home_credit_default_risk

November 24, 2019

0.1 Table of Contents

0.1.1 Chapter 1 - Udacity Final Project

This directory contain all code that was used for the Udacity Data Scientist Nanodegree Program

0.1.2 Chapter 2 - Step 1: Define the Problem

For this project, the problem statement is given to us , develop an algorithm to predict the default of home credit .

Project Summary: 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.

In this project, we ask you to complete the analysis of which customers of home credit were likely default. In particular, we ask you to apply the tools of machine learning to predict which customers defaulted.

Project Metrics: Default customer can be predicted using less variable at credit risk perspective. So selected model specification must be explainable and applicable.

Practice Skills

- Binary classification
- Python

0.1.3 Chapter 3 - Step 2: Gather the Data

The dataset is given to us as test and train data at Kaggle's Home Credit Default Risk

3.1 Import Libraries The following code is written in Python 3.x. Libraries provide pre-written functionality to perform necessary tasks.

```
In [1]: #load packages

        import pandas as pd
        import numpy as np

        import seaborn as sns
        import matplotlib.pyplot as plt
        %matplotlib inline

        import random as rnd
```

3.11 Load Data Modelling Libraries We will use the popular scikit-learn library to develop our machine learning algorithms and for data visualization, we will use the matplotlib and seaborn library. Below are common classes to load.

```
In [2]: #Common Model Algorithms

        from sklearn.linear_model import LogisticRegression
        from sklearn.preprocessing import LabelEncoder
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.svm import SVC,LinearSVC
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.naive_bayes import GaussianNB
        from sklearn.linear_model import Perceptron
        from sklearn.linear_model import SGDClassifier
        from sklearn.tree import DecisionTreeClassifier

        #Visualization
        import matplotlib as mpl
        import matplotlib.pyplot as plt
        import matplotlib.pylab as pylab
        import seaborn as sns
```

0.1.4 Chapter 4 - Step 3: Prepare the Data

To begin this step, The data is imported firstly . Next we use the info() and head() function, to get a quick and dirty overview of variable datatypes (i.e. qualitative vs quantitative). Click here for the Source Data Dictionary.

```
In [3]: train_df = pd.read_csv("application_train.csv")
        test_df = pd.read_csv("application_test.csv")
        #bureau_df = pd.read_csv("bureau.csv")
        #bureau_balance_df = pd.read_csv("bureau_balance.csv")
        #credit_card_balance_df = pd.read_csv("credit_card_balance.csv")
        #HomeCredit_columns_description_df=pd.read_csv("HomeCredit_columns_description.csv")
```

```
#installments_payments_df=pd.read_csv("installments_payments.csv")
#POS_CASH_balance_df=pd.read_csv("POS_CASH_balance.csv")
#previous_application_df=pd.read_csv("previous_application.csv")
#sample_submission_df=pd.read_csv("sample_submission.csv")
```

In [4]:
```
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
```

In [5]:
```
# train_df
# preview the data

train_df.head(10)
```

Out[5]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_
0	100002	1	Cash loans	M	N	Y	
1	100003	0	Cash loans	F	N	N	
2	100004	0	Revolving loans	M	Y	Y	
3	100006	0	Cash loans	F	N	Y	
4	100007	0	Cash loans	M	N	Y	
5	100008	0	Cash loans	M	N	Y	
6	100009	0	Cash loans	F	Y	Y	
7	100010	0	Cash loans	M	Y	Y	
8	100011	0	Cash loans	F	N	Y	
9	100012	0	Revolving loans	M	N	Y	

	LIVINGAPARTMENTS_AVG	LIVINGAREA_AVG	NONLIVINGAPARTMENTS_AVG	NONLIVINGAREA_AVG	AI
0	0.0202	0.0190	0.0000	0.0000	
1	0.0773	0.0549	0.0039	0.0098	
2	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	
5	NaN	NaN	NaN	NaN	
6	NaN	NaN	NaN	NaN	
7	NaN	NaN	NaN	NaN	
8	NaN	NaN	NaN	NaN	
9	NaN	NaN	NaN	NaN	

	FLAG_DOCUMENT_11	FLAG_DOCUMENT_12	FLAG_DOCUMENT_13	FLAG_DOCUMENT_14	FLAG_DOCUMEN
0	0	0	0	0	
1	0	0	0	0	
2	0	0	0	0	
3	0	0	0	0	
4	0	0	0	0	
5	0	0	0	0	
6	0	0	0	1	
7	0	0	0	0	
8	0	0	0	0	
9	0	0	0	0	

```
In [6]: # train_df
        #data info

        train_df.info(max_cols=1000)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 122 columns):
SK_ID_CURR                      307511 non-null int64
TARGET                          307511 non-null int64
NAME_CONTRACT_TYPE              307511 non-null object
CODE_GENDER                     307511 non-null object
FLAG_OWN_CAR                    307511 non-null object
FLAG_OWN_REALTY                 307511 non-null object
CNT_CHILDREN                    307511 non-null int64
AMT_INCOME_TOTAL                307511 non-null float64
AMT_CREDIT                      307511 non-null float64
AMT_ANNUITY                     307499 non-null float64
AMT_GOODS_PRICE                 307233 non-null float64
NAME_TYPE_SUITE                 306219 non-null object
NAME_INCOME_TYPE                307511 non-null object
NAME_EDUCATION_TYPE             307511 non-null object
NAME_FAMILY_STATUS              307511 non-null object
NAME_HOUSING_TYPE               307511 non-null object
REGION_POPULATION_RELATIVE      307511 non-null float64
DAYS_BIRTH                      307511 non-null int64
DAYS_EMPLOYED                   307511 non-null int64
DAYS_REGISTRATION               307511 non-null float64
DAYS_ID_PUBLISH                 307511 non-null int64
OWN_CAR_AGE                     104582 non-null float64
FLAG_MOBIL                      307511 non-null int64
FLAG_EMP_PHONE                  307511 non-null int64
FLAG_WORK_PHONE                 307511 non-null int64
FLAG_CONT_MOBILE                307511 non-null int64
FLAG_PHONE                      307511 non-null int64
FLAG_EMAIL                      307511 non-null int64
OCCUPATION_TYPE                 211120 non-null object
CNT_FAM_MEMBERS                 307509 non-null float64
REGION_RATING_CLIENT            307511 non-null int64
REGION_RATING_CLIENT_W_CITY     307511 non-null int64
WEEKDAY_APPR_PROCESS_START      307511 non-null object
HOUR_APPR_PROCESS_START         307511 non-null int64
REG_REGION_NOT_LIVE_REGION      307511 non-null int64
REG_REGION_NOT_WORK_REGION      307511 non-null int64
LIVE_REGION_NOT_WORK_REGION     307511 non-null int64
REG_CITY_NOT_LIVE_CITY          307511 non-null int64
REG_CITY_NOT_WORK_CITY          307511 non-null int64
LIVE_CITY_NOT_WORK_CITY         307511 non-null int64
```

```
ORGANIZATION_TYPE              307511 non-null object
EXT_SOURCE_1                   134133 non-null float64
EXT_SOURCE_2                   306851 non-null float64
EXT_SOURCE_3                   246546 non-null float64
APARTMENTS_AVG                 151450 non-null float64
BASEMENTAREA_AVG               127568 non-null float64
YEARS_BEGINEXPLUATATION_AVG    157504 non-null float64
YEARS_BUILD_AVG                103023 non-null float64
COMMONAREA_AVG                 92646 non-null float64
ELEVATORS_AVG                  143620 non-null float64
ENTRANCES_AVG                  152683 non-null float64
FLOORSMAX_AVG                  154491 non-null float64
FLOORSMIN_AVG                  98869 non-null float64
LANDAREA_AVG                   124921 non-null float64
LIVINGAPARTMENTS_AVG           97312 non-null float64
LIVINGAREA_AVG                 153161 non-null float64
NONLIVINGAPARTMENTS_AVG        93997 non-null float64
NONLIVINGAREA_AVG              137829 non-null float64
APARTMENTS_MODE                151450 non-null float64
BASEMENTAREA_MODE              127568 non-null float64
YEARS_BEGINEXPLUATATION_MODE   157504 non-null float64
YEARS_BUILD_MODE               103023 non-null float64
COMMONAREA_MODE                92646 non-null float64
ELEVATORS_MODE                 143620 non-null float64
ENTRANCES_MODE                 152683 non-null float64
FLOORSMAX_MODE                 154491 non-null float64
FLOORSMIN_MODE                 98869 non-null float64
LANDAREA_MODE                  124921 non-null float64
LIVINGAPARTMENTS_MODE          97312 non-null float64
LIVINGAREA_MODE                153161 non-null float64
NONLIVINGAPARTMENTS_MODE       93997 non-null float64
NONLIVINGAREA_MODE             137829 non-null float64
APARTMENTS_MEDI                151450 non-null float64
BASEMENTAREA_MEDI              127568 non-null float64
YEARS_BEGINEXPLUATATION_MEDI   157504 non-null float64
YEARS_BUILD_MEDI               103023 non-null float64
COMMONAREA_MEDI                92646 non-null float64
ELEVATORS_MEDI                 143620 non-null float64
ENTRANCES_MEDI                 152683 non-null float64
FLOORSMAX_MEDI                 154491 non-null float64
FLOORSMIN_MEDI                 98869 non-null float64
LANDAREA_MEDI                  124921 non-null float64
LIVINGAPARTMENTS_MEDI          97312 non-null float64
LIVINGAREA_MEDI                153161 non-null float64
NONLIVINGAPARTMENTS_MEDI       93997 non-null float64
NONLIVINGAREA_MEDI             137829 non-null float64
FONDKAPREMONT_MODE             97216 non-null object
HOUSETYPE_MODE                 153214 non-null object
```

```
TOTALAREA_MODE                  159080 non-null float64
WALLSMATERIAL_MODE              151170 non-null object
EMERGENCYSTATE_MODE             161756 non-null object
OBS_30_CNT_SOCIAL_CIRCLE        306490 non-null float64
DEF_30_CNT_SOCIAL_CIRCLE        306490 non-null float64
OBS_60_CNT_SOCIAL_CIRCLE        306490 non-null float64
DEF_60_CNT_SOCIAL_CIRCLE        306490 non-null float64
DAYS_LAST_PHONE_CHANGE          307510 non-null float64
FLAG_DOCUMENT_2                 307511 non-null int64
FLAG_DOCUMENT_3                 307511 non-null int64
FLAG_DOCUMENT_4                 307511 non-null int64
FLAG_DOCUMENT_5                 307511 non-null int64
FLAG_DOCUMENT_6                 307511 non-null int64
FLAG_DOCUMENT_7                 307511 non-null int64
FLAG_DOCUMENT_8                 307511 non-null int64
FLAG_DOCUMENT_9                 307511 non-null int64
FLAG_DOCUMENT_10                307511 non-null int64
FLAG_DOCUMENT_11                307511 non-null int64
FLAG_DOCUMENT_12                307511 non-null int64
FLAG_DOCUMENT_13                307511 non-null int64
FLAG_DOCUMENT_14                307511 non-null int64
FLAG_DOCUMENT_15                307511 non-null int64
FLAG_DOCUMENT_16                307511 non-null int64
FLAG_DOCUMENT_17                307511 non-null int64
FLAG_DOCUMENT_18                307511 non-null int64
FLAG_DOCUMENT_19                307511 non-null int64
FLAG_DOCUMENT_20                307511 non-null int64
FLAG_DOCUMENT_21                307511 non-null int64
AMT_REQ_CREDIT_BUREAU_HOUR      265992 non-null float64
AMT_REQ_CREDIT_BUREAU_DAY       265992 non-null float64
AMT_REQ_CREDIT_BUREAU_WEEK      265992 non-null float64
AMT_REQ_CREDIT_BUREAU_MON       265992 non-null float64
AMT_REQ_CREDIT_BUREAU_QRT       265992 non-null float64
AMT_REQ_CREDIT_BUREAU_YEAR      265992 non-null float64
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
```

In [7]:
```python
# train_df
# data describe

train_df.describe()
```

Out[7]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT
count	307511.000000	307511.000000	307511.000000	3.075110e+05	3.075110e+05
mean	278180.518577	0.080729	0.417052	1.687979e+05	5.990260e+05
std	102790.175348	0.272419	0.722121	2.371231e+05	4.024908e+05
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04

```
25%     189145.500000        0.000000        0.000000       1.125000e+05  2.700000e+05   16
50%     278202.000000        0.000000        0.000000       1.471500e+05  5.135310e+05   24
75%     367142.500000        0.000000        1.000000       2.025000e+05  8.086500e+05   34
max     456255.000000        1.000000       19.000000       1.170000e+08  4.050000e+06  258

        ENTRANCES_MODE  FLOORSMAX_MODE  FLOORSMIN_MODE  LANDAREA_MODE  LIVINGAPARTMENTS_
count    152683.000000   154491.000000    98869.000000  124921.000000           97312.00
mean          0.145193        0.222315        0.228058       0.064958               0.10
std           0.100977        0.143709        0.161160       0.081750               0.09
min           0.000000        0.000000        0.000000       0.000000               0.00
25%           0.069000        0.166700        0.083300       0.016600               0.05
50%           0.137900        0.166700        0.208300       0.045800               0.07
75%           0.206900        0.333300        0.375000       0.084100               0.13
max           1.000000        1.000000        1.000000       1.000000               1.00

        AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  AMT_REQ_CREDIT_BUREAU_QR
count                265992.000000              265992.000000               265992.00000
mean                      0.034362                   0.267395                    0.26547
std                       0.204685                   0.916002                    0.79405
min                       0.000000                   0.000000                    0.00000
25%                       0.000000                   0.000000                    0.00000
50%                       0.000000                   0.000000                    0.00000
75%                       0.000000                   0.000000                    0.00000
max                       8.000000                  27.000000                  261.00000
```

In [8]:
```
# train_df
# data describe for object

categorical_varaible=train_df.describe(include=['O'])
categorical_varaible
```

Out[8]:
```
       NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR FLAG_OWN_REALTY NAME_TYPE_SUITE NAM
count              307511      307511       307511          307511          306219
unique                  2           3            2               2               7
top            Cash loans           F            N               Y   Unaccompanied
freq               278232      202448       202924          213312          248526
```

What is the distribution of categorical features?

- Contract type as two possible values with 90% Cash loans (top=Cash loans, freq=278232/count=307511).
- Gender variable as three possible values with 66% female (top=female, freq=202448/count=307511).
- Own Car variable as two possible values with 66% "No" (top=N, freq=202924/count=307511).
- Own Realty variable as two possible values with 69% "Yes" (top=Y, freq=213312/count=307511).
- Suite Type variable as seven possible values with 81% unaccompanied (top=Unaccompanied, freq=248526/count=306219).

- Income Type variable as eight possible values with 81% Working (top=Working, freq=248526/count=307511).
- Education Type variable as five possible values with 71% unaccompanied (top=Secondary / secondary special, freq=218391/count=307511).
- Family status variable as six possible values with 64% "Married" (top=Married, freq=196432/count=307511).
- Housing type variable as six possible values with 89% "House / apartment " (top=House / apartment, freq=272868/count=307511).
- Occupation type variable as eighteen possible values with 26% "Laborers" (top=Laborers, freq=55186/count=211120).
- Weekday aproval process start day variable as seven possible values with 18% "TUESDAY" (top=TUESDAY, freq=53901/count=307511).
- Organization type variable as fifty eight possible values with 22% "Business Entity Type 3" (top=Business Entity Type 3, freq=67992/count=307511).
- Fondkapremont mode variable as four possible values with 76% "reg oper account" (top=reg oper account, freq=73830/count=97216).
- House type variable as three possible values with 98% "block of flats" (top=block of flats, freq=150503/count=153214).
- Walls material variable as seven possible values with 44% "Panel" (top=No, freq=66040/count=151170).
- Emergency state variable as two possible values with 99% "No" (top=No, freq=159428/count=161756).

In [9]:
```
# bureau_df
# preview the data

bureau_df.head(10)
```

Out[9]:

	SK_ID_CURR	SK_ID_BUREAU	CREDIT_ACTIVE	CREDIT_CURRENCY	DAYS_CREDIT	CREDIT_DAY_OVEI
0	215354	5714462	Closed	currency 1	-497	
1	215354	5714463	Active	currency 1	-208	
2	215354	5714464	Active	currency 1	-203	
3	215354	5714465	Active	currency 1	-203	
4	215354	5714466	Active	currency 1	-629	
5	215354	5714467	Active	currency 1	-273	
6	215354	5714468	Active	currency 1	-43	
7	162297	5714469	Closed	currency 1	-1896	
8	162297	5714470	Closed	currency 1	-1146	
9	162297	5714471	Active	currency 1	-1146	

0.1.5 Chapter 5 - The 4 C's of Data Cleaning: Correcting, Completing, Creating, and Converting

In this stage, data should have been cleaned 1. Correcting abnormal values and outliers 2. Completing missing information 3. Creating new features for analysis 4. Converting fields to the correct format for calculations and presentation.

Correcting: Reviewing the data, there should have been analyzed to be any abnormal or non-acceptable data inputs. In addition, age and income may have outlier values.Exploratory analysis will done to find reasonable values. Outliers should been elimated in dataset.It should be noted, that if unreasonable values were , for example age is 1000 then it also should be elimaneted.

Completing: There are null values or missing data in dataset. Missing values can be bad, because some algorithms don't know how-to handle null values and will fail. While others, like decision trees, can handle null values. Thus, it's important to fix before modeling will started because several models will have compared. There are two common methods, either delete the record or populate the missing value using a reasonable input. It is not recommended to delete the record, especially a large percentage of records, unless it truly represents an incomplete record. Instead, it's best to impute missing values. A basic methodology for qualitative data is impute using mode. A basic methodology for quantitative data is impute using mean, median, or mean + randomized standard deviation.

Creating: Feature engineering is when we use existing features to create new features to determine if they provide new signals to predict our outcome.

Converting: Last, but certainly not least, we'll deal with formatting. There are no date or currency formats, but datatype formats. Our categorical data imported as objects, which makes it difficult for mathematical calculations. For this dataset, we will convert object datatypes to categorical dummy variables

0.1.6 5.1 Correcting

We have been analyzed for dataset. We have seen the maximumum count of childred variable. So maximum age is 19. Outlier have elimated for count of children=19.
We have not seen any anormaly dataset. We check this step for dataset

```
In [29]: train_df=train_df[train_df.CNT_CHILDREN !=19 ]

In [30]: test_df=test_df[test_df.CNT_CHILDREN !=19 ]

In [31]: train_df=train_df[train_df.DAYS_LAST_PHONE_CHANGE.notnull()]

In [32]: test_df=test_df[test_df.DAYS_LAST_PHONE_CHANGE.notnull()]

In [33]: train_df.describe()
```

Out[33]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT
count	307508.000000	307508.00000	307508.000000	3.075080e+05	3.075080e+05
mean	278180.610368	0.08073	0.416932	1.687984e+05	5.990296e+05
std	102790.006413	0.27242	0.720568	2.371242e+05	4.024910e+05
min	100002.000000	0.00000	0.000000	2.565000e+04	4.500000e+04
25%	189145.750000	0.00000	0.000000	1.125000e+05	2.700000e+05
50%	278201.500000	0.00000	0.000000	1.471500e+05	5.135310e+05
75%	367142.250000	0.00000	1.000000	2.025000e+05	8.086500e+05
max	456255.000000	1.00000	14.000000	1.170000e+08	4.050000e+06

	FLOORSMAX_MODE	FLOORSMIN_MODE	LANDAREA_MODE	LIVINGAPARTMENTS_MODE
count	154489.000000	98868.000000	124919.000000	97311.000000
mean	0.222314	0.228059	0.064958	0.105645
std	0.143710	0.161161	0.081751	0.097881
min	0.000000	0.000000	0.000000	0.000000

```
25%          0.166700       0.083300      0.016600               0.054200      0
50%          0.166700       0.208300      0.045800               0.077100      0
75%          0.333300       0.375000      0.084100               0.131300      0
max          1.000000       1.000000      1.000000               1.000000      1

       AMT_REQ_CREDIT_BUREAU_MON  AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEA
count              265990.000000              265990.000000              265990.0000
mean                    0.267397                   0.265476                   1.8999
std                     0.916006                   0.794058                   1.8692
min                     0.000000                   0.000000                   0.0000
25%                     0.000000                   0.000000                   0.0000
50%                     0.000000                   0.000000                   1.0000
75%                     0.000000                   0.000000                   3.0000
max                    27.000000                 261.000000                  25.0000
```

0.1.7 5.2 Completing

We can analyze the missing value. But some data can not completed for missing value. Because some values normally is missingç For example customers who have no credit bureau information and related coloumns have no information about customer. We can filter variable of occupation_type because of 96391 missing value.

In [34]:
```
train_df.isnull().sum()
```

Out[34]:
```
SK_ID_CURR                         0
TARGET                             0
NAME_CONTRACT_TYPE                 0
CODE_GENDER                        0
FLAG_OWN_CAR                       0
FLAG_OWN_REALTY                    0
CNT_CHILDREN                       0
AMT_INCOME_TOTAL                   0
AMT_CREDIT                         0
AMT_ANNUITY                       12
AMT_GOODS_PRICE                  278
NAME_TYPE_SUITE                 1292
NAME_INCOME_TYPE                   0
NAME_EDUCATION_TYPE                0
NAME_FAMILY_STATUS                 0
NAME_HOUSING_TYPE                  0
REGION_POPULATION_RELATIVE         0
DAYS_BIRTH                         0
DAYS_EMPLOYED                      0
DAYS_REGISTRATION                  0
DAYS_ID_PUBLISH                    0
OWN_CAR_AGE                   202927
FLAG_MOBIL                         0
FLAG_EMP_PHONE                     0
```

FLAG_WORK_PHONE	0
FLAG_CONT_MOBILE	0
FLAG_PHONE	0
FLAG_EMAIL	0
OCCUPATION_TYPE	96390
CNT_FAM_MEMBERS	2
REGION_RATING_CLIENT	0
REGION_RATING_CLIENT_W_CITY	0
WEEKDAY_APPR_PROCESS_START	0
HOUR_APPR_PROCESS_START	0
REG_REGION_NOT_LIVE_REGION	0
REG_REGION_NOT_WORK_REGION	0
LIVE_REGION_NOT_WORK_REGION	0
REG_CITY_NOT_LIVE_CITY	0
REG_CITY_NOT_WORK_CITY	0
LIVE_CITY_NOT_WORK_CITY	0
ORGANIZATION_TYPE	0
EXT_SOURCE_1	173376
EXT_SOURCE_2	659
EXT_SOURCE_3	60963
APARTMENTS_AVG	156060
BASEMENTAREA_AVG	179942
YEARS_BEGINEXPLUATATION_AVG	150006
YEARS_BUILD_AVG	204486
COMMONAREA_AVG	214863
ELEVATORS_AVG	163890
ENTRANCES_AVG	154827
FLOORSMAX_AVG	153019
FLOORSMIN_AVG	208640
LANDAREA_AVG	182589
LIVINGAPARTMENTS_AVG	210197
LIVINGAREA_AVG	154349
NONLIVINGAPARTMENTS_AVG	213512
NONLIVINGAREA_AVG	169680
APARTMENTS_MODE	156060
BASEMENTAREA_MODE	179942
YEARS_BEGINEXPLUATATION_MODE	150006
YEARS_BUILD_MODE	204486
COMMONAREA_MODE	214863
ELEVATORS_MODE	163890
ENTRANCES_MODE	154827
FLOORSMAX_MODE	153019
FLOORSMIN_MODE	208640
LANDAREA_MODE	182589
LIVINGAPARTMENTS_MODE	210197
LIVINGAREA_MODE	154349
NONLIVINGAPARTMENTS_MODE	213512
NONLIVINGAREA_MODE	169680

```
APARTMENTS_MEDI                 156060
BASEMENTAREA_MEDI               179942
YEARS_BEGINEXPLUATATION_MEDI    150006
YEARS_BUILD_MEDI                204486
COMMONAREA_MEDI                 214863
ELEVATORS_MEDI                  163890
ENTRANCES_MEDI                  154827
FLOORSMAX_MEDI                  153019
FLOORSMIN_MEDI                  208640
LANDAREA_MEDI                   182589
LIVINGAPARTMENTS_MEDI           210197
LIVINGAREA_MEDI                 154349
NONLIVINGAPARTMENTS_MEDI        213512
NONLIVINGAREA_MEDI              169680
FONDKAPREMONT_MODE              210293
HOUSETYPE_MODE                  154296
TOTALAREA_MODE                  148430
WALLSMATERIAL_MODE              156340
EMERGENCYSTATE_MODE             145754
OBS_30_CNT_SOCIAL_CIRCLE          1021
DEF_30_CNT_SOCIAL_CIRCLE          1021
OBS_60_CNT_SOCIAL_CIRCLE          1021
DEF_60_CNT_SOCIAL_CIRCLE          1021
DAYS_LAST_PHONE_CHANGE               0
FLAG_DOCUMENT_2                      0
FLAG_DOCUMENT_3                      0
FLAG_DOCUMENT_4                      0
FLAG_DOCUMENT_5                      0
FLAG_DOCUMENT_6                      0
FLAG_DOCUMENT_7                      0
FLAG_DOCUMENT_8                      0
FLAG_DOCUMENT_9                      0
FLAG_DOCUMENT_10                     0
FLAG_DOCUMENT_11                     0
FLAG_DOCUMENT_12                     0
FLAG_DOCUMENT_13                     0
FLAG_DOCUMENT_14                     0
FLAG_DOCUMENT_15                     0
FLAG_DOCUMENT_16                     0
FLAG_DOCUMENT_17                     0
FLAG_DOCUMENT_18                     0
FLAG_DOCUMENT_19                     0
FLAG_DOCUMENT_20                     0
FLAG_DOCUMENT_21                     0
AMT_REQ_CREDIT_BUREAU_HOUR       41518
AMT_REQ_CREDIT_BUREAU_DAY        41518
AMT_REQ_CREDIT_BUREAU_WEEK       41518
AMT_REQ_CREDIT_BUREAU_MON        41518
```

```
AMT_REQ_CREDIT_BUREAU_QRT        41518
AMT_REQ_CREDIT_BUREAU_YEAR       41518
dtype: int64
```

In [35]:
```
test_df.isnull().sum()
```

Out[35]:
```
SK_ID_CURR                          0
NAME_CONTRACT_TYPE                  0
CODE_GENDER                         0
FLAG_OWN_CAR                        0
FLAG_OWN_REALTY                     0
CNT_CHILDREN                        0
AMT_INCOME_TOTAL                    0
AMT_CREDIT                          0
AMT_ANNUITY                        24
AMT_GOODS_PRICE                     0
NAME_TYPE_SUITE                   911
NAME_INCOME_TYPE                    0
NAME_EDUCATION_TYPE                 0
NAME_FAMILY_STATUS                  0
NAME_HOUSING_TYPE                   0
REGION_POPULATION_RELATIVE          0
DAYS_BIRTH                          0
DAYS_EMPLOYED                       0
DAYS_REGISTRATION                   0
DAYS_ID_PUBLISH                     0
OWN_CAR_AGE                     32312
FLAG_MOBIL                          0
FLAG_EMP_PHONE                      0
FLAG_WORK_PHONE                     0
FLAG_CONT_MOBILE                    0
FLAG_PHONE                          0
FLAG_EMAIL                          0
OCCUPATION_TYPE                 15605
CNT_FAM_MEMBERS                     0
REGION_RATING_CLIENT                0
REGION_RATING_CLIENT_W_CITY         0
WEEKDAY_APPR_PROCESS_START          0
HOUR_APPR_PROCESS_START             0
REG_REGION_NOT_LIVE_REGION          0
REG_REGION_NOT_WORK_REGION          0
LIVE_REGION_NOT_WORK_REGION         0
REG_CITY_NOT_LIVE_CITY              0
REG_CITY_NOT_WORK_CITY              0
LIVE_CITY_NOT_WORK_CITY             0
ORGANIZATION_TYPE                   0
EXT_SOURCE_1                    20532
EXT_SOURCE_2                        8
```

EXT_SOURCE_3	8668
APARTMENTS_AVG	23887
BASEMENTAREA_AVG	27641
YEARS_BEGINEXPLUATATION_AVG	22856
YEARS_BUILD_AVG	31818
COMMONAREA_AVG	33495
ELEVATORS_AVG	25189
ENTRANCES_AVG	23579
FLOORSMAX_AVG	23321
FLOORSMIN_AVG	32466
LANDAREA_AVG	28254
LIVINGAPARTMENTS_AVG	32780
LIVINGAREA_AVG	23552
NONLIVINGAPARTMENTS_AVG	33347
NONLIVINGAREA_AVG	26084
APARTMENTS_MODE	23887
BASEMENTAREA_MODE	27641
YEARS_BEGINEXPLUATATION_MODE	22856
YEARS_BUILD_MODE	31818
COMMONAREA_MODE	33495
ELEVATORS_MODE	25189
ENTRANCES_MODE	23579
FLOORSMAX_MODE	23321
FLOORSMIN_MODE	32466
LANDAREA_MODE	28254
LIVINGAPARTMENTS_MODE	32780
LIVINGAREA_MODE	23552
NONLIVINGAPARTMENTS_MODE	33347
NONLIVINGAREA_MODE	26084
APARTMENTS_MEDI	23887
BASEMENTAREA_MEDI	27641
YEARS_BEGINEXPLUATATION_MEDI	22856
YEARS_BUILD_MEDI	31818
COMMONAREA_MEDI	33495
ELEVATORS_MEDI	25189
ENTRANCES_MEDI	23579
FLOORSMAX_MEDI	23321
FLOORSMIN_MEDI	32466
LANDAREA_MEDI	28254
LIVINGAPARTMENTS_MEDI	32780
LIVINGAREA_MEDI	23552
NONLIVINGAPARTMENTS_MEDI	33347
NONLIVINGAREA_MEDI	26084
FONDKAPREMONT_MODE	32797
HOUSETYPE_MODE	23619
TOTALAREA_MODE	22624
WALLSMATERIAL_MODE	23893
EMERGENCYSTATE_MODE	22209

```
OBS_30_CNT_SOCIAL_CIRCLE           29
DEF_30_CNT_SOCIAL_CIRCLE           29
OBS_60_CNT_SOCIAL_CIRCLE           29
DEF_60_CNT_SOCIAL_CIRCLE           29
DAYS_LAST_PHONE_CHANGE              0
FLAG_DOCUMENT_2                     0
FLAG_DOCUMENT_3                     0
FLAG_DOCUMENT_4                     0
FLAG_DOCUMENT_5                     0
FLAG_DOCUMENT_6                     0
FLAG_DOCUMENT_7                     0
FLAG_DOCUMENT_8                     0
FLAG_DOCUMENT_9                     0
FLAG_DOCUMENT_10                    0
FLAG_DOCUMENT_11                    0
FLAG_DOCUMENT_12                    0
FLAG_DOCUMENT_13                    0
FLAG_DOCUMENT_14                    0
FLAG_DOCUMENT_15                    0
FLAG_DOCUMENT_16                    0
FLAG_DOCUMENT_17                    0
FLAG_DOCUMENT_18                    0
FLAG_DOCUMENT_19                    0
FLAG_DOCUMENT_20                    0
FLAG_DOCUMENT_21                    0
AMT_REQ_CREDIT_BUREAU_HOUR       6049
AMT_REQ_CREDIT_BUREAU_DAY        6049
AMT_REQ_CREDIT_BUREAU_WEEK       6049
AMT_REQ_CREDIT_BUREAU_MON        6049
AMT_REQ_CREDIT_BUREAU_QRT        6049
AMT_REQ_CREDIT_BUREAU_YEAR       6049
dtype: int64
```

```
In [36]: train_df.drop(['OCCUPATION_TYPE'],axis=1,inplace=True)
```

```
In [37]: test_df.drop(['OCCUPATION_TYPE'],axis=1,inplace=True)
```

0.1.8 5.3 Creating

Days_employed variable divided by Days_birts variable is calculated days_employed_perc in train and test dataset

```
In [38]: train_df['DAYS_EMPLOYED_PERC'] = train_df['DAYS_EMPLOYED'] / train_df['DAYS_BIRTH']
         train_df['DAYS_EMPLOYED_PERC']
```

```
Out[38]: 0           0.067329
         1           0.070862
         2           0.011814
         3           0.159905
```

```
4          0.152418
             ...
307506     0.025303
307507   -17.580890
307508     0.529266
307509     0.400134
307510     0.074869
Name: DAYS_EMPLOYED_PERC, Length: 307508, dtype: float64
```

In [39]:
```python
train_df['AGE_CAL']=-train_df['DAYS_BIRTH']/365
train_df['AGE_CAL']

test_df['AGE_CAL']=-test_df['DAYS_BIRTH']/365
test_df['AGE_CAL']
```

Out[39]:
```
0        52.715068
1        49.490411
2        54.898630
3        38.290411
4        35.726027
           ...
48739    54.712329
48740    30.646575
48741    43.621918
48742    38.268493
48743    38.252055
Name: AGE_CAL, Length: 48744, dtype: float64
```

In [40]:
```python
train_df
```

Out[40]:
```
        SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR FLAG_OWN_REALT
0           100002       1         Cash loans           M            N
1           100003       0         Cash loans           F            N
2           100004       0    Revolving loans           M            Y
3           100006       0         Cash loans           F            N
4           100007       0         Cash loans           M            N
...            ...     ...                ...         ...          ...
307506      456251       0         Cash loans           M            N
307507      456252       0         Cash loans           F            N
307508      456253       0         Cash loans           F            N
307509      456254       1         Cash loans           F            N
307510      456255       0         Cash loans           F            N

        LIVINGAREA_AVG  NONLIVINGAPARTMENTS_AVG  NONLIVINGAREA_AVG  APARTMENTS_MODE
0               0.0190                   0.0000             0.0000           0.0252
1               0.0549                   0.0039             0.0098           0.0924
2                  NaN                      NaN                NaN              NaN
3                  NaN                      NaN                NaN              NaN
4                  NaN                      NaN                NaN              NaN
```

```
...                ...                         ...                   ...                ...
307506          0.1965                      0.0753                0.1095             0.1008
307507          0.0257                      0.0000                0.0000             0.0252
307508          0.9279                      0.0000                0.0000             0.1050
307509          0.0061                         NaN                   NaN             0.0126
307510          0.0791                         NaN                0.0000             0.0756

        FLAG_DOCUMENT_12  FLAG_DOCUMENT_13  FLAG_DOCUMENT_14  FLAG_DOCUMENT_15  FLAG_D
0                      0                 0                 0                 0
1                      0                 0                 0                 0
2                      0                 0                 0                 0
3                      0                 0                 0                 0
4                      0                 0                 0                 0
...                  ...               ...               ...               ...
307506                 0                 0                 0                 0
307507                 0                 0                 0                 0
307508                 0                 0                 0                 0
307509                 0                 0                 0                 0
307510                 0                 0                 0                 0

[307508 rows x 123 columns]
```

In [41]:
```
test_df['DAYS_EMPLOYED_PERC'] = test_df['DAYS_EMPLOYED'] / test_df['DAYS_BIRTH']
test_df['AGE_CAL']=-test_df['DAYS_BIRTH']/365
```

In [42]:
```
f,ax=plt.subplots(1,2,figsize=(18,8))
sns.violinplot("CODE_GENDER", "AGE_CAL", hue="TARGET", data=train_df,split=True,ax=ax
ax[0].set_title('CODE_GENDER and AGE_CAL vs TARGET')
ax[0].set_yticks(range(0,110,10))

sns.violinplot("NAME_CONTRACT_TYPE","AGE_CAL", hue="TARGET", data=train_df,split=True
ax[1].set_title('NAME_CONTRACT_TYPE and AGE_CAL vs TARGET')
ax[1].set_yticks(range(0,110,10))
plt.show()
```

Age grouping have been appeared need in this graphs. We think age group have been in below side * 18-30 * 30-45 * +45

In [43]:
```
#https://stackoverflow.com/questions/21702342/creating-a-new-column-based-on-if-elif-

def f(row):
    if row['AGE_CAL'] < 30:
        AGE_BIN = 1
    elif row['AGE_CAL'] < 45:
        AGE_BIN = 2
    else:
        AGE_BIN = 3
    return AGE_BIN
train_df['AGE_BIN'] = train_df.apply(f, axis=1)
```

In [44]:
```
f,ax=plt.subplots(1,2,figsize=(18,8))
sns.violinplot("AGE_BIN", "DAYS_EMPLOYED", hue="TARGET", data=train_df,split=True,ax=a
ax[0].set_title('DAYS_EMPLOYED and AGE_BIN vs TARGET')
ax[0].set_yticks(range(0,110,10))
```

Out[44]:
```
[<matplotlib.axis.YTick at 0xb678588>,
 <matplotlib.axis.YTick at 0xb662eb8>,
 <matplotlib.axis.YTick at 0xb6322b0>,
 <matplotlib.axis.YTick at 0x165a909b0>,
 <matplotlib.axis.YTick at 0x165a90e80>,
 <matplotlib.axis.YTick at 0x165a90908>,
 <matplotlib.axis.YTick at 0x165a89550>,
 <matplotlib.axis.YTick at 0x165a89a20>,
 <matplotlib.axis.YTick at 0x165a89ef0>,
 <matplotlib.axis.YTick at 0x165a82400>,
 <matplotlib.axis.YTick at 0x165a828d0>]
```

```
In [45]: def density_plot (df,varaible):
             plt.figure(figsize = (10, 8))

             # KDE plot of loans that were repaid on time
             sns.kdeplot(df.loc[df['TARGET'] == 0, varaible], label = 'target == 0')

             # KDE plot of loans which were not repaid on time
             sns.kdeplot(df.loc[df['TARGET'] == 1, varaible], label = 'target == 1')

             # Labeling of plot
             plt.xlabel(varaible); plt.ylabel('Density'); plt.title(varaible);

In [46]: density_plot(train_df,'AMT_CREDIT')
```

In [47]:
```
#https://stackoverflow.com/questions/21702342/creating-a-new-column-based-on-if-elif-

def f(row):
    if row['AMT_CREDIT'] < 5000000:
        AMT_CREDIT_BIN = 1
    elif row['AMT_CREDIT'] < 10000000:
        AMT_CREDIT_BIN = 2
    else:
        AMT_CREDIT_BIN = 3
    return AMT_CREDIT_BIN
train_df['AMT_CREDIT_BIN'] = train_df.apply(f, axis=1)
```

In [48]:
```
density_plot(train_df,'AMT_GOODS_PRICE')
```

```
C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
```

```
In [49]: #https://stackoverflow.com/questions/21702342/creating-a-new-column-based-on-if-elif-

         def f(row):
             if row['AMT_GOODS_PRICE'] < 5000000:
                 AMT_GOODS_PRICE_BIN = 1
             elif row['AMT_GOODS_PRICE'] < 10000000:
                 AMT_GOODS_PRICE_BIN = 2
             else:
                 AMT_GOODS_PRICE_BIN = 3
             return AMT_GOODS_PRICE_BIN
         train_df['AMT_GOODS_PRICE_BIN'] = train_df.apply(f, axis=1)

In [50]:  density_plot(train_df,'OWN_CAR_AGE')

C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
```

In [51]:
```
#https://stackoverflow.com/questions/21702342/creating-a-new-column-based-on-if-elif-

def f(row):
    if row['OWN_CAR_AGE'] == 'NaN':
        OWN_CAR_AGE_BIN = -9
    elif row['OWN_CAR_AGE'] < 10:
        OWN_CAR_AGE_BIN = 1
    elif row['OWN_CAR_AGE'] < 20:
        OWN_CAR_AGE_BIN = 2
    else:
        OWN_CAR_AGE_BIN = 3
    return OWN_CAR_AGE_BIN
train_df['OWN_CAR_AGE_BIN'] = train_df.apply(f, axis=1)
```

In [52]:
```
density_plot(train_df,'APARTMENTS_AVG')
```

```
C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
```

```
X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
```

In [53]:
```python
#https://stackoverflow.com/questions/21702342/creating-a-new-column-based-on-if-elif-

def f(row):
    if row['APARTMENTS_AVG'] == 'NaN':
        APARTMENTS_AVG_BIN = -9
    elif row['APARTMENTS_AVG'] < 0.15:
        APARTMENTS_AVG_BIN = 1
    elif row['APARTMENTS_AVG'] < 0.30:
        APARTMENTS_AVG_BIN = 2
    else:
        APARTMENTS_AVG_BIN = 3
    return APARTMENTS_AVG_BIN
train_df['APARTMENTS_AVG_BIN'] = train_df.apply(f, axis=1)
```

In [54]:
```python
density_plot(train_df,'DAYS_LAST_PHONE_CHANGE')
```

In [55]:
```
#https://stackoverflow.com/questions/21702342/creating-a-new-column-based-on-if-elif-

def f(row):
    if row['DAYS_LAST_PHONE_CHANGE'] == 'NaN':
        DAYS_LAST_PHONE_CHANGE_BIN = -9
    elif row['DAYS_LAST_PHONE_CHANGE'] < -3000:
        DAYS_LAST_PHONE_CHANGE_BIN = 1
    elif row['DAYS_LAST_PHONE_CHANGE'] < -1000:
        DAYS_LAST_PHONE_CHANGE_BIN = 2
    elif row['DAYS_LAST_PHONE_CHANGE'] == 0:
        DAYS_LAST_PHONE_CHANGE_BIN = 0
    else:
        DAYS_LAST_PHONE_CHANGE_BIN = 3
    return DAYS_LAST_PHONE_CHANGE_BIN
train_df['DAYS_LAST_PHONE_CHANGE_BIN'] = train_df.apply(f, axis=1)
```

In [56]:
```
# train_df
# data describe

train_df.describe()
```

```
Out[56]:          SK_ID_CURR         TARGET   CNT_CHILDREN  AMT_INCOME_TOTAL    AMT_CREDIT    A
         count  307508.000000  307508.00000  307508.000000      3.075080e+05  3.075080e+05  307
         mean   278180.610368       0.08073       0.416932      1.687984e+05  5.990296e+05   27
         std    102790.006413       0.27242       0.720568      2.371242e+05  4.024910e+05   14
         min    100002.000000       0.00000       0.000000      2.565000e+04  4.500000e+04    1
         25%    189145.750000       0.00000       0.000000      1.125000e+05  2.700000e+05   16
         50%    278201.500000       0.00000       0.000000      1.471500e+05  5.135310e+05   24
         75%    367142.250000       0.00000       1.000000      2.025000e+05  8.086500e+05   34
         max    456255.000000       1.00000      14.000000      1.170000e+08  4.050000e+06  258

                FLOORSMAX_MODE  FLOORSMIN_MODE  LANDAREA_MODE  LIVINGAPARTMENTS_MODE  LIVINGARE
         count   154489.000000    98868.000000  124919.000000           97311.000000     153159
         mean         0.222314        0.228059       0.064958               0.105645          0
         std          0.143710        0.161161       0.081751               0.097881          0
         min          0.000000        0.000000       0.000000               0.000000          0
         25%          0.166700        0.083300       0.016600               0.054200          0
         50%          0.166700        0.208300       0.045800               0.077100          0
         75%          0.333300        0.375000       0.084100               0.131300          0
         max          1.000000        1.000000       1.000000               1.000000          1

                AMT_REQ_CREDIT_BUREAU_MON  AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEA
         count              265990.000000              265990.000000              265990.00000
         mean                    0.267397                   0.265476                   1.89996
         std                     0.916006                   0.794058                   1.86929
         min                     0.000000                   0.000000                   0.00000
         25%                     0.000000                   0.000000                   0.00000
         50%                     0.000000                   0.000000                   1.00000
         75%                     0.000000                   0.000000                   3.00000
         max                    27.000000                 261.000000                  25.00000
```

```
In [57]: density_plot(train_df,'DAYS_EMPLOYED')
```

In [58]:
```
#https://stackoverflow.com/questions/21702342/creating-a-new-column-based-on-if-elif-

def f(row):
    if row['DAYS_EMPLOYED'] == 'NaN':
        DAYS_EMPLOYED_BIN = -9
    elif row['DAYS_EMPLOYED'] < 0:
        DAYS_EMPLOYED_BIN = 1
    elif row['DAYS_EMPLOYED'] < 2000:
        DAYS_EMPLOYED_BIN = 2
    else:
        DAYS_EMPLOYED_BIN = 3
    return DAYS_EMPLOYED_BIN
train_df['DAYS_EMPLOYED_BIN'] = train_df.apply(f, axis=1)
```

In [59]:
```
density_plot(train_df,'OBS_30_CNT_SOCIAL_CIRCLE')
```

```
C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
```

```
In [60]: density_plot(train_df,'DEF_30_CNT_SOCIAL_CIRCLE')

C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
C:\Users\UTKU\Anaconda3\lib\site-packages\statsmodels\nonparametric\kde.py:448: RuntimeWarning
  X = X[np.logical_and(X > clip[0], X < clip[1])] # won't work for two columns.
```

0.1.9 5.4 Converting

The data set will contain categorical variables. We will convert from categorical variables to numeric variables

Converting methodologies have been existed about this link

References: https://pbpython.com/categorical-encoding.html

```
In [61]: categorical_varaible=train_df.describe(include=['O'])
         categorical_varaible_col = set(categorical_varaible.columns)
```

```
In [62]: train_df=pd.get_dummies(train_df, columns=categorical_varaible_col)
         train_df.head(5)
```

Out[62]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_G
0	100002	1	0	202500.0	406597.5	24700.5	
1	100003	0	0	270000.0	1293502.5	35698.5	
2	100004	0	0	67500.0	135000.0	6750.0	
3	100006	0	0	135000.0	312682.5	29686.5	
4	100007	0	0	121500.0	513000.0	21865.5	

	LANDAREA_MODE	LIVINGAPARTMENTS_MODE	LIVINGAREA_MODE	NONLIVINGAPARTMENTS_MODE	N
0	0.0377	0.022	0.0198	0.0	
1	0.0128	0.079	0.0554	0.0	
2	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	

	AMT_REQ_CREDIT_BUREAU_QRT	AMT_REQ_CREDIT_BUREAU_YEAR	DAYS_EMPLOYED_PERC	AGE_CA
0	0.0	1.0	0.067329	25.9205
1	0.0	0.0	0.070862	45.9315
2	0.0	0.0	0.011814	52.1808
3	NaN	NaN	0.159905	52.0684
4	0.0	0.0	0.152418	54.6082

	WEEKDAY_APPR_PROCESS_START_THURSDAY	WEEKDAY_APPR_PROCESS_START_TUESDAY	WEEKDAY_AI
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	1	0	

	NAME_HOUSING_TYPE_Rented apartment	NAME_HOUSING_TYPE_With parents	NAME_CONTRACT_T
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	0	

	ORGANIZATION_TYPE_Industry: type 6	ORGANIZATION_TYPE_Industry: type 7	ORGANIZATIO
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	0	

	ORGANIZATION_TYPE_Transport: type 3	ORGANIZATION_TYPE_Transport: type 4	ORGANIZAT
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	0	

In [63]:
```
test_df=pd.get_dummies(test_df, columns=categorical_varaible_col)
test_df.head(5)
```

Out[63]:

	SK_ID_CURR	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRI
0	100001	0	135000.0	568800.0	20560.5	450000

```
1      100005            0         99000.0    222768.0     17370.0          180000
2      100013            0        202500.0    663264.0     69777.0          630000
3      100028            2        315000.0   1575000.0     49018.5         1575000
4      100038            1        180000.0    625500.0     32067.0          625500

   LANDAREA_MODE  LIVINGAPARTMENTS_MODE  LIVINGAREA_MODE  NONLIVINGAPARTMENTS_MODE  NO
0            NaN                    NaN           0.0526                       NaN
1            NaN                    NaN              NaN                       NaN
2            NaN                    NaN              NaN                       NaN
3         0.2089                 0.2626           0.3827                    0.0389
4            NaN                    NaN              NaN                       NaN

   AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR    AGE_CAL  DAYS_EMPLOYED_PE
0                        0.0                         0.0  52.715068             0.1210
1                        0.0                         3.0  49.490411             0.2473
2                        1.0                         4.0  54.898630             0.2224
3                        0.0                         3.0  38.290411             0.1335
4                        NaN                         NaN  35.726027             0.1680

   FONDKAPREMONT_MODE_org spec account  FONDKAPREMONT_MODE_reg oper account  FONDKAPRE
0                                    0                                    0
1                                    0                                    0
2                                    0                                    0
3                                    0                                    1
4                                    0                                    0

   FLAG_OWN_CAR_Y  ORGANIZATION_TYPE_Advertising  ORGANIZATION_TYPE_Agriculture  ORGA
0               0                              0                              0
1               0                              0                              0
2               1                              0                              0
3               0                              0                              0
4               1                              0                              0

   ORGANIZATION_TYPE_Legal Services  ORGANIZATION_TYPE_Medicine  ORGANIZATION_TYPE_Mi
0                                 0                           0
1                                 0                           0
2                                 0                           0
3                                 0                           0
4                                 0                           0
```

In [64]:
```python
train_df['TARGET'].value_counts()
```

Out[64]:
```
0    282683
1     24825
Name: TARGET, dtype: int64
```

In [65]:
```python
# train_df
#data info
```

```
train_df.info(max_cols=1000)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 307508 entries, 0 to 307510
Data columns (total 237 columns):
SK_ID_CURR                                          307508 non-null int64
TARGET                                              307508 non-null int64
CNT_CHILDREN                                        307508 non-null int64
AMT_INCOME_TOTAL                                    307508 non-null float64
AMT_CREDIT                                          307508 non-null float64
AMT_ANNUITY                                         307496 non-null float64
AMT_GOODS_PRICE                                     307230 non-null float64
REGION_POPULATION_RELATIVE                          307508 non-null float64
DAYS_BIRTH                                          307508 non-null int64
DAYS_EMPLOYED                                       307508 non-null int64
DAYS_REGISTRATION                                   307508 non-null float64
DAYS_ID_PUBLISH                                     307508 non-null int64
OWN_CAR_AGE                                         104581 non-null float64
FLAG_MOBIL                                          307508 non-null int64
FLAG_EMP_PHONE                                      307508 non-null int64
FLAG_WORK_PHONE                                     307508 non-null int64
FLAG_CONT_MOBILE                                    307508 non-null int64
FLAG_PHONE                                          307508 non-null int64
FLAG_EMAIL                                          307508 non-null int64
CNT_FAM_MEMBERS                                     307506 non-null float64
REGION_RATING_CLIENT                                307508 non-null int64
REGION_RATING_CLIENT_W_CITY                         307508 non-null int64
HOUR_APPR_PROCESS_START                             307508 non-null int64
REG_REGION_NOT_LIVE_REGION                          307508 non-null int64
REG_REGION_NOT_WORK_REGION                          307508 non-null int64
LIVE_REGION_NOT_WORK_REGION                         307508 non-null int64
REG_CITY_NOT_LIVE_CITY                              307508 non-null int64
REG_CITY_NOT_WORK_CITY                              307508 non-null int64
LIVE_CITY_NOT_WORK_CITY                             307508 non-null int64
EXT_SOURCE_1                                        134132 non-null float64
EXT_SOURCE_2                                        306849 non-null float64
EXT_SOURCE_3                                        246545 non-null float64
APARTMENTS_AVG                                      151448 non-null float64
BASEMENTAREA_AVG                                    127566 non-null float64
YEARS_BEGINEXPLUATATION_AVG                         157502 non-null float64
YEARS_BUILD_AVG                                     103022 non-null float64
COMMONAREA_AVG                                      92645 non-null float64
ELEVATORS_AVG                                       143618 non-null float64
ENTRANCES_AVG                                       152681 non-null float64
FLOORSMAX_AVG                                       154489 non-null float64
FLOORSMIN_AVG                                       98868 non-null float64
LANDAREA_AVG                                        124919 non-null float64
```

```
LIVINGAPARTMENTS_AVG            97311 non-null float64
LIVINGAREA_AVG                  153159 non-null float64
NONLIVINGAPARTMENTS_AVG         93996 non-null float64
NONLIVINGAREA_AVG               137828 non-null float64
APARTMENTS_MODE                 151448 non-null float64
BASEMENTAREA_MODE               127566 non-null float64
YEARS_BEGINEXPLUATATION_MODE    157502 non-null float64
YEARS_BUILD_MODE                103022 non-null float64
COMMONAREA_MODE                 92645 non-null float64
ELEVATORS_MODE                  143618 non-null float64
ENTRANCES_MODE                  152681 non-null float64
FLOORSMAX_MODE                  154489 non-null float64
FLOORSMIN_MODE                  98868 non-null float64
LANDAREA_MODE                   124919 non-null float64
LIVINGAPARTMENTS_MODE           97311 non-null float64
LIVINGAREA_MODE                 153159 non-null float64
NONLIVINGAPARTMENTS_MODE        93996 non-null float64
NONLIVINGAREA_MODE              137828 non-null float64
APARTMENTS_MEDI                 151448 non-null float64
BASEMENTAREA_MEDI               127566 non-null float64
YEARS_BEGINEXPLUATATION_MEDI    157502 non-null float64
YEARS_BUILD_MEDI                103022 non-null float64
COMMONAREA_MEDI                 92645 non-null float64
ELEVATORS_MEDI                  143618 non-null float64
ENTRANCES_MEDI                  152681 non-null float64
FLOORSMAX_MEDI                  154489 non-null float64
FLOORSMIN_MEDI                  98868 non-null float64
LANDAREA_MEDI                   124919 non-null float64
LIVINGAPARTMENTS_MEDI           97311 non-null float64
LIVINGAREA_MEDI                 153159 non-null float64
NONLIVINGAPARTMENTS_MEDI        93996 non-null float64
NONLIVINGAREA_MEDI              137828 non-null float64
TOTALAREA_MODE                  159078 non-null float64
OBS_30_CNT_SOCIAL_CIRCLE        306487 non-null float64
DEF_30_CNT_SOCIAL_CIRCLE        306487 non-null float64
OBS_60_CNT_SOCIAL_CIRCLE        306487 non-null float64
DEF_60_CNT_SOCIAL_CIRCLE        306487 non-null float64
DAYS_LAST_PHONE_CHANGE          307508 non-null float64
FLAG_DOCUMENT_2                 307508 non-null int64
FLAG_DOCUMENT_3                 307508 non-null int64
FLAG_DOCUMENT_4                 307508 non-null int64
FLAG_DOCUMENT_5                 307508 non-null int64
FLAG_DOCUMENT_6                 307508 non-null int64
FLAG_DOCUMENT_7                 307508 non-null int64
FLAG_DOCUMENT_8                 307508 non-null int64
FLAG_DOCUMENT_9                 307508 non-null int64
FLAG_DOCUMENT_10                307508 non-null int64
FLAG_DOCUMENT_11                307508 non-null int64
```

```
FLAG_DOCUMENT_12                                    307508 non-null int64
FLAG_DOCUMENT_13                                    307508 non-null int64
FLAG_DOCUMENT_14                                    307508 non-null int64
FLAG_DOCUMENT_15                                    307508 non-null int64
FLAG_DOCUMENT_16                                    307508 non-null int64
FLAG_DOCUMENT_17                                    307508 non-null int64
FLAG_DOCUMENT_18                                    307508 non-null int64
FLAG_DOCUMENT_19                                    307508 non-null int64
FLAG_DOCUMENT_20                                    307508 non-null int64
FLAG_DOCUMENT_21                                    307508 non-null int64
AMT_REQ_CREDIT_BUREAU_HOUR                          265990 non-null float64
AMT_REQ_CREDIT_BUREAU_DAY                           265990 non-null float64
AMT_REQ_CREDIT_BUREAU_WEEK                          265990 non-null float64
AMT_REQ_CREDIT_BUREAU_MON                           265990 non-null float64
AMT_REQ_CREDIT_BUREAU_QRT                           265990 non-null float64
AMT_REQ_CREDIT_BUREAU_YEAR                          265990 non-null float64
DAYS_EMPLOYED_PERC                                  307508 non-null float64
AGE_CAL                                             307508 non-null float64
AGE_BIN                                             307508 non-null int64
AMT_CREDIT_BIN                                      307508 non-null int64
AMT_GOODS_PRICE_BIN                                 307508 non-null int64
OWN_CAR_AGE_BIN                                     307508 non-null int64
APARTMENTS_AVG_BIN                                  307508 non-null int64
DAYS_LAST_PHONE_CHANGE_BIN                          307508 non-null int64
DAYS_EMPLOYED_BIN                                   307508 non-null int64
WALLSMATERIAL_MODE_Block                            307508 non-null uint8
WALLSMATERIAL_MODE_Mixed                            307508 non-null uint8
WALLSMATERIAL_MODE_Monolithic                       307508 non-null uint8
WALLSMATERIAL_MODE_Others                           307508 non-null uint8
WALLSMATERIAL_MODE_Panel                            307508 non-null uint8
WALLSMATERIAL_MODE_Stone, brick                     307508 non-null uint8
WALLSMATERIAL_MODE_Wooden                           307508 non-null uint8
FLAG_OWN_REALTY_N                                   307508 non-null uint8
FLAG_OWN_REALTY_Y                                   307508 non-null uint8
NAME_EDUCATION_TYPE_Academic degree                 307508 non-null uint8
NAME_EDUCATION_TYPE_Higher education                307508 non-null uint8
NAME_EDUCATION_TYPE_Incomplete higher               307508 non-null uint8
NAME_EDUCATION_TYPE_Lower secondary                 307508 non-null uint8
NAME_EDUCATION_TYPE_Secondary / secondary special   307508 non-null uint8
NAME_FAMILY_STATUS_Civil marriage                   307508 non-null uint8
NAME_FAMILY_STATUS_Married                          307508 non-null uint8
NAME_FAMILY_STATUS_Separated                        307508 non-null uint8
NAME_FAMILY_STATUS_Single / not married             307508 non-null uint8
NAME_FAMILY_STATUS_Unknown                          307508 non-null uint8
NAME_FAMILY_STATUS_Widow                            307508 non-null uint8
WEEKDAY_APPR_PROCESS_START_FRIDAY                   307508 non-null uint8
WEEKDAY_APPR_PROCESS_START_MONDAY                   307508 non-null uint8
WEEKDAY_APPR_PROCESS_START_SATURDAY                 307508 non-null uint8
```

```
WEEKDAY_APPR_PROCESS_START_SUNDAY                    307508 non-null uint8
WEEKDAY_APPR_PROCESS_START_THURSDAY                  307508 non-null uint8
WEEKDAY_APPR_PROCESS_START_TUESDAY                   307508 non-null uint8
WEEKDAY_APPR_PROCESS_START_WEDNESDAY                 307508 non-null uint8
FONDKAPREMONT_MODE_not specified                     307508 non-null uint8
FONDKAPREMONT_MODE_org spec account                  307508 non-null uint8
FONDKAPREMONT_MODE_reg oper account                  307508 non-null uint8
FONDKAPREMONT_MODE_reg oper spec account             307508 non-null uint8
NAME_INCOME_TYPE_Businessman                         307508 non-null uint8
NAME_INCOME_TYPE_Commercial associate                307508 non-null uint8
NAME_INCOME_TYPE_Maternity leave                     307508 non-null uint8
NAME_INCOME_TYPE_Pensioner                           307508 non-null uint8
NAME_INCOME_TYPE_State servant                       307508 non-null uint8
NAME_INCOME_TYPE_Student                             307508 non-null uint8
NAME_INCOME_TYPE_Unemployed                          307508 non-null uint8
NAME_INCOME_TYPE_Working                             307508 non-null uint8
HOUSETYPE_MODE_block of flats                        307508 non-null uint8
HOUSETYPE_MODE_specific housing                      307508 non-null uint8
HOUSETYPE_MODE_terraced house                        307508 non-null uint8
EMERGENCYSTATE_MODE_No                               307508 non-null uint8
EMERGENCYSTATE_MODE_Yes                              307508 non-null uint8
NAME_TYPE_SUITE_Children                             307508 non-null uint8
NAME_TYPE_SUITE_Family                               307508 non-null uint8
NAME_TYPE_SUITE_Group of people                      307508 non-null uint8
NAME_TYPE_SUITE_Other_A                              307508 non-null uint8
NAME_TYPE_SUITE_Other_B                              307508 non-null uint8
NAME_TYPE_SUITE_Spouse, partner                      307508 non-null uint8
NAME_TYPE_SUITE_Unaccompanied                        307508 non-null uint8
NAME_HOUSING_TYPE_Co-op apartment                    307508 non-null uint8
NAME_HOUSING_TYPE_House / apartment                  307508 non-null uint8
NAME_HOUSING_TYPE_Municipal apartment                307508 non-null uint8
NAME_HOUSING_TYPE_Office apartment                   307508 non-null uint8
NAME_HOUSING_TYPE_Rented apartment                   307508 non-null uint8
NAME_HOUSING_TYPE_With parents                       307508 non-null uint8
NAME_CONTRACT_TYPE_Cash loans                        307508 non-null uint8
NAME_CONTRACT_TYPE_Revolving loans                   307508 non-null uint8
CODE_GENDER_F                                        307508 non-null uint8
CODE_GENDER_M                                        307508 non-null uint8
CODE_GENDER_XNA                                      307508 non-null uint8
FLAG_OWN_CAR_N                                       307508 non-null uint8
FLAG_OWN_CAR_Y                                       307508 non-null uint8
ORGANIZATION_TYPE_Advertising                        307508 non-null uint8
ORGANIZATION_TYPE_Agriculture                        307508 non-null uint8
ORGANIZATION_TYPE_Bank                               307508 non-null uint8
ORGANIZATION_TYPE_Business Entity Type 1             307508 non-null uint8
ORGANIZATION_TYPE_Business Entity Type 2             307508 non-null uint8
ORGANIZATION_TYPE_Business Entity Type 3             307508 non-null uint8
ORGANIZATION_TYPE_Cleaning                           307508 non-null uint8
```

```
ORGANIZATION_TYPE_Construction                     307508 non-null uint8
ORGANIZATION_TYPE_Culture                          307508 non-null uint8
ORGANIZATION_TYPE_Electricity                      307508 non-null uint8
ORGANIZATION_TYPE_Emergency                        307508 non-null uint8
ORGANIZATION_TYPE_Government                       307508 non-null uint8
ORGANIZATION_TYPE_Hotel                            307508 non-null uint8
ORGANIZATION_TYPE_Housing                          307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 1                 307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 10                307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 11                307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 12                307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 13                307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 2                 307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 3                 307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 4                 307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 5                 307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 6                 307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 7                 307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 8                 307508 non-null uint8
ORGANIZATION_TYPE_Industry: type 9                 307508 non-null uint8
ORGANIZATION_TYPE_Insurance                        307508 non-null uint8
ORGANIZATION_TYPE_Kindergarten                     307508 non-null uint8
ORGANIZATION_TYPE_Legal Services                   307508 non-null uint8
ORGANIZATION_TYPE_Medicine                         307508 non-null uint8
ORGANIZATION_TYPE_Military                         307508 non-null uint8
ORGANIZATION_TYPE_Mobile                           307508 non-null uint8
ORGANIZATION_TYPE_Other                            307508 non-null uint8
ORGANIZATION_TYPE_Police                           307508 non-null uint8
ORGANIZATION_TYPE_Postal                           307508 non-null uint8
ORGANIZATION_TYPE_Realtor                          307508 non-null uint8
ORGANIZATION_TYPE_Religion                         307508 non-null uint8
ORGANIZATION_TYPE_Restaurant                       307508 non-null uint8
ORGANIZATION_TYPE_School                           307508 non-null uint8
ORGANIZATION_TYPE_Security                         307508 non-null uint8
ORGANIZATION_TYPE_Security Ministries              307508 non-null uint8
ORGANIZATION_TYPE_Self-employed                    307508 non-null uint8
ORGANIZATION_TYPE_Services                         307508 non-null uint8
ORGANIZATION_TYPE_Telecom                          307508 non-null uint8
ORGANIZATION_TYPE_Trade: type 1                    307508 non-null uint8
ORGANIZATION_TYPE_Trade: type 2                    307508 non-null uint8
ORGANIZATION_TYPE_Trade: type 3                    307508 non-null uint8
ORGANIZATION_TYPE_Trade: type 4                    307508 non-null uint8
ORGANIZATION_TYPE_Trade: type 5                    307508 non-null uint8
ORGANIZATION_TYPE_Trade: type 6                    307508 non-null uint8
ORGANIZATION_TYPE_Trade: type 7                    307508 non-null uint8
ORGANIZATION_TYPE_Transport: type 1                307508 non-null uint8
ORGANIZATION_TYPE_Transport: type 2                307508 non-null uint8
ORGANIZATION_TYPE_Transport: type 3                307508 non-null uint8
```

```
ORGANIZATION_TYPE_Transport: type 4                    307508 non-null uint8
ORGANIZATION_TYPE_University                           307508 non-null uint8
ORGANIZATION_TYPE_XNA                                  307508 non-null uint8
dtypes: float64(67), int64(48), uint8(122)
memory usage: 317.9 MB
```

0.2 Chapter 6 - Step 4: Perform Exploratory Analysis with Statistics

0.2.1 6.1 Correlation Elimination

All variable analyze the correlation of target. We will choose higher than 0.05 or lower than -0.005. Correlations are very useful in many applications, especially when conducting regression analysis. However, it should not be mixed with causality and misinterpreted in any way. I should also always check the correlation between different variables in our dataset and gather some insights as part of my exploration and analysis.

```
In [66]: #correlation heatmap of dataset
         def correlation_heatmap(df):
             _ , ax = plt.subplots(figsize =(14, 12))
             colormap = sns.diverging_palette(220, 10, as_cmap = True)

             _ = sns.heatmap(
                 df.corr(),
                 cmap = colormap,
                 square=True,
                 cbar_kws={'shrink':.9 },
                 ax=ax,
                 annot=True,
                 linewidths=0.1,vmax=1.0, linecolor='white',
                 annot_kws={'fontsize':12 }
             )

             plt.title('Pearson Correlation of Features', y=1.05, size=15)
```

```
In [67]: v1={'TARGET',
         'CNT_CHILDREN',
         'AMT_INCOME_TOTAL',
         'AMT_CREDIT',
         'AMT_ANNUITY',
         'AMT_GOODS_PRICE',
         'REGION_POPULATION_RELATIVE',
         'DAYS_BIRTH',
         'DAYS_EMPLOYED',
         'DAYS_REGISTRATION',
         'DAYS_ID_PUBLISH',
         'OWN_CAR_AGE',
         'FLAG_MOBIL',
         'FLAG_EMP_PHONE',
```

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

v2={'TARGET',
'REGION_RATING_CLIENT_W_CITY',
'HOUR_APPR_PROCESS_START',
'REG_REGION_NOT_LIVE_REGION',
'REG_REGION_NOT_WORK_REGION',
'LIVE_REGION_NOT_WORK_REGION',
'REG_CITY_NOT_LIVE_CITY',
'REG_CITY_NOT_WORK_CITY',
'LIVE_CITY_NOT_WORK_CITY',
'EXT_SOURCE_1',
'EXT_SOURCE_2',
'EXT_SOURCE_3',
'APARTMENTS_AVG',
'BASEMENTAREA_AVG',
'YEARS_BEGINEXPLUATATION_AVG',
'YEARS_BUILD_AVG',
'COMMONAREA_AVG',
'ELEVATORS_AVG',
'ENTRANCES_AVG',
'FLOORSMAX_AVG',
'FLOORSMIN_AVG'}

v3={'TARGET',
'LANDAREA_AVG',
'LIVINGAPARTMENTS_AVG',
'LIVINGAREA_AVG',
'NONLIVINGAPARTMENTS_AVG',
'NONLIVINGAREA_AVG',
'APARTMENTS_MODE',
'BASEMENTAREA_MODE',
'YEARS_BEGINEXPLUATATION_MODE',
'YEARS_BUILD_MODE',
'COMMONAREA_MODE',
'ELEVATORS_MODE',
'ENTRANCES_MODE',
'FLOORSMAX_MODE',
'FLOORSMIN_MODE',
'LANDAREA_MODE',
'LIVINGAPARTMENTS_MODE',
'LIVINGAREA_MODE',
```

```
'NONLIVINGAPARTMENTS_MODE',
'NONLIVINGAREA_MODE',
'APARTMENTS_MEDI'
}

v4={'TARGET',
'BASEMENTAREA_MEDI',
'YEARS_BEGINEXPLUATATION_MEDI',
'YEARS_BUILD_MEDI',
'COMMONAREA_MEDI',
'ELEVATORS_MEDI',
'ENTRANCES_MEDI',
'FLOORSMAX_MEDI',
'FLOORSMIN_MEDI',
'LANDAREA_MEDI',
'LIVINGAPARTMENTS_MEDI',
'LIVINGAREA_MEDI',
'NONLIVINGAPARTMENTS_MEDI',
'NONLIVINGAREA_MEDI',
'TOTALAREA_MODE',
'OBS_30_CNT_SOCIAL_CIRCLE',
'DEF_30_CNT_SOCIAL_CIRCLE',
'OBS_60_CNT_SOCIAL_CIRCLE',
'DEF_60_CNT_SOCIAL_CIRCLE',
'DAYS_LAST_PHONE_CHANGE',
'FLAG_DOCUMENT_2'}
v5={'TARGET',
'FLAG_DOCUMENT_3',
'FLAG_DOCUMENT_4',
'FLAG_DOCUMENT_5',
'FLAG_DOCUMENT_6',
'FLAG_DOCUMENT_7',
'FLAG_DOCUMENT_8',
'FLAG_DOCUMENT_9',
'FLAG_DOCUMENT_10',
'FLAG_DOCUMENT_11',
'FLAG_DOCUMENT_12',
'FLAG_DOCUMENT_13',
'FLAG_DOCUMENT_14',
'FLAG_DOCUMENT_15',
'FLAG_DOCUMENT_16',
'FLAG_DOCUMENT_17',
'FLAG_DOCUMENT_18',
'FLAG_DOCUMENT_19',
'FLAG_DOCUMENT_20',
'FLAG_DOCUMENT_21',
'AMT_REQ_CREDIT_BUREAU_HOUR'}
v6={
```

```
'TARGET',
'AMT_REQ_CREDIT_BUREAU_DAY',
'AMT_REQ_CREDIT_BUREAU_WEEK',
'AMT_REQ_CREDIT_BUREAU_MON',
'AMT_REQ_CREDIT_BUREAU_QRT',
'AMT_REQ_CREDIT_BUREAU_YEAR',
'DAYS_EMPLOYED_PERC',
'AGE_CAL',
'FONDKAPREMONT_MODE_not specified',
'FONDKAPREMONT_MODE_org spec account',
'FONDKAPREMONT_MODE_reg oper account',
'FONDKAPREMONT_MODE_reg oper spec account',
'WEEKDAY_APPR_PROCESS_START_FRIDAY',
'WEEKDAY_APPR_PROCESS_START_MONDAY',
'WEEKDAY_APPR_PROCESS_START_SATURDAY',
'WEEKDAY_APPR_PROCESS_START_SUNDAY',
'WEEKDAY_APPR_PROCESS_START_THURSDAY',
'WEEKDAY_APPR_PROCESS_START_TUESDAY',
'WEEKDAY_APPR_PROCESS_START_WEDNESDAY',
'CODE_GENDER_F',
'CODE_GENDER_M'}
v7={
    'TARGET',
'CODE_GENDER_XNA',
'ORGANIZATION_TYPE_Advertising',
'ORGANIZATION_TYPE_Agriculture',
'ORGANIZATION_TYPE_Bank',
'ORGANIZATION_TYPE_Business Entity Type 1',
'ORGANIZATION_TYPE_Business Entity Type 2',
'ORGANIZATION_TYPE_Business Entity Type 3',
'ORGANIZATION_TYPE_Cleaning',
'ORGANIZATION_TYPE_Construction',
'ORGANIZATION_TYPE_Culture',
'ORGANIZATION_TYPE_Electricity',
'ORGANIZATION_TYPE_Emergency',
'ORGANIZATION_TYPE_Government',
'ORGANIZATION_TYPE_Hotel',
'ORGANIZATION_TYPE_Housing',
'ORGANIZATION_TYPE_Industry: type 1',
'ORGANIZATION_TYPE_Industry: type 10',
'ORGANIZATION_TYPE_Industry: type 11',
'ORGANIZATION_TYPE_Industry: type 12',
'ORGANIZATION_TYPE_Industry: type 13'}

v8={
    'TARGET',
'ORGANIZATION_TYPE_Industry: type 2',
'ORGANIZATION_TYPE_Industry: type 3',
```

```
'ORGANIZATION_TYPE_Industry: type 4',
'ORGANIZATION_TYPE_Industry: type 5',
'ORGANIZATION_TYPE_Industry: type 6',
'ORGANIZATION_TYPE_Industry: type 7',
'ORGANIZATION_TYPE_Industry: type 8',
'ORGANIZATION_TYPE_Industry: type 9',
'ORGANIZATION_TYPE_Insurance',
'ORGANIZATION_TYPE_Kindergarten',
'ORGANIZATION_TYPE_Legal Services',
'ORGANIZATION_TYPE_Medicine',
'ORGANIZATION_TYPE_Military',
'ORGANIZATION_TYPE_Mobile',
'ORGANIZATION_TYPE_Other',
'ORGANIZATION_TYPE_Police',
'ORGANIZATION_TYPE_Postal',
'ORGANIZATION_TYPE_Realtor',
'ORGANIZATION_TYPE_Religion',
'ORGANIZATION_TYPE_Restaurant'}

v9={
    'TARGET',
'ORGANIZATION_TYPE_School',
'ORGANIZATION_TYPE_Security',
'ORGANIZATION_TYPE_Security Ministries',
'ORGANIZATION_TYPE_Self-employed',
'ORGANIZATION_TYPE_Services',
'ORGANIZATION_TYPE_Telecom',
'ORGANIZATION_TYPE_Trade: type 1',
'ORGANIZATION_TYPE_Trade: type 2',
'ORGANIZATION_TYPE_Trade: type 3',
'ORGANIZATION_TYPE_Trade: type 4',
'ORGANIZATION_TYPE_Trade: type 5',
'ORGANIZATION_TYPE_Trade: type 6',
'ORGANIZATION_TYPE_Trade: type 7',
'ORGANIZATION_TYPE_Transport: type 1',
'ORGANIZATION_TYPE_Transport: type 2',
'ORGANIZATION_TYPE_Transport: type 3',
'ORGANIZATION_TYPE_Transport: type 4',
'ORGANIZATION_TYPE_University',
'ORGANIZATION_TYPE_XNA',
'NAME_HOUSING_TYPE_Co-op apartment'}

v10={
    'TARGET',
'NAME_HOUSING_TYPE_House / apartment',
'NAME_HOUSING_TYPE_Municipal apartment',
'NAME_HOUSING_TYPE_Office apartment',
'NAME_HOUSING_TYPE_Rented apartment',
```

```
'NAME_HOUSING_TYPE_With parents',
'NAME_EDUCATION_TYPE_Academic degree',
'NAME_EDUCATION_TYPE_Higher education',
'NAME_EDUCATION_TYPE_Incomplete higher',
'NAME_EDUCATION_TYPE_Lower secondary',
'NAME_EDUCATION_TYPE_Secondary / secondary special',
'NAME_CONTRACT_TYPE_Cash loans',
'NAME_CONTRACT_TYPE_Revolving loans',
'EMERGENCYSTATE_MODE_No',
'EMERGENCYSTATE_MODE_Yes',
'NAME_TYPE_SUITE_Children',
'NAME_TYPE_SUITE_Family',
'NAME_TYPE_SUITE_Group of people',
'NAME_TYPE_SUITE_Other_A',
'NAME_TYPE_SUITE_Other_B',
'NAME_TYPE_SUITE_Spouse, partner'}

v11={
    'TARGET',
'NAME_TYPE_SUITE_Unaccompanied',
'FLAG_OWN_REALTY_N',
'FLAG_OWN_REALTY_Y',
'WALLSMATERIAL_MODE_Block',
'WALLSMATERIAL_MODE_Mixed',
'WALLSMATERIAL_MODE_Monolithic',
'WALLSMATERIAL_MODE_Others',
'WALLSMATERIAL_MODE_Panel',
'WALLSMATERIAL_MODE_Stone, brick',
'WALLSMATERIAL_MODE_Wooden',
'NAME_FAMILY_STATUS_Civil marriage'}

v12={
    'TARGET',
    'NAME_FAMILY_STATUS_Married',
'NAME_FAMILY_STATUS_Separated',
'NAME_FAMILY_STATUS_Single / not married',
'NAME_FAMILY_STATUS_Unknown',
'NAME_FAMILY_STATUS_Widow',
'NAME_INCOME_TYPE_Businessman',
'NAME_INCOME_TYPE_Commercial associate',
'NAME_INCOME_TYPE_Maternity leave',
'NAME_INCOME_TYPE_Pensioner',
'NAME_INCOME_TYPE_State servant',
'NAME_INCOME_TYPE_Student',
'NAME_INCOME_TYPE_Unemployed',
'NAME_INCOME_TYPE_Working',
'FLAG_OWN_CAR_N',
'FLAG_OWN_CAR_Y',
```

```
'HOUSETYPE_MODE_block of flats',
'HOUSETYPE_MODE_specific housing',
'HOUSETYPE_MODE_terraced house'}

v13={
    'TARGET',
'AGE_BIN',
'AMT_CREDIT_BIN',
'AMT_GOODS_PRICE_BIN',
'OWN_CAR_AGE_BIN',
'APARTMENTS_AVG_BIN',
'DAYS_LAST_PHONE_CHANGE_BIN',
'DAYS_EMPLOYED_BIN'
}
```

In [68]:
```
train_df_v1=train_df[v1]
train_df_v2=train_df[v2]
train_df_v3=train_df[v3]
train_df_v4=train_df[v4]
train_df_v5=train_df[v5]
train_df_v6=train_df[v6]
train_df_v7=train_df[v7]
train_df_v8=train_df[v8]
train_df_v9=train_df[v9]
train_df_v10=train_df[v10]
train_df_v11=train_df[v11]
train_df_v12=train_df[v12]
train_df_v13=train_df[v13]
```

In [69]:
```
correlation_heatmap(train_df_v13)
```

```
In [70]: correlation_heatmap(train_df_v1)
```

REGION_RATING_CLIENT AND DAYS_ID_PUBLISH is higher than 0.05 correlation. So 2 vairables are selected as final vairables

```
In [71]: correlation_heatmap(train_df_v2)
```

REGION_RATING_CLIENT_W_CITY, EXT_SOURCE_1 EXT_SOURCE_2,EXT_SOURCE_3 is higher than 0.05 correlation. So 4 vairables are selected as final vairables

```
In [72]: correlation_heatmap(train_df_v3)
```

```
In [73]: correlation_heatmap(train_df_v4)
```

DAYS_LAST_PHONE_CHANGE is higher than 0.05 correlation. So 1 vairable S selected as final vairables

```
In [74]: correlation_heatmap(train_df_v5)
```

```
In [75]: correlation_heatmap(train_df_v6)
```

AGE_CALC, CODE_GENDER_F,CODE_GENDER_M are higher than 0.05 correlation. So 3 vairables are selected as final vairables

```
In [76]: correlation_heatmap(train_df_v7)
```

```
In [77]: correlation_heatmap(train_df_v8)
```

```
In [78]: correlation_heatmap(train_df_v9)
```

In [79]: correlation_heatmap(train_df_v10)

NAME_EDUCATION_TYPE_Secondary / secondary special are higher than 0.05 correlation. So 1 vairable are selected as final vairable

```
In [80]: correlation_heatmap(train_df_v11)
```

```
In [81]: correlation_heatmap(train_df_v12)
```

NAME_INCOME_TYPE_Working is higher than 0.05 correlation. So 1 vairable are selected as final vairable

Finally NAME_INCOME_TYPE_Working, NAME_EDUCATION_TYPE_Secondary / secondary special, AGE_CAL, CODE_GENDER_F,CODE_GENDER_M ,DAYS_LAST_PHONE_CHANGE, REGION_RATING_CLIENT_W_CITY, EXT_SOURCE_1 EXT_SOURCE_2,EXT_SOURCE_3 are selected final variables

```
In [82]: final_list={'NAME_INCOME_TYPE_Working', 'NAME_EDUCATION_TYPE_Secondary / secondary spe
         train_df_final_list=train_df[final_list]
         correlation_heatmap(train_df_final_list)
```

```
In [83]: final_list_V1={'NAME_INCOME_TYPE_Working', 'NAME_EDUCATION_TYPE_Secondary / secondary
         train_df_final_list_V1=train_df[final_list_V1]
         correlation_heatmap(train_df_final_list_V1)
```

0.2.2 6.1.1 Elimination List

1. EXT_SOURCE_1 variable is elimanted because of high correlated age_cal. I can choose the age_cal
2. EXT_SOURCE_3 variable is elimanted because of high correlated age_cal. I can choose the age_cal
3. Age_bin variable is elimanted because of high correlated age_cal. I can choose the age_cal
4. Gender_M variable is elimanted because of high correlated Gender_F. I can choose the Gender_F
5. REGION_RATING_CLIENT_W_CITY is elimanted because of moderate correlated ext_source_2. I can choose the ext_source_2
6. NAME_INCOME_TYPE_Working is elimanted because of moderate correlated age_cal. I can choose the age_cal

```
In [84]: final_list_V2_with_target={ 'NAME_EDUCATION_TYPE_Secondary / secondary special', 'AGE_
```

```
final_list_V2={ 'NAME_EDUCATION_TYPE_Secondary / secondary special', 'AGE_CAL', 'CODE_
test_final_list_V2={ 'NAME_EDUCATION_TYPE_Secondary / secondary special', 'AGE_CAL',


final_list_V3={'AGE_CAL', 'CODE_GENDER_F' ,'DAYS_LAST_PHONE_CHANGE'}
test_final_list_V3={'AGE_CAL', 'CODE_GENDER_F' ,'DAYS_LAST_PHONE_CHANGE'}

train_df_final_list_V2=train_df[final_list_V2]
test_df_final_list_V2=test_df[test_final_list_V2]

train_df_final_list_V3=train_df[final_list_V3]
test_df_final_list_V3=test_df[test_final_list_V3]

train_df_final_list_V2_with_target=train_df[final_list_V2_with_target]
```

```
correlation_heatmap(train_df_final_list_V2_with_target)
```

```
In [85]: train_df_final_list_V2.describe()
```

Out[85]:

	DAYS_LAST_PHONE_CHANGE	CODE_GENDER_F	AGE_CAL	NAME_EDUCATION_TYPE_Secon
count	307508.000000	307508.000000	307508.000000	
mean	-962.854518	0.658344	43.937135	
std	826.806465	0.474266	11.956076	
min	-4292.000000	0.000000	20.517808	
25%	-1570.000000	0.000000	34.008219	
50%	-757.000000	1.000000	43.150685	
75%	-274.000000	1.000000	53.923288	
max	0.000000	1.000000	69.120548	

0.3 6.2 Train Test Cross Validation Split

the data we use is usually split into training data and test data. The training set contains a known output and the model learns on this data in order to be generalized to other data later on. We have the test dataset (or subset) in order to test our model's prediction on this subset. In order to avoid this, we can perform something called cross validation. It's very similar to train/test split, but it's applied to more subsets. I decided to split size in below side

- Train dataset %60
- Test dataset %20
- Cross Validation dataset %20

References: https://tarangshah.com/blog/2017-12-03/train-validation-and-test-sets/

```
In [86]: from sklearn.model_selection import train_test_split, cross_val_score
         from sklearn.metrics import accuracy_score, classification_report, precision_score, re
         from sklearn.metrics import confusion_matrix, precision_recall_curve, roc_curve, auc,
```

1 7. Model Data

1.1 7.1 Modelling selection

In literature logistic regression have been used for credit risk modeling. So I selected logistic regression modelling approach.

References: https://smartdrill.com/pdf/Credit%20Risk%20Analysis.pdf

1.2 7.2 Model Implementaion

```
In [87]: #Model Alternative 1

         X=train_df_final_list_V2
         y = train_df['TARGET']



         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=

         X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.25, ra


         #X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.95, random_sta
         #old


         # check classification scores of logistic regression
         logreg = LogisticRegression()
         logreg.fit(X_train, y_train)
         y_pred = logreg.predict(X_test)
         y_pred_proba = logreg.predict_proba(X_test)[:, 1]
```

```python
[fpr, tpr, thr] = roc_curve(y_test, y_pred_proba)
print('Train/Test split results:')
print(logreg.__class__.__name__+" accuracy is %2.3f" % accuracy_score(y_test, y_pred))
print(logreg.__class__.__name__+" log_loss is %2.3f" % log_loss(y_test, y_pred_proba))
print(logreg.__class__.__name__+" auc is %2.3f" % auc(fpr, tpr))



idx = np.min(np.where(tpr > 0.95)) # index of the first threshold for which the sensi

plt.figure()
plt.plot(fpr, tpr, color='coral', label='ROC curve (area = %0.3f)' % auc(fpr, tpr))
plt.plot([0, 1], [0, 1], 'k--')
plt.plot([0,fpr[idx]], [tpr[idx],tpr[idx]], 'k--', color='blue')
plt.plot([fpr[idx],fpr[idx]], [0,tpr[idx]], 'k--', color='blue')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate (1 - specificity)', fontsize=14)
plt.ylabel('True Positive Rate (recall)', fontsize=14)
plt.title('Receiver operating characteristic (ROC) curve')
plt.legend(loc="lower right")
plt.show()

print("Using a threshold of %.3f " % thr[idx] + "guarantees a sensitivity of %.3f " %
      "and a specificity of %.3f" % (1-fpr[idx]) +
      ", i.e. a false positive rate of %.2f%%." % (np.array(fpr[idx])*100))
```

```
Train/Test split results:
LogisticRegression accuracy is 0.920
LogisticRegression log_loss is 0.271
LogisticRegression auc is 0.626
```

Using a threshold of 0.043 guarantees a sensitivity of 0.950 and a specificity of 0.109, i.e. a

```
In [88]: #Model Alternative 1
         #Cross Validation
         # check classification scores of logistic regression
         logreg = LogisticRegression()
         logreg.fit(X_train, y_train)
         y_pred = logreg.predict(X_val)
         y_pred_proba = logreg.predict_proba(X_val)[:, 1]
         [fpr, tpr, thr] = roc_curve(y_val, y_pred_proba)
         print('Train/Test split results:')
         print(logreg.__class__.__name__+" accuracy is %2.3f" % accuracy_score(y_val, y_pred))
         print(logreg.__class__.__name__+" log_loss is %2.3f" % log_loss(y_val, y_pred_proba))
         print(logreg.__class__.__name__+" auc is %2.3f" % auc(fpr, tpr))



         idx = np.min(np.where(tpr > 0.95)) # index of the first threshold for which the sensi

         plt.figure()
         plt.plot(fpr, tpr, color='coral', label='ROC curve (area = %0.3f)' % auc(fpr, tpr))
         plt.plot([0, 1], [0, 1], 'k--')
```

```
plt.plot([0,fpr[idx]], [tpr[idx],tpr[idx]], 'k--', color='blue')
plt.plot([fpr[idx],fpr[idx]], [0,tpr[idx]], 'k--', color='blue')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate (1 - specificity)', fontsize=14)
plt.ylabel('True Positive Rate (recall)', fontsize=14)
plt.title('Receiver operating characteristic (ROC) curve')
plt.legend(loc="lower right")
plt.show()

print("Using a threshold of %.3f " % thr[idx] + "guarantees a sensitivity of %.3f " %
      "and a specificity of %.3f" % (1-fpr[idx]) +
      ", i.e. a false positive rate of %.2f%%." % (np.array(fpr[idx])*100))
```

```
Train/Test split results:
LogisticRegression accuracy is 0.919
LogisticRegression log_loss is 0.274
LogisticRegression auc is 0.621
```

```
Using a threshold of 0.042 guarantees a sensitivity of 0.950 and a specificity of 0.092, i.e. a
```

The variable is selected that least correlation of target. After that the variable will be eliminated in model. If the Auc value is not decreasing too much, the variable will be eliminated

In [89]:
```
#Model Alternative 2

X=train_df_final_list_V3
y = train_df['TARGET']



X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=

X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.25, ra


#X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.95, random_sta
#old


# check classification scores of logistic regression
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
y_pred_proba = logreg.predict_proba(X_test)[:, 1]
[fpr, tpr, thr] = roc_curve(y_test, y_pred_proba)
print('Train/Test split results:')
print(logreg.__class__.__name__+" accuracy is %2.3f" % accuracy_score(y_test, y_pred)
print(logreg.__class__.__name__+" log_loss is %2.3f" % log_loss(y_test, y_pred_proba)
print(logreg.__class__.__name__+" auc is %2.3f" % auc(fpr, tpr))



idx = np.min(np.where(tpr > 0.95)) # index of the first threshold for which the sensi

plt.figure()
plt.plot(fpr, tpr, color='coral', label='ROC curve (area = %0.3f)' % auc(fpr, tpr))
plt.plot([0, 1], [0, 1], 'k--')
plt.plot([0,fpr[idx]], [tpr[idx],tpr[idx]], 'k--', color='blue')
plt.plot([fpr[idx],fpr[idx]], [0,tpr[idx]], 'k--', color='blue')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate (1 - specificity)', fontsize=14)
plt.ylabel('True Positive Rate (recall)', fontsize=14)
plt.title('Receiver operating characteristic (ROC) curve')
plt.legend(loc="lower right")
plt.show()

print("Using a threshold of %.3f " % thr[idx] + "guarantees a sensitivity of %.3f " %
      "and a specificity of %.3f" % (1-fpr[idx]) +
      ", i.e. a false positive rate of %.2f%%." % (np.array(fpr[idx])*100))
```

```
Train/Test split results:
LogisticRegression accuracy is 0.920
LogisticRegression log_loss is 0.273
LogisticRegression auc is 0.608
```

```
Using a threshold of 0.047 guarantees a sensitivity of 0.950 and a specificity of 0.106, i.e. a
```

```
In [90]: #Model Alternative 2
         #Cross Validation
         # check classification scores of logistic regression
         logreg = LogisticRegression()
         logreg.fit(X_train, y_train)
         y_pred = logreg.predict(X_val)
         y_pred_proba = logreg.predict_proba(X_val)[:, 1]
         [fpr, tpr, thr] = roc_curve(y_val, y_pred_proba)
         print('Train/Test split results:')
         print(logreg.__class__.__name__+" accuracy is %2.3f" % accuracy_score(y_val, y_pred))
         print(logreg.__class__.__name__+" log_loss is %2.3f" % log_loss(y_val, y_pred_proba))
         print(logreg.__class__.__name__+" auc is %2.3f" % auc(fpr, tpr))
```

```
idx = np.min(np.where(tpr > 0.95)) # index of the first threshold for which the sensi

plt.figure()
plt.plot(fpr, tpr, color='coral', label='ROC curve (area = %0.3f)' % auc(fpr, tpr))
plt.plot([0, 1], [0, 1], 'k--')
plt.plot([0,fpr[idx]], [tpr[idx],tpr[idx]], 'k--', color='blue')
plt.plot([fpr[idx],fpr[idx]], [0,tpr[idx]], 'k--', color='blue')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate (1 - specificity)', fontsize=14)
plt.ylabel('True Positive Rate (recall)', fontsize=14)
plt.title('Receiver operating characteristic (ROC) curve')
plt.legend(loc="lower right")
plt.show()

print("Using a threshold of %.3f " % thr[idx] + "guarantees a sensitivity of %.3f " %
      "and a specificity of %.3f" % (1-fpr[idx]) +
      ", i.e. a false positive rate of %.2f%%." % (np.array(fpr[idx])*100))
```

```
Train/Test split results:
LogisticRegression accuracy is 0.919
LogisticRegression log_loss is 0.276
LogisticRegression auc is 0.609
```

```
Using a threshold of 0.047 guarantees a sensitivity of 0.950 and a specificity of 0.101, i.e. a
```

1.2.1 7.3 Model Refinement

Model altenative 2 is better than first model. Beucause of Cross Validation AUC value is higer than train dataset. In addition to between Model alternative 1 and Model alternative 2 Auc is similiar so I choose the Model Alternative 2

1.2.2 7.4 Model Result

We selected Model alternative 2 * Model Auc is 0.609 * Model Accuray is 0.920
Model varaibles are * AGE_CAL * CODE_GENDER_F * DAYS_LAST_PHONE_CHANGE

1.2.3 7.5 Model Implementation for Test dataset

We will implement the test data set

In [92]:
```
log_reg_pred = logreg.predict_proba(test_df_final_list_V3)[:, 1]
```

In [93]:
```
#Implementation test data set

# Submission dataframe
submit = test_df[['SK_ID_CURR']]
submit['TARGET'] = log_reg_pred

submit.head()
```

```
C:\Users\UTKU\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/user_guide/i
  """
```

Out[93]:

	SK_ID_CURR	TARGET
0	100001	0.048173
1	100005	0.101670
2	100013	0.076617
3	100028	0.065348
4	100038	0.115715

1.3 8 Conclusion

My expectation would be credit type,credit amount or income type for final variable modelling. But these variable were eliminated correlation step. I am surprised for this happen. This proeject aimed to end to end data processing and data modelling in credit risk data. I enjoyed to analyze and create this project

1.4 8.1 Improvement

Gradient Boosting method is more higher creating auc value than logistic regression in literature. For example, Data Scienctist have used Graditent Boosting and reached better than logistic regression results. https://www.kaggle.com/ashishpatel26/home-credit-default-analysis, Model detail contains in this link https://lightgbm.readthedocs.io/en/latest/

2 9 References

- Udacity Data Scientist Nanodegree Program
- Kaggle's Home Credit Default Risk
- Source Data Dictionary
- Creating New Column
- Correlation Elimination
- Train Test Cross Validation
- Credit Risk Modelling
- Lgistic Regression
- Gradient Boosting
- LightGBM's documentation
- Pandas Data Frame Describe
- Pandas Missing Data