Home Credit Default Risk

Team Members

- Anuj Mahajan - anujmaha@iu.edu
- Shubham Jambhale - sjambhal@iu.edu
- Siddhant Patil - sidpatil@iu.edu
- Shashwati Diware - sdiware@iu.edu

Shubham Jambhale
sjambhal@iu.edu

Siddhant Patil
sidpatil@iu.edu

Anuj Mahajan
anujmaha@iu.edu

Shashwati Diware
sdiware@iu.edu

1.0 FP*GroupN* 11 HCDR

1.1 Phase Leader Plan

Phase	Contributor	Contribution Description
Phase 1: Project Planning	Anuj Mahajan	Download Data, go through data, and load libraries. Create a pipeline diagram and describe the pipeline design. Describe Preprocessing,
Phase 1: Project Planning	Shashwati Diware	Project Abstract, ML Algorithm Names, and describe Metrics.
Phase 1: Project Planning	Shubham Jambhale(Phase Leader)	Understanding the problem statement, and writing table descriptions. Schedule meetings, coordinate tasks, plan phase
Phase 1: Project Planning	Siddhant Patil	Machine Learning Pipeline Steps and describes pipeline components.
Phase 2: Base Line Modelling and EDA	Anuj Mahajan (Phase Leader)	Creating Block Diagram EDA and one slide of the presentation. Schedule meetings, coordinate tasks, plan phase
Phase 2: Base Line Modelling and EDA	Shashwati Diware	Result Analysis EDA and one slide of the presentation.
Phase 2: Base Line Modelling and EDA	Shubham jambhale	Result Analysis and two slides of the presentation
Phase 2: Base Line Modelling and EDA	Siddhant Patil	Result Analysis and two slides of the presentation
Phase 3: Hyperparameter Tuning	Shashwati Diware (Phase Leader)	Testing Accuracy matrix and Schedule meetings, coordinating tasks, the planning phase
Phase 3: Hyperparameter Tuning	Siddhant Patil	Create and develop code for Hyperparameter tuning
Phase 3: Hyperparameter Tuning	Shubham Jambhale	Run and create analysis by testing the confusion / AUC matrix. Coordinate Tasks and one slide of the presentation
Phase 3: Hyperparameter Tuning	Anuj Mahajan	Run and analyze Lasso and ridge regression losses. Coordinate tasks and one slide of the presentation
Phase 4: Final Report Generation	Siddhant Patil (Phase Leader)	Plan Phase Schedule Meetings and Coordinate Tasks, analyze and go through the final results
Phase 4: Final Report Generation	Anuj Mahajan	Rearrange everything and go through the final documentation, list down the final recordings
Phase 4: Final Report Generation	Shashwati Diware	Prepare the final presentation
Phase 4: Final Report Generation	Shubham Jambhale	Check everything and submit the assignment before the deadline

1.2 Credit Assignment Plan

Phase 1:

Task	Task Description	Hours spent	Assigned to	Start	End
Understanding problem statement	Go through the problem statement to understand the requirements	6	Shubham	11/05/22	11/07/22
Data Exploration	Explore and analyze the data for a better understanding	6	Anuj	11/07/22	11/09/22
Project Proposal	Creating the project proposal and preparing a basic report with Abstract, ML models, and Gantt diagram	20	Group	11/09/22	11/14/22

Phase 2:

Task	Task Description	Hours Spent	Assigned to	Start	End
Creating Block Diagram	Creating the block diagram of the basic flow of execution.	5	Anuj	11/13/22	11/15/22
Creating Pipeline Diagram	Creating the pipeline diagram of the machine learning model from analyzing the data till the result analysis	5	Shashwati	11/13/22	11/15/22
Result Analysis	Analyzing the Result	10	Group	11/26/22	11/29/22
PowerPoint Presentation	Simultaneously prepare the PowerPoint presentation and add the analyzed data into it as per need	10	Group	11/20/22	11/29/22

Phase 3:

Task	Task Description	Hours spent	Assigned to	Start	End
Create and develop code for hyperparameter tuning	Design and develop python helper function for hyperparameter tuning	16	Siddhant	11/20/22	11/25/22
Result Analysis	Analysis of Obtained Result	2	Group	12/02/22	12/03/22
Testing Accuracy matrix	Analyzing accuracy using accuracy matrix	2	Shashwati	12/03/22	12/04/22
Analyzing the Loss and AUC	Analyzing Loss and AUC matrix	2	Shubham	12/03/22	12/04/22
Creating Powerpoint presentation	Simultaneously prepare the PowerPoint presentation and add the analyzed data into it as per need	2	Anuj	12/03/22	12/04/22

Phase 4:

Task	Task Description	Hours Spent	Assigned To	Start	End
Build MLP using Pytorch	Build a neural network model using Pytorch	10	Group	12/03/22	12/08/22
Final Documentation	Rearrange everything and go through the final documentation, list down the final recordings	10	Anuj	12/03/22	12/08/22
Final Results	Analyze final results obtained after the final testing	6	Siddhant	12/05/22	12/08/22
Final Presentation	Prepare the final presentation	4	Shashwati	12/06/22	12/08/22
Assignment Submission	Check everything and submit the assignment before the deadline	1	Shubham	12/08/22	12/09/22

1.3 Abstract

Based on historical credit histories and repayment trends utilizing machine learning modeling, Home Credit offers unsecured lending. A user-generated credit score is calculated using criteria like the balance that the user has maintained. As part of this project, we are predicting the customer repayment status such as if the user is a defaulter or not using machine learning pipelines and models using the datasets provided by Kaggle. The data collection includes seven separate tables that aid in determining the user status, including bureau balance, credit card balance, home credit column detection, Installments payments, POS CASH balance, and previous applications. In phase 3, we provide feature engineering, hyperparameter tuning, and modeling pipelines. We experimented with selected features for Logistic regression, Decision Making Tree, Random Forest, Lasso, and Ridge Regressions. The Decision Tree has the highest test accuracy with 92.12, followed by Logistic regression and Random Forest with a test accuracy of 91.98. We received 0.5 ROC AUC from a Kaggle submission.

1.4 Data and Task Description

- Data source:
  - We are planning to use the existing datasets provided by Kaggle. Source: https://www.kaggle.com/c/home-credit-default-risk/data
- POS_CASH_balance.csv:
  - This dataset gives information about previous credit information such as contract status, the number of installments left to pay, DPD(days past due), etc. of the current application.

	SK_ID_PREV	SK_ID_CURR	MONTHS_BALANCE	CNT_INSTALMENT	CNT_INSTALMENT_FUTURE	NAME_CONTRACT_STATUS	SK_DPD	SK_DPD_DEF
0	1803195	182943	-31	48.0	45.0	Active	0	0
1	1715348	367990	-33	36.0	35.0	Active	0	0
2	1784872	397406	-32	12.0	9.0	Active	0	0
3	1903291	269225	-35	48.0	42.0	Active	0	0
4	2341044	334279	-35	36.0	35.0	Active	0	0

- bureau.csv
  - This dataset gives information about the type of credit, debt, limit, overdue, maximum overdue, annuity, remaining days for previous credit, etc.

	SK_ID_BUREAU	MONTHS_BALANCE	STATUS
0	5715448	0	C
1	5715448	-1	C
2	5715448	-2	C
3	5715448	-3	C
4	5715448	-4	C

- bureau_balance.csv:
  - This dataset gives information about the Status of the Credit Bureau loan during the month, the Month of balance relative to the application date, Recoded ID of the Credit Bureau credit. Each row is one month of a previous credit, and a single previous credit can have multiple rows, one for each month of the credit length.

	SK_ID_BUREAU	MONTHS_BALANCE	STATUS
0	5715448	0	C
1	5715448	-1	C
2	5715448	-2	C
3	5715448	-3	C
4	5715448	-4	C

  - credit_card_balance.csv:
    - This dataset gives information about financial transactions aggregated values such as amount received, drawings, number of transactions of previous credit, installments, etc. Each row is one month of a credit card balance, and a single credit card can have many rows.

	SK_ID_PREV	SK_ID_CURR	MONTHS_BALANCE	AMT_BALANCE	AMT_CREDIT_LIMIT_ACTUAL	AMT_DRAWINGS_ATM_CURRENT	AMT_DRAWINGS_CURRENT
0	2562384	378907	-6	56.970	135000	0.0	877.5
1	2582071	363914	-1	63975.555	45000	2250.0	2250.0
2	1740877	371185	-7	31815.225	450000	0.0	0.0
3	1389973	337855	-4	236572.110	225000	2250.0	2250.0
4	1891521	126868	-1	453919.455	450000	0.0	11547.0

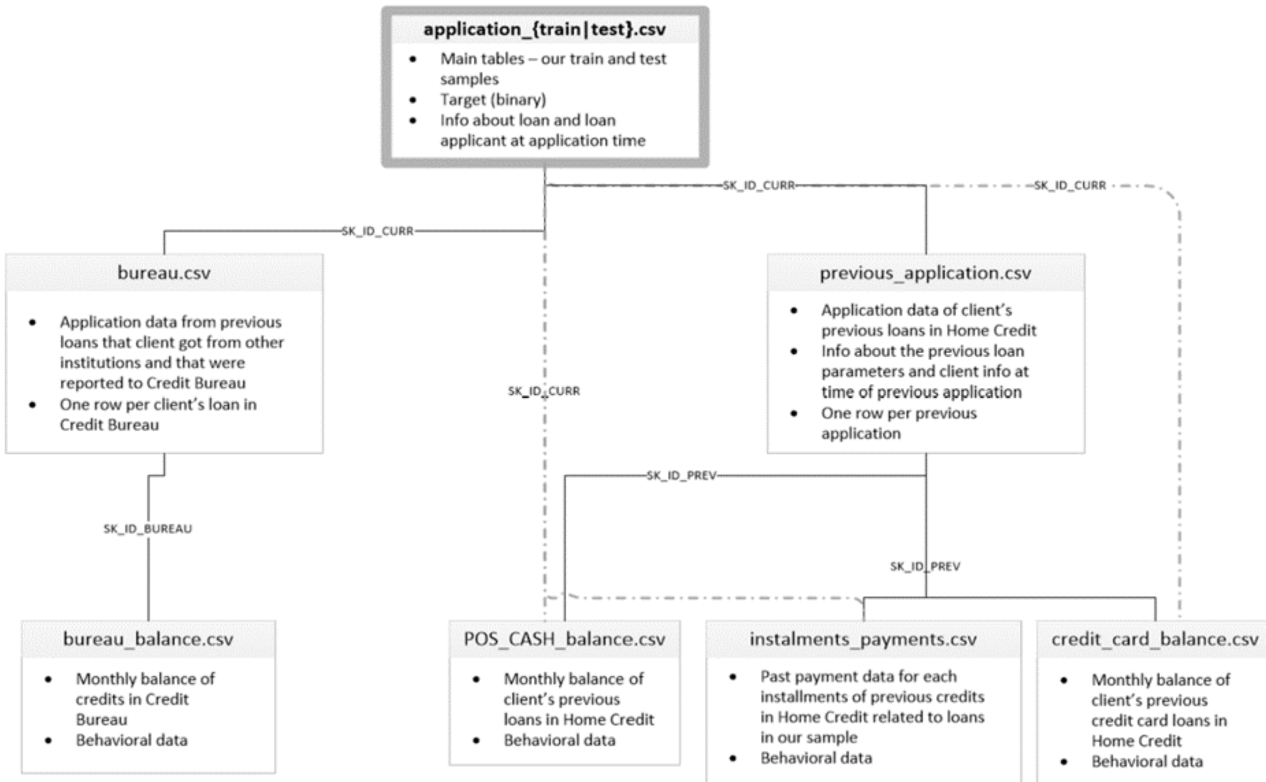
- installments_payments.csv:
  - This dataset gives information about payments, installments supposed to be paid, and their details. There is one row for every made payment and one row for every missed payment.

	SK_ID_PREV	SK_ID_CURR	NUM_INSTALMENT_VERSION	NUM_INSTALMENT_NUMBER	DAYS_INSTALMENT	DAYS_ENTRY_PAYMENT	AMT_INSTALMENT
0	1054186	161674	1.0	6	-1180.0	-1187.0	6948.360
1	1330831	151639	0.0	34	-2156.0	-2156.0	1716.525
2	2085231	193053	2.0	1	-63.0	-63.0	25425.000
3	2452527	199697	1.0	3	-2418.0	-2426.0	24350.130
4	2714724	167756	1.0	2	-1383.0	-1366.0	2165.040

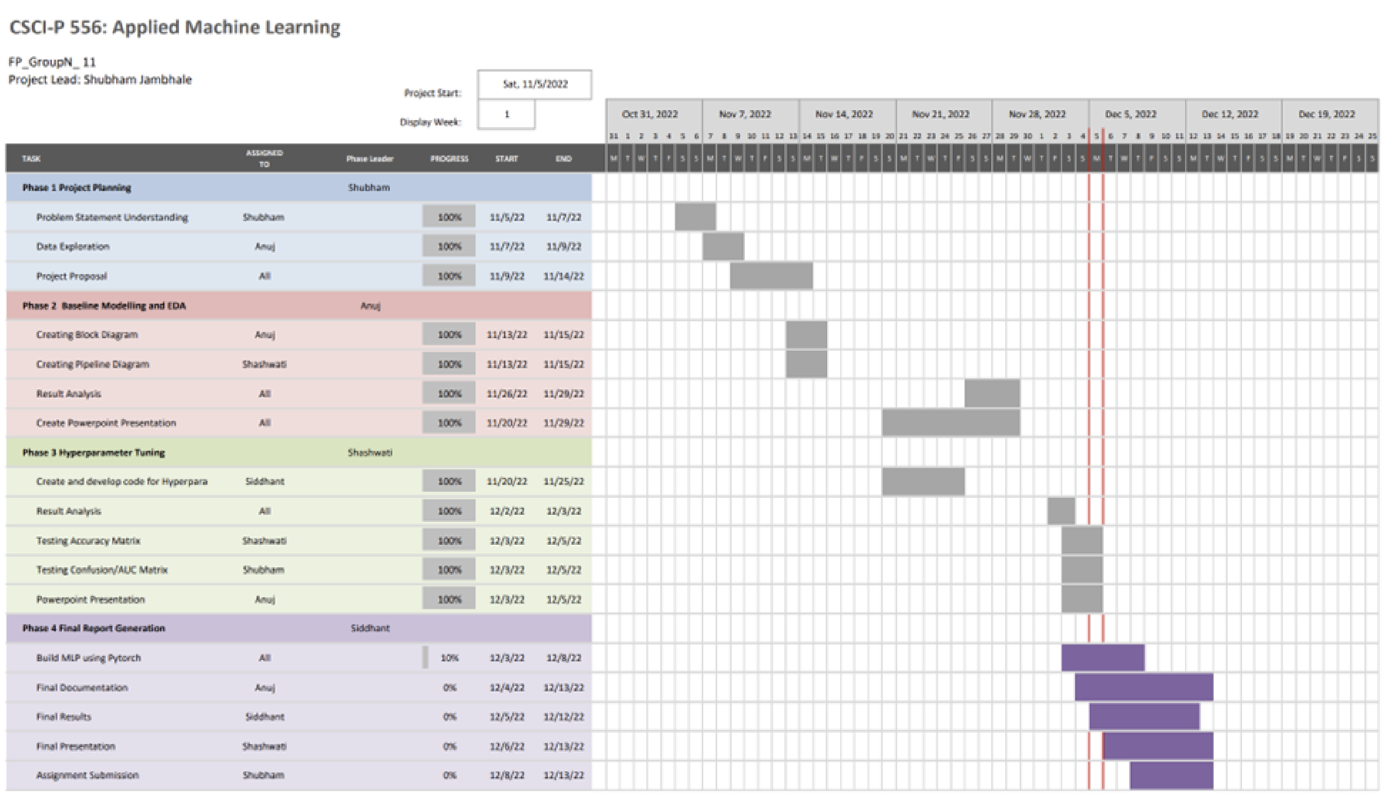
  - previous_application.csv
    - This dataset contains information about previous application details of an application. Each current loan in the application data can have multiple previous loans. Each previous application has one

row and is identified by the feature SK_ID_PREV.

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT
0	2030495	271877	Consumer loans	1730.430	17145.0	17145.0	0.0
1	2802425	108129	Cash loans	25188.615	607500.0	679671.0	NaN
2	2523466	122040	Cash loans	15060.735	112500.0	136444.5	NaN
3	2819243	176158	Cash loans	47041.335	450000.0	470790.0	NaN
4	1784265	202054	Cash loans	31924.395	337500.0	404055.0	NaN



1.5 Gantt Chart



1.6 Machine Learning Algorithms and Metrics

The outcome of this project is to predict, whether the customer will repay the loan or not. That's why this is a classification task where the outcome is 0 or 1. To classify this problem we will be building the following machine-learning models:

1. Logistics Regression:
   • In our case, the number of features is relatively small i.e. <1000, and no. of examples is large. Hence logistic regression can be a good fit here for the classification.

2. Decision Tree:
   • Decision trees are better for categorical data and our target data is also categorical in nature that's why decision trees are a good fit.

3. Random Forest:
   • Random Forest works well with a mixture of numerical and categorical features. • As we have a good amount of mixture of both types of features random forest can be a good fit.

4. Lasso Regression:
   • The bias-variance trade-off is the basis for Lasso's superiority over least squares. The lasso solution can result in a decrease in variance at the cost of a slight increase in bias when the variance of the least squares estimates is very large. Consequently, this can produce predictions that are more accurate.

5. Ridge Regression:
   • Any data that exhibits multicollinearity can be analyzed using the model-tuning technique known as ridge regression. This technique carries out L2 regularization. Predicted values differ much from real values when the problem of multicollinearity arises, least-squares are unbiased, and variances are significant.

1.6.1 Loss Function

- Log loss
  - How closely the forecast probability matches the associated real or true value is indicated by log-loss (0 or 1 in case of binary classification). The higher the log-loss number, the more the predicted probability deviates from the actual value.

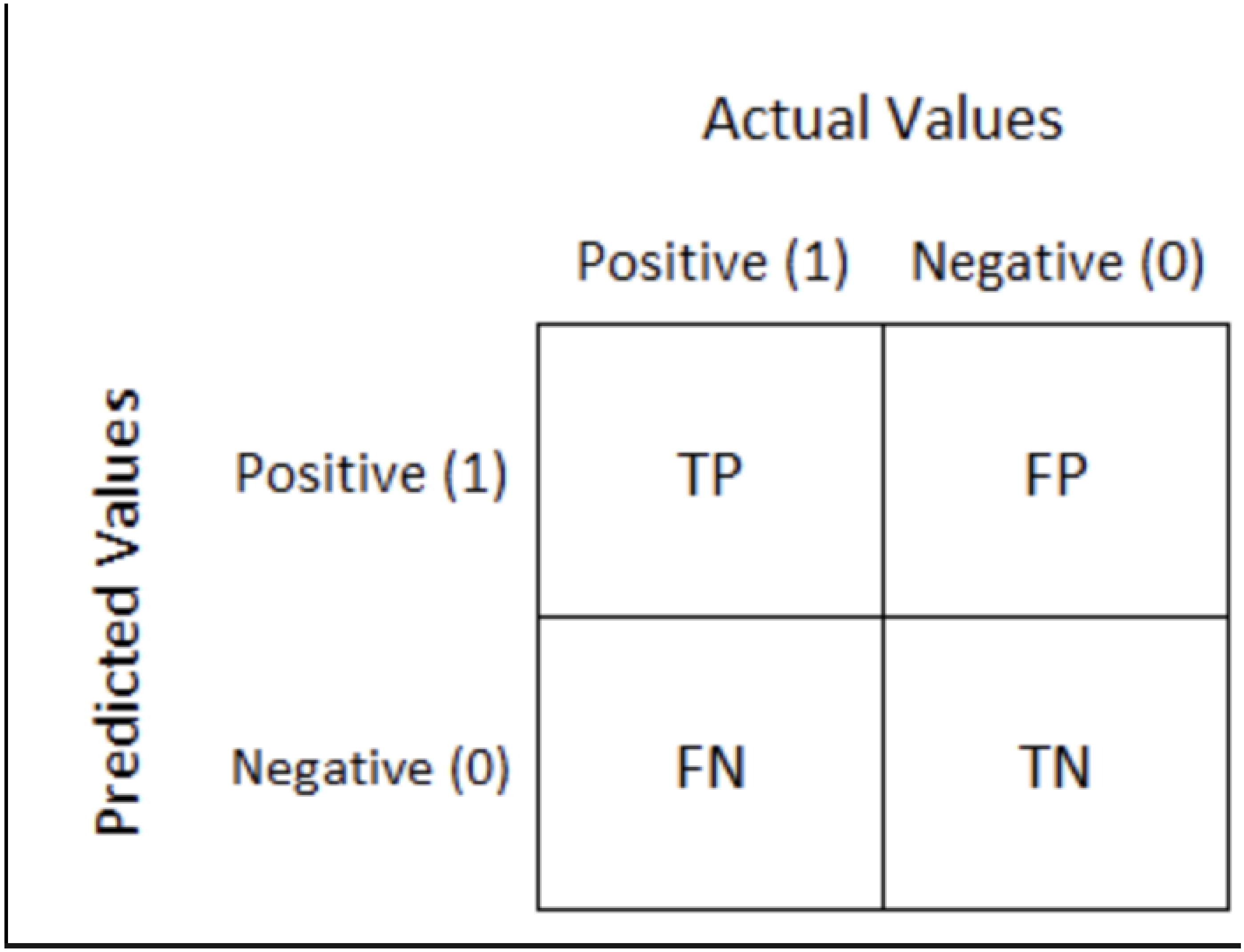
1.6.2 Metrics

In [1]:
```
!pip install latexify-py==0.2.0
import math
import latexify
```

```
Requirement already satisfied: latexify-py==0.2.0 in d:\anaconda_installation\lib\site-pac
kages (0.2.0)
Requirement already satisfied: dill>=0.3.2 in d:\anaconda_installation\lib\site-packages
(from latexify-py==0.2.0) (0.3.6)
```

1. Confusion Metrics:
   - A confusion matrix, also called an error matrix, is used in the field of machine learning and more specifically in the challenge of classification. Confusion matrices show counts between expected and observed values. The result "TN" stands for True Negative and displays the number of negatively classed cases that were correctly identified. Similar to this, "TP" stands for True Positive and denotes the quantity of correctly identified positive cases. The term "FP" denotes the number of real negative cases that were mistakenly categorized as positive, while "FN" denotes the number of real positive examples

that were mistakenly classed as negative. Accuracy is one of the most often used metrics in classification.



1. AUC:
   - AUC stands for "Area under the ROC Curve." It measures the entire two-dimensional area underneath the entire ROC curve from (0,0) to (1,1). It is a widely used accuracy method for binary classification problems.
2. Accuracy:
   - The accuracy score is used to gauge the model's effectiveness by calculating the ratio of total true positives to total true negatives across all made predictions. Accuracy is generally used to calculate binary classification models.

In [2]:

```
@latexify.function(use_math_symbols=True)
def Accuracy():
    return(True_Positives + True_Negatives) / (True_Positives +
True_Negatives + False_Positives + False_Negatives)

Accuracy
```

Out[2]:

$$\mathrm{Accuracy}() = \frac{True_{P}ositives + True_{N}egatives}{True_{P}ositives + True_{N}egatives + False_{P}ositives + False_{N}egatives}$$

In [3]:

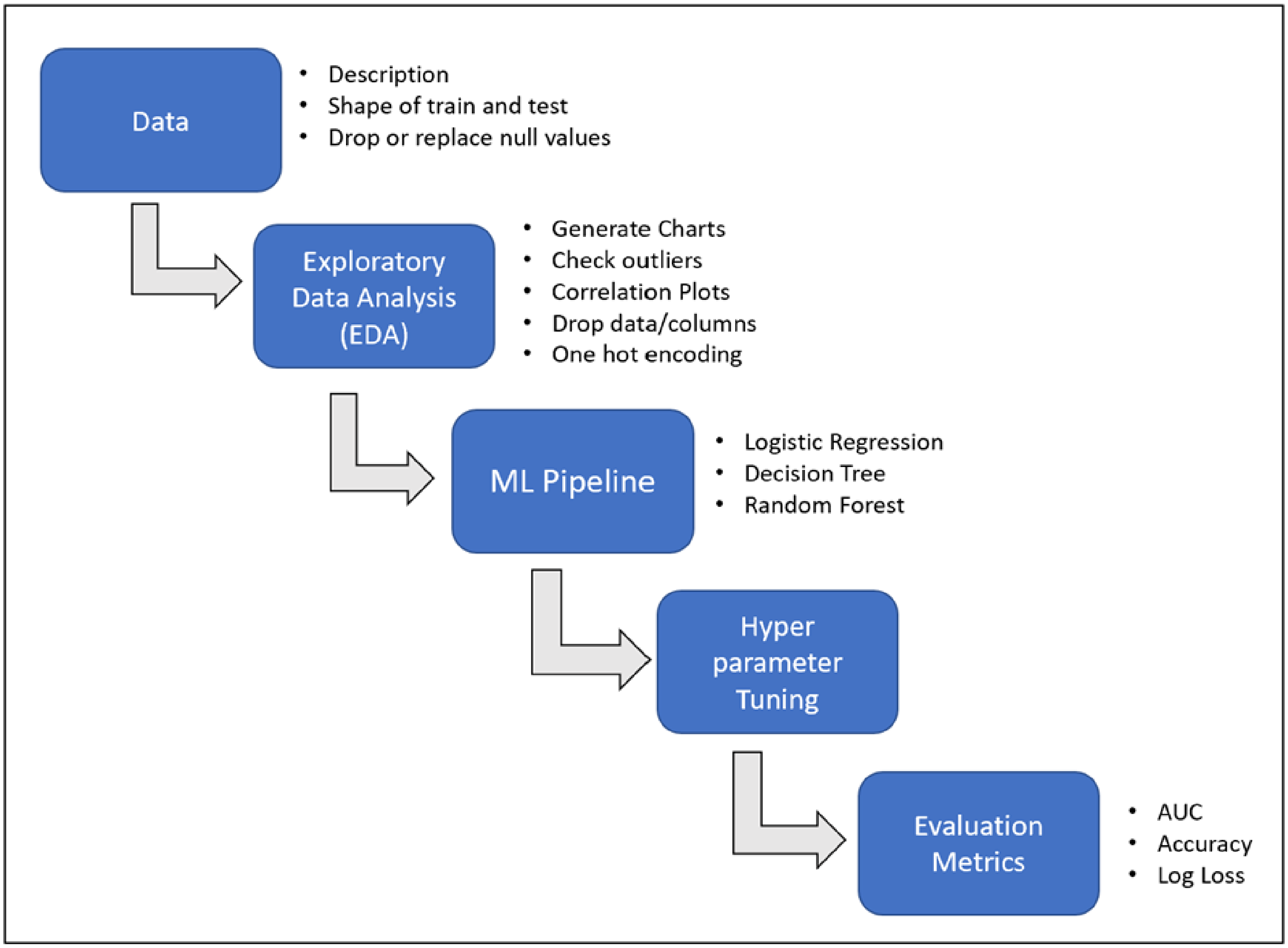
```
@latexify.function(use_math_symbols=True)
```

```
def logloss():
    return (-1/m*(sum(y*np.log(p)+(1- y)*np.log(1-p))))
logloss
```

Out[3]:

$$\mathrm{logloss}() = \frac{-1}{m} \sum (y \log(p) + (1-y)\log(1-p))$$

1.7 Machine Learning Pipeline Steps



- Data Preprocessing:
  • Convert the raw data set into a clean data set for processing.
  • First, Obtain Kaggle's raw data.
  • On this Raw Data. Analyze exploratory data.

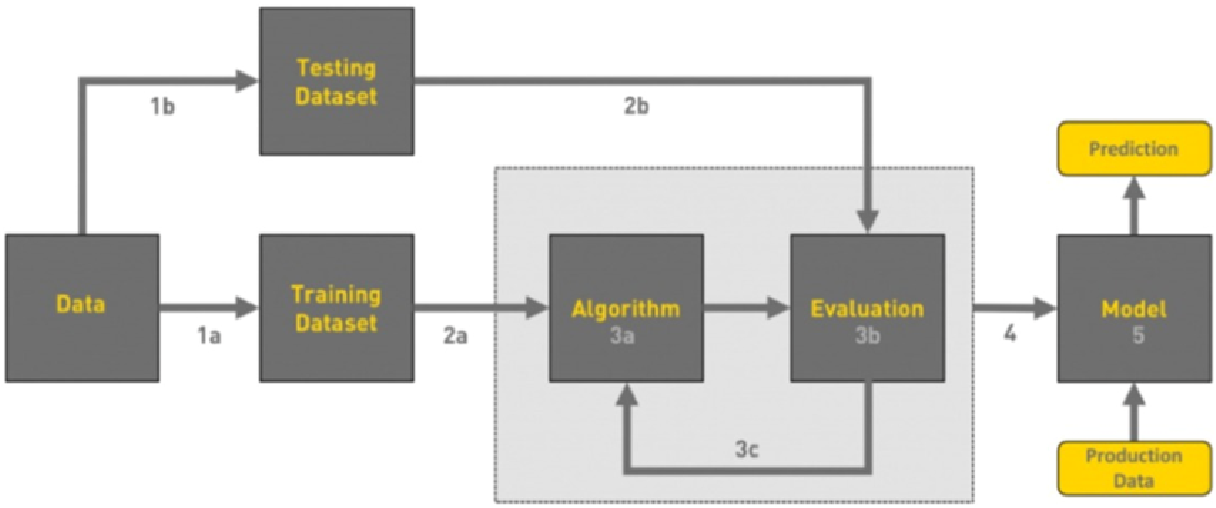
- Feature Engineering:
  • Create a suitable input dataset by performing feature engineering and other processing techniques.
  • Pipeline must not only select the features it wants to create from an unlimited pool of possibilities, but it must also process vast amounts of data to do so. This makes the data appropriate for the model.

- Model Selection:
  • Here, we try on different models for various option purposes.
  • Develop and test several candidate models, such as Random Forest, Decision Making Trees, and Logistic Regression.
  • Using the evaluation function, pick the top model with a good evaluation score.

• For this selection purposes, employ many measures for evaluation criteria, including "Accuracy," "F1 Score,".

- Prediction Generation:
  • The top performer is then chosen as the winning model when the models are tested on a new set of data that wasn't used during training.
  • Once the best model has been chosen, use it to forecast outcomes based on the fresh data.
  • It is then used to make predictions across all your objects.

1.8 Block Diagram



Overview of the Workflow of ML

Referenced from: https://towardsdatascience.com/workflow-of-a-machine-learning-project-ec1dba419b94

In [4]:
```
#!pip install opendatasets
```

In [5]:
```
#pip install pandas
```

In [6]:
```
#pwd
```

In [7]:
```
#ls -l .kaggle\kaggle.json
```

In [8]:
```
#!mkdir .kaggle
```

In [9]:
```
#!mkdir ~\.kaggle
```

In [10]:
```
#mkdir \.kaggle
```

In [11]:
```
#ls -l .kaggle
```

In [12]:
```
#pwd
```

In [13]:
```
#!chmod 600 C:\\Users\\jambh\\.kaggle\\kaggle.json
```

In [14]:
```
#!kaggle competitions
```

In [15]:
```
#DATA_DIR = './././Data/home-credit-default-risk'
```

In [16]:
```
#!mkdir DATA_DIR
```

In [17]:
```
#!kaggle competitions download home-credit-default-risk -p .\\Data\\home-credit-default-ri
#! kaggle competitions download home-credit-default-risk -p $DATA_DIR
```

In [18]:
```
'''import zipfile
unzippingReq = True #True
if unzippingReq: #please modify this code
    zip_ref = zipfile.ZipFile('./DATA_DIR/home-credit-default-risk.zip', 'r')
    zip_ref.extractall('./DATA_DIR')
    zip_ref.close()'''
```

Out[18]:
```
"import zipfile\nunzippingReq = True #True\nif unzippingReq: #please modify this code \n
  zip_ref = zipfile.ZipFile('./DATA_DIR/home-credit-default-risk.zip', 'r')\n    zip_ref.e
xtractall('./DATA_DIR') \n    zip_ref.close()"
```

In [2]:
```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
import os
import zipfile
from sklearn.base import BaseEstimator, TransformerMixin
import seaborn as sns
from sklearn.linear_model import Lasso,Ridge,LogisticRegression
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import MinMaxScaler
from sklearn.pipeline import Pipeline, FeatureUnion
from pandas.plotting import scatter_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import OneHotEncoder
import warnings
warnings.filterwarnings('ignore')

import warnings
warnings.simplefilter('ignore')
import seaborn as sea
import matplotlib.pyplot as plt
%matplotlib inline

from sklearn.model_selection import train_test_split
import re
from time import time
from scipy import stats
import json
from sklearn.model_selection import ShuffleSplit
from sklearn.linear_model import LogisticRegression
```

```python
#from sklearn.svm import SVC
#from sklearn.linear_model import SGDClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import roc_auc_score, log_loss, accuracy_score
from sklearn.metrics import confusion_matrix

from IPython.display import display, Math, Latex

def load_data(in_path, name):
    df = pd.read_csv(in_path)
    print(f"{name}: shape is {df.shape}")
    print(df.info())
    display(df.head(5))
    return df

datasets={}
ds_name = 'application_train'
DATA_DIR='./DATA_DIR'
datasets[ds_name] = load_data(os.path.join(DATA_DIR, f'{ds_name}.csv'), ds_name)

datasets['application_train'].shape
```

```
application_train: shape is (307511, 122)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
None
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDI
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 rows × 122 columns

Out[2]:

```
(307511, 122)
```

In [3]:

```python
ds_name = 'application_test'
datasets[ds_name] = load_data(os.path.join(DATA_DIR, f'{ds_name}.csv'), ds_name)
```

```
application_test: shape is (48744, 121)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48744 entries, 0 to 48743
Columns: 121 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(40), object(16)
memory usage: 45.0+ MB
None
```

	SK_ID_CURR	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT
0	100001	Cash loans	F	N	Y	0	
1	100005	Cash loans	M	N	Y	0	

	SK_ID_CURR	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT
2	100013	Cash loans	M	Y	Y	0	
3	100028	Cash loans	F	N	Y	2	
4	100038	Cash loans	M	Y	N	1	

5 rows × 121 columns

In [6]:

```
%%time
ds_names = ("application_train", "application_test", "bureau","bureau_balance","credit_car
            "previous_application","POS_CASH_balance")

for ds_name in ds_names:
    datasets[ds_name] = load_data(os.path.join(DATA_DIR, f'{ds_name}.csv'), ds_name)
```

```
application_train: shape is (307511, 122)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
None
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDI
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 rows × 122 columns

```
application_test: shape is (48744, 121)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48744 entries, 0 to 48743
Columns: 121 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(40), object(16)
memory usage: 45.0+ MB
None
```

	SK_ID_CURR	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT
0	100001	Cash loans	F	N	Y	0	
1	100005	Cash loans	M	N	Y	0	
2	100013	Cash loans	M	Y	Y	0	
3	100028	Cash loans	F	N	Y	2	
4	100038	Cash loans	M	Y	N	1	

5 rows × 121 columns

```
bureau: shape is (1716428, 17)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1716428 entries, 0 to 1716427
Data columns (total 17 columns):
 #   Column                  Dtype
---  ------                  -----
```

```
 0   SK_ID_CURR              int64
 1   SK_ID_BUREAU            int64
 2   CREDIT_ACTIVE           object
 3   CREDIT_CURRENCY         object
 4   DAYS_CREDIT             int64
 5   CREDIT_DAY_OVERDUE      int64
 6   DAYS_CREDIT_ENDDATE     float64
 7   DAYS_ENDDATE_FACT       float64
 8   AMT_CREDIT_MAX_OVERDUE  float64
 9   CNT_CREDIT_PROLONG      int64
 10  AMT_CREDIT_SUM          float64
 11  AMT_CREDIT_SUM_DEBT     float64
 12  AMT_CREDIT_SUM_LIMIT    float64
 13  AMT_CREDIT_SUM_OVERDUE  float64
 14  CREDIT_TYPE             object
 15  DAYS_CREDIT_UPDATE      int64
 16  AMT_ANNUITY             float64
dtypes: float64(8), int64(6), object(3)
memory usage: 222.6+ MB
None
```

	SK_ID_CURR	SK_ID_BUREAU	CREDIT_ACTIVE	CREDIT_CURRENCY	DAYS_CREDIT	CREDIT_DAY_OVERDUE	DAYS_CRE
0	215354	5714462	Closed	currency 1	-497	0	
1	215354	5714463	Active	currency 1	-208	0	
2	215354	5714464	Active	currency 1	-203	0	
3	215354	5714465	Active	currency 1	-203	0	
4	215354	5714466	Active	currency 1	-629	0	

```
bureau_balance: shape is (27299925, 3)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27299925 entries, 0 to 27299924
Data columns (total 3 columns):
 #   Column          Dtype 
---  ------          ----- 
 0   SK_ID_BUREAU    int64 
 1   MONTHS_BALANCE  int64 
 2   STATUS          object
dtypes: int64(2), object(1)
memory usage: 624.8+ MB
None
```

	SK_ID_BUREAU	MONTHS_BALANCE	STATUS
0	5715448	0	C
1	5715448	-1	C
2	5715448	-2	C
3	5715448	-3	C
4	5715448	-4	C

```
credit_card_balance: shape is (3840312, 23)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3840312 entries, 0 to 3840311
Data columns (total 23 columns):
 #   Column                      Dtype  
---  ------                      -----  
 0   SK_ID_PREV                  int64  
 1   SK_ID_CURR                  int64  
```

```
 2   MONTHS_BALANCE              int64
 3   AMT_BALANCE                 float64
 4   AMT_CREDIT_LIMIT_ACTUAL     int64
 5   AMT_DRAWINGS_ATM_CURRENT    float64
 6   AMT_DRAWINGS_CURRENT        float64
 7   AMT_DRAWINGS_OTHER_CURRENT  float64
 8   AMT_DRAWINGS_POS_CURRENT    float64
 9   AMT_INST_MIN_REGULARITY     float64
 10  AMT_PAYMENT_CURRENT         float64
 11  AMT_PAYMENT_TOTAL_CURRENT   float64
 12  AMT_RECEIVABLE_PRINCIPAL    float64
 13  AMT_RECIVABLE               float64
 14  AMT_TOTAL_RECEIVABLE        float64
 15  CNT_DRAWINGS_ATM_CURRENT    float64
 16  CNT_DRAWINGS_CURRENT        int64
 17  CNT_DRAWINGS_OTHER_CURRENT  float64
 18  CNT_DRAWINGS_POS_CURRENT    float64
 19  CNT_INSTALMENT_MATURE_CUM   float64
 20  NAME_CONTRACT_STATUS        object
 21  SK_DPD                      int64
 22  SK_DPD_DEF                  int64
dtypes: float64(15), int64(7), object(1)
memory usage: 673.9+ MB
None
```

	SK_ID_PREV	SK_ID_CURR	MONTHS_BALANCE	AMT_BALANCE	AMT_CREDIT_LIMIT_ACTUAL	AMT_DRAWINGS_ATM
0	2562384	378907	-6	56.970	135000	
1	2582071	363914	-1	63975.555	45000	
2	1740877	371185	-7	31815.225	450000	
3	1389973	337855	-4	236572.110	225000	
4	1891521	126868	-1	453919.455	450000	

5 rows × 23 columns

```
installments_payments: shape is (13605401, 8)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13605401 entries, 0 to 13605400
Data columns (total 8 columns):
 #   Column                  Dtype  
---  ------                  -----  
 0   SK_ID_PREV              int64  
 1   SK_ID_CURR              int64  
 2   NUM_INSTALMENT_VERSION  float64
 3   NUM_INSTALMENT_NUMBER   int64  
 4   DAYS_INSTALMENT         float64
 5   DAYS_ENTRY_PAYMENT      float64
 6   AMT_INSTALMENT          float64
 7   AMT_PAYMENT             float64
dtypes: float64(5), int64(3)
memory usage: 830.4 MB
None
```

	SK_ID_PREV	SK_ID_CURR	NUM_INSTALMENT_VERSION	NUM_INSTALMENT_NUMBER	DAYS_INSTALMENT	DAYS_EI
0	1054186	161674	1.0	6	-1180.0	
1	1330831	151639	0.0	34	-2156.0	
2	2085231	193053	2.0	1	-63.0	
3	2452527	199697	1.0	3	-2418.0	
4	2714724	167756	1.0	2	-1383.0	

```
previous_application: shape is (1670214, 37)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 37 columns):
 #   Column                       Non-Null Count    Dtype  
---  ------                       --------------    -----  
 0   SK_ID_PREV                   1670214 non-null  int64  
 1   SK_ID_CURR                   1670214 non-null  int64  
 2   NAME_CONTRACT_TYPE           1670214 non-null  object 
 3   AMT_ANNUITY                  1297979 non-null  float64
 4   AMT_APPLICATION              1670214 non-null  float64
 5   AMT_CREDIT                   1670213 non-null  float64
 6   AMT_DOWN_PAYMENT             774370 non-null   float64
 7   AMT_GOODS_PRICE              1284699 non-null  float64
 8   WEEKDAY_APPR_PROCESS_START   1670214 non-null  object 
 9   HOUR_APPR_PROCESS_START      1670214 non-null  int64  
 10  FLAG_LAST_APPL_PER_CONTRACT  1670214 non-null  object 
 11  NFLAG_LAST_APPL_IN_DAY       1670214 non-null  int64  
 12  RATE_DOWN_PAYMENT            774370 non-null   float64
 13  RATE_INTEREST_PRIMARY        5951 non-null     float64
 14  RATE_INTEREST_PRIVILEGED     5951 non-null     float64
 15  NAME_CASH_LOAN_PURPOSE       1670214 non-null  object 
 16  NAME_CONTRACT_STATUS         1670214 non-null  object 
 17  DAYS_DECISION                1670214 non-null  int64  
 18  NAME_PAYMENT_TYPE            1670214 non-null  object 
 19  CODE_REJECT_REASON           1670214 non-null  object 
 20  NAME_TYPE_SUITE              849809 non-null   object 
 21  NAME_CLIENT_TYPE             1670214 non-null  object 
 22  NAME_GOODS_CATEGORY          1670214 non-null  object 
 23  NAME_PORTFOLIO               1670214 non-null  object 
 24  NAME_PRODUCT_TYPE            1670214 non-null  object 
 25  CHANNEL_TYPE                 1670214 non-null  object 
 26  SELLERPLACE_AREA             1670214 non-null  int64  
 27  NAME_SELLER_INDUSTRY         1670214 non-null  object 
 28  CNT_PAYMENT                  1297984 non-null  float64
 29  NAME_YIELD_GROUP             1670214 non-null  object 
 30  PRODUCT_COMBINATION          1669868 non-null  object 
 31  DAYS_FIRST_DRAWING           997149 non-null   float64
 32  DAYS_FIRST_DUE               997149 non-null   float64
 33  DAYS_LAST_DUE_1ST_VERSION    997149 non-null   float64
 34  DAYS_LAST_DUE                997149 non-null   float64
 35  DAYS_TERMINATION             997149 non-null   float64
 36  NFLAG_INSURED_ON_APPROVAL    997149 non-null   float64
dtypes: float64(15), int64(6), object(16)
memory usage: 471.5+ MB
None
```

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN
0	2030495	271877	Consumer loans	1730.430	17145.0	17145.0	
1	2802425	108129	Cash loans	25188.615	607500.0	679671.0	
2	2523466	122040	Cash loans	15060.735	112500.0	136444.5	
3	2819243	176158	Cash loans	47041.335	450000.0	470790.0	
4	1784265	202054	Cash loans	31924.395	337500.0	404055.0	

5 rows × 37 columns

```
POS_CASH_balance: shape is (10001358, 8)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10001358 entries, 0 to 10001357
Data columns (total 8 columns):
 #   Column                 Dtype  
---  ------                 -----  
```

```
 0   SK_ID_PREV             int64
 1   SK_ID_CURR             int64
 2   MONTHS_BALANCE         int64
 3   CNT_INSTALMENT         float64
 4   CNT_INSTALMENT_FUTURE  float64
 5   NAME_CONTRACT_STATUS   object
 6   SK_DPD                 int64
 7   SK_DPD_DEF             int64
dtypes: float64(2), int64(5), object(1)
memory usage: 610.4+ MB
None
```

	SK_ID_PREV	SK_ID_CURR	MONTHS_BALANCE	CNT_INSTALMENT	CNT_INSTALMENT_FUTURE	NAME_CONTRACT_S
0	1803195	182943	-31	48.0	45.0	
1	1715348	367990	-33	36.0	35.0	
2	1784872	397406	-32	12.0	9.0	
3	1903291	269225	-35	48.0	42.0	
4	2341044	334279	-35	36.0	35.0	

```
Wall time: 27.7 s
```

In [7]:

```
for ds_name in datasets.keys():
    print(f'dataset {ds_name:24}: [ {datasets[ds_name].shape[0]:10,}, {datasets[ds_name].s
```

```
dataset application_train       : [    307,511, 122]
dataset application_test        : [     48,744, 121]
dataset bureau                  : [  1,716,428, 17]
dataset bureau_balance          : [ 27,299,925, 3]
dataset credit_card_balance     : [  3,840,312, 23]
dataset installments_payments   : [ 13,605,401, 8]
dataset previous_application    : [  1,670,214, 37]
dataset POS_CASH_balance        : [ 10,001,358, 8]
```

In [8]:

```
data = datasets['application_train'].copy()
y = data['TARGET']
X = data.drop(['SK_ID_CURR','TARGET'], axis = 1)
```

EXPLORATORY DATA ANALYSIS

In [9]:

```
application_test = datasets['application_test'].copy()
application_train = datasets['application_train'].copy()
```

In [10]:

```
def Exploratory_Data_Analysis(dataframe,dataframe_name):
    print("Test description; data type: {}".format(dataframe_name))
    print(dataframe.dtypes)
    print("\n--------------------------------------------------------------------------\n'
    print(" Dataset size (rows columns): {}".format(dataframe_name))
    print(dataframe.shape)
    print("\n--------------------------------------------------------------------------\n'
    print("Summary statistics: {}".format(dataframe_name))
    print(dataframe.describe())
    print("\n--------------------------------------------------------------------------\n'
    print("Correlation analysis: {}".format(dataframe_name))
    print(dataframe.corr())
    print("\n--------------------------------------------------------------------------\n'
    print("Other Analysis: {}".format(dataframe_name))
    print("1. Checking for Null values: {}".format(dataframe_name))
```

```
    print(dataframe.isna().sum())
    print("\n2. Info")
    print(dataframe.info())
```

In [11]:

```
Exploratory_Data_Analysis(application_train,'APPLICTION_TRAIN_DATA')
```

```
Test description; data type: APPLICTION_TRAIN_DATA
SK_ID_CURR                       int64
TARGET                           int64
NAME_CONTRACT_TYPE              object
CODE_GENDER                     object
FLAG_OWN_CAR                    object
                                 ...
AMT_REQ_CREDIT_BUREAU_DAY      float64
AMT_REQ_CREDIT_BUREAU_WEEK     float64
AMT_REQ_CREDIT_BUREAU_MON      float64
AMT_REQ_CREDIT_BUREAU_QRT      float64
AMT_REQ_CREDIT_BUREAU_YEAR     float64
Length: 122, dtype: object

---------------------------------------------------------------------------

 Dataset size (rows columns): APPLICTION_TRAIN_DATA
(307511, 122)

---------------------------------------------------------------------------

Summary statistics: APPLICTION_TRAIN_DATA
          SK_ID_CURR         TARGET   CNT_CHILDREN  AMT_INCOME_TOTAL  \
count  307511.000000  307511.000000  307511.000000      3.075110e+05   
mean   278180.518577       0.080729       0.417052      1.687979e+05   
std    102790.175348       0.272419       0.722121      2.371231e+05   
min    100002.000000       0.000000       0.000000      2.565000e+04   
25%    189145.500000       0.000000       0.000000      1.125000e+05   
50%    278202.000000       0.000000       0.000000      1.471500e+05   
75%    367142.500000       0.000000       1.000000      2.025000e+05   
max    456255.000000       1.000000      19.000000      1.170000e+08   

         AMT_CREDIT    AMT_ANNUITY  AMT_GOODS_PRICE  \
count  3.075110e+05  307499.000000     3.072330e+05   
mean   5.990260e+05   27108.573909     5.383962e+05   
std    4.024908e+05   14493.737315     3.694465e+05   
min    4.500000e+04    1615.500000     4.050000e+04   
25%    2.700000e+05   16524.000000     2.385000e+05   
50%    5.135310e+05   24903.000000     4.500000e+05   
75%    8.086500e+05   34596.000000     6.795000e+05   
max    4.050000e+06  258025.500000     4.050000e+06   

       REGION_POPULATION_RELATIVE     DAYS_BIRTH  DAYS_EMPLOYED  ...  \
count               307511.000000  307511.000000  307511.000000  ...   
mean                     0.020868  -16036.995067   63815.045904  ...   
std                      0.013831    4363.988632  141275.766519  ...   
min                      0.000290  -25229.000000  -17912.000000  ...   
25%                      0.010006  -19682.000000   -2760.000000  ...   
50%                      0.018850  -15750.000000   -1213.000000  ...   
75%                      0.028663  -12413.000000    -289.000000  ...   
max                      0.072508   -7489.000000  365243.000000  ...   

       FLAG_DOCUMENT_18  FLAG_DOCUMENT_19  FLAG_DOCUMENT_20  FLAG_DOCUMENT_21  \
count     307511.000000     307511.000000     307511.000000     307511.000000   
mean           0.008130          0.000595          0.000507          0.000335   
std            0.089798          0.024387          0.022518          0.018299   
min            0.000000          0.000000          0.000000          0.000000   
25%            0.000000          0.000000          0.000000          0.000000   
```

```
50%          0.000000         0.000000         0.000000         0.000000
75%          0.000000         0.000000         0.000000         0.000000
max          1.000000         1.000000         1.000000         1.000000

       AMT_REQ_CREDIT_BUREAU_HOUR  AMT_REQ_CREDIT_BUREAU_DAY  \
count               265992.000000              265992.000000
mean                     0.006402                   0.007000
std                      0.083849                   0.110757
min                      0.000000                   0.000000
25%                      0.000000                   0.000000
50%                      0.000000                   0.000000
75%                      0.000000                   0.000000
max                      4.000000                   9.000000

       AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
count               265992.000000              265992.000000
mean                     0.034362                   0.267395
std                      0.204685                   0.916002
min                      0.000000                   0.000000
25%                      0.000000                   0.000000
50%                      0.000000                   0.000000
75%                      0.000000                   0.000000
max                      8.000000                  27.000000

       AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
count              265992.000000               265992.000000
mean                    0.265474                    1.899974
std                     0.794056                    1.869295
min                     0.000000                    0.000000
25%                     0.000000                    0.000000
50%                     0.000000                    1.000000
75%                     0.000000                    3.000000
max                   261.000000                   25.000000

[8 rows x 106 columns]

---------------------------------------------------------------------------

Correlation analysis: APPLICTION_TRAIN_DATA
                            SK_ID_CURR    TARGET  CNT_CHILDREN  \
SK_ID_CURR                    1.000000 -0.002108     -0.001129
TARGET                       -0.002108  1.000000      0.019187
CNT_CHILDREN                 -0.001129  0.019187      1.000000
AMT_INCOME_TOTAL             -0.001820 -0.003982      0.012882
AMT_CREDIT                   -0.000343 -0.030369      0.002145
...                                ...       ...           ...
AMT_REQ_CREDIT_BUREAU_DAY    -0.002193  0.002704     -0.000366
AMT_REQ_CREDIT_BUREAU_WEEK    0.002099  0.000788     -0.002436
AMT_REQ_CREDIT_BUREAU_MON     0.000485 -0.012462     -0.010808
AMT_REQ_CREDIT_BUREAU_QRT     0.001025 -0.002022     -0.007836
AMT_REQ_CREDIT_BUREAU_YEAR    0.004659  0.019930     -0.041550

                            AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY  \
SK_ID_CURR                         -0.001820   -0.000343    -0.000433
TARGET                             -0.003982   -0.030369    -0.012817
CNT_CHILDREN                        0.012882    0.002145     0.021374
AMT_INCOME_TOTAL                    1.000000    0.156870     0.191657
AMT_CREDIT                          0.156870    1.000000     0.770138
...                                      ...         ...          ...
AMT_REQ_CREDIT_BUREAU_DAY           0.002944    0.004238     0.002185
AMT_REQ_CREDIT_BUREAU_WEEK          0.002387   -0.001275     0.013881
AMT_REQ_CREDIT_BUREAU_MON           0.024700    0.054451     0.039148
AMT_REQ_CREDIT_BUREAU_QRT           0.004859    0.015925     0.010124
AMT_REQ_CREDIT_BUREAU_YEAR          0.011690   -0.048448    -0.011320

                            AMT_GOODS_PRICE  REGION_POPULATION_RELATIVE  \
```

```
SK_ID_CURR                   -0.000232                    0.000849
TARGET                       -0.039645                   -0.037227
CNT_CHILDREN                 -0.001827                   -0.025573
AMT_INCOME_TOTAL              0.159610                    0.074796
AMT_CREDIT                    0.986968                    0.099738
...                                ...                         ...
AMT_REQ_CREDIT_BUREAU_DAY     0.004677                    0.001399
AMT_REQ_CREDIT_BUREAU_WEEK   -0.001007                   -0.002149
AMT_REQ_CREDIT_BUREAU_MON     0.056422                    0.078607
AMT_REQ_CREDIT_BUREAU_QRT     0.016432                   -0.001279
AMT_REQ_CREDIT_BUREAU_YEAR   -0.050998                    0.001003

                            DAYS_BIRTH  DAYS_EMPLOYED  ...  FLAG_DOCUMENT_18  \
SK_ID_CURR                   -0.001500       0.001366  ...          0.000509
TARGET                        0.078239      -0.044932  ...         -0.007952
CNT_CHILDREN                  0.330938      -0.239818  ...          0.004031
AMT_INCOME_TOTAL              0.027261      -0.064223  ...          0.003130
AMT_CREDIT                   -0.055436      -0.066838  ...          0.034329
...                                ...            ...  ...               ...
AMT_REQ_CREDIT_BUREAU_DAY     0.002255       0.000472  ...          0.013281
AMT_REQ_CREDIT_BUREAU_WEEK   -0.001336       0.003072  ...         -0.004640
AMT_REQ_CREDIT_BUREAU_MON     0.001372      -0.034457  ...         -0.001565
AMT_REQ_CREDIT_BUREAU_QRT    -0.011799       0.015345  ...         -0.005125
AMT_REQ_CREDIT_BUREAU_YEAR   -0.071983       0.049988  ...         -0.047432

                            FLAG_DOCUMENT_19  FLAG_DOCUMENT_20  \
SK_ID_CURR                          0.000167          0.001073
TARGET                             -0.001358          0.000215
CNT_CHILDREN                        0.000864          0.000988
AMT_INCOME_TOTAL                    0.002408          0.000242
AMT_CREDIT                          0.021082          0.031023
...                                      ...               ...
AMT_REQ_CREDIT_BUREAU_DAY           0.001126         -0.000120
AMT_REQ_CREDIT_BUREAU_WEEK         -0.001275         -0.001770
AMT_REQ_CREDIT_BUREAU_MON          -0.002729          0.001285
AMT_REQ_CREDIT_BUREAU_QRT          -0.001575         -0.001010
AMT_REQ_CREDIT_BUREAU_YEAR         -0.007009         -0.012126

                            FLAG_DOCUMENT_21  AMT_REQ_CREDIT_BUREAU_HOUR  \
SK_ID_CURR                          0.000282                   -0.002672
TARGET                              0.003709                    0.000930
CNT_CHILDREN                       -0.002450                   -0.000410
AMT_INCOME_TOTAL                   -0.000589                    0.000709
AMT_CREDIT                         -0.016148                   -0.003906
...                                      ...                         ...
AMT_REQ_CREDIT_BUREAU_DAY          -0.001130                    0.230374
AMT_REQ_CREDIT_BUREAU_WEEK          0.000081                    0.004706
AMT_REQ_CREDIT_BUREAU_MON          -0.003612                   -0.000018
AMT_REQ_CREDIT_BUREAU_QRT          -0.002004                   -0.002716
AMT_REQ_CREDIT_BUREAU_YEAR         -0.005457                   -0.004597

                            AMT_REQ_CREDIT_BUREAU_DAY  \
SK_ID_CURR                                  -0.002193
TARGET                                       0.002704
CNT_CHILDREN                                -0.000366
AMT_INCOME_TOTAL                             0.002944
AMT_CREDIT                                   0.004238
...                                               ...
AMT_REQ_CREDIT_BUREAU_DAY                    1.000000
AMT_REQ_CREDIT_BUREAU_WEEK                   0.217412
AMT_REQ_CREDIT_BUREAU_MON                   -0.005258
AMT_REQ_CREDIT_BUREAU_QRT                   -0.004416
AMT_REQ_CREDIT_BUREAU_YEAR                  -0.003355

                            AMT_REQ_CREDIT_BUREAU_WEEK  \
SK_ID_CURR                                    0.002099
```

```
TARGET                                                  0.000788
CNT_CHILDREN                                           -0.002436
AMT_INCOME_TOTAL                                        0.002387
AMT_CREDIT                                             -0.001275
...                                                          ...
AMT_REQ_CREDIT_BUREAU_DAY                               0.217412
AMT_REQ_CREDIT_BUREAU_WEEK                              1.000000
AMT_REQ_CREDIT_BUREAU_MON                              -0.014096
AMT_REQ_CREDIT_BUREAU_QRT                              -0.015115
AMT_REQ_CREDIT_BUREAU_YEAR                              0.018917

                            AMT_REQ_CREDIT_BUREAU_MON  \
SK_ID_CURR                                      0.000485
TARGET                                         -0.012462
CNT_CHILDREN                                   -0.010808
AMT_INCOME_TOTAL                                0.024700
AMT_CREDIT                                      0.054451
...                                                  ...
AMT_REQ_CREDIT_BUREAU_DAY                      -0.005258
AMT_REQ_CREDIT_BUREAU_WEEK                     -0.014096
AMT_REQ_CREDIT_BUREAU_MON                       1.000000
AMT_REQ_CREDIT_BUREAU_QRT                      -0.007789
AMT_REQ_CREDIT_BUREAU_YEAR                     -0.004975

                            AMT_REQ_CREDIT_BUREAU_QRT  \
SK_ID_CURR                                      0.001025
TARGET                                         -0.002022
CNT_CHILDREN                                   -0.007836
AMT_INCOME_TOTAL                                0.004859
AMT_CREDIT                                      0.015925
...                                                  ...
AMT_REQ_CREDIT_BUREAU_DAY                      -0.004416
AMT_REQ_CREDIT_BUREAU_WEEK                     -0.015115
AMT_REQ_CREDIT_BUREAU_MON                      -0.007789
AMT_REQ_CREDIT_BUREAU_QRT                       1.000000
AMT_REQ_CREDIT_BUREAU_YEAR                      0.076208

                            AMT_REQ_CREDIT_BUREAU_YEAR
SK_ID_CURR                                      0.004659
TARGET                                          0.019930
CNT_CHILDREN                                   -0.041550
AMT_INCOME_TOTAL                                0.011690
AMT_CREDIT                                     -0.048448
...                                                  ...
AMT_REQ_CREDIT_BUREAU_DAY                      -0.003355
AMT_REQ_CREDIT_BUREAU_WEEK                      0.018917
AMT_REQ_CREDIT_BUREAU_MON                      -0.004975
AMT_REQ_CREDIT_BUREAU_QRT                       0.076208
AMT_REQ_CREDIT_BUREAU_YEAR                      1.000000

[106 rows x 106 columns]

---------------------------------------------------------------------------

Other Analysis: APPLICTION_TRAIN_DATA
1. Checking for Null values: APPLICTION_TRAIN_DATA
SK_ID_CURR                        0
TARGET                            0
NAME_CONTRACT_TYPE                0
CODE_GENDER                       0
FLAG_OWN_CAR                      0
                              ...
AMT_REQ_CREDIT_BUREAU_DAY     41519
AMT_REQ_CREDIT_BUREAU_WEEK    41519
AMT_REQ_CREDIT_BUREAU_MON     41519
AMT_REQ_CREDIT_BUREAU_QRT     41519
```

```
AMT_REQ_CREDIT_BUREAU_YEAR    41519
Length: 122, dtype: int64

2. Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
None
```

In [12]:

```
bureau = datasets['bureau'].copy()
Exploratory_Data_Analysis(bureau,'Bureau_Data')
```

```
Test description; data type: Bureau_Data
SK_ID_CURR                  int64
SK_ID_BUREAU                int64
CREDIT_ACTIVE              object
CREDIT_CURRENCY            object
DAYS_CREDIT                 int64
CREDIT_DAY_OVERDUE          int64
DAYS_CREDIT_ENDDATE       float64
DAYS_ENDDATE_FACT         float64
AMT_CREDIT_MAX_OVERDUE    float64
CNT_CREDIT_PROLONG          int64
AMT_CREDIT_SUM            float64
AMT_CREDIT_SUM_DEBT       float64
AMT_CREDIT_SUM_LIMIT      float64
AMT_CREDIT_SUM_OVERDUE    float64
CREDIT_TYPE                object
DAYS_CREDIT_UPDATE          int64
AMT_ANNUITY               float64
dtype: object

---------------------------------------------------------------------------

 Dataset size (rows columns): Bureau_Data
(1716428, 17)

---------------------------------------------------------------------------

Summary statistics: Bureau_Data
         SK_ID_CURR  SK_ID_BUREAU   DAYS_CREDIT  CREDIT_DAY_OVERDUE  \
count  1.716428e+06  1.716428e+06  1.716428e+06        1.716428e+06
mean   2.782149e+05  5.924434e+06 -1.142108e+03        8.181666e-01
std    1.029386e+05  5.322657e+05  7.951649e+02        3.654443e+01
min    1.000010e+05  5.000000e+06 -2.922000e+03        0.000000e+00
25%    1.888668e+05  5.463954e+06 -1.666000e+03        0.000000e+00
50%    2.780550e+05  5.926304e+06 -9.870000e+02        0.000000e+00
75%    3.674260e+05  6.385681e+06 -4.740000e+02        0.000000e+00
max    4.562550e+05  6.843457e+06  0.000000e+00        2.792000e+03

       DAYS_CREDIT_ENDDATE  DAYS_ENDDATE_FACT  AMT_CREDIT_MAX_OVERDUE  \
count         1.610875e+06       1.082775e+06            5.919400e+05
mean          5.105174e+02      -1.017437e+03            3.825418e+03
std           4.994220e+03       7.140106e+02            2.060316e+05
min          -4.206000e+04      -4.202300e+04            0.000000e+00
25%          -1.138000e+03      -1.489000e+03            0.000000e+00
50%          -3.300000e+02      -8.970000e+02            0.000000e+00
75%           4.740000e+02      -4.250000e+02            0.000000e+00
max           3.119900e+04       0.000000e+00            1.159872e+08

       CNT_CREDIT_PROLONG  AMT_CREDIT_SUM  AMT_CREDIT_SUM_DEBT  \
count        1.716428e+06    1.716415e+06         1.458759e+06
mean         6.410406e-03    3.549946e+05         1.370851e+05
```

```
std        9.622391e-02   1.149811e+06        6.774011e+05
min        0.000000e+00   0.000000e+00       -4.705600e+06
25%        0.000000e+00   5.130000e+04        0.000000e+00
50%        0.000000e+00   1.255185e+05        0.000000e+00
75%        0.000000e+00   3.150000e+05        4.015350e+04
max        9.000000e+00   5.850000e+08        1.701000e+08

       AMT_CREDIT_SUM_LIMIT  AMT_CREDIT_SUM_OVERDUE  DAYS_CREDIT_UPDATE  \
count          1.124648e+06            1.716428e+06        1.716428e+06
mean           6.229515e+03            3.791276e+01       -5.937483e+02
std            4.503203e+04            5.937650e+03        7.207473e+02
min           -5.864061e+05            0.000000e+00       -4.194700e+04
25%            0.000000e+00            0.000000e+00       -9.080000e+02
50%            0.000000e+00            0.000000e+00       -3.950000e+02
75%            0.000000e+00            0.000000e+00       -3.300000e+01
max            4.705600e+06            3.756681e+06        3.720000e+02

        AMT_ANNUITY
count  4.896370e+05
mean   1.571276e+04
std    3.258269e+05
min    0.000000e+00
25%    0.000000e+00
50%    0.000000e+00
75%    1.350000e+04
max    1.184534e+08

--------------------------------------------------------------------------

Correlation analysis: Bureau_Data
                        SK_ID_CURR  SK_ID_BUREAU  DAYS_CREDIT  \
SK_ID_CURR                1.000000      0.000135     0.000266
SK_ID_BUREAU              0.000135      1.000000     0.013015
DAYS_CREDIT               0.000266      0.013015     1.000000
CREDIT_DAY_OVERDUE        0.000283     -0.002628    -0.027266
DAYS_CREDIT_ENDDATE       0.000456      0.009107     0.225682
DAYS_ENDDATE_FACT        -0.000648      0.017890     0.875359
AMT_CREDIT_MAX_OVERDUE    0.001329      0.002290    -0.014724
CNT_CREDIT_PROLONG       -0.000388     -0.000740    -0.030460
AMT_CREDIT_SUM            0.001179      0.007962     0.050883
AMT_CREDIT_SUM_DEBT      -0.000790      0.005732     0.135397
AMT_CREDIT_SUM_LIMIT     -0.000304     -0.003986     0.025140
AMT_CREDIT_SUM_OVERDUE   -0.000014     -0.000499    -0.000383
DAYS_CREDIT_UPDATE        0.000510      0.019398     0.688771
AMT_ANNUITY              -0.002727      0.001799     0.005676

                        CREDIT_DAY_OVERDUE  DAYS_CREDIT_ENDDATE  \
SK_ID_CURR                        0.000283             0.000456
SK_ID_BUREAU                     -0.002628             0.009107
DAYS_CREDIT                      -0.027266             0.225682
CREDIT_DAY_OVERDUE                1.000000            -0.007352
DAYS_CREDIT_ENDDATE              -0.007352             1.000000
DAYS_ENDDATE_FACT                -0.008637             0.248825
AMT_CREDIT_MAX_OVERDUE            0.001249             0.000577
CNT_CREDIT_PROLONG                0.002756             0.113683
AMT_CREDIT_SUM                   -0.003292             0.055424
AMT_CREDIT_SUM_DEBT              -0.002355             0.081298
AMT_CREDIT_SUM_LIMIT             -0.000345             0.095421
AMT_CREDIT_SUM_OVERDUE            0.090951             0.001077
DAYS_CREDIT_UPDATE               -0.018461             0.248525
AMT_ANNUITY                      -0.000339             0.000475

                        DAYS_ENDDATE_FACT  AMT_CREDIT_MAX_OVERDUE  \
SK_ID_CURR                      -0.000648                0.001329
SK_ID_BUREAU                     0.017890                0.002290
DAYS_CREDIT                      0.875359               -0.014724
```

```
CREDIT_DAY_OVERDUE          -0.008637              0.001249
DAYS_CREDIT_ENDDATE          0.248825              0.000577
DAYS_ENDDATE_FACT            1.000000              0.000999
AMT_CREDIT_MAX_OVERDUE       0.000999              1.000000
CNT_CREDIT_PROLONG           0.012017              0.001523
AMT_CREDIT_SUM               0.059096              0.081663
AMT_CREDIT_SUM_DEBT          0.019609              0.014007
AMT_CREDIT_SUM_LIMIT         0.019476             -0.000112
AMT_CREDIT_SUM_OVERDUE      -0.000332              0.015036
DAYS_CREDIT_UPDATE           0.751294             -0.000749
AMT_ANNUITY                  0.006274              0.001578

                        CNT_CREDIT_PROLONG  AMT_CREDIT_SUM  \
SK_ID_CURR                       -0.000388        0.001179
SK_ID_BUREAU                     -0.000740        0.007962
DAYS_CREDIT                      -0.030460        0.050883
CREDIT_DAY_OVERDUE                0.002756       -0.003292
DAYS_CREDIT_ENDDATE               0.113683        0.055424
DAYS_ENDDATE_FACT                 0.012017        0.059096
AMT_CREDIT_MAX_OVERDUE            0.001523        0.081663
CNT_CREDIT_PROLONG                1.000000       -0.008345
AMT_CREDIT_SUM                   -0.008345        1.000000
AMT_CREDIT_SUM_DEBT              -0.001366        0.683419
AMT_CREDIT_SUM_LIMIT              0.073805        0.003756
AMT_CREDIT_SUM_OVERDUE            0.000002        0.006342
DAYS_CREDIT_UPDATE                0.017864        0.104629
AMT_ANNUITY                      -0.000465        0.049146

                        AMT_CREDIT_SUM_DEBT  AMT_CREDIT_SUM_LIMIT  \
SK_ID_CURR                        -0.000790             -0.000304
SK_ID_BUREAU                       0.005732             -0.003986
DAYS_CREDIT                        0.135397              0.025140
CREDIT_DAY_OVERDUE                -0.002355             -0.000345
DAYS_CREDIT_ENDDATE                0.081298              0.095421
DAYS_ENDDATE_FACT                  0.019609              0.019476
AMT_CREDIT_MAX_OVERDUE             0.014007             -0.000112
CNT_CREDIT_PROLONG                -0.001366              0.073805
AMT_CREDIT_SUM                     0.683419              0.003756
AMT_CREDIT_SUM_DEBT                1.000000             -0.018215
AMT_CREDIT_SUM_LIMIT              -0.018215              1.000000
AMT_CREDIT_SUM_OVERDUE             0.008046             -0.000687
DAYS_CREDIT_UPDATE                 0.141235              0.046028
AMT_ANNUITY                        0.025507              0.004392

                        AMT_CREDIT_SUM_OVERDUE  DAYS_CREDIT_UPDATE  \
SK_ID_CURR                           -0.000014            0.000510
SK_ID_BUREAU                         -0.000499            0.019398
DAYS_CREDIT                          -0.000383            0.688771
CREDIT_DAY_OVERDUE                    0.090951           -0.018461
DAYS_CREDIT_ENDDATE                   0.001077            0.248525
DAYS_ENDDATE_FACT                    -0.000332            0.751294
AMT_CREDIT_MAX_OVERDUE                0.015036           -0.000749
CNT_CREDIT_PROLONG                    0.000002            0.017864
AMT_CREDIT_SUM                        0.006342            0.104629
AMT_CREDIT_SUM_DEBT                   0.008046            0.141235
AMT_CREDIT_SUM_LIMIT                 -0.000687            0.046028
AMT_CREDIT_SUM_OVERDUE                1.000000            0.003528
DAYS_CREDIT_UPDATE                    0.003528            1.000000
AMT_ANNUITY                           0.000344            0.008418

                        AMT_ANNUITY
SK_ID_CURR                -0.002727
SK_ID_BUREAU               0.001799
DAYS_CREDIT                0.005676
CREDIT_DAY_OVERDUE        -0.000339
DAYS_CREDIT_ENDDATE        0.000475
```

```
DAYS_ENDDATE_FACT          0.006274
AMT_CREDIT_MAX_OVERDUE     0.001578
CNT_CREDIT_PROLONG        -0.000465
AMT_CREDIT_SUM             0.049146
AMT_CREDIT_SUM_DEBT        0.025507
AMT_CREDIT_SUM_LIMIT       0.004392
AMT_CREDIT_SUM_OVERDUE     0.000344
DAYS_CREDIT_UPDATE         0.008418
AMT_ANNUITY                1.000000

---------------------------------------------------------------------------

Other Analysis: Bureau_Data
1. Checking for Null values: Bureau_Data
SK_ID_CURR                      0
SK_ID_BUREAU                    0
CREDIT_ACTIVE                   0
CREDIT_CURRENCY                 0
DAYS_CREDIT                     0
CREDIT_DAY_OVERDUE              0
DAYS_CREDIT_ENDDATE        105553
DAYS_ENDDATE_FACT          633653
AMT_CREDIT_MAX_OVERDUE    1124488
CNT_CREDIT_PROLONG              0
AMT_CREDIT_SUM                 13
AMT_CREDIT_SUM_DEBT        257669
AMT_CREDIT_SUM_LIMIT       591780
AMT_CREDIT_SUM_OVERDUE          0
CREDIT_TYPE                     0
DAYS_CREDIT_UPDATE              0
AMT_ANNUITY               1226791
dtype: int64

2. Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1716428 entries, 0 to 1716427
Data columns (total 17 columns):
 #   Column                  Dtype  
---  ------                  -----  
 0   SK_ID_CURR              int64  
 1   SK_ID_BUREAU            int64  
 2   CREDIT_ACTIVE           object 
 3   CREDIT_CURRENCY         object 
 4   DAYS_CREDIT             int64  
 5   CREDIT_DAY_OVERDUE      int64  
 6   DAYS_CREDIT_ENDDATE     float64
 7   DAYS_ENDDATE_FACT       float64
 8   AMT_CREDIT_MAX_OVERDUE  float64
 9   CNT_CREDIT_PROLONG      int64  
 10  AMT_CREDIT_SUM          float64
 11  AMT_CREDIT_SUM_DEBT     float64
 12  AMT_CREDIT_SUM_LIMIT    float64
 13  AMT_CREDIT_SUM_OVERDUE  float64
 14  CREDIT_TYPE             object 
 15  DAYS_CREDIT_UPDATE      int64  
 16  AMT_ANNUITY             float64
dtypes: float64(8), int64(6), object(3)
memory usage: 222.6+ MB
None
```

In [13]:

```
bureau_balance = datasets['bureau_balance'].copy()
Exploratory_Data_Analysis(bureau_balance,'Bureau_balance_Data')
```

```
Test description; data type: Bureau_balance_Data
SK_ID_BUREAU       int64
```

```
MONTHS_BALANCE      int64
STATUS             object
dtype: object

---------------------------------------------------------------------------

 Dataset size (rows columns): Bureau_balance_Data
(27299925, 3)

---------------------------------------------------------------------------

Summary statistics: Bureau_balance_Data
       SK_ID_BUREAU  MONTHS_BALANCE
count  2.729992e+07    2.729992e+07
mean   6.036297e+06   -3.074169e+01
std    4.923489e+05    2.386451e+01
min    5.001709e+06   -9.600000e+01
25%    5.730933e+06   -4.600000e+01
50%    6.070821e+06   -2.500000e+01
75%    6.431951e+06   -1.100000e+01
max    6.842888e+06    0.000000e+00

---------------------------------------------------------------------------

Correlation analysis: Bureau_balance_Data
                SK_ID_BUREAU  MONTHS_BALANCE
SK_ID_BUREAU        1.000000        0.011873
MONTHS_BALANCE      0.011873        1.000000

---------------------------------------------------------------------------

Other Analysis: Bureau_balance_Data
1. Checking for Null values: Bureau_balance_Data
SK_ID_BUREAU      0
MONTHS_BALANCE    0
STATUS            0
dtype: int64

2. Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27299925 entries, 0 to 27299924
Data columns (total 3 columns):
 #   Column          Dtype 
---  ------          ----- 
 0   SK_ID_BUREAU    int64 
 1   MONTHS_BALANCE  int64 
 2   STATUS          object
dtypes: int64(2), object(1)
memory usage: 624.8+ MB
None
```

In [14]:

```
credit_card_balance = datasets['credit_card_balance'].copy()
Exploratory_Data_Analysis(credit_card_balance,'credit_card_balance')
```

```
Test description; data type: credit_card_balance
SK_ID_PREV                      int64
SK_ID_CURR                      int64
MONTHS_BALANCE                  int64
AMT_BALANCE                   float64
AMT_CREDIT_LIMIT_ACTUAL         int64
AMT_DRAWINGS_ATM_CURRENT      float64
AMT_DRAWINGS_CURRENT          float64
AMT_DRAWINGS_OTHER_CURRENT    float64
AMT_DRAWINGS_POS_CURRENT      float64
AMT_INST_MIN_REGULARITY       float64
```

```
AMT_PAYMENT_CURRENT           float64
AMT_PAYMENT_TOTAL_CURRENT     float64
AMT_RECEIVABLE_PRINCIPAL      float64
AMT_RECIVABLE                 float64
AMT_TOTAL_RECEIVABLE          float64
CNT_DRAWINGS_ATM_CURRENT      float64
CNT_DRAWINGS_CURRENT            int64
CNT_DRAWINGS_OTHER_CURRENT    float64
CNT_DRAWINGS_POS_CURRENT      float64
CNT_INSTALMENT_MATURE_CUM     float64
NAME_CONTRACT_STATUS           object
SK_DPD                          int64
SK_DPD_DEF                      int64
dtype: object

---------------------------------------------------------------------------

 Dataset size (rows columns): credit_card_balance
(3840312, 23)

---------------------------------------------------------------------------

Summary statistics: credit_card_balance
         SK_ID_PREV    SK_ID_CURR  MONTHS_BALANCE   AMT_BALANCE  \
count  3.840312e+06  3.840312e+06    3.840312e+06  3.840312e+06
mean   1.904504e+06  2.783242e+05   -3.452192e+01  5.830016e+04
std    5.364695e+05  1.027045e+05    2.666775e+01  1.063070e+05
min    1.000018e+06  1.000060e+05   -9.600000e+01 -4.202502e+05
25%    1.434385e+06  1.895170e+05   -5.500000e+01  0.000000e+00
50%    1.897122e+06  2.783960e+05   -2.800000e+01  0.000000e+00
75%    2.369328e+06  3.675800e+05   -1.100000e+01  8.904669e+04
max    2.843496e+06  4.562500e+05   -1.000000e+00  1.505902e+06

       AMT_CREDIT_LIMIT_ACTUAL  AMT_DRAWINGS_ATM_CURRENT  \
count             3.840312e+06              3.090496e+06
mean              1.538080e+05              5.961325e+03
std               1.651457e+05              2.822569e+04
min               0.000000e+00             -6.827310e+03
25%               4.500000e+04              0.000000e+00
50%               1.125000e+05              0.000000e+00
75%               1.800000e+05              0.000000e+00
max               1.350000e+06              2.115000e+06

       AMT_DRAWINGS_CURRENT  AMT_DRAWINGS_OTHER_CURRENT  \
count          3.840312e+06                3.090496e+06
mean           7.433388e+03                2.881696e+02
std            3.384608e+04                8.201989e+03
min           -6.211620e+03                0.000000e+00
25%            0.000000e+00                0.000000e+00
50%            0.000000e+00                0.000000e+00
75%            0.000000e+00                0.000000e+00
max            2.287098e+06                1.529847e+06

       AMT_DRAWINGS_POS_CURRENT  AMT_INST_MIN_REGULARITY  ...  \
count              3.090496e+06             3.535076e+06  ...
mean               2.968805e+03             3.540204e+03  ...
std                2.079689e+04             5.600154e+03  ...
min                0.000000e+00             0.000000e+00  ...
25%                0.000000e+00             0.000000e+00  ...
50%                0.000000e+00             0.000000e+00  ...
75%                0.000000e+00             6.633911e+03  ...
max                2.239274e+06             2.028820e+05  ...

       AMT_RECEIVABLE_PRINCIPAL  AMT_RECIVABLE  AMT_TOTAL_RECEIVABLE  \
count              3.840312e+06   3.840312e+06          3.840312e+06
mean               5.596588e+04   5.808881e+04          5.809829e+04
```

```
std             1.025336e+05  1.059654e+05         1.059718e+05
min            -4.233058e+05 -4.202502e+05        -4.202502e+05
25%             0.000000e+00  0.000000e+00         0.000000e+00
50%             0.000000e+00  0.000000e+00         0.000000e+00
75%             8.535924e+04  8.889949e+04         8.891451e+04
max             1.472317e+06  1.493338e+06         1.493338e+06

       CNT_DRAWINGS_ATM_CURRENT  CNT_DRAWINGS_CURRENT  \
count              3.090496e+06          3.840312e+06
mean               3.094490e-01          7.031439e-01
std                1.100401e+00          3.190347e+00
min                0.000000e+00          0.000000e+00
25%                0.000000e+00          0.000000e+00
50%                0.000000e+00          0.000000e+00
75%                0.000000e+00          0.000000e+00
max                5.100000e+01          1.650000e+02

       CNT_DRAWINGS_OTHER_CURRENT  CNT_DRAWINGS_POS_CURRENT  \
count                3.090496e+06              3.090496e+06
mean                 4.812496e-03              5.594791e-01
std                  8.263861e-02              3.240649e+00
min                  0.000000e+00              0.000000e+00
25%                  0.000000e+00              0.000000e+00
50%                  0.000000e+00              0.000000e+00
75%                  0.000000e+00              0.000000e+00
max                  1.200000e+01              1.650000e+02

       CNT_INSTALMENT_MATURE_CUM        SK_DPD    SK_DPD_DEF
count               3.535076e+06  3.840312e+06  3.840312e+06
mean                2.082508e+01  9.283667e+00  3.316220e-01
std                 2.005149e+01  9.751570e+01  2.147923e+01
min                 0.000000e+00  0.000000e+00  0.000000e+00
25%                 4.000000e+00  0.000000e+00  0.000000e+00
50%                 1.500000e+01  0.000000e+00  0.000000e+00
75%                 3.200000e+01  0.000000e+00  0.000000e+00
max                 1.200000e+02  3.260000e+03  3.260000e+03

[8 rows x 22 columns]

--------------------------------------------------------------------------

Correlation analysis: credit_card_balance
                            SK_ID_PREV  SK_ID_CURR  MONTHS_BALANCE  \
SK_ID_PREV                    1.000000    0.004723        0.003670
SK_ID_CURR                    0.004723    1.000000        0.001696
MONTHS_BALANCE                0.003670    0.001696        1.000000
AMT_BALANCE                   0.005046    0.003510        0.014558
AMT_CREDIT_LIMIT_ACTUAL       0.006631    0.005991        0.199900
AMT_DRAWINGS_ATM_CURRENT      0.004342    0.000814        0.036802
AMT_DRAWINGS_CURRENT          0.002624    0.000708        0.065527
AMT_DRAWINGS_OTHER_CURRENT   -0.000160    0.000958        0.000405
AMT_DRAWINGS_POS_CURRENT      0.001721   -0.000786        0.118146
AMT_INST_MIN_REGULARITY       0.006460    0.003300       -0.087529
AMT_PAYMENT_CURRENT           0.003472    0.000127        0.076355
AMT_PAYMENT_TOTAL_CURRENT     0.001641    0.000784        0.035614
AMT_RECEIVABLE_PRINCIPAL      0.005140    0.003589        0.016266
AMT_RECIVABLE                 0.005035    0.003518        0.013172
AMT_TOTAL_RECEIVABLE          0.005032    0.003524        0.013084
CNT_DRAWINGS_ATM_CURRENT      0.002821    0.002082        0.002536
CNT_DRAWINGS_CURRENT          0.000367    0.002654        0.113321
CNT_DRAWINGS_OTHER_CURRENT   -0.001412   -0.000131       -0.026192
CNT_DRAWINGS_POS_CURRENT      0.000809    0.002135        0.160207
CNT_INSTALMENT_MATURE_CUM    -0.007219   -0.000581       -0.008620
SK_DPD                       -0.001786   -0.000962        0.039434
SK_DPD_DEF                    0.001973    0.001519        0.001659
```

	AMT_BALANCE	AMT_CREDIT_LIMIT_ACTUAL \
SK_ID_PREV	0.005046	0.006631
SK_ID_CURR	0.003510	0.005991
MONTHS_BALANCE	0.014558	0.199900
AMT_BALANCE	1.000000	0.489386
AMT_CREDIT_LIMIT_ACTUAL	0.489386	1.000000
AMT_DRAWINGS_ATM_CURRENT	0.283551	0.247219
AMT_DRAWINGS_CURRENT	0.336965	0.263093
AMT_DRAWINGS_OTHER_CURRENT	0.065366	0.050579
AMT_DRAWINGS_POS_CURRENT	0.169449	0.234976
AMT_INST_MIN_REGULARITY	0.896728	0.467620
AMT_PAYMENT_CURRENT	0.143934	0.308294
AMT_PAYMENT_TOTAL_CURRENT	0.151349	0.226570
AMT_RECEIVABLE_PRINCIPAL	0.999720	0.490445
AMT_RECIVABLE	0.999917	0.488641
AMT_TOTAL_RECEIVABLE	0.999897	0.488598
CNT_DRAWINGS_ATM_CURRENT	0.309968	0.221808
CNT_DRAWINGS_CURRENT	0.259184	0.204237
CNT_DRAWINGS_OTHER_CURRENT	0.046563	0.030051
CNT_DRAWINGS_POS_CURRENT	0.155553	0.202868
CNT_INSTALMENT_MATURE_CUM	0.005009	-0.157269
SK_DPD	-0.046988	-0.038791
SK_DPD_DEF	0.013009	-0.002236

	AMT_DRAWINGS_ATM_CURRENT	AMT_DRAWINGS_CURRENT \
SK_ID_PREV	0.004342	0.002624
SK_ID_CURR	0.000814	0.000708
MONTHS_BALANCE	0.036802	0.065527
AMT_BALANCE	0.283551	0.336965
AMT_CREDIT_LIMIT_ACTUAL	0.247219	0.263093
AMT_DRAWINGS_ATM_CURRENT	1.000000	0.800190
AMT_DRAWINGS_CURRENT	0.800190	1.000000
AMT_DRAWINGS_OTHER_CURRENT	0.017899	0.236297
AMT_DRAWINGS_POS_CURRENT	0.078971	0.615591
AMT_INST_MIN_REGULARITY	0.094824	0.124469
AMT_PAYMENT_CURRENT	0.189075	0.337343
AMT_PAYMENT_TOTAL_CURRENT	0.159186	0.305726
AMT_RECEIVABLE_PRINCIPAL	0.280402	0.337117
AMT_RECIVABLE	0.278290	0.332831
AMT_TOTAL_RECEIVABLE	0.278260	0.332796
CNT_DRAWINGS_ATM_CURRENT	0.732907	0.594361
CNT_DRAWINGS_CURRENT	0.298173	0.523016
CNT_DRAWINGS_OTHER_CURRENT	0.013254	0.140032
CNT_DRAWINGS_POS_CURRENT	0.076083	0.359001
CNT_INSTALMENT_MATURE_CUM	-0.103721	-0.093491
SK_DPD	-0.022044	-0.020606
SK_DPD_DEF	-0.003360	-0.003137

	AMT_DRAWINGS_OTHER_CURRENT \
SK_ID_PREV	-0.000160
SK_ID_CURR	0.000958
MONTHS_BALANCE	0.000405
AMT_BALANCE	0.065366
AMT_CREDIT_LIMIT_ACTUAL	0.050579
AMT_DRAWINGS_ATM_CURRENT	0.017899
AMT_DRAWINGS_CURRENT	0.236297
AMT_DRAWINGS_OTHER_CURRENT	1.000000
AMT_DRAWINGS_POS_CURRENT	0.007382
AMT_INST_MIN_REGULARITY	0.002158
AMT_PAYMENT_CURRENT	0.034577
AMT_PAYMENT_TOTAL_CURRENT	0.025123
AMT_RECEIVABLE_PRINCIPAL	0.066108
AMT_RECIVABLE	0.064929
AMT_TOTAL_RECEIVABLE	0.064923
CNT_DRAWINGS_ATM_CURRENT	0.012008
CNT_DRAWINGS_CURRENT	0.021271

```
CNT_DRAWINGS_OTHER_CURRENT                   0.575295
CNT_DRAWINGS_POS_CURRENT                     0.004458
CNT_INSTALMENT_MATURE_CUM                   -0.023013
SK_DPD                                      -0.003693
SK_DPD_DEF                                  -0.000568

                            AMT_DRAWINGS_POS_CURRENT  AMT_INST_MIN_REGULARITY  \
SK_ID_PREV                                  0.001721                 0.006460
SK_ID_CURR                                 -0.000786                 0.003300
MONTHS_BALANCE                              0.118146                -0.087529
AMT_BALANCE                                 0.169449                 0.896728
AMT_CREDIT_LIMIT_ACTUAL                     0.234976                 0.467620
AMT_DRAWINGS_ATM_CURRENT                    0.078971                 0.094824
AMT_DRAWINGS_CURRENT                        0.615591                 0.124469
AMT_DRAWINGS_OTHER_CURRENT                  0.007382                 0.002158
AMT_DRAWINGS_POS_CURRENT                    1.000000                 0.063562
AMT_INST_MIN_REGULARITY                     0.063562                 1.000000
AMT_PAYMENT_CURRENT                         0.321055                 0.333909
AMT_PAYMENT_TOTAL_CURRENT                   0.301760                 0.335201
AMT_RECEIVABLE_PRINCIPAL                    0.173745                 0.896030
AMT_RECIVABLE                               0.168974                 0.897617
AMT_TOTAL_RECEIVABLE                        0.168950                 0.897587
CNT_DRAWINGS_ATM_CURRENT                    0.072658                 0.170616
CNT_DRAWINGS_CURRENT                        0.520123                 0.148262
CNT_DRAWINGS_OTHER_CURRENT                  0.007620                 0.014360
CNT_DRAWINGS_POS_CURRENT                    0.542556                 0.086729
CNT_INSTALMENT_MATURE_CUM                  -0.106813                 0.064320
SK_DPD                                     -0.015040                -0.061484
SK_DPD_DEF                                 -0.002384                -0.005715

                            ...  AMT_RECEIVABLE_PRINCIPAL  AMT_RECIVABLE  \
SK_ID_PREV                  ...                  0.005140       0.005035
SK_ID_CURR                  ...                  0.003589       0.003518
MONTHS_BALANCE              ...                  0.016266       0.013172
AMT_BALANCE                 ...                  0.999720       0.999917
AMT_CREDIT_LIMIT_ACTUAL     ...                  0.490445       0.488641
AMT_DRAWINGS_ATM_CURRENT    ...                  0.280402       0.278290
AMT_DRAWINGS_CURRENT        ...                  0.337117       0.332831
AMT_DRAWINGS_OTHER_CURRENT  ...                  0.066108       0.064929
AMT_DRAWINGS_POS_CURRENT    ...                  0.173745       0.168974
AMT_INST_MIN_REGULARITY     ...                  0.896030       0.897617
AMT_PAYMENT_CURRENT         ...                  0.143162       0.142389
AMT_PAYMENT_TOTAL_CURRENT   ...                  0.149936       0.149926
AMT_RECEIVABLE_PRINCIPAL    ...                  1.000000       0.999727
AMT_RECIVABLE               ...                  0.999727       1.000000
AMT_TOTAL_RECEIVABLE        ...                  0.999702       0.999995
CNT_DRAWINGS_ATM_CURRENT    ...                  0.302627       0.303571
CNT_DRAWINGS_CURRENT        ...                  0.258848       0.256347
CNT_DRAWINGS_OTHER_CURRENT  ...                  0.046543       0.046118
CNT_DRAWINGS_POS_CURRENT    ...                  0.157723       0.154507
CNT_INSTALMENT_MATURE_CUM   ...                  0.003664       0.005935
SK_DPD                      ...                 -0.048290      -0.046434
SK_DPD_DEF                  ...                  0.006780       0.015466

                            AMT_TOTAL_RECEIVABLE  CNT_DRAWINGS_ATM_CURRENT  \
SK_ID_PREV                              0.005032                  0.002821
SK_ID_CURR                              0.003524                  0.002082
MONTHS_BALANCE                          0.013084                  0.002536
AMT_BALANCE                             0.999897                  0.309968
AMT_CREDIT_LIMIT_ACTUAL                 0.488598                  0.221808
AMT_DRAWINGS_ATM_CURRENT                0.278260                  0.732907
AMT_DRAWINGS_CURRENT                    0.332796                  0.594361
AMT_DRAWINGS_OTHER_CURRENT              0.064923                  0.012008
AMT_DRAWINGS_POS_CURRENT                0.168950                  0.072658
AMT_INST_MIN_REGULARITY                 0.897587                  0.170616
AMT_PAYMENT_CURRENT                     0.142371                  0.142935
```

AMT_PAYMENT_TOTAL_CURRENT	0.149914	0.125655
AMT_RECEIVABLE_PRINCIPAL	0.999702	0.302627
AMT_RECIVABLE	0.999995	0.303571
AMT_TOTAL_RECEIVABLE	1.000000	0.303542
CNT_DRAWINGS_ATM_CURRENT	0.303542	1.000000
CNT_DRAWINGS_CURRENT	0.256317	0.410907
CNT_DRAWINGS_OTHER_CURRENT	0.046113	0.012730
CNT_DRAWINGS_POS_CURRENT	0.154481	0.108388
CNT_INSTALMENT_MATURE_CUM	0.005959	-0.103403
SK_DPD	-0.046047	-0.029395
SK_DPD_DEF	0.017243	-0.004277

	CNT_DRAWINGS_CURRENT	CNT_DRAWINGS_OTHER_CURRENT \
SK_ID_PREV	0.000367	-0.001412
SK_ID_CURR	0.002654	-0.000131
MONTHS_BALANCE	0.113321	-0.026192
AMT_BALANCE	0.259184	0.046563
AMT_CREDIT_LIMIT_ACTUAL	0.204237	0.030051
AMT_DRAWINGS_ATM_CURRENT	0.298173	0.013254
AMT_DRAWINGS_CURRENT	0.523016	0.140032
AMT_DRAWINGS_OTHER_CURRENT	0.021271	0.575295
AMT_DRAWINGS_POS_CURRENT	0.520123	0.007620
AMT_INST_MIN_REGULARITY	0.148262	0.014360
AMT_PAYMENT_CURRENT	0.223483	0.017246
AMT_PAYMENT_TOTAL_CURRENT	0.217857	0.014041
AMT_RECEIVABLE_PRINCIPAL	0.258848	0.046543
AMT_RECIVABLE	0.256347	0.046118
AMT_TOTAL_RECEIVABLE	0.256317	0.046113
CNT_DRAWINGS_ATM_CURRENT	0.410907	0.012730
CNT_DRAWINGS_CURRENT	1.000000	0.033940
CNT_DRAWINGS_OTHER_CURRENT	0.033940	1.000000
CNT_DRAWINGS_POS_CURRENT	0.950546	0.007203
CNT_INSTALMENT_MATURE_CUM	-0.099186	-0.021632
SK_DPD	-0.020786	-0.006083
SK_DPD_DEF	-0.003106	-0.000895

	CNT_DRAWINGS_POS_CURRENT \
SK_ID_PREV	0.000809
SK_ID_CURR	0.002135
MONTHS_BALANCE	0.160207
AMT_BALANCE	0.155553
AMT_CREDIT_LIMIT_ACTUAL	0.202868
AMT_DRAWINGS_ATM_CURRENT	0.076083
AMT_DRAWINGS_CURRENT	0.359001
AMT_DRAWINGS_OTHER_CURRENT	0.004458
AMT_DRAWINGS_POS_CURRENT	0.542556
AMT_INST_MIN_REGULARITY	0.086729
AMT_PAYMENT_CURRENT	0.195074
AMT_PAYMENT_TOTAL_CURRENT	0.183973
AMT_RECEIVABLE_PRINCIPAL	0.157723
AMT_RECIVABLE	0.154507
AMT_TOTAL_RECEIVABLE	0.154481
CNT_DRAWINGS_ATM_CURRENT	0.108388
CNT_DRAWINGS_CURRENT	0.950546
CNT_DRAWINGS_OTHER_CURRENT	0.007203
CNT_DRAWINGS_POS_CURRENT	1.000000
CNT_INSTALMENT_MATURE_CUM	-0.129338
SK_DPD	-0.018212
SK_DPD_DEF	-0.002840

	CNT_INSTALMENT_MATURE_CUM	SK_DPD	SK_DPD_DEF
SK_ID_PREV	-0.007219	-0.001786	0.001973
SK_ID_CURR	-0.000581	-0.000962	0.001519
MONTHS_BALANCE	-0.008620	0.039434	0.001659
AMT_BALANCE	0.005009	-0.046988	0.013009
AMT_CREDIT_LIMIT_ACTUAL	-0.157269	-0.038791	-0.002236

```
AMT_DRAWINGS_ATM_CURRENT                  -0.103721 -0.022044   -0.003360
AMT_DRAWINGS_CURRENT                      -0.093491 -0.020606   -0.003137
AMT_DRAWINGS_OTHER_CURRENT                -0.023013 -0.003693   -0.000568
AMT_DRAWINGS_POS_CURRENT                  -0.106813 -0.015040   -0.002384
AMT_INST_MIN_REGULARITY                    0.064320 -0.061484   -0.005715
AMT_PAYMENT_CURRENT                       -0.079266 -0.030222   -0.004340
AMT_PAYMENT_TOTAL_CURRENT                 -0.023156 -0.022475   -0.003443
AMT_RECEIVABLE_PRINCIPAL                   0.003664 -0.048290    0.006780
AMT_RECIVABLE                              0.005935 -0.046434    0.015466
AMT_TOTAL_RECEIVABLE                       0.005959 -0.046047    0.017243
CNT_DRAWINGS_ATM_CURRENT                  -0.103403 -0.029395   -0.004277
CNT_DRAWINGS_CURRENT                      -0.099186 -0.020786   -0.003106
CNT_DRAWINGS_OTHER_CURRENT                -0.021632 -0.006083   -0.000895
CNT_DRAWINGS_POS_CURRENT                  -0.129338 -0.018212   -0.002840
CNT_INSTALMENT_MATURE_CUM                  1.000000  0.059654    0.002156
SK_DPD                                     0.059654  1.000000    0.218950
SK_DPD_DEF                                 0.002156  0.218950    1.000000

[22 rows x 22 columns]

---------------------------------------------------------------------------

Other Analysis: credit_card_balance
1. Checking for Null values: credit_card_balance
SK_ID_PREV                         0
SK_ID_CURR                         0
MONTHS_BALANCE                     0
AMT_BALANCE                        0
AMT_CREDIT_LIMIT_ACTUAL            0
AMT_DRAWINGS_ATM_CURRENT      749816
AMT_DRAWINGS_CURRENT               0
AMT_DRAWINGS_OTHER_CURRENT    749816
AMT_DRAWINGS_POS_CURRENT      749816
AMT_INST_MIN_REGULARITY       305236
AMT_PAYMENT_CURRENT           767988
AMT_PAYMENT_TOTAL_CURRENT          0
AMT_RECEIVABLE_PRINCIPAL           0
AMT_RECIVABLE                      0
AMT_TOTAL_RECEIVABLE               0
CNT_DRAWINGS_ATM_CURRENT      749816
CNT_DRAWINGS_CURRENT               0
CNT_DRAWINGS_OTHER_CURRENT    749816
CNT_DRAWINGS_POS_CURRENT      749816
CNT_INSTALMENT_MATURE_CUM     305236
NAME_CONTRACT_STATUS               0
SK_DPD                             0
SK_DPD_DEF                         0
dtype: int64

2. Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3840312 entries, 0 to 3840311
Data columns (total 23 columns):
 #   Column                      Dtype
---  ------                      -----
 0   SK_ID_PREV                  int64
 1   SK_ID_CURR                  int64
 2   MONTHS_BALANCE              int64
 3   AMT_BALANCE                 float64
 4   AMT_CREDIT_LIMIT_ACTUAL     int64
 5   AMT_DRAWINGS_ATM_CURRENT    float64
 6   AMT_DRAWINGS_CURRENT        float64
 7   AMT_DRAWINGS_OTHER_CURRENT  float64
 8   AMT_DRAWINGS_POS_CURRENT    float64
 9   AMT_INST_MIN_REGULARITY     float64
 10  AMT_PAYMENT_CURRENT         float64
```

```
 11   AMT_PAYMENT_TOTAL_CURRENT    float64
 12   AMT_RECEIVABLE_PRINCIPAL     float64
 13   AMT_RECIVABLE                float64
 14   AMT_TOTAL_RECEIVABLE         float64
 15   CNT_DRAWINGS_ATM_CURRENT     float64
 16   CNT_DRAWINGS_CURRENT         int64
 17   CNT_DRAWINGS_OTHER_CURRENT   float64
 18   CNT_DRAWINGS_POS_CURRENT     float64
 19   CNT_INSTALMENT_MATURE_CUM    float64
 20   NAME_CONTRACT_STATUS         object
 21   SK_DPD                       int64
 22   SK_DPD_DEF                   int64
dtypes: float64(15), int64(7), object(1)
memory usage: 673.9+ MB
None
```

In [15]:

```
installments_payments = datasets['installments_payments'].copy()
Exploratory_Data_Analysis(installments_payments,'installments_payments')
```

```
Test description; data type: installments_payments
SK_ID_PREV                  int64
SK_ID_CURR                  int64
NUM_INSTALMENT_VERSION    float64
NUM_INSTALMENT_NUMBER       int64
DAYS_INSTALMENT           float64
DAYS_ENTRY_PAYMENT        float64
AMT_INSTALMENT            float64
AMT_PAYMENT               float64
dtype: object

--------------------------------------------------------------------------

 Dataset size (rows columns): installments_payments
(13605401, 8)

--------------------------------------------------------------------------

Summary statistics: installments_payments
         SK_ID_PREV    SK_ID_CURR  NUM_INSTALMENT_VERSION  \
count  1.360540e+07  1.360540e+07            1.360540e+07
mean   1.903365e+06  2.784449e+05            8.566373e-01
std    5.362029e+05  1.027183e+05            1.035216e+00
min    1.000001e+06  1.000010e+05            0.000000e+00
25%    1.434191e+06  1.896390e+05            0.000000e+00
50%    1.896520e+06  2.786850e+05            1.000000e+00
75%    2.369094e+06  3.675300e+05            1.000000e+00
max    2.843499e+06  4.562550e+05            1.780000e+02

       NUM_INSTALMENT_NUMBER  DAYS_INSTALMENT  DAYS_ENTRY_PAYMENT  \
count           1.360540e+07     1.360540e+07        1.360250e+07
mean            1.887090e+01    -1.042270e+03       -1.051114e+03
std             2.666407e+01     8.009463e+02        8.005859e+02
min             1.000000e+00    -2.922000e+03       -4.921000e+03
25%             4.000000e+00    -1.654000e+03       -1.662000e+03
50%             8.000000e+00    -8.180000e+02       -8.270000e+02
75%             1.900000e+01    -3.610000e+02       -3.700000e+02
max             2.770000e+02    -1.000000e+00       -1.000000e+00

       AMT_INSTALMENT   AMT_PAYMENT
count    1.360540e+07  1.360250e+07
mean     1.705091e+04  1.723822e+04
std      5.057025e+04  5.473578e+04
min      0.000000e+00  0.000000e+00
25%      4.226085e+03  3.398265e+03
50%      8.884080e+03  8.125515e+03
```

```
75%      1.671021e+04  1.610842e+04
max      3.771488e+06  3.771488e+06

--------------------------------------------------------------------------

Correlation analysis: installments_payments
                        SK_ID_PREV  SK_ID_CURR  NUM_INSTALMENT_VERSION  \
SK_ID_PREV                1.000000    0.002132                0.000685
SK_ID_CURR                0.002132    1.000000                0.000480
NUM_INSTALMENT_VERSION    0.000685    0.000480                1.000000
NUM_INSTALMENT_NUMBER    -0.002095   -0.000548               -0.323414
DAYS_INSTALMENT           0.003748    0.001191                0.130244
DAYS_ENTRY_PAYMENT        0.003734    0.001215                0.128124
AMT_INSTALMENT            0.002042   -0.000226                0.168109
AMT_PAYMENT               0.001887   -0.000124                0.177176

                        NUM_INSTALMENT_NUMBER  DAYS_INSTALMENT  \
SK_ID_PREV                          -0.002095         0.003748
SK_ID_CURR                          -0.000548         0.001191
NUM_INSTALMENT_VERSION              -0.323414         0.130244
NUM_INSTALMENT_NUMBER                1.000000         0.090286
DAYS_INSTALMENT                      0.090286         1.000000
DAYS_ENTRY_PAYMENT                   0.094305         0.999491
AMT_INSTALMENT                      -0.089640         0.125985
AMT_PAYMENT                         -0.087664         0.127018

                        DAYS_ENTRY_PAYMENT  AMT_INSTALMENT  AMT_PAYMENT
SK_ID_PREV                        0.003734        0.002042     0.001887
SK_ID_CURR                        0.001215       -0.000226    -0.000124
NUM_INSTALMENT_VERSION            0.128124        0.168109     0.177176
NUM_INSTALMENT_NUMBER             0.094305       -0.089640    -0.087664
DAYS_INSTALMENT                   0.999491        0.125985     0.127018
DAYS_ENTRY_PAYMENT                1.000000        0.125555     0.126602
AMT_INSTALMENT                    0.125555        1.000000     0.937191
AMT_PAYMENT                       0.126602        0.937191     1.000000

--------------------------------------------------------------------------

Other Analysis: installments_payments
1. Checking for Null values: installments_payments
SK_ID_PREV                   0
SK_ID_CURR                   0
NUM_INSTALMENT_VERSION       0
NUM_INSTALMENT_NUMBER        0
DAYS_INSTALMENT              0
DAYS_ENTRY_PAYMENT        2905
AMT_INSTALMENT               0
AMT_PAYMENT               2905
dtype: int64

2. Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13605401 entries, 0 to 13605400
Data columns (total 8 columns):
 #   Column                  Dtype
---  ------                  -----
 0   SK_ID_PREV              int64
 1   SK_ID_CURR              int64
 2   NUM_INSTALMENT_VERSION  float64
 3   NUM_INSTALMENT_NUMBER   int64
 4   DAYS_INSTALMENT         float64
 5   DAYS_ENTRY_PAYMENT      float64
 6   AMT_INSTALMENT          float64
 7   AMT_PAYMENT             float64
dtypes: float64(5), int64(3)
```

```
memory usage: 830.4 MB
None
```

In [16]:

```python
previous_application   = datasets['previous_application'].copy()
Exploratory_Data_Analysis(previous_application ,'previous_application')
```

```
Test description; data type: previous_application
SK_ID_PREV                      int64
SK_ID_CURR                      int64
NAME_CONTRACT_TYPE             object
AMT_ANNUITY                   float64
AMT_APPLICATION               float64
AMT_CREDIT                    float64
AMT_DOWN_PAYMENT              float64
AMT_GOODS_PRICE               float64
WEEKDAY_APPR_PROCESS_START     object
HOUR_APPR_PROCESS_START         int64
FLAG_LAST_APPL_PER_CONTRACT    object
NFLAG_LAST_APPL_IN_DAY          int64
RATE_DOWN_PAYMENT             float64
RATE_INTEREST_PRIMARY         float64
RATE_INTEREST_PRIVILEGED      float64
NAME_CASH_LOAN_PURPOSE         object
NAME_CONTRACT_STATUS           object
DAYS_DECISION                   int64
NAME_PAYMENT_TYPE              object
CODE_REJECT_REASON             object
NAME_TYPE_SUITE                object
NAME_CLIENT_TYPE               object
NAME_GOODS_CATEGORY            object
NAME_PORTFOLIO                 object
NAME_PRODUCT_TYPE              object
CHANNEL_TYPE                   object
SELLERPLACE_AREA                int64
NAME_SELLER_INDUSTRY           object
CNT_PAYMENT                   float64
NAME_YIELD_GROUP               object
PRODUCT_COMBINATION            object
DAYS_FIRST_DRAWING            float64
DAYS_FIRST_DUE                float64
DAYS_LAST_DUE_1ST_VERSION     float64
DAYS_LAST_DUE                 float64
DAYS_TERMINATION              float64
NFLAG_INSURED_ON_APPROVAL     float64
dtype: object

---------------------------------------------------------------------------

 Dataset size (rows columns): previous_application
(1670214, 37)

---------------------------------------------------------------------------

Summary statistics: previous_application
         SK_ID_PREV    SK_ID_CURR   AMT_ANNUITY  AMT_APPLICATION  \
count  1.670214e+06  1.670214e+06  1.297979e+06     1.670214e+06
mean   1.923089e+06  2.783572e+05  1.595512e+04     1.752339e+05
std    5.325980e+05  1.028148e+05  1.478214e+04     2.927798e+05
min    1.000001e+06  1.000010e+05  0.000000e+00     0.000000e+00
25%    1.461857e+06  1.893290e+05  6.321780e+03     1.872000e+04
50%    1.923110e+06  2.787145e+05  1.125000e+04     7.104600e+04
75%    2.384280e+06  3.675140e+05  2.065842e+04     1.803600e+05
max    2.845382e+06  4.562550e+05  4.180581e+05     6.905160e+06

         AMT_CREDIT  AMT_DOWN_PAYMENT  AMT_GOODS_PRICE  \
```

```
count  1.670213e+06      7.743700e+05     1.284699e+06
mean   1.961140e+05      6.697402e+03     2.278473e+05
std    3.185746e+05      2.092150e+04     3.153966e+05
min    0.000000e+00     -9.000000e-01     0.000000e+00
25%    2.416050e+04      0.000000e+00     5.084100e+04
50%    8.054100e+04      1.638000e+03     1.123200e+05
75%    2.164185e+05      7.740000e+03     2.340000e+05
max    6.905160e+06      3.060045e+06     6.905160e+06

       HOUR_APPR_PROCESS_START  NFLAG_LAST_APPL_IN_DAY  RATE_DOWN_PAYMENT  \
count             1.670214e+06            1.670214e+06      774370.000000
mean              1.248418e+01            9.964675e-01           0.079637
std               3.334028e+00            5.932963e-02           0.107823
min               0.000000e+00            0.000000e+00          -0.000015
25%               1.000000e+01            1.000000e+00           0.000000
50%               1.200000e+01            1.000000e+00           0.051605
75%               1.500000e+01            1.000000e+00           0.108909
max               2.300000e+01            1.000000e+00           1.000000

       ...  RATE_INTEREST_PRIVILEGED  DAYS_DECISION  SELLERPLACE_AREA  \
count  ...               5951.000000   1.670214e+06      1.670214e+06
mean   ...                  0.773503  -8.806797e+02      3.139511e+02
std    ...                  0.100879   7.790997e+02      7.127443e+03
min    ...                  0.373150  -2.922000e+03     -1.000000e+00
25%    ...                  0.715645  -1.300000e+03     -1.000000e+00
50%    ...                  0.835095  -5.810000e+02      3.000000e+00
75%    ...                  0.852537  -2.800000e+02      8.200000e+01
max    ...                  1.000000  -1.000000e+00      4.000000e+06

        CNT_PAYMENT  DAYS_FIRST_DRAWING  DAYS_FIRST_DUE  \
count  1.297984e+06       997149.000000   997149.000000
mean   1.605408e+01       342209.855039    13826.269337
std    1.456729e+01        88916.115834    72444.869708
min    0.000000e+00        -2922.000000    -2892.000000
25%    6.000000e+00       365243.000000    -1628.000000
50%    1.200000e+01       365243.000000     -831.000000
75%    2.400000e+01       365243.000000     -411.000000
max    8.400000e+01       365243.000000   365243.000000

       DAYS_LAST_DUE_1ST_VERSION  DAYS_LAST_DUE  DAYS_TERMINATION  \
count              997149.000000  997149.000000     997149.000000
mean                33767.774054   76582.403064      81992.343838
std                106857.034789  149647.415123     153303.516729
min                 -2801.000000   -2889.000000      -2874.000000
25%                 -1242.000000   -1314.000000      -1270.000000
50%                  -361.000000    -537.000000       -499.000000
75%                   129.000000     -74.000000        -44.000000
max                365243.000000  365243.000000     365243.000000

       NFLAG_INSURED_ON_APPROVAL
count              997149.000000
mean                    0.332570
std                     0.471134
min                     0.000000
25%                     0.000000
50%                     0.000000
75%                     1.000000
max                     1.000000

[8 rows x 21 columns]

---------------------------------------------------------------------------

Correlation analysis: previous_application
                           SK_ID_PREV  SK_ID_CURR  AMT_ANNUITY  \
SK_ID_PREV                   1.000000   -0.000321     0.011459
```

SK_ID_CURR	-0.000321	1.000000	0.000577
AMT_ANNUITY	0.011459	0.000577	1.000000
AMT_APPLICATION	0.003302	0.000280	0.808872
AMT_CREDIT	0.003659	0.000195	0.816429
AMT_DOWN_PAYMENT	-0.001313	-0.000063	0.267694
AMT_GOODS_PRICE	0.015293	0.000369	0.820895
HOUR_APPR_PROCESS_START	-0.002652	0.002842	-0.036201
NFLAG_LAST_APPL_IN_DAY	-0.002828	0.000098	0.020639
RATE_DOWN_PAYMENT	-0.004051	0.001158	-0.103878
RATE_INTEREST_PRIMARY	0.012969	0.033197	0.141823
RATE_INTEREST_PRIVILEGED	-0.022312	-0.016757	-0.202335
DAYS_DECISION	0.019100	-0.000637	0.279051
SELLERPLACE_AREA	-0.001079	0.001265	-0.015027
CNT_PAYMENT	0.015589	0.000031	0.394535
DAYS_FIRST_DRAWING	-0.001478	-0.001329	0.052839
DAYS_FIRST_DUE	-0.000071	-0.000757	-0.053295
DAYS_LAST_DUE_1ST_VERSION	0.001222	0.000252	-0.068877
DAYS_LAST_DUE	0.001915	-0.000318	0.082659
DAYS_TERMINATION	0.001781	-0.000020	0.068022
NFLAG_INSURED_ON_APPROVAL	0.003986	0.000876	0.283080

	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT \
SK_ID_PREV	0.003302	0.003659	-0.001313
SK_ID_CURR	0.000280	0.000195	-0.000063
AMT_ANNUITY	0.808872	0.816429	0.267694
AMT_APPLICATION	1.000000	0.975824	0.482776
AMT_CREDIT	0.975824	1.000000	0.301284
AMT_DOWN_PAYMENT	0.482776	0.301284	1.000000
AMT_GOODS_PRICE	0.999884	0.993087	0.482776
HOUR_APPR_PROCESS_START	-0.014415	-0.021039	0.016776
NFLAG_LAST_APPL_IN_DAY	0.004310	-0.025179	0.001597
RATE_DOWN_PAYMENT	-0.072479	-0.188128	0.473935
RATE_INTEREST_PRIMARY	0.110001	0.125106	0.016323
RATE_INTEREST_PRIVILEGED	-0.199733	-0.205158	-0.115343
DAYS_DECISION	0.133660	0.133763	-0.024536
SELLERPLACE_AREA	-0.007649	-0.009567	0.003533
CNT_PAYMENT	0.680630	0.674278	0.031659
DAYS_FIRST_DRAWING	0.074544	-0.036813	-0.001773
DAYS_FIRST_DUE	-0.049532	0.002881	-0.013586
DAYS_LAST_DUE_1ST_VERSION	-0.084905	0.044031	-0.000869
DAYS_LAST_DUE	0.172627	0.224829	-0.031425
DAYS_TERMINATION	0.148618	0.214320	-0.030702
NFLAG_INSURED_ON_APPROVAL	0.259219	0.263932	-0.042585

	AMT_GOODS_PRICE	HOUR_APPR_PROCESS_START \
SK_ID_PREV	0.015293	-0.002652
SK_ID_CURR	0.000369	0.002842
AMT_ANNUITY	0.820895	-0.036201
AMT_APPLICATION	0.999884	-0.014415
AMT_CREDIT	0.993087	-0.021039
AMT_DOWN_PAYMENT	0.482776	0.016776
AMT_GOODS_PRICE	1.000000	-0.045267
HOUR_APPR_PROCESS_START	-0.045267	1.000000
NFLAG_LAST_APPL_IN_DAY	-0.017100	0.005789
RATE_DOWN_PAYMENT	-0.072479	0.025930
RATE_INTEREST_PRIMARY	0.110001	-0.027172
RATE_INTEREST_PRIVILEGED	-0.199733	-0.045720
DAYS_DECISION	0.290422	-0.039962
SELLERPLACE_AREA	-0.015842	0.015671
CNT_PAYMENT	0.672129	-0.055511
DAYS_FIRST_DRAWING	-0.024445	0.014321
DAYS_FIRST_DUE	-0.021062	-0.002797
DAYS_LAST_DUE_1ST_VERSION	0.016883	-0.016567
DAYS_LAST_DUE	0.211696	-0.018018
DAYS_TERMINATION	0.209296	-0.018254
NFLAG_INSURED_ON_APPROVAL	0.243400	-0.117318

	NFLAG_LAST_APPL_IN_DAY	RATE_DOWN_PAYMENT	...
SK_ID_PREV	-0.002828	-0.004051	...
SK_ID_CURR	0.000098	0.001158	...
AMT_ANNUITY	0.020639	-0.103878	...
AMT_APPLICATION	0.004310	-0.072479	...
AMT_CREDIT	-0.025179	-0.188128	...
AMT_DOWN_PAYMENT	0.001597	0.473935	...
AMT_GOODS_PRICE	-0.017100	-0.072479	...
HOUR_APPR_PROCESS_START	0.005789	0.025930	...
NFLAG_LAST_APPL_IN_DAY	1.000000	0.004554	...
RATE_DOWN_PAYMENT	0.004554	1.000000	...
RATE_INTEREST_PRIMARY	0.009604	-0.103373	...
RATE_INTEREST_PRIVILEGED	0.024640	-0.106143	...
DAYS_DECISION	0.016555	-0.208742	...
SELLERPLACE_AREA	0.000912	-0.006489	...
CNT_PAYMENT	0.063347	-0.278875	...
DAYS_FIRST_DRAWING	-0.000409	-0.007969	...
DAYS_FIRST_DUE	-0.002288	-0.039178	...
DAYS_LAST_DUE_1ST_VERSION	-0.001981	-0.010934	...
DAYS_LAST_DUE	-0.002277	-0.147562	...
DAYS_TERMINATION	-0.000744	-0.145461	...
NFLAG_INSURED_ON_APPROVAL	-0.007124	-0.021633	...

	RATE_INTEREST_PRIVILEGED	DAYS_DECISION
SK_ID_PREV	-0.022312	0.019100
SK_ID_CURR	-0.016757	-0.000637
AMT_ANNUITY	-0.202335	0.279051
AMT_APPLICATION	-0.199733	0.133660
AMT_CREDIT	-0.205158	0.133763
AMT_DOWN_PAYMENT	-0.115343	-0.024536
AMT_GOODS_PRICE	-0.199733	0.290422
HOUR_APPR_PROCESS_START	-0.045720	-0.039962
NFLAG_LAST_APPL_IN_DAY	0.024640	0.016555
RATE_DOWN_PAYMENT	-0.106143	-0.208742
RATE_INTEREST_PRIMARY	-0.001937	0.014037
RATE_INTEREST_PRIVILEGED	1.000000	0.631940
DAYS_DECISION	0.631940	1.000000
SELLERPLACE_AREA	-0.066316	-0.018382
CNT_PAYMENT	-0.057150	0.246453
DAYS_FIRST_DRAWING	NaN	-0.012007
DAYS_FIRST_DUE	0.150904	0.176711
DAYS_LAST_DUE_1ST_VERSION	0.030513	0.089167
DAYS_LAST_DUE	0.372214	0.448549
DAYS_TERMINATION	0.378671	0.400179
NFLAG_INSURED_ON_APPROVAL	-0.067157	-0.028905

	SELLERPLACE_AREA	CNT_PAYMENT	DAYS_FIRST_DRAWING
SK_ID_PREV	-0.001079	0.015589	-0.001478
SK_ID_CURR	0.001265	0.000031	-0.001329
AMT_ANNUITY	-0.015027	0.394535	0.052839
AMT_APPLICATION	-0.007649	0.680630	0.074544
AMT_CREDIT	-0.009567	0.674278	-0.036813
AMT_DOWN_PAYMENT	0.003533	0.031659	-0.001773
AMT_GOODS_PRICE	-0.015842	0.672129	-0.024445
HOUR_APPR_PROCESS_START	0.015671	-0.055511	0.014321
NFLAG_LAST_APPL_IN_DAY	0.000912	0.063347	-0.000409
RATE_DOWN_PAYMENT	-0.006489	-0.278875	-0.007969
RATE_INTEREST_PRIMARY	0.159182	-0.019030	NaN
RATE_INTEREST_PRIVILEGED	-0.066316	-0.057150	NaN
DAYS_DECISION	-0.018382	0.246453	-0.012007
SELLERPLACE_AREA	1.000000	-0.010646	0.007401
CNT_PAYMENT	-0.010646	1.000000	0.309900
DAYS_FIRST_DRAWING	0.007401	0.309900	1.000000
DAYS_FIRST_DUE	-0.002166	-0.204907	0.004710
DAYS_LAST_DUE_1ST_VERSION	-0.007510	-0.381013	-0.803494

```
DAYS_LAST_DUE                  -0.006291     0.088903              -0.257466
DAYS_TERMINATION               -0.006675     0.055121              -0.396284
NFLAG_INSURED_ON_APPROVAL      -0.018280     0.320520               0.177652

                           DAYS_FIRST_DUE  DAYS_LAST_DUE_1ST_VERSION  \
SK_ID_PREV                      -0.000071                   0.001222
SK_ID_CURR                      -0.000757                   0.000252
AMT_ANNUITY                     -0.053295                  -0.068877
AMT_APPLICATION                 -0.049532                  -0.084905
AMT_CREDIT                       0.002881                   0.044031
AMT_DOWN_PAYMENT                -0.013586                  -0.000869
AMT_GOODS_PRICE                 -0.021062                   0.016883
HOUR_APPR_PROCESS_START         -0.002797                  -0.016567
NFLAG_LAST_APPL_IN_DAY          -0.002288                  -0.001981
RATE_DOWN_PAYMENT               -0.039178                  -0.010934
RATE_INTEREST_PRIMARY           -0.017171                  -0.000933
RATE_INTEREST_PRIVILEGED         0.150904                   0.030513
DAYS_DECISION                    0.176711                   0.089167
SELLERPLACE_AREA                -0.002166                  -0.007510
CNT_PAYMENT                     -0.204907                  -0.381013
DAYS_FIRST_DRAWING               0.004710                  -0.803494
DAYS_FIRST_DUE                   1.000000                   0.513949
DAYS_LAST_DUE_1ST_VERSION        0.513949                   1.000000
DAYS_LAST_DUE                    0.401838                   0.423462
DAYS_TERMINATION                 0.323608                   0.493174
NFLAG_INSURED_ON_APPROVAL       -0.119048                  -0.221947

                           DAYS_LAST_DUE  DAYS_TERMINATION  \
SK_ID_PREV                      0.001915          0.001781
SK_ID_CURR                     -0.000318         -0.000020
AMT_ANNUITY                     0.082659          0.068022
AMT_APPLICATION                 0.172627          0.148618
AMT_CREDIT                      0.224829          0.214320
AMT_DOWN_PAYMENT               -0.031425         -0.030702
AMT_GOODS_PRICE                 0.211696          0.209296
HOUR_APPR_PROCESS_START        -0.018018         -0.018254
NFLAG_LAST_APPL_IN_DAY         -0.002277         -0.000744
RATE_DOWN_PAYMENT              -0.147562         -0.145461
RATE_INTEREST_PRIMARY          -0.010677         -0.011099
RATE_INTEREST_PRIVILEGED        0.372214          0.378671
DAYS_DECISION                   0.448549          0.400179
SELLERPLACE_AREA               -0.006291         -0.006675
CNT_PAYMENT                     0.088903          0.055121
DAYS_FIRST_DRAWING             -0.257466         -0.396284
DAYS_FIRST_DUE                  0.401838          0.323608
DAYS_LAST_DUE_1ST_VERSION       0.423462          0.493174
DAYS_LAST_DUE                   1.000000          0.927990
DAYS_TERMINATION                0.927990          1.000000
NFLAG_INSURED_ON_APPROVAL       0.012560         -0.003065

                           NFLAG_INSURED_ON_APPROVAL
SK_ID_PREV                                  0.003986
SK_ID_CURR                                  0.000876
AMT_ANNUITY                                 0.283080
AMT_APPLICATION                             0.259219
AMT_CREDIT                                  0.263932
AMT_DOWN_PAYMENT                           -0.042585
AMT_GOODS_PRICE                             0.243400
HOUR_APPR_PROCESS_START                    -0.117318
NFLAG_LAST_APPL_IN_DAY                     -0.007124
RATE_DOWN_PAYMENT                          -0.021633
RATE_INTEREST_PRIMARY                       0.311938
RATE_INTEREST_PRIVILEGED                   -0.067157
DAYS_DECISION                              -0.028905
SELLERPLACE_AREA                           -0.018280
CNT_PAYMENT                                 0.320520
```

```
DAYS_FIRST_DRAWING                              0.177652
DAYS_FIRST_DUE                                 -0.119048
DAYS_LAST_DUE_1ST_VERSION                      -0.221947
DAYS_LAST_DUE                                   0.012560
DAYS_TERMINATION                               -0.003065
NFLAG_INSURED_ON_APPROVAL                       1.000000

[21 rows x 21 columns]

---------------------------------------------------------------------------

Other Analysis: previous_application
1. Checking for Null values: previous_application
SK_ID_PREV                           0
SK_ID_CURR                           0
NAME_CONTRACT_TYPE                   0
AMT_ANNUITY                     372235
AMT_APPLICATION                      0
AMT_CREDIT                           1
AMT_DOWN_PAYMENT                895844
AMT_GOODS_PRICE                 385515
WEEKDAY_APPR_PROCESS_START           0
HOUR_APPR_PROCESS_START              0
FLAG_LAST_APPL_PER_CONTRACT          0
NFLAG_LAST_APPL_IN_DAY               0
RATE_DOWN_PAYMENT               895844
RATE_INTEREST_PRIMARY          1664263
RATE_INTEREST_PRIVILEGED       1664263
NAME_CASH_LOAN_PURPOSE               0
NAME_CONTRACT_STATUS                 0
DAYS_DECISION                        0
NAME_PAYMENT_TYPE                    0
CODE_REJECT_REASON                   0
NAME_TYPE_SUITE                 820405
NAME_CLIENT_TYPE                     0
NAME_GOODS_CATEGORY                  0
NAME_PORTFOLIO                       0
NAME_PRODUCT_TYPE                    0
CHANNEL_TYPE                         0
SELLERPLACE_AREA                     0
NAME_SELLER_INDUSTRY                 0
CNT_PAYMENT                     372230
NAME_YIELD_GROUP                     0
PRODUCT_COMBINATION                346
DAYS_FIRST_DRAWING              673065
DAYS_FIRST_DUE                  673065
DAYS_LAST_DUE_1ST_VERSION       673065
DAYS_LAST_DUE                   673065
DAYS_TERMINATION                673065
NFLAG_INSURED_ON_APPROVAL       673065
dtype: int64

2. Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 37 columns):
 #   Column                       Non-Null Count    Dtype
---  ------                       --------------    -----
 0   SK_ID_PREV                   1670214 non-null  int64
 1   SK_ID_CURR                   1670214 non-null  int64
 2   NAME_CONTRACT_TYPE           1670214 non-null  object
 3   AMT_ANNUITY                  1297979 non-null  float64
 4   AMT_APPLICATION              1670214 non-null  float64
 5   AMT_CREDIT                   1670213 non-null  float64
 6   AMT_DOWN_PAYMENT             774370 non-null   float64
 7   AMT_GOODS_PRICE              1284699 non-null  float64
```

```
 8   WEEKDAY_APPR_PROCESS_START   1670214 non-null  object
 9   HOUR_APPR_PROCESS_START      1670214 non-null  int64
 10  FLAG_LAST_APPL_PER_CONTRACT  1670214 non-null  object
 11  NFLAG_LAST_APPL_IN_DAY       1670214 non-null  int64
 12  RATE_DOWN_PAYMENT            774370 non-null   float64
 13  RATE_INTEREST_PRIMARY        5951 non-null     float64
 14  RATE_INTEREST_PRIVILEGED     5951 non-null     float64
 15  NAME_CASH_LOAN_PURPOSE       1670214 non-null  object
 16  NAME_CONTRACT_STATUS         1670214 non-null  object
 17  DAYS_DECISION                1670214 non-null  int64
 18  NAME_PAYMENT_TYPE            1670214 non-null  object
 19  CODE_REJECT_REASON           1670214 non-null  object
 20  NAME_TYPE_SUITE              849809 non-null   object
 21  NAME_CLIENT_TYPE             1670214 non-null  object
 22  NAME_GOODS_CATEGORY          1670214 non-null  object
 23  NAME_PORTFOLIO               1670214 non-null  object
 24  NAME_PRODUCT_TYPE            1670214 non-null  object
 25  CHANNEL_TYPE                 1670214 non-null  object
 26  SELLERPLACE_AREA             1670214 non-null  int64
 27  NAME_SELLER_INDUSTRY         1670214 non-null  object
 28  CNT_PAYMENT                  1297984 non-null  float64
 29  NAME_YIELD_GROUP             1670214 non-null  object
 30  PRODUCT_COMBINATION          1669868 non-null  object
 31  DAYS_FIRST_DRAWING           997149 non-null   float64
 32  DAYS_FIRST_DUE               997149 non-null   float64
 33  DAYS_LAST_DUE_1ST_VERSION    997149 non-null   float64
 34  DAYS_LAST_DUE                997149 non-null   float64
 35  DAYS_TERMINATION             997149 non-null   float64
 36  NFLAG_INSURED_ON_APPROVAL    997149 non-null   float64
dtypes: float64(15), int64(6), object(16)
memory usage: 471.5+ MB
None
```

In [17]:

```
POS_CASH_balance = datasets['POS_CASH_balance'].copy()
Exploratory_Data_Analysis(POS_CASH_balance ,'POS_CASH_balance')
```

```
Test description; data type: POS_CASH_balance
SK_ID_PREV                 int64
SK_ID_CURR                 int64
MONTHS_BALANCE             int64
CNT_INSTALMENT           float64
CNT_INSTALMENT_FUTURE    float64
NAME_CONTRACT_STATUS      object
SK_DPD                     int64
SK_DPD_DEF                 int64
dtype: object

--------------------------------------------------------------------------

 Dataset size (rows columns): POS_CASH_balance
(10001358, 8)

--------------------------------------------------------------------------

Summary statistics: POS_CASH_balance
         SK_ID_PREV    SK_ID_CURR  MONTHS_BALANCE  CNT_INSTALMENT  \
count  1.000136e+07  1.000136e+07    1.000136e+07    9.975287e+06
mean   1.903217e+06  2.784039e+05   -3.501259e+01    1.708965e+01
std    5.358465e+05  1.027637e+05    2.606657e+01    1.199506e+01
min    1.000001e+06  1.000010e+05   -9.600000e+01    1.000000e+00
25%    1.434405e+06  1.895500e+05   -5.400000e+01    1.000000e+01
50%    1.896565e+06  2.786540e+05   -2.800000e+01    1.200000e+01
75%    2.368963e+06  3.674290e+05   -1.300000e+01    2.400000e+01
max    2.843499e+06  4.562550e+05   -1.000000e+00    9.200000e+01
```

```
       CNT_INSTALMENT_FUTURE        SK_DPD    SK_DPD_DEF
count           9.975271e+06  1.000136e+07  1.000136e+07
mean            1.048384e+01  1.160693e+01  6.544684e-01
std             1.110906e+01  1.327140e+02  3.276249e+01
min             0.000000e+00  0.000000e+00  0.000000e+00
25%             3.000000e+00  0.000000e+00  0.000000e+00
50%             7.000000e+00  0.000000e+00  0.000000e+00
75%             1.400000e+01  0.000000e+00  0.000000e+00
max             8.500000e+01  4.231000e+03  3.595000e+03

--------------------------------------------------------------------------

Correlation analysis: POS_CASH_balance
                       SK_ID_PREV  SK_ID_CURR  MONTHS_BALANCE  CNT_INSTALMENT  \
SK_ID_PREV               1.000000   -0.000336        0.001835        0.003820
SK_ID_CURR              -0.000336    1.000000        0.000404        0.000144
MONTHS_BALANCE           0.001835    0.000404        1.000000        0.336163
CNT_INSTALMENT           0.003820    0.000144        0.336163        1.000000
CNT_INSTALMENT_FUTURE    0.003679   -0.000559        0.271595        0.871276
SK_DPD                  -0.000487    0.003118       -0.018939       -0.060803
SK_DPD_DEF               0.004848    0.001948       -0.000381       -0.014154

                       CNT_INSTALMENT_FUTURE    SK_DPD  SK_DPD_DEF
SK_ID_PREV                          0.003679 -0.000487    0.004848
SK_ID_CURR                         -0.000559  0.003118    0.001948
MONTHS_BALANCE                      0.271595 -0.018939   -0.000381
CNT_INSTALMENT                      0.871276 -0.060803   -0.014154
CNT_INSTALMENT_FUTURE               1.000000 -0.082004   -0.017436
SK_DPD                             -0.082004  1.000000    0.245782
SK_DPD_DEF                         -0.017436  0.245782    1.000000

--------------------------------------------------------------------------

Other Analysis: POS_CASH_balance
1. Checking for Null values: POS_CASH_balance
SK_ID_PREV                   0
SK_ID_CURR                   0
MONTHS_BALANCE               0
CNT_INSTALMENT           26071
CNT_INSTALMENT_FUTURE    26087
NAME_CONTRACT_STATUS         0
SK_DPD                       0
SK_DPD_DEF                   0
dtype: int64

2. Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10001358 entries, 0 to 10001357
Data columns (total 8 columns):
 #   Column                 Dtype
---  ------                 -----
 0   SK_ID_PREV             int64
 1   SK_ID_CURR             int64
 2   MONTHS_BALANCE         int64
 3   CNT_INSTALMENT         float64
 4   CNT_INSTALMENT_FUTURE  float64
 5   NAME_CONTRACT_STATUS   object
 6   SK_DPD                 int64
 7   SK_DPD_DEF             int64
dtypes: float64(2), int64(5), object(1)
memory usage: 610.4+ MB
None
```

In [18]:

```
males = application_train[application_train['CODE_GENDER']=='M']['TARGET'].value_counts().
males['count_percent'] = males['user_count']/males['user_count'].sum()*100
```

```
males['CODE_GENDER'] = 'M'
females = application_train[application_train['CODE_GENDER']=='F']['TARGET'].value_counts
females['count_percent'] = females['user_count']/females['user_count'].sum()*100
females['CODE_GENDER'] = 'F'
gender_data = males.append(females, ignore_index=True,sort=False)
gender_data
```

Out[18]:

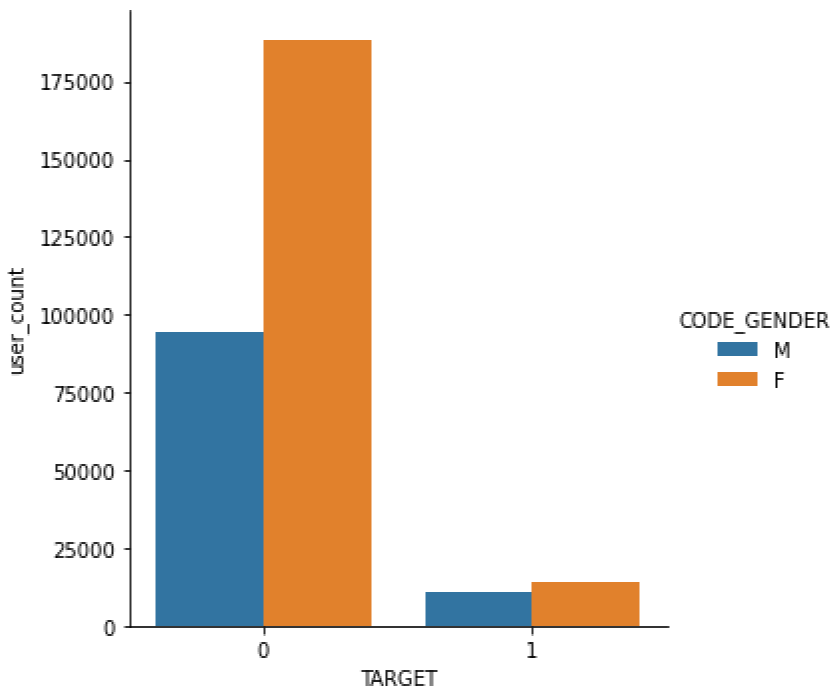
	TARGET	user_count	count_percent	CODE_GENDER
0	0	94404	89.858080	M
1	1	10655	10.141920	M
2	0	188278	93.000672	F
3	1	14170	6.999328	F

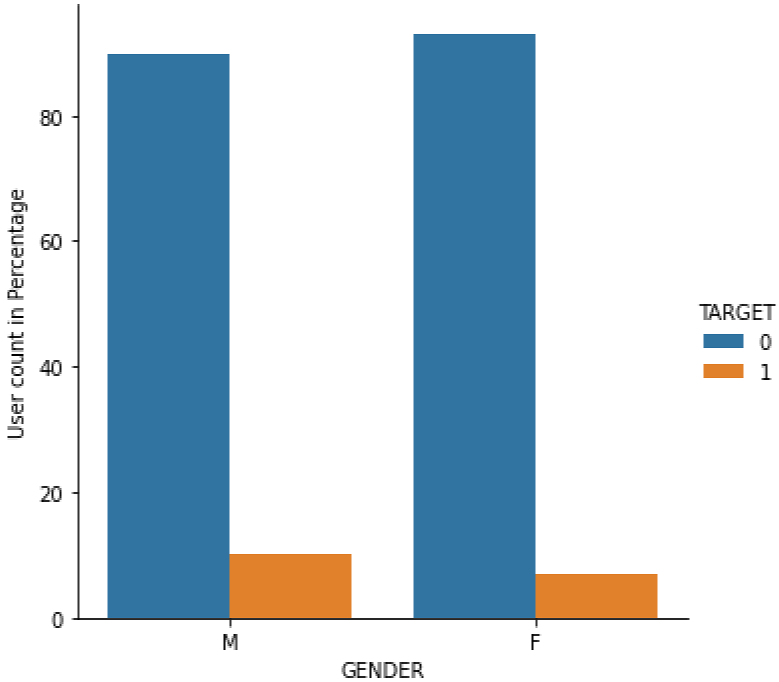
In [19]:

```
sea.catplot(data=gender_data, kind="bar", x="TARGET", y="user_count", hue="CODE_GENDER")
sea.catplot(data=gender_data, kind="bar", x="CODE_GENDER", y="count_percent", hue="TARGET"
plt.xlabel("GENDER")
plt.ylabel('User count in Percentage')
```

Out[19]:

```
Text(27.075538194444448, 0.5, 'User count in Percentage')
```





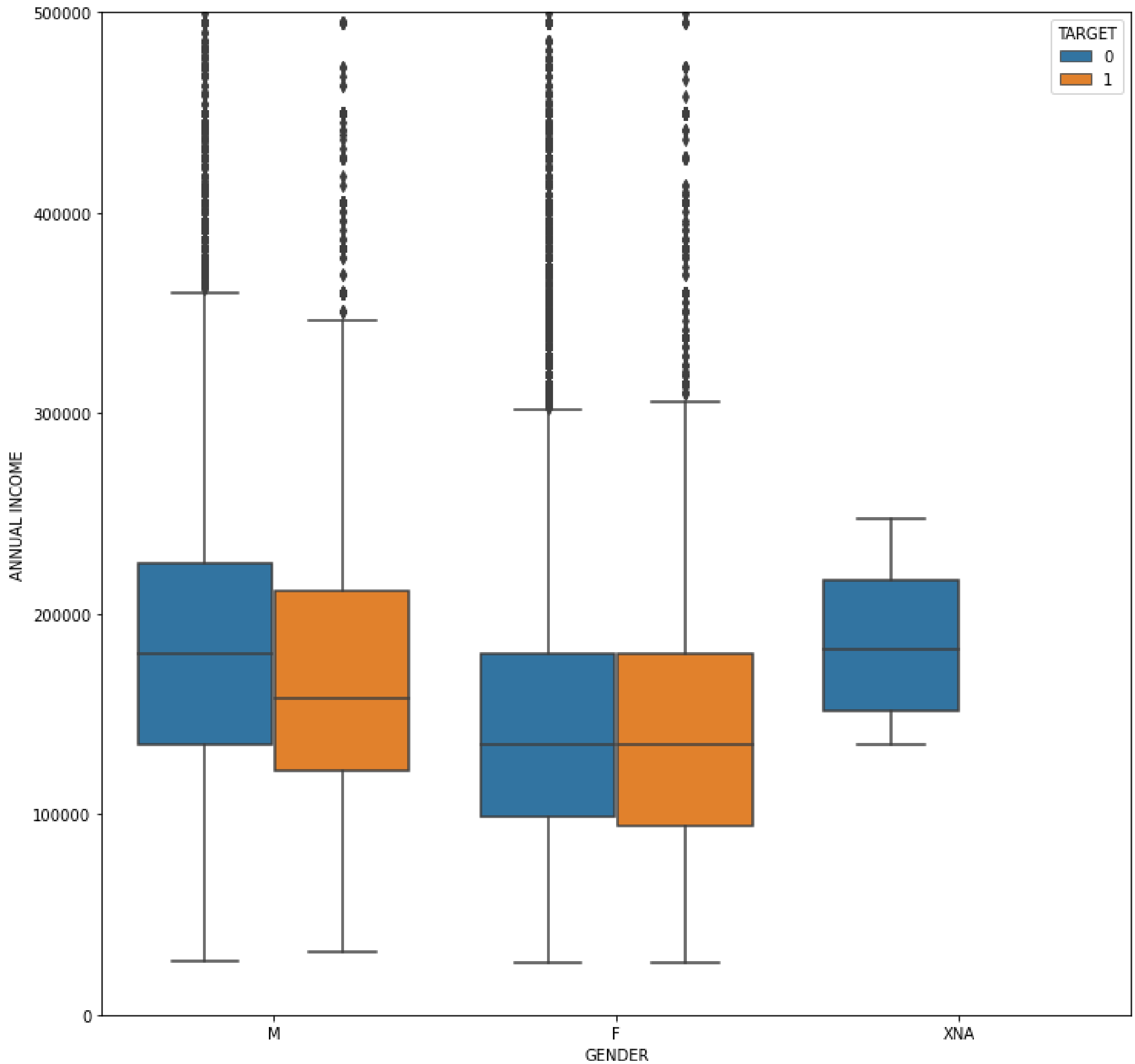
GENDER Vs INCOME based on Target

In [20]:

```
figure,ax = plt.subplots(figsize = (12,12))
sea.boxplot(x='CODE_GENDER',hue = 'TARGET',y='AMT_INCOME_TOTAL', data=application_train)
plt.ylim(0, 500000)
plt.xlabel("GENDER")
plt.ylabel('ANNUAL INCOME')
```

Out[20]:

```
Text(0, 0.5, 'ANNUAL INCOME')
```



OWN HOUSE COUNT based on Target

In [21]:

```
own_house = application_train[application_train['FLAG_OWN_REALTY']=='Y']['TARGET'].value_c
own_house['OWN_HOUSE'] = 'Y'
own_house['count_percent'] = own_house['user_count']/own_house['user_count'].sum()*100
not_own_house = application_train[application_train['FLAG_OWN_REALTY']=='N']['TARGET'].val
not_own_house['OWN_HOUSE'] = 'N'
not_own_house['count_percent'] = not_own_house['user_count']/not_own_house['user_count'].s
own_house = own_house.append(not_own_house,ignore_index=True,sort=False)
own_house
```

Out[21]:

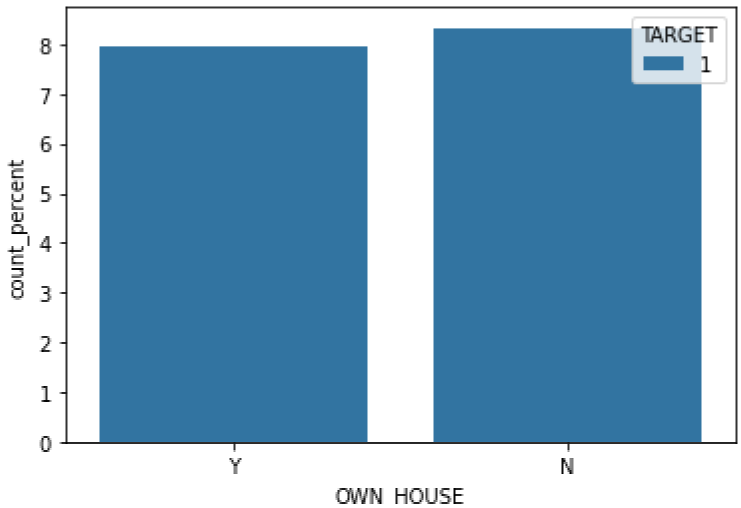
	TARGET	user_count	OWN_HOUSE	count_percent
0	0	196329	Y	92.038423
1	1	16983	Y	7.961577
2	0	86357	N	91.675071
3	1	7842	N	8.324929

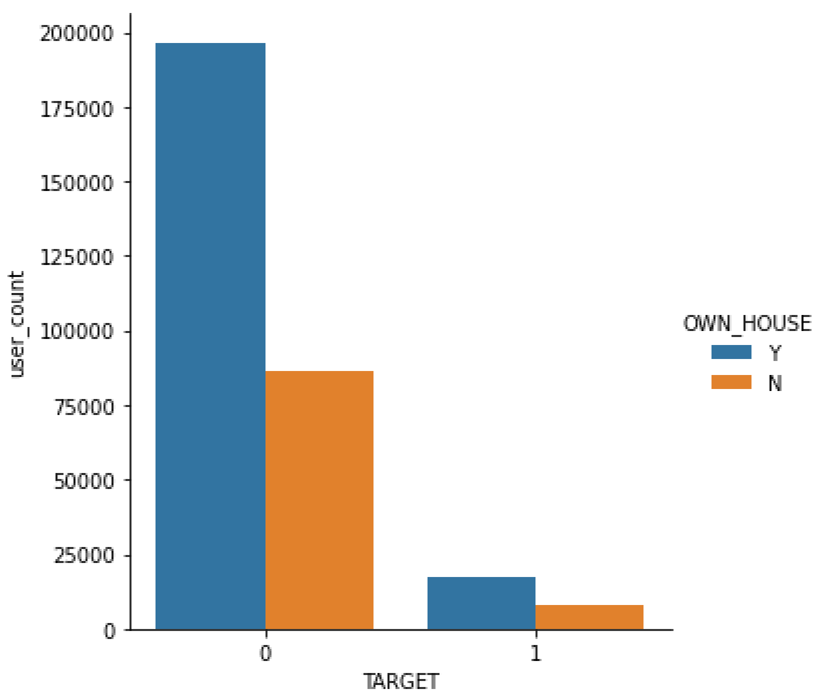
In [22]:

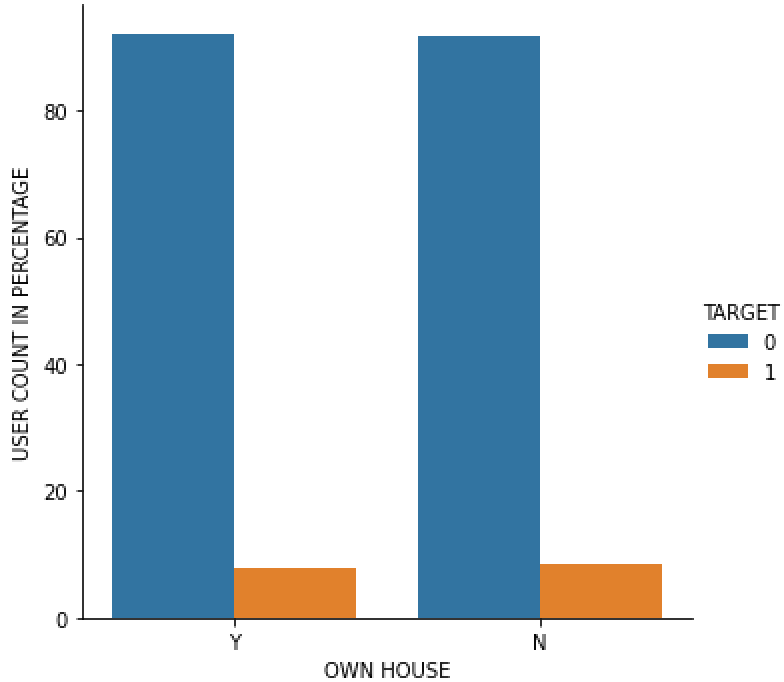
```
sea.barplot(x='OWN_HOUSE',y='count_percent',hue = 'TARGET',data=own_house[own_house['TARGE
sea.catplot(data=own_house, kind="bar", x="TARGET", y="user_count", hue="OWN_HOUSE")

sea.catplot(data=own_house, kind="bar", x="OWN_HOUSE", y="count_percent", hue="TARGET")
plt.xlabel("OWN HOUSE")
plt.ylabel('USER COUNT IN PERCENTAGE')
```

Out[22]:

```
Text(27.075538194444448, 0.5, 'USER COUNT IN PERCENTAGE')
```







OWN CAR COUNT based on Target

In [23]:

```
own_car = application_train[application_train['FLAG_OWN_CAR']=='Y']['TARGET'].value_counts
own_car['FLAG_OWN_CAR'] = 'Y'
own_car['count_percent'] = own_car['user_count']/own_car['user_count'].sum()*100
not_own_car = application_train[application_train['FLAG_OWN_CAR']=='N']['TARGET'].value_co
not_own_car['FLAG_OWN_CAR'] = 'N'
not_own_car['count_percent'] = not_own_car['user_count']/not_own_car['user_count'].sum()*1
own_car = own_car.append(not_own_car,ignore_index=True,sort=False)
own_car
```

Out[23]:

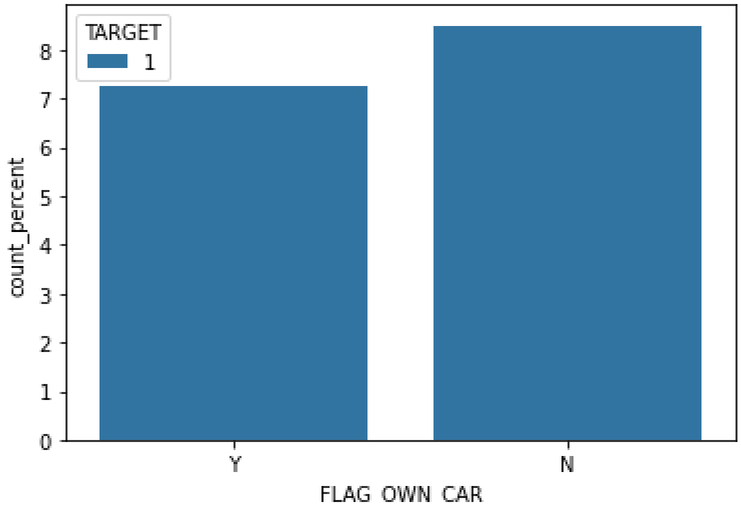
	TARGET	user_count	FLAG_OWN_CAR	count_percent
0	0	97011	Y	92.756270
1	1	7576	Y	7.243730
2	0	185675	N	91.499773
3	1	17249	N	8.500227

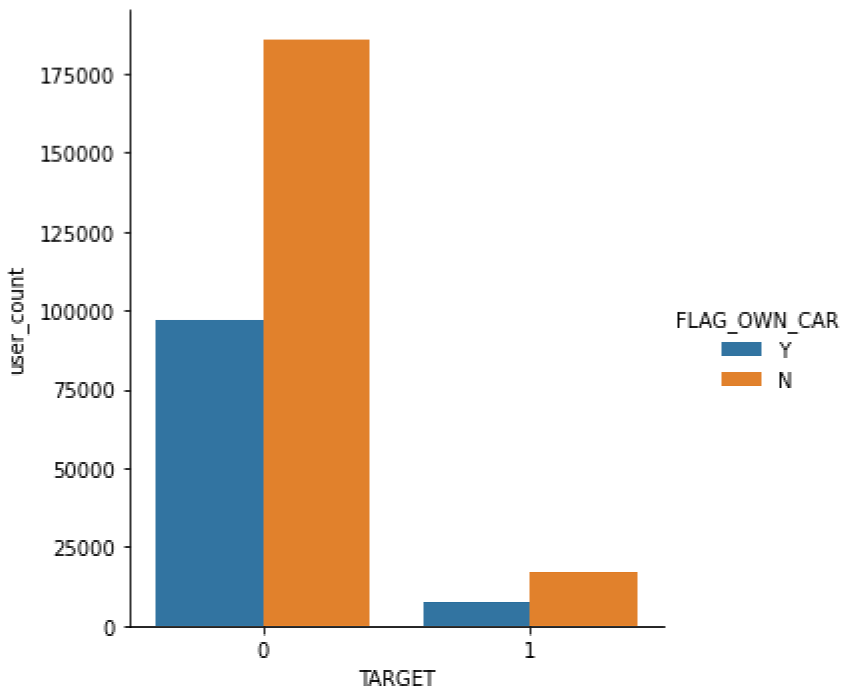
In [24]:

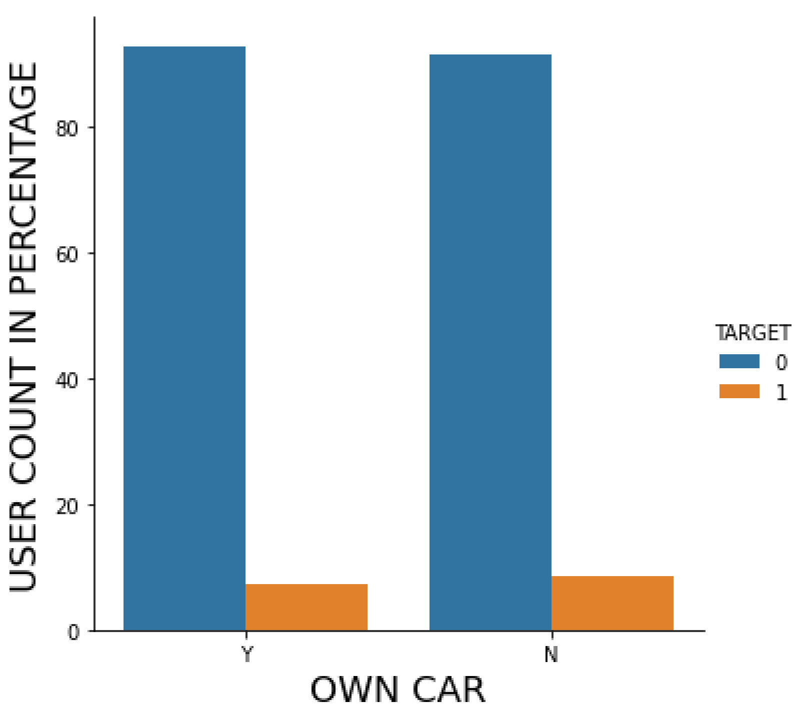
```
sea.barplot(x='FLAG_OWN_CAR',y='count_percent',hue = 'TARGET',data=own_car[own_car['TARGET

sea.catplot(data=own_car, kind="bar", x="TARGET", y="user_count", hue="FLAG_OWN_CAR")

sea.catplot(data=own_car, kind="bar", x="FLAG_OWN_CAR", y="count_percent", hue="TARGET")
plt.xlabel("OWN CAR",fontsize = 18)
plt.ylabel('USER COUNT IN PERCENTAGE',fontsize = 18)
```

Out[24]:

```
Text(27.075538194444448, 0.5, 'USER COUNT IN PERCENTAGE')
```







BORROWER OWNING A CAR are more likely to Pay

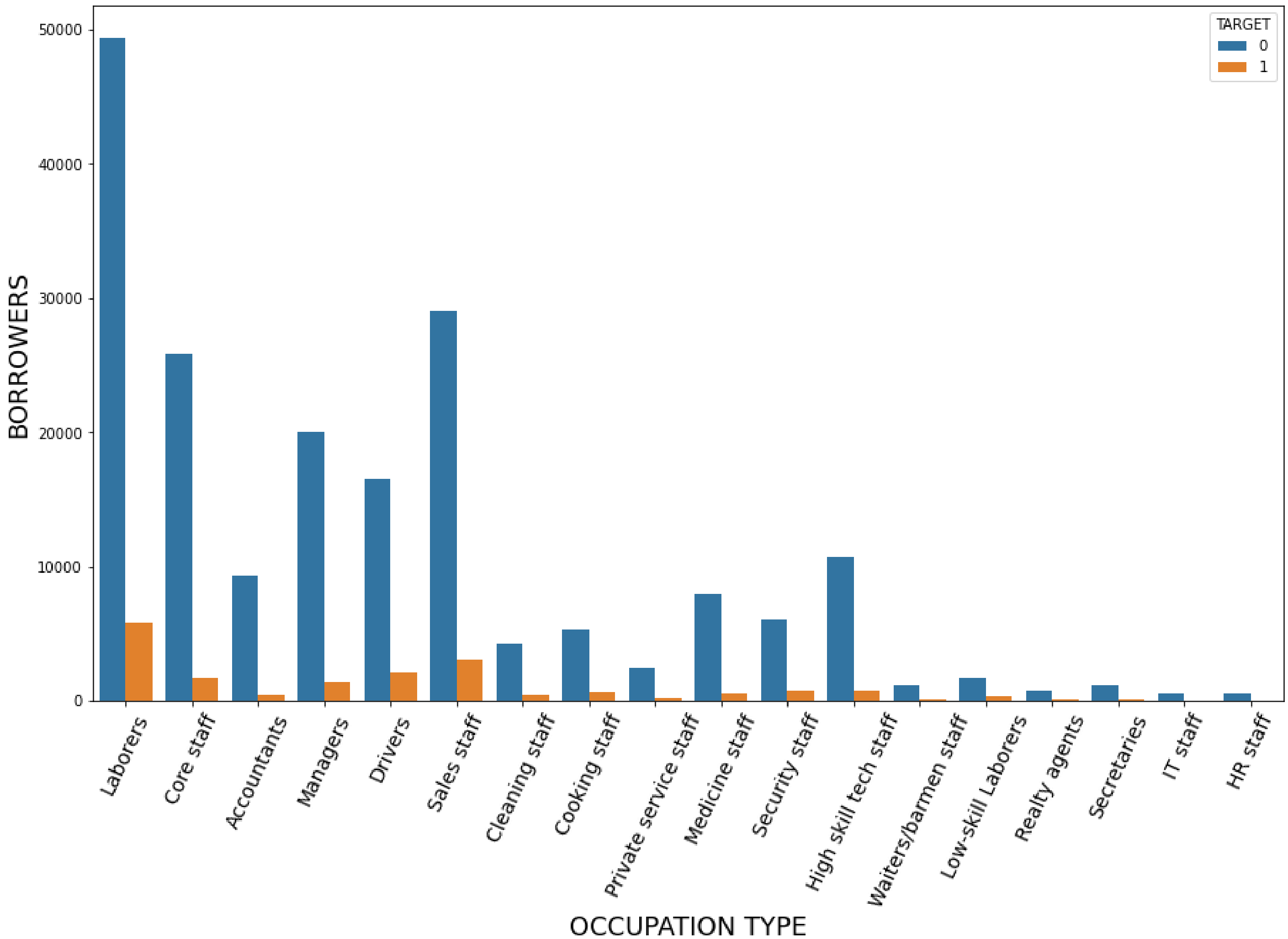
OCCUPATION TYPE COUNT based on Target

In [25]:

```
fig, ax = plt.subplots(figsize=(15,9))
sea.countplot(x='OCCUPATION_TYPE', hue = 'TARGET',data=application_train)
plt.xlabel("OCCUPATION TYPE",fontsize = 18)
plt.ylabel('BORROWERS',fontsize = 18)
plt.xticks(fontsize=14, rotation=65)
```

Out[25]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17]),
 [Text(0, 0, 'Laborers'),
  Text(1, 0, 'Core staff'),
  Text(2, 0, 'Accountants'),
  Text(3, 0, 'Managers'),
  Text(4, 0, 'Drivers'),
  Text(5, 0, 'Sales staff'),
  Text(6, 0, 'Cleaning staff'),
  Text(7, 0, 'Cooking staff'),
  Text(8, 0, 'Private service staff'),
  Text(9, 0, 'Medicine staff'),
  Text(10, 0, 'Security staff'),
  Text(11, 0, 'High skill tech staff'),
  Text(12, 0, 'Waiters/barmen staff'),
  Text(13, 0, 'Low-skill Laborers'),
  Text(14, 0, 'Realty agents'),
  Text(15, 0, 'Secretaries'),
  Text(16, 0, 'IT staff'),
  Text(17, 0, 'HR staff')])
```



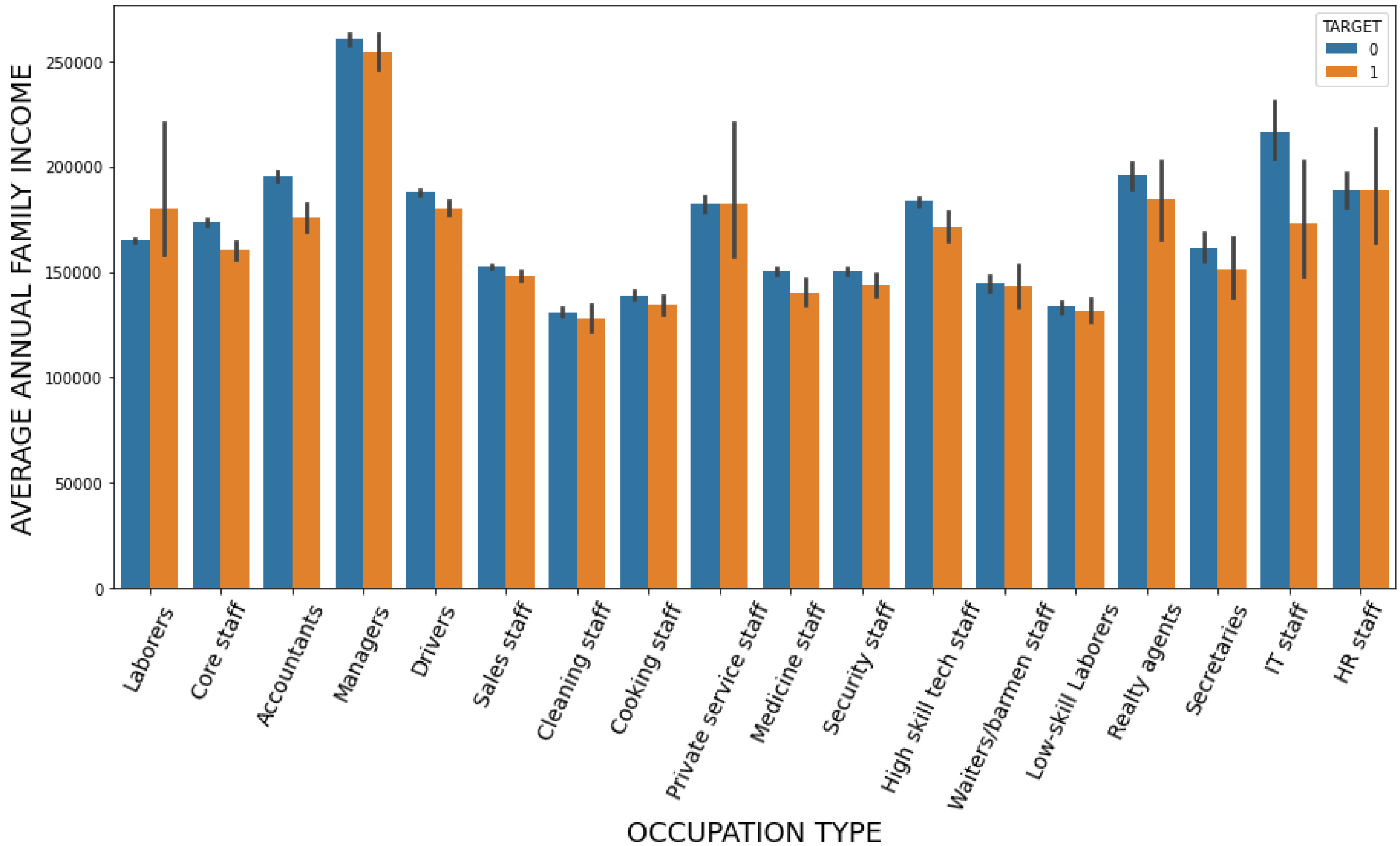
OCCUPATION TYPE vs INCOME based on Target

In [26]:

```
fig, ax = plt.subplots(figsize=(15,7))
sea.barplot(x='OCCUPATION_TYPE',y='AMT_INCOME_TOTAL',hue = 'TARGET',data=application_trair
plt.xticks(rotation=65,fontsize = 14)
plt.xlabel("OCCUPATION TYPE",fontsize = 18)
plt.ylabel("AVERAGE ANNUAL FAMILY INCOME",fontsize = 18)
```

Out[26]:

```
Text(0, 0.5, 'AVERAGE ANNUAL FAMILY INCOME')
```



In [27]:

```
income_credit_ratio_data = application_train[['AMT_INCOME_TOTAL','AMT_CREDIT','TARGET']]
income_credit_ratio_data['IC_ratio'] = income_credit_ratio_data['AMT_INCOME_TOTAL']/income
income_credit_ratio_data['quantile'] = pd.qcut(income_credit_ratio_data['IC_ratio'],q = 10
income_credit_ratio_data
```

Out[27]:

	AMT_INCOME_TOTAL	AMT_CREDIT	TARGET	IC_ratio	quantile
0	202500.0	406597.5	1	0.498036	7
1	270000.0	1293502.5	0	0.208736	2
2	67500.0	135000.0	0	0.500000	7
3	135000.0	312682.5	0	0.431748	6
4	121500.0	513000.0	0	0.236842	3
...	...	...	...	...	...
307506	157500.0	254700.0	0	0.618375	8
307507	72000.0	269550.0	0	0.267112	4
307508	153000.0	677664.0	0	0.225776	3
307509	171000.0	370107.0	1	0.462029	7
307510	157500.0	675000.0	0	0.233333	3

307511 rows × 5 columns

In [28]:

```
income_credit_ratio_data = income_credit_ratio_data.groupby(['quantile','TARGET'])['AMT_IN
income_credit_ratio_data['count_percent'] = income_credit_ratio_data.apply(lambda x: x['us
income_credit_ratio_data
```

Out[28]:

quantile	TARGET	user_count	count_percent

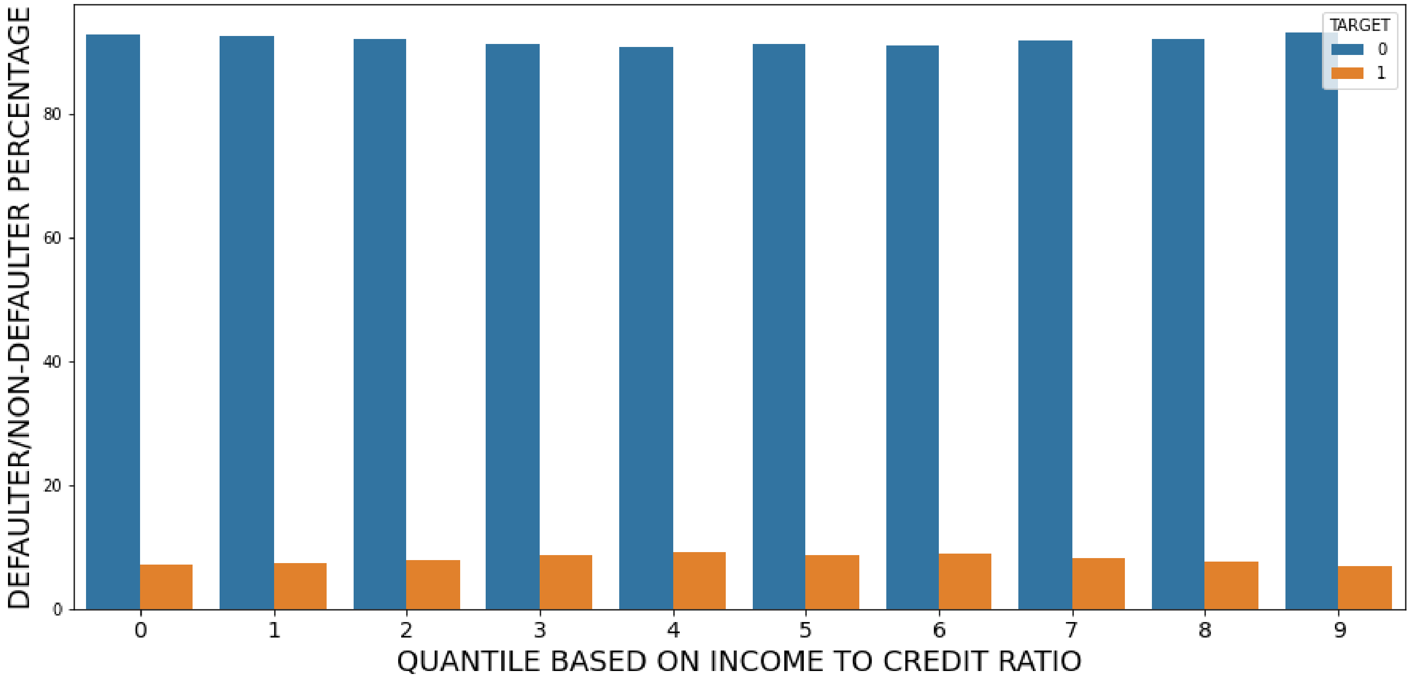
	quantile	TARGET	user_count	count_percent
0	0	0	28613	92.929523
1	0	1	2177	7.070477
2	1	0	28499	92.577313
3	1	1	2285	7.422687
4	2	0	28241	92.035196
5	2	1	2444	7.964804
6	3	0	28128	91.375110
7	3	1	2655	8.624890
8	4	0	27899	90.805234
9	4	1	2825	9.194766
10	5	0	28298	91.307434
11	5	1	2694	8.692566
12	6	0	27764	91.023539
13	6	1	2738	8.976461
14	7	0	28498	91.863839
15	7	1	2524	8.136161
16	8	0	28126	92.264795
17	8	1	2358	7.735205
18	9	0	28620	93.088307
19	9	1	2125	6.911693

In [29]:

```
fig, ax = plt.subplots(figsize=(15,7))
sea.barplot(x='quantile',y='count_percent',hue = 'TARGET',data=income_credit_ratio_data)
plt.xticks(rotation=0,fontsize = 14)
plt.xlabel("QUANTILE BASED ON INCOME TO CREDIT RATIO",fontsize = 18)
plt.ylabel("DEFAULTER/NON-DEFAULTER PERCENTAGE",fontsize = 18)
```

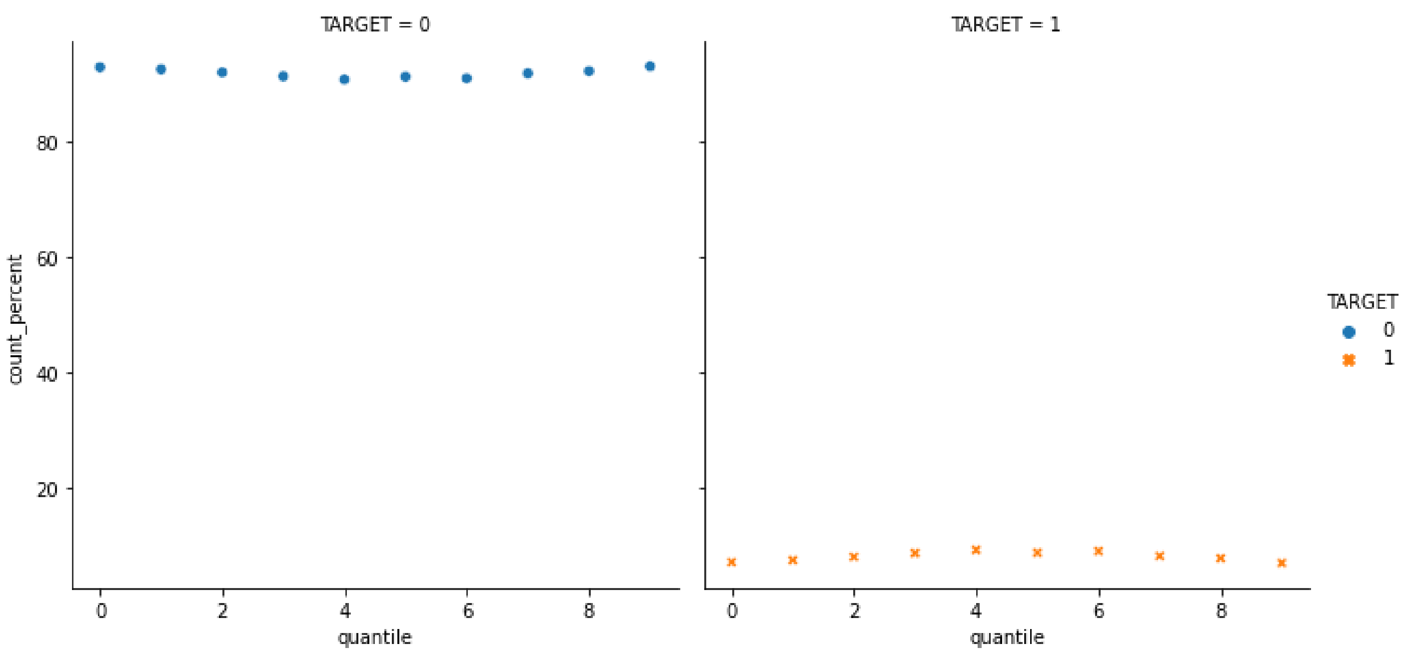
Out[29]:

```
Text(0, 0.5, 'DEFAULTER/NON-DEFAULTER PERCENTAGE')
```



In [30]:
```
sns.relplot(
    data=income_credit_ratio_data, x="quantile", y="count_percent",
    col="TARGET", hue="TARGET", style="TARGET",
    kind="scatter"
)
```

Out[30]: <seaborn.axisgrid.FacetGrid at 0x1db047eff10>



Defaulter Percentage is less than IC_ratiois either low or High

--

REPAYERS TO APPLICATION RATIO

In [31]:
```
occ_data = pd.DataFrame(data=application_train.groupby(['OCCUPATION_TYPE','TARGET']).count
occ_data = occ_data.reset_index()
value_counts = occ_data['SK_ID_CURR'].values
def repayers_to_applicants_ratio(values):
    flag = 1
    ratios = []
    for count in range(len(values)):
        if flag == 1:
            current_number = values[count]
            next_number = values[count+1]
            ratios.append(current_number/(current_number+next_number))
            ratios.append(current_number/(current_number+next_number))
        flag=flag*-1
    return ratios
occ_data['Ratio R/A'] = repayers_to_applicants_ratio(value_counts)
occ_ratio = occ_data.groupby(['OCCUPATION_TYPE','Ratio R/A']).count().drop(['TARGET', 'SK_
occ_ratio = occ_ratio.reset_index()
occ_ratio = occ_ratio.sort_values(['Ratio R/A'],ascending=False)
occ_ratio
```

Out[31]:

	OCCUPATION_TYPE	Ratio R/A
0	Accountants	0.951697
6	High skill tech staff	0.938401
10	Managers	0.937860
3	Core staff	0.936960
5	HR staff	0.936057
7	IT staff	0.935361
12	Private service staff	0.934012
11	Medicine staff	0.932998
15	Secretaries	0.929502
13	Realty agents	0.921438
1	Cleaning staff	0.903933
14	Sales staff	0.903682
2	Cooking staff	0.895560
8	Laborers	0.894212
16	Security staff	0.892576
17	Waiters/barmen staff	0.887240
4	Drivers	0.886739
9	Low-skill Laborers	0.828476

CORRELATION OF POSITIVE DAYS SINCE BIRTH AND TARGET

In [32]:
```
application_train['DAYS_BIRTH'] = abs(application_train['DAYS_BIRTH'])
-1*(application_train['DAYS_BIRTH'].corr(application_train['TARGET']))
```

Out[32]:
```
0.07823930830984513
```

CORRELATION OF POSITIVE DAYS SINCE EMPLOYMENT AND TARGET

In [33]:

```
application_train['DAYS_EMPLOYED'] = abs(application_train['DAYS_EMPLOYED'])
-1*(application_train['DAYS_EMPLOYED'].corr(application_train['TARGET']))
```

Out[33]:

```
0.04704582521599873
```

FETCHING IMPORTANT RELAVENT FEATURES

In [34]:

```
imp_features = ['FLOORSMAX_MEDI', 'ELEVATORS_MEDI', 'AMT_GOODS_PRICE', 'EMERGENCYSTATE_MOI
imp_features = ['CODE_GENDER','FLAG_OWN_REALTY','FLAG_OWN_CAR','AMT_CREDIT','AMT_ANNUITY',
imp_features = list(set(imp_features))
```

In [35]:

```
experimentLog = pd.DataFrame(columns=["ExpID", "Cross fold train accuracy", "Test Accuracy
```

In [36]:

```
def rounding(x):
    return round(100*x,1)

class DataFrameSelector(BaseEstimator, TransformerMixin):
    def __init__(self, attribute_names):
        self.attribute_names = attribute_names
    def fit(self, X, y=None):
        return self
    def transform(self, X):
        return X[self.attribute_names].values

def LossBinaryClassifier(actual, predicted):
    return(-1/ len(actual)*(sum(actual * np.log(predicted) + (1 - actual) * np.log(1 - pre
```

In [37]:

```
null_value = X.isna().sum().reset_index().rename(columns={'index':'column_name',0:'null_va
null_value['count%'] = null_value['null_value_count']/len(X)*100
null_value = null_value[null_value['count%'] <= 30]
null_value
```

Out[37]:

	column_name	null_value_count	count%
0	NAME_CONTRACT_TYPE	0	0.000000
1	CODE_GENDER	0	0.000000
2	FLAG_OWN_CAR	0	0.000000
3	FLAG_OWN_REALTY	0	0.000000
4	CNT_CHILDREN	0	0.000000
...	...	...	...
115	AMT_REQ_CREDIT_BUREAU_DAY	41519	13.501631
116	AMT_REQ_CREDIT_BUREAU_WEEK	41519	13.501631
117	AMT_REQ_CREDIT_BUREAU_MON	41519	13.501631
118	AMT_REQ_CREDIT_BUREAU_QRT	41519	13.501631

	column_name	null_value_count	count%
119	AMT_REQ_CREDIT_BUREAU_YEAR	41519	13.501631

70 rows × 3 columns

In [38]:

```
selected_features = null_value['column_name'].tolist() + ['TARGET']
print(selected_features)
```

```
['NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN',
'AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE', 'NA
ME_INCOME_TYPE', 'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'REGION
_POPULATION_RELATIVE', 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLIS
H', 'FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE', 'FLAG_PHONE',
'FLAG_EMAIL', 'CNT_FAM_MEMBERS', 'REGION_RATING_CLIENT', 'REGION_RATING_CLIENT_W_CITY', 'W
EEKDAY_APPR_PROCESS_START', 'HOUR_APPR_PROCESS_START', 'REG_REGION_NOT_LIVE_REGION', 'REG_
REGION_NOT_WORK_REGION', 'LIVE_REGION_NOT_WORK_REGION', 'REG_CITY_NOT_LIVE_CITY', 'REG_CIT
Y_NOT_WORK_CITY', 'LIVE_CITY_NOT_WORK_CITY', 'ORGANIZATION_TYPE', 'EXT_SOURCE_2', 'EXT_SOU
RCE_3', 'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF_30_CNT_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCL
E', 'DEF_60_CNT_SOCIAL_CIRCLE', 'DAYS_LAST_PHONE_CHANGE', 'FLAG_DOCUMENT_2', 'FLAG_DOCUMEN
T_3', 'FLAG_DOCUMENT_4', 'FLAG_DOCUMENT_5', 'FLAG_DOCUMENT_6', 'FLAG_DOCUMENT_7', 'FLAG_DO
CUMENT_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_DOCU
MENT_17', 'FLAG_DOCUMENT_18', 'FLAG_DOCUMENT_19', 'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21',
'AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK',
'AMT_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR', 'T
ARGET']
```

In [39]:

```
null_value['column_type'] = null_value['column_name'].apply(lambda x: X[x].dtype)
null_value[null_value['count%'] > 0]
```

Out[39]:

	column_name	null_value_count	count%	column_type
7	AMT_ANNUITY	12	0.003902	float64
8	AMT_GOODS_PRICE	278	0.090403	float64
9	NAME_TYPE_SUITE	1292	0.420148	object
27	CNT_FAM_MEMBERS	2	0.000650	float64
40	EXT_SOURCE_2	660	0.214626	float64
41	EXT_SOURCE_3	60965	19.825307	float64
89	OBS_30_CNT_SOCIAL_CIRCLE	1021	0.332021	float64
90	DEF_30_CNT_SOCIAL_CIRCLE	1021	0.332021	float64
91	OBS_60_CNT_SOCIAL_CIRCLE	1021	0.332021	float64
92	DEF_60_CNT_SOCIAL_CIRCLE	1021	0.332021	float64
93	DAYS_LAST_PHONE_CHANGE	1	0.000325	float64
114	AMT_REQ_CREDIT_BUREAU_HOUR	41519	13.501631	float64
115	AMT_REQ_CREDIT_BUREAU_DAY	41519	13.501631	float64
116	AMT_REQ_CREDIT_BUREAU_WEEK	41519	13.501631	float64
117	AMT_REQ_CREDIT_BUREAU_MON	41519	13.501631	float64
118	AMT_REQ_CREDIT_BUREAU_QRT	41519	13.501631	float64
119	AMT_REQ_CREDIT_BUREAU_YEAR	41519	13.501631	float64

In [40]:
```
X_feature = data[selected_features]
X_feature['NAME_TYPE_SUITE'].fillna('Other_C', inplace=True)
X_feature.head()
```

Out[40]:

	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOT
0	Cash loans	M	N	Y	0	20250
1	Cash loans	F	N	N	0	27000
2	Revolving loans	M	Y	Y	0	6750
3	Cash loans	F	N	Y	0	13500
4	Cash loans	M	N	Y	0	12150

5 rows × 71 columns

In [41]:
```
temp_columns = null_value[null_value['null_value_count'] != 0].reset_index(drop=True)['col
for col in temp_columns:
    if 'AMT_REQ_CREDIT' in col:
        print("columns to be filled with 0 is: {}".format(col))
        X_feature[col].fillna(0,inplace=True)
```

```
columns to be filled with 0 is: AMT_REQ_CREDIT_BUREAU_HOUR
columns to be filled with 0 is: AMT_REQ_CREDIT_BUREAU_DAY
columns to be filled with 0 is: AMT_REQ_CREDIT_BUREAU_WEEK
columns to be filled with 0 is: AMT_REQ_CREDIT_BUREAU_MON
columns to be filled with 0 is: AMT_REQ_CREDIT_BUREAU_QRT
columns to be filled with 0 is: AMT_REQ_CREDIT_BUREAU_YEAR
```

In [42]:
```
for col in temp_columns:
    if 'CNT_SOCIAL_CIRCLE' in col:
        print("columns to be filled with 0 is: {}".format(col))
        X_feature[col].fillna(0,inplace=True)
```

```
columns to be filled with 0 is: OBS_30_CNT_SOCIAL_CIRCLE
columns to be filled with 0 is: DEF_30_CNT_SOCIAL_CIRCLE
columns to be filled with 0 is: OBS_60_CNT_SOCIAL_CIRCLE
columns to be filled with 0 is: DEF_60_CNT_SOCIAL_CIRCLE
```

In [43]:
```
for col in temp_columns:
    if 'CNT_FAM_MEMBERS' in col:
        print("columns to be filled with median is: {}".format(col))
        X_feature[col].fillna(X_feature[col].median(),inplace=True)
```

```
columns to be filled with median is: CNT_FAM_MEMBERS
```

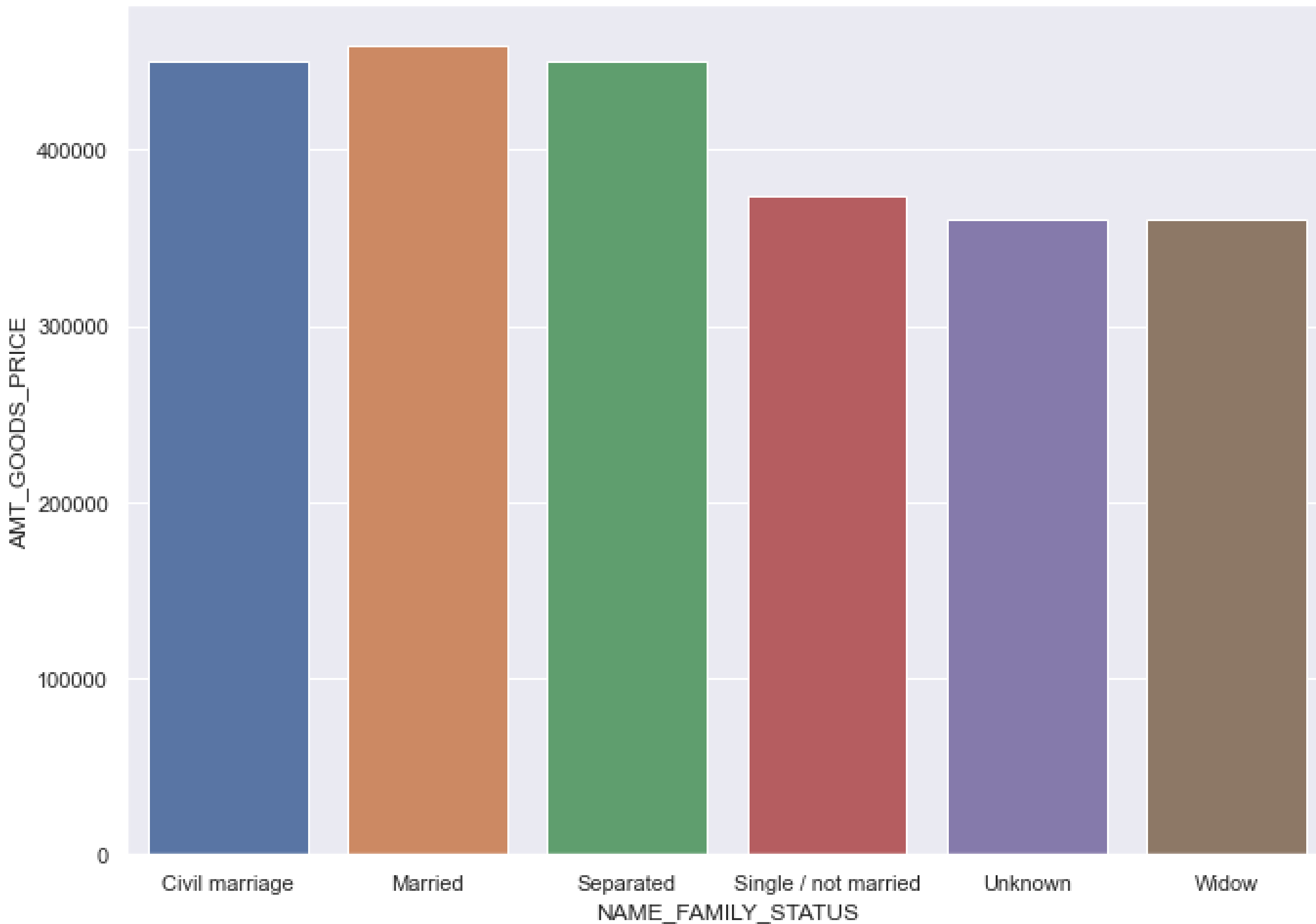
In [44]:
```
temp_vis = X_feature[['AMT_GOODS_PRICE','NAME_FAMILY_STATUS']]
temp_vis = temp_vis.groupby('NAME_FAMILY_STATUS')['AMT_GOODS_PRICE'].median().reset_index
temp_vis['AMT_GOODS_PRICE'] = temp_vis['AMT_GOODS_PRICE'].fillna(temp_vis['AMT_GOODS_PRICE
temp_vis.head()
```

Out[44]:

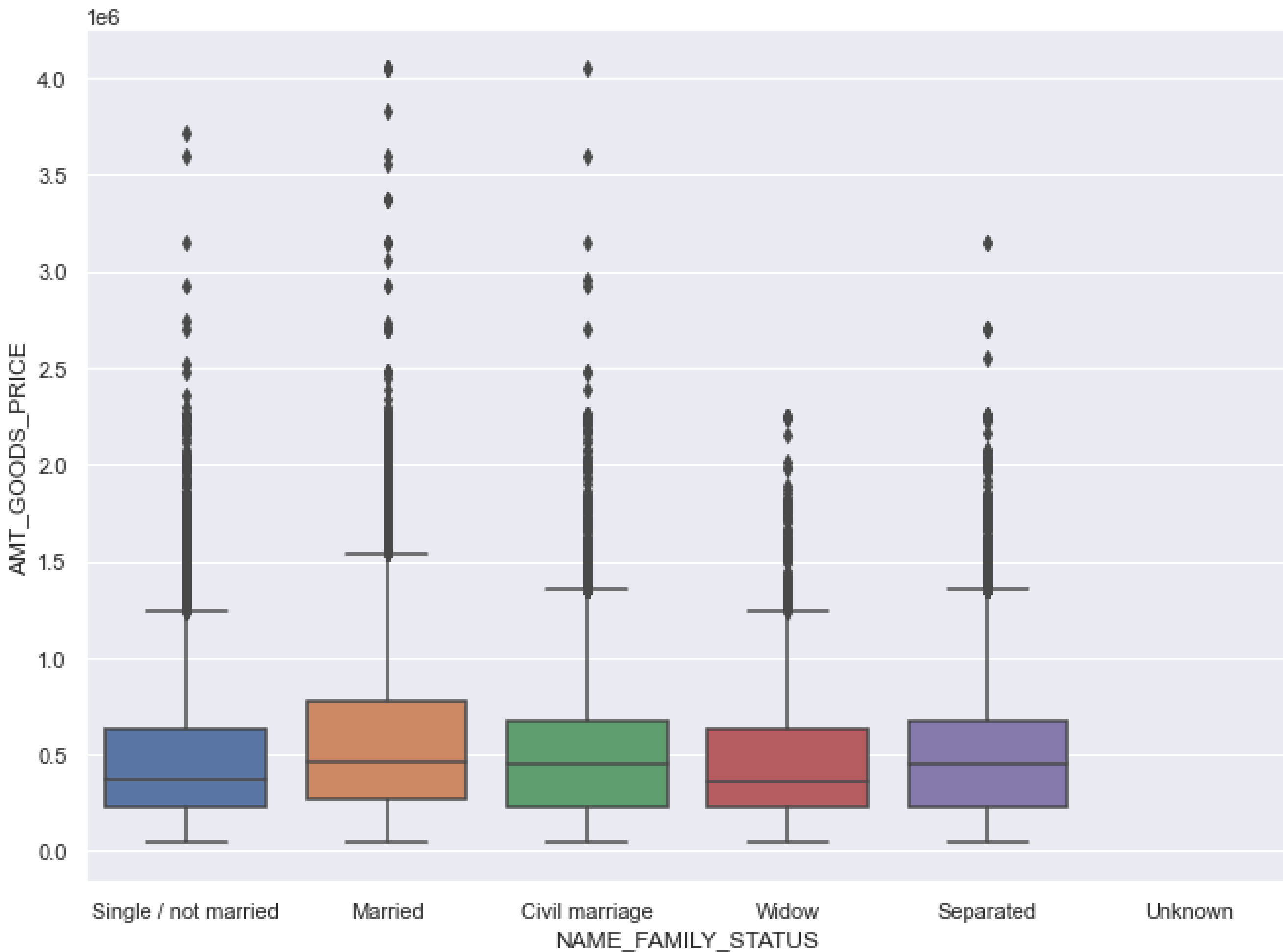
	NAME_FAMILY_STATUS	AMT_GOODS_PRICE
0	Civil marriage	450000.0
1	Married	459000.0
2	Separated	450000.0

	NAME_FAMILY_STATUS	AMT_GOODS_PRICE
3	Single / not married	373500.0
4	Unknown	360000.0

In [45]:
```
sns.set(rc={'figure.figsize':(11,8)})
ax = sns.barplot(x="NAME_FAMILY_STATUS", y="AMT_GOODS_PRICE", data=temp_vis)
```



In [46]:
```
sns.set(rc={'figure.figsize':(11,8)})
ax = sns.boxplot(x="NAME_FAMILY_STATUS", y="AMT_GOODS_PRICE", data=X_feature)
```



In [47]:

```
def correct_cat_val(c):
    if c['AMT_GOODS_PRICE'] == np.inf:
        return temp_vis[temp_vis['NAME_FAMILY_STATUS']==c['NAME_FAMILY_STATUS']]['AMT_GOOI
    else:
        return c['AMT_GOODS_PRICE']

for col in temp_columns:
    X_feature['AMT_GOODS_PRICE'] = X_feature['AMT_GOODS_PRICE'].fillna(np.inf)
    if 'AMT_GOODS_PRICE' in col:
        print("columns to be filled with category median is: {}".format(col))
        X_feature['AMT_GOODS_PRICE'] = X_feature.apply(lambda c: correct_cat_val(c),axis=1
```

```
columns to be filled with category median is: AMT_GOODS_PRICE
```

In [48]:

```
X_feature.dropna(subset=['DAYS_LAST_PHONE_CHANGE'], inplace=True)
```

In [49]:

```
X_feature.dropna(subset=['AMT_ANNUITY'], inplace=True)
```

In [50]:

```
X_feature = X_feature.reset_index(drop=True)
```

In [51]:

```
from sklearn.preprocessing import LabelEncoder

correlation_df = pd.DataFrame()

for col in X_feature.columns.tolist():
    if X_feature[col].dtype == 'int':
        l = [col, X_feature['EXT_SOURCE_2'].corr(X_feature[col])]
    else:
```

```
        l = [col, X_feature['EXT_SOURCE_2'].corr(pd.DataFrame(LabelEncoder().fit_transforr
    correlation_df = correlation_df.append(pd.Series(l),ignore_index=True)
correlation_df = correlation_df.rename(columns={0:'col_name',1:'correlation_with_EXT_2'})
correlation_df['correlation_with_EXT_2'] = abs(correlation_df['correlation_with_EXT_2'])
correlation_df.sort_values(by='correlation_with_EXT_2',ascending=False).head(6).tail(5)
```

Out[51]:

	col_name	correlation_with_EXT_2
26	REGION_RATING_CLIENT	0.292903
27	REGION_RATING_CLIENT_W_CITY	0.288306
43	DAYS_LAST_PHONE_CHANGE	0.195766
5	AMT_INCOME_TOTAL	0.170547
70	TARGET	0.160471

In [52]:

```
region_rating_client = X_feature.groupby('REGION_RATING_CLIENT')['EXT_SOURCE_2'].median().

def correct_ext_source_2(e):
    if e['EXT_SOURCE_2'] == np.inf:
        return region_rating_client[region_rating_client['REGION_RATING_CLIENT']==e['REGIO
    else:
        return e['EXT_SOURCE_2']

X_feature['EXT_SOURCE_2'] = X_feature['EXT_SOURCE_2'].fillna(np.inf)
X_feature['EXT_SOURCE_2'] = X_feature.apply(lambda e: correct_ext_source_2(e),axis=1)
```

In [53]:

```
correlation_df = pd.DataFrame()

for col in X_feature.columns.tolist():
    if X_feature[col].dtype == 'int':
        l = [col, X_feature['EXT_SOURCE_3'].corr(X_feature[col])]
    else:
        l = [col, X_feature['EXT_SOURCE_3'].corr(pd.DataFrame(LabelEncoder().fit_transforr
    correlation_df = correlation_df.append(pd.Series(l),ignore_index=True)
correlation_df = correlation_df.rename(columns={0:'column_name',1:'correlation_with_EXT_3'
correlation_df['correlation_with_EXT_3'] = abs(correlation_df['correlation_with_EXT_3'])
correlation_df = correlation_df.sort_values(by='correlation_with_EXT_3',ascending=False).h
correlation_df
```

Out[53]:

	column_name	correlation_with_EXT_3
15	DAYS_BIRTH	0.205475
70	TARGET	0.178929
18	DAYS_ID_PUBLISH	0.131610
20	FLAG_EMP_PHONE	0.115284
37	EXT_SOURCE_2	0.109728
17	DAYS_REGISTRATION	0.107570
5	AMT_INCOME_TOTAL	0.088906
36	ORGANIZATION_TYPE	0.087994
34	REG_CITY_NOT_WORK_CITY	0.079706

In [54]:

```
ext_source_3 = X_feature[correlation_df['column_name'].tolist()+['EXT_SOURCE_3']]
```

```python
for col in ext_source_3.columns.tolist():
    if col != 'EXT_SOURCE_3':
        ext_source_3[col] = LabelEncoder().fit_transform(X_feature[[col]])

ext_source_3_train = ext_source_3[ext_source_3['EXT_SOURCE_3'].notnull()]
ext_source_3_test = ext_source_3[ext_source_3['EXT_SOURCE_3'].isnull()]
ext_source_3_train.shape, ext_source_3_test.shape
```

Out[54]:
```
((246535, 10), (60963, 10))
```

In [55]:
```python
ext_source_3_y_train = ext_source_3_train[['EXT_SOURCE_3']]
ext_source_3_X_train = ext_source_3_train.drop(columns=['EXT_SOURCE_3'])
ext_source_3_X_test = ext_source_3_test.drop(columns=['EXT_SOURCE_3'])
```

In [56]:
```python
from sklearn.linear_model import LinearRegression

model = LinearRegression().fit(ext_source_3_X_train, ext_source_3_y_train)
ext_source_3_y_pred = model.predict(ext_source_3_X_test)

ext_source_3_output = ext_source_3_X_test
ext_source_3_output['exs3_y'] = ext_source_3_y_pred
ext_source_3_output
```

Out[56]:

	DAYS_BIRTH	TARGET	DAYS_ID_PUBLISH	FLAG_EMP_PHONE	EXT_SOURCE_2	DAYS_REGISTRATION	AMT_IN
1	8382	0	5876	1	85079	14501	
3	6142	0	3730	1	90559	5854	
4	5215	0	2709	1	36021	11376	
9	10676	0	2175	1	110724	1373	
14	10562	0	4111	1	89028	15072	
...	...	...	...	...	...	...	
307471	12298	0	6132	1	109216	13156	
307488	12184	0	2387	1	76369	14289	
307491	8442	0	5908	1	68374	5889	
307493	15818	0	4185	1	96856	7231	
307494	4372	0	2077	0	11578	11299	

60963 rows × 10 columns

In [57]:
```python
ext_source_3_output = ext_source_3_output.reset_index().rename(columns={'index':'index_to_
for i in ext_source_3_output['index_to_be_updated'].tolist():
    X_feature['EXT_SOURCE_3'].iloc[i] = ext_source_3_output[ext_source_3_output['index_to_
```

In [58]:
```python
X_feature.isna().sum()
```

Out[58]:
```
NAME_CONTRACT_TYPE            0
CODE_GENDER                   0
FLAG_OWN_CAR                  0
FLAG_OWN_REALTY               0
CNT_CHILDREN                  0
```

```
                                 ..
AMT_REQ_CREDIT_BUREAU_WEEK    0
AMT_REQ_CREDIT_BUREAU_MON     0
AMT_REQ_CREDIT_BUREAU_QRT     0
AMT_REQ_CREDIT_BUREAU_YEAR    0
TARGET                        0
Length: 71, dtype: int64
```

In [59]:
```
X_feature['AMT_CREDIT_TO_ANNUITY_RATIO'] = X_feature['AMT_CREDIT'] / X_feature['AMT_ANNUIT
X_feature['Tot_EXTERNAL_SOURCE'] = X_feature['EXT_SOURCE_2'] + X_feature['EXT_SOURCE_3']
X_feature['Salary_to_credit'] = X_feature['AMT_INCOME_TOTAL']/X_feature['AMT_CREDIT']
X_feature['Annuity_to_salary_ratio'] = X_feature['AMT_ANNUITY']/X_feature['AMT_INCOME_TOTA
```

In [60]:
```
train = X_feature
train.shape
```

Out[60]:
```
(307498, 75)
```

In [61]:
```
features = train.columns.tolist()
features.remove('TARGET')
len(features)
```

Out[61]:
```
74
```

In [62]:
```
le_dict = {}

for col in features:
    if train[col].dtype == 'object':
        le = LabelEncoder()
        train[col] = le.fit_transform(train[col])
        le_dict['le_{}'.format(col)] = le
```

In [63]:
```
X = train[features]
y = train['TARGET']
```

In [64]:
```
experimentLog = pd.DataFrame(columns=["ExpID", "Cross fold train accuracy", "Test Accuracy
```

In [65]:
```
def hyperTunedLogReg(x, y, experimentLog, crossFold, choice):
    num_attribs = []
    cat_attribs = []

    for col in x.columns.tolist():
        if x[col].dtype in (['int','float']):
            num_attribs.append(col)
        else:
            cat_attribs.append(col)
    num_pipeline =Pipeline([('selector',DataFrameSelector(num_attribs)),
                           ('scaler', StandardScaler()),
                          ('imputer', SimpleImputer(strategy = 'median'))
                           ])

    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.15, random_state
    x_train, x_valid, y_train, y_valid = train_test_split(x_train, y_train, test_size=0.2,

    logistic = LogisticRegression(solver = 'saga', random_state = 42)
    tunedLogRegPipe = Pipeline(steps=[("logistic", logistic)])
    if choice == 1:
```

```
        param_grid = {
            "logistic__C": (10,1,0.1,0.01),
            "logistic__tol":(0.001,0.01,0.1)
        }
        logC = '(10,1,0.1,0.01)'
        logTolerance = '(0.001,0.01,0.1)'
    elif choice == 2:
        param_grid = {
            "logistic__C": (10,1,0.1,0.01,0.001,0.0001),
            "logistic__tol":(0.001,0.01,0.1)
        }
        logC = '(10,1,0.1,0.01,0.001,0.0001)'
        logTolerance = '(0.001,0.01,0.1)'
    else:
        param_grid = {
            "logistic__C": (10,1,0.1,0.01,0.001,0.0001),
            "logistic__tol":(0.0001,0.001,0.01,0.1)
        }
        logC = '(10,1,0.1,0.01,0.001,0.0001)'
        logTolerance = '(0.0001,0.001,0.01,0.1)'

    # Time and score test predictions
    start = time()

    tumedLogReg_search = GridSearchCV(tunedLogRegPipe, param_grid, cv = crossFold, n_jobs=
    tumedLogReg_search.fit(x_train, y_train)

    train_time = np.round(time() - start, 4)

    trainAcc  = tumedLogReg_search.score(x_train, y_train)
    validAcc  = tumedLogReg_search.score(x_valid, y_valid)
    start = time()
    testAcc  = tumedLogReg_search.score(x_test, y_test)
    test_time = np.round(time() - start, 4)
    AUC = roc_auc_score(y_test,tumedLogReg_search.predict(x_test))

    loss = log_loss(y_test,tumedLogReg_search.predict_proba(x_test))
    cnfs_mtrx = confusion_matrix(y_test, tumedLogReg_search.predict(x_test))
    denominator = cnfs_mtrx[0][0] + cnfs_mtrx[0][1] + cnfs_mtrx[1][0] + cnfs_mtrx[1][1]
    accuracy = ((cnfs_mtrx[0][0] + cnfs_mtrx[1][1]) / denominator) * 100

    input_count = x.shape[1]

    temp_df = pd.DataFrame()
    temp_df = temp_df.append(pd.Series(["Hypertuned Logisitc rgeression with {} inputs".fo
                  AUC, loss, accuracy, train_time, test_time,
                 "Hypertuned LogisticRegression with {} cv, logistic__C {} and logistic__to
    temp_df.columns = experimentLog.columns

    experimentLog = experimentLog.append(temp_df,ignore_index=True)
    return tumedLogReg_search, experimentLog
```

In [66]:

```
tumedLogReg_search, experimentLog = hyperTunedLogReg(X,y,experimentLog, 4, 2)
```

```
Fitting 4 folds for each of 18 candidates, totalling 72 fits
```

In [67]:

```
print(tumedLogReg_search.best_params_)
```

```
{'logistic__C': 10, 'logistic__tol': 0.001}
```

In [83]:

```
experimentLog
```

Out[83]:

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Parai
0	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	120.5867	0.0364	Hypertuned LogisticRegression with 4 cv, logis...	{'logis 'logisti

In [84]:
```
tumedLogReg_search, experimentLog = hyperTunedLogReg(X,y,experimentLog, 5, 3)
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
```

In [85]:
```
print(tumedLogReg_search.best_params_)
```

```
{'logistic__C': 10, 'logistic__tol': 0.0001}
```

In [86]:
```
experimentLog
```

Out[86]:

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Parai
0	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	120.5867	0.0364	Hypertuned LogisticRegression with 4 cv, logis...	{'logis 'logisti
1	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	206.4487	0.0150	Hypertuned LogisticRegression with 5 cv, logis...	{'logis 'logisti (

In [87]:
```
tumedLogReg_search, experimentLog = hyperTunedLogReg(X,y,experimentLog, 10, 1)
```

```
Fitting 10 folds for each of 12 candidates, totalling 120 fits
```

In [88]:
```
print(tumedLogReg_search.best_params_)
```

```
{'logistic__C': 10, 'logistic__tol': 0.001}
```

In [89]:
```
experimentLog
```

Out[89]:

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Parai
0	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	120.5867	0.0364	Hypertuned LogisticRegression with 4 cv, logis...	{'logis 'logisti

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Parai
1	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	206.4487	0.0150	Hypertuned LogisticRegression with 5 cv, logis...	{'logis 'logisti (
2	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	207.5919	0.0150	Hypertuned LogisticRegression with 10 cv, logi...	{'logis 'logisti

In [90]:
```
tumedLogReg_search, experimentLog = hyperTunedLogReg(X,y,experimentLog, 5, 2)
```

```
Fitting 5 folds for each of 18 candidates, totalling 90 fits
```

In [91]:
```
print(tumedLogReg_search.best_params_)
```

```
{'logistic__C': 10, 'logistic__tol': 0.001}
```

In [92]:
```
experimentLog
```

Out[92]:

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Parai
0	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	120.5867	0.0364	Hypertuned LogisticRegression with 4 cv, logis...	{'logis 'logisti
1	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	206.4487	0.0150	Hypertuned LogisticRegression with 5 cv, logis...	{'logis 'logisti (
2	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	207.5919	0.0150	Hypertuned LogisticRegression with 10 cv, logi...	{'logis 'logisti
3	Hypertuned Logisitc rgeression with 74 inputs	91.91%	91.98%	91.96%	0.5	0.287519	91.976152	142.8426	0.0140	Hypertuned LogisticRegression with 5 cv, logis...	{'logis 'logisti

In [68]:
```
experimentLog = pd.DataFrame(columns=["ExpID", "Cross fold train accuracy", "Test Accuracy
```

In [69]:
```
def hyperTunedDecTree(x, y, experimentLog, crossFold, choice):
```

```
    DMT_pipe = Pipeline([
        ('scaler', StandardScaler()),
        ('imputer',SimpleImputer(strategy='median')),
        ('decisionTree', DecisionTreeClassifier(random_state=42))
        ])

    if choice == 1:
        param_grid = [{
        'decisionTree__max_depth': range(1,3),
        'decisionTree__min_samples_leaf': range(1,5),
        'decisionTree__criterion':['gini','entropy'],
        }]
        maxDepth = 'range(1,3)'
        sampleLeaf = 'range(1,5)'
    elif choice == 2:
        param_grid = [{
        'decisionTree__max_depth': range(1,5),
        'decisionTree__min_samples_leaf': range(1,5),
        'decisionTree__criterion':['gini','entropy'],
        }]
        maxDepth = 'range(1,5)'
        sampleLeaf = 'range(1,5)'
    else:
        param_grid = [{
        'decisionTree__max_depth': range(1,10),
        'decisionTree__min_samples_leaf': range(1,5),
        'decisionTree__criterion':['gini','entropy'],
        }]
        maxDepth = 'range(1,10)'
        sampleLeaf = 'range(1,5)'

    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.15, random_state
    x_train, x_valid, y_train, y_valid = train_test_split(x_train, y_train, test_size=0.2,

    # Time and score test predictions
    start = time()

    tumedDecTree_search = GridSearchCV(DMT_pipe, param_grid, cv = crossFold, n_jobs=-1,ver
    tumedDecTree_search.fit(x_train, y_train)

    train_time = np.round(time() - start, 4)

    trainAcc  = tumedDecTree_search.score(x_train, y_train)
    validAcc  = tumedDecTree_search.score(x_valid, y_valid)
    start = time()
    testAcc  = tumedDecTree_search.score(x_test, y_test)
    test_time = np.round(time() - start, 4)
    AUC = roc_auc_score(y_test,tumedDecTree_search.predict(x_test))

    loss = log_loss(y_test,tumedDecTree_search.predict_proba(x_test))
    cnfs_mtrx = confusion_matrix(y_test, tumedDecTree_search.predict(x_test))
    denominator = cnfs_mtrx[0][0] + cnfs_mtrx[0][1] + cnfs_mtrx[1][0] + cnfs_mtrx[1][1]
    accuracy = ((cnfs_mtrx[0][0] + cnfs_mtrx[1][1]) / denominator) * 100

    input_count = x.shape[1]

    temp_df = pd.DataFrame()
    temp_df = temp_df.append(pd.Series(["Hypertuned Decision Tree with {} inputs".format(i
                 AUC, loss, accuracy, train_time, test_time,
                "Hypertuned Decision Tree with {} cv, Max Depth {} and Min Sample Leaf {}
    temp_df.columns = experimentLog.columns

    experimentLog = experimentLog.append(temp_df,ignore_index=True)
    return tumedDecTree_search, experimentLog
```

In [70]:
```
tumedDecTree_search, experimentLog = hyperTunedDecTree(X,y,experimentLog,2,3)
```
```
Fitting 2 folds for each of 72 candidates, totalling 144 fits
```
In [71]:
```
print(tumedDecTree_search.best_params_)
```
```
{'decisionTree__criterion': 'gini', 'decisionTree__max_depth': 8, 'decisionTree__min_sampl
es_leaf': 4}
```
In [97]:
```
experimentLog
```
Out[97]:

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hyp
0	Hypertuned Decision Tree with 74 inputs	92.19%	92.13%	92.03%	0.535858	0.248343	92.130081	71.7969	0.053	Hypertuned Decision Tree with 2 cv, Max Depth ...	{'decision 'g

In [98]:
```
tumedDecTree_search, experimentLog = hyperTunedDecTree(X,y,experimentLog,3,2)
```
```
Fitting 3 folds for each of 32 candidates, totalling 96 fits
```
In [99]:
```
print(tumedDecTree_search.best_params_)
```
```
{'decisionTree__criterion': 'gini', 'decisionTree__max_depth': 1, 'decisionTree__min_sampl
es_leaf': 1}
```
In [100…
```
experimentLog
```
Out[100…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hyp
0	Hypertuned Decision Tree with 74 inputs	92.19%	92.13%	92.03%	0.535858	0.248343	92.130081	71.7969	0.0530	Hypertuned Decision Tree with 2 cv, Max Depth ...	{'decision 'g
1	Hypertuned Decision Tree with 74 inputs	91.91%	91.98%	91.96%	0.500000	0.257044	91.976152	56.4094	0.0515	Hypertuned Decision Tree with 3 cv, Max Depth ...	{'decision 'g

In [101…
```
tumedDecTree_search, experimentLog = hyperTunedDecTree(X,y,experimentLog,5,2)
```
```
Fitting 5 folds for each of 32 candidates, totalling 160 fits
```
In [102…
```
print(tumedDecTree_search.best_params_)
```
```
{'decisionTree__criterion': 'gini', 'decisionTree__max_depth': 1, 'decisionTree__min_sampl
es_leaf': 1}
```

In [103…

```
experimentLog
```

Out[103…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hy
0	Hypertuned Decision Tree with 74 inputs	92.19%	92.13%	92.03%	0.535858	0.248343	92.130081	71.7969	0.0530	Hypertuned Decision Tree with 2 cv, Max Depth ...	{'decisio ',
1	Hypertuned Decision Tree with 74 inputs	91.91%	91.98%	91.96%	0.500000	0.257044	91.976152	56.4094	0.0515	Hypertuned Decision Tree with 3 cv, Max Depth ...	{'decisio ',
2	Hypertuned Decision Tree with 74 inputs	91.91%	91.98%	91.96%	0.500000	0.257044	91.976152	109.6553	0.0590	Hypertuned Decision Tree with 5 cv, Max Depth ...	{'decisio ',

In [104…

```
tumedDecTree_search, experimentLog = hyperTunedDecTree(X,y,experimentLog,5,1)
```

```
Fitting 5 folds for each of 16 candidates, totalling 80 fits
```

In [105…

```
print(tumedDecTree_search.best_params_)
```

```
{'decisionTree__criterion': 'gini', 'decisionTree__max_depth': 1, 'decisionTree__min_sampl
es_leaf': 1}
```

In [106…

```
experimentLog
```

Out[106…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hy
0	Hypertuned Decision Tree with 74 inputs	92.19%	92.13%	92.03%	0.535858	0.248343	92.130081	71.7969	0.0530	Hypertuned Decision Tree with 2 cv, Max Depth ...	{'decisio ',
1	Hypertuned Decision Tree with 74 inputs	91.91%	91.98%	91.96%	0.500000	0.257044	91.976152	56.4094	0.0515	Hypertuned Decision Tree with 3 cv, Max Depth ...	{'decisio ',
2	Hypertuned Decision Tree with 74 inputs	91.91%	91.98%	91.96%	0.500000	0.257044	91.976152	109.6553	0.0590	Hypertuned Decision Tree with 5 cv, Max Depth ...	{'decisio ',

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hy
3	Hypertuned Decision Tree with 74 inputs	91.91%	91.98%	91.96%	0.500000	0.257044	91.976152	55.6498	0.0525	Hypertuned Decision Tree with 5 cv, Max Depth ...	{'decisio '

In [72]:

```
experimentLog = pd.DataFrame(columns=["ExpID", "Cross fold train accuracy", "Test Accuracy
```

In [73]:

```
def hyperTunedRandomForest(x, y, experimentLog, crossFold, choice):

    RFC_pipe = Pipeline([
        ('scaler', StandardScaler()),
        ('imputer',SimpleImputer(strategy='median')),
        ('randomForest', RandomForestClassifier(random_state=42))
        ])

    if choice == 1:
        param_grid = [{
         'randomForest__max_depth': range(1,5),
         'randomForest__min_samples_leaf':range(1,3),
         'randomForest__n_estimators': [50, 100, 150],
         }]
        maxDepth = 'range(1,3)'
        sampleLeaf = 'range(1,3)'
        estimators  = '[50,100,150]'
    elif choice == 2:
        param_grid = [{
         'randomForest__max_depth': range(1,5),
         'randomForest__min_samples_leaf':range(1,5),
         'randomForest__n_estimators': [100,200],
         }]
        maxDepth = 'range(1,5)'
        sampleLeaf = 'range(1,5)'
        estimators  = '[50,200]'
    else:
        param_grid = [{
         'randomForest__max_depth': range(1,3),
         'randomForest__min_samples_leaf':range(1,3),
         'randomForest__n_estimators': [50,100],
         }]
        maxDepth = 'range(1,3)'
        sampleLeaf = 'range(1,3)'
        estimators  = '[50,100]'

    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.15, random_state
    x_train, x_valid, y_train, y_valid = train_test_split(x_train, y_train, test_size=0.2,

    # Time and score test predictions
    start = time()

    tumedRandomForest_search = GridSearchCV(RFC_pipe, param_grid, cv = crossFold, n_jobs=-
    tumedRandomForest_search.fit(x_train, y_train)

    train_time = np.round(time() - start, 4)

    trainAcc  = tumedRandomForest_search.score(x_train, y_train)
    validAcc  = tumedRandomForest_search.score(x_valid, y_valid)
```

```python
    start = time()
    testAcc  = tumedRandomForest_search.score(x_test, y_test)
    test_time = np.round(time() - start, 4)
    AUC = roc_auc_score(y_test,tumedRandomForest_search.predict(x_test))

    loss = log_loss(y_test,tumedRandomForest_search.predict_proba(x_test))
    cnfs_mtrx = confusion_matrix(y_test, tumedRandomForest_search.predict(x_test))
    denominator = cnfs_mtrx[0][0] + cnfs_mtrx[0][1] + cnfs_mtrx[1][0] + cnfs_mtrx[1][1]
    accuracy = ((cnfs_mtrx[0][0] + cnfs_mtrx[1][1]) / denominator) * 100

    input_count = x.shape[1]

    temp_df = pd.DataFrame()
    temp_df = temp_df.append(pd.Series(["Hypertuned random Forest with {} inputs".format(i
              AUC, loss, accuracy, train_time, test_time,
              "Hypertuned Random Forest with {} cv, Max Depth {}, Min Sample Leaf {} a
    temp_df.columns = experimentLog.columns

    experimentLog = experimentLog.append(temp_df,ignore_index=True)
    return tumedRandomForest_search, experimentLog
```

In [74]:

```python
tumedRandomForest_search, experimentLog = hyperTunedRandomForest(X,y,experimentLog, 3, 1)
```

```
Fitting 3 folds for each of 24 candidates, totalling 72 fits
```

In [75]:

```python
print(tumedRandomForest_search.best_params_)
```

```
{'randomForest__max_depth': 1, 'randomForest__min_samples_leaf': 1, 'randomForest__n_estim
ators': 50}
```

In [111…

```python
experimentLog
```

Out[111…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hy
0	Hypertuned random Forest with 74 inputs	91.91%	91.98%	91.96%	0.5	0.269295	91.976152	147.6134	0.125	Hypertuned Random Forest with 3 cv, Max Depth ...	{'randomFore 1, 'r

In [112…

```python
tumedRandomForest_search, experimentLog = hyperTunedRandomForest(X,y,experimentLog, 3, 2)
```

```
Fitting 3 folds for each of 32 candidates, totalling 96 fits
```

In [113…

```python
print(tumedRandomForest_search.best_params_)
```

```
{'randomForest__max_depth': 1, 'randomForest__min_samples_leaf': 1, 'randomForest__n_estim
ators': 100}
```

In [114…

```python
experimentLog
```

Out[114…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hy

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hy
0	Hypertuned random Forest with 74 inputs	91.91%	91.98%	91.96%	0.5	0.269295	91.976152	147.6134	0.125	Hypertuned Random Forest with 3 cv, Max Depth ...	{'randomFore 1, 'r
1	Hypertuned random Forest with 74 inputs	91.91%	91.98%	91.96%	0.5	0.268306	91.976152	262.9420	0.191	Hypertuned Random Forest with 3 cv, Max Depth ...	{'randomFore 1, 'r

In [115…

```
tumedRandomForest_search, experimentLog = hyperTunedRandomForest(X,y,experimentLog, 5, 3)
```

```
Fitting 5 folds for each of 8 candidates, totalling 40 fits
```

In [116…

```
print(tumedRandomForest_search.best_params_)
```

```
{'randomForest__max_depth': 1, 'randomForest__min_samples_leaf': 1, 'randomForest__n_estim
ators': 50}
```

In [117…

```
experimentLog
```

Out[117…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Loss	Accuracy	Train Time(s)	Test Time(s)	Experiment description	Best Hy
0	Hypertuned random Forest with 74 inputs	91.91%	91.98%	91.96%	0.5	0.269295	91.976152	147.6134	0.125	Hypertuned Random Forest with 3 cv, Max Depth ...	{'randomFore 1, 'r
1	Hypertuned random Forest with 74 inputs	91.91%	91.98%	91.96%	0.5	0.268306	91.976152	262.9420	0.191	Hypertuned Random Forest with 3 cv, Max Depth ...	{'randomFore 1, 'r
2	Hypertuned random Forest with 74 inputs	91.91%	91.98%	91.96%	0.5	0.269295	91.976152	68.3533	0.126	Hypertuned Random Forest with 5 cv, Max Depth ...	{'randomFore 1, 'r

In [76]:

```
experimentLog = pd.DataFrame(columns=["ExpID", "Cross fold train accuracy", "Test Accuracy
```

In [77]:

```
def hyperTunedLassoReg(x, y, experimentLog, crossFold, choice):
    lasso_pipeline = Pipeline([
                     ('scaler',StandardScaler()),
                     ('lasso',Lasso())
                   ])

    if choice == 1:
```

```
        lasso_search = GridSearchCV(lasso_pipeline,
                       {'lasso__alpha':[0.0001,0.001,0.01]},
                       cv = crossFold, scoring="neg_mean_squared_error",verbose=3)
        alpha = '[0.0001,0.001,0.01,0.1]'
    elif choice == 2:
        lasso_search = GridSearchCV(lasso_pipeline,
                       {'lasso__alpha':[0.0001,0.001,0.01,0.1]},
                       cv = crossFold, scoring="neg_mean_squared_error",verbose=3)
        alpha = '[0.0001,0.001,0.01,0.1]'
    else:
        lasso_search = GridSearchCV(lasso_pipeline,
                       {'lasso__alpha':[0.0001,0.001,0.01,0.1,1]},
                       cv = crossFold, scoring="neg_mean_squared_error",verbose=3)
        alpha = '[0.0001,0.001,0.01,0.1,1]'

    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.15, random_state
    x_train, x_valid, y_train, y_valid = train_test_split(x_train, y_train, test_size=0.2,

    # Time and score test predictions
    start = time()

    lasso_search.fit(x_train, y_train)

    train_time = np.round(time() - start, 4)

    trainAcc  = lasso_search.score(x_train, y_train)
    validAcc  = lasso_search.score(x_valid, y_valid)
    start = time()
    testAcc  = lasso_search.score(x_test, y_test)
    test_time = np.round(time() - start, 4)
    AUC = roc_auc_score(y_test,lasso_search.predict(x_test))

    input_count = x.shape[1]

    temp_df = pd.DataFrame()
    temp_df = temp_df.append(pd.Series(["Hypertuned Lasso Regression with {} inputs".forma
              AUC, train_time, test_time, "Hypertuned Lasso Regression with {} cv and
    temp_df.columns = experimentLog.columns

    experimentLog = experimentLog.append(temp_df,ignore_index=True)
    return lasso_search, experimentLog
```

In [78]:

```
lasso_search, experimentLog = hyperTunedLassoReg(X,y,experimentLog,5, 1)
```

```
Fitting 5 folds for each of 3 candidates, totalling 15 fits
[CV 1/5] END .............lasso__alpha=0.0001;, score=-0.070 total time=   4.4s
[CV 2/5] END .............lasso__alpha=0.0001;, score=-0.070 total time=   4.5s
[CV 3/5] END .............lasso__alpha=0.0001;, score=-0.067 total time=   4.5s
[CV 4/5] END .............lasso__alpha=0.0001;, score=-0.069 total time=   4.7s
[CV 5/5] END .............lasso__alpha=0.0001;, score=-0.069 total time=   4.6s
[CV 1/5] END ..............lasso__alpha=0.001;, score=-0.070 total time=   0.9s
[CV 2/5] END ..............lasso__alpha=0.001;, score=-0.070 total time=   0.9s
[CV 3/5] END ..............lasso__alpha=0.001;, score=-0.067 total time=   0.9s
[CV 4/5] END ..............lasso__alpha=0.001;, score=-0.070 total time=   1.0s
[CV 5/5] END ..............lasso__alpha=0.001;, score=-0.069 total time=   0.9s
[CV 1/5] END ...............lasso__alpha=0.01;, score=-0.071 total time=   0.4s
[CV 2/5] END ...............lasso__alpha=0.01;, score=-0.071 total time=   0.4s
[CV 3/5] END ...............lasso__alpha=0.01;, score=-0.068 total time=   0.4s
[CV 4/5] END ...............lasso__alpha=0.01;, score=-0.070 total time=   0.4s
[CV 5/5] END ...............lasso__alpha=0.01;, score=-0.070 total time=   0.4s
```

In [121…

```
print(lasso_search.best_params_)
coefficients = lasso_search.best_estimator_.named_steps['lasso'].coef_
```

```
importance = np.abs(coefficients)
len(np.array(features)[importance > 0])
```

```
{'lasso__alpha': 0.0001}
```

Out[121…

```
69
```

In [122…

```
experimentLog
```

Out[122…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
0	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	37.7049	0.026	Hypertuned Lasso Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

In [123…

```
lasso_search, experimentLog = hyperTunedLassoReg(X,y,experimentLog,4, 2)
```

```
Fitting 4 folds for each of 4 candidates, totalling 16 fits
[CV 1/4] END ..............lasso__alpha=0.0001;, score=-0.070 total time=   4.3s
[CV 2/4] END ..............lasso__alpha=0.0001;, score=-0.069 total time=   4.7s
[CV 3/4] END ..............lasso__alpha=0.0001;, score=-0.069 total time=   4.5s
[CV 4/4] END ..............lasso__alpha=0.0001;, score=-0.069 total time=   4.1s
[CV 1/4] END ...............lasso__alpha=0.001;, score=-0.070 total time=   0.9s
[CV 2/4] END ...............lasso__alpha=0.001;, score=-0.069 total time=   0.9s
[CV 3/4] END ...............lasso__alpha=0.001;, score=-0.069 total time=   0.8s
[CV 4/4] END ...............lasso__alpha=0.001;, score=-0.069 total time=   0.8s
[CV 1/4] END ................lasso__alpha=0.01;, score=-0.071 total time=   0.4s
[CV 2/4] END ................lasso__alpha=0.01;, score=-0.069 total time=   0.4s
[CV 3/4] END ................lasso__alpha=0.01;, score=-0.070 total time=   0.4s
[CV 4/4] END ................lasso__alpha=0.01;, score=-0.070 total time=   0.4s
[CV 1/4] END .................lasso__alpha=0.1;, score=-0.076 total time=   0.3s
[CV 2/4] END .................lasso__alpha=0.1;, score=-0.073 total time=   0.3s
[CV 3/4] END .................lasso__alpha=0.1;, score=-0.074 total time=   0.3s
[CV 4/4] END .................lasso__alpha=0.1;, score=-0.074 total time=   0.4s
```

In [124…

```
print(lasso_search.best_params_)
coefficients = lasso_search.best_estimator_.named_steps['lasso'].coef_
importance = np.abs(coefficients)
len(np.array(features)[importance > 0])
```

```
{'lasso__alpha': 0.0001}
```

Out[124…

```
69
```

In [125…

```
experimentLog
```

Out[125…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
0	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	37.7049	0.0260	Hypertuned Lasso Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
1	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	31.7134	0.0262	Hypertuned Lasso Regression with 4 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

In [126…

```
lasso_search, experimentLog = hyperTunedLassoReg(X,y,experimentLog,3, 3)
```

```
Fitting 3 folds for each of 5 candidates, totalling 15 fits
[CV 1/3] END ..............lasso__alpha=0.0001;, score=-0.070 total time=   4.0s
[CV 2/3] END ..............lasso__alpha=0.0001;, score=-0.068 total time=   3.8s
[CV 3/3] END ..............lasso__alpha=0.0001;, score=-0.069 total time=   3.8s
[CV 1/3] END ...............lasso__alpha=0.001;, score=-0.071 total time=   0.7s
[CV 2/3] END ...............lasso__alpha=0.001;, score=-0.068 total time=   0.8s
[CV 3/3] END ...............lasso__alpha=0.001;, score=-0.069 total time=   0.8s
[CV 1/3] END ................lasso__alpha=0.01;, score=-0.071 total time=   0.4s
[CV 2/3] END ................lasso__alpha=0.01;, score=-0.068 total time=   0.3s
[CV 3/3] END ................lasso__alpha=0.01;, score=-0.070 total time=   0.3s
[CV 1/3] END .................lasso__alpha=0.1;, score=-0.076 total time=   0.3s
[CV 2/3] END .................lasso__alpha=0.1;, score=-0.073 total time=   0.3s
[CV 3/3] END .................lasso__alpha=0.1;, score=-0.075 total time=   0.3s
[CV 1/3] END ...................lasso__alpha=1;, score=-0.076 total time=   0.3s
[CV 2/3] END ...................lasso__alpha=1;, score=-0.073 total time=   0.3s
[CV 3/3] END ...................lasso__alpha=1;, score=-0.075 total time=   0.3s
```

In [127…

```
print(lasso_search.best_params_)
coefficients = lasso_search.best_estimator_.named_steps['lasso'].coef_
importance = np.abs(coefficients)
len(np.array(features)[importance > 0])
```

```
{'lasso__alpha': 0.0001}
```

Out[127…

```
69
```

In [128…

```
experimentLog
```

Out[128…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
0	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	37.7049	0.0260	Hypertuned Lasso Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
1	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	31.7134	0.0262	Hypertuned Lasso Regression with 4 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
2	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	24.2078	0.0270	Hypertuned Lasso Regression with 3 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

In [79]:

```
def hyperTunedRidgeReg(x, y, experimentLog, crossFold, choice):

    ridge_pipeline = Pipeline([
                 ('scaler',StandardScaler()),
                 ('ridge',Ridge())
                ])

    if choice == 1:
        ridge_search = GridSearchCV(ridge_pipeline,
                      {'ridge__alpha':[0.0001,0.001,0.01]},
                      cv = crossFold, scoring="neg_mean_squared_error",verbose=3)
        ridge = '[0.0001,0.001,0.01]'
    elif choice == 2:
        ridge_search = GridSearchCV(ridge_pipeline,
                      {'ridge__alpha':[0.0001,0.001,0.01,0.1]},
                      cv = crossFold, scoring="neg_mean_squared_error",verbose=3)
        ridge = '[0.0001,0.001,0.01,0.1]'
    else:
        ridge_search = GridSearchCV(ridge_pipeline,
                      {'ridge__alpha':[0.0001,0.001,0.01,0.1,1]},
                      cv = crossFold, scoring="neg_mean_squared_error",verbose=3)
        ridge = '[0.0001,0.001,0.01,0.1,1]'

    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.15, random_state
    x_train, x_valid, y_train, y_valid = train_test_split(x_train, y_train, test_size=0.2,

    # Time and score test predictions
    start = time()

    ridge_search.fit(x_train, y_train)

    train_time = np.round(time() - start, 4)

    trainAcc  = ridge_search.score(x_train, y_train)
    validAcc  = ridge_search.score(x_valid, y_valid)
    start = time()
    testAcc  = ridge_search.score(x_test, y_test)
    test_time = np.round(time() - start, 4)
    AUC = roc_auc_score(y_test,ridge_search.predict(x_test))


    input_count = x.shape[1]

    temp_df = pd.DataFrame()
    temp_df = temp_df.append(pd.Series(["Hypertuned Ridge Regression with {} inputs".forma
              AUC, train_time, test_time, "Hypertuned Ridge Regression with {} cv and
    temp_df.columns = experimentLog.columns

    experimentLog = experimentLog.append(temp_df,ignore_index=True)
    return ridge_search, experimentLog
```

In [80]:

```
ridge_search, experimentLog = hyperTunedRidgeReg(X,y,experimentLog,5,1)
```

```
Fitting 5 folds for each of 3 candidates, totalling 15 fits
[CV 1/5] END ..............ridge__alpha=0.0001;, score=-0.070 total time=   0.4s
[CV 2/5] END ..............ridge__alpha=0.0001;, score=-0.070 total time=   0.4s
[CV 3/5] END ..............ridge__alpha=0.0001;, score=-0.067 total time=   0.4s
[CV 4/5] END ..............ridge__alpha=0.0001;, score=-0.069 total time=   0.4s
[CV 5/5] END ..............ridge__alpha=0.0001;, score=-0.069 total time=   0.4s
[CV 1/5] END ...............ridge__alpha=0.001;, score=-0.070 total time=   0.4s
[CV 2/5] END ...............ridge__alpha=0.001;, score=-0.070 total time=   0.4s
[CV 3/5] END ...............ridge__alpha=0.001;, score=-0.067 total time=   0.4s
[CV 4/5] END ...............ridge__alpha=0.001;, score=-0.069 total time=   0.4s
[CV 5/5] END ...............ridge__alpha=0.001;, score=-0.069 total time=   0.4s
[CV 1/5] END ................ridge__alpha=0.01;, score=-0.070 total time=   0.4s
[CV 2/5] END ................ridge__alpha=0.01;, score=-0.070 total time=   0.4s
[CV 3/5] END ................ridge__alpha=0.01;, score=-0.067 total time=   0.4s
[CV 4/5] END ................ridge__alpha=0.01;, score=-0.069 total time=   0.4s
[CV 5/5] END ................ridge__alpha=0.01;, score=-0.069 total time=   0.4s
```

In [131…

```
print(ridge_search.best_params_)
coefficients = ridge_search.best_estimator_.named_steps['ridge'].coef_
importance = np.abs(coefficients)
len(np.array(features)[importance > 0])
```

```
{'ridge__alpha': 0.01}
```

Out[131…

```
73
```

In [132…

```
experimentLog
```

Out[132…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
0	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	37.7049	0.0260	Hypertuned Lasso Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
1	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	31.7134	0.0262	Hypertuned Lasso Regression with 4 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
2	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	24.2078	0.0270	Hypertuned Lasso Regression with 3 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
3	Hypertuned Ridge Regression with 74 inputs	-6.89%	-6.87%	-6.89%	0.756875	8.0231	0.0250	Hypertuned Ridge Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

In [133…

```
ridge_search, experimentLog = hyperTunedRidgeReg(X,y,experimentLog,4,2)
```

```
Fitting 4 folds for each of 4 candidates, totalling 16 fits
[CV 1/4] END ..............ridge__alpha=0.0001;, score=-0.070 total time=   0.4s
```

```
[CV 2/4] END ..............ridge__alpha=0.0001;, score=-0.069 total time=   0.4s
[CV 3/4] END ..............ridge__alpha=0.0001;, score=-0.069 total time=   0.4s
[CV 4/4] END ..............ridge__alpha=0.0001;, score=-0.069 total time=   0.4s
[CV 1/4] END ...............ridge__alpha=0.001;, score=-0.070 total time=   0.4s
[CV 2/4] END ...............ridge__alpha=0.001;, score=-0.069 total time=   0.4s
[CV 3/4] END ...............ridge__alpha=0.001;, score=-0.069 total time=   0.4s
[CV 4/4] END ...............ridge__alpha=0.001;, score=-0.069 total time=   0.4s
[CV 1/4] END ................ridge__alpha=0.01;, score=-0.070 total time=   0.3s
[CV 2/4] END ................ridge__alpha=0.01;, score=-0.069 total time=   0.4s
[CV 3/4] END ................ridge__alpha=0.01;, score=-0.069 total time=   0.3s
[CV 4/4] END ................ridge__alpha=0.01;, score=-0.069 total time=   0.4s
[CV 1/4] END .................ridge__alpha=0.1;, score=-0.070 total time=   0.3s
[CV 2/4] END .................ridge__alpha=0.1;, score=-0.069 total time=   0.4s
[CV 3/4] END .................ridge__alpha=0.1;, score=-0.069 total time=   0.3s
[CV 4/4] END .................ridge__alpha=0.1;, score=-0.069 total time=   0.4s
```

In [134…

```
print(ridge_search.best_params_)
coefficients = ridge_search.best_estimator_.named_steps['ridge'].coef_
importance = np.abs(coefficients)
len(np.array(features)[importance > 0])
```

```
{'ridge__alpha': 0.1}
```

Out[134…

```
73
```

In [135…

```
experimentLog
```

Out[135…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
0	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	37.7049	0.0260	Hypertuned Lasso Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
1	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	31.7134	0.0262	Hypertuned Lasso Regression with 4 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
2	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	24.2078	0.0270	Hypertuned Lasso Regression with 3 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
3	Hypertuned Ridge Regression with 74 inputs	-6.89%	-6.87%	-6.89%	0.756875	8.0231	0.0250	Hypertuned Ridge Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
4	Hypertuned Ridge Regression with 74 inputs	-6.89%	-6.87%	-6.89%	0.756874	8.1149	0.0278	Hypertuned Ridge Regression with 4 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

In [136…

```
ridge_search, experimentLog = hyperTunedRidgeReg(X,y,experimentLog,3,3)
```

```
Fitting 3 folds for each of 5 candidates, totalling 15 fits
[CV 1/3] END ..............ridge__alpha=0.0001;, score=-0.070 total time=   0.3s
[CV 2/3] END ..............ridge__alpha=0.0001;, score=-0.068 total time=   0.4s
[CV 3/3] END ..............ridge__alpha=0.0001;, score=-0.069 total time=   0.3s
[CV 1/3] END ...............ridge__alpha=0.001;, score=-0.070 total time=   0.3s
[CV 2/3] END ...............ridge__alpha=0.001;, score=-0.068 total time=   0.3s
[CV 3/3] END ...............ridge__alpha=0.001;, score=-0.069 total time=   0.3s
[CV 1/3] END ................ridge__alpha=0.01;, score=-0.070 total time=   0.3s
[CV 2/3] END ................ridge__alpha=0.01;, score=-0.068 total time=   0.3s
[CV 3/3] END ................ridge__alpha=0.01;, score=-0.069 total time=   0.4s
[CV 1/3] END .................ridge__alpha=0.1;, score=-0.070 total time=   0.3s
[CV 2/3] END .................ridge__alpha=0.1;, score=-0.068 total time=   0.3s
[CV 3/3] END .................ridge__alpha=0.1;, score=-0.069 total time=   0.3s
[CV 1/3] END ...................ridge__alpha=1;, score=-0.070 total time=   0.3s
[CV 2/3] END ...................ridge__alpha=1;, score=-0.068 total time=   0.4s
[CV 3/3] END ...................ridge__alpha=1;, score=-0.069 total time=   0.3s
```

In [137…

```
print(ridge_search.best_params_)
coefficients = ridge_search.best_estimator_.named_steps['ridge'].coef_
importance = np.abs(coefficients)
len(np.array(features)[importance > 0])
```

```
{'ridge__alpha': 1}
73
```

Out[137…

In [138…

```
experimentLog
```

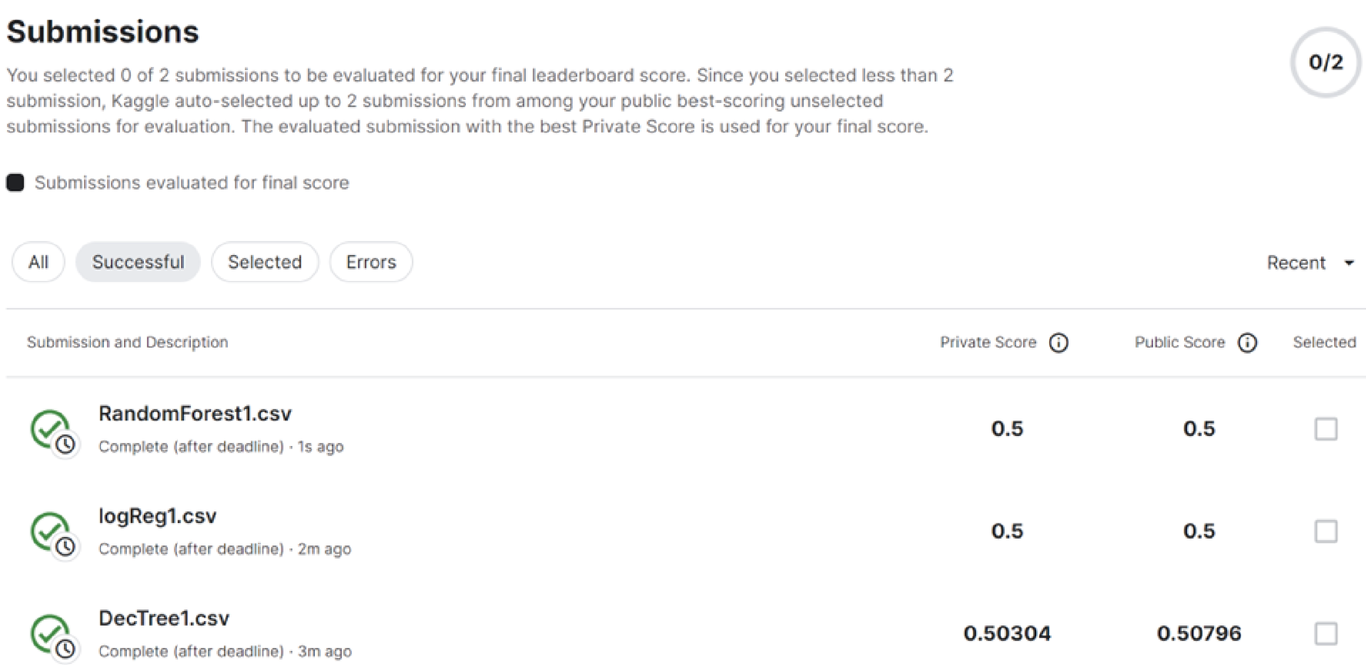
Out[138…

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
0	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	37.7049	0.0260	Hypertuned Lasso Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
1	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	31.7134	0.0262	Hypertuned Lasso Regression with 4 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
2	Hypertuned Lasso Regression with 74 inputs	-6.90%	-6.87%	-6.89%	0.755738	24.2078	0.0270	Hypertuned Lasso Regression with 3 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
3	Hypertuned Ridge Regression with 74 inputs	-6.89%	-6.87%	-6.89%	0.756875	8.0231	0.0250	Hypertuned Ridge Regression with 5 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...
4	Hypertuned Ridge Regression with 74 inputs	-6.89%	-6.87%	-6.89%	0.756874	8.1149	0.0278	Hypertuned Ridge Regression with 4 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

	ExpID	Cross fold train accuracy	Test Accuracy	Valid Accuracy	AUC	Train Time(s)	Test Time(s)	Experiment description	Best Hyper Parameters
5	Hypertuned Ridge Regression with 74 inputs	-6.89%	-6.87%	-6.89%	0.756862	7.0266	0.0270	Hypertuned Ridge Regression with 3 cv and alph...	{'randomForest__max_depth': 1, 'randomForest__...

Result and Discussion

From the experiment log table above, describes the accuracy, AUC, and loss of hyper tuned machine learning model logistic regression, Decision Tree, random forest, lasso regression and ridge regression. For the hyper tuned model decision tree model, we can see that the train (92.19) and test (92.13) accuracy has increased significantly as compared to its baseline model, which means it is performing well on the provided dataset. The log loss for decision tree is on the lower side which is 0.24 and has significantly dropped as compared to its baseline model as well as its AUC is also 0.53. So, the algorithm is performing well for given set of input features. The overall accuracy of decision tree has increased by comparatively large margin and went upto 92 %. Both Random Forest and logistic regression have approximately the same train and test accuracy and log loss as compared to baseline. There is no significant improvement on their hyper tuned parameter model. But hyper tuned Decision Tree remains the best-fit algorithm as it beats others by a very small margin in all the criteria. We observed an increase of 6 percent in test accuracy, and 6 percent in overall accuracy. The log loss for decision tree (0.24) has significantly decreased as compared to its baseline model and is on the lower side and hence it beats the other models. For Lasso and Ridge Regression we observed that AUC has increased to .75. So, both the models seem to predict the target quite correctly as compared to all other models, but, on the other hand accuracy has decreased dramatically. So, even if the models have high AUC, Lasso and Ridge are not the models to look for. They fail to perform appropriately on HCDR dataset.



Conclusion

The HCDR project's goal is to forecast the population's capacity for payback among those who are underserved financially. Because both the lender and the borrower want reliable estimates, this project is crucial. Real-time Home credit's ML pipelines, which acquire data from the data sources via APIs, run EDA, and fit it to the model to generate scores, which allows them to present loan offers to their consumers with the greatest amount and APR. Hence if NPA expected to be less than 5% in order to maintain a profitable firm, risk analysis becomes extremely important. Credit history is an indicator of a user's trustworthiness that is created using parameters such as the average, minimum, and maximum balances that the user maintains, Bureau scores that are reported, salary, etc. Repayment patterns can be analysed using the timely defaults and repayments that the user has made in the past. Other criteria such as location information, social media data, calling/SMS data, etc. are included in alternative data. As part of this project, we would create machine learning pipelines, do exploratory data analysis on the datasets provided by Kaggle, and evaluate the models using a variety of evaluation measures before deploying one. Phase 3 involved the estimation of several models. Data imputation and feature selection were done. We started by selecting features and imputed values. The values of certain features that were missing were filled in. Then, based on our past understanding, we chose to include pertinent features. We trained and assessed several models, including Random Forest, Decision Tree Model, Logistic Regression, Lasso Regression, and Ridge Regression to discover the best one. We hyper tuned them on the best parameters using GridSearch. We have concluded from phase 3 that the Lasso, Ridge and Logistic Regression models is unable to defeat the other hyper parameter tuned models. The decision tree model performs the best out of all the models. In phase 4 we plan to implement MLP using PyTorch

Bibliography

- Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition
    - by Aurélien Géron
- Lab-End_to_end_Machine_Learning_Project
    - by James Shahnan

In [ ]:

```

```