bank-loan-risk-assessment

May 13, 2024

0.1 Introduction

In this case study, we'll dive into the practical application of Exploratory Data Analysis (EDA) within the banking and financial services sector. my main focus will be on understanding how data analysis helps in managing risk when lending money to customers or simply reducing potential financial losses..

0.2 Business Understanding:

As per my Experience with one of India's premier financial institutions ICICI, I've observed that, one of the key challenges to the NBFCs OR BFSI is assessing the risk of lending to customers with insufficient or no credit history while taking care of thier business. Some customers would take advantage of this situation and become defaulters, leading to financial losses for the bank.

So, it is crucial to analyze the patterns present in the data to ensure that loan applicants capable of repaying are not rejected while minimizing the risk of approving loans to potential defaulters.

When a customer applies for a loan, the company must decide whether to approve or reject the application based on the applicant's profile. Two risks are associated with this decision:

1. If the applicant is likely to repay the loan, but the loan is not approved, it results in a loss of business for the company.
2. If the applicant is unlikely to repay the loan (i.e., they are likely to default), approving the loan may lead to a financial loss for the company.

The available data contains information about loan applications, including two types of scenarios:

- Clients with payment difficulties: They had late payments for more than a specified number of days on at least one of the initial loan installments.
- All other cases: Payments were made on time.

The company can take one of four decisions regarding a loan application:

- Approved: The loan application is approved.
- Cancelled: The client cancelled the application during the approval process, either due to a change of mind or unfavorable pricing due to higher risk.
- Refused: The company rejected the loan application (e.g., the client did not meet the requirements).
- Unused offer: The loan was cancelled by the client at different stages of the process.

In this case study, we will use Exploratory Data Analysis (EDA) to understand how consumer attributes and loan attributes influence the tendency of default and risk when lending money to customers.g money to customers.ers.le lending at ICICI Bank.tendency of default.

0.3 Business Objectives

This case study aims to identify patterns which indicate if a client has difficulty paying their installments which may be used for taking actions such as denying the loan, reducing the amount of loan, lending (to risky applicants) at a higher interest rate, etc. This will ensure that the consumers capable of repaying the loan are not rejected. Identification of such applicants using EDA is the aim of this case study.

In other words, the company wants to understand the driving factors (or driver variables) behind loan default, i.e. the variables which are strong indicators of default. The company can utilise this knowledge for its portfolio and risk assessment.

The data utilized in this analysis has been sourced from Kaggle.

The file application_data.csvontains s comprehensive client information at the time of application, particularly focusing on discerning if a client encounters payment difficulties.

Additionally, the dataset previous_application.csv provides insights into the client's historical loan data, detailing whether past applications were approved, cancelled, refused, or resulted in an unused offer.er

0.4 1.Importing Libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
plt.style.use('ggplot')
import plotly.express as px
#surpress warning
import warnings
warnings.filterwarnings('ignore')
```

0.5 2. Reading and Inspection

[2]:
```
df = pd.read_csv(r'D:
↪\PortfolioProject\PythonPortfolioProjects\Bank\application_data.csv')
df.sample(5)
```

[2]:
```
        SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR
FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY
AMT_GOODS_PRICE NAME_TYPE_SUITE      NAME_INCOME_TYPE
NAME_EDUCATION_TYPE NAME_FAMILY_STATUS  NAME_HOUSING_TYPE
REGION_POPULATION_RELATIVE  DAYS_BIRTH  DAYS_EMPLOYED  DAYS_REGISTRATION
DAYS_ID_PUBLISH  OWN_CAR_AGE  FLAG_MOBIL  FLAG_EMP_PHONE  FLAG_WORK_PHONE
FLAG_CONT_MOBILE  FLAG_PHONE  FLAG_EMAIL OCCUPATION_TYPE  CNT_FAM_MEMBERS
```

```
REGION_RATING_CLIENT  REGION_RATING_CLIENT_W_CITY WEEKDAY_APPR_PROCESS_START
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       ORGANIZATION_TYPE  EXT_SOURCE_1  EXT_SOURCE_2
EXT_SOURCE_3  APARTMENTS_AVG  BASEMENTAREA_AVG  YEARS_BEGINEXPLUATATION_AVG
YEARS_BUILD_AVG  COMMONAREA_AVG  ELEVATORS_AVG  ENTRANCES_AVG  FLOORSMAX_AVG
FLOORSMIN_AVG  LANDAREA_AVG  \
32491       137660       1         Cash loans           M            Y
N             1          135000.0    495000.0      39816.0         495000.0
Unaccompanied  Commercial associate  Secondary / secondary special
Married  House / apartment                    0.007305      -10646
-807            -2917.0            -3121         17.0           1
1                1                 1           1            0        Laborers
3.0                     3                            3
TUESDAY                       14                          0
0                            0                            0                      1
1  Business Entity Type 3      0.102731      0.399314      0.363945
NaN              NaN                          NaN              NaN
NaN             NaN            NaN             NaN             NaN            NaN
23865       127765       0         Cash loans           F            Y
Y             2          180000.0    675000.0      50598.0         675000.0
Unaccompanied              Working              Higher education     Civil
marriage  House / apartment                    0.022625      -11329
-4097            -2866.0             -177         10.0           1
1                1                 1           1            1      Core staff
4.0                     2                            2
FRIDAY                       19                          0
0                            0                            0                      0
0            Kindergarten      0.302644      0.517505           NaN
0.3299            0.2038                      0.9836              NaN
NaN            0.40         0.3448         0.3333             NaN            NaN
239209      377050       0         Cash loans           F            N
N             0          180000.0    509400.0      40243.5         450000.0
Unaccompanied         State servant  Secondary / secondary special
Married  House / apartment                    0.035792      -12887
-3152             -331.0            -4142          NaN           1
1                1                 1           0            0  Medicine staff
2.0                     2                            2
WEDNESDAY                       10                          0
0                            0                            0                      0
0                  Other      0.274025      0.554157      0.459690
0.1649            0.0520                      0.9990           0.9864
0.0658            0.16         0.1379         0.3750         0.4167           1.0
10339       112034       0         Cash loans           F            N
Y             1          135000.0    325908.0      12415.5         247500.0
Unaccompanied              Working              Higher education
Married  House / apartment                    0.020246      -14509
```

```
-6370            -8653.0            -5301          NaN           1
1                0                  1            1            1            NaN
3.0                   3                              3
SATURDAY                      7                           0
0                           0                       0                       0
0  Business Entity Type 2      0.477193      0.363472      0.436506
0.1407           0.1101                      0.9846          0.7892
0.0000           0.16         0.1379         0.3333         0.3750          0.0
211968      345638       0         Cash loans           F            N
Y             0           90000.0     76410.0       9198.0          67500.0
Unaccompanied  Commercial associate  Secondary / secondary special
Married  House / apartment                   0.018850      -10444
-984            -2186.0            -1948          NaN          1
1                0                  1            0            0     Accountants
2.0                   2                              2
THURSDAY                     15                           0
0                           0                       0                       0
0  Business Entity Type 3           NaN      0.475568      0.486653
0.1082              NaN                      0.9767              NaN
NaN            NaN          0.0345         0.1667            NaN           NaN

        LIVINGAPARTMENTS_AVG  LIVINGAREA_AVG  NONLIVINGAPARTMENTS_AVG
NONLIVINGAREA_AVG  APARTMENTS_MODE  BASEMENTAREA_MODE
YEARS_BEGINEXPLUATATION_MODE  YEARS_BUILD_MODE  COMMONAREA_MODE  ELEVATORS_MODE
ENTRANCES_MODE  FLOORSMAX_MODE  FLOORSMIN_MODE  LANDAREA_MODE
LIVINGAPARTMENTS_MODE  LIVINGAREA_MODE  NONLIVINGAPARTMENTS_MODE
NONLIVINGAREA_MODE  APARTMENTS_MEDI  BASEMENTAREA_MEDI
YEARS_BEGINEXPLUATATION_MEDI  YEARS_BUILD_MEDI  COMMONAREA_MEDI  ELEVATORS_MEDI
ENTRANCES_MEDI  FLOORSMAX_MEDI  FLOORSMIN_MEDI  LANDAREA_MEDI
LIVINGAPARTMENTS_MEDI  LIVINGAREA_MEDI  NONLIVINGAPARTMENTS_MEDI
NONLIVINGAREA_MEDI FONDKAPREMONT_MODE  HOUSETYPE_MODE  TOTALAREA_MODE
WALLSMATERIAL_MODE EMERGENCYSTATE_MODE  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  \
32491                   NaN             NaN                      NaN
NaN              NaN                NaN                          NaN
NaN              NaN             NaN              NaN             NaN
NaN            NaN                   NaN              NaN
NaN                  NaN              NaN                NaN
NaN               NaN              NaN             NaN              NaN
NaN             NaN           NaN                    NaN              NaN
NaN                  NaN                NaN             NaN              NaN
NaN                  NaN                      6.0                      1.0
6.0                    1.0                    -821.0                0
0                0                0                0                0
```

1 0 0
23865 NaN 0.2231 NaN
0.5058 0.3361 0.2115 0.9836
NaN NaN 0.4028 0.3448 0.3333
NaN NaN NaN 0.2325
NaN 0.5355 0.3331 0.2038
0.9836 NaN NaN 0.40 0.3448
0.3333 NaN NaN NaN 0.2271
NaN 0.5164 NaN block of flats 0.2855
Panel No 1.0 0.0
1.0 0.0 -1877.0 0
1 0 0 0 0
0 0 0
239209 0.1210 0.1376 0.0618
0.0448 0.1681 0.0540 0.9990
0.9869 0.0664 0.1611 0.1379 0.3750
0.4167 1.0 0.1322 0.1434
0.0623 0.0474 0.1665 0.0520
0.9990 0.9866 0.0662 0.16 0.1379
0.3750 0.4167 1.0 0.1231 0.1401
0.0621 0.0457 reg oper account block of flats 0.1698
Stone, brick No 1.0
0.0 1.0 0.0 -1840.0
0 1 0 0 0
0 0 0 0
10339 0.1143 0.1417 0.0019
0.0822 0.1355 0.1133 0.9836
0.7844 0.0000 0.1611 0.1379 0.3333
0.3750 0.0 0.1175 0.1387
0.0000 0.0000 0.1421 0.1101
0.9846 0.7920 0.0000 0.16 0.1379
0.3333 0.3750 0.0 0.1163 0.1443
0.0019 0.0840 reg oper account block of flats 0.1182
Panel No 2.0 0.0
2.0 0.0 -1880.0 0
1 0 0 0 0
0 0 0
211968 NaN 0.0543 NaN
0.0462 0.1103 NaN 0.9767
NaN NaN NaN 0.0345 0.1667
NaN NaN NaN 0.0566
NaN 0.0489 0.1093 NaN
0.9767 NaN NaN NaN 0.0345
0.1667 NaN NaN NaN 0.0553
NaN 0.0472 NaN block of flats 0.0528
Stone, brick No 2.0
0.0 2.0 0.0 -483.0

```
0                 1                 0                 0                 0
0                 0                 0                  0

        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
32491                  0                 0                 0                 0
0                 0                 0                 0                 0
0                 0                        0.0                        0.0
0.0                        0.0                        0.0
0.0
23865                  0                 0                 0                 0
0                 0                 0                 0                 0
0                 0                        NaN                        NaN
NaN                        NaN                        NaN
NaN
239209                 0                 0                 0                 0
0                 0                 0                 0                 0
0                 0                        0.0                        0.0
1.0                        0.0                        0.0
6.0
10339                  0                 0                 0                 0
0                 0                 0                 0                 0
0                 0                        0.0                        1.0
0.0                        0.0                        0.0
4.0
211968                 0                 0                 0                 0
0                 0                 0                 0                 0
0                 0                        0.0                        0.0
0.0                        0.0                        0.0
1.0
```

[3]:
```
df.shape
```

[3]:
```
(307511, 122)
```

[4]:
```
df.describe().T[0:10]
```

[4]:
```
                             count          mean            std           min
25%            50%            75%            max
SK_ID_CURR                 307511.0   278180.518577  102790.175348  100002.00000
189145.500000  278202.00000   367142.500000  4.562550e+05
TARGET                     307511.0        0.080729       0.272419       0.00000
0.000000        0.00000        0.000000  1.000000e+00
CNT_CHILDREN               307511.0        0.417052       0.722121       0.00000
```

```
0.000000        0.00000        1.000000  1.900000e+01
AMT_INCOME_TOTAL            307511.0  168797.919297  237123.146279   25650.00000
112500.000000  147150.00000  202500.000000  1.170000e+08
AMT_CREDIT                  307511.0  599025.999706  402490.776996   45000.00000
270000.000000  513531.00000  808650.000000  4.050000e+06
AMT_ANNUITY                 307499.0   27108.573909   14493.737315    1615.50000
16524.000000   24903.00000   34596.000000   2.580255e+05
AMT_GOODS_PRICE             307233.0  538396.207429  369446.460540   40500.00000
238500.000000  450000.00000  679500.000000  4.050000e+06
REGION_POPULATION_RELATIVE  307511.0       0.020868       0.013831       0.00029
0.010006        0.01885        0.028663  7.250800e-02
DAYS_BIRTH                  307511.0  -16036.995067    4363.988632  -25229.00000
-19682.000000  -15750.00000  -12413.000000 -7.489000e+03
DAYS_EMPLOYED               307511.0   63815.045904  141275.766519  -17912.00000
-2760.000000   -1213.00000    -289.000000   3.652430e+05
```

[5]:
```
df.dtypes
```

[5]:
```
SK_ID_CURR                      int64
TARGET                          int64
NAME_CONTRACT_TYPE             object
CODE_GENDER                    object
FLAG_OWN_CAR                   object
FLAG_OWN_REALTY                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
OWN_CAR_AGE                   float64
FLAG_MOBIL                      int64
FLAG_EMP_PHONE                  int64
FLAG_WORK_PHONE                 int64
FLAG_CONT_MOBILE                int64
FLAG_PHONE                      int64
FLAG_EMAIL                      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
EXT_SOURCE_1                    float64
EXT_SOURCE_2                    float64
EXT_SOURCE_3                    float64
APARTMENTS_AVG                  float64
BASEMENTAREA_AVG                float64
YEARS_BEGINEXPLUATATION_AVG     float64
YEARS_BUILD_AVG                 float64
COMMONAREA_AVG                  float64
ELEVATORS_AVG                   float64
ENTRANCES_AVG                   float64
FLOORSMAX_AVG                   float64
FLOORSMIN_AVG                   float64
LANDAREA_AVG                    float64
LIVINGAPARTMENTS_AVG            float64
LIVINGAREA_AVG                  float64
NONLIVINGAPARTMENTS_AVG         float64
NONLIVINGAREA_AVG               float64
APARTMENTS_MODE                 float64
BASEMENTAREA_MODE               float64
YEARS_BEGINEXPLUATATION_MODE    float64
YEARS_BUILD_MODE                float64
COMMONAREA_MODE                 float64
ELEVATORS_MODE                  float64
ENTRANCES_MODE                  float64
FLOORSMAX_MODE                  float64
FLOORSMIN_MODE                  float64
LANDAREA_MODE                   float64
LIVINGAPARTMENTS_MODE           float64
LIVINGAREA_MODE                 float64
NONLIVINGAPARTMENTS_MODE        float64
NONLIVINGAREA_MODE              float64
APARTMENTS_MEDI                 float64
BASEMENTAREA_MEDI               float64
YEARS_BEGINEXPLUATATION_MEDI    float64
YEARS_BUILD_MEDI                float64
```

```
COMMONAREA_MEDI                  float64
ELEVATORS_MEDI                   float64
ENTRANCES_MEDI                   float64
FLOORSMAX_MEDI                   float64
FLOORSMIN_MEDI                   float64
LANDAREA_MEDI                    float64
LIVINGAPARTMENTS_MEDI            float64
LIVINGAREA_MEDI                  float64
NONLIVINGAPARTMENTS_MEDI         float64
NONLIVINGAREA_MEDI               float64
FONDKAPREMONT_MODE                object
HOUSETYPE_MODE                    object
TOTALAREA_MODE                   float64
WALLSMATERIAL_MODE                object
EMERGENCYSTATE_MODE               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
FLAG_DOCUMENT_2                    int64
FLAG_DOCUMENT_3                    int64
FLAG_DOCUMENT_4                    int64
FLAG_DOCUMENT_5                    int64
FLAG_DOCUMENT_6                    int64
FLAG_DOCUMENT_7                    int64
FLAG_DOCUMENT_8                    int64
FLAG_DOCUMENT_9                    int64
FLAG_DOCUMENT_10                   int64
FLAG_DOCUMENT_11                   int64
FLAG_DOCUMENT_12                   int64
FLAG_DOCUMENT_13                   int64
FLAG_DOCUMENT_14                   int64
FLAG_DOCUMENT_15                   int64
FLAG_DOCUMENT_16                   int64
FLAG_DOCUMENT_17                   int64
FLAG_DOCUMENT_18                   int64
FLAG_DOCUMENT_19                   int64
FLAG_DOCUMENT_20                   int64
FLAG_DOCUMENT_21                   int64
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
dtype: object
```

[6]:
```
df.select_dtypes(include='object').columns
```

[6]:
```
Index(['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', 'FONDKAPREMONT_MODE',
'HOUSETYPE_MODE', 'WALLSMATERIAL_MODE', 'EMERGENCYSTATE_MODE'], dtype='object')
```

[7]:
```
df.select_dtypes(include='number').columns
```

[7]:
```
Index(['SK_ID_CURR', 'TARGET', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT',
'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'REGION_POPULATION_RELATIVE', 'DAYS_BIRTH',
'DAYS_EMPLOYED',
       …
       '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=106)
```

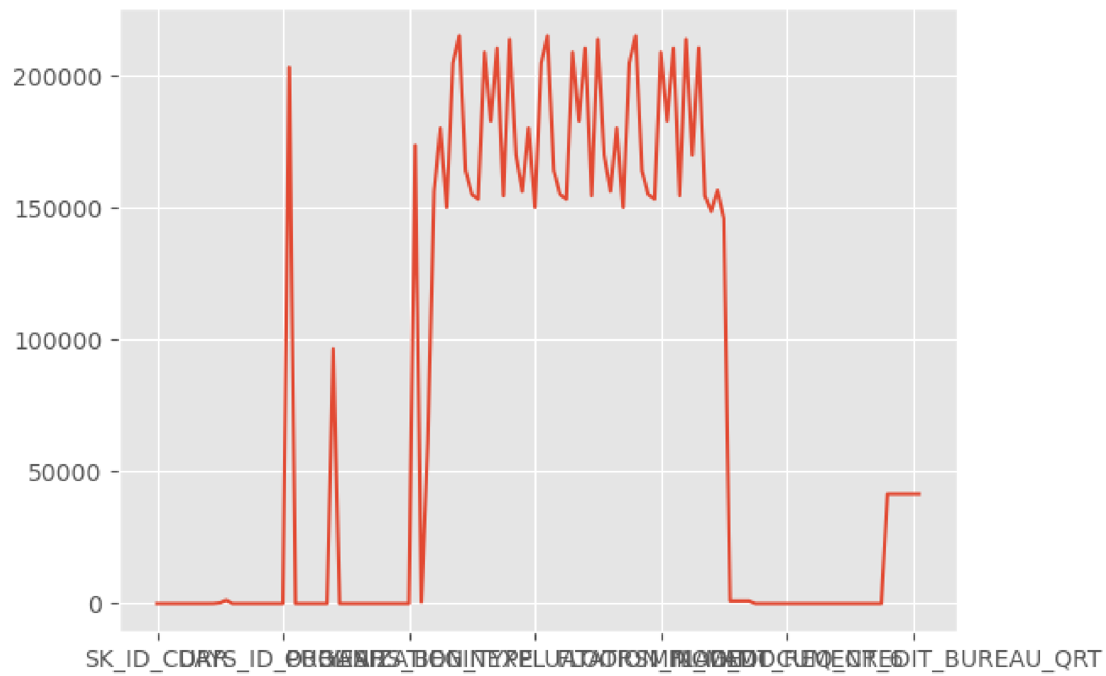
0.6 3. Checking null values

[8]:
```
df.shape
```

[8]:
```
(307511, 122)
```

[9]:
```
df.isnull().sum().plot(kind='line')
```

[9]:
```
<Axes: >
```



Insight: Based on the above graph, it is evidednt that the dataset has many missing values. Let's check for each column what is the % of missing values

[10]:
```
round((df.isnull().sum()/len(df))*100, 2).sort_values(ascending=False)
```

[10]:
```
COMMONAREA_MEDI              69.87
COMMONAREA_AVG               69.87
COMMONAREA_MODE              69.87
NONLIVINGAPARTMENTS_MODE     69.43
NONLIVINGAPARTMENTS_AVG      69.43
NONLIVINGAPARTMENTS_MEDI     69.43
FONDKAPREMONT_MODE           68.39
LIVINGAPARTMENTS_MODE        68.35
LIVINGAPARTMENTS_AVG         68.35
LIVINGAPARTMENTS_MEDI        68.35
FLOORSMIN_AVG                67.85
FLOORSMIN_MODE               67.85
FLOORSMIN_MEDI               67.85
YEARS_BUILD_MEDI             66.50
YEARS_BUILD_MODE             66.50
YEARS_BUILD_AVG              66.50
OWN_CAR_AGE                  65.99
LANDAREA_MEDI                59.38
LANDAREA_MODE                59.38
LANDAREA_AVG                 59.38
```

BASEMENTAREA_MEDI	58.52
BASEMENTAREA_AVG	58.52
BASEMENTAREA_MODE	58.52
EXT_SOURCE_1	56.38
NONLIVINGAREA_MODE	55.18
NONLIVINGAREA_AVG	55.18
NONLIVINGAREA_MEDI	55.18
ELEVATORS_MEDI	53.30
ELEVATORS_AVG	53.30
ELEVATORS_MODE	53.30
WALLSMATERIAL_MODE	50.84
APARTMENTS_MEDI	50.75
APARTMENTS_AVG	50.75
APARTMENTS_MODE	50.75
ENTRANCES_MEDI	50.35
ENTRANCES_AVG	50.35
ENTRANCES_MODE	50.35
LIVINGAREA_AVG	50.19
LIVINGAREA_MODE	50.19
LIVINGAREA_MEDI	50.19
HOUSETYPE_MODE	50.18
FLOORSMAX_MODE	49.76
FLOORSMAX_MEDI	49.76
FLOORSMAX_AVG	49.76
YEARS_BEGINEXPLUATATION_MODE	48.78
YEARS_BEGINEXPLUATATION_MEDI	48.78
YEARS_BEGINEXPLUATATION_AVG	48.78
TOTALAREA_MODE	48.27
EMERGENCYSTATE_MODE	47.40
OCCUPATION_TYPE	31.35
EXT_SOURCE_3	19.83
AMT_REQ_CREDIT_BUREAU_HOUR	13.50
AMT_REQ_CREDIT_BUREAU_DAY	13.50
AMT_REQ_CREDIT_BUREAU_WEEK	13.50
AMT_REQ_CREDIT_BUREAU_MON	13.50
AMT_REQ_CREDIT_BUREAU_QRT	13.50
AMT_REQ_CREDIT_BUREAU_YEAR	13.50
NAME_TYPE_SUITE	0.42
DEF_30_CNT_SOCIAL_CIRCLE	0.33
OBS_30_CNT_SOCIAL_CIRCLE	0.33
OBS_60_CNT_SOCIAL_CIRCLE	0.33
DEF_60_CNT_SOCIAL_CIRCLE	0.33
EXT_SOURCE_2	0.21
AMT_GOODS_PRICE	0.09
CNT_CHILDREN	0.00
FLAG_DOCUMENT_8	0.00
NAME_CONTRACT_TYPE	0.00

CODE_GENDER	0.00
FLAG_OWN_CAR	0.00
DAYS_LAST_PHONE_CHANGE	0.00
FLAG_DOCUMENT_2	0.00
FLAG_DOCUMENT_3	0.00
FLAG_DOCUMENT_4	0.00
FLAG_DOCUMENT_5	0.00
FLAG_DOCUMENT_6	0.00
FLAG_DOCUMENT_7	0.00
FLAG_DOCUMENT_9	0.00
FLAG_DOCUMENT_21	0.00
FLAG_DOCUMENT_10	0.00
FLAG_DOCUMENT_11	0.00
FLAG_OWN_REALTY	0.00
FLAG_DOCUMENT_13	0.00
FLAG_DOCUMENT_14	0.00
FLAG_DOCUMENT_15	0.00
FLAG_DOCUMENT_16	0.00
FLAG_DOCUMENT_17	0.00
FLAG_DOCUMENT_18	0.00
FLAG_DOCUMENT_19	0.00
FLAG_DOCUMENT_20	0.00
FLAG_DOCUMENT_12	0.00
AMT_CREDIT	0.00
AMT_INCOME_TOTAL	0.00
FLAG_PHONE	0.00
LIVE_CITY_NOT_WORK_CITY	0.00
REG_CITY_NOT_WORK_CITY	0.00
TARGET	0.00
REG_CITY_NOT_LIVE_CITY	0.00
LIVE_REGION_NOT_WORK_REGION	0.00
REG_REGION_NOT_WORK_REGION	0.00
REG_REGION_NOT_LIVE_REGION	0.00
HOUR_APPR_PROCESS_START	0.00
WEEKDAY_APPR_PROCESS_START	0.00
REGION_RATING_CLIENT_W_CITY	0.00
REGION_RATING_CLIENT	0.00
CNT_FAM_MEMBERS	0.00
FLAG_EMAIL	0.00
FLAG_CONT_MOBILE	0.00
ORGANIZATION_TYPE	0.00
FLAG_WORK_PHONE	0.00
FLAG_EMP_PHONE	0.00
FLAG_MOBIL	0.00
DAYS_ID_PUBLISH	0.00
DAYS_REGISTRATION	0.00
DAYS_EMPLOYED	0.00

```
DAYS_BIRTH                     0.00
REGION_POPULATION_RELATIVE     0.00
NAME_HOUSING_TYPE              0.00
NAME_FAMILY_STATUS             0.00
NAME_EDUCATION_TYPE            0.00
NAME_INCOME_TYPE               0.00
AMT_ANNUITY                    0.00
SK_ID_CURR                     0.00
dtype: float64
```

[11]:
```python
#Dropping off all the columns with more than 46% null values
df.drop(columns=df.columns[(df.isna().sum()/len(df)*100 > 46)],inplace=True)
```

There are still columns with a notably high null percentage, one may eliminate these columns based on their use or impute them with the appropriate value. I selected to eliminate columns based on my intuition or if they had a high rate of null values.

[12]:
```python
df.drop(columns=['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',␣
 ↪'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE',
    ␣
 ↪'FLAG_PHONE','CNT_FAM_MEMBERS','REGION_RATING_CLIENT_W_CITY','EXT_SOURCE_3'],␣
 ↪inplace=True)
```

[13]:
```python
np.array(df.columns[(df.isna().mean()>0)])
```

[13]:
```
array(['AMT_ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE',
       'OCCUPATION_TYPE', 'EXT_SOURCE_2', 'OBS_30_CNT_SOCIAL_CIRCLE',
       'DEF_30_CNT_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCLE',
       'DEF_60_CNT_SOCIAL_CIRCLE', 'DAYS_LAST_PHONE_CHANGE',
       'AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY',
       'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUREAU_MON',
       'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR'],
      dtype=object)
```

We will examine the above mentioned columns for anomalies and potential imputers.

[ ]:
```python

```

0.6.1 4. Checking columns for values to impute

'AMT_ANNUITY' Variable

[14]:
```python
#Null values in AMT_ANNUITY column
df['AMT_ANNUITY'].isna().sum()
```

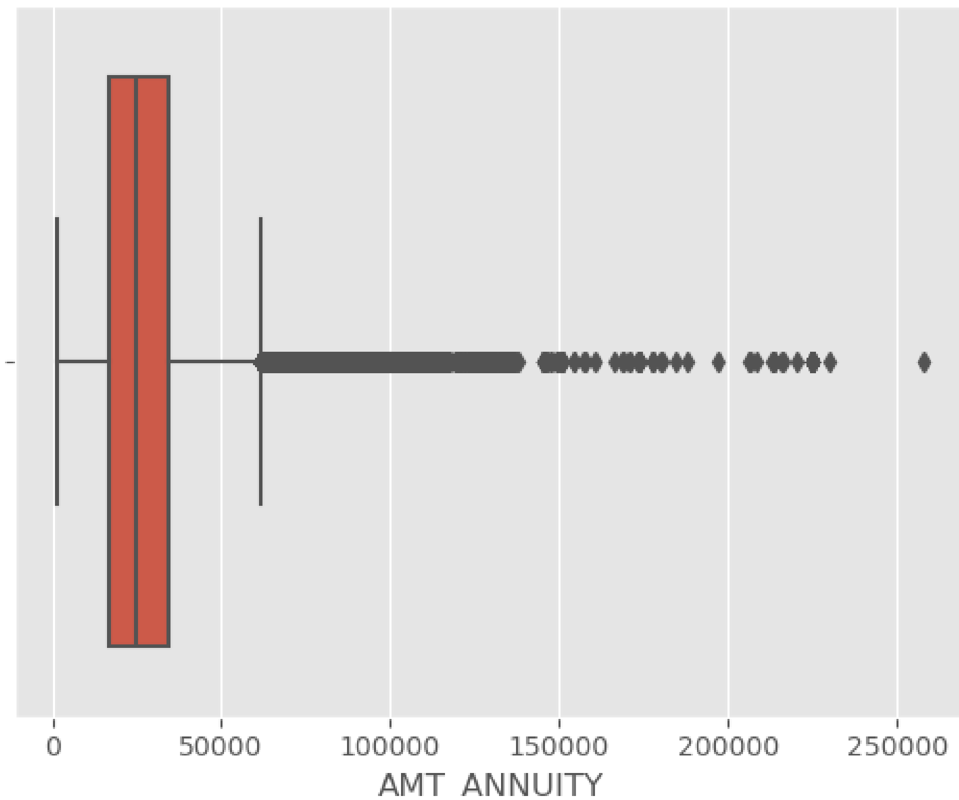
[14]: 12

[15]:
```python
df['AMT_ANNUITY'].isna().sum()/len(df)
```

[15]: 3.9022994299390916e-05

[16]:
```python
#Checking for any outliers using a box plot
sns.boxplot(data=df,x='AMT_ANNUITY')
```

[16]: <Axes: xlabel='AMT_ANNUITY'>



We can see the outliers are present in the data and the difference between max and min is significant so, impute null values with median value rather than replacing with mean.

[17]:
```python
df['AMT_ANNUITY'].fillna(value=df['AMT_ANNUITY'].median(),inplace=True)
```

[18]:
```python
df['AMT_ANNUITY'].isna().sum()
```

[18]: 0

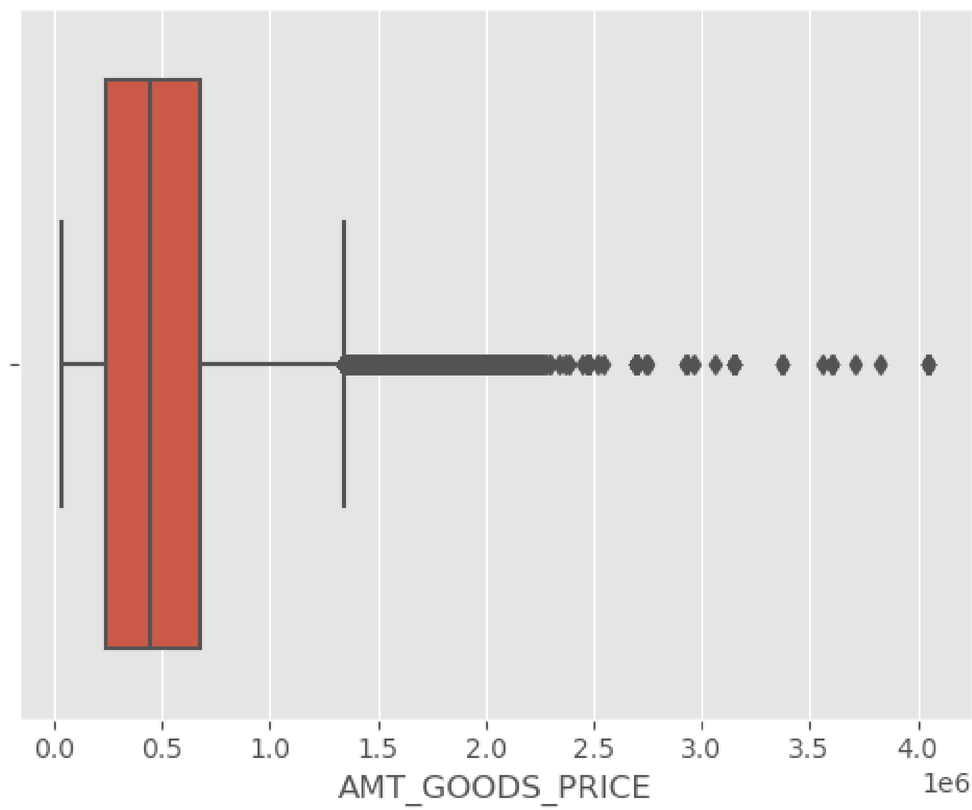
'AMT_GOODS_PRICE' Variable

[19]:
```
#Null values in AMT_GOODS_PRICE column
df['AMT_GOODS_PRICE'].isna().sum()/len(df)
```

[19]: 0.0009040327012692228

[20]:
```
#Checking for any outliers using a box plot
sns.boxplot(data=df,x='AMT_GOODS_PRICE')
```

[20]: <Axes: xlabel='AMT_GOODS_PRICE'>



[21]:
```
# Imputing null values with median
df['AMT_GOODS_PRICE'].fillna(value=df['AMT_GOODS_PRICE'].median(),inplace=True)
```

[22]:
```
df['AMT_GOODS_PRICE'].isna().sum()
```

[22]: 0

'NAME_TYPE_SUITE' Variable

[23]:
```
# Checking for the percentage of null values in NAME_TYPE_SUITE categorical␣
↪variable
df['NAME_TYPE_SUITE'].isna().sum()/len(df)*100
```

[23]: 0.42014757195677555

As a categorical variable, 'NAME TYPE SUITE' contains around 0.42% missing values. Therefore, we may impute the missing data with the most common group, "Unaccompanied."

[24]:
```
df['NAME_TYPE_SUITE'] = df['NAME_TYPE_SUITE'].fillna(df['NAME_TYPE_SUITE'].
↪value_counts().index[0])
```

[25]:
```
df['NAME_TYPE_SUITE'].isna().sum()
```

[25]: 0

'OCCUPATION_TYPE' Variable

[26]:
```
df['OCCUPATION_TYPE'].isna().sum()/len(df)*100
```

[26]: 31.345545362604916

[27]:
```
df['OCCUPATION_TYPE'].value_counts().index[0]
```

[27]: 'Laborers'

As a categorical variable, 'OCCUPATION_TYPE' contains around 31.3% missing values. Therefore, we may impute the missing data with the most common group, "Laborers."

[28]:
```
df['OCCUPATION_TYPE'] = df['OCCUPATION_TYPE'].fillna(df['OCCUPATION_TYPE'].
↪value_counts().index[0])
```

[29]:
```
df['OCCUPATION_TYPE'].isna().sum()
```

[29]: 0

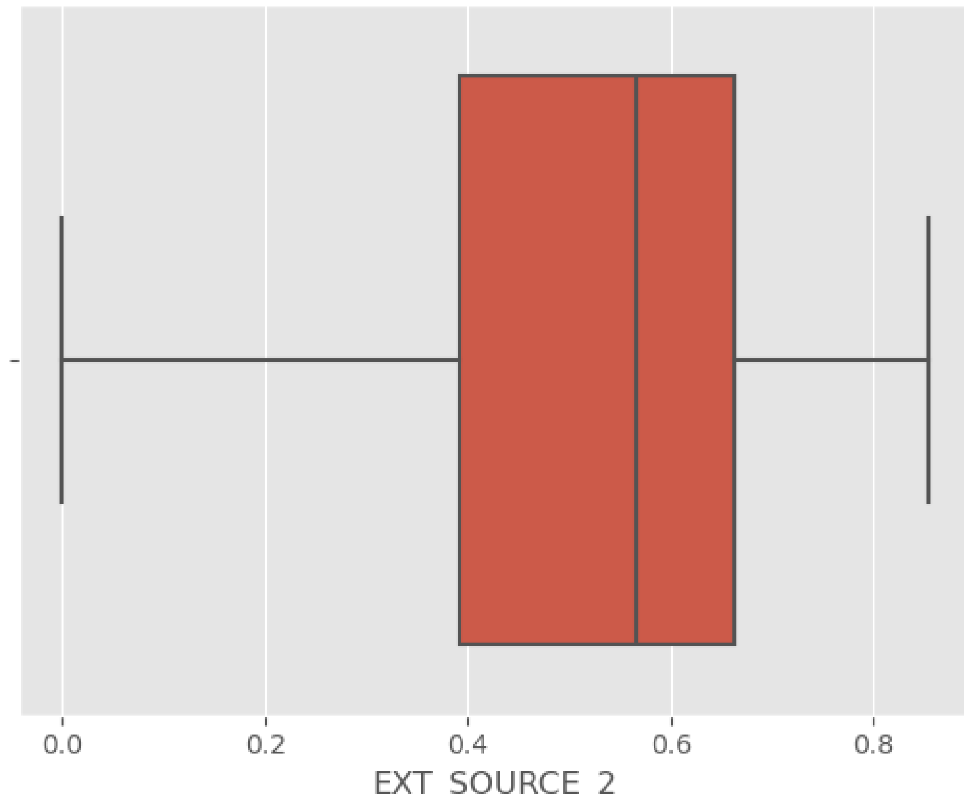
'EXT_SOURCE_2' Variable

[30]:
```
df['EXT_SOURCE_2'].isna().sum()/len(df)*100
```

[30]: 0.21462646864665005

[31]:
```
#Checking for any outliers using a box plot
sns.boxplot(data=df,x='EXT_SOURCE_2')
```

[31]: <Axes: xlabel='EXT_SOURCE_2'>



Because there are no outliers in the data, we may impute missing values using the mean.

[32]:
```
# Imputing null values with mode
df['EXT_SOURCE_2'].fillna(value=df['EXT_SOURCE_2'].mean(),inplace=True)
```

[33]:
```
df['EXT_SOURCE_2'].isna().sum()
```

[33]:
```
0
```

[34]:
```
df['CODE_GENDER'].value_counts()
```

[34]:
```
CODE_GENDER
F      202448
M      105059
XNA         4
Name: count, dtype: int64
```

As can be seen, Female(F) is the majority, and just four rows include XNA values. Thus, updating those columns with Gender = 'F' will not have a significant effect on the dataset.

[35]:
```
df.loc[df['CODE_GENDER']=='XNA','CODE_GENDER']='F'
```

[36]:
```
df['DAYS_LAST_PHONE_CHANGE'].fillna(df.DAYS_LAST_PHONE_CHANGE.mode()[0],inplace␣
↪= True)
```

[37]:
```
df['AMT_REQ_CREDIT_BUREAU_HOUR'] = df['AMT_REQ_CREDIT_BUREAU_HOUR'].
↪fillna(df['AMT_REQ_CREDIT_BUREAU_HOUR'].value_counts().index[0])
```

[38]:
```
df['AMT_REQ_CREDIT_BUREAU_DAY'] = df['AMT_REQ_CREDIT_BUREAU_DAY'].
↪fillna(df['AMT_REQ_CREDIT_BUREAU_DAY'].value_counts().index[0])
```

[39]:
```
df['AMT_REQ_CREDIT_BUREAU_WEEK'] = df['AMT_REQ_CREDIT_BUREAU_WEEK'].
↪fillna(df['AMT_REQ_CREDIT_BUREAU_WEEK'].value_counts().index[0])
```

[40]:
```
df['AMT_REQ_CREDIT_BUREAU_MON'] = df['AMT_REQ_CREDIT_BUREAU_MON'].
↪fillna(df['AMT_REQ_CREDIT_BUREAU_MON'].value_counts().index[0])
```

[41]:
```
df['AMT_REQ_CREDIT_BUREAU_QRT'] = df['AMT_REQ_CREDIT_BUREAU_QRT'].
↪fillna(df['AMT_REQ_CREDIT_BUREAU_QRT'].value_counts().index[0])
```

[42]:
```
df['AMT_REQ_CREDIT_BUREAU_YEAR'] = df['AMT_REQ_CREDIT_BUREAU_YEAR'].
↪fillna(df['AMT_REQ_CREDIT_BUREAU_YEAR'].value_counts().index[0])
```

[43]:
```
(df.isnull().sum()/len(df)*100).sort_values(ascending=False).head()
```

[43]:
```
OBS_30_CNT_SOCIAL_CIRCLE    0.332021
DEF_60_CNT_SOCIAL_CIRCLE    0.332021
OBS_60_CNT_SOCIAL_CIRCLE    0.332021
DEF_30_CNT_SOCIAL_CIRCLE    0.332021
ORGANIZATION_TYPE           0.000000
dtype: float64
```

[44]:
```
df[df['ORGANIZATION_TYPE']=='XNA']['NAME_INCOME_TYPE'].head()
```

[44]:
```
8     Pensioner
11    Pensioner
23    Pensioner
38    Pensioner
43    Pensioner
Name: NAME_INCOME_TYPE, dtype: object
```

We can see, for almost all the instances where 'ORGANIZATION_TYPE' = 'XNA' they fall in Pensioner Income category

[45]:
```
df['ORGANIZATION_TYPE'] = df['ORGANIZATION_TYPE'].replace('XNA','Pensioner')
```

[46]:
```
num_values = pd.to_numeric(df['CNT_CHILDREN'], errors='coerce')
non_numeric = df['CNT_CHILDREN'][num_values.isna()]
```

```
print(non_numeric)
```

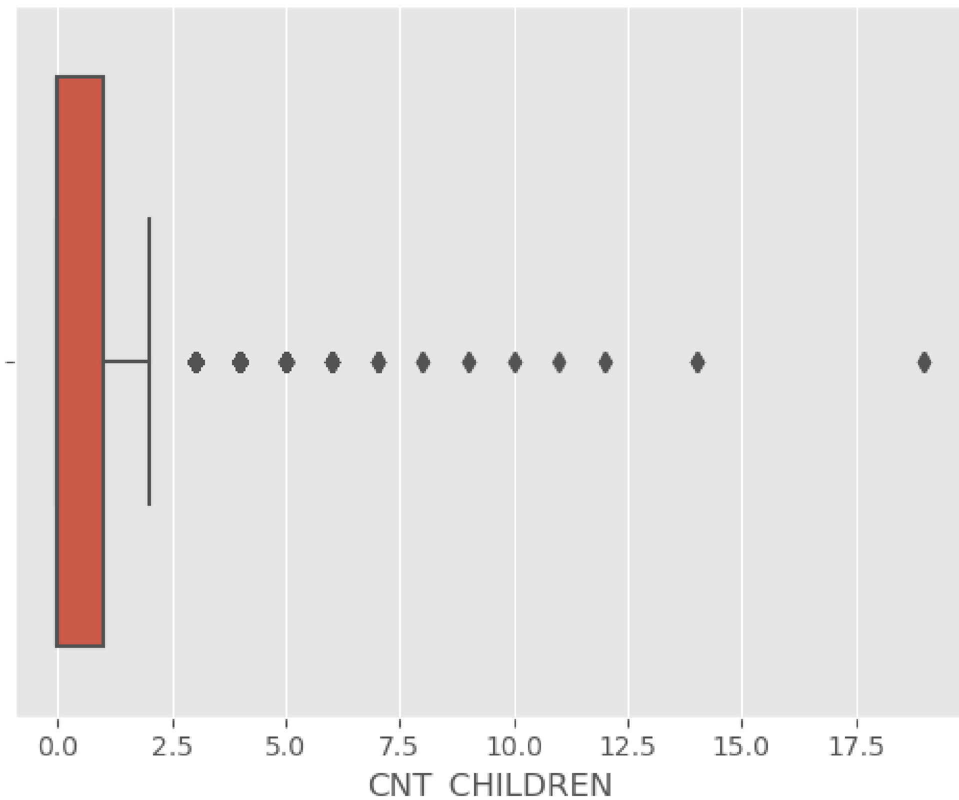
```
Series([], Name: CNT_CHILDREN, dtype: int64)
```

there are for datatypes that are non-numeric that are f let's remove it

[47]:
```
numeric_values = pd.to_numeric(df['CNT_CHILDREN'], errors='coerce')
df = df[~numeric_values.isna()]
```

[48]:
```
sns.boxplot(data=df,x='CNT_CHILDREN')
```

[48]: <Axes: xlabel='CNT_CHILDREN'>



0.6.2 5. Changing Data Types

we already see there are many columns that have float datatypes so All can be converted to integer data types.

[49]:
```
cols=df.select_dtypes(include='number').columns
cols
```

[49]:

```
Index(['SK_ID_CURR', 'TARGET', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT',
'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'REGION_POPULATION_RELATIVE', 'DAYS_BIRTH',
'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH', 'FLAG_MOBIL',
'FLAG_EMAIL', 'REGION_RATING_CLIENT', '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', 'EXT_SOURCE_2',
'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF_30_CNT_SOCIAL_CIRCLE',
'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCLE',
'DAYS_LAST_PHONE_CHANGE', 'AMT_REQ_CREDIT_BUREAU_HOUR',
'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK',
'AMT_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT',
'AMT_REQ_CREDIT_BUREAU_YEAR'], dtype='object')
```

[50]:

```
df[cols]= df[cols].astype('int64',errors='ignore')
```

[51]:

```
cols=df.select_dtypes(include='object').columns
cols
```

[51]:

```
Index(['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'], dtype='object')
```

[52]:

```
df[cols]= df[cols].astype('str',errors='ignore')
```

[53]:

```
df['DAYS_BIRTH'] = abs(df['DAYS_BIRTH'])//365
df['DAYS_BIRTH']
```

[53]:

```
0         25
1         45
2         52
3         52
4         54
          ..
307506    25
307507    56
307508    41
307509    32
307510    46
Name: DAYS_BIRTH, Length: 307511, dtype: int64
```

[54]:

```
df['DAYS_EMPLOYED'] = round(abs(df['DAYS_EMPLOYED'])/365,2)
df['DAYS_EMPLOYED']
```

[54]:

```
0            1.75
1            3.25
```

```
2          0.62
3          8.33
4          8.32
           …
307506     0.65
307507  1000.67
307508    21.70
307509    13.11
307510     3.46
Name: DAYS_EMPLOYED, Length: 307511, dtype: float64
```

[55]:
```python
df['DAYS_REGISTRATION'] = round(abs(df['DAYS_REGISTRATION']/365),2)
df['DAYS_REGISTRATION']
```

[55]:
```
0          9.99
1          3.25
2         11.67
3         26.94
4         11.81
          …
307506    23.17
307507    12.02
307508    18.46
307509     7.02
307510    14.05
Name: DAYS_REGISTRATION, Length: 307511, dtype: float64
```

[56]:
```python
df['DAYS_ID_PUBLISH'] = round(abs(df['DAYS_ID_PUBLISH']/365),2)
df['DAYS_ID_PUBLISH']
```

[56]:
```
0          5.81
1          0.80
2          6.93
3          6.68
4          9.47
          …
307506     5.43
307507    11.21
307508    14.11
307509     2.55
307510     1.12
Name: DAYS_ID_PUBLISH, Length: 307511, dtype: float64
```

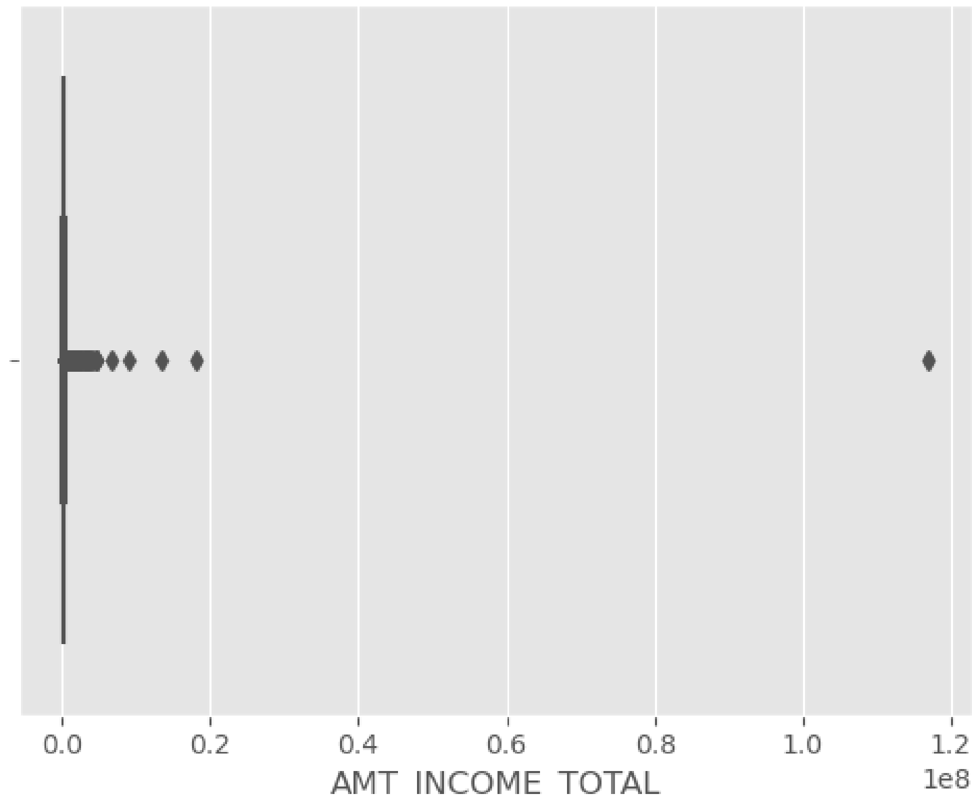
0.6.3 6. Dealing with outliers

[57]:
```
df.columns
```

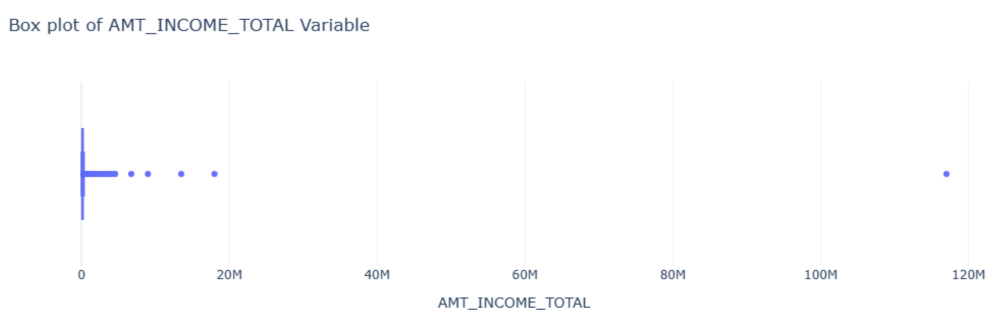
[57]:
```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL',
'AMT_CREDIT', 'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE',
'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS',
'NAME_HOUSING_TYPE', 'REGION_POPULATION_RELATIVE', 'DAYS_BIRTH',
'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH', 'FLAG_MOBIL',
'FLAG_EMAIL', 'OCCUPATION_TYPE', 'REGION_RATING_CLIENT',
'WEEKDAY_APPR_PROCESS_START', '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', 'ORGANIZATION_TYPE',
'EXT_SOURCE_2', 'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF_30_CNT_SOCIAL_CIRCLE',
'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCLE',
'DAYS_LAST_PHONE_CHANGE', 'AMT_REQ_CREDIT_BUREAU_HOUR',
'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK',
'AMT_REQ_CREDIT_BUREAU_MON',
       'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR'],
      dtype='object')
```

[58]:
```
sns.boxplot(df, x='AMT_INCOME_TOTAL')
```

[58]:
```
<Axes: xlabel='AMT_INCOME_TOTAL'>
```



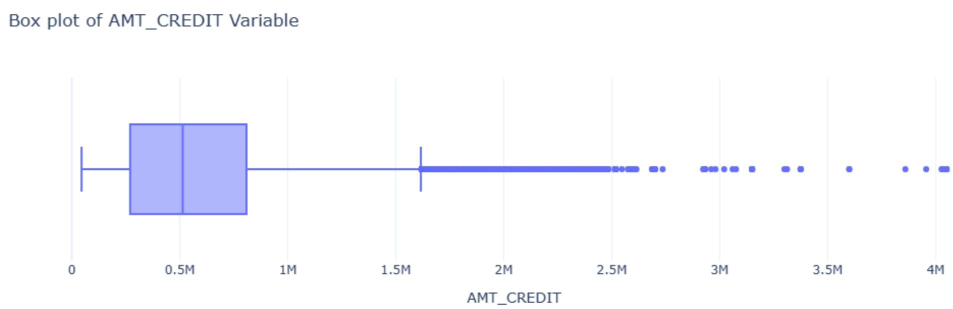
[59]:
```
px.box(df,x='AMT_INCOME_TOTAL',height=300,width=700,template='plotly_white',
       title="Box plot of AMT_INCOME_TOTAL Variable")
```



Observing that around 30% of the data are outliers indicates that the data is widely spread and requires additional investigation.

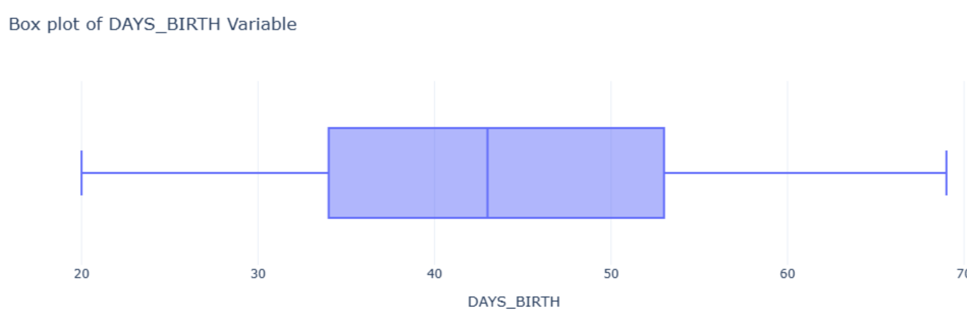
We can see that there is no statistically significant difference across quantiles, and as income is a continuous variable, it varies from person to person.

[127]:
```
px.box(df,x='AMT_CREDIT',height=300,width=700,template='plotly_white',
       title="Box plot of AMT_CREDIT Variable")
```



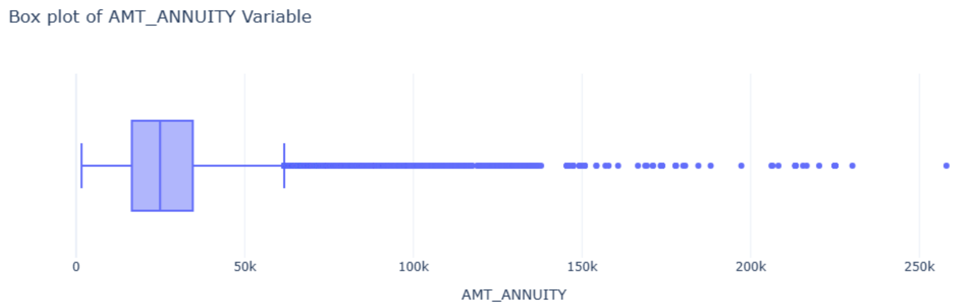
We can see that there is no statistically significant difference across quantiles, and as credit is a continuous variable, it varies from person to person.

[128]:
```
px.box(df,x='DAYS_BIRTH',height=300,width=700,template='plotly_white',
       title="Box plot of DAYS_BIRTH Variable")
```



According to the box plot, there are no outliers. There is no substantial gap between the mean and the median.

[129]:
```
px.box(df,x='AMT_ANNUITY',height=300,width=700,template='plotly_white',
       title="Box plot of AMT_ANNUITY Variable")
```



Observing that 30% of the data are outliers indicates that the data is widely spread and requires additional investigation.We can see that there Observing that 30% of the data are outliers indicates that the data is widely spread and requires additional investigation.We can see that there is no statistically significant difference across quantiles, and as income is a continuous variable, it varies from person to person. no statistically significant difference across quantiles, and as income is a continuous variable, it varies from person to person.

[130]:
```
px.box(df,x='AMT_GOODS_PRICE',height=300,width=700,template='plotly_white',
       title="Box plot of AMT_GOODS_PRICE Variable")
```

Observing the box plot, we can say that around 30% of the data are outliers indicates that the data is widely spread and requires additional investigation.We can see that there is no statisticalObserving the box plot, we can say that 30% of the data are outliers indicates that the data is widely spread and requires additional investigation.We can see that there is no statistically significant difference across quantiles, and as income is a continuous variable, it varies from person to person.ly significant difference across quantiles, and as income is a continuous variable, it varies from person to person.

[ ]:
```

```

0.6.4 7. Continous variables binning

[ ]:
```
#Creating a categorical variable based on income total
df['AMT_INCOME_TOTAL_CAT'] = pd.qcut(df['AMT_INCOME_TOTAL'],q=[0,0.2,0.5,0.8,0.
↪95,1],
       labels=['VeryLow','Low','Medium','High','VeryHigh'])
```

[131]:
```
df['AMT_INCOME_TOTAL_CAT'].head()
```

[131]:
```
0     Medium
1       High
2    VeryLow
3        Low
4        Low
Name: AMT_INCOME_TOTAL_CAT, dtype: category
Categories (5, object): ['VeryLow' < 'Low' < 'Medium' < 'High' < 'VeryHigh']
```

[ ]:
```
#Creating a categorical variable based on credit amount
df['AMT_CREDIT_CAT'] = pd.qcut(df['AMT_CREDIT'],q=[0,0.2,0.5,0.8,0.95,1],␣
↪labels=['VeryLow','Low','Medium','High','VeryHigh'])
```

[132]:
```
df['AMT_CREDIT_CAT'].head()
```

[132]:
```
0        Low
1       High
2    VeryLow
3        Low
4        Low
Name: AMT_CREDIT_CAT, dtype: category
Categories (5, object): ['VeryLow' < 'Low' < 'Medium' < 'High' < 'VeryHigh']
```

[ ]:
```
#Creating a categorical variable based on total_amount of goods
df['AMT_GOODS_PRICE_CAT'] = pd.qcut(df['AMT_GOODS_PRICE'],q=[0,0.2,0.5,0.8,0.
↪95,1],
       labels=['VeryLow','Low','Medium','High','VeryHigh'])
```

[133]:
```
df['AMT_GOODS_PRICE_CAT'].head()
```

[133]:
```
0        Low
1       High
2    VeryLow
3        Low
4     Medium
Name: AMT_GOODS_PRICE_CAT, dtype: category
Categories (5, object): ['VeryLow' < 'Low' < 'Medium' < 'High' < 'VeryHigh']
```

[ ]:
```
#Creating a categorical variable based on total_amount of annuity
df['AMT_ANNUITY_CAT'] = pd.qcut(df['AMT_ANNUITY'],q=[0,0.2,0.5,0.8,0.95,1],
       labels=['VeryLow','Low','Medium','High','VeryHigh'])
```

[134]:
```
df['AMT_ANNUITY_CAT'].head()
```

[134]:
```
0        Low
1     Medium
2    VeryLow
3     Medium
4        Low
Name: AMT_ANNUITY_CAT, dtype: category
Categories (5, object): ['VeryLow' < 'Low' < 'Medium' < 'High' < 'VeryHigh']
```

[135]:
```
df['DAYS_BIRTH'].max()
```

[135]:
```
69
```

[ ]:
```
#Creating a categorical variable based on total_amount of annuity
bins=[0,20,30,40,50,60,70]#Creating a categorical variable based on␣
 ↪total_amount of annuity
bins=[0,20,30,40,50,60,70]
labels=['0-20','21-30','31-40','41-50','51-60','61-70']
df['DAYS_BIRTH_CAT'] = pd.
 ↪cut(df['DAYS_BIRTH'],bins=bins,labels=labels,right=True)
labels=['0-20','21-30','31-40','41-50','51-60','61-70']
df['DAYS_BIRTH_CAT'] = pd.
 ↪cut(df['DAYS_BIRTH'],bins=bins,labels=labels,right=True)
```

[ ]:
```
df['DAYS_BIRTH_CAT'].head()
```

[ ]:
```
# Creating a new column determining the ratio of AMT_CREDIT and␣
 ↪AMT_INCOME_TOTAL.
df['CREDIT_INCOME_RATIO']=round((df['AMT_CREDIT']/df['AMT_INCOME_TOTAL']),1)
```

[ ]:
```
# Creating a new column determining the proportion of the individual's social␣
 ↪circle who defaulted after 30DPD.
df['30DPD_default_social_circle']=df['DEF_30_CNT_SOCIAL_CIRCLE']/
 ↪df['OBS_30_CNT_SOCIAL_CIRCLE']
```

[ ]:
```
df.
 ↪drop(columns=['DEF_30_CNT_SOCIAL_CIRCLE','OBS_30_CNT_SOCIAL_CIRCLE'],inplace=True)
```

[ ]:
```
df['30DPD_default_social_circle'] =␣
 ↪round(df['30DPD_default_social_circle']*100,2)
```

[ ]:
```
df['30DPD_default_social_circle'].fillna(value=0,inplace=True)
```

[136]:
```
df['30DPD_default_social_circle']
```

[136]:
```
0         100.0
1           0.0
2           0.0
3           0.0
4           0.0
          …
307506      0.0
307507      0.0
307508      0.0
307509      0.0
307510      0.0
Name: 30DPD_default_social_circle, Length: 307511, dtype: float64
```

[ ]:
```
# Creating a new column determining the proportion of the individual's social␣
↪circle who defaulted after 60DPD.

df['60DPD_default_social_circle']=df['DEF_60_CNT_SOCIAL_CIRCLE']/
↪df['OBS_60_CNT_SOCIAL_CIRCLE']
```

[ ]:
```
df.
↪drop(columns=['DEF_60_CNT_SOCIAL_CIRCLE','OBS_60_CNT_SOCIAL_CIRCLE'],inplace=True)
```

[ ]:
```
df['60DPD_default_social_circle'].fillna(value=0,inplace=True)
```

[ ]:
```
df['60DPD_default_social_circle'] =␣
↪round(df['60DPD_default_social_circle']*100,2)
df['60DPD_default_social_circle']
```

[ ]:
```
df.drop(columns=['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',
                ␣
↪'WEEKDAY_APPR_PROCESS_START','HOUR_APPR_PROCESS_START'],inplace=True)
```

[ ]:

[ ]:

0.6.5 8. Analysis

[ ]:
```
df['TARGET'].value_counts()
```

[ ]:
```
import plotly.graph_objects as go
fig = go.Figure()
```

```
fig = px.pie(df,names='TARGET', title='TARGET Variable - DEFAULTER Vs␣
 ↪NONDEFAULTER',
      height=450,width=600,template='plotly_white')
fig.update_traces(hoverinfo='label+percent',textfont_size=13,
                 textinfo='label+percent',pull=[0,0.2])
fig.update_layout(font_size=10)
fig.show()
```

Using a pie chart to identify the number of defaulters (TARGET = 1) and those who paid on time (TARGET = 0), it is evident that there is an imbalance between those who defaulted and those who did not.

[138]:
```
df['TARGET'].unique()
```

[138]:
```
array([1, 0], dtype=int64)
```

[139]:
```
#Splitting the data in two
df_1 = df[df['TARGET']==1]
df_0 = df[df['TARGET']==0]
```

[ ]:
```

```

0.6.6 8.1 Univariate Analysis of Categorical Variables

[ ]:
```
# function to count plot for categorical variables
def cat_plot(col):
    plt.style.use('ggplot')
    sns.despine
    fig,(ax1,ax2) = plt.subplots(1,2,figsize=(13,6))
    sns.countplot(x=col, data=df_1,ax=ax1,palette='Set1')
    ax1.set_ylabel('Total Count',fontweight="bold")
    ax1.set_xlabel(f'{col}', fontweight="bold")
    ax1.set_title(f'{col} disribution for Defaulters',fontsize=10)
    ax1.set_xticklabels(ax1.get_xticklabels(), rotation=45, ha="right")

# Adding the normalized percentage for easier comparision between defaulter and␣
 ↪non-defaulter
    for p in ax1.patches:
        ax1.annotate('{:.2f}%'.format((p.get_height()/len(df_1))*100), (p.
 ↪get_x()+0.05, p.get_height()+50))

    sns.countplot(x=col, data=df_0,ax=ax2,palette='Set1')
    ax2.set_ylabel('Total Count',fontweight="bold")
    ax2.set_xlabel(f'{col}', fontweight="bold")
    ax2.set_title(f'{col} distribution for NON_Defaulters',fontsize=10)
    ax2.set_xticklabels(ax2.get_xticklabels(), rotation=45, ha="right")
```

```
    # Adding the normalized percentage for easier comparision between defaulter␣
↪and non-defaulter
    for p in ax2.patches:
        ax2.annotate('{:.2f}%'.format((p.get_height()/len(df_0))*100), (p.
↪get_x()+0.05, p.get_height()+50))
    plt.subplots_adjust(wspace=0.2,hspace=.3)
    plt.show()
```

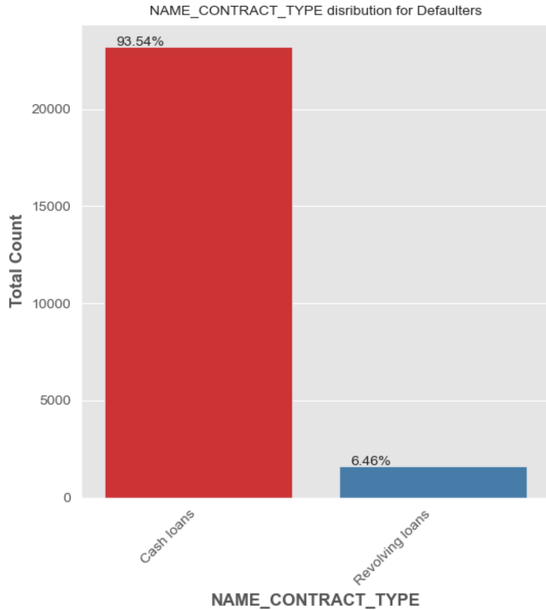
[ ]:
```
cat_cols=df.select_dtypes(exclude='number').columns
```

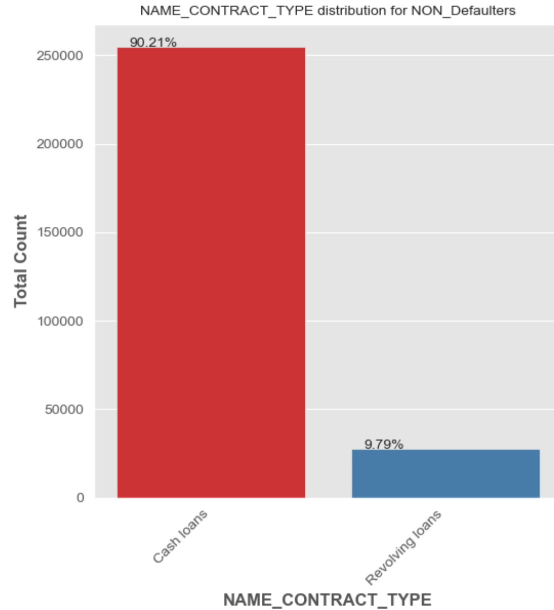
[162]:
```
for col in cat_cols:
    print(col, df[col].dtypes)
```

```
NAME_CONTRACT_TYPE object
CODE_GENDER object
FLAG_OWN_CAR object
FLAG_OWN_REALTY object
NAME_TYPE_SUITE object
NAME_INCOME_TYPE object
NAME_EDUCATION_TYPE object
NAME_FAMILY_STATUS object
NAME_HOUSING_TYPE object
AMT_INCOME_TOTAL_CAT category
AMT_CREDIT_CAT category
AMT_GOODS_PRICE_CAT category
AMT_ANNUITY_CAT category
DAYS_BIRTH_CAT category
```

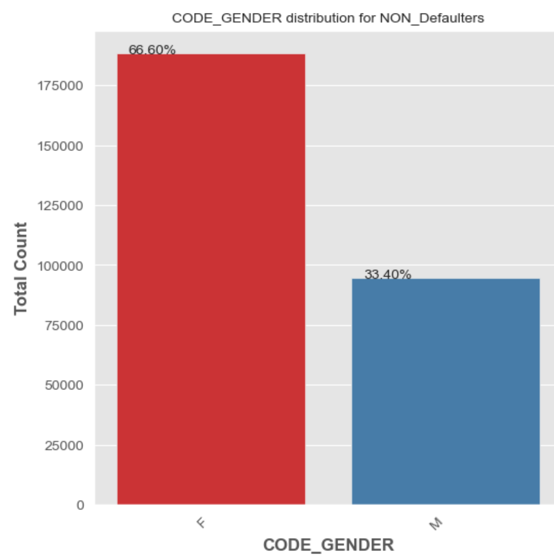
[ ]:
```
cat_cols=cat_cols.drop(['OCCUPATION_TYPE','ORGANIZATION_TYPE'])
```

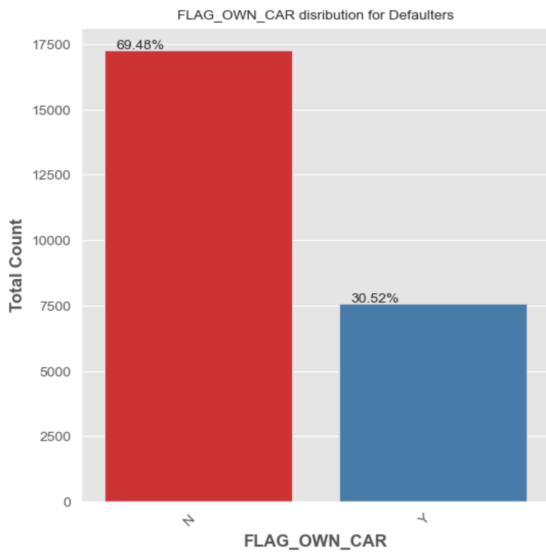
[126]:
```
for col in cat_cols:
    cat_plot(col)
```

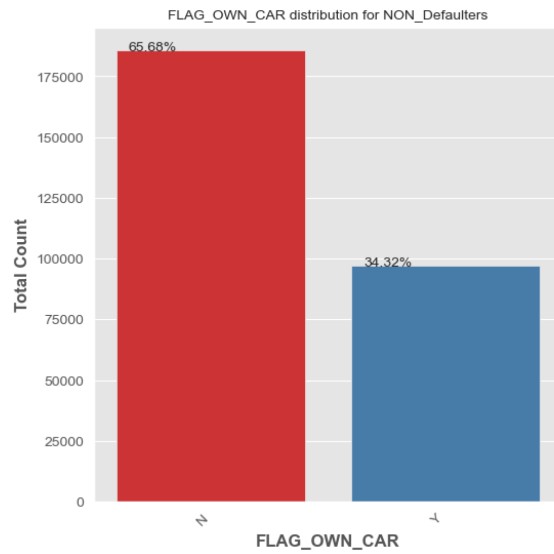

NAME_CONTRACT_TYPE disribution for Defaulters
93.54%
6.46%
Total Count
Cash loans
Revolving loans
NAME_CONTRACT_TYPE

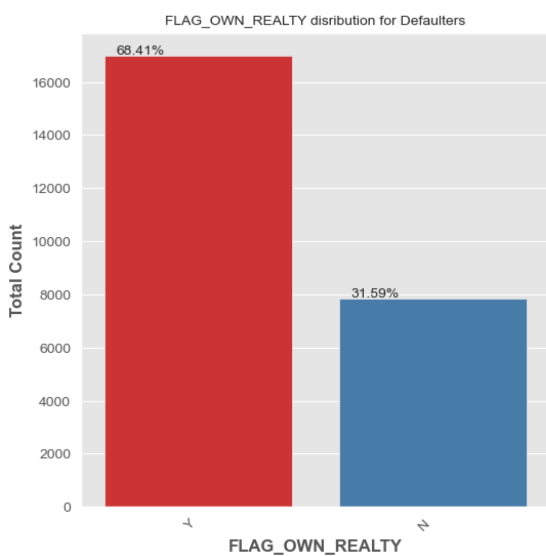

NAME_CONTRACT_TYPE distribution for NON_Defaulters
90.21%
9.79%
Total Count
Cash loans
Revolving loans
NAME_CONTRACT_TYPE

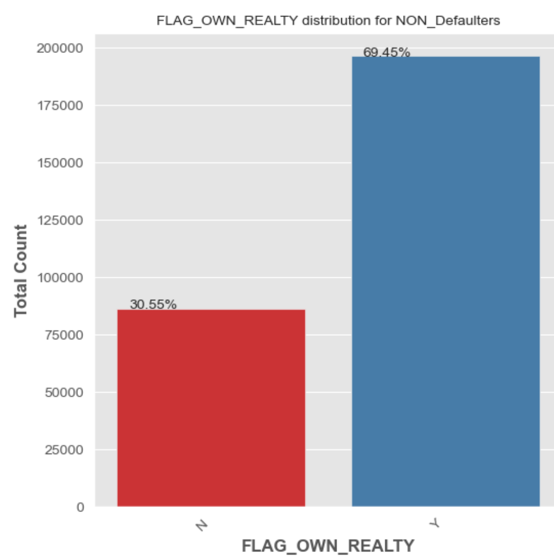

CODE_GENDER disribution for Defaulters
57.08%
42.92%
Total Count
M
F
CODE_GENDER

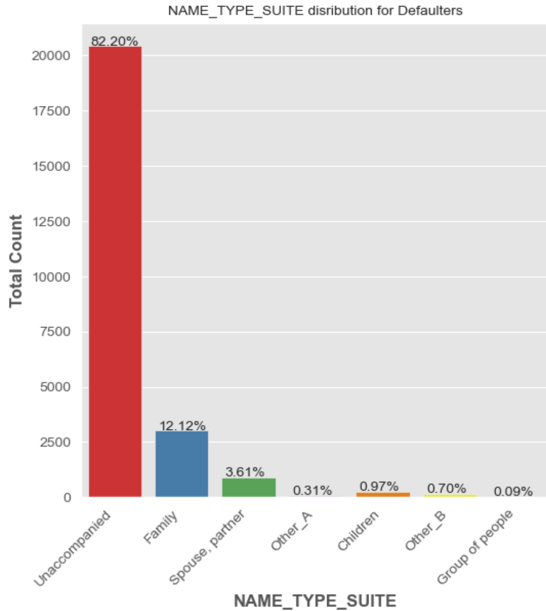

CODE_GENDER distribution for NON_Defaulters
66.60%
33.40%
Total Count
F
M
CODE_GENDER

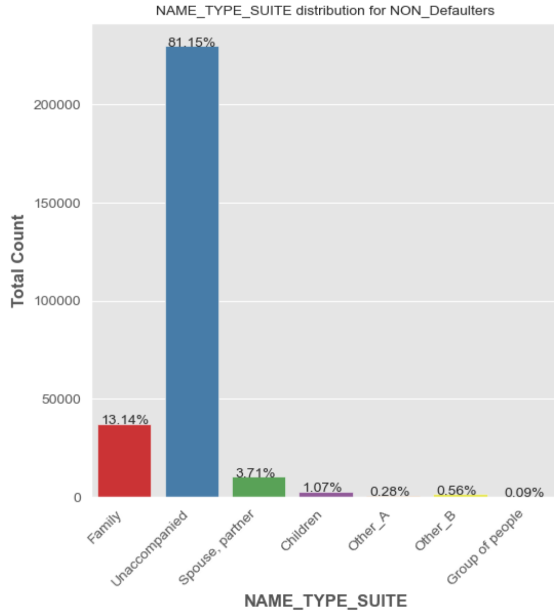

FLAG_OWN_CAR disribution for Defaulters
Total Count
0
2500
5000
7500
10000
12500
15000
17500
69.48%
30.52%
N
Y
FLAG_OWN_CAR

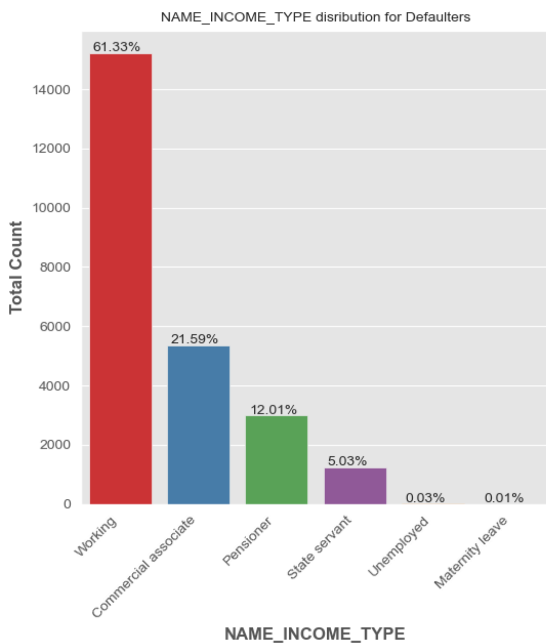

FLAG_OWN_CAR distribution for NON_Defaulters
Total Count
0
25000
50000
75000
100000
125000
150000
175000
65.68%
34.32%
N
Y
FLAG_OWN_CAR

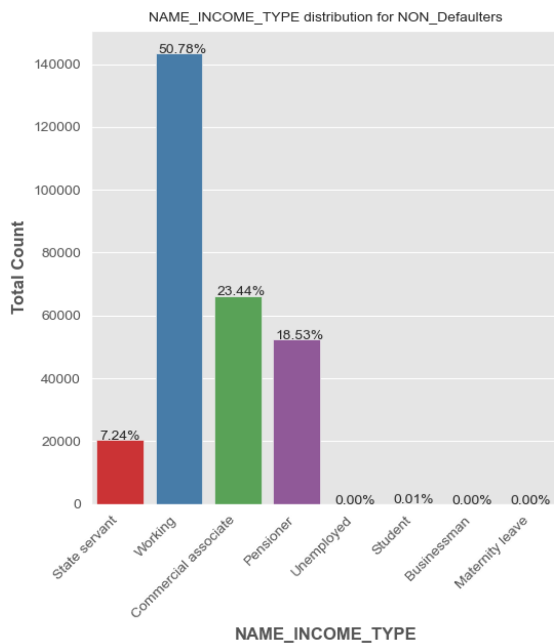

FLAG_OWN_REALTY disribution for Defaulters
Total Count
0
2000
4000
6000
8000
10000
12000
14000
16000
68.41%
31.59%
Y
N
FLAG_OWN_REALTY

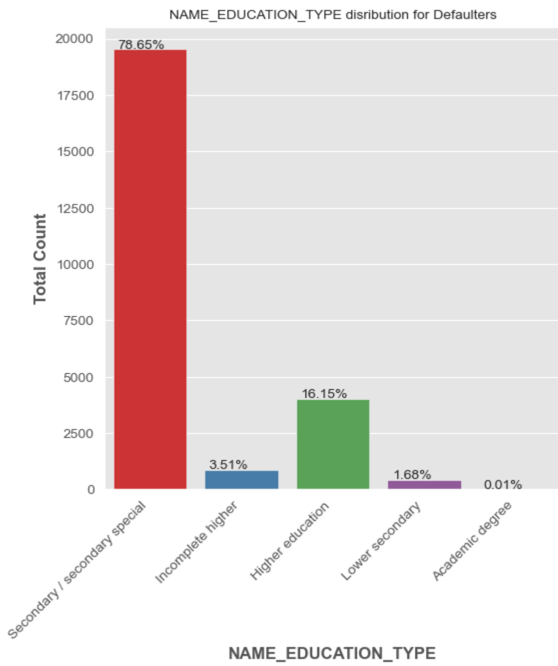

FLAG_OWN_REALTY distribution for NON_Defaulters
Total Count
0
25000
50000
75000
100000
125000
150000
175000
200000
30.55%
69.45%
N
Y
FLAG_OWN_REALTY

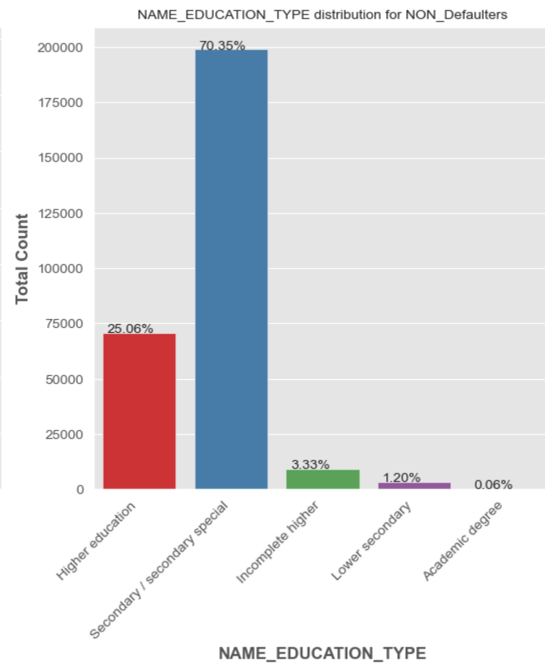

NAME_TYPE_SUITE disribution for Defaulters
Total Count
0
2500
5000
7500
10000
12500
15000
17500
20000
82.20%
12.12%
3.61%
0.31%
0.97%
0.70%
0.09%
Unaccompanied
Family
Spouse, partner
Other_A
Children
Other_B
Group of people
NAME_TYPE_SUITE

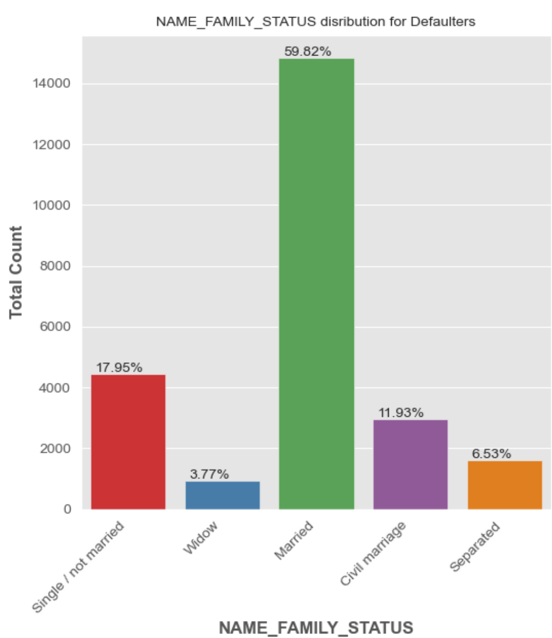

NAME_TYPE_SUITE distribution for NON_Defaulters
Total Count
0
50000
100000
150000
200000
13.14%
81.15%
3.71%
1.07%
0.28%
0.56%
0.09%
Family
Unaccompanied
Spouse, partner
Children
Other_A
Other_B
Group of people
NAME_TYPE_SUITE

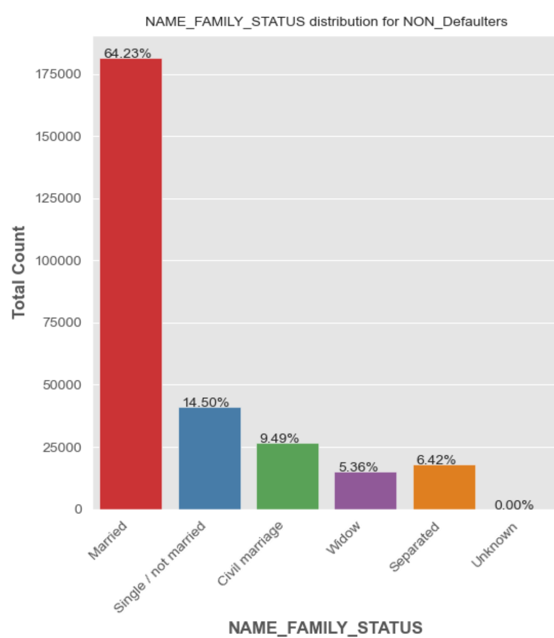

NAME_INCOME_TYPE disribution for Defaulters
Total Count
0
2000
4000
6000
8000
10000
12000
14000
61.33%
21.59%
12.01%
5.03%
0.03%
0.01%
Working
Commercial associate
Pensioner
State servant
Unemployed
Maternity leave
NAME_INCOME_TYPE

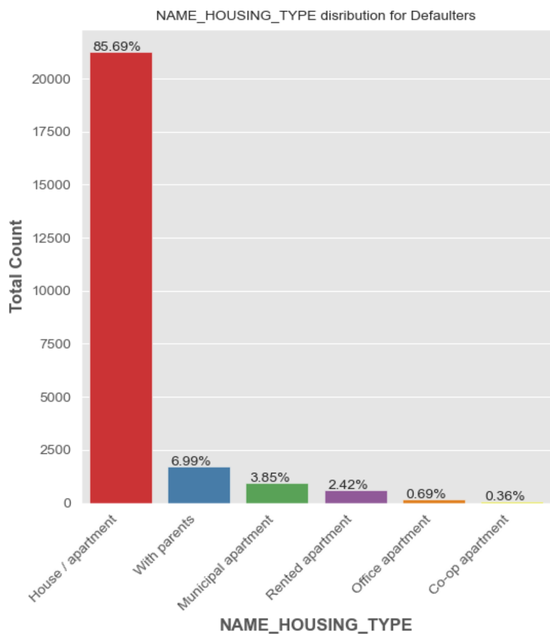

NAME_INCOME_TYPE distribution for NON_Defaulters
Total Count
0
20000
40000
60000
80000
100000
120000
140000
7.24%
50.78%
23.44%
18.53%
0.00%
0.01%
0.00%
0.00%
State servant
Working
Commercial associate
Pensioner
Unemployed
Student
Businessman
Maternity leave
NAME_INCOME_TYPE

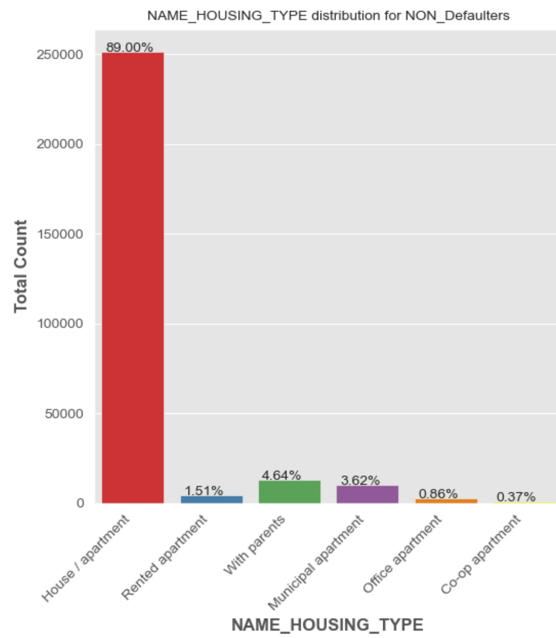

NAME_EDUCATION_TYPE disribution for Defaulters
Total Count
NAME_EDUCATION_TYPE
78.65%
3.51%
16.15%
1.68%
0.01%
Secondary / secondary special
Incomplete higher
Higher education
Lower secondary
Academic degree

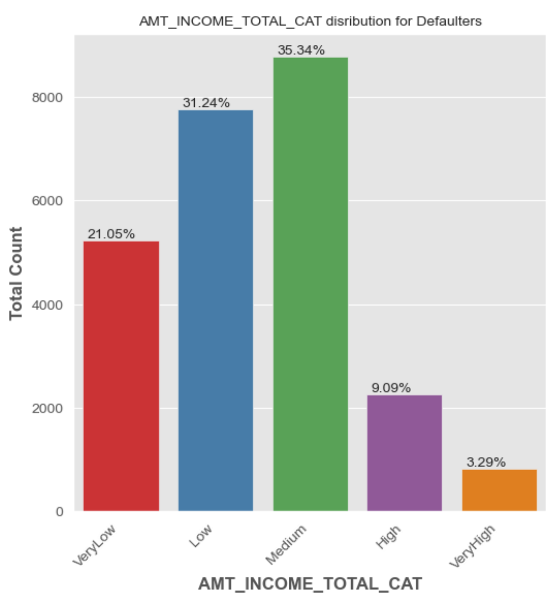

NAME_EDUCATION_TYPE distribution for NON_Defaulters
Total Count
NAME_EDUCATION_TYPE
25.06%
70.35%
3.33%
1.20%
0.06%
Higher education
Secondary / secondary special
Incomplete higher
Lower secondary
Academic degree

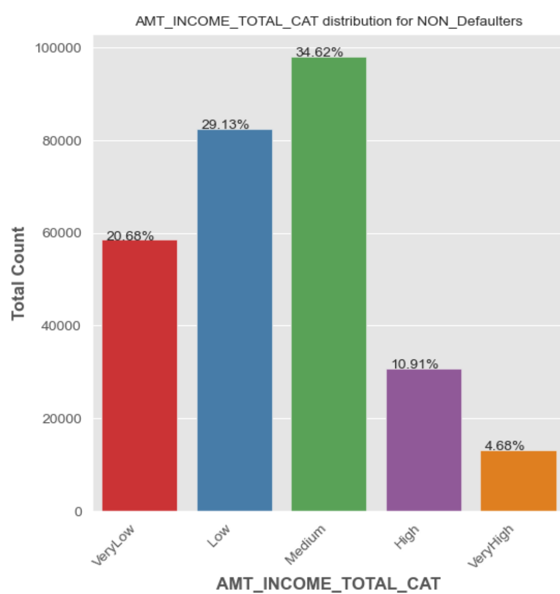

NAME_FAMILY_STATUS disribution for Defaulters
Total Count
NAME_FAMILY_STATUS
17.95%
3.77%
59.82%
11.93%
6.53%
Single / not married
Widow
Married
Civil marriage
Separated

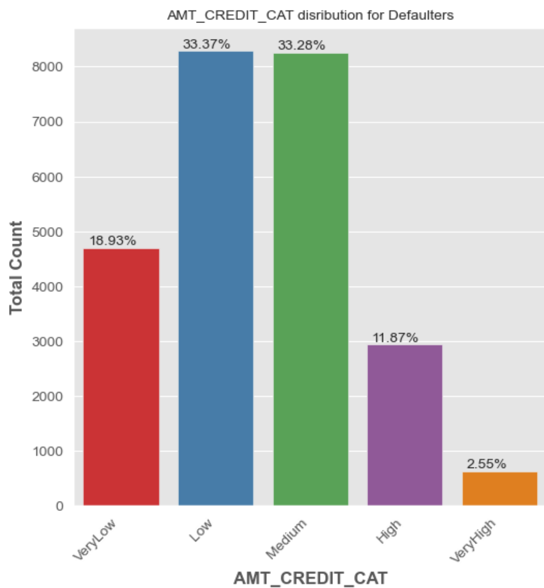

NAME_FAMILY_STATUS distribution for NON_Defaulters
Total Count
NAME_FAMILY_STATUS
64.23%
14.50%
9.49%
5.36%
6.42%
0.00%
Married
Single / not married
Civil marriage
Widow
Separated
Unknown

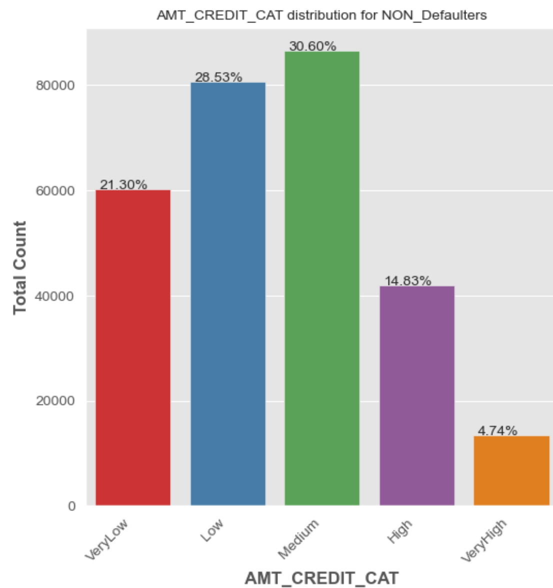

NAME_HOUSING_TYPE disribution for Defaulters
Total Count
NAME_HOUSING_TYPE
House / apartment
With parents
Municipal apartment
Rented apartment
Office apartment
Co-op apartment
85.69%
6.99%
3.85%
2.42%
0.69%
0.36%

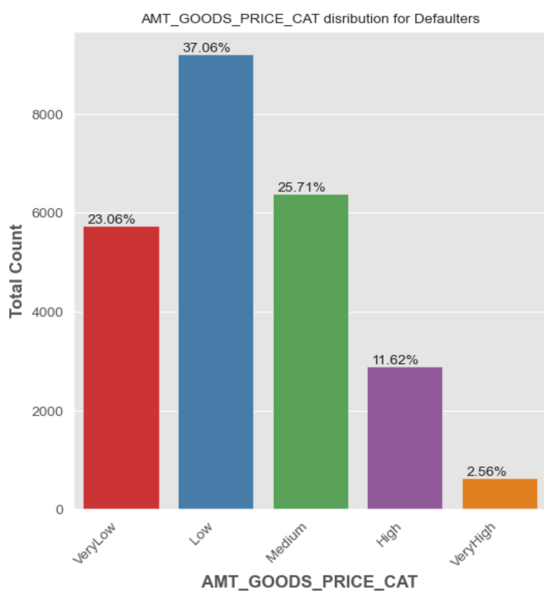

NAME_HOUSING_TYPE distribution for NON_Defaulters
Total Count
NAME_HOUSING_TYPE
House / apartment
Rented apartment
With parents
Municipal apartment
Office apartment
Co-op apartment
89.00%
1.51%
4.64%
3.62%
0.86%
0.37%

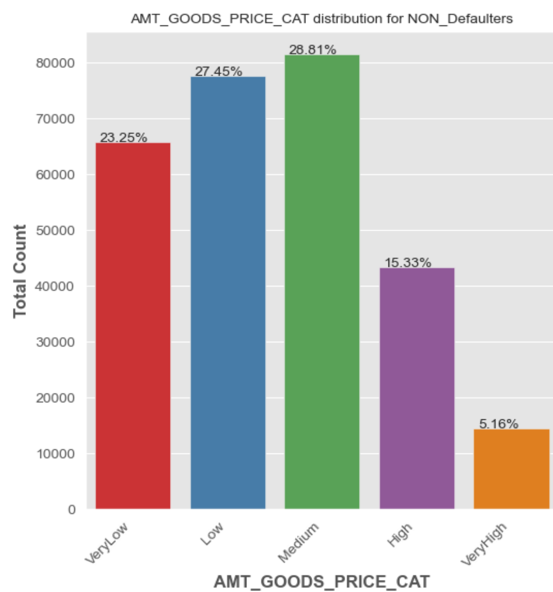

AMT_INCOME_TOTAL_CAT disribution for Defaulters
Total Count
AMT_INCOME_TOTAL_CAT
VeryLow
Low
Medium
High
VeryHigh
21.05%
31.24%
35.34%
9.09%
3.29%

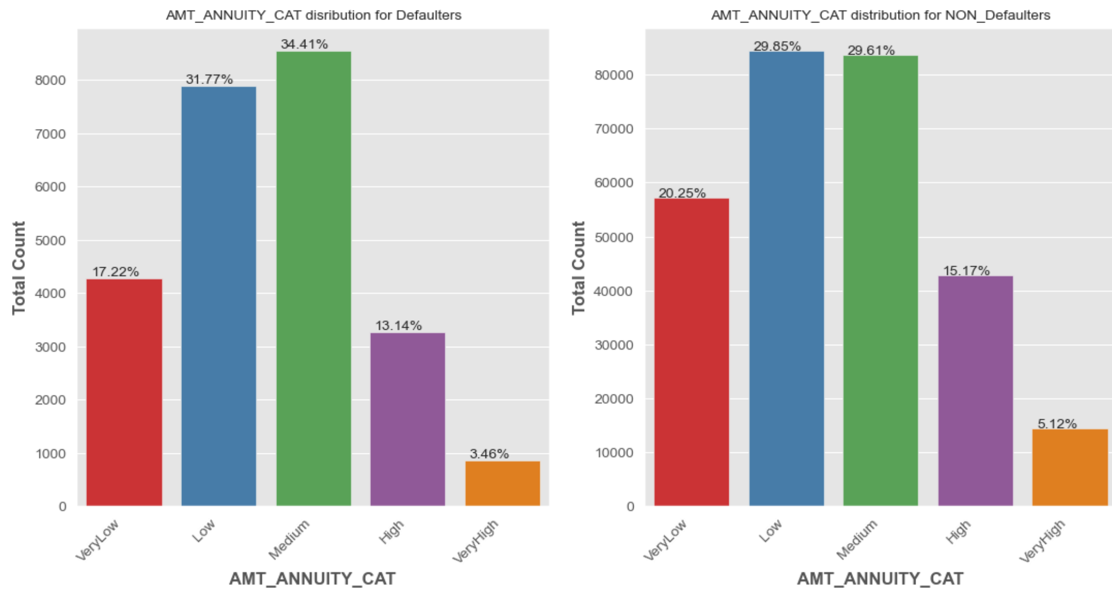

AMT_INCOME_TOTAL_CAT distribution for NON_Defaulters
Total Count
AMT_INCOME_TOTAL_CAT
VeryLow
Low
Medium
High
VeryHigh
20.68%
29.13%
34.62%
10.91%
4.68%

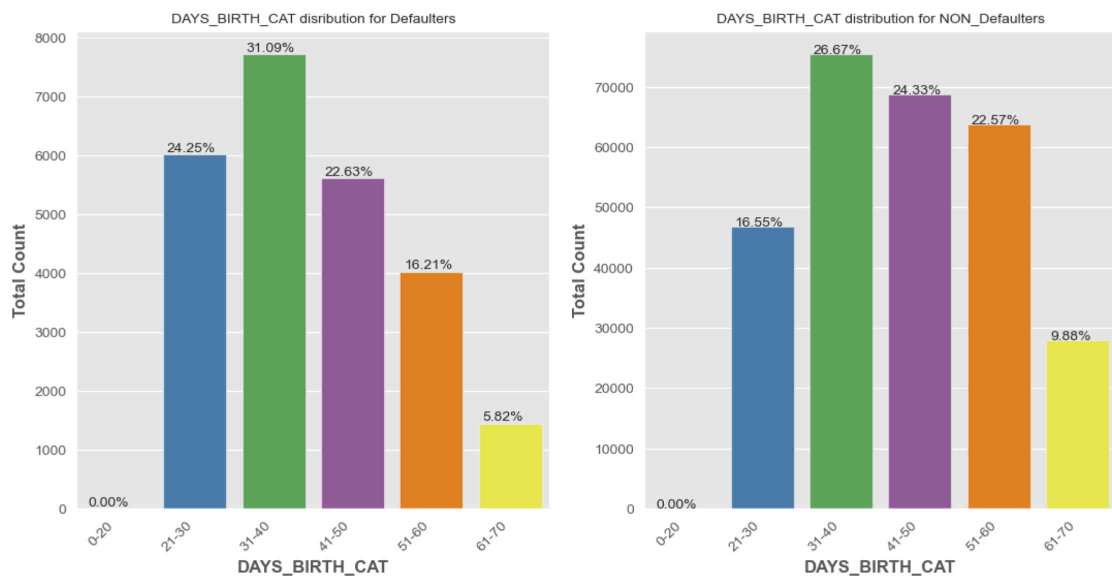

AMT_CREDIT_CAT disribution for Defaulters
Total Count
AMT_CREDIT_CAT
VeryLow
Low
Medium
High
VeryHigh
18.93%
33.37%
33.28%
11.87%
2.55%


AMT_CREDIT_CAT distribution for NON_Defaulters
Total Count
AMT_CREDIT_CAT
VeryLow
Low
Medium
High
VeryHigh
21.30%
28.53%
30.60%
14.83%
4.74%


AMT_GOODS_PRICE_CAT disribution for Defaulters
Total Count
AMT_GOODS_PRICE_CAT
VeryLow
Low
Medium
High
VeryHigh
23.06%
37.06%
25.71%
11.62%
2.56%


AMT_GOODS_PRICE_CAT distribution for NON_Defaulters
Total Count
AMT_GOODS_PRICE_CAT
VeryLow
Low
Medium
High
VeryHigh
23.25%
27.45%
28.81%
15.33%
5.16%






```
[ ]: from IPython.core.interactiveshell import InteractiveShell
     InteractiveShell.ast_node_interactivity = "all"
```

Inferences:

NAME_CONTRACT_TYPE :

- Cash Loan contracts have a higher number of credit than revolving loan contracts (Defaulters).

- Cash Loan contracts have a higher number of credit than revolving loan contracts (Non-Defaulters).
- Count of females is more.

CODE_GENDER :

- Females account for 67% of the non-defaulters and 55% of the defaulters, respectively. We may assume that because more women seek loans than men, there are also more women who fail on their payments. However, the default rate of FEMALES is much lower than that of their MALE counterparts.

FLAG_OWN_CAR :

- Automobile owners make up 65.7% of the non-defaulters and 69.5% of the defaulters. While persons with vehicles are more likely to default, the explanation may be that there are more people without cars. Considering the percentages in both figures, we can deduce that the default rate of automobile owners is lower than that of carless individuals.

NAME_TYPE_SUIT :

- The majority of loan applicants were accompanied throughout the application process. And with a few customers, a family member was present for both Defaulters and Non-Defaulters, but the presence of a family member during loan application had no impact on default. Moreover, both populations have identical proportions.

NAME_INCOME_TYPE :

- State Servant and Businessman are at minimal risk of default.
- Most of the loans are distributed to working-class people. Additionally, we see that the working class contributes 51% to non-defaulters but 61% to defaulters. Clearly, the likelihood of default is greater for them.

NAME_EDUCATION_TYPE :

- Except for those with a higher level of education, who are less likely to fail, and those with a secondary level of education, who are more likely to default, almost all Education groups are equally likely to default.

NAME_FAMILY_STATUS :

- Married Clients seem to apply most for the loan compared to others for both Defaulters and Non-Defaulters.
- The graph, however, reveals that Single/non-Married individuals contribute 14.5% to Non-Defaulters and 18% to Defaulters. Therefore, there is a greater danger associated with them.

NAME_HOUSING_TYPE :

- It is evident from the graph that homeowners/tenants are more likely to seek loans. People who live with their parents tend to default more often than others. Due to their parents living with them, their living expenditures might be greater.

NAME_INCOME_CAT :

- The Very High Income category defaults less often. They provide 12.4% to the overall number of defaulters but 15.6% to the number of Non-Defaulters.

AMT_CREDIT_CAT :

- The Very High Credit category defaults less often, the greatest risk is related to those who belong to the Low to Medium Credit amount category.

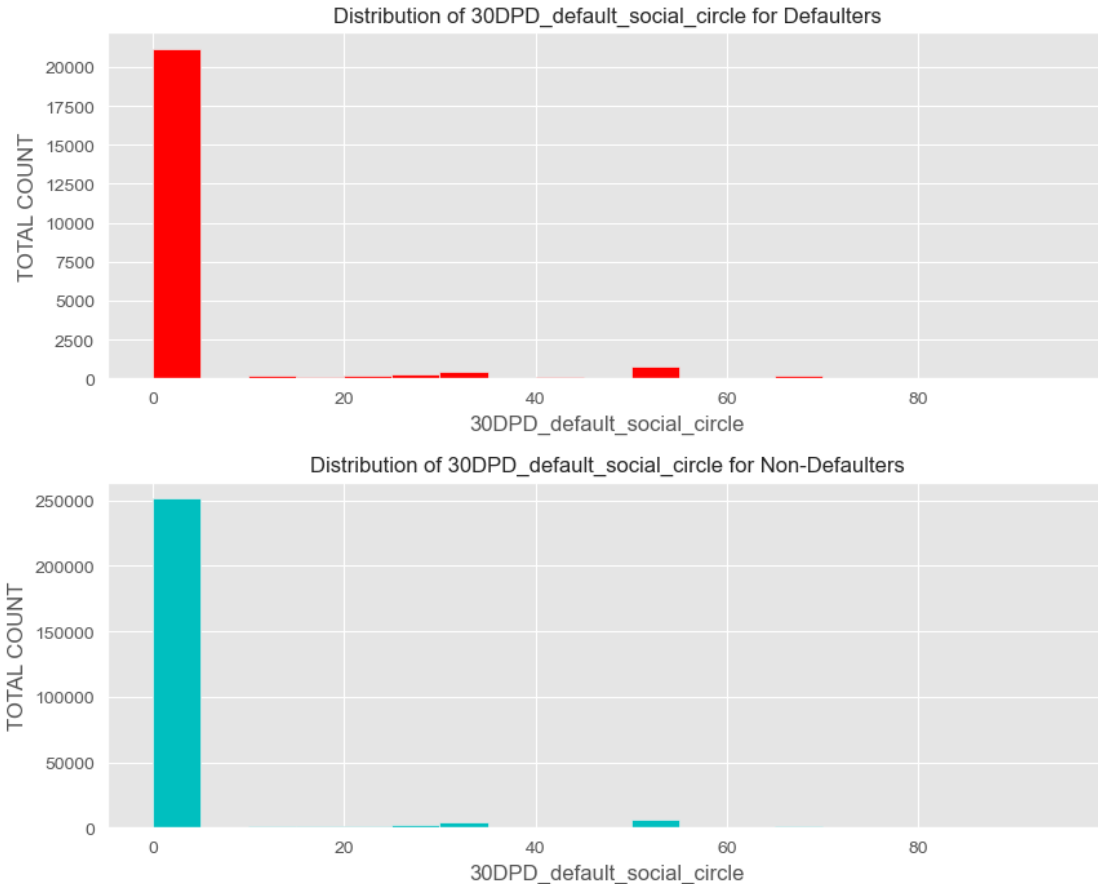
AMT_GOODS_PRICE_CAT :

- The High & Very High Good Price category defaults less often, the greatest risk is related to those who belong to the Low and Very Low Good price category.

DAYS_BIRTH_CAT :

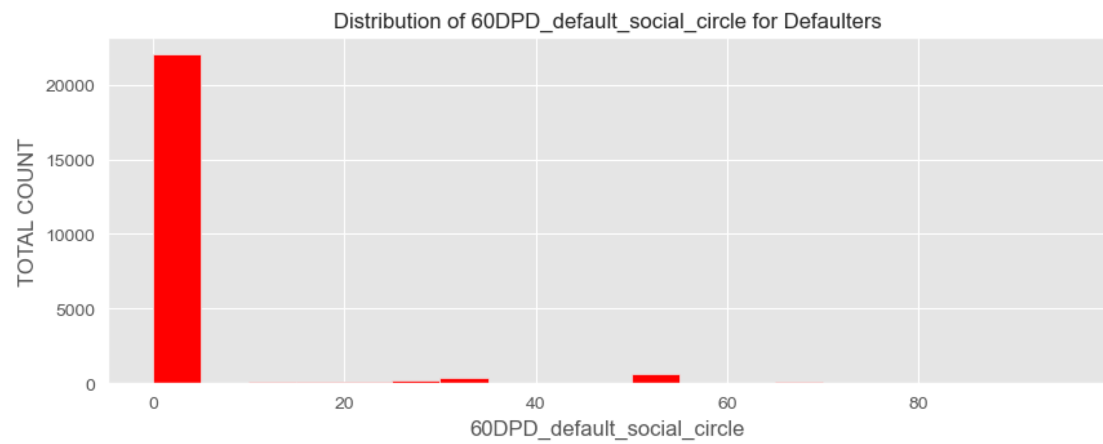
- We see that those between the ages of 20 and 40 tend to default more often. Therefore, they are the riskiest borrowers. Beginning at age 40, individuals tend to default less often as their age increases. One of the reasons might be because people find employment around that age, and their income improves with age.

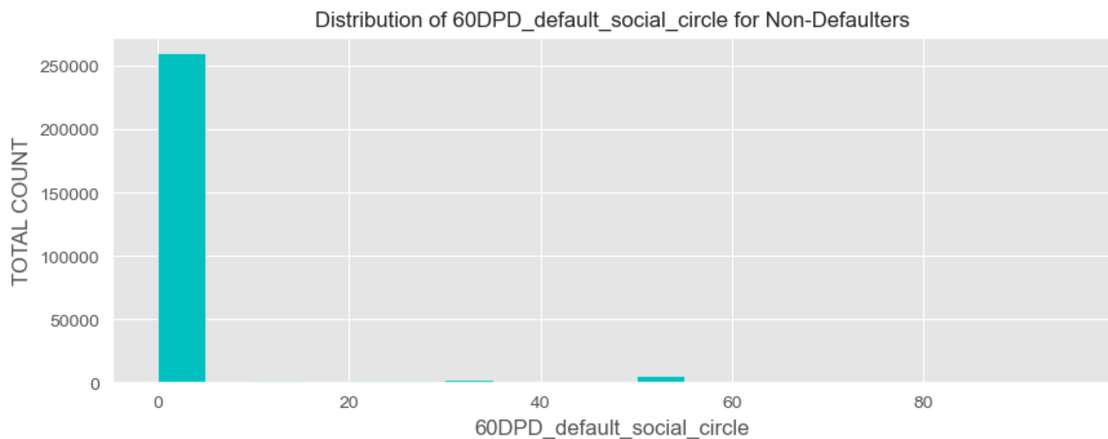
[140]:
```
plt.figure(figsize=(10,8))

plt.subplot(211)
df_1['30DPD_default_social_circle'].plot.hist(bins=np.arange(0,100,5),color="r")
plt.title('Distribution of 30DPD_default_social_circle for␣
 ↪Defaulters',fontsize=12)
plt.xlabel('30DPD_default_social_circle')
plt.ylabel('TOTAL COUNT')

plt.subplot(212)
df_0['30DPD_default_social_circle'].plot.hist(bins=np.arange(0,100,5),color='c')
plt.title('Distribution of 30DPD_default_social_circle for␣
 ↪Non-Defaulters',fontsize=12)
plt.xlabel('30DPD_default_social_circle')
plt.ylabel('TOTAL COUNT')

plt.subplots_adjust(hspace=.3)

plt.show();
```



[141]:

```
plt.figure(figsize=(10,8))

plt.subplot(211)
df_1['60DPD_default_social_circle'].plot.hist(bins=np.arange(0,100,5),color='r')
plt.title('Distribution of 60DPD_default_social_circle for␣
  ↪Defaulters',fontsize=12)
plt.xlabel('60DPD_default_social_circle')
plt.ylabel('TOTAL COUNT')

plt.subplot(212)
df_0['60DPD_default_social_circle'].plot.hist(bins=np.arange(0,100,5),color='c')
plt.title('Distribution of 60DPD_default_social_circle for␣
  ↪Non-Defaulters',fontsize=12)
plt.xlabel('60DPD_default_social_circle')
plt.ylabel('TOTAL COUNT')

plt.subplots_adjust(hspace=.3)

plt.show();
```



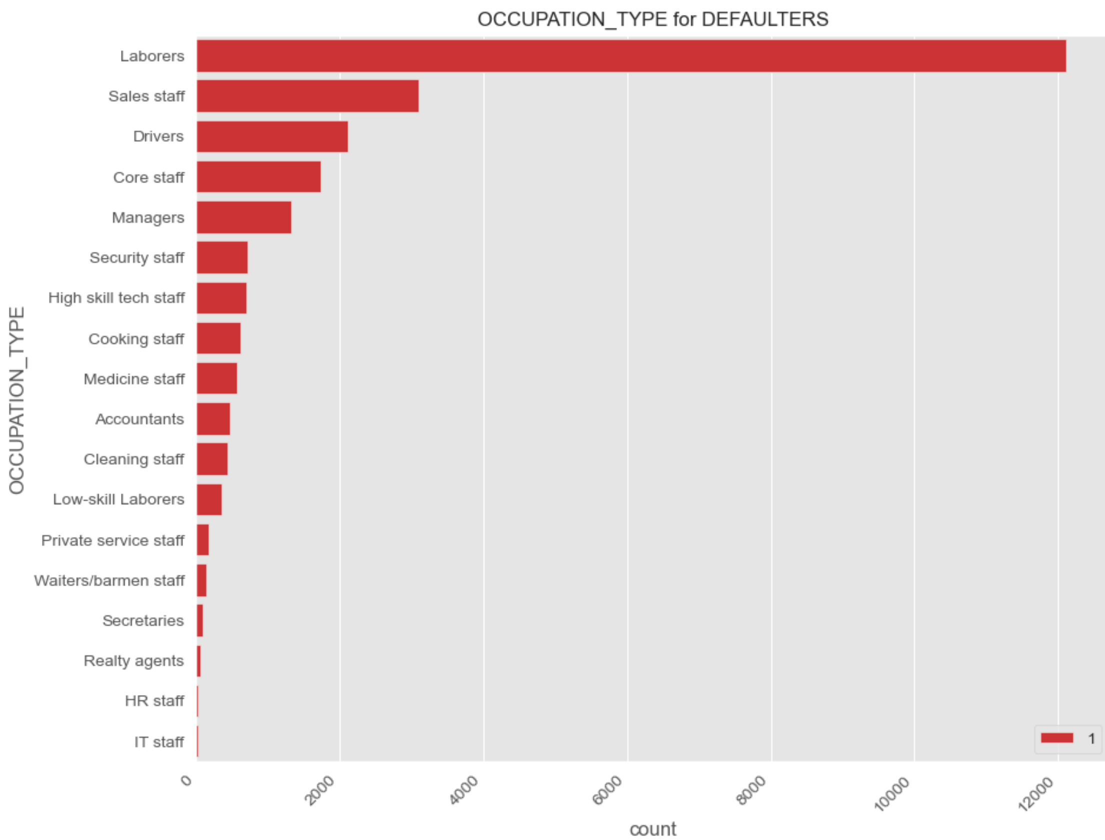


[ ]:

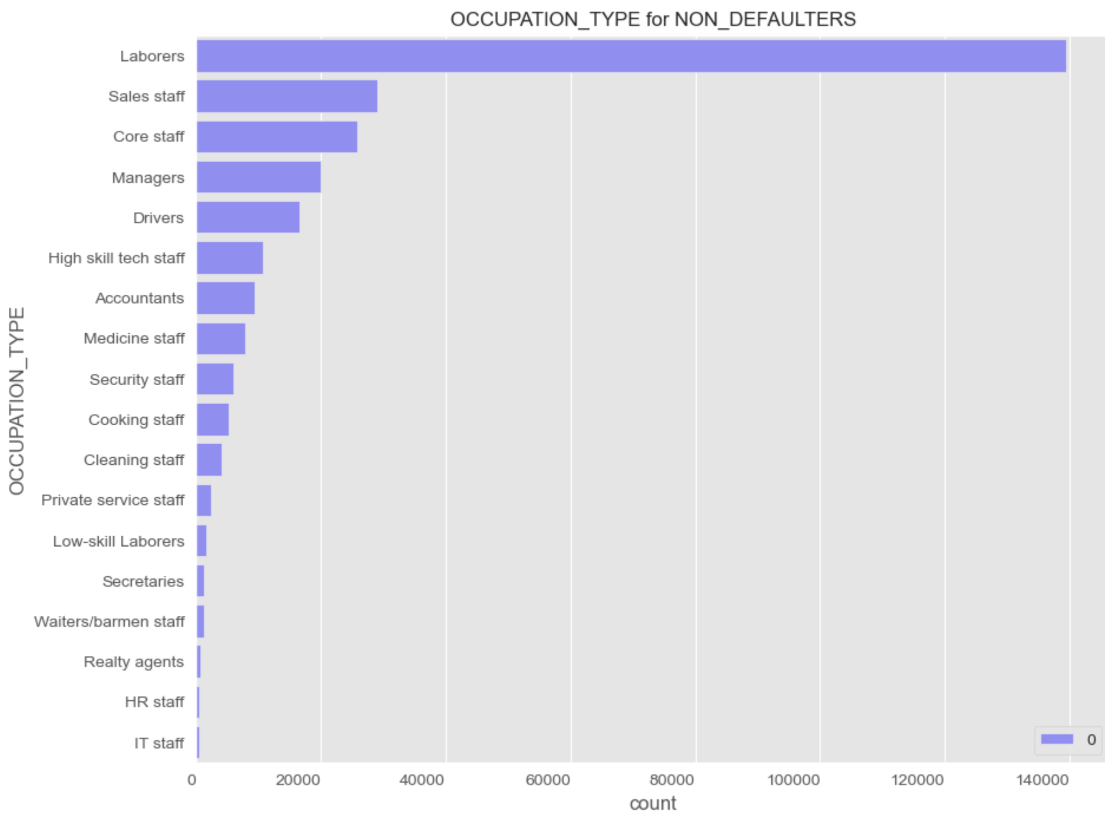
```
```

[142]:

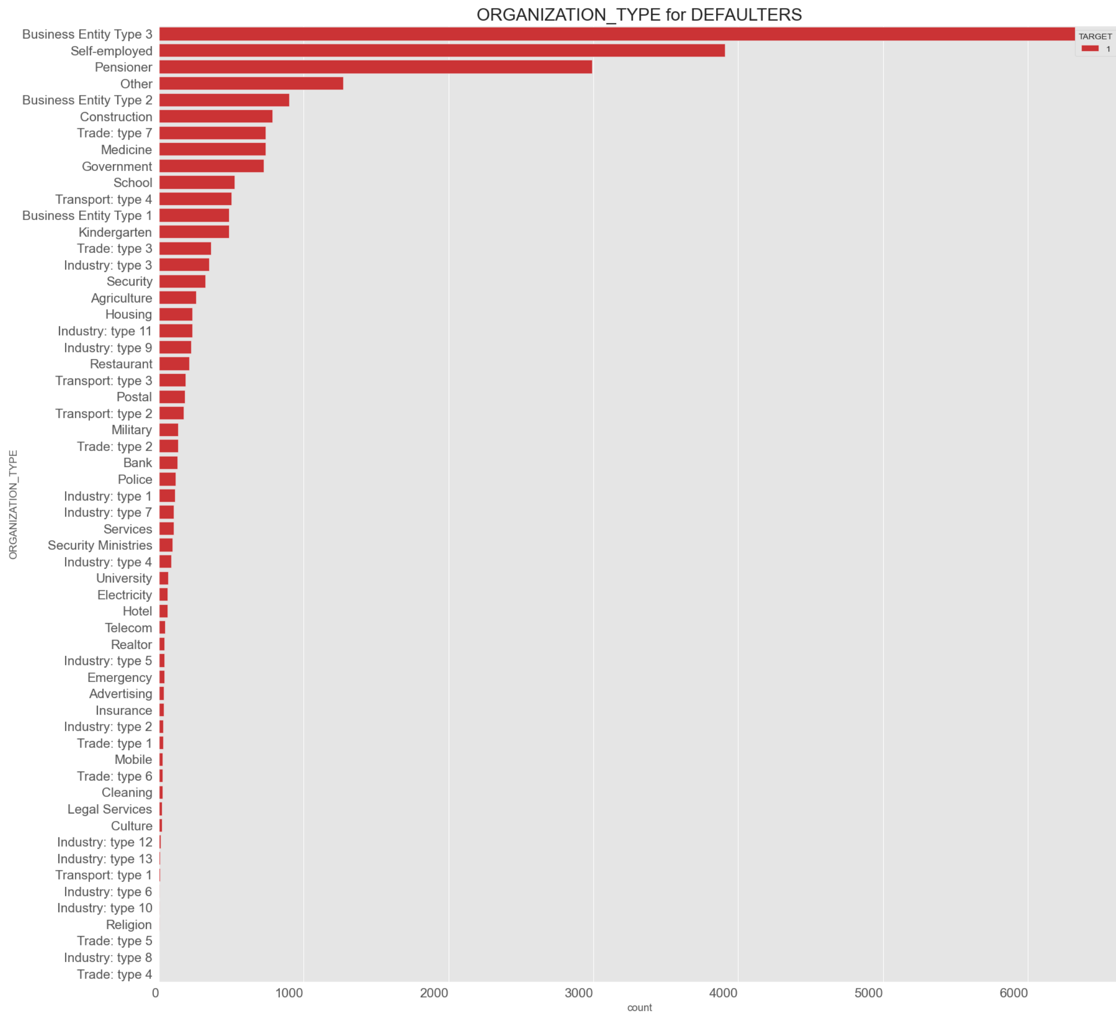
```
# Convert the 'TARGET' column to strings
df_1['TARGET'] = df_1['TARGET'].astype(str)

plt.figure(figsize=(10,8))

sns.countplot(data=df_1, y='OCCUPATION_TYPE', hue='TARGET',
              order=df_1['OCCUPATION_TYPE'].value_counts(ascending=False).
 ↪index, palette='Set1')

plt.title('OCCUPATION_TYPE for DEFAULTERS', fontsize=12)
plt.xticks(rotation=45, ha="right", fontsize=10)
plt.legend(loc='lower right')

plt.show();
```



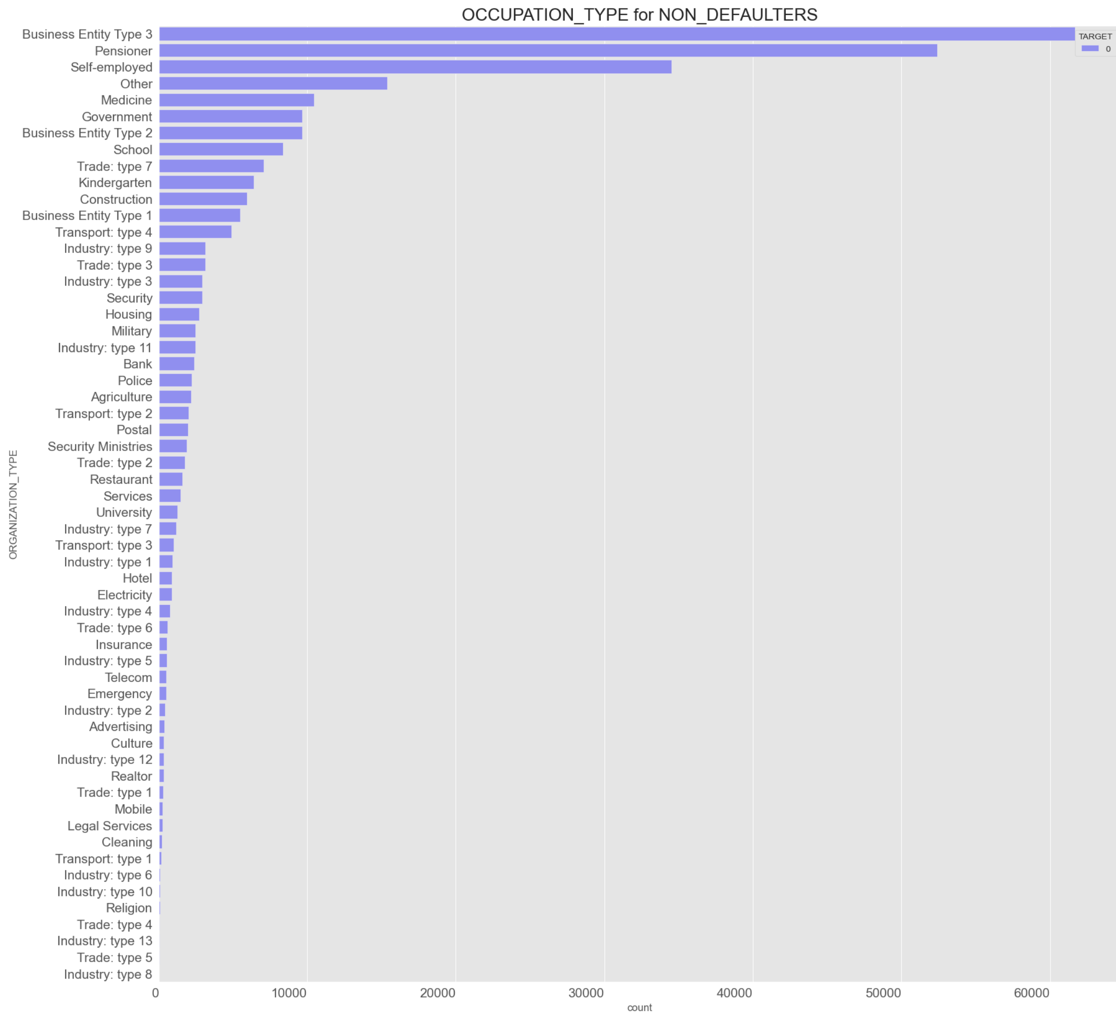
[143]:
```
df_0['TARGET'] = df_0['TARGET'].astype(str)

plt.figure(figsize=(10,8))

sns.countplot(data=df_0, y='OCCUPATION_TYPE', hue='TARGET',
              order=df_0['OCCUPATION_TYPE'].value_counts().index,␣
 ↪palette='cool')

plt.title('OCCUPATION_TYPE for NON_DEFAULTERS', fontsize=12)
plt.xticks(rotation=360, ha="right")
plt.yticks(fontsize=10)
plt.legend(loc='lower right')

plt.show();
```



[144]:
```
plt.figure(figsize=(20,20))

sns.countplot(data=df_1,y='ORGANIZATION_TYPE',hue='TARGET',
              order=df_1['ORGANIZATION_TYPE'].value_counts().
 ↪index,palette='Set1')
plt.title('ORGANIZATION_TYPE for DEFAULTERS',fontsize=20)
plt.xticks( rotation=360, ha="right",fontsize=15)
plt.yticks(fontsize=15)

plt.show();
```



[145]:
```
plt.figure(figsize=(20,20))

sns.countplot(data=df_0,y='ORGANIZATION_TYPE',hue='TARGET',
              order=df_0['ORGANIZATION_TYPE'].value_counts().
 ↪index,palette='cool')
plt.title('OCCUPATION_TYPE for NON_DEFAULTERS',fontsize=20)
plt.xticks( rotation=360, ha="right",fontsize=15)
plt.yticks( fontsize=15)

plt.show();
```



The majority of credit-seeking clients are Business entity Type 3, Self-employed, Other, Medicine, and Government organisations. Fewer customers come from Industry types 8, 6, 10, religion and trade types 5, 4

[ ]:

```

```

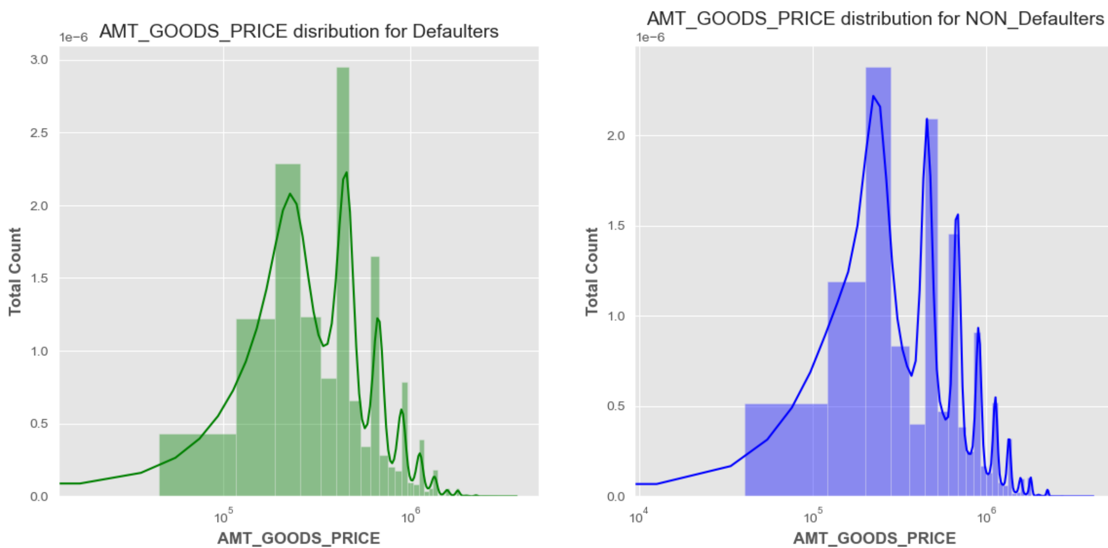
0.6.7 8.2 Univariate Analysis of Numerical Variables

[146]:

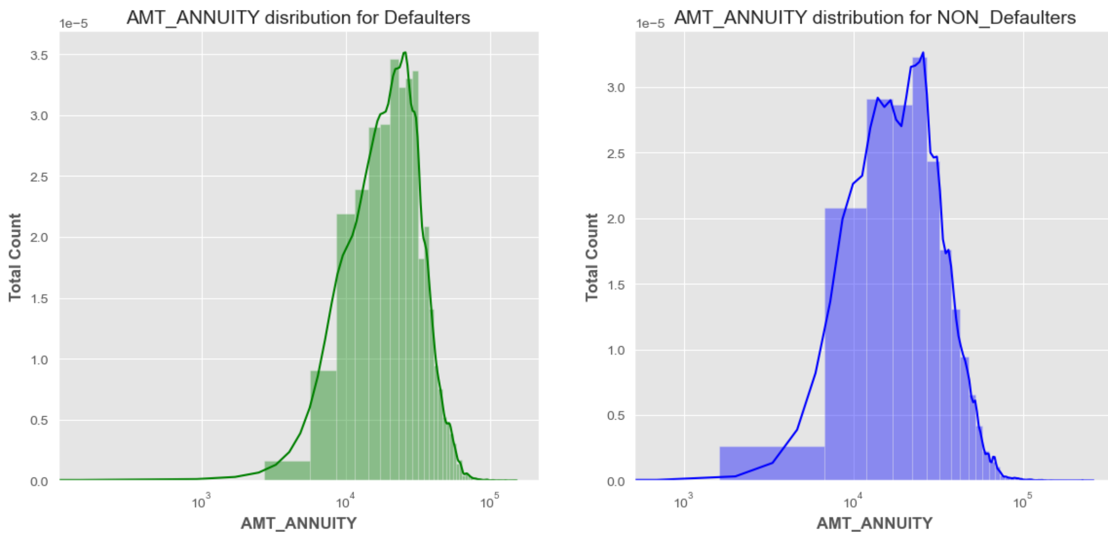
```
# function to count plot for numerical variables
def num_plot(col):
    plt.style.use('ggplot')
    sns.despine
    fig,(ax1,ax2) = plt.subplots(1,2,figsize=(14,6))
    sns.distplot(x=df_1[col],ax=ax1,kde=True,color='green')
    ax1.set_ylabel('Total Count',fontweight="bold")
    ax1.set_xlabel(f'{col}', fontweight="bold")
```

```
    ax1.set_title(f'{col} disribution for Defaulters',fontsize=14)
    ax1.set_xscale('log')

    sns.distplot(x=df_0[col],ax=ax2,kde=True,color='b')
    ax2.set_ylabel('Total Count',fontweight="bold")
    ax2.set_xlabel(f'{col}', fontweight="bold")
    ax2.set_title(f'{col} distribution for NON_Defaulters',fontsize=14)
    ax2.set_xscale('log')

    plt.subplots_adjust(wspace=0.2)
    plt.show()
```

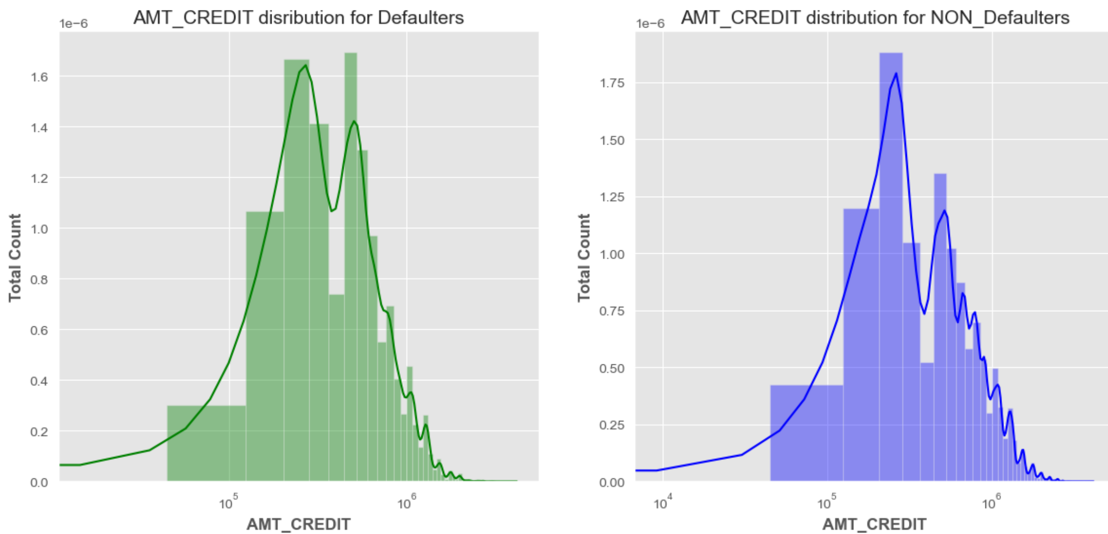
[147]:
```
num_plot('AMT_GOODS_PRICE')
```



[148]:
```
num_plot('AMT_ANNUITY')
```



[149]:
```
num_plot('AMT_CREDIT')
```

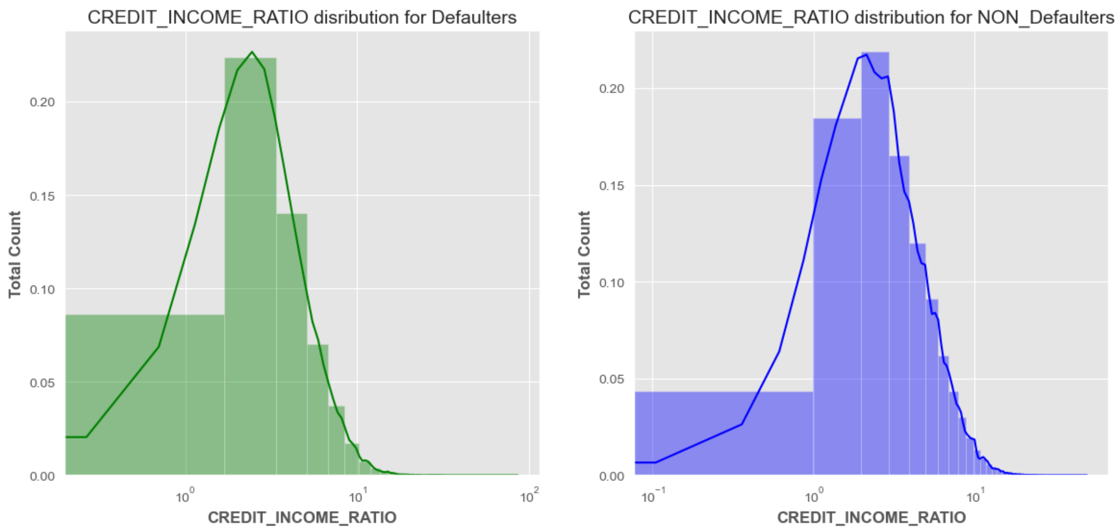


Dist. plot highlights the curve shape which is wider for Defaulters in comparison to Non-Defaulters which is narrower with well-defined edges.

People with Payment difficulties has largely staggered income as compared to people who dosen't.

Dist. plot clearly shows that the shape in Income total, Annuity, Credit and Good Price is similar for Target 0 and similar for Target 1

[150]:
```
num_plot('CREDIT_INCOME_RATIO')
```



CREDIT_INCOME_RATIO = CREDIT_AMOUNT/AMT_INCOME_TOTAL

Although there does not seem to be an obvious distinction between the group that defaulted and the group that did not, we can see that when the CREDIT INCOME RATIO is more than 50, individuals default.

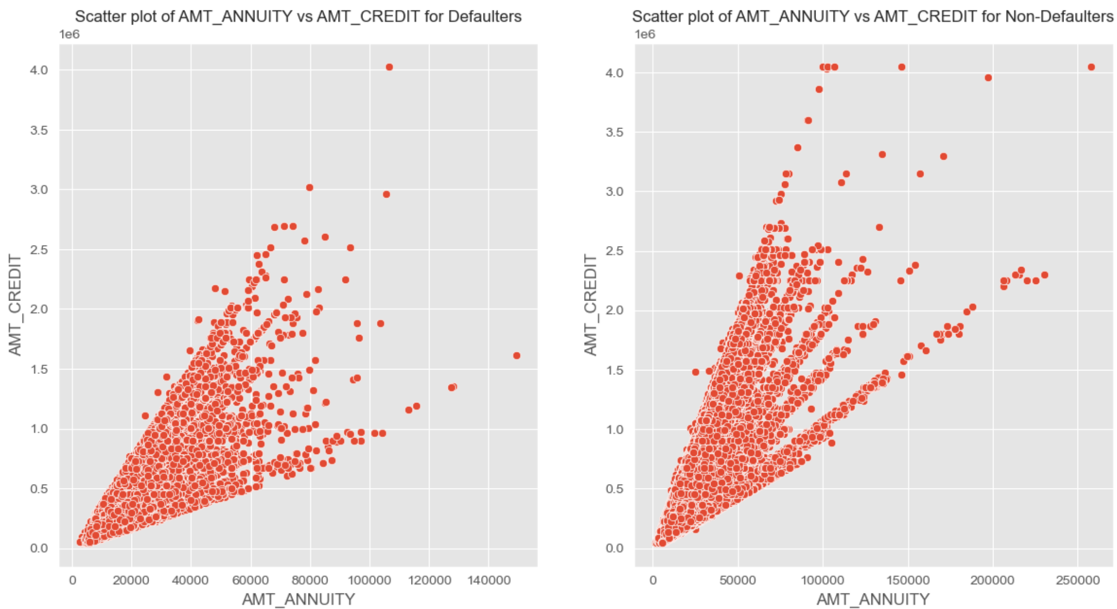
[ ]:
```

```

0.6.8 9.Bivariate Analysis

9.1 Bivariate Analysis of Numerical Columns

[151]:
```
def bivarnum_plot(col1,col2):
    fig,(ax1,ax2) = plt.subplots(1,2,figsize=(14,7))
    sns.scatterplot(data=df_1,x=col1,y=col2,ax=ax1,)
    ax1.set_xlabel(col1)
    ax1.set_ylabel(col2)
    ax1.set_title(f'Scatter plot of {col1} vs {col2} for␣
↪Defaulters',fontsize=12)

    sns.scatterplot(data=df_0,x=col1,y=col2,ax=ax2)
    ax2.set_xlabel(col1)
    ax2.set_ylabel(col2)
    ax2.set_title(f'Scatter plot of {col1} vs {col2} for␣
↪Non-Defaulters',fontsize=12)
    plt.show()
```

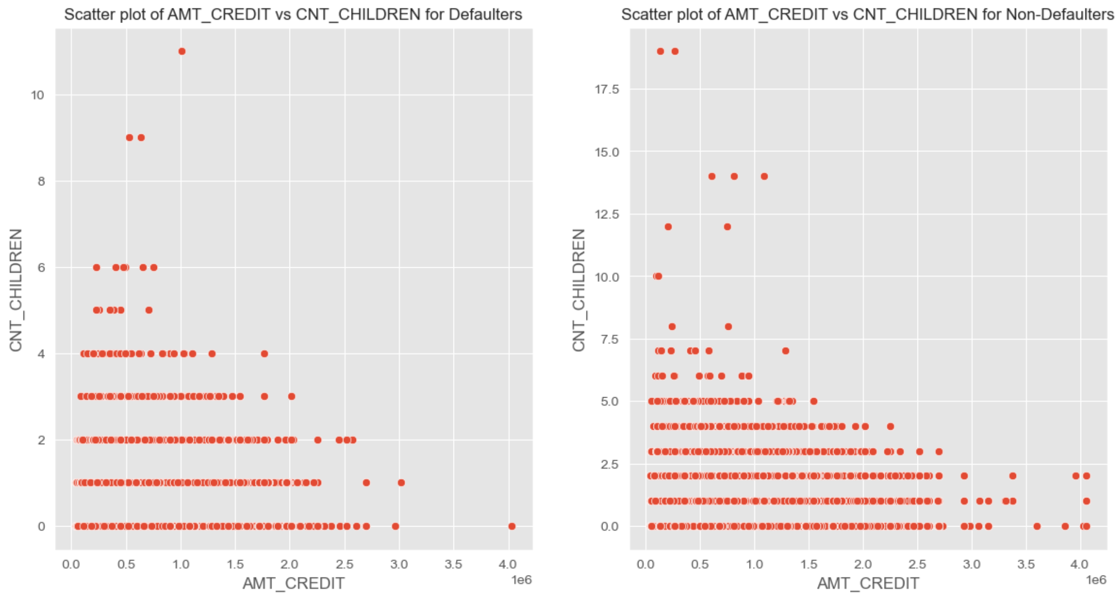
[152]:
```
bivarnum_plot('AMT_ANNUITY','AMT_CREDIT')
```



[153]:
```
bivarnum_plot('AMT_GOODS_PRICE','AMT_CREDIT')
```

The scatter plots reveal that persons who have not defaulted on their loans have a steeper slope than those who have had payment troubles. This means that for each unit rise in annuity and good price for which the loan is taken, the amount of credit taken by a Non-Defauter would grow higher than the amount of credit taken by a Defaulter.

```
[154]: bivarnum_plot('AMT_CREDIT','CNT_CHILDREN')
```

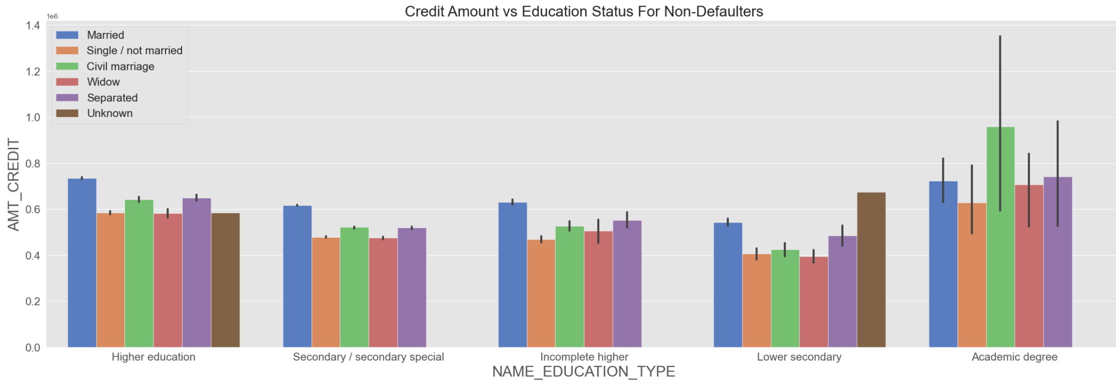


We can observe that the density in the bottom left corner of both cases is comparable, meaning individuals are equally likely to default if both the number of children and the AMT CREDIT are little. We may note that households with more children and bigger AMT CREDIT defaults occur less often.

```
[ ]:
```

Bivariate Analysis of Categorical and Numerical Columns¶

```
[155]: sns.
  ↪catplot(data=df_0,x='NAME_EDUCATION_TYPE',y='AMT_CREDIT',hue='NAME_FAMILY_STATUS',
            palette='muted', kind='bar',height=7,aspect=3,legend=False)
plt.title('Credit Amount vs Education Status For Non-Defaulters',fontsize=20)
plt.xlabel('NAME_EDUCATION_TYPE',fontsize=20)
plt.ylabel('AMT_CREDIT',fontsize=20)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.legend(loc='best',fontsize=15)
plt.show();
```



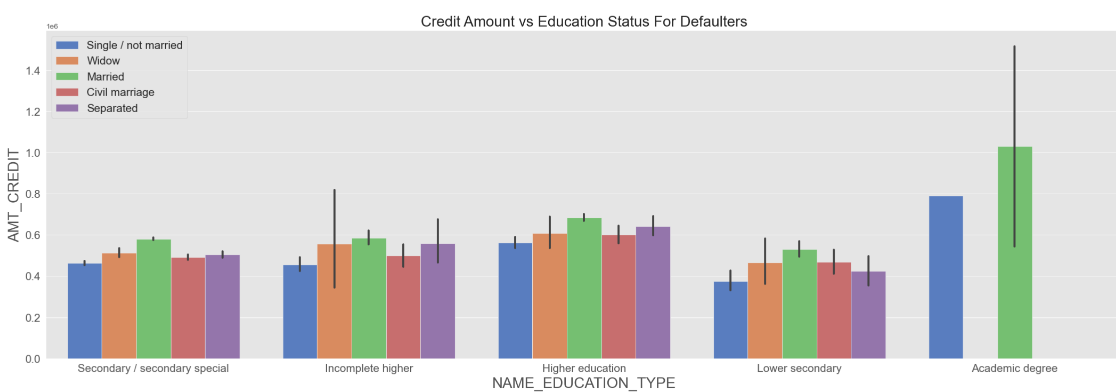
Conclusions to be drawn from the above graph for Non-Defaulters

Clients who are married are having greater amount of Credit amout to clear, except from those who did pursue lower secondary and academic degrees

Customers with a academic degree have bigger credit limits, with the Civil Marriage category being the highest.

Lower-educated consumers tend to have lower credit limits, with widows being the lowest.

[156]:
```
sns.
 ↪catplot(data=df_1,x='NAME_EDUCATION_TYPE',y='AMT_CREDIT',hue='NAME_FAMILY_STATUS',
           palette='muted', kind='bar',height=7,aspect=3,legend=False)
plt.title('Credit Amount vs Education Status For Defaulters',fontsize=20)
plt.xlabel('NAME_EDUCATION_TYPE',fontsize=20)
plt.ylabel('AMT_CREDIT',fontsize=20)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.legend(loc='best',fontsize=15)
plt.show();
```



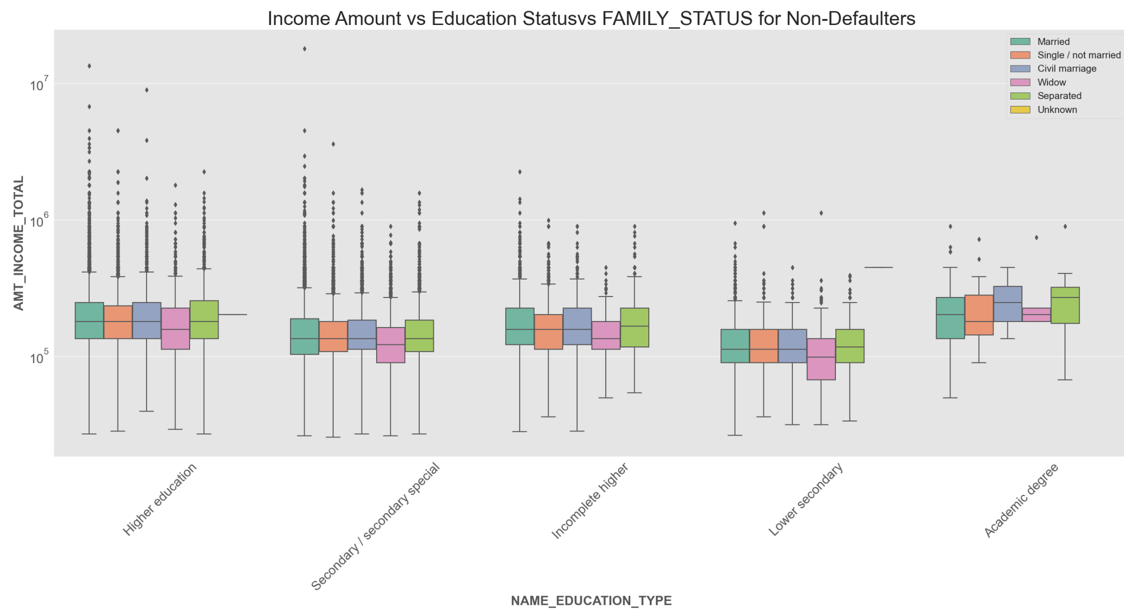
Conclusions to be drawn from the above graph for Defaulters

Customers with lower education have a lower average credit limit. Customers with an academic degree who are married have a greater credit limit and a higher default rate. Across all education segments, married customers have a higher credit amount. Single and Married are the only 2 family types present in academic degree

[ ]:

[157]:

```
plt.figure(figsize=(30,12))
plt.yscale('log')

sns.boxplot(data =df_0, x='NAME_EDUCATION_TYPE',y='AMT_INCOME_TOTAL',
            hue ='NAME_FAMILY_STATUS',orient='v',palette='Set2')
plt.title('Income Amount vs Education Statusvs FAMILY_STATUS for␣
 ↪Non-Defaulters',fontsize= 30)
plt.xlabel("NAME_EDUCATION_TYPE",fontsize= 20, fontweight="bold")
plt.ylabel("AMT_INCOME_TOTAL",fontsize= 20, fontweight="bold")
plt.xticks(rotation=45, fontsize=20)
plt.yticks(rotation=360, fontsize=20)
plt.legend( loc = 'best',fontsize=15)

plt.show();
```



The clients whose marital status is separated have the highest mean income compared to others.

Clients with a Higher Academic degree are having the greatest average salary.

Clients who are married have widely varying income statistics.

Lower secondary civil marriage family incomes are lower than those of others.

Clients with a Lower Secondary degree are having the lowest average salary.

Widow Clients with an Academic degree have a small number of outliers and lack the First and Third Quartiles. In addition, there are considerably fewer outliers among clients with academic degrees than among those with other levels of education.

[ ]:

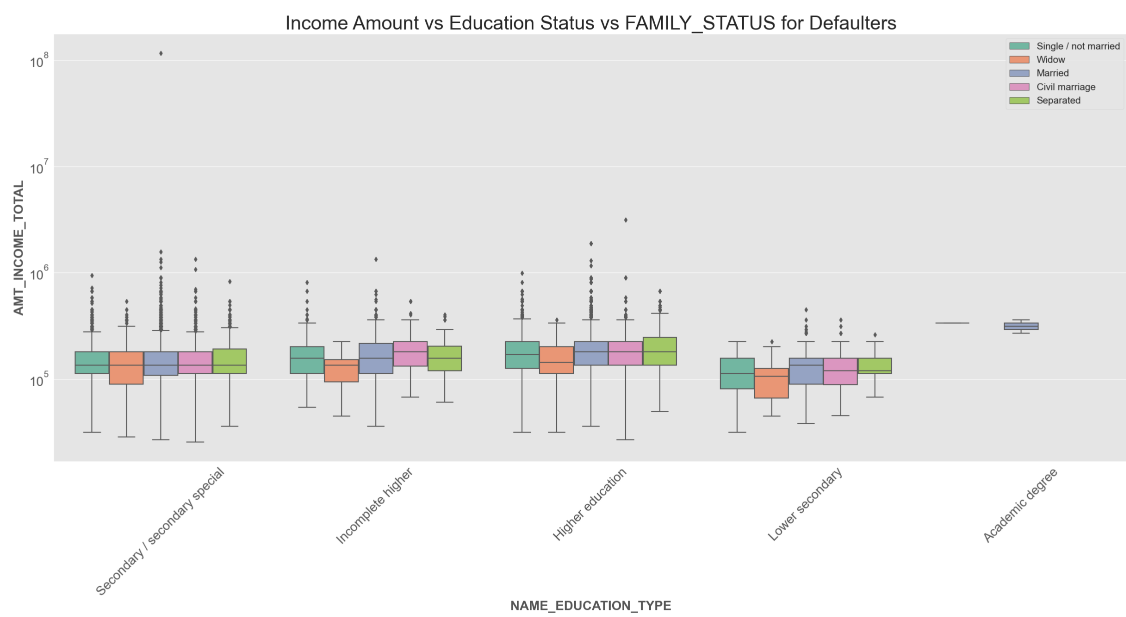
```

```

[ ]:

```

```

[158]:

```
plt.figure(figsize=(30,12))
plt.yscale('log')

sns.boxplot(data =df_1, x='NAME_EDUCATION_TYPE',y='AMT_INCOME_TOTAL',
            hue ='NAME_FAMILY_STATUS',orient='v',palette='Set2')
plt.title('Income Amount vs Education Status vs FAMILY_STATUS for␣
 ↪Defaulters',fontsize= 30)
plt.xlabel("NAME_EDUCATION_TYPE",fontsize= 20, fontweight="bold")
plt.ylabel("AMT_INCOME_TOTAL",fontsize= 20, fontweight="bold")
plt.xticks(rotation=45, fontsize=20)
plt.yticks(rotation=360, fontsize=20)
plt.legend( loc = 'best',fontsize=15)

plt.show();
```



Similar to Target0, based on the above boxplot for Education type 'Higher education', the income amount is the same regardless of family status

Clients who default on their loans have comparatively lower income than Non-defaulters.

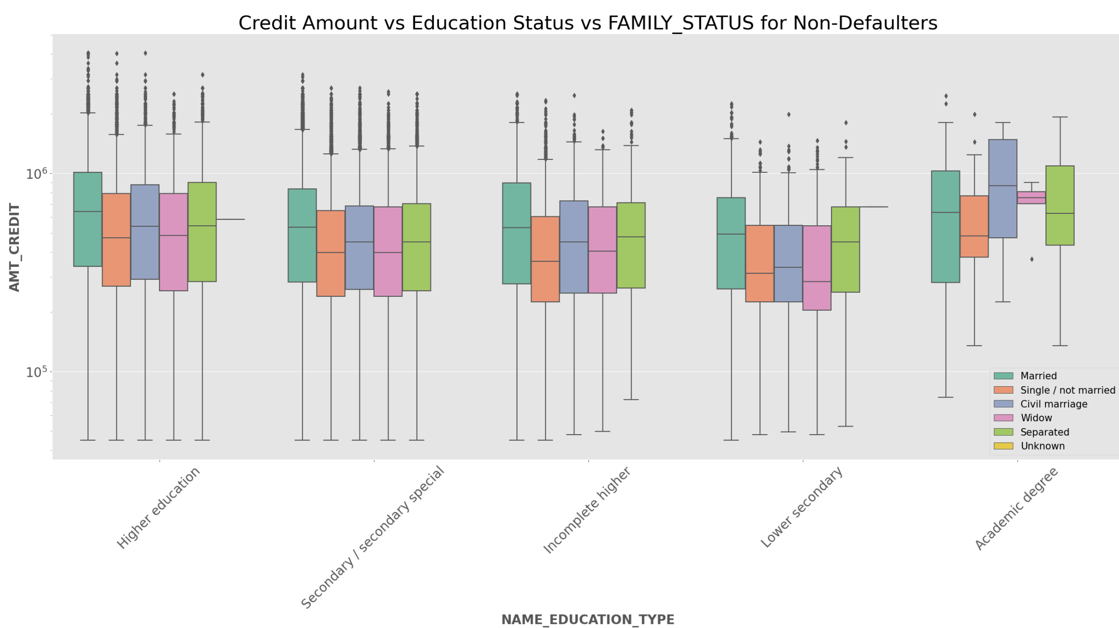
Fewer outliers for those who own an Academic degree, yet their salary is rather more than those with a Higher education.

[ ]:

```

```

[115]:

```
plt.figure(figsize=(30,12))
plt.yscale('log')

sns.boxplot(data =df_0, x='NAME_EDUCATION_TYPE',y='AMT_CREDIT',
            hue ='NAME_FAMILY_STATUS',orient='v',palette='Set2')
plt.title('Credit Amount vs Education Status vs FAMILY_STATUS for␣
 ↪Non-Defaulters',fontsize= 30)
plt.xlabel("NAME_EDUCATION_TYPE",fontsize= 20, fontweight="bold")
plt.ylabel("AMT_CREDIT",fontsize= 20, fontweight="bold")
plt.xticks(rotation=45, fontsize=20)
plt.yticks(rotation=360, fontsize=20)
plt.legend( loc = 'best',fontsize=15)

plt.show();
```



There are more outliers among clients with a higher degree and family statuses of marriage,single and civil marriage while clients who are having Academic degree are having fewer outliers.

Clients who are married tend to take bigger higher credit loans.

Widows and clients with an academic degree prefer to take out higher credit loans.

Clients who completed their Higher education tend to take greater credit loans.

[ ]:

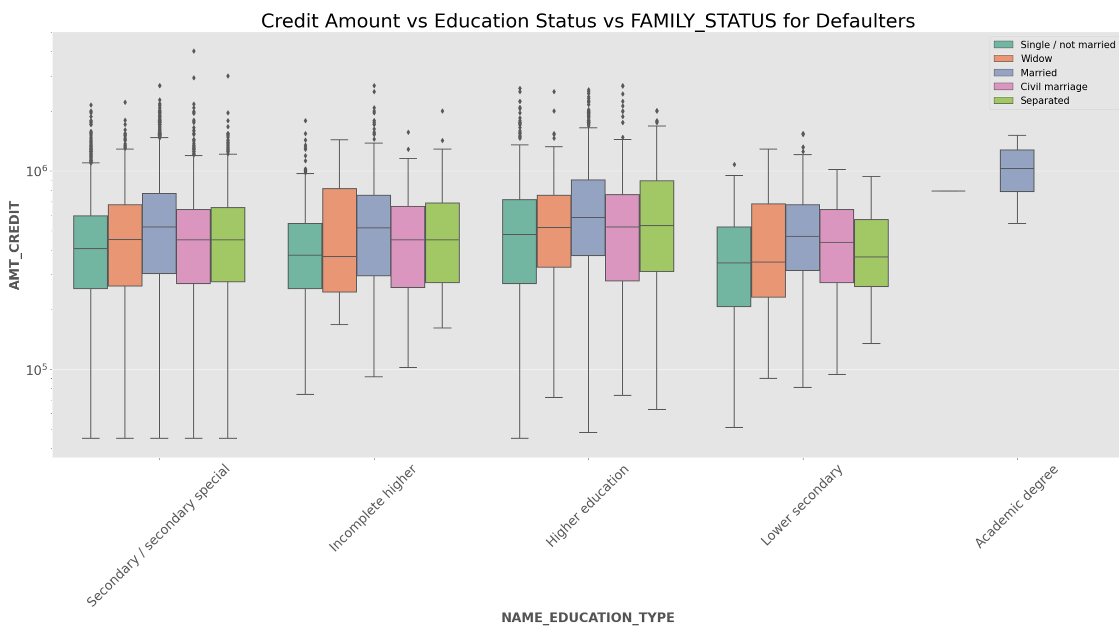
```

```

[ ]:

```

```

[116]:

```
plt.figure(figsize=(30,12))
plt.yscale('log')

sns.boxplot(data =df_1, x='NAME_EDUCATION_TYPE',y='AMT_CREDIT',
            hue ='NAME_FAMILY_STATUS',orient='v',palette='Set2')
plt.title('Credit Amount vs Education Status vs FAMILY_STATUS for␣
 ↪Defaulters',fontsize= 30)
plt.xlabel("NAME_EDUCATION_TYPE",fontsize= 20, fontweight="bold")
plt.ylabel("AMT_CREDIT",fontsize= 20, fontweight="bold")
plt.xticks(rotation=45, fontsize=20)
plt.yticks(rotation=360, fontsize=20)
plt.legend( loc = 'best',fontsize=15)


plt.show();
```



The majority of outliers belong to the Education types Higher education and Secondary.

Clients who are having Academic degree and involved in a civil marriage are taking greater amount of credit loan

According to the boxplot, customers who are married have the highest mean credit loan amount.

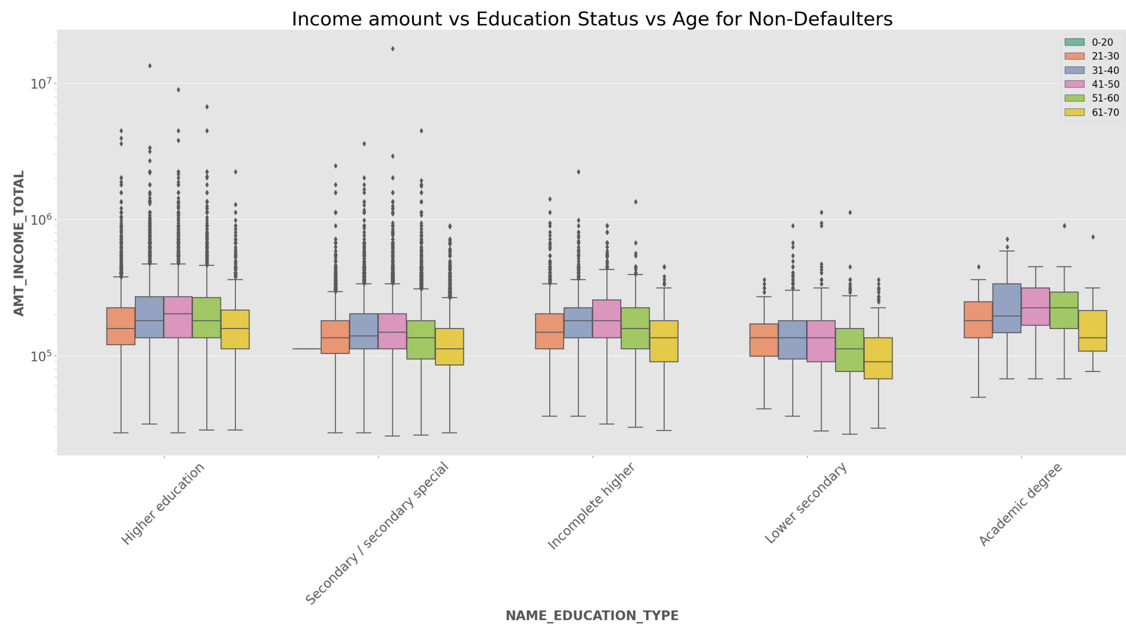
Clients who are married with academic degree applied for a larger credit loan. And is free of outliers and Single clients with academic degrees have a very slim boxplot with no outliers

Clients who completed their higher educaion tend to have the habit of taking greater amount of credit loans.

[ ]:

[ ]:

[117]:
```
plt.figure(figsize=(30,12))
plt.yscale('log')

sns.boxplot(data =df_0, x='NAME_EDUCATION_TYPE',y='AMT_INCOME_TOTAL',
            hue ='DAYS_BIRTH_CAT',orient='v',palette='Set2')
plt.title('Income amount vs Education Status vs Age for␣
 ↪Non-Defaulters',fontsize= 30)
plt.xlabel("NAME_EDUCATION_TYPE",fontsize= 20, fontweight="bold")
plt.ylabel("AMT_INCOME_TOTAL",fontsize= 20, fontweight="bold")
plt.xticks(rotation=45, fontsize=20)
plt.yticks(rotation=360, fontsize=20)
plt.legend( loc = 'best',fontsize=15)

plt.show();
```



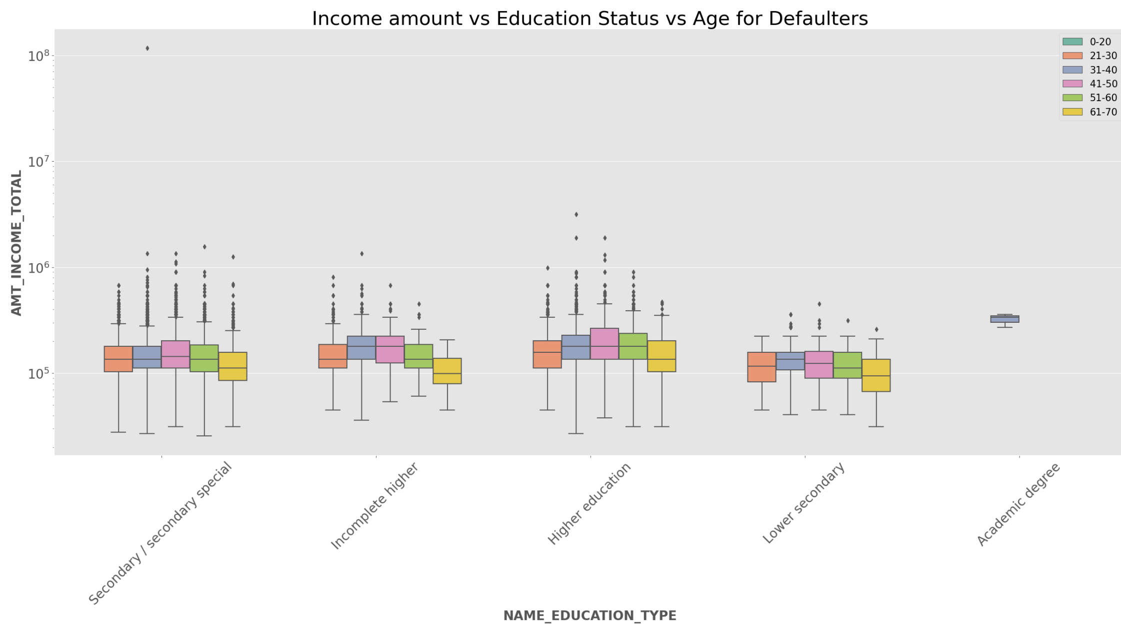
Clients between the ages of 41-50 seem to be having the highest mean of income compared to others.

Clients between the ages of 61-70 seem to be having the lowest mean of income compared to others.

[ ]:

[118]:

```
plt.figure(figsize=(30,12))
plt.yscale('log')

sns.boxplot(data =df_1, x='NAME_EDUCATION_TYPE',y='AMT_INCOME_TOTAL',
            hue ='DAYS_BIRTH_CAT',orient='v',palette='Set2')
plt.title('Income amount vs Education Status vs Age for Defaulters',fontsize=␣
↪30)
plt.xlabel("NAME_EDUCATION_TYPE",fontsize= 20, fontweight="bold")
plt.ylabel("AMT_INCOME_TOTAL",fontsize= 20, fontweight="bold")
plt.xticks(rotation=45, fontsize=20)
plt.yticks(rotation=360, fontsize=20)
plt.legend( loc = 'best',fontsize=15)
plt.show();
```



Clients who fail on their loans often have a lower income than those who have non-payment issues.

For Defaulters, Clients who are having Academic degree and ages between 31-40 are having the highestmean of income

[119]:

```
pd.pivot_table(data=df,index=['CODE_GENDER','AMT_INCOME_TOTAL_CAT'],
               columns=['NAME_EDUCATION_TYPE'],values='TARGET',aggfunc=np.mean)
```

[119]:

```
NAME_EDUCATION_TYPE             Academic degree  Higher education  Incomplete
higher  Lower secondary  Secondary / secondary special
CODE_GENDER AMT_INCOME_TOTAL_CAT
F           VeryLow                    0.000000          0.056068
0.086399         0.080193                       0.076778
```

```
            Low                            0.000000        0.049022
0.080075         0.113889                          0.079523
            Medium                         0.000000        0.050254
0.078431         0.096983                          0.075692
            High                           0.105263        0.041516
0.074313         0.038961                          0.070736
            VeryHigh                       0.076923        0.037289
0.082251         0.066667                          0.065930
M           VeryLow                        0.000000        0.080411
0.123967         0.125000                          0.118066
            Low                            0.000000        0.073305
0.097778         0.142857                          0.123693
            Medium                         0.000000        0.070086
0.095130         0.150515                          0.113466
            High                           0.000000        0.055911
0.074627         0.081633                          0.093484
            VeryHigh                       0.000000        0.044080
0.077586         0.064516                          0.089939
```

Male Clients with Lower Secondary Education who earn a very low to moderate income have a high chance of default.

Male Clients with Secondary Education who earn a very low to moderate income have a high chance of default.

Customers who are male, have an incomplete education, and earn very low salaries are at a significant risk of default.

Male Clients with a Academic degree do not fail on their loans. Women with an academic degree and a high income have a greater likelihood of defaulting on their loans.

[ ]:

```

```

0.6.9 10. Correlation

[120]:

```python
num_cols = df.select_dtypes('number')
num_cols.columns
```

[120]:

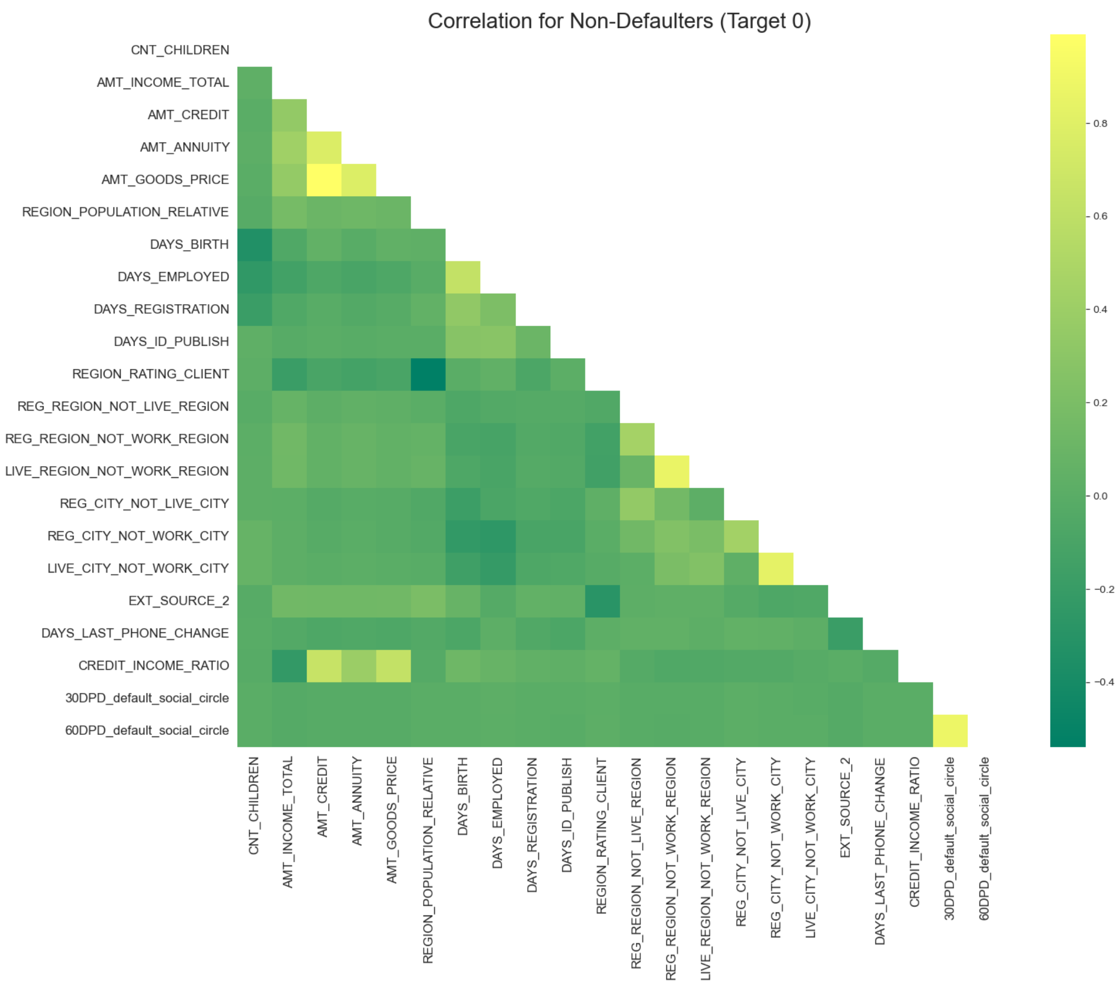
```
Index(['SK_ID_CURR', 'TARGET', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT',
'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'REGION_POPULATION_RELATIVE', 'DAYS_BIRTH',
'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH', 'FLAG_MOBIL',
'FLAG_EMAIL', 'REGION_RATING_CLIENT', 'REG_REGION_NOT_LIVE_REGION',
'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',
'EXT_SOURCE_2', 'DAYS_LAST_PHONE_CHANGE', 'CREDIT_INCOME_RATIO',
'30DPD_default_social_circle', '60DPD_default_social_circle'], dtype='object')
```

[121]:
```
corr1 = df_1.iloc[0:,2:]
corr0 = df_0.iloc[0:,2:]
corr1.drop(columns=['FLAG_MOBIL','FLAG_EMAIL'],inplace=True)
corr0.drop(columns=['FLAG_MOBIL','FLAG_EMAIL'],inplace=True)
```

[122]:
```
corr1 = corr1.select_dtypes(include = 'number')
```

[123]:
```
corr0 = corr0.select_dtypes(include = 'number')
```

[124]:
```
# Create mask for upper triangle
mask = np.zeros_like(corr0.corr())
triangle_indices = np.triu_indices_from(mask)
mask[triangle_indices] = True

plt.figure(figsize=(16, 12))
sns.set_style('white')

# Adjust the font size and annotation settings
sns.heatmap(corr0.corr().round(2), mask=mask, annot=True, fmt=".2f",␣
↪cmap='summer', annot_kws={'size':25})
plt.title('Correlation for Non-Defaulters (Target 0)', fontsize=20)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

plt.show();
```



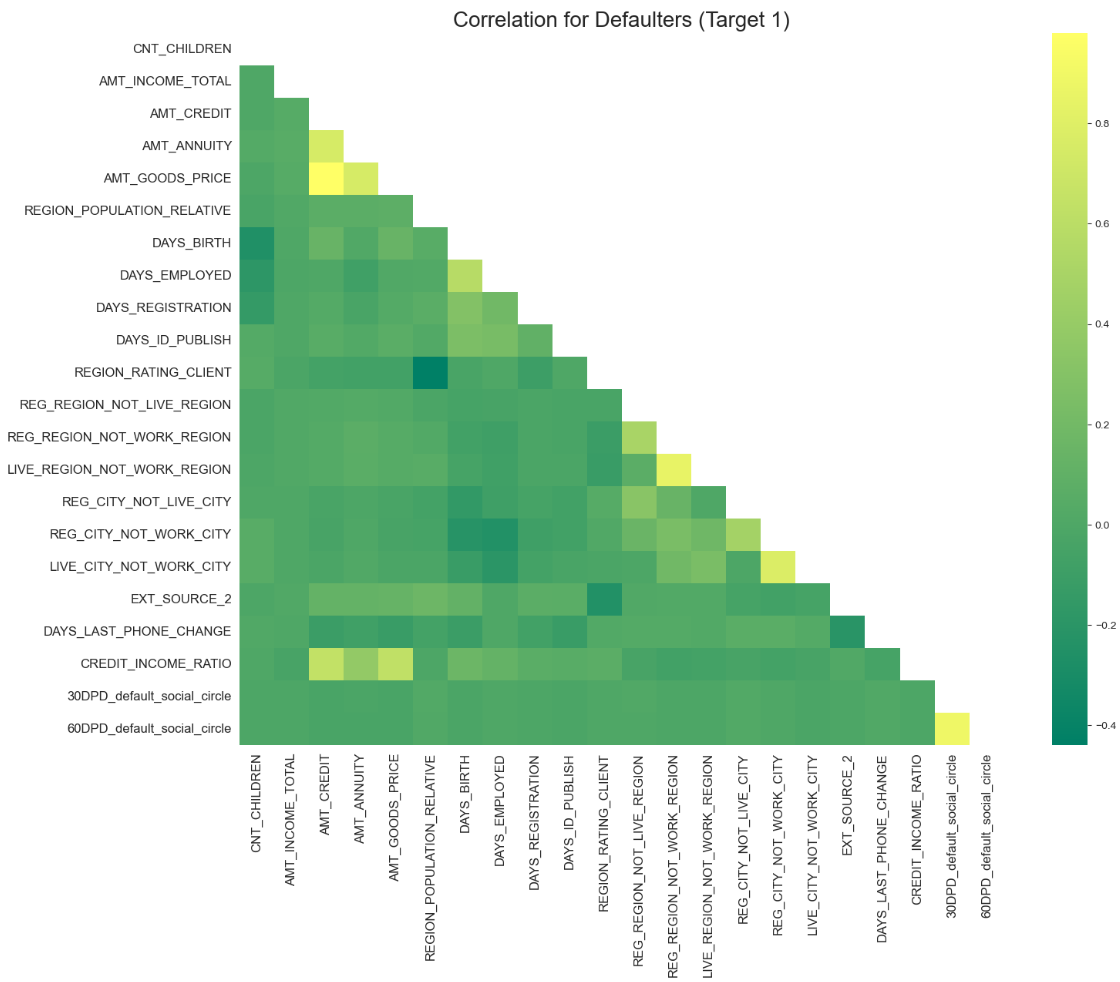
Observation:

- AMT_CREDIT, AMT_ANNUITY, AMT_INCOME_TOTAL and AMT_GOODS_PRICE are strongly related.
- CREDIT_INCOME_RATIO increases with increase in AMT_CREDIT and AMT_GOODS_PRICE, decreases with increase in AMT_INCOME_TOTAL.
- Dense population(REGION_POPULATION_RELATIVE) indicates a quality grade(REGION_RATING_CLIENT)
- Those residing in prime locations(REGION_RATING_CLIENT) earn higher salaries(AMT_INCOME_TOTAL) comparing to people who dosen't.
- Individuals who have defaulted on a 30-day payment are also likely to default on a 60-day payment.
- Elderly individuals have higher credit to income ratios.
- Credit amounts are greater in densely populated regions.

- Clients have fewer children in densely crowded areas.

[ ]:

```
[125]: # Create mask for upper triangle
mask = np.zeros_like(corr1.corr())
triangle_indices = np.triu_indices_from(mask)
mask[triangle_indices] = True

plt.figure(figsize=(16, 12))
sns.set_style('white')

# Adjust the font size and annotation settings
sns.heatmap(corr1.corr().round(2), mask=mask, annot=True, fmt=".2f",␣
 ↪cmap='summer', annot_kws={'size':15})
plt.title('Correlation for Defaulters (Target 1)', fontsize=20)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

plt.show();
```



Observation:

- "AMT_INCOME_TOTAL" has a very high positive correlation (close to 1) with itself, which is expected as it represents the perfect correlation of a variable with itself.
- There are several variables that have moderate to high positive correlations with "AMT_INCOME_TOTAL", such as "AMT_CREDIT", "AMT_ANNUITY", and
- "AMT_GOODS_PRICE". This suggests that these variables are positively associated with the target variable, meaning higher values of these variables tend to correspond with higher income levels.
- Variables like "REGION_POPULATION_RELATIVE", "DAYS_BIRTH", and "DAYS_EMPLOYED" have low or near-zero correlations with "AMT_INCOME_TOTAL", indicating that they have little or no linear relationship with the target variable.
- Some variables, such as "REG_CITY_NOT_LIVE_CITY" and "LIVE_CITY_NOT_WORK_CITY", have moderate negative correlations with "AMT_INCOME_TOTAL", suggesting that higher values of these variables are associated with lower income levels.
- The correlation matrix appears to be symmetric about the diagonal, which is expected since the correlation between variable A and variable B should be the same as the correlation between variable B and variable A.

0.6.10 Recommendations:

Bank should give focus on providing cash loans rather than revolving loans, as cash loans are less likely to default.

Female borrowers have a lower default rate. So, the bank should give a slight priority to female applicants.

Clients who do not have any accompanying applicants should be the focus group.

The safest segments of employment for lending are workers, commercial associates, and pensioners.

Clients with higher education degrees should be given more loans.

Married clients are safer than unmarried clients.

Homeowners and tenants are more likely to seek loans and have a lower risk of defaulting compared to those living with their parents. The bank should prioritize loan applications from individuals who own or rent housing.

People having a house or apartment are safer candidates for providing loans.

Low-skill laborers and drivers should be given less priority, as they have a higher probability of making defaults.

People with income less than 1 million and taking loans close to 1 million have a higher chance of defaults, so they should not be the focus.

Married couples with fewer than five children are considered safe for providing loans.

Clients with an annuity of less than 100K are on the safer side for the bank.

Customers with higher education degrees, particularly those who are married, tend to take out larger loan amounts and have a lower risk of default. The bank could consider offering higher credit limits or loan amounts to this segment.

80-90% of customers who were previously canceled or refused are repayers. So, the bank has to reverify those applications.

Clients who previously had an unused loan offer should not be given new loans despite having high incomes, as these clients have the maximum chance of defaulting on loans.

Male borrowers with lower secondary or secondary education levels and very low to moderate incomes exhibit a higher chance of default. The bank should scrutinize such applications more carefully and potentially implement stricter lending criteria for this segment.

Borrowers in the 20-40 age group tend to have a higher risk of default, while those above 40 exhibit lower default rates. The bank should factor in age-related risk patterns when assessing loan applications and adjust lending policies accordingly.

The analysis revealed strong correlations between variables like AMT_CREDIT, AMT_ANNUITY, AMT_INCOME_TOTAL, and AMT_GOODS_PRICE. The bank could develop credit scoring models or risk assessment frameworks based on these correlated variables to better evaluate loan applications.