EDA Case Study

Business Understanding:

Understanding of risk analytics in banking and financial services and understand how data is used to minimise the risk of losing money while lending to customers.

The loan providing companies find it hard to give loans to the people due to their insufficient or non-existent credit history. Because of that, some consumers use it as their advantage by becoming a defaulter. Suppose you work for a consumer finance company which specialises in lending various types of loans to urban customers. You have to use EDA to analyse the patterns present in the data. This will ensure that the applicants are capable of repaying the loan are not rejected.

When the company receives a loan application, the company has to decide for loan approval based on the applicant's profile. Two types of risks are associated with the bank's decision:

If the applicant is likely to repay the loan, then not approving the loan results in a loss of business to the company

If the applicant is not likely to repay the loan, i.e. he/she is likely to default, then approving the loan may lead to a financial loss for the company.

When a client applies for a loan, there are four types of decisions that could be taken by the client/company):

1. **Approved:** The Company has approved loan Application
2. **Cancelled:** The client cancelled the application sometime during approval. Either the client changed her/his mind about the loan or in some cases due to a higher risk of the client he received worse pricing which he did not want.
3. **Refused:** The Company had rejected the loan (because the client does not meet their requirements etc.).
4. **Unused offer:** Loan has been cancelled by the client but on different stages of the process.

In this case study, will use EDA to understand how consumer attributes and loan attributes influence the tendency of default.

Business Objectives:

This case study aims to identify patterns which indicate if a client has difficulty paying their instalments which may be used for taking actions such as denying the loan, reducing the amount of loan, lending (too 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.

Problem Statements:

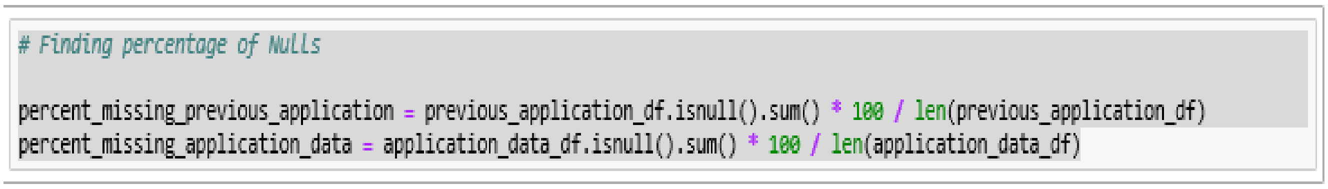
1. Identify the missing data and use appropriate method to deal with it. (Remove columns/or replace it with an appropriate value)

Hint: Note that in EDA, since it is not necessary to replace the missing value, but if you have to replace the missing value, what should be the approach. Clearly mention the approach.

Answer:

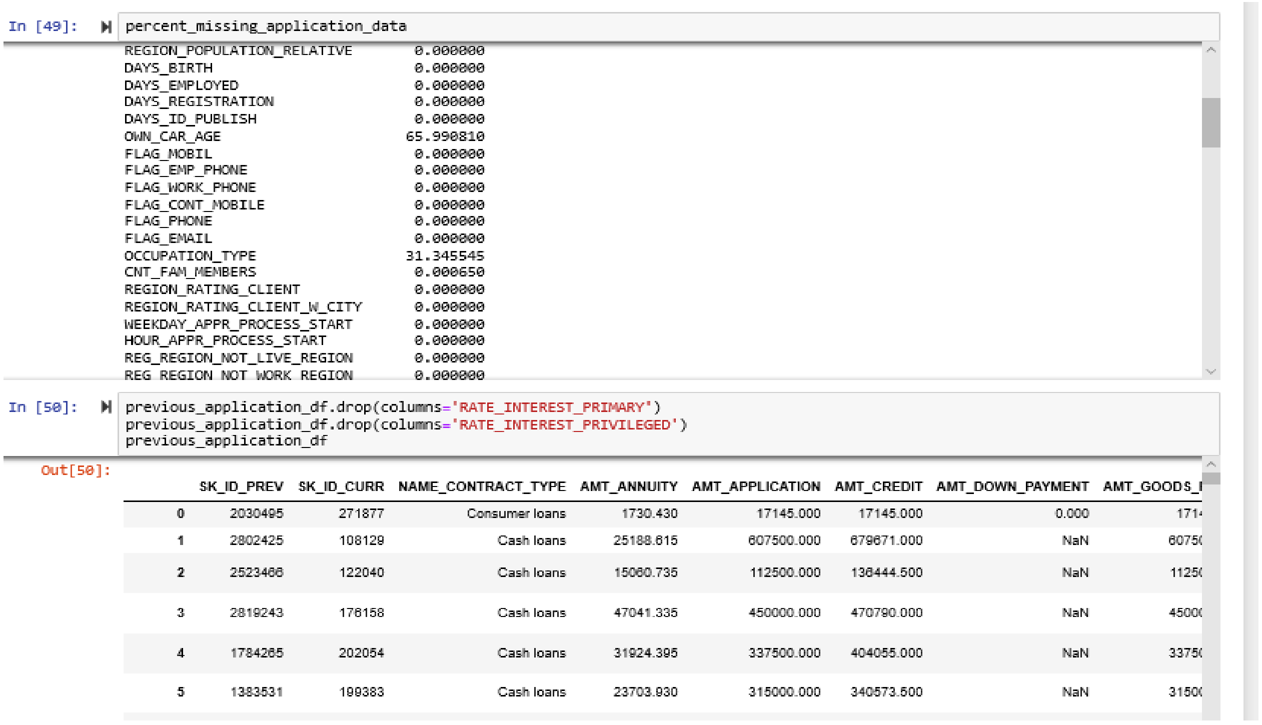
First we need to understand the data. If any null/Nan are found in the data, then find out the percentage of nulls for that column.

If the percentage is 99-100 we can drop the column. If the columns has more null percentage but necessary then we need to replace the values with mean/median.



```
# Finding percentage of Nulls

percent_missing_previous_application = previous_application_df.isnull().sum() * 100 / len(previous_application_df)
percent_missing_application_data = application_data_df.isnull().sum() * 100 / len(application_data_df)
```



```
In [49]:   percent_missing_application_data
REGION_POPULATION_RELATIVE       0.000000
DAYS_BIRTH                       0.000000
DAYS_EMPLOYED                    0.000000
DAYS_REGISTRATION                0.000000
DAYS_ID_PUBLISH                  0.000000
OWN_CAR_AGE                     65.990810
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                 31.345545
CNT_FAM_MEMBERS                  0.000650
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
```

```
In [50]:   previous_application_df.drop(columns='RATE_INTEREST_PRIMARY')
           previous_application_df.drop(columns='RATE_INTEREST_PRIVILEGED')
           previous_application_df
```

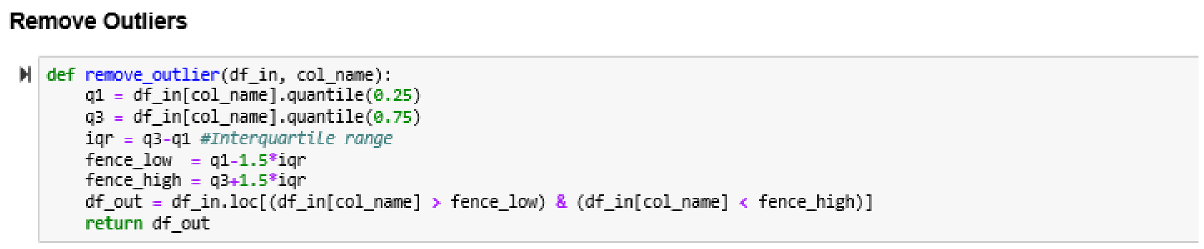
Out[50]:

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT	AMT_GOODS_
0	2030495	271877	Consumer loans	1730.430	17145.000	17145.000	0.000	171
1	2802425	108129	Cash loans	25188.615	607500.000	679671.000	NaN	6075
2	2523466	122040	Cash loans	15060.735	112500.000	136444.500	NaN	1125
3	2819243	176158	Cash loans	47041.335	450000.000	470790.000	NaN	4500
4	1784265	202054	Cash loans	31924.395	337500.000	404055.000	NaN	3375
5	1383531	199383	Cash loans	23703.930	315000.000	340573.500	NaN	3150

2. Identify if there are outliers in the dataset. Also, mention why you think it is an outlier. Again, remember that for this exercise, it is not necessary to remove any data points.

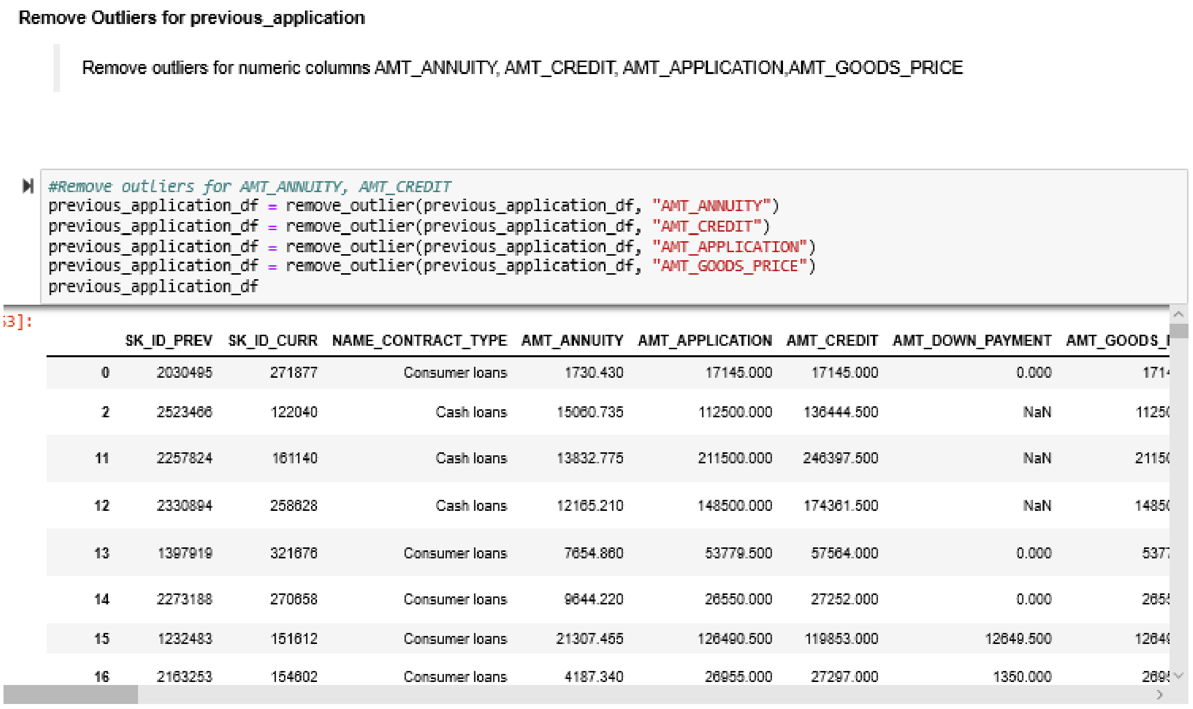
Answer:

First we need to identify the numerical columns, if the values are having too high and too low that means 25 percentile and 75-90 percentile.



Remove Outliers

```python
def remove_outlier(df_in, col_name):
    q1 = df_in[col_name].quantile(0.25)
    q3 = df_in[col_name].quantile(0.75)
    iqr = q3-q1 #Interquartile range
    fence_low  = q1-1.5*iqr
    fence_high = q3+1.5*iqr
    df_out = df_in.loc[(df_in[col_name] > fence_low) & (df_in[col_name] < fence_high)]
    return df_out
```



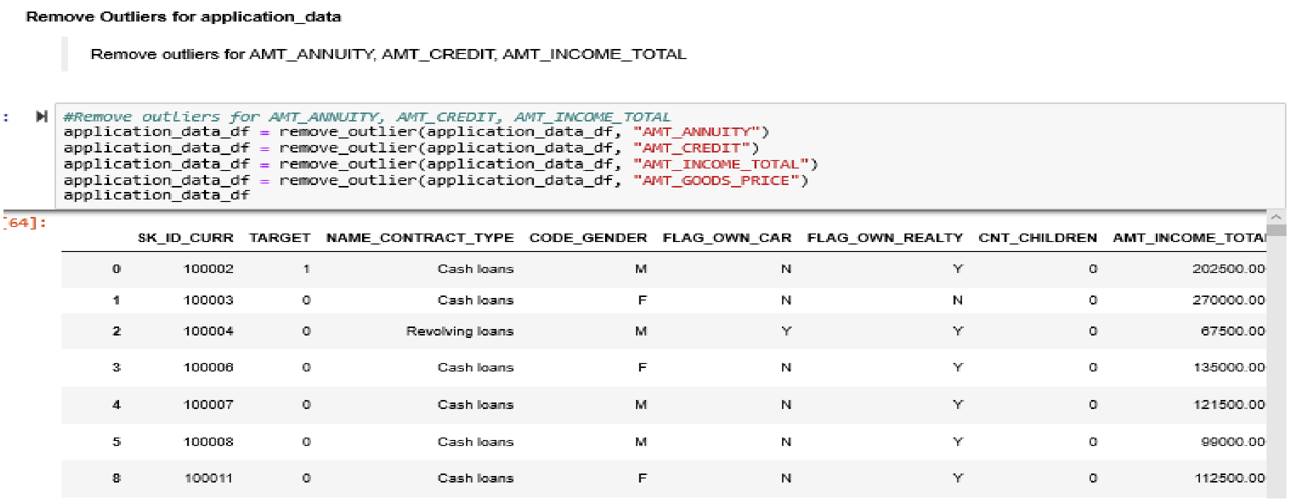
Remove Outliers for previous_application

> Remove outliers for numeric columns AMT_ANNUITY, AMT_CREDIT, AMT_APPLICATION,AMT_GOODS_PRICE

```python
#Remove outliers for AMT_ANNUITY, AMT_CREDIT
previous_application_df = remove_outlier(previous_application_df, "AMT_ANNUITY")
previous_application_df = remove_outlier(previous_application_df, "AMT_CREDIT")
previous_application_df = remove_outlier(previous_application_df, "AMT_APPLICATION")
previous_application_df = remove_outlier(previous_application_df, "AMT_GOODS_PRICE")
previous_application_df
```

3]:

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT	AMT_GOODS_I
0	2030495	271877	Consumer loans	1730.430	17145.000	17145.000	0.000	171
2	2523466	122040	Cash loans	15060.735	112500.000	136444.500	NaN	1125
11	2257824	161140	Cash loans	13832.775	211500.000	246397.500	NaN	2115
12	2330894	258628	Cash loans	12165.210	148500.000	174361.500	NaN	1485
13	1397919	321676	Consumer loans	7654.860	53779.500	57564.000	0.000	537
14	2273188	270658	Consumer loans	9644.220	26550.000	27252.000	0.000	265
15	1232483	151612	Consumer loans	21307.455	126490.500	119853.000	12649.500	1264
16	2163253	154602	Consumer loans	4187.340	26955.000	27297.000	1350.000	269



Remove Outliers for application_data

> Remove outliers for AMT_ANNUITY, AMT_CREDIT, AMT_INCOME_TOTAL

```python
#Remove outliers for AMT_ANNUITY, AMT_CREDIT, AMT_INCOME_TOTAL
application_data_df = remove_outlier(application_data_df, "AMT_ANNUITY")
application_data_df = remove_outlier(application_data_df, "AMT_CREDIT")
application_data_df = remove_outlier(application_data_df, "AMT_INCOME_TOTAL")
application_data_df = remove_outlier(application_data_df, "AMT_GOODS_PRICE")
application_data_df
```

64]:

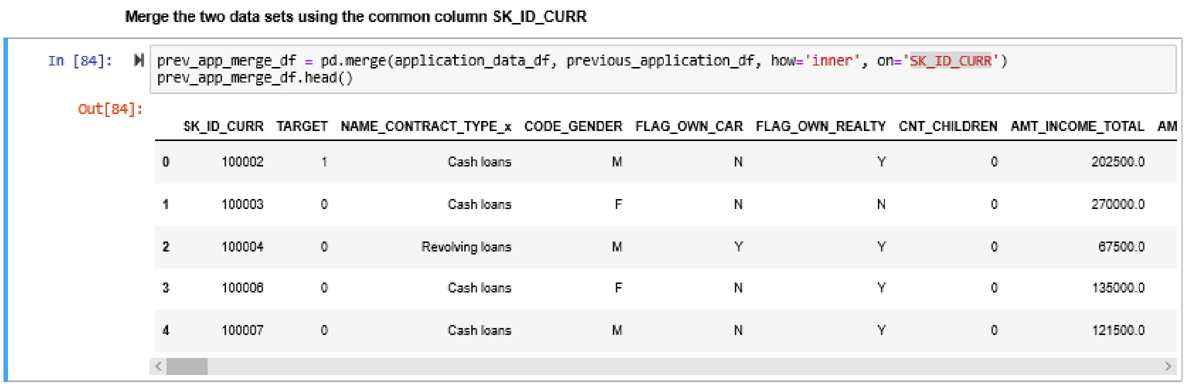
	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAI
0	100002	1	Cash loans	M	N	Y	0	202500.00
1	100003	0	Cash loans	F	N	N	0	270000.00
2	100004	0	Revolving loans	M	Y	Y	0	67500.00
3	100006	0	Cash loans	F	N	Y	0	135000.00
4	100007	0	Cash loans	M	N	Y	0	121500.00
5	100008	0	Cash loans	M	N	Y	0	99000.00
8	100011	0	Cash loans	F	N	Y	0	112500.00

3. Identify if there is data imbalance in the data. Find the ratio of data imbalance.

Hint: How will you analyse the data in case of data imbalance? You can plot more than one type of plot to analyse the different aspects due to data imbalance. For example, you can choose your own scale for the graphs, i.e. one can plot in terms of percentage or absolute value. Do this analysis for the **'Target variable'** in the dataset (**clients with payment difficulties** and **all other cases**). Use a mix of univariate and bivariate analysis etc.

Answer:

In previous_application there is no Target variable so merge the two datasets with the common column 'SK_ID_CURR'.



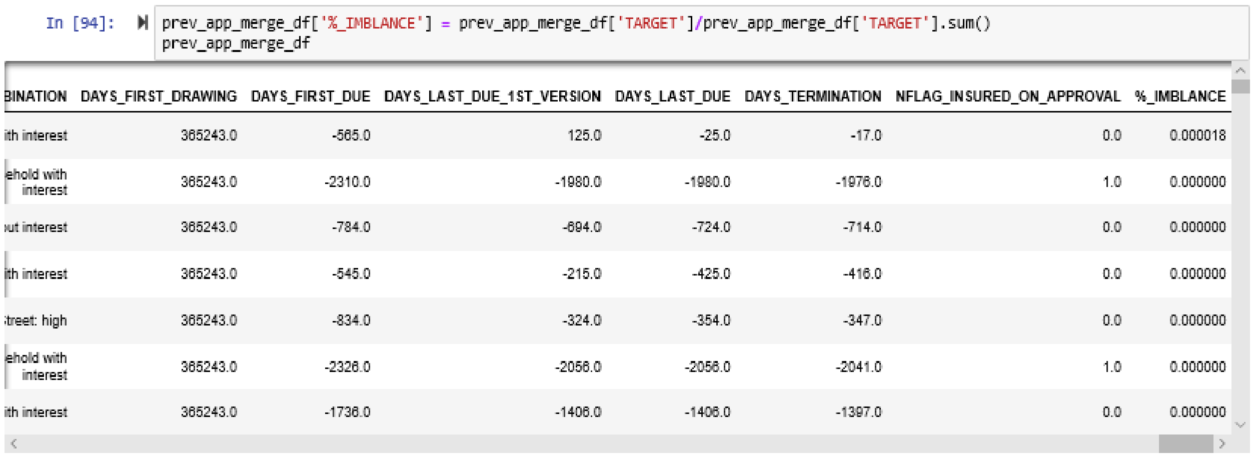
Merge the two data sets using the common column SK_ID_CURR

```
In [84]:  prev_app_merge_df = pd.merge(application_data_df, previous_application_df, how='inner', on='SK_ID_CURR')
          prev_app_merge_df.head()
```

Out[84]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AM
0	100002	1	Cash loans	M	N	Y	0	202500.0	
1	100003	0	Cash loans	F	N	N	0	270000.0	
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	
3	100006	0	Cash loans	F	N	Y	0	135000.0	
4	100007	0	Cash loans	M	N	Y	0	121500.0	

Now find the % of Imbalance



```
In [94]:  prev_app_merge_df['%_IMBLANCE'] = prev_app_merge_df['TARGET']/prev_app_merge_df['TARGET'].sum()
          prev_app_merge_df
```

BINATION	DAYS_FIRST_DRAWING	DAYS_FIRST_DUE	DAYS_LAST_DUE_1ST_VERSION	DAYS_LAST_DUE	DAYS_TERMINATION	NFLAG_INSURED_ON_APPROVAL	%_IMBLANCE
ith interest	365243.0	-565.0	125.0	-25.0	-17.0	0.0	0.000018
ehold with interest	365243.0	-2310.0	-1980.0	-1980.0	-1976.0	1.0	0.000000
ut interest	365243.0	-784.0	-694.0	-724.0	-714.0	0.0	0.000000
ith interest	365243.0	-545.0	-215.0	-425.0	-416.0	0.0	0.000000
treet: high	365243.0	-834.0	-324.0	-354.0	-347.0	0.0	0.000000
ehold with interest	365243.0	-2326.0	-2056.0	-2056.0	-2041.0	1.0	0.000000
ith interest	365243.0	-1736.0	-1406.0	-1406.0	-1397.0	0.0	0.000000

4. Explain the results of univariate, segmented univariate, bivariate analysis, etc. in business terms.

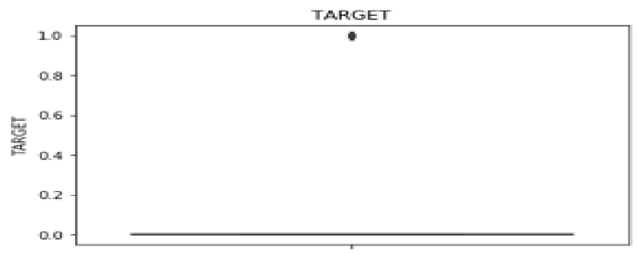
Answer:

Boxplots are a great way to visualise univariate data

univariate for target variable

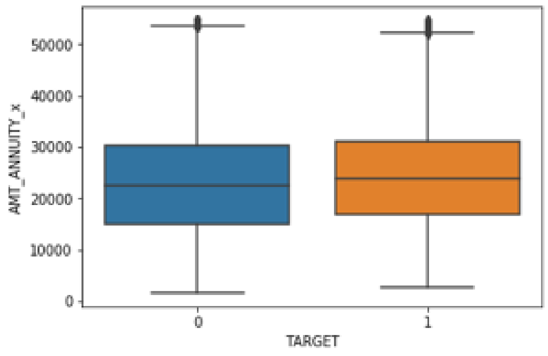
```
In [85]:  sns.boxplot(y=prev_app_merge_df['TARGET'])
          plt.title('TARGET')

          plt.show()
```

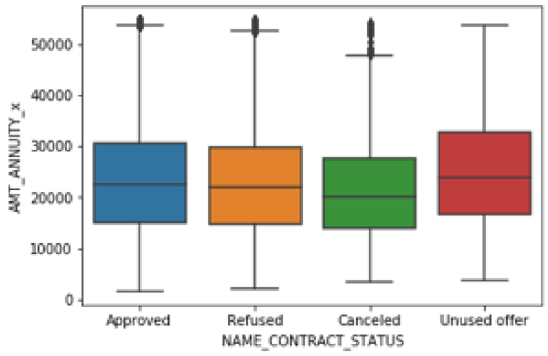


Bivariate Analysis

```
In [86]:  sns.boxplot(x='TARGET', y='AMT_ANNUITY_x', data=prev_app_merge_df)
          plt.show()
```



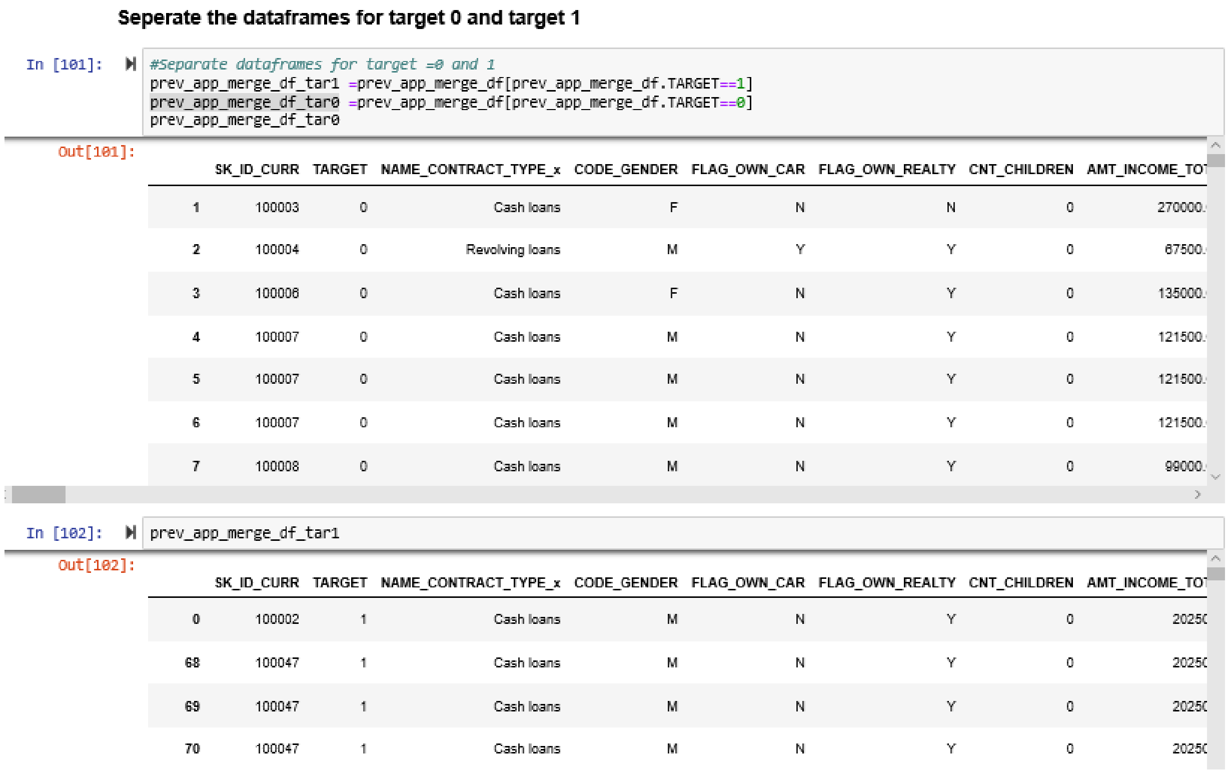
```
In [87]:  sns.boxplot(x='NAME_CONTRACT_STATUS', y='AMT_ANNUITY_x', data=prev_app_merge_df)
          plt.show()
```



5. Find the top 10 correlation for the **Client with payment difficulties** and **all other cases** (Target variable). Note that you have to find the top correlation by segmenting the data frame w.r.t to the target variable and then find the top correlation for each of the segmented data and find if any insight is there. Say, there are 5+1(target) variables in a dataset: **Var1, Var2, Var3, Var4, Var5, Target**. And if you have to find top 3 correlation, it can be: Var1 & Var2, Var2 & Var3, Var1 & Var3. Target variable will not feature in this correlation as it is a categorical variable and not a continuous variable which is increasing or decreasing.

Answer:

First segment the Target 0 and Target 1. Then find the correlation between top 10 numeric value columns.



Seperate the dataframes for target 0 and target 1

```
In [101]:
#Separate dataframes for target =0 and 1
prev_app_merge_df_tar1 =prev_app_merge_df[prev_app_merge_df.TARGET==1]
prev_app_merge_df_tar0 =prev_app_merge_df[prev_app_merge_df.TARGET==0]
prev_app_merge_df_tar0
```

Out[101]:

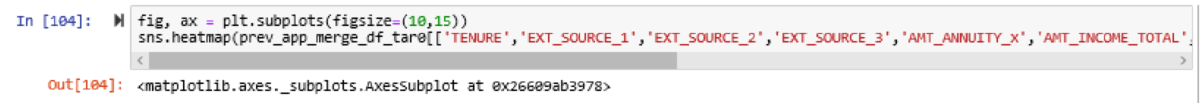
	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOT
1	100003	0	Cash loans	F	N	N	0	270000.
2	100004	0	Revolving loans	M	Y	Y	0	67500.
3	100006	0	Cash loans	F	N	Y	0	135000.
4	100007	0	Cash loans	M	N	Y	0	121500.
5	100007	0	Cash loans	M	N	Y	0	121500.
6	100007	0	Cash loans	M	N	Y	0	121500.
7	100008	0	Cash loans	M	N	Y	0	99000.

```
In [102]:
prev_app_merge_df_tar1
```

Out[102]:

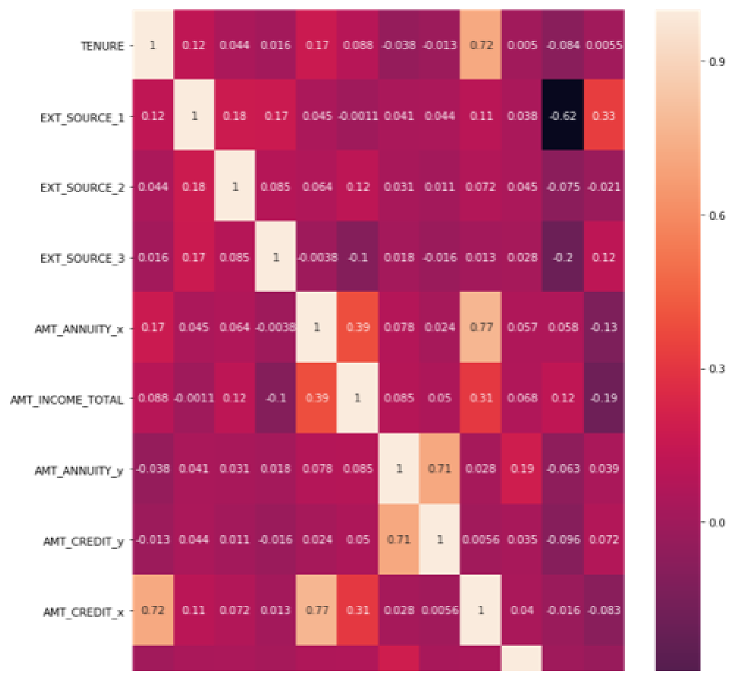
	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOT
0	100002	1	Cash loans	M	N	Y	0	2025C
68	100047	1	Cash loans	M	N	Y	0	2025C
69	100047	1	Cash loans	M	N	Y	0	2025C
70	100047	1	Cash loans	M	N	Y	0	2025C

Correlation for Target 0:



```
In [104]:
fig, ax = plt.subplots(figsize=(10,15))
sns.heatmap(prev_app_merge_df_tar0[['TENURE','EXT_SOURCE_1','EXT_SOURCE_2','EXT_SOURCE_3','AMT_ANNUITY_X','AMT_INCOME_TOTAL',
```

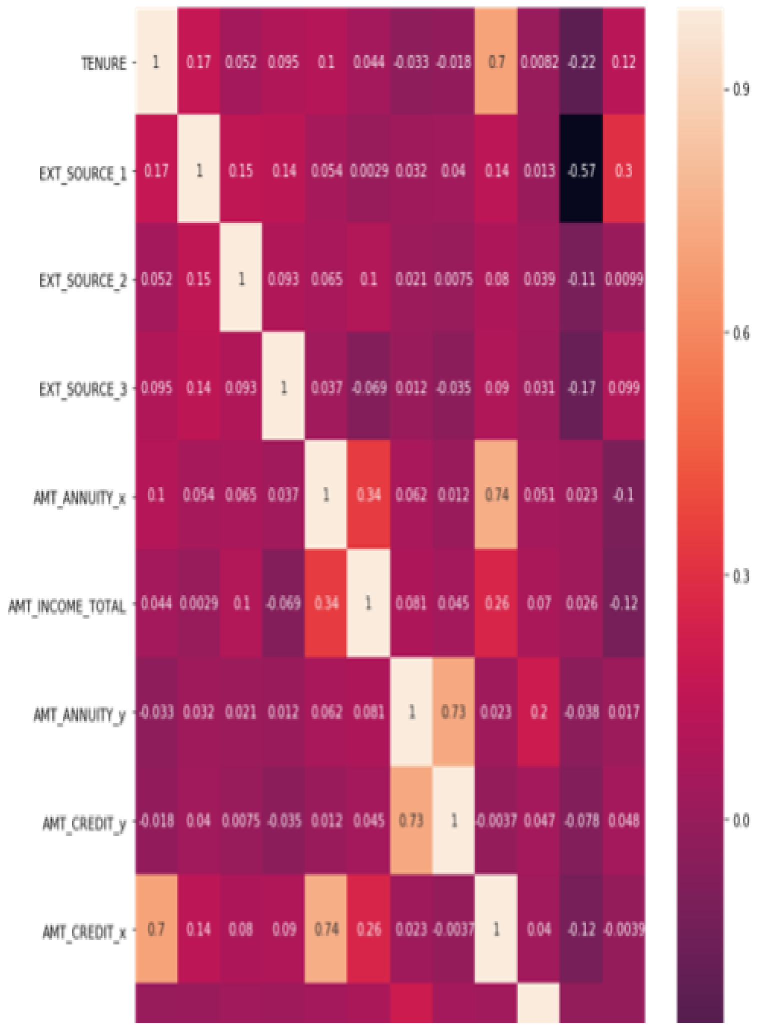
```
Out[104]: <matplotlib.axes._subplots.AxesSubplot at 0x26609ab3978>
```



Correlation for Target 1:

```
In [105]:  fig, ax = plt.subplots(figsize=(10,15))
           sns.heatmap(prev_app_merge_df_tar1[['TENURE','EXT_SOURCE_1','EXT_SOURCE_2','EXT_SOURCE_3','AMT_ANNUITY_x','AMT_INCOME_TOTAL',

Out[105]: <matplotlib.axes._subplots.AxesSubplot at 0x26607ee8b70>
```

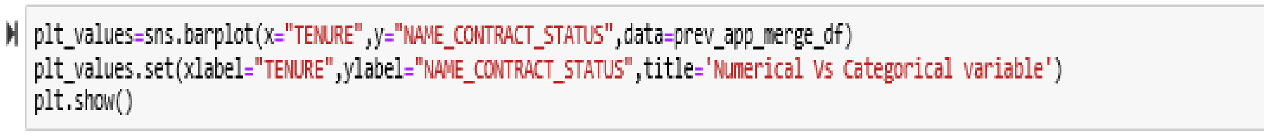


6. Include visualisations and summarise the most important results in the presentation. You are free to choose the graphs which explain the numerical/categorical variables. Insights should explain why the variable is important for differentiating the **clients with payment difficulties with all other cases.**

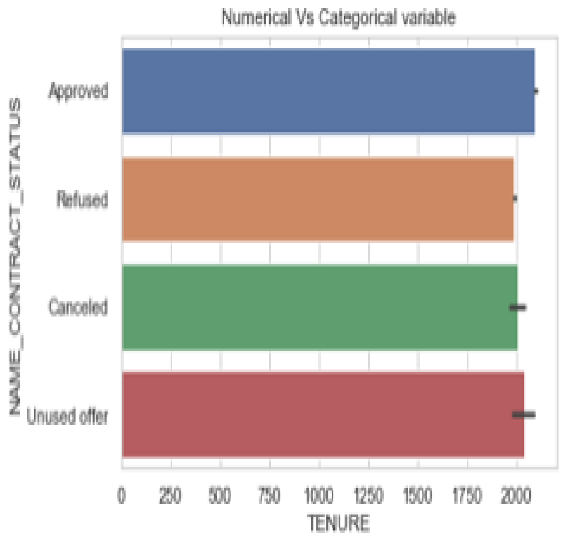
Answer:

We can plot Tenure and NAME_CONTRACT_STATUS (numerical vs categorical variables).

Best plot is to use Bar graph:



```
plt_values=sns.barplot(x="TENURE",y="NAME_CONTRACT_STATUS",data=prev_app_merge_df)
plt_values.set(xlabel="TENURE",ylabel="NAME_CONTRACT_STATUS",title='Numerical Vs Categorical variable')
plt.show()
```



From the above graph we can see clearly that how many clients got loan approved and how many got cancelled and how many are refused and remaining are unused offers.