PROJECT REPORT

Home Credit Default Risk
(FINANCE)

Submitted towards the partial fulfillment of the criteria for award of Post Graduate Program in Analytics and Artificial Intelligence by Imarticus

Submitted By:

YASHWANTH P

Course and Batch: PGAA Jan 2021

IMARTICUS LEARNING

Acknowledgements

We are using this opportunity to express our gratitude to Prof **Arun Kumar S** and everyone who supported us throughout the course of this group project. We are thankful for their aspiring guidance, invaluably constructive criticism and friendly advice during the project work. We are sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project. Further, we were fortunate to have great teachers who readily shared their immense knowledge in data analytics and guided us in a manner that the outcome resulted in enhancing our data skills. We wish to thank, all the faculties, as this project utilized knowledge gained from every course that formed the PGAA program. We certify that the work done by us for conceptualizing and completing this project is original and authentic.

Date: Aug 29, 2020 Yashwanth P

Place: Chennai

Certificate of Completion

I hereby certify that the project titled "**Home Credit default risk (Finance)**" was undertaken and completed under my supervision by Yashwanth P from the batch of PGAA-01 (Jan 2021)

Date: Aug 29, 2021

Place: Chennai

Scope & Objective

- To Predict the capability of each applicant in repaying the loan with the provided data which has the credit history and other details regarding the applicant

- Classification of a defaulter is important as it will help the applicant and the loan lending company in decision making

Business Problem Statement

Given customer details and history of customer's loans by the organization with one who paid the loan and the defaulters, to predict and classify whether each applicant can repay the loan.

Data Sources

1. application_train.csv
    a. This is the main table
    b. Static data for all applications. One row represents one loan in our data sample.

2. bureau.csv
    a. 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).
    b. 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.

3. bureau_balance.csv
    a. Monthly balances of previous credits in Credit Bureau.
    b. This table has one row for each month of history of every previous credit reported to Credit Bureau – i.e the table has (loans in sample of relative previous credits of months where we have some history observable for the previous credits) rows.

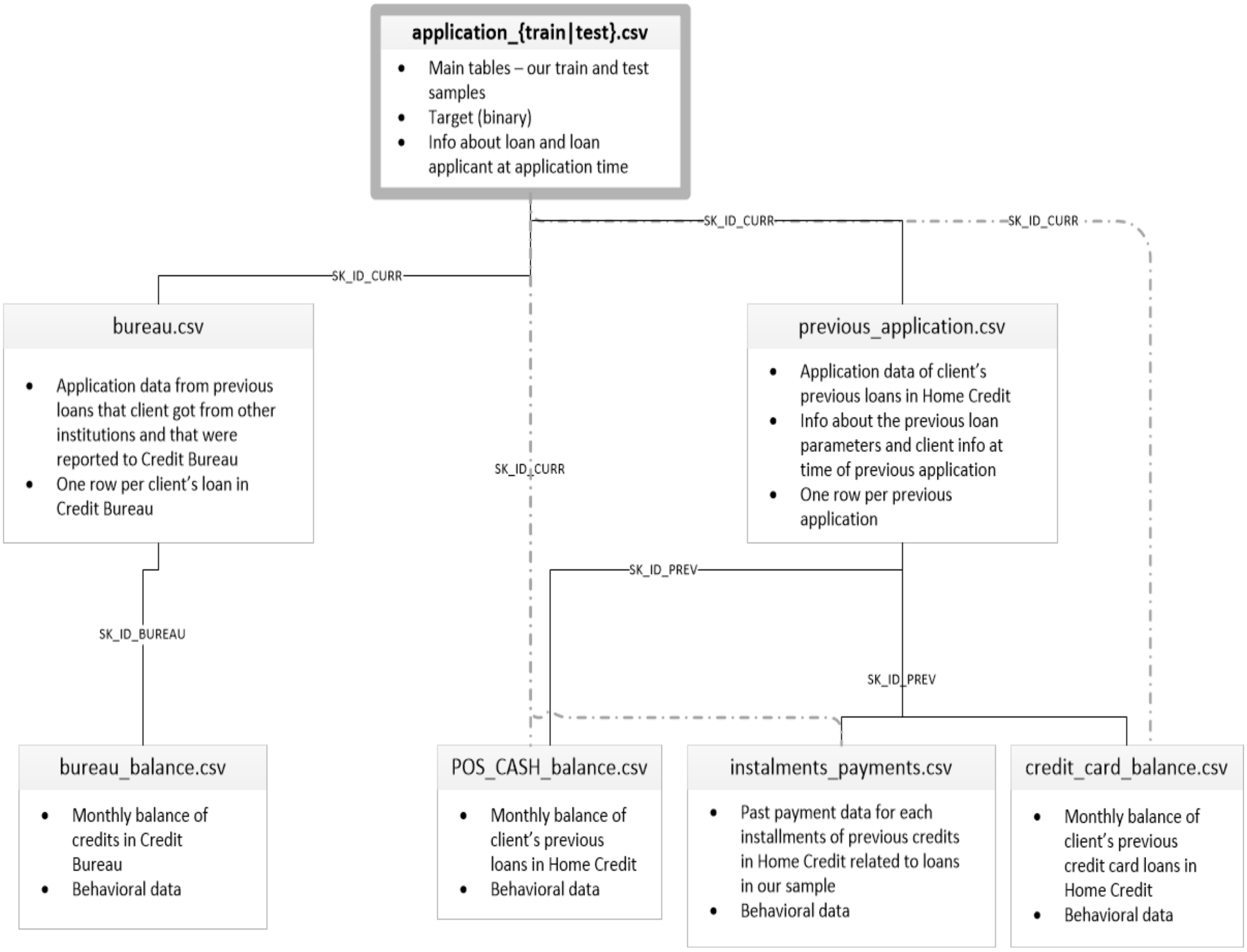
4. POS_CASH_balance.csv
    a. Monthly balance snapshots of previous POS (point of sales) and cash loans that the applicant had with Home Credit.
    b. This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in our sample – i.e. the table has (loans in sample of relative previous credits of months in which we have some history observable for the previous credits) rows.

5. credit_card_balance.csv
    a. Monthly balance snapshots of previous credit cards that the applicant has with Home Credit.
    b. This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in our sample – i.e. the table has (#loans in sample * # of relative previous credit cards * # of months where we have some history observable for the previous credit card) rows.
6. previous_application.csv
    a. All previous applications for Home Credit loans of clients who have loans in our sample.

b. There is one row for each previous application related to loans in our data sample.

7. installments_payments.csv
    a. Repayment history for the previously disbursed credits in Home Credit related to the loans in our sample.
    b. There is a) one row for every payment that was made plus b) one row each for missed payment.
    c. One row is equivalent to one payment of one instalment OR one instalment corresponding to one payment of one previous Home Credit's credit related to loans in our sample.

8. HomeCredit_columns_description.csv
    a. This file contains descriptions for the columns in the various data files.



application_{train|test}.csv
Main tables – our train and test samples
Target (binary)
Info about loan and loan applicant at application time
SK_ID_CURR
SK_ID_CURR
SK_ID_CURR
SK_ID_CURR
bureau.csv
Application data from previous loans that client got from other institutions and that were reported to Credit Bureau
One row per client's loan in Credit Bureau
previous_application.csv
Application data of client's previous loans in Home Credit
Info about the previous loan parameters and client info at time of previous application
One row per previous application
SK_ID_BUREAU
SK_ID_PREV
SK_ID_PREV
bureau_balance.csv
Monthly balance of credits in Credit Bureau
Behavioral data
POS_CASH_balance.csv
Monthly balance of client's previous loans in Home Credit
Behavioral data
instalments_payments.csv
Past payment data for each installments of previous credits in Home Credit related to loans in our sample
Behavioral data
credit_card_balance.csv
Monthly balance of client's previous credit card loans in Home Credit
Behavioral data

Fig: Flow chart of Data used for classification

Dataset Info:

1. application_train.csv contains **307511** rows and **122** columns.
    a. The response variable is “**TARGET**” with ‘**0**’ for non-defaulters and ‘**1**’ for defaulters.
2. Meta data:
    a. bureau.csv contains **1716428** rows and **17** columns
    b. bureau_balance.csv contains **27299925** rows and **3** columns
    c. POS_CASH_balance.csv contains **10001358** rows and **8** columns
    d. credit_card_balance.csv contains **3840312** rows and **23** columns
    e. previous_application.csv contains **1670214** rows and **37** columns
    f. installments_payments.csv contains **13605401** rows and **8** columns

PRIMARY ANALYSIS: (for application_train.csv)

1. Visualization of Target class balance (dependent variable to be predicted):

Non-defaulters: 91.93 %
Defaulters: 8.07 %

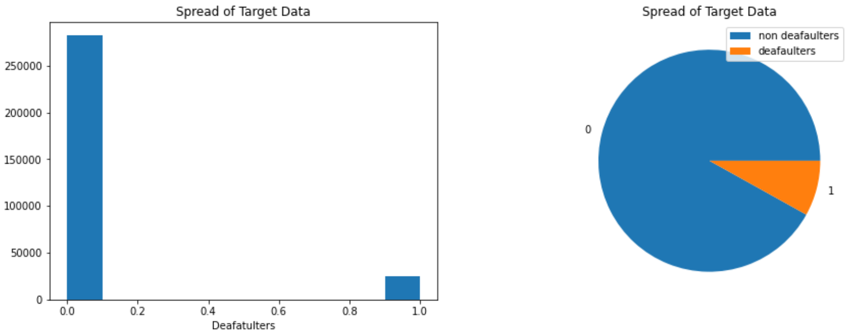


Fig: Target class balance

2. Number of duplicate entries: 0
3. Number of features with high skewness: 87
4. Number of features with missing values: 67
5. Number of features with outliers: 40

MISSING VALUE ANALYSIS:

1. Missing values detection for application_train.csv

Total number of features with missing values are 67. Almost 55% of the features are missing from the available main data. Following is the visualisation with count of missing values in each feature.

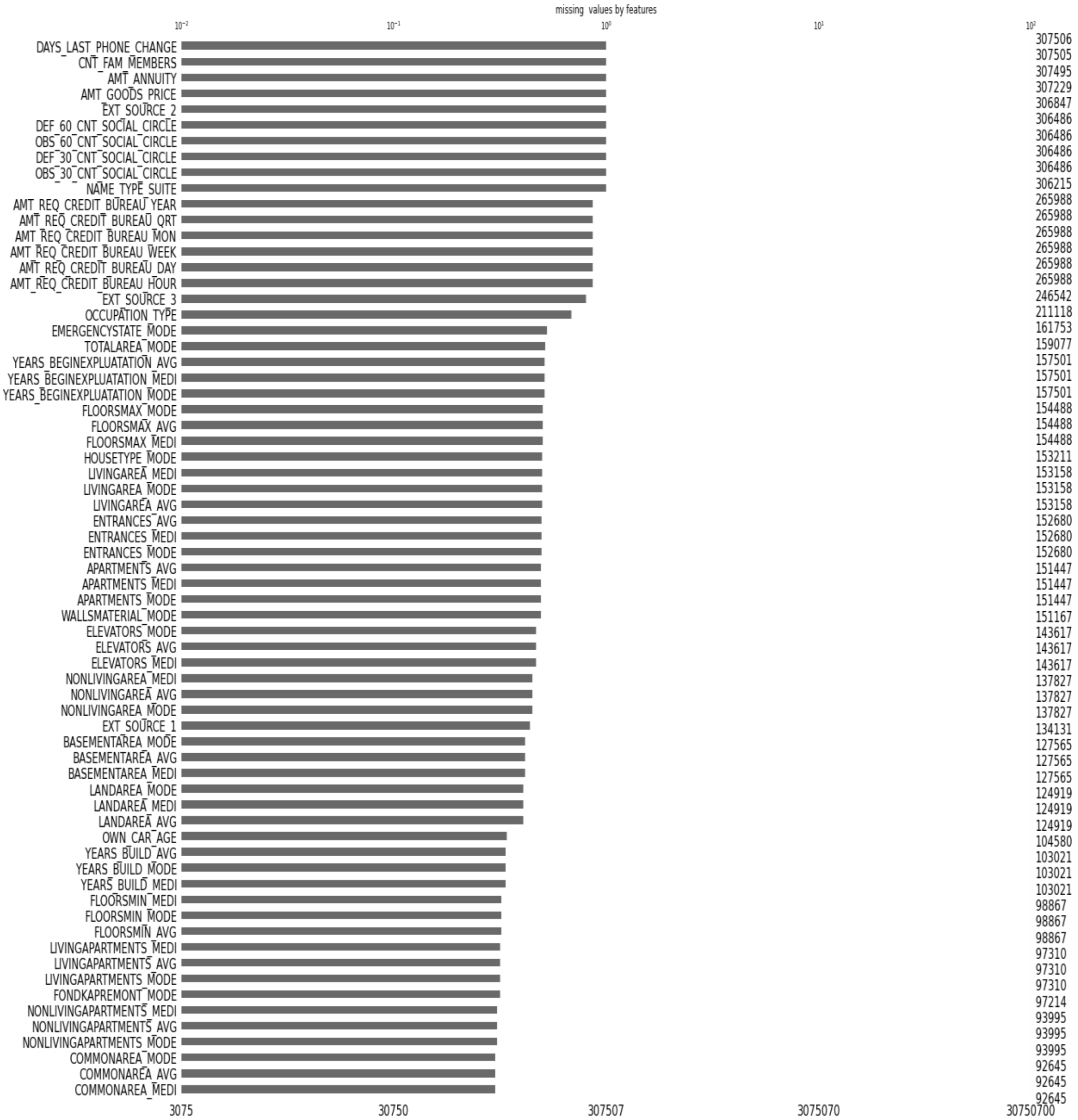


Fig: Features with missing values with count

2. Relationships in missing values:

From the following figure we can able to understand the common missing rows between the features. It is clear that most of the features with missing values are highly correlated with some other feature with missing value.

The horizontal line in the below figure represents the common rows missing between the features and the vertical line represents the quantity of rows, higher the line higher the missing rows.

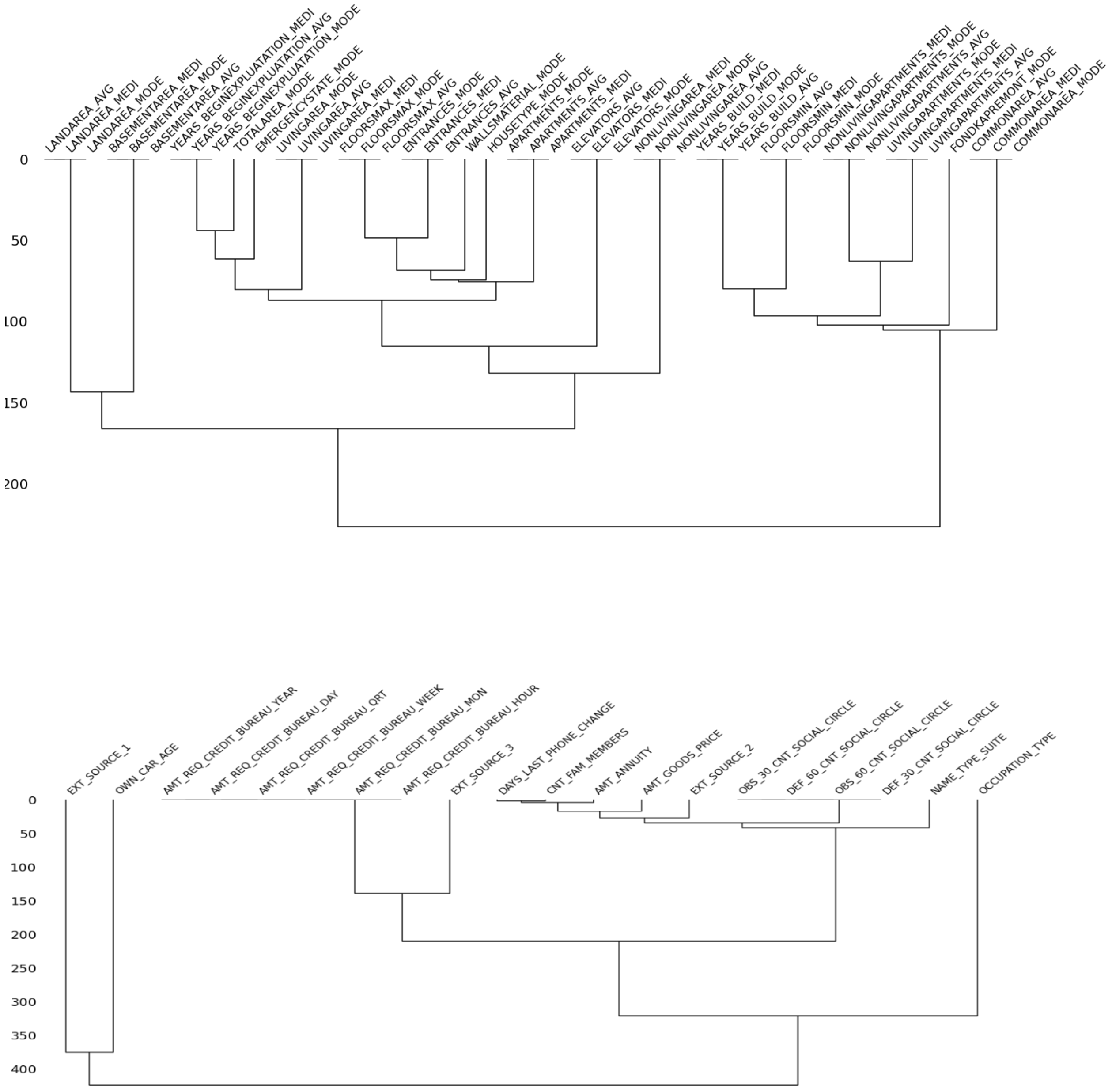
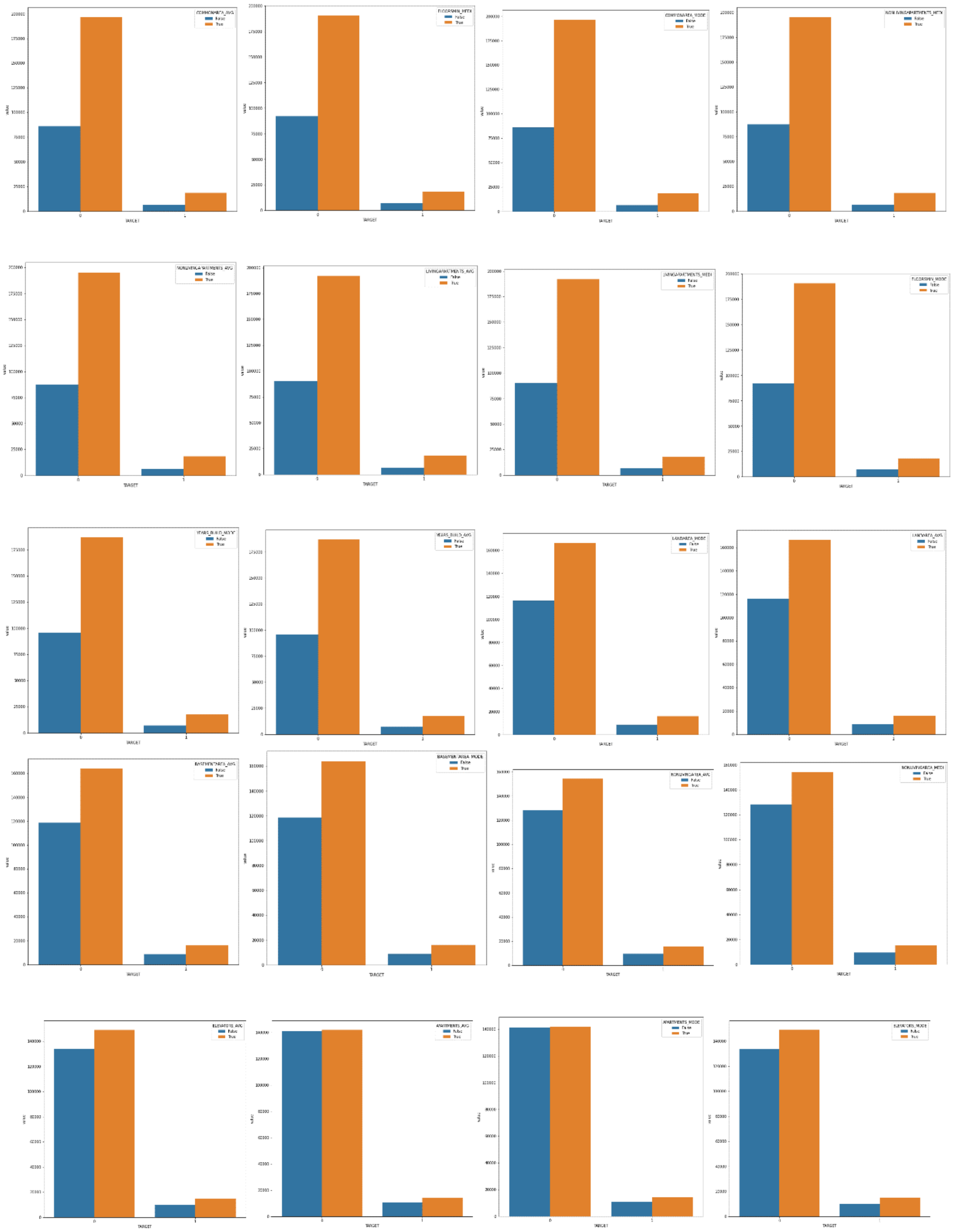


Fig: Missing row relationship between the features.

3. Visualization of missing values proportion with respect to Target variable

The missing values are in equal proportion to either class so there is no relation between the missing values and class, this can be understood by the following graphs,



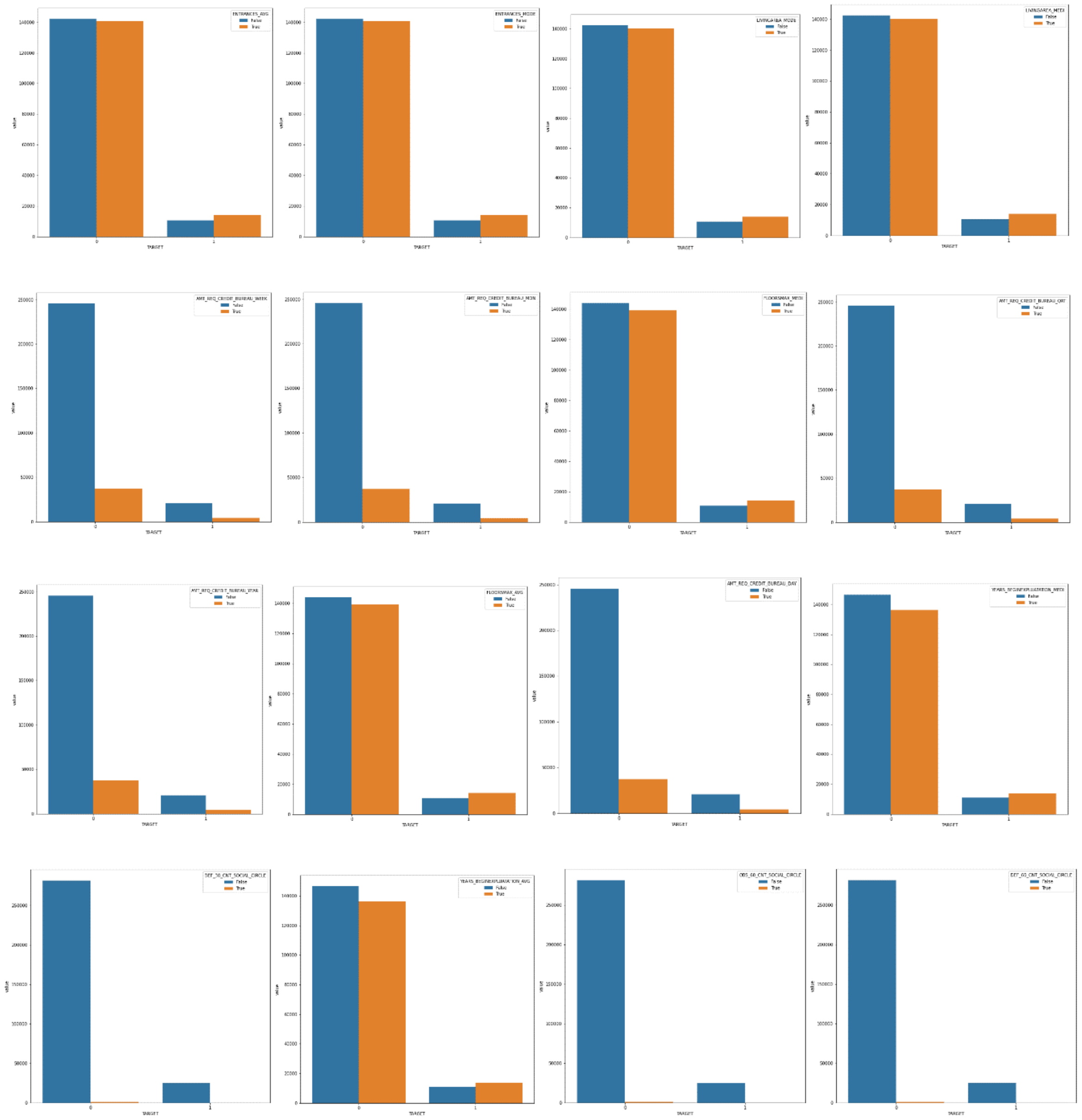


Fig: Missing values proportion with respect to Target variable

Interpretations:

1. The features with missing value greater than 50% can be removed since the imputation may lead to bias or reduced explained variance

2. The features which are missing not at random which has a pattern and high correlation with few other features:

   - AMT_REQ_CREDIT...
   - .... SOCIAL_CIRCLE

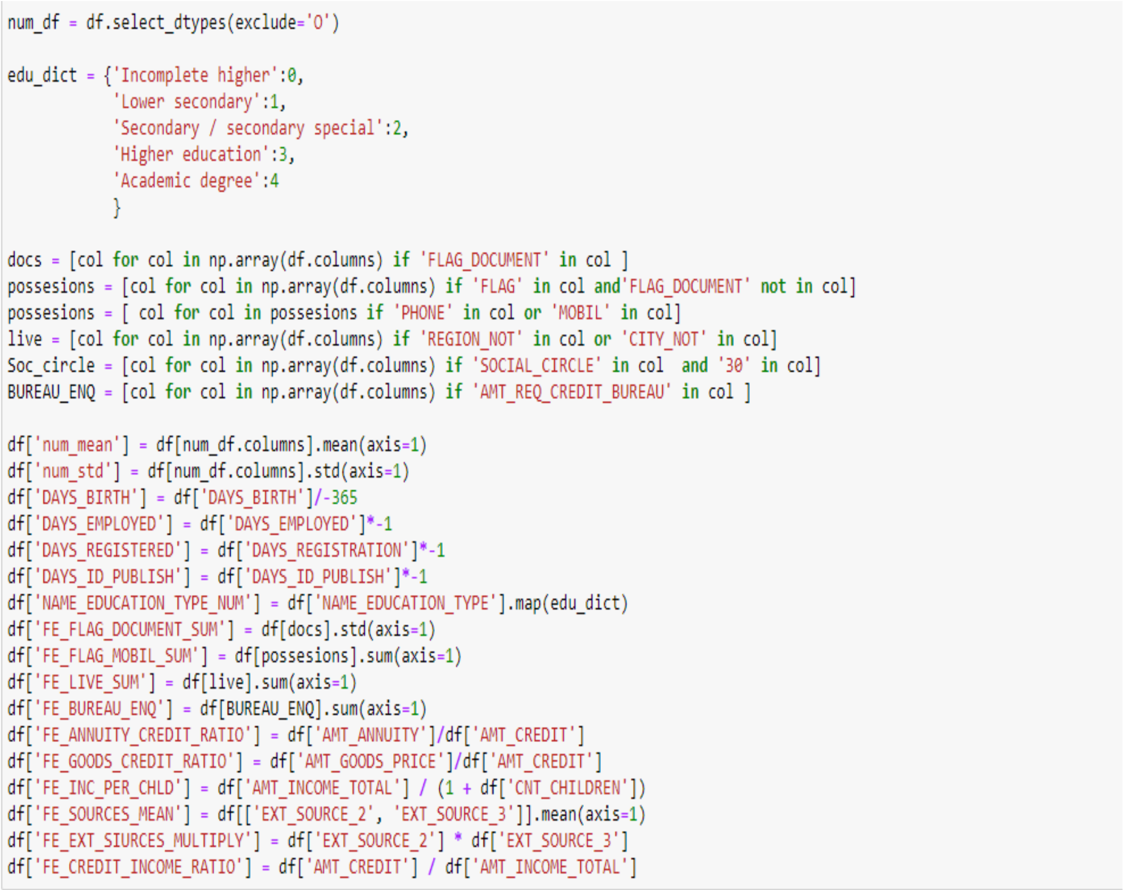
3. Proportion of missing values among the both classes are balanced

4. Values for age of car is assigned to -1 when one has no car.

Missing value treatment:

- Total number of dropped columns which have missing value greater than 50% = 40
- Total number of columns with missing value after dropping = 26
- total number of dropped rows in percentage = 15.980417670864497 %

Feature Engineering

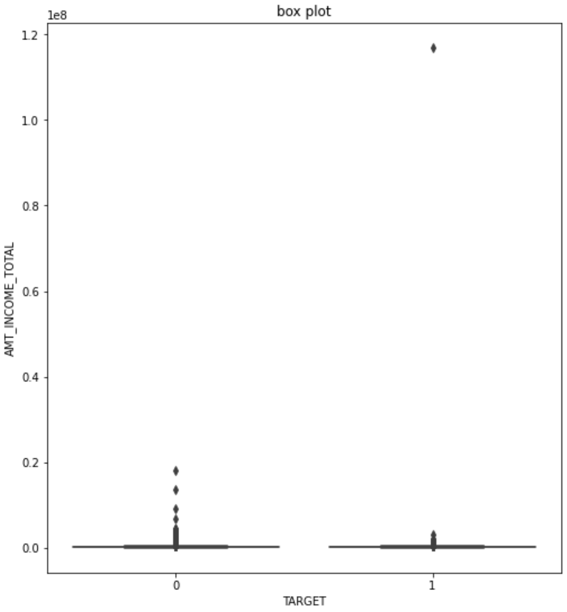


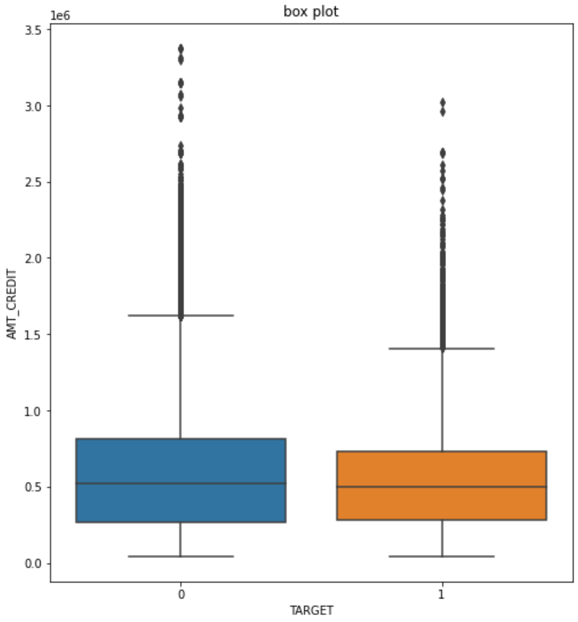
```
num_df = df.select_dtypes(exclude='O')

edu_dict = {'Incomplete higher':0,
            'Lower secondary':1,
            'Secondary / secondary special':2,
            'Higher education':3,
            'Academic degree':4
            }

docs = [col for col in np.array(df.columns) if 'FLAG_DOCUMENT' in col ]
possesions = [col for col in np.array(df.columns) if 'FLAG' in col and'FLAG_DOCUMENT' not in col]
possesions = [ col for col in possesions if 'PHONE' in col or 'MOBIL' in col]
live = [col for col in np.array(df.columns) if 'REGION_NOT' in col or 'CITY_NOT' in col]
Soc_circle = [col for col in np.array(df.columns) if 'SOCIAL_CIRCLE' in col  and '30' in col]
BUREAU_ENQ = [col for col in np.array(df.columns) if 'AMT_REQ_CREDIT_BUREAU' in col ]

df['num_mean'] = df[num_df.columns].mean(axis=1)
df['num_std'] = df[num_df.columns].std(axis=1)
df['DAYS_BIRTH'] = df['DAYS_BIRTH']/-365
df['DAYS_EMPLOYED'] = df['DAYS_EMPLOYED']*-1
df['DAYS_REGISTERED'] = df['DAYS_REGISTRATION']*-1
df['DAYS_ID_PUBLISH'] = df['DAYS_ID_PUBLISH']*-1
df['NAME_EDUCATION_TYPE_NUM'] = df['NAME_EDUCATION_TYPE'].map(edu_dict)
df['FE_FLAG_DOCUMENT_SUM'] = df[docs].std(axis=1)
df['FE_FLAG_MOBIL_SUM'] = df[possesions].sum(axis=1)
df['FE_LIVE_SUM'] = df[live].sum(axis=1)
df['FE_BUREAU_ENQ'] = df[BUREAU_ENQ].sum(axis=1)
df['FE_ANNUITY_CREDIT_RATIO'] = df['AMT_ANNUITY']/df['AMT_CREDIT']
df['FE_GOODS_CREDIT_RATIO'] = df['AMT_GOODS_PRICE']/df['AMT_CREDIT']
df['FE_INC_PER_CHLD'] = df['AMT_INCOME_TOTAL'] / (1 + df['CNT_CHILDREN'])
df['FE_SOURCES_MEAN'] = df[['EXT_SOURCE_2', 'EXT_SOURCE_3']].mean(axis=1)
df['FE_EXT_SIURCES_MULTIPLY'] = df['EXT_SOURCE_2'] * df['EXT_SOURCE_3']
df['FE_CREDIT_INCOME_RATIO'] = df['AMT_CREDIT'] / df['AMT_INCOME_TOTAL']
```

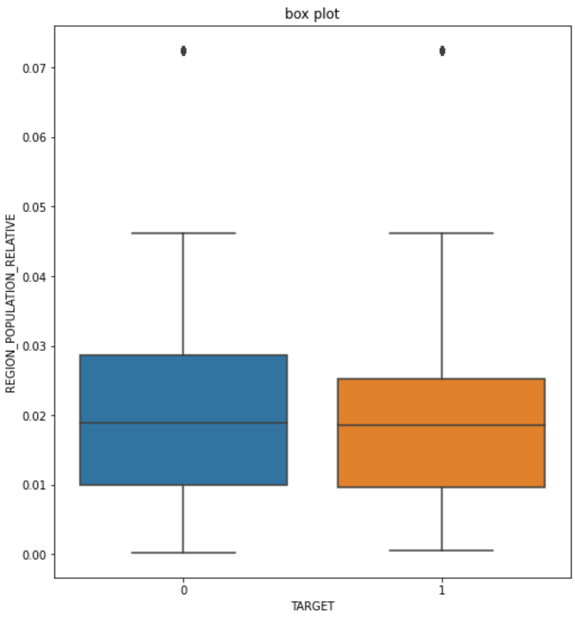
In the application_train.csv, the above feature engineering is done where aggregation, type conversion and mapping values for ordinal categories is done for better interpretability of the data.

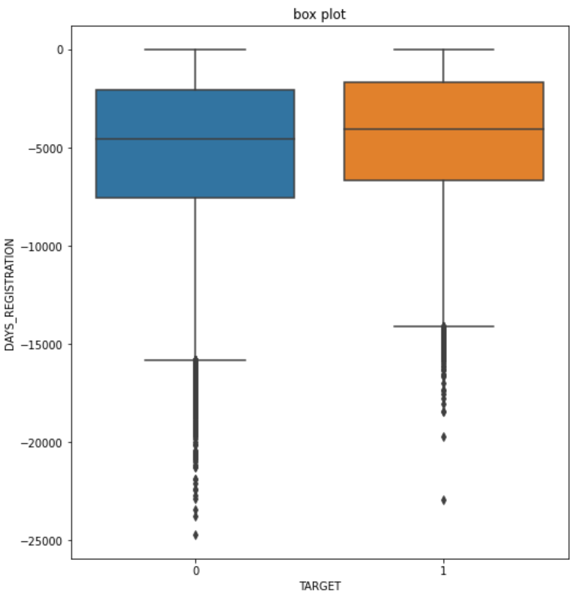
OUTLIER ANALYSIS

The main data has features which have outliers close to 60 and the visualization below will help understanding it better. In the below box plot graph one can understand the outliers in each feature with respect to dependent variable (TARGET).









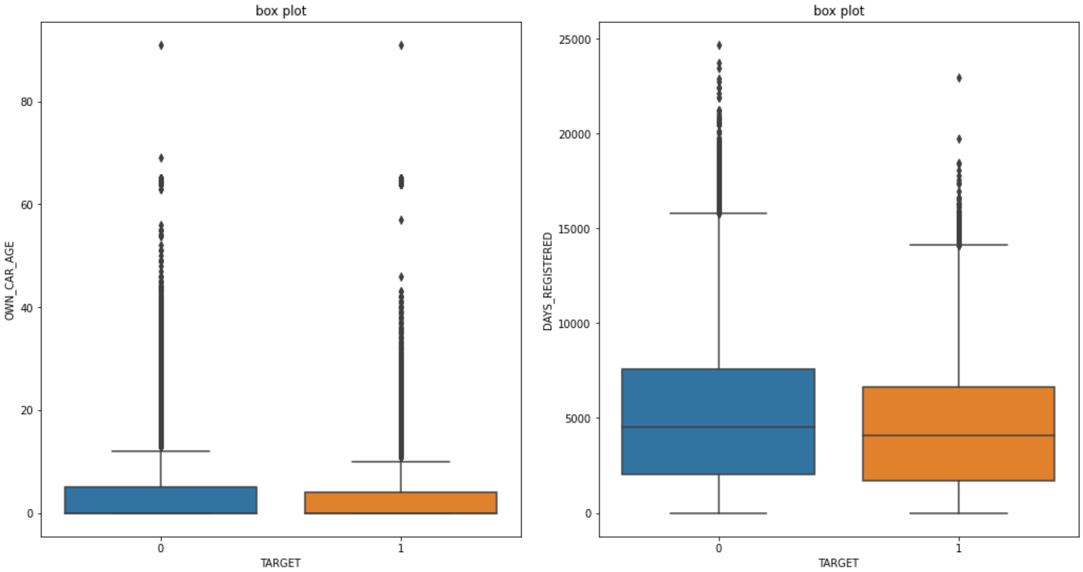


Fig : OUTLIER VISUAL ANALYSIS

OUTLIER TREATMENT:

From the above graphs we can understand that the most outliers are proportional with respect to dependent class and we can drop the records which has large outliers.

- Dropped records in data frame where value of AMT_INCOME_TOTAL is greater than 11000000
- Dropped records in data frame where value of OBS_30_CNT_SOCIAL_CIRCLE is greater than 350
- Dropped records in data frame where value of DEF_30_CNT_SOCIAL_CIRCLE is greater than 30
- Dropped records in data frame where value of OBS_60_CNT_SOCIAL_CIRCLE is greater than 300
- Dropped records in data frame where value of DEF_60_CNT_SOCIAL_CIRCLE is greater than 20
- Dropped records in data frame where value of AMT_REQ_CREDIT_BUREAU_QRT is greater than 200

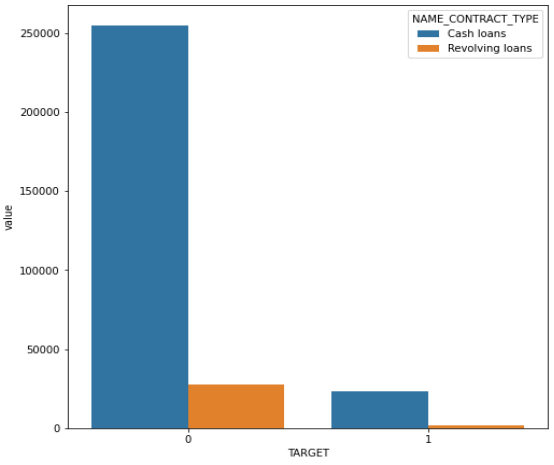
Here OBS_30_CNT_SOCIAL_CIRCLE, DEF_30_CNT_SOCIAL_CIRCLE, OBS_60_CNT_SOCIAL_CIRCLE and DEF_60_CNT_SOCIAL_CIRCLE have a common record as an outlier.

EXPLORATORY DATA ANALYSIS (EDA)

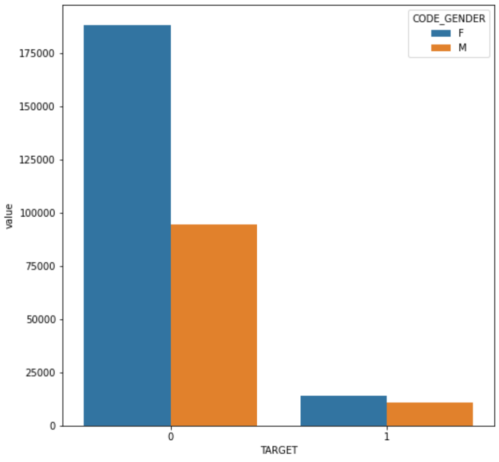
1. Visual EDA for categorical features:

All the features have equal split of the categories with respect to the dependent variable and it can be observed in the below bar graphical representation.

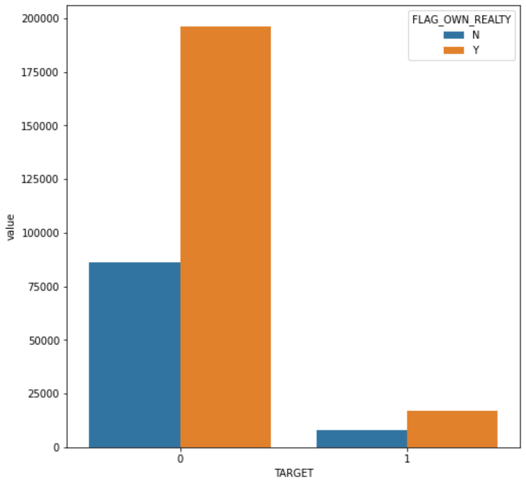
NAME_CONTRACT_TYPE



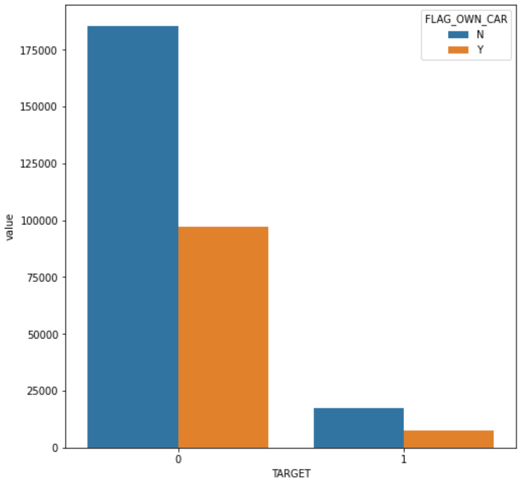
CODE_GENDER



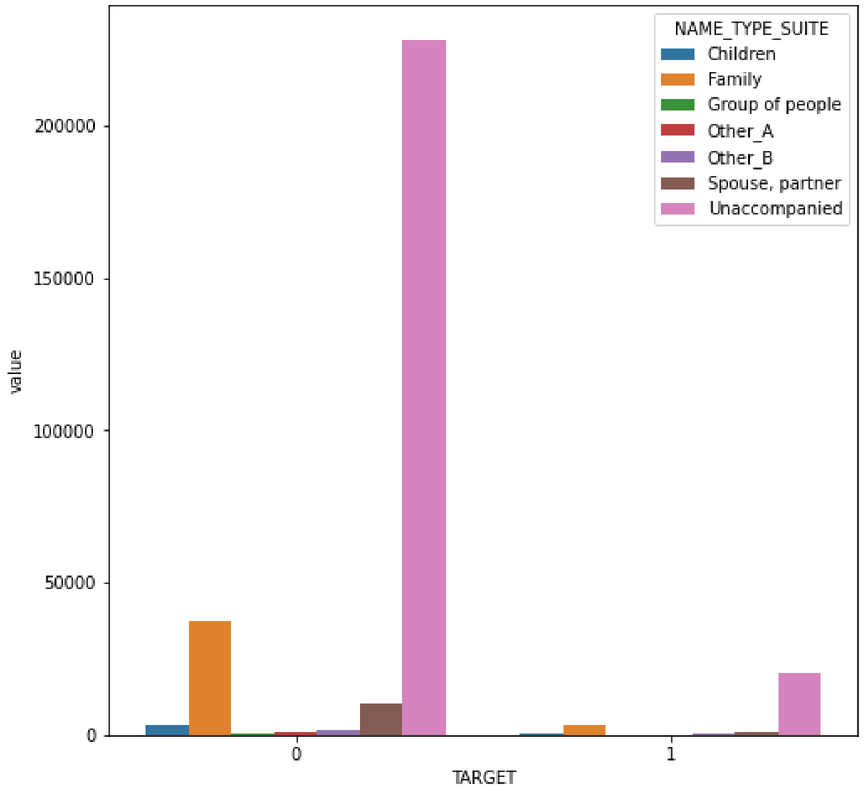
FLAG_OWN_CAR



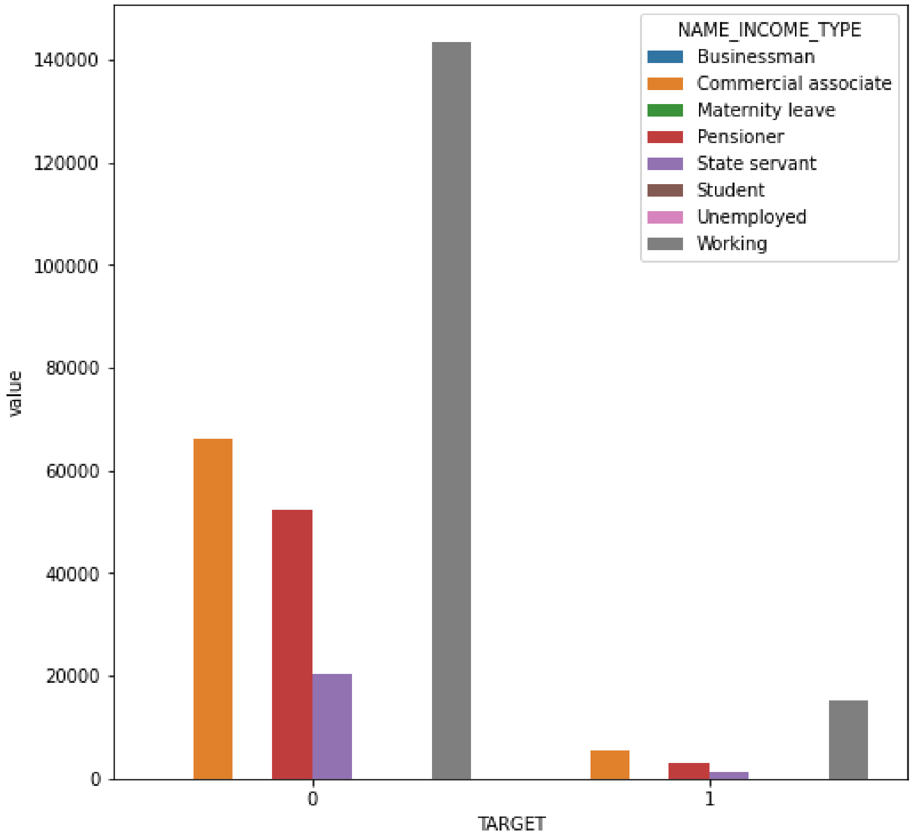
FLAG_OWN_REALTY



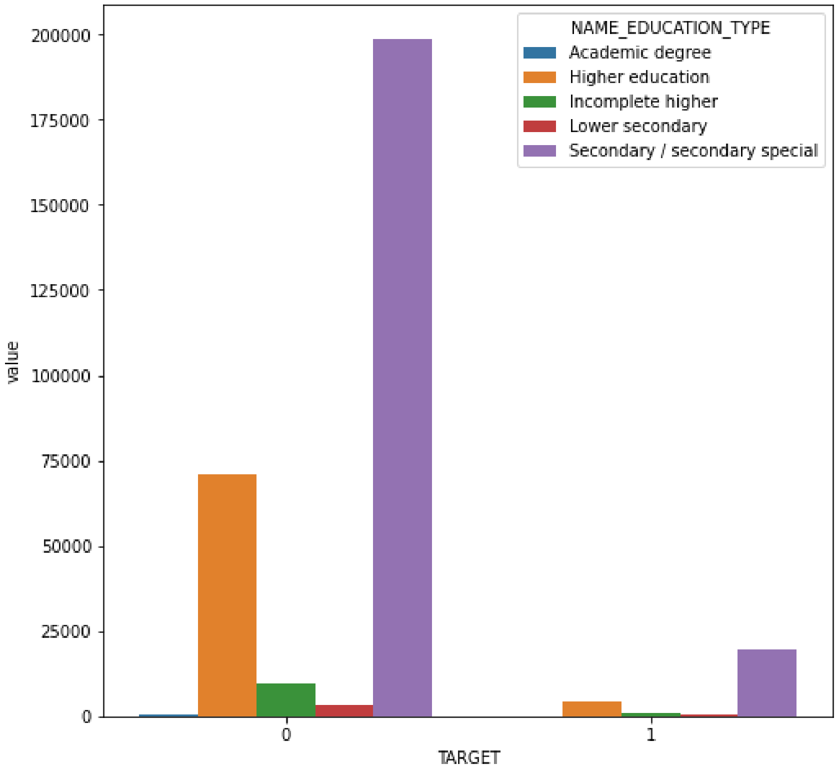
NAME_TYPE_SUITE



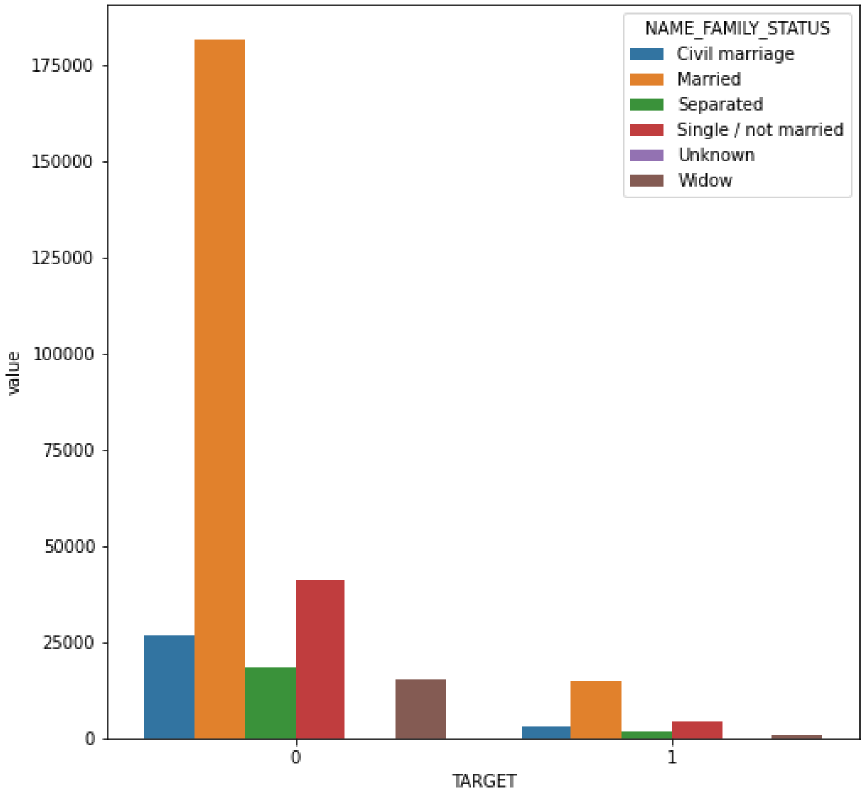
NAME_INCOME_TYPE



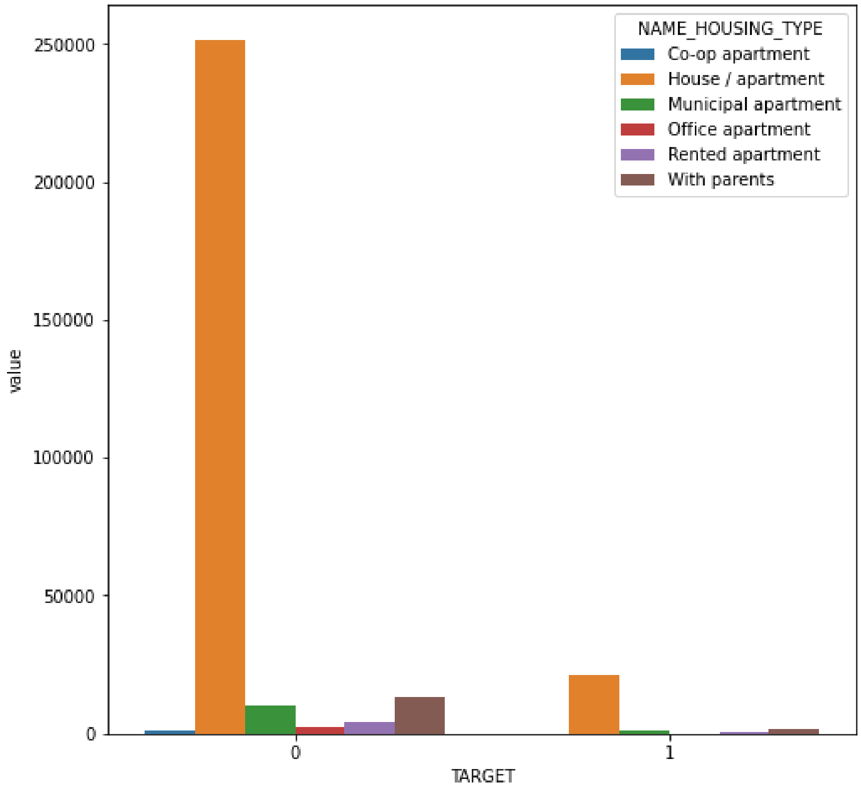
NAME_EDUCATION_TYPE



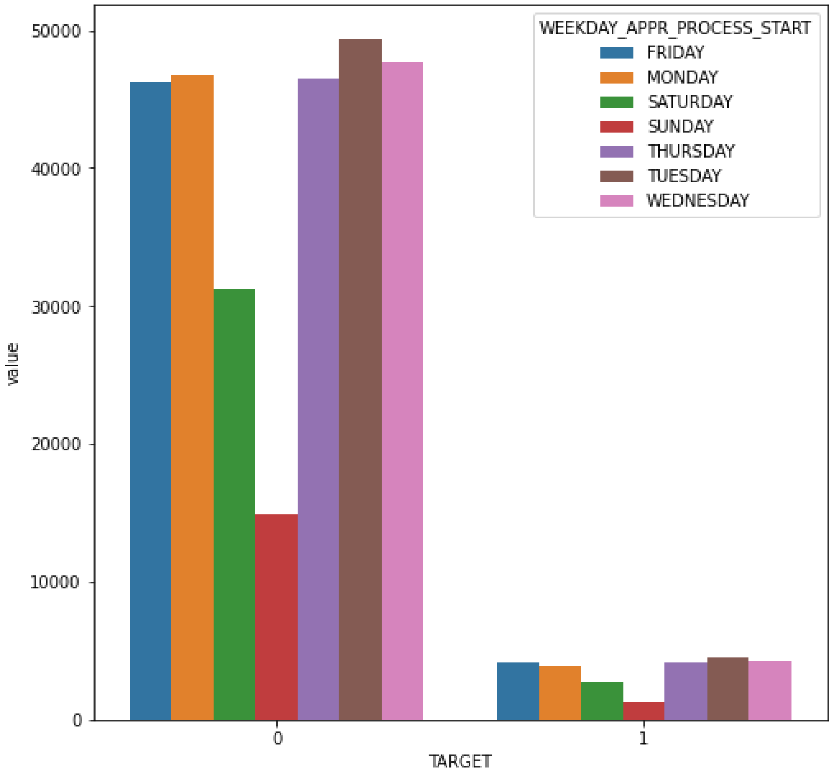
NAME_FAMILY_STATUS



NAME_HOUSING_TYPE



WEEKDAY_APPR_PROCESS_START



EMERGENCYSTATE_MODE

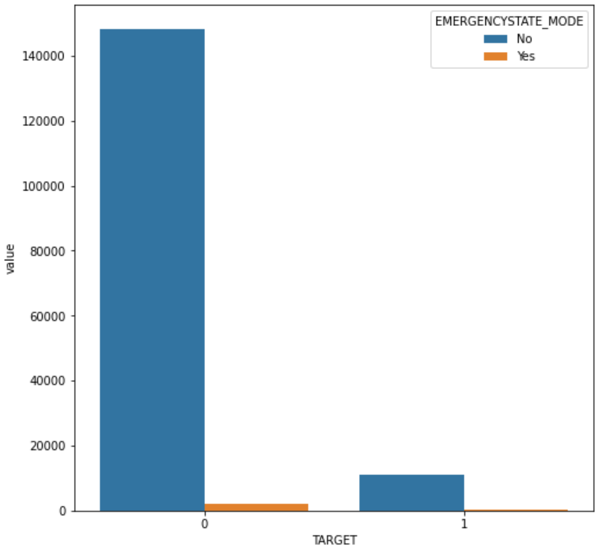
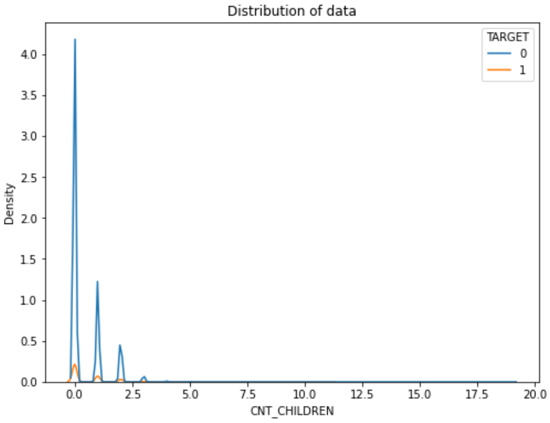
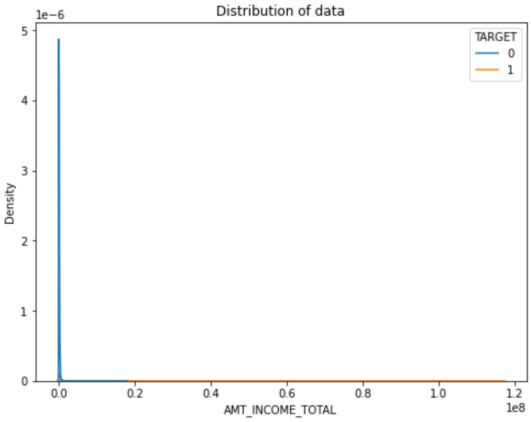


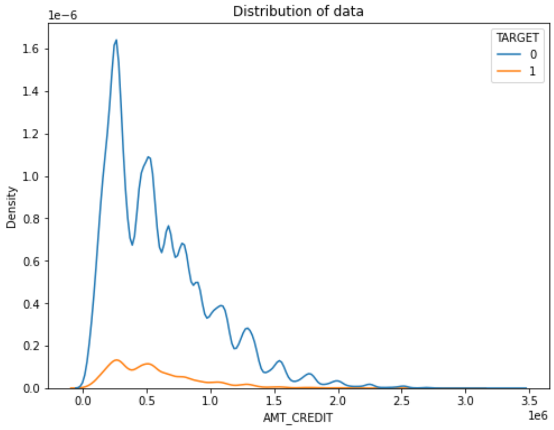
Fig: Data available for each class in categorical Variables

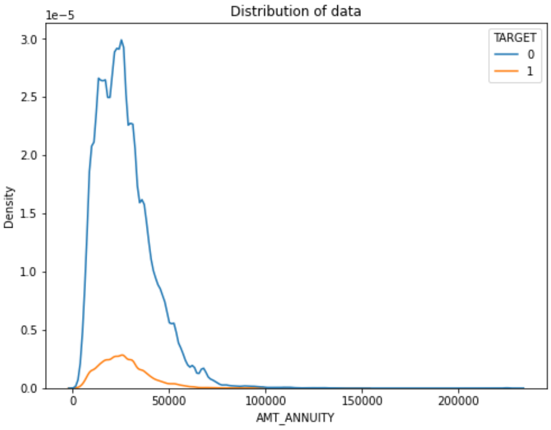
2. Visual EDA for features with numerical values:

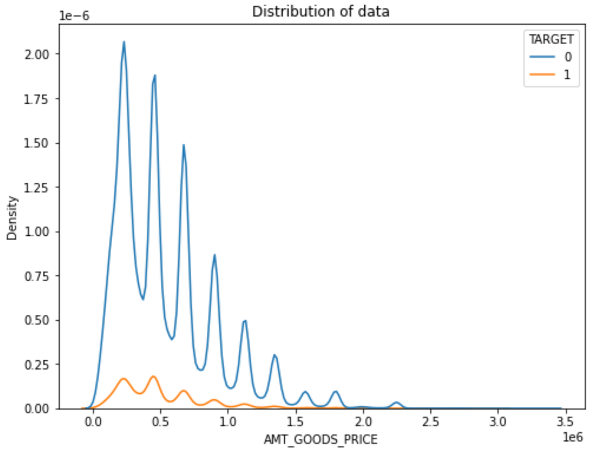
From the below graph it is clear that there is not much difference in the spread of the data for both the classes across all features, only the count of records varies. It is very clear that there is no variance among the spread of data between class 1 and 0 of dependent variable. The visual representation of the spread of the data will help to understand it better.

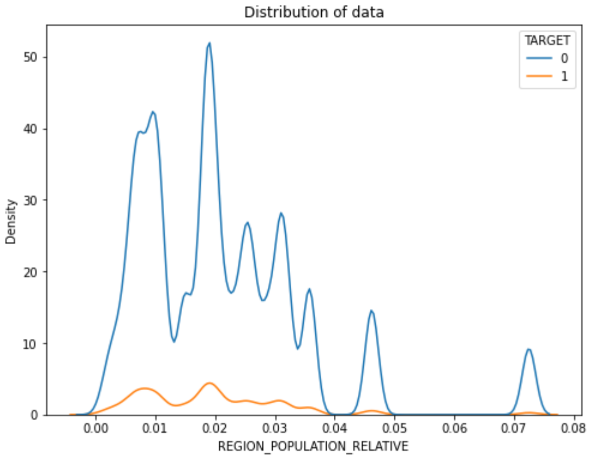


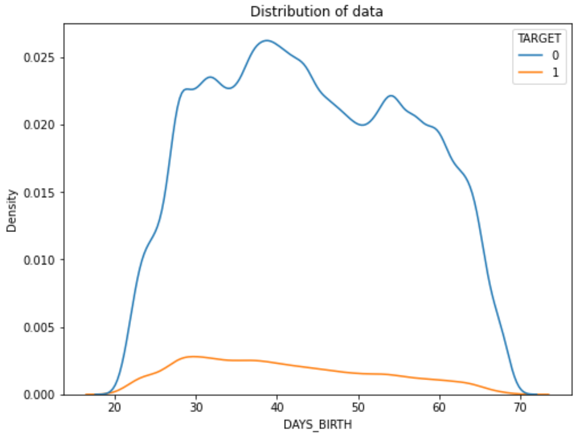


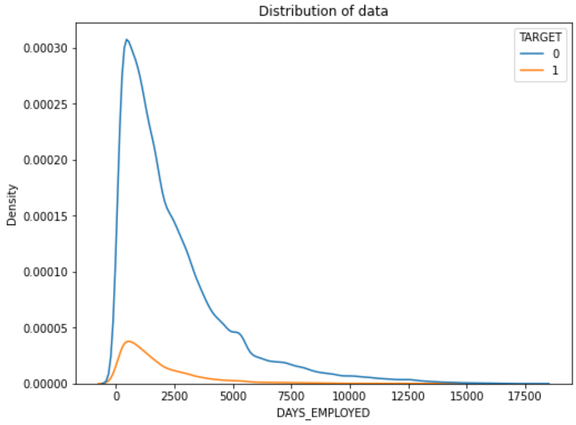

Distribution of data

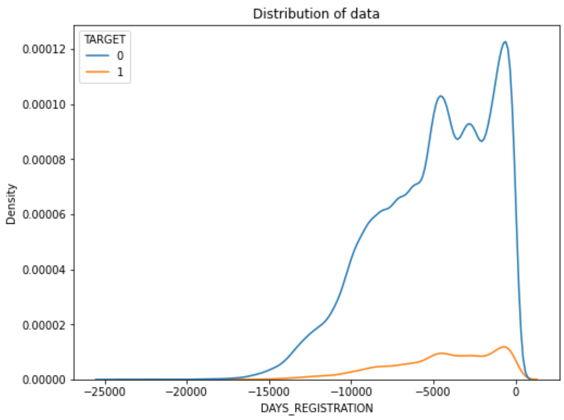

Distribution of data

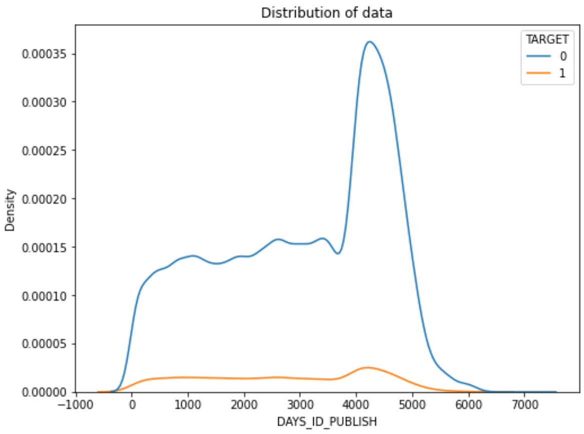

Distribution of data

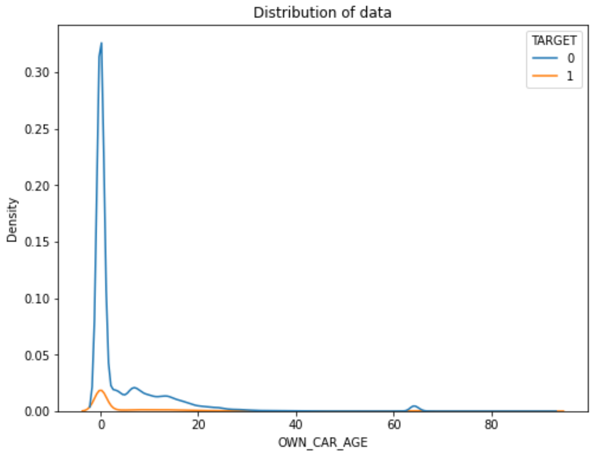

Distribution of data

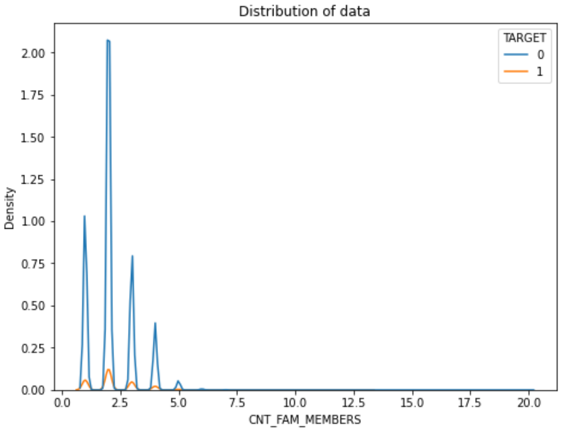

Distribution of data

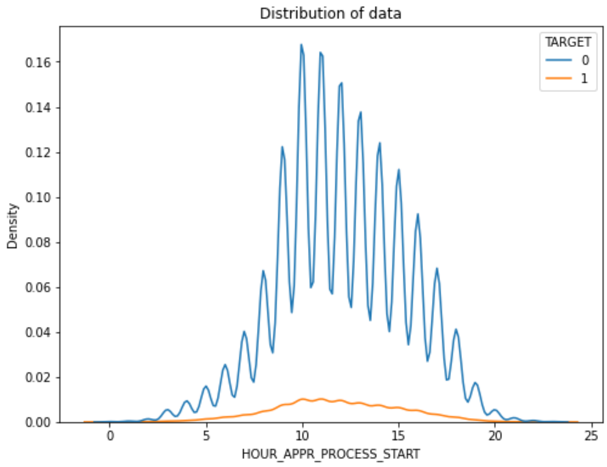

Distribution of data

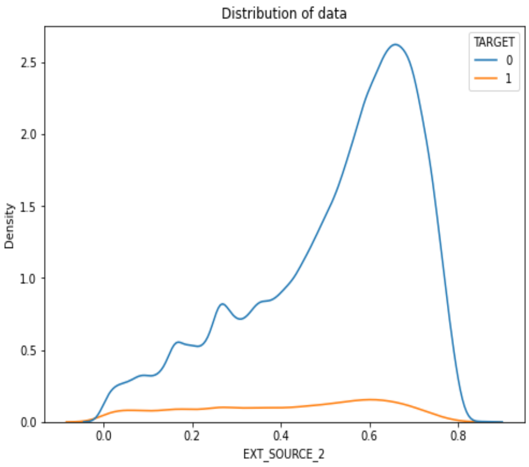

Distribution of data
TARGET
0
1
Density
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
−25000
−20000
−15000
−10000
−5000
0
DAYS_REGISTRATION

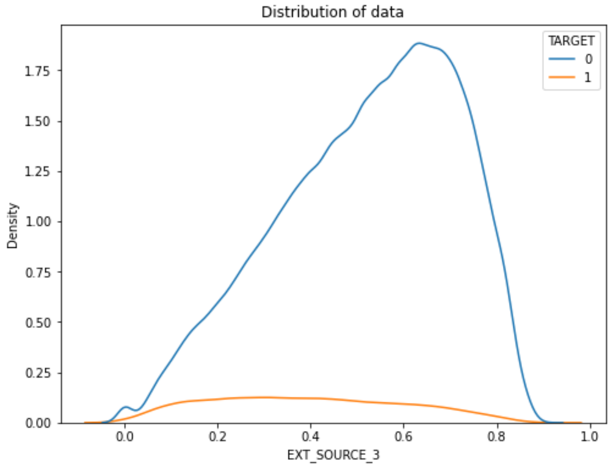

Distribution of data
TARGET
0
1
Density
0.00035
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
−1000
0
1000
2000
3000
4000
5000
6000
7000
DAYS_ID_PUBLISH

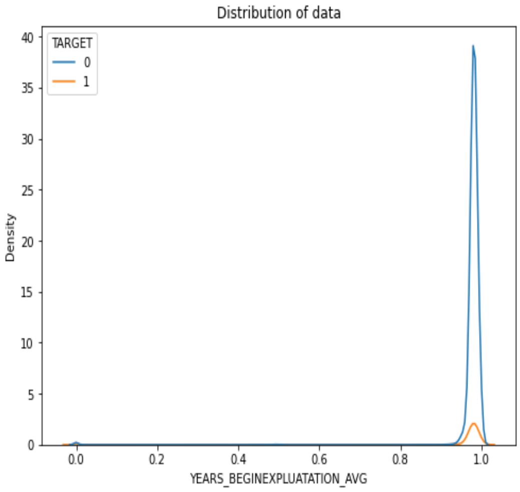

Distribution of data
TARGET
0
1
Density
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
20
40
60
80
OWN_CAR_AGE

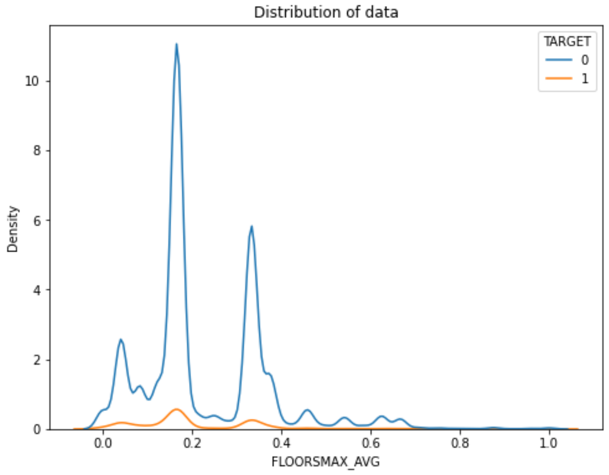

Distribution of data
TARGET
0
1
Density
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
CNT_FAM_MEMBERS

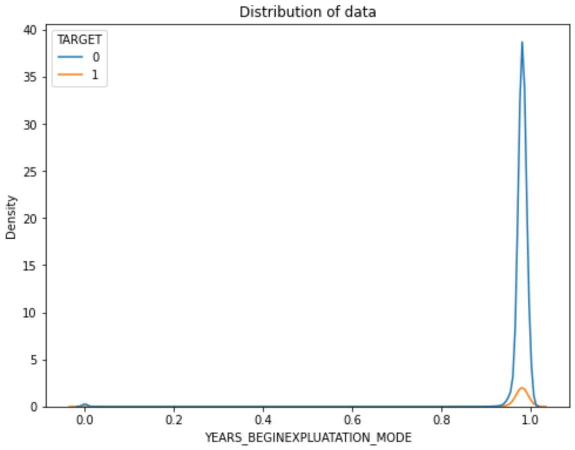

Distribution of data
TARGET
0
1
Density
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25
HOUR_APPR_PROCESS_START

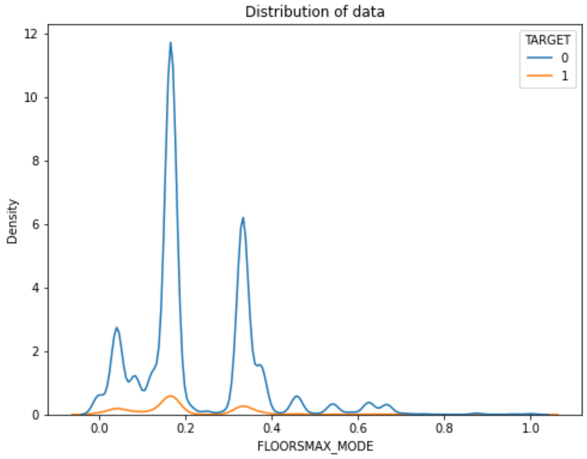

Distribution of data
TARGET
0
1
Density
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
EXT_SOURCE_2

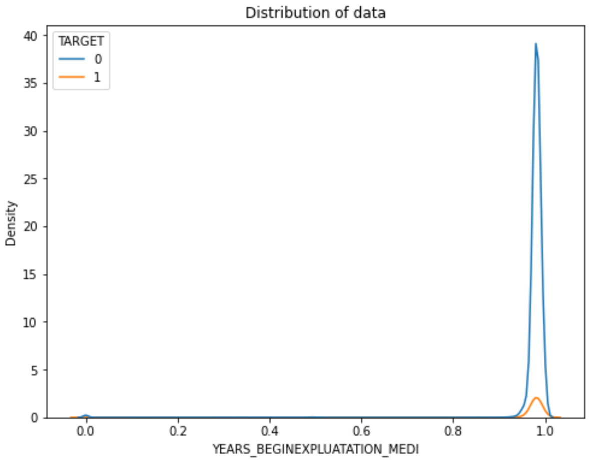

Distribution of data

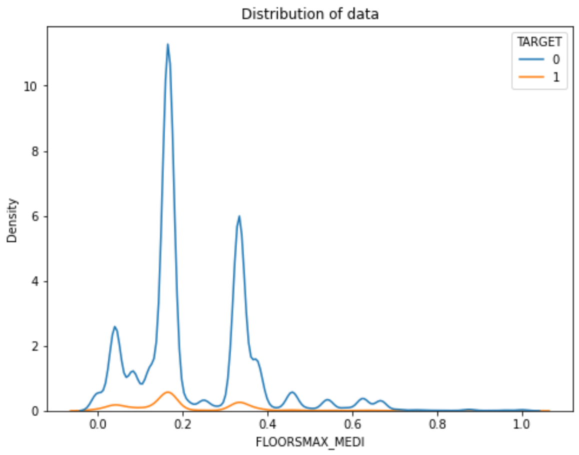

Distribution of data

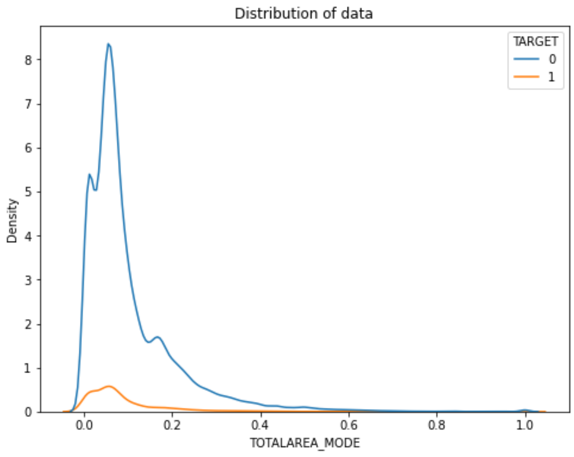

Distribution of data

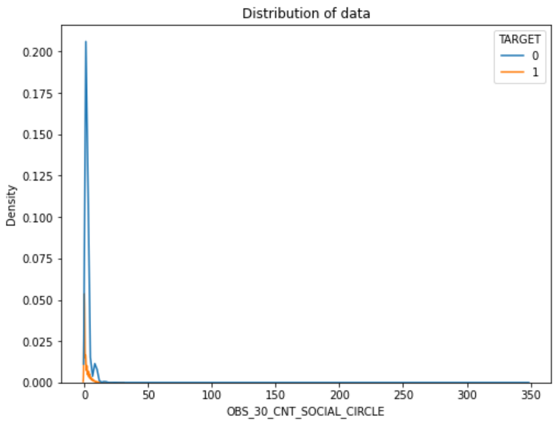

Distribution of data

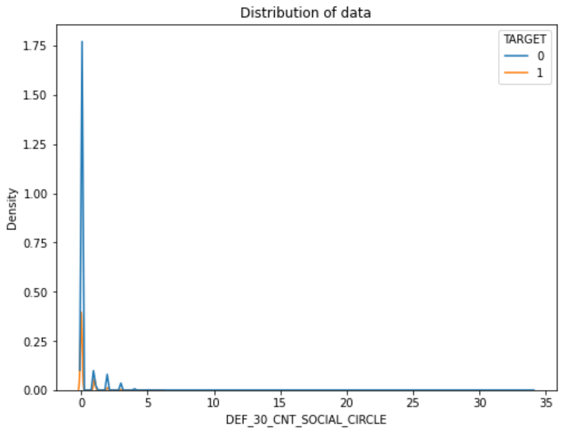

Distribution of data

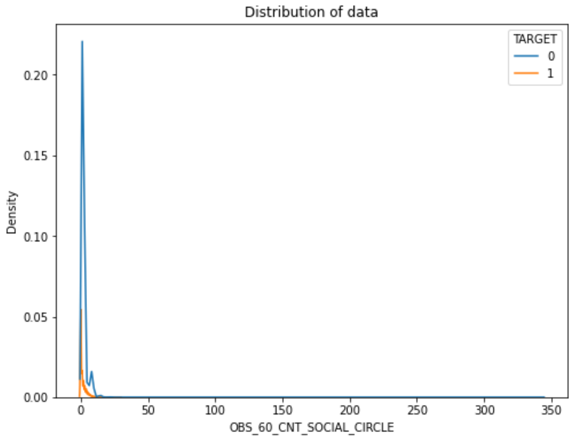

Distribution of data

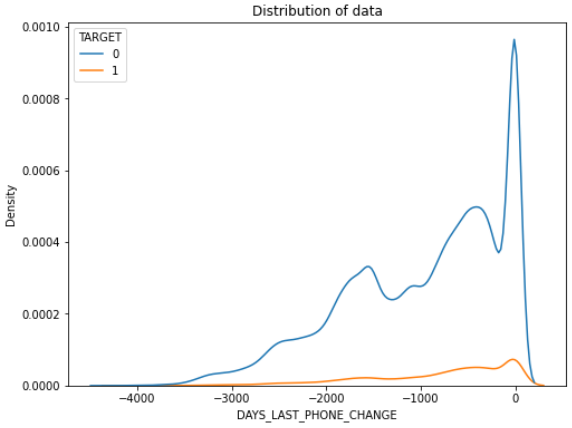

Distribution of data
TARGET
0
1
Density
0
2
4
6
8
10
0.0
0.2
0.4
0.6
0.8
1.0
FLOORSMAX_MEDI

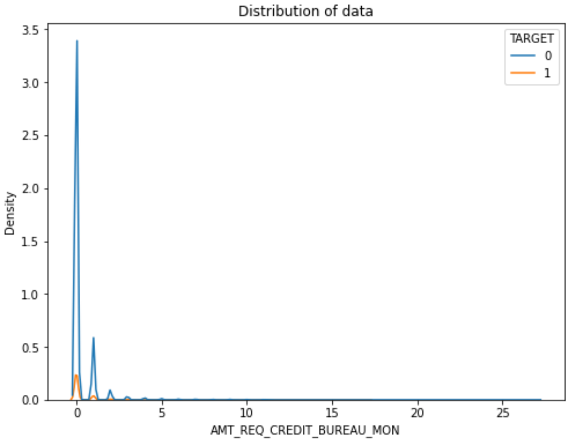

Distribution of data
TARGET
0
1
Density
0
1
2
3
4
5
6
7
8
0.0
0.2
0.4
0.6
0.8
1.0
TOTALAREA_MODE

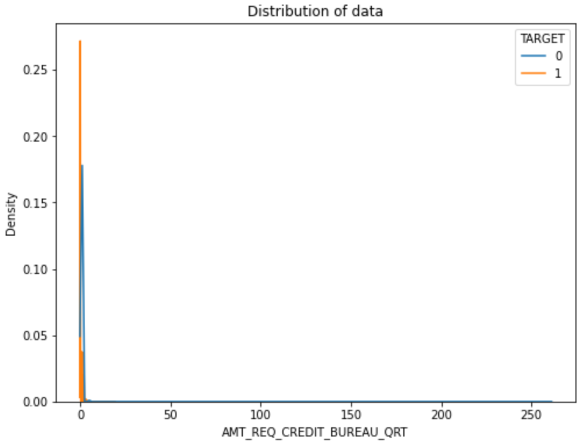

Distribution of data
TARGET
0
1
Density
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200
0
50
100
150
200
250
300
350
OBS_30_CNT_SOCIAL_CIRCLE

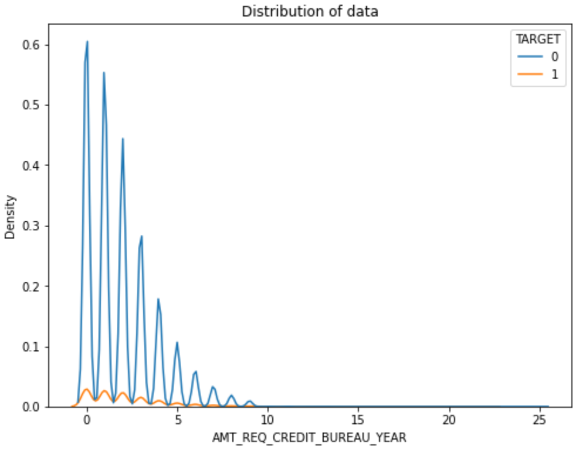

Distribution of data
TARGET
0
1
Density
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
0
5
10
15
20
25
30
35
DEF_30_CNT_SOCIAL_CIRCLE

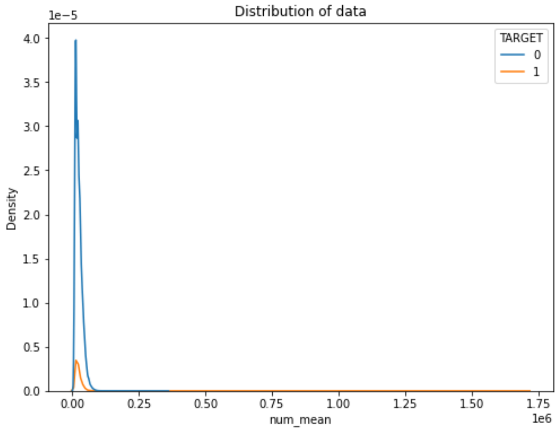

Distribution of data
TARGET
0
1
Density
0.00
0.05
0.10
0.15
0.20
0
50
100
150
200
250
300
350
OBS_60_CNT_SOCIAL_CIRCLE

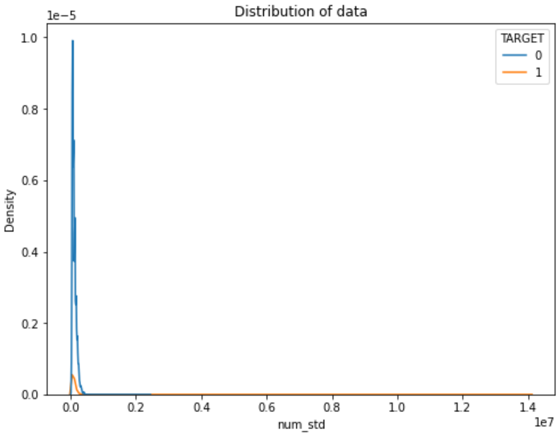

Distribution of data
TARGET
0
1
Density
0.0000
0.0002
0.0004
0.0006
0.0008
0.0010
−4000
−3000
−2000
−1000
0
DAYS_LAST_PHONE_CHANGE

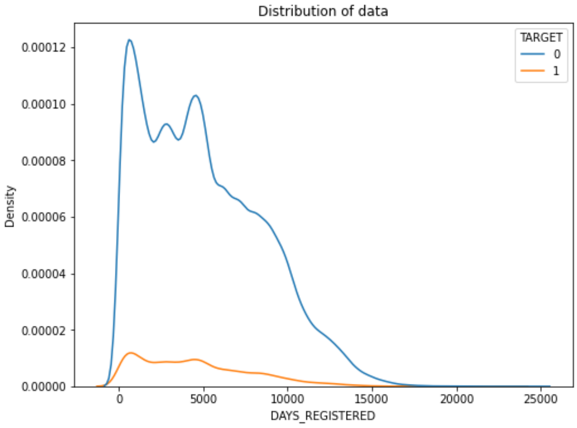

Distribution of data
Density
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
5
10
15
20
25
AMT_REQ_CREDIT_BUREAU_MON
TARGET
0
1


Distribution of data
Density
0.25
0.20
0.15
0.10
0.05
0.00
0
50
100
150
200
250
AMT_REQ_CREDIT_BUREAU_QRT
TARGET
0
1


Distribution of data
Density
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20
25
AMT_REQ_CREDIT_BUREAU_YEAR
TARGET
0
1


Distribution of data
1e−5
Density
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
1e6
num_mean
TARGET
0
1


Distribution of data
1e−5
Density
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1e7
num_std
TARGET
0
1


Distribution of data
Density
0.00012
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
0
5000
10000
15000
20000
25000
DAYS_REGISTERED
TARGET
0
1

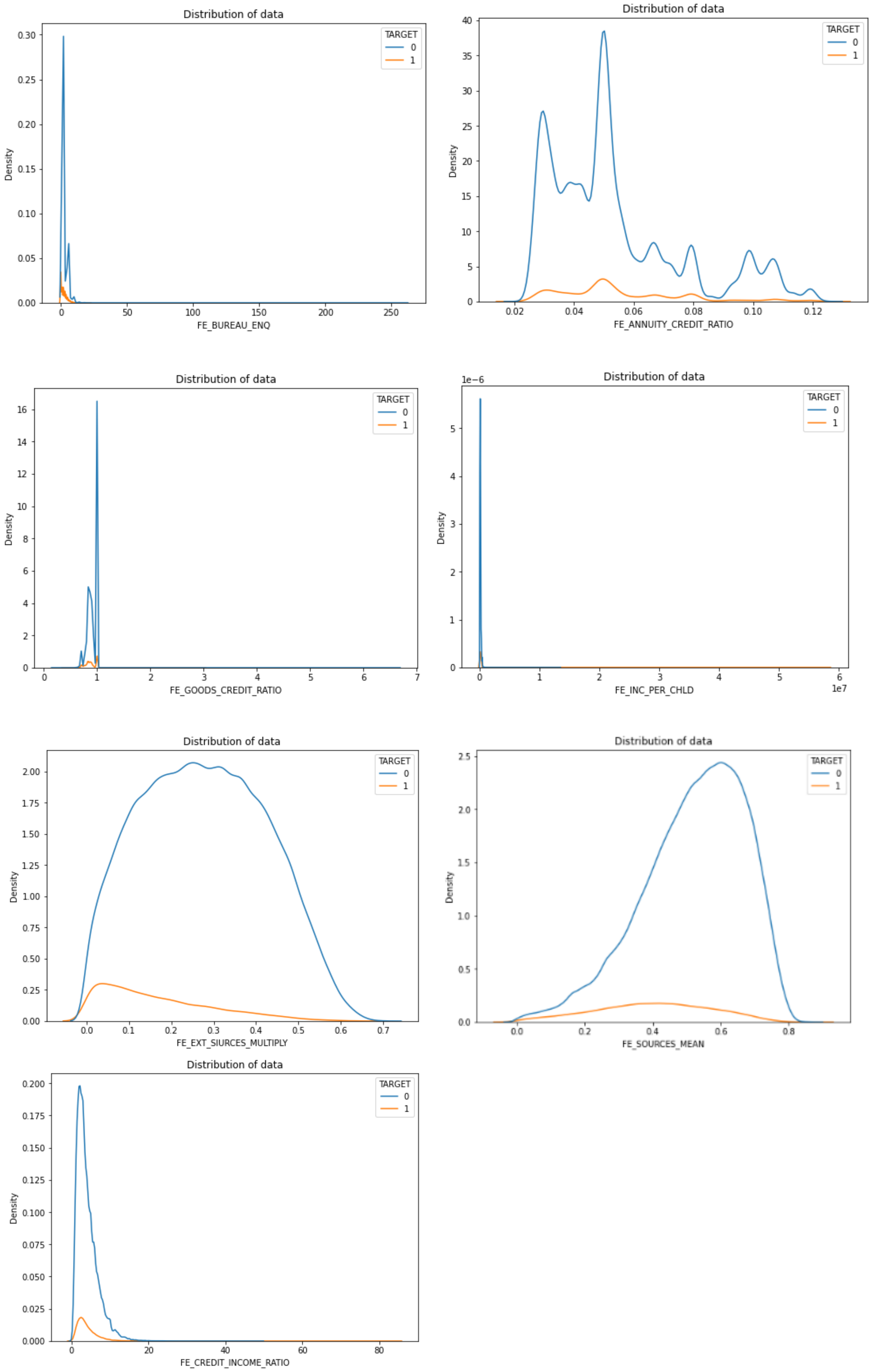


Fig: Distribution of numerical variables respect to target variable

Interpretation from EDA:

1. The spread of the data for class 1 and 0 is very similar for maximum features which makes the classification harder as there is no much discrimination observed to classify the classes

2. Skewness is observer in most of the features which should be corrected for better classification.

Merging the data with meta data

The main data (Application_train.csv) which is cleaned is to be merged with other data sets provided for additional features/variables which may help us in improving the classification.

Datasets to be merged:

- bureau.csv contains 1716428 rows and 17 columns

- bureau_balance.csv contains 27299925 rows and 3 columns

- POS_CASH_balance.csv contains 10001358 rows and 8 columns

- credit_card_balance.csv contains 3840312 rows and 23 columns

- previous_application.csv contains 1670214 rows and 37 columns

- installments_payments.csv contains 13605401 rows and 8 columns

Common operations and feature engineering done to perform merging:

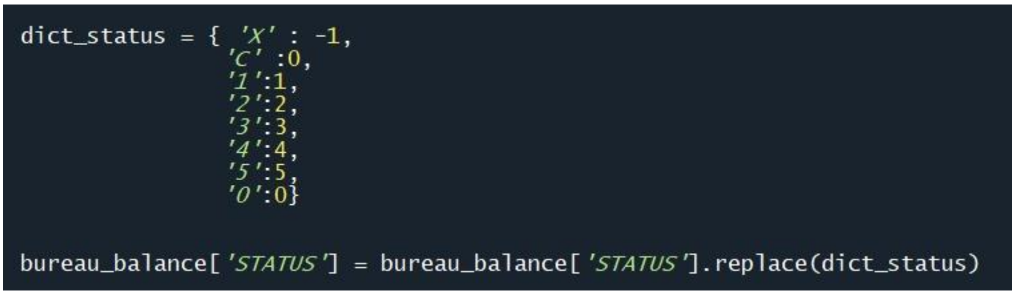
1. Missing values are removed which has missing value greater than 60%, other missing values imputed with zero.

2. Aggregated the value with below condition.

   a. Numeric values = min, max, mean.
   b. Binary values = mode, mean.
   c. categorical values (encoded) = mean.

Feature Engineering & Observed Correction Meta data:

1. **bureau_balance.csv:** converting the ordinal categorical feature into numerical feature



```
dict_status = { 'X' : -1,
               'C' :0,
               '1':1,
               '2':2,
               '3':3,
               '4':4,
               '5':5,
               '0':0}

bureau_balance['STATUS'] = bureau_balance['STATUS'].replace(dict_status)
```

a. Total number of dropped columns for threshold 0.6: 0
b. Total number of columns with missing value :0
c. shape of merged df = (1716428, 27)

2. **bureau.csv:**

   a. Total number of dropped columns for threshold 0.6: 2
   b. Total number of columns with missing value :14
   c. Columns dropped: ['AMT_CREDIT_MAX_OVERDUE', 'AMT_ANNUITY']

3. **POS_CASH_balance.csv:**

   a. Total number of dropped columns for threshold 0.6: 0
   b. Total number of columns with missing value :2

4. **credit_card_balance.csv:**

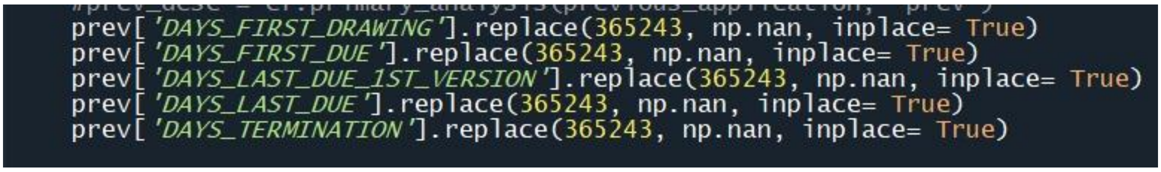
   a. Total number of dropped columns for threshold 0.6: 0
   b. Total number of columns with missing value :9
   c. shape of merged df = (265134, 260)

5. **installments_payments.csv:**

   a. Total number of dropped columns for threshold 0.6: 0
   b. Total number of columns with missing value :2

6. **previous_application.csv:**

   a. Observer correction in previous_application.csv



```
#prev_desc = cf.primary_analysis(previous_application, 'prev')
prev['DAYS_FIRST_DRAWING'].replace(365243, np.nan, inplace= True)
prev['DAYS_FIRST_DUE'].replace(365243, np.nan, inplace= True)
prev['DAYS_LAST_DUE_1ST_VERSION'].replace(365243, np.nan, inplace= True)
prev['DAYS_LAST_DUE'].replace(365243, np.nan, inplace= True)
prev['DAYS_TERMINATION'].replace(365243, np.nan, inplace= True)
```

   b. Total number of dropped columns for threshold 0.6: 3
   c. Total number of columns with missing value :13
   d. Columns dropped: ['RATE_INTEREST_PRIMARY', 'RATE_INTEREST_PRIVILEGED', 'DAYS_FIRST_DRAWING']

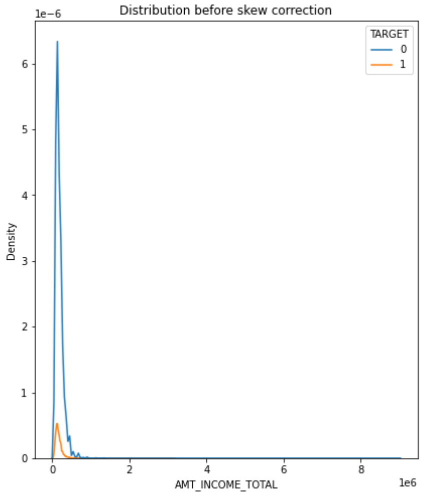
- df (final data frame) contains 265134rows and 477columns

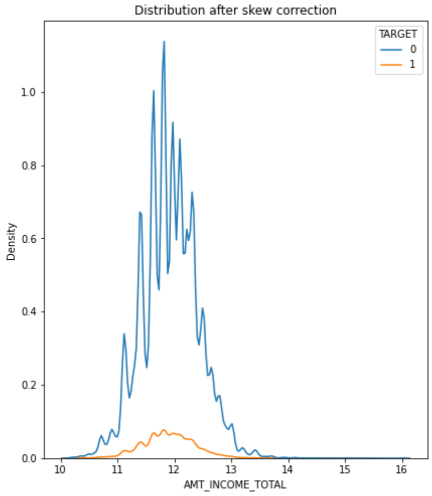
- Total number of dropped features from df for missing value greater than 0.45 = 71

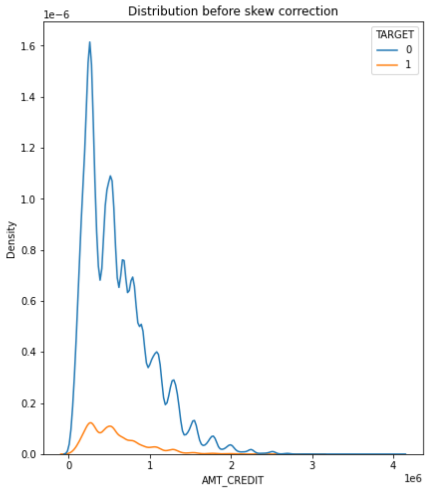
- Other features which have missing values are imputed with an arbitrary value (-999)

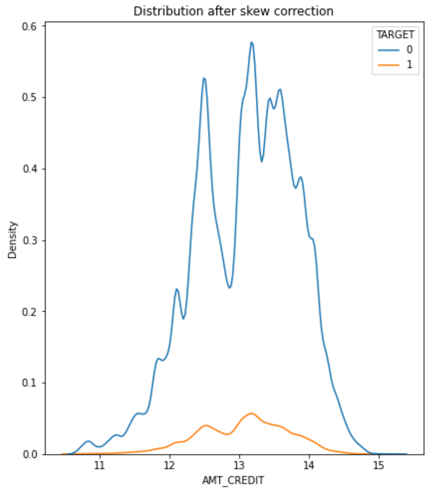
SKEW CORRECTION

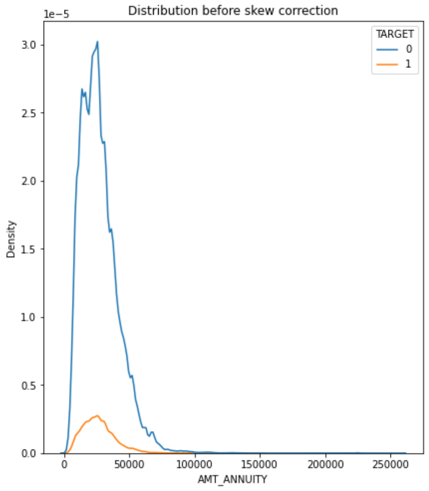
If we have a skewed data then it may harm our results. So, in order to use a skewed data, we have to apply a **log transformation** over the whole set of values to discover patterns in the data and make it usable for the model. The visualization of the skew in the features is shown in below graphs.

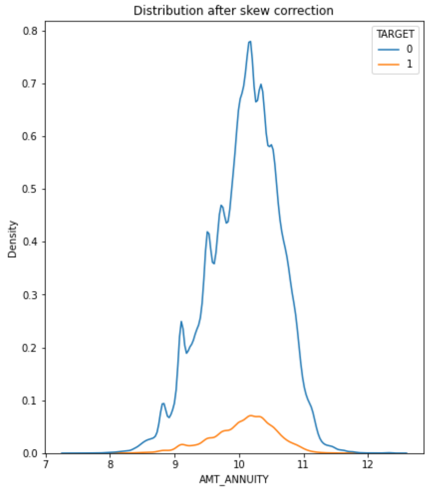


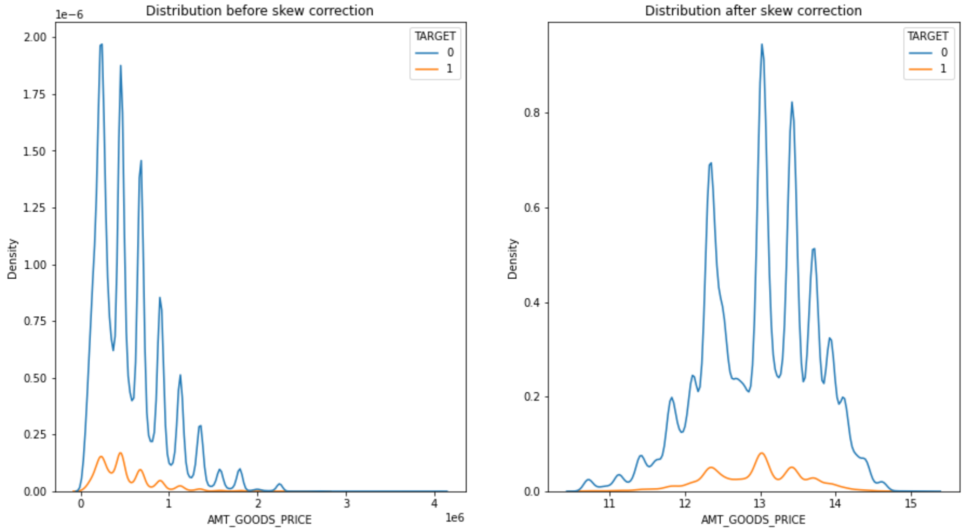


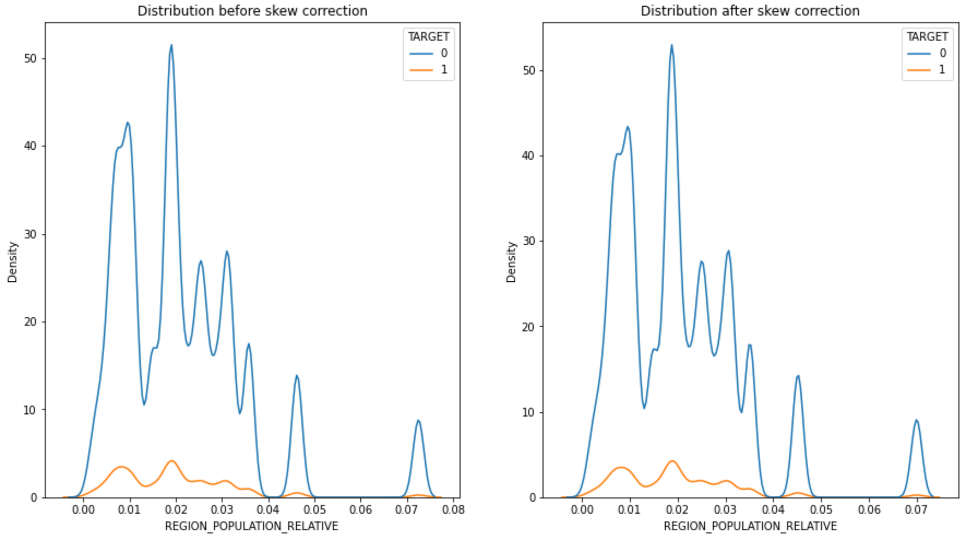


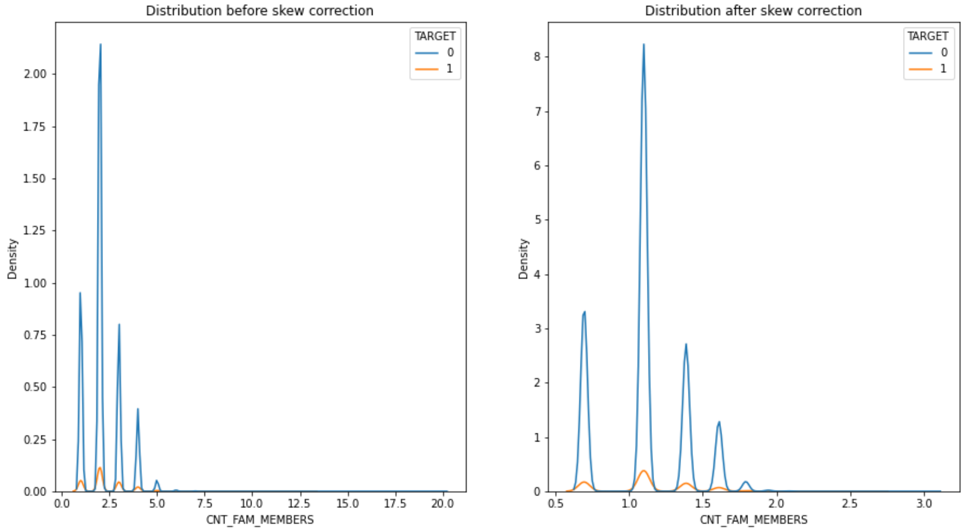


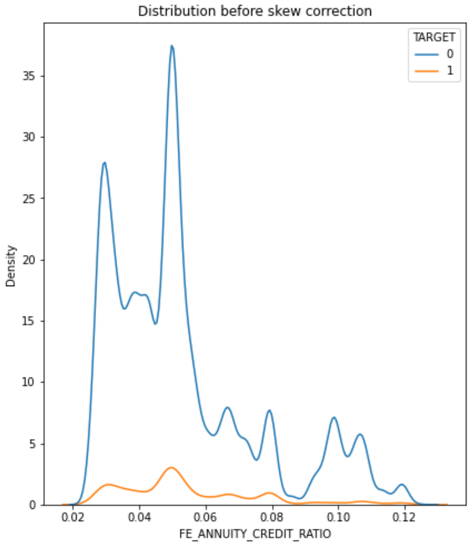


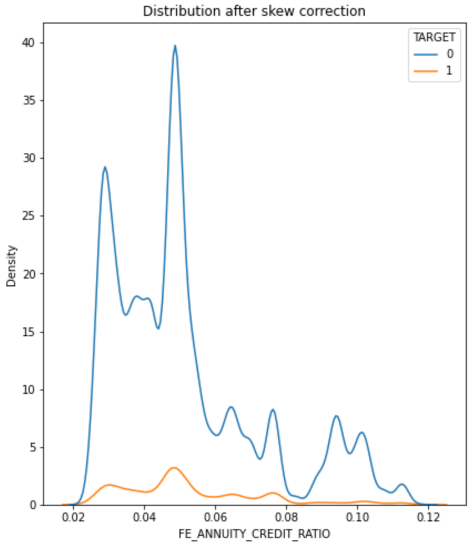


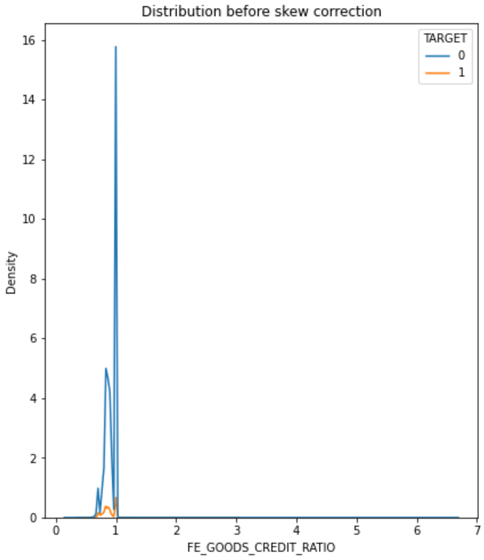

Distribution before skew correction
1e−6
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Density
0
1
2
3
4
1e6
AMT_GOODS_PRICE
TARGET
0
1
Distribution after skew correction
0.8
0.6
0.4
0.2
0.0
Density
11
12
13
14
15
AMT_GOODS_PRICE
TARGET
0
1

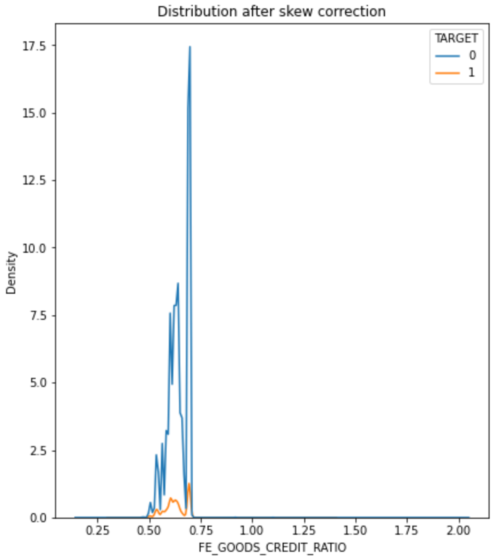

Distribution before skew correction
50
40
30
20
10
0
Density
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
REGION_POPULATION_RELATIVE
TARGET
0
1
Distribution after skew correction
50
40
30
20
10
0
Density
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
REGION_POPULATION_RELATIVE
TARGET
0
1

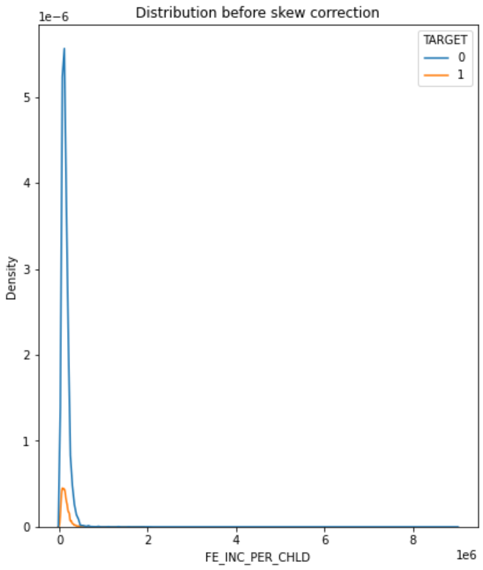

Distribution before skew correction
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Density
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
CNT_FAM_MEMBERS
TARGET
0
1
Distribution after skew correction
8
7
6
5
4
3
2
1
0
Density
0.5
1.0
1.5
2.0
2.5
3.0
CNT_FAM_MEMBERS
TARGET
0
1

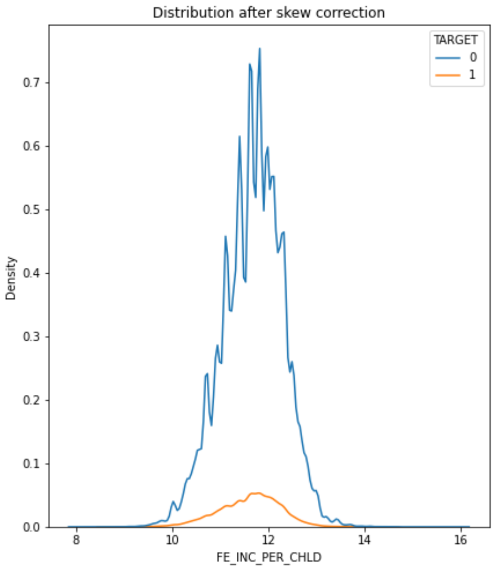

Distribution before skew correction
TARGET
0
1
Density
0
5
10
15
20
25
30
35
0.02
0.04
0.06
0.08
0.10
0.12
FE_ANNUITY_CREDIT_RATIO

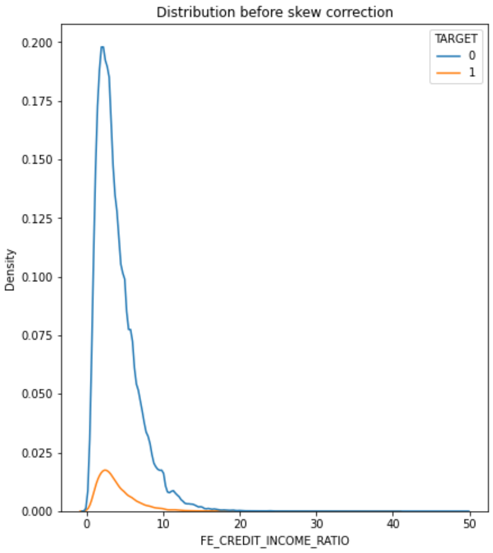

Distribution after skew correction
TARGET
0
1
Density
0
5
10
15
20
25
30
35
40
0.02
0.04
0.06
0.08
0.10
0.12
FE_ANNUITY_CREDIT_RATIO

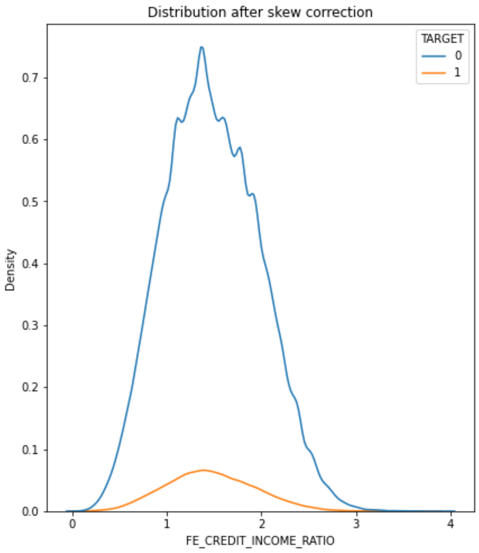

Distribution before skew correction
TARGET
0
1
Density
0
2
4
6
8
10
12
14
16
0
1
2
3
4
5
6
7
FE_GOODS_CREDIT_RATIO

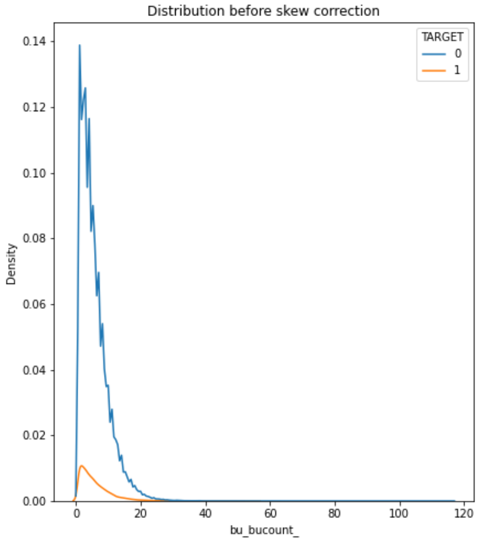

Distribution after skew correction
TARGET
0
1
Density
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
FE_GOODS_CREDIT_RATIO

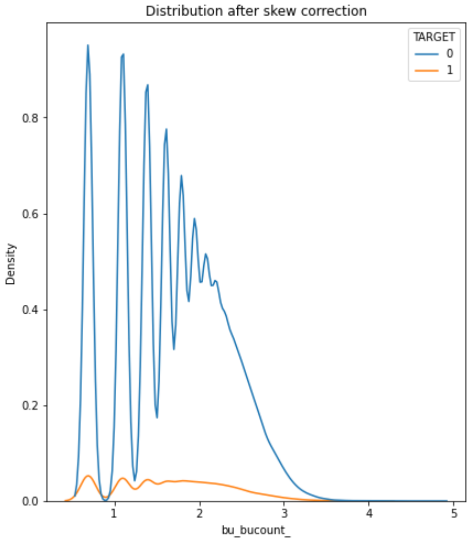

1e−6
Distribution before skew correction
TARGET
0
1
Density
0
1
2
3
4
5
0
2
4
6
8
1e6
FE_INC_PER_CHLD

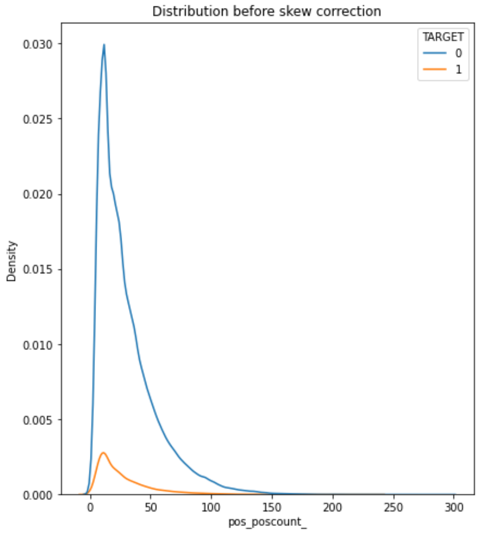

Distribution after skew correction
TARGET
0
1
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
8
10
12
14
16
FE_INC_PER_CHLD

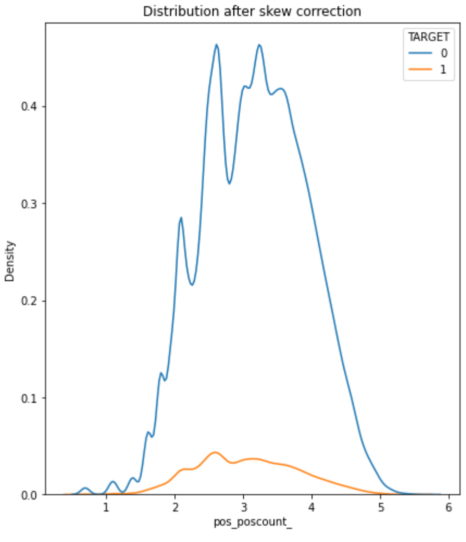

Distribution before skew correction
TARGET
0
1
Density
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0
10
20
30
40
50
FE_CREDIT_INCOME_RATIO

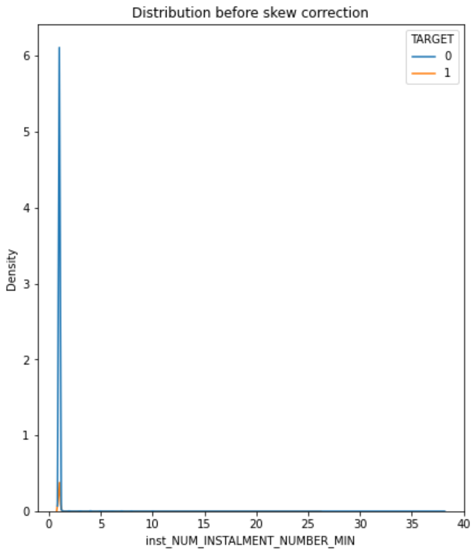

Distribution after skew correction
TARGET
0
1
Density
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
1
2
3
4
FE_CREDIT_INCOME_RATIO

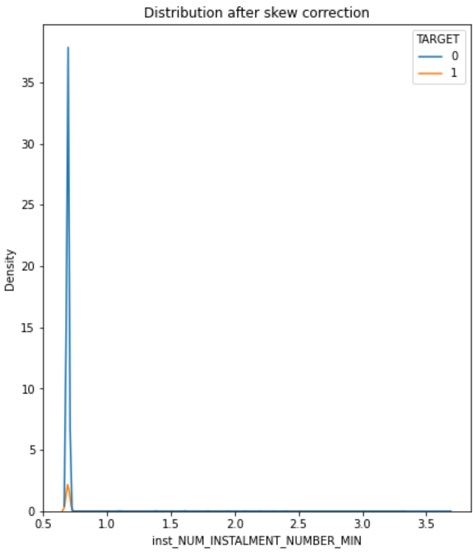

Distribution before skew correction
TARGET
0
1
Density
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
20
40
60
80
100
120
bu_bucount_

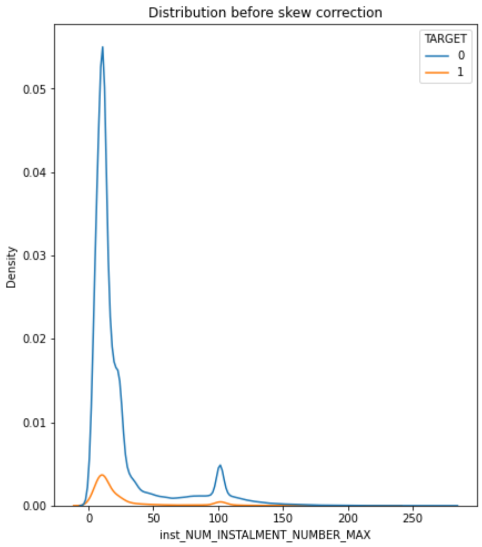

Distribution after skew correction
TARGET
0
1
Density
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
bu_bucount_

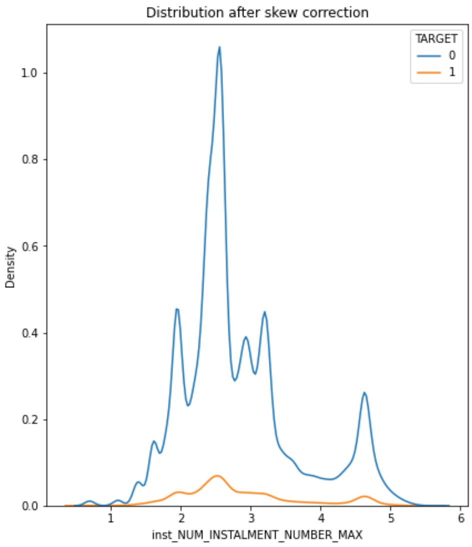

Distribution before skew correction
TARGET
0
1
Density
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
50
100
150
200
250
300
pos_poscount_

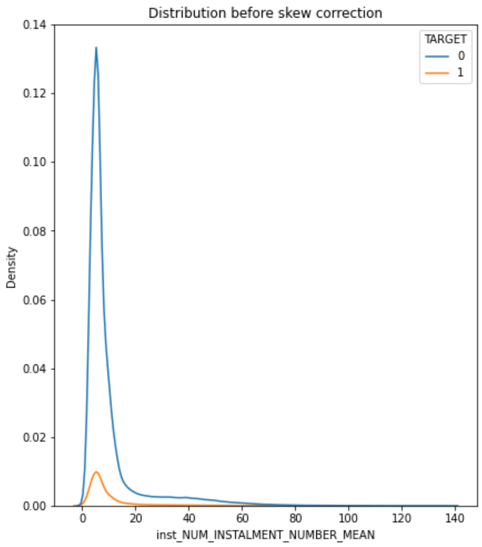

Distribution after skew correction
TARGET
0
1
Density
0.4
0.3
0.2
0.1
0.0
1
2
3
4
5
6
pos_poscount_

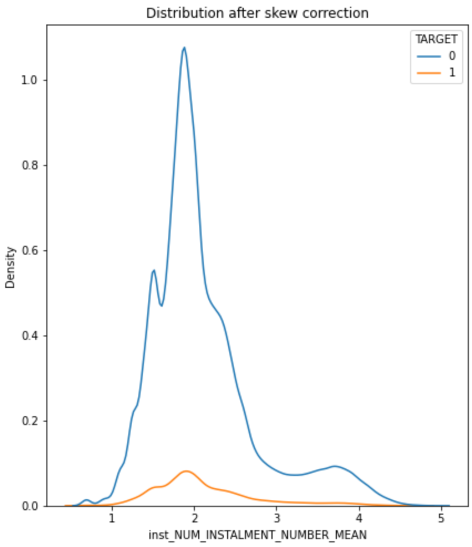

Distribution before skew correction
TARGET
0
1
Density
0
1
2
3
4
5
6
0
5
10
15
20
25
30
35
40
inst_NUM_INSTALMENT_NUMBER_MIN


Distribution after skew correction
TARGET
0
1
Density
0
5
10
15
20
25
30
35
0.5
1.0
1.5
2.0
2.5
3.0
3.5
inst_NUM_INSTALMENT_NUMBER_MIN


Distribution before skew correction
TARGET
0
1
Density
0.00
0.01
0.02
0.03
0.04
0.05
0
50
100
150
200
250
inst_NUM_INSTALMENT_NUMBER_MAX


Distribution after skew correction
TARGET
0
1
Density
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
6
inst_NUM_INSTALMENT_NUMBER_MAX


Distribution before skew correction
TARGET
0
1
Density
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0
20
40
60
80
100
120
140
inst_NUM_INSTALMENT_NUMBER_MEAN


Distribution after skew correction
TARGET
0
1
Density
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
4
5
inst_NUM_INSTALMENT_NUMBER_MEAN

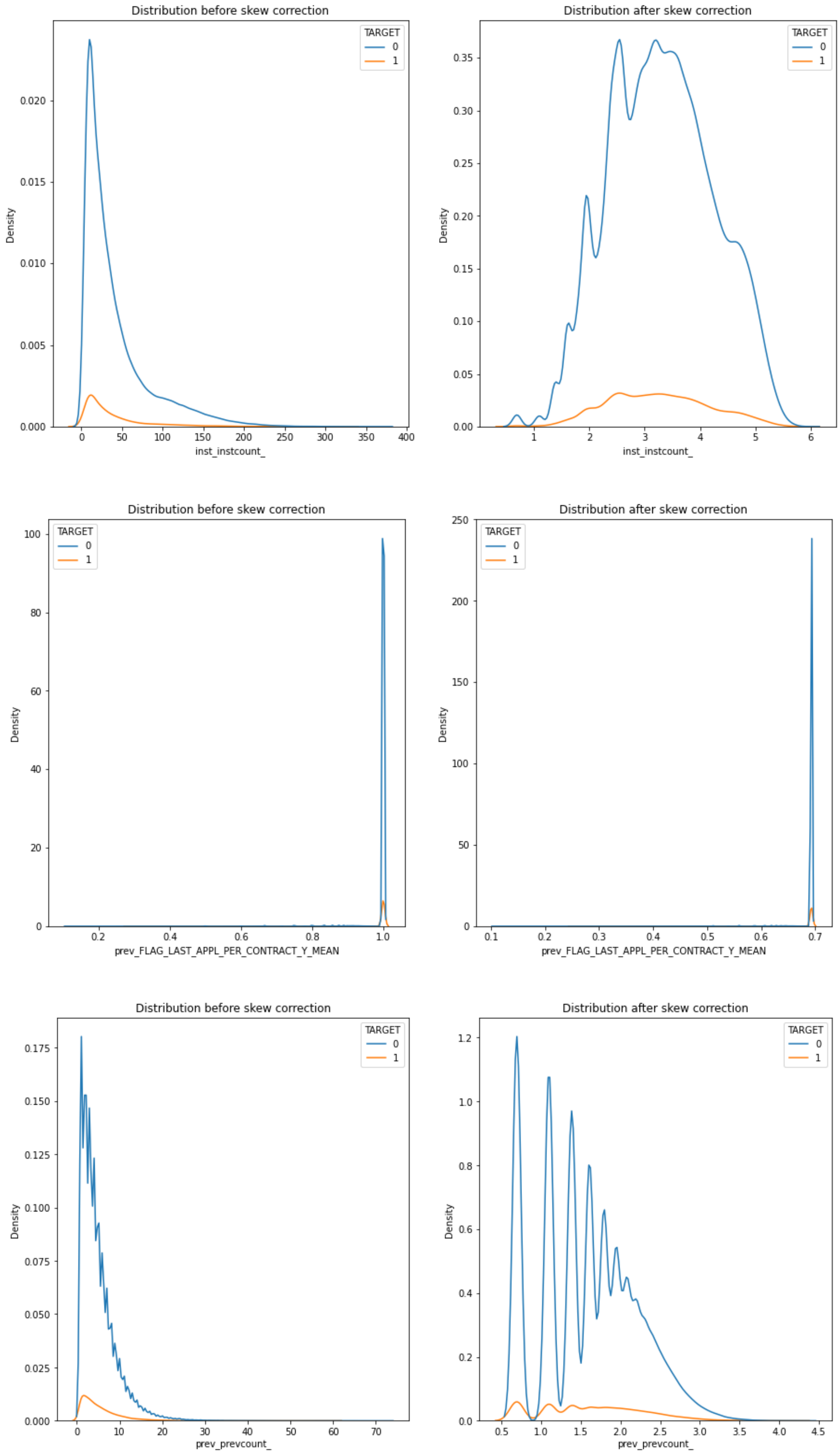


Fig: Distribution of data before and after skew correction

skew corrected for total number of columns: 18

Model building

- **Standard Scaler** is used to scale the **numerical features** for reducing the computational cost.
- **One Hot Encoder** is used for encoding the **categorical features**.
- Independent features and dependent feature are separated
- **Cross-validation** is done to ensure the stability of the data

Mean performance metric = 0.92
SDT of the metric = 0.0010347872990414194

Outcomes by cv fold

Fold 1 = 0.9225866068229392
Fold 2 = 0.923755822505516
Fold 3 = 0.9208704999339958
Fold 4 = 0.9220774322514945
Fold 5 = 0.9211707464262815

From the above observation of cross validated score using XG Boost, the **data is stable**.

- **Train and test are divided** where the test data is of **30% size** of the train data size.
- Over sampling of the data is done to maintain the balance in the training set.

Records supporting class **0** after over sampling = 171302
Records supporting class **1** after over sampling = 171302

- Reducing the size of the data frame by **reducing the memory size**, that is from 64 bit to 32 bit.

Original Memory Usage: 0.87 GB.
New Memory Usage: 0.45 GB.

- **Feature importance using** XG boot model:

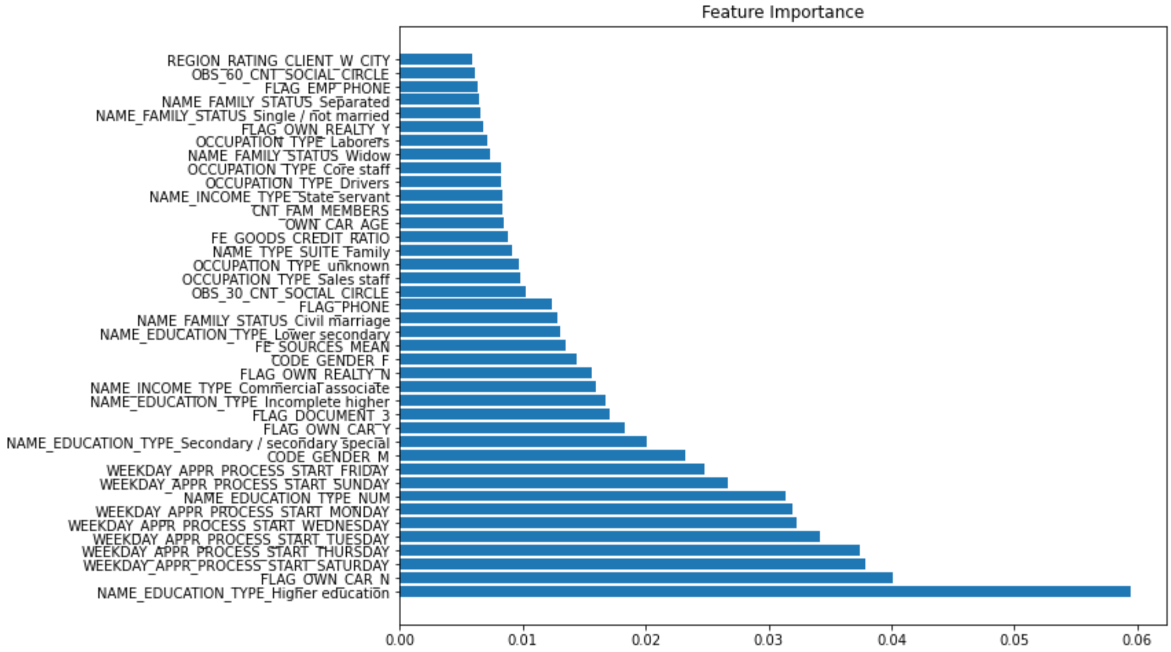


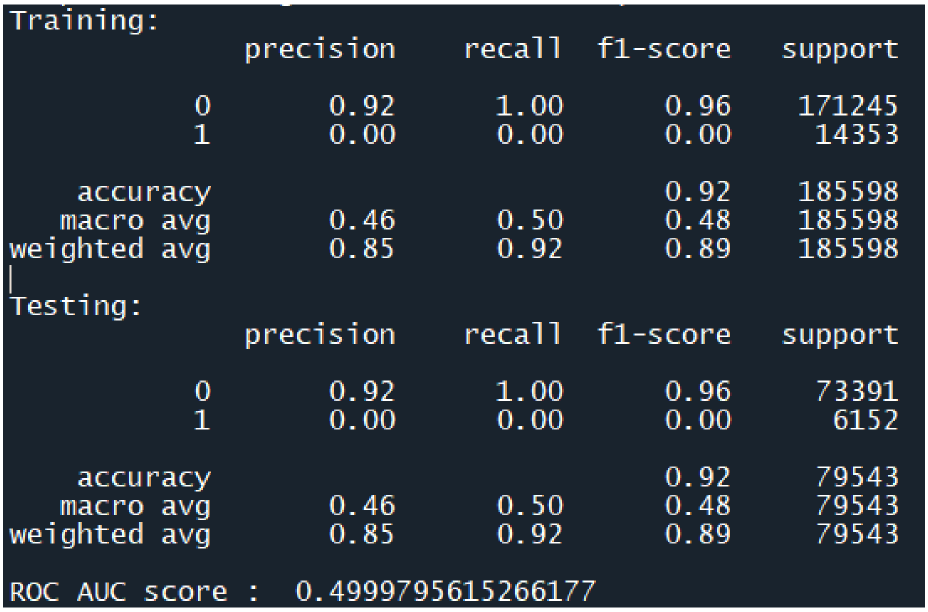
Fig: Feature importance score

Model used for classification: Logistic regression(random_state=89)

Logistic regression is a **simple and more efficient method for binary and linear classification problems**. It is a classification model, which is very easy to realize and achieves very good performance with linearly separable classes

Metrics used to evaluate: ROC AUC score

Initial stage Scores and Evaluation



```
Training:
              precision    recall  f1-score   support

           0       0.92      1.00      0.96    171245
           1       0.00      0.00      0.00     14353

    accuracy                           0.92    185598
   macro avg       0.46      0.50      0.48    185598
weighted avg       0.85      0.92      0.89    185598

Testing:
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     73391
           1       0.00      0.00      0.00      6152

    accuracy                           0.92     79543
   macro avg       0.46      0.50      0.48     79543
weighted avg       0.85      0.92      0.89     79543

ROC AUC score :  0.4999795615266177
```

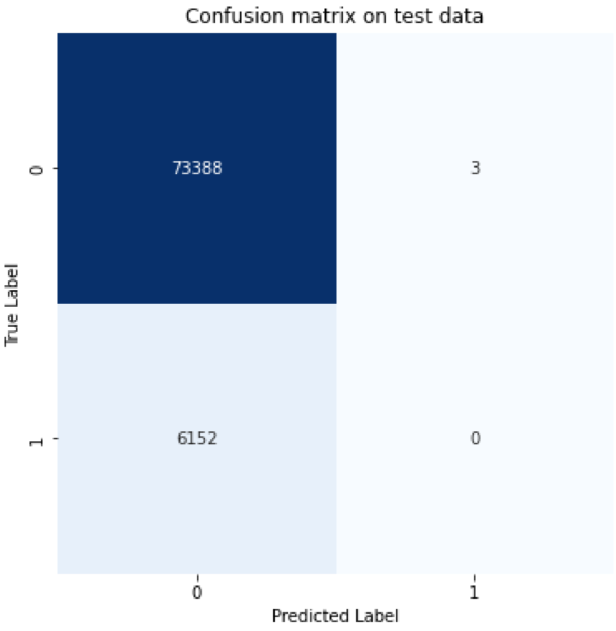


Fig: initial Confusion matrix

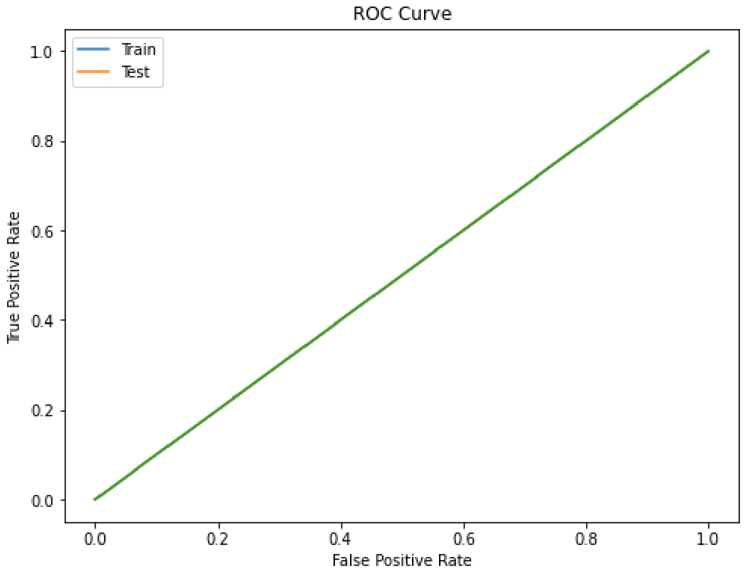
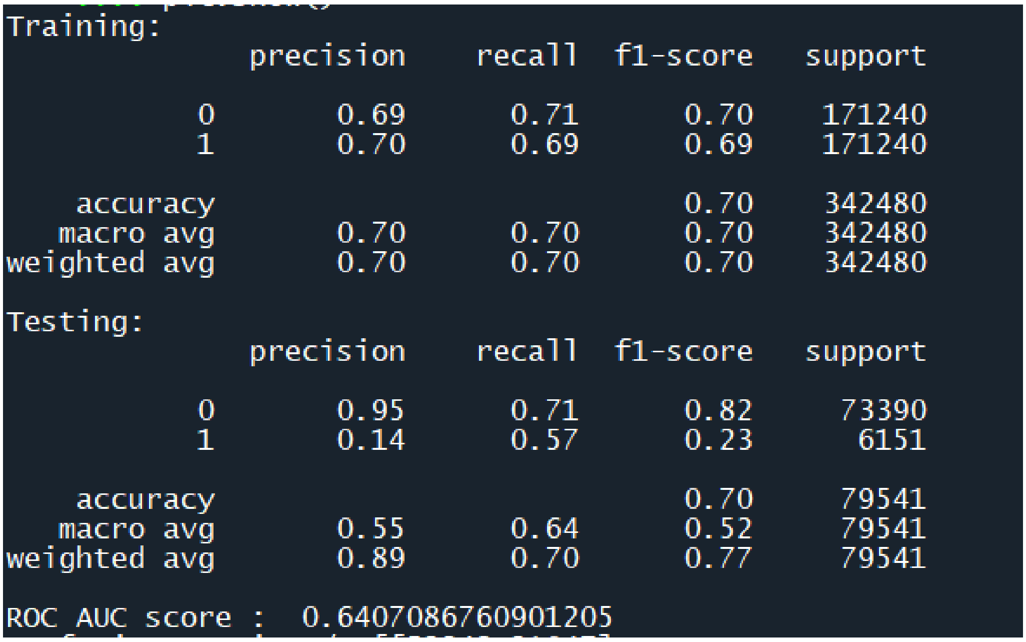


Fig: Initial ROC curve (AUC ROC score: 0.49)

After Optimizations Scores and Evaluation



```
Training:
              precision    recall  f1-score   support

           0       0.69      0.71      0.70    171240
           1       0.70      0.69      0.69    171240

    accuracy                           0.70    342480
   macro avg       0.70      0.70      0.70    342480
weighted avg       0.70      0.70      0.70    342480

Testing:
              precision    recall  f1-score   support

           0       0.95      0.71      0.82     73390
           1       0.14      0.57      0.23      6151

    accuracy                           0.70     79541
   macro avg       0.55      0.64      0.52     79541
weighted avg       0.89      0.70      0.77     79541

ROC AUC score :  0.6407086760901205
```

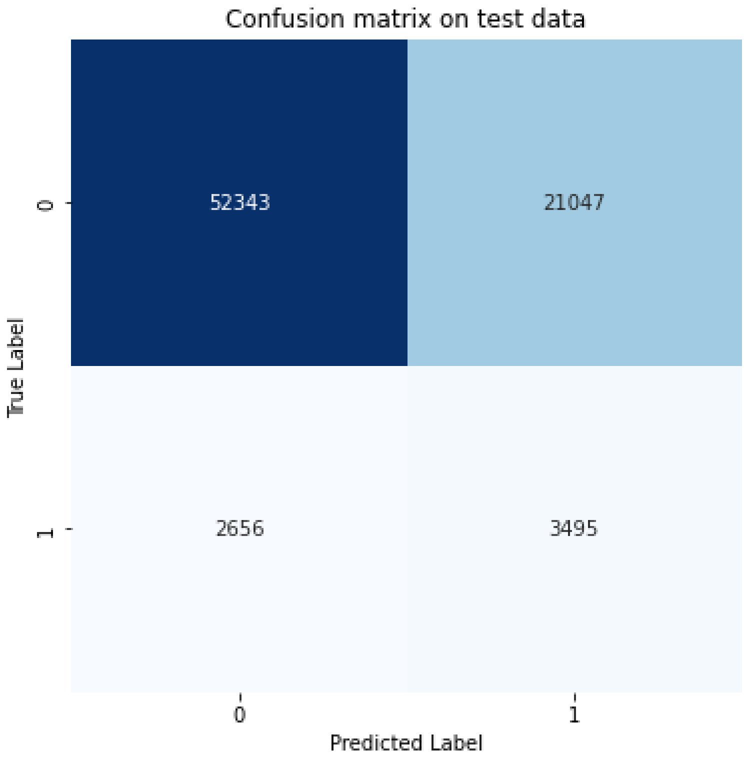


Fig: Final Confusion matrix

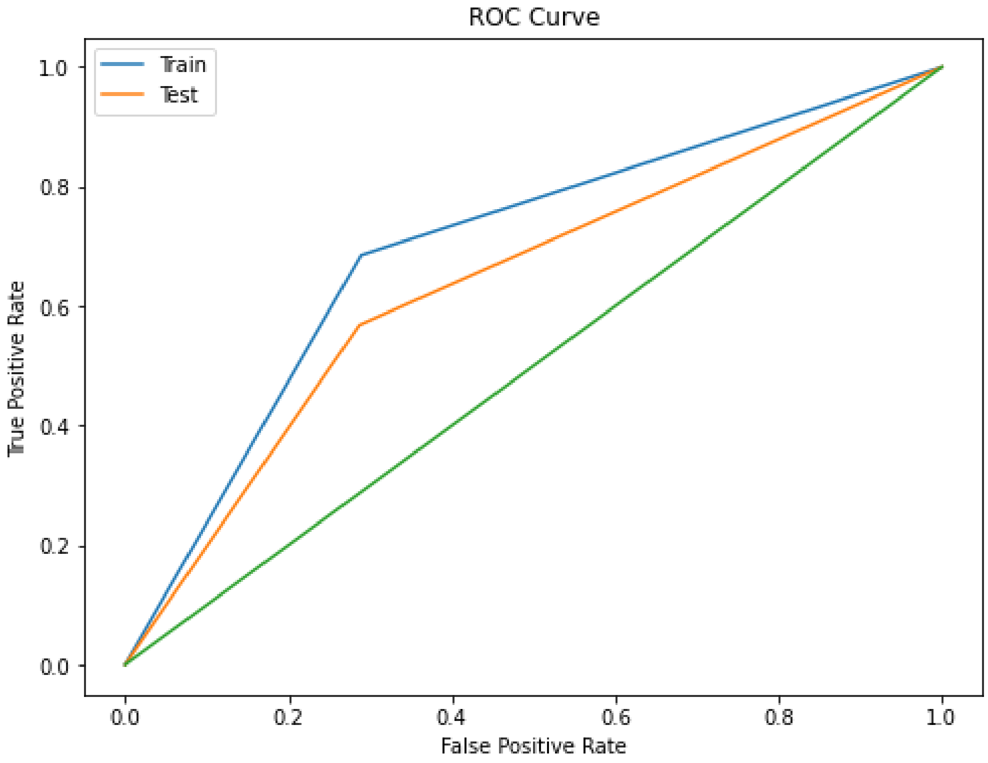


Fig: Final ROC curve (AUC ROC score: 0.64)