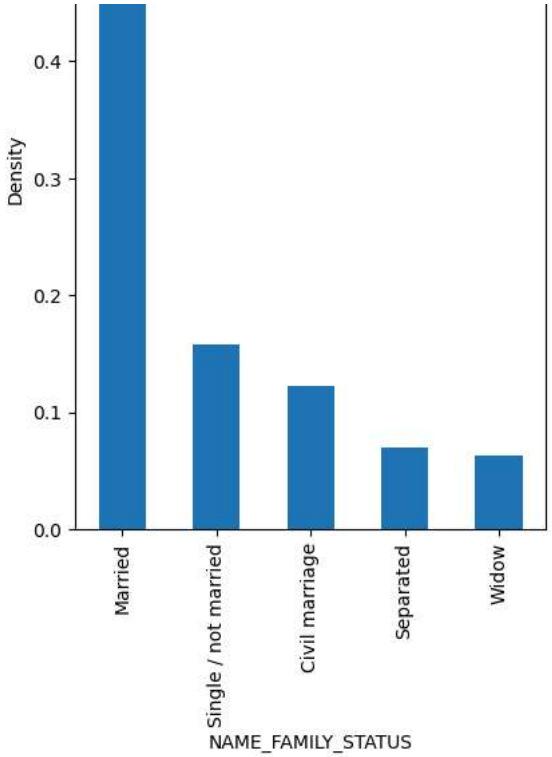




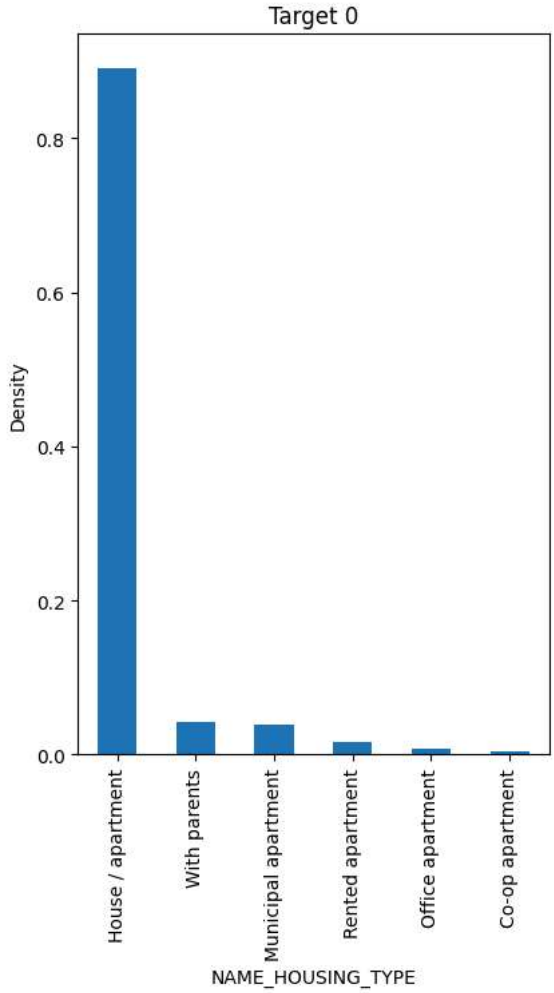
Plot on NAME_FAMILY_STATUS for target 0 and 1

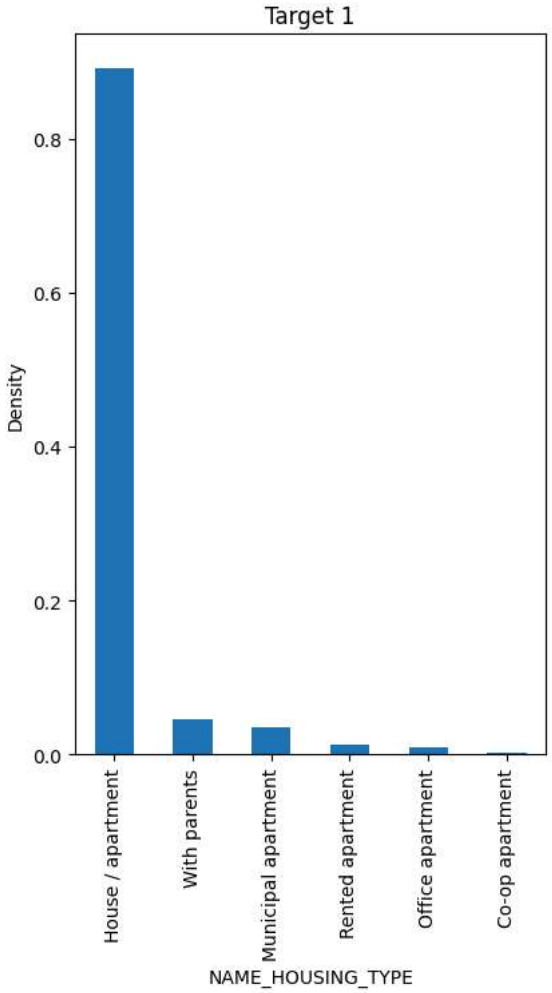




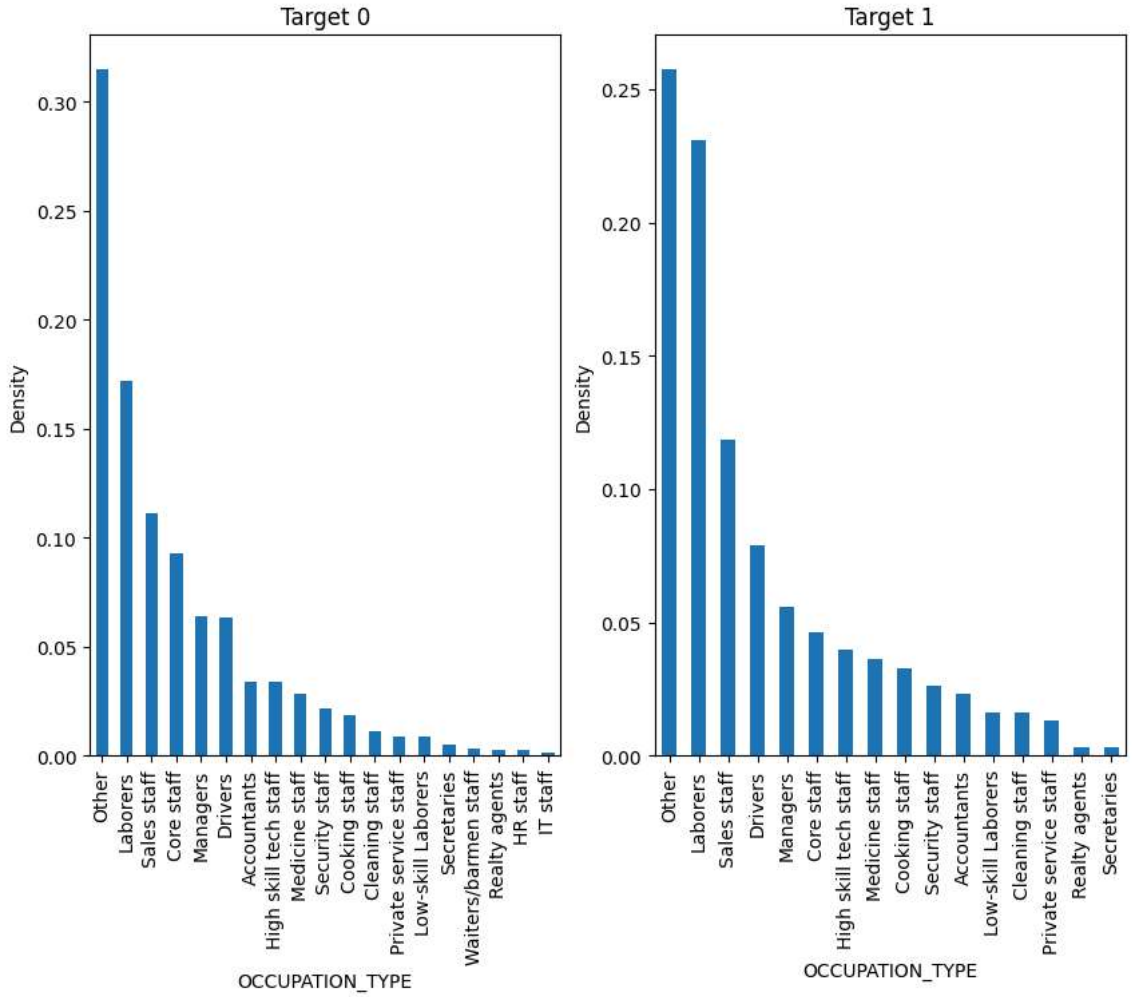


Plot on NAME_HOUSING_TYPE for target 0 and 1

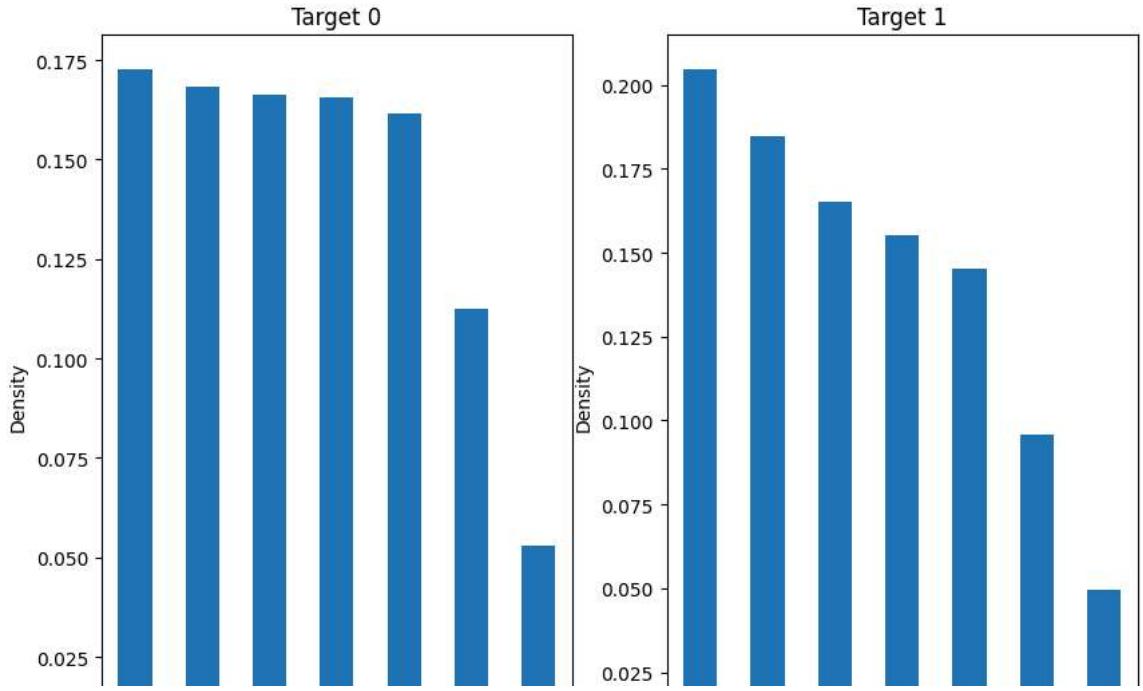


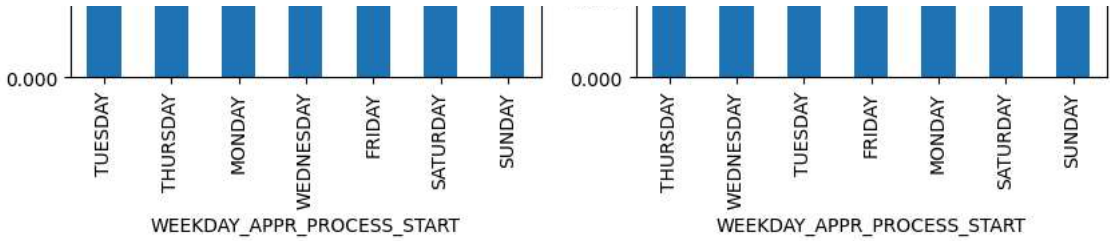


Plot on OCCUPATION_TYPE for target 0 and 1

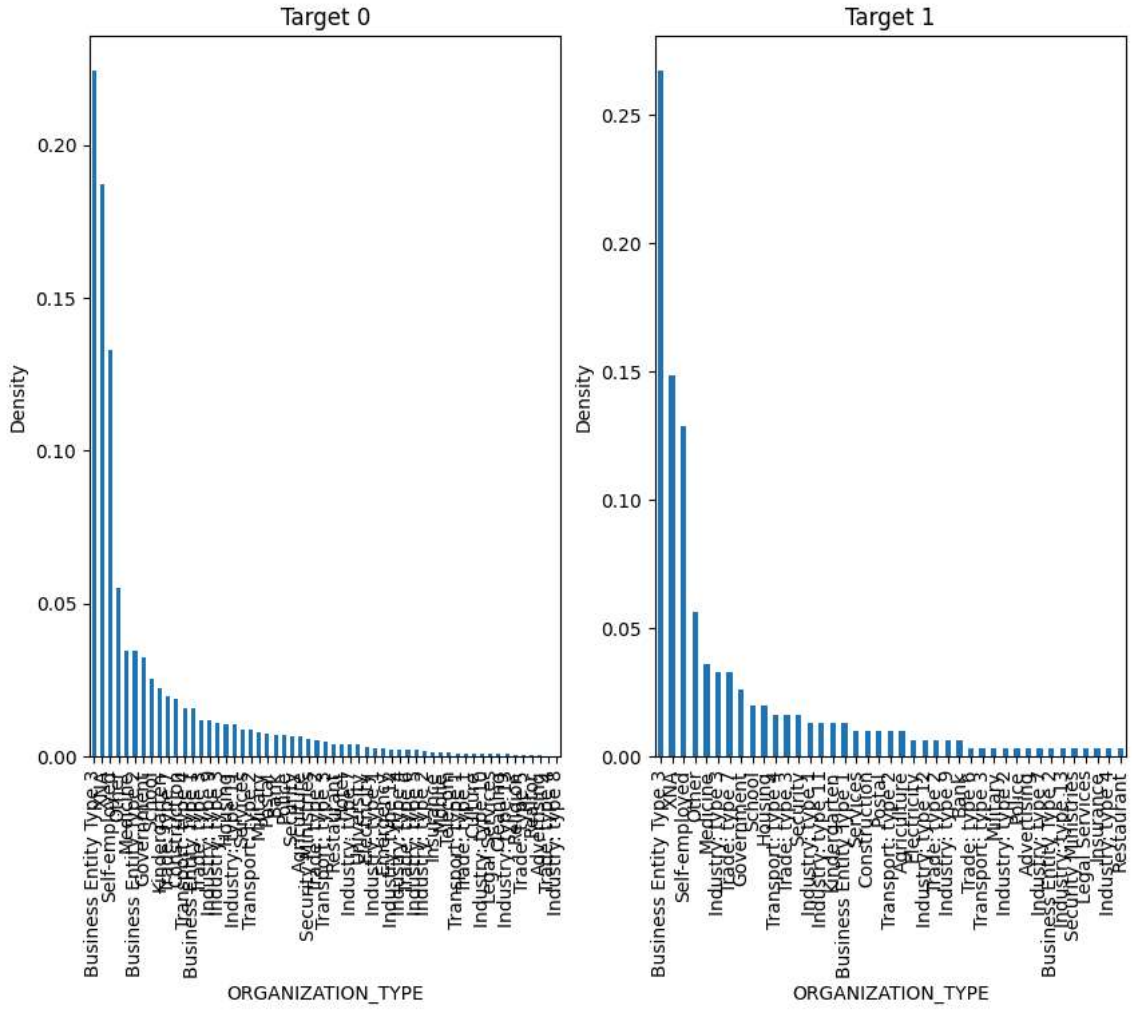


Plot on WEEKDAY_APPR_PROCESS_START for target 0 and 1

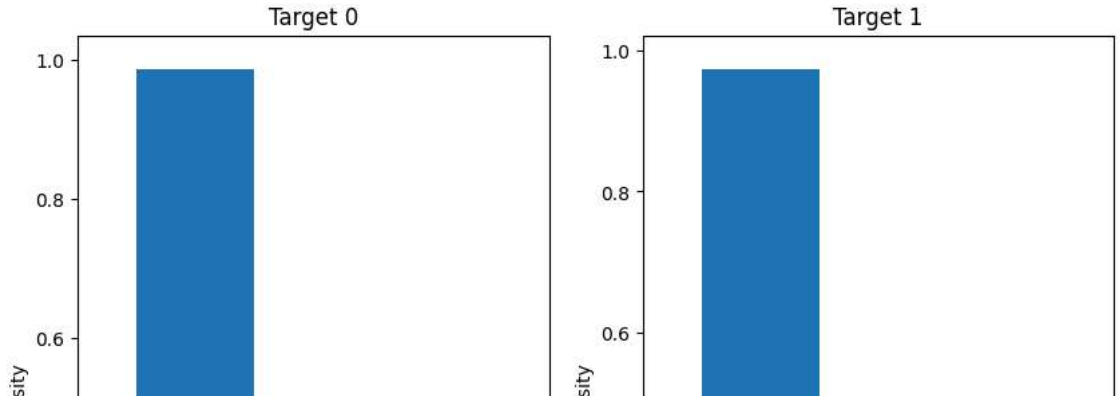


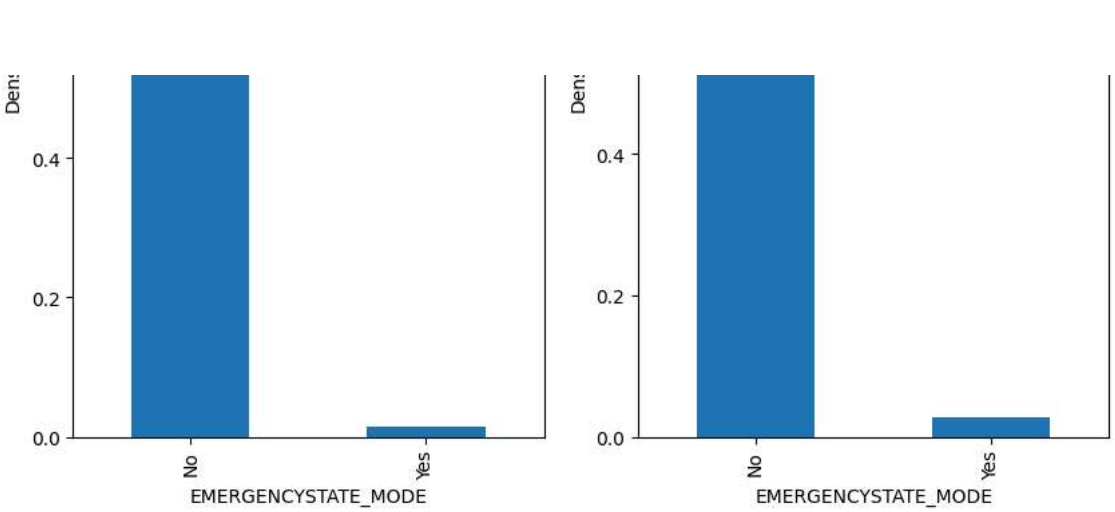


Plot on ORGANIZATION_TYPE for target 0 and 1



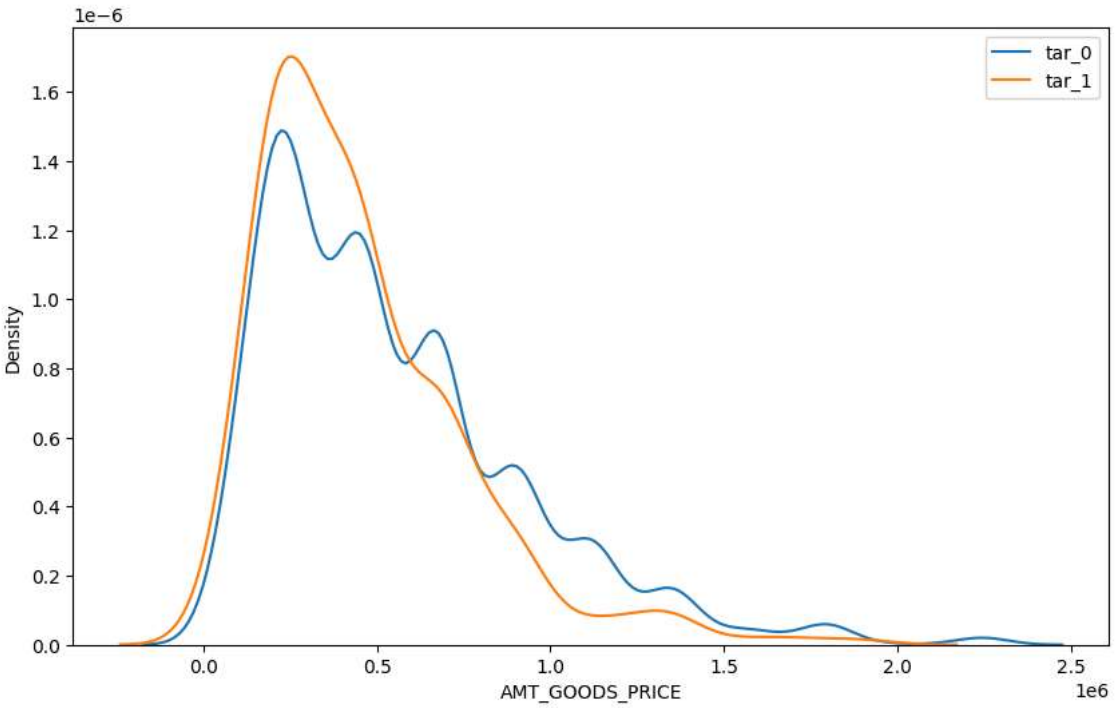
Plot on EMERGENCYSTATE_MODE for target 0 and 1





```
#conclusion-
#1.NAME_CONTRACT_TYPE -the applicants are receiving more cash loans than receiving loans both for target 0 and 1
#2.CODE_GENDER - number of female applicants are twice the male applicants for target 0 and 1
#3.FLAG_OWN_CAR-most (70%)of the applicants do not own car for target 0 and 1
#4.FLAG_OWN_REALTY-most (70%)of the applicants do not own a house for target 0 and 1
#5.NAME_TYPE_SUITE-most(81%) of the applicants are unaccomplished for target 0 and 1
#6.NAME_INCOME_TYPE-most (51%) of the applicants are working for target 0 and 1
#7.NAME_EDUCATION_TYPE-for both target 0 and 1,almost 71%of the applicants have completed secondary education
#8.NAME_FAMILY_STATUS-most (63%) of the applicants are married for target 0 and 1
#9.NAME_HOUSING_TYPE-most (88%) of the applicants own a house for target 0 and 1
#10.OCCUPATION_TYPE-most (31%) of the applicants have other occupation type for target 0 and 1
#11.WEEKDAY_APPR_PROCESS_START- MOST of the applicant have applied the loan on tuesday for target 0 and 1
#12.ORGANIZATION_TYPE-most of the oreganization type of employees are business entity type 3 for target 0 and 1
```

```
plt.figure(figsize=(10,6))
sns.distplot(tar_0['AMT_GOODS_PRICE'],label='tar_0',hist=False)
sns.distplot(tar_1['AMT_GOODS_PRICE'],label='tar_1',hist=False)
plt.legend()
plt.show()
```



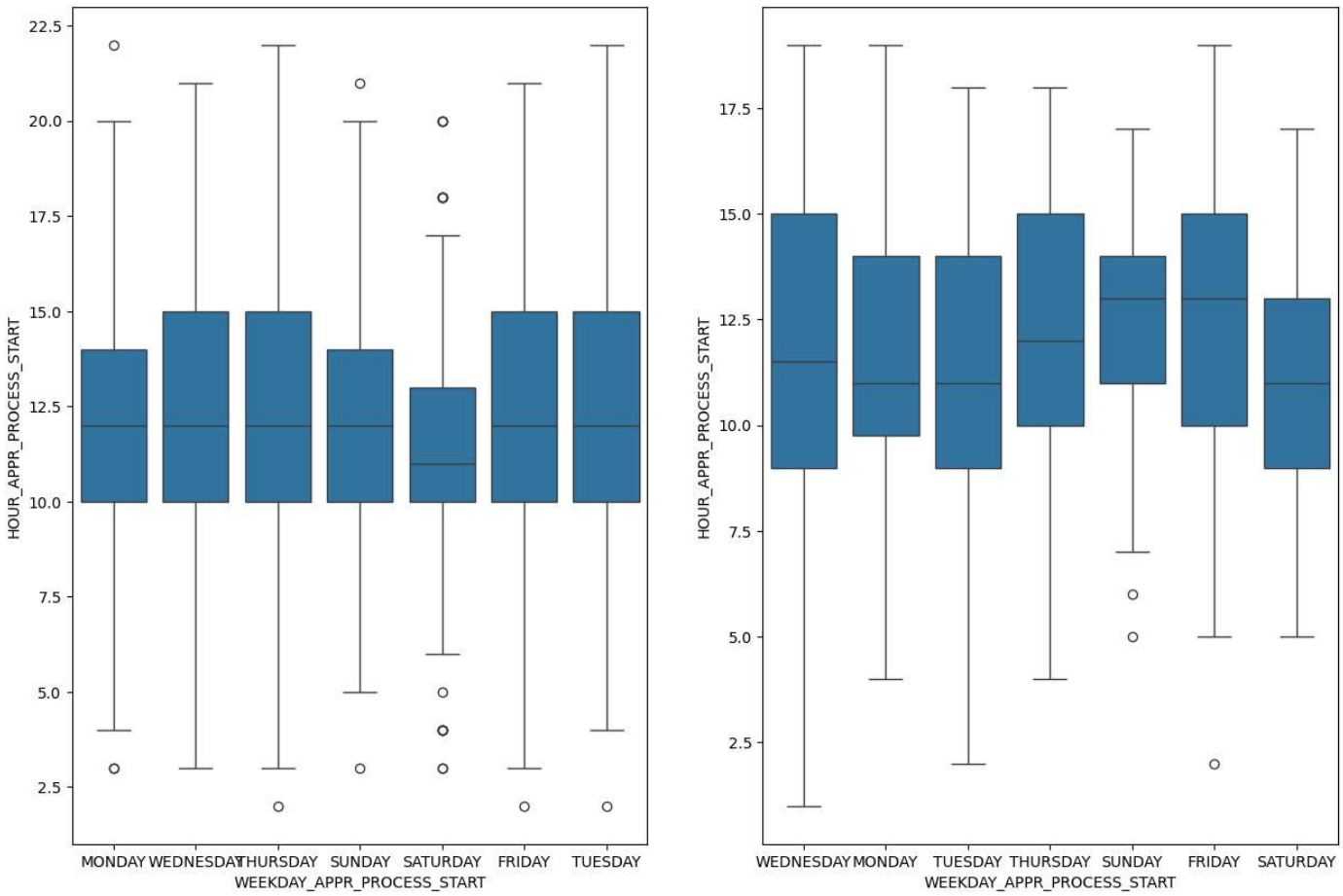
```
#conclusion  the price of the goods for which loan is given has the same variation for target 0 and 1
```

```
#BIVARIATE AND MULTIVARIATE ANALYSIS

#bivariate analysis between WEEKDAY_APPR_PROCESS_START vs HOUR_APPR_PROCESS_START


plt.figure(figsize=(15,10))
plt.subplot(1,2,1)
sns.boxplot(x='WEEKDAY_APPR_PROCESS_START',y='HOUR_APPR_PROCESS_START',data=tar_0)
plt.subplot(1,2,2)
sns.boxplot(x='WEEKDAY_APPR_PROCESS_START',y='HOUR_APPR_PROCESS_START',data=tar_1)
plt.show()

```




```
# conclusion>>
#1.the bank operates between 10am to 3pm except for saturdaty ,its between 10am to 2pm
#2.we can observe that around 11:30am to 12pm around 50% of customers visit the branch for loans application on all the days except for
#3.the loan defaulters have applied for the loan between 9:30am-10am and 2pm where as the applicants who repay the loan on time have app
```

```
# bivariate analysis between AGE_CATEGORY vs AMT_CREDIT

plt.figure(figsize=(15,10))
plt.subplot(1,2,1)
sns.boxplot(x='AGE_CATEGORY',y='AMT_CREDIT',data=tar_0)
plt.subplot(1,2,2)
sns.boxplot(x='AGE_CATEGORY',y='AMT_CREDIT',data=tar_1)
plt.show()

```