GROUP6
PRESENTATION

HOME CREDIT DEFAUL RISK

KEY PEOPLE

NGUYEN NGOC DAT

25%

LE HUY PHONG

20%

NGUYEN MINH DUC

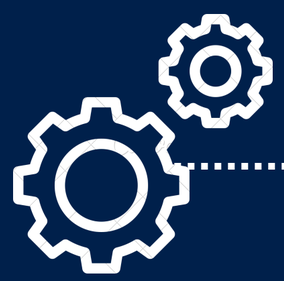
30%

LE QUOC MINH

25%

PROBLEM RESEARCH

Many people struggle to get loans due to insufficient or non-existent credit histories. And, unfortunately, this population is often taken advantage of by untrustworthy lenders.
Home Credit strives to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. In order to make sure this underserved population has a positive loan experience, Home Credit makes use of a variety of alternative data--including telco and transactional information--to predict their clients' repayment abilities



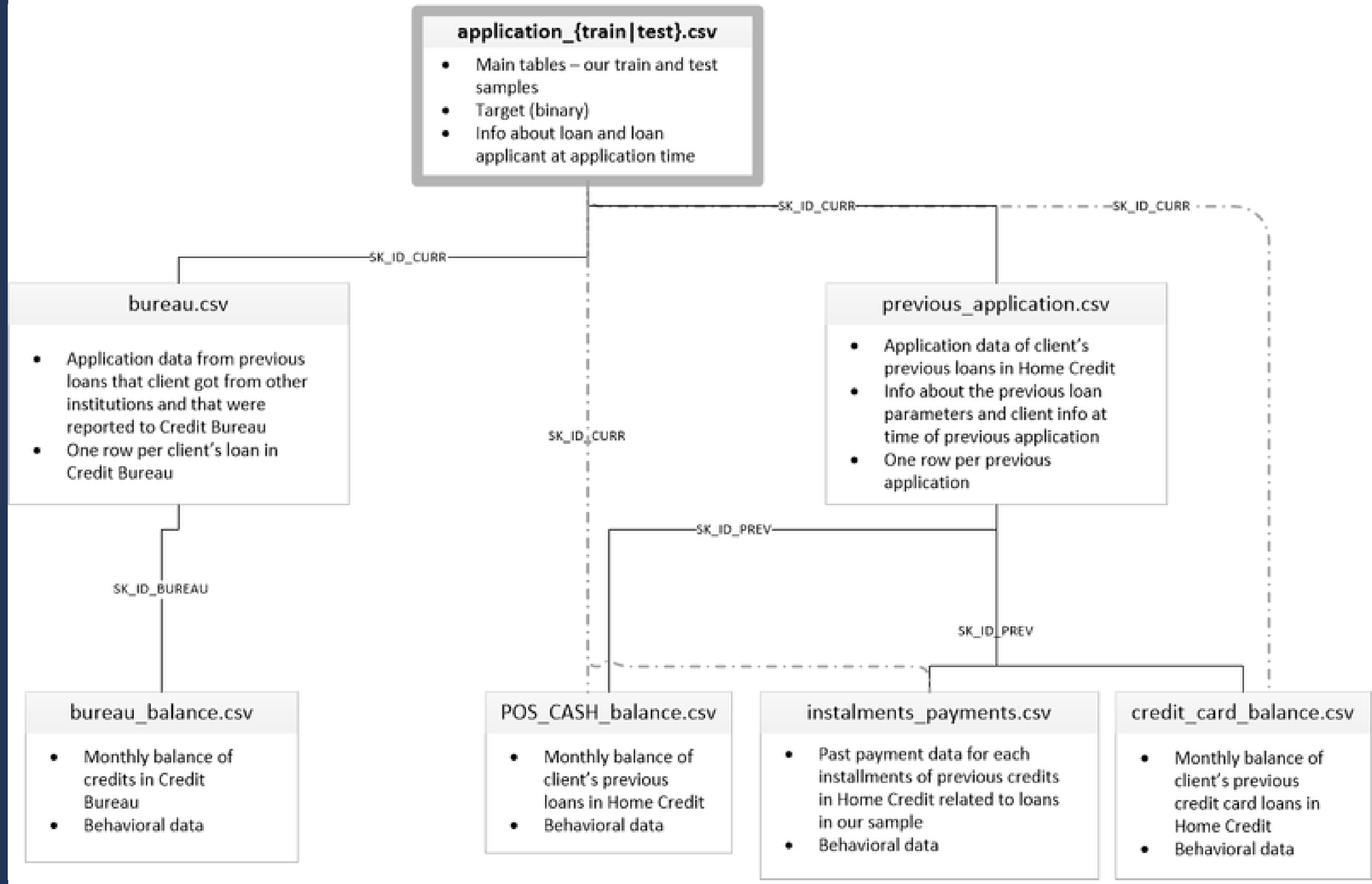
Step 1
EDA Handle Process

Step 2
EDA Of each data

Level 3
Feature selection
& Engineering

DATASET

- **The main dataset consists of 122 columns and 307511 rows.**
- **With 1 column target feature ["TARGET"]**



EDA HANDLE PROCESS

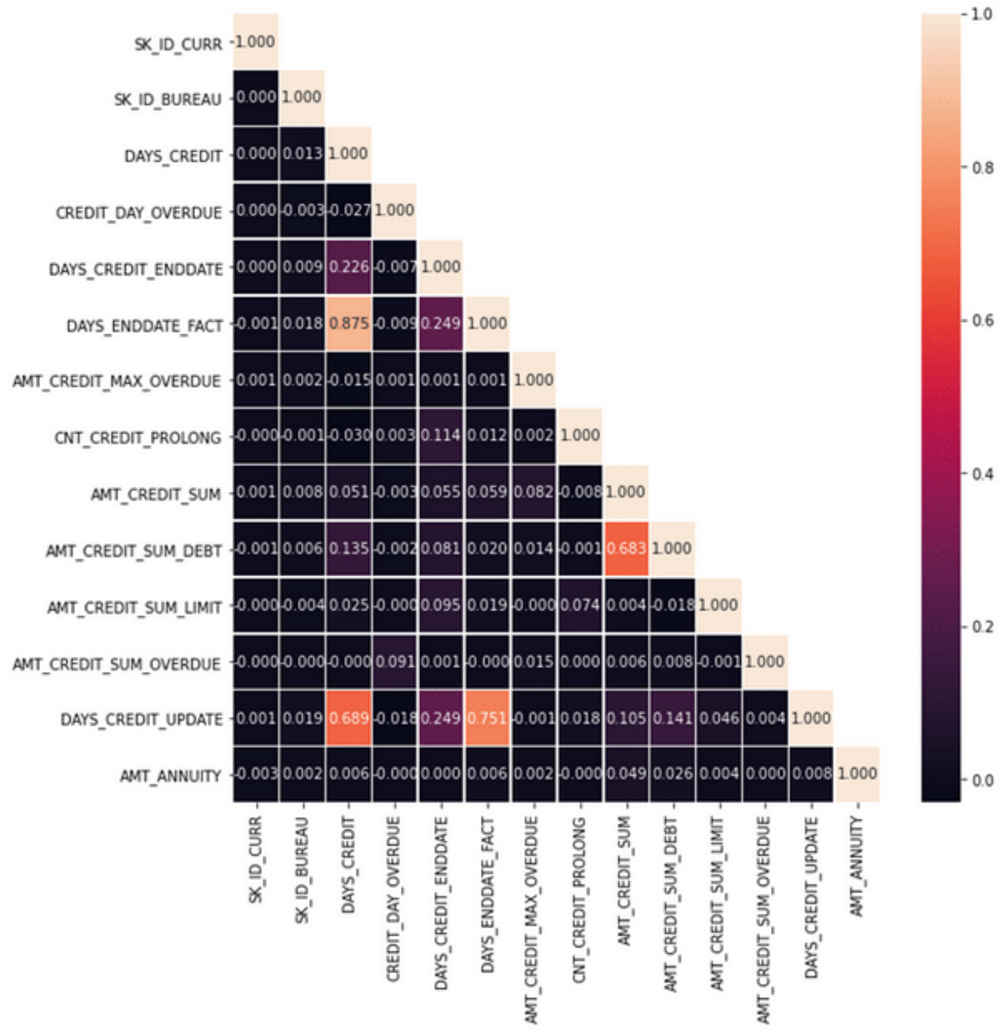
A. FIRST SIGHT OF OUR DATA

1. *df.head()*
2. *Understanding data*
3. *df.info()*
4. *df.duplicated.sum()*
5. *df.nunique().sum()*
6. *df.describe()*
7. *df.isnull.sum()*

B. CORRELATION HEATMAP

```
1  numeric_df = data._get_numeric_data()
2  numeric_df_corr = numeric_df.corr()
3  numeric_df_corr
4
5  plt.figure(figsize=(10,10))
6  mask = np.zeros_like(numeric_df_corr, dtype=np.bool)
7  mask[np.triu_indices_from(mask)] = True
8  mask[np.diag_indices_from(mask)] = False
9  sns.heatmap(numeric_df_corr,mask= mask, annot= True, fmt=".3f", linewidth=.5)
```

```
<AxesSubplot:>
```



Example from data bureau.csv

C. CLEAN SOME UNMEANING FEATURE

```
data['DAYS_CREDIT_ENDDATE'][data['DAYS_CREDIT_ENDDATE'] > -50*365] = np.nan
data['DAYS_ENDDATE_FACT'][data['DAYS_ENDDATE_FACT'] > -50*365] = np.nan
data['DAYS_CREDIT_UPDATE'][data['DAYS_CREDIT_UPDATE'] > -50*365] = np.nan
```

D. HANDLING MISSING DATA

- Drop column with the percentage of null > 90%
- Drop column with high percentage and dont have much meaning for EDA purpose
- Fill NaN

```
1  def missing_data(data):
2      total = data.isnull().sum().sort_values(ascending = False)
3      percent = (data.isnull().sum()/data.isnull().count()*100).sort_values(ascending = False)
4      return pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
```

```
1  missing_data(data)
```

	Total	Percent
DAYS_ENDDATE_FACT	1716427	99.999942
DAYS_CREDIT_UPDATE	1716333	99.994465
DAYS_CREDIT_ENDDATE	1716294	99.992193
AMT_ANNUITY	1226791	71.473490
AMT_CREDIT_MAX_OVERDUE	1124488	65.513264
AMT_CREDIT_SUM_LIMIT	591780	34.477415
AMT_CREDIT_SUM_DEBT	257669	15.011932
AMT_CREDIT_SUM	13	0.000757
CREDIT_DAY_OVERDUE	0	0.000000
DAYS_CREDIT	0	0.000000
CREDIT_CURRENCY	0	0.000000
SK_ID_BUREAU	0	0.000000
CNT_CREDIT_PROLONG	0	0.000000
AMT_CREDIT_SUM_OVERDUE	0	0.000000
CREDIT_TYPE	0	0.000000
CREDIT_ACTIVE	0	0.000000
SK_ID_CURR	0	0.000000

E. EDA

1. Merge with TARGET column in Application_train.csv to get more insights
2. Categorical feature
3. Numeric features

```python
def plot_categorical_variables_bar(data, column_name, figsize = (18,6),
                                   percentage_display = True,
                                   plot_defaulter = True, rotation = 0,
                                   horizontal_adjust = 0, fontsize_percent = 'xx-small'):

    print(f"Total Number of unique categories of {column_name} = {len(data[column_name].unique())}")

    plt.figure(figsize = figsize, tight_layout = False)
    sns.set(style = 'whitegrid', font_scale = 1.2)

    #plotting overall distribution of category
    plt.subplot(1,2,1)
    data_to_plot = data[column_name].value_counts().sort_values(ascending = False)
    ax = sns.barplot(x = data_to_plot.index, y = data_to_plot, palette = 'Set1')

    if percentage_display:
        total_datapoints = len(data[column_name].dropna())
        for p in ax.patches:
            ax.text(p.get_x() + horizontal_adjust, p.get_height() + 0.005 * total_datapoints,
                    '{:1.02f}%'.format(p.get_height() * 100 / total_datapoints), fontsize = fontsize_percent)

    plt.xlabel(column_name, labelpad = 10)
    plt.title(f'Distribution of {column_name}', pad = 20)
    plt.xticks(rotation = rotation)
    plt.ylabel('Counts')

    #plotting distribution of category for Defaulters
    if plot_defaulter:
        percentage_defaulter_per_category = (data[column_name][data.TARGET == 1].value_counts() * 100
                                             / data[column_name].value_counts()).dropna().sort_values(ascending =

        plt.subplot(1,2,2)
        sns.barplot(x = percentage_defaulter_per_category.index, y = percentage_defaulter_per_category,
                    palette = 'Set2')
        plt.ylabel('Percentage of Defaulter per category')
        plt.xlabel(column_name, labelpad = 10)
        plt.xticks(rotation = rotation)
        plt.title(f'Percentage of Defaulters for each category of {column_name}', pad = 20)
    plt.show()
```

```python
def plot_continuous_variables(data, column_name, plots = ['distplot', 'countplot', 'box'],
                              scale_limits = None, figsize = (20,8), histogram = True, log_scale = False):

    data_to_plot = data.copy()
    if scale_limits:
        #taking only the data within the specified limits
        data_to_plot[column_name] = data[column_name][(data[column_name] > scale_limits[0])
                                                      & (data[column_name] < scale_limits[1])]

    number_of_subplots = len(plots)
    plt.figure(figsize = figsize)
    sns.set_style('whitegrid')

    for i, ele in enumerate(plots):
        plt.subplot(1, number_of_subplots, i + 1)
        plt.subplots_adjust(wspace=0.25)

        if ele == 'distplot':
            sns.distplot(data_to_plot[column_name][data['TARGET'] == 0].dropna(),
                         label='Non-Defaulters', hist = False, color='red')
            sns.distplot(data_to_plot[column_name][data['TARGET'] == 1].dropna(),
                         label='Defaulters', hist = False, color='black')
            plt.xlabel(column_name)
            plt.ylabel('Probability Density')
            plt.legend(fontsize='medium')
            plt.title("Dist-Plot of {}".format(column_name))
            if log_scale:
                plt.xscale('log')
                plt.xlabel(f'{column_name} (log scale)')

        if ele == 'box':
            sns.boxplot(x='TARGET', y=column_name, data=data_to_plot)
            plt.title("Box-Plot of {}".format(column_name))
            if log_scale:
                plt.yscale('log')
                plt.ylabel(f'{column_name} (log Scale)')

        if ele == 'countplot':
            sns.countplot(data_to_plot[column_name],hue=data_to_plot['TARGET'])
#             ax.set_xticklabels(ax.get_xticklabels(), rotation= 60)
#             plt.show()

    plt.show()
```

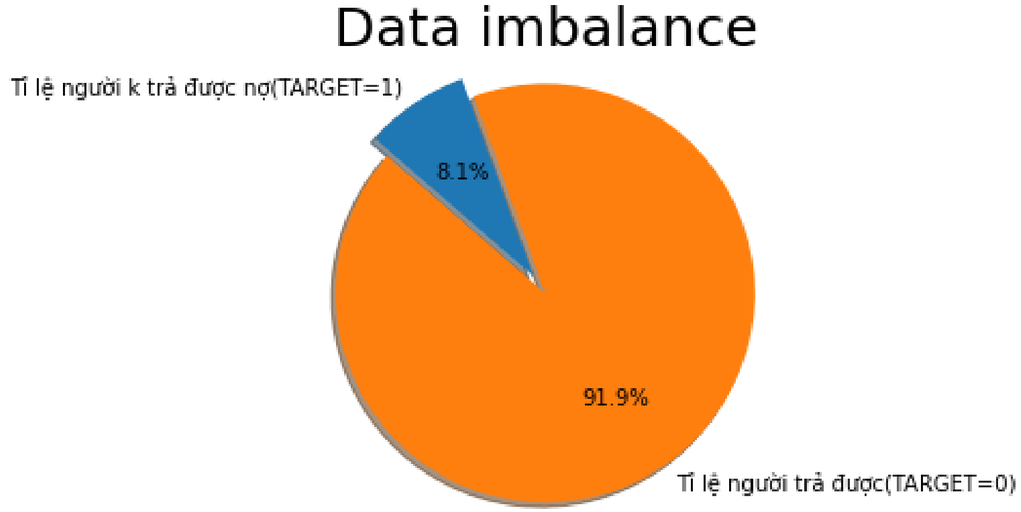
EDA OF EACH DATA

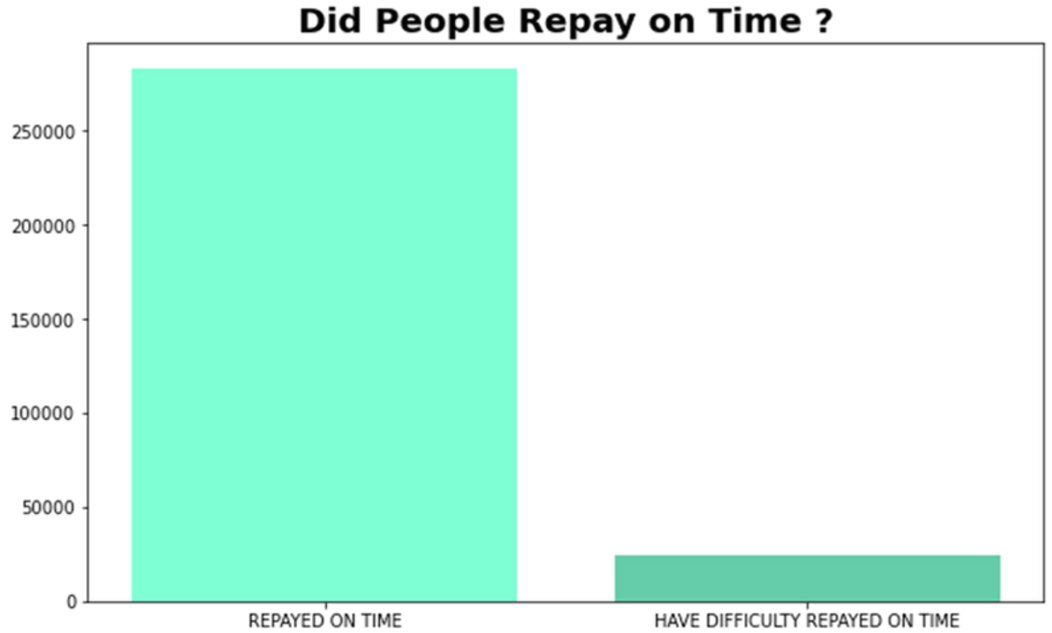
APPLICATION TRAIN

- THIS IS THE MAIN TABLE, BROKEN INTO TWO FILES FOR TRAIN (WITH TARGET) (IE. THE PREDICTION PROVIDED) AND TEST (WITHOUT TARGET).

- STATIC DATA FOR ALL APPLICATIONS. ONE ROW REPRESENTS ONE LOAN IN OUR DATA SAMPLE.

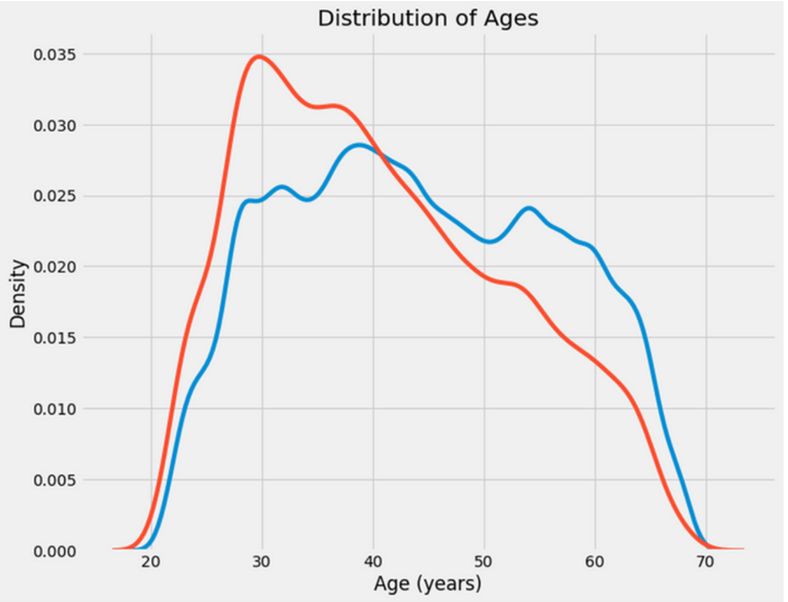
EXPLONATARY DATA ANALYSIS (EDA)





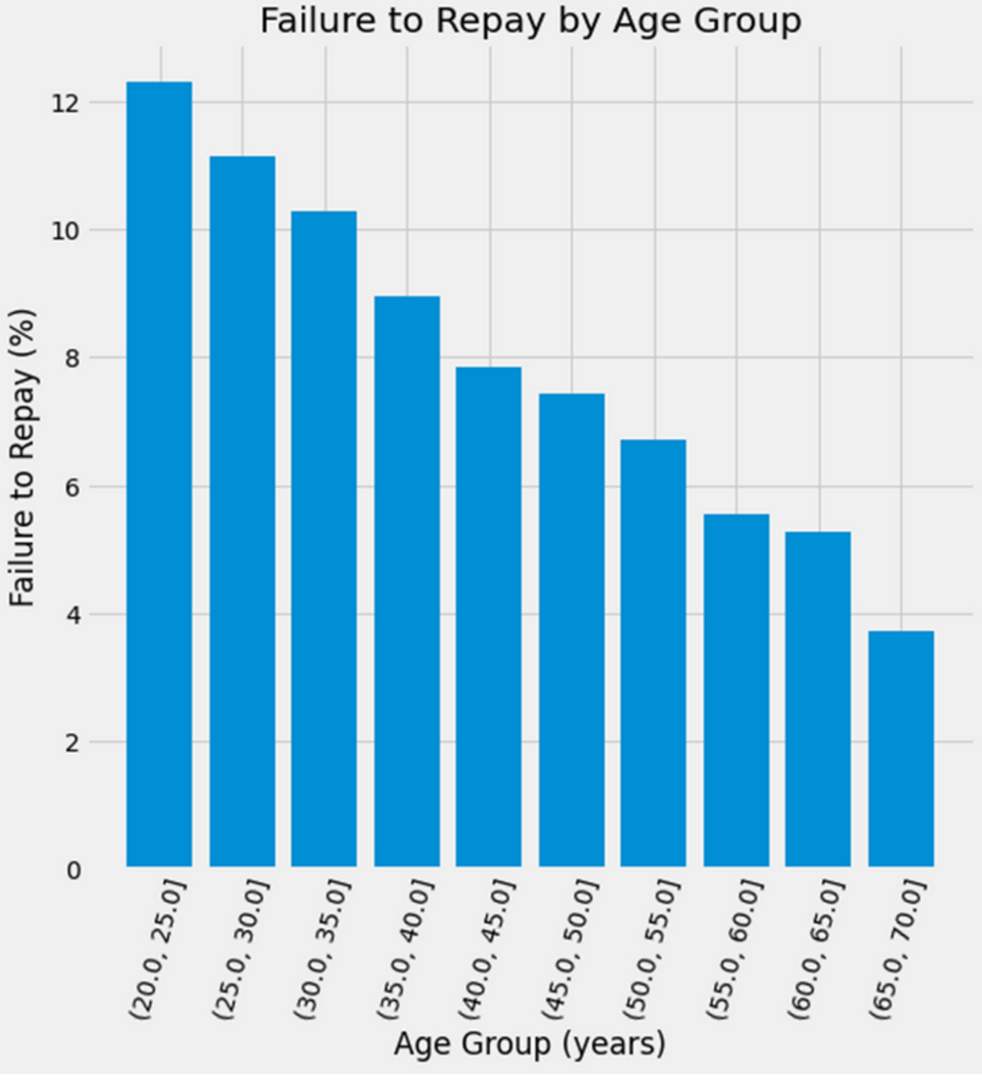
Most of the time customer paid it on time
But, you can see that the data have an imbalanced dataset.

EXPLONATARY DATA ANALYSIS (EDA)-AGE



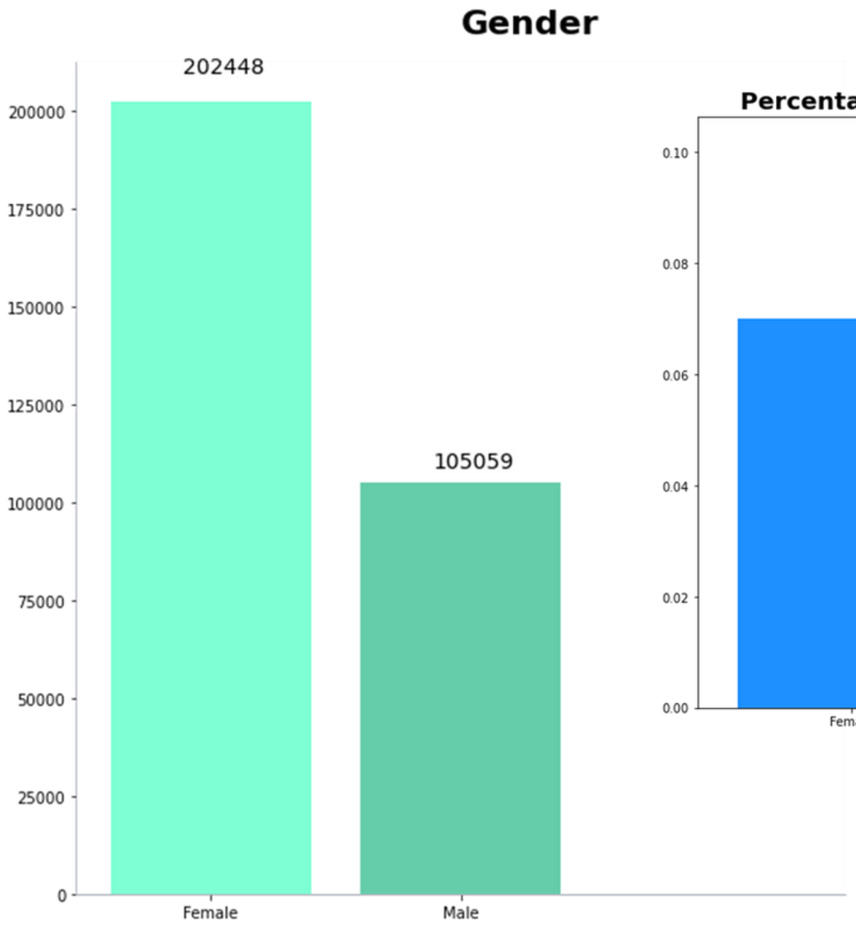
Looking at the Age distribution model of customers we can notice that most of the clients belong to the age of 40. And in the range of age 30 is where there is most of the number of defaulter and continuously decrease when the range increase

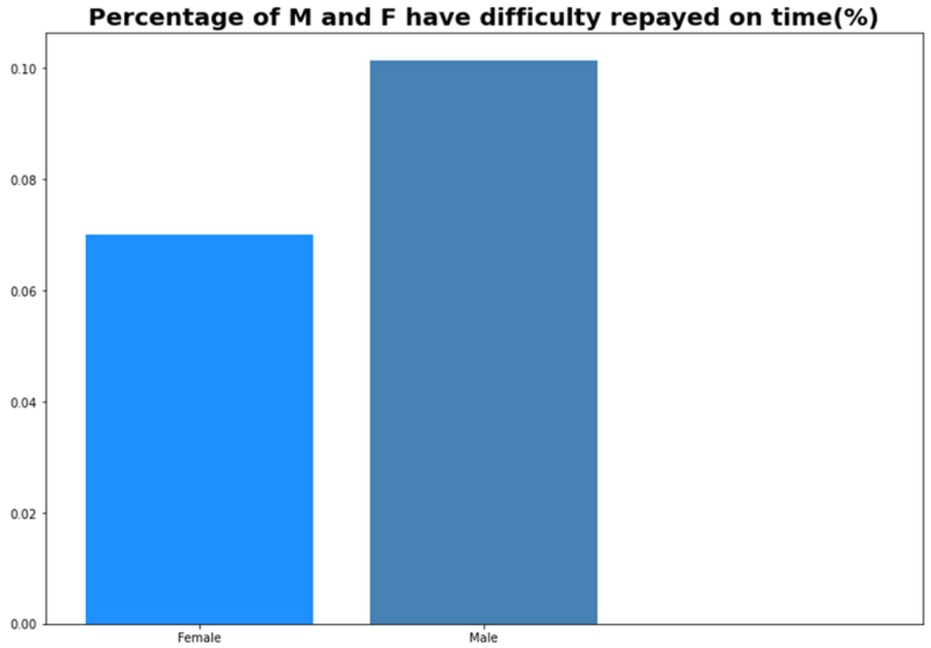
EXPLONATARY DATA ANALYSIS (EDA)-AGE



Looking at the graph and all of the previous information, we can go to conclusion is that the young's customer usually cant paid the debt on time.

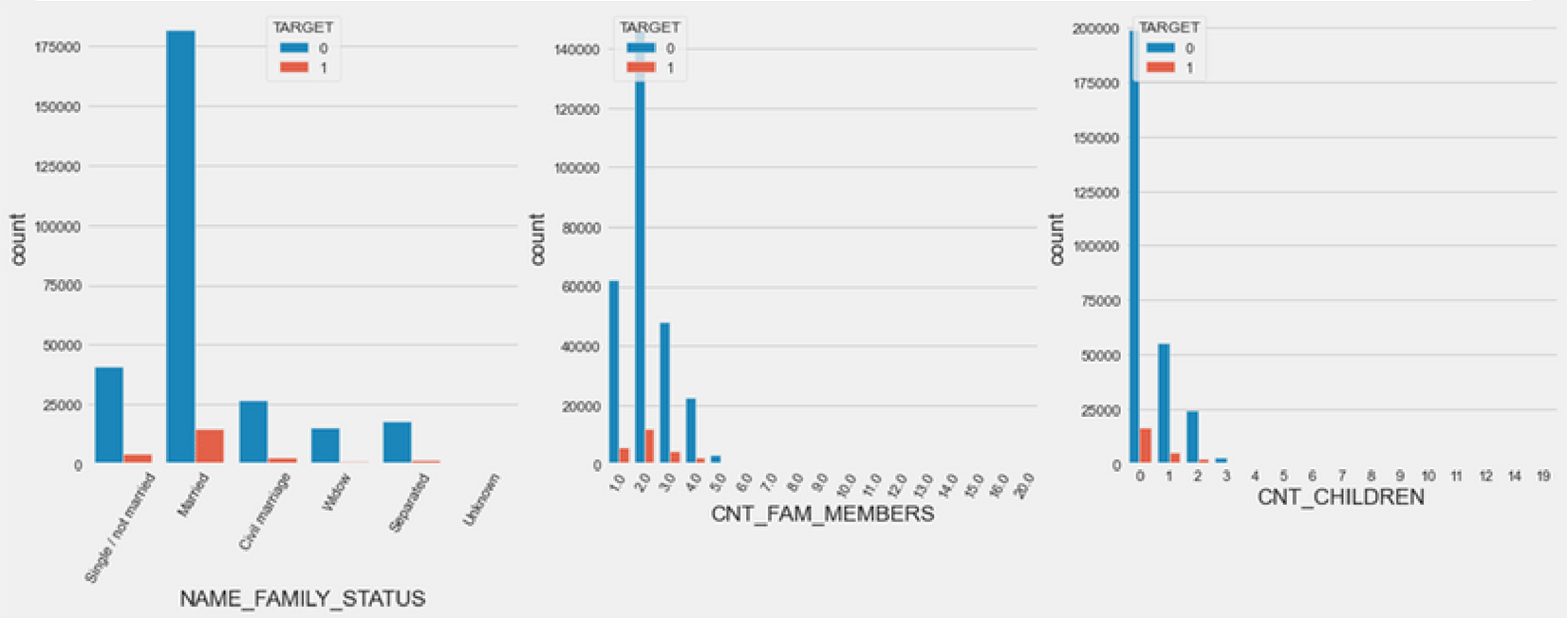
EXPLONATARY DATA ANALYSIS (EDA)-GENDER





The amount of female borrowers is bigger than that of male borrowers. Although the amount of female borrowers is almost double the other one but there is only 7% of them can not paid off the debt, whereas the men have 10% of them who cannot paid it

EXPLONATARY DATA ANALYSIS (EDA)-FAMILY INFO

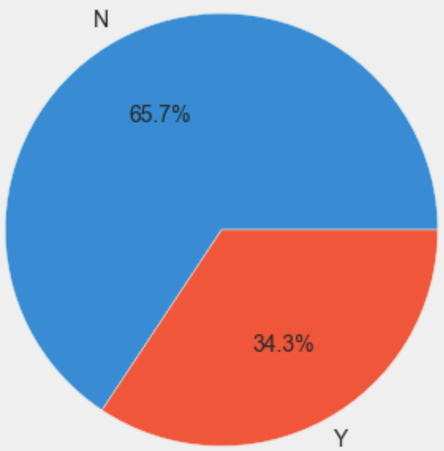


- In terms of default rate, Civil Marriage has the highest default rate (10%), with Widow having the lowest (except for Unknow).
- For customers with 9 or 11 children, the non-repayable loan rate is 100%.
- Customers with family members of 2 people are the most, followed by 1 (single person), 3 (family with one child) and 4. However, the number is getting less and less as the number of members increases.

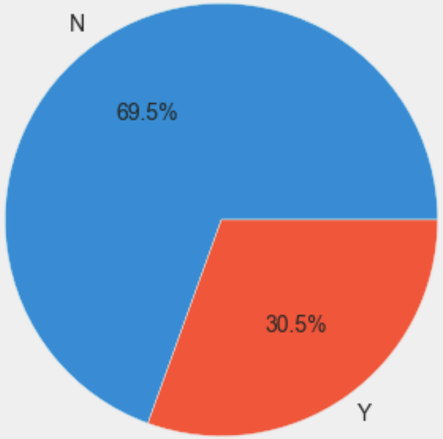
EXPLONATARY DATA ANALYSIS (EDA)-ASSET DETAIL

FLAG_OWN_CAR

Non-defaulter

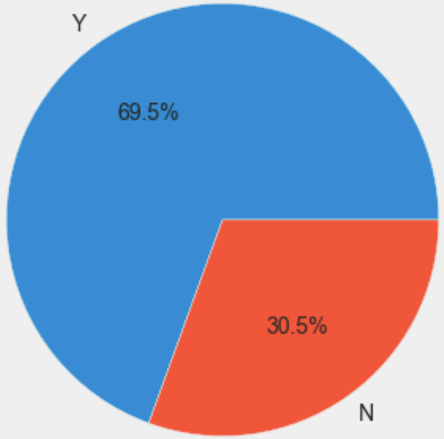


Defaulter

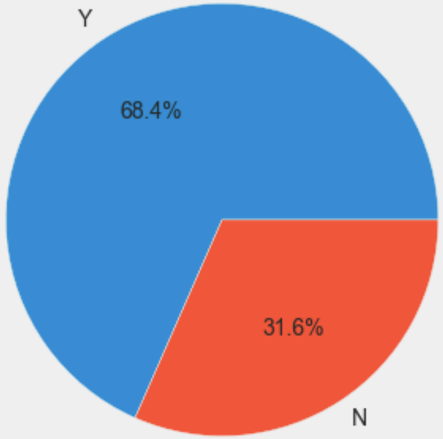


FLAG_OWN_REALTY

Non-defaulter

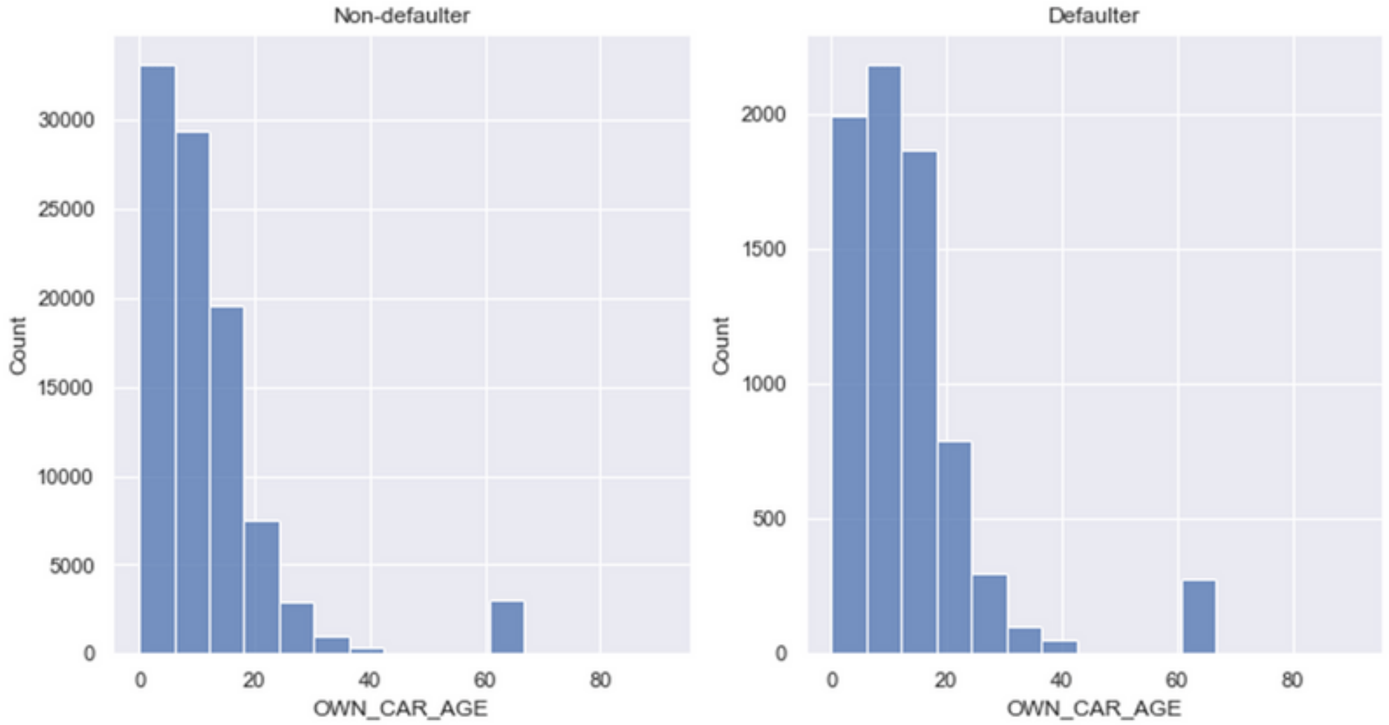


Defaulter



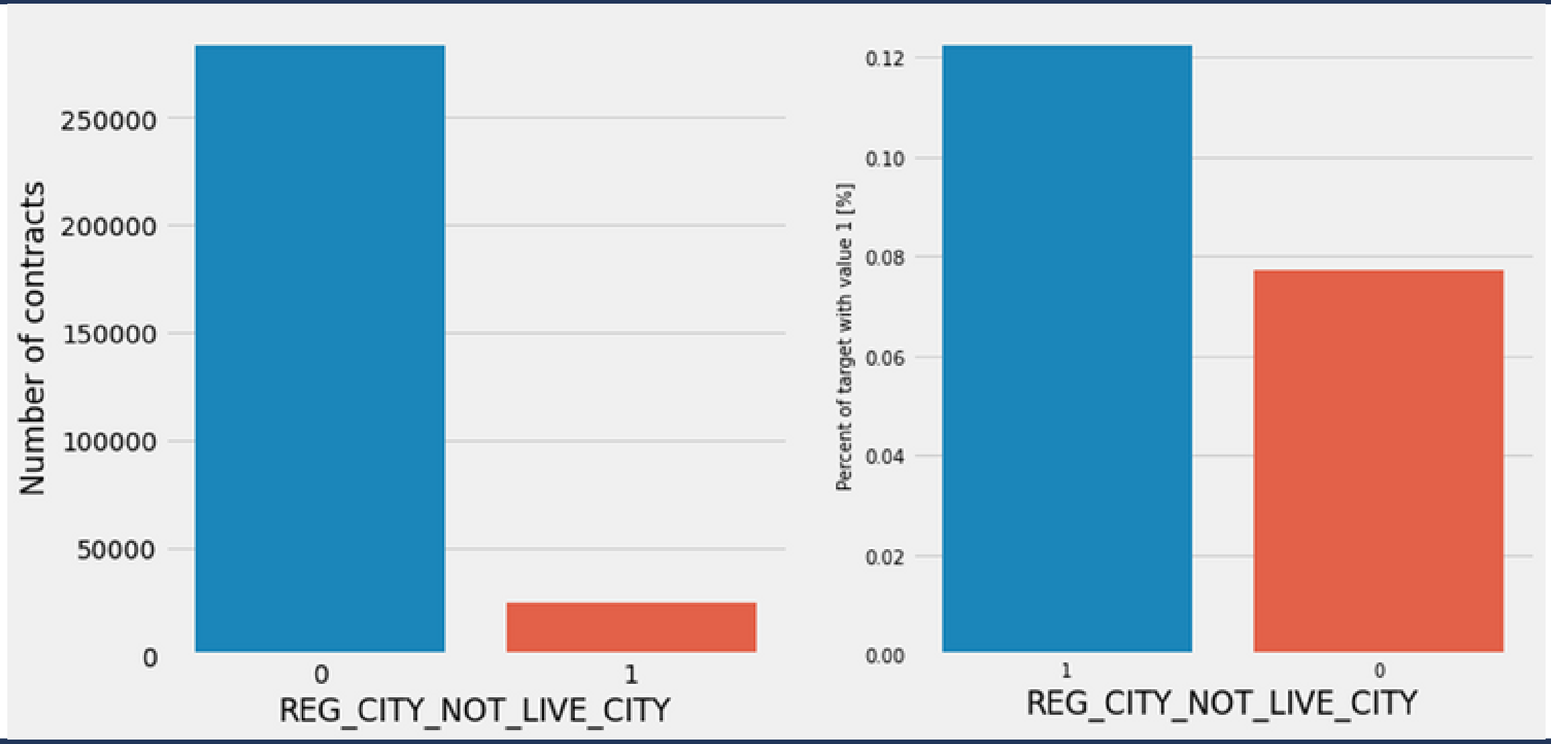
- The clients that owns a car are almost a half of the ones that doesn't own one. The clients that owns a car are less likely to not repay a car that the ones that own. Both categories have not-repayment rates around 8%.
- The clients that owns real estate are more than double of the ones that doesn't own. Both categories (owning real estate or not owning) have not-repayment rates less than 8%.

EXPLONATARY DATA ANALYSIS (EDA)-ASSET DETAIL



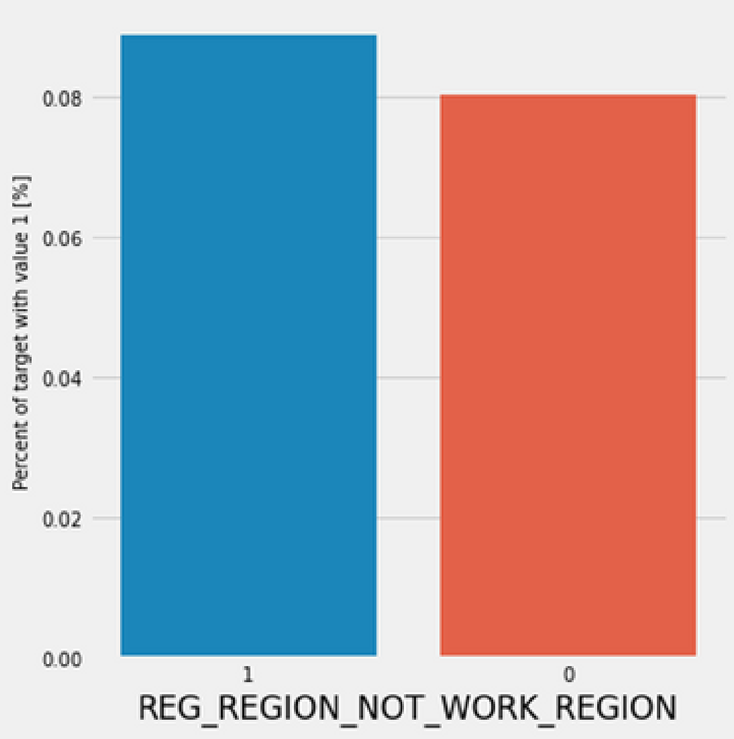
- Most of the customers own car is betwweb the age of 0-25 and a small number of car owner belong to the age of 40 and 60.

EXPLONATARY DATA ANALYSIS (EDA)-REGION INFO



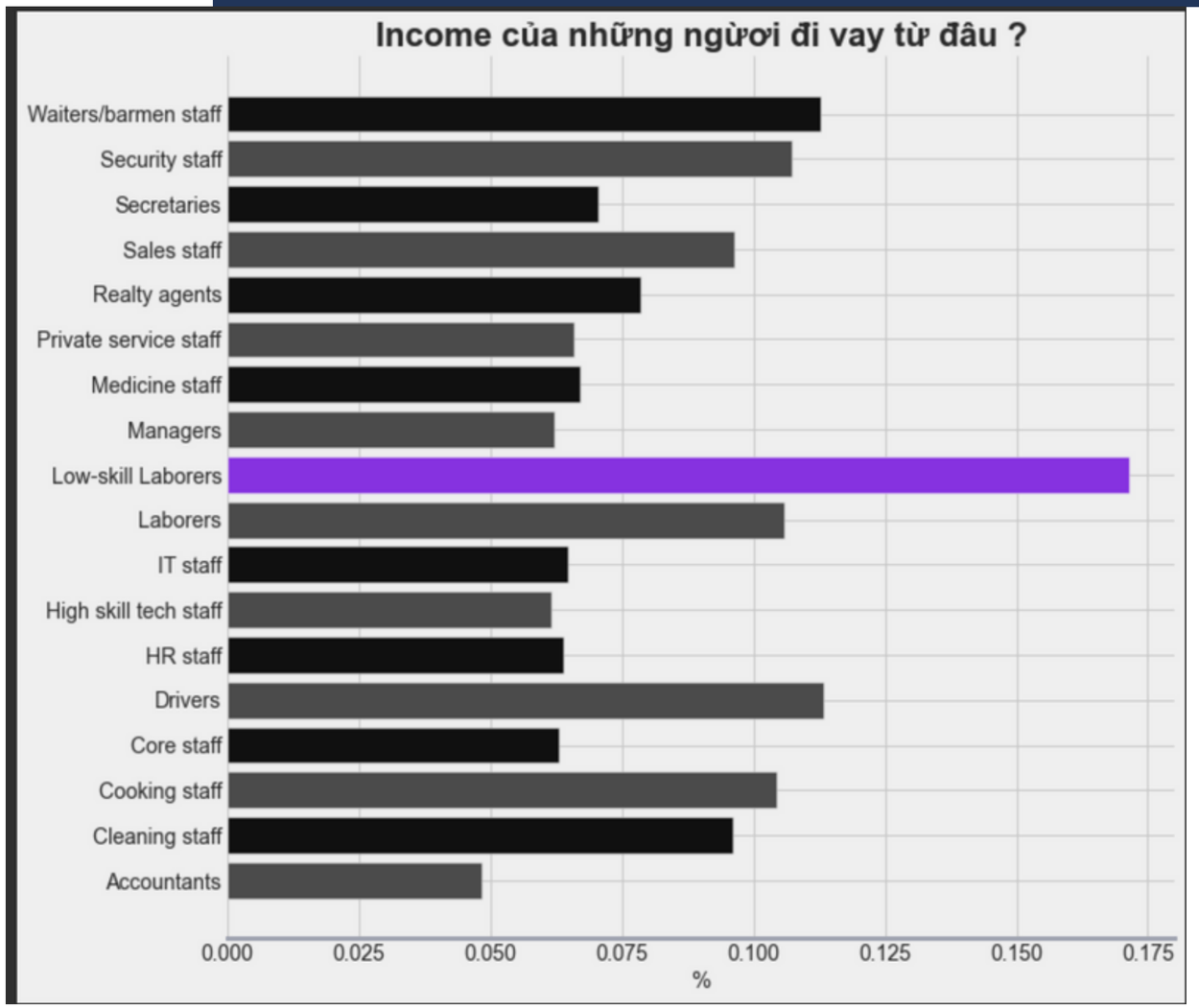
- In general, there are more people registered in the city they live or work in (a large number are registered in a different working city than the city in which they live).
- People who apply in a city other than the one they work or live in are more likely to default on their loans than those who apply in the same city .

EXPLONATARY DATA ANALYSIS (EDA)-REGION INFO



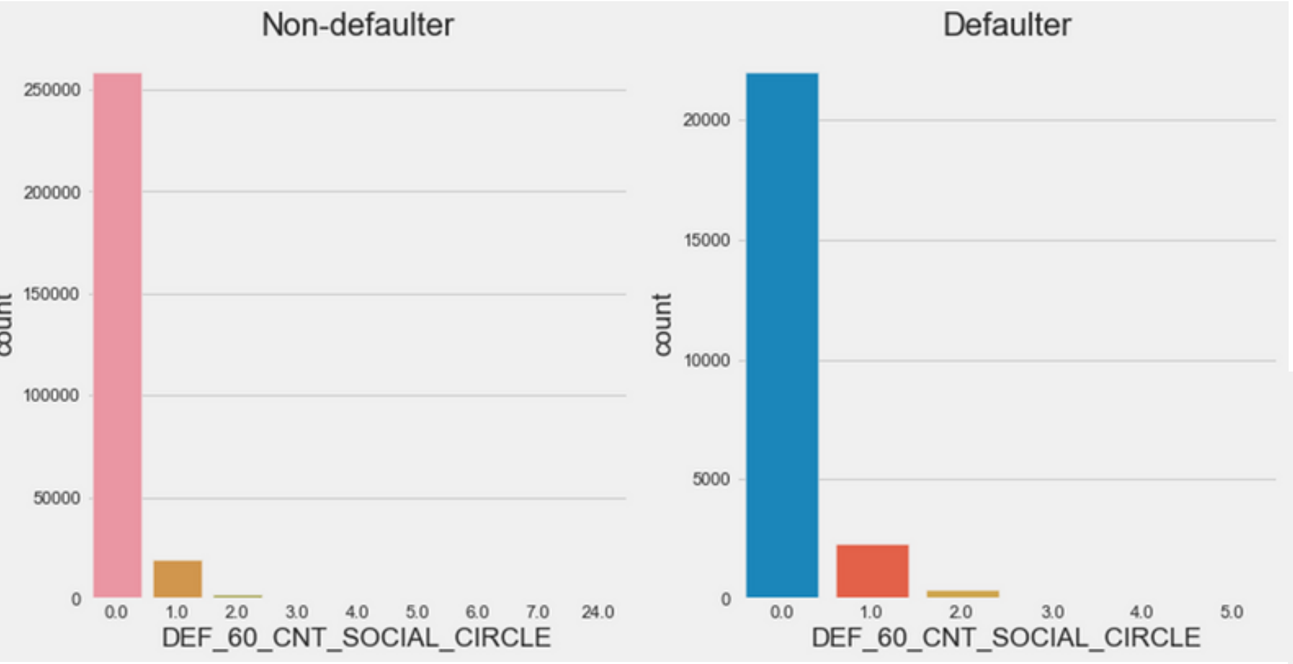
- Very few people are registered in the area where they do not live or work. In general, the no-return rate for these cases is slightly greater than for the rest (slightly above 8%)

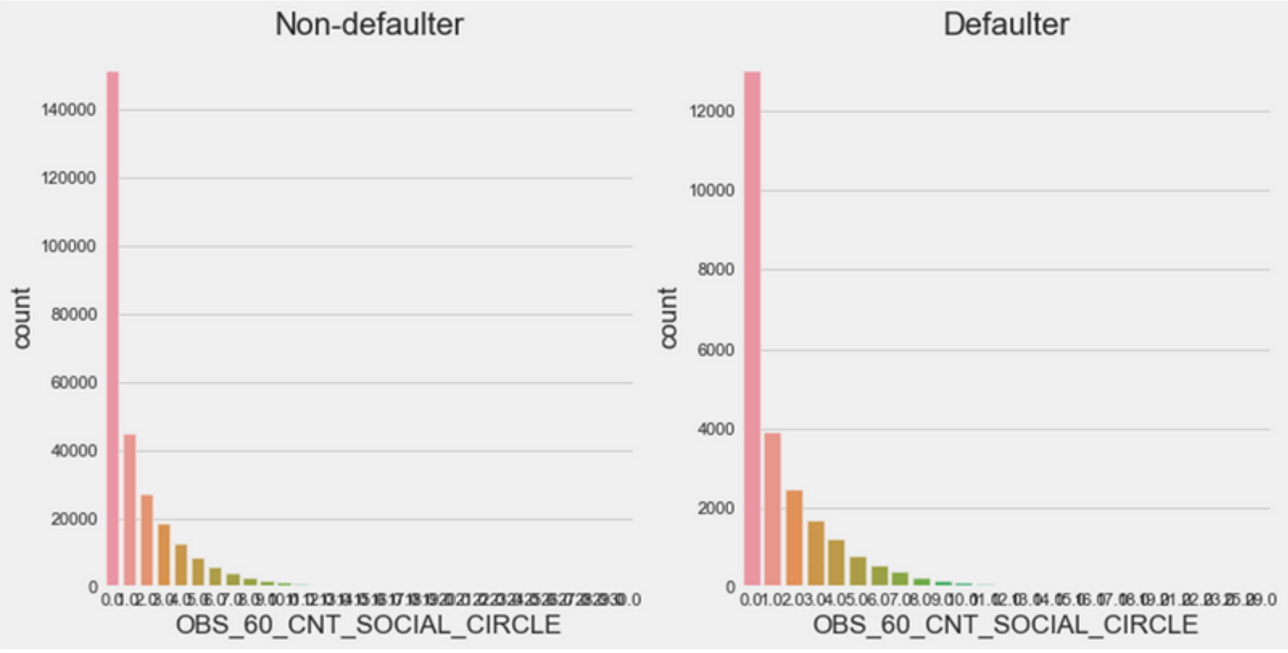
EXPLONATARY DATA ANALYSIS (EDA)-REGION INFO



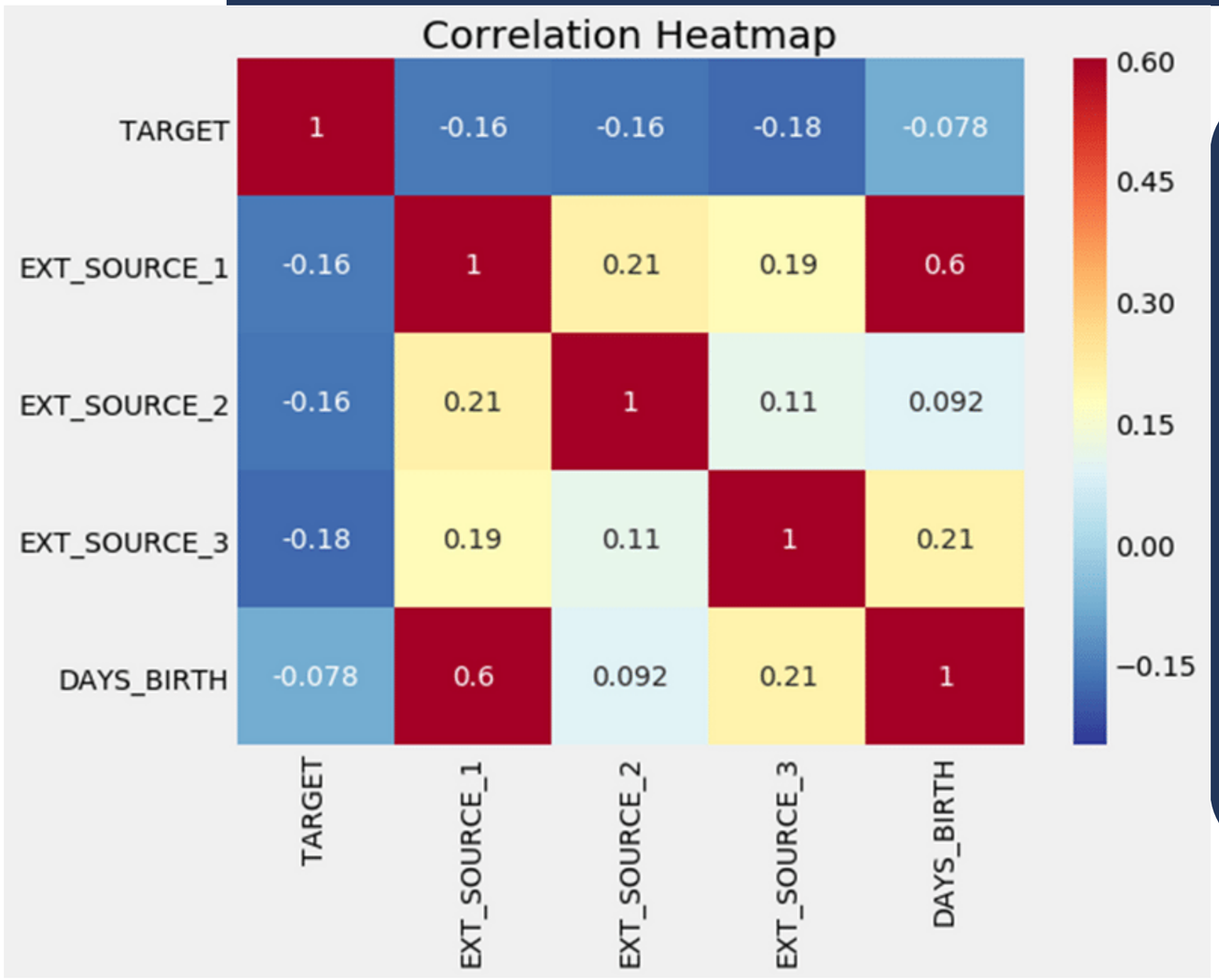
- The category with highest percent of not repaid loans are Low-skill Laborers (above 17%)
- This can be explain because most of the job is hand labor so the income is not much which make them hard to pay it

EXPLONATARY DATA ANALYSIS (EDA)-FEATURE



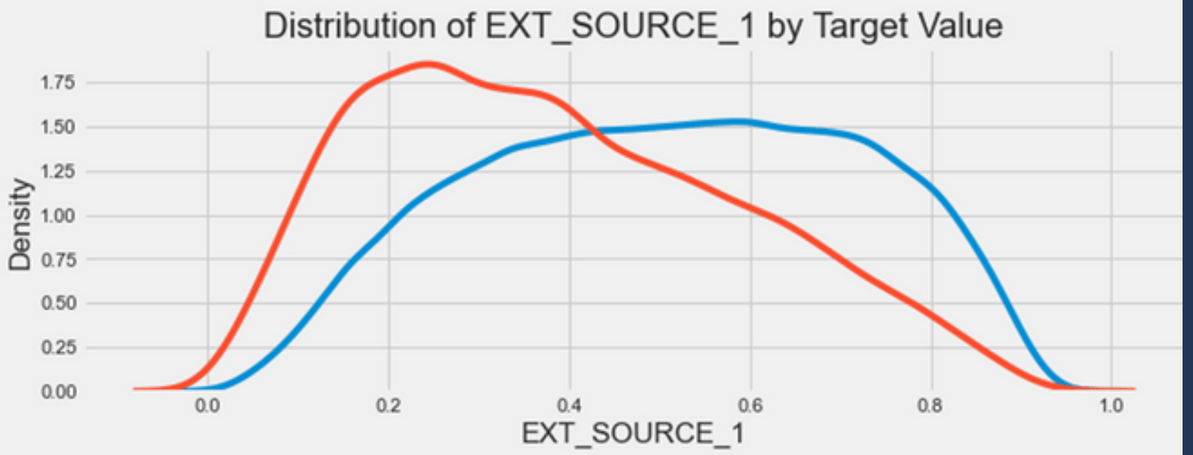


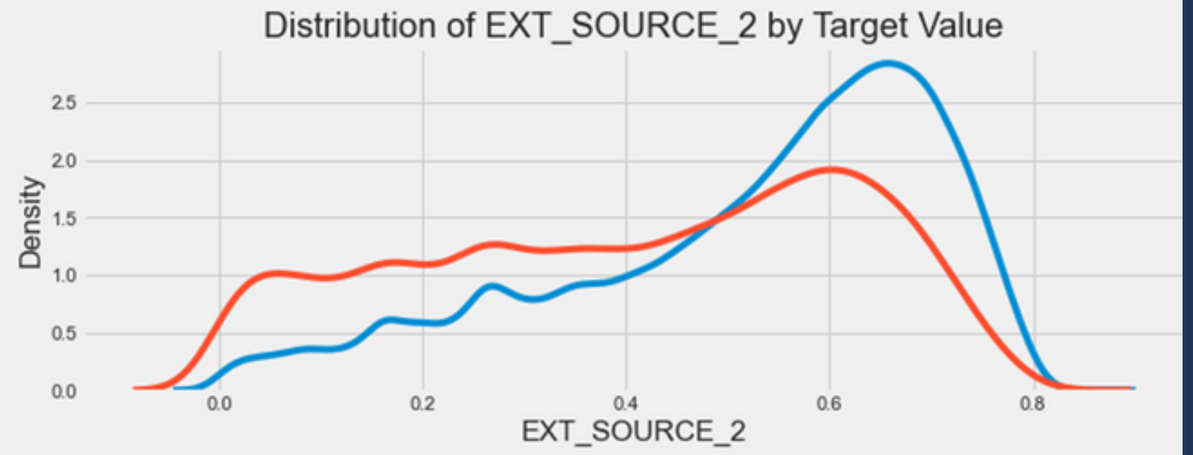
EXPLONATARY DATA ANALYSIS (EDA)-REGION INFO

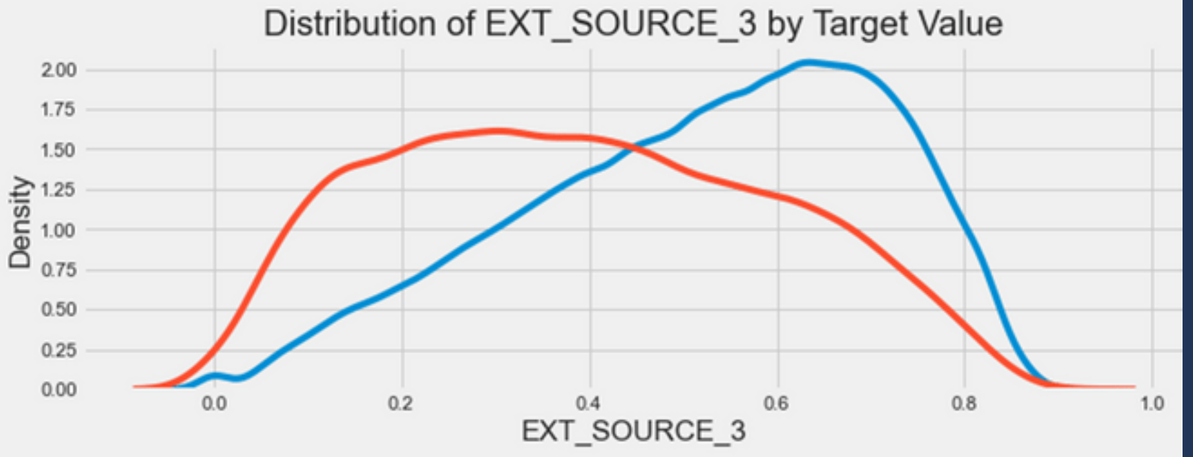


- All three EXT_SOURCE features have negative correlations with the target, indicating that as the value of the EXT_SOURCE increases, the client is more likely to repay the loan. We can also see that DAYS_BIRTH is positively correlated with EXT_SOURCE_1 indicating that maybe one of the factors in this score is the client age.

EXPLONATARY DATA ANALYSIS (EDA) FEATURE





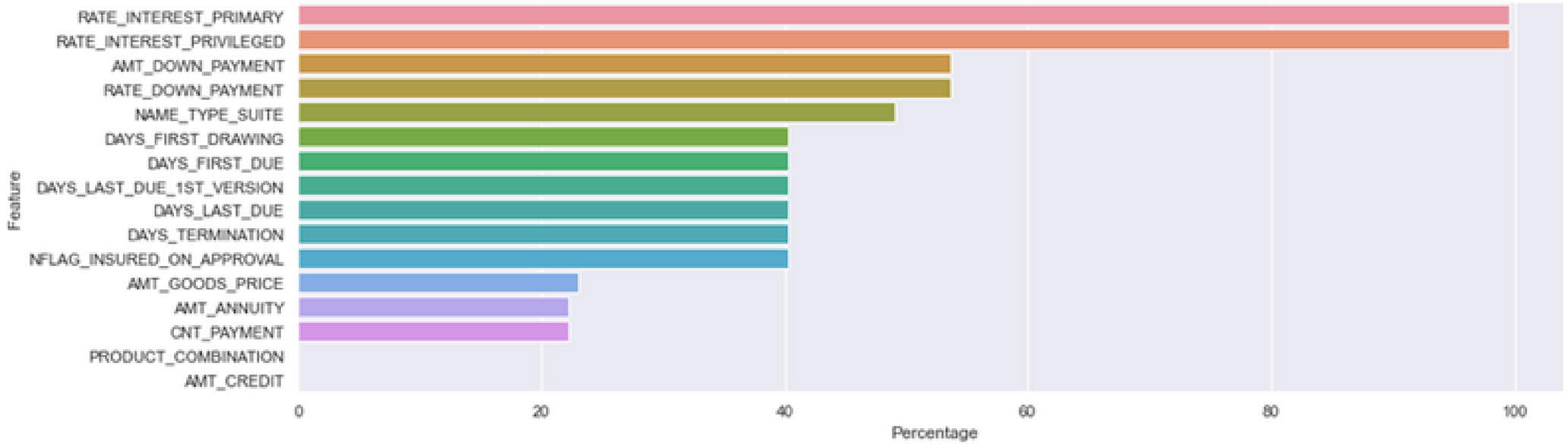


- **EXT_SOURCE_3 displays the greatest difference between the values of the target. We can clearly see that this feature has some relationship to the likelihood of an applicant to repay a loan.**
- **The relationship is not very strong. In fact they are all considered very weak, but these variables will still be useful for a machine learning model to predict whether or not an applicant will repay a loan on time.**

PREVIOUS APPLICATION

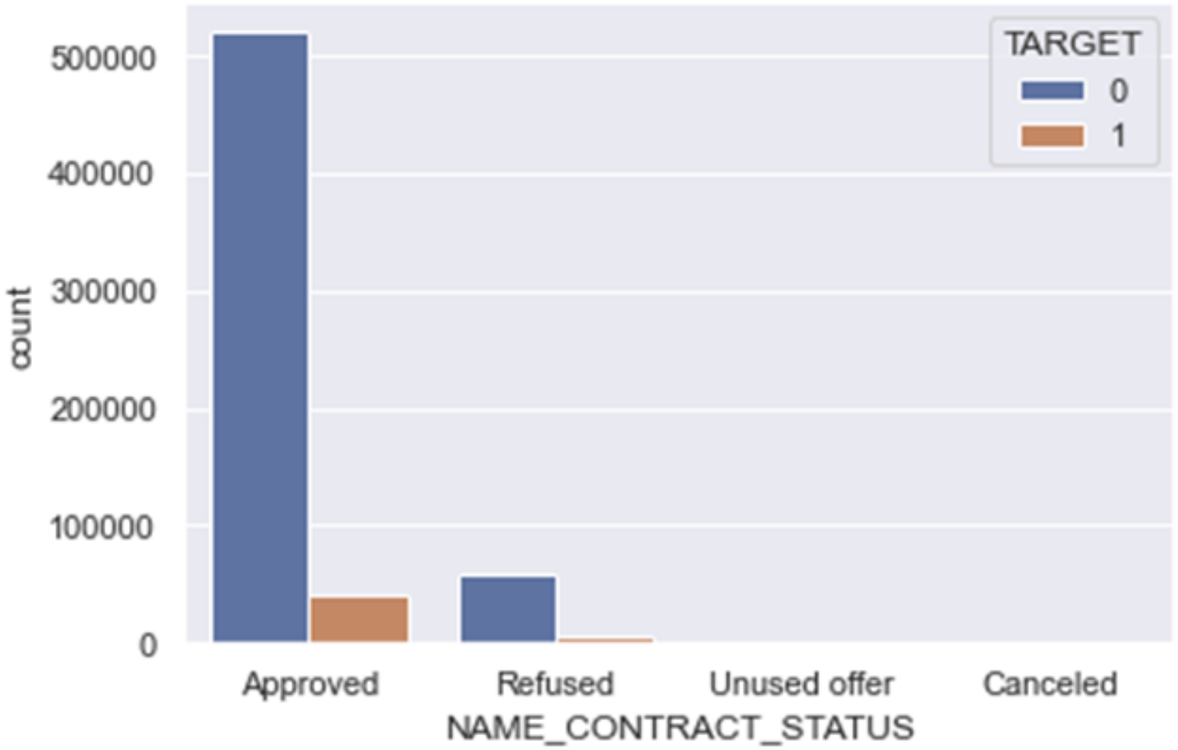
- All previous applications for Home Credit loans of clients who have loans in our sample
- There is one row for each previous application related to loans in our data sample

EXPLONATARY DATA ANALYSIS (EDA)



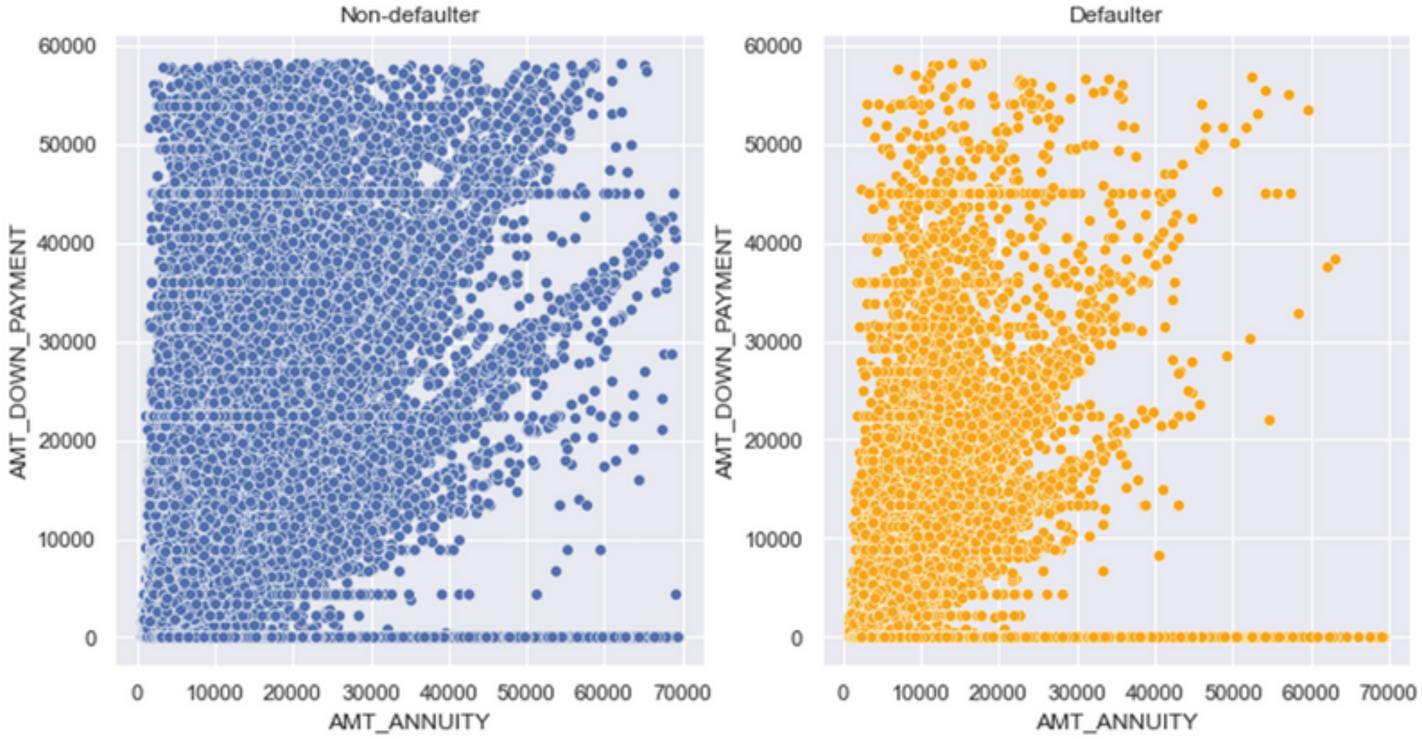
The graph above show us the null percentage of the previous-application data. You can see in the graph is that the column RATE_INTEREST_PRIMARY, RATE_INTEREST_PRIVILEGED have more than 99% of missing data so we dropped it. On the other hand, the column AMT_CREDIT, PRODUCT_COMBINATION have a very small percentage of null value so we will fill it.

EXPLONATARY DATA ANALYSIS (EDA)



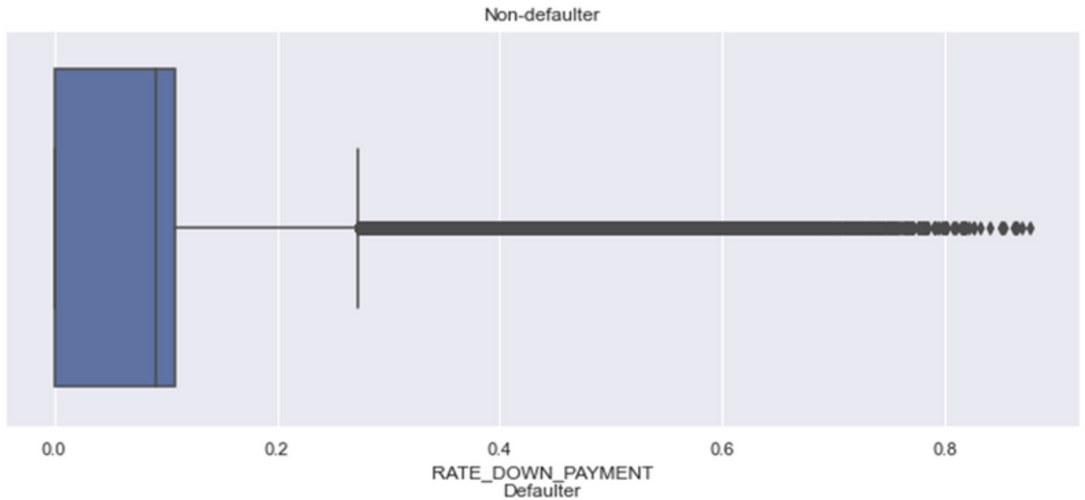
- Customers with previously approved loans are more likely to pay their current loan on time than customers whose previous loan got declined
- 7% among the customers that have their previously loan approved doesn't have any current loan
- 90% of the customer group whose previously loan got declined are capable of paying their current loan

EXPLONATARY DATA ANALYSIS (EDA)



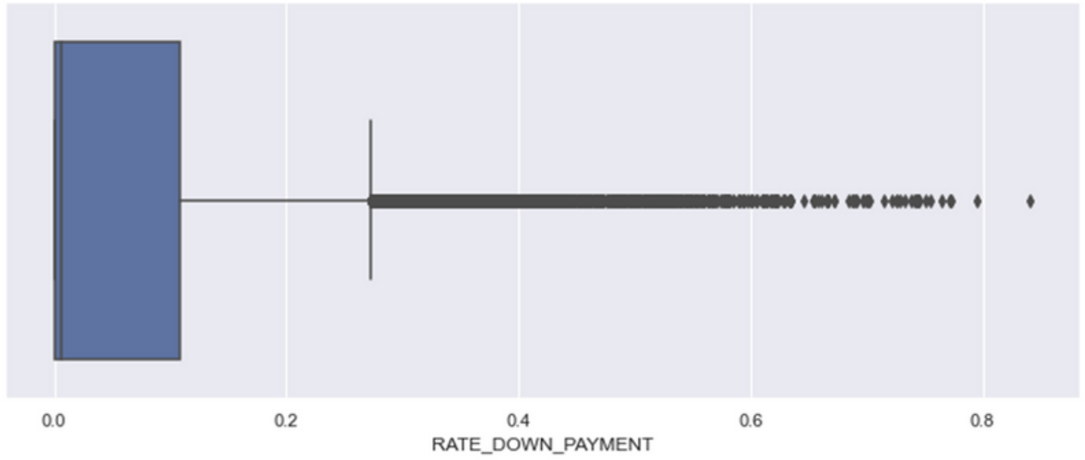
Looking at the graph, we can see that as the AMT_ANNUITY getting higher, the lower the number of defaulter. On the contrary, the higher the DOWN_PAYMENT, the more the number of defaulter appear.

EXPLONATARY DATA ANALYSIS (EDA)



Quantile

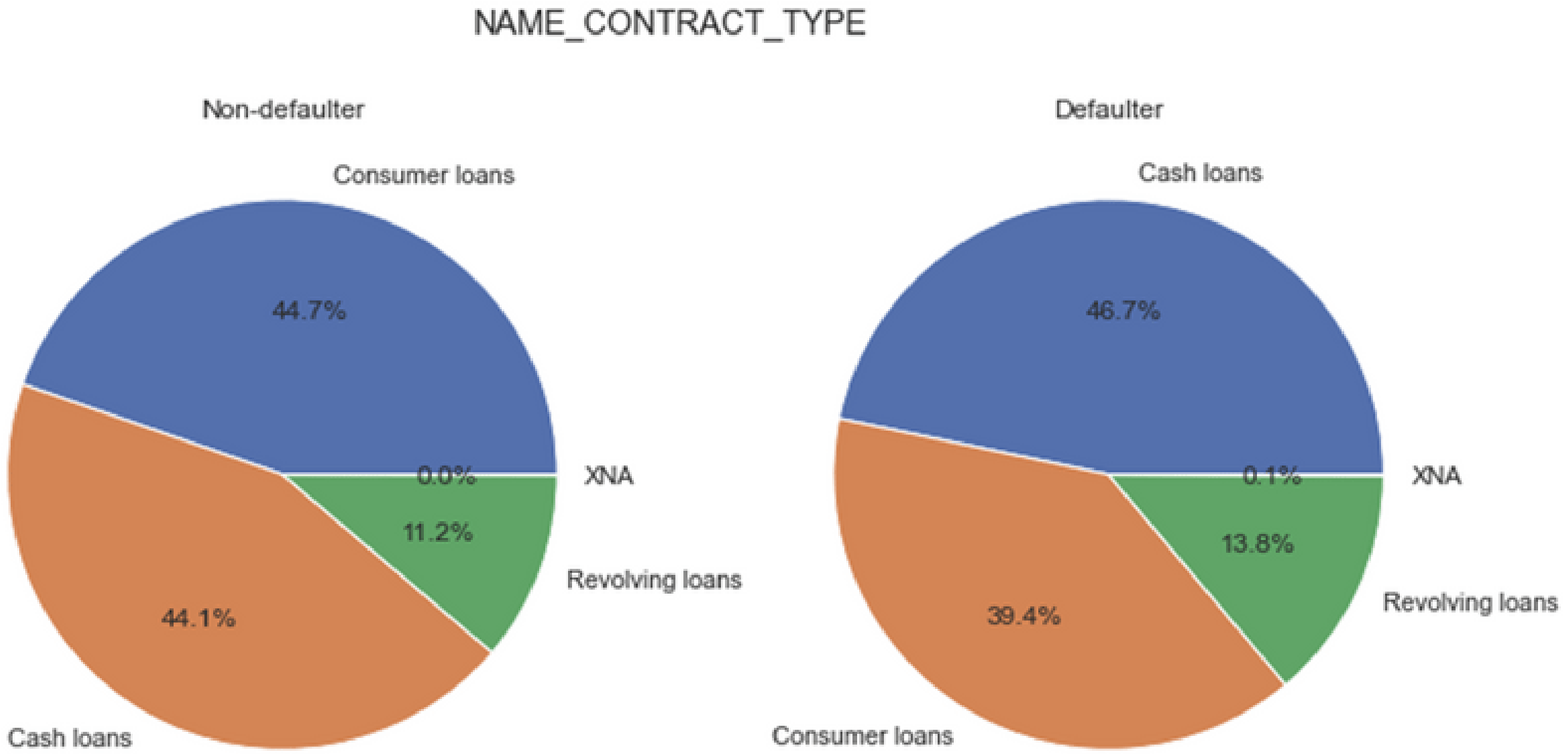
0.50	0.067233
0.70	0.108909
0.90	0.212315
0.95	0.299530
0.99	0.522992



0.50	0.000000
0.70	0.103965
0.90	0.199209
0.95	0.237696
0.99	0.477945

The customers whose RATE_DOWN_PAYMENT is low in data previous application, more likely to become a defaulter

EXPLONATARY DATA ANALYSIS (EDA)

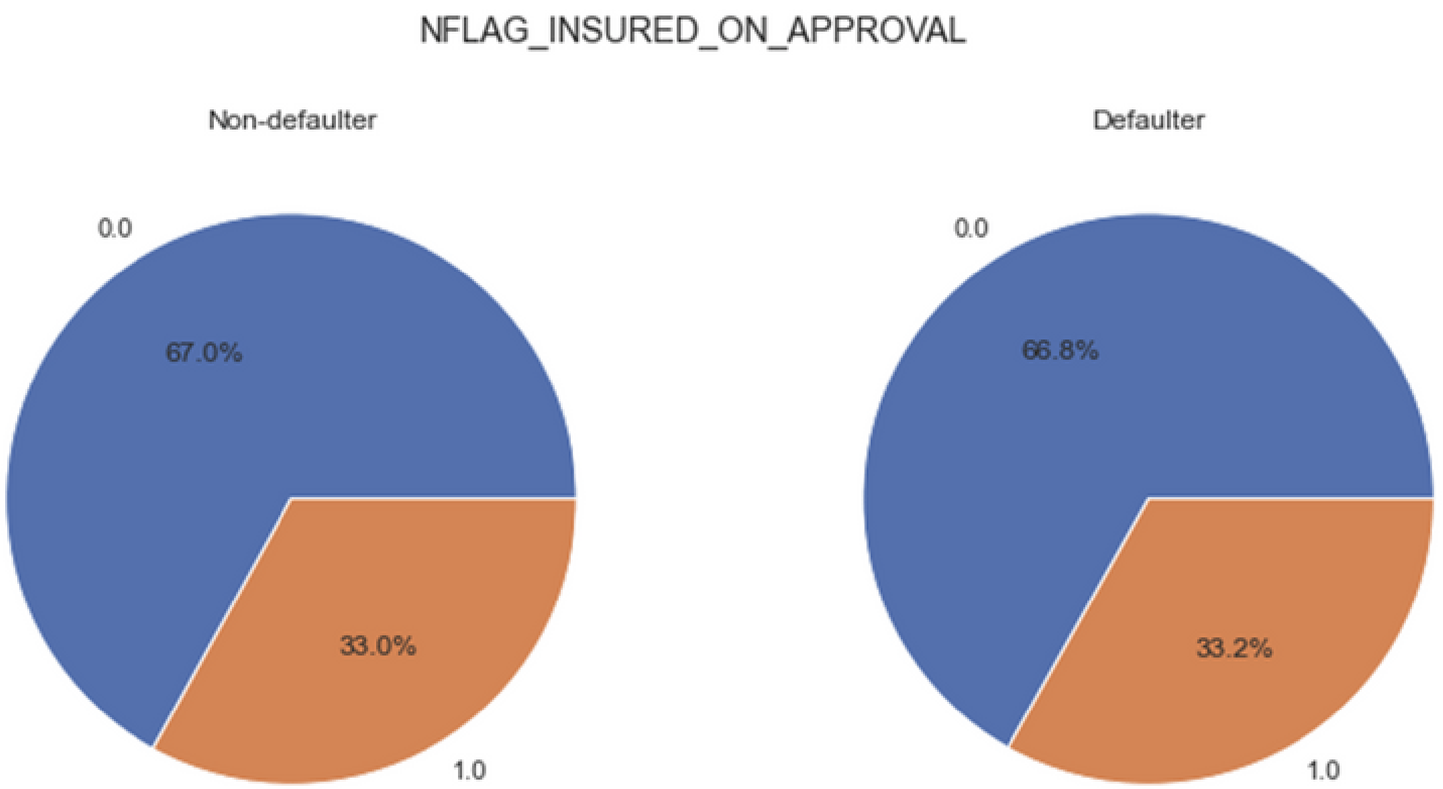


Most of the credit loan is consumer loans

EXPLONATARY DATA ANALYSIS (EDA)

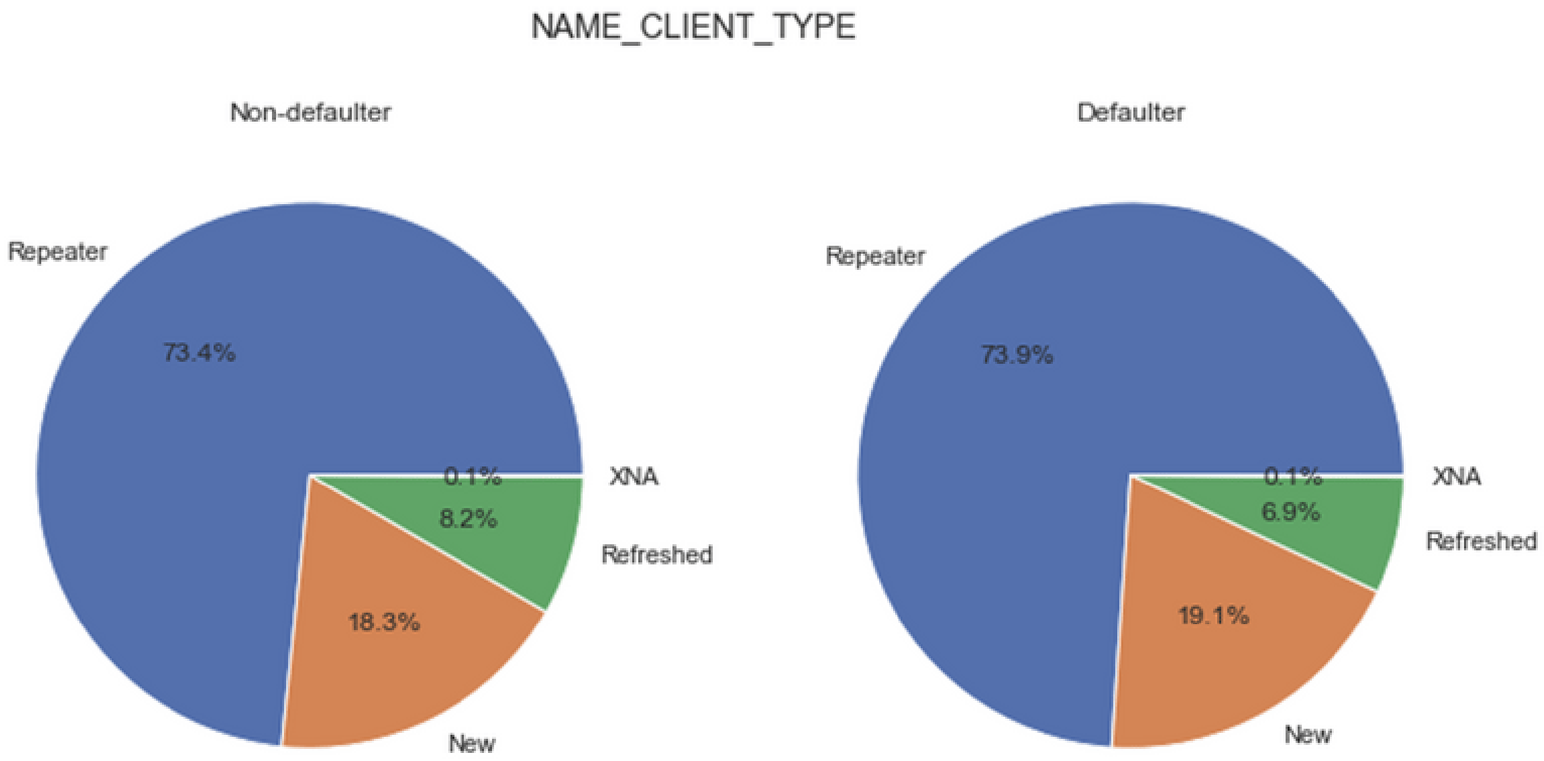
“SCO”, “LIMIT”, “HC" are the most rejected reasons

EXPLONATARY DATA ANALYSIS (EDA)



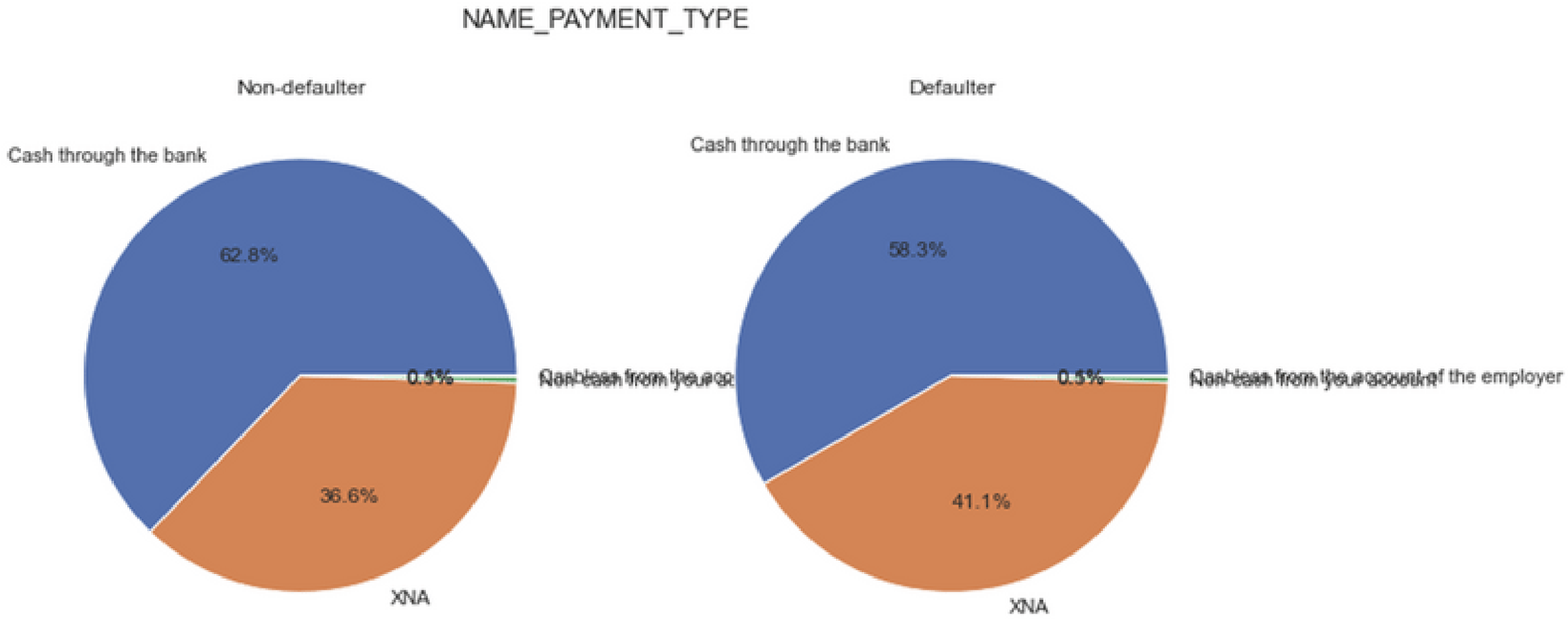
Most people do not require insurance during the previous loan application process.

EXPLONATARY DATA ANALYSIS (EDA)

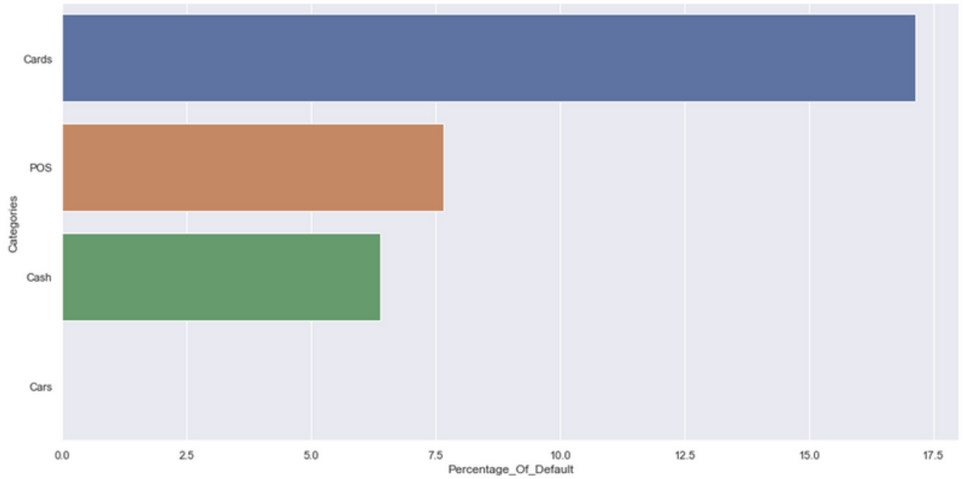


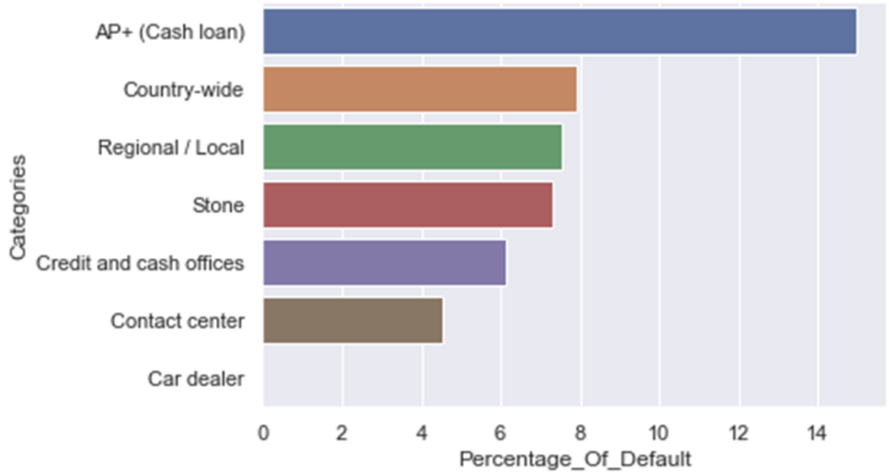
Most of the borrower is Repeater (Loyal customers)

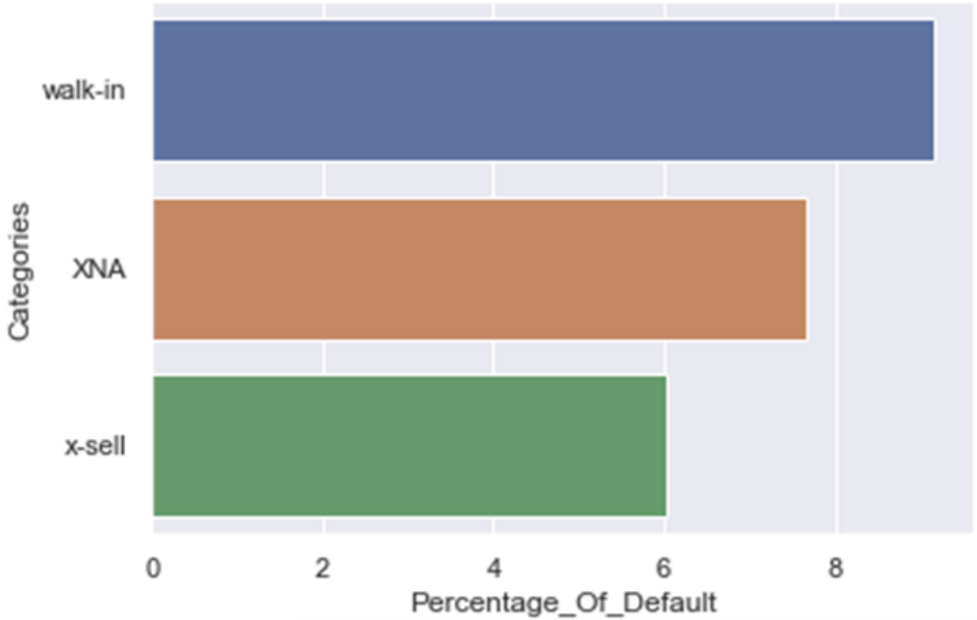
EXPLONATARY DATA ANALYSIS (EDA)

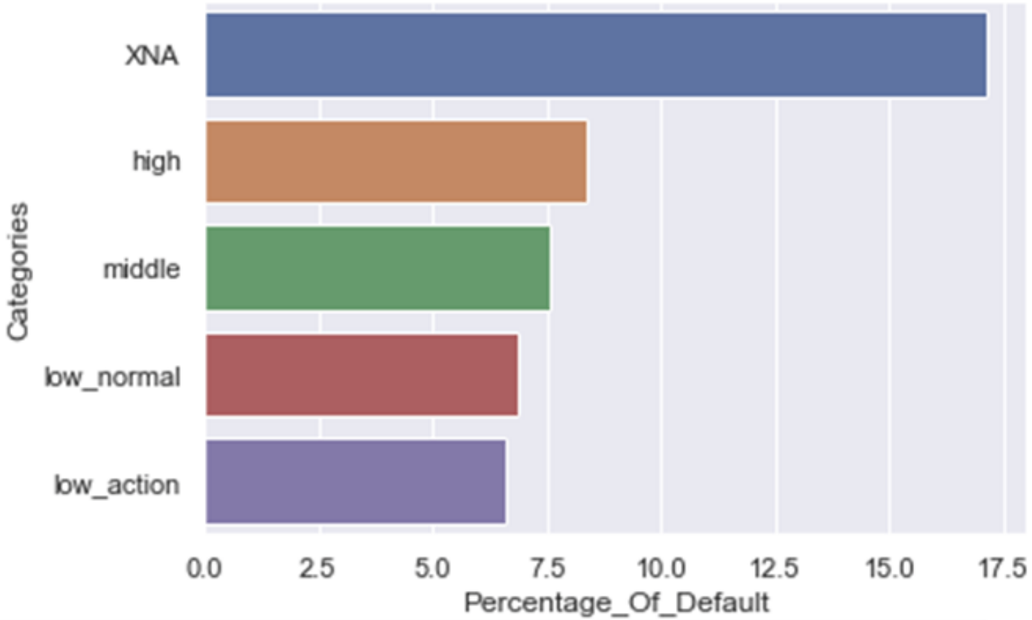


Cash through bank is the most popular method to pay

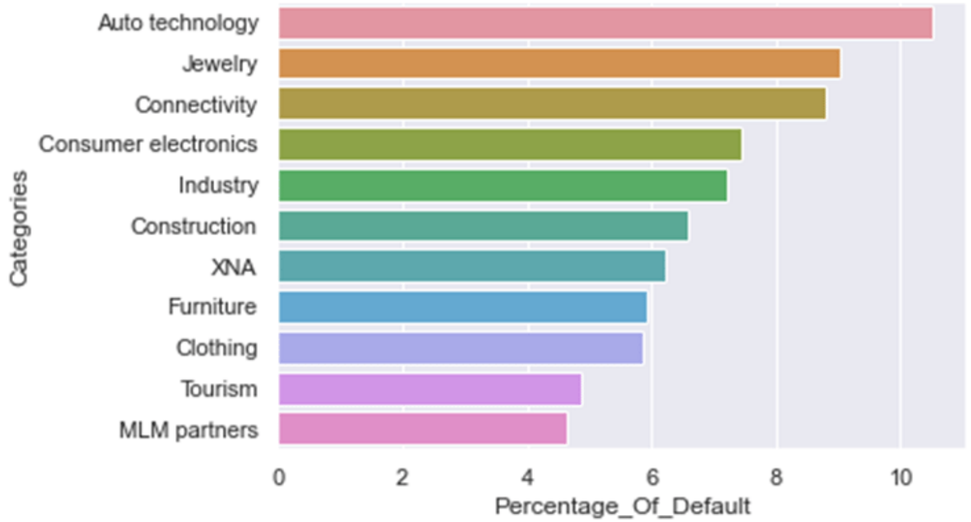


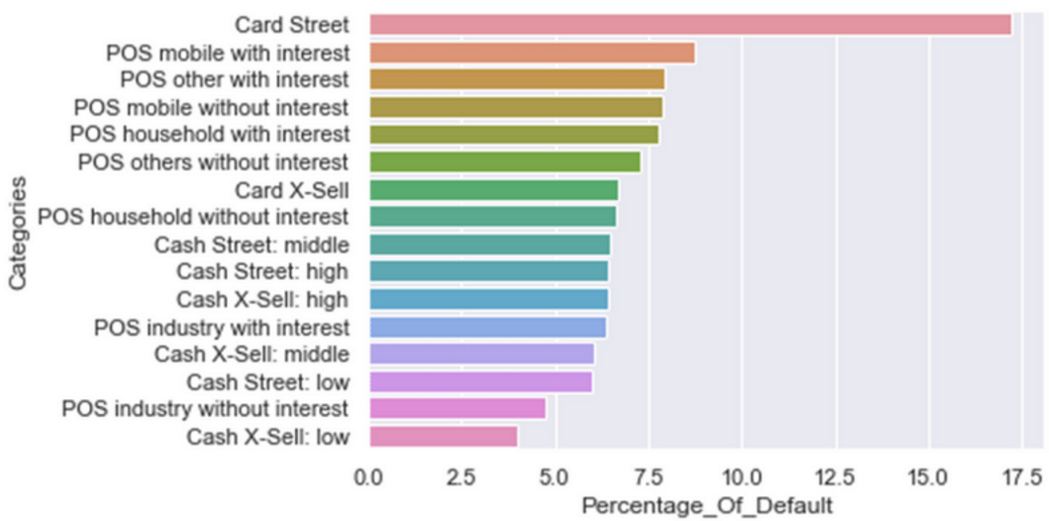


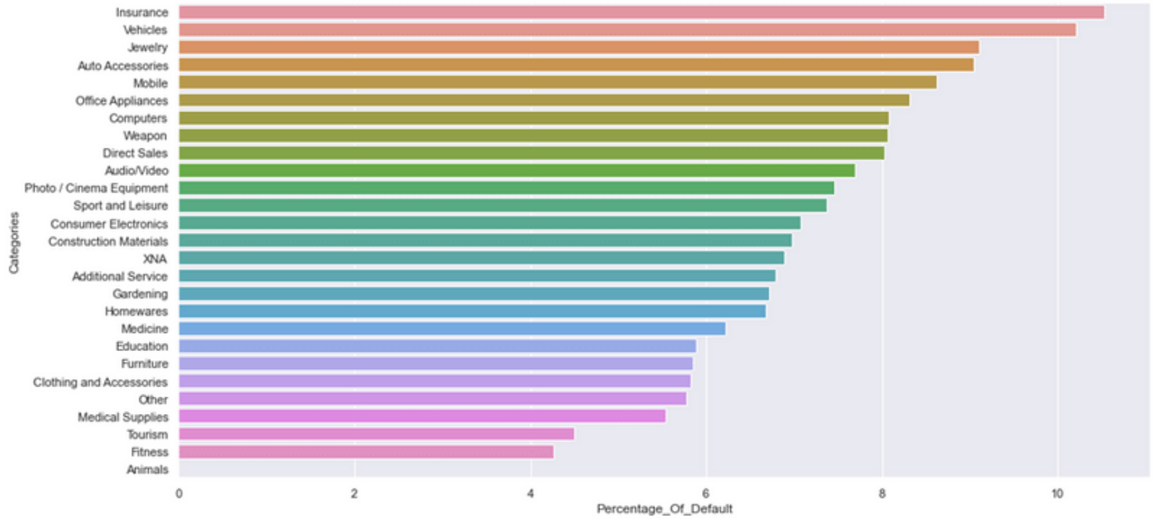




- XNA has the highest default rate
- AP+(Cash loan) has the highest default rate
- Walk-in has the highest default rate
- Cards has the highest default rate





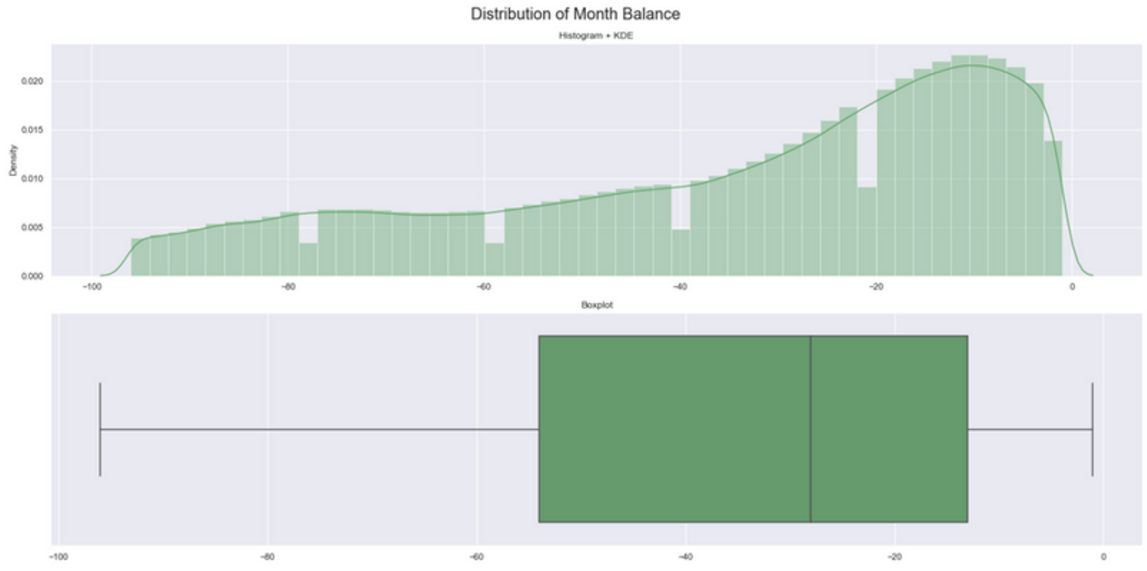


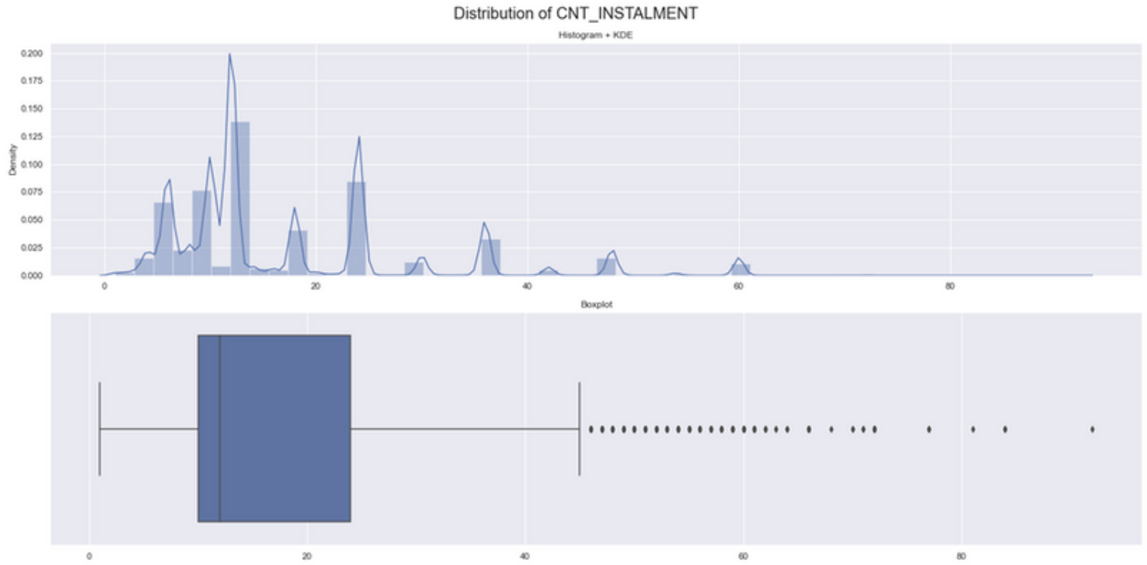
- Customer use the loan moneys for Insurance or vehicles is most likely to become defaulter
- In the seller industry “Auto technology” has the highest default rate
- Card Street has the highest default rate

POS-CASH BALANCE

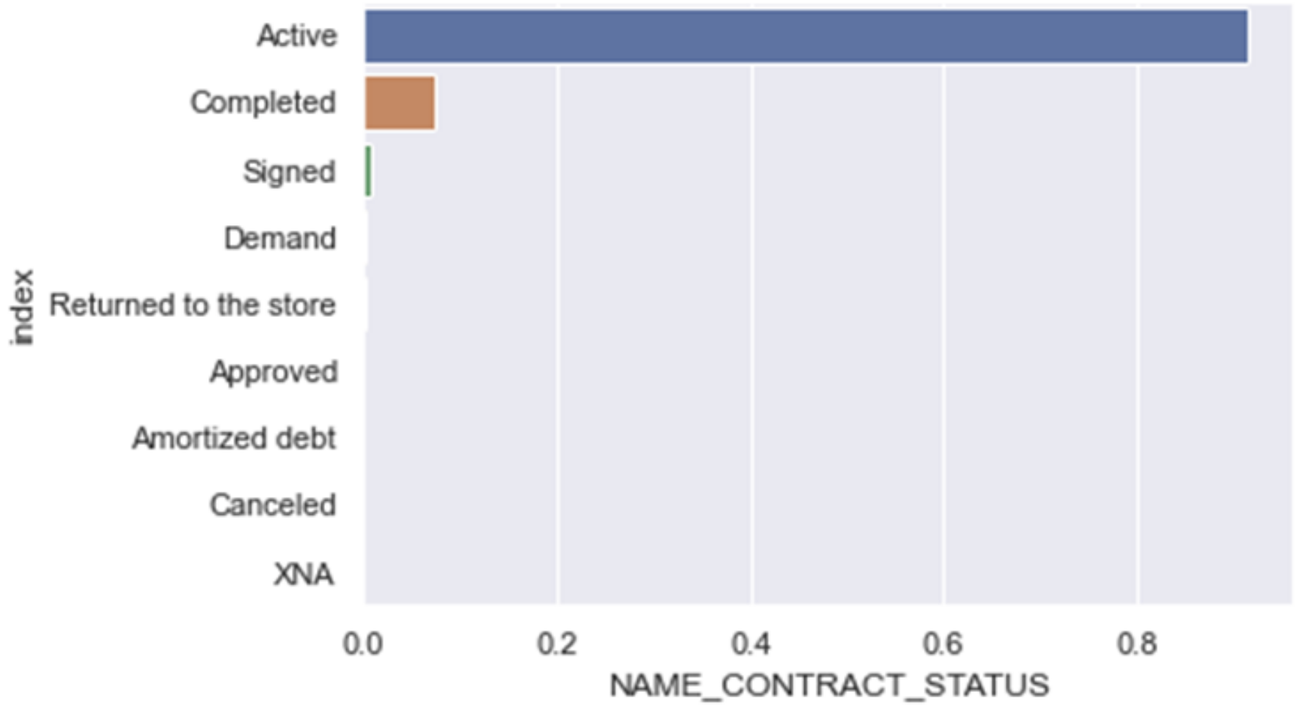
- MONTHLY BALANCE SNAPSHOTS OF PREVIOUS POS (POINT OF SALES) AND CASH LOANS THAT THE APPLICANT HAD WITH HOME CREDIT.

EXPLONATARY DATA ANALYSIS (EDA) - Distribution



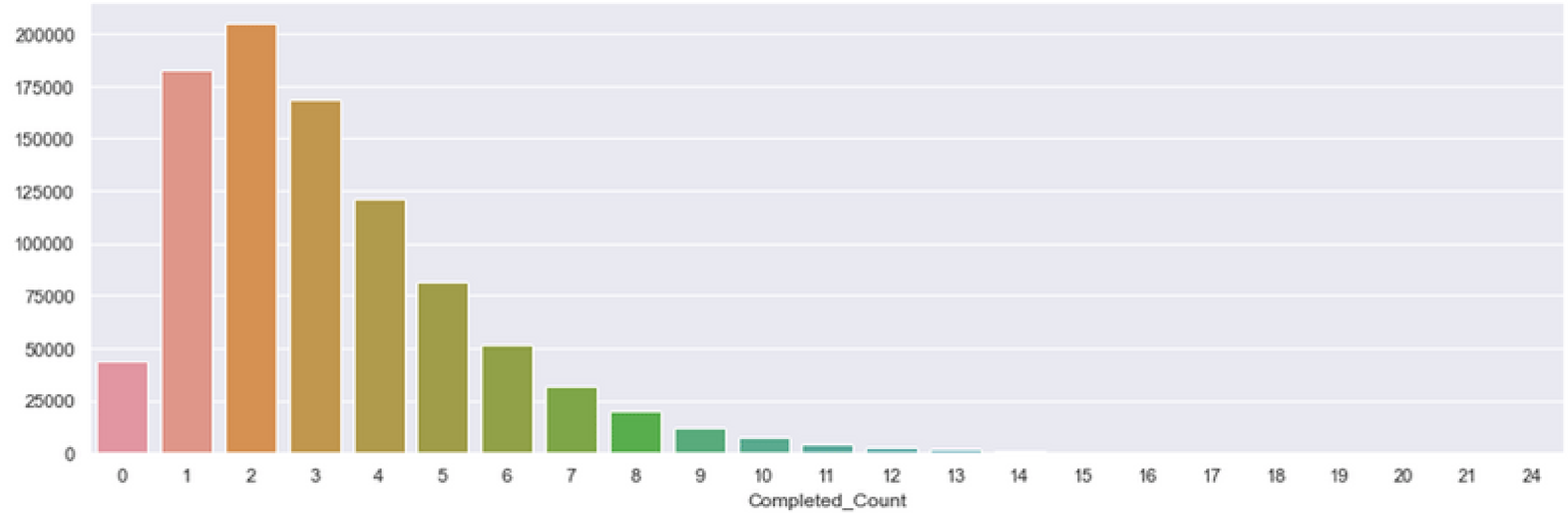


EXPLONATARY DATA ANALYSIS (EDA)



Most of the customers contract status is in active, about 10% is completed and only a small percentage of it is just signed

EXPLONATARY DATA ANALYSIS (EDA)

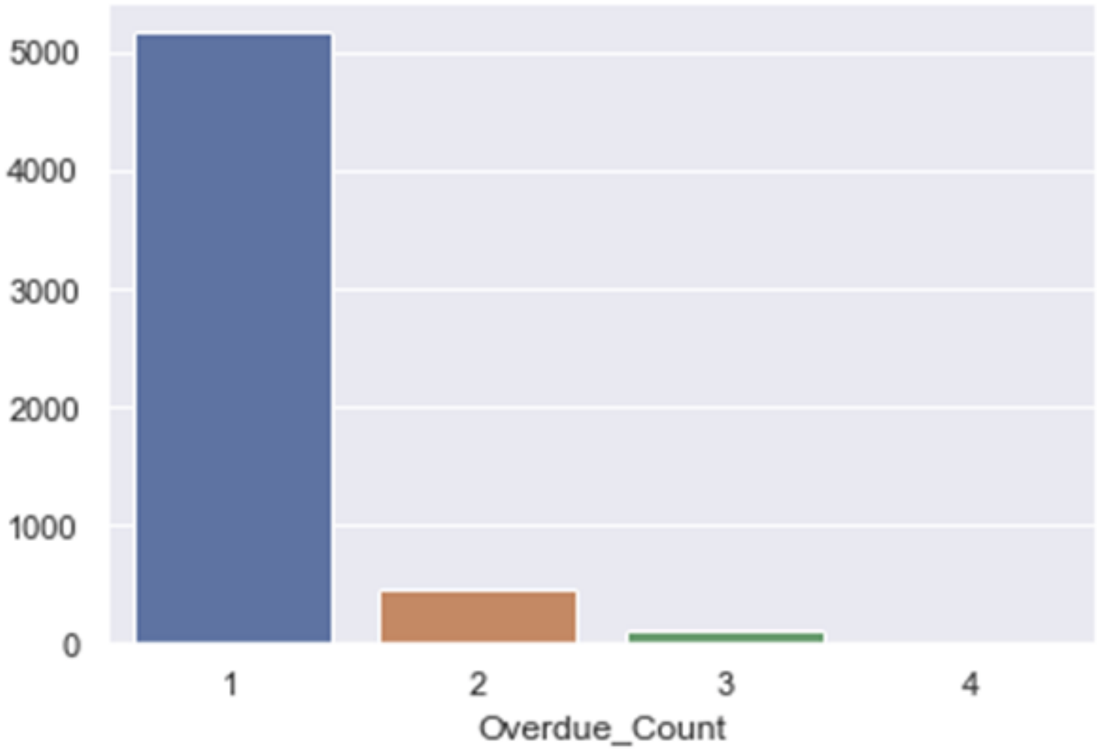


About more than 200,000 customers has completed 2 payments, more than 180,000 have completed at least 1 payment and approximately 170,000 have done 3.

Other than that, the higher the number of completed payment, the lower of customer has done it.

We also can notice that there is about almost 50,000 of customers have completed none of it

EXPLONATARY DATA ANALYSIS (EDA)

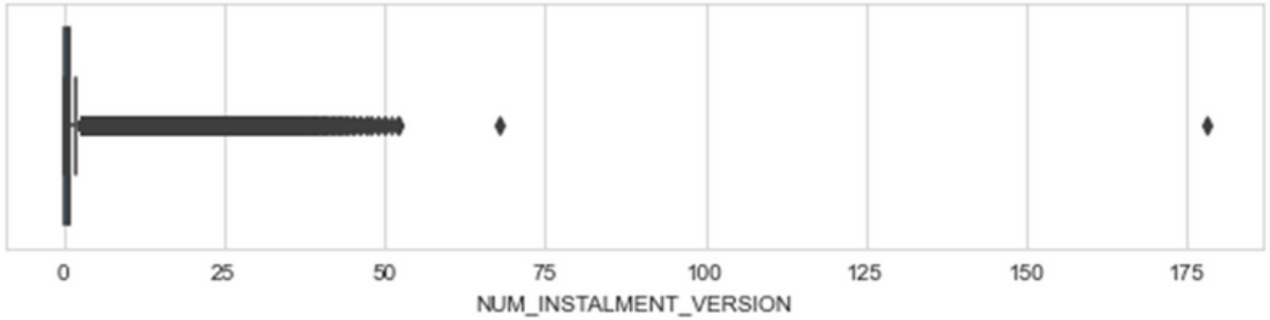


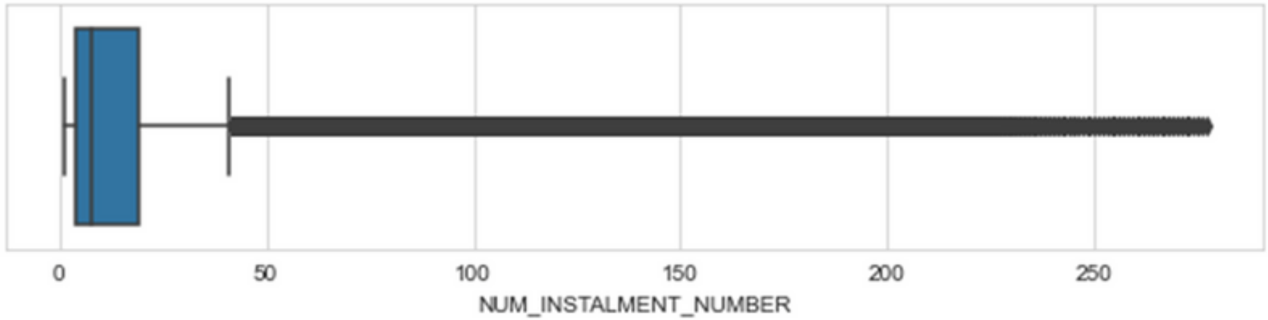
Over 5000 of customers has overdue at least once and a small amount of customers has done it more than 2 times

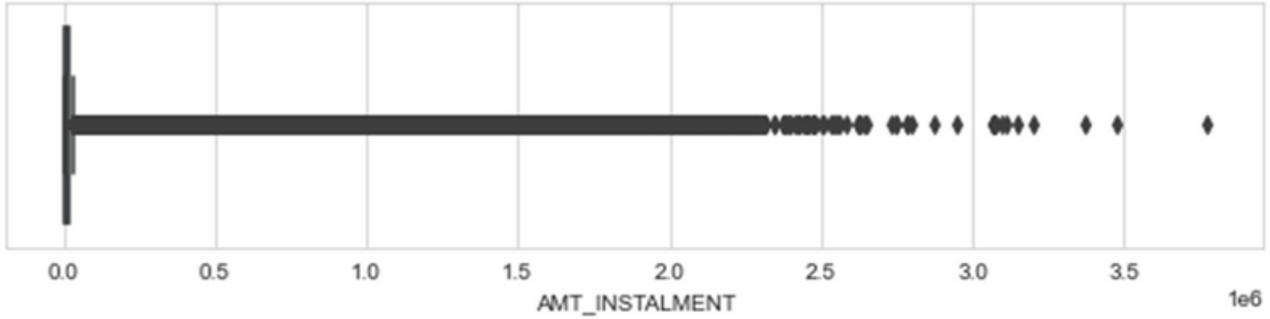
INSTALMENT PAYMENT

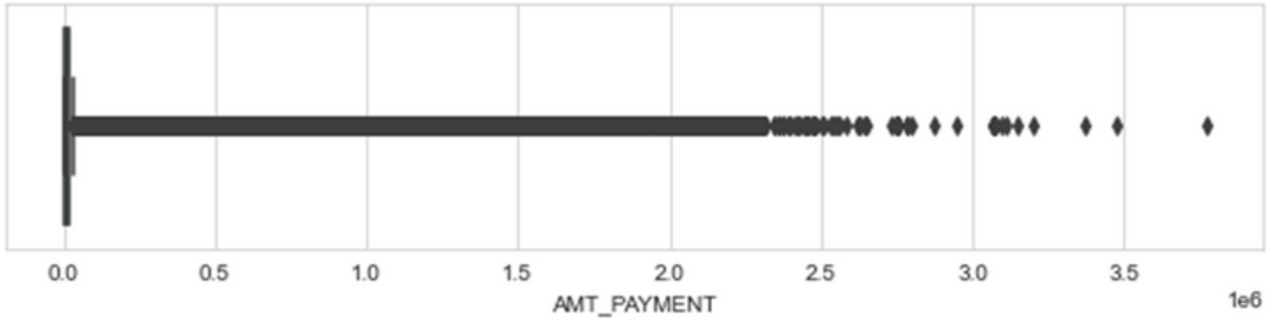
- REPAYMENT HISTORY FOR THE PREVIOUSLY DISBURSED CREDITS IN HOME CREDIT RELATED TO THE LOANS IN OUR SAMPLE.
- THERE IS ONE ROW FOR EVERY PAYMENT THAT WAS MADE PLUS ONE ROW EACH FOR MISSED PAYMENT.
- ONE ROW IS EQUIVALENT TO ONE PAYMENT OF ONE INSTALLMENT OR ONE INSTALLMENT CORRESPONDING TO ONE PAYMENT OF ONE PREVIOUS HOME CREDIT CREDIT RELATED TO LOANS IN OUR SAMPLE.

EXPLONATARY DATA ANALYSIS (EDA)-DISTRIBUTION



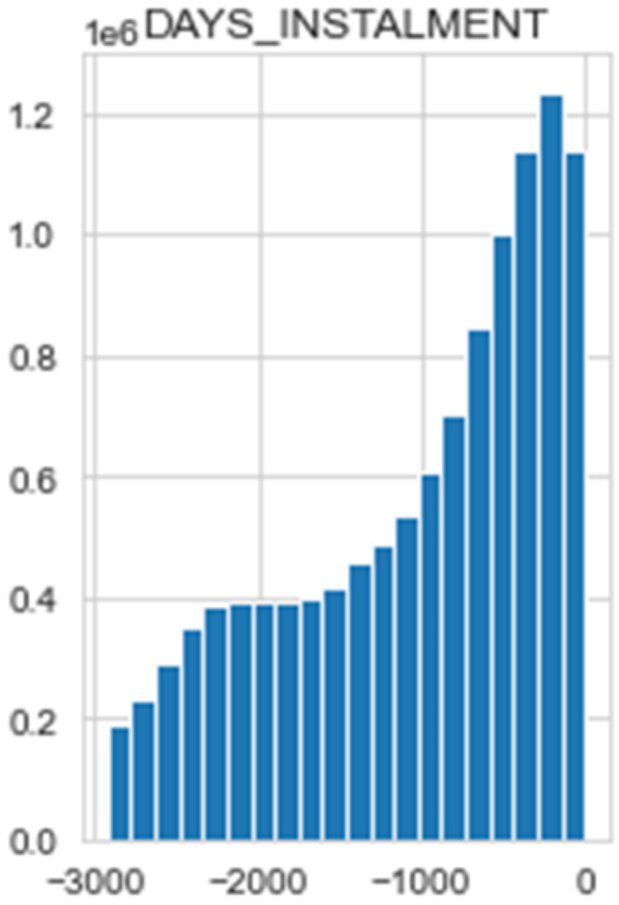


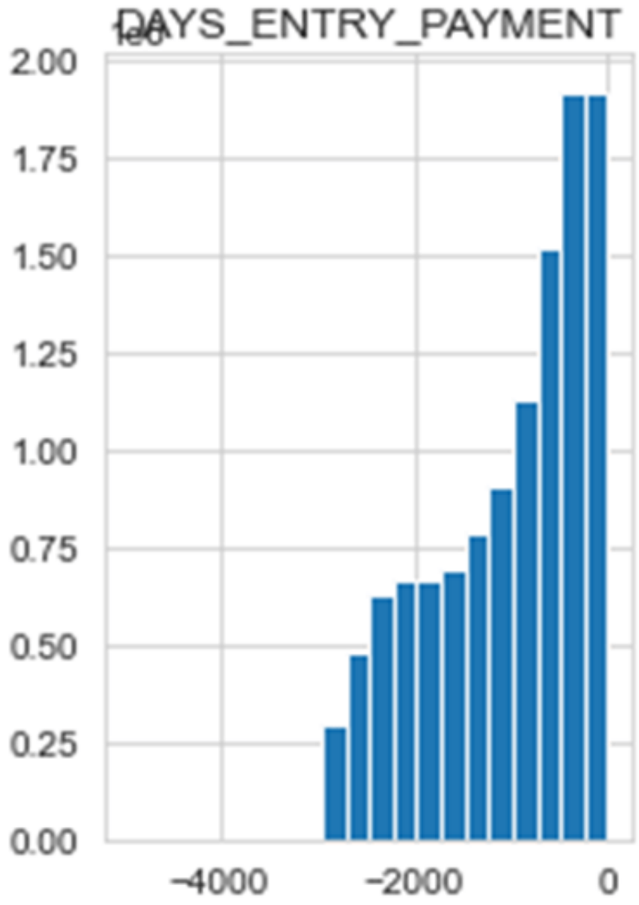




These variables contain alot of outliner

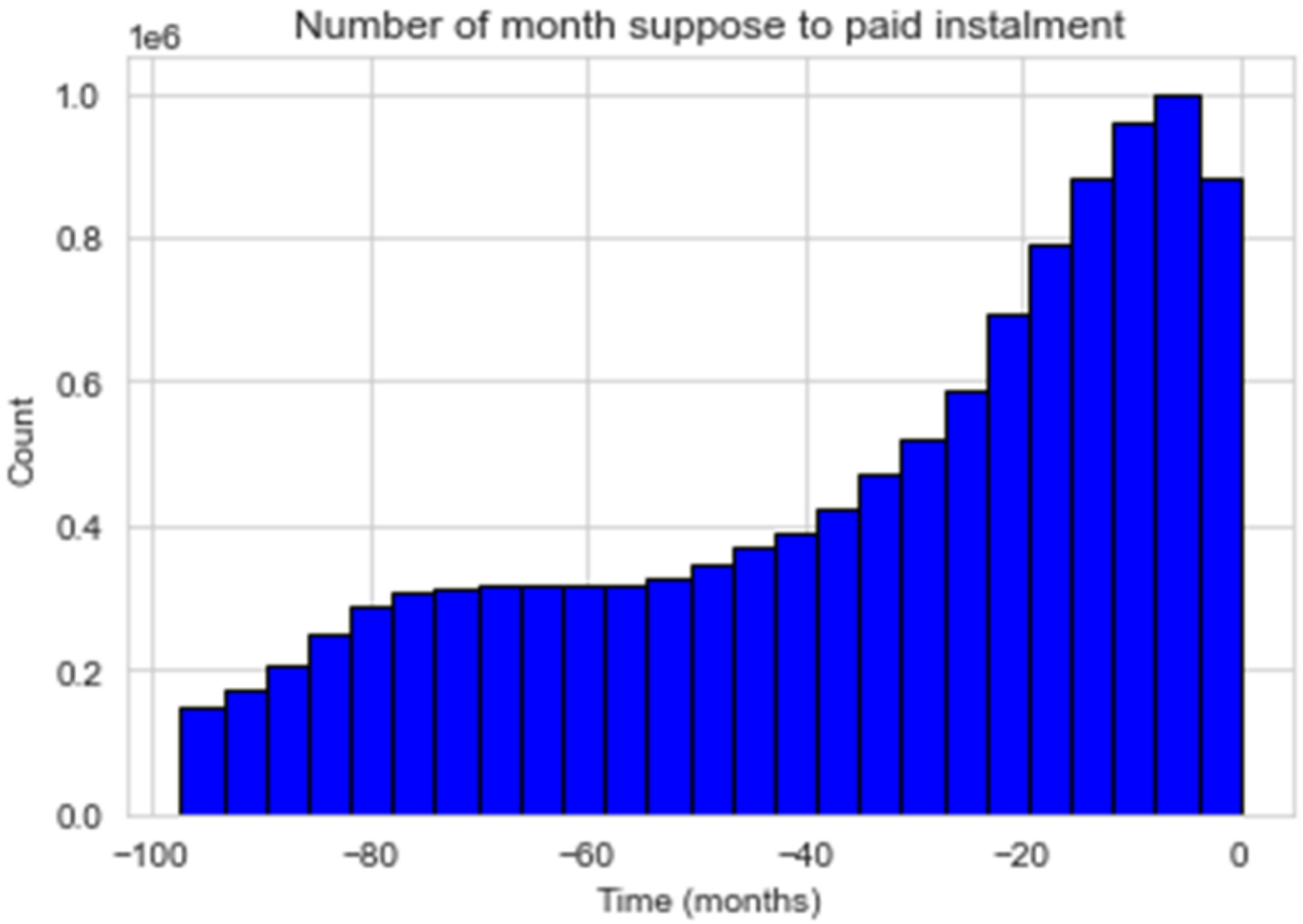
EXPLONATARY DATA ANALYSIS (EDA)





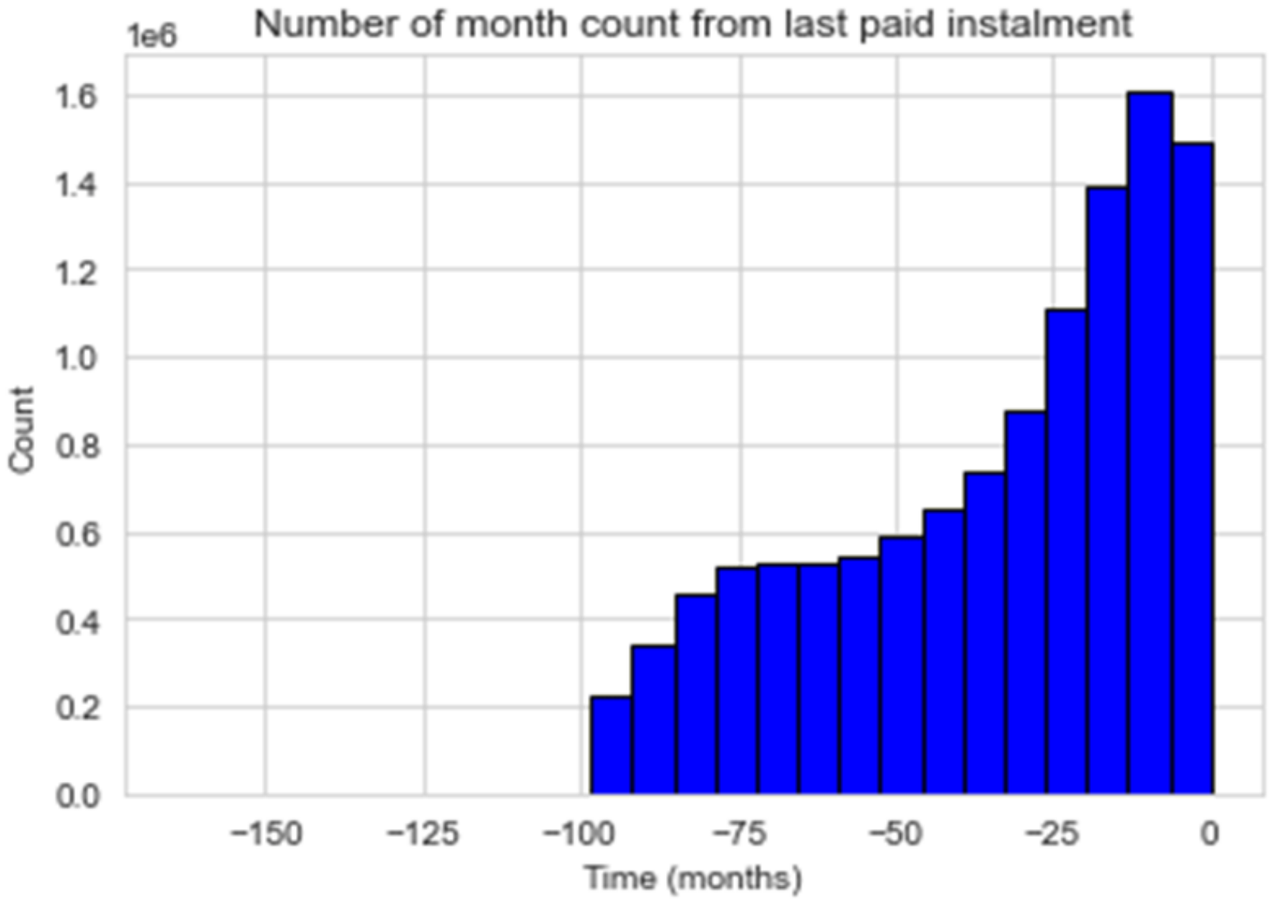
The graph show the amount of time (days) from when it acttualy paid till now in DAYS_ENTRY_PAYMENT
and the amount of time(days) from the days the instalment suppose to be paid till now in DAYS_INSTALMENT

EXPLONATARY DATA ANALYSIS (EDA)-FEATURES



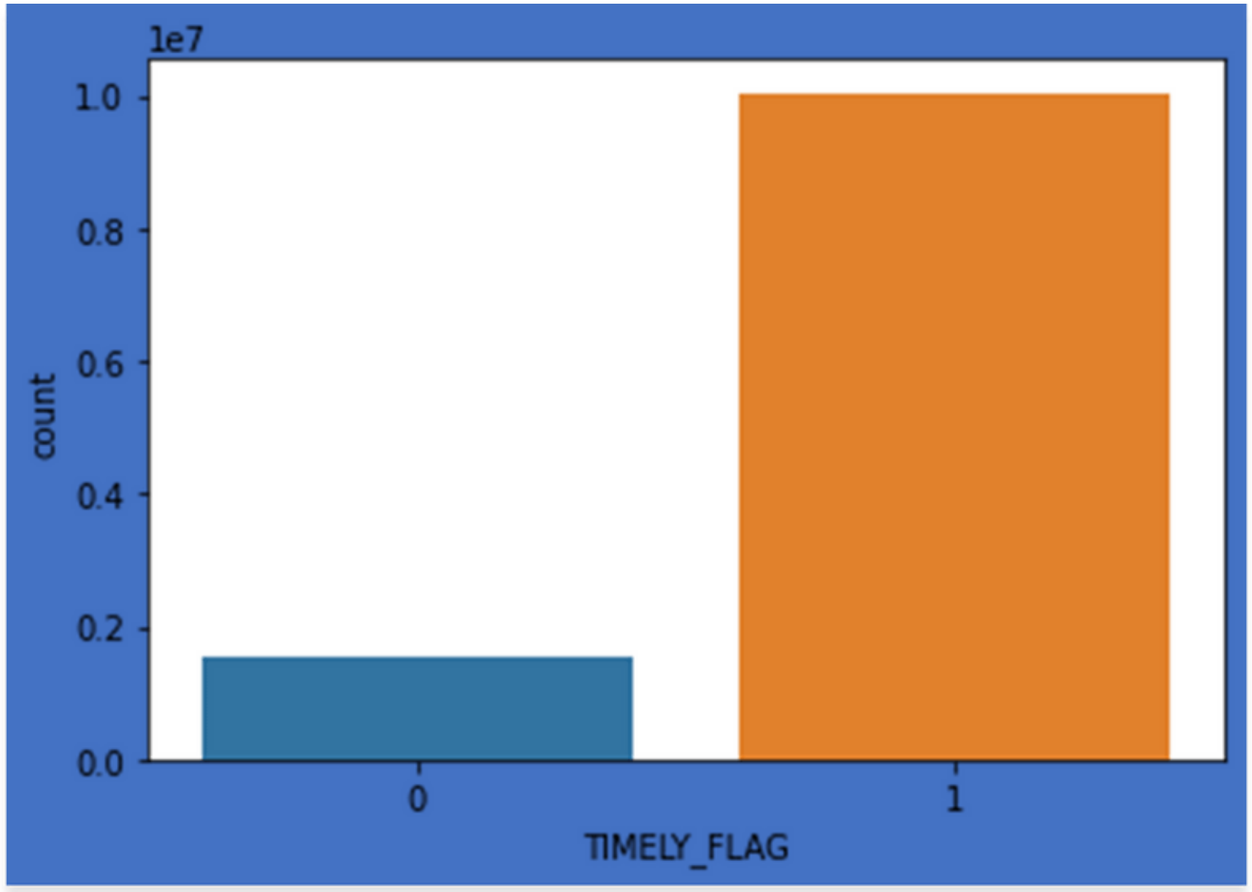
The graph shows us the time it should take for customers to pay their debts,by months, observing that most of the customers are overdue.

EXPLONATARY DATA ANALYSIS (EDA)-FEATURES



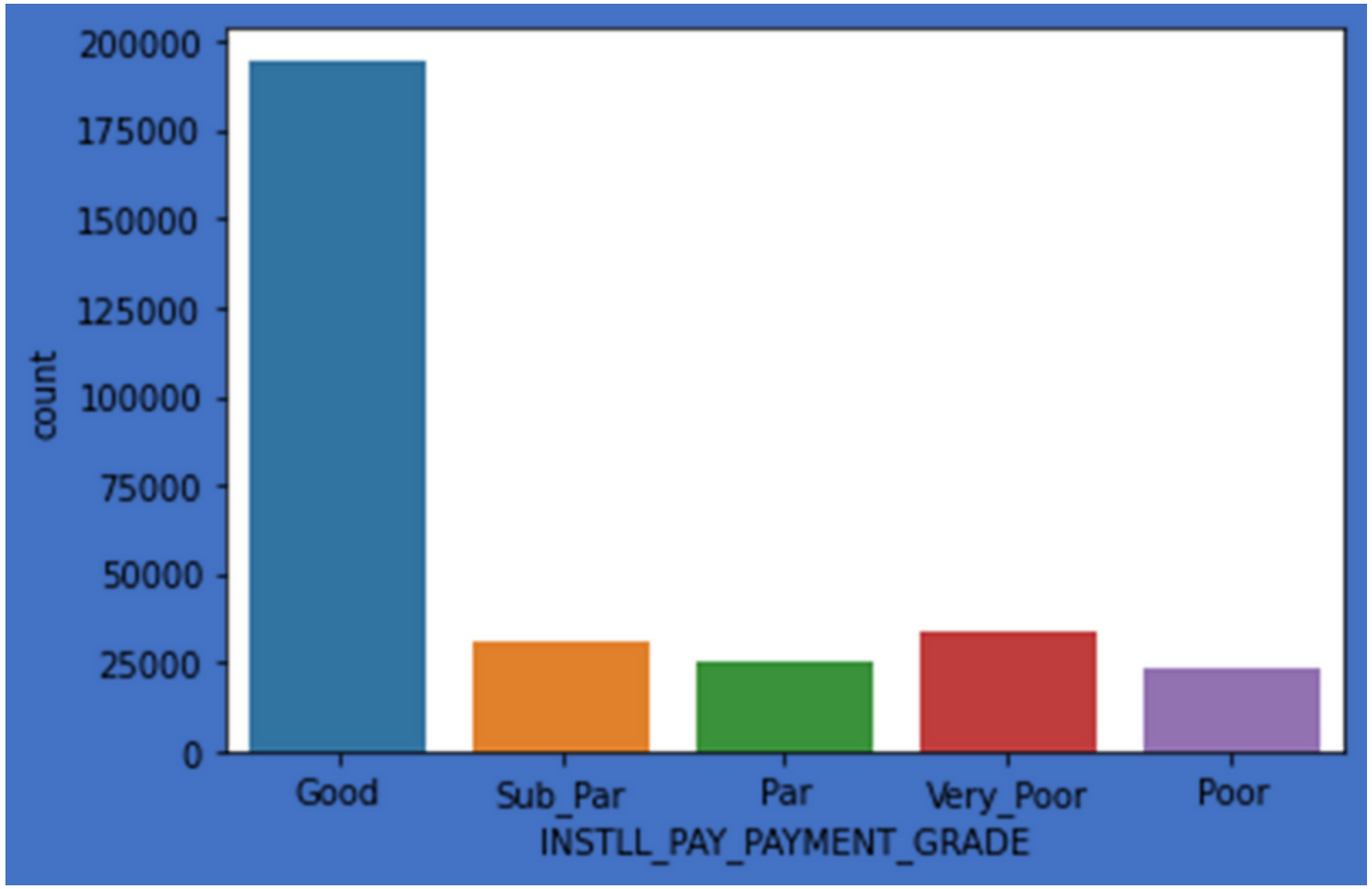
The graph shows us the time it take for customers to pay their debts,by months. Most of the cases it only took them about 25 month or less to completed it

EXPLONATARY DATA ANALYSIS (EDA)-FEATURES



The graph shows whether the customer paid the loan on time and in full or not. Looking at the graph we can see that most of the clients have done it on time and in full.

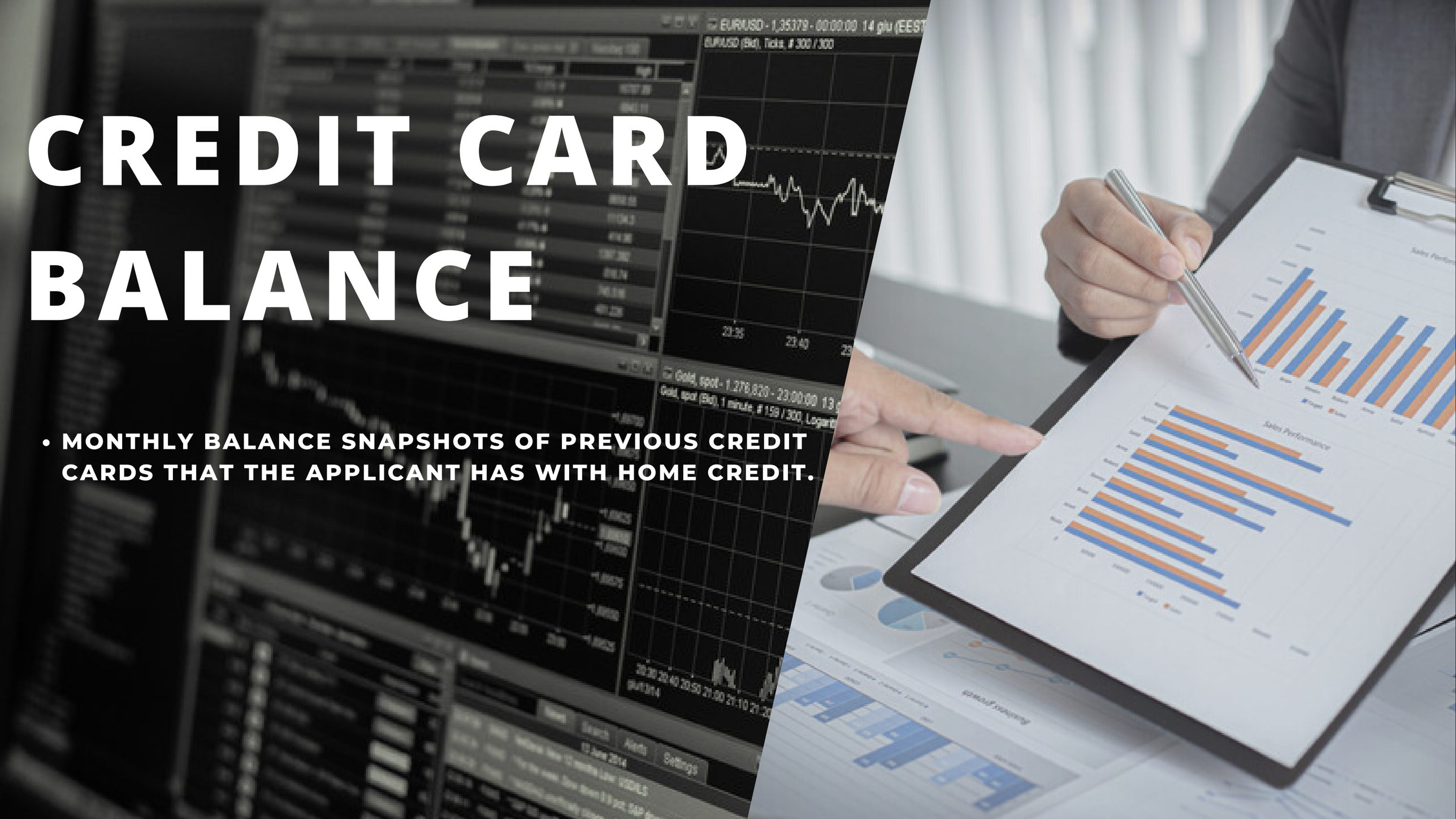
EXPLONATARY DATA ANALYSIS (EDA)-FEATURES



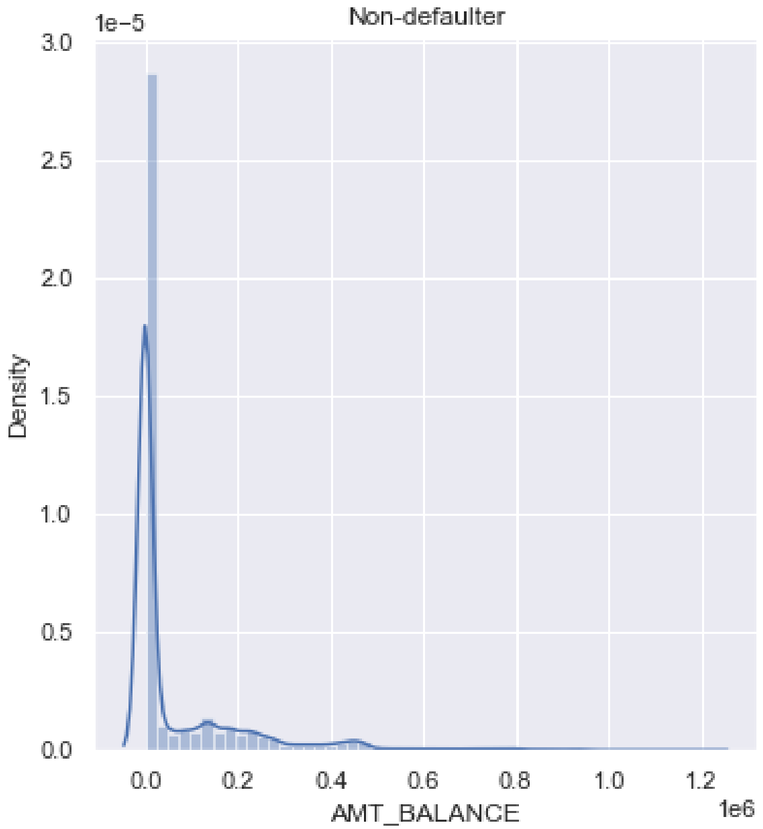
The graph above shows the division of customers based on their ability to pay on time and in full

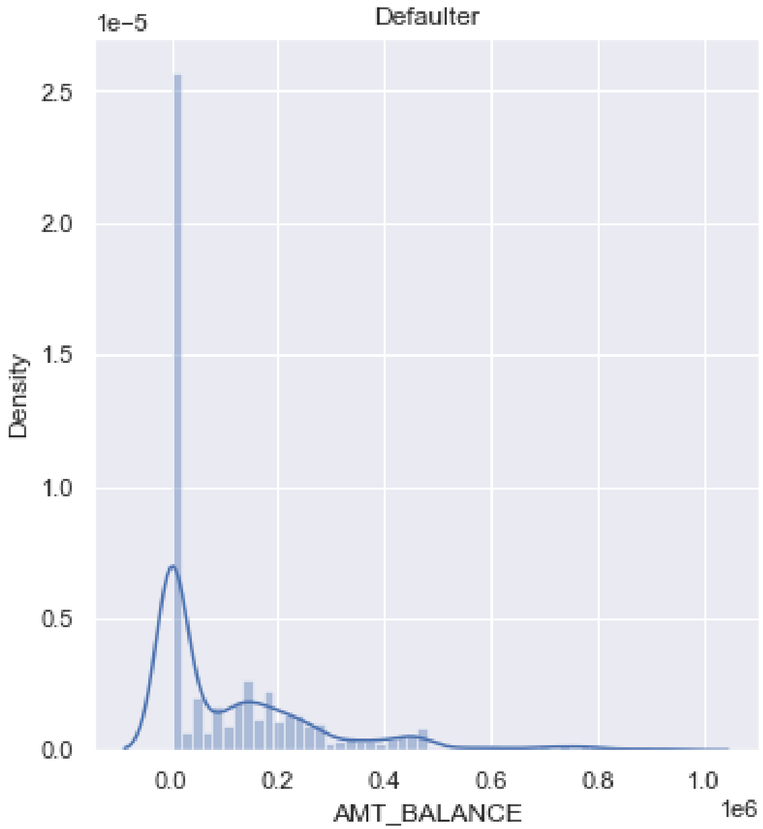
CREDIT CARD BALANCE

- MONTHLY BALANCE SNAPSHOTS OF PREVIOUS CREDIT CARDS THAT THE APPLICANT HAS WITH HOME CREDIT.



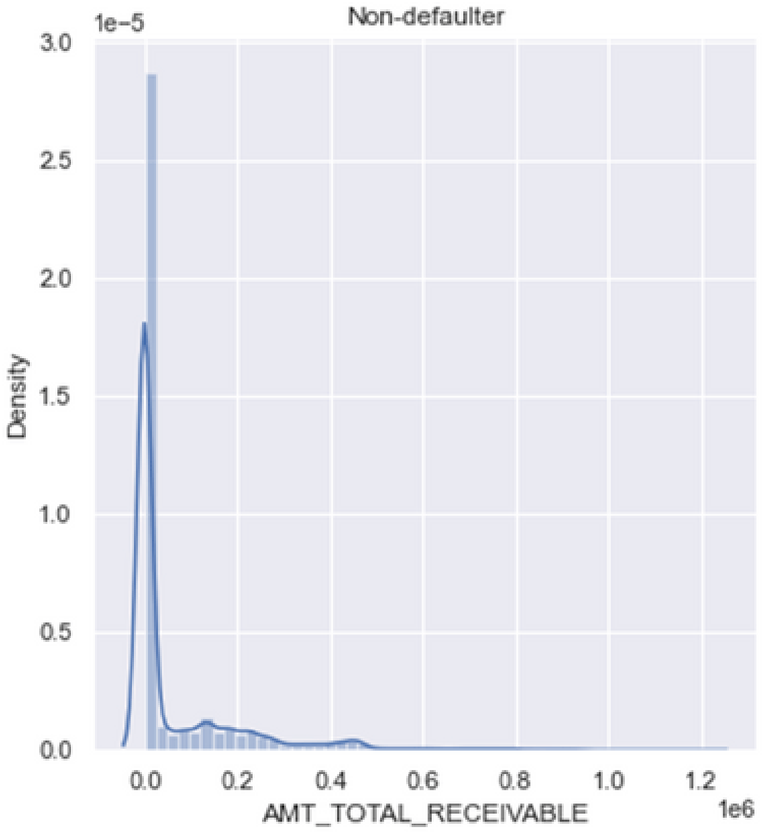
EXPLONATARY DATA ANALYSIS (EDA)

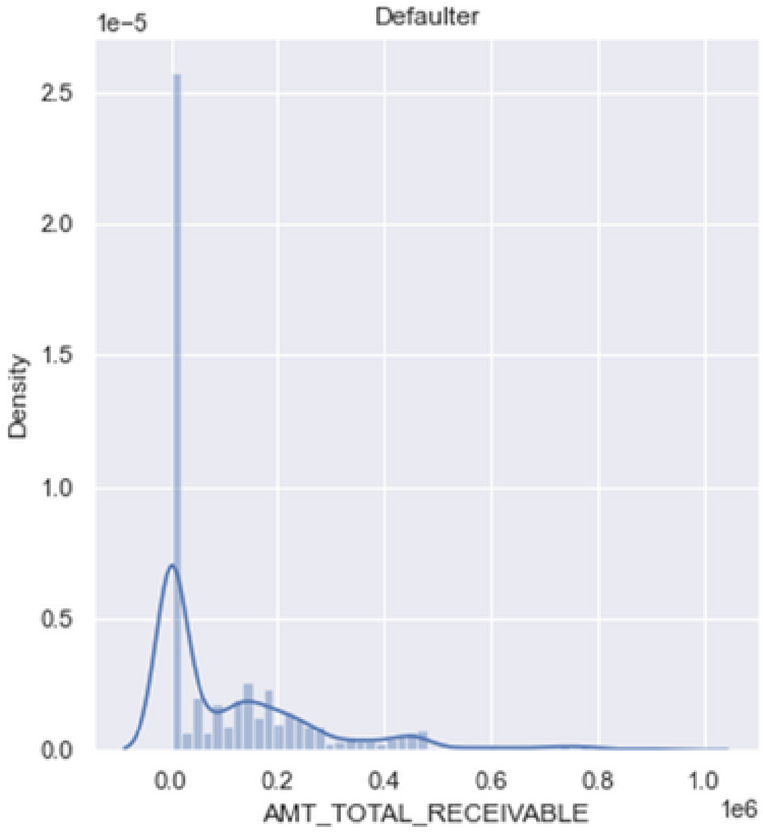




The graph shows the distribution of the data AMT_BALANCE between Defaulter and Non-Defaulter.
Oberserving the graph we can see that the balance of Defaulter is slightly higher than that of non-defaulter

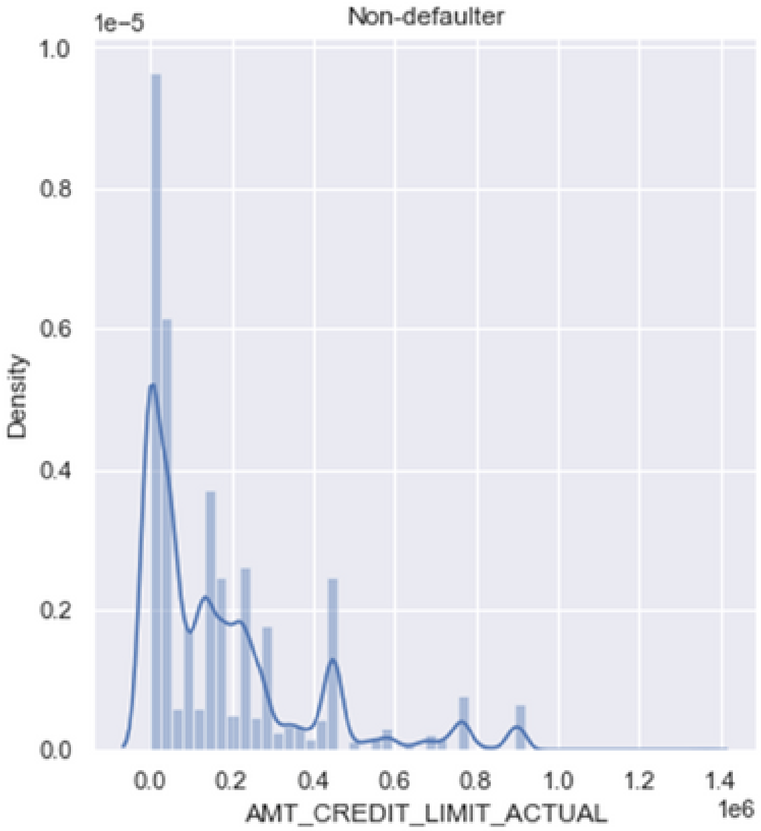
EXPLONATARY DATA ANALYSIS (EDA)

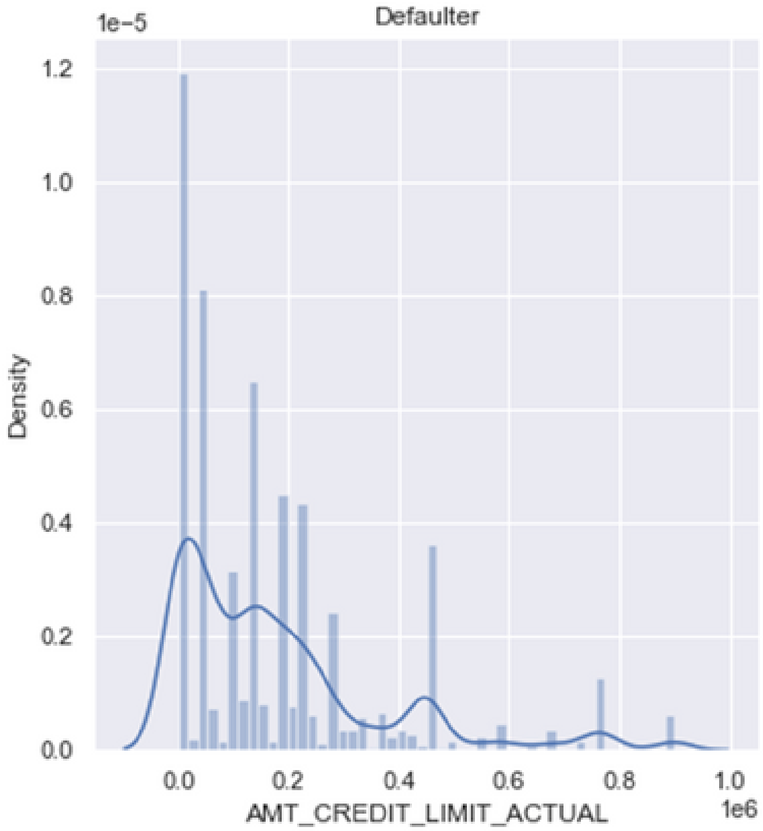




The graph shows the distribution of the data AMT_TOTAL_RECEIVABLE between Defaulter and Non-Defaulter.
Oberserving the graph we can see that the total amount of receivable of Defaulter is slightly higher than that of non-defaulter

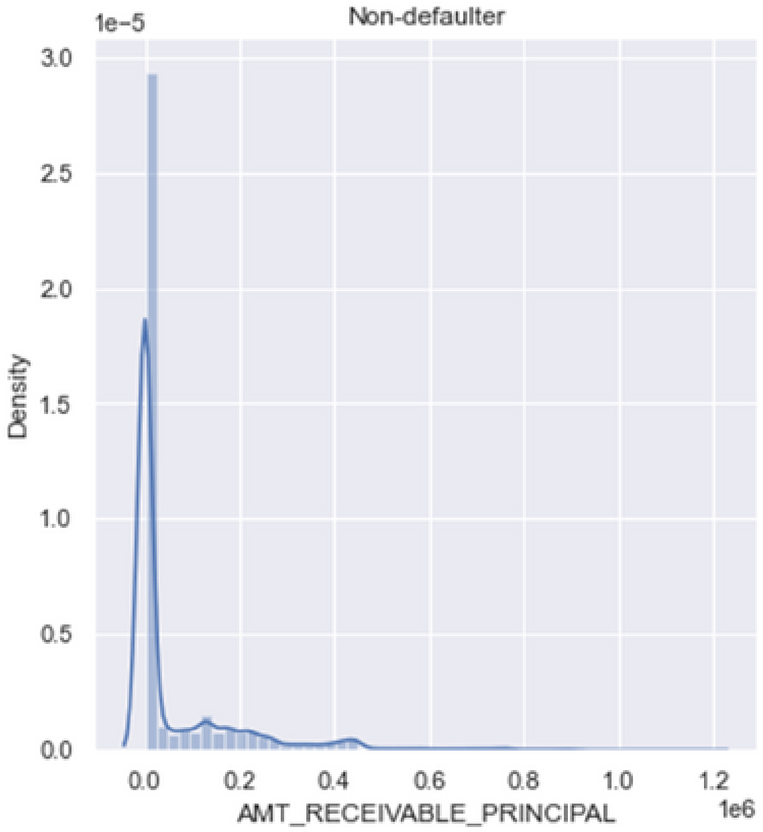
EXPLONATARY DATA ANALYSIS (EDA)

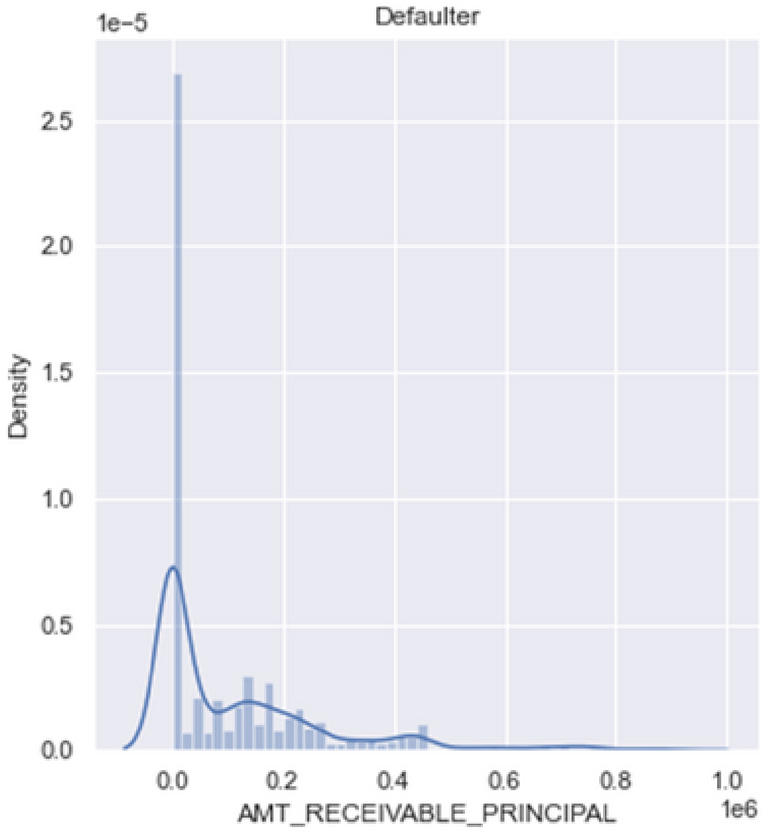




The graph shows the distribution of the data AMT_CREDIT_LIMIT_ACTUAL between Defaulter and Non-Defaulter. Oberserving the graph we can see that there is more non-defaulter than defaulter but the defaulter credit limit is higher than non-defaulter

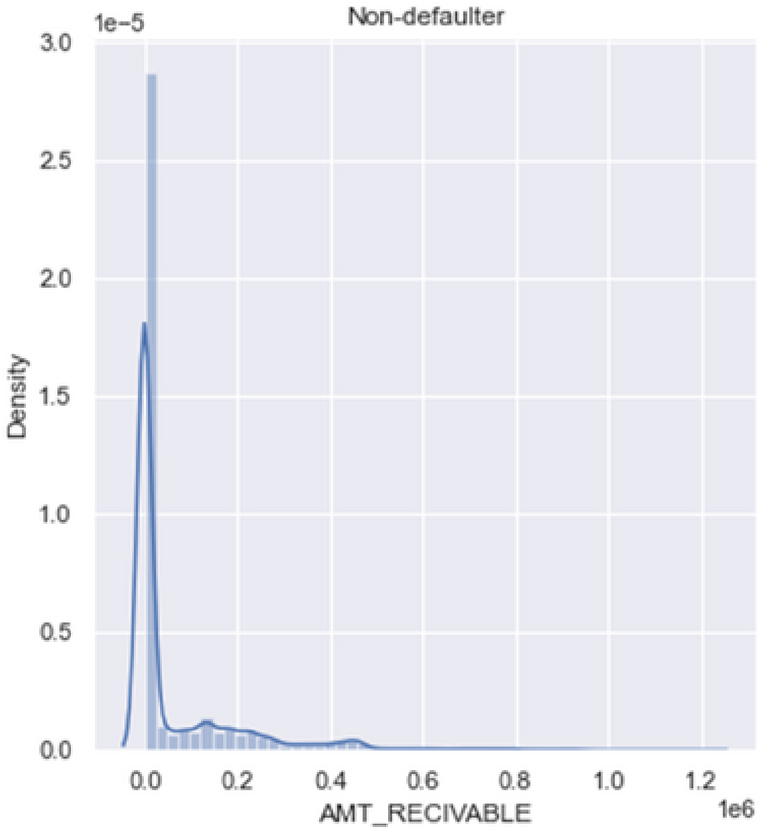
EXPLONATARY DATA ANALYSIS (EDA)

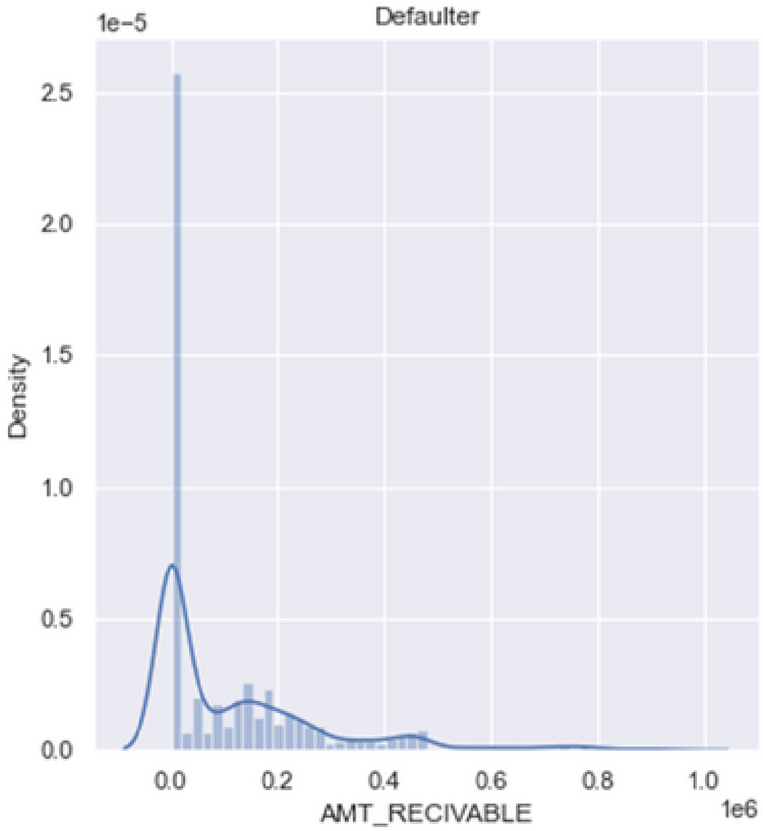




The graph shows the distribution of the data AMT_RECEIVABLE_PRINCIPAL between Defaulter and Non-Defaulter.

EXPLONATARY DATA ANALYSIS (EDA)





The graph shows the distribution of the data AMT_RECEIVABLE between Defaulter and Non-Defaulter.

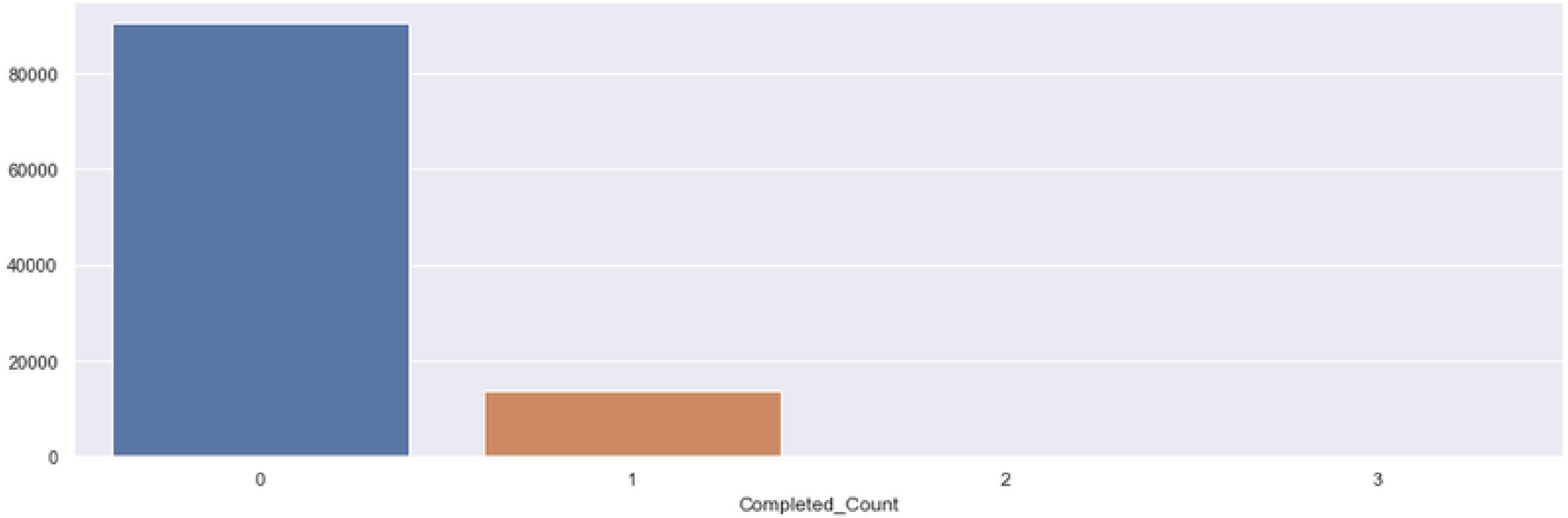
EXPLONATARY DATA ANALYSIS (EDA)

The graph shows the status of customer's loan. Looking at it, we can see that most of the contract is in active and a fair amount of them has been completed and some small number is signed and in demand.

1 - Defaulter

0 - Non-Defaulter

EXPLONATARY DATA ANALYSIS (EDA)

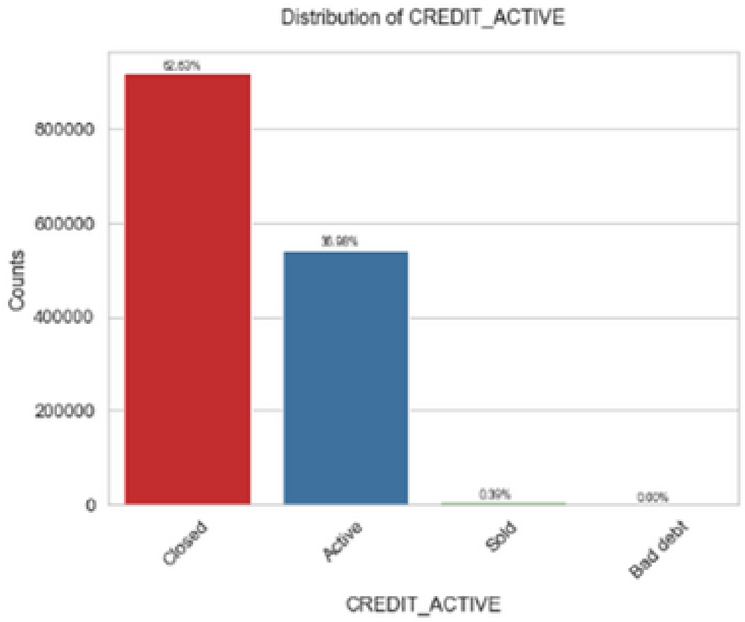


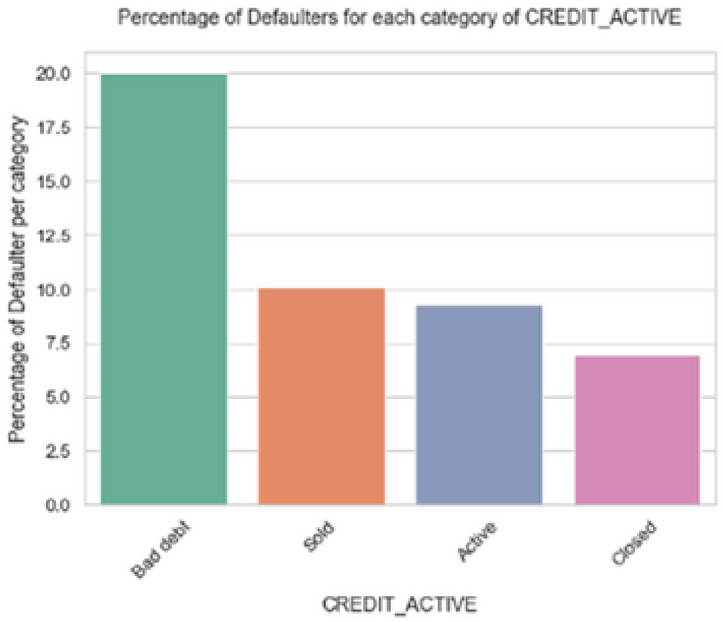
The graph show how much contract have been completed per customer, noticing that most of them have completed none, a small number has done it once and a really few of them have done it more than one.

BUREAU.CSV DATA

- All client’s previous credits provided by other financial institutions that were reported to Credit Bureau (for clients who have a loan in our sample).
- For every loan in our sample, there are as many rows as number of credits the client had in Credit Bureau before the application date.

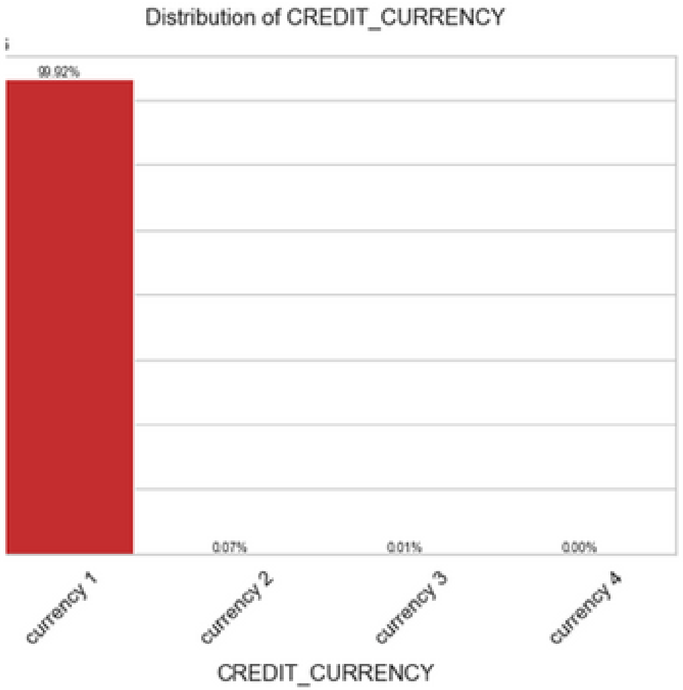
EXPLONATARY DATA ANALYSIS (EDA)

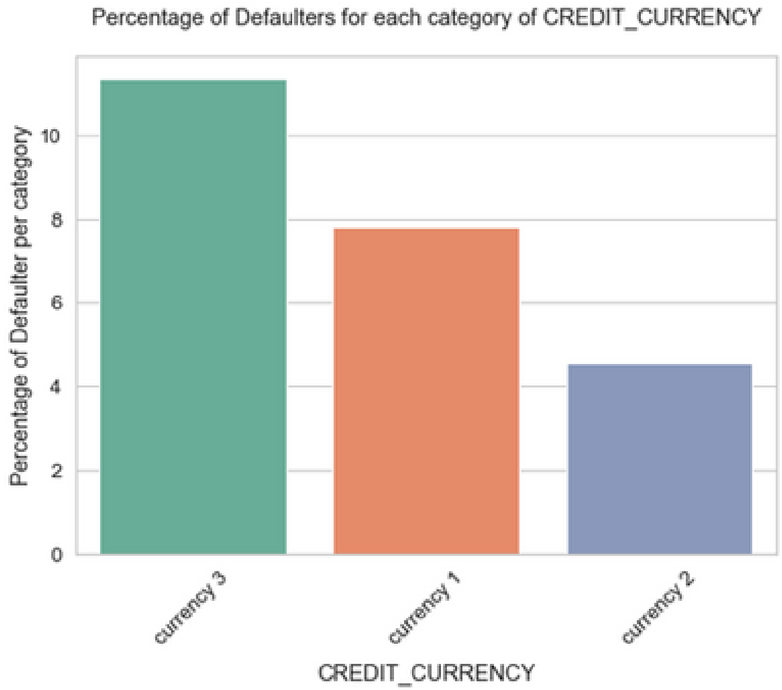




As we can easily see, the main status of the credits will be Closed with 1079273 debts accounting for 62.9%. There are more than 630000 debts still active, accounting for about 36%. The remaining only about 0.4% of debts were sold and <0.1% were bad debts.

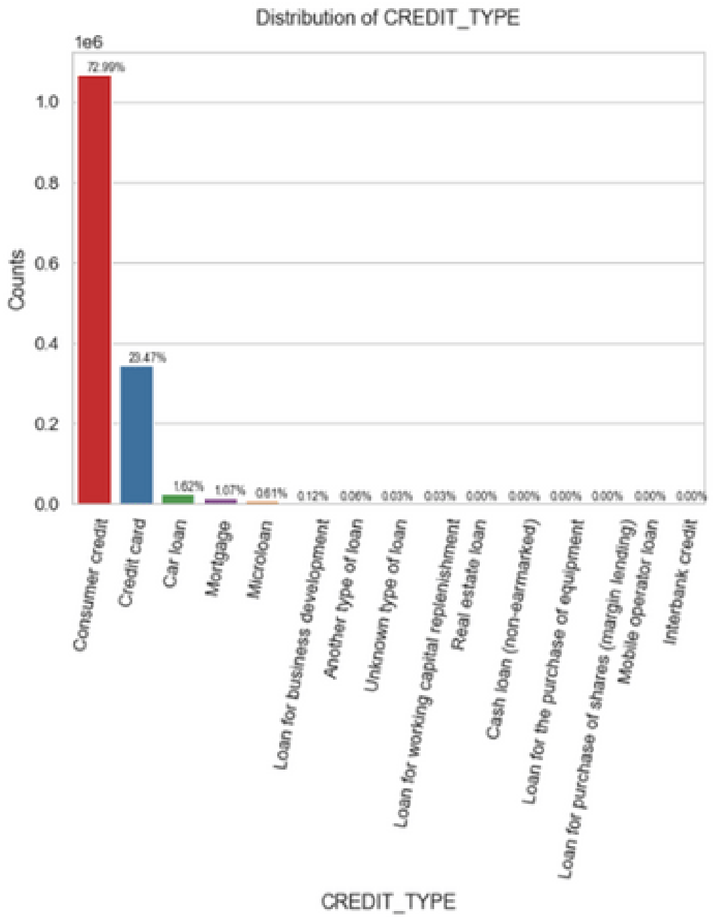
EXPLONATARY DATA ANALYSIS (EDA)

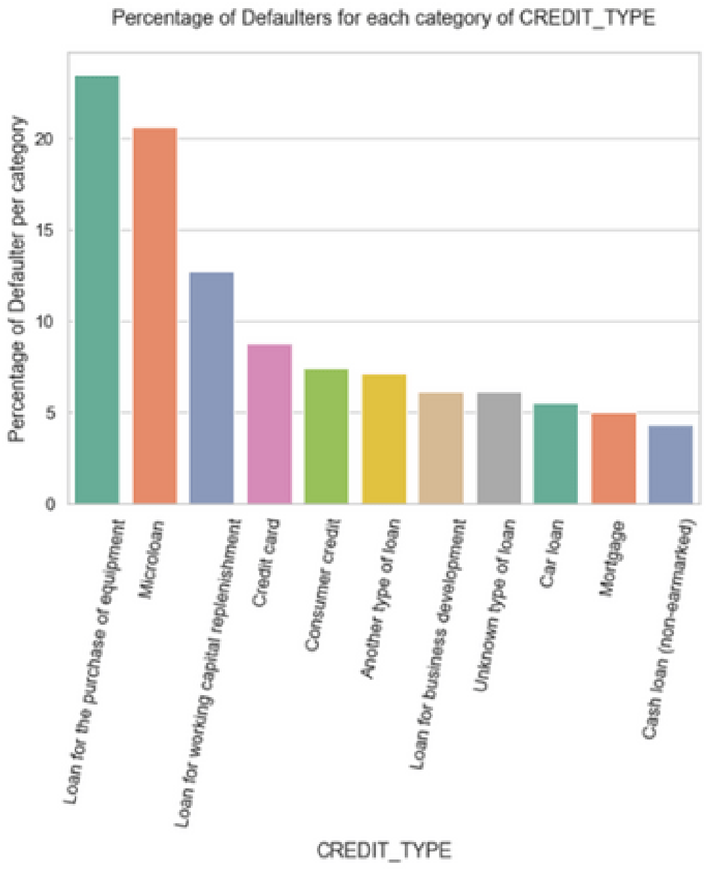




Among the 4 currencies used to make credits, the number 1 dominates with more than 1715000 accounts used this currency, accounting for 99%. Other currencies are not so significant

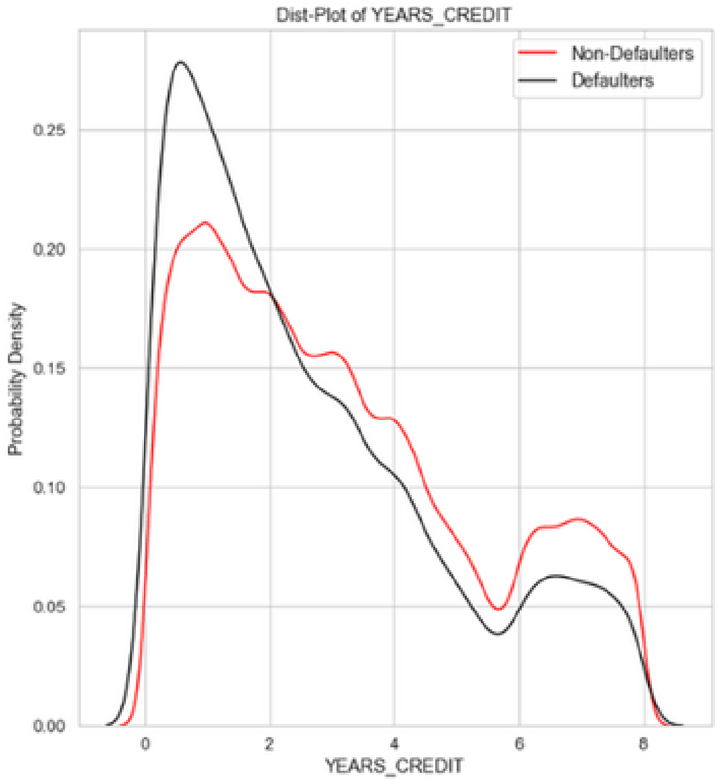
EXPLONATARY DATA ANALYSIS (EDA)

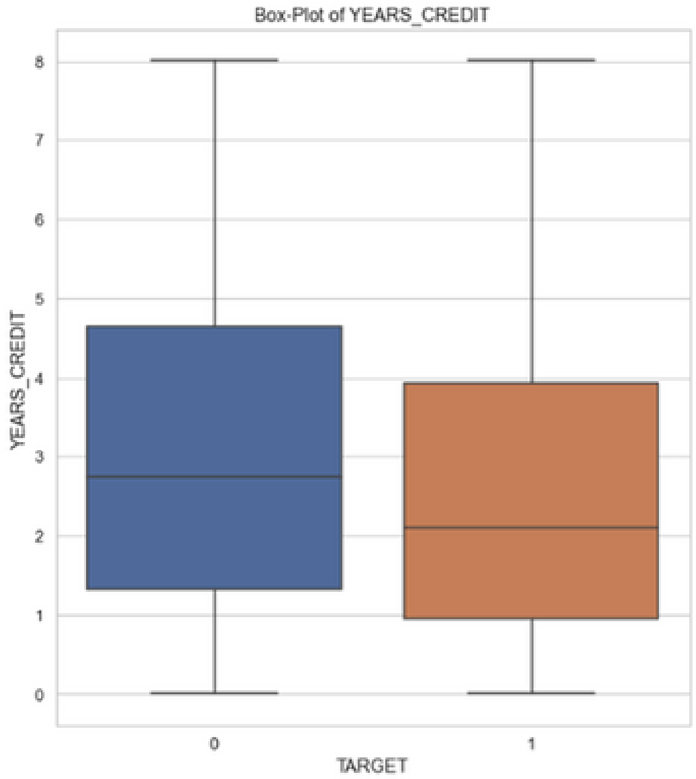




There are up to 15 types of credit such as consumer credit, credit card, car loan, mortgage credit, etc. But the type of credit that customers make the most is credit. Consumer use with more than 1251000 credits is this type accounting for nearly 80%. In addition, credit cards accounted for 23.4%, car loans accounted for 1.6%.

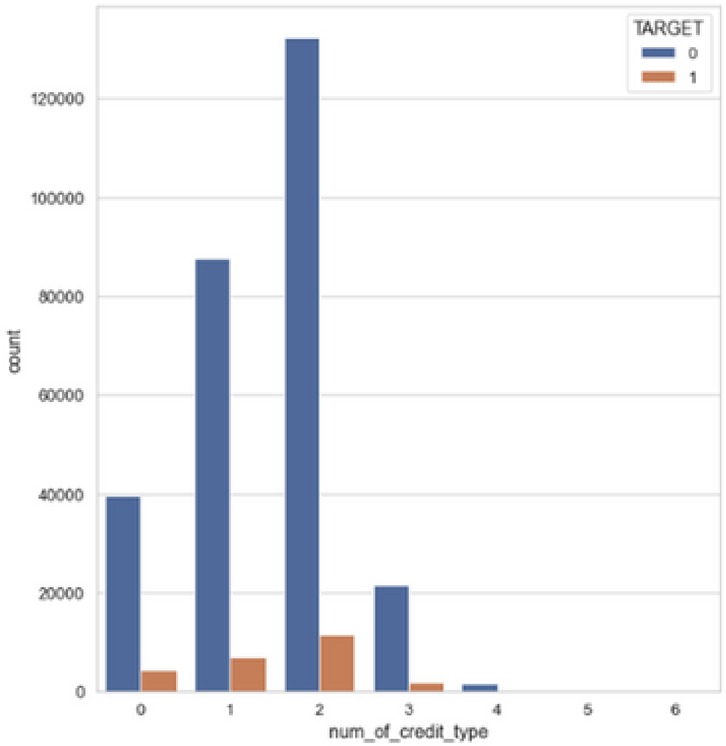
EXPLONATARY DATA ANALYSIS (EDA)

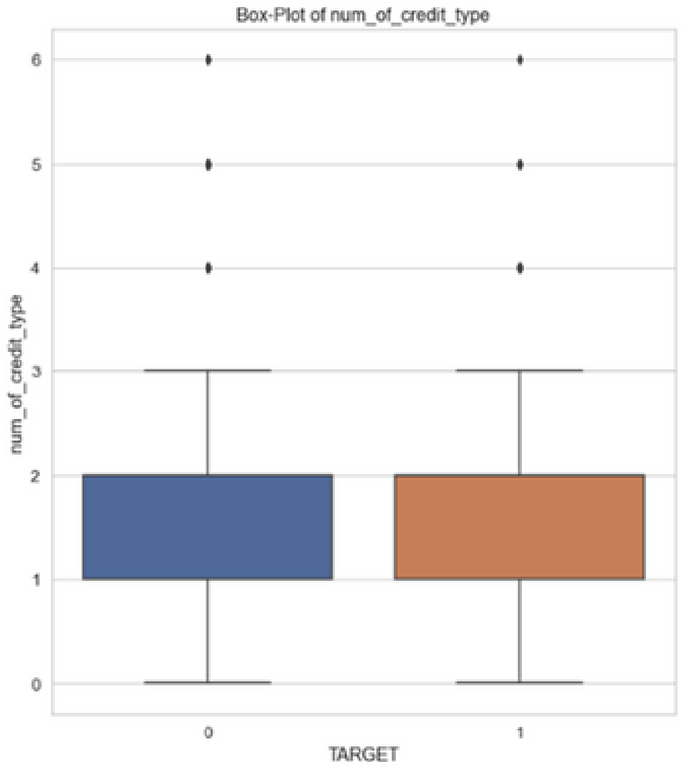




- From this plot, we observe that the Non-Defaulters usually have longer periods of Credits as compared to Defaulters.
- We can also easily see that customers mainly have loans before a year with respect to the time of having a credit in Home Credit
- The Defaulters have a higher Peak in PDF in lower YEARS_CREDIT range of values.

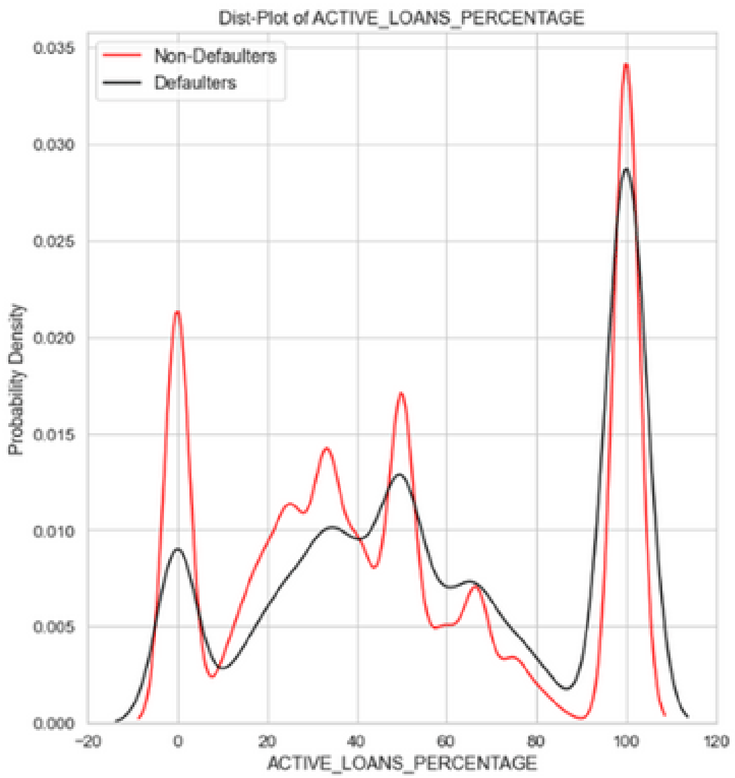
EXPLONATARY DATA ANALYSIS (EDA)

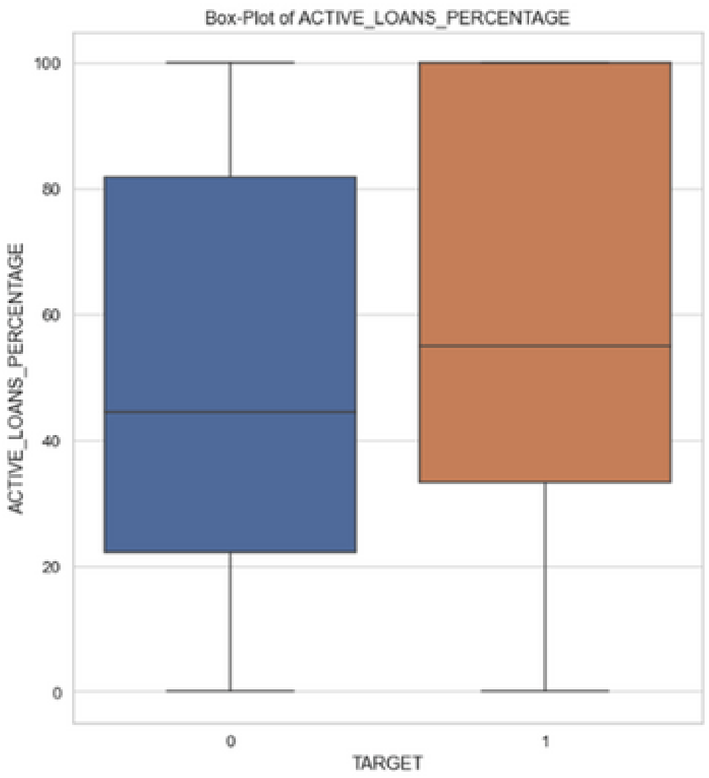




- In general, most customers will have 2 types of loans before applying for Home Credit
- We can see that the distribution of the number of types of credit in the total debt of customers is quite similar.

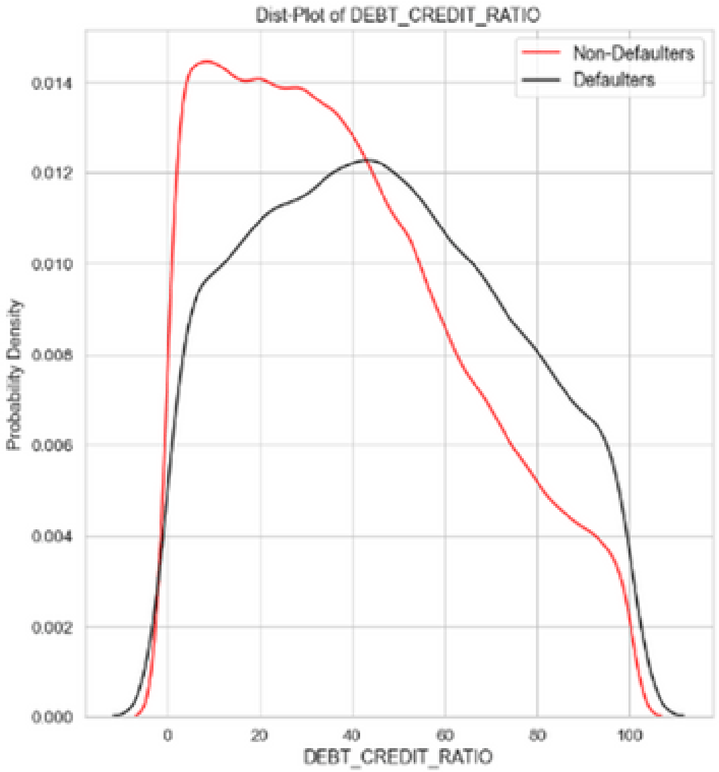
EXPLONATARY DATA ANALYSIS (EDA)

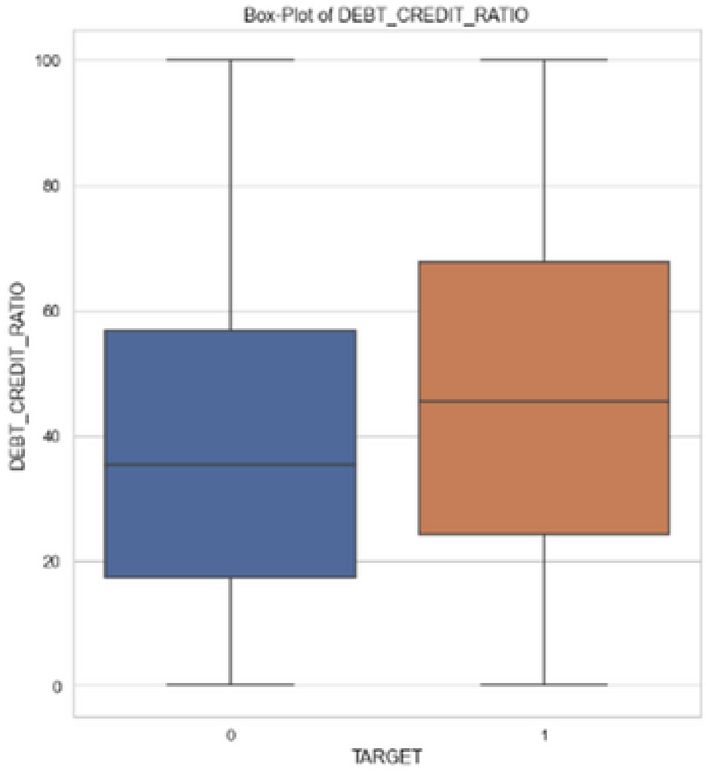




- We can clearly see that defaulters often have a higher percentage of their active debt than non-defaulters
- Customer with high percentage of active loan may have potential is a defaulter

EXPLONATARY DATA ANALYSIS (EDA)



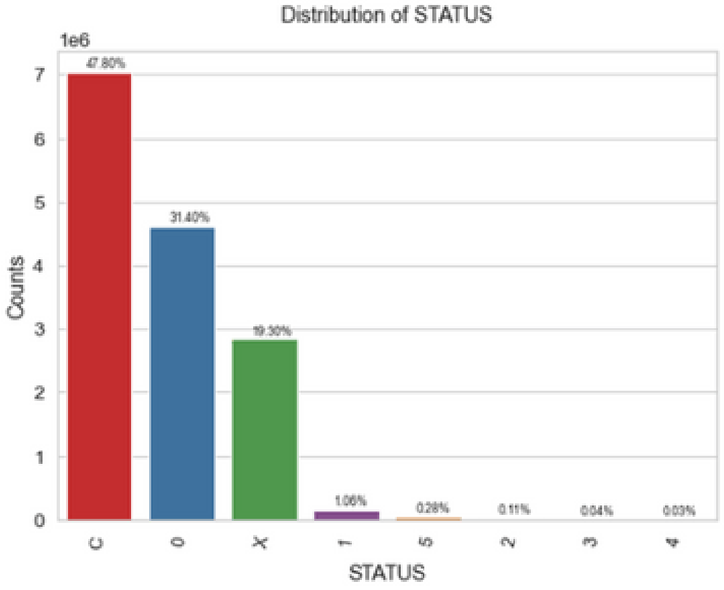


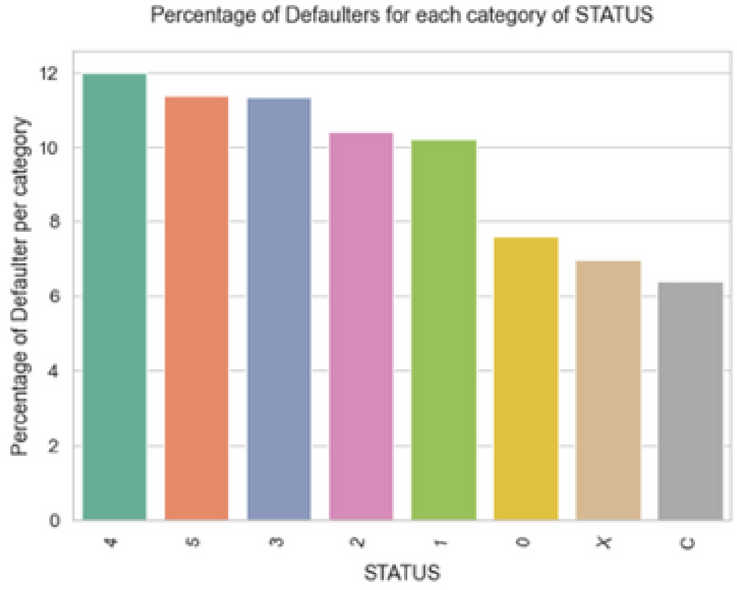
- We can see that defaulters have a wide range of ratio of total debt to total credit percentage which distributes from 20-90%.
- Non defaulter have a smaller range which distributes from 10-40%
- Customer with a high ratio of total debt to total credit may have potential is a defaulter

BUREAU_BALANCE DATA

- Monthly balances of previous credits in Credit Bureau.

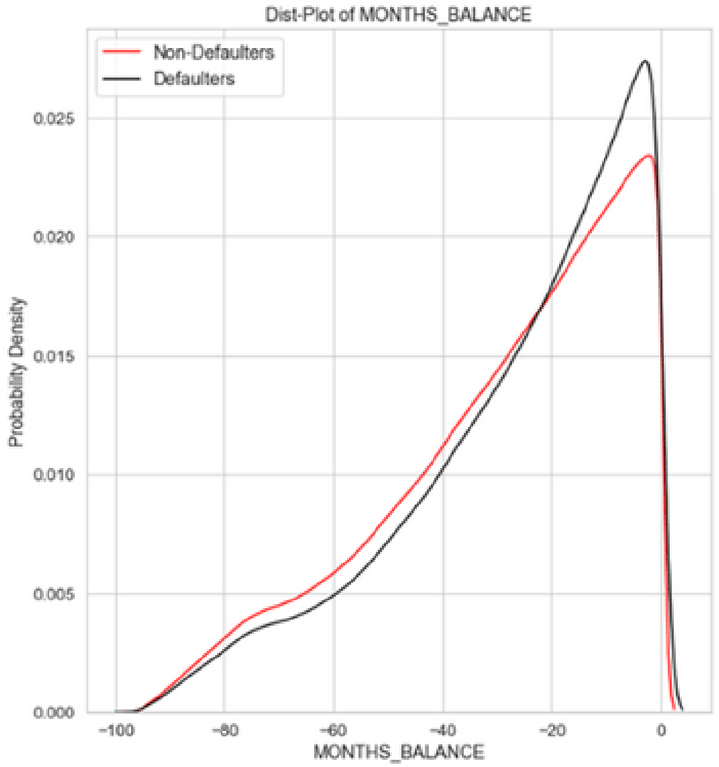
EXPLONATARY DATA ANALYSIS (EDA)

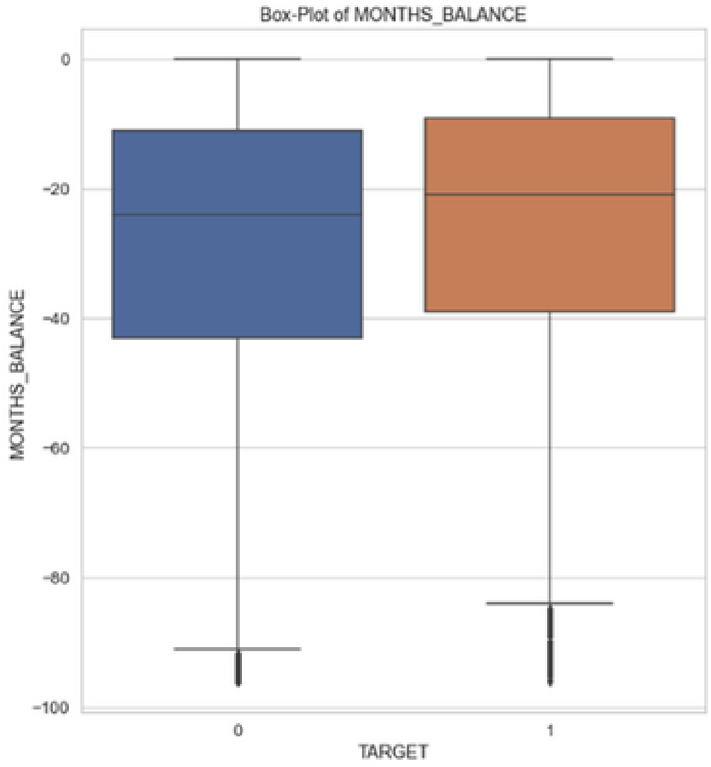




- There are many types of status of liabilities such as C is closed, X is unspecified status, etc.
- Then we can see that up to 13646993 outstanding debts are closed, nearly 7500000 debts are unpaid according to the arranged schedule and are subject to late fees and at the end of the day there are more than 5800000 the debt is still unidentified status

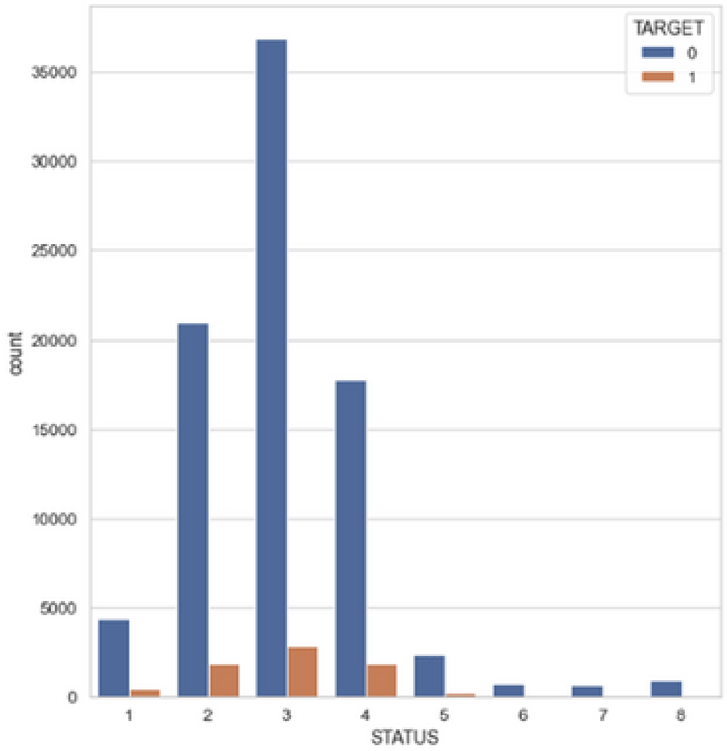
EXPLONATARY DATA ANALYSIS (EDA)

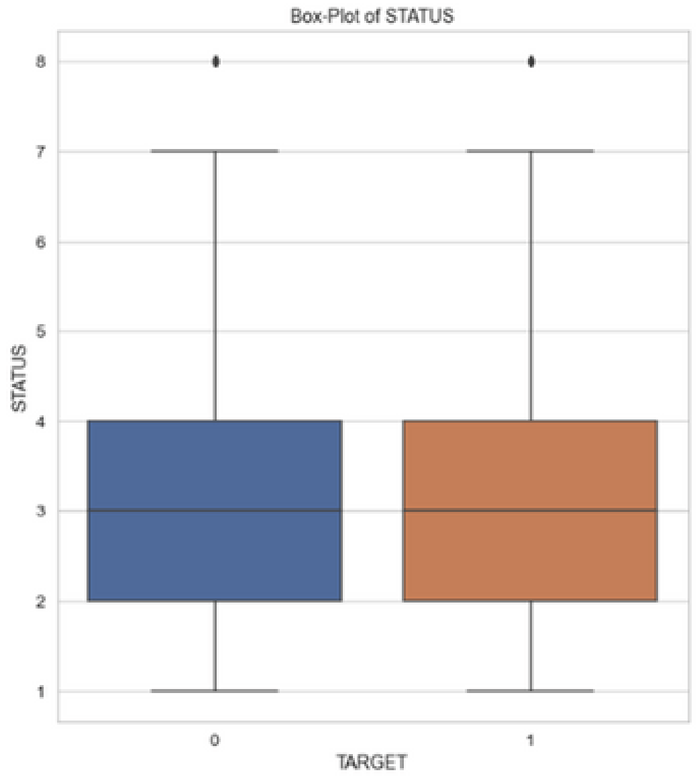




We can clearly see that the distribution is mainly in the range from -20 to 0. This means that the debt status will still be updated for at least 20 days of that month.

EXPLONATARY DATA ANALYSIS (EDA)



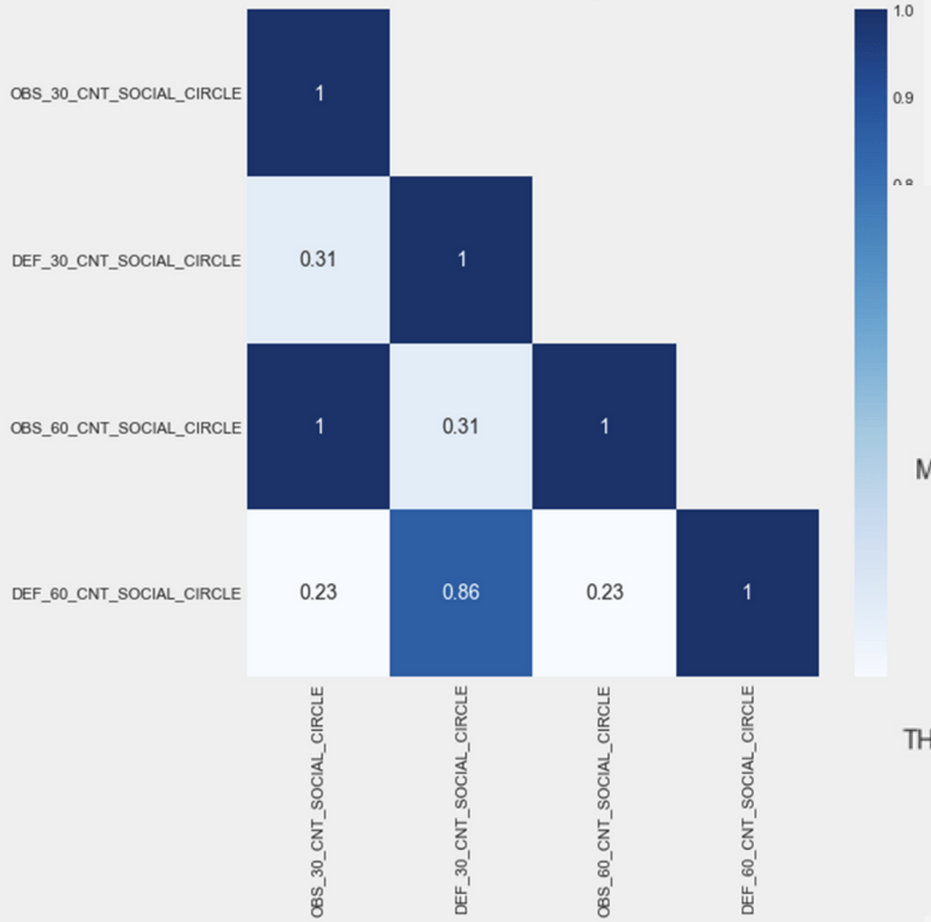


- We can easily see that in the total number of debts of each customer, most of those debts will have 3 statuses.
- Since the distribution is quite similar, we can assume that the factor of the number of positions of loans per customer has no effect on whether the customer has difficulty in repaying the loan or not.

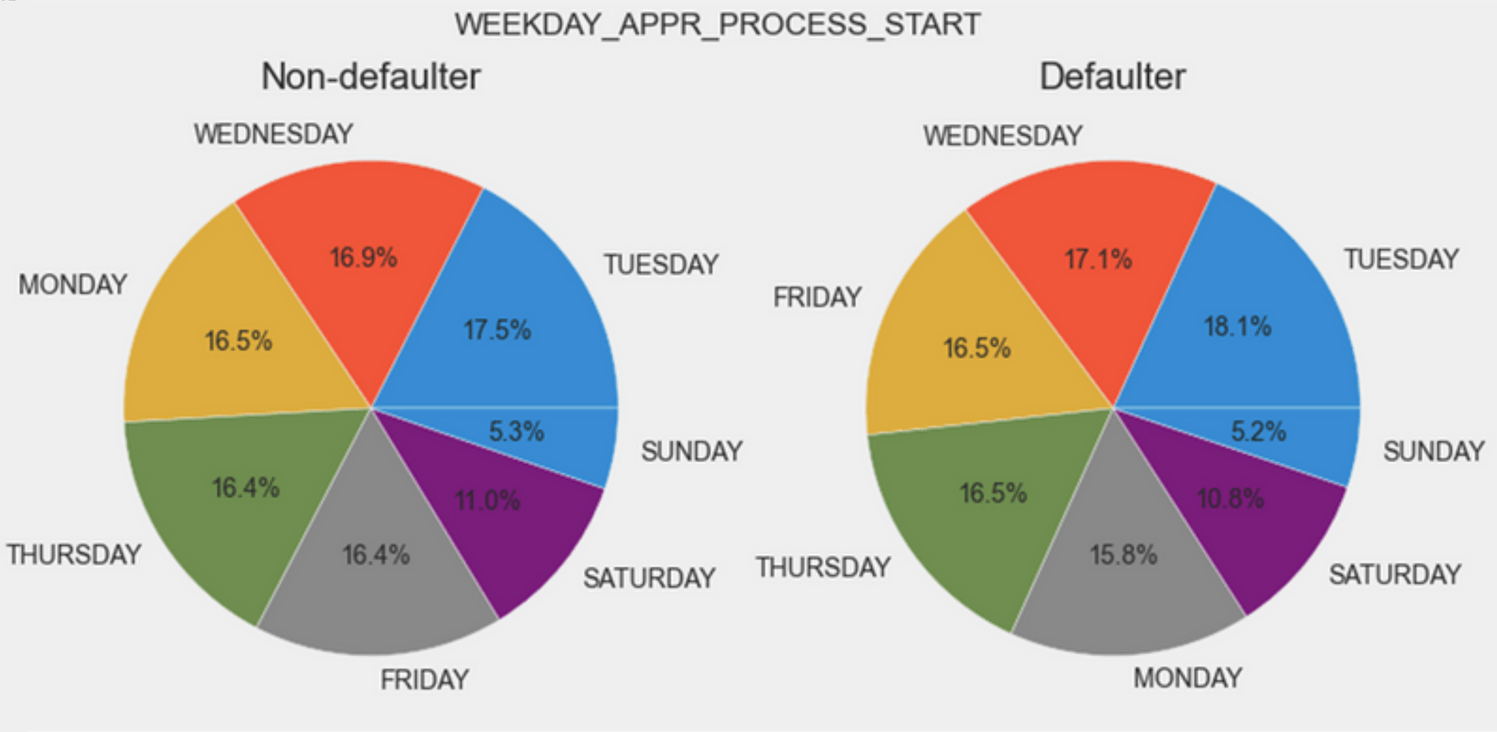
FEATURES SELECTION & ENGINEERING

Feature Engineering

In Application_Train:
Drop unnecessary features



Example: OBS_60_CNT_SOCIAL_CIRCLE and OBS_30_CNT_SOCIAL_CIRCLE are identity columns



WEEKDAY_APPR_PROCESS_START have the similar percentage per day between Non-Defaulter and defaulter

Feature Engineering

In other Data Table:

Merge all the features from every table into application_train table

- numerical features (count, max, min, mean, median, std)

- category features (nunique, count flag, last)

```
agg_POS_df = POS_df.groupby('SK_ID_CURR')["MONTHS_BALANCE","CNT_INSTALMENT", "CNT_INSTALMENT_FUTURE","SK_DPD", "SK_DPD_DEF"].agg( ['min', 'max', 'mean
agg_POS_df
```

	MONTHS_BALANCE				CNT_INSTALMENT				CNT_INSTALMENT_FUTURE				SK_DPD				SK_DPD_DEF			
	min	max	mean	std	min	max	mean	std	min	max	mean	std	min	max	mean	std	min	max	mean	std
SK_ID_CURR																				
100001	-96	-53	-72.555556	20.863312	4.0	4.0	4.000000	0.000000	0.0	4.0	1.444444	1.424001	0	7	0.777778	2.333333	0	7	0.777778	2.333333
100002	-19	-1	-10.000000	5.627314	24.0	24.0	24.000000	0.000000	6.0	24.0	15.000000	5.627314	0	0	0.000000	0.000000	0	0	0.000000	0.000000
100003	-77	-18	-43.785714	24.640162	6.0	12.0	10.107143	2.806597	0.0	12.0	5.785714	3.842811	0	0	0.000000	0.000000	0	0	0.000000	0.000000
100004	-27	-24	-25.500000	1.290994	3.0	4.0	3.750000	0.500000	0.0	4.0	2.250000	1.707825	0	0	0.000000	0.000000	0	0	0.000000	0.000000
100005	-25	-15	-20.000000	3.316625	9.0	12.0	11.700000	0.948683	0.0	12.0	7.200000	3.614784	0	0	0.000000	0.000000	0	0	0.000000	0.000000

Example from POS data

Features Selection

```
df_merged_final = pd.merge(df_01, df_02, how = 'left', on=['SK_ID_CURR'])
df_merged_final = pd.merge(df_merged_final, df_03, how = 'left', on=['SK_ID_CURR'])
df_merged_final = pd.merge(df_merged_final, df_05, how = 'left', on=['SK_ID_CURR'])
df_merged_final = pd.merge(df_merged_final, df_06, how = 'left', on=['SK_ID_CURR'])
df_merged_final = pd.merge(df_merged_final, df_07, how = 'left', on=['SK_ID_CURR'])
df_merged_final
```

- Data merge

```
## check null
def check_missed_values(df):
    df_null_percentage = df.isnull().sum() / df.shape[0] * 100
    df_null_percentage = df_null_percentage.drop(df_null_percentage[df_null_percentage == 0].index).sort_values(ascending= False).reset_i
    df_null_percentage.columns = ["Feature", "Percentage"]
    return df_null_percentage

null_df = check_missed_values(df_merged_final)
null_df

drop_null_col =null_df[null_df["Percentage"] > 70]["Feature"].tolist()
drop_null_col
```

- Null check (Drop if the percentage >70%)

```
numeric_df = df_merged_final._get_numeric_data()
numeric_df.drop([ "SK_ID_CURR", "TARGET"], axis= 1, inplace= True)

count_list =[]
for i in numeric_df.columns:
    fre_count = pd.Series(numeric_df[i].value_counts(normalize= True))
    count_list.append([i, fre_count.index[0], fre_count.values[0] * 100])

frequent_df = pd.DataFrame(count_list)
frequent_df.columns = ["Feature", "Value", "Frequent"]
frequent_df = frequent_df.sort_values(by= "Frequent",ascending= False)
to_drop = frequent_df[frequent_df["Frequent"] > 75]["Feature"].tolist()
to_drop
```

- **Drop all the column with too many zero result (higher than 75%)**

```
upper_tri = corr.where(np.triu(np.ones(corr.shape),k=1).astype(np.bool))
to_drop = [column for column in upper_tri.columns if any(upper_tri[column] > 0.90)]
to_drop
```

- Corralation calculation (drop if too high)

```
df_merged_final["CNT_FAM_MEMBERS"].fillna(df_merged_final["CNT_FAM_MEMBERS"].mode()[0] , inplace = True)
for i in ['BUR_MEAN_ACTIVE_LOANS_PERCENTAGE','BUR_MEAN_CREDIT_TYPE_COUNT','BUR_MEAN_PAST_LOAN_COUNT','BUR_MEAN_AMT_CREDIT_MAX_OVERDUE','B
      df_merged_final[i].fillna(value=df_merged_final[i].median(),inplace=True)

df_merged_final.fillna(0, inplace= True)
```

- Outliner hadle - Fill NaN (avoid when calculating Q1 get NaN)

```
def lower_bound(name_column):
    IQR = df_merged_final[name_column].quantile(0.75) - df_merged_final[name_column].quantile(0.25)
    lower_bridge= df_merged_final[name_column].quantile(0.25)-(IQR * 1.5)
    return lower_bridge

def upper_bound(name_column):
    IQR = df_merged_final[name_column].quantile(0.75) - df_merged_final[name_column].quantile(0.25)
    upper_bridge= df_merged_final[name_column].quantile(0.75) + (IQR * 1.5)
    return upper_bridge

def replace_max(name_column):
    q3 = upper_bound(name_column)
    df_merged_final.loc[df_merged_final[name_column] >= q3, name_column] = q3

def replace_min(name_column):
    q1 = lower_bound(name_column)
    df_merged_final.loc[df_merged_final[name_column] <= q1, name_column] = q1
```

- **Outliner handle - Calculating Q1, Q3**

```
for i in ['AMT_ANNUITY','BUR_MEAN_DEBT_CREDIT_RATIO','BUR_MEAN_ACTIVE_LOANS_PERCENTAGE',
          'POS_BAL_MONTHS_BALANCE_STD','POS_BAL_CNT_INSTALMENT_MIN','POS_BAL_CNT_INSTALMENT_MAX',
          'POS_BAL_CNT_INSTALMENT_MEAN','POS_BAL_CNT_INSTALMENT_STD','INS_BAL_ON_TIME_SUM_FIRST',
          'INS_BAL_INSTALMENT_SUM_FIRST','INS_BAL_TIME_SPAN_SUM','INS_BAL_PAYMENT_RATIO_',
          'PREV_INTEREST_RATE_MAX','PREV_INTEREST_RATE_MIN','PREV_INTEREST_RATE_MEAN',
          'PREV_INTEREST_MAX','PREV_INTEREST_MIN','PREV_INTEREST_MEAN',
          'PREV_CNT_PAYMENT_SUM','PREV_AMT_APPLICATION_MAX','PREV_AMT_APPLICATION_MIN',
          'PREV_AMT_APPLICATION_MEAN','PREV_AMT_ANNUITY_MAX','PREV_AMT_ANNUITY_MIN',
          'PREV_AMT_ANNUITY_SUM','PREV_AMT_ANNUITY_MEAN','REGION_POPULATION_RELATIVE',
          "AMT_CREDIT",'AMT_INCOME_TOTAL']:
    replace_max(i)
```

```
for i in ['BUR_MEAN_DAYS_CREDIT','POS_BAL_MONTHS_BALANCE_MAX','POS_BAL_MONTHS_BALANCE_MEAN',
          'INS_BAL_ON_TIME_PERCENT_','INS_BAL_PAYMENT_RATIO_','PREV_INTEREST_MAX',
          'PREV_INTEREST_MIN','PREV_INTEREST_MEAN','PREV_CNT_PAYMENT_SUM']:
    replace_min(i)
```

- Outliner handle - value that > Q1 will be repalced by Q1, value that >Q3 will be replaced by Q3
- Note: All features is handled by hand because there are variables that needed to count

```
from sklearn.feature_selection import SelectKBest, chi2
from sklearn.preprocessing import OneHotEncoder, OrdinalEncoder

ordinal_encoder = OrdinalEncoder
one_hot_encoder = OneHotEncoder

X_ordinal = df_merged_final[['INS_BAL_INS_ON_TIME_GRADE_','INS_BAL_PAYMENT_GRADE_']]
X_one_hot = df_merged_final[[ 'NAME_FAMILY_STATUS', 'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE', 'OCCUPATION_TYPE', ]]
y = df_merged_final["TARGET"]

X_ordinal = OrdinalEncoder().fit_transform(X_ordinal)
X_one_hot = OneHotEncoder().fit_transform(X_one_hot)
```

- Encode categorical features

```
from sklearn.feature_selection import SelectKBest, chi2

X_cat_use = SelectKBest(chi2, k = 10).fit_transform(X_cat, y)
X_cat_use= pd.DataFrame(X_cat_use)
X_cat_use
```

- Select 10 most important categorical features

```
from sklearn.preprocessing import MinMaxScaler

X_num = numeric_df.drop(["SK_ID_CURR", "TARGET"], axis=1 )

scaler = MinMaxScaler()
X_num = scaler.fit_transform(X_num)
X_num = pd.DataFrame(X_num)
X_num
```

- Encode all Numerical Features column with MinMaxScaler

```
from sklearn.feature_selection import f_classif
X_num_use = SelectKBest(f_classif, k = 30).fit_transform(X_num, y)
X_num_use = pd.DataFrame(X_num_use)
X_num_use
```

- Select 30 important numeric features

```
X_train = pd.concat([y,X_num_use, X_cat_use], axis= 1)
X_train
```

- Merge data into Train dataframe

THANK YOU